MEASUREMENT OF THE THERMAL PROPERTIES OF A WEAKLY-COUPLED COMPLEX (DUSTY) PLASMA

Except where reference is made to the work of others, the work described in this dissertation is my own or was done in collaboration with my advisory committee. This dissertation does not include proprietary or classified information.

Jeremiah Williams

Certificate of Approval:

Edward Thomas, Jr. Associate Professor, Chair Physics	Stephen F. Knowlton Professor Physics
Yu Lin Professor Physics	Allen Landers Assistant Professor Physics

Joe F. Pittman
Interim Dean
Graduate School

MEASUREMENT OF THE THERMAL PROPERTIES OF A WEAKLY-COUPLED COMPLEX (DUSTY) PLASMA

Jeremiah Williams

A Dissertation

Submitted to

the Graduate Faculty of

Auburn University

in Partial Fulfillment of the

Requirements for the

Degree of

Doctor of Philosophy

Auburn, Alabama
December 15, 2006

MEASUREMENT OF THE THERMAL PROPERTIES OF A WEAKLY-COUPLED COMPLEX (DUSTY) PLASMA

Permission is granted to Auburn University to make copies of this dissertation at its discretion, upon request of individuals or institutions and at their expense. The author reserves all publication rights.

Signature of Author

December 15, 2006
Date of Graduation

VITA

Jeremiah Williams, son of Douglas and Mary Williams, was born on August 30, 1976 in Binghamton, New York. He graduated from John F. Kennedy High School in 1994. He attended Dickinson College and graduated with a Bachelor of Science degree in Physics and Mathematics in 1998. He entered Graduate School at the University of California, Los Angeles and graduated with a Masters of Science degree in Physics in 2000. After spending three years as a faculty member in the Department of Physics at Illinois Wesleyan University, he entered Graduate School at Auburn University to pursue a Doctoral degree in Physics in August 2003. He was married to Anne-Evan (Kale) Williams, daughter of W. Wilford Kale Jr. and Louise Lambert Kale, on December 20, 2003. Their son, Franklin Douglas Williams, IV, was born on July 6, 2005.

DISSERTATION ABSTRACT

MEASUREMENT OF THE THERMAL PROPERTIES OF A WEAKLY-COUPLED DUSTY PLASMA

Jeremiah Williams

Doctor of Philosophy, December 15, 2006
(M.S., University of California, Los Angeles 2000)
(B.S., Dickinson College, 1998)

315 Typed Pages

Directed by Edward Thomas, Jr.

Knowledge of the velocity space distribution function facilitates the characterization of the thermal and dynamical properties of a system. In this work, a comprehensive series of experiments and numerical simulations are used to measure the velocity space distribution function of the microparticle component of a dusty plasma. In particular, this work introduces the use of stereoscopic particle image velocimetry techniques in the area of dusty plasma. This dissertation presents the results of extensive simulations of the PIV measurement technique, as applied to dusty plasmas, and the discusses the first measurements of the three-dimensional velocity space distribution function of a weakly-coupled dusty plasma.

This work demonstrates that it is possible to simultaneously measure all three velocity components over an illuminated slice of a dust cloud, allowing for a more accurate measure of the transport and energy properties of the dust cloud. From these measurements, the velocity space distribution function of a stable weakly-coupled complex dusty plasmas has been constructed for a wide range of experimental conditions.

From the simulations, it is found that there is a unique mapping function that relates measured PIV distribution to the underlying particle velocity space distribution function provided that the PIV analysis is applied to dust particles that have a size distribution that is nearly monodisperse. This mapping function is strongly dependent on the number density of the microparticle component that is being measured.

Experiments are performed using dust clouds composed of either 1.2 ± 0.5 μm alumina particles, 6.22 μm diameter melamine microspheres or 3.02 μm diameter silica microspheres. From the measured distribution functions, the bulk kinetic temperature of the microparticle component was extracted as a function of the neutral gas pressure. It was found that the bulk kinetic temperature of the microparticle component was anisotropic and significantly larger than the other plasma species. The heating mechanism responsible for these large temperatures is a function of the neutral pressure and appears to be more efficient with higher dust number densities.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Edward Thomas, Jr., for his support, advice and insight throughout this process. They were invaluable. In addition to the few tricks up his sleeve, I would also like to thank him for his encouragement and for keeping Lord Elroy at bay.

Beyond my advisor, there are many people who I would like to thank for fruitful discussions. An incomplete list includes Dr. Steve Anderson for his assistance with some of the technical aspects of the PIV software and technique, Dr. John Goree for his insight on measuring the number density of the dust clouds. Dr. Andrew Post-Zwicker for general discussions and providing the SEM images of the dust particles, Dr. Allen Landers for his insight on the statistical aspects of the analysis, Dr. Jianjun Dong for his insight on distribution functions and thermodynamics and Mr. William Williams for his insight in the area of optics. Finally, I would like to thank my former instructors, my colleagues, the members of the Plasma Sciences Laboratory, and all of the people whose names currently elude me for their support and encouragement over the years.

Finally, I would like to thank my family for their unending support and encouragement. Without their support and sacrifice, I would not have made it to this point. And of course, I must acknowledge my wife, Anne-Evan, for her support, patience and understanding over the last few years. In particular, thank you for picking up the slack over the last few months when I needed to spend time on this research.

Style manual or journal used_AIP Style Manual__________

Computer software used_____Microsoft Word____________

TABLE OF CONTENTS

LIST OF FIGURES xii

LIST OF TABLES xxv

CHAPTER 1: INTRODUCTION 1

- 1.1 HISTORICAL OVERVIEW 1
- 1.2 BASICS OF DUSTY PLASMA 3
  - 1.2.1 What is a dusty plasma? 3
  - 1.2.2 Examples of man-made dusty plasmas 11
  - 1.2.3 Examples of naturally-occurring dusty plasmas 15
- 1.3 PREVIOUS WORK 20
  - 1.3.1 Thermal studies involving phase transitions 21
  - 1.3.2 Thermal studies involving laser heating 32
  - 1.3.3 Other thermal studies 34
- 1.4 SCOPE OF THIS THESIS 37

CHAPTER 2: APPARATUS 38

- 2.1 EXPERIMENTAL DEVICE 38
  - 2.1.1 Vacuum Vessel 38
  - 2.1.2 Vacuum and Gas Systems 40
  - 2.1.3 Plasma and Dusty Plasma Generation 41
- 2.2 PARTICLE IMAGE VELOCIMETRY 45
  - 2.2.1 Principles of Particle Image Velocimetry 50
  - 2.2.2 Two-Dimensional Particle Image Velocimetry 52
  - 2.2.3 Stereoscopic Particle Image Velocimetry 62
    - 2.2.3.1 Image Dewarping and Calibration 71
    - 2.2.3.2 Reconstructing three-dimensional velocity information 75
  - 2.2.4 Stereoscopic Particle Image Velocimetry System at Auburn University 77
    - 2.2.4.1 Description of the system 77
    - 2.2.4.2 Verification of the system 79
    - 2.2.4.3 Measurement of the resolution limt of the system 86
- 2.3 PARTICLE DENSITY MEASUREMENT 90
  - 2.3.1 Density Measurement by Light Scattering Technique 92
  - 2.3.2 Monodisperse vs. polydisperse distributions of the microparticle component 94
  - 2.3.3 Determination of the scattering efficiency 95

CHAPTER 3: SIMULATION OF THE PIV MEASUREMENT 97
3.1 MOTIVATION 97
3.2 COMPUTATIONAL METHODOLOGY 99
3.3 RESULTS 103
3.3.1 Uniform Mass 104
3.3.2 Mass Distribution 110
3.3.2.1 Fixed width of particle size distribution 115
3.3.2.2 Effect of width of particle size distribution 118
3.3.2.3 Weighted PIV calculation 122
3.4 DISCUSSION OF SIMULATION RESULTS 125
CHAPTER 4: EXPERIMENTAL RESULTS 128
4.1 EXPERIMENTAL METHODOLOGY 128
4.2 RESULTS WITH MICROPARTICLES HAVING A POLYDISPERSE SIZE DISTRIBUTION 139
4.3 RESULTS WITH MICROPARTICLES HAVING A MONODISPERSE SIZE DISTRIBUTION 150
4.4 DISCUSSION OF RESULTS 161
CHAPTER 5:CONCLUSIONS 176
5.1 SUMMARY OF SIMULATIONS OF THE PIV MEASUREMENT TECHNIQUE 176
5.2 SUMMARY OF THE EXPERIMENTAL MEASUREMENTS 178
5.3 SUMMARY OF RESULTS 180
5.4 FUTURE DIRECTIONS 181
REFERENCES 184
APPENDIX 1: MATHEMATICAL BACKGROUND 189
A.1.1: MATHEMATICAL FOUNDATION OF THE 2D PIV TECHNIQUE 189
A.1.1.1 Mathematical description of image formation 189
A.1.1.2 Mathematical description of the Cross Correlation Analysis Technique 194
A1.2: PROOF OF THE CORRELATION THEOREM 201
A1.3: MATHEMATICS OF THE PINHOLE CALIBRATION MODEL 203
APPENDIX 2: C++ CODE TO SIMULATE THE PIV MEASUREMENT 207
A2.1: CONFIGURATION FILE FOR THE PIV SIMULATIONS 207
A2.2: C++ CODE FOR THE PIV SIMULATION 209
A2.3: ADDITIONAL LIBRARIES 221
APPENDIX 3: IGOR MACROS 229
A3.1: DESCRIPTION OF IGOR 229
A3.2: ANALYSIS CODE FOR SIMULATION DATA WITH MONODISPERSE PARTICLES 230
A3.3: ANALYSIS CODE FOR SIMULATION DATA WITH POLYDISPERSE PARTICLES 238
A3.4: ANALYSIS CODE FOR EXPERIMENTAL DATA 248
A3.5: 1D DRIFTING MAXWELLIAN FUNCTION FIT CODE 253

APPENDIX 4: LABVIEW CODE 254

A4.1: DESCRIPTION OF LABVIEW 254
A4.2: MASTER LABVIEW CODE 255
A4.3: RUN_ME_FOR_VECTOR_PREPROCESSING_USE.VI 257
A4.3.1: Convert_Data_to_Matrix_for_Vector_Preprocessing.vi 259
A4.3.2: Select_ROI_3_for_Vector_Preprocessing_AUTO.vi 261
A4.3.2.1: Predefined_Thresholding.vi 263
A4.3.2.2: Extract_Data_ROI.vi 264
A4.3.2.3: Count_Zeros.vi 265
A4.3.2.4: Valid_vectors_method_1.vi 266
A4.3.3: Get_File_List_for_Vector_Preprocessing.vi 268
A4.3.4: Generate_File_Names_for_Vector_Preprocessing.vi 269
A4.3.5: Select_ROI_3_for_Vector_Preprocessing.vi 270
A4.3.5.1: Manual Thresholding.vi 271
A4.3.6: Read_Single_Data_File_m_to_n_measurements.vi 273
A4.4: RUN_ME_TO COMPILE_Z_LOCATIONS_INTO_CLOUD.VI 274
A4.4.1: Get_File_List_to Compile_z_locations_into_cloud.vi 276
A4.5: RUN_ME_FOR_SCATTERING_EFFICEINCY.VI 277
A4.5.1: Manual_Particle_Counting_for_Scattering_efficeincy.vi 278
A4.5.2: Read_in_Images_(directory structure)_for_Scattering_efficeincy.vi 280
A4.5.2.1: Get_directory_file_list_for_Scattering_efficeincy.vi 282
A4.5.2.2: Read_in_Image_data_for_particle_density.vi 284
A4.5.3: Automatic_Particle_Counting_for_Scattering_efficeincy.vi 285
A4.5.3.1: Adjust Image Display Range (fixed value).vi 287
A4.6: RUN_ME_FOR_PIV_FACTOR.VI 289
A4.6.1: Get_File_List_PIV_factor.vi 291
A4.6.2: Compute_PIV_factor_to_file.vi 292
A4.6.3: Density_Range.vi 292
A4.7: RUN_ME_TO_MEASURE_PARTICLE_DENSITY.VI 293
A4.7.1: Generate_File_List_for_particle_density.vi 297
A4.7.2: User_defines_ROI_for_particle_density.vi 298
A4.7.3: Average n images (float, single set of images).vi 300
A4.7.4: Compute Particle Density (known factor).vi 301
A4.7.4.1: Length.vi 303
A4.7.5: Save 2D Data.vi 304
A4.7.5.1: Generate 2D Array to Save.vi 305
A4.7.6: Save 3D Data.vi 307
A4.7.6.1: Generate 3D Array to Save.vi 308
A4.8: RUN_ME_M_TO_N_MEASUREMENTS.VI 309
A4.8.1: Select_ROI_m_to_n_measurements.vi 311
A4.8.2: Extract_ROI_m_to_n_measurements.vi 312
A4.8.3: Convert_Data_to_Matrix_m_to_n_measurements.vi 314

LIST OF FIGURES

Figure 1.1: Image of particles illuminated by a laser in a plasma reactor. The bluish region is a region of dust particles illuminated using laser light scattering. Under this region, a silicon wafer is visible. The inset shows an SEM image of one of these dust particles, which is grown inside the reactor. 12

Figure 1.2: SEM images of microparticle formation from the ASDEX-U tokamak [25]. 13

Figure 1.3: Depicting where dust clouds are suspended in (a) an rf discharge and (b) – (d) a dc discharge. 14

Figure 1.4: Image of suspended dust particles in an argon plasma. The fluctuation of the particle density indicates the presence of a dust-acoustic wave. 14

Figure 1.5: Showing examples of dusty plasmas near earth: (a) images of a flame and soot particles suspended in the flame (b) lightening near the launch pad prior to the launch of STS-8, (c) noctilucent clouds and (d) exhaust from the Apollo 10 space shuttle launched on May 18, 1969. (Figures courtesy of NASA)........ 16

Figure 1.6: Showing examples of dusty plasmas in the solar system: (a) an image taken by the Hubble Space Telescope of Comet LINEAR after breaking into several “mini-comets” and (b) images of the spokes in Saturn’s B rings taken by Voyager 2. The spokes are highlighted in the inset. (Figures courtesy of NASA)........ 17

Figure 1.7: Showing examples of dusty plasmas in the interstellar medium: (a) Sharpless 140 (Image courtesy of NASA/JPL-Caltech), (b) a piece of cosmic dust (Image courtesy of NASA) and (c) NGC 1999 (Image courtesy of NASA and The Hubble Heritage Team (STScI)). 19

Figure 1.8: Showing the observed phases of a plasma crystal: (a) the crystalline state, (b) the floe and flow state, (c) the vibrational state and (d) the fluid-like state. (Images courtesy of H. Thomas, Max Planck Institute for Exterrestrial Physics, Garching, Germany)........ 23

Figure 2.1: A schematic of the 3DPX device showing (a) the top view of the experimental setup and (b) a detailed drawing of Section 1 of the experimental device where the experiments are conducted. The coordinate system shown in this figure will be used throughout this document. The *x*-*y* plane is defined by the plane of the laser sheet, while the *z*-direction is perpendicular to the laser sheet. It is noted that gravity acts in the *y*-direction. 39

Figure 2.2: A photograph of the 3DPX device and the stereoscopic particle image velocimetry diagnostic system 40

Figure 2.3: A photograph of the gas and vacuum subsystems on the 3DPX device. The tube leaving the bottom of the photograph is connected to a roughing pump. 41

Figure 2.4: A photograph of the flange and electrode assembly used to generate a plasma in the 3DPX device. 42

Figure 2.5: Photographs of the dust trays used in the experiments involving (a) polydisperse and (b) monodisperse microparticles. In all of the experiments presented in this dissertation, the dust tray is electrically floating 43

Figure 2.6: Schematic drawing showing the locations where dust is typically trapped in the 3DPX device. In (a), the dust is suspended below the anode in an anode sheath trap. In (b), the dust is suspended above the dust tray, in a sheath trap. 44

Figure 2.7: Images of a dust cloud suspended in (a) the anode sheath trap and (b) the sheath trap. The dust cloud is illuminated using (a) a Nd:YAG laser (λ = 532 nm) and (b) a diode laser (λ = 632 nm). The purple glow that is visible in the background is an argon plasma 45

Figure 2.8: Video images of a dust cloud in the presence of a Langmuir probe. As a bias is applied to the probe, a void is formed inside the dust cloud. 46

Figure 2.9: A schematic showing the typical setup for a two-dimensional quantitative imaging system. 48

Figure 2.10: Characteristic images taken when using (a) particle streak velocimetry (Image courtesy of H. Thomas, Max Planck Institute for Exterrestrial Physics, Garching, Germany), (b) particle tracking velocimetry, (c) particle image velocimetry and (d) laser speckle velocimetry. 49

Figure 2.11: A cartoon depicting a 2D PIV system (Figure used with permission of LaVision, Inc.) 51

Figure 2.12: Orientation of a stereo-PIV system in the (a) forward-backward (b) forward-forward and (c) backward-backward scattering configuration. In the forward-backward setup, the cameras are located on the same side of the flow of interest, while the cameras observe the flow of interest on opposite sides of flow in the backward-backward and forward-forward setup. (Figures used with permission of LaVision, Inc.) 52

Figure 2.13: A cartoon depicting the PIV analysis process (Figure courtesy of LaVision, Inc.) 53

Figure 2.14: Flow diagram for the PIV analysis process. 54

Figure 2.15: A lens system consisting of a spherical lens (L_2) with focal length f_2 and a cylindrical lens (L_1) with focal length f_1 is used to generate a laser sheet of height L_D, for illumination of the tracer particles. Incident on L_1 is a laser beam have a beam diameter L_d. 54

Figure 2.16: Diagram showing the imaging of a particle in the laser sheet on the CCD camera. 56

Figure 2.17: Implementation of the cross correlation technique using fast Fourier Transforms 59

Figure 2.18: Depicting how sub-pixel resolution is achieved in the PIV analysis. The bars denote the computed values of the cross-correlation function, while the curve depicts a best fit Gaussian to the computed values in the (a) x-direction and (b) y-direction. The peak from the curve fit yields the desired displacement with far greater accuracy than can be found from the raw cross correlation data. The actual displacement in the images used in this analysis was 8 pixels in the x-direction and 8.5 pixels in the y-direction. The displacement that is extracted from the fit was found to be 7.92 pixels in the x-direction and 8.55 pixels in the y-direction. It is noted that the cross correlation analysis performed in the analysis generating this figure was far simpler than what is done in the *DaVis* software. 61

Figure 2.19: Showing the motion observed by the PIV system, $\vec{v}_{measured}$, as opposed to the actual motion, $\vec{v}_{actual}$, of the particle. The true motion of the particle is depicted by the solid arrow, while the dashed arrow depicts the measured motion. 63

Figure 2.20: Plot of the ratio of the velocity measured by the PIV system, $v_{measured}$, to the true velocity, v_{actual}, as a function of the angle α. For $\alpha > 25°$, the error exceeds ten percent. 64

Figure 2.21: Depicting the orientation of the cameras in the two standard stereoscopic particle image velocimetry orientation: (a) translational method and (b) angular displacement method. (Figures used with permission of LaVision, Inc.) 65

Figure 2.22: Depicting how the out-of-plane component is extracted using the stereoscopic set-up. A difference in the measured displacements is observed if there is motion in the out-of-plane direction and is related to the displacement in the out-of-plane motion. (Figure used with permission of LaVision, Inc.) 66

Figure 2.23: Depicting the limited depth of field that arises due to the angular displacement method of stereoscopy. It is noted that only the dots on the right hand side of the image are in focus. 67

Figure 2.24: Depicting the Scheimpflug criteria, where the image, object and lens planes all intersect at a single point. 67

Figure 2.25: Showing the possible lens tilts that can be encountered when enforcing the Scheimpflug criteria. In (a) the lens is not tilted, (b) the lens has not been tilted enough, (c) the lens is titled to the Scheimpflug angle and (d) has the lens has tilted too far. 68

Figure 2.26: Depicting the image seen in Figure 2.23, once Scheimpflug criteria has been enforced. 69

Figure 2.27: Depicting the image seen in Figure 2.24 once the image has been dewarped into the global coordinate system. The red grid depicts the anticipated location of the dots on one plane of the calibration target. It is noted that the distortion observed in Figure 2.26 is gone and the magnification is uniform across the field of view. 70

Figure 2.28: Showing two 3D calibration targets used on the 3DPX device. The large target, shown in (a), consist of an array of Maltese crosses, while the smaller target, shown in (b), consist of an array of dots. 72

Figure 2.29: Showing the error that is introduced when the laser sheet and calibration plate are not properly aligned. 73

Figure 2.30: Flow diagram of the self-calibration process. 74

Figure 2.31: Depicting the geometry of the stereo-PIV system as viewed from (a) above and (b) the side. It is noted that the angles α and β do not have to be the same for both camera. In this example, the laser sheet propagates defines the x-y plane. 76

Figure 2.32: Schematic depicting the top view of the experimental setup and the orientation of the stereo-PIV hardware. 79

Figure 2.33: Showing the PIV analysis process used by the DaVis software. 80

Figure 2.34: SEM image of the silica microspheres used in experiments verifying the functionality of the stereo-PIV system. {SEM image courtesy of Andrew Post-Zwicker, Princeton Plasma Physics Laboratory} 81

Figure 2.35: Orientation of the electrodes and dust tray used to verify the functionality of the stereo-PIV system. 82

Figure 2.36: Schematic showing the orientation of the stereo-PIV and LLS imaging systems used to verify the functionality of the stereo-PIV system. 83

Figure 2.37: A video image sequence acquired using the second camera system confirming the displacement in the z-direction. For increased clarity, the cloud is outlined in red. The yellow rectangle in the first image denotes the location of the laser sheet for the stereo-PIV system. The dust tray that holds the source powder can be seen in the bottom, center of the images, while the perturbation electrode is visible in the lower right of the images. 84

Figure 2.38: Four images of the reconstructed velocity field from the stereo-PIV system for times t = 0.8, 1.4, 1.8, and 2.4 seconds. The red-green shading at the center of each of the four panels represents the z-component of the velocity of the microparticle cloud. The image labeled t = 0.8 sec, is before the application of the pulse and there is no net z-motion. At t = 1.4 sec, the pulse is applied and the cloud is moving in the $+z$-direction as indicated by the bright red contours. At t = 1.8 sec, the pulse has reversed and the particle cloud is moving in the $-z$-direction as indicated by the green contours. By the last image, t = 2.4 sec, the perturbation pulse has ended and the cloud has returned to a state of no net z-motion. 85

Figure 2.39: Distribution of the z-component of velocity for the four cases shown in Figure 2.37. 86

Figure 2.40: Showing the error in the stereo-PIV system in the (a) x-, (b) y- and (c) z-direction. The optical setup for this measurement corresponds to the one that was used in taking the data in Section 4.2 of this dissertation. It is noted that the error is comparable to what one observes in a two dimensional PIV system. 89

Figure 2.41: An inverted image of a strongly-coupled dusty plasma. (Figure courtesy of H. Thomas, Max Plank Institute for Extreterestrial Physics, Garching, Germany) 91

Figure 2.42: An inverted image of a dust cloud showing a region of known density, highlighted by the box. The image has been inverted to better see the dust particles. 93

Figure 2.43: Video image of dust clouds composed of multiple particle sizes. The original particle distributions included spheriglass particles having a mean size of 10 μm and 325 mesh (≤ 45 μm) silica. The banding structure that is observed is a characteristic feature of dust clouds containing a size distribution. 95

Figure 2.44: Showing the fluctuation level in the scattering efficiency, η, as a function of image number used in the calibration procedure. The fluctuation level is five percent. 96

Figure 3.1: Showing the effect of the PIV measurement on the measured distribution. In (a), one sees the velocity distribution that is being measured, while (b) shows the distribution that is measured using the PIV technique. It is noted that the measured distribution has a narrower width, which is a result of an averaging that is intrinsic to the measurement technique. Physically, the result of this averaging would be the measurement of a smaller temperature, *i.e.* $T_{PIV} < T_{dust}$. 98

Figure 3.2: A cartoon illustrating the PIV measurement. The dots represent the particles that seed the flow of interest and the red and blue boxes denote interrogation regions over which the PIV analysis is performed. Here, the interrogation region would be a 2 × 2 bin region. For the experiments presented in Chapter 4, a bin corresponds to a 6 × 6 pixel region of the acquired images. The PIV technique returns a vector that describes the average motion of all of the particles in the interrogation region. Two examples are seen in red and blue. It is noted that there is a 50% overlap in the interrogation region. 100

Figure 3.3: The result for a simulation of an ideal PIV measurement is depicted. The crosses depict the distribution of velocities that is input into the PIV simulation, while the solid line depicts the distribution of velocities that is returned by the simulated PIV measurement. The expected narrowing of the distribution of velocities is observed. 103

Figure 3.4: Plot of the ratio of the width of the output to the input velocity distribution, σ_{out}/σ_{in}, as a function of the effective particle density for tracer particles having a uniform size. Here, the results for alumina particles (r = 0.6 μm, ρ = 3800 kg/m^3) are shown. The blue symbol represents the mean value of σ_{out}/σ_{in} for 24 simulations over a wide range of kinetic temperatures, while the error bars represents the extreme values observed for the ratio σ_{out}/σ_{in}. 105

Figure 3.5: Plot of the ratio of the width of the output to the input velocity distribution, σ_{out}/σ_{in}, as a function of the effective particle density for tracer particles having a uniform size. Here, the result of a PIV simulation comparing two types of tracer particles is shown. The crosses represent alumina particles (r = 0. 6 μm, ρ = 3800 kg/m^3) and circles represent silica particles (r = 1.45 μm, ρ = 2600 kg/m^3). 107

Figure 3.6: Showing the results of a simulated PIV measurement when the tracer particles have a non-zero drift velocity. Here, the red curve shows the underlying velocity space distribution that is being measured, while the blue curve depicts the distribution of velocities that is measured using the PIV technique. It is noted that the narrowing of the measured distribution is observed, but there is no observed effect of the value of the drift velocity 108

Figure 3.7: Showing the effect that the drift velocity has on the PIV measurement. In (a), the ratio of widths is plotted as a function of the effective particle density. Here the crosses denote the result when there is no drift velocity and the circles represent the result when there is a non-zero drift velocity. The drift velocities used in the simulation are seen in (b). 109

Figure 3.8: A plot showing the overlay of the velocity distribution for monodisperse silica particles, r = 1.45 μm, in red and polydisperse silica particles, r = 1.45 ± 0.5 μm, in blue for the same kinetic temperature, T_{dust} = 15 eV. It is noted that the velocity distribution for polydisperse particles has a larger width. 111

Figure 3.9: Showing how the weighted average was used to find the effective width of the velocity distribution when the particles had a size distribution in the PIV simulations. 112

Figure 3.10: Showing the number of iterations needed to compute a meaningful value for the weighted average. 114

Figure 3.11: Plot of the ratio of the width of the output to the input velocity distribution, σ_{out}/σ_{in}, as a function of the effective particle density for tracer particles having a size distribution. Here, the results for alumina particles (r = 0.6 ± 0.25 μm, ρ = 3800 kg/m^3) are shown. The blue symbol represents the mean value of σ_{out}/σ_{in} for 24 simulations over a wide range of kinetic temperatures, while the error bars represent the extreme values observed for the ratio σ_{out}/σ_{in}. 116

Figure 3.12: Plot of the ratio of the width of the output to the input velocity distribution, σ_{out}/σ_{in}, as a function of the effective particle density for tracer particles having a size distribution. Here, the results from a simulation involving alumina particles having a size distribution ($r = 0.6 \pm 0.25$ µm, ρ = 3800 kg/m^3), depicted by crosses, are compared with the results from a simulation involving monodisperse alumina particles (r = 0.6 µm, ρ =3800 kg/m^3), depicted by circles. 117

Figure 3.13: Plot of the ratio of the width of the output to the input velocity distribution, $\sigma_{\mathrm{out}}/\sigma_{\mathrm{in}}$, as a function of particle density for tracer particles having a size distribution with a varying width, δr, relative to the mean radius, μ_r,. In all three plots, the crosses depict a size distribution with μ_r, = 1.5 µm, the triangles depict a size distribution with μ_r, = 3.0 µm and the dots depict a size distribution having μ_r, = 50 µm. In (a) the width of the size distribution relative to the mean value, $\delta r/\mu_r$, of 0.25, in (b) $\delta r/\mu_r = 0.5$ and is (c) $\delta r/\mu_r = 0.75$. 119

Figure 3.14: Plot of the ratio of the width of the output to the input velocity distribution as a function of mean particle size for simulations with $\delta r/\mu_r = 0.75$. Here, crosses represent an effective particle density, $n_{d,effective}$, of approximately 4.9, circles represent $n_{d,effective} \approx 3$ and triangles represent $n_{d,effective} \approx 2$. The dashed curve depicts the portion of the distribution included due to the truncation of the size distribution (*e.g.* $r \geq 0.25$ µm). 120

Figure 3.15: Simulation results showing the effect on the measured distribution when on includes the weighting due to particle size in the PIV technique. The solid black curve depicts the underlying velocity distribution, the dotted red curve depicts what is measured by the PIV technique when all particle contribute equally and the dashed blue curve depicts what is measured by the PIV technique when the particle size is a factor in the cross correlation analysis. The strong peak at zero velocity that occurs when including the weighting due to particle size is not observed experimentally. 123

Figure 3.16: A curve relating the effective particle density that is used in the simulations to particle densities that would be measured in an experiment. It is important to note that the slope of this curve will depend on the particular calibration of the PIV system. 126

Figure 3.17: An inverted image of a typical dust cloud. Superimposed on this image is a grid that represents the interrogation regions that are used in the PIV calculation. It is noted that the particle density is not uniform of the interrogation region, which corresponds to a 2 × 2 box bin (12 × 12 pixels). 127

Figure 4.1: A photograph of (a) the electrodes used in the experiments described in this dissertation and (b) the location of the dust cloud when the electrodes are arranged in the cathode-anode-tray configuration. In (b), the lower electrode (anode) is visible. A dust cloud composed of 1.51. μm diameter silica microspheres is illuminated by a laser diode (λ = 632 nm) and is visible in red. The purplish background is an argon plasma. 129

Figure 4.2: A typical video image of a weakly-coupled dusty plasma for (a) 1.2 ± 0.5 μm diameter alumina microparticles, (b) 6.22 μm diameter melamine microspheres and (c) 3.02 μm diameter silica microspheres. 130

Figure 4.3: A plot depicting the oscillation of the PIV camera when the linear translation stage comes to rest. Here, the (a) vertical and (b) horizontal motion of a fixed location inside of the 3DPX device is tracked over the first ten images in a measurement sequence. 131

Figure 4.4: Showing the process by which the underlying velocity space distribution function is obtained. In (a), the measured velocity distribution, seen in red, and the resolution error in the stereo-PIV system, shown in blue, are overlaid. In (b), the measured velocity distribution, seen in red, and the resolution error in the stereo-PIV system, shown in blue, are added to yield an intermediate distribution, shown in green. In (c), the mapping function corrects the width of the intermediate function, shown in green, to account for the intrinsic averaging, which yields the velocity space distribution, seen in black. The velocity space distribution function shown in (d), again in black, can then be used to study the thermal properties of the microparticle component of the dust cloud. 135

Figure 4.5: Showing the kinetic dust temperature of 6.22 μm diameter melamine microspheres as a function of the number of PIV measurements included in the distribution function. The symbol represents the mean temperature from the 45 - *n* measured temperatures, while the error bars depict the range of temperatures that are measured. The values for n used in this calculation are n = 1, 2, 5, 10, 25 and 45. It is observed that the average temperature is relatively constant. However, the level of fluctuation decreases as more measurements are included in the velocity distribution. 137

Figure 4.6: Showing the measured velocity distributions for different numbers of included measurements. The red curve depicts the velocity in the *x*-direction, the green curve depicts the velocity in the y-direction and the blue curve depicts the velocity in the *z*-direction. It is observed that there is less fluctuation in the shape of the velocity distribution as the number of measurements included increases. 138

Figure 4.7: Showing (a) the size distribution and (b) an SEM image of the microparticles used in the experiments presented in Section 4.2. In (a) the red circles depict measured values for the particle size, while the blue curve is a Gaussian fit. (SEM image courtesy of Andrew Post-Zwicker, Princeton Plasma Physics Laboratory) 139

Figure 4.8: Typical velocity distributions in the (a) *x*-, (b) *y*- and (c) *z*-direction. The crosses depict the measured distribution of velocities, while the solid curve represents the fit to a drifting Maxwellian velocity distribution. The experimental parameters for this data are p = 120 mTorr, V_{anode} = 231 V and $V_{cathode}$ = -91 V. 142

Figure 4.9: Multiple measurements of the kinetic temperature of the microparticle component taken over a period of an hour with constant experimental conditions (p = 110 mTorr, V_{anode} = 230 V, $V_{cathode}$ = -89 V). 144

Figure 4.10: Kinetic temperature of the microparticle component in the (a) *x*-, (b) *y*- and (c) *z*-direction as a function of neutral gas pressure. The different colors indicate data from different experimental runs. It is noted that the error bars are statistical and do not account for fluctuations in the background plasma. 146

Figure 4.11: A plot of the kinetic temperature for dust clouds composed of polydisperse alumina microparticles ($r = 0.6 \pm 0.25$ μm. $\rho = 3800$ kg/m^3), normalized using Equation 4.3, as a function of neutral gas pressure 148

Figure 4.12: A plot of (a) the number density of the microparticle component as a function of the neutral gas pressure and (b) the kinetic temperature of the microparticle component as a function of the number density of the microparticle component. It is noted that the color coding is consistent with Figure 4.10, with each color representing data taken using a single dust cloud. 149

Figure 4.13: Kinetic temperature of the microparticle component in the (a) *x*-, (b) *y*- and (c) *z*-direction as a function of neutral gas pressure. The different colors indicate data from different experimental runs. It is noted that the error bars are statistical and do not account for fluctuations in the background plasma. 151

Figure 4.14: Schematic showing a side view of the electrode orientation. 153

Figure 4.15: Kinetic temperature of the microparticle component in the (a) *x*-, (b) *y*- and (c) *z*-direction as a function of the neutral gas pressure. The data from Cloud 3 is shown in green, while the red data indicates data from Cloud 4. ... 155

Figure 4.16: Normalized temperature as a function of neutral gas pressure for dust clouds composed of (a) melamine and (b) silica microspheres. In (a), the red data corresponds to the cloud that was generated at low pressure, while the blue data corresponds to the cloud that was generated at high pressure. In (b), the data in black, purple, green and red corresponds to Clouds 1 – 4 respectively. ... 157

Figure 4.17: Normalized temperature in the (a) *x*- (b) *y*- and (c) *z*-direction as a function of the neutral gas pressure for the dust clouds composed of silica microspheres. The data in black, purple, green and red corresponds to Clouds 1 – 4 respectively. ... 159

Figure 4.18: A plot of the kinetic temperature as a function of the number density of the microparticle component for dust clouds composed of (a) melamine and (b) silica dust. In (b), the green and blue data corresponds to Cloud 3, while the red data corresponds to Cloud 4. ... 160

Figure 4.19: An image depicting the location of a cloud that is suspended in the (a) tray sheath and (b) anode sheath. In (a) the lower electrode is not visible, but the dust tray can be seen in the lower central region of the image. In (b), the upper electrode is visible, but the dust tray is not. In (c), the relative location of the equilibrium position for a dust cloud in the anode and tray sheath are shown. ... 164

Figure 4.20: Distribution of velocities in a single slice from the central region of a dust cloud suspended in the (a) tray sheath and (b) anode sheath. It is observed that the asymmetry seen in the anode region is significantly reduced when the dust cloud is suspended in the tray sheath. ... 165

Figure 4.21: A plot of the width of the measured velocity distribution as a function of the time between the laser pulses. It is observed that the width of the distribution gets smaller as the time between laser pulses increases. The lines indicate the estimated resolution limit of the stereo-PIV system. ... 167

Figure 4.22: A plot of the ratio of the widths: $\sigma_y:\sigma_x$ in red and $\sigma_z:\sigma_x$, in blue, as function of time between laser pulses. It is noted that the asymmetry is preserved regardless of the time between laser pulses. ... 169

Figure 4.23: Distribution of velocities throughout the volume of a dust cloud suspended in the anode spot. Each plot represents the distribution of velocities measured using the stereo-PIV system at a z-location given in the upper right hand corner of the plot. It is observed that the fluctuations in the measured distribution is smallest in the central region of the cloud, where the clouds are bigger and more vectors are reconstructed. Additionally, it is observed that the same asymmetry is observed throughout the cloud and in the distribution for the entire cloud seen in the lower right tile. 170

Figure 4.24: Contour plots of the velocity (left column) and speed (right column) in the x- (top row) y- (middle row) and z-direction (bottom row) for a single slice of a weakly-coupled dusty plasma. The lines depict contours of constant velocity/speed, while the shaded background more clearly illustrate the spatial variation of the motion. It is observed that there is a great deal of spatial structure, particularly in the z-direction.. 172

Figure 4.25: Showing the spatial variation in the z-direction of (a) the number density, (b) T_z, (c) T_y and (d) T_z.. 175

Figure A.1.1: Imaging of a particle in the laser sheet on the CCD camera. The fluid flow is illuminated by a laser sheet at the object plane and imaged on the CCD array at the image plane. In this figure, lower case letters denote quantities measured by the CCD camera, while capital letters denote quantities illuminated by the laser sheet. The position of a particles at time t_0 are denoted by x_1 and t_1 by x_1'. 190

Figure A.1.2: Depicting the correlation function. The displacement used to generate this correlation plane was 8 pixels in both x and y.................. 197

Figure A.1.3: Depicting the process used to construct a PIV vector. Identical $n \times n$ evaluation regions from Image 1, taken at time t_0, and Image 2, taken at time t_1, are extracted. A cross correlation analysis of these two image subsets is done and the peak in the correlation plane yields the desired in-plane displacement, $\vec{d}$. 198

Figure A.1.4: Showing the formation of a correlation plane. A square region of the two acquired images is extracted. The subset of Image 2 is shifted around the subset of Image 1 and the quantity in Equation A.1.17 is computed... 199

Figure A.1.5: Cross correlation function for simulated images. In (a) and (b), the generated images used in this analysis are seen. Each image contains a tracer particle and random noise that defines the noise floor. The defined shift in the tracer particle was 3 pixels in x and y. In (c), a top view of the correlation plane is seen, while (d) shows a topographic view of the correlation plane............................... 200

Figure A.1.6: Showing the basic idea of the pinhole model... 203

Figure A.1.7: Top view of the 3D pinhole model. Particles are located in the world coordinate system and are illuminated by a laser sheet................ 205

LIST OF TABLES

Table 1.1: The results from thermal studies made during the phase transitions observed in a plasma crystal. It is observed that the experiments were performed over a wide range of experimental conditions, but they all exhibited the same key result: a notably larger temperature in the fluid-like regime. 29

Table 1.2: Showing the results from thermal studies where a phase transition was induced by heating the plasma crystal using an external means (laser heating). As observed in Section 1.3.1, there is a notably larger temperature in the fluid-like regime. 32

Table 1.3: Showing other thermal studies involving a dusty plasma. 35

CHAPTER 1: INTRODUCTION

In Section 1.1, a short historical overview plasmas and dusty plasmas is presented. In Section 1.2, a basic introduction to dusty plasmas, their formation and examples of their existence in nature, the laboratory and the industrial setting is given. In Section 1.3, the work that has been done to date studying the thermal properties of a dusty plasma is summarized. Finally, in Section 1.4, the purpose of this research, the basic structure of this dissertation and a summary of the major findings will be presented.

1.1 HISTORICAL OVERVIEW

In nature, matter naturally exists in one of four states: solid, liquid, gas and plasma. In a solid, the atoms (or molecules) that make up the material are closely packed together and locked in a rigid lattice structure. Additionally, the atoms (or molecules) interact strongly. As energy is added to the system, the atoms (or molecules) gain energy and are able to break the bonds that hold them in the rigid lattice structure. This process denotes a phase transition from a solid state to a liquid state. In the liquid state, the atoms (or molecules) that make up the matter are closely spaced but interact weakly. Consequently, the liquid state does not hold a fixed shape, which is a characteristic of the solid state. As more energy is added, another phase transition is observed into the gas state. In this state, the atoms (or molecules) that make up the matter are spread far apart and interact very weakly. Typically, the only interactions between the atoms (or molecules) in the gas state are due to collisions with other atoms (or molecules). As more energy is added, electrons within the atoms gain enough energy to break free from the atoms forming an ionized gas. This quasi-neutral collection of ions, electrons and neutral particles makes up the final, and most common state of matter, plasma [1].

Sir William Crooke first observed the plasma state in the laboratory in 1879 [2]. In these early studies, a discharge tube, known now as a Crookes Tube, was used and the plasma state was referred to as "radiant matter." The field was pioneered by Irving Langmuir, who coined the term "plasma" in 1928 based on the similarity that he observed between the way that blood plasma carries red and white blood cells and the way that he imagined that plasma as an electrified fluid that carried ions and electrons [3].

Among the observations made by Langmuir was perhaps the first observation of a dusty plasma [4]. In these experiments, an argon plasma was generated in a Pyrex glass tube by a heated tungsten filament (cathode) and disc shaped anode. By temporarily lowering the temperature of the cathode; tungsten was sputtered from the filament. These bits of tungsten were observed to have a profound impact on the discharge and under certain conditions, were observed to streak through the discharge at incredibly large velocities, on the order of 10 to 30 cm/s. The existence of these streamers led to this type of discharge being referred to as a "streamer discharge." [4]

After these initial observations, little work was done examining plasmas that included particulate matter. However, interest in such plasmas was revived in 1982 with the observation of spokes in Saturn's B ring [5]. This was followed a number of other discoveries that have lead to an explosive growth in this area of research [6]. Among these discoveries were the realization that the observed contamination of semiconductor materials in plasma processing facilities was due to the growth of particles in the plasma [7, 8], the plasma crystal [9-13] and a variety of new plasma modes [14-18].

Additionally, the field of dusty plasmas has led to an increase in the basic understanding of many fundamental aspects of plasma physics. These include research in the areas of charge particle collection by Langmuir probes [19, 20] and the electrical structure of plasma sheaths [21, 22]. Recently, dusty plasmas have proven to be a model system for studying the properties of fluids at the kinetic level [23, 24] and there has been an increased interest in understanding the generation and transport of dust as applied to applications involving fusion plasmas [25- 28]. Furthermore, because the microparticle component of this type of plasma can be visualized directly, their properties can be studied at the kinetic level and it is possible, in principle, to measure the full distribution function of the microparticle component.

1.2 BASICS OF DUSTY PLASMA

1.2.1 WHAT IS A DUSTY PLASMA?

Dusty plasmas (also referred to as complex plasmas, colloidal plasmas and fine particulate plasmas) are a four-component system composed of ions, electrons, neutral particles and charged particulate matter (*i.e.* the dust component) [29, 30]. The size of the particulate matter in this state of matter covers many orders of magnitude. In the laboratory, the dust component consists of particles ranging in size from the nanometer to the micrometer scale. In space, the particulate matter can be a few meters in size. As a result, the dust component can be considered both macroscopic, when compared to the other plasma components, and at the same time microscopic, when compared to the length scales of the plasma environment that they are a part of. In both the laboratory environment and nature, the particulate matter becomes charged and acts as a third charged species. The presence of this third charged species significantly increases the complexity of the plasma. Hence, these systems are often referred to as complex plasmas.

Once the particles are in the plasma, they will become charged through a number of possible mechanisms [29-31]. These mechanisms include the direct collection of plasma particles (*i.e.* ion and electron currents), thermionic emission, ionizing radiation (*i.e.* nuclear particles) and photoionization (*i.e.* ultraviolet, UV, radiation). Nuclear effects have great relevance in space environment, in particular in stellar regions. UV effects play a large role in the dusty plasmas in space and were studied in early experiments that examined the charging of dust grains.

In the discharge plasmas that are investigated in this work, the relevant charging mechanisms are the electron and ion flux to the particles from the surrounding plasma. In this environment, one can model the dust grains similar to an electrically floating probe. It is noted that new insight has been gained into the understanding of the standard probe diagnostics of plasma physics due to research in this field [19, 20].

In a typical discharge plasma, the negative species (electrons) are much more mobile than the positively charged species (ions). As a result, the dust grains reach a negative floating potential with respect to the plasma. The electrostatic potential of the dust grain can be found by balancing the electron repulsion and the ion collection given in Equation 1.1.

$$\frac{dq_d}{dt} = I_e + I_i \tag{1.1}$$

where q_d is the charge accumulated on the particulate matter, I_e is the electron current and I_i is the ion current. There are two key results that come from Equation 1.1. First, the charge on the dust grain is a dynamic quantity. As a result, the charge on the dust grain must be solved for self-consistently with other plasma parameters, such as the ion and electron densities. Secondly, for a given set of plasma conditions, Equation 1.1 implies that there will be a finite charging time for the grain to reach its equilibrium charge. For typical lab conditions, this is on the order of tens of microseconds. Because of the short charging time relative to the typical experimental time scales (~seconds) considered here, it is assumed that the dust grains have a constant equilibrium charge that will be estimated using the following steps.

The ion and electrons are typically approximated using orbit-limited charging currents [31], where it is assumed that the ions and the electrons obey a Maxwellian distribution at temperatures T_i and T_e, respectively. For systems where the dust grains have a negative surface potential with respect to the plasma, ϕ_d, the orbit-limited electron and ion currents are given by

$$I_e = -\left[4\pi r_d{}^2 n_e e\sqrt{\frac{k_B T_e}{2\pi m_e}}\right]\exp\left\{\frac{e\phi_d}{k_B T_e}\right\} \tag{1.2}$$

$$I_i = \left[4\pi r_d{}^2 n_i e\sqrt{\frac{k_B T_i}{2\pi m_i}}\right]\left(1-\frac{Z_i e\phi_d}{k_B T_i}\right) \tag{1.3}$$

where e is the electron charge, k_B is Boltzmann's constant, Z_i is the ion charge, n_e and n_ι are respectively the electron and ion number density, m_e and m_ι are respectively the electron and ion mass and r_d is the dust radius. The $\exp\left\{\frac{e\phi_d}{k_B T_e}\right\}$ term is Equation 1.2 represents the electron repulsion, while the $\left(1-\frac{Z_i e\phi_d}{k_B T_i}\right)$ term in Equation 1.3 represents the ion collection.

It is noted that the model for the charging currents given in Equations 1.2 and 1.3 are derived assuming the "dust in plasma" limit. In this low density limit, $r_d \ll \lambda_D \ll a$, where r_d is the radius of the dust grain, λ_D is the effective Debye length and a is the interparticle spacing. In the dusty plasma limit, where $r_d < \lambda_D \sim a$, the scattering of ions and electrons by the dust component is more complicated. When this more complicated, and realistic, scattering is considered, it is found that the currents given in Equations 1.2 and 1.3 are slightly larger than the actual currents to the dust grain. As a result, the value of the grain charge computed using this model for the charging currents will provide an upper bound [29].

The effect of this third charged species on the plasma is significant. First, the amount of negative charge that is tied to the dust grain can be quite substantial. As a result, there can be a significant reduction is the local free electron or ion populations in the vicinity of dust grains, depending on whether the dust grain acquires a net positive or negative charge. For the typical discharge plasma that contains a relatively high concentration of dust particles, it is possible that $Z_d n_d >> n_e$. Similarly, in plasmas where photoionization is a dominant charging mechanism, photoemission is responsible for the ejection of electrons from the dust grains surface. In this case, it is possible that $Z_d n_d >> n_i$. As a result, the standard condition of quasi-neutrality in a plasma must be modified.

$$n_i = n_e + Z_d n_d \tag{1.4}$$

where Z_d is the number of electron charge on the dust particle, while n_i, n_e and n_d are the number density of the ion, electron and dust populations respectively. Using this condition of quasineutrality, it is possible to solve for the potential of the of the dust particle when the charging currents have reached an equilibrium value, *i.e.* $I_i + I_e = 0$. By combining Equations 1.1 – 1.4 and treating the dust particles as spherical capacitors (*i.e.* $q_d = 4\pi\varepsilon_o r_d \phi_d$), it is found that

$$\left[1 + P\left(\frac{e\phi_d}{k_B T_e}\right)\right]\sqrt{\frac{m_i T_e}{m_e T_i}}\exp\left\{\frac{e\phi_d}{k_B T_e}\right\} = \left(1 - \frac{e\phi_d}{k_B T_i}\right) \tag{1.5}$$

where the Havnes parameter, $P = 4\pi n_d r_d \left(\frac{\varepsilon_o k_B T_e}{e n_i} \right)$ [32], is introduced. The Havnes parameter is a dimensionless quantity that describes an effective screening in a dusty plasma. By numerically solving Equation 1.5, the surface potential of the dust grain is determined. Using typical discharge conditions for the plasma studied in this dissertation (*e.g.* an argon plasma with alumina microparticles where $T_i = 0.025$ eV, $T_e = 3$ eV, $n_d = 1 \times 10^{10}$ m^{-3}, $n_e = n_i = 5 \times 10^{14}$ m^{-3}, $r_d = 0.6 \times 10^{-6}$ m), a dust potential of -6.7 V and a dust charge of approximately 2800 electrons is found. It is noted that for these conditions under the condition of quasineutrality, $Z_d n_d / n_i \sim 0.056$; *i.e.* the dust has collected just over 5% of the electrons from the local plasma.

The second effect that this third charged species has on a plasma involves modifying the screening length. As a result, the screening length inside a dusty plasma can be quite different than what is observed in the standard ion-electron plasma. The standard screening (Debye) length, λ_D, in a dusty plasma is given by a combination of the ion and electron Debye lengths [33] as

$$\frac{1}{\lambda_D^{\ 2}} = \frac{1}{\lambda_{D,i}^{\ 2}} + \frac{1}{\lambda_{D,e}^{\ 2}} \tag{1.6}$$

where $\lambda_{D,\alpha} = \sqrt{\frac{\varepsilon_o k_B T_\alpha}{Z_\alpha n_\alpha e^2}}$ is the standard Debye length for species α (ions and electrons) and ε_o is the permittivity of free space. For the plasma parameters noted above, $\lambda_{D,e} = 5.76 \times 10^{-4}$ m, $\lambda_{D,i} = 5.26 \times 10^{-5}$ m and $\lambda_D = 5.23 \times 10^{-5}$ m $\sim \lambda_{D,i}$. At the present time, experiments appear to give different results as to the appropriateness of $\lambda_{D,e}$ or λ_D as the scaling length and this is an active topic of research in the dusty plasma community.

A third effect of the third charged species is an increase, when compared with those in a traditional two-component plasma, in the time scales for dusty plasma phenomena, *i.e.* the plasma reacts more slowly. In large part, this is due to the very large mass of the dust grains, when compared to the ion and electron masses, *i.e.* m_i/m_d, $m_e/m_d << 1$. This new time scale is parameterized by the dust-plasma frequency, ω_{pd}.

$$\omega_{pd} = \sqrt{\frac{4\pi n_d Z_d^2 e^2}{m_d}} \tag{1.7}$$

For the plasma parameters noted above, $\omega_{pd} = 171.45$ rad/s. This new time scale is of particular benefit to the experimentalist, as it allows a great variety of optical imaging technologies – some as simple as basic CCD cameras – to be used to explore plasma physics phenomena. This is a powerful advantage for the study of dusty plasma and one that is exploited in this work.

A fourth effect arises from the fact that the dust has a very small charge to mass ratio, when compared to the other plasma species (*e.g.* the ions and the electrons). Consequently, the dynamic response of the dust gives rise to many new wave phenomena. A number of these, including the dust-acoustic [17, 34, 35] and dust lattice [18] wave, have been extensively explored both in theory and in experiment and have proved to be a valuable tool for testing the non-linear properties of a dusty plasma.

Finally, because of the high charge, the electrostatic interaction between dust particles and between the dust particles and the surrounding plasma can become quite large. In some cases, the electrostatic energy can exceed the kinetic energy, in which case the system becomes highly coupled. In this state, a wide range of new phenomena, such as the crystalline plasma state, can be observed [9-13]. To quantify this, the Coulomb coupling parameter, which is a ratio of the electrostatic potential energy to the thermal energy, is used

$$\Gamma = \left[\frac{Z_d^{\,2} e^2}{4\pi\varepsilon_o a k_B T_d}\right]\exp\left\{-\frac{r_d}{\lambda_D}\right\} \tag{1.8}$$

where λ_D is the Debye length given in Equation 1.6, and T_d and Z_d are the temperature and dust charge respectively. It is noted that the term in brackets represents the ratio of electrostatic to thermal energy, while the exponential term accounts for screening effects. For $\Gamma << 1$, the system is in a gas like state. For $\Gamma \sim 1$, the system is in a liquid-like state and for $\Gamma >> 1$, the system is in a solid, or crystalline, state. For the plasma crystals to occur, $\Gamma \geq 170$ [36].

1.2.2 EXAMPLES OF MAN-MADE DUSTY PLASMAS

Dusty plasmas can be generated in research or industrial plasma devices. In some cases, the presence of the dust particles can be quite detrimental to the desired plasma environment. In other cases, the presence of the dust is exploited as a valuable commodity. This section gives a few examples of these types of dusty plasmas applications.

An early example of a dusty plasma was first observed by Gary Selwyn at IBM in the fabrication of microelectronics, as seen in Figure 1.1. Here, dust particles are produced during plasma etching of silicon wafers. The bluish glow is from laser light that is scattered from the dust particles that form during the etching process [7]. Dust particle formation can also occur in thin film deposition using reactive gases [8, 37, 38]. In these cases, the dust is an undesirable contaminant that needs be removed from or controlled in the plasma environment [39-41].

On the other hand, the presence of dust can also be quite beneficial in some industrial applications. A few examples where the presence of dust can be beneficial include the growth of diamond films, the fabrication of particles of known size and shape [42] or of particle with desired surface and optical properties [43, 44], and the implantation of nanoparticles to enhance surface properties (*i.e.* amorphous silicon solar cells) [45, 46].

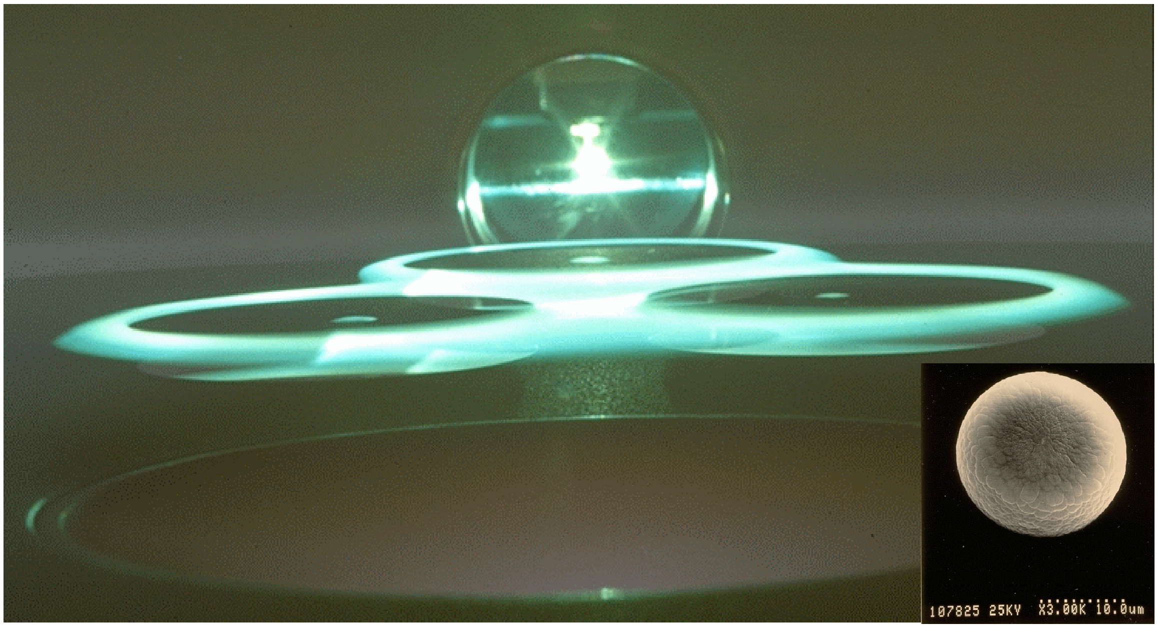


Figure 1.1: Image of particles illuminated by a laser in a plasma reactor. The bluish region is a region of dust particles illuminated using laser light scattering. Under this region, a silicon wafer is visible. The inset shows an SEM image of one of these dust particles, which is grown inside the reactor.

Recently, the presence of dust in fusion devices has become a topic of great interest [25-28]. When the fusion plasma comes into contact with the wall of the device or with other plasma facing components, material can be vaporized or sputtered off of the surface. The results of these plasma-wall interactions is illustrated in Figure 1.2. Here, SEM images of material that accumulates on the floor of a fusion device are shown. This dust is not confined by the magnetic fields and if these particles travel from the wall into the core plasma, they can become a major energy loss mechanism. Additionally, the dust can transport tritium in the plasma, as well as become radioactive themselves, making the dust also an additional safety concern.

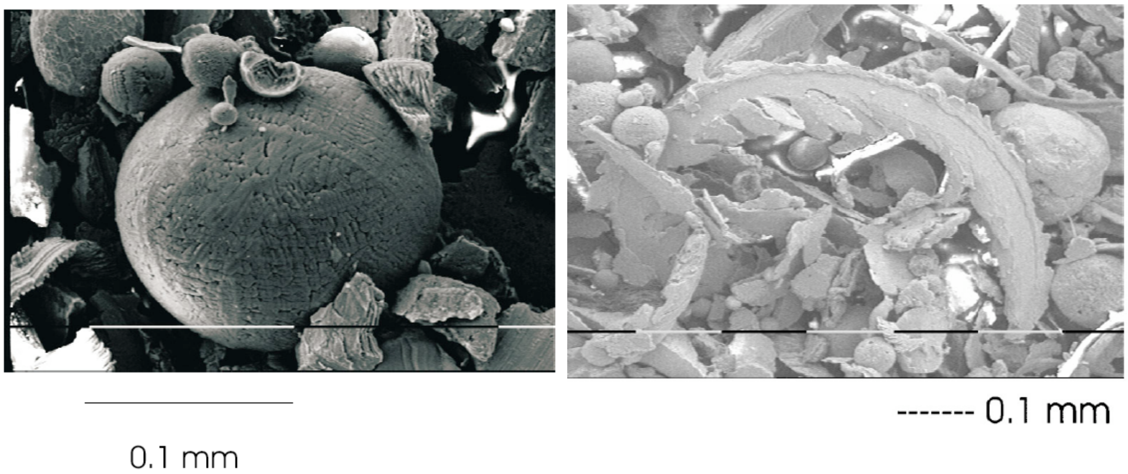


Figure 1.2: SEM images of microparticle formation from the ASDEX-U tokamak [25].

For the majority of laboratory studies, dusty plasma are generated in either radio frequency (rf) and direct current (dc) discharge plasmas. Additionally, the rf discharge plasma is also quite common in the industrial setting. In both types of discharges, the dust particles are suspended in regions of high electric field, where the electric force balances the gravitational force, i.e. $m_d g = q_d E$, where m_d is the mass of the dust grain, g is the acceleration due to gravity, $q_d = Z_d e$ is the charge of the dust grain and E is the local electric field. Typical suspension locations can be seen in Figure 1.3.

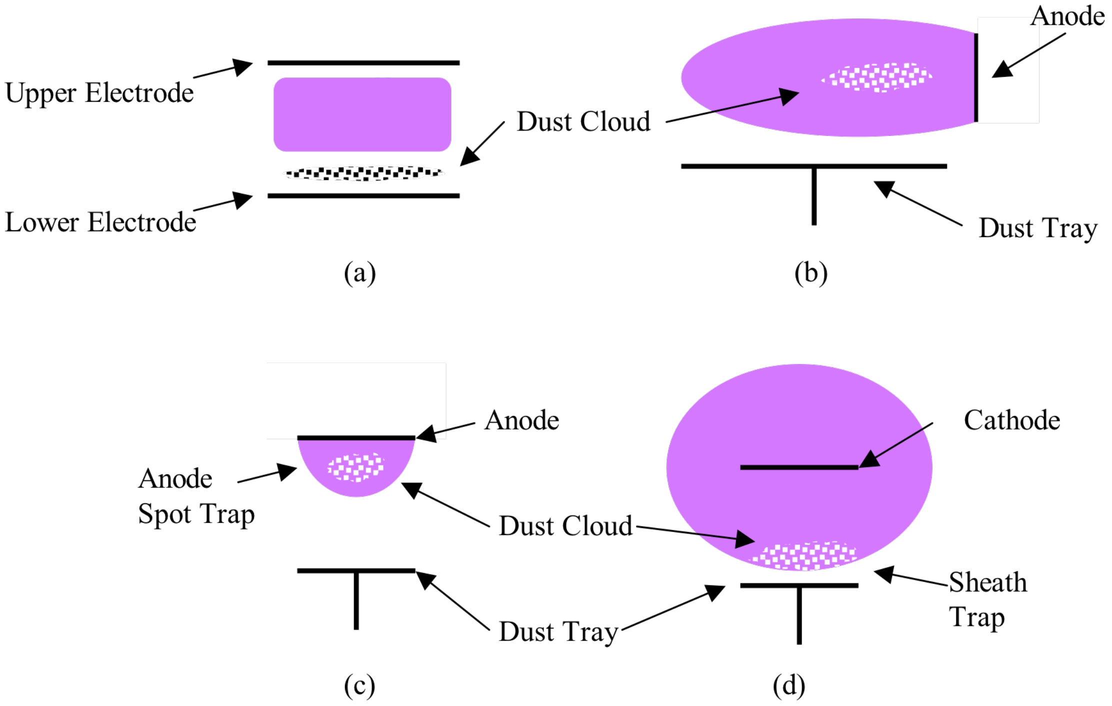


Figure 1.3: Depicting where dust clouds are suspended in (a) an rf discharge and (b) – (d) a dc discharge.

Figure 1.4 shows an example of dusty plasma formed in the laboratory using a dc discharge device.

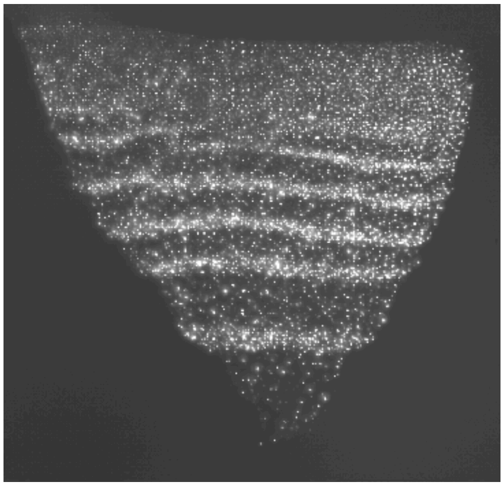
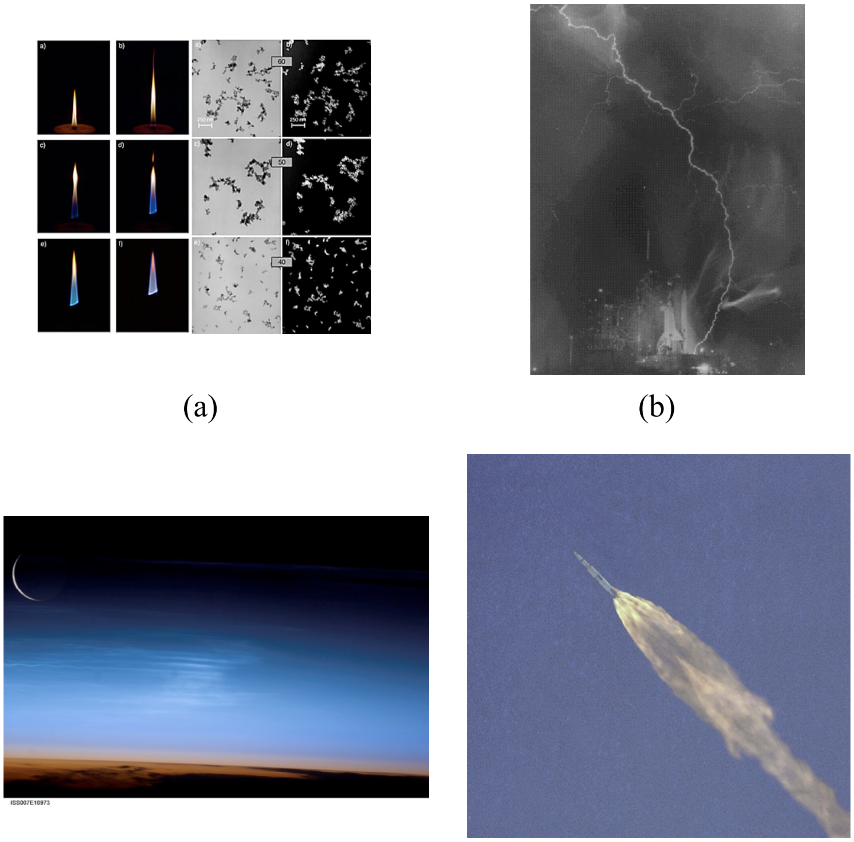


Figure 1.4: Image of suspended dust particles in an argon plasma. The fluctuation of the particle density indicates the presence of a dust-acoustic wave.

1.2.3 EXAMPLES OF NATURALLY-OCCURRING DUSTY PLASMAS

Dusty plasmas are somewhat ubiquitous in nature. Near the earth, examples of dusty plasma include flames, lightning and noctilucent clouds. Flames, as seen in Figure 1.5(a) are an example of a weakly-coupled ionized plasma. The glow that is observed is due to blackbody radiation of soot particles that are suspended in the plasma [29]. When lightning occurs, a plasma is created in an atmosphere that is full of water droplets and pollutants. These bits of particulate matter become charged, forming a dusty plasma, Figure 1.5(b). Another example of lightning, known as ball lightning, is possibly due to the oxidation of nano-sized silicon particles that are suspended in the air after a lightning strike occurs [47].

Further from the earth's surface, ~85 km above the surface, noctilucent clouds [48] are observed, as seen in Figure 1.5(c). Here micron sized ice-coated particles are suspended in a plasma. A final example, Figure 1.5(d), is seen in the exhaust of a spacecraft as it is launched into space.

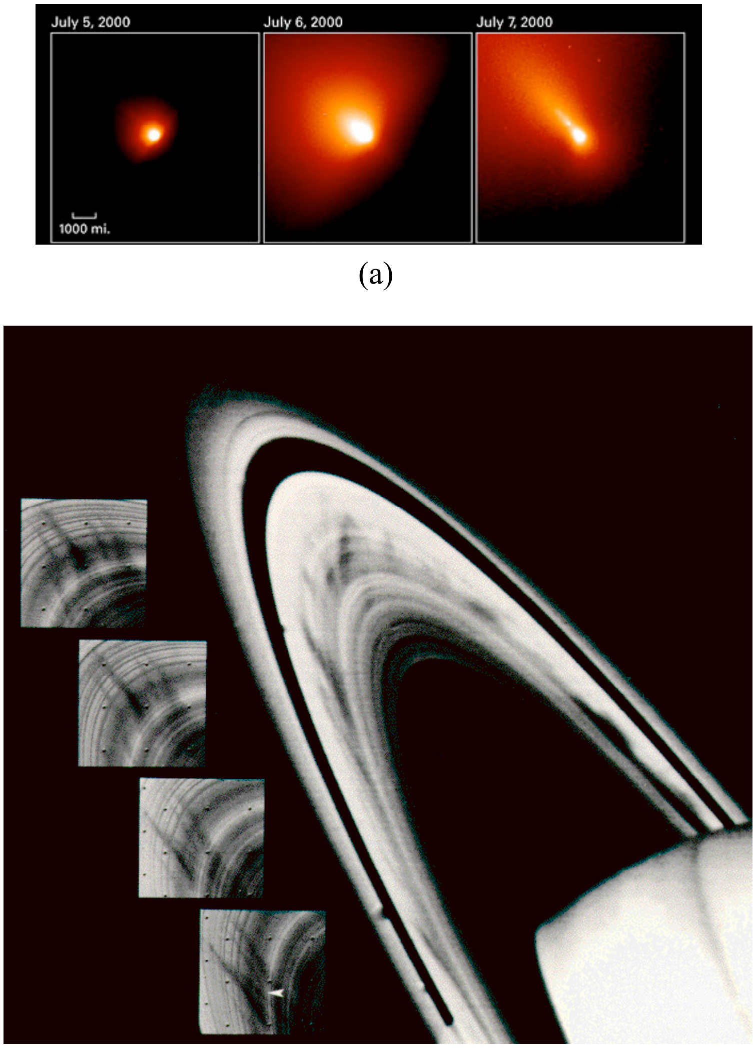


(a) (b)

(c) (d)

Figure 1.5: Showing examples of dusty plasmas near earth: (a) images of a flame and soot particles suspended in the flame (b) lightening near the launch pad prior to the launch of STS-8, (c) noctilucent clouds and (d) exhaust from the Apollo 10 space shuttle launched on May 18, 1969. (Figures courtesy of NASA)

As one moves away from the earth and into space, the abundance of dust is observed by the existence of planets and stellar objects. As a result, there are many examples of dusty plasmas in space. In our immediate solar system, two examples of dusty plasmas include comets and planetary rings. Comets [49], Figure 1.6(a), can be thought of as "dirty" snowballs consisting of ice and other particular matter on the order of a few microns in size. Saturn's rings are another example, one that led to a resurgence in this area of study. In particular, a great success of dusty plasmas is the description of the spoke features that have been observed in Saturn's rings [50], Figure 1.6(b).



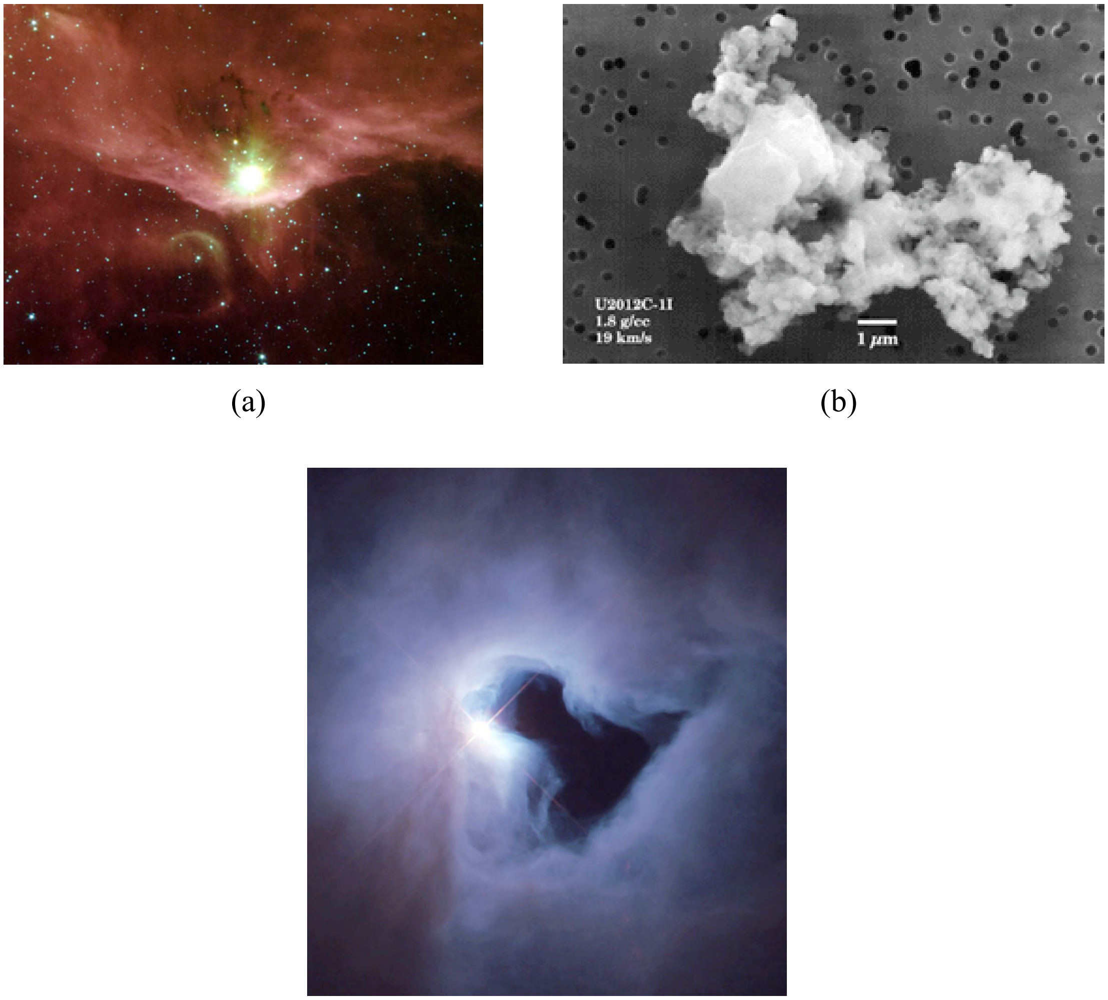
(a)

(b)

Figure 1.6: Showing examples of dusty plasmas in the solar system: (a) an image taken by the Hubble Space Telescope of Comet LINEAR after breaking into several "mini-comets" and (b) images of the spokes in Saturn's B rings taken by Voyager 2. The spokes are highlighted in the inset. (Figures courtesy of NASA)

As one moves into the interstellar media, there are numerous examples of dusty plasmas consisting of ice, silicates, graphite and so on. Some examples include the interstellar clouds [51, 52], space nebulae [53], and zodiacal light clouds [54]. In fact, the formation of objects in the universe come from first being in the dusty state and it is not surprising that dusty plasmas are the most common structural form of matter in the universe. Some of these structures can be seen in Figure 1.7. An image in the infrared of Sharpless 140 taken with the Spitzer Space Telescope is seen in Figure 1.7(a). Here, three stars are surrounded by dense dust, which prevents them from being observed in the visible region of the spectrum. The bright red bowl that is visible is the dust cloud. Figure 1.7(b) shows an SEM image of a piece of cosmic dust. In Figure 1.7(c), an image of the NGC 1999 nebula taken with the Hubble Space Telescope is seen. This reflection nebula is visible because a young star (V380 Orionis), located to the left of center illuminates the surrounding dust. In the darken region to the right of V380 Orionis is a Bok globule, a region of gas and cosmic dust so dense that it block all light.

It is noted that the presence of this dust can be an important factor in the behavior of the system. For instance, the dust may provide a dominant cooling mechanisms. Additionally, the dust can be an important plasma source or an important recombination mechanism. Dust can also provide coupling between the electromagnetic and gravitational fields [55-58]. Further, the presence of dust in the interstellar medium can be an important aspect of planet and stellar formation [59-64]. Indeed, given the abundance of dust in the plasma environment, one might even ask if there is such a thing as a non-dusty plasma [6]. However, over the last twenty years, much work has gone into this field [6], leading to a more fundamental understanding of a wide range of physics phenomena.



(a) (b)

(c)

Figure 1.7: Showing examples of dusty plasmas in the interstellar medium: (a) Sharpless 140 (Image courtesy of NASA/JPL-Caltech), (b) a piece of cosmic dust (Image courtesy of NASA) and (c) NGC 1999 (Image courtesy of NASA and The Hubble Heritage Team (STScI)).

1.3 PREVIOUS WORK

The measurement of the thermodynamic properties of a dusty plasma is experimentally challenging problem because of the difficulties associated with measuring the three-dimensional velocities of the microparticles. In spite of this, there have been a number of measurements made of the kinetic temperature of the dust component using the plasma crystal [10, 65, 66] and, more recently, the dust cluster [67, 68]. In particular, there has been a great deal of interest in studying the evolution of the dust kinetic temperature during the observed melting transition that occurs in the plasma crystal. More recently, there have been a number of experiments that have measured the temperature of the dust component while simulating random heating of the microparticle component using lasers [69, 70]. For the most part, all of these experiments have been performed using microparticles suspended in an rf discharge plasmas. In what follows, a review of measurements of the thermal properties of a dusty plasma is presented. Section 1.3.1 presents results from thermal studies on systems during observed phase transitions from a strongly-coupled, crystalline state to weakly-coupled fluid like state. Section 1.3.2 presents the results of thermal studies done while actively heating a dusty plasma using lasers. Section 1.3.3 presents other studies involving the thermal properties of a dusty plasma.

1.3.1 THERMAL STUDIES INVOLVING PHASE TRANSITIONS

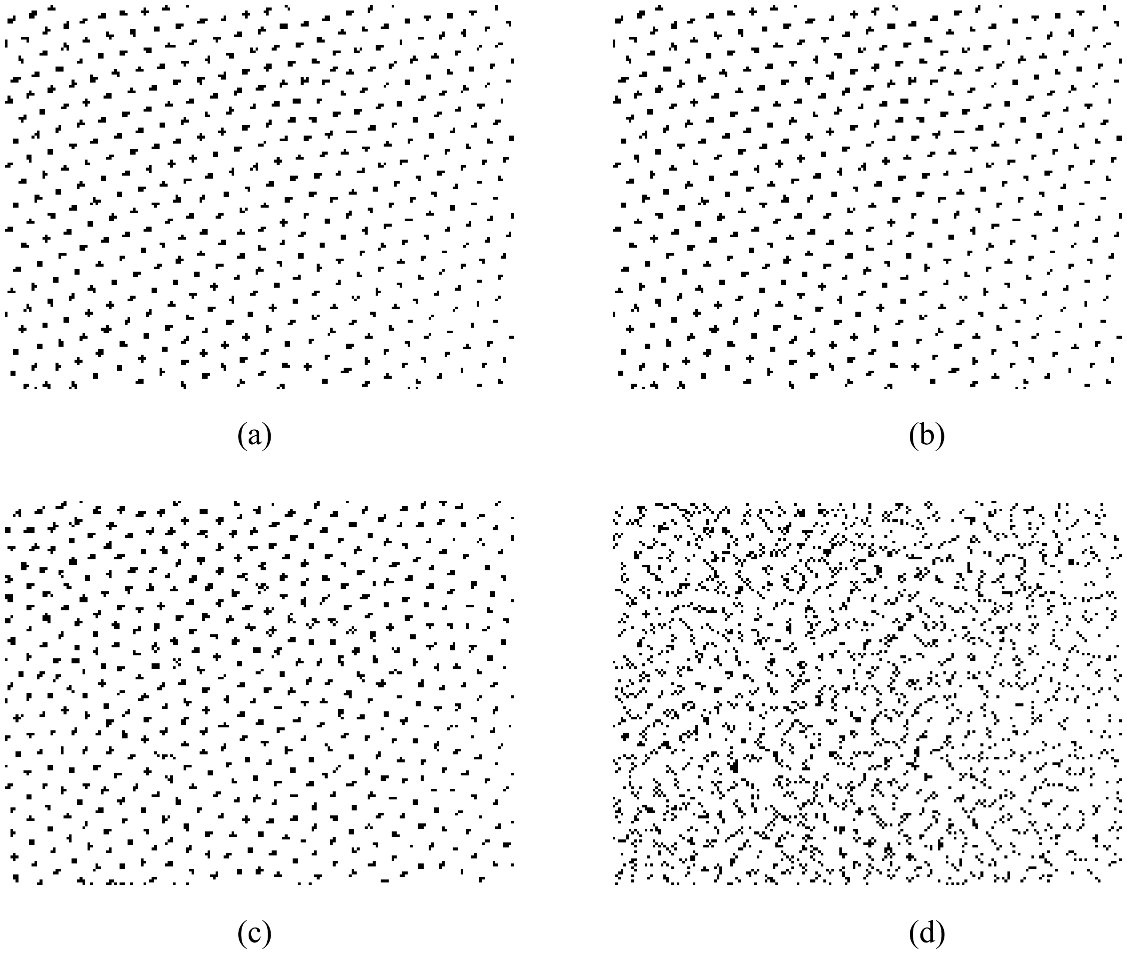
Early measurements of the thermal properties of a dusty plasma were performed on strongly-coupled systems and focused on measuring the temperature of the dust component as a strongly coupled, two dimensional plasma crystal underwent a phase transition to a fluid-like state. Once a strongly-coupled plasma crystal was generated, a phase transition was induced by varying experimental parameters. Typically, there are three mechanisms that can be used to initiate a phase transition [71].

1. Increasing the applied rf power (in a dc system, this corresponds to increasing the current). As the power (current) increases, the plasma density increases. This results in a decrease in the Debye shielding length and the dust particles move closer together. This leads to a decrease in the interaction energy with surrounding particles and a transition from an ordered to a disordered state occurs.
2. Decreasing the neutral gas pressure. This mechanism induces a phase transition through two mechanisms. First, a decrease in the neutral gas pressure changes the viscosity of the background gas (*i.e.* collisions between neutral and dust particles) and the rate at which energy is dissipated. Second, a change in the pressure will change the background plasma parameters. Together, these two effects can initiate a phase transition from an ordered to a disordered state.
3. Adding particles to the system that are more massive than those in the plasma crystal. In this case, the more massive particles will move to an equilibrium position below the plasma crystal. The interaction of the dust particles and the

confining field generates an instability that leads to a phase transition. If a confining field is not present, this method cannot be used.

Of these three methods, varying the neutral pressure is an experimentally convenient method of controlling the phase transition.

During the observed phase transition, four distinct states were identified [71]: crystalline, "flow and floe," vibrational and fluid. In the crystalline state, the dust particles are aligned in a rigid lattice structure. It is possible that the dust grain exists in multiple vertical planes. The dust particles tend to interact strongly with surrounding particles in the same plane. However the vertical spacing of the planes is such that the particles in different layers tend to not interact strongly. As a result, these systems, even if they are three dimensional, can be treated as a two dimensional system. The "flow and floe" state is characterized by regions where the dust is aligned in a crystalline state (floes) and regions of directed transient flows (flows). In this state, the thermal velocities of the microparticles are close to the thermal speed of the background neutral gas and the transient flows have magnitudes that are about half of the thermal speed of the background neutral gas. Additionally, in this state, one occasionally observes vertical particle migration to other other layers. In the vibrational state, the orientational ordering increases and there is a decrease in the flow regions. The vibrational and thermal energies increase, as does the vertical migration. Additionally, there is a decrease in the translational order. In the fluid state, there is no longer any observed crystalline structure. These states were identified by examining orientational correlation functions, motion of the particles, viscosity, interaction cross sections and self-diffusion. Each of these states can be seen in Figure 1.8.



(a) (b) (c) (d)

Figure 1.8: Showing the observed phases of a plasma crystal: (a) the crystalline state, (b) the floe and flow state, (c) the vibrational state and (d) the fluid-like state. (Images courtesy of H. Thomas, Max Planck Institute for Exterrestrial Physics, Garching, Germany)

To extract a kinetic temperature, the motion of the particles was observed by scattering laser light from the microparticles and recording video images of the scattered light. From the video images that were acquired a velocity for each of the microparticles was obtained from the measured displacements and the time between acquired video images. This is possible because the lower particle number density of the plasma crystals permits each particle to be clearly identified and tracked over time. Using these velocities, one of four techniques were used to extract a kinetic temperature of the microparticle component, T_j. It is noted that the index, j, refers to a symbol ($j = M$, MD, MKE, NM) denoting the method used to obtain the temperature and that this temperature T_j refers to the quantity k_BT_j, where k_B is Boltzmann's constant. It is noted that three of the four methods described below (Maxwellian, Mean Kinetic Energy and Normal Modes) measure the same temperature.

In the first method, which will be denoted by the symbol M, the distribution of measured velocities was fit to a Maxwellian velocity distribution, Equation 1.9.

$$f = \sqrt{\frac{m}{2\pi T_M}}\exp\left\{-\frac{m\left(v - v_d\right)^2}{2T_M}\right\} \tag{1.9}$$

where v is the velocity of a particle, v_d is the drift velocity, m is the mass of the dust grain and T_M is the kinetic temperature of the microparticle component. From the fit, a temperature is then extracted.

In the second method, which will be denoted by the symbol MD, a temperature was extracted by measuring a velocity from the maximum observed displacement, Δx_{max}. By knowing the time, t, over which this displacement was measured, Equation 1.10 can be used to extract a temperature, T_{MD}. It is noted that this is a measure of the maximum energy, not a bulk temperature measurement.

$$T_{MD} = m\left(\frac{\Delta x_{\max}}{t}\right)^2 \tag{1.10}$$

In the third method, which will be denoted by the symbol MKE, the mean kinetic energy of all of the dust particles is calculated. This energy is then equated to a temperature using Equation 1.11.

$$T_{MKE} = m\left\langle \left(v - \langle v \rangle\right)^2 \right\rangle \tag{1.11}$$

It is noted that this method may overestimate the temperature, for a number of reasons [66]. If Equation 1.11 is expanded, the temperature is defined as the difference between the total kinetic energy and the mean drift energy. The kinetic energy has contributions from the thermal motion and correlated wave motion. In measuring the kinetic temperature, one is interested only in the contribution from the thermal motion. However, it is difficult to compute the kinetic energy stored in coherent oscillations from experimental data. This difficulty lies in the way in which the data is acquired. Identification of coherent oscillations requires observing the motion of many particles over large distances. However, an accurate measurement of the microparticle velocity requires visualizing a small number of particles. In a number of the earlier experiments, only a small region of the plasma crystal was imaged. As a result, the energy stored in coherent oscillations may be underestimated. In a similar vein, the drift velocity can be difficult to measure. If the sampling time is large, compared to the time for drift velocities to change direction, then the drift velocity is averaged to zero. With care, it is possible to address these concerns [66]. However, it is likely that the temperatures reported in measurements 1, 2, 3 and 5 in Table 1.1 may be overestimated. It is unclear what error is introduced in these measurements due to these concerns. Nonetheless, it is also the case that the true temperature will be strongly correlated with the reported values.

The final method used to determine the kinetic temperature of the microparticle component involves examining the normal modes of the plasma crystal [67]. In this approach, the normal modes are computed from the dynamical matrix given in Equation 1.12.

$$E_{\alpha,\beta,i,j} = \frac{\partial^2 E}{\partial r_{\alpha,i} \partial r_{\beta,j}} \tag{1.12}$$

where $r_{\alpha,i}$ denotes the x or y coordinate of the i^{th} particle with (α, $\beta = x$, y) and E is the total energy of the system, given by Equation 1.13.

$$E = \tfrac{1}{2} m \omega_o^2 \sum_{i=0}^{N} r_i^2 + \frac{Z^2 e^2}{4\pi\varepsilon_o^2} \sum_{i>j}^{N} \frac{\exp\left\{-\frac{r_{ij}}{\lambda_D}\right\}}{r_{ij}} \tag{1.13}$$

Here, the first term denotes the potential energy of the radial confinement with resonance frequency ω_o and the second term denotes the screened Coulomb interaction between particles with screening length λ_D. It is noted that the radial position of the i^{th} particle, r_i, is with respected to the center of mass and $r_{ij} = |r_i - r_j|$. The eigenvectors of the matrix defined by Equation 1.12 are the normal modes. One then projects the measured velocities, v_i, are projected onto the normal modes vectors, $e_{i,l}$ as defined in Equation 1.14.

$$v_l(t) = \sum_{i=1}^{N} \vec{v}_i(t) \cdot \vec{e}_{i,l} \tag{1.14}$$

where the normal modes are denoted by the index $l = 1, \ldots, 2N$. From this, the normal modes are calculated in the form of a spectral density

$$S_l(\omega) = \frac{2}{T}\left| \int_{-T/2}^{T/2} v_l(t) \exp(-i\omega t)\, d\omega \right|^2 \tag{1.15}$$

Each mode will have an effective temperature given by Equation 1.16.

$$T_{NM} = \frac{1}{2N} \sum_{l=0}^{2N} m \int_0^{\infty} S_l(\omega)\, d\omega \tag{1.16}$$

where $\left\langle v_l^2 \right\rangle = \int_0^{\infty} S_l(\omega)\, d\omega$. Summing the contribution of all the modes will give the mode temperature, T_{NM}.

The measurement of the kinetic temperature that were made during the observed phase transition from an ordered to a disordered state are summarized in Table 1.1. All of the reported measurements, with the exception of Measurement 4, were performed using either a plasma crystal or a dust cluster. In each of these measurements, the motion of the microparticles was observed by illuminating the microparticles with a laser sheet.

As a result, there is an implicit assumption that the motion in these experiments is two dimensional, *i.e.* the motion is confined in the plane illuminated by the laser sheet. For these plasma crystal or dust clusters, this is a reasonable approximation. This assumption may not be as valid for Measurement 4, resulting in an underestimation in the kinetic dust temperature.

In all of these experiments, there is a common observation: a significant increase in the kinetic temperature of the microparticle component is observed as the system moves from the crystalline state to the fluid state. Further, the kinetic temperature in the fluid-like state is found to be larger, and in most cases significantly larger, than the temperature of any of the plasma components (*e.g.* the electrons, ions and background neutrals). Finally, it is noted that the kinetic temperature of the dust particle is very different than the surface temperature of the dust particle. Indeed, it has been observed that the surface temperature of the dust particles is close to room temperature [72].

In Experiment 1 [65], a plasma crystal composed of monodisperse melamin-formaldehyde microspheres having a diameter of 9.4 μm was generated at a neutral gas pressure of 640 mTorr in an rf discharge plasma. A phase transition was induced using two mechanisms: reduction of the neutral gas pressure and increasing the applied rf power. The motion of the microparticles was observed using a 50 fps video camera and a kinetic temperature was computed using the MKE method described above. Further, the temperature that was extracted using the MKE method was used to generate a Maxwellian velocity distribution that fit the experimentally observed velocity distribution with excellent agreement. It is noted that the temperature of 0.7 eV represents the

	State	Pressure [mTorr]	T_{dust} [eV]	Method	Reference
1	Ordered	640	0.7*	MKE	Melzer *et al.*, Phys. Rev. E 53, 2757 (1996)
	Disordered	290	~50		
2	Crystal	315	$T_x = 0.036$; $T_y = 0.019$	M, MKE	Thomas and Morfill, Nature (London) 379, 806 (1996).
	Flow and Floe	270	$T_x = 0.039$; $T_y = 0.026$		
	Vibrational	218	$T_x = 2.375$; $T_y = 3.434$		
	Disordered	165	$T_x = 4.428$; $T_y = 4.357$		
3	Ordered	90	~.1	MKE	Quinn and Goree, Phys. Plasmas, 7, 3904 (2000)
	Disordered	30	~10		
4	Ordered	100	~1	MKE, NM	Ichiki *et al.*, Phys. Rev. E, **70**, 066404, (2004)
	Disordered	70	~18		
5	Ordered	0.35	24	MD	Zhakhovskii *et al.*, JETP Letters, 66, 419 (1997)
	Disordered		2.4×10^3		
6	Crystal	135	0.037	MKE, NM	Ivanov and Melzer, Phys. Plasma, 12, 0721100 (2005)
	Liquid	38	~10		

Table 1.1: The results from thermal studies made during the phase transitions observed in a plasma crystal. It is observed that the experiments were performed over a wide range of experimental conditions, but they all exhibited the same key result: a notably larger temperature in the fluid-like regime.

resolution limit of the video camera used. As a result, it is likely that the kinetic temperature in the crystalline state was lower than what is reported.

In Experiment 2 [10], a plasma crystal composed of monodisperse melamin-formaldehyde microspheres having a diameter of 6.9 μm was generated at a neutral gas pressure of 315 mTorr in an rf discharge plasma. A phase transition was induced by reducing the neutral gas pressure. The motion of the microparticles was observed using a CCD camera and a kinetic temperature was computed using the MKE and M methods described above. It was observed that these two methods yielded the same results.

In Experiment 3 [66], a plasma crystal composed of monodisperse melamin-formaldehyde microspheres having a diameter of 9.4 μm was generated at a neutral gas pressure of 90 mTorr in an rf discharge plasma. A phase transition was induced by reducing the neutral gas pressure. The motion of the microparticles was observed using a 30 fps CCD camera with a shutter speed of 1/250 s. The kinetic temperature was computed using the MKE method described above.

In Experiment 4 [67], a plasma crystal composed of monodisperse melamin-formaldehyde microspheres having a diameter of 9.55 μm was generated at a neutral gas pressure of 100 mTorr in an rf discharge plasma. A phase transition was induced by decreasing the neutral gas pressure. The motion of the microparticles was observed using a 50 fps video camera and a kinetic temperature was computed using the MKE and NM methods described above. The temperatures computed using these two methods were in agreement.

In Experiment 5 [73], a plasma crystal composed of monodisperse melamin-formaldehyde microspheres having a diameter of 10.24 μm was generated at a neutral gas pressure of 0.35 mTorr in a dc discharge plasma. In this experiment, the suspended dust cloud exhibited two distinct regions. In the upper region of the cloud, the microparticles existed in strongly-coupled state, while the microparticles in the lower region of the cloud were in a fluid-like state. Unlike the other experiments, the motion of the microparticles may not haven been limited to the two-dimensional plane of the laser sheet. However, it is also noted that a detailed study of the temperature was not preformed and that the reported temperatures represent a maximum energy in the illuminated plane of the suspended dust cloud, not a bulk measurement as is the case in the other experiments discussed here.

In Experiment 6 [68], a dust cluster composed of monodisperse melamin-formaldehyde microspheres having a diameter of 7.7 μm, 9.5 μm or 12.26 μm was generated at a neutral gas pressure of 135 mTorr in an rf discharge plasma. A phase transition was induced using two mechanisms: reduction of the neutral gas pressure and increasing the applied rf power. The motion of the microparticles was observed using a 25 - 100 fps video camera, depending on the state of the dust cluster. The kinetic temperature was computed using the MKE and NM methods described above. It is noted that each method yielded the same kinetic temperature for the microparticle component. Finally, it was observed that smaller particle obtained a higher kinetic temperature at all values of neutral gas pressure.

In total, the results of experiments 1-4 and 6 in Table 1.1 suggests two significant results that will be evaluated in this work. First, in all cases, the distribution of the microparticle velocities was determined to obey a Maxwellian distribution for these strongly-coupled dusty plasmas. It will be shown that this is also valid for the weakly-coupled systems studied in this work. Second, as discussed at the outset of this section, increasing the neutral gas pressure is an effective method of controlling the microparticle temperature. In particular, increases in the neutral pressure corresponds to a lowering of the dust temperature. This will be evaluated for weakly-coupled dusty plasmas composed of both uniform and non-uniform particle size distributions in this work.

1.3.2 THERMAL STUDIES INVOLVING LASER HEATING

While the experiments in Section 1.3.1 focused on measuring the kinetic temperature of the microparticle component that naturally occurred at a given set of experimental conditions, there have also been a number of studies that involved the active modification of the dust particle temperature [69, 70]. In these experiments, a plasma crystal or dust cluster was generated and heated using a laser. To simulate an equilibrium heating process, the laser excitation was randomly applied to various locations in the plasma crystal. The results of these experiments are summarized in Table 1.2 and are discussed briefly below.

	State	Pressure [mTorr]	T_{dust} [eV]	Method	Reference
1	Crystalline Liquid	37.5 or 75	0.1 8	MKE	Wolter *et al.*, Phys Rev E, **71**, 036414 (2005)
2	Crystalline Liquid	5	~0.1 ~20	MKE	Nosenko e*t al.*, Phys Plasmas, **13**, 032106 (2006)

Table 1.2: Showing the results from thermal studies where a phase transition was induced by heating the plasma crystal using an external means (laser heating). As observed in Section 1.3.1, there is a notably larger temperature in the fluid-like regime.

In the first of these experiments [69], a dust cluster consisting of 10 – 200 monodisperse melamine formaldehyde microspheres having a diameter of 9.55 μm or 12.25 μm was generated at a neutral gas pressure of 37.5 or 75 mTorr in an rf discharge plasma. A phase transition was induced by heating the dust cluster using a variable power (0-200 mW) Nd:YAG laser. The motion of the microparticles was observed using a 50 fps CMOS camera and a kinetic temperature was computed using the MKE method previously described. Further, it was observed that the distribution of velocities was Maxwellian and excellent agreement between the width of the velocity distribution and the kinetic temperature computed using the MKE method was observed. As was observed in the experiments discussed in Section 1.3.1, there was a significant increase in the kinetic temperature of the microparticle component. It is noted that the experiment from [69] highlighted in Table 1.2 was performed on a dust cluster consisting of 44 microparticles. A final observation from this experiment was that the heating was much stronger, under identical experimental conditions, for smaller particles.

In the second of these experiments [70], a plasma crystal consisting of approximately 6700 melamine formaldehyde microspheres having a diameter of 8.09 μm was generated at a neutral gas pressure of 5 mTorr in an rf discharge plasma. A phase transition was induced by heating the plasma crystal using two laser beams ($\lambda = 532$ nm) from opposite sides of the plasmas crystal with a total heating power from 0 to 10.55 W. A region of the plasma crystal consisting of 1100 microspheres was observed using a 29.97 fps video camera and a kinetic temperature was computed using the MKE method described above. Further, it was observed that the distribution of velocities was

Maxwellian. As a result of the heating, a phase transition from the crystal to disordered state was observed. As was observed in the experiments discussed in Section 1.3.1, there was a signifigant increase in the kinetic temperature of the microparticle component, in this case from ~0.1 eV in the crystal state to ~20 eV in the disordered state

1.3.3 OTHER THERMAL STUDIES

Beyond the experiments described in the previous sections, there have been four other experiments that have measured the kinetic dust temperature [74-76]. One of these experiments (Experiment 1 in Table 1.3) focused on examining wave modes in a dusty plasma in the various states of a plasma crystal, two of these experiments (Experiments 2 and 3 in Table 1.3) involved looking at shocks in a complex dusty plasma, while the fourth (Experiment 4 in Table 1.3) measured the kinetic temperature of a dust cloud trapped in an rf discharge over a wide range of experimental conditions. Unlike the experiments previously discussed, there was no observed phase transition in Experiment 4 in Table 1.3. The results of these experiments are summarized in Table 1.3 and are discussed briefly below.

	State	Pressure [mTorr]	T_{dust} [eV]	Method	Reference
1	Crystaline Partially Melted Liquid	7.5	0.037 14.5 54.9	M	S. Nunomura, *et al.*, Phys Rev Lett, 94, 045001 (2005)
2	Before Shock After Shock	13.5	0.5 ~300	MKE	D. Samsonov, *et al.*, Phys Rev Lett, **92**, 255004 (2004)
3	Ahead, H Ahead, V Behind, H Behind, V	375	1.7 9.2 2.0 3.1	M	V.E. Fortov et. al., Phys Rev E, **71**, 036413 (2005)
4	Axial Azimuthal	10-50	≤ 500 ≤ 1000	MKE	M. Schabel *et al.*, J. Appl . Phys. **86**, 1834 (1999).

Table 1.3: Showing other thermal studies involving a dusty plasma.

In Experiment 1 [74], a plasma crystal composed of ~5000 monodisperse plastic microspheres having a diameter of 8.9 μm was generated at a neutral gas pressure of 7.5 mTorr in an rf discharge plasma. A phase transition was induced by adding a small amount of 12.7 μm monodisperse plastic microspheres. The motion of the microparticles was observed using a video camera operating at 22 – 154 fps and a kinetic temperature was computed using the M method described above. It is noted that the velocity distributions were observed to be very close to Maxwellian.

In Experiment 2 [75], a plasma crystal composed of monodisperse plastic microspheres having a diameter of 8.9 μm was generated at a neutral gas pressure of 7.5 mTorr in an rf discharge plasma. A 0.1 mm diameter tungsten wire located 4 mm below the plasma crystal was pulsed to -100 V for 50 ms, inducing a compressional shock in the plasma crystal. This shock caused the plasma crystal to melt immediately behind the propagating shock wave. The motion of the microparticles was observed using a video camera and a kinetic temperature was computed using the MKE method described above.

In Experiment 3 [76], a dust cloud composed of monodisperse plastic microspheres having a diameter of 1.87 μm was generated at a neutral gas pressure of 375 mTorr in a dc discharge plasma. To induce a shock, a battery of 1.2 mF capacitors was charged to 1.2 kV and discharged through a 16 turn copper coil using a 14 Ω resistor (~90 A of current through the coil). This pulse causes the plasma to move upward for a very short period of time. During this time, the dust particles are no longer in mechanical equilibrium and begin to fall under the influence of gravity. When the plasma returns to its original configuration after the pulse has ended, the particles are pulled back to their original location. The particles that are located at a lower position in the vertical direction have fallen a greater distance and as a result, a region of high dust density is created. The motion of the microparticles in the horizontal (H) and vertical (V) directions was observed using a video camera operating at 1000 fps and a kinetic temperature was computed using the M method described above.

In Experiment 4 [77], a dust cloud composed of monodisperse polystyrene latex microspheres having a diameter of 10.2 μm was generated over a range of neutral gas pressure from 10 to 50 mTorr in an rf discharge plasma. The motion of the microparticles was measured using laser doppler velocimetry, LDV, and confirmed using LLS techniques. From the velocities measured using the LDV technique, a kinetic temperature was computed using the MKE method described above for the motion parallel (azimuthal), $T_{\parallel}$, and perpendicular (axial), $T_{\perp}$, to the electrode. For the particle concentrations observed in this experiment, the motion of the microparticle component was observed to be random and uncorrelated. Additionally, the velocity distributions measured were independent of the filtering conditions for the LDV measurement. There are two key results that were observed in this experiment that are relevant to the work presented in this dissertation. First, the temperature increased with decreasing neutral gas pressure and increasing power. Second, the observed temperatures were anisotropic, with $T_{\perp} : T_{\parallel} \approx 2$ across a wide range of experimental conditions.

1.4 SCOPE OF THIS THESIS

In this dissertation, two new results are presented. First, the use of a new diagnostic, stereoscopic particle image velocimetry (stereo-PIV), in the study of dusty plasmas is introduced. Using this new diagnostic, the first detailed measurements of the thermal properties of a dc glow discharge dusty plasma are then presented. In Chapter 2, the experimental apparatus and procedures used in the experiments presented in this dissertation are discussed. Chapter 3 will discuss simulations of the PIV measurement technique, while Chapter 4 will present the first experimental measurements of the thermal properties of a weakly-coupled dusty plasma in a dc glow discharge. Finally, conclusions and directions for future work will be discussed in Chapter 5.

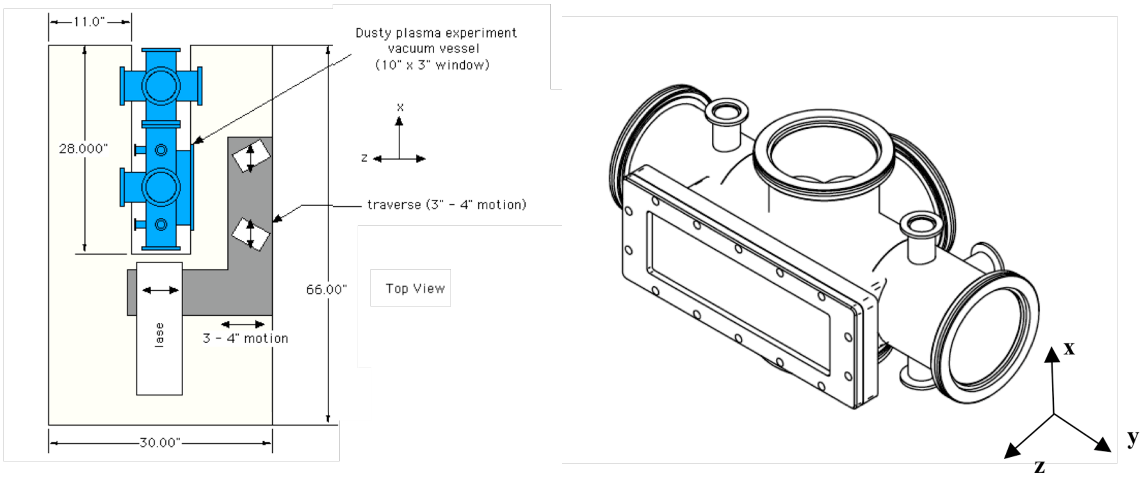
CHAPTER 2: APPARATUS

The work described in this dissertation was performed using the 3D-Dusty Plasma Experiment (3DPX) device [78]. As is the case with all plasma devices, 3DPX consists of a number of subsystems, including the vacuum vessel, vacuum and gas systems, plasma and dusty plasma generation systems, and diagnostic systems; each of which will be discussed in the following chapter. Section 2.1 describes the 3DPX device [78], as well as the techniques employed for the plasma and dusty plasma generation. Section 2.2 will describe the PIV technique and the stereoscopic particle image velocimetry diagnostic [79], which is both the centerpiece of the 3DPX device and the primary diagnostic for the experiments described in this dissertation. Section 2.3 describes how measurements of the particle density are performed [80].

2.1 EXPERIMENTAL DEVICE

2.1.1 VACUUM VESSEL

The 3DPX vacuum vessel [78] is composed of two 10 cm (4”) inner diameter stainless steel six-way crosses. One of these crosses, which will be referred to as Section 1, is a modified six-way cross having a length of 47.5 cm. The most prominent feature of Section 1 is a large rectangular window, 7 cm tall by 24 cm wide, on the front of the chamber. Experiments are performed in Section 1 of the device, where the large rectangular window allows for optimal optical access. Section 2 of the 3DPX vacuum vessel is a standard ISO100 stainless steel six-way crosses. This section holds all of the vacuum pumps and gas feed systems. In this configuration, the experimental region of the system is separated from the gas flows that are generated by the gas and vacuum subsystems. The overall length of the chamber is 71 cm (28”). A schematic of the 3DPX device is shown in Figure 2.1.



(a) (b)

Figure 2.1: A schematic of the 3DPX device showing (a) the top view of the experimental setup and (b) a detailed drawing of Section 1 of the experimental device where the experiments are conducted. The coordinate system shown in this figure will be used throughout this document. The *x*-*y* plane is defined by the plane of the laser sheet, while the *z*-direction is perpendicular to the laser sheet. It is noted that gravity acts in the *y*-direction.

A photograph of the 3DPX device can be seen in Figure 2.2.

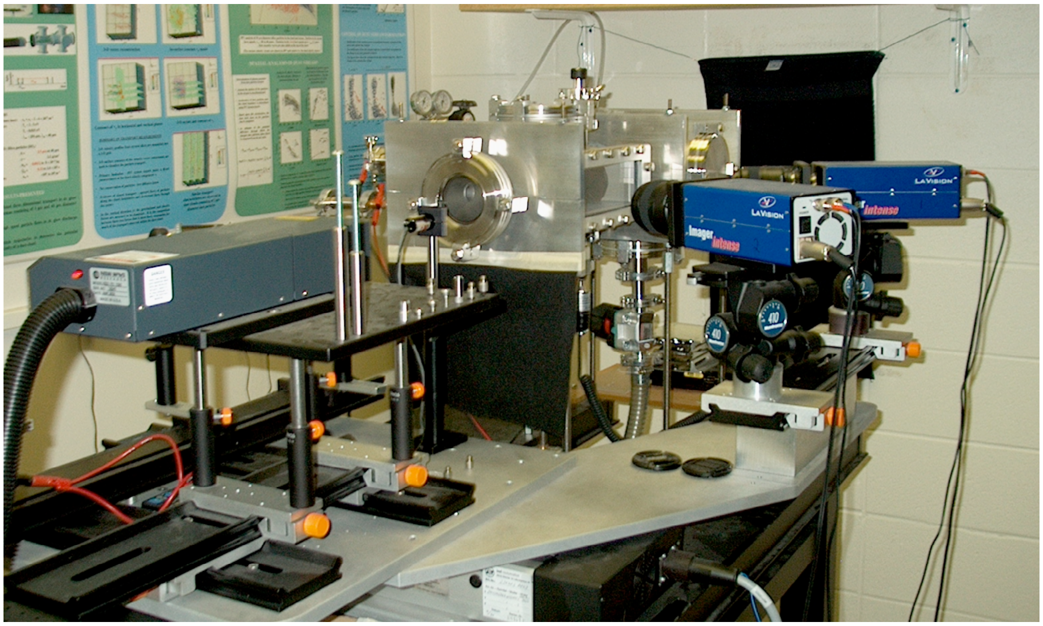


Figure 2.2: A photograph of the 3DPX device and the stereoscopic particle image velocimetry diagnostic system.

2.1.2 VACUUM AND GAS SYSTEMS

A dynamic vacuum is maintained in 3DPX using a 1.0 *l*/s ULVAC GLD-051 roughing pump. An ultimate base pressure of $p < 10$ mTorr can be maintained in the 3DPX device. For most dusty plasma experiments performed in 3DPX, experiments are done at neutral gas pressures ranging from $p = 90$ to 300 mTorr.

Gas, typically argon, enters the 3DPX vacuum vessel through a two-stage system. First, the gas flows through a ¼” diameter NUPRO cutoff valve which maintains a hard seal for the system. Next, the gas is sent through an Edwards LV10K fine control leak valve. Both the gas feeds and vacuum pumps are located in section 2 of 3DPX to minimize neutral gas flows that may effect the confinement and behavior of the dust particle suspended in the plasma. An photograph of the gas and vacuum subsystems can be seen in Figure 2.3.

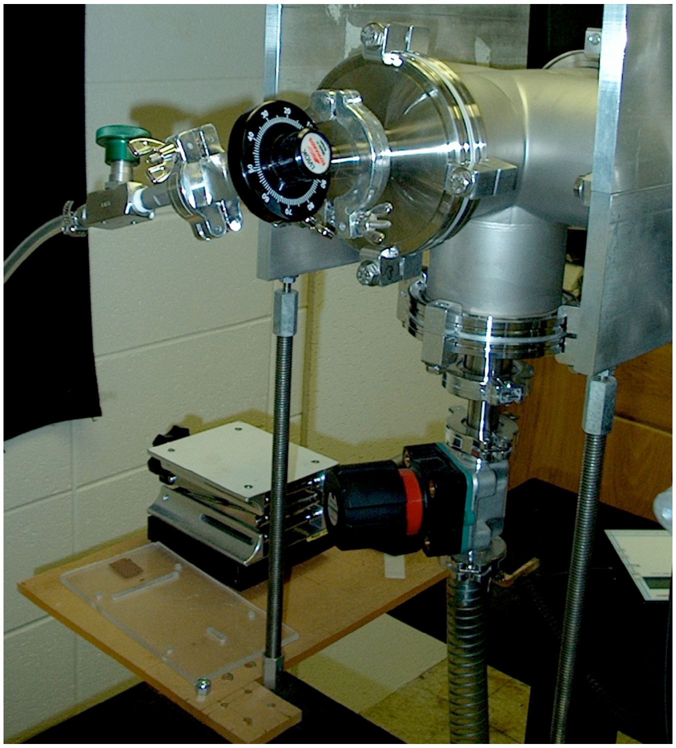


Figure 2.3: A photograph of the gas and vacuum subsystems on the 3DPX device. The tube leaving the bottom of the photograph is connected to a roughing pump.

2.1.3 PLASMA AND DUSTY PLASMA GENERATION

A dc glow discharge plasma is generated in 3DPX using a pair of electrodes that are biased relative the grounded chamber wall, *i.e.* an anode and cathode. The electrodes are square stainless steel plates, each having a length of 2.5 cm. The electrodes are soldered to 18 gauge wire and enter the 3DPX device through ½" and ¼" coupling assemblies that are located on the back side of the chamber. The coupling assembly consists of two, ¼" and one, ½" coupling equally spaced feedthroughs on an ISO100 flange. Each electrode can be moved in the *z*-direction. A side view of this setup can be seen in Figure 2.4.

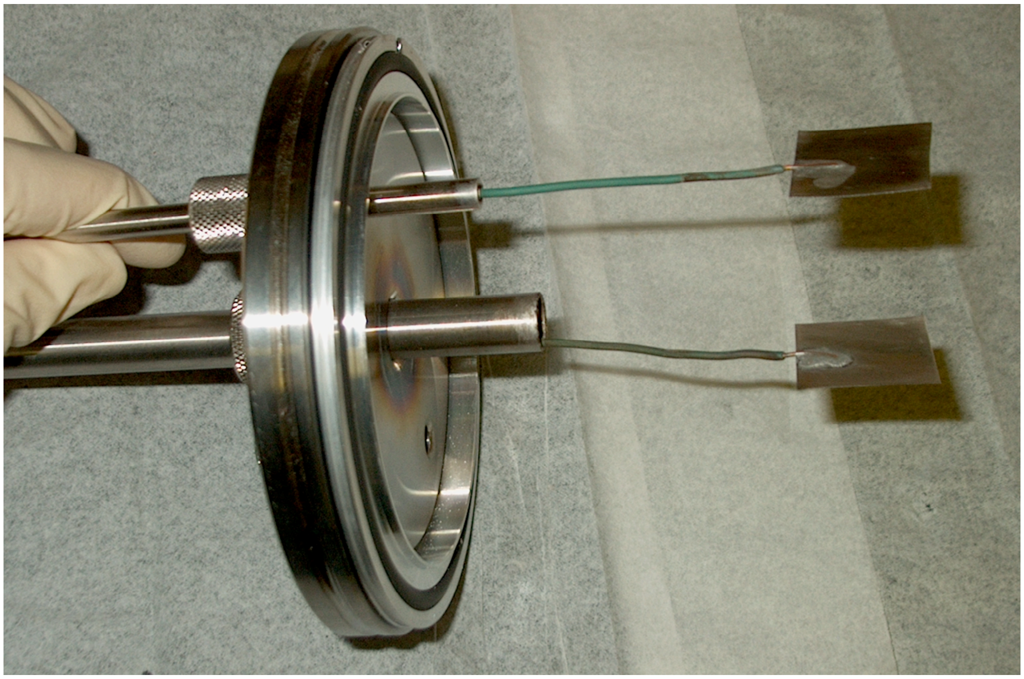


Figure 2.4: A photograph of the flange and electrode assembly used to generate a plasma in the 3DPX device.

To generate a plasma, argon gas is introduced into the chamber until the desired dynamic pressure is achieved (typically, $p \sim 100$ mTorr). The anode is biased at $V_{anode} \sim 250$ V and the cathode is biased at $\mathrm{V}_{\mathrm{cathode}} \sim -100$ V, relative to the grounded chamber wall. Once the plasma is generated, the pressure and bias voltages are adjusted until the desired experimental conditions are reached.

In these experiments, the dust is located on a small tray located below the electrodes. For experiments using alumina microparticles (Al_2O_3, $r_d = 0.6 \pm 0.25$ μm), the dust tray is seen in Figure 2.5(a) was used. For experiments involving monodisperse dust, the dust tray seen in Figure 2.5(b) is used.

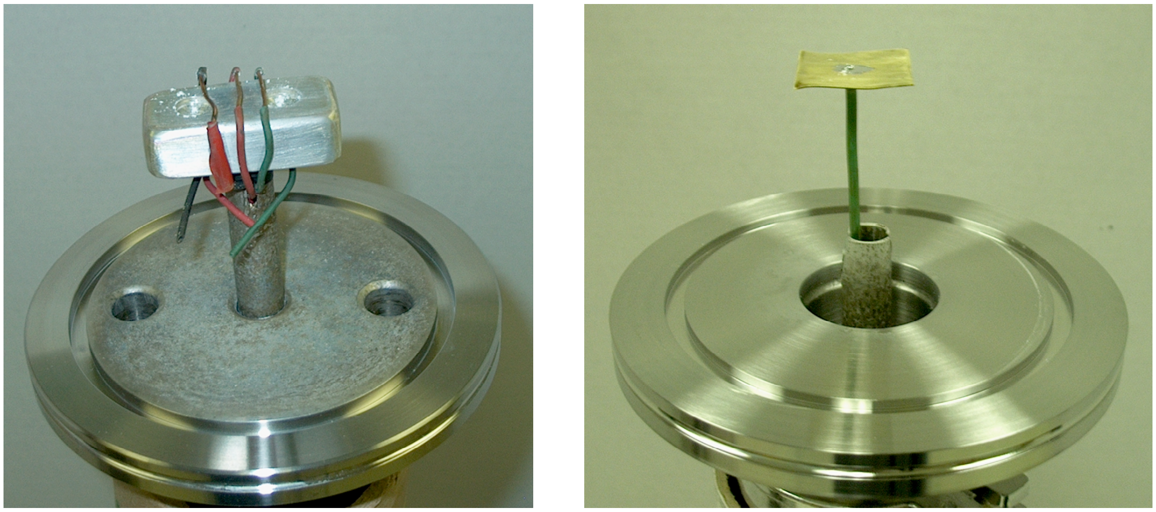

(a) (b)

Figure 2.5: Photographs of the dust trays used in the experiments involving (a) polydisperse and (b) monodisperse microparticles. In all of the experiments presented in this dissertation, the dust tray is electrically floating.

The dust tray rests on a hollow, ½” diameter cylindrical stainless steel tube that enters the chamber through a linear feed through mounted an ISO100 flange located at the bottom of Section 1 of the 3DPX device. The tray can be moved vertically (*y*-direction) in the vacuum vessel and is electrically floating. Once the plasma has formed, a flux of ions and electrons from the plasma charges dust particles according to Equations 1.1 – 1.5. If enough charge has accumulated on the dust particle to allow the electric force on these dust particles to exceed the gravitational and adhesion forces that hold the dust particles to the surface of the chamber, the dust particles enter the plasma and become levitated. Typically, these particles become trapped in the sheath of the anode, immediately below the anode, or the sheath of the dust tray, immediately above the dust tray. This is depicted in Figure 2.6. For the experiments in this dissertation, the dust clouds were primarily suspended in the anode spot trap.

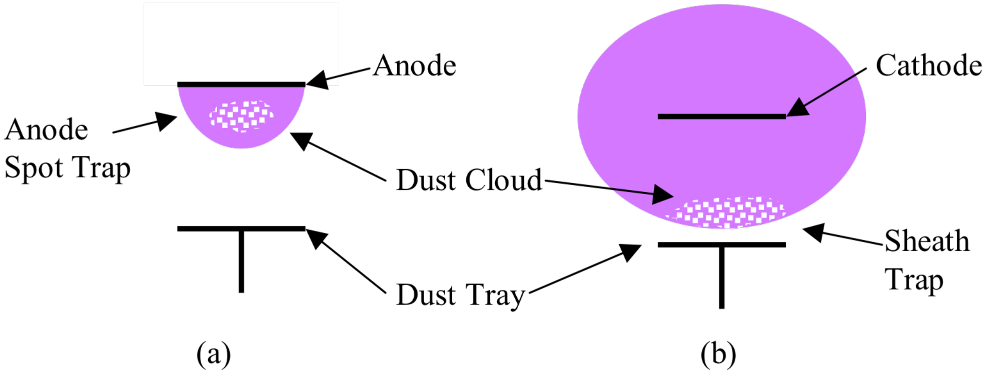
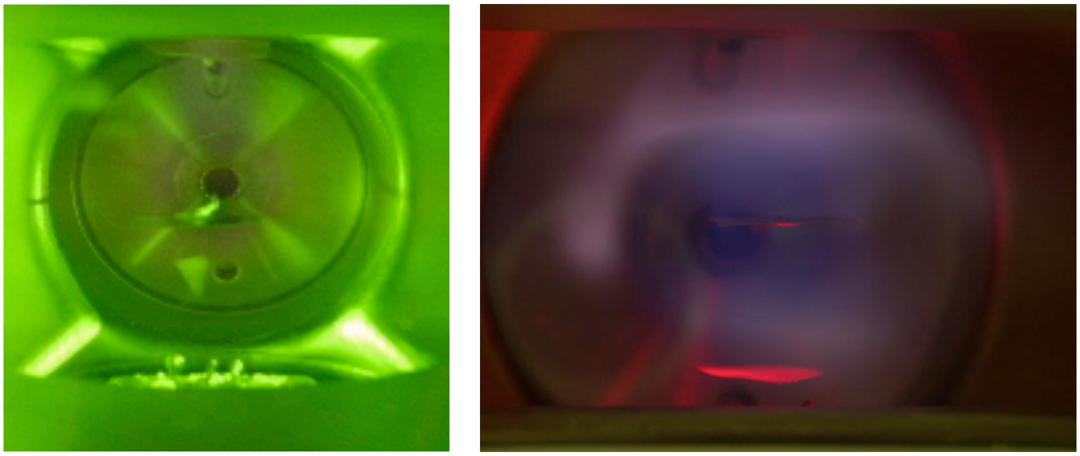


Figure 2.6: Schematic drawing showing the locations where dust is typically trapped in the 3DPX device. In (a), the dust is suspended below the anode in an anode sheath trap. In (b), the dust is suspended above the dust tray, in a sheath trap.

Photographs of dust clouds suspended in these regions can be seen in Figure 2.7. As is the case in other experiments involving a dc glow discharge [81-83], the dust clouds in 3DPX are characterized by sharply defined boundaries between the dust cloud and surrounding plasma.



(a) (b)

Figure 2.7: Images of a dust cloud suspended in (a) the anode sheath trap and (b) the sheath trap. The dust cloud is illuminated using (a) a Nd:YAG laser ($\lambda = 532$ nm) and (b) a diode laser ($\lambda = 632$ nm). The purple glow that is visible in the background is an argon plasma

2.2 PARTICLE IMAGE VELOCIMETRY

Much like other plasmas, dusty plasmas are an example of a complex, self-organized non-linear system. However, dusty plasmas have the added advantage of allowing for the direct visualization of one component of the system on the kinetic level. Nonetheless, diagnostic studies of dusty plasmas often presents a non-trivial problem. The standard diagnostic probe techniques of plasma physics, *e.g.* in-situ probes such as Langmuir or emissive probes, perturb the suspended particle cloud as seen in Figure 2.8. As a result, it is necessary to study these systems using non-invasive, typically optical, techniques.

The most common of the optical techniques is laser light scattering (LLS) [8, 84], depicted in Figure 2.9. Generally, a laser, whose wavelength is shorter than the size of the microparticles, is used to illuminate the particle cloud. The light that is scattered by the particles is then recorded, typically using a CCD, CMOS or still digital camera. In practice, a CW red ($\lambda \sim 632$ nm) or green ($\lambda \sim 532$ nm) laser is used for these types of measurements.

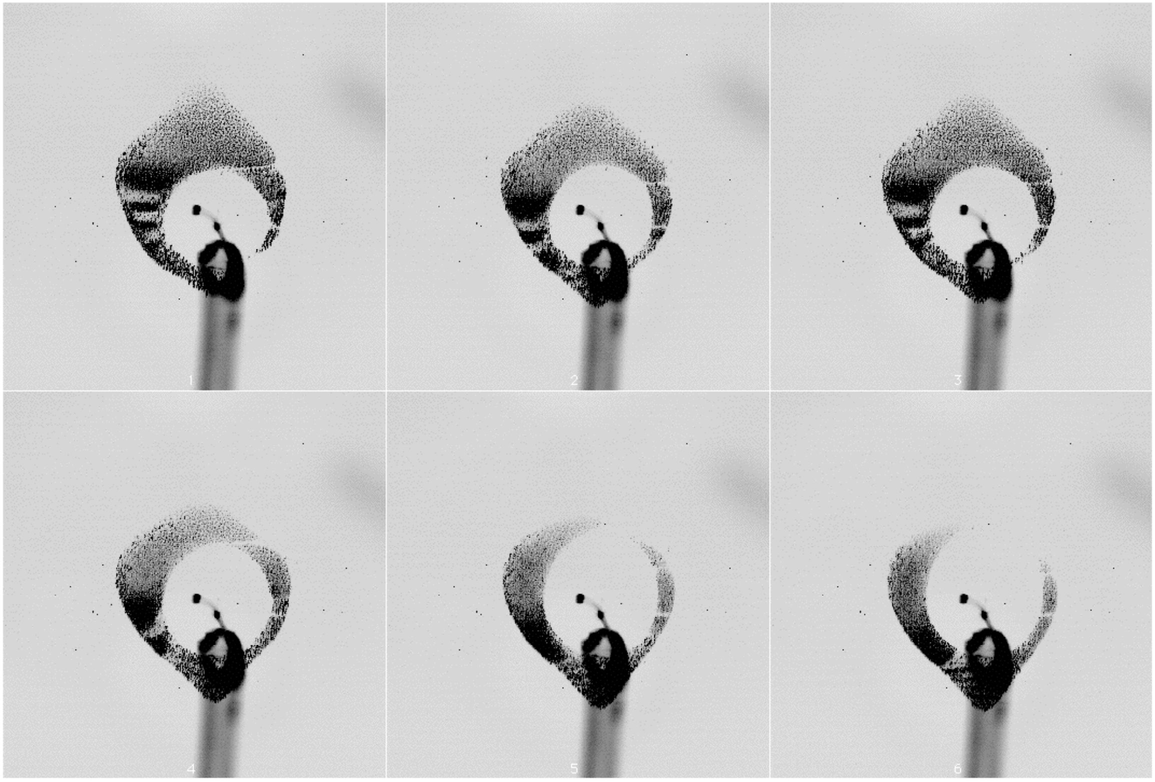


Figure 2.8: Video images of a dust cloud in the presence of a Langmuir probe. As a bias is applied to the probe, a void is formed inside the dust cloud.

Many of the LLS techniques were developed in the fluid mechanics community and are collectively known as particle-based quantitative imaging techniques [84]. The simplest of these LLS techniques are most effective for low density dusty plasmas were the motion of the individual particles can be resolved. For higher density cases, *e.g.* when particle overlap can become problematic, more advanced LLS techniques becomes necessary.

When one is interested in measuring the velocity field in a flow, there are a number of quantitative imaging techniques that can be used. In a typical fluid measurement, these techniques involve injecting discrete particles into the flow field, a process known as "seeding", and then observing the motion of these tracer particles over a large region of the flow. For these techniques to yield meaningful information, there are two of conditions that these tracer particles must satisfy. First, they must be large enough to be observed optically. Second, the tracer particles must be neutrally buoyant with respect to the flow, *i.e.* they must be sufficiently small such that they travel with the fluid flow without disturbing the flow.

The most common of these particle-based quantitative imaging techniques [84] are particle streak velocimetry (PSV), particle tracking velocimetry (PTV), particle image velocimetry (PIV) and laser speckle velocimetry (LSV). The setup for each of these techniques is essentially the same. Light, from any number of sources, is used to illuminate the flow of interest. Typically, a laser beam is expanded into a two-dimensional light sheet, which is oriented in the direction of the flow of interest. Light from the laser sheet then scatters off of the tracer particles located in the flow of interest and is recorded by a camera that is oriented perpendicular to the laser sheet. It is noted that if digital imaging (*i.e.* a CCD camera) is used in the measurement, then the above techniques are denoted as digital, *i.e.* digital particle image velocimetry (DPIV) [85]. A typical setup for measuring two-dimensional flows is shown in Figure 2.9.

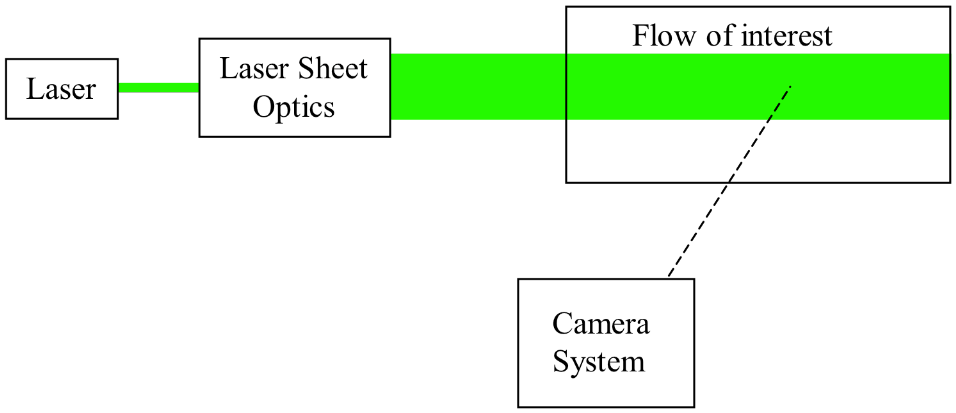
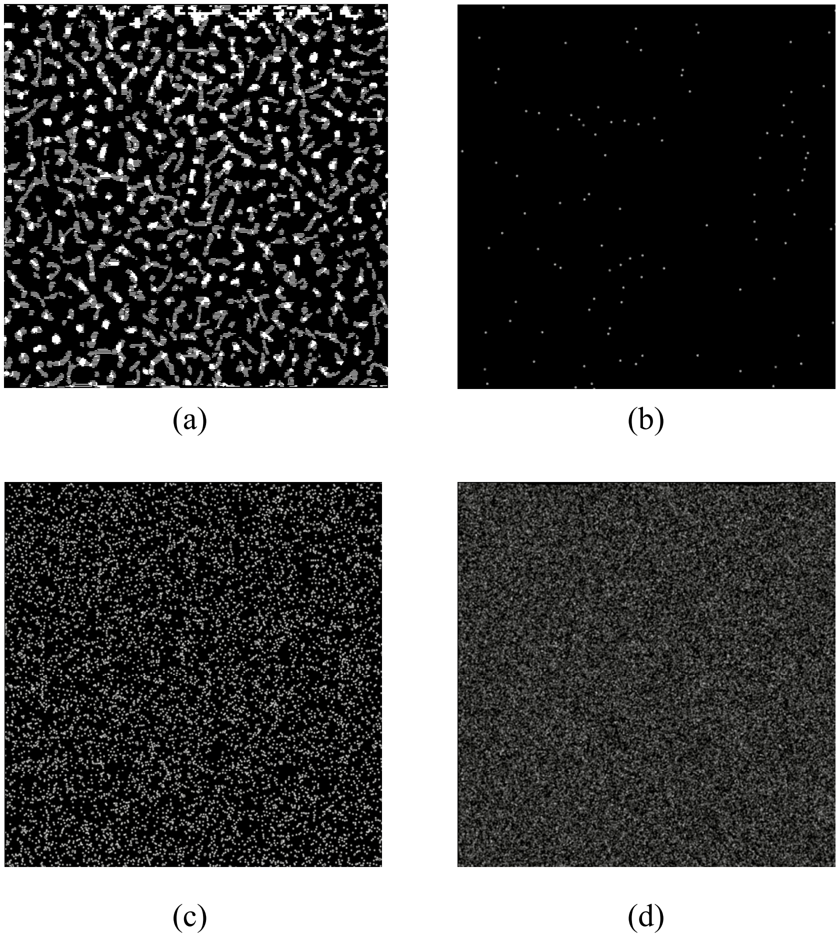


Figure 2.9: A schematic showing the typical setup for a two-dimensional quantitative imaging system.

There are, however, very important differences in these techniques. In particle streak velocimetry, PSV, the image exposure time is long relative to the time that a tracer particle occupies a given position. The result is a streak in the recorded image, the length of which can be used to extract a velocity using the exposure time. In particle tracking velocimetry, PTV, the seeding density of the tracer particles is sufficiently low that each particle image is clearly observable without overlap or interference with other imaged particles. Images are acquired in a short period of time so that the particles are observed as discrete dots and acquired with frequency sufficiently large to allow the motion of each individual particle to be tracked. In laser speckle velocimetry, LSV, the seeding density of tracer particles is sufficiently high that individual particles cannot be detected due to overlap with neighboring particles. As a result, one observes a speckle pattern. In particle image velocimetry, PIV, the seeding density of the tracer particles lies in an intermediate range between what is used in the PTV and LSV technique. Here, the individual tracer particles can be seen, but there is some overlap with neighboring particles. Figure 2.10 depicts images from each of these diagnostic techniques.



(a) (b)

(c) (d)

Figure 2.10: Characteristic images taken when using (a) particle streak velocimetry (Image courtesy of H. Thomas, Max Planck Institute for Exterrestrial Physics, Garching, Germany), (b) particle tracking velocimetry, (c) particle image velocimetry and (d) laser speckle velocimetry.

For the experiments presented in this dissertation, the PIV measurement technique was employed. The details of the PIV measurement technique are presented in the following sections. Section 2.2.1 discusses the basic features of the PIV measurement technique. Section 2.2.2 discusses the implementation of two-dimensional PIV technique. Section 2.2.3 discusses an extension of two-dimensional PIV technique to measure three-dimensional velocities. Finally, the system that was used in making the measurements described in this dissertation will be discussed in Section 2.2.4.

2.2.1 PRINCIPLES OF PARTICLE IMAGE VELOCIMETRY

Digital particle image velocimetry is a whole-field, non-invasive optical technique that allows instantaneous fluid flow measurements [85]. In the typical PIV experimental configuration [79, 86], a flow is seeded with micron-sized particles that are then carried by the fluid flow of interest. These tracer particles are sufficiently small that they travel with the flow of interest without disturbing it and are then illuminated by a light source. Typically, a pair of laser pulses, that are aligned to follow the same optical path, illuminate the tracer particles at a user-defined time interval, Δt_{laser}. The duration of the laser pulses is sufficiently small (on the order of nanoseconds) that the particles do not move more than 10% of their diameter. This rule of thumb value prevents streaking of particle images, which significantly reduces the quality of the computed velocities.

These laser pulses are normally shaped into a two-dimensional light sheet using an assortment of optical components. Light from the laser sheet will scatter off of the tracer particles and is then recorded using either a CCD or film camera that is, typically, oriented perpendicular to the laser sheet. These images, once in digital form, are divided into a series of interrogation regions of size $n \times n$ pixels. It is then possible to extract the mean displacement for all of the particles that are located in each interrogation region using the PIV technique. Using this displacement and the time between subsequent illuminations of the tracer particles, Δt_{laser}, one can compute a velocity.

A sketch of a "classic" PIV system, 2D PIV, is seen in Figure 2.11. Using the 2D PIV setup, one is able to recover the projection of the motion in the plane of the laser sheet, *i.e.* a single PIV measurement provides a "snapshot" of the flow velocities over the whole field.

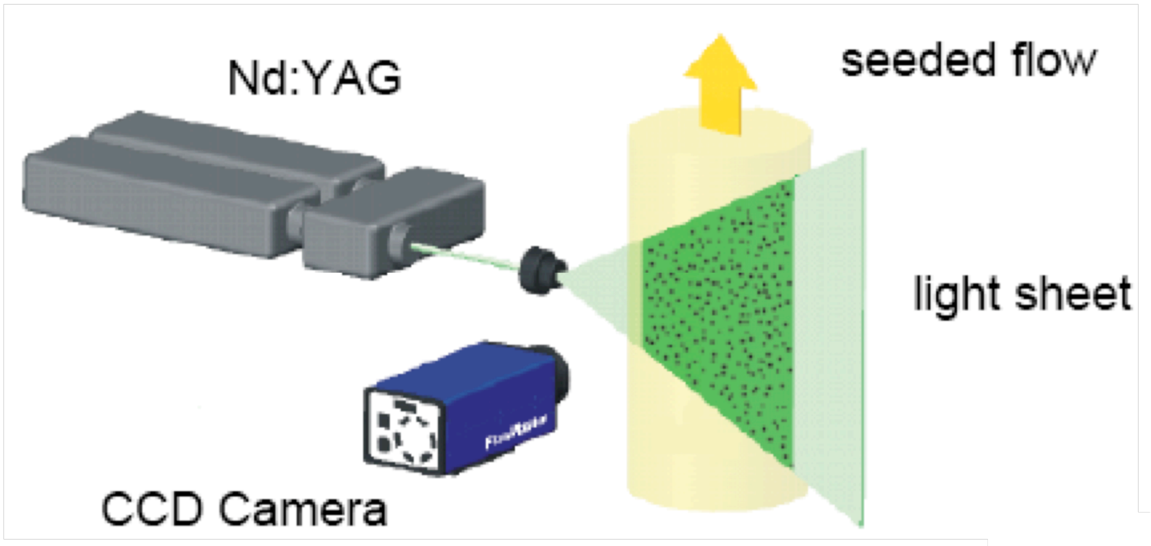
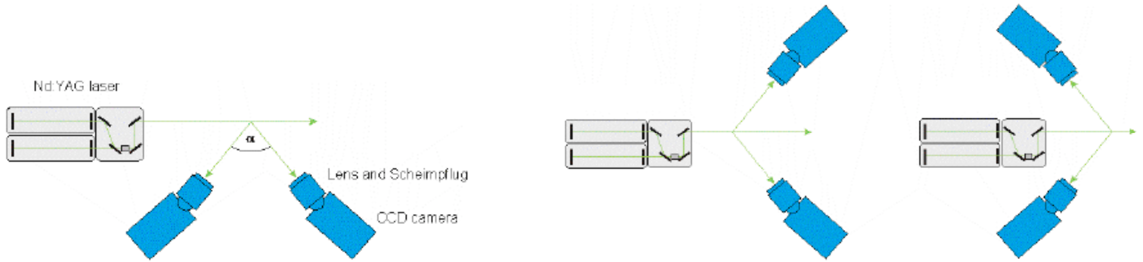


Figure 2.11: A cartoon depicting a 2D PIV system (Figure used with permission of LaVision, Inc.)

In the experiments that are discussed in this dissertation, an extension of the two-dimensional PIV technique, known as stereoscopic particle image velocimetry or stereo-PIV, is used [87]. Stereo-PIV techniques allow one to measure the three-dimensional velocities (v_x, v_y and v_z) of the tracer particles. In this approach, two cameras are oriented obliquely to the laser sheet, as seen in Figure 2.12. Here, each camera functions as a two-dimensional PIV system. Using the velocity profiles that are generated from each camera system ($v_{x,1}$, $v_{y,1}$, $v_{x,2}$, $v_{y,2}$) and the orientation of the cameras relative to the laser sheet, one can compute the three-dimensional velocity profile.



(a) (b) (c)

Figure 2.12: Orientation of a stereo-PIV system in the (a) forward-backward (b) forward-forward and (c) backward-backward scattering configuration. In the forward-backward setup, the cameras are located on the same side of the flow of interest, while the cameras observe the flow of interest on opposite sides of flow in the backward-backward and forward-forward setup. (Figures used with permission of LaVision, Inc.)

It is noted that the application of the PIV technique in dusty plasmas is somewhat novel, when compared to the typical applications of this technique in the fluid dynamics community. Unlike the typical use of these techniques in fluid mechanics [79], where the tracer particles seed the flow of interest, in dusty plasmas, the tracer particles are the suspended microparticles (*i.e.* the system of interest). Despite this difference, the PIV technique has been successfully applied to many types of systems in dusty plasmas [78, 88-91].

2.2.2 Two-Dimensional Particle Image Velocimetry

In this section, the "classical", two-dimensional PIV technique is described. It is noted that there are numerous algorithms and recording techniques that have been developed for computing the velocity vectors [79]. In what follows, the discussion is restricted to the techniques that are used in processing the data that is acquired on the 3DPX device.

The images from which velocities are computed are recorded using the double-frame/single exposure PIV recording technique. Using this approach, a laser fires at times t_0 and $t_1 = t_0 + \Delta t_{laser}$. The light that scatters off of the tracer particles from the first laser pulse, t_0, is recorded in the first frame, while the light from the second laser pulse, t_1, is stored in the second frame. The image is broken into evaluation windows of size $n \times n$ pixels and a cross-correlation between identical evaluation regions in these two frames is performed to compute the average displacement of all the tracer particles located in the evaluation region. This is depicted in Figure 2. 13.

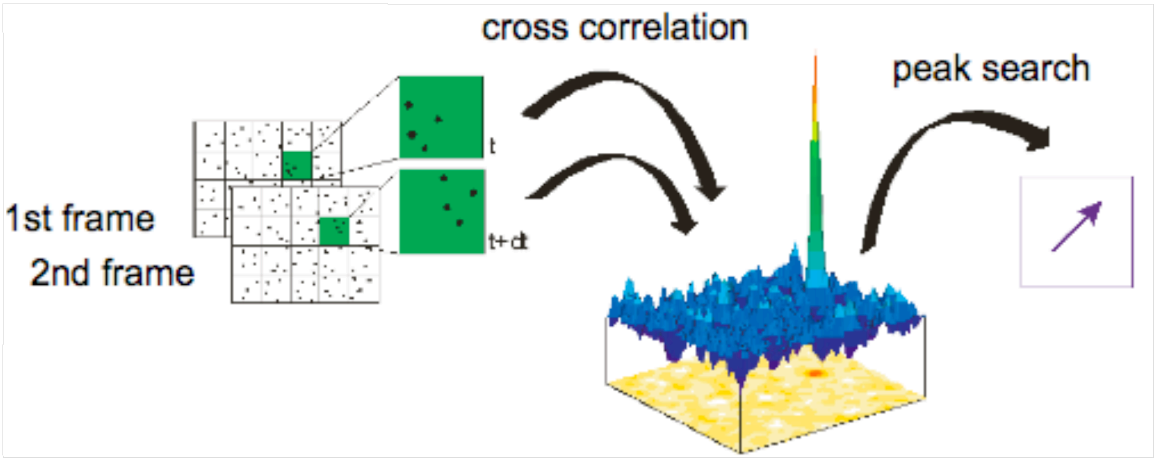


Figure 2.13: A cartoon depicting the PIV analysis process (Figure courtesy of LaVision, Inc.)

The cross-correlation is done using the “standard” two-dimensional Fourier cross-correlation technique, which is the most commonly used algorithm in the fluid dynamics community. A flow diagram for this process is seen in Figure 2.14.

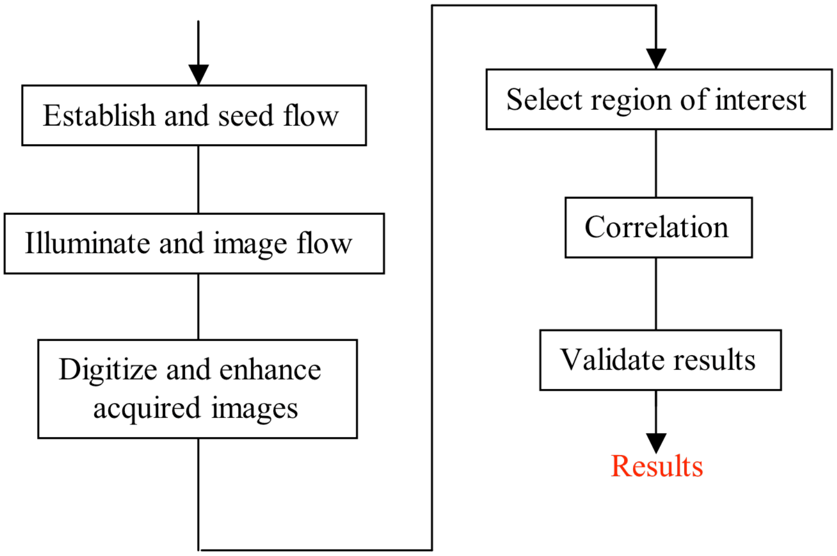

Establish and seed flow
Illuminate and image flow
Digitize and enhance acquired images
Select region of interest
Correlation
Validate results
Results

Figure 2.14: Flow diagram for the PIV analysis process.

To visualize the seeded flow, a laser beam is shaped into a laser sheet using a combination of spherical (L_2) and cylindrical (L_1) lenses to illuminates a slice of this fluid flow, as depicted in Figure 2.15.

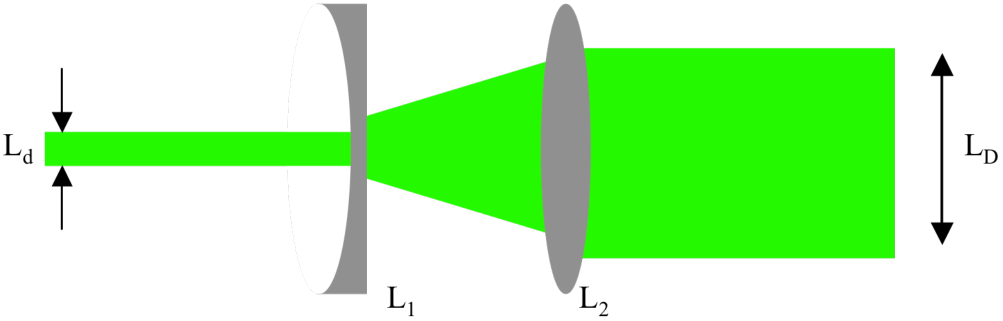


Figure 2.15: A lens system consisting of a spherical lens (L_2) with focal length f_2 and a cylindrical lens (L_1) with focal length f_1 is used to generate a laser sheet of height L_D, for illumination of the tracer particles. Incident on L_1 is a laser beam have a beam diameter L_d.

The height and width of the laser sheet can be computed using geometric optics. For the system shown in Figure 2.15, these quantities are given by Equation 2.1

$$
\begin{aligned}
L_D &= \frac{f_2}{f_1} L_d \\
w &= 2\frac{2.44\lambda f_2}{d}
\end{aligned}
\tag{2.1}
$$

where L_D is the height of the laser sheet, L_d is the diameter of the beam, f_i are the focal lengths associated with the lenses as numbered in Figure 2.15 and w is the waist (thickness) of the beam.

In a typical PIV measurement [79], the nomenclature used has the tracer particles located in the object plane and the CCD array is the image plane. At time t_0, the particles at a location $\vec{X}_i$ are illuminated by the laser and imaged on the CCD array at a location $\vec{x}_i$. At a later time, t_1, the particles have moved to a location $\vec{X}_i'$ and are imaged on the CCD array at a location $\vec{x}_i'$ x_i'. As a result, the real displacement, $\vec{D}$, appears as displacement $\vec{d}$ on the CCD array. This is depicted in Figure 2.16.

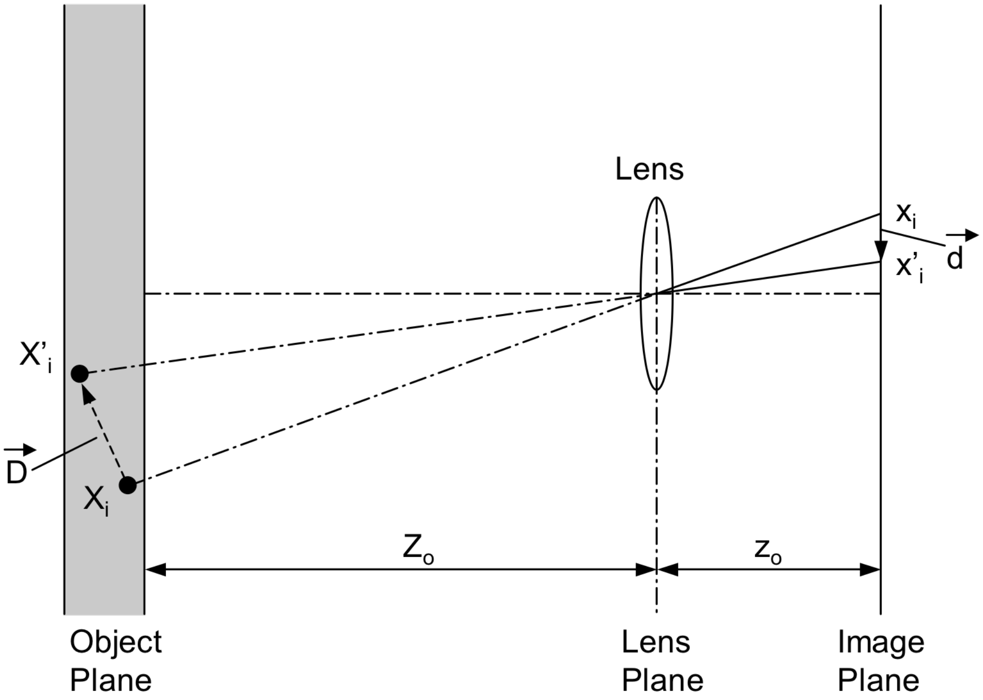


Figure 2.16: Diagram showing the imaging of a particle in the laser sheet on the CCD camera.

Assuming that there is no distortion in the imaging optics and that there is a uniform magnification across the field of the field of view, the displacement in the recorded image, $\vec{d} = \vec{x}_i' - \vec{x}_i$, is related to the real displacement, $\vec{D} = \vec{X}_i' - \vec{X}_i$, through Equation 2.2.

$$\begin{aligned} d_x &= x_i' - x_i = -M\left(D_x + D_z \frac{x_i'}{z_o}\right) \\ d_y &= y_i' - y_i = -M\left(D_y + D_z \frac{y_i'}{z_o}\right) \end{aligned} \tag{2.2}$$

A key assumption in the application of 2D PIV is the displacement is predominately in the plane of the laser sheet (*i.e.* the *x*-*y* plane), which is to say that $D_z \approx 0$. In this limit, the observed motion is described by Equation 2.3.

$$\begin{pmatrix} d_x \\ d_y \end{pmatrix} = M \begin{pmatrix} D_x \\ D_y \end{pmatrix} \tag{2.3}$$

The mathematical details of the imaging process can be found in Appendix A.1.1.1.

To calculate the motion of the particles, a cross correlation analysis [92] is performed between a subset of an image taken at time t_0, which is described by an intensity function $I_0(x, \Gamma)$, and a subset of an image taken at time t_1, which is described an by intensity function $I_1(x, \Gamma)$. In practice, one does not analytically attempt to estimate the displacement function, $\vec{d}$. Instead, one statistically finds the best match between the two images, t_0 and t_1, using the discrete cross correlation function given by Equation 2.4.

$$R(k,l) = \sum_{k=-K}^{K} \sum_{l=-L}^{L} I(x,y) I'(x+k, y+l) \tag{2.4}$$

where I and I' are subsets of images $I_0(x, \Gamma)$ and $I_1(x, \Gamma)$, with I' being larger than I, centered at location (x, y) of the CCD array. Physically, the image element I is shifted by an amount (k, l) over the image element I'. This shifting operation is done over a range of values ($-M \leq k \leq M$, $-N \leq l \leq N$). At each location, the quantity in Equation 2.4 is computed, forming a correlation plane of size $2M+1$ by $2N+1$. The mathematical details of this are given in Appendix A.1.1.2.

There are two points that bear mentioning on the application of the cross correlation technique. First, and as noted previously, it is only possible to recover linear displacements using this technique. To account for this in practice, one must use a sufficiently small Δt_{laser} and an interrogation region that is small enough that there are no second order displacements. Second, this is a computationally intensive process, requiring $O[N^2]$ computations to form a correlation plane of spatial dimension $N \times N$.

However, there is a more computationally efficient means of computing the correlation function in Equation 2.4. This technique makes use of the correlation theorem, which is derived in Appendix A.1.2. The correlation theorem states that cross correlation of two functions, I and I' is equivalent to a complex conjugate multiplication of their Fourier transforms, or

$$R(x,y) \Leftrightarrow \hat{I} \cdot \hat{I}'^{*} \qquad (2.5)$$

where $\hat{I}$ and $\hat{I}'^{*}$ are the Fourier transform of I and I', respectively. The Fourier Transforms provide a computationally efficient approach to computing the cross correlation function. The flow diagram of the PIV analysis is seen in Figure 2.17.

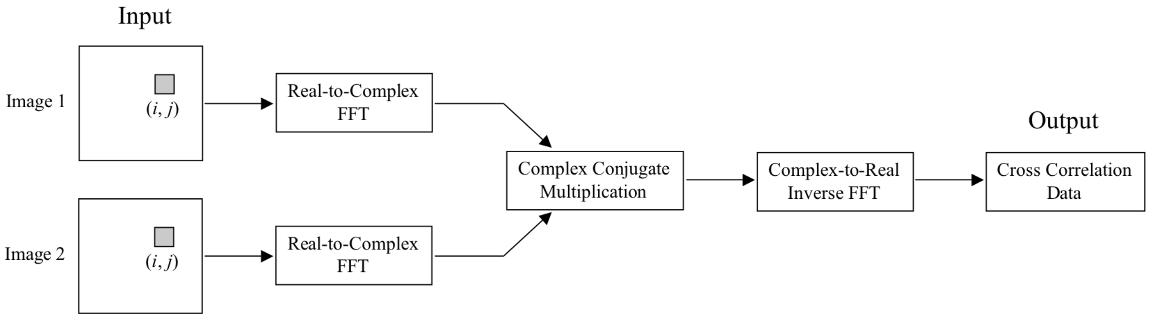


Figure 2.17: Implementation of the cross correlation technique using fast Fourier Transforms.

Once the cross correlation plane has been computed, the largest peak in the correlation plane is fitted using a Gaussian curve. In the *DaVis* software, three adjoining points from the correlation plane are fit using a Gaussian peak fit in both the x and y direction. To do this, the correlation plane is scanned for the maximum value, $R_{(i,j)}$. The adjoining four correlation points in the correlation plane, $R_{(i-1,j)}$, $R_{(i+1,j)}$, $R_{(i,j-1)}$ and $R_{(i,j+1)}$, are extracted. A Gaussian curve, defined in Equation 2.6, is then fit in both the x- and y-directions. From these fits, the location of the correlation peak can be identified to within 0.05 pixels, depending on the quality of the images used.

$$f(r) = C\exp\left\{-\frac{(r_o - r)^2}{k}\right\} \tag{2.6}$$

where r denotes a coordinate position (here, it can denote either the x- or the y-position), C and k are constants, and the r_o denotes an offset in either the x- or y-direction defined in Equation 2.7.

$$\begin{aligned} x_o &= i + \frac{\ln R_{(i-1,j)} - \ln R_{(i+1,j)}}{2\ln R_{(i-1,j)} - 4\ln R_{(i,j)} + 2\ln R_{(i+1,j)}} \\ y_o &= j + \frac{\ln R_{(i,j-1)} - \ln R_{(i,j+1)}}{2\ln R_{(i,j-1)} - 4\ln R_{(i,j)} + 2\ln R_{(i,j+1)}} \end{aligned} \tag{2. 7}$$

where i and j denote discrete locations in the correlation plane and $R_{(i,j)}$ denotes the value of the correlation value at location (i, j). This process is depicted in Figure 2.18.

Although the PIV algorithm described above yields generally good results, there are additional refinements that can be used to improve the quality of the results generated in the PIV analysis. The first technique involves overlapping evaluation regions. In this work, a 50% overlap is used. This more than doubles the number of independent vectors that are computed and more accurately maps the velocity field. It is noted that a 50% overlap is the maximum overlap that can be used and still obtain independent PIV vectors. The second technique is multi-pass correlation. In this process, the correlation analysis is repeated *m* times. In the first pass, a displacement map is generated. In the next pass, when the correlation analysis is repeated, the subregion extracted from the second image used in the correlation analysis is offset in the direction of displacements that were computed in the first correlation analysis. This process is then repeated until the *m* iterations have been completed. For each iteration, the displacement field that was generated in the previous correlation analysis is used to generate the image shift for the sub region in the second image.

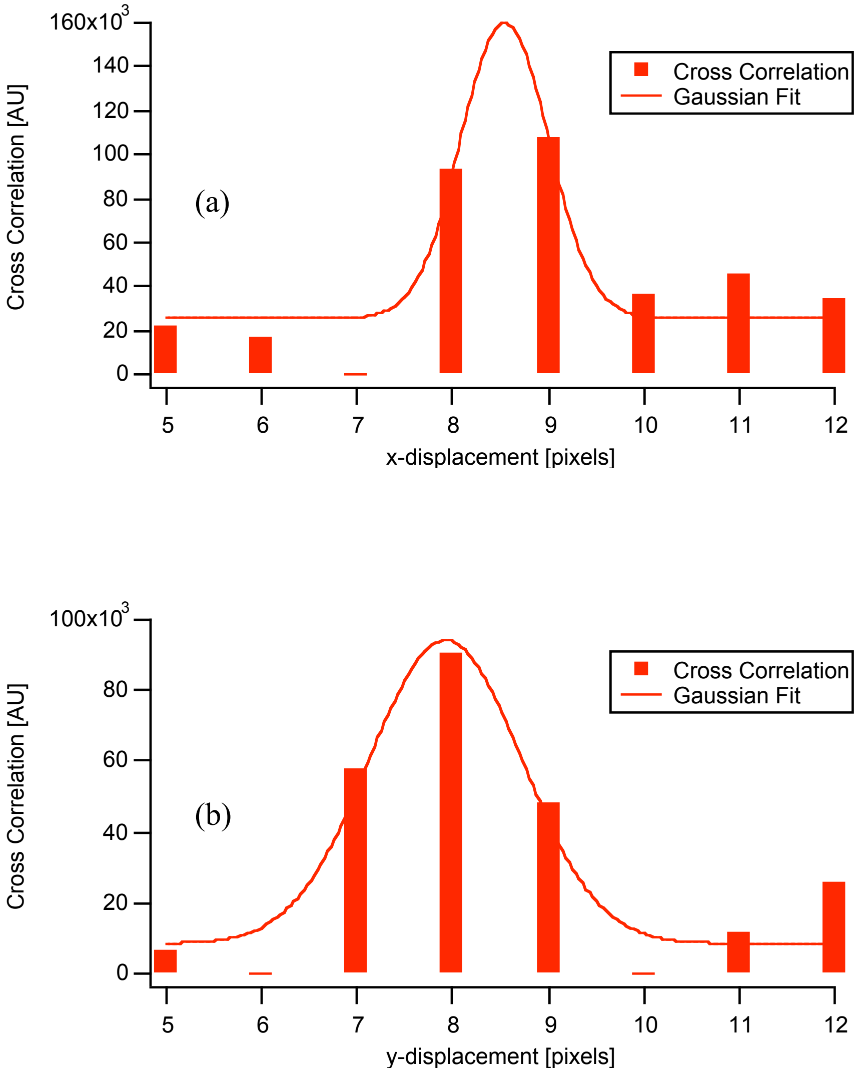


Figure 2.18: Depicting how sub-pixel resolution is achieved in the PIV analysis. The bars denote the computed values of the cross-correlation function, while the curve depicts a best fit Gaussian to the computed values in the (a) x-direction and (b) y-direction. The peak from the curve fit yields the desired displacement with far greater accuracy than can be found from the raw cross correlation data. The actual displacement in the images used in this analysis was 8 pixels in the x-direction and 8.5 pixels in the y-direction. The displacement that is extracted from the fit was found to be 7.92 pixels in the x-direction and 8.55 pixels in the y-direction. It is noted that the cross correlation analysis performed in the analysis generating this figure was far simpler than what is done in the *DaVis* software.

The multi-pass correlation analysis yields a more accurate result for four reasons. First, this approach minimizes the error in the measured displacement by keeping the correlation peak near the origin of the correlation plane. Second, the image shift that is used is locally determined. As a result, a shift that more closely resembles the flow is used than what would otherwise be obtained using a global offset. Third, this approach allows one to use a smaller evaluation region, which will generate a higher spatial resolution. Finally, this approach results in a more accurate velocity field, as fewer particles are likely to leave the evaluation region [93]. In the work in this dissertation, a two pass correlation analysis is performed. On the first pass, a 16 × 16 pixel evaluation region is used and a 12 × 12 pixel evaluation region is used in the second pass.

2.2.3 STEREOSCOPIC PARTICLE IMAGE VELOCIMETRY

While the PIV approach is a powerful measurement technique, it does have limitations. Foremost among these limitations is that the "classical" PIV measurement can only measure the projection of the velocity vector in the plane of the laser sheet. This error can be seen in Figure 2.19 and can lead to a potentially significant source of error: any motion that is transverse to the plane of the laser sheet is lost. Further, this lack of information regarding this third component of velocity can introduce an unrecoverable error in the in-plane motion. To quantify this, consider the motion that is shown in Figure 2.19.

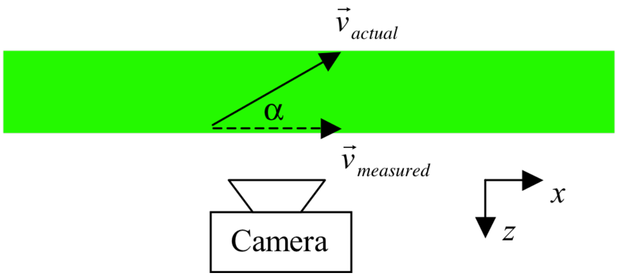


Figure 2.19: Showing the motion observed by the PIV system, $\vec{v}_{measured}$, as opposed to the actual motion, $\vec{v}_{actual}$, of the particle. The true motion of the particle is depicted by the solid arrow, while the dashed arrow depicts the measured motion.

In Figure 2.19, the laser is directed along the x-axis, while the PIV camera is oriented along the direction of the *z*-axis, allowing for visualization of motion in the *x*-*y* plane. Suppose that the particle is moving purely in the *x*-*z* plane with a speed v_{actual} at an angle α with respect to the *x*-axis. The velocity that is measured by the PIV system will be $\vec{v}_{measured} = \left|\vec{v}_{actual}\right| \cos\alpha \;\; \hat{i}$. The error is negligible for motion that is essentially restricted to the *x*-axis, *i.e.* along the direction of the illumination sheet of light. As α increases, the error becomes progressively larger as indicated in Figure 2.20. It is clear that there can be a strong need for knowledge of this third component of the motion.

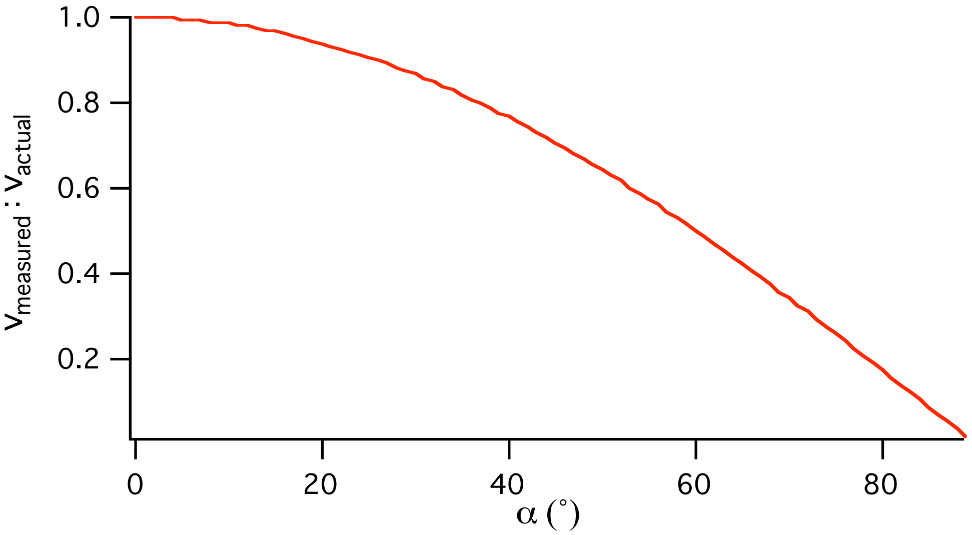
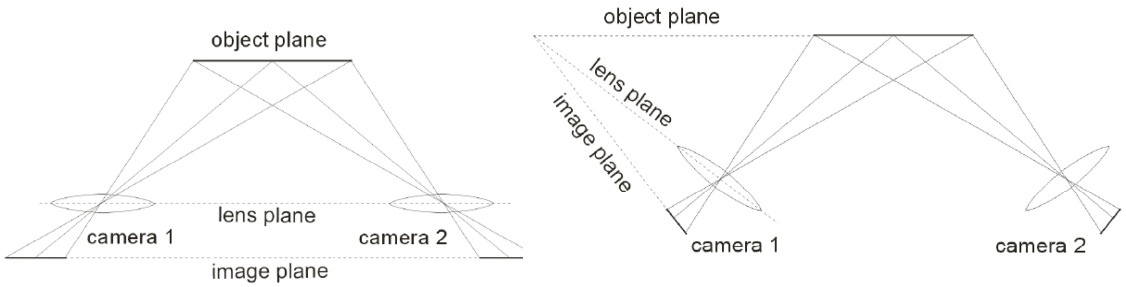


Figure 2.20: Plot of the ratio of the velocity measured by the PIV system, $v_{measured}$, to the true velocity, v_{actual}, as a function of the angle α. For $\alpha > 25°$, the error exceeds ten percent.

There are numerous methods that can be implemented to extract information about the third velocity component [94]. The most straightforward of these methods involves the addition of a second 2D PIV measurement from a different viewing axis [87]. This approach is generally referred to as stereoscopic particle imaging velocimetry. In practice, there are two typical orientations for the two camera stereo-PIV system, referred to as the lens translational and the angular displacement method. The orientation of the cameras in each of these approaches is shown in Figure 2.21. For the work that is presented in this dissertation, the angular displacement method is employed.



(a) (b)

Figure 2.21: Depicting the orientation of the cameras in the two standard stereoscopic particle image velocimetry orientation: (a) translational method and (b) angular displacement method. (Figures used with permission of LaVision, Inc.)

Figure 2.22 provides a qualitative illustration of the "out of plane" velocity reconstruction using two cameras. Because of the different orientations, each camera will measure a different particle displacement. However, if the orientation of each camera, relative to the light sheet, is known, it is possible to reconstruct the out-of-plane velocity component of the motion. This process is quantitatively described in Section 2.2.3.2.

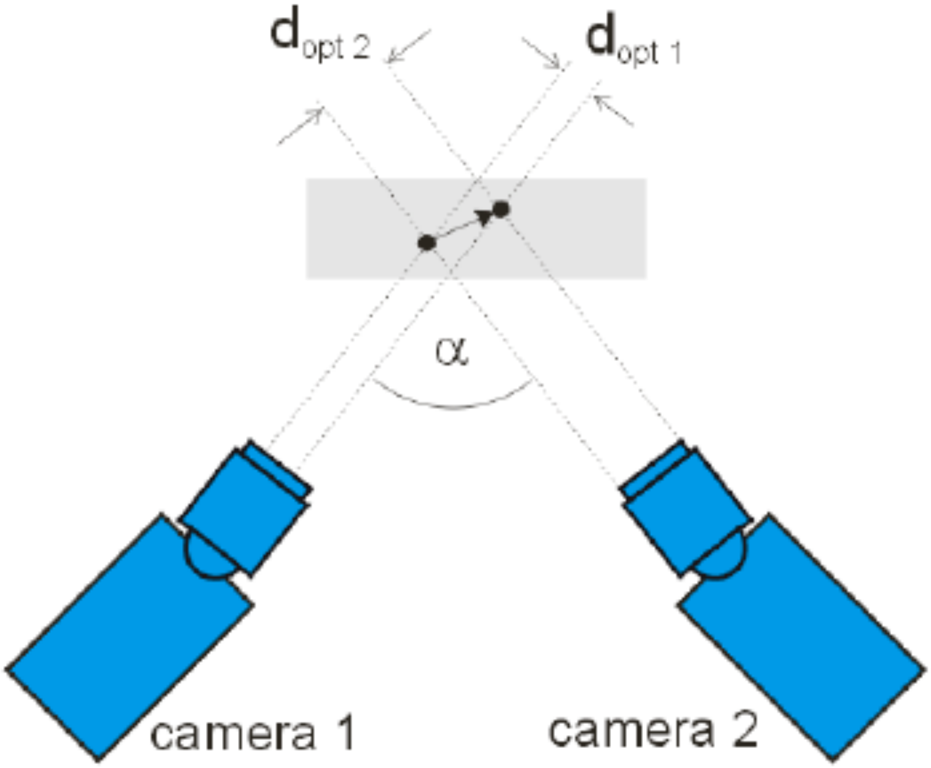


Figure 2.22: Depicting how the out-of-plane component is extracted using the stereoscopic set-up. A difference in the measured displacements is observed if there is motion in the out-of-plane direction and is related to the displacement in the out-of-plane motion. (Figure used with permission of LaVision, Inc.)

For a stereoscopic system using the angular displacement method, the measurement precision of the out-of-plane component increases as the angle between the cameras increases. However, this oblique viewing results in a limited depth of field. As a result, it is difficult to obtain an image that is focused over the entire field of view, as seen in Figure 2.23.

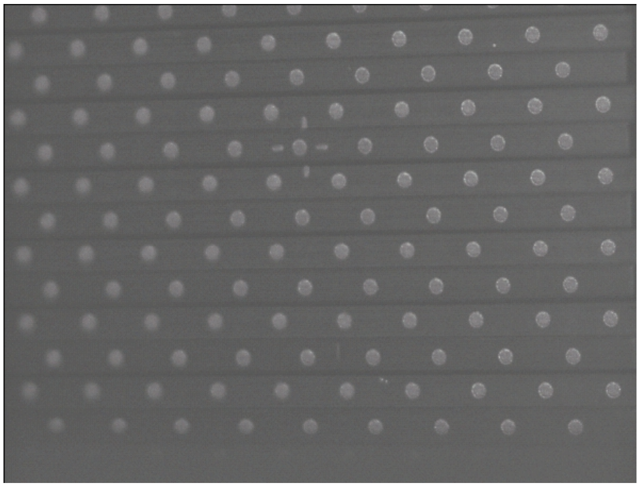


Figure 2.23: Depicting the limited depth of field that arises due to the angular displacement method of stereoscopy. It is noted that only the dots on the right hand side of the image are in focus.

This limited depth of field can only be accommodated for by tilting the image plane such that the image, lens and object plane intersect at a common point. This is known as the Scheimpflug criteria (or the "Keystone Correction") [95] and is illustrated in Figure 2.24.

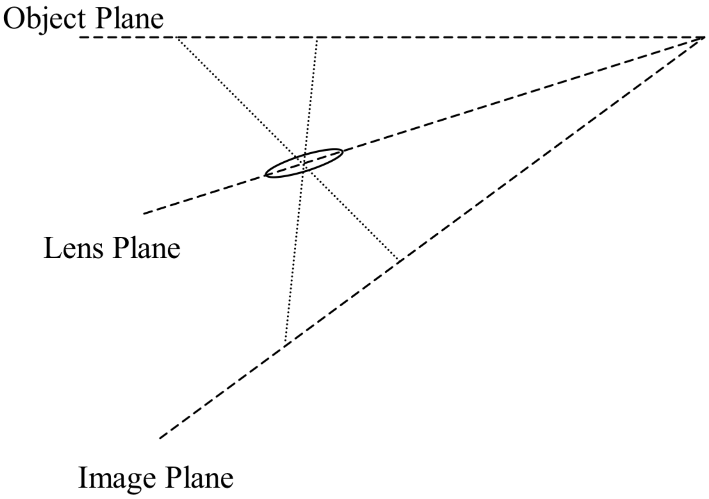


Figure 2.24: Depicting the Scheimpflug criteria, where the image, object and lens planes all intersect at a single point.

To enforce the Scheimpflug criteria, the lens for the CCD camera is mounted on a Scheimpflug adapter. This allows for adjustment of the lens plane until the three planes are properly aligned. When the three planes are properly aligned, the entire field of view will be in focus. Figure 2.25 shows how the Scheimpflug criteria is enforced.

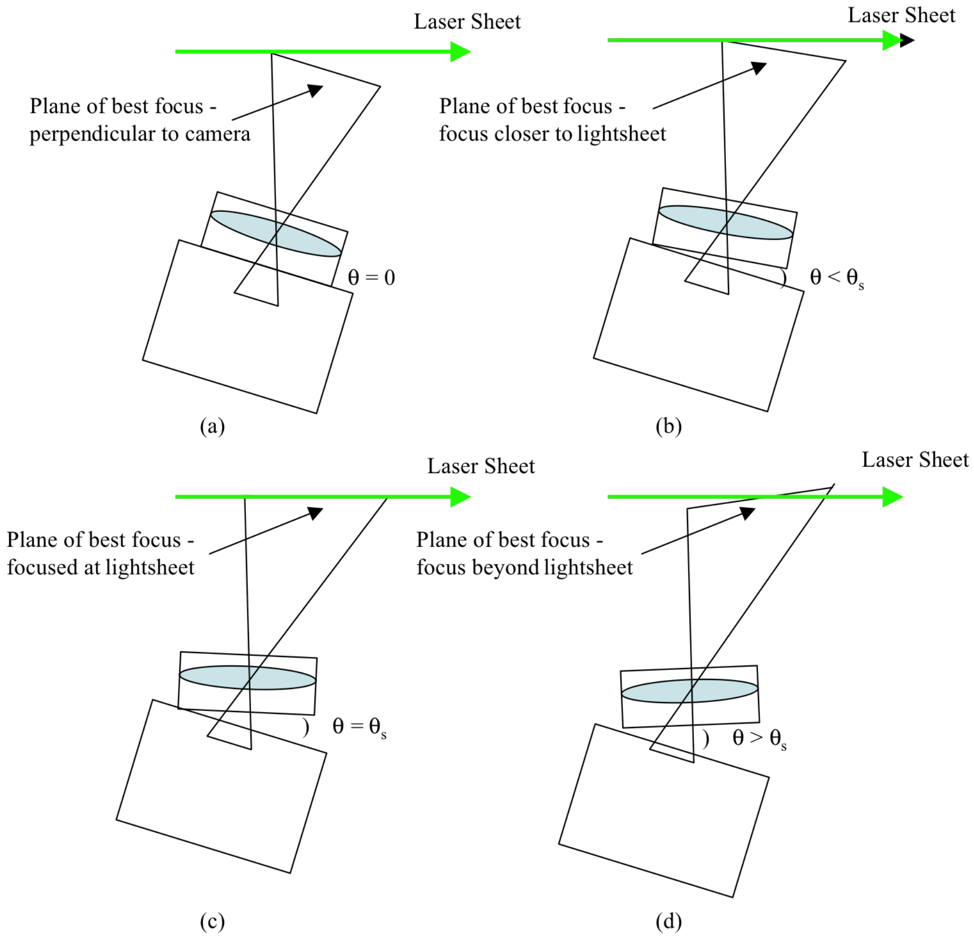


Figure 2.25: Showing the possible lens tilts that can be encountered when enforcing the Scheimpflug criteria. In (a) the lens is not tilted, (b) the lens has not been tilted enough, (c) the lens is titled to the Scheimpflug angle and (d) has the lens has tilted too far.

In practice, one must manually adjust the Scheimpflug adapter until the image appears to be in focus over the entire field of view. In Figure 2.26, one sees the image from Figure 2.23 once the Scheimpflug criteria has been enforced.

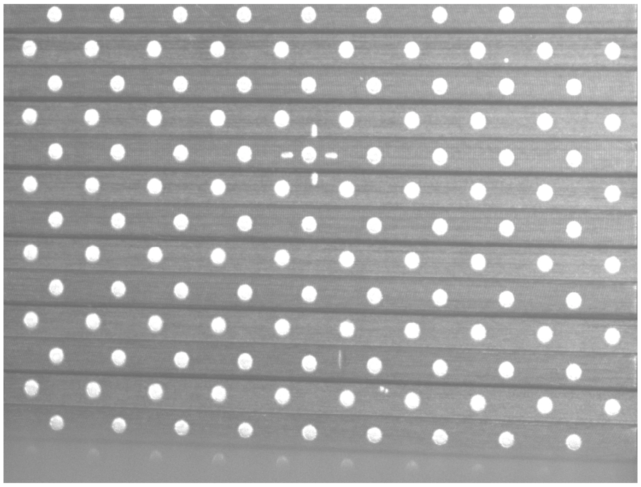


Figure 2.26: Depicting the image seen in Figure 2.23, once Scheimpflug criteria has been enforced.

Enforcement of the Scheimpflug criteria introduces a strong perspective distortion, which can be seen in Figure 2.26. As a result, a rectangular image appears as an isosceles trapezoid. This is due to two types of distortion: anamorphic and shearing distortion. The anamorphic distortion generates a foreshortening of the *x*-dimension (horizontal direction) by the cosine of the object tilt angle. The shearing distortion results in a non-uniform magnification in the *y*-dimension (vertical direction) as one moves across the *x*-dimension. To compensate for these two forms of distortion, the PIV camera must be empirically calibrated. This is done with the aid of a 3-D "target" seen in Figure 2.27. This calibration process [96], which is more completely discussed in Section 2.2.3.1, will generate a mapping function for each PIV camera which will transform, or "dewarp," the acquired images from the image plane (*i.e.* what is acquired by the cameras) to the global coordinate system (*i.e.* where the actual displacements are computed). In Figure 2.27, one sees the image from Figure 2.26 dewarped into the gloabal coordinate system for the camera orientation used to acquire the image seen in Figure 2.26. The region of the image that is black are lost in the dewarping process, while the red cross depicts the expected location of the dots on one plane of the calibration target.

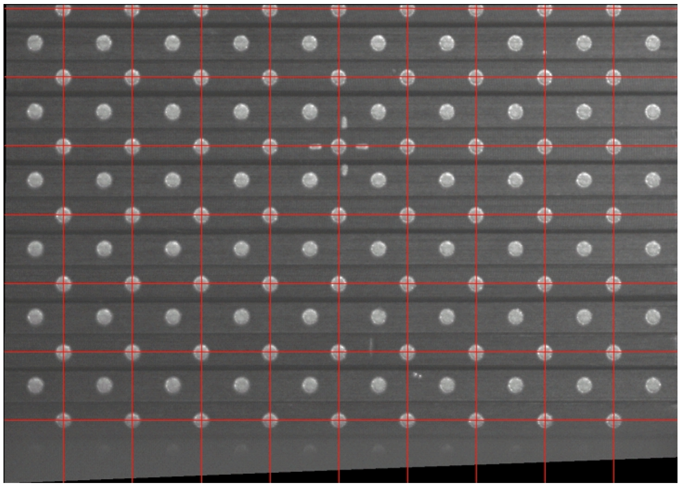


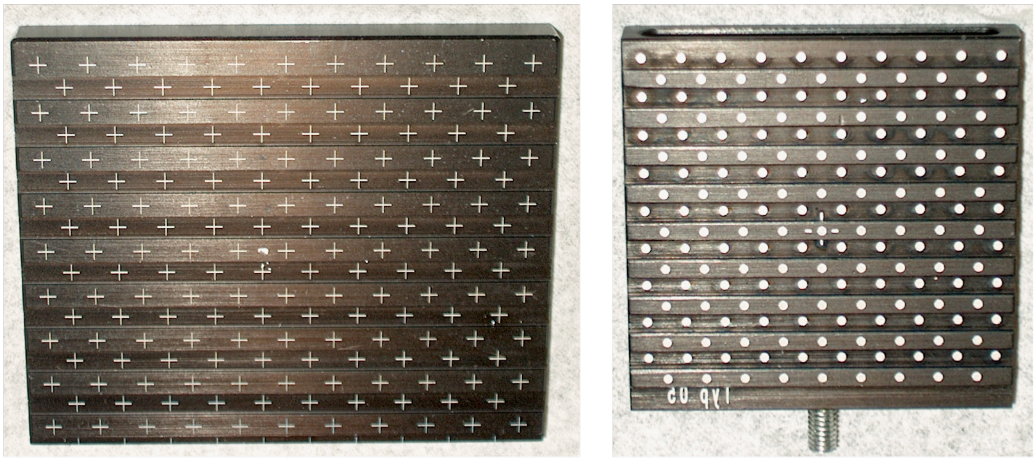
Figure 2.27: Depicting the image seen in Figure 2.24 once the image has been dewarped into the global coordinate system. The red grid depicts the anticipated location of the dots on one plane of the calibration target. It is noted that the distortion observed in Figure 2.26 is gone and the magnification is uniform across the field of view.

Once the acquired images have been dewarped into the global coordinate system using the mapping functions that were generated in the empirical calibration process, two sets of 2D velocity vectors, one for each camera, are generated. It is noted that the requirement of uniform image magnification that is assumed in the derivation of the imaging process used in the 2D PIV analysis has been satisfied once the images have been dewarped into the global coordinate system. Additionally, from the calibration process, one gains knowledge of the orientation of the cameras relative to each other and the laser sheet. Using this geometry, one can extract the three-dimensional velocity field. It is noted that the system is over specified, *i.e.* the three velocity vector components are computed from the two known two dimensional velocity vectors. As a result, the velocities that are computed using the stereo-PIV technique tend to be more accurate that those found using the classical 2D PIV approach.

Section 2.2.3.1 describes the calibration techniques, which are used on the 3DPX experiment to compute the image dewarping parameters and to determine the orientation of the two cameras relative to each other. Section 2.2.3.2 describes how to extract the three-dimensional velocity information.

2.2.3.1 IMAGE DEWARPING AND CALIBRATION

In order to compute the three-dimensional velocities acquired using this stereoscopic setup, the two-dimensional vectors from each PIV setup must be computed in the same coordinate system, *i.e.* the global coordinate system. To determine the mapping function from the image plane to the global coordinate system, an empirical calibration process is performed using Tsai's pinhole model [97]. In this approach, each camera images a three-dimensional, two-level calibration plate containing a defined grid of marks. Two calibration plates that are used on the 3DPX device are seen in Figure 2.28. Using these images, a mapping function for each PIV camera is generated using Tsai's pinhole. The pinhole model is based on the theorem of intersecting lines and maps a particle, which exist in the "global" 3-D coordinate system (X_W, Y_W, Z_W), to a location on the CCD array, the camera coordinate system (x, y). The mathematical details of this model can be found in Appendix A.1.3.



(a) (b)

Figure 2.28: Showing two 3D calibration targets used on the 3DPX device. The large target, shown in (a), consist of an array of Maltese crosses, while the smaller target, shown in (b), consist of an array of dots.

One critical aspect in the calibration process is the need for a perfect alignment of the laser sheet and the calibration target [96]. Any misalignment will result in an error in the mapping function. As a result, the images will not be dewarped into the same world coordinate system, leading to an error in the computed velocity field. This is seen in Figure 2.29. Here, the laser is not exactly aligned with the global coordinate system. As a result, each camera will detect that the particle appears in slightly different positions in the global coordinate systems. The particle will appear shifted to the left by camera 1 and to the right by camera 2. The difference in these positions defines a disparity vector and can be a major source of error in the resulting velocity field.

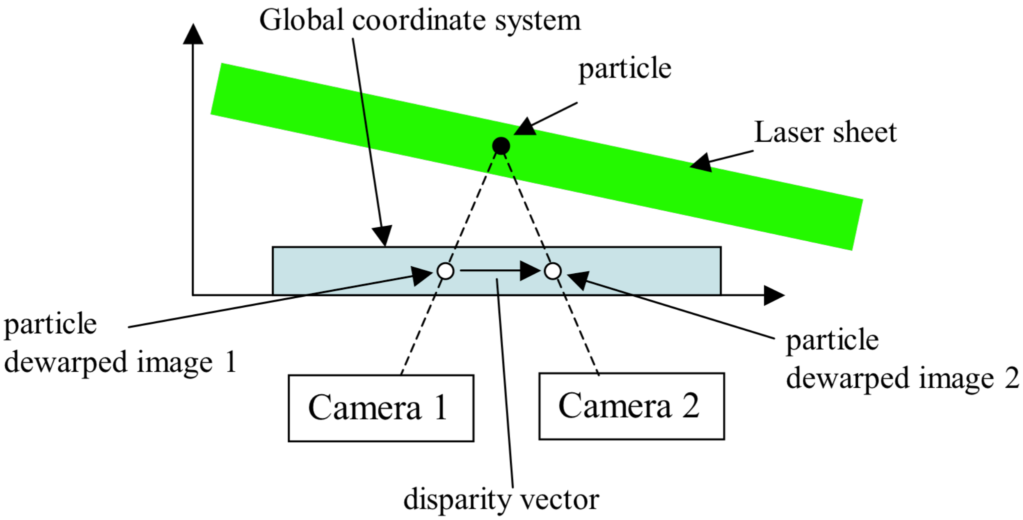


Figure 2.29: Showing the error that is introduced when the laser sheet and calibration plate are not properly aligned.

It is possible, however to tweak the calibration to account for possible misalignments between the laser sheet and calibration plate that existed during the initial calibration. This process, known as self-calibration [96], simplifies the calibration process and results in significantly smaller error. A flow diagram for the self-calibration process is seen in Figure 2.30 and described below.

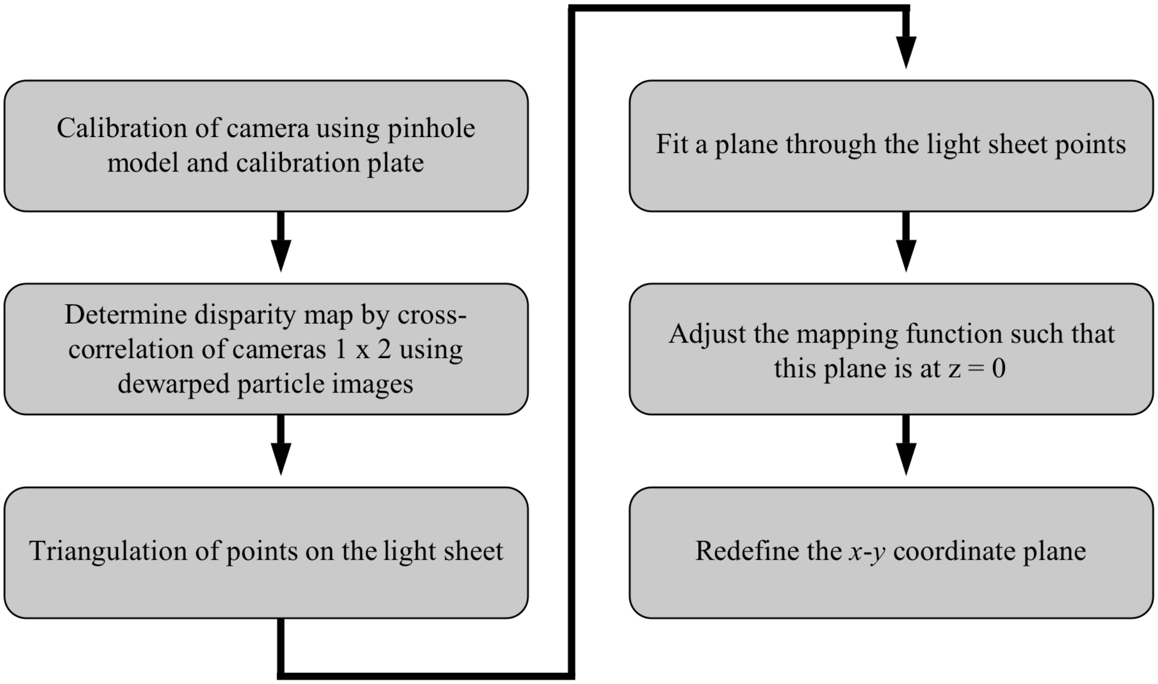

Calibration of camera using pinhole model and calibration plate
Determine disparity map by cross-correlation of cameras 1 x 2 using dewarped particle images
Triangulation of points on the light sheet
Fit a plane through the light sheet points
Adjust the mapping function such that this plane is at z = 0
Redefine the x-y coordinate plane

Figure 2.30: Flow diagram of the self-calibration process.

After the system has been calibrated using a calibration plate and the pinhole model, a sequence of ~50 images of a particle pattern is acquired. By cross correlating the acquired images of the particle pattern, it is possible to determine if there is deviation between the two camera's field of views. If the dewarped images are exactly the same, then there will be no displacement field generated. If there is a deviation, the resulting displacement field, known as a disparity vector map, can be used to correct the camera calibration. Using the disparity map, one can determine the corresponding world points in the measurement plane using a standard triangulation method. These world points are then fitted to a three dimensional plane, which is used to correct the mapping function of each camera. It is noted that the triangulation method requires a volume mapping function, which is why a pinhole model is used in the calibration process. This process can be repeated to improve the results. However, the fit tends to converge after a few (~3) passes.

2.2.3.2 RECONSTRUCTING THREE-DIMENSIONAL VELOCITY INFORMATION

Using geometric optics, the particle image displacement for the system seen in Figure 2.16 was found in Section 2.2.2 to be

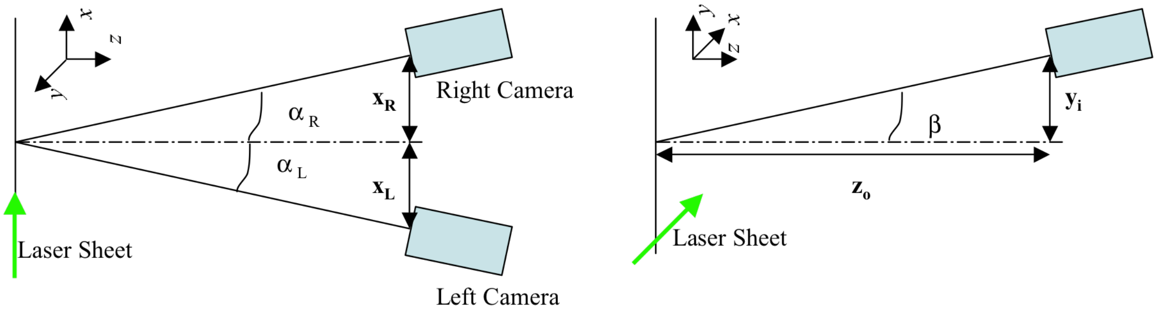
$$\begin{aligned} d_x &= x_i' - x_i = -M(D_x + D_z \frac{x_i'}{z_o}) \\ d_y &= y_i' - y_i = -M(D_y + D_z \frac{y_i'}{z_o}) \end{aligned} \tag{2.8}$$

In the stereoscopic system, each camera will generate a two-dimensional velocity field, $\vec{V} = (U_{W,i}, V_{W,i})$, given by Equation 2.9.

$$\begin{aligned} U_{W,i} &= -\frac{x_i' - x_i}{M\Delta t} \\ V_{W,i} &= -\frac{y_i' - y_i}{M\Delta t} \end{aligned} \tag{2.9}$$

where *i* is an index denoting the camera (right or left). It is noted that the displacements and velocities given by Equations 2.8 and 2.9 respectively have been computed in the global coordinate system. Consequently, it is assumed that the acquired images have been dewarped as described in Section 2.2.3.1 prior to the PIV calculation being performed. In practice, this is the case.

To extract the three-dimensional velocities, the following geometry is assumed for the two PIV cameras: α_i denotes the angle in the x-z plane between the z-axis and the ray that connects the tracer particle and the recording plane through the lens center and β_i denote the corresponding angle in the y-z plane. This is illustrated in Figure 2.31.



(a) (b)

Figure 2.31: Depicting the geometry of the stereo-PIV system as viewed from (a) above and (b) the side. It is noted that the angles α and β do not have to be the same for both camera. In this example, the laser sheet propagates defines the x-y plane.

For the geometry shown in Figure 2.31, the angles α_i and β_i are defined as

$$\begin{aligned} \tan\alpha_i &= \frac{x_i'}{z_o} \\ \tan\beta_i &= \frac{y_i'}{z_o} \end{aligned} \qquad (2.10)$$

where *i* is again an index denoting the camera (right or left).. Using these angles and the velocities that are computed from each PIV camera, the velocities in the global coordinate system, $\vec{V} = \left(V_x, V_y, V_z\right)$, are given by Equation 2.11.

$$\begin{aligned} V_x &= \frac{U_{W,r}\tan\alpha_l + U_{W,l}\tan\alpha_r}{\tan\alpha_r + \tan\alpha_l} \\ V_y &= \frac{V_{W,r}\tan\beta_l + V_{W,l}\tan\beta_r}{\tan\beta_r + \tan\beta_l} = \frac{V_{W,r} + V_{W,l}}{2} + \frac{V_z}{2}\left(\tan\beta_l - \tan\beta_r\right) \\ V_z &= \frac{U_{W,r} - U_{W,l}}{\tan\alpha_r + \tan\alpha_l} \end{aligned} \qquad (2.11)$$

2.2.4 STEREOSCOPIC PARTICLE IMAGE VELOCIMETRY SYSTEM AT AUBURN UNIVERSITY

This section describes the hardware of stereo-PIV system that was used in the studies presented in this dissertation. Section 2.2.4.1 describes the hardware and software components. Section 2.2.4.2 describes initial measurements that were made to verify the functionality of this system. Finally, Section 2.2.4.3 describes how the uncertainty in the system is measured.

2.2.4.1 DESCRIPTION OF THE SYSTEM

The stereo-PIV system used in the Plasma Sciences Laboratory at Auburn University was purchased from LaVision Inc. in 2003. It consists of a pair of frequency doubled neodymium-doped yttrium aluminium garnet, Nd:YAG, lasers (λ = 532 nm) and a pair of cross-correlation CCD cameras oriented in the backscattering configuration. The laser used in this system is a Solo PIV III, manufactured by New Wave Research. This dual cavity pulsed laser has a repetition rate of 15 Hz and an output energy of 50 mJ per pulse. The laser pulses are shaped into a laser sheet using a -10 mm focal length cylindrical lens.

The CCD cameras used in this system are *Imager Intense* cross correlation CCD cameras. This 12-bit progressive scan CCD camera has a 1376 × 1040 pixel array that acquires images at a frame rate of 10 frames per second. To improve the collection efficiency of the CCD elements of the camera, the cameras are cooled to an operating temperature of -10°C using Peltier elements. In the stereoscopic arrangement, the cameras operate at a maximum frame rate of 5 Hz. Each camera is configured with a Scheimpflug lens adapter and a 60 mm Nikon MicroNikkor lens. To reduce extraneous light signals, each camera is fitted with a narrow band pass filter, $\lambda = 532 \pm 10$ nm. Both the laser and the camera systems are mounted on a single axis linear translation stage that has 500 mm of travel, allowing for measurements of the velocity vectors anywhere in the volume of the 3DPX device. A sketch of the experimental setup is seen in Figure 2.32.

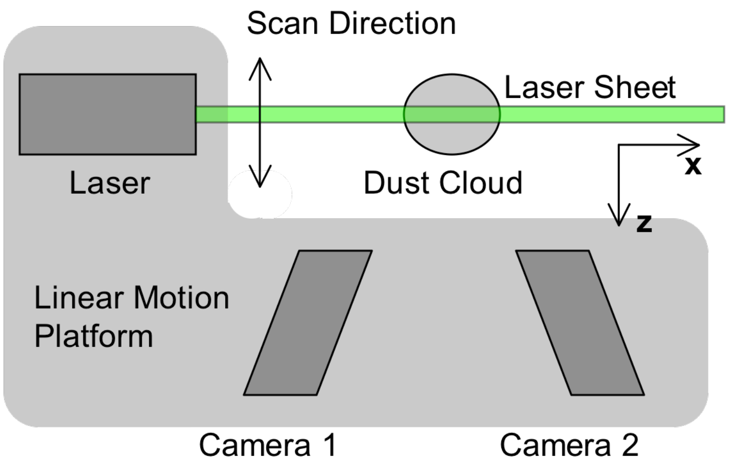


Figure 2.32: Schematic depicting the top view of the experimental setup and the orientation of the stereo-PIV hardware.

A software system from LaVision Inc., *DaVis* (*Data Visualization*), is used to control the PIV hardware (laser, cameras and translation stage) and to compute the three-dimensional velocity vectors. A flow diagram of the full PIV analysis process used by the *DaVis* software seen in Figure 2.33.

2.2.4.2 VERIFICATION OF THE SYSTEM

Shortly after acquiring the stereo-PIV system, an experiment was performed to verify the system's three-dimensional measurement capabilities [78]. In this investigation, a dusty plasma was generated and a perturbation was applied to induce an out-of-plane (*i.e., z*-direction) displacement of the dust cloud. Two independent imaging systems (the stereo-PIV system and a separate LLS system) were used to confirm the motion of the cloud.

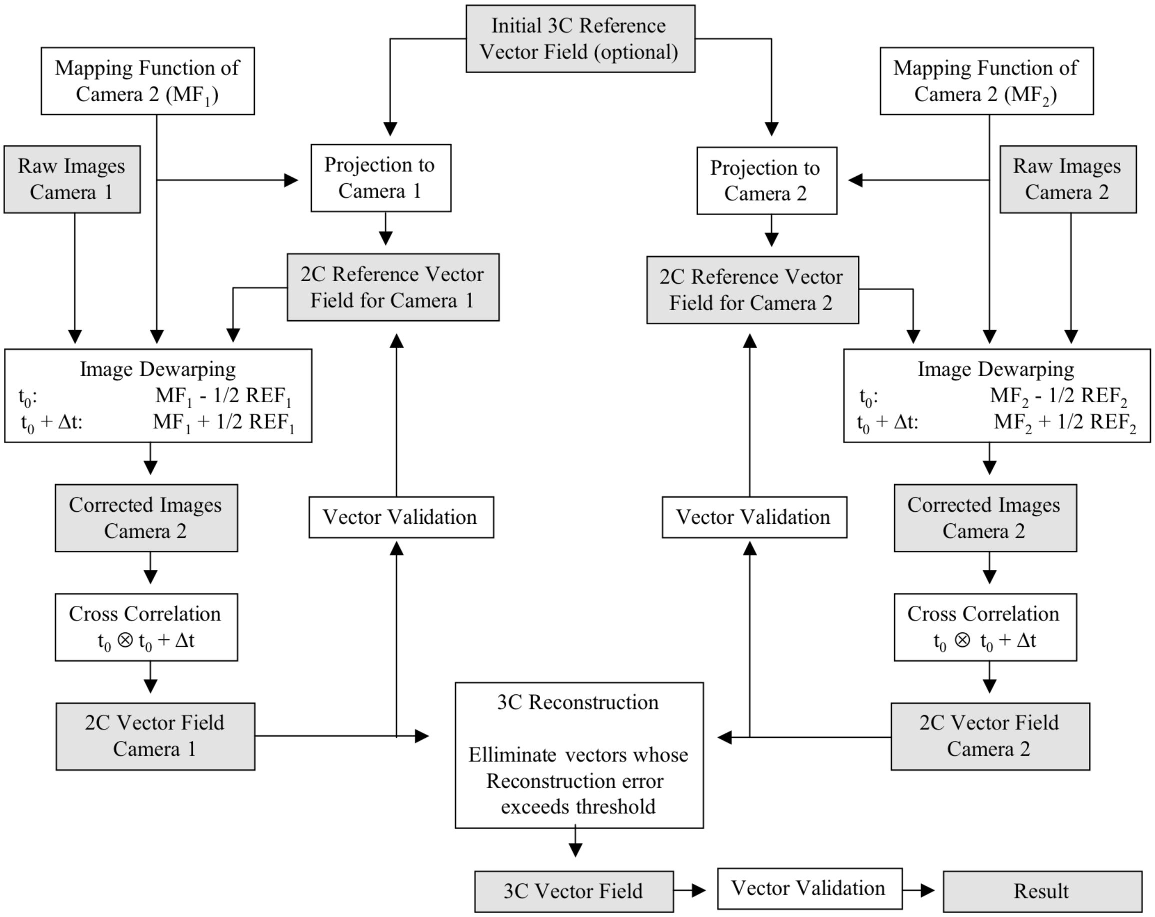

Initial 3C Reference Vector Field (optional)
Mapping Function of Camera 2 (MF1)
Mapping Function of Camera 2 (MF2)
Raw Images Camera 1
Projection to Camera 1
Projection to Camera 2
Raw Images Camera 2
2C Reference Vector Field for Camera 1
2C Reference Vector Field for Camera 2
Image Dewarping
t0: MF1 - 1/2 REF1
t0 + Δt: MF1 + 1/2 REF1
Image Dewarping
t0: MF2 - 1/2 REF2
t0 + Δt: MF2 + 1/2 REF2
Corrected Images Camera 2
Vector Validation
Vector Validation
Corrected Images Camera 2
Cross Correlation t0 ⊗ t0 + Δt
Cross Correlation t0 ⊗ t0 + Δt
2C Vector Field Camera 1
3C Reconstruction
Elliminate vectors whose Reconstruction error exceeds threshold
2C Vector Field Camera 2
3C Vector Field
Vector Validation
Result

Figure 2.33: Showing the PIV analysis process used by the DaVis software.

In this experiment, an argon dc glow discharge plasma is generated using a pair of square, stainless steel electrodes that are 2.5 cm on a side. The electrodes are biased with respect to the electrically grounded vacuum vessel walls. The upper electrode functions as the anode and is biased to V_{anode} = 200 V. The lower electrode functions as the cathode and is biased to $V_{cathode}$ = -90 V. The chamber is filled with argon gas to a pressure p = 114 mTorr. In this configuration, the dust cloud formed in the sheath of the dust tray. The microparticles used in this experiment are silica microspheres having a size distribution defined by $r = 1.5 \pm 0.5$ μm. The source of the particles for this experiment was located on a square tray that is ~ 2.5 cm below the cathode. An SEM image of the microparticles used in this experiment is shown in Figure 2.34.

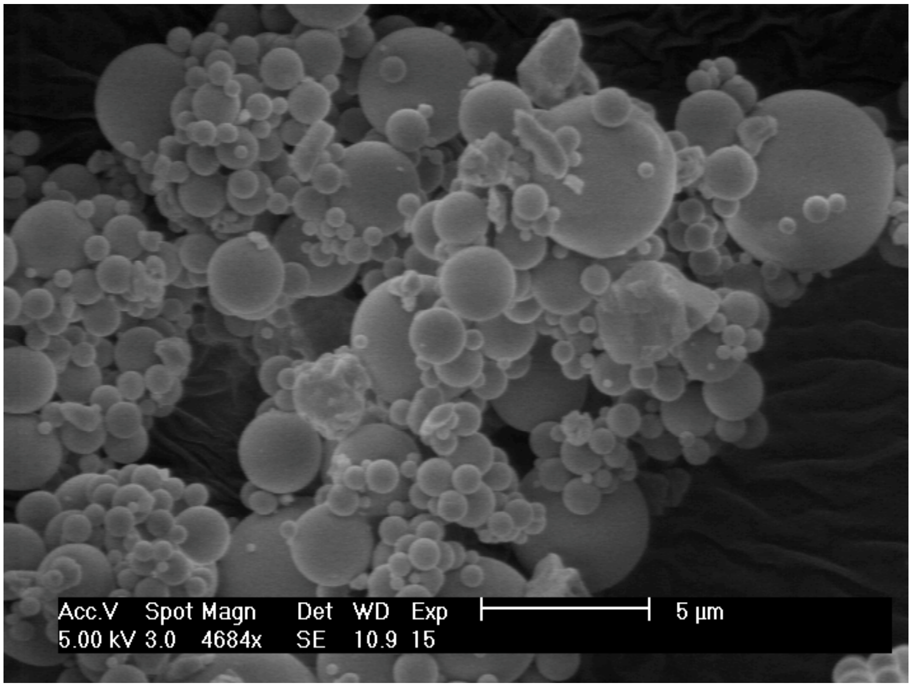


Figure 2.34: SEM image of the silica microspheres used in experiments verifying the functionality of the stereo-PIV system. {SEM image courtesy of Andrew Post-Zwicker, Princeton Plasma Physics Laboratory}

To induce the perturbation in the z-direction, a third electrode is used. This perturbation electrode is rectangular copper plate and is located below and behind the anode and cathode. The orientation of the electrodes and dust tray are shown in Figure 2.35.

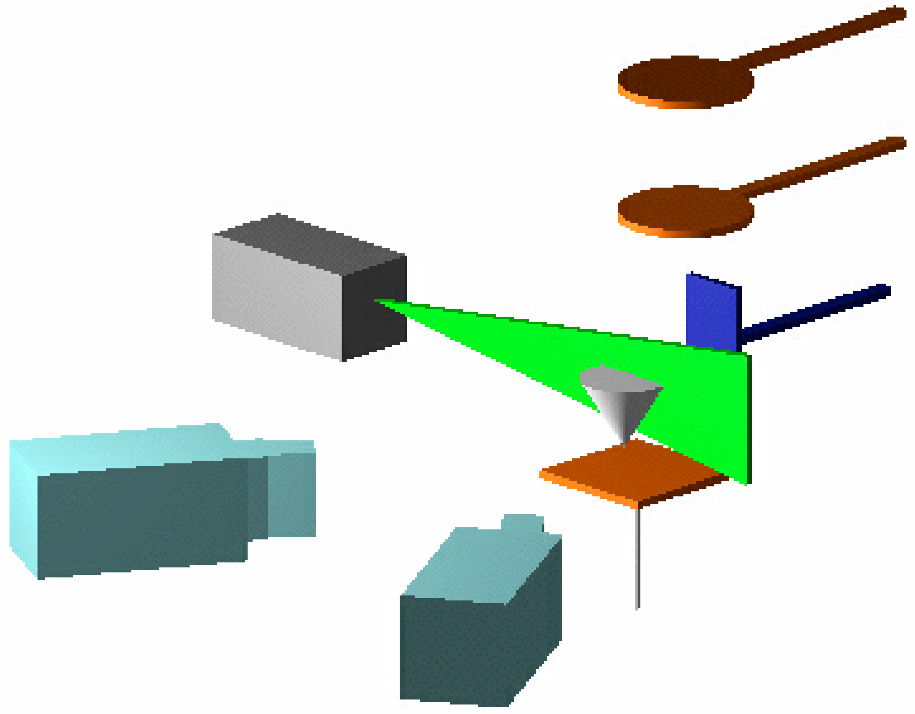


Figure 2.35: Orientation of the electrodes and dust tray used to verify the functionality of the stereo-PIV system.

To verify that a shift in the *z*-direction occurred, a second video system, oriented perpendicular to the PIV laser sheet, is used. This second system consisted of small laser diode (λ = 635 nm, red), a third, 30 frame per second CCD video camera, and a video recorder. The orientation of the PIV system and this secondary video system are seen in Figure 2.36.

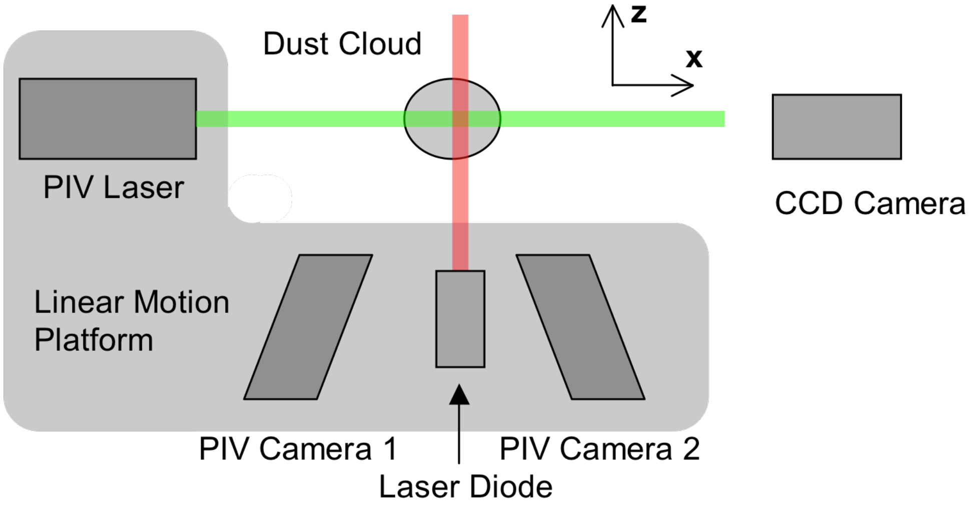


Figure 2.36: Schematic showing the orientation of the stereo-PIV and LLS imaging systems used to verify the functionality of the stereo-PIV system.

A triangular pulse of duration 650 ms raises the voltage on the perturbation electrode from V_p = 50 V to V_p = 100 V. As a result of this pulse, the particle cloud experiences both a vertical (y-direction) and an out-of-plane (z-direction) shift. Images from the second camera system that verify the motion in the z-direction are seen in Figure 2.37.

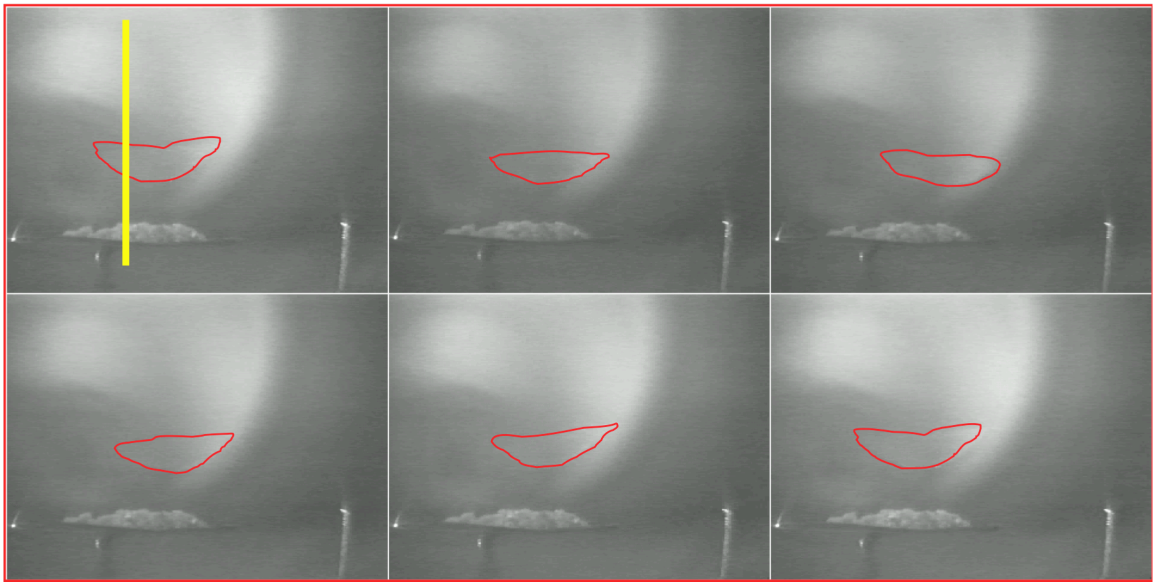


Figure 2.37: A video image sequence acquired using the second camera system confirming the displacement in the z-direction. For increased clarity, the cloud is outlined in red. The yellow rectangle in the first image denotes the location of the laser sheet for the stereo-PIV system. The dust tray that holds the source powder can be seen in the bottom, center of the images, while the perturbation electrode is visible in the lower right of the images.

A twenty image pair sequence (40 images per camera) was acquired using the stereoscopic PIV system. The lasers are fired with a time separation of Δt_{laser} = 2.5 ms and the time separation between successive image pairs is Δt_{image} = 0.2 sec. Four of the reconstructed velocity measurements from the 20-image sequence are shown in Figure 2.38. Here, the arrows represent the two-dimensional velocity of the microparticles in the plane of the laser sheet, while the contours represent the *z*-component of velocity (*i.e.* the out-of-plane motion). The measurements indicate that in the first (t = 0.8 sec) and fourth (t = 2.4 sec) images shown, the microparticles, on average, have no directed *z*-motion. However, at times t = 1.4 sec and t = 1.8 sec, there is a significant out-of-plane motion. The particle cloud is first pushed in the +*z*-direction with the application of the pulse, and then restored to its original position by returning in the -z-direction.

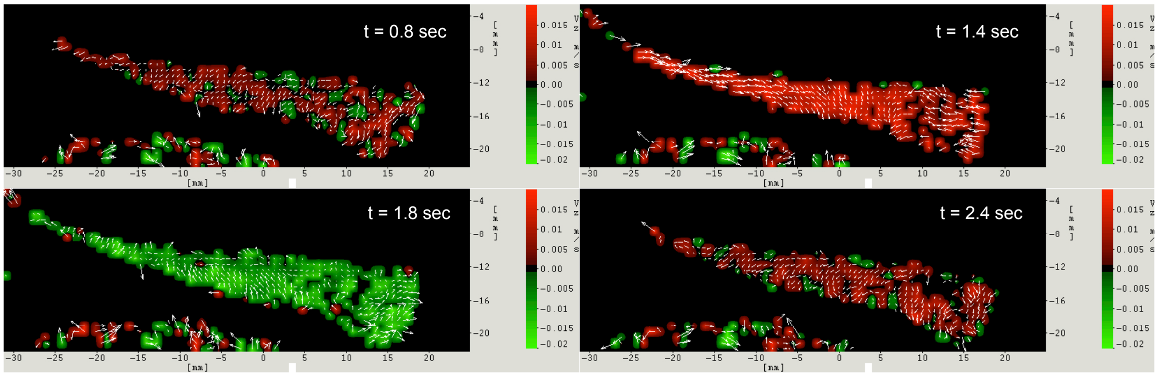


Figure 2.38: Four images of the reconstructed velocity field from the stereo-PIV system for times t = 0.8, 1.4, 1.8, and 2.4 seconds. The red-green shading at the center of each of the four panels represents the z-component of the velocity of the microparticle cloud. The image labeled t = 0.8 sec, is before the application of the pulse and there is no net z-motion. At t = 1.4 sec, the pulse is applied and the cloud is moving in the +z-direction as indicated by the bright red contours. At t = 1.8 sec, the pulse has reversed and the particle cloud is moving in the –z-direction as indicated by the green contours. By the last image, t = 2.4 sec, the perturbation pulse has ended and the cloud has returned to a state of no net z-motion.

This out-of-plane movement is further confirmed by plotting the distribution of the particle velocities in the z-direction as shown in Figure 2.39. Here, the distribution of the z-component of velocity, v_z, is plotted for each of the four cases shown in Figure 2.37. Again, it is noted that for first (t = 0.8 sec) and fourth (t = 2.4 sec) images shown, the velocity distributions are centered at v_z = 0 m/s. The t = 1.4 sec case is centered about v_z = 12 mm/s and the t = 1.8 sec case is centered about v_z = -10 mm/s.

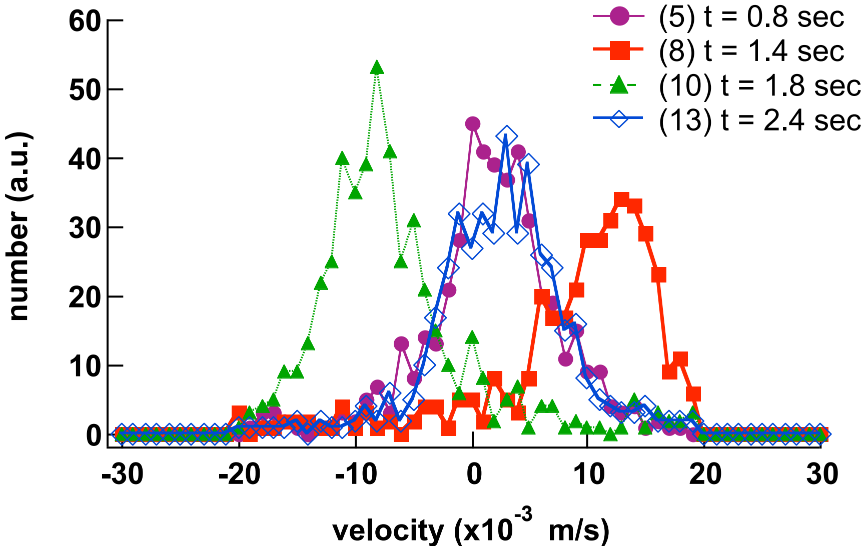


Figure 2.39: Distribution of the *z*-component of velocity for the four cases shown in Figure 2.37.

2.2.4.3 MEASUREMENT OF THE RESOLUTION LIMT OF THE SYSTEM

As discussed in Section 2.2.3, the stereo-PIV system is a complicated optical diagnostic. As a result, it is not surprising that it is difficult to directly calculate the measurement uncertainty for this system. In the past, there have been theoretical [98] and experimental [99] studies that have attempted to quantify the uncertainty of the stereo-PIV technique. In the theoretical study, a simple geometric error model for stereoscopic particle image velocimetry technique was developed. This model provides insight into the differences between the translational and angular configurations. However, the results are limited as a number of errors are not included in the error model developed. These sources of errors include lens aberrations, differences in the laser power used for measuring each PIV image and errors in the algorithm used in the cross-correlation analysis. Not surprisingly, there were observed differences between the theoretical results and what was measured experimentally. Further, one cannot necessarily apply the results from the experimental studies to other systems because there will be different sources errors and different quantities of errors in all of the elements of the system. As a result, the experimental and theoretical studies provide a guideline for uncertainty but they cannot necessarily be directly applied to other experimental setups.

Beyond the sources of errors that exist in the hardware components of the stereo-PIV system, there is an additional error that is introduced by the software algorithm that computes the velocity vectors. This numerical error can, and has been, quantified for various analysis schemes by using standardized, “ideal” images from the PIV Challenge (http://www.pivchallenge.org). It is noted that in this case, the images that are analyzed represent what one might call an “ideal” PIV measurement, where there are no errors in the system (*e.g.* error due to optical distortion) and the exact calibration is known. Using these images, the DaVis software used in these studies was found to have an error of 0.03 pixels [100].

It is, however, possible to experimentally measure the total uncertainty of a stereo-PIV system, using a “zero-displacement” test. To do this, two images are taken with a sufficiently small time increment between them that there will be effectively no motion between the two images. By running these images through the PIV analysis software, one can determine the uncertainty in the PIV measurement. Such a measurement will include all errors due to hardware and software sources. By plotting the distribution of vectors that are reported in this measurement, one can extract the uncertainty in the stereo-PIV measurement. This result depends on the calibration and optical setup. Consequently, this zero-displacement test needs to be repeated whenever the system is recalibrated.

The results of one such measurement can be seen in Figure 2.40. In this measurement the optical setup that was used in the experiments presented in Section 4.2 was used and the time between images, Δt_{laser}, was set to 1 μs. It is noted that there was no motion observed in the images taken in the zero-displacement test. Further, under the assumption that the particles having a peak velocity of 10 mm/s and an image resolution of 35 μm/pixel; a $\Delta t_{laser,}$ of 1 μs corresponds to a displacement of 0.0002 pixels. These values, which are typical for the experimental systems used in this dissertation, yield a displacement that is well below the limitation of the software algorithm used.

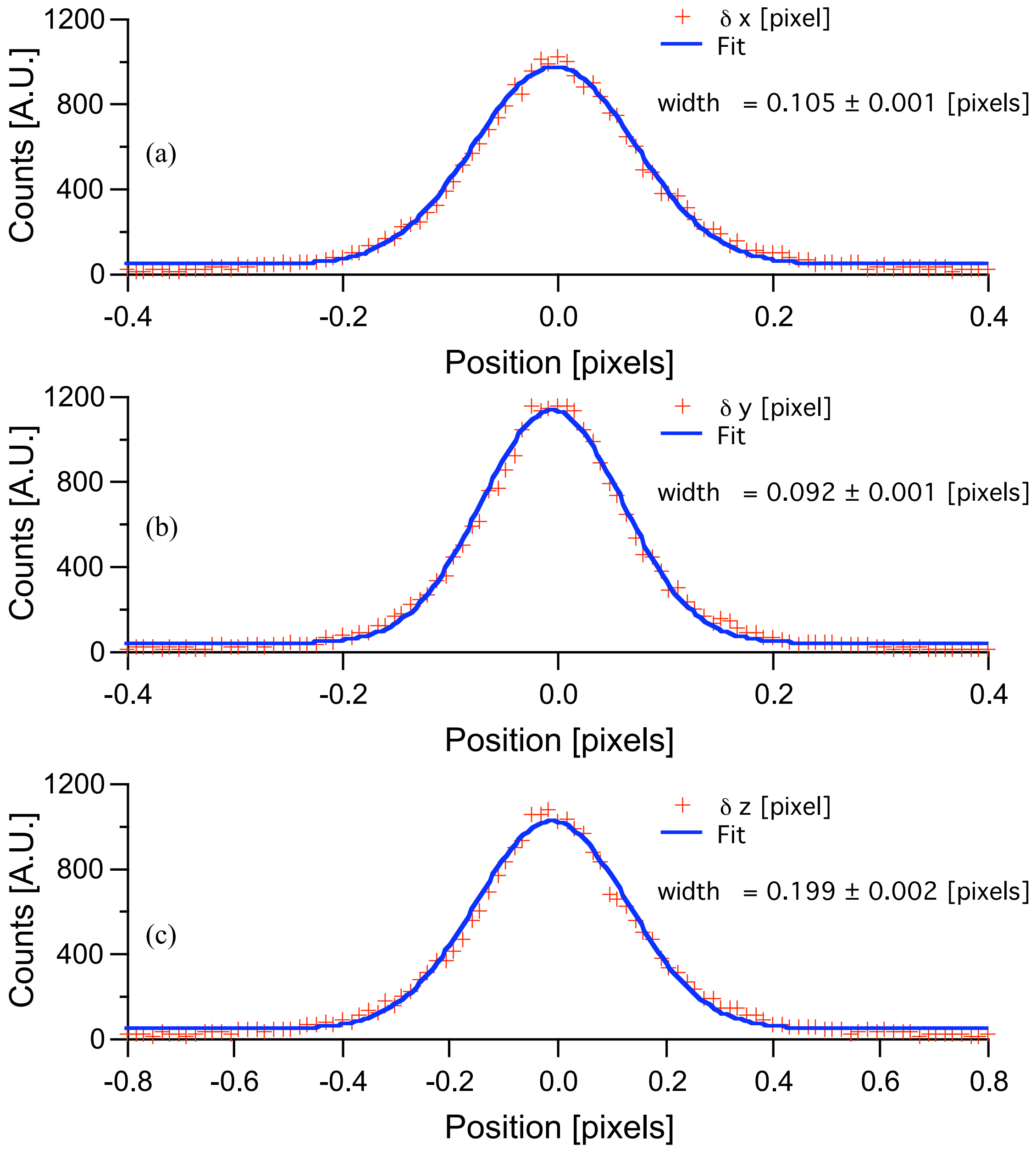


Figure 2.40: Showing the error in the stereo-PIV system in the (a) *x*-, (b) *y*- and (c) *z*-direction. The optical setup for this measurement corresponds to the one that was used in taking the data in Section 4.2 of this dissertation. It is noted that the error is comparable to what one observes in a two dimensional PIV system.

2.3 PARTICLE DENSITY MEASUREMENT

While the PIV technique is particularly well suited for studying the dusty plasmas that exist in the weakly-coupled regime, there are a few drawbacks to this technique. Unlike the particle tracking techniques that were used in the previous studies of the thermal properties of a strongly-coupled dusty plasma, the PIV technique does not return a velocity for each of the particles in the system being studied. Instead, what is obtained is a velocity that represents the average motion of a group of particles. The result of this is an intrinsic averaging in the velocities that are measured and is a well-known feature of the PIV technique [85, 101, 102] in the fluid dynamics community. In traditional PIV measurements, this effect is negligible and as a result, there have been no studies to quantify this. In the next chapter, the results of a computational study of this phenomenon are presented. However, at this time, two key results from these computational studies are noted.

First, the distribution of velocities that are measured using PIV techniques are narrower that the velocity distribution that is being measured. There is, however, a unique mapping function that directly relates what is measured to the underlying velocity space distribution function. This mapping function depends strongly on two features of the dust cloud: the number density of the suspended microparticles and the size distribution of microparticles that are suspended. Consequently, it is critical to know what the number density (*i.e.* particles/volume) of the microparticle component. Additionally, it is necessary to have knowledge of what is suspended in the particle cloud (*i.e.* are the suspended particles monodisperse or do they have a size distribution).

To address the issue of the number density of the suspended dust particles, three techniques have been used in previous experiments. The simplest approach involves directly counting the total number of particles from the acquired images. This technique requires each particle to occupy multiple pixels in the image of the dust cloud and very low number densities, such as what is common in the strongly-coupled systems seen in Figure 2.41.

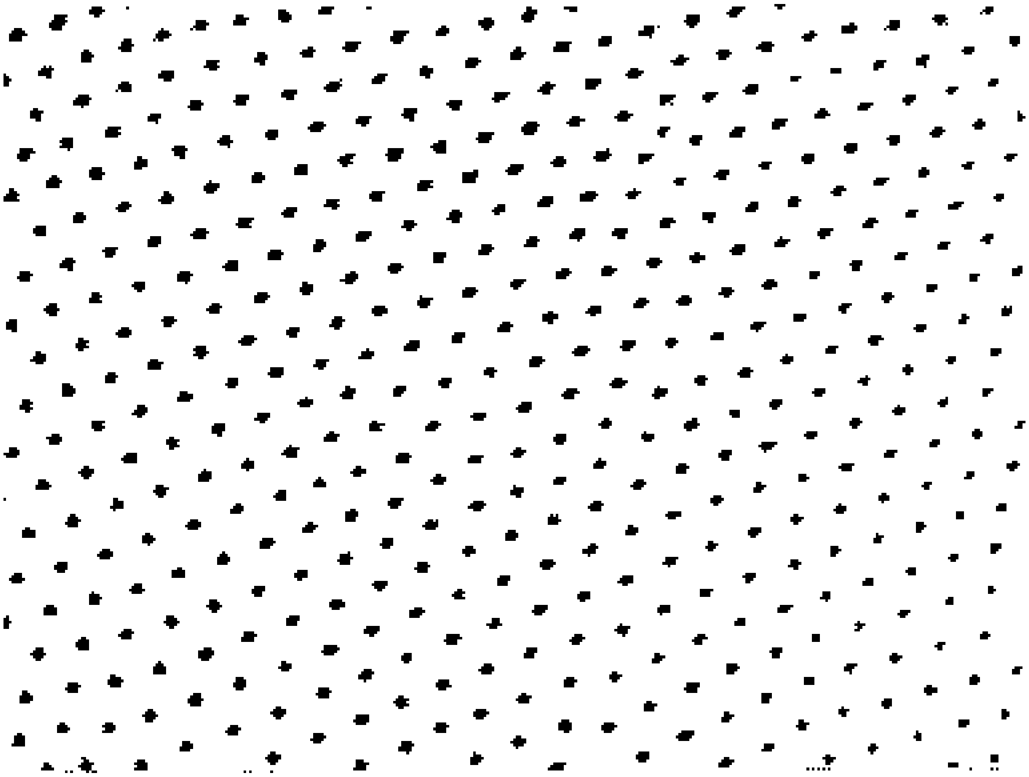


Figure 2.41: An inverted image of a strongly-coupled dusty plasma. (Figure courtesy of H. Thomas, Max Plank Institute for Extreterestrial Physics, Garching, Germany)

The second technique involves measuring the optical extinction and then, using Mie Theory, use this value to determine the number density [103-105]. Recently, a third method, which is an extension of the first technique, has been used [80]. In this approach, the amount of light scattered by a known number of particles is determined and used to compute the particle density, even in regions where the particle density is sufficiently high that individual particles can not be resolved. In the systems studied here, the number densities are sufficiently high that the first technique cannot be used. Additionally, the second technique requires, for the experiments presented in this dissertation, measuring extinction ratios are too small to be reliably measured. In Section 2.3.1, the third technique, which is used to determine the particle density, is discussed.

2.3.1 DENSITY MEASUREMENT BY LIGHT SCATTERING TECHNIQUE

In a recent study by the Piel group at Kiel University in Germany, a new technique for estimating the number density directly from video images was introduced [80]. In this method, the density of the microparticle component can be determined from the video images using Equation 2.12

$$n_{dust} = \frac{I_{total}}{V_{cloud}}\eta \tag{2.12}$$

where n_{dust} is the number density of the microparticle component, I_{total} is the amount of light scattered by the dust cloud, V_{cloud} is the volume of the dust cloud illuminated by the laser sheet and η is a scattering efficiency. The amount of light scattered by the dust cloud and the volume of the dust cloud are measured directly from the video images. The scattering efficiency is a measure of the amount of light scattered by a single dust particle and is measured from the video images in regions of known particle density, *i.e.* in a region where one can manually count the number of particles such as what is seen in Figure 2.42.

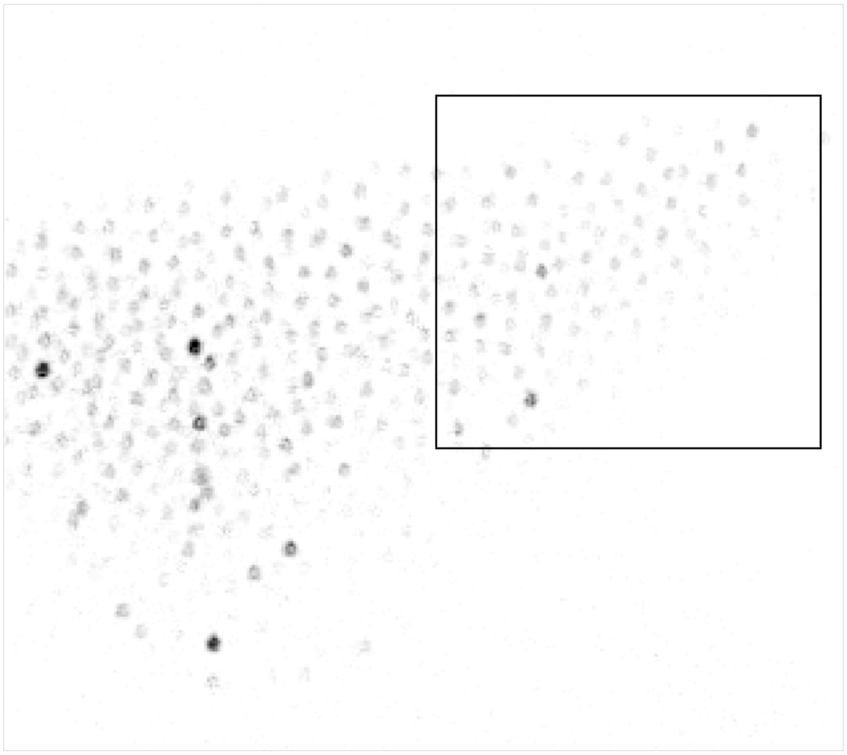


Figure 2.42: An inverted image of a dust cloud showing a region of known density, highlighted by the box. The image has been inverted to better see the dust particles.

There are a few caveats to this technique that bear mentioning. First, the images of the dust cloud analyzed cannot have any saturated pixels. For the data presented here, the PIV cameras used have sufficient depth, 12 bits of resolution or 4096 grayscale levels compared to 256 levels for a typical 8 bit camera, that this was not an issue. Secondly, the suspended microparticles must be essentially monodisperse. This result is supported by observations described in Section 2.3.2, and by simulation results, to be discussed in Section 3.3.2.3. Finally, the amount of light scattered per particle, *i.e.* the scattering effeciency η, needs to constant. In regions where one can measure the particle density from the video images, this was found to be the case to within 5% and is described in Section 2.3.5.2.

2.3.2 MONODISPERSE VS. POLYDISPERSE DISTRIBUTIONS OF THE MICROPARTICLE COMPONENT

If a dusty plasma is composed of microparticles that have a size distribution, there is an outstanding question regarding what subset of the total distribution is suspended in the plasma. Although there have been attempts to measure the suspended distribution in past studies [106], this remains a difficult and unresolved issue. However, it is possible to determine if the suspended microparticles have a size distribution or if the suspended microparticles are monodisperse. There are two methods of determining this.

The first method comes from examining the shape of the velocity distribution that is measured. Simulations of the PIV measurement have revealed that there is an anamolous peak that appears at zero in the velocity distribution measured using the PIV technique. This will be discussed in detail in Section 3.3.2.3. The second method is to examine the images of the suspended dust particles. Based on studies by McKee *et al.* [107] and Annaratone *et al.* [108], the suspension of different sized particles gives rise to unique structures in the observed dust clouds. In particular, the dust clouds tend to orient themselves into distinct band or regions of different particle size. In Figure 2.43, an image of a dust cloud with multiple particle sizes is seen. Here, the suspended cloud contained two distinct particle sizes, spheriglass having a mean size of 10 µm and 325 mesh (≤ 45 µm) silica. While it is unclear what is being suspended from size distribution of the dust used in these experiment, the banding structure indicates that particles having multiple sizes are suspended.

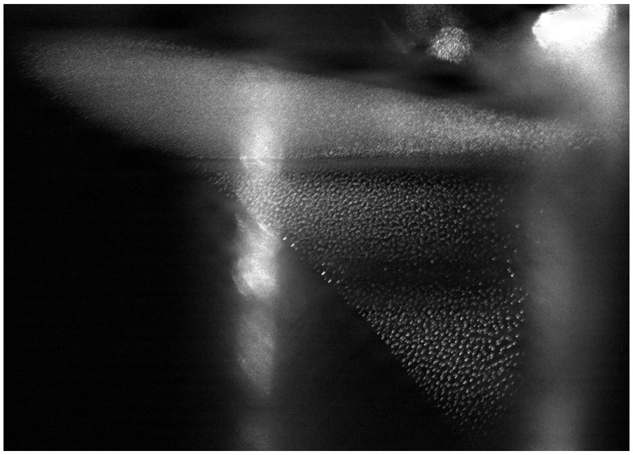


Figure 2.43: Video image of dust clouds composed of multiple particle sizes. The original particle distributions included spheriglass particles having a mean size of 10 µm and 325 mesh (≤ 45 µm) silica. The banding structure that is observed is a characteristic feature of dust clouds containing a size distribution.

2.3.3 DETERMINATION OF THE SCATTERING EFFICIENCY.

A key parameter in the method described in Section 2.3.1 to determine the number density of the microparticle component is amount of light scattered per particle, η. This parameter is measured using a series of images where the number density is sufficiently low that it is possible to count the number of particles in the video images. The total pixel intensity is computed in this region and the ratio of the number of particles to total pixel intensity yields the quantity η.

To do this, a series of 45 - 60 images were analyzed. After a background was subtracted, the image sequence was processed using the code found in Appendix A.4.5. In this code, each image was opened and the user selected a region of countable particle density. If the number density was sufficiently low that the particles could be uniquely identified, an automatic routine was used. If the particle density was too high for this to be done, *i.e.* if there was overlap of particles such that the software could not distinguish all of the particles in the defined region, the number of particles was manually counted. The software then computed the total pixel intensity in the region of interest. From these quantities, the intensity per particle, η, is computed. In the regions where the user had to manually count the number of particles, there was a fluctuation level of approximately 5%. Figure 2.44 shows the typical fluctuation level in this parameter.

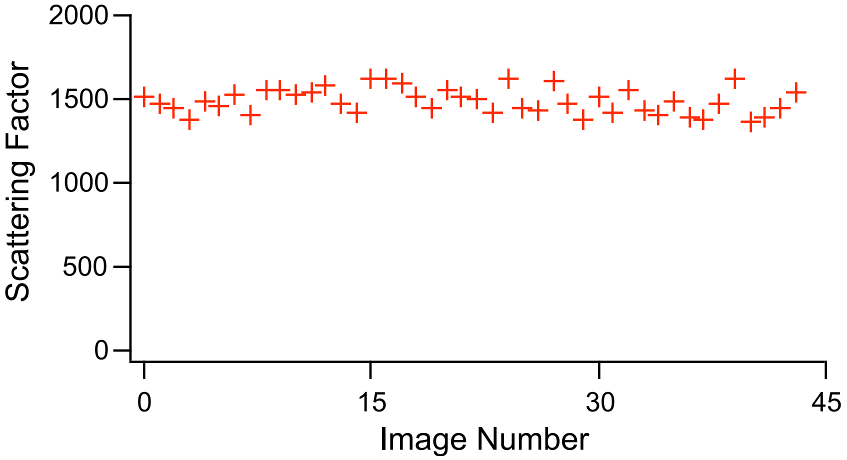


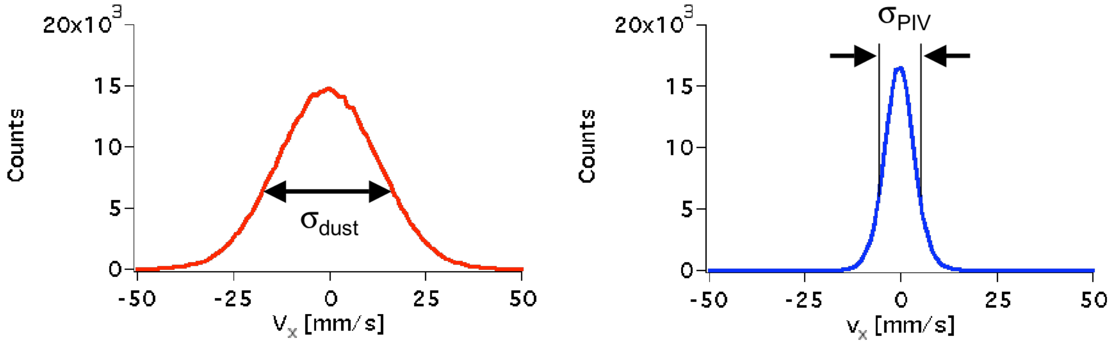
Figure 2.44: Showing the fluctuation level in the scattering efficiency, η, as a function of image number used in the calibration procedure. The fluctuation level is five percent.

CHAPTER 3: SIMULATION OF THE PIV MEASUREMENT

3.1 MOTIVATION

Knowledge of the kinetic velocity space distribution function allows one to fully describe the thermodynamic state of a given system. Through the use of stereoscopic PIV techniques, it is possible to measure the three-dimensional distribution of kinetic velocities for the microparticles component of a dusty plasma. In these experiments, the stereo-PIV system measures the three-dimensional velocities of the particles illuminated using a one millimeter thick laser sheet. By scanning the stereo-PIV system through the entire cloud, a reconstruction the full three-dimensional distribution of velocities is determined. This measurement is described in detail in Chapter 4.

An important concern that has been demonstrated in studies within the fluid mechanics community is that PIV measurements have an inherent bias toward smaller values of velocity [85, 101, 102]. This arises from the fact that in the PIV measurement technique, each velocity vector represents the average motion of a cluster of tracer particles. In this averaging process, the motion of the higher speed particles tends to be suppressed. As a result, a distribution of velocities that is measured using PIV techniques is generally narrower than the underlying velocity distribution function, as illustrated in Figure 3.1. Consequently, the distribution of velocities that is measured using PIV techniques is not the true velocity space distribution function of the system being studied, but rather something that is closely related.



(a)

(b)

Figure 3.1: Showing the effect of the PIV measurement on the measured distribution. In (a), one sees the velocity distribution that is being measured, while (b) shows the distribution that is measured using the PIV technique. It is noted that the measured distribution has a narrower width, which is a result of an averaging that is intrinsic to the measurement technique. Physically, the result of this averaging would be the measurement of a smaller temperature, i.e. $T_{PIV} < T_{dust}$.

This chapter describes a series of simulations that attempt to relate the distribution of velocities measured using the PIV technique to the underlying velocity space distribution. With such a mapping function, it will be possible to estimate the temperature of the dust component of a dusty plasma. It will be shown that this mapping function is influenced by the size distribution and by the particle density of the suspended mircoparticles.

The existence of a mapping function and the effect that these experimental parameters have on that mapping function are examined by numerically simulating the PIV measurement. In Section 3.2, the simulation methodology is described. Section 3.3 presents the results of these simulations. Section 3.4 contains a discussion of these results and how these simulations are connected to real experiments.

3.2 COMPUTATIONAL METHODOLOGY

When a PIV measurement of a dusty plasma is made, two video images, separated in time by Δt_{laser}, are acquired. Each video image is decomposed into $n \times n$ pixel interrogation regions, where n is typically between 12 and 32. Using sequential pairs of images, the average displacement, d, for all of the particles located in an interrogation region is computed. Using these two quantities, a spatially averaged velocity vector is constructed for each interrogation region. Experimentally, velocities are computed on a grid with a 50% overlap to maximize the accuracy of the reconstructed velocity field. This process is illustrated in Figure 3.2. The simulations described in this chapter seek to reproduce this process.

A simulation of the PIV measurement has been developed to compare the distribution that is produced by the PIV measurement to the underlying distribution function. In the simulation code, an analysis grid of 400 × 400 bins is created. Each bin is assigned a random number of particles up to ten.

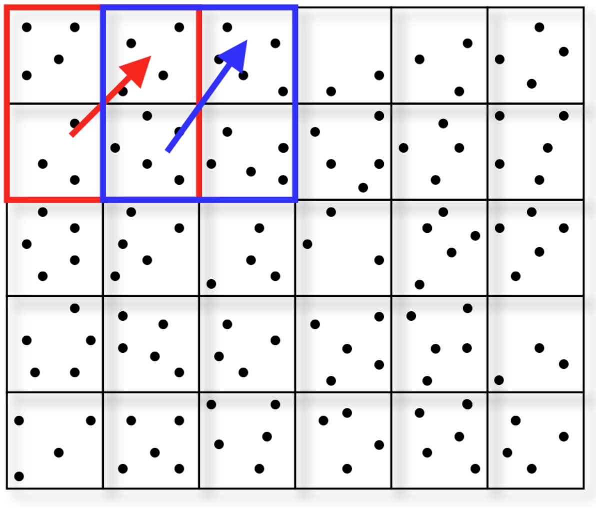


Figure 3.2: A cartoon illustrating the PIV measurement. The dots represent the particles that seed the flow of interest and the red and blue boxes denote interrogation regions over which the PIV analysis is performed. Here, the interrogation region would be a 2 × 2 bin region. For the experiments presented in Chapter 4, a bin corresponds to a 6 × 6 pixel region of the acquired images. The PIV technique returns a vector that describes the average motion of all of the particles in the interrogation region. Two examples are seen in red and blue. It is noted that there is a 50% overlap in the interrogation region.

Each particle in the simulation is assigned three physical attributes: a radius, a mass density and a three-dimensional velocity vector. In assigning these values, the following assumptions are made. All of the particles are assumed to be spherical. Therefore, the selection of a mass density, which represents the material composition of the particles, *e.g.* silica, alumina or melamine formaldehyde, and a radius simulataneously selects the particle mass. Additionally, all of particles are assumed to have radii large enough to be detectable by the PIV imaging system. In the experiments performed in this dissertation, the lasers used have a wavelength of 532 nm. As such, the simulation particles are constrained to have a radii $r \geq 0.25$ μm.

The assignment of the velocity attribute depends on the type of particle size distribution, monodisperse or polydisperse, being studied. For simulations having monodisperse (uniform size) particles, the mass is computed directly from the particle radius and mass density. For each vector direction, the particles are assumed to follow a 1D Maxwellian velocity distribution that is parameterized by the particle mass, m, and a kinetic temperature, T_d, as defined in Equation 3.1.

$$f\left(v_j\right) = \sqrt{\frac{m}{2\pi k_B T_{d,j}}}\exp\left\{-\frac{m\left(v_j - v_{d,j}\right)^2}{2k_B T_{d,j}}\right\} \tag{3.1}$$

where, v_j is the velocity of a particle, $v_{d,j}$ is the drift velocity, k_B is Boltzmann's constant and the subscript j refers to the vector direction ($j = x, y, z$). The velocity of the tracer particle is then extracted at random from the velocity distribution. It is noted that, for most of the simulations presented, the drift velocity, $v_{d,j}$, was set to zero. Further, the velocities are selected for each vector direction (x, y and z) from their respective 1D Maxwellian distribution.

In the simulations involving a polydisperse size distribution, there is an additional step because the particles have different sizes. First, the particle radius, $r_{d,j}$, was extracted at random from a normal distribution of radii, defined by the mean radius, μ_r, and the width, δr, of the size distribution in Equation 3.2.

$$r_{d,j}(\mu_r, \delta r) = \frac{1}{\sqrt{2\pi\delta r}}\exp\left\{-\frac{\left(r - \mu_r\right)^2}{2\delta r}\right\} \qquad r \geq 0.25\ \mu m \tag{3.2}$$

The mass, $m_{d,j}$, of the j^{th} particle is then computed from this radius. It is noted that the truncation of particle sizes in Equation 3.2 corresponds to the diffraction limit that is observed experimentally. Under the assumption that all of the particles have the same kinetic temperature in a given vector direction, this mass and the kinetic temperature are used to generate a Maxwellian velocity distribution from which the velocities, $v_{d,j}$, are extracted at random in the same fashion described previously

To simulate the PIV measurement, the average velocity of all of the particles in an interrogation region of size 2 × 2 bins is computed. For reference, a bin is represented by a single box in Figure 3.2. This is repeated over the entire simulation grid using a 50% overlap in the evaluation region, which is consistent with what is done in the PIV velocity reconstruction algorithm used in the experimental work presented in this dissertation. A single bin corresponds to a 6 × 6 pixel region of the image used in the PIV analysis.

To verify the accuracy of the code, a simulation of an ideal PIV measurement is made. In this simulation, particles having a uniform size and temperature (*i.e.* $T_x = T_y = T_z = T$) are uniformly distributed over the simulation grid. The results of this simulation is seen in Figure 3.3,where the input velocity distribution, represented by crosses, and the velocity distribution that is generated by the simulated PIV measurement, represented by the solid curve, are plotted. The simulated PIV measurement clearly shows a narrower velocity distribution than the input distribution. This is consistent with the expected suppression of larger particle velocities and gives confidence in the performance of the code.

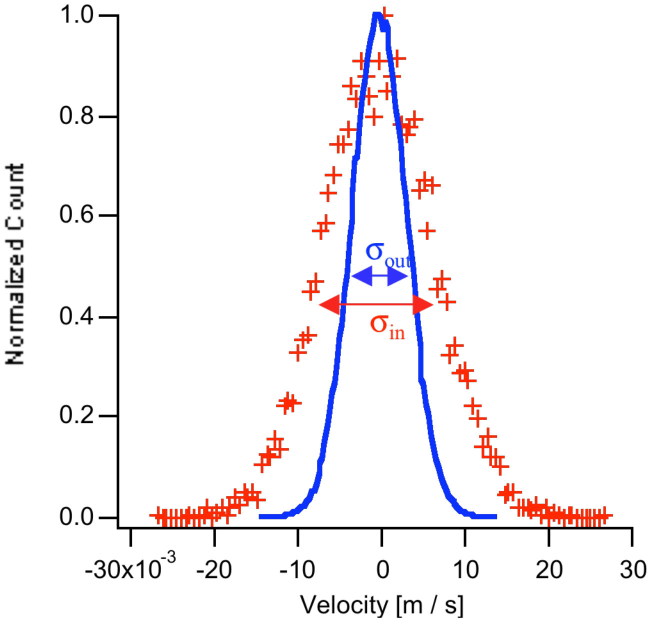


Figure 3.3: The result for a simulation of an ideal PIV measurement is depicted. The crosses depict the distribution of velocities that is input into the PIV simulation, while the solid line depicts the distribution of velocities that is returned by the simulated PIV measurement. The expected narrowing of the distribution of velocities is observed.

3.3 RESULTS

In Section 3.2, it was demonstrated that the simulation code yields the anticipated results for an ideal PIV measurement. In this section, simulation results are presented covering a wide range of parameters that seek to characterize the relationship between the input and output velocity distributions. In these simulations, several important parameters are considered for their potential influence on the output velocity distribution: radius (mass) of the tracer (dust) particle, kinetic temperature of the tracer (dust) particles, number density of the tracer (dust) particles and, in the case of particles having a size distribution, the width of the size distribution. It will be shown in this section that the number density of the particles and the width of the size distribution play a dominant role in determining the relationship between the input and output velocity distributions.

For most of the simulation results presented, the data is presented in the format of the ratio of the width of the output (PIV measurement) to the input (velocity space distribution) velocity distribution, σ_{out}/σ_{in}, as a function of the effective number density of the tracer (dust) particles, which is computed directly over the simulation grid. It is noted that the value of σ is a function of the particle mass and kinetic temperature. In these results, the mass is known and as a result, the value of σ can be regarded as equivalent to the kinetic temperature.

3.3.1 UNIFORM MASS

The first series of simulation results involve particles having a uniform radius/mass. Here, alumina particles with a mass density of $\rho = 3800$ kg/m^3 and a radius $r = 0.6$ μm are considered. A series of simulations are performed considering effective number densities ranging 0.01 to 5 particles/bin. For each value of the effective number density, 24 simulations are run in which the kinetic dust temperature that defines the width of the input distribution is varied between 1 eV $\leq T_d \leq$ 300 eV. The results of these simulations are shown in Figure 3.4, where the blue bar represents the mean value for σ_{out}/σ_{in} for all of the temperature cases and the error bars in red represent the maximum and minimum values for σ_{out}/σ_{in} for each effective particle density.

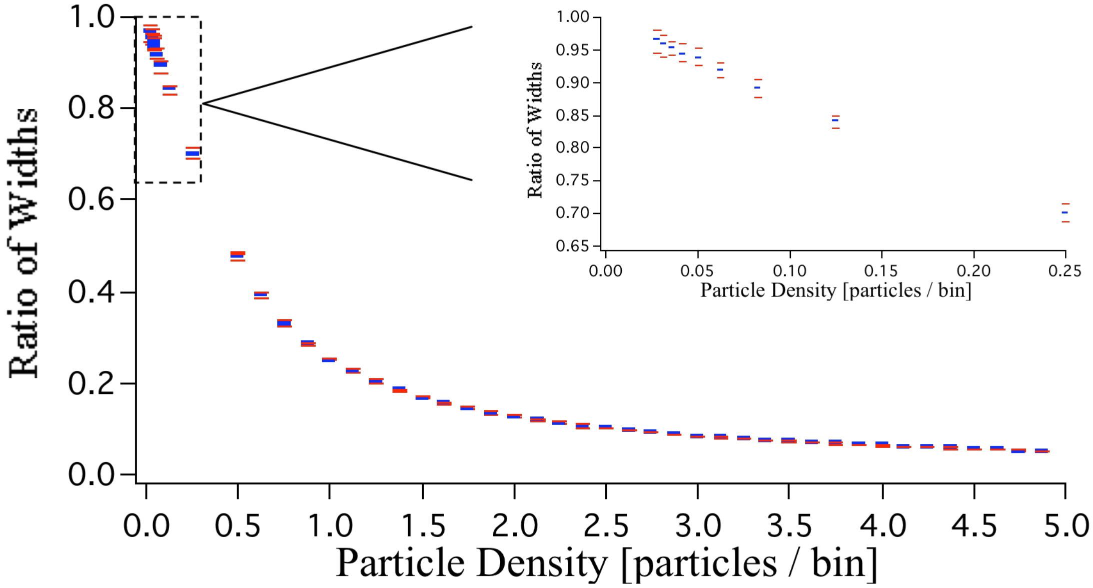


Figure 3.4: Plot of the ratio of the width of the output to the input velocity distribution, σ_{out}/σ_{in}, as a function of the effective particle density for tracer particles having a uniform size. Here, the results for alumina particles (r = 0.6 µm, ρ = 3800 kg/m^3) are shown. The blue symbol represents the mean value of σ_{out}/σ_{in} for 24 simulations over a wide range of kinetic temperatures, while the error bars represents the extreme values observed for the ratio σ_{out}/σ_{in}.

It is observed that the temperature of the particles has no significant impact on the width of the output velocity distribution at any given density. However, the number density of particles clearly has a strong influence on the simulation results. It is believed that this is due to the averaging that occurs in the PIV measurement.

Since a PIV measurement yields the average motion of all of the particles located in an evaluation region, the particles whose velocities are toward the tails of the distribution function (*i.e.* generally those with higher velocities) tend to be suppressed in the averaging process. As the number density of particles increases, the simulation result suggests that this averaging process becomes increasingly important and the width of the measured distribution of velocities becomes smaller.

By contrast, as the particle density decreases, the ratio of widths approaches unity. Experimentally, this corresponds to the PIV system functioning as a particle tracking system. In this low-density limit, each individual particle is measured directly and there is minimal averaging. In the limit of the effective particle density going to zero, the distribution of velocities that is measured is the underlying distribution function and one would expect that the ratio of widths would go to one. This result is a further validation of the correct functioning of the simulation code.

In Figure 3.5, two separate simulations are shown comparing two different particle masses with a fixed kinetic temperature, T_{dust} = 15 eV. Here, alumina particles (r = 0.6 μm, ρ = 3800 kg/m^3) are depicted by crosses, while silica particles (r = 1.45 μm, ρ = 2600 kg/m^3) are depicted by circles. For these two particle parameters, the mass of the silica particles is larger than the mass of the alumina particles by a factor of approximately ten. This simulation result suggests that the particle mass has no impact on the results of the PIV measurement. This result is not surprising, considering that the kinetic temperature was observed to have no effect on the PIV measurement. Both the mass and the kinetic temperature of the particles will define the width of the velocity distribution function that is being measured (*i.e.* the input into the simulations being discussed here). However, this does not influence the measurement process. As a result, the ratio of the width of the output to the input velocity distribution is unchanged.

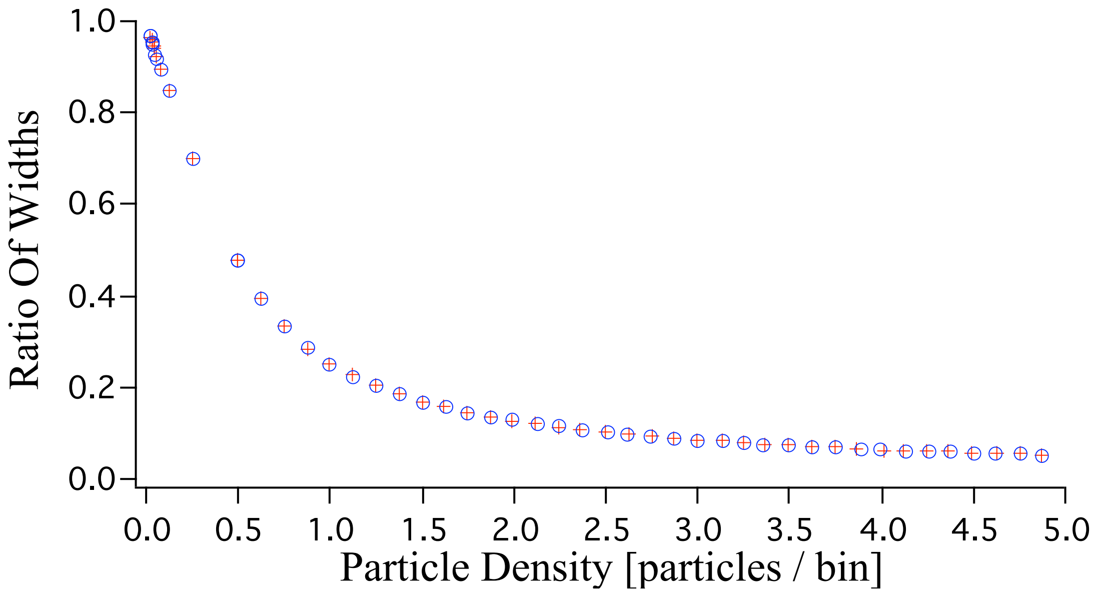


Figure 3.5: Plot of the ratio of the width of the output to the input velocity distribution, σ_{out}/σ_{in}, as a function of the effective particle density for tracer particles having a uniform size. Here, the result of a PIV simulation comparing two types of tracer particles is shown. The crosses represent alumina particles (r = 0. 6 µm, ρ = 3800 kg/m^3) and circles represent silica particles (r = 1.45 µm, ρ = 2600 kg/m^3).

In the simulations presented above the drift velocity of the particles was set to zero. When including a drift velocity, there is no observed effect on the PIV measurement. This is seen in Figure 3.6, where the input and output velocity distribution for an effective number density of 3.5 particles per bin are overlaid. As was seen previously, a narrowing in width of the distribution returned by the PIV measurement is observed. It is also noted that there is no effect on the value of the drift velocity, v_d = 0.2 m/s.

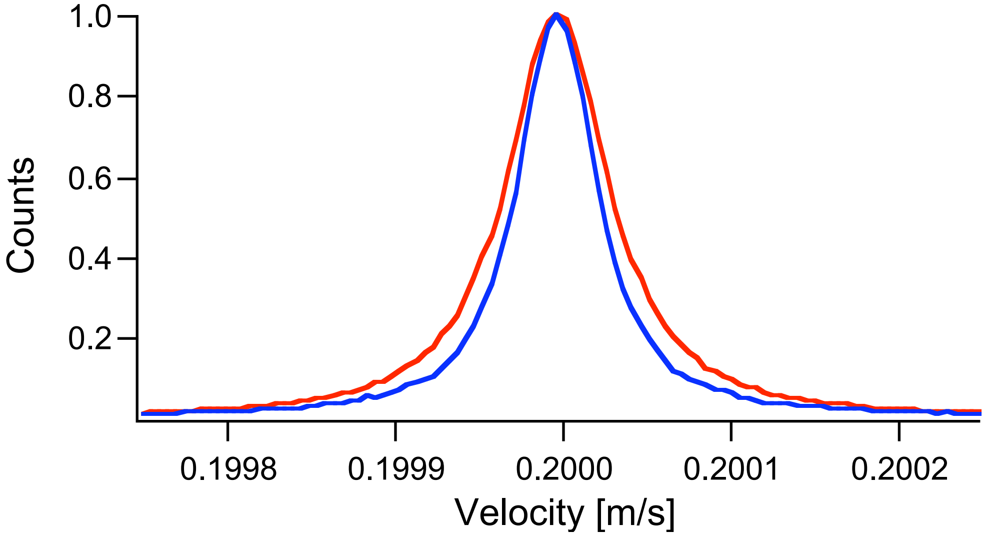


Figure 3.6: Showing the results of a simulated PIV measurement when the tracer particles have a non-zero drift velocity. Here, the red curve shows the underlying velocity space distribution that is being measured, while the blue curve depicts the distribution of velocities that is measured using the PIV technique. It is noted that the narrowing of the measured distribution is observed, but there is no observed effect of the value of the drift velocity

In Figure 3.7(a), the ratio of widths is plotted as a function of effective number density for simulations dust particles having different drift velocities. Here, the circles represent the simulation where the drift velocity was nominally 0.2 m/s, while the crosses represent the results when $v_d = 0$ m/s. In Figure 3.7(b), the computed drift velocities for the results seen are plotted as a function of the effective number density. These results again show that a net drift velocity has no impact on the resulting PIV distributions.

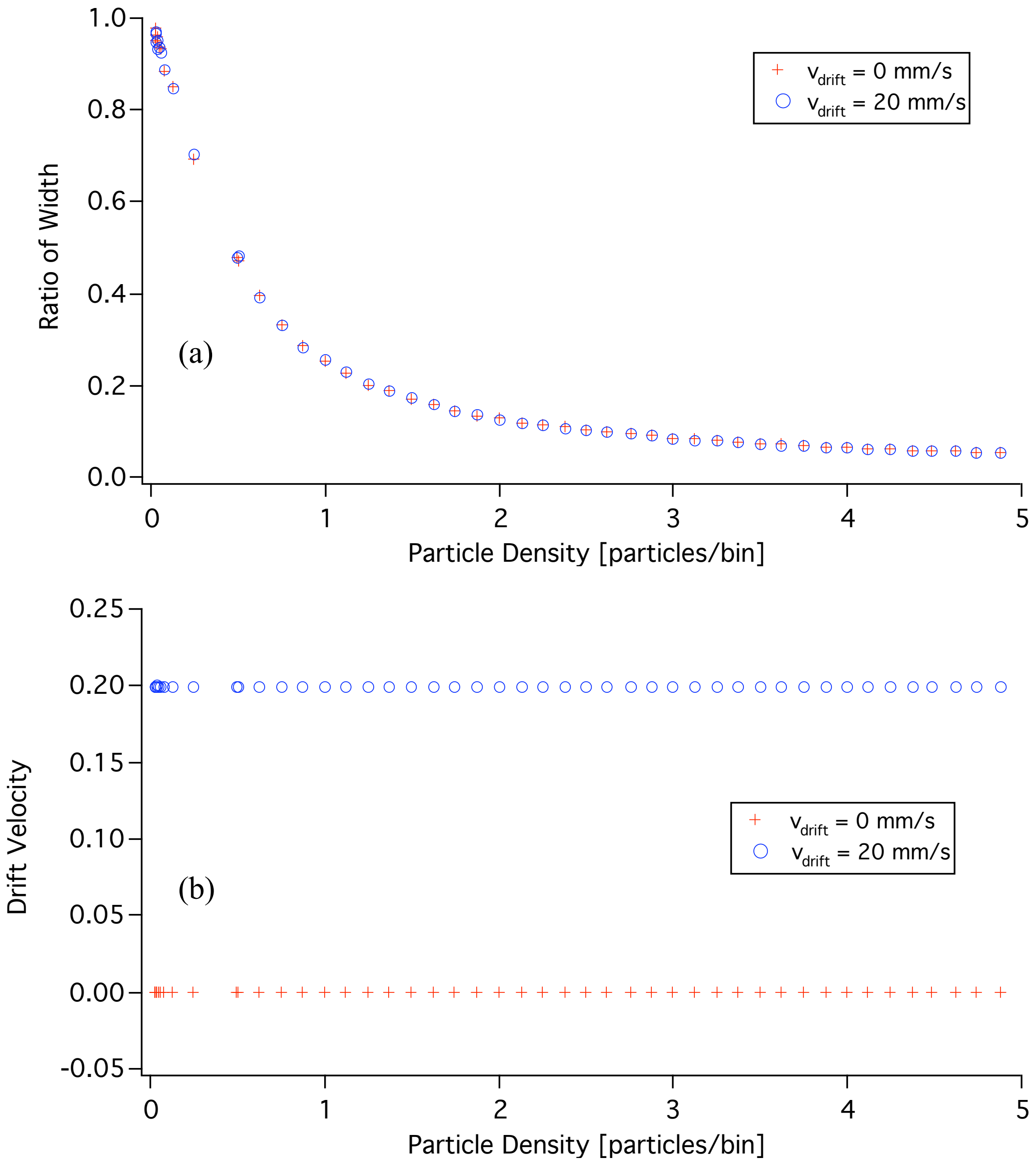


Figure 3.7: Showing the effect that the drift velocity has on the PIV measurement. In (a), the ratio of widths is plotted as a function of the effective particle density. Here the crosses denote the result when there is no drift velocity and the circles represent the result when there is a non-zero drift velocity. The drift velocities used in the simulation are seen in (b).

3.3.2 MASS DISTRIBUTION

In most of the early experiments, the dust particles used were polydisperse; *i.e.* there was a distribution of particle sizes. Therefore it is necessary to determine the effect of a size (mass) distribution has on the velocity distributions that are measured using the PIV technique. This section presents the results from simulations where the particles measured using the PIV technique had a size distribution. Because of the size distribution, care must be taken in the simulation process and in the analysis of the data. In particular, there are three issues that need to be considered: the assignment of particle velocities, the determination of the width of the velocity distribution and the way in which the PIV calculation is performed.

The process by which the velocities are assigned is complex, when compared to the work that was discussed in Section 3.3.1. A particle size is extracted at random from a distribution of particle sizes defined by the mean radius, μ_r, and the width of the size distribution δr, defined in Equation 3.2. This size (mass) is then used to generate a velocity distribution for a defined kinetic temperature of the dust, under the assumption that all of the particles have the same temperature. Velocities are then extracted at random from this distribution. This process is repeated to form the velocity distribution that the simulation will measure. The resulting velocity distribution will have an effective width that is defined by an input temperature and a mass distribution defined by the size distribution, $\mu_r \pm \delta r$. As seen in Figure 3.8, the velocity distributions that are generated when there is a size distribution appears Maxwellian with a larger width than those that are generated for a monodisperse particle size for the same kinetic temperature.

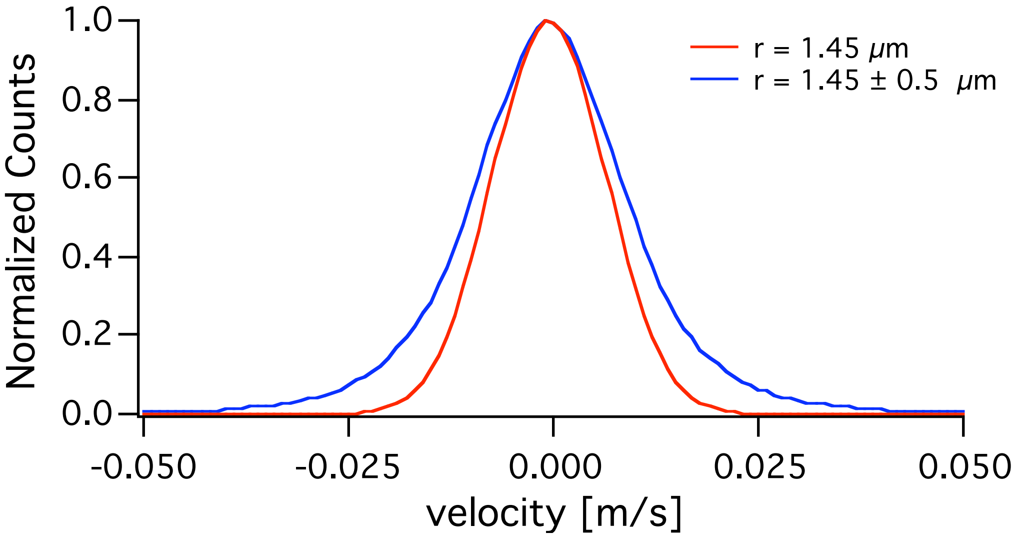


Figure 3.8: A plot showing the overlay of the velocity distribution for monodisperse silica particles, $r = 1.45$ μm, in red and polydisperse silica particles, $r = 1.45 \pm 0.5$ μm, in blue for the same kinetic temperature, $T_{dust} = 15$ eV. It is noted that the velocity distribution for polydisperse particles has a larger width.

It is no longer possible to simply fit the input (*i.e.* the underlying velocity space distribution that is being "measured" in the simulation) and output velocity distribution (*i.e.* what is "measured" in the simulation) to a Maxwellian. The reason for this lies in the multiple possible value of the mass. When there is a distribution of particle sizes, there is not a single mass to use in the fit and as a result a range of masses can be used in the fit. This leads to a distribution of possible temperatures.

As a result, an alternative method is needed to compute an effective width of the velocity distributions. In this method, the velocity distribution is fit to a Maxwellian assuming that all of the particles have a known radius, r_i. The resulting width is then weighted to the probability, α_i, that a particle has that given size. The process is depicted in Figure 3.9.

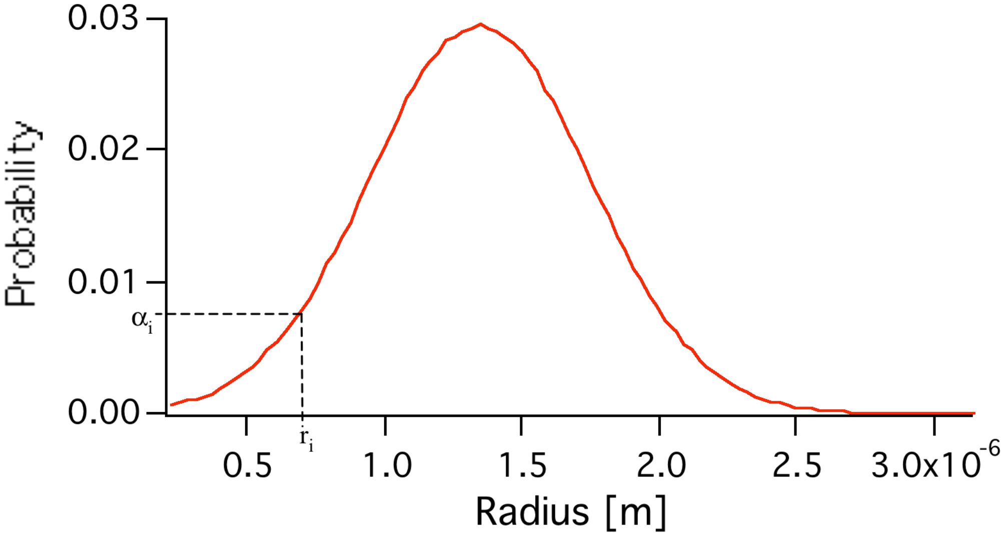


Figure 3.9: Showing how the weighted average was used to find the effective width of the velocity distribution when the particles had a size distribution in the PIV simulations.

Here, the distribution of particle radii were used in the simulations are fit to a normal distribution. From this fit, the probability, α_i, that a particle has a particular radius, $r_{i.}$, is extracted as seen in Figure 3.9. Starting at the minimum detectable particle size, $r = 0.25$ µm, the distribution of velocities that is generated in the simulated PIV measurement is fit to a Maxwellian, under the assumption that all of the particles have this radius. This process is then repeated by moving the assumed radius through the size distribution N times using equally spaced increments for the particle size. At each size, r_i, a width, σ_i, is extracted from the fit of the velocity distribution and probability that the particle has that radius, α_i, is extracted from the fit of the size distribution. The effective width of the velocity distribution, $\sigma_{effecctive}$, is computed using Equation 3.3.

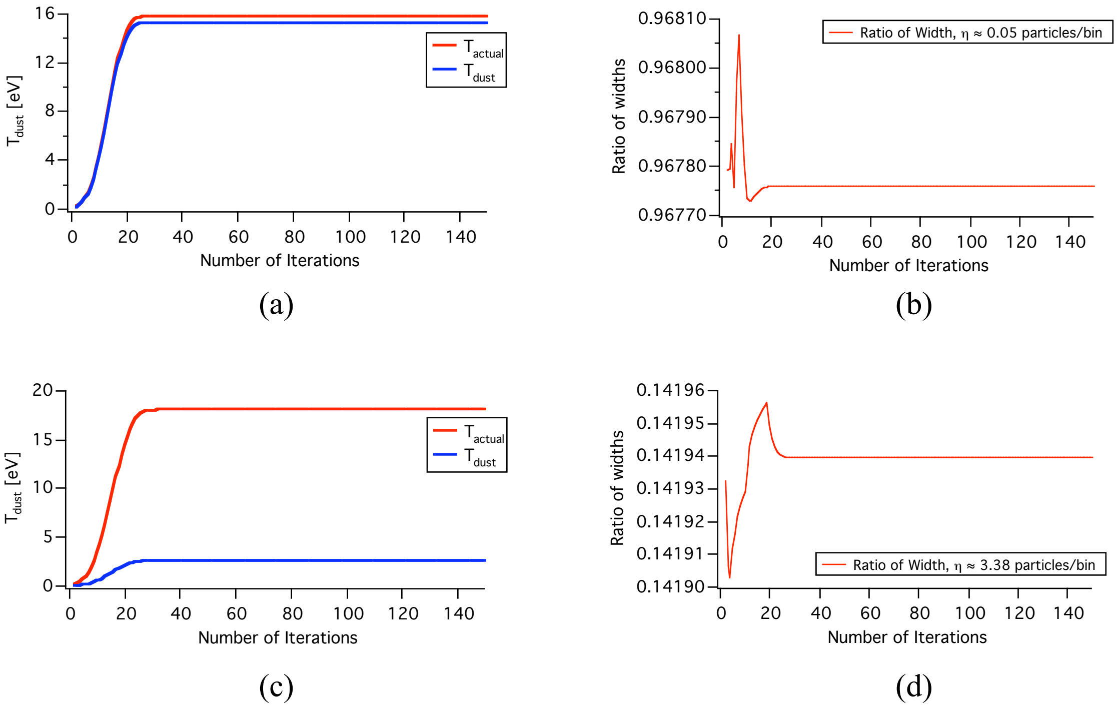
$$\sigma_{effective} = \frac{\sum_i \alpha_i \sigma_i}{\sum_i \alpha_i} \tag{3.3}$$

where the sum runs from 1 (*i.e.* a particle of having radius r = 0.25 μm) to N (*i.e.* a particle of having radius $r = \mu_r + 9.5\sigma$, where σ is the width of the size distribution). In is noted that in the limit of monodisperse dust particles (*i.e.* of $\delta r \rightarrow 0$), the effective width defined in Equation 3.3 reduces the width that was used in Section 3.3.1. The value of N was selected such that it was large enough that the effective width that was computed no longer fluctuated, as seen in Figure 3.10; typically, $N \geq 100$. The same process is used to find an effective width of the velocity distribution that is input into the PIV simulation.

Finally, the process by which the PIV measurement is simulated needs to be addressed. In the previous section involving monodisperse particles, the PIV measurement is simulated by computing the average velocity of all of the particles in the interrogation region using Equation 3.4.

$$v_j = \frac{1}{N_j} \sum_i v_i \tag{3.4}$$

where N_j is the number of particles in the j^{th} interrogation region and v_i is the velocity of the i^{th} particle, which is located in the j^{th} interrogation region.



(a) (b)

(c) (d)

Figure 3.10: Showing the number of iterations needed to compute a meaningful value for the weighted average.

When the particles are monodisperse, this provides an accurate model of the measurement, as all of the particles will contribute equally in the cross correlation analysis that generates the PIV velocities. However, when the particles are polydisperse, all of the particles will not contribute equally in the cross correlation analysis. Assuming that the particles have a diameter larger than the wavelength of light used, it is known from Rayleigh Scattering that the amount of light that scatters off of the a particles scales with the square of the particle size. As a result, the contribution to the cross correlation will go like the size of the particle to the fourth power. Consequently, larger particles will contribute more in the calculation of the PIV vectors. Experimentally, this can be accounted for by using multi-intensity layer PIV [109]. In this approach, one thresholds the image and performs the PIV calculation using a small range of pixel intensities. This analysis is repeated over the range of pixel intensities that occur in the image. Once this has been done a number of times spanning the intensity in the acquired images, one combines the results of each evaluation into a single vector field. Experimentally, this technique is not used in the work presented in this thesis and it is necessary to understand what effect this would have on the measured distribution.

Nevertheless, there is insight that can be gained into the PIV technique by assuming that polydisperse particles contribute equally in the PIV calculation. As a result, the results of simulations involving polydisperse particles using Equation 3.4 for the PIV calculation will be presented in Sections 3.3.2.1 and 3.3.2.2. In Section 3.3.2.3, simulations that account for the weighting due to particle size will be presented.

3.3.2.1 FIXED WIDTH OF PARTICLE SIZE DISTRIBUTION

The next series of simulation results focus on simulations involving tracer particles having a size (mass) distribution defined by $r = \mu_r \pm \delta r$. In Figure 3.11, the results for simulations involving particles having a distribution given by $r = 0.6 \pm 0.25$ µm and a mass density of $\rho = 3800$ kg/m^3 (alumina) are shown.

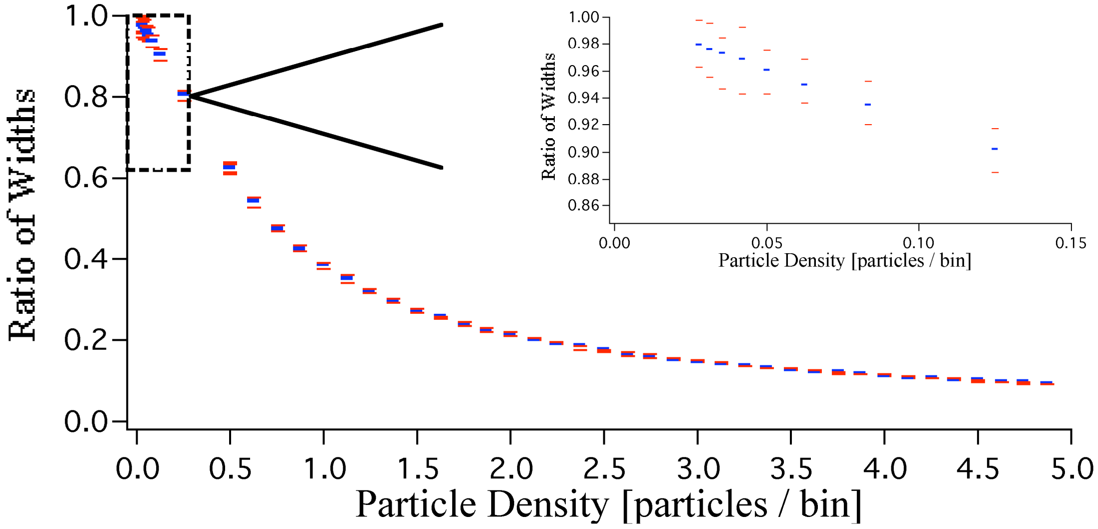


Figure 3.11: Plot of the ratio of the width of the output to the input velocity distribution, σ_{out}/σ_{in}, as a function of the effective particle density for tracer particles having a size distribution. Here, the results for alumina particles (r = 0.6 ± 0.25 μm, ρ = 3800 kg/m^3) are shown. The blue symbol represents the mean value of σ_{out}/σ_{in} for 24 simulations over a wide range of kinetic temperatures, while the error bars represent the extreme values observed for the ratio σ_{out}/σ_{in}.

As before, a series of simulations are performed over a wide range of particle densities and temperatures. In this case, 24 simulations were run in which the temperature of the input distribution is varied between 1 eV < T < 50 eV. The blue symbol represents the mean value for σ_{out}/σ_{in} and the error bars in red represent the maximum and minimum values for σ_{out}/σ_{in} for each effective particle density. It is again observed that the kinetic temperature of the particles has no significant impact on the width of the output velocity distribution at any given density. Additionally, the density of particles continues to have a strong influence on the simulation results.

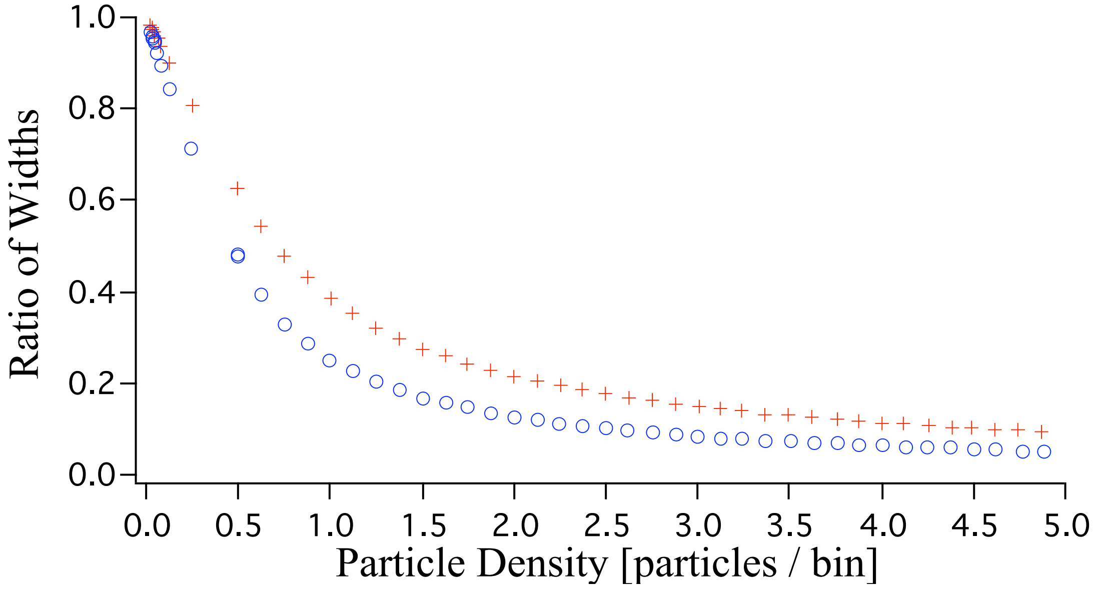


Figure 3.12: Plot of the ratio of the width of the output to the input velocity distribution, σ_{out}/σ_{in}, as a function of the effective particle density for tracer particles having a size distribution. Here, the results from a simulation involving alumina particles having a size distribution ($r = 0.6 \pm 0.25$ µm, ρ = 3800 kg/m^3), depicted by crosses, are compared with the results from a simulation involving monodisperse alumina particles ($r = 0.6$ µm, ρ =3800 kg/m^3), depicted by circles.

Figure 3.12 compares the results of simulations for particles having a mass distribution ($r = 0.6 \pm 0.25$ µm, ρ = 3800 kg/m^3), depicted by crosses, and a uniform mass ($r = 0.6$ µm, ρ = 3800 kg/m^3), depicted by circles, at a fixed dust temperature of $T_{dust} = 15$ eV. While it is observed that the effective particle density has a significant impact on the width of the output velocity distribution at any given density, the effect is notably stronger when monodisperse particles are used. These simulation results suggest that the mass distribution broadens the width of the velocity distribution that is measured using the PIV technique, which is consistent with what was observed in Figure 3.8.

3.3.2.2 EFFECT OF WIDTH OF PARTICLE SIZE DISTRIBUTION

The next set of simulations seeks to characterize the influence of the width of the size distribution on the width of the output velocity distribution. That is, if the particle size distribution is specified by $r = \mu_r \pm \delta r$, does the size of the quantity $\delta r/\mu_r$ influence the width of the output velocity distribution? To examine this, three cases are considered for a fixed temperature: $\delta r/\mu_r = 0.25$, 0.5 and 0.75 for a range of mean particle sizes ranging from 1.5 μm to 50 μm. For reference, the simulation results presented in Figure 3.11 had $\delta r/\mu_r \approx 0.42$.

The results for these simulations are seen in Figure 3.13, where (a) shows the results for $\delta r/\mu_r = 0.25$, (b) shows the results for $\delta r/\mu_r = 0.5$ and (c) shows the results for $\delta r/\mu_r = 0.75$. In all three plots, the crosses represent the simulations where $\mu_r = 1.5$ μm, the triangles represent the simulations where $\mu_r = 3.0$ μm and the dots represent the simulations where $\mu_r = 50$ μm. There are three observations of note here. First, as has been the case in all other simulations, the density of particles has a strong influence on the width of the velocity distribution that is measured using the PIV technique. Second, as the ratio, $\delta r/\mu_r$, increases, the effect that the PIV measurement has on the width of the

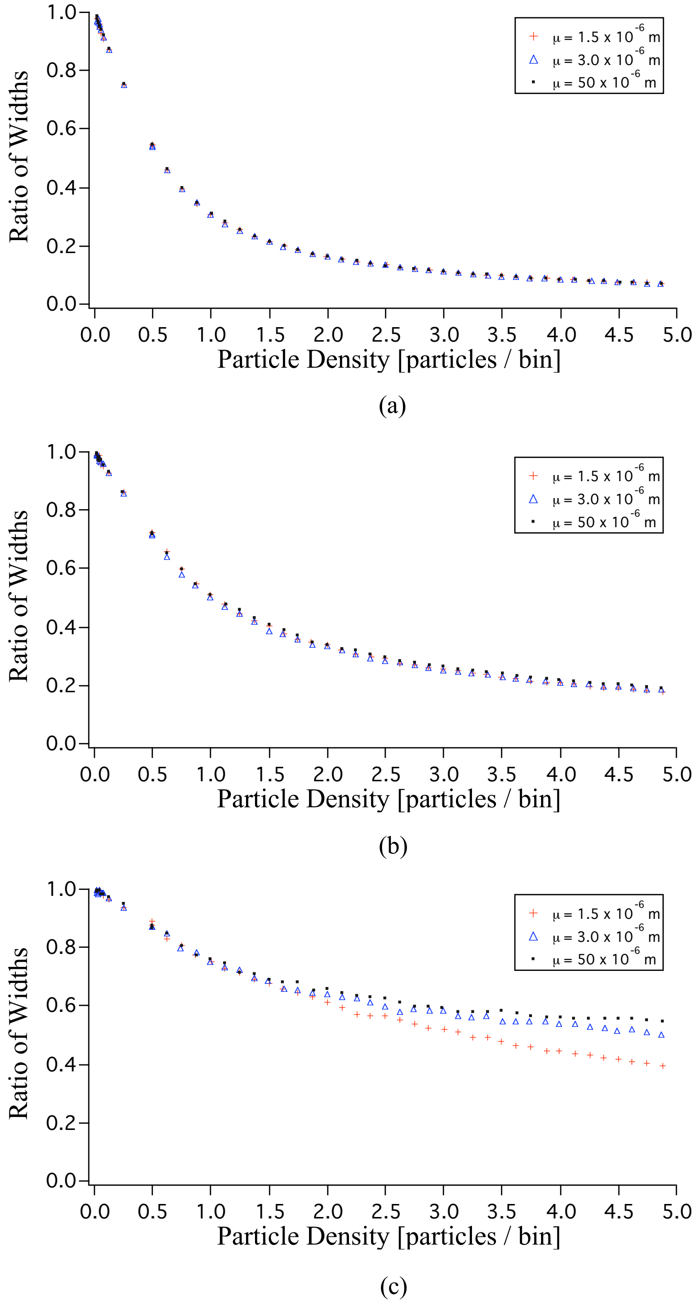


Figure 3.13: Plot of the ratio of the width of the output to the input velocity distribution, σ_{out}/σ_{in}, as a function of particle density for tracer particles having a size distribution with a varying width, δr, relative to the mean radius, μ_r,. In all three plots, the crosses depict a size distribution with μ_r, = 1.5 µm, the triangles depict a size distribution with μ_r, = 3.0 µm and the dots depict a size distribution having μ_r, = 50 µm. In (a) the width of the size distribution relative to the mean value, $\delta r/\mu_r$, of 0.25, in (b) $\delta r/\mu_r$ = 0.5 and is (c) $\delta r/\mu_r$ = 0.75.

output velocity distribution becomes less severe. This is consistent with previous observations that the size distribution broadens the output velocity distribution. Further, it suggests that this broadening of the output velocity distribution increases as the width of the size distribution ($\delta r/\mu_r$) increases. Finally, there is an interesting result that is particularly clear in Figure 3.13. For a fixed particle density, the mean size of the particles, μ_r, increases, the effect that the PIV measurement has on the width of the output velocity distribution appears to approach a constant value.

In an attempt to characterize this result, the ratio of the output to input velocity distribution widths, σ_{out}/σ_{in}, is plotted as a function of the mean particle size, μ_r, for three different effective particle densities with $\delta r/\mu_r = 0.75$ in Figure 3.14.

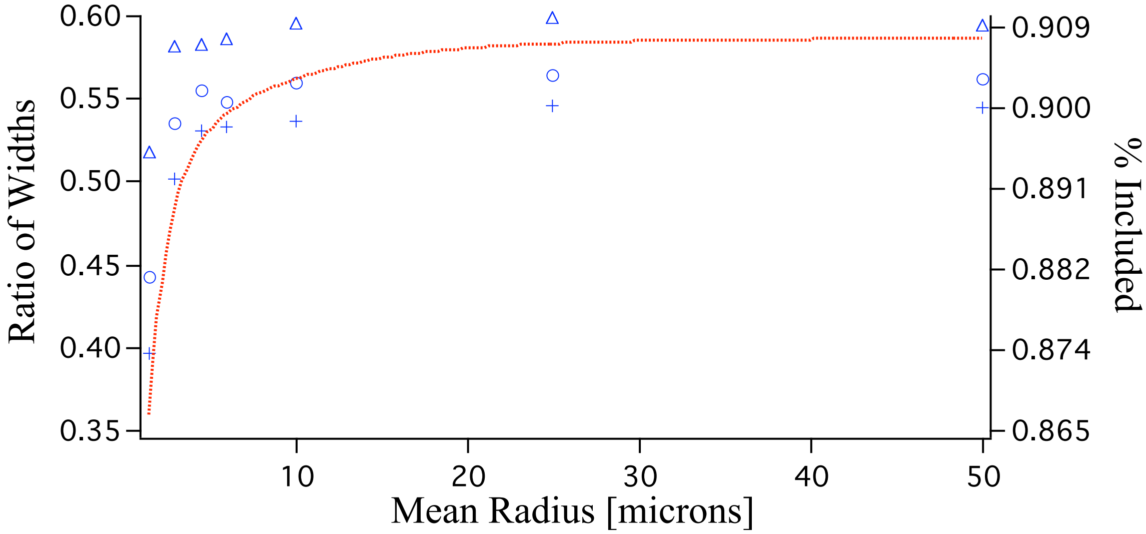


Figure 3.14: Plot of the ratio of the width of the output to the input velocity distribution as a function of mean particle size for simulations with $\delta r/\mu_r = 0.75$. Here, crosses represent an effective particle density, $n_{d,effective}$, of approximately 4.9, circles represent $n_{d,effective} \approx 3$ and triangles represent $n_{d,effective} \approx 2$. The dashed curve depicts the portion of the distribution included due to the truncation of the size distribution (*e.g.* $r \geq 0.25$ μm).

Here, the crosses represent an effective particle density, $n_{d,effective}$, of approximately 4.9 particles per bin, the circles represent an effective particle density of approximately 3 particles per bin and the triangles represent an effective particle density of approximately 2 particles per bin. It is observed that as the mean particle size increases, the ratio of widths has an asymptotic behavior. This is a common for each of the particle densities, suggesting that it is intrinsic to the PIV measurement and not an effect of the particle density. A similar result is observed when $\delta r/\mu_r = 0.5$. However, in this case, the trend much less pronounced. This trend was not seen in any of the other the previously discussed simulations. Consequently, it appears that if the quantity $\delta r/\mu_r$ is roughly 0.5 or larger, this effect is observed.

This effect is a result of the fraction of the size distribution that is included in the PIV simulation. As previously noted, the size distribution of the tracer particles is truncated such that particles having a radius less than 0.25 μm are not included in the simulation. This truncation in the size distribution has an indirect effect on the values of velocity that can be included in the simulation. The velocities are selected from a Maxwellian velocity distribution whose width is proportional to the temperature of the particles and inversely proportional to the mass of the tracer particles. As $\delta r/\mu_r$ increases, the fraction of the size distribution that is lost by this truncation increases as the mean value, μ_r, decreases. This is seen in the dashed curve in Figure 3.14, where the percent of the distribution that is included for $\delta r/\mu_r = 0.75$ is plotted as a function of the mean particle size.

By not including the smaller sized particles at a fixed temperature, the larger values of velocity will not be included in the simulation. Consequently, the velocity distribution that the PIV measurement acts on will be narrower and the averaging that takes place in the PIV measurement will not be as severe. As a result, the narrowing effect that the PIV measurement has on the width of the output velocity distribution, for fixed values of $\delta r/\mu_r$, would be smaller as the mean size of the particle distribution decreases.

In the majority of the simulations that have been presented, the width of the size distribution has been relatively small compared to the mean value and this truncation of the size distribution has had little effect. However, as this truncation becomes more severe, the fraction of the particle size distribution that is included in the simulation impacts the shape of the mapping function that relates the underlying velocity distribution function to what is measured by the PIV technique.

3.3.2.3 WEIGHTED PIV CALCULATION

In the previous two sections, an underlying assumption in the simulations presented was that the particles all contribute equally in the PIV measurement. However, under the assumption of Rayleigh scattering, the particles will scatter light that is proportional to r_d^2. This will lead to an r_d^4 weighting in the cross correlation analysis that generates the PIV velocities. To account for this, the computation of the average velocity of the particles in an interrogation region was modified to Equation 3.5.

$$v_j = \frac{1}{N_j} \sum_i \frac{r_i^4 v_i}{\mu_r^2} \tag{3.5}$$

When one accounts for the weighting that occurs due to the amount of light that is scattered by the microparticles (i.e. the microparticles size), one observes a notably different looking distribution of velocities, as seen in Figure 3.15.

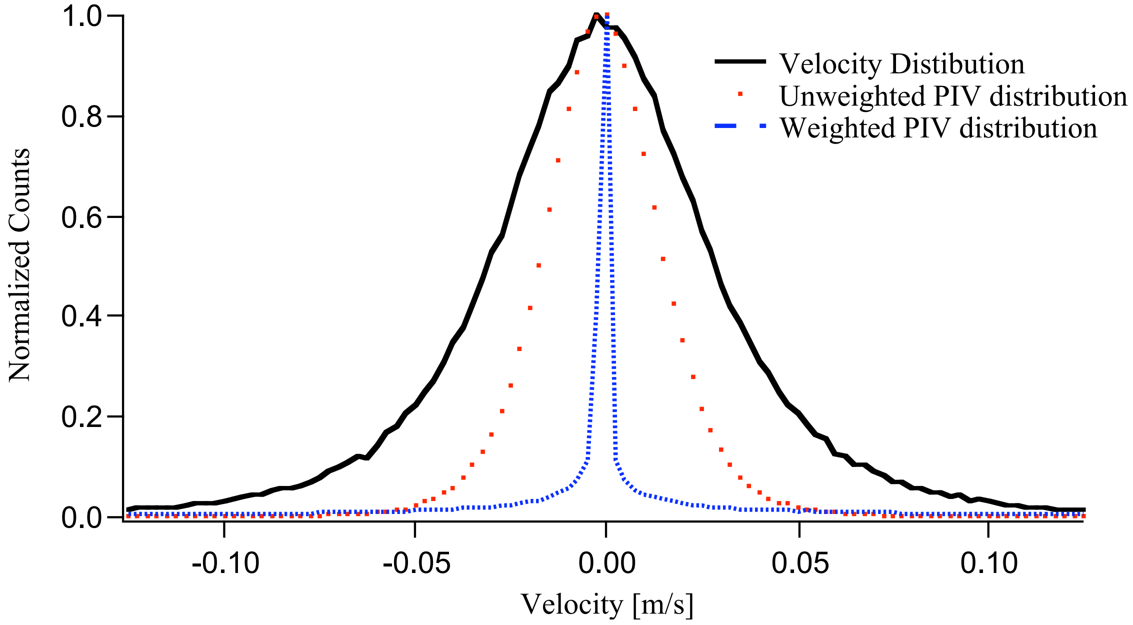


Figure 3.15: Simulation results showing the effect on the measured distribution when on includes the weighting due to particle size in the PIV technique. The solid black curve depicts the underlying velocity distribution, the dotted red curve depicts what is measured by the PIV technique when all particle contribute equally and the dashed blue curve depicts what is measured by the PIV technique when the particle size is a factor in the cross correlation analysis. The strong peak at zero velocity that occurs when including the weighting due to particle size is not observed experimentally.

Here, the underlying velocity distribution is depicted by the solid black curve, what is measured assuming a uniform contribution of particles sizes (*i.e.* Equation 3.4) is depicted by the red dotted curve and what is measured assuming a that the particle size is a factor in the cross correlation analysis is depicted by the dashed blue curve (*i.e.* Equation 3.5). While it is observed that the expected narrowing in the measured distribution occurs, the narrowing is much more severe. In particular, there is a very strong peak at zero velocity. This peak was characteristic in all of the simulations performed using Equation 3.5. Further, it is noted that the distributions are no longer Maxwellian and, in this case, there is no longer a mapping function that relates the measured distribution to the underlying velocity space distribution function.

It is noted, however, that this strong peak at zero velocity was not observed experimentally. As a result, detailed simulations of this phenomenon were not performed. It is noted however, that this result is significant in that it suggest many of the dust clouds studied in this work, even for cases where the original powder was polydisperse, led to particle suspensions that were fairly monodisperse. If this were not the case, the measured distributions would have a strong peak at zero velocity. Additionally, the light intensity recorded by the PIV camera remains generally constant over the particles, supporting the assertion that the suspended particles may be nearly monodisperse.

3.4 DISCUSSION OF SIMULATION RESULTS

Simulations of the PIV measurement have shown that there is a unique mapping function that relates the widths of the input and output velocity distribution provided that the PIV technique is applied to dust particles that are essentially monodisperse. In order generate a mapping function that will connect the distribution of velocities that is measured using the PIV technique to the underlying velocity space distribution, what remains is to relate the effective particle density that is used in the simulations and the particle density that appears in the experiment, *i.e.* it is necessary to identify a relationship between the units of particle density used in the simulation (particles/bin) and what can be measured in an experiment (*e.g.* particles/cm^3). In the simulations, a 2 × 2 bin region corresponds to the volume of the evaluation region in a PIV experiment. The volume of an evaluation region is given by $\gamma^2 n^2 t$, where n is half the side of the interrogation region, t is the thickness of the laser sheet and γ is a conversion factor from pixels to meters. The value of γ will depend on the calibration of the camera system used in making the PIV measurement and n will depend on the size of the interrogation region used in the PIV analysis (*i.e.* for a 12 × 12 pixel interrogation region, $n = 6$). For one such calibration on the 3DPX device, $n = 6$ pixels, $t = 1$ mm and $\gamma \approx 43$ μm/pixels. Using these values, the calibration connecting the effective particle density used in the simulations and the particle density that would be measured in an experiment is seen in Figure 3.16.

Typically, the density of the tracer particles is not uniform over the whole field that is measured using a PIV system. This is certainly the case in studies of the weakly-coupled dusty plasmas that occur in a dc discharge plasma, as can be seen by inspection of the inverted image of a dust cloud in Figure 3.17. It is noted that simulations where the particle density was uniform over the simulation grid yielded the same results for a wide variety of functional forms for the density, provided that the effective particle density was the same. Consequently, the particle density of interest would be the particle density over the volume that the PIV measurement is made.

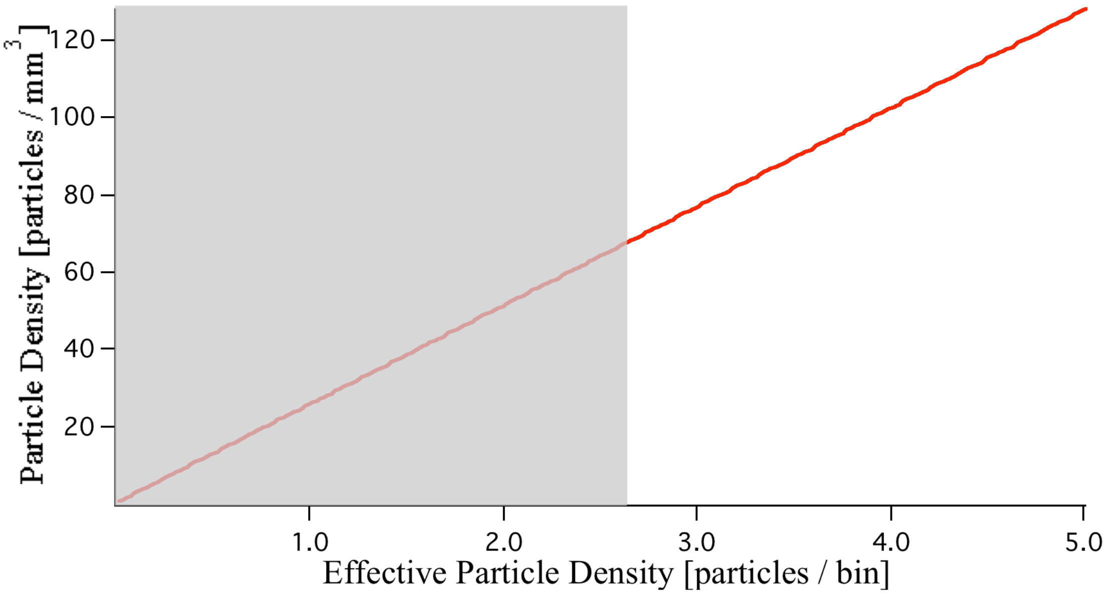


Figure 3.16: A curve relating the effective particle density that is used in the simulations to particle densities that would be measured in an experiment. It is important to note that the slope of this curve will depend on the particular calibration of the PIV system.

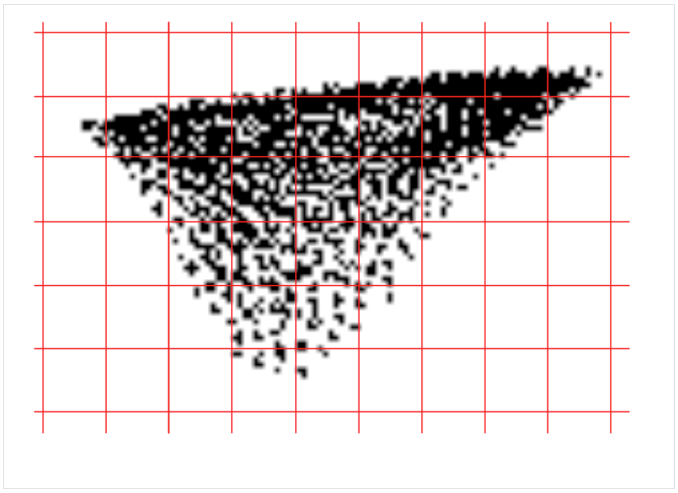


Figure 3.17: An inverted image of a typical dust cloud. Superimposed on this image is a grid that represents the interrogation regions that are used in the PIV calculation. It is noted that the particle density is not uniform of the interrogation region, which corresponds to a 2 × 2 box bin (12 × 12 pixels).

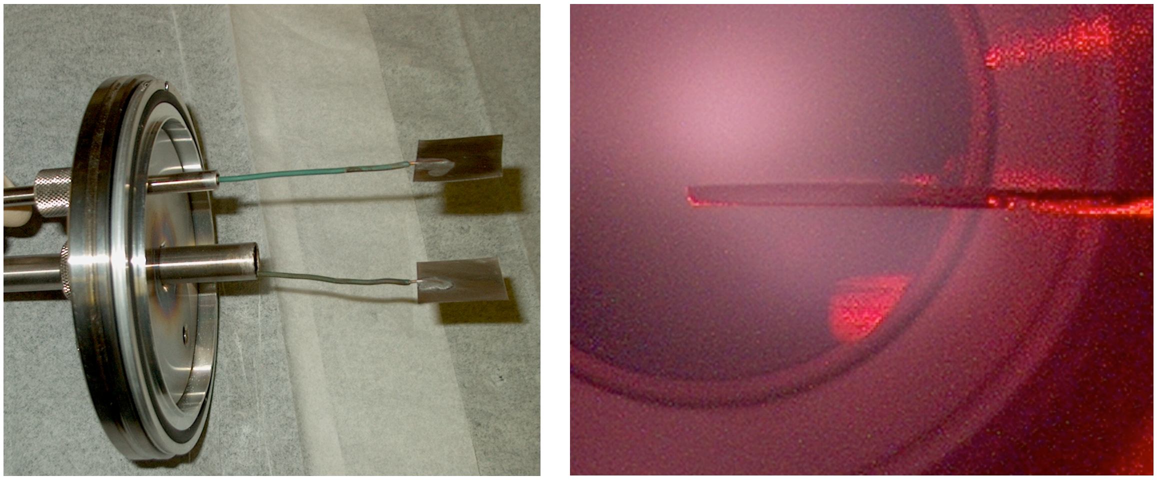
CHAPTER 4: EXPERIMENTAL RESULTS

In this chapter, the first extensive measurements of the thermal properties of a weakly-coupled dusty plasma in a dc glow discharge are presented. In particular, the velocity space distribution function of the microparticle component is measured and from this, a kinetic temperature is extracted. The experimental methodology is described in Section 4.1. Results for microparticles having a polydisperse distribution are presented in Section 4.2 and results obtained using monodisperse microparticles are presented in Section 4.3. Section 4.4 contains a discussion of these results.

4.1 EXPERIMENTAL METHODOLOGY

The primary diagnostic technique used in the experiments presented in this chapter is stereoscopic particle image velocimetry (stereo-PIV). In these experiments, a thin slice (~1 mm) of the dusty plasma is illuminated by a pair of laser pulses, shaped into "two-dimensional" sheets using a cylindrical lens. The time between the laser pulses for the experiments described in this chapter, unless otherwise noted, is held fixed at 250 μs to facilitate measurement of the kinetic motion of the microparticles. The electrodes used to generate the plasma are seen in Figure 4. 1.

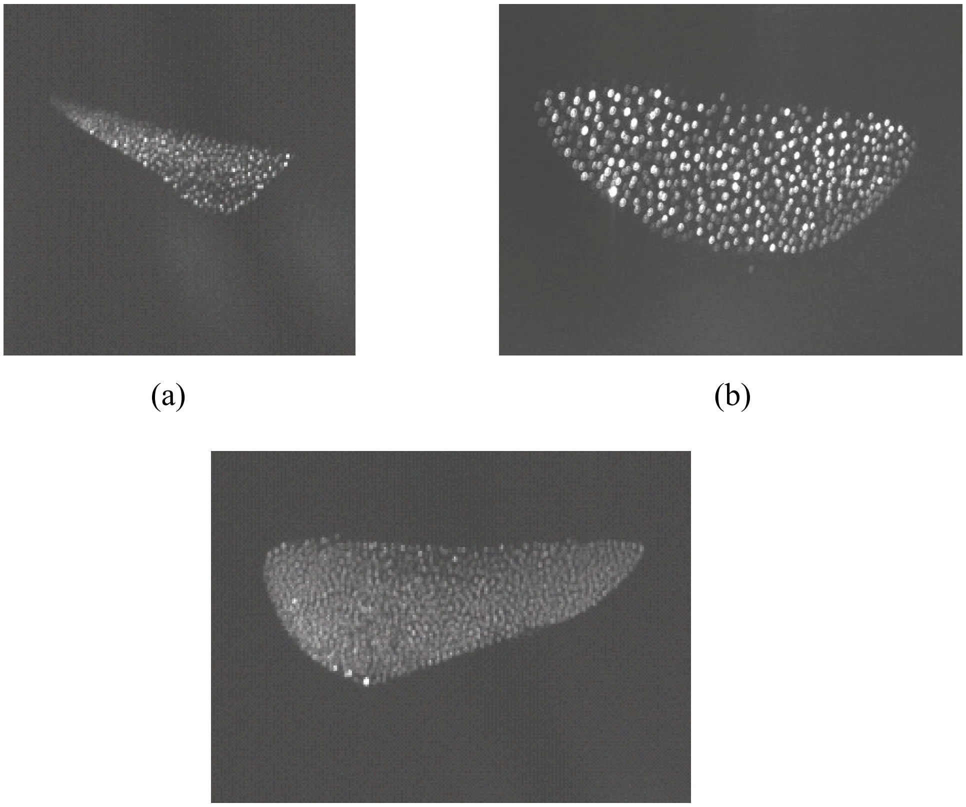
There were two electrode configurations that were used in the experiments in this chapter: the cathode-anode-tray and the anode-cathode-tray configuration. The bulk of the measurements were made using the cathode-anode-tray configuration. In this configuration, the upper electrode was biased negative relative to the grounded chamber wall (cathode) and the lower electrode was biased positive with respect to the chamber wall (anode). In the anode-cathode-tray configuration, the electrode biases are switched. In both configurations, the dust tray was located below both of the electrodes and was electrically floating.



(a) (b)

Figure 4.1: A photograph of (a) the electrodes used in the experiments described in this dissertation and (b) the location of the dust cloud when the electrodes are arranged in the cathode-anode-tray configuration. In (b), the lower electrode (anode) is visible. A dust cloud composed of 1.51. μm diameter silica microspheres is illuminated by a laser diode (λ = 632 nm) and is visible in red. The purplish background is an argon plasma.

As noted in Section 1.3, previous studies of the thermal properties of dusty plasmas were restricted to strongly-coupled systems, mostly in rf discharge systems. In those experiments, the microparticles were organized into “rigid” uniform structures with a fixed inter-particle spacing that is comparable to the electron Debye length, ~100 – 300 μm. Consequently, for most strongly-coupled experiments, tracking of individual particles is sufficient to determine the particle velocities. By contrast, the dust clouds in these dc glow discharge experiments, as seen in Figure 4.2, contain no obvious spatial ordering. Further, the interparticle spacing can vary across the cloud from tens of microns (~λ_{Di}) to a few hundred microns (~λ_{De}).



(a) (b)

(c)

Figure 4.2: A typical video image of a weakly-coupled dusty plasma for (a) 1.2 ± 0.5 μm diameter alumina microparticles, (b) 6.22 μm diameter melamine microspheres and (c) 3.02 μm diameter silica microspheres.

Using the stereo-PIV system, measurements of the three-dimensional velocity vectors of the microparticles are made over the entire volume of the suspended particle cloud. This is accomplished by stepping the laser sheet through the entire cloud volume in 1 mm increments in the *z*-direction, the direction perpendicular to the plane of the laser sheet. At each *z*-location, a sequence of *n*, typically 60, PIV measurements are made. Since the effective PIV measurement rate is 5 Hz, the recording of these 60 images requires ~12 seconds. However, after moving the laser/camera assembly to a new *z*-location, the camera oscillates for a short period of time, ~0.8 to 1.0 seconds as seen in Figure 4.3.

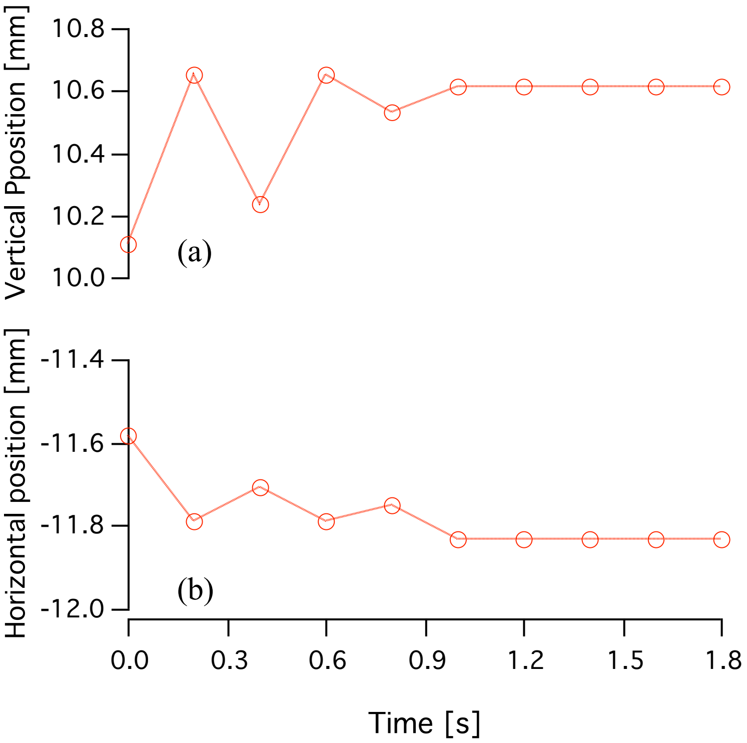


Figure 4.3: A plot depicting the oscillation of the PIV camera when the linear translation stage comes to rest. Here, the (a) vertical and (b) horizontal motion of a fixed location inside of the 3DPX device is tracked over the first ten images in a measurement sequence.

To account for the oscillation of the cameras that is observed, the first fifteen images (~3 seconds) of each measurement sequence are excluded from the analysis. All of the subsequent velocities that were measured over the entire cloud volume are combined to obtain an experimentally measured distribution of velocities for each velocity component (v_x, v_y and v_z) for the entire cloud.

In the experiments presented in this chapter, a dc glow argon discharge plasma is generated in the 3DPX device [78] using a pair of biased electrodes. For these experiments, the cathode-anode-tray configuration was used. Once the plasma has been established, dust particles are introduced and a dust cloud forms in the anode sheath below the lower electrode (anode), as seen in Figure 4.1(b).

The experiments described in the following sections were performed over a range of pressures. It is noted that while these experiments were performed over a wide range of experimental conditions, care was taken to ensure that there was no visible global or collective transport of the microparticles in the dust cloud. Specifically, there was no observed change in the phase of the dust cloud, as was observed in the aforementioned rf experiments. Further, as the pressure was varied, the anode and cathode voltages were adjusted to maintain stable dust clouds with no observable collective effects (i.e. waves, oscillations or other large scale transport structures). Once a stable dust cloud has been formed, a distribution of velocities is measured and the pressure is varied. Once a new set of experimental conditions has been established, the system is not disturbed for a period of time (~3 – 5 minutes) before taking the next measurement.

The dust clouds that were studied in this experiment are a closed system in the sense that there are no dust particles entering or leaving the dust cloud over the duration of the experiment. At the same time, these systems are open in the sense that energy flows freely between the background plasma and the suspended dust cloud. As a result, the dust clouds studied here represent a non-equilibrium thermodynamic system where the dust clouds are in local equilibrium with the surrounding plasma. Consequently, one can use some ideas from equilibrium thermodynamics to study these systems.

To interpret the experimentally measured velocity distributions, there is one assumption that is used in the analysis for both the polydisperse and monodisperse microparticles: it is assumed that the microparticles have reached a local equilibrium with the background plasma. As a result of this assumption, it is assumed that the velocity distribution for each vector directions can be modeled as a drifting 1D Maxwellian, as given in Equation 4.1.

$$f = \sqrt{\frac{m}{2\pi k_B T_d}} \exp\left\{ -\frac{m(v - v_d)^2}{2k_B T_d} \right\} \tag{4.1}$$

where v is the velocity, m is the mass of the dust particle, k_B is Boltzmann's constant, T_d is the dust temperature and v_d is the drift velocity. There are three points that bear mentioning. First, the temperatures that are measured here represent the bulk temperature of the entire cloud. Second, it is noted that a drifting Maxwellian was used for completeness and there was no observed drift in the data that is presented in this chapter. Finally, from this fit it is possible to extract the underlying distribution function after accounting for the error in the PIV system and the averaging that is intrinsic to the PIV measurement.

Once the velocity distribution has been measured, the first step is to account for the uncertainty in the stereo-PIV system. For clarity, the uncertainty here refers to the resolution limit of the stereo-PIV system that was discussed in Section 2.2.4.3, *i.e.* the error that is due to hardware (*e.g.* lens error) and software (e.g. limits of the PIV algorithm) sources. As seen in Section 2.2.4.3, the resolution error in the stereo-PIV system is Maxwellian. Since we are measuring the thermal properties of a dusty plasma by measuring the distribution of particle velocities, which are also Maxwellian in nature, the error seen in Figure 2.39 can be directly accounted for using Equation 4.2.

$$\sigma_{true}^{2} = \sigma_{measured}^{2} - \sigma_{resolution}^{2} \tag{4.2}$$

where $\sigma_{measured}$ is the dispersion that is measured, $\sigma_{resolution}$ is the dispersion that comes from the error measurement discussed in Section 2.2.4.3 and σ_{true} is the actual dispersion.

To account for the averaging that is intrinsic to the PIV technique, as described in Chapter 3, the density of the microparticle component is determined using the technique described in Section 2.3. With this value, the mapping function developed in Section 3.3.1 is used to extract the underlying velocity space distribution function, from which the thermal properties can be extracted. This analysis process is seen in Figure 4.4.

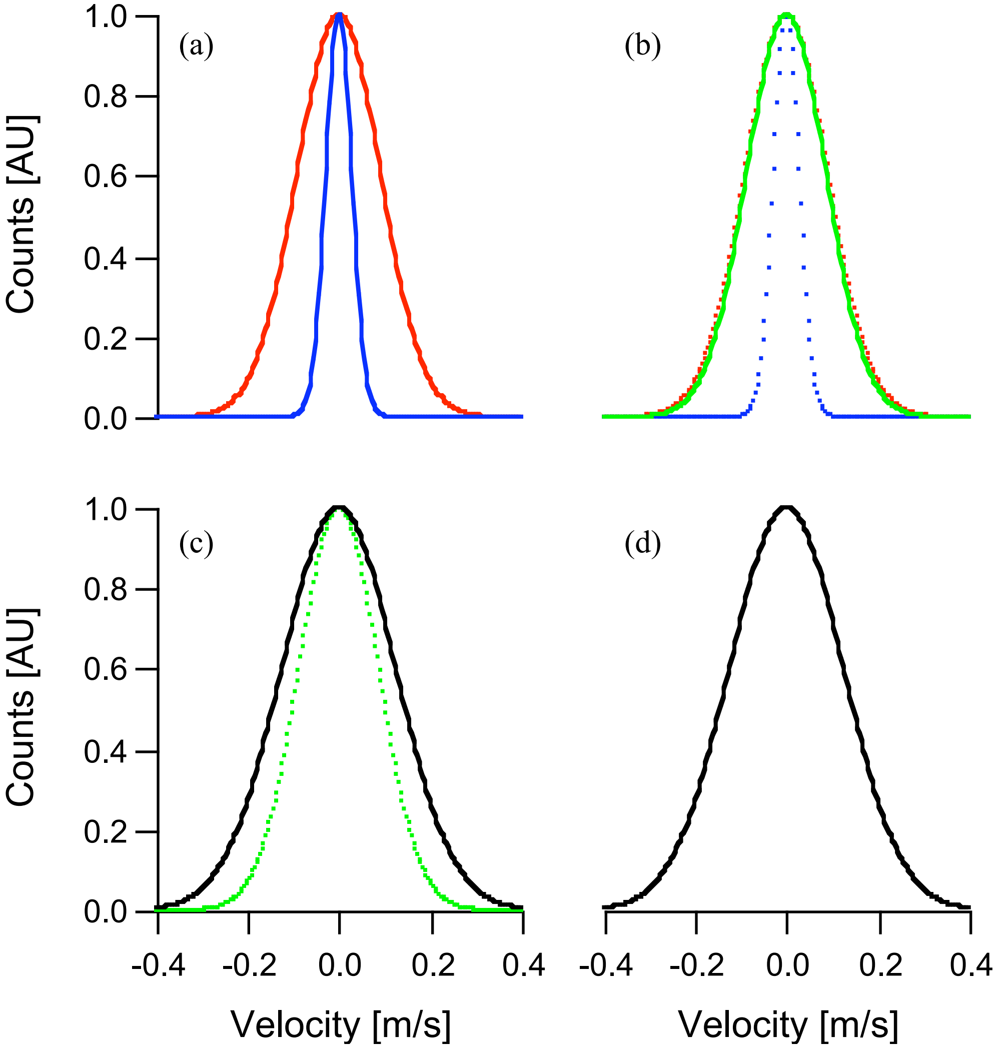


Figure 4.4: Showing the process by which the underlying velocity space distribution function is obtained. In (a), the measured velocity distribution, seen in red, and the resolution error in the stereo-PIV system, shown in blue, are overlaid. In (b), the measured velocity distribution, seen in red, and the resolution error in the stereo-PIV system, shown in blue, are added to yield an intermediate distribution, shown in green. In (c), the mapping function corrects the width of the intermediate function, shown in green, to account for the intrinsic averaging, which yields the velocity space distribution, seen in black. The velocity space distribution function shown in (d), again in black, can then be used to study the thermal properties of the microparticle component of the dust cloud.

At this point, it is important to emphasize that information is not lost due to the time that is required to complete the measurement of a distribution of velocities. When a single measurement is made, the duration of this measurement is 250 μs. At each *z*-

location, 60 measurements are made with a frequency of 5 Hz. For a cloud that is 9 mm thick, the distribution function for the stable dust cloud that is fit to Equation 4.1 includes approximately 405 measurements taken over a period of roughly three minutes. Given this extended time, it is important to verify that information is not lost due to the sampling rate.

To verify that this is not the case, a series of 45 stereo-PIV measurements were made at a single *z*-location. In this experiment, a dust cloud composed of monodisperse (*i.e.* uniform) melamine microspheres having a radius of 3.11 μm was generated in an argon dc glow discharge with p = 160 mTorr, V_{anode} = 208 V and $V_{cathode}$ = -73 V.

The data was then analyzed in the following fashion: the vectors from n measurements were used to generate a distribution of velocities. This was done 60 - n times; *e.g.* for n = 3, the data from measurements 1 - 3, 2 - 4, 3 - 5, . . ., 58 - 60 would be analyzed and a set of temperatures (T_{1-3}, T_{2-4}, T_{3-5}, . . ., T_{58-60}) was extracted. Using this technique, we analyzed this series of data using a variable number of measurements. The results from this analysis is seen in Figure 4.5.

It is noted that the temperatures that are reported here come directly from the fit of the measured distribution. The mapping functions that were discussed in Chapter 3 were not used. However, over the duration of the measurement, the appearance and structure (*e.g.* the size and density) of the dust cloud did not change. Consequently, the PIV correction factor would be the same for all of the data and the net result would be a constant vertical shift in the data shown. Consequently, the essential physics is preserved.

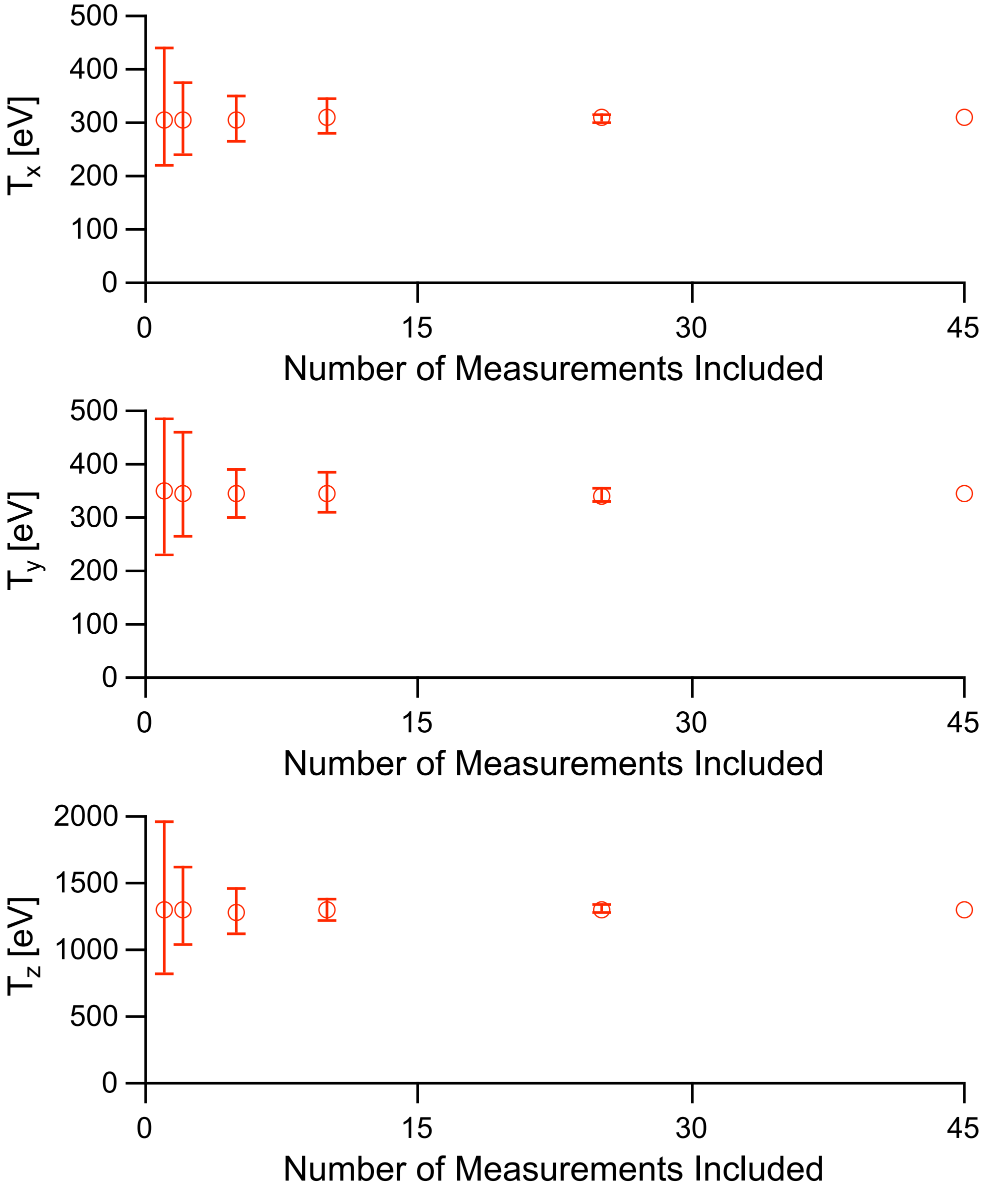


Figure 4.5: Showing the kinetic dust temperature of 6.22 µm diameter melamine microspheres as a function of the number of PIV measurements included in the distribution function. The symbol represents the mean temperature from the 45 - *n* measured temperatures, while the error bars depict the range of temperatures that are measured. The values for n used in this calculation are n = 1, 2, 5, 10, 25 and 45. It is observed that the average temperature is relatively constant. However, the level of fluctuation decreases as more measurements are included in the velocity distribution.

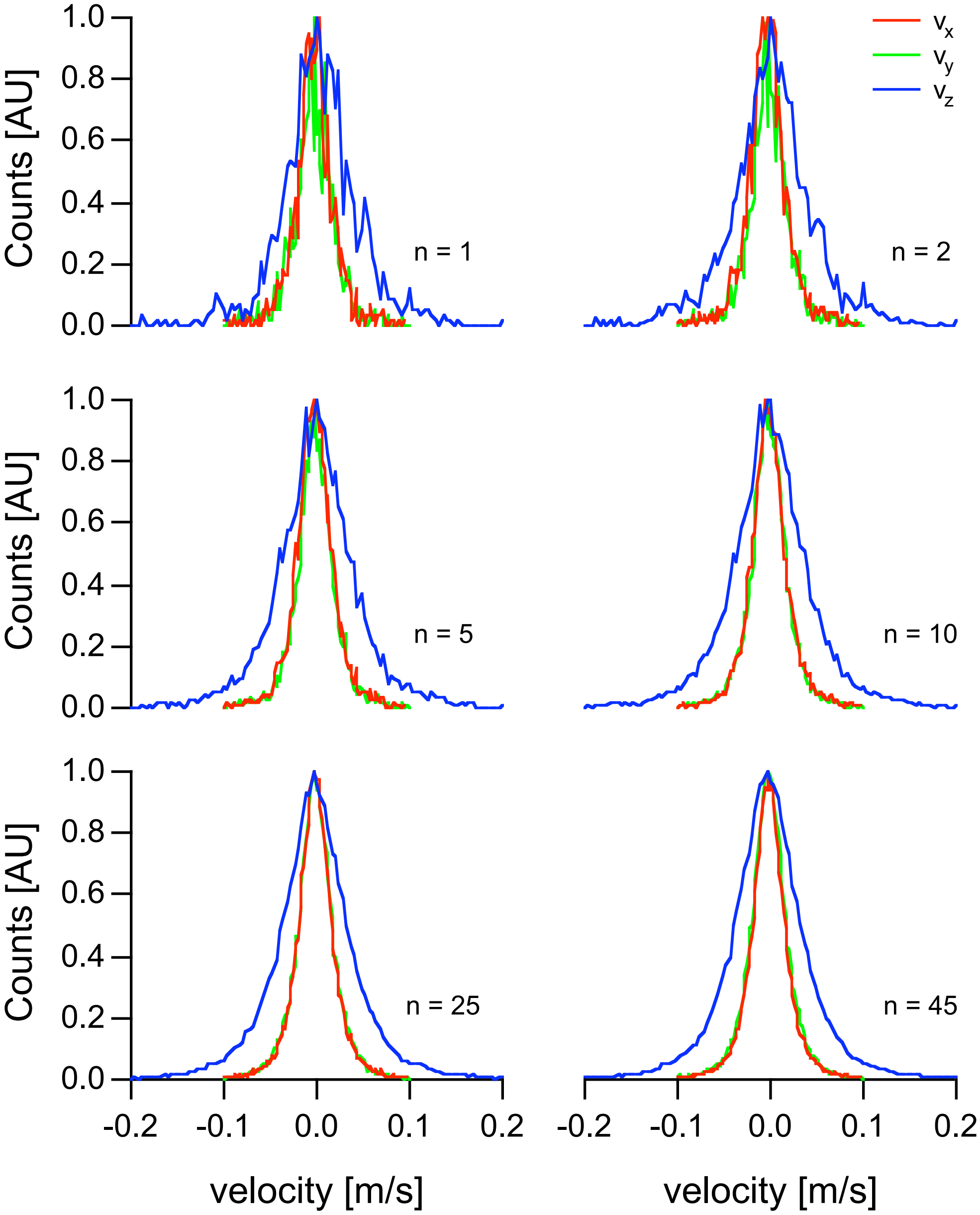
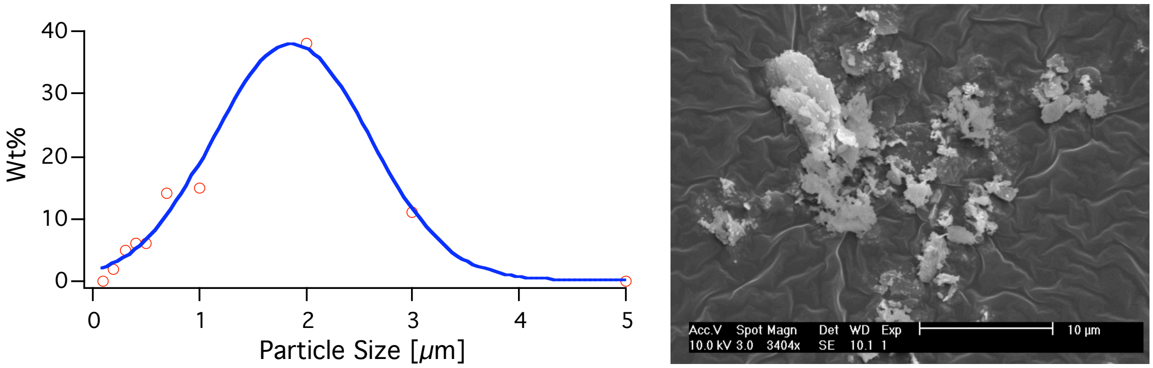


Figure 4.6: Showing the measured velocity distributions for different numbers of included measurements. The red curve depicts the velocity in the *x*-direction, the green curve depicts the velocity in the y-direction and the blue curve depicts the velocity in the *z*-direction. It is observed that there is less fluctuation in the shape of the velocity distribution as the number of measurements included increases.

It is observed that as the number of measurements included in the distribution increased, the level of fluctuation in the measured temperature decreased. Consequently, the use of a large number of images results in better statistics, but not a loss of physical information. This is supported in the observed distribution of velocities, as seen in Figure 4.6. It is noted that the same experimental test was run using the 1.2 ± 0.5 μm diameter alumina microparticles and the same results were observed.

4.2 RESULTS WITH MICROPARTICLES HAVING A POLYDISPERSE SIZE DISTRIBUTION

In the experiments described in this section, dust clouds composed of 1.2 ± 0.5 μm diameter alumina microparticles were suspended in an argon plasma over a range of neutral gas pressures from 100 to 260 mTorr. The size distribution for the dust particles and an SEM image of the dust particles used can be seen in Figure 4.7.



(a) (b)

Figure 4.7: Showing (a) the size distribution and (b) an SEM image of the microparticles used in the experiments presented in Section 4.2. In (a) the red circles depict measured values for the particle size, while the blue curve is a Gaussian fit. (SEM image courtesy of Andrew Post-Zwicker, Princeton Plasma Physics Laboratory)

In order to analyze the measured velocity distributions, there are two additional assumptions: the subset of microparticles that are actually suspended in the plasma are both spherical and monodisperse. Based on what is seen in Figure 4.7, the microparticles are clearly not spherical. However, this assumption is only used to compute a mass using the mean particle size of 0.6 μm ($m_d = 3.44 \times 10^{-15}$ kg). If the second assumption, that the suspended microparticles are monodisperse, is valid, then the mass that is computed using the assumption of spherical particles may be incorrect. However, the key results will be preserved, as the error will result in a shift in the numerical value of the temperatures measured. Consequently, the second assumption, that the suspended particles are monodisperse, is important but its validity is not immediately obvious.

One of the challenges of this experiment is knowing precisely which particles from the source distribution of particle sizes becomes suspended in the plasma. This uncertainty will clearly have an impact on the evaluation of the measured velocity distributions. Nonetheless, there is experimental evidence that the size distribution of the suspended particles is significantly more narrow, possibly even monodisperse, than the source powder. Evidence for this is three-fold. First, the intensity of the scattered light imaged by the CCD cameras does not vary in any significant way across the image of the dust cloud. If the suspended particles had a size distribution, the intensity would fluctuate in a fashion that is not consistent with observations. Further, the fluctuation in the intensity level that was observed for dust clouds whose source powder had a polydisperse size distribution was comparable to what was observed for dust clouds whose source powder were monodisperse microspheres used in the experiments presented in Section 4.3. Second, there are several observations that when particles of different sizes are suspended in a dusty plasma, the particle become segregated according to size, often with sharp boundaries between the different particle sizes [107, 108]. None of these characteristic structures are observed in the dust clouds used in the experiments presented in this section. Finally, the measured velocity distributions do not exhibit a strong peak at zero velocity, which is suggested by the simulation results discussed in Section 3.3.2.3. As a result, there is sufficient evidence to proceed with the assumption that an effectively monodisperse population of microparticles are suspended in the plasma.

Provided that all of the aforementioned conditions are satisfied, it is possible to use the velocity distributions to extract a bulk kinetic temperature for the entire cloud. It is noted that the measured distribution, as seen in Figure 4.8, is in reasonably good agreement with the assumption of a Maxwellian distribution of microparticle velocities, which supports the underlying assumptions that the microparticles have achieved a local thermodynamic equilibrium.

It is noted that the widths of the velocity distributions (*e.g.* temperature) seen in Figure 4.8 are highly anisotropic. As seen in Measurement 2 in Table 1.1 and Measurement 4 in Table 1.3, there was an observed anisotropy in the temperatures measured in the previous experiments. However, unlike many of the rf dusty plasma experiments in which the dust particles remain confined in highly well-ordered 2D structures, the particle clouds in this experiment often have extended 3D structures. Furthermore, as shown in Figure 4.1, the orientation of the electrodes and trays do not allow any of the vector directions to be symmetric to another. In the *y*-direction, gravity

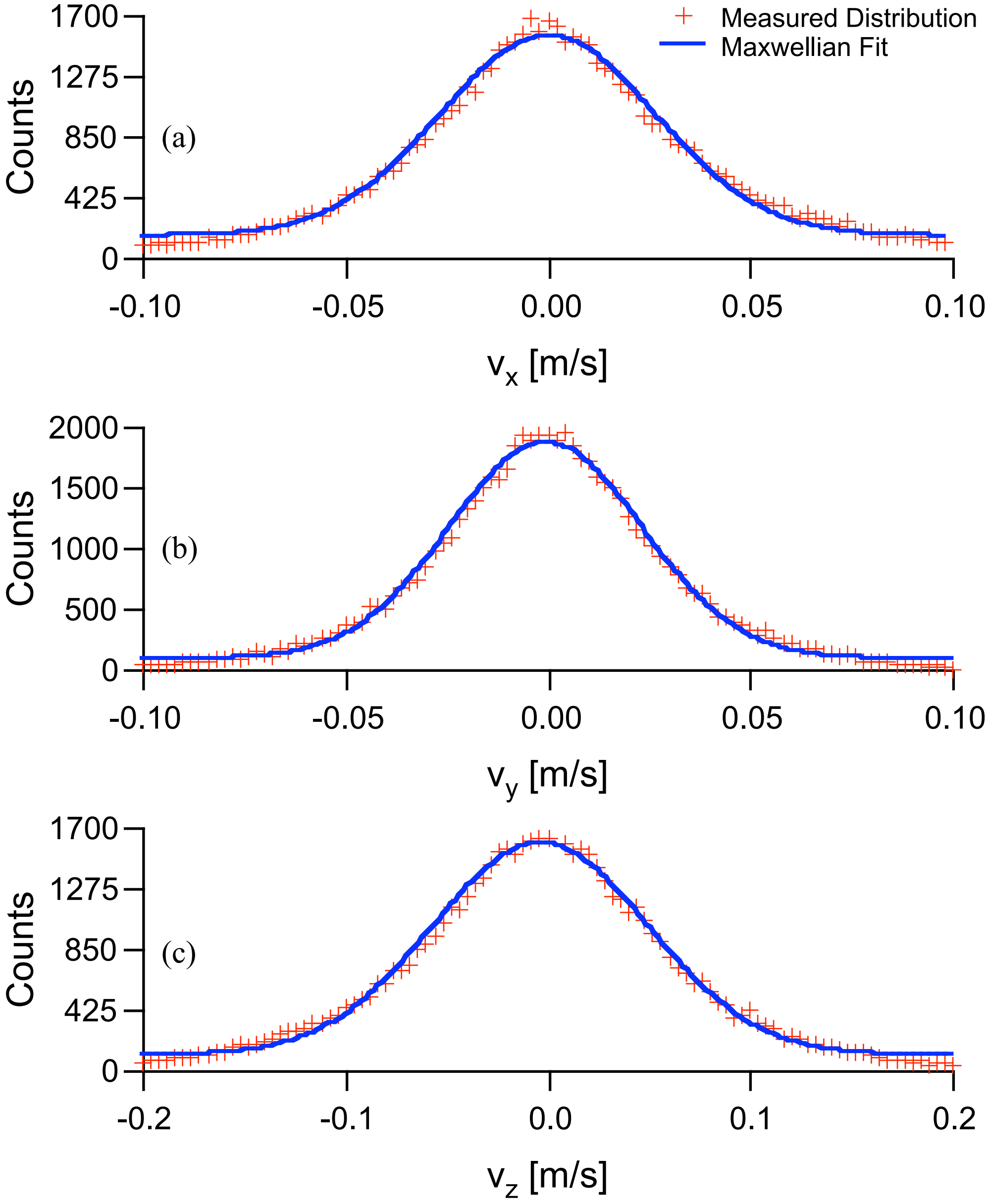


Figure 4.8: Typical velocity distributions in the (a) *x*-, (b) *y*- and (c) *z*-direction. The crosses depict the measured distribution of velocities, while the solid curve represents the fit to a drifting Maxwellian velocity distribution. The experimental parameters for this data are p = 120 mTorr, V_{anode} = 231 V and $V_{cathode}$ = -91 V.

breaks the symmetry with the *x*- and *z*-directions. The symmetry in the *x*- and *z*-direction is broken by the electrodes being fed into the chamber in the *z*-direction and that the electrodes are not located directly above each other. Therefore, some degree of asymmetry may be expected among the distributions of the three velocity components. A full discussion of the interpretation of the temperature asymmetry in the experiment will be given in Section 4.4.

With the confirmation that the stereo-PIV system can measure a velocity distribution, it is now possible to evaluate the thermal properties of the dusty plasma over a variety of plasma conditions. An initial experiment tests the reproducibility of the temperature measurement for a stable cloud in an argon dc glow discharge plasma. Here, six measurements are made of a single dust cloud over a period of an hour. During these measurement, the experimental parameters were held constant with p = 110 mTorr, V_{anode} = 230 V, $V_{cathode}$ = -89 V and there was no observable change in the dust cloud or in the background plasma over the duration of the experiment. In the absence of external perturbations, it is expected that the temperature of the microparticle component should remain constant. The result of this measurement is seen in Figure 4.9. The error bars are due to uncertainty in the number density of the microparticle component, which is then propagated through the mapping function developed in Section 3.3.1. As was noted previously, the temperature is observed to be anisotropic with the largest value in the *z*-direction.

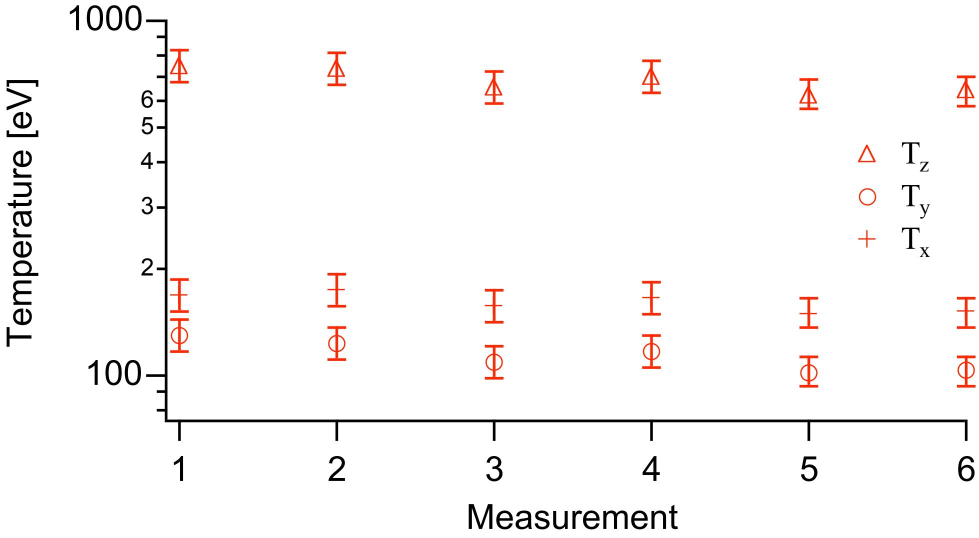


Figure 4.9: Multiple measurements of the kinetic temperature of the microparticle component taken over a period of an hour with constant experimental conditions (p = 110 mTorr, V_{anode} = 230 V, $V_{cathode}$ = -89 V).

It is observed that the measured temperatures are significantly larger than the other plasma components. For the 3DPX device, the electron, ion and neutral temperatures are typically 2-3 eV, ≤ 0.1 eV and 0.037 eV respectively. However, the values of the kinetic dust temperature that are reported in Figure 4.9 are not unexpected. As was seen in Table 1.1, the temperatures that were measured in previous work involving the strongly-coupled dusty plasmas exhibited a temperature in the fluid-like state that was far in excess of the other plasma components. By comparison, the experimental conditions of this study are further into the fluid regime than the previous experiments. This observation, at least qualitatively, is immediately obvious upon examining the images of the dust clouds that are studied in this dissertation. On the other hand, it is much more difficult to quantitatively justify this claim.

Typically, the coupling parameter Γ, defined in Equation 1.8, is used to quantify the state of the system in the strongly-coupled dusty plasmas. Using the parameters for the systems in this dissertation, the values of Γ ranged from $\Gamma \sim 1$ in the x- and y-directions to $\Gamma \sim 0.1$ in the z-direction. However, the definition of Γ, which assumes an isotropic dust temperature and spatially uniform dust densities, is difficult to apply to the thermally anisotropic, spatially extended three-dimensional dust cloud structures that are observed in this experiment. Nonetheless, the estimated values for the coupling parameter place this system clearly in the weakly-coupled regime ($\Gamma \leq 1$), therefore it is not unexpected that this system has a higher temperature than was reported in the earlier studies. Further, of the studies presented in Table 1.1, only measurement 5, was done in a DC glow discharge. The measurements reported in that work represent the maximum energy observed and are in excess of the values reported here.

In a second experiment, the objective is to investigate the scaling of the kinetic dust temperature with changes in the neutral gas pressure. The result of this measurement is seen in Figure 4.10, where the kinetic temperature of three dust cloud are shown as a function of neutral gas pressure. In taking this data, a dust cloud was generated at a low neutral pressure. As the neutral pressure was increased ($100 \leq p \leq 260$ mTorr), the electrode biases were adjusted to maintain a stable dust cloud ($227 \leq V_{anode} \leq 263$ V, $-80 \leq V_{cathode} \leq -93$ V).

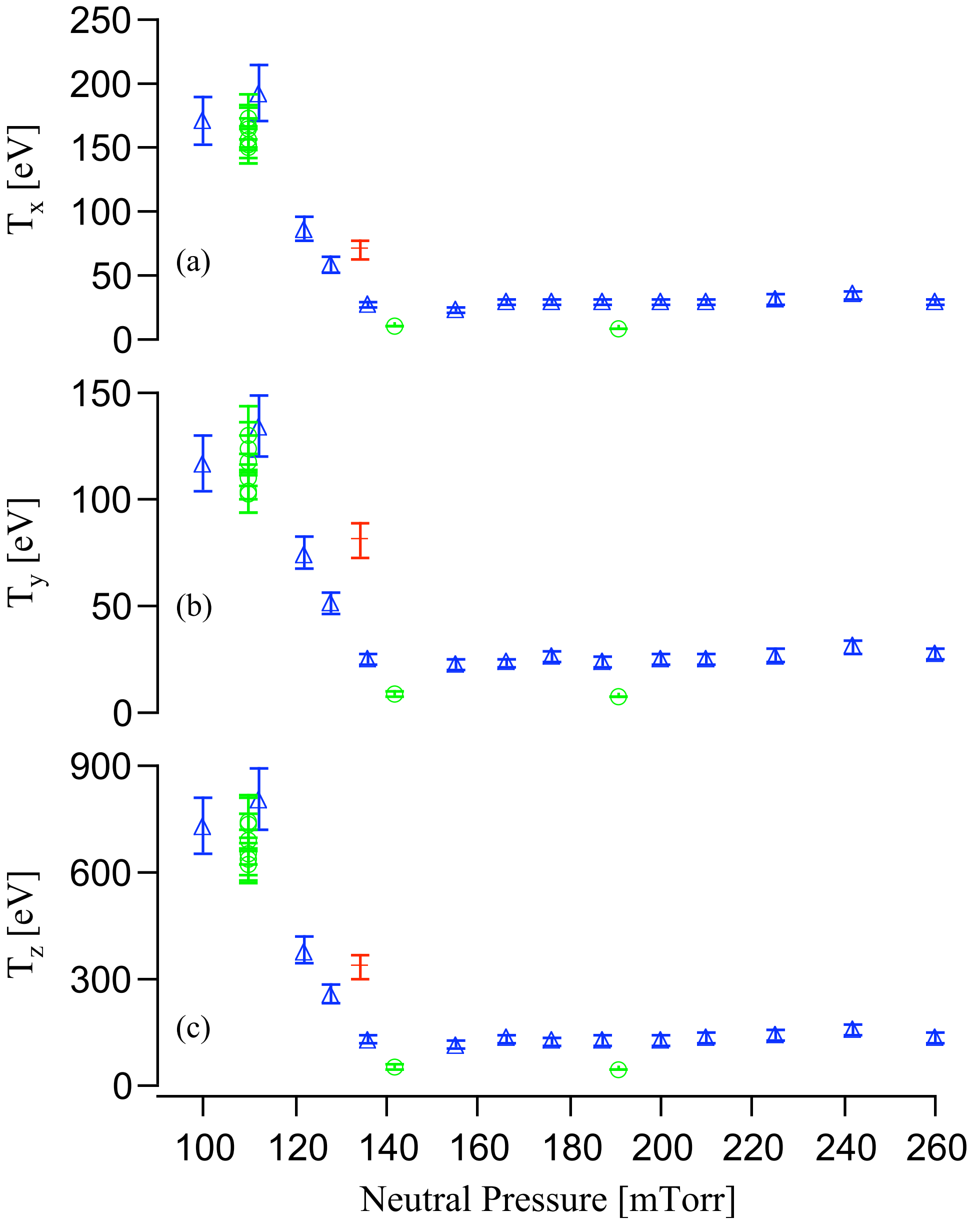


Figure 4.10: Kinetic temperature of the microparticle component in the (a) *x*-, (b) *y*- and (c) *z*-direction as a function of neutral gas pressure. The different colors indicate data from different experimental runs. It is noted that the error bars are statistical and do not account for fluctuations in the background plasma.

First, it is noted that the data for all three dust clouds exhibit the same qualitative behavior as a function of neutral pressure. Initially, at lower neutral pressures, there is a higher temperature in each vector direction. As the neutral pressure increases, the temperature suddenly increased at 110 mTorr. Beyond this pressure, there is a decrease in the kinetic temperature of the dust component between 110 and 140 mTorr. At pressures above $p = 140$ mTorr, the temperature of the dust cloud remains constant and smaller than what was observed at lower neutral pressures.

While the kinetic temperatures reported here are anisotropic and are on very different scales, the temperatures variation is the same for all three vector directions. This is shown in Figure 4.11. Here the measured temperatures have been normalized in each direction using Equation 4.3.

$$T_{normalized,\ i} = \frac{T - T_{\mathrm{min},\ i}}{T_{\mathrm{max},\ i} - T_{\mathrm{min},\ i}} \tag{4.3}$$

where T_i is the temperature being normalized, $T_{min,i}$ and $T_{max,i}$ is the minimum and maximum temperature, respectively, in a given vector direction denoted by the index $i = x$, y, and z. This suggests that the mechanism responsible for heating the microparticle to the observed kinetic temperatures acts in all three components, though not equally, *e.g.* there is preferential heating.

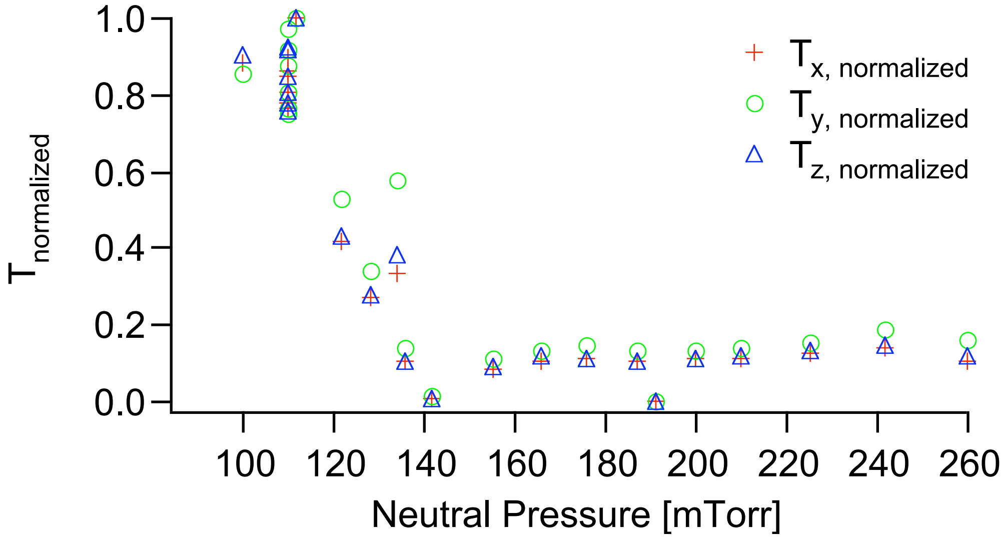


Figure 4.11: A plot of the kinetic temperature for dust clouds composed of polydisperse alumina microparticles (r = 0.6 ± 0.25 μm. $\rho = 3800\ \mathrm{kg/m^3}$), normalized using Equation 4.3, as a function of neutral gas pressure

Two interesting observations are noted during the decrease in temperature between 110 and 140 mTorr. First, there was a change in the structure of the dust cloud that corresponds to the observed change in the temperature. At the lower pressures, the dust clouds are physically larger, have a higher density and are highly disordered. By comparison, all of the dust clouds at a neutral gas pressure greater than 140 mTorr are observed to have the same size and structure, which is notably smaller, significantly lower number density and are more structured than the clouds observed at lower pressures. Despite this change in the physical structure of the cloud, the dust clouds in the experiments presented here remained solidly in a fluid-like regime. Using the definition of $\Gamma = (\Gamma_x, \Gamma_y, \Gamma_z)$ given in Equation 1.8, a value of $\Gamma \sim (0.25, 0.35, 0.05)$ was observed at pressures below 110 mTorr and a value of $\Gamma \sim (1.15, 1.40, 0.25)$ at pressures greater than 140 mTorr. Secondly, there appears to be a correlation between the dust density and the temperature, which is seen in Figure 4.12.

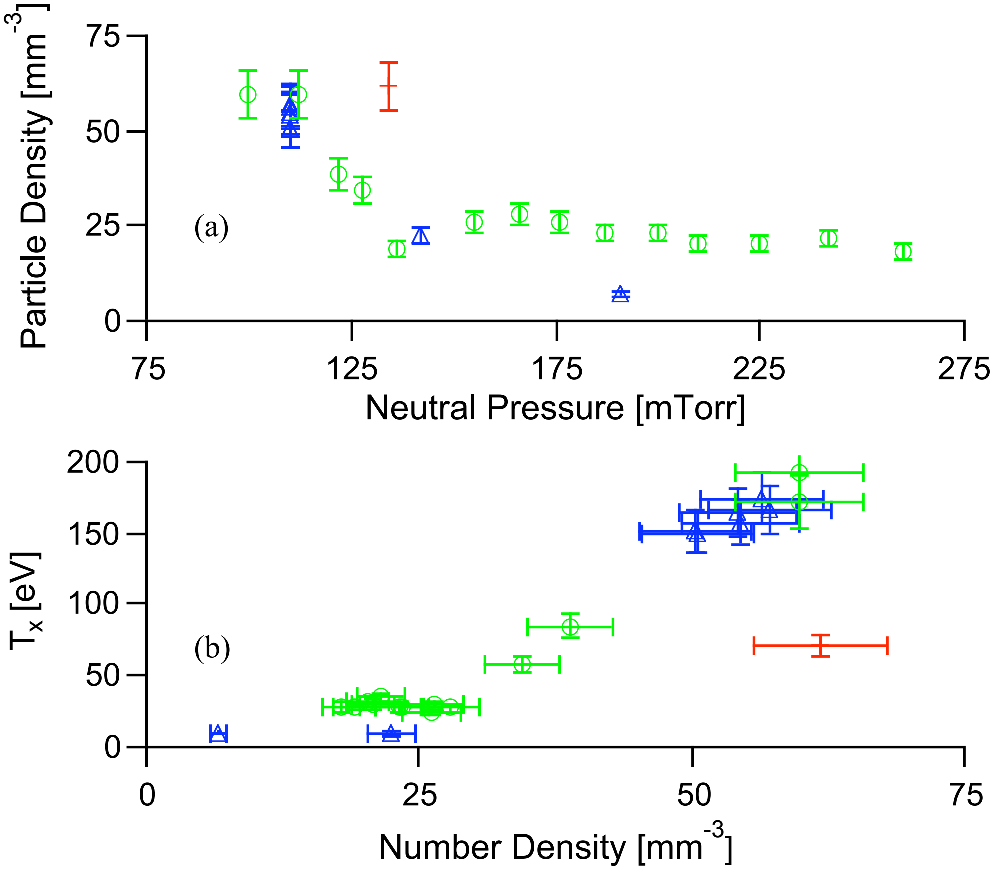


Figure 4.12: A plot of (a) the number density of the microparticle component as a function of the neutral gas pressure and (b) the kinetic temperature of the microparticle component as a function of the number density of the microparticle component. It is noted that the color coding is consistent with Figure 4.10, with each color representing data taken using a single dust cloud.

In Figure 4.12(a), there is a sharp drop in the number density of the microparticle component that is observed as the neutral gas pressure increases. This drop in density corresponds to the observed drop in temperature. This is more clearly seen in Figure 4.12(b), where the kinetic temperature in the *x*-direction is plotted as a function of the number density of the microparticle component. It is noted that the same result is seen in both the *y*- and *z*-direction. Based upon this, it appears that the heating mechanism is more efficient with higher dust density.

4.3 RESULTS WITH MICROPARTICLES HAVING A MONODISPERSE SIZE DISTRIBUTION

The objective of the experiments described in this section is to investigate the scaling of the kinetic dust temperature with changes in the neutral gas pressure for dust clouds composed of monodisperse microspheres. To accomplish this, separate experiments were performed using dust clouds composed of either 6.22 μm diameter melamine (ρ = 1510 kg/m^3) or 3.02 μm diameter silica (ρ = 2000 kg/m^3) microspheres. The particles were suspended in an argon plasma over a range of neutral gas pressures from 98 to 215 mTorr. The measurement and analysis techniques developed for the studies with polydisperse particles are used to analyze these two systems.

In Figure 4.13, the kinetic temperature of two dust clouds composed of the melamine microspheres is shown as a function of the neutral gas pressure. For the data shown in red, a dust cloud was generated at a low pressure, p = 102 mTorr. As the pressure was increased ($102 \leq p \leq 169$ mTorr), the electrode biases were adjusted to maintain a stable dust cloud ($208 \leq V_{anode} \leq 211$ V, $-76 \leq V_{cathode} \leq -73$ V). For the data shown in blue, a dust cloud was generated at a high pressure, p = 172 mTorr. As the pressure was decreased ($172 \geq p \geq 134$ mTorr), the electrode biases were held constant (V_{anode} = 209 V, $V_{cathode}$ = -172 V) maintaining a stable dust cloud.

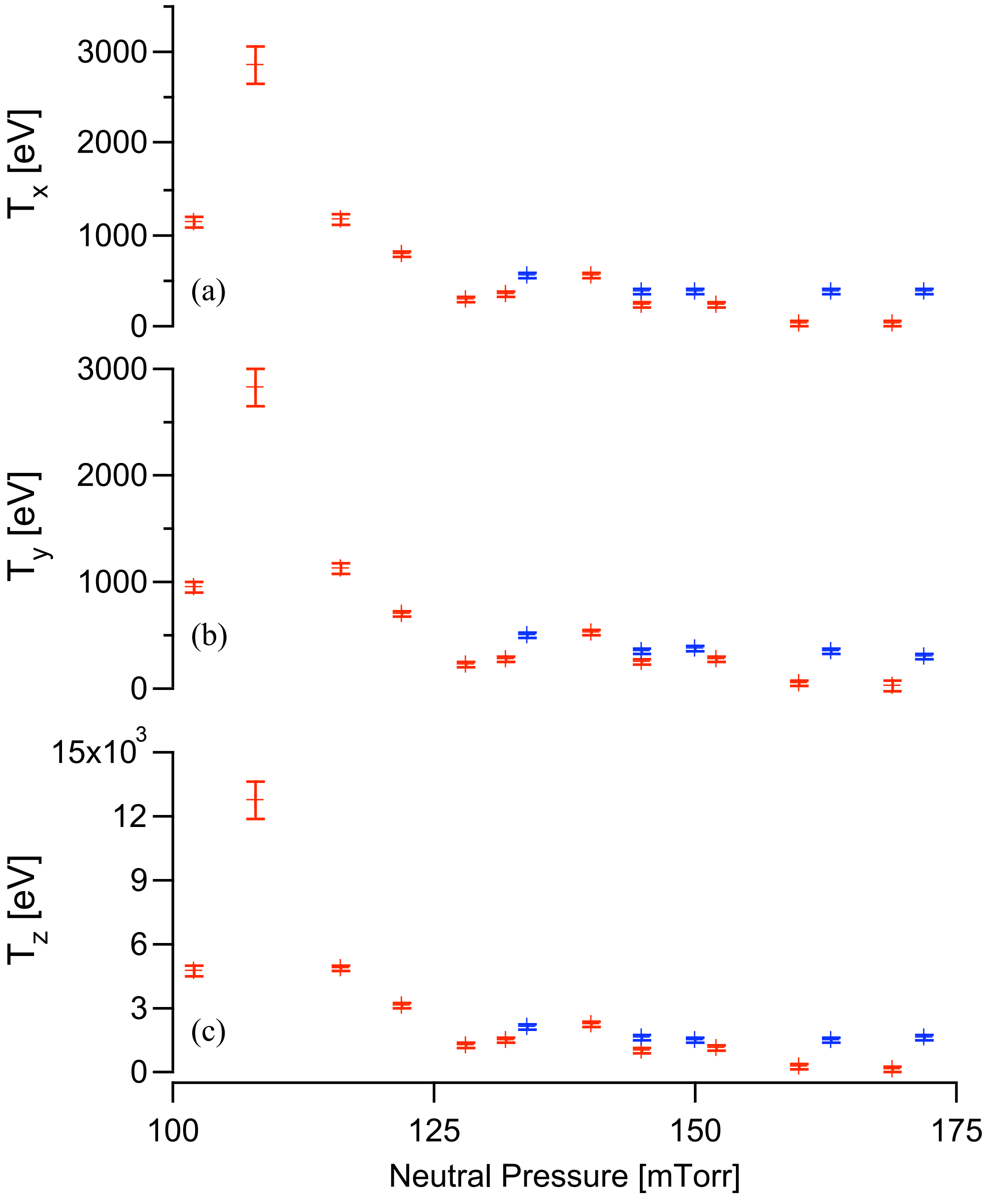


Figure 4.13: Kinetic temperature of the microparticle component in the (a) *x*-, (b) *y*- and (c) *z*-direction as a function of neutral gas pressure. The different colors indicate data from different experimental runs. It is noted that the error bars are statistical and do not account for fluctuations in the background plasma.

First, it is noted that the data for both dust clouds exhibit the same qualitative behavior observed in the dust clouds whose source particles was polydisperse. Initially, at lower neutral gas pressures, there is a higher temperature in each vector direction. Additionally, with the exception of 108 mTorr, the temperature appears to be relatively constant in this region. At 108 mTorr, the measured temperature is significantly larger than the other measurements. As the neutral pressure increases, there is a decrease in the kinetic temperature of the dust component between 110 and 130 mTorr. Beyond $p = 130$ mTorr, the temperature of the dust cloud remains relatively constant.

Further, over all of the pressures examined, the measured temperatures exhibit a strong anisotropy with the temperature being significantly larger in z-direction (transverse to he plane of the laser sheet). To quantify this anisotropy, the ratio of the width of the distribution function in the y-direction to the width of the distribution function in the x-direction, $\sigma_y:\sigma_x$, and the corresponding ratio between the z- and x-direction, $\sigma_z:\sigma_x$, is computed. These ratios provide an effective measure of the asymmetry in the system. In both the alumina and melamine data, this ratio was found to be constant. For the alumina dust clouds, $\sigma_z:\sigma_x = 4.49 \pm 0.05$ and $\sigma_y:\sigma_x = 0.83 \pm 0.02$. For the melamine dust clouds, $\sigma_z:\sigma_x = 4.41 \pm 0.11$ and $\sigma_y:\sigma_x = 0.90 \pm 0.03$. Given the large anisotropy in the z-direction and that this anisotropy appears to be independent of the particle mass or charge, it is important to confirm whether this observation is an experimental artifact caused by the measurement technique or a true measurement of the behavior of the dusty plasma. This is investigated in detail in the next section. However, to determine if it was possible to control this anisotropy, additional experiments were performed on dust clouds compose of 3.02 μm diameter monodisperse silica microspheres. In an attempt to change the observed anisotropy, the orientation of the electrodes was slightly changed. As seen in Figure 4.14, the electrodes are not vertically aligned. There is a small shift in the electrode position in the *z*-direction, by an amount Δ*z*, to facilitates confinement of the dust clouds. For the measurements described below involving the silica microspheres, the shift, Δ*z*, was decreased, relative to the shift that was used in the measurements involving the melamine microspheres and alumina dust.

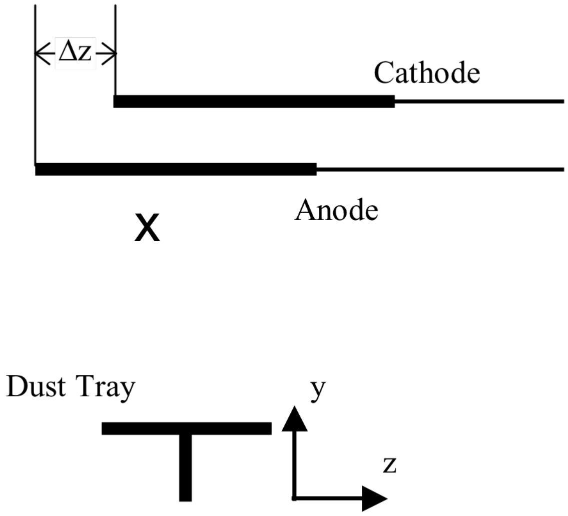


Figure 4.14: Schematic showing a side view of the electrode orientation.

In this new configuration, the temperature of four dust clouds was measured over a wide range of neutral gas pressures. For Cloud 1, a dust cloud was generated at a low pressure, p = 91 mTorr. As the pressure was increased ($91 \leq p \leq 145$ mTorr), the electrode biases were held constant (V_{anode} = 207 V, $V_{cathode}$ = -90 V) maintaining a stable dust cloud. For Cloud 2, a dust cloud was generated at a low pressure, p = 98 mTorr. As the pressure was increased ($98 \leq p \leq 126$ mTorr), the electrode biases were adjusted to maintain a stable dust cloud ($199 \leq V_{anode} \leq 200$ V, $-95.8 \leq V_{cathode} \leq -92.5$ V). For Cloud 3, a dust cloud was generated at a low pressure, $p = 100$ mTorr. As the pressure was increased ($100 \leq p \leq 215$ mTorr), the electrode biases were adjusted to maintain a stable dust cloud ($209 \leq V_{anode} \leq 215$ V, $-101.0 \leq V_{cathode} \leq -89.5$ V). For Cloud 4 , a dust cloud was generated at a high pressure, $p = 189$ mTorr. As the pressure was decreased ($189 \geq p \geq 100$ mTorr), the electrode biases were adjusted to maintain a stable dust cloud ($208 \leq V_{anode} \leq 209$ V, $-89.9 \leq V_{cathode} \leq -76$V).

Of these four clouds, Clouds 3 and 4 were generated in background plasmas that are comparable to what was observed with the melamine and alumina dust clouds. The remaining clouds were less dense. As a result, Clouds 1 and 2 can be quantitatively compared with the other data in a normalized fashion only. In Figure 4.15, the kinetic temperature of Clouds 3 and 4 are shown as a function of pressure.

It is noted that the data for both dust clouds exhibit the same qualitative behavior that was observed using alumina dust and melamine microspheres. Initially, at lower neutral pressures, there is a higher temperature in each vector direction. There is a sharp increase in temperature at neutral gas pressures near 110 mTorr, which is consistent with previous measurements involving dust clouds composed of melamine microspheres. As the neutral gas pressure increases, there is a decrease in the kinetic temperature of the dust component until $p \sim 125$ mTorr. Beyond $p = 125$ mTorr, the temperature of the dust cloud remains constant. It is interesting to note that this sharp increase in temperature appears to be independent on the neutral gas pressure, as it occurs with increasing and decreasing neutral gas pressure.

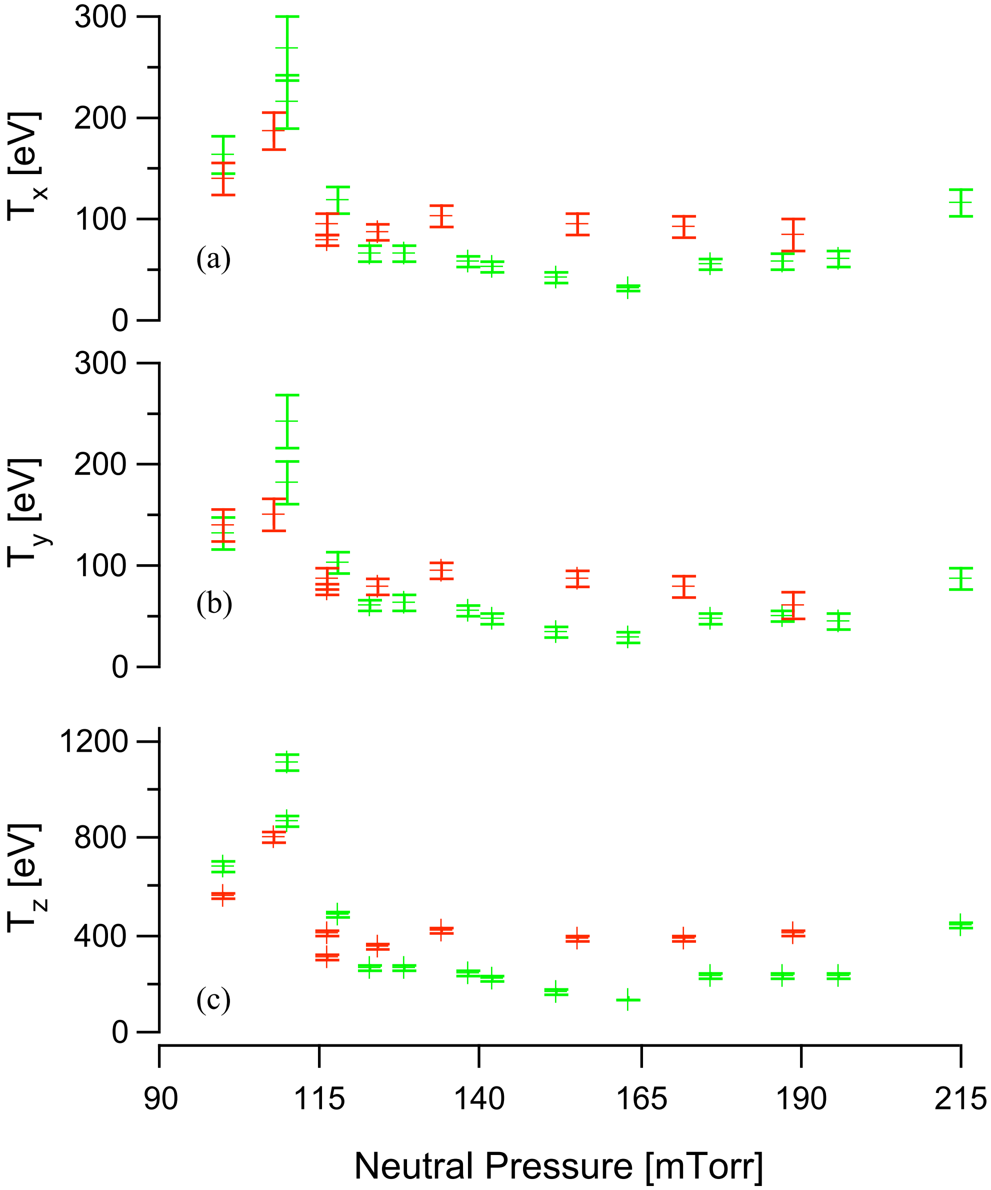


Figure 4.15: Kinetic temperature of the microparticle component in the (a) *x*-, (b) *y*- and (c) *z*-direction as a function of the neutral gas pressure. The data from Cloud 3 is shown in green, while the red data indicates data from Cloud 4.

One of the motivating factors in these experiments with the silica microspheres was examining if it was possible to alter the anisotropy in temperature that was observed. As was previously observed, the anisotropy was constant over all of the data that was taken. By shifting the position of the electrodes in the z-direction, but leaving their orientation otherwise unchanged, it was expected that this would cause a change in the anisotropy in the z-direction. Here, it was observed that the anisotropy in the z-direction is smaller. For these dust clouds, $\sigma_z:\sigma_x = 4.09 \pm 0.04$ and $\sigma_y:\sigma_x = 0.86 \pm 0.01$.

Despite the anisotropic temperatures, the variation in the kinetic temperature for both the melamine and silica dust, is the same in all three vector directions. This is seen in Figure 4.16, where the kinetic temperatures are normalized according to Equation 4.3, which is repeat below.

$$T_{normalized,\ i} = \frac{T - T_{\min,\ i}}{T_{\max,\ i} - T_{\min,\ i}}$$

In Figure 4.16(a), the data for the melamine microspheres is plotted, while the silica microspheres is seen in Figure 4.16(b). For the silica data, the data for Clouds 1-4 are plotted. Again, this is suggestive that the mechanism responsible for heating the microparticle component to the observed kinetic temperatures acts in all three components, though not equally.

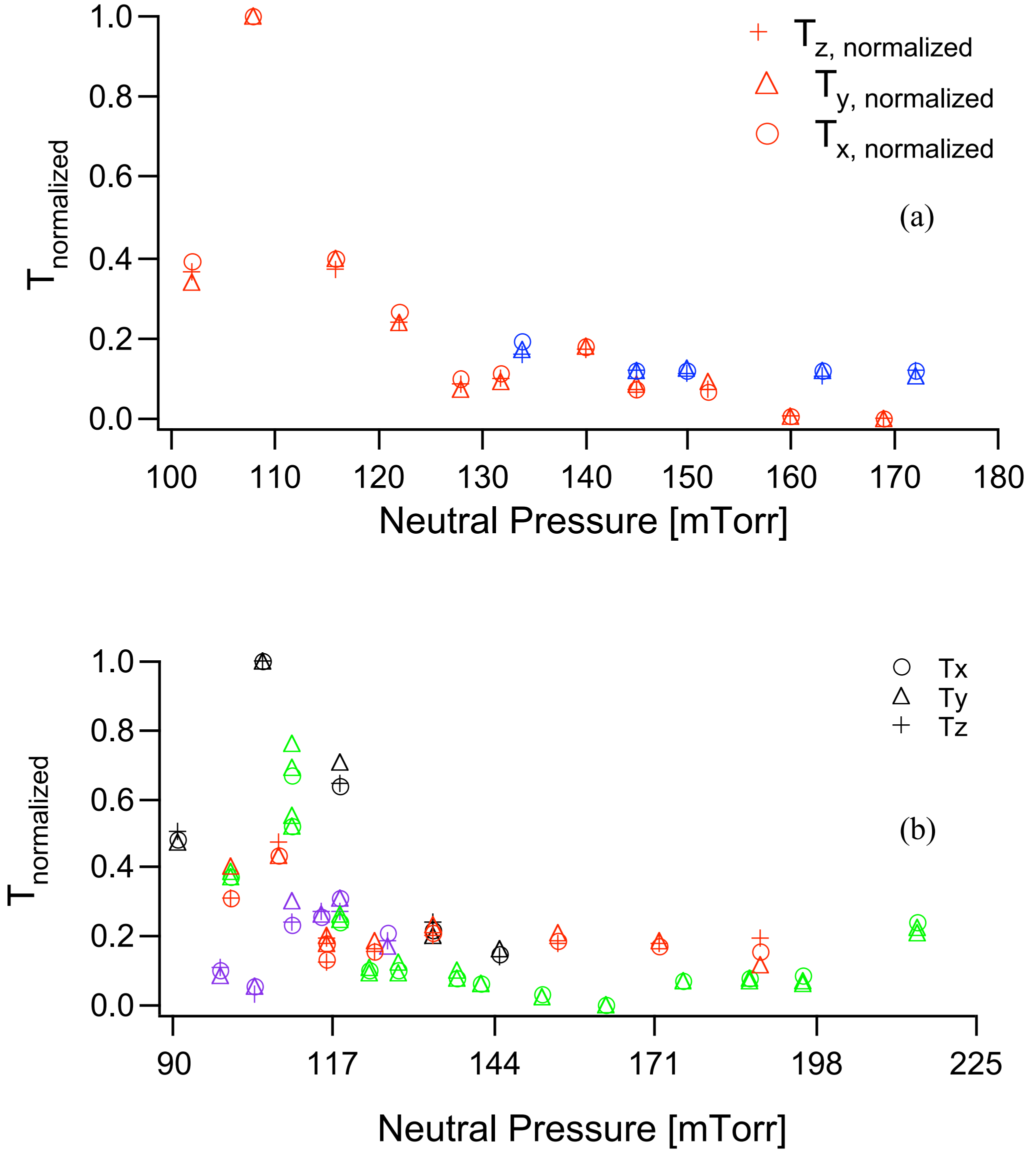


Figure 4.16: Normalized temperature as a function of neutral gas pressure for dust clouds composed of (a) melamine and (b) silica microspheres. In (a), the red data corresponds to the cloud that was generated at low pressure, while the blue data corresponds to the cloud that was generated at high pressure. In (b), the data in black, purple, green and red corresponds to Clouds 1 – 4 respectively.

In both sets of data, a sharp decrease in the temperature is observed. For the melamine microspheres, this decrease occurs between 110 and 130 mTorr. For the silica microspheres, this decrease occurs between 105 and 125 mTorr. In both cases, the observed decrease in temperature is comparable to the range of neutral pressures where a strong decrease in the kinetic temperature was observed for the polydisperse data.

By more closely examining the silica data, it is possible to more clearly see the structure of the sharp increase in temperature that is observed in around p = 110 mTorr. This is seen in Figure 4.17, where the normalized temperature for dust clouds composed of silica microspheres is plotted as a function of pressure over the range of pressures where the kinetic temperature is observed to suddenly increase. In order to better see the structure, the kinetic temperature was normalized according to Equation 4. 4.

$$T_{normalized,\ i} = \frac{T_i}{T_{\max,\ i}} \tag{4.4}$$

where T_i is the temperature being normalized, $T_{max,i}$ is the maximum temperature for a given cloud in a given vector direction denoted by the index $i = x, y$, and z. While the pressure at which this increase in temperature occurs varies in the four cases seen here, all of the data exhibits the same type of structure with neutral gas pressure.

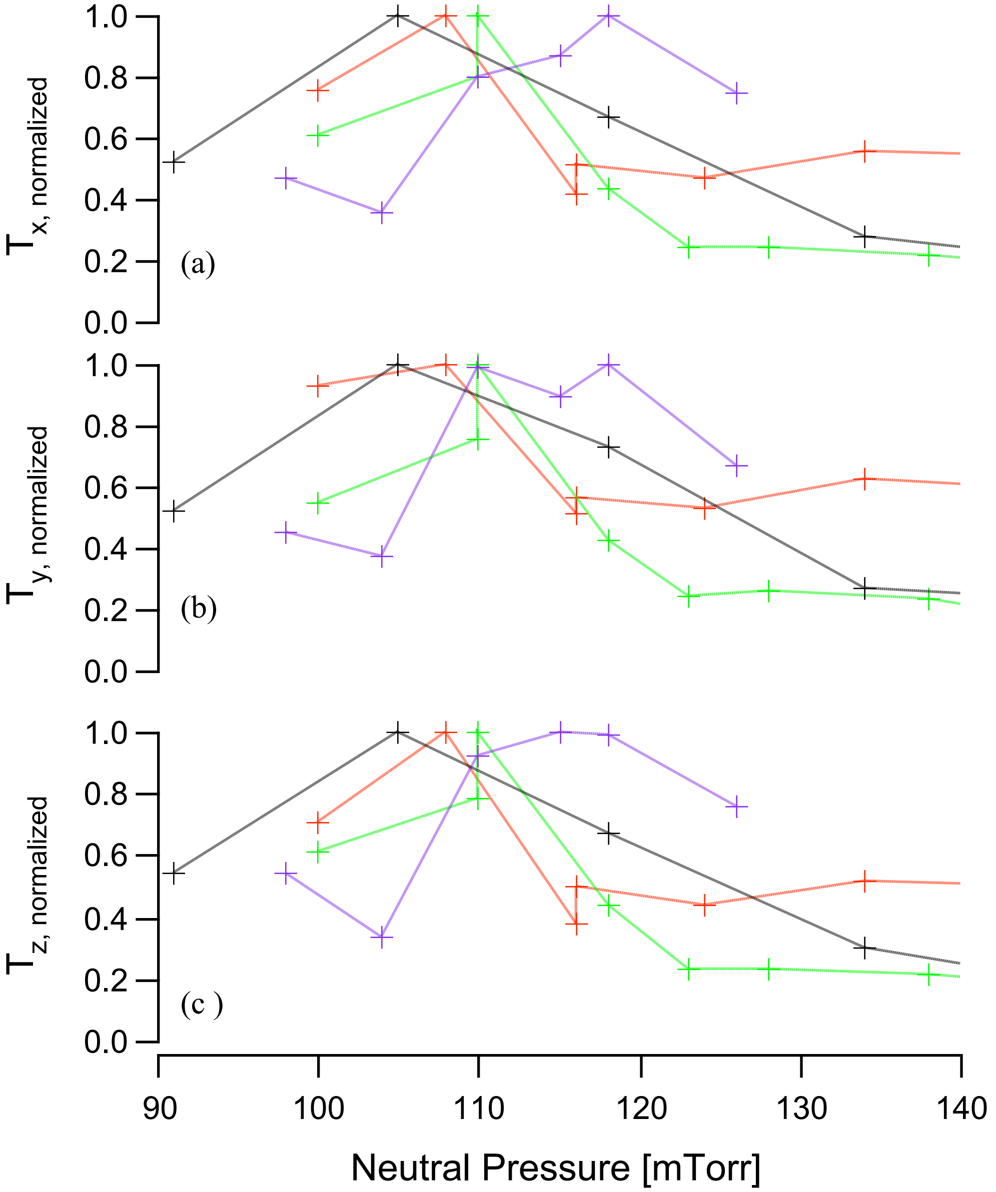


Figure 4.17: Normalized temperature in the (a) *x*- (b) *y*- and (c) *z*-direction as a function of the neutral gas pressure for the dust clouds composed of silica microspheres. The data in black, purple, green and red corresponds to Clouds 1 – 4 respectively.

Additionally, the dependence of the kinetic temperature on the number density of the microparticle component and the kinetic temperature are seen in Figure 4.18 for (a) melamine and (b) silica dust.

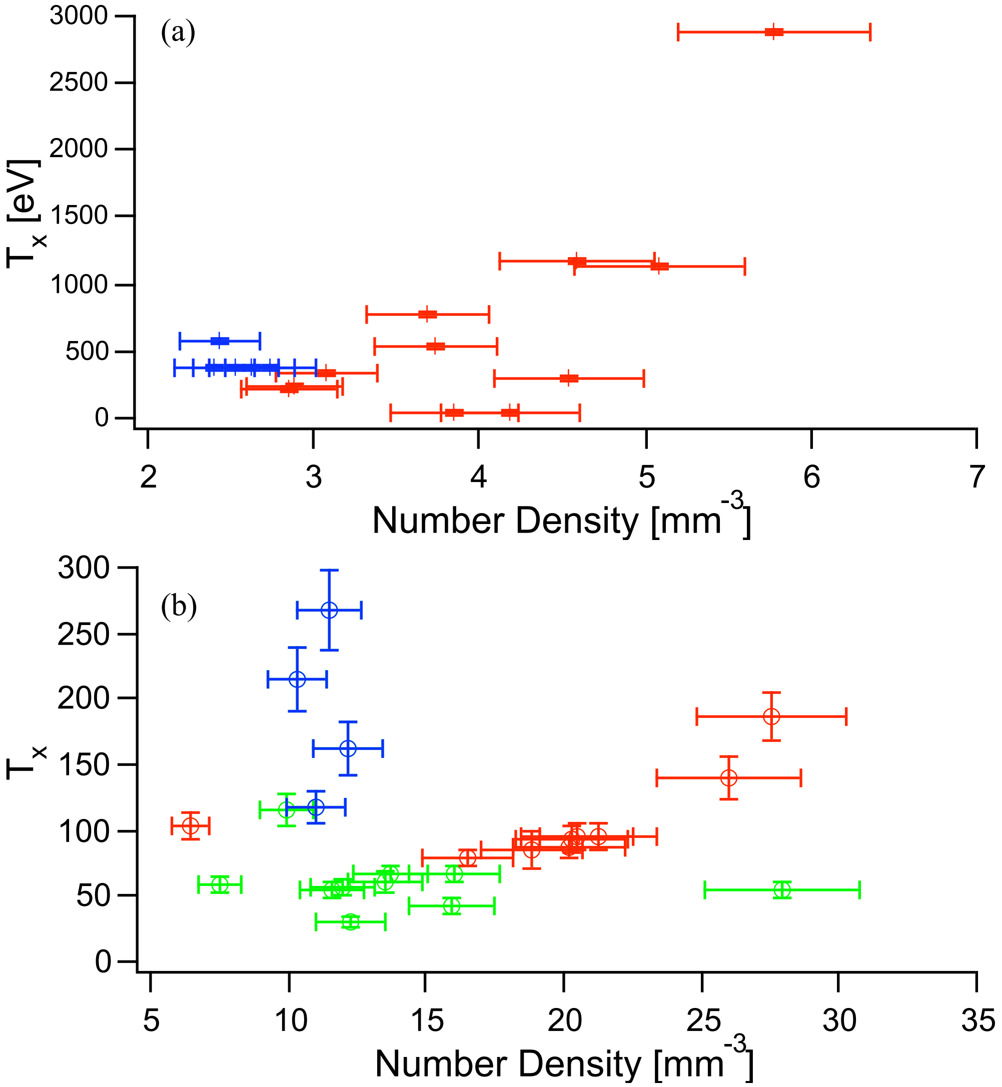


Figure 4.18: A plot of the kinetic temperature as a function of the number density of the microparticle component for dust clouds composed of (a) melamine and (b) silica dust. In (b), the green and blue data corresponds to Cloud 3, while the red data corresponds to Cloud 4.

In Figure 4.18(a), it is clear that the kinetic temperature of the melamine dust exhibits the same dependence on dust number density as seen in the dust clouds composed of alumina microparticles. In the case of the silica data, however, a very different result is observed. It is observed that, for increasing pressure, the number density increased and the kinetic temperature decreased, which is exactly opposite what was observed in the alumina and melamine data. Further, the data plotted in blue in Figure 4.18(b) corresponds to the region where the kinetic temperature was observed to increase at lower pressures. On the other hand, with decreasing pressure, the silica data behaved in the same fashion as the alumina and melamine data. Consequently, this data suggests that a small change in the experimental geometry (i.e. a shift in the z-position of the electrodes) can lead to a significant change in the behavior of the dust.

4.4 DISCUSSION OF RESULTS

In the experiments described in the previous sections, there were two results that were consistently observed: the measured temperatures are highly anisotropic and significantly larger than the temperature of the background plasma. While it is presently unclear what the heating mechanism is responsible for the observed dust temperature, it is clear that the heating mechanism is more efficient for higher particle number densities. These results are consistent with previous observations made using the plasma crystal [65-68, 73, 77, 110].

Within the dusty plasma literature, there are a couple of theoretical models that may explain the observed heating of the microparticles. In papers by Meltzer, *et al.* [65]

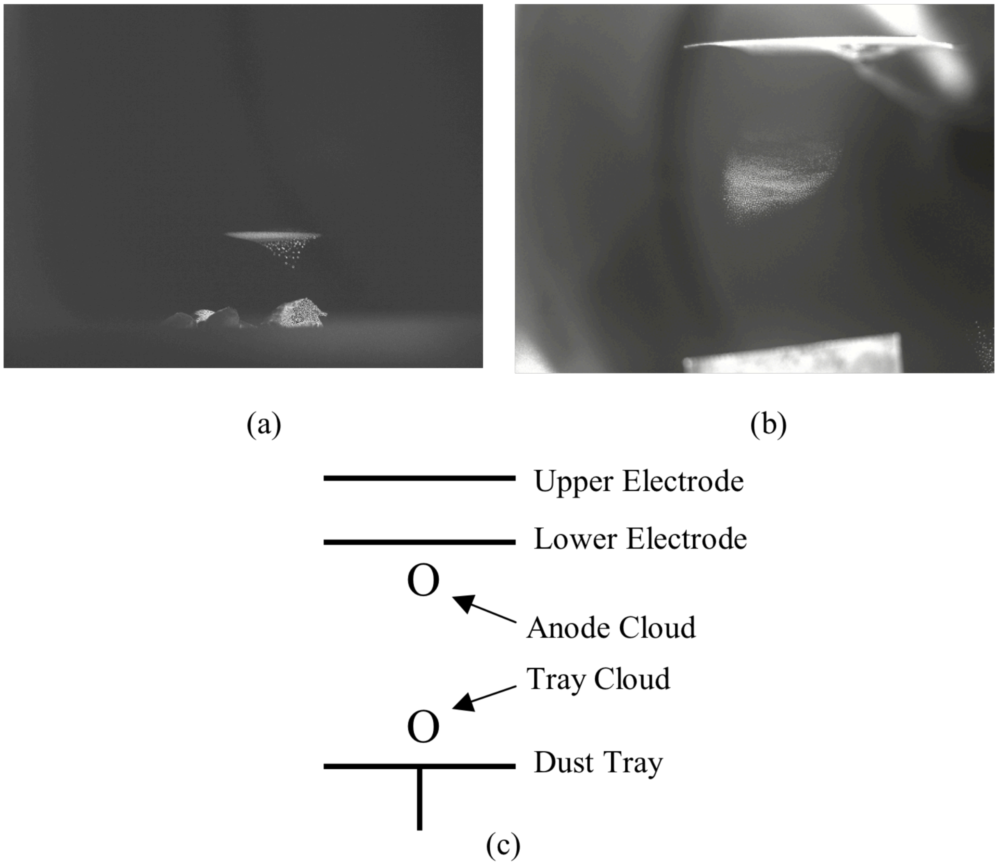
and Joyce, *et al.* [111], it is proposed that the streaming of ions past the dust particles can trigger an instability that leads to dust particle heating, notably, as a function of the neutral gas pressure. Although this work is primarily focused on microparticles that are trapped in the sheath region of rf discharges, the microparticles in this experiment may be subject to a similar mechanism due to the ion flow.

Alternatively, fluctuations in the spatial location or the microparticle charge have been proposed as mechanisms that can lead to microparticle heating [112]. The observations of this study, notably Figures 4.12 and 4.18 give an indication that increasing particle density, or conversely, reduced inter-particle spacing, corresponds to an increase in the microparticle temperature. With reduced interparticle spacing, the microparticles can more closely interact with their neighboring particles giving rise to an enhanced heating of the microparticles. It is noted, however, that this mechanism would have to compete with the reduction in charge due to screening effects from the increased dust density. In either case, it is clear that the heating of the microparticles is significant in this experiment. Additionally, the heating mechanism occurs simultaneously, though asymmetrically, in all three vector directions.

By contrast, the observation of highly asymmetric temperatures is dramatic. It is important to determine if the observed asymmetry is an artifact of the stereo-PIV measurement technique or an indication of the underlying processes that are occurring in the experiment. Recall that the orientation of the electrodes and dust tray in the interior of the 3DPX device, as seen in Figure 4.14, does not suggest that there is any reason for symmetry in 3DPX.

In particular, it is noted that the experimental setup does not have a preferred direction of symmetry. The force of gravity acts in the *y*-direction, breaking the symmetry in the *x*-*y* plane (*e.g.* the plane of the laser sheet). Similarly, there is no reason to expect symmetry between in the *x*-*z* plane. First, the electrodes are fed into the chamber from the back, *e.g.* in the *z*-direction. Further, for improved confinement, the upper electrode is shifted behind the lower electrode, *i.e.* the anode and cathode are not vertically aligned. As a result, the plasma sheath extends around the anode, forming the region where the dust clouds are confined. Additionally, the dust clouds described in the previous sections are suspended in the anode sheath, immediately below the lower electrode, where the behavior of the plasma is quite complex.

In order to explore the asymmetric behavior of the kinetic dust temperature measurements in 3DPX, results from three different experiments are compared. In all of the experiments discussed to this point in this dissertation, the electrodes have been oriented the cathode-anode-tray configuration and the dust is confined in the region immediately below the anode, as seen in Figure 4.19(b). However, this is not the only experimental orientation for which dust can become trapped in the 3DPX device. There is an alternative configuration, the anode-cathode-tray or A-C-T configuration, that can be used to confine the dust particles [113]. In this arrangement, the dust clouds become confined in the region immediately above the dust tray, *i.e.* in the sheath region of the dust tray, as seen in Figure 4.19(a). For comparison, the equilibrium location for the dust clouds in both configurations are schematically depicted in Figure 4.19(c).

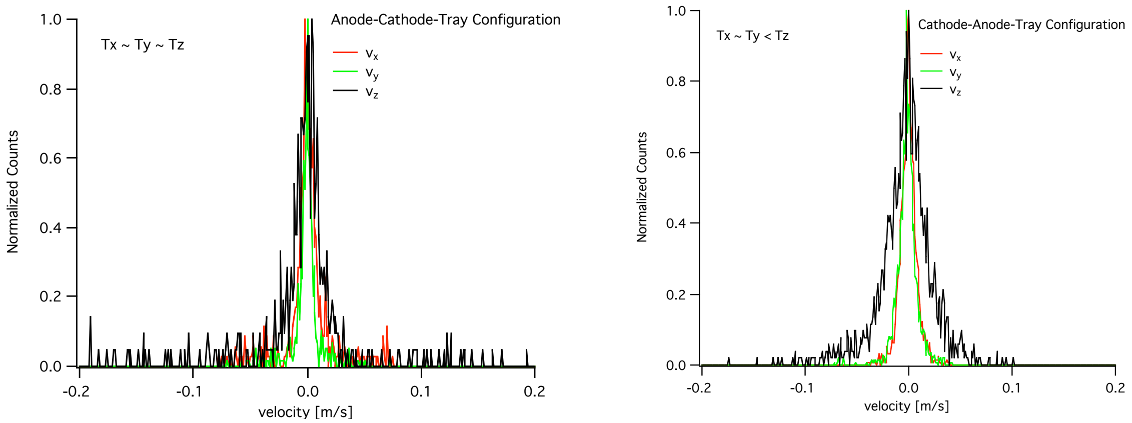

(a) (b)

(c)

Figure 4.19: An image depicting the location of a cloud that is suspended in the (a) tray sheath and (b) anode sheath. In (a) the lower electrode is not visible, but the dust tray can be seen in the lower central region of the image. In (b), the upper electrode is visible, but the dust tray is not. In (c), the relative location of the equilibrium position for a dust cloud in the anode and tray sheath are shown.

The dust clouds in the A-C-T configuration are significantly further from the electrodes that generate the plasma. Given that the equilibrium location of the dust cloud in this configuration is further from the plasma source, it is possible that the dust cloud may be less susceptible to any asymmetry associated with the plasma source.

A comparison is made between the distribution of velocities in a single slice of a dust cloud suspended in the tray and anode sheath shown in Figure 4.20



(a) (b)

Figure 4.20: Distribution of velocities in a single slice from the central region of a dust cloud suspended in the (a) tray sheath and (b) anode sheath. It is observed that the asymmetry seen in the anode region is significantly reduced when the dust cloud is suspended in the tray sheath.

In Figure 4.20(a), a dust cloud was suspended in an argon plasma (p = 102 mTorr, V_{anode} = 173 V, $V_{cathode}$ = -140 V) with the electrodes configured in the A-C-T configuration. The time between laser pulses (*i.e.* the duration of a single measurement) was set to Δt_{laser} = 1.0 ms. In Figure 4.20(b), a dust cloud was suspended in an argon plasma (p = 120 mTorr, V_{anode} = 215 V, $V_{cathode}$ = -97.5 V) with the electrodes configured in the C-A-T configuration. The time between laser pulses was set to Δt_{laser} = 0.5 ms. It is noted that different times between laser pulses were used to extract slightly different information about the type of motion (*i.e.* kinetic or fluid-like) of interest.

As has been discussed previously, a crucial part of this project is determining the validity of the measurements of the distribution function made using the stereo-PIV system. Typically, this can be assessed by comparing the width of the distribution that is measured to the width of the distribution measured using zero-displacement test described in Section 2.2.4.3. This was done for all of the data that has been presented in the previous sections of this chapter, however this measurement was not made for the optical configuration used in the experiments described in this section. However, the zero-displacement test measures the error in the software algorithm and the errors in the optical components of the stereo-PIV system. Given that sources of error have remained relatively unchanged over the course of all of the experiments that have been presented in this dissertation, it is possible to make an assessment as to whether or not the asymmetric distributions are valid.

Typically, the measurement error in the *z*-direction is on the order of 0.2 pixels, while the error in the *x*- and *y*-directions are on the order of 0.1 pixels. Using the time between laser pulses, Δt_{laser}, and the camera calibration (i.e. the number of microns per pixel), it is possible to estimate the resolution limit. The widths of the distributions seen in Figure 4.20 are larger than the resolution error giving confidence in the validity of measured data.

The data presented in Figure 4.20 also suggests that the asymmetries observed in the velocity distributions can vary significantly depending on the trapping location of the cloud. While this does not eliminate the possibility that there may be a contribution due to the PIV diagnostic, the observed asymmetries in the temperature measurements are, in part, a “real” feature of the experiment. Nonetheless, there remain two unresolved questions. First, it is not clear if the smaller asymmetry observed in the tray sheath is due to the larger time interval between laser pulses. Second, the distributions shown in Figure 4.20 are from a single slice of the dust cloud. As a result, it is not clear if the asymmetry is due to the location where we are looking or if the observed asymmetry is a global feature.

To address the first question, a stable dust cloud was generated using the anode-cathode-tray configuration. In this configuration, the dust cloud was suspended in the tray sheath using the following experimental conditions: p = 106 mTorr, V_{anode} = 197.2 V and $V_{cathode}$ = -117.3 V. In this experiment, a series of 40 measurements were made at a single slice of the dust cloud and the time between the laser pulses was varied from 250 μs to 2.5 ms. For all values of Δt_{laser} used, the final interrogation region was a 12 × 12 pixel region of the acquired PIV images. The resulting distributions were fit and the width of the distribution in each vector direction is plotted as a function of time between laser pulses in Figure 4.21

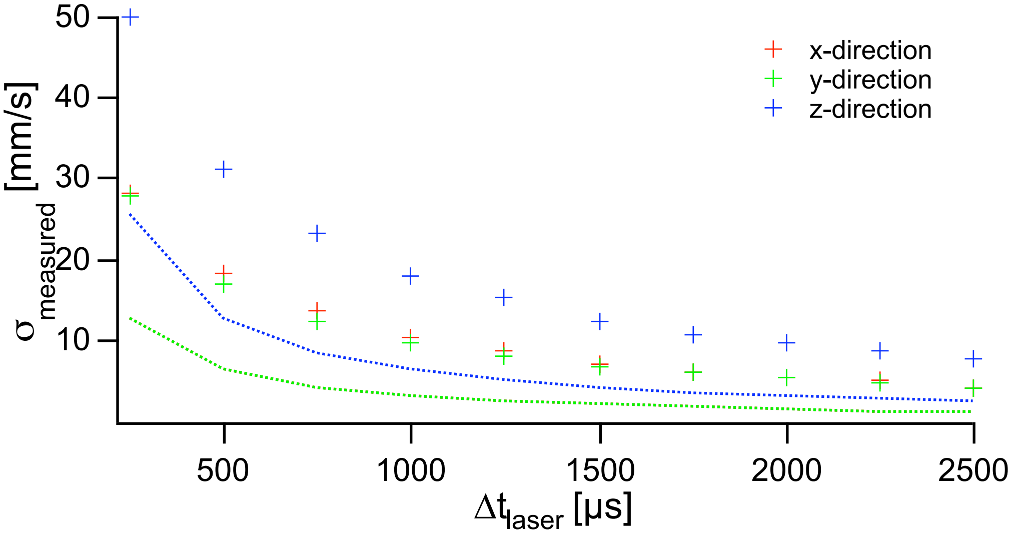


Figure 4.21: A plot of the width of the measured velocity distribution as a function of the time between the laser pulses. It is observed that the width of the distribution gets smaller as the time between laser pulses increases. The lines indicate the estimated resolution limit of the stereo-PIV system.

As was the case for the data in Figure 4.18, the zero displacement test was not performed for the optical setup used in this measurement. However, based on the previously measured values for the resolution error in the stereo-PIV system, the measurement limit can be estimated and is found to be smaller than what is measured. The estimated resolution limit is indicated by the dashed lines in Figure 4.21. Additionally, it is observed that the measured width becomes smaller as the time between laser pulses increases. It is thought that this may be due to more particles leaving the interrogation region, held constant at a 12 × 12 pixel, as the time between laser pulses increases. This shows that the size of the interrogation region used is crucial and depends strongly on the time between laser pulses. As described previously, the interrogation size of 12 × 12 pixel works very well for a time between laser pulses of 250 μs, *e.g.* the majority of the microparticles do not leave the interrogation region between laser pulses.

To gain insight into the nature of the asymmetry that is being measured, the ratio of the width in the *y*-direction to the width in the *x*-direction, $\sigma_y{:}\sigma_x$, and the ratio of the width in the *z*-direction to the width in the *x*-direction, $\sigma_z{:}\sigma_x$, is computed. These ratios, which are an effective measure of the asymmetry in the system, are plotted as a function of time between laser pulses in Figure 4.22

It is observed that the ratio of widths, $\sigma_y{:}\sigma_x$ and $\sigma_z{:}\sigma_x$, is maintained regardless of the time between the laser pulses. For comparison, the ratios in Figure 4.18(a) are $\sigma_z{:}\sigma_x = 1.74$ and $\sigma_y{:}\sigma_x = 0.68$. The ratios in Figure 4.18(b) are $\sigma_y{:}\sigma_x = 1.00$ and $\sigma_z{:}\sigma_x = 2.59$. As a result, the data seen in Figure 4.19 can be compared directly using this ratio of widths, even though the time between laser pulses is different.

These measurements show that the observed asymmetry in temperature is, in large part, due to the microparticles themselves and is not simply a function of the PIV diagnostic. At the present time, it is unclear why the asymmetry varies as it is observed to and it remains an open question as to why the asymmetry is so much stronger when the dust cloud is suspended in the anode spot. Nonetheless, the observed asymmetry is real and not a systematic error in the stereo-PIV system.

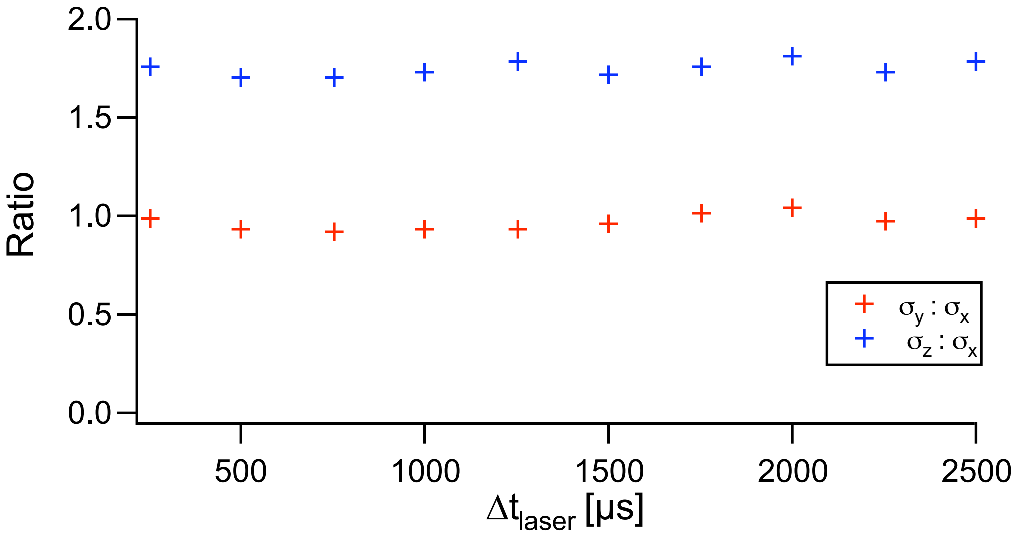


Figure 4.22: A plot of the ratio of the widths: σ_y:σ_x in red and σ_z:σ_x, in blue, as function of time between laser pulses. It is noted that the asymmetry is preserved regardless of the time between laser pulses.

Finally, it is noted that the asymmetry that is observed in the anode spot, while present in the bulk distribution of the dust cloud, is also present throughout the cloud. Figure 4.23 shows the distribution of velocities at each z-location through the entire volume of a dust cloud suspended in the anode spot. This suggests that if an asymmetry is present in a single slice of the cloud, then it will be present throughout the cloud and will be observed in the bulk cloud.

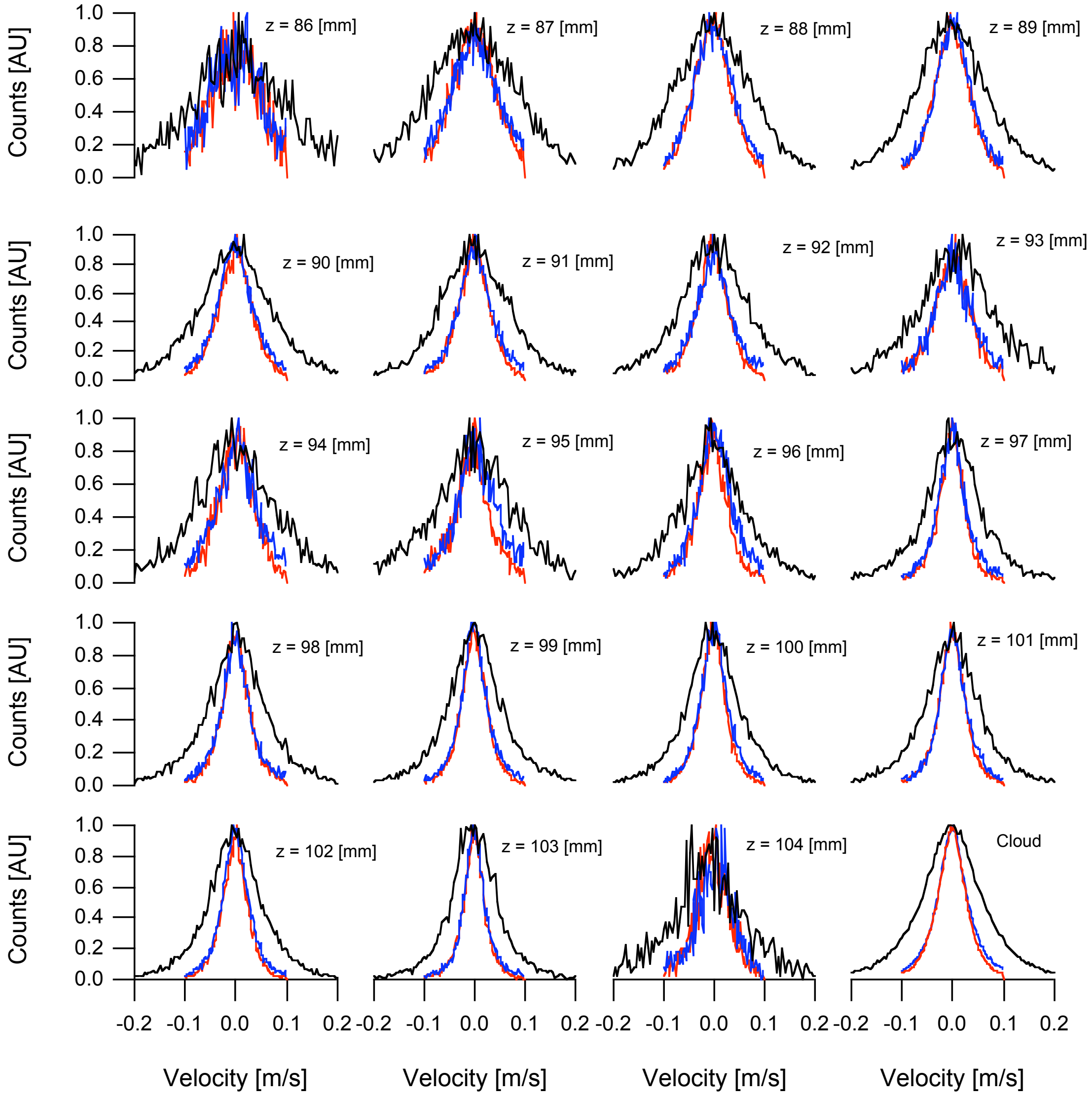


Figure 4.23: Distribution of velocities throughout the volume of a dust cloud suspended in the anode spot. Each plot represents the distribution of velocities measured using the stereo-PIV system at a *z*-location given in the upper right hand corner of the plot. It is observed that the fluctuations in the measured distribution is smallest in the central region of the cloud, where the clouds are bigger and more vectors are reconstructed. Additionally, it is observed that the same asymmetry is observed throughout the cloud and in the distribution for the entire cloud seen in the lower right tile.

Finally, it is noted that the temperatures that are reported here describe the bulk temperature of the entire dust cloud. While this measurement is consistent with the previous experiments involving the rf systems, it is not entirely clear that this definition of temperature is entirely accurate or necessarily appropriate for the weakly-coupled dusty plasmas studied in this dissertation. This is hinted at in Figure 4.23, where it appears that the width of the distributions varies throughout the cloud. Indeed, a detailed examination of the velocity measurements reveals that there is a great deal of structure in the motion of the dust particles and this observed spatial variation in the motion is significantly larger than what was observed in the experiments involving the rf systems. This is seen in Figure 4.24, where contours of velocity in the *x*-, *y*- and *z*-direction are plotted for a single slice of a dust cloud.

A cursory inspection of Figure 4.24 shows that not only is there a large spatial variation in the observed motion, there is much more structure in the *z*-direction. While this justifies the need for the stereo-PIV diagnostic for these studies, it raises an interesting question about the appropriateness of the usual Maxwellian definition of temperature based upon the width of a distribution function. In the kinetic description, it is expected that the kinetic temperature for a system should be the same, regardless of whether one examines the entire systems or a small subset of the system. While this applicable for the experimental studies in the rf systems, it appears that this may not be the case for the weakly-coupled dusty plasmas studied here

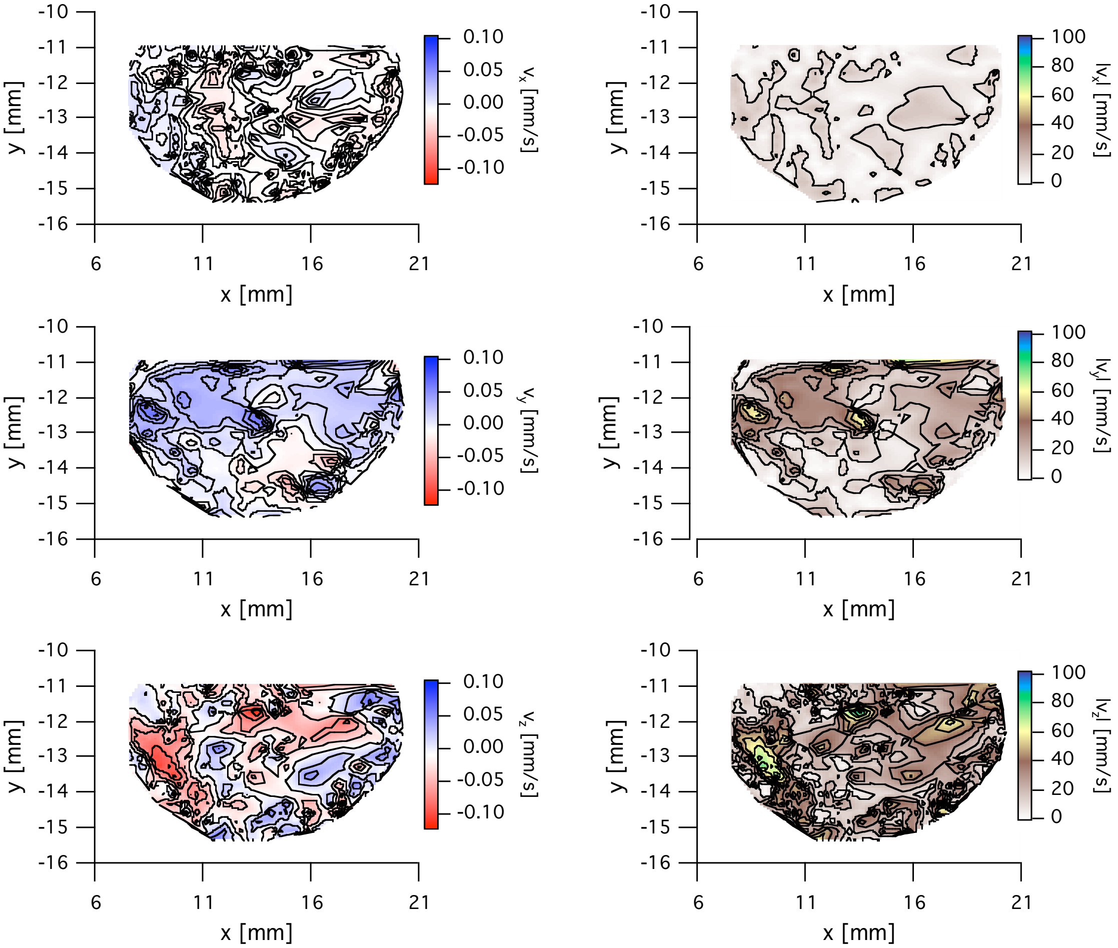


Figure 4.24: Contour plots of the velocity (left column) and speed (right column) in the *x*- (top row) *y*- (middle row) and *z*-direction (bottom row) for a single slice of a weakly-coupled dusty plasma. The lines depict contours of constant velocity/speed, while the shaded background more clearly illustrate the spatial variation of the motion. It is observed that there is a great deal of spatial structure, particularly in the *z*-direction.

To quantify this, the temperature is measured as a function of the *z*-location for a dust cloud composed of 6.2 μm diameter melamine microspheres in an argon plasma generated with p = 122 mTorr, V_{anode} = 209 V and $V_{cathode}$ = -76 V. The result of this measurement is seen in Figure 4.25. It is noted that the large uncertainty in the reported temperature at z = 86 mm is due to the poor statistics that occur at the front of the cloud.

There are three comments that bear mentioning on what is observed. First, the number of measurements at the front of this cloud was relatively small, which accounts for the larger uncertainty in the measurement. Second, these measurements still involve averaging over the *x*-*y* plane and as a result, there is already a loss of spatial information. Nonetheless, it is observed that there is a significant variation in the temperature throughout the dust cloud. However, it does appear that a weighted average of the temperature throughout the cloud is related to the bulk temperature, T_{bulk}, of the cloud using Equation 4.5.

$$T_{bulk} = \frac{\sum_i T_i \rho_i}{\sum_i \rho_i} \tag{4.5}$$

where T_i and ρ_i are the temperature and number density of the microparticle component in the i^{th} slice and the index i extends over the extent of the cloud. The values for the temperature that is extracted from the entire velocity distribution of the cloud, T_{avg}, and temperature found using the weighted average defined in Equation 4.5, T_{bulk}, are reported in Figure 4.25. It is noted that the temperature that is measured using the PIV measurements over the entire dust cloud is consistent to the average temperature at each *z*-location, when weighted by the number density. The differences in temperature are likely due to the large uncertainty in the front of the dust cloud. Consequently, the temperatures that are reported here are consistent with previous measurements, it is not clear if this necessarily the best measurement for the energy of the dust cloud. Clearly, it is necessary to examine the spatial structure of the temperature for these systems in more detail.

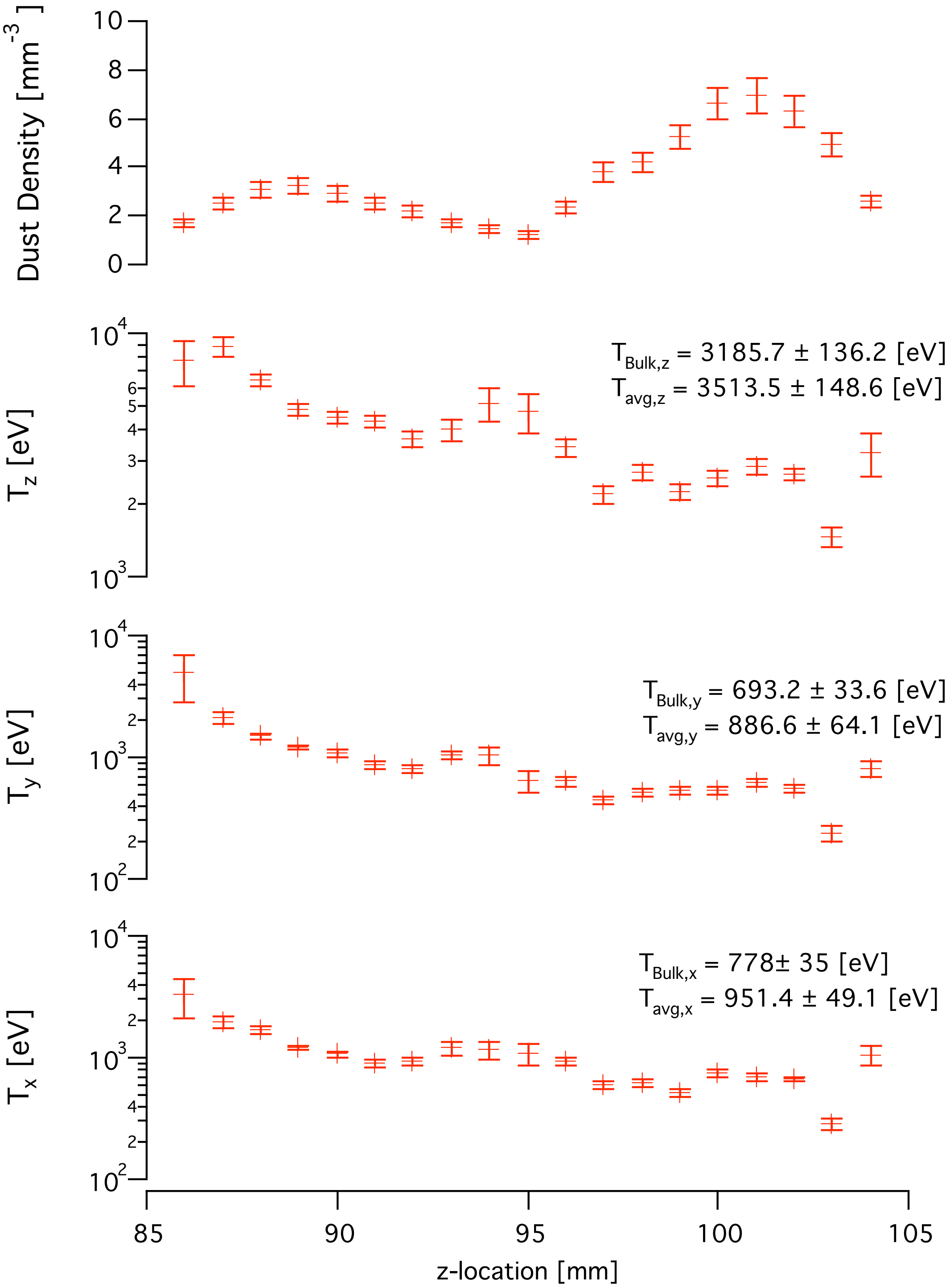


Figure 4.25: Showing the spatial variation in the z-direction of (a) the number density, (b) T_z, (c) T_y and (d) T_z.

CHAPTER 5:CONCLUSIONS

The work in this dissertation presents the first detailed measurements of the velocity space distribution function of a weakly-coupled dusty plasma. Using the 3DPX device, the velocity space distribution of a dust cloud was measured over a wide range of experimental conditions. Using these velocity distributions and the results of extensive simulations of the PIV measurement technique, a bulk kinetic temperature for the entire dust cloud was determined. A summary of the work presented in this dissertation and directions for future research are given in this chapter. In Section 5.1, a summary of the simulations of the PIV measurement technique is presented. In Section 5.2, a summary of the experimental measurements is presented. Finally, directions for future research are discussed in Section 5.3

5.1 SUMMARY OF SIMULATIONS OF THE PIV MEASUREMENT TECHNIQUE

The measurements in this dissertation were made using the particle image velocimetry technique. This technique is a valuable tool in the dusty plasma community, as it provides a whole-field, non-invasive instantaneous measurement of the microparticle transport over the entire illuminated volume of the suspended dust cloud. Unlike particle tracking techniques, this technique returns the average velocity of a number of particles.

While this approach is particularly useful for studying flows and higher number density dusty plasmas where individual particles can not be easily resolved, it does present a challenge when attempting to measure the velocity space distribution function. The reason for this lies in the underlying averaging that occurs in this measurement technique, which results in the suppression of the higher speed particles and an inherent bias towards smaller values of velocity. While this is a well-known feature of the PIV technique in the fluid mechanics community, little work has been done to determine if it is possible to relate what is measured using PIV techniques to the underlying velocity distribution function.

Extensive simulations of the PIV measurement technique were performed to relate the measured distribution of velocities to the underlying velocity distribution function. Assuming that the underlying distribution is Maxwellian, it was found that when the PIV analysis is restricted to an essentially monodisperse distribution of tracer particles, the measured distribution is also Maxwellian and has a width that is equal to or smaller than the width (temperature) of the distribution being measured. These simulations demonstrate that there is a unique mapping function that relates the measured and underlying velocity distribution and depends on the size distribution and the number density of the suspended microparticle component.

At relatively low particle number densities, the simulations showed that the PIV system acts as a particle tracking system, which is well known experimentally. As the particle number density increases, the width of the measured distribution becomes smaller. Indeed, the factor that relates the width of the measured distribution to the underlying velocity distribution exhibits a clear functional dependence on the number density of the microparticle component.

It was also shown that for a distribution of particle sizes, *e.g.* a polydisperse distribution, the curvature of this mapping function is influenced by the width of the particle size distribution. The curvature becomes more extreme as the width of the suspended size distribution becomes smaller. It is noted that in order to observe this effect, the PIV technique must be applied to an essentially monodisperse distribution of tracer particles, which can be accomplished using the multi-intensity PIV technique. If the suspended microparticles have a size distribution and the multi-intensity PIV technique is not used, the measured distributions are not Maxwellian and exhibit a strong peak at zero velocity. In this case, there is no mapping function to relate the measured velocity distribution to the underlying velocity distribution.

5.2 SUMMARY OF THE EXPERIMENTAL MEASUREMENTS

Knowledge of the kinetic velocity space distribution function is an important first step in the description of the thermodynamic state of a given system. In this dissertation, it was shown that through the use of stereoscopic PIV techniques, it is possible to measure the three-dimensional distribution of kinetic velocities of the microparticle component of a dusty plasma.

The first new result presented was an experimental study verifying the applicability of the stereo-PIV technique to study of dusty plasmas. In particular, a perturbation was applied to induce a displacement out of the plane of the laser sheet.

This motion was then observed using the stereo-PIV system. As a result, a new diagnostic technique was added to the field of dusty plasmas.

Using stereoscopic particle image velocimetry, the kinetic velocity space distribution function of a weakly-coupled dusty plasmas was measured for dust clouds composed of polydisperse 1.2 ± 0.5 μm diameter alumina dust, monodisperse 6.22 μm diameter melamine microspheres or monodisperse 3.02 μm diameter silica microspheres. This was accomplished by scanning the stereo-PIV system through the entire dust cloud, making measurements in 1 mm increments and reconstructing the full three-dimensional distribution of velocities. From these measured distributions, a bulk kinetic temperature in each spatial direction of the suspended dust clouds was determined. These experiments were performed in the 3DPX device using dust clouds suspended in an argon dc glow discharge plasma over a wide range of experimental conditions. The electrode configuration for these experiments was a cathode-anode-tray arrangement.

From these measurements, there were three key results observed. First, it was observed that the kinetic temperatures of the dust clouds are substantially larger than the other plasma components. This result is consistent with and larger than previous measurements that were made using strongly-coupled dusty plasmas. Secondly, it was observed that the temperatures are asymmetric. This asymmetry was shown to be a real effect, and not a consequence of the PIV measurement technique. While it is not clear what mechanism is responsible for heating the dust particles, there are a few conclusions that can be drawn about the behavior of the mechanism that is responsible. The heating mechanism is preferential, generating a significantly larger temperature in the *z*-direction of the experimental geometry. Additionally, the heating mechanism is more efficient at lower neutral gas pressures and appears to be more efficient with higher number densities of the microparticle component. From the literature, there are two mechanisms that may be responsible for the observed heating. However, further work is needed to determine if either candidate can explain the observed heating.

5.3 SUMMARY OF RESULTS

In this dissertation, a new diagnostic technique, stereoscopic particle image velocimetry, was introduced in the study of dusty plasmas. Using this technique, the velocity space distribution function of a stable weakly-coupled dust cloud composed of a variety of particle sizes was measured as a function of the neutral gas pressure. It was observed that the distributions were anisotropic. Further, it was shown that this anisotropy is real and was varied significantly. Using the measured velocity space distribution function, a bulk kinetic temperature of the microparticle component was determined. It was found that the kinetic temperature was significantly larger than the other plasma components, but it is presently unclear what is mechanism is responsible for the preferential heating that was observed.

In the process of completing this work, there were a number of issues that have been uncovered that require further investigation. These will be discussed in the following section.

5.4 FUTURE DIRECTIONS

The introduction of the stereo-PIV diagnostic technique opens a wide range of possible studies for weakly-coupled dusty plasmas, including more accurate measurements of transport phenomena, studies of the thermodynamic properties of a dusty plasma, and the visualization of these systems using a phase-space like construction. Additionally, there are a number of issues that have been uncovered in this dissertation that require further investigation.

The simulations that have been presented here yield a great deal of insight into the behavior of the PIV measurement technique and the mapping functions that have been developed are of great use. It may be possible to develop an analytic framework to generate these mapping functions. Additionally, it would be of great value to experimentally measure such a mapping function. This would most easily be done using a two-dimensional plasma crystal, where it is far easier to control the particle density. Further, there is an outstanding question on the size of the anisotropy, particularly in the *z*-direction. While it was shown in this dissertation that this anisotropy is a real effect caused by the underlying heating measurement, it is unclear if a portion of this anisotropy is due to PIV technique. By using a system that is symmetric, such as the Coulomb or Yukawa balls that have recently been observed in an rf discharge [88, 114, 115], one can measure the existence of an anisotropy in the PIV technique directly. Additionally, using the Coulomb balls, one can measure directly the three-dimensional velocity space distribution function [114] allowing one to again directly test the dependence of the mapping functions that have been developed through simulation.

Beyond the outstanding issues involving the stereo-PIV technique, there are a number of physics questions that have been raised in this dissertation that need closer examination. The most obvious entails identifying the heating mechanism that is responsible for the preferential heating that is observed. One possible solution that has been proposed is a two-stream instability between the ions and dust particles. As a result, it is necessary to measure the ion flow in the vicinity of the equilibrium location of the dust cloud. Further, there are a number of interesting details that could be further examined. For instance, there was an anomalously large temperature at a particular set of experimental conditions (6.22 μm diameter melamine microspheres, p = 108 mTorr, V_{anode} = 209 V and $V_{cathode}$ = -76 V). One possible explanation of would be a resonance in the heating mechanism. This could be tested by replacing the manual leak valve with a mass flow controller, which would allow one to carefully and systematically measure the bulk temperature of the dust cloud as a function of pressure. Further insight into the heating mechanism could be gained by examining if the temperature of the microparticle temperature is a function of the equilibrium charge on the dust grain. An initial study of this dependence would involve generating a dust cloud under similar experimental parameters using different sized dust grains. A final experiment that could provide insight into the heating mechanism would involve examining the effect of the location of the dust cloud in the plasma (*i.e.* anode or tray sheath) on the bulk kinetic dust temperature of the dust cloud.

Beyond these topics, there are two additional areas where additional research is needed. In analyzing the experimental data, it was assumed that the distributions were Maxwellian. To first order, this is certainly the case. However, closer examination reveals variations from a Maxwellian distribution. It is unclear if these variations are a consequence of the measurement technique or an indication of some underlying physics. Consequently, it would be of interest to study these variations in some detail. Finally, it would be interesting to examine the spatial structure of the temperature and velocity space distribution functions. While preliminary analysis does not suggest that the temperature that is found by analyzing a single slice of a cloud is directly related to the bulk temperature of the cloud, it does appear that a weighted average of the temperature throughout the cloud is related to the bulk temperature, T_{bulk}, of the cloud. Presently, little work has been done in examining the spatial structure of the temperature and as a result, this is potentially a topic that is rich with new physics.

REFERENCES

[1] Francis F. Chen, *Introduction to Plasma Physics and Controlled Fusion, Volume I: Plasma Physics*, 2 ed. (Spring, 2006).

[2] Sir William Crooke, in *Landmarks of Science* (E.J. Darvey, London, 1879).

[3] Irving Langmuir, Proceedings of the National Academy of Sciences of the United States of America **14**, 627 (1928).

[4] Irving Langmuir, C.G. Found, and A.F. Dittmer, Science **60**, 392 (1924).

[5] Bradford A. Smith, Laurence Soderblom, Raymond Batson et al., Science **215**, 504 (1982).

[6] Robert L. Merlino, in *Fourth International Conference on the Physics of Dusty Plasmas*, edited by M. Mikkikian and P.K. Shukla L. Boufendi (AIP Conference Proceedings, Orleans, France, 2005), Vol. 799, pp. 594.

[7] Gary S. Selwyn, John E. Heidenreich, and Kurt L. Haller, Applied Physics Letters **57**, 1876 (1990).

[8] G. S. Selwyn, J. Singh, and R. S. Bennett, Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films **7**, 2758 (1989).

[9] J H Chu, Ji-Bin Du, and Lin I, Journal of Physics D: Applied Physics **27**, 296 (1994).

[10] H. Thomas, G. E. Morfill, V. Demmel et al., Physical Review Letters **73**, 652 (1994).

[11] Yasuaki Hayashi and Kunihide Tachibana, Japanese Journal of Applied Physics **33**, L804 (1994).

[12] H. Ikezi, Physics of Fluids **29**, 1764 (1986).

[13] A. Melzer, T. Trottenberg, and A. Piel, Physics Letters A **191**, 301 (1994).

[14] J. B. Peiper and J. Goree, Physical Review Letters **77**, 3137 (1996).

[15] Shukla P K and Yu M Y Rao N N, Planetary and Space Science **38**, 543 (1990).

[16] Merlino R L Barkan A, and D'Angelo N, Physics of Plasmas **2**, 3563 (1995).

[17] Jr. Edward Thomas, Physics of Plasmas (13), 042107 (2006).

[18] Frank Melandsø, Physics of Plasmas **3**, 3890 (1996).

[19] J. E. Allen, B. M. Annaratone, and U. de Angelis, Journal of Plasma Physics **63**, 299 (2000).

[20] Martin Lampe, Rajiv Goswami, Zoltan Sternovsky et al., Physics of Plasmas **10** (5), 1500 (2003).

[21] E. B. Tomme, D. A. Law, B. M. Annaratone et al., Physical Review Letters **85**, 2518 (2000).

[22] B. M. Annaratone, T. Antonova, H. M. Thomas et al., Physical Review Letters **93**, 185001 (2004).

[23] S. Ratynskaia, K. Rypdal, C. Knapek et al., Physical Review Letters **96**, 105010 (2006).

[24] Gregor E. Morfill, Milenko Rubin-Zuzic, Hermann Rothermel et al., Physical Review Letters **92**, 175004 (2004).

[25] J. Winter, Physics of Plasmas **7**, 3862 (2000).

[26] A. W. Kleyn, N. J. Lopes Cardozo, and U. Samm, Physical Chemisty Chemical Physics **8**, 1761 (2006).

[27] J Winter, Plasma Physics and Controlled Fusion **46** (2004).

[28] U. de Angelis, Physics of Plasmas **13**, 12514 (2006).

[29] P. K. Shukla and A. A. Mamun, *Introduction to Dusty Plasma Physics*. (Institute of Physics Publishing, Bristol and Philadelphia, 2002).

[30] S.V. Vladimirov, K. Ostrikov, and A.A. Samarian, *Physics and Applications of Complex Plasmas*. (Imperial College Press, 2005).

[31] E.C. Whipple, Reports on Progress in Physics **44**, 1197 (1981).

[32] O. Havnes, T. K. Aanesen, and F. Melandsø, Journal of Geophysical Research **92**, 6581 (1987).

[33] P. K. Shukla, Physics of Plasmas **1**, 1362 (1994).

[34] N.N. Rao, P.K. Shukla, and M.Y. Yu, Planetary and Space Science **38**, 543 (1990).

[35] A. Barkan, R.L. Merlino, and N. D'Angelo, Physics of Plasmas **2**, 3563 (1995).

[36] Daniel H. E. Dubin, Physical Review A **42**, 4972 (1990).

[37] A. Bouchoule, A. Plain, L. Boufendi et al., Journal of Applied Physics **70** (1991) (1991).

[38] G. M. Jellum and D. B. Graves, Journal of Applied Physics **67**, 6490 (1990).

[39] W. Bohme, W.E. Kohler, M. Romheld et al., IEEE Transactions on Plasma Science **22**, 110 (1994).

[40] G Lapenta and J U Brackbill, Plasma Sources Science and Technology **6**, 61 (1997).

[41] G. S. Selwyn, Plasma Sources Science and Technology **3**, 340 (1994).

[42] E. Stoffels, W.W. Stoffels, G. Ceccone et al., Journal of Applied Physics **86**, 3442 (1999).

[43] H. Hofmeister, J. Duta, and H. Hofmann, Physical Review B **54**, 1136 (1996).

[44] H. Kersten, P. Schmetz, and G.M.W. Kroesen, Surface and Coatings Technology **108-109**, 597 (1998).

[45] P. Cabarrocas, P. Gay, and A. Hadjadj, Journal of Vacuum Science Technology A **14**, 655 (1996).

[46] P. Cabarrocas, N. Layadi, M. Kunst et al., Journal of Vacuum Science Technology A **16**, 436 (1998).

[47] J. Abrahamson and J. Dinniss, Nature **403**, 519 (2000).

[48] O. Havnes, T. Aslaksen, and A. Brattli, Physica Scripta **T89**, 133 (2001).

[49] G.A. Wurden, A.J. Wurden, and I.M. Gladstone Jr., IEEE Transactions on Plasma Science **27**, 142 (1999).

[50] M. Horanyi, T.W. Hartquist, O. Havnes et al., Reviews of Geophysics **42**, RG4002 (2004).

[51] S. Messenger, Nature **404**, 968 (2000).

[52] S. Messenger, L. Kelley, and D. Lauretta, Science **309**, 737 (2005).
[53] U. de Angelis, Physica Scripta **45**, 465 (1992).
[54] J. P. Simpson and F. C. Witteborn, Applied Optics **16**, 2051 (1977).
[55] D.A. Mendis, H.L.F Houpis, and J.R. Hill, Journal of Geophysical Research **87**, 3449 (1982).
[56] J. E. Howard, H. R. Dullin, and M. Horányi, Physical Review Letters **84**, 3244 (2000).
[57] D.P. Hamilton, Icarus **101**, 244 (1993).
[58] H. R. Dullin, M. Horányi, and J. E. Howard, Physica D **171**, 178 (2002).
[59] S. V. W. Beckwith, T. Henning, and Y. Nakagawa, in *Protostars and Planets IV*, edited by V. Mannings, A. P. Boss, and S. S. Russell (University of Arizona Press, 2000), pp. 533.
[60] J. Blum and M. Münch, Icarus **106**, 151 (1993).
[61] J. Blum, G. Wurm, S. Kempf et al., Physical Review Letters **85**, 2426 (2000).
[62] M. Horanyi and C. K. Goertz, Astrophysical Journal **361**, 155 (1990).
[63] S. J. Weidenschilling and J. N. Cuzzi, in *Protostars and Planets III*, edited by E. H. Levy and J. I. Lunine (University of Arizona Press, 1993), pp. 1031.
[64] G. Wurm and J. Blum, Icarus **132** (135) (1998).
[65] A. Melzer, A. Homann, and A. Piel, Physical Review E **53**, 2757 (1996).
[66] R. A. Quinn and J. Goree, Physics of Plasmas **7**, 3904 (2000).
[67] R. Ichiki, Y. Ivanov, M. Wolter et al., Physical Review E **70**, 066404 (2004).
[68] Yuriy Ivanov and André Melzer, Physics of Plasmas **12**, 072110 (2005).
[69] Matthias Wolter and André Melzer, Physical Review E **71**, 036414 (2005).
[70] V. Nosenko, J. Goree, and A. Piel, Physics of Plasmas **13**, 032106 (2006).
[71] G. E. Morfill, H. M. Thomas, U. Konopka et al., Physics of Plasmas **6**, 1769 (1999).
[72] G. H. P. M. Swinkels, H. Kersten, H. Deutsch et al., Journal of Applied Physics **88**, 1747 (2000).
[73] V. V. Zhakhovskii, V. I. Molotkov, A. P. Nefedov et al., Journal of Experimental and Theoretical Physics Letters **66**, 419 (1997).
[74] S. Nunomura, S. Zhdanov, D. Samsonov et al., Physical Review Letters **94**, 0454001 (2005).
[75] D. Samsonov, S. K. Zhdanov, R. A. Quinn et al., Physical Review Letters **92**, 255004 (2004).
[76] V. E. Fortov, O. F. Petrov, V. I. Molotkov et al., Physical Review E **71**, 036413 (2005).
[77] M. Schabel, T. Peterson, J. Sinclair et al., Journal of Applied Physics **86**, 1834 (1999).
[78] Jr. E. Thomas, J. D. Williams, and J. Silver, Physics of Plasmas **11**, L37 (2004).
[79] M. Raffel, C. Willert, and J. Kompenhans, *Particle Image Velocimetry: A Practical Guide*. (Springer, 1998).
[80] T. Trottenberg, D. Block, and A. Piel, Physics of Plasmas **13**, 042105 (2006).
[81] Jr. Edward Thomas and Michael Watson, Physics of Plasmas **6**, 4111 (1999).
[82] Shota Nunomura, Noriyasu Ohno, and Shuichi Takamura, Japanese Journal of Applied Physics **36**, L949 (1997).

[83] V. I. Molotkov, A. P. Nefedov, V. M. Torchinskii et al., Journal of Experimental and Theoretical Physics Letters **89** (477) (1999).

[84] Ronald J. Adrian, Annual Review of Fluid Mechanics **23**, 261 (1991).

[85] J. Westerweel, Measurement, Science and Technology **8**, 1392 (1997).

[86] M. Stanislas and J.C. Monnier, Measurement Science and Technology **8**, 1417 (1997).

[87] A. K. Prasad and R. J. Adrian, Experiments in Fluids **15**, 49 (1993).

[88] O. Arp, D. Block, M. Klindworth et al., Physics of Plasmas **12**, 122102 (2005).

[89] Jr. Edward Thomas, Physics of Plasmas **6**, 2672 (1999).

[90] Jr. Edward Thomas, Physics of Plasmas **8**, 329 (2001).

[91] Jr. Edward Thomas and Jeremiah Williams, Physics of Plasmas **13**, 055702 (2006).

[92] Richard D. Keane and Ronald J. Adrian, Applied Scientific Research **49**, 191 (1992).

[93] J. Westerweel, D. Dabiri, and M. Gharib, Experiments in Fluids **23**, 20 (1997).

[94] K.D. Hinsch, Measurement Science and Technology **6**, 742 (1995).

[95] A.K. Prasad and K. Jensen, Applied Optics **34**, 7092 (1995).

[96] B. Weinke, Experiments in Fluids **39**, 267 (2005).

[97] R.Y. Tsai, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'86)* (Miami, FL, 1986).

[98] N.J. Lawson and J. Wu, Measurement Science and Technology **8**, 894 (1997).

[99] N.J. Lawson and J. Wu, Measurement Science and Technology **8**, 1455 (1997).

[100] Dr. Steve Anderson (private communication).

[101] H. Huang, D. Dabiri, and M. Gharib, Measurement Science and Technology **8**, 1427 (1997).

[102] D.J. Forliti, P.J. Strykowski, and K. Debatin, Experiments in Fluids **28**, 436 (2000).

[103] D. Samsonov and J. Goree, Physical Review E **59** (1), 1047 (1999).

[104] A. Plain, Journal of Applied Physics **83**, 4012 (1998).

[105] L.C. Mountford, R.A. Smith, and M.H.R. Hutchinson, Review of Scientific Instruments **69**, 3780 (1998).

[106] C. Thompson, A. Barkan, N. D'Angelo et al., Physics of Plasmas **4**, 2331 (1997).

[107] J. McKee and E. Thomas, in *APS Division of Plasma Physics* (Bulletin of the American Physical Society, 2006), Vol. 51.

[108] B. M. Annaratone, S. A. Khrapak, P. Bryant et al., Physical Review E **66**, 056411 (2002).

[109] V. R. Palero and Y. Ikeda, Measurement Science and Technology **13**, 1050 (2002).

[110] H. Thomas and G. Morfill, Nature **379**, 806 (1996).

[111] G. Joyce, M. Lampe, and G. Ganguli, Physical Review Letters **88**, 095006 (2002).

[112] O.S. Vaulina, S.A. Khrapak, A.P. Nefedov et al., Physical Review E **60**, 5959 (1999).

[113] Jr. Edward Thomas and Jeremiah Williams, Physical Review Letters **95**, 055001 (2005).

[114] Sebastian Käding and André Melzer, Physics of Plasmas **13**, 090701 (2006).

[115] O. Arp, D. Block, A. Piel et al., Physical Review Letters **93**, 165004 (2004).

APPENDIX 1: MATHEMATICAL BACKGROUND

In Chapter 2, an overview of the PIV measurement technique and calibration process was given. Here, the mathematical details are presented. Section A.1. 1 presents the mathematical details of the two dimensional PIV technique. Section A.1. 2 presents a proof of the correlation theorem. Section A.1. 3 presents the mathematical details of the pinhole model.

A.1.1: MATHEMATICAL FOUNDATION OF THE 2D PIV TECHNIQUE

The following description follows the presentation of Adrian [79].

A.1.1.1 MATHEMATICAL DESCRIPTION OF IMAGE FORMATION

Consider a fluid flow illuminated by a laser sheet and depicted in Figure A.1.1.

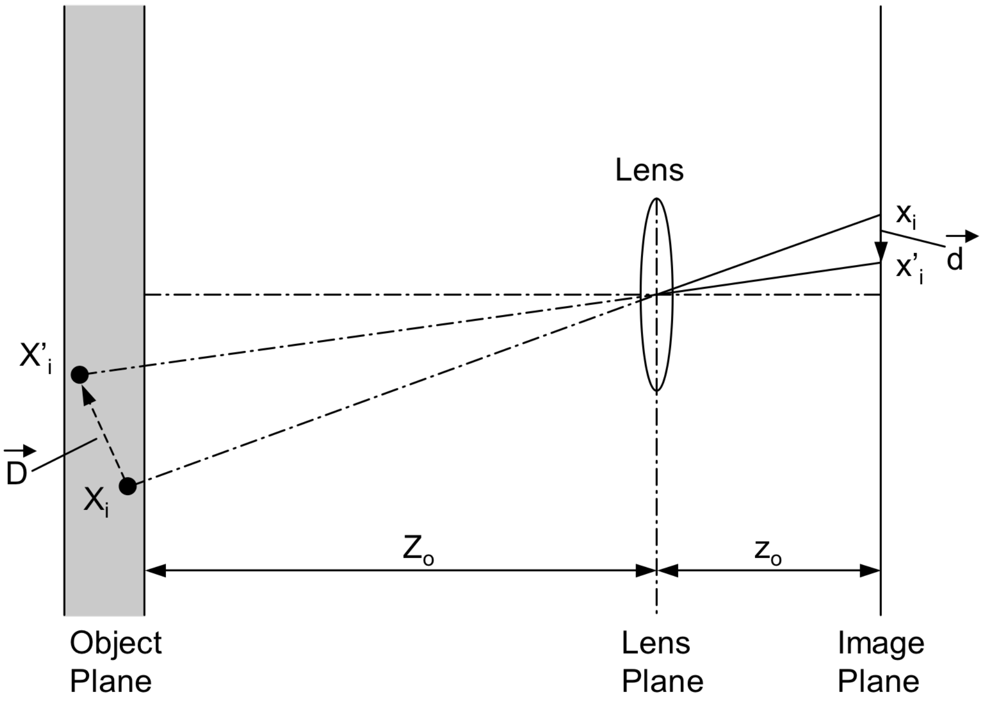


Figure A.1.1: Imaging of a particle in the laser sheet on the CCD camera. The fluid flow is illuminated by a laser sheet at the object plane and imaged on the CCD array at the image plane. In this figure, lower case letters denote quantities measured by the CCD camera, while capital letters denote quantities illuminated by the laser sheet. The position of a particles at time t_0 are denoted by x_1 and t_1 by x_1'.

In a given region of this illuminated slice, *i.e.* the object plane, there are N tracer particles. At time t_0, the position, $\vec{X}_i$, of the tracer particles, Γ_0, in this interrogation volume can be described by Equation A.1.1.

$$\Gamma_0 = \begin{pmatrix} \vec{X}_1 \\ \vec{X}_2 \\ \vdots \\ \vec{X}_N \end{pmatrix} \quad with \quad \vec{X}_i = \begin{pmatrix} X_i \\ Y_i \\ Z_i \end{pmatrix} \tag{A.1.1}$$

where the position of the i^{th} particle in coordinate space is give by X_i, Y_i and Z_i. These particles will be imaged on the CCD element at location $\vec{x}_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix}$, where x_i and y_i denote a position on the CCD array. Using a similar notation, the location of these particles can be described by Equation A.1.2.

$$\vec{\gamma} = \begin{pmatrix} \vec{x}_1 \\ \vec{x}_2 \\ \vdots \\ \vec{x}_N \end{pmatrix} \quad with \quad \vec{x}_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix} \tag{A.1.2}$$

Here, we note the use of the convention of capital letters to denote objects in the object plane and lower case letters denote objects in the image plane. Further, it is noted that positions in the image and object plane are related to each other by Equation A.1.3.

$$\begin{aligned} x_i &= MX_i \\ y_i &= MY_i \end{aligned} \tag{A.1.3}$$

where M is the magnification. Implicit in Equation A.1.3 is the assumption that the magnification is uniform across the interrogation volume. This assumption is reasonable in the application of 2D PIV, where the camera is oriented perpendicular to the laser sheet.

At time $t_1 = t_o + \Delta t$, the N particles in the interrogation volume have moved to locations $\vec{X}_i' = \vec{X}_i + \vec{D}$, where $\vec{X}_i'$ and $\vec{D}_i = \begin{pmatrix} D_{i,x} \\ D_{i,y} \\ D_{i,z} \end{pmatrix}$ denote the new position and the displacement of the ith particle respectively. Regardless of how complicated the motion, the displacement of the particles can be described as a linear translation, $\vec{D}$, provided that the time between successive images, Δt, is small enough. Under this assuming, the position of the original N particles is given by Equation A.1.4.

$$\Gamma_1 = \begin{pmatrix} \vec{X}_1' \\ \vec{X}_2' \\ \vdots \\ \vec{X}_N' \end{pmatrix} = \begin{pmatrix} \vec{X}_1 + \vec{D} \\ \vec{X}_2 + \vec{D} \\ \vdots \\ \vec{X}_N + \vec{D} \end{pmatrix} with \ \vec{X}_1' = \begin{pmatrix} X_i + D_x \\ Y_i + D_y \\ Z_i + D_z \end{pmatrix} \qquad \text{(A.1.4)}$$

The displacement which is described by Equation A.1.4 is depicted in Figure A.1.1.

Assuming that there are no distortions in the imaging optics, the location of the particle in the second image, $\vec{x}_i'$, can be related to the location of the particle in the first image, $\vec{x}_i$, using the matrix formulation of optics, defined in Equation A.1.5.

$$\vec{x}_i' = \overleftrightarrow{P} \circ \overleftrightarrow{D} \circ \overleftrightarrow{P}^{-1} \circ \vec{x}_i \qquad \text{(A.1.5)}$$

where $\vec{x}'$ defines a location in the plane of the (virtual) light sheet, $\vec{x}_i$ defines a location in the fluid flow, $\overleftrightarrow{P}^{-1}$ defines the transformation of a point in fluid flow to a point on the image plane (*i.e.* the CCD array), $\overleftrightarrow{D}$ defines the displacement of the particles in the fluid flow, $\overleftrightarrow{P}$ is the transformation of a point on the image plane onto the plane of the (virtual) light sheet. These quantities, in their matrix form are given below.

$$\overleftrightarrow{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{z_o} & 1 \end{bmatrix} \tag{A.1.6}$$

$$\overleftrightarrow{D} = \begin{bmatrix} 1 & 0 & 0 & D_X \\ 0 & 1 & 0 & D_Y \\ 0 & 0 & 1 & D_Z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.1.7}$$

$$\overleftrightarrow{P}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -z_o(1+M) \\ 0 & 0 & 0 & -M \end{bmatrix} \tag{A.1.8}$$

From this, one can see that the displacement in the recorded image, $\vec{d} = \vec{x}_i' - \vec{x}_i$, is related to the real displacement, $\vec{D}$, through Equation A.1.9.

$$
\begin{aligned}
d_x &= x_i' - x_i = -M(D_x + D_z \frac{x_i'}{z_o}) \\
d_y &= y_i' - y_i = -M(D_y + D_z \frac{y_i'}{z_o})
\end{aligned}
\tag{A.1.9}
$$

A key assumption in the application of 2D PIV is that the displacement is predominately in the plane of the laser sheet (i.e. the *x*-*y* plane), which is to say that $D_z \approx 0$. In this limit, the observed motion is described by Equation A.1.10.

$$
\begin{pmatrix} d_x \\ d_y \end{pmatrix} = M \begin{pmatrix} D_x \\ D_y \end{pmatrix} \tag{A.1.10}
$$

A.1.1.2 MATHEMATICAL DESCRIPTION OF THE CROSS CORRELATION ANALYSIS TECHNIQUE

The intensity profile, *I,* of an image of the tracer particles that is formed on the CCD array can be described using Equation A.1.11.

$$
I_0 = I_0(\vec{x}, \vec{\Gamma}) = \sum_{i=1}^{N} V(\vec{X}_i)\tau(\vec{x}_i)\delta(\vec{x} - \vec{x}_i) = \sum_{i=1}^{N} V(\vec{X}_i)\tau(\vec{x} - \vec{x}_i) \tag{A.1.11}
$$

where $V(\vec{X}_i)$ is a function that describes the light that is scattered off of a particle located at $\vec{X}_i$ and $\tau(\vec{x}_i)$ is a function that describes the spread of light scattered off of a particle located at $\vec{X}_i$ measured at $\vec{x}_i$ due to all optical elements. Similarly, the image formed on the CCD array at time t_1 would then be described by Equation A.1.12.

$$I_1 = I_1\left(\vec{x},\vec{\Gamma}\right) = \sum_{j=1}^{N} V\left(\vec{X}_j + \vec{D}\right)\delta\left(\vec{x} - \vec{x}_j\right)\tau\left(\vec{x} - \vec{x}_j - \vec{d}\right) \qquad \text{(A.1.12)}$$

It is noted that a different index, *j*, is used here when summing over the particles in the image to highlight that this describes the image at t_1 (*i.e.* the second image).

The next step in the process is to determine the displacements. This is typically done by locally cross correlating the two images, t_0 and t_1. The cross correlation of these two images, describe by Equations A.1.11 and A.1.12 respectively, is given by Equation A.1.13.

$$\begin{aligned} R\left(\vec{s},\vec{\Gamma},\vec{D}\right) &= \frac{1}{A}\int_A I_0\left(\vec{x},\vec{\Gamma}\right)I_1\left(\vec{x},\vec{\Gamma}\right) \\ &= \frac{1}{A}\sum_{i,j} V\left(\vec{X}_i\right)V\left(\vec{X}_j + \vec{D}\right)\int_A \tau\left(\vec{x} - \vec{x}_i\right)\tau\left(\vec{x} - \vec{x}_j + \vec{s} - \vec{d}\right) \\ &= \sum_{i,j} V\left(\vec{X}_i\right)V\left(\vec{X}_j + \vec{D}\right)R_\tau\left(\vec{x}_i - \vec{x}_j + \vec{s} - \vec{d}\right) \end{aligned} \qquad \text{(A.1.13)}$$

where $\vec{s}$ is the separation vector in the correlation plane, *A* is the area of the interrogation region, $R_\tau\left(\vec{x}_i - \vec{x}_j + \vec{s} - \vec{d}\right) \equiv \frac{1}{A}\int_A \tau\left(\vec{x} - \vec{x}_i\right)\tau\left(\vec{x} - \vec{x}_j + \vec{s} - \vec{d}\right)$ and the indices are used to denote particles in different frames (*i.e.* t_0 and t_1). Equation (A.1.13) can be clarified by breaking the sum into the $i = j$ and the $i \neq j$ terms.

$$
\begin{aligned}
R\left(\vec{s},\vec{\Gamma},\vec{D}\right) = R_{\tau}\left(\vec{s}-\vec{d}\right)\sum_{i=1}^{N} V\left(\vec{X}_i\right)V\left(\vec{X}_i+\vec{D}\right) + \\
\sum_{i\neq j} V\left(\vec{X}_i\right)V\left(\vec{X}_j+\vec{D}\right)R_{\tau}\left(\vec{x}_i-\vec{x}_j+\vec{s}-\vec{d}\right)
\end{aligned}
\qquad \text{(A.1.14)}
$$

where, the first term in the summation contains from the $i = j$ terms, while the second term contains the $i \neq j$ terms. Physically, the first term contains the correlation information between identical particles (*i.e.* the desired displacement information). The second term is a convolution of the mean intensities of the images and the correlations between random particles. The second term defines the noise level in the correlation plane. Using the notation of Adrian [79], the correlation function given in Equation A.1.15 can be written as

$$
R\left(\vec{s},\vec{\Gamma},\vec{D}\right) = R_D\left(\vec{s},\vec{\Gamma},\vec{D}\right) + R_F\left(\vec{s},\vec{\Gamma},\vec{D}\right) + R_C\left(\vec{s},\vec{\Gamma},\vec{D}\right) \qquad \text{(A.1.15)}
$$

where $R_D\left(\vec{s},\vec{\Gamma},\vec{D}\right)$ represents the correlation information between identical particles (*i.e.* the first terms in Equation A.1.14), $R_F\left(\vec{s},\vec{\Gamma},\vec{D}\right)$ represents the correlations between random particles and $R_C\left(\vec{s},\vec{\Gamma},\vec{D}\right)$ represents the convolution of the mean intensities. Figure A.1.2 depicts a typical cross correlation function.

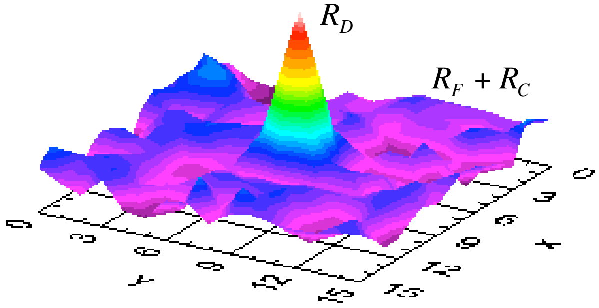


Figure A.1.2: Depicting the correlation function. The displacement used to generate this correlation plane was 8 pixels in both *x* and *y*.

Let us now more closely examine the term of interest, $R_D\left(\vec{s},\vec{\Gamma},\vec{D}\right)$.

$$R_D\left(\vec{s},\vec{\Gamma},\vec{D}\right) = R_\tau\left(\vec{s}-\vec{d}\right)\sum_{i=1}^{N} V\left(\vec{X}_i\right)V\left(\vec{X}_i+\vec{D}\right) \tag{A.1.16}$$

This function is maximum when $\vec{s}=\vec{d}$. Provided that the signal of interest $R_D\left(\vec{s},\vec{\Gamma},\vec{D}\right)$ exceeds the noise floor, one needs to find the location of the peak for this portion of correlation function. It is noted that when this analysis is done, one specifies how much larger this portion of correlation function must be, relative to the noise floor defined by the quantity $R_F\left(\vec{s},\vec{\Gamma},\vec{D}\right)+R_C\left(\vec{s},\vec{\Gamma},\vec{D}\right)$, *i.e.* a signal to noise ratio. In the experiments presented in this thesis, this value is set to 1.2.

The location of this peak directly yields the desired average in-plane displacement, $\vec{d}$. Typically, the correlation function is fitted and the peak is found from the fitted function, allowing for sub-pixel resolution in the measured displacement.

The process to implement the cross-correlation technique is relatively straightforward and can be seen in Figure A.1.3.

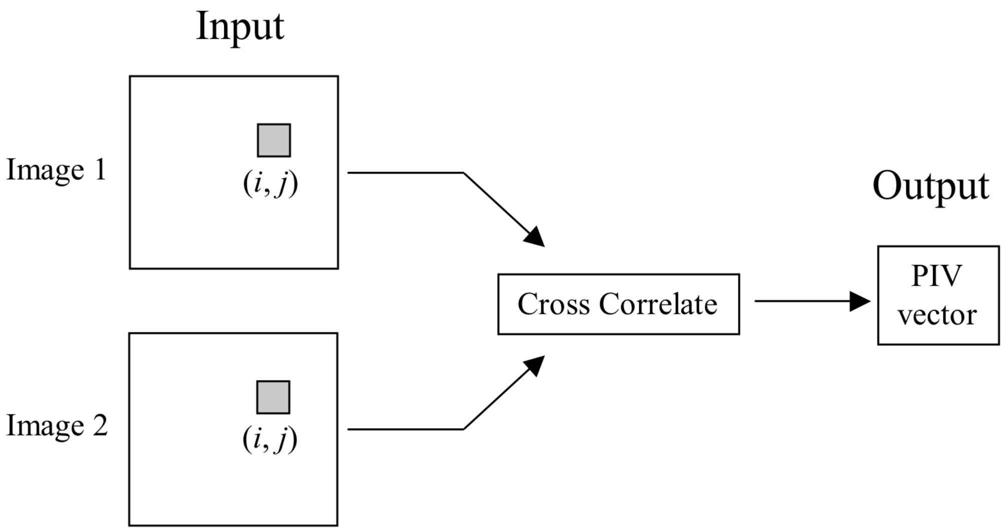


Figure A.1.3: Depicting the process used to construct a PIV vector. Identical $n \times n$ evaluation regions from Image 1, taken at time t_0, and Image 2, taken at time t_1, are extracted. A cross correlation analysis of these two image subsets is done and the peak in the correlation plane yields the desired in-plane displacement, $\vec{d}$.

In practice, one does not analytically attempt to estimate the displacement function, $\vec{d}$. Instead, one statistically finds the best match between the two images, t_0 and t_1, using the discrete cross correlation function,

$$R(k,l) = \sum_{k=-K}^{K} \sum_{l=-L}^{L} I_0(x,y) I_1(x+k, y+l) \qquad \text{(A.1.17)}$$

where I and I’ are subsets of images $I_0(x, \Gamma)$ and $I_1(x, \Gamma)$, with I' being larger than I, centered at location (x, y) of the CCD array. Physically, the image element I is shifted by an amount (k, l) over the image element I'. This shifting operation is done over a range of values ($-M \leq k \leq M$, $-N \leq l \leq N$). At each shift location, the quantity in Equation A.1.17 is computed to form a correlation plane of size $2M+1$ by $2N+1$. This is seen in Figure A.1.4.

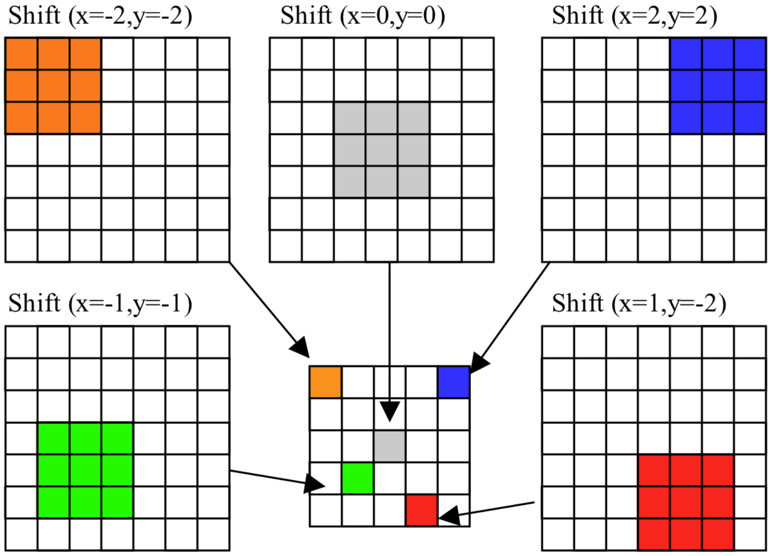


Figure A.1.4: Showing the formation of a correlation plane. A square region of the two acquired images is extracted. The subset of Image 2 is shifted around the subset of Image 1 and the quantity in Equation A.1.17 is computed

For a particularly simple case, the results of this operation is seen in Figure A.1.5.

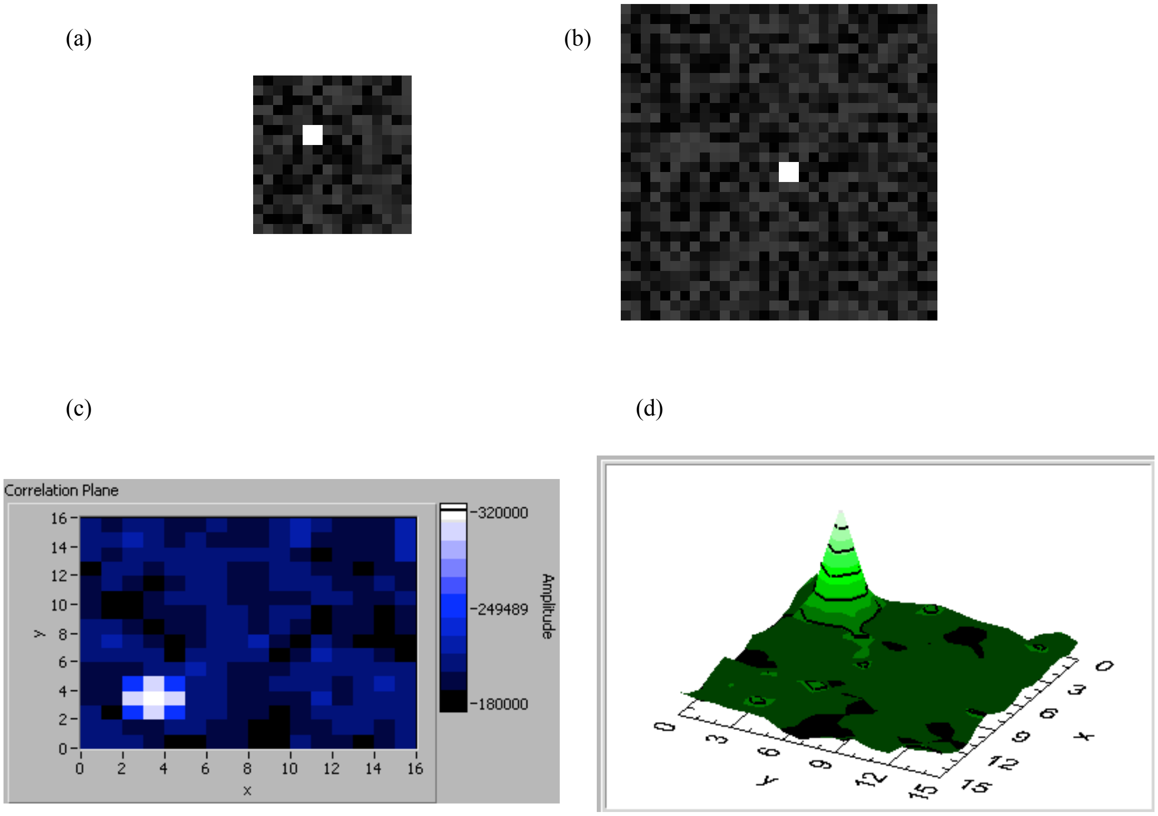


Figure A.1.5: Cross correlation function for simulated images. In (a) and (b), the generated images used in this analysis are seen. Each image contains a tracer particle and random noise that defines the noise floor. The defined shift in the tracer particle was 3 pixels in *x* and *y*. In (c), a top view of the correlation plane is seen, while (d) shows a topographic view of the correlation plane.

There are two points that bear mentioning on the application of the cross correlation technique. First, and as noted previously, it is only possible to recover linear displacements using this technique. To account for this in practice, one must use a sufficiently small Δt and a small enough interrogation region that there are no second order displacements, such as a displacement gradient. Second, this process is computationally intensive process, requiring $O[N^2]$ computations to form a correlation plane of spatial dimension $N \times N$. However, there is a computationally more efficient means of computing the correlation function in Equation A.1.17. This technique makes use of the correlation theorem, which states that cross correlation of two functions, *I* and *I'* is equivalent to a complex conjugate multiplication of their Fourier transforms, or

$$R(x,y) \Leftrightarrow \hat{I} \cdot \hat{I}'^{*} \tag{A.1.18}$$

where $\hat{I}$ and $\hat{I}'^{*}$ are the Fourier transform of *I* and *I'*, respectively. As a result, the cross correlation analysis can be done using Fourier Transforms, which is how the PIV vectors are computed in practice. The proof of Equation A.1.18 is given in the next section.

A1.2: PROOF OF THE CORRELATION THEOREM

In the following, consider two functions: $f(x,y)$ and $g(x,y)$. By definition, the cross correlation of these two functions, $h(\delta x,\delta y)$, is given by Equation A.1.19.

$$h(\delta x,\delta y) = g(x,y) \otimes f(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x,y) \cdot f^{*}(x-\delta x, y-\delta y)\,dxdy \tag{A.1.19}$$

The Fourier Transform of *f(x, y)* and *g(x, y)* are, by definition, given in Equation A.1.20.

$$\begin{gathered} G(p,q) = \frac{1}{4\pi^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x,y)\exp\{-i(px+qy)\}dxdy \\ F(p,q) = \frac{1}{4\pi^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\exp\{-i(px+qy)\}dxdy \end{gathered} \tag{A.1.20}$$

We now consider the convolution of $f(x,y)$ and $g(x,y)$ as the inverse Fourier transform of the product of $G(p,q)$ and $F^*(p,q)$ in Equation A.1. 21.

$$j(\delta x,\delta y) = g \otimes f = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} G(p,q)F^*(p,q)\exp\{i(p\delta x + q\delta y)\}dpdq \qquad \text{(A.1.21)}$$

Substituting the definition of $G(p,q)$ from Equation A.1.20, we find

$$j(\delta x,\delta y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left\{\frac{1}{4\pi^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x,y)\exp\{-i(px + qy)\}dxdy\right\}F^*(p,q)\exp\{i(p\delta x + q\delta y)\}dpdq$$

(A.1.22)

Changing the order of integration,

$$j(\delta x,\delta y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x,y)\left\{\frac{1}{4\pi^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F^*(p,q)\exp\{-i(p(\delta x - x) + q(\delta x - x))\}dpdq\right\}dxdy$$

(A.1.23)

But, $\frac{1}{4\pi^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F^*(p,q)\exp\{-i(p(\delta x - x) + q(\delta x - x))\}dpdq$ is a phase shifted Fourier Transform of $F(p,q)$. Thus,

$$j(\delta x,\delta y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x,y)f(x - \delta x, y - \delta y)dxdy \qquad \text{(A.1.24)}$$

Consequently, $h(\delta x,\delta y)$ and $j(\delta x,\delta y)$ are the same function and the cross correlation of two functions is equivalent to complex conjugate multiplication of the same two functions. This is the correlation theorem.

A1.3: MATHEMATICS OF THE PINHOLE CALIBRATION MODEL

The pinhole model is a mathematical model that describes a camera setup and is defined by eleven parameters. It is based on the theorem of intersecting lines and provides a complete mapping of volume of interest. The basic idea, in two dimensions, of this model is shown in Figure A.1.6.

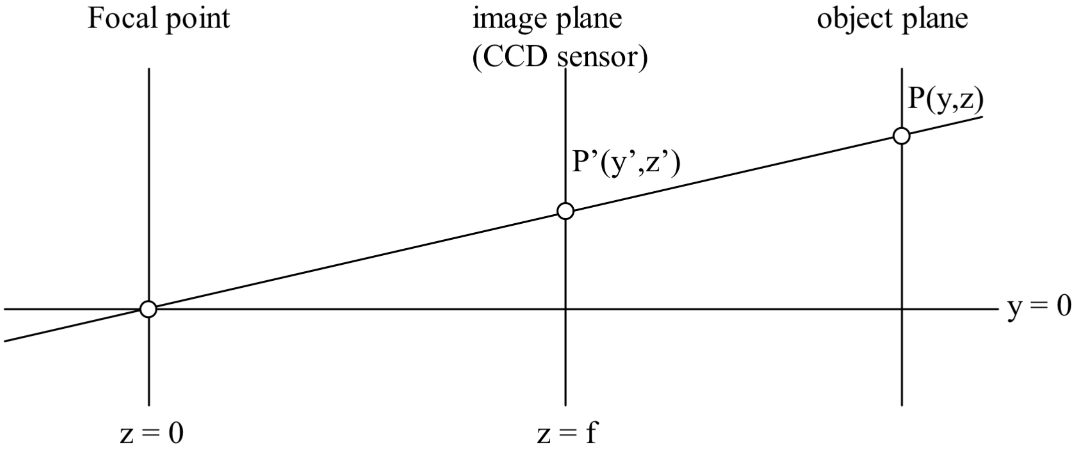


Figure A.1.6: Showing the basic idea of the pinhole model.

Here, a point in the real world at *P(y, z)*, located in the object plane, is projected onto the image plane at location *P'(y', z')*. Based on the geometry of Figure A.1.6, a relationship between the position *P(y, z)* and *P'(y', z')* is given by A.1.25.

$$\frac{y'}{z'} = \frac{y}{z} \quad \Rightarrow \quad y' = f\,\frac{y}{z} \tag{A.1.25}$$

Extending this into three dimensions, one can map a three dimensional point in the global coordinate system, (X_W, Y_W, Z_W) to image coordinates on the CCD array (x, y). For applications to a three dimensional system, an additional transformation is needed. First, the illuminated object, which exist in "global" 3D coordinates (X_W, Y_W, Z_W), must be projected onto a 2D "object plane," (X_C, Y_C, Z_C), which is then imaged by the CCD array in the camera. In order to map a location in the global coordinate system (*i.e.* what is illuminated by the laser sheet) onto the camera coordinate system (*i.e.* what is imaged by the camera), as seen in Figure A.1.7, a rotation and a translation are performed. This is described mathematically in Equation A.1.26.

$$\vec{X}_C = \overset{\leftrightarrow}{R}\ \vec{X}_W + \overset{\leftrightarrow}{T} \tag{A.1. 26}$$

where $\vec{X}_C = \left(X_C, Y_C, Z_C\right)$ is a position in the camera coordinates, $\vec{X}_W = \left(X_W, Y_W, Z_W\right)$ is a position in the global coordinate system, and $\overset{\leftrightarrow}{R}$ and $\overset{\leftrightarrow}{T}$ describe the rotation and translation necessary to move from the global coordinate system and the camera system. It is noted that the rotation and translation here are the six external parameters in the pinhole model used here.

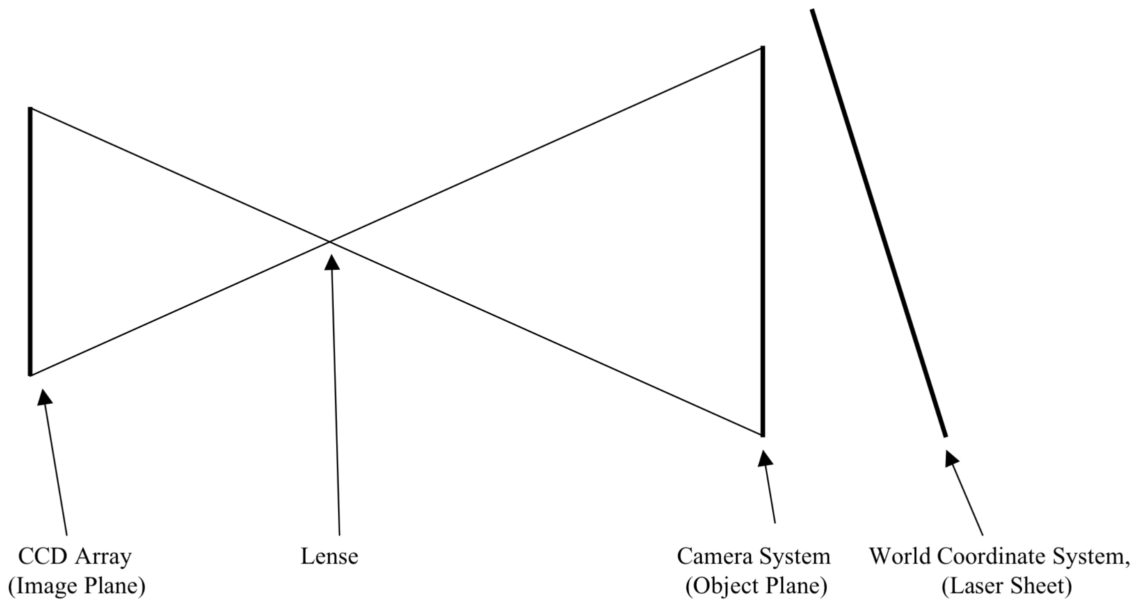


Figure A.1.7: Top view of the 3D pinhole model. Particles are located in the world coordinate system and are illuminated by a laser sheet.

Once the illuminated particles have been projected into camera coordinate system, a generalized version of Equation A.1.25 relates a position in the camera system to where it is projected, assuming that there are no sources of distortion in the system.

$$\begin{aligned} x_u &= f\frac{X_C}{Z_C} \\ y_u &= f\frac{Y_C}{Z_C} \end{aligned} \tag{A.1. 26}$$

where f is related to the focal length of the lens and the distance from the image to the object plane, T_z.

$$f = f_{lens}\left(1 + \frac{f_{lens}}{T_z + f_{lens}}\right) \qquad \text{(A.1.27)}$$

It is noted that the distortion in the system may cause $(x_u,\ y_u)$ to not be the exact location where the image forms on the CCD array. These location are, however, related. The location where the image actually forms on the CCD array is given by

$$\begin{aligned} x &= S_x \frac{x_d}{S_{pixel}} + x_o \\ y &= \frac{y_d}{S_{pixel}} + y_o \end{aligned} \qquad \text{(A.1.28)}$$

where S_x is a distortion parameter, S_{pixel} is the size of a pixel, (x_o, y_o) is the principle point (typically near the center of the CCD array) and (x_d, y_d) are the distorted camera position given by

$$\begin{aligned} x_d &= x_u\left(1 + k_1 r + k_2 r^2\right) \\ y_d &= y_u\left(1 + k_1 r + k_2 r^2\right) \end{aligned} \qquad \text{(A.1.29)}$$

where k_1 and k_2 are radial distortion terms and $r^2 = x_d^{\ 2} + y_d^{\ 2}$. Of these, (f, S_x, k_1, x_o, y_o) define the five internal camera parameters of the pinhole model used.

APPENDIX 2: C++ CODE TO SIMULATE THE PIV MEASUREMENT

Simulations of the PIV measurement were presented in Chapter 3. In this appendix, the basic code that was written in C++ to simulate this measurement technique is presented. Appendix A.2.1 contains a copy of the configuration that was used, while the PIV simulation code is found in Appendix A.2.3 and the code for non-standard libraries used are included in Appendix A.2.4.

A2.1: CONFIGURATION FILE FOR THE PIV SIMULATIONS

```
# This is the configuration file. The '#' at the beginning of a
# line followed by a space if a comment. Anything after this is
# ignored. There cannot be any blank lines either. There is a bug
# in the code that doesn't allow this. Hopefully it'll be fixed in
# the next version.
# **************************************************************
#
# Ion Temperature [eV]:
0.025
#
# Electron Temperature [eV]:
3
#
# Tx Dust Temperature [eV] (must be an integer, for now):
30
#
# Ty Dust Temperature [eV] (must be an integer, for now):
45
#
# Tz of Temperatures for Dust (must be an integer, for now):
310
```

```
#
# Tracer particles have a size distribution(t/f for yes/no):
t
#
#
# Mean radius of dust particle [m]:
0.00000145
#
# Width of dust particle distribution [m]:
0.0000005
#
# Mass density of dust:
2600
#
# Number of electrons on dust particle:
2100
#
# Column dimensions of simulation grid:
400
#
# Row dimensions of simulation grid
400
#
# Intermediate (t) or Low density(f) case:
t
#
# Initial number of particles per bin:
# [0 for low density, 1 for intermediate density]
0
#
# Delta number of particles per bin:
1
#
# Number of iterations for particles per bin:
# [4 for low density, 11 for intermediate density]
11
#
# Ion Density:
5000000000000000
#
# Grid size to use in simulation:
2
```

A2.2: C++ CODE FOR THE PIV SIMULATION

```
/************************************************************************************
*                                                                                   *
*  File: 3DPIV_final.cpp                                                            *
*  Author: Jeremiah Williams                                                        *
*  Written: August 22, 2005                                                         *
*  Last Modified: November 20, 2006 (JW)                                            *
*                                                                                   *
*  Simulates the PIV measurement for monodisperser tracer particles or tracer       *
*  particles having a mass distribution.  This version allows for a temperture      *
*  anisotropy.  This will run both the low density cases (i.e. particle density     *
*  < 0.5 particles per bin) and the intermediate density case (i.e. particle        *
*  density from 0.5 to 5 particles per bin).  The configuration file is:            *
*  "configurations_final.data".                                                     *
*                                                                                   *
************************************************************************************/


#include <assert.h>
#include <time.h>
#include <cstdlib>
#include <ctype.h>
#include <fstream>
#include <iostream>
#include <cmath>
#include <string>
#include <sstream>
#include <iomanip>

#include "StringUtils.h"

// this line is needed to use the standard library
using namespace std;

// function prototypes
double normalDistribution(const double & mean, const double & stdev);
int zero_round(double val);
void matfree(void *matt, int *dim);
void *matloc(int *dim, int cellsize);

//constants
const string CONFIGURATION_FILE = "configurations_final.data";
const double pi_const = 4 * atan(1.0);


int main()
{
   // array[row index][column index][depth][4], where depth is equal to the maximum
   // number of particles and 3 corresponds to the three vector components, i.e. [0]
   // will hold the particles mass, [1] will hold vx, and [2] will hold vy and [3]
   // will hold vz - Recall: indices start at zero for an array.
```

```
//set seed for random number generator based on clock time
srand(static_cast<unsigned>(time(0)));

//define variables to read in from configuration file
double Ti;                  // Ion Temperature
double Te;                  // Electron Temperature
int Td_x;                   // Dust Temperature in x
int Td_y;                   // Dust Temperature in y
int Td_z;                   // Dust Temperature in z
bool size_distribution;     // Include a size distribution
double rdmean;              // mean radius of dust particle distribution
double rdsigma;             // width of dust particle distribution
double rhod;                // mass density of dust
int dustcharge;             // Number of electrons on dust particle
int icolumn;                // dimensions of simulation grid
int irow;
bool type_of_simulation;    // Intermediate or low density case
int Cpart_initial;          // Initial Maximum number of particles per bin
int Cpart_delta;            // Particles per bin Increment
int Cpart_number;           // Particles per bin values run
double ni;                  // ion density
int gs;                     // gride size to use in simulation


// read from configuration file
ifstream configs(CONFIGURATION_FILE.c_str());
string line;
int input_item = 1;
while(getline(configs, line))
{
   istringstream istr(line.c_str());
   string info;
   istr >> info;
   if (info != "#")
   {
      switch (input_item)
      {
         case 1: Ti = StringToDouble(info);
                 input_item++; break;
         case 2: Te = StringToDouble(info);
                 input_item++; break;
         case 3: Td_x = StringToInt(info);
                 input_item++; break;
         case 4: Td_y = StringToInt(info);
                 input_item++; break;
         case 5: Td_z = StringToInt(info);
                 input_item++; break;
         case 6: size_distribution = StringToBool(info);
```

```
                    input_item++; break;
            case 7: rdmean = StringToDouble(info);
                    input_item++; break;
            case 8: rdsigma = StringToDouble(info);
                    input_item++; break;
            case 9: rhod = StringToDouble(info);
                    input_item++; break;
            case 10: dustcharge = StringToInt(info);
                    input_item++; break;
            case 11: icolumn = StringToInt(info);
                    input_item++; break;
            case 12: irow = StringToInt(info);
                    input_item++; break;
            case 13: type_of_simulation = StringToBool(info);
                    input_item++; break;
            case 14: Cpart_initial = StringToInt(info);
                    input_item++; break;
            case 15: Cpart_delta= StringToInt(info);
                    input_item++; break;
            case 16: Cpart_number= StringToInt(info);
                    input_item++; break;
            case 17: ni = StringToDouble(info);
                    input_item++; break;
            case 18: gs = StringToInt(info);
                    input_item++; break;

            // default: // should probably flush the stream
        }
    }
}
configs.close(); // closes configuration file


// Define other constants
double me = 9.11e-31;                    // electron mass
double qe = 1.6e-19;                     // electron charge
double mi = 40 * 1.67e-27;               // ion mass, assuming argon
int overlap = gs / 2;                    // overlap of bins for PIV Simulation

// other constants that may be needed
int Td, Cpart, counter_for_simulation, counter_1, counter_2;

double part_den_array[Cpart_number * 9];  // stores the particle density


if(type_of_simulation)
{
   // use this case for intermediate density cases
   counter_1 = 10;
```

```
}

else
{
   //use this acse for the low density cases
   counter_1 = 1;
}


for(int counter_1 = 1; counter_1 < 10; counter_1++)
{
   for(int counter_Cpart = 0; counter_Cpart < Cpart_number; counter_Cpart++)
   {
      if(type_of_simulation)
      {
         // use this case for intermediate density cases
         counter_for_simulation = counter_1;
         Cpart = Cpart_initial + counter_Cpart * Cpart_delta;
      }

      else
      {
         //use this acse for the low density cases
         counter_for_simulation = Cpart_initial + counter_Cpart * Cpart_delta;
         Cpart = 1;
      }


      // Define other variables
      int npts = icolumn * irow * Cpart;  // number of points for simulation
      double particledensity;             // effective particle density over
                                          // the simulation grid
      double r;                           // radius of particle
      double mdust;                       // mass of the dust
      double vwidth_x;                    // width of velocity distribution in x
      double vwidth_y;                    // width of velocity distribution in y
      double vwidth_z;                    // width of velocity distribution in z
      double total;                       // keep track of the total number of
                                          // particles
      double vxsum;                       // keep track of x velocity in PIV
                                          // simulation region
      double vysum;                       // keep track of y velocity in PIV
                                          // simulation region
      double vzsum;                       // keep track of z velocity in PIV
                                          // simulation region
      double particlecount;               // keep track of the number of particles
                                          // in PIV simulation region
      int temp = npts * 4;
```

```
// Define arrays that will be needed
int partdata[irow][icolumn];                   // stores the number of
                                               // particles in each bin
double PIV_result[irow - 1][icolumn - 1][3]; // stores the PIV velocity
                                               // vectors

// generates the array that will store the raw mass and velocity vector data
// Note: the bigmatrixsizes[n] has 0 terminated array of dimension size
double ****simulation_grid;
int bigmatrixsizes[5]={irow, icolumn, counter_for_simulation + 1, 3, 0};
simulation_grid = matloc(bigmatrixsizes,sizeof(double));


for(int i = 0;  i< irow; i++)
{
   for(int j = 0; j< icolumn; j++)
   {
      if(type_of_simulation)
      {
         //  use this line for the intermediate density cases
         partdata[i][j] = zero_round((float) rand() / RAND_MAX *
                        (counter_for_simulation + ((i + 100 * Cpart) / 500)));
      }

      else
      {
         // use these lines for low density cases
         if(zero_round((float) rand() /
            RAND_MAX * counter_for_simulation) == 0)
         {
            partdata[i][j] = zero_round((float) rand() /
                           RAND_MAX * Cpart);
         }

         else
         {
         partdata[i][j] = 0;
         }
      }
   }
}


// Compute the effective particle density
particledensity = 0;

for(int i = 0;  i< irow; i++)
{
   for(int j = 0; j< icolumn; j++)
   {
      particledensity += partdata[i][j];
   }
}

particledensity /= (irow * icolumn);
part_den_array[counter_Cpart * 9 + counter_for_simulation - 1] =
                     particledensity;


// Assign a mass and velocity information to the particles in the simulation
// grid.  If there are no particles in the bin, a value of zero is used as a
// place holder
for(int i = 0;  i< irow; i++)
{
   for(int j = 0; j< icolumn; j++)
   {
      if(type_of_simulation)
      {
         // use this case for intermediate density cases
         counter_2 = Cpart;
      }

      else
      {
         //use this case for the low density cases
            counter_2 = counter_for_simulation;
      }

      for(int k = 0; k < counter_2; k++)
      {

         if(size_distribution)
         {
            // use this case for cases involving a size distribution generate
            // non-negative radius from normal distribution.  The extra factor
            // in the NormalDistribution function returns a distribution of
            // the desired width.  The factor of 0.707326 is needed to
            // generate a distribution that matches the nominal value.
            r = normalDistribution(rdmean,rdsigma * 0.707326);

            // chop at detection limit of experiment
            while(r < 0.25e-6)
            {
               r = normalDistribution(rdmean,rdsigma * 0.707326);
            }
         }
```

```
            else
            {
               //use this case for the low density cases
               r = rdmean;
            }


            // compute particle mass and width of velocity distribution.
            // The factor of 0.750995 is needed to generate a distribution
            // that matches the nominal value
            mdust = 4/3 * pi_const * pow(r,3) * rhod;
            vwidth_x = sqrt((qe * Td_x * 0.750995) / mdust);
            vwidth_y = sqrt((qe * Td_y * 0.750995) / mdust);
            vwidth_z = sqrt((qe * Td_z * 0.750995) / mdust);


            if(k < partdata[i][j])
            {
               // record particle information (mass and velocity)
               simulation_grid[i][j][k][0] = mdust;
               simulation_grid[i][j][k][1] = normalDistribution(0, vwidth_x);
               simulation_grid[i][j][k][2] = normalDistribution(0, vwidth_y);
               simulation_grid[i][j][k][3] = normalDistribution(0, vwidth_z);
            }

            else
            {
               // record zeros if there are no particles in that location
               // on the simulation grid
               simulation_grid[i][j][k][0] = 0;
               simulation_grid[i][j][k][1] = 0;
               simulation_grid[i][j][k][2] = 0;
               simulation_grid[i][j][k][3] = 0;
            }
         }
      }
   }


   // simulate PIV measurement
   for(int i = 0;  i< irow - 1; i++)
   {
      for(int j = 0; j< icolumn - 1; j++)
      {
         //reset counters
         vxsum = 0;
         vysum = 0;
         vzsum = 0;
         particlecount = 0;
```

```
            // sums all of the velocities in the PIV analysis region
            for(int k = 0; k < counter_2; k++)
            {
               vxsum += simulation_grid[i][j][k][1] + simulation_grid[i][j + 1][k][1]
                        + simulation_grid[i + 1][j][k][1]
                        + simulation_grid[i + 1][j + 1][k][1];
               vysum += simulation_grid[i][j][k][2] + simulation_grid[i][j + 1][k][2]
                        + simulation_grid[i + 1][j][k][2]
                        + simulation_grid[i + 1][j + 1][k][2];
               vzsum += simulation_grid[i][j][k][3] + simulation_grid[i][j + 1][k][3]
                        + simulation_grid[i + 1][j][k][3]
                        + simulation_grid[i + 1][j + 1][k][3];
            }


            // determines the number of particles in the PIV analysis region
            particlecount = partdata[i][j] + partdata[i][j + 1] + partdata[i + 1][j]
                            + partdata[i + 1][j + 1];

            // calculate and store the PIV velocity.  If there were no particles,
            // then a dummy value (9999) is stored in the array and will be used
            // as a flag to prevent this data from being written to file for later
            // analysis.
            if(particlecount == 0)
            {
               PIV_result[i][j][0] = 9999;
               PIV_result[i][j][1] = 9999;
               PIV_result[i][j][2] = 9999;
            }

            else
            {
               PIV_result[i][j][0] = vxsum / particlecount;
               PIV_result[i][j][1] = vysum / particlecount;
               PIV_result[i][j][2] = vzsum / particlecount;
            }
         }
      }


      // Convert the particledensity to a string to include in filename for output
      // file
      string out_to_file;
      stringstream send_to_file;
      send_to_file << particledensity;        // insert double to stream
      send_to_file >> out_to_file;            // read double to string
```

```
//generate filenames
string filename_mass = IntToString(Td_z, 0) + "/mass_" + out_to_file + ".txt";
string filename_velocity = IntToString(Td_z, 0) + "/velocity_" + out_to_file
                           + ".txt";
string filename_piv = IntToString(Td_z, 0) + "/piv_" + out_to_file + ".txt";
string filename_part_dist = IntToString(Td_z, 0) + "/part_dist_"
                            + out_to_file + ".txt";


// Write Data to File
ofstream mass_file(filename_mass.c_str());
assert(mass_file);
for(int i = 0;  i< irow; i++)
{
   for(int j = 0; j< icolumn; j++)
   {
      for(int k = 0; k < partdata[i][j]; k++)
      {
         mass_file << setprecision(25) << simulation_grid[i][j][k][0] << endl;
      }
   }
}
mass_file.close();


ofstream raw_v_file(filename_velocity.c_str());
assert(raw_v_file);
for(int i = 0;  i< irow; i++)
{
   for(int j = 0; j< icolumn; j++)
   {
      for(int k = 0; k < partdata[i][j]; k++)
      {
         raw_v_file  << setprecision(25) << simulation_grid[i][j][k][1]
                  << '\t' << simulation_grid[i][j][k][2] << '\t'
                  << simulation_grid[i][j][k][3] << '\t'
                  << sqrt(pow(simulation_grid[i][j][k][1], 2) +
                  pow(simulation_grid[i][j][k][2], 2) +
                  pow(simulation_grid[i][j][k][3], 2)) << endl;
      }
   }
}
raw_v_file.close();


ofstream piv_file(filename_piv.c_str());
assert(piv_file);
for(int i = 0;  i< irow - 1; i++)
```

```
      {
         for(int j = 0; j< icolumn - 1; j++)
         {
            // only write to file if there was an actual PIV velocity
            if(PIV_result[i][j][2] != 9999)
            {
               piv_file << setprecision(25) << PIV_result[i][j][0] << '\t'
                      << PIV_result[i][j][1] << '\t'<< PIV_result[i][j][2]
                      << '\t' << sqrt(pow(PIV_result[i][j][0], 2)
                      + pow(PIV_result[i][j][1], 2) + pow(PIV_result[i][j][2], 2))
                     << endl;
            }
         }
      }
      piv_file.close();


      ofstream part_dist_file(filename_part_dist.c_str());
      assert(part_dist_file);
      for(int i = 0;  i< irow; i++)
      {
         for(int j = 0; j< icolumn; j++)
         {
            part_dist_file << partdata[i][j] << '\t';
         }
         part_dist_file << endl;
      }
      part_dist_file.close();

      cout << "Tdust = " << Td_z << ":" << '\t' << Cpart << '\t'
           <<  out_to_file <<  endl;
   }


   // Write Particle Density Data to File
   string filename_partden = IntToString(Td_z, 0) + "/particledensity.txt";
   ofstream partden_file(filename_partden.c_str());
   assert(partden_file);
   for(int i = 0;  i < Cpart * 9; i++)
   {
      partden_file << part_den_array[i] << endl;
   }
   partden_file.close();


   // Write Temperature to File
   string filename_tdust = IntToString(Td_z, 0) + "/temperature.txt";
   ofstream tdust_file(filename_tdust.c_str());
   assert(tdust_file);
```

```
      tdust_file << "Tx" << '\t' << "Ty" << '\t' << "Tz" << endl;
      tdust_file << Td_x << '\t' << Td_y << '\t' << Td_z << endl;
      tdust_file.close();
   }


   return(0);
}

//*****************************
double normalDistribution(const double & mean, const double & stdev)
// Use the polar form of the Box-Muller Transformation to generate a random number from
// a normal distribution defined by mean and stdev
{
   double value1, value2, w;

   do
   {
      value1 = 2.0 * rand() / RAND_MAX - 1.0;
      value2 = 2.0 * rand() / RAND_MAX - 1.0;
      w = value1 * value1 + value2 * value2;
   } while ( w >= 1.0 );

   return value2 * sqrt( (-2.0 * log( w ) ) / w ) * stdev + mean;
}

//********************************
int zero_round(double val)
// Rounds to zero based on the decimal value
{
  return (int)(trunc(val+(val>=0.0?0.5:-0.5)));
}

//********************************
void matfree(void *matt, int *dim)
{
   int bob;

   if (!*dim) return;

   if (!*(dim+1)) free(matt);

   else
   {
      for(bob=0;bob<*dim;bob++)
      matfree(((void**)matt)[bob],dim+1);
      free(matt);
   }
}
```

```
//*******************************
void *matloc(int *dim, int cellsize)
{
   void **matt;
   int bob;

   if (!*dim) return 0;

   if (!*(dim+1)) return (void*)malloc(*dim*cellsize);

   matt = (void**)malloc(*dim*sizeof(void*));

   if (!matt) return 0;

   for(bob=0;bob<*dim;bob++)

   if (!(matt[bob]=matloc(dim+1,cellsize)))
   {
      while (bob--)
      matfree(matt[bob],dim+1);
      free(matt);
      return 0;
   }

   return (void*)matt;
}
```

A2.3: Additional Libraries

/Users/williams/Desktop/jw/Code/C++/3DPIV/StringUtils.h

```
/*************************************************************************************
*                                                                                    *
*  File: StringUtils.h                                                               *
*  Last Modified: July 6, 1998 (DK)                                                  *
*  More recent Modification: October 25, 2002 (JW)                                   *
*                                                                                    *
*  Comments:                                                                         *
*     added string IntToString(const int & num, const bool & plusSign = false)       *
*     added int StringToInt(const string & str)                                      *
*     added string FloatToString(float num)                                          *
*     added int StringToFloat(const string & str)                                    *
*     added string TimeToString(tm & time)                                           *
*     added tm StringToTime(const string & str) *** not tested!!!                    *
*     added bool StringToBool(const string & str) *** not tested!!!                  *
*     added string BoolToString(const bool & state) *** not tested!!!                *
*                                                                                    *
*************************************************************************************/

#ifndef _String_Utils_
   #define _String_Utils_

   #include <string>
   #include <time.h>

   using std::string;   // needed to compile using Mac OS 10.2 and later


   /* This function compares two strings alphabetically. If the first string is less
      than the second it returns 1.  If the second string is less it returns a 2.
      If the strings areequal it returns a 0.  The strings can be compared as case-
      sensitive or not.                                                             */
   int StringCompare(const string & s1, const string & s2, bool caseSensitive);


   /* This functions parses the string s, from pos till it reaches the delimiter
      (delim) or the end of the string.  If the end of the string is reached, pos is
      set to 0                                                                      */
   string StringParse(const string & s,const string & delim,int &pos);


   /* This function converts an integer into a string.  It basically has the opposite
      effect of atoi. If you want the '+' to be there on positive numbers, make
      plusSign true                                                                 */
   string IntToString(const int & num, const bool & plusSign = false);


   /* This function converts a string to an integer. It uses the atoi function, but
      it also handles the cases if there is a '+' or '-' in front of the number. atoi
      would return 0 in either of these cases.                                      */
   int StringToInt(const string &str);



   /* This function converts a float into a string.  It basically has the opposite
   effect of atof. If you want the '+' to be there on positive numbers, make plusSign
   true.                                                                          */
// string FloatToString(float num, const bool & plusSign = false);
   string FloatToString(const float & num, const bool & plusSign = false);


   /* This function converts a string to a float. It uses the atof function, but it
      also handles the cases if there is a '+' or '-' in front of the number. atof
      would return 0 in either of these cases.                                    */
   float StringToFloat(const string & str);

   /* This function converts a string to a double. It uses the atof function, but it
      also handles the cases if there is a '+' or '-' in front of the number. atof
      would return 0 in either of these cases.                                    */
   double StringToDouble(const string & str);


   /* This function tries to parse out a bool value. It looks the string to start with
      either 't' or 'f' or tries to parse out an int, which is converted to a bool, 0
      is false, greater than 0 is true.                                           */
   bool StringToBool(const string & str);


   /* This function converts a bool value to a string. false is converted into "false"
      and true is converted into "true".                                          */
   string BoolToString(const bool & state);


   /* This function converts the tm struct found in time.h (ctime) into a string of the
      form "xy/xy/yyyy xy:yy:yy" where the x's are optional values depending on the
      number and y are required numbers. For example June 3, 1998 3:01:00 PM would be
      6/3/1998 15:01:00.  Note the use of military time.                          */
    string TimeToString(const tm & theDate);


   /* This function converts str into a tm struct found in time.h (ctime). The form of
      the string must be "xy/xy/yyyy xy:yy:yy" where the x's are optional values
      depending on the number and y are required numbers. For example June 3, 1998
      3:01:00 PM would be 6/3/1998 15:01:00. Note the use of military time. This does
      not fill in all the fields of the struct, only the ones we need for this
      project                                                                     */
   tm StringToTime(string theDate);
   /* This function returns the current time in the format that we are using of a
      string.                                                                   */
   string GetCurrentTime();
#endif
```

```
/*********************************************************************************
*                                                                               *
*  File: StringUtils.cpp                                                        *
*  Last Modified: July 24, 1998 (DK)                                            *
*                                                                               *
*********************************************************************************/

#include <string>
#include <string.h>
#include <ctype.h> // to convert to C style strings
#include <iostream.h>
#include <cstdlib> // for int abs(int)
#include <time.h>
#include <ctime>
#include <cstdlib>
#include <stdio.h>
#include <fstream.h>

#include "StringUtils.h"

const int ZERO = 0;

int StringCompare(const string &s1, const string &s2,bool  caseSensitive)
{
   int length;

   if (s1.length() > s2.length())
      length = s2.length();
   else
      length = s1.length();

   for (int i=0;i<length;i++)
   {
      if (caseSensitive)
      {
         if (s1[i] > s2[i])
            return 2;
         else if (s2[i] > s1[i])
            return 1;
         else
            continue;

      }
      else
      {
         if (toupper(s1[i]) > toupper(s2[i]))
            return 2;
         else if (toupper(s2[i]) > toupper(s1[i]))
            return 1;
         else
            continue;
      }

   }
   if (s1.length() > s2.length())
      return 2;
   else if (s1.length() < s2.length())
      return 1;
   else
      return 0;


}


string StringParse(const string &s,const string &delim,int &pos)
{
   int start;

   // Deal with the problems now.
   if (pos < 0)
      return "";
   if (pos >= s.length())
   {
      pos = -1;
      return "";
   }
   // If we are at the begining, there is no delimiter first, so we just go to the first
   // delimiter
   if (pos == 0)
   {
      start = pos;
      pos = s.find(delim,start);
   }
   //Since we are at a delimiter, we must search from beyond it
   else
   {
      start = pos + delim.length();
      pos = s.find(delim,start);
   }
   return  s.substr(start, (pos - start));
}



string IntToString(const int & num, const bool & plusSign)
{
    char buffer[20];
   string str;
   //first add the plus sign if needed
   if ((num > 0) && (plusSign))
      sprintf(buffer,"+%d",num);
   else
      sprintf(buffer,"%d",num);
   str = buffer;
   return str;
//*** change this to be the same as the float to string
/* string str;
   int offset = 0;
   if (num == 0) { str.append(1, '0'); return str; }
   else if (num < 0) { str.insert(offset, 1, '-'); offset = 1;}
   else if ((num > 0) && plusSign) { str.insert(offset, 1, '+'); offset = 1;}
   int modBy = 10;
   int divBy = 1;
   while (abs(num) >= divBy)
   {
      // 48 = '0'
      str.insert(offset, 1, ((abs(num) % modBy) / divBy) + 48 );
      modBy *= 10;
      divBy *= 10;
   }
   return str;
*/
}


int StringToInt(const string & str)
{
   int num;
   // if '+' or '-', then length has to be at least 2
   if ( (str.length() > 1) && (str[0] == '-') )
     { num = (atoi( (str.substr(1)).c_str() )) * -1; }
   else if ( (str.length() > 1) && (str[0] == '+') )
     { num = atoi( (str.substr(1)).c_str() ); }
   else // this is if it's just a positive number
     { num = atoi(str.c_str()); }
   return num;
}


string FloatToString(const float & num, const bool & plusSign)
{
   char buffer[50];
   string str;
   //first add the plus sign if needed
   if ((num > 0) && (plusSign))
```

```
      sprintf(buffer,"+%g",num);
   else
      sprintf(buffer,"%g",num);
   str = buffer;
   return str;
}


float StringToFloat(const string & str)
{
   float num;
   // if '+' or '-', then length has to be at least 2
   if ( (str.length() > 1) && (str[0] == '-') )
     { num = float((atof( (str.substr(1)).c_str() )) * -1); }
   else if ( (str.length() > 1) && (str[0] == '+') )
     { num = float(atof( (str.substr(1)).c_str() )); }
   else // this is if it's just a positive number
     { num = float(atof(str.c_str())); }
   return num;
}

double StringToDouble(const string & str)
{
   double num;
   // if '+' or '-', then length has to be at least 2
   if ( (str.length() > 1) && (str[0] == '-') )
     { num = (atof( (str.substr(1)).c_str() )) * -1; }
   else if ( (str.length() > 1) && (str[0] == '+') )
     { num = atof( (str.substr(1)).c_str() ); }
   else // this is if it's just a positive number
     { num = atof(str.c_str()); }
   return num;
}


bool StringToBool(const string & str)
{
   bool state;
   if ((str.length() > 0) && tolower(str[0]) == 't')
     { state = true; }
   else if ((str.length() > 0) && tolower(str[0]) == 'f')
     { state = false; }
   else if (StringToInt(str) == 0)
     { state = false; }
   else if (StringToInt(str) > 0)
     { state = true; }
   return state;
}
```

```
string BoolToString(const bool & state)
{
   string str;
   if (state) { str = "true"; }
   else { str = "false"; }
   return str;
}


string TimeToString(const tm & theDate)
{
    char dateCstr[20];
    strftime(dateCstr,20,"%m/%d/%Y %H:%M:%S",&theDate);
    string dateString(dateCstr);
    return dateString;
}


tm StringToTime(string theDate)
{
    tm dateStruct;
    sscanf( &theDate[0],"%d/%d/%d %d:%d:%d",
            &dateStruct.tm_mon,
            &dateStruct.tm_mday,
            &dateStruct.tm_year,
            &dateStruct.tm_hour,
            &dateStruct.tm_min,
            &dateStruct.tm_sec);

    //The tm struct stores year-1900.
    dateStruct.tm_year = dateStruct.tm_year - 1900;

    // The tm struct seems to store month-1???
    // I couldn't find any documentation on this but that is
    // the way it seems to work on both CW and g++.
    dateStruct.tm_mon--;

    return dateStruct;
}


string GetCurrentTime()
{
    time_t t;
    time(&t);
    tm *time = localtime(&t);
    return TimeToString(*time);
}
```

APPENDIX 3: IGOR MACROS

A3.1: DESCRIPTION OF IGOR

IGOR Pro is an application for the presentation and analysis of data. This application, published by WaveMetrics, Inc., has a powerful scripting language. In this appendix, we present the code that was written to analyze both the experimental and simulation data. Appendix A.3.2, we present the code used to analyze the simulation data having a uniform size, while the code for analyzing the simulation results involving a distribution of microparticle sizes is found in Appendix A.3.3. Appendix A.3.4 contains the code that was used to analyze the experimental data, while Appendix A3.5 contains the code that was used to fit data to a 1D drifting Maxwellian velocity distribution function.

A3.2: ANALYSIS CODE FOR SIMULATION DATA WITH MONODISPERSE PARTICLES

```
#pragma rtGlobals=1        // Use modern global access method.

Function Simulation_uniform_particles(file_vel_Head,file_piv_Head,filestart,fileend, num_of_bins,
                                        rho, r_d)
    // Reads in data files containing velocity vectors and extracts the temperature for simulation data
    // having a uniform mass.  The data files should be sequentially numbered.  The base file name for
    // files containing the underlying velocity distribution data are defined by the variable
    // "file_vel_Head", while the base file name for files containing the PIV velocity distribution data
    // are defined by the variable "file_piv_Head".  The numbering starts with "filestart" and ends with
    // "fileend".  The code does not allow for zero padding and requires that the files have a ".txt"
    //  extenstion.  For each file, the data should be stored in tab delimited text files having the format:
    //
    //  vx  vy  vz
    //
    // The remaining variables are defined below:
    //
    //      r_d:                radius of the dust grain (assuming monodisperse, spherical dust)
    //      rho:                mass denisty of the dust in [kg/m^3]
    //      num_of_bins:        number of bins to use when generating the distributions
    //
    // An example would be as follows:
    //
    //  Simulation_uniform_particles("vel","piv",1, 46, 100, 3800, 0.6e-6)
    //
    // Here the underlying velocity distribution  data is stored in files names: vel1.txt, vel2.txt, . . .,
    // vel46.txt and the PIV distribution  data is stored in files names: piv1.txt, piv2.txt, . . ., piv46.txt.
    // For analysis, the particles here have a radius of 0.6 μm and a mass density of 3800.  The
    // distributions that are generated will have 101 bins.  All of the data files need to be in the same
    // directory as this file and the information about the particle denisty should be stored in the same
    // directory and be titled "particledensity.txt"


    // define the variables that are fed into the function variables
    String file_vel_Head, file_piv_Head //Header for file
    Variable filestart, fileend, num_of_bins, rho, r_d

    PathInfo/S home                                 // Get path of local directory
    NewPath/O path1 S_path                          // Set up path name for use in Igor file load
    SetDataFolder root:                             // return to the root folder
    KillWaves/A                                     // start with some housecleaning


    // Define constants that will be used in the code
    Variable k_b = 1.3806503e-23                    //Boltzmann's constant
    Variable m_d = 4/3 * Pi * (r_d)^3 * rho         // Compute the mass of the dust
    Variable gamma =  m_d / (2 * k_B)               // Compute a constant for the fit
    Variable count = 0                              // Counter for use in the code
    Variable file                                   // Remaining variables to be used
```

```
// Generate a folder to hold the results of this analysis - this should prevent confusion with later
// analysis
NewDataFolder/O $"uniformdistribution"

// Waves to store the temperature that is extracted from the fit
Make/D/N=(fileend - filestart + 1) Tx_vel, sigma_Tx_vel, Ty_vel, sigma_Ty_vel
Make/D/N=(fileend - filestart + 1) Tz_vel, sigma_Tz_vel
Make/D/N=(fileend - filestart + 1) Tx_piv, sigma_Tx_piv, Ty_piv, sigma_Ty_piv
Make/D/N=(fileend - filestart + 1) Tz_piv, sigma_Tz_piv
Make/D/N=(fileend - filestart + 1) Tx_norm, Ty_norm, Tz_norm
Make/D/N=(fileend - filestart + 1) vx_drift_vel, vy_drift_vel, vz_drift_vel
Make/D/N=(fileend - filestart + 1) sigma_vx_drift_vel, sigma_vy_drift_vel, sigma_vz_drift_vel
Make/D/N=(fileend - filestart + 1) vx_drift_piv, vy_drift_piv, vz_drift_piv
Make/D/N=(fileend - filestart + 1) sigma_vx_drift_piv, sigma_vy_drift_piv, sigma_vz_drift_piv

SetDataFolder root:                                 // Make sure that we are in the root
                                                    // directory

/// Read in a file containing the particle denisty for each simulation run.
LoadWave/Q/A='data'/G/D/L={0, 0, 0, 0, 1}/P=path1 "particledensity.txt"
if (V_flag==0)       // No waves loaded. Perhaps user canceled.
    return -1
endif

PathInfo/S home                                     // Get path of local directory


Rename data0,x_axis                                 // Rename the waves using more meaningful
Duplicate/O x_axis, root:uniformdistribution:x_axis      // names

// Read in each file using a loop.  Each file is loaded into separate data folders
for (file=filestart;file<=fileend;file+=1)

    // Generate the filename to read in and the name of the folder to hold the data
    String file_vel_Name = file_vel_Head+num2istr(file)+".txt"
    String file_piv_Name = file_piv_Head+num2istr(file)+".txt"
    String folderName = "set"+num2istr(file)

    // Generate folder structure structure to store the data and move to that folder
    NewDataFolder/O $folderName
    SetDataFolder folderName


    // load the velocity file to a wave
    LoadWave/Q/A='data'/G/D/L={0, 0, 0, 0, 3}/P=path1 file_vel_Name
    if (V_flag==0)       // No waves loaded. Perhaps user canceled.
        return -1
    endif
```

```
Rename data0,vx_vel_data                                    // Rename the waves using
Rename data1,vy_vel_data                                    // more meaningful names
Rename data2,vz_vel_data


// Generate histogram for the raw data
Make/D/N=(num_of_bins)  histo_vx_vel,histo_vy_vel,          // Generate waves to store
                        histo_vz_vel                        // distribution functions
Histogram/B=1  vx_vel_data, histo_vx_vel                    // Generate the distribution
Histogram/B=1  vy_vel_data, histo_vy_vel                    // functions
Histogram/B=1  vz_vel_data, histo_vz_vel


// Generates a wave to store the initial fit parameters
Make/D/N=5/O  W_initial_1                                   // Generate waves to store
                                                            // the fit parameters


// Fit the x-component of the velocity to a Gaussian - the fit parameters will be used as seed
// values for the 1D drifting Maxwellian fit function
CurveFit/Q gauss histo_vx_vel /D
Wave W_coef                                                 // Needed to access the fit
                                                            // parameters
KillWaves   W_sigma                                         // Needed to extract
                                                            // uncertainties in the fit


// Compute the seed values to fit the x-component of the velocity to a 1D drifting Maxwellian
W_initial_1[0] = gamma * W_coef[3]^2
W_initial_1[1]  =  gamma^(3/2)
W_initial_1[2]  =  gamma
W_initial_1[3]  =  W_coef[0]
W_initial_1[4]  =  W_coef[2]

// Fit the x-component to a 1D drifting Maxwellian
FuncFit/Q/H="001"  Speed1Ddrift  W_initial_1  histo_vx_vel/D

// Extract the temperature and uncertainty from the fit and save in the appropriate wave
// Currently the width is extracted, a conversion will be added later
Wave W_sigma
root:Tx_vel[count]  =  W_initial_1[0]  /  11600
root:sigma_Tx_vel[count] = W_sigma[0] / 11600
root:vx_drift_vel[count]  =  W_initial_1[4]
root:sigma_vx_drift_vel[count] = W_sigma[4]
KillWaves   W_sigma
```

```
// Fit the y-component of the velocity to a Gaussian - the fit parameters will be used as seed
// values for the 1D drifting Maxwellian fit function
CurveFit/Q gauss histo_vy__vel /D
Wave W_coef                                                 // Needed to access the fit
                                                            // parameters
KillWaves  W_sigma                                          // Needed to extract
                                                            // uncertainties in the fit


// Compute the seed values to fit the y-component of the velocity to a 1D drifting Maxwellian
W_initial_1[0] = gamma * W_coef[3]^2
W_initial_1[1] = gamma^(3/2)
W_initial_1[2] = gamma
W_initial_1[3] = W_coef[0]
W_initial_1[4] = W_coef[2]

// Fit the y-component to a 1D drifting Maxwellian
FuncFit/Q/H="001" Speed1Ddrift W_initial_1 histo_vy_vel/D

// Extract the temperature and uncertainty from the fit and save in the appropriate wave
// Currently the width is extracted, a conversion will be added later
Wave W_sigma
root:Ty_vel[count] = W_initial_1[0] / 11600
root:sigma_Ty_vel[count] = W_sigma[0] / 11600
root:vy_drift_vel[count] = W_initial_1[4]
root:sigma_vy_drift_vel[count] = W_sigma[4]
KillWaves  W_sigma


// Fit the z-component of the velocity to a Gaussian - the fit parameters will be used as seed
// values for the 1D drifting Maxwellian fit function
CurveFit/Q gauss histo_vz /D
Wave W_coef                                                 // Needed to access the fit
                                                            // parameters
KillWaves  W_sigma                                          // Needed to extract
                                                            // uncertainties in the fit


// Compute the seed values to fit the x-component of the velocity to a 1D drifting Maxwellian
W_initial_1[0] = gamma * W_coef[3]^2
W_initial_1[1] = gamma^(3/2)
W_initial_1[2] = gamma
W_initial_1[3] = W_coef[0]
W_initial_1[4] = W_coef[2]


// Fit the z-component to a 1D drifting Maxwellian
FuncFit/Q/H="001" Speed1Ddrift W_initial_1 histo_vz_vel/D
```

```
// Extract the temperature and uncertainty from the fit and save in the appropriate wave
// Currently the width is extracted, a conversion will be added later
Wave W_sigma
root:Tz_vel[count] = W_initial_1[0] / 11600
root:sigma_Tz_vel[count] = W_sigma[0] / 11600
root:vz_drift_vel[count] = W_initial_1[4]
root:sigma_vz_drift_vel[count] = W_sigma[4]
KillWaves  W_sigma


// load the PIV velocity file to a wave
LoadWave/Q/A='data'/G/D/L={0, 0, 0, 0, 3}/P=path1 file_piv_Name
if (V_flag==0)      // No waves loaded. Perhaps user canceled.
    return -1
endif

Rename data0,vx_piv_data                                  // Rename the waves using
Rename data1,vy_piv_data                                  // more meaningful names
Rename data2,vz_piv_data


// Generate histogram for the piv data
Make/D/N=(num_of_bins) histo_vx_piv, histo_vy_piv,        // Generate waves to store
                       histo_vz_piv                       // the distribution functions
Histogram/B=1 vx_piv_data, histo_vx_piv                   // Generate the distribution
Histogram/B=1 vy_piv_data, histo_vy_piv                   // functions
Histogram/B=1 vz_piv_data, histo_vz_piv


// Fit the x-component of the velocity to a Gaussian - the fit parameters will be used as seed
// values for the 1D drifting Maxwellian fit function
CurveFit/Q gauss histo_vx_piv /D
Wave W_coef                                               // Needed to access the fit
                                                          // parameters
KillWaves  W_sigma                                        // Needed to extract
                                                          // uncertainties in the fit


// Compute the seed values to fit the x-component of the velocity to a 1D drifting Maxwellian
W_initial_1[0] = gamma * W_coef[3]^2
W_initial_1[1] = gamma^(3/2)
W_initial_1[2] = gamma
W_initial_1[3] = W_coef[0]
W_initial_1[4] = W_coef[2]

// Fit the x-component to a 1D drifting Maxwellian
FuncFit/Q/H="001" Speed1Ddrift W_initial_1 histo_vx_piv/D
```

```
// Extract the temperature and uncertainty from the fit and save in the appropriate wave
// Currently the width is extracted, a conversion will be added later
Wave W_sigma
root:Tx_piv[count] = W_initial_1[0] / 11600
root:sigma_Tx_piv[count] = W_sigma[0] / 11600
root:vx_drift_piv[count] = W_initial_1[4]
root:sigma_vx_drift_piv[count] = W_sigma[4]
KillWaves  W_sigma


// Fit the y-component of the velocity to a Gaussian - the fit parameters will be used as seed
// values for the 1D drifting Maxwellian fit function
CurveFit/Q gauss histo_vy_piv /D
Wave W_coef                                            // Needed to access the fit
                                                       // parameters
KillWaves  W_sigma                                     // Needed to extract
                                                       // uncertainties in the fit


// Compute the seed values to fit the y-component of the velocity to a 1D drifting Maxwellian
W_initial_1[0] = gamma * W_coef[3]^2
W_initial_1[1] = gamma^(3/2)
W_initial_1[2] = gamma
W_initial_1[3] = W_coef[0]
W_initial_1[4] = W_coef[2]

// Fit the y-component to a 1D drifting Maxwellian
FuncFit/Q/H="001" Speed1Ddrift W_initial_1 histo_vy_piv/D

// Extract the temperature and uncertainty from the fit and save in the appropriate wave
// Currently the width is extracted, a conversion will be added later
Wave W_sigma
root:Ty_piv[count] = W_initial_1[0] / 11600
root:sigma_Ty_piv[count] = W_sigma[0] / 11600
root:vy_drift_piv[count] = W_initial_1[4]
root:sigma_vy_drift_piv[count] = W_sigma[4]
KillWaves  W_sigma


// Fit the z-component of the velocity to a Gaussian - the fit parameters will be used as seed
// values for the 1D drifting Maxwellian fit function
CurveFit/Q gauss histo_vz_piv /D
Wave W_coef                                            // Needed to access the fit
                                                       // parameters
KillWaves  W_sigma                                     // Needed to extract
                                                       // uncertainties in the fit
```

```
        // Compute the seed values to fit the x-component of the velocity to a 1D drifting Maxwellian
        W_initial_1[0] = gamma * W_coef[3]^2
        W_initial_1[1] = gamma^(3/2)
        W_initial_1[2] = gamma
        W_initial_1[3] = W_coef[0]
        W_initial_1[4] = W_coef[2]

        // Fit the z-component to a 1D drifting Maxwellian
        FuncFit/Q/H="001" Speed1Ddrift W_initial_1 histo_vz_piv/D


        // Extract the temperature and uncertainty from the fit and save in the appropriate wave
        // Currently the width is extracted, a conversion will be added later
        Wave W_sigma
        root:Tz_piv[count] = W_initial_1[0] / 11600
        root:sigma_Tz_piv[count] = W_sigma[0] / 11600
        root:vz_drift_piv[count] = W_initial_1[4]
        root:sigma_vz_drift_pivl[count] = W_sigma[4]
        KillWaves  W_sigma


        SetDataFolder root:                                         // Return to root folder
        Print count                                                 // Display the iteration number
        count += 1                                                  // for the user and increment

    endfor


    // Move waves to the desired folder
    MoveWave root:Tx_vel,root:uniformdistribution:
    MoveWave root:sigma_Tx_vel,root:uniformdistribution:
    MoveWave root:Ty_vel,root:uniformdistribution:
    MoveWave root:sigma_Ty_vel,root:uniformdistribution:
    MoveWave root:Tz_vel,root:uniformdistribution:
    MoveWave root:sigma_Tz_vel,root:uniformdistribution:
    MoveWave root:Tx_piv,root:uniformdistribution:
    MoveWave root:sigma_Tx_piv,root:uniformdistribution:
    MoveWave root:Ty_piv,root:uniformdistribution:
    MoveWave root:sigma_Ty_piv,root:uniformdistribution:
    MoveWave root:Tz_piv,root:uniformdistribution:
    MoveWave root:sigma_Tz_piv,root:uniformdistribution:
    MoveWave root:Tx_norm,root:uniformdistribution:
    MoveWave root:Ty_norm,root:uniformdistribution:
    MoveWave root:Tz_norm,root:uniformdistribution:
    MoveWave root:vx_drift_vel,root:uniformdistribution:
    MoveWave root:sigma_vx_drift_vel,root:uniformdistribution:
    MoveWave root:vy_drift_vel,root:uniformdistribution:
    MoveWave root:sigma_vy_drift_vel,root:uniformdistribution:
    MoveWave root:vz_drift_vel,root:uniformdistribution:
```

```
	MoveWave  root:sigma_vz_drift_vel,root:uniformdistribution:
	MoveWave  root:vx_drift_piv,root:uniformdistribution:
	MoveWave  root:sigma_vx_drift_piv,root:uniformdistribution:
	MoveWave  root:vy_drift_piv,root:uniformdistribution:
	MoveWave  root:sigma_vy_drift_piv,root:uniformdistribution:
	MoveWave  root:vz_drift_piv,root:uniformdistribution:
	MoveWave  root:sigma_vz_drift_piv,root:uniformdistribution:

	SetDataFolder $"root:uniformdistribution"


	Tx_norm = Tx_piv / Tx_vel							// Compute the ratio of widths
	Ty_norm = Ty_piv / Ty_vel
	Tz_norm = Tz_piv / Tz_vel

	Edit x_axis, Tx_norm, Ty_norm, Tz_norm					// Display the ratio of widths in
											// a table


	Display  Tz_norm,Ty_norm,Tx_norm vs x_axis				// Plot the ratio of widths as a
	Label left "Normalized Temperature";DelayUpdate			// function of the particle
	Label bottom "Particle Density [particles / bin]";DelayUpdate	// density
	SetAxis left 0,1 ;DelayUpdate
	SetAxis bottom 0,5
	ModifyGraph  mode=3,rgb(Ty_norm)=(0,65535,0),rgb(Tx_norm)=(0,0,65535)

End
```

A3.3: ANALYSIS CODE FOR SIMULATION DATA WITH POLYDISPERSE PARTICLES

```
#pragma rtGlobals=1         // Use modern global access method.

Function Simulation_polydisperse_particles(fileHead_mass, fileHead_vel, fileHead_piv,filestart,
                                            fileend, histo_low, histo_high, num_of_bins, rmin,
                                            rmax, num_of_iter, rho)
   // Reads in data files containing velocity vectors and extracts the temperature for simulation data
   // having a uniform mass.  The data files should be sequentially numbered.  The base file name for
   // files containing the underlying velocity distribution data are defined by the variable
   // "file_vel_Head", while the base file name for files containing the PIV velocity distribution data
   // are defined by the variable "file_piv_Head" and the base file name for files containing the the
   // mass data are defined by the variable "fileHead_mass."  The numbering starts with "filestart"
   // and ends with "fileend".  The code does not allow for zero padding and requires that the files have
   // a ".txt" extenstion.  For each file, the data should be stored in tab delimited text files having the
   // format:
   //
   //  vx  vy  vz
   //
   // The mass file should contain a single column of data.  The remaining variables are defined below:
   //
   //       histo_low:
   //       histo_high:
   //       r_min:              the smallest radius of the dust grain, typically the diffraction limit
   //                           (i.e. the wavelength of the laser / 2.
   //       r_max:              the largest radius of the dust grain, typically the nominal mean of the
   //                           size distribution plus 9.5 times the nominal dispersion of the size
   //                           distribution
   //       num_of_iter:        number of interations to use between r_min and r_max, typically 145
   //       rho:                mass denisty of the dust in [kg/m^3]
   //       num_of_bins:        number of bins to use when generating the distributions
   //
   // An example would be as follows:
   //
   //  Simulation_polydisperse_particles("mass", "vel", "piv",1,47, -0.025, 0.025, 100, 0.25e-6,
   //                                    10.125e-6, 145, 2600)
   //
   // Here the underlying velocity distribution  data is stored in files names: vel1.txt, vel2.txt, . . .,
   // vel46.txt and the PIV distribution  data is stored in files names: piv1.txt, piv2.txt, . . ., piv46.txt.
   // The mass data is stored in files named: mass1.txt, mass2.txt, . . ., mass46.txt.   All of the data files
   // need to be in the same directory as this file and the information about the particle denisty should
   // be stored in the same directory and be titled "particledensity.txt"


   // define the variables that are fed into the function variables
   String fileHead_mass, fileHead_vel, fileHead_piv
   Variable filestart, fileend, histo_low, histo_high, num_of_bins, rmin, rmax, num_of_iter, rho
```

```
    PathInfo/S home                                         // Get path of local directory
    NewPath/O path1 S_path                                  // Set up path name for use in Igor
                                                            // file load
    SetDataFolder root:                                     // return to the root folder
    KillWaves/A                                             // start with some housecleaning


    // Define constants that will be used in the code
    Variable k_b = 1.3806503e-23                            //Boltzmann's constant
    Variable delta_r = (rmax-rmin) / num_of_iter            // increment for the step size used
                                                            // when moving through the size
                                                            //  distribution
    Variable delta_histo = (histo_high - histo_low) / num_of_bins // the bin size when generating the
                                                            // distribution function
    Variable counter = 0                                    // Counter for use in the code
    Variable count, file                                    // Remaining variables to be used


    // Generate a folder to hold the results of this analysis  and move to that folder - this should prevent
    // confusion with later analysis
    String folderName = "massdistribution"
    NewDataFolder/O $folderName
    SetDataFolder folderName


    // Waves to store the temperature that is extracted from the fit
    Make/D/N=(fileend - filestart + 1) Tx_rdist_vel, sigma_rdist_Tx_vel, Ty_rdist_vel
    Make/D/N=(fileend - filestart + 1) sigma_rdist_Ty_vel, Tz_rdist_vel, sigma_rdist_Tz_vel
    Make/D/N=(fileend - filestart + 1) Tx_rdist_piv, sigma_Tx_rdist_piv, Ty_rdist_piv
    Make/D/N=(fileend - filestart + 1) sigma_Ty_rdist_piv, Tz_rdist_piv, sigma_Tz_rdist_piv

    // Waves to store the drift velocity that is extracted from the fit
    Make/D/N=(fileend - filestart + 1) vx_rdist_vel, sigma_rdist_vx_vel, vy_rdist_vel
    Make/D/N=(fileend - filestart + 1) sigma_rdist_vy_vel, vz_rdist_vel, sigma_rdist_vz_vel
    Make/D/N=(fileend - filestart + 1) vx_rdist_piv, sigma_rdist_vx_piv, vy_rdist_piv
    Make/D/N=(fileend - filestart + 1) sigma_rdist_vy_piv, vz_rdist_piv, sigma_rdist_vz_piv

    // Waves to store the normalized temperatures
    Make/D/N=(fileend - filestart + 1) Tx_norm, Ty_norm, Tz_norm


    SetDataFolder root:                                     // Make sure that we are in the root
                                                            //  directory

    // Read in a file containing the particle denisty for each simulation run.
    LoadWave/Q/A='data'/G/D/L={0, 0, 0, 0, 1}/P=path1 "particledensity.txt"
    if (V_flag==0)        // No waves loaded. Perhaps user canceled.
        return -1
    endif
```

```
Rename data0,x_axis                                        // Rename the waves using more
Duplicate/O x_axis, root:massdistribution:x_axis           // meaningful names


// Read in each file using a loop.  Each file is loaded into separate data folders
for  (file=filestart;file<=fileend;file+=1)

    // Variables to store the temperature as it is being computed
    Variable alpha = 0
    Variable Tx_dummy = 0
    Variable Ty_dummy = 0
    Variable Tz_dummy = 0
    Variable alpha_sum = 0


    // Generate the filename to read in and the name of the folder to hold the data
    String fileName_mass = fileHead_mass+num2istr(file)+".txt"
    String fileName_vel = fileHead_vel+num2istr(file)+".txt"
    String fileName_piv = fileHead_piv+num2istr(file)+".txt"
    folderName = "set"+num2istr(file)


    // Generate folder structure structure to store the data and move to that folder
    NewDataFolder/O $folderName
    SetDataFolder folderName


    // Load the mass file to a wave
    LoadWave/Q/A='data'/G/D/L={0, 0, 0, 0, 1}/P=path1 fileName_mass
    if (V_flag==0)      // No waves loaded. Perhaps user canceled.
        return -1
    endif


    // Convert mass data to radius and rename the wave
    Rename data0,radius_data
    Wave radius_data
    radius_data = ((3 * radius_data) / (4 * Pi * rho))^(1/3)


    // Generate size distribution for the particle size and fit to a Gaussian
    Make/D/N=(num_of_bins) histo_rad
    Histogram/B=1 radius_data, histo_rad
    CurveFit/Q gauss histo_rad /D


    // Write the fit coefficients to a wave for later use
    Rename W_coef,r_coef
    Wave r_coef
```

```
// Load the velocity file to a wave
LoadWave/Q/A='data'/G/D/L={0, 0, 0, 0, 3}/P=path1 fileName_vel
if (V_flag==0)      // No waves loaded. Perhaps user canceled.
    return -1
endif

Rename data0,vx_vel_data                                 // Rename the waves using
Rename data1,vy_vel_data                                 // more meaningful names
Rename data2,vz_vel_data


// Generate histogram for the raw data
Make/D/N=(num_of_bins + 1) histo_vx_vel,histo_vy_vel,histo_vz_vel
Histogram/B={histo_low, delta_histo, num_of_bins + 1} vx_vel_data, histo_vx_vel
Histogram/B={histo_low, delta_histo, num_of_bins + 1} vy_vel_data, histo_vy_vel
Histogram/B={histo_low, delta_histo, num_of_bins + 1} vz_vel_data, histo_vz_vel


//  Generate a wave to store radii to use in the analysis.
Make/D/N=(num_of_iter) radius

//  Generate a wave to store weighting factor to be used to compute a temperature
// using a weighted average
Make/D/N=(num_of_iter) alpha_array

// Generate a wave to store the fit parameters
Make/D/N=3/O W_initial
Make/D/N=3/O W_sigma

// Waves to save the results of the fit at each size used from the size distribution
Make/D/N=(num_of_iter) Tx_vel, sigma_Tx_vel, Ty_vel, sigma_Ty_vel
Make/D/N=(num_of_iter) Tz_vel, sigma_Tz_vel
Make/D/N=(num_of_iter) vx_rdist_vel, sigma_rdist_vx_vel, vy_rdist_vel
Make/D/N=(num_of_iter) sigma_rdist_vy_vel, vz_rdist_vel, sigma_rdist_vz_vel

// Compute a weighted average for the temperature
for (count= 0;count!=num_of_iter;count+=1)

    // Define remaining constants
    Variable r_d = rmin + count * delta_r                // radius of the particle
    Variable m_d = 4/3 * Pi * (r_d)^3 * rho              // mass of the particle
    Variable gamma =  m_d / (2 * k_B)                    // constant to use in the fit


    // Calculated the weighting factor that is needed to compute the average temperature.
    alpha = r_coef[1] * exp(-((r_d - r_coef[2]) / r_coef[3])^2)
    alpha_sum = alpha_sum + alpha
```

```
Make/D/N=5/O  W_initial_1                                    // Generate waves to store
                                                             // the fit parameters


// Fit the x-component of the velocity to a Gaussian - the fit parameters will be used as seed
// values for the 1D drifting Maxwellian fit function
CurveFit/Q gauss histo_vx_vel /D
Wave W_coef                                                  // Needed to access the fit
                                                             // parameters
KillWaves   W_sigma                                          // Needed to extract
                                                             // uncertainties in the fit


// Fit the x-component of the velocity to a Gaussian - the fit parameters will be used as seed
// values for the 1D drifting Maxwellian fit function
CurveFit/Q gauss histo_vx_vel /D
Wave W_coef                                                  // Needed to access the fit
                                                             // parameters
KillWaves   W_sigma                                          // Needed to extract
                                                             // uncertainties in the fit


// Compute the seed values to fit the x-component of the velocity to a 1D drifting Maxwellian
W_initial_1[0] = gamma * W_coef[3]^2
W_initial_1[1]  =  gamma^(3/2)
W_initial_1[2]  =  gamma
W_initial_1[3]  =  W_coef[0]
W_initial_1[4]  =  W_coef[2]


// Fit the x-component to a 1D drifting Maxwellian
FuncFit/Q/H="001"  Speed1Ddrift W_initial_1  histo_vx_vel/D

// Extract the temperature and uncertainty from the fit and save in the appropriate wave
// Currently the width is extracted, a conversion will be added later
root:Tx_vel[count]  =  W_initial_1[0]  /  11600
root:sigma_Tx_vel[count] = W_sigma[0] / 11600
root:vx_drift_vel[count]  =  W_initial_1[4]
root:sigma_vx_drift_vel[count] = W_sigma[4]
KillWaves   W_sigma


// Fit the y-component of the velocity to a Gaussian - the fit parameters will be used as seed
// values for the 1D drifting Maxwellian fit function
CurveFit/Q gauss histo_vy__vel /D
Wave W_coef                                                  // Needed to access the fit
                                                             // parameters
KillWaves   W_sigma                                          // Needed to extract
                                                             // uncertainties in the fit
// Compute the seed values to fit the y-component of the velocity to a 1D drifting Maxwellian
W_initial_1[0] = gamma * W_coef[3]^2
W_initial_1[1] = gamma^(3/2)
W_initial_1[2] = gamma
W_initial_1[3] = W_coef[0]
W_initial_1[4] = W_coef[2]


// Fit the y-component to a 1D drifting Maxwellian
FuncFit/Q/H="001" Speed1Ddrift W_initial_1 histo_vy_vel/D

// Extract the temperature and uncertainty from the fit and save in the appropriate wave
// Currently the width is extracted, a conversion will be added later
root:Ty_vel[count] = W_initial_1[0] / 11600
root:sigma_Ty_vel[count] = W_sigma[0] / 11600
root:vy_drift_vel[count] = W_initial_1[4]
root:sigma_vy_drift_vel[count] = W_sigma[4]
KillWaves  W_sigma


// Fit the z-component of the velocity to a Gaussian - the fit parameters will be used as seed
// values for the 1D drifting Maxwellian fit function
CurveFit/Q gauss histo_vz /D
Wave W_coef                                          // Needed to access the fit
                                                     // parameters
KillWaves  W_sigma                                   // Needed to extract
                                                     // uncertainties in the fit


// Compute the seed values to fit the x-component of the velocity to a 1D drifting Maxwellian
W_initial_1[0] = gamma * W_coef[3]^2
W_initial_1[1] = gamma^(3/2)
W_initial_1[2] = gamma
W_initial_1[3] = W_coef[0]
W_initial_1[4] = W_coef[2]


// Fit the z-component to a 1D drifting Maxwellian
FuncFit/Q/H="001" Speed1Ddrift W_initial_1 histo_vz_vel/D

// Extract the temperature and uncertainty from the fit and save in the appropriate wave
// Currently the width is extracted, a conversion will be added later
root:Tz_vel[count] = W_initial_1[0] / 11600
root:sigma_Tz_vel[count] = W_sigma[0] / 11600
root:vz_drift_vel[count] = W_initial_1[4]
root:sigma_vz_drift_vel[count] = W_sigma[4]
KillWaves  W_sigma
```

```
		//Save Results to a wave for later display
		radius[count] = r_d
		alpha_array[count] = alpha

		// Calculated the next term in the temperature weighting function, needed to determine
		// the average temperature.
		Tx_dummy += alpha * Tx_vel[count]
		Ty_dummy += alpha * Ty_vel[count]
		Tz_dummy += alpha * Tz_vel[count]

	endfor

	//finishes the average temperature calculation and appends the result to the table
	root:Tx_rdist_vel[counter] = Tx_dummy / alpha_sum
	root:Ty_rdist_vel[counter] = Ty_dummy / alpha_sum
	root:Tz_rdist_vel[counter] = Tz_dummy / alpha_sum


	Tx_dummy = 0								// Reset the temperature
	Ty_dummy = 0								//  variables
	Tz_dummy = 0


	// load the PIV velocity file to a wave
	LoadWave/Q/A='data'/G/D/L={0, 0, 0, 0, 3}/P=path1 file_piv_Name
	if (V_flag==0)		// No waves loaded. Perhaps user canceled.
		return -1
	endif

	Rename data0,vx_piv_data						// Rename the waves using
	Rename data1,vy_piv_data						// more meaningful names
	Rename data2,vz_piv_data


	// Generate histogram for the piv data
	Make/D/N=(num_of_bins) histo_vx_piv, histo_vy_piv,		// Generate waves to store
				histo_vz_piv				// the distribution functions
	Histogram/B=1 vx_piv_data, histo_vx_piv				// Generate the distribution
	Histogram/B=1 vy_piv_data, histo_vy_piv				// functions
	Histogram/B=1 vz_piv_data, histo_vz_piv


	// Make waves to store the results of the analysis on the PIV data
	Make/D/N=(num_of_iter) Tx_piv, sigma_Tx_piv, Ty_piv, sigma_Ty_piv
	Make/D/N=(num_of_iter) Tz_piv, sigma_Tz_piv
```

```
for (count= 0;count!=num_of_iter;count+=1)

     // Define remaining constants
     Variable r_d = radius[count]                                  // radius of the particle
     Variable m_d = 4/3 * Pi * (r_d)^3 * rho                       // mass of the particle
     Variable gamma =  m_d / (2 * k_B)                             // constant to use in the fit


     // Calculated the weighting factor that is needed to compute the average temperature.
     alpha = alpha_array[count]
     alpha_sum = alpha_sum + alpha


     // Fit the x-component of the velocity to a Gaussian - the fit parameters will be used as seed
     // values for the 1D drifting Maxwellian fit function
     CurveFit/Q gauss histo_vx_piv /D
     Wave W_coef                                                   // Needed to access the fit
                                                                   // parameters
     KillWaves  W_sigma                                            // Needed to extract
                                                                   // uncertainties in the fit


     // Compute the seed values to fit the x-component of the velocity to a 1D drifting Maxwellian
     W_initial_1[0] = gamma * W_coef[3]^2
     W_initial_1[1]  =  gamma^(3/2)
     W_initial_1[2]  =  gamma
     W_initial_1[3]  =  W_coef[0]
     W_initial_1[4]  =  W_coef[2]

     // Fit the x-component to a 1D drifting Maxwellian
     FuncFit/Q/H="001"  Speed1Ddrift  W_initial_1  histo_vx_piv/D


     // Extract the temperature and uncertainty from the fit and save in the appropriate wave
     // Currently the width is extracted, a conversion will be added later
     root:Tx_piv[count]  =  W_initial_1[0]  /  11600
     root:sigma_Tx_piv[count]  =  W_sigma[0]  /  11600
     root:vx_drift_piv[count]  =  W_initial_1[4]
     root:sigma_vx_drift_piv[count]  =  W_sigma[4]
     KillWaves  W_sigma


     // Fit the y-component of the velocity to a Gaussian - the fit parameters will be used as seed
     // values for the 1D drifting Maxwellian fit function
     CurveFit/Q gauss histo_vy_piv /D
     Wave W_coef                                                // Needed to access the fit
                                                                // parameters
     KillWaves  W_sigma                                         // Needed to extract
                                                                // uncertainties in the fit
// Compute the seed values to fit the y-component of the velocity to a 1D drifting Maxwellian
W_initial_1[0] = gamma * W_coef[3]^2
W_initial_1[1] = gamma^(3/2)
W_initial_1[2] = gamma
W_initial_1[3] = W_coef[0]
W_initial_1[4] = W_coef[2]

// Fit the y-component to a 1D drifting Maxwellian
FuncFit/Q/H="001" Speed1Ddrift W_initial_1 histo_vy_piv/D

// Extract the temperature and uncertainty from the fit and save in the appropriate wave
// Currently the width is extracted, a conversion will be added later
root:Ty_piv[count] = W_initial_1[0] / 11600
root:sigma_Ty_piv[count] = W_sigma[0] / 11600
root:vy_drift_piv[count] = W_initial_1[4]
root:sigma_vy_drift_piv[count] = W_sigma[4]
KillWaves  W_sigma


// Fit the z-component of the velocity to a Gaussian - the fit parameters will be used as seed
// values for the 1D drifting Maxwellian fit function
CurveFit/Q gauss histo_vz_piv /D
Wave W_coef                                          // Needed to access the fit
                                                     // parameters
KillWaves  W_sigma                                   // Needed to extract
                                                     // uncertainties in the fit



// Compute the seed values to fit the x-component of the velocity to a 1D drifting Maxwellian
W_initial_1[0] = gamma * W_coef[3]^2
W_initial_1[1] = gamma^(3/2)
W_initial_1[2] = gamma
W_initial_1[3] = W_coef[0]
W_initial_1[4] = W_coef[2]

// Fit the z-component to a 1D drifting Maxwellian
FuncFit/Q/H="001" Speed1Ddrift W_initial_1 histo_vz_piv/D


// Extract the temperature and uncertainty from the fit and save in the appropriate wave
// Currently the width is extracted, a conversion will be added later
Wave W_sigma
root:Tz_piv[count] = W_initial_1[0] / 11600
root:sigma_Tz_piv[count] = W_sigma[0] / 11600
root:vz_drift_piv[count] = W_initial_1[4]
root:sigma_vz_drift_pivl[count] = W_sigma[4]
KillWaves  W_sigma
```

```
            // Calculated the next term in the temperature weighting function, needed to determine
            // the average temperature.
            Tx_dummy += alpha * Tx_piv[count]
            Ty_dummy += alpha * Ty_piv[count]
            Tz_dummy += alpha * Tz_piv[count]

        endfor

        //finishes the average temperature calculation and appends the result to the table
        root:Tx_rdist_piv[counter] = Tx_dummy / alpha_sum
        root:Ty_rdist_piv[counter] = Ty_dummy / alpha_sum
        root:Tz_rdist_piv[counter] = Tz_dummy / alpha_sum


        SetDataFolder root:                                     // Return to root folder
        Print count                                             // Display the iteration number
        count += 1                                              // for the user and increment


        // Compute the ratio of widths
        Tx_norm[counter] = Tx_rdist_piv[counter] / Tx_rdist_vel[counter]
        Ty_norm[counter] = Ty_rdist_piv[counter] / Ty_rdist_vel[counter]
        Tz_norm[counter] = Tz_rdist_piv[counter] / Tz_rdist_vel[counter]


    endfor


    Edit x_axis, Tx_norm, Ty_norm, Tz_norm                      // Display the ratio of widths in
                                                                // a table

End
```

A3.4: ANALYSIS CODE FOR EXPERIMENTAL DATA

```
#pragma rtGlobals=1        // Use modern global access method.

Function Process_Data(file_data_Head, filestart, fileend, r_d, rho, error_vx, error_vy, error_vz,
                      num_of_bins, file_correction_factor)
    // Reads in data files containing velocity vectors and extracts the temperature.
    // The data files should be sequentially numbered.  The base file name is defined by the variable
    // "file_data_Head", with the numbering starting with "filestart" and ending with "fileend".
    // The code does not allow for zero padding and requires that the files have a ".txt" extenstion.
    // The data should be stored in tab delimited text files having the format:
    //
    // x-pos    y-pos   z-pos   vx  vy  vz  vmag
    //
    // The PIV corrective factor and density should be stored in a second tab delimited text file defined
    // by the variable "file_correction_factor", where the extension entered by the user.  The file should
    // have the following format:
    //
    // Cloud      PIV factor   Density
    //
    // The remaining variables are defined below:
    //
    //       r_d:                radius of the dust grain (assuming monodisperse, spherical dust)
    //       rho:                mass denisty of the dust in [kg/m^3]
    //       error_vx:           resolution limit of the stereo-PIV system in the x-direction
    //       error_vy:           resolution limit of the stereo-PIV system in the y-direction
    //       error_vz:           resolution limit of the stereo-PIV system in the z-direction
    //       num_of_bins:        number of bins to use when generating the distributions
    //
    //
    // An example would be as follows:
    //
    //Process_Data("Set ",1,8, 0.6e-6, 3800, 0.0037547, 0.0032762, 0.00679,101, "Factor.txt")
    //
    // Here the velocity data is stored in files names: Set 1.txt,Set 2.txt, . . ., Set 8.txt.  The
    // PIV correction data and density would be stores in a file named Factor.txt.  For analysis,
    // the particles here have a radius of 0.6 µm and a mass density of 3800.  The results from
    // the zero displacement test are 0.0037547 in x, 0.0032762 in y and 0.00679 in z.  The
    // distributions that are generated will have 101 bins.  All of the data files need to be in the same
    // directory as this file.

    // define the variables that are the user defines when running the code
    String file_data_Head, file_correction_factor
    Variable filestart, fileend, r_d, rho, num_of_bins, error_vx, error_vy, error_vz

    PathInfo/S home                              // Get path of local directory
    NewPath/O path1 S_path                       // Set up path name for use in Igor file load
    SetDataFolder root:                          // return to the root folder
    KillWaves/A                                  // start with some housecleaning
```

```
// Define constants that will be used in the code
Variable k_b = 1.3806503e-23                    //Boltzmann's constant
Variable m_d = 4/3 * Pi * (r_d)^3 * rho         // Compute the mass of the dust
Variable gamma =  m_d / (2 * k_B)               // Compute a constant for the fit
Variable count = 0                              // Counter for use in the code
Variable T, file                                // Remaining variables to be used

// Generate waves to store the results that come from the fit of the data
Make/D/N=(fileend - filestart + 1) Tx, sigma_Tx, Ty, sigma_Ty, Tz, sigma_Tz,Tmag, sigma_Tmag
Make/D/N=(fileend - filestart + 1) vx_drift, vy_drift, vz_drift, sigma_vx_drift, sigma_vy_drift
Make/D/N=(fileend - filestart + 1) sigma_vz_drift, lost_vy, lost_vz, lost_vmag, n_vectors
Make/D/N=(fileend - filestart + 1) widthvx, widthvy, widthvz


// Read in each file using a loop - each file is loaded into separate data folders
for (file = filestart; file <= fileend; file += 1)


    // Generate the filename to read in and the name of the folder to hold the data
    String file_data_Name = file_data_Head+num2istr(file)+".txt"
    String folderName = "set"+num2istr(file)

    // Generate folder structure structure to store the data and move to that folder
    NewDataFolder/O $folderName
    SetDataFolder folderName


    // load the velocity file to a wave
    LoadWave/Q/A='data'/G/D/L={0, 0, 0, 0, 4}/P=path1 file_data_Name
    if (V_flag==0)      // No waves loaded. Perhaps user canceled.
        return -1
    endif

    Rename data0,vx_data                                        // Rename the waves using
    Rename data1,vy_data                                        // more meaningful names
    Rename data2,vz_data
    Rename data3,vmag_data;


    Make/D/N=(num_of_bins + 1) histo_vx,histo_vy, histo_vz      // Generate waves to store
    Make/D/N=(2 * num_of_bins + 1) histo_vmag                   // the distribution functions

    Histogram/B = {-0.1,0.002,101} vx_data, histo_vx            // Generate the distribution
    Histogram/B = {-0.1,0.002,101} vy_data,histo_vy             // functions
    Histogram/B = {-0.2,0.004,101} vz_data,histo_vz
    Histogram/B = {0,0.001,201} vmag_data,histo_vmag
```

```
// Compute some relevant statistics - number of data points and the percentage of the data that was
// not included in the histogram
Wavestats/Q vx_data
root:lost_vy[count] = 1 - Sum(histo_vy, -1,1) / V_npnts
root:lost_vz[count] = 1 - Sum(histo_vz, -1,1) / V_npnts
root:lost_vmag[count] = 1 - Sum(histo_vmag, -1,1) / V_npnts
root:n_vectors[count] = V_npnts


Make/D/N=4/O  W_initial                                          // Generate waves to store
Make/D/N=5/O  W_initial_1                                        // the fit parameters

// Fit the x-component of the velocity to a Gaussian - the fit parameters will be used as seed
// values for the 1D drifting Maxwellian fit function
CurveFit/Q gauss histo_vx /D
Wave W_coef                                                      // Needed to access the fit
                                                                 // parameters
KillWaves  W_sigma                                               // Needed to extract
                                                                 // uncertainties in the fit


// Compute the seed values to fit the x-component of the velocity to a 1D drifting Maxwellian
W_initial_1[0] = W_coef[3]
W_initial_1[1] = W_coef[1] * sqrt(W_coef[3])
W_initial_1[2] = W_coef[0]
W_initial_1[3] = W_coef[2]

// Fit the x-component to a 1D drifting Maxwellian
FuncFit/Q/H="00001" Speed1Ddrift W_initial_1 histo_vx/D

// Extract the temperature in the x-direction, accounting for the resolution limit of the
// experimental setup
T = gamma * (W_initial_1[0]^2 -  error_vx^2)


Wave W_sigma                                                     // Needed to access the fit
                                                                 // parameters

widthvx[count] = W_initial_1[0]                                  // Write the relevant
root:Tx[count] = T/11600                                         // valued to a wave
root:sigma_Tx[count] = (2 * gamma * W_initial_1[0]  * W_sigma[0]) /
                       (11600 * sqrt(1-(error_vx  /  W_initial_1[0])^2))
root:vx_drift[count] = W_initial_1[3]
root:sigma_vx_drift[count] = W_sigma[3]

KillWaves  W_sigma                                               // Free up this wave, so it ca
                                                                 // be used again
// Fit the y-component of the velocity to a Gaussian - the fit parameters will be used as seed
// values for the 1D drifting Maxwellian fit function
CurveFit/Q gauss histo_vy /D
Wave W_coef                                              // Needed to access the fit
                                                         // parameters
KillWaves  W_sigma                                       // Needed to extract
                                                         // uncertainties in the fit


// Compute the seed values to fit the x-component of the velocity to a 1D drifting Maxwellian
W_initial_1[0] = W_coef[3]
W_initial_1[1] = W_coef[1] * sqrt(W_coef[3])
W_initial_1[2] = W_coef[0]
W_initial_1[3] = W_coef[2]

// Fit the x-component to a 1D drifting Maxwellian
FuncFit/Q/H="00001" Speed1Ddrift W_initial_1 histo_vy/D

// Extract the temperature in the x-direction, accounting for the resolution limit of the
// experimental setup
T = gamma * (W_initial_1[0]^2 -  error_vy^2)


Wave W_sigma                                             // Needed to access the fit
                                                         // parameters


widthvy[count] = W_initial_1[0]                          // Write the relevant
root:Ty[count] = T/11600                                 // values to a wave
root:sigma_Ty[count] = (2 * gamma * W_initial_1[0]  * W_sigma[0]) /
                     (11600 * sqrt(1-(error_vy  /  W_initial_1[0])^2))
root:vy_drift[count] = W_initial_1[3]
root:sigma_vy_drift[count] = W_sigma[3]

KillWaves  W_sigma                                       // Free up this wave, so
                                                         // it can be used again


// Fit the z-component of the velocity to a Gaussian - the fit parameters will be used as seed
// values for the 1D drifting Maxwellian fit function
CurveFit/Q gauss histo_vz /D
Wave W_coef                                              // Needed to access the fit
                                                         // parameters
KillWaves  W_sigma                                       // Needed to extract
                                                         // uncertainties in the fit


// Compute the seed values to fit the x-component of the velocity to a 1D drifting Maxwellian
W_initial_1[0] = W_coef[3]
W_initial_1[1] = W_coef[1] * sqrt(W_coef[3])
```

```
        W_initial_1[2] = W_coef[0]
        W_initial_1[3] = W_coef[2]

        // Fit the x-component to a 1D drifting Maxwellian
        FuncFit/Q/H="00001" Speed1Ddrift W_initial_1 histo_vz/D

        // Extract the temperature in the x-direction, accounting for the resolution limit of the expreimer
        // setup
        T = gamma * (W_initial_1[0]^2 -  error_vz^2)


        Wave W_sigma                                                    // Needed to access the fit
                                                                        // parameters

        widthvz[count] = W_initial_1[0]                                 // Write the relevant values
        root:Tz[count] = T/11600                                        // to a wave
        root:sigma_Tz[count] = (2 * gamma * W_initial_1[0]  * W_sigma[0]) /
                               (11600 * sqrt(1-(error_vz / W_initial_1[0])^2))
        root:vz_drift[count] = W_initial_1[3]
        root:sigma_vz_drift[count] = W_sigma[3]

        KillWaves  W_sigma                                              // Free up this wave, so it ca
                                                                        // be used again


        SetDataFolder root:                                             // Return to root folder
        Print count                                                     // Display the iteration numl
        count += 1                                                      // for the user and incremen
                                                                        // counter
    endfor


    Make/N=(count)/D Tz_fraction = sigma_Tz / Tz                        // Compute the fraction
    Make/N=(count)/D Ty_fraction = sigma_Ty / Ty                        // uncertainties in the
    Make/N=(count)/D Tx_fraction = sigma_Tx / Tx                        // temperatures

    // Read in correction factor to account for PIV narrowing effect
    LoadWave/Q/A='data'/G/D/L={0, 0, 0, 1, 3}/P=path1 file_correction_factor
    if (V_flag==0)      // No waves loaded. Perhaps user canceled.
        return -1
    endif

    Rename data0,Factor                                                 // Rename the waves using
    Rename data1,sigma_Factor                                           // more meaningful names
    Rename data2,Factor_fraction


    Wave Factor                                                         // Needed for Igor to know th
    Wave Factor_fraction                                                // the wave exists
    Make/N=(count)/D Tx_corrected = Tx * Factor                    // Compute the corrected
    Make/N=(count)/D Ty_corrected = Ty * Factor                    // temperature
    Make/N=(count)/D Tz_corrected = Tz * Factor


    // Compute the uncertainty in the temperature
    Make/N=(count)/D sigma_Tz_corrected = Tz_corrected * sqrt(Factor_fraction^2 + Tz_fraction^2)
    Make/N=(count)/D Fraction_Tz_corrected = sigma_Tz_corrected / Tz_corrected
    Make/N=(count)/D sigma_Ty_corrected = Ty_corrected * sqrt(Factor_fraction^2 + Ty_fraction^2)
    Make/N=(count)/D Fraction_Ty_corrected = sigma_Ty_corrected / Ty_corrected
    Make/N=(count)/D sigma_Tx_corrected = Tx_corrected * sqrt(Factor_fraction^2 + Tx_fraction^2)
    Make/N=(count)/D Fraction_Tx_corrected = sigma_Tx_corrected / Tx_corrected


    //Appends the results to the table
    Edit  vx_drift, sigma_vx_drift, vy_drift, sigma_vy_drift, vz_drift, sigma_vz_drift
    Edit lost_vy, lost_vz, n_vectors
    Edit Tx, sigma_Tx, Ty, sigma_Ty, Tz, sigma_Tz
    Edit widthvx, widthvy, widthvz
End
```

A3.5: 1D DRIFTING MAXWELLIAN FUNCTION FIT CODE

```
#pragma rtGlobals=1         // Use modern global access method.

Function Speed1Ddrift(w,x) : FitFunc
// 1D Speed Distribution function fit with drift
    Wave w
    Variable x

    //CurveFitDialog/ These comments were created by the Curve Fitting dialog. Altering them will
    //CurveFitDialog/ make the function less convenient to work with in the Curve Fitting dialog.
    //CurveFitDialog/ Equation:
    //CurveFitDialog/ f(x) = Amplitude * width^(-1/2) * Exp(-((x-vdrift)/width)^2 ) + offset
    //CurveFitDialog/ End of Equation
    //CurveFitDialog/ Independent Variables 1
    //CurveFitDialog/ x
    //CurveFitDialog/ Coefficients 4
    //CurveFitDialog/ w[0] = width
    //CurveFitDialog/ w[1] = Amplitude
    //CurveFitDialog/ w[2] = offset
    //CurveFitDialog/ w[3] = vdrift

    return w[1] * w[0]^(-1/2) * Exp(-((x-w[3])/w[0])^2 ) + w[2]
End
```

APPENDIX 4: LABVIEW CODE

A4.1: DESCRIPTION OF LABVIEW

LabVIEW is a general-purpose graphical programming environment that includes all of the standard features of text-based programming languages. However, unlike these text based programming languages, *LabVIEW* uses a graphical programming language, known as *G*, to create programs known as “virtual instruments”, or “VIs.” It is published by National Instruments.

A VI consists of two parts, the front panel and the block diagram. The front panel provides a customizable interactive user interface, while the block diagram provides a graphical representation of the program code. National Instruments provides numerous libraries of functions and sub-routines for the rapid development of code for data acquisition, analysis and presentation.

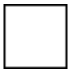
The VI uses a hierarchical and modular structure. This means that the top-level VI may contain a number of sub-VI’s, small applications that perform a specific function not initially provided by National Instruments.

There were many applications that were written in *LabVIEW* for use in this dissertation. The code found in Appendix A4.2 allows the user to access and run the code used in this dissertation to reduce the PIV vectors generated by *DaVis* and analyze the video data to determine the particle density of the microparticle component. The code used to reduce and the raw PIV vectors at each z-location is found Appendix A4.3, while the code used to compile the PIV vectors at each z-location into a single cloud is found in Appendix A4.4. The code used to determine the scattering efficiency, η, is found Appendix A4.5, while the code that is used to extract the factor relating the distribution of velocities measured using the PIV technique to the underlying velocity space distribution function can be found in Appendix A4.6. The code used to measure the number density of the dust clouds can be found Appendix A4.7 and the code used to generate a velocity distribution using only a portion of the acquired data is found in Appendix 4.8.

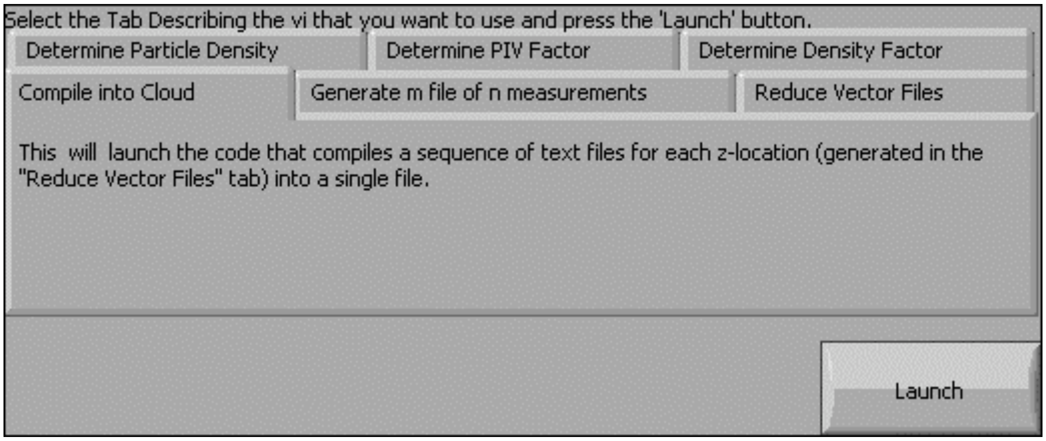
A4.2: MASTER LABVIEW CODE

This vi will launch all of the code that is related to the data processing that is done for the measurements of the distribution function for a dusty plasma.

Connector Pane

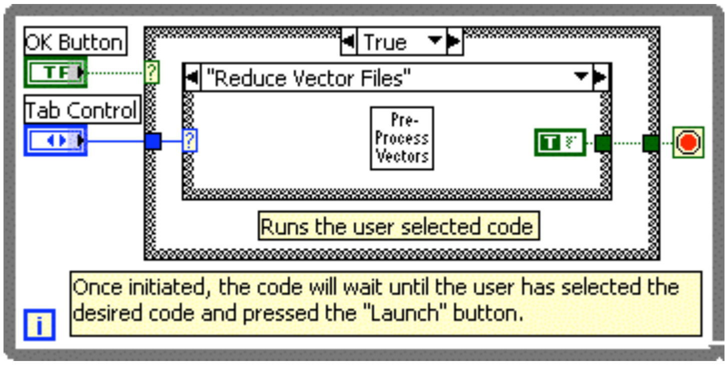


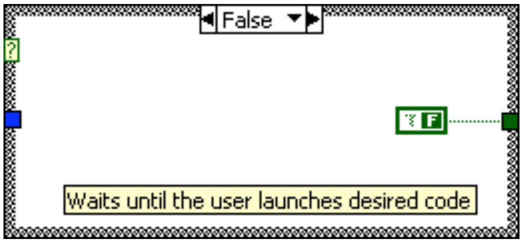
Front Panel

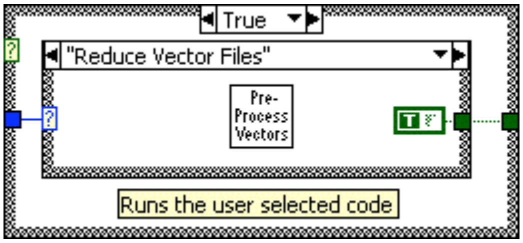

Select the Tab Describing the vi that you want to use and press the 'Launch' button.

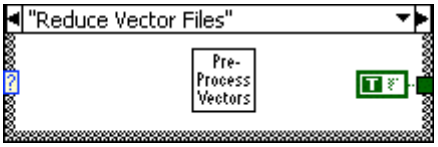
This will launch the code that compiles a sequence of text files for each z-location (generated in the "Reduce Vector Files" tab) into a single file.

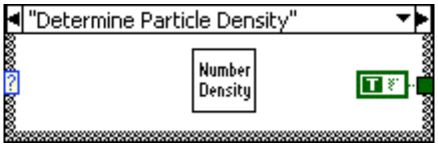
Block Diagram

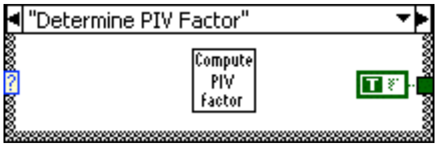

Once initiated, the code will wait until the user has selected the desired code and pressed the "Launch" button.

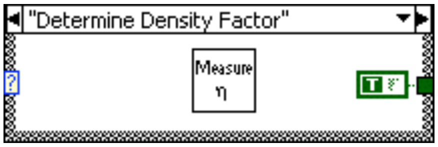


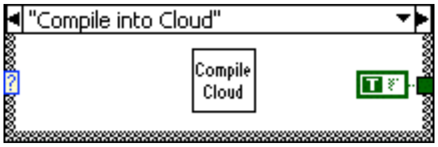


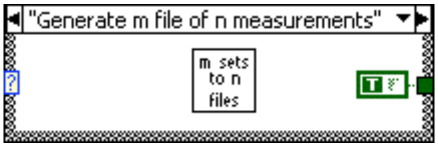












List of SubVIs and Express VIs with Configuration Information

Run_Me_for_Vector_Preprocessing_Use.vi

Appendix 4.3

Run_Me_to Compile_z_locations_into_cloud.vi

Appendix 4.4

Run_Me_for_Scattering_efficeincy.vi

Appendix 4.5

Run_Me_for_PIV_factor.vi

Appendix 4.6

Run_Me_to_measure_particle_density.vi

Appendix 4.7

Run_Me_m_to_n_measurements.vi

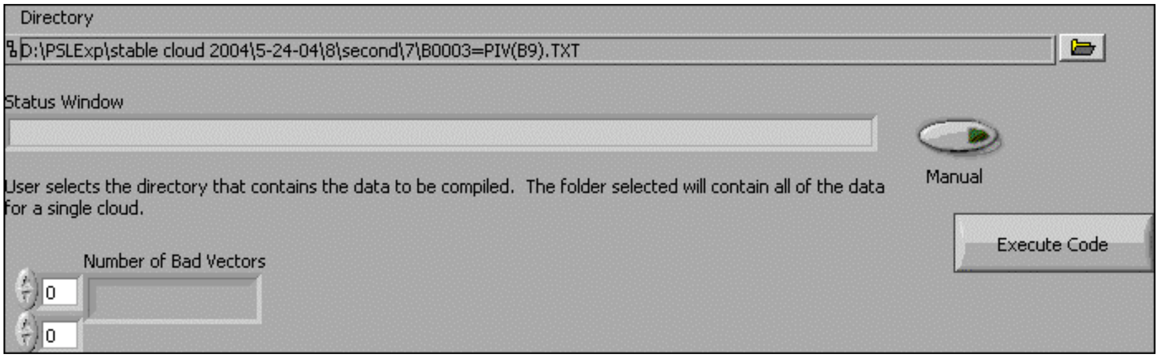
Appendix 4.8

A4.3: RUN_ME_FOR_VECTOR_PREPROCESSING_USE.VI

This code extracts the non-zero vectors that are generated in the PIV analysis.

Connector Pane

Front Panel

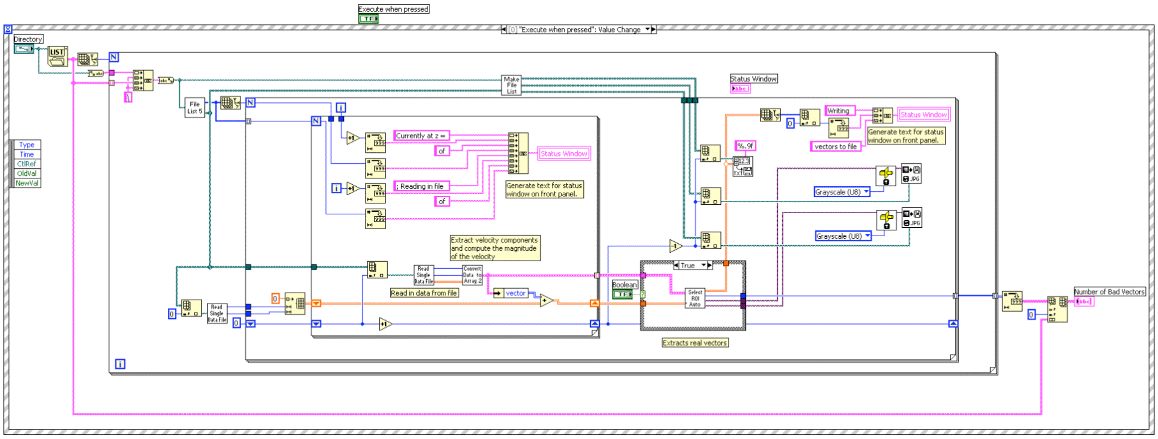


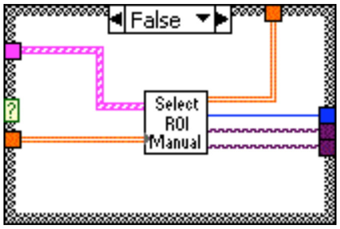
Directory: D:\PSLExp\stable cloud 2004\5-24-04\8\second\7\B0003=PIV(B9).TXT

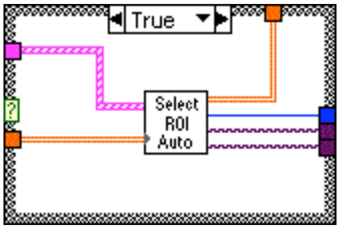
User selects the directory that contains the data to be compiled. The folder selected will contain all of the data for a single cloud.

Number of Bad Vectors: 0, 0

Block Diagram







List of SubVIs and Express VIs with Configuration Information

Convert_Data_to_Matrix_for_Vector_Preprocessing.vi

Appendix 4.3.1

Select_ROI_3_for_Vector_Preprocessing_AUTO.vi

Appendix 4.3.2

Get_File_List_for_Vector_Preprocessing.vi

Appendix 4.3.3

Generate_File_Names_for_Vector_Preprocessing.vi

Appendix 4.3.4

Select_ROI_3_for_Vector_Preprocessing.vi

Appendix 4.3.5

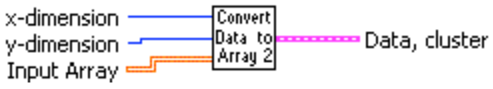
Read_ Single_Data_File_m_to_n_measurements.vi

Appendix 4.3.6

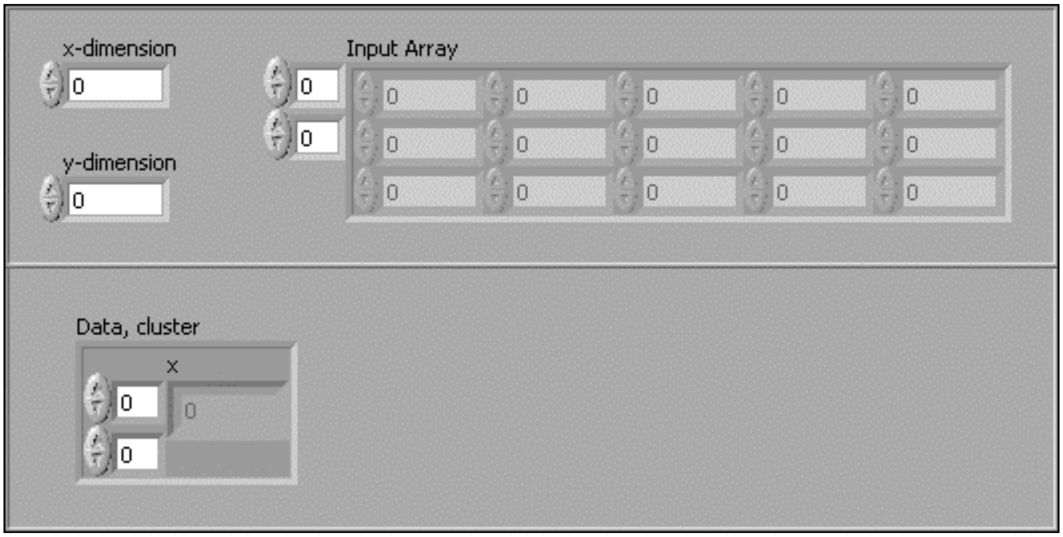
A4.3.1: Convert_Data_to_Matrix_for_Vector_Preprocessing.vi

Converts the vector file (exported from DaVis in .txt format) into a grid form and returns and cluster array that holds all of the data.

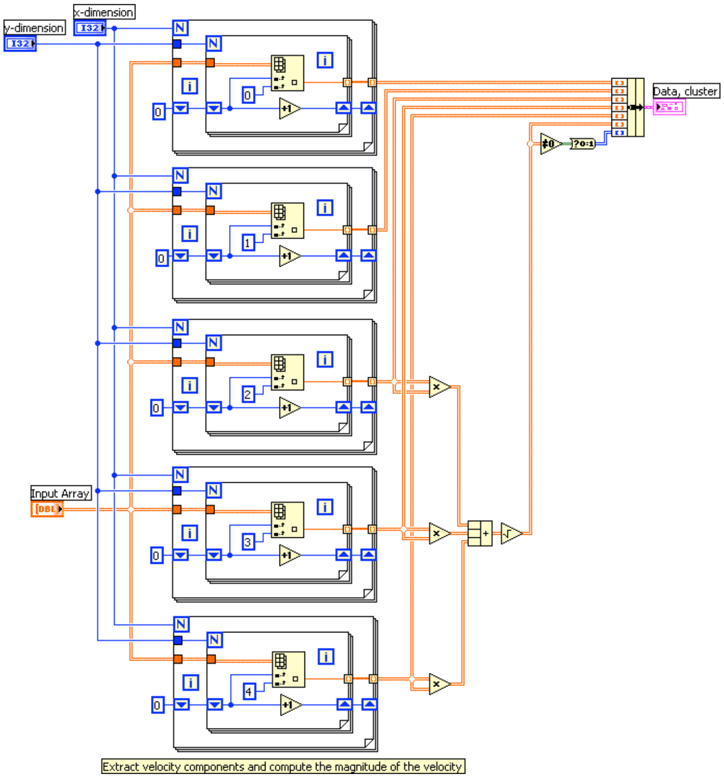
Connector Pane



Front Panel



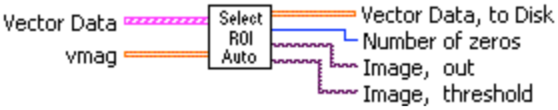
Block Diagram


y-dimension
x-dimension
I32
Input Array
DBL
N
i
0
1
2
3
4
Data, cluster
Extract velocity components and compute the magnitude of the velocity

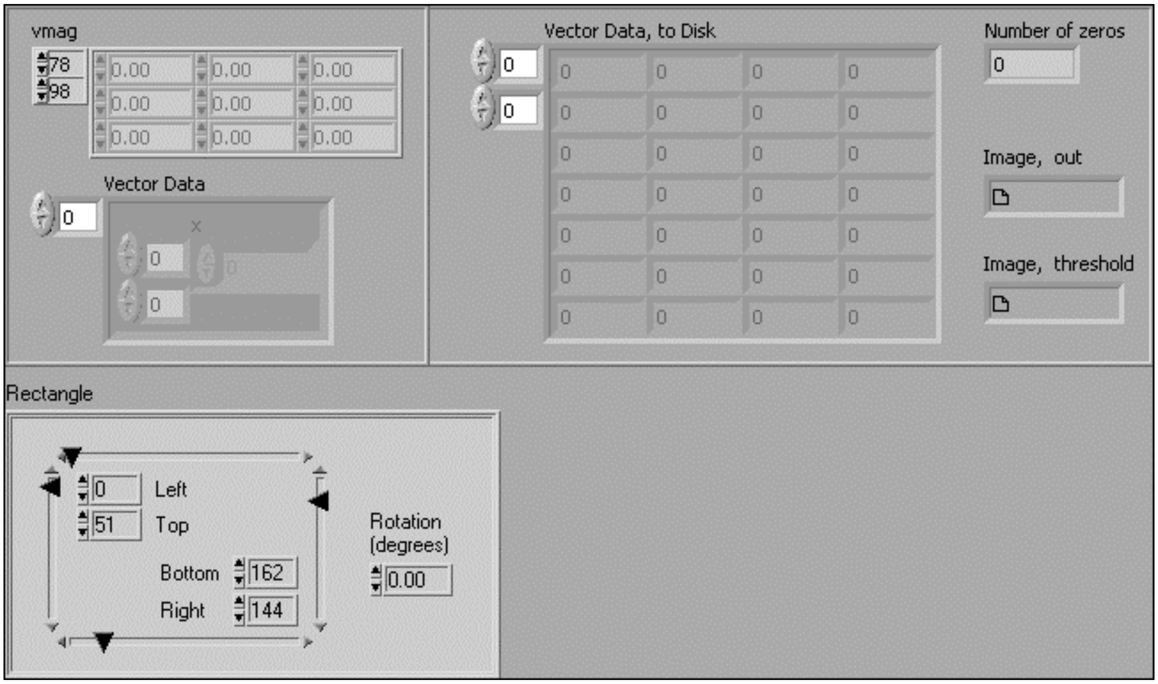
A4.3.2: Select_ROI_3_for_Vector_Preprocessing_AUTO.vi

The code automatically identifies the location of the cloud. Images are generated to check for accuracy of results.

Connector Pane



Front Panel


vmag: 78, 98

Vector Data: 0

Vector Data, to Disk: 0, 0

Number of zeros: 0

Image, out

Image, threshold

Rectangle

Left: 0

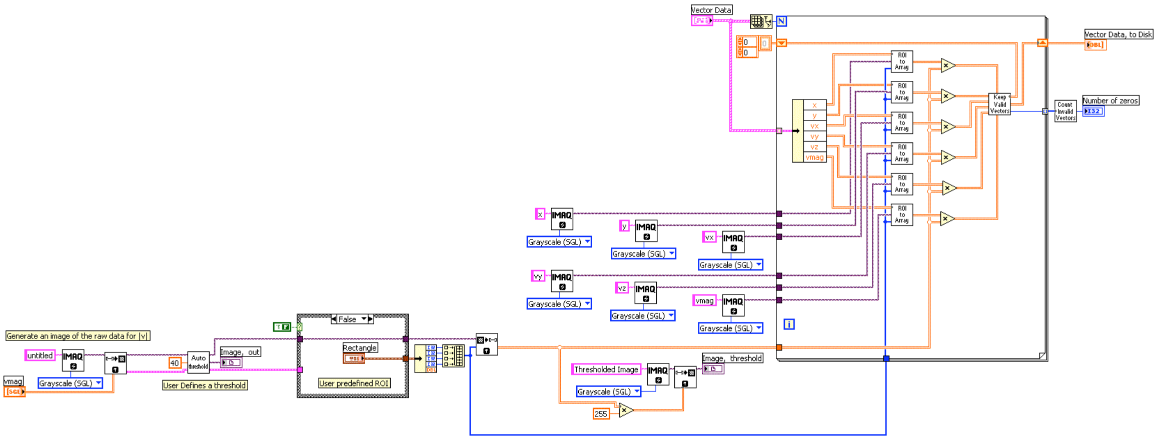
Top: 51

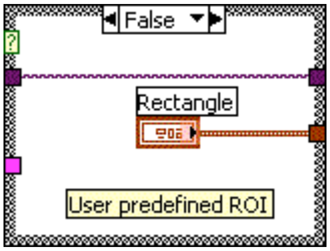
Bottom: 162

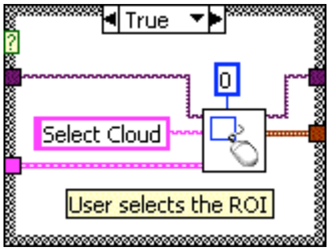
Right: 144

Rotation (degrees): 0.00

Block Diagram







List of SubVIs and Express VIs with Configuration Information

Auto threshold

Predefined_Thresholding.vi

Appendix 4.3.2.1

ROI to Array

Extract_Data_ROI.vi

Appendix 4.3.2.2

Count Invalid Vectors

Count_Zeros.vi

Appendix 4.3.2.3

Keep Valid Vectors

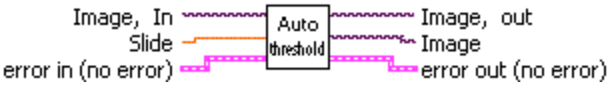
Valid_vectors_method_1.vi

Appendix 4.3.2.4

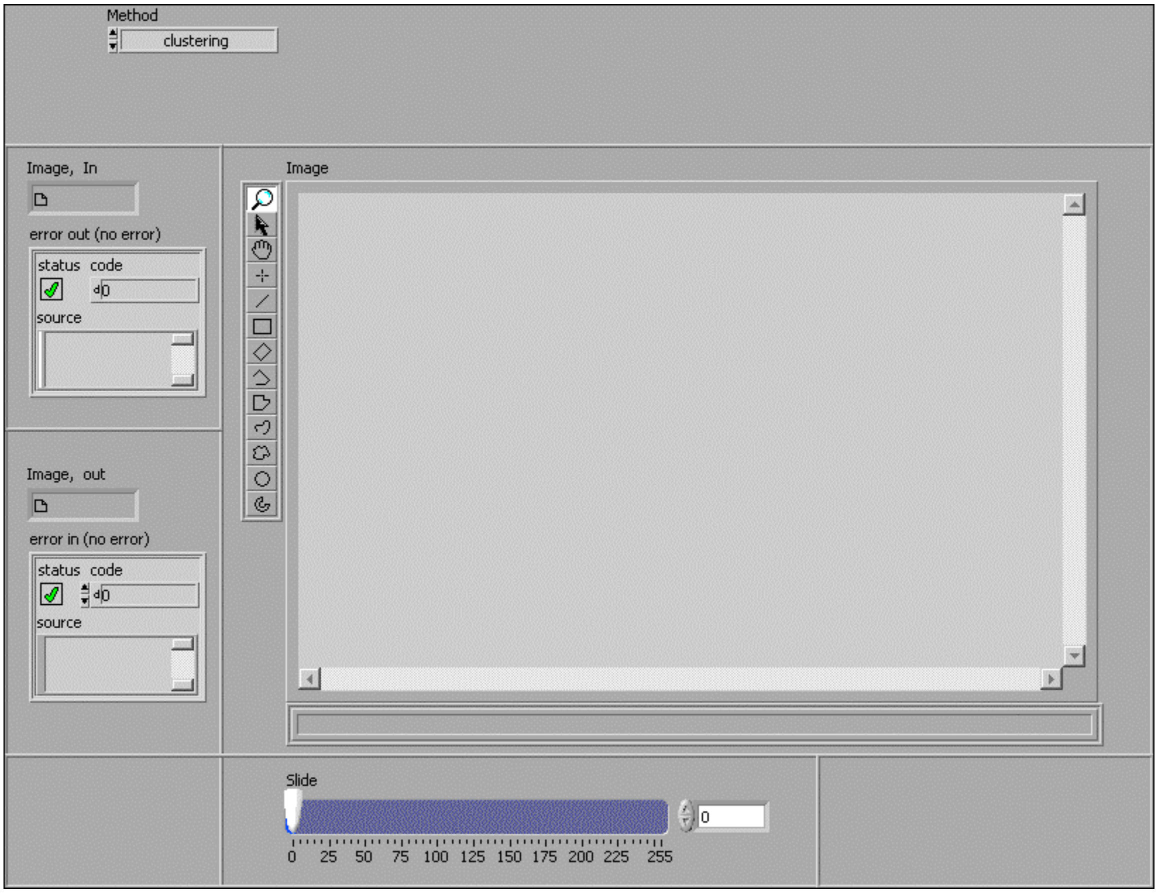
A4.3.2.1: PREDEFINED_THRESHOLDING.VI

The user generates a Mask that defines the location of the dust cloud.

Connector Pane



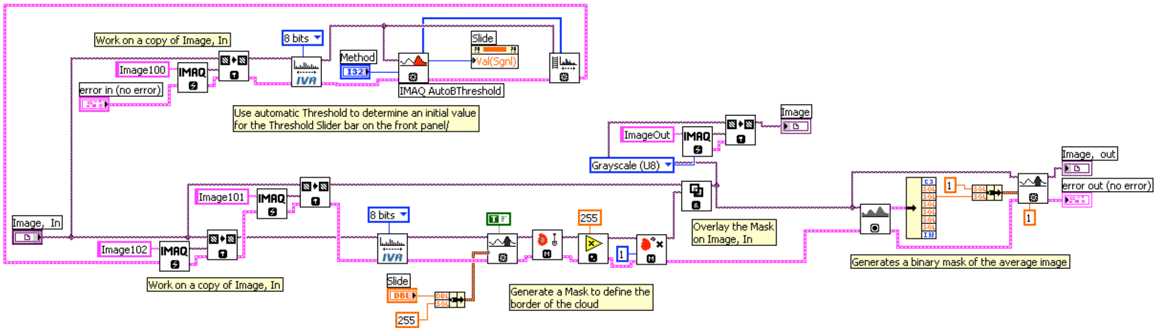
Front Panel


Method: clustering

status code 0

Slide 0

Block Diagram

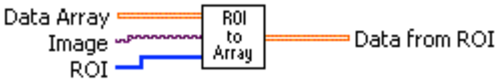


Work on a copy of Image, In
8 bits
Method
Slide
Val(Sgnl)
Image100
IMAQ
error in (no error)
IMAQ AutoBThreshold
Use automatic Threshold to determine an initial value for the Threshold Slider bar on the front panel/
Image
ImageOut
IMAQ
Grayscale (U8)
Image, out
error out (no error)
Image, In
Image101
IMAQ
8 bits
Image102
IMAQ
Work on a copy of Image, In
255
Overlay the Mask on Image, In
1
Slide
255
Generate a Mask to define the border of the cloud
Generates a binary mask of the average image

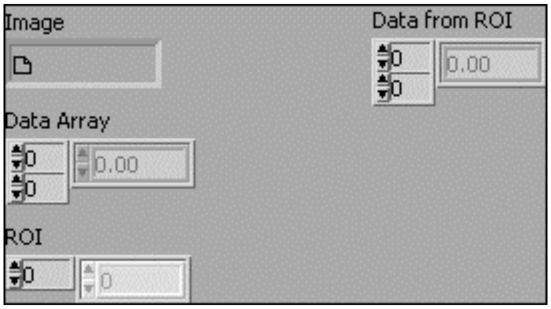
A4.3.2.2: Extract_Data_ROI.vi

Converts a user-defined ROI in an image to array form.

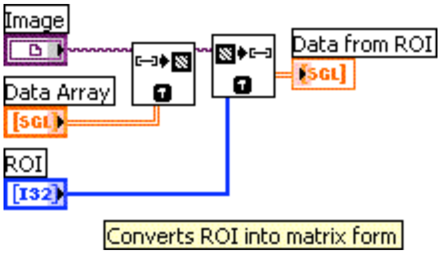
Connector Pane



Front Panel



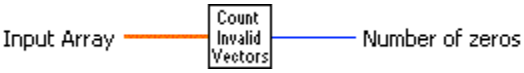
Block Diagram



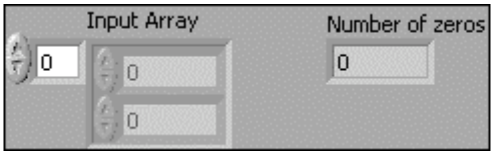
A4.3.2.3: COUNT_ZEROS.VI

Counts the number of zeros that occur in the "Input Array" and returns this value, "Number of zeros"

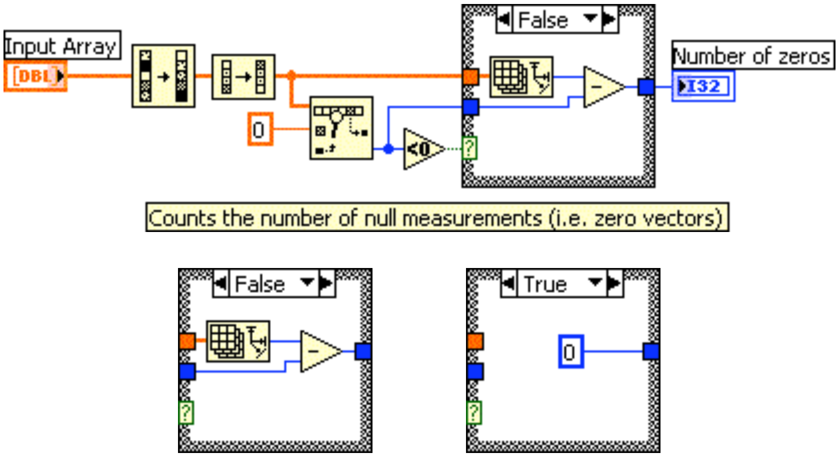
Connector Pane



Front Panel



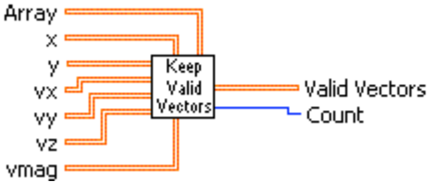
Block Diagram



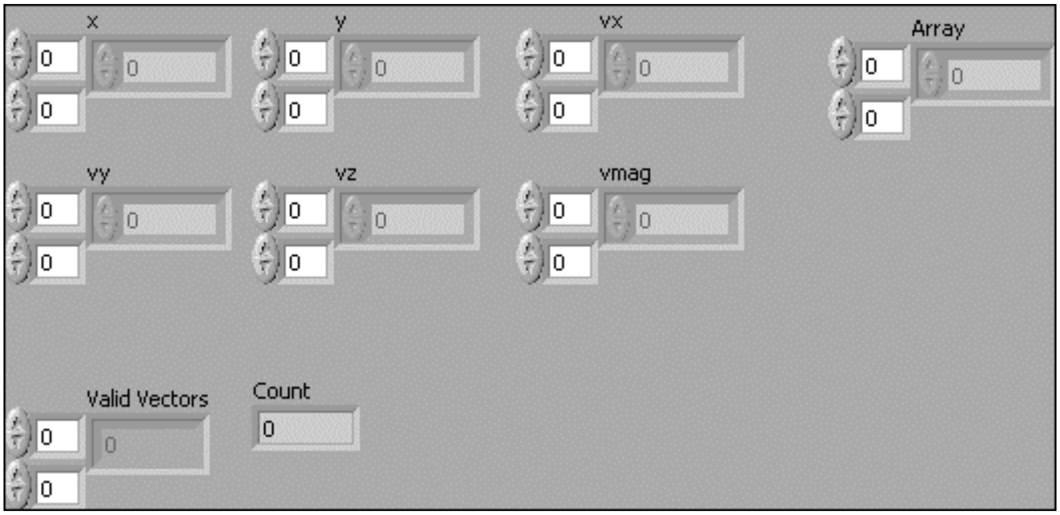
A4.3.2.4: VALID_VECTORS_METHOD_1.VI

Keep only the valid vectors

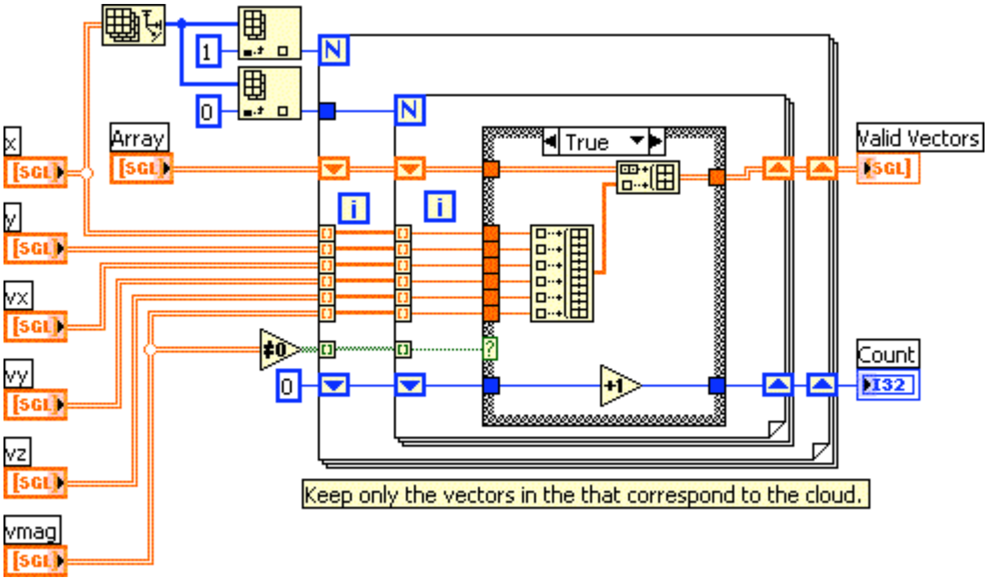
Connector Pane



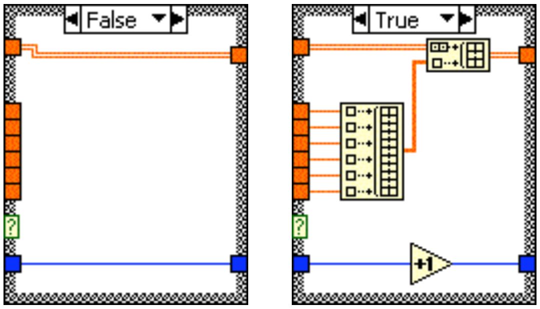
Front Panel



Block Diagram



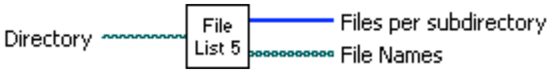
Keep only the vectors in the that correspond to the cloud.



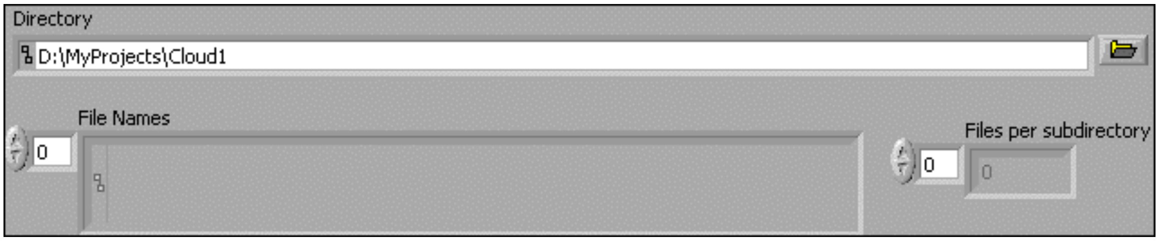
A4.3.3: Get_File_List_for_Vector_Preprocessing.vi

This code generates a list of files to be opened.

Connector Pane



Front Panel



Directory

D:\MyProjects\Cloud1

File Names

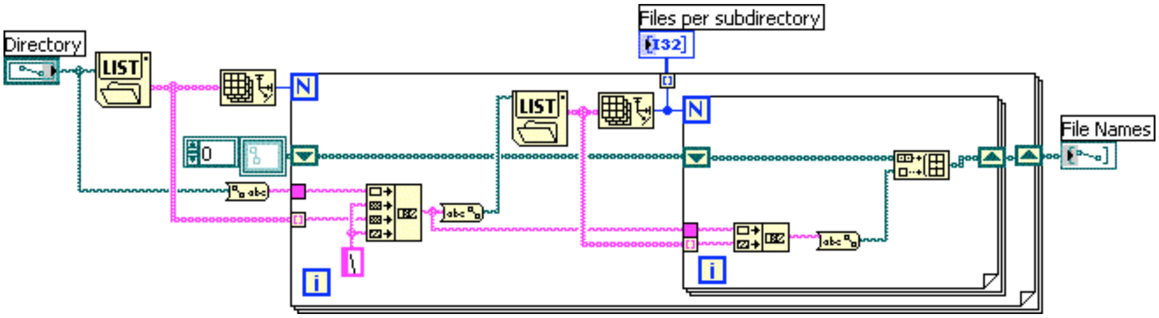
0

Files per subdirectory

0

0

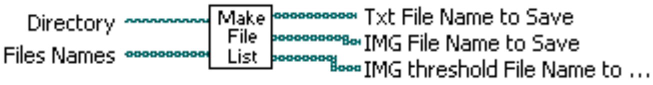
Block Diagram



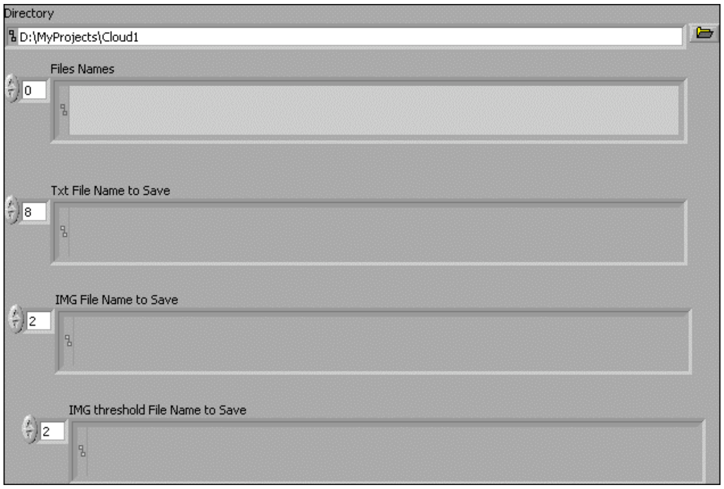
A4.3.4: GENERATE_FILE_NAMES_FOR_VECTOR_PREPROCESSING.VI

Generates list of filenames to save results to disk.

Connector Pane



Front Panel



Directory
D:\MyProjects\Cloud1

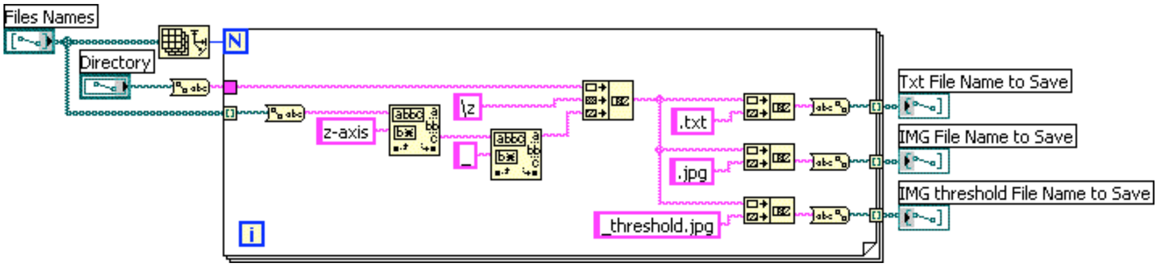
Files Names
0

Txt File Name to Save
8

IMG File Name to Save
2

IMG threshold File Name to Save
2

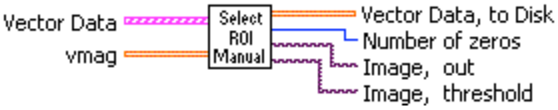
Block Diagram



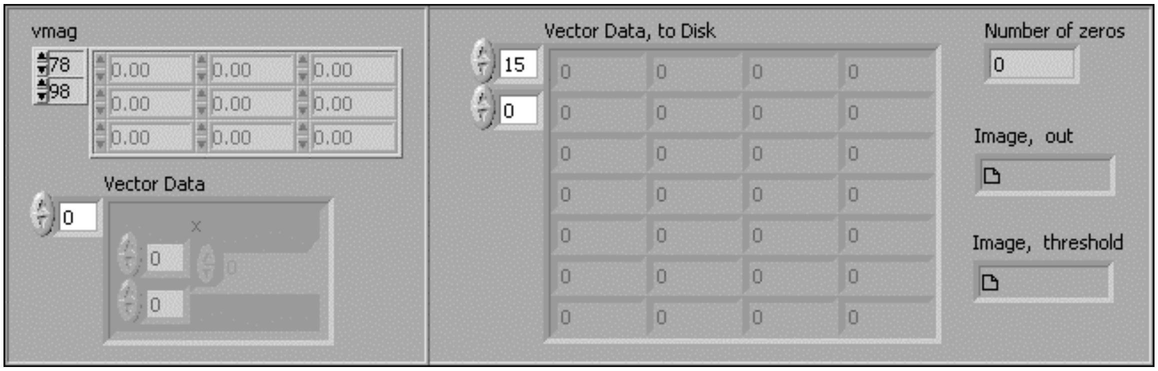
A4.3.5: Select_ROI_3_for_Vector_Preprocessing.vi

The valid vectors from a defined ROI are kept, while the others are discarded.

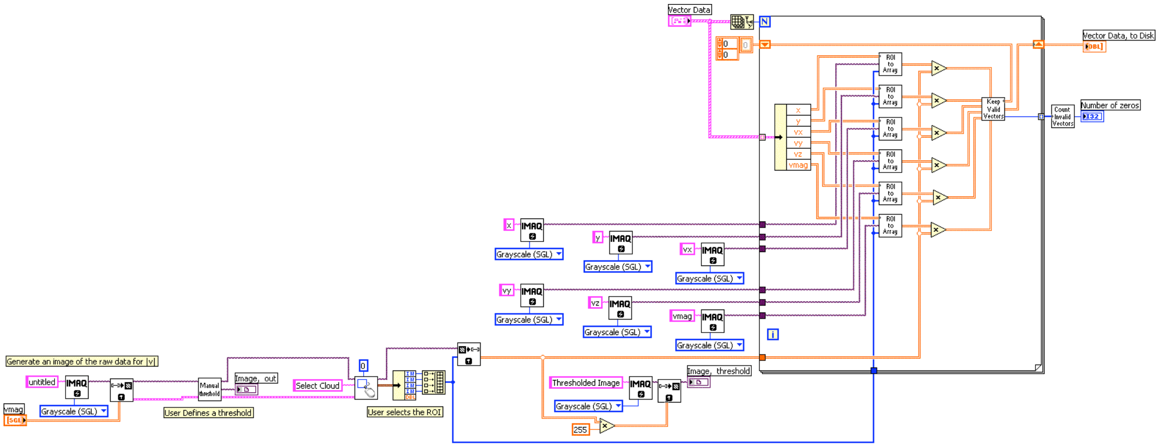
Connector Pane



Front Panel



Block Diagram



List of SubVIs and Express VIs with Configuration Information

Manual threshold

Manual Thresholding.vi

Appendix 4.3.5.1

ROI to Array

Extract_Data_ROI.vi

Appendix 4.3.5.2

Count Invalid Vectors

Count_Zeros.vi

Appendix 4.3.5.3

Keep Valid Vectors

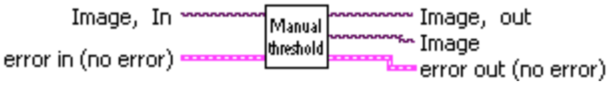
Valid_vectors_method_1.vi

Appendix 4.3.5.4

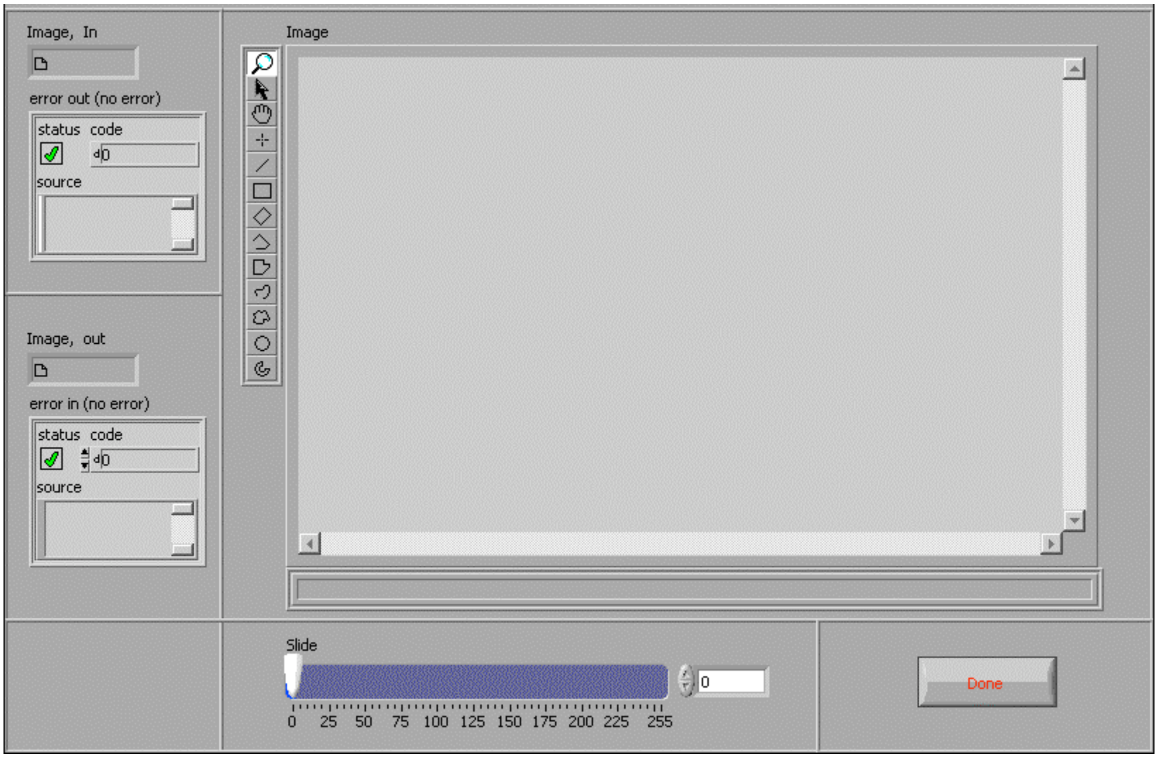
A4.3.5.1: MANUAL THRESHOLDING.VI

The user generates a Mask that defines the location of the dust cloud.

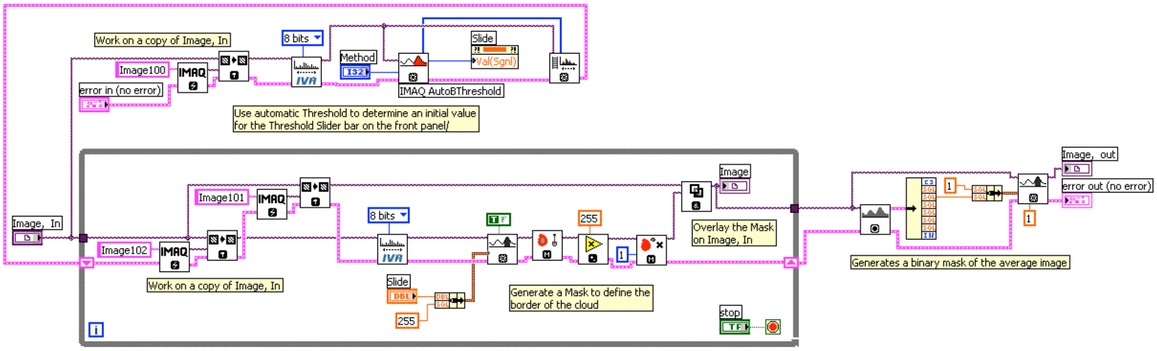
Connector Pane



Front Panel



Block Diagram


Work on a copy of Image, In

Use automatic Threshold to determine an initial value for the Threshold Slider bar on the front panel/

Work on a copy of Image, In

Generate a Mask to define the border of the cloud

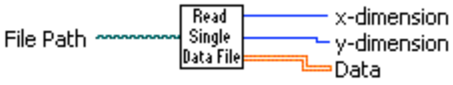
Overlay the Mask on Image, In

Generates a binary mask of the average image

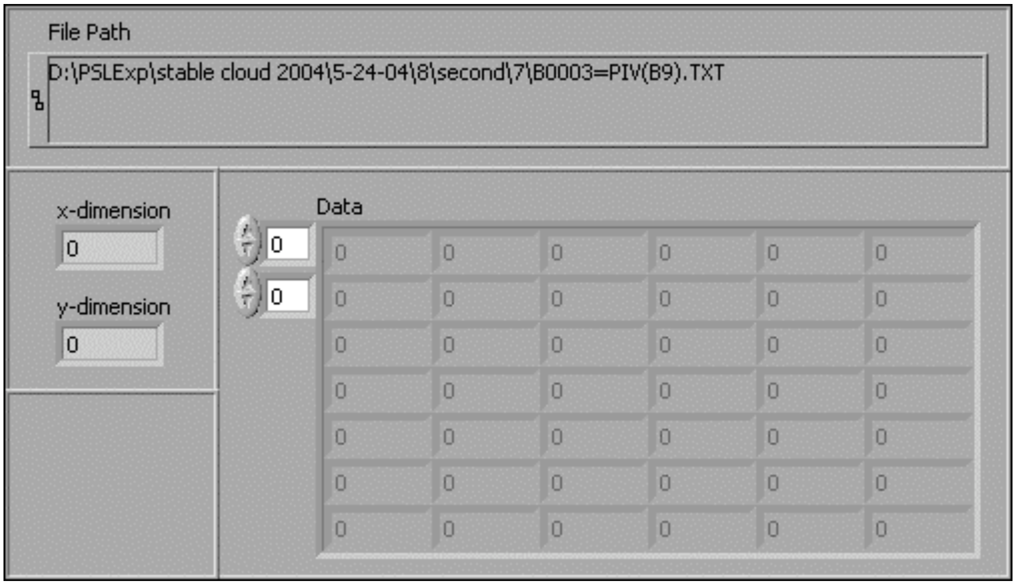
A4.3.6: Read_Single_Data_File_m_to_n_measurements.vi

Reads in a single vector file exported from DaVis in .txt format.

Connector Pane



Front Panel



File Path

D:\PSLExp\stable cloud 2004\5-24-04\8\second\7\B0003=PIV(B9).TXT

x-dimension

0

y-dimension

0

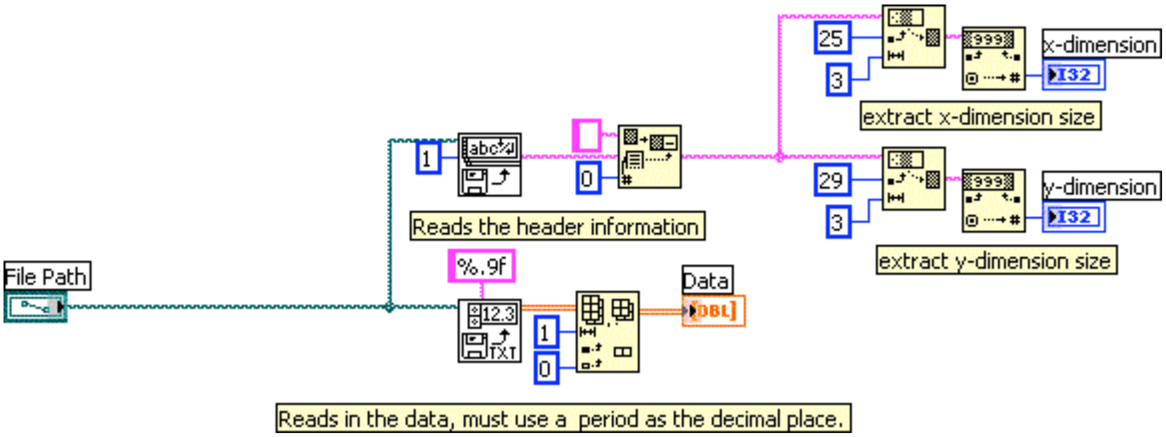
Data

0

0

0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

Block Diagram

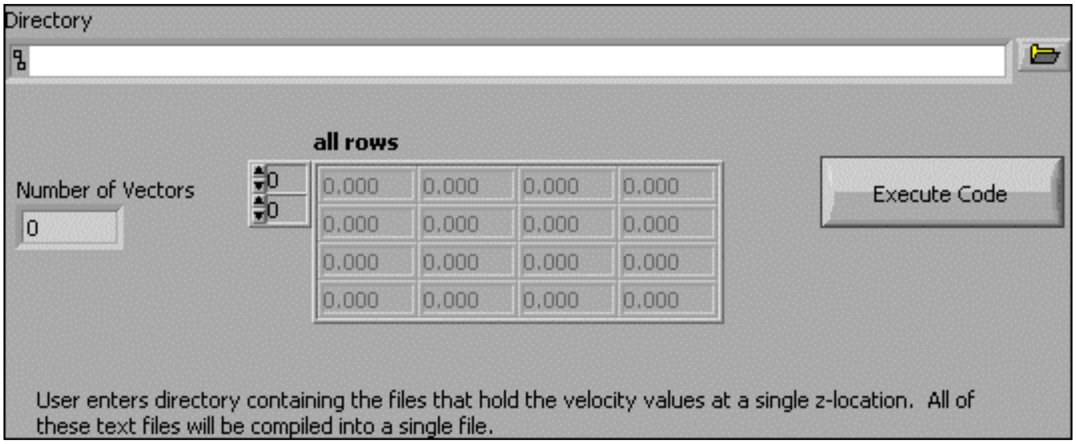


A4.4: Run_Me_to Compile_z_locations_into_cloud.vi

This code compiles the files at each z-location into a single file, i.e. all of the vectors across the cloud.

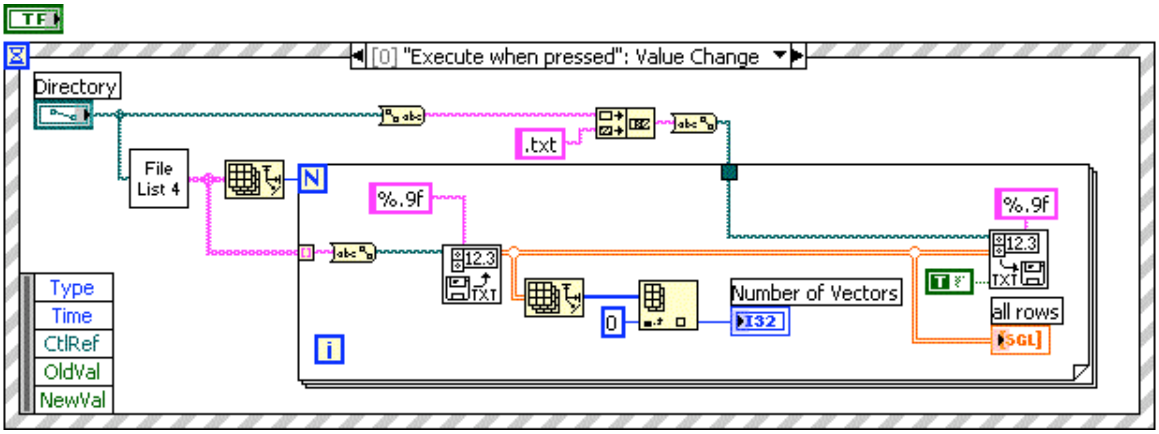
Connector Pane

Front Panel



User enters directory containing the files that hold the velocity values at a single z-location. All of these text files will be compiled into a single file.

Block Diagram



List of SubVIs and Express VIs with Configuration Information

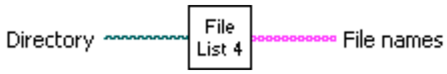
File List 4 Get_File_List_to Compile_z_locations_into_cloud.vi

Appendix 4.4.1

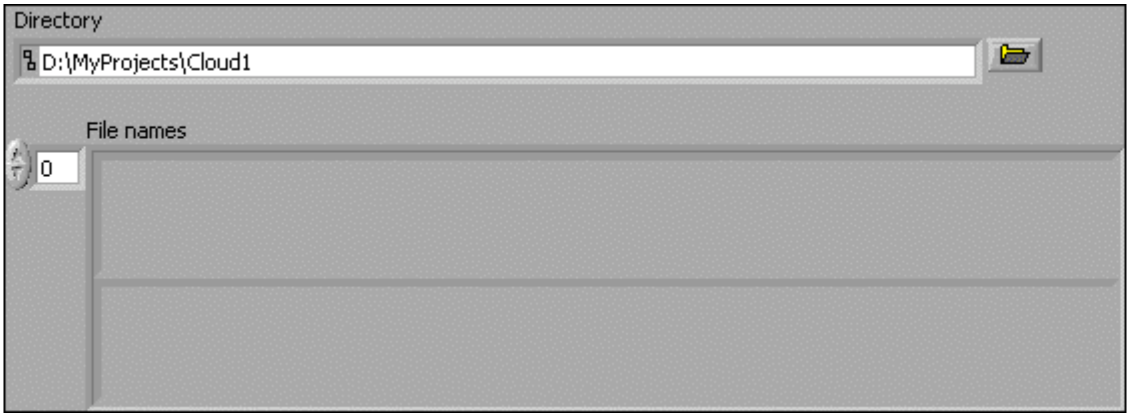
A4.4.1: Get_File_List_to Compile_z_locations_into_cloud.vi

Generates an array, "File Names", that list all of the files in a folder, "Directory." The file structure is coded for the Windows directory structure.

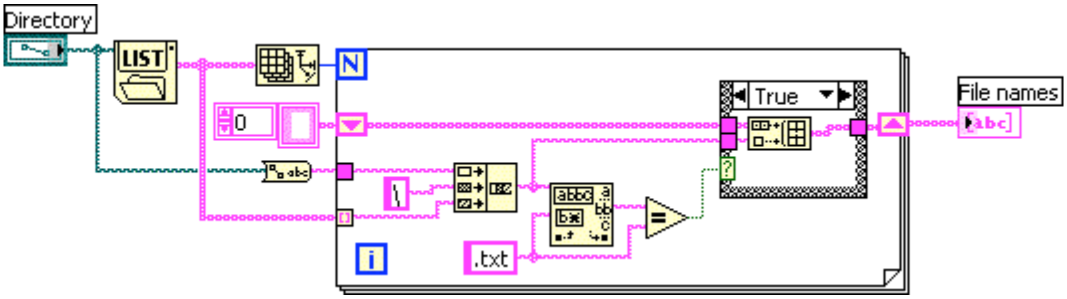
Connector Pane

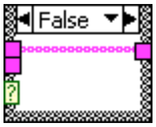


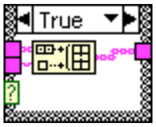
Front Panel


Directory
D:\MyProjects\Cloud1
File names
0

Block Diagram







A4.5: RUN_ME_FOR_SCATTERING_EFFICEINCY.VI

This code is used to determine the pixel intensity per particle for images from DaVis.

Connector Pane

Front Panel

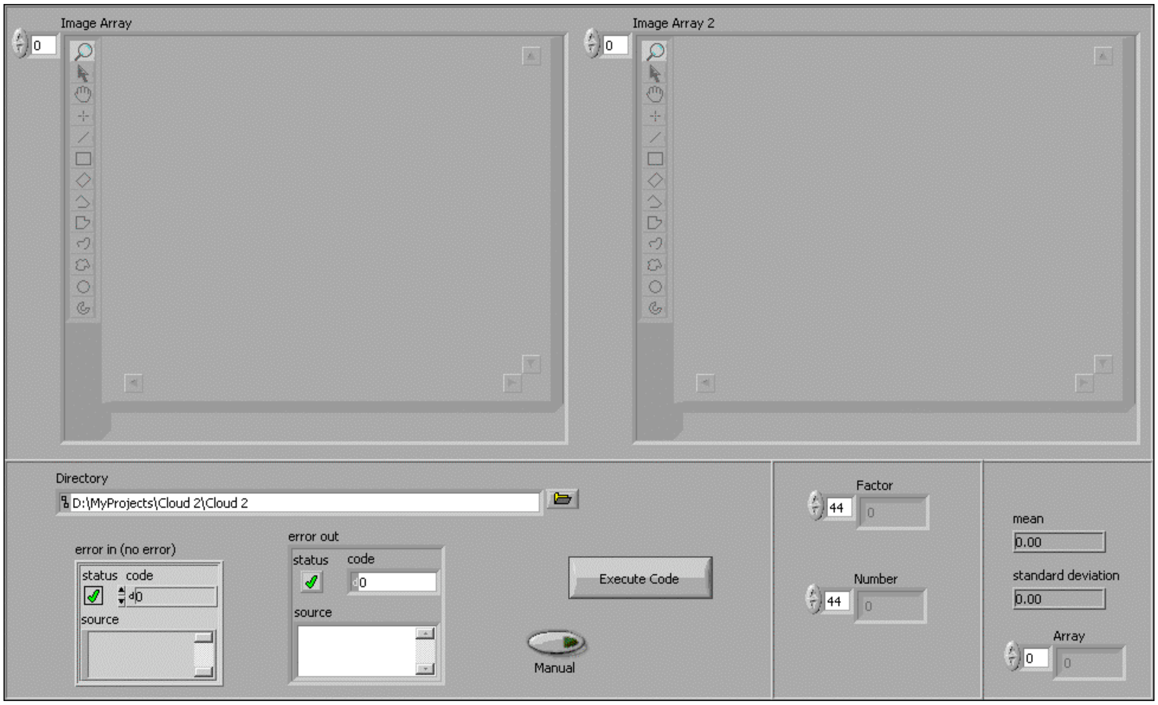
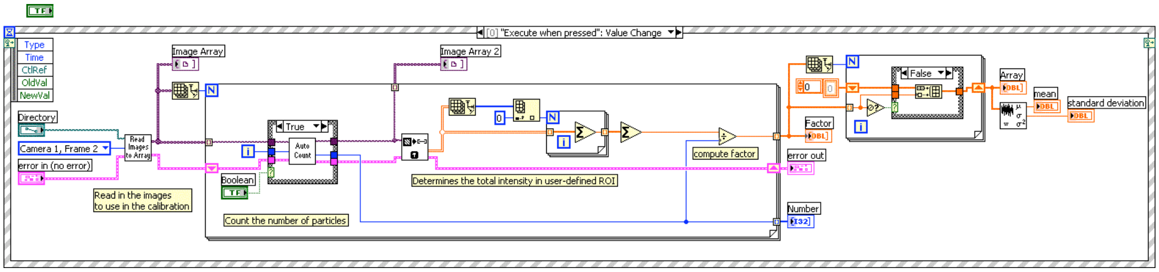
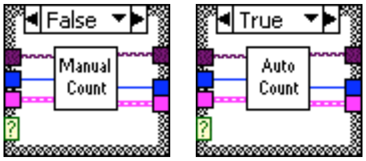
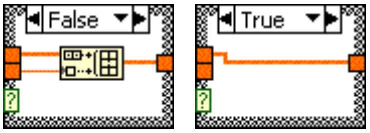


Image Array: 0
Image Array 2: 0
Directory: D:\MyProjects\Cloud 2\Cloud 2
error in (no error): status ✓ code 0 source
error out: status ✓ code 0 source
Factor: 44 0
Number: 44 0
mean: 0.00
standard deviation: 0.00
Array: 0 0

Block Diagram







List of SubVIs and Express VIs with Configuration Information

Manual Count

Manual_Particle_Counting_for_Scattering_efficeincy.vi

Appendix 4.5.1

Read Images to Array

Read _in_Images_(directory structure)_for_Scattering_efficeincy.vi

Appendix 4.5.2

Auto Count

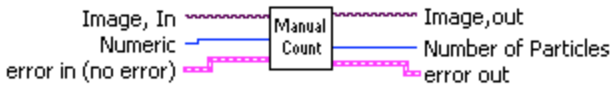
Automatic_Particle_Counting_for_Scattering_efficeincy.vi

Appendix 4.5.3

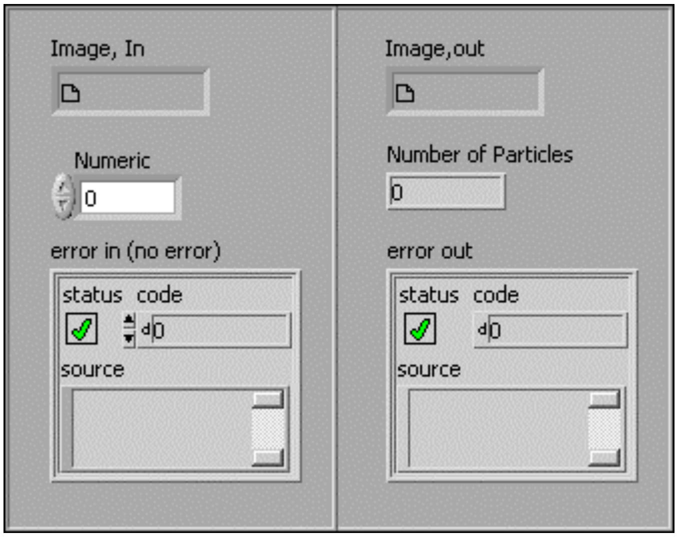
A4.5.1: MANUAL_PARTICLE_COUNTING_FOR_SCATTERING_EFFICEINCY.VI

User selects a region of known density and enters the number of particles in that region.

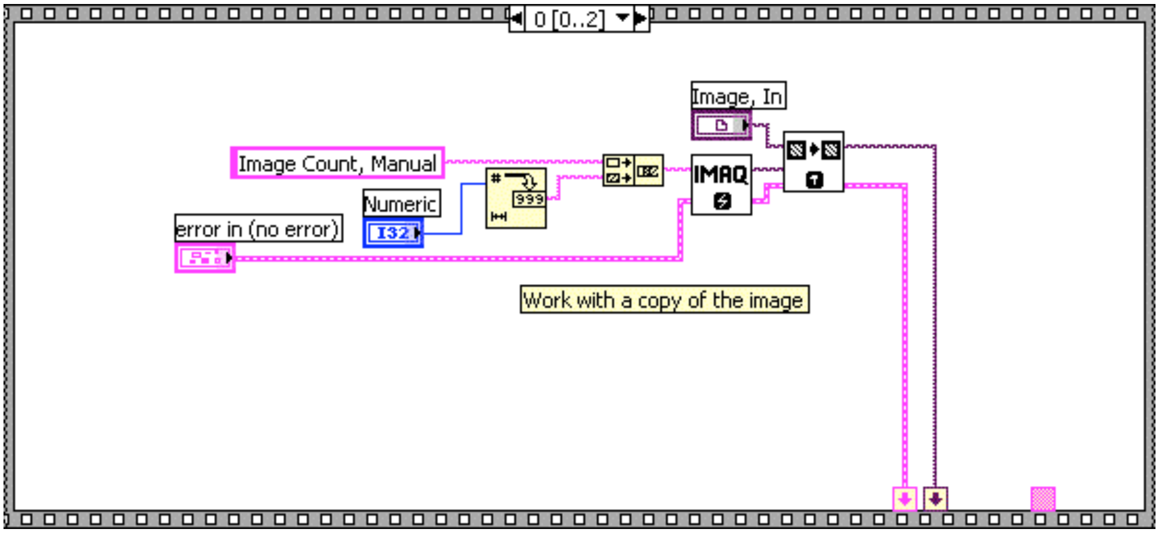
Connector Pane

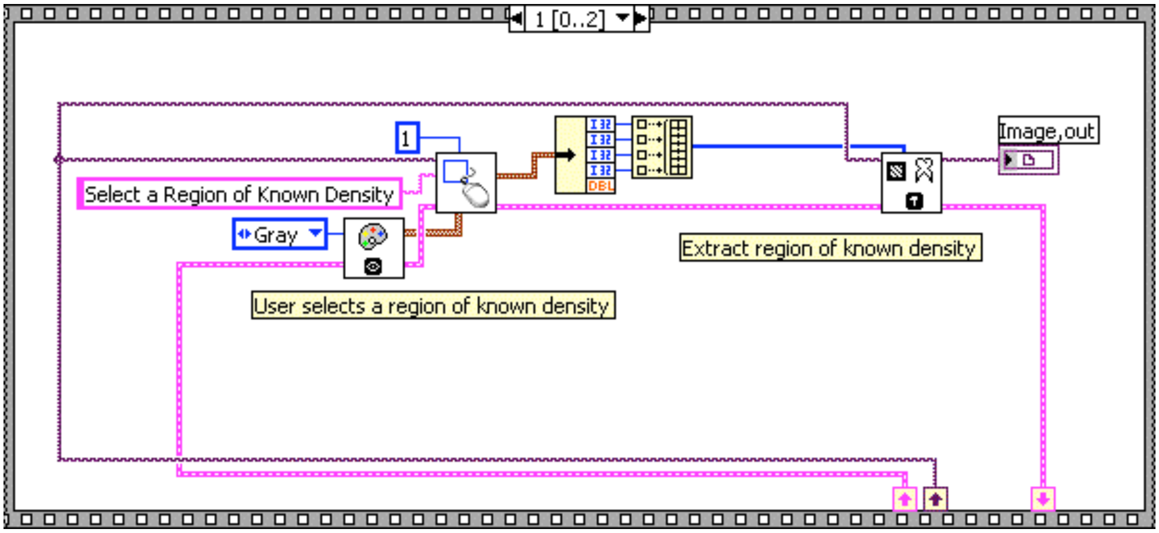


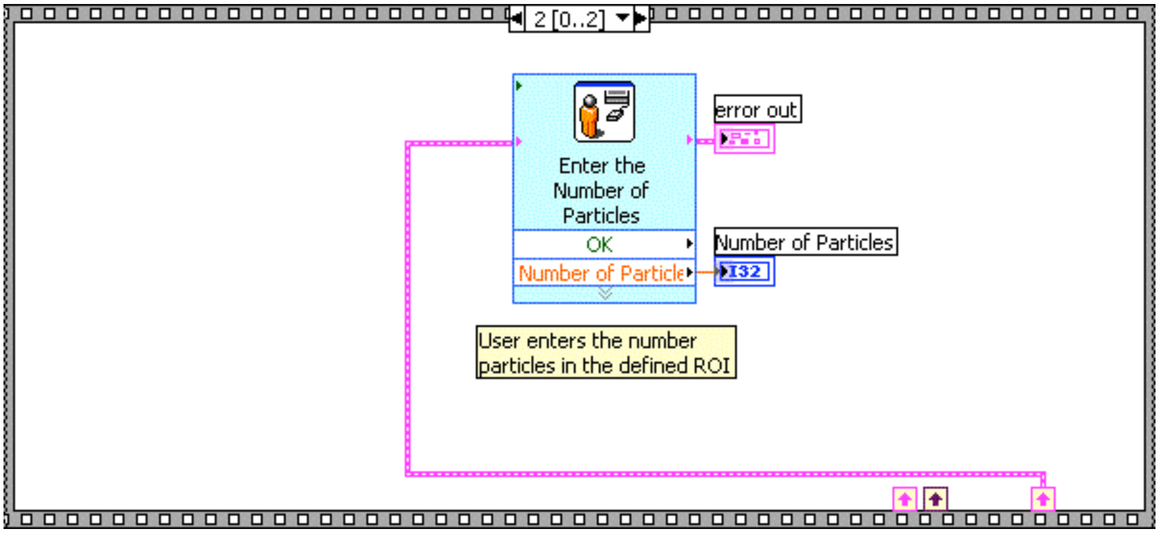
Front Panel



Block Diagram



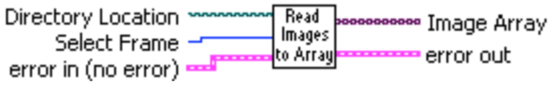

1 [0..2]
1
Select a Region of Known Density
Gray
User selects a region of known density
DBL
Extract region of known density
Image out


2 [0..2]
Enter the Number of Particles
OK
Number of Particle
error out
Number of Particles
I32
User enters the number particles in the defined ROI

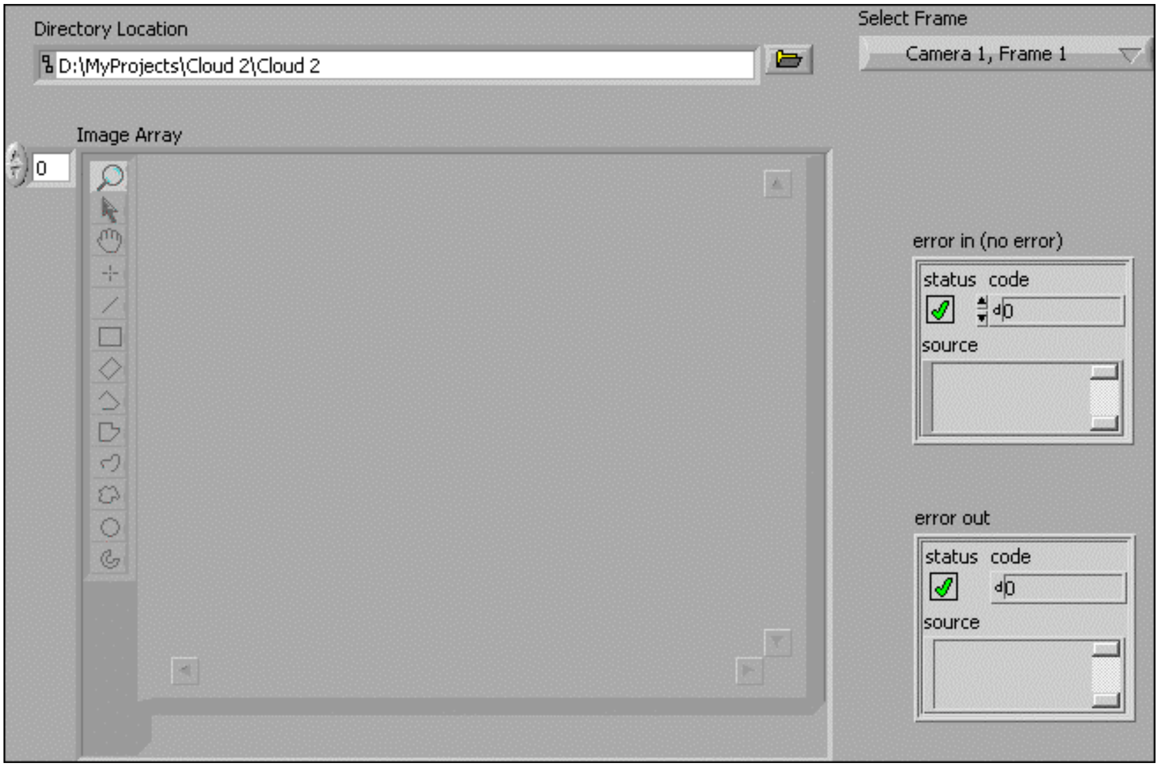
A4.5.2: READ_IN_IMAGES_(DIRECTORY STRUCTURE)_FOR_SCATTERING_EFFICEINCY.VI

Reads in IMX, IM7 (LaVision image format) images stored in "Directory Location". The user specifies which frame to read in using "Select Frame" and what region of the image is wanted using ROI. This portion of the image is stored in an array, "Image Array".

Connector Pane



Front Panel


Directory Location: D:\MyProjects\Cloud 2\Cloud 2

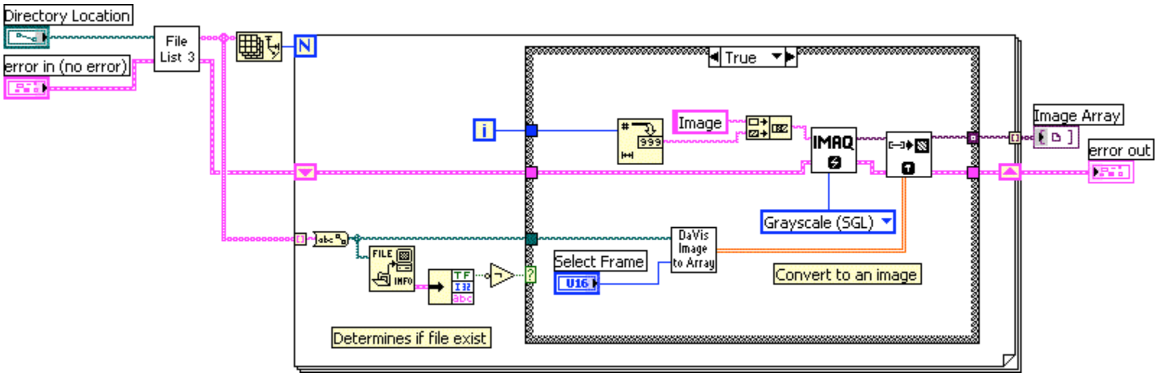
Select Frame: Camera 1, Frame 1

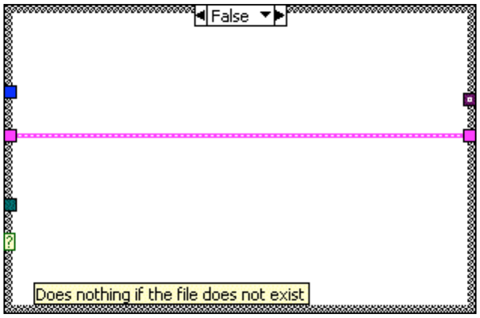
Image Array: 0

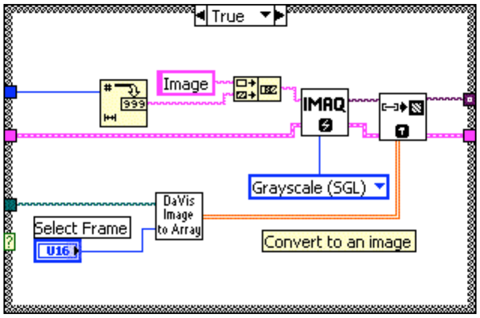
error in (no error): status ✓ code 0

error out: status ✓ code 0

Block Diagram







List of SubVIs and Express VIs with Configuration Information

File List 3 Get_directory_file_list_for_Scattering_efficeincy.vi

Appendix 4.5.2.1

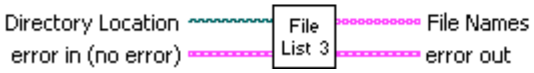
DaVis Image to Array Read_in_Image_data_for_particle_density.vi

Appendix 4.5.2.2

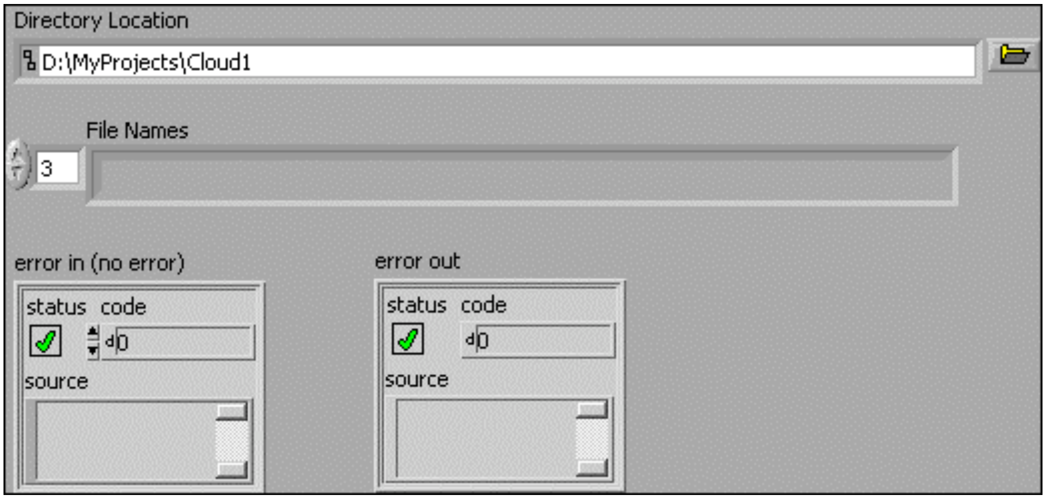
A4.5.2.1: GET_DIRECTORY_FILE_LIST_FOR_SCATTERING_EFFICEINCY.VI

Generates an array, "File Names", that list all of the files in a folder, "Directory Location". The file structure is coded for the Windows directory structure.

Connector Pane



Front Panel


Directory Location

D:\MyProjects\Cloud1

File Names

3

error in (no error)

status code

✓ 0

source

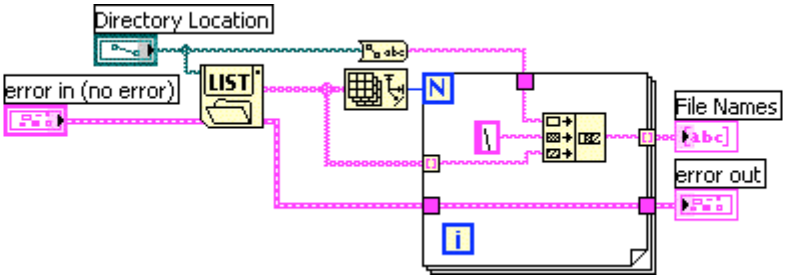
error out

status code

✓ 0

source

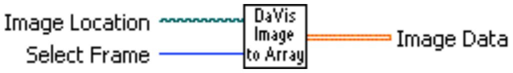
Block Diagram



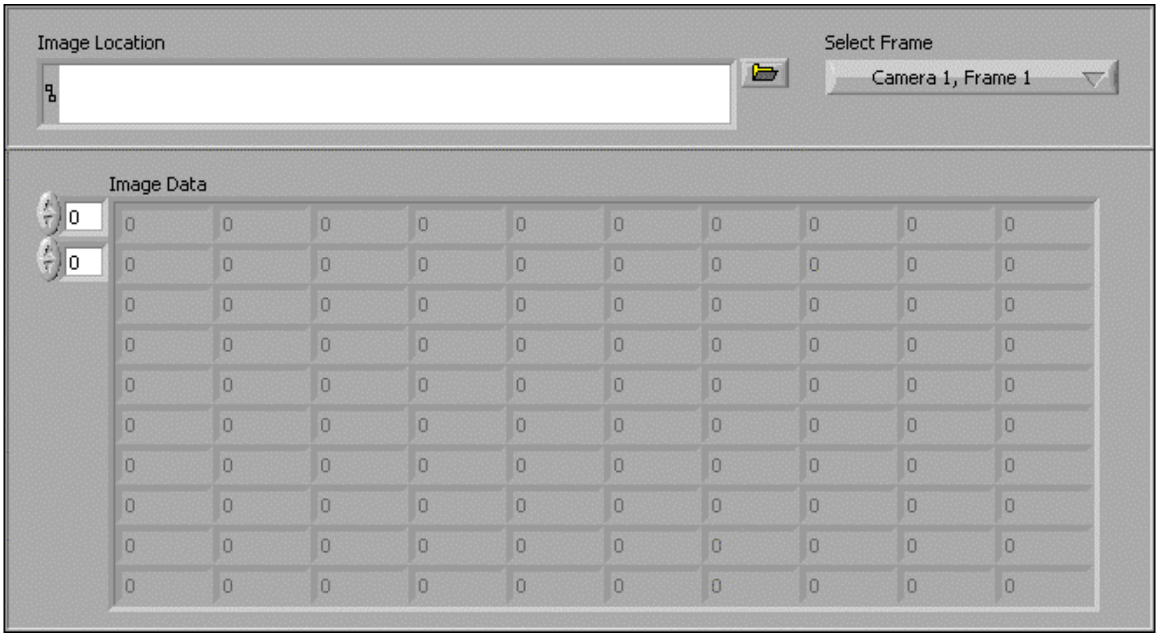
A4.5.2.2: READ_IN_IMAGE_DATA_FOR_PARTICLE_DENSITY.VI

Reads in IMX, IM7 (LaVision image format) images and returns the image data in an array.

Connector Pane



Front Panel

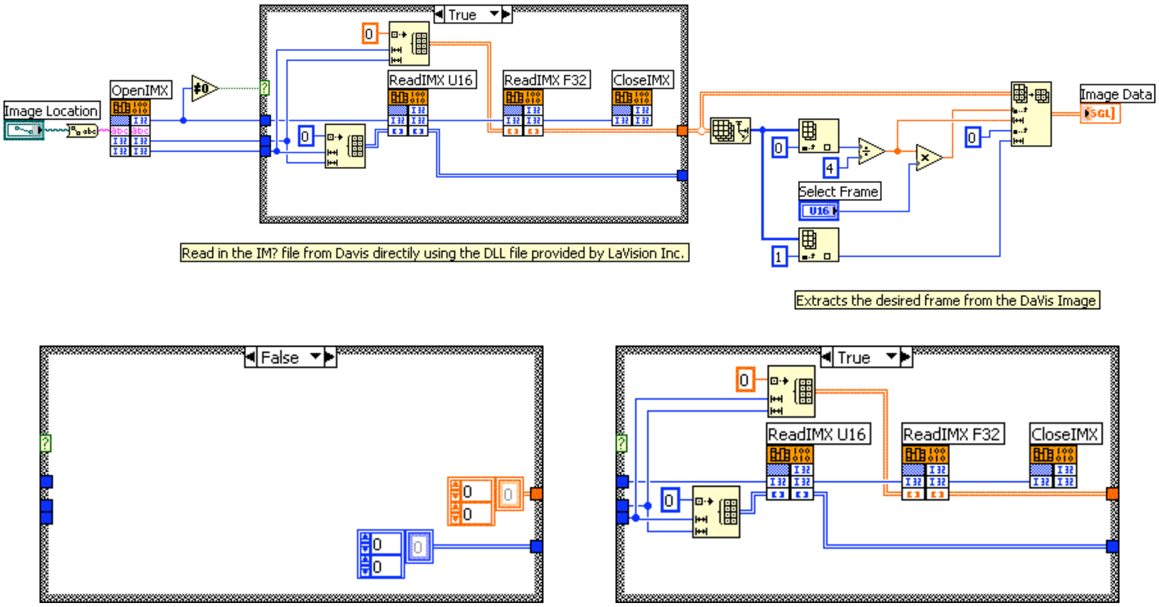


Select Frame: Camera 1, Frame 1

Image Data (indices 0, 0):

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

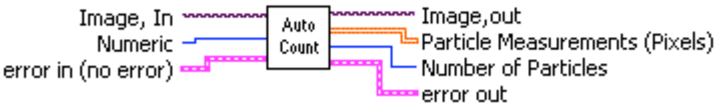
Block Diagram


True
Image Location
OpenIMX
ReadIMX U16
ReadIMX F32
CloseIMX
Image Data
SGL
Select Frame
U16
Read in the IM? file from Davis directily using the DLL file provided by LaVision Inc.
Extracts the desired frame from the DaVis Image
False
True
ReadIMX U16
ReadIMX F32
CloseIMX

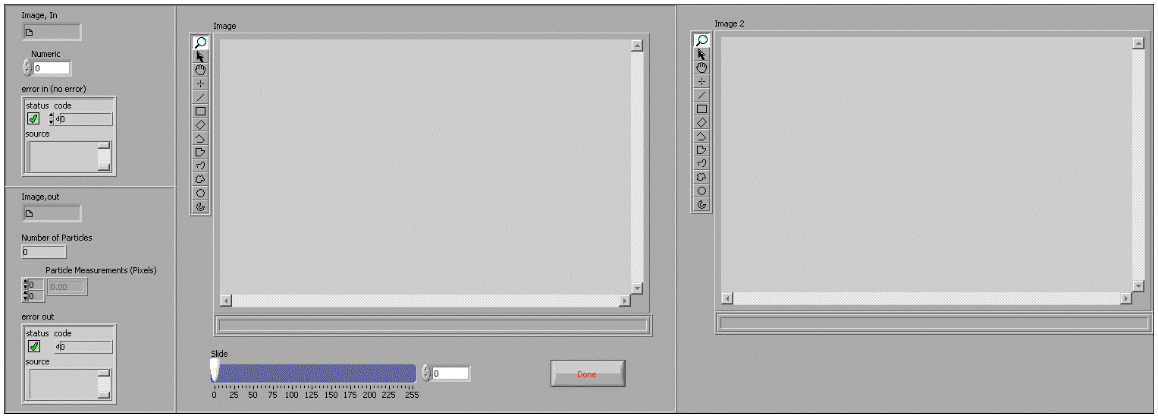
A4.5.3: Automatic_Particle_Counting_for_Scattering_efficeincy.vi

Automatically counts the number of particles in a region.

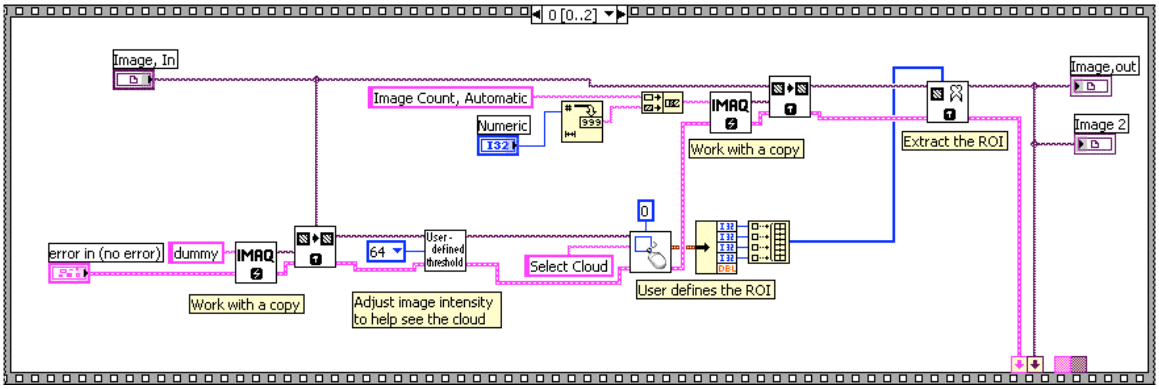
Connector Pane

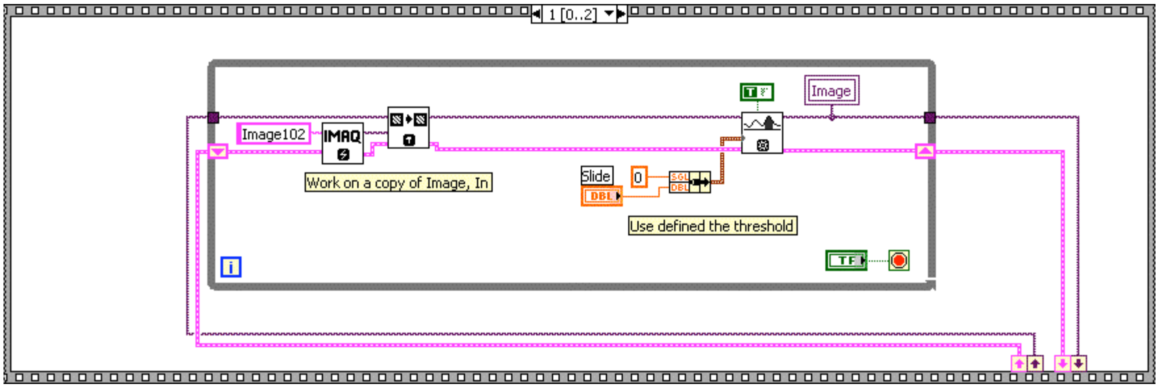


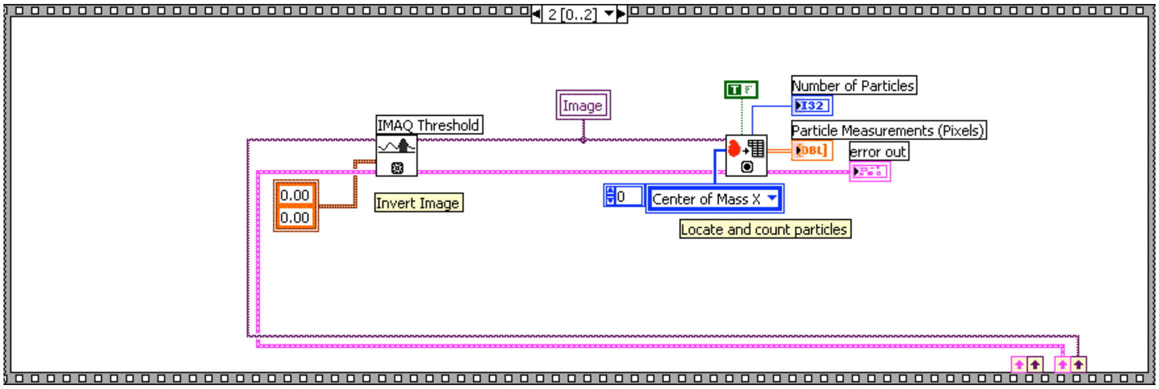
Front Panel



Block Diagram







List of SubVIs and Express VIs with Configuration Information

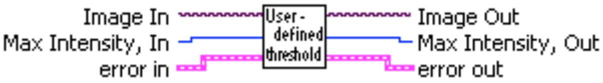
Adjust Image Display Range (fixed value).vi

Appendix 4.5.3.1

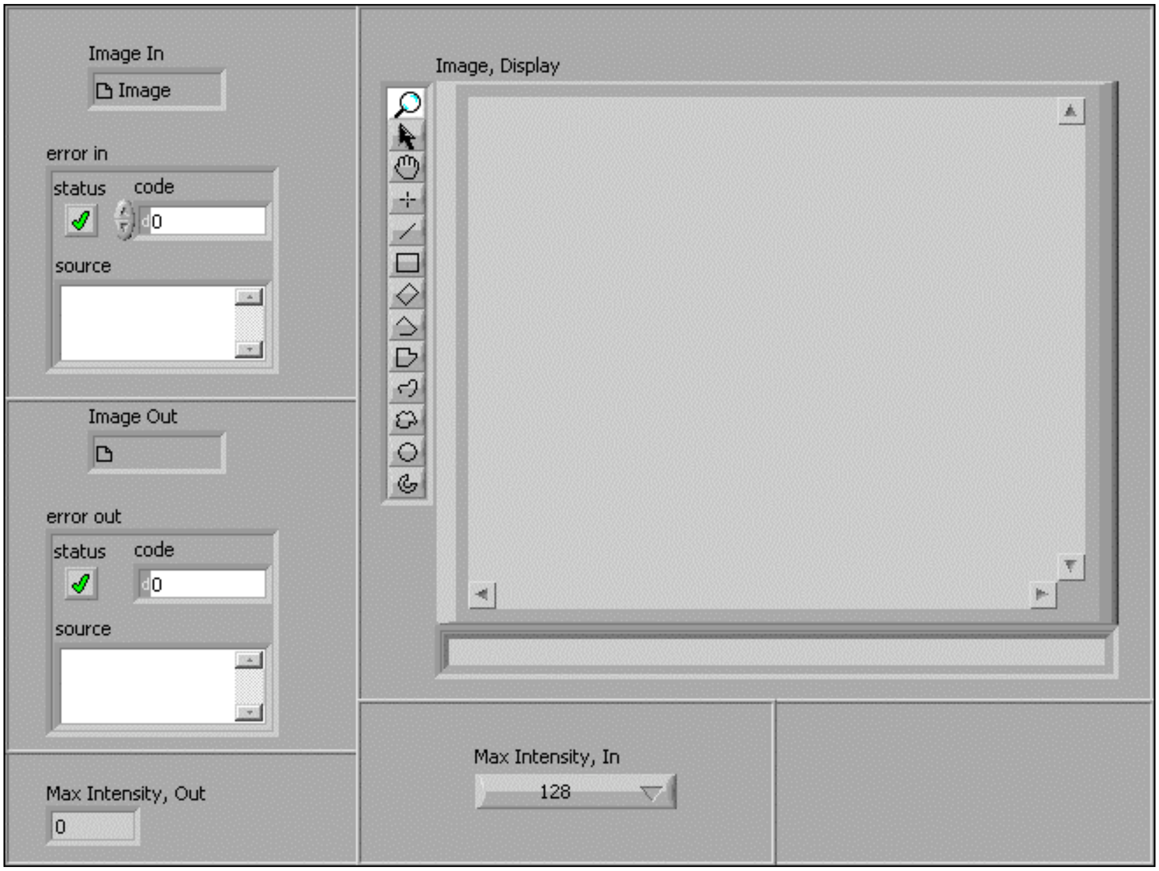
A4.5.3.1: ADJUST IMAGE DISPLAY RANGE (FIXED VALUE).VI

Image values are capped at a User Selected value. Intensities greater than this value are set to zero.

Connector Pane



Front Panel

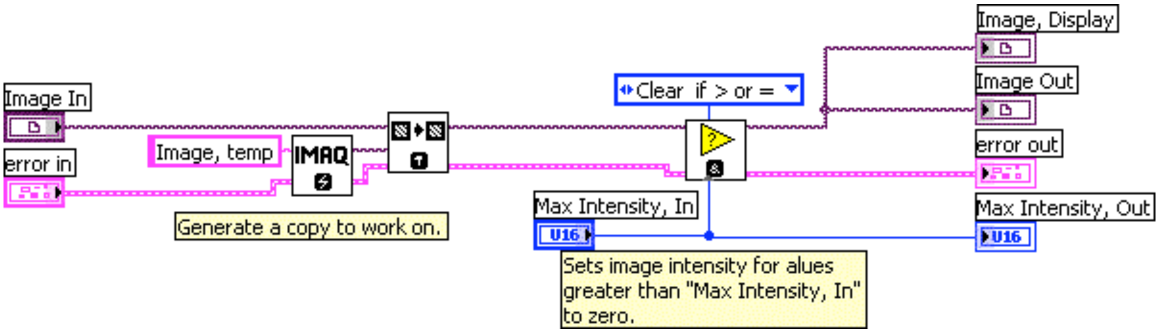


code 0

Max Intensity, Out 0

Max Intensity, In 128

Block Diagram



Generate a copy to work on.

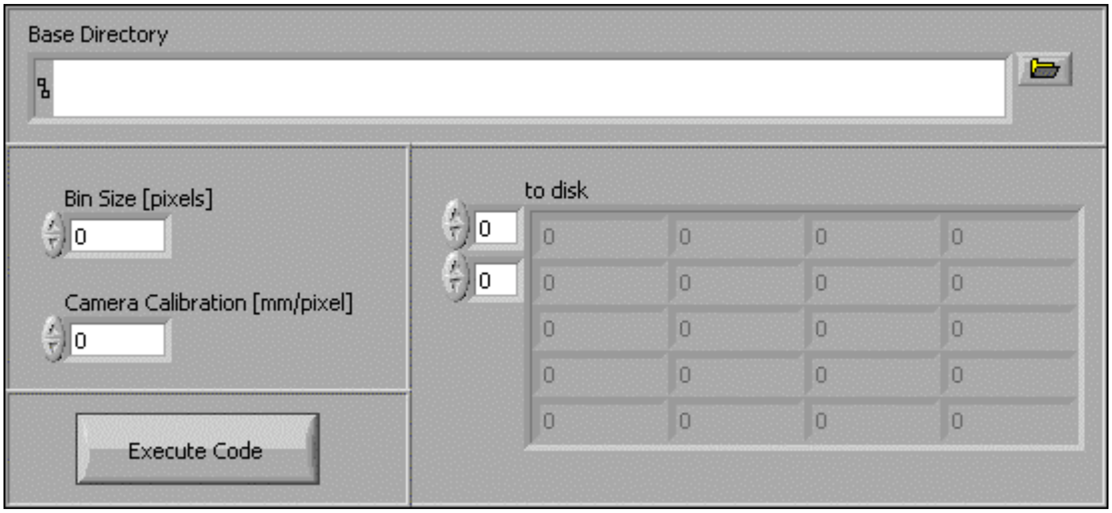
Sets image intensity for alues greater than "Max Intensity, In" to zero.

A4.6: RUN_ME_FOR_PIV_FACTOR.VI

This code will determine the scattering effeciency that is needed to extract a particle denisty from the images acquired using the stereo-PIV system

Connector Pane

Front Panel



Base Directory

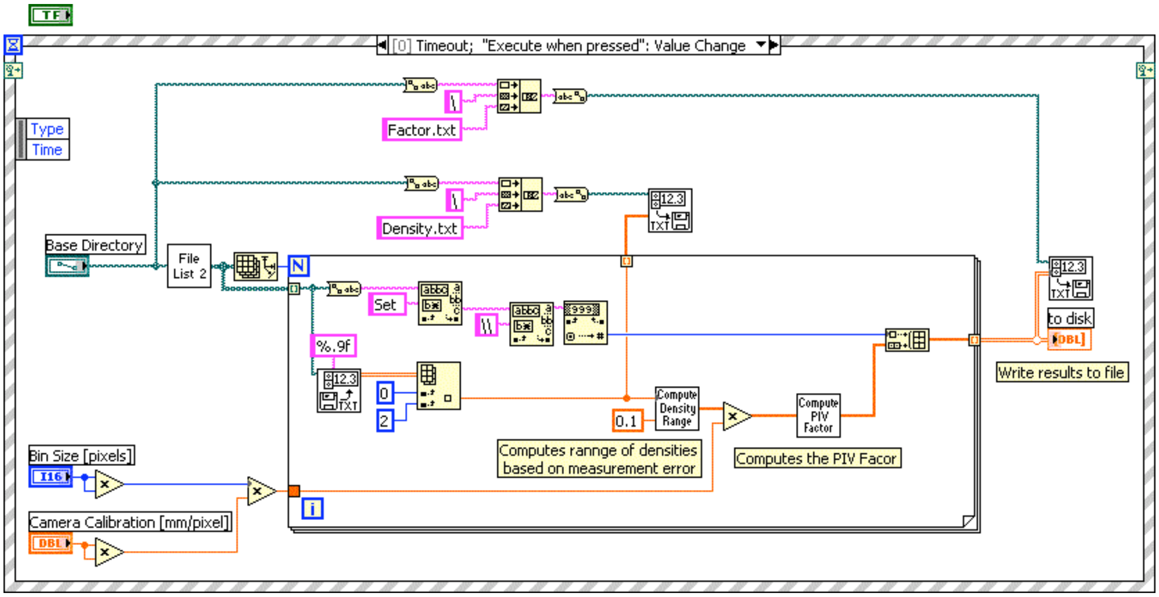
Bin Size [pixels] 0

Camera Calibration [mm/pixel] 0

to disk 0 0

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

Block Diagram



List of SubVIs and Express VIs with Configuration Information

File List 2

Get_File_List_PIV_factor.vi

Appendix 4.6.1

Compute PIV Factor

Compute_PIV_factor_to_file.vi

Appendix 4.6.2

Compute Density Range

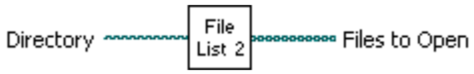
Density _Range.vi

Appendix 4.6.3

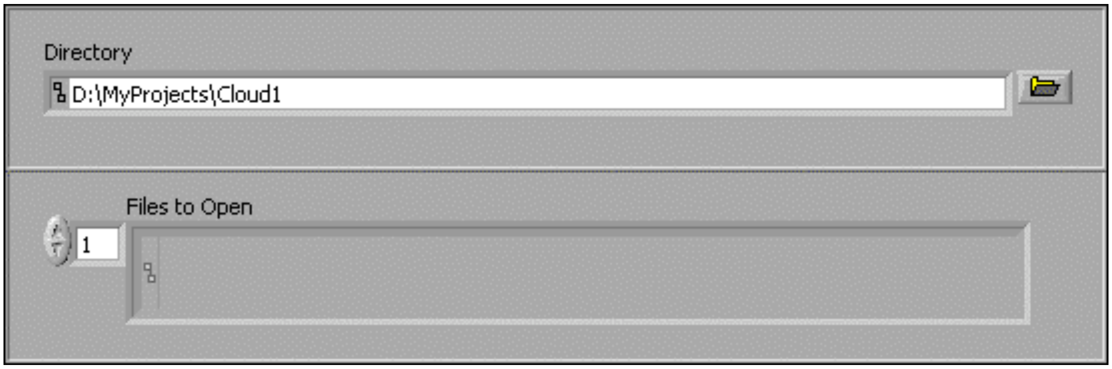
A4.6.1: Get_File_List_PIV_factor.vi

Generates a list of text files to open. It is assumed that the the file to open, "Summary.txt," is stored in a series of folders located in the folder defined by "Directory." Returns in an array containing the file names.

Connector Pane

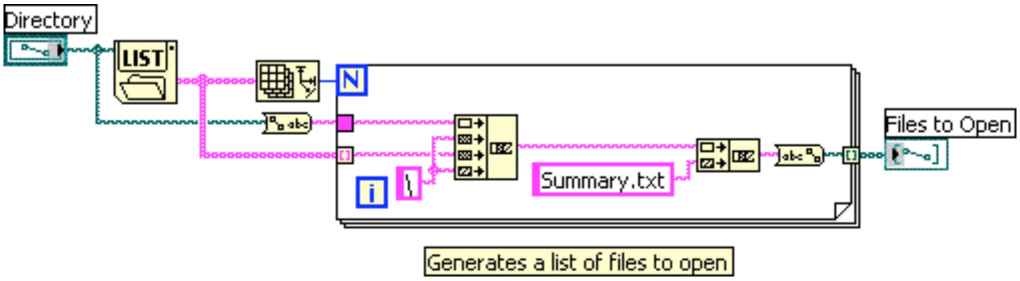


Front Panel


Directory
D:\MyProjects\Cloud1

Files to Open
1

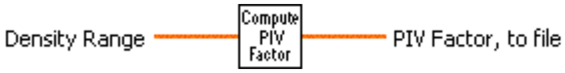
Block Diagram



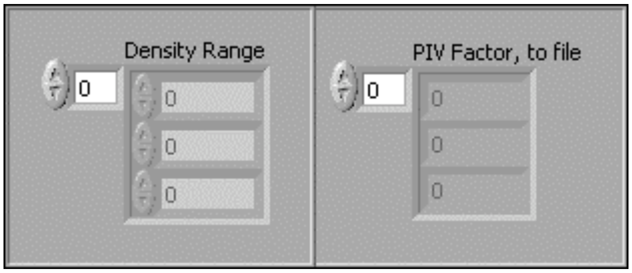
A4.6.2: COMPUTE_PIV_FACTOR_TO_FILE.VI

Interpolates the PIV Factor from the mapping function generated by the simulations of the PIV measurement.

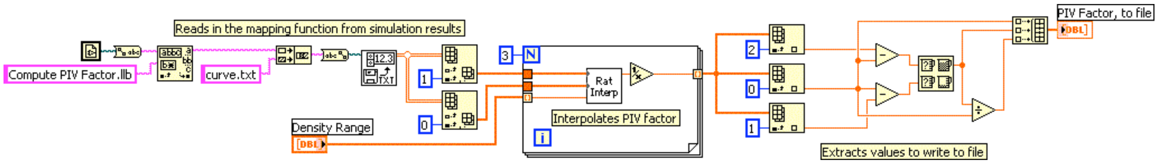
Connector Pane



Front Panel



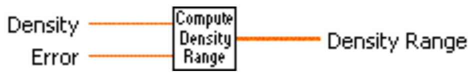
Block Diagram



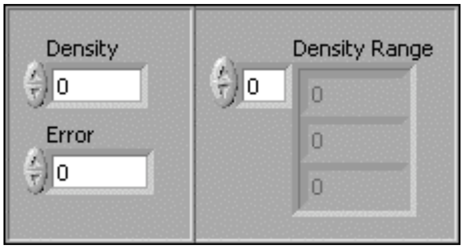
A4.6.3: DENSITY_RANGE.VI

Computes the range of densities, based on the experimental uncertainty.

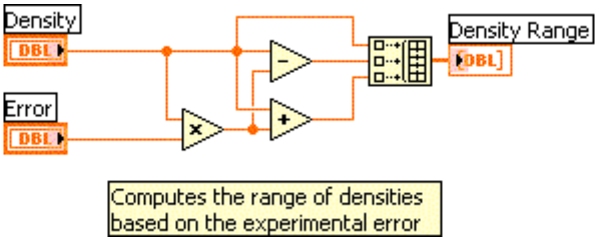
Connector Pane



Front Panel



Block Diagram



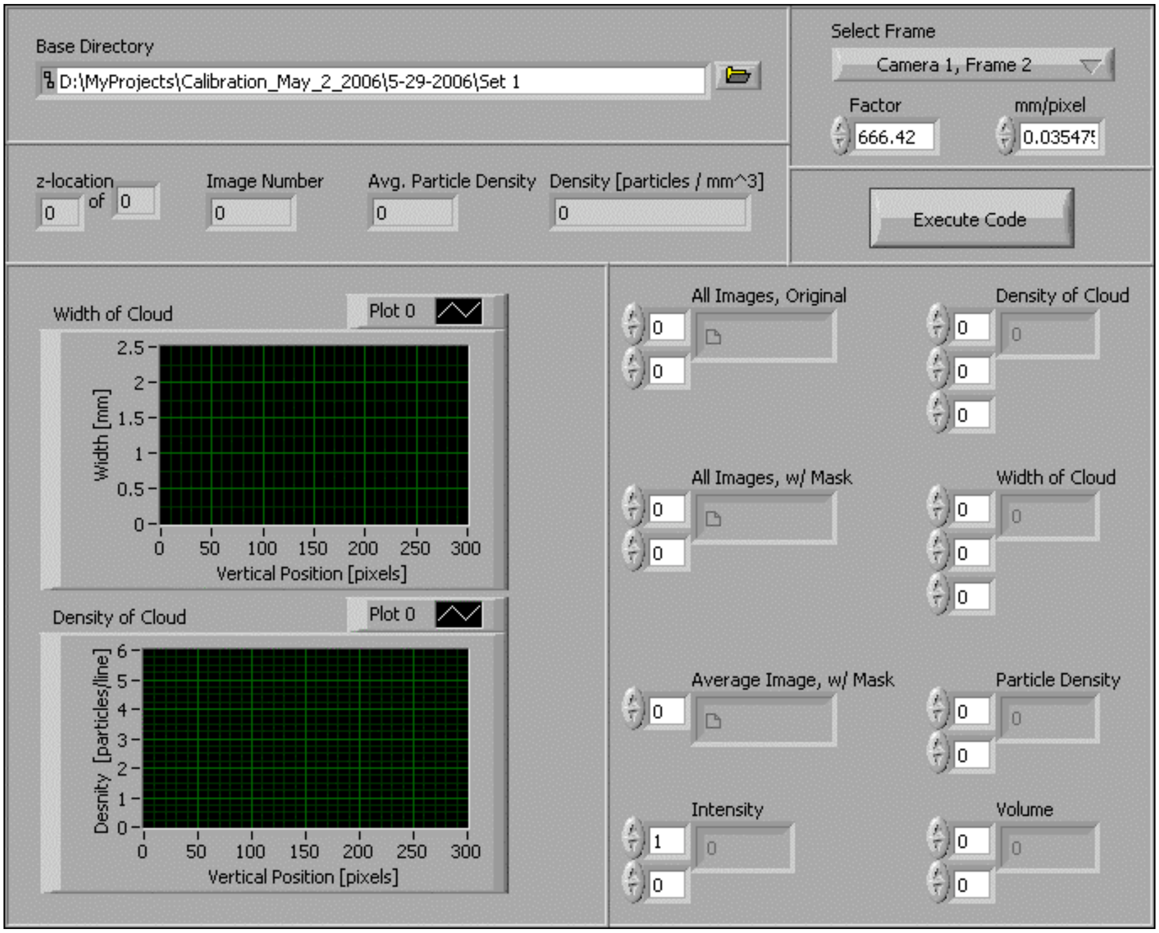
Computes the range of densities based on the experimental error

A4.7: Run_Me_to_measure_particle_density.vi

This code will read in images taken by DaVis and then determine the paticle density of a cloud.

Connector Pane

Front Panel



Base Directory: D:\MyProjects\Calibration_May_2_2006\5-29-2006\Set 1

Select Frame: Camera 1, Frame 2

Factor: 666.42

mm/pixel: 0.03547

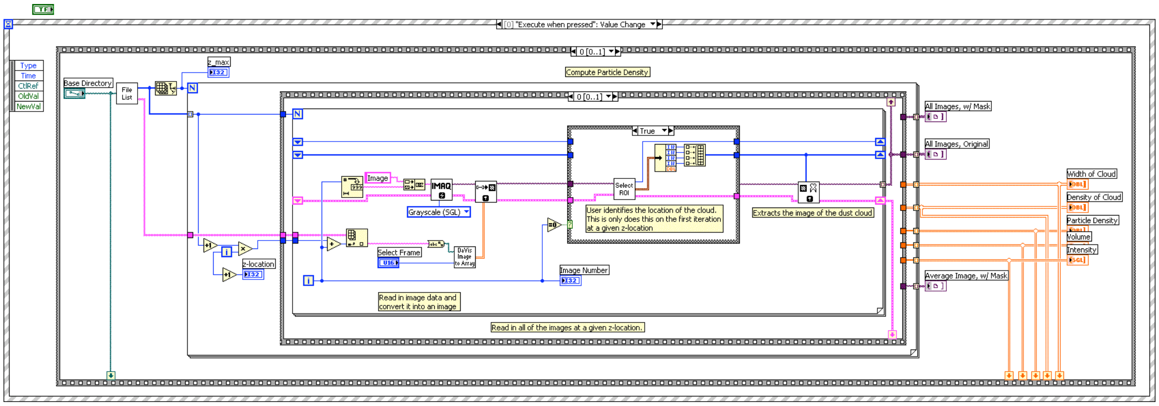
z-location: 0 of 0

Image Number: 0

Avg. Particle Density: 0

Density [particles / mm^3]: 0

Block Diagram



[0] "Execute when pressed": Value Change

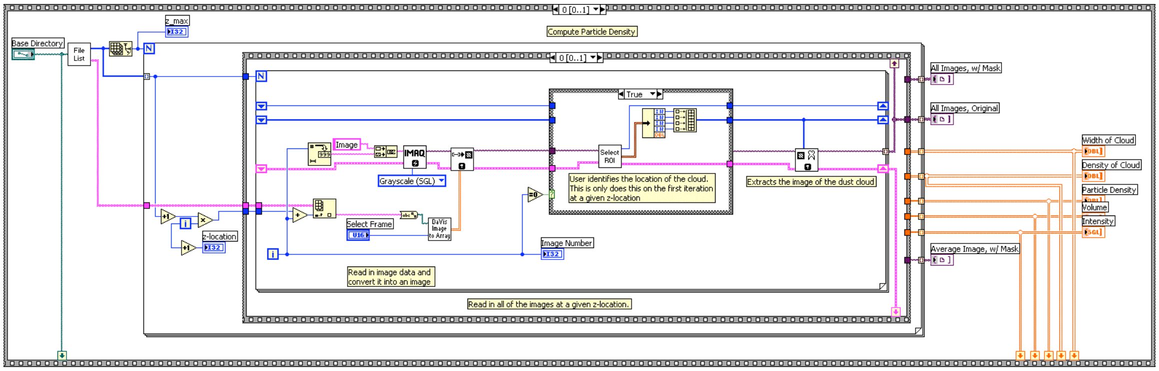
Compute Particle Density

User identifies the location of the cloud. This is only does this on the first iteration at a given z-location

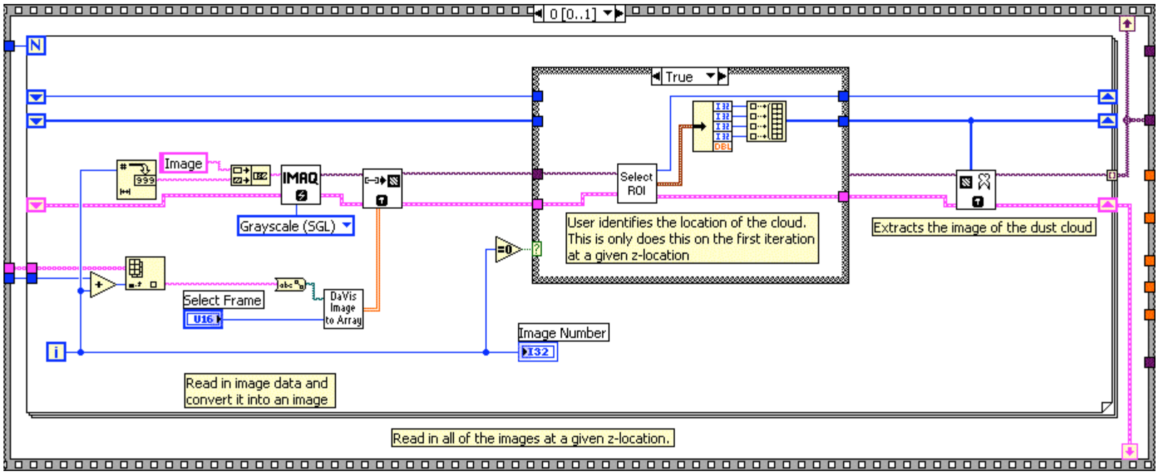
Extracts the image of the dust cloud

Read in image data and convert it into an image

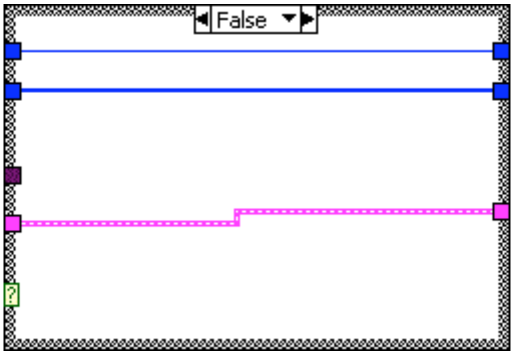
Read in all of the images at a given z-location.



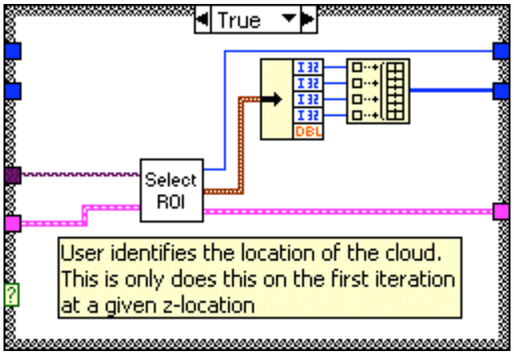
0 [0..1]
z_max
I32
Compute Particle Density
Base Directory
File List
N
0 [0..1]
True
Image
IMAQ
Grayscale (SGL)
Select ROI
User identifies the location of the cloud.
This is only does this on the first iteration
at a given z-location
Extracts the image of the dust cloud
Select Frame
U16
DaVis Image to Array
Image Number
I32
z-location
I32
Read in image data and convert it into an image
Read in all of the images at a given z-location.
All Images, w/ Mask
All Images, Original
Width of Cloud
DBL
Density of Cloud
DBL
Particle Density
Volume
Intensity
SGL
Average Image, w/ Mask



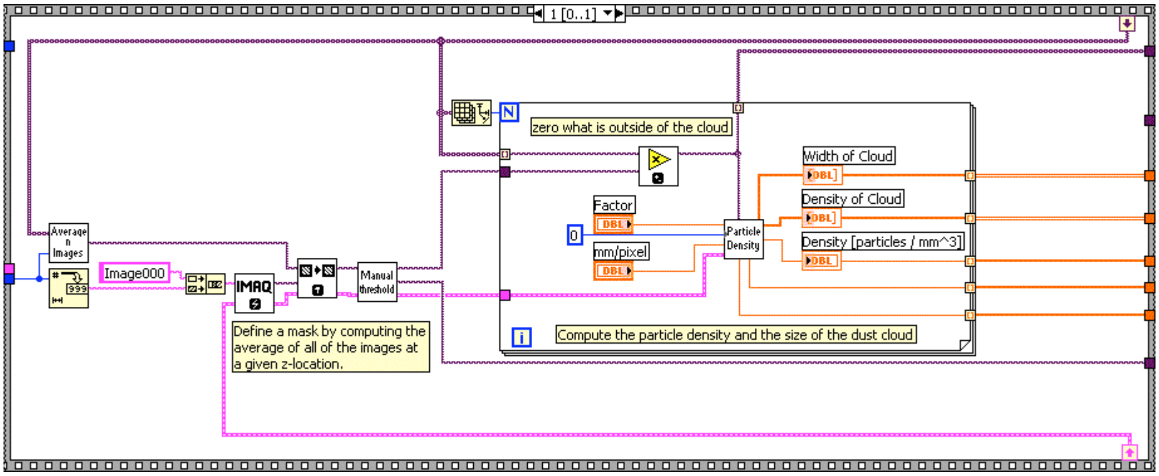
0 [0..1]
N
True
Image
IMAQ
Grayscale (SGL)
Select ROI
I32
I32
I32
I32
DBL
User identifies the location of the cloud.
This is only does this on the first iteration
at a given z-location
Extracts the image of the dust cloud
Select Frame
U16
DaVis Image to Array
i
Image Number
I32
Read in image data and convert it into an image
Read in all of the images at a given z-location.

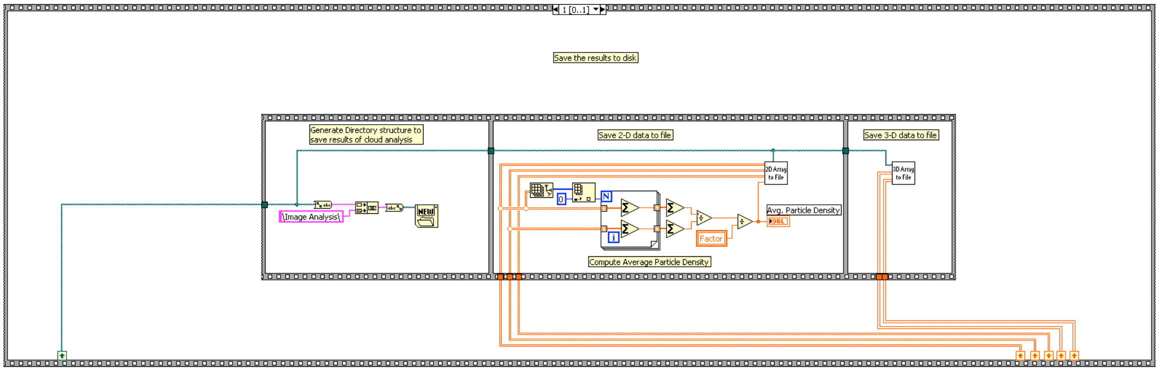


False
?



True
I32
I32
I32
I32
DBL
Select ROI
User identifies the location of the cloud.
This is only does this on the first iteration
at a given z-location
?


1 [0..1]
zero what is outside of the cloud
Width of Cloud
DBL
Density of Cloud
DBL
Density [particles / mm^3]
DBL
Factor
DBL
mm/pixel
DBL
0
Particle Density
Average n Images
999
Image000
IMAQ
Manual threshold
Define a mask by computing the average of all of the images at a given z-location.
N
i
Compute the particle density and the size of the dust cloud



List of SubVIs and Express VIs with Configuration Information

File List

Generate_File_List_for_particle_density.vi

Appendix 4.7.1

DaVis Image to Array

Read_in_Image_data_for_particle_density.vi

Appendix 4.5.5.2

Select ROI

User_defines_ROI_for_particle_density.vi

Appendix 4.7.2

Average n Images

Average n images (float, single set of images).vi

Appendix 4.7.3

Particle Density Compute Particle Density (known factor).vi

Appendix 4.7.4

Manual threshold Manual Thresholding.vi

Appendix 4.3.5.1

2D Array to File Save 2D Data.vi

Appendix 4.7.5

3D Array to File Save 3D Data.vi

Appendix 4.7.6

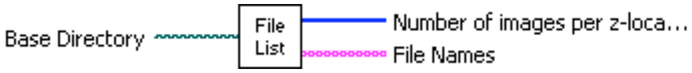
A4.7.1: GENERATE_FILE_LIST_FOR_PARTICLE_DENSITY.VI

Generates a list of image files to open. It is assumed that the camera images are stored in "Base Directory" and the images to be used are stored in the following location:

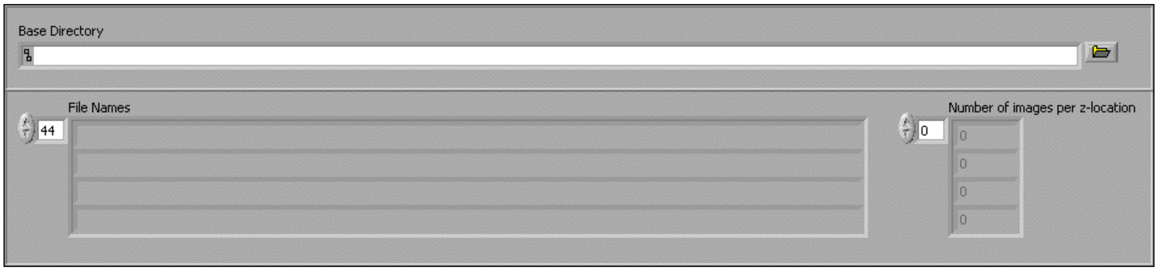
Base Directory: Image@ z-location:Image - Background: Raw -> World Image

Returned in an array of number that stores the number of images at each z-location, "Number of images per z-location: and an array of the file names.

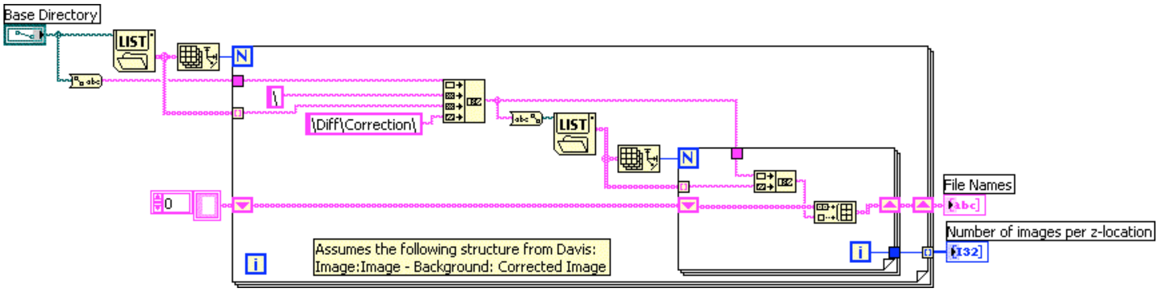
Connector Pane



Front Panel



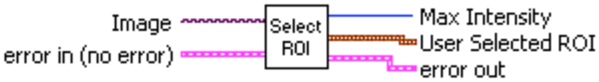
Block Diagram


Base Directory
LIST
N
\
\Diff\Correction\
LIST
N
0
i
Assumes the following structure from Davis:
Image:Image - Background: Corrected Image
i
File Names
abc
Number of images per z-location
I32

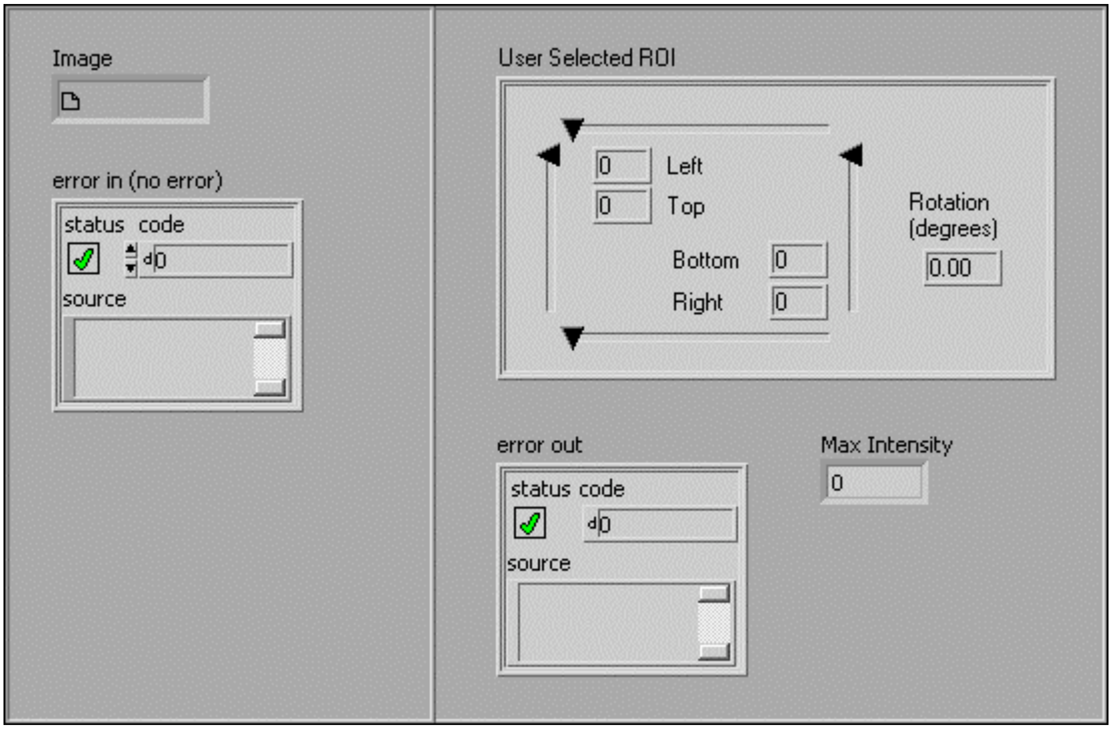
A4.7.2: USER_DEFINES_ROI_FOR_PARTICLE_DENSITY.VI

The user defines a region of interest in "Image".

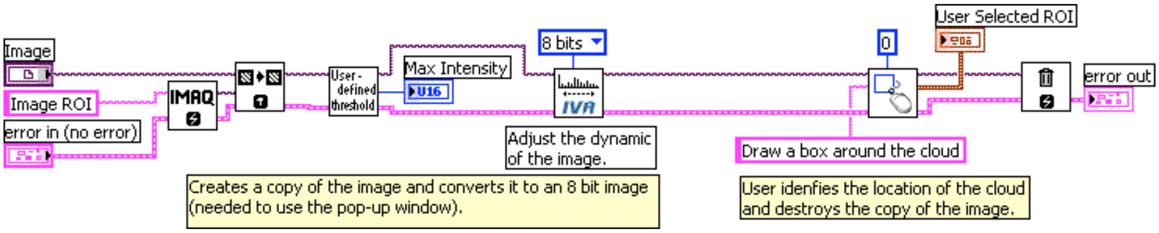
Connector Pane



Front Panel



Block Diagram



Adjust the dynamic of the image.

Creates a copy of the image and converts it to an 8 bit image (needed to use the pop-up window).

Draw a box around the cloud

User idenfies the location of the cloud and destroys the copy of the image.

List of SubVIs and Express VIs with Configuration Information

User-defined threshold

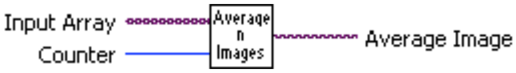
Adjust Image Display Range (fixed value).vi

Appendix 4.5.3.1

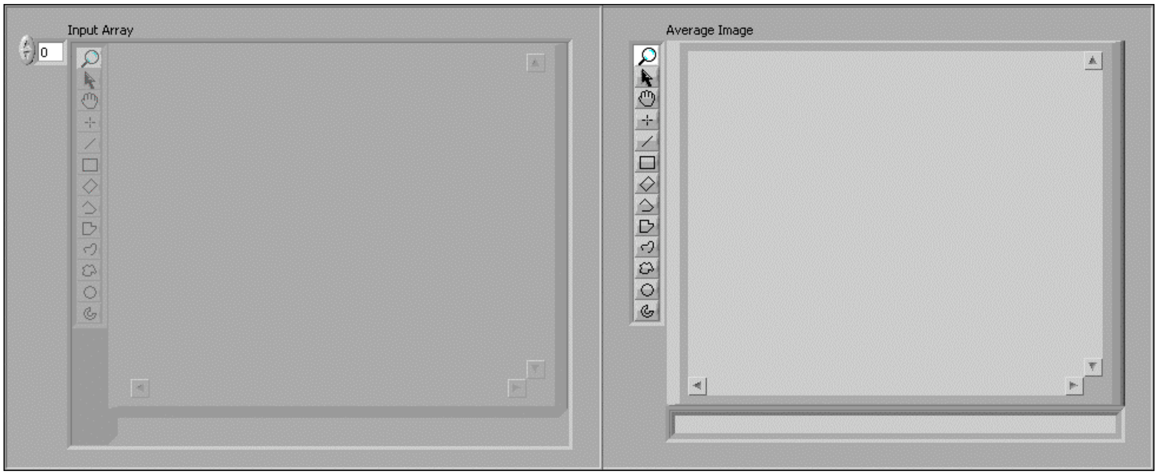
A4.7.3: AVERAGE N IMAGES (FLOAT, SINGLE SET OF IMAGES).VI

Returns the average of the n-images stored in the "Input array"

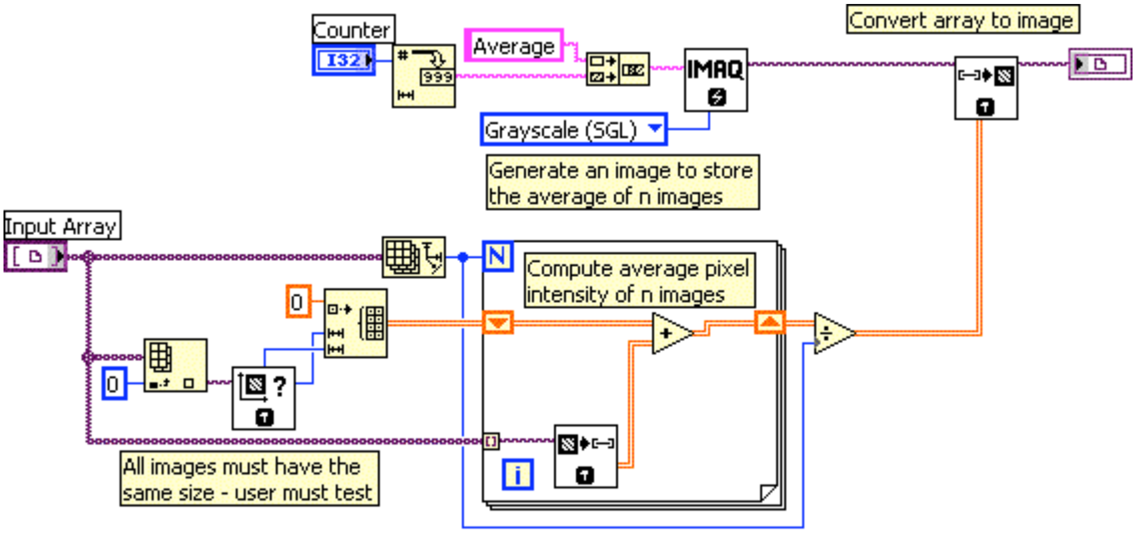
Connector Pane



Front Panel



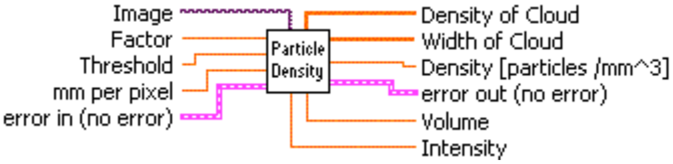
Block Diagram


Counter
I32
Average
IMAQ
Convert array to image
Grayscale (SGL)
Generate an image to store the average of n images
Input Array
N
Compute average pixel intensity of n images
0
0
i
All images must have the same size - user must test

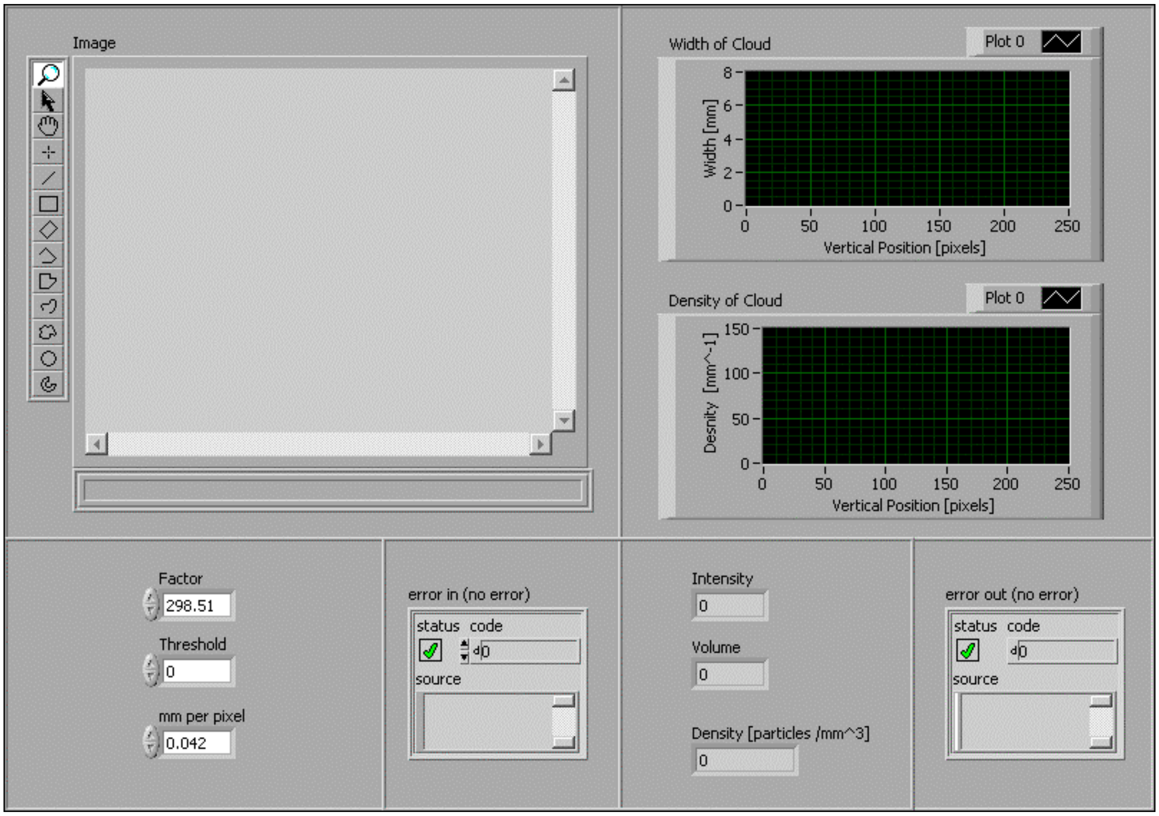
A4.7.4: COMPUTE PARTICLE DENSITY (KNOWN FACTOR).VI

Compute the size and particle density of a dust cloud.

Connector Pane



Front Panel



Factor
298.51

Threshold
0

mm per pixel
0.042

error in (no error)
status code
✓ 0
source

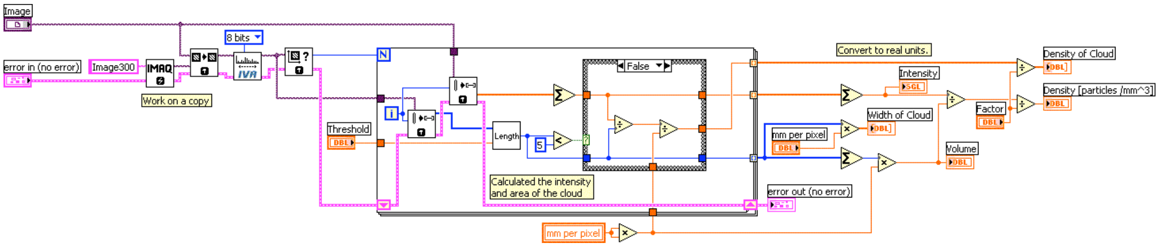
Intensity
0

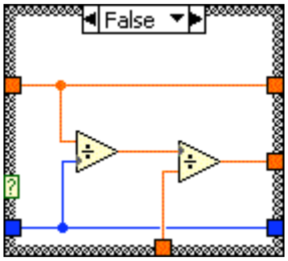
Volume
0

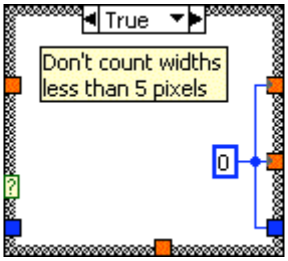
Density [particles /mm^3]
0

error out (no error)
status code
✓ 0
source

Block Diagram







List of SubVIs and Express VIs with Configuration Information

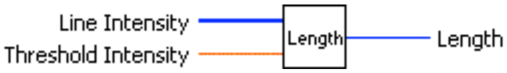
Length.vi

Appendix 4.7.4.1

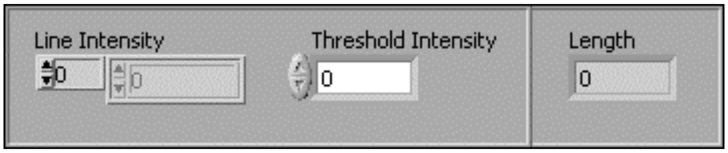
A4.7.4.1: LENGTH.VI

Determines the length of a cloud. It is assumed that the cloud is solid and only counts values that exceed the threshold intensity.

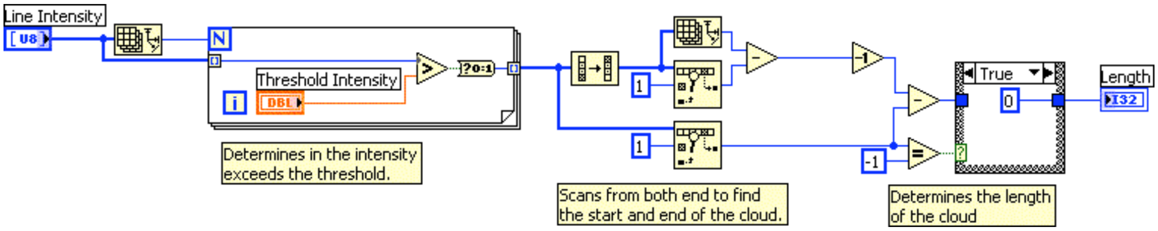
Connector Pane



Front Panel



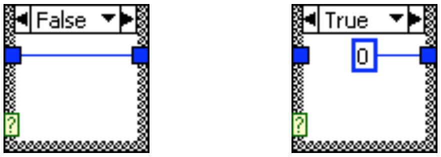
Block Diagram



Determines in the intensity exceeds the threshold.

Scans from both end to find the start and end of the cloud.

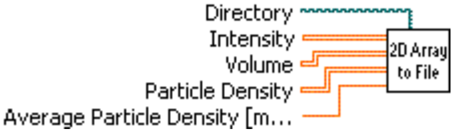
Determines the length of the cloud



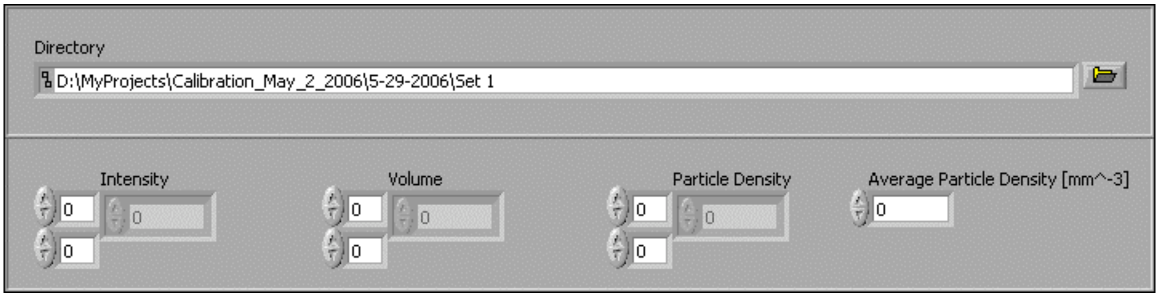
A4.7.5: SAVE 2D DATA.VI

Saves 2D Array to a tab-delimited text file.

Connector Pane



Front Panel



Directory: D:\MyProjects\Calibration_May_2_2006\5-29-2006\Set 1

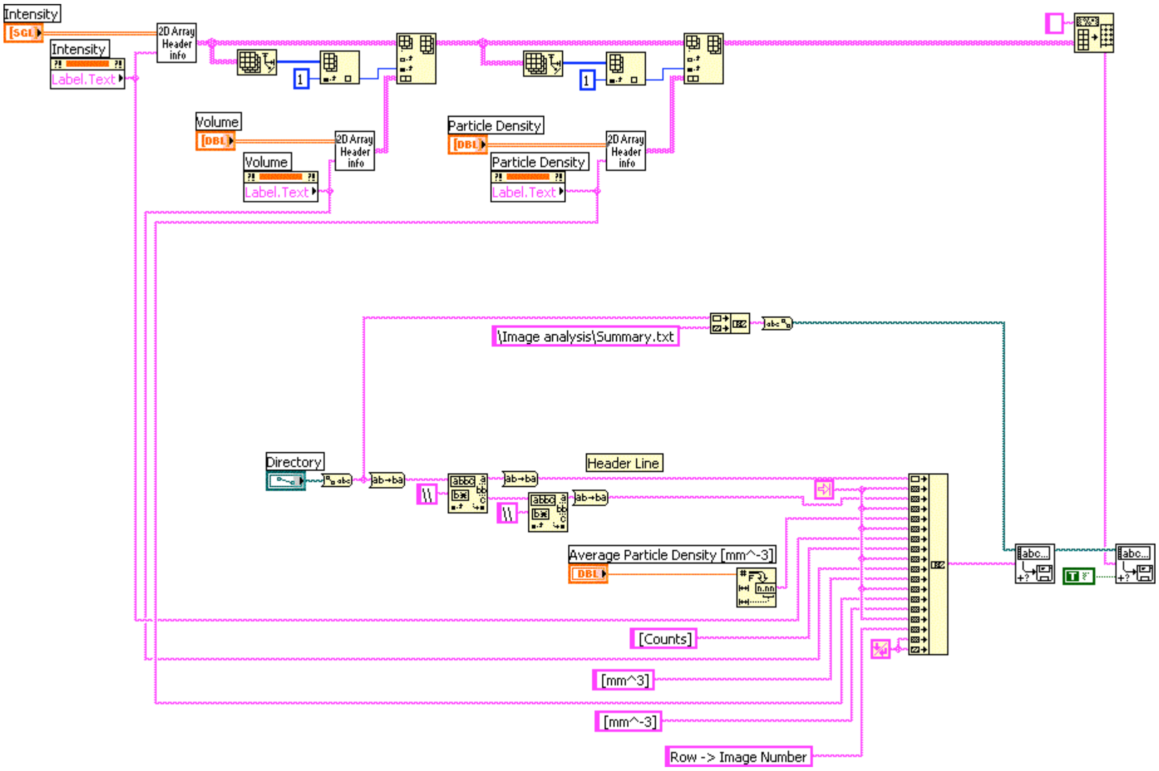
Intensity: 0, 0, 0

Volume: 0, 0, 0

Particle Density: 0, 0, 0

Average Particle Density [mm^-3]: 0

Block Diagram



List of SubVIs and Express VIs with Configuration Information

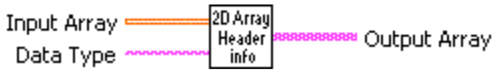
Generate 2D Array to Save.vi

Appendix 4.7.5.1

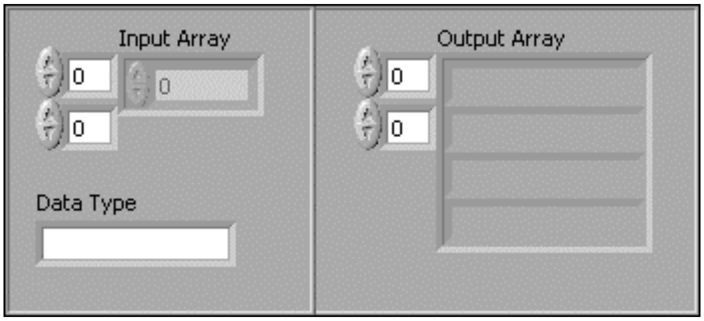
A4.7.5.1: GENERATE 2D ARRAY TO SAVE.VI

Generates the header information for the 2D arrays to be saved to disk.

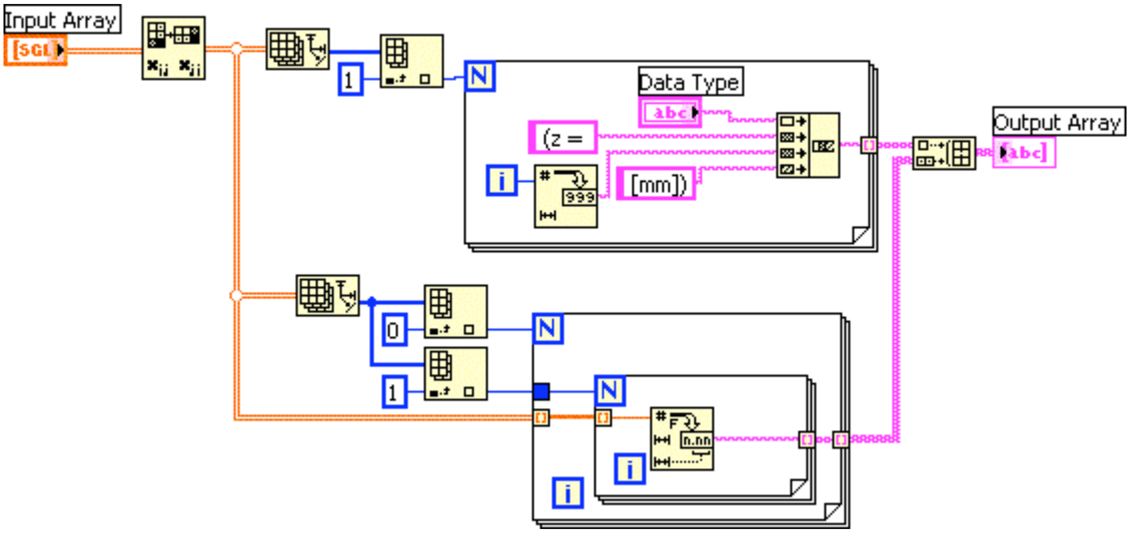
Connector Pane



Front Panel



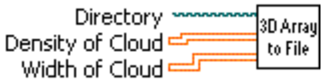
Block Diagram



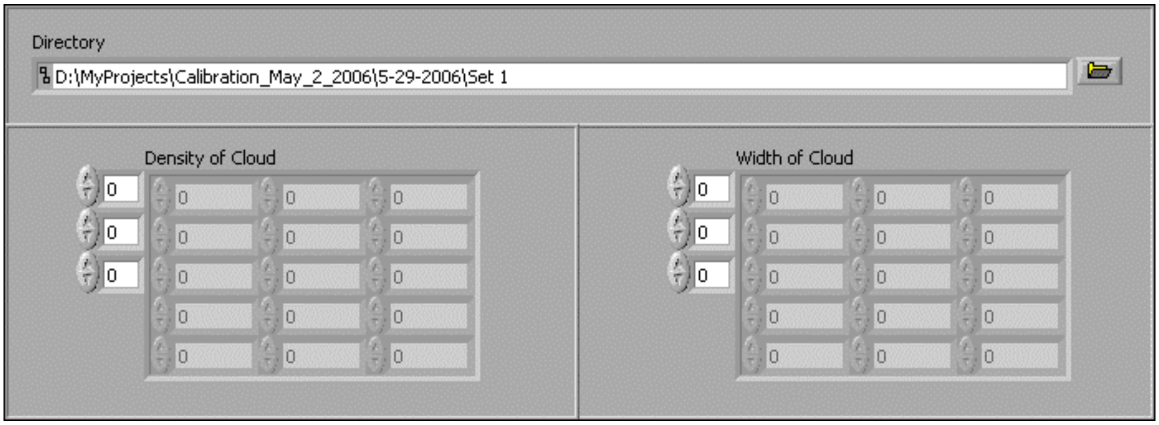
A4.7.6: SAVE 3D DATA.VI

Saves a 3D array to a series of-tab delimited text files. Each page of the array is written to a file.

Connector Pane



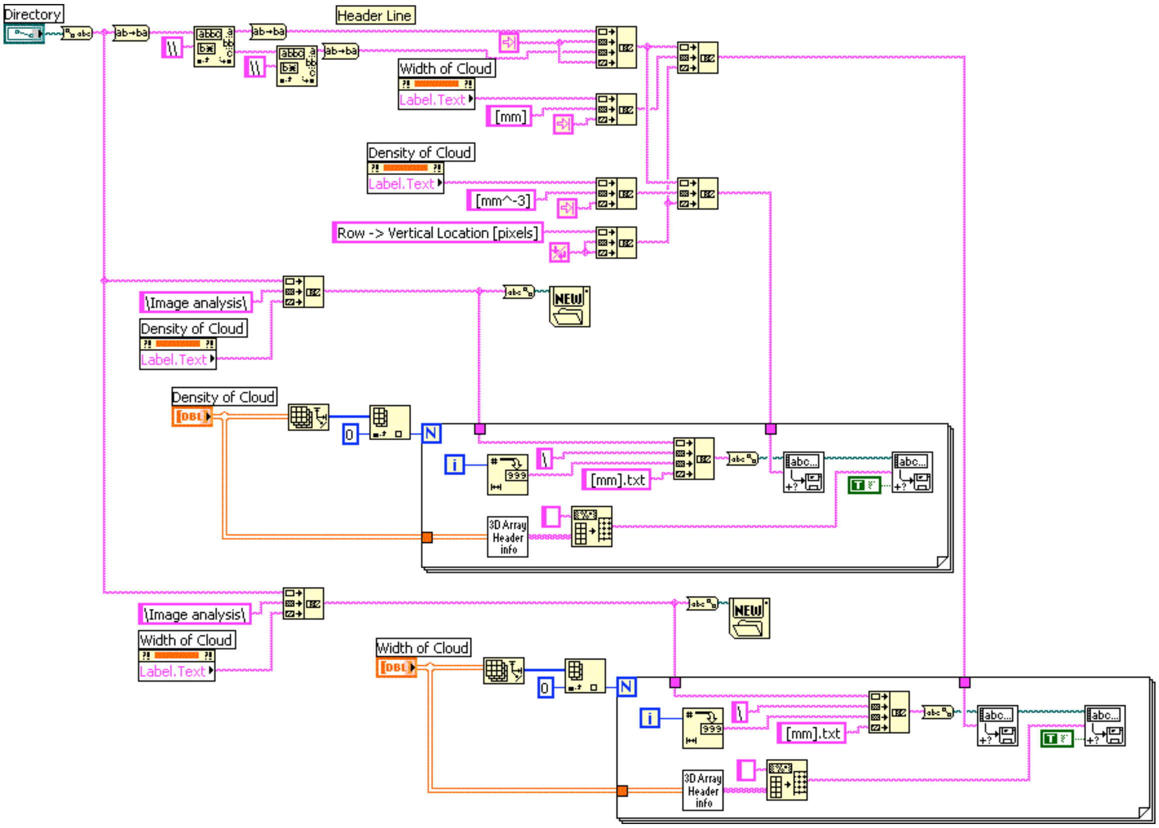
Front Panel



Directory

D:\MyProjects\Calibration_May_2_2006\5-29-2006\Set 1

Block Diagram



List of SubVIs and Express VIs with Configuration Information

3D Array Header info

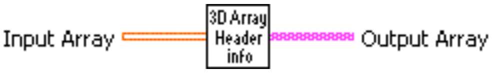
Generate 3D Array to Save.vi

Appendix 4.7.5.1

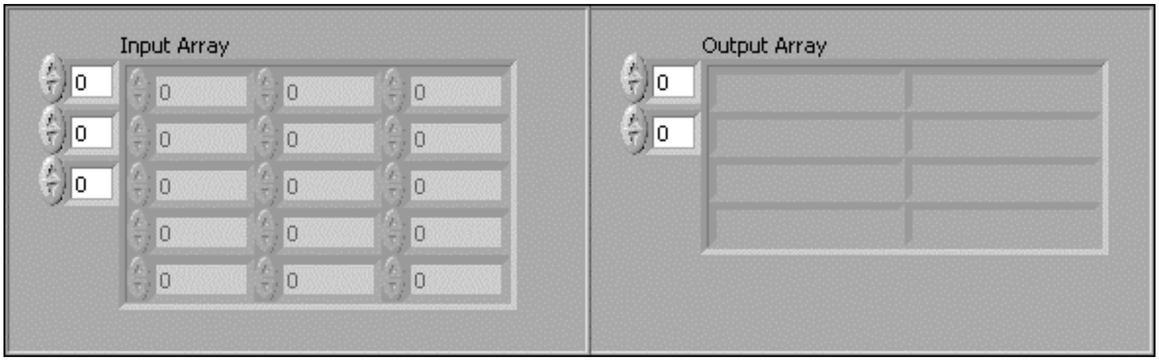
A4.7.6.1: GENERATE 3D ARRAY TO SAVE.VI

Generates the header information for the 3D arrays to be saved to disk.

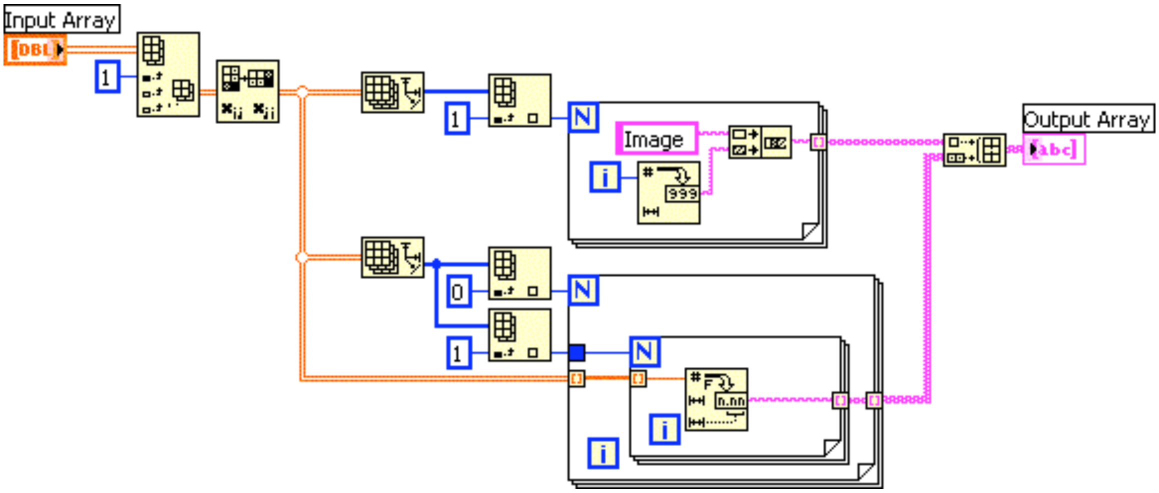
Connector Pane



Front Panel



Block Diagram

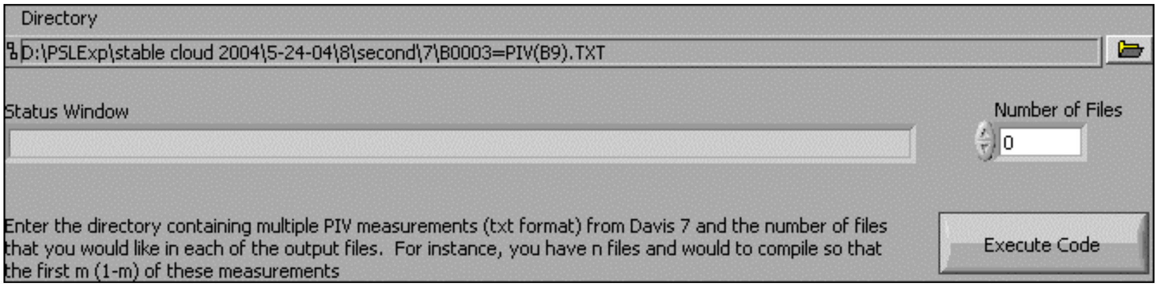


A4.8: RUN_ME_M_TO_N_MEASUREMENTS.VI

This code generate m sets of data containing n measurements.

Connector Pane

Front Panel



Directory

D:\PSLExp\stable cloud 2004\5-24-04\8\second\7\B0003=PIV(B9).TXT

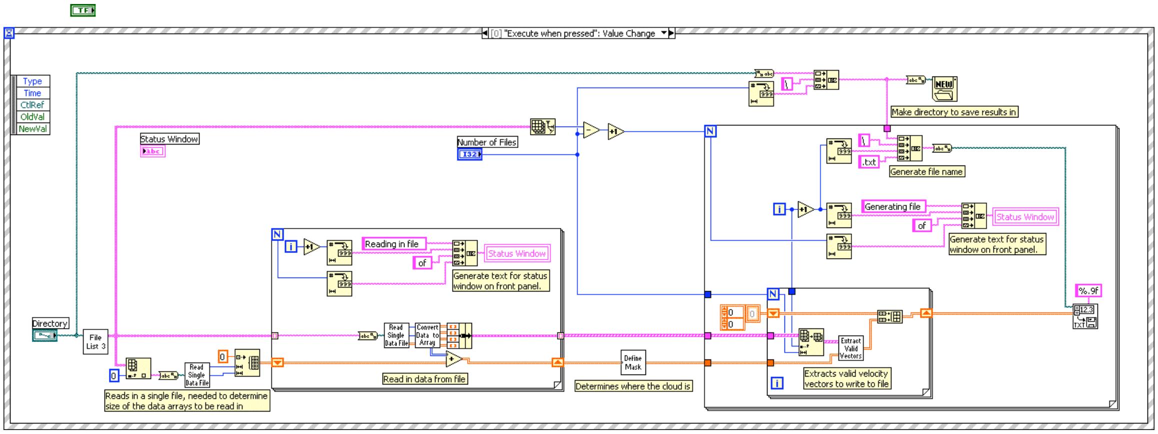
Status Window

Number of Files

0

Enter the directory containing multiple PIV measurements (txt format) from Davis 7 and the number of files that you would like in each of the output files. For instance, you have n files and would to compile so that the first m (1-m) of these measurements

Block Diagram


TF
[0] "Execute when pressed": Value Change
Type
Time
CtlRef
OldVal
NewVal
Status Window
abc
Number of Files
I32
Directory
File List 3
0
Read Single Data File
0
Reads in a single file, needed to determine size of the data arrays to be read in
N
i
+1
Reading in file
of
Status Window
Generate text for status window on front panel.
Read Single Data File
Convert Data to Array
Read in data from file
Define Mask
Determines where the cloud is
\
NEW
Make directory to save results in
N
\
.txt
Generate file name
i
+1
Generating file
of
Status Window
Generate text for status window on front panel.
N
0
0
0
Extract Valid Vectors
Extracts valid velocity vectors to write to file
i
%.9f
12.3
TXT

List of SubVIs and Express VIs with Configuration Information

Read_ Single_Data_File_m_to_n_measurements.vi

Appendix 4.3.6

Select_ROI_m_to_n_measurements.vi

Appendix 4.8.

Extract_ROI_m_to_n_measurements.vi

Appendix 4.8.2

Get_directory_file_list_for_Scattering_efficeincy.vi

Appendix 4.5.2.1

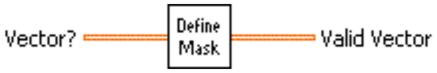
Convert_Data_to_Matrix_m_to_n_measurements.vi

Appendix 4.8.3

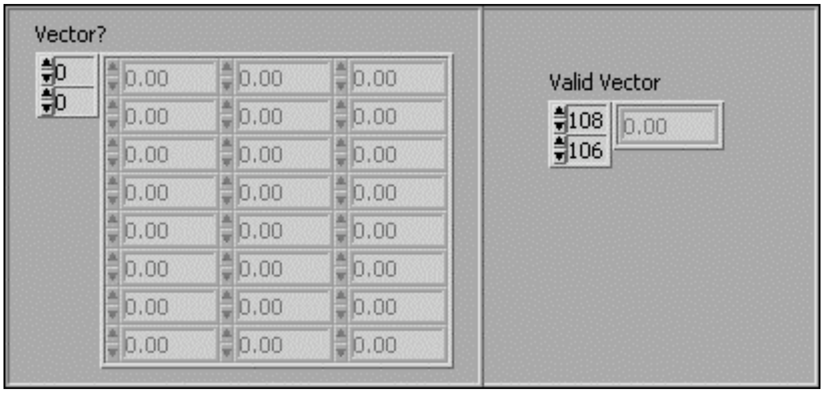
A4.8.1: SELECT_ROI_M_TO_N_MEASUREMENTS.VI

User defines a mask of the cloud.

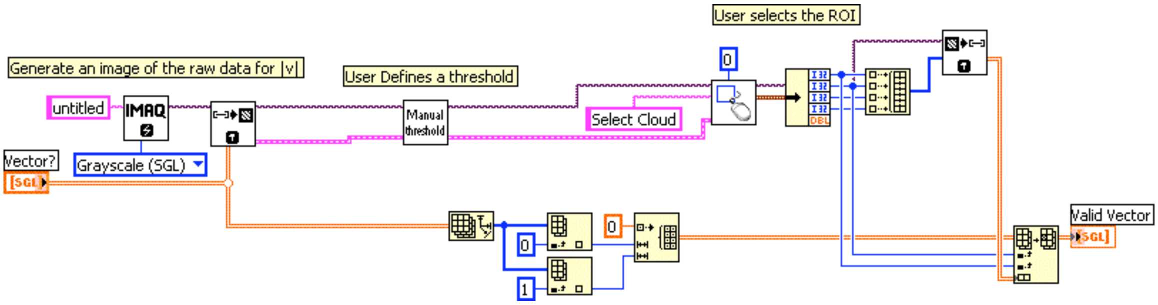
Connector Pane



Front Panel



Block Diagram



List of SubVIs and Express VIs with Configuration Information

Manual threshold

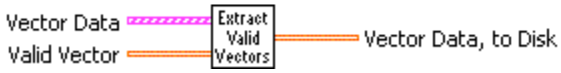
Manual Thresholding.vi

Appendix 4.3.5.1

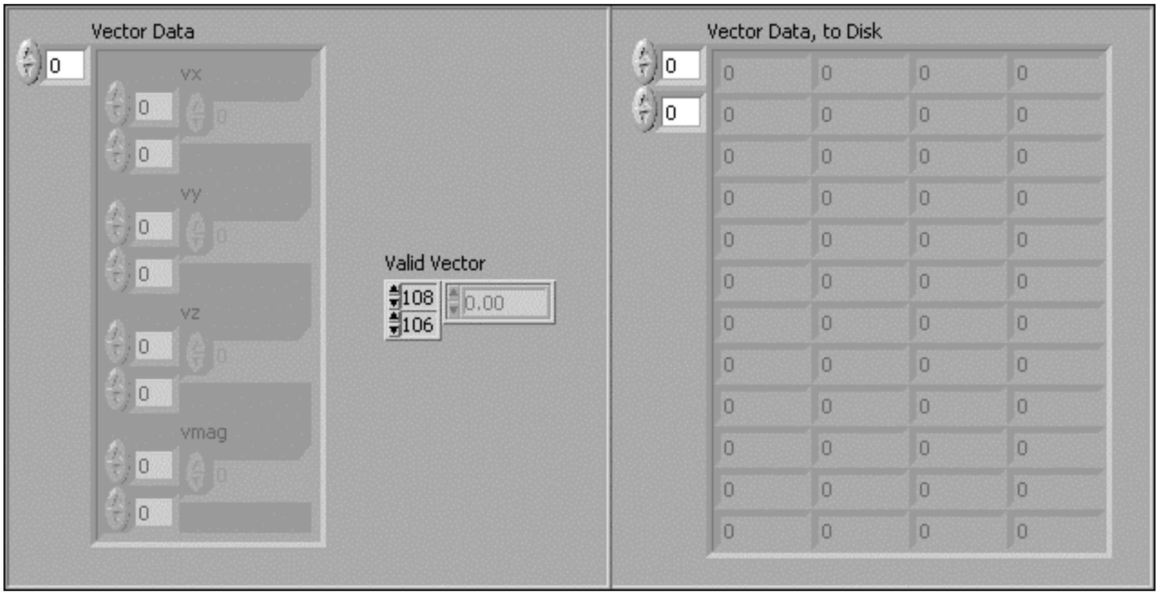
A4.8.2: Extract_ROI_m_to_n_measurements.vi

Extracts valid vectors.

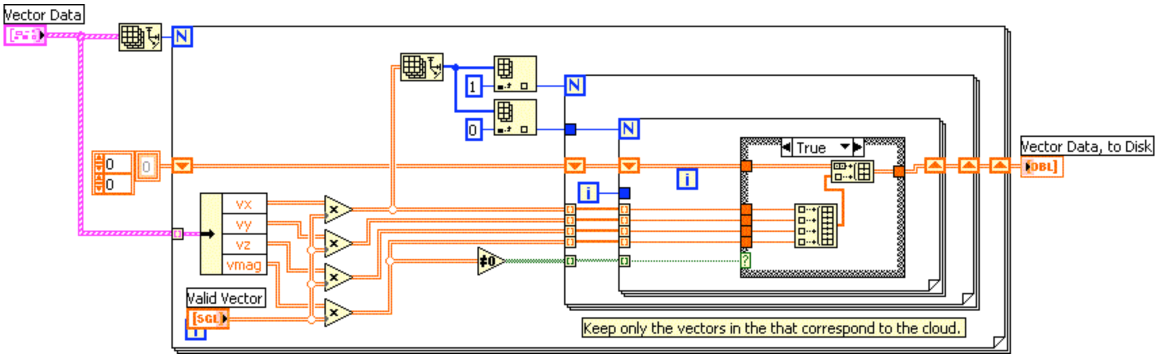
Connector Pane



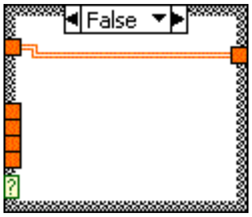
Front Panel

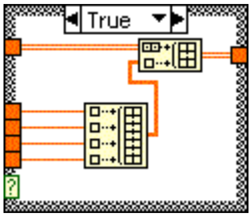


Block Diagram



Keep only the vectors in the that correspond to the cloud.

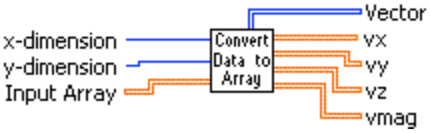




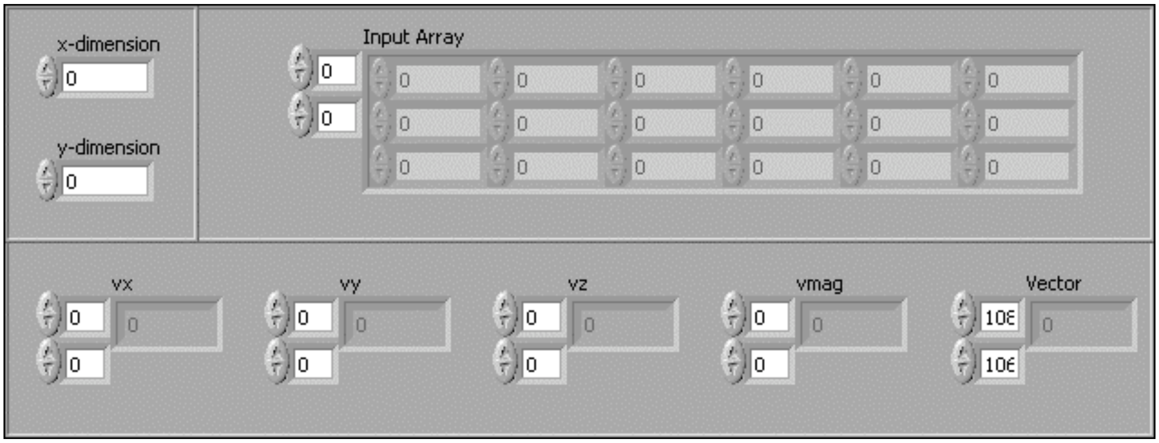
A4.8.3: CONVERT_DATA_TO_MATRIX_M_TO_N_MEASUREMENTS.VI

Converts the vector file (exported from DaVis in .txt format) into a grid form.

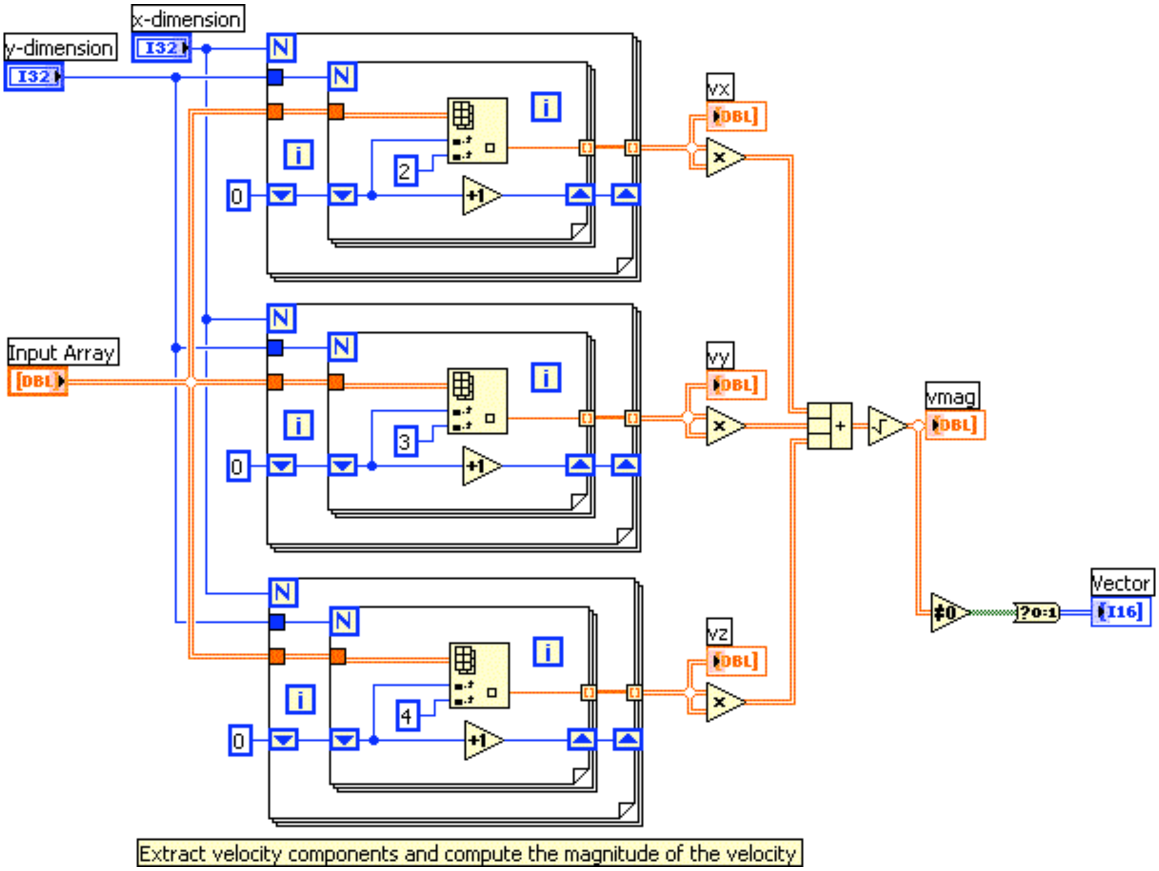
Connector Pane



Front Panel



Block Diagram



y-dimension
I32
x-dimension
I32
Input Array
DBL
N
i
0
2
3
4
+1
vx
DBL
vy
DBL
vz
DBL
vmag
DBL
Vector
I16
≠0
?0:1
Extract velocity components and compute the magnitude of the velocity