THE DEVELOPMENT OF A FREQUENCY CONTROL SYSTEM OF A SEEDED LASER FOR DGV APPLICATION

A Thesis

by

BRENT NELSON

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

December 2008

Major Subject: Mechanical Engineering

THE DEVELOPMENT OF A FREQUENCY CONTROL SYSTEM

OF A SEEDED LASER FOR DGV APPLICATION

A Thesis

by

BRENT NELSON

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	G. L. Morrison
Committee Members,	D. Banerjee
	J. C. Han
	G. Falcone
Head of Department,	D. O'Neal

December 2008

Major Subject: Mechanical Engineering

ABSTRACT

The Development of a Frequency Control System

of a Seeded Laser for DGV Application. (December 2008)

Brent Nelson, B.S., Rose-Hulman Institute of Technology

Chair of Advisory Committee: Dr. G. L. Morrison

For an effective Doppler Global Velocimeter (DGV), there is a requirement to accurately record and tune the frequency content of the laser used. The laser used for this experiment was an ND:YAG. Adjusting the mean frequency of the ND:YAG is accomplished by controlling the seed laser diode output, which also narrows the bandwidth of the laser to below 20 MHz. The exact frequency of operation is critical for the operation of the system. Standard interferometry techniques that measure laser frequency content, such as Fabre-Perot and grating based systems, are not able to provide an adequate spectrum resolution for the 9 ns pulse duration of the ND:YAG laser. A method was developed that employs a CCD line camera and a laser reference cell to effectively and cost efficiently solve this problem. The hardware and software for this real time monitoring system were developed and used with a real time feedback loop to stabilize the laser operating frequency at a specified value. The receiving optics of this DGV system were upgraded with 12 bit CCD cameras and a temperature controlled laser reference cell to decrease the uncertainty to the velocity measurement from over 4

m/s to less than 1 m/s. Recommendations to the effectiveness of the system and future improvements are provided.

DEDICATION

This work is dedicated to my friends and family who supported me though my time at Texas A&M University and especially to personal friend, Dan Freve whose help and expertise was indispensable.

ACKNOWLEDGEMENTS

I would like to thank Dr. G. L. Morrison for his guidance and support throughout my time at Texas A&M University and during the course of this research. In addition, I would also like to thank Mr. Eddie Denk for his technical assistance. Finally, thanks to my friends and family for their help and support. They have made my time at Texas A&M University unforgettable.

NOMENCLATURE

ALF	Absorption Line Filter
c	Speed of Light, m/s
CCD	Charged Coupled Device
D	Seed Particle Diameter, μ
DGV	Doppler Global Velocimeter
i	Horizontal Pixel Location, Pixels
I_{camera}	Intensity at Each Pixel on Camera
$I_{Doppler}$	Doppler Shift Intensity
j	Vertical Pixel Location, Pixels
LDA	Laser Doppler Anemometry
LDV	Laser Doppler Velocimetry
$\hat{l}$	Laser Beam Direction Unit Vector
M	Mach Number
$m_{L,a,b}$	Measurement in Lumen Levels in: a: x or y direction b: ALF or REF camera
ND	Neutral Density Filter
$\hat{o}$	Observer or Receiving Optics Direction Unit Vector
OD	Optical Density
PIV	Particle Image Velocimetry

REF	Reference Camera
T	Transitivity
T_f	Transfer Function
$\vec{V}$	Velocity Vector of Moving Particle
v_0	Laser Light Frequency, Hz
Δv	Doppler Shift of Light Frequency, Hz
x	Vertical Location After Spatial Calibration, mm
Δx	Amount of Misalignment Previously Determined
y	Horizontal Location After Spatial Calibration, mm
$\frac{\partial L}{\partial x}$	Local Measured Lumen Gradient
θ	Scattering Angle, deg
ω_L	Uncertainty in Lumen Levels
$\omega_{L,a,b}$	Uncertainty in Lumen Levels in: a: x or y direction b: ALF or REF camera
$\omega_{L,misalign}$	Uncertainty Due to Misalignment

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

Page

LIST OF TABLES

1. INTRODUCTION

Since the inception of fluid dynamics there has been a desire to measure the velocity distribution of a fluid flow to understand the fundamental physics of said flow. The first methods involved inserting a physical object (or probe) into the flow. These devices include Pitot and Pitot-static probes, along with hot-wire and hot-film anemometry. These methods are able to very cost effectively measure the velocity of the flow with the hot-wire being able to provide a relatively high frequency response. By using three of these probes in different orientations at close to the same point in the flow a full three-dimensional velocity vector can be resolved. The primary problems with a probe method of measurement are that the physical presence of the probe itself can disturb the flow and thus reduce the measurement accuracy and that the measurement is at a single point in the flow. Other nonintrusive forms of anemometry have been able to provide a solution to the physical disturbance problem.

The majorly researched and developed nonintrusive laser based forms of anemometry include Laser Doppler Velocimetry (LDV), Particle Image Velocimetry (PIV), Michelson interferometry, and Doppler Global Velocimetry (DGV). Developed and patented by Komine [1], DGV has the advantage over the other methods of being able measure unsteady flow fields and is close to being able to supply this data in real time. By illuminating a plane with a sheet of laser light, DGV is able to simultaneously

---

This thesis follows the style of *Trans. of ASME Journal of Fluid Engineering.*

measure the global three-dimensional velocity of a seeded flow field. This is done by measuring the Doppler shift in the light reflected from small seed particles in the flow.

LDV uses a method of converging two laser beams onto a desired point and measuring the frequency of the difference between the Doppler shifts of light reflected by seed particles from the two beams. This allows measurements of one velocity component. More beams can be added for additional velocity components. As a point by point method, it is not time efficient to highly resolve an area in a flow field, therefore LDV is relegated to steady state or periodic flows where measurements can be taken over a given time frame while the flow remains relatively unchanged. This limits the system's ability to measure turbulence spatial structure.

Of the global techniques, PIV has the simplest concept; Take two pictures a known time apart and measure the distance a particle has traveled. PIV is able to measure the two-dimensional flow by capturing particles caught in a sheet of laser light produced by a double (or more) pulsed laser with a photographic or digital camera. This simple concept has made PIV the most common of the global techniques. The major problems with PIV are the accuracy and resolution. The inaccuracies arise in more complicated flow fields where turbulence may occur or particles may move perpendicular to the light sheet. The complex mathematical algorithms used to extract the velocity information have difficulties accounting for these movements when seed particles do not remain in the field of view. Because PIV measures the distance a particle moves, the spatial analysis region must contain the particles within one grid interrogation area, which is composed of many pixels, whereas DGV has the ability to

measure the velocity at each of the pixel locations of the camera. This requirement of the grid measurement area to be larger than one pixel of the camera for PIV reduces the spatial resolution. Three dimensional velocity information can be obtained by using two cameras at different angles and thickening the laser sheet so particles remain in the sheet.

Similar to DGV, Michelson interferometry has the capability of producing a three-dimensional Doppler image of a global flow field. Fringe patters is the physics phenomena that is used to measure the constant phase shift of the Doppler shifted light with this system. The major draw backs to Michelson interferometry are the complicated arrangement of the optic and the complexities of processing the fringe patterns.

The DGV system produced has positive aspects over all the systems covered above. This global system uses a laser of narrow frequency and a notch filter to measure the Doppler shift across a flow field. The major drawback to the DGV is that it is relegated to relatively high flow speeds due to small Doppler shifts being difficult to resolve. The objectives of this study were to expand upon Morrison et al. [2] and develop a control system for a ND:YAG pulsed laser system. This system includes an innovative method to ND:YAG frequency control, an improvement to previous CCD camera bit resolution, and software for data collection and post processing. These improvements should effectively lower extend the limit of measureable flow speeds for a DGV system. The development of this system is desired to be able to resolve the instantaneous velocities and turbulence of flow fields in complex flow geometries previously unmeasured.

The development of hardware and software to monitor and control the frequency of the ND:YAG was achieved and implemented. Once a system alignment and intensity calibration were performed a velocity calibration using a rotating disk was attempted. This indicated the bandwidth of the laser light was too large which lead to inconclusive data. A program to determine the spatial distribution of the spectral content along a profile of the laser beam using data from the frequency control system was developed to further investigate the problem.

2. PRINCIPLE OF OPERATION

The LDV system developed in 1964 by Yeh and Cummins [3] provide the blueprint for the optical arrangement used in DGV today, Figure 1. The laser light reflected by the moving fluid produces a Doppler shift. The goal is to measure the Doppler shifted frequency of the light and relate it back to the fluid velocity. An approach differing from the LDV system was developed and patented by Komine [1] using a frequency to intensity converting optical filter. This special optical filter, or absorption line filter (ALF), contains molecules which absorb light near certain laser frequencies. An ND:YAG laser operates at a frequency of 5.6×10^{14} Hz (532nm). The absorption properties of molecular iodine vapor have a notch that is adequately close to the frequency of this laser and suits well as an ALF filter. The properties of an ALF are very similar to that of an electrical notch filter and are represented in Figure 2.

For the principle used by Komine to work, the laser frequency must be stable, monochromatic, and have a frequency bandwidth much narrower than the width of the absorption profile. In Figure 2, the laser frequency v_0 is tuned to a point midway along the high frequency side of the absorption line profile. For this setting, 50 % of the light scattered from a stationary object is transmitted though the cell. If the object or particle is moving, Doppler shifted light frequencies that are higher than v_0, result in increased transmission through the cell indicating a positive velocity. A decrease in transmission would indicate a negative velocity. By tuning the laser to the lower frequency side of the absorption profile, the relationship of transmission to frequency will be reversed.

This change in transmission is proportional to the Doppler frequency shift of the scattered light as produced by the particle's motion along the measurement vector show in Figure 1. The governing equation is given by:

$$\Delta v = v_0(\hat{o} - \hat{l}) \bullet \frac{\vec{V}}{c} \qquad (1)$$

Where Δv is the Doppler frequency shift, $\hat{o}$ and $\hat{l}$ are the unit vectors in the scattering and laser light propagation directions respectively. $\vec{V}$ is the velocity vector of the moving particle or object, v_0 is the laser light frequency, and c is the speed of light. For a given optical geometry with one transmitting ($\hat{l}$) and one receiving ($\hat{o}$) component, the DGV system measures a single velocity component. Because equation (1) is vectorial, it is possible to assemble a three dimension system by using three different observation directions ($\hat{o}$) or three different laser beams oriented in three different directions ($\hat{l}$).

The reason that a DGV system is able to be a global device is due to the ability to make a relatively large diameter ALF cell. LDV is relegated to a single point device because that method analyzes scattered light from a single particle. Rather, with DGV, the laser beam is expanded into a sheet of light with numerous particles providing Doppler shifts. This light intensity is collected by a camera rather than a point detector, in the case of the LDV, to complete the transformation to a global velocity measurement system. The images captured by the cameras are an instantaneous global snapshot of intensity levels at each pixel, whose relationship is a function of velocity via equation (1) and the relation:

$$I_{camera}(x, y, z) = I_{Doppler}(x, y, z, \Delta v) \bullet T_f(\Delta v) \qquad (2)$$

Where $I_{Doppler}$ is the intensity due to the Doppler shifted light reflected from the flow, T_f is the transfer function of the ALF cell, and I_{camera} is the resulting intensity recorded by the camera at each pixel. The orientation of the vectors from equation (1) and one arrangement of a single camera that records a single velocity component is shown in Figure 3.

When using a camera to perform whole field measurements, the seed density, seed size and laser light sheet illumination are not necessarily uniform. This results in intensity variations not associated with the optical attenuations caused by the iodine vapor induced by the Doppler shift in frequency. To compensate for these effects, a reference image is also recorded to provide the reflected light intensity before filtered by the ALF cell. The image is separated from the primary optical train by placing a transfer lens and beam splitter ahead of the ALF cell, this image is recorded simultaneously with the image filtered by the ALF cell. By means of this reference image (REF), the image transmitted through the ALF cell can be normalized to yield an intensity map which corresponds exclusively to the velocity caused variations in the illuminated flow field. Figure 4 illustrates a typical optical arrangement of lenses and cameras necessary to accomplish this process.

It is also possible to tune the laser's operating frequency using the properties of the ALF cell. The laser frequency control system is designed to maintain the laser at a constant frequency, thus maintaining a desired transmission rate though the ALF cell. A low power beam is separated from the main laser beam and split into two beams.

Comparing the intensity of the laser light that goes though the ALF cell to that of the reference intensity results in a measure of the laser's operating frequency, optical schematic shown in Figure 5. The light intensity of both beams are measured using a CCD line camera. This produces a radial profile of each beam's intensity, shown in Figure 6. The power in each beam is obtained by integrating each beam's profile. The ratio of these two values is the measured ratio which is then compared to a desired ratio. The voltage to the laser seed diode, which controls the frequency at which the laser operates, is varied to amend the difference, thus maintaining the laser at a frequency that yields the desired ratio. The optical train for this control system is similar to that of Figure 4; with the beam splitter and the ALF cell, the actual laser beam is traveling thought the train in this scheme to a single CCD line camera, capturing both the ALF and REF beams, shown in Figure 6.

3. LITERATURE REVIEW

The idea of using an iodine filled optical cell as a frequency dependent optical filter was patented by H. Komine working at the Northrop Corporation in 1990. He called it a Doppler Global Velocimeter, or DGV. This patent is the basis for several papers with topics on filtered particle scattering (FPS). The initial setup, that many subsequent systems have been based upon, contains a frequency controlled laser projected into a sheet that intersects a seeded flow with the receiving optics gathering the scattered light. The receiving optics train first splits the incoming scattered light into two equal intensity images. One passes through a molecular iodine cell to a CCD array, and the other is directly recorded by a second CCD array. Early systems were based on this initial setup, subsequent papers modified different components to gain more accurate results and evolve into the systems used today.

Komine and Brosnan [4] and Komine, et al. [5], in proof of concept papers, expanded on the idea of the original patent by using both a continuous wave (CW) and a pulsed laser for illumination, and video frame grabbers for image acquisition. According to Meyers and Komine [6], there are three main responsibilities of the signal processing scheme: to synchronize the reference and signal cameras, to overlay and normalize the images, and to correct for varying pixel sensitivities across the CCD array. This paper also noted that for real-time data acquisition, it is necessary to divide these two images with an analog divider, then sample or record the resultant image. A much more

accurate, albeit significantly slower, method involves simultaneously sampling each raw image, then digitally dividing pixel by pixel to produce the normalized image.

DGV data was taken by Meyers, et al. [7] on a rotating wheel, a subsonic jet, and a 75º delta wing in a wind tunnel. Qualitatively, the results were as they should be, and agreed with known velocities. However, there was still no direct quantitative comparison with previous or concurrently measured data. Their results from these experiments using both the analog and digital approach to the signal processing are presented in more detail in a follow-up paper by Meyers, Lee, and Cavone [7]. As work on this system progressed, Meyers [8,9] added more signal processing steps to the data reduction scheme for the images acquired by the video cameras. One realization arising from the wind tunnel experiments was the need for exact pixel overlay, which could not be achieved by simply physical alignment the cameras. After warping algorithms were applied, spatial cross correlation routines were used to correct any remaining misalignment during data taking by identifying the proper pixel position at the peak value of the correlation. The pixels were then shifted to that position, providing maximum pixel-to-pixel correlation for the entire image.

In an application oriented paper (Gorton, et al., [10]), Meyers used the DGV system in place at NASA Langley to measure aerodynamic rotor-tail-fuselage interaction on a small scale helicopter model. The results were compared with LDV measurements taken in the same locations, and significant errors in the DGV data were apparent. The main source of these errors was found to be large temperature fluctuations in each of the iodine cells used for frequency discrimination. They recommended future system should

include sealed iodine cells, which contain no iodine crystals, only vapor. With no crystals present, the vapor pressure, and therefore the absorption characteristics of the cell are much less likely to vary with small changes in temperature.

Ford and Tatam [11] pioneered the use of digital image acquisition by using CCD cameras with purely digital signals with their single component system roughly based on the patent by Komine. This was contrary to the digital to analog systems previously used. Calculations showed that the error due to angular variation across the field of view is on the order of the resolution of the system when the angle reaches 5 degrees on either side of center, and 10 degrees of incident beam angle.

Smith and Northam [12] and Smith et al. [13] present a system utilizing only one CCD camera in the DGV image acquisition system. Smith and Northam illuminate a seeded flow, use a frequency controlled pulsed laser to produce the light sheet, and a single camera to image both signal and reference images. The single camera feature allows the use of higher quality, more expensive cameras, or allows more velocity components to be implemented since cameras are a major cost item. Since both images are placed side by side on the same CCD array, errors due to manufacturing inconsistencies should be less than they would be between two different cameras. Smith and Northam (1995) also derived an equation for determining the uncertainty that laser speckle adds to a measurement image. The effect of speckle on a measurement is a function of several factors including a direct proportionality with the magnification of the image and the f-number of the lens.

In a subsequent paper, Smith [14] investigated a compressible jet with a single component, pulsed laser DGV system. The flow had a novel core/co-flow seeding apparatus, which enabled velocity measurements throughout the mixing layer of the jet. In a separate but related effort, Smith [15] has researched the problem of reducing speckle noise for DGV systems utilizing pulsed lasers, and applied his findings in this paper. The primary noise reduction mechanism found was to influence the size and concentration of speckle "dots" on the CCD array. One of the most effective methods was to decrease the f-number and the signal to noise ratio of speckle induced images.

McKenzie [16] also used a pulsed laser and a single camera for each velocity component in his setup. In this paper, he showed that it was desirable to use one high quality camera than to use two average quality cameras. Also, it was shown that the scientific grade, cooled, slow-scan CCD arrays had less noise than many other types of photo sensitive devices, including photo multiplier tubes and photo diodes. The importance of frequency monitoring of the pulsed laser was investigated. It was found that laser drift contributes both random, pulse to pulse noise, and systematic, long-term error. Finally, McKenzie examined in depth the effects of temperature and vapor pressure on the iodine cell frequency filter and its many transmission lines. A stem was added to the iodine cell, the significance to the stem temperature on the depth and slope of a transmission line within the tuning range of the pulsed laser is reported to show its affect on broadening and steepening of the slope. In addition, if the transmission line is made deep enough to block all light of a certain frequency, the cell can effectively function as a notch filter. A thorough error analysis followed, which concluded that the

CCD camera array is the limiting error source. Even though the camera is a scientific grade, cooled, slow-scan, 12-bit camera, all other sources of error have been minimized to a point below the accuracy of the camera.

The most recent effort from McKenzie [17,18]focused on a pulsed laser, planar system. Unique among his methods of reducing these errors is a 3x3 binning scheme whereby each pixel value in a 3x3 grid is replaced by the sum of all pixel values in the grid. This reduces many of the sampling and mapping errors associated with acquiring and overlaying two images read from a CCD array but reduces spatial resolution. A discussion of laser speckle was also presented, and an analysis of speckle noise contained in images collected at different focal lengths and apertures concludes that speckle noise is reduced in direct inverse proportion to the f-number of the receiving optics, as Smith had previously concluded.

Thorpe, et al. [19] investigated the free jet further by time-averaging the acquired images and carefully documenting the accuracy of their system. Time averaging was accomplished by opening the shutter of the camera for extended periods of time. In this manner, average velocity fields were attained by processing only one image.

An early paper by Chan, et al. [20] is one of the first to explicate the advantages of a single camera, split image DGV system. Simplified electronics, lower cost, and higher quality images are all potential benefits of adopting a velocity system configuration containing a single camera. To improve the pixel-to-pixel image alignment in the near field, the authors refined the procedure used to adjust the image position on the CCD array by viewing a speckle image formed by scattered laser light. For

suppression of the high frequency noise, which was superimposed on the linear ratio measurements, two different digital filters were tested. The first filter was a 3x3 spatial averaging filter, which removed some of the high frequency content from the data, but introduced a bias in the trend of the variation. The second filter was a 3x3 median filter, where the center pixel in the array was replaced by the median of the pixel values of all nine pixels. The median filter was less effective at reducing the noise level, and also induced a bias in the data.

Work performed by Reeder [21] extended the split image concept to a two-component, one iodine filter, and one camera system with four separate images captured on one CCD array. While this obviously reduced the spatial resolution for the data images, data taken for a supersonic jet showed velocities to within 10% of both PIV and Pitot probe data, showing promise for the possibility of research of this measurement technique on a much smaller budget.

Researchers from Texas A&M University (Morrison, Gaharan, and DeOtte, [22]) developed a one-component DGV system based on the setup used by Meyers and developed by Komine. Their paper centered on the difficulties of setting up and obtaining accurate measurements from such a system. Problems included difficulty in setting the gains on the cameras purchased, inoperative or damaged CCD sensor elements, and non-uniform response to light intensity. Optics problems involving lenses, beam splitters and polarization effects were also discussed. By directly calibrating the velocity relative to light intensity, Morrison's group chose a much less complex method of resolving velocity data. In this calibration light intensities are related to 'lumen' levels

in an intensity calibration, and then intensity is related to a velocity constant in a velocity calibration. Using this type of calibration assumes that many variables not included in the calibration will remain constant. Consequently, the results of an experiment measuring the centerline velocity of an axisymmetric jet show wide scattering (±20%) on an instantaneous basis when compared with LDV data. The data compared much better when the results were averaged.

Roehle [23] used a three component system to measure flow exiting a swirl nozzle, as well as the flow field behind a scale model of an automobile. Careful alignment, using micro positioning equipment, of the viewing angle of each camera precluded the use of any dewarping software to obtain corresponding pixel registration. The author plans to include such a dewarping scheme in future work. As in the point measurement system discussed earlier by the same author, active control of the laser frequency was achieved to within 1MHz of the set point. Control of the laser frequency of that precision nullifies the need for a frequency monitoring and compensation system. Roehle emphasizes the comparatively short time required to acquire and process a DGV measurement as the greatest advantage of the method.

Three-dimensional DGV measurements are being carried out by Muller et al. [24] in Germany. The system utilizes three coplanar light sheets that intersect from different angles perpendicular to a pipe flow. The image is then reflected off a mirror downstream of the pipe out to receiving optics. It was found that the deviations in the first measurements were relatively high, therefore for a DGV system to measure pipe flow the absolute accuracy of the DGV-setup must be increased.

Nobes, Ford and Tatum [25] used fiber bundles to obtain 3 different viewing angles to produce a three-dimensional DGV image. With the use of fiber bundles they are able to conduct three different observer angels to a single DGV camera set up. The CCD camera then recorded four split images. This idea is similar to the two images of the ALF and REF combinations on a single CCD camera. A proof of concept has been developed and the calibration data presented.

A comparison of the two previous three-dimensional systems was performed by Willert et al. [26], also comparing the data to LDA measurements as a benchmark. The agreement between the various measurement techniques is within 2 m/s, with DGV slightly underestimating local velocity gradients. Both DGV systems were able to present comparable sensitivity and geometry, it is noted that the fiber bundle system was able to accomplish this in one third the acquisitions time.

Two pulsed lasers operating at different wavelengths are used in a two-color approach to DGV in Arnette, et al. [27]. One laser emits green light at 532 nm, and the other, red light at 618 nm. Both beams are spread into a co-planar sheet and the scattered light is passed through an iodine cell and imaged on a color camera. The green laser light is tuned so that it is attenuated, by the iodine cell, in direct proportion with the frequency shift, as is the norm with DGV systems. The red light intensity in the images has no frequency dependence. Velocity measurements taken in a compressible jet flow showed potential for this type of system, although the increased cost of an additional laser may outweigh the benefit of needing one less camera or having a simpler data acquisition and reduction algorithm, making this novel approach cost prohibitive. Another concern was

that the seed particle size needed to be increased beyond the Rayleigh limit and the ratio of the red and green scattering signals will depend on the particle size.

Willert et al [28] utilize the combination of two measurement systems rather than two different lasers frequencies to provide three dimensional DGV system. With the combination of PIV and DGV a three dimensional flow was resolved within a combustor can. Three different laser sheets were used and stereoscopic measurements were taken to provide the volume. Within each sheet measurement the PIV system is able to provide the in plane velocities, while the DGV system recovers the projections of the velocity vector onto the difference vector between the light-sheet and the observation vectors. As the DGV-component is not co-planar with the plane spanned by the two PIV components, the three (orthogonal) velocity components can be determined by a straight forward matrix inversion.

An alternative to the iodine cell as a frequency discriminator is presented by Bloom, et al. [29] in their setup for a long range Doppler Velocimeter. The scattered light is focused through a polarizing beam splitter, and then each half is passed through a frequency discriminator. The filters are filled with atomic cesium, which when heated and placed within a magnetic field, rotate the polarization by an amount proportional to the frequency of the light passing through it, the strength of the magnetic field, and the temperature. The technical details of the cesium cell line filtering method are outlined in a prior paper by Menders, et al. [30].

Another system that uses cesium as a filter is a frequency modulation DGV designed by Fischer et al. [31]. A signal with 2 kHz modulation, 400 MHz modulation

amplitude, and -50 Mhz relative center frequency is applied to a flow in a similar manner as other DGV systems. This is very similar to lasers used in other DGV applications, but it is desirable for this laser to periodically change frequency. The abortion cell is then able to allow only a select spectrum of Doppler shifted light to pass. This system is able to obtain data without the use of a reference camera, eliminating many sources of error. Measurements on a rotating disk showed a minimum standard deviation of about 0.02 m/s. Measurements were also conducted on a low speed flow where accurate measurements were also possible.

Coupland [32] devised a DGV system that does have some capabilities at lower flows, as well. This system employs a rotating mirror that reflects multiple laser beams into the view of a single CCD camera that captures 'smears.' These smears are the image of the laser light as the mirror reflects it to the CCD camera with a reference image overlaid from fiber optics. The data presented shows the effectiveness at slower flows, but further investigation would have to be done to determine the effectiveness at higher speeds.

Normally, when molecular iodine is used as the frequency discriminator, a CW argon ion or pulsed ND:YAG laser is used for the interrogating laser beam. Research performed by Leporcq, et al. [33], focused on using a narrow bandwidth, tunable dye laser. This type of laser has a line width of approximately 500 kHz, compared to a 12 MHz line width for a CW argon ion laser. An even more attractive feature is the ability of the dye laser to be tuned over a much larger frequency range, allowing the selection of an iodine absorption line to complement the experimental setup and expected velocity

range. Rotating wheel data was presented to show the validity of the use of this type of laser.

In more recent years, Chan et al. [34], Chen et al. [35], and Lazar et al. [36] have independently research into iodine cell technology to improve accuracy of their DGV systems. Chan et al. determined seven primary factors that affect ALF cell performance: purity of the iodine, maintaining pure iodine during degassing and filling of the ALF cell, cell materials, method of construction, control of cell performance (use of buffer gasses), relationship between dimensions and operating conditions for an ALF, end windows, and temperature control. Using their guideline an optimum cell can be designed for the user's system. Chen et al. was able to discover several groups of stronger absorption lines of iodine that were observed with high signal to noise ratios which would be available to be reference lines for diode laser frequency standards. The common line used is at wavelength 632.99 (nm). Five notable wavelengths that were discovered are: 632.962, 632.967, 632.985, 632.987 and 633.009 (nm). Lazar et al. were able to present technology of iodine cell preparations and arrangement for the fluorescent purity investigation of the iodine following the manufacturing process, iodine purification and filling. This will produce a more accurate iodine cell, thus a more accurate DGV system.

All of the research discussed previously used particles to generate scattered light, either by seeding the flow with various aerosols or relying on the flow's own contaminants. The laser light is scattered by the air molecules themselves in a process called Filtered Rayleigh Scattering (FRS), which uses no seed particles. Miles, Forkey,

and Lempert [37] incorporate this method of scattering in their FRS Velocimeter. In order to collect enough scattered light on the CCD arrays to analyze, the laser power must be very high. Currently, pulsed lasers are the only practical choice when very high power is needed. When enough scattered signal is achieved, that signal can be interpreted as follows: the intensity of the scattered light is proportional to the density of the air, the line width is proportional to temperature (so called temperature broadening), and the frequency shift is proportional to velocity.

In a follow up paper, Forkey, Finkelstein, Lempert, and Miles [38] analyzed the uncertainty of an FRS system, and performed measurements of stationary room air and a Mach 2 jet. Temperature stabilization of the iodine cell, found to be critical in previous work, was also employed here. Total uncertainty was on the order of 5 m/s, and largely due to background scatter and laser drift. Elliott, et al. [39] are also researching an FRS instrument. Their system incorporates many of the same techniques and hardware as Miles' FRS system. They discuss the possibility of reducing the ± 8% error to about ± 3% when accounting for the laser drift by measuring it on a frame-by-frame basis as other researchers have done.

4. OBJECTIVES

The primary objectives are to increase the accuracy of a one dimensional DGV system, developed by Gaharan [40], and to implement the use of an ND:YAG pulsed laser to replace the Argon-Ion laser previously used.

5. APPROACH

This study will utilize the basic system outlined in Morrison et al. [2] and develop an enhanced DGV system. This is achieved thought the use of a different lazing system and the upgrade of optics, cameras, and software. The primary change came from the lasing system. The previous laser used was an argon-ion laser. This type of laser produces a continuous beam; therefore to capture an instant in time, the exposure time of the cameras must be adjusted to be adequately short enough to capture any flow fluctuations. With the move to an ND:YAG laser, a pulsed laser, the exposure time of the cameras were lengthened to ensure the capture of just one of the 9 ns pulses of the laser. The power output of the ND:YAG is also superior to the argon-ion laser. This can increase the accuracy of the readings by having more accurate intensity calibrations and larger intensity changes over full scale velocity readings.

An Argon-Ion laser in stable enough to control its frequency manually using a temperature stabilized etalon and presented little drift. An initial study of the ND:YAG checked the stability and drift characteristics of the laser. The frequency was checked at one minute time intervals over a duration of approximately 15 minutes to analyze the drift characteristics. To check the stability, the frequency was monitored for a 10 Hz pulse rate for a minute. Both diagnostics showed the need for automated control and an active monitoring system. Because of the 9 ns pulse duration of the ND:YAG, special considerations had to be made for the monitoring system. Standard interferometry techniques could not handle the spectrum resolution required and commercially

available systems carried exorbitant costs. The development of a method that employs a CCD line camera was created to effectively solve this problem. The hardware and software of this real time monitoring system are capable of controlling the voltage to the seeder of the ND:YAG, effectively eliminating the drift while also capturing the information needed to account for stability.

The transmitting and receiving optics were upgraded in this setup to increase accuracy. A Telecentric lens was used in the DGV receiving optics. Telecentric lenses correct perspective errors that yield variations in magnification through the depth of field. This reduces inaccuracies called edge effects. Edge effects occur because of the convex shape of a standard telephoto lens and produce a perspective error. The telecentric lens conducted the image to true 12-bit Dalsa brand CCD cameras. This 12-bit capability can dramatically increase the accuracy over the 8-bit cameras previously used by increasing the gray scale resolution of the intensity reading from 256 to 4096 increments. The spatial resolution of the camera (number of pixels) was also increased to 1024 by 1024 pixels from 512 by 256. This allows for either a view of a larger area with the same spatial accuracy as before or increased accuracy in the same area viewed as the lower spatial resolution CCD camera. Increases in spatial and grayscale resolution are the primary modes to increase accuracy and lower the velocities at which a DGV system can function, according to C. Gaharan's PhD dissertation [40].

With the hardware assembled and the need for a controller identified, compiling of the software began. LabVIEW is the primary method of coding for this system. Call Library Function Nodes were used to call a C++ wrapper function that called the SDK

function of the CCD line camera that was used to control the laser. The Dalsa CCD cameras (used for DGV) were able to be controlled through the IMAQ package of LabVIEW. Both sub systems of camera control were combined into one LabVIEW program to record the DGV images and laser information simultaneously.

Utilizing this hardware and software arrangement for an effective DGV, calibrations were conducted. First, a spatial calibration was performed to account for any misalignments in the transmitting optics and the Dalsa CCD cameras. These misalignments may occur in production of the receiving optics, differences in the mounting brackets in the cameras, slight misalignment in the beam splitter, or a difference in the telephoto lenses on each of the receiving optics of the Dalsa CCD cameras. A spatial calibration can correct all these abnormalities so that each pixel, being compared in both cameras, refers to the same reference point in space. An Intensity calibration was conducted from zero light to the maximum intensity on the Dalsa CCD cameras. This is done for a conversion from pixel intensity to “lumens”. The final calibration is to compare these lumens to an object of known velocity. A spinning disk was used to conduct a velocity calibration.

At this point it was identified that the laser was not operating as anticipated. The final task was to take measurement of a 30 m/s seeded flow to compare with previous findings. Because the laser was not functioning properly new tasks were identified. A nondimensionalized autocorrelation of the laser spectrum across the laser beam was conducted and it was found that the laser was not as coherent as needed to conduct DGV readings. It was identified that a LabVIEW program to autocorrelate the readings from

the CCD line camera can be used to measure and monitor a laser beam's spatial spectral content. This system was incorporated into the LabVIEW DGV control/acquisition program developed in this research.

6. EXPERIMENTAL APPARATUS

This section covers an overview of the DGV components used to test the proof of concept for the advancements made to the findings presented in the C. Gaharan's PhD dissertation [40], the developer of the first DGV system at Texas A&M University. This system follows the initial findings of the DGV first introduced by Komine [1], the inventor and patent holder of the DGV system. Upgrades have been made to each of the four major subsystems of Gaharan's initial device; light source, frequency control unit, Doppler image analyzer unit, and Data acquisition method. The calibration method has remained relatively the same with a minor change in the intensity calibration made available with an increase in computing power.

6.1 Light Source

Previously a continuous-wave, Argon-Ion laser was used and a moment was captured in time by allowing the cameras to have a short exposure time. This was not the case with the ND:YAG used with this system. The ND:YAG has such a short pulse duration, 9 ns, that the only requirement was that the cameras shutters had to be open when the ND:YAG lased. The short pulse duration effectively freezes the turbulence patterns in the flow field. A Spectra-Physics Quanta Ray Lab 130 was selected, whose specifications include a 9 ns pulse duration, 0.003 cm^{-1} (15 MHz) bandwidth, and frequency tunable using a seed diode.

Both the Argon-Ion and ND:YAG operate at frequencies that match a notch in the abortion line of iodine, $5.83x10^{14}$ Hz (514.5 nm) and $5.63x10^{14}$ Hz (532 nm) respectively. The ND:YAG was seeded by a Lightwave laser Diode (model 6350), that effectively reduces the bandwidth of the laser to 15 MHz, which is much smaller than the 2 GHz width of the molecular iodine line. This bandwidth is very similar to the Argon-Ion laser which used an oven stabilized etalon to narrow and stabilize the output frequency. The oven temperature was used to adjust the frequency of the Argon-Ion laser. The ND:YAG laser's frequency was controlled by the diode seed laser. A DC voltage supplied to the laser changes the frequency as the voltage is varied.

The output beam of the ND:YAG laser is approximately 1 cm in diameter, conducting a maximum energy of 160 mJ per pulse as 532 nm at pulse rates between 9 and 11 Hz. This is a higher energy than the Argon-Ion laser which leads to an increase in accuracy because higher intensity reading are capable on the cameras. One percent of this beam is split off for use by the frequency control/monitoring unit, while the rest of the beam is conducted though an optical train that produces a sheet of light. The beam is first made taller by expanding thought a plano-concave lens before being collected by a plano-convex lens. The same process is repeated, in alternate order, to make the sheet thinner, producing a laser sheet approximately 88.9 mm tall and 3mm thick.

To communicate the timing of the 10 Hz pulse rate of the laser to the CCD line camera a Hewlett Packard 8012B pulse generator was used. The laser pulse output trigger was used as the trigger input to the pulse generator, with a 1 second delay on the output signal of 4.0 V to the CCD line camera. The output was also routed to a Berkley

Nucleonics Corp model 500 pulse generator. This pulse generator was used to multiply the period of the pulse to reduce the frequency of the image collection on the ALF and REF CCD cameras. This was needed since the computer was not able to process the images at a 10 Hz rate so a factor of 2 was use to reduce the rate to 5 Hz.

6.2 Frequency Control Unit

In C. Gaharan's PhD dissertation [40] a Fabre-Perot interferometer was used to measure the frequency of the Argon-Ion laser. In this application a Fabre-Perot interferometer cannot scan the spectrum fast enough to capture the 9 ns pulse of the ND:YAG laser. In Danczyk's PhD dissertation [41] a spectrometer utilizing a photo diode array operating with a diffraction gradient was used with the ND:YAG laser. It is fast enough to capture the event, but it was found that it did not have the frequency resolution necessary to measure the small changes in the frequency necessary for this application. This project developed a frequency monitoring system based upon the use of a CCD line camera to resolve both the frequency response and resolution issues. A USB2.0 3648-Pixel 16-bit CCD Line Camera with External Trigger from Mightex was utilized. The CCD allows the capture of light contained in the short duration of the pulse and downloads the light intensity information to the computer.

Similar to the optics of the Doppler image analyzer, a small portion of the laser's main beam is split off and conducted though an optical train that produced the equivalence of an ALF and REF signal, as seen in Figure 6. The signal beam is split into

two beams using a nonpolarizing beam splitter cube. One beam is aligned to illuminate one end of the CCD array. The other beam passes through the same type of absorption line filter used in the DGV and is aligned on the other end of the CCD line. The intensity levels of each the ALF and REF areas are summed and a ratio of ALF/REF can be calculated. This method was utilized to determine the stability and drift characteristics of the laser. It was determined that a monitoring and control system would need to be implemented to conduct accurate measurements due to the continual drift of the laser.

The CCD line camera was able to monitor the laser output at a rate of 10 Hz, providing data to LABView. This data was 16-bit resolution intensities similar to Figure 6, which in real time could be analyzed, determining the value of the ALF/REF ratio for each pulse for comparison to the desired ratio assigned by the user. LABView then, modified a digital to analog voltage output (USB-1208FS from Measurement Computing) that supplied the voltage to the diode that seeded the ND:YAG as needed to resolve the difference in the current and desired ratios, thus effectively controlling the drift of the laser. It was also observed that the laser would have a slightly different frequency (ALF/REF ratio) each pulse, showing poor stability of the laser. The data supplied to LABView for each pulse was saved into an ASCII file to later be used in processing the images. Knowing the ratio of each pulse may be adequate for solving the stability issue.

6.3 Doppler Image Analyzer Unit

The Doppler Image Analyzer, shown in figure 4, consists of a green filter, telecentric lens, transfer lens, beam splitter, mirror, ALF cell, and two CCD cameras with telephoto lenses. These components are housed in a sheet metal aluminum housing to reduce the noise caused by stray light. This analyzer unit followed the same design as Gaharan's PhD dissertation [40] and expanded upon his findings by improving hardware components to increase accuracy. The components that comprise the DGV system were bolted onto a highly rigid flat aluminum plate optical bench, allowing precise optical alignment and minimizing variations in alignment.

A green filter is the first optic to receive the image in the DGV system. This filter is used to reduce noise from stray light sources, ensuring that only the green light of the laser is passing thorough. The filter used is a 62 mm HOYA green colored glass filter (Item number G-533). This filter has a transmittance of about 55% at the wavelength of the ND:YAG (532 nm), less than 1% transmittance for wavelengths less than 410 nm, and less than 5% transmittance of wavelengths greater than 660.

The telecentric lens used to collect the Doppler shifted light was an Edmund Optics Techspec Silver Series 0.16X TML Telecentric Lens (Stock number NT56-675). Using a telecentric lens has a couple of advantages over a conventional telephoto lens. The primary advantage a telecentric lens brings to the optical train is that it reduces what are called edge effects. In a standard lens as you approach the outer rim of the lens the image becomes distorted and the light intensity decreased due to the curvature of the

lens, this is not the case with a telecentric lens. Thus, a telecentric lens conveys a more accurate image. The lens used had a telecentricity of less than 0.1º, distortion of less than 0.3% with a primary magnification of 0.16x, for a field of view of 40 mm, a depth of view of ±19.7 mm and a working distance of 175mm. A 100.0 mm focal length biconvex transfer lens from THORLABS (serial number LB1676-A) was paired with this optic to maintain a constant image size though the remainder of the optical train.

The next component is a nonpolarized 50/50 beam splitter that redirects half of the image to a front surface mirror that bounces a beam to the reference camera (REF) and the other half continues through the ALF cell to the ALF camera. The ALF cell in this system was upgraded to a sealed Quartz Reference Cell with iodine distributed by Thorlabs (Part Number CQ19100-I) from an O-ring sealed version used by Gaharan. The new cell is mounted in reference cell ovens (Part number GCH18-100) controllable by a TC200 temperature controller. This allows stabilization of the ALF cell temperature.

The key element of the DGV system is the ALF cell, which converts the Doppler frequency shift at each point in the illuminated plane into a map of varying intensity. When installed in front of the CCD camera, this filter ensures that each pixel stores an intensity value which corresponds to the local velocity vector. This cylindrical cell consisted of quartz with an internal length of 100mm and an inside diameter of 19 mm. This diameter was used to maintain a constant medium thickness over the cell's aperture. The cell windows are made from UV fused silica material. The windows are designed with a 2 degrees wedge to eliminate the etalon effects of parallel surfaced windows. A

fill stem is included to ensure that if condensation occurs in the cell it happens out of the viewing path. This fill stem consists of a glass tube shorter than 10 mm added on the side of the cell. The characteristics of the cell are temperature dependant. Therefore the electrical heater system specified previously is used. For the current study, it was turned off.

The ALF cell consists of an absorption line that is affected by three basic broadening processes described by Elliott [42]: natural broadening due to the finite lifetime of the excited state, Doppler broadening due to the random motion of the molecules, and Lorentz broadening due to inter-molecular collisions. Although natural broadening is a function only of the composition of the absorbing molecule, Doppler and Lorentz broadening are functions of the thermodynamic state of the absorbing medium. At moderate temperatures and low pressures, the absorption profile is primarily Doppler broadened with a Gaussian shape. However, when the absorption profile is dominated by Lorentz broadening, the sloping region near the edges of the absorption line occupies a much larger frequency range and its corresponding transmission typically only reaches half of the peak of a Doppler broadening profile. The cell used was sealed at 7.6×10^{-5} atm, this ensured that Doppler broadening profile was present. Heaters were present, but not needed, as the absorption line was found to have an adequate slope to make measurements. Heating increases the slope of the ALF cell transmission curve, increasing the sensitivity to Doppler Shift, Gaharan [40].

Nikon 70-210 mm zoom lenses were used to focus the image onto the major upgrade to the system, the CCD camera. The current DGV used detectors consisting of

charged-coupled device (CCD) elements which measured the optical energy of the flow field in question. The CCD consisted of an array of elements which made use of the electro-optical properties of a semiconductor photo detector in which the charge generation process was regarded as naturally linear, provided that the light intensity remained high enough for electron generation without reaching the unwanted range of over saturation. Thus a collection of CCD elements, such as video cameras, were used as detectors for image acquisition of the flow field. These consisted of a pair of Dalsa Pantera TF 1M30, 12-bit , monochrome video cameras with a 12.3 mm x 12.3 mm CCD arrays which utilized an active imaging area of 1024 horizontal by 1024 vertical pixels. This 12-bit capability can dramatically increase the accuracy over the 8-bit cameras previously used by increasing the gray scale resolution of the intensity reading from 256 to 4096 increments. The spatial resolution of the camera (number of pixels) was also increased to 1024 by 1024 pixels from 739 by 480. This allows the user to either view of a larger area with the same spatial accuracy as before or increased accuracy in the same area observed by a lower spatial resolution CCD camera. Increases in spatial and grayscale resolution are the primary modes to increase accuracy and lower the velocities at which a DGV system can function and accuracy of the velocity measurement, according to C. Gaharan's PhD dissertation [40].

6.4 Data Acquisition

LABView was the software implemented for data acquisition. A PCIe-1430 image acquisition card for each camera along with NI Vision Acquisition Software (NI_IMAQ) was used to communicate with the Dalsa CCD cameras. A parallel flat sequence was implemented to ensure that the ALF and REF images were recorded at the same time. The CCD arrays were brought into LABView and could be stored in numerous formats. The formats utilized were PNG, for its small storage requirements and its ability to be read into Sigma Scan Pro to be analyzed, and TIFF because this format retained true 12-bit gray scale resolution. It was found that the computer, a HP xw6400 workstation operating at 1.6 GHz with 2 GB of RAM, could not save the TIFF format at the rate the laser pulsed, 10 Hz, so an option was coded in to have images saved every other pulse. This solved the lag issue the LABView program was having operating at 10 Hz.

Conversely, the line CCD camera was able to be monitored and recorded at 10 Hz. The array of 3648 pixel intensities were brought into LABView, a ratio was calculated from summed values over an area that was lased by either the ALF or reference beam. A program had to be written in c++ called a wrapper, see Appendix E. This was needed because the SDK for the camera, Appendix F, also written in c++, utilized a callback function not supported by LABView's Call Library Function Node, used to call functions in an outside program stored in a dll. LABView then called a function in the wrapper dll that called a function in the SDX dll that was able to pass the

array into LABView. Once this array was in LABView, the calculated ratio was appended that of the pixel intensities, this new array was saved to an ASCII file for post processing.

In the final code used to control data the DGV system and record, there are a number of independent codes that are utilized within LABView. They can be broken up into different states of the process. First the user initializes the system. Ensure all components are on, including the laser, both pulse generators, ALF and REF CCD cameras, and CCD line camera. When the laser is turned on to full power the LABView program SLPLLring5.vi can be opened. This program utilizes the master-slave format, viewable in Figure (28). The user must input the destination file where data is to be save, the working mode (generally 1, for trigger mode), and the exposure time (100000 (μs) for our use). The CCD line camera is automatically initialized upon starting the program, using a Call Library Node function. An example of this function is shown in Figure (29). The initialization process is a number of functions that communicate with the CCD camera to ensure that the camera functions properly. These functions are and in this order: Initialize device, Add device to working set, Start camera engine, Set exposure time, Start working mode, Start frame grab, and Install Frame Hooker. Technical information about the functions can be found in Appendix (Y2). The exposure time is set to 100000 us (100 ms) to ensure that the camera captures the 9 ns pulse of the ND:YAG. The Working mode is set to Trigger or 1 so that it is triggered by the laser.

When the process 1 is activated in SLPLLring5.vi, named ‘View CCD camera Manual Controls’, images from the CCD line camera will begin to be imported. The array that is imported is displayed on the computer monitor in two graphs, ‘Waveform Graph’ and ‘XY Graph’. The cursers on the waveform graph are used to identify the pixels where the two laser beams (ALF and REF) of the laser monitoring/control system are located. A start and stop cursor are available for both beams. The pixel number corresponding to each curser position are displayed. The values of these should be entered into the point inputs to the right. The values for curser 1 entered for point 1, curser 2 for point 2 and so on. These are used to set the upper and lower pixel limit for each beams power calculation on which the ratio used to monitor the laser frequency is calculated. Once the values are entered for the point inputs, the current ratio output will begin displaying a value. At this point, a desired ratio should be selected and a voltage should be applied to the seed diode, both using the designated sliders. A gain should also be assigned. This gain is used to set the feedback loop sensitivity relating the change in seed diode control voltage to the difference in the actual and desired laser operating frequency as measured by the ratio. A value of 0.003 was typical for use with the ND:YAG. It may be to be negative depending on which side of the ALF notch filter is being used.

Once the ratio is relatively close to the desired ratio by manually controlling the voltage, Process 2 should be activated, ‘Active Voltage Controls.’ This will monitor the average of the last 50 ratios, subtract it from the desired ratio, multiply the difference by the gain, subtract this value from 1, and multiply the previous voltage by the result to

produce the new voltage administered to the seed diode thought the Analog out function. This system actively controls the seed diode and stabilizes the long term drift identified in the ND:YAG.

Next, Process 4 is activated, 'Dalsa Cameras,' that starts monitoring the Dalsa CCD ALF and REF cameras on Image port 0 and 1. This utilizes the HL Snap4-tif.vi modified to operate at the proper frequency. This is done just before capturing images because it significantly slows down the computer.

Finally, Process 3 is activated, "Save CCD output and ratio (current).' This saves the measured ratio of the frequency control unit for the current pulse, along with the output from the CCD line camera to an ASCII text file. The Dalsa CCD cameras are also saved in a TIFF format at the designated file location, all recorded for a single pulse of the laser. The number of data sets (one for each pulse) are designated by the user. The Dalsa cameras are currently set to capture at a rate of 5 Hz, which was found to be the maximum rate at which the computer could save complete data sets.

7. IMAGE PROCESSING TECHNIQUE

This section outlines the methodology used to setup and present a DGV velocity image using the system outlined in previous sections. In order that the final reduction code is capable of producing an accurate velocity profile, two calibrations must be completed first. A spatial calibration that includes centroid estimation and a transform from (i,j) pixel number to (x,y) spatial coordinates must be conducted to ensure proper alignment between the cameras. The intensity calibration correlates pixel intensity to "lumens" that can be directly related to velocity after a velocity calibration.

7.1 Spatial Calibration

Early publications of Meyers [9, 10] and Usry [43] concluded that it is essential for the signal and reference images to be accurately aligned with one another prior to calculating the intensity ratio. Large errors in the measured velocity may result from relatively small errors in the alignment of the two images. Those early studies did not attempt to quantify the magnitude of the expected velocity error, or to investigate the requirements for image alignment. A requirement for more accurate alignment suggests that physical manual alignment methods employed in early systems are inadequate and that a comprehensive image processing strategy for Doppler Global Velocimetry is necessary.

As a consequence of imperfections in the optical system such as astigmatism and imperfect alignment of optical components, the two images may differ in a number of ways. One image maybe slightly rotated relative to the other, a slightly different magnification relative to the other, or distorted in a more complex way due to the behavior of the lenses in the system. It is clear that the accuracy, with which the images must be aligned, in order to yield accurate quantitative velocity measurements, is such that a sophisticated image processing approach is required. Therefore, the following process was enacted to accurately and systematically align the images of the ALF and REF cameras for DGV measurement use.

7.1.1 Centroid Estimation

The method of using centroid estimation and linear transforms are implemented to effectively calibrate the pixel location of each camera to a known physical plane, graph paper. This is to ensure that when the ratio of the two pictures is taken that both images are of the same area in space. The spatial calibration picture was taken with HL Snap4-png.vi in LABView for both the ALF and REF Dalsa cameras. This program makes use of the NI-IMAQ software package along with the PCIe-1430 image acquisition device to communicate with the cameras, load and save the images. A picture is taken of a paper with black dots gridded 5mm apart, illuminated with an incandescent light and saved as a .png file. Examples are presented in Figures 7 and 8.

Each .png file is loaded into Sigma Scan Pro 5 where an intensity threshold was applied to illuminate only the black dots. A range of 0 to 121 was used, with whole scale being 255. This is done in order to fully illuminate the circles, so that an accurate measurement of their centroid may be taken. This was an option because of the significant disparity between the intensity of the dots and the paper, black and white respectively. Next in the measurements and setting pull down menu an area was calculated along with center of mass in the i and j directions. These measurements were conducted for both the ALF and REF images and saved to spreadsheets.

These data were opened in Excel where the data were sorted by area. Items identified with an area larger than 1500 and less than 900 were deleted, as they were either darkened corners or not fully illuminated dots that could lead to a faulty calibration. Next the data were sorted by i, the horizontal pixel location, dots in row 1 were assigned 0 mm for their x, the calibrated horizontal location. Row 2 was assigned 5 mm, and so on to 35 mm. The same was done to convert j to y. The results are graphed. Figures 9 and 10 present the centroidal i vs. j, pixel location 0 to 1024. It was noticeable that the dots are aligned at an angle. The corrected x vs. y, 0 to 35 mm, graph in which all the dots were grid-like are shown in Figures 11 and 12.

7.1.2 Transform; (i,j) to (x,y)

The worksheet of (x,y) and (i,j) locations was saved and later loaded into TableCurve 3D. A curve fit was performed by specifying x(i,j), y(i,j), i(x,y), and j(x,y).

These spatial transform pairs between (i,j) and (x,y) are used in the analysis of the data. To obtain these equations, each set of data had to be curve fit. For example, the location 'x' was graphed in terms of 'i' and 'j' and an equation generated. The process was repeated for each of the other three coordinate transformations. A set of transformations were obtained for each camera. This allowed for forward and backwards transformations. Under the 'Process' and 'Surface' pull down menus a 'Fit Simple Equations' function was specified. Equation 1 was used, with the form z=a+b*x+c*y and an r-squared value of .9999. Graphs of these fits can be seen in Figures 13-20. This completes the curve fit for spatial transformations. Now for a given location in space, (x,y), there is a pixel location, (i,j), for both the ALF and REF cameras that correspond to that location. The equations obtained that convert between both coordinate systems are presented in Tables 1 and 2. In the use of these transformations, the values of i and j may not be integers. In that case, linear interpolation is used to estimate the light intensity at the desired location.

7.2 Intensity Calibration

One of the reasons CCD cameras are used in DGV systems is because of their linear response to light intensity. Thus, with a simple calibration from intensity level of the pixel to 'lumens' and then comparing this lumen level to a known light source, the system can be effectively calibrated. The intensity calibration utilized the HL Snap4-tif.vi in LABView, seen in Appendix D, to capture an image of a white piece of paper

illuminated by an incandescent light and saves it in the 12-bit TIFF format. On top of a green filter, so that only a sensitivity to green light was recorded, ND filters ranging from 0.3 to 0.9 OD, optical density, along with combinations of 0.9 with 0.3 and 0.9 with 0.6 lenses were used to cover the intensity range of the Dalsa CCD cameras. This ranged from almost saturated to no change, on the dark end. This allows for more accurate interpolating of data intensities. The transitivity of ND filters is given by:

$$OD = \log_{10}(1/T) \quad \text{or} \quad T = 10^{-OD} \qquad (3)$$

The set of five tiff images (0.3, 0.6, 0.9, 0.9+0.3, and 0.9+0.6) for each camera are supplied to the program IntensityCal.m in MatLab, seen in Appendix Z. A linear fit for each pixel location is performed relating pixel intensity to transitivity, in the form y=a*x+b. An example of one of these linear fits can be seen in Figure 21. Information on the polyfit function used in the Matlab code can be found in Appendix H. The variable ‘a’ and ‘b’ for each pixel for each camera are stored into respective matrices and saved in a .mat format to be called upon by the reduction code.

It was found that there are more inaccuracies close to the edges and corners of the image. These, likely, can be related to lens effects of the telephoto lens and also because portions of the CCD image view is outside of the DGV image (portion where there is no measurable image). Thus, the transfer function is more accurate in the center of the image and so the measurements will also be more accurate in the center of the image. This fact should be kept in mind when recording images.

7. 3 Reduction Codes

The data reduction codes are designed to automate the process of converting the output from the LABView data acquisition code to a viewable, measureable image. The code was written in Mat Lab, datareduction.m, and is viewable in Appendix G. Datareduction.m utilizes the outputs from the spatial and intensity calibrations to analyze data from the DGV system and compute the ratio of ALF/REF at specified spatial locations. The results are presented in graphical gray scale plots and stored in digital form.

First, the user is prompted for ‘x’ and ‘y’ min, max, and step size. This is used to calculate the number of cells required for the desired grid sizing. Using the (i,j) to (x,y) transformations determined in the spatial calibration, a check is conducted to ensure that the desired grid remains in the i-j data set (i.e. no x or y values go above the available 1024 pixels or negative where there are no pixels to be read). If an error is returned, a request for a new x-y grid is made; otherwise the x-y grid is produced and saved in memory for later use.

The .mat files saved from the intensity calibration are loaded in along with the TIFF files and ratio information from the CCD line camera of the data set that is desired to be reduced. The intensity levels of each pixel are converted to ‘lumens’. This is the step where the linear interpolation from the intensity calibration is used. With the function form: y=a*x+b, where y is lumens, and x is intensity. There is an ‘a’, ‘b’, and

intensity for each pixel location saved in 1024 x 1024 matrices, the algebra is conducted to form a matrix of ‘lumens’.

Next, each point of the user defined x-y grid has its value in ‘lumens’ from the i-j matrix calculated using linear interpolation. This is done by transforming the (x,y) coordinates to(i,j) coordinates using the transform from the spatial calibration, then using a function called Interp2. Information in Interp2 is available in Appendix H, to linearly interpolate the light intensity at the i-j point from the surrounding 4 points. This is necessary since the accuracy associated with the gridding always places you in between pixels in both the i and j directions.

The ratio of the two ‘lumen’ matrices, ALF and REF, is calculated by dividing the ALF ‘lumen’ value by the REF. The resulting matrix is graphed on a gray scale with the ratio from the CCD line camera in the title. With a proper velocity calibration this image would produce accurate velocity measurements possibly with a resolution of 0.96 m/s (see section 8.2).

7.3.1 Velocity Calibration

The final calibration to be done was to compare the readings produced by the DGV system to that of a know velocity field. A Dremel tool was used to rotate 4” metal disk whose speed was monitored using a photo-tachometer to a speed of approximately 22,500 rpm. The disk was painted white to reflect the maximum amount of light. The disk was placed such that the light sheet illuminated its face evenly, with the analyzer

unit located at an angle of 90º from the output laser beam. Images were taken of the ALF and REF Dalsa camera, along with data from the CCD line camera to monitor laser frequency.

When it was evident that the laser was not operating properly, the first thought was that perhaps the speckle, common in ND:YAG lasers, was causing problems with the readings. The datareduction.m code was modified and called ptremover.m (Appendix G) to remove these points, replace them with a linear interpolation between surrounding points and conduct the calculations again. The mean of the pixel intensity of each ALF and REF images are calculated, this should be equal to the desired ratio. If the intensity of a given pixel is greater than that mean, this means that it is a speckle from the laser, it was removed from the matrix and replaced with the linearly interpolated value. This did not resolve the issue.

It was evident in the attempt to perform the velocity calibration that the laser was not operating properly. A sample spinning wheel velocity calibration ALF/REF image is shown in Figure (22). Instead of horizontal bands of constant value which decrease in the vertical direction as seen by Gaharan (Figure (23)), a relatively random distribution was obtained. A correlation was conducted on the data from the Line CCD camera, correlating the ALF filtered beam to the REF beam. The correlation was archived and is presented in Figure (24). When the data at each ALF and reference beam correlated pixel was compared an approximately horizontal line was expected at the desired ratio indicating the specified 15MHz laser beam band width. This was not the case, as seen in Figure (25). Figure (25) shows the ratio of the correlation varying

from 0.3 to 1.0. This shows that the transmission ratio through the ALF cell varied from 0.3 to 1.0 across the laser beam profile. Based upon the ALF cell frequency/Transmission profile published by Elliot and Samimy [44], the laser frequency is varying over 600 MHz across the beam, not the 15 MHz specified. This would make the frequency bandwidth significantly larger than required for the DGV device to operate. A program was written in LABView to conduct this autocorrelation in real time to help with the diagnosis of the laser and added to the data acquisition and laser stability program.

8. RESULTS AND DISCUSSION

This section presents the results obtained pertaining to laser stability, sources of error and information about the laser malfunction. These topics were important to the objectives of this system development and will be important to continued research in this area.

8. 1 Laser Stability

Laser stability is of the utmost concern in a DGV system, since Doppler shifts are measured relative to the laser output light frequency during the integration period of the CCD camera. It was evident early on that a manual control system previously used with an Argon-Ion laser would not suffice for the ND:YAG. The time constant of drift was too short to manage manually, and there did not appear to be a settling point after a warm up period. Also, each pulse was of a great enough frequency difference that it would have significantly skewed results. Thus, the CCD line camera scheme was devised to monitor, control and record the frequency of the laser. This system was designed to atone for both the drift and stability issues of the ND:YAG. Because of a malfunction of the laser, it was impossible to quantify the laser stability. Ideally this system would improve upon the numbers presented by C. Gaharan's PhD dissertation [40], with a frequency variation of less than $-4.47 < \Delta\nu < 5.62$ MHz which produced a

fixed velocity uncertainty, on average, of ±0.2 m/s that equally affected each measurement location.

As stated before, a CCD line camera monitoring the ALF and REF beams of the ND:YAG laser was implemented to control the frequency of the laser. Without control there was significant drift, as evident in Figure (26). With a control system that monitored the ratio of each pulse of the laser it was determined that to maintain an average pulse frequency a 50 pulse average of the system necessary to maintain the desired ration, shown in Figure (27). The pulse to pulse variance about this mean was plus or minus 0.5 of the desired ratio as the DC voltage controlling the laser frequency continually increased over time. To compensate for this variation, the ratio was stored with the images. Hypothetically this information may be used to compensate for this instability. The hypothesis was never evaluated since the laser beam frequency bandwidth was approximately 600 MHz rather than the required 15 MHz for accurate measurements.

8.2 Summary of Errors

In C. Gaharan's PhD dissertation [40] the DGV system developed was considered to contain five significant sources of error. These were as follows: laser frequency drift ±0.2 m/s, iodine cell stability ±0.3m/s, pixel sensitivity calibration ±0.8 m/s, "average" image misalignment ±2.3 m/s and observer ($\hat{o}$) laser orientation ($\hat{l}$)

±0.34 m/s. Therefore, the overall error is estimated to be $\left[\sum\left(\omega_i\right)^2\right]^{\frac{1}{2}}$ =3.94 m/s. One of the project's goals was to reduce this uncertainty. The following items address this goal.

From Gaharan's calculation, the iodine cell stability and observer ($\hat{o}$) laser orientation ($\hat{l}$) would remain the same for the present study. The reduction in the laser frequency drift would not be appreciable relative to the other two larger errors, so ±0.2 m/s can be assumed as a worst case scenario. The pixel sensitivity calibration should be reduced by a factor of 16 due to the change from 8-bit (256 intensity levels) cameras to 16-bit (4096 intensity levels). This would reduce the error seen in the pixel sensitivity calibration from ±0.8 m/s to ±0.05 m/s.

Finally, the error due to misalignment (previously ±2.3 m/s) can be considered. This was produced from an error in Lumen gradient given by:

$$\omega_L = \frac{\partial L}{\partial x}\Delta x \qquad (4)$$

where $\frac{\partial L}{\partial x}$ is the local measured Lumen gradient, Δx is the amount of misalignment previously determined, and ω_L is the uncertainty in Lumen levels. Equation (4) must be calculated at each pixel for both directions and cameras thereby producing four separate sets of uncertainties called: $\omega_{L,x,ALF}$, $\omega_{L,y,ALF}$, $\omega_{L,x,REF}$, and $\omega_{L,y,REF}$ where the subscripts denote corresponding directions and cameras. These errors can be combined using a Kline-McKlintock uncertainty analysis to obtain:

$$\omega_{L,misalign} = \left[\left(\frac{1}{m_{L,x,REF} + m_{L,y,REF}} \right) \omega_{L,x,ALF}^2 + \left(\frac{1}{m_{L,x,REF} + m_{L,y,REF}} \right) \omega_{L,y,ALF}^2 + \ldots \right.$$

$$\left. \ldots \left(\frac{m_{L,x,ALF} + m_{L,y,ALF}}{m_{L,x,REF}^2} \right) \omega_{L,x,REF}^2 + \left(\frac{m_{L,x,ALF} + m_{L,y,ALF}}{m_{L,y,REF}^2} \right) \omega_{L,y,REF}^2 \right]^{\frac{1}{2}} \qquad (5)$$

where m denotes the measurement in lumens at each point using the same subscripts as the error. Once again, changing form the 8-bit to 12-bit cameras will reduce the Lumen gradient uncertainty by a factor of 16. Also, increasing the spatial resolution by a factor of two will reduce Δx by factor of 1.5 as well. Total factor of reduction, 24, is valid for each of the interpolated Lumen error, thus also reducing the misalignment error by the same factor reducing the total uncertainty to ±0.1 m/s.

With the new components in place, this new DGV system has the potential of reducing the uncertainty to 0.99 m/s. This is a reduction by a factor of 4 from Gaharan's uncertainty calculations. The new system has effectively reduced the two largest contributors to the error by magnitudes, now shifting the focus onto the remaining components, laser frequency drift, iodine cell stability and observer ($\hat{o}$) laser orientation ($\hat{l}$) uncertainties, for the next generation DGV.

9. CONCLUSIONS AND RECOMMENDATIONS

The prior stated objective of increasing the DGV measurement accuracy has been achieved and a proof of concept has been produced with a minor error in the lasing system. The fundamentals are now in place so that when the laser is functioning properly a DGV system that will be able to capture turbulence modes within a flow field is available.

9.1 Conclusions

Updates to the optical train of the DGV, telecentric lens, 12-bit CCD cameras and sealed iodine cell, have produced a more accurate and stable system. These additions were able to reduce the uncertainty by over a factor of four. This will effectively increase the working rage of a DGV system into lower velocities where older systems have experienced too much uncertainty to produce an accurate measurement.

The use of a pulsed laser presents a very exciting development to DGV. This system will have the ability to capture turbulence modes within a flow field, previously unmeasureable by DGV. This system still retains the ability to average multiple images to produce a mean velocity field image.

The development of a real time spectrum analyzer was a success. The use of a CCD line camera to capture a pulse, along with the software written to analyze the data

and control the laser diode produced a robust system for short pulse lasers and can be utilized with numerous other lasing systems as well.

The intensity calibration used to determine the slope for each pixel sensitivity, produced a more accurate calibration than previously used. This was a more time consuming and computational intensive procedure, but the arrays of linear variables produced would facilitate the migration to a real time DGV measurement and monitoring system.

9.2 Recommendations

Many improvements can be made upon the DGV system that would increase the accuracy of the data.

First, an improvement over previously used CCD cameras was shown. The implementation of more accurate cameras as they become available will continue to increase accuracy. It is notable that uncertainties in other systems should be reduced as well as they will become the primary sources and changes in CCD accuracy will increase accuracy negligibly. A method using one camera can be implemented to simplify the DGV system. This is achieve by placing an image combining mirror in front of the single CCD camera, enabling the two images to be recorded side by side. Thorpe et al. [45], Chan et al. [20], and Smith et al. [13] implemented this method.

Also, as computer technology and processing speeds increase, data reduction times will decrease and can eventually become a real time system if the image can be

processed faster than the pulse rate. Use of multicore processors and software that utilizes these cores can also reduce computational time. National Instruments has begun to develop LABView software that is capable of harnessing this processing power. Research into the application to this system should be considered for future iterations.

A telecentric lens was implemented in the optical train of this DGV system. Replacement of the Nikon 70-220 mm telephoto zoom lenses with telecentric zoom lenses should be considered. This would reduce the inaccuracies in the spatial and intensity calibration even further.

Reduction of the, now major, sources of error should be researched. These are as follows: Laser frequency drift, iodine cell stability, and observer laser orientation. There can be a significant increase in accuracy with a focus on these areas in the next iteration.

Finally, diagnosing the problem with the ND:YAG laser system is paramount in the success of this system.

By following these recommendations, many obstacles can be eliminated in the development and data acquisition of a DGV system.

REFERENCES

[1] Komine, H., 1990, "System for Measuring Velocity Field of Fluid Flow Utilizing a Laser-Doppler Spectral Image Converter," United States Patent No. 4,919,536, Northrop Corp., Hawthorne, CA.

[2] Morrison, G. L., and C. Gaharan, 1998, "On the Development of a Doppler Planer Velocimeter," *Proceedings of ASME Fluids Engineering Division Summer Meeting*, Washington, DC.

[3] Yeh, Y., and H. Z. Cummins, 1964, "Localized Fluid Flow Measurements with a He-Ne Laser Spectrometer," *Applied Physics Letters*, **4** (11), pp.176-178.

[4] Komine, H., and S. J. Brosnan, 1991, " Instantaneous, Three-Component, Doppler Global Velocimetry", *Proceedings of the ASME 4th International Conference on Laser Anemometry, Advances and Applications*, **1**, pp. 273-277.

[5] Komine H., S. J. Brosnan, A. B. Litton and E. A. Stappaerts , 1991, "Real-time, Doppler Global Velocimetry," *AIAA Paper* 91-0337.

[6] Meyers, J. F. and H. Komine, 1991, "Doppler Global Velocimetry - A New Way to Look at Velocity," *Proceedings of the ASME 4th International Conference on Laser Anemometry, Advances and Applications*, Cleveland, OH, **1**, pp. 289-296.

[7] Meyers, J. F., J. W. Lee, and A. A. Cavonne, 1991, "Signal Processing schemes for Doppler Global Velocimetry," *IEEE – 14th International Congress of Instrumentation in Aerospace Simulation Facilities*, Rockville, MD, pp. 321-328.

[8] Meyers, J. F., 1992, "Doppler Global Velocimetry - The Next Generation," *AIAA Paper* 92-3897.

[9] Meyers, J. F., 1994, "Development of Doppler Global Velocimetry for Wind Tunnel Testing," *AIAA Paper* 94-2582.

[10] Gorton, S. A., J. F. Meyers and J. D. Berry, 1996, "Velocity Measurements near the Empennage of a Small-scale Helicopter Model," *Proceedings of the 1996 American Helicopter Society 52nd Annual Forum*, **1**, pp. 517-530.

[11] Ford, H. D. and R. P. Tatam, 1995, "Imaging System Considerations in Doppler Global Velocimetry," *Proceedings of SPIE - The International Society for Optical Engineering, Optical Techniques in Fluid, Thermal, and Combustion Flow*, **2546,** pp. 454- 464.

[12] Smith, M. W. and G. B. Northam, 1995, “Application of Absorption Filter-Planar Doppler Velocimetry to Sonic and Supersonic Jets,” *AIAA Paper* 95-0299.

[13] Smith, M. W., G. B. Northam and J. P. Drummond, 1996, “Application of Absorption Filter Planar Doppler Velocimetry to Sonic and Supersonic Jets,” *AIAA 1996,* **34** (3), pp. 434-441.

[14] Smith, M. W., 1998, “Application of a Planar Doppler Velocimetry System to a High Reynolds Number Compressible Jet,” *AIAA Paper 98-0428.*

[15] Smith, M. W., 1998, “The Reduction of Laser Speckle Noise in Planar Doppler Velocimetry,” *AIAA Paper 98-2607.*

[16] McKenzie, R. L., 1996, “Measurement Capabilities of Planar Doppler Velocimetry Using Pulsed Lasers,” *Applied Optics*, **35** (6), pp. 948-964.

[17] McKenzie, R. L., 1997, **“**Planar Doppler Velocimetry Performance in Low-speed Flows,” *AIAA Paper* 97-0498.

[18] McKenzie, R. L., 1997**, “**Planar Doppler Velocimetry for Large-scale Wind Tunnel Applications,” *AGARD Fluid Dynamics Panel 81st Meeting and Symposium on Advanced Aerodynamic Measurement Technology,* Seattle, WA., paper no.9.

[19] Thorpe, S. J., R. W. Ainsworth and R. J. Manners, 1996, “Time-averaged Free-jet Measurements Using Doppler Global Vvelocimetry,” *Proceedings of the Fluids Engineering Division,* ASME 1996, **4**, pp. 59-66.

[20] Chan, V. S., D. I. Robinson, J. T. Turner, 1995, “A Simplified Doppler Global Velocimeter,” *Proceedings of Laser Anemometry,* ASME 1995, **229**, pp. 9-16.

[21] Reeder, M. F., 1996, “Jet Entrainment Measurements via Doppler Global Velocimetry,” *Proceedings of the Fluids Engineering Division,* ASME 1996, **4**, pp 129-134.

[22] Morrison, G. L., C. A. Gaharan and R. E DeOtte Jr, 1995, “Doppler Global Velocimetry: Problems and Pitfalls,” *Flow Measurement and Instrumentation,* **6** (2), pp. 83-91.

[23] Roehle, I., 1996, “Three-dimensional Doppler Global Velocimetry in the Flow of a Fuel Spray Nozzle and in the Wake of a Car,” *Flow Measurement and Instrumentation* **7** (3-4), pp. 287-294.

[24] Muller, H., Lemacher, T., Pape, N., Strunck, V., Dophenide, D., 2008, "3D-DGV for Flow Field Investigation in Pipes," Technical paper, Physikalisch-Technische Bundesanstalt, Branschweig, Germany.

[25] Nobes, D. S., Ford, H. D., Tatam, R. P., 2008, "Three Component Planar Doppler Velocimetry Using Imaging Fibre Bundles," Technical paper, Optical Sensors Group Center for Photonics and Optical Engineering School for Engineering, Cranfield University, Cranfield, Bedfordshire, UK.

[26] Willert, C., Stockhausen, G., Klinner, J., Lempereur, C., Barricau, P., Loiret, P., and Raynal, J. C., 2007, "Performance and Accuracy Investigations of Two Doppler Global Velocimetry Systems Applied in Parallel," *Measurement Science Technology*, **18** (2007), pp. 2504-2512.

[27] Arnette, S. A., Elliott, G. S., Mosedale, A. D. and Carter, C. D., 2000, "Two-color Planar Doppler Velocimetry," *AIAA Journal* **38** (11), pp. 2001-2006.

[28] Willert, C., Hassa, C., Stokenhausen, G., Jarius, M., Voges, M., and Kilnner, J., 2008, "Combined PIV and DGV Applied to a Pressurized Gas Turbine Combustion Facility," technical paper, Insititue of Propulsion Technology, Germany Aerospace Center, Koeln, Germany.

[29] Bloom, S., Kremer, R., Searcy, P. A., Rivers, M., Menders, J. and Korevaar, E., 1991, "Long-range, Noncoherent Laser Doppler Velocimetry," *Optics Letters* **16** (22), pp. 1794-1796.

[30] Menders, J., Benson, K., Bloom, S. H., Liu, C. S. and Korevaar, E., 1991, "Ultranarrow Line Filtering Using a Cs Faraday Filter at 852 nm," *Optics Letters* **16** (11), pp. 846-848.

[31] Fischer, A., Buttner, L., Czarske, J., Eggert, M., Muller, H., 2008, "Investigation of Time-resolved Single Detector Doppler Global Velocimetry Using Sinusoidal Laser Frequency Modulation," Technical paper, Dresden University of Technology, Dresden, Germany.

[32] Coupland, J., 2000, "Coherent Detection in Doppler Global Velocimetry: A Simplified Method to Measure Subsonic Fluid Flow Fields," *Applied Optics*, **39** (10) , pp. 1505-1510.

[33] Leporcq, B., Le Roy, J. F., Pinchemel, B. and Dufour, C., 1996, "Interest of a CW Dye Laser in Doppler Global Velocimetry," *Proceedings of the ASME Fluids Engineering Division Summer Meeting* **239** (4), pp. 115-122.

[34] Chan, V. S. S., Heyes, A. l., Robinson, D. I., Turner, J. T., 1995, "Iodine Absorption Filters for Doppler Global Velocimetry," *Measurement Science Technology*, **6** (1995), pp. 784-794.

[35] Chen, X., Zhang, K., Wang, Y., 1999, "Hyperfine Spectra of Iodine Molecule Available to Diode Laser Frequency Standard at 630-640nm (Experiment)," *Cleo/Pacific Rim*, **2** (90), pp. 1004-1005.

[36] Lazar, J., Hrabina, J., Petru, F., Jedlicka, P., Cip, O., and Smid, R., 2007, "Absolute Frequency Shifts of Stabilized ND:YAG Lasers – A Question of Iodine Cell Technology," IEEE.

[37] Miles, R. B., Lempert, W. R. and Forkey, J., 1991, "Instantaneous Velocity Fields and Background Suppression by Filtered Rayleigh Scattering," *AIAA Paper* 91-0357.

[38] Miles, R., Band, Lempert, W. R., 1996, "Three-dimensional Diagnostics in Air and Water by Molecular Tagging and Molecular Scattering," *27th AIAA Fluid Dynamics Conference* (New Orleans, LA) *AIAA Paper* 96-1963.

[39] Elliott, G. S., Samimy, M. and Arnette, S. A., 1994, "Details of a Molecular Filter-based Velocimetry Technique," *AIAA Paper* 94-0490.

[40] Gaharan, C. A., 1996, "The Development of a Doppler Global Velocimeter and its Image Processing Schemes for Whole-Field Measurements of Velocity," PhD Dissertation, Texas A&M University.

[41] Danczyk, S. A., 2002, "Experimental and Computational Investigation of the Flow Field Inside an Axial Fan," PhD Dissertation, Texas A&M University.

[42] Elliot, G.S., M. Samimy, and S. A. Arnette, 1992, "Filtered Rayleigh Scattering Based Measurements in Compressible Mixing Layers," AIAA paper 92-3543, AIAA 30th Aerospace Sciences Meeting and Exhibit, Reno, NV.

[43] Usry, J. W., J. F. Meyers, and L. S. Miller, 1992, "Doppler Global Velocimeter Measurements of the Vortical Flow Above a Thin Delta wing," AIAA paper 92-0005.

[44] Elliot, G. S., M. Samimy, 1996, "A Molecular Filter Based Technique for Simultaneous Measurements of Velocity and Thermodynamic Properties, " AIAA Paper 96-0304.

[45] Thorpe, S. J., R. W. Ainsworth, R. J. Manners, 1995, “The Development of a Doppler Global Velocimeter and its Application to free Jet Flow,” *Proceedings of Laser Anemometry*, ASME 1995, **229**, pp. 131-138.

APPENDIX A

FIGURES OF THE DGV SYSTEM

COMPONENTS AND SETUP

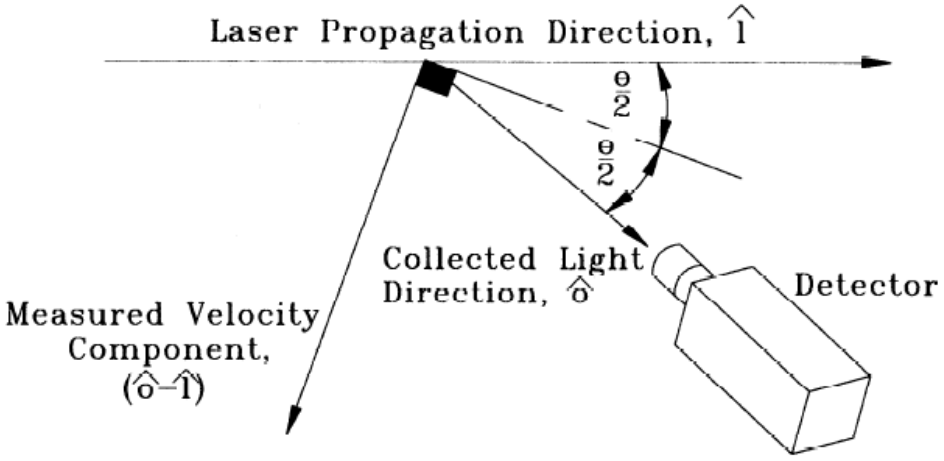


Figure 1. – Determination of measured velocity components direction based on laserbeam propagation and receiver location

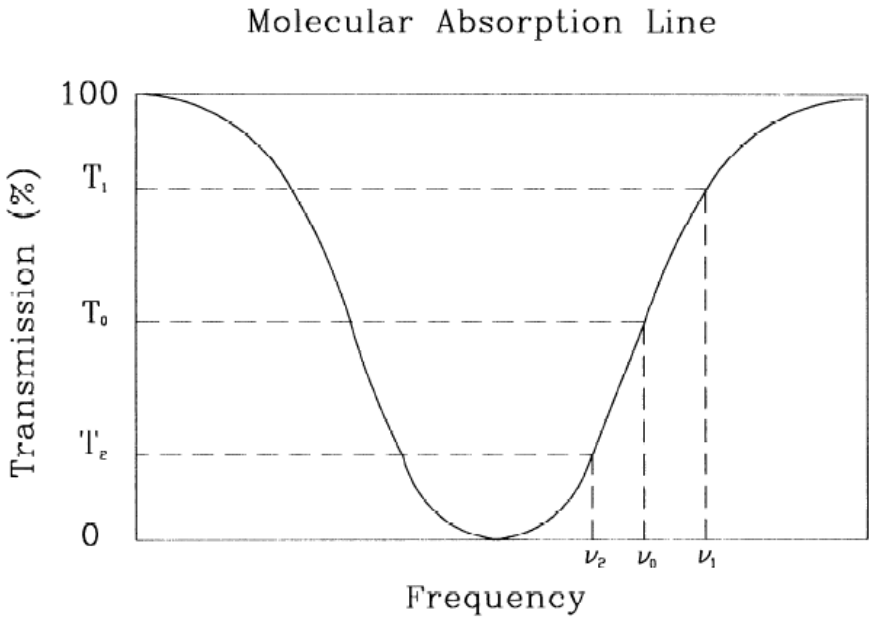


Figure 2. – Tranmission curve vs. frequency of scattered light

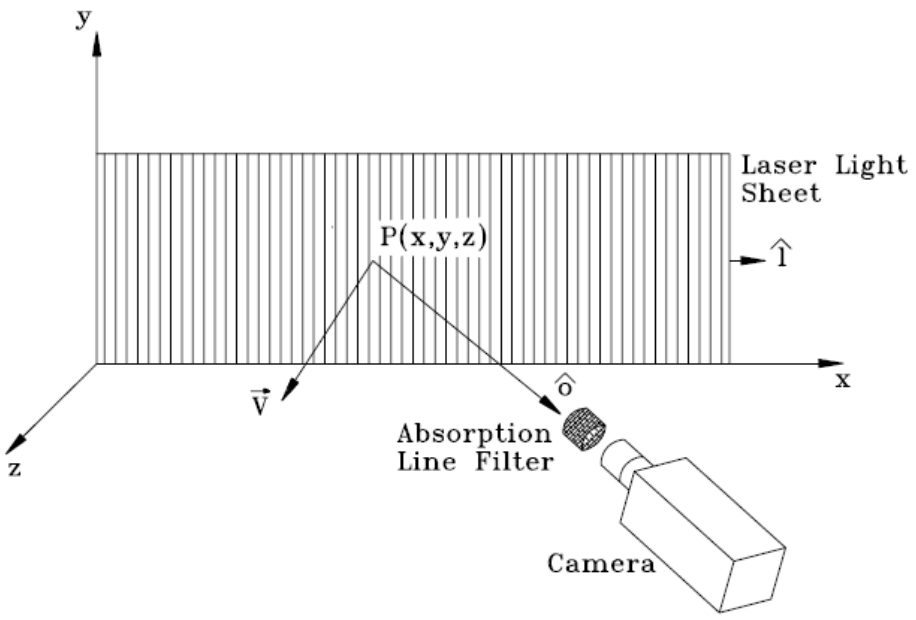


Figure 3. – Global Doppler shift in laser light sheet

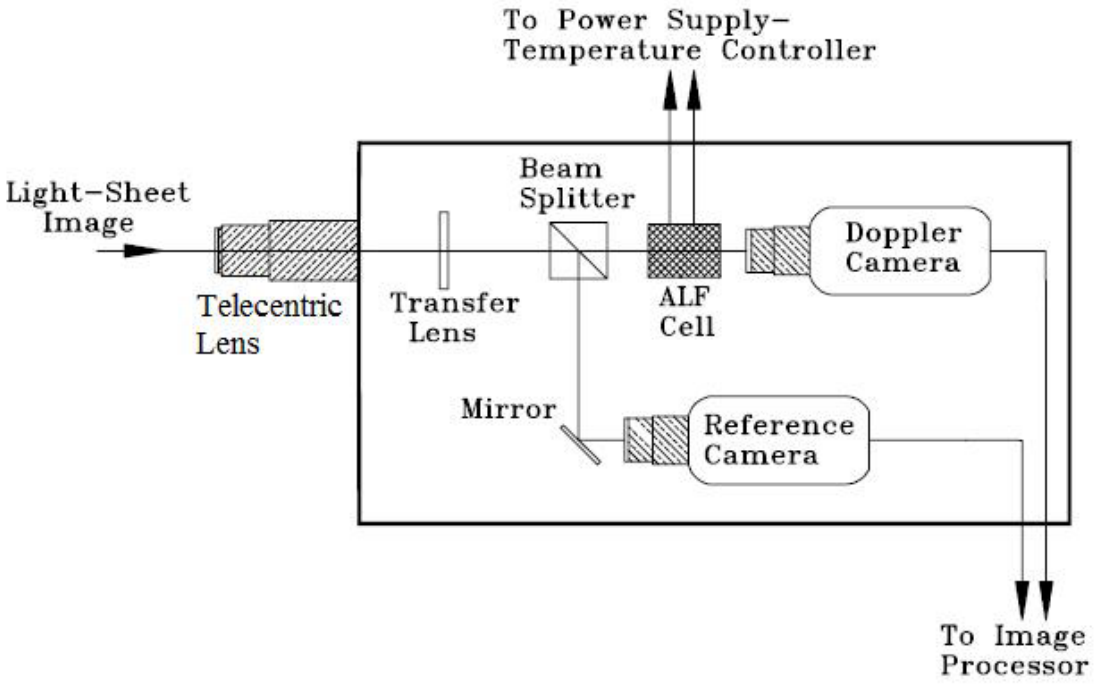


Figure 4. – Optical schematic of Doppler image analyzer

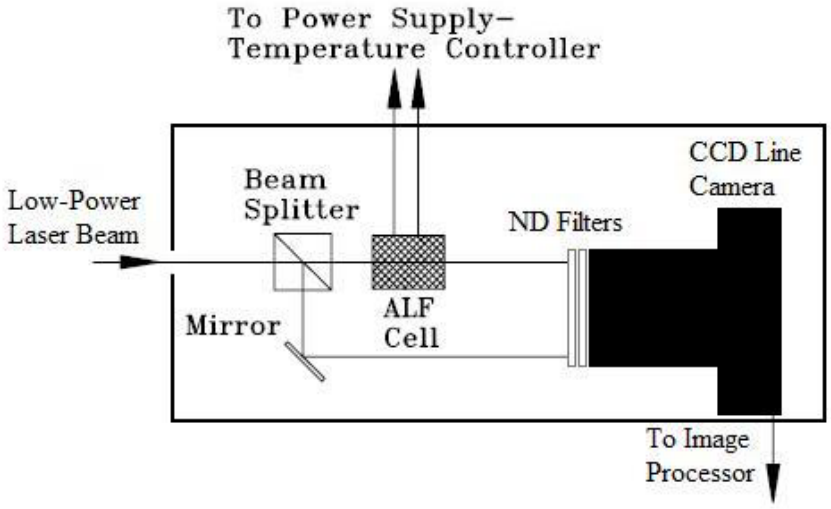


Figure 5. – Optical schematic of Frequency Control Device

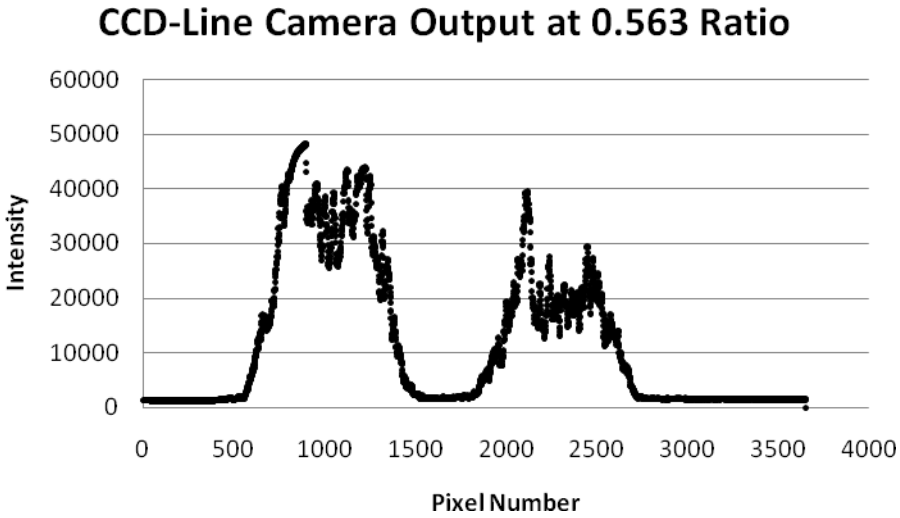


Figure 6. – Example of CCD-Line camera output; Intensity vs. pixle location

APPENDIX B

CALIBRATION DATA

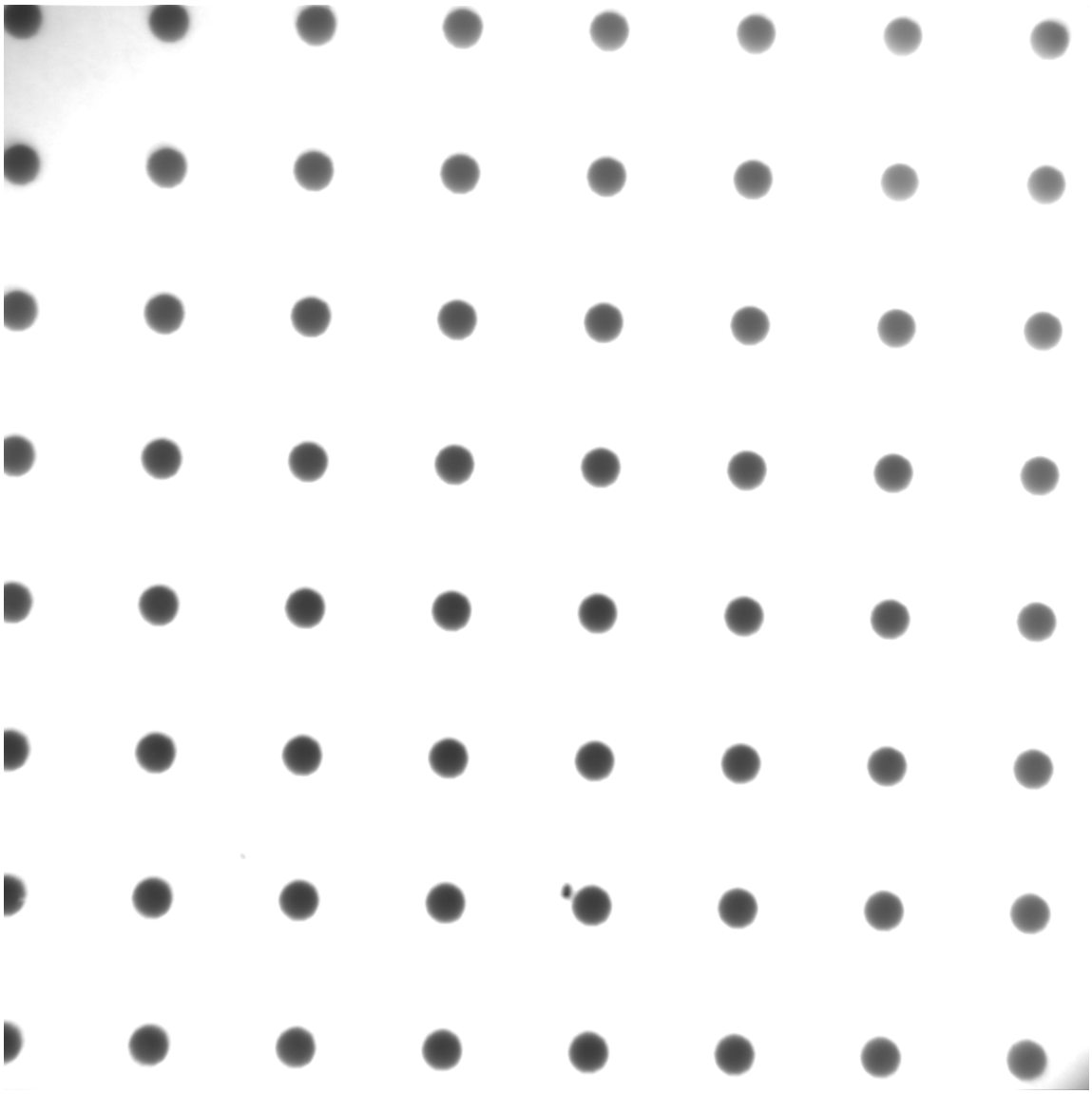


Figure 7. – Picture of dotted graph paper seen by ALF camera for spatial calibration

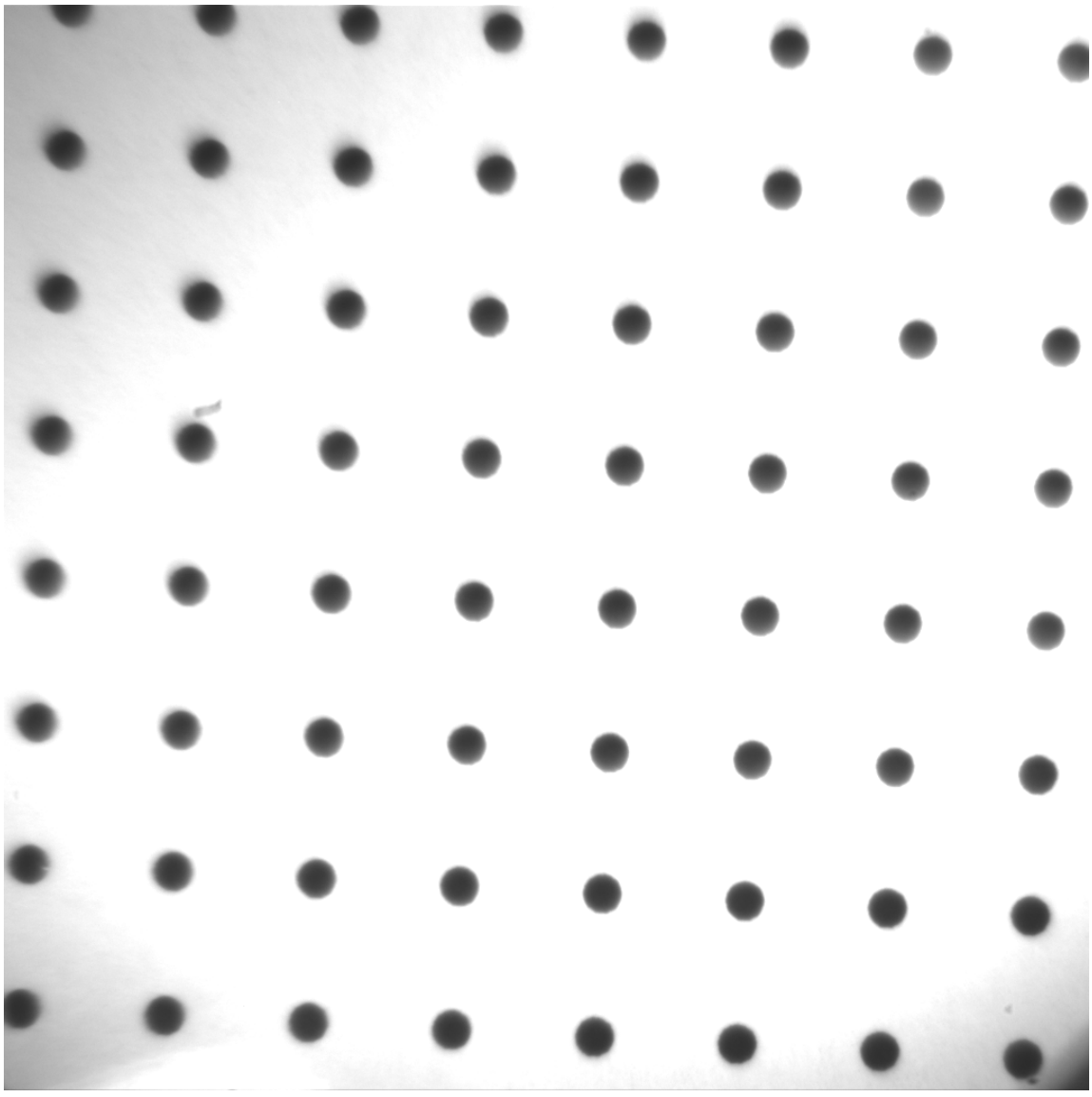


Figure 8. – Picture of dotted graph paper seen by reference camera for spatial calibration

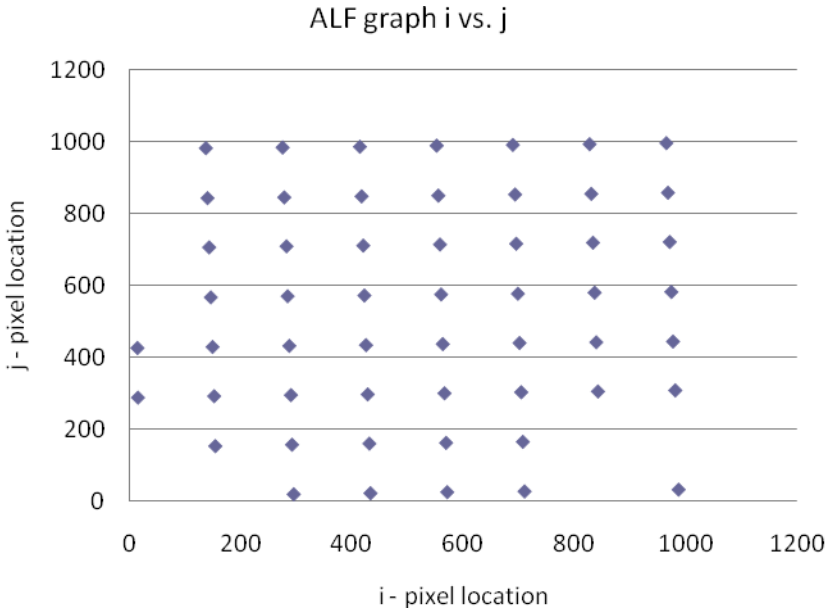


Figure 9. – Centroid locations of dots for ALF camera in terms of pixel location

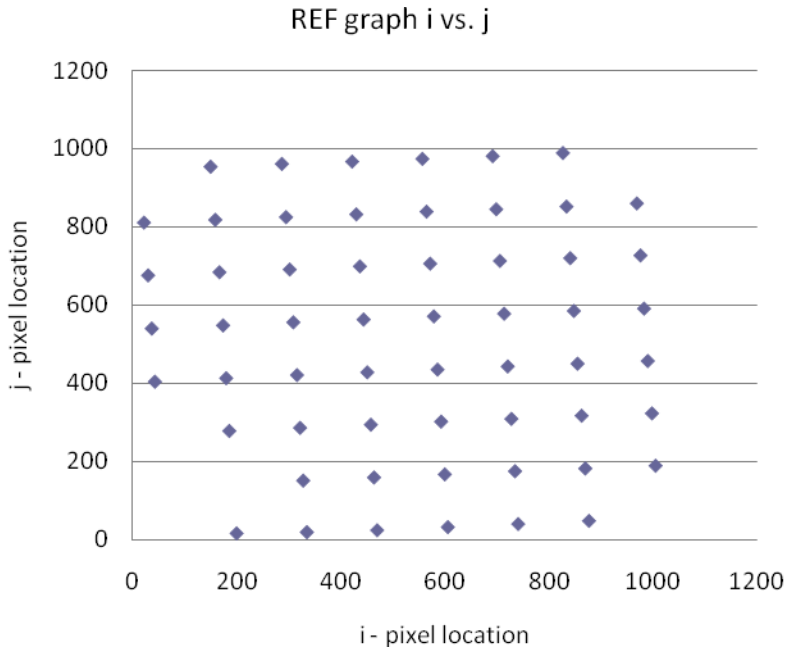


Figure 10. – Centroid locations of dots for REF camera in terms of pixel location

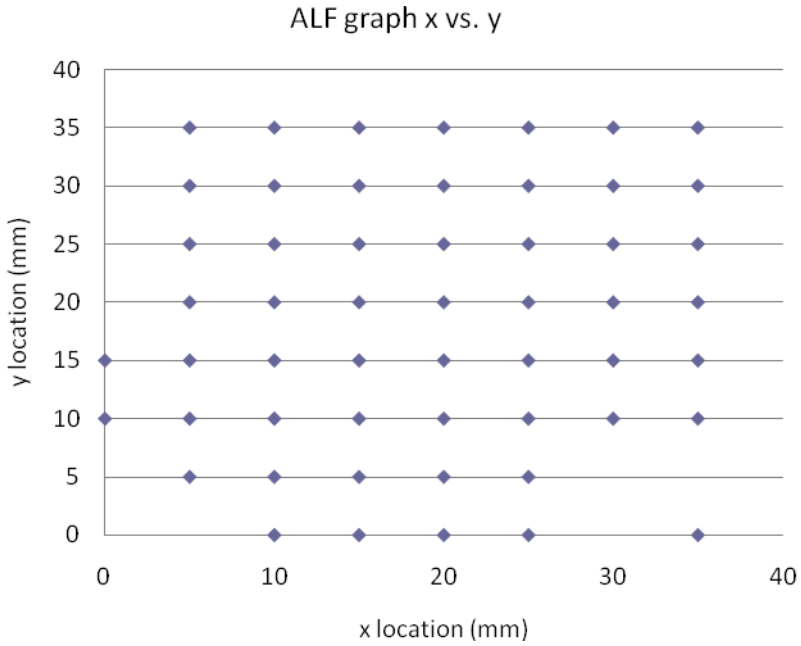


Figure 11. – Centroid locations of dots for ALF camera in terms of physical location

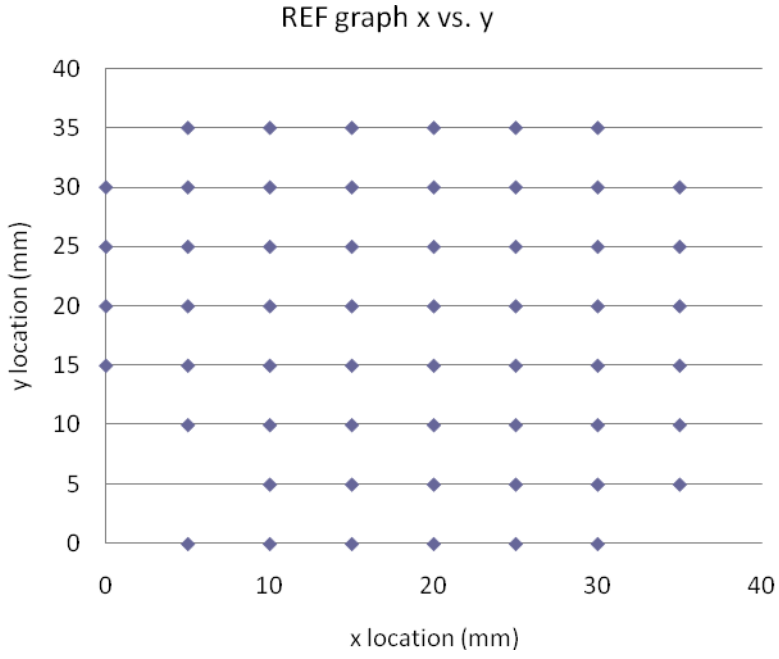


Figure 12. – Centroid locations of dots for REF camera in terms of physical location

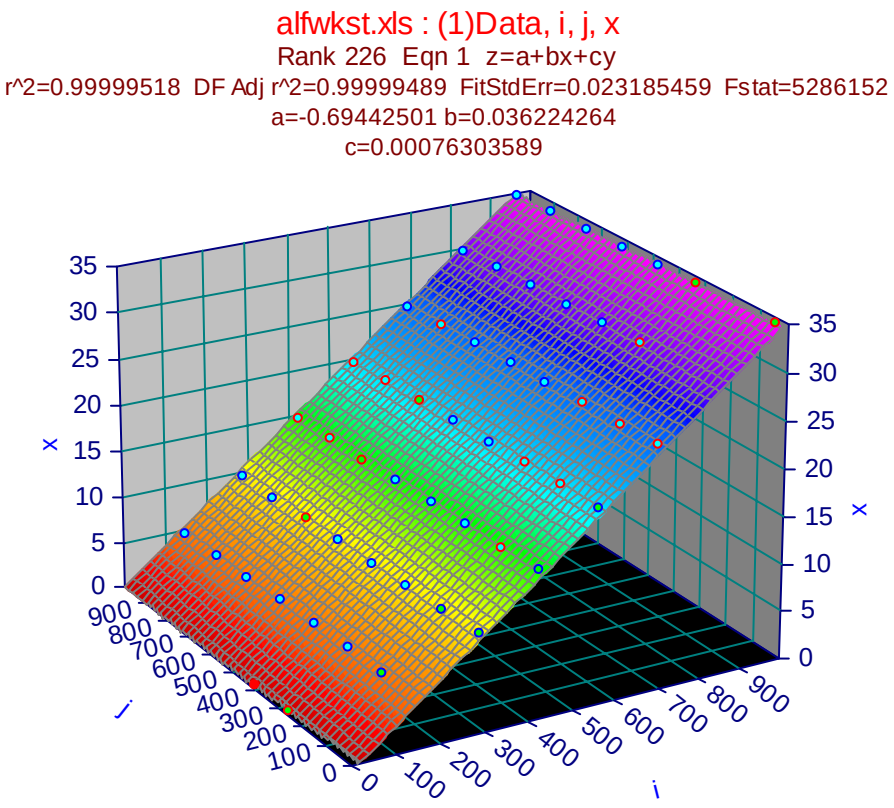


Figure 13. – ALF transformation from (i,j) to x

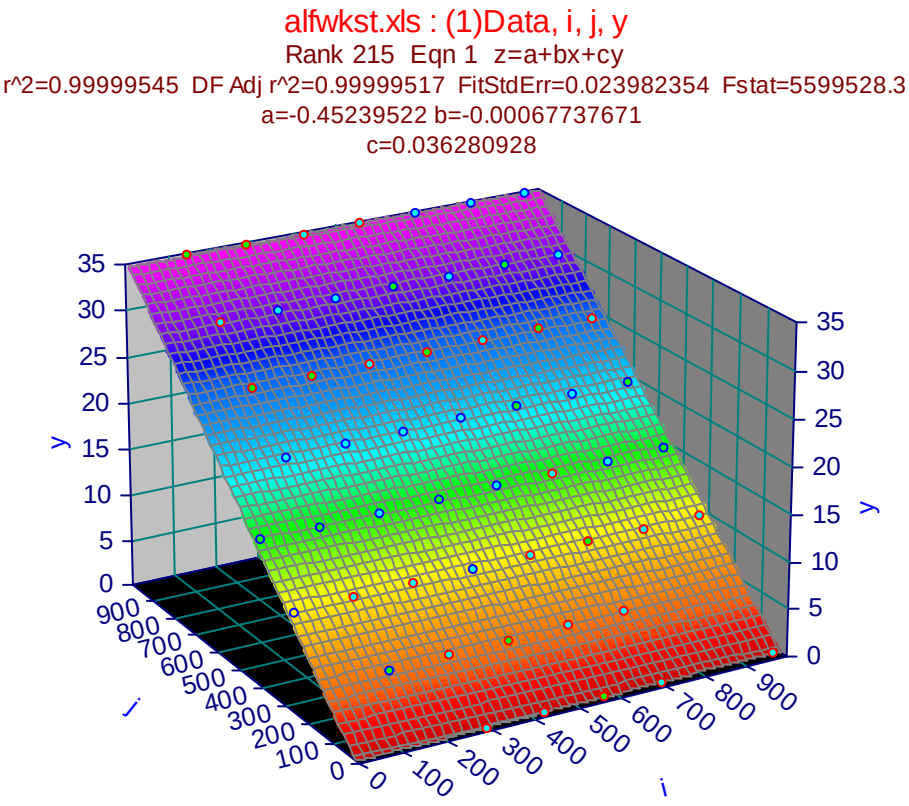


Figure 14. – ALF transformation from (i,j) to y

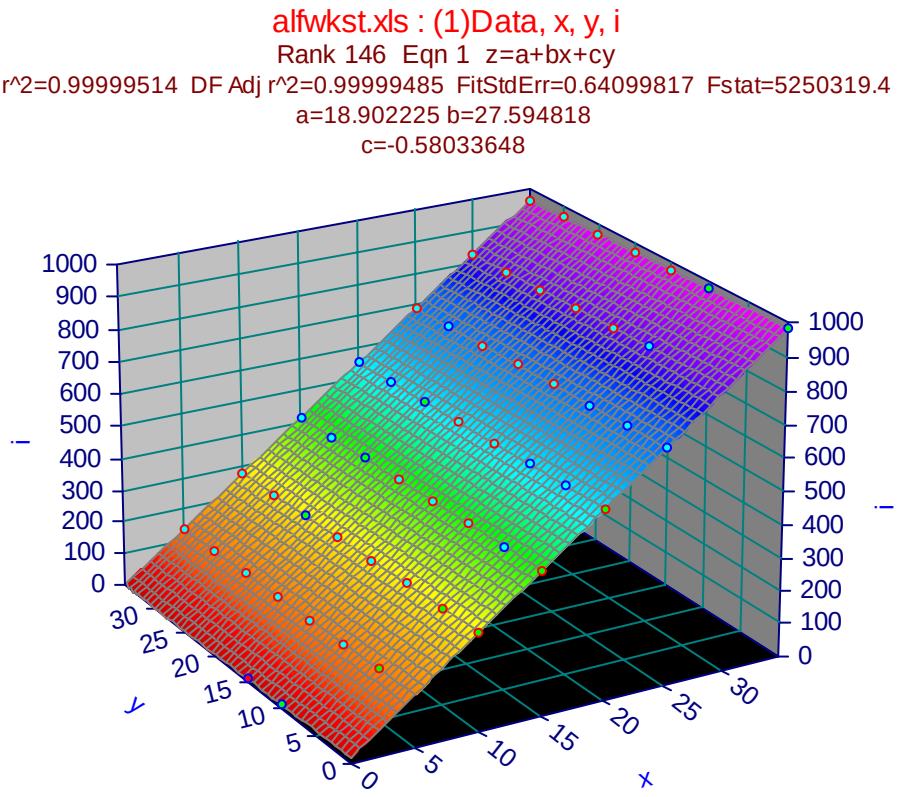


Figure 15. – ALF transformation from (x,y) to i

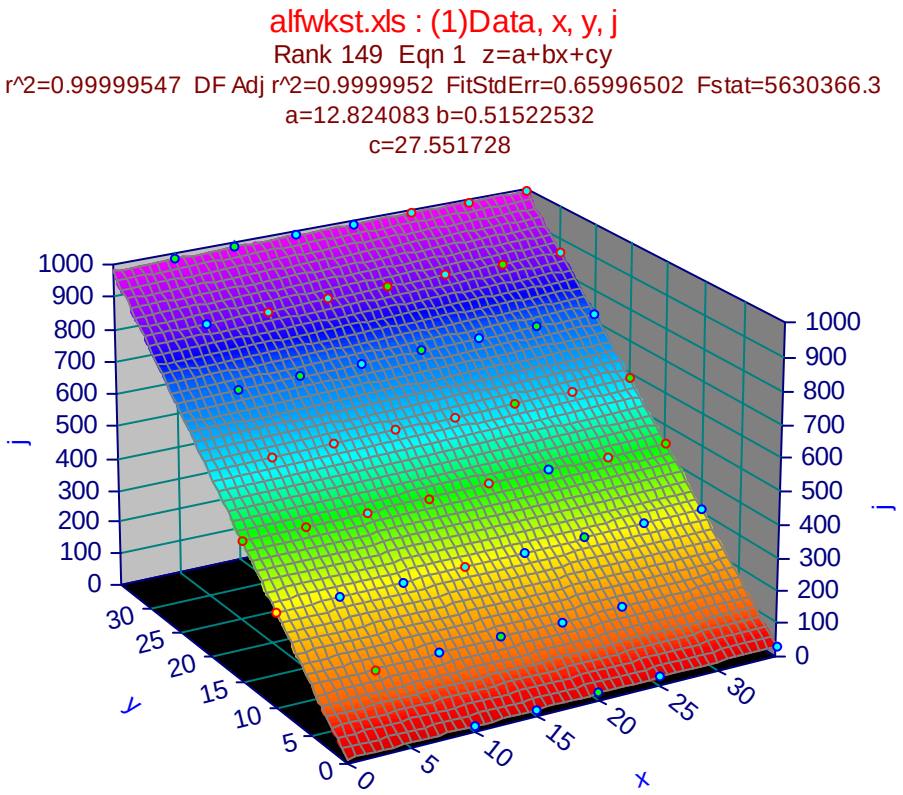


Figure 16. – ALF transformation from (x,y) to j

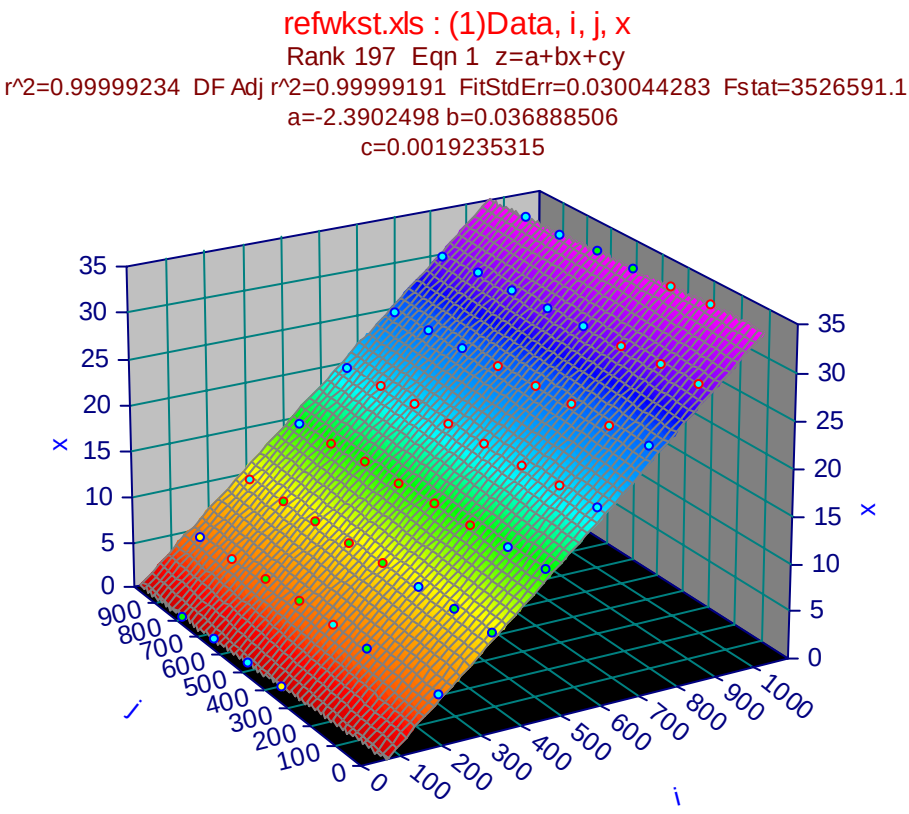


Figure 17. – REF transformation from (i,j) to x

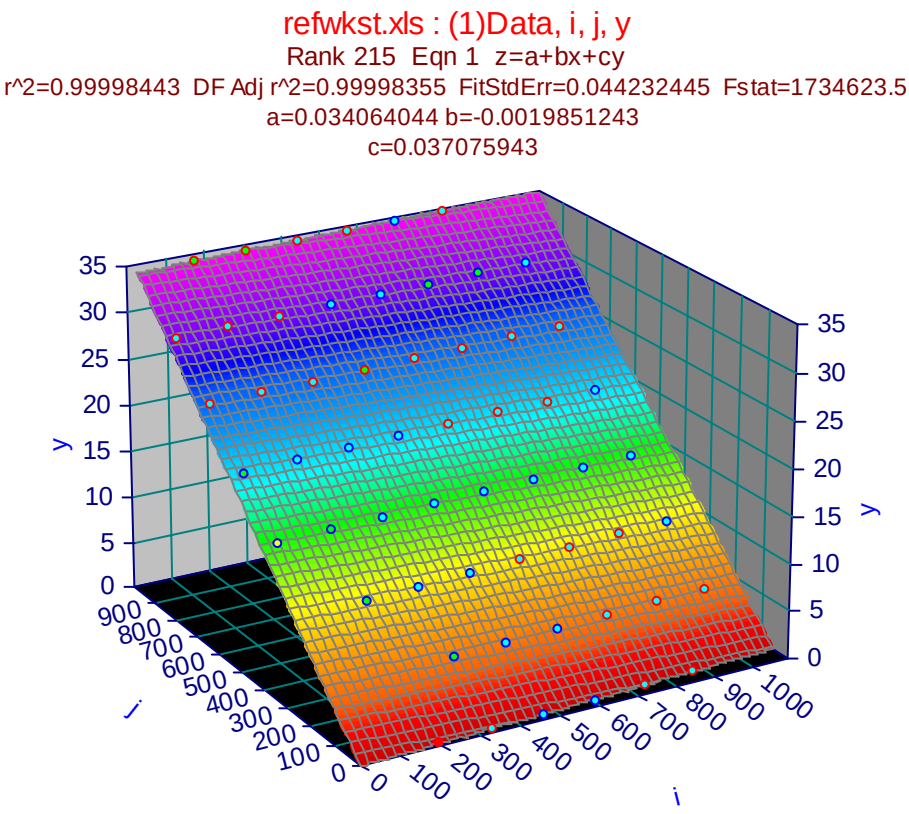


Figure 18. – REF transformation from (i,j) to y

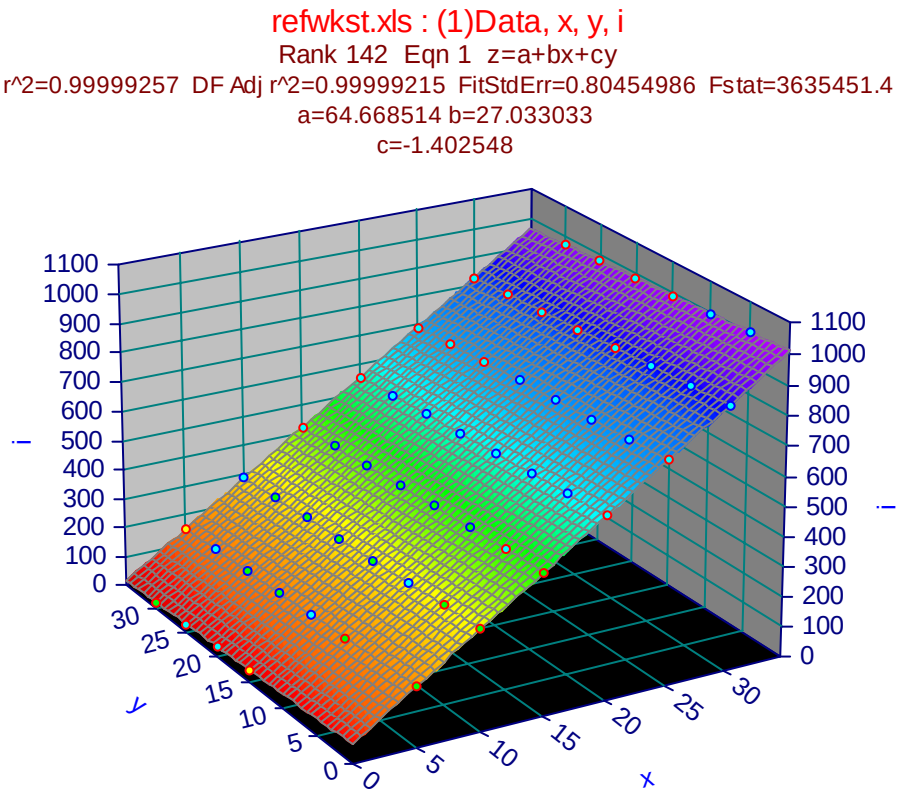


Figure 19. – REF transformation from (x,y) to i

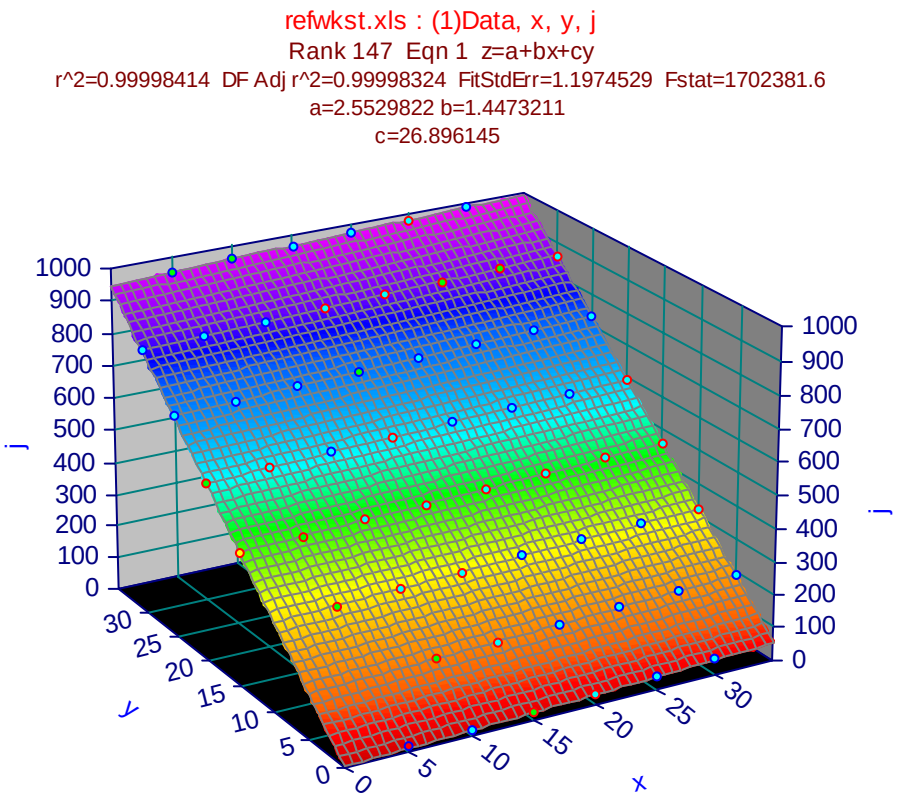


Figure 20. – REF transformation from (x,y) to j

Table 1. – ALF transformation equations

x,y,z	Eq. form	a	b	c
ijx:	z=a+bx+cy	-0.69442501	0.036224264	0.000763036
ijy:	z=a+bx+cy	-0.45239522	-0.00067738	0.036280928
xyi:	z=a+bx+cy	18.9022249	27.59481828	-0.58033648
xyj:	z=a+bx+cy	12.8240833	0.515225318	27.55172832

Table 2. – REF transformation equations

x,y,z	Eq. form	a	b	c
ijx:	z=a+bx+cy	-2.39024982	0.036888506	0.001923531
ijy:	z=a+bx+cy	0.03406404	-0.00198512	0.037075943
xyi:	z=a+bx+cy	64.6685141	27.03303263	-1.40254797
xyj:	z=a+bx+cy	2.55298216	1.44732106	26.89614515

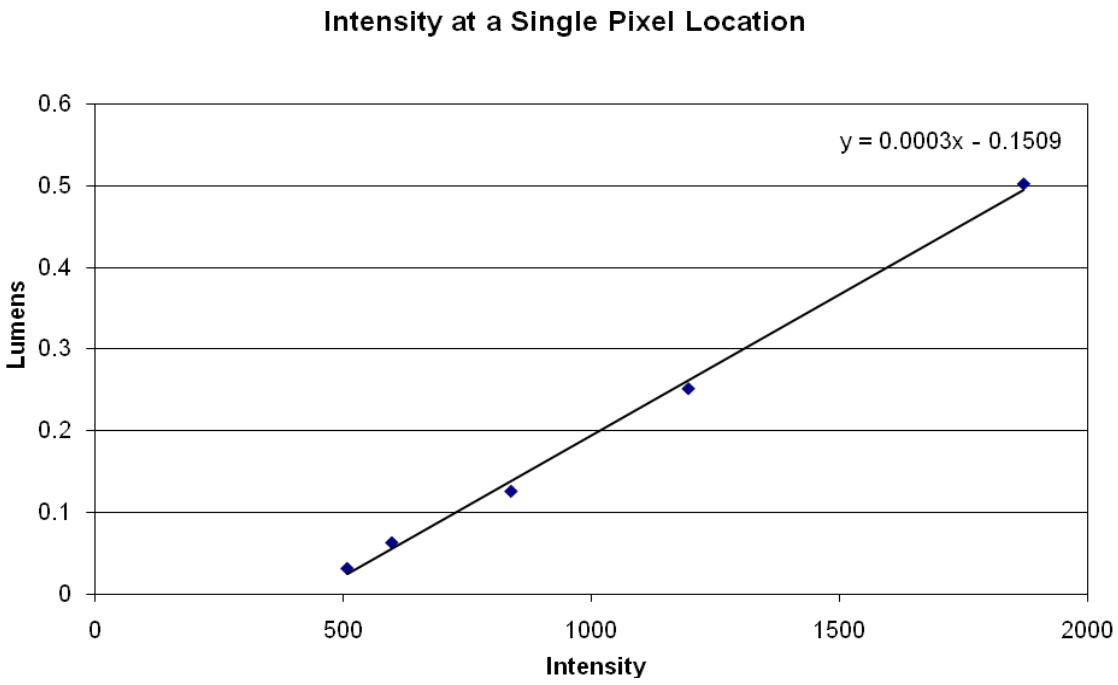


Figure 21. – Example of linear fit relating intensity to ‘lumens’

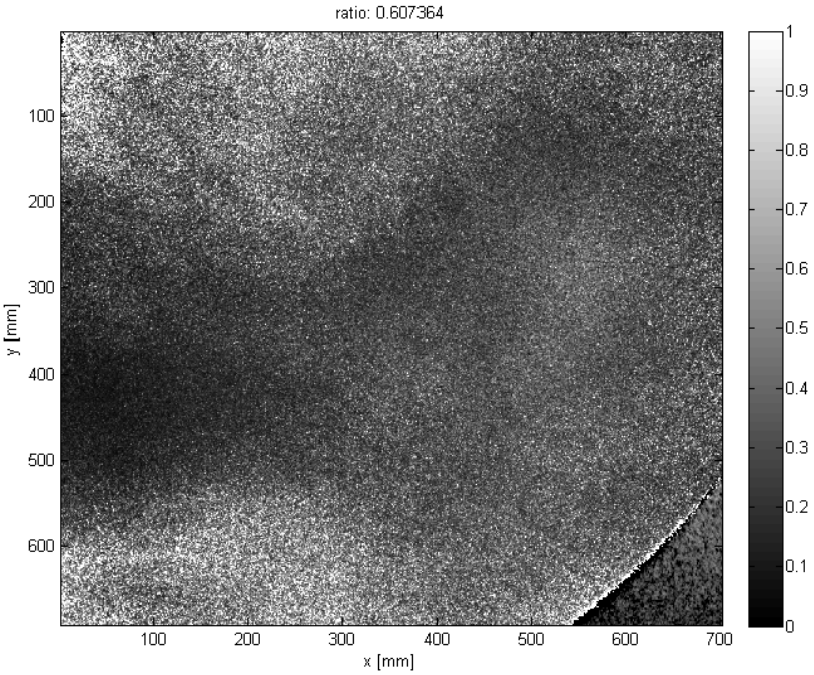


Figure 22. – Example of velocity calibration data

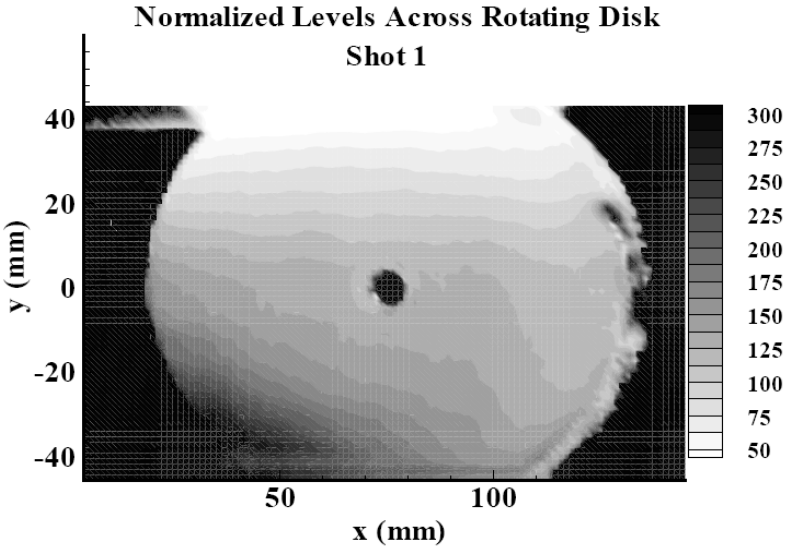


Figure 23. –Velocity calibration data from Gaharan's PhD Dissertation

APPENDIX C

LASER FREQUENCY ANALYSIS

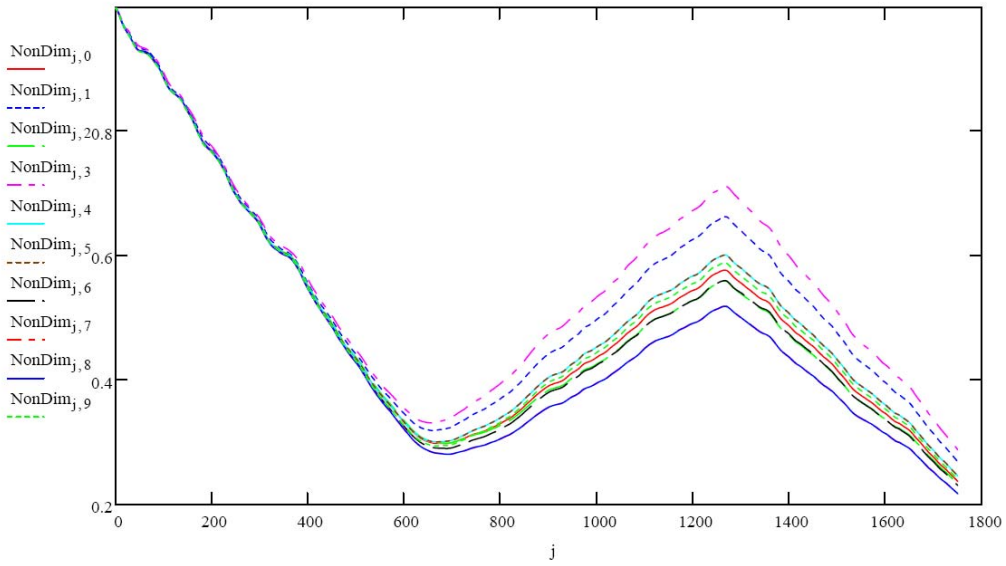


Figure 24. – Nondimensionalized correlation between ALF and REF signals to CCD line camera

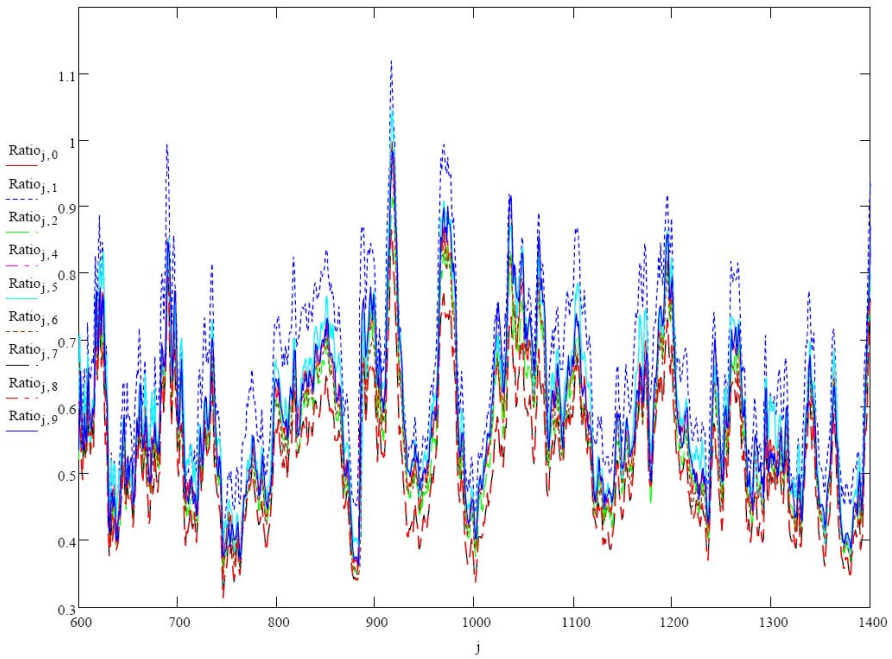


Figure 25. – ALF/REF pixel ratio at point of maximum correlation

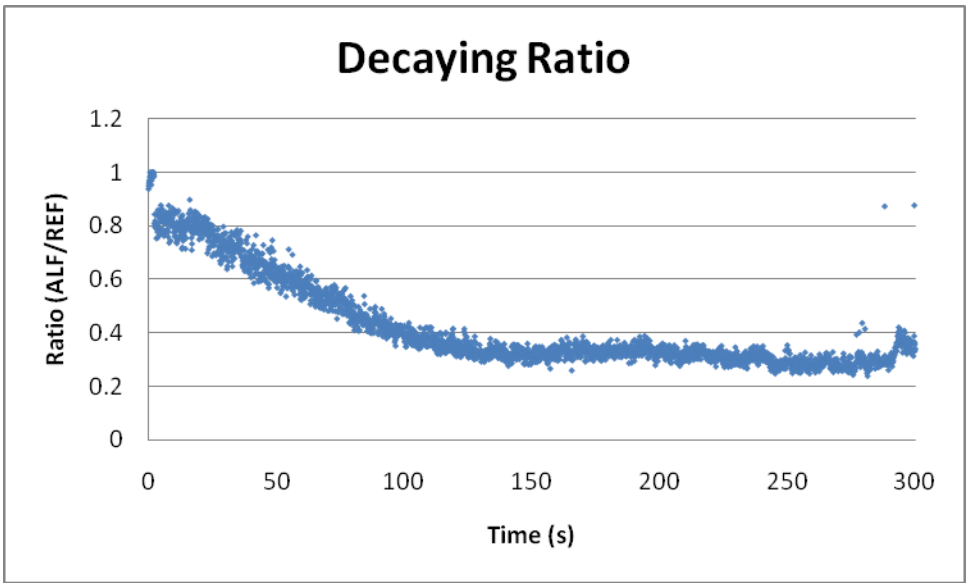


Figure 26. – ALF/REF ratio related to time with no control system

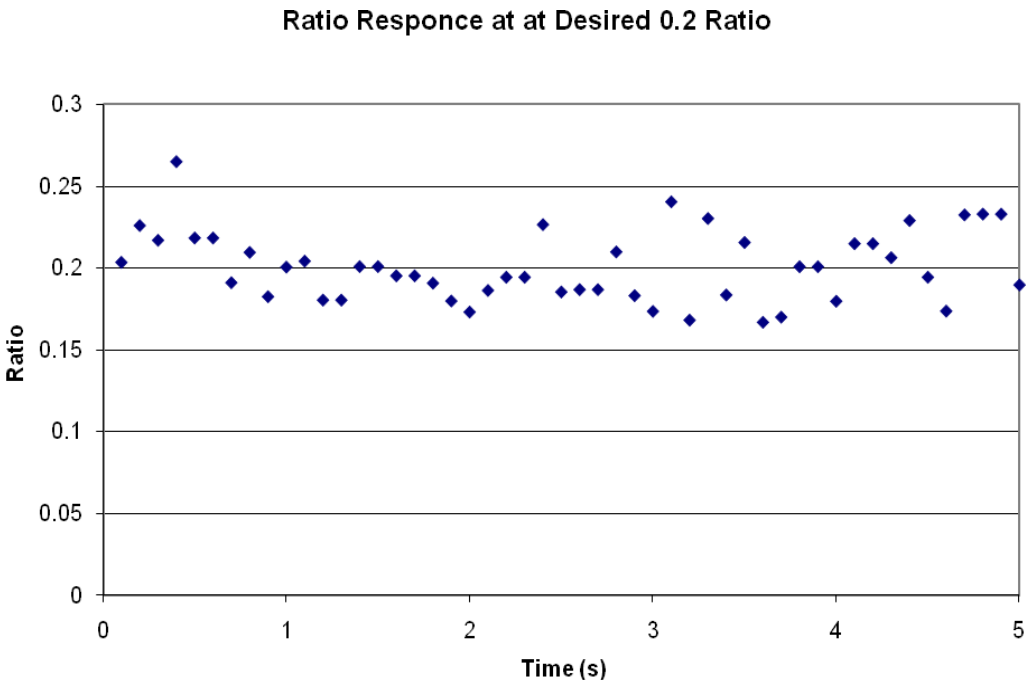


Figure 27. – ALF/REF ratio with active control system

APPENDIX D

LABVIEW PROGRAMS

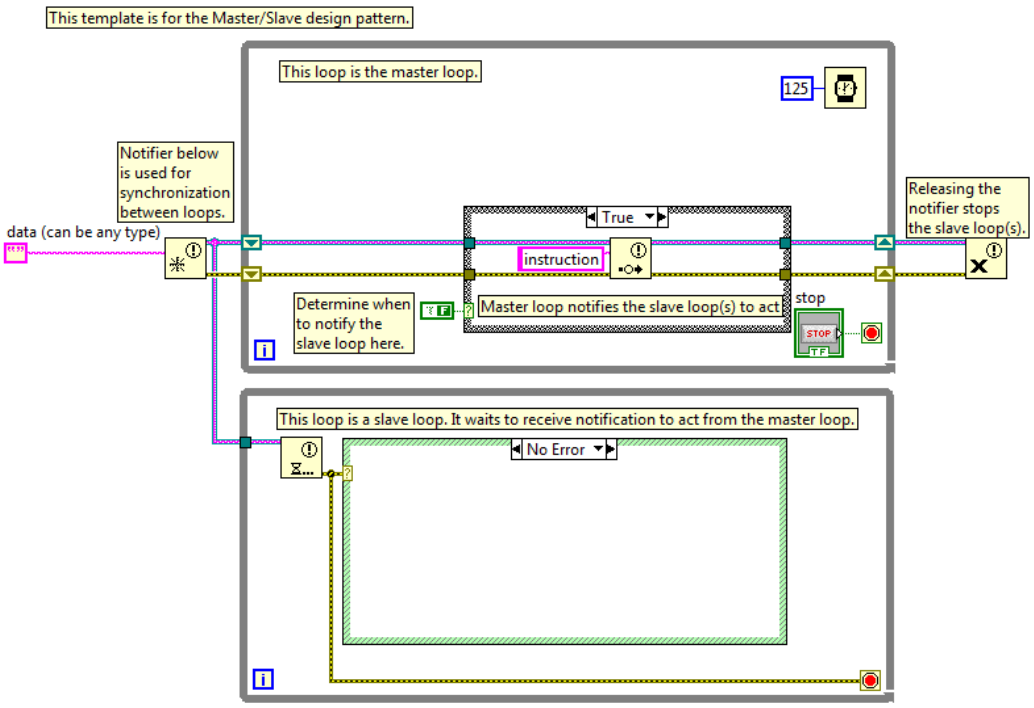


Figure 28. – Master-slave programming format

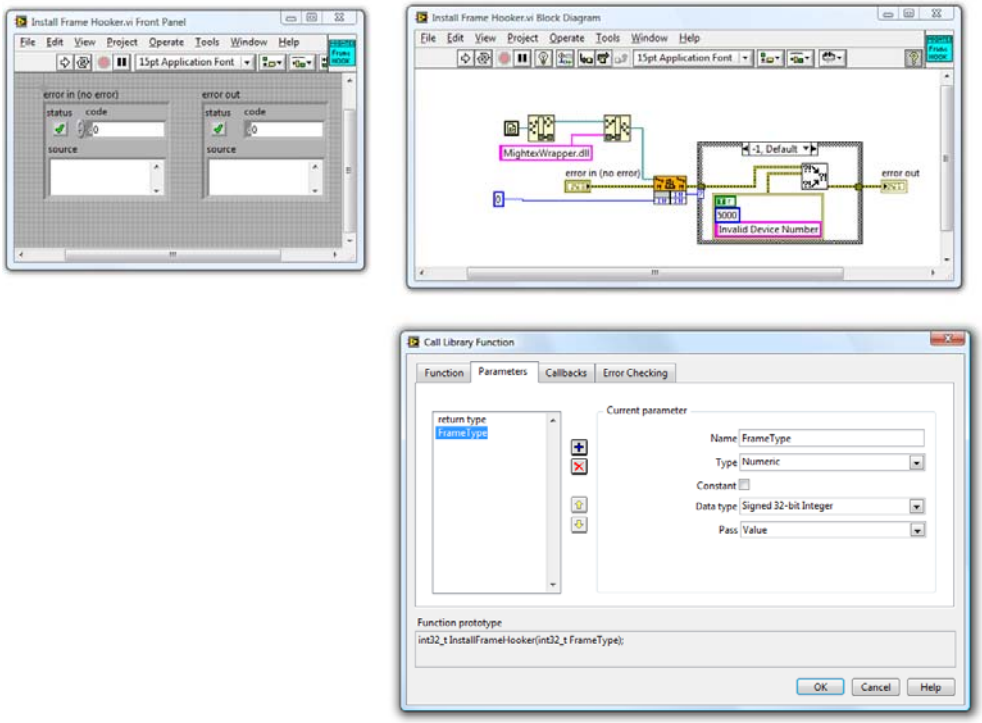


Figure 29. – Example of call library node function

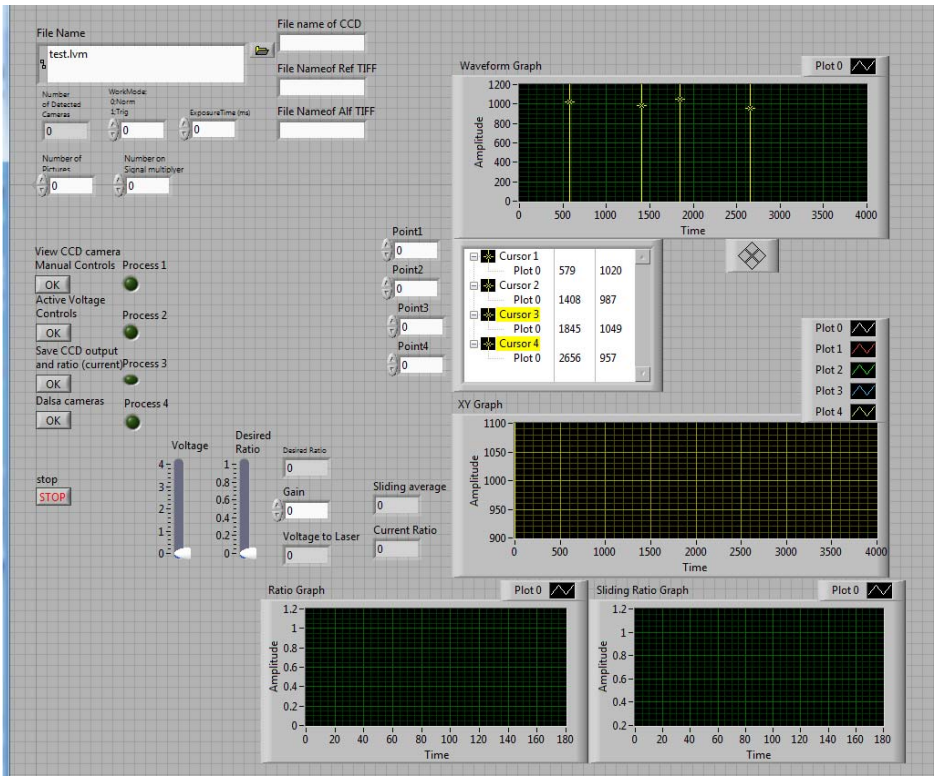


Figure 30. – Control half of SLPLLring5.vi front panel

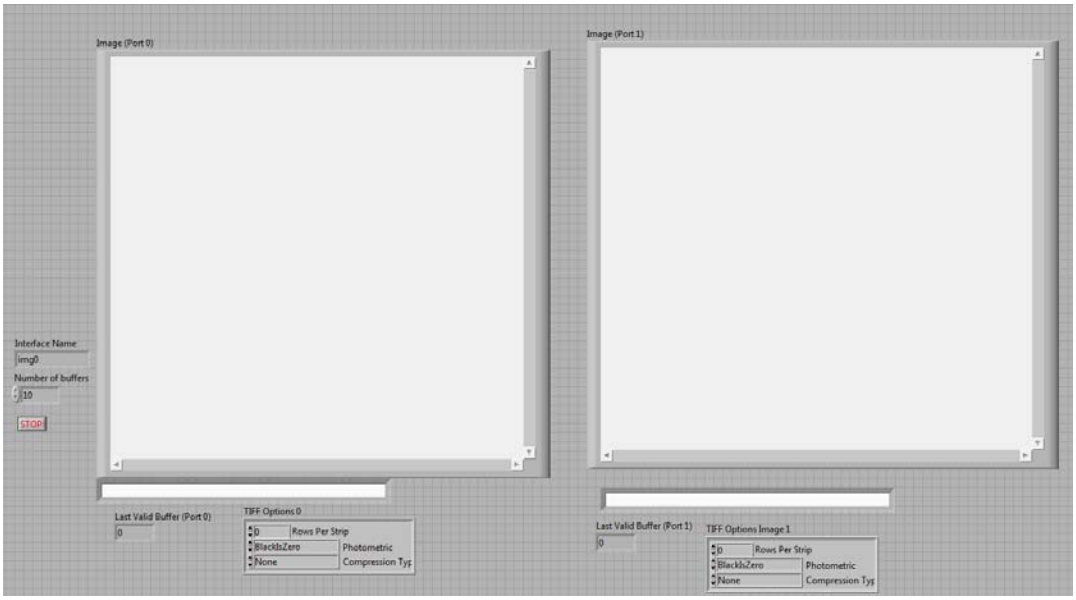


Figure 31. – Dalsa image ports of SLPLLring5.vi front panel

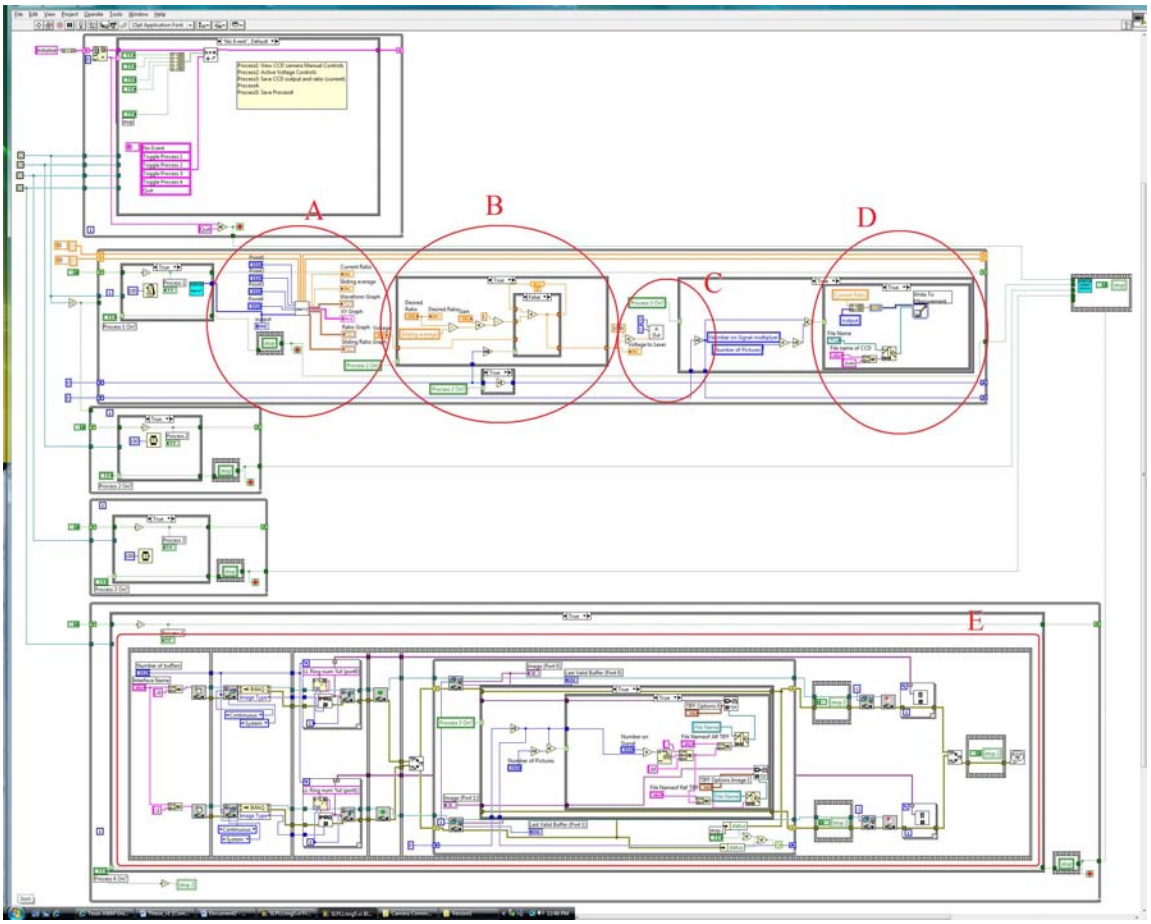


Figure 32. – Block diagram of SLPLLring5.vi with point of interest

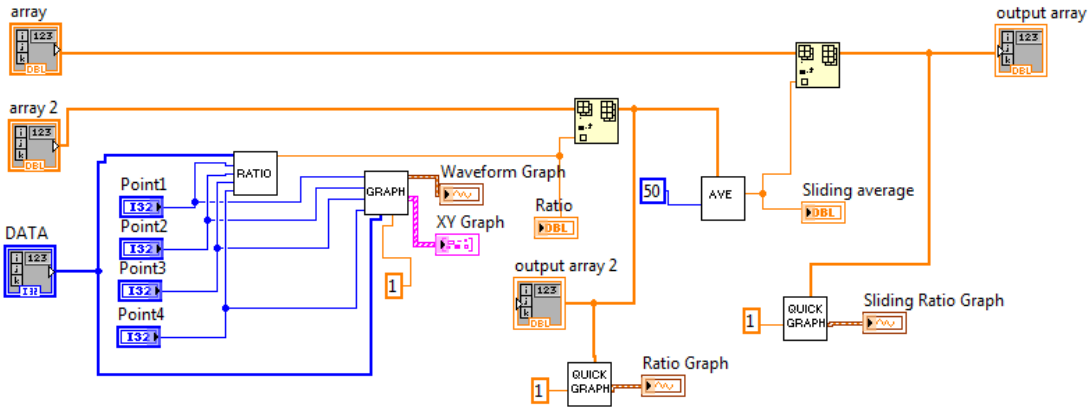


Figure 33. – Block diagram of analysis function (point of interest A)

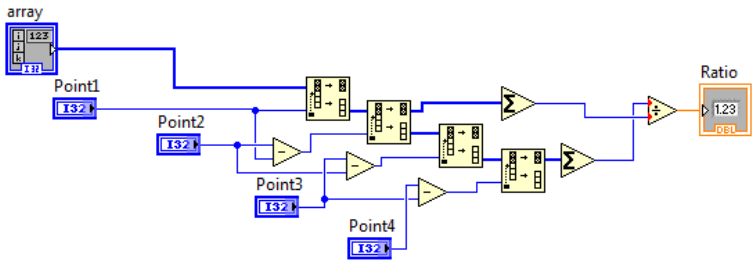


Figure 34. – Block diagram of ratio function

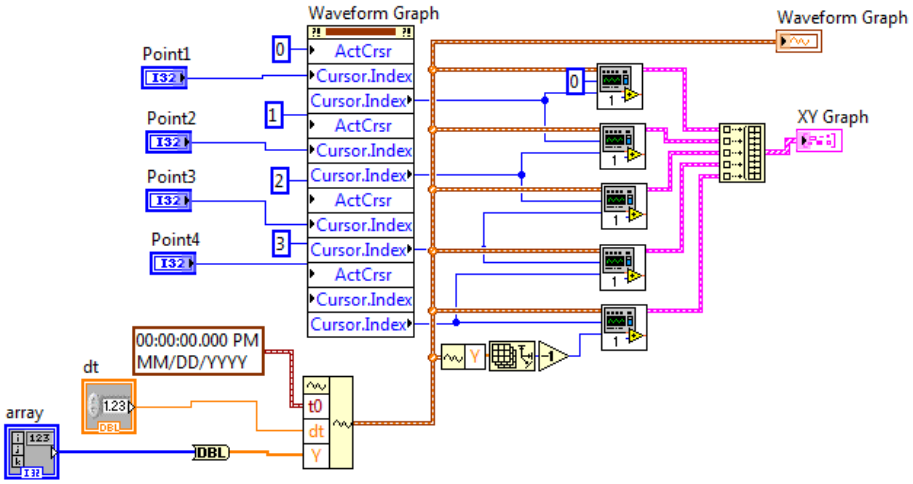


Figure 35. – Block diagram of graph function

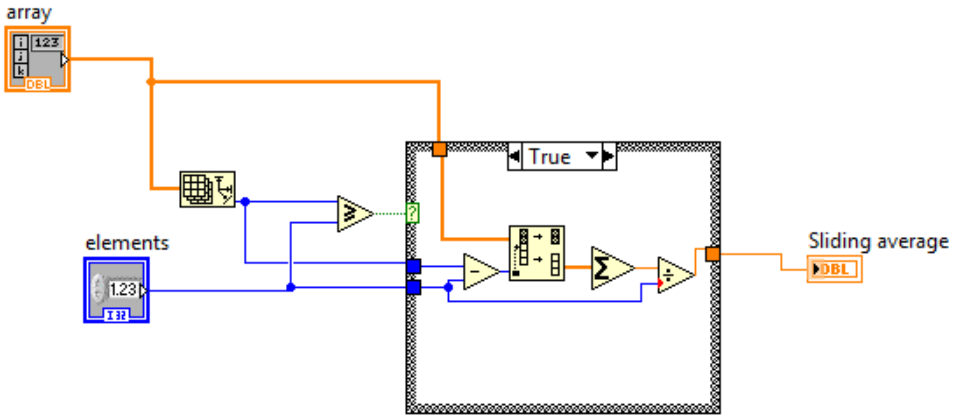


Figure 36. – Block diagram of sliding average function

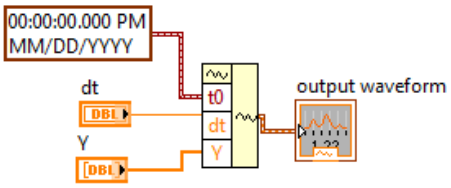


Figure 37. – Block diagram of quick graph function

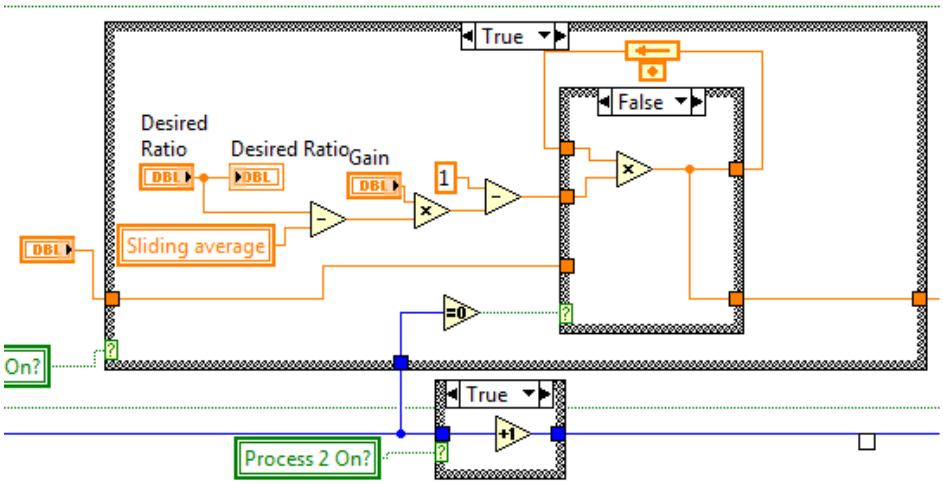


Figure 38. – Block diagram of control scheme (point of interest B)

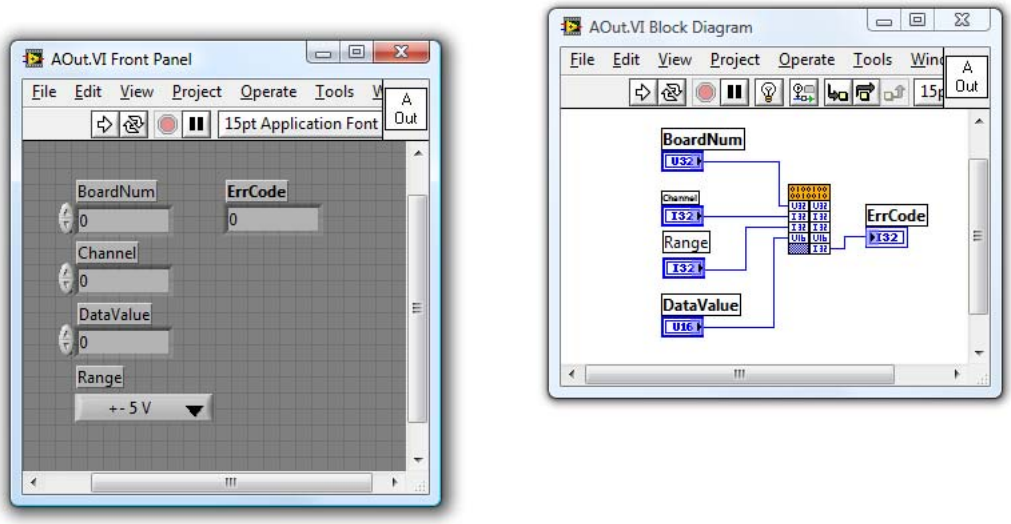


Figure 39. – AOut.vi function (point of interest C)

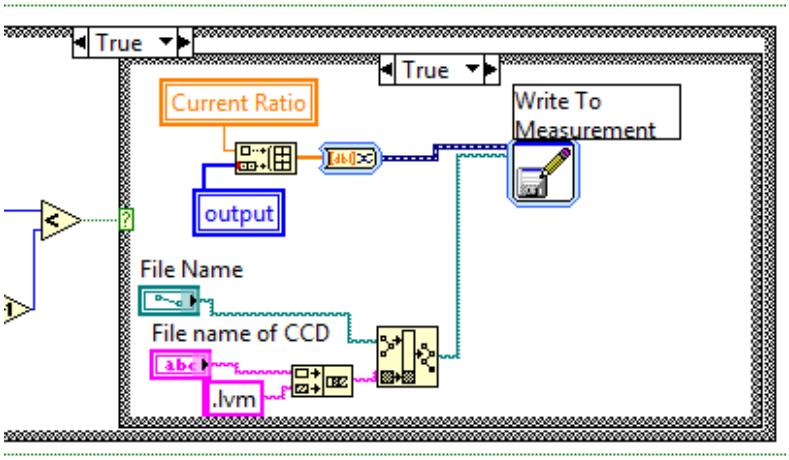


Figure 40. – CCD line camera save block diagram (point of interest D)

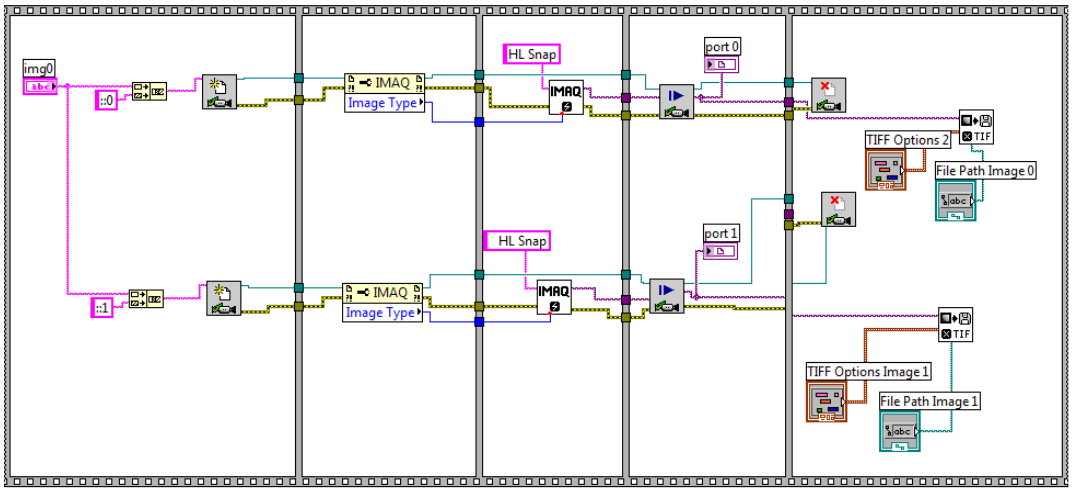


Figure 41. – HL Snap4.tif.vi (point of interest E)

APPENDIX E

C++ PROGRAM

Source Files:

MightexWrapper.cpp

```
// MightexWrapper.cpp : Defines the initialization routines for the
DLL.
//

#include "stdafx.h"
#include "MightexWrapper.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#endif

unsigned short frameData[3648];

//
//TODO: If this DLL is dynamically linked against the MFC DLLs,
//          any functions exported from this DLL which call into
//          MFC must have the AFX_MANAGE_STATE macro added at the
//          very beginning of the function.
//
//          For example:
//
//          extern "C" BOOL PASCAL EXPORT ExportedFunction()
//          {
//                AFX_MANAGE_STATE(AfxGetStaticModuleState());
//                // normal function body here
//          }
//
//          It is very important that this macro appear in each
//          function, prior to any calls into MFC.  This means that
//          it must appear as the first statement within the
//          function, even before any object variable declarations
//          as their constructors may generate calls into the MFC
//          DLL.
//
//          Please see MFC Technical Notes 33 and 58 for additional
//          details.
//


// CMightexWrapperApp

BEGIN_MESSAGE_MAP(CMightexWrapperApp, CWinApp)
END_MESSAGE_MAP()


// CMightexWrapperApp construction

CMightexWrapperApp::CMightexWrapperApp()
{
```

```
	// TODO: add construction code here,
	// Place all significant initialization in InitInstance
}


// The one and only CMightexWrapperApp object

CMightexWrapperApp theApp;


void FrameCallBack( int FrameType, int Row, int Col,
                              TProcessedDataProperty* Attributes,
unsigned char *Frameptr );


// CMightexWrapperApp initialization

BOOL CMightexWrapperApp::InitInstance()
{
	CWinApp::InitInstance();

	return TRUE;
}
void FrameCallBack( int FrameType, int Row, int Col,
                              TProcessedDataProperty* Attributes,
unsigned char *Frameptr )
{
	int i;
	//short frameData[3648];
	short *p;

	if ( Attributes->CameraID == 1 ) // The working camera
	{
		// All frame related attributes is in Attributes.
		// All frame data is pointed by Frameptr
		p = (short *)Frameptr;
		for ( i=0; i< 3647; i++)
		{
			frameData[i] = *p;
			p++;
		}
	}
//	frameData = Frameptr;

}
int CMightexWrapperApp::InitDevice( void )
{
	return CCDUSB_InitDevice();
}

int CMightexWrapperApp::StartCameraEngine( HWND ParentHandle )
{
	return CCDUSB_StartCameraEngine( ParentHandle );
}
```

```
int CMightexWrapperApp::InstallFrameHooker( int FrameType )
{
      return CCDUSB_InstallFrameHooker( FrameType, FrameCallBack );
}

int CMightexWrapperApp::AddDeviceToWorkingSet( int DeviceID )
{
      return CCDUSB_AddDeviceToWorkingSet( DeviceID );
}

int CMightexWrapperApp::RemoveDeviceFromWorkingSet( int DeviceID )
{
      return CCDUSB_RemoveDeviceFromWorkingSet( DeviceID );
}
int CMightexWrapperApp::StopCameraEngine( void )
{
      return CCDUSB_StopCameraEngine();
}
int CMightexWrapperApp::SetCameraWorkMode( int DeviceID, int WorkMode )
{
      return CCDUSB_SetCameraWorkMode( DeviceID, WorkMode );
}
int CMightexWrapperApp::StartFrameGrab( void )
{
      return CCDUSB_StartFrameGrab( 0x8888 );
}
int CMightexWrapperApp::StopFrameGrab( void )
{
      return CCDUSB_StopFrameGrab();
}
int CMightexWrapperApp::SetExposureTime( int DeviceID, int
ExposureTime)
{
      return CCDUSB_SetExposureTime( DeviceID, ExposureTime, false );
}
int CMightexWrapperApp::UnInitDevice( void )
{
      return CCDUSB_UnInitDevice();
}
void CMightexWrapperApp::Import(double *input, int input_length, int
*output)
{
  int i;

  /* Calculate the floor of the square of each value. */
  for(i = 0; i < input_length; i++)
  {
    output[i] = (int)frameData[i];
  }
}
```

MightexWrapper.def

```
; MightexWrapper.def : Declares the module parameters for the DLL.

LIBRARY      "MightexWrapper"

EXPORTS

InitDevice
StartCameraEngine
InstallFrameHooker
AddDeviceToWorkingSet
RemoveDeviceFromWorkingSet
StopCameraEngine
SetCameraWorkMode
StartFrameGrab
StopFrameGrab
SetExposureTime
UnInitDevice
Import

    ; Explicit exports can go here
```

stdafx.cpp

```
// stdafx.cpp : source file that includes just the standard includes
// MightexWrapper.pch will be the pre-compiled header
// stdafx.obj will contain the pre-compiled type information

#include "stdafx.h"
```

Header Files:

CCD_USBCamera_SDK.h

```
// The following ifdef block is the standard way of creating macros
//which make exporting a DLL simpler. All files within this DLL are
//compiled with the MTUSBDLL_EXPORTS symbol defined on the command
//line. this symbol should not be defined on any project that uses this
//DLL. This way any other project whose source files include this file
//see MTUSBDLL_API functions as being imported from a DLL, wheras this
//DLL sees symbols defined with this macro as being exported.

typedef int SDK_RETURN_CODE;
typedef unsigned int DEV_HANDLE;

#ifdef SDK_EXPORTS
#define SDK_API extern "C" __declspec(dllexport) SDK_RETURN_CODE _cdecl
```

```
#define SDK_HANDLE_API extern "C" __declspec(dllexport) DEV_HANDLE
_cdecl
#else
#define SDK_API extern "C" __declspec(dllimport) SDK_RETURN_CODE _cdecl
#define SDK_HANDLE_API extern "C" __declspec(dllimport) DEV_HANDLE
_cdecl
#endif

#define GRAB_FRAME_FOREVER    0x8888

#pragma pack(1)

typedef struct {
      int Revision;
    // For Image Capture
    int Resolution;
    int ExposureTime;
    // GPIO Control
    unsigned char GpioConfigByte; // Config for Input/Output for each
pin.
    unsigned char GpioCurrentSet; // For output Pins only.
} TImageControl;

typedef struct {
    int CameraID;
    int ExposureTime;
    int TimeStamp;
    int TriggerOccurred;
    int TriggerEventCount;
      int OverSaturated;
} TProcessedDataProperty;

#pragma pack()

typedef TImageControl *PImageCtl;

typedef void (* DeviceFaultCallBack)( int DeviceType );
typedef void (* FrameDataCallBack)(int FrameType, int Row, int Col,

TProcessedDataProperty* Attributes, unsigned char *BytePtr );

// Export functions:
SDK_API CCDUSB_InitDevice( void );
SDK_API CCDUSB_UnInitDevice( void );
SDK_API CCDUSB_GetModuleNoSerialNo( int DeviceID, char *ModuleNo, char
*SerialNo);
SDK_API CCDUSB_AddDeviceToWorkingSet( int DeviceID );
SDK_API CCDUSB_RemoveDeviceFromWorkingSet( int DeviceID );
SDK_API CCDUSB_StartCameraEngine( HWND ParentHandle );
SDK_API CCDUSB_StopCameraEngine( void );
SDK_API CCDUSB_SetCameraWorkMode( int DeviceID, int WorkMode );
SDK_API CCDUSB_StartFrameGrab( int TotalFrames );
SDK_API CCDUSB_StopFrameGrab( void );
SDK_API CCDUSB_ShowFactoryControlPanel( int DeviceID, char *passWord );
```

```
SDK_API CCDUSB_HideFactoryControlPanel( void );
SDK_API CCDUSB_SetExposureTime( int DeviceID, int ExposureTime, bool
Store );
SDK_API CCDUSB_InstallFrameHooker( int FrameType, FrameDataCallBack
FrameHooker );
SDK_API CCDUSB_InstallUSBDeviceHooker( DeviceFaultCallBack
USBDeviceHooker );
SDK_API CCDUSB_SetGPIOConifg( int DeviceID, unsigned char ConfigByte );
SDK_API CCDUSB_SetGPIOInOut( int DeviceID, unsigned char OutputByte,
                             unsigned char *InputBytePtr );
```

MightexWrapper.h

```
// MightexWrapper.h : main header file for the MightexWrapper DLL
//

#pragma once

#ifndef __AFXWIN_H__
      #error "include 'stdafx.h' before including this file for PCH"
#endif

#include "resource.h"         // main symbols
#include "CCD_USBCamera_SDK.h"


// CMightexWrapperApp
// See MightexWrapper.cpp for the implementation of this class
//

class CMightexWrapperApp : public CWinApp
{
public:
      CMightexWrapperApp();

      //wrapper function declarations
      int InitDevice( void );
      int StartCameraEngine( HWND ParentHandle );
      int InstallFrameHooker( int FrameType );
      int AddDeviceToWorkingSet( int DeviceID );
      int RemoveDeviceFromWorkingSet( int DeviceID );
      int StopCameraEngine( void );
      int SetCameraWorkMode( int DeviceID, int WorkMode );
      int StartFrameGrab( void );
      int StopFrameGrab( void );
      int SetExposureTime( int DeviceID, int ExposureTime);
      int UnInitDevice( void );
      void Import (double *input, int input_length, int *output);
```

```
// Overrides
public:
	virtual BOOL InitInstance();


	DECLARE_MESSAGE_MAP()
};
```

Resource.h

```
//{{NO_DEPENDENCIES}}
// Microsoft Visual C++ generated include file.
// Used by MightexWrapper.RC
//

// Next default values for new objects
//
#ifdef APSTUDIO_INVOKED
#ifndef APSTUDIO_READONLY_SYMBOLS

#define _APS_NEXT_RESOURCE_VALUE	1000
#define _APS_NEXT_CONTROL_VALUE		1000
#define _APS_NEXT_SYMED_VALUE		1000
#define _APS_NEXT_COMMAND_VALUE		32771
#endif
#endif
```

stdafx.h

```
// stdafx.h : include file for standard system include files,
// or project specific include files that are used frequently, but
// are changed infrequently

#pragma once

#ifndef VC_EXTRALEAN
#define VC_EXTRALEAN		// Exclude rarely-used stuff from Windows
headers
#endif

// Modify the following defines if you have to target a platform prior
to the ones specified below.
// Refer to MSDN for the latest info on corresponding values for
different platforms.
#ifndef WINVER				// Allow use of features specific
to Windows XP or later.
#define WINVER 0x0501		// Change this to the appropriate value
to target other versions of Windows.
```

```
#endif

#ifndef _WIN32_WINNT          // Allow use of features specific to
Windows XP or later.
#define _WIN32_WINNT 0x0501   // Change this to the appropriate value
to target other versions of Windows.
#endif

#ifndef _WIN32_WINDOWS        // Allow use of features specific to
Windows 98 or later.
#define _WIN32_WINDOWS 0x0410 // Change this to the appropriate value
to target Windows Me or later.
#endif

#ifndef _WIN32_IE             // Allow use of features specific to IE
6.0 or later.
#define _WIN32_IE 0x0600      // Change this to the appropriate value
to target other versions of IE.
#endif

#define _ATL_CSTRING_EXPLICIT_CONSTRUCTORS      // some CString
constructors will be explicit

#include <afxwin.h>         // MFC core and standard components
#include <afxext.h>         // MFC extensions

#ifndef _AFX_NO_OLE_SUPPORT
#include <afxole.h>         // MFC OLE classes
#include <afxodlgs.h>       // MFC OLE dialog classes
#include <afxdisp.h>        // MFC Automation classes
#endif // _AFX_NO_OLE_SUPPORT

#ifndef _AFX_NO_DB_SUPPORT
#include <afxdb.h>                  // MFC ODBC database classes
#endif // _AFX_NO_DB_SUPPORT

#ifndef _AFX_NO_DAO_SUPPORT
#include <afxdao.h>                 // MFC DAO database classes
#endif // _AFX_NO_DAO_SUPPORT

#ifndef _AFX_NO_OLE_SUPPORT
#include <afxdtctl.h>         // MFC support for Internet Explorer 4
Common Controls
#endif
#ifndef _AFX_NO_AFXCMN_SUPPORT
#include <afxcmn.h>                 // MFC support for Windows Common
Controls
#endif // _AFX_NO_AFXCMN_SUPPORT
```

APPENDIX F

MIGHTEX CCD LINE CAMERA SDK

(In Canada)
2343 Brimley Road
Suite 868
Toronto, Ontario M1S 3L6
CANADA
Tel: 1-416-840 4991
Fax: 1-416-840 6541

(In US)
1032 Serpentine Lane
Suite 113
Pleasanton, CA 94566
USA
Tel: 1-925-218 1885
Email: sales@mightex.com

Mightex CCD Line Camera SDK Manual

Version 1.0.0

Mar. 13, 2007

Relevant Products

Part Numbers
TCD-1304-U

Revision History

Revision	Date	Author	Description
1.0.0	Mar. 13, 2007	JT Zheng	Initial Revision

Mightex USB 2.0 CCD Line camera is designed for low cost spectrometer and machine vision applications, With USB 2.0 high speed interface and powerful PC camera engine, the camera delivers CCD Line image data at high frame rate. GUI demostration application and SDK are provided for user's application developments.

IMPORTANT:

Mightex USB Camera is using USB 2.0 for data collection, USB 2.0 hardware MUST be present on user's PC and Mightex device driver MUST be installed properly before developing application with SDK. **For installation of Mightex device driver, please refer to *Mightex CCD Line Camera User Manual*.**

SDK FILES:

The SDK includes the following files:

\LIB directory:

CCD_USBCamera_SDK.h	--- Header files for all data prototypes and dll export functions.
CCD_USBCamera_SDK.dll	--- DLL file exports functions.
CCD_USBCamera_SDK.lib	--- Import lib file, user may use it for VC++ development.
LinearCameraUsblib.dll	--- DLL file used by "CCD_USBCamera_SDK.dll" .

\Documents directory:

MighTex CCD Line Camera SDK Manual.pdf

\Examples directory

\Delphi --- Delphi project example.
\VC++ --- VC++ 6.0 project example.

\Firmware directory: The latest firmware.

Note

1). The Camera engine supports Multiple Cameras, user may invoke functions to get the number of cameras currently present on USB and add the subset of the cameras into current "Working Set", all the cameras are in "Working Set" are active cameras which camera engine will get frames from them.
2). Although Line cameras are USB Devices, camera engine is **NOT** supporting Plug&Play of the cameras, it's NOT recommended to Plug or Unplug cameras while the camera engine is grabbing frames from the cameras.
3). The code examples are for demonstration of the DLL functions only, device fault conditions are not fully handled in these examples, user should handle those error conditions properly.

HEADER FILE:

The "CCD_USBCamera_SDK.h" is as following:

```
typedef int SDK_RETURN_CODE;
typedef unsigned int DEV_HANDLE;

#ifdef SDK_EXPORTS
#define SDK_API extern "C" __declspec(dllexport) SDK_RETURN_CODE _cdecl
#define SDK_HANDLE_API extern "C" __declspec(dllexport) DEV_HANDLE _cdecl
#else
#define SDK_API extern "C" __declspec(dllimport) SDK_RETURN_CODE _cdecl
#define SDK_HANDLE_API extern "C" __declspec(dllimport) DEV_HANDLE _cdecl
#endif

#define GRAB_FRAME_FOREVER                    0x8888

#pragma pack(1)

typedef struct {
                                                int Revision;
   // For Image Capture
   int Resolution;
```

```
    int ExposureTime;
    // GPIO Control
    unsigned char GpioConfigByte; // Config for Input/Output for each pin.
    unsigned char GpioCurrentSet; // For output Pins only.
} TImageControl;

typedef struct {
    int CameraID;
    int ExposureTime;
    int TimeStamp;
    int TriggerOccurred;
    int TriggerEventCount;
    int OverSaturated;
} TProcessedDataProperty;

#pragma pack()

typedef TImageControl *PImageCtl;

typedef void (* DeviceFaultCallBack)( int FaultType );
typedef void (* FrameDataCallBack)(int FrameType, int Row, int Col,
  TProcessedDataProperty* Attributes, unsigned char *BytePtr );

// Export functions:
SDK_API CCDUSB_InitDevice( void );
SDK_API CCDUSB_UnInitDevice( void );
SDK_API CCDUSB_GetModuleNoSerialNo( int DeviceID, char *ModuleNo, char *SerialNo);
SDK_API CCDUSB_AddDeviceToWorkingSet( int DeviceID );
SDK_API CCDUSB_RemoveDeviceFromWorkingSet( int DeviceID );
SDK_API CCDUSB_StartCameraEngine( HWND ParentHandle );
SDK_API CCDUSB_StopCameraEngine( void );
SDK_API CCDUSB_SetCameraWorkMode( int DeviceID, int WorkMode );
SDK_API CCDUSB_StartFrameGrab( int TotalFrames );
SDK_API CCDUSB_StopFrameGrab( void );
SDK_API CCDUSB_ShowFactoryControlPanel( int DeviceID, char *passWord );
SDK_API CCDUSB_HideFactoryControlPanel( void );
SDK_API CCDUSB_SetExposureTime( int DeviceID, int exposureTime, bool Store );
SDK_API CCDUSB_InstallFrameHooker( int FrameType, FrameDataCallBack FrameHooker );
SDK_API CCDUSB_InstallUSBDeviceHooker( DeviceFaultCallBack USBDeviceHooker );
SDK_API CCDUSB_SetGPIOConifg( int DeviceID, unsigned char ConfigByte );
SDK_API CCDUSB_SetGPIOInOut( int DeviceID, unsigned char OutputByte,
                    unsigned char *InputBytePtr );.
```

// Please check the header file itself of latest information, we may add functions from time to time.

Basically, only ONE data structure *TProcessedDataProperty* data structure is defined and used for the all following functions, mainly for frame property. Note that "#pragma (1)" should be used (as above) for the definition of this structure, as DLL expects the variable of this data structure is "BYTE" alignment.

EXPORT Functions:

CCD_USBCamera_SDK.dll exports functions to allow user to easily and completely control the Line camera and get image frame. Part of the features such as firmware version queries, firmware upgrade…etc. are provided by a built-in windows, which is:

Mightex CCD Line Camera Factory Control Window

User may simply invoke **CCDUSB_ShowFactoryControlPanel()** and **CCDUSB_HideFactoryControlPanel()** to show and hide this window.

SDK_API CCDUSB_InitDevice(void);

This is first function user should call for his own application, this function communicates with the installed device driver and reserve resources for further operations.
Arguments: None
Return: The number of Mightex CCD Line cameras currently attached to the USB 2.0 Bus, if there's no Mightex USB camera attached, the return value is 0.
Note: There's **NO** device handle needed for calling further camera related functions, after invoking CCDUSB_InitDevice, camera engine reserves resources for all the attached cameras. For example, if the returned value is 2, which means there TWO cameras currently presented on USB, user may use "1" or "2" as DeviceID to call further device related functions, "1" means the first device and "2" is the second device. By default, all the devices are in "inactive" state, user should invoke *CCDUSB_AddCameraToWorkingSet(deviceID)* to set the camera as active.

SDK_API CCDUSB _UnInitDevice(void);

This is the function to release all the resources reserved by CCDUSB_InitDevice(), user should invoke it before application terminates.
Arguments: None
Return: Always return 0.

SDK_API CCDUSB_GetModuleNoSerialNo(int DeviceID, char *ModuleNo, char *SerialNo);

For a present device, user might get its Module Number and Serial Number by invoking this function.
Argument: DeviceID – the number (ONE based) of the device, Please refer to the notes of **CCDUSB_InitDevice()** function for it.
ModuleNo – the pointer to a character buffer, the buffer should be available for at least 16 characters.
SerialNo – the pointer to a character buffer, the buffer should be available for at least 16 characters.
Return: -1 If the function fails (e.g. invalid device number)
1 if the call succedds.

SDK_API CCDUSB_AddDeviceToWorkingSet(int DeviceID);

For a present device, user might add it to current "Working Set" of the camera engine, and the camera becomes active.
Argument: DeviceID – the number (ONE based) of the device, Please refer to the notes of **CCDUSB_InitDevice()** function for it.
Return: -1 If the function fails (e.g. invalid device number)
1 if the call succedds.
Note: Camera Engine will only grab frames from active cameras (the cameras in "Working Set").

SDK_API CCDUSB_RemoveDeviceFromWorkingSet(int DeviceID);

User might remove the camera from the current "Working Set", after invoking this function, the camera become inactive.
Argument: DeviceID – the number (ONE based) of the device, Please refer to the notes of **CCDUSB_InitDevice()** function for it.
Return: -1 If the function fails (e.g. invalid device number)
1 if the call succedds.
Note: Camera Engine will only grab frames from active cameras (the cameras in "Working Set").

SDK_API CCDUSB_StartCameraEngine(HWND ParentHandle);

We have a multiple threads camera engine internally, which is responsible for all the frame grabbing, raw data processing…etc. functions. User MUST start this engine for all the following camera related operations
Argument: ParentHandle – The window handle of the main form of user's application, as the engine relies on Windows Message Queue, it needs a parent window handle which mostly should be the handle of the main window of user's application.
Return: -1 If the function fails (e.g. invalid device handle)

1 if the call succedds.

SDK_API CCDUSB_StopCameraEngine(void);

This function stops the started camera engine.
Argument: None.
Return: it always returns 1.
Important: For properly operating the cameras, usually the application should have the following sequence for device initialization and opening:

```
CCDUSB_InitDevice();     // Get the devices
CCDUSB_AddCameraToWorkingSet( deviceID);
MTUSB_StartCameraEngine();
…… Operations ……
MTUSB_StopCameraEngin();
MTUSB_UnInitDevice()
```

Note that we don't need to explicitly open and close the opened device, as CCDUSB_InitDevice() will actually open all the current attached cameras and reserve resources for them, however, by default, all of them are inactive, user needs to set them as active by invoking *CCDUSB_AddCameraToWorkingSet(deviceID).*

SDK_API CCDUSB_SetCameraWorkMode(int DeviceID, int WorkMode);

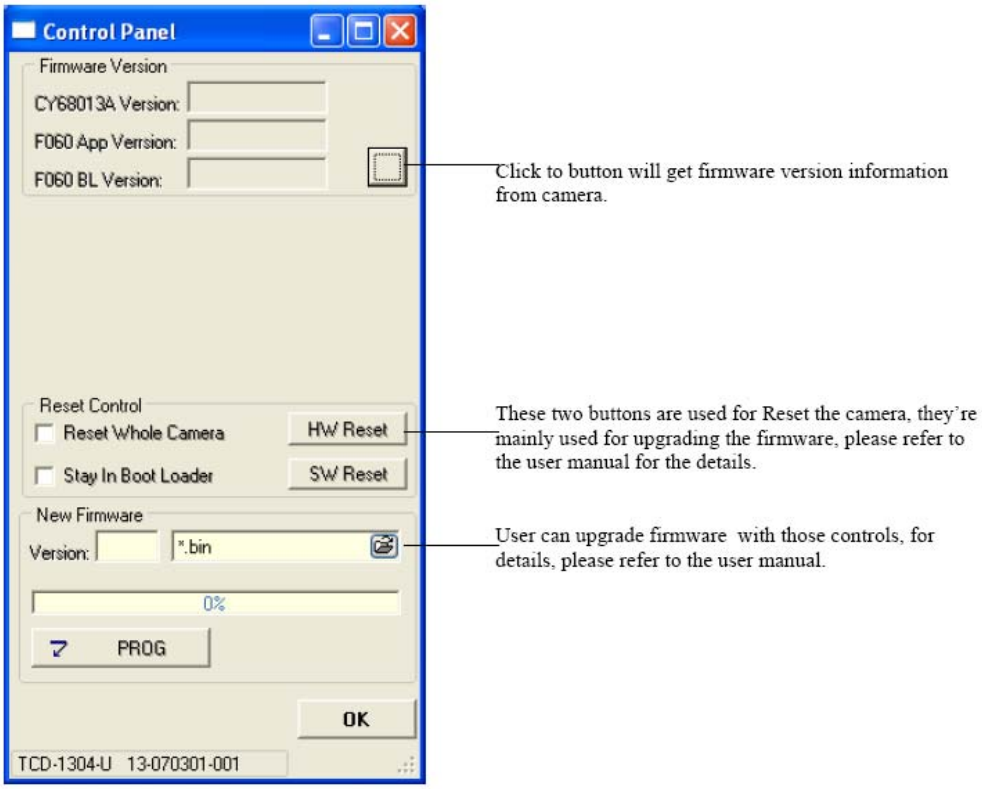
By default, the Camera is working in "**NORMAL**" mode in which camera deliver frames to PC continuously, however, user may set it to **"TRIGGER"** Mode, in which the camera is waiting for an external trigger signal and capture ONE frame for each trigger signal.
Argument: *DeviceID*e – the device number which identifies the camera.
WorkMode – 0: **NORMAL** Mode, 1: **TRIGGER** Mode.
Return: -1 If the function fails (e.g. invalid device number)
1 if the call succeeds.
Note: Basically, **NORMAL** mode and **TRIGGER** mode are the same, the only difference is that:
NORMAL mode – When Camera is in this mode, camera will always grab frame as long as there's available memory for a new frame, while host (in most cases, it's a PC) is continuing get frame from the camera, the camera is continuously grabbing frames from CCD sensor.
TRIGGER mode – When Camera is in this mode, camera will only grab a frame from CCD sensor while there's an external trigger asserted (and there's available memory for a new frame), in this case, host will usually poll the camera for a new coming frame.

SDK_API CCDUSB_StartFrameGrab(int TotalFrames);
SDK_API CCDUSB_StopFrameGrab(void);

When camera engine is started, the engine prepares all the resources, but it does NOT start the frame grabbing , until CCDSB_StartFrameGrab() function is invoked. After it's successfully called, camera engine starts to grab frames from cameras in current "Working Set". User may call CCDUSB_StopFrameGrab() to stop the engine from grabbing frames from camera.
Argument: TotalFrames – This is for CCDUSB_StartFrameGrab() only, after grabbing this frames, the camera engine will automatically stop frame grabbing, if user doesn't want it to be stopped, set this number to 0x8888, which will doing frame grabbing forever (in NORMAL mode), until user calls CCDUSB_StopFrameGrab().
Return: -1 If the function fails (e.g. invalid device number or if the engine is NOT started yet)
1 if the call succedds.

SDK_API CCDUSB_ShowFactoryControlPanel(int DeviceID, char *passWord);

For user to access the factory features conveniently and easily, the library provides its second dialog window which has all the features on it.
Argument: DeviceID – the device number.
Password – A pointer to a characters which is the password, "661016" is recommended in most cases.
Return: -1 If the function fails (e.g. invalid device number)
1 If the call succeeds.



SDK_API CCDUSB_HideFactoryControlPanel(void);

This function hides the factory control panel, which is shown by invoking **CCDUSB_ShowFactoryControlPanel().**
Argument: None.
Return: it always returns 1.

SDK_API CCDUSB_SetExposureTime(int DeviceID, int exposureTime, bool Store);

User may set the exposure time by invoking this function.
Argument: DeviceNo – the device number which identifies the camera will be operated.
exposureTime – the Exposure Time is set, note it's in "Microsecond", and as the camera's minimum resolution for exposure is 100us, so it should be multiple of 100us (including 100us).
Store – Whether camera should put it in its NV memory, 0 means "Do not put", 1 means "Put".
Return: -1 If the function fails (e.g. invalid device number)
1 if the call succedds.
Important: In most cases, it's recommended to set the third parameter as "0", and let the PC to memorize and set back camera's parameters (including working mode and exposure time) while it starts the application (e.g. after initialize the camera engine).

SDK_API CCDUSB_InstallFrameHooker(int FrameType, FrameDataCallBack FrameHooker);
Argument: FrameType – 0: Raw Data
1: Processed Data.
FrameHooker – Callback function installed.
Return: -1 If the function fails (e.g. invalid Frame Type).
1 if the call succedds.
Important: The call back function will only be invoked while the frame grabbing is started, host will be notified every time the camera engine get a new frame (from any cameras in current working set).
Note:
1). Current there's NO difference between Raw Data and Processed Data, actually, no matter the FrameType is, camera engine always return the ADC data (16bit) of the frame.
2). The callback has the following prototype:

```
typedef void (* FrameDataCallBack)(int FrameType, int Row, int Col,
                                   TProcessedDataProperty* Attributes, unsigned char *BytePtr );
```

The TProcessedDataPropertys defined as:

```
typedef struct {
   int CameraID;
   int ExposureTime;
   int TimeStamp;
   int TriggerOccurred;
   int TriggerEventCount;
   int OverSaturated;
} TProcessedDataProperty;
```

Arguments of Call Back function:
FrameType – The same value passed in when the callback in stalled.
Row, Col – the Row and Column size of the frame, in our case, it's a Line camera, we always have Row = 1, Col = 3648.
Attributes – This is an important data structure which contains information of this particular frame, it has the following elements:
CameraID – This is the camera number (the same as the deviceID used in all the APIs), as camera engine might get frames from more than one cameras (there might be multiple cameras in current working set), this identifies the camera which generates the frame.
Exposure Time – The exposure time camera was used for generating this frame, as there's frame buffer on device, and on PC camera engine, it's especially useful for application to know the exposure time of a certain frame during the PC is adjusting a camera's exposure time.
TimeStamp – Camera firmware will mark each frame with a time stamp, this is a number from 0 – 65535 (and it's automatically round back) which is generated by the internal timer of the firmware, the unit of it is 100us (which is also the minimum step for exposure time). For example, if one Frame's time stamp is 1000 and the next frame's stamp is 1200, the time interval between them is 200x100us = 20ms.
TriggerOccurred – While the camera is in **NORMAL** mode, camera is grabbing frames continuously (as long as host is fetching frames from it), If an external trigger signal asserted during a frame grabbing, camera will set this flag to ONE (otherwise it's ZERO). This gives host an choice to poll the trigger assertion even it's in **NORMAL** mode. This flag should NOT be used while it's in **TRIGGER** mode.
TriggerEventCount – While the camera is in **TRIGGER** mode, camera will grab ONE frame after it's triggered by the external signal, each trigger will increase this count by ONE, this gives host a clue for the frame and its

trigger signal. Note that this count is reset to ZERO whenever host set the work mode to TRIGGER (so host can reset the count by invoking **CCDUSB_SetCameraWorkMode(DeviceID, 1),** even the camera is already in **TRIGGER** mode). Also note that the frame read time is around 7.5ms (3694 x 2us), if trigger signals are generated more frequently than this interval, the count is still physically reflect the trigger signal assertions. For example, the TriggerEventCount for a frame is 1, next frame's TriggerEventCount might be 6, which means assertion 2 to 5 are occurred during the reading time of first frame…and they're actually ignored. The 6th assertion occurs right after the reading of the frame, so it generates the next frame grabbing.

OverSaturated – The CCD sensor only works properly under conditional lighting, if it's over exposured, the CCD will be over saturated and in this case, the frame data grabbing back doesn't make sense, As the electronics accumulated for a frame is NOT released completely and accumulatively affects the next frame. While this flag is set to 1, host should reduce the exposure time (or setting proper external lighting condition) to make it back to 0.

BytePtr – The pointer to the memory buffer which holds the frame data.

Note that this callback is invoked in the main thread of the host application, GUI operations are allowed in this callback, however, blocking in this callback will slow down the frame rate of the camera engine.

SDK_API CCDUSB_InstallUSBDeviceHooker(DeviceFaultCallBack USBDeviceHooker);

User may call this function to install a callback, which will be invoked by camera engine while camera engine encounter camera faults.

Argument: USBDeviceHooker – the callback function registered to camera engine.

Return: It always return 1.

Note:

1). The camera engine doesn't support Plug&Play of the cameras, while the camera engine is starting work, any plug or unplug of the CCD Line cameras is NOT recommended. If plug or unplug occurs, the camera engine will stop its grabbing and invoke the installed callback function. This notifies the host the occurrence of the device configuration change and it's recommended for host to arrange house clean works. Host might simply do house keeping and terminate OR host might let user to re-start the camera engine. (Please refer to the Delphi and VC++ example code for this)

2). The callback function has the following prototype:

typedef void (DeviceFaultCallBack)(int FaultType);*

The FaultType is always ZERO in current version.

SDK_API CCDUSB_SetGPIOConifg(int DeviceID, unsigned char ConfigByte);

User may call this function to configure GPIO pins.

Argument : DeviceID – The device number, which identifies the camera is being operated.
ConfigByte – Only the 4 LSB are used, bit0 is for GPIO1, bit1 is for GPIO2 and so on. Set a certain bit to 1 configure the corresponding GPIO to output, otherwise it's input.

Return: -1 If the function fails (e.g. invalid device number)
1 if the call succedds.

SDK_API CCDUSB_SetGPIOInOut(int DeviceID, unsigned char OutputByte,
unsigned char *InputBytePtr);

User may call this function to set GPIO output pin states and read the input pins states.

Argument : DeviceID – The device number, which identifies the camera is being operated.
OutputByte – Only the 4 LSB are used, bit0 is for GPIO1, bit1 is for GPIO2 and so on. Set a certain bit to 1 will output High on the corresponding GPIO pin, otherwise it outputs Low. Note that it's only working for those pins are configured as "Output".
InputBytePtr – the Address of a byte, which will contain the current Pin States, only the 4 LSB bits are used, note that even a certain pin is configured as "output", we can still get its current state.

Return: -1 If the function fails (e.g. invalid device number)handle or it's camera WITHOUT GPIO)
1 if the call succedds.

APPENDIX G

MATLAB PROGRAMS

```
%*************************************************************************
%  IntensityCal.m
%
%  PROGRAM DESCRIPTION
%  The following program is designed to load in the TIFF pictures from an
%  intensity calibration.  Perform a linear fit on this data, comparing the
%  intensity level to the transmission through the lens for each pixel,
%  form: y=a*x+b.  Finally output the coefficients 'a' and 'b' for each
%  camera ALF and REF in the form of a .mat file.
%
%  Input:   12-bit TIFF pictures, spatial resolution: 1024 x 1024
%
%  Output:  aalf: a-coeficient for ALF camera pixels
%           balf: b-coeficient for ALF camera pixels
%           aref: a-coeficient for REF camera pixels
%           bref: b-coeficient for REF camera pixels
%
%  Written by: Brent Nelson
%              7-23-08
%*************************************************************************

clear all

% Names of the intensity calibration camera data files

localf1=['F:\IntensityCal070808\alf_0p3'];
locref1=['F:\IntensityCal070808\ref_0p2';
localf2=['F:\IntensityCal070808\alf_0p6'];
locref2=['F:\IntensityCal070808\ref_0p6'];
localf3=['F:\IntensityCal070808\alf_0p9'];
locref3=['F:\IntensityCal070808\ref_0p9'];
localf4=['F:\IntensityCal070808\alf_0p9_0p3'];
locref4=['F:\IntensityCal070808\ref_0p9_0p3'];
localf5=['F:\IntensityCal070808\alf_0p9_0p6'];
locref5=['F:\IntensityCal070808\ref_0p9_0p6'];

% Transmission; T=10^(-OD)

T1=10^(-.3);
T2=10^(-.6);
T3=10^(-.9);
T4=10^(-.3)*10^(-.9);
T5=10^(-.6)*10^(-.9);
```

```
% Vectorfrom for polyfit function

y=[T1; T2; T3; T4; T5];


% Load the TIFF files

P1alf=double(imread (localf1, 'TIFF', 'PixelRegion', {[0 1024], [0 1024]})-2^15);
P1ref=double(imread (locref1, 'TIFF', 'PixelRegion', {[0 1024], [0 1024]})-2^15);
P2alf=double(imread (localf2, 'TIFF', 'PixelRegion', {[0 1024], [0 1024]})-2^15);
P2ref=double(imread (locref2, 'TIFF', 'PixelRegion', {[0 1024], [0 1024]})-2^15);
P3alf=double(imread (localf3, 'TIFF', 'PixelRegion', {[0 1024], [0 1024]})-2^15);
P3ref=double(imread (locref3, 'TIFF', 'PixelRegion', {[0 1024], [0 1024]})-2^15);
P4alf=double(imread (localf4, 'TIFF', 'PixelRegion', {[0 1024], [0 1024]})-2^15);
P4ref=double(imread (locref4, 'TIFF', 'PixelRegion', {[0 1024], [0 1024]})-2^15);
P5alf=double(imread (localf5, 'TIFF', 'PixelRegion', {[0 1024], [0 1024]})-2^15);
P5ref=double(imread (locref5, 'TIFF', 'PixelRegion', {[0 1024], [0 1024]})-2^15);

% Reserve space in memory for the variables aalf, balf, aref, and bref

aalf=zeros(1024,1024);
balf=aalf;
aref=aalf;
bref=aalf;

% Perfrom a linera fit for each of the pixels and stor the variables 'a'
% and 'b' in to their respective matrix; aalf, balf, aref, bref

for i=1:1024
   for j=1:1024
      xalf=[P1alf(i,j); P2alf(i,j); P3alf(i,j); P4alf(i,j); P5alf(i,j)];
      palf=polyfit(xalf,y,1);
      aalf(i,j)=palf(1,1);
      balf(i,j)=palf(1,2);
      xref=[P1ref(i,j); P2ref(i,j); P3ref(i,j); P4ref(i,j); P5ref(i,j)];
      pref=polyfit(xref,y,1);
      aref(i,j)=pref(1,1);
      bref(i,j)=pref(1,2);
   end
end

% Save the matrices into a .mat format

save('C:\Documents and Settings\saikishan_s\Desktop\aalf.mat','aalf');
```

```
save('C:\Documents and Settings\saikishan_s\Desktop\balf.mat','balf');
save('C:\Documents and Settings\saikishan_s\Desktop\aref.mat','aref');
save('C:\Documents and Settings\saikishan_s\Desktop\bref.mat','bref');
```

```
%*********************************************************************
%  datareduction.m
%
%  PROGRAM DESCRIPTION
%  Datareduction.m utilizes the outputs from the spatial and intensity
%  calibrations to analyze data from DGV system.  User is prompted for
%  desired grid size.  Interp2 function is utilized to interpolate these
%  grid points from the pixel data.  The ratio of ALF/REF is calculated and
%  presented in graphical gray scale plot.
%
%  Input:   xmin:   minimum value for x grid
%           delx:   step size for x grid
%           xmax:   maximum vlaue for x grid
%           ymin:   minimum value for y grid
%           dely:   step size for y grid
%           ymax:   maximum vlaue for y grid
%     aalf, balf:   'a' and 'b' coeficients for linear fit of alf camera
%     aref, bref:   'a' and 'b' coeficients for linear fit of ref camera
%
%  Output:  PLOT:
%           Sub-plot1:  Alf Lumens
%           Sub-plot2:  Ref Lumens
%           Sub-plot3:  Ratio, gray scale
%
%  Written by: Brent Nelson
%            7-14-08
%*********************************************************************

clear all


flag=1;

while flag==1

    % User inputs for max, min and step size

   xmin=input('     X min:');
   delx=input('X step size:');
   xmax=input('      X max:');

   ymin=input('      Y min:');
   dely=input('Y step size:');
   ymax=input('      Y max:');
```

```
% Calcualtion of number of cells needed for matrices

maxx=(xmax-xmin)/delx+1;
maxy=(ymax-ymin)/dely+1;


%conversion from xy to ij from spatial calibration

%Alf equations
   %i=18.90222486+27.59481828*x-0.58033648*y
   %j=12.82408328+0.515225318*x+27.55172832*y

%Ref equations
   %i=64.66851413+27.03303263*x-1.40254797*y
   %j=2.552982162+1.44732106*x+26.89614515*y

alfi=zeros (maxx,maxy);
alfj=alfi;
refi=alfi;
refj=alfj;

p=0;
for x = xmin : delx : xmax
   p=p+1;
   q=0;
   for y = ymin : dely : ymax
      q=q+1;
      alfi(p,q)=18.90222486+27.59481828*x-0.58033648*y;
      alfj(p,q)=12.82408328+0.515225318*x+27.55172832*y;
      refi(p,q)=64.66851413+27.03303263*x-1.40254797*y;
      refj(p,q)=2.552982162+1.44732106*x+26.89614515*y;
   end
end

%check to ensure with in piture frame

A=min([min(min(alfi)),min(min(refi))]);
B=min([min(min(alfj)),min(min(refj))]);
C=max([max(max(alfi)),max(max(refi))]);
D=max([max(max(alfj)),max(max(refj))]);


%if A<0 then there is an error!
```

```
%if B>1024 then there is an error!

if (A < 0)
   disp('ERROR! X Minimum goes negative! reset grid.')
   fprintf('X min= %d\n',A)
elseif (B < 0)
   disp('ERROR! Y Minimum goes negative! reset grid.')
   fprintf('Y min= %d\n',B)
elseif (C > 1024)
   disp('ERROR! X Mmaximum goes over 1024! reset grid.')
   fprintf('X max= %d\n',C)
elseif (D > 1024)
   disp('ERROR! Y Mmaximum goes over 1024! reset grid.')
   fprintf('X max= %d\n',D)
else
   flag=0; %terminates program after complete iteration if successful


   %Intensity calibration

   load('C:\Documents and Settings\saikishan_s\Desktop\aalf.mat');
   load('C:\Documents and Settings\saikishan_s\Desktop\balf.mat');
   load('C:\Documents and Settings\saikishan_s\Desktop\aref.mat');
   load('C:\Documents and Settings\saikishan_s\Desktop\bref.mat');

   %Pre-allocates memory for matrices
   aT = zeros (1024,1024);
   rT = zeros (1024,1024);
   ratio = zeros (maxx,maxy);
   counter=0;

   for n=2:2:8;
      foldernumber=num2str(n);

      for m=1:24;
         counter=counter+1;

         filenumber=num2str(m);

         %locations of data

         localf=['C:\Documents and Settings\saikishan_s\Desktop\ValCal3\Vel …
                 … Cal0710080p',foldernumber,'\test1port1_',filenumber];
         locref=['C:\Documents and Settings\saikishan_s\Desktop\ValCal3\Vel …
```

```
      …  Cal0710080p',foldernumber,'\test1port0_',filenumber];

% Loads in tiff files

aA=double(imread (localf, 'TIFF', 'PixelRegion', {[0 1024], …
      … [0 1024]})-2^15);
rA=double(imread (locref, 'TIFF', 'PixelRegion', {[0 1024], …
      … [0 1024]})-2^15);


%Interpolates tiff data at desired locations

for i=1:1024
   for j=1:1024
      aT(i,j)=aalf(i,j)*aA(i,j)+balf(i,j);
      rT(i,j)=aref(i,j)*rA(i,j)+bref(i,j);
   end
end

calculated = interp2(aT,alfi,alfj);
calculated2 = interp2(rT,refi,refj);

 % Calculates the ratio

for x = 1 : maxx
   for y = 1 : maxy
      ratio(x,y)=calculated(x,y)/calculated2(x,y);
   end
end

%special conciderations for filenumbering of CCD camera
%data

filenumber2=num2str(2*m+1);

%loads CCD camera data a picks off alf/ref ratio

location=['C:\Documents and Settings\saikishan_s\Desktop\ValCal3\ …
      …  VelCal0710080p',foldernumber,'\test_',filenumber2,'.lvm'];
fid=fopen (location);
C=textscan(fid,'%s %s','Delimiter',',');
fclose(fid);

B=cell2mat(C{1,2}(23)); %str2num()
```

```
            %Plots of Alf, Ref and Ratio

            figure
            subplot(1,3,1)
            imagesc(calculated)
            colormap(gray)
            title('Alf Lumens')
            xlabel('x [mm]')
            ylabel('y [mm]')

            subplot(1,3,2)
            imagesc(calculated2)
            colormap(gray)
            title('Ref Lumens')
            xlabel('x [mm]')
            ylabel('y [mm]')

            subplot(1,3,3)
            clims = [0 1];
            imagesc(ratio,clims)
            colormap(gray)
            title(['ratio: ',B])
            xlabel('x [mm]')
            ylabel('y [mm]')
            colorbar ('location','eastoutside')
          end
        end
      end
    end
```

```
%*************************************************************************
%  ptremover.m
%
%  PROGRAM DESCRIPTION
%  This program can be inserted in to Datareduction.m to remove speckle from
%   the image.  ptremover.m will calculate the mean of the image, then check each
%   pixel to determine if it is below that mean or not, if the pixel is, the value of
%   that pixel will be replace by the mean value.  Insert at line 148
%
%  Written by: Brent Nelson
%            8-15-08
%*************************************************************************


%removing points less than mean

meana=mean(mean(aT));
meanr=mean(mean(rT));

lenga=length(find(aT>=meana));
lengr=length(find(rT>=meanr));

alf=zeros(lenga,3);
ref=zeros(lengr,3);

counter=1;
for i=1:1024
   for j=1:1024
      if aT(i,j) >=meana
         alf(counter,1)=-0.69442501+0.036224264*i+0.000763036*j;
         alf(counter,2)=-0.45239522-0.00067738*i+0.036280928*j;
         alf(counter,3)=aT(i,j);
         counter=counter+1;
      end
   end
end

counter=1;
for i=1:1024
   for j=1:1024
      if rT(i,j) >= meanr
         ref(counter,1)=-2.39024982+0.036888506*i+0.001923531*j;
         ref(counter,2)=0.034064044-0.00198512*i+0.037075943*j;
         ref(counter,3)=rT(i,j);
```

```
            counter=counter+1;
        end
    end
end
```

APPENDIX H

MATLAB SUPPLEMENTARY INFORMATION

polyfit

Polynomial curve fitting

Syntax

```
p = polyfit(x,y,n)
[p,S] = polyfit(x,y,n)
[p,S,mu] = polyfit(x,y,n)
```

Description

`p = polyfit(x,y,n)` finds the coefficients of a polynomial `p(x)` of degree `n` that fits the data, `p(x(i))` to `y(i)`, in a least squares sense. The result `p` is a row vector of length `n+1` containing the polynomial coefficients in descending powers

$$p(x) = p_1 x^n + p_2 x^{n-1} + \ldots + p_n x + p_{n+1}$$

`[p,S] = polyfit(x,y,n)` returns the polynomial coefficients `p` and a structure `S` for use with `polyval` to obtain error estimates or predictions. If the errors in the data `y` are independent normal with constant variance, `polyval` produces error bounds that contain at least 50% of the predictions.

`[p,S,mu] = polyfit(x,y,n)` finds the coefficients of a polynomial in

$$\hat{x} = \frac{x - \mu_1}{\mu_2}$$

where $\mu_1 = \text{mean}(x)$ and $\mu_2 = \text{std}(x)$. `mu` is the two-element vector $[\mu_1, \mu_2]$. This centering and scaling transformation improves the numerical properties of both the polynomial and the fitting algorithm.

Examples

This example involves fitting the error function, `erf(x)`, by a polynomial in `x`. This is a risky project because `erf(x)` is a bounded function, while polynomials are unbounded, so the fit might not be very good.

First generate a vector of `x` points, equally spaced in the interval $[0, 2.5]$; then evaluate `erf(x)` at those points.

```
x = (0: 0.1: 2.5)';
y = erf(x);
```

The coefficients in the approximating polynomial of degree 6 are

```
p = polyfit(x,y,6)

p =

 0.0084  -0.0983   0.4217   -0.7435  0.1471   1.1064  0.0004
```

There are seven coefficients and the polynomial is

$$0.0084x^6 - 0.0983x^5 + 0.4217x^4 - 0.7435x^3 + 0.1471x^2 + 1.1064x + 0.0004$$

To see how good the fit is, evaluate the polynomial at the data points with

```
f = polyval(p,x);
```

A table showing the data, fit, and error is

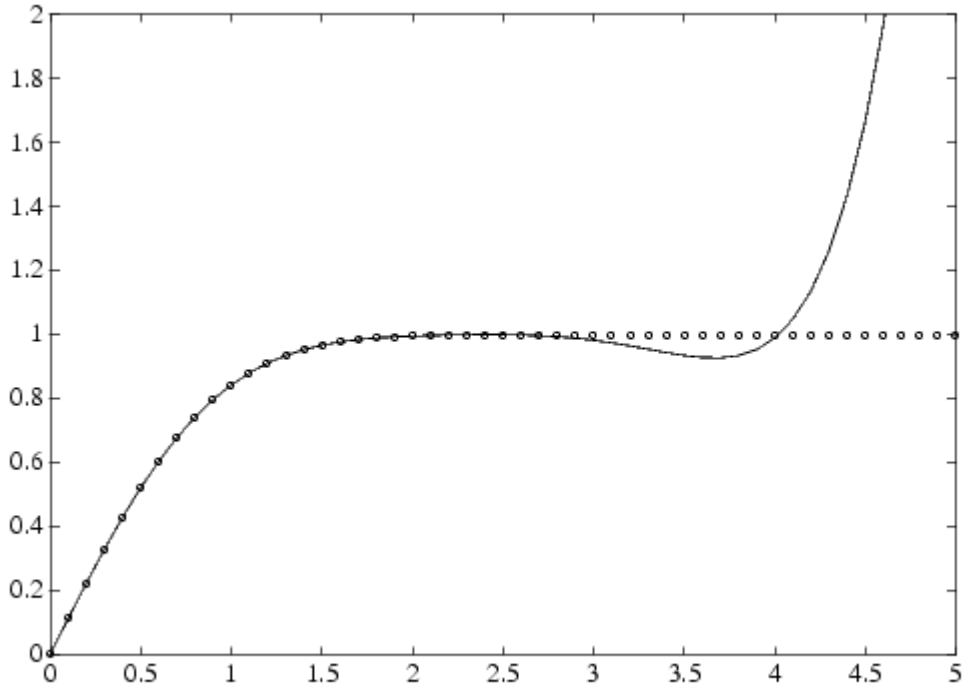
```
 table = [x y f y-f]

 table =

    0          0          0.0004    -0.0004
    0.1000     0.1125     0.1119     0.0006
    0.2000     0.2227     0.2223     0.0004
    0.3000     0.3286     0.3287    -0.0001
    0.4000     0.4284     0.4288    -0.0004
    ...
    2.1000     0.9970     0.9969     0.0001
    2.2000     0.9981     0.9982    -0.0001
    2.3000     0.9989     0.9991    -0.0003
    2.4000     0.9993     0.9995    -0.0002
    2.5000     0.9996     0.9994     0.0002
```

So, on this interval, the fit is good to between three and four digits. Beyond this interval the graph shows that the polynomial behavior takes over and the approximation quickly deteriorates.

```
x = (0: 0.1: 5)';
y = erf(x);
f = polyval(p,x);
plot(x,y,'o',x,f,'-')
axis([0  5  0  2])
```



Algorithm

The `polyfit` M-file forms the Vandermonde matrix, V, whose elements are powers of x.

$$v_{i,j} = x_i^{n-j}$$

It then uses the backslash operator, \, to solve the least squares problem

$$Vp \cong y$$

You can modify the M-file to use other functions of x as the basis functions.

interp2

Two-dimensional data interpolation (table lookup)

Syntax

```
ZI = interp2(X,Y,Z,XI,YI)
ZI = interp2(Z,XI,YI)
ZI = interp2(Z,ntimes)
ZI = interp2(X,Y,Z,XI,YI,method)
```

Description

`ZI = interp2(X,Y,Z,XI,YI)` returns matrix `ZI` containing elements corresponding to the elements of `XI` and `YI` and determined by interpolation within the two-dimensional function specified by matrices `X`, `Y`, and `Z`. `X` and `Y` must be monotonic, and have the same format ("plaid") as if they were produced by `meshgrid`. Matrices `X` and `Y` specify the points at which the data `Z` is given. Out of range values are returned as `NaNs`.

`XI` and `YI` can be matrices, in which case `interp2` returns the values of `Z` corresponding to the points `(XI(i,j),YI(i,j))`. Alternatively, you can pass in the row and column vectors `xi` and `yi`, respectively. In this case, `interp2` interprets these vectors as if you issued the command `meshgrid(xi,yi)`.

`ZI = interp2(Z,XI,YI)` assumes that `X = 1:n` and `Y = 1:m`, where `[m,n] = size(Z)`.

`ZI = interp2(Z,ntimes)` expands `Z` by interleaving interpolates between every element, working recursively for `ntimes`. `interp2(Z)` is the same as `interp2(Z,1)`.

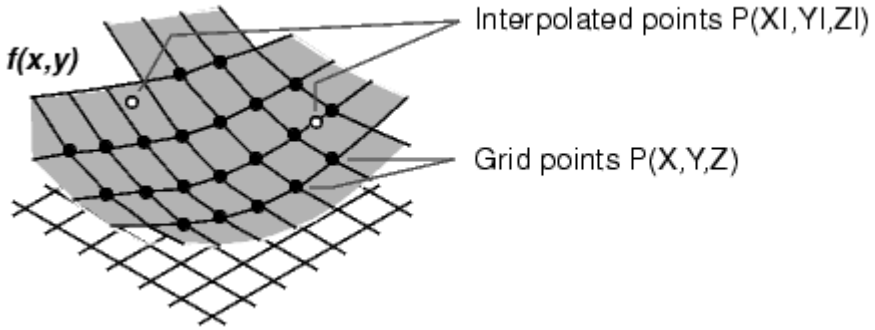
`ZI = interp2(X,Y,Z,XI,YI,method)` specifies an alternative interpolation method:

`'nearest'`	Nearest neighbor interpolation
`'linear'`	Bilinear interpolation (default)
`'spline'`	Cubic spline interpolation
`'cubic'`	Bicubuc interpolation

All interpolation methods require that `X` and `Y` be monotonic, and have the same format ("plaid") as if they were produced by `meshgrid`. If you provide two monotonic vectors, `interp2` changes them to a plaid internally. Variable spacing is handled by mapping the given values in `X`, `Y`, `XI`, and `YI` to an equally spaced domain before interpolating. For faster interpolation when `X` and `Y` are equally spaced and monotonic, use the methods `'*linear'`, `'*cubic'`, `'*spline'`, or `'*nearest'`.

Remarks

The `interp2` command interpolates between data points. It finds values of a two-dimensional function $f(x, y)$ underlying the data at intermediate points.

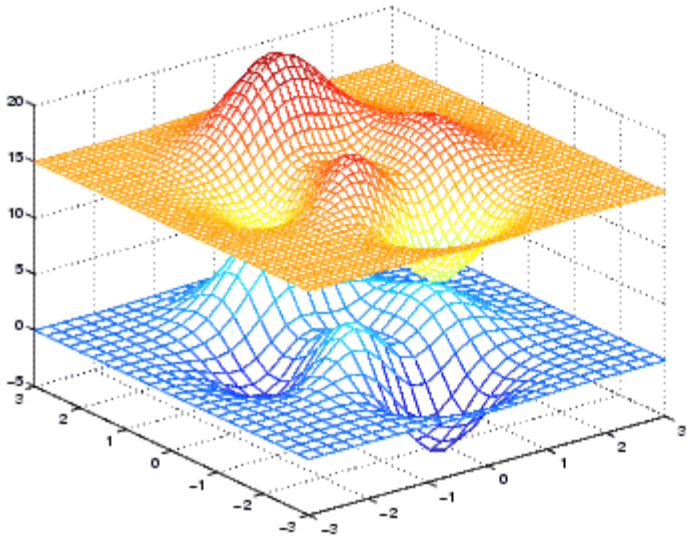


Interpolation is the same operation as table lookup. Described in table lookup terms, the table is `tab = [NaN,Y; X,Z]` and `interp2` looks up the elements of `XI` in `X`, `YI` in `Y`, and, based upon their location, returns values `ZI` interpolated within the elements of `Z`.

Examples

Example 1. Interpolate the `peaks` function over a finer grid.

```
[X,Y] = meshgrid(-3:.25:3);
Z = peaks(X,Y);
[XI,YI] = meshgrid(-3:.125:3);
ZI = interp2(X,Y,Z,XI,YI);
mesh(X,Y,Z), hold, mesh(XI,YI,ZI+15)
hold off
axis([-3 3 -3 3 -5 20])
```



Example 2. Given this set of employee data,

```
years = 1950:10:1990;
service = 10:10:30;
wage = [150.697 199.592 187.625
          179.323 195.072 250.287
          203.212 179.092 322.767
          226.505 153.706 426.730
          249.633 120.281 598.243];
```

it is possible to interpolate to find the wage earned in 1975 by an employee with 15 years' service:

```
w = interp2(service,years,wage,15,1975)
w =
  190.6287
```

VITA

Name:	Brent Nelson
Address:	Texas A&M University Department of Mechanical Engineering Attn: Gerald Morrison 3123 TAMU College Station, TX 77843-3123
Email Address:	nelsonba@tamu.edu
Education:	B.S., Mechanical Engineering, Rose-Hulman Institute of Technology, 2002 M.S., Mechanical Engineering, Texas A&M University, 2008