DEVELOPMENT OF A SCALABLE REAL-TIME LAGRANGIAN PARTICLE TRACKING SYSTEM FOR VOLUMETRIC FLOW FIELD CHARACTERIZATION

BY

DOUGLAS E. BARKER

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Agricultural and Biological Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2012

Urbana, Illinois

Doctoral Committee:

Professor Yuanhui Zhang, Chair
Professor Richard S. Gates
Professor Anthony M. Jacobi
Professor Brian G. Thomas
Dr. Yigang Sun, Senior Research Engineer, University of Illinois at Urbana-Champaign

Abstract

Lagrangian particle tracking (LPT) is important for the study of turbulence and the inertial behavior of particles in natural and industrial flow fields. This work describes the development of a new scalable real-time LPT system which is able to track particles in large scale 3D turbulent flow fields. A statistical accumulator grid concept was developed to attribute the measured Lagrangian velocities and accelerations of particles to a Cartesian framework for comparison with Eularian based measurements and simulations. Real-time processing was achieved through development of parallel frameworks based on fine and coarse grain problem decomposition for heterogeneous computing architectures. The first framework is based on exposing task and data parallelism through a streaming pipeline within each multi-core processor node. The second framework is based on pipelining temporal data through a unique message passing algorithm in order to utilize large clusters containing hundreds of processor nodes. A sensitivity analysis was completed based on the derivation of measurement uncertainty. It was shown that by utilizing groups of four cameras in the 3D reconstruction process, the overall sensitivity to camera location, image noise and propagated uncertainty could be reduced significantly.

A six camera prototype system was developed and an experimental analysis was conducted to assess the uncertainty in the 3D position, velocity and acceleration measurements of observed particles. The 3D position combined standard uncertainty was 0.16 mm and accuracy was comparable to a caliper in measuring distances between static particles. The velocity measurements were shown to be less than 1% of the calculated value for an object rotating with constant angular velocity. Acceleration was accurate to within 1% for low frame rates but diverged from the calculated value at higher frame rates. The system was used to characterize the dynamic motion of neutrally buoyant helium filled soap bubbles in an unconfined round turbulent jet. The results for axial velocity decay and transverse velocity profile all matched well with widely accepted models. In addition, the profiles of Reynolds shear stress and axial turbulence intensity were in good agreement in both profile and magnitude of those found in literature. The validated LPT system was then applied to a turbulent forced air vortex and particles were successfully tracked with complex 3D paths.

Acknowledgements

I thank my advisor Yuanhui Zhang for his guidance and friendship throughout my six years at the University of Illinois. His devotion to creativity and full support opened the path for me to explore and ultimately find my research passion in computational science and engineering.

I would like to acknowledge my fellow graduate students in Computer Science and Electrical Engineering at Illinois whom tolerated a novice coder and by way of example helped me to develop as a programmer. It was a great learning process and I would like to especially thank Jonathan Lifflander and Anshu Arya for their guidance in Charm++.

I thank my committee members Prof. Gates, Prof. Jacobi, Prof Thomas and Dr. Sun for their critical feedback and for pushing me to get to the heart of the matter.

Finally I want to thank my family and friends for all their love and support.

Table of Contents

Acronyms

CFD	Computational fluid dynamics
CPU	Central processing unit
FFT	Fast Fourier transform
FIFO	First-in-first-out data structure
FPGA	Field programmable gate array
GPU	Graphics processor unit
ILA	Instantaneous Lagrangian acceleration
LPT	Lagrangian particle tracking
OpenCV	Open source computer vision library
PDF	Probability distribution function
PIV	Particle image velocimetry
PTV	Particle tracking velocimetry
RANS	Reynolds averaged Navier-Stokes
SAG	Statistical accumulator grid
SOR	Successive over relaxation
SVD	Singular value decomposition
TKE	Turbulent kinetic energy
VOI	Volume of interest
VTK	Visualization toolkit

1 Introduction

1.1 Motivation

Vision based particle tracking techniques have been widely developed and implemented over the last two decades. A large base of the particle tracking research has focused on hardware and algorithm development for 3D fluid velocity measurement. Particle tracking in this field, often referred to as particle tracking velocimetry (3D-PTV) or Lagrangian Particle Tracking (LPT), is emerging as a vital research tool for its benefits over statistic based image correlation approaches such as particle image velocimetry (PIV). These benefits include the ability to measure velocity fields with higher spatial resolution through sub-pixel accuracy in particle localization (Pereira, Stuer, Graft, & Gharib, 2006) and the ability to reconstruct long particle trajectories with high temporal resolution(Ouellette, Xu, & Bodenschatz, 2006). Higher order spatial and temporal derivatives can be directly evaluated from these long particle trajectories, which enable many new studies in fundamental fluid mechanics including experimental characterization of anisotropic turbulence (B. Lüthi, Tsinober, & Kinzelbach, 2005; Virant & Dracos, 1997).

Often a goal of particle tracking experiments is to build a statistical representation of chaotic object motion through the largest number of observations possible. Naturally there is motivation to develop particle tracking systems with higher resolution and accuracy, which are inherently limited by the hardware (cameras, computers, illumination, etc.) employed. To increase spatial resolution, the number of resolved tracer particles per unit volume must be increased. This can be most readily achieved through three imaging hardware improvements: 1) increase image sensor resolution (> 1 mega pixel) (Hoyer et al., 2005; Ouellette et al., 2006) , 2) add more cameras to increase the chance of particles being observed in three or more view planes (Straw, Branson, Neumann, & Dickinson, 2010; Virant & Dracos, 1997), and 3) Increase the camera frame rate to increase the particle spacing-displacement ratio allowing higher particle densities to be resolved in time (Malik, Dracos, & Papantoniou, 1993) .

The resolution and accuracy of the particle tracking systems are naturally limited by the hardware (cameras, computers, illumination, etc.) they employ. Enormous data generation rates, low transfer rates, and finite camera memory combine to severely limit recording time to only a

few seconds in most cases. For example, a PTV system with four 1 mega-pixel cameras recording at 500 frames per second generates 120 GB in just 60 seconds. This limit leads to convergence issues in statistical analysis of Lagrangian motion (Hoyer et al., 2005). Processing on the computer has also been limited as the data does not fit into the faster Random Access Memory (RAM) for a single processor, which is usually about 4 GB. Such limitations prevent the possibility of efficiently handling the ever growing data sets and eventually achieving real-time analysis.

Fueled by algorithmic development, most particle tracking systems have reached their current limits of accuracy and resolution by riding a steady wave of hardware improvements including; increased sensor resolution (pixels and frame rate), cheaper memory, and faster CPU clock frequencies. Recently, "smart cameras" for machine vision have emerged that utilize embedded architectures such as Field Programmable Gate Arrays (FPGA) and provide real-time image processing capabilities on the camera, reducing data transfer by up to 1000x (M. Kreizer & Liberzon, 2010). However, CPU clock frequencies have begun to level off as limits on heat dissipation have emerged. CPU manufacturers have shifted their focus to increasing the number of cores on multi-core processors. At the same time, accelerator technologies such as the Graphic Processing Unit (GPU) have emerged to introduce a finer grain parallelism that can speed up current algorithms by over 100x (NVIDIA, 2012). At the time of writing this dissertation, the June 2012 Top 500 list of the world's most powerful supercomputers listed three heterogeneous systems, clusters which combine multi-core CPU nodes with GPU accelerators, among the top ten fastest. Leading this pack was the Chinese Tianhe-1 supercomputer, which shows the power of harnessing multiple levels of parallelism with the GPU. However, efficient utilization of these state-of-the art architectures through parallel programming remains a challenging endeavor. For particle tracking systems to continue to improve in accuracy, resolution and processing speed, they must make full utilization of the technologies available.

The question that motivates this proposed research is: Can we utilize the latest parallel processing architectures and computing paradigms to develop a scalable Lagrangian particle tracking system that can observe larger volumes, longer time frames, and visualize the data in real-time?

1.2 Objectives

The main goal of this research is to develop a new Lagrangian Particle Tracking (LPT) platform capable of efficiently utilizing state-of-the-art parallel computing architectures for real-time application. The following objectives were set to meet this goal.

1) **Create *a conceptual design of a real-time particle tracking system:***

   This concept design will include high-level hardware and software specification and define the future real-time LPT instrument that will be prototyped and validated.

2) **Develop *scalable real-time LPT algorithms :***

   Design new particle tracking algorithms for processing on CPU and GPU computing architectures. Show that real-time processing is achievable for hundreds of particles at a frame rate of 120 frames per second. Show that the new parallel algorithms are scalable up to hundreds of processors and an arbitrary number of cameras.

3) **Validate *the real-time particle tracking platform:***

   Derive procedures to determine measurement uncertainty and validate the 3D position, velocity and acceleration measurements with experimental analysis. Complete an uncertainty and sensitivity analysis to identify important factors impacting experimental design. Show that the real-time LPT platform can accurately track particles in turbulent flow fields.

4) **Apply *the real-time platform to analyze inertial particle motion in turbulent flows:***

   Develop guidelines for setting up the LPT system and conduct the experiment to measure particle acceleration and velocity in unconfined stationary flows of a round turbulent jet and an unconfined forced vortex. Compare the experimentally measured particle velocity statistics with literature.

1.3 **Approach**

Much research in the area of optical based particle tracking focuses on developing increasingly complex algorithms in order to gain resolution, reduce errors and uncertainty, and improve system usability, as hardware was extremely limiting. In the past this approach was necessary for particle tracking techniques to provide value to the research community as the camera and computing technology was indeed the limiting factor. Now, however, the pace of technology has far surpassed our ability to utilize it for Lagrangian particle tracking with current algorithms.

In this research, it is hypothesized that by integrating state-of-the art technologies (parallel computing paradigms, heterogeneous computing architectures, smart cameras, and open source programming tools) the particle tracking limitations can be solved in a scalable way. Therefore our approach is fundamentally different and can be summarized by the following four guidelines.

1. **Use more cameras** to image the flow from more points of view
    a. Addresses the particle overlap issues
    b. Addresses the observable volume issue
2. **Use higher frame rate cameras** to minimize the particle displacement in each frame
    a. Addresses the tracking issues and reduces ambiguity
    b. Addresses the ability to track higher densities of seed particles
3. **Use smart cameras** to do image processing and segmentation
    a. Addresses the data transfer bottleneck between camera and computer
    b. Addresses the data storage issue by eliminated full image storage
4. **Use parallel computing** to handle the data
    a. Addresses the influx of data from increasing the number of cameras/frame rates
    b. Addresses data storage issues through real-time processing and discarding unneeded intermediate data (images, etc.).

With this approach, it is hypothesized that by "over observing" the flow through many camera angles and at higher frame rates the particle tracking task can be completed with simpler and faster algorithms to facilitate real-time processing. This dissertation is comprised of seven

chapters. Chapter two will cover the background literature which guides the algorithm selection and system design. Chapter three discusses algorithm development for real-time LPT and discusses their implementation. Chapter four covers the development of the real-time processing framework where the LPT algorithms are parallelized and tested on shared memory systems (multi-core processors and GPUs) and on large scale distributed memory clusters. Chapter five discusses the hardware used in the real-time LPT system prototype and covers the derivation of measurement uncertainty followed by a sensitivity analysis to help guide experimental setup. Chapter six contains the validation tests for the measurement of particle position, velocity and acceleration. The second half of Chapter six covers the application of the LPT system to characterize particle motion in turbulent round jet and unconfined forced vortex air flow fields.

1.4 **Justification**

Lagrangian analysis of turbulence is one of the most important applications of Lagrangian particle tracking systems. Current LPT techniques are limited to small volumes and therefore limit the range of Reynolds numbers that can be observed in the Lagrangian reference frame (Toschi & Bodenschatz, 2009). One of the goals for researchers in this field is to be able to observe turbulent motion over the entire inertial range of eddies. To observe these characteristics at higher Reynolds numbers, this means reducing the kinematic viscosity of the fluid or drastically increasing the length scale (Toschi & Bodenschatz, 2009). Therefore, an LPT system that is scalable to large volumes has great potential to make an immediate impact in the field of turbulence research. Current commercial particle tracking velocimetry (PTV) systems are only capable of measuring flow field velocities, without particle trajectory reconstruction, in small scale flows (<0.001 m^3) due to imaging and illumination system limitations (Toschi & Bodenschatz, 2009). Several studies have been published for large volume application of prototype 3D-PTV systems, however they were limited in scale by the number of cameras available (Biwole, Yan, Zhang, & Roux, 2009; Lobutova, Resagk, & Putze, 2010).

One major aspect of the new LPT platform will be the ability to scale up to an arbitrary number of cameras. Currently most PTV or LPT systems only work with a fixed number of cameras between three and four. Adding more cameras increases the observable volume and reduces the

number of ambiguities when matching particle images between cameras (H.G. Maas, 1992). Straw proved this concept by synchronizing up to eleven low cost cameras to track a few flies in motion (Straw et al., 2010). Solving the multi-camera correspondence problem in real-time for a large number of tracer particles remains a challenge.

Through this research a complete real-time LPT system has been developed including: particle detection, multi-camera correspondence, 3D reconstruction, tracking and interactive visualization. Real-time processing alleviates data accumulation in memory and eliminates current limits on experimental duration, which allows better statistical characterization of chaotic phenomena, such as turbulence and inertial particle motion. By redesigning these algorithms for massively parallel processing on modern computing architectures, the LPT system can readily scale-up with rapidly improving camera technology.

2 Literature review

2.1 Lagrangian study of particles in turbulent flow

One of the most significant applications of Lagrangian Particle Tracking (LPT) is the study of particle motion in turbulent flow fields (Toschi & Bodenschatz, 2009). The study of inertial particle dynamics in turbulent flow has direct relevance to basic sciences related to climate and atmospheric sciences in addition to many industrial applications including combustion of liquid fuel sprays. In fundamental research, LPT has potential to shed light on the nature of inertial particle dynamics in fluid turbulence. Ni (2012) applied particle tracking techniques to evaluate Lagrangian acceleration in turbulent Rayleigh-Bénard convection, one of the most studied fundamental flow fields (Ni, Huang, & Xia, 2012). Mercado (2012) used LPT to study Lagrangian statistics of light micro-bubbles in water (Mercado, Prakash, Tagawa, Sun, & Lohse, 2012).

Particles in a fluid flow can behave inertially for two reasons, 1) their density is greater or less than the surrounding fluid, and 2) their size is finite and large compared to the eddy dissipation scale (Mercado et al., 2012; Ouellette, O'Malley, & Gollub, 2008; Qureshi, Bourgoin, Baudet, Cartellier, & Gagne, 2007; Toschi & Bodenschatz, 2009). A pair of useful dimensionless numbers Φ and Γ can be used to characterize a particle's size and density relative to the dissipation scale and flow density as defined in equations 2-1 and 2-3 respectively (Qureshi et al., 2007).

$$\Phi = \frac{d}{\eta} \tag{2-1}$$

$$\Gamma = \frac{\rho_p}{\rho_f} \tag{2-2}$$

Where d is the diameter of the particle, ρ_p is the density of the particle, ρ_f is the density of the fluid, $\eta = \left(\nu^3/\varepsilon\right)^{1/4}$ and is the eddy dissipation scale (also called the Kolmogorov length), ν is the kinematic viscosity of the fluid, and ε is the average dissipation rate of turbulence kinetic energy per unit mass.

Traditionally, the dimensionless Stokes number is considered a good indicator of how well a certain type of particle will follow the underlying flow field. The Stokes number, $St = \tau_p / \tau_\eta$, is defined as the particle's response time τ_p, divided by the characteristic eddy time $\tau_\eta = (\nu/\varepsilon)^{1/2}$. At small Stokes numbers, particles are believed to move as ideal fluid tracers and their trajectories match those of infinitesimally small fluid particles. At larger Stokes numbers, St > 0.1 (Ouellette et al., 2008), the particles' trajectories are believed to deviate significantly from the underlying flow. This deviation has been observed as preferential concentration of inertial particles, where particles accumulate in areas of the flow field after being expelled from small eddies, and changes in the probability distribution function of acceleration variance (Ouellette et al., 2008).

However, recent research has called into question the significance of the Stokes number at predicting how inertial particles will behave in a turbulent flow field (Bourgoin, Qureshi, Baudet, Cartellier, & Gagne, 2011; Ouellette et al., 2008; Qureshi et al., 2007; Toschi & Bodenschatz, 2009). Qureshi and coworkers (2007) studied the dynamics of millimeter scale helium filled soap bubbles in grid generated turbulence for bubble size and density ranges of $10 < \Phi < 30$ and $1 < \Gamma < 70$. They found that the particle single point velocity statistics, characterized by the probability distribution functions of mean particle velocity and variance, were not affected by changes in bubble size or density compared with ideal fluid tracers. In addition, they observed that the single point acceleration mean PDF was also not significantly impacted. Only the variance of acceleration was significantly impacted by the fact that the particles were inertial (Bourgoin et al., 2011; Qureshi et al., 2007). Ouellette (2008) conducted an experimental study with an LPT system to simultaneously observe the motion of small (d = 80 μm, St $\approx 10^{-4}$) and large (d = 2 mm, St $\approx 10^{-2}$) inertial particles (Ouellette et al., 2008). From their study they confirmed that there was no observable difference in the single point velocity or acceleration statistics of the particles and that acceleration variance was significantly affected. In addition, Ouellette observed significant differences in the actual trajectories of the large particles compared with the smaller tracers. Since the single point PDFs of mean velocity and acceleration were unaffected by the size or density of the particles within the range tested , they concluded that inertial effects do not scale with Stokes number as is traditionally thought (Ouellette et al., 2008).

Although it was confirmed that finite-size inertial particles do follow different paths than much smaller flow tracers (Ouellette et al., 2008), the single point velocity and acceleration statistics of inertial particles tend match those of the underlying flow. Therefore, in anisotropic turbulent flows, the location of turbulence intensities observed through the motion of inertial particles will be displaced from the true location of those values. However, the scale of this effect is still an open question in the field and there may be flows where large neutrally buoyant particles, such as helium filled soap bubbles, may provide a close approximation to the characteristics of the flow field. Kerho and Bragg (1994) found through experimental and numerical analysis that neutrally buoyant helium soap bubbles, which ranged in diameter from 1 to 5 mm, could follow the flow streamlines around an airfoil if their density was in fact equal to the surrounding fluid (Kerho & Bragg, 1994) . However, they also observed deviations from the streamlines proportional to the density ratio when neutral buoyancy was not achieved. The study of inertial particles dynamics in turbulent flow remains an open question and traditional thought of the Stokes number as a key indicator for particle behavior in turbulent flow has been called into question. LPT tools can play a role in better understanding the interaction of inertial particles in turbulence (Toschi & Bodenschatz, 2009).

2.2 Particle tracking techniques

This section will provide a brief overview common particle tracking methods. While many variations of 3D Lagrangian Particle Tracking (LPT) techniques have been developed for over two decades, most share the same basic underlying approach (H. G. Maas, Gruen, & Papantoniou, 1993; Shindler, Moroni, & Cenedese, 2010). This basic particle tracking technique is generally divided into six major tasks as given below (Beat Lüthi & Liberzon, 2012). The hardware components of a traditional particle tracking system (Figure 1) include: particle seeding and illumination systems, cameras, data transfer and storage, and data processers. In this traditional system, particles are seeded into the flow field and imaged by two to four cameras. All images obtained during an experiment are digitized and stored to a hard drive for offline processing.

a) **Image acquisition and processing:** Images of a particle laden flow from three or four cameras are first processed to segment the particle images from the background

b) **Particle detection and centroid localization:** The particle "blob" images are detected and analyzed to determine the centroid of individual particles in pixel coordinates.

c) **Image correspondence:** The image correspondence problem between particles imaged in multiple camera planes is solved using epipolar geometry.

d) **3D reconstruction:** The matched 3D location of each particle is reconstructed in object space from the corresponding 2D pixel locations and camera calibration parameters.

e) **Temporal tracking:** The reconstructed 3D particle data is analyzed from frame to frame to identify the temporal correspondence of particles in object space.

f) **Post processing and visualization:** The reconstructed trajectories are used to calculate the particles' instantaneous velocity and acceleration. The Lagrangian velocities and accelerations can be interpolated to a fixed grid for calculation and visualization of Eularian properties (velocity, vorticity, etc.).

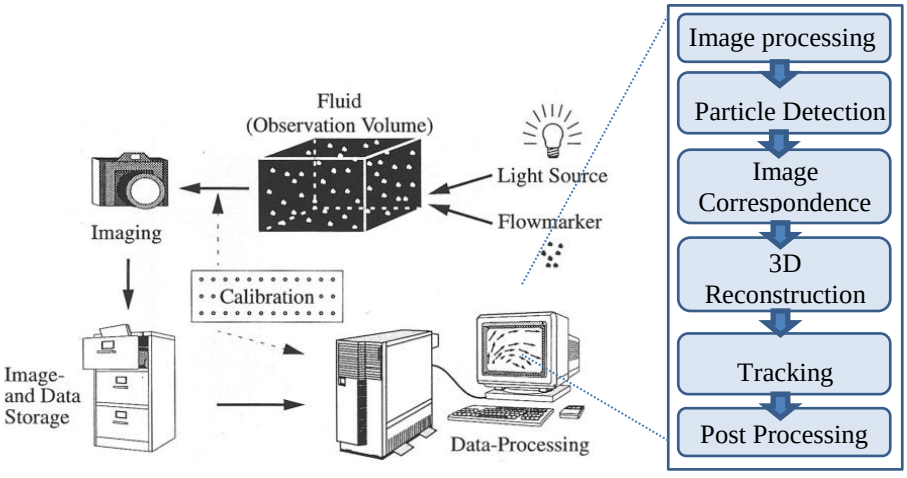

Figure 1: Particle tracking system diagram; image credit (Beat Lüthi & Liberzon, 2012)

2.2.1 **Image processing, particle detection and centroid localization**

In 3D particle tracking experiments, the first step is processing the image to segment the particle "blob" images from the background and identify the image centroids of each. A common method for image segmentation and particle localization consists of the following: 1) remove the background through image subtraction, 2) apply a high-pass filter and Gaussian blur to remove noise, 3) perform a pixel intensity threshold operation to create a binary image, and 4) identify the centroid through a mass-weighted average or a Gaussian estimator (Biwole et al., 2009; H. G. Maas et al., 1993). The difficult task in this technique is selecting the proper threshold value which maximizes the number of particle identifications in the presence of image noise, non-uniform illumination, and overlapping particle images (Shindler et al., 2010). This crucial step defines the spatial density of particle information and amount of measurement error propagated through to the 3D position reconstruction step. Ouellette (2006) tested several particle image centroid localization methods: weighted average, 1D and 2D Gaussian estimators, and a neural network approach (Ouellette et al., 2006). Their work showed that for low noise images the 1D Gaussian estimator is more accurate and computationally efficient, but for images containing noise the neural network approach was superior. Shindler (2010) created a feature based particle identification and centroid calculation technique, using the optical flow equation and the 1D Gaussian estimator. This method was shown to perform better in the presence of noise and non-uniform illumination than the basic intensity threshold and weighted average centroid identification technique (Shindler et al., 2010). Currently, a single all-purpose algorithm optimal for all cases does not exist.

2.2.2 **Image correspondence and 3D reconstruction**

The problem of identifying image correspondence between particles located in different image planes has been solved traditionally using the geometric epipolar constraint which has been thoroughly discussed in the publication by (H. G. Maas et al., 1993) and derived in detail by Trucco (Trucco & Verri, 1998). When only the epipolar constraint is used to solve the image correspondence problem, a minimum of three cameras must be employed. Adding cameras beyond this minimum number significantly reduces the correspondence ambiguities in a real camera system (H.G. Maas, 1992). If the epipolar constraint for a single particle can be satisfied in four or more image planes then its spatial correspondence is identified with nearly

100% confidence (H.G. Maas, 1992; Virant & Dracos, 1997). The cameras are calibrated to determine both the intrinsic and extrinsic parameters which can be used to derive the proper transformations from 2D pixel to 3D object space. Several techniques have been attempted to reduce image correspondence ambiguity include using additional particle information such as size, color, intensity, and temporal correspondence between particle images within each image plane (Willneff & Gruen, 2002). Willneff found that the most effective of these methods is to use the temporal correspondence of particles images within each image plane to resolve unmatched or missing particles in the set of 3D reconstructed particles. This extra step resulted in an average 20% more particles for which the image correspondence and 3D reconstruction problems could be solved (Willneff & Gruen, 2002).

2.2.3 **Temporal tracking**

The temporal tracking algorithm establishes the temporal correspondence of particles from one time step to the next through object space in order to reconstruct trajectories. Many variations of the temporal tracking algorithm exist, but at the most general level, they share a common structure. Nearly all take the input of 3D particle coordinates grouped by time step (frame) and output the reconstructed trajectories. The most common include the two-frame and multi-frame algorithms, but several others have shown promise including the neural network approach (Pereira et al., 2006) and the Extended Kalman Filter approach (Straw et al., 2010).

The multi-frame algorithm has been shown to be superior for reconstructing long trajectories in the presence of image noise (Kitzhofer & Bruecker, 2010; Ouellette et al., 2006; Shindler et al., 2010). In a multi-frame temporal tracking algorithm the next position of each particle is predicted based on its location in up to five previous time steps using a kinematic model of velocity and acceleration (equation 2-3). This use of the trajectory's history allows particles to be tracked over extended periods of time in the presence of position errors introduce through image noise. In general, the multi-frame temporal tracking algorithm has the following procedure:

1) For an initial frame n-1, initiate two point trajectories by adding the nearest neighboring particle in frame f to each particle in frame n-1.

2) Extrapolate the trajectories to estimate the particle's position in the following frame n+1 ($\hat{\mathbf{x}}_i^{n+1}$) with an approximation of velocity (u_i^n) and acceleration (a_i^n).

$$\hat{\mathbf{x}}_i^{n+1} = \mathbf{x}_i^n + u_i^n \Delta t + a_i^n \Delta t^2 \qquad (2\text{-}3)$$

3) Evaluate the quality of each candidate particle for addition to each trajectory using a cost function.

4) Move to the next frame and complete steps 2-4 until all frames have been processed.

Common trajectory extrapolation methods include; 1st order finite difference approximation of velocity, 1st order velocity approximation with 2nd order acceleration approximation (Malik et al., 1993; Shindler et al., 2010), and 2nd order polynomial regression (Li, Zhang, Sun, & Yan, 2008). Multiple cost functions have been proposed for selecting particles for addition to a trajectory including; nearest neighbor (Malik et al., 1993), minimum acceleration (Malik et al., 1993), minimum change in acceleration (Malik et al., 1993), four frame best estimate (Ouellette et al., 2006), and ratio of regression residual to geometric mean displacement (Biwole et al., 2009; Li et al., 2008).

2.3 **Real-time approaches**

A real-time system is defined by an explicit constraint on the system's response time (Dougherty & Laplante, 1995). For a real-time particle tracking system this response time can be based on two different scenarios, 1) preventing data accumulation in the camera or computer which limits experiment run time (Chan, Stich, & Voth, 2007; Hoyer et al., 2005) and 2) active monitoring of dynamic objects for feedback in an online control system (Straw et al., 2010). The former has been the focus of most researchers seeking real-time particle tracking because it stands as a current limitation for measurement durations and inhibits a thorough statistical analysis of chaotic phenomena. The latter represents a more strict time constraint and would enable entirely new applications of particle tracking including online industrial control systems. Straw and coworkers (2010) have demonstrated the online control concept by developing a real-time tracking system for behavioral investigations of flying insects (Straw et al., 2010). In their work they developed a tracking system that could trace the trajectory of several flies and trigger a

secondary high speed camera based on a fly's location in real-time. Straw concluded that to track hundreds or thousands of objects (flies) in real-time, parallel processing would be required (Straw et al., 2010).

To achieve a real-time particle tracking system which meets both real-time scenarios for thousands of particles, all aspects of the particle tracking method need to be accelerated. This effort can be characterized by two fundamental approaches: (1) reducing the net data transfer from the cameras (M. Kreizer & Liberzon, 2010) and (2) increasing data processing throughput in the computer (Meinhart, Prasad, & Adrian, 1993). These approaches will be discussed in the following two subsections.

2.3.1 **Data accumulation limitation**

As the resolution of particle tracking systems has increased with advancing camera technology, the ability to efficiently to transfer, process and store the enormous amount of image data has persisted as a limiting factor to achieving higher spatial-temporal resolution, longer observations, and real-time processing (Adrian, 1991; M Kreizer, Ratner, & Liberzon, 2009). The main bottleneck occurs in data transfer between camera and computer. With enormous data generation rates, the cameras must store images in buffer memory. For example, a PTV system with four 1 mega-pixel 8-bit monochrome cameras recording for 60 seconds at 500 frames per second will generate 120 GB of image data. Hoyer (2005) observed that the measurement duration with their 500Hz cameras was limited to only four seconds due to camera memory, which led to convergence issues in their statistical analysis (Hoyer et al., 2005).

This limit on recording time can be derived from the net data accumulation rate (B/s) and memory capacity of the camera. The net data accumulation rate is equal to the data generation rate minus the data transfer rate. Using this analysis and assuming an 8-bit monochrome camera connected to a host computer with a 1 Gigabit IEE-802.3 standard cable, Figure 2 shows the theoretical maximum recording time per gigabyte of camera memory with respect to frame rate over several common sensor resolutions.

2.3.2 **Relevant work in real-time image processing**

Recently there has been significant work with new camera technologies that can eliminate the memory transfer bottleneck for most applications. Chan (2007) developed a data compression system that reduced data transfer between the camera and computer by up to 1000 times, increasing the continuous recording time from 6.5 seconds up to a week (Chan et al., 2007). Kreizer (2010) has shown significant progress towards real-time 3D-PTV by addressing the data accumulation limitation through the use of smart cameras with embedded Field Programmable Gate Arrays (FPGA). The FPGA processes each pixel in parallel offering a fast method to filter noise, remove background and locate particle centroids in real-time prior to transferring data to the host computer (M. Kreizer & Liberzon, 2010). Therefore, instead of transferring an image, these smart cameras only output the pixel coordinates of the particles' centroids, thus the overall data transfer was reduced by factor of up to 1000 (M. Kreizer & Liberzon, 2010). The impact of using FPGA based cameras on experimental run time is shown in Figure 2; assuming a 1280x1024 pixel CMOS camera, 1024 particles per frame, and 20 Bytes per particle. The FPGA smart camera is theoretically able to record indefinitely at up to about 2000 Hz, while a traditional camera of similar resolution at that frame-rate would be limited to only a few seconds per GB of buffer memory.

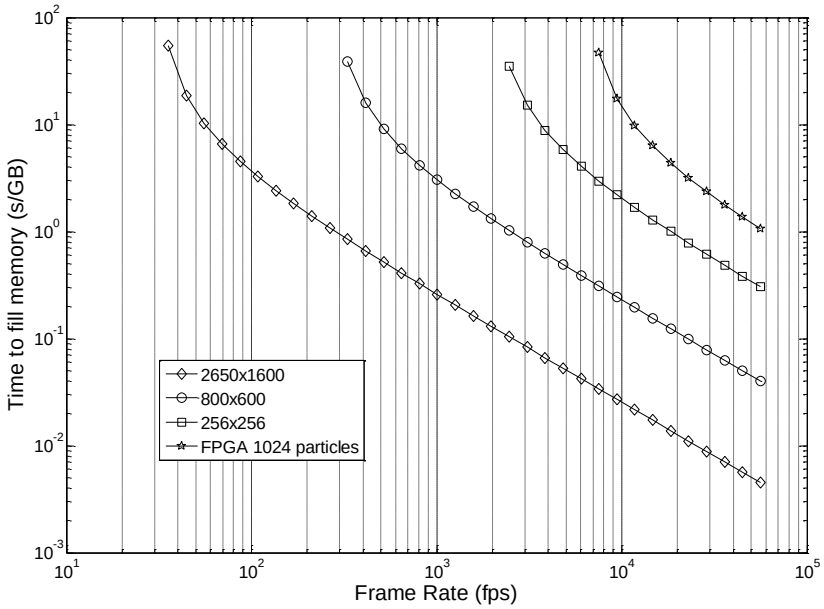

Figure 2: Camera recording time per GB of memory consumed as a function of frame rate and sensor resolution

2.3.3 **Relevant work in parallel processing**

Many researchers have predicted that parallel processing will speed up particle tracking run-times and is necessary to achieve real-time measurements (Pereira et al., 2006; Straw et al., 2010). By parallelizing the particle tracking algorithm, processing time can be greatly reduced, allowing for higher fidelity experiments: more particles, higher frame rates, and longer measurements. In addition, memory limitations can be overcome by distributing data across multiple processors. However, very few parallel particle tracking algorithms have been published and data processing in the computer has become a bottleneck for real-time processing. Computationally intensive algorithms such as holographic PTV, which may run for 10 hours for only a few seconds of image data, have led to a few specialized parallel implementations (Satake et al., 2008; Satake et al., 2007).

Parallel processing has been used since the early days of 3D particle tracking and related PIV research as a means of expediting processing times. Most have focused on parallelizing the expensive Fast Fourier Transform (FFT) used in PIV cross-correlation and holographic PTV. Meinhart (1993) was one of the first in PIV research to use parallel processing to gain speedup. At that time, processing a thousand PIV generated vectors required about three hours. They parallelized the FFT operation and achieved a speedup of ten-fold (Meinhart et al., 1993). This approach was platform specific since multi-platform standard parallel programming paradigms had not yet been developed. Satake (2007) developed a parallel algorithm for holographic PTV based on the Message Passing Interface (MPI) parallel programming library and achieved a 100x speedup for the FFT operation (Satake et al., 2007). Satake continued this work and developed a Windows-based grid system and evaluated both spatial and temporal data decomposition methods for parallelization of the FFT. They concluded that temporal decomposition of the image video provided an efficient method of parallelization and suggested that this approach could be useful for both PIV and PTV in the future (Satake et al., 2008). However, there is still no general parallel particle tracking algorithm that can efficiently utilize multiple processors, and current algorithms are not scalable and unable to complete real-time tracking at higher frame rates (> 500 fps).

2.4 **Parallel processing**

Performance of computer architectures has been steadily increasing over the last 30 years, roughly doubling in processor speed every 18 months. However, in the last decade processing speeds of CPUs have begun to level off and manufacturers have shifted their focus toward increasing the number of processors per chip. At the same time, the computer gaming market has fueled development of graphics processor units (GPUs) capable of making massively parallel computations as needed for pixel rendering in 3D games. Recently GPUs have transitioned from the consumer market into the computational science arena as accelerators for data-parallel operations.

GPU accelerators have become widely accepted for inexpensive general purpose computing due to their ability to achieve over 100 times speed-up for common data-parallel tasks versus commodity CPUs (Wu, 2008). This style architecture is now under heavy development in industry and can achieve nearly 100 fold increase in processing throughput and bandwidth compared with CPUs. The reason for this is because the GPU devotes more transistors to data processing than the CPU (Wu, 2008).

Both CPU and GPU can conduct parallel processing, however the relative size of the parallel work unit or thread is different. Since the CPU devotes more transistors to control and data management, they are ideal for processing larger units of work or heavy threads such as simultaneously running two programs. The GPU on the other hand is ideal for very lightweight threads, such as simultaneously increasing pixel intensity for an entire digital image. Therefore if a big data intensive problem can be decomposed into large threads, each of which can be further decomposed into finer threads, then an entire cluster of CPUs and GPUs could be used in unison to achieve significant speedups over sequential processing.

Efficient parallel programming remains a paramount challenge due to added complexity required for synchronous and asynchronous thread execution and management. Several programming models such as MPI, Charm++ and NVIDIA's Compute Unified Device Architecture (CUDA) have been designed to help programmers fully utilize available hardware. The Charm++ framework allows algorithm performance to scale efficiently from single dual-core desktops to supercomputing clusters without code modification (Kale &

Krishnan, 1993), and CUDA allows algorithms to execute on NVIDIA GPU's inherently parallel architecture (NVIDIA, 2012).

2.5 **Summary and conceptual design**

The smart camera with embedded FPGA makes it possible to build Lagrangian particle tracking systems that can record indefinitely while achieving both a higher frame rate and higher spatial resolution than was previously possible. For a future real-time LPT system, it makes sense to utilize these "smart" cameras to conduct the image acquisition, processing and particle detection. The higher frame rate achievable will make temporal tracking easier, which increases the number of particles to process through the remaining steps. In this case, the processing bottleneck is eliminated in the camera but the remaining steps including; solving the correspondence problem, 3D reconstruction, and temporal tracking become the bottlenecks.

To handle the increased amount of data due to increased resolution, a scalable method is required to spread the data across a heterogeneous (CPU/GPU) computer cluster. The cameras would send data over the internet, which provide significant flexibility in installation, allowing the cameras to be physically separated from the computer system. The conceptual design used as the basis for the proposed work is as follows: A network of FPGA smart cameras will be used to conduct image processing and particle detection, while a heterogeneous cluster processes the multi-camera correspondence, 3D reconstruction, temporal tracking and result visualization (Figure 3).

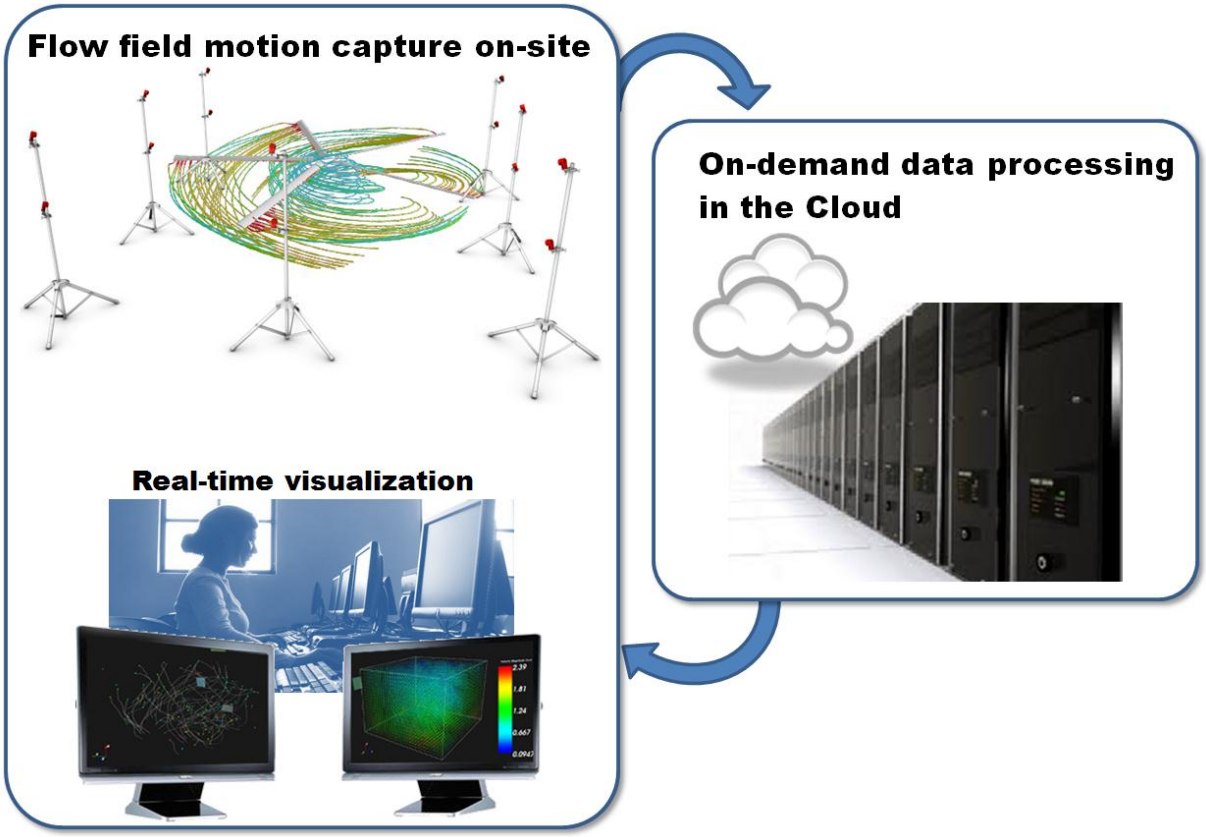

Figure 3: Real-time Lagrangian Particle Tracking system concept

In an experiment, the cameras send the identified particle pixel coordinates or segmented image in real-time to a cluster of computers located in the cloud. The cluster would be comprised of an array of nodes each with multi-core CPU processors and GPU accelerators. The cluster will process frames of particles in parallel across the array of nodes in a streaming pipeline as they are received. Each node will receive a small set of frames, then divide the particles within each frame and process them in parallel on the GPU to determine the multi-camera correspondence of particle images, followed by 3D reconstruction, and tracking. Each node will communicate with a local group to merge trajectory segments to form longer trajectories. The resulting trajectories will be processed to determine Lagrangian and Eularian properties of the particle flow field and send the results back to the researcher for real-time visualization.

3 Lagrangian particle tracking algorithms

3.1 Overview and objectives

The objective of this chapter is to identify the algorithms that will be used to develop the real-time Lagrangain Particle Tracking (LPT) system. The algorithms are broken down and will be described in five sections; 1) image processing and object detection, 2) multi-camera correspondence, 3) 3D reconstruction, 4) temporal tracking and 5) result processing and visualization. For each algorithm the relevant literature and fundamental mathematics will be provided and then the selected algorithm will be discussed.

The implementation approach for the real-time LPT system is to leverage high quality open source libraries for proven algorithms to maintain a high quality scalable code. The algorithms discussed were all implemented in C++, and some of the methods were selected from open source C++ libraries listed in Table 1.

Table 1: Open source libraries used to implement the LPT algorithms

Open source C++ library	Usage
Open source Computer Vision (OpenCV)	Camera calibration, image processing, object detection
Visualization Toolkit (VTK)	Foundation building blocks for the interactive rendering environment
Boost C++ Libraries	Statistical accumulator framework

3.2 Image processing and particle detection

The image processing and particle detection algorithm is the first step in the LPT process. The selection of image processing methods and detection schemes is dependent on the type of illumination and seed particles used. For this project, helium filled soap bubbles were selected as the seed particle for LPT experiments in large volume flow fields. Therefore, the image processing and particle detection algorithm was based on the work by Biwole (Biwole et al., 2009). Biwole and coworkers (2009) found that the 1D Gaussian estimator used by Ouellette (2006) did not fare well when used with neutrally buoyant helium filled soap bubbles as tracer

particles because the bubbles' transparency caused non uniform images. Biwole concluded that for bubbles the following method works best:

1. Image Segmentation: Apply a uniform threshold operation to create a segmented binary image, where pixels with higher intensity than the threshold value are assigned a value of one and the remaining background pixels are set to zero. The threshold is selected empirically and the optimal threshold depends on illumination intensity, particle size, exposure time, and sensitivity of the sensor.
2. Structural image transformations: iteratively dilate and erode the image to fill in the non-uniformities of the segmented particle images.
3. Calculate the centroid of each processed particle image through the weighted-average technique, where the distorted particle centroid pixel locations are given by equation 3-1 and the pixel intensity at pixel position (x,y) is represented by $I(x,y)$.

$$x_{pd} = \frac{\sum xI(x,y)}{\sum I(x,y)}$$
$$y_{pd} = \frac{\sum yI(x,y)}{\sum I(x,y)} \qquad (3\text{-}1)$$

Image processing and particle detection was implemented with the open source computer vision library (OpenCV)(Itseez, 2012), which provides a number of functions for image processing, object detection and camera calibration. The final algorithm was as follows.

1. Apply a three point kernel Gaussian filter to the image to effectively blur the image and reduce the impact of pixel noise.
2. Threshold the image based on a user defined value from 0-255. All pixels below the threshold are assigned 0 and all above are assigned 1. The result is each particle "blob" image becoming white against a dark background.
3. Apply the *findContours* function of OpenCV which identifies the perimeter contour of each segmented "blob" in the binary image. The x and y pixel coordinates of the perimeter pixels are stored in an array for each contour found.
4. The spatial moments of each contour found are calculated and the distorted pixel coordinates of the particle image centroid is then determined based on equation 3-2.

$$m_{ij} = \sum_{x,y} I(x,y) x^j y^i$$

$$x_{pd} = \bar{x} = \frac{m_{10}}{m_{00}} \qquad (3\text{-}2)$$

$$y_{pd} = \bar{y} = \frac{m_{01}}{m_{00}}$$

Lens distortion can be significant in real lenses and needs to be removed numerically to reduce uncertainty in the reconstructed 3D particle positions. The goal is to create a transformation that will correct the observed particle image centroid location (as determined by equation 3-2) for lens distortion based on a distortion model and calibration coefficients. In this study, distortion is approximated through the nonlinear model given in equation 3-3 and 3-4 (Itseez, 2012). This model includes up to six radial distortion coefficients (k_1 to k_6) and two tangential distortion coefficients (p_1 and p_2). These eight coefficients together with the focal length (f_x, f_y) and principle point (c_x, c_y) are the camera's intrinsic parameters.

$$x'' = x' \frac{1 + k_1 r^2 + k_2 r^4 + k_3 r^6}{1 + k_4 r^2 + k_5 r^4 + k_6 r^6} + 2 p_1 x' y' + p_2 \left(r^2 + 2 x'^2 \right)$$
$$y'' = y' \frac{1 + k_1 r^2 + k_2 r^4 + k_3 r^6}{1 + k_4 r^2 + k_5 r^4 + k_6 r^6} + p_1 \left(r^2 + 2 y'^2 \right) + 2 p_2 x' y' \qquad (3\text{-}3)$$

$$r^2 = x'^2 + y'^2 \qquad (3\text{-}4)$$

In equations above and below, (x'' , y'') and (x' , y') represent the distorted and undistorted homogeneous coordinates of an image respectively. The homogeneous camera coordinates are non-dimensional and simply the in plane camera coordinates x_c and y_c normalized by the out-of-plane coordinate z_c. The undistorted homogenous coordinates (x' , y') are determined by iteratively solving the nonlinear equations in 3-3, where the distorted homogenous camera coordinates are found from the intrinsic parameters and the observed distorted pixel coordinates of an image point x_{pd} and y_{pd} as shown in equation 3-5.

$$x'' = \frac{x_{pd} - c_x}{f_x}$$
$$y'' = \frac{y_{pd} - c_y}{f_y} \qquad (3\text{-}5)$$

Once the undistorted homogeneous coordinates of a particle image are known, the undistorted image pixel coordinates can be found through equation 3-6 given the focal length and principle point of the lens.

$$x_p = f_x x' + c_x$$
$$y_p = f_y y' + c_y \qquad (3\text{-}6)$$

3.3 Multi-camera correspondence

The criterion used in this research to determine spatial correspondence between particle images in multiple image planes is called the epipolar geometry constraint. The epipolar geometry between two cameras is shown in Figure 4. Point **X** represents a point in object space and $\mathbf{X}_R$ and $\mathbf{X}_L$ are the corresponding image coordinates of **X** for the right and left image plane respectively.

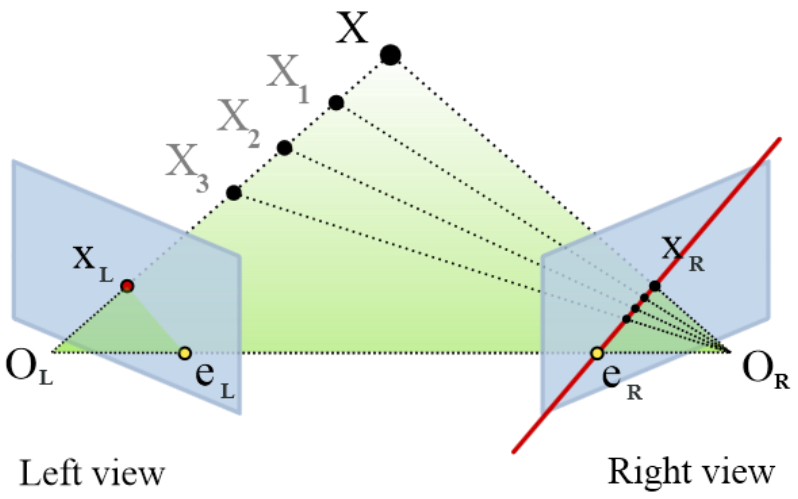

Figure 4: Epipolar geometry in stereo vision: image credit (Nordmann, 2007)

The epipolar line $\mathbf{e}_R\mathbf{X}_R$ on the right image plane is created from the ray between the origin of the left image plane $\mathbf{O}_L$ and point **X**. Correspondence ambiguities in the right camera can exist

along the epipolar line as shown in the figure. For example, the right image can clearly distinguish the four points **X**, $\mathbf{X_1}$, $\mathbf{X_2}$ and $\mathbf{X_3}$ along ray $\mathbf{O}_L\,\mathbf{X}$, while the left image can only distinguish one point represented by $\mathbf{X}_L$. In this case a third camera is required to eliminate the ambiguities and conclude that $\mathbf{X}_L$ and $\mathbf{X}_R$ correspond to **X** and not the other three points. The epipolar constraint equation for a stereo camera pair is derived next.

For two cameras, the image coordinates $\mathbf{X}_R$ and $\mathbf{X}_L$ corresponding to a single object point **X** can be related through a rotational matrix **R** and translational vector **t** as in equation 3-7 (Trucco & Verri, 1998).

$$\mathbf{X}_R = \mathbf{R}(\mathbf{X}_L - \mathbf{t}) \tag{3-7}$$

The epipolar plane as shown in the figure above is defined by the location of object point **X** and the origins of the right and left image planes ($\mathbf{O}_R$ and $\mathbf{O}_L$). This geometry creates the coplanar condition in equation 3-8 for the left and right image coordinates.

$$(\mathbf{X}_L - \mathbf{t})^T \mathbf{t} \times \mathbf{X}_R = 0 \tag{3-8}$$

Substituting the coplanar condition into equation 3-8 results in 3-9

$$(\mathbf{R}^T \mathbf{X}_R)^T \mathbf{t} \times \mathbf{X}_L = 0 \tag{3-9}$$

Trucco and Verri (1998) showed that by using a vector product rule for rank deficient matrices equation 3-9 can be refactored to the form given in equation 3-10.

$$\mathbf{t} \times \mathbf{X}_L = \mathbf{S}\mathbf{X}_L = \begin{bmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{bmatrix} \mathbf{X}_L \tag{3-10}$$

The epipolar constraint in equation 3-11 is the result of this derivation and **E** is the essential matrix which is defined as **E**=**RS** where **R** is the rotation matrix. Using this constraint, two points in camera coordinates can be inserted into equation 3-11 and if the result is zero then the points satisfy the epipolar constraint, if the result is greater than zero then no match is made.

$$\mathbf{X}_R{}^T \mathbf{E} \mathbf{X}_L = 0 \tag{3-11}$$

$$\mathbf{E} = \mathbf{RS} = \mathbf{R}\begin{bmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{bmatrix} \tag{3-12}$$

The epipolar constraint can be rewritten in terms of the image pixel coordinates using the camera intrinsic parameters, such the matrix **M** is the camera matrix is:

$$\mathbf{M} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3-13}$$

Then the image pixel coordinates $\mathbf{x}_L$ relate to the point's camera coordinates as 3-14.

$$\mathbf{X}_L = \mathbf{M}^{-1}\mathbf{x}_L \tag{3-14}$$

The fundament matrix **F** then relates to the essential matrix through the camera matrices of the stereo pair as follows.

$$\mathbf{F} = \mathbf{M}_R^{-\mathrm{T}}\mathbf{E}\mathbf{M}_L^{-1} \tag{3-15}$$

The epipolar constraint can then be written in terms of the pixel coordinates of the object in the right and left image planes ($\mathbf{x}_R$ and $\mathbf{x}_L$) as shown in 3-16.

$$\mathbf{x}_R{}^{\mathrm{T}}\mathbf{F}\mathbf{x}_L = 0 \tag{3-16}$$

Since the epipolar constraint leaves ambiguities for stereo cameras a third camera is needed. The following four step process describes how matches are made in a three camera system: 1) For a particle imaged in camera one an epipolar line is projected in camera two, 2) for all particles in camera two falling within a set distance from the epipolar line of camera one, respective epipolar lines are projected from cameras one and two into camera three, 3) Any particle within the intersecting epipolar lines from camera one and two in camera three is a correct match. However, as described by Maas (1992), three cameras may still not be able to eliminate all correspondence ambiguities completely.

Maas (1992) concluded that, for higher seed particle densities, the use of four cameras is best to drive down the ambiguities by a factor of 100 and nearly eliminates them all together (H.G. Maas, 1992). The challenge with the four camera epipolar matching algorithm is that it is computationally expensive. For each particle found by camera one an epipolar line has to be projected into the other three cameras to search for particle images that satisfy the epipolar constraint.

For the real-time LPT system, the correspondence algorithm needed to be scalable to more than four cameras. Therefore this algorithm was implemented to scan over an arbitrary number of cameras searching for particles that satisfy the four camera matching criteria described by Maas (1992). This multi-camera correspondence algorithm was separated into two stages where the first stage identifies particles image pairs from each unique two-camera combination that satisfies the epipolar constraint within a tolerance. For a six camera system, this leads to 15 unique cameras combinations from which to search for matching particle image pairs. If the result of evaluating equation 3-16 is less than a set threshold then the corresponding particle image pair will be stored and passed on to stage two. This first stage is shown in pseudo code below:

Multi-camera Correspondence Algorithm: Stage 1 – Find all epipolar matches in stereo pairs

```
for all camera two combinations (A, B)
    for all camera A particles x_i
        for all camera B particles x_j
            Solve d = x_i F x_j (Equation 3-16)
            if (d < threshold)
                Add x_i to camera B's match list
                Add x_j to camera A's match list
```

In the second stage all unique four-camera combinations will be evaluated to determine if the matched two-camera particle image pairs satisfy the epipolar constraint for all four cameras. This involves searching all possible combinations of cameras within unique four-camera groups, labeled A, B, C, and D. For a 6 camera LPT system this results in 15 unique four-camera groups. This search becomes a combinatorial nested loop as shown in pseudo code below.

Multi-camera Correspondence Algorithm: Stage 2–Evaluate four-camera epipolar constraint

```
n = total number of cameras
for a = 0 to n

    for b = a to n

        for c = b to n - 1

            for d = c to n - 1

                Search each camera's match list for corresponding
                particles (x_a , x_b , x_c , x_d)

                If four camera correspondence is found
                    Add x_a, x_b, x_c, x_d to list for 3D reconstruction
```

The reason this algorithm was split into two parts was to separate the computation (evaluation of the epipolar constraint equation) from the judgment (comparing each two-camera particle image pair list to find unique four-camera correspondence). This exposes parallelism in stage 1 where computational cost is high. Chapter 4 describes how this parallelism is utilized to create a real time algorithm for an arbitrary number of cameras. Once all four-camera correspondences are found they are sent to the 3D reconstruction algorithm.

3.4 **3D reconstruction**

The 3D reconstruction algorithm is responsible for taking the four-camera matched particle images centroid locations (in pixel coordinates) from the multi-camera correspondence algorithm and computing a single 3D position in object world coordinates. This is accomplished through the classic camera projection model, which relates camera and world coordinate systems through a rotation **R** and translation t, which are known as the extrinsic camera parameters (Figure 5). Given a pinhole camera model as described by Trucco and Verri (1998) equation 3-10 describes the a projection from a point with world coordinates $\mathbf{X}_w$ to an camera coordinates $\mathbf{X}_c$ assuming the 3 x 3 rotation matrix **R** and 3x1 translation vector **t** are known.

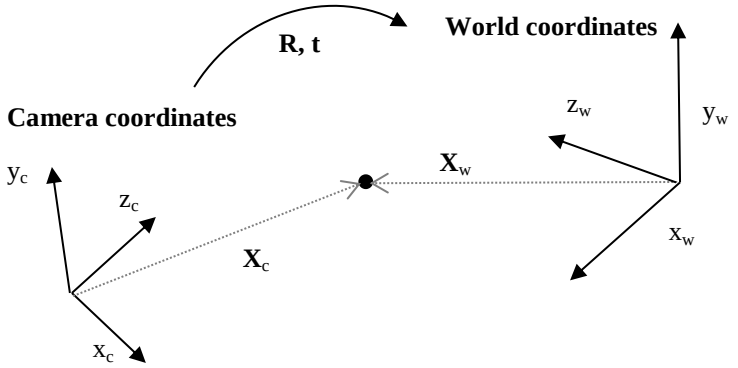

Figure 5: Transformation from world to camera coordinates

$$\mathbf{X}_c = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \mathbf{R}\mathbf{X}_w + \mathbf{t} = \mathbf{R}\begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} + \mathbf{t} \tag{3-17}$$

The goal is to utilize equation 3-17 to create a transformation from camera coordinates, given as pixel locations, to world coordinates. Equation 3-17 can be written in element notation as follows.

$$\begin{aligned} x_c &= r_{11}x_w + r_{12}x_w + r_{13}x_w + T_x \\ y_c &= r_{21}x_w + r_{22}y_w + r_{23}z_w + T_y \\ z_c &= r_{31}x_w + r_{32}y_w + r_{33}z_w + T_z \end{aligned} \tag{3-18}$$

The camera coordinates can be normalized to for the homogeneous coordinates (x' , y').

$$\begin{aligned} x' &= \frac{x_c}{z_c} \\ y' &= \frac{y_c}{z_c} \end{aligned} \tag{3-19}$$

Then substituting equation 3-18 into 3-19 provides a simple linear function relating the homogenous camera coordinates to the world coordinates through the external parameters of the camera.

$$x' = \frac{r_{11}x_w + r_{12}y_w + r_{13}z_w + T_x}{r_{31}x_w + r_{32}y_w + r_{33}z_w + T_z} \qquad (3\text{-}20)$$
$$y' = \frac{r_{21}x_w + r_{22}y_w + r_{23}z_w + T_y}{r_{31}x_w + r_{32}y_w + r_{33}z_w + T_z}$$

The intrinsic camera parameters, focal length (f_x, f_y) and image center (c_x, c_y) in units of pixels, along with the undistorted pixel coordinates (x_p, y_p) can be used to calculate the homogenous coordinates for substitution into 3-21.

$$x' = \frac{x_p - c_x}{f_x} \qquad (3\text{-}21)$$
$$y' = \frac{y_p - c_y}{f_y}$$

The direct transformation from 2D undistorted image pixel coordinates to 3D world coordinates is complete. This transformation can be given in two equations for each camera, which can be rearranged into the form $\mathbf{AX}_w = \mathbf{b}$ as shown in equations 3-22 to 3-24.

$$(x'r_{31} - r_{11})x_w + (x'r_{32} - r_{12})y_w + (x'r_{33} - r_{13})z_w = T_x - x'T_z \qquad (3\text{-}22)$$
$$(y'r_{31} - r_{21})x_w + (y'r_{32} - r_{22})y_w + (y'r_{33} - r_{23})z_w = T_y - y'T_z$$

Where $\mathbf{A}$ and $\mathbf{b}$ are comprised of sub matrices $\mathbf{A}_i$ and $\mathbf{b}_i$ for each camera *i* to N:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_i \\ \vdots \\ \mathbf{A}_N \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_i \\ \vdots \\ \mathbf{b}_N \end{bmatrix} \qquad (3\text{-}23)$$

$$\mathbf{A}_i = \begin{bmatrix} x_{n,i}r_{31} - r_{11} & x_{n,i}r_{32} - r_{12} & x_{n,i}r_{33} - r_{13} \\ y_{n,i}r_{31} - r_{21} & y_{n,i}r_{32} - r_{22} & y_{n,i}r_{33} - r_{23} \end{bmatrix}; \quad \mathbf{b}_i = \begin{bmatrix} T_x - x_{n,i}T_z \\ T_y - y_{n,i}T_z \end{bmatrix} \qquad (3\text{-}24)$$

This linear system of equations can be solved for 3D coordinates $\mathbf{X}_w$ using a linear least squares solver including QR factorization, such as a Householder rotation and backward substitution, or Singular Value Decomposition (SVD) (Heath, 2002). The resulting 3D reconstruction algorithm is shown in pseudo code below.

3D Reconstruction Algorithm

for camera i = 1 to N

k = i * 2;

$x' = (x_{p,i} - c_{x,i}) / f_{x,i}$

$y' = (y_{p,i} - c_{y,i}) / f_{y,i}$

$A(k-1, 1) = x' * R_i(3,1) - R_i(1,1)$

$A(k-1, 2) = x' * R_i(3,2) - R_i(1,2)$

$A(k-1, 3) = x' * R_i(3,3) - R_i(1,3)$

$A(k, 1) = y' * R_i(3,1) - R_i(2,1)$

$A(k, 2) = y' * R_i(3,2) - R_i(2,2)$

$A(k, 3) = y' * R_i(3,3) - R_i(2,3)$

$B(k-1) = T_i(1) - x' * T_i(3)$

$B(k) = T_i(2) - y' * T_i(3)$

end

Solve $\mathbf{AX_w=B}$ for world coordinate $\mathbf{X_w}$ using QR factorization or SVD

While 3D reconstruction can be completed with a minimum of two cameras, it is hypothesized that the utilization of all four cameras from each four-camera correspondence group will improve the conditioning and sensitivity of the least squares problem. This hypothesis is tested and discussed in Chapter 5.

3.5 Temporal tracking

The temporal tracking algorithm establishes temporal correspondence between reconstructed particles from consecutive image frames to form trajectories through space and time. Two temporal tracking algorithms were ultimately implemented in this research. 1) a multi-frame regression based tracking algorithm developed by Li (Li et al., 2008) and 2) a new algorithm, called particle priority strict matching, based on finite difference approximation of particle velocity and strict two judgment approach with computationally inexpensive cost functions.

3.5.1 Multi-frame regression based tracking algorithm

A multi-frame algorithm base on linear regression was implemented due to its proven robustness against input noise and cross-gap (missing frame) tracking capability. This multi-frame algorithm is described in detail by Li for 2D (Li et al., 2008), and was extended in this research to 3D as described below.

1) **Initialize trajectories:** for all particles in the current frame *n*, draw a search sphere around the i^{th} particle's location $\mathbf{x}_i^n$ and identify all particles at the next frame, *n*+1 within the sphere. Create a two point trajectory for each of these candidate particles and the original particle in frame *n*.

2) **Extrapolate trajectory:** proceed to next frame and fit existing trajectories with polynomials through linear least squares regression. The order of the polynomial used is a function of the number of particles in the trajectory; two point trajectories are fit exactly with first order polynomials while longer trajectories use a second order approximation as shown in equation 3-25. The i^{th} particle's extrapolated position in the next frame $\hat{\mathbf{x}}_i^{n+1}$ can then be determined by equation 3-25. **A**, **B** and **C** are 3 by 1 vectors determined by regression where t_{n+1} is the time at the *n*+1 frame.

$$\hat{\mathbf{x}}_i^{n+1} = \mathbf{A}t_{n+1}^2 + \mathbf{B}t_{n+1} + \mathbf{C} \qquad (3\text{-}25)$$

3) **Establish search region:** create a search sphere with radius *r* around $\hat{\mathbf{x}}_i^{n+1}$ based on the estimated velocity from the previously connected points in the trajectory as shown in equation 3-26, where *t* is time corresponding to the frame index subscript and α is a user defined constant. Then identify all particles that fall inside this sphere as candidates for the trajectory.

$$r = \alpha \frac{t_{n+1} - t_n}{t_n - t_{n-1}} \left| \mathbf{x}_i^n - \mathbf{x}_i^{n-1} \right| \qquad (3\text{-}26)$$

4) **Evaluate cost function:** for each candidate, a cost function is evaluated based on the smoothness of the trajectory. A candidate particle *j* with a cost value Φ_j below a given threshold is added to the trajectory. If more than one candidate particle falls within the cost threshold then the trajectory is copied and a new trajectory branch is tracked in future frames. The cost function used in this study is based on the work of Li (2008) equation 3-27.

$$\phi_j = \frac{\sqrt{\sum_{k=0}^{3} \left| \mathbf{D}_k - \mathbf{G}\tau_k - \mathbf{H} \right|^2}}{\sqrt{\sum_{k=0}^{3} \left| \mathbf{D}_k \right|^2}} \tag{3-27}$$

$$\hat{\mathbf{D}}_k = \mathbf{G}\tau_k + \mathbf{H} \tag{3-28}$$

τ_k is the mid time between frame *k* and *k*+1. $\mathbf{D}_k$ is the particle displacement between frames *k* and *k*+1. **G** and **H** are 3 by 1 matrices determined from the regression process of equation 3-28.

5) **Remove short trajectories:** check that each trajectory is active by identifying how many frames exist where no candidate particles were found. If a trajectory has not found a candidate particle for N_{gap} consecutive frames and has fewer than N_{True} frames then the trajectory is not tracked any further.

6) **Proceed:** track the particles with existing trajectories through each frame completing steps (1) through (5) until all frames have been processed.

One of the major drawbacks of the regression based method is that its computational cost is significant. Also since it allows multiple trajectories to match with the same particle in the new frame, tracking ambiguities can exist. Dealing with these ambiguities adds additional computational overhead and makes the regression based method not ideal for real-time application. Therefore, a new temporal tracking algorithm was developed and will be discussed next.

3.5.2 **Particle priority strict matching algorithm**

To achieve real-time processing a new temporal tracking algorithm was developed to minimize the computational cost and reduce tracking ambiguities. This algorithm prevents tracking ambiguities by ensuring that candidate particles in the next frame (*n*+1) can match with only one trajectory at the current frame (*n*) based on minimizing a cost function. This is a strict matching algorithm because the matching cost is efficiently judged in two steps. First, candidate particles are assigned the trajectory that minimizes the cost function among all existing trajectories. In this

step, a single trajectory may be an optimal match with several candidate particles. Therefore a second judgment step is used to break ambiguities. In the second step, trajectories sort their list of optimal candidate particles and select the one with the lowest cost.

The cost function is only evaluated once. The cost function can be changed based on the type of experiment being run, and can include any of the cost functions described by Maas (H. G. Maas et al., 1993) and Ouellette (Ouellette et al., 2006). In this research, the Euclidian distance between a candidate at frame n+1 with the predicted trajectory position at frame n+1 was used as the cost function. This approach is fast and provides good tracking for cases where particle displacement is small relative to particle spacing in each frame.

This new algorithm is implemented as follows:

1) **Initialize trajectories:** For all unmatched particles in frame n-1, initiate two point trajectories ($\mathbf{x}^{n-1}$, $\mathbf{x}^{n}$) by adding the nearest neighboring particle in frame n to each particle in frame n-1, with each particle in frame n being matched with at most one particle in frame n-1. Proceed to the next frame.

2) **Extrapolate trajectory and search region:** Extrapolate the trajectories to estimate the particle's position in the new frame n+1 with an approximation of velocity through the first order backward difference scheme. Then calculate the search radius r based on the magnitude of velocity multiplied by the time step. This search radius will define a sphere in object space centered at $\hat{\mathbf{x}}_i^{n+1}$.

$$\hat{\mathbf{x}}_i^{n+1} = \mathbf{x}_i^n + \mathbf{u}_i^n \Delta t \qquad (3\text{-}29)$$

$$\mathbf{u}_i^n = \frac{\mathbf{x}_i^n - \mathbf{x}_i^{n-1}}{\Delta t} + O(\Delta t) \qquad (3\text{-}30)$$

$$r = \|\mathbf{u}\|_2 \Delta t \qquad (3\text{-}31)$$

3) **Evaluate cost function:** For each candidate particle $\mathbf{x}_c^{n+1}$ in frame n+1, loop through all trajectories calculating the distance d from the candidate particle $\mathbf{x}_c^{n+1}$ to the extrapolated point of the trajectory $\hat{\mathbf{x}}_i^{n+1}$ (equation 3-32). If d is less than r then a cost function is

evaluated. If the cost of adding the current candidate particle $\mathbf{x}_c^{n+1}$ to the trajectory is less than any prior trajectory then this optimal candidate particle/trajectory match is temporarily stored. Once the candidate particle has evaluated the cost associated with all trajectories, it stored as a match in the optimal trajectory's list of optimal matches.

$$d = \left\| \mathbf{x}_c^{n+1} - \hat{\mathbf{x}}_i^{n+1} \right\|_2 \tag{3-32}$$

4) **Sort optimal candidates and select best match:** Loop through each trajectory to sort its optimal candidate list and select the lowest cost candidate particle to extend the trajectory by one frame.

5) **Proceed:** track the particles with existing trajectories through each frame completing steps (1) through (4) until all frames have been processed. If a trajectory fails to find an optimal candidate particle in the following frame, then remove the trajectory from the active trajectory list. All unmatched candidate particles in a frame will be used to initialize new trajectories in the following frame.

3.6 Data analysis and visualization

One goal of this research is to achieve real-time processing and visualization of results as the experiment is in progress. Therefore this section will describe the data analysis that can be completed on the Lagrangian particle trajectories and how these results can be visualized and interacted with in real-time. The Visualization Toolkit (VTK) was used to develop an interactive visualization module for the real-time LPT system. This module provides the capability to view Lagrangian trajectories and/or a Eularian grid containing statistical values of the particle flow properties during an experiment. The derived properties that are calculated for each reference frame are given in Table 2.

Table 2: Derived properties calculated for each reference frame

Lagrangian	Eularian
Velocity	Velocity
Acceleration	Mass residual
Pressure gradient	Vorticity
	Reynolds Stress
	Turbulence Intensity
	Static Pressure

In order to calculate the Eularian properties of the particle flow field, the concept of a statistical accumulator grid (SAG) is introduced. The Eularian flow properties will be derived and calculated assuming the particles are ideal tracers, which may not hold true for inertial particles in turbulent flows. Therefore, pseudo Eularian flow properties will be calculated in the case inertial particles are being tracked. The dynamics of inertial particles in turbulence are still an open area of research. Many researchers have found that current theories of particle response to turbulent forcing based on Stokes number may not hold true for inertial particles (Bourgoin et al., 2011; Ouellette et al., 2008; Qureshi et al., 2007; Toschi & Bodenschatz, 2009). It may be possible that by observing divergence between the pseudo flow properties and true flow properties, key steps can be made toward better understanding the dynamics of inertial particles.

3.6.1 **Lagrangian reference frame**

3.6.1.1 ***Particle velocity and acceleration***

The Lagrangian velocity and acceleration of a particle along a trajectory can be calculated based on second order accurate central difference schemes using the reconstructed particle positions $\mathbf{x}^n$ for each time step n as given in equations 3-33 and 3-34. In order for these central difference schemes to accurately estimate the velocity and acceleration of the particles, the frame rate must be high enough for the trajectory to appear linear in the domain of the positions used in the finite difference stencil as depicted in Figure 6. The velocity and acceleration calculations will be sensitive to uncertainties in the particle position, which will scale with $1/\Delta t$ for velocity and $1/\Delta t^2$ for acceleration. These uncertainties will be discussed further in Chapter 5.

$$\mathbf{u} \approx \frac{\mathbf{x}^{n+1} - \mathbf{x}^{n-1}}{2\Delta t} + O(\Delta t^2) \tag{3-33}$$

$$\mathbf{a} \approx \frac{\mathbf{x}^{n+1} - 2\mathbf{x}^{n} + \mathbf{x}^{n-1}}{\Delta t^2} + O(\Delta t^2) \tag{3-34}$$

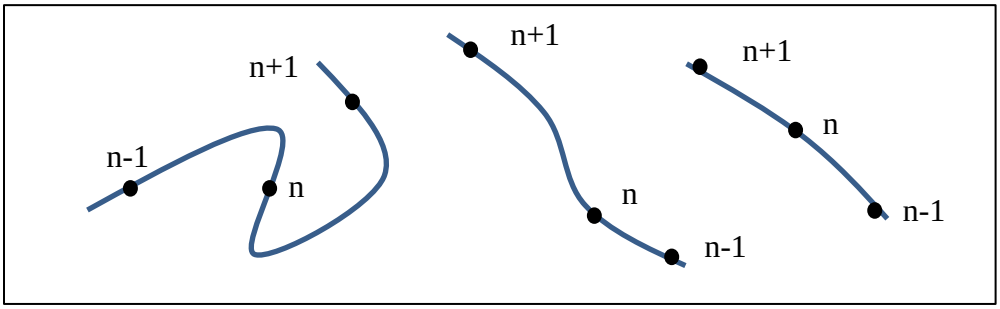

Figure 6: Trajectory sampling with central difference stencils. Left and center trajectories are under sampled and will not be able to accurately estimate velocity or acceleration

3.6.1.2 *Static pressure gradient*

In the real-time LPT system, the static pressure gradient along a trajectory is calculated in real-time as the trajectories are being constructed. If the velocity vector (**v)** and acceleration vector (**a**) are known for an ideal particle being transported in a fluid, then the instantaneous static pressure gradient of the fluid can be calculated based on the Navier-Stokes equations given below in vector notation (equation 3-35). For inertial particles this would be a pseudo pressure gradient which may deviate from the actual flow when the particles behave as non-ideal flow tracers. The resulting pressure gradients will be used below in the Eularian framework to calculate static pressures in cells of a finite volume grid through as discussed in section 3.6.2.4. This method can be referred to as the Instantaneous Lagrangian Acceleration (ILA) method:

$$\mathbf{a} = \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = \frac{1}{\rho}\left(-\nabla p + \mu \nabla^2 \mathbf{u}\right) \tag{3-35}$$

$$\nabla p = -\rho \mathbf{a} + \mu \nabla^2 \mathbf{u} \tag{3-36}$$

In equations 3-35 and 3-36 μ is the dynamic viscosity and ρ is the density of the fluid. The acceleration and diffusion terms are discretized using second order central finite differences.

The discretized acceleration term is given in 3-24 and the discretized Laplacian of velocity ($\nabla^2\mathbf{v}$) is given in equation 3-38.

$$\mathbf{a} \approx \frac{(x^{n+1} - 2x^n + x^{n-1})}{\Delta t^2}\mathbf{i} + \frac{(y^{n+1} - 2y^n + y^{n-1})}{\Delta t^2}\mathbf{j} + \frac{(z^{n+1} - 2z^n + z^{n-1})}{\Delta t^2}\mathbf{k} \tag{3-37}$$

(3-38)

$$\nabla^2\mathbf{u} \approx \left[\frac{(u^{n+1} - 2u^n + u^{n-1})}{\Delta x^2} + \frac{(u^{n+1} - 2u^n + u^{n-1})}{\Delta y^2} + \frac{(u^{n+1} - 2u^n + u^{n-1})}{\Delta z^2}\right]\mathbf{i} \left[\frac{(v^{n+1} - 2v^n + v^{n-1})}{\Delta x^2} + \frac{(v^{n+1} - 2v^n + v^{n-1})}{\Delta y^2} + \frac{(v^{n+1} - 2v^n + v^{n-1})}{\Delta z^2}\right]\mathbf{j} \left[\frac{(w^{n+1} - 2w^n + w^{n-1})}{\Delta x^2} + \frac{(w^{n+1} - 2w^n + w^{n-1})}{\Delta y^2} + \frac{(w^{n+1} - 2w^n + w^{n-1})}{\Delta z^2}\right]\mathbf{k}$$

3.6.1.3 *Trajectory visualization*

The Visualization Toolkit (VTK), an open source C++ visualization library, was used to develop an interactive rendering environment where the particles, their trajectories, and Eularian properties of the flow (as discussed in section 3.6.2) can be observed in real-time. The particles can be rendered individually as colored spheres based on the velocity or acceleration, or combined with a rendering of their trajectory history. This provides a powerful verification method, where trajectories can be visually inspected to provide instant feedback on the impact of changes to system parameters (frame-rate, image segmentation threshold, etc.). Visualization of the particle paths can help to visually identify regions of high and low velocity and acceleration without interpolating to a Eularian grid. The user can interact with the rendering environment by zooming and/or rotating the virtual 3D environment to observe the flow field from alternative perspectives.

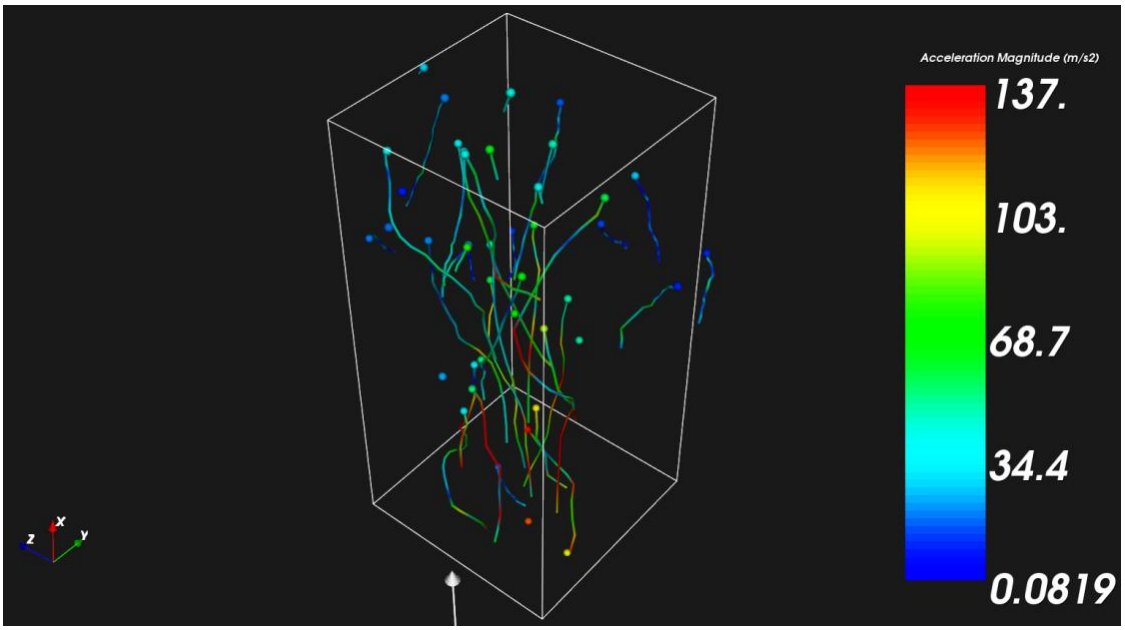

Figure 7: Trajectory visualization in the interactive rendering environment

3.6.2 Eularian reference frame

It is often useful to interpolate or otherwise attribute the Lagrangian properties to a fixed Eularian reference frame so that comparisons can be made with traditional point wise flow measurement equipment, such as hot-wire anemometers, and CFD simulations. In addition, by structuring the data on a grid, other properties of the flow field can be calculated including: continuity, vorticity, Reynolds stress, turbulent kinetic energy, and static pressure. The method for collecting, storing and representing Lagrangian properties on a Cartesian grid is discussed next.

3.6.2.1 *Statistical accumulator grid*

A mechanism of statistically storing the Lagrangian properties over time and attributing them to fixed Cartesian grid was created and will be referred to as the statistical accumulator grid (SAG). The SAG consists of a structured Cartesian virtual finite volume grid, where each cubic cell (or voxel) contains an array of statistical accumulators (Figure 8). These accumulators are objects, in the object-orientated programming sense, which accept instantaneous values of the Lagrangian properties and actively calculate statistical values for the each property including mean, variance and/or covariance. Each cubic cell in the SAG contains 12 statistical accumulators: three for velocity [*u*,*v*,*w*], three for acceleration, three for pressure gradient and

three for the covariance of velocity [Cov(u,v), Cov(u,w),Cov(v,w)]. This statistical accumulator grid is represented in the interactive rendering environment as a rectangular cube which can be sized and placed to cover a volume of interest within the observed volume of the LPT cameras.

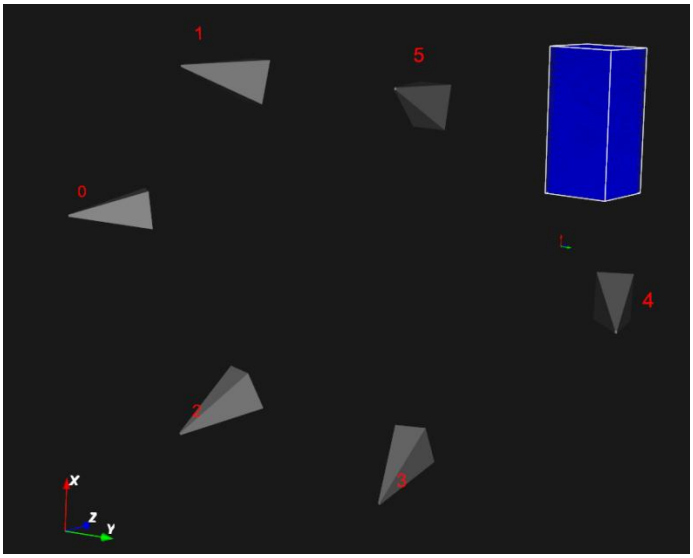

Figure 8: Visualization of six cameras observing the statistical accumulator grid (SAG) in blue

The method for adding Lagrangian particle properties to the SAG is based on weighted means and variances. Particle properties are added to all cubic cells in a defined cell neighborhood with an associated weight based on the distance from the particle to the center of each cell. A straight-forward implementation of the cell neighborhood includes only the 26 cells immediately surrounding the cubic cell which contains the particle. The weight factors (a) are calculated based on the inverse of the distance as in equation 3-39.

$$a = \left\| \mathbf{x}_p - \mathbf{x}_{cell} \right\|_2^{-1} \tag{3-39}$$

The weighted mean $\hat{\mu}$ of values accumulated in a given cell are then calculated based on the instantaneous values X and weight factors a according to equation 3-40.

$$\hat{\mu} = \frac{\sum_{i=1}^{n} a_i X_i}{\sum_{i=1}^{n} a_i} \tag{3-40}$$

The weighted variance and covariance are calculated based on iterative Monte Carlo estimators in equations 3-41 and 3-42 respectively.

$$\hat{\sigma}_n^2 = \frac{\bar{a}_n - a_n}{\bar{a}_n}\hat{\sigma}_{n-1}^2 + \frac{a_n}{\bar{a}_n - a_n}\left(X_n - \hat{\mu}_n\right)^2 \tag{3-41}$$

$$\hat{c}_n = \frac{\bar{a}_n - a_n}{\bar{a}_n}\hat{c}_{n-1} + \frac{a_n}{\bar{a}_n - a_n}\left(X_{1,n} - \hat{\mu}_{1,n}\right)\left(X_{2,n} - \hat{\mu}_{2,n}\right) \tag{3-42}$$

The mean values and variance (covariance) of velocity, acceleration and pressure gradient will serve as the foundation to calculate the distributions of mass residual, vorticity, Reynolds stress, turbulent kinetic energy, and static pressure within each cell of the statistical accumulator grid.

3.6.2.2 ***Mass residual and vorticity***

Conservation of mass within the statistical accumulator grid can be a good indicator of how well the LPT system is tracking particles within the flow field and how well the particles follow the flow field. If particles experience preferential concentration due to inertial effects, then the flow field observed through filter of particle motion can be compressible. This might be detectable if the mass residual resulting from solving the continuity equation under the assumption of incompressibility is significant.

The continuity equation simplifies equation 3-43 for incompressible flow fields.

$$\frac{\partial \bar{u}}{\partial x} + \frac{\partial \bar{v}}{\partial y} + \frac{\partial \bar{w}}{\partial z} = 0 \tag{3-43}$$

The mass residual (*MR*) in each cell (i,j,k) of size (Δx , Δy, Δz) can be calculated based on second order central finite difference approximations of the velocity gradients as shown in equation 3-44.

$$MR_{\mathrm{i,j,k}} = \left(\frac{\bar{u}_{i+1} - \bar{u}_{i-1}}{2\Delta x}\right) + \left(\frac{\bar{v}_{j+1} - \bar{v}_{j-1}}{2\Delta y}\right) + \left(\frac{\bar{w}_{k+1} - \bar{w}_{k-1}}{2\Delta z}\right) \tag{3-44}$$

Once the mean velocities $\bar{u}$, $\bar{v}$ and $\bar{w}$ are determined through hundreds or thousands of particle observations per cell, the mass residuals can be calculated in a domain marching scheme

starting from one corner of the grid and traversing each row until all rows and columns have been evaluated.

Vorticity is a useful property derived from the velocity field that can provide an alternative way of inspecting a flow field. It is a vector field corresponding to rotation in the flow and can be calculated from the spatial gradient of velocity crossed with the velocity vector (equation 3-45).

$$\begin{aligned}\omega &= \nabla \times \mathbf{v} \\ &= \left(\frac{\partial w}{\partial y} - \frac{\partial v}{\partial z}\right)\mathbf{i} + \left(\frac{\partial u}{\partial z} - \frac{\partial w}{\partial x}\right)\mathbf{j} + \left(\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}\right)\mathbf{k}\end{aligned} \tag{3-45}$$

The discretized form of the vorticity equation is based on second order central differencing of the velocity field as given in equation 3-46.

$$\begin{aligned}\omega &= \nabla \times \bar{\mathbf{v}} \\ &\approx \left(\frac{\bar{w}_{j+1} - \bar{w}_{j-1}}{2\Delta y} - \frac{\bar{v}_{k+1} - \bar{v}_{k-1}}{2\Delta z}\right)\mathbf{i} \\ &+ \left(\frac{\bar{u}_{k+1} - \bar{u}_{k-1}}{2\Delta z} - \frac{\bar{w}_{i+1} - \bar{w}_{i-1}}{2\Delta x}\right)\mathbf{j} \\ &+ \left(\frac{\bar{v}_{i+1} - \bar{v}_{i-1}}{2\Delta x} - \frac{\bar{u}_{j+1} - \bar{u}_{j-1}}{2\Delta y}\right)\mathbf{k}\end{aligned} \tag{3-46}$$

3.6.2.3 *Reynolds Stress and turbulent kinetic energy*

Reynolds stress is a very important set of properties used to understand the anisotropic nature of turbulence in fluids. Reynolds stress is a tensor that describes variance and covariance of the fluctuating velocity components in turbulent flow and serves as a key component in Reynolds Averaged Navier-Stokes (RANS) turbulence models, such as the k-ε model. To see how Reynolds stress plays a role in fluid flow, start with the Navier-Stokes equations for flow of Newtonian fluids which can be summarized in vector notation given in equation 3-47 and in Einstein summation convention as given in equation 3-48.

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u}\right) = -\nabla p + \mu \nabla^2 \mathbf{u} \tag{3-47}$$

$$\rho\left(\frac{\partial \mathrm{u}_i}{\partial t}+\mathrm{u}_j\frac{\partial}{\partial x_j}\mathrm{u}_i\right)=-\frac{\partial p}{\partial x_i}+\mu\frac{\partial^2 \mathrm{u}_i}{\partial {x_j}^2} \tag{3-48}$$

When the statistical nature of the flow is of interest for studies of turbulence, the Reynolds Averaged Navier-Stokes form of the equations can be used. In this formulation the velocities are represented as two components, the mean component of velocity $\bar{u}_i$ and fluctuating component of velocity u_i'.

$$\mathrm{u}_i=\bar{\mathrm{u}}_i+\mathrm{u}_i' \tag{3-49}$$

Based on this definition and through rules of ensemble averaging, the well-known Reynolds Averaged Navier-Stokes equations are given in 3-50:

$$\rho\left(\frac{\partial \bar{\mathrm{u}}_i}{\partial t}+\bar{\mathrm{u}}_j\frac{\partial \bar{\mathrm{u}}_i}{\partial x_j}\right)=\frac{\partial}{\partial x_j}\left(-\bar{p}\delta_{ij}+\mu\frac{\partial \bar{\mathrm{u}}_i}{\partial x_j}-\tau_{ij}'\right) \tag{3-50}$$

Where τ_{ij}' is the Reynolds stress tensor defined by the following notation.

$$\begin{aligned}\tau_{ij}' &= \rho\overline{\mathrm{u}_i'\mathrm{u}_j'} \\ &= \rho\begin{bmatrix}\overline{\mathrm{u}_1'\mathrm{u}_1'} & \overline{\mathrm{u}_1'\mathrm{u}_2'} & \overline{\mathrm{u}_1'\mathrm{u}_3'}\\ \overline{\mathrm{u}_2'\mathrm{u}_1'} & \overline{\mathrm{u}_2'\mathrm{u}_2'} & \overline{\mathrm{u}_2'\mathrm{u}_3'}\\ \overline{\mathrm{u}_3'\mathrm{u}_1'} & \overline{\mathrm{u}_3'\mathrm{u}_2'} & \overline{\mathrm{u}_3'\mathrm{u}_3'}\end{bmatrix} \\ &= \rho\begin{bmatrix}\overline{u'^2} & \overline{u'v'} & \overline{u'w'}\\ \overline{v'u'} & \overline{v'^2} & \overline{v'w'}\\ \overline{w'u'} & \overline{w'v'} & \overline{w'^2}\end{bmatrix}\end{aligned} \tag{3-51}$$

The Reynolds stress tensor is symmetric and contains six independent values, three shear stress terms ($\overline{u'v'}$, $\overline{u'w'}$, and $\overline{v'w'}$) corresponding to the covariance between each of the velocity components and three normal stress terms ($\overline{u'^2}$, $\overline{v'^2}$, and $\overline{w'^2}$) corresponding to the variance of the three velocity components. Reynolds stress components are derived from the mean and instantaneous velocity components according to the definition of samples variance and covariance as shown below (equations 3-52 and 3-53).

$$\overline{u'^2} = \mathrm{Var}(u) = \frac{\sum_{r=1}^{n}\left(u_r - \bar{u}\right)^2}{n} \tag{3-52}$$

$$\overline{u'v'} = \mathrm{Cov}(u,v) = \frac{\sum_{r=1}^{n}\left(u_r - \bar{u}\right)\left(v_r - \bar{v}\right)}{n} \tag{3-53}$$

By accumulating the instantaneous velocity components of particles as they move through the statistical accumulator grid, the Reynolds stress tensor at each cell is simply calculated as the variance and covariance of the accumulated velocities within the cell.

The Reynolds stress values are an important indicator of the nature of turbulence. For example in anisotropic turbulence often found in natural flows, the normal and shear terms can provide a better characterization of how the turbulences varies in each direction. This information can be helpful in the design of fluid systems. With the ability to derive distributions of Reynolds stress, the LPT system will provide a way to quantitatively compare measured values with results from common RANS turbulence models.

The turbulent kinetic energy (TKE) is the kinetic energy per unit mass contained in small eddies of a flow field. TKE at a given point in space is measured as the root-mean-square of velocity fluctuations at that point (equation 3-54).

$$TKE = \frac{1}{2}\overline{\mathrm{u}_i'^2} = \frac{1}{2}\left(\overline{u'^2} + \overline{v'^2} + \overline{w'^2}\right) \tag{3-54}$$

3.6.2.4 ***Static pressure***

Over the last decade there has have been progress towards obtaining the static pressure field of a flow based on the particle motion of fluid tracers (Charonko, King, Smith, & Vlachos, 2010; De Kat & Van Oudheusden, 2012; Jaw, Chen, & Hwang, 2009; Murai, Nakada, Suzuki, & Yamamoto, 2007). Most of this research has focused on deriving the pressure field from velocity data obtained through Particle Image Velocimetry (PIV). One of the limitations of PIV

is that it is typically used to measure instantaneous flow velocities in a plane. Therefore, assumptions have to be made regarding the out of plane terms in the Navier-Stokes equation in order to solve for the pressure field (Violato, Moore, & Scarano, 2011). Very few studies have been published for 3D measurements (3D-PTV or LPT) and their use in pressure field calculation. For 3D LPT, velocities in all three dimensions are known therefore the solution for pressure becomes much more accessible. Two static pressure calculation methods were implemented in the real-time LPT system and will be discussed here; 1) RANS method and 2) Instantaneous Lagrangian Acceleration (ILA) method.

For incompressible flows the Navier-Stokes equations can be solved for the pressure gradient as shown in equation 3-55. In this case, if the velocity vector field **u** is known through measurement, then the resulting pressure field can be calculated based on solving the Pressure Poisson given in equation 3-56. The Poisson equation is typically solved with an iterative sparse-matrix relaxation method such as Successive Over Relaxation (SOR) (Heath, 2002; Moin, 2001) or a line integration method (Charonko et al., 2010). The boundary conditions are typically Neumann, where the pressure gradient is specified through 3-55 using first or second order forward differences (Charonko et al., 2010).

$$\nabla p = -\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) + \mu\nabla^2\mathbf{u} \tag{3-55}$$

$$\nabla^2 p = \nabla\left\{-\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) + \mu\nabla^2\mathbf{u}\right\} \tag{3-56}$$

$$\nabla^2 \bar{p} = \nabla\Phi$$

Implementing this process in a single step is problematic due to the source term found on the right-hand-side of equation 3-56. The large number of terms deriving from taking the gradient of the right-hand-side of equation 3-55 can prove challenging to implement in code. Therefore, in this implementation the Pressure Poisson equation in 3-56 is evaluated in a two-step process. In the first step, the 3D pressure gradients are evaluated in each cell of the domain. The second step is then to use these pressure gradients as source terms to solve the Poisson equation 3-56, where the symbol Φ represents the three pressure gradient components in a given cell.

The right and left hand sides of the pressure Poisson equation are discretized using second order central differences for the first and second derivatives as given in 3-57 and 3-58 respectively.

$$\nabla^2 \bar{p} \approx \frac{\bar{p}_{i+1} - 2\bar{p}_i + \bar{p}_{i-1}}{\Delta x^2} + \frac{\bar{p}_{j+1} - 2\bar{p}_j + \bar{p}_{j-1}}{\Delta y^2} + \frac{\bar{p}_{k+1} - 2\bar{p}_k + \bar{p}_{k-1}}{\Delta z^2} \tag{3-57}$$

$$\nabla \Phi \approx \frac{\Phi_{i+1} - \Phi_{i-1}}{2\Delta x} + \frac{\Phi_{j+1} - \Phi_{j-1}}{2\Delta y} + \frac{\Phi_{k+1} - \Phi_{k-1}}{2\Delta z} \tag{3-58}$$

After the pressure gradients at each location in the grid have been calculated, as will be discussed below, the pressure Poisson equation (3-56) can be solved iteratively using SOR (Moin, 2001). The pressure gradient is specified at the boundaries to create a Neumann boundary condition, where equation 3-55 is evaluated with first order forward differences. The two methods used to calculate the pressure gradients will be discussed next.

Reynolds Averaged Navier-Stokes (RANS) method:

For stationary flow fields the steady state Reynolds Averaged Navier-Stokes formulation can be used to estimate the gradient of mean pressure as given in 3-59. For the real-time LPT system, the mean velocity components and Reynolds stress tensor in each cell of the statistical accumulator grid are known through measurement, and therefore equation 3-59 can be readily evaluated using the discretized second order accurate finite difference approximations given in 3-60, 3-61, and 3-62.

$$\nabla \bar{p} = -\rho \bar{\mathrm{u}}_j \frac{\partial \bar{\mathrm{u}}_i}{\partial x_j} + \mu \frac{\partial^2 \bar{\mathrm{u}}_i}{\partial x_j^2} - \frac{\partial \tau'_{ij}}{\partial x_j} = \Phi \tag{3-59}$$

$$\begin{aligned} \bar{\mathrm{u}}_j \frac{\partial \bar{\mathrm{u}}_i}{\partial x_j} \approx & \left(\bar{u} \frac{\bar{u}_{i+1} - \bar{u}_{i-1}}{2\Delta x} + \bar{v} \frac{\bar{u}_{j+1} - \bar{u}_{j-1}}{2\Delta y} + \bar{w} \frac{\bar{u}_{k+1} - \bar{u}_{k-1}}{2\Delta z} \right) \mathbf{i} \\ & + \left(\bar{u} \frac{\bar{v}_{i+1} - \bar{v}_{i-1}}{2\Delta x} + \bar{v} \frac{\bar{v}_{j+1} - \bar{v}_{j-1}}{2\Delta y} + \bar{w} \frac{\bar{v}_{k+1} - \bar{v}_{k-1}}{2\Delta z} \right) \mathbf{j} \\ & + \left(\bar{u} \frac{\bar{w}_{i+1} - \bar{w}_{i-1}}{2\Delta x} + \bar{v} \frac{\bar{w}_{j+1} - \bar{w}_{j-1}}{2\Delta y} + \bar{w} \frac{\bar{w}_{k+1} - \bar{w}_{k-1}}{2\Delta z} \right) \mathbf{k} \end{aligned} \tag{3-60}$$

$$\frac{\partial^2 \overline{\mathrm{u}}_i}{\partial x_j^2} \approx \left(\frac{\overline{u}_{i+1} - 2\overline{u} + \overline{u}_{i-1}}{\Delta x^2} + \frac{\overline{u}_{j+1} - 2\overline{u} + \overline{u}_{j-1}}{\Delta y^2} + \frac{\overline{u}_{k+1} - 2\overline{u} + \overline{u}_{k-1}}{\Delta z^2} \right) \mathbf{i}$$
$$+ \left(\frac{\overline{v}_{i+1} - 2\overline{v} + \overline{v}_{i-1}}{\Delta x^2} + \frac{\overline{v}_{j+1} - 2\overline{v} + \overline{v}_{j-1}}{\Delta y^2} + \frac{\overline{v}_{k+1} - 2\overline{v} + \overline{v}_{k-1}}{\Delta z^2} \right) \mathbf{j} \qquad (3\text{-}61)$$
$$+ \left(\frac{\overline{w}_{i+1} - 2\overline{w} + \overline{w}_{i-1}}{\Delta x^2} + \frac{\overline{w}_{j+1} - 2\overline{w} + \overline{w}_{j-1}}{\Delta y^2} + \frac{\overline{w}_{k+1} - 2\overline{w} + \overline{w}_{k-1}}{\Delta z^2} \right) \mathbf{k}$$

(3-62)

$$\frac{\partial \tau'_{ij}}{\partial x_j} \approx \rho \left(\begin{array}{l} \dfrac{\left(\overline{u'^2}_{i+1} - \overline{u'^2}_{i-1}\right) + \left(\overline{v'u'}_{i+1} - \overline{v'u'}_{i-1}\right) + \left(\overline{w'u'}_{i+1} - \overline{w'u'}_{i-1}\right)}{2\Delta x} \\ + \dfrac{\left(\overline{u'^2}_{j+1} - \overline{u'^2}_{j-1}\right) + \left(\overline{v'u'}_{j+1} - \overline{v'u'}_{j-1}\right) + \left(\overline{w'u'}_{j+1} - \overline{w'u'}_{j-1}\right)}{2\Delta y} \\ + \dfrac{\left(\overline{u'^2}_{k+1} - \overline{u'^2}_{k-1}\right) + \left(\overline{v'u'}_{k+1} - \overline{v'u'}_{k-1}\right) + \left(\overline{w'u'}_{k+1} - \overline{w'u'}_{k-1}\right)}{2\Delta z} \end{array} \right) \mathbf{i}$$
$$+ \rho \left(\begin{array}{l} \dfrac{\left(\overline{u'v'}_{i+1} - \overline{u'v'}_{i-1}\right) + \left(\overline{v'^2}_{i+1} - \overline{v'^2}_{i-1}\right) + \left(\overline{w'v'}_{i+1} - \overline{w'v'}_{i-1}\right)}{2\Delta x} \\ + \dfrac{\left(\overline{u'v'}_{j+1} - \overline{u'v'}_{j-1}\right) + \left(\overline{v'^2}_{j+1} - \overline{v'^2}_{j-1}\right) + \left(\overline{w'v'}_{j+1} - \overline{w'v'}_{j-1}\right)}{2\Delta y} \\ + \dfrac{\left(\overline{u'v'}_{k+1} - \overline{u'v'}_{k-1}\right) + \left(\overline{v'^2}_{k+1} - \overline{v'^2}_{k-1}\right) + \left(\overline{w'v'}_{k+1} - \overline{w'v'}_{k-1}\right)}{2\Delta z} \end{array} \right) \mathbf{j}$$
$$+ \rho \left(\begin{array}{l} \dfrac{\left(\overline{u'w'}_{i+1} - \overline{u'w'}_{i-1}\right) + \left(\overline{v'w'}_{i+1} - \overline{v'w'}_{i-1}\right) + \left(\overline{w'^2}_{i+1} - \overline{w'^2}_{i-1}\right)}{2\Delta x} \\ + \dfrac{\left(\overline{u'w'}_{j+1} - \overline{u'w'}_{j-1}\right) + \left(\overline{v'w'}_{j+1} - \overline{v'w'}_{j-1}\right) + \left(\overline{w'^2}_{j+1} - \overline{w'^2}_{j-1}\right)}{2\Delta y} \\ + \dfrac{\left(\overline{u'w'}_{k+1} - \overline{u'w'}_{k-1}\right) + \left(\overline{v'w'}_{k+1} - \overline{v'w'}_{k-1}\right) + \left(\overline{w'^2}_{k+1} - \overline{w'^2}_{k-1}\right)}{2\Delta z} \end{array} \right) \mathbf{k}$$

Instantaneous Lagrangian Acceleration (ILA) Method:

An alternative approach to calculating the static pressure field was derived based on the instantaneous Lagrangian acceleration of seed particles and resulting local trajectory pressure gradient, as discussed in the Lagrangian section 3.6.1.2. In this method, the local pressure gradient along each trajectory is accumulated into the statistical accumulator grid in the same weighted fashion as described above for velocity and acceleration. Then the method to solve for the individual cell pressures is the same as for the RANS method, where equation 3-56 is solved using SOR and iterated until convergence.

3.6.2.5 *3D visualization*

The Eularian properties of the particle flow field can be visualized using the interactive rendering environment, implemented with VTK. The vector fields of velocity and vorticity are displayed as vectors colored by magnitude (Figure 9). The vectors can be set to uniform scaling or sized based on the magnitude to emphasize the relative spatial differences. As with the Lagrangian trajectory visualization, the user can interactively change the perspective view of the vector field.

An interactive view plane tool has been implemented to probe the volume and view planar distributions of mass residual, turbulence intensity, velocity magnitude, velocity variance, vorticity magnitude, static pressure and sample count per cell. Figure 10 shows the interactive view plane tool displaying turbulence intensity profile along the center plane of an axisymmetric jet. The user can click, grab, move and orient the plane in order to view different slices of the 3D domain.

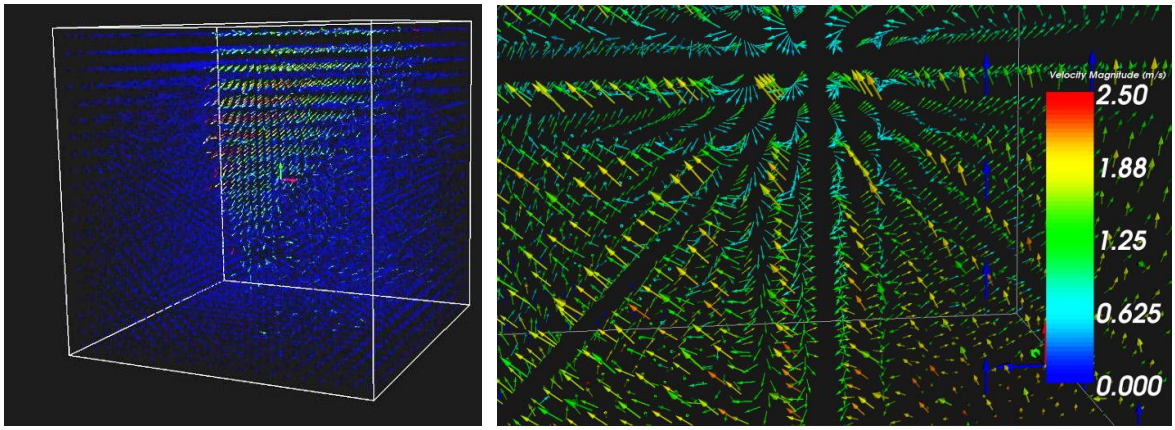

Figure 9: Visualization of the statistical accumulator grid with velocity vectors displayed

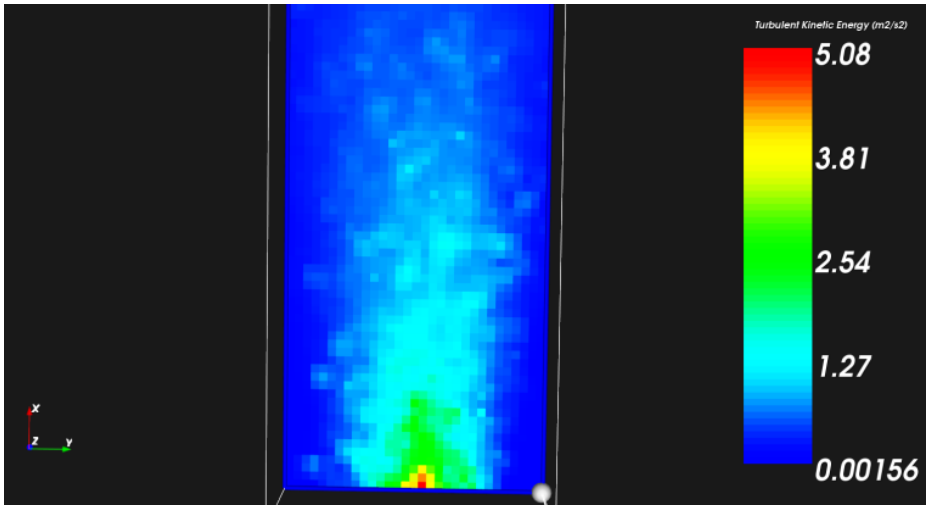

Figure 10: Visualization of time averaged values of Turbulent Kinetic Energy in the center plane of the statistical accumulator grid

3.7 Summary of LPT algorithm development

The major algorithms of the real-time LPT system were described in detail following the general flow of data processing through the system: imaging and particle detection, solving the multi-camera correspondence problem, 3D reconstruction, temporal tracking, data analysis and visualization. The key points are summarized as follows:

- The image processing and particle detection algorithm is based on image segmentation and centroid calculation based on the spatial moments of the contours around particle image "blobs". Particle image locations are corrected for lens distortion through the use of an eight parameter nonlinear lens distortion model.

- The multi-camera correspondence algorithm was based on strict match criteria in which four cameras must simultaneously satisfy the epipolar constraint in order for a set of particle images to be considered a match. The algorithm was divided into two stages to expose parallelism in evaluation of the epipolar constraint, which will be discussed in the next chapter. The combinatorial algorithm is designed to work with an arbitrary number of cameras to allow easy scaling beyond a traditional four camera system.

- The 3D reconstruction algorithm was based on the classical pinhole camera projection model and a particle's 3D world coordinates are solved for based on the linear least squares formulation involving all cameras in a the four-camera correspondence group.

- A new temporal tracking algorithm was developed based on a new approach which is less computationally complex than regression based multi-frame tracking algorithms. This new method sorts matches based on a computationally inexpensive cost function and minimizes tracking ambiguity by strictly matching each candidate particle with only one existing trajectory.

- A data analysis and visualization module was created to display particle tracking results in Lagrangian and Eularian reference frames. The Lagrangian properties calculated include instantaneous velocity, acceleration and static pressure gradient. These Lagrangian properties are attributed to a structured statistical accumulator grid comprised of cubic elements in a Eularian reference frame. The weighted means and variance (covariance) of velocity and acceleration are used to calculate time-averaged Eularian properties including mass residual, vorticity, Reynolds stress, turbulence intensity and static pressure.

4 Development of the real-time data processing framework

4.1 Overview and objectives

This chapter will describe the development and performance evaluation of a real-time streaming implementation of the LPT algorithms discussed in Chapter 3. To meet the real-time particle tracking objective of this research, a multi-layer parallel processing framework was developed.

The general approach to achieving real-time processing was to utilize fine and coarse grain parallelism on heterogeneous computing architectures. In general, this type of machine would consist of a cluster of computing nodes where each node is comprised of a multi-core CPU and a set of graphics processor units (GPU). A general representation of the target computing system is shown in Figure 11. The LPT real-time processing framework is divided into two key components: 1) Node based streaming framework, and 2) Cluster based message passing framework.

The node based streaming framework is designed around a pipelining scheme where each of the five major LPT tasks (particle detection, multi-camera correspondence, 3D reconstruction, temporal tracking, and visualization) are connected in a pipeline. Data passing through this pipeline is processed in parallel with multiple CPU threads assigned to each of the tasks. The most computationally intensive tasks of particle detection and multi-camera correspondence are broken down further to expose parallelization across multiple CPU threads and GPU accelerators. This framework is designed to work on any multi-core computer including commodity workstations and laptops in addition to nodes of a high performance cluster.

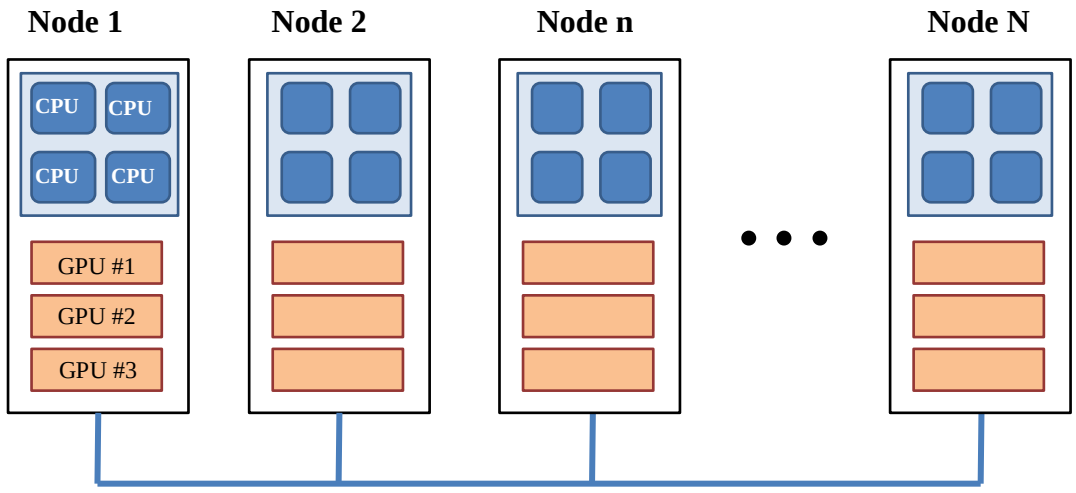

Figure 11: Heterogeneous computing architecture model - A cluster composed of nodes, each with a multi-core processor and set of GPU accelerators

The cluster based message passing framework is designed to pipeline the LPT data by groups of frames to be processed simultaneously on a cluster of nodes each running the node based streaming framework (Barker, Lifflander, Arya, & Zhang, 2012). This multi-layer parallel processing approach will allow the real-time LPT system to scale with an increasing number of cameras, frame rates and particle concentrations.

4.2 Node based streaming framework

4.2.1 Multi-threaded pipelining scheme

The objective of the node based streaming framework is to fully utilize multi-core processors by pipelining the LPT data across CPU cores as it streams in from the cameras. Each LPT task is assigned a group of CPU threads to process the data in parallel. A thread of execution is a unit of work that is processed by a CPU, and each CPU can be scheduled to work on multiple threads at a time. The multi-threaded data processing pipeline is implemented as a classical multiple producer/consumer model, where each task is a consumer of the data output by the preceding task and producer of data for the subsequent task. The difficulty is in synchronizing the CPU threads and ensuring that data flows through the pipeline efficiently without accumulating and overcoming the computer's memory.

To implement the multi-threaded pipeline, a concurrent queue data structure was utilized. The concurrent queue is a First-In-First-Out FIFO data structure, where data packets placed in the back of the queue by a producer thread(s) are removed from the front of the queue by a consumer thread(s). In the LPT system the packets of data will be referred to as frames, which could be arrays of image frames from the cameras, arrays of 2D particle image centroids, or arrays of 3D particle locations at a given time step. The node based streaming pipeline structure is shown in Figure 12. As frames stream in from the cameras they are pushed into the back of the first queue and popped from the front of the queue by the image processing thread. The image processing thread detects the particles, removes image distortion, pushes an array of particle centroids into the second queue, and grabs another set of images from the first queue. The multi-camera correspondence thread removes the arrays of particle centroids from queue two, finds all the matching particle images, places the matches in queue three, and grabs the next set of particle centroids. The 3D reconstruction thread removes the matches from queue 3, solves for the 3D coordinates of each particle and places the frames of 3D particles in queue 4 for the tracking thread.

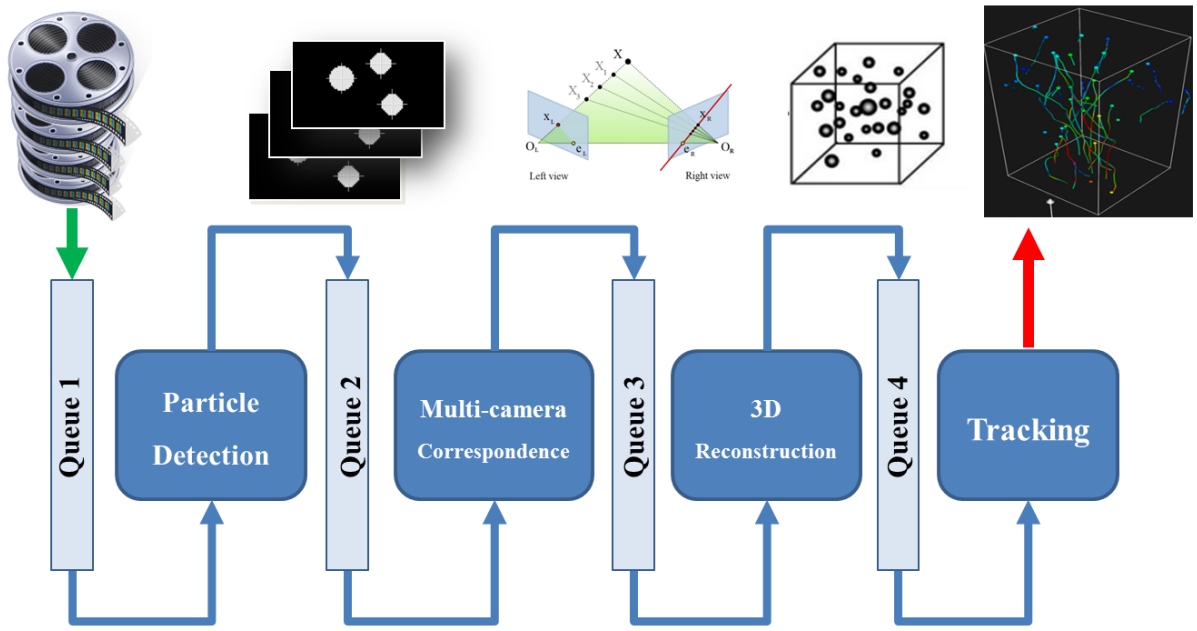

Figure 12: Streaming pipeline utilizing concurrent queue data structures

Each queue is assigned a fixed amount of memory. Therefore each thread in the pipeline must consume frames fast enough to prevent a queue from overflowing its memory allocation. If a

thread cannot keep up and the queue's memory limit is reached, then the pipeline breaks down and data are lost. To prevent this, each of the four major tasks has to be optimized to reach a balanced frame processing rate across all tasks. The first two tasks, particle detection and multi-camera correspondence, are the most computationally expensive. These two tasks were selected for optimization and are discussed next.

4.2.2 Image processing and particle detection parallelization

As discussed in Chapter 2, the image processing task can be more efficiently handled by "smart" cameras. In this way, the camera completes the image segmentation to identify the particle image "blobs" and transfers up to 1000x less data to the computer. Therefore the image processing and threshold segmentation steps were given to the cameras and the computer was left with the centroid detection and distortion removal steps. The remaining particle detection and distortion removal steps were parallelized on the computer by assigning a thread to each camera. This method was shown to be very successful for the six camera prototype with will be discussed in Chapter 5. The remaining bottleneck in the LPT pipeline was the multi-camera correspondence problem.

4.2.3 Camera correspondence algorithm parallelization

Multi-camera correspondence is very computationally intensive and was selected for parallelization on the GPU. The core approach to optimizing this routine is to break it into two threads; one thread to evaluate the epipolar constraint equation for all particles of unique camera pairs and one thread to test for satisfaction of the four camera correspondence criteria. As noted in Chapter 3, the first stage of this problem is the most computationally expensive, but it also exposes a significant level of parallelism. In each camera pair AB, the particles from camera A can be compared with those in camera B without any dependencies. Therefore this is a perfect problem to be solved on the GPU as it is highly data parallel, where the same operation (epipolar constraint evaluation) is computed for different data.

The GPU implementation of the camera correspondence problem was done with NVIDIA's Compute Unified Device Architecture (CUDA) programming paradigm. CUDA is an extension of the C/C++ language to allow efficient programming of graphics processors for general compute purposes. The parallel GPU algorithm for the correspondence problem is as follows:

1. Initialize the GPU with the Fundamental matrices for all camera pairs and allocate memory for particle data.
2. Grab a frame of particle centroid data from queue and asynchronously copy all particles from all cameras to the GPU
3. On the GPU, a thread is generated for each particle in camera A of each camera pair. Each thread loops through all particles in camera B and evaluates the epipolar constraint equation.
4. The CPU copies the resulting epipolar constraint residuals for each particle and sends the data to the second stage to evaluate four camera correspondence criteria.

4.2.4 **Time performance**

The compute node pipelining scheme was tested using the PIV Standard images data set #352 as described in section 4.3.2.2. This data set consists of 145 frames of particle data with approximately 300 particles per frame. Each of the LPT algorithms was verified for accuracy and ultimately tested within the nodal pipelining scheme to determine the real-time capability of the system. For the streaming algorithm to be considered real-time, no data accumulation can occur in the pipeline. Therefore the slowest task in the pipeline determines the ability of the whole pipeline to reach real time processing with respect to the input camera frame rate. Testing the four cameras and data set #352, the slowest task was the image processing task which was able to operate at 467 frames per second assuming a seeding density of 300 particles per frame. With six cameras, the slowest task was the multi-camera correspondence algorithm which processed frames at 273 frames per second.

4.3 Cluster based message passing framework

The goal of developing a cluster based message passing framework is to achieve scaling with increasing number of cameras, frame rates, and seed particle concentrations. This is achieved by designing the cluster based parallel algorithm to meet the following three requirements.

- **Consistent:** must provide results that are consistent with the serial version and not introduce tracking errors in the form of erroneous trajectories
- **Scalable:** must scale up from one to hundreds of processors without significant reductions in speedup per processor added
- **Adaptable:** must be modular and able to run any form of the compute node streaming framework discussed in the previous section.

4.3.1 Parallel implementation strategy

The LPT cluster based parallel algorithm was implemented in C++ and Charm++, allowing the algorithm to be programmed using object-oriented techniques. Charm++ is an object-oriented parallel programming paradigm that acts as an extension to the C++ language (Kale & Krishnan, 1993). It allows programming objects (ie: data structures, classes, etc.) to be distributed across multiple processors and asynchronously communicate by sending and receiving messages.

The general strategy used in this algorithm is to first decompose the LPT data into multiple sets of consecutive frames. These frame-sets are distributed between a group of processors where trajectory segments are built in parallel by executing the node based pipelining scheme to simultaneously execute all LPT algorithms. The trajectory segments from each frame-set are then compared with all segments in adjacent frame-sets to be merged into longer global trajectories. This approach is shown in Figure 13.

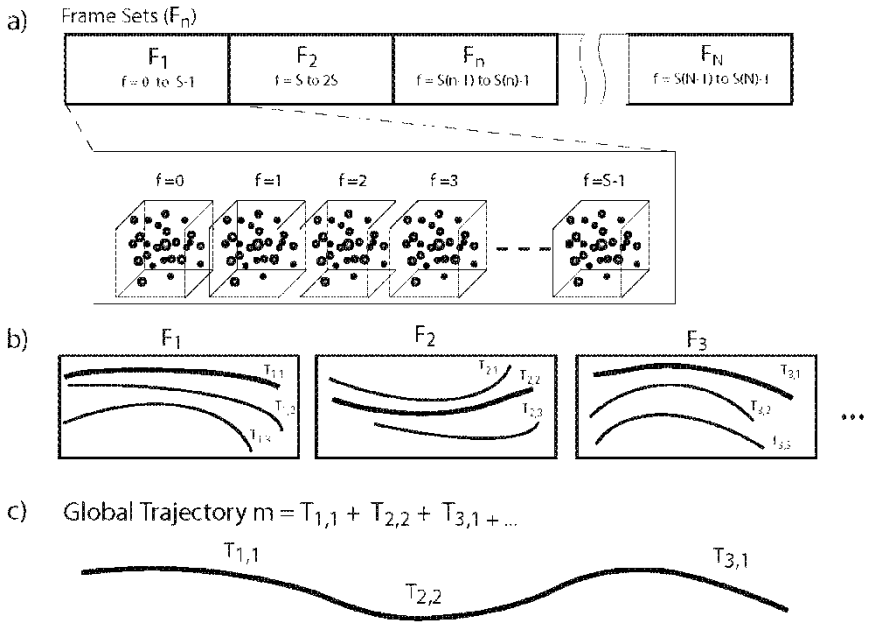

Figure 13: Parallel Implementation Strategy: (a) particle data are divided into *N* frame sets *F* of size *S* frames each, (b) trajectory segments are built in parallel and given local ID $T_{n,i}$, (c) trajectory segments are merged to create global trajectories and printed to file

Where $T_{n,i}$ is the local trajectory segment ID for trajectory segment i in frame-set n, S is the frame-set size, a tunable parameter for parallel decomposition, F_n Frame-set ID for the n^{th} frame-set, N is the total number of frame-sets

4.3.1.1 *Data decomposition*

The first step in parallelizing the LPT algorithms across clusters of nodes was to decompose the problem into multiple work units to be distributed across many processors. There are three possible data decomposition strategies for LPT: distributed particles, distributed frames, and distributed object space. The distribution of particles or object space would require extensive communication between processors that do not share memory. Therefore, frame decomposition was chosen because the communication costs are low and it exposed sufficient parallelism for both shared and distributed memory systems. In this decomposition, the data are divided into frame sets of size *S* consecutive frames which are distributed across processors as shown in Figure 13. The number of frames in a set *S*, is the parallel decomposition factor or frame-set size, and is left as a tunable parameter with a minimum value of eight frames as required for the trajectory merge operation. These frame sets can be processed in parallel using any sequential tracking algorithm. The key challenge to this approach lies in merging the disjoint trajectory segments without significant processor communication overhead.

4.3.1.2 *Trajectory Merging*

A merge operation is required to concatenate local trajectory segments spanning across a single frame-set into global trajectories that span multiple frame-sets. This operation falls between steps *b* and *c* in Figure 13 and begins once all local trajectory segments from two adjacent frame sets have been constructed. Therefore trajectory merging can happen asynchronously without waiting for all trajectory segments from all frame sets to be constructed. The linear regression based cost function developed by Li (2008) was used to determine if two local trajectory segments from adjacent frame-sets constitute a single trajectory. Only the tails of each trajectory, composed of first four and last four linked particles, are sent to the merge function in order to minimize data transfer between processors. The merge function compares all trajectories constructed within frame-set *n* to those in frame-set *n+1*, where *n* is the frame-set index from 1 to *N-1*. If the first particle of the trajectory segment from frame-set *n*+1 is within certain proximity to the last particle from frame-set *n* then the cost function is evaluated. As shown in Figure 14, the cost evaluation requires four iterations per candidate trajectory to fully examine the quality of fit for each particle in the tails. If the cost associated with each of the four iterations is below a set threshold beta then the two local trajectory segments are paired for merger.

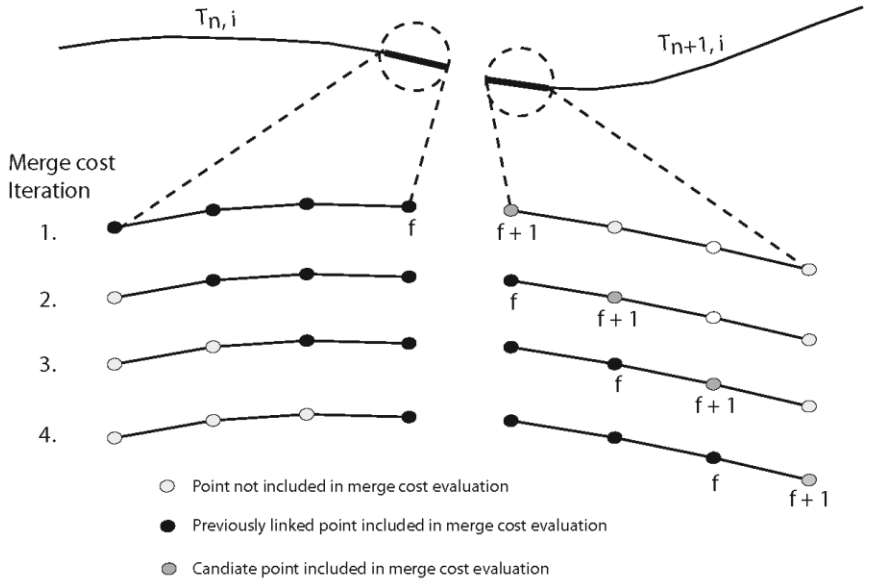

Figure 14: Trajectory merge operation: four cost function iterations are required to evaluate the merge of the last four linked particles of trajectories in frame-set F_n are with the first four linked particles of the trajectories in frame-set F_{n+1}

Table 3: Global merge map: final instructions for global trajectory assembly from local trajectory segments

	Frame set number				
Global Trajectory ID	1	2	n	$\cdots$	N
1	$T_{1,1}$	$T_{2,2}$	$T_{n,1}$	$\cdots$	$T_{N,13}$
2	$T_{1,34}$	$T_{2,7}$	$T_{n,19}$	$\cdots$	$T_{N,15}$
m	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
M	$T_{1,3}$	$T_{2,1}$	$T_{n,24}$	$\cdots$	$T_{N,4}$

Where *M* is the total number of global trajectories, *n* is the frame-set index, and *m* is the global trajectory index.

4.3.1.3 *Parallel communication and data flow*

Parallel particle tracking occurs in five steps 1) data input and distribution, 2) tracking, 3) merge identification, 4) global trajectory construction, and 5) trajectory data output to file. Figure 15 shows a simplified example of the parallel communication and flow using only two processors and particle data divided into four frame-sets, two per processor. In the first step the 3D particle location data are read from memory and distributed in frame-sets of *S* frames to the pool of processors. Next each processor runs the sequential tracking algorithm on its frame-sets to build local trajectory segments. Once the trajectory segments spanning two adjacent frame-sets have been constructed, the merge operation is conducted as discussed in the previous section. The result of the merge operation is a local mapping of trajectory pairings between adjacent frame-sets. The actual trajectory data remains fragmented across processors at this point and only the locally paired trajectory segment IDs are known by each processor. Once all local trajectory merges have been identified between each adjacent frame-set, the global trajectory construction process can begin. The purpose of this phase is to consolidate the trajectory segments belonging to a single global trajectory on the same processor. First, a set of instructions is generated that defines the segments to be merged along with their respective frame-set IDs and host processor. A sample of these instructions is shown in Table 3. Next, each processor selects an equal subset of global trajectories and begins communicating with the other processors to obtain the segments needed for their construction. Once a processor has received all of the contiguous trajectory segments and built the global trajectories it outputs them to a single file and exits. The final results are a series of files (one per processor) containing full length trajectories.

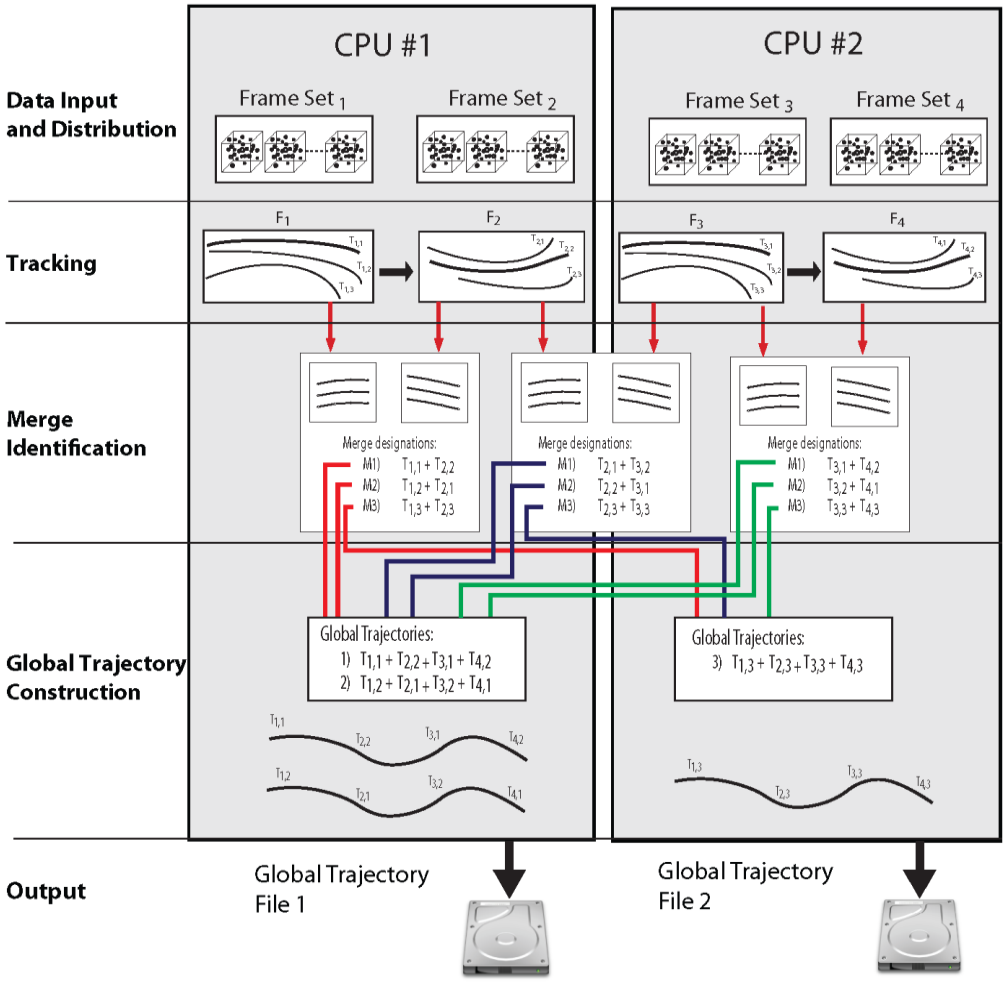

Figure 15: Parallel tracking algorithm flow diagram for a simplified case of three global trajectories spanning four frame sets on two processors

4.3.2 Parallel performance evaluation and results

The cluster based parallel LPT algorithm was evaluated for accuracy, consistency and scaling using three data sets. In this evaluation the only the tracking algorithm was used from the node based streaming framework. Thus the data input consisted on 3D particle locations grouped by frame and the output was reconstructed trajectories. The sequential regression based tracking algorithm by Li (2008), as described in Chapter 3, was used to evaluate the cluster based message passing framework.

The first data set was used to evaluate consistency with the sequential version and was obtained from the PIV 3D standard images data set #352 (Okamoto, Nishio, Kobayashi, Saga, & Takehara, 2000). The second set consisted of a large data set with uniform characteristics and

was used to evaluate the optimal parallel performance of the algorithm on several large clusters. The third data set was generated using computational fluid dynamics (CFD) and used to test the parallel performance with non-uniform data and inherent work load imbalance across processors. A wide range of machines were used in the evaluation including a desktop workstation, a moderate-size cluster (Turing) and one very large cluster (BlueGene/P). The specifications of these machines can be found in Table 4.

Table 4: Computer systems used in parallel algorithm evaluation

System Name	Number of Processors	CPU Architecture	RAM
Multi-core workstation	2	Intel(R) Xeon(R) E5530 2.4 GHz quad-core CPUs	16 GB/CPU
Turing Cluster	1536	Apple G5 2 GHz X-serve cluster	4 GB/node
BlueGene/P Cluster	163,840	PowerPC 450 CPUs 850 MHz	2 GB/node

4.3.2.1 *Performance metrics*

The trajectory reconstruction accuracy of the algorithm is measured by two key metrics: the coverage ratio and correct ratio as shown in the equations below. Coverage ratio ($\gamma_{coverage}$) is the ratio of correct two-frame particle links made during the tracking process ($L_{correct}$) to the total number of known input links (L_{total}) (Li et al., 2008). A coverage value of 1.0 indicates that all of the input particles were tracked correctly. Correct ratio ($\gamma_{correct}$) refers to the number of correct links made with respect to the total number of links established in the tracking process ($L_{tracked}$) (Li et al., 2008). A correct ratio of 1.0 indicates all established particle links were accurately reconstructed.

$$\gamma_{coverage} = \frac{L_{correct}}{L_{total}}$$
$$\gamma_{correct} = \frac{L_{correct}}{L_{tracked}} \qquad (4\text{-}1)$$

Parallel performance can be measured in terms of speedup and scalability. The speedup of the parallel algorithm is the ratio of serial execution time to parallel execution time given the same work (equation 4-2). The algorithm is timed from data input to data output excluding reading and writing of data from/to the hard drive. The measure of how well a parallel application scales is the ratio of speedup achieved to the number of processors. The optimal case is when the speedup divided by the number of processors is equal to one, in which case perfect scaling is

observed. However, in real applications adding processors creates overhead and eventually a loss in parallel efficiency is observed.

$$speedup = \frac{\text{sequential time}}{\text{parallel time}} \tag{4-2}$$

4.3.2.2 *PIV standard 3D images data set #352*

The standard 3D images data set #352 from Okamoto (2000) was selected to test the parallel algorithm for trajectory reconstruction accuracy and consistency in comparison with the serial version. This simulated data set consists of an average of 300 particles per frame in three cameras over 145 frames (Okamoto et al., 2000) and was obtained at www.piv.jp/image3d/image352. The flow field is 2 cm x 2 cm x 2 cm and contains a jet impinging on a wall with inlet speed of 15 cm/s and Reynolds number of 3000. A subset of 3D trajectories from this set is shown in Figure 16. Accuracy was measured by comparing the output trajectories with the true trajectories from the known input data. This data set is too small for a full evaluation of the parallel scaling and speedup, which are evaluated in the following sections.

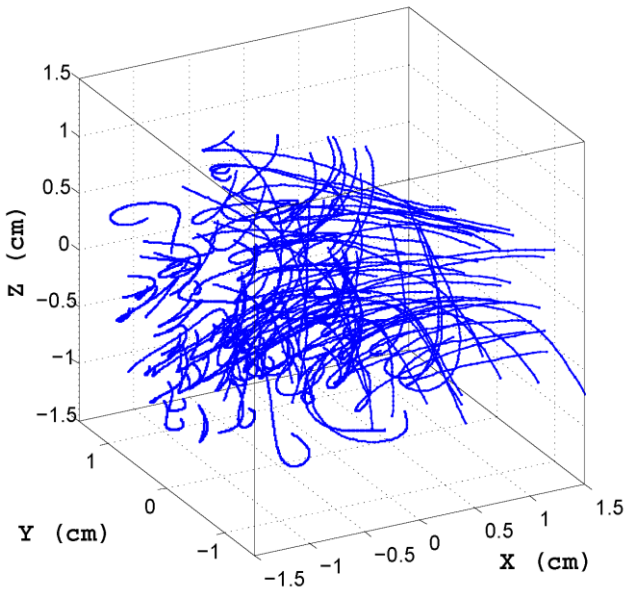

Figure 16: Trajectories from standard PIV data set #352 spanning 75 frames or greater (Okamoto et al., 2000)

Five tracking runs were completed on this data set as shown in Table 5. The first run was conducted with the serial algorithm to build the performance benchmark followed by four runs of the parallel algorithm with different frame set decomposition sizes (8, 16, 32, and 64 frames) on a desktop workstation with two quad-core processors.

Table 5: Trajectory reconstruction results using standard PIV data set with no noise #352 (Okamoto et al., 2000)

Number of Processors	Frames / Set	Run Time (s)	$\gamma_{correct}$	$\gamma_{coverage}$	Avg. Trajectory Length	Trajectories Tracked
2	8	0.619	0.984	0.923	32	1400
2	16	0.662	0.981	0.942	32	1444
2	32	0.692	0.980	0.954	32	1463
2	64	0.707	0.980	0.958	32	1468
1 (serial)	145	0.640	0.979	0.960	32	1471

To evaluate the algorithm in the presence of noise, the data set was heavily modified and used for reevaluation. Ten percent of the known particles were randomly selected for removal to simulate occlusion, ten percent more ghost particles were randomly added throughout the domain to simulate false detections and all particle positions were perturbed by an average of 0.003 cm in each dimension (equivalent to a 0.5 pixel error in particle centroid localization) to simulate common detection uncertainty.

Table 6: Trajectory reconstruction results using standard PIV data set with heavy noise #352 (Okamoto et al., 2000)

Number of Processors	Frames / Set	Run Time (s)	$\gamma_{correct}$	$\gamma_{coverage}$	Avg. Trajectory Length	Trajectories Tracked
2	8	0.485	0.992	0.245	9	1375
2	16	0.529	0.990	0.288	9	1627
2	32	0.554	0.980	0.311	9	1762
2	64	0.570	0.989	0.319	9	1805
1 (serial)	145	0.510	0.989	0.324	9	1831

The results show that tracking was consistent between the serial and parallel versions, achieving average tracking correct ratios of 0.98 and average coverage ratios of 0.94. The average length of the trajectories remains at 32 frames and is consistent with the serial results and input data. This indicates that the merge operation is performing successfully. When the frame set size, *S*, is reduced from 64 frames to 8 the correct tracking ratio increases slightly while the coverage ratio decreases slightly. This is acceptable deviation since the accuracy (correctness) of the tracked particles remains constant and no tracking errors are introduced. Overall, the parallel algorithm was successful in preserving long and accurate trajectories when noise is low. In the presence of heavy noise and uncertainty, the performance diminishes significantly. The results of the parallel algorithm's performance in the presence of noise are shown in Table 6. While

the coverage ratio decreases with added noise it is important to note that the correct ratio is still near 99 percent.

4.3.2.3 *Simulated vortex for parallel performance evaluation*

A large uniform data set was created to test the optimal parallel performance of the algorithm under near perfect load balancing for up to 512 processors. This data set consists of a 1024 particles moving with uniform acceleration in a downward spiral through a 2m x 2m x 6m domain as shown in Figure 17. The spacing-displacement ratio was greater than 10 in order to ensure 100 percent tracking coverage and accuracy. All trajectories are equal length and span 8192 frames, therefore each frame contains the same number of particles and parallel workload is balanced. This type of data set eliminates the possibility of tracking errors and permits isolated evaluation of the parallel performance in terms of scaling efficiency and speedup. To assist in the evaluation, the data set was parsed to create six total sets representing three variations in total particles (1024, 512 and 256 particles) and three trajectory lengths (8192, 4096 and 2048 frames). The particle trajectories are described by equations 4-3, where θ and *d* are the angle and diameter of rotation, and $[x_o, y_o, z_o]$ is the particle's random location in the initial frame.

$$
\begin{aligned}
x &= x_o + \frac{d}{2} sin(\theta) \\
y &= y_o + \frac{d}{2} cos(\theta) \\
z &= z_o + \frac{d}{2\pi}\theta
\end{aligned}
\tag{4-3}
$$

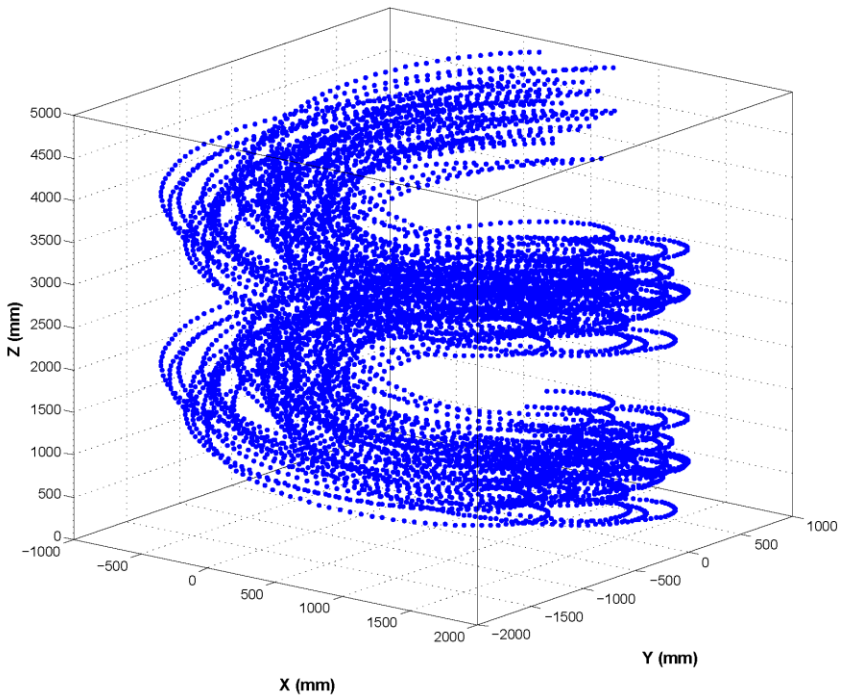

Figure 17: Simulated trajectories for scaling analysis

The Turing cluster was used to evaluate the impact varying the frame decomposition (frame-set size) from 8 to 256 frames on the parallel performance for a fixed number of processors. The BlueGene/P cluster was used to test the scalability and speedup when the number of frames and particles are varied.

Table 7: Impact of parallel decomposition factor, frame set size (*S*), on parallel speedup (Turing cluster) 1024 particles 8192 frames

Number of processors	frame-set size S	frame-sets per processor	time (s)	speed up	processed frames per second
1 (serial)	8192	1	3662.57	1.00	2.23
32	8	32	66.79	54.84	122.65
32	16	16	69.22	52.91	118.35
32	32	8	67.95	53.90	120.56
32	64	4	69.37	52.80	118.09
32	128	2	70.04	52.29	116.96
32	256	1	85.74	42.72	95.54

Table 7 shows how the parallel decomposition factor, the frame-set size *S*, impacts speedup. With 32 processors working on the 1024 particle 8192 frame data set, the speedup remains nearly constant for all frame-set sizes until the number of frame-sets per processor approaches one. Once this happens, the processors are unable to hide communication latency by overlapping communication with computation. Thus, two or more frame-sets should be assigned to each processor to minimize idle time. A frame-set size of $S = 8$ frames was selected

for the following analysis to ensure at least 512 processors could be used with the largest data set. The speed up of 54 achieved in this evaluation was greater than the number of processors used indicating that the parallel algorithm has better memory characteristics than the sequential version due to slight differences in implementation.

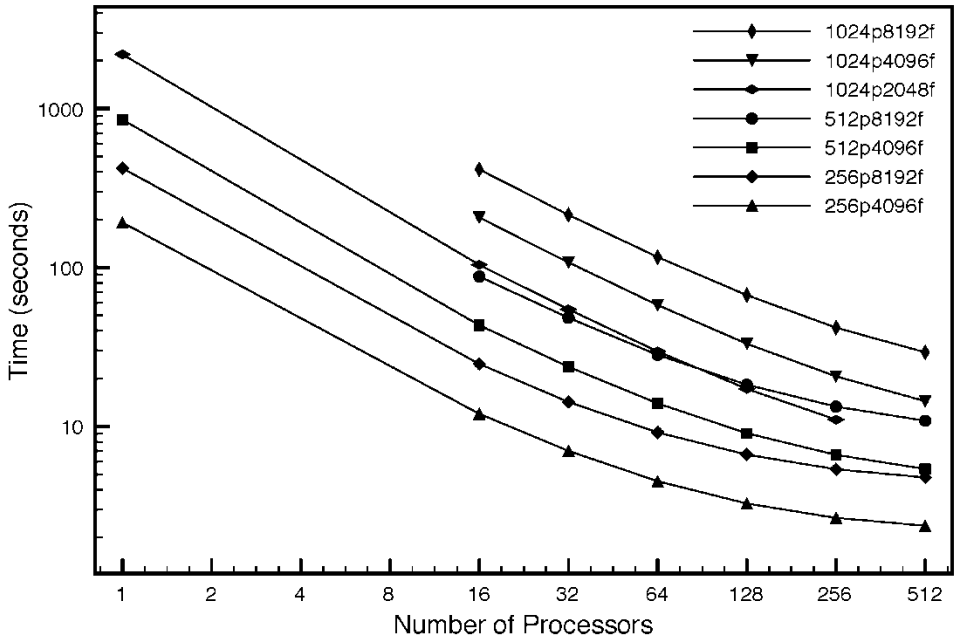

Figure 18: Scaling results from parallel execution on BlueGene/P (data sets are labeled by ApBf where A is the number of particles and B is the number of frames)

Figure 18 shows strong scaling of the six data sets up to 512 processors on the BlueGene/P cluster. This graph demonstrates the impact of diminishing returns and loss of parallel efficiency as the number of processors increase. The run time for the data set with 1024 particles remains at a near constant slope with added processors while the data set with only 256 particles begins to reduce in slope as inefficiencies arise. Clearly the data set with more particles has more work and can be processed more efficiently with a larger number of processors. Thus the program scales very well with an increasing number of particles tracked. The slopes of the performance curves for data sets of common particle numbers are nearly equal when the number of frames is 4096 or 8192, indicating that the number of frames processed has little impact on the scaling performance.

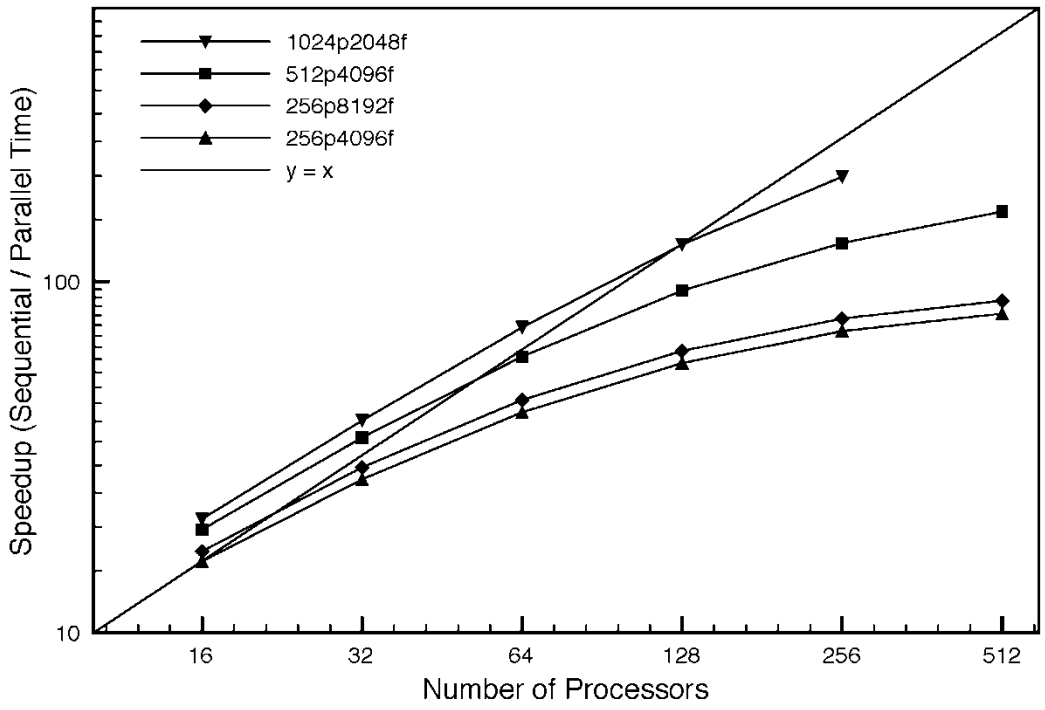

Figure 19: Speedup graph from parallel execution on BlueGene/P (data sets are labeled by ApBf where A is the number of particles and B is the total number of frames)

Figure 19 shows the speedup over the sequential algorithm. The straight line represents a linear speedup and perfect scaling. For the first two points on the 512 particles 4096 frames data set and the three points on the 1024 particles 2048 frames data set, a super-linear speedup is observed. This phenomenon is normally due to differences between the sequential and parallel algorithms or cache effects (the parallel version has better memory characteristics).

As the number of processors increases (the problem size remaining constant), the curve becomes sub-linear due to a decline in parallel efficiency. As the amount of work per processor decreases, the communication is more prevalent (because of less overlap with computation) and this decreases performance (less communication is being overlapped with computation). Again, this graph clearly demonstrates that the algorithm scales very well with an increasing number of particles, and the number of frames has little effect. A maximum speed up of roughly 200 is achieved with 256 processors for 1024 particles and 2048 frames. The speedup would continue to increase for this number of processors if larger data sets (particles) were used.

4.3.2.4 *Simulated displacement ventilation flow*

A CFD simulated indoor air flow field was used to test the trajectory reconstruction accuracy and parallel performance of the new tracking algorithm in the presence of large velocity gradients and non-uniform particle seeding over time. This is done to determine how the algorithm performs when the computational load is unbalanced across processors. The data set includes 1540 particles tracked over 4096 frames to accurately evaluate parallel speedup. The flow domain was a large room (3m x 3m x 6m) with a slot inlet spanning the width of the room and located on the front wall near the ceiling and a slot outlet located on the opposite wall near the floor (Figure 20). The inlet boundary condition was a constant uniform velocity of 4 m/s and the outlet boundary was a standard pressure outlet set to atmospheric conditions. Turbulence was modeled using a Reynolds Averaged Navier-Stokes approach. The resulting steady state flow field solution is shown in Figure 20.

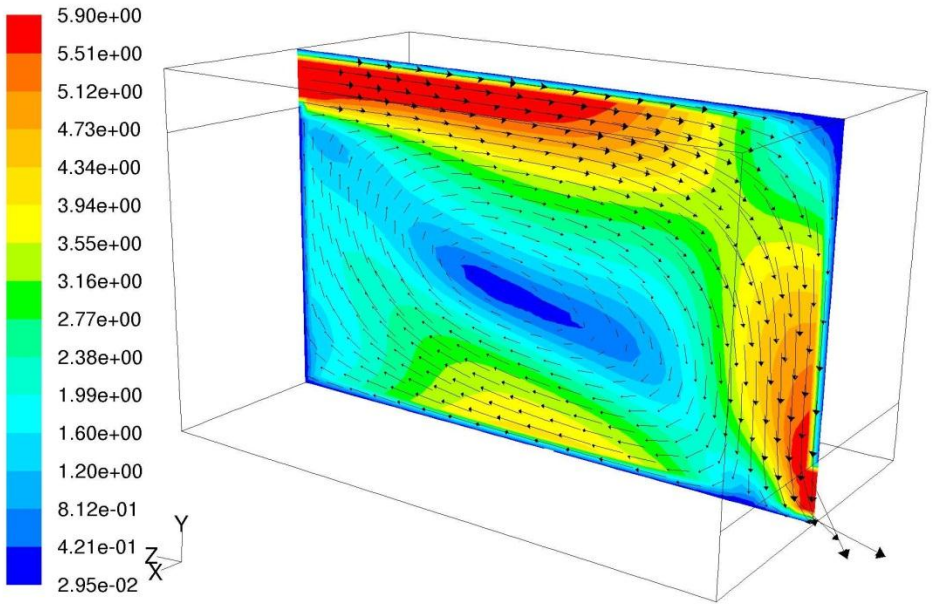

Figure 20: CFD simulated displacement indoor air ventilation velocity vector field and velocity magnitude contours (m/s)

Particle trajectories were simulated using a Lagrangian tracking model, assuming massless particles shown in Figure 21. Particles were injected throughout the domain at two instances in time (frames 0 and 2000) to obtain a non-uniform number of particles per frame as shown in Figure 22. The data was further unbalanced due to the presence of large velocity gradients which caused a portion of the particles to leave the domain more quickly than others, leading to variation in trajectory lengths.

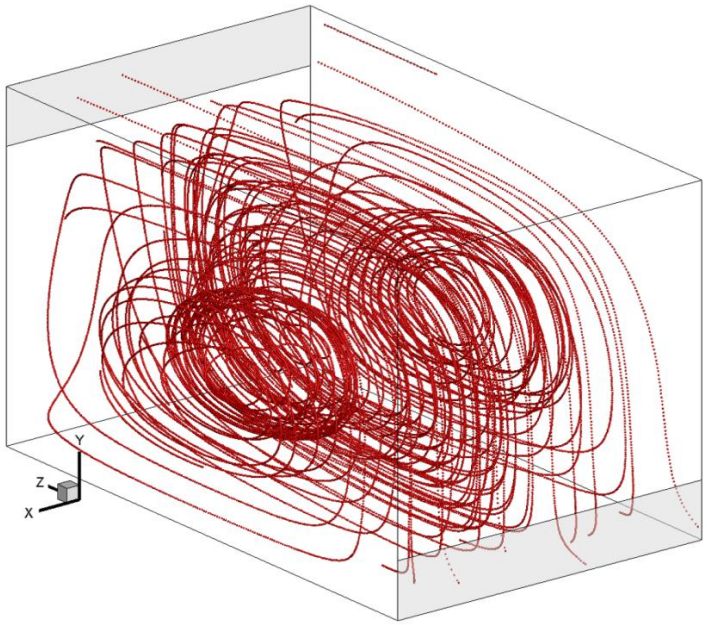

Figure 21: Subset of the CFD simulated particle trajectories in displacement indoor air ventilation

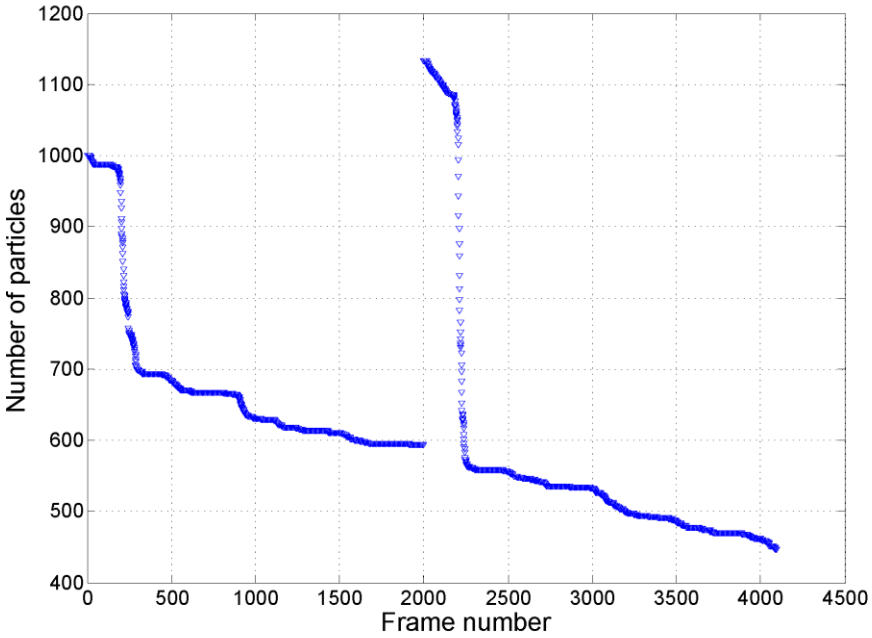

Figure 22: Fluctuation of the number of particles per frame for the CFD simulated data set

The results from the trajectory reconstruction analysis are presented in Table 8. The algorithm worked well and reconstructed the trajectories with nearly 100 percent coverage and correctness. However, the parallel algorithm constructed more trajectories and had a lower average trajectory length than the serial version which indicates that some shorter trajectories did not

completely merge. This is likely due to a locally small particle spacing-displacement ratio near the boundaries of several frame-sets, which resulted in match ambiguity. This however, does not introduce tracking errors as seen in no reduction of the correct tracking ratio values and therefore may be acceptable tradeoff for increased processing speed and scalability of data storage. The frame-set size of eight resulted in the highest percent correct trajectories and was used for the parallel performance analysis.

Table 8: Trajectory reconstruction accuracy results for CFD data set (Workstation)

Frame set size S	$\gamma_{correct}$	$\gamma_{coverage}$	Avg. Trajectory Length	Trajectories Tracked
8	0.999	0.995	1234	2057
64	0.998	0.998	1232	2070
256	0.998	0.999	1228	2079
512	0.998	0.999	1222	2088
4096 (serial)	0.998	0.999	1359	1879

Table 9: Parallel performance results for CFD data set, Frame-set size (*S*) = 8 frames

System	Number of Processors	Run time (s)	speedup	processed frames per second (fps)
Workstation	8	10.18	7.02	402.36
	4	19.54	3.66	209.62
	2	59.01	1.21	69.41
	1 (serial)	71.43	1.00	57.34
Turing Cluster	128	6.98	42.02	586.82
	64	8.68	33.79	471.89
	32	12.41	23.63	330.06
	16	27.42	10.70	149.38
	8	37.88	7.74	108.13
	1 (serial)	293.31	1.00	13.96

The Turing cluster and multi-core workstation were used to evaluate the parallel performance with the non-uniform data set and the results are given in Table 9. As expected, the speedups achieved were lower than those for the uniform data sets in the previous section due to the inherent load imbalance, which caused an increase in processor idle time. On the Turing cluster, the maximum speed up of 42 was achieved with 128 processors for a processed frame rate of 586 fps. The multi-core workstation achieved a maximum speed up of 7 with 8 processors and processed 402 fps. The workstation with 2.4 GHz processor cores and shared memory was four times faster than the Turing cluster with 2.0 GHz cores and distributed memory when processing the sequential code. These results show that real-time processing of the tracking algorithm for a camera frame rate of 100 fps could be possible with either machine.

4.3.3 **Summary of message passing framework development and evaluation**

Parallel processing of the particle tracking algorithm is a key step in achieving a scalable real-time LPT measurement system where large data sets are seamlessly distributed and processed across many computers. Such a scalable system directly addresses the data management issues experienced in LPT experiments and could eventually lead to real-time measurement capabilities for very high speed cameras. A parallel processing framework was developed and evaluation on three simulated data sets proved that it was consistent with the serial version and could efficiently scale to over 500 processors. The algorithm was based on frame decomposition and programmed using object-oriented C++ with the Charm++ extensions for asynchronous message passing between distributed objects. One key aspect of the parallel algorithm was the asynchronous trajectory merge operation that minimizes processor idle time and data transfer between nodes.

Evaluation of the new algorithm with the PIV standard 3D images dataset #352 demonstrated that it was consistent with the optimized serial version in terms of trajectory reconstruction accuracy as quantified by the correct tracking ratio. This data set also validated the new algorithm's ability to handle merging of trajectories of non-uniform length distributed across many processors. In a few instances several local trajectory segments did not merge due to short trajectories formed near the frame-set intersections. However, this may be an acceptable tradeoff for runtime speedup and scalability since major tracking errors were not introduced into the results. Future work can be conducted to optimize the merge function.

The parallel performance evaluation showed that the new algorithm scaled well with an increasing number of particles tracked, while the number of frames processed had very little impact on the scaling performance. This implies that the parallel performance of the algorithm will remain nearly constant if only a few frames are processed at a time, as in real-time data streaming from "smart cameras", or if thousands of frames are processed in batch. If camera resolution is increased to grow the number of resolved tracer particles then a proportional number of processors could be added to maintain parallel performance. The parallel decomposition factor (frame-set size S) did not influence the speedup significantly as long as each processor was assigned more than one frame-set. A significant speedup of up to 200 was obtained with 256 processors for the optimal case of inherently load balanced data (1024

particles and 2048 frames). While for a more realistic data set containing non-uniform trajectories, it was observed that the speedups were still significant: 42 on 128 processors at 586 frames processed per second. While scaling is consistent from multi-core workstations to large clusters, the magnitude of speedup achieved is very dependent on the specific architecture of the system including cache size and processor clock rate.

4.4 Summary of real-time processing framework development

The chapter described the development of two parallel frameworks for real-time processing of LPT data on node based multi-core processors with GPU accelerators and a message passing framework for use on high performance clusters. These two frameworks would eventually be used together in order to allow massive scaling of the LPT system to include many more cameras and higher frame rates.

The results from tests showed that the of the LPT algorithms, the image processing and detection algorithm is the bottleneck of the system when only four cameras are used. However, as the number of cameras is increased to six or more the multi-camera correspondence algorithm becomes the bottleneck. With four cameras and a data set of 300 particles per frame, the compute node streaming framework was able to achieve a processed frame rate of nearly 500 frames per second. With six cameras the added processing load slowed the rate to 270 frames per second. Thus in a real 6 camera system, real-time processing can be achieved on just one node for frame rates less than 270 fps.

If frame rates are increased then the compute cluster message passing framework can be used to distribute the frames by group across a large compute cluster. Results from tests of the compute cluster message passing framework showed that scaling could be achieved for 500 processors. With this information it can be concluded that the objective of demonstrating the ability to reach real-time processing of LPT data scalable to hundreds of processors for high frame rate cameras can be achieved.

5 Prototype design and sensitivity analysis

5.1 Objective and approach

The goal of this chapter is to overview the important hardware components of the real-time Lagrangian particle tracking (LPT) system and complete a sensitivity analysis based on derived measurement uncertainties to help guide experimental design.

Section 5.2 will cover the LPT system hardware and key selection decisions for real-time tracking. Procedures for calculating the uncertainty in reconstructed 3D particle positions, velocity and acceleration will be derived in section 5.3. Finally in section 5.4 a sensitivity analysis will be conducted to characterize the impact of camera parameter uncertainty and camera placement on the particle position measurement uncertainty of the system.

5.2 System hardware

The real-time LPT system is composed of four major hardware components 1) Imaging system including cameras and lenses, 2) illumination system and seed particle generator, and 3) computational resources. The following section will cover the selection of each of these components for the real-time LPT prototype used in this research. At the end of this section the calibration procedures will be described.

5.2.1 Imaging system

The imaging system is composed of cameras and lenses, each of which introduces important selection decisions when designing the real-time LPT system. The important factors when selecting cameras can be broken down in the following 1) frame rate, 2) sensor resolution, 3) synchronization and controllability, and 4) image processing capabilities and data transfer bandwidth.

The sensor resolution and frame rate directly impact the type of particle flow that can be characterized with the LPT system. For example, the frame rate directly relates to the maximum particle velocity which can be tracked based on maintaining a minimum particle spacing-displacement ratio.

As noted by Malik, a key criterion to predict tracking difficulty for a given particle tracking system and flow field is the particle spacing-displacement ratio p (Malik et al., 1993).

$$p = \frac{\Delta s}{u \Delta t} \tag{5-1}$$

This ratio is defined by the mean spacing between particles (Δs), and the mean particle displacement from frame to frame ($u\Delta t$). If $p >> 1$ then tracking can be completed with high accuracy and relative ease (Malik et al., 1993).

If 500 particles are to be tracked in a 1.0 m^3 volume then the maximum measureable particle speed, from a tracking efficiency standpoint, would vary with the LPT system's camera frame rate as shown in Figure 23. As the particle spacing displacement ratio increases, tracking efficiency increases along with the length of the tracked trajectories. If the ratio is equal to 1 then tracking is virtually impossible. Therefore, the frame rate of the cameras should be selected based on the ability to over-sample a particle's trajectory at the maximum expected velocity of the flow field, desired seed particle density, and length of desired trajectories.

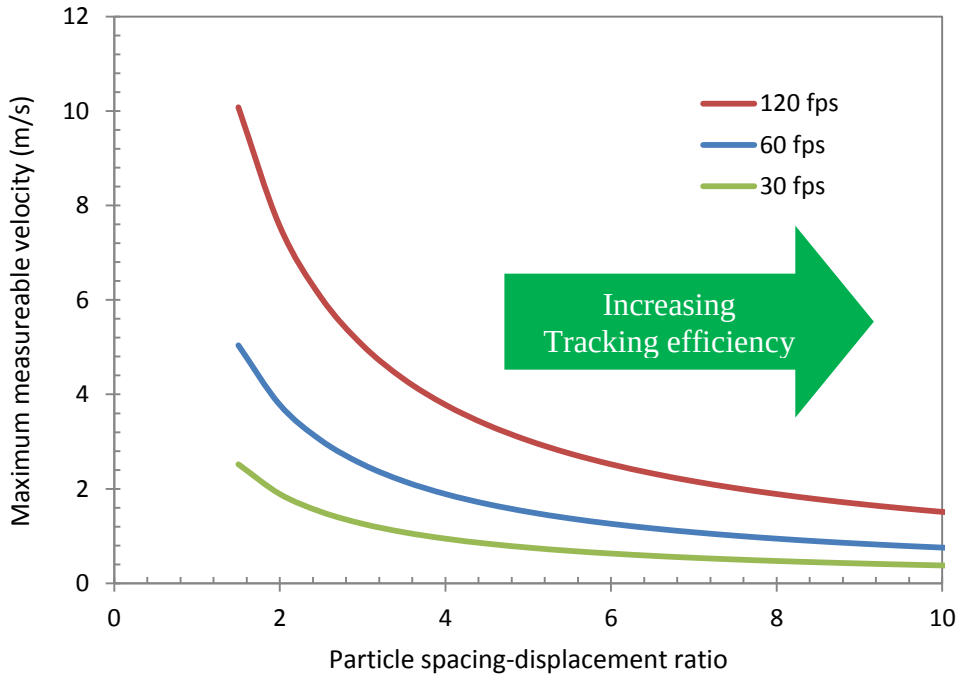

Figure 23: Frame rate limitation on maximum particle speed due to tracking efficiency

The sensor resolution is a major factor which impacts the density of particles that can be resolved along with the uncertainty of the particle centroid location. However, the importance of spatial resolution may be less than that of frame rate when the goal is to track fewer particles over longer trajectories. Lower resolution sensors also cost less, which allows more cameras to be purchased for the same budget. Maas (1992) reported using four low-resolution (720 x 574 pixel) CCD cameras to detect up to 1500 particles in a small aquarium resulting in 800 particles being tracked (H.G. Maas, 1992). This level of particle density is acceptable for this research; therefore, priority was given to camera frame rate, multi-camera synchronization, data compression and controllability.

Real-time operation is one of the objectives of this research, therefore the cameras needed to utilize embedded Field Programmable Gate Arrays (FPGA) to complete on camera image processing and reduce overall data transfer rates. In addition these cameras must be able to synchronize among groups up to hundreds of cameras. Such cameras have been made commercially available for the movie and video game industry in the area of motion capture. Motion capture cameras have been designed with FPGAs to detect objects through image segmentation and centroid localization, and then are able to send the object coordinates directly to the host computer.

For the prototype real-time LPT system, six low cost motion capture cameras (NaturalPoint Inc. Optitrack Model V120:SLIM) were purchased based on the availability of a C++ control interface, hard wire synchronization, flexible lens mounts, and built in FPGA for particle detection and image compression (Figure 24). The cameras chosen contained VGA resolution (640 x 480 pixels) monochrome sensors which were housed in aluminum housings with standard CS lens mounts with C mount adaptors (NaturalPoint, 2012). The six cameras form 15 unique two-camera combinations, and 15 unique four-camera combinations which were enough to test the viability of the real-time multi-camera correspondence approach described in Chapter 3. The variable zoom lenses had a focal length range from 2.8 - 8.0 mm and aperture of F/1.2 with a CS mount (Tamron Model No. M13VM288IR). The lenses were IR-corrected and designed for 3 megapixel sensor, which meant that the image sensors of this system would not be limited by the resolving power of the lenses (Tamron, 2012).

Table 10: Camera system specifications

Parameter	Specification
Frame rate	30, 60, 120 fps
Resolution	640 x 480
Sensor type	CMOS monochrome
Bit depth	8
Pixel size	6 x 6 µm
Imager size	4.5 x 2.88 mm
Shutter type	Global
Shutter speed	1 ms - 20 µs
Image processing abilities	MJPEG compression Image segmentation Object centroid detection
Control	C++ camera SDK
Synchronization	Synch breakout cable
Communication	USB 2.0

Figure 24: Motion capture camera system used for real-time LPT prototype development

5.2.2 Illumination and seed particle generator

The decisions on types of illumination systems and seed particle generators are closely linked. In many particle tracking velocimetry and particle image velocimetry studies the illumination source is a high power pulse laser and the particles are small flow tracers on the order of 10 microns in diameter. For the LPT system in this research the goal was to observe flows on larger scales and to create a system that is not cost prohibitive. Therefore low cost LED light panels and a commercial generator for neutrally buoyant helium filled soap bubbles were selected. Each LED light panel (FancierStudio, 2012), contained 500 individual light emitting

diodes and had a total power consumption of 50 watts. The 1-4mm helium filled soap bubbles were generated with a Sage Action Inc. Model 5 bubble generator (SAI, 2012). These bubbles are easily visible using the LED system at a distance of 1.5 m from the camera. This combination provided a safe, affordable, and scalable solution to move to larger room scale flows in the future.

5.2.3 **Computational resources**

A multi-core workstation was purchased from Colfax International (http://www.colfax-intl.com) to provide the computational power needed to achieve real-time Lagrangian particle tracking and visualization with the six camera system. This workstation contained two quad-core Intel Xeon 2.4GHz CPUs, 32 GB of random access memory (RAM), and four graphics processing units (GPU). Three of the graphics cards were NVIDIA Tesla C1060 GPU compute accelerators strictly for data processing and one card was an NVIDIA Quadro FX3700 for 3D visualization. The two quad core CPUs were hyper threaded, to provided 16 virtual processors for multi-threading. The six cameras were connected to the computer using a 7 port USB 2.0 hub.

5.2.4 **System calibration**

The cameras were calibrated before each experiment using an open source camera calibration routine based on the work by Zhang (2000) and MATLAB® toolbox developed by Bouguet (Bouguet, 2010; Z. Zhang, 2000). The calibration algorithm is based finding the optimal camera parameters (focal length, principle point, and distortion coefficients) that minimize the reprojection error between known image points and their specified 3D world coordinates. This method has been derived in great detail in the publication by Zhang (2000) and will not be covered here for brevity. The calibration procedure used in this research consisted of the following. First, images of a planar calibration board containing 44 black circles (Figure 25) are recorded in at least 50 orientations for each camera. Then the circles are detected their centroids are computed. Finally the list of centroid coordinates and 3D object coordinates (relative to the board plane) were processed with the method by Zhang to determine the optimal camera parameters based on all 50 images. The reprojection residuals from the optimal camera parameters were on the order of 0.1 pixels for each camera.

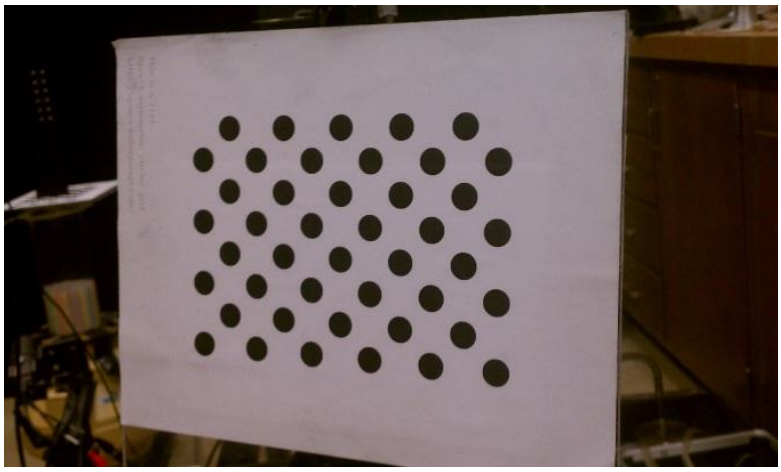

Figure 25: Planar calibration board containing 44 detectable points

The extrinsic parameters are determined based on the known correspondence between 2D image coordinates and 3D world coordinates of the calibration board. These extrinsic parameters define the pose of the camera relative to the object as defined by a rotation matrix and translation vector. These parameters are determined through the use of an OpenCV function called *solvePnP*, which estimates an object's pose from a set of detected image points and known object points for the same planar calibration object. The extrinsic parameters of all six cameras can be determined at once when they share a common view of the static calibration board. Once imaged and detected in each camera, the rotation and translation vectors for each camera are determined by minimizing the reprojection error, which is the Euclidian norm of the difference between the actual imaged point locations and the reprojected image points from known object coordinates and intrinsic parameters of the camera. The resulting intrinsic and extrinsic parameters are used to calculate the Fundamental matrices for each unique two-camera combination through equations 3-12 and 3-15.

5.3 Derivation of measurement uncertainties

The objective of this section is to derive methods to identify the combined standard uncertainty of 3D particle position, velocity and acceleration measurements. These uncertainty calculation methods will be used to conduct a sensitivity analysis in the following section and ultimately determine the combined standard uncertainties of the measurements made by the prototype real-time LPT system.

5.3.1 **3D position combined standard uncertainty**

3D positions of the particles are the fundamental output of the particle tracking system. This sub section will derive methods for calculating the combined standard uncertainty of the 3D position in terms of a single value with units of length. This value defines the radius of a sphere of uncertainty around the measured particle position. The combined standard uncertainty of the 3D world position measurement is based on uncertainty propagation from three factors 1) detected particle centroid coordinates, 2) camera intrinsic parameters including distortion coefficients, principle point and focal length, and 3) the extrinsic parameters defining the cameras translation and rotation from the world coordinate system.

In this analysis, the particle's world coordinate $\mathbf{x}_w$ is the measurand of interest and can be defined as a function of the particle's camera coordinates along with the rotation and translation vectors of camera involved in the 3D reconstruction, as given in equation 5-2.

$$\begin{aligned}\mathbf{x}_w &= f\left(p_1, p_2, \ldots p_n\right) \\ &= f\left(\left[\mathbf{x}_{c_1}, \mathbf{R}_1, \mathbf{t}_1\right], \left[\mathbf{x}_{c_2}, \mathbf{R}_2, \mathbf{t}_2\right], \ldots \left[\mathbf{x}_{c_n}, \mathbf{R}_n, \mathbf{t}_n\right]\right)\end{aligned} \tag{5-2}$$

Here *f* is the 3D reconstruction function described in Chapter 3. The uncertainty associated with the world coordinates follows the combined standard uncertainty equation (ISO, 2009; Kirkup & Frenkel, 2006), where all parameters *p* in the measurement equation are assumed to be independent from one another.

$$u^2\left(\mathbf{x}_w\right) = \sum_{i=1}^{N}\left(\frac{\partial f}{\partial p_i}\right)^2 u^2\left(p_i\right) \tag{5-3}$$

The particle's coordinates in each camera, $\mathbf{x}_c$ are related to the world coordinates through the following equation.

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \mathbf{R}\begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} + \mathbf{t} \tag{5-4}$$

Its homogenous coordinates (x' and y') are defined by normalizing the x and y components by the z component.

$$x' = x_c / z_c$$
$$y' = y_c / z_c \tag{5-5}$$

The first factor introducing uncertainty into the 3D reconstruction process is the uncertainty in the lens distortion coefficients k_1, k_2, p_1 and p_2 as determined through calibration. The distorted image coordinate of a particle $\mathbf{x}_{pd}$ and the undistorted coordinate $\mathbf{x}_p$ are related through a nonlinear function *g* and the distortion coefficients.

$$\mathbf{x}_p = g\left(\mathbf{x}_{pd}, k_1, k_2, p_1, p_2\right) \tag{5-6}$$

The combined standard uncertainty in the undistorted particle image coordinate is then a combination of the standard uncertainty in the original distorted coordinate $\mathbf{x}_{pd}$ and the standard uncertainty in each distortion coefficient as follows.

$$u^2\left(\mathbf{x}_p\right) = \left(\frac{\partial g}{\partial \mathbf{x}_{pd}}\right)^2 u^2\left(\mathbf{x}_{pd}\right) + \left(\frac{\partial g}{\partial k_1}\right)^2 u^2\left(k_1\right) + \left(\frac{\partial g}{\partial k_2}\right)^2 u^2\left(k_2\right) + \left(\frac{\partial g}{\partial p_1}\right)^2 u^2\left(p_1\right) + \left(\frac{\partial g}{\partial p_2}\right)^2 u^2\left(p_2\right) \tag{5-7}$$

In practice, the standard uncertainty of the distortion coefficients can be ascertained by repeating the calibration process with different sets of calibration images and finding the standard deviation in the resulting optimal camera parameters. The standard uncertainty in the distorted image coordinate $\mathbf{x}_{pd}$ is directly a result of the particle centroid finding algorithm and random image noise; this term is dependent on experimental conditions and will vary during the experiment. The partial derivatives of the undistortion function *g* are not easily determined, since it represents an iterative non-linear optimization algorithm. Therefore, these sensitivities are determined by perturbing the distortion coefficients by their standard uncertainty and solving for the undistorted coordinates. The change in the undistorted coordinates is then recorded as the uncertainty contribution from that coefficient in equation 5-7.

The homogenous coordinates are a function of the image coordinates through the following relation.

$$x' = \frac{x_p - c_x}{f_x}$$
$$y' = \frac{y_p - c_y}{f_y} \tag{5-8}$$

The combined standard uncertainties for the homogenous coordinates are then easily determined by taking the partial derivatives of 5-8 to obtain:

$$u_c^2(x') = \left(\frac{1}{f_x}\right)^2 u^2(x_p) + \left(\frac{-1}{f_x}\right)^2 u^2(c_x) + \left(\frac{c_x - x_p}{f_x^2}\right)^2 u^2(f_x)$$
$$u_c^2(y') = \left(\frac{1}{f_y}\right)^2 u^2(y_p) + \left(\frac{-1}{f_y}\right)^2 u^2(c_y) + \left(\frac{c_y - y_p}{f_y^2}\right)^2 u^2(f_y) \tag{5-9}$$

The combined standard uncertainty of the 3D position can then be written as a sum of squares of all the parameter uncertainties multiplied by the squared sensitive of the reconstruction function *f* for *N* cameras used in the reconstruction.

$$u_c^2(\mathbf{x}_w) = \sum_{c=1}^{N} \left[\begin{array}{l} \left(\frac{\partial f}{\partial x_c'}\right)^2 u^2(x_c') + \left(\frac{\partial f}{\partial y_c'}\right)^2 u^2(y_c') \\ + \left(\frac{\partial f}{\partial \mathbf{R}_c}\right)^2 u^2(\mathbf{R}_c) + \left(\frac{\partial f}{\partial \mathbf{t}_c}\right)^2 u^2(\mathbf{t}_c) \end{array} \right] \tag{5-10}$$

The sensitivity of the reconstruction function *f*, which is a linear least squares problem, depends on both the conditioning of the matrix **A** and the magnitude of the residual **r**. Where the least squares problem is defined as:

$$\mathbf{A}\mathbf{x}_w \approx \mathbf{b}$$
$$\mathbf{r} = \mathbf{A}\mathbf{x}_w - \mathbf{b} \tag{5-11}$$

For 3D reconstruction this linear system of equations can be formed with a minimum of two cameras and takes the form shown in below.

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_i \\ \vdots \\ \mathbf{A}_N \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_i \\ \vdots \\ \mathbf{b}_N \end{bmatrix} \tag{5-12}$$

Where for the matched imaged point in camera *i* the sub matrices $\mathbf{A}_i$ and $\mathbf{b}_i$ with respect to the normalized image point (x_i' , y_i') are

$$\mathbf{A}_i = \begin{bmatrix} x_i' r_{31} - r_{11} & x_i' r_{32} - r_{12} & x_i' r_{33} - r_{13} \\ y_i' r_{31} - r_{21} & y_i' r_{32} - r_{22} & y_i' r_{33} - r_{23} \end{bmatrix}; \quad \mathbf{b}_i = \begin{bmatrix} t_x - x_i' t_z \\ t_y - y_i' t_z \end{bmatrix} \tag{5-13}$$

The sensitivity of the solution vector $\mathbf{x}_w$ as a function of a perturbation in the vector $\mathbf{b}$, called $\Delta\mathbf{b}$, has been derived by Heath (2002) to be a function of the condition number of matrix $\mathbf{A}$ and the angle θ between $\mathbf{b}$ and $\mathbf{A}\mathbf{x}_w$ (Heath, 2002).

$$\frac{\|\Delta\mathbf{x}_w\|_2}{\|\mathbf{x}_w\|_2} \le \text{cond}(\mathbf{A}) \frac{1}{\cos(\theta)} \frac{\|\Delta\mathbf{b}\|_2}{\|\mathbf{b}\|_2} \tag{5-14}$$

The angle θ can be derived from the L2 norms of $\mathbf{A}\mathbf{x}_w$ and $\mathbf{b}$ as follows.

$$\cos(\theta) = \frac{\|\mathbf{A}\mathbf{x}_w\|_2}{\|\mathbf{b}\|_2} \tag{5-15}$$

Sensitivity of the solution $\mathbf{x}_w$ as a function of the perturbation in the matrix $\mathbf{A}$, called $\mathbf{E}$ was derived by Heath (2002) to be:

$$\frac{\|\Delta\mathbf{x}_w\|_2}{\|\mathbf{x}_w\|_2} \le \left(\text{cond}(\mathbf{A})^2 \tan(\theta) + \text{cond}(\mathbf{A})\right) \frac{\|\mathbf{E}\|_2}{\|\mathbf{A}\|_2} \tag{5-16}$$

Singular Value Decomposition (SVD) of the matrix $\mathbf{A}$ can be used to determine the L2 norm and condition number of $\mathbf{A}$ which are required to compute the sensitivities of $\mathbf{x}_w$. The SVD of $\mathbf{A}$ is:

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{5-17}$$

Where $\mathbf{\Sigma}$ is a diagonal matrix containing the positive singular values labeled σ_1 through σ_N.

$$\mathbf{\Sigma} = diag(\sigma_1, \sigma_2 ... \sigma_N)$$
$$\sigma_i \geq 0 \tag{5-18}$$

The L2 norm of **A** is equal to the largest singular value, while the condition number of **A** is equal to the ratio of the largest singular value divided by the smallest. The condition number defines the bounds of the ratio of relative change in the solution **x** to a change in the matrix **A** (Heath, 2002). Thus the 3D reconstruction problem will be highly sensitive to uncertainties in any of the input parameters if the resulting matrix **A** is ill-conditioned, which corresponds to a condition number much greater than one ($\text{cond}(\mathbf{A}) >> 1$).

$$\|\mathbf{A}\|_2 = \sigma_{\max} \tag{5-19}$$

$$\text{cond}(\mathbf{A}) = \frac{\sigma_{\max}}{\sigma_{\min}} \tag{5-20}$$

Now the sensitivities of the 3D reconstruction function *f* can be calculated for each parameter in equation 5-10, as below where the terms for each input parameter are generalized by Δp_i .

$$\frac{\partial f}{\partial p_i} \approx \frac{\|\Delta \mathbf{x}_w\|_2}{\Delta p_i} \tag{5-21}$$

The 3D position combined standard uncertainty can then be calculated for each set of match particle images as follows:

1. Calibrate cameras and obtain uncertainties for internal and external parameters
2. Specify or measure the particle image centroid uncertainty $u(\mathbf{x}_{pd})$
3. For a specified group of *N* cameras, select a matched group of corresponding particle image coordinates
4. Solve equation 5-7 for the uncertainties in the *N* sets of undistorted pixel coordinates
5. Solve equation 5-9 for the uncertainties in the *N* sets of homogenous image coordinates
6. Build the matrix **A** based on equation 5-13 and compute its Singular Value Decomposition

7. Calculate
    a. $\mathbf{x}_w$ based on the SVD of **A**
    b. L2 norm of **A**, $\mathbf{A}\mathbf{x}_w$, and **b**, equation 5-19
    c. condition number of **A**, equation 5-20
8. Calculate least squares solution sensitivities for x', y', **R** and **t** of each camera using equations 5-14, 5-16, and 5-21
9. Substitute the sensitivities into equation 5-10 and compute the 3D position combined standard uncertainty $u(\mathbf{x}_w)$
10. Repeat for all 3D points and store the uncertainty values in an array
11. Calculate the mean and variance for the array of uncertainty values

5.3.2 **Velocity and acceleration combined standard uncertainties**

The combined standard uncertainties for particle velocity and acceleration can be derived from the 3D position combined standard uncertainties and finite difference approximations shown in equations 5-22 and 5-24 respectively. For velocity the second order central difference uses a two point stencil and therefore uncertainty is propagated from the *n*+1 and *n*-1 particle positions as follows:

$$\mathbf{v} \approx \frac{\mathbf{x}^{n+1} - \mathbf{x}^{n-1}}{2\Delta t} + O(\Delta t^2) \tag{5-22}$$

$$u_c^2(\mathbf{v}) = \left(\frac{1}{2\Delta t}\right)^2 u^2\left(\mathbf{x}^{n+1}\right) + \left(\frac{-1}{2\Delta t}\right)^2 u^2\left(\mathbf{x}^{n-1}\right) + \left(-\frac{\mathbf{x}^{n+1} - \mathbf{x}^{n-1}}{2\Delta t^2}\right)^2 u^2(\Delta t) \tag{5-23}$$

The acceleration approximation utilizes a three point central difference stencil and results in an uncertainty equation as follows:

$$\mathbf{a} \approx \frac{\mathbf{x}^{n+1} - 2\mathbf{x}^{n} + \mathbf{x}^{n-1}}{\Delta t^2} + O(\Delta t^2) \tag{5-24}$$

$$u_c^2(\mathbf{a}) = \left(\frac{1}{\Delta t^2}\right)^2 u^2\left(\mathbf{x}^{n+1}\right) + \left(-\frac{2}{\Delta t^2}\right)^2 u^2\left(\mathbf{x}^{n}\right) + \left(\frac{1}{\Delta t^2}\right)^2 u^2\left(\mathbf{x}^{n-1}\right) + \left(-2\frac{\mathbf{x}^{n+1} - 2\mathbf{x}^{n} + \mathbf{x}^{n-1}}{\Delta t^3}\right)^2 u^2(\Delta t) \quad (5\text{-}25)$$

These uncertainty equations take into account both the uncertainties in particle positions, but also the uncertainty in the time step. In practice the time step uncertainty is on the order of 10^{-6} s and this term can be neglected.

5.4 Sensitivity analysis

The objective of this section is to evaluate the impact of camera calibration, camera placement, and the selection of camera correspondence groups for 3D reconstruction on the 3D position combined standard uncertainty and 3D reconstruction error. Ultimately the results of this analysis will help guide the camera placement for the experimental analysis in Chapter 6. For the following analyses the 3D reconstruction error is defined as:

$$E_{3D} = \left\|\mathbf{x} - \hat{\mathbf{x}}\right\|_2 = \sqrt{\left(x - \hat{x}\right)^2 + \left(y - \hat{y}\right)^2 + \left(z - \hat{z}\right)^2} \quad (5\text{-}26)$$

Where $\mathbf{x}$ is the known "true" position of a particle and $\hat{\mathbf{x}}$ is the estimated 3D position resulting from the 3D reconstruction process. This error can only be calculated when utilizing synthetic datasets in the reconstruction process, where the 3D positions of the particles are known in advance.

5.4.1 Impact of camera calibration parameter uncertainty on 3D reconstruction

In this section the goal is to answer the following questions:

1) What are the potential values for 3D position uncertainty?
2) Which parameter is the most influential on the 3D position uncertainty?

To answer these questions, the camera calibration parameter uncertainties were estimated based on the estimated maximum range of each parameter. Uncertainties are estimated to be 1/1000 of the estimated maximum range of each parameter. This provides a reasonable set of values

which can be used to understand the relative importance of each parameter during an experiment. The table below shows the estimated uncertainty values for each parameter based on 1/1000 of the parameters scale. For example the estimated 0.1% uncertainty for the distorted particle image position $\mathbf{x}_{pd}$ based on an image sensor with 640 x 480 pixels is 0.64 pixels.

Table 11: Estimated 0.1% uncertainties for calibration parameters based on maximum range of each parameter

		0.1% uncertainties		
	Max range	x	y	z
u(c)	320 x 240 pixels	0.32 pix	0.24 pix	-
u(f)	8 mm, 6μm/pixel	1.33 pix	1.33 pix	-
u(X)	640 pixels	0.64 pix	0.64 pix	-
u(T)	1120 mm	0 mm	0 mm	1.12 mm
u(R)	1.57	1.57E-03	1.57E-03	1.57E-03

To determine the impact of camera calibration parameter uncertainty on the 3D position standard combined uncertainty, a 16 x 16 x 16 virtual 3D point grid was generated with 10 mm spacing between points. Two virtual cameras were placed at 45 degrees relative to the front face of the 300 mm x 300 mm x 300 mm cubic domain (Figure 26). The resulting average condition number of **A** for the two camera combination was 2.64 ± 0.196, as determined from taking the mean value of condition number for all points in the grid.

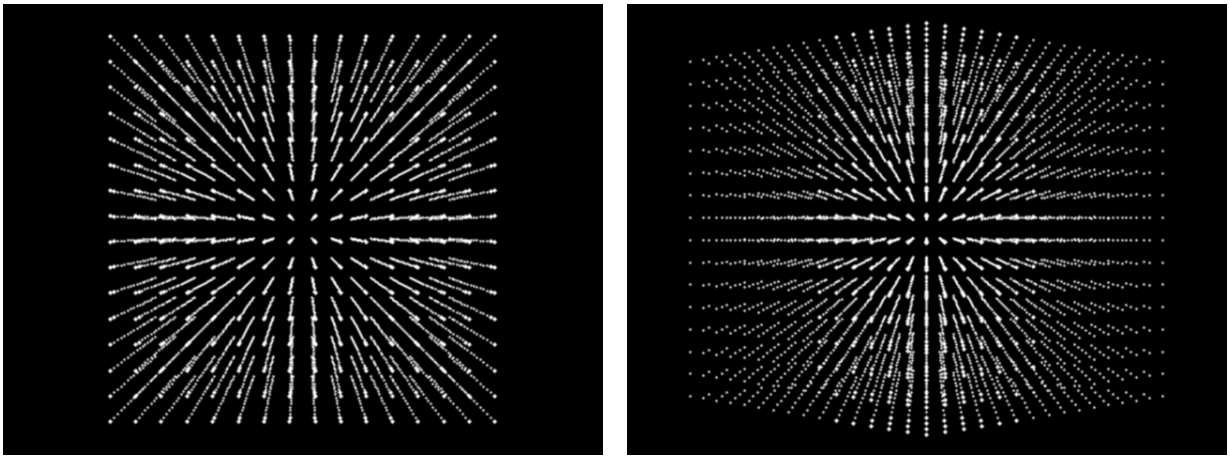

Figure 26: virtual 3D point grid viewed at 0 and 45 degrees from the XY plane

The 3D position combined standard uncertainty was calculated for all points in the synthetic grid and an average was calculated by summing over all points and dividing by the total number

of points. Average combined standard uncertainties were calculated for six cases, based on isolating the influence of each of the five parameters in Table 11 with an additional calculation taking the combination of all parameter uncertainties into account. The resulting average 3D position combined standard uncertainties normalized by the absolute domain length of 300 mm and are shown in Figure 27. The centroid position is the most influential uncertainty factor. When this value is 0.64 pixels (0.1%) of the imager's horizontal resolution, the resulting 3D position combined standard uncertainty reaches nearly 2.7 mm (0.9%) in world coordinates. The next most important parameter was the principle point of the imager which produced nearly 0.4% position uncertainty alone. When combined, the 0.1% parameter uncertainties created 1.0% uncertainty in the particle's position corresponding to a 3 mm sphere of uncertainty. Therefore, focus should be placed on minimizing the uncertainty for the centroid localization through reducing image noise and improving the robustness of the particle detection and centroid localization algorithm.

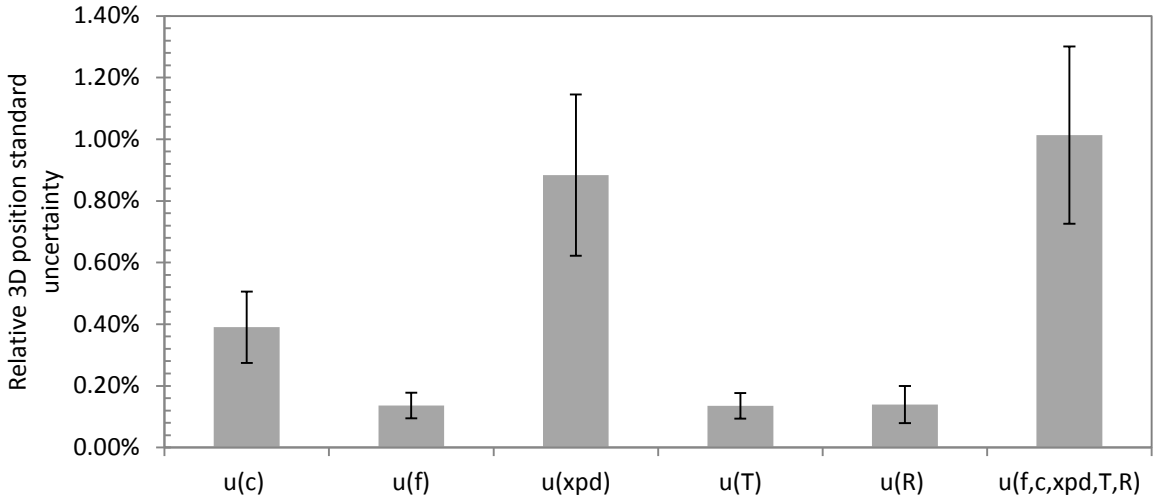

Figure 27: Relative 3D position standard uncertainties corresponding to 0.1% uncertainty in camera parameters and particle centroid location

5.4.2 **Impact of camera placement on 3D position combined standard uncertainty**

The objective of this section is to answer the following two questions in order to facilitate camera placement to minimize uncertainty in the experimental setup:

1. How does the angle between two cameras placed at equal distances from the center of an observation volume impact the 3D position combined standard uncertainty and 3D reconstruction error in the presence of image noise?
2. How does the selection of lens focal length and working distance impact the 3D position combined standard uncertainty and 3D reconstruction error in presence of image noise?

The approach to answering the first question was to calculate the 3D position combined standard uncertainties for two cameras with an increasing angle of separation around a fixed virtual 3D point grid. The 3D position combined standard uncertainty and 3D reconstruction error were compared with the condition number of the **A** matrix.

The approach to answering the second question was to vary the lens focal length and location of the cameras to maintain the same observation volume and observe the impact to uncertainty and error. For example a camera with a wide angle lens, f = 3 mm, can be placed at a distance of 0.33 m to observe the same volume as a camera with a narrower field of view, f = 8 mm, at a distance of 1.3 m. Which is better to reduce measurement uncertainty?

In each analysis errors in particle image centroid localization were simulated by adding in a specified amount of random noise to the centroid *x* and *y* pixel positions. This type of error would occur due to particle image overlap and is expected during an experiment with high particle seed densities. Therefore, it is important to understand how this type of error propagates through to the reconstructed 3D position.

5.4.2.1 ***Angle between cameras***

The angle between cameras is a major factor that impacts the sensitivity and conditioning of the 3D reconstruction problem. Two cameras placed very close together would cause the reconstruction problem to be ill-conditioned and sensitive to image noise and calibration parameter uncertainties. This analysis will characterize the sensitivity over the likely range of angles between cameras placed equal distances from an observation volume. Cameras were

placed at five degree intervals in the range from 0 and 180 degrees around the center of the 3D virtual point grid as shown in Figure 28. The distances from the cameras to the center of the grid were fixed at 1.3 m and a lens focal length of 8 mm was used. The estimated camera parameter uncertainties were based on three percentages 1.0%, 0.1% and 0.01% of the maximum range for each parameter.

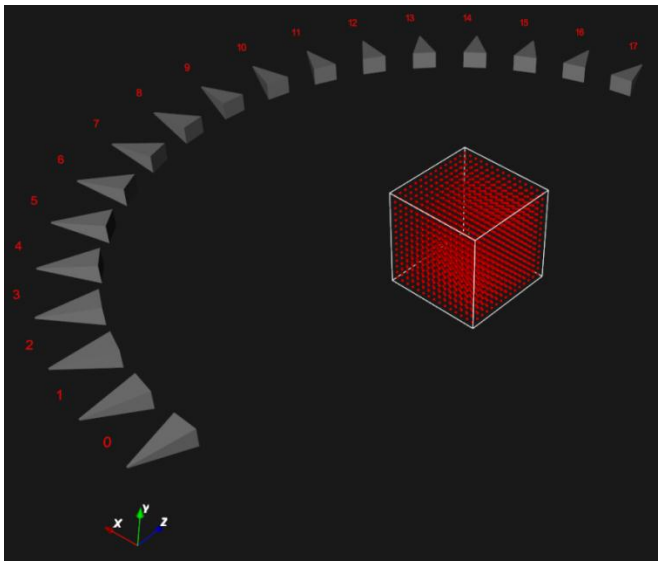

Figure 28: Visualization of camera placement for analysis of angle of separation on 3D reconstruction

The resulting averaged condition numbers of each two-camera arrangement for all points in the grid are shown in Figure 29. The zone of ill-conditioning can be seen when cameras are placed less than 20 degrees apart around the perimeter of the volume of interest and when nearly opposite each other at 160-180 degrees. The corresponding 3D position relative combined standard uncertainty, based on the length scale of 300 mm, varied with separation angle as shown in Figure 30 for the three levels of camera parameter uncertainty. The figure displays the impact of the camera parameter uncertainty level, which narrows the range of acceptable camera placement if increased.

Finally the impacts of centroid detection errors were analyzed at random noise levels of 0.5, 1.0, and 2.0 pixels and were plotted with respect to camera separation angle. Figure 31 highlights the sensitivity issue arising from using only two cameras placed in a similar location for the 3D reconstruction process. When cameras are placed at a right angle to each other the relative 3D position error resulting from a 2.0 pixel centroid error in both cameras is less than 1.0%.

However if the two cameras are placed 5 degrees apart, the resulting error would be magnified by a factor of four. Overall the condition number is a direct indicator of the sensitivity of the reconstruction process to position uncertainty and error. Using more the two cameras in the reconstruction process can greatly reduce the sensitivity of the system as will be shown in section 5.3.3.

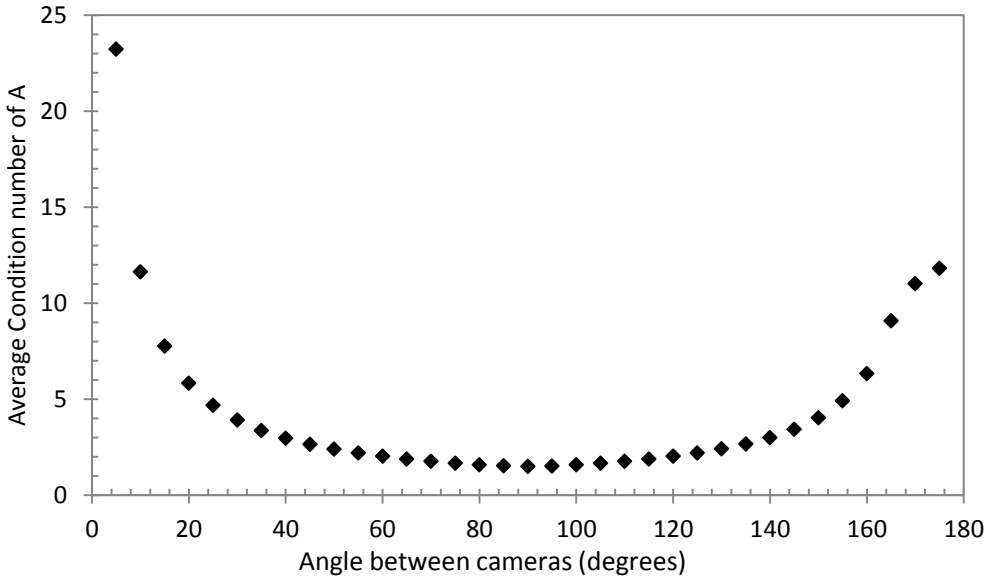

Figure 29: Condition number of 3D reconstruction matrix A as a function of angle between two cameras

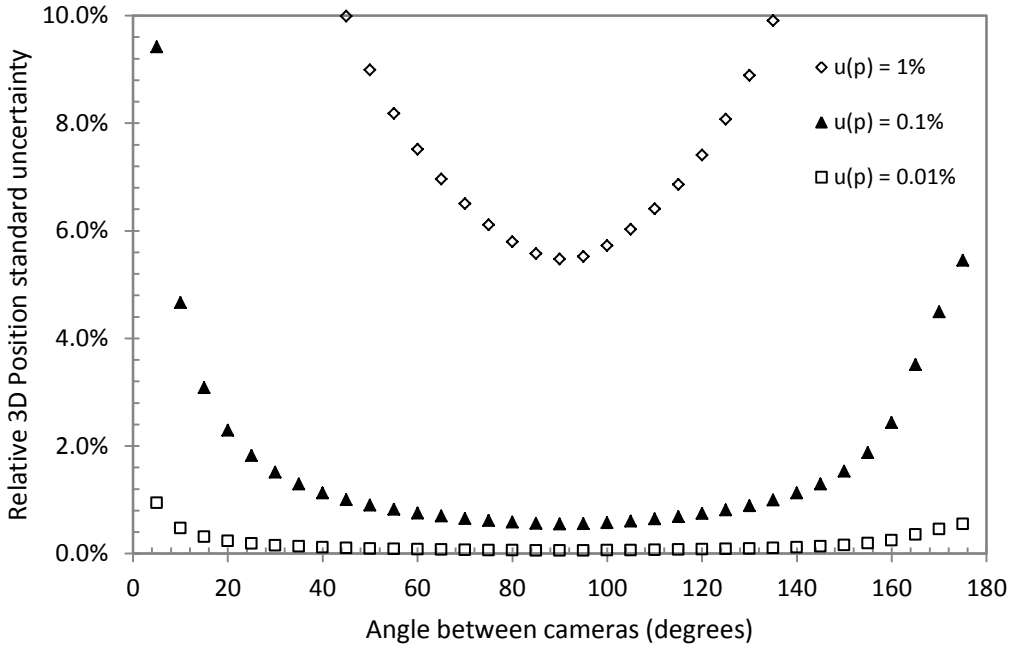

Figure 30: Relative 3D standard uncertainty variation with of angle between two cameras

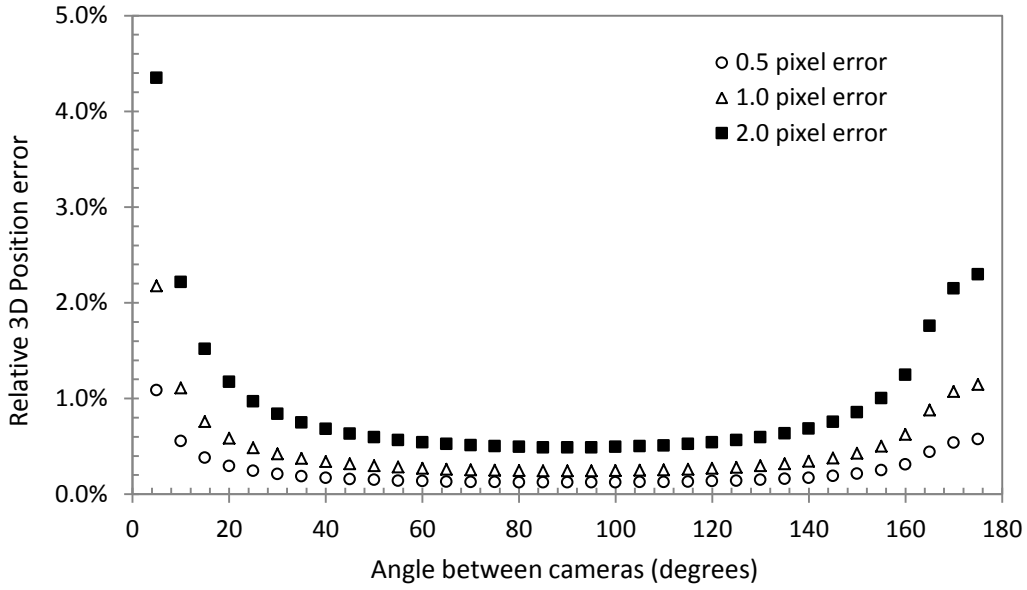

Figure 31: Relative 3D position error resulting from centroid location errors and variation with angle between two cameras

5.4.2.2 *Focal length and focus distance*

To determine the impact of lens selection and camera working distance on 3D position uncertainty and error, six equivalent camera setups were evaluated. Six focal lengths ranging from 8mm to 3mm and corresponding working distances required to maintain a common observed volume were selected and are given in Table 12. These values correspond to the zoom range and senor size of the lenses and cameras selected for this research. Two virtual cameras were used for the 3D reconstruction, with both cameras were placed equal distances from the center of the grid and separated by an angle of 45 degrees. A two-pixel centroid position error was introduced to test the sensitivity of resulting 3D position error.

Table 12: Equivalent camera setups based on lens focal and focus distance settings

f (mm)	max distance (mm)	max width (mm)	max height (mm)	focus distance (mm)	Depth of Field (mm)
3.00	500.00	640.00	480.00	333.83	249.44
4.00	666.67	640.00	480.00	485.44	284.99
5.00	833.33	640.00	480.00	641.68	311.63
6.00	1000.00	640.00	480.00	800.71	332.34
7.00	1166.67	640.00	480.00	961.54	348.91
8.00	1333.33	640.00	480.00	1123.60	362.46

The results from this analysis are shown in Figure 32. The six setups show no statistically significant difference in relative 3D position uncertainty. The same is observed for position error resulting from random centroid detection errors up to 2.0 pixels. Therefore, it can be

concluded that focal length and working distance do not theoretically impact the 3D reconstruction process. However, this assumes lens distortion model fits the real lens distortion equally well for large and small focal length lenses.

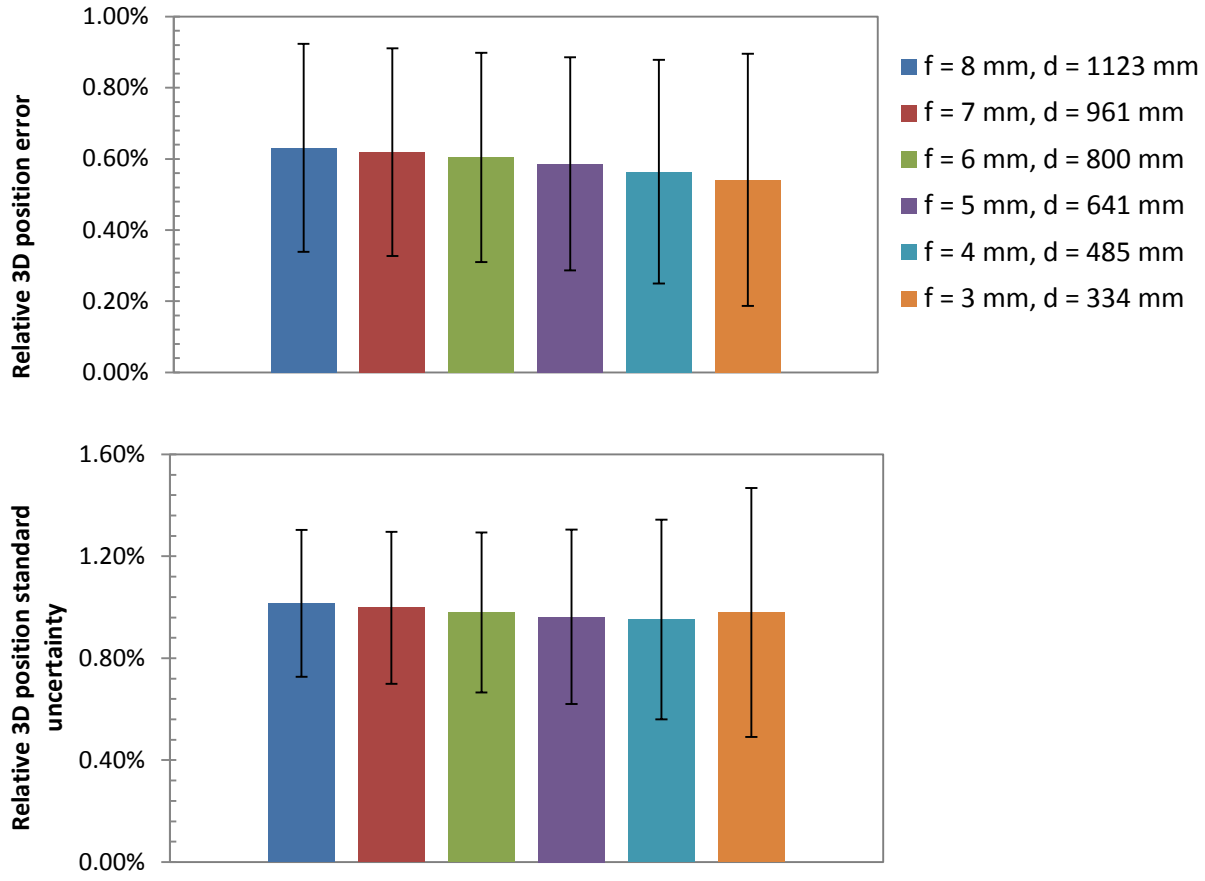

Figure 32: Impact of camera lens focal length and working distance on the relative 3D position uncertainty and error

5.4.3 Multiple cameras

Increasing the number of cameras involved in solving the image correspondence problem from two to four reduces the particle image matching ambiguities to near zero in a real experiment (H.G. Maas, 1992).The goal of this section is to characterize the impact of increasing the number of cameras involved in the 3D reconstruction process from two to four on the average 3D position combined standard uncertainty and 3D reconstruction error.

Two cases were studied (Figure 33): Case 1) Four cameras well positioned at 45 degrees apart arranged with their projective centers in a square, Case 2) same as Case 1 but with cameras 2 and 3 ill-positioned with only 10 degrees of separation. For each case the average relative 3D

position combined standard uncertainty over all points in the 16 x 16 x 16, (300 mm x 300 mm x 300 mm), grid were calculated based on all possible unique two, three and four-camera combinations. Then the relative 3D reconstruction error was calculated for each camera combination when all cameras in the group contain uniformly distributed random ± 0.5, 1.0 and 2.0 pixel detection errors.

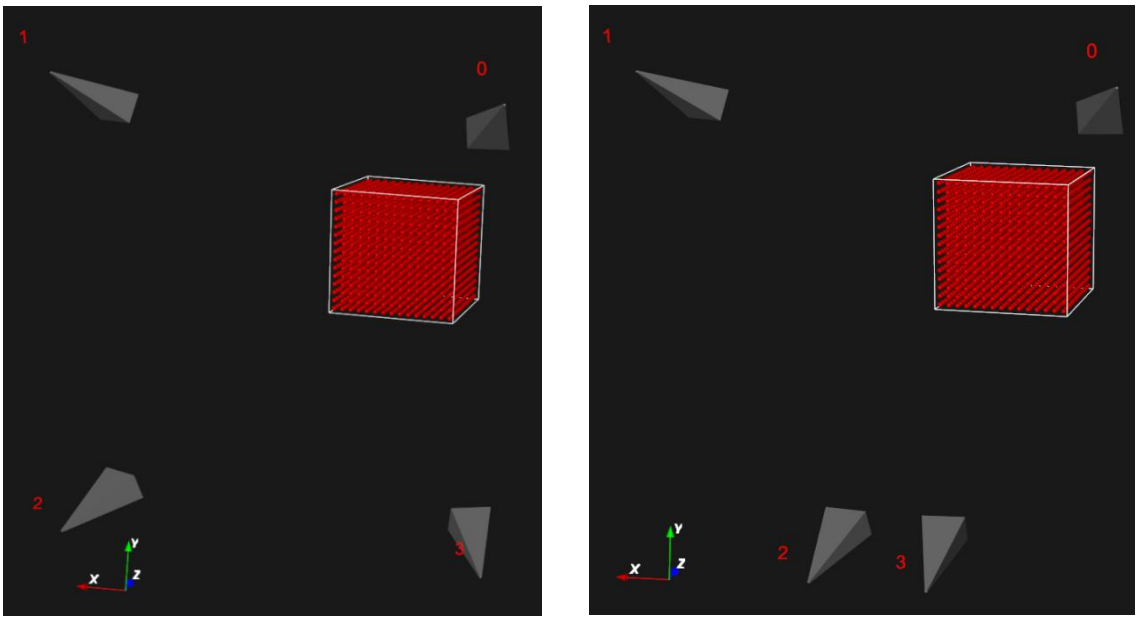

Figure 33: Camera placement for multi-camera sensitivity analysis: Case 1 (left) shows cameras well positioned at 45 degrees separation, Case 2 (right) shows two cameras ill conditioned at 10 degrees separation

Figure 34 and Figure 35 show the average condition number and relative 3D particle position uncertainty respectively for each camera group. The ill-conditioned camera pair (2-3) from Case 2 is easily identified by the high condition number compared to other pairs. The interesting result is that by adding a third or fourth camera (placed at a good angle from the others) to the ill-conditioned pair converts the group into a well-conditioned system. In Case 2, camera group 2-3 has an average condition number of 11 resulting in an average relative 3D position uncertainty of 4.7 ± 2.7%, but with the addition of camera 0 the group 0-2-3 has a condition number of 2.5 and relative position uncertainty of 1.0 ± 0.24%. Adding a fourth camera (group 0-1-2-3) further reduces the average condition number to 2.1 and relative 3D position uncertainty to 0.81 ± 0.17%. Therefore 3D particle position uncertainty of the system can be made less sensitive to camera placement by utilizing more than two cameras in the 3D reconstruction process.

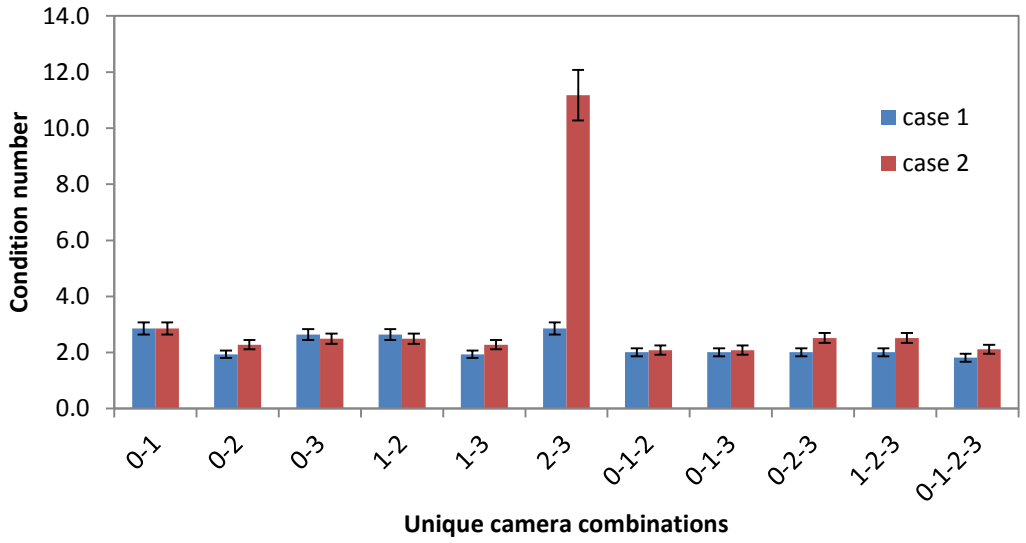

Figure 34: Comparison of condition number for Case 1 and 2 highlighting the ill-conditioned camera pair (2-3) in case 2 and robustness of the three and four camera combinations to ill-conditioning of a single pair

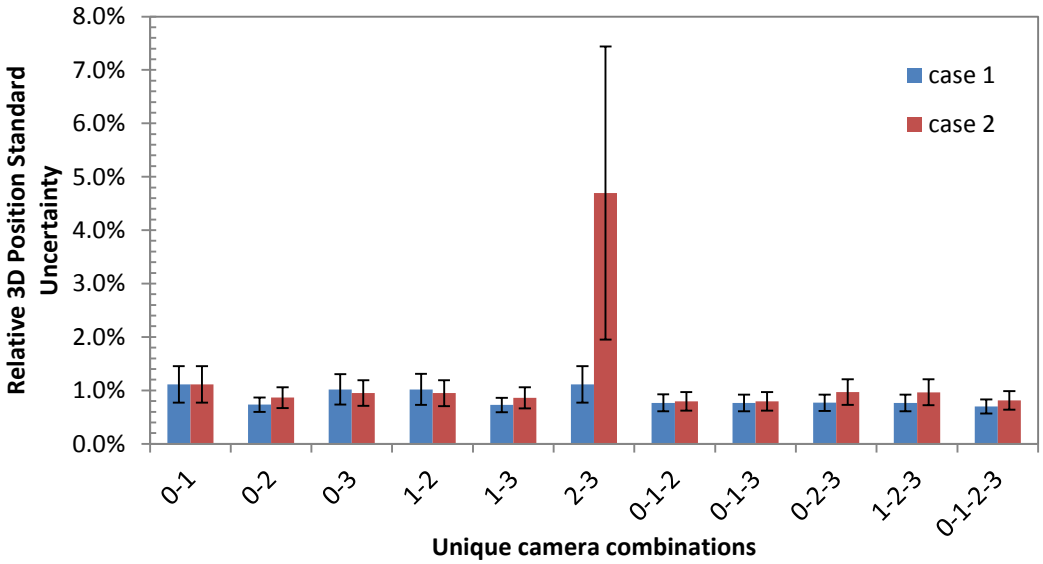

Figure 35: Comparison of relative position uncertainty for Case 1 and 2 highlighting the ill-conditioned camera pair (2-3) in case 2

The impact of camera grouping and particle image detection errors on the 3D position error are shown in Figure 36 and Figure 37 for Case 1 and 2 respectively. Again the ill-conditioned pair 2-3 of Case 2 is easily spotted in Figure 37, where random particle image detection errors of ±2.0 pixels result in a relative position error of 2.13 ± 1.47% verses 0.66 ± 0.32% for the same pair in Case 1. The same stabilizing effect is seen when the ill-conditioned pair is grouped with a third and fourth camera. In both cases the four camera group had the lowest condition number,

lowest 3D position uncertainty and was the most robust against random errors in the detected particle centroid location.

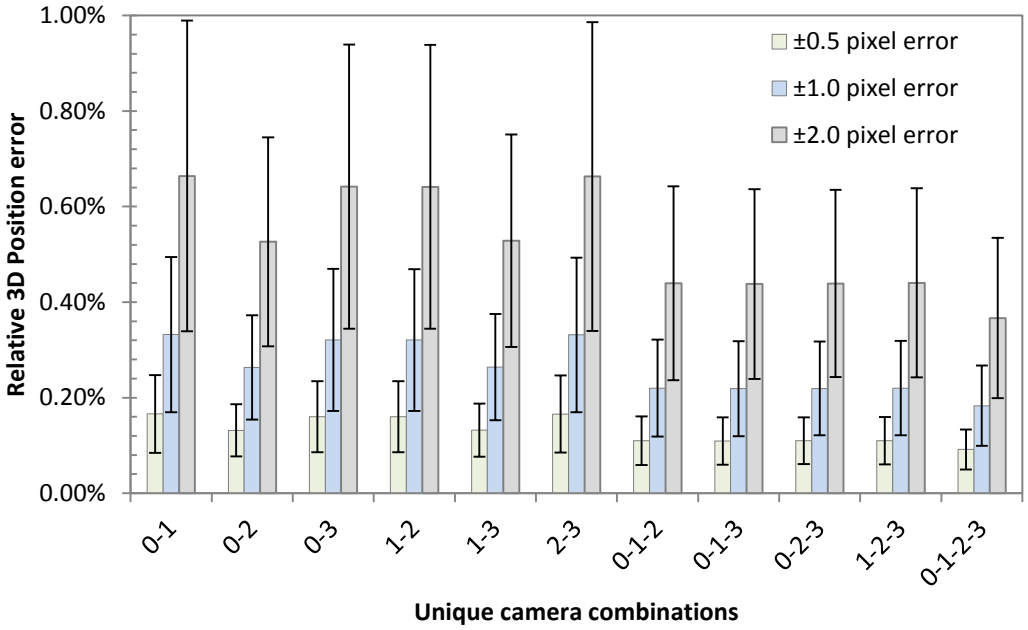

Figure 36: Average relative position error for Case 1, random ±0.5, 1.0, and 2.0 pixel detection errors applied to particle images in all cameras

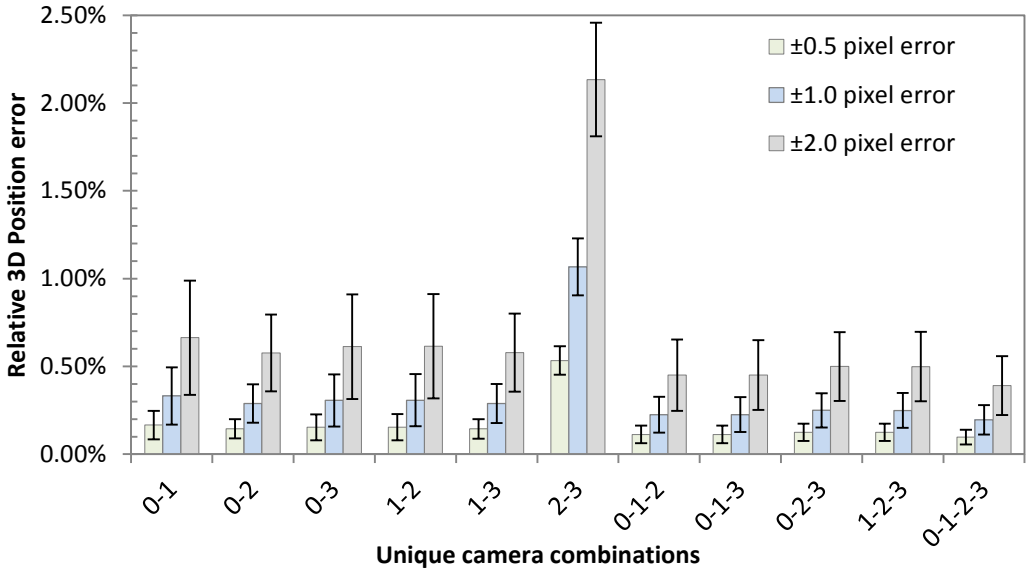

Figure 37: Average relative position error for Case 2, random ±0.5, 1.0, and 2.0 pixel detection errors applied to particle images in all cameras

5.5 Conclusions from uncertainty and sensitivity analysis

This chapter covered the design decisions made in selecting the LPT system hardware and derived the procedure to calculate measurement standard uncertainty for 3D particle position, velocity and acceleration. The uncertainty evaluation procedure was used to conduct sensitivity analyses to determine the impact of camera calibration parameter uncertainty, camera placement and usage of multiple cameras in the 3D reconstruction process. The conclusions from these studies were:

- The most influential factor in uncertainty propagation into the 3D reconstructed particle position was found to be the pixel location of the particle image centroid. When all camera parameters were given a relative uncertainty of 0.1%, the particle centroid location uncertainty lead to a 3D position uncertainty of nearly 1.0% relative to the length scale of the observed volume (300 mm). Therefore it is important to ensure that the particle centroid detection algorithm minimizes the uncertainty due to random image noise as much as possible.
- When two cameras are used for 3D reconstruction, the goal in positioning the two cameras should be to minimize the condition number of the **A** matrix in the least squares 3D reconstruction problem. This occurs when two cameras are placed with their image planes perpendicular to each other.
- The selections of lens focal length and camera working distance do not significantly impact the 3D position uncertainty or error, for equivalent camera setups (large focal length and large working distance or short focal length and short working distance). Therefore other factors such as light intensity and lens distortion should guide the decision on lens selection and camera working distance for a given volume.
- It is best to use four cameras in the 3D reconstruction process. Grouping the cameras by fours makes the system very robust to image centroid detection errors and inherent uncertainty in the camera parameters. It was shown that by adding one or two cameras to a pair of ill-placed (nearly linearly dependent) cameras, the sensitivity of the 3D reconstruction to input errors and uncertainties can be greatly reduced.

6 Experimental validation and applications

6.1 Overview

The goal of the experimental analysis is to validate the 3D position, velocity and acceleration measurements of the real-time Lagrangian particle tracking (LPT) system and evaluate the derivative parameters including velocity, turbulence intensity, Reynolds stress, and static pressure in known flow fields. In the first subsection, the accuracy and uncertainty of the 3D reconstruction process will be quantified using a known static calibration object. In the second subsection the accuracy and uncertainty of velocity and acceleration measurements will be evaluated using a known object motion. In the third subsection the particle tracking system will be applied to characterize an unconfined round turbulent air jet, which has been well documented in literature. The round jet flow will be used to validate the measured distributions of velocity, turbulence intensity, and Reynolds stress. Finally the LPT system will be applied to characterize an unconfined forced vortex and the resulting velocity, Reynolds stress, turbulent kinetic energy, and static pressure distributions will be analyzed to evaluate the system's capability of measuring particle transport in complex flow fields.

6.2 Camera setup and calibration

For the experimental analysis, the camera system was arranged in hexagon pattern with all cameras angled towards a central point roughly 1.0 m from the lenses as shown in Figure 38. The volume of interest for the following experiments was set to 160 x 160 x 335.5 mm, which was much less than the overall observable volume. The volume of interest was discretized by a virtual finite volume grid containing 31 x 31 x 65 cubic cells of side length 5.16 mm. The finite volume grid contains statistical accumulators, implemented as C++ objects, for all major flow parameters at the center of each cubic cell as described in Chapter 3. This grid is called the Statistical Accumulator Grid (SAG) in the following sections. The virtual representation of the camera setup and SAG are shown in Figure 39.

The cameras were calibrated to determine the intrinsic and extrinsic parameters along with associated uncertainties which are given in Table 13 through Table 15. The intrinsic parameter

uncertainty was obtained by running the calibration algorithm for 25 sets of images, where each set is formed by randomly selecting 35 calibration images from a total population of 50 images. Each image is of the calibration board at different orientations with respect to the sensor plane. The calibration algorithm determines a group of optimal camera parameters for each set of images, and then the uncertainty in each parameter is determined by finding the standard deviation of the optimal parameters among the sets. The uncertainty of the principle point is negligible as it was held constant during the optimization process in order to facilitate convergence. The extrinsic parameter uncertainty was obtained by taking the standard deviation of the translation and rotation vectors over 25 images of a static calibration board. The particle image centroid location uncertainty (pixels) is dependent on the experimental conditions (particle characteristics, illumination, distance from cameras, etc.) and therefore cannot be determined in advance. The image centroid uncertainty is determined in section 6.3 and used to calculate the average 3D position combined standard uncertainty of the setup. The standard uncertainties for the calibration parameters are determined to be Type A as described by the Guide to the Expression of Uncertainty in Measurements (GUM) approach (ISO, 2009; Kirkup & Frenkel, 2006) and are calculated as the standard deviation of the mean (equation 6-1).

$$u(\bar{q}) = \frac{s(\bar{q})}{\sqrt{n}} \tag{6-1}$$

Table 13: Intrinsic camera parameters and associated Type A standard uncertainties (sample size = 25)

Camera	Focal length (pixels)		Principal point (pixels)	
	$f_x \pm u(f_x)$	$f_y \pm u(f_y)$	$c_x \pm u(c_x)$	$c_y \pm u(c_y)$
0	1284.06 ± 0.09	1284.06 ± 0.09	314.44 ± 2.04E-14	250.50 ± 1.09E-14
1	1207.81 ± 0.08	1207.81 ± 0.08	346.17 ± 1.84E-14	262.93 ± 1.06E-14
2	1212.57 ± 0.06	1212.57 ± 0.06	332.82 ± 1.17E-14	264.29 ± 1.06E-14
3	1289.06 ± 0.11	1289.06 ± 0.11	310.93 ± 1.27E-14	238.02 ± 8.28E-15
4	1251.30 ± 0.07	1251.30 ± 0.07	337.11 ± 1.17E-14	271.29 ± 0.00E+00
5	1295.28 ± 0.05	1295.28 ± 0.05	301.52 ± 1.27E-14	237.78 ± 1.09E-14

Table 14: Lens distortion coefficients and associated Type A standard uncertainties (sample size = 25)

Camera	Radial distortion coefficients		Tangential distortion coefficients	
	$k_1 \pm u(k_1)$	$k_2 \pm u(k_2)$	$p_1 \pm u(p_1)$	$p_2 \pm u(p_2)$
0	-3.61E-01 ± 4.91E-04	3.39E-01 ± 9.34E-03	-1.62E-04 ± 1.53E-05	-2.56E-03 ± 2.13E-05
1	-3.49E-01 ± 6.35E-04	2.46E-01 ± 1.40E-02	3.88E-04 ± 2.16E-05	-3.93E-04 ± 2.18E-05
2	-3.53E-01 ± 6.05E-04	2.19E-01 ± 9.20E-03	-4.13E-04 ± 1.77E-05	1.21E-03 ± 1.55E-05
3	-3.43E-01 ± 3.51E-04	2.20E-01 ± 7.01E-03	-7.40E-04 ± 2.07E-05	-1.31E-03 ± 2.12E-05
4	-3.44E-01 ± 3.24E-04	1.50E-01 ± 4.34E-03	9.20E-04 ± 1.24E-05	4.98E-04 ± 1.58E-05
5	-3.45E-01 ± 2.63E-04	1.99E-01 ± 4.39E-03	2.46E-04 ± 1.09E-05	-2.50E-03 ± 8.25E-06

Table 15: Extrinsic camera parameters and associated Type A standard uncertainties (sample size = 25)

Camera	Translation Vector			Rotation Vector		
	$T_1 \pm u(T_1)$ mm	$T_2 \pm u(T_2)$ mm	$T_3 \pm u(T_3)$ mm	$r_1 \pm u(r_1)$	$r_2 \pm u(r_2)$	$r_3 \pm u(r_3)$
0	-208.98 ± 1.57E-02	-56.37 ± 9.41E-03	1090.67 ± 1.05E-01	6.03E-01 ± 1.58E-04	-6.88E-02 ± 1.51E-04	5.75E-02 ± 5.72E-05
1	211.79 ± 7.98E-03	67.20 ± 5.15E-03	1085.08 ± 5.33E-02	-3.60E-01 ± 3.64E-04	3.88E-01 ± 2.75E-04	3.09E+00 ± 5.06E-05
2	201.81 ± 1.36E-02	65.43 ± 8.58E-03	1036.67 ± 6.92E-02	-6.34E-01 ± 3.03E-04	-3.74E-01 ± 3.36E-04	-3.01E+00 ± 6.60E-05
3	-200.71 ± 1.15E-02	-67.40 ± 5.45E-03	1087.47 ± 5.63E-02	-2.24E-01 ± 2.81E-04	-4.14E-01 ± 2.61E-04	5.78E-03 ± 5.01E-05
4	223.20 ± 7.30E-03	61.43 ± 7.41E-03	1189.36 ± 4.56E-02	1.25E-01 ± 2.22E-04	-8.88E-01 ± 1.69E-04	2.96E+00 ± 7.75E-05
5	-197.31 ± 9.42E-03	-58.74 ± 4.71E-03	1135.30 ± 4.55E-02	-2.35E-01 ± 1.57E-04	2.33E-01 ± 1.48E-04	-8.44E-03 ± 3.81E-05

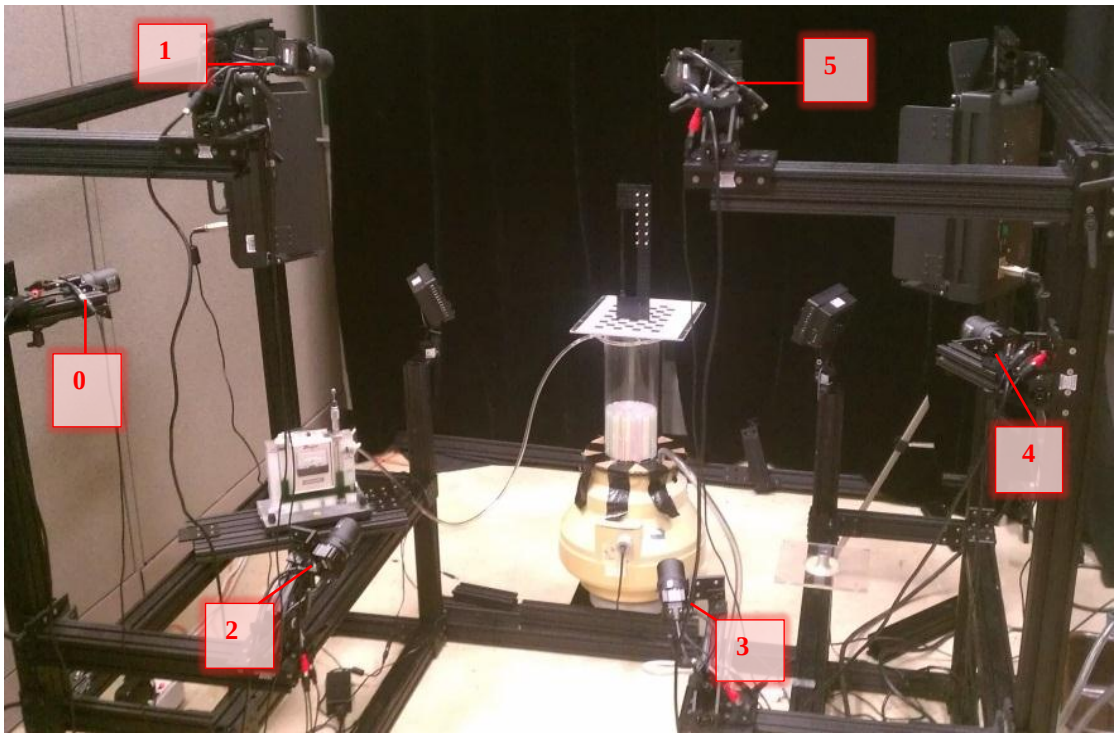

Figure 38: Camera arrangement for experimental analysis

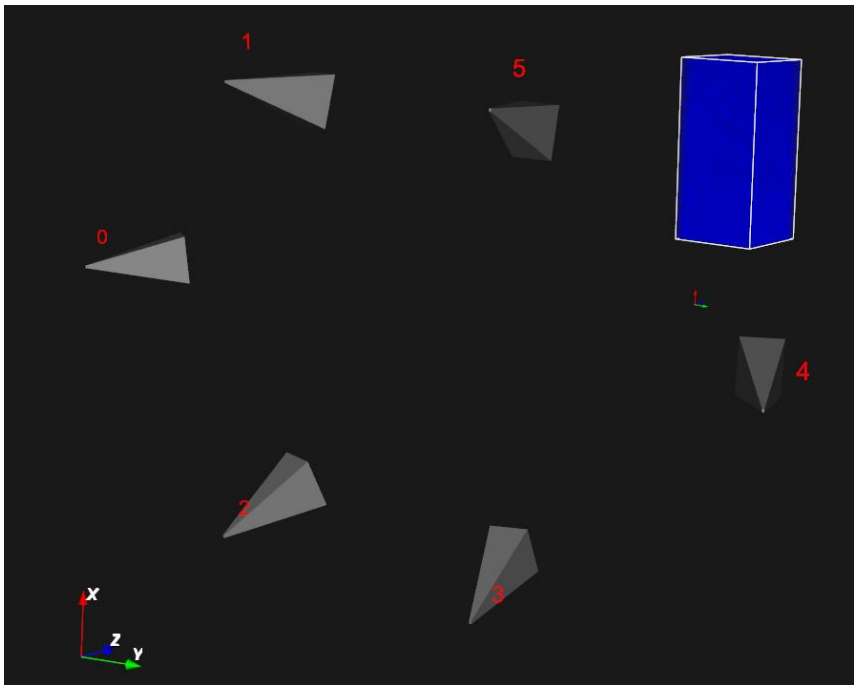

Figure 39: Virtual view of camera setup and the statistical accumulator grid (SAG)

6.3 Validation of 3D position measurement and uncertainty

6.3.1 Objective and approach

The first step towards validating the system's 3D position measurement performance is verifying the accuracy and uncertainty from the image processing, particle detection, and 3D reconstruction process. The goal of this section is to answer the following three questions.

1) What is the actual particle image centroid position uncertainty for each camera in real experiment conditions?
2) What is the actual 3D position combined standard uncertainty for the setup?
3) Is the position measurement accurate within the calculated 3D position uncertainty?

To accomplish this, a static object with fixed particles 3.0 mm in diameter, roughly the size of the helium filled soap bubbles used in later flow experiments, was imaged and the 3D particle positions were reconstructed. The distances between adjacent particles were measured with the LPT system and compared with measurements made using a caliper on the static object. The 3D position measurement was deemed valid if the averaged distance measurements from both methods are equal within standard uncertainty.

6.3.2 **Methodology**

The static object held 12 white beads, roughly 3.0 mm in diameter, in a structured grid spaced 20 mm apart as shown in Figure 40. The 12 particles form the vertices of five squares, and the lengths of the 16 line segments representing the sides of these squares serve as the basis for comparison. The line segments were first measured using a caliper with a smallest marked increment of 1/32 inch to create the accepted benchmark for comparison. These segment lengths are labeled 1 to 16 starting with the vertical segments from the top left point going downward, then the top right point downward followed by the horizontal segments top to bottom. The caliper measurements and associated uncertainties are given in Table 16. The standard uncertainty in the caliper measurement was calculated according the Type B classification as described by the Guide to the Expression of Uncertainty in Measurements (GUM) approach (ISO, 2009; Kirkup & Frenkel, 2006).

Tests were run at three frame rates, 30, 60 and 120 fps to collect 100 images at three positions. The static object was placed at three positions by varying the depth away from the cameras: Position 1 = [0,0,80] (mm), Position 2 = [0,0,0] (mm) and Position 3 = [0,0,-80]. These positions define the extents of the volume of interest shown in Figure 39. The static object is shown at position 1 in Figure 38. Each of the 15 unique four-camera combinations was used to image the particles and reconstruct their 3D positions. Random variations in the particle image centriod location are the major sources of uncertainty in the reconstructed 3D particle positions during an experiment as determined in Chapter 5. The average particle image centroid location standard uncertainty (in pixels) for each camera was determined by averaging the pixel variances of all points on the static object.

Table 16: Distances between known particles on the static object measured with a caliper

	length measured	
Line segment	**inches**	**mm**
1	0.781	19.8
2	0.781	19.8
3	0.813	20.6
4	0.781	19.8
5	0.781	19.8
6	0.781	19.8
7	0.781	19.8
8	0.781	19.8
9	0.781	19.8
10	0.797	20.2
11	0.781	19.8
12	0.781	19.8
13	0.797	20.2
14	0.797	20.2
15	0.781	19.8
16	0.781	19.8
Mean length	0.786	**20.0**
Standard Deviation	0.009	0.24
Type B Uncertainty	**0.009**	**0.23**

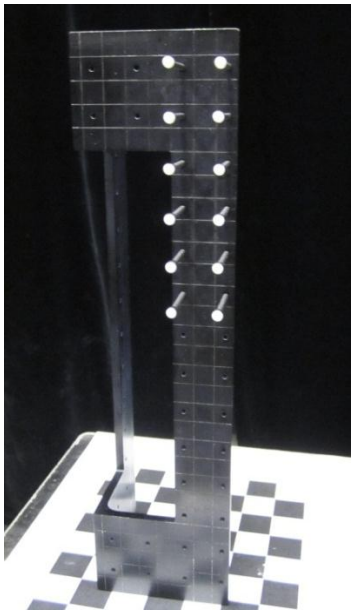

Figure 40: Static object for 3D particle position validation

6.3.3 **Results**

6.3.3.1 ***Particle image centroid fluctuation:***

Figure 41 shows the averaged particle image centroid position uncertainties over all three static object positions when all major experimental conditions (illumination, exposure, position) are held constant. Therefore the remaining source of variability being measured is due to random image noise affecting the centroid localization algorithm. It is clear that the average uncertainty increases with frame rate in all cameras, where the highest average of 0.09 ± 0.06 pixels was observed at 120 fps in camera 4. The direct impact of this variation can be seen as fluctuations in the 3D reconstructed particle positions, which are shown in Figure 42. The average 3D fluctuation due to image noise increases from 0.04 mm to 0.07 mm with an increase in frame rate from 30 fps to 120 fps respectively. Based on this data, a conservative value of 0.1 pixels was selected as the centroid location standard uncertainty for all cameras in the 3D position combined uncertainty analysis.

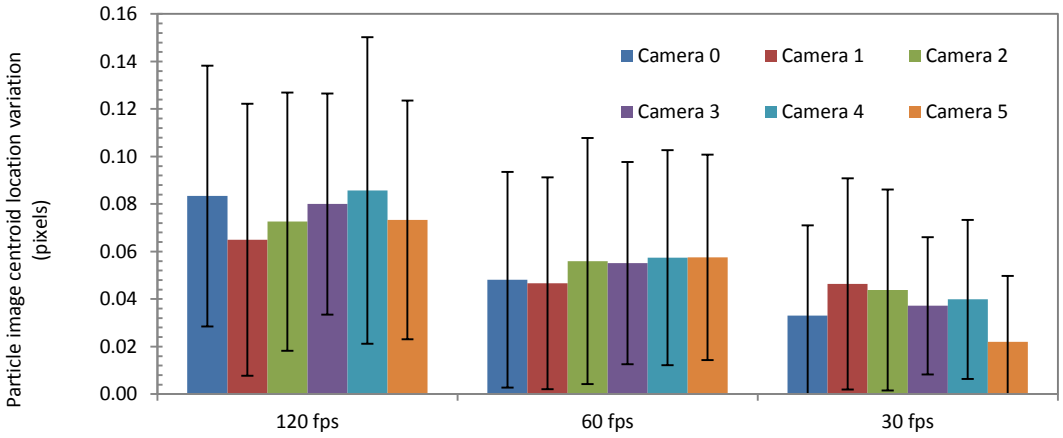

Figure 41: Observed particle image centroid location variation averaged over all three static object positions (error bars denote standard deviation)

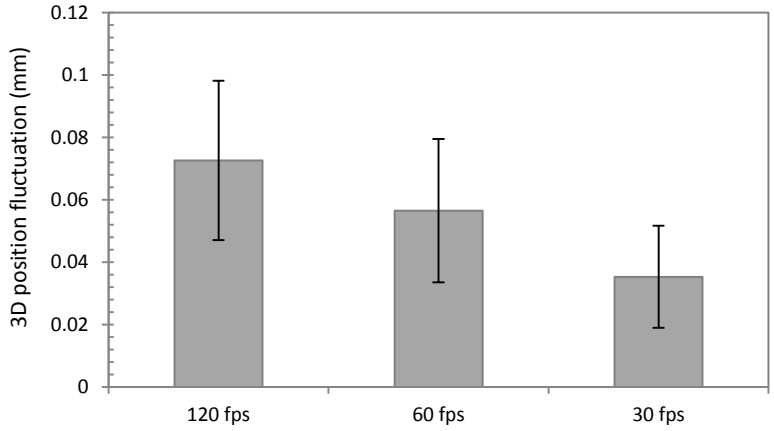

Figure 42: Observed particle 3D position fluctuation averaged over all three static object positions (error bars denote standard deviation)

6.3.3.2 *3D position combined standard uncertainty:*

Following the 3D position combined standard uncertainty calculation procedure derived in Chapter 5, the uncertainties of all 12 static object point reconstructions were determined at each of the three positions. These uncertainties were determined for each of the possible 15 unique groups of four cameras. Figure 43 shows the resulting averaged a) condition number for each camera group over all static particles and object positions and b) 3D position combined standard uncertainty. The condition number for each group is very similar with an overall mean of 2.52 ± 0.02. This indicates that the cameras are well placed in the domain and camera groups can be randomly selected for the 3D reconstruction of particle positions with little to no difference in

the resulting position uncertainty. The average 3D position combined standard uncertainty does vary between camera groups, however for the following analysis the average 3D position combined standard uncertainty of all groups, $u_c(\mathbf{x}_w) = 0.16$ mm, was used.

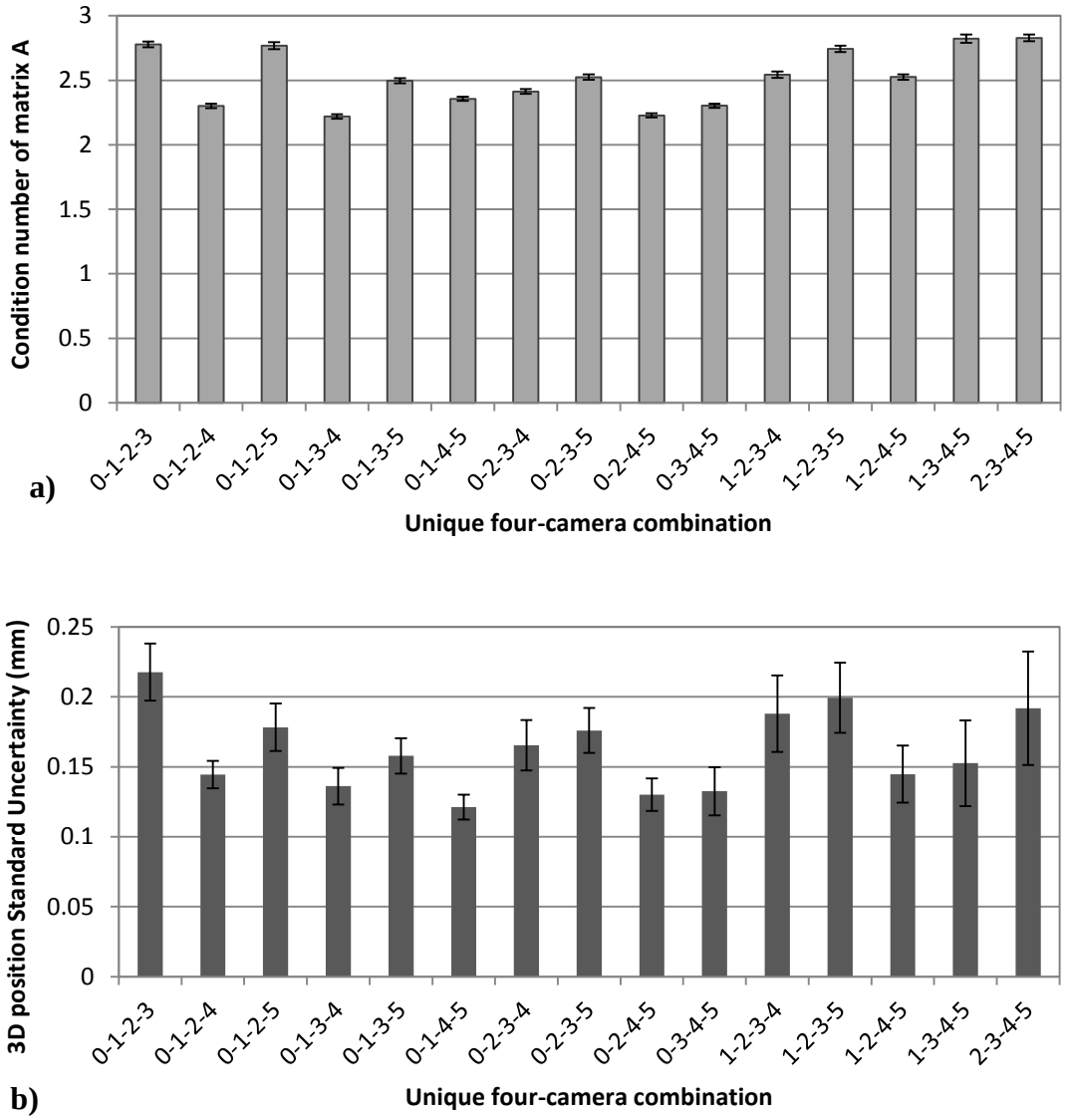

Figure 43: a) Mean and standard deviation of the condition number for 3D reconstruction and b) Mean and standard deviation of 3D position combined standard uncertainty (averages are over all static object points and positions for each camera combination)

6.3.3.3 *Comparison of two methods for point to point distance measurements:*

The average distance measurement of the 16 particle to particle segments at each static object position are shown in Figure 44 for each camera group. Overall the measurement was very precise for each camera group with an average standard deviation within each group of 0.33 mm.

All segment measurements were then averaged for all static object positions to obtain the mean segment length, which is then compared to the same value from the caliper measurement as shown in Table 17. The mean segment length measured by the particle tracking system was 20.1 ± 0.23 mm combined standard uncertainty over all segments. The measurements standard uncertainty of 0.23 mm was calculated by the root mean square of the standard uncertainty, $u_c(\mathbf{x}_w)$ = 0.16 mm, for the two points in each the segment. This agrees well with the caliper method which measured an average segment length of 20.0 ± 0.23 mm standard uncertainty. Therefore, it can be concluded that the particle tracking system accurately reconstructs 3D particle positions.

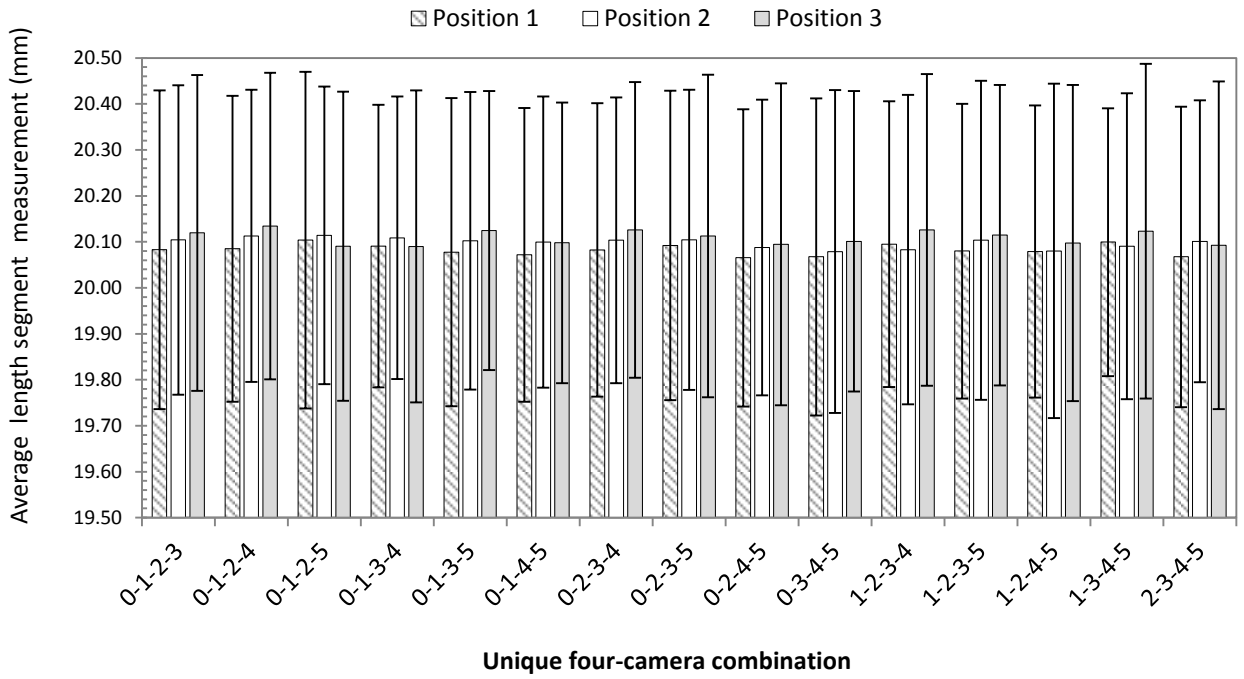

Figure 44: Distance measurement averages for the 16 particle to particle length segments

Table 17: Comparison of two measurement methods

Measurement method	Number of measurements	Average segment length (mm)	Average Standard Uncertainty (mm)
Caliper	16	20.0	0.23
Particle Tracking System	4800	20.1	0.23

6.3.4 **Discussion and summary**

Based on the 3D particle position validation test the following conclusions can be made regarding the camera setup and 3D reconstruction algorithms.

- The random image noise of the system increases with frame rate and causes average fluctuations of 0.07 mm in particle 3D position at 120 fps. This random fluctuation is less than the calculated combined standard uncertainty of 0.16 mm, which results from the combination of all calibration parameter standard uncertainties.

- The particle centroid location variation for small spherical objects 3 mm in diameter placed roughly 1.0 m away from camera lenses will be conservatively 0.1 pixels due to random image noise

- The average particle 3D position combined standard uncertainty for all unique four-camera groups is 0.23 mm based on the standard uncertainty of camera calibration parameters and uncertainty in particle centroid location.

- The particle tracking system was able to accurately measure the known distance between static particles and agrees with measurements made using a caliper

6.4 Validation of velocity and acceleration measurement

6.4.1 Objective and approach

The objective of the work described in this section is to experimentally validate the velocity and acceleration measurements made by the particle tracking system. Three key questions will be answered:

1. What is the actual velocity and acceleration measurement uncertainty for the setup?
2. Is the velocity and acceleration calculation accurate within a known uncertainty?
3. How does the frame rate impact the accuracy and uncertainty of the measurement?

The approach to answering these questions will be to use the system to observe a particle moving with a known velocity and acceleration, and then compare the measured magnitudes of these vectors with another measurement method. Validity of the measurements will be based on the two methods agreeing to within measurement uncertainty.

6.4.2 Methodology

To determine accuracy of the velocity and acceleration measurements, a known particle motion was created by fixing a 3 mm diameter bead to a rod and rotating the rod at a constant angular velocity of 60 rpm using an AC Geared Hysteresis Synchronous motor (Hansen Corporation) shown in Figure 45. The constant angular velocity and radius of rotation determine the magnitudes of velocity and acceleration as given by the following equations.

$$\|\mathbf{v}\| = v_t = \omega r$$
$$\|\mathbf{a}\| = a_r = \omega^2 r \qquad (6\text{-}2)$$

Figure 45: Test object to provide constant tangential velocity to a 3mm white bead

A caliper was used to measure the radius of rotation (length of the rod) and a stop watch was used to verify the angular velocity. The uncertainties of each of these measurements, $u(r)$ and $u(\omega)$, combine through the following equation to yield the standard uncertainty associated with each magnitude $u(v_t)$ and $u(a_r)$.

$$u^2(v_t) = \left[\omega \cdot u(r)\right]^2 + \left[r \cdot u(\omega)\right]^2$$
$$u^2(a_r) = \left[\omega^2 \cdot u(r)\right]^2 + \left[2r\omega \cdot u(\omega)\right]^2 \qquad (6\text{-}3)$$

The type B uncertainty of the length measurement was 1/64" which was one half of the smallest division on the caliper. The uncertainty in omega was determined by recording the time the motor took to complete 120 revolutions and finding the standard deviation of six measurements which was 0.44 rpm. The resulting "known" values for velocity and acceleration magnitudes are given in Table 18.

Table 18: Measured values and standard uncertainties of the "known" velocity and acceleration magnitudes

Parameter	Measured value	Type B standard uncertainty
Angular velocity, rpm [rad/s]	60 [6.28]	0.44[0.046]
Radius, inch[m]	3.89[0.0988]	0.0156[0.0004]
Measurand		**Combined standard uncertainty**
Velocity, m/s	0.621	0.005
Acceleration, m/s^2	3.90	0.059

For the particle tracking system the velocity and acceleration uncertainty are due to 3D position uncertainties propagated through the finite difference schemes as derived in the previous chapter. The uncertainty of the frame rate is estimated to be on the order of 10^{-6} s and therefore its impact on overall velocity and acceleration uncertainty is neglected. The standard uncertainties of the velocity and acceleration magnitudes then follow as:

$$u^2\left(\|\mathbf{v}\|\right) = \left(\frac{1}{2\Delta t}\right)^2 u^2\left(\|\mathbf{x}^{n+1}\|\right) + \left(\frac{-1}{2\Delta t}\right)^2 u^2\left(\|\mathbf{x}^{n-1}\|\right) \qquad (6\text{-}4)$$

$$u^2\left(\|\mathbf{a}\|\right) = \left(\frac{1}{\Delta t^2}\right)^2 u^2\left(\|\mathbf{x}^{n+1}\|\right) + \left(-\frac{2}{\Delta t^2}\right)^2 u^2\left(\|\mathbf{x}^{n}\|\right) + \left(\frac{1}{\Delta t^2}\right)^2 u^2\left(\|\mathbf{x}^{n-1}\|\right) \quad (6\text{-}5)$$

Where the 3D position uncertainties of a particle at any point in time is assumed similar

$$u\left(\|\mathbf{x}^{n}\|\right) \approx u\left(\|\mathbf{x}^{n+1}\|\right) \approx u\left(\|\mathbf{x}^{n-1}\|\right) \quad (6\text{-}6)$$

The uncertainty of 3D position is assumed to be similar for all particles observed in the volume of interest, which is supported by the findings the previous section. Therefore the value of $u\left(\|\mathbf{x}\|\right)$ = 0.16 mm found in the previous section was used.

The experimental analysis to measure velocity and acceleration using the particle tracking system were as follows:

1. Position the rotating object in three orientations near the center of volume of interest by setting the axis of rotation to approximately:
    a. parallel to the Z-axis, referred to as Z-0
    b. rotated 45 degrees to the Z-axis, referred to as Z-45
    c. rotated 45 degrees to the Y-axis, referred to as Y-45
2. Record velocity and acceleration magnitudes for the rotating particle for 10 revolutions and report the mean and standard deviation of each value
3. Repeat the test for frame rate values of 30, 60 and 120 fps

The positioning of the rotating object was not measured exactly as the goal of testing different orientations was to see if any measurement variation manifested when the axis of rotation changed.

6.4.3 **Results**

6.4.3.1 ***Velocity measurement and uncertainty:***

The velocity magnitude measurement results from the six experiments are shown in Figure 46. The mean velocity magnitude measurement did not vary with any statistical significance

between the three frame rates or axis of rotation orientation. The difference of the known velocity and measure value are shown in Table 19 with the associated uncertainties. The maximum velocity uncertainty associated with the 3D position combined standard uncertainty of 0.16 mm is 0.014 m/s at the high frame rate of 120 fps. The maximum absolute difference of the average measured velocity magnitude with the known value was 0.004 m/s, a relative difference of 0.72%. This shows good agreement with the known value and is well within the level of uncertainty for the particle tracking system measurement.

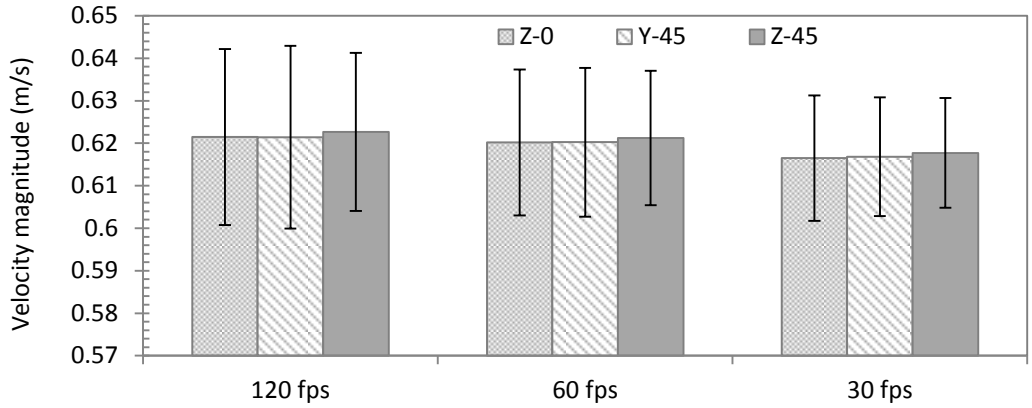

Figure 46: Average measured velocity magnitude of the rotating partcle at three orientations and three frame rates

Table 19: Measured velocity magnitudes at three rotation orientations and calculated uncertainty with frame rate

Axis of rotation	frame rate (fps)	Velocity magnitude (m/s)		Difference Known velocity = 0.621 ± 0.005 m/s	
		mean	$u_c(v)$	absolute (m/s)	relative
Z-0	120	0.621	0.014	5.30E-04	0.09%
	60	0.620	0.007	7.48E-04	0.12%
	30	0.616	0.003	4.44E-03	0.72%
Y-45	120	0.621	0.014	4.96E-04	0.08%
	60	0.620	0.007	6.79E-04	0.11%
	30	0.617	0.003	4.10E-03	0.66%
Z-45	120	0.623	0.014	1.75E-03	0.28%
	60	0.621	0.007	3.01E-04	0.05%
	30	0.618	0.003	3.20E-03	0.52%

6.4.3.2 *Acceleration measurement and uncertainty:*

The average measured acceleration magnitude values are shown in Figure 47. It is clear from the figure that as the frame rate increased the deviation in measured acceleration also increased significantly. This agrees with the uncertainty calculation which shows scaling with $1/\Delta t^2$. The measured acceleration for 30 and 60 fps are similar near 4 m/s^2 and do not vary significantly

with respect to the axis of rotation. The relative difference between the known magnitude of acceleration and the measured value is on the order of 1% for 30 fps, 10% for 60 fps and 100% for 120 fps. Therefore for lower frame rates the measured acceleration is in good agreement with the "known" value, but the "error" scales exponentially with respect to doubling the frame rate. This is a much greater sensitivity to frame rate than was expected, where the uncertainty should scale linearly with the time step squared. Possible explanations for this increased growth rate in error as a function of time step will be discussed below.

Nevertheless, the acceleration measurement is extremely sensitive to uncertainty in the particle positions. At 120 fps, the particle position combined standard uncertainty of 0.16 mm for the current system yields a large uncertainty in acceleration of 5.64 m/s^2. This is of great concern and brings into question the validity of utilizing the acceleration measurement.

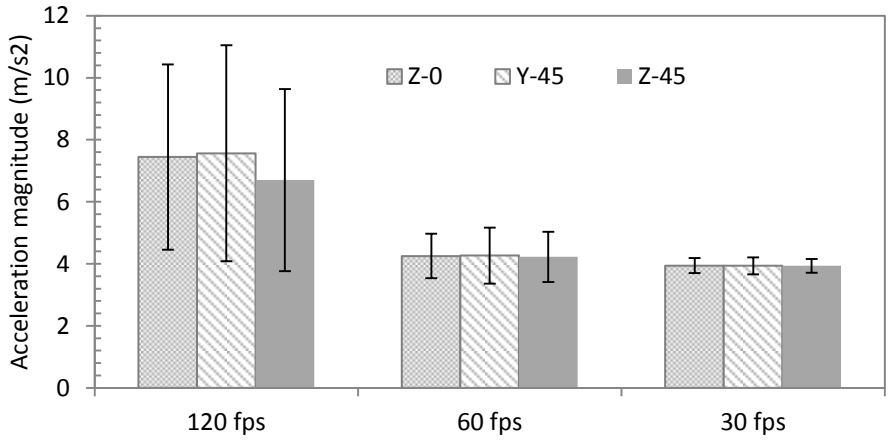

Figure 47: Average measured acceleration magnitude of the rotating partcle at three orientations and three frame rates

Table 20: Measured acceleration magnitudes at three rotation orientations and calculated uncertainty with frame rate

Axis of rotation	frame rate (fps)	Acceleration magnitude (m/s^2)		Difference Known acceleration = 3.90 ± 0.059 m/s^2	
		mean	$u_c(a)$	absolute (m/s^2)	relative
Z-0	120	7.44	5.64	3.542	90.78%
	60	4.25	1.41	0.352	9.03%
	30	3.94	0.35	0.039	1.00%
Y-45	120	7.56	5.64	3.663	93.90%
	60	4.27	1.41	0.364	9.33%
	30	3.94	0.35	0.037	0.96%
Z-45	120	6.70	5.64	2.799	71.75%
	60	4.23	1.41	0.325	8.32%
	30	3.94	0.35	0.035	0.91%

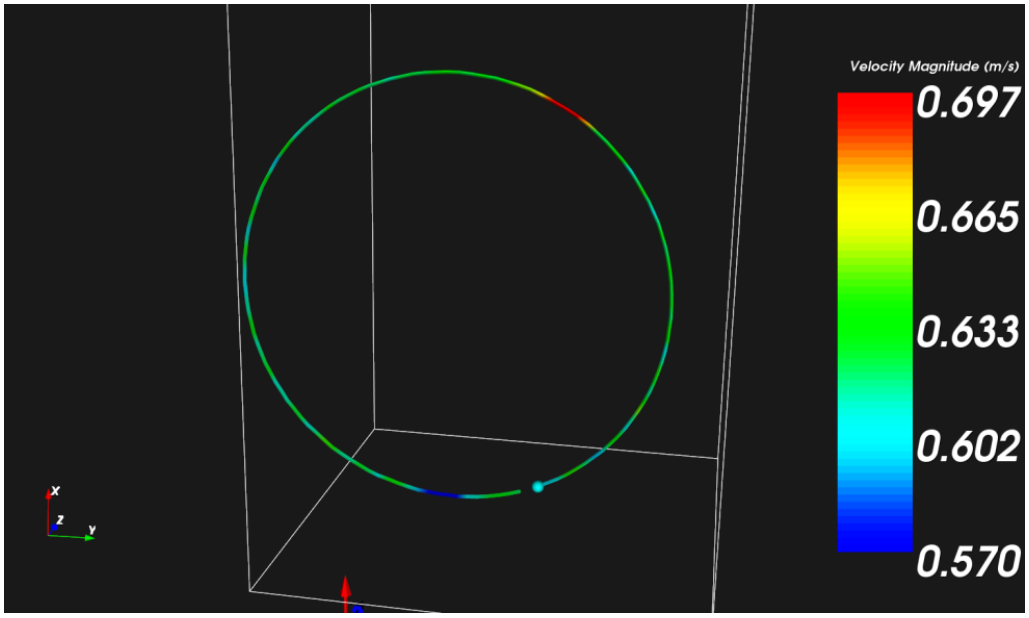

Figure 48: Visualization of rotation particle trajectory colored by velocity magnitude measured at 120 fps

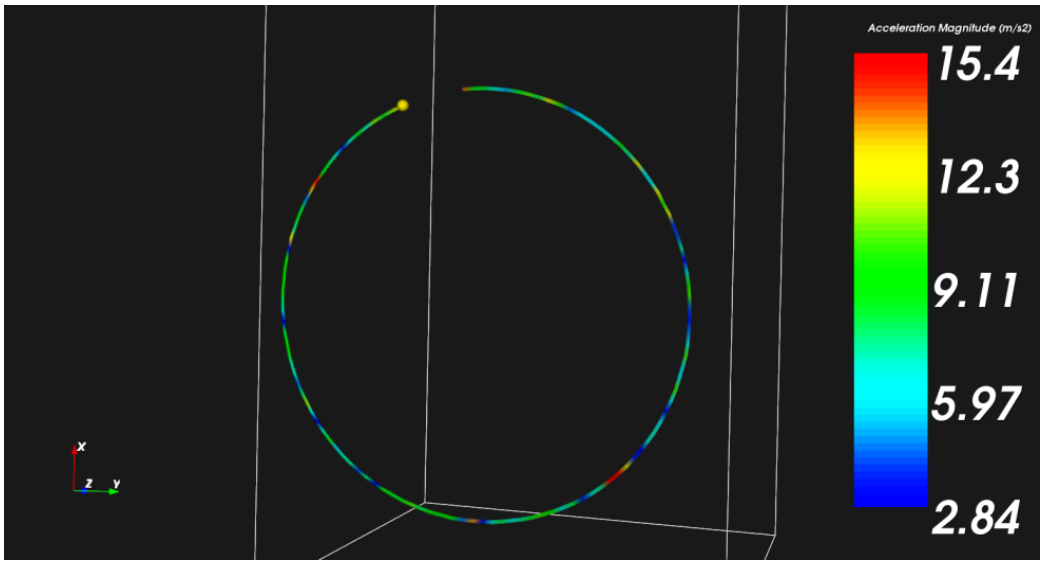

Figure 49: Visualization of rotation particle trajectory colored by acceleration magnitude measured at 120 fps

6.4.4 **Discussion and summary**

The velocity measurement was both accurate and precise for the three frame rates and rotational axis orientations. From the uncertainty analysis it is apparent that the velocity uncertainty scales linearly with the frame rate, however the lowest frame rate of 30 fps made the least accurate predictions of the tangential speed of the rotation particle with an average relative difference of 0.63%. One explanation for this is that the assumption that the synchronous motor maintained a constant angular velocity is not valid. Therefore the tangential speed fluctuates due to "slip" in the motor's gear box. Figure 48 shows the velocity magnitude visualization along the particle's trajectory and it is clear to see that the velocity was not constant along the path. The particle actually slows down momentarily on the upswing and speeds up on the down swing due to gravity and a slight slippage in the gearing. While the rpm of the motor maintains 60 rpm as measured, it actual accelerates and decelerates slightly during a single revolution. At lower frame rates this may be missed due to under sampling of the trajectory and therefore the overall tangential speed measurement over ten revolutions will diverge from the actual value.

Figure 49 shows the rotating object's trajectory visualization imaged at 120fps colored by acceleration magnitude. There appears to be bands of high then low acceleration ranging from 2.8 to 15 m/s^2. While a portion of this fluctuation is due to the slippage in the motor causing systematic localized acceleration events along the path, most is due to increased random image noise and 3D position uncertainty.

The extraordinarily high uncertainty in the acceleration measurement resulting from a 3D position combined standard uncertainty of 0.16 mm is cause for concern. The 3D position combined standard uncertainty takes into account all uncertainties associated with the calibration parameters which are fixed at the time of the experiment.

The follow conclusions can be made from the rotating particle experiment:

- The velocity measurement was precise and accurate within the measurement uncertainties for each frame rate 120, 60 and 30 fps. The highest average relative difference with the known tangential speed was 0.74% at 30 fps. The difference could not be fully accounted for by the calculated measurement uncertainty and therefore can be attributed to an observed local velocity fluctuation in the synchronous motor due to gear slip.

- The velocity and acceleration measurements did not vary with any significance in the volume of interest as the axis of rotation of the known object was changed. Therefore this confirms that the uncertainty associated with each measurement is nearly uniform throughout the measurement volume and does not depend on the particle's path within the volume of interest.

- The average acceleration magnitude measurement was accurate within 1% of the predicted value for lower frame rates, but differed significantly, up to 90%, for the higher frame rate of 120 fps. This was attributed to the increased uncertainty due to the inherent scaling with frame rate squared.

6.5 Application to free round air jet flow

6.5.1 Objective and approach

The objective of the experiment described in this section is to validate the capability of the real-time Lagrangian particle tracking (LPT) system to 1) track particles in three dimensional turbulent flow, 2) correctly store the instantaneous velocities to the statistical accumulator grid, and 3) to characterize velocity profiles, turbulence intensities and Reynolds shear stresses through the filter of inertial particle motion.

While the main objective is to validate the real-time LPT system as described, by using neutrally buoyant helium soap bubbles (d_p = 1 to 5 mm), this analysis may also serve as a test for the conclusions by (Bourgoin et al., 2011) and (Ouellette et al., 2008), who found that velocity statistics of a turbulent flow are not changed significantly if observed through the motion inertial particles. If the turbulent flow field (intensities and spatial distributions) can be characterized within accepted ranges by velocities of inertial particles then this experiment may also lead to a conclusion that Lagrangian analysis of such particles is a valid approach to characterize flows of the selected type and Reynolds number.

The general approach of this case study was to create a free air round turbulent jet and use the real-time LPT system to characterize the velocity and turbulence distributions. The centerline axial velocity decay, transverse velocity profiles, axial turbulence intensities, and Reynolds shear stresses will be compared with those of hot-wire anemometry measurements found in literature.

Round turbulent jets have been widely studied over the years and serve as a fundamental building block of turbulent flow research in fluid dynamics (Boguslawski & Popiel, 1979; Ferdman, Otugen, & Kim, 2000; Quinn, Pollard, & Marsters, 1985). Past research on round turbulent jets has included experimental investigations through hot wire anemometry (Fellouah, Ball, & Pollard, 2009), Particle Image velocimetry, and Laser Doppler anemometry, in addition numerical analysis. In particular researchers have characterized the velocity profiles and found that jet flow can be broken down into three regimes the *near field*, *intermediate field* and *far field* (Fellouah et al., 2009). The near and intermediate fields $0 \leq x/d \leq 30$ contain the region where the flow is developing. In the far field, $x/d > 30$, the flow is fully developed and is

characterized by self-similarity. This is defined by the observation that fully developed symmetrical jet flows have mean velocity and turbulence intensity profiles that follow the same shape function, only varying by a scale factor along the axis of the jet. Research has shown that the self-similar transverse profiles of axial velocity (U) normalized by the centerline axial velocity U_c can be characterized by the following Gaussian function (Xu & Antonia, 2002).

$$\frac{U(x,y)}{U_c(x,0)} = \exp\left(-A\left(y/x\right)^2\right) \tag{6-7}$$

Where *A* is a constant equal to about 75.2 for an axisymmetric round jet (Xu & Antonia, 2002). The coordinate system commonly used is positive X in the axial distance from the jet source while Y and Z define planes in the transverse direction.

Another key observation is that the centerline axial velocity decay rate in the X direction is constant for a self-similar fully developed turbulent jet and can be described by equation 6-8 (Fellouah et al., 2009; Ferdman et al., 2000).

$$\frac{U_j}{U_c} = \frac{\left(x - x_0\right)}{Bd} \tag{6-8}$$

Where *B* is the decay constant ranging between 5.7 and 6.7, *d* is the diameter of the jet, and x_0 is the distance to the virtual source of the jet which has been found to lie between 0 and 7*d* for different types of round turbulent jets (Fellouah et al., 2009).

The turbulence intensities and Reynolds stress profiles of round turbulent jets have also been well characterized in published research and a range of accepted magnitudes are known. For maximum turbulence intensities of axial velocity, the nominal range for a round axisymmetric jet has been reported to be between 0.24 and 0.35 (Ferdman et al., 2000). The Reynolds shear stress term representing the covariance of the axial and transverse velocities has been reported by many researchers to be anti-symmetrical about the axial centerline and on the order of -0.02 $\leq \overline{u'v'}/U_c^2 \leq 0.02$ (Ferdman et al., 2000; Xu & Antonia, 2002).

6.5.2 **Methodology**

The turbulent jet was created using a ¾" diameter nozzle. The air flow was supplied by a 150 W 2700 rpm variable speed vane axial fan which supplied through a 10.5 cm flow straightener made from drinking straws and then into a 12" long 4.5" ID acrylic tube with the nozzle fixed to the top. The experimental apparatus is shown in Figure 50.

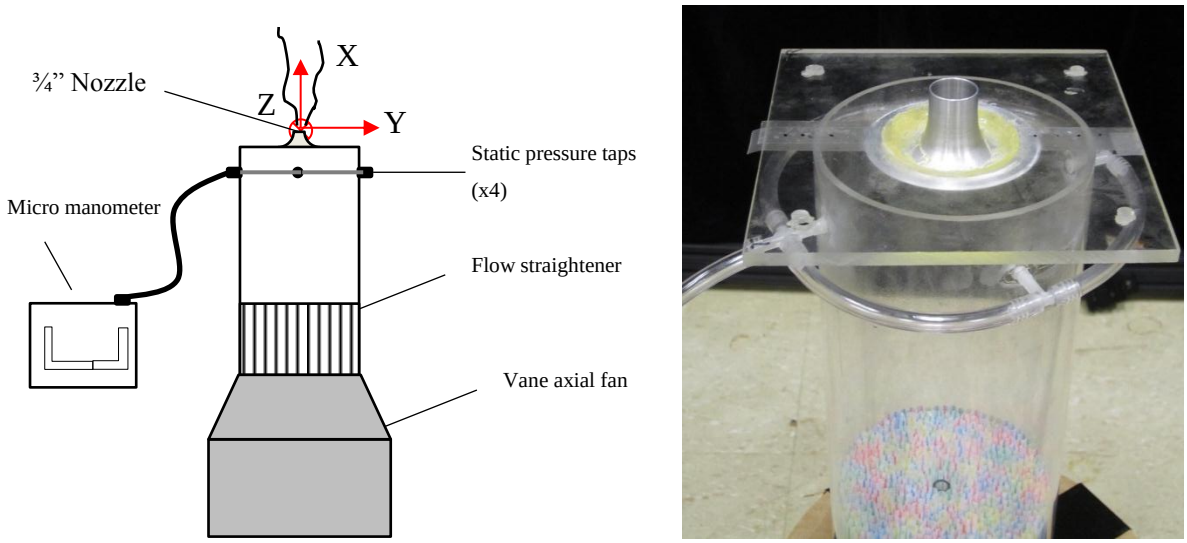

Figure 50: Experimental apparatus for creating turbulent circular jet flow

The nozzle was of the type used for air flow measurement as defined in ANSI/ASHRAE 51, 1999. The air flow through the nozzle was controlled by monitoring the static pressure drop created across the nozzle using a high resolution water manometer with a range of 0.0 to 2.0 ± 0.0005" H_2O (model: Dwyer 1430). Air flow rate was calculated based on the measured static pressure drop (ΔP), diameter of the nozzle (d), ambient wet bulb temperature (T_w), dry bulb temperature (T_d) and barometric pressure (P) using the equations and methods given in (ANSI/ASHRAE/AMCA, 1999).

The exit velocity was set to 9.73 m/s which resulted in a Reynolds number base on nozzle diameter of 11900. The observed volume of interested during the experiment was $15 \leq x/d \leq 33$ to allow the intermediate to far region of the jet to be observed. The overall standard uncertainty of the jet outlet bulk velocity was 0.34 m/s.

The cameras were set to record at 120 frames per second. The tracer particles were neutrally buoyant Helium filled soap bubbles ranging in diameter from 1 to 4 mm and generated through

an SAGE Action Model 5 bubble generator (SAI, 2012). The bubbles were injected into the flow field through a small tube in the center of the flow straightener. The stationary turbulent air jet flow was observed for 20 min with roughly 300 bubbles tracked at each time step (Δt = 1/120 s). The statistical accumulator grid collected up to 2300 observations per cell, which were used to characterize the turbulence and mean velocity profiles of the flow field.

6.5.3 **Results**

The visualized helium soap bubbles for the experiment are shown in Figure 51. The LED illuminate provided enough light to easily differentiate and segment the bubbles from the background. The maximum velocity magnitude of the jet (Re = 11900) in the volume of interest was approximately 4.0 m/s. Particles were sparsely seeded into the flow field (~300 particles per frame) to allow long trajectories to be successfully tracked through the length of the jet. A sample of the reconstructed particle trajectories colored by velocity and acceleration are shown in Figure 51 and Figure 52 respectively. The particles took approximately one second to traverse the volume of interest and the longest trajectories were on the order of 120 frames.

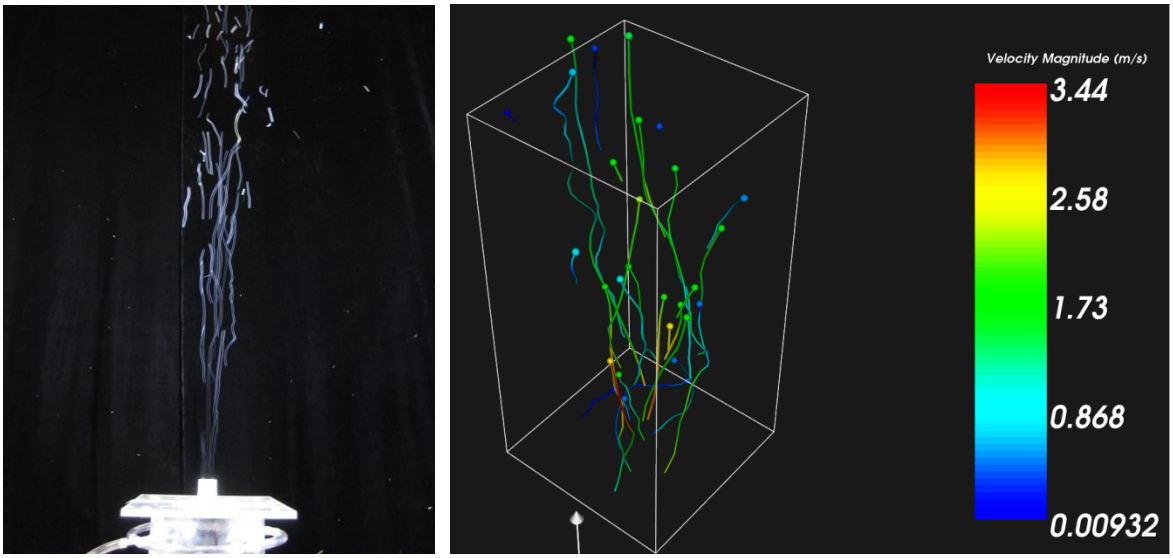

Figure 51: Trajectory visualization in the volume of interest around a round jet colored by velocity magnitude, Re = 11900, x/d = 15 - 33

The statistical accumulator grid was collecting all instantaneous particle velocity and accelerations over the twenty minute experiment resulting in an average of greater than a thousand observations for each cell in the shear and center line layers of the jet Figure 53. The

resulting mean velocity magnitude in the XY plane is shown in Figure 53. Symmetry about the jet core is clearly visible along the axial direction.

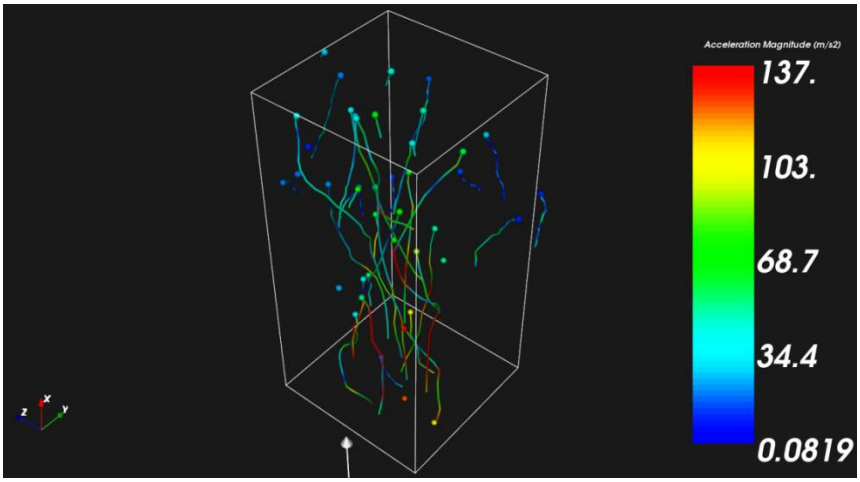

Figure 52: Trajectory visualization in the volume of interest around a round jet colored by acceleration magnitude, Re = 11900, x/d = 15 to 33

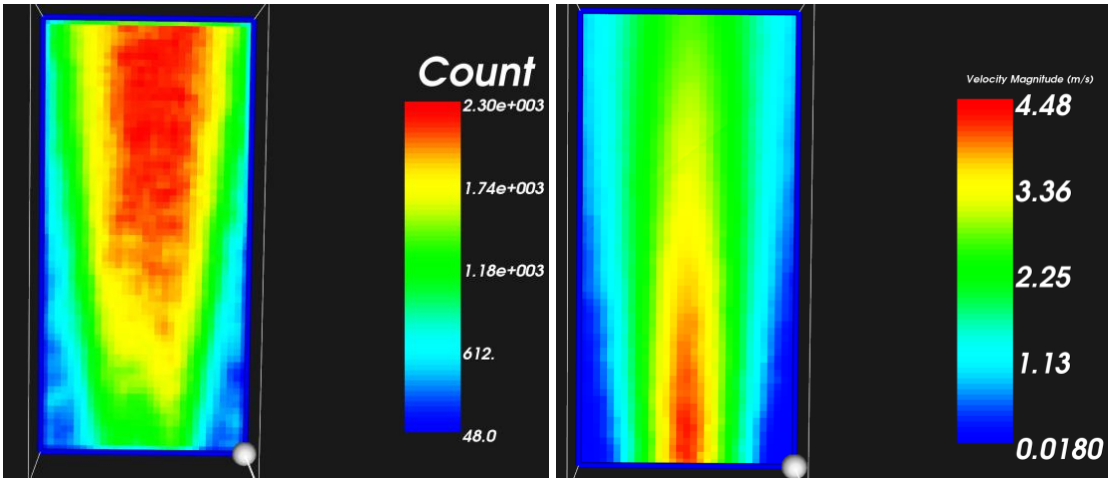

Figure 53: (left) Number of observations per cell in the XY plane, (right) mean velocity magnitudes in the XY plane; z = 0; x/d = 15 to 33

The axial centerline mean velocity profile (Figure 54) was normalized by the jet exit velocity (U_j) in order to compare with published hot-wire anemometry based studies as shown in Figure 55. The observation starts in the intermediate range at x/d = 15 and it is clear that the jet is still developing until approximately x/d = 25 as the center line velocity decay rate becomes constant. The decay constant B from equation 6-8 for the range of $25 \leq x/d \leq 33$ was 6.39 which is in good agreement with the range of 5.7 to 6.7 from previous studies reported by Fellouah (Fellouah et al., 2009).

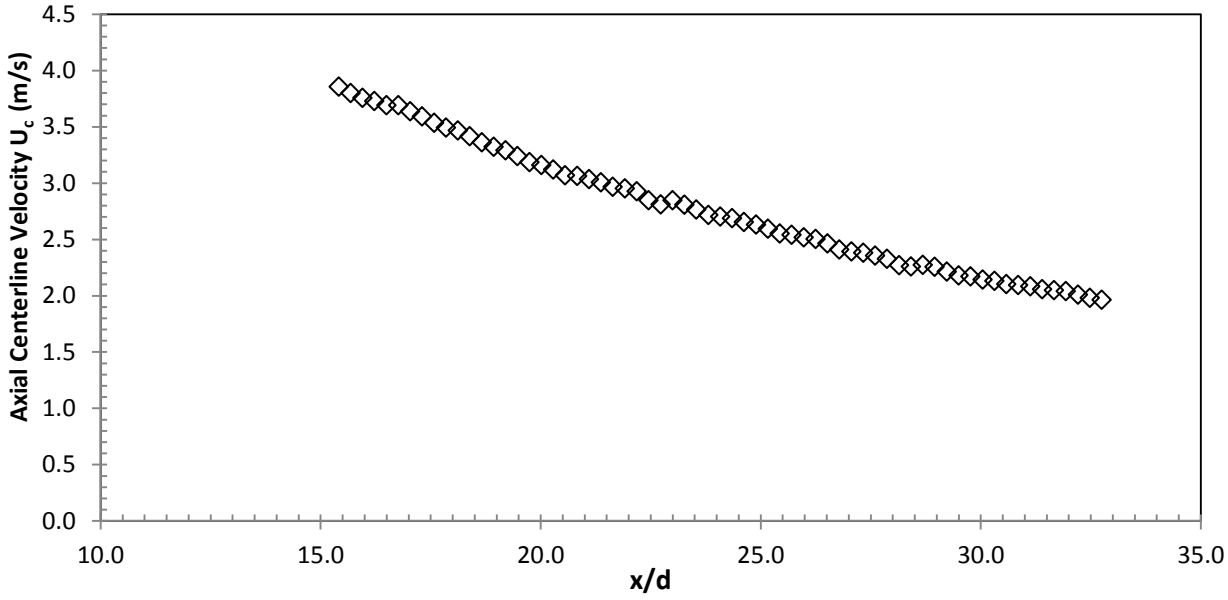

Figure 54: Axial centerline mean velocity distribution

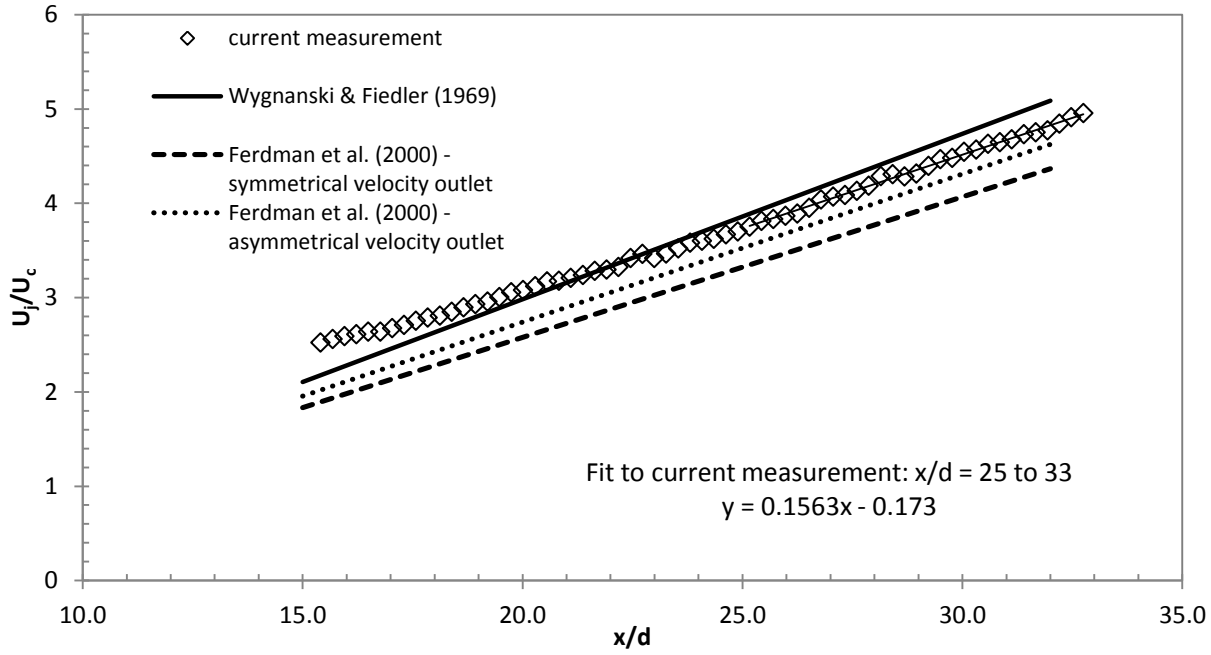

Figure 55: Normalized axial centerline mean velocity compared to predicted values from equation 6-8

The mean transverse profiles of axial velocities in the XY and XZ planes are shown in Figure 56 and Figure 57. The predicted Gaussian distributions, based on equation 6-6 where A = 75.2 for a round turbulent jet, are also plotted for y/d = z/d = 16, 25, and 32. The transverse axial velocity profiles of the current measurement are in very good agreement with the predicted

Gaussian distribution for a round jet (Xu & Antonia, 2002). The profiles at x/d = 16 deviate the most as expected since the flow is assumed to be still developing and has not reach a state of self-similarity. The profiles at x/d = 32 follow the Gaussian distribution closely for both the XY and XZ planes, where there is little difference between the two planes.

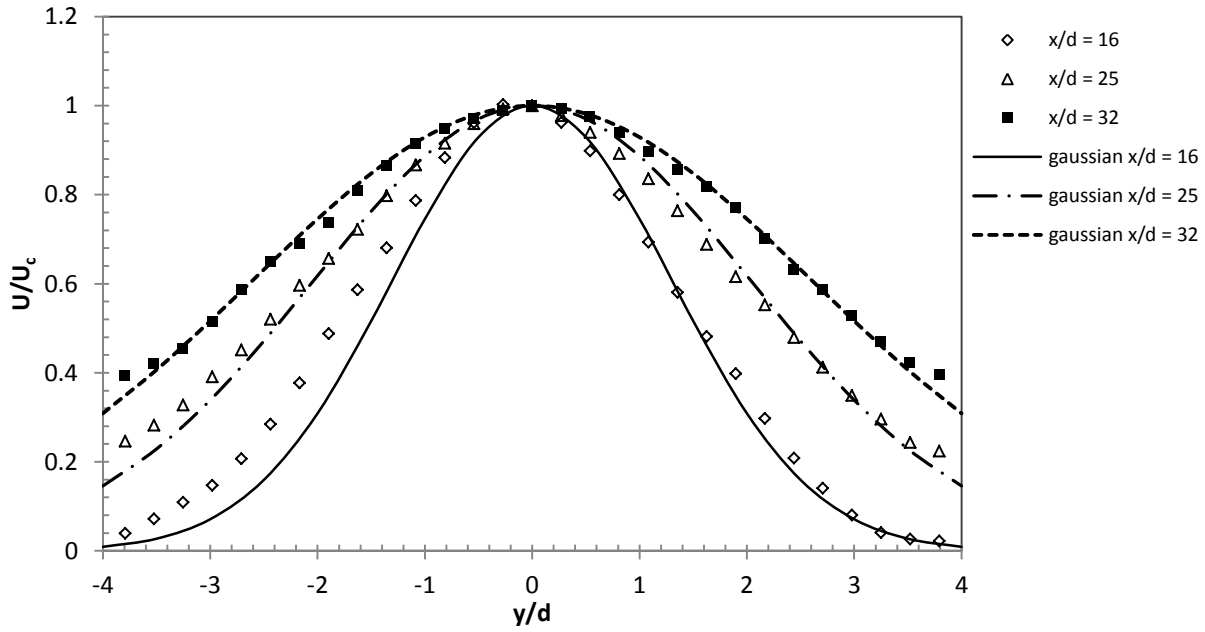

Figure 56: Transverse profiles of axial velocity on the XY center plane

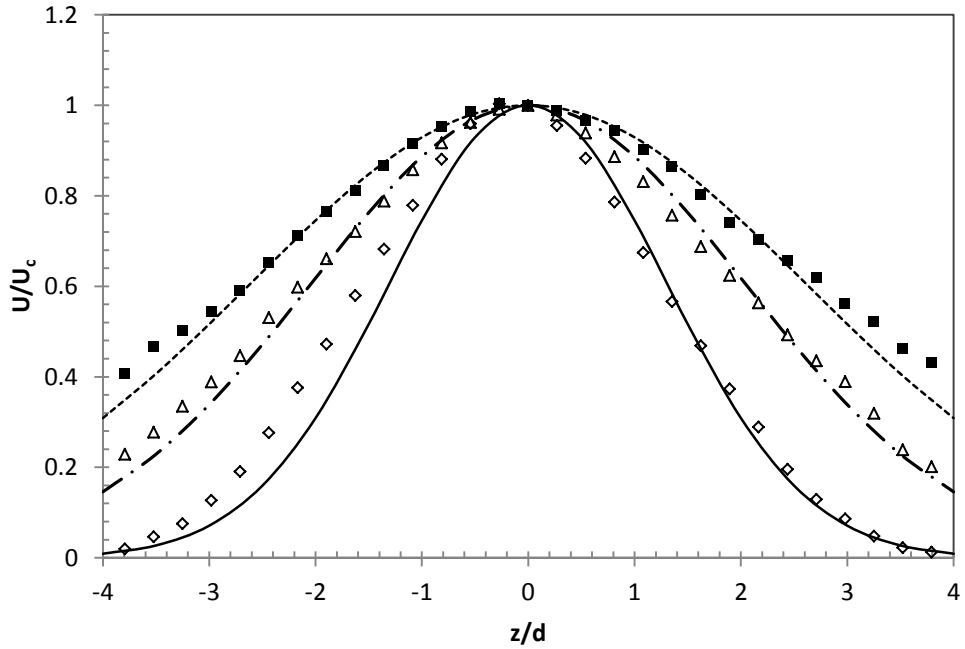

Figure 57: Transverse profiles of axial velocity on the XZ center plane (legend in Figure 56)

The transferse profiles of axial velocity were normalized by the axial position and plotted in Figure 58 and Figure 59 for the XY and XZ center planes respectively. The three profiles for x/d = 16, 25, and 32 all appear on top of each other, which supports the observation of self-similarity. Again, there is slight deviation for x/d = 16 where the flow is still developing.

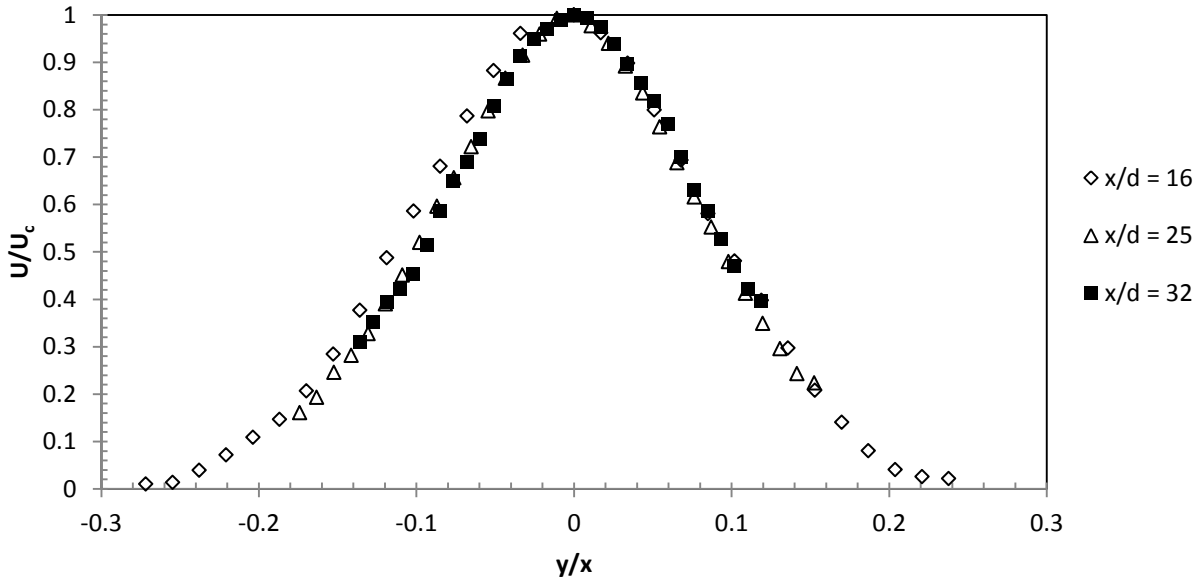

Figure 58: Axial velocity profiles with transverse coordinate normalized by the axial distance on the XY center plane

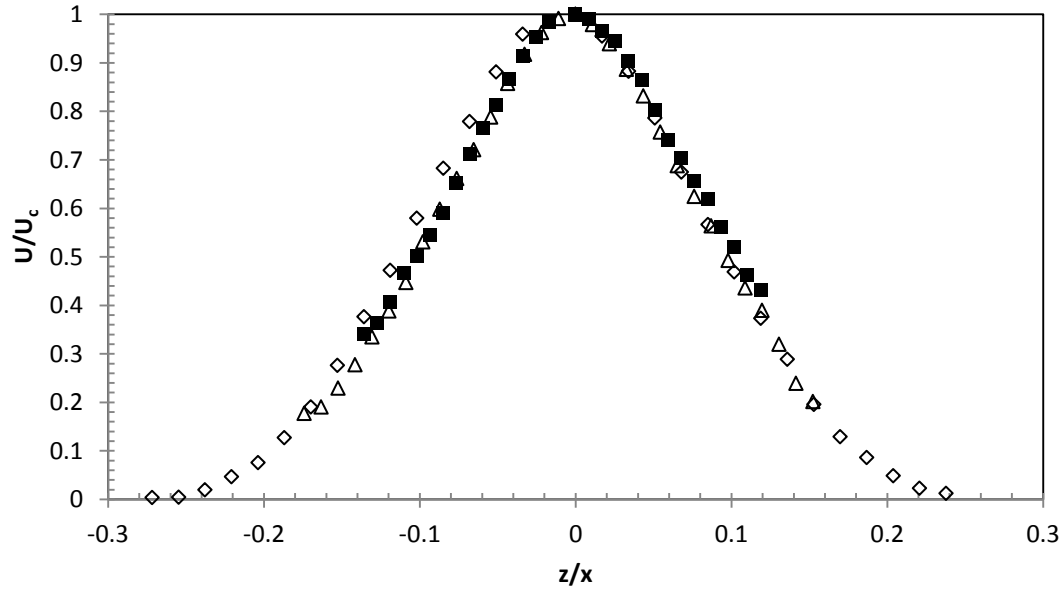

Figure 59: Axial velocity profiles with transverse coordinate normalized by the axial distance on the XZ center plane

The turbulent kinetic energy distribution for the XY plane at z = 0 is shown in Figure 60. The maximum turbulent kinetic energy in the observed domain was located at the center and nearest to the jet. The turbulent kinetic energy quickly dissipates and spreads as the distance from the jet increases and progresses towards fully developed turbulent flow.

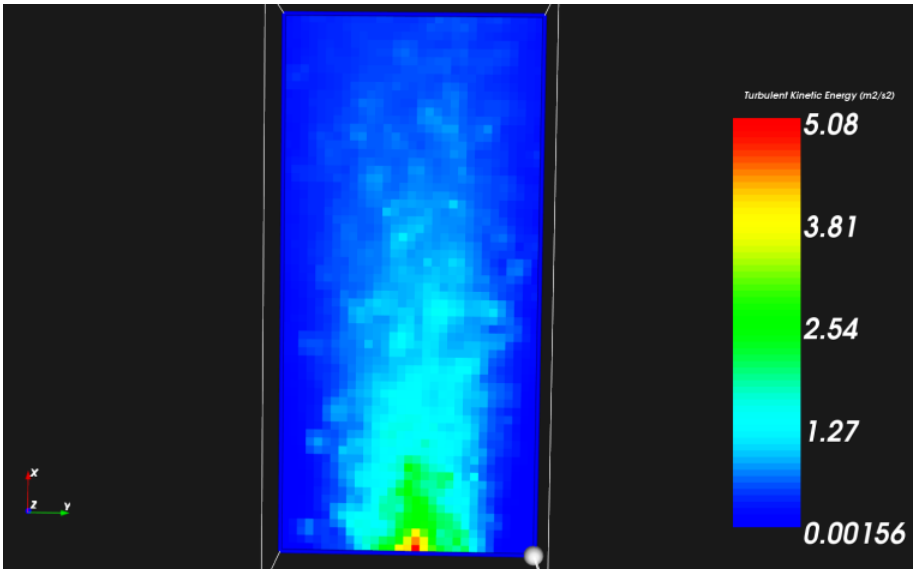

Figure 60: Measured turbulent kinetic energy distribution in the XY plane, z = 0; x/d = 15 – 33

The stream wise turbulence intensity transverse profiles for the XY and XZ planes are shown in Figure 61. The maximum stream wise turbulence intensity observed between $15 \leq x/d \leq 33$ is approximately 0.3 which corresponds well to the values observed in hot-wire anemometry experiments as described by Ferdman (Ferdman et al., 2000) . The profiles are symmetric about the centerline in both the XY and XZ planes, and spread out as x/d increase. At x/d = 32 the mean turbulence intensity is approximately 0.23 for $-4 \leq y/d \leq 4$. The peak axial turbulence intensity occurs at the center line for x/d = 15 and spreads into a dual peak on either side of the centerline for x/d = 32. The observation of a slight local minimum of axial turbulence intensity at the center line in the intermediate region is consistent with other researchers' findings (Fellouah et al., 2009; Ferdman et al., 2000). A local maximum axial turbulence intensity of 0.29 can be seen at z/d = 2.6 in the XZ plane at x/d = 32. This does not fit the expected profile and is possibly due to tracking errors as will be discussed below.

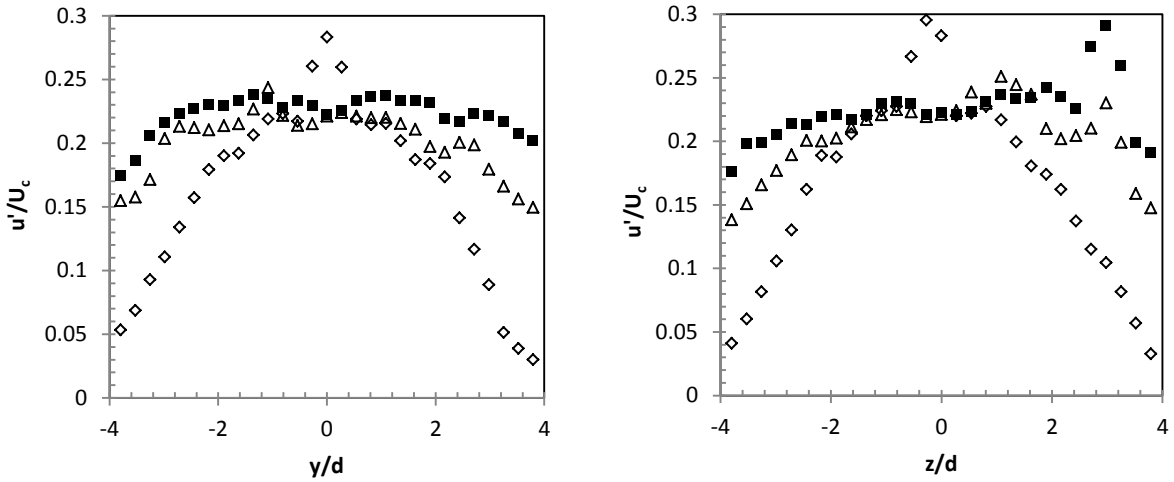

Figure 61: Axial stream wise turbulence intensity profiles for (left) XY plane and (right) XZ plane

The transverse turbulence intensity (as scaled by the axial mean centerline velocity U_c) is shown in Figure 62 for the XY and XZ center planes. The transverse turbulence intensities are slightly less than the axial turbulence intensities but follow the same general profiles in the transverse and axial directions. Symmetry about the centerline and linear spreading of the profiles with axial distance is observed as in the case of axial turbulence intensity. This is similar to the observations by Ferdman (2000) and Fellouah (2009). The plots of out-of-plane turbulence intensities are given in Appendix A.

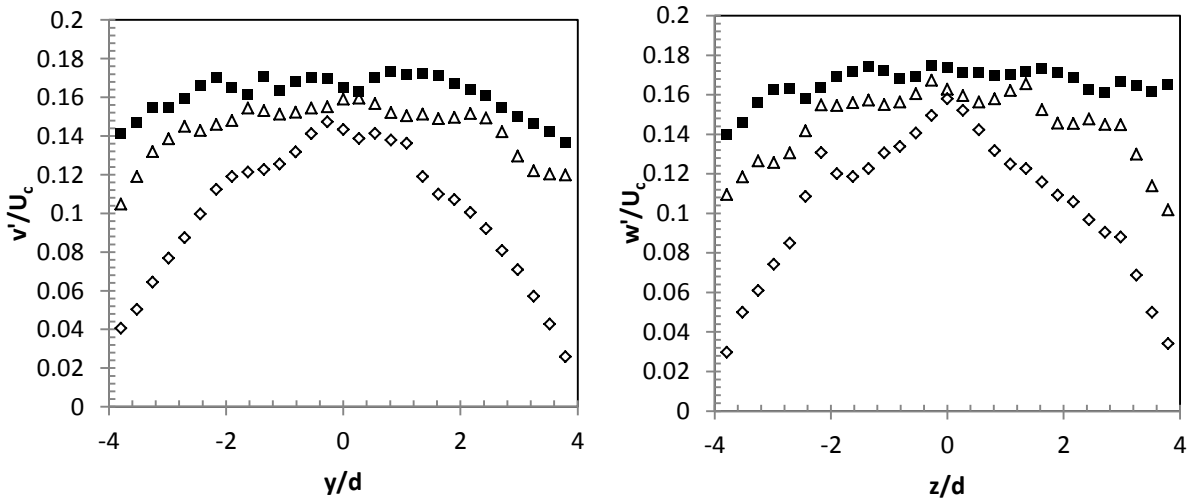

Figure 62: Transverse turbulence intensity profiles for (left) XY plane and (right) XZ plane

The Reynolds shear stresses $\overline{u'v'}$ in the XY plane and $\overline{u'w'}$ in the XZ plane are normalized by the axial centerline velocity squared and plotted in Figure 63. The transverse profiles show anti-symmetry about the centerline and have peaks near y/d = z/d = -2 and 2. The maximum normalized shear stress is on average between 0.015 and -0.015 and nearly identical for the XY and XZ planes. The shear stress grows in magnitude as x/d increases indicating a transfer of turbulent energy into the shear layer. In magnitude, the observed Reynolds shear stresses are consistent with those found by other researchers (Fellouah et al., 2009; Ferdman et al., 2000; Xu & Antonia, 2002).

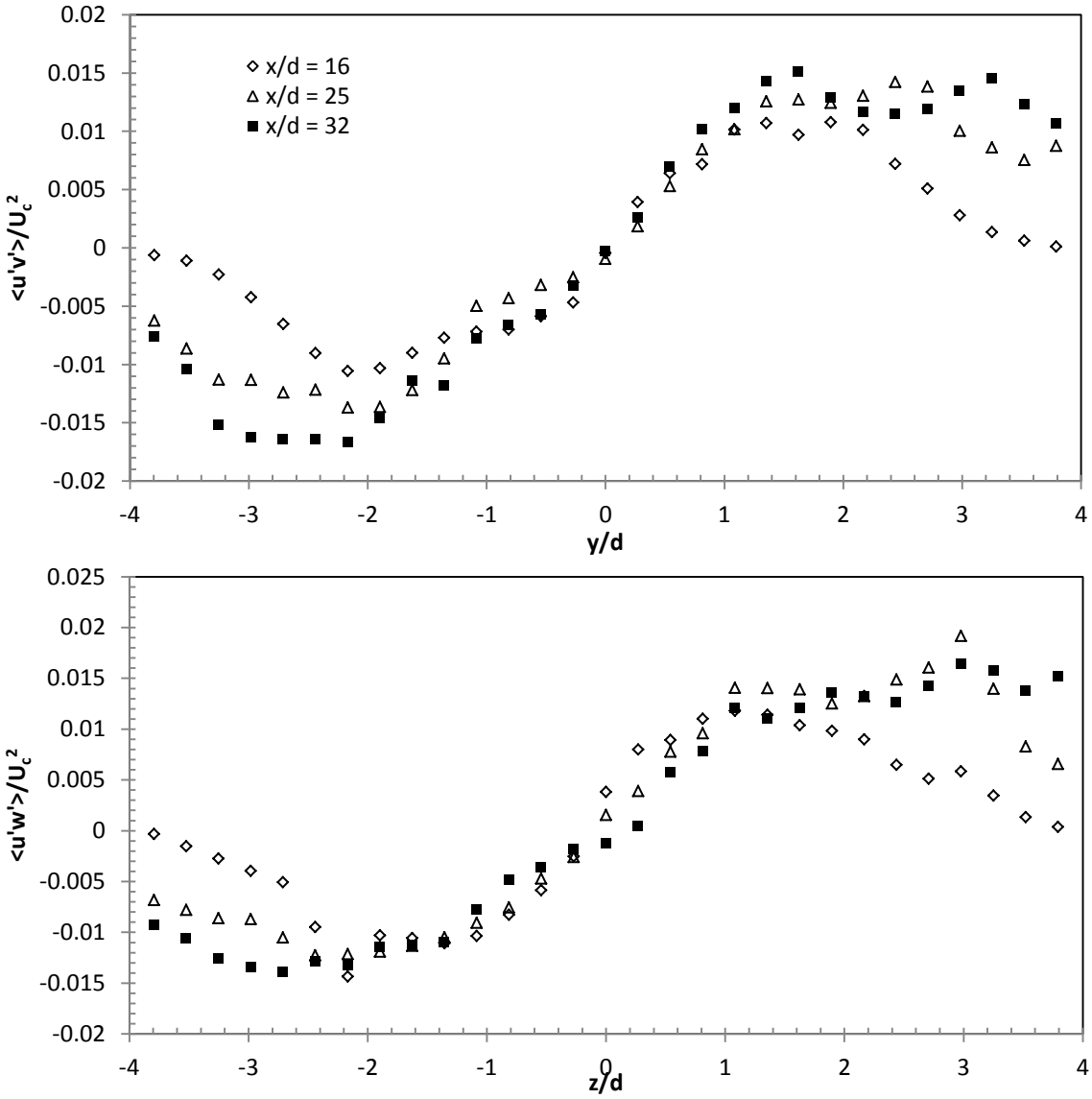

Figure 63: Reynolds shear stress profiles for (top) XY plane and (bottom) XZ plane

6.5.4 **Discussion and summary**

The goal of this experimental analysis was to measure a well characterized flow field using the new particle tracking system and compare the results with those from other researchers. For round turbulent jet flow, four main characteristics could be validated 1) Linear decay rate of mean centerline axial velocity, 2) Gaussian axial velocity profiles in the transverse direction, 3) Axial turbulence intensities on the order of 0.25 to 0.33, 4) Anti-symmetric Reynolds shear stress profiles about the center line with amplitudes the order of 0.015 as normalized by the center line axial velocity squared.

The results agreed well with all four of these flow characteristics. The mean centerline axial velocity decay in the observed volume was linear between $15 \leq x/d \leq 33$, and produced a decay constant B = 6.39 which is within the expected range of 5.7 - 6.7 corresponding well to observations by other researchers. The axial velocity distribution over the transverse directions also matches well with the predicted Gaussian profiles (equation 6-6). The best fit to the Gaussian distribution given was observed at observed transverse plane x/d = 33 from the nozzle indicating that the flow was reaching a fully developed self-similar state. The axial velocity profiles at x/d = 16 deviate the most as expected since the flow is assumed to be still developing and has not reach a state of self-similarity.

The turbulence intensities also correlated well with the observations by other researchers in terms of distribution and amplitude. The maximum axial velocity turbulence intensity observed at x/d = 15 was 0.30 which matched those reported by Ferdman (2000) who suggested an accepted range between 0.24 and 0.35 for round turbulent jets. The Reynolds shear stress terms normalized by axial mean centerline velocity squared were also within the range observed by other researchers of $-0.02 \leq \overline{u'v'}/U_c^2 \leq 0.02$. The transverse profiles of these shear stresses were anti-symmetric about the centerline and zero at the axis which fits the classical understanding of axisymmetric round turbulent jet flow.

Looking at the plot in Figure 61 of axial stream-wise turbulence intensities, it appears that several tracking errors (false trajectory matches) have occurred leading to larger velocity variances than are predicted by jet flow theory. This is manifest as a secondary peak in the turbulence intensity profile in the shear region of the flow. The location of this peak, in the

positive Z side of the central jet core, is on the opposite side of the core from the cameras. Therefore the tracking errors may be explained by the fact that this point would suffer the most from particle image overlaps and thus more likely contain tracking errors than other locations. A solution would be to spread the cameras more around the flow or install more cameras to better observe the back side of the jet.

The following conclusions can be made from the experimental analysis of round turbulent jet flow:

- The Lagrangian particle tracking system was demonstrated to successfully detect and track hundreds of particles over hundreds of frames in real-time at 120 fps through a three dimensional turbulent air jet flow.

- The evidence of comparing particle tracking system measurements of a turbulent round jet with those of hot-wire anemometry measurements found in literature support the conclusion that the system is able to accurately track particles in turbulent flow and utilize the Lagrangian velocity statistics of the particles to characterize a flow field of the type studied.

- The use of neutrally buoyant helium filled soap bubbles, which behaved as inertial particles, did provide reasonable velocity averages and variances for the jet flow studied. Therefore our data agrees with the conclusions by (Bourgoin et al., 2011) and (Ouellette et al., 2008) that velocity statistics of turbulent flow are not impacted significantly when viewed through filter of inertial particle motion.

- The spatial distribution of the velocity statistics appears to match reasonably well with published measurements of turbulent jet flow. Therefore the use of neutrally buoyant helium filled soap bubbles was suitable as flow tracers for the type of analysis conducted here (round turbulent jet flow in the intermediate to far field flow regime, Reynolds number = 11900 based on jet diameter).

6.6 Application to unconfined forced vortex flow

6.6.1 Objectives and approach

The main objective of this experimental analysis is to apply the particle tracking system to a stationary forced vortex (swirling) flow field and test the capability to track particles with more complex 3D trajectories. The secondary objective is to evaluate the turbulence intensities, Reynolds shear stresses and static pressure field calculations. Two static pressure calculation methods will be compared including a) Reynolds Average Navier-Stokes (RANS) method and b) Instantaneous Lagrangian acceleration (ILA) method.

The general approach is to develop an unconfined forced vortex flow and evaluate the particle tracking systems ability to track particles as the trajectories traverse the volume with mean velocity components in all three dimensions. This type of flow is likely axisymmetric and therefore observations of symmetry in two perpendicular center planes (XY and XZ) will indicate that the particle tracking system is consistent in tracking particle through complex 3D motions. Asymmetries will likely indicate tracking errors or poor ability to track particles due to overlapping and obstruction.

Unconfined forced vortex flow is of particular interest in the area of fluid dynamics as it has many applications in industrial processes including aerosol separation in pollution control equipment (uniflow and reverse flow cyclones) and trapped vortex combustion in Integrated Gasification Combined Cycle (IGCC) plants. Whereas the round turbulent jet flow field consisted of a mean axial velocity and near zero mean velocities in the other two dimensions, the unconfined forced vortex has mean velocity components in all three dimensions. Therefore it will be a more challenging test for the real-time Lagrangian particle tracking system and provide insights into its ability to characterize complex turbulent flow fields.

6.6.2 Methodology

An unconfined forced vortex was created with the vane section from a uniflow cyclone (Y. Zhang, 2005) as a swirl generator, which was placed inside the 0.1143 m (4.5 in) ID acrylic tube. The discharge angle of the vanes was 60 degrees from the vertical (X) and the diameter of the inner pipe to which the vanes were affixed was 0.0508 m (2.0 in).The air flow was supplied

by a 150 W 2700 rpm variable speed vane axial fan. The flow was passed through a 10.5 cm long flow straightener made from drinking straws and then into a 12.0 inch long 4.5 inch ID acrylic tube with the swirl generator located immediately after the flow straightener. The experimental apparatus and swirl generator are shown in Figure 64.

The flow rate was set to 0.026 m^3/s which lead to a velocity magnitude of 7.4 m/s exiting which was measured by traversing a hot-wire anemometer over the outlet of the vanes, with the anemometer oriented parallel to the vane (TSI Model 8330). Since the flow exit condition was essentially swirling annular flow, the hydraulic diameter (d_h = 0.0635 m [2.5 inch]) was used for all length normalizations. The Reynolds number was 31000 based on the velocity magnitude and the hydraulic diameter at the outlet. The frame rate of the cameras was set to 120 fps and the experiment was run for 25 minutes to observe the stationary turbulent vortex. The tracer particles were 1 to 4 mm diameter neutrally buoyant Helium filled soap bubbles generated through an SAGE Action Model 5 bubble generator. The bubbles were injected into the flow field downstream of the flow separator. The resulting flow field seeded with bubbles can be seen in Figure 65.

The six camera system was arranged in hexagon pattern of approximate side length of 0.5 m with all cameras angled towards a central point roughly 1.0 m from the lenses. The volume of interest was a 300 x 300 x 206 mm, which was much less than the overall observable volume. The volume of interest was discretized by a virtual finite volume grid containing 45 x 45 x 31 cubic elements of side length 6.67 mm placed 0.258 m in the positive X direction from the tube exit. The finite volume grid contains statistical accumulators at each cell center for particle velocity, acceleration, shear stresses, and static pressure gradients.

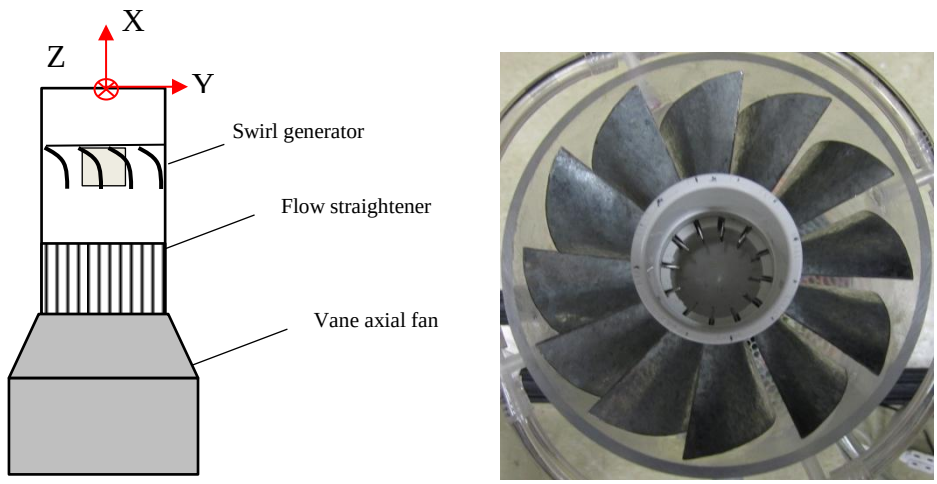

Figure 64: Apparatus for the forced vortex flow experiment

6.6.3 **Results**

The particle tracking system was able to track up to approximately 500 particles per time step ($\Delta t = 1/120$ s) and visualize their trajectories in real-time. A screen shot from the experiment showing particle velocity and acceleration are shown in Figure 66. The highest velocity magnitude of a particle in the volume of interest during the experiment was approximately 3.5 m/s. The average trajectory was approximately 100 frames long. Long trajectories could be tracked in the vortex flow and trajectories could be seen wrapping around the core as shown in Figure 66. The distributions of mean velocity magnitudes at the end of the experiment are shown in Figure 67 for the YZ plane looking at the vortex from above and Figure 68 for the XY plane which views the vortex from the side.

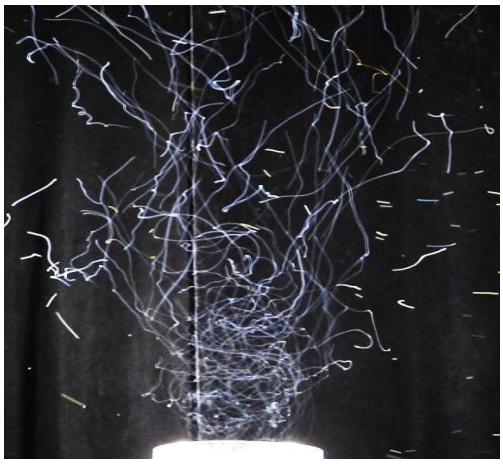

Figure 65: Trajectories of neutrally buoyant helium bubbles in the forced vortex flow field

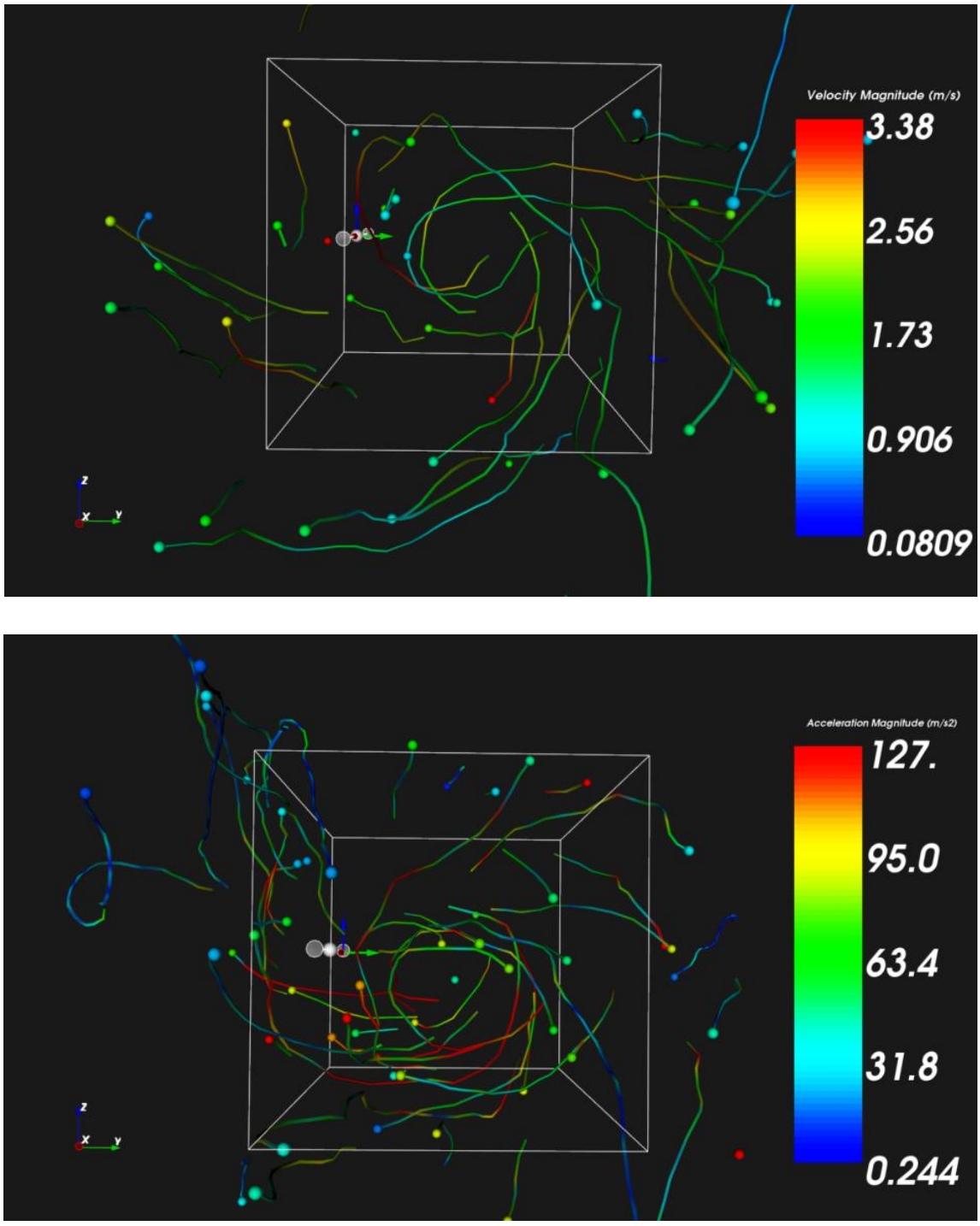

Figure 66: Particle trajectory visualization, from above (ZY-plane), during forced vortex flow experiment color by velocity (top) and acceleration (bottom)

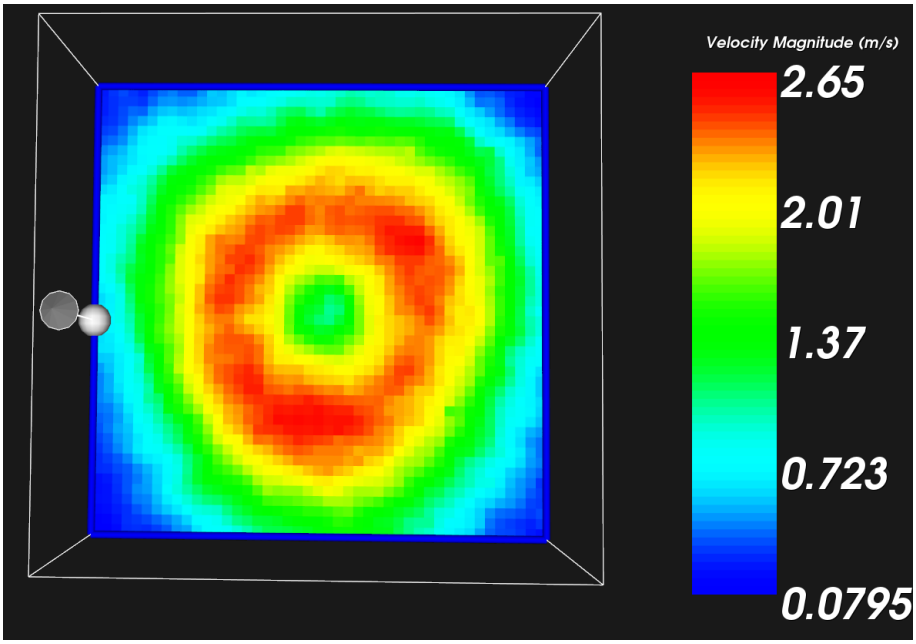

Figure 67: Distribution of mean velocity magnitude in the YZ plane spanning 4.6 d_h in each direction, x = 4.2 d_h

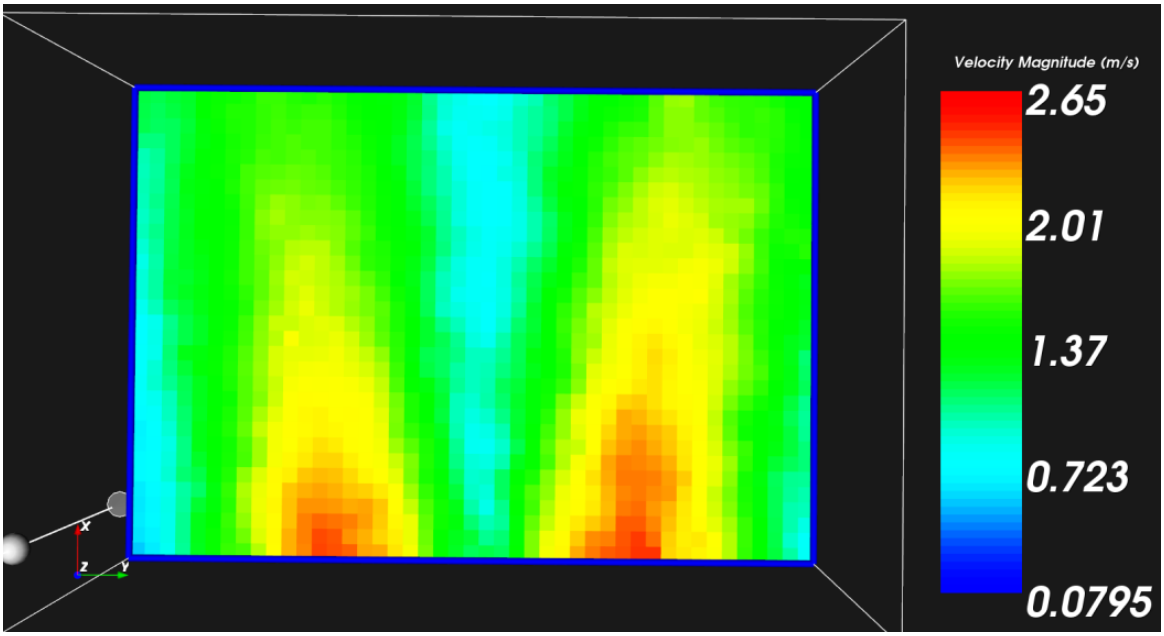

Figure 68: Distribution of mean velocity magnitude in the XY plane, z = 0

The central core of the vortex is clearly seen as a region of lower mean velocities surrounded by a ring of high mean velocities. This profile is shown in greater detail in the graphs of mean velocity in the X-direction, u, for two perpendicular planes (XY and XZ) at the center of the domain in Figure 69. The profile is nearly symmetrical and peaks at about 2.1 m/s at x/d_h = 4.12

on each side of the vortex core. The axial velocity decreases with distance from the vortex outlet and the profile begins to flatten out as energy is dissipated. The flow appears to be axisymmetric as expected, and the XY and XZ profiles are in good agreement.

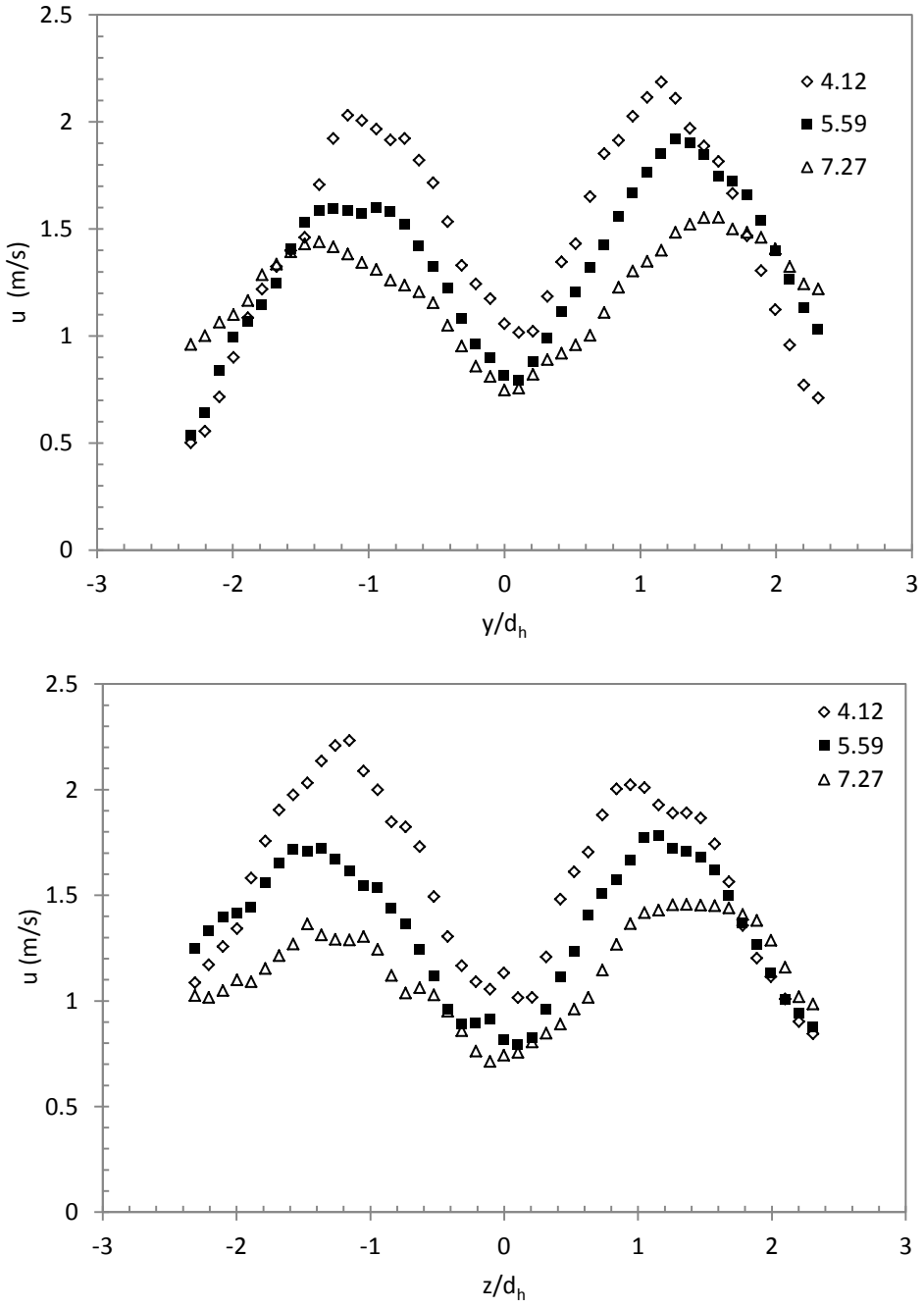

Figure 69: Mean X-velocity, u, in the XY center plane (top) and XZ center plane (bottom), x/d_h = 4.12, 5.59 and 7.27, d_h= 63.5 mm

The tangential velocity profiles, Z-component, w, as viewed from the XY center plane and Y-component, v, as viewed from the XZ center plane, are shown in Figure 70. The tangential velocity profiles are anti-symmetric about the centerline and good indicators of the location of the vortex core as they exhibit very sharp gradients near the centerline. Again, the XY and XZ center plane profiles are in agreement and the peak tangential velocity at $x/d_h = 4.12$ is shown to be about 1.5 m/s on all four sides of the vortex in the two perpendicular planes.

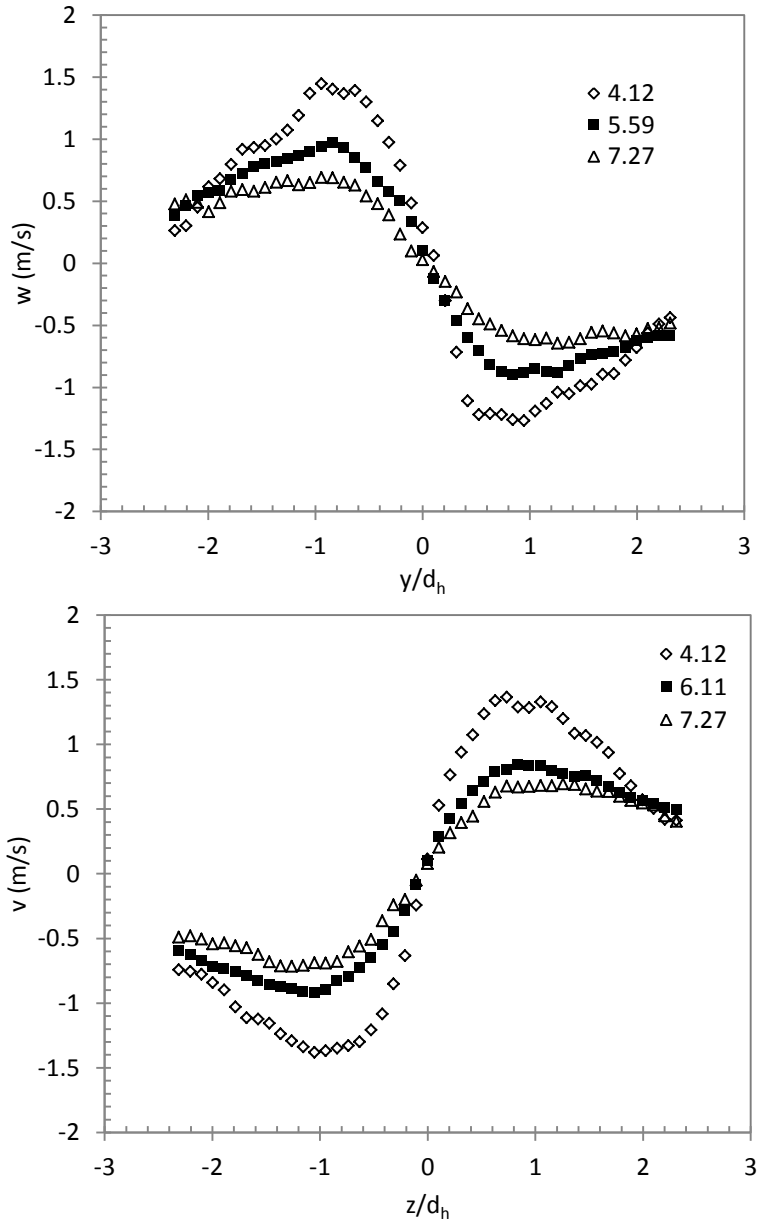

Figure 70: Mean tangential velocity *w* in the XY center plane (top) and *v* in the XZ center plane (bottom), $x/d_h = 4.12$, 5.59 and 7.27, d_h= 63.5 mm

The turbulent kinetic energy was calculated from the variances of u, v, and w and is shown for the XY center plane in Figure 71. The turbulence is highest in the vortex core closest to the outlet and appears to dissipate quickly as the flow progresses to the top of the domain. The turbulent kinetic energy distribution is shown in more detail for the XZ plane in Figure 72. Again, symmetry about the center line can be seen as the turbulent kinetic energy peaks at $z/d_h = 0$ and maintains symmetry throughout the observation volume.

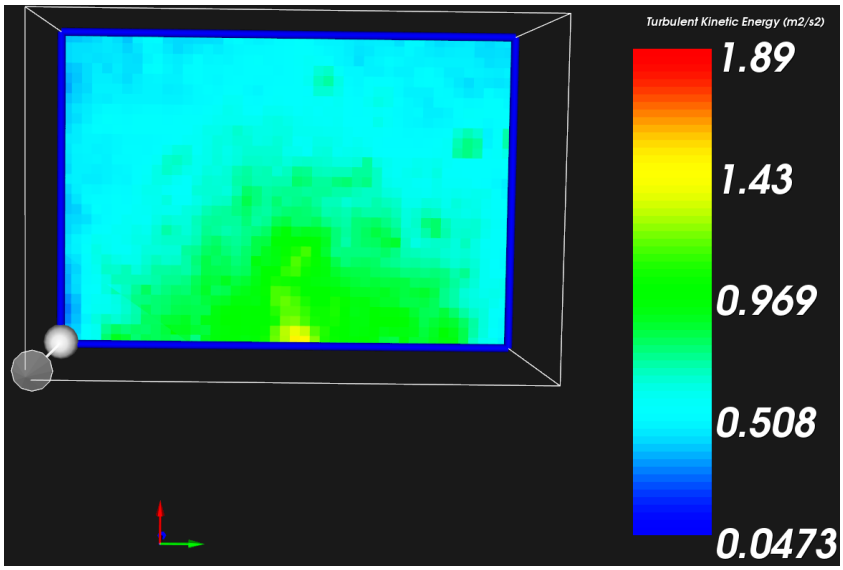

Figure 71: Turbulent kinetic energy distribution in the XZ center plane

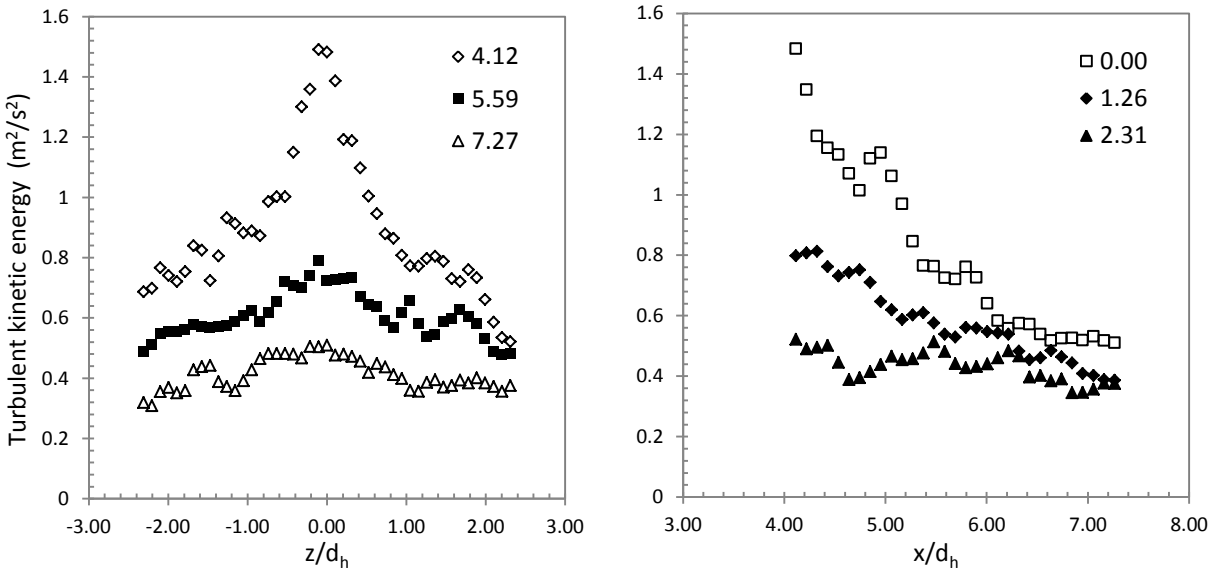

Figure 72: Turbulent kinetic energy (left) profile in XZ plane at x/d_h = 4.12, 5.59 and 7.27, (right) axial profile for z/d_h = 0.00, 1.26 and 2.31

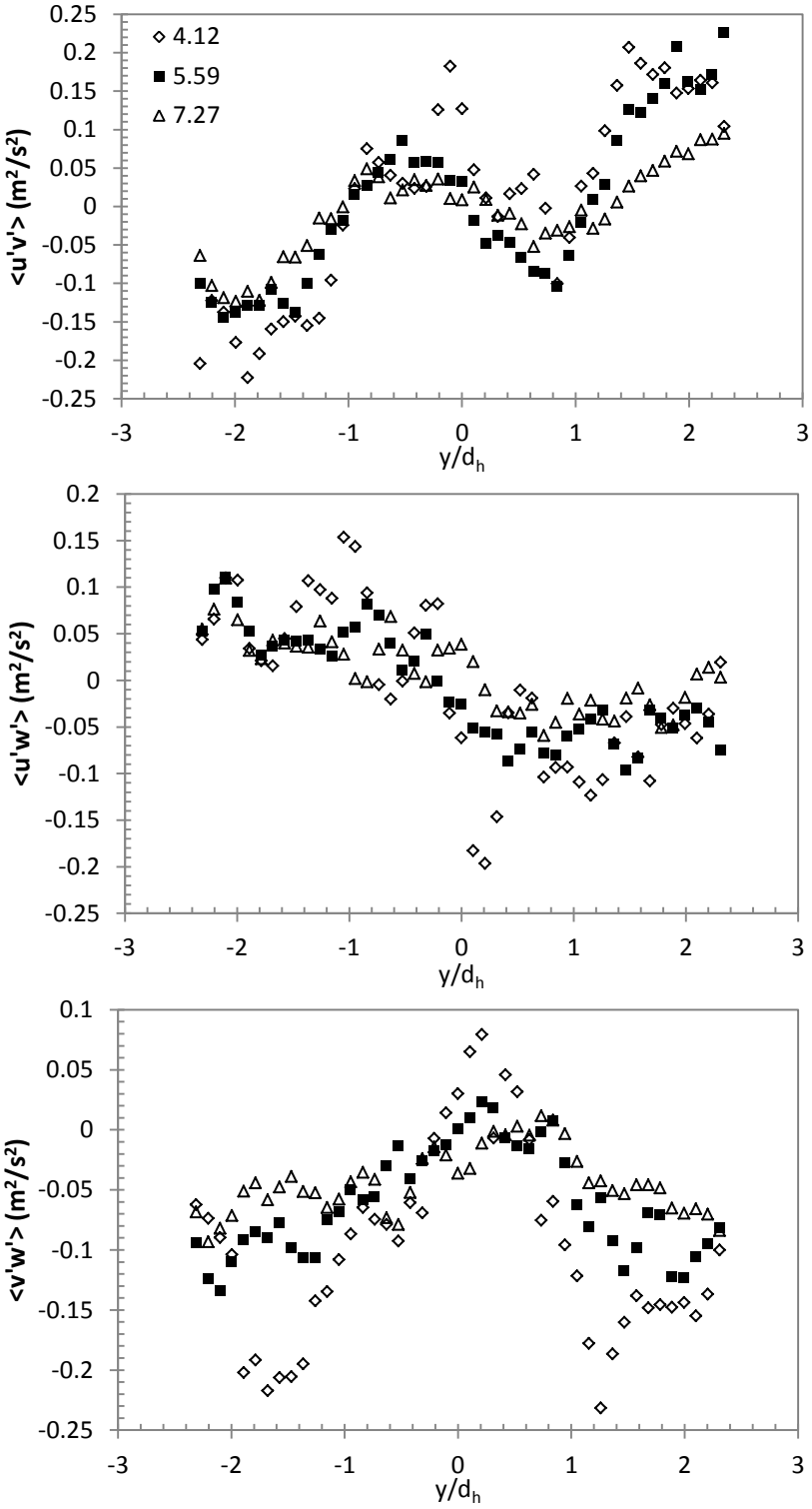

Figure 73: Reynolds shear stress profiles in the XY center plane at x/d_h = 4.12, 5.59 and 7.27

The Reynolds shear stress terms (Figure 73) also appeared to display symmetry about the vortex core and were roughly on the same order of magnitude for all three components, only varying in distribution. The $\overline{u'v'}$ and $\overline{u'w'}$ profiles exhibited anti-symmetry about the core which is more apparent for larger x/d_h. The $\overline{v'w'}$ profile peaks at the core center for $x/d_h = 4.12$ but is nearly zero at the center line farther downstream. The Reynolds shear stress plots for the XZ plane are given in Appendix B for comparison.

The static pressure distribution in the flow field was calculated based on the collected Lagrangian velocity and acceleration statistics as described in section 3.6 using two methods. The first method applied was the Reynolds Averaged Navier-Stokes (RANS) method. In this method the RANS equations are simplified by applying the steady state assumption to force the Eularian acceleration term to zero and then solving for the pressure gradient term. All six components of the Reynolds stress tensor are known from measurement, along with the mean velocities in each direction. The pressure gradients at each element in the virtual finite volume grid are then calculated using a spatial marching finite difference scheme. The second method is the Instantaneous Lagrangian Acceleration method (ILA). In this method the instantaneous Lagrangian acceleration of a particle is used to solve the Navier-Stokes Equations for the instantaneous pressure gradients around the particle. The instantaneous pressure gradients are collected and averaged for each element in the virtual finite volume grid.

Both methods resulted in a pressure gradient field (x, y, z pressure gradients stored in each element) at the end of the experiment. The pressure gradients were applied as source terms to the pressure Poisson equation and the iterative Successive Over Relaxation (SOR) method was used to solve for pressures at each element. The static pressures were found relative to a corner element in the domain. The results from the static pressure calculations are shown in Figure 74. Both methods calculate low pressures in the center of the vortex compared with the rest of the domain as theory predicts. The peak negative pressure is near -4 Pa for both methods at $x/d_h = 4.12$. The major difference between the two is the pressure profiles farther downstream and overall smoothness of the distribution. While the overall shape still indicates a low pressure core, the RANS method's profile for $x/d_h = 7.27$ become slightly asymmetrical and remains negative while the ILA method yields pressures in the 1 Pa range at the farthest observed point downstream.

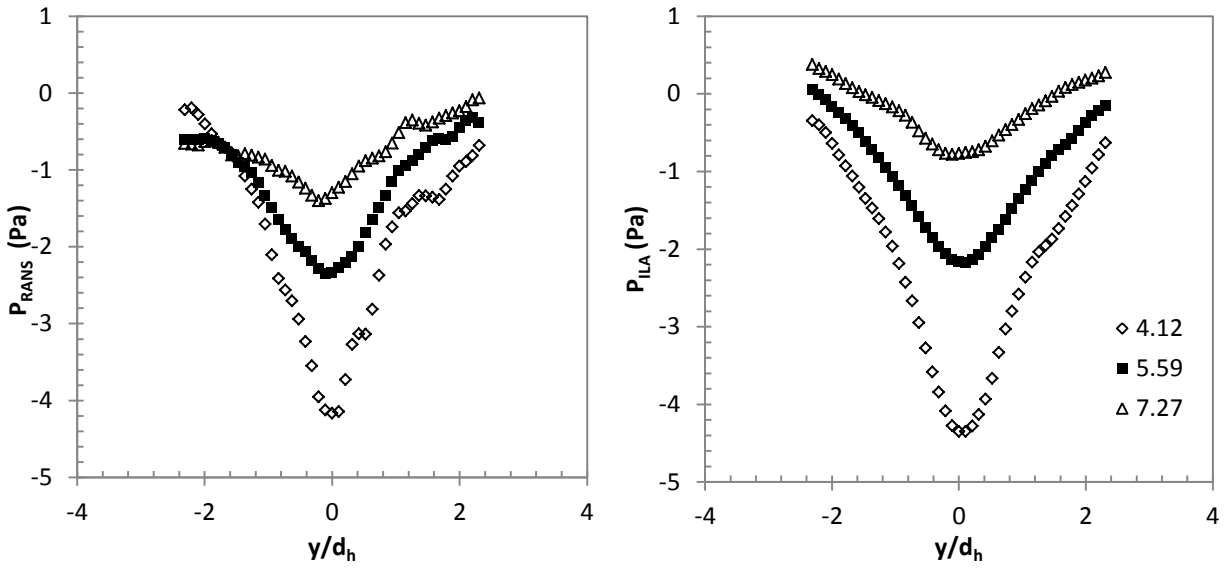

Figure 74: Calculated static pressure on the XY center plane at x/d_h = 4.12, 5.59 and 7.27 based on two methods (left) Reynolds Averaged Navier-Stokes (RANS), method; (right) Instantaneous Lagrangian Acceleration (ILA) method

6.6.4 Discussion and summary

The particle tracking system was able to reliably track particles in the unconfined forced vortex flow field as supported by the nearly symmetric mean velocity profiles observed. The axial and tangential velocity profiles appear to be a qualitative match to what would be theoretically predicted for a vortex flow field. The axial velocity profile has two peaks on either side of a vortex core, while the tangential velocity is anti-symmetric about the center line of the core. The velocities of the particles where consistent in all three dimensions when compared in the XY and XZ planes indicating that the system can faithfully track particles as they traverse all three dimensions in complex paths.

The Reynolds stress terms $\overline{u'^2}$, $\overline{v'^2}$, $\overline{w'^2}$ and $\overline{u'v'}$, $\overline{u'w'}$ and $\overline{v'w'}$ of the particle field were plotted and showed signs of symmetry about the vortex core. The $\overline{u'v'}$ shear stress profile in the XY plane is particularly interesting as it exhibits anti-symmetry about the center line with two local peaks (positive and negative) on each side of the vortex. This pattern is closely approximated by a fifth order polynomial as shown for x/d_h = 5.59 in Figure 75. The $\overline{u'v'}$ shear stress reaches zero three times in the profile, at -1.1d_h, 0d_h and 1.2d_h. These locations of zero $\overline{u'v'}$ shear stress nearly line up with the mean axial velocity (u) peaks as shown in the figure. This observation is

congruent with the theory of forced vortex flow where shear equals zero in regions of fixed body rotation.

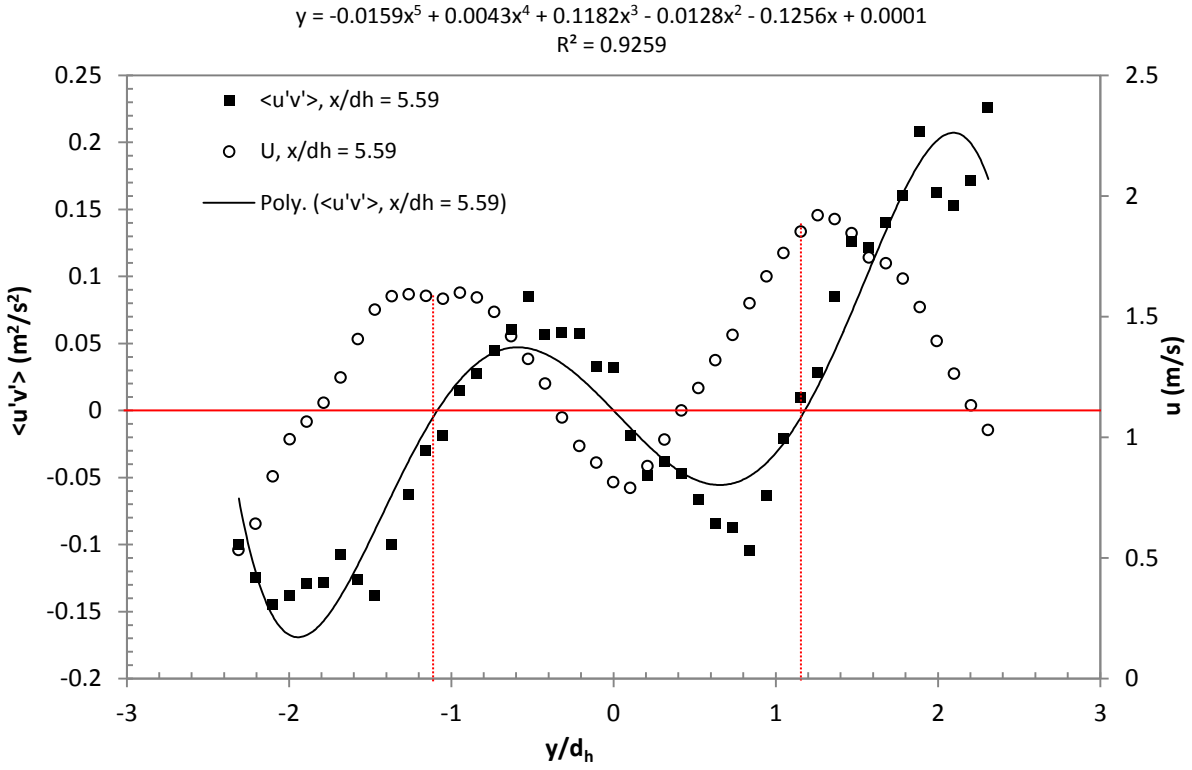

Figure 75: Reynolds shear stress <u'v'> and mean axial velocity <u> profiles in the XY center plane at $x/d_h = 5.59$

The turbulent kinetic energy was a maximum in the core region, which may be supported by the following observation. While conducting the experiment it was observed that the flow was pulsating and generating a steady acoustic signal. The pressure fluctuation could be felt by placing a hand in the flow field above the swirl generator. This pulsing is likely the result of the vortex core continuously collapsing due to the low pressures created at the center and high tangential velocities in the surrounding fluid. This would explain the peak of turbulence at the core where the mean velocity magnitudes are low. The static pressure calculations, although not validated for accuracy, qualitatively support this theory as the pressure profiles found contained sharp gradients towards a local minimum at the center of the vortex core.

The Instantaneous Lagrangian Acceleration (ILA) and Reynolds Averaged Navier-Stokes (RANS) based static pressure calculation methods were in agreement on the magnitude and spatial distribution of the static pressure field around the vortex core. Both found a low

pressure region peaking at -4 Pa at the very center of the vortex core. The ILA method created smoother and more symmetric pressure profiles, while the RANS method formed more asymmetric profiles as the distances from the swirl generator increased. This difference may be explained by the source terms used in each method. The RANS method is based on the Reynolds Stress tensor, which in effect stores information about the flow accelerations through variances and covariance of the velocity field in space. The pressure gradients from this method are calculated based on statistical means and variances of velocity only. The ILA method collects the instantaneous pressure gradients, calculated from instantaneous velocities and acceleration of individual particles, and then averages the gradients. In this way the solution to the Navier-Stokes equations should be more accurate when calculated on a per particle basis as the material acceleration term is solved using a second order finite difference scheme. The RANS method on the other hand will suffer from greater truncation and numerical error as it utilizes first order accurate finite difference schemes at the boundaries when solving the Navier-Stokes equations.

The following conclusions were made for the unconfined forced vortex flow case:

- The particle tracking system was able to faithfully track particles moving in complex 3D paths through a turbulent flow field. This was first verified by consistently observing the reconstruction of long (<100 frame) trajectories that wrapped around the vortex core.
- Analysis of the mean velocity field indicated rotation about a defined vortex core. The vortex flow field appeared as a ring of high velocity magnitude surrounding the low velocity core. The shear stress was zero where velocity reaches a maximum around the ring which is consistent with forced vortex theory.
- The two static pressure calculation methods based on RANS and ILA formulations were in agreement on the magnitude and structure of the pressure field which was qualitatively congruent with theory. However, the RANS approach appeared to suffer from numerical errors due to the use of first order finite difference schemes at the boundaries of the domain and through velocity averaging within the domain. More studies are required to validate the accuracy of these methods and better compare their advantages and disadvantages with different flow regimes.

6.7 Conclusions from experimental analysis

This chapter covered four experiments which were designed to validate the particle tracking system for the following: 1) 3D position reconstruction accuracy of static points, 2) accuracy and uncertainty of the velocity and acceleration measurements, 3) ability to track particles in turbulent jet flow, operation of the statistical accumulator framework, and characterize a flow field based on statistical properties of particle motion, 4) ability to track particles in a complex forced vortex flow field through inspection of statistical properties of the particle motion and evaluation of two static pressure calculation methods. The following conclusions were made.

6.7.1 3D Position measurement

- The average particle 3D position combined standard uncertainty for all unique four camera groups is 0.16mm based on the uncertainty of camera calibration parameters and particle centroid location uncertainty. This was independent of the location within the volume of interest and with the camera group used to complete the reconstruction.

- The particle tracking system was able to accurately measure the known distance between static particles and agrees with measurements made using a caliper within the standard uncertainty

6.7.2 Velocity and acceleration measurement

- The highest average relative difference with the test object's known tangential speed was 0.74% at 30 fps. The average acceleration magnitude measurement was accurate within 1% of the predicted value for lower frame rates, but differed significantly, up to 90%, for the higher frame rate of 120 fps. This was attributed to both the increased uncertainty as due to the inherent scaling with frame rate squared, and the fact that the particle experienced local velocity fluctuations which were better sampled at 120 fps.

- The standard uncertainty of velocity magnitude at 120 frames per second was 0.014 m/s, and that for acceleration is between 5.64 m/s^2. At 30 frames per second the combined standard uncertainties were 0.003 m/s and 0.35 m/s^2.

6.7.3 Tracking particles in turbulent jet flow

- The particle tracking systems measurements of a turbulent round jet compared well with those of hot-wire anemometry measurements found in literature. The axial velocity decay rate measured matched the range offered by several researchers. The transverse velocity profiles matched the Gaussian distribution of a fully developed self-similar jet flow.

- The Reynolds stresses matched the magnitudes and distributions reported in literature. The maximum axial turbulence intensity was on the order of 0.3 with slight peaks on both sides of the jet core. The maximum normalized Reynolds shear stress was on the order of 0.02 and had an anti-symmetric profile about the centerline of the jet. These profiles were observed in perpendicular planes indicating successful tracking in all three dimensions.

6.7.4 Tracking particles in forced vortex flow and static pressure calculations

- The particle tracking system was able to faithfully track particles moving in complex 3D paths through a turbulent flow field. This was first verified by consistently observing the reconstruction of long (>100 frame) trajectories that wrapped around the vortex core and analysis of the collect particle velocity statistics indicated symmetrical flow properties including Reynolds stresses and mean velocities. The profiles of these properties were consistent in perpendicular planes indicating successful particle tracking through complex 3D paths.

- The two static pressure calculation methods based on RANS and ILA formulations were in agreement on the magnitude and structure of the pressure field and qualitatively correct in the prediction of a local pressure minimum in the vortex core. The RANS approach appeared to suffer from numerical errors due to the use of first order finite difference schemes at the boundaries of the domain. More studies are required to validate the accuracy of these methods and better compare their advantages and disadvantages with different flow regimes.

7 Conclusions and recommendations

7.1 Conclusions

A real-time Lagrangian particle tracking (LPT) system was conceptualized, developed and validated for tracking inertial particles in fully turbulent 3D flow fields. The major algorithms of the real-time LPT system included image segmentation and particle detection, multi-camera correspondence, 3D reconstruction, tracking and interactive data visualization. These algorithms were implemented with the help of several key open source C++ libraries. A data analysis and visualization module was created to display results in both the Lagrangian and Eularian reference frame. The Lagrangian properties calculated include velocity, acceleration and static pressure gradient. These Lagrangian properties were attributed to a structured Cartesian grid comprised of statistical accumulators, through weighted means, variance and covariance. Eularian properties including mass residual, vorticity, Reynolds stress, turbulence intensity and static pressure were derived from the mean and variance of the velocity field.

Two parallel frameworks were developed to achieve real-time processing of LPT data. The first was a compute node based framework for multi-core processors with GPU accelerators. The second was a message passing framework for high performance clusters. These two frameworks can be used together to allow massive scaling of the LPT system through the addition of many more cameras recording at higher frame rates to spatially and temporally scale up the measurement domain. Real-time processing was achieved with synthetic data sets up to frame rates of 500 fps with 300 particles per frame on a single node. The cluster message passing framework showed that scaling could be achieved for 500 processors and speedups on the order of 200 fold were achieved with respect to an optimized sequential code. With this information it can be concluded that the objective of demonstrating the ability to reach real-time processing of LPT data scalable to hundreds of processors for high frame rate cameras was achieved.

A prototype real-time LPT system was created, utilizing six motion capture “smart” cameras, LED illumination and a generator of neutrally buoyant helium filled soap bubbles. An uncertainty analysis procedure was derived and used to conduct sensitivity studies on camera placement and calibration. The most influential factor in uncertainty propagation into the 3D reconstructed particle position was found to be the particle centroid location, resulting from the

image processing and particle detection algorithm. Regarding camera positioning, the selection of focal length and camera working distance were shown to not significantly impact the 3D position combined standard uncertainty for equivalent camera setups (large focal length and large working distance or short focal length and short working distance). It was shown that using four cameras in the 3D reconstruction process is very advantageous in reducing the sensitivity of the system to uncertainties and errors.

An experimental analysis was conducted to assess the standard uncertainty associated with 3D position, velocity and acceleration measurements. The average particle 3D position standard uncertainty for all unique four-camera groups was 0.16 mm. The position uncertainty was independent of the location within the volume of interest and the camera group used to complete the reconstruction. The particle tracking system was able to accurately measure the known distance between static particles and validated using a caliper as an alternative measurement method. The velocity and acceleration measurements were evaluated using a known object motion. The highest average velocity relative difference with a rotating test object's known tangential speed was 0.74% at 30 fps. The average acceleration magnitude measurement was accurate within 1% of the predicted value for lower frame rates, but differed significantly, up to 90%, for the higher frame rate of 120 fps. This was due in part to local velocity fluctuations in the object which were better resolved at higher frame rates, but mostly linked to the inherent amplification of uncertainty at smaller time scales. The combined standard uncertainty of velocity magnitude at 120 frames per second was 0.014 m/s, and that for acceleration was 5.64 m/s^2.

The LPT system's measurements of a turbulent round jet compared well with those of hot-wire anemometry measurements found in literature. The measured axial velocity decay rate matched the range reported by several researchers with a decay constant of 6.39 while the literature values range between 5.7 and 6.7. The transverse velocity profiles matched the predicted Gaussian distribution of a fully developed self-similar jet flow. The Reynolds stresses matched the magnitudes and distributions reported in literature. The maximum axial turbulence intensity was on the order of 0.3 with slight peaks on both sides of the jet core. The maximum normalized Reynolds shear stress was on the order of 0.02 and had an anti-symmetric profile

about the centerline of the jet. Very similar profiles were observed in perpendicular planes indicating successful tracking in all three dimensions.

With validation complete, the LPT system was applied to an unconfined forced vortex flow. The particle tracking system was able to faithfully track particles moving in complex 3D paths through the rotating turbulent flow field. This was first verified by consistently visually observing the reconstruction of long (>100 frame) trajectories that wrapped around the vortex core and analysis of the collect particle velocity statistics indicated symmetrical flow properties including Reynolds stresses and mean velocities. The profiles of these properties were consistent in perpendicular planes indicating successful particle tracking through complex 3D paths. The two static pressure calculation methods based on RANS and ILA formulations derived in Chapter 3 were in agreement on the magnitude and structure of the pressure field and qualitatively correct in the prediction of a local pressure minimum in the vortex core.

The LPT system proved to be robust and able to track particles in a variety of flow fields. The fact that the Eularian properties, which were calculated from inertial Lagrangian particle trajectories, were nearly identical to Eularian hot-wire based measurements (in the jet flow case) supports recent studies which found that inertial particles have nearly identical velocity statistics to ideal flow tracers.

7.2 Recommendations

- The velocities and accelerations calculated based on the raw trajectories can suffer from significant errors and uncertainties at higher frame rates. It is suggested that a smoothing kernel, polynomial approximation, or piecewise cubic spline be used to approximate a smooth trajectory through the points prior to calculating derivative properties. This approach is commonly used and has be discussed by (Willneff & Gruen, 2002).

- A very interesting observation during the jet flow experiment was the high correlation between the measured turbulence intensities and Reynolds shear stresses with values from literature even though large inertial particles were used. Further testing could be done in an attempt to measure grid generated turbulence in wind tunnels in order to identify the range of Reynolds numbers at which the system can accurately characterize the velocity statistics and Reynolds stresses. The experiment by (Bourgoin et al., 2011) could be recreated and compared with real-time LPT measurements.

- The calculation of static pressure from a measured velocity field is a very interesting area of this work and should be pursued further. An experiment can be designed to directly validate these models with a comparison of the calculated pressure field with measurements from high precision pressure transducers. Comparisons with CFD simulations would also be of interest.

- The sensitivity analysis showed that particle image centroid uncertainty played the largest role in creating uncertainty in the 3D particle position. For the case of neutrally buoyant helium bubbles, the illumination technique may play a role in creating particle images with multiple intensity peaks. The angle of illumination relative to the image plane should be evaluated for the impact on image type and quality.

- The system has been tested and validated with six cameras. The scalability of the system should be tested in real-life by adding up to twenty-five cameras and imaging a much larger volume. The system could be directly applied to larger scale flow fields.

- The real-time LPT system does not currently correct for light refraction caused by varying refractive indexes of transparent mediums. In order to conduct experiments where water, glass or other medium bends the light rays prior to reaching the cameras' lenses, a light refraction model should be implemented to correct such distortions. This will greatly expand the applicability of the real-time LPT system.

- The experimental study in this work focused on stationary flow fields, a round turbulent jet and an unconfined forced vortex. The real-time LPT system should be expanded to study non-stationary flow fields. The Eularian properties (velocity, static pressure, turbulent kinetic energy, etc.) of stationary flows were determined by temporal and spatial averaging of the instantaneous Lagrangian velocity, acceleration and static pressure gradient. This may be achieved by limiting the temporal averaging time scales, perhaps through a rolling average over a set number of frames, and/or increasing the spatial concentration of seed particles.

References

Adrian, R. J. (1991). Particle-imaging techniques for experimental fluid mechanics. *Annual Review of Fluid Mechanics, 23*(1), 261-304.

ANSI/ASHRAE/AMCA. (1999). Laboratory Methods of Testing Fans for Aerodynamic Performance Rating *51*.

Barker, Douglas, Lifflander, Jonathan, Arya, Anshu, & Zhang, Yuanhui. (2012). A parallel algorithm for 3D particle tracking and Lagrangian trajectory reconstruction. *Measurement Science and Technology, 23*(2), 025301.

Biwole, P. H., Yan, W., Zhang, Y., & Roux, J. J. (2009). A complete 3D particle tracking algorithm and its applications to the indoor airflow study. *Measurement Science and Technology, 20*(11), 115403.

Boguslawski, L., & Popiel, Cz.O. (1979). Flow Structure Of The Free Round Turbulent Jet In The Initial Region. *Journal of Fluid Mechanics, 90*(pt 3), 531 - 539.

Bouguet, J.Y. (2010). Camera Calibration Toolbox for Matlab. Retrieved 6/1/2012, 2012, from http://www.vision.caltech.edu/bouguetj/calib_doc/index.html

Bourgoin, Mickael, Qureshi, Nauman M., Baudet, Christophe, Cartellier, Alain, & Gagne, Cartellier. (2011). *Turbulent transport of finite sized material particles*, Warsaw, Poland.

Chan, K. Y., Stich, D., & Voth, G. A. (2007). Real-time image compression for high-speed particle tracking. *Review of Scientific Instruments, 78*(2), 023704.

Charonko, John J, King, Cameron V, Smith, Barton L, & Vlachos, Pavlos P. (2010). Assessment of pressure field calculations from particle image velocimetry measurements. *Measurement Science and Technology, 21*(10), 105401.

De Kat, R., & Van Oudheusden, B.W. (2012). Instantaneous planar pressure determination from PIV in turbulent flow. *Experiments in Fluids, 52*(5), 1089 - 1106.

Dougherty, E. R., & Laplante, P. A. (1995). *Introduction to real-time imaging*: Wiley-IEEE Press.

FancierStudio. (2012). LED Lighting Products. Retrieved January 23, 2012, from http://fancierstudio.com/

Fellouah, H., Ball, C.G., & Pollard, A. (2009). Reynolds number effects within the development region of a turbulent round free jet. *International Journal of Heat and Mass Transfer, 52*(17-18), 3943 - 3954.

Ferdman, E., Otugen, M. V., & Kim, S. (2000). Effect of Initial Velocity Profile on the Development of Round Jets. *J. Propulsion Power, 16*.

Heath. (2002). *Scientific Computing: An Introductory Survey* (2nd edition ed.): The McGraw-Hill Companies, Inc.

Hoyer, K., Holzner, M., Lüthi, B., Guala, M., Liberzon, A., & Kinzelbach, W. (2005). 3D scanning particle tracking velocimetry. *Experiments in Fluids, 39*, 923-934.

ISO. (2009). Uncertainty of measurement – Part 1: Introduction to the expression of the uncertainty in measurement *ISO/IEC Guide 98-1:2009*. Geneva, Switzerland: International Organization for Standardization (ISO).

Itseez. (2012). Open Source Computer Vision Library (OpenCV). Retrieved September 25, 2012, from http://opencv.org/

Jaw, S.Y., Chen, C.J., & Hwang, R.R. (2009). Measurement of instantaneous pressure distribution from PIV experiments. *Journal of Visualization 12*(1), 27-35.

Kale, L. V., & Krishnan, S. (1993). CHARM++: A portable concurrent object oriented system based on C++ (pp. 91-108).

Kerho, M.F., & Bragg, M.B. (1994). Neutrally buoyant bubbles used as flow tracers in air. *Experiments in Fluids, 16*(6), 393 - 400.

Kirkup, L., & Frenkel, R.B. (2006). *An Introduction to Uncertainty in Measurement: Using the GUM (Guide to the Expression of Uncertainty in Measurement)*: Cambridge University Press.

Kitzhofer, J., & Bruecker, C. (2010). Tomographic particle tracking velocimetry using telecentric imaging. *Experiments in Fluids, 49*, 1307-1324.

Kreizer, M, Ratner, D, & Liberzon, A. (2009). Real-time image processing for particle tracking velocimetry. *Experiments in Fluids, 48*(1), 105-110.

Kreizer, M., & Liberzon, A. (2010). Three-dimensional particle tracking method using FPGA-based real-time image processing and four-view image splitter. *Experiments in Fluids*, 1-8.

Li, D, Zhang, Y, Sun, Y, & Yan, W. (2008). A multi-frame particle tracking algorithm robust against input noise. *Measurement Science and Technology, 19*, 105401.

Lobutova, E, Resagk, C, & Putze, T. (2010). Investigation of large-scale circulations in room air flows using three-dimensional particle tracking velocimetry. *Building and Environment, 45*(7), 1653 - 1662.

Lüthi, B., Tsinober, A., & Kinzelbach, W. (2005). Lagrangian measurement of vorticity dynamics in turbulent flow. *Journal of Fluid Mechanics, 528*(-1), 87-118.

Lüthi, Beat, & Liberzon, Alex. (2012). 3D Particle Tracking Velocimetry software on Github. Retrieved September 24, 2012, from http://3dptv.github.com/

Maas, H. G., Gruen, A., & Papantoniou, D. (1993). Particle tracking velocimetry in three-dimensional flows: Part 1. Photogrammetric determination of particle coordinates. *Experiments in Fluids, 15*(2), 133-146.

Maas, H.G. (1992). Complexity analysis for the determination of image correspondences in dense spatial target fields. *International Archives of Photogrammetry and Remote Sensing, XXIX*, 102-107.

Malik, N A, Dracos, T, & Papantoniou, D A. (1993). Particle tracking velocimetry in three-dimensional flows: Part II. Particle tracking. *Experiments in Fluids, 15*(4), 279-294.

Meinhart, C D, Prasad, A K, & Adrian, R J. (1993). A parallel digital processor system for particle image velocimetry. *Measurement Science and Technology, 4*, 619-626.

Mercado, Julian Martinez, Prakash, Vivek N., Tagawa, Yoshiyuki, Sun, Chao, & Lohse, Detlef. (2012). Lagrangian statistics of light particles in turbulence. *Physics of Fluids, 24*(5), 055106.

Moin, Parviz. (2001). *Fundamentals of Engineering Numerical Analysis*. NY 10013-2473, USA: Cambridge University Press.

Murai, Yuichi, Nakada, Taishi, Suzuki, Takao, & Yamamoto, Fujio. (2007). Particle tracking velocimetry applied to estimate the pressure field around a Savonius turbine. *Measurement Science and Technology, 18*(8), 2491 - 2503.

NaturalPoint. (2012). Optitrack V120:SLIM Motion capture cameras. Retrieved May 5, 2012, from http://www.naturalpoint.com/optitrack/products/v120-slim/

Ni, Rui, Huang, Shi-Di, & Xia, Ke-Qing. (2012). Lagrangian acceleration measurements in convective thermal turbulence. *Journal of Fluid Mechanics, 692*, 395-419. doi: doi:10.1017/jfm.2011.520

Nordmann, Arne. (2007). Epipolar geometry. In Epipolar_geometry.svg (Ed.), (pp. Digital image). http://commons.wikimedia.org/wiki/File:Epipolar_geometry.svg: Wikimedia Commons.

NVIDIA. (2012). NVIDIA CUDA C Programming Guide. Retrieved June 2012, from http://developer.download.nvidia.com/compute/DevZone/docs/html/C/doc/CUDA_C_Programming_Guide.pdf

Okamoto, K, Nishio, S, Kobayashi, T, Saga, T, & Takehara, K. (2000). Evaluation of the 3D-PIV Standard Images (PIV-STD Project). *J. Vis., 3*, 115-123.

Ouellette, N. T., O'Malley, P. J., & Gollub, J. P. (2008). Transport of finite-sized particles in chaotic flow. *Phys Rev Lett, 101*(17), 174504.

Ouellette, N. T., Xu, H., & Bodenschatz, E. (2006). A quantitative study of three-dimensional Lagrangian particle tracking algorithms. *Experiments in Fluids, 40*(2), 301-313.

Pereira, F., Stuer, H., Graft, E. C., & Gharib, M. (2006). Two-frame 3D particle tracking. *Measurement Science and Technology, 17*(7), 1680 - 1692.

Quinn, W.R., Pollard, A., & Marsters, G.F. (1985). Mean Velocity And Static Pressure Distributions In A Three-dimensional Turbulent Free Jet. *AIAA journal, 23*(6), 971 - 973.

Qureshi, Nauman M., Bourgoin, Mickael, Baudet, Christophe, Cartellier, Alain, & Gagne, Yves. (2007). Turbulent transport of material particles: An experimental study of finite size effects. *Physical Review Letters, 99*(18).

SAI. (2012). Sage Action Inc. Bubble Generator Systems for Air Flow Visualization and Measurement. Retrieved June 2, 2012, from http://www.sageaction.com/

Satake, S. I., Anraku, T., Kanamori, H., Kunugi, T., Sato, K., & Ito, T. (2008). Study on high speed parallel algorithm using PC grid environment for visualization measurements by Digital Holographic Particle Tracking Velocimetry. *Computer Physics Communications, 178*(1), 1 - 7.

Satake, S. I., Kanamori, H., Kunugi, T., Sato, K., Ito, T., & Yamamoto, K. (2007). Parallel computing of a digital hologram and particle searching for microdigital-holographic particle-tracking velocimetry. *Appl. Opt., 46*(4), 538-543.

Shindler, L., Moroni, M., & Cenedese, A. (2010). Spatial–temporal improvements of a two-frame particle-tracking algorithm. *Measurement Science and Technology, 21*(11), 115401.

Straw, A. D., Branson, K., Neumann, T. R., & Dickinson, M. H. (2010). Multi-camera Realtime 3D Tracking of Multiple Flying Animals. *Journal of the Royal Society Interface, rsif.2010.0230v1-rsif20100230*.

Tamron. (2012). Product data sheet for Lens model M13VM288IR. Retrieved October 12, 2011, from http://www.tamron.co.jp/en/data/cctv/m13vm288ir.html

Toschi, Federico, & Bodenschatz, Eberhard. (2009). Lagrangian properties of particles in turbulence. *Annual Review of Fluid Mechanics, 41*, 375 - 404.

Trucco, E., & Verri, A. (1998). *Introductory techniques for 3-D computer vision*: Prentice Hall, Upper Saddle River, NJ.

Violato, Daniele, Moore, Peter, & Scarano, Fulvio. (2011). Lagrangian and Eulerian pressure field evaluation of rod-airfoil flow from time-resolved tomographic PIV. *Experiments in Fluids, 50*, 1057 - 1070.

Virant, M., & Dracos, T. (1997). 3D PTV and its application on Lagrangian motion. *Measurement science and technology, 8*, 1539-1552.

Willneff, J., & Gruen, A. (2002). A new spatio-temporal matching algorithm for 3D-particle tracking velocimetry (pp. 14).

Wu, Enhua. (2008). Emerging technology about GPGPU. *APCCAS 2008 2008 IEEE Asia Pacific Conference on Circuits and Systems*(60773030), 618-622.

Xu, G., & Antonia, R.A. (2002). Effect of different initial conditions on a turbulent round free jet. *Experiments in Fluids, 33*(5), 677 - 683.

Zhang, Y. (2005). *Indoor Air Quality Engineering*. Boca Raton, Florida 33431: CRC Press.

Zhang, Z. (2000). A flexible new technique for camera calibration. *Pattern Analysis and Machine Intelligence, IEEE Transactions on, 22*(11), 1330 - 1334.

Appendix A: Additional data plots from round jet flow experiment

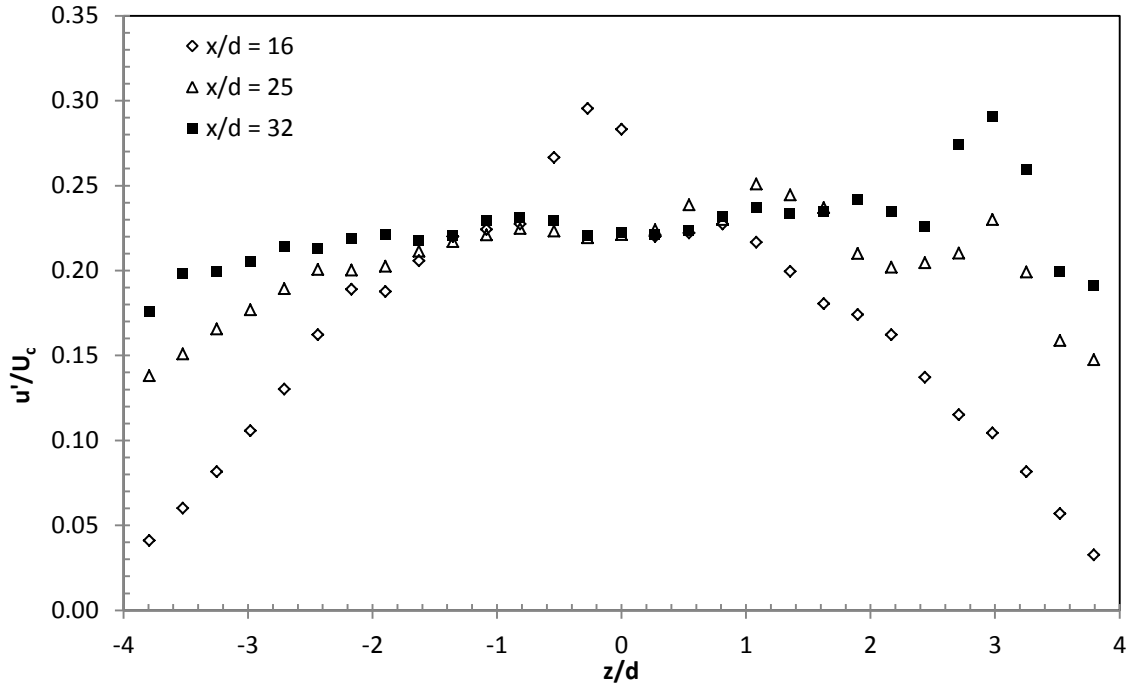

Figure 76: Round jet flow - Axial turbulence intensity profile on XZ center plane

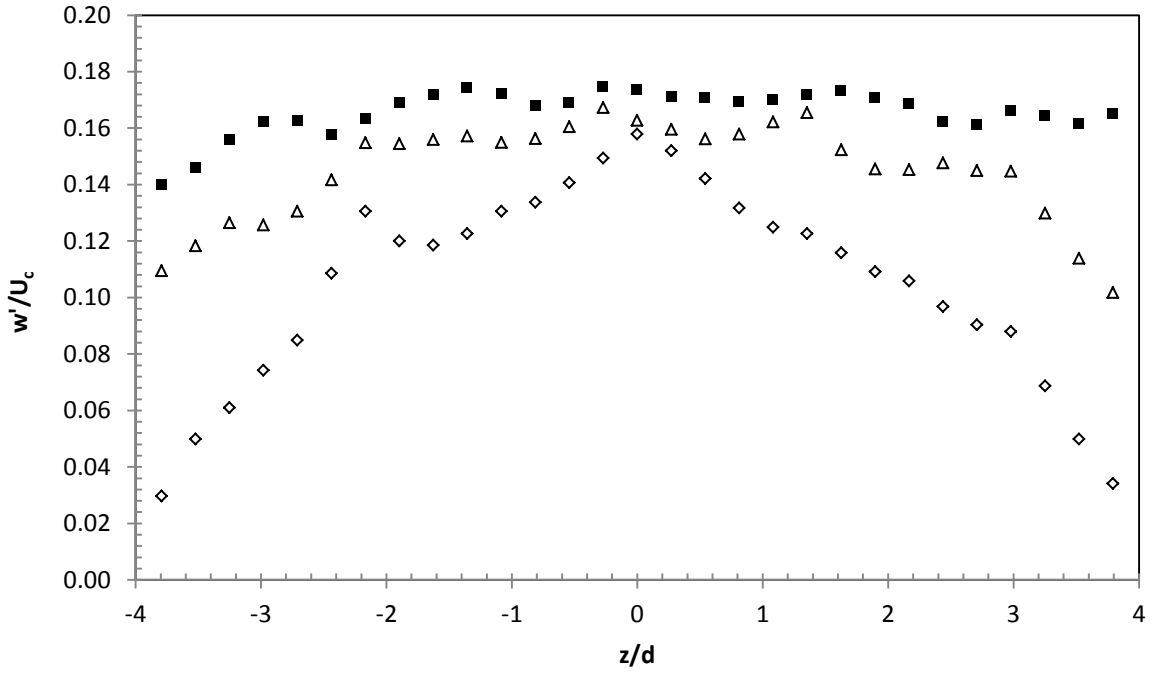

Figure 77: Round jet flow - Transverse turbulence intensity profile on XZ center plane

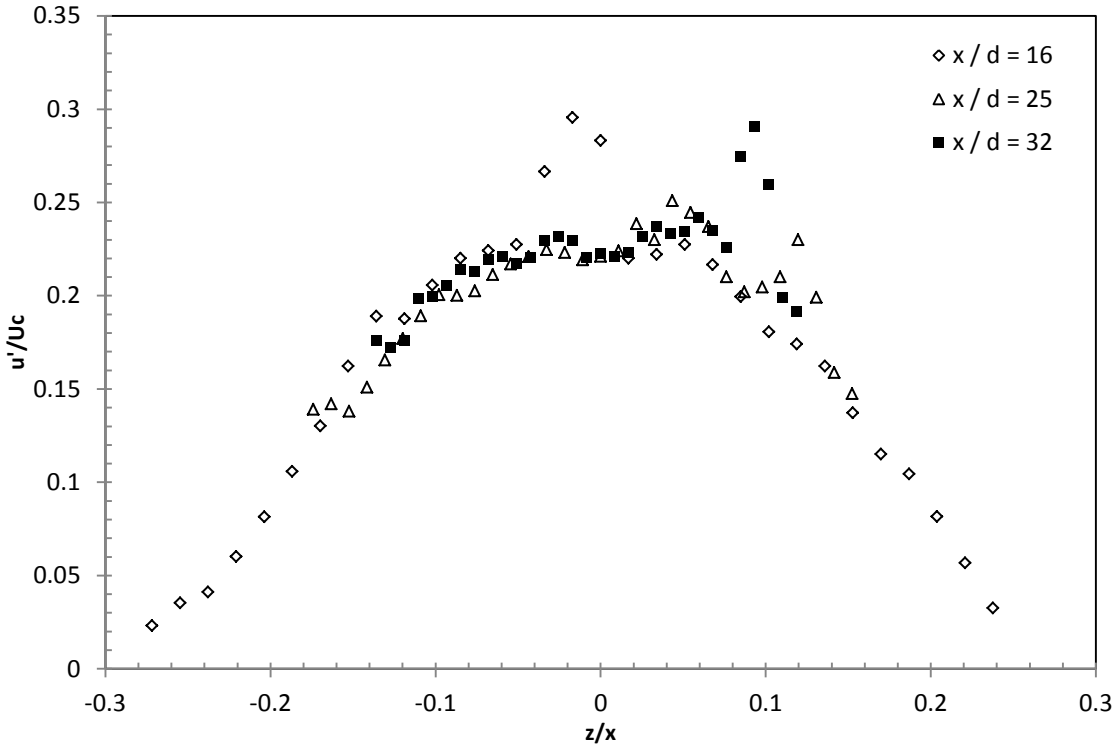

Figure 78: Round jet flow - Axial turbulence intensity profile on XZ center plane

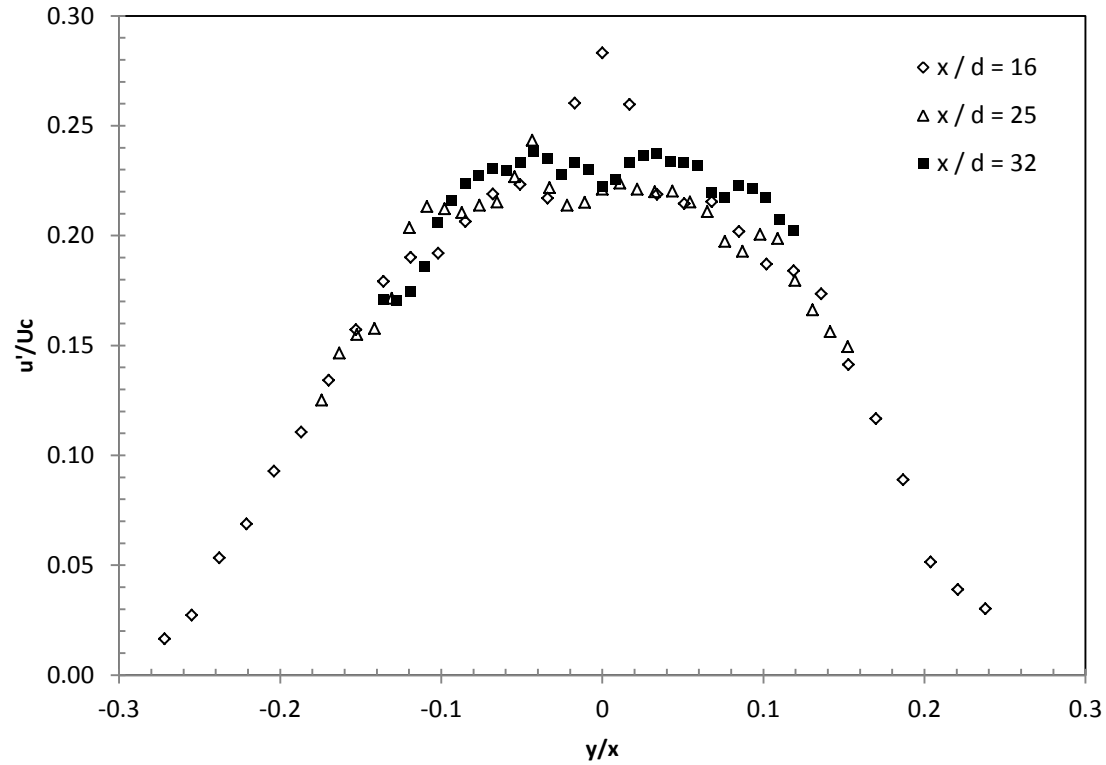

Figure 79: Round jet flow - Axial turbulence intensity profile on XY center plane

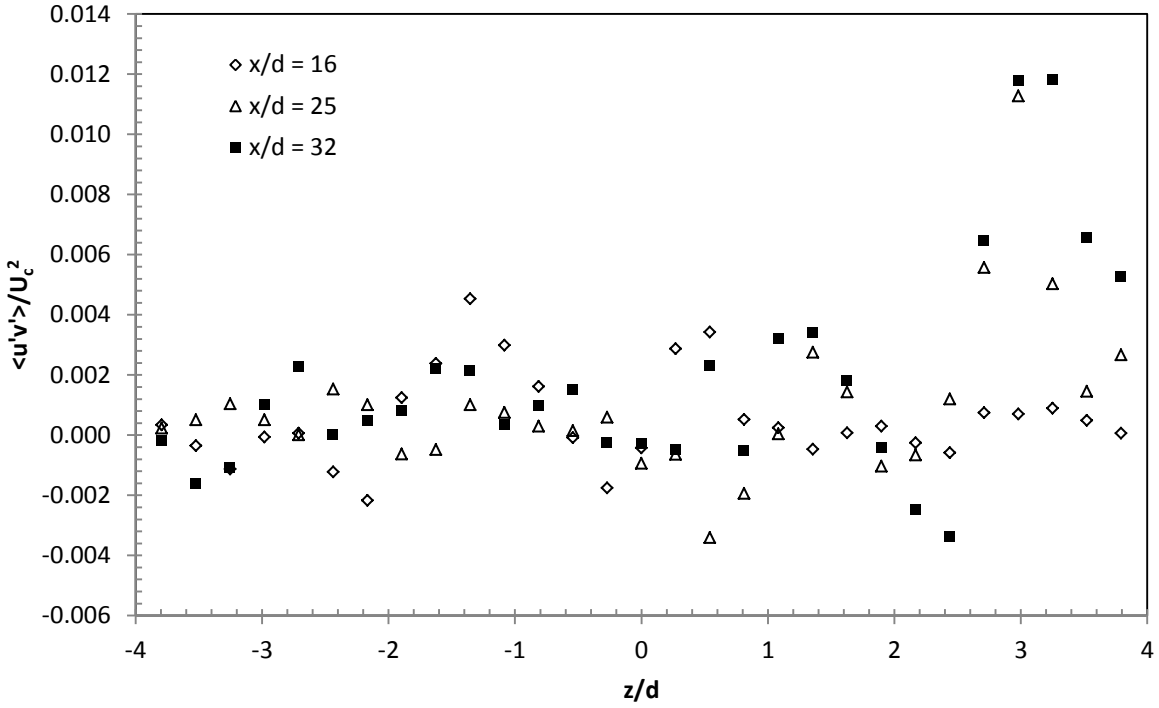

Figure 80: Round jet flow - Reynolds shear stress <u'v'> profile on XZ center plane

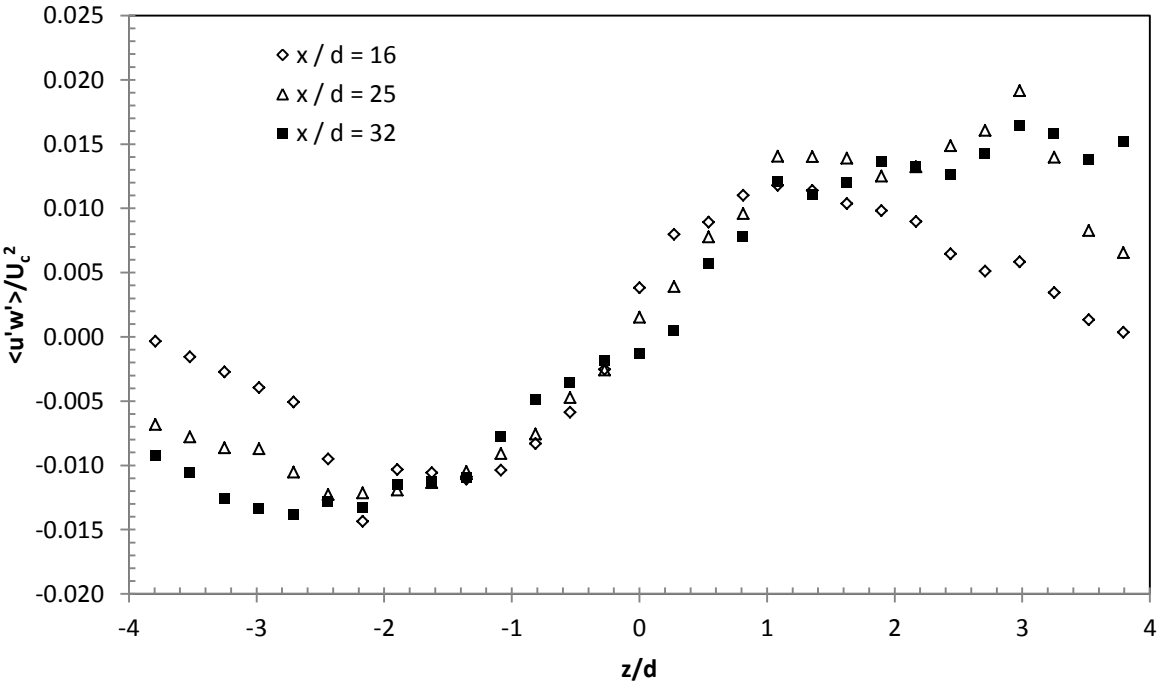

Figure 81: Round jet flow - Reynolds shear stress <u'w'> profile XZ center plane

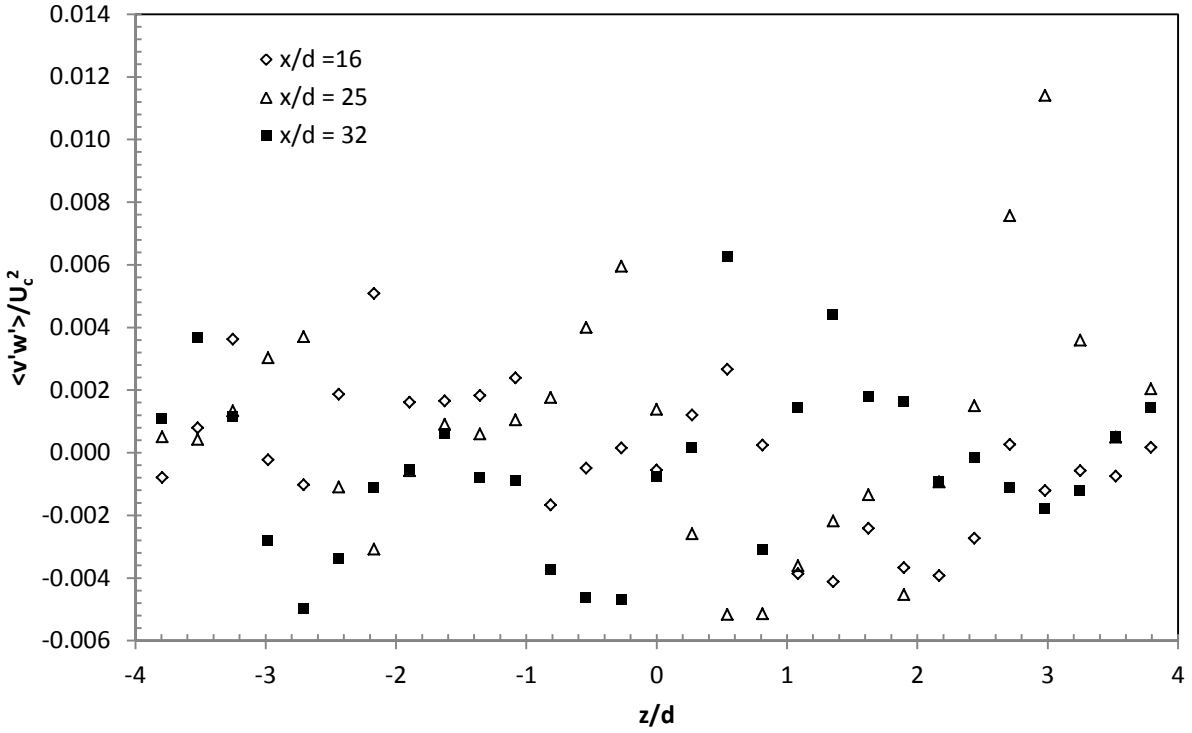

Figure 82: Round jet flow - Reynolds stress <v'w'> profile on XZ center plane

Appendix B: Additional data plots from forced vortex flow experiment

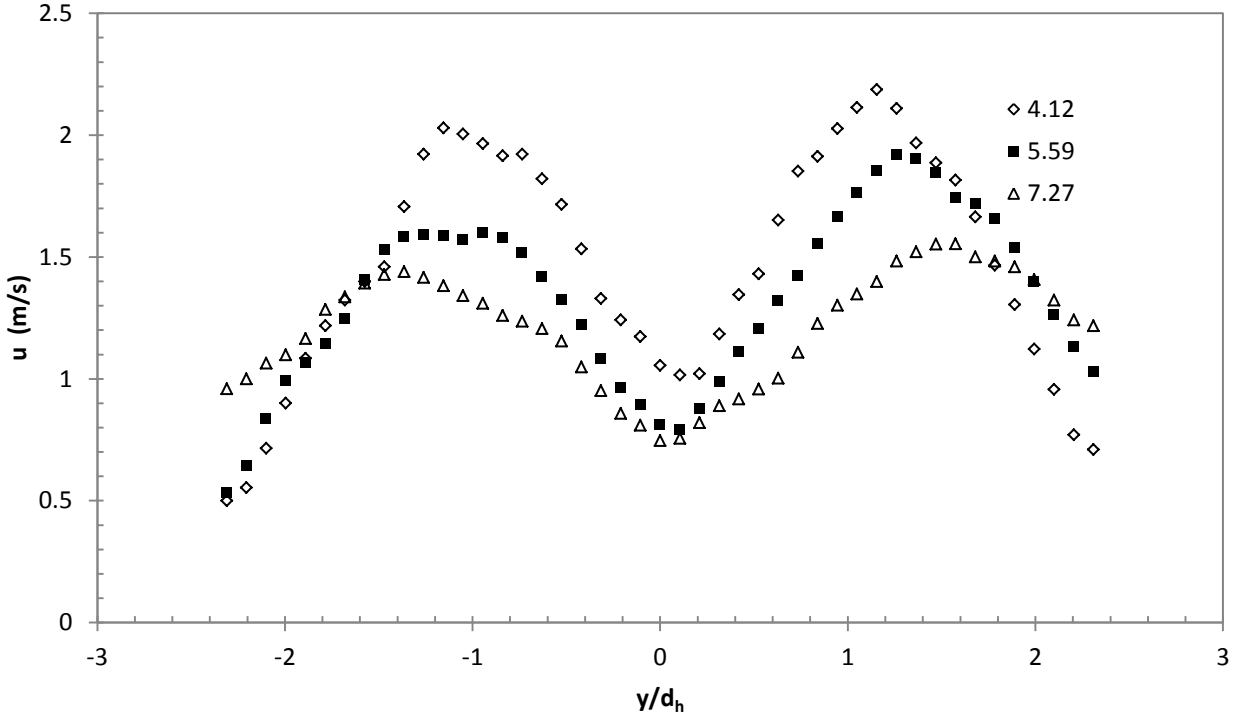

Figure 83: Forced vortex flow – Mean axial velocity <u> on XY center plane at x/d_h = 4.12, 5.59 and 7.27

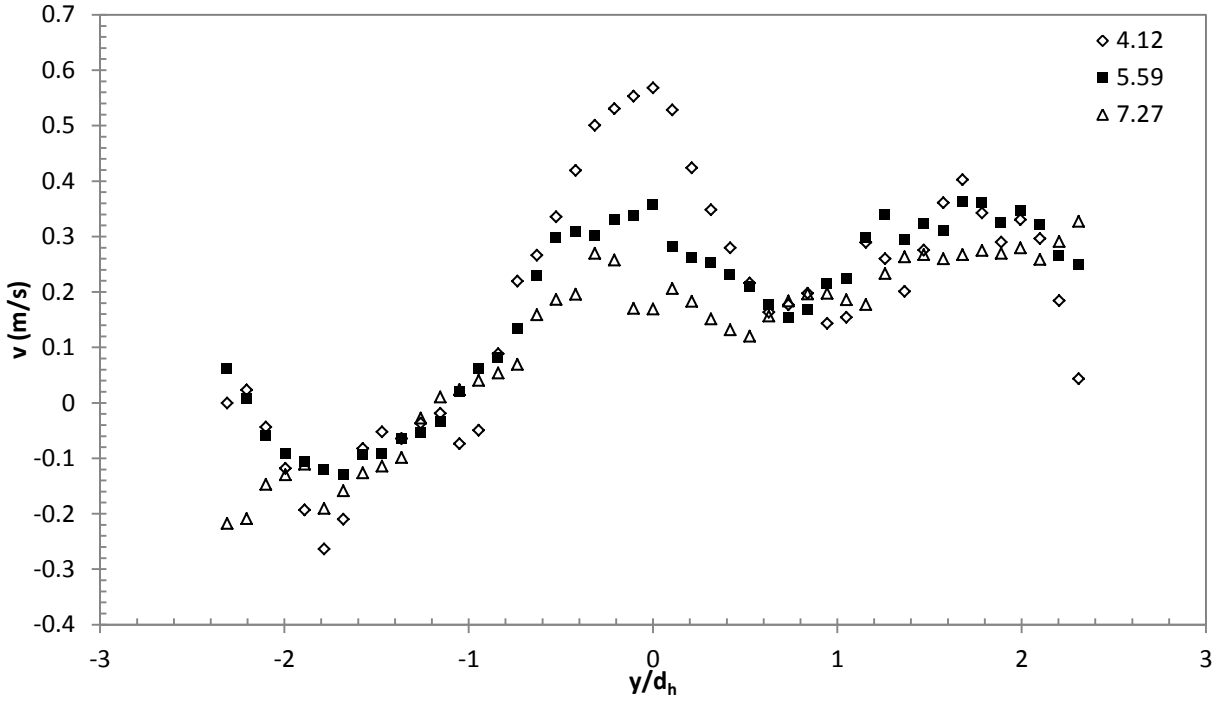

Figure 84: Forced vortex flow – Mean transverse velocity <v> on XY center plane at x/d_h = 4.12, 5.59 and 7.27

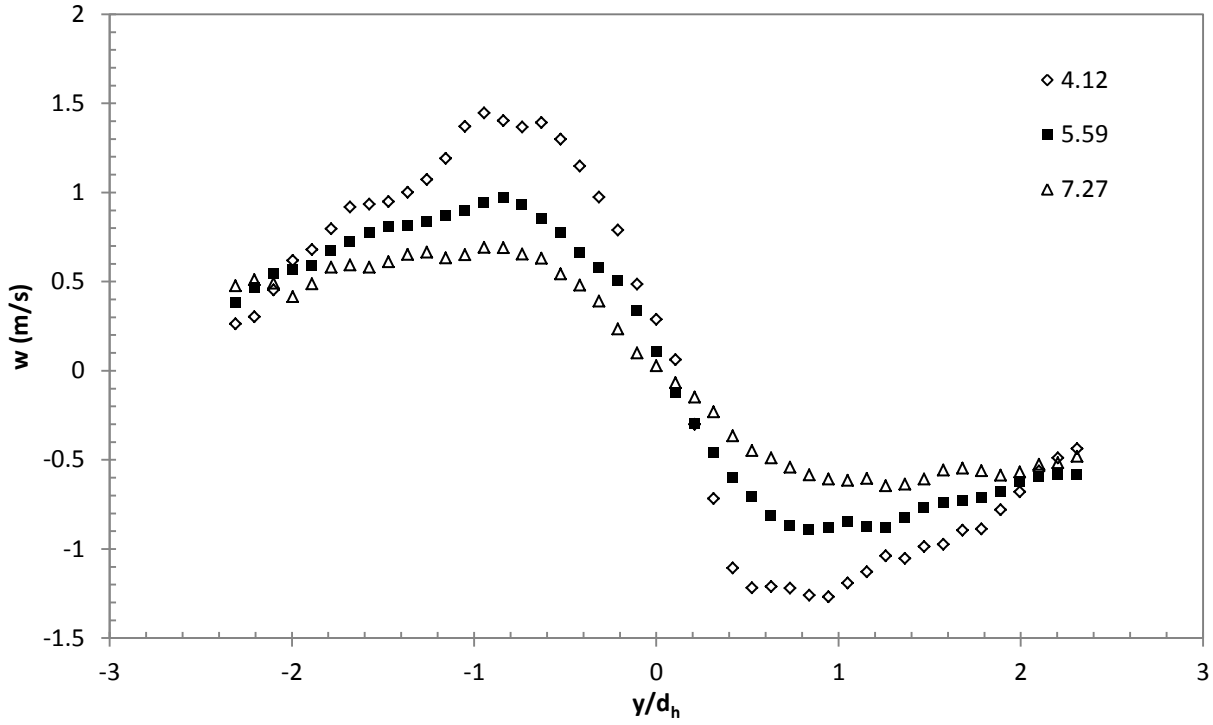

Figure 85: Forced vortex flow – Mean transverse velocity <w> on XY center plane at x/d_h = 4.12, 5.59 and 7.27

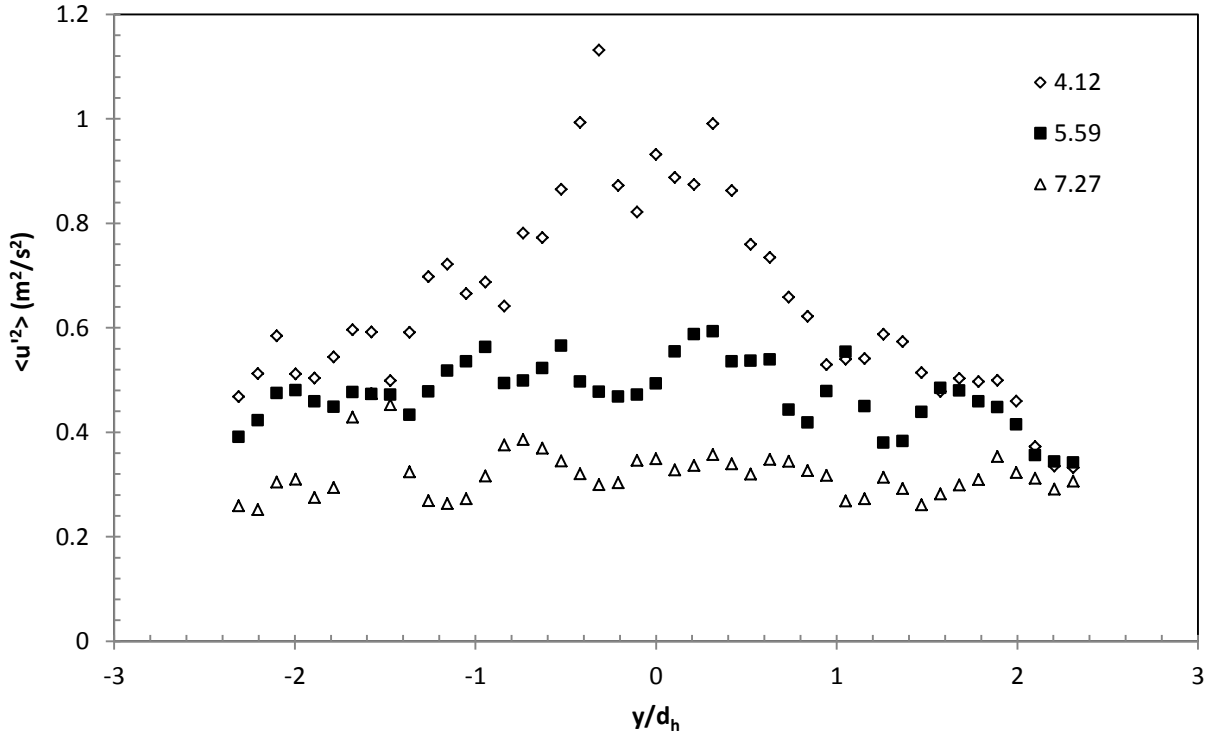

Figure 86: Forced vortex flow –Axial velocity variance $<u'^2>$ on XY center plane at x/d_h = 4.12, 5.59 and 7.27

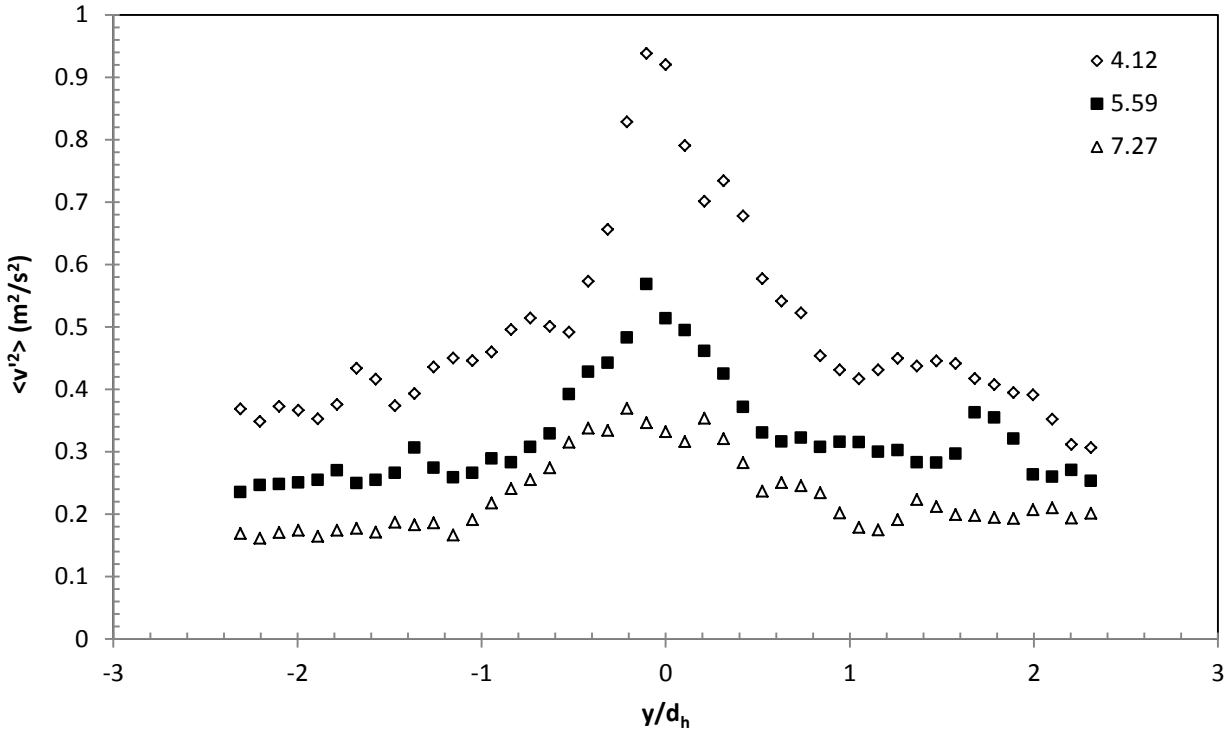

Figure 87: Forced vortex flow – Transverse velocity variance <v'^{2}> on XY center plane at x/d_h = 4.12, 5.59 and 7.27

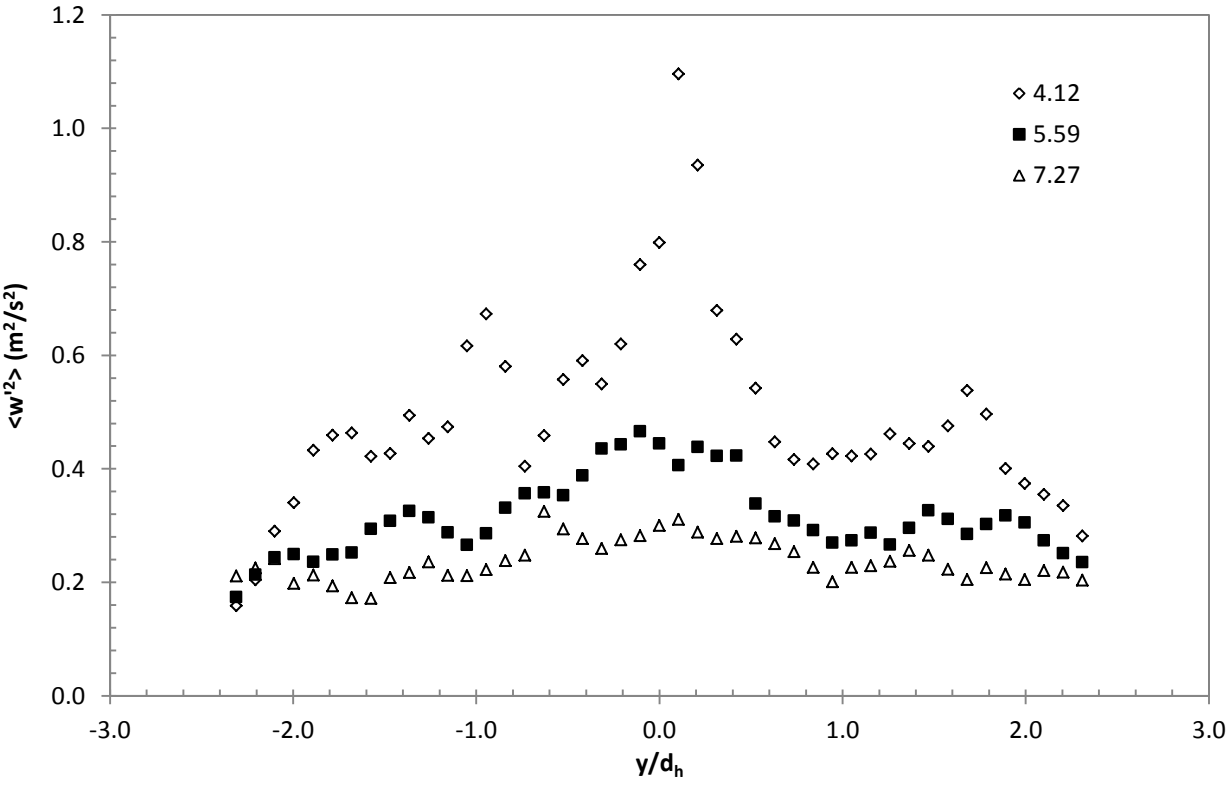

Figure 88: Forced vortex flow – Transverse velocity variance <w'^{2}> on XY center plane at x/d_h = 4.12, 5.59 and 7.27

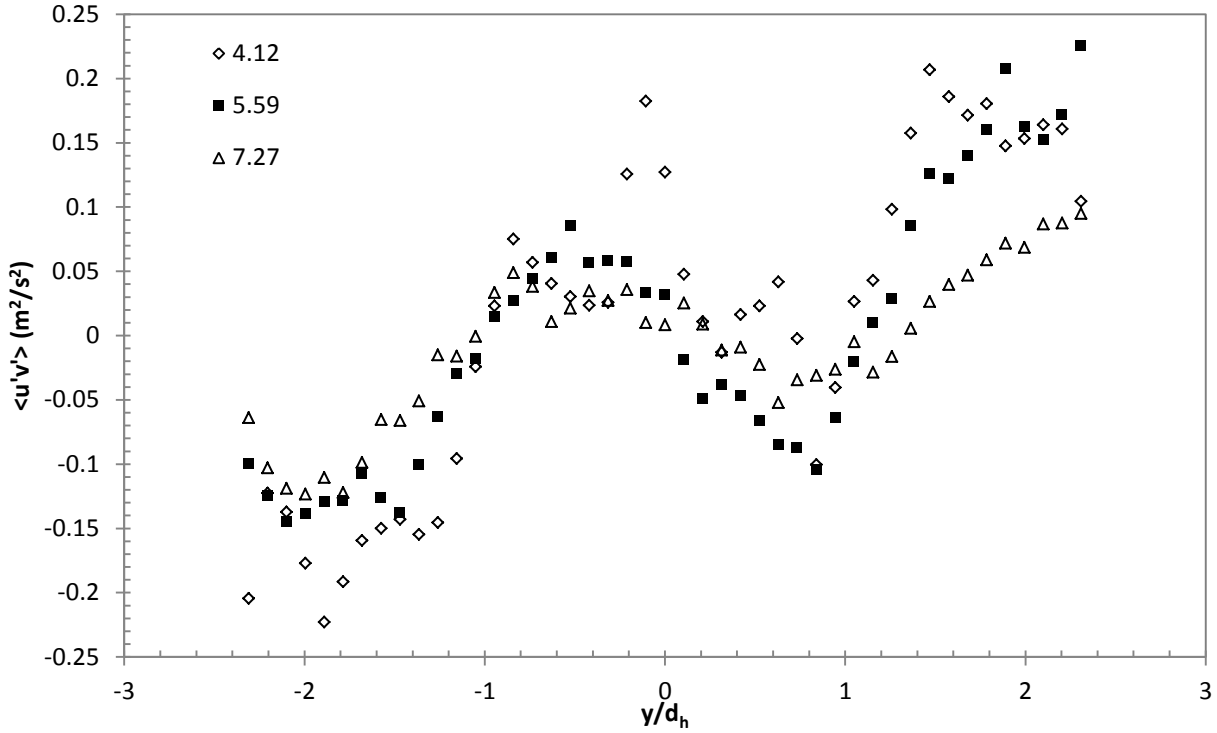

Figure 89: Forced vortex flow – Reynolds shear stress <u'v'> on XY center plane at x/d_h = 4.12, 5.59 and 7.27

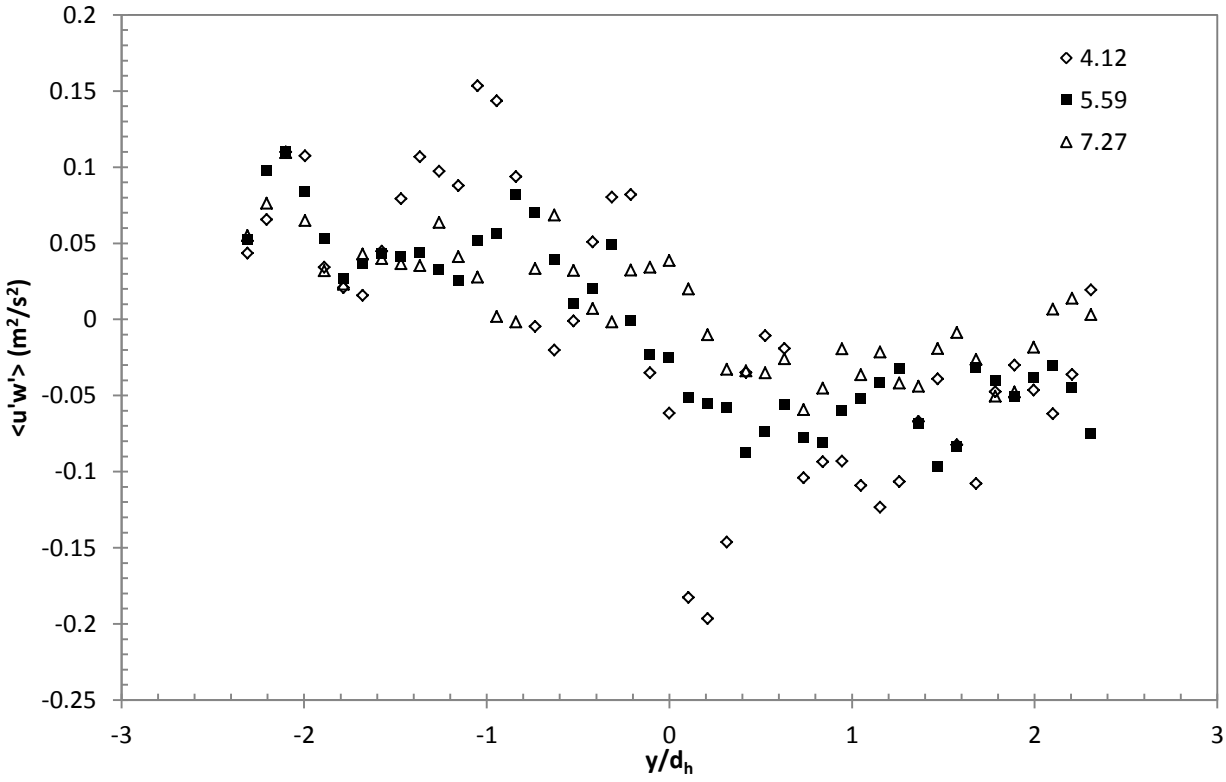

Figure 90: Forced vortex flow – Reynolds shear stress <u'w'> on XY center plane at x/d_h = 4.12, 5.59 and 7.27

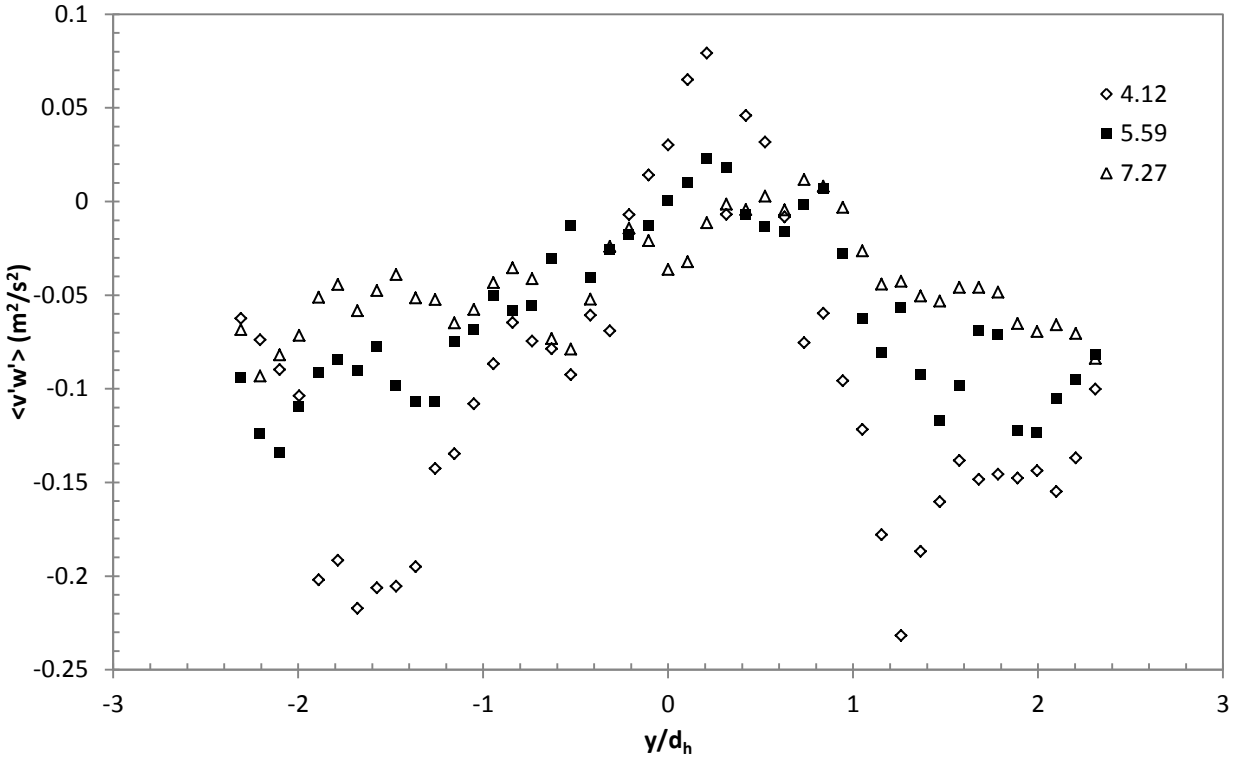

Figure 91: Forced vortex flow – Reynolds shear stress <v'w'> on XY center plane at x/d_h = 4.12, 5.59 and 7.27

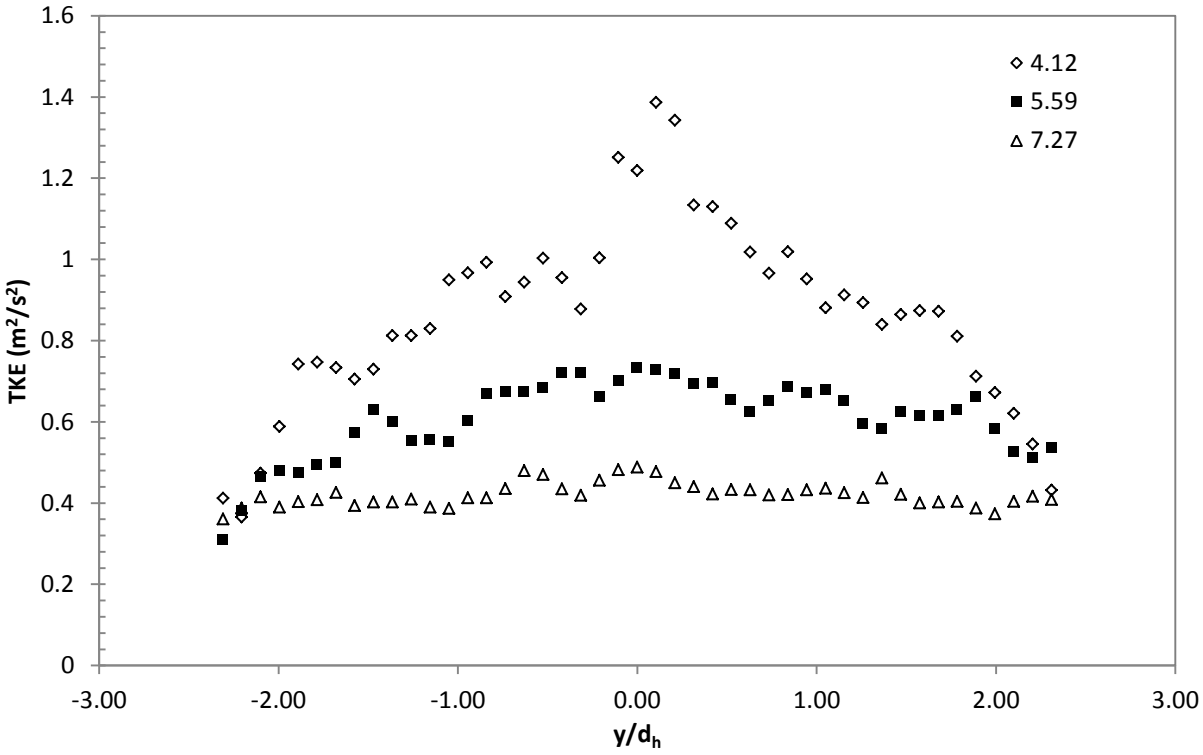

Figure 92: Forced vortex flow – Turbulent kinetic energy on XY center plane at x/d_h = 4.12, 5.59 and 7.27

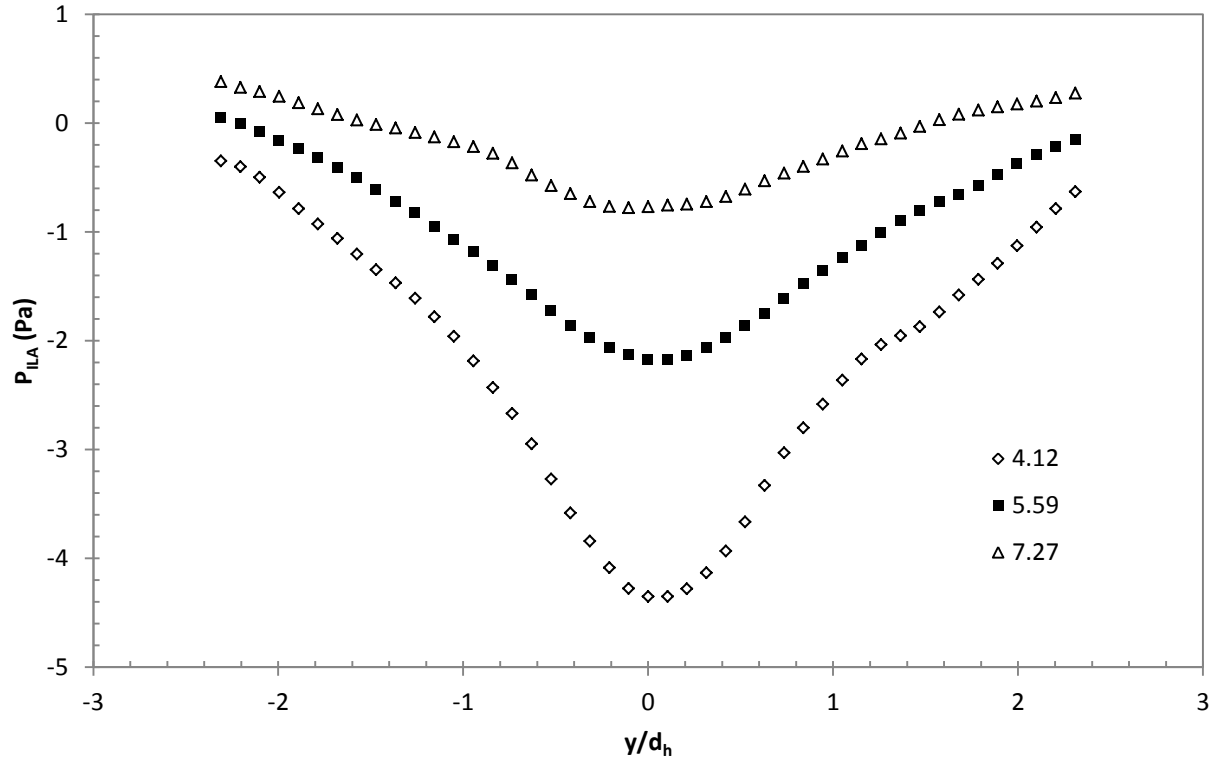

Figure 93: Forced vortex Flow- calculated static pressure on the XY center plane at x/d_h = 4.12, 5.59 and 7.27 based on Instantaneous Lagrangian Acceleration (ILA) method

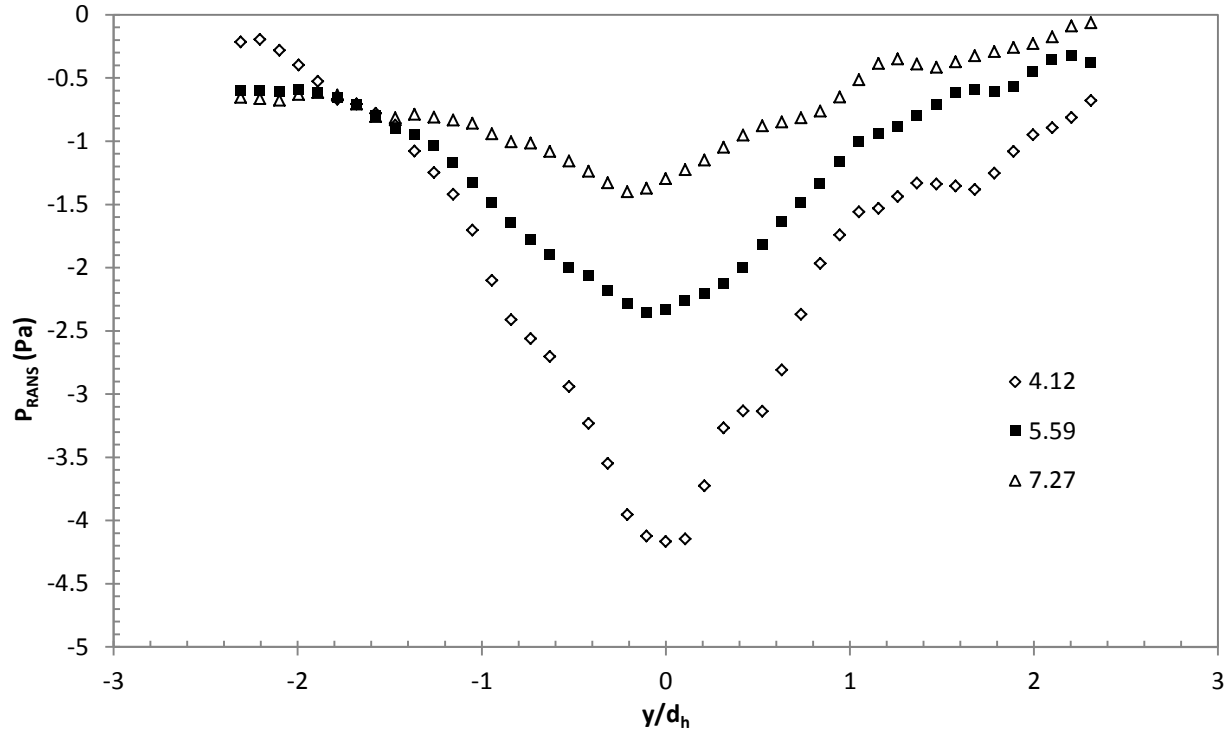

Figure 94: Forced vortex flow - calculated static pressure on the XY center plane at x/d_h = 4.12, 5.59 and 7.27 based on Reynolds Averaged Navier-Stokes (RANS) method

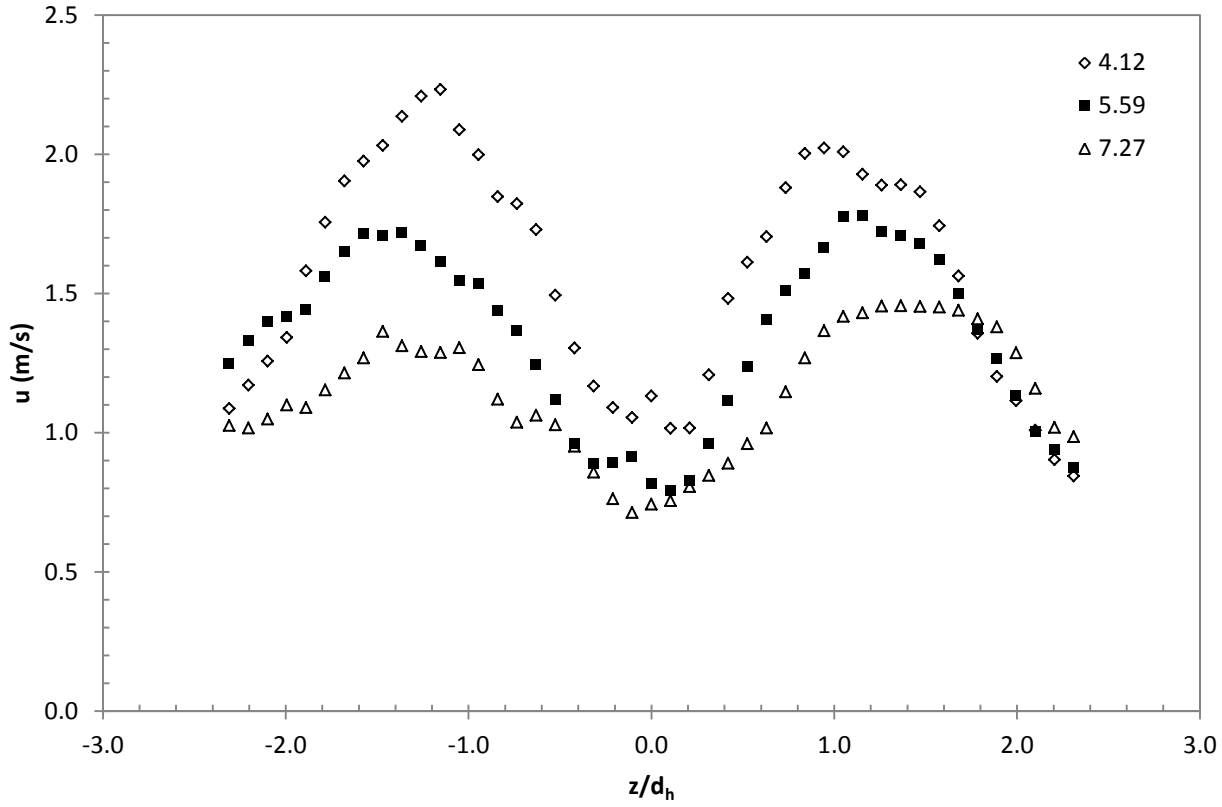

Figure 95: Forced vortex flow – Mean axial velocity <u> on XZ center plane at x/d_h = 4.12, 5.59 and 7.27

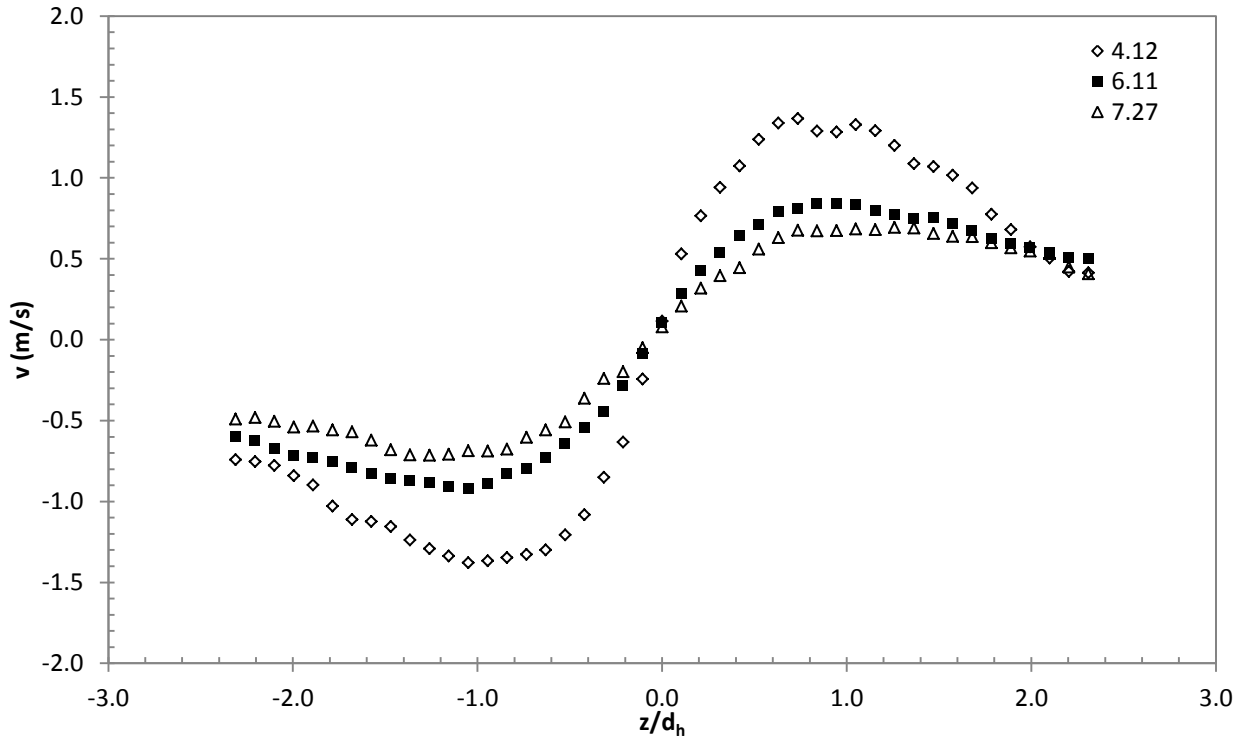

Figure 96: Forced vortex flow – Mean transverse velocity <v> on XZ center plane at x/d_h = 4.12, 5.59 and 7.27

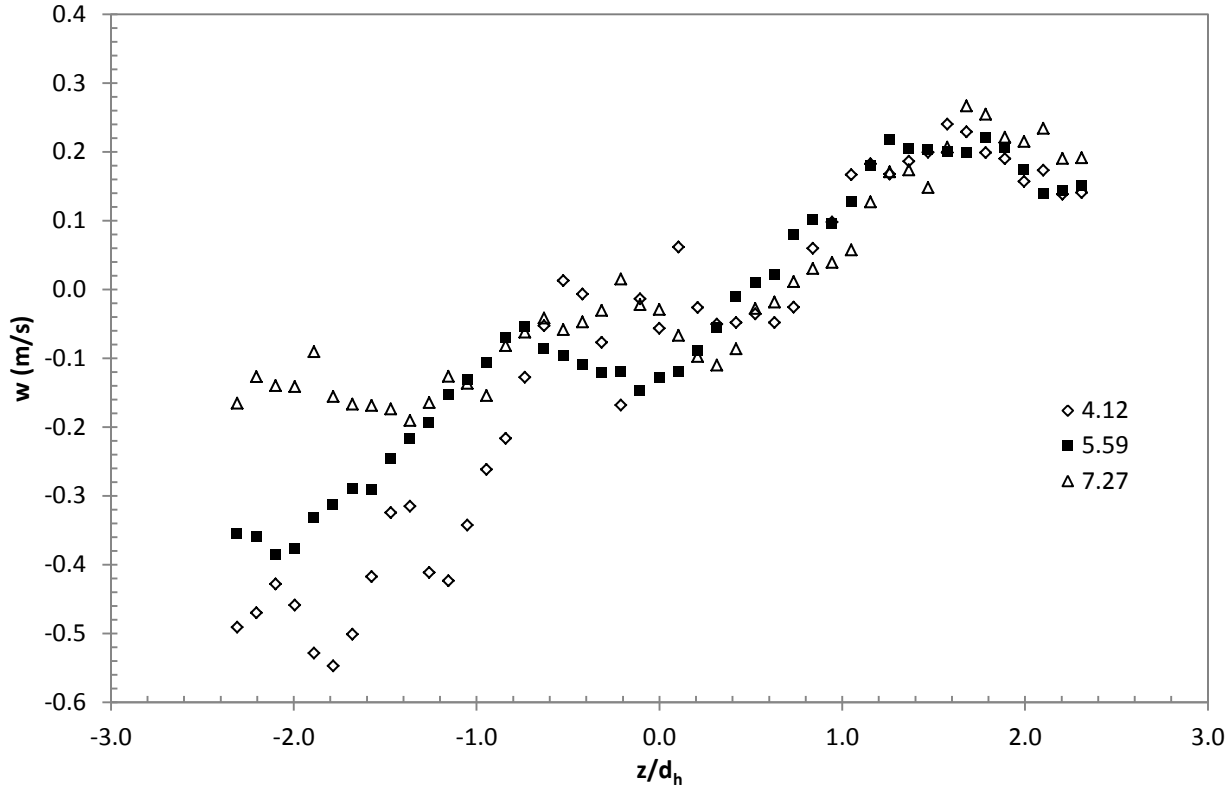

Figure 97: Forced vortex flow – Mean transverse velocity <w> on XZ center plane at x/d_h = 4.12, 5.59 and 7.27

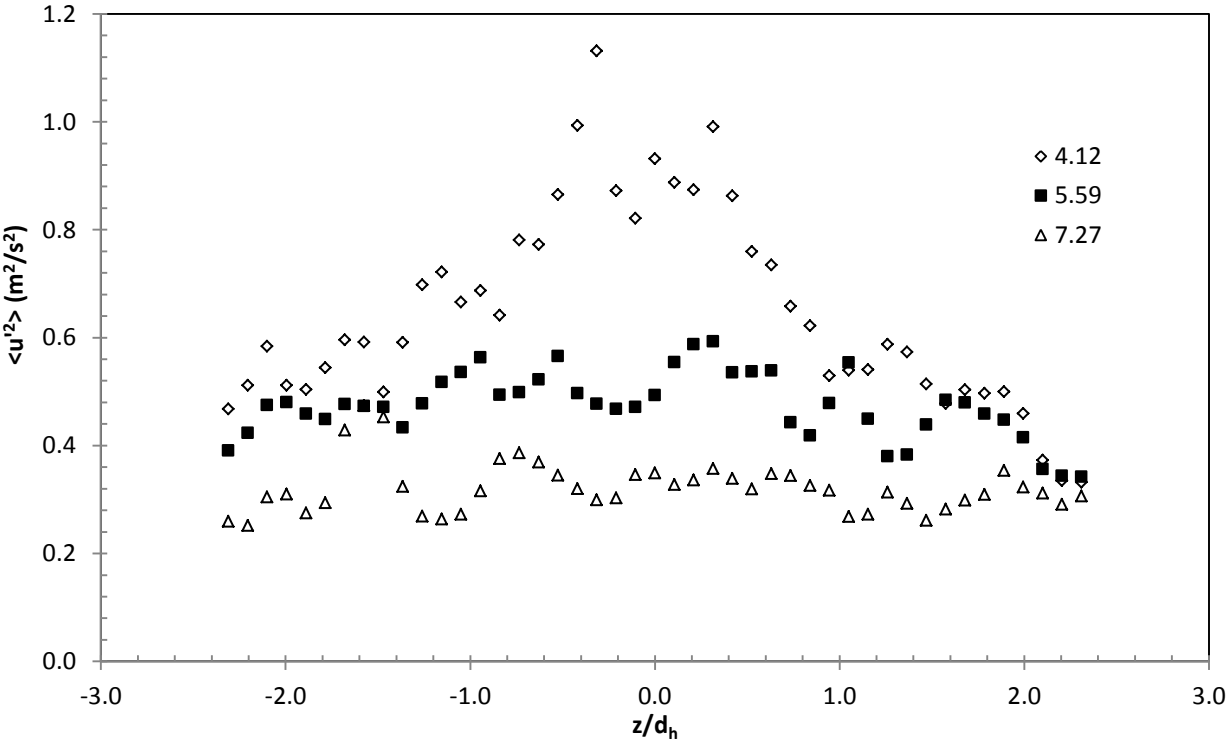

Figure 98: Forced vortex flow – Axial velocity variance $<u'^2>$ on XZ center plane at x/d_h = 4.12, 5.59 and 7.27

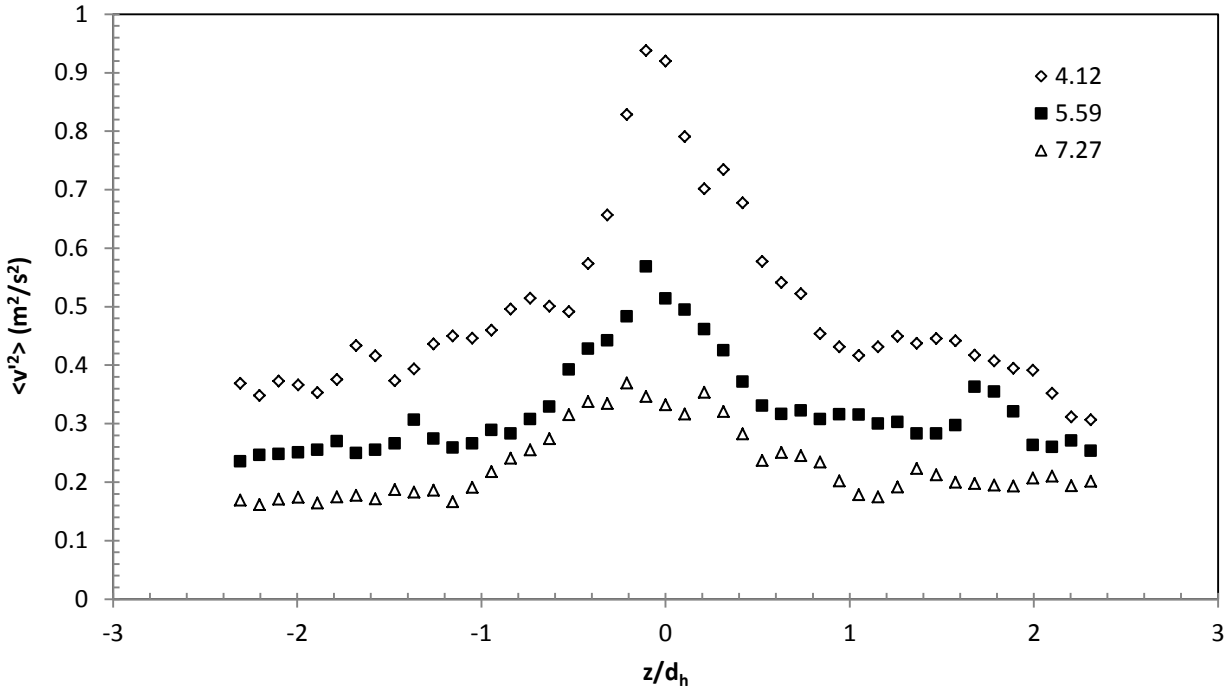

Figure 99: Forced vortex flow – Transverse velocity variance <v'²> on XZ center plane at x/d_h = 4.12, 5.59 and 7.27

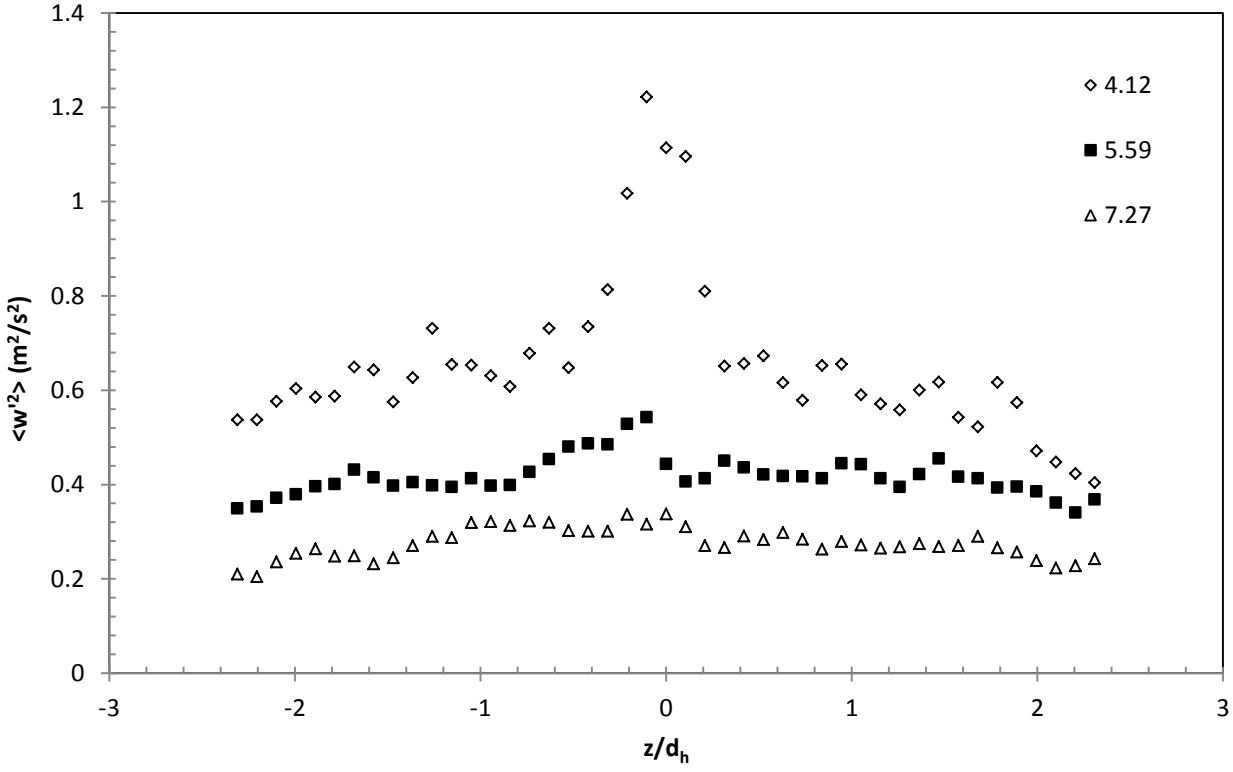

Figure 100: Forced vortex flow – Transverse velocity variance <w'²> on XZ center plane at x/d_h = 4.12, 5.59 and 7.27

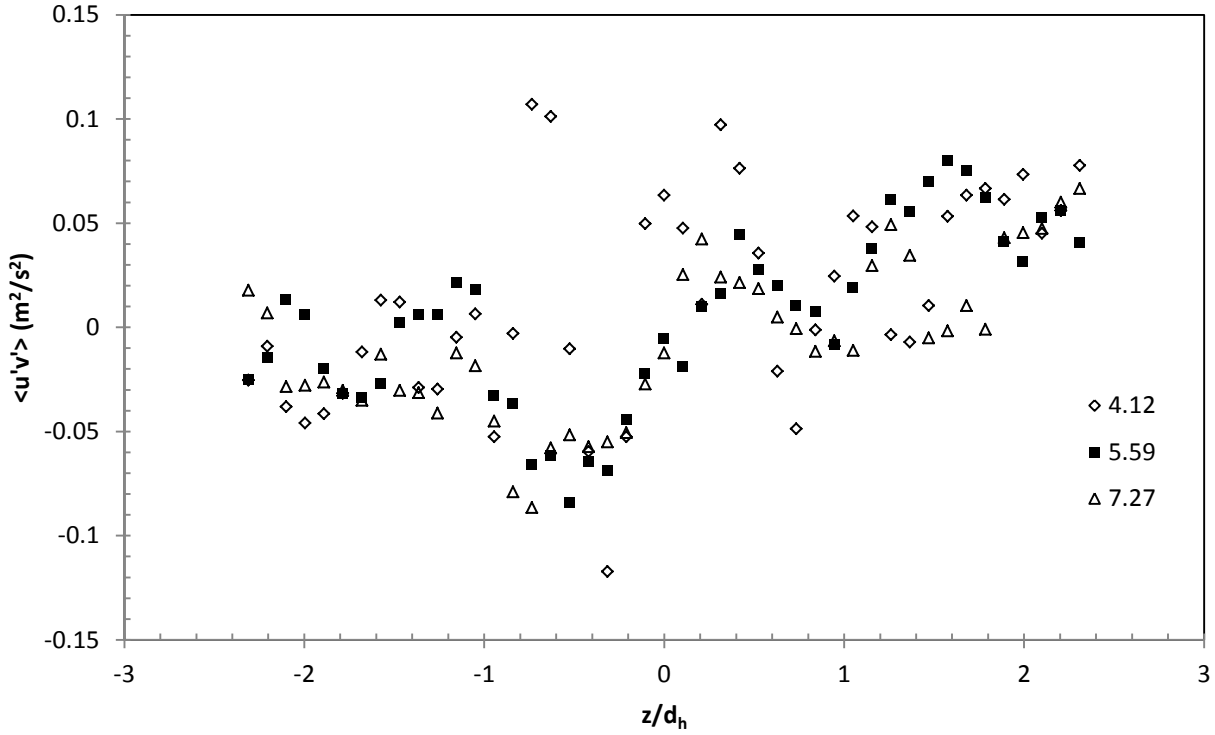

Figure 101: Forced vortex flow – Reynolds shear stress <u'v'> on XZ center plane at x/d_h = 4.12, 5.59 and 7.27

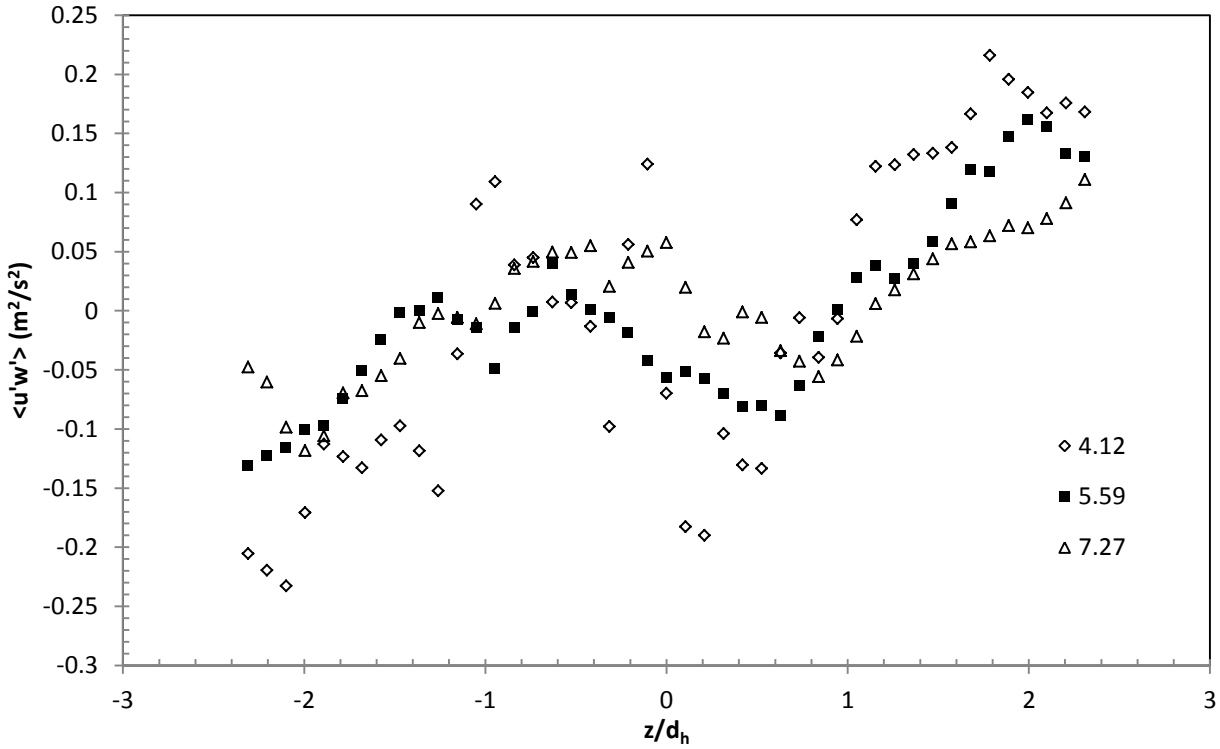

Figure 102: Forced vortex flow – Reynolds shear stress <u'w'> on XZ center plane at x/d_h = 4.12, 5.59 and 7.27

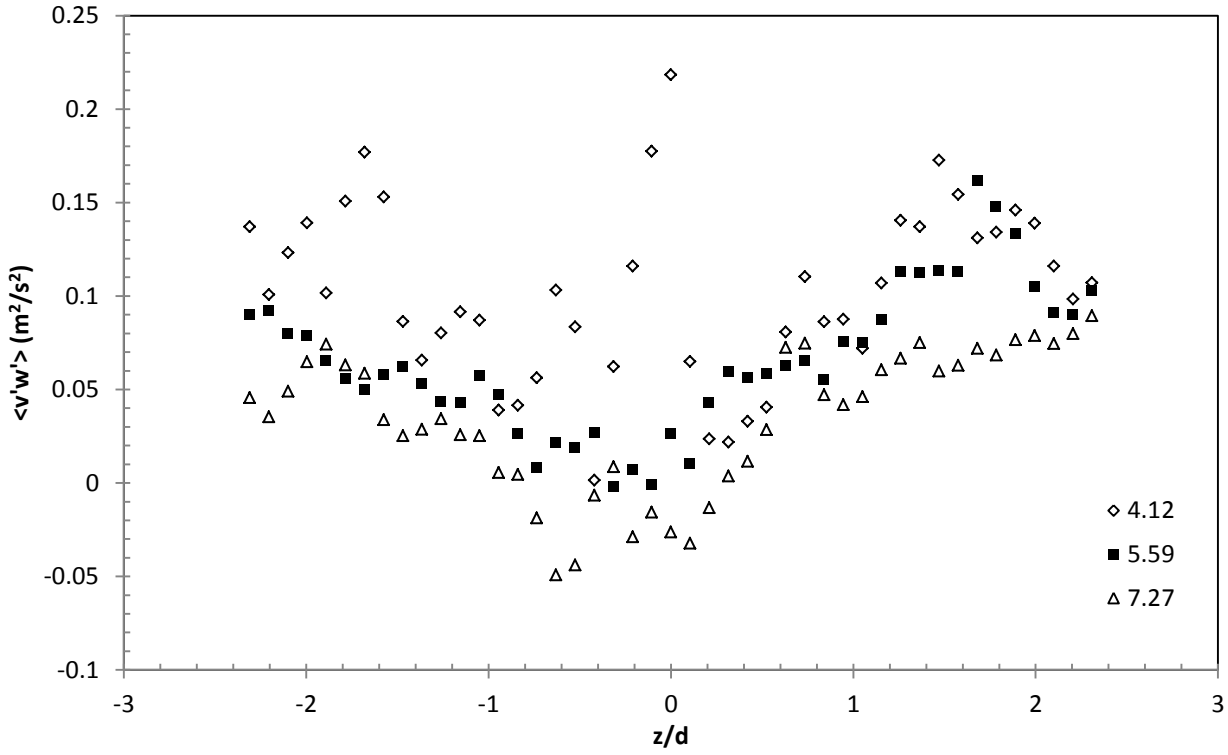

Figure 103: Forced vortex flow – Reynolds shear stress <v'w'> on XZ center plane at x/d_h = 4.12, 5.59 and 7.27

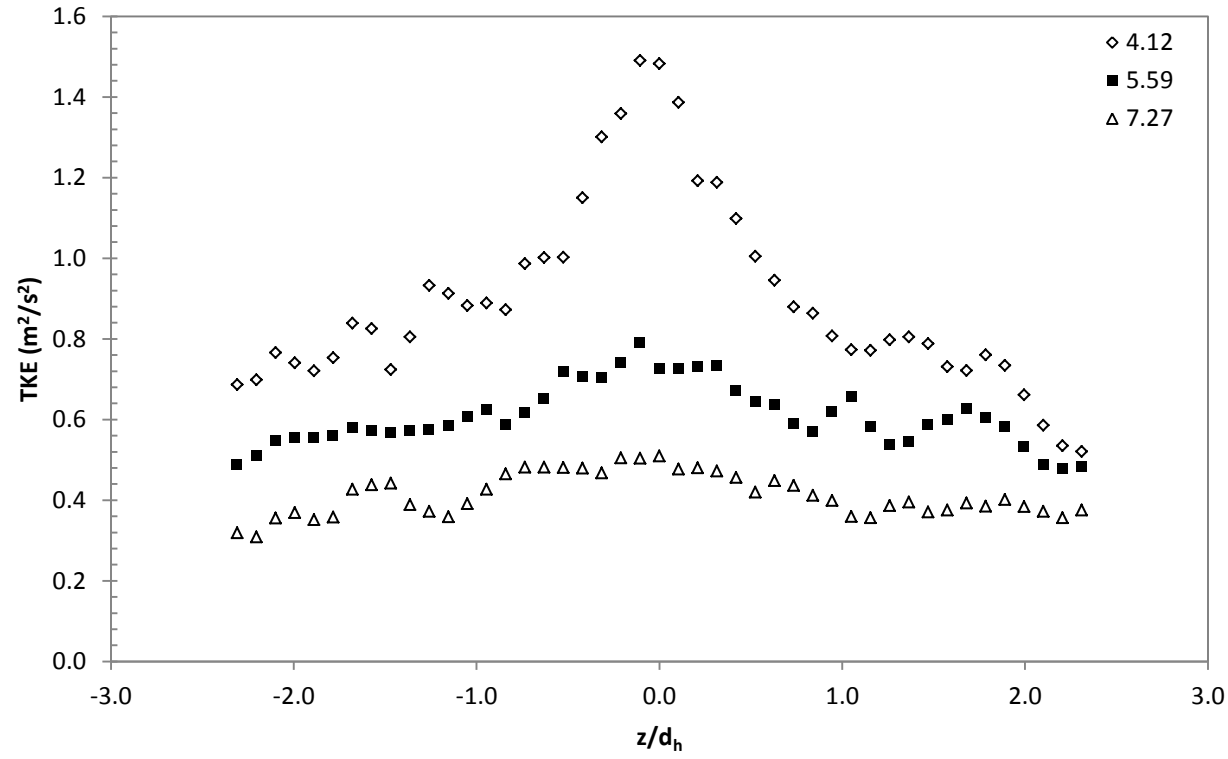

Figure 104: Forced vortex flow – Turbulent kinetic energy on XZ center plane at x/d_h = 4.12, 5.59 and 7.27

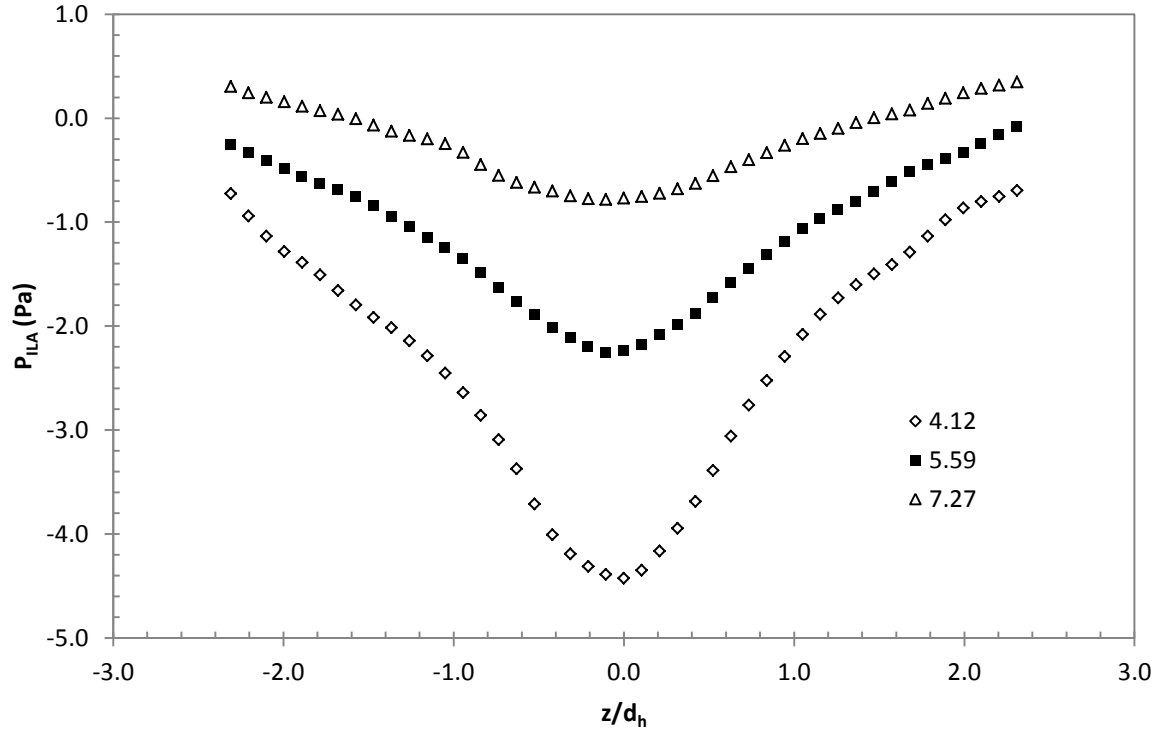

Figure 105: Forced vortex Flow- calculated static pressure on XZ center plane at x/d_h = 4.12, 5.59 and 7.27 based on Instantaneous Lagrangian Acceleration (ILA) method

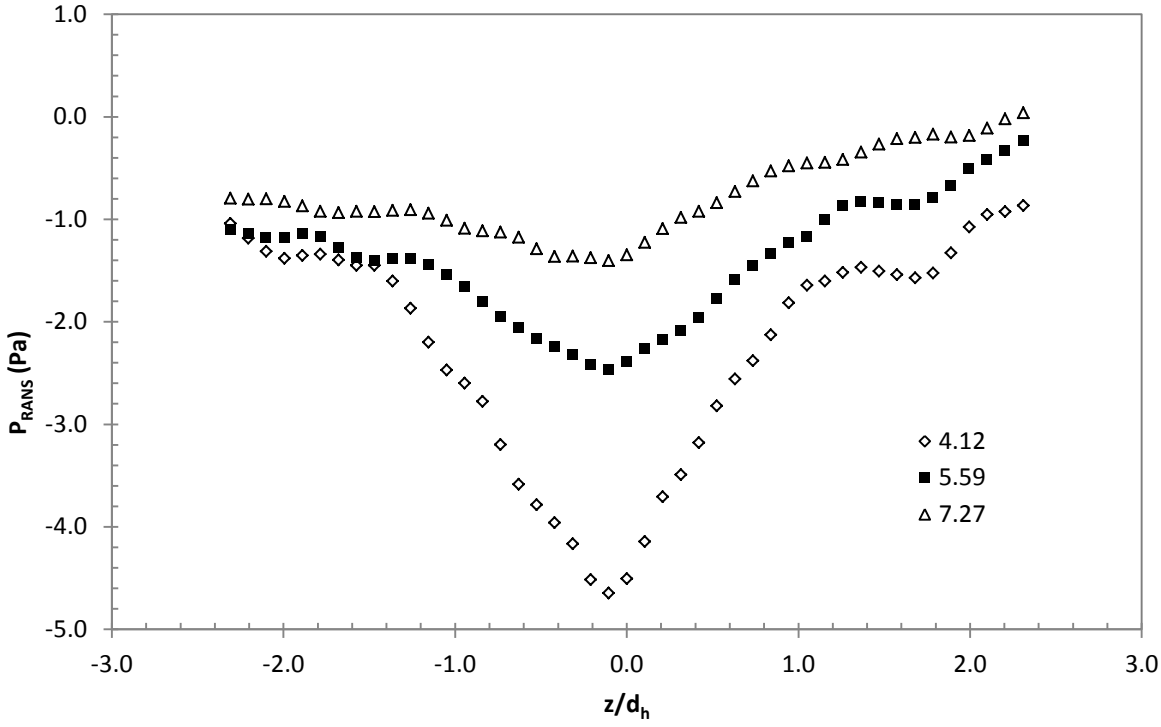

Figure 106: Forced vortex flow - calculated static pressure on the XZ center plane at x/d_h = 4.12, 5.59 and 7.27 based on Reynolds Averaged Navier-Stokes (RANS) method