Development of the 3D Multi-View Particle Tracking Velocimetry with Multi-Pass Robust Initialization Tracking Algorithm

vorgelegt von
M. Sc. Tunç Aşkan
ORCID: 0000-0003-2751-7918

von der Fakultät III - Prozesswissenschaften
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften
- Dr. -Ing. -

genehmigte Dissertation

Promotionsausschuss:

Vorsitzender	: Prof. Dr. Frank Behrendt
Gutachter	: Prof. Dr. -Ing. Martin Kriegel
Gutachter	: Prof. Dr. -Ing. Dirk Müller

Tag der wissenschaftliche Aussprache: 19. August 2019

Berlin 2019

For my two uncles

Abstract

In this study a new flow visualization method the 3D Multi-View Particle Tracking Velocimetry has been developed. This multi-view vision based measurement system can capture large-scale flow structures with optical obstacles. In order to increase the tracking performance of this method, it is enhanced with a novel tracking algorithm, the multi-pass tracking algorithm with robust initialization.

The use of multiple webcams in this fluid visualization method reduces the hardware costs significantly and makes it feasible to capture flow structures for various types of flow. The workflow developed for this measurement system enables a systematic execution of the measurements. The conventional multi-view camera calibration process with bundle adjustment algorithm is improved with an image filtering step, which reduces the resulting mean reprojection error by 90%. In order to find the camera set with the lowest triangulation error, two multi-view triangulation approaches are developed and compared. In this context, the results of the multi-view triangulation with precalculation were closer to the lowest possible triangulation error than the multi-view triangulation with elimination.

In order to increase the capabilities of the development process, a software-in-the-loop environment has been developed as well. Using this software-in-the-loop environment, the multi-pass tracking algorithm with robust initialization is compared with a conventional tracking algorithm. The results show that this tracking algorithm delivers significantly higher tracking efficiencies which do not decrease dramatically with ascending seeding rates. Finally, experiments for an indoor flow case were carried out using this new measurement system. The comparison of the experimental results with a comparison measurement using hot-sphere probes showed that this measurement system delivers plausible results.

Kurzfassung

In dieser Studie wurde eine neue Strömungsvisualisierungsmethode, 3D Multi-View Particle Tracking Velocimetry, entwickelt. Dieses Multi-View Vision basierte Messsystem kann großräumige Strömungsstrukturen mit optischen Hindernissen erfassen. Um die Trackingfähigkeit dieser Messmethode zu erhöhen, setzt es einen neuartigen Tracking-Algorithmus ein, den Multi-Pass Tracking-Algorithmus mit robuster Initialisierung.

Die Verwendung mehrerer Webcams in dieser Strömungsvisualisierungsmethode reduziert die Hardwarekosten erheblich und ermöglicht es, Strömungsstrukturen für verschiedene Strömungsfälle zu erfassen. Der für dieses Messsystem entwickelte Workflow erlaubt eine systematische Durchführung der Messungen. Die herkömmliche Methode der Multi-View Kamerakalibrierung mit Bündelblockausgleichungsalgorithmus wird durch eine Bildfilterung verbessert, die den resultierenden mittleren Reprojektionsfehler um 90% reduziert. Um das Kameraset mit dem geringsten Triangulationsfehler zu finden, werden zwei Multi-View Triangulationsansätze entwickelt und verglichen. In diesem Zusammenhang waren die Ergebnisse der Multi-View Triangulation mit Vorberechnung näher am kleinstmöglichen Triangulationsfehler als die Multi-View Triangulation mit Elimination.

Um die Fähigkeiten des Entwicklungsprozesses zu erhöhen, wurde auch eine Software-in-the-Loop Umgebung entwickelt. In dieser Software-in-the-Loop-Umgebung wurde der Multi-Pass Tracking-Algorithmus mit robuster Initialisierung mit einem herkömmlichen Tracking-Algorithmus verglichen. Die Ergebnisse zeigen, dass dieser Tracking-Algorithmus eine signifikant höhere Trackingseffizienz liefert, welche mit steigender Seedingmenge nicht dramatisch abnimmt. Schließlich wurden mit diesem neuen Messsystem Experimente für einen Innenraumströmungsfall durchgeführt. Der Vergleich der experimentellen Ergebnisse mit einer Vergleichsmessung mit Hitzdrahtanemometern zeigt, dass dieses Messsystem plausible Ergebnisse liefert.

Acknowledgments

This study has been carried out between 2009 and 2018 at the Hermann-Rietschel-Institut of the Technische Universität Berlin. I am grateful to my supervisor Prof. Dr. -Ing. Martin Kriegel for his supervision and for motivating me to carry on my research on this topic. I would also like to thank Prof. Dr. -Ing. Dirk Müller for his critical and constructive comments on this study. In the last 9 years the staff of the Hermann-Rietschel-Insitut provided a great work atmosphere. The numerous discussions in and out of work were really illuminating. Thank you very much for these nice 9 years! I am also thankful to Logitech GmbH for their hardware support for this work.

I am particularly grateful to my grandfather, who educated me with handcraft skills which were indispensable during the experimental part of my research. Thank you very much Dr. Tamer Söyler for the long interdisciplinary discussions. My special thanks goes to Cem Aybek, who introduced me to a beautiful Mediterranean nature and encouraged me to also focus on the flow of life during this work. Reena Agarwal and Stefan Sayer made an enormous contribution with their proofreading and comments, thank you very much! I appreciate my family and my friends, who also enjoyed the life with me during my research. My deepest thanks to Lise for her support. This has been not only in the last years, but since I have known her. Finally, thank you to Sasha for your never ending smile.

Contents

List of Tables

List of Figures

List of Symbols

Abbreviations

$3D-MVPTV$	3D Multi-View Particle Tracking Velocimetry
APNR	Automatic Plate Number Recognition
CFD	Computational Fluid Dynamics
CL	Camera link
COTS	Commercial Off-The-Shelf
CPU	Central Processing Unit
CSV	Comma Separated Values
FHD	Full High Definition, 1920 × 1080 pixels
GPU	Graphics Processing Unit
LED	Light-emitting diodes
LIF	Laser Induced Fluorescene
LM	Levenberg-Marquardt Algorithm
MP	Megapixel
PIV	Particle Image Velocimetry

PST	Particle Streak Tracking
PTV	Particle Tracking Velocimetry
RGB	Red Green Blue
SIL	Software-in-the-Loop

Subscripts

Δz_0	Light sheet thickness
η_T	Tracking efficiency
Λ_0	Average displacement between objects
Λ_t	Average displacement of the objects between the two following frames
c_x	X coordinate of the camera center on the image plane
c_y	Y coordinate of the camera center on the image plane
d_e	Resolution cell diameter
d_I	Interrogation cell diameter
f_L	Focal length
f_{max}	Maximum frame number
N_C	Number of cameras
N_I	Image density
N_S	Source density
N_F	Number of frames
$N_{o,f}$	Number of objects in the f^{th} frame

$N_{T,i}$	Number of tracks in image space
$N_{T,w}$	Number of tracks in world space
$N_{VT,i}$	Number of virtual tracks in image space
$N_{VT,w}$	Number of virtual tracks in world space
p_f	Object position in image space for the f^{th} frame
r_p	Particle spacing displacement ratio
r_s	Search radius
r_{max}	Maximum radius
S_V	Virtual tracking vector
v_x,v_y,v_z	Velocity magnitude components
v_{max}	Maximum velocity magnitude

Symbols

α	Under relaxation factor
Δt	Time between frames
ρ	Density
σ	Standard deviation
A	Object area
B	Transformation matrix
C	Mean number of particles per unit volume
D	Tracer particle diameter

F	Fundamental matrix
f	Frame number
H	Tracking decision matrix
K	Initialization decision matrix
M	Image magnification
R	Rotation matrix
r	Radius
S	Tracking vector
s	Pixel density
T	Translation vector
v	Velocity magnitude
x, y	Components of image point coordinates
X, Y, Z	Components of world point coordinates

Superscripts

Δp^f	Distance between estimated and actual positions of an object for the f^{th} frame
$\overline{D}$	Mean tracer particle diameter
$\vec{v}$	Velocity vector
d''	Midpoint deviation for five frames
d'	Midpoint deviation for three frames

f'' Average distance between calculated and real midpoints in first, third and fifth frames of five consecutive frames.

f' Average distance between calculated and real midpoints in three consecutive frames.

1 Introduction

Research studies on the numerical modeling of fluid flows have been continuously increasing in the last few decades. One of the important factors supporting this development is the significant improvements in computation capacities. The fluid visualization is another crucial method utilized to understand various flow phenomena. These fields are strongly linked with each other and advanced fluid visualization methods are especially vital for the validation of the numerical models. Another technology field improved rapidly in the last few decades is imaging technologies, which increases integration of computer vision based applications in our daily lives. The fluid visualization profits from the imaging and computer vision technologies however its improvement is slower compared to numerical modeling of the fluid flows.

Nowadays the conventional fluid visualization applications are restricted to small observation volumes. This is due to the use of two to four costly scientific cameras in volumes without optical obstacles. However, the latest improvements in the imaging, computer vision and computational technologies can overcome such a restriction and spread the usage of flow visualization. This study represents the development of a new flow visualization method based on multi-view vision, the 3D Multi-View Particle Tracking Velocimetry (3D-MVPTV). Its first scope of application was large-scale indoor air flows however it can also be utilized for other flow structures. During the development of this measurement system several properties such as accuracy, applicability, ease of use, flexibility, and affordability were focused on. This study has two main focus points. The first focus point, which is crucial for the implementation of multi-view vision in flow visualization is the calibration of multiple cameras. The second focus point is object tracking. To increase the tracking efficiencies compared to the conventional tracking methods, a new tracking algorithm, the multi-pass tracking algorithm with robust initialization, is developed.

This study consists of eight chapters. Initially there is an *Introduction* and then chapter two *Fundamentals* looks at the fundamentals of flow visualization. Here, the brief historical background of the flow visualization is described and then is followed by the common flow visualization techniques. Following this the main components of a flow visualization system are represented in detail. This chapter also describes the fundamentals of computer vision. At this point, the mathematical description of a computer vision system is shown through its analogy to the human visual system. That is followed by the representation of the two-view vision, namely the stereoscopy. Finally, in this chapter multi-vision is described and the bundle adjustment algorithm which is used to define a multi-view vision setup is shown.

The third chapter *3D Multi-View Particle Tracking Velocimetry* exhibits the 3D-MVPTV, which is the flow visualization technique developed in this study. Here, first the concept of the 3D-MVPTV and its background are represented. That is followed by the definition of the 3D-MVPTV workflow, which includes the main steps of this flow visualization method. Before the definition of this workflow, the software environment used for the development is represented. In this workflow, the steps camera distribution, frame synchronized video capture, object recognition, track matching, multi-view triangulation, and post-processing are described in detail. As the main focus of this work are the steps camera calibration and object tracking, they are represented later in separate chapters.

In the fourth chapter, *Real and Virtual Development Environments for 3D-MVPTV*, the two development environments used in this work are described. Here the 3D-MVPTV test stand represents the experimental facility in the Hermann-Rietschel-Institut. This test stand was utilized for the first tests, camera calibrations, and 3D-MVPTV measurements. However, the use of a real experimental environment during the 3D-MVPTV development was impractical and limited, a software-in-the-loop environment (SIL) has been developed. The details of the main steps of this SIL environment are also described in this chapter.

Chapter five *Calibration of Multiple Cameras* first exhibits the stereo camera calibration process. Following that there is a description of the multiple camera calibration. To increase the accuracy of the calibration process, the existing bundle adjustment algorithm is extended with an image filtering step. At the end of this chapter, the common factors to consider by choosing the camera set for multi-view triangulation are represented. This part is one of the most important parts in this work since it exhibits the essential information regarding the practical implementation of the multi-view vision in the developed measurement system.

The sixth chapter *Multi-Pass Tracking Algorithm with Robust Initialization* represents the object tracking algorithm developed for the 3D-MVPTV. Before that, it describes the existing tracking algorithms and the details of the conventional tracking algorithm which is utilized during the first tests with the 3D-MVPTV. After the detailed definition of the multi-pass tracking algorithm with robust initialization, the SIL simulations and their results are exhibited in which the performance of this tracking algorithm is tested. In this context, two different types of SIL simulations were carried out with mathematically defined tracks and with tracks generated by computational fluid dynamics (CFD) simulations.

In the seventh chapter *3D-MVPTV Measurements* first, the experiments with this recently developed flow visualization method are represented. Following that the comparison measurement performed with hot-sphere probes are described. These comparison measurements have the aim to check the plausibility of the 3D-MVPTV results compared with conventional measurement equipment. At the end of this chapter, the possible error sources for the 3D-MVPTV and comparison measurements are discussed.

In the last chapter *Conclusion and Outlook*, the results of this study regarding camera calibration, object tracking algorithms, and 3D-MVPTV measurements are summarized and discussed. This chapter also represents the possible improvements can be carried out in the 3D-MVPTV workflow and for the 3D-MVPTV itself. Finally, the essential points for the further development of this measurement system are presented.

2 Fundamentals

This chapter describes the theoretical background of two important disciplines. The first is flow visualization and the second is computer vision, on which the 3D-MVPTV bases. Following this, the chapters then refer to these theories and represents the application developed in detail. Hence, the circumference of these both fields are quite comprehensive and the main focus stays on the subjects which are crucial for the application in this work which are the three-dimensional spatial and temporal velocity measurement of indoor air flows.

2.1 Fundamentals of Flow Visualization

Flow visualization is the field of science which deals with capturing flow phenomena on a medium and processing it to gain qualitative and often quantitative information about the desired physical variable or variables. The history of flow visualization goes back to prehistoric ages, where our ancestors used basic fluid mechanics knowledge for designing things such as boats and water supply systems. The first flow visualization studies, were done 500 years ago at the time of Leonardo da Vinci. With his artistic and scientific skills, he put various flow phenomena on paper like water flow over obstacles, the motion of clouds, and pouring water into a pool as shown in figure 2.1. This kind of visual documentation, which we today call scientific imaging, opened up new possibilities for developing theories on various physical phenomena.

After the invention of photography at the beginning of the 19th century and the introduction of the first commercial camera at the end of the same century, the image capture process of flow visualization gained new dimensions. Complex unsteady flow structures, which were not possible to capture before, now could be saved even on a video film. Theories describing such flow structures could then be developed using these visual data. Scientists like Osborne Reynolds, Ludwig Prandtl, and Ernst Mach utilized flow visualization while developing the fundamental theories of fluid dynamics. However, it was found that usage of photography and video was not the only improvement in flow visualization and scientists also developed different methods according to the needs of their applications. For example, Osborne Reynolds used dyed water for his well-known research on pipe flow. As we can see through the historical development of fluid visualization, these historical advances in imaging technologies made it possible for scientists to profit more from flow visualization, as well as those who had no artistic skills like Leonardo da Vinci. That helped them to invent further theories for fluid flow.

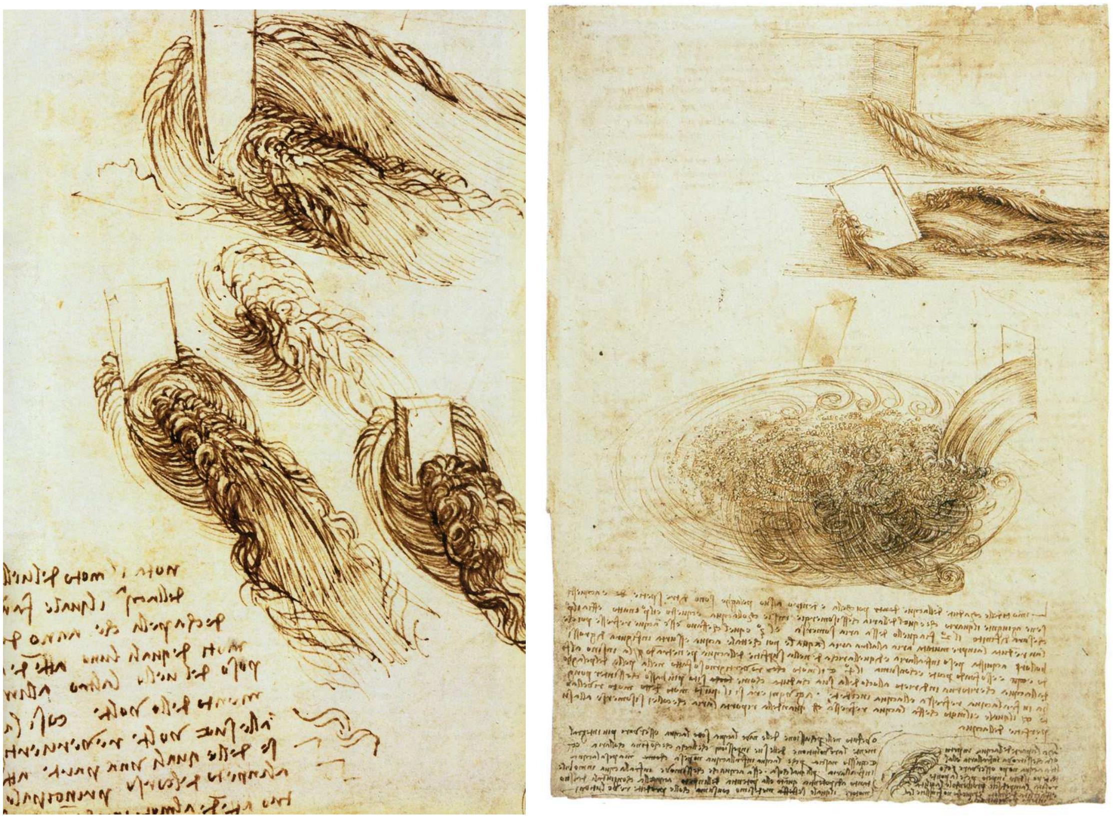


Figure 2.1: Drawings of Leonardo da Vinci for different flow structures [12]

In the last decades beside the advances in imaging technologies, improvements in computer technologies allowed scientists to go even further with flow visualization. With continuously increasing computational power, they started to use flow visualization not only to generate qualitative flow images but also to extract more quantitative data from these images. On the other hand, the advances in computer technologies did not just allow flow visualization techniques to go further but also other fields in fluid mechanics like numerical simulations. Using more computational resources, we became able to solve complex flow problems using computational fluid dynamics, which yielded to develop more detailed physical models like numerous turbulence models.

This increasing utilization of numerical methods in fluid mechanics made flow visualization a standard validation tool for the numerical simulations and a standard verification tool for the developed physical models. Today, modern flow visualization techniques can capture time-resolved two and three-dimensional flow field data with high spatial resolution. However, it is still not possible or computational too expensive to solve numerous flow problems numerically. Peter Freymuth explains this aspect in his article on flow visualization in fluid mechanics as follows: „First, one can solve only a few fluid mechanics problems purely by mathematics. Second, a renewed need for flow visualization arises directly as a result of computational fluid dynamics.“ [19]

2.1.1 Flow Visualization Techniques

Figure 2.2 shows the function principle of flow visualization based on its analogy to the human visual pathway. The fluid flows either with or without additives are captured under specific light circumstances using a camera. In this process chain, a camera works like a human eye, where light signals are converted to electrical signals. These electrical signals are then sent to the microprocessor in which useful information is extracted from them similarly in the human brain. This information then can be displayed and stored depending on the objective of the flow visualization carried out.

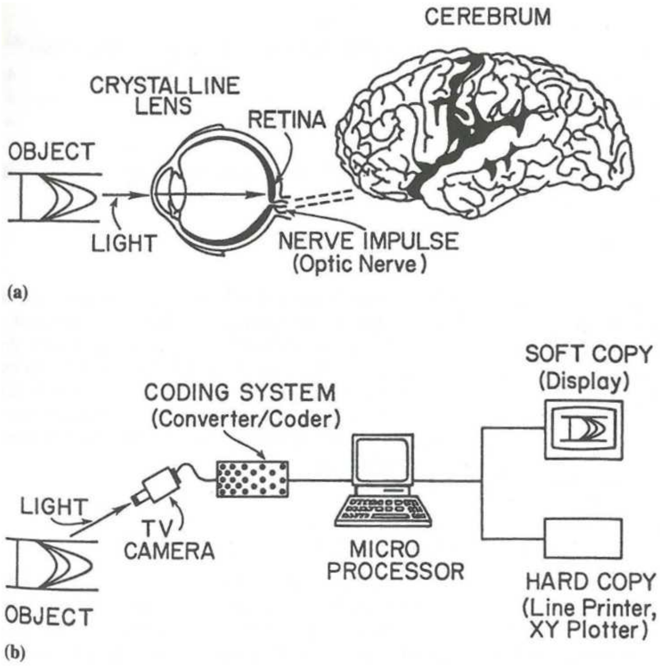


Figure 2.2: Analogy between the human visual pathway and flow visualization

[46]

Depending on the needs of scientists there are various flow visualization techniques and they are commonly based on the function principle described above. Researchers classify these techniques through various perspectives in different ways to achieve an overview in this broad scientific field. For example Merzkirch groups the flow visualization methods into three groups [34]. In the first group, a foreign material (so-called tracer particle) is brought into the flow which is visible and behaves similarly to the investigated fluid. The flow visualization is done indirectly through this material. In the second group, the refractive index of the fluid is used. Through the change in material properties (e.g., change of density by compressible flows) optical phases in the fluid are formed which can be directly visualized using the refractive index. The third group is similar to the second group; however, the optical phases are formed through external sources (e.g., an external energy supply).

Wen-Jei Yang groups flow visualization methods mainly in two groups: first and second generation [46]. He also uses the terms classical and modern flow visualization to distinguish these two groups. The main difference between these two groups is that in modern/second generation methods, quantitative information is extracted using computational post-processing. In the same way, a similar category „quantitative flow visualization“ is used by M. Gharib and D. Dabiri [1]. Since the term „flow visualization“ emphasizes a qualitative character, it is better and less confusing to use the type of information produced by flow visualization techniques for general classification, namely: qualitative and quantitative methods.

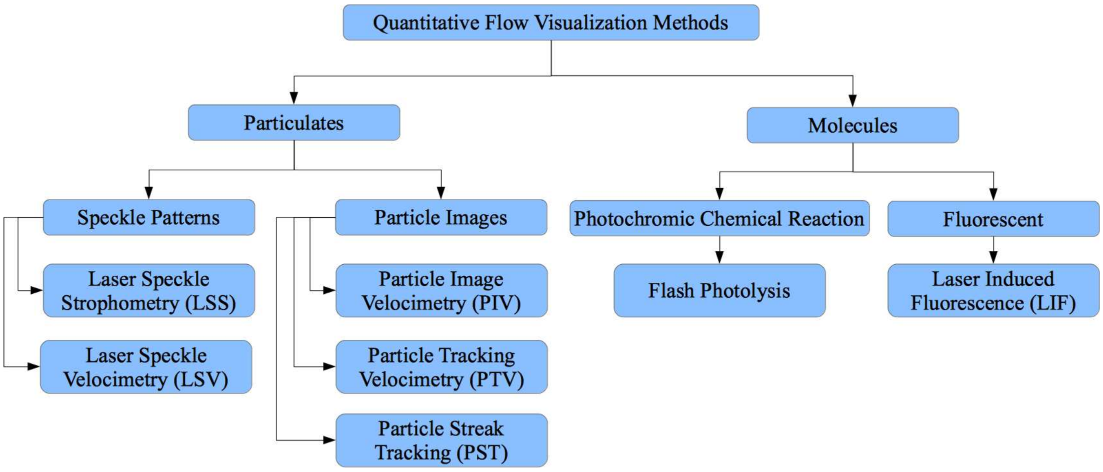

Quantitative Flow Visualization Methods
Particulates
Molecules
Speckle Patterns
Particle Images
Photochromic Chemical Reaction
Fluorescent
Laser Speckle Strophometry (LSS)
Laser Speckle Velocimetry (LSV)
Particle Image Velocimetry (PIV)
Particle Tracking Velocimetry (PTV)
Particle Streak Tracking (PST)
Flash Photolysis
Laser Induced Fluorescence (LIF)

Figure 2.3: Classification of Quantitative the Flow Visualization Methods based on Wen-Jei Yang's Classification [46]

As shown in figure 2.3 quantitative flow visualization methods[1] can be classified primarily into two groups according to the participation scale of tracers. In this context, tracers can contribute to the measurement in particulate scale or molecular scale; in molecular scale their molecular properties also play a role in flow visualization. Two

[1] Since the qualitative flow visualization methods are not in scope for this work, they are not included in this classification.

well-known molecular scale methods are flash photolysis and Laser Induced Fluorescence (LIF). In the flash photolysis, the flow field is first exposed to a strong light source which initiates the chemical reaction to be visualized. This method belongs to the third group in Merzkirch's classification mentioned above, in which change in optical phases is induced through an external source. In contrast to that, the Laser Induced Fluorescence does not need an external source to visualize the molecular scale. Here the utilized fluorescent tracer particles emit illumination with a longer wavelength than the underlying flow medium, which can be captured using, e.g., a narrow-band filter.

The particulate scale methods can differ from each other using source density N_S as shown in equation 2.1 according to Adrian [2]. In this equation, C refers to the mean number of particles per unit volume, Δz_0 the light sheet thickness, d_e the diameter of the so-called *resolution cell*, and M the image magnification. $N_S \gg 1$ yields more than one particle in a resolution cell, which results in overlapping particles on the image plane. In this case, it is not possible to work with individual particles any more so speckle patterns must be analyzed to achieve flow information. If, on the other hand, $N_S \ll 1$ it is possible to work with particles instead of speckle patterns.

$$N_S = C\Delta z_0 \frac{\pi {d_e}^2}{4M^2} \tag{2.1}$$

Similarly, Adrian [2] defines the image density N_I as shown in equation 2.2 to distinguish between particle image-based flow visualization methods. He defines *interrogation cell* with a diameter d_I in which the image density is calculated. If $N_I \ll 1$ the probability for finding more than one particle in the interrogation cell is small, and such a system can be classified as *low image density*. If, on the other hand, $N_I \gg 1$ the probability to find more than one particle in an interrogation cell is high, which is defined as *high image density*.

$$N_I = C\Delta z_0 \frac{\pi {d_I}^2}{4M^2} \tag{2.2}$$

In the following sections, three particle image-based flow visualization methods are described. Particle Image Velocimetry (PIV), Particle Streak Tracking (PST), and Particle Tracking Velocimetry (PTV) are the most frequently used flow visualization methods for indoor air flows. These methods use different approaches to capture the flow field, which yields results with different temporal and spatial resolutions.

2.1.1.1 Particle Image Velocimetry (PIV)

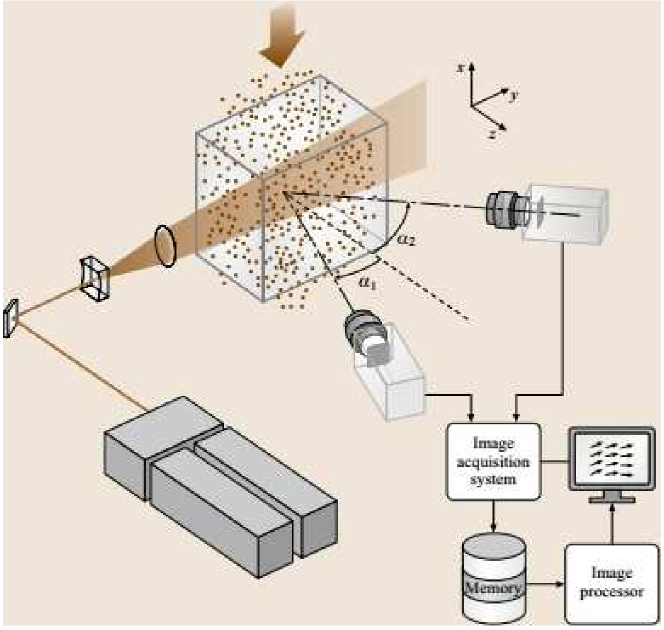


Figure 2.4: Functional Schema of Particle Imaging Velocimetry [7]

Figure 2.4 represents the functional principle of PIV. In this flow visualization method, tracer particles are exposed to a pulsed laser light sheet while they move through the observation volume. The motion of these tracer particles is then captured using video cameras [2]. The captured images are then processed using a computer. The primary objective of this computational process is to recognize the captured tracer particles and finding the same objects in two subsequent frames so that velocity vectors can be calculated using this information. This object matching step is generally based on a cross-correlation algorithm.

2.1.1.2 Particle Tracking Velocimetry (PTV)

Regarding its experimental setup, the PTV method is quite similar to PIV. The main difference with this method is that every tracer particle entering the observation volume is tracked separately, namely instead of a Eulerian approach a Lagrangian approach is used. In order to track every tracer particle, the flow field is seeded with lower image densities. Adrian emphasizes this point as follows: „Hence, the low-image-density mode of PIV is often referred to as particle tracking velocimetry or PTV.“ [2]. Beyond the image density Malik, Dracos, and Papantoniou define a so-called *particle spacing displacement ratio* for PTV as shown in equation 2.3 [38] [7]. In this equation, Λ_0 defines the average displacement between objects and Λ_t is the average displacement of the objects between two consecutive frames. For $r_p \gg 1$ particle tracking is simple, and its difficulty increases around $r_p \approx 1$. If $r_p \ll 1$ particle tracking becomes impossible.

[2] Since nowadays primarily digital cameras are used for fluid visualization, the adjective „digital“ is renounced for cameras in this work.

$$r_p = \frac{\Lambda_0}{\Lambda_t} \tag{2.3}$$

Besides its low resolution due to seeding with lower image densities, PTV results include, contrary to other methods, spatial and temporal information of each tracer particle. Using this information content, in addition to the velocity vectors further scalars like residence time can be calculated. Ventilation effectiveness, which is an important parameter for indoor air environments, can be extracted in such a way.

2.1.1.3 Particle Streak Tracking (PST)

The PST method focuses on capturing streaks of the tracer particles and extracting velocity vectors through these streaks. In order to generate particle streaks, long exposure times with a photo camera and a pulsing light sheet are utilized. The light sheet pulse durations are varied, which results in different stroke lengths between two subsequent images to determine the flow direction as shown in figure 2.5. The magnitude of the velocity vector is then calculated using the distance between two consecutive strokes. Using this method flow visualization can be carried out either in 2D or 3D: A 3D measurement runs with two cameras and multiple light sheets [37].

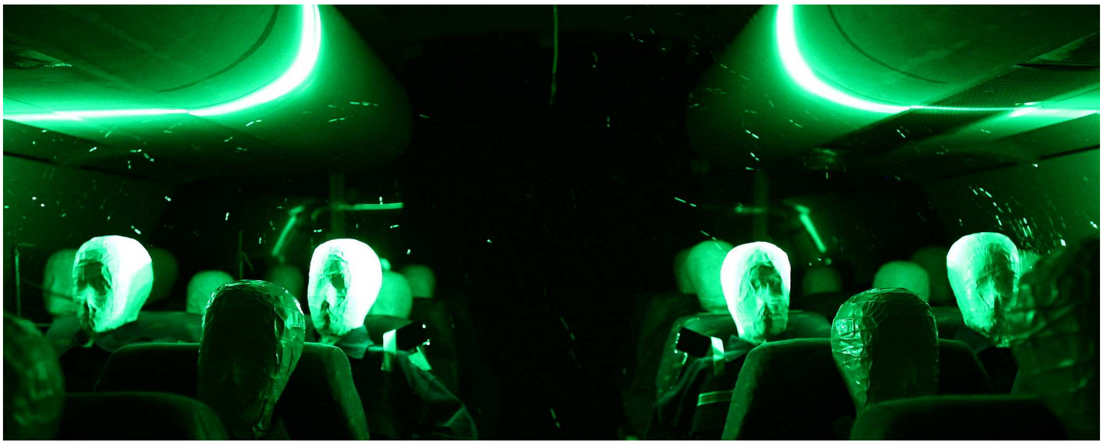


Figure 2.5: An image from a 2D-PST measurement in an aircraft cabin [23]

2.1.2 Main Components of a Flow Visualization System

The previous section described the main function principle of flow visualization and represented a general classification of quantitative flow visualization methods. This was then followed by the theoretical background of three common particle-based flow visualization methods. This section describes the main components of these methods in detail. Based on an analogy with the human visual pathway, as shown in figure 2.2, the components of a flow visualization system can be categorized into four groups: tracer particles, illumination, image acquisition system, and data processing unit.

2.1.2.1 Tracer Particles

Tracer particles are the interfaces in flow visualization between the image acquisition system and the investigated flow structure. They make an invisible fluid motion visible so that an optical device can capture it. There are various tracer particles available for different flow visualization methods in different fields of application. However, there are some common properties which they must satisfy. Since flow visualization is a non-intrusive measurement method, tracer particles must not influence the flow phenomenon investigated and must behave as close to the fluid in motion as possible. An important condition for this is neutral buoyancy, which becomes more important with decreasing velocities in the flow, e.g., natural convective flow. Polyethylene ($\rho = 0.920 g/cm^3$) and polyamide ($\rho = 1.15 g/cm^3$) microspheres are the most common materials used for visualizing water ($\rho = 1 g/cm^3$) flows. The number of possible neutrally-buoyant tracer particles decreases when the investigated fluid is a gas like air. There are few materials in nature with a density equal or less than air. Furthermore, most of these materials are either explosive (like methane, hydrogen, and nitrogen) or rare and expensive (like neon). Under these circumstances, helium is the most used gas as a tracer particle for air flows, in the form of encapsulated in soap bubbles. Foss, Tropea, and Yarin emphasize another important property of the material used as tracer particle, its chemical character: „The material should not be hazardous or toxic if inhaled, it should not be corrosive or reactive when in contact with parts of the flow facility or other instrumentation.“ [7].

Besides its physical properties, optical properties of the tracer particle in combination with the illumination source and the optical capture device also play an important role. Scattering properties of the tracer particles are most important in this context: The tracer particle should scatter the light exposed to it homogeneously on its surface. For solid opaque tracer particles, this requirement is not as challenging as for transparent helium-filled soap bubbles which are used for air flows. As shown in figure 2.6 right, helium-filled soap bubbles do not scatter the exposed light homogeneously but locally as spots. This phenomenon is measured for a water droplet in the air as shown

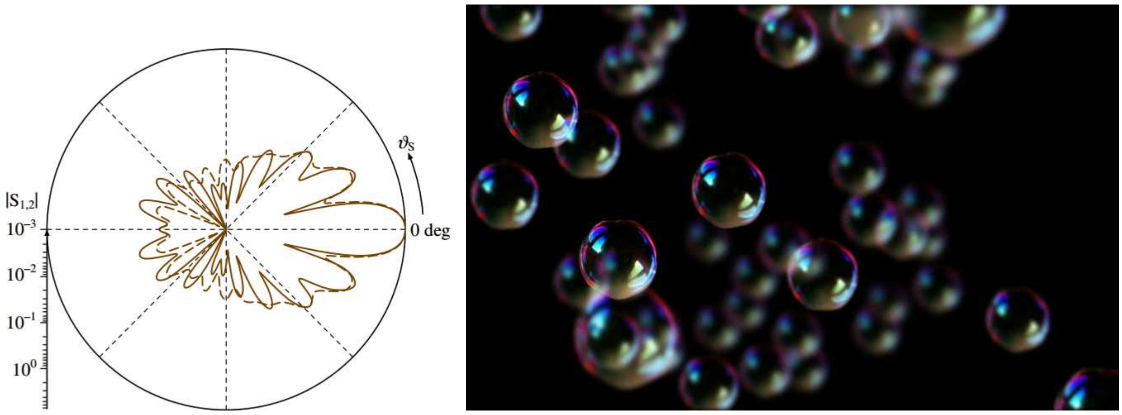


Figure 2.6: Mie scattering diagram of a water droplet in the air [7], soap bubbles scattering in different directions [43]

in figure 2.6 left. As can be seen from this scattering diagram, scattering on a water droplet in the air has four peaks in the direction of observation. This inhomogeneous scattering leads to an error during the determination of tracer particles' centers, which propagates to an error in velocity calculated and therefore must be considered during error estimation.

In combination with the used illumination sources, tracer particles should also deploy a high contrast to the background in order to make it possible to recognize them in the next step. In this context, Foss, Tropea, and Yarin mention the big advantage of fluorescent tracer particles as follows: „This makes it possible to separate the tracer particles from other scattering objects, such as walls, bubbles, droplets, and other particles, by means of an appropriate optical filter that blocks the light with the wavelength of the incident light (viz. light sheet) and passes the fluorescent light.“ [7] Other essential requirements for the tracer particles are their stability with varying flow conditions (e.g., temperature), their low-cost, and—preferably—biodegradability.

2.1.2.2 Illumination

The task for the illumination source in flow visualization is to expose the observed domain with light so that the image acquisition system can capture the tracer particles. The light produced by the illumination source must exhibit a specific intensity and a homogeneous distribution. Illumination in flow visualization systems can be categorized according to their implementation in two groups as planar and volumetric. Methods like PIV (2D & 3D), PST (2D & 3D), and 2D-PTV measure velocity components on planar light sheets, however, 3D-PTV tracks volumetric illuminated tracer particles.

PIV uses laser light sheets, mostly Nd:YAG lasers. The advantages of the laser are mentioned by Foss, Tropea, and Yarin as: „Lasers are used as illumination sources since they can produce a pulsed, collimated and monochromatic light beam that can be easily shaped into a thin light sheet." [7] A disadvantage of laser illumination is the lacking laser safety: It is not possible to use such a system in every environment (e.g., field measurements). Müller utilized a gas-discharge lamp for 2D and an argon-ion laser for 3D PST applications [37]. Biwole used halogen lamps for volumetric illumination in his study on a large scale PTV [6]. He emphasizes in this work the low heat generation rate of these illumination sources, which is an essential factor to consider, especially for low-velocity measurements where convective effects have a significant influence on the flow field. Groß, Brevis, and Jirka used LED (light-emitting diodes) illumination sources for PTV and PIV measurements in their research [11]. According to this study LED lamps have numerous advantages like safety, flexibility, and being economical compared to laser illumination systems.

2.1.2.3 Image Acquisition System

The image acquisition system has the task to capture images of the illuminated tracer particles in the observed flow field using one or more cameras and store them so that further data processing can be carried out in the next steps. Such a system should deliver images with high quality, resolution, and color depth. At the same time, the camera speed should be preferably controllable and accurate, especially for multi-camera systems. The image sensor, which converts optical into electrical signals, defines the resolution and color depth of a camera. The total number of pixels on it defines the resolution of the image sensor in megapixels (MP). The color depth stands for the resolution of color in bits, e.g., $8-bit$ has $2^8 = 256$ tones to represent a color. Both the lens and the image sensors have an impact on the quality of the captured images. It should have a wide and controllable aperture so that as much as possible scattered light can be projected on the image sensor and the depth of focus can be controlled depending on the investigated flow field.

The camera speed, images captured per second (frames per second, fps), should be chosen according to the observed flow speed so that the tracer particles are captured without any blur. Shutter speed accuracy should also be high so that the time deviation between successive frames stays low. This is especially important when multiple cameras are used. In that case, all cameras must capture all frames at the same time; a time shift between the dataset from different cameras could lead to erroneous results.

Figure 2.7 shows a high-speed scientific camera which is used for PIV measurements. It can capture images with a resolution of four megapixels and up to 400 frames per second.

Figure 2.7: A high-speed scientific camera [4]

An image acquisition system consists not just of cameras, but it also contains usually a workstation which controls these cameras and stores the acquired data from them for further data processing. There are various interface standards between workstation and cameras. The most used interface standard for the high-speed scientific cameras nowadays is *Camera Link (CL)*, which can run at a bandwidth of up to 850*MBs* [45]. For the flow visualization methods where the synchronization between illumination source and cameras is crucial (e.g., PST), the image acquisition system also has the task to control the illumination system.

2.1.2.4 Data Processing Unit

The components of a flow visualization system described up to this point are responsible for the image acquisition, namely transferring a flow phenomenon to a storage medium as Leonardo da Vinci made 500 years ago as shown in figure 2.1. At the time he was not capable extracting quantitative information from his images, whereas now with increasing computational resources we can gain more quantitative information from these images. However, as described by Adrian, this step is the most challenging part of the whole workflow: „The analysis of the recorded image field is one of the most important steps in the entire process, as it couples with the image-acquisition process to determine the accuracy, reliability, and spatial resolution of the measurements; it is also the most time-consuming part of the process.“ [2] The data processing unit is generally a computer (e. g., workstation, cluster) on which captured images are processed using software to gain quantitative flow information. Section 3.2 describes the details of this data processing step. The requirements of the data processing unit depend on the observed system, but generally, it should satisfy enough computational power, storage, and memory for the concerning task.

2.2 Fundamentals of Computer Vision

Computer vision is the scientific field which deals not only with the visual capture of the real world but also the analysis and interpretation of these images. In other words, computer vision generates—using the infinitive visual data from the real world—, numerical results regarding a specific aim. The first computer vision applications were already developed in the 1960s and the first international conference on computer vision took place in London in 1987. The rapid development of information technologies[3] has also influenced computer vision technologies in the last decades. Nowadays it is possible to see computer vision applications in diverse fields of daily life like:

- Automatic plate number recognition (APNR) uses computer vision to save vehicle plate numbers in flowing traffic. With this system, it is possible to charge toll fees without interrupting flow traffic, reducing fuel consumption and emissions.
- Mobile 3D scanners make it possible to quickly generate high-resolution 3D geometry data of objects in the real world. This computer vision application brings new possibilities to workflows in which primarily 3D geometry data are processed.
- Adaptive cruise control systems help to retain a security distance to the vehicle in the front, keeping vehicle velocity, and change the lane automatically using in-vehicle installed cameras, which increases the security in traffic.

Beside these application examples, computer vision is used in various industries like aerospace, medical services, and robotics. In this study, computer vision is used to capture 3D flow velocity. Section 3.2 defines the practical implementation of this application in detail. In the following, the computer vision theory, which is essential for this application, will be described briefly.

2.2.1 The Human Visual System

The eye is the organ responsible for visually capturing the environment. It has the function of converting the 3D world to nerve signals—just like as a camera. This process takes place through three components in the eye: the lens, the pupil, and the retina. The pupil adjusts—like the diaphragm in a camera—the amount of the light entering the eye. This light is projected onto the retina by the lens. The retina with its

[3] The employment growth in information technologies between 1990 and 2011 is circa 300% according to the U.S. Bureau of Labor Statistics.

125 million photoreceptors [21]—like a sensor in a digital camera—converts the light signals to electrical nerve signals.

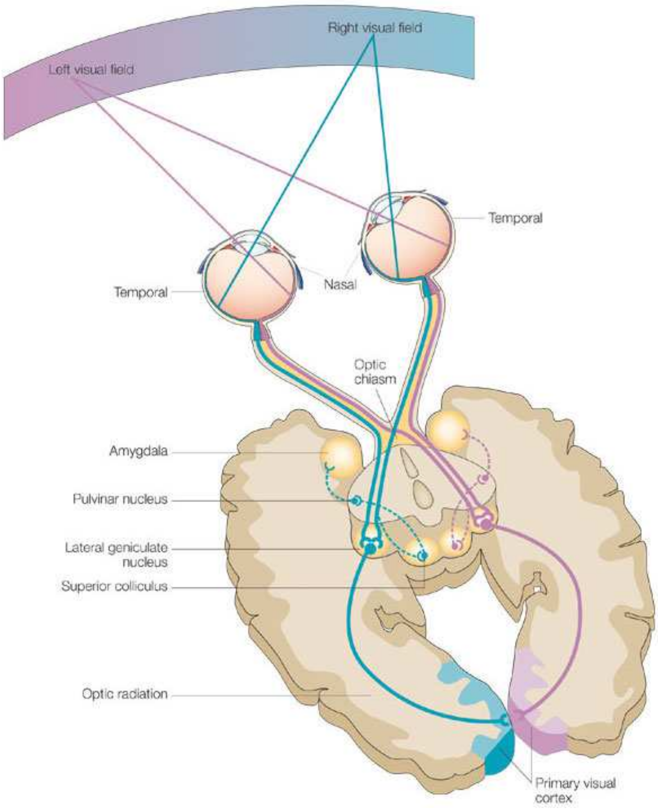
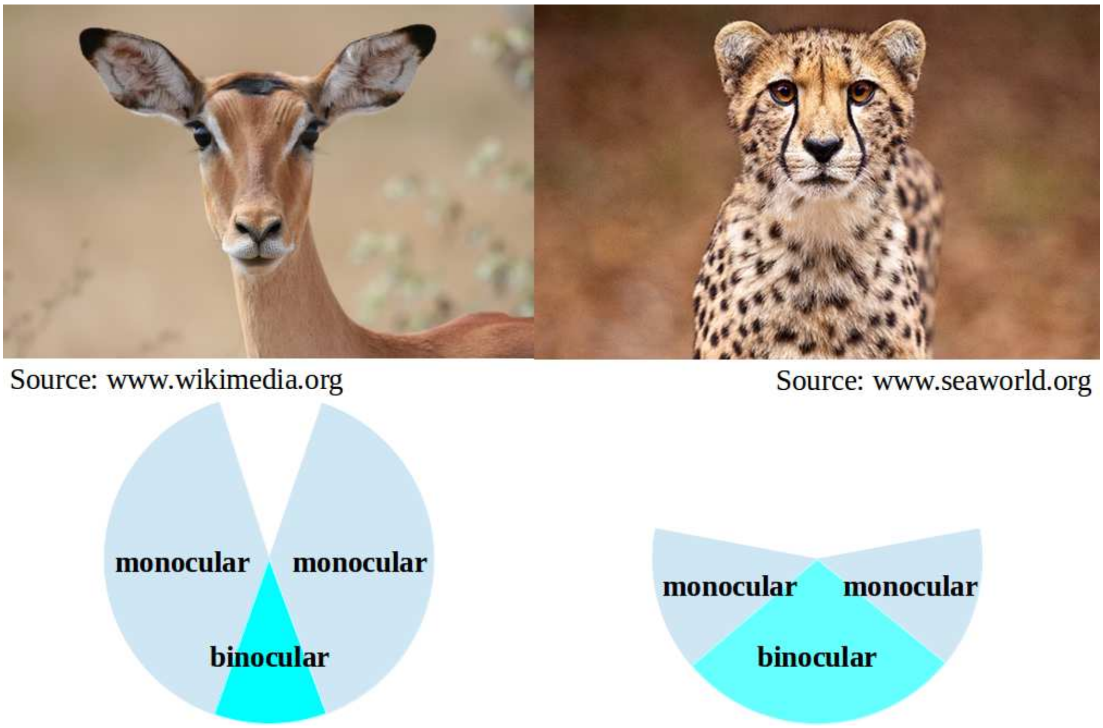


Figure 2.8: Human visual pathway [14]

These nerve signals then are processed in different parts of the brain where the visual perception takes place, as shown in figure 2.8. The depth dimension, which is lost during the projection of the 3D world onto the 2D retina surface, must also be calculated in this step. The monocular vision and the binocular vision are two main processes during this depth estimation. In monocular vision, the signals of just one eye is used, supported by various criteria. One of the most important of these criteria is the motion parallax. When an observer moves (e.g. in a train journey), the closer objects are perceived with a higher relative velocity than the objects further away. Furthermore, learning through experiences is an essential part of the monocular vision. When the absolute size of an object is known (namely learned) the distance to this object can be estimated.

Other than the monocular vision there is the binocular vision where two eyes are used. The depth can be estimated using information from two different perspectives, so-called binocular disparity. Compared to the monocular vision the binocular vision is more precise, which can also be observed in the evolution. The eyes of most of predators are located in the front of the head so that more binocular vision is possible. That is essential because the precise calculation of the distance to the prey is necessary. On the other side for the prey the size of the field of view is more important than the precision of depth estimation to the predator so that they can recognize the dangers earlier. Therefore, preys utilize more monocular vision compared to predators as shown in figure 2.9.



Source: www.wikimedia.org

Source: www.seaworld.org

Figure 2.9: Prey vs. predator and their monocular and binocular field of views

2.2.2 The Pinhole Camera Model and Single Camera Calibration

The pinhole camera is the simplest camera that can be used to capture 3D objects of the real world on a 2D plane. Using this camera, 3D objects are projected on a 2D plane with the light passing through a tiny hole. Because no lens is used in this camera, no image distortion occurs. These optical phenomena were already observed 2500 ago years in China. However, the humankind had to wait 2300 more years till French Nicéphore Niépce made the first photograph of the world on a chemically coated surface in 1826-1827. Nearly 150 years after the first photograph of the world American Steve Sasson developed in 1975 the first digital camera. In a digital camera, light signals are converted to electrical signals using an image sensor. Nowadays this technology is widely used for image and video capture.

Independently of the capture method (photo film, video film or image sensor) the pinhole camera model mathematically defines the projection of 3D objects of the real world on a 2D plane. As shown in figure 2.10, using this model a world point

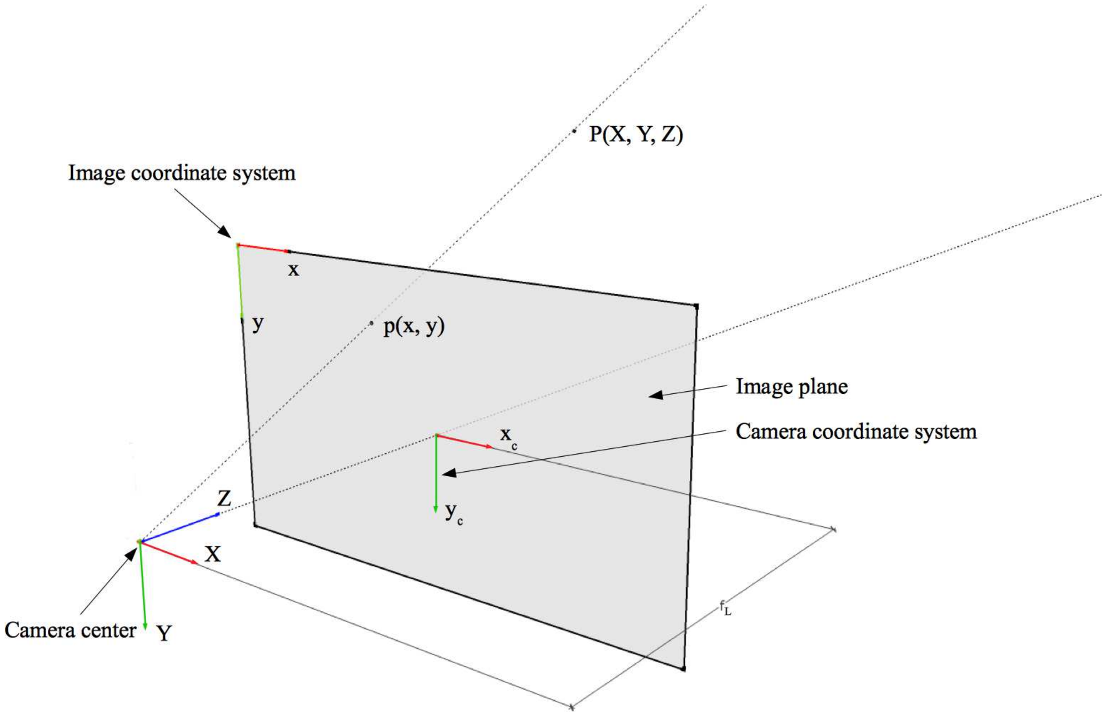


Figure 2.10: The pinhole camera model

$P(X,Y,Z)$, which is defined according to the world coordinate system[4] with its origin in the camera center, is projected onto a 2D plane (e.g., image plane). The result is the image point $p(x,y)$, which is defined according to the camera coordinate system. The image coordinate system is used to define the positions of the objects on the image plane by the capture device. The following equation describes the pinhole camera model:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_L & 0 & c_x & 0 \\ 0 & f_L & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.4}$$

In equation 2.4, f_L defines the focal length; c_x and c_y the origin coordinates of the camera coordinate system according to the image coordinate system. The 4×3 matrix in this equation with the camera parameters is called the camera matrix. Using this equation and a given camera matrix, it is possible to carry out the coordinate transformation from the world coordinate system to the image coordinate system. In other words, given a camera matrix and world coordinates it is possible to calculate the po-

[4] In this study, the coordinate systems are oriented as in MATLAB® [31]

sition of an object on the image sensor. The general form of the camera matrix can be decomposed into two matrices: the intrinsic and the extrinsic matrix.

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} R_{xx} & R_{xy} & R_{xz} & T_x \\ R_{yx} & R_{yy} & R_{yz} & T_y \\ R_{zx} & R_{zy} & R_{zz} & T_z \end{bmatrix}}_{\text{Extrinsic matrix}} \underbrace{\begin{bmatrix} f_L & 0 & c_x \\ 0 & f_L & c_y \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Intrinsic matrix}} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.5}$$

The intrinsic matrix contains parameters which characterize the camera itself as shown in equation 2.5. The extrinsic matrix, on the other hand, is similar to the transformation matrix and defines the orientation of the camera. The first three columns of the extrinsic matrix build the rotation matrix and last column the translation vector.

To use the pinhole camera model the camera matrix must be known. This can be achieved through camera calibration. During this process, a pattern including objects with known world coordinates, usually a checkerboard, is captured by the camera. The resulting image points are then recognized and matched with known world points, so that the camera matrix can be calculated. However, the exact solution of this problem would only be possible if the image points could be determined exactly, which is impossible due to the finite sensor resolution. In order to solve the equation system 2.5, which has three equations and 15 unknowns, at least six points are needed, which results in 18 equations and 15 unknowns. This resulting over-determined equation system is solved by minimizing a cost function. This cost function is generally the mean reprojection error, which defines the average distance between captured and reprojected image points (captured image points and calculated image points using estimated camera matrix). That means the final mean reprojection error also defines the accuracy of the camera calibration.

2.2.3 Stereoscopy and Stereo Camera Calibration

The pinhole camera model defined above is a mathematical model for one camera. Extending this model to two cameras (this method being known as stereoscopy), it is possible to calculate the 3D world coordinates of an object from its image coordinates on each camera sensor. The graphical representation of this problem is simplified to the calculation of a 2D world point using two 1D image points in figure 2.11. This calculation is usually carried out on calibrated cameras whose camera matrices are known.

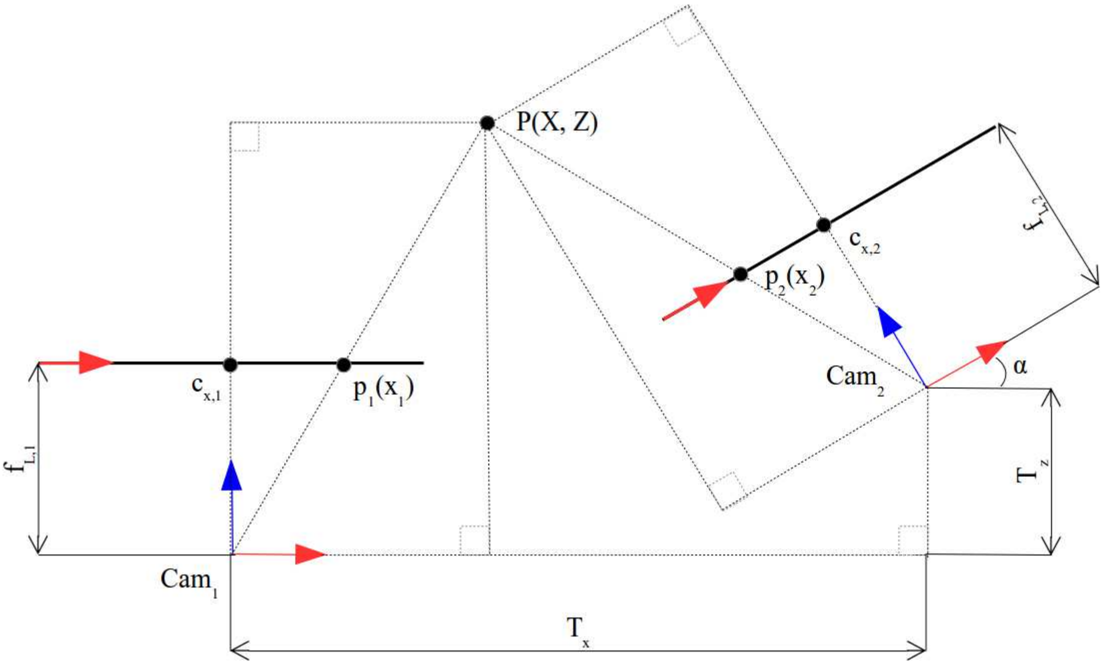


Figure 2.11: Triangulation of a 2D point

$$\begin{aligned} \frac{x_1 - c_{x,1}}{f_{L,1} * s_1} &= \frac{X}{Z} \\ \frac{x_2 - c_{x,2}}{f_{L,2} * s_2} &= \frac{X * cos(\alpha) + T_x}{Z * sin(\alpha) + T_z} \end{aligned} \tag{2.6}$$

In order to find the coordinates of the point P, the pinhole camera model must be applied to both cameras and a coordinate transformation must be carried out according to the camera coordinate system, in which the result is desired (it is often the first camera). The mathematical formulation of this operation is shown in the equation system 2.6. s_1 and s_2 are the pixel densities of the cameras and are used for dimension conversion of the focal length to the image coordinate system, namely to pixels. With its two equations and two unknowns, this linear equation system can be solved directly[5] which is also called triangulation. Similarly, one more dimension can be added to this equation system, which results in a triangulation of a 3D point using two 2D image points.

For the triangulation to a 3D world point using two cameras, first its 2D projections on every camera must be known. In other words, two matched 2D image points are

[5] e.g., using singular value decomposition which is also used by MATLAB®.

necessary to calculate the 3D point they are representing. In this context, the epipolar constraint defines whether two 2D image points on different cameras can represent a 3D world point. Hartley and Zisserman [41] describe the epipolar constraint shown in figure 2.12 as follows: „If the epipolar constraint is satisfied, then these two rays lie in a plane, and so intersect in a point X in 3D-space.“. Equation 2.7 shows the mathematical formulation of this description, whereby x and x' are two corresponding image points and F is the fundamental matrix. The fundamental matrix is defined in [17] as: „Fundamental matrix contains the same information as essential matrix[6] in addition to the information about the intrinsics of both cameras so that we can relate the two cameras in pixel coordinates.“ The fundamental matrix is calculated during stereo camera calibration (see section 5.1), in which predefined matched image points from a calibration pattern are used.

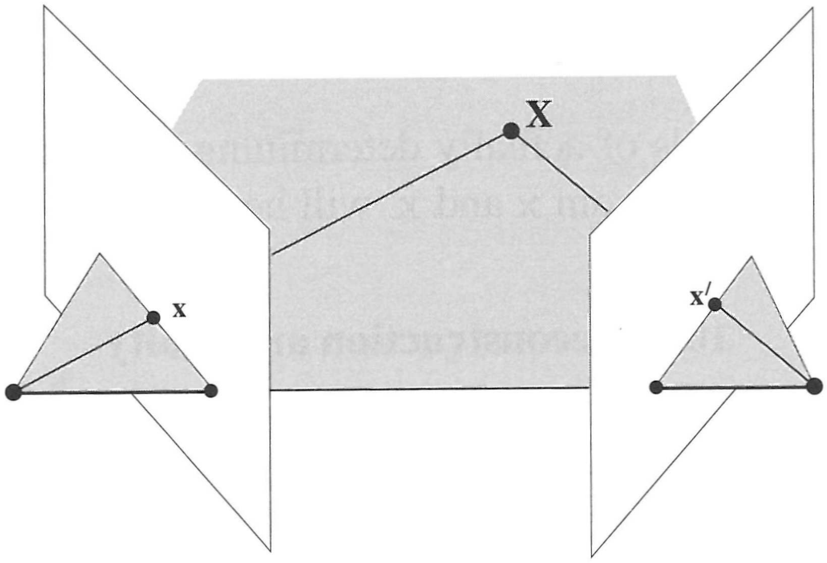


Figure 2.12: Graphical representation of epipolar constraint [41]

$$x'^T F x = 0 \tag{2.7}$$

Like in single camera calibration described in section 2.2.2, during the stereo camera calibration the aim is the estimation of camera matrices. This time as a first step the calibration pattern is captured simultaneously by both cameras, which results in one set of world points and two sets of image points (one set for each camera). The resulting

[6] The same source defines the essential matrix as: „Essential Matrix contains the information about translation and rotation, which describe the location of the second camera relative to the first in global coordinates.“

over-determined linear equation system is then solved again for minimizing the mean reprojection error. Since the camera matrices can just be calculated up to a certain scale, the mean reprojection error of the stereo camera calibration defines not only the accuracy of the camera calibration, but this time also the accuracy of the triangulation.

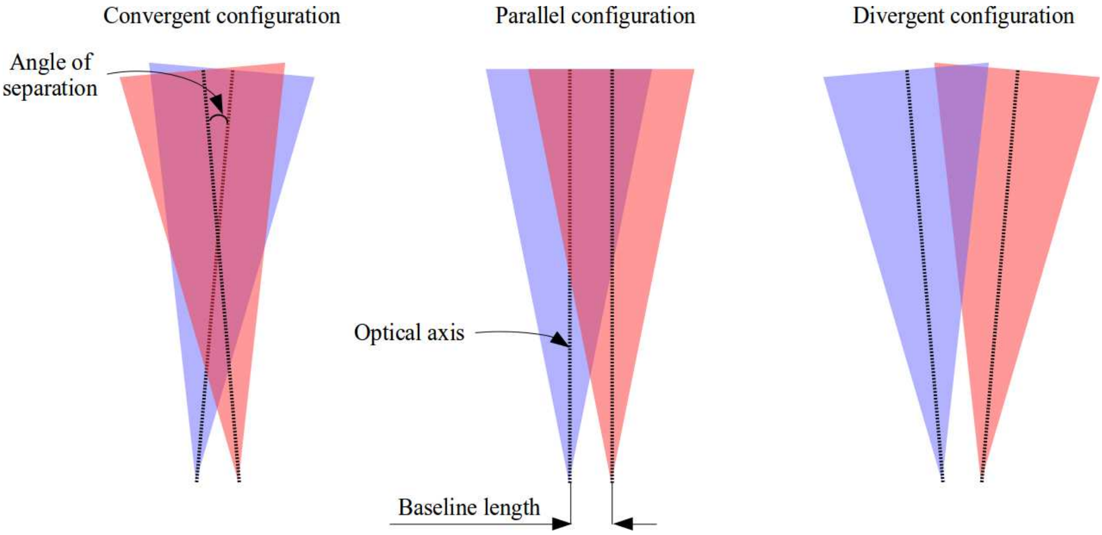


Figure 2.13: Stereo vision configuration types

Unlike in single camera calibration which estimates the intrinsic matrix of the camera, with two cameras the user can influence the accuracy of the calibration while positioning the cameras, namely preconditioning the extrinsic matrix of the second camera. The positioning of the cameras in a camera set depends on various factors like camera specifications, application of interest, light and environment conditions. Generally, an ideal stereo configuration should have the largest field of view with the smallest calibration error. Before discussing the factors which are essential for this ideal configuration, figure 2.13 shows the common stereo configuration types. If the axes of both cameras are parallel to each other, this configuration is called a parallel configuration. For the other cases where the optical axes are not parallel to each other, convergent and divergent configurations are defined. In a convergent configuration, the optical axes meet at a point in the field of view, which is not the case for divergent configurations. The angle between the optical axes is called the angle of separation, and the distance between camera centers is called the baseline length.

Kap Luk Chan and Andrew K. Forrest found out in their study [8] that the triangulation error due to calibration decreases approximately logarithmically with increasing angle of separation. They suggest an optimum angle of separation between 35° and 45°. In the same study, they express the relationship between baseline length and triangulation error as follows: „At a given focal length, a wider baseline reduces tri-

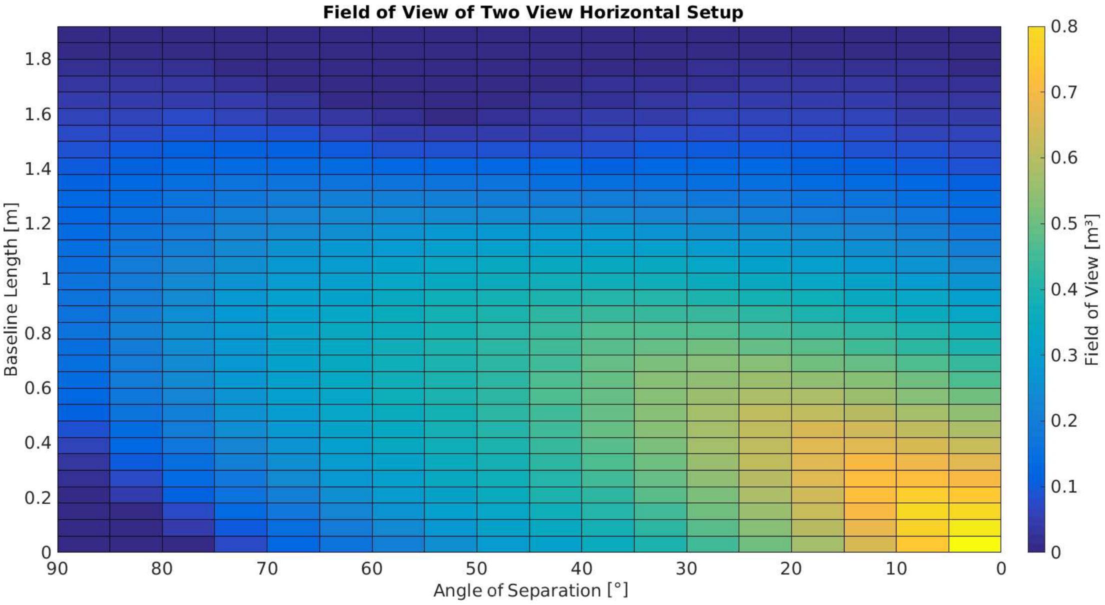


Figure 2.14: Field of view of horizontal two view setup

angulation error but increases both geometric and photometric distortions in images which in turn makes matching of these images more difficult.“ This results can also be seen in the experimental studies of Kytö, Nuutinen, and Oittinen [35]. On the other hand with decreasing baseline length the stereo field of view increases and reaches its maximum when both cameras converge on each other. The field of view for a stereo configuration depends on the baseline length, separation angle, and optical specifications of cameras. Figure 2.14 represents an example of the change in the field of view for two Logitech HD Pro C920 Webcams with the baseline length and separation angle. As expected the highest field of view occurs when both cameras are lying on the same place, which is practically not possible.

2.2.4 Multiple View Geometry and Bundle Adjustment

Like the inspiration from the birds while developing aircraft, humans were inspired by their own vision system for the computer vision development. However, after stereoscopy multiple view systems have extended its borders beyond the human visual system. A multiple view system is defined as a system in which more than two cameras are used. The triangulation process for multiple view systems is similar to stereo view, with known camera parameters and matched image points. However, the calibration process with more than two cameras is more complex and needs further steps.

One big difference while calibrating multiple cameras is that often it is not possible

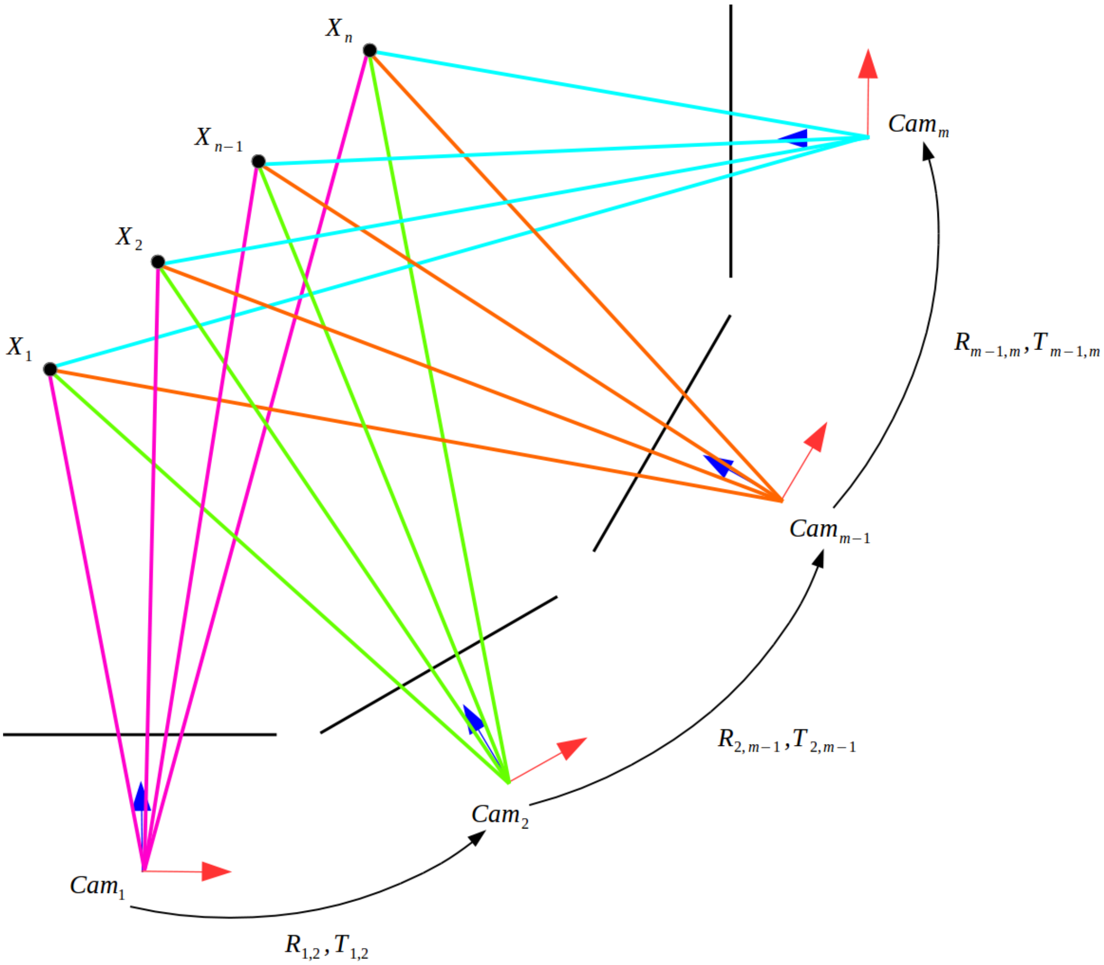


Figure 2.15: Schematic representation of bundle adjustment

to capture the calibration pattern simultaneously with the all cameras in the camera set. Practically it is possible to cover a maximum 180° with a 2D calibration pattern. Even if it is possible to capture that angle, working with large angles of separation increases the calibration error as defined in section 2.2.3. One approach to overcome these problems is decomposing a multiple camera calibration into several stereo camera calibration processes between subsequent camera tuples as shown in figure 2.15 for n calibration points and m views. After estimating intrinsic and extrinsic camera parameters, the extrinsic camera parameters (rotation matrices and translation vectors) can be recalculated regarding the first camera as shown in equation 2.8:

$$\begin{aligned} R_{1,m} &= R_{m-1,m}R_{1,m-1} \\ T_{1,m} &= T_{1,m-1} + T_{m-1,n}R_{1,m-1} \end{aligned} \tag{2.8}$$

In this equation, $R_{1,m}$ stands for the rotation matrix and $T_{1,m}$ stands for the translation vector from first to m^{th} camera. However, with the increasing number of cameras, the extrinsic calibration error of each camera leads to error propagation. In order to optimize the camera parameters in the multiple view setup, the bundle adjustment algorithm is used. This algorithm minimizes, as in the camera calibration process, the reprojection error for the given initial camera matrices. Its name bundle adjustment is related to the bundles between camera centers and world points as shown in figure 2.15. After the bundle adjustment calculation, these bundles should converge on each other, namely adjusted.

$$\min_{a_j,b_i} \sum_{i=1}^{n} \sum_{j=1}^{m} d(Q(a_j,b_i),x_{ij})^2 \tag{2.9}$$

Equation 2.9 shows the mathematical formulation of bundle adjustment defined by Lourakis and Argyros [29]. In this equation, a represents camera matrices and b world points. Q is the reprojected image points and d stand for the distance between this point and the captured image point x. This problem is solved by the Levenberg-Marquardt Algorithm (LM) in a bundle adjustment. Lourakis and Argyros describe the Levenberg-Marquardt Algorithm as follows [29]:

> „It has become a standard technique for nonlinear least-squares problems, widely adopted in various disciplines for dealing with data-fitting applications. LM can be thought of as a combination of steepest descent and the Gauss-Newton method. When the current solution is far from a local minimum, the algorithm behaves like a steepest descent method slow but guaranteed to converge. When the current solution is close to a local minimum, it becomes a Gauss-Newton method and exhibits fast convergence.“

3 3D Multi-View Particle Tracking Velocimetry

Most of the modern flow visualization techniques are based on the optical capture of the observed environment as described in the chapter 2.1. That means in order to carry out measurements using these techniques no disturbance between the investigated domain and the optical device is allowed. On the other hand, use of scientific cameras greatly elevates the hardware costs in these measurement systems. In a study from 2009 [6] 27% of all hardware costs (48849,73 €) is invested in the three cameras which are used in the measurement system. These two drawbacks restrict the usage of flow visualization techniques to small-scale problems, not only in field measurements but also in laboratory experiments. Measurements using these techniques in large volumes, which are very common in indoor air applications, therefore are nowadays not possible. The 3D-MVPTV, which will be described in this chapter, was developed to overcome these restrictions.

3.1 What is the 3D-MVPTV?

The concept of the 3D-MVPTV is inspired by the CFD. To accelerate the calculation, in CFD parallel processing is used, in which several computational cores are involved in the calculation. During parallel processing, a computational domain is decomposed and sent to several computational cores in which the calculations take place independently from each other. The final result is then obtained by recomposing the results from all computational cores. Using parallel processing it is possible to reduce the calculation time linearly proportional to the number of computational cores available.

Similarly, it is possible to apply the decomposition and recomposition processes of CFD to the flow visualization among others to the PTV. In this case, the measurement domain will be decomposed into several volumes, in which PTV measurements take place independently from each other. Finally, the PTV results from each volume are recomposed to receive the final result. These decomposition and recomposition processes take place through the distribution of multiple cameras in the domain. Utilizing multiple cameras eliminates problems due to optical obstacles and enables to conduct measurements in larger domains. However, using more cameras increases the complexity of the measurement system. This is not a fundamental problem and can be indeed overcome through a systematic approach. The major problem though is the rapid increase in hardware costs due to the number of cameras.

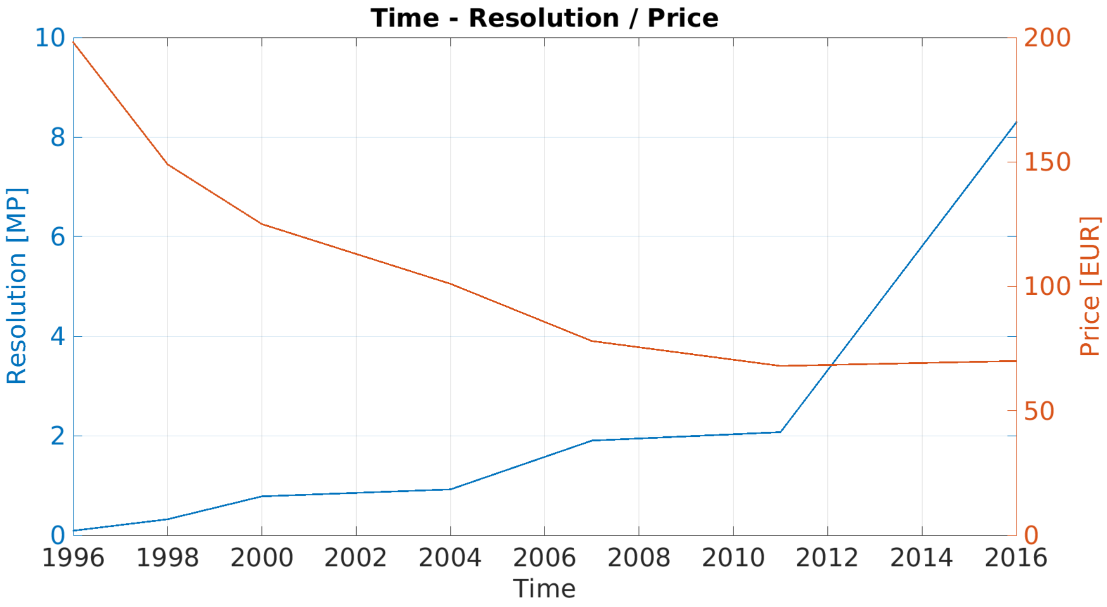


Figure 3.1: The development of the webcam market in the last two decades.

This problem can be solved using webcams instead of scientific cameras. If the development of the webcam market in the last two decades is considered (figure 3.1[7]), it can be seen that even with permanently increasing resolution the price of the webcams falls continuously. Today, most of the webcams have at least full high definition (FHD) resolution and cost less than 100 €. The sensors and optics in these webcams make them suitable for use in scientific studies. Another advantage of working with webcams is their conventional data transfer interfaces. Such conventional interfaces prevent the need for specific data acquisition equipment and make it possible to work with normal COTS (commercial off-the-shelf) workstations. 3D-MVPTV is a future-oriented concept: Extrapolating the past webcam market development mentioned above, in the future it will be possible to work either with higher resolution or for the same resolution with lower hardware costs and a less complicated system.

[7] Action cams, which also became dominant in the same resolution and price segment in the last years, are also included in this diagram.

3.2 The 3D-MVPTV Workflow

A 3D-MVPTV measurement consists of several sequential steps, starting with camera distribution and ending with post-processing of the results. All these steps are categorized and assembled in the 3D-MVPTV workflow to ensure the systematic execution of measurements. The following sections describe every step of this workflow. However, before detailing these steps, in section 3.2.1 the software environment which is used for realizing the 3D-MVPTV is introduced.

The 3D-MVPTV workflow starts with camera distribution in which cameras are positioned in experimental space. After this step, cameras should be calibrated, resulting in the mathematical description of the measurement system. Following camera calibration, frame synchronized video capture takes place in which tracer particles are captured in video files. When this step has succeeded, the object recognition step extracts these tracer particles from video files with their various properties. In the next step these recognized objects must be tracked, which happens in the object tracking step. In order to calculate 3D world points of the tracks, first image points of each track that are captured by several cameras must be matched in the track matching step. Following this step triangulation can be carried out, resulting in the 3D world points. Finally, these results can be post-processed according to user demands.

In this study besides the development of a 3D-MVPTV system and the 3D-MVPTV measurements, it is focused on two crucial steps of the 3D-MVPTV workflow: camera calibration and object tracking. The camera calibration step differs from other reviewed fluid visualization systems through the utilization of multiple cameras. Therefore, a specific calibration approach is developed. On the data processing side during the realization of object tracking, several improvement potentials compared to conventional tracking methods have been seen, which succeed in developing a new object tracking method.

3.2.1 Software Environment for the 3D-MVPTV Development

In order to develop such an application as 3D-MVPTV and keep it updated with new technologies, the software environment plays a crucial role. There are several aspects which should be taken into account when deciding on an appropriate software environment. The most fundamental aspect is that the software must be capable of realizing needed steps in practice. For 3D-MVPTV that is a comprehensive task since the idea includes such different steps such as image and video capture, camera calibration, image processing, and various mathematical calculations. Other vital properties expected from the software are service and support. The software should be developed continuously following the new technologies and according to users' demands. Also in case of a problem, the software developer should be able to deliver professional and timely support to overcome the users' problems. Increasing the interdisciplinary character of technical problems, the software should also have flexible hardware and software interfaces which can communicate with diverse external sources if needed. Especially for non-academic use, license costs also play a crucial role. Finally, users' previous experience with software and development exhibit a critical factor when choosing an appropriate development software.

The first steps of the presented 3D-MVPTV were started using the software environment OpenCV in 2010. The first task was a stereo camera calibration, which was carried out successfully. However, due to the lacking service and support for this open-source tool, all other tasks were processed with MATLAB® and its toolboxes. With the launch of the Computer Vision System Toolbox™ in 2014, the camera calibration step was also moved to MATLAB®, which combined all the steps of 3D-MVPTV in one software environment. Today the 3D-MVPTV runs with MATLAB® and its following toolboxes:

- „Image Acquisition Toolbox™ provides functions and blocks that enable to connect industrial and scientific cameras to MATLAB® and Simulink®." [31]
- „Image Processing Toolbox™ provides a comprehensive set of reference-standard algorithms and workflow apps for image processing, analysis, visualization, and algorithm development." [31]
- „Computer Vision System Toolbox™ provides algorithms, functions, and apps for designing and simulating computer vision and video processing systems." [31]
- „Parallel Computing Toolbox™ lets to solve computationally and data-intensive problems using multicore processors, GPUs and computer clusters." [31]

The 3D-MVPTV measurement system works with MATLAB® and its toolboxes without any problems. This software environment is developed continuously (two releases per year nowadays) parallel to new technologies, which is proved, e.g., with the launch of a bundle adjustment algorithm in 2016 that is already integrated into the 3D-MVPTV workflow. Besides the possibility to contact the developer team, this software environment made it possible to realize the 3D-MVPTV concept efficiently.

On the other hand, it is also possible to integrate this workflow into OpenCV, useful especially if a commercial application without license costs is desired. The MATLAB Compiler^{TM} is another possibility for commercial applications, in which developed software can be utilized as a standalone application.

3.2.2 Camera Distribution

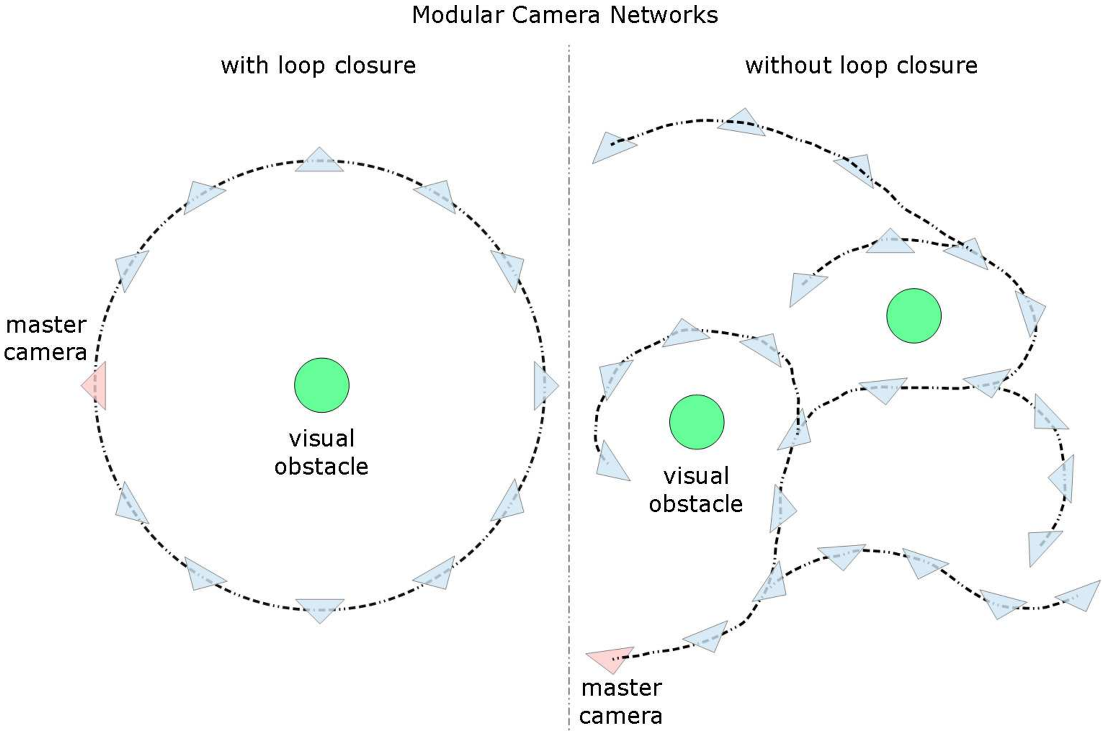


Figure 3.2: Modular camera networks

The 3D-MVPTV concept is based in principle on using multiple cameras. In this context, unlike conventional methods, rather than camera pairs modular camera networks are at the point of interest. A modular camera network is a calibrated set of cameras which are distributed in an experimental domain to capture the desired volume of interest. Its modular structure makes it possible to relocate the cameras in the network, recalibrate them, and run new measurements on a different volume of interests. Figure 3.2 shows two examples of such modular camera networks. The master camera in these representations is the camera whose coordinate system is also the global coordinate system for the measurements. That means the triangulated 3D world point coordinates are oriented according to this global coordinate system, even if the master camera is not participating in the triangulation process. Depending on the geometrical restrictions of the experimental domain, a modular camera network can be defined with or without loop closure. In a modular camera network with loop closure, the last camera in the network is also associated with the master camera during the camera calibration process, which theoretically delivers more robust results during camera calibration.

On the one hand, the number of cameras used in such a modular camera network depends on the size of the volume to be captured. Finding the optimum number and distribution of cameras is a mathematical optimization problem, which is not covered in this study. On the other hand, the number of cameras is responsible for the triangulation accuracy, the complexity and the total cost of the system as well. This relation between camera distribution and triangulation accuracy is going to be investigated in chapter 5.3. Generally, the most important boundary conditions for an appropriate choice of camera distribution can be listed as:

- Every point in the measurement domain must be captured at least with two cameras to calculate 3D world points with triangulation.
- The tracer particles must be captured at least with one pixel on the camera sensor.
- Minimum possible triangulation error should be achieved.
- Minimum possible number of cameras should be used.

3.2.3 Camera Calibration

This step of the 3D-MVPTV workflow is described separately in chapter 5 since its approach differs from conventional methods and—together with object tracking—it represents one of the two main focus points of this study.

3.2.4 Frame Synchronized Video Capture

Table 3.1 shows the specifications of the webcam utilized in this study Logitech HD Pro Webcam C920[8]. It can capture uncompressed 24 bits FHD-videos with a frame rate of 5 fps. At the end of 2016, this webcam could be acquired for circa 65 €, which makes it suitable to use within 3D-MVPTV.

In order to calculate the 3D world points of tracer particles, frame capture must be carried out by each camera in the same instant of time. Otherwise different frame cap-

[8] Logitech support has delivered some of these technical specifications.

Focal length	3.67 mm
Field of view	78°
Horizontal x vertical resolution	1920 x 1080 pixels
Compressed frame rate at full resolution	30 fps
Uncompressed frame rate at full resolution	5 fps

Table 3.1: Logitech HD Pro Webcam C920 technical specifications [28]

ture times between cameras lead to different image point positions of a tracer particle for every camera, which results in an erroneous calculation of the 3D world points. One important factor to minimize the deviation between frame capture times is an undisturbed operation of data flow in the image acquisition system from capture device to storage. That means bandwidths of the devices and interfaces through the image acquisition system must be capable of processing the captured data flow.

$$(1080 \times 1920)\frac{pixel}{frame} \times 3 \times 8\frac{bit}{pixel} \times \frac{1}{8}\frac{byte}{bit} \times \frac{1}{2^{20}}\frac{MB}{byte} \times 5\frac{frame}{second} = 29,663MBps \tag{3.1}$$

The amount of data generated by one webcam per second can be calculated as shown in equation 3.1. Each pixel in every layer of the RGB color space is defined using 8 bits. This yields a data generation rate of 29,663 MBps per camera and 474,608 MBps for the whole system with 16 webcams used in this study. Components of the image acquisition system, which must be capable of processing this amount of data, can be divided into two groups: the hardware and the software. Figure 3.3 shows the four main components of a typical hardware setup for the image acquisition system used in 3D-MVPTV: webcams, USB host controllers, a workstation, and a storage device. Webcams, the first elements of the image acquisition chain, act as data source. From webcams the captured data is transferred through USB host controller to the workstation's mainboard. A host controller can be used for one or more webcams depending on the webcam generated data and the bandwidth of the host controller. In this study, four cameras are directly connected to the two host controllers on the mainboard. The other twelve webcams are connected through three PCI express cards each with four separate USB host controller per lane. Every USB host controller has a bandwidth of 5 GBps, which ensures a failure-free data transfer.

Captured data entered to the mainboard must be immediately stored to keep the data flow undisturbed. At the time of developing the image acquisition system using an existing workstation, it was not possible to store captured data directly to the storage

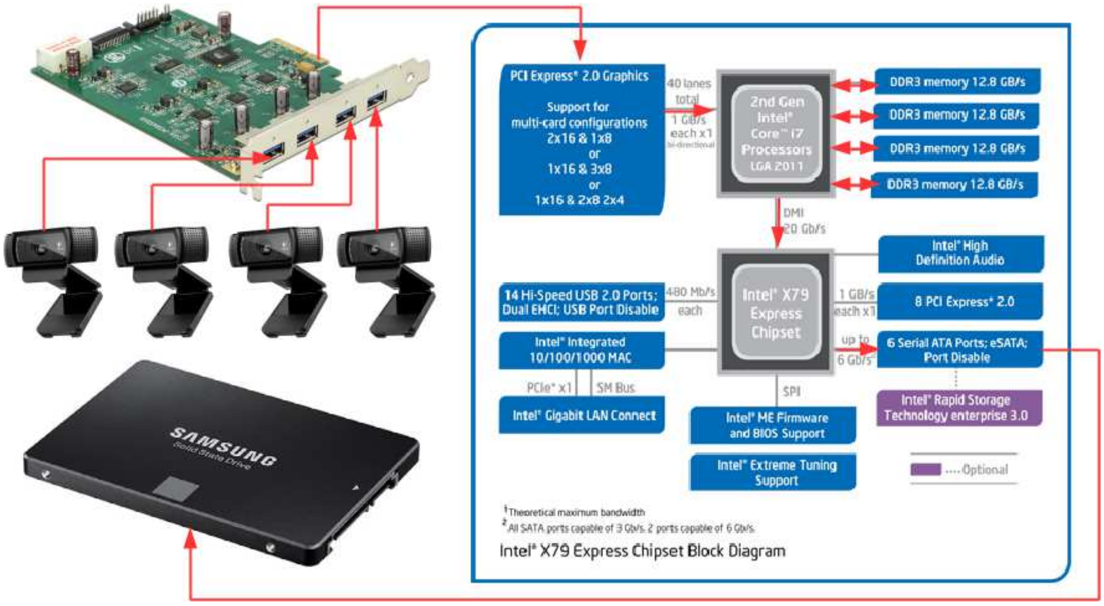

SAMSUNG
Solid State Drive
PCI Express* 2.0 Graphics
Support for multi-card configurations 2x16 & 1x8 or 1x16 & 3x8 or 1x16 & 2x8 2x4
40 lanes total
1 GB/s each x1 bi-directional
2nd Gen Intel® Core™ i7 Processors LGA 2011
DDR3 memory 12.8 GB/s
DDR3 memory 12.8 GB/s
DDR3 memory 12.8 GB/s
DDR3 memory 12.8 GB/s
DMI 20 Gb/s
Intel® High Definition Audio
14 Hi-Speed USB 2.0 Ports; Dual EHCI; USB Port Disable
480 Mb/s each
Intel® X79 Express Chipset
1 GB/s each x1
8 PCI Express* 2.0
Intel® Integrated 10/100/1000 MAC
up to 6 Gb/s
6 Serial ATA Ports; eSATA; Port Disable
PCIe* x1
SM Bus
SPI
Intel® Gigabit LAN Connect
Intel® ME Firmware and BIOS Support
Intel® Rapid Storage Technology enterprise 3.0
Intel® Extreme Tuning Support
Optional
Theoretical maximum bandwidth
All SATA ports capable of 3 Gb/s. 2 ports capable of 6 Gb/s.
Intel® X79 Express Chipset Block Diagram

Figure 3.3: Hardware components of the image acquisition system for four webcams [15] [24] [28] [42]

device due to the limitations of the drive bandwidth, driver and software-based restrictions as well. Therefore, captured video data is temporarily stored in the memory to avoid errors during video capture and moved to a storage device at the end of the video capture process. This workaround limits the total video capture time depending on the total memory size of the workstation, in this study 32 GB. Equation 3.2 shows the theoretical maximum capture duration. Considering other running system services and programs on the workstation, which also allocate memory, maximum video capture time in this study is 60 s.

$$\frac{32GB \times \frac{1024}{1}\frac{MB}{GB}}{474,608MBps} = 69,042s \tag{3.2}$$

Before the final data storage on a storage device, captured $24-bit$ RGB video data is converted to $8-bit$ grayscale data and saved in a three-dimensional array with a size of vertical resolution × horizontal resolution × number of frames. This conversion reduces the data size to its one-third. Since color information of tracer particles is not necessary for the following steps of the workflow, this step is vital to save disk space. Direct video capture in $8-bit$ grayscale color space is not possible, as no grayscale driver for the Logitech HD Pro Webcam C920 webcam was available during the development of this system.

On the software side running on this hardware setup, there are two essential components: webcam driver and image acquisition software. Webcam driver defines the interface between webcam (hardware) and operating system (software). Two possible drivers for the Logitech HD Pro Webcam C920 are available: a vendor driver and a generic operating system driver. Since uncompressed video capture is only possible using a generic operating system driver, this driver is used. Working with the vendor driver could not ensure frame synchronized video capture because compression and decompression steps are used, which also demand computational resources from the workstation.

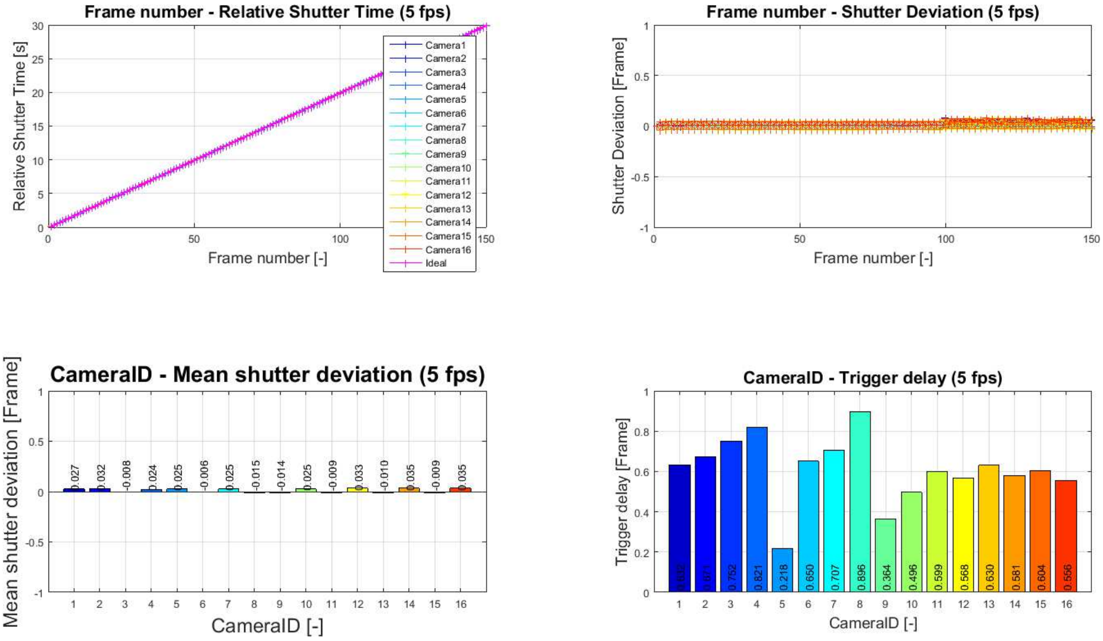


Figure 3.4: Frame synchronized video capture statistics with 16 webcams

Frame synchronized video capture is carried out using MATLAB® and its Image Acquisition Toolbox™. The Image Acquisition Toolbox™ can communicate with webcams using the webcam driver described above, control their parameters like frame rate, exposure, and focus level; start and stop video capture, and store captured data. Furthermore, it can store timestamps of the frames, which records the absolute system time when the frame is captured. This analysis feature is very crucial and makes it possible to determine if the video capture is carried out frame synchronized or not. At the end of each video capture, timestamps from each webcam are analyzed and plotted using a standardized layout as shown in figure 3.4. In order to interpret these statistics independently of the frame rate, frame-based normalization is applied. In these graphs, if every camera captures each frame at the same time (see graph top left), it can be determined how big the normalized shutter deviation for each frame is (see graph top right). These graphs also represent the average normalized shutter deviation through the video capture period (see graph bottom left) and the normalized trigger delay for each camera (see graph bottom right).

For the image acquisition step, a function was programmed which can automate this process with variable inputs. The input parameters for this function are the camera IDs to be used, the frame rate, and the desired duration of the synchronized video capture. This function first initializes all the cameras to be used and ensures that all the camera parameters like zoom level, focus level, and frame rate are set to the same value. After that, synchronized video capture takes place, and resulting video data is stored in arrays. Finally, the standardized synchronized video capture statistics, as shown in figure 3.4, are plotted and stored.

3.2.5 Object Recognition

Once video data is stored on a storage device, it can be processed to achieve 3D-MVPTV results. Object recognition is the step, where pixel-based information of tracer particles is converted to mathematical data. Concretely, tracer particles are defined as pixels before the object recognition step, and their coordinates define them on the image plane of the webcam with further geometrical information after this step. In this step, MATLAB® and its Image Processing Toolbox™ are used.

Before the object recognition step $8-bit$ grayscale image data, which is generated in the previous step, must be converted to $1-bit$ monochrome image data. This conversion is needed since the object recognition is carried out on $1-bit$ monochrome image data. The difference between $8-bit$ grayscale and $1-bit$ monochrome color depth is about how detailed every pixel can represent the real world. $8-bit$ grayscale color depth uses for it $2^8 = 256$ colors and $1-bit$ monochrome color depth uses $2^1 = 2$ colors, namely black and white.

One critical question regarding this color depth conversion is: Which color intensity for this conversion should be used? In other words, where is the border which defines the separation point between black and white pixels in $1-bit$ monochrome image data? In order to define this color intensity, global thresholding using Otsu' s method is implemented. Figure 3.5 shows the result of this conversion. An original $8-bit$ grayscale image with tracer particles is shown on the top left and the same image converted to a $1-bit$ monochrome image with global thresholding using Otsu' s method is shown on the top right. In the images on the bottom, conversion results with manual thresholding using extreme values are represented. As can be seen in these two

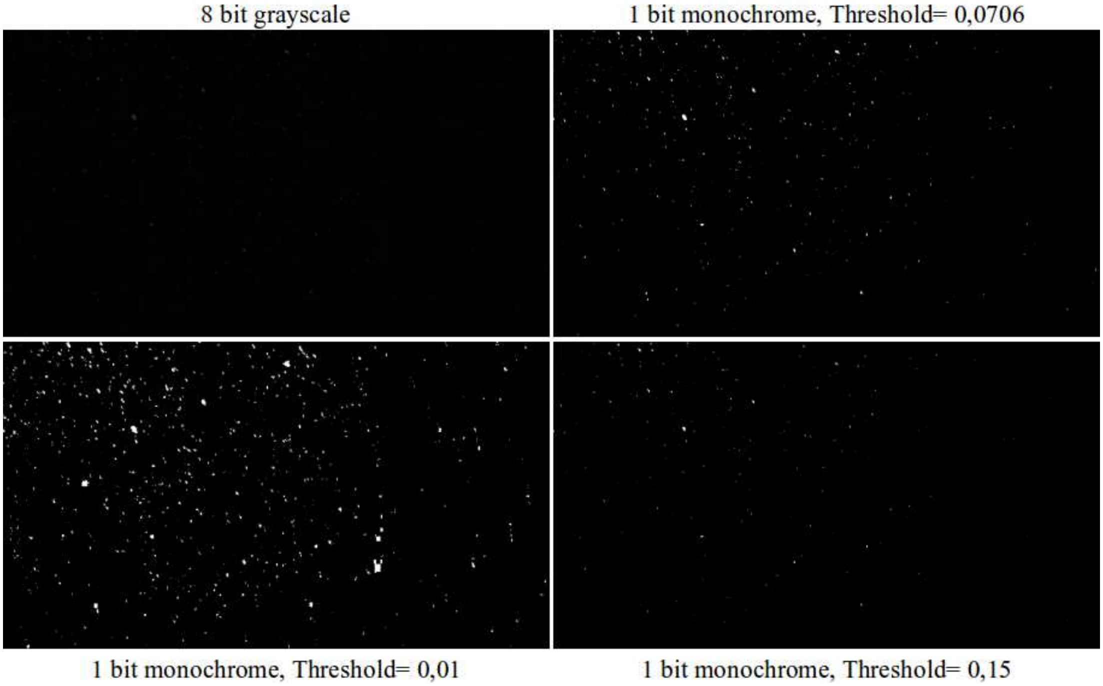

8 bit grayscale
1 bit monochrome, Threshold= 0,0706
1 bit monochrome, Threshold= 0,01
1 bit monochrome, Threshold= 0,15

Figure 3.5: $8-bit$ to $1-bit$ conversion with different threshold levels

images, using a small value (0,01) can yield extra noise and using a high value (0,15) can result in loss of captured tracer particles.

After the image conversion into $1-bit$ monochrome space, image segmentation takes place. „Segmentation subdivides an image into its constituent regions or objects.“ [40]. As emphasized in this definition of Eddins, Gonzalez, and Woods, image segmentation extracts tracer particles from $1-bit$ monochrome images. This extraction is based on the boundaries of the pixels, which define tracer particles. Here, an 8-connected neighborhood [31] is implemented, which does not just consider edges of a pixel as in 4-connected neighborhood, but also its corners. Once the image segmentation is carried out, and tracer particles in an image are defined, the next and last step of image recognition is to collect some characteristic properties of these objects shown in figure 3.6.

These object properties are required in the next step of the workflow, object tracking, to associate the same tracer particles in the following frames. The question „Which object properties should be stored for the segmented objects?“ is an application specific question. Table 3.2 represents the answer to this question regarding tracer particles in this 3D-MVPTV. These properties define the geometry of the tracer particles and their position (centroid) on the image plane. In order to capture the deviation of tracer particles from their sphere geometry due to illumination and motion blur, properties

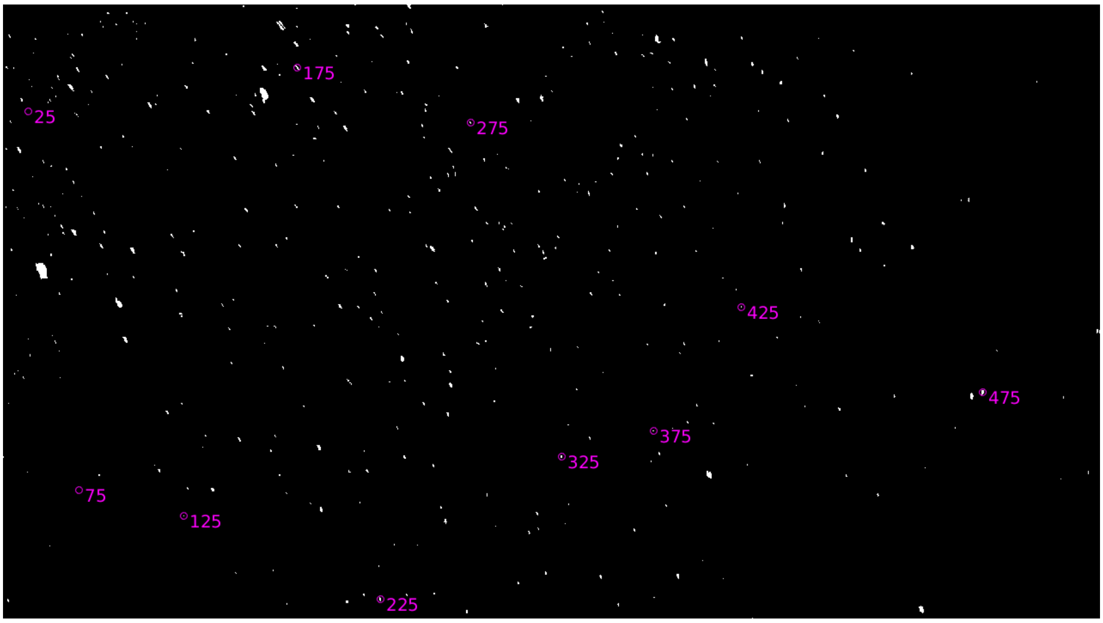


Figure 3.6: Tracer particles represented with pixels in $1-bit$ monochrome space

like eccentricity, major axis length, minor axis length, and orientation are stored. The bounding box property is used in the post-processing, so that tracked particles can be plotted later on. The object area is an important property which depends on the distance of the tracer particle to the camera since the tracer particle size deviation is relatively small (Christenfeldt [9]). Taking this property into account enables grouping tracer particles during object tracking.

Property Name	**Description**
Area	A scalar that specifies the actual number of pixels in the region.
Bounding Box	The smallest rectangle containing the region.
Centroid	A 1-by-Q vector that specifies the center of mass of the region.
Eccentricity	A scalar that specifies the eccentricity of the ellipse that has the same second-moments as the region.
Major Axis Length	A scalar that specifies the length (in pixels) of the major axis of the ellipse that has the same normalized second central moments as the region.
Minor Axis Length	A scalar that specifies the length (in pixels) of the minor axis of the ellipse that has the same normalized second central moments as the region.
Orientation	A scalar that specifies the angle between the x-axis and the major axis of the ellipse that has the same second-moments as the region.

Table 3.2: Object properties stored during image recognition [31]

For the presented 3D-MVPTV, a function is programmed, which can handle the object recognition process with variable inputs. This function reads $1-bit$ monochrome frames, which are stored in an array and recognizes the object properties shown in table 3.2. The recognized objects for each frame, are then stored in an array as shown in table 3.3[9]. The first column of this array consists the unique object IDs for each recognized object in the corresponding frame. The third and fourth columns are the image coordinates of the object centroids, which are defined according to the image coordinate system (see figure 2.10).

Object ID	Area	X-Coor.	Y-Coor.	Eccentricity	Major Axis Length	Minor Axis Length	Orientation
25	2	47	186.5	0.86603	2.3094	1.1547	90
75	1	136	848	0	1.1547	1.1547	0
125	5	319.6	893.8	0.76509	3.3066	2.1292	-71.565
175	23	517.17	110.65	0.96211	11.044	3.0111	-52.484
225	19	662	1039	0.90034	7.5807	3.2991	-83.736
275	12	819.83	206.92	0.93764	6.9482	2.4153	-47.93
325	9	978.56	790.22	0.90794	5.3992	2.2628	85.269
375	5	1139.2	745.4	0.76509	3.3066	2.1292	-18.435
425	8	1292.5	528.5	0.86603	4.6188	2.3094	90
475	32	1713.7	678.16	0.74473	8.4421	5.634	77.589

Table 3.3: Exemplary object properties for the plotted objects in figure 3.6

Since the object recognition step can be executed separately in each frame, with other words the frame sequence does not play a role in object recognition, this step can be run in parallel. Using the Parallel Computing Toolbox™, frames are distributed between computational cores[10], on which parallel object recognition processes can run independently of each other. The parallelization of the object recognition step is linearly scalable, that means the computational time decreases with the increasing number of computational cores available.

3.2.6 Object Tracking

This step of the 3D-MVPTV workflow is described in chapter 6 in detail since its approach differs from conventional methods and—together with the camera calibration—it represents one of the two main focus points of this study.

[9] In this exemplary array, the bounding box object properties are not shown due to clarity.

[10] In this study, 16 cores in the computer cluster of the Hermann-Rietschel-Institute are used.

3.2.7 Track Matching

After object tracking is carried out successfully, the tracks captured by multiple cameras must be associated with each other. That is crucial in order to calculate 3D coordinates of the objects in each track using multi-view triangulation in the next step. The track matching process requires two inputs: Stereo camera parameters including the fundamental matrices which are gained in the camera calibration step and tracking matrices as results of the object tracking step. The outputs of track matching are *matched tracks* matrices for every track in the world space. Each of these matched tracks matrices consists of track IDs in image space and the corresponding camera IDs they belong to. The size of such a matrix is $N_{T,i} \times N_C$, in which $N_{T,i}$ represents the number of tracks in image space and N_C is the number of cameras.

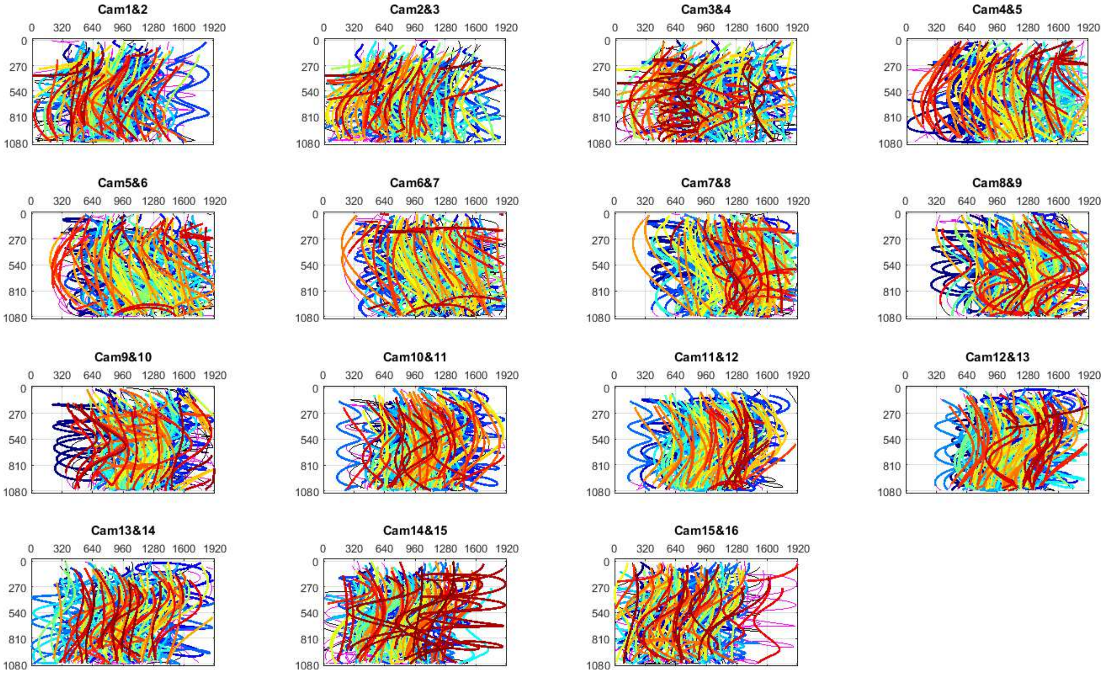


Figure 3.7: Matched tracks in image space of multiple cameras

The track matching process starts with undistorting image points in the tracks since the matching calculation described below requires undistorted image points. The track matching is carried out between two consecutive camera pairs, which means for N_C cameras N_C steps for a setup with loop closure and $N_C - 1$ steps for a setup without loop closure must be run. Here, tracks captured by a camera are cross-correlated with the tracks of the consecutive camera. The cost function for this cross-correlation is the epipolar constraint, which defines if two 2D image points on two different cameras can represent a 3D world point (see section 2.2.3). Since the camera calibration and object recognition steps have their precision errors, the right-hand side of the equation

2.7 approaches 0. In this context, the track matching algorithm uses the mean value of the right-hand side calculated using the image points in the tracks. This value is then normalized before the cross-correlation is carried out. For enhancing cross-correlation, a filter is implemented, which uses the track pairs with at least two common frames. That is necessary since after the triangulation step at least two frames are required to calculate the velocity components. Figure 3.7 shows the results of such a track matching, in which tracks with the same colors represent matched tracks between subsequent cameras.

3.2.8 Multi-View Triangulation

The track matching in image space delivers the information which tracks appear on which cameras. The next step is the multi-view triangulation of image points in these tracks in order to calculate their world points. For the multi-view triangulation, three inputs are necessary: The image points of the objects in the tracks, the matrix with matched tracks calculated in the previous step and the camera matrices which includes intrinsic and extrinsic camera parameters. The critical question for multi-view triangulation is: Which cameras must participate in the triangulation process? To answer this question, as a first step the total number of all possible camera set combinations can be calculated using equation 3.3, whereby N_C stands for the number of cameras.

$$\sum_{i=2}^{N_C} \binom{N_C}{i} = 2^{N_C} - \binom{N_C}{1} - \binom{N_C}{0} = 2^{N_C} - N_C - 1 \tag{3.3}$$

Since in this work a maximum of 16 cameras is used, 65619 different camera set combinations for triangulation are possible. While choosing the most suitable camera set between these combinations, the most appropriate cost function would be the triangulation error. In other words, the camera set combination must be chosen which delivers the minimum deviation between real world points and triangulated world points. In the early phases of the 3D-MVPTV development, before the development of the 3D-MVPTV-SIL-Environment described in section 4.2, an experiment was carried out to figure out the relationship between the camera sets used for triangulation and the resulting triangulation errors.

This experiment was carried out with a single object, a table tennis ball with a diameter of $40mm$. It was positioned in an orbit with a radius of circa $100mm$. The motion of the table tennis ball in this orbit is ensured using an electric motor, on which the ball is installed using a metal rod. This construction enables to control the angular velocity of the ball by controlling the motor speed using a dimmer unit. This experiment's aim is to compare the known orbit radius of the ball with the triangulated results using the different camera sets. In order to achieve this aim, the ball is moved in the orbit with a constant angular velocity for $30s$ so it can have a couple of orbits. This motion is captured by all 16 webcams simultaneously so that the 3D world points of the orbit can be calculated by recognizing the midpoint of the ball as shown in figure 3.8.

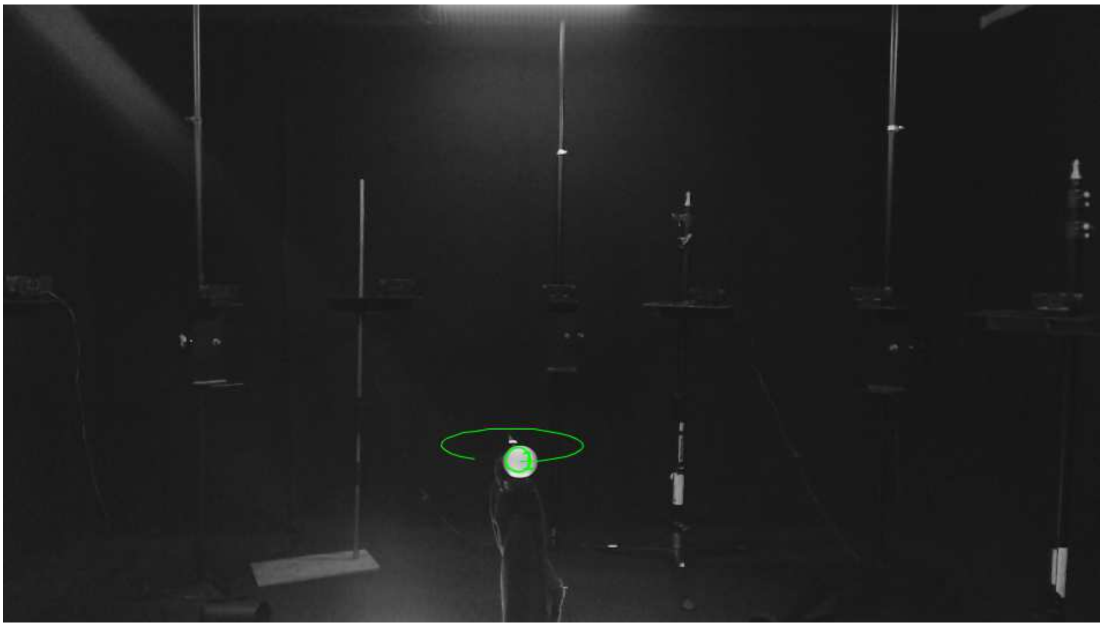


Figure 3.8: Triangulated world points of the table tennis balls orbit

Figure 3.9 represents the results of this experiment. In this graph, the bars show the number of combinations for the camera sets with a different number of cameras. For the triangulation with the same number of cameras, the minimum, maximum, mean and standard deviation values for the difference between the maximum and minimum orbit radii are plotted using the horizontal axis on the left. The maximum, mean and standard deviation values decrease with ascending number of cameras. When all 16 cameras are used for the triangulation, the difference between the maximum and minimum orbit radius is $5mm$. However, if we consider the minimum values of the combinations, a camera set with nine cameras delivers less than $1mm$, which is much closer to the ideal value of the difference between the maximum and minimum orbit radii, $0mm$.

Through this experiment with the table tennis ball, a general rule to find the camera set resulting in minimum triangulation error could not be found. Therefore, during the

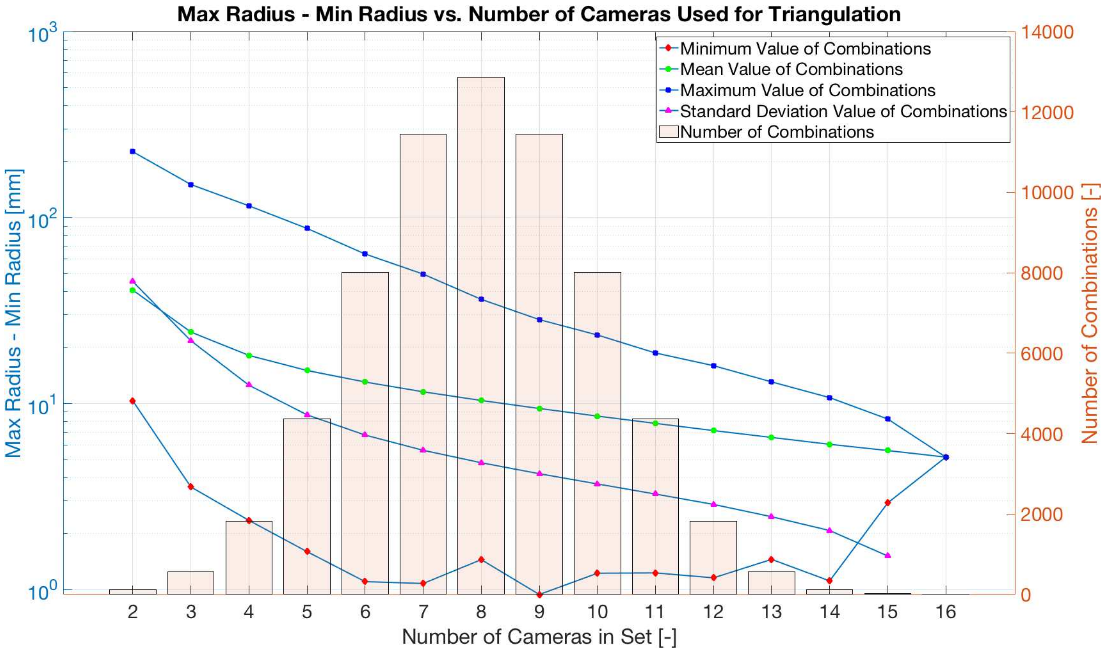


Figure 3.9: Triangulation results of the experiment with a table tennis ball

development of the 3D-MVPTV two triangulation methods were developed which try to find and use the camera set delivering the minimum triangulation error possible. The algorithm of the first method can decide the camera set to use automatically by considering the mean reprojection error of the cameras. Figure 3.10 shows the flowchart of this algorithm. This algorithm first triangulates a track using all available cameras and calculates the mean reprojection error for every camera. If the number of cameras participated in the multi-view triangulation are more than two, the camera with the highest mean reprojection error is eliminated, and the triangulation is carried out with

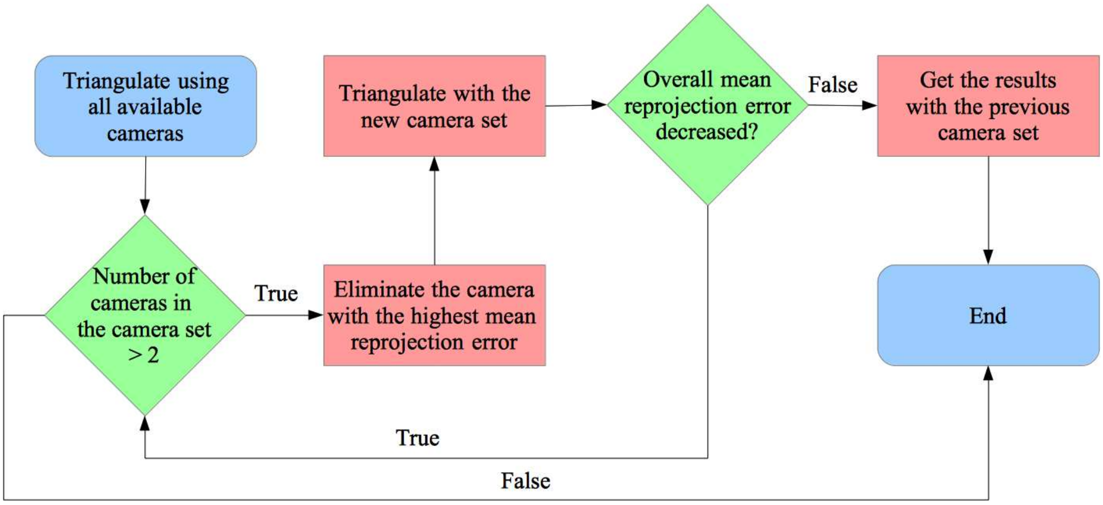

Triangulate using all available cameras
Number of cameras in the camera set > 2
True
Eliminate the camera with the highest mean reprojection error
Triangulate with the new camera set
Overall mean reprojection error decreased?
False
Get the results with the previous camera set
End
True
False

Figure 3.10: Flowchart of the multi-view triangulation algorithm with the elimination

the new camera set. If the resulting overall mean reprojection error[11] is smaller than the overall mean reprojection error without camera elimination, further camera elimination steps are carried out. If the overall mean reprojection error does not decrease after the camera elimination, the triangulation results before the camera elimination are used.

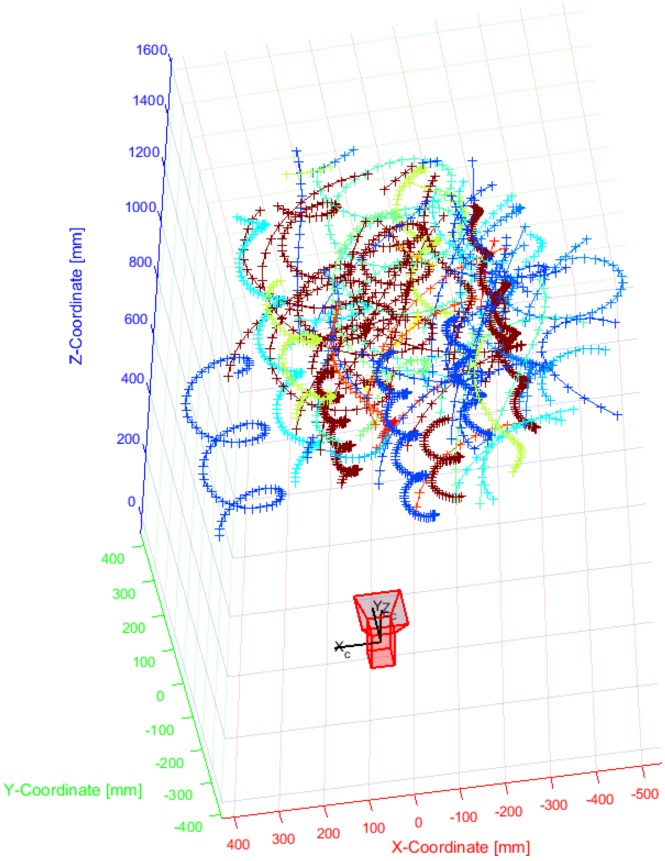


Figure 3.11: World points of tracked tracer particles after multi-view triangulation

The second method for the multi-view triangulation utilizes a so-called triangulation map, which is precalculated under the 3D-MVPTV-SIL-Environment. Therefore, the triangulation algorithm with precalculation needs an extra step compared to the triangulation algorithm with elimination. For the calculation of the triangulation map, numerous virtual world points are first distributed in the measurement domain. These world points are then projected on the image sensors. Finally, triangulations for all camera set possibilities are carried out with these image points, and resulting triangulation errors for every world point are calculated. The triangulation map includes the mean triangulation for every camera set combination grouped according to the number of cameras in the camera set and sorted with ascending mean triangulation error. During the triangulation with precalculation, the camera set delivering the lowest mean triangulation error is chosen from the triangulation map, and multi-view triangulation is carried out using this camera set. A detailed comparison of these two triangulation methods and the choice of camera set for multi-view triangulation is presented in section 5.3.

The multi-view triangulation is carried out using data in which at least two cameras

[11] Here *overall* is used to represent the mean reprojection error of all cameras.

are employed. The resulting world points are defined according to the coordinate system of the first camera as shown in figure 3.11, which is also the global coordinate system of the 3D-MVPTV system. The results are exported in the *track world points* cell array [31] with a dimension $N_{T,w} \times 1$. Each element of this cell array is an $N_F \times 4$ matrix, whereby N_F represents the number of frames for the measurement. The first three columns of this matrix have the world coordinates of the tracked, and the fourth column includes the mean reprojection error for each point for further error analysis.

3.2.9 Post-processing

Equation 3.4 calculates the three-dimensional velocity components using calculated world points of objects in tracks. In this equation, $\vec{v}$ stands for the velocity vector, X, Y, and Z for the world point coordinates. Δt is the time between frames, which can be calculated using frame rate, e.g., for $5fps$, $\Delta t = 1/5 = 0.2s$. The index f represents the actual frame and $f+1$ the following frame.

$$\vec{v}_f = \begin{bmatrix} \frac{X_{f+1}-X_f}{\Delta t} & \frac{Y_{f+1}-Y_f}{\Delta t} & \frac{Z_{f+1}-Z_f}{\Delta t} \end{bmatrix}^T \tag{3.4}$$

These calculated variables are in the next step written into a generic CSV (Comma Separated Values) file, so that it can be processed with various post-processing tools. This CSV file contains the following columns: *X-Coordinate [m], Y-Coordinate [m], Z-Coordinate [m], X-Velocity [m/s], Y-Velocity [m/s], Z-Velocity [m/s], Velocity Magnitude [m/s], Time [s], Track ID [-].* The column Time [s] makes it possible to visualize time-dependent flow field, and Track ID [-] can be used for extracting tracks for plotting the path lines. The values in this result file can also be

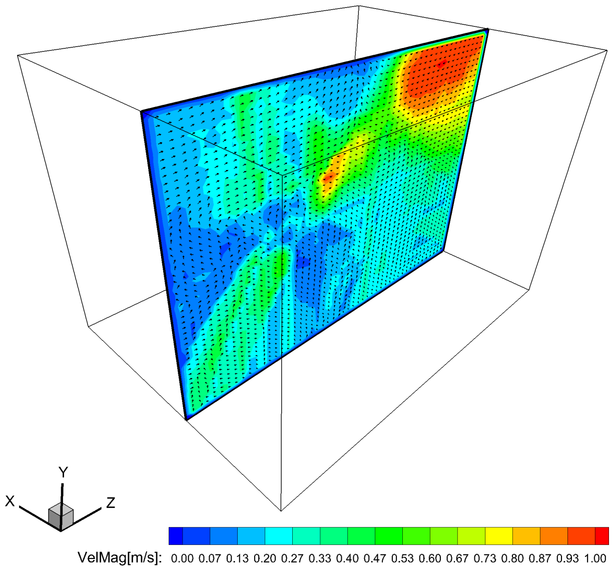


Figure 3.12: Spatially interpolated results on a section plane

interpolated spatially and temporally as well using a post-processor if required. Figure 3.12 shows the spatial interpolation of the 3D-MVPTV results on a section plane.

4 Real and Virtual Development Environments for 3D-MVPTV

This chapter describes the environments utilized during the 3D-MVPTV development. These environments can be categorized into two groups: real and virtual development environments. The real development environment is an experimental test stand, in which the hardware is installed, and 3D-MVPTV measurements can be carried out. On the other hand, the virtual development environment is a software-in-the-loop environment (SIL) whereby the development takes place in a software environment without using hardware.

4.1 3D-MVPTV Test Stand

In order to carry out the experiments during the 3D-MVPTV development, a former test stand at the Hermann-Rietschel-Institut was modified to become the 3D-MVPTV test stand. As shown in figure 4.1, this test stand has dimensions of $2.5m \times 2.5m \times 1.5m$. The inlet near to the floor is equipped with three axial fans (Sharkoon System Fan P $120 \times 120 \times 25mm$) in horizontal order, which can supply air for the test stand with a velocity up to circa $0.5m/s$. These fans are connected to the power supply through a dimmer unit so that the air velocity flowing into the test stand can be adjusted. The turbulent vortex structures generated by the axial fans are reduced using a flow laminator with honeycomb structures. The outlet opening is placed near the ceiling. In the middle of the room, there is a cylinder with a diameter of $27cm$ and a height of $56cm$. It serves as an optical obstacle which is mainly used to demonstrate the multi-view strength of the 3D-MVPTV. This cylinder can be used at the same time as a heat source, which makes it possible to run experiments with buoyancy-driven flow structures.

The existing body construction of the webcam can rotate around the X-axis. This body construction is installed on a horizontal mount plate, which is then attached to a clamp mount with a free rotation around the Y and Z axes. Finally, the clamp mount including mount plate with the webcam on it is installed on a vertical bar which is stretched between floor and ceiling. The sliding clamp mount on this bar makes it possible to quickly adjust the height of the camera from the floor (circa 65*cm*). The webcams can easily be distributed in the test stand as desired using this construction. A mini bubble level is used to ensure the vertical orientation of the bar and the horizontal orientation of the webcam. All 16 Logitech C920 webcams are then positioned on a circular ring with a diameter of circa 2.47*m*.

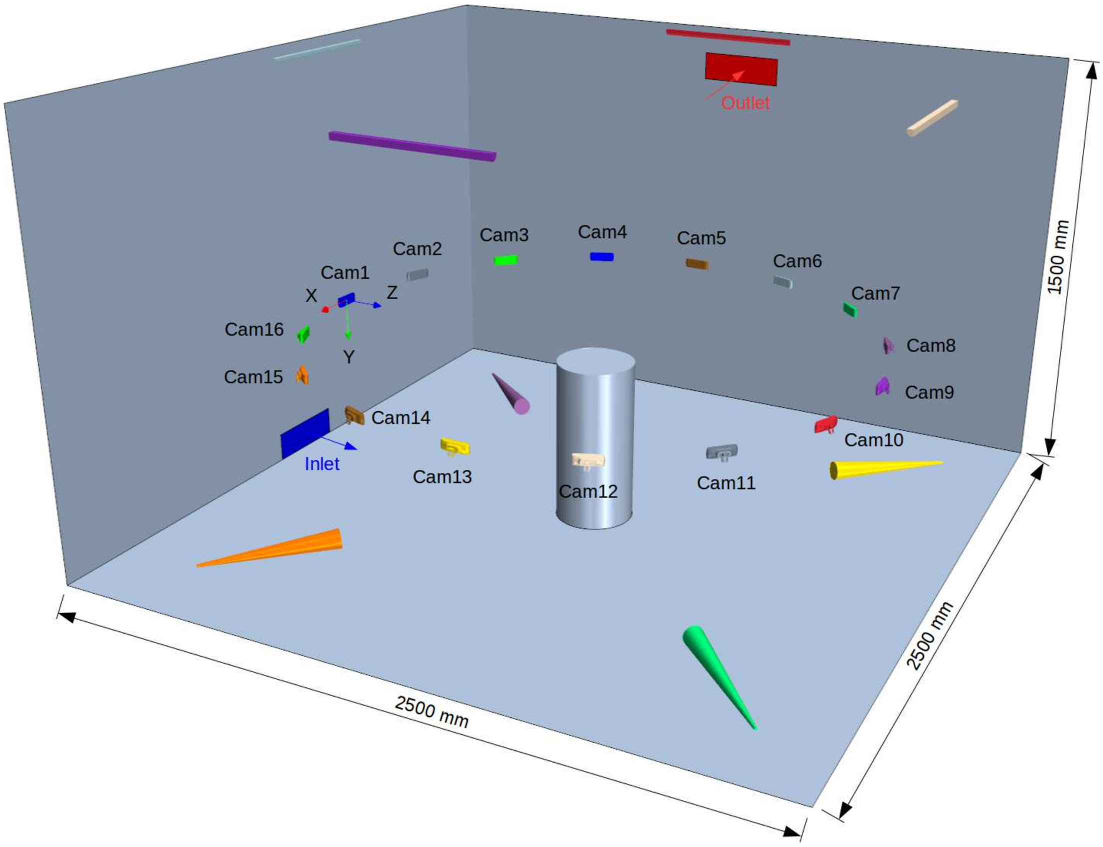


Figure 4.1: 3D-MVPTV Test Stand

For indoor air applications the 3D-MVPTV system uses helium-filled soap bubbles as tracer particles. These bubbles are generated using a so-called bubble generator shown in figure 4.2, which is developed and produced at the Hermann-Rietschel-Institut. The bubble generator mixes the appropriate amount of compressed helium and air and packs this mixture in a tenside based soap bubble using a coaxial micro pipe system. The proper amount and mixture of the gases in the bubble and the thickness of the bubble shell are adjusted manually based on previous experiences to generate density neutral tracer particles which also move with the smallest impulses in the air.

Christenfeldt measured an average diameter of $2mm$ for the helium-filled soap bubbles [9]. With its four cyclones the bubble generator can produce circa 2000 tracer particles per minute, which can live for circa 5 minutes [23]. Tracer particles are transported into the test stand with separate pipes connected to each of the four cyclones. Four diffusers are connected to the end of these pipes to reduce the impulses on tracers particles due to transport. These diffusers are positioned on the floor of the test stand as shown in figure 4.1.

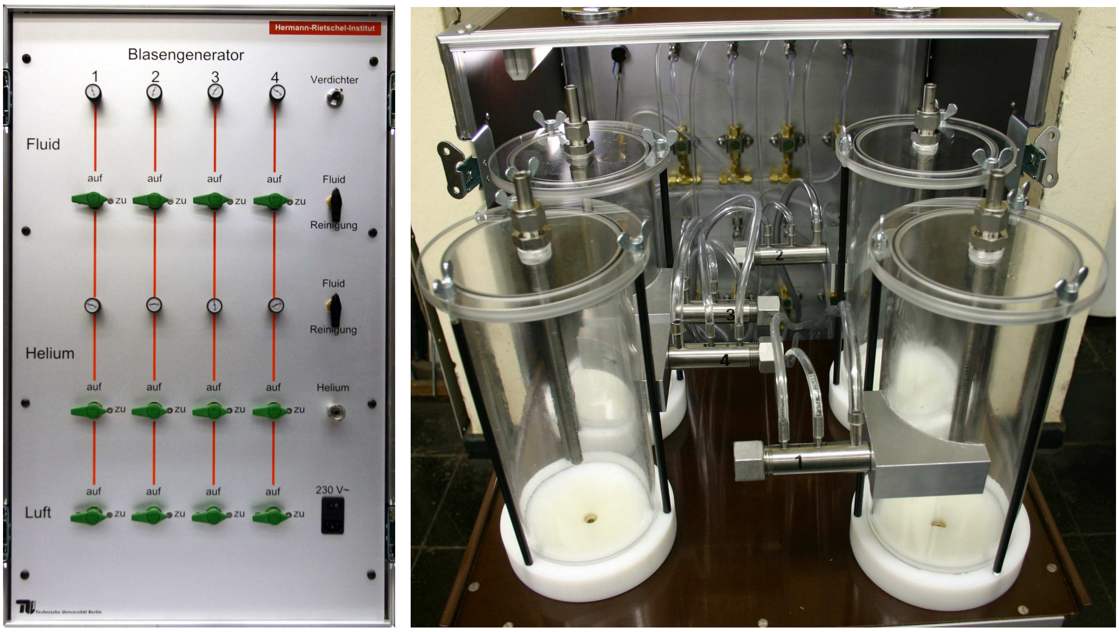


Figure 4.2: Front and back view of the bubble generator [23]

Illumination of the tracer particles is carried out using LED strip lights mounted centered on the ceiling in four directions (see figure 4.1). Each LED strip light has a length of $50cm$ with a luminous flux of $375lm$ and a power of $4.8W$. These LED strip lights produce warm white light with a color temperature between $3500K$ and $4000K$. Their beam angle is $120°$, which provides enough illumination for the measurement volume. Similar to the axial fans, the LED light strips are also controlled using a dimmer unit to adjust the desired illumination flux. In order to sustain high contrast between the tracer particles and the surfaces at the background, all surfaces of the test stand except the ceiling are covered with black stage molton with a fabric weight of $300g/m^2$.

4.2 Development of the 3D-MVPTV-SIL-Environment

„In software-in-the-loop phase, the actual production software code is incorporated into the mathematical simulation that contains the models of the physical system [36].“ Today SIL is frequently used in product development workflows of various industries. On the one hand, with this method, it is possible to accelerate the development phase and on the other hand costs due to experimental studies with hardware can be reduced.

Because of various aspects a SIL approach is crucial for the 3D-MVPTV development. However, in this study, a SIL-Environment for the 3D-MVPTV is developed mainly for focusing two main aspects: tracking efficiency and triangulation accuracy. The tracking efficiency can be defined as the percentage of the successfully recognized tracks of the total tracks. It characterizes the quality of the tracking algorithm. With real tracer particles, it is nearly impossible to gain information like the absolute coordinates of the tracer particles and the tracks they belong to. This drawback of the real tracer particles prevents calculating tracking efficiency using an experimental test stand. However, in SIL simulations virtual tracks can be defined with their assigned objects and the absolute coordinates of these objects. Using these tracks in a SIL 3D-MVPTV workflow enables not only the calculation of the tracking efficiency but also the 3D-MVPTV triangulation error.

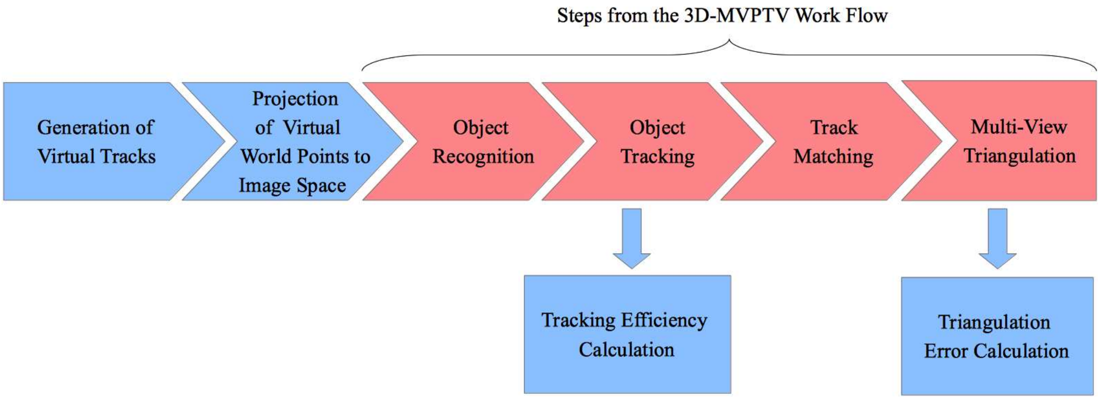

Steps from the 3D-MVPTV Work Flow
Generation of Virtual Tracks
Projection of Virtual World Points to Image Space
Object Recognition
Object Tracking
Track Matching
Multi-View Triangulation
Tracking Efficiency Calculation
Triangulation Error Calculation

Figure 4.3: 3D-MVPTV Software-in-the-Loop Work Flow

Figure 4.3 shows the SIL workflow for the 3D-MVPTV. In the first step virtual tracks are generated. Section 4.2.1 describes the methods used to define these tracks[12] with different properties. In the following step, the objects in the virtual tracks with their world coordinates are projected onto the image sensors of the cameras. Section

[12] In the SIL environment, „tracks“ refer to virtual tracks and „particles“ refer to virtual particles.

4.2.2 describes this step in detail. After the projection, the 3D-MVPTV workflow steps from object recognition to multi-view triangulation (see section 3.2) are applied. Finally, tracking efficiency and triangulation error can be calculated by comparing the results to the generated data which was used in the first place.

4.2.1 Generation of Virtual Tracks

In this study, two approaches to define the virtual tracks are implemented in the SIL environment: mathematically defined tracks and tracks generated by CFD simulations. If the investigated flow structure can be described mathematically (i.e., potential flow), the first method is used. For the complex flow structures which can not be easily defined mathematically, CFD simulations with Lagrangian particles are utilized. Modeling tracer particles as Lagrangian particles is a realistic approach in which interaction between tracer particles, surfaces, and Eulerian flow field are taken into account. Due to the different level of details of these two approaches, mathematically defined tracks are used in the early development steps whereby the functionality of the algorithms can be tested with simple track definitions. In the later development steps, tracks generated by CFD simulations are used to test the algorithms in complex geometries with realistic flow structures.

The two main variables needed to define a virtual tracer particle are its diameter and the world coordinates of its center (P_1). Figure 4.4 represents such a virtual tracer particle. As shown in this figure, six more points (P_2 to P_7) are used to define the minimum and maximum coordinates of this spherical tracer particles boundaries. These points can be calculated easily using particles' center coordinates and diameter. Through these points, spherical virtual tracer particles are mathematically defined as regular octahedrons. Section 4.2.2 describes the necessity for the further six boundary points used for each virtual tracer particle.

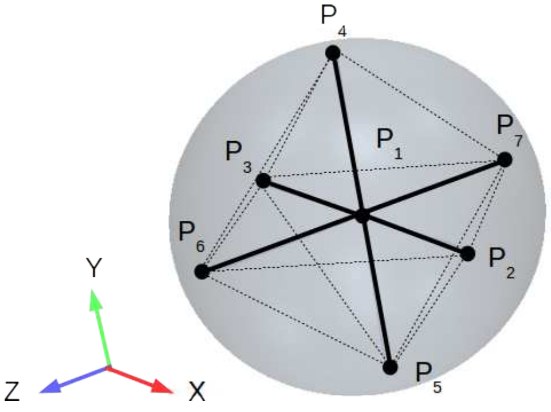


Figure 4.4: World points used to define a virtual tracer particle

Whether mathematically or by CFD generated, a bunch of variables is necessary to

Variable Name	Definition	Unit
Frame Rate	The frame rate of the cameras used.	fps
Maximum Physical Time	The time window in which the tracks are defined.	s
Track Type	Desired track type to generate. Five different track types are predefined: horizontal, vertical, sloped, helicoidal, and CFD generated.	n.A.
Velocity	The velocity of the tracer particles. It can be defined as constant or variable.	mm/s
Min. and Max. Particle Diameters	Minimum and maximum particle diameters of the tracer particles.	mm
Number of Starting Tracks per Second	The number of tracks starting each second.	s^{-1}
Virtual Measurement Volume Boundaries	Minimum and maximum coordinates of the virtual measurement volume, in which all particles are captured at least with two cameras for multi-view triangulation.	mm

Table 4.1: Variables for Virtual Track Definition

describe tracks. Table 4.1 shows an overview of these variables. The orientations of the tracks in this table are defined according to the camera coordinate system; for example vertical tracks are parallel to the y-axis. Similarly, the slope of the tracks is described through the angle between the track and the positive y-axis. Their radii and pitches describe helicoidal tracks. The variable velocity is defined as shown in equation 4.1. This equation is the mathematical formulation of a fully developed laminar pipe flow profile, whereby v is the flow velocity, and r is the pipe radius. The size of the particles is distributed randomly between minimum and maximum particle diameters for mathematically defined tracks. For the tracks generated by CFD, a log-normal distribution function is used as shown in equation 4.2 [44]. In this equation, D stands for the particle diameter, $\overline{D}$ for the mean particle diameter and σ for the standard deviation. In order to extract the boundaries of the virtual measurement volume, a function is developed. This function can distribute world points equidistant in every direction of the desired coordinate system (Cartesian, cylindrical or spherical). These world points are then projected on the sensor of each camera. The world points which are captured with less than two cameras are then excluded. This function is then extended for the calculation of triangulation error distribution in the virtual measurement volume (see section 4.2.4).

$$v(r) = v_{max}\left(1 - \frac{r^2}{{r_{max}}^2}\right) \tag{4.1}$$

$$F(D) = \frac{1}{2}\left[1 + erf\left(\frac{\ln D - \ln \overline{D}}{\sigma\sqrt{2}}\right)\right] \tag{4.2}$$

Once the virtual tracks are defined, and those outside the virtual measurement volume are excluded, each track is stored in a matrix. The size of these matrices is $N_F \times 21$, where N_F stands for the number of frames and 21 for the coordinates of the seven world points which define a tracer particle (see figure 4.4). Finally, these matrices are stored in an $N_{VT,w} \times 1$ cell array, where $N_{VT,w}$ is the number of the virtual tracks in world space.

4.2.2 Projection of Virtual World Points to Image Space

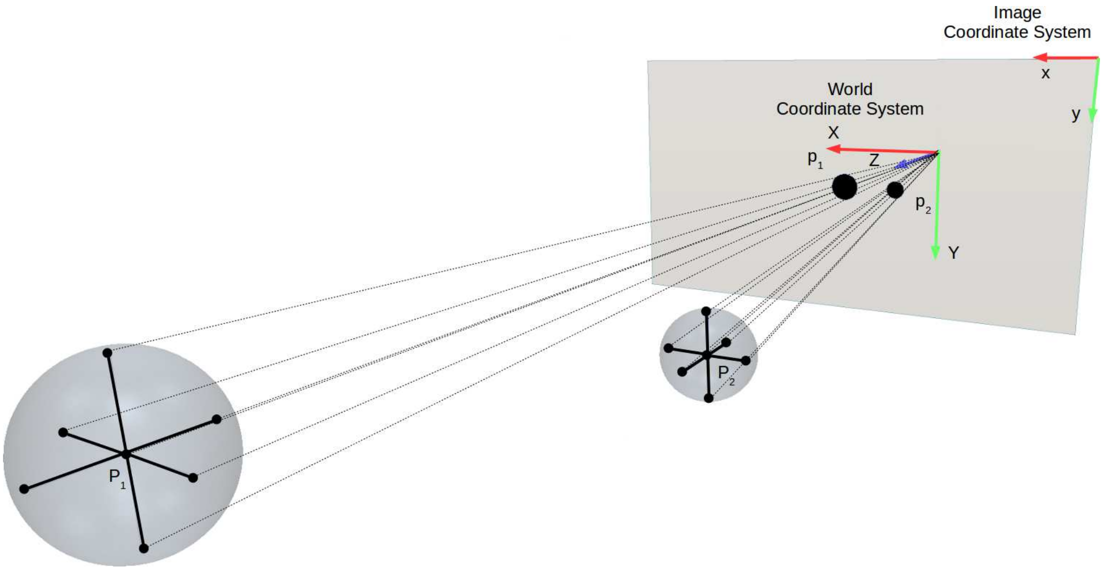


Figure 4.5: Projection of virtual tracer particles on a camera sensor

The projection step is similar to the image capture step, in which coordinate transformation from world coordinate system to image coordinate takes place. The inputs of this steps are the virtual tracks, intrinsic and extrinsic camera parameters. Either the camera parameters can be delivered in a generic definition, or as defined in chapter 3.2.3 they can be achieved by camera calibration. During the projection, the world points (P_1, P_2) in each frame are first projected on the image sensor of each camera as shown in figure 4.5. If one of these projected image points for a particle is outside the image sensor, this particle is then excluded so that just whole particles on the image sensor are considered. In case of one or more interruptions in the track due to this exclusion, the track is then split retrospectively into multiple tracks. Following that, the pixels inside the projected boundaries of each particle are set to 1 (white) for each frame. This step delivers image frames with tracer particles in $1-bit$ monochrome color space, which can be used in the object recognition step (see figure 4.3).

4.2.3 Tracking Efficiency Calculation

A significant advantage of the SIL-Environment is that using the information from the virtual tracks it is able to evaluate the algorithm's efficiency. After the virtual track generation and projection of the virtual tracks onto camera sensors, virtual tracking matrices can be generated. A virtual track matrix is an $N_{VT,i} \times N_F$ matrix, whereby $N_{VT,i}$[13] stands for the number of virtual tracks in image space and N_F for the number of frames. In a virtual tracking matrix, each row represents a virtual track filled with object IDs for each frame. Frames in a track without an object are filled with zeros. Similarly to virtual tracking matrices, tracking matrices are generated for each camera during object tracking step of the 3D-MVPTV workflow as described in chapter 6.

$$\eta_T = \frac{\sum\limits_{i=1}^{N_{VT,i}} \sum\limits_{j=1}^{N_{T,i}} \frac{\dim\left((S_{V\,i} = S_j)_{|S_{V\,i} \neq 0 \wedge S_j \neq 0} \right)}{\dim\left(S_{V\,i|\neq 0} \right)}}{N_{VT,i}} \tag{4.3}$$

The tracking efficiency η_T evaluates the tracking algorithm utilized. Equation 4.3 shows the mathematical definition of tracking efficiency. In this calculation, every track in tracking matrices, which are tracking vectors S and virtual tracking vectors S_V, is compared with each other. The size of the vector including identical nonzero elements of these vectors delivers the number of successfully tracked objects, as described in the numerator of this formulation. The division of this value by the number of tracked objects in the virtual tracking vector results in the tracking efficiency for each track. The summation of this value over all comparison possibilities between

[13] The number of the virtual tracks in image space can be higher than in world space if a track enters the image space more than once (i.e., helicoidal tracks) in which these tracks are then split.

tracking matrix and virtual tracking matrix, followed by averaging over the number of virtual tracks finally defines an average tracking efficiency. If a tracking algorithm can regenerate the virtual tracking matrix completely; it delivers a tracking efficiency of 100%.

4.2.4 Triangulation Error Calculation

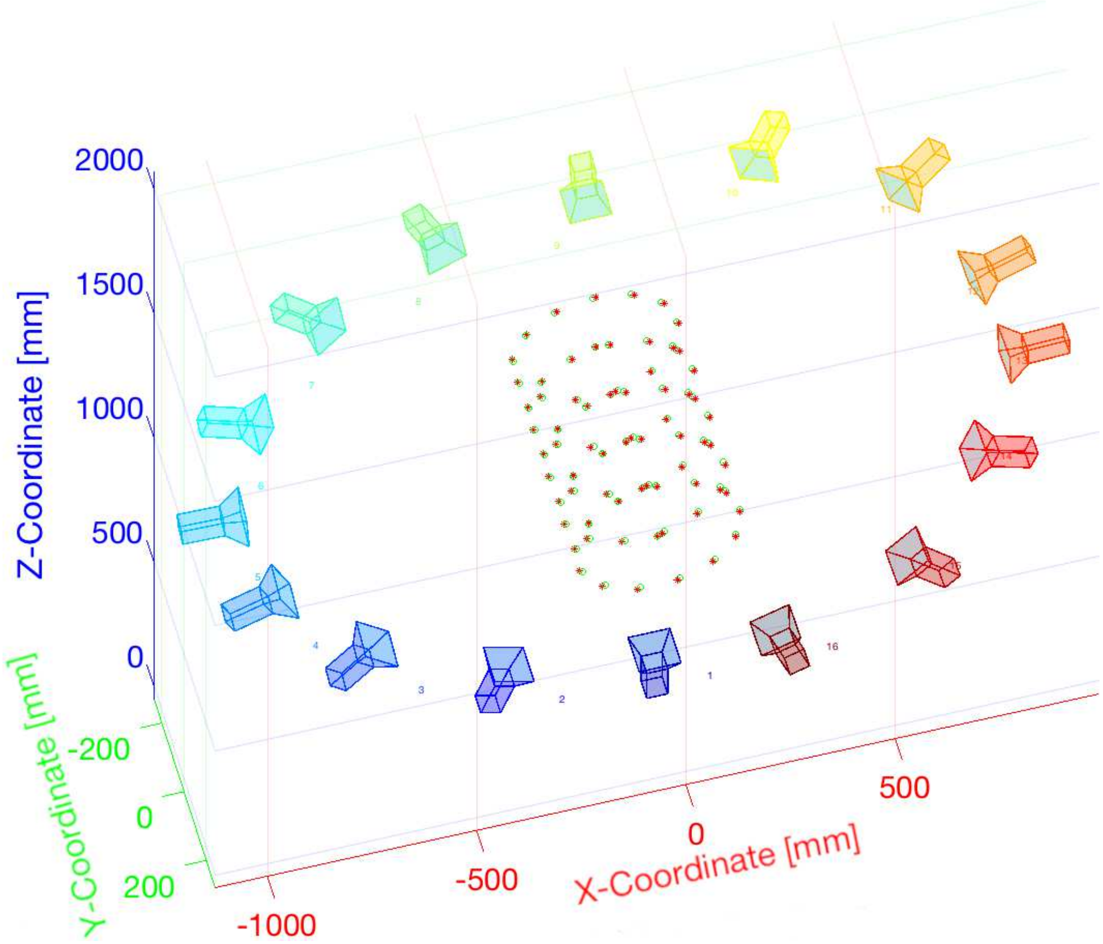


Figure 4.6: Triangulation error calculation

Another advantage of the 3D-MVPTV-SIL-Environment is its capability to determine the triangulation error. Unlike the reprojection error, the triangulation error cannot be calculated during real 3D-MVPTV measurements. However, compared to the reprojection error it is more relevant in practice since its results are represented in the world space, not in the image space. The 3D-MVPTV-SIL-Environment is extended with a function to calculate the triangulation error. In the first step of this function, world points are defined in the measurement volume. The number and the distribution of these world points can be controlled anisotropically in three different coordinate systems: cartesian, cylindrical, and spherical. Using one of these three coordinate systems makes it possible to achieve the desired point distribution for different modular camera networks. Once the world points are defined, in the next step these world points are projected onto the camera sensors. After this step, the world points and their corresponding image points on each camera are known. With this data, the multi-view triangulation is carried out which delivers the calculated positions of the world points. Figure 4.6 shows the defined and the triangulated world points. The triangulation error for each point is then the distance between their defined and calculated positions. This implemented function represents the results in minimum, mean and maximum values for the defined points.

This triangulation error calculation function can be utilized mainly for two purposes. First, it can be used after the camera calibration to estimate the accuracy of the calibrated camera set, thereby the accuracy of the 3D-MVPTV measurement. According to the results, the calibration can be enhanced either using further calibration images or repositioning the cameras in the set and repeating the calibration process. Similarly, this function can also be used to figure out how the choice of cameras used for the multi-view triangulation affects the triangulation accuracy, namely the triangulation error. Section 5.3 exhibits an experimental investigation of these two points.

5 Calibration of Multiple Cameras

In order to calculate the 3D world points using 3D-MVPTV, the modular camera network must be mathematically defined which happens in the camera calibration step of the 3D-MVPTV workflow. As described in section 2.2.4 the calibration of multiple cameras succeeds with several stereo camera calibrations followed by an optimization process using the bundle adjustment algorithm. This chapter describes these operations in the 3D-MVPTV workflow in detail. At the end of this chapter, several factors affecting the triangulation error are represented with experimental results. These experiments were carried out using the 3D-MVPTV-SIL-Environment described in the previous chapter.

5.1 Stereo Camera Calibration

Stereo camera calibration has two goals: Determination of intrinsic and extrinsic camera parameters. This is carried out in three main steps:

1. Several images of a calibration pattern are captured in different positions and orientations.

2. Image points of the calibration patterns' known features in the captured images are detected and associated with their world points.

3. Camera parameters are calculated utilizing numerical methods using these two point sets (image and world points).

Figure 5.1 shows the calibration board which was used in the first step of stereo camera calibration in this study. This calibration board has 558 internal corners to be detected, and the edge length of its squares is 18*mm*. It is digitally printed on an aluminum dibond plate so that its planarity and precision are guaranteed. The aspect ratio of this board is 16 : 9, which provides a similar resolution in both directions as the camera sensor used in this work. Furthermore, the calibration board ends with differently colored squares to distinguish the directions in case a square calibration board is used. These two properties of the calibration board define a sequence in which inner corners can be sorted so that calibration board rotation would not disturb this sequence.

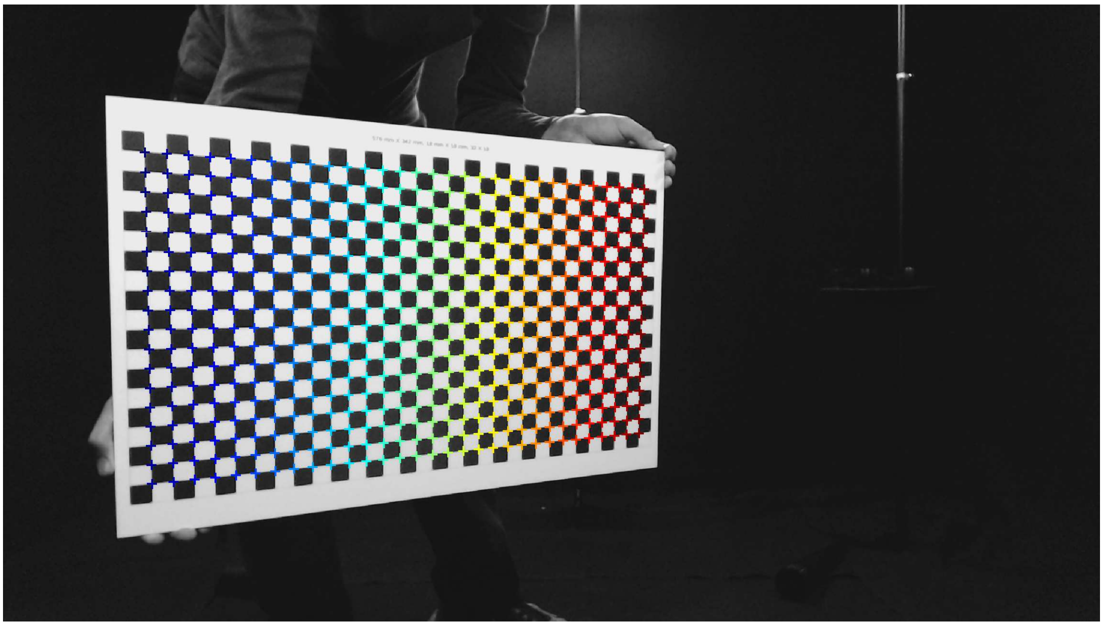


Figure 5.1: Calibration board with its detected 558 inner corners

In order to achieve acceptable calibration results, it is recommended to use 10 to 20 images per camera, in which the calibration pattern is positioned in the image plane with an angle of less than 45° in different orientations [31]. Image capture for camera calibration is carried out with MATLAB® and its Image Acquisition Toolbox™. After the images are successfully captured, using a function in Computer Vision System Toolbox the internal corners of the calibration pattern are detected as shown in figure 5.1. In this figure, the sequence of the inner corners is highlighted with a color legend from blue to red, in which dark blue represents the first and dark red the last image point. As mentioned above, in addition to the detection the sequencing of these detected points is crucial in order to capture different pattern orientations.

The Computer Vision System Toolbox uses singular value decomposition to estimate the camera properties from detected image calibration points and their associated world points. With this toolbox three further parameter sets are additionally to the practical implementation of the pinhole camera model on real cameras. The radial distortion coefficients define the image distortion due to the lens construction. In this study, three coefficients are used to describe the radial distortion. The tangential distortion defines the nonparallelism between lens and image planes. Finally, the last parameters estimate the skew between the image axes.

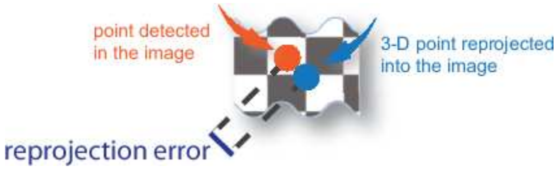


Figure 5.2: Graphical Representation of Reprojection Error [31]

The accuracy of camera calibration is evaluated using the reprojection error. In order to calculate the reprojection error, every detected point is reprojected back onto the image sensor using the estimated camera parameters. The reprojection error is the distance between the detected point and its reprojection in pixels as shown in figure 5.2. Then the mean value of the reprojection error for each calibration image is calculated. According to MATLAB® online documentation [31], „mean reprojection errors of less than one pixel are acceptable.“ In order to achieve calibration results with acceptable reprojection errors, a multi-pass filter algorithm is implemented. This algorithm checks mean reprojection errors for each calibration image. Following that, it excludes images with a reprojection error higher than one pixel from the calibration set and reruns the calibration till maximum reprojection error for each calibration image stays below one pixel.

Defining a modular camera network (see section 3.2.2) mathematically needs the estimation of relative transformation matrices, namely the extrinsic parameters between subsequent cameras and their intrinsic parameters. That is achieved utilizing stereo camera calibrations between subsequent stereo camera pairs. For N_C cameras, in case of a loop closure this results in N_C stereo camera pairs, and for the case without a loop closure, $N_C - 1$ stereo camera pairs can be defined. The stereo calibration environment developed in this study captures calibration images, runs corner detections and stereo calibrations for given stereo camera pairs automatically. The corner detection and the stereo calibration steps are executed in parallel on the cluster so that the calculation time does not increase dramatically with the increasing number of stereo camera pairs. The results of the stereo camera calibration are stored in a cell array for use in the next steps.

5.2 Multiple Camera Calibration With Bundle Adjustment

The absolute camera positions[14] can theoretically be calculated using the relative camera positions calculated in the previous step as shown in equation 5.1. However, with an increasing number of cameras, the estimation errors propagate with the multiplication of the transformation matrices B as described in section 2.2.4. The bundle adjustment optimizes this calculation and reduces the error of the estimated camera positions.

$$B_{i,j} = \prod_{n=i}^{j-1} B_{n,n+1} \tag{5.1}$$

A typical procedure for bundle adjustment is as follows:

1. Several calibration images are captured simultaneously by consecutive cameras. In the existing circular camera setup, a minimum of four successive cameras are distributed on circa 67.5°. This setup guarantees that these four consecutive cameras could recognize all the inner points in the calibration images. This operation is carried out sequentially for all cameras with different orientations of the calibration board.

2. With given relative camera positions[15] and matched image pairs for two subsequent cameras, the bundle adjustment algorithm (see 2.2.4) is run to reduce the mean squared reprojection error of the detected points of the image pairs. The result of this step is optimized camera positions.

3. A further camera is added to the existing optimized camera set with its relative camera position and matched images points. The absolute position of this new camera is calculated using equation 5.1. The new matched images are added to the set and this time the bundle adjustment is run for the new camera set with all three cameras.

4. Matching images between non-subsequent cameras (i.e., between camera one and three) are searched and added to the existing optimization set followed by a

[14] Coordinate system of the first camera represents the origin.

[15] Relative positions of the first two subsequent cameras are also their absolute positions since the camera coordinate system of the first camera represents the origin.

further bundle adjustment run.

5. The second and third steps are continuously carried out till the last camera in the set.

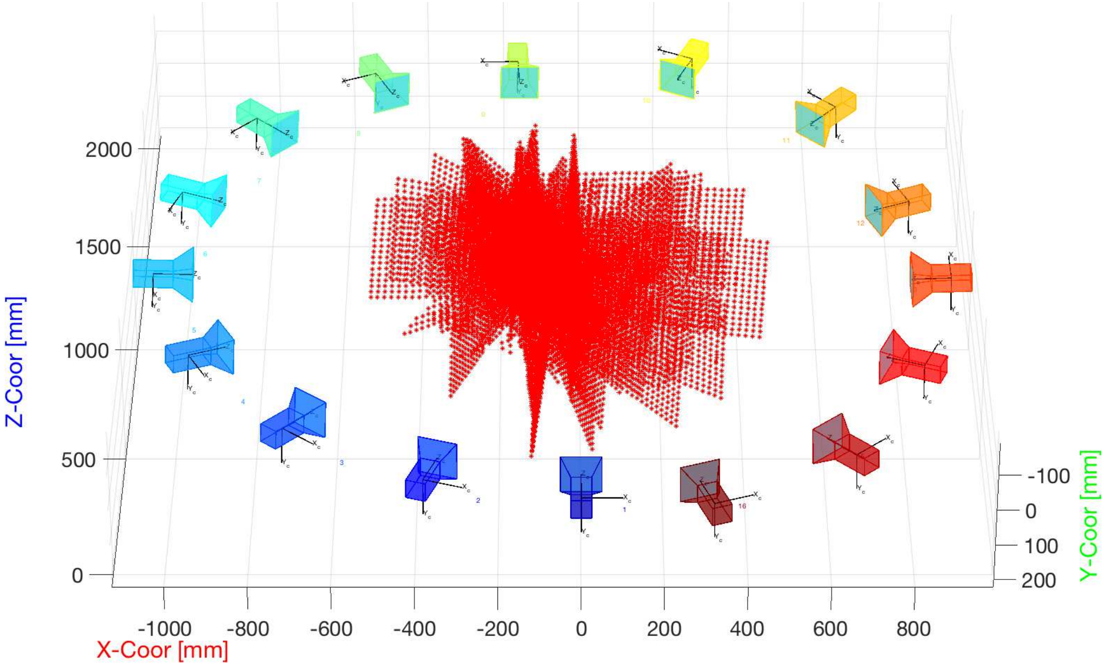


Figure 5.3: Resulting camera positions and world points used for bundle adjustment

Figure 5.3 shows the result of such a bundle adjustment calculation. In this figure, the camera positions and the inner corners of the calibration patterns used for the bundle adjustment are plotted. Similarly to the stereo camera calibration, the accuracy of the bundle adjustment is also determined with a mean reprojection error. An acceptable accuracy is achieved with the bundle adjustment if the mean reprojection error is below $10px$ [32]. That was not the case during the first camera calibrations in this study. The development of the mean reprojection error during the bundle adjustment with an increasing number of cameras in one of these first camera calibrations is shown figure 5.4 in red. This plot shows that the mean reprojection error stays around $30px$.

In order to improve the accuracy of the bundle adjustment, the existing algorithm was extended with a filtering option. To achieve this aim, after each bundle adjustment run, reprojection errors of every calibration pattern on each camera are separately calculated. These errors are then compared with the mean reprojection error of the bundle adjustment. The calibration images on the cameras with reprojection errors higher than the mean reprojection error of the bundle adjustment are excluded, and the bundle adjustment is rerun. This filtering cycle repeats until a considerable improvement is not possible anymore. Figure 5.4 also shows in blue the result of a bundle adjustment run with filtering. Compared to the bundle adjustment run without filtering, this filtering option reduces the mean reprojection error by 50% in the early steps and by 90% in the final ones resulting in a mean reprojection error of circa $3px$.

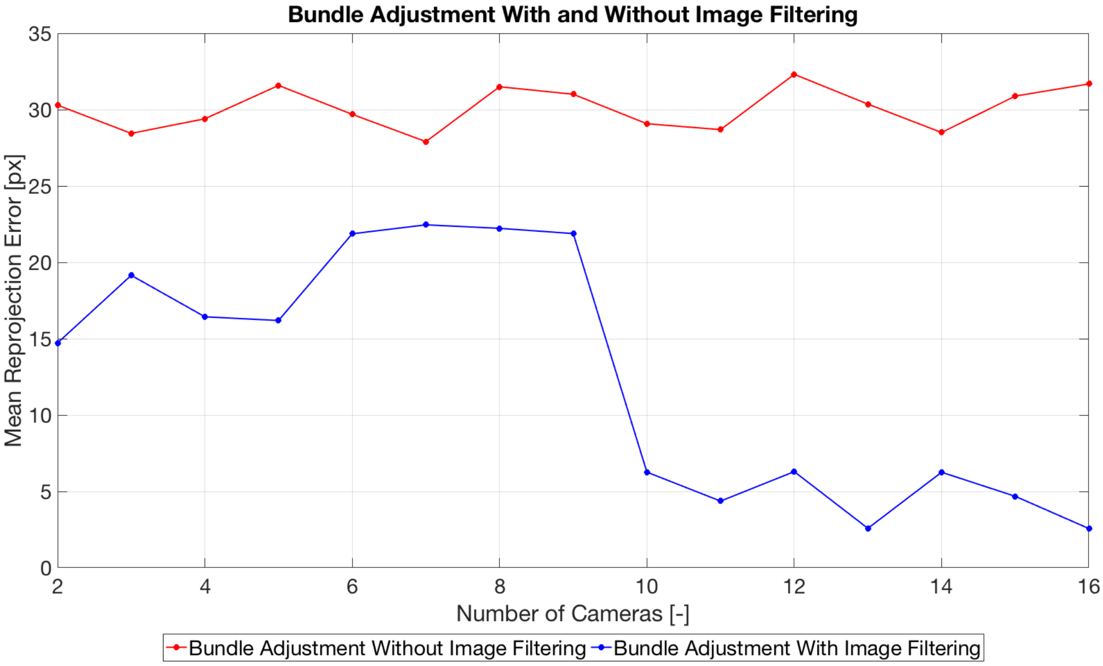


Figure 5.4: Mean reprojection error during bundle adjustment with and without image filtering

The bundle adjustment algorithm in the Computer Vision System Toolbox has a couple of restrictions. First, it is only capable of working with one sole intrinsic matrix (in version R2017b), which means all the cameras for the bundle adjustment must be identical. That can lead an error even for the identical cameras due to low precision in the camera manufacturing process. The second restriction is the maximum angle between subsequent cameras. This angle must not be too large (i.e., 90°).

5.3 Choice of the Camera Set for Multi-View Triangulation

The experiment with the table tennis ball described in section 3.2.8 gives a first idea about the relationship between the camera set used and the triangulation error. However, it is questionable whether the results can be generally applied to the 3D-MVPTV. The main reason for this is that the observed object regularly moves in a tiny constant part of the measurement volume. Also, the error during the detection of the table tennis balls center is different when helium-filled bubbles with an average diameter of 2*mm* are used. For these reasons the 3D-MVPTV-SIL-Environment, which is described in section 4.2 was developed.

The 3D-MVPTV-SIL-Environment was utilized to find out the relationship between the number of cameras used for the triangulation and the resulting triangulation error. 120 virtual world points were distributed in the observed volume under this SIL-Environment as shown in figure 4.6 so that 16 cameras could capture all of these points.

Figure 5.5 below shows the mean values of the triangulation errors of the 120 points for all camera set combinations. The camera sets with the same number of cameras are grouped and annotated on the horizontal axis. In order to figure out the relationship of the angle between cameras and the triangulation error, the mean angles between cameras were calculated. The data points in this graph are colored according to these values. It can be seen that the lowest mean triangulation errors of circa 4*mm* occur for the camera sets with five and six cameras. On the other hand, the highest mean triangulation error occurs with two cameras, whereby the angle between cameras is circa 90°. This graph also shows that the lowest mean triangulation errors are observed when the mean angle between the cameras is less than 50°.

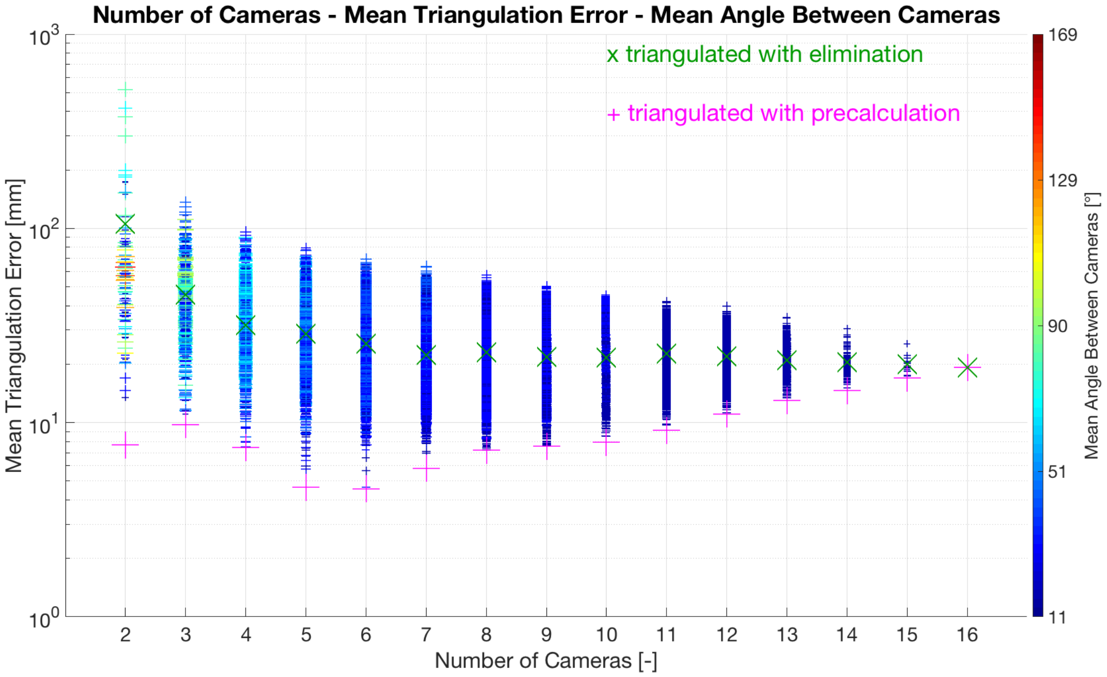


Figure 5.5: Mean triangulation error and mean angle between cameras for all camera set combinations

The two multi-view triangulation methods described in section 3.2.8 were also compared under the SIL-Environment. The aim of this comparison is the evaluation of these triangulation methods regarding their ability to find the camera set with the minimum triangulation error. In order to achieve this, the same 120 virtual world points were triangulated using these methods but with a restricted number of cameras so that they can also be easily compared with the existing results. Figure 5.5 exhibits the results using multi-view triangulation with elimination in dark green. It can be seen that the results with this algorithm are within the upper half of the existing mean triangulation errors. On the other hand, the multi-view triangulation method with precalculation finds the optimum camera set for all number of cameras. This result is not unexpected since the cost function of this method is the mean triangulation error.

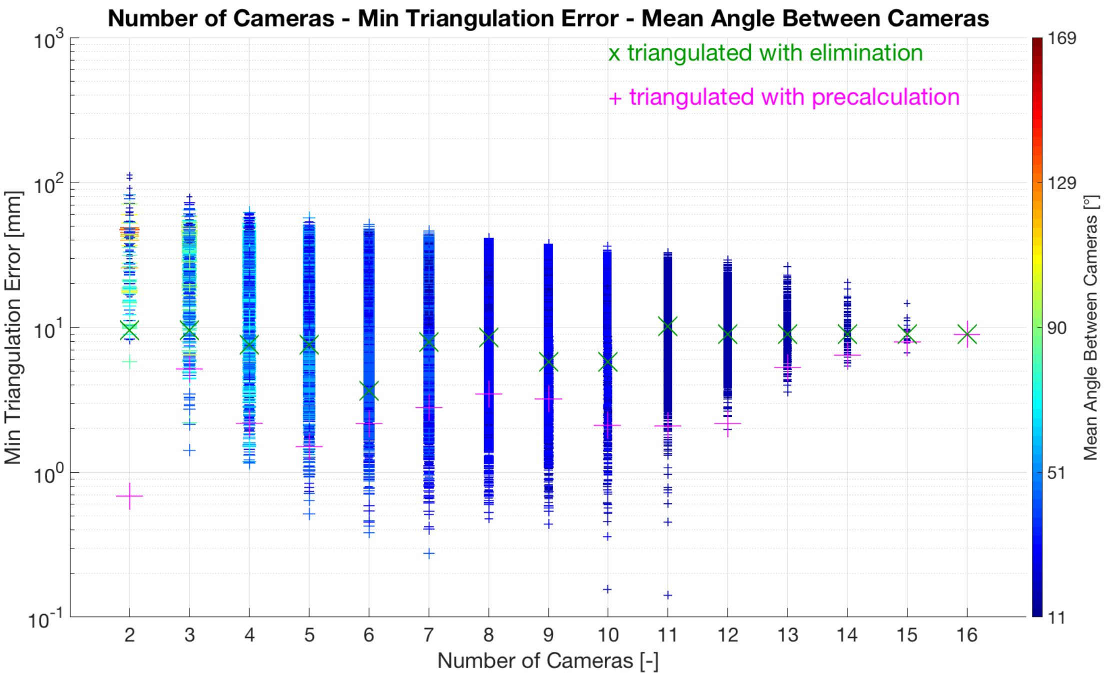


Figure 5.6: Minimum triangulation error and mean angle between cameras for all camera set combinations

For a better representation of the triangulation error distribution in the observed volume, figure 5.6 shows the minimum triangulation errors. Regarding the minimum triangulation errors, triangulations with ten and eleven cameras deliver the best results with around $0.1mm$. The mean angle between cameras resulting in the best value for the minimum triangulation error increases up to circa 90°. A triangulation with two cameras still delivers the highest value when the minimum triangulation error in the observation volume is considered. In summary, a comparison of the multi-view triangulation methods shows that the triangulation with precalculation delivers considerably better results than the triangulation with elimination. However, due to its underlying algorithm, it is far from finding the camera set that delivers the minimum triangulation error.

Figure 5.7 represents the maximum triangulation errors resulting from all camera set possibilities. Similarly to the result of the minimum and mean values, the maximum triangulation error occurs when a camera set with two cameras is used. Similarly to the mean values plotted in figure 5.5, the lowest values for the maximum triangulation errors result when the mean angle between cameras are around a bit less than 50°. The camera sets with five, six and seven cameras deliver these lowest values around $10px$. The comparison between multi-view triangulation methods shows that the triangulation with precalculation reaches the lowest and the triangulation with elimination reaches the highest values of the maximum triangulation error.

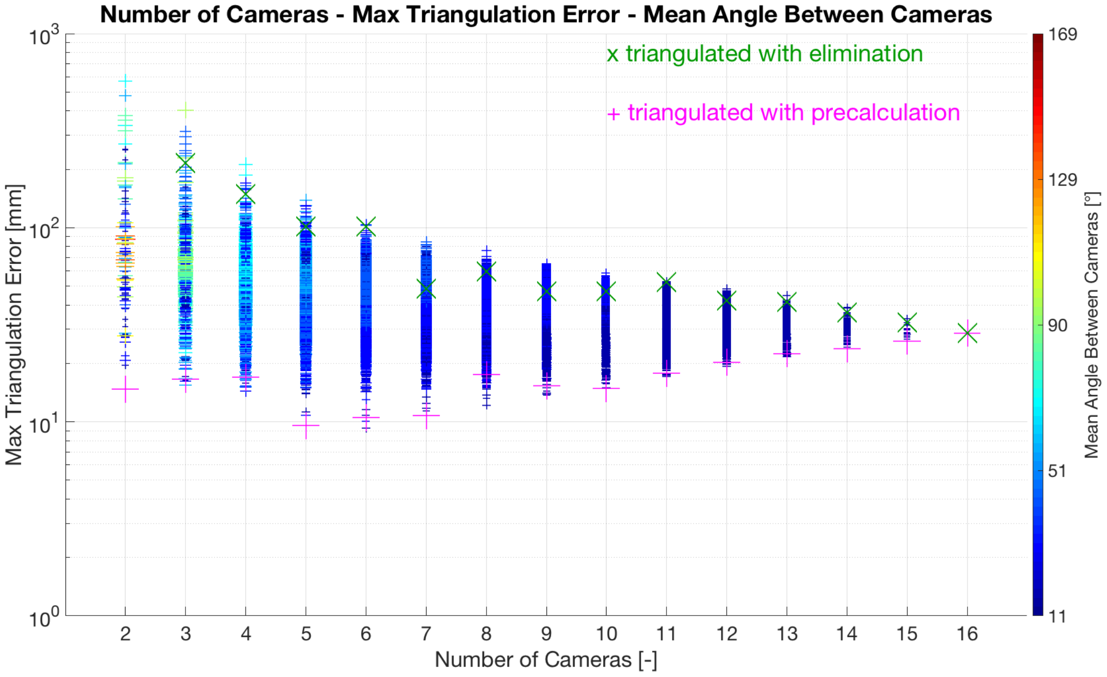


Figure 5.7: Maximum triangulation error and mean angle between cameras for all camera set combinations

The results in the last three figures above show the mean, minimum and maximum values for the triangulation error in the observed domain. Additionally to these values the standard deviations of the triangulation errors help us to figure out the error distribution in the volume. In this context, lower standard deviations of triangulation errors exhibit a more homogenous error distribution. Figure 5.8 shows the standard deviations of the triangulation errors. In this figure, the lowest values of the standard deviation are observed for the triangulations with four, five and six cameras. As for the mean, minimum and maximum values, the highest value for the standard deviation is delivered when two cameras are used. An interesting result regarding the mean angle between cameras and the standard deviation of the triangulation errors is that the lowest value for three cameras is achieved with an angle of 70°, which is slightly higher than the camera sets with other numbers of cameras.

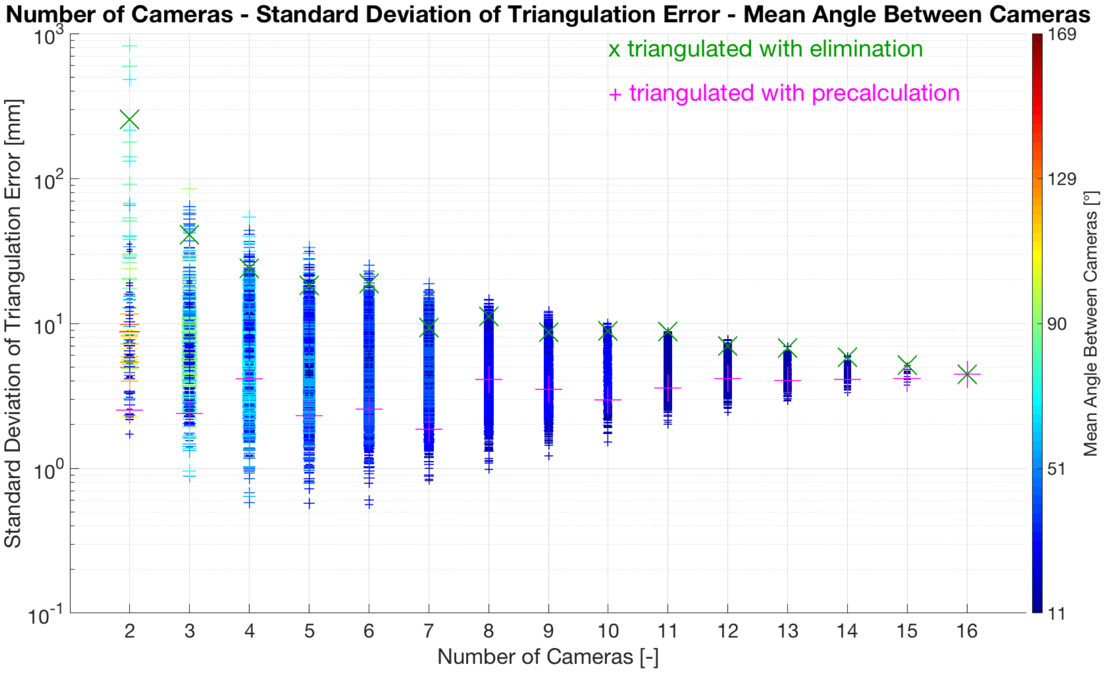


Figure 5.8: Standard deviation of the triangulation error and mean angle between cameras for all camera set combinations

The comparison of the multi-view triangulation methods regarding the standard deviations of the triangulation errors delivers similar results as for the minimum, mean and maximum values. The triangulation with elimination results significantly higher standard deviation values than the triangulation with precalculation, which are very close to the highest values. On the other hand, even though the triangulation method with precalculation cannot reach the lowest values of the standard deviation, it stays below the lower half of the data points.

When all the data points represented in detail above are statistically processed, the relationship between the number of cameras used, the triangulation error and the reprojection error becomes more clear. In figure 5.9 top, the minimum and mean values of these values are shown. With the ascending number of cameras, the minimum reprojection error increases showing a linear character. Compared to this value the minimum triangulation error stays nearly constant till eleven cameras and then increases also linearly. The trends for the mean values of the triangulation and reprojection error grow in the opposite direction. While the mean triangulation error decreases with the ascending number of cameras asymptotically, the mean reprojection error increases.

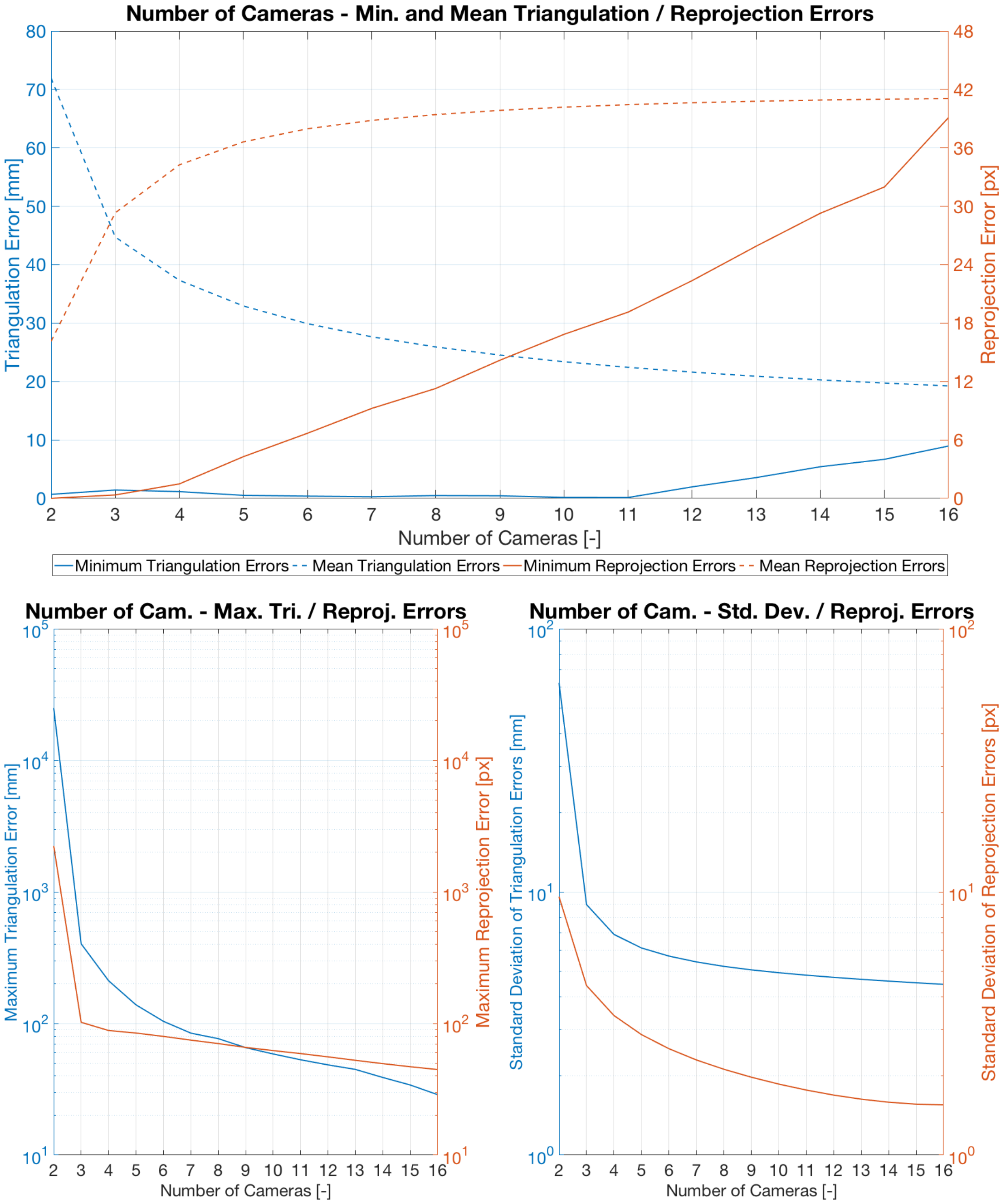


Figure 5.9: Minimum, mean, maximum and standard deviation values of triangulation and reprojection errors depending on the number of cameras utilized

When we consider the maximal values of the triangulation and reprojection errors as shown in figure 5.9 bottom left, these values decrease with the ascending number of cameras. In the same figure on the bottom right side, the trends for the standard deviations of the triangulation and reprojection errors are shown. These values behave similar to the maximum values and decrease with the ascending number of cameras.

Considering these generalized data in figure 5.9, choosing a camera set with eleven cameras would be a good choice. This will lead to circa $0.1px$, $23px$, $50px$ of minimum, mean and maximum triangulation errors with a standard deviation of circa $5px$. However, if we consider the individual data points in figures 5.5, 5.6, 5.7, and 5.8, choosing a camera set with five cameras would be a better choice. Such a camera set delivers $1.5px$, $4.5px$, $10px$ of minimum, mean and maximum triangulation errors with a standard deviation of circa $2px$. While the minimum triangulation error increases, the mean and maximum values decrease by circa 80% and the standard deviation by 60%, which means a more homogeneous error distribution. This comparison shows clearly how a significantly lower triangulation error can be achieved when instead of generalized data individual data points are considered.

6 Multi-Pass Tracking Algorithm with Robust Initialization

During the development of the 3D-MVPTV concept, a conventional tracking algorithm was already implemented. However, due to its decreasing tracking efficiency with increasing number of tracer particles, the necessity for a more comprehensive tracking algorithm became apparent. Dabiri and Gharib also mention this point: „However, because of the errors involved in identifying particle pairs in high particle density images, the design of an automatic particle tracking method, especially for three-dimensional flows has been extremely challenging.“ [1] This section presents the state of the art tracking algorithms, followed by the conventional tracking algorithm which here was implemented first. Following that, the newly developed multi-pass tracking algorithm with robust initialization is presented. At the end of this section, the performance of the conventional tracking algorithm and the new tracking algorithm is compared using SIL simulations.

6.1 State of the Art Tracking Algorithms

The methods of tracking algorithms can be generally classified into two groups according to Foss, Tropea, and Yarin: region-based and feature-based methods [7]. Region-based methods work with image patterns in order to calculate the motion of the tracer particles. Image patterns of subsequent frames are compared to carry out this calculation. One flow visualization technique in which region-based methods are utilized is PIV. Feature-based methods on the other hand first extract the features of the tracer particles which are tracked as described in section 3.2.5. After this step, the tracking takes place using the properties of these extracted objects. PTV works with feature-based methods to track every tracer particle separately. Since in this study a new PTV method is developed, feature-based methods are primarily in focus.

A comprehensive overview given by Foss, Tropea, and Yarin [7] categorizes the feature-based tracking algorithms in five groups. *Two-frame tracking* uses a nearest neighbor search, and is suitable for low particle densities. *Multiframe tracking* is similar to the two-frame tracking method. Unlike that, the possible position of the particle in the next frame is calculated over multiple frames. Using this calculated position as the center and a search radius, it restricts the search field for the new candidate particles. *Combinatorial optimization* uses a cost function to associate particles in two successive frames. *Statistical techniques* calculate the possible positions of the objects in the next frames using information from the previous frames and use this information for associating particles. *Hybrid methods using velocity estimation techniques* use a combination of region-based and feature-based methods. An example of the hybrid methods is using PIV for the estimation of the velocity field to track with combinatorial optimization.

Each of the feature-based tracking algorithms described above has their advantages and disadvantages. Among these, using present velocity and, if available, the acceleration information is a huge advantage. With this step, not only the object properties but also the flow information is integrated into the tracking algorithms, which generalizes their applicability. However, initial velocity information is needed, which increases the importance of initialization for successful tracking. Hybrid methods are utilized to overcome such difficulties.

6.2 Conventional Tracking

The conventional tracking algorithm was implemented in the early stages of the 3D-MVPTV-Development. For the initialization of particles appearing first time on the image plane, in this tracking algorithm a cross-correlation based method is used. This cross-correlation is run on an $N_{o,f} \times N_{o,f-1}$ initialization decision matrix K, whereby $N_{o,f}$ stands for the number of objects in the f^{th} frame. This initialization decision matrix is calculated using a cost function shown in equation 6.1. This cost function calculates the deviations of the ratios between areas and positions of the objects in two successive frames. Hereafter, these deviations are summed using an under-relaxation factor for the object area. Handily, a zero element of the initialization decision matrix stands for an object which does not move between two successive frames.

$$K_{f,f-1} = \alpha \left| \frac{A_f}{A_{f-1}} - 1 \right| + (1-\alpha) \left| \frac{\frac{x_f}{x_{f-1}} + \frac{y_f}{y_{f-1}}}{2} - 1 \right| \tag{6.1}$$

A filtering function using a search radius r_s is implemented to increase the probability of a successful initialization. This filtering function eliminates the pairs in the initialization decision matrix if the distance between them is less than this search radius. The maximum value of this search radius can be chosen as the half of the image sensors hypotenuse, whereby image points of three successive frames are located on the longest possible linear distance. The minimum search radius can be chosen depending on the velocity of the tracer particles on the image sensor. This initialization method used in the conventional tracking algorithm is represented graphically in figure 6.1. The figure shows three different tracks a, b, and c with their image points p in the first frames. For this case, search radius filtering results in just one possible successful match for the tracks a and b. However, for the track c two possible matches, $p_{c,f+1}$ and $p_{a,f+1}$, are available which are associated using cross-correlation with the initialization decision matrix.

After the initialization, the object tracking is carried out using a cross-correlation based algorithm as well. In this case, similar to the initialization decision matrix, a tracking decision matrix H is defined as shown in equation 6.2. In this equation, the distance between the object positions in a frame p_f and the estimated positions of the objects in the previous frame for the same frame p'_{f-1} is calculated. Equation 6.3 shows the calculation of the estimated positions for the actual frame of the objects seen in the previous frame. In this equation, p_{f-1} stands for the object position in the previous frame and p_{f-2} for the object position in the frame before this previous frame. The estimated position of the object p'_{f-1} is calculated under the assumption that the object velocity stays constant over these three frames.

$$H_{f,f-1} = \sqrt{(p_f - p'_{f-1})^2} \tag{6.2}$$

$$p'_{f-1} = (p_{f-1} - p_{f-2}) + p_{f-1} \tag{6.3}$$

All these initialization and tracking calculations are combined in the conventional tracking algorithm, whose flowchart is shown in figure 6.2. This algorithm first checks if there are any tracks with objects inside from the last two frames. If that is not the

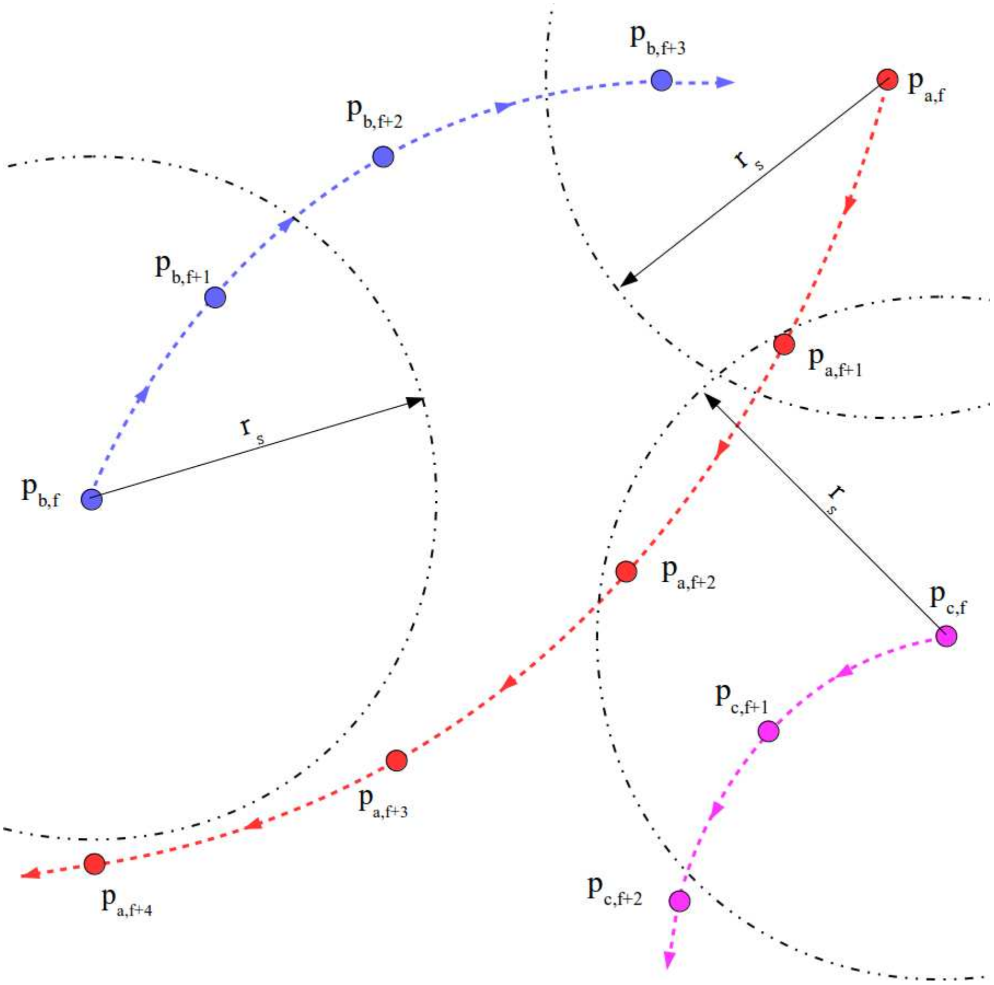


Figure 6.1: Initialization method utilized in the conventional tracking algorithm

case, it initializes the current frame and goes to the next frame. Otherwise, it runs tracking in the current frame using the tracking decision matrix. After the tracking completed for the current frame, untracked objects in the current and previous frames are initialized. This calculation cycle is carried out from second to the last frame, which means the algorithm runs one-directional and forwards.

During tracking it is possible that some tracks are split. One common reason for this is that the search radius for the initialization is chosen not large enough to include the right object. In this case, this track is reinitialized which results in two split tracks instead of one track. In order to correct such errors an algorithm is developed which combines these split tracks after tracking is finished. This algorithm first chooses possible split tracks, in which there are no frames between start or end frames of these tracks. Following that, the objects' positions in the previous and the next frames before and after the start frame $p'_{b,f}$ and the end frame $p'_{a,f+1}$ are estimated as shown in figure 6.3. This position estimation calculation is carried out under a constant ve-

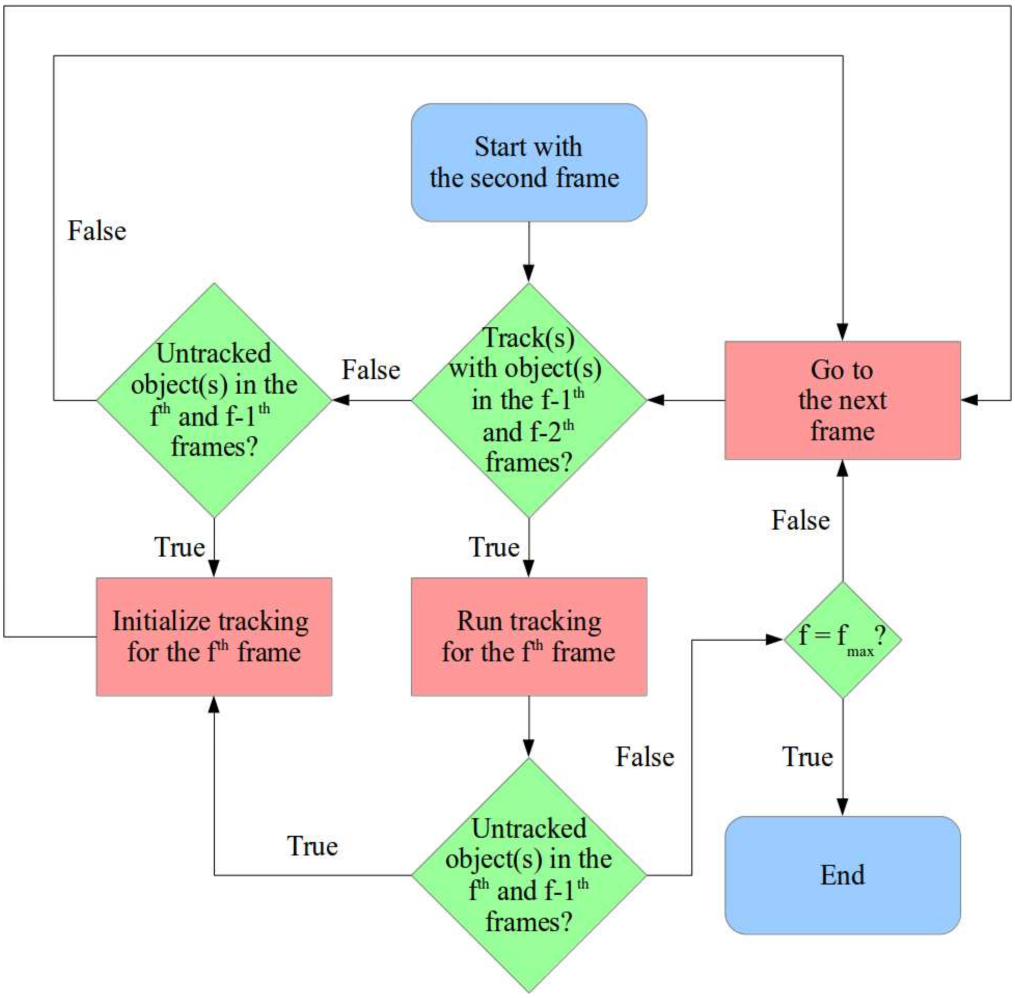

Start with the second frame
Track(s) with object(s) in the f-1th and f-2th frames?
Untracked object(s) in the fth and f-1th frames?
Go to the next frame
Initialize tracking for the fth frame
Run tracking for the fth frame
Untracked object(s) in the fth and f-1th frames?
f = fmax?
End
False
False
True
True
False
True
False
True

Figure 6.2: Flowchart of the conventional tracking algorithm

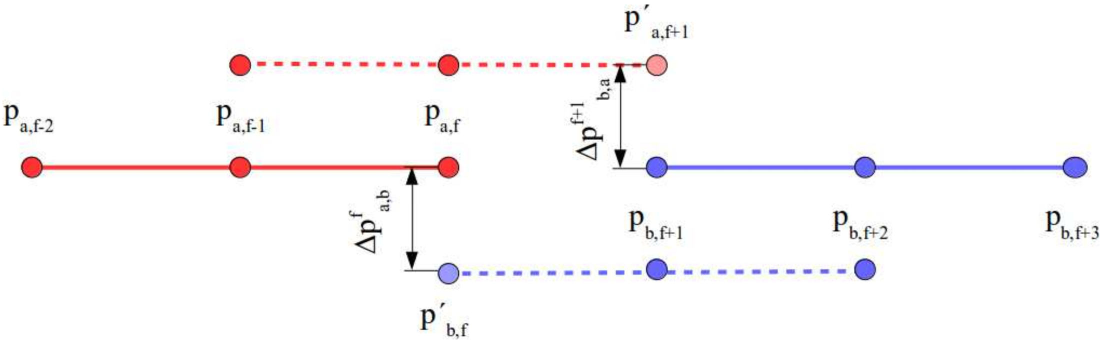


Figure 6.3: Graphical representation of the algorithm combining split tracks

locity assumption and calculated using the last two frames as in the tracking decision matrix. In the next step, the deviation of the estimated positions from the candidate object positions $\Delta p^{f}{}_{a,b}$, $\Delta p^{f+1}{}_{b,a}$ is calculated. It is essential to carry out this deviation calculation bi-directionally to reduce possible erroneous track combinations. Finally, these two deviation values are first filtered using a maximum value and then their mean value is used in a cross-correlation based matching calculation. Since more than two splits of a single track is possible, this operation is continued in multiple passes till no tracks to combine are left.

6.3 Multi-Pass Tracking With Robust Initialization

The conventional tracking algorithm described above has some characteristics which quite possibly reduce the tracking efficiency. The first significant problem is that the initialization is not carried out with flow relevant variables like velocity, but with object area and minimum distance. That can lead to erroneous initializations, on which a successful tracking is not possible. Another problem is choosing a search radius for the initialization. Different flow regimes can have different track velocities which can lead to different velocities on image planes. Under these circumstances, using a constant search radius size[16] for all tracks through all frames can result in failed tracking results. The conventional tracking algorithm processes image sequences from beginning to end with ascending frame numbers one-dimensionally. This one-dimensional one-pass methodology can fail especially for complex flow structures with high seeding rates.

The multi-view based 3D-MVPTV developed in this study needs a more stable tracking algorithm to deliver better results especially under „extremely challenging“ conditions like high particle densities as mentioned by Dabiri and Gharib [1]. The inspiration for developing a tracking algorithm with higher tracking efficiency comes from a similar field to flow visualization, from CFD. In CFD the solvers work iteratively often on a uniformly initialized flow field. A new method for initialization of flow fields is the Grid Sequencing Initialization. This method calculates the flow field inviscidly for the initialization in a couple of seconds, which delivers a better flow regime for the further iterations. The results are less computational time due to fewer iterations for a converged solution and a more robust simulation [3].

This idea—using a better initialization resulting in a more robust calculation—was the first step in developing a new tracking algorithm. During development of the so-called „Robust Initialization“ two main objectives were followed: Using parameters which are relevant for the investigated flow regime and finding the most suitable frame for the initialization instead of initializing with the first frame. In order to achieve the latter, first all combinations of the midpoints using the objects from the first and

[16] Section 6.5 and 6.4 represent the relation between search radius size and tracking efficiency in detail with SIL simulations.

the third frame in three consecutive frames are calculated. Thereafter, the average deviation between these estimated midpoints and the objects in the second frame is calculated. Equation 6.4 shows the mathematical formulation of this calculation. In this equation, $N_{o,f}$ stands for the number of objects in the f^{th} frame and $d'(f)$ for the average midpoint deviation as mentioned above.

$$d'(f) = \frac{\sum_{i=1}^{N_{o,f-1}} \sum_{j=1}^{N_{o,f}} \sum_{k=1}^{N_{o,f+1}} \sqrt{\left(\frac{p_{i,f-1}+p_{k,f+1}}{2}\right)^2 + p_{j,f}{}^2}}{N_{o,f-1}N_{o,f}N_{o,f+1}} \tag{6.4}$$

If three frames are used for the robust initialization, there is just one average midpoint deviation for each frame to calculate. That is the case in figure 6.4 for the track c, which is plotted in magenta. The most suitable frame to initialize, namely the frame with minimum average midpoint deviation, is found out using equation 6.5. In this equation average midpoint deviation is calculated for every possible frame, whereby f_{max} stands for the maximum frame number, and the frame with the minimum average midpoint deviation is chosen as the initialization frame.

$$\underset{f \in \{2,\ldots,f_{max}-1\}}{\operatorname{argmin}} \quad d'(f) \tag{6.5}$$

However, during the development the experience was made that using just three frames for the initialization can lead to wrong results, especially with high particle densities. In order to prevent this, one more frame can be included in the initialization calculation, so that it is carried out using four frames. Figure 6.4 shows this scenario for the track b, which is plotted in blue. If four consecutive frames are used for the initialization, two average midpoint deviation calculations are possible. These two values are then averaged as shown in equation 6.6.

$$\underset{f \in \{2,\ldots,f_{max}-2\}}{\operatorname{argmin}} \quad \frac{d'(f)+d'(f+1)}{2} \tag{6.6}$$

Alternatively, the initialization can be carried out with five consecutive frames. Using five frames for initialization delivers four possible average midpoints deviations as shown in the track a in figure 6.4, which is plotted in red. Other than the initializations with three and four frames, using five frames makes it possible to run the average midpoint deviation calculations not only through three consecutive frames but also through the first, third and fifth frames of a sequence with five successive frames. In this case, the average midpoint deviation $d''(f)$ is calculated using equation 6.7. The initialization frame with minimum average midpoint deviation is decided using the equation 6.8.

$$d''(f) = \frac{\sum_{i=1}^{N_{o,f-2}} \sum_{j=1}^{N_{o,f}} \sum_{k=1}^{N_{o,f+2}} \sqrt{\left(\frac{p_{i,f-2}+p_{k,f+2}}{2}\right)^2 + p_{j,f}{}^2}}{N_{o,f-2} N_{o,f} N_{o,f+2}} \tag{6.7}$$

$$\underset{f \in \{3,\ldots,f_{max}-2\}}{\operatorname{argmin}} \frac{d'(f-1)+d'(f)+d'(f+1)+d''(f)}{4} \tag{6.8}$$

The tracking algorithm used after the robust initialization differs from the conventional tracking algorithm. Since the initialized tracks could have objects in the previous and the following frames, tracking must be carried out bi-directionally, forwards and backward. Therefore, the tracking decision matrix H is calculated for both directions as shown in equation 6.9. Because the robust initialization delivers information from more than two frames, not only velocity but also the acceleration is taken into account during the calculation of the estimated object positions in the previous and following frames (see equation 6.10).

$$H_{f,f\pm1} = \sqrt{(p_f - p'_{f\pm1})^2} \tag{6.9}$$

$$p'_{f\pm1} = \begin{cases} \frac{(p_{f\pm1}-p_{f\pm2})^2}{(p_{f\pm2}-p_{f\pm3})} + p_{f\pm1} & \text{if } p_{f\pm3} \neq 0 \\ (p_{f\pm1} - p_{f\pm2}) + p_{f\pm1} & \text{if } p_{f\pm3} = 0 \end{cases} \tag{6.10}$$

The second important feature of this new tracking algorithm beside its robust initialization is the multiple pass character. The multi-pass tracking is not carried out sequentially with one frame after the other as in the conventional tracking algorithm. Instead, the tracks are initialized using robust initialization, and these tracks are then tracked bi-directionally in all adjacent frames till no corresponding objects could be found. Figure 6.5 represents this multi-pass workflow through a flowchart. The al-

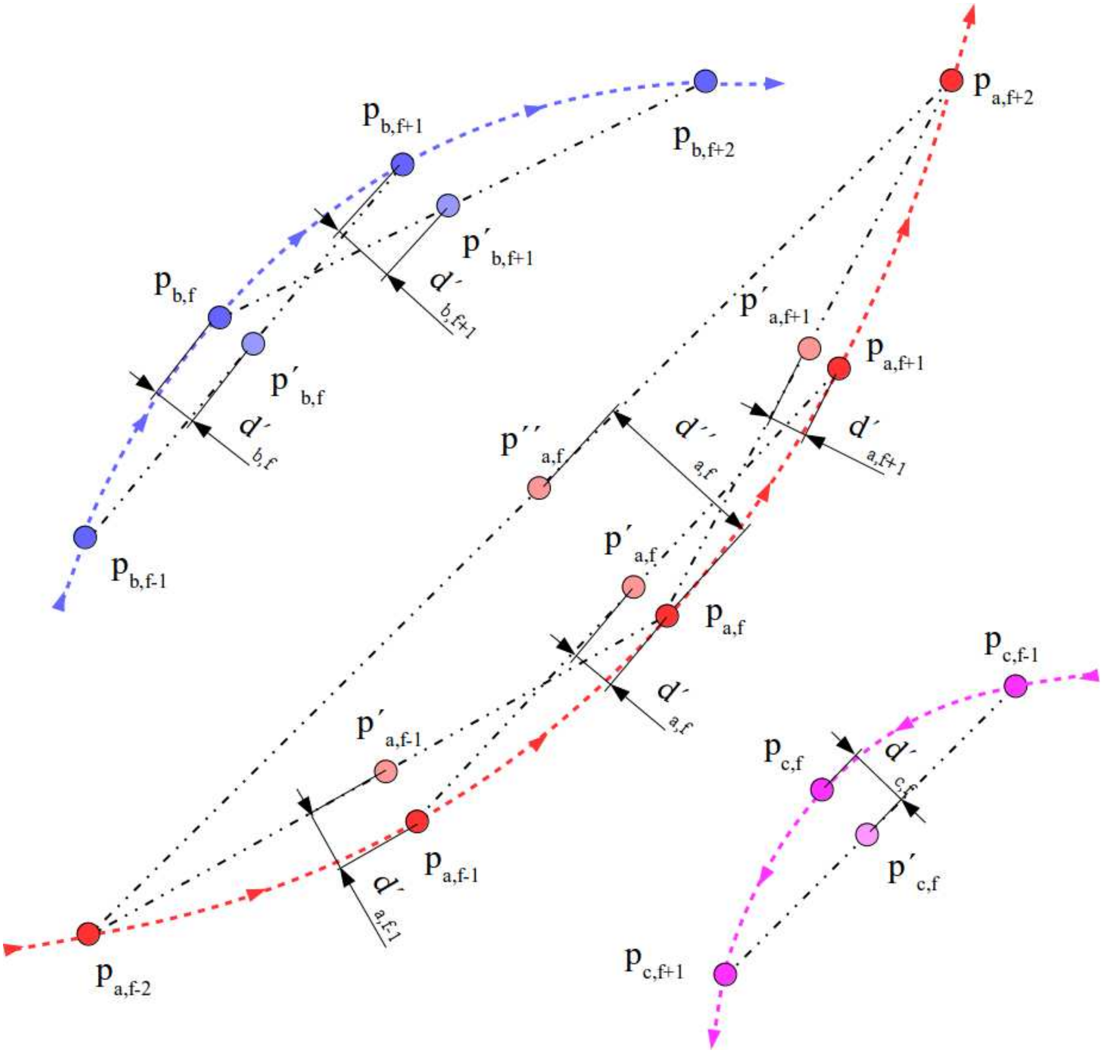


Figure 6.4: Successful robust initializations using five, four and three frames

gorithm starts with the calculation of the initialization decision matrices using five consecutive frames and tries to initialize tracking using the most successful frame for the initialization. If that is possible, as mentioned above, bi-directional tracking is carried out, and the tracked objects are excluded from the initialization decision matrix before continuing with the next tracking pass. If the initialization using five frames is not possible, the number of frames used for the initialization is gradually decreased down to three frames, and the same tracking algorithm is rerun in multi-passes until no objects are left for tracking.

The multi-pass tracking algorithm with robust-initialization is computationally more expensive than the conventional tracking algorithm. Especially for the robust initialization using five consecutive frames, the number of combinations to consider can increase drastically. For instance, for five consecutive frames with 10^2 objects to track

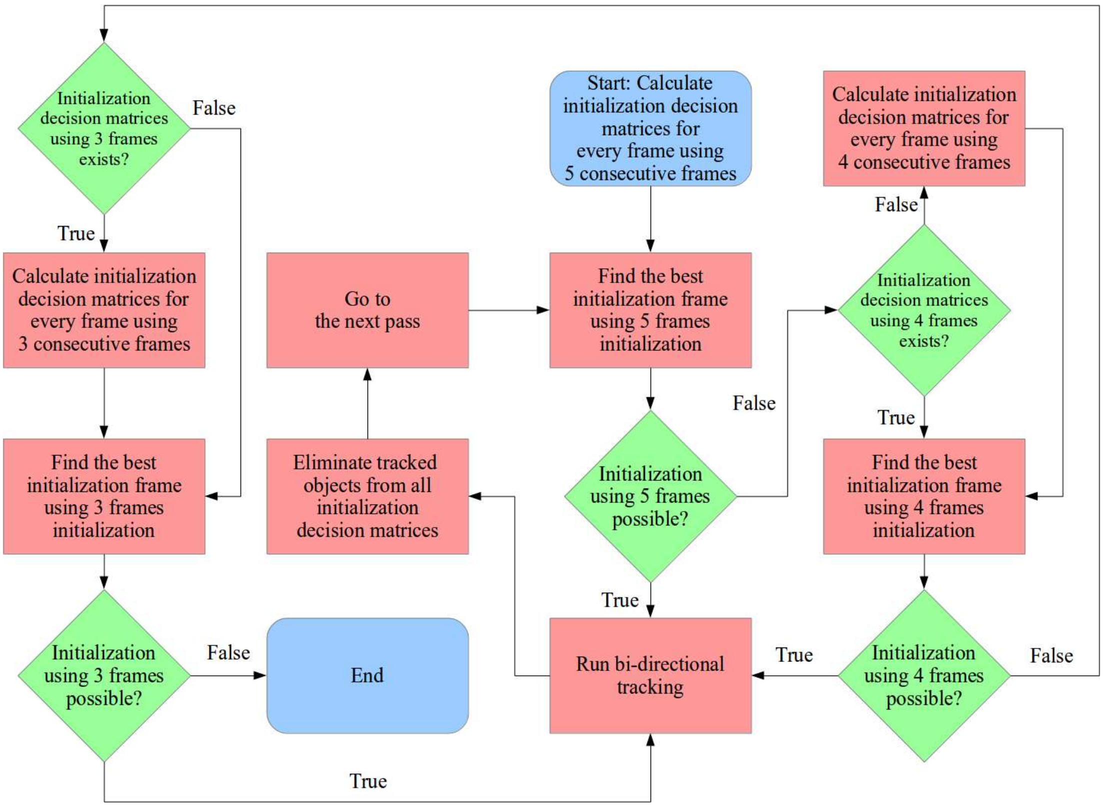

Initialization decision matrices using 3 frames exists?
False
True
Calculate initialization decision matrices for every frame using 3 consecutive frames
Find the best initialization frame using 3 frames initialization
Initialization using 3 frames possible?
False
End
True
Go to the next pass
Eliminate tracked objects from all initialization decision matrices
Start: Calculate initialization decision matrices for every frame using 5 consecutive frames
Find the best initialization frame using 5 frames initialization
Initialization using 5 frames possible?
False
True
Run bi-directional tracking
Calculate initialization decision matrices for every frame using 4 consecutive frames
False
Initialization decision matrices using 4 frames exists?
True
Find the best initialization frame using 4 frames initialization
Initialization using 4 frames possible?
True
False

Figure 6.5: Flowchart of the multi-pass tracking algorithm with robust initialization

in each frame, the number of total initialization combinations to consider would be $(10^2)^5 = 10^{10}$. The robust initialization algorithm reduces this problem by using four combinations with three frames in each to $4 \cdot (10^2)^3 = 4 \cdot 10^6$. However, the number of combinations can be still reduced by filtering unnecessary combinations. This filtering is implemented in the robust initialization algorithm using the following six filters:

- *Maximum Area Change Ratio:* Ratio between change in object area between two frames.
- *Maximum Distance:* Maximum distance between calculated and actual mid-points in pixels.
- *Maximum Displacement Ratio:* The maximum ratio between the average object displacement and the object displacement in the last two frames.
- *Maximum Direction Change:* Maximum direction change of the objects in degrees in the first two and the last two frames of a sequence with three frames.
- *Minimum Displacement:* Minimum displacement of the objects in two consecutive frames in pixels.
- *Maximum Displacement:* Maximum displacement of the objects in two consecutive frames in pixels.

All these filters are used gradually during the calculation of the initialization decision matrices so that the size of these matrices would not grow unnecessarily uncontrolled.

6.4 SIL Simulations with Mathematically Defined Tracks

The newly developed multi-pass tracking algorithm with robust initialization was first tested with 96 mathematically defined virtual tracks. For this purpose four different track types were defined: vertical, 15° slope, horizontal, and helicoidal. The vertical tracks have a bottom-up flow direction. The tracks with 15° slope deviate 15° to this vertical flow axis and flow also in the bottom-up direction. The horizontal tracks flow in the $+Z$ direction according to the global coordinate system, namely the coordinate system of the first camera. Finally, the helicoidal tracks flow bottom-up through a helix structure. The maximum velocity of these four velocity variations is determined according to the frame rate of the cameras ($5fps$) so that at least three frames of an object are visible on image planes. Starting from this maximum velocity, three further velocity variations defined by decreasing the velocity by 50% to test the tracking algorithm with different velocity levels. For every track type four velocity variations are considered: 0.05, 0.1, 0.2, and $0.4m/s$. Each of these four velocity variations is tested with a constant velocity profile and also with laminar flow profile as defined in equation 6.11. This equation defines the fully developed laminar flow profile in circular pipes in cylindrical coordinates. Maximum velocity v_{max} occurs in the center of the pipe where the radius r equals to zero. Similarly, the velocity magnitude v is zero where the radius equals to the maximum radius r_{max} on the pipe walls.

$$v(r) = v_{max}\left(1 - \frac{r^2}{{r_{max}}^2}\right) \tag{6.11}$$

Seed points are also defined using cylindrical coordinates. Here, after the camera calibration, the volume boundaries are determined in which particles can be visualized with at least two cameras. Thereafter the minimum axial coordinate and maximum radial coordinate is used to define the randomly distributed seed points. In order not to start the tracks always at the same axial coordinate, the axial coordinate is extended in both directions with maximum 20*mm* in which the starting points are defined randomly. Regarding seeding, three different seed rates are investigated: 5, 10, and 20*pps*.

These mathematically defined 96 tracks with four velocities, two velocity distributions, three seeding rates, and four track type variations are chosen in such a way so that the tracking algorithm can be tested with different complexity levels. Table 6.1 shows an overview of the virtual tracks. The last 12 virtual tracks in this table are generated through CFD simulations, which is described in detail in section 6.5.

No	**Velocity [m/s]**	**Velocity Distribution**	**Seeding Rate [pps]**	**Track Type**
1-12	0.05, 0.1, 0.2, 0.4	constant	5, 10, 20	vertical
13-24	0.05, 0.1, 0.2, 0.4	laminar flow profile	5, 10, 20	vertical
25-36	0.05, 0.1, 0.2, 0.4	constant	5, 10, 20	15° slope
37-48	0.05, 0.1, 0.2, 0.4	laminar flow profile	5, 10, 20	15° slope
49-60	0.05, 0.1, 0.2, 0.4	constant	5, 10, 20	horizontal
61-72	0.05, 0.1, 0.2, 0.4	laminar flow profile	5, 10, 20	horizontal
73-84	0.05, 0.1, 0.2, 0.4	constant	5, 10, 20	helicoidal
85-96	0.05, 0.1, 0.2, 0.4	laminar flow profile	5, 10, 20	helicoidal
97-108	0.05, 0.1, 0.2, 0.4	CFD generated	5, 10, 20	CFD generated

Table 6.1: Virtual track variations used in the SIL simulations

Figure 6.6 shows the world points of two variations of the mathematically defined virtual tracks, whereby individual cameras and tracks are colored with different colors. The vertical tracks on the top with low flow velocity and seed rate exhibit a simple case for object tracking. On the other hand, the helicoidal tracks on the bottom with high flow velocity and seed rate show a very complicated case for object tracking. Though such a flow structure is nearly impossible in practice, it is used during the SIL simulations to figure out the tracking capabilities of both implemented tracking algorithms.

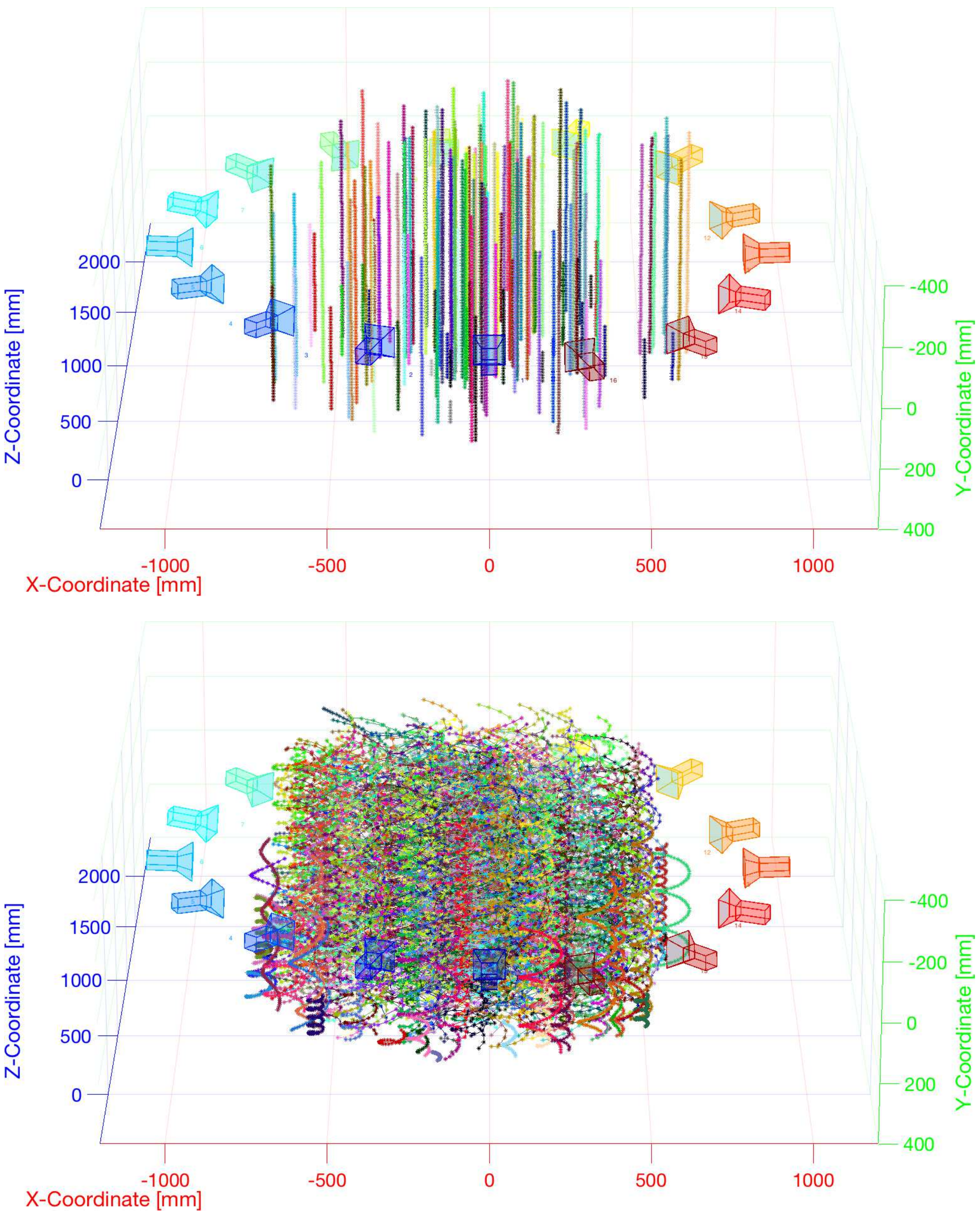


Figure 6.6: World points of vertical virtual tracks with a constant flow profile, $0.05m/s$ velocity and $5pps$ seeding (top) and helicoidal virtual tracks with a laminar flow profile, $0.4m/s$ velocity, and $20pps$ seeding

As defined in section 6.2, in the conventional tracking algorithm a search radius r_s is utilized. Due to its definition, the multi-pass tracking algorithm with robust initialization does not need such a variable though. This variable prevents the comparison of these two tracking algorithms for the same virtual tracks since the optimum search radius resulting in the highest tracking efficiency is unknown and changes for every camera with track type, flow velocity, and seed rate. In order to overcome this problem, the conventional tracking algorithm was utilized using ten different search radii for every camera in every virtual track variation. These search radii start with $5px$, and are sequentially doubled up to $2560px$, which is the first value after the biggest distance on the image sensor[17]. After the tracking with conventional tracking algorithm for all cameras in all track variations with ten different search radii was carried out, the results with the highest tracking efficiency was used.

Figure 6.7 shows the result of this study. In this graph, the mean search radii for every track are plotted in relation to the flow velocity and the total number of objects (in other words the seed rate). It can be easily seen that the mean search radii increase with the ascending flow velocities. Another outcome which can be extracted from this data set is that the higher seed rates need smaller search radii compared to the lower seed rates. Beyond the averaged values in this graph, it is important to mention that the search radii above $320px$ never delivered the highest tracking efficiency. In other words, in this study, the search radii for the conventional tracking algorithm vary between $5px$ and $320px$.

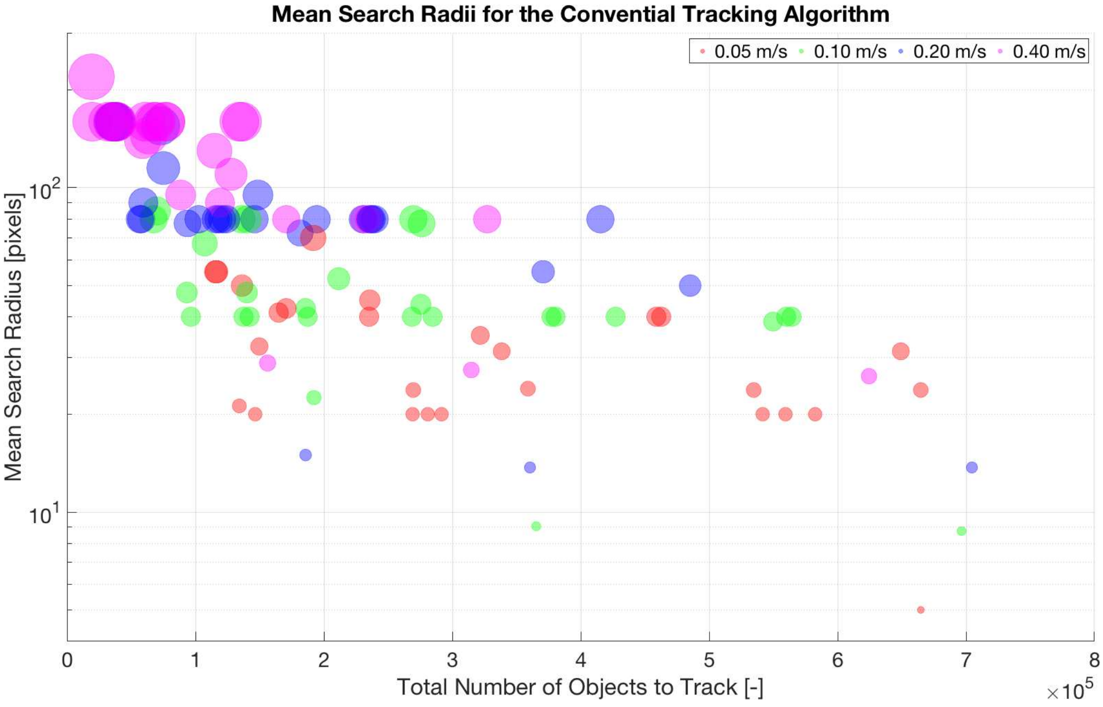


Figure 6.7: Mean search radii utilized in the conventional tracking algorithm

Figure 6.8 shows the comparison of tracking algorithms for vertical tracks with a constant flow profile. Here it can be observed that with ascending flow velocity

[17] The hypotenuse of the image sensor is $2202.9px$

the tracking efficiency increases when the multi-pass tracking algorithm with robust initialization is used. On the other hand, the tracking efficiency of the conventional tracking algorithm decreases with ascending flow velocity. Another outcome of this comparison is that with ascending seed rate, the tracking efficiency decreases if the conventional tracking algorithm is utilized. The multi-pass tracking algorithm with robust initialization prevents this, as the tracking efficiency stays constant or increases with the ascending seed rate. For the vertical tracks with constant velocity profile, the multi-pass tracking algorithm with robust initialization delivers considerably higher tracking efficiencies compared to the conventional tracking algorithm. The difference between these two algorithms increases with the ascending seed rate and decreases with the descending flow velocity. For the tracks with a velocity of $0.5m/s$ and seed rate of $5pps$, the conventional tracking algorithm delivers a slightly higher tracking efficiency.

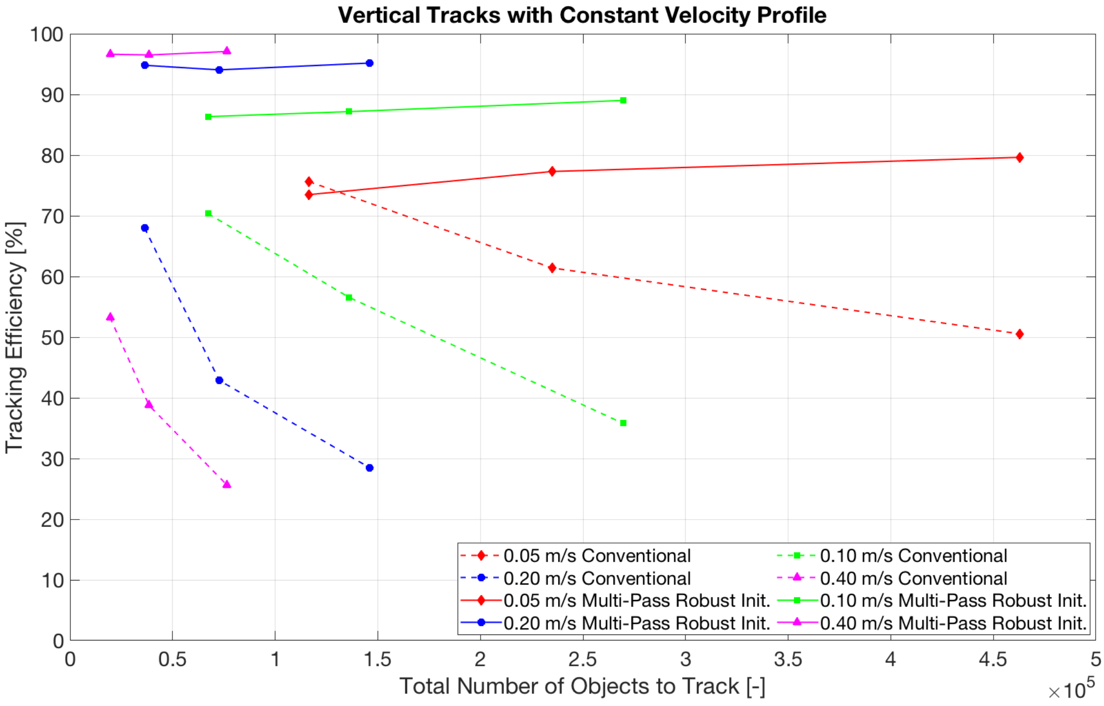


Figure 6.8: Comparison of the tracking algorithms for the vertical tracks with a constant flow profile

Figure 6.9 shows that the tracking efficiencies have similar trends if instead of a constant flow profile, a laminar flow profile is used for the vertical tracks. However, the tracking efficiency of the multi-pass tracking algorithm with robust initialization decreases when the laminar flow profile is used. On the other hand, the tracking efficiency of the conventional tracking algorithm increases with the laminar flow, especially for the lower flow velocities. These characteristics decrease the differences between the tracking efficiencies of the tracking algorithms. In vertical tracks with a laminar velocity profile, in 3 of 12 variations, conventional tracking algorithm delivers better results, especially with low velocities and low seed rates.

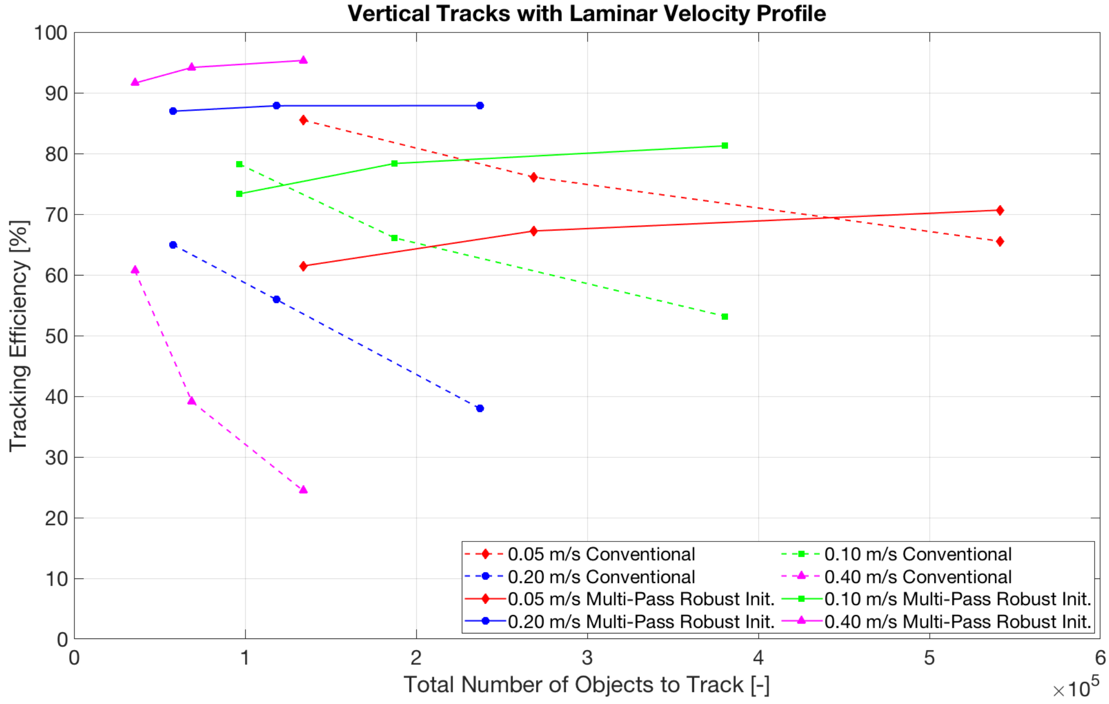


Figure 6.9: Comparison of the tracking algorithms for vertical flow with a laminar flow profile

Figure 6.10 shows the comparison of the two tracking algorithms for the tracks with 15° slope and constant flow profile. The results delivered by the conventional tracking algorithm are quite similar to the results of the vertical flow with a constant flow profile. However, for the multi-pass tracking algorithm with robust initialization, the relationship between the flow velocity and the tracking efficiency changes. Here, the tracking efficiency increases with the ascending flow velocity till $0.20 m/s$ but then drops suddenly for the tracks with a flow velocity of $0.40 m/s$. If the two tracking algorithms are compared for 15° sloped flow with a constant flow profile, the multi-pass tracking algorithm with robust initialization delivers for all variations better results than the conventional tracking algorithm.

Figure 6.11 shows the resulting tracking efficiencies for 15° sloped tracks with a laminar flow profile. The utilization of a laminar flow profile in 15° sloped flow delivered a similar change as in vertical flow, namely the tracking efficiency of the conventional tracking algorithm increases and the efficiency of the multi-pass tracking algorithm with robust initialization decreases. However, this change is not as high as with the vertical flow, so that only in 1 of 12 variations the conventional tracking algorithm

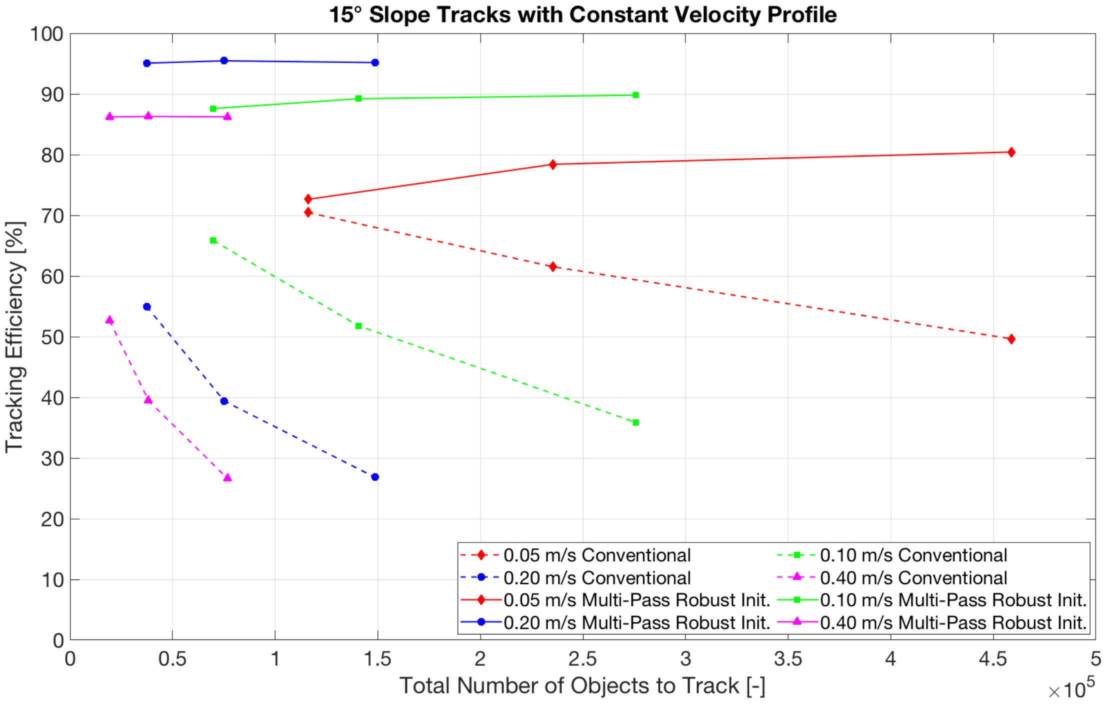


Figure 6.10: Comparison of the tracking algorithms for 15° sloped flow with a constant flow profile

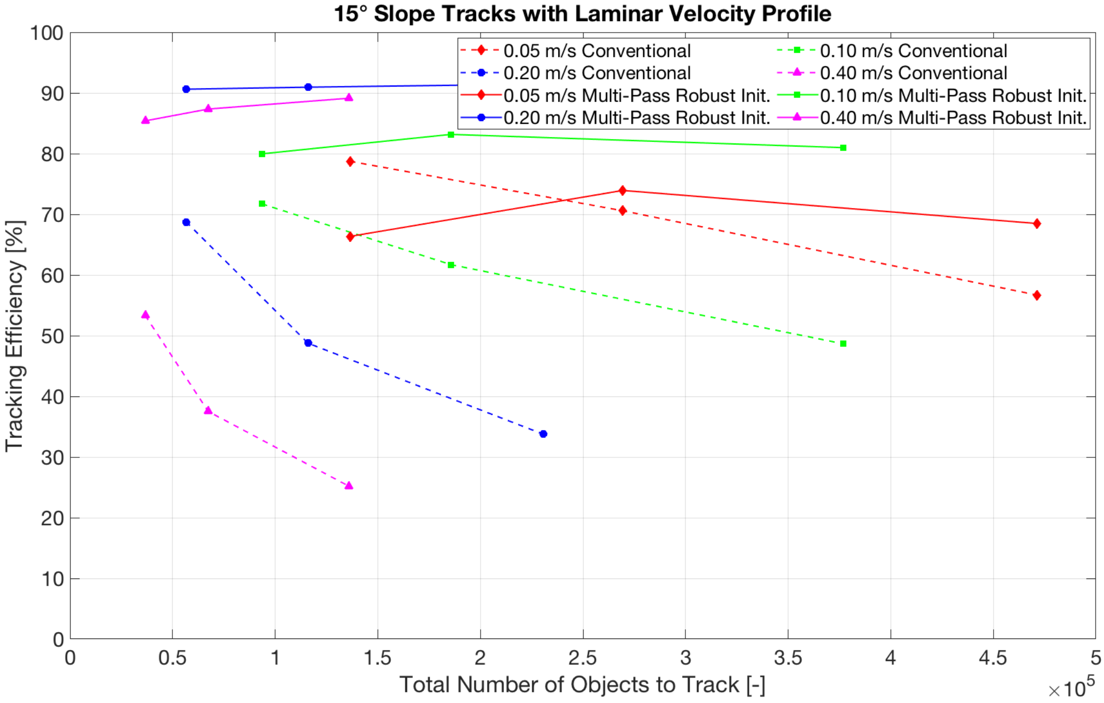


Figure 6.11: Comparison of the tracking algorithms for 15° sloped flow with a laminar flow profile

delivers better results than the multi-pass tracking algorithm with robust initialization.

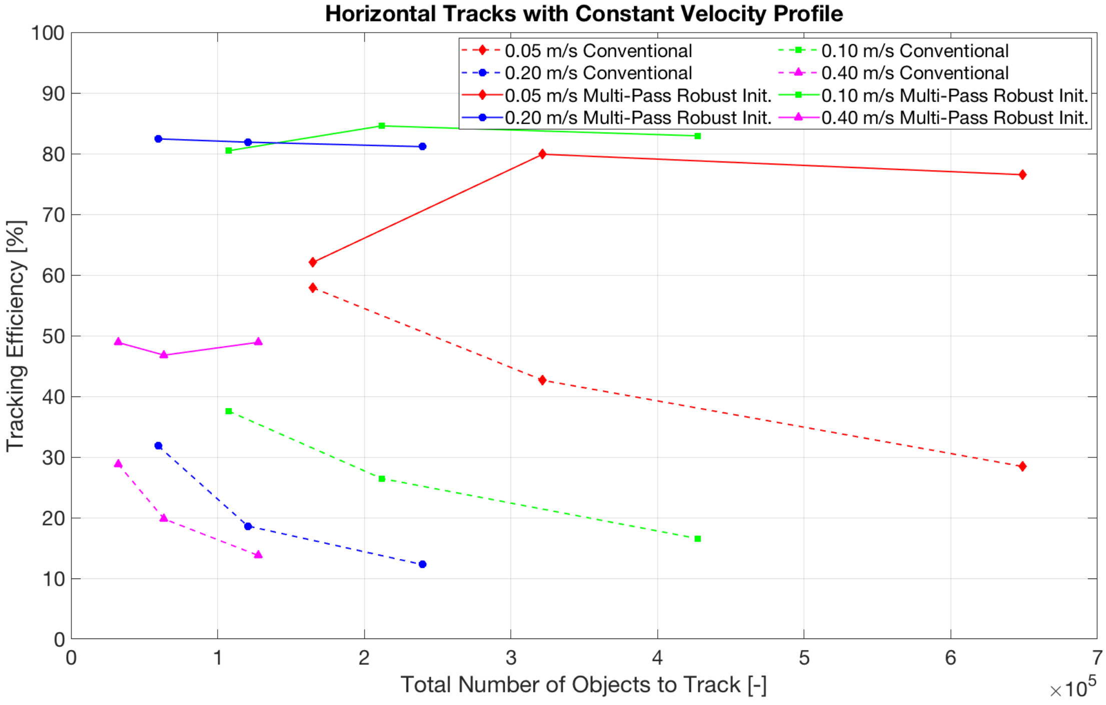


Figure 6.12: Comparison of the tracking algorithms for horizontal flow with a constant flow profile

In figure 6.12 the results for horizontal flow with a constant flow profile are represented. Compared to vertical and 15° sloped flows with a constant flow profile, the conventional tracking algorithm delivers for horizontal flow with a constant flow profile lower tracking efficiencies. However, their trends are mostly independent of flow velocity and seed rate. The tracking efficiencies with the multi-pass tracking algorithm with robust initialization also decreases for the horizontal flow with constant flow profile. Here the tracking efficiencies stay nearly constant or increase slightly with the ascending number of objects to track. However, the tracking efficiencies for the two higher velocities ($0.40 m/s$ and $0.20 m/s$) decrease significantly. Overall, for the horizontal flow with a constant flow profile, the multi-pass tracking algorithm with robust initialization delivers higher tracking efficiencies than the conventional tracking algorithm for the all variations.

As shown in figure 6.13 , other than the vertical and 15° sloped flows, for the horizontal flow the tracking efficiency of the conventional tracking algorithm decreases if a laminar flow profile is utilized instead of constant one. For a horizontal flow with a laminar flow profile, the multi-pass tracking algorithm with robust initialization exhibits a similar character as for the 15° sloped flows with laminar flow profile, whereby the tracking efficiency is decreased especially for the variations with the highest flow velocity of $0.40 m/s$. Generally, the multi-pass tracking algorithm with robust initial-

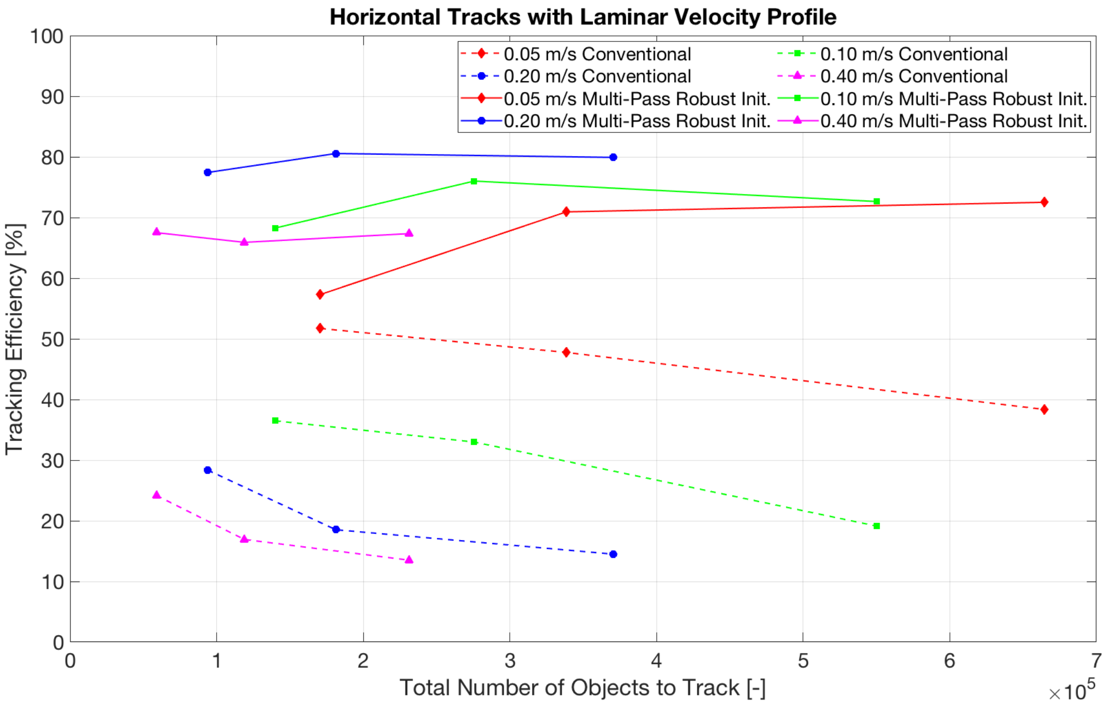


Figure 6.13: Comparison of the tracking algorithms for horizontal flow with a laminar flow profile

ization delivers for all variations of the horizontal flow with constant and laminar flow profiles better results than the conventional tracking algorithm.

Figure 6.14 shows the tracking efficiencies for the helicoidal flow with a constant flow profile. This flow structure along with its laminar flow profile variation exhibits the most complex virtual tracks observed in this study. Compared to the previously presented tracks, the helicoidal tracks with constant flow profile show significantly lower tracking efficiencies. Here, the tracking efficiencies stay below 50% which was not observed in the previously presented tracks. It still can be seen that the tracking efficiency decreases with the ascending seed rate and flow velocity for the conventional tracking algorithm. For the multi-pass tracking algorithm with robust initialization, such a relationship between these parameters cannot be observed. However, other than the previously discussed tracks, here the lowest and highest velocities ($0.05m/s$ and $0.4m/s$) exhibit the highest tracking efficiencies. Among the 12 variations for only one variation the conventional tracking algorithm has a higher tracking efficiency.

The results for the last tracks type along the mathematically defined tracks, helicoidal flow with a laminar flow profile, are shown in figure 6.15. Here, the results with both tracking algorithms are slightly better than the results with a constant flow profile, whereby the conventional tracking algorithm still delivers only for one of twelve vari-

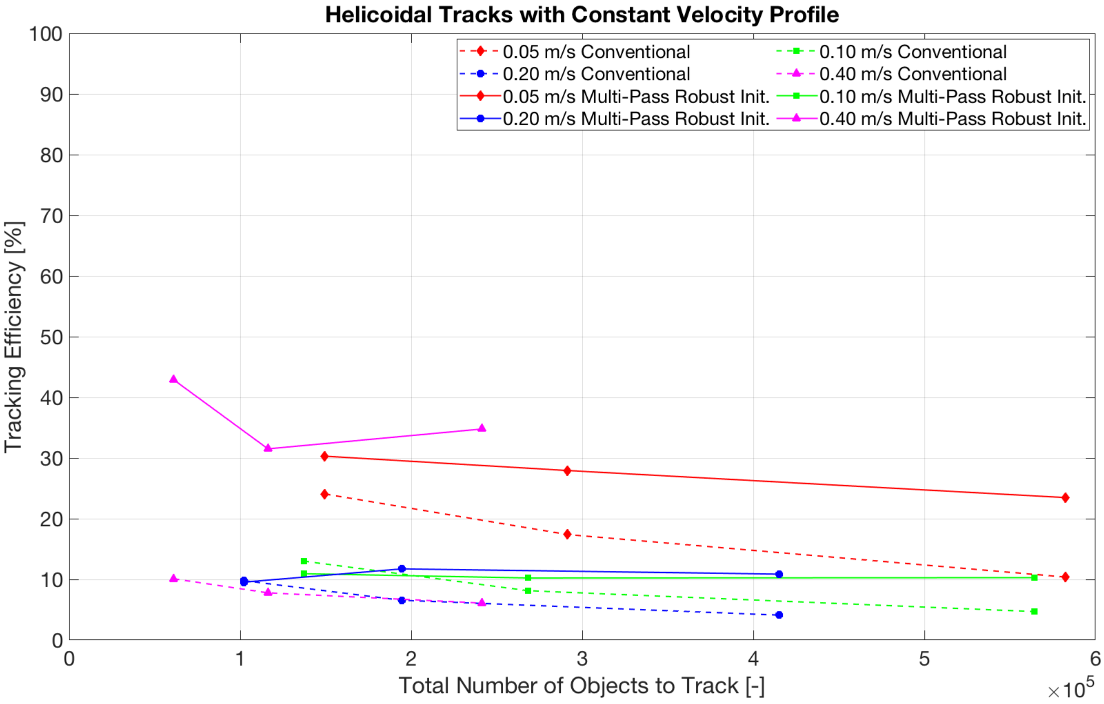


Figure 6.14: Comparison of the tracking algorithms for helicoidal flows with a constant flow profile

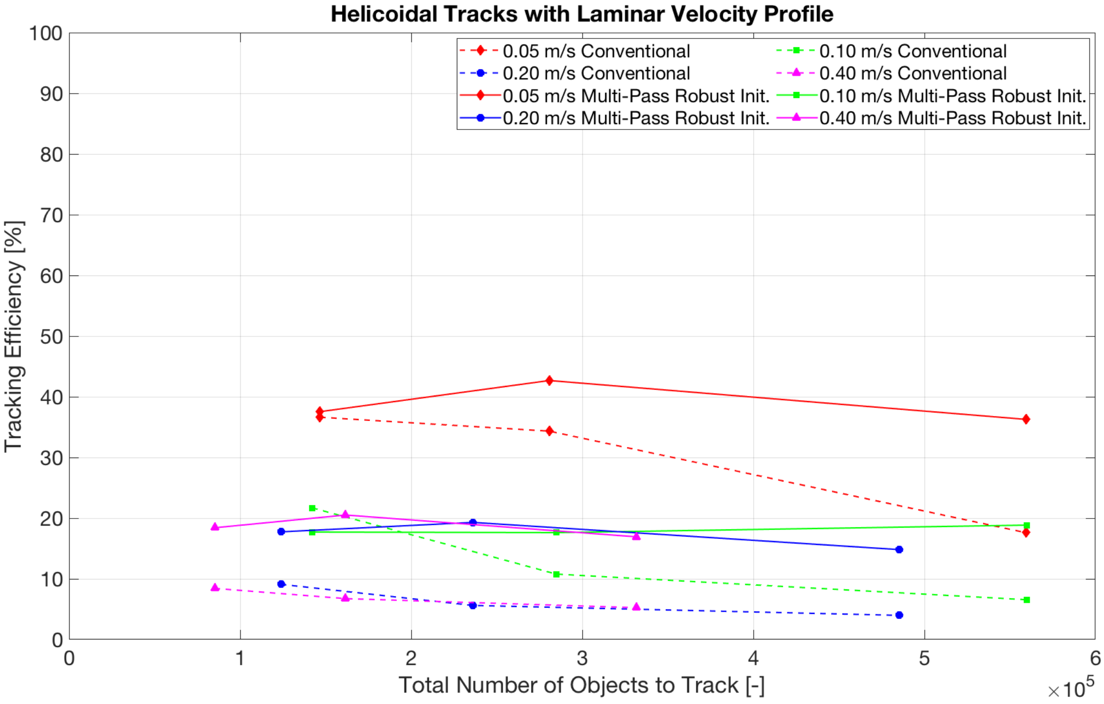


Figure 6.15: Comparison of the tracking algorithms for helicoidal flows with a laminar flow profile

ations better results than the multi-pass tracking algorithm with robust initialization. One big difference compared to helicoidal flow with a constant flow profile, is that for the laminar profile the tracking efficiencies for the highest velocity $0.4m/s$ drop significantly.

The comparison of the conventional tracking algorithm and multi-pass tracking algorithm with the mathematically defined tracks shows that the multi-pass tracking delivers significantly better results in general. Only in 7 of 96 observed variations, the tracking efficiency of the conventional tracking algorithm is higher. Another positive aspect observed with the multi-pass tracking algorithm with robust initialization is that the tracking efficiency stays mostly constant or increases slightly with ascending seed rate. For the conventional tracking algorithm on the other hand, with increasing seed rate the tracking efficiency drops up to 40%. Considering the flow velocity for the conventional tracking algorithm the tracking efficiency reaches its highest values for higher velocity, which is the opposite for the multi-pass tracking algorithm with robust initialization.

6.5 SIL-Simulations with Tracks Generated by CFD Simulations

CFD simulations for track generation were carried out with the CFD code STAR-CCM+ v13.04.010. The geometry used for the CFD simulations is the geometry of the 3D-MVPTV test stand described in section 4.1. An unstructured numerical mesh for this geometry is generated using polyhedral cells with prism layers. The maximum cell size of this mesh is $5cm$, whereby the surface size varies between $1.25cm$ and $5cm$. Three prism layers with a total height of $1cm$ are used, in which the growth ratio between the layers is 20%. The resulting mesh with these parameters consists of 121993 cells, and is shown in figure 6.16.

The CFD simulations are carried out isothermally using constant density. The realizable k-epsilon turbulence model is utilized for modeling the turbulence, whereas the wall treatment is calculated using all y+ wall function. The Euler-Lagrangian approach is used to model the tracer particles in the air flow. This approach makes it possible to define individual size distributions and seeding behaviors for the tracer particles. Both Lagrangian and Eulerian phases in CFD simulations are modeled as air. In the later phases, different materials can also be used as Langragian particles (i.e., helium-filled soap bubbles), so that the influence of the tracer particles on the accuracy of the measurement can be quantified. Another advantage of this approach is its ability to model detailed particle behavior including particle-particle and particle-surface interactions. On the other hand, the interaction between the two phases is one-way coupled. That means the Eulerian phase influences the Lagrangian phase but not the other way around. The particle size distribution if defined using a log-normal distribution as shown in section 4.2.1. The tracer particles in this distribution have a minimum size of $1mm$, a maximum size of $3mm$ and an average size of $2mm$. A resulting size distribution according to this definition is shown in figure 6.17.

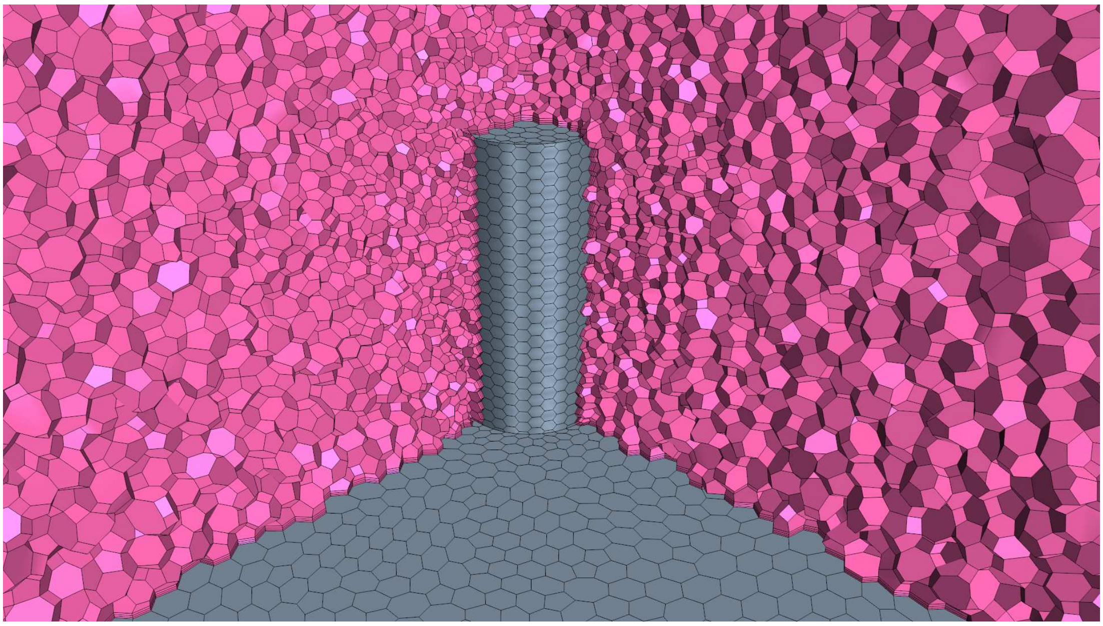


Figure 6.16: Details of the numerical mesh used for the CFD simulations

The CFD simulations for track generation were carried out transiently. In order to obtain an initial flow field before the seeding, the unsteady solver and the Lagrangian multiphase solver were frozen at the beginning of the simulations. A maximum of 500 iterations were calculated under this setup so that the flow field reached a quasi-steady state which is evaluated according to the fluctuations of the turbulent kinetic energy. After a quasi-steady state is achieved, the transient CFD simulation with Lagrangian particles is run for $30s$. Lagrangian particles are seeded from a horizontal plane with a diameter of circa $80cm$, just below the top surface of the cylinder. Their initial velocity is $0m/s$ so that they can quickly reach the flow velocity and represent the flow field.

Figure 6.18 shows the result of a CFD simulation after $30s$ with an inlet velocity of $0.4m/s$ and seeding of $20pps$. The horizontal section plane in this figure is located at the half height of the inlet to show the flow entering into the domain. The vertical section plane is located in the middle of the room to demonstrate the flow field around the cylinder. It can be seen that the seeded particles follow the airflow above the

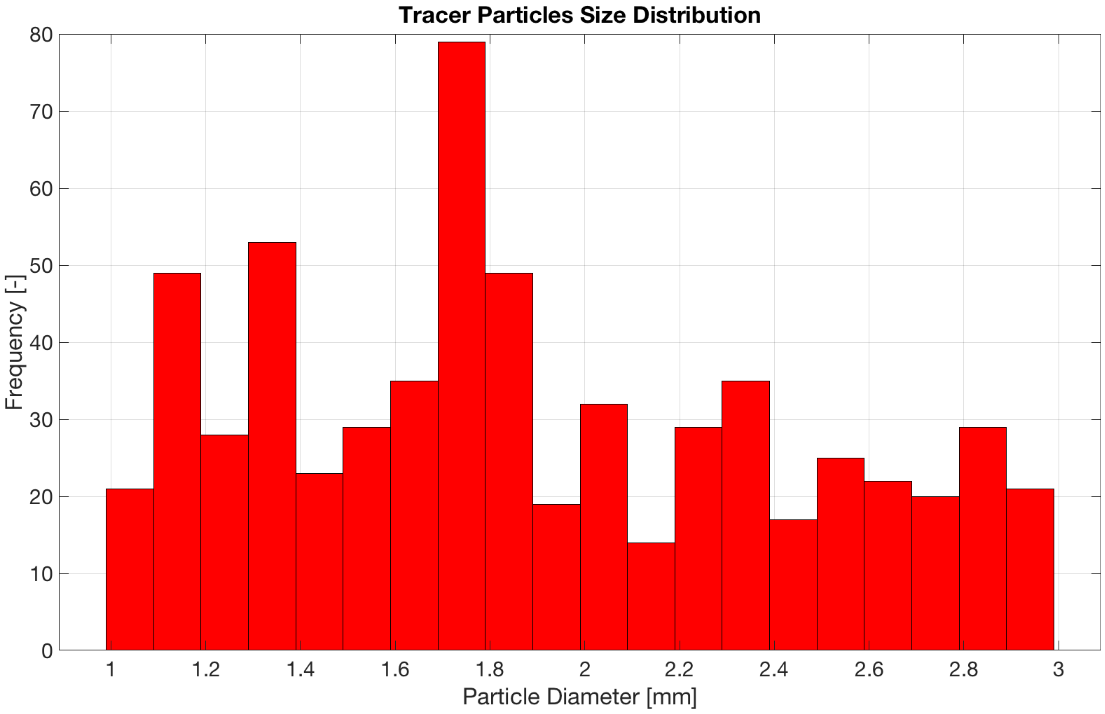


Figure 6.17: Size distribution of the Lagrangian particles for seeding with $20pps$ and an inlet velocity of $0.4m/s$

cylinder, in which the impulse from the inlet leads them to the opposite wall. Once Lagrangian tracks are simulated using CFD, they should be transferred to the 3D-MVPTV-SIL-Environment for further steps. For this purpose, a MATLAB function is developed, which converts the Lagrangian particle tracks to the virtual tracks.

Figure 6.19 shows the results of the object tracking for the tracks generated by CFD simulations. Here, the tracking efficiency for the multi-pass tracking algorithm with robust initialization mostly increases with the ascending flow velocity as for the tracks with the vertical flow. However, it decreases with ascending seed rate. For the conventional tracking algorithm, the tracking efficiency also mostly decreases with ascending seed rate. There is not an exact dependency between the flow velocity and the tracking efficiency for this case. The only significant outcome is that the conventional tracking algorithm delivers very low tracking efficiencies for the highest velocity ($0.40m/s$). Generally, the conventional tracking algorithm has lower tracking efficiencies in 11 of 12 track variations than the multi-pass tracking algorithm with robust initialization.

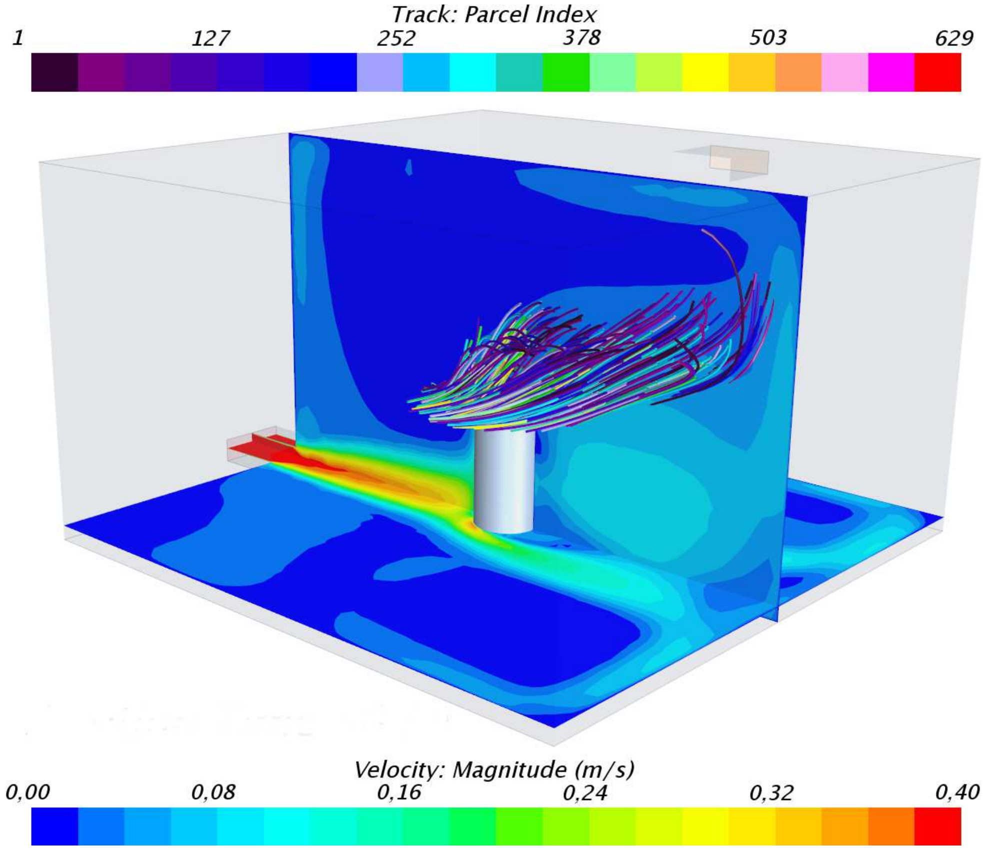


Figure 6.18: Flow field and tracks of Lagrangian particles for seeding with 20pps and an inlet velocity of 0.4m/s

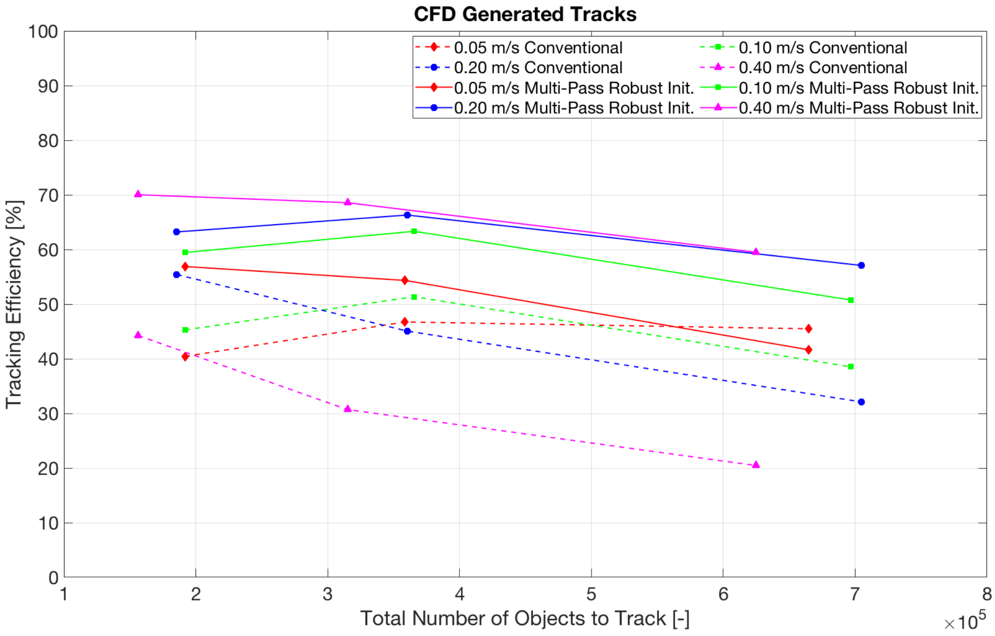


Figure 6.19: Comparison of the tracking algorithms for the tracks generated by CFD

7 3D-MVPTV Measurements

After the development of the 3D-MVPTV with focus on camera calibration and tracking algorithm, as a last step 3D-MVPTV measurements are compared with measurements carried out using hot-sphere probes. Since the hot-sphere probes are just capable of measuring the velocity magnitude, the aim of these measurements is to ensure that the 3D-MVPTV delivers plausible velocity magnitudes compared to a conventional measurement method. Both measurements are carried out in the 3D-MVPTV test stand described in section 4.1. Following the comparison of both measurements, section 7.3 discusses the reasons for the deviations between these two measurement methods.

7.1 3D-MVPTV Measurements

The 3D-MVPTV measurements were undertaken for three different inlet velocities: $0.47 m/s$, $0.24 m/s$, and $0.14 m/s$. The first velocity represents the maximum velocity which can be achieved by the axial fans installed in the inlet duct. This velocity was then reduced in two steps to test the developed flow measurement system with lower velocities. The control of the inlet velocity occurs manually using a dimmer unit and a hot-sphere probe which is positioned just after the inlet duct as shown in figure 7.2 right. The tracer particles were seeded in the 3D-MVPTV test stand using four diffusers as shown in figure 4.1. Before the webcams captured these particles, a time interval of five minutes was let pass so that the flow regime could reach a quasi-stationary state and tracer particles could fill the observed volume. Then, frame synchronized video capture was carried out with 16 cameras for $30 s$, and further steps in the 3D-MVPTV workflow like object recognition, object tracking, track matching, and multi-view triangulation were carried out. Figure 7.1 shows the resulting tracks after multi-view triangulation for the measurement with the velocity $0.14 m/s$. In this figure, every individual track has a different color. Finally, in the post-processing step, the three dimensional velocity components and the resulting velocity magnitudes were calculated using equations 3.4 and 7.1, which can then be compared with the measurements carried out using a conventional method in the next step.

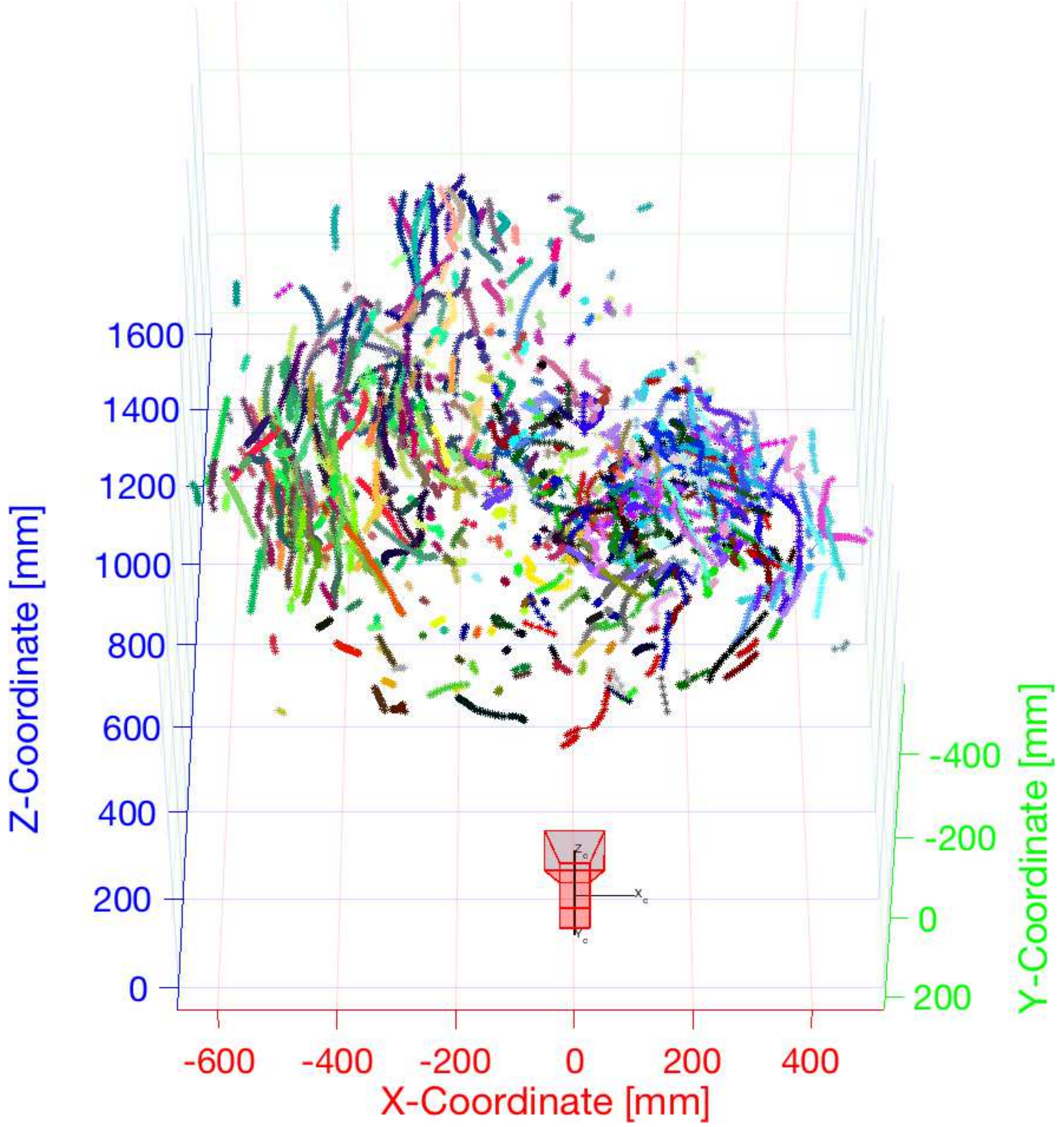


Figure 7.1: World points of the tracks for the measurement with the velocity $0.14 m/s$

$$v = \sqrt{{v_x}^2 + {v_y}^2 + {v_z}^2} \tag{7.1}$$

7.2 Comparison Measurements

The comparison measurements were carried out using four available Dantec Dynamics 54T33 hot-sphere probes in the HRI. These hot-sphere probes have an accuracy of $\pm 2\%$ of reading $\pm 0.02m/s$ between $0-1m/s$ and $\pm 5\%$ of reading between $1-5m/s$ [13]. Each probe has two quartz spheres with a diameter of $3mm$. One of these spheres is held at a constant over-temperature as shown in figure 7.2 on the left. The energy consumed to keep the hot-sphere at a constant temperature is then correlated with the air velocity magnitude. The acceptance angle of the hot-sphere relative to the probe axis is $\pm 160^\circ$ between $0-1m/s$, and it varies from $+50^\circ$ to $+130^\circ$ for the velocity range $1-5m/s$ [13].

The positions of the hot-sphere probes in the 3D-MVPTV test stand are needed in order to compare their results with the 3D-MVPTV measurements. In order to get the position of these probes using the same coordinate system as in the 3D-MVPTV measurements and with similar accuracy, multi-view triangulation was utilized. Here, images of the hot-sphere probes with webcams were captured during the comparison measurements. Thereafter the positions of the hot spheres on the probes were marked manually on these images as shown in figure 7.2 on the right. In this figure the hot sphere on the bottom is marked with red, the one in the middle with green and the one on the top with blue. Finally, multi-view triangulation was carried out with these image points from all cameras to calculate the positions of the hot-sphere probes.

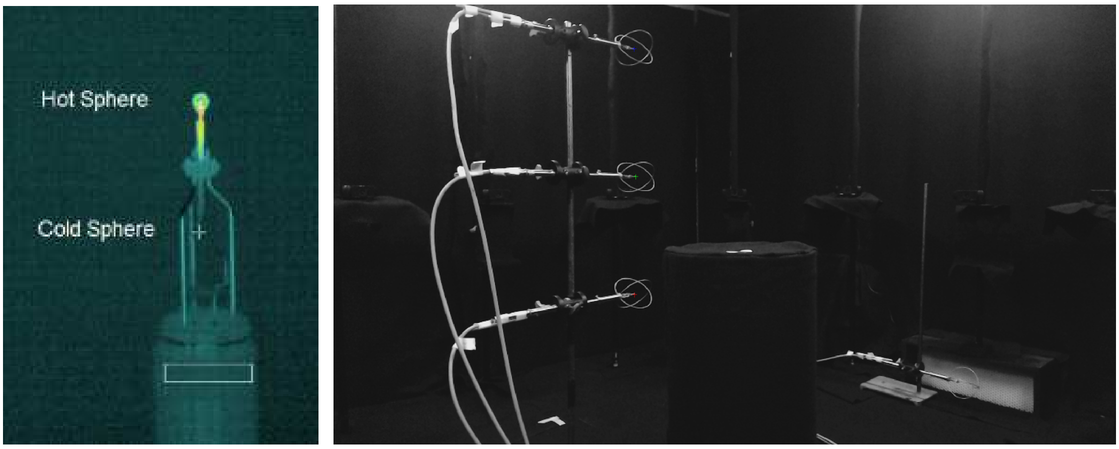


Figure 7.2: Thermography image of an hot-sphere probe [30], the positioning of the hot-sphere probes

It was not possible to capture the flow field to be observed simultaneously using three hot-sphere probes[18] and 3D-MVPTV. Therefore, eight positions in tangential direction and two positions in radial directions were chosen resulting in a total of 16 positions (see figure 7.3 left). The probe stand with three hot-sphere probes positioned in three different vertical positions (see figure 7.3 right) was moved consecutively on these 16 positions resulting 48 probe points in total. The locations of these probe points were chosen to achieve an equidistant distribution of the probes in the observed flow field. During comparison measurements, after each time the probe stand carrying hot-sphere probes moved to a new position, a 60s pause before the data acquisition was utilized so that the disruption of the flow field could be minimized. Following this 60s break, velocity magnitudes from all four hot-sphere probes are captured for 60s.

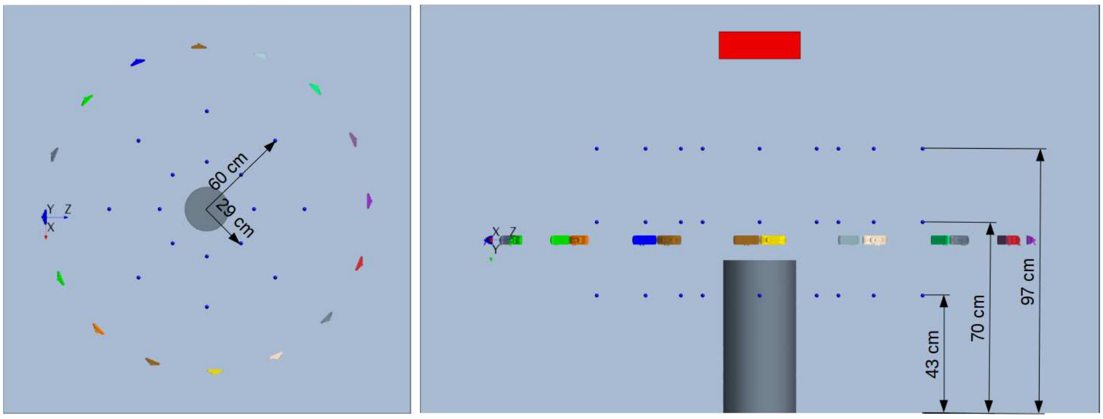


Figure 7.3: Positions of the hot-sphere probes in the 3D-MVPTV test stand

While during the comparison measurements the hot-sphere probes could be positioned at any desired position easily, capturing the whole flow field using these probes is very expensive or time-consuming due to the number of probes or measuring steps required. On the other hand, the 3D-MVPTV can measure the entire flow field simultaneously; it can deliver not just velocity magnitudes but also the velocity vectors. However, homogeneous distribution of the tracer particles under various flow conditions can be quite challenging. That makes it nearly impossible to find a 3D-MVPTV data point which has the same coordinate as one of the hot-sphere probes. In order to carry out a plausible comparison between these measurement methods, the 3D-MVPTV data points should be filtered according to their distance to the nearest hot-sphere probe. After this filtering the 3D-MVPTV data points corresponding to a hot-sphere sensor were averaged.

Figure 7.4 represents the results of such a comparison. In these graphs, the blue bars represent the difference between the 3D-MVPTV and comparison measurements for

[18] One of the four hot-sphere probes was used to control the inlet velocity as described in section 7.1.

the data points, which are annotated on vertical axes. The horizontal axes show the distance between data points of 3D-MVPTV and comparison measurements in ascending order. These diagrams do not exhibit any dependency between these data point distances and the velocity difference between the two measurement methods. The data limits plotted with orange represent the accuracy range of the hot-sphere probes used. It can be easily seen that the deviations between the measurements increase and leave this accuracy range with increasing flow velocity. This increase exists most probably due to the decreasing accuracy of the hot-sphere probes with ascending flow velocity.

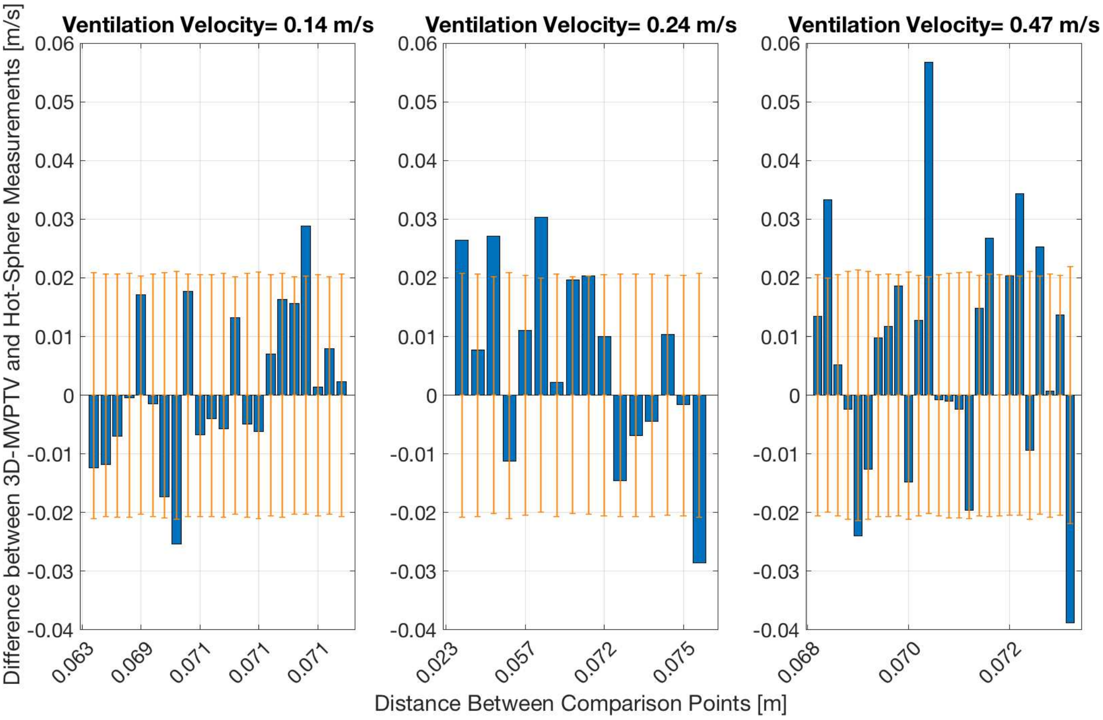


Figure 7.4: Velocity differences between MVPTV and hot-sphere probe measurements, according to the distance between the data points

Figure 7.5 shows a further comparison of these two measurement methods. Here the hot-sphere results in the same height were averaged as the bottom, middle and top. Similarly, the 3D-MVPTV results at the same height as the hot-sphere probes are clustered with tolerance and averaged. The comparison of all three velocity levels shows that the differences of the averaged velocities between the two measurements for the middle data points are the lowest. For the bottom data points, hot-sphere measurements exhibit an increasing velocity trend parallel to the rising ventilation velocity. This result is expected since the bottom data points are the closest data points to the ventilation duct. However, such a trend can not be observed with the 3D-MVPTV measurements. That can be explained with the heterogeneous distribution of the data points (namely the tracer particles) and the tolerance used during clustering of the results explained above. When the velocity trends of the three heights are considered for each measurement, for the hot-probe measurement, the velocities first rise and then decrease. It is not possible to obtain a similar relationship for the 3D-MVPTV measurements for the same observation. However, for a better comparison, it would be easier to use the results of the CFD simulations for the SIL simulations with similar velocity levels. Their results are plotted in figure 7.5 with red lines. They exhibit for all velocity levels a descending trend, which is plausible due to the inlet duct position on the bottom. If we compare the measurements with the CFD results, the only measurement with the similar character is the highest velocity $0.47 m/s$.

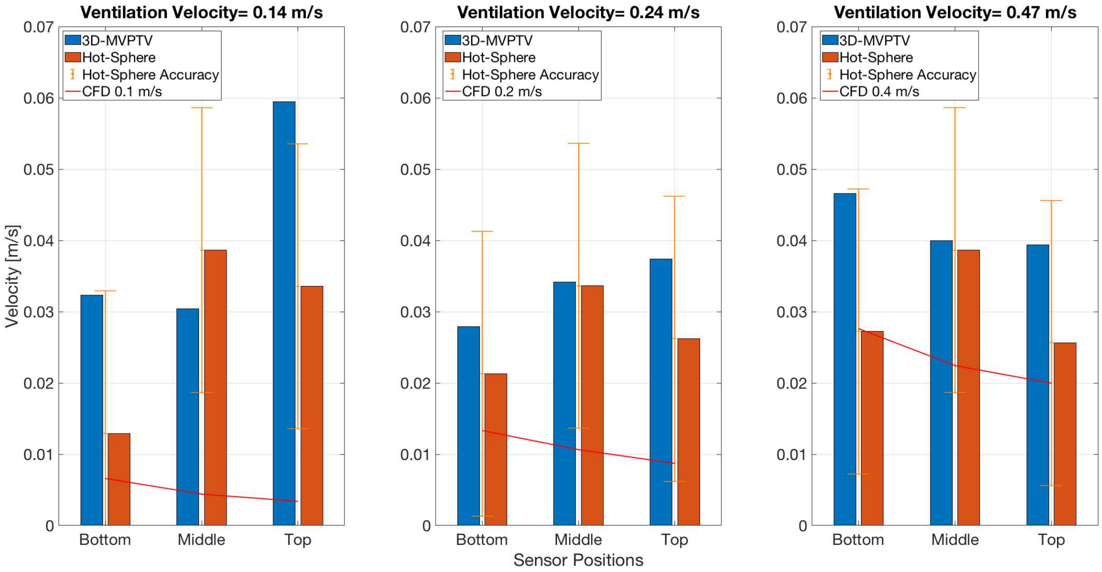


Figure 7.5: Comparison of average velocities between hot-sphere probe and MVPTV measurements

7.3 Analysis of Experimental Errors

The differences between the results of the two measurements methods represented above result from systematic and random errors of each measurement method. Below the common error sources of these differences for each measurement method are presented and discussed.

7.3.1 Common Sources of Errors in the Hot-Sphere Anemometer Measurements

- *The Accuracy of the Probes:* The hot-sphere probes used in this study have an accuracy of $\pm 2\%$ of reading $\pm 0.02 m/s$ and an acceptance angle relative to the probe axis of $\pm 160^\circ$ for the observed velocity range $0 - 1 m/s$ according to manufacturer's data sheet [13]. The probes accuracy is considered during the comparison of the two measurement methods, and the 3D-MVPTV measurements stay mostly within this error range. The acceptance angle range of 320° covers 89% of the upstream flow zone relative to the probe axis. Since the observed flow phenomena in this study exhibit a 3D structure which is not easy to pre-estimate, positioning the hot-sphere probes within the $\pm 160^\circ$ acceptance angle range relative to the probe axis is mostly not possible. That results in random errors which becomes more dominant when the flow field is captured through several discrete measurements as described in the next point.

- *Discrete Acquisition of Velocity Magnitudes:* As described in section 7.2, the hot-sphere anemometer measurements were carried out with four probes. One of these probes was reserved for the control of the axial fans in the inlet duct. The remaining three probes were positioned vertically and using these probes discrete acquisition of velocity magnitudes for 16 measurement locations were undertaken. Even though the positions of probes were recalculated after every repositioning and sufficient time after every repositioning was left to eliminate the disturbances, the discrete acquisition of the velocity magnitudes leads to random errors. It is possible to estimate these random errors, through the simultaneous acquisition of velocity magnitudes using more probes.

- *Calculation of the Probe Positions:* The positions of the probes were calculated using the following steps of the 3D-MVPTV as described in section 7.2: camera calibration, frame synchronized video capture, object recognition, and multi-view triangulation. The common error sources of these steps which are represented in section 7.3.2 in detail are also responsible for the errors during the calculation of the probe positions. However, two of these four steps differ from their versions in the 3D-MVPTV workflow. Instead of frame synchronized videos as for the tracer particles, it is sufficient to capture images to determine the positions of the probes. Since the observed objects are not moving and just their images are captured, the errors in the frame synchronized video capture are not relevant in this context. The second difference occurs in the object recognition step. Due to the contrast levels in the captured images as shown in figure 7.3, it is difficult to carry out an automatic recognition of the image points of the hot-spheres. Therefore, this step was carried out manually, which represents another source for random errors.

7.3.2 Common Sources of Errors in the 3D-MVPTV Measurements

- *Camera Calibration:* The calibration of the 16 webcams utilized in this study, were carried out iteratively using bundle adjustment as defined in section 3.2.3. The mean reprojection error was reduced until circa $3px$ with image filtering. However, this iterative calibration process with the mean projection error as a cost function cannot deliver an exact mathematical definition of the camera system used. Therefore, the further steps in which the calibration results are used are affected by this finite mean projection error.

- *Frame Synchronized Video Capture:* To calculate the world points of a tracer particle at a specific moment in time, first the image points of this tracer particle must be captured at the same moment in time by all participating cameras. If a significant time delay between the cameras for capturing the same moment occurs, this will lead to further errors in the next steps of the 3D-MVPTV workflow. The first step affected by this time delay is the object recognition. Since the tracer particles are moving, a time delay would lead to a deviation by the positions of the image points of the tracer particles. This deviation can result in erroneous object tracking especially for the faster tracer particles which are also closer to the cameras. Track matching which bases on object tracks with errors would also lead to erroneous results. Finally, during multi-view triangulation, not only the erroneously matched tracks but also the image points with deviation would deliver unreliable results.

- *Tracer Particles:* The helium-filled soap bubbles used as tracer particles in the 3D-MVPTV measurements are generated using the bubble generator as defined in section 4.1. The buoyancy neutrality of these tracer particles is controlled visually. If the helium-filled soap bubbles are sinking or rising, their buoyancy is readjusted manually through changing the proportions of helium, air, and soap utilized. This manual workflow makes it possible to achieve a qualitative global buoyancy. Since not all generated tracer particles are identical, local buoyancy neutrality deviations occur. On the other hand, as the qualitative global buoyancy is controlled manually, it is also possible to have deviations due to the competence of the person operating the bubble generator. These local and global random errors due to the manually controlled bubble generation process lead to underestimation or overestimation of velocities which are calculated through the tracer particles' motions.

- *Object Recognition:* The miscalculation of the object positions in the image space can result from the inhomogeneous illumination and erroneous object recognition. Inhomogeneous illumination of the tracer particles results in a deviation of the object centroid positions in the image space. Also, the threshold used for the conversion of $8-bit$ images to the $1-bit$ images before the object recognition, can lead to a similar deviation and also result in added noise or loss of captured tracer particles. That will lead to an error propagation to the next steps especially for the object tracking and track matching.

- *Object Tracking:* The multi-pass tracking algorithm with robust initialization developed in this study utilizes primarily the object positions in image space but also the object size. Therefore, the errors during the determination of these parameters propagate as further errors into the object tracking step. If only the tracking algorithm is considered, the tracking efficiency parameter defined in section 4.2.3 describes its accuracy. The SIL-Simulations show that the mean value of this accuracy is 80% under simple flow conditions like vertical and horizontal flows. For helicoidal flow cases, which exhibit extremely complex flow structures, the mean accuracy drops to 22%. The realistic flow cases generated with CFD simulations exhibit a mean accuracy of 59%.

- *Track Matching:* The track matching step uses the fundamental matrices, the image point coordinates, and the tracking matrices as inputs as described in section 3.2.7. Therefore, the errors in the track matching step base directly on the camera calibration, the object recognition and the object tracking steps. The errors during the calculation of the fundamental matrices, the determination of the coordinates of the image points and the tracking of the objects mislead the calculation using the epipolar constraint.

- *Multi-View Triangulation:* As one of the final steps in the 3D-MVPTV workflow, the multi-view triangulation is affected by its all preceding steps. However, the most dominant factors influencing the triangulation accuracy is the camera calibrations step and the choice of the camera set for multi-view triangulation. As discussed in section 5.3, with the same camera calibration results the minimum triangulation error varies between 0.1 and 100*mm* depending on which camera set is in use.

- *Post-processing:* In the final step of the 3D-MVPTV workflow the velocity vectors are calculated using the triangulated world points of the tracks. That means the factors affecting the accuracy of the object tracking and the multi-view triangulation also affects the calculation of the velocity vectors. However, the frame synchronized video capture step has also a significant influence on this calculation. As shown in equation 3.4 the time interval between frames Δt is also included in the calculation of the velocity vector. Therefore, if this time interval does not remain constant during the video capture, namely the video capture is not synchronized, an erroneous calculation of the velocity component is not preventable.

As mentioned in most of these points, the errors in each step are linked with the others and propagate towards to the last step. The errors in the main focus points of this study—object tracking and multi-view triangulation including camera calibration—were quantified using the SIL-Environment. In this context, the SIL-Environment makes further detailed investigations possible not only for the accuracy of each step but also for the links and error propagation between them. Such a detailed analysis and quantification of the error is also necessary for the further development of the individual steps of the 3D-MVPTV workflow.

8 Conclusion and Outlook

In this study a new flow visualization method, the 3D Multi-View Particle Tracking Velocimetry (3D-MVPTV), and a new tracking algorithm, the Multi-Pass Tracking Algorithm with Robust Initialization, were developed. With the utilization of multi-view vision, the 3D-MVPTV is capable of carrying out fluid visualization in large indoor environments even in the presence of optical obstacles. The system uses webcams which incur lower cost compared to the usually utilized scientific cameras. The workflow developed for this measurement system makes it possible to perform the individual steps of the measurements in a systematic way. In this study, a software-in-the-loop (SIL) environment is also developed, which enables detailed analysis and further improvements for this measurement system.

One of the two main focuses in this study, the camera calibration, was carried out successfully with 16 webcams. Its accuracy characterized by the mean reprojection error was too high after a conventional implementation. However, this could be reduced by 90% by implementing an image filtering step. The final mean reprojection error of this camera set with 16 cameras is with $3px$ in the lower segment of the acceptable range. A critical question related to multi-view vision and its calibration is the choice of camera set for multi-view triangulation, which was also investigated in this work using two different methods. Here the first method, choosing the camera set with elimination according to the mean reprojection error, helped to find a good camera set with less triangulation error. However, the second implemented method with precalculation, which delivers a triangulation map for the observed domain, delivered much better results. As this method uses the global average of the triangulation error, it can also be improved using local values of the triangulation error. That will enable a further reduction of the triangulation error depending on the positions of the tracer particles.

For the second main focus of this study, the object tracking, two tracking algorithms were developed. The conventional tracking algorithm uses a monodirectional, one-pass tracking with a simple initialization process. On the other hand, the multi-pass tracking algorithm with robust initialization has a bidirectional, multi-pass character with a robust initialization. Both tracking algorithms were then compared with 108 virtual tracks with various flow structures in the SIL environment. In 100 of these SIL simulations, the multi-pass tracking algorithm with robust initialization delivered considerably better results. If the overall tracking efficiencies are compared, this algorithm also delivers with 65% significantly higher results than the conventional tracking algorithm with 37%. The efficiency of the multi-pass tracking algorithm with robust initialization also does not decrease with ascending seed rates.

In the last part of this study, 3D-MVPTV measurements were carried out in the test stand, which were then compared with a conventional measurement method using hot-sphere probes. This comparison showed that the 3D-MVPTV delivers plausible results. The possible error sources causing the difference in this comparison were analyzed. Beside the systematic errors, the random errors of the hot-sphere probes were difficult to quantify. In this context, further detailed error analyses for the individual steps of the 3D-MVPTV measurements and its workflow are essential. These points can be carried out in further development steps using advanced measurement methods like PIV and the SIL environment.

One of the most important error sources in the 3D-MVPTV measurements was the size and distribution of the tracer particles. The bubble generator used in this study is controlled manually to produce the tracer particles. That leads to significant oscillations in size, number, and distribution of the tracers particles depending on the capability of the operator of the bubble generator. However, a recently developed bubble generator can generate smaller (down to $0.3mm$) tracer particles for air flows with automatically controlled size and seeding rate [22]. In the future carrying out 3D-MVPTV using such a bubble generator would not only reduce the errors but also increase the resolution of the system due to smaller particle sizes and higher seeding rates.

Parallel computing is utilized in most steps of the 3D-MVPTV workflow and the SIL environment. Here a linear acceleration in computation times is achieved, especially in the camera calibration, object recognition and multi-view triangulation steps. On the other hand, the complexity of the new multi-pass tracking algorithm with robust initialization results in a rise of computational times. However, the implementation of parallel processing using CPUs for this algorithm is restricted due to its sequential execution. Here, the implementation of parallel processing using GPUs will help to decrease the computational times greatly. A further necessary improvement of the software implementation would be the development of a grayscale driver for the webcams. This will not only improve the image acquisition performance but also reduce the size of the data captured.

As described at the beginning of this study, Leonard da Vinci carried out the first fluid visualization studies using his eyes and artistic as well as scientific skills. Nearly 500 years later, the measurement method developed in this study runs with 16 webcams, a bunch of software components and computational resources. The huge improvement achieved through this would probably continue to accelerate in the next years with novelties in imaging and computational technologies. In this context, especially software-in-the-loop environments, also utilized in this study, will enable faster and more precise development of flow visualization.

Bibliography

Books

[1] T. T. Lim A. J. Smits. *Flow Visualization: Techniques and Examples*. 2nd. London: Imperial College Press, 2012. ISBN: 9781848167919.

[5] Asim Bhatti. *Stereo Vision*. 1st. In-Teh, 2008. ISBN: 978-953-7619-22-0.

[6] Pascal Henry Biwole. *Large Scale Particle Tracking Velocimetry for 3-Dimensional Indoor Airflow Study*. National Institute of Applied Sciences of Lyon, 2009.

[7] John F. Foss Cameron Tropea Alexander L. Yarin. *Springer Handbook of Experimental Fluid Mechanics*. Berlin, Heildelberg: Springer-Verlag, 2007. ISBN: 978-3-540-25141-5.

[9] Marc Christenfeldt. *Entwicklung eines Mess- und Reglermoduls für einen Blasengenerator*. 2014.

[10] Peter Corke. *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*. 1st. Springer, 2011. ISBN: 978-3-642-20143-1.

[16] Alessandro Verri Emanuele Trucco. *Introductory Techniques for 3-D Computer Vision*. 1st. Prentice Hall, 1998. ISBN: 978-0132611084.

[18] Olivier Faugeras. *Three-dimensional computer vision: a geometric viewpoint*. 1st. The MIT Press, 1993. ISBN: 978-1-4471-6320-6.

[20] Adrian Kaehler Gary Bradski. *Learning OpenCV*. 1st. O'Reilly, 2008. ISBN: 978-0-596-51613-0.

[21] Eugen Hecht. *Optics*. 4th. London: Addison Wesley, 2002. ISBN: 0-321-18878-0.

[26] Reinhard Klette. *Concise Computer Vision: An Introduction into Theory and Algorithms*. 1st. Springer, 2014. ISBN: 0-262-06158-9.

[27] Robert Laganiere. *OpenCV 2 Computer Vision Application Programming Cookbook*. 1st. Packt Publishing, 2011. ISBN: 978-1-849513-24-1.

[34] Wolfgang Merzkirch. *Flow Visualization*. 2nd. Orlando, Florida: Academic Press Inc., 1987. ISBN: 9780080506586.

[37] Dirk Müller. *Optische Erfassung und numerische Berechnung von zwei- und dreidimensionalen Geschwindigkeitsfeldern mit niedrigen turbulenten Reynolds-Zahlen*. Aachen: Shaker, 2000. ISBN: 3-8265-8222-5.

[39] Frans T. M. Nieuwstadt. *Flow Visualization and Image Analysis*. 1st. Springer, 1993. ISBN: 978-94-010-5191-0.

[40] Steven L. Eddins Rafael C. Gonzalez Richard E. Woods. *Digital Image Processing Using MATLAB*. 2nd. Gatesmark Publishing, 2009. ISBN: 9780982085400.

[41] Andrew Zisserman Richard Hartley. *Multi View Geometry in Computer Vision*. 2nd. The Edinburgh Building, Cambridge CB2 8RU, UK: Cambridge University Press, 2003. ISBN: 978-0-521-54051-3.

[46] Wen-Jei Yang. *Handbook of Flow Visualization*. 2nd. New York: Taylor & Francis, 2001. ISBN: 1-56032-417-1.

Articles

[2] Ronald J. Adrian. „Particle-Imaging Techniques for Experimental Fluid Mechanics“. In: *Annu. Rev. Fluid Mech.* 23 (1991), pp. 261–304.

[8] K. L. Chan and A. K. Forrest. „An empirical study on the effects of spatial discretization error in a stereo vision system“. In: (1990), pp. 48.1–48.6.

[11] G. H. Jirka D. Groß W. Brevis. „Development of a LED-based PIV/PTV system: Characterization of the flow within a cylinder wall-array in a shallow flow“. In: *River Flow 2010* (2010), pp. 1665–1672.

[14] Daniel J. Simons Deborah E. Hannula and Neal J. Cohen. „Imaging implicit perception: promise and pitfalls“. In: *Nature Reviews Neuroscience* 6 (2005), pp. 247–255.

[19] Peter Freymuth. „Flow visualization in fluid mechanics“. In: *Review of Scientific Instruments* 64 (1993).

[25] Andrew K. Forrest Kap Luk Chan. „An empirical study on the effects of spatial discretization error in a stereo vision system“. In: (1990), pp. 48.1–48.6.

[29] Manolis I. A. Lourakis and Antonis A. Argyros. „SBA: A Software Package for Generic Sparse Bundle Adjustment“. In: *ACM Transactions on Mathematical Software* 36 (2009).

[30] Rasmus Lund Jensen Martin Heine Kristensen Jakob Søland Jensen. „Air Temperature Measurements Using Dantec Draught Probes“. In: (2015). ISSN: 1901-726X.

[33] Kenneth I. Joy Mauricio Hess-Flores Shawn Reckerz. „Uncertainty, Baseline, and Noise Analysis for L1 Error-Based Multi-View Triangulation“. In: (2014).

[35] Pirkko Oittinen Mikko Kytö Mikko Nuutinen. „Method for measuring stereo camera depth accuracy based on stereoscopic vision“. In: (2011).

[38] D. A. Papantoniou N. A. Malik Th. Dracos. „Particle tracking velocimetry in three-dimensional flows“. In: *Experiments in Fluids* 15 (1993), pp. 279–294.

Electronic resources

[3] *Advanced Turbomachinery Simulation using STAR-CCM+*. URL: `http://mdx2.plm.automation.siemens.com/sites/default/files/Presentation/5_CD-adapco-CHC.pdf`.

[4] *Allied Vision*. URL: `https://www.alliedvision.com`.

[12] *Da Vinci' s blobs*. URL: `https://lebbeuswoods.wordpress.com/2010/12/03/da-vincis-blobs/`.

[13] *Dantec Dynamics Comfort Sense*. URL: `https://www.dantecdynamics.com/docs/products-and-services/thermal-comfort/PI264_ComfortSense.pdf`.

[15] *Delock PCI Express Card x4: 4 x external USB 3.0-A (Quad Channel) product description*. URL: `http://www.delock.com/produkte/G_89325/merkmale.html`.

[17] *Epipolar Geometry*. URL: `http://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_calib3d/py_epipolar_geometry/py_epipolar_geometry.html`.

[22] *Helium-filled Soap Bubble Generator*. URL: https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=2ahUKEwiB9-yj6OncAhVSxxoKHT8GBrEQFjAAegQICRAC&url=https%3A%2F%2Fwww.lavision.de%2Fen%2Fdownload.php%3Fid%3D2677&usg=AOvVaw2E7zI3C2tV1_qqhrY5FueO.

[24] *Intel X79 Express Chipset block diagramm*. URL: http://www.pcstats.com/articleview.cfm?articleid=2647&page=2.

[28] *Logitech HD Pro Webcam C920 product description*. URL: http://www.logitech.com/en-us/product/hd-pro-webcam-c920?crid=34.

[31] *MATLAB® online documentation*. URL: https://de.mathworks.com/help/matlab/index.html.

[36] *Model-Based Design Definition of Terms*. URL: https://www.autonomie.net/references/model_based_design_defs_24c.html.

[42] *Samsung EVO 850 Series product description*. URL: http://www.samsung.com/us/computing/memory-storage/solid-state-drives/ssd-850-evo-2-5-sata-iii-120gb-mz-75e120b-am/.

[43] *shutterstock*. URL: https://www.shutterstock.com.

[45] *Wikipedia*. URL: https://en.wikipedia.org.

Other resources

[23] *Hermann Rietschel Institut, Technische Universität Berlin*.

[32] *MATLAB® technical support*.

[44] *STAR-CCM+ Version 13.02.011-R8 User Guide, Siemens PLM Software*.