4D Flow Visualization with Dynamic Vision Sensors

Author(s):
Borer, David J.

Publication Date:
2014

Permanent Link:

https://doi.org/10.3929/ethz-a-010344783 →

DISS. ETH NO. 21956

4D Flow Visualization with Dynamic Vision Sensors

A dissertation submitted to
ETH ZÜRICH

for the degree of
Doctor of Sciences

presented by

David Johannes Borer

MSc ETH Masch.-Ing., ETH Zürich
born May 5, 1979
citizen of Erschwil SO, Switzerland

accepted on the recommendation of

Prof. Dr. Thomas Rösgen, examiner
Prof. Dr. Tobi Delbrück, co-examiner

2014

4D Flow Visualization with Dynamic Vision Sensors

Published and distributed by:

Institute of Fluid Dynamics
ETH Zürich
ETH Zentrum, ML H 35
CH-8092 Zürich
Switzerland

http://www.ifd.mavt.ethz.ch

Printed in Switzerland by:

Druckzentrum ETH Zentrum
ETH Zentrum, HG D 48.2
CH-8092 Zürich
Switzerland

Abstract

Flow visualization in wind tunnel testing can provide an intuitive insight into flow phenomena. The simplicity, cost and responsiveness of the visualization method is of crucial importance for practical use. In this work we aim to develop a cost-effective and fast visualization method based on tracking neutrally buoyant soap bubbles with a set of novel cameras. The "Dynamic Vision Sensor" registers the temporal change in illumination with very low latency, capturing fast processes at a low data rate. In contrast to frame based cameras, the pixels of the Dynamic Vision Sensor operate individually. The pixels are sensitive to the time derivative of the logarithm of the intensity; this design offers a high dynamic range of reportedly $120dB$. Intensity changes are reported as a stream of events; an event is the basic unit of information consisting of the pixel address, the time instant of intensity change and whether the change is positive or negative.

The use of helium filled soap bubbles as tracers is an established technique for large scale applications. The advantage are good visibility at distances on the order of meters and low interia. In this work a new type of bubble generator was designed and built to enable low bubble densities and constant conditions during testing.

For a 3D reconstruction of the particle tracks three cameras are used. Two cameras already provide an operable 3D tracking system; a third camera is used to increase the probability of detection in areas of poor lighting or in areas with reflections and poor background contrast.

The particle tracking scheme employs a Kalman filter to estimate the particles' position and velocity. The Kalman filter offers an ideal framework to process the sequential data from the Dynamic Vision Sensors and can handle 2D tracking and 3D reconstruction in a similar manner. The 3D tracking algorithm combines the data from multiple cameras in a straightforward fashion; the number of cameras is not limited. This work provides the proof of concept with three cameras.

The system performance is evaluated with experimental data from wind

tunnel measurements with three different test objects. The results are presented and discussed.

Zusammenfassung

Strömungsvisualisierung in Windkanälen erlaubt eine intuitive Einsicht in fluiddynamische Prozesse und Strömungsphänomene. Für die praktische Anwendung ist es von Vorteil eine Visualisierungsmethode einzusetzen, die einfach im Aufbau ist und zeitnahe Resultate liefern kann. Das Ziel dieser Arbeit ist die Entwicklung einer Methode für volumetrische Geschwindigkeitsmessungen mittels einer neuartigen Art von Kamera, des sogenannten "Dynamic Vision Sensor" (kurz DVS). Die DVS registriert nur die Änderung in Lichtintensität, wobei jedes Pixel individuell gesteuert wird und mit einer hoher zeitlichen Auflösung im Bereich von Mikrosekunden reagieren kann. Die Datenrate gegenüber normalen Kameras wird drastisch reduziert, gepaart mit einer hohen dynamischen Bandbreite von 120 dB ermöglicht sie einen neuartigen Ansatz für den Einsatz im Bereich der Strömungsvisualisierung. Die Änderungen werden in Form von "Ereignissen" übertragen, die jeweils die Pixeladresse, den Zeitpunkt der Änderung und die Änderungsrichtung kodieren.

Die Methode verwendet auftriebsneutrale, heliumgefüllte Seifenblasen als Tracer. Der Vorteil der Seifenblasen sind gute Sichtbarkeit und eine geringe Eigendynamik, die eine direkte Korrelation zwischen Strömung und Geschwindigkeit der Tracer ermöglicht. Ein neuer Typ von Seifenblasengenerator wurde entwickelt und gebaut um eine geringe Seifenblasendichte und konstante Testbedingungen zu ermöglichen.

Um die Bahnlinien von Seifenblasen in einem Volumen zu erfassen werden 3 Kameras eingesetzt. Dies erlaubt eine dreidimensionale Rekonstruktion und verringert die Gefahr von Datenlücken durch ungenügende Beleuchtung und fehlerhafte Rekonstruktion der Bahnlinien.

Ein Kalman Filter wird eingesetzt um die Position und Geschwindigkeit der Seifenblasen zu bestimmen. Der Kalman Filter bietet ein ideales Grundgerüst um den sequentiellen Datenstrom der DVS zu verarbeiten und kann sowohl für 2D Tracking als auch für 3D Rekonstruktion eingesetzt werden. Der implementierte 3D Tracking Algorithmus kann mit Daten von mehre-

ren Kameras umgehen wobei die Anzahl der Kameras grundsätzlich nicht begrenzt sind. Das Konzept wird in dieser Arbeit anhand 3 Kameras demonstriert.

The Leistungsfähigkeit des Systems wird anhand experimenteller Messungen im Windkanal demonstriert. Die Ergebnisse werden präsentiert und diskutiert.

Aknowledgement

I would like to thank Prof. Rösgen for the possibility to work in his group at the Institute of Fluid Dynamics. I am grateful for his numerous and resourceful ideas and suggestions and especially for his continuous support during all ups and downs of the project. I also want to thank Prof. Tobi Delbrück for his advice and support on the Dynamic Vision Sensor, his great interest in the project and for acting as co–examiner. Raphael Berner, formerly from the Institute of Neuroinformatics, and Luca Longinotti and Sim Bamford from Inilabs I thank for their technical support and uncomplicated help on any issues.

Many thanks go to Bianca Maspero and Sonja Atkinson for the help with dealing with all kinds of administrative matters and to our technicians Pius Stachel and René Holliger for their technical support. I thank Lucian Hanimann, Giovanni Mattucci, Georg Borter and Franziska Siebenbrock for their work on student projects which contributed to the project. I also want to thank Florian Küng for proof reading.

Special thanks go to all the people at IFD who made my time at the institute a very memorable and enjoyable time. Finally I want to thank my family and friends for all their support over the last years.

Zürich, December 2014 David Borer

Contents

Nomenclature

Abbreviations

AER	Adress Event Representation
CoC	Circle of Confusion
DLT	Direct Linear Transformation
DVS	Dynamic Vision Sensor
FIBUS	Forschungsinstitut für Bildverarbeitung, Umwelttechnik und Strömungsmechanik, Hamburg, Germany
HFSB	Helium Filled Soap Bubbles
JPDAF	Joint Probabilistic Data Association Filter
LDV	Laser Doppler Velocimetry
LED	Light Emitting Diode
MHT	Multi Hypothesis Tracker
NACA	National Advisory Committee for Aeronautics
PDAF	Probabilistic Data Association Filter
PIV	Particle Image Velocimetry
PMHT	Probabilistic Multi Hypothesis Tracker
POD	Proper Orthogonal Decomposition
PSV	Particle Streak Velocimetry
PTV	Particle Tracking Velocimetry
PWM	Pulse Width Modulation
RANSAC	RANdom SAmple Consensus
SMD	Surface Mounted Device
SNR	Signal to Noise Ratio

Greek Symbols

$\boldsymbol{\omega}$	Rotation vector
Δ_p	Pixel pitch
ϵ_{epi}	Error of the fundamental matrix constraint
ϵ_{rp}	Reprojection error
ϵ_{tri}	Reconstruction error
η	Validation region for the Kalman filter
σ	Standard deviation

Roman Symbols

$\hat{\mathbf{x}}_{k\|k-1}$	The *a priori* state estimate of the Kalman filter
$\hat{\mathbf{x}}_{k\|k}$	The *a posteriori* state estimate of the Kalman filter
$\mathbf{E}$	Essential matrix
$\mathbf{F}$	Fundamental matrix
$\mathbf{K}$	Calibration matrix
$\mathbf{P}$	Camera matrix
$\mathbf{P}_{k\|k-1}$	Covariance of the *a priori* error of the Kalman filter
$\mathbf{P}_{k\|k}$	Covariance of the *a posteriori* error of the Kalman filter
$\mathbf{Q}_k$	Covariance of the process noise of the Kalman filter
$\mathbf{R}$	Rotation matrix
$\mathbf{R}_k$	Covariance of the measurement noise of the Kalman filter
$\mathbf{S}_k$	Covariance of the measurement residual of the Kalman filter
$\mathbf{T}$	Camera center in world coordinates
$\mathbf{t}$	Camera center in camera coordinates
$\mathbf{v}_k$	Measurement noise of the Kalman filter
$\mathbf{w}_k$	Process noise of the Kalman filter
$\mathbf{X}$	Position in $\mathbb{R}^3$ in world coordinates
$\mathbf{x}_d$	Image position in distorted coordinates
$\mathbf{x}_k$	The true state of the Kalman filter
$\mathbf{x}_n$	Image position in normalized coordinates
$\mathbf{x}_p$	Image position in pixel coordinates
$\mathbf{y}_k$	Measurement residual of the Kalman filter
$\mathbf{z}_k$	Position of event as input for the Kalman filter
c_x, c_y	Principal point (pixel units)
$C_{temporal}$	Temporal contrast
$C_{threshold}$	Temporal contrast threshold
$f/\#$	f-number
f_c	Focal length (SI units)
f_x, f_y	Focal length (pixel units)
I	Light intensity
k_{1-5}	Distortion parameters (pixel units)

p_{dvs}	Polarity of event
px	Pixel
q	Scaling factor for the covariance of the process noise of the Kalman filter
r	Variance of the measurement noise of the Kalman filter
t_{dvs}	Timestamp of event
x_{dvs}	Row–index of pixel
y_{dvs}	Column–index of pixel

Subscripts

$(\cdot)_c$	Camera centric coordinate system
$(\cdot)_k$	For Kalman filter: Timestep
$(\cdot)_k$	For calibration: Index of view
$(\cdot)_m$	Index of camera
$(\cdot)_t$	Target centric coordinate system

Chapter 1

Introduction

The objective of this work is to establish the use of a new type of camera called the "Dynamic Vision Sensor" (DVS) in the framework of flow visualization, that is to quantitatively measure flow velocities in a volumetric test section. The method makes use of tracers to track the flow and an imaging device to capture the tracers' position. There are several established techniques following this approach, such as Particle Image Velocimetry, Particle Tracking Velocimetry or Particle Streak Velocimetry, each with their own requirements, advantages and disadvantages. Other methods such as Laser Doppler Velocimetry make use of the velocity dependent frequency shift of light scattered from particles, so that the particle has not to be optically resolved. A recent example is the work by Meier (2012).

The DVS was developed at the Institute of Neuroinformatics at the University of Zürich. It is a temporal contrast sensor with a low latency and high dynamic bandwidth. Each pixel registers individually the change in light intensity, the result is a continuous stream of data with high temporal resolution. This enables fast and time-accurate processing of the tracer signatures.

We aim to provide a measurement technique suitable for large scale applications in wind tunnels. The emphasis lies on a fast and easy to use technique with minimal requirements in experimental preparation, i.e. installation of illumination and measurement hardware. The measurement results are intended to give a fast and representative overview of large scale flow features, such as flow separation and vortices. The final goal is to extract the path lines of the tracers as a visual indication of the behaviour of the flow in the regions of interest. At the present, some performance features of the DVS are limited by its comparatively low resolution of 128×128 pixels, but with the next sensor generation this may be improved. Devices

with resolutions up to 640×480 pixels are under development.

1.1. Previous research

For large scale applications the tracers have to be visible from distances on the order of meters. It has been reported that atomized oil droplets with a diameter of 1 μm can be used to visualize the flow in a planar section with an area of $0.7\,m^2$ using PIV, see Bosbach *et al.* (2009). For 3D3C [1] measurements there has been extensive research in the past decade into Tomographic PIV; the measurement volumes are however usually in the $100 \times 100 \times 100\,mm^3$ range, see Doh *et al.* (2012). For more detailed information see the recent overview of measurement techniques by Cao *et al.* (2014).

In the framework of large measurement volumes it is necessary to use tracers which can be optically distinguished and fulfill the requirements of a passive tracer, they should have low inertia and be neutrally buoyant. The choice generally falls on helium filled soap bubbles (HFSB). The first referenced use in wind tunnels goes back to 1936 by Redon and Vinsonneau (mentioned in Müller (1996)).

For a list of publications with HFSB as tracers before 2003 see Macháček (2002) p. 96. Since then flow visualization involving HFSB has seen a number of new applications. A few examples are given as follows.

The feasibility of using a commercial motion capture system for tracking large numbers of HFSB in 3D has recently been demonstrated by the Motor Industry Research Association (MIRA) in 2009. A 12 camera T40 motion capture system from Vicon is used to track bubbles with a diameter of $3\,mm$ in the wake of a car to investigate the separated flow. There is not much information available about the performance of the system. The case study made available to the public concentrates on the low velocity region of the separated flow. For more information see the online article by Shelley (2009).

Lobutova *et al.* (2009) investigated the flow in a Rayleigh-Bénard cell using a setup consisting of 4 cameras. First correspondences of the imaged particle positions between the calibrated cameras are established from which the 3D positions are triangulated. The tracers are then tracked with a PTV approach in 3D. Two types of tracers were used, in addition to the HFSB with a diameter of $4\,mm$ they tracked latex balloons with a diameter of $15\,cm$, resulting in longer measurement runs due to the longer lifetime of the balloons. The HSFB were used to measure the instantaneous flow field, whereas the latex balloons gave a better measurement for the Lagrangian flow over several minutes. Still, the bubble solution fluid for the HFSB

[1] Short for three-dimensional three-component: measurement of 3 velocity components in a 3D volume.

used in their work produced bubbles with a lifetime of about 4 minutes.

Biwole *et al.* (2009) demonstrated the capability of using a PTV algorithm to reconstruct the time resolved 3D tracks of more than 1400 bubbles in volumes up to $3 \times 3 \times 1.2\ m^3$. Bubbles are initially tracked in 2D image space. The resulting trajectories are checked for correspondences between the cameras and then reconstructed in 3D space. Three cameras were used with frame rates up to 100 fps. The maximum velocities of the investigated flows did not exceed 0.8 m/s.

An new approach using Particle Streakline Tracking was introduced by Rosenstiel & Grigat (2010). It is based on imaging streaks of HFSB in a measurement plane illuminated with a laser light sheet. The approach encodes the streaks by triggering the laser with two different pulse lengths during a single image acquisition. This technique has been used before in PTV. The result is two streaks of different lengths from one bubble, from which the bubble's direction as well as the velocity (corresponding to the length of the two streaks) can be determined. The measurements were conducted in a full–scale Airbus cabin mockup with an area of 4 m^2 with the intention to investigate the flow generated by the air conditioning system.

Bosbach *et al.* (2009) also conducted measurements in an Airbus cabin mockup using HFSB. In this case the flow visualization is carried out by means of PIV. For this the number of bubbles had to be sufficiently high, with about 30'000 bubbles in the field of view for a measurement area of up to $3.1 \times 2.4\ m^2$. The acquired particle density was on the order of 25 particles per interrogation window of 32×32 pixels. The area imaged by a single pixel was larger than the size of the bubbles. It proved to be necessary to defocus the camera, an aspect which also plays a role in the current work, see Section 3.4.

A significant aspect of applications using HSFB concerns the bubble generation. A description of the existing systems is given in Section 2.2.

1.2. Reference case

This work is the continuation of the research conducted by Macháček (2002) at the Institute of Fluid Dynamics of ETH Zürich. His results are used as a baseline for the current work to establish the advantages using a DVS system for tracking. Following is an overview of Macháček's work.

Macháček used a Particle Streakline Velocimetry approach with a stereo camera setup to reconstruct the three–dimensional particle paths. The cameras used were model TM6710 from Pulnix with the specifications listed in Table 1.1.

The disadvantages of the cameras are the low dynamic range and low sensitivity. To achieve a reasonable temporal resolution (which determines

Table 1.1.: Specification of the TM6710

	TM6710
Frame rate:	60 / 120 Hz
Pixel size:	$9 \times 9\ \mu m^2$
Resolution:	$640 \times 480\ px^2$
Chip size:	1/2 inches
S/N Ratio:	45 dB min.
Dynamic range:	8 bit

the accuracy of the velocity reconstruction) the frame rate had to be sufficiently high. A trade–off had to be made between dynamic range and frame rate. Using the full frame rate resulted in about 36.8 MByte/s of data for each camera. Two PCs with 600 MHz processors were necessary to record the data. A typical measurement run lasted about 27.5 seconds, resulting in 3300 frames per camera. The data is then post–processed. After the initial background subtraction the streaks are extracted and associated pair–wise from each camera via the epipolar constraint. Streak pairs are reconstructed to the 3D streak line, from which the velocity is calculated. The tracking algorithm did not handle crossing particle streaks; the seeding density was kept accordingly low with approximately 10 bubbles per image. The system was able to measure velocities up to 18 m/s with a reported accuracy of 0.024 m/s. The major issues of the approach were the low dynamic range of the cameras and the large amount of image data. Measuring, recording and post–processing set high requirements for the hardware.

This thesis addresses the issues encountered by Macháček and will show the advantages and limits of tracking particles with the Dynamic Vision Sensor. In the next chapter the experimental setup and hardware will be introduced first.

Chapter 2

Experimental environment

2.1. Introduction

This chapter will give an overview of the experimental equipment and the test facilities. The DVS cameras will be discussed separately in Chapter 3 in more detail. At the end of this chapter the work flow of the data acquisition procedure will be summarized. The main components of the experiments are given as follows.

- Helium filled soap bubble generator
- Wind tunnel environment
- Illumination
- Test models
- DVS cameras

2.2. Helium Filled Soap Bubbles

The advantages of Helium filled soap bubbles (HFSB) are neutral buoyancy and low inertia, therefore the fluid velocity can be directly inferred from the velocity of the bubble. The size of the bubbles is in the order of a few millimetres and they can be seen from large distances, making them ideal for large scale applications. The disadvantage is their low specular reflection, which can be less than 5% of the incident light. The short life span can be also be a problem for experiments that investigate slow flow fields, such as natural convection. Since bubbles cannot be stored, they have to be produced on–site. Following is an overview of bubble generation methods used in the past and ongoing research.

Macháček (2002) used a FIBUS type nozzle (see Müller *et al.* (1997)). According to Bosbach *et al.* (2009) it is capable of a generation rate of up to 200'000 bubbles per second.

A new approach was presented by Lobutova *et al.* (2009). The injection tube was redesigned to enable a more precise production of bigger bubbles. An alternating air volume flow rate is induced by the pressure waves generated by a loud speaker, which are used for an active bubble separation at the conically shaped nozzle exit.

To ensure neutral buoyancy a so–called cyclone can be positioned after the nozzle. The working principle of this device is to produce a swirling flow inside a cylinder with a feeding tube tangent to the wall where bubbles and a co–flow enter. Bubbles lighter than the air experience a buoyant force towards the center of the cylinder (in direction of the increase of centrifugal potential). Here they are extracted by a tube. Heavier bubbles remain in the cylinder and eventually burst.

The visibility can be enhanced by using fluorescent dyes and a monochromatic light source. The light emitted from the dye has a longer wavelength than the light of the illumination. A filter can be used to block scattered light from the surroundings to increase the contrast of the bubbles. Macháček (2002) investigated the use of fluorescent dyes in his work. However, the fluorescent signal of the HFSB proved to be too weak, and the CCD camera sensitivity too low for detection at moderate wind speeds in the wind tunnel. Lobutova *et al.* (2009) also describes the use of dyes and additionally considered filling the bubbles with smoke, but no detailed information is given on the actual improvement during experiments.

It is possible to increase the life span by using additives in the soap solution, which form a membrane after the water in the bubble film has evaporated. This approach was not further expanded on in this work, as the residence times in the wind tunnel were on the order of seconds. The use of a standard soap solution proved sufficient.

The next generation of bubble generators is described in Christina Politz & Ranasinghe (2013). It is a high throughput system for large bubble generation rates. The system is intended to be applied to visualize the wake vortex generated by airplanes. Multiple nozzles are mounted onto a plate with a layered feed system for soap, helium and air.

2.2.1. Requirements

Previous work generally required a high seeding density. For PIV this requirement is essential, for PTV or especially PST a lower density may be advantageous for post processing, but disadvantageous in terms of data collection, i.e. the information content per acquired image is low. For the DVS the information content is (ideally and neglecting noise) equal

to the recorded data, hence the only limit may lie in the time required to collect a sufficiently large data set of bubble tracks. The main disadvantage at present is the low resolution of the DVS and, depending on the setup (especially at higher velocities), a generally lossy particle signature with gaps in the streaks. Theses gaps are caused by insufficient background contrast, for example in the case of reflections from the lighting on the model and walls or due to parts of the wind tunnel that are not painted black. Gaps may also occur if the particles' image is smaller than a pixel and passes past the light sensitive area of the pixel. This can be seen when the streak direction is approximately along pixel lines; long gaps occur when the particle crosses from one pixel line to the next. To avoid conflicts in the case of particle streaks crossing each other (from the view of one camera), a low and controllable bubble generation rate is desired.

Quantitative information about HFSB visibility in a typical setup was difficult to acquire in the development stage of this measurement technique with the existing bubble generator due to the high bubble density generated with the existing FIBUS type nozzle. The derivation of statistics from a multitude of tracks crossing and obscuring each other with varying velocities did not work satisfactorily and the data generated from the initial studies was difficult to interpret. To ensure a reliable measurement environment it was considered necessary to acquire the data with a low and robust bubble generation rate over a reasonably long period, typically on the order of one to two continuous hours for parameter studies. Another reason for a new bubble generator was to facilitate the data acquisition and post–processing in the measurement stage.

In a student project a new working principle for a bubble generator was tested and implemented. Borter (2013) built a first working prototype which was later adapted and integrated into the old bubble generator used by Macháček (2002).

Following are the requirements listed in decreasing order of importance:

1. Neutrally buoyant bubbles.
2. Monodisperse bubbles (constant size) for consistent optical visibility between measurement runs.
3. Control over the generation rate (typically 5-20 HFSB per second).
4. Control over the size (typically 2-3 mm in diameter).
5. No filter process after the bubble formation required.
6. Simple and accessible construction.

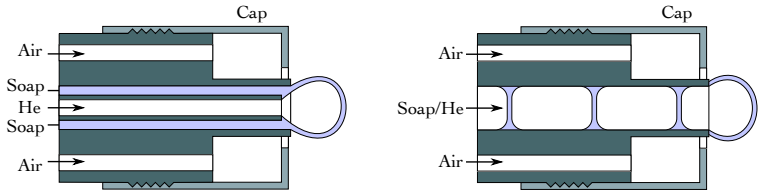
The last requirement is intended for cleaning and maintenance. With the older nozzles (e.g. a commercial product from SAGE ACTION, Inc.) a major problem after usage was that the residue of the soap solution would dry up and clog the pipe. Due to the compact build with right angled pipes, unblocking was nearly impossible.

In other systems bubbles are first filtered for buoyancy in a cyclone. This step is not necessary with the new system.

2.2.2. Bubble generator design

With the FIBUS type nozzle (Müller *et al.* (1997)) bubble formation takes place at the nozzle tip. A pipe-in-pipe system feeds the helium in the inner pipe while the outer pipe provides the soap solution. A new bubble is created when there is enough soap to form a film over the helium pipe, initializing the bubble growth by the flow of the helium. At low bubble generation rates this process is not very reliable.

In Figure 2.1a a FIBUS type nozzle is shown. In this design the cap has an adjustable position. With this system two types of nozzle can be distinguished, Okuno *et al.* (1993) classified them as pitot type nozzle and orifice type nozzle. The pitot type nozzle injects the co–flow before the pipe exit, as seen in Figure 2.1a. The co-flow shears the bubble from the pipe exit when the bubble has reached a certain size. The orifice type nozzle places the cap after the pipe exit, forming an orifice through which the bubble and the co–flow streams. The orifice is generally smaller than the soap/helium pipe diameter, thus squeezing the bubble and forcing bubble separation after the nozzle exit. This configuration has the advantage of higher bubble generation rates than the pitot type nozzle and requires lower co–flow speeds.



(a) Old Fibus type nozzle **(b)** New two–phase nozzle

Figure 2.1.: Nozzle types

The aforementioned system relies on the fluid dynamic processes taking place at the nozzle exit, especially depending on the behaviour of the soap solution forming a film over the inner pipe. The new system uses a T–

junction to form a soap–helium slug flow, prescribing the size of the bubble through the amount of helium enclosed by the soap solution and the mass of the bubble by the combined mass of helium and soap in each periodic compartment. This allows to precisely define the bubble size before the actual bubble formation at the exit. In Figure 2.1b the resulting slug flow is shown. Figure 2.2 shows the implementation of the T–junction. To detach the bubble from the exit a pitot type nozzle is used.

In Equation 2.1 the condition for buoyancy is used to derive an expression for the volume ratio of soap and helium under the assumption that the soap film thickness is much smaller than the radius of the bubble (for a more detailed derivation see Borter (2013)).

$$\frac{V_{soap}}{V_{He}} = \frac{1}{\rho_{soap}} \left(\rho_{air} - \frac{p_{\text{inf}}}{R_{He}T} - \frac{4\sigma}{rR_{He}T} \right) \tag{2.1}$$

where r is the radius of the bubble, R_{He} is the specific gas constant for helium, T is the temperature of the helium, $\rho_{(.)}$ are the densities, σ is the surface tension of the soap solution and p_{inf} is the air pressure. The volume of helium for a single bubble is given by

$$V_{He} = \frac{4}{3}\pi r^3. \tag{2.2}$$

The volume ratio for a bubble of diameter 4 mm is $1.2 \cdot 10^{-3}$. To produce a slug flow with such high differences in volume at low frequencies proved to be difficult to achieve. Initial tests using valves to control the volume flow produced unstable and strongly varying results.

For a generation rate of $f = 5\ Hz$ and a bubble diameter of about 4 mm, the volume flow for helium amounts to $\dot{V}_{He} = 9.1\ ml/min$ and for the soap solution to $\dot{V}_{soap} = 10.8\ \mu l/min$. These values are based on practical experience obtained using the new bubble generation system and represent the minimal stable bubble generation rate. Delivering the soap solution at such low flow rates and to produce a slug flow with the required volume ratio in the T-junction presented the main problem.

The solution was to actively control the volume rates. In the case of the soap solution a pulsed delivery system is required. For the helium a flow controller from Bronkhorst is used, with a resolution better than the required ml/min and capable of delivering a flow rate with an accuracy of $0.35\ ml/min$ as specified by the manufacturer. The problem of delivering the soap solution was solved by using a micro-valve with a fast response time in the μs range. See Appendix A for more detailed information.

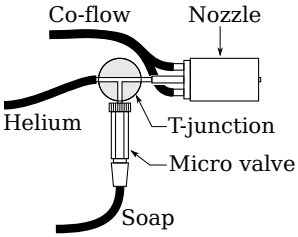


Figure 2.2.: New bubble generator nozzle

2.3. Wind tunnel

All experiments were conducted in the wind tunnel of the Institute of Fluid Dynamics, which was built under Prof. Jakob Ackeret in 1934. It is a medium sized low speed wind tunnel with a closed loop design and two fans driven by two electric motors of 2×173 kW, capable of a maximum speed of 60 m/s. The test section measures 2.1×3×4 m (height×width×length). The tunnel offers an entrance and two acrylic windows, on top and opposite the entrance. It has a built–in force balance below the test–section for precise force measurements.

(a) View towards the entrance of the test section.

(b) View inside the test section (Photo @Adrien Barakat).

Figure 2.3.: Wind tunnel at the Institute of Fluid Dynamics, ETH Zürich.

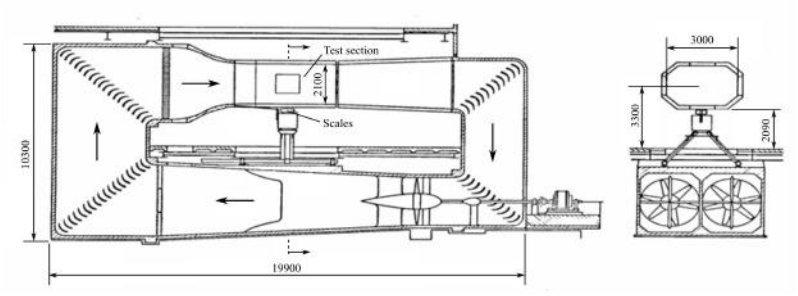


Figure 2.4.: Diagram of the wind tunnel facility (dimensions in mm).

2.4. Illumination

The illumination is essential for the visibility of the bubbles. Required is a high illumination brightness, since we are interested in a volumetric measurement. Care has to be taken dealing with reflections and scattered light from the walls and test models.

Macháček (2002) used two halogen spotlights for illumination, of the kind used in film and in studio photography. The power output could be selectively set to 120 W, 1000 W or 2000 W. Positioning of the spotlights was on top of the wind tunnel directed through an acrylic ceiling window into the test section. For a higher contrast the wind tunnel walls were painted black, allowing a better distinction of the HFSB from the background.

The development of LEDs in the last decades has led to a compact integration of high power illumination at low cost and low power consumption. In comparison to halogen lamps the same luminosity with LEDs is achieved with a power consumption of a factor 6 to 7 less. The unwanted heat production is reduced. For this work we use the 50 W LED Light Engine from Power Systems (`http://www.power-systems.de`). Each LED is approximately equivalent to a 300 W halogen light and comes at a cost of about 100.- Swiss Francs per unit. In total 12 LEDs are used, resulting in approximately 3.6 kW halogen equivalent light intensity in total. The LED Light engines are grouped in pairs of two together on a swivel arm making the direction of illumination easily adjustable (Figure 2.5). Each LED is fitted with a reflector with a 50° illumination angle. The LEDs can be individually controlled and dimmed using a DALI digital bus system.

Figure 2.5.: High power LED; pair–wise mount.

The reason to use multiple light sources is to enhance the visibility of the HFSB. Light from multiple directions will lead to more reflections on the bubble and better detectability by the cameras. In Figure 2.6 the positions and the naming conventions of the LEDs are shown. This arrangement allowed for a very even illumination from all views. Four of the LED pairs are positioned inside the tunnel (A,C,D and F) for an even light distribution. For safety reasons high wind speeds were not applied as the swivel arms were not tested for high loads resulting from aerodynamic drag. This is only a minor drawback since the particle signal declines with speed, see Section 3.3. At the time of this work with the current DVS cameras it was difficult to achieve good measurements with speeds higher than 15 m/s.

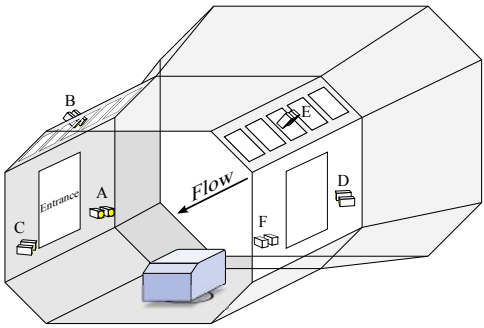


Figure 2.6.: Wind tunnel with positioning of the LEDs and the naming convention for the positions. B and E are positioned in the top windows, A, C, D and F are positioned inside the wind tunnel.

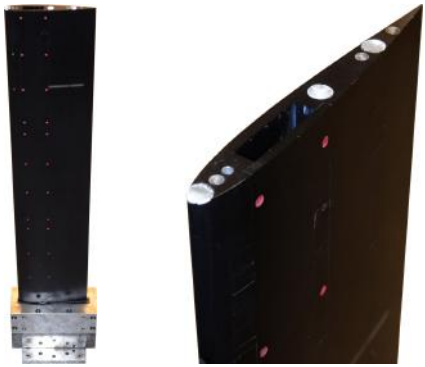
2.5. Test cases

To check the performance of the tracking system three models were used. Each model offers different flow characteristics. Table 2.1 gives an overview of the models.

Table 2.1.: Test models with dimensions and flow feature.

	Dimensions $[mm]$	Flow feature
NACA wing	1000×200, height and chord	Tip vortex
Delta wing	80, side of equilateral triangle	Vortex pair
Ahmed Body	$288 \times 389 \times 1044$	Separation

The wing has a symmetric profile defined as NACA[1] 0012 according to the NACA 4-digit family of airfoils. The thickness of the profile is 12% of the chord length of 20 cm. The wing was set at an angle to the flow to generate a tip vortex. No end plate was used in the present configuration.



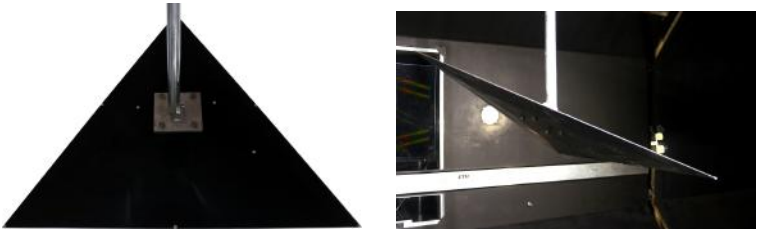
(a) Side view.

(b) Close–up of the profile

Figure 2.7.: NACA wing.

[1] National Advisory Committee for Aeronautics

The delta wing is an equilateral triangle with a side length of $0.8\,m$. The leading edge is blunt with a thickness of $3\,mm$ thick. The profile thickness increases continuously to $18\,mm$ towards the center of the wing. It can be mounted at a variable angle to the flow, typically around $20°$. At the leading edges two counter rotating vortices are induced even at low velocities. This model has also been used by Macháček (2002) for validation.



(a) View from the back. **(b)** Mounted in the wind tunnel.

Figure 2.8.: Delta wing.

The Ahmed body was introduced by Ramm & Faltin (1984) as a representative model to investigate ground wakes of vehicles. It has since become an established model in the field of vehicle aerodynamics, the experimental results have been used for validation of numerical flow simulation codes and turbulence models (CFD). It is a bluff body with a tapered ending with a slant angle of usually $25°$ or $35°$. The form is intended to induce flow separation similar to the flow around vehicles.

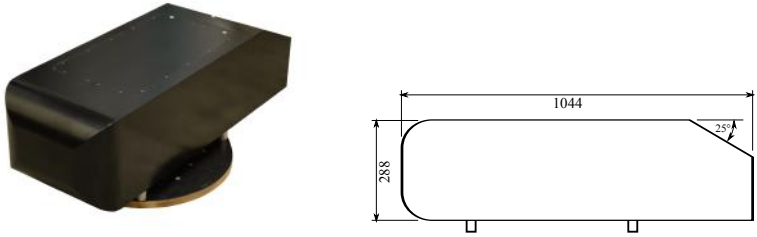


Figure 2.9.: Ahmed body with $25°$ slant angle. Dimensions are in mm.

2.6. Work flow

Following is an overview of the necessary steps to acquire a data set for a measurement, including the calibration. The calibration step involves a target with LEDs to measure points inside the test section. This will be covered in more detail in Chapter 4. This overview documents the interplay of the numerous hardware and software components required to perform a typical data acquisition cycle.

1. Install the model in the wind tunnel. Preferably ensure dark and non–reflective surfaces.

2. Install LEDs at pre–defined positions inside the wind tunnel, direct towards test section.

3. Install DVS cameras where appropriate. Connect the synchronization cables with a designated master and 2 daisy–chained slaves. Connect the USB cables.

4. Start the PC. Start MATLAB and the main routines. Start the jAERViewer and open 3 windows for the 3 cameras. Set the master camera.

5. Initialize bubble generator. Turn on the piezo driver, wave function generator and Bronkhorst volume flow controller. Insert bottle with soap solution, open the main valves for pressurized air and helium. Flood the soap supply by opening the micro–valve.

6. Start the wind tunnel at low speed and start the bubble generator. Fix the nozzle upstream of the measurement section.

7. Set the bubble generator to a stable working point and ensure the production of neutrally buoyant bubbles by adjusting the volume flow of soap and helium.

8. Focus the DVS cameras so the bubbles are best visible (this has so far still to be judged by the user, in future a filter with online extraction of streak statistics could help), adjust the LED illumination if necessary.

9. With fixed aperture and focus calibrate the cameras with the target board. Dim the illumination LEDs for better recognition of the target LEDs. Start the acquisition and move the board throughout the measurement volume. An acquisition time of 30 seconds is usually satisfactory. The first position of the target board can be used to set the coordinate reference.

10. Start the measurement. Scan the volume by guiding the bubble generator nozzle upstream of the test section.

Chapter 3

Dynamic Vision Sensor

The Dynamic Vision Sensor (DVS) is a temporal contrast sensor with a 128×128 pixel resolution and was developed at the Institute of Neuroinformatics at the University of Zürich. The basic idea behind the DVS is to mimic the behaviour of biological vision. Conventional digital cameras are limited by equal exposure times of all pixels, identical pixel gain and the finite pixel capacity for integrated photo charges. Problems arise in scenes with high dynamic range, when information of low contrast features is lost. To capture fast processes a high frame rate is required, while often only part of the image data contains the actual information. The DVS follows a different approach by using independently operating pixels, each can in essence choose its own exposure time and act on luminance changes with a very low latency. The pixel responds to a relative change in intensity by generating an event. Each event is encoded with the time instant and the sign of the change of luminance (referred to as polarity). An ON event occurs for positive polarity and an OFF event for negative polarity. The output of the DVS is an asynchronous stream of data using an address event representation (AER) protocol. The data is sparse and time accurate, with a reported latency of 15 μs in good lighting conditions. For a detailed description of the DVS and previous research see Lichtsteiner (2006), Lichtsteiner *et al.* (2008) and Berner (2011).

The DVS is sensitive to the temporal contrast $C_{temporal}$

$$C_{temporal} = \frac{1}{I(t)} \frac{\mathrm{d}I(t)}{\mathrm{d}t} = \frac{\mathrm{d}(\ln(I(t)))}{\mathrm{d}t}. \tag{3.1}$$

I is the photocurrent produced by the photodiode and is proportional to the pixel illumination. Because the DVS detects changes in logarithmic intensity, it reacts similarly to local changes independent of the global

illumination. The event generation rate is approximately proportional to

$$Event\ rate \approx \frac{C_{temporal}}{C_{threshold}}. \tag{3.2}$$

The contrast threshold $C_{threshold}$ is the minimum relative illumination change needed to trigger an event and is the effective resolution in intensity change.

Table 3.1 gives an overview of the specification of the DVS, taken from Berner (2011), p. 11.

Table 3.1.: Specification of the DVS128

	DVS128
Process technology:	0.35 μm, 4M2P
Pixel size:	$40 \times 40\,\mu m^2$
Fill factor:	8.1 %
Resolution:	$128 \times 128\,px^2$
Dynamic Range:	120 dB
Fixed pattern noise FPN:	2.1 %
Power consumption mW:	24

The jAER software project provides the software to control the DVS. It is open source and can be downloaded at `http://jaerproject.net/`. The software is written in Java and supports multiple cameras. The present version requires a viewer, called jAERViewer, to be opened for each DVS camera.

The chip includes a programmable bias generator that enables easy parameter adjustment. A simplified set of 4 settings is implemented in the software to enable an intuitive adjustment of the behaviour.

Bandwidth Tweaks bandwidth of the pixel front end (slower to faster).

Threshold Adjusts event threshold (less/more events to higher/less events).

Max firing rate Adjusts maximum firing rate, inverse of refractory period (slower to faster).

Balance Adjusts balance between ON and OFF events (more OFF events to more ON events).

All bias settings can be set in the jAERViewer. Optionally, a command can be sent from proprietary software to a specified port to remotely control

the bias, for which values in the range of -1 to 1 cover the available settings range. Another command can be used in the same manner to start and stop the data acquisition. In this work Matlab is used to control the software.

Each DVS is identified by its serial number and is named accordingly in the thesis. Table 3.2 gives an overview of the DVS that were used. The reason to exchange individual DVS cameras during testing was due to differences in behaviour and sensitivity in the initial measurements (i.e. DVS 111) as well as general problems during recording. A common problem showed up as excessive event generation without any stimulus (i.e. DVS 304). A reason for this problem may have been due to an insufficient power supply by the USB ports of the PC motherboard. This problem eventually recurred and was solved by installing a new USB PCI card with a dedicated power supply connected to the PC's power supply.

Table 3.2.: Serial numbers of DVS used in this thesis.

	Serial number		
Initial set used until ca. Oct. 2013	111	304	315
Final set used for the final tests and measurements	222	227	315

The objectives used throughout this work are from Schneider Kreuznach with a focal length of $8mm$ and a maximum aperture of $f/1.4$. They offer a wide field view of ca. 35° in case of the DVS and low distortion. The large aperture provided sufficient flexibility for an extensive parameter study to find the optimal settings.

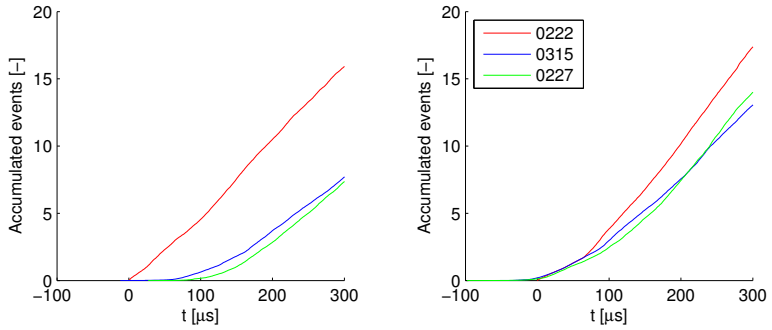
3.1. Synchronization

The DVS has a maximum bandwidth of 1 $MEPS$ (mega events per seconds) and is able to achieve a timing precision of 1 μs. Synchronization of the timing between multiple cameras is necessary to enable a correct registration and subsequent correlation of events. A cable connecting two DVS cameras is used to transmit a synchronization signal where one DVS acts as master providing the signal and the other DVS acts as slave receiving the signal and adjusting its time accordingly. Multiple camera can be daisy–chained with one master and several slaves. In a previous version of the DVS firmware a 1 MHz signal was used as synchronization signal. Positioning the DVS cameras further apart caused some problems because the longer cable could not carry a 1 MHz signal. A new implementation uses a synchronization signal with a 100 μs period, in between the slaves use their own clocks to measure the time and adjust upon receiving the

next synchronization signal.

The accuracy of the synchronization was tested by recording a pulse from a single LED with multiple DVS cameras. The LED is defocused to cover a circle of approximately 10 pixels in diameter. A time resolved signal is calculated as a cumulative sum of events originating from pixels viewing the LED. The LED pulse is evident as a strong increase in event generation. In Figure 3.1 the signals from 10 individual pulses are averaged. Two test series of 6 runs each were conducted. In each series the connection order of the synchronization cables was varied, with each camera acting as master in two runs. Two synchronization cables with lengths of $5\,m$ and $3\,m$ were used. The length difference of the cables is due to the initial setup where one camera was positioned on the opposite side of the wind tunnel and required a longer synchronization cable.

The initial trial shown in Figure 3.1a showed a timing discrepancy on the order of $100\,\mu s$ between cameras. The reason proved to be due to differing aperture settings. The objectives were set to focus on the LED without much thought given to the aperture. In a second trial shown in Figure 3.1b the aperture of all objectives are set to the same value.

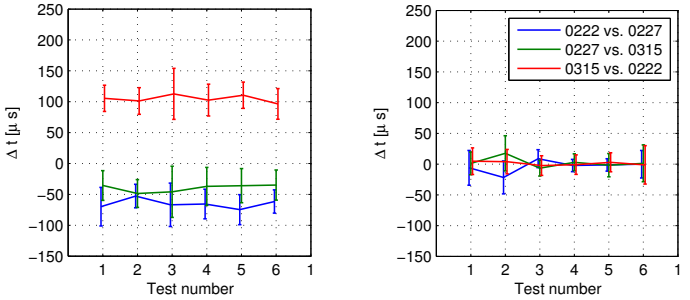


(a) All apertures set to different values. **(b)** All apertures set to the same value.

Figure 3.1.: Effect of the optical setup on the registration of a LED pulse between 3 synchronized DVS cameras. The curves are the phase average of the accumulated events from 10 LED pulses.

Figure 3.2 depicts the mean and standard deviation of the time differences of the pulse registration between cameras in the tests.

The apparent differences between the two series is due to the optical setup of the cameras. In the first run the cameras each had different settings for the aperture. The DVS 222 had the lowest aperture, hence collecting more



(a) All apertures set to different values.

(b) All apertures set to the same value.

Figure 3.2.: Time difference between pulses for test runs with varying connection of the synchronization cables.

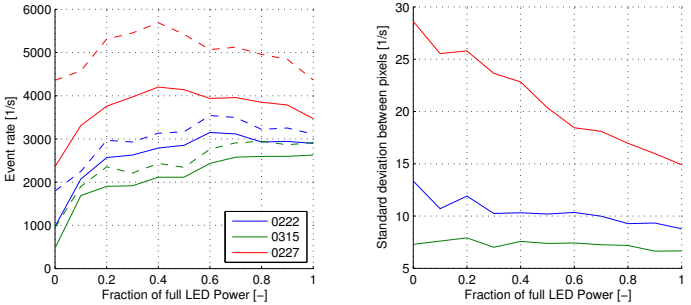
light. In the second run all objectives are set to the same aperture; the pulse registration between cameras is now simultaneous. Lichtsteiner *et al.* (2008) demonstrated the interdependence between the chip illuminance and the latency, with differences easily reaching 100 μs at lower lighting conditions with the bias settings optimized for speed. A direct comparison is not possible as the chip illuminance in our test was unknown. The synchronization cable connectivity had no influence as all variations produce similar results. Overall the results show similar behaviour with values in the expected order of magnitude.

The conclusion derived from this test in regard to particle tracking is that an equal optical setup of the cameras during the measurements should be strived for, if possible. However, if the overall situation during the measurement is taken into account a difference between latency should not present a major problem. The accuracy required for tracking can be approximated as follows: Assuming a velocity of 20 m/s for the particle, a synchronization error of 100 μs will lead to a spatial mismatch of 2 mm between events from different cameras. This is less than the spatial resolution of about 7.5 mm per pixel achieved for applications in the wind tunnel with the current setup.

3.2. Noise

A list of different noise sources is given in Lichtsteiner (2006), wherein he also states that at low illumination, shot noise (photon counting noise) will likely dominate other noise sources. In this section an attempt is under-

taken to set a base line of the background event generation for the experimental setup with the aim to check for any problems with illumination. A major concern for the choice of illumination was the rectification of AC to DC by the LED power suppliers via pulse–width–modulation (PWM). Any ripple in the current driving the LED might be registered by the DVS. A simple test was conducted by illuminating a black, non–reflective piece of cloth as background with 4 individual LED engines at varying power. All the DVS cameras were set to nominal bias and defocused on the background. No movement is present; all events are due to either electronic noise or temporal changes in illumination.



(a) Event generation rate (continuous line for positive events, dashed line for negative events).

(b) Standard deviation of event count between pixels.

Figure 3.3.: Events registered on illuminated black background.

Figure 3.3 gives an overview of the test results. The event generation rate increases with illumination intensity; the event generation rate between all cameras is on average approximately constant at an illumination higher than 30%. The individual pixel response has a higher variance at lower illumination. The results do suggest an influence of the illumination on the background noise. The differences between cameras however make a definite interpretation difficult. Furthermore, the noise was expected to higher at dimmed LED operation where the PWM would effect the steadiness of the illumination the most; the results show no clear indication of this effect. In the experiments the LEDs did provide a good particle visibility (depending mainly on the positioning of cameras and LEDs to avoid reflections and direct illumination). The PWM of the LED power suppliers did not impede the measurements in a significant manner.

3.3. Event generation

The signal generated by a particle moving across the field of view of a DVS camera will depend on the change of the temporal contrast, given in Equations 3.1 and 3.2. An idealized expression for the event generation of a single moving particle is derived as follows. The particle image is given as a time invariant function $I_p = f(\xi)$, where ξ is a coordinate centerd at the position of the particle moving with velocity u. The moving frame ξ of the particle and the stationary reference frame x of the camera are related with $\xi = x + ut$. For simplicity a linear function for $I(\xi) := \alpha\xi$ is assumed (as seen in Fig. 3.4, only the increasing part of the intensity is considered to calculate the number of ON events).

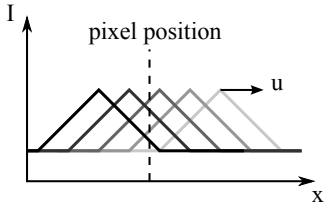


Figure 3.4.: Simple linearized signal of passing particle

The intensity change seen by the camera at a fixed position x is given by the material derivative. By integration the intensity is derived.

$$\frac{\mathrm{D}I}{\mathrm{D}t} = \frac{\partial f}{\partial t} + u\frac{\partial f}{\partial \xi} = \alpha u \;\rightarrow\; I = \alpha u t + I_{bg} \tag{3.3}$$

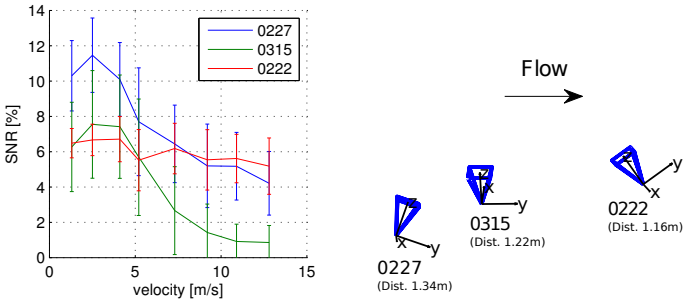
The offset I_{bg} can be regarded as the intensity of the background. With this relation the event rate can by approximated by

$$Event\ rate \propto \frac{\alpha u}{\alpha u t + I_{bg}}. \tag{3.4}$$

The event count per pixel is calculated by integrating the event rate over the time which the particle image takes to cross the pixel area. Assuming a length scale L of the image, the time duration is $\Delta t = \frac{L}{u}$. The maximum intensity is now given as $I_{max} = \alpha L$. The total event count of ON events is proportional to

$$Event\ count = \int_0^{\Delta t} (Event\ rate)dt \propto \log\left(\frac{I_{max}}{I_{bg}} + 1\right). \tag{3.5}$$

The same result is derived choosing a Gaussian function as representation for the particle image. Based on this relation the total event generation should ideally be independent of the velocity of a passing particle. In the measurements however a different behaviour is observed. At higher velocities the particle signature has a lower event count. As an example for a real application measurements at varying velocities around the tip of a NACA wing were conducted. The wind speeds varied between 1 and 13 m/s. As illumination all 12 LEDs were set to full power and the 3 cameras were situated near the entrance of the wind tunnel. The focus was set individually to maximize the particle signature, the aperture was set to f/2.8 for all cameras.



(a) SNR of flow at varying speeds past NACA profile

(b) Camera positions with distance to particle tracks (top view).

Figure 3.5.: Experimental setup with 3 cameras in the wind tunnel.

The calculation of the SNR is found in Equation 5.1. Camera 222 is looking at an angle of approximately 45° towards the flow, resulting in a lower velocity of the particle image compared to the other cameras. For cameras 227 and 315 the signal decreases considerably with higher speeds.

The reason for this behaviour might be due to two properties of the DVS pixel design: A built-in functionality, called the *refractory period*, during which the pixel cannot generate another event; and the low–pass behaviour, shown in Figure 3.6. The refractory period limits the maximum firing rate of individual pixels to prevent small groups of pixels from taking the entire bus capacity (Lichtsteiner *et al.* (2008)). The temporal contrast resolution is reduced for large temporal contrasts, e.g. for fast moving objects. To date no study was conducted to check the influence of the refractory period on the signal of moving particles; this is left to future studies and might lead to more optimal settings for higher velocity applications. The conclusion from the test is that the DVS is less suited for higher velocities.

Assuming the image of a particle is represented by a Gaussian function with a variance on the order of $\sigma_I = 0.3\,px$ (corresponding to an image size of about $1\,px$ in diameter), it is possible to derive the frequency spectrum of the intensity signal of a passing particle by applying the Fourier transform. The result is again a Gaussian function.

$$I(t) = I_{max} e^{-\frac{(ut)^2}{2\sigma_I^2}}, \quad \mathcal{F}\{I(t)\}(f) = \frac{I_{max}\sigma_I}{u}\sqrt{2\pi} e^{-\frac{-2\pi^2\sigma_I^2 f^2}{u^2}} \tag{3.6}$$

The frequency range can be given by the variance of the Fourier transform with

$$\sigma_f = \frac{u}{2\pi\sigma_I}. \tag{3.7}$$

The velocity u is the velocity of the image in px/s, given by $u = \frac{U f_c}{Z\Delta_p}$ with velocity U, distance to camera Z, focal length f_c and pixel pitch Δ_p. Lichtsteiner showed a limit of event generation for these frequencies at lower illumination levels, see Figure 3.6. With increasing bandwidth the amplitude of the frequency response also decreases inversely proportional with the velocity.

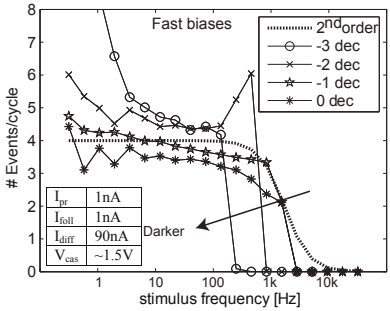


Figure 3.6.: Frequency response of the DVS, Lichtsteiner (2006).

The frequency range can be expressed as the cut–off frequency at which the response declines by 3 dB.

$$f_{-3dB} = \frac{\log(2)}{2\pi}\frac{U f_c}{Z\Delta_p\sigma_I} \tag{3.8}$$

For a typical setup the velocity at $f_{-3dB} = 1\ kHz$ is 16 m/s. The decline in signal quality was observed for velocities in this range.

3.4. Imaging HFSB

The DVS has been used in a variety of applications, of interest for this work is the use for 2D particle tracking in a pipe flow by Drazen *et al.* (2011). The experiment used particles of 950 μm diameter, this was equal to an optical image of 3 pixels in diameter. Particles were identified by coherent activity of neighbouring pixels in a given time window. Tracking the particles was achieved by a nearest neighbour approach. The difference to the work carried out here is that the HFSB are under–resolved if the optical system is set to focus. Defocusing may increase the particle image but leads to a lower temporal contrast.

The DVS offers a high dynamic range of 120 dB, which should be advantageous for HFSB imaging. Visualizing the specular reflectance of the HFSB in general and especially at larger scales and high velocities proved to be non–trivial. Macháček (2002) for example went to such lengths as repainting the wind tunnel walls of the IFD in black to get sufficiently good contrast.

A few issues arose in imaging HFSB with the DVS:

- Discrete events with no information on the magnitude of illuminance. To differentiate between events is complicated in that they only show the polarity of the intensity change, leading to low level noise having the same weight as actual bubble generated events.

- The low fill factor of the DVS of 8.1%. It is actually disadvantageous to have the particles in focus, particle tracks may exhibit gaps where the image passes the light sensitive area. Defocusing is necessary to get an optical image of at least 1 pixel in diameter. The loss of illuminance however will diminish the measurable signal.

- To determine the correct position of the particle center the size of the optical image should preferably be larger than 1 pixel, due to the aforementioned loss of illuminance this is not always feasible.

- The illuminance is inversely proportional to the square of the distance between camera and HFSB. Placing the cameras closer produces a better signal but reduces the measurement volume and increases the velocity of the particle image, the limitation of which was shown in Section 3.3.

The following sections establish a reference with a standard camera and give an overview of the behaviour of the DVS with respect to the available parameters.

3.4.1. Reference case with the TM6710

The DVS cameras are not capable of producing "regular" images so that as an example and to establish a reference, a measurement run with the TM6710 cameras used by Macháček was conducted. In Section 1.1 details of the TM6710 are described. The images were acquired at the maximum frame rate of 120 frames per second with the maximum exposure time (the period between images minus the readout time). The setup is the same as in the measurements with the DVS, and a TM6710 with a Schneider 8 mm objective was positioned at various distances in the wind tunnel entrance window.

The wind tunnel was set to a speed of 2 m/s and no model is used, the free flow with the HFSB is recorded. The HFSB have a diameter of $2-3mm$ and are generated at a frequency of 5 Hz. The illumination uses 4 LED pairs (A, C, D, and F) at full power. The LED pair E is excluded because it shone directly into the camera and LED pair B is excluded because it produced reflections on the acrylic window opposite the entrance.

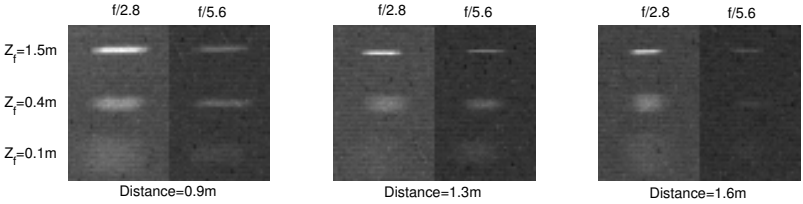


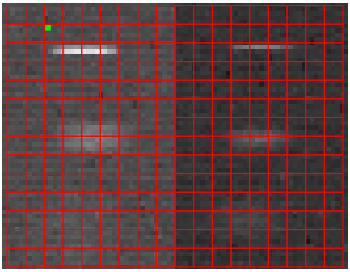
Figure 3.7.: Streak images with TM6710.

In Figure 3.7 an overview of the measurement is shown. The aperture was set to $f/2.8$, $f/5.6$ and $f/11$. The results with $f/11$ showed no discernible bubble images and are neglected here. Using an aperture of $f/1.4$ did not show any improvement in the results with the DVS and was therefore not considered. Z_f is the distance of the focal plane to the camera.

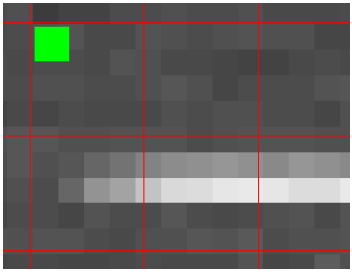
In comparison to the DVS the pixel size of the TM6710 is much smaller at 9 μm. For a visual aid the pixel size of the DVS is overlaid on the images shown in Figure 3.8. The apparent difference shows the resolution mismatch between a particle in focus and the DVS pixel with the light sensitive area in green.

3.4.2. Optical settings

The immediate question before the measurement regards the settings of the optical system, that is the aperture and focus of the objective and the distance of the cameras towards the test section. The distance determines



(a) Comparison to DVS, pixel size in red overlaid on data from TM6710



(b) Close-up of bubble streak, the green area is the size of the light sensitive area of the DVS.

Figure 3.8.: Comparison of pixel size between DVS and TM6710.

for one the amount of light that can be collected by the objective and on the other hand the size of the observable volume. As has been discussed, the particles' image is under–resolved by the resolution of the DVS at typical scales in the wind tunnel and defocussing presents a necessary tool to increase the particles' image size. Using the concept of the circle of confusion (CoC) the minimal requirements for defocussing can be determined. The diameter of the CoC is the finite diameter ϕ of the image of a point source. For ideal optics a point source in focus will produce a point as image as well, optical aberrations however lead to a finite sized image. For a point source out of focus even ideal optics with a finite sized aperture will produce a finite sized image, the relations given in Equations 3.9 and 3.10 are derived from this case. The CoC can be used to determine the focal depth at which the image of a point is indistinguishable from a point, i.e. if ϕ is smaller than the pixel pitch of the sensor. Regarding the DVS with a pixel size of 40 μm we aim to achieve an image with minimally the size of the pixel. Given the diameter of the CoC the far and near point define the range of the focal depth in which the the image is smaller than ϕ.

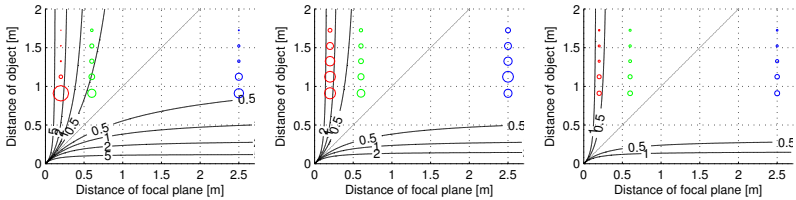
$$\text{Near point:} \quad d_n = \frac{Z(d_h - f)}{(d_h - f) + (Z - f)} \tag{3.9}$$

$$\text{Far point:} \quad d_f = \begin{cases} \frac{Z(d_h - f)}{(d_h - f) - (Z - f)} & \text{if } Z < d_h \\ \infty & \text{if } Z \geq d_h \end{cases} \tag{3.10}$$

with the hyperfocal distance $d_h = f\left(\frac{f}{\phi f/\#} + 1\right)$, focal length f, f–number $f/\#$ and distance between camera and object Z.

The same parameter study as with the TM6710 was performed with a single DVS with a Schneider 8 mm objective. The distances of the camera to the HFSB ranged from 0.9 - 1.8 m. At a distance of 0.9 m a pixel views an area of $4.5 \times 4.5\ mm^2$, covering about the twice the size of a single bubble.

An overview of the study is shown in Figure 3.9 with circles placed according to the distance of the HFSB (Z) and the distance of the focal plane (Z_f) to the camera. The diameter of the circle is proportional to the SNR value. The iso–contour lines represent the near and far point for a given diameter of the CoC. An important aspect was the determination of the distance of the focal plane. For the TM6710 this can be done in a straightforward manner. However for the DVS this is not easy, it became apparent that the values derived for the TM6710 do not apply, this is possibly be due to a variation in back focal length. The distance of the focal plane for the DVS is an approximation at best, derived by looking at the edges of a moving target. Especially the estimation of 2.5 m is quite inaccurate, the value should be higher, maybe even already focused at infinity. For a comparison however a reasonable value for graphical display was chosen.

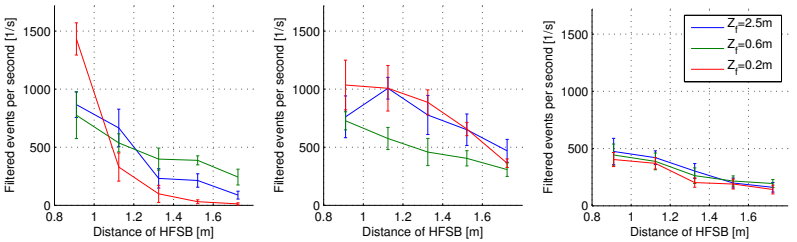


(a) $f/2.8$ **(b)** $f/5.6$ **(c)** $f/11$

Figure 3.9.: The lines show iso–contours of the diameter of the CoC (in pixel units). The DVS measurements from Figure 3.11 are plotted as circles with corresponding colours. The circle diameters are proportional to the total number of events after noise filtering.

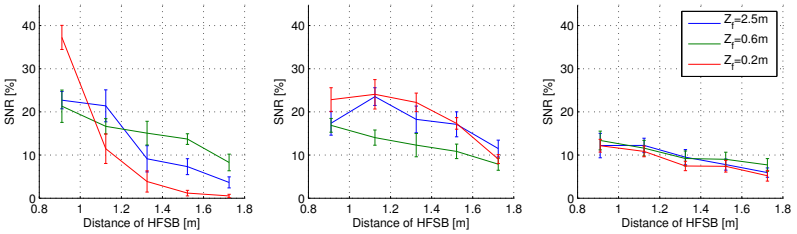
In Figure 3.10 the total number of events after noise filtering is shown. The trends are the same given in Figure 3.11, where the SNR is depicted.

The best results are achieved at a low aperture and cameras close to the HFSB. For practical use the aperture can be set to a higher value as this gives an even SNR over a larger range, as can be seen for $f/5.6$. However, for higher stops the results decline, apparent for $f/11$. Defocussing is necessary for good results, measurements near the focal plane show a decline in quality.



(a) $f/2.8$ **(b)** $f/5.6$ **(c)** $f/11$

Figure 3.10.: Total number of events after noise filtering for apertures $f/2.8$, $f/5.6$ and $f/11$ in function of distance of objects (Z) and focal plane (Z_f) to camera.

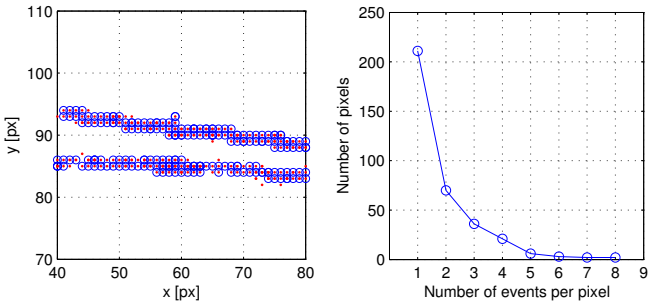


(a) $f/2.8$ **(b)** $f/5.6$ **(c)** $f/11$

Figure 3.11.: Signal-noise-ratio according to Equation 5.1 for apertures $f/2.8$, $f/5.6$ and $f/11$ in function of distance of objects (Z) and focal plane (Z_f) to camera.

A detailed picture of the tracks with the highest SNR at $f/2.8$ and $Z_f = 0.2m$ is shown in Figure 3.12. To filter out the noise a simple neighbour filter was sufficient. The track has an approximate width of 2 pixels. The event count per pixel is as high as 8 events, indicating a high temporal contrast. There are no gaps apparent and the track is continuous throughout the length of the observed volume.

The conclusion from this section was to place the DVS cameras nearer to the HFSB. The measurement volume is reduced, but the data proves to be much more reliable. The initial approach of placing the DVS camera in the top windows of the wind tunnel always suffered from a low particle visibility and was eventually abandoned. Other problems were poor contrast for



(a) Extracted track after filtering, width at 2 pixels

(b) Event count per pixel for filtered track.

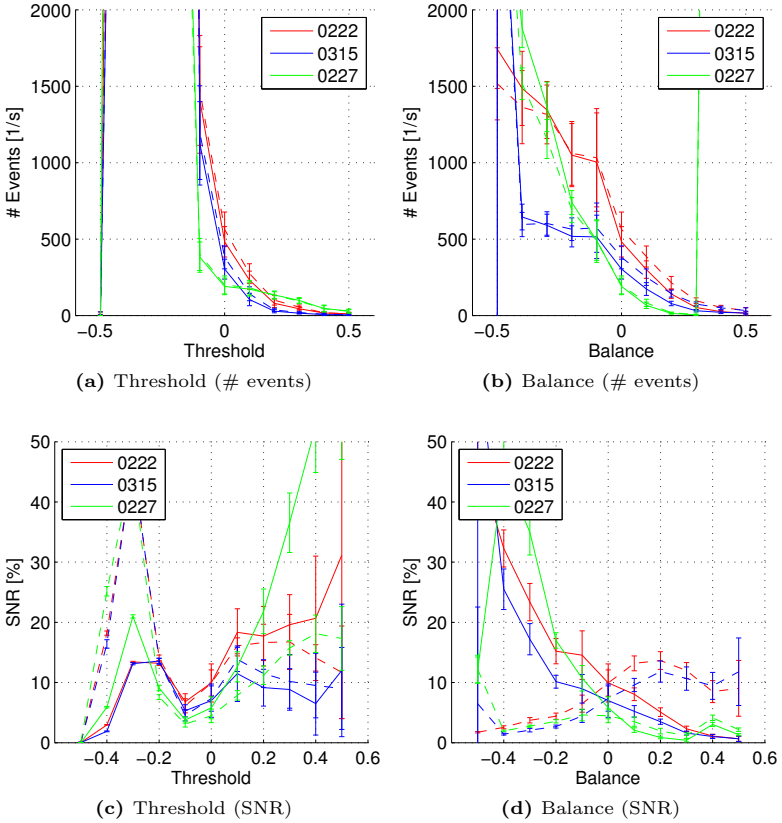
Figure 3.12.: Track with highest SNR.

measurements near the wind tunnel floor and the test object, placing the cameras on the side and closer to the test object reduced these effects as well.

This study is a guideline for the initial optical settings, for each measurement however the settings need to be adjusted for optimal results. This is achieved by running the bubble generator and adjusting the objective by visual inspection of the streamed data. At the moment there are no on–the–fly quality criteria implemented to check the quality of the measurement, using the calculation of the SNR might however be a reasonable solution.

3.4.3. Camera bias

Adjusting the camera biases did not show a great improvement on the particle visibility. For completion an overview of a parameter study is shown in Figure 3.13. The measurement is the same as described in Section 3.4.1. Here all three camera at equal optical settings are used to check for differences. The aperture is set to $f/5.6$ to minimize effects due to focusing.



(a) Threshold (# events)

(b) Balance (# events)

(c) Threshold (SNR)

(d) Balance (SNR)

Figure 3.13.: Variation of bias settings with $f/5.6$, full lines are ON events, dashed lines are OFF events.

A complete overview for apertures at $f/2.8$ and $f/5.6$ is shown in Appendix E.

Some values produced excessive event generation. The only settings showing a strong influence are the threshold and the balance. Raising the threshold did increase the SNR, but not consistently. Lowering the balance produces a higher event generation of ON events with a higher SNR. Otherwise the nominal settings proved to be a good set point.

The individual DVS cameras did show some variation. It is not clear

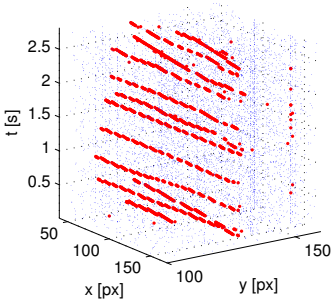
what the reason is. Ensuring an equal focus is rather difficult and has an influence even at $f/5.6$.

3.5. Outlook

Currently a successor to the DVS is being developed at the Institute of Neuroinformatics with a higher pixel resolution of 240×180 and a fill factor of 23%. Featuring an integrated analog–to–digital converter it has the image capturing capabilities of a standard frame–based camera.

A test was conducted with a prototype, measuring the flow of HSFB in the wind tunnel. The camera was placed in a top window, for the DVS this was not an ideal position, however the new camera managed to capture the HFSB without a lot of tuning.

(a) Image of bubble generator and HFSB (left part of image is missing due to error in the chip production).



(b) Extracted tracks as red dots, small blue dots are noise.

Figure 3.14.: Performance of the successor to the DVS

Just regarding the region around the bubble tracks, a SNR of 0.38 was calculated, indicating a better performance than the current "production" DVS.

Chapter 4

Calibration

4.1. Introduction

Measuring the position and velocity of a particle requires knowledge of the relation of the optical imaging system to the environment of the measurement. For a three dimensional reconstruction at least 2 cameras are required, more cameras offer the advantage of redundant information, useful in cases of occlusion or impaired vision from lighting conditions or visibility issues (e.g. discussed in Chapter 3). A stereo setup with two cameras also does not guarantee ambiguity free correspondences; an investigation on the reduction of ambiguity in regard to the number of cameras is given by Maas (1993). In this work three cameras are used as proof of principle, however the number of cameras is not restricted. It is necessary to calibrate the optical system, this is nowadays an established procedure in machine vision. The standard textbooks of reference are Hartley & Zisserman (2003) and Faugeras *et al.* (2001), in which all theoretical aspects of calibration used in this thesis are covered. Basically two calibration methods can be distinguished, the photogrammetric calibration, where image points are associated with known world coordinates, and the self–calibration schemes depending solely on correspondences of points between cameras.

Tsai (1986) introduced the most commonly used calibration method, based on correspondences between image points and a 3D target. It separates the computation of the extrinsic and the intrinsic parameters (including radial distortion) in a two–stage process, without need for initialization.

Macháček (2002) introduced a two–step calibration procedure for a stereo camera setup. In the first step the intrinsic camera parameters are computed by photogrammetric calibration. This can be done outside of the experiment. A black board with a matrix of white dots at precisely spaced locations is used as target and traversed through a volume at defined po-

sitions and the prescribed positions of the white dots are used as input. The second step determines the extrinsic parameters of the cameras in the experimental setup, assuming the established intrinsic parameters do not change by adjusting the focus and aperture. This is achieved by recording a calibration bar with two LEDs separated by a known distance and minimizing the reprojection error.

A simpler calibration procedure was introduced by Svoboda *et al.* (2005), which provides a self-calibration for multiple cameras. A single LED is sufficient to calibrate the scene up to a scale factor. This method uses a projective factorization scheme with Euclidean stratification. The missing scale of the scene can be determined in a subsequent step from the image of a known length.

An established tool for stereo calibration is the Calibration Toolbox from Caltech (Bouguet (2008)), which served as a guideline for the calibration procedure followed in this thesis, but is extended for use with multiple cameras. Using bundle adjustment the whole scene is calibrated in one step, this allows for a fast and easy calibration of each measurement. Acquisition time for a calibration is generally 30 seconds and post-processing the data takes another few minutes, in total calibration takes around 3 minutes. The whole process is automated and no user input is required. A summary of the steps required for calibration can be found in the description of the work flow in Section 2.6.

4.1.1. Additional remarks

Each event of the DVS is recorded as a 32 bit address. A recording is saved as a sequence of events in a file with the ending `*.aedat`. The pixel identification is encoded with 15 bits in total. The rest is reserved for the timestamp and the polarity. The conversion of the AE–data to an image coordinate is given as follows.

$$\begin{aligned} x_{dvs} &= 127 - d_{2-8} \\ y_{dvs} &= d_{9-15} \end{aligned} \tag{4.1}$$

d is the corresponding decimal value of a fragment of the 32 bit address and the subscript denotes the range of the respective bits involved. x_{dvs} and y_{dvs} correspond to the standard graphical representation. With this definition the direction of z_{dvs} points in the negative viewing direction considering a right handed coordinate system. This is contrary to the convention for the pinhole camera given in Section 4.2.1. For the calibration and post–processing we resort to a right hand system by exchanging x_{dvs} and y_{dvs}. The definition given in Equation 4.1 is used for graphical representation.

4.2. Multiple view geometry

Following is an overview of the theory of projective geometry with its application to optical systems. The basis of projective transformations is the homogeneous representation of points. By adding an additional coordinate (the homogeneous coordinate) points are written as $[x,\ y,\ w]^T$, this is by convention equivalent to the point $[x/w,\ y/w,\ 1]^T$. Homogeneous points are invariant to multiplication. The usefulness of this representation becomes apparent in the next Section.

4.2.1. Pinhole camera model

The convention for the camera coordinate system denotes Z as depth and X and Y as tangential to the image plane x and y. The camera center coincides with the coordinate origin, at the point O in Figure 4.1.

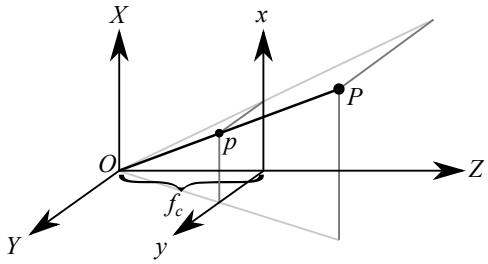


Figure 4.1.: Projection for a pinhole camera. P in the camera coordinate system XYZ is projected onto the point p in the image plane xy at the focal length f_c.

Any point along the ray O-P is projected onto point p in the image plane at focal depth f_c. The relation for the image point is simply

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f_c X/Z \\ f_c Y/Z \end{bmatrix}. \tag{4.2}$$

The projection equation relates a coordinate in $\mathbb{R}^3$ to the pixel position of the imaged point in $\mathbb{R}^2$. The intrinsic parameters for a general camera are the focal lengths f_x, f_y, the principal point c_x, c_y and the skew α. The focal lengths include information of the pixel size; they are related to the lens focal length f_c by $f_c = f_x\Delta_x = f_y\Delta_y$ with pixel pitch Δ_x and Δ_y in the respective dimension. The principal point is the point at which the line of sight is normal to the image plane. The skew factor α is included for the case of non-orthogonal pixel rows and columns, for modern cameras the skew can safely be set to zero. The projection for a general camera is

given in explicit formulation with Equation 4.3.

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} (f_x X + \alpha f_x Y + c_x)/Z \\ (f_y Y + c_y)/Z \end{bmatrix}. \tag{4.3}$$

The relation can be arranged in matrix form with the calibration matrix $\mathbf{K}$, which contains all intrinsic parameters of the pinhole camera and makes use of the homogeneous coordinate representation.

$$\mathbf{K} = \begin{bmatrix} f_x & \alpha f_x & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{4.4}$$

The extrinsic parameters describe the pose of the camera, relating the camera coordinate system to the world coordinate system. This is defined by a Euclidean transformation with 6 degrees of freedom. The extrinsic parameters are given by the rotation matrix $\mathbf{R}$, describing the orientation of the camera, and the camera center in world coordinates $\mathbf{T}$. The camera center in camera coordinates is $\mathbf{t} = -\mathbf{R}\mathbf{T}$. The transformation between world and camera coordinates is given as

$$\mathbf{X}_c = \mathbf{R}\mathbf{X} + \mathbf{t}. \tag{4.5}$$

The resulting projection can be written in homogeneous formulation as

$$\mathbf{x}_p = \mathbf{K}\,[\mathbf{R},\ \mathbf{t}]\,\mathbf{X}. \tag{4.6}$$

The combination of all matrices is called the camera matrix and contains all intrinsic and extrinsic parameter with a total of 11 degrees of freedom.

$$\mathbf{P} = \mathbf{K}\,[\mathbf{R},\ \mathbf{t}] \tag{4.7}$$

4.2.2. Distortion

Real optical systems deviate from the pinhole model. This can be caused by a dependence of the magnification on the distance to the optical axis (radial distortion), by physical elements in a lens not being perfectly aligned (tangential distortion) or other imperfections of the lens system. The deviation can be corrected by a distortion model to achieve an accuracy up to pixel level. The most commonly used distortion model was introduced by Brown (1966), which treats the radial and tangential distortion as a polynomial function of the undistorted image points originating from the pinhole model. Other distortion models only account for the radial distortion, which in general is the dominant component. Wang *et al.* (2009) introduced a distortion function based on the distorted points. The advantage is a more robust estimation of the distortion center and it additionally

leads to an explicit formulation for calculating the undistorted image coordinates.

The Schneider Kreuznach 8 *mm* lenses show very low distortion, with a maximum for the DVS on the order of 3 pixels. The calibration procedure using the model by Brown is robust and shows good results (see Section 4.4). Following is a summary of the distortion model.

For a calibrated camera the calibration matrix is equal to the identity matrix, $\mathbf{K} = \mathbf{I}$. This gives the normalized coordinates,

$$\mathbf{x}_n = [\mathbf{R},\ \mathbf{t}]\mathbf{X}, \tag{4.8}$$

which only depend on the extrinsic parameters.
The distortion is calculated from the normalized coordinates as

$$\mathbf{x}_d = (1 + k_1 r_n^2 + k_2 r_n^4 + k_5 r_n^6)\mathbf{x}_n + \mathbf{dx} \tag{4.9}$$

$$\mathbf{dx} = \begin{bmatrix} 2k_3 x_n y_n + k_4(r_n^2 + 2x_n^2) \\ k_3(r_n^2 + 2y_n^2) + 2k_4 x_n y_n \end{bmatrix} \tag{4.10}$$

where $r_n = \sqrt{x_n^2 + y_n^2}$ is the distance to the optical axis.

The first term on the left hand side of Equation 4.9 is the radial distortion, Equation 4.10 is the tangential distortion. The distortion is described by the parameters k_{1-5} (defined in the same manner as Bouguet (2008)).

The pixel coordinates are retrieved by applying the calibration matrix $\mathbf{K}$.

$$\mathbf{x}_p = \mathbf{K}\mathbf{x}_d \tag{4.11}$$

Normalized coordinates are useful due to the independence of the intrinsic parameters (including the distortion parameters). The camera matrix is then reduced to the calibrated camera matrix $\mathbf{P} = [\mathbf{R},\ \mathbf{t}]$. Once the intrinsic parameters are known, the normalized coordinates can be calculated by solving Equations 4.9 and 4.10. An iterative solution is given in the Calibration Toolbox by Bouguet (2008) (with the Matlab function `normalize.m`) and was used in this work.

4.2.3. Fundamental matrix

To introduce the multi–camera approach used in this work, first the basics of a stereo camera setup are considered. For stereo cameras image points of the same, but unknown, point in $\mathbb{R}^3$ are constrained by the concept of the fundamental matrix. A short excursion into the basics is given as follows.

Using homogeneous notation a point in $\mathbb{R}^2$ is $\mathbf{x} = [x,\ y,\ 1]^T$. Equally a line can defined as $\mathbf{l} = [a,\ b,\ c]^T$. Point $\mathbf{x}$ coincides with line $\mathbf{l}$ if

$$\mathbf{x}^T\mathbf{l} = ax + by + c = 0. \tag{4.12}$$

This is simply the implicit equation for a line. The same holds for $(ka)x + (kb)y + (kc) = 0$ with a constant factor k. Points and lines in homogeneous coordinates are invariant to multiplication.

A few basic properties between lines $\mathbf{l} = [a, b, c]^T$ and points $\mathbf{x} = [x, y, 1]^T$ are summarized in Table 4.1.

Table 4.1.: Relations between points and lines in $\mathbb{R}^2$ in homogeneous representation.

Intersection of two lines	$\mathbf{x} = \mathbf{l}_1 \times \mathbf{l}_2$
Line through two points	$\mathbf{l} = \mathbf{x}_1 \times \mathbf{x}_2$
Distance between $\mathbf{x}$ and $\mathbf{l}$	$d = \left(\mathbf{l}^T\mathbf{x}\right) / \sqrt{a^2 + b^2}$
Nearest point to $\mathbf{x}$ on $\mathbf{l}$	$\mathbf{x}_l = \mathbf{x} - \left(\mathbf{l}^T\mathbf{x}\right) / \left(a^2 + b^2\right) [a,\ b,\ 0]^T$

The distance d between $\mathbf{l}$ and $\mathbf{x}$ can have negative values based on which side $\mathbf{x}$ is situated (positive if $\mathbf{x}$ is in direction of the normal $[a,\ b,\ 0]^T$). The scaling with $1/\sqrt{a^2 + b^2}$ is necessary to retrieve the Euclidean distance with the same units as x and y.

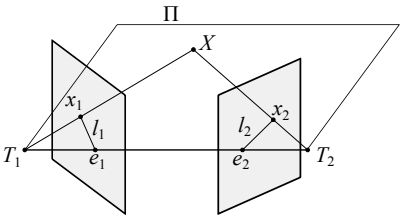


Figure 4.2.: Epipolar geometry

In the situation of two cameras viewing point $\mathbf{X}$, the image points $\mathbf{x}_1$ and $\mathbf{x}_2$ for each respective camera, the camera centers $\mathbf{T}_1$ and $\mathbf{T}_2$, and the point $\mathbf{X}$ lie in the same plane, denoted as Π. This is the only geometrical relation between elements of the two views. Given image point $\mathbf{x}_1$ in the first camera, one can define the epipolar line $\mathbf{l}_2$ as the projection of the ray going through the camera center $\mathbf{T}_1$ and image point $\mathbf{x}_1$ (the line of sight of point $\mathbf{x}_1$) onto the second camera. In other words, the epipolar

line is the intersection between the image plane and plane Π. The epipolar point ($\mathbf{e}_1$, $\mathbf{e}_2$) is the intersection point of all possible epipolar lines, this is the intersection between the image plane and the line connecting the two camera centers.

The fundamental matrix describes the mapping between point $\mathbf{x}_1$ and epipolar line $\mathbf{l}_2$.

$$\mathbf{l}_2 = \mathbf{F}\mathbf{x}_1 \tag{4.13}$$

Point $\mathbf{x}_2$ has to lie on $\mathbf{l}_2$ for a matching point pair, from this the fundamental matrix constraint is derived.

$$\mathbf{x}_2^T\mathbf{l}_2 = \mathbf{x}_2^T\mathbf{F}\mathbf{x}_1 = 0 \tag{4.14}$$

The inverse relation from camera 2 to camera 1 is simply given by

$$\mathbf{x}_1^T\mathbf{F}^T\mathbf{x}_2 = 0. \tag{4.15}$$

The fundamental matrix is only directly applicable for a linear projection relation, i.e. for the pinhole camera. Distortion is not accounted for. Rewriting Equation 4.14 for normalized coordinates results in the essential matrix. The advantage is independence from intrinsic parameters, especially the distortion parameters.

$$\mathbf{x}_{n,2}^T\mathbf{E}\mathbf{x}_{n,1} = 0 \tag{4.16}$$

For two pinhole cameras the fundamental and essential matrices are given as

$$\begin{aligned}\mathbf{E} &= [\mathbf{t}]_\times\mathbf{R} \\ \mathbf{F} &= \mathbf{K}_2^{-T}\mathbf{E}\mathbf{K}_1^{-1},\end{aligned} \tag{4.17}$$

where $[\cdot]_\times$ is the skew symmetric matrix, see Appendix C.2.

The rotation matrix $\mathbf{R}$ and translation vector $\mathbf{t}$ correspond to a coordinate transform between the two camera coordinate systems, given in Equation 4.18.

$$\mathbf{X}_{c,2} = \mathbf{R}\mathbf{X}_{\mathbf{c},\mathbf{1}} + \mathbf{t} \tag{4.18}$$

For arbitrary pose of the two cameras the transform is calculated with Equation 4.19.

$$\begin{aligned}\mathbf{R} &= \mathbf{R}_2\mathbf{R}_1^T \\ \mathbf{t} &= \mathbf{R}_2(\mathbf{T}_1 - \mathbf{T}_2)\end{aligned} \tag{4.19}$$

Regarding the relation for the distance between a point and a line given in Table 4.1 an error measure for the fundamental matrix constraint can be derived in units of $\mathbf{x}_2$ as

$$\epsilon_{epi} = \frac{\mathbf{x}_2^T \mathbf{l}_2}{\sqrt{a_2^2 + b_2^2}}, \quad \text{with} \quad \mathbf{l}_2 = \mathbf{F}\mathbf{x}_1 = [a_2,\ b_2,\ c_2]^T. \tag{4.20}$$

This definition is only valid for distortion–free coordinates. Using the essential matrix formulation is a possible solution, the downside is that the normalized coordinates have no physical unit of length, and interpreting the error would prove to be difficult. One can regard the normalized coordinates as a non–dimensional representation with the corresponding length scale given by the focal length. To convert back to a physical length scale one can simply multiply with the approximate focal length (neglecting distortion and differences in f_x and f_y). This is however not mathematically rigid. Another solution is to use distortion free coordinates in pixel units, based on the normalized coordinates,

$$\tilde{\mathbf{x}}_p = \mathbf{K}\mathbf{x}_n. \tag{4.21}$$

The use of the fundamental matrix equation is accurate with this definition.

4.2.4. Reconstruction

Reconstruction describes the triangulation of an unknown point $\mathbf{X}$ in $\mathbb{R}^3$ from its image points from 2 or more cameras. With known intrinsic and extrinsic parameters the position of the point $\mathbf{X}$ can be reconstructed using the projection equation $\mathbf{x}_m = \mathbf{P}_m\mathbf{X}$, where subscript $(\cdot)_m$ denotes the respective camera. Taking the cross product of the right and left hand side of the equation gets rid of the homogeneous scale factor. This leads to 3 equations for each camera, of which only 2 are linearly independent. Equation 4.22 is the equation system for one camera,

$$\mathbf{A}_m\mathbf{X} = 0 \quad \text{with} \quad \mathbf{A}_m = \begin{bmatrix} x_m\mathbf{p}_m^{3T} - \mathbf{p}_m^{1T} \\ y_m\mathbf{p}_m^{3T} - \mathbf{p}_m^{2T} \end{bmatrix} \tag{4.22}$$

where the camera matrix is split into its rows,

$$\mathbf{P}_m = \begin{bmatrix} \mathbf{p}_m^{1T} \\ \mathbf{p}_m^{2T} \\ \mathbf{p}_m^{3T} \end{bmatrix}. \tag{4.23}$$

The solution for $\mathbf{X}$ (4 unknowns) requires input from at least 2 cameras to build a matrix with rank 4. Obviously this has to be the case, since at least

2 cameras at different positions have to provide information to triangulate a point in 3 dimensional space. This approach is easily adaptable for multiple cameras by concatenating the matrices for each camera.

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \vdots \\ \mathbf{A}_M \end{bmatrix} \tag{4.24}$$

The solution to this system is discussed in Appendix C.3.

4.2.5. Error definitions

The reconstruction error ϵ_{tri} is the vector with the smallest distance of the reconstructed point $\tilde{\mathbf{X}}$ to the line–of–sight of the normalized image point $\mathbf{x}_n$. Equation 4.25 gives the error in the respective camera coordinate system with $\tilde{\mathbf{X}}_c = [\mathbf{R},\ \mathbf{t}]\tilde{\mathbf{X}}$. The derivation of the error is given in Appendix B.4.

$$\epsilon_{tri} = \tilde{\mathbf{X}}_c - \frac{\mathbf{x}_n^T \tilde{\mathbf{X}}_c}{\mathbf{x}_n^T \mathbf{x}_n} \mathbf{x}_n \tag{4.25}$$

The scalar product of $\mathbf{x}_n$ and $\mathbf{X}_c$ requires $\mathbf{x}_n$ to be given in homogeneous formulation and $\mathbf{X}_c$ in inhomogeneous formulation.
The reprojection error ϵ_{rp} is given as the difference between the extracted image point $\mathbf{x}_p$ and the backward projection $\tilde{\mathbf{x}}_p$ of the reconstructed point $\tilde{\mathbf{X}}$.

$$\epsilon_{rp} = \tilde{\mathbf{x}}_p - \mathbf{x}_p \tag{4.26}$$

4.2.6. Accuracy

Using the linearized uncertainty estimation for the triangulation, derived in Appendix B.5, the accuracy of the reconstruction can be calculated based on the accuracy with which the position of a particle can be determined in an image. Assuming the error of the image position to be normally distributed with the standard deviation σ_x, the standard deviation of the error of the reconstructed position ($\sigma_{X,Y,Z}$) is calculated. Two idealized setups are considered, with 2 and 3 cameras.

For a stereo camera setup, the standard deviation of the reconstruction error is given by Equation 4.27.

$$\sigma_X = \frac{L}{f_c\sqrt{2}}\sigma_x, \quad \sigma_Y = \frac{L}{f_c\sqrt{2}\cos(\alpha)}\sigma_x, \quad \sigma_Z = \frac{L}{f_c\sqrt{2}\sin(\alpha)}\sigma_x \tag{4.27}$$

f_c is the focal length, L is the distance of the particle to the cameras, which are positioned in the Y-Z plane, on either side and symmetric to the

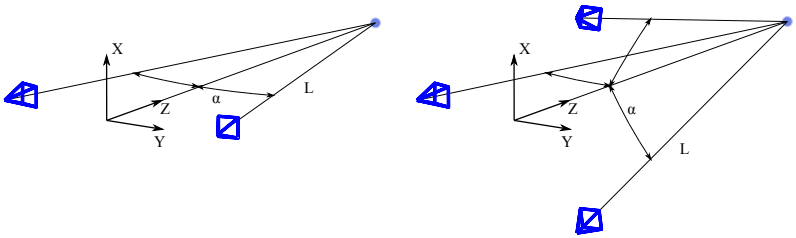


Figure 4.3.: Camera placement with 2 and 3 cameras.

Z axis, and α is the angle between camera and Z axis, see Figure 4.3. The optimal angle is at 45° (90° angle between cameras); at this angle σ_Y is equal to σ_Z.

For a setup with 3 cameras, equally distributed around the Z axis at an angle of α, the distribution of the positional error is given by

$$\sigma_X = \sigma_Y = \frac{L}{f_c}\sqrt{\frac{2}{3(\cos^2(\alpha)+1)}}\sigma_x, \quad \sigma_Z = \frac{L}{f_c\sqrt{3}\sin(\alpha)}\sigma_x. \tag{4.28}$$

The accuracy in Z is slightly better by a factor of $\sqrt{2/3} = 0.817$. The optimal angle with the lowest overall deviation (least squares solution) is given by $\alpha = \arccos(1/\sqrt{3}) = 54.7°$, the angle between two individual cameras is again 90°.

Based on the experimental setup, approximate values for the distance and angle are, respectively, $1.5\,m$ and 15°. Assuming we can determine the image position with a 1 pixel accuracy, σ_x can be set to $0.25px$. That means the image position lies with 95,4% certainty inside the range of $x \pm 2\sigma_x$, the accuracy is therefore about four times the standard deviation. Table 4.2 shows the results for the standard deviation of the reconstruction error.

Table 4.2.: Theoretical accuracy of the reconstruction using multiple cameras.

	$4\sigma_X$	$4\sigma_Y$	$4\sigma_Z$
2 cameras	$5.3\,mm$	$5.5\,mm$	$20.5\,mm$
3 cameras	$4.4\,mm$	$4.4\,mm$	$16.7\,mm$

Regarding the projection equation given by $x = \frac{X}{L}f_c$, a change in image position of $\Delta x = 1\,px$ results in a length difference of $\Delta X = 7.5\,mm$, about

6 times the standard deviation in X. Overall, the accuracy can be increased by using multiple cameras.

4.3. Calibration procedure

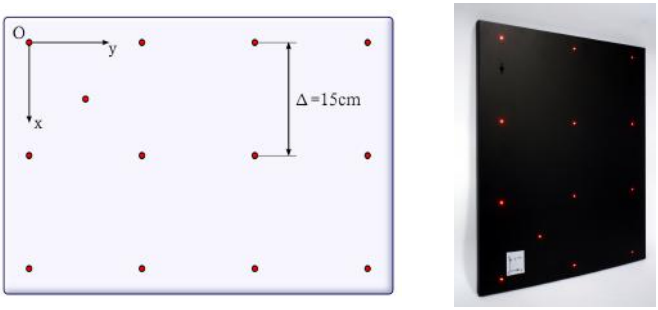
The first step of the calibration is to establish some sort of correspondence between cameras. The self–calibration uses correspondences of image points between cameras, $\mathbf{x}_1 \leftrightarrow \mathbf{x}_2$. As target any object can be used, such as a single source point or a calibration bar with known length. Photogrammetric calibration uses correspondences between image points and a target with known structure, $\mathbf{x}_m \leftrightarrow \mathbf{X}_{target}$. This approach was chosen because it is straightforward to implement and also allows to use the target as a reference coordinate system. The whole calibration can be achieved with the same target in one step. The main restriction lies in the size of the calibration target, which has to be large enough to produce images with distinguishable features and reasonably small to handle in the wind tunnel environment.

The calibration procedure is based on acquiring the image points of the target from multiple views for multiple cameras. Generally the target for frame–based cameras is a checkerboard pattern, for the DVS this is not a reasonable choice. Instead a LED board with 13 SMD LEDs is used The LEDs produce a light spot of $2\,mm$ in diameter, suitable for the DVS. Each LED is driven by a pulsed current with a frequency of 200 Hz to enhance visibility for the DVS cameras. The 13 LEDs are arranged in a rectangular pattern of 3×4 LEDs distanced at $15\,cm$, with the last LED in the center of the first quadrant, this ensures a unambiguous recognition of the board from each view. Every LED is assigned a coordinate in $\mathbb{R}^3$, shown in Figure 4.4. The Z coordinate is simply set to zero. This convention defines the target centric coordinate system.

The target acquisition for the calibration procedure can be summarized as follows:

- Position the cameras. It is recommended to check the visibility of the HFSB first and adjust the objectives accordingly (see Section 3.4.2 for guidelines on the optical adjustment).
- Dim the illumination to enhance the visibility of the target.
- The calibration target can also be used to set the world coordinate system. For this the target can be placed in the required position at the start of the data acquisition. The optimization will provide the pose of the cameras in the target reference frame of all views of the target, hence the pose of the first view can be used for the camera positions in world coordinates.

- Start the acquisition. Move the target through the test section. It does not matter if the target is not seen by all cameras. For the calibration it is sufficient to have pair–wise views (i.e. at least two cameras fully recognize the target at the same time instant).



(a) Target coordinate system.

(b) Calibration target.

Figure 4.4.: Calibration target with convention of target coordinate system and center of origin (O).

It is not necessary for all cameras to recognize the target in each view, missing data is complemented by the code. Even single views are used, the information flows into the optimization of the intrinsic parameters of the respective camera.

4.3.1. Target matching

An image based process is used to determine the LED positions. In a first step an image is reconstructed from the events recorded in a given sampling interval. The first approach for image reconstruction is to simply sum up events for each pixel. This works fairly well, especially since the LEDs are operated at a moderately high frequency of 200 Hz, thus showing up on the DVS sensor as a high event count. A better approach is use a lock–in filter, since the operating frequency of the LEDs is prescribed and can be directly inserted into the continuous summation of the events. The filter has been implemented in Matlab for post–processing purposes. The lock–in

equations for a sample of discrete events are given as follows.

$$\begin{aligned} R &= \sum_i p_i \cos(\omega t_i) \\ Q &= \sum_i p_i \sin(\omega t_i) \\ S &= \sqrt{R^2 + Q^2} \end{aligned} \tag{4.29}$$

where p_i is the event polarity and t_i is the time stamp.
Using a lock–in approach the amplitude S is enhanced for each pixel viewing an intensity signal with the appropriate frequency and is reduced for signals with arbitrary frequency (e.g. noise). The phase of the driving signal is not required but may be calculated as well according to Equation 4.30.

$$\tan(\phi) = \frac{Q}{R} \tag{4.30}$$

An example for the reconstructed image is shown in Figure 4.5. The sampling interval is $\Delta t = 50\,ms$. With a driving frequency of $200\,Hz$ this equals 10 periods or in other words 10 ON/OFF cycles.

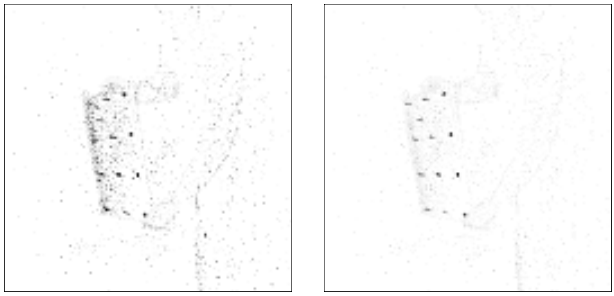


Figure 4.5.: Reconstructed image from events. Left: Sum over events. Right: Signal S of the lock–in filter.

The amplitude S shows a clearer picture of the LED positions. The movement of the board produces a high event count but is reduced by the lock–in filter. A closer picture is given in Figure 4.6. The summation of events shows no discernible peaks, contrary to the filtered image.

Extracting the positions with sub–pixel accuracy is achieved via a Gaussian fit. The approach is motivated by the CLEAN algorithm used for determining the location of objects in stellar images in radio astronomy, see Högbom (1974). We use an iterative procedure, which looks for the pixel

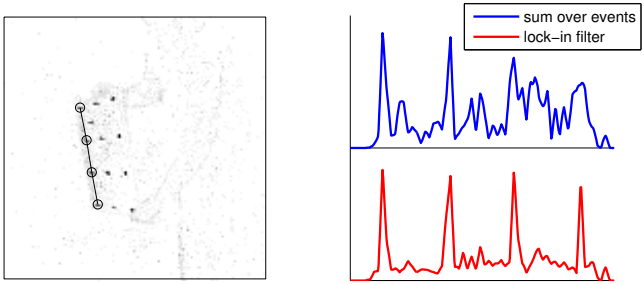


Figure 4.6.: Lock–in filter with noisy data.

with the highest value and fits a Gaussian function within the neighbourhood of the maximum. As a result the center of the Gaussian is retrieved as the position of the LED. The evaluated Gaussian fit is subsequently subtracted, from the resulting image the next pixel with the highest value is again selected for fitting. This is repeated for the number of positions in question. Due to the noisy data[1], a standard PIV sub–pixel interpolation scheme (e.g. see Honkanen & Nobach (2005)) works poorly. Instead a non-linear fit is used (non-linear due to the use of an offset value). The fit incorporates the values from of a window of 13×13 pixels around the maximum. The larger window size reduces the influence of noise compared to the standard sub pixel interpolation scheme with a window of 3×3. The Gaussian function used is given in Equation 4.31.

$$I(x,y) = I_0 + Ae^{-\frac{(x-x_0)^2}{2\sigma_x^2} - \frac{(y-y_0)^2}{2\sigma_y^2}} \tag{4.31}$$

The matching procedure tries to fit and associate the given target coordinates to the extracted positions in the image. This is achieved by computing the hull of the extracted positions (i.e. the 4 corners of the grid), and computing the homography to the hull of the target coordinates, $\mathbf{x} = \mathbf{H}\mathbf{X}_{target}$. This is done for all possible assignments of target to image coordinates of the hull. Applying the homography to all target positions allows to associate the image points to the nearest projection of the target coordinates. All associations with errors higher than 5 pixels are regarded as a false match. The match with the lowest error is chosen to be correct.

The matching procedure may produce no result in case of occlusion of a LED or a wrong extraction of the LED position. The LED board is

[1] The image reconstruction via the lock–in filter still shows an uneven signal distribution around the peak signal of the LED.

constructed with a 13^{th} LED to ensure the correct orientation of the match.

4.3.2. Initialization

Generally a good initial guess for the principle points is the center of the chip. The focal lengths may be extracted from other constraints, e.g. the Kruppa equations or from the vanishing points of parallel lines. Only one type of camera and objective is used in this work, the intrinsic parameters are relatively well known and are provided as initial guess. The distortion parameters are initially set to zero.

Once the intrinsic parameters are given, the normalized coordinates can be calculated. With the established correspondence between image and target point, the extrinsic parameters for each view of the target can be estimated by linear methods. By using a planar target, a homography between $\mathbf{x}_n$ and $\mathbf{X}_{target}$ can be computed.

$$\mathbf{x}_n = [\mathbf{R},\ \mathbf{t}][X,\ Y,\ 0,\ 1]^T = \mathbf{H}[X,\ Y,\ 1]^T \tag{4.32}$$

The homography can be calculated by a standard DLT (direct linear transformation) method, more details can be found in Hartley & Zisserman (2003). The first two columns of the rotation matrix $\mathbf{R}$ are directly obtained as the first two columns of $\mathbf{H}$. However, it is necessary to scale to unity and correct for orthogonality. The third column is simply the cross product of the first two columns. The translation $\mathbf{t}$ is obtained by the third column. This step makes use of the Calibration Toolbox function, `compute_extrinsic.m`, which has a robust implementation of the algorithm and even a separate solution algorithm for non-planar targets.

4.3.3. Optimization

The main calibration step uses a non-linear optimization of the reprojection error given in Equation 4.26. The aim is to find a consistent set of intrinsic and extrinsic parameters for which the projection of a triangulated point in $\mathbb{R}^3$ matches the original image point within a reasonable tolerance. The input to the calibration are the extracted LED positions from an arbitrary number of cameras and views of the target. All variables that are to be optimized are organized into a vector $\boldsymbol{\phi}$. These are the focal lengths $\mathbf{f}_m$, principal points $\mathbf{c}_m$, distortion coefficients $\mathbf{k}_m$, the rotation vectors $\boldsymbol{\omega}_{c,m}$ and $\boldsymbol{\omega}_{t,mk}$ (given with the Rodrigues formulation, see Appendix C. This directly corresponds to the rotation matrix $\mathbf{R}$.) and the camera centers $\mathbf{t}_{c,m}$ and $\mathbf{t}_{t,mk}$.

$$\boldsymbol{\phi} = [\mathbf{f}_m,\ \mathbf{c}_m,\ \mathbf{k}_m,\ \boldsymbol{\omega}_{c,m},\ \mathbf{t}_{c,m},\ \boldsymbol{\omega}_{t,mk},\ \mathbf{t}_{t,mk}] \tag{4.33}$$

The extrinsic parameters are based on the condition that the cameras have a fixed relative position towards each other, given by the camera pose

$\boldsymbol{\omega}_{c,m}$ and $\mathbf{t}_{c,m}$. The pose of the camera towards the target is given by $\boldsymbol{\omega}_{t,mk}$ and $\mathbf{t}_{t,mk}$ in the target centric coordinate system. The subscripts $(\cdot)_m$ and $(\cdot)_k$ denote, respectively, the index of the camera and of the view. It is only necessary to calculate the pose of the first camera towards the target. The pose of the other cameras in the target reference frame are given by

$$\begin{aligned} \mathbf{R}_{t,mk} &= \mathbf{R}_{c,m}\mathbf{R}_{c,1}^{T}\mathbf{R}_{t,1k} \\ \mathbf{t}_{t,mk} &= \mathbf{t}_{c,m} + \mathbf{R}_{c,m}\mathbf{R}_{c,1}^{T}(\mathbf{t}_{t,1k} - \mathbf{t}_{c,1}) \end{aligned} \tag{4.34}$$

The predefined target coordinates (see Figure 4.4) are retrieved by defining the calibrated camera $\mathbf{P}_{mk}$ matrix in the target reference frame. The resulting projection equation is

$$\tilde{\mathbf{x}}_{n,mk} = \mathbf{P}_{mk}\mathbf{X}_{target} = [\mathbf{R}_{t,mk},\ \mathbf{t}_{t,mk}]\mathbf{X}_{target}, \tag{4.35}$$

where $\tilde{\mathbf{x}}_{n,mk}$ are the reprojected normalized points of the target. An advantage of this procedure is that a reference coordinate frame can be directly set by placing the target in the desired position at the start of the calibration acquisition. This approach needs to have one camera pose fixed. The pose of the other cameras is defined relative to the pose of this camera. In this work the pose of the first camera is set to $\mathbf{R}_{c,1} = \mathbf{I}$ and $\mathbf{t}_{c,1} = \mathbf{0}$. Alternatively, the pose of the first camera can be set manually by the user.

In total the number of parameters that have to be optimized for M cameras and K views is given as

Intrinsic parameters	$9M$
Extrinsic parameters for cameras	$6(M-1)$
Extrinsic parameters for target views	$6K$
Total number of parameters	$9M + 6(M-1) + 6K$

In compact form the projection equations and the reprojection error are given as

$$\tilde{\mathbf{x}}_p = f(\boldsymbol{\phi}, \mathbf{x}_p), \quad \boldsymbol{\epsilon}_{rp} = \tilde{\mathbf{x}}_p - \mathbf{x}_p. \tag{4.36}$$

Optimization seeks the vector $\boldsymbol{\phi}$ which minimizes the reprojection error in a least squares sense. The optimization uses the Levenberg–Marquard algorithm. An initial guess for vector $\boldsymbol{\phi}$ is provided through prior knowledge of the optical system and the linear methods described in Section 4.3.2. The algorithm makes use of the Jacobi matrix, given by

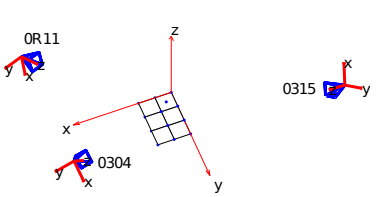
$$\mathbf{J} = \frac{\mathrm{d}\boldsymbol{\epsilon}_{rp}}{\mathrm{d}\boldsymbol{\phi}} = \frac{\mathrm{d}f(\boldsymbol{\phi}, \mathbf{x}_p)}{\mathrm{d}\boldsymbol{\phi}}. \tag{4.37}$$

For a more detailed description see Appendix C. The solver used is `lsqnonlin.m` from Matlab. Optionally, the structure $\mathbf{X}_{target}$ can be optimized as well. This is only necessary if the target is not accurately manufactured, with the positions of the markers subject to deviations from the predefined coordinates.

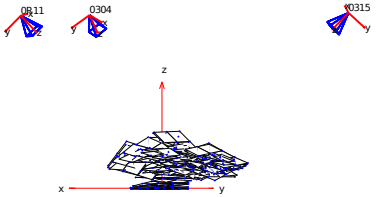
4.4. Results

An example for a calibration with 3 cameras is presented. The acquisition time is 30 s, image reconstruction uses a sampling interval of 50 ms (with a LED driving frequency of 200 Hz the interval includes 10 ON/OFF cycles, sufficient to give a clear picture with any noise strongly reduced). This gives 600 pictures for each camera from which the positions are extracted. Some pictures show occlusion of a LED or a part of the board outside the viewing area. This occlusion results in some missing views.

In total the optimization problem consisted of 14170 reprojection errors with 1644 degrees of freedom. Optimization takes less than 10 seconds on a modern PC.



(a) Centered view with the target board defining the origin of the world coordinate system.



(b) Positions of the target board used in the optimization.

Figure 4.7.: Example for a calibration with 3 DVS cameras in the wind tunnel. The target is used to represent the world coordinate system. Flow direction is in positive y–direction, the 3 cameras are positioned in the top windows of the wind tunnel.

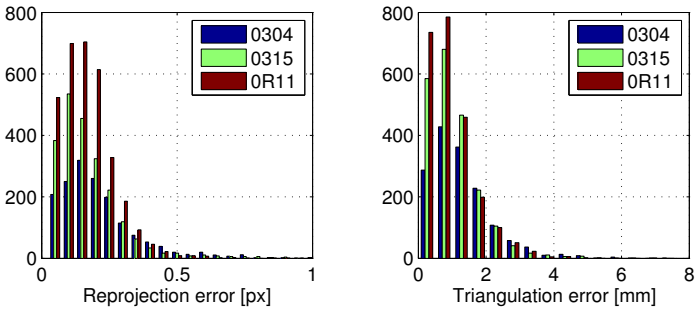


Figure 4.8.: Histogram of errors for a typical calibration of the DVS cameras.

The error distribution is shown in Figure 4.8. Table 4.3 gives the average error for reprojection and reconstruction. The calibration produces an accurate solution with low errors, comparable to the theoretical values given in Section 4.2.6.

Table 4.3.: Average calibration errors of reference measurement.

	DVS 304	DVS 315	DVS 111
Number of views	124	171	250
Reconstruction error	1.28 mm	1.02 mm	0.94 mm
Reprojection error	0.21 px	0.17 px	0.17 px

4.4.1. Convergence of calibration

To check the number of views required for an accurate calibration, a convergence study was conducted. Based on the same data set from Section 4.4 a subsample of views is randomly chosen and calibrated. This is repeated 50 times for each subsample size and the resulting uncertainty is averaged.

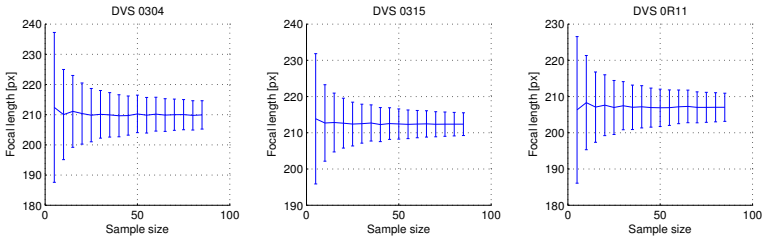


Figure 4.9.: Convergence of focal length.

It appears sufficient to use about 50 images from each camera. Higher sample sizes do not lead to any significant improvement.

4.4.2. Distortion

For a complete picture of the results the distortion is shown as well. The objectives offer a very low and consistent distortion with a maximal distortion of about 2 pixels.

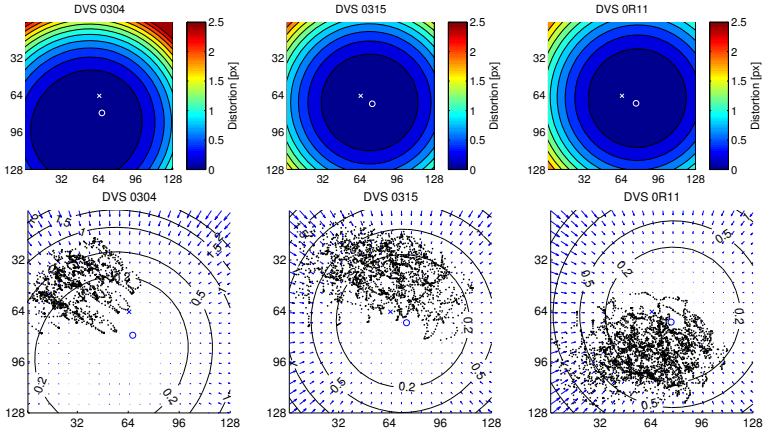


Figure 4.10.: Distortion of the DVS cameras using Schneider Kreuznach 8 *mm* lenses. Each black dot in the lower pictures represents the image position of a LED.

The results show a high distortion in the top corners for camera 304. This might be due to the distribution of the LED image points which are located in a small area of the whole image plane. The calibration has to extrapolate the result for the rest of the image plane; in this region the calibration does not necessarily provide very accurate results. This is however not a drawback since we are generally only interested in the volume which has been scanned by the calibration target. Care has to be taken during calibration to ensure the measurement volume is fully traversed with the calibration target.

The next chapter describes basic considerations and strategies to process the event–driven data for tracking and reconstructing the bubble paths. The Kalman filter will be introduced as a suitable basis for a real–time tracking algorithm.

Chapter 5

Tracking

5.1. Introduction

Established methods in flow velocimetry are often image based (where discernible tracers are used), such as Particle Image Velocimetry, Particle Tracking Velocimetry or Particle Streak Velocimetry. Time resolved measurements require high speed cameras, resulting in high data volumes and high demands for processing power. The advantage of a temporal contrast sensor is the inherent background subtraction and the time–accurate registration of movement. Processing the data requires a different approach than the established methods offer if one wants to take advantage of the nature of the data. It is delivered as a continuous stream, the optimal solution is to use a continuous process. The output of a suitable algorithm is the particle track, from which the Eulerian velocity field can be derived.

The search for solutions to this problem leads to the field of motion tracking. Piatkowska *et al.* (2012) summarizes the methods as point based, feature based, tracking of contours or silhouettes, and stochastic trackers. For further reading see the reference list given in Piatkowska *et al.* (2012).

The most relevant type of trackers for this application are the stochastic trackers, as they use a continuous process to estimate a set of state variables given a reduced set of measurements and some knowledge of the expected noise. The most commonly used type of stochastic tracker is the Kalman filter, based on the assumption that the noise is normally distributed. More complex variations exist, such as particle filters, which make no assumption on the distribution of the noise, but are more demanding to implement as well as computationally more expensive. In this thesis the Kalman filter is used as tracker.

This chapter will first give an overview of previous research, followed by a summary of the filters used in this work. The three main aims of the

filters are noise reduction, feature extraction and tracking, for which a few select solutions will be presented. The main part of the chapter is about the tracking procedure using a Kalman filter. A parameter study with artificially generated data is performed to quantify the performance of the filter.

5.1.1. Previous work

This section is intended to give an overview of the state of the art of tracking techniques used previously in connection with the DVS.

Litzenberger *et al.* (2006) presented a tracking algorithm where events are assigned to circular clusters based on a distance criterion. The position of the object is represented by the center of the circle and is updated with the position of newly associated events. The radius of the cluster is adapted dynamically to the size of the object. The system was demonstrated within a traffic surveillance application.

Conradt *et al.* (2009) used a pair of DVS cameras as input to control a balancing pencil. The algorithm uses a continuous Hough transform to determine the angle and position of the pencil in real time.

Schraml *et al.* (2010) presented a tracking algorithm for a 3D stereo setup, using a pair of DVS cameras to track people within a 4 m range at an effective refresh rate of 200 Hz. The cameras are calibrated and aligned, so that pairs of conjugate epipolar lines are collinear. This reduces the stereo correspondence problem to one dimension, saving computational effort and time. The algorithm computes a disparity between left and right camera based on a similarity metric; in this case the normalized sum of absolute differences [1] was employed. Making use of the address event representation only pixels with spiking events have to be considered in the calculation, avoiding a full cross correlation evaluation across the whole image range. Triangulation delivers the depth information of the object. A cluster algorithm is incorporated for tracking objects by forming a bounding box around groups of events and matching between the size of the box, the number of enclosed events and the distance of center points. For a typical multi-object tracking task a cluster buffer of 20 clusters is sufficient and only requires a few kilobytes of memory.

Drazen *et al.* (2011) presented an application for real time particle tracking in high Reynolds number flows. The flow under investigation is a pipe flow with $Re \approx 10^5$ and a field of view of 6 cm^2. Polystyrene particles of size $D = 950\ \mu m$ are used for seeding. The particles are fully resolved, with an image diameter of approximately 3 pixels for the DVS system. For reference, the same measurements are carried out with a high speed

[1] Difference of the sum of events per pixel in a given time interval.

camera from Photron. A cluster algorithm is used for tracking, assigning events to nearby particles, otherwise introducing a new particle. A particle mass is calculated, decremented in time and increasing with the number of associated events. A low mass depicts a "dead" particle, which can be dropped from the tracking list. The advantage of the approach is the computational simplicity and therefore fast calculation. Necessary however is a relatively noise free acquisition and high event count per particle, for sparse real events the algorithm might not recognize tracks.

Piatkowska *et al.* (2012) introduced a cluster algorithm for tracking people. It is based on the Gaussian Mixture Model, where the event distribution is approximated by a weighted sum of Gaussian functions, each representing an object. The model is updated every 10 ms, processing all collected events in this time interval. The algorithm is able to handle multiple objects in situations with high occlusion by using a maximum a–posteriori (MAP) algorithm to estimate the association probability between events and objects. This is described as a significant advantage over conventional distance–based clustering methods which usually assume hard cluster boundaries. The method can identify cases of occlusion and is able to correctly assign events to either occluded or occluding objects and keep track of both.

5.2. Filters

Following is an overview of the filters used in this work. The filters are implemented in Matlab and used off–line after the data acquisition.

Filters for noise reduction:

Pair filter: An illuminated bubble traveling in front of a dark background will first generate an ON event which should be followed by an OFF event. This condition has to be met inside a prescribed time window for the event pair to be considered valid.

Neighbour filter: The neighbour filter checks for clusters of events inside a prescribed time window and the neighbouring pixels. Isolated events are considered to be noise and rejected.

Suppression filter: Particle signatures often show "trailing events", these events follow the initial detection of the particle and in certain cases persist for a longer period than the actual residence time of the particle, that is the time the particle is in the view of the pixel. To reduce this effect the pixel activity is suppressed for a prescribed time interval after generating an event, i.e. all following events are neglected. This is a software implementation of the refractory period, for the definition see Section

3.3. Additionally, the number of trailing events can be extracted and used as a measure of strength of the particle signature.

Filters with feature extraction: They check for streak–like features in the data and can also be used to identify outliers.

POD filter: The direction of a streak at a given event location is determined by the direction with the highest variance of the event positions. This is calculated by applying the eigen-decomposition to the covariance matrix and the eigenvector with the largest associated eigenvalue (largest covariance) gives the direction of the streak. The width of the streak corresponds to the second largest eigenvalue. A weighted covariance matrix based on the distance to the given location is used to reduce the effect of outliers.

Polynomial fit with RANSAC: To check for curved streaks a fit with a polynomial of higher order can be used. Since the calculation of the fitting function is sensitive to outliers, a good approach is to use a RANSAC scheme. A random subset of events is chosen to calculate the polynomial fitting function, from which the error for the whole set is calculated and the outliers can be determined. This is repeated a number of times to find the best fit. This approach gives good results, but is computationally expensive.

Filters for tracking: Perform association of events to individual particles and estimation of velocities.

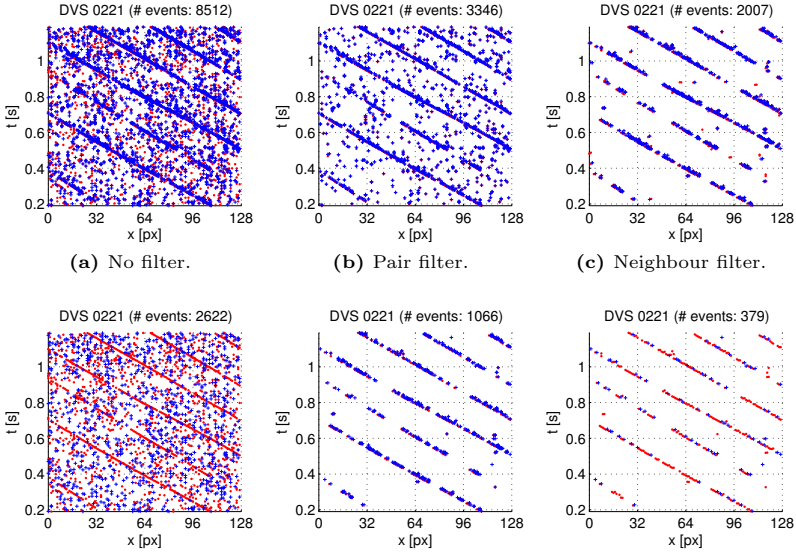
Kalman filter: The Kalman filter offers a simple way to introduce the velocities via a kinematic model into the tracking procedure. It also allows sequential processing of the data, suitable for the data delivered from the DVS. Additionally, it provides the error statistics of the state variables as a basis for associating events to particle tracks. The general formulation behind the theory of the Kalman filter makes it ideal for use as a tracker in $\mathbb{R}^2$, as well as in $\mathbb{R}^3$ for multiple cameras. The main issues are initialization of the state variables and the association of measurements to tracks.

5.2.1. Noise reduction

The first issue that arises after data acquisition is the determination and reduction of the noise encountered. By manual inspection it is often quite clear what is a particle track and what is random noise generated by the cameras. As a base–line a data set from a measurement in the wind tunnel

is investigated. The filters taken into account are the pair–, neighbour– and suppression–filter, as these are computationally low–cost and prove the most useful for an initial noise reduction.

The measurement data was acquired in the empty wind tunnel in a free–flow scenario. The illumination uses LED pairs A,C,D and F at full power, and the wind speed was set to a low value of 1.1 m/s to provide good visibility of the HFSB. The DVS cameras are situated in the top windows of the wind tunnel with the bias set to the nominal settings.



(a) No filter. **(b)** Pair filter. **(c)** Neighbour filter.

(d) Suppression filter. **(e)** Combination of pair and neighbour filter. **(f)** Combination of all filters.

Figure 5.1.: Result of the individual filters (red dots are ON events, blue crosses are OFF events). No colour scale is used; multiple events in one pixel appear the same as a single event.

Applying the filters individually is not the best approach, as can be seen in Figure 5.1. The only filter with a good performance on its own is the neighbour–filter. The best approach is to combine the filters. This is achieved by applying first the neighbour– and pair–filter individually and combining the results. The suppression–filter is applied afterwards on these filtered events.

Parameters for the combined filter are the respective time windows Δt_{nb},

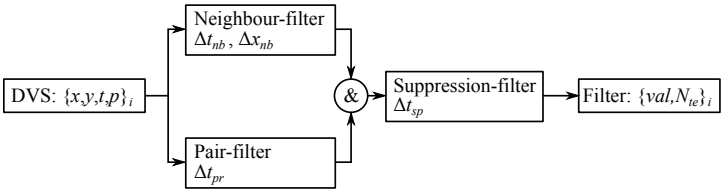


Figure 5.2.: Filter chain.

Δt_{pr}, Δt_{sp} in seconds, and the spatial neighbourhood for the neighbour–filter, Δx_{nb} in pixel. The output for each event is the logical value val, which is true if the event is considered valid and the number of trailing events N_{te}. A good choice for the values of the filters is given in Table 5.1.

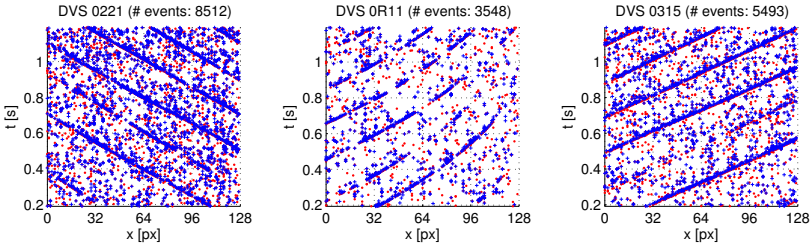
Table 5.1.: Parameters for filter.

	Description	Value
Δx_{nb}	Includes events within $\pm\Delta x_{nb}$ pixels	$1\,px$
Δt_{nb}	Checks for events in interval $[t - \Delta t_{nb}/2,\ t + \Delta t_{nb}/2]$	$10\,ms$
Δt_{pr}	Checks for OFF events in interval $[t,\ t + \Delta t_{pr}]$	$10\,ms$
Δt_{sp}	Checks for events in interval $[t,\ t + \Delta t_{sp}]$	$10\,ms$

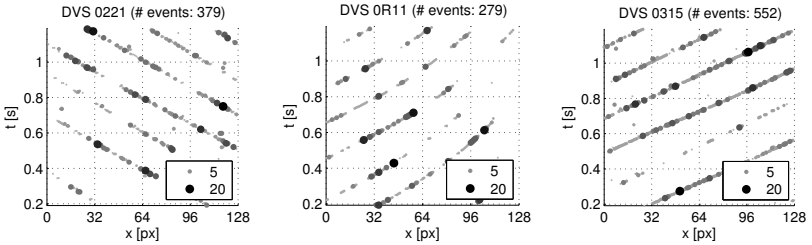
The results for all cameras are shown in Figure 5.3.

The suppression filter plays an important role in the filtering process. For tracking purposes one wants to have a good estimation of the particles' position at a given time. The trailing events counteract this by smearing out the particle's true position over a longer time period, as well as complicating the tracking process with redundant, albeit biased, data. The trailing events also do not show a clear trend, the number and duration depends on the visibility of the particle and the illumination. With the suppression filter all trailing events are remapped onto the first event generated by the particle. This ensures less ambiguity about the particle's position and may still take the strength of the particle signature into account by including the sum of the trailing events as a weight in the tracking algorithm. An example is shown in Figure 5.4 for a flow at a higher velocity.

Based on the results, it is possible to define a signal to noise ratio. Since the filtered events show a clear picture of the track with little noise left, the signal can be defined approximately as the sum of all events from the pair– and neighbour–filter. The noise can be regarded as the sum of all remaining



(a) Original data set.



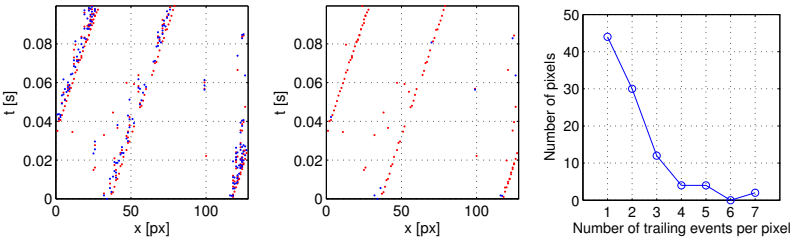
(b) Using all filters and extracting the sum of the trailing events, depicted as size and darkness of the marker.

Figure 5.3.: Result of noise filtering.

events, i.e. all events that are rejected by the pair– and neighbour–filter. The suppression–filter is only used to reduce the track to a precise position in time, but keeps the number of the trailing events associated to the respective track.

$$SNR \approx \frac{\sum_i N_{te,i}}{N_{tot} - \sum_i N_{te,i}} \tag{5.1}$$

where N_{tot} is the total number of events, and $\sum_i N_{te,i}$ is the sum of the output from the suppression filter and corresponds to the number of valid events from the pair– and neighbour–filter.



(a) Trailing events after filtering with pair– and neighbour–filter.

(b) Tracks after subsequent filtering with suppression–filter.

(c) Histogram of trailing events.

Figure 5.4.: Trailing events with flow velocity of $3\,m/s$.

The SNR values for each camera for the presented experimental test case are given below. The values are representative of the range of SNR encountered in most measurements.

$$\begin{aligned} &\text{DVS 0221:} \quad SNR = 8.6\% \\ &\text{DVS 0111:} \quad SNR = 14.7\% \\ &\text{DVS 0315:} \quad SNR = 20.1\% \end{aligned}$$

The values are quite low. The majority of the generated events seem to originate from noise; this, however, is not to be interpreted as a measure of the performance of the DVS. The determination of the SNR is only an approximation. The intention is to filter out isolated events and preserve the particle tracks. It is deliberately defined quite strict and will also filter out events generated by the moving bubbles.

5.2.2. Filters with feature extraction

This section describes two of the initial attempts to derive information from the event driven data. The basic approach is to process a collection of events. This can be either a sample within a certain time interval, or a sequence with a defined number of events. The aim is to determine if an event is part of the particle signature and to calculate the velocity of the particle at the time instant of the event. This is basically a velocimetry technique. The two following procedures were not used in the final tracking algorithm but are included in the thesis as possible complementary strategies.

POD

The Proper Orthogonal Decomposition (POD) has found use in flow visualization as a tool to analyse the flow field by an approximate description,

extracting the large scales and coherent structures. For this application each event is assigned a three dimensional position with time as the third coordinate, $\mathbf{x} = [x, y, t]^T$. A streak is present if a collection of events shows a strong variation in one direction. The main direction is derived from the POD as the direction with the highest variance.

The POD looks for a transformation of the data into a set of uncorrelated variables while maintaining the information content (i.e. the variation) of the given data set. This is achieved with an eigenvalue transform of the covariance matrix $\mathbf{\Sigma}$.

$$\mathbf{\Sigma} = \frac{1}{N} \sum_{i=1}^{N} (\mathbf{x}_i - \mathbf{x}_0)(\mathbf{x}_i - \mathbf{x}_0)^T \tag{5.2}$$

$\mathbf{\Sigma}$ in this case is defined with respect to the position of the event of interest $\mathbf{x}_0$. This differs from the standard definition of the variance, where the mean of the data is subtracted. A major problem is that events with a large deviation from the current event also have a high influence in the result. A solution is to use a weighted covariance matrix, where a higher weight is given to the near events.

$$\mathbf{\Sigma}^* = \frac{1}{N} \sum_{i=1}^{N} (\mathbf{x}_i - \mathbf{x}_0) w_i (\mathbf{x}_i - \mathbf{x}_0)^T \tag{5.3}$$

A straightforward approach to calculate the weights is to use a normal distribution with a predefined covariance matrix $\hat{\mathbf{\Sigma}}$. The weights are calculated according to Equation 5.4.

$$w_i = e^{-\frac{1}{2}(\mathbf{x}_i - \mathbf{x}_0)^T \hat{\mathbf{\Sigma}}^{-1} (\mathbf{x}_i - \mathbf{x}_0)} \tag{5.4}$$

The weights are additionally used to determine outliers. If the nearest neighbours of an event have low weights, the event can discarded. This criterion is similar to the neighbour filter. The principal axis corresponds to the eigenvector $\mathbf{v} = [v_1, v_2, v_3]^T$ associated with the largest eigenvalue. The velocity can then be calculated with

$$\mathbf{u} = \frac{1}{v_3} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}. \tag{5.5}$$

The eigenvalues provide information on the shape of the cloud of events. A streak is characterized by a distinctive main axis; the ratio of the largest and second largest eigenvalue describes the aspect ratio of the feature and needs to be much larger than 1 to be considered a streak.

An example is given in Figure 5.5. The distribution of the weights was set to $\hat{\mathbf{\Sigma}} = \begin{bmatrix} 25 & 0 & 0 \\ 0 & 25 & \\ 0 & 0 & 10000 \end{bmatrix}$.

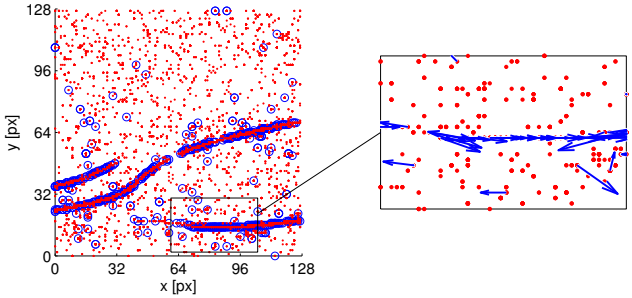


Figure 5.5.: POD applied to a DVS recording of soap bubble tracks.

The velocity estimation is not robust and sensitive to noise and trailing events. The procedure is also computationally expensive. The use for tracking particles is limited.

Polynomial fit with RANSAC

A more robust velocity estimation than the POD filter can be achieved by fitting a polynomial function (position as a function of time: $x = at^2 + bt + c$) to a collection of events. The main problem is the sensitivity of the fitting procedure to outliers. The Random Sample Consensus algorithm was introduced as a solution to this type of problem. It is an iterative algorithm, which selects a subsample of the data for fitting. An error measure for the whole data set is recorded and the procedure is repeated a number of times to ensure robustness. As a result the fit with the lowest error is retrieved, from which the outliers can be determined.

A good estimate for the of number of iterations N needed to get a good model with probability p is given with $N = \frac{\log(1-p)}{\log(1-(1-o)^n)}$. o is the outlier probability and n the minimal number of data points required for the fitting procedure. In the work of Hanimann (2011) about 300 iterations using 8 data points for evaluating the fit proved to provide good results.

The disadvantage of this procedure is that it requires a relatively high number of iterations (on the order of $\mathcal{O}(10^2)$) which makes it computationally expensive. The procedure is not well suited for tracking multiple particles especially when the particle tracks are close or overlapping. The advantage is a robust detection of curved streamlines (see Hanimann

(2011)), as well as the absence of any assumptions on the data properties or the feature size.

An example is given in Figure 5.6. The track estimation is robust and can provide smooth results. The disadvantage is the computational cost and the difficulty to track multiple targets.

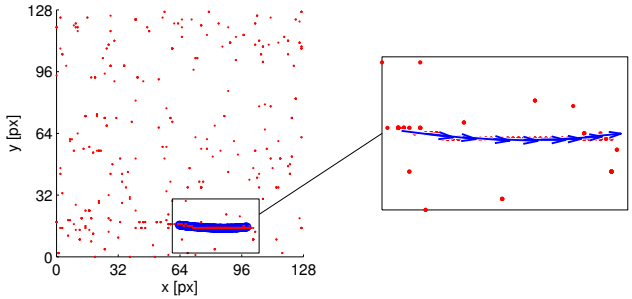


Figure 5.6.: RANSAC applied to a DVS recording of soap bubble tracks.

5.3. Kalman Filter

The Kalman filter was introduced by Kalman (1960). It has been established as an important tool in a variety of fields, in particular for tracking applications. It is well suited for handling dynamic systems, where the state of a system is evolved over time given the dynamic model, a series of noisy measurements and external control inputs. The Kalman filter provides a statistically optimal estimation of the state variables that is better than the estimate from a single measurement. The state variables can include underlying quantities that are not, or cannot, be measured, or cannot be directly derived from the measurements. The Kalman filter is based on the assumption that the dynamic model and the measurements are subject to noise and that all noise terms are normally distributed, in case of multiple variables given by a multivariate Gaussian distribution. Blackman (1986) describes it as the general solution to the recursive, minimized mean–square estimation problem and lists the following advantages for use in tracking applications (p. 25).

1. The Kalman gain sequence automatically adapts to changing detection histories, in particular varying sampling intervals and missing data.

2. The Kalman filter provides a convenient measure of the estimation accuracy through the evolution of the covariance matrix of the state

variables. This is essential for an accurate association of measurements to individual targets.

3. The Kalman filter can be adapted to varying target and measurement environments by changing a few key parameters, thus providing a flexible basis for a tracking algorithm.

Point 1 is very important for the use with the DVS, as it allows a straightforward handling of the temporal stream of data.

The basic Kalman filter assumes a linear dynamic system and measurement equation. A more general approach is given with the extended Kalman filter for non–linear systems, where the covariance estimation is evolved using a Taylor series approximation; this will be described at the end of this section.

Following is an overview of the algorithm with the basic definitions and equations. The notation used here is found in Reid & Term (2001) and on the Wikipedia page about the Kalman filter, which offers a very good oversight of the topic. Considering a linear dynamic system, the state prediction equation is given by Equation 5.6, where subscript $(\cdot)_k$ stands for the current time step.

$$\mathbf{x}_k = \mathbf{F}_k \mathbf{x}_{k-1} + \mathbf{B}_k \mathbf{u}_{k-1} + \mathbf{w}_k. \tag{5.6}$$

$\mathbf{F}_k$ is the state transition model, $\mathbf{B}_k$ is the control–input model and $\mathbf{w}_k$ is the process noise. The control input is not considered in this thesis and not discussed further. The state is related to the measurements via the measurement (or observation) equation

$$\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k, \tag{5.7}$$

with the measurement matrix $\mathbf{H}_k$ and measurement noise $\mathbf{v}_k$.
The process and measurement noise are modeled by zero mean, multivariate normal distributions, which are described by the covariance matrices $\mathbf{Q}_k$ and $\mathbf{R}_k$. A more detailed description of the covariance matrices is found in Sections 5.3.1 and 5.3.2.

$$\begin{aligned} \mathbf{w}_k &\sim \mathcal{N}(0, \mathbf{Q}_k) \\ \mathbf{v}_k &\sim \mathcal{N}(0, \mathbf{R}_k) \end{aligned} \tag{5.8}$$

The matrices $\mathbf{F}_k$, $\mathbf{H}_k$ and the covariances matrices $\mathbf{Q}_k$ and $\mathbf{R}_k$ are presumed known and possibly time varying.

The first step of the Kalman filter is to predict the current state from the previous state with the state transition model. The second step updates the state with the current measurement in an optimal way. Following are the definitions of the variables.

$\mathbf{z}_k$: The measurement at time step k. In this work this corresponds to the pixel position of an event.

$\mathbf{x}_k$: The true state at time step k. The state is a vector with entries corresponding to position and velocity (and might include acceleration as well) of the particle in either $\mathbb{R}^2$ or $\mathbb{R}^3$.

$\hat{\mathbf{x}}_{k|k-1}$: The *a priori* state estimate at time step k, that is *without* knowledge of the measurement $\mathbf{z}_k$.

$\hat{\mathbf{x}}_{k|k}$: The *a posteriori* state estimate at time step k, that is *with* knowledge of the measurement $\mathbf{z}_k$.

$\mathbf{y}_k$: The measurement residual is the difference between measurement $\mathbf{z}_k$ and the *a priori* state estimate $\hat{\mathbf{x}}_{k|k-1}$.

The Kalman filter provides the statistics of the estimation errors in form of covariance matrices and provides a measure of the estimation accuracy. Following is a list of the definitions.

$\mathbf{P}_{k|k-1} = \mathrm{cov}(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1})$: The covariance of the *a priori* error.

$\mathbf{P}_{k|k} = \mathrm{cov}(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})$: The covariance of the *a posteriori* error.

$\mathbf{S}_k = \mathrm{cov}(\mathbf{y}_k)$: The covariance of the measurement residual.

The optimal Kalman gain $\mathbf{K}_k$ is calculated by minimizing the variance of the *a posteriori* state estimation $\mathbf{x}_k - \hat{\mathbf{x}}_{k|k}$, i.e. the trace of $\mathbf{P}_{k|k}$. The covariance matrices of the estimates are evolved together with the state variables. The resulting equations for the optimal Kalman gain are given in Table 5.2.

The measurement residual $\mathbf{y}_k$ (also called the innovation) multiplied with the optimal Kalman gain provides the update of the state. The innovation plays an important role as a measure of the performance of the Kalman filter. It is also used to estimate the likelihood of the measurement (the probability of the state given the measurement).

For an optimal operation of the Kalman filter a robust estimation of the process and measurement noise is required (i.e. the covariance matrices $\mathbf{Q}_k$ and $\mathbf{R}_k$). There has been extensive research on this topic, for further reading see Anderson & Moore (2012), Matisko & Havlena (2012). In this work the optimal values are determined from simulation and experiment.

The basic formulation of the Kalman filter (as discussed so far in this work) assumes a linear relation for the state transition model and the observation equation. The extended Kalman filter provides an approximation

Table 5.2.: Overview of the Kalman filter equations

Prediction	
Predicted state estimate	$\hat{\mathbf{x}}_{k\|k-1} = \mathbf{F}_k\hat{\mathbf{x}}_{k-1\|k-1}$
Predicted estimate covariance	$\mathbf{P}_{k\|k-1} = \mathbf{F}_k\mathbf{P}_{k-1\|k-1}\mathbf{F}_k^T + \mathbf{Q}_k$
Update	
Innovation	$\mathbf{y}_k = \mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k\|k-1}$
Innovation covariance	$\mathbf{S}_k = \mathbf{H}_k\mathbf{P}_{k\|k-1}\mathbf{H}_k^T + \mathbf{R}_k$
Optimal Kalman gain	$\mathbf{K}_k = \mathbf{P}_{k\|k-1}\mathbf{H}_k^T\mathbf{S}_k^{-1}$
Updated state estimate	$\hat{\mathbf{x}}_{k\|k} = \hat{\mathbf{x}}_{k\|k-1} + \mathbf{K}_k\mathbf{y}_k$
Updated estimate covariance	$\mathbf{P}_{k\|k} = (I - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_{k\|k-1}$

for the case of non–linear models. The state transition and observation equations can take the form of any general function.

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}), \quad \mathbf{z}_k = h(\mathbf{x}_k) \tag{5.9}$$

The state transition model matrix and the measurement matrix used in the calculation of the covariance matrices are approximated by the first order derivatives.

$$\mathbf{F}_k = \left.\frac{\partial f}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_{k-1,k-1}}, \quad \mathbf{H}_k = \left.\frac{\partial h}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_{k,k-1}} \tag{5.10}$$

This approach is only valid if the non-linearity is sufficiently small and may otherwise lead to an erroneous estimation of the state and its covariance matrix.

5.3.1. Kinematic models for tracking in $\mathbb{R}^2$

A common approach for a tracking model is to include the velocity in the state variables and to assume it to be constant. The state vector is given by Equation 5.11.

$$\mathbf{x}_k = [x,\ y,\ u,\ v]^T \tag{5.11}$$

The measurement only includes the positions. The velocities are unknown.

$$\mathbf{z}_k = [x_k,\ y_k]^T \tag{5.12}$$

The measurement is related to the state vector by the measurement matrix $\mathbf{H}_k$.

$$\mathbf{H}_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{5.13}$$

The covariance matrix of the measurement error $\mathbf{v}_k$ is described with the variance r and, assuming the error terms are independent, becomes

$$\mathbf{R}_k = E\left[\mathbf{v}_k \mathbf{v}_k^{\mathbf{T}}\right] = \begin{bmatrix} r & 0 \\ 0 & r \end{bmatrix}. \tag{5.14}$$

With the assumption of constant velocity the state prediction matrix is simply given by the kinematic relation $\mathbf{x}_{k+1} = \mathbf{F}_k \mathbf{x}_k$.

$$\mathbf{F}_k = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.15}$$

Under the assumption of constant velocity the acceleration is equal zero. Therefore, by analogy, the process error can be modeled as an acceleration term, as this is the leading–order term of the deviation from the prescribed kinematic model. This discretized ansatz gives a physics based estimation of the magnitudes of the process noise, and allows the inclusion of the variation of velocity into the tracking algorithm. Equation 5.16 describes the kinematic motion for a single time step. The noise term $\mathbf{w}_k$ can be described as the deviation from constant velocity by the acceleration terms.

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ u_{k+1} \\ v_{k+1} \end{bmatrix} = \begin{bmatrix} x_k + u_k \Delta t + a_x \Delta t^2/2 \\ y_k + v_k \Delta t + a_y \Delta t^2/2 \\ u_k + a_x \Delta t \\ v_k + a_y \Delta t \end{bmatrix} \tag{5.16}$$

$$\mathbf{w}_k = \begin{bmatrix} a_x \Delta t^2/2 \\ a_y \Delta t^2/2 \\ a_x \Delta t \\ a_y \Delta t \end{bmatrix} \tag{5.17}$$

Assuming the acceleration terms a_x and a_y are independent and normally distributed with variance q and zero mean, the process noise covariance

matrix can be calculated.

$$\mathbf{Q}_k = E\left[\mathbf{w}_k \mathbf{w}_k^T\right] = \begin{bmatrix} \Delta t^4/4 & 0 & \Delta t^3/2 & 0 \\ 0 & \Delta t^4/4 & 0 & \Delta t^3/2 \\ \Delta t^3/2 & 0 & \Delta t^2 & 0 \\ 0 & \Delta t^3/2 & 0 & \Delta t^2 \end{bmatrix} q \tag{5.18}$$

A similar formulation is given in Bar-Shalom & Fortmann (1988), p. 84 and is in contrast to the discretized motion equation (Equation 5.18) derived by integration of the continuous motion equation in the time interval $[0,\ \Delta t]$. The resulting covariance matrix for the acceleration term is given in Equation 5.19. For more information on the derivation the reader is referred to the literature.

$$\mathbf{Q}_k = \begin{bmatrix} \Delta t^3/3 & 0 & \Delta t^2/2 & 0 \\ 0 & \Delta t^3/3 & 0 & \Delta t^2/2 \\ \Delta t^2/2 & 0 & \Delta t & 0 \\ 0 & \Delta t^2/2 & 0 & \Delta t \end{bmatrix} q \tag{5.19}$$

Another and simpler formulation is used by Reid (1979). The resulting process noise covariance matrix is given as

$$\mathbf{Q}_k = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & \Delta t \end{bmatrix} q. \tag{5.20}$$

The main difference here is that the process noise just effects the velocity. The prediction of the new position only depends on the previous position and velocity.

The model can be refined by prescribing a constant acceleration. The state vector is then given as

$$\mathbf{x}_k = [x,\ y,\ u,\ v,\ a_x,\ a_y]^T. \tag{5.21}$$

The corresponding state prediction matrix is

$$\mathbf{F}_k = \begin{bmatrix} 1 & 0 & \Delta t & 0 & \Delta t^2 & 0 \\ 0 & 1 & 0 & \Delta t & 0 & \Delta t^2 \\ 0 & 0 & 1 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & 1 & 0 & \Delta t \end{bmatrix}. \tag{5.22}$$

The process noise covariance matrix can then be calculated as (Bar-Shalom & Fortmann (1988), p. 85)

$$\mathbf{Q}_k = \begin{bmatrix} \Delta t^5/20 & 0 & \Delta t^4/8 & 0 & \Delta t^3/6 & 0 \\ 0 & \Delta t^5/20 & 0 & \Delta t^4/8 & 0 & \Delta t^3/6 \\ \Delta t^4/8 & 0 & \Delta t^3/3 & 0 & \Delta t^2/2 & 0 \\ 0 & \Delta t^4/8 & 0 & \Delta t^3/3 & 0 & \Delta t^2/2 \\ \Delta t^3/6 & 0 & \Delta t^2/2 & 0 & \Delta t & 0 \\ 0 & \Delta t^3/6 & 0 & \Delta t^2/2 & 0 & \Delta t \end{bmatrix} q. \quad (5.23)$$

In the above formulations for the process noise covariance matrix the factor q is not derived from physical properties and needs to be determined. In the remainder of the thesis q is referred to as a scaling factor for the process noise covariance.

5.3.2. Measurement equation for tracking in $\mathbb{R}^3$

Since we are interested in the 3D reconstruction of the particle path, the state vector can be defined with the 3D position and velocity.

$$\mathbf{x} = [X,\ Y,\ Z,\ U,\ V,\ W]^T \quad (5.24)$$

The measurement and the state vector are related via a projective transformation, for which the homogeneous representation is used.

$$\mathbf{X} = [X,\ Y,\ Z, 1]^T, \quad \mathbf{U} = [U,\ V,\ W,\ 0]^T \quad (5.25)$$

The projective relation between state vector and measurement is given as

$$\mathbf{z} = h(\mathbf{x}) = \begin{bmatrix} X^*/Z^* \\ Y^*/Z^* \end{bmatrix}. \quad (5.26)$$

The camera centric coordinates X^*, Y^*, Z^* of position $\mathbf{X}$ are calculated with the camera matrix $\mathbf{P}_m$ [2] as

$$\mathbf{X}^* = [X^*,\ Y^*,\ Z^*]^T = \mathbf{P}_m \mathbf{X}. \quad (5.27)$$

The projective equation is non–linear, therefore the extended Kalman filter has to be used to derive the linearized measurement matrix.

$$\mathbf{H} = \left.\frac{\partial h(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}_{k|k-1}} = \frac{1}{{Z^*_{k|k-1}}^2} \begin{bmatrix} Z^*_{k|k-1}\mathbf{q}_m^{1T} - X^*_{k|k-1}\mathbf{q}_m^{3T}, & 0,\ 0,\ 0 \\ Z^*_{k|k-1}\mathbf{q}_m^{2T} - Y^*_{k|k-1}\mathbf{q}_m^{3T}, & 0,\ 0,\ 0 \end{bmatrix} \quad (5.28)$$

[2] This corresponds to the projection equation without scaling by the homogeneous coordinate, in this case Z^*.

where $\mathbf{q}_m^{iT}$ are the rows of the product of calibration and rotation matrix $\mathbf{K}_m\mathbf{R}_m$.

The process noise covariance matrix can be chosen in the same manner as for the 2D tracker (Equations 5.18, 5.19, 5.20).

The measurement noise can be included in two ways, either as noise acting on the position of the projection,

$$\mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{v}_k \tag{5.29}$$

or as noise acting on the particle's position,

$$\mathbf{z}_k = h(\mathbf{x}_k + \mathbf{v}_k). \tag{5.30}$$

The first assumption can be handled in the same way as with the 2D tracker. The second assumption can be handled with a first order approximation to derive a modified covariance matrix used in the update step.

$$\hat{\mathbf{R}} = \mathbf{M}\mathbf{R}\mathbf{M}^T, \quad \text{where} \quad \mathbf{M} = \left.\frac{\partial h}{\partial \mathbf{v}}\right|_{\hat{\mathbf{x}}_{k,k-1}}. \tag{5.31}$$

With the projection equation the resulting matrix is given as

$$\hat{\mathbf{R}} = \frac{1}{{Z^*_{k|k-1}}^4} \begin{bmatrix} {X^*_{k|k-1}}^2 + {Z^*_{k|k-1}}^2 & X^*_{k|k-1}Y^*_{k|k-1} \\ X^*_{k|k-1}Y^*_{k|k-1} & {Y^*_{k|k-1}}^2 + {Z^*_{k|k-1}}^2 \end{bmatrix} r \tag{5.32}$$

where r is the measurement variance of the particles' position in sensor space (in pixel units).

5.3.3. Association

A major research topic in tracking is the problem of associating measurements to a specific target. In a cluttered multi–target environment this is not trivial, and several solution strategies with varying degrees of complexity exist. The basis of most of these approaches is the Kalman filter. Following is a list of commonly used strategies in order of increasing complexity. For further reading see Blackman (1986) and Bar-Shalom & Fortmann (1988).

Gating The simplest procedure uses an error measure of the target innovation to associate a measurement to a track. This is a straightforward, non–iterative approach, but is not very well suited in situations where tracks are close or cross.

PDAF The Probabilistic Data Association Filter includes association probabilities of the latest measurements based on the distribution of the state. It assumes there is only one target of interest, and that the state has been initialized.

JPDAF The Joint Probabilistic Data Association Filter is an extension of the PDAF for multiple targets, where the association probability of the latest set of measurements is calculated jointly across the targets. It assumes the number of targets is known and constant.

MHT The Multi Hypothesis Tracker is the most general tracking algorithm. It uses an extensive search of all feasible measurement sequences (the hypothesis) and checks the validity of the track resulting from each hypothesis. The number of hypotheses grows exponentially with each new measurement set. It does not need initialization and makes no assumptions on the number of targets. However, it is also computationally expensive. See Reid (1979) for an example of the implementation.

PMHT The Probabilistic Multi Hypothesis Tracker is an extension of the MHT. Instead of checking each feasible hypothesis, it seeks possible solutions by using an expectation-maximization algorithm. A comparison to the JDPAF is given in Rago *et al.* (1995).

For this work only Gating has been considered for use in the tracking procedure. The experimental setup provides a test environment with a sparse target density, in order to avoid conflicts with crossing particle paths and to have a good basis for validation of the accuracy of the tracking procedure. In future applications it is advisable to look to more sophisticated procedures to enable working with higher tracer densities and noisier data.

Gating

The likelihood of a measurement–to–track association can be evaluated based on the measurement residual $\mathbf{y}_k = \mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}$ and its distribution given by the covariance matrix $\mathbf{S}_k = \mathrm{cov}(\mathbf{y}_k)$. The multivariate distribution function of the measurement residual is given as

$$f(\mathbf{y}_k) = \frac{1}{(2\pi)^{N/2}\sqrt{|\mathbf{S}_k|}} e^{-\frac{d^2}{2}}, \quad d^2 = \mathbf{y}_k^T \mathbf{S}_k^{-1} \mathbf{y}_k \tag{5.33}$$

d is the norm of the residual, N is the dimension and $|\mathbf{S}_k|$ is the determinant.

Blackman (1986) uses the distribution function to introduce a maximum likelihood gate, such that a measurement falling within that gate is more likely to be from that track than from an extraneous source. The condition is then given as

$$d \leq 2\log\left(\frac{P_D}{(1-P_D)\beta(2\pi)^{N/2}\sqrt{|\mathbf{S}_k|}}\right) \tag{5.34}$$

where P_D is the detection probability and β is the expected number of new sources (targets and false alarms) that arise per unit volume per unit scan time. Since neither P_D or β is known in our case, it is more convenient to use a simper approach used by Reid (1979) by defining a validation region η with the condition

$$\mathbf{y}_k^T \mathbf{S}_k^{-1} \mathbf{y}_k \leq \eta^2. \tag{5.35}$$

If this condition holds, the measurement is associated to the track and the state is updated accordingly. This is also called an ellipsoidal gate. It is very easy to implement and to use for a continuous stream of measurements. Conflicts arise when the measurement lies in the validation region of more than one track.

5.4. Implementation

This section will introduce the Kalman filter into the framework of tracking with event driven data. The feasibility of the approach will be demonstrated by using simulated AER data from randomly generated particle tracks and concludes with initial results from experimental data.

5.4.1. Simulated data

To validate the performance of the tracking algorithm a code was implemented to simulate the AER signature of a moving particle as seen by a DVS camera. The projected path of a moving particle is transformed to an address event representation, where a Gaussian function is used to approximate the particles' image. The intensity seen by a pixel is then given as

$$I(t) = I_{max} e^{-\frac{||\mathbf{x}_p(t) - \mathbf{x}_{pixel}||^2}{2\sigma^2}}. \tag{5.36}$$

$\mathbf{x}_p(t)$ is the projected position of the particle and $\mathbf{x}_{pixel}$ is the coordinate of the respective pixel.

The event generation rate is based on the change of the particles' reflectance according to Equation 3.1. An event is generated if $\log(I)$ reaches defined discrete levels $\Delta \log(I)_i$, in accordance with the operating principle shown in Figure 5.7. The parameters are chosen such that the simulated data resembles the actual data, the simulation however is only an approximate model and not a physical model of the pixel design and the optics.

Noise is introduced as *a)* background noise due to the electronic circuit and *b)* missed detection of the particle signature due to low visibility. The background noise is included by adding uniformly distributed events to the event stream. Missed detection is included by deleting a given fraction of the events of the particle signature.

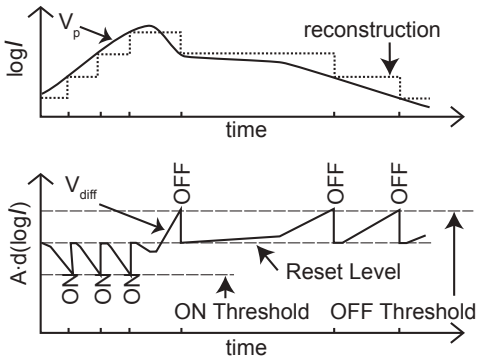
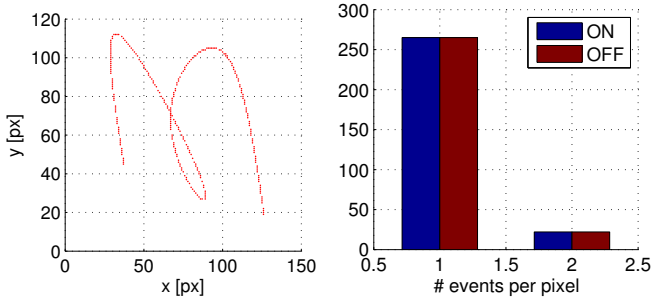


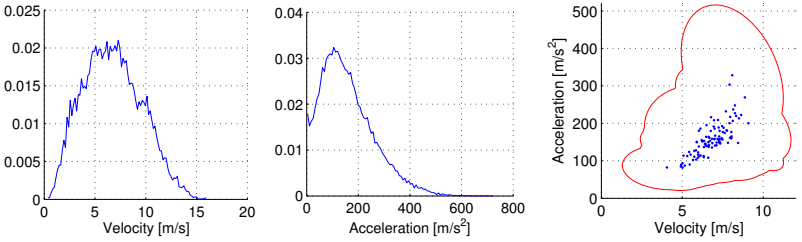
Figure 5.7.: Operating principle of the DVS pixel design (Figure from Lichtsteiner (2006)).



(a) Simulated events for a track. **(b)** Histogram of events per pixel.

Figure 5.8.: Simulated track.

In Figure 5.9 an example of a random track is shown. The width of the track is 1 pixel, with a maximum of 2 events generated per pixel. The tracks have a maximum velocity of 10 m/s and are uniformly distributed inside a volume of $0.8 \times 0.8 \times 0.8\ m^3$. The tracks are supposed to cover a wide range of scenarios of particle movements. The aim is to check the robustness of the tracker for a wide range of applications.



(a) Distribution of velocity magnitude (from all track points).

(b) Distribution of acceleration magnitude (from all track points).

(c) Distribution of mean velocity and acceleration of tracks (red line shows extent of standard deviation).

Figure 5.9.: Statistics for 100 simulated tracks.

5.4.2. Tracking in 2D

The Kalman filter was implemented using the "Gating" method described in Section 5.3.3. It applies a straightforward association with a simple check in the case of multiple feasible tracks. A track is defined as a list of events associated to the path of a single bubble. Included in the track data is the estimation of the position and velocity from the Kalman filter. A list with a given number of tracks is used to keep track of multiple targets. The following rules are applied.

- The event is associated to a track if it is inside the validation region, i.e. $\mathbf{y}_k^T \mathbf{S}_k^{-1} \mathbf{y}_k \leq \eta^2$.
- If an event is inside the validation region of multiple tracks it is assigned to the track with the lowest measurement residual norm $||\mathbf{y}_k||$.
- If an event cannot be associated to any track, a new track is initialized with zero velocity, $\mathbf{x}_{init} = [x_p,\ y_p,\ 0,\ 0]^T$, respectively with zero acceleration $\mathbf{x}_{init} = [x_p,\ y_p,\ 0,\ 0,\ 0,\ 0]^T$. The state may be initialized with a known velocity, if that information is available.
- If the maximum number of tracks are used and a new track has to be initialized, the track with the oldest state update is overwritten.
- The track is deleted from the list if it is not updated for a certain amount of time.
- A track is considered valid if it contains at least a given number of events (minimally about 10 events). Since each event is assigned to

an existing track or otherwise to a new track, this step takes care of tracks consisting of isolated events or an isolated group of events (in other words, noisy events).

An initial value for the covariance matrix $\mathbf{P}_{k|k-1}$ of the a priori state estimation has to be provided as well. Since it does not depend on the measurements, an initial guess can be computed off-line given the parameters q, r and a mean value for the time step Δt, this corresponds to the statistically steady–state (constant covariance of the estimation errors). For the model by Reid (1979) an explicit equation can be derived from the Ricatti equations, see Appendix D.2.

Parameter study

The simulated data was used to obtain optimal values for the parameters q and r. Three models were compared, see Section 5.3.1 for details.

1. Reid's model (constant velocity).
2. Constant velocity model (with $\mathbf{Q}_k$ according to Equation 5.18).
3. Constant acceleration model.

The validation uses two simulated data sets, one set has no background noise and the other is set to $SNR = 0.5$. The missed detection ratio is set to 0.1 for both cases. The errors are calculated as the difference of the filtered track $\hat{\mathbf{x}}$ to the original track $\mathbf{x}$, this allows to give an exact quantification of the performance. Three average error values are considered: *a)* the positional error in pixels, *b)* the relative velocity error, scaled by the maximum velocity of the particle, and *c)* the detection error, that is the fraction of falsely associated events. A parameter study was performed with a wide range of values for q and r, where q is the scaling factor for the process noise covariance matrix (see Equations 5.18, 5.20 and 5.23) and r is the variance of the measurement noise. In total 100 randomly generated particle paths were evaluated and the errors averaged. The results are shown in Figures 5.10 and 5.11.

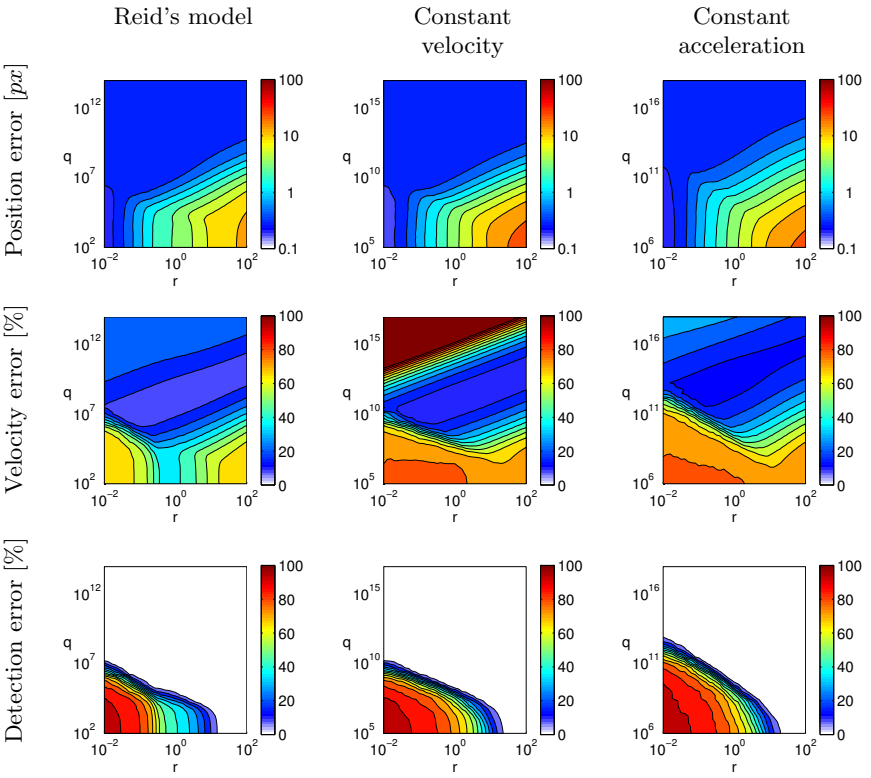


Figure 5.10.: Parameter study for q and r. No background noise. The colour map is defined with white at the minimum, white areas show the optimal performance.

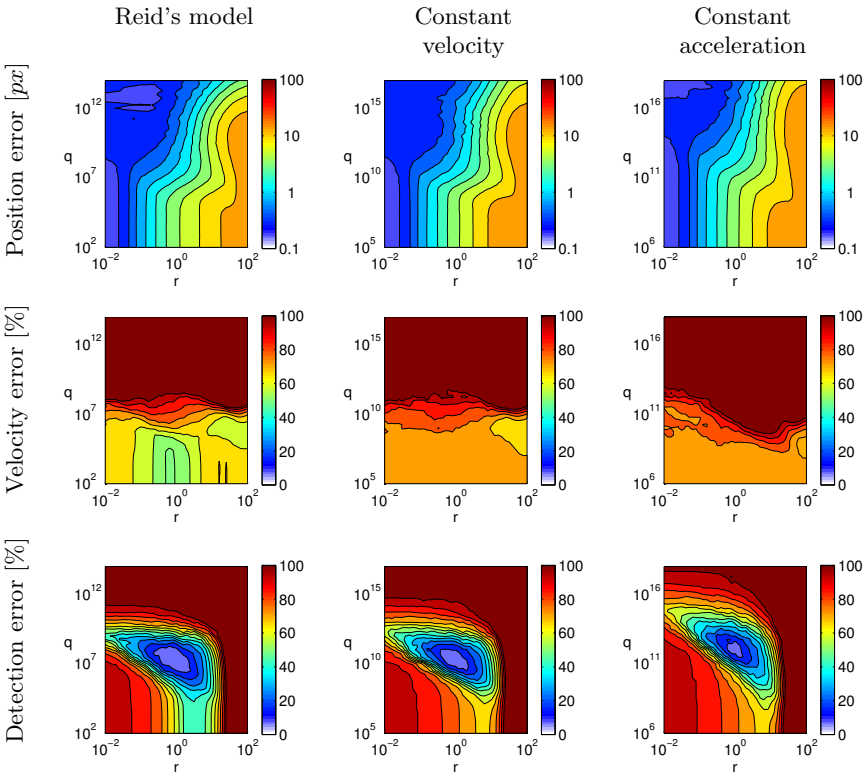


Figure 5.11.: Parameter study for q and r. $SNR = 0.5$.

Optimal values for q and r were calculated as the minimal error for position under the condition that the fraction of wrongly associated events is lower than 5%, wherever possible. The results are given in Table 5.3.

Table 5.3.: Kalman filter: Errors and optimal parameters.

	Reid's model	Constant velocity	Constant acceleration
No noise, no initialization			
Optimal r	0.032	0.016	0.02
Optimal q	$5.62 \cdot 10^6$	$1.78 \cdot 10^{10}$	$5.62 \cdot 10^{12}$
Position error	0.32 px	0.32 px	0.33 px
Velocity error	13.45 %	17.9 %	17.83 %
Detection error	4.94 %	4.97 %	4.95 %
Noise ($SNR = 0.5$), no initialization			
Optimal r	0.79	0.63	1.25
Optimal q	$3.16 \cdot 10^7$	$5.62 \cdot 10^{10}$	$1 \cdot 10^{12}$
Position error	1.24 px	1.05 px	2.14 px
Velocity error	88.96 %	93.56 %	115.67 %
Detection error	7.29 %	8.07 %	11.45 %

It is interesting to see that Reid's model, which is the simplest model, is also the most robust model for a large range of q and r values where the detection error is in an acceptable range. It also produces the lowest errors for the position and velocity estimation and was therefore chosen for further evaluation. The large error for the velocity estimation shows that a post–processing of the tracks is necessary, this can be achieved with a smoothing step, where the tracks are again put through the Kalman filter with adapted parameters for the measurement and process noise.

Experimental data

The tracking algorithm was tested on an experimental data set to check the consistency between simulation and reality. The experimental data is from the flow around the NACA profile, with an angle of attack of 20° to the flow. The flow velocity was 3.1 m/s. The flow was recorded with the DVS cameras placed near the entrance of the wind tunnel, perpendicular to the flow at a distance of 1.5 m to the NACA profile. The recording time was 5 s. With a bubble generation rate of 5 Hz this results in 25 bubble paths.

The bubbles are seeded upstream of the NACA wing tip, to show the tip vortex generated by the wing. In Figure 5.12 the integrated images of the data is shown. The static features visible in the left side of the pictures are tufts attached to the wing. This did not lead to any difficulties in the processing, as the noise filters filtered out a substantial part of the events generated by the tufts. The remaining events do not show any correlation to a track feature and were easily disregarded by the Kalman filter.

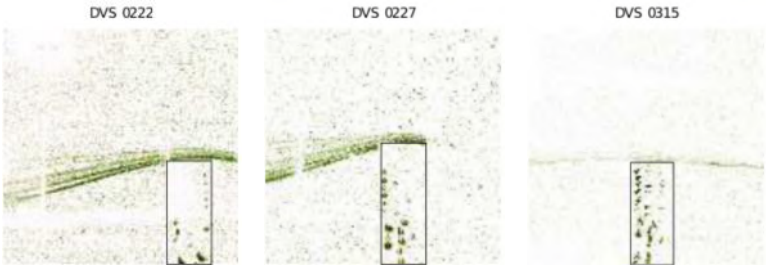


Figure 5.12.: Integrated images of the flow around the tip of a wing (flow is from right to left). The position of the wing is outlined.

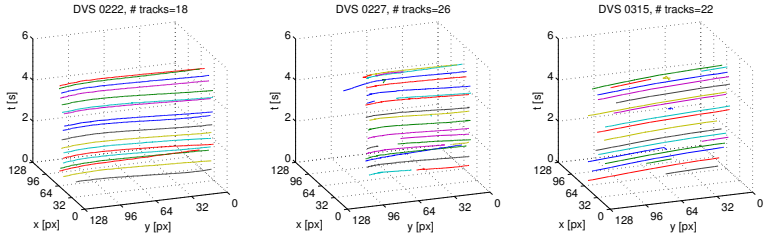


Figure 5.13.: Extracted bubble paths from the Kalman tracker (Reid's model with $r = 1$, $q = 10^7$, $\eta = 1$). The number of tracks recorded per camera is 18, 22, 26, respectively.

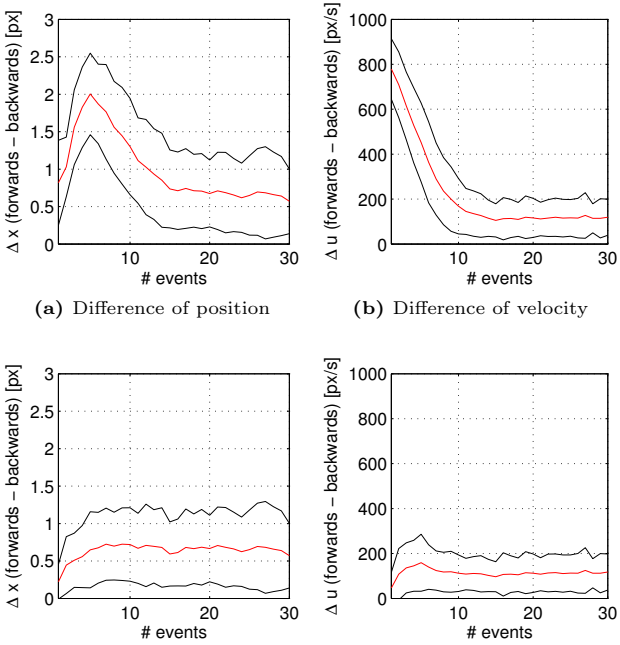
In Figure 5.13 the bubble paths are shown. The tracking algorithm uses the model by Reid, with parameters chosen according to Table 5.3. The expiration time for a track is set to 0.1 s [3], higher values generally lead to false correlations between tracks and events. This is due to the definition of the process noise covariance matrix $\mathbf{Q}$ with Δt, which has the consequence that the validation region is larger for tracks which have not been updated

[3] Regarding a typical velocity in the range of $1 - 10\ m/s$, the corresponding track length is $0.1 - 1\ m$, for a DVS camera at a distance of $1.5\ m$ this corresponds to $14 - 140$ pixels.

for a while. The number of events per identified track is in the range of 81–167.

Initialization

The initialization of the state assumes zero velocity as there is no inherent information on the velocity from the measurement. For a particular experiment it may be possible to give an estimate for the velocities, e.g the free flow in a wind tunnel, this however is not considered in this work due to loss of generality. For a better estimation of the initial state one can use a re–initialization step by applying the Kalman filter backwards in time, once a track has been established. This approach still requires a robust estimation of the track from an uninitialized state (i.e. with zero velocity). An example of the improvement of the estimation is shown in Figure 5.14.



(a) Difference of position

(b) Difference of velocity

(c) Difference of position with reinitialization.

(d) Difference of velocity with reinitialization.

Figure 5.14.: Difference between backwards and forwards propagation of the Kalman tracker (with $r = 1$, $q = 10^6$).

The results in Figure 5.14 are calculated by applying the Kalman filter forwards and backwards in time and calculating the difference between the two estimations. The data has been already filtered by the Kalman tracker and a a few tracks are selected to examine the effect of initialization on the estimation. The results show the average difference of estimation for the first 30 events of the forwards propagation of the Kalman filter for the selected tracks. The largest divergence shows up in the initial phase of the state propagation. The lower set of pictures in Figure 5.14 show the results with reinitialization. Reinitialization is achieved by using a backwards propagation of the Kalman filter when a track has accumulated 10 events. This procedure leads to a better estimation of the initial position and velocity.

5.4.3. Reconstructing the 3D tracks

Two basic strategies for reconstructing the 3D tracks can be distinguished. The first strategy tracks the bubbles in 2D (sensor space) for each camera and merges them into the respective 3D tracks. The second strategy correlates and updates an established 3D track with events from individual cameras.

The first strategy uses the fundamental matrix constraint (Equation 4.14) to check for corresponding 2D tracks between two cameras. A possible match is given if the epipolar error between two tracks is lower than a given threshold; the two tracks are then triangulated to calculate the 3D track. The reconstruction problem is complicated by the use of multiple cameras. For M cameras one has to check for $\mathcal{O}(N^M)$ possible correspondences, where N is a typical number of tracks from each camera. This is feasible if N is low, as is the case here, since the bubble generation is typically around $5\,Hz$ and the residence time of the bubbles is short.

An optimal reconstruction has to take care of missing data, as a track from one camera might have a corresponding track from a single or multiple other cameras (if at all). Considering this, a simplified approach can be chosen, where the tracks are checked between 2 cameras for all combinations of cameras. The number of checks amounts to $\mathcal{O}(MN^2)$. The reconstructed tracks can be evaluated further by checking if tracks from different camera pairs overlap and then can be combined and averaged. Figure 5.15 gives an overview of the procedure.

The recombination can be summarized as follows. For each pair of tracks from a pair of cameras do:

- Check for temporally overlapping sections of tracks. Collect all events that are in a shared time interval, $t_i \in \Delta t_{overlap}$.
- Interpolate the state of each track at the collected timestamps t_i. The

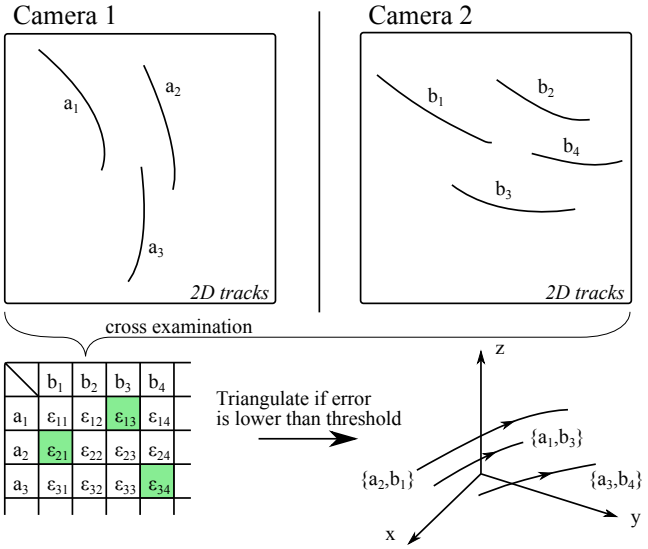


Figure 5.15.: Stereo check of matching tracks for reconstruction.

result is two compatible sets of states of equal number and at equal time steps.

- Check the epipolar constraint for the interpolated values, i.e. calculate the epipolar error given in Equation 4.20.
- If the epipolar constraint holds (given some tolerance), the tracks can be reconstructed.

The downside with this approach is the appearance of "ghost tracks"[4], as the epipolar constraint is not ambiguity–free. Reconstruction of seemingly corresponding projections of particle tracks can lead to erroneous results. The longer the track, the less likely this is to happen. For experiments with low SNR and particle paths with low particle count and gaps, this approach does not perform very well. It also cannot compensate for missing data and for the parts of the track where no overlapping occurs.

The extended Kalman filter from Section 5.3.2 provides a better strategy for reconstruction. As it is based on the framework of the Kalman filter, it offers all the advantages discussed in Section 5.3. The reconstruction

[4]The wording is borrowed from tomographic PIV, where false reconstructions are called "ghost particles".

of the 3D track is handled in a sequential manner, same as for the 2D tracking approach. Checking for feasible combinations of 2D tracks is not necessary, the measurements are directly related to the 3D track via the projection equation, for association a suitable approach can be used (e.g. a gating criteria). The difficulty arises in the initialization step. This is a straightforward process in 2D, where the state and measurement are directly coupled. In the case of a 3D tracker we are looking at a projection of the state to a two–dimensional plane. The measurement can originate from a one dimensional manifold of the state space and there is no straightforward initialization using measurements from a single camera.

The sequential handling of the data with a 3D Kalman filter is shown in Figure 5.16. The input is an ordered list of events from all cameras with information on the source of the event, that is the respective camera and the corresponding camera matrix. The events represent the measurement. The state is represented by the position and the velocity of the particle.

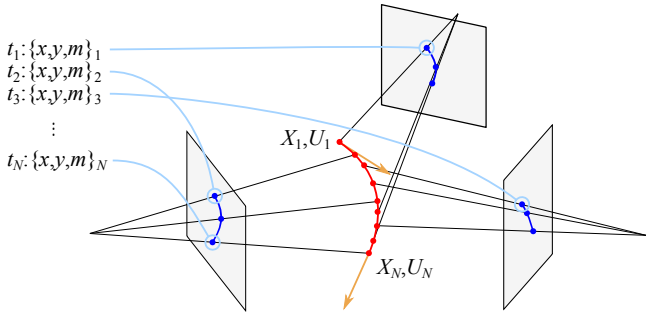


Figure 5.16.: Sequential reconstruction from events of various cameras. The sequence is ordered in time and for each event the respective came matrix is used to calculate the innovation.

The measurements are compared to the projected position of the 3D track, the innovation is essentially the reprojection error of the track[5]. The update follows for each measurement which is inside the validation range of the innovation. If only information from one camera is present, this represents an update of the state components tangential to the image plane. There is no new information on the state components parallel to the line of sight, which will be subject solely to the kinematic equation of the prediction step.

[5] More precisely, the innovation is the *prediction* of the reprojection error. A better measure for the reprojection error may be calculated with the *a posteriori* state estimate $\hat{\mathbf{x}}_{k|k}$.

For an initial evaluation of the 3D tracker a parameter study is conducted in the same manner as for the 2D tracker. Since the original particle path is known, the initialization is straightforward. The measurement noise was defined according to Equation 5.29, where the error acts on the projected position. The parameter r therefore represents the noise variance in units of px^2. The parameter study is intended to check for a few characteristics of the 3D tracker:

- The ability to reconstruct the particle path from sequential data (compared to direct reconstruction from image points of different views at the same time instant).
- The ability to provide a good estimate of the particles' velocity.
- The convergence of the state to a good estimate if no proper initialization of the state is provided. For this the state is initialized with a position which lies in the center of the measurement volume and with zero velocity.

The results show a good performance of the 3D Kalman filter. Under ideal circumstances (no noise and correct initialization) the filter is able to provide a good estimate of the velocity. In the presence of noise this is however not the case. The filter is able to detect the track even with noise and improper initialization In conclusion it is a good approach to establish the track association between events from multiple cameras. Care has to be taken for the initialization, especially for multiple track scenarios. This can be achieved by running a 2D tracker in parallel and checking for correspondences between cameras, the reconstruction can be used then used to initialize a 3D tracker.

Table 5.4.: Kalman filter in $\mathbb{R}^3$: Optimal parameters and errors for Reid's model (mean±standard deviation).

	No noise, Initialized	$SNR = 0.5$, Initialized	$SNR = 0.5$, Non–initialized
Optimal r	10^3	0.063	0.63
Optimal q	10^6	$1.78 \cdot 10^4$	$5.62 \cdot 10^3$
Position error	$1.93 \pm 0.66\, mm$	$3.46 \pm 1.08\, mm$	$13.06 \pm 44.2\, mm$
Velocity error	4.7 ± 0.9 %	117.8 ± 16.4%	110.9 ± 21.5 %
Detection error	0 %	1.88 ± 7.19 %	5.55 ± 11.81 %

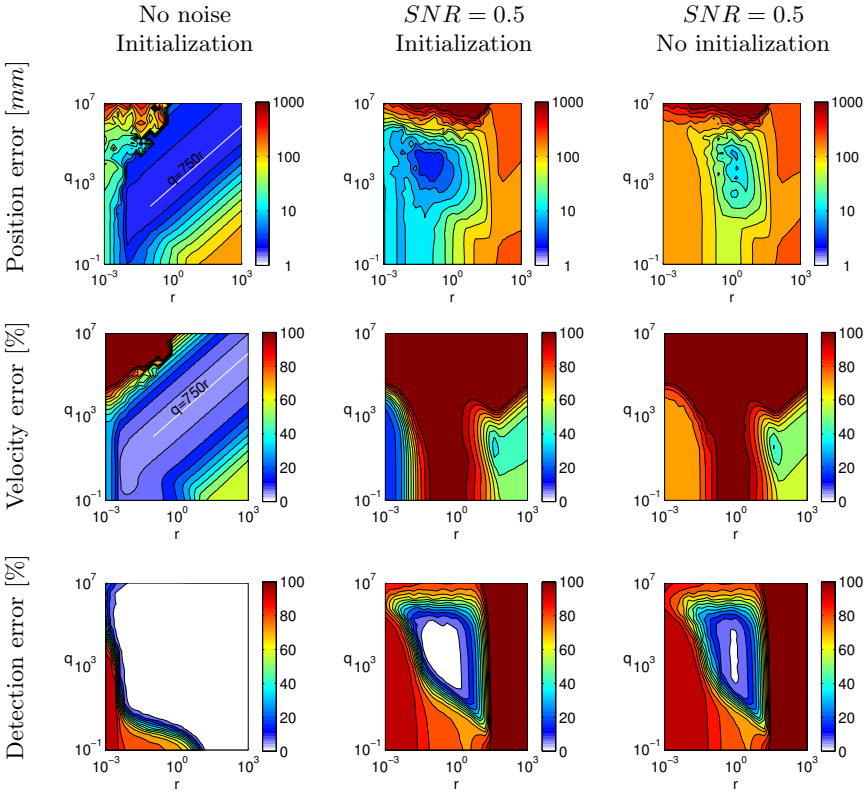


Figure 5.17.: Parameter study for q and r using the 3D tracker and simulated data.

For the idealized case, where the state is perfectly initialized and no noisy events are present, the parameters r and q can be chosen within a wide range for good results. The optimal range for the velocity estimation is given with:

$$\frac{q}{r} \approx 750, \quad \text{with} \quad r > 10^{-1} \tag{5.37}$$

The value of 750 is extracted from the parameter study, it is not known if an analytical formulation for this result exists. It is very likely limited to this test case. However, since it is derived from a wide range of tracks

in the expected range of the experiments, this should present a reasonable baseline to use in the further evaluation.

5.4.4. Truncation

Additionally to the implemented error checks it is very useful to include information about the size of the measurement volume. This is easily implemented by truncating the depth of the line of sight from an image point. The equation

$$\mathbf{X}_c = \mathbf{R}_c\mathbf{X} + \mathbf{t}_c = [X_c,\ Y_c,\ Z_c]^T \tag{5.38}$$

gives the depth with Z_c and a feasible range may be set in advance with knowledge of the measurement test section and camera positions. Generally the expected range of Z_c is within 0.5-2.5 m for the experiments conducted in this work. This criterion is included in the 3D tracker and the initialization from a pair of 2D tracks.

5.4.5. Algorithm

In the implementation of the tracking algorithm a maximum number of N_t tracks in 2D (for each camera) and a maximum number of N_t tracks in 3D is used. The algorithm applies a sequential processing with backtracking for correction of the initialization. A gating criterion is used to associate events to tracks (for 2D and 3D). Following is an overview of the procedure.

For each event do the following:

- **Check 2D association:** Calculate the measurement residual for all tracks and check the gating condition.
  - **If no tracks fulfill condition:** Initialize a new 2D track.
  - **Otherwise:** Select the track with the lowest measurement residual and update the track with the current event via the Kalman filter.
    - **If the updated track has reached N_v events:** Backtrack with the associated events to establish a better initialization.
- **Check 3D association:** Calculate the measurement residual for all tracks and check the gating condition.
  - **If no tracks fulfill condition:** Initialize a new 3D track; do the following:
    - **Check the associated 2D track:** This is a validation step to avoid initialization with 2D tracks that are not established.
      - **If the 2D track has less than N_v events:** Do nothing.
      - **Otherwise:**. The track is considered valid. Cross check between 2D tracks of other cameras (with more than N_v associated events) with the epipolar error.
        - **If no track pairs match:** Do nothing.
        - **Otherwise:** For valid track pairs, reconstruct the position and velocity. Use the reconstruction with the lowest reconstruction error to initialize a new 3D track.
  - **Otherwise:** Select the track with the lowest measurement residual and update the track with the current event via the Kalman filter.

The computational cost is given in Table 5.5. M is the number of cameras involved, and N_e is the average number of filtered events per camera. N_{bt} is the number of Kalman updates used for backtracking. Backtracking is only applied if a track has reached a defined number of events N_v, in the worst case N_{bt} is equal to MN_e. The computational cost scales (maximally) with M^2.

Table 5.5.: Computational requirements for 3D tracking algorithm.

Checks between events and 2D tracks	$N_t M N_e$
Checks between events and 3D tracks	$N_t M N_e$
Check between 2D track pairs (max.)	$(M-1)N_t M N_e$
Kalman update of 2D tracks (max.)	$MN_e + N_{bt}$
Kalman update of 3D tracks (max.)	MN_e
Total computational cost	$\mathcal{O}((M+1)MN_eN_t)$

The tracking algorithm is applied off–line after the data acquisition phase. The algorithm however might be incorporated as a filter into the jAERViewer software, as the code was designed with real–time capability in mind. So far the algorithm has only been implemented into Matlab to test the feasibility of the approach and as a proof–of–principle.

Chapter 6

Results

This section will show the results of the test cases and discuss the issues encountered in the data acquisition and post processing stages. The initial preparation of the data is summarized as follows:

Process the calibration data Calculate the intrinsic and extrinsic camera parameters based on a recording of the calibration target acquired before the measurement.

Filter noise Apply the noise filters (described in Section 5.2.1) to the recorded measurement data for each camera.

Correct for distortion Calculate the normalized coordinates of the events and convert back to pixel units with Equation 4.21. The use of pixel units is more intuitive than the direct use of the normalized coordinates, which have no physical units.

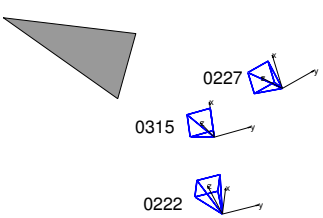
Collect all events in sequential order Collect the events from all cameras into a single list with timestamps in increasing order. Add information on the source of the event, i.e. from which camera it originated. Each event is described by $[x,\ y,\ t,\ p,\ m]$, where m is the respective camera and the other variables are the standard AE information (x and y are now distortion free).

6.1. Delta wing

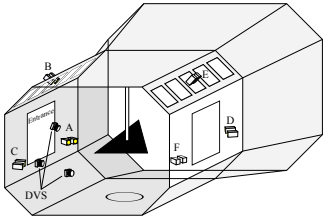
The delta wing is a good test case for the tracking system. The delta wing is tilted at an angle of 25° to the flow. Even at low velocities the delta wing generates a vortex.

A measurement series was conducted at varying velocities in the range of 2-18 m/s. The cameras are placed inside the tunnel to acquire a good

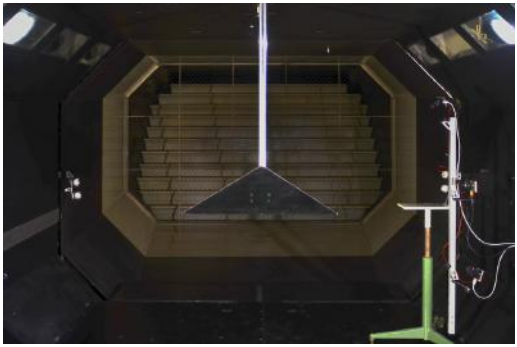
particle signature. The distance of the cameras to the delta wing is about 1.5 m. Figure 6.1 gives an overview of the arrangement. The cameras are viewing the delta wing from a sideways and backwards position (in relation to the flow). The position of the delta winging in regard to the cameras was reconstructed by placing a single LED at the 3 corners. The reconstruction is not very accurate, as the LED was placed in the vicinity of the delta wing corner in such a way that the cameras can see it. The positional reconstruction is only intended for visualization purposes.



(a) Positions of cameras and delta wing from calibration data.



(b) Arrangement in the wind tunnel for cameras and LEDs.

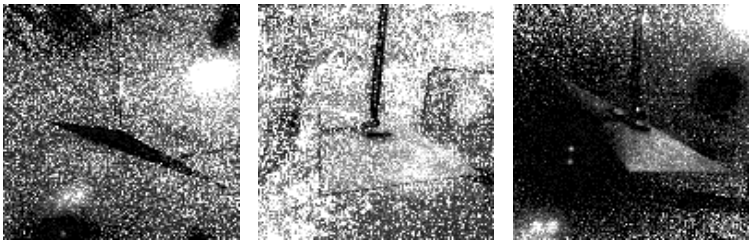


(c) View from the front. The cameras are mounted on a stand placed inside the wind tunnel.

Figure 6.1.: Measurement arrangement for the delta wing.

It is possible to image the wing with the DVS cameras by changing the illumination during recording. The images in Figure 6.2 show the delta wing from the view of each camera. To produce images in this way is not straightforward. A simple summation of the number of events per pixel results in images with few discernible details. It is necessary to manually

adjust the grey scale mapping for each image to achieve a good contrast.

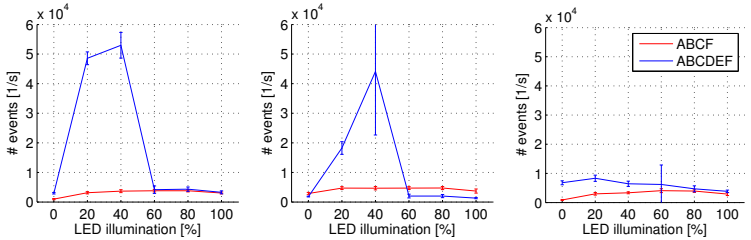


(a) DVS 222 **(b)** DVS 227 **(c)** DVS 315

Figure 6.2.: Image of delta wing. Images are acquired by changing the illumination during recording.

6.1.1. Illumination and contrast

The first issue in a measurement run concerns the illumination. The advantage with the installed LED illumination system is the flexibility to adapt to different scenarios. In a first attempt the illumination was set to its maximum with all LED pairs involved. The problem in this case arises with the LED pairs D and E which are shining directly into the cameras. To compare the influence, an wind–off run was conducted without HFSB. Two scenarios are considered, one with all LEDs involved and an improved situation with LED pairs D and E turned off. For both situations the illumination intensity is varied. In Figure 6.3 the results are shown.



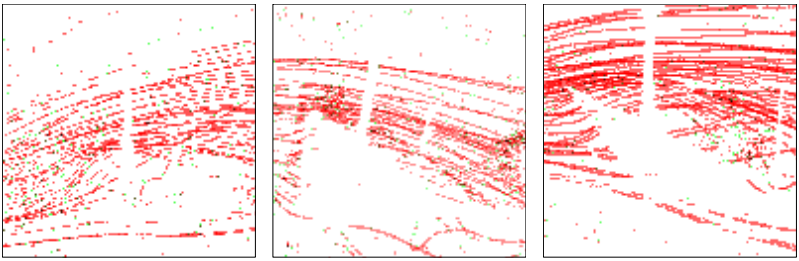
(a) DVS 222 **(b)** DVS 227 **(c)** DVS 315

Figure 6.3.: Events generated in wind tunnel at wind–off in regard to the illumination level (in percent of the full illumination). Two scenarios are considered: 1.) LEDs A,B,C,F are turned on, 2.) all LEDs are turned on.

The increase of noise events at lower illumination levels is apparent for the case with frontal illumination. The reason is the AC-DC conversion of the LED drivers which uses a high frequency pulse width modulation for dimming, that can still be registered by the DVS if the intensity is high enough. At full power the influence is negligible and the noise levels converge.

The more important issue is however the influence of the illumination direction on the particle signature of the HFSB. Viewing against a strong light source reduces or even prevents the visibility of the bubbles. Reflections on the wind tunnel walls and on the model have the same effect and can lead to areas with no information on the particle movement. All surfaces should ideally be dark with low reflective properties.

Figure 6.4 gives an example for a measurement with the delta wing. LEDs A, B, C and F are operated at full level, LEDs D and E are turned off. The images show the integrated particle signatures for a measurement at 6 m/s.



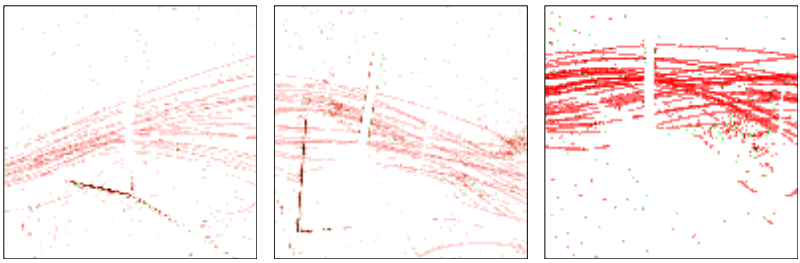
(a) DVS 222 **(b)** DVS 227 **(c)** DVS 315

Figure 6.4.: Integrated images of particle streaks around a delta wing.

The delta wing and holder are clearly visible as empty areas with no particle tracks. Here the use of multiple cameras offers a good method to compensate for missing information.

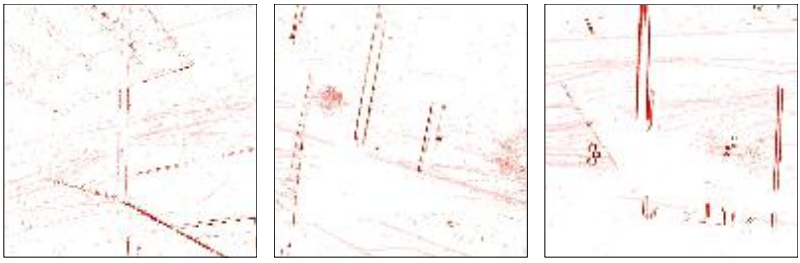
A further example shows the problems with moving objects other than the bubbles. At a wind speed of 10 m/s the delta wing started to shake. This is shown in Figure 6.5.

Overall the measurements did provide good data. Only at the highest velocity of 18 m/s were the cameras not able to capture the particle paths sufficiently. Problematic for the acquisition was the vibration of the cameras and the model due to the flow, this lead to a higher count of events from the apparent movement of the scenery.



(a) DVS 222 **(b)** DVS 227 **(c)** DVS 315

Figure 6.5.: Integrated images of particle streaks showing vibration of delta wing.



(a) DVS 222 **(b)** DVS 227 **(c)** DVS 315

Figure 6.6.: Integrated images of particle streaks at the highest measured velocity of 18 m/s.

For reference the signal–to–noise ratio (SNR) value of the noise filter is shown in Figure 6.7. At higher velocities the SNR value is not representative due to events generated by movement of the scenery, which is especially seen with camera 315.

6.1.2. Tracking results

The tracking algorithm described in Section 5.4.5 is applied to the data obtained in the measurement run introduced in the previous section. To achieve good results it was necessary to adapt the settings, in particular the values of q for the covariance of the process noise. The relation given in Equation 5.37 did not produce good results. This means that the simulated data could not fully represent the measured data. However, the range of the

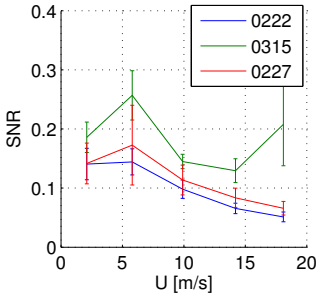


Figure 6.7.: Approximate signal to noise ratio of measurements of the delta wing (SNR according to Equation 5.1).

values used for the final validation are not far off from the suggested values of the parameter study with simulated data. The values were determined manually by inspection of the results. It is generally quite obvious if the values are acceptable, this can be seen by the ability to properly track the bubbles over the whole measurement area. The values used in the final post processing step are shown in Table 6.1.

Table 6.1.: Values for the process noise covariance scaling factor q and measurement noise variance r of the Kalman filter for the delta wing measurement.

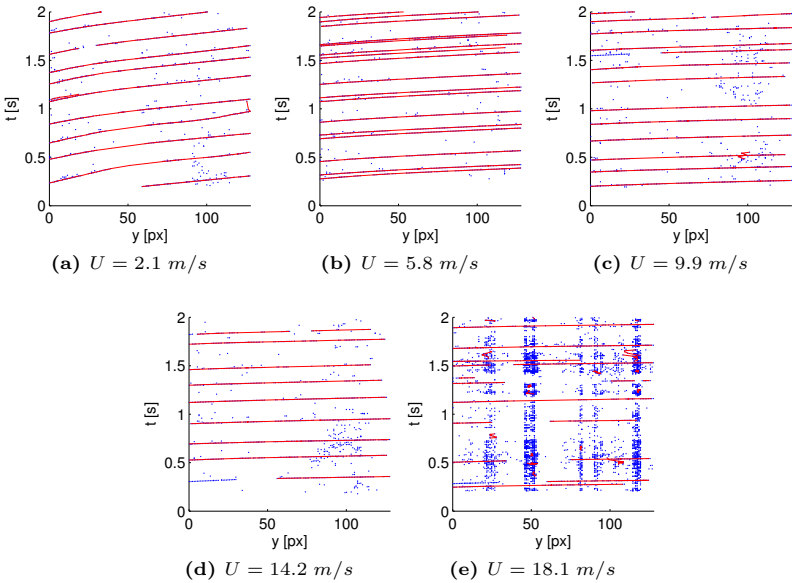
Velocity $[m/s]$	2.1	5.8	9.9	14.2	18.1
r for 2D and 3D	1	1	1	1	1
q for 2D	10^7	10^8	10^9	10^9	10^9
q for 3D	10^2	10^2	10^2	10^2	10^2

For all measurements the validation region is set at $\eta = 8$. After the initial processing of the data, a cleanup routine reapplies the Kalman filter to the extracted tracks with a lower value for η with back– and forward tracking. This is a more rigorous treatment to detect outliers in the tracks or false tracks. The first step is to enable an initial association, the second step deletes "bad" tracks. In this application η is reset to 2 in case of the 2D Kalman filter, and is kept at 8 for the 3D Kalman filter.

The value for r is based on the parameter study and is a reasonable choice in all applications. The values for q show a dependence on the velocity in the case of the 2D Kalman filter, for the lowest velocity the value for q

does correspond to the optimal value obtained in the parameter study, albeit the parameter study did consider tracks with velocities in the range up to 10 m/s. For the 3D Kalman filter it is sufficient to use a constant value for q. According to Equation 5.37 the optimal value should be 750, for comparison. Figure 5.17 does show $q = 100$ still to be in the region with good association capabilities for the simulated data affected by noisy events.

The results for the 2D tracking Kalman are shown in Figure 6.8. The filter produces good results, even for the highest velocity, where a large number of events are due to movement in the scenery. The manual trial–and–error procedure to evaluate the values showed that a good estimation for the 2D tracks is necessary to achieve good results with the 3D Kalman filter. The 2D Kalman filter is able to handle gaps in the bubble signature, if the parameters are chosen properly.



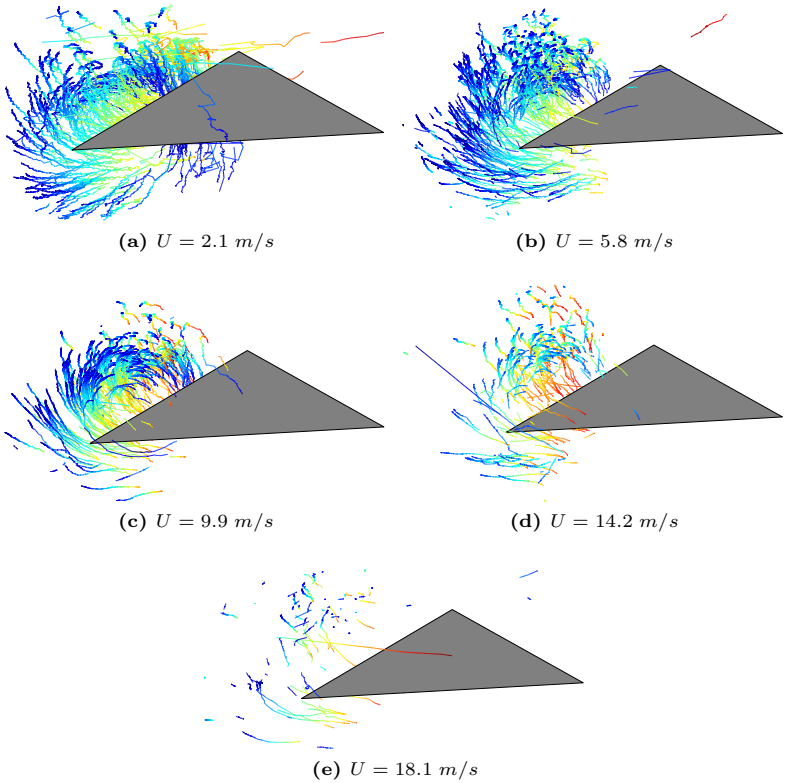
(a) $U = 2.1\ m/s$ **(b)** $U = 5.8\ m/s$ **(c)** $U = 9.9\ m/s$

(d) $U = 14.2\ m/s$ **(e)** $U = 18.1\ m/s$

Figure 6.8.: 2D tracks for the DVS 315. Blue dots show all events processed by the tracking algorithm. Red lines show the extracted tracks.

The results for the 3D Kalman filter are shown in Figure 6.9. The view is from behind the wing to show the vortex. Overall the tracks are well

recovered. There are a few obviously false tracks, showing up as directed normal to the majority of tracks. The tracking algorithm is able to handle velocities up to 14 m/s, for the measurement at 18 m/s the results are considerably worse. A few tracks are visible, but often do not extend over longer distances.



(a) $U = 2.1\ m/s$ **(b)** $U = 5.8\ m/s$

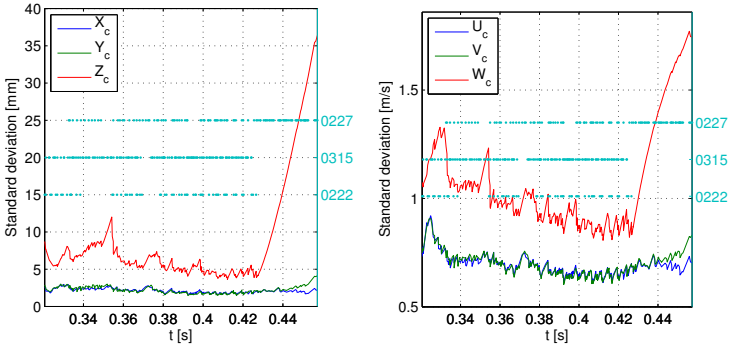
(c) $U = 9.9\ m/s$ **(d)** $U = 14.2\ m/s$

(e) $U = 18.1\ m/s$

Figure 6.9.: 3D tracks around the delta wing. Tracks are colour–coded with the stream–wise coordinate (red indicates the bubbles' position is in front of the wing, blue behind the wing).

To get a clearer picture of the performance of the 3D Kalman filter, a single track is investigated in detail. The track is from the measurement with a velocity of $5.8m/s$. The temporal evolution of the standard deviation of the a posteriori estimation is shown in Figure 6.10, additionally the

source of each event is plotted. The standard deviation is given as the square root of the diagonal elements of $\mathbf{R}_m \mathbf{P}^*_{k|k} \mathbf{R}_m^T$, where $\mathbf{P}^*_{k|k}$ is the covariance of either the position or the velocity estimation. The rotation matrix is included to calculate the covariance in the respective camera coordinate system for each event. The aim is to show how the Kalman filter handles the depth, of which no information is available.



(a) Standard deviation of position in respective camera coordinates.

(b) Standard deviation of velocity in respective camera coordinates.

Figure 6.10.: Standard deviation of updated estimate ($\mathbf{P}_{k|k}$) for one track (free stream velocity of 5.8 m/s). The dots show the camera source of the events.

The estimation suffers in the case of missing information from all but one camera, as evident in the last section of the track. Here, only events from a single camera are available to update the track. The result is an increase of the depth variance. The tracks generally exhibit an arbitrary behaviour in the direction of the line of sight. Gaps in the track lead to the same effect, but are compensated as soon as new information from other cameras is available. The components tangential to the image plane are unaffected. It is apparent that the depth component has always the highest variance, in other words the highest uncertainty. This is consistent with the derivation of the reconstruction accuracy given in Section 4.2.6. For the case shown the standard deviation of the depth increases by 10 mm over 10 ms, as an upper bound for a sufficient reconstruction this is about the time scale after which events from a different source need to be available. A simple measure to reduce this effect is to truncate the track at the beginning and ending where only information from one camera is present. This is essentially a reduction of the measurement volume to the volume where camera views

are overlapping and reconstruction is possible (beyond is just an inaccurate depth extrapolation).

The velocity estimation lies within the expected range. The velocity distribution for all measurements is given in Figure 6.11. The mean velocity matches the free stream velocity, an exact error measure can however not be derived, as we are looking at a more complex flow.

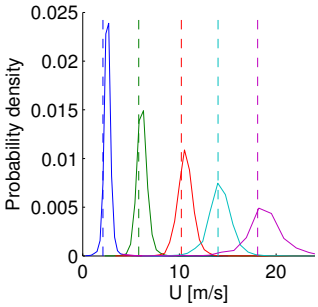


Figure 6.11.: Distribution of the velocity magnitude for all measurements. Dashed lines show the free stream velocity.

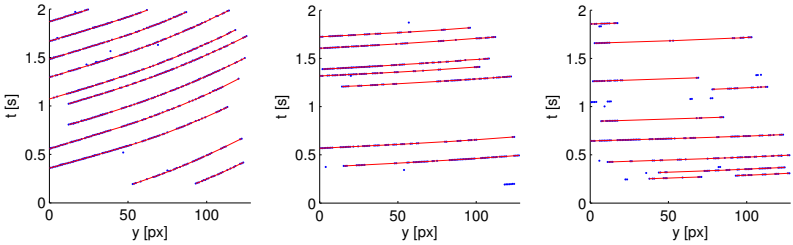
6.2. NACA wing

The NACA wing is a simpler test case than the delta wing to validate the tracking algorithm. The wing is set at an angle of 20° to the flow. The region of interest is behind the wing tip, where a vortex is induced. The measurements are conducted at speeds up to 12 m/s, the data acquisition duration per measurement is 30 seconds. In this case it was again necessary to manually determine suitable values for q. In Table 6.2 the values for the Kalman filter and the velocities of the measurements are shown.

Table 6.2.: Values for the process noise covariance scaling factor q and measurement noise variance r of the Kalman filter for the delta wing measurement for the NACA wing measurement.

Velocity $[m/s]$	1.2	2.5	4.1	5.2	7.3	9.2	10.9	12.8
r for 2D and 3D	1	1	1	1	1	1	1	1
q for 2D	10^4	10^5	10^6	10^7	10^8	10^8	10^9	10^9
q for 3D	0.5	1	1	1	10	50	100	100

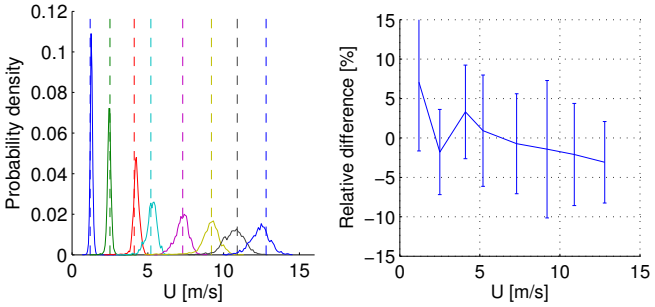
For all calculations the validation region was set to $\eta = 10$, for the secondary validation step η is set to 2 for the 2D Kalman filter. The values for q for the 3D Kalman filter vary considerably compared to the test with the delta wing. It is not quite clear why this is the case. The data has already been used in the evaluation of the velocity influence on the event generation in Section 3.3, see Figure 3.5. Figure 6.12 offers an overview of the 2D tracks. At higher velocities the data shows a low particle signature, however the Kalman filter is still able to track the bubbles despite the gaps apparent in the data.



(a) $U = 1.2\ m/s$ **(b)** $U = 7.3\ m/s$ **(c)** $U = 12.8\ m/s$

Figure 6.12.: 2D tracks for DVS 227.

The distribution of the velocity estimation is shown in Figure 6.13. The results correspond well to the free stream velocity. As the particles largely follow the free stream with small deviations due to the tip vortex this case is chosen to estimate the accuracy of the velocity reconstruction. The relative difference between the velocity estimation $\hat{U}$ and free stream velocity U_{fs} is given in Figure 6.13b as $\frac{\hat{U}-U_{fs}}{U_{fs}}$. The differences are in the order of ca. 5-10%. This deviation corresponds roughly to the result achieved in Section 5.4.3 for the simulated data, which gave the reconstruction error of the velocity at 5% for a track with no noisy events involved.

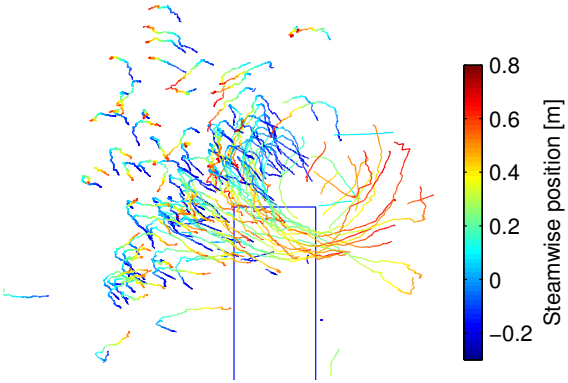


(a) Distribution of the velocity magnitude for all measurements. Dashed lines show the free stream velocity.

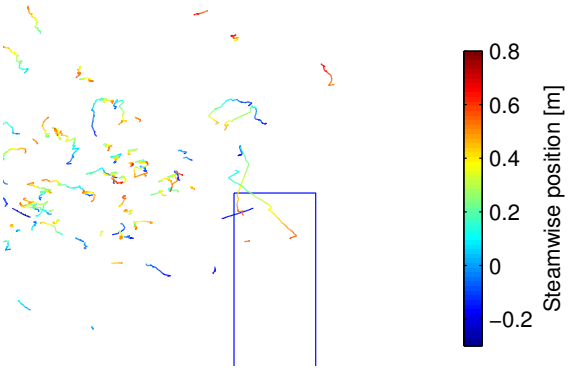
(b) Relative difference to free stream flow (± standard deviation).

Figure 6.13.: Velocity statistics of the tip vortex of a NACA wing.

The 3D tracks in Figure 6.14 show few outliers or false reconstructions. At the highest velocity the quality of the data declines and with it the number of reconstructed tracks. The flow is no longer captured very well.



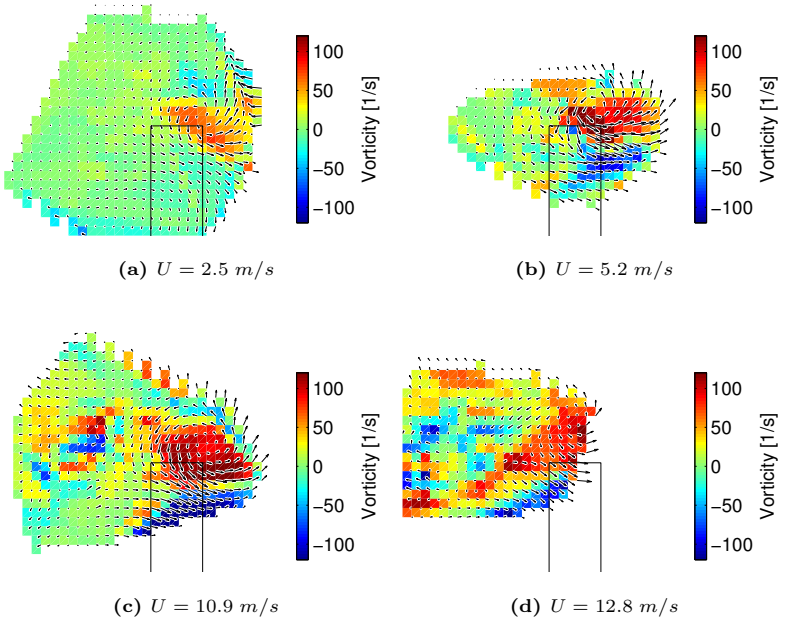
(a) $U = 2.5\ m/s$



(b) $U = 12.8\ m/s$

Figure 6.14.: 3D tracks around the NACA wing. Tracks are colour–coded with the streamwise position (red is downstream of blue).

To get a clearer picture of the flow, the tracks may be interpolated onto a grid[1]. The interpolated results are shown in Figure 6.15. To enhance the quality of the results the acquisition time may be prolonged to acquire more particle tracks. In this measurement the maximum number of bubbles amounts 150, as the bubble generation rate was set to 5 Hz at an acquisition time of 30 s. With this number of bubbles an initial overview of the flow can already be provided.



(a) $U = 2.5\ m/s$ **(b)** $U = 5.2\ m/s$

(c) $U = 10.9\ m/s$ **(d)** $U = 12.8\ m/s$

Figure 6.15.: Tip vortex $0.15m$ behind the NACA wing. The view is from behind the wing tip downstream of the flow. The colour surface shows the streamwise vorticity.

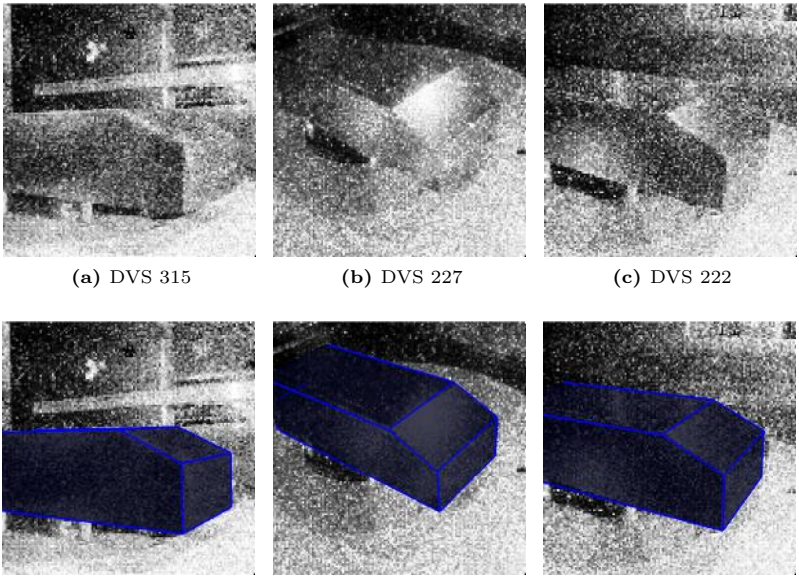
6.3. Ahmed body

The Ahmed body is a standardized model commonly used for investigating separated flows of vehicles. In this work it is used to show the validity of the approach for a real–world application. The main difference to the

[1] The interpolation is computed with the Matlab routine TriScatteredInterp.

two previously investigated flows (NACA wing and delta wing) is a more complex flow with separation. The proximity of the region of interest to the model's surface and the wind tunnel walls proved to be more demanding in terms of background contrast. More care had to be taken to adjust the illumination and optical settings to reduce the negative impact of reflections on the particle visibility. For the measurement the Ahmed body was set at an angle of 15° to the free stream flow to induce a pronounced separation flow.

Figure 6.16 shows the reconstructed images of the Ahmed body. The illumination intensity is varied during recording to achieve a dynamic scene change that can be detected by the DVS cameras. The position of the Ahmed body is determined through the resulting images. The geometry of the Ahmed body is overlaid on the reconstructed images in Figures 6.16d, 6.16e and 6.16f.

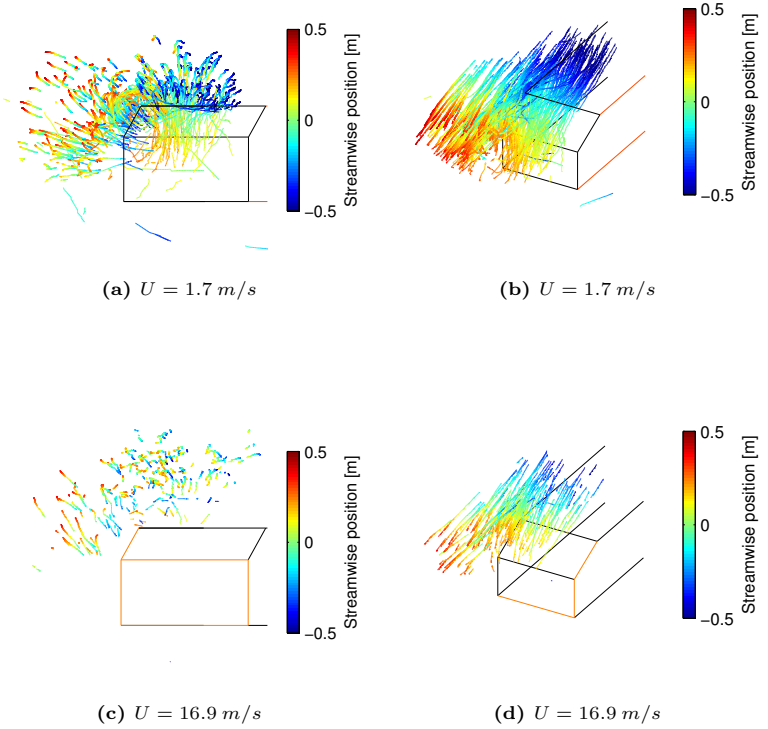


(a) DVS 315 **(b)** DVS 227 **(c)** DVS 222

(d) DVS 315 **(e)** DVS 227 **(f)** DVS 222

Figure 6.16.: Image reconstruction with event driven data in a dynamically illuminated scene by turning the LED lighting on and off during recording. The bottom images show the computed position of the Ahmed body in the calibrated scene.

Figure 6.17 shows the particle tracks for two measurements at low and high velocity. The tracking algorithm shows a lower detection rate of tracks at higher velocities. Some falsely reconstructed tracks are present. Overall the algorithm is capable of tracking the HFSB at velocities up to approximately 17 m/s.



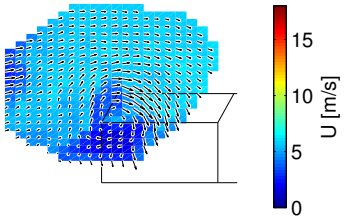
(a) $U = 1.7\,m/s$ **(b)** $U = 1.7\,m/s$

(c) $U = 16.9\,m/s$ **(d)** $U = 16.9\,m/s$

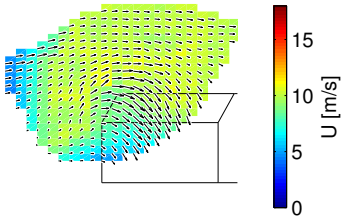
Figure 6.17.: Particle tracks around the rear of an Ahmed body at an angle of 15° to the free flow. The position in direction of the free stream flow is colour–coded as visual aid.

To visualise the flow around the edge of the Ahmed body the velocities are interpolated onto a grid as shown in Figure 6.18. The vortical flow

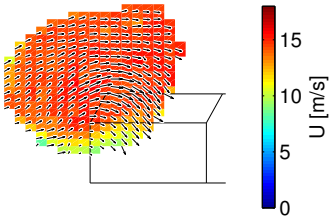
around the edge is clearly visible.



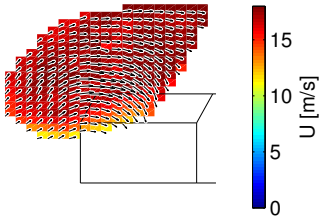
(a) $U = 5.6\ m/s$



(b) $U = 9.9\ m/s$



(c) $U = 14.7\ m/s$



(d) $U = 16.9\ m/s$

Figure 6.18.: Particle tracks around the rear of an Ahmed body at an angle of 15° to the free flow. View is from the back of the Ahmed body. The velocity is colour–coded.

6.4. Discussion

The data was processed by a Matlab routine on a 2.5 GHz laptop using one CPU core. On average the processing time per event is 0.8 ms for the association step and 1.1 ms for the validation step. The total event generation rate for all cameras in the case of the NACA wing is in the range of 630-1350 events per second. The association step can be calculated in real time, the validation step requires about 60 % more time. The implementation was not optimized for speed, but it appears feasible to achieve real time processing.

At this stage, a lot of manual work has to go into the post processing step, especially in determining the values for q. Overall, the Kalman filter does present a good basis for processing the data from the DVS. Since the Kalman filter offers an extensive field of research with more advanced approaches available, the tracking algorithm might be improved to reduce the amount of user input required at the moment.

The Ahmed body represented the most demanding test case, the data shows a lot of noise and it was difficult to achieve a good particle signature due to reflections on the walls and model.

Chapter 7

Conclusions

The Dynamic Vision Sensor offers a new and viable tool for flow visualization. The inherent data reduction enables fast data acquisition and processing. The time accurate registration of particle tracks may otherwise only be achieved with high speed cameras. The current system is capable of measuring flow speeds up to about $18\,m/s$ depending on the background contrast of the particles. This limitation could be overcome with the new generation of sensors and will be subject to future studies.

The illumination is crucial for the optimal performance of the system. A steady light source is required; temporal changes due to AC electric power or PWM operated drivers at low frequencies on the order of $100\,Hz$ will be detected by the DVS cameras. Care has to be taken when positioning the illumination relative to the cameras. It proved to be advantageous to illuminate the test section evenly from all directions. Direct illumination towards the cameras should be avoided. Reflections on the wind tunnel walls and on the model as seen from the cameras need to be avoided where possible.

A novel type of bubble generator was built with the aim to provide a constant generation rate in the range of 5-10 bubbles per second for evaluation of the tracking system. The low generation rate results in a low bubble density and simplifies the tracking of multiple particles. The system has not been tested with high bubble densities. For steady flow fields the low generation rate is compensated by longer acquisition times. For future work it might be interesting to use high particle densities to measure transient flow phenomena. The tracking algorithm however would need to be further optimized for this application.

Calibration of the cameras follows the established approach of calculating the intrinsic and extrinsic parameters with multiple views of a known target

by an iterative non–linear optimization scheme. The target is adapted for the DVS cameras by using pulsed LEDs to increase temporal contrast. The calibration code handles multiple cameras in a single optimization step. The calibration is automated and requires in total about 5 minutes. The reconstruction accuracy of the calibration for the current system is on the order of 2 mm. The calibration accuracy is however higher than that of the tracking procedure as the LED positions are determined with sub–pixel accuracy. The tracking accuracy was not determined experimentally but is expected to lie within 5 mm based on a measurement uncertainty of one pixel.

Tracking and reconstruction of the particle's path proved to be a novel problem in comparison to standard techniques with frame based cameras. The Kalman filter was identified as a promising framework to process the event driven data of the DVS. The sequential nature of both camera and filter complement each other well. Care has to be taken with the parameters for the Kalman filter in order to achieve accurate results. This work established the feasibility of the approach with a simple implementation of the Kalman filter. While more advanced methods have been introduced in the literature, applicability to the current DVS–based configuration will have to be subject of further studies. This is recommended so as to reduce the amount of user input required and to enable a robust tracking algorithm which can adapt to a wide range of flow characteristics. Furthermore the implementation of a real–time tracking algorithm would be useful for feed back to fully scan and measure the region of interest, respectively to identify regions with relevant flow features during measurement.

Appendix A

Bubble generator

The bubble generator used in the work of Macháček (2002) offered a relatively high bubble generation rate on the order of 100 bubbles per second. This generation rate proved to be difficult for a quantitative assessment of the performance of the DVS tracking system. The previous system was a valve controlled device to supply helium, air and soap solution to the nozzle. A new system was developed by Borter (2013) with active control of the helium supply and the soap solution delivery.

The helium flow rate is controlled by a flow controller from Bronkhorst. The accuracy of the flow controller is in the range of $0.35\,ml/min$. The flow rate in the experiments was on the order of $9.1\,ml/min$ (26 % of the maximal flow rate).

The soap solution is delivered by a micro valve in pulses. The micro valve is a SMLD 300G from Fritz Gyger AG and has a fast response time in the microsecond range. The valve is operated by a waveform generator connected to a piezo driver. The piezo driver is necessary to deliver the required power in the duration of the pulse. The pulse width was set to $160\,\mu s$.

The co–flow has to be set by hand with a micrometer valve to ensure a proper bubble separation at the nozzle exit and had to be readjusted at higher flow velocities in the wind tunnel.

The baseline in the experiments was a bubble generation rate of $5\,Hz$ and a bubble diameter of about $3.9\,mm$. The exact size was not measured and is calculated from the volume flow rate of the helium. The generation rate can be increased to about $10\,Hz$ after which the bubble generator tended to produce small bubbles at higher rates than prescribed. This is probably due to improper formation of the slug–flow in the T-tube or of unstable separation of the bubble at the exit of the nozzle.

Figure A.1 shows a picture of the bubble generator with all components except the nozzle. The soap solution is contained in glass bottle. The bottle is pressurized with up to 1.5 *bar* and is placed inside an acrylic cube for security reasons. The nozzle is shown in Figure A.2. In operation the nozzle is mounted on a pole with a cover to shield it from the airflow in the wind tunnel (not shown in the image).

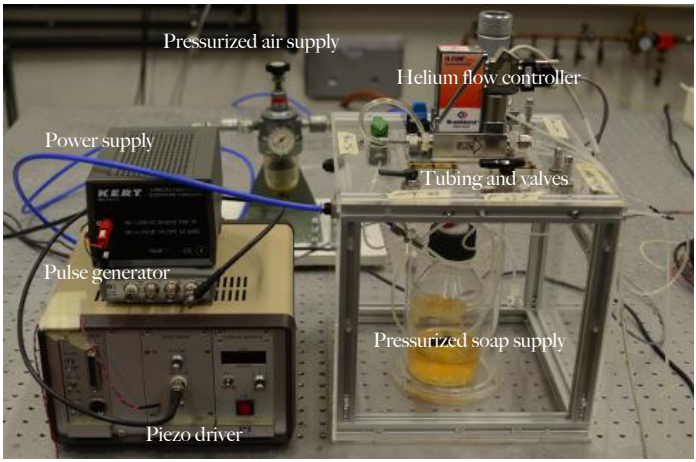


Figure A.1.: Bubble generator.

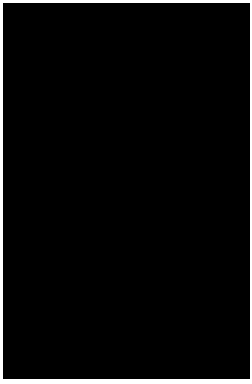


Figure A.2.: Nozzle head with micro-valve.

Figure A.3 shows a diagram of the tubing for the bubble generator. The micrometer valve in the bottom right regulates the co–flow. This is the only manual adjustment needed from the user. The helium flow controller and the function generator for the micro-valve are controlled from a PC.

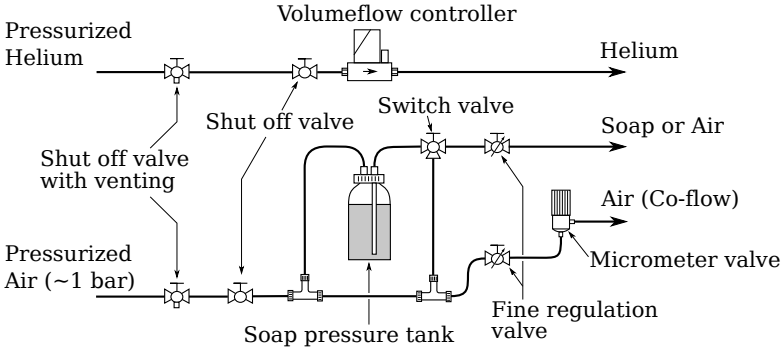


Figure A.3.: Tubing of bubble generator supply system.

Appendix B

Projection and reconstruction

B.1. Projection of velocity

Let us look at the trajectory of a particle given with $\mathbf{X}^* = \mathbf{X} + \mathbf{U}t$, where $\mathbf{X}$ is the position at $t = 0$ and $\mathbf{U}$ is the velocity. We want to calculate the velocity of the projection of $\mathbf{X}^*$, i.e. $\mathbf{u}_p = \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}_p\big|_{t=0}$. At first the velocity of the normalized coordinates is calculated.

$$\mathbf{x}_n = \mathbf{P}\mathbf{X}^* = \mathbf{P}(\mathbf{X} + \mathbf{U}t) = \mathbf{X}_c + \mathbf{U}_c t \tag{B.1}$$

$\mathbf{X}_c$ and $\mathbf{U}_c$ are given in the camera coordinate system.

$$\begin{aligned} \mathbf{X}_c &= \mathbf{R}\mathbf{X} + \mathbf{t} = [X_c,\ Y_c,\ Z_c]^T \\ \mathbf{U}_c &= \mathbf{R}\mathbf{U} = [U_c,\ V_c,\ W_c]^T \end{aligned} \tag{B.2}$$

The components of $\mathbf{x}_n$ are given as

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} X_c/Z_c \\ Y_c/Z_c \end{bmatrix}. \tag{B.3}$$

The velocity is calculated as the derivative of the projected point at $t = 0$.

$$\mathbf{u}_n = \begin{bmatrix} u_n \\ v_n \end{bmatrix} = \frac{\mathrm{d}\mathbf{x}_n}{\mathrm{d}t}\bigg|_{t=0} \tag{B.4}$$

As the result describes the velocity of the normalized coordinates it may be called the normalized velocity.

$$\mathbf{u}_n = \frac{1}{Z_c^2}\begin{bmatrix} U_c Z_c - W_c X_c \\ V_c Z_c - W_c Y_c \end{bmatrix} = \frac{1}{Z_c}\begin{bmatrix} U_c - x_n W_c \\ V_c - y_n W_c \end{bmatrix}. \tag{B.5}$$

The distortion can be included into the calculation in the same manner.

$$\mathbf{u}_d = \begin{bmatrix} u_d \\ v_d \end{bmatrix} = \left.\frac{\mathrm{d}\mathbf{x}_d}{\mathrm{d}t}\right|_{t=0} \tag{B.6}$$

The resulting velocity is given as follows.

$$\mathbf{u}_d = (1 + k_1 r_n^2 + k_2 r_n^4 + k_5 r_n^6)\mathbf{u}_n + 2s_n(k_1 + 2k_2 r_n^2 + 3k_5 r_n^4)\mathbf{x}_n + \mathbf{dx}_t \tag{B.7}$$

with

$$\mathbf{dx}_t = \begin{bmatrix} 2k_3(u_n y_n + x_n v_n) + 2k_4(s_n + 2x_n u_n) \\ 2k_3(s_n + 2y_n v_n) + 2k_4(u_n y_n + x_n v_n) \end{bmatrix} \tag{B.8}$$

where $r_n^2 = x_n^2 + y_n^2$ and $s_n = x_n u_n + y_n v_n$.
The velocity of the image point in pixel units is calculated as follows.

$$\mathbf{u}_p = \left.\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}_p\right|_{t=0} = \begin{bmatrix} f_x u_d \\ f_y v_d \end{bmatrix} \tag{B.9}$$

B.2. Reconstruction of velocity

The reconstruction of a real world position from image points of 2 or more cameras has been handled in the main part of this thesis. With the projection equations for the velocity (derived in the previous section) the velocity can be reconstructed in a similar fashion. For simplicity the reconstruction does not take distortion into account.

Reformulating Equation B.5 and combining with 4.22 results in an equation system for one camera (analogous to Equation 4.22).

$$\mathbf{A}_m = \begin{bmatrix} x_m \mathbf{p}_m^{3T} - \mathbf{p}_m^{1T}, & 0,\ 0,\ 0 \\ y_m \mathbf{p}_m^{3T} - \mathbf{p}_m^{2T}, & 0,\ 0,\ 0 \\ u_m \mathbf{p}_m^{3T}, & x_m \mathbf{q}_m^{3T} - \mathbf{q}_m^{1T} \\ v_m \mathbf{p}_m^{3T}, & y_m \mathbf{q}_m^{3T} - \mathbf{q}_m^{2T} \end{bmatrix} \tag{B.10}$$

with the rows of the camera matrix $\mathbf{P}_m = \begin{bmatrix} \mathbf{p}_m^{1T} \\ \mathbf{p}_m^{2T} \\ \mathbf{p}_m^{3T} \end{bmatrix}$. $\mathbf{q}_m^{nT}$ consists of only the first 3 elements of the rows of $\mathbf{p}_m^{nT}$. The equation $\mathbf{A}\mathbf{\Phi} = 0$ is solved for $\mathbf{\Phi} = [X,\ Y,\ Z,\ H,\ U,\ V,\ W]^T$ where H is the homogeneous coordinate of the position.

As this solution only minimizes the norm $||\mathbf{A}\mathbf{\Phi}||$ the result might produce erroneous results as velocity and position need not be of similar scale. The velocity reconstruction is preferably divided into reconstruction of the

position with a subsequent reconstruction of the velocity by inserting the position $\mathbf{X}$ into equation B.5 and solving the resulting equation system for $\mathbf{U}$.

$$\begin{bmatrix} -u_m \mathbf{p}_m^{3T}\mathbf{X} \\ -v_m \mathbf{p}_m^{3T}\mathbf{X} \end{bmatrix} = \begin{bmatrix} x_m \mathbf{q}_m^{3T} - \mathbf{q}_m^{1T} \\ y_m \mathbf{q}_m^{3T} - \mathbf{q}_m^{2T} \end{bmatrix} \mathbf{U} \tag{B.11}$$

with $\mathbf{X} = [X,\ Y,\ Z,\ H]^T$ and $\mathbf{U} = [U,\ V,\ W]^T$.

B.3. Reconstruction with event driven data

Under the assumption of constant velocity in a given time interval, the 3D pathline of a particle can be directly reconstructed from DVS events. We start with a set of events generated by a passing particle in a given time range. We want to reconstruct the particle position and its velocity under the assumption that all events are generated by the same particle and no outliers are present. The reconstruction procedure builds upon the procedure given by Hartley, see Section 4.2.4. We approximate the particles' path with a first order relation $\mathbf{X} = \mathbf{X}_0 + \mathbf{U}t$, where $\mathbf{X}_0$ is the position at time $t = 0$ and $\mathbf{U}$ is the average velocity in the time range.

Using normalized coordinates the projection of the position is given by $\mathbf{x}_n = \mathbf{P}_m(\mathbf{X}_0 + \mathbf{U}t)$, with the camera matrix $\mathbf{P}_m$ from the m^{th} camera. Analog to the derivation of Equation 4.22, taking the cross product of both sides of the previous equation gets rid of the homogeneous scale factor. This leads to 3 equations for each event, of which only 2 are linearly independent.

$$\mathbf{x}_n \times (\mathbf{P}_m \mathbf{X}_0 + \mathbf{P}_m \mathbf{U}t) = 0 \tag{B.12}$$

Considering a single event $[x,\ y,\ t]$ leads to the following equations:

$$\begin{bmatrix} x\mathbf{p}_m^{3T} - \mathbf{p}_m^{1T}, & (x\mathbf{q}_m^{3T} - \mathbf{q}_m^{1T})t \\ y\mathbf{p}_m^{3T} - \mathbf{p}_m^{2T}, & (y\mathbf{q}_m^{3T} - \mathbf{q}_m^{2T})t \end{bmatrix} \mathbf{\Phi} = 0 \tag{B.13}$$

with the same definitions as in Equation B.10.

For a set of events from multiple cameras equations are collected and solved in the same manner as for the reconstruction of of a single point from multiple projections. A total of at least four image points from two cameras are required to solve the system, where each camera view provides two image points.

Figure B.1 shows the reprojection of a reconstructed pathline from artificial data.

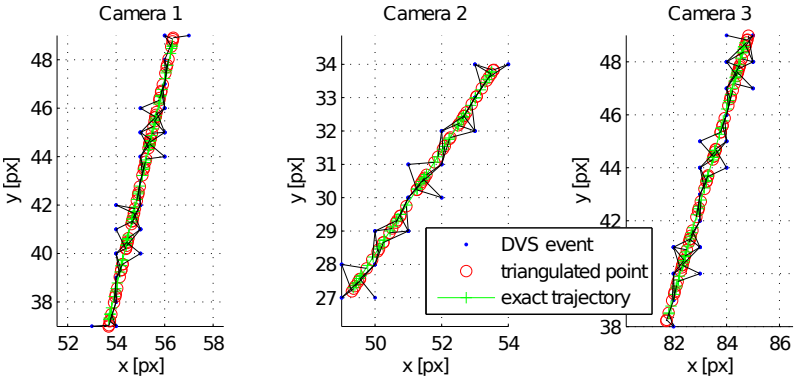


Figure B.1.: Reprojection of a reconstructed pathline from a set of recorded events from 3 calibrated cameras. The velocity is assumed to be constant.

Figure B.2 shows the histogram of the triangulation error for the position and velocity of the reconstructed pathline.

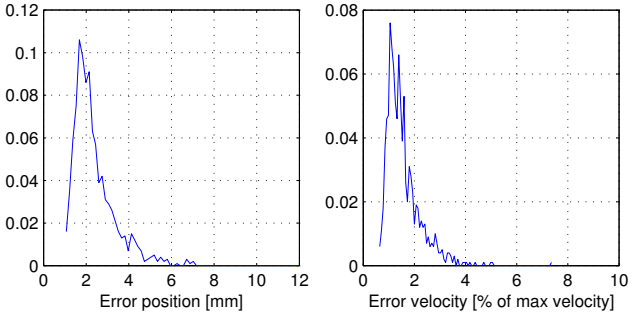


Figure B.2.: Histogram of triangulation errors of position and velocity for a reconstruction from event driven data.

This method was not employed in the main part of this thesis as it only provides a reconstruction under the assumption that all events are generated by the same moving particle, does not deliver information on the association problem (event to particle) and is a batch process where a collection of events has to be processed. It is however included in the thesis

for completion and as a possible means of initialization for the 3D Kalman filter.

B.4. Reconstruction error of position and velocity

The triangulation error in world coordinates will be defined as the distance of the triangulated point to the line of sight of each camera, for which the derivation is given as follows. We start with the projective equation for normalized coordinates in the camera coordinate system.

$$\begin{bmatrix} x_n \\ y_n \\ 1 \end{bmatrix} Z_c = \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} \tag{B.14}$$

with X_c, Y_c and Z_c in the camera coordinate system.
The set of points $\{\mathbf{X}_c\}$ with $Z_c > 0$ (the point has to be in front of the camera), satisfying Equation B.14, is projected onto the same image point and is indistinguishable from the viewpoint of the camera. Rearranging for $\mathbf{X}$ results in a line equation with parameter $w := Z_c$, giving the equation for the line of sight.

$$\mathbf{X}(w) = (\mathbf{R}^T\mathbf{x}_n)w - \mathbf{R}^T\mathbf{t} \tag{B.15}$$

The triangulation error is given by $\epsilon_{tri} = \mathbf{X}_{min} - \mathbf{X}^*$ where $\mathbf{X}_{min}$ is the point with the minimal distance of $\mathbf{X}(w)$ to the triangulated point $\mathbf{X}^*$. By minimizing $||\epsilon_{tri}||$ in a least squares approach the point $\mathbf{X}_{min}$ can be calculated. It is easily done in the camera coordinate system using Equation B.14.

$$\mathbf{X}_{min,c} = \frac{\mathbf{x}_n^T\mathbf{X}_c^*}{\mathbf{x}_n^T\mathbf{x}_n}\mathbf{x}_n \tag{B.16}$$

The triangulation error in world coordinates is given by

$$\epsilon_{tri} = \mathbf{R}^T\mathbf{X}_{min,c} + \mathbf{T} - \mathbf{X}^*. \tag{B.17}$$

The difference between reprojection error and triangulation error comes for one from the units used (px versus m). The triangulation error additionally includes a dependence on the distance of the triangulated point to the camera center, and will increase linearly with the distance Z_c.

The triangulation error for the reconstruction of the velocity follows the same procedure. The projection of the velocity given by Equation B.5 is reformulated as

$$\begin{bmatrix} u_n \\ v_n \\ 0 \end{bmatrix} Z_c = \begin{bmatrix} U_c \\ V_c \\ W_c \end{bmatrix} - \begin{bmatrix} x_n \\ y_n \\ 1 \end{bmatrix} W_c. \tag{B.18}$$

Rearranged in matrix notation then gives

$$\mathbf{U}_c = W_c \mathbf{x}_n + Z_c \mathbf{u}_n. \tag{B.19}$$

In the camera coordinate system the velocity is given by

$$\mathbf{U}_c = \mathbf{R}\mathbf{U}. \tag{B.20}$$

By minimizing the velocity error

$$||\epsilon||^2 = ||\mathbf{U}_c - \mathbf{U}_c^*||^2 \tag{B.21}$$

with a given or triangulated velocity $\mathbf{U}_c^*$ the best matching velocity $\mathbf{U}_{min,c}$ can be calculated as

$$\mathbf{U}_{min,c} = \frac{(\mathbf{x}_n^T\mathbf{x}_n)(\mathbf{x}_n^T\mathbf{U}_c^*) - (\mathbf{x}_n^T\mathbf{u}_n)(\mathbf{x}_n^T\mathbf{X}_c^*)}{(\mathbf{x}_n^T\mathbf{x}_n)^2}\mathbf{x}_n + \frac{\mathbf{x}_n^T\mathbf{X}_c^*}{\mathbf{x}_n^T\mathbf{x}_n}\mathbf{u}_n. \tag{B.22}$$

B.5. Estimation of triangulation uncertainty

The uncertainty calculation is based on a linear relation between statistics for normally distributed variables. As the projection equation is not a linear function the first order approximation is used.

First we look at the propagation of uncertainty for a linear relation between two sets of variables $y_i = \sum_{j=1}^{N} a_{ij}x_j$. In matrix form the relation may be written as $\mathbf{y} = \mathbf{A}\mathbf{x}$.

The covariance matrix of $\mathbf{x}$ is given as

$$\mathbf{\Sigma}_x = \begin{bmatrix} \sigma_1^2 & cov_{12} & \dots \\ cov_{21} & \sigma_2^2 & \dots \\ \vdots & \vdots & \ddots \end{bmatrix}. \tag{B.23}$$

The covariance matrix for $\mathbf{y}$ is then given with

$$\mathbf{\Sigma}_y = \mathbf{A}\mathbf{\Sigma}_x\mathbf{A}^T. \tag{B.24}$$

For a non-linear relation between $\mathbf{y}$ and $\mathbf{x}$ the first order approximation of $\mathbf{A}$ is given by the Jacobian matrix $\mathbf{J} = \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{x}}$.

This leads to

$$\mathbf{\Sigma}_y = \mathbf{J}\mathbf{\Sigma}_x\mathbf{J}^T. \tag{B.25}$$

For an implicit equation $f(\mathbf{x}, \mathbf{y}) = 0$ one can reformulate the approach. In a first order approximation the covariance matrices of $\mathbf{x}$ and $\mathbf{y}$ are related as

$$\mathbf{J}_y \mathbf{\Sigma}_y \mathbf{J}_y^T = \mathbf{J}_x \mathbf{\Sigma}_x \mathbf{J}_x^T. \tag{B.26}$$

$\mathbf{J}_x$ is the Jacobian of f in regard to $\mathbf{x}$ and $\mathbf{J}_y$ is the Jacobian of f in regard to $\mathbf{y}$. The covariance matrix of $\mathbf{y}$ can then be calculated as

$$\mathbf{\Sigma}_y = \mathbf{J}_y^{\Delta} (\mathbf{J}_x \mathbf{\Sigma}_x \mathbf{J}_x^T) \mathbf{J}_y^{\Delta\, T} \tag{B.27}$$

with the pseudo inverse $\mathbf{J}_y^{\Delta} = (\mathbf{J}_y^T \mathbf{J}_y)^{-1} \mathbf{J}_y^T$.

For the reconstruction Equation 4.22 we have $f(\mathbf{x}, \mathbf{X}) = 0$ where $\mathbf{x}$ is the image coordinate and $\mathbf{X}$ is the world coordinate. The Jacobi matrices are given as follows.

$$\mathbf{J}_{\mathbf{X}} = \begin{bmatrix} x_1 \mathbf{q}_1^{3T} - \mathbf{q}_1^{1T} \\ y_1 \mathbf{q}_1^{3T} - \mathbf{q}_1^{2T} \\ x_2 \mathbf{q}_2^{3T} - \mathbf{q}_2^{1T} \\ y_2 \mathbf{q}_2^{3T} - \mathbf{q}_2^{2T} \end{bmatrix} \tag{B.28}$$

The definitions are the same as in Equation B.10.

$$\mathbf{J}_{\mathbf{x}} = \begin{bmatrix} z_1 & 0 & 0 & 0 \\ 0 & z_1 & 0 & 0 \\ 0 & 0 & z_2 & 0 \\ 0 & 0 & 0 & z_2 \end{bmatrix} \tag{B.29}$$

with $z_m = \mathbf{p}_m^{3T} \mathbf{X}$ for $m = (1,\ 2)$.

An estimate for $\mathbf{\Sigma}_{\mathbf{x}}$ is provided by the Kalman filter, from which $\mathbf{\Sigma}_{\mathbf{X}}$ may be calculated.

For the triangulation of velocity one may derive a similar formulation.

$$\mathbf{J}_{\mathbf{X}} = \begin{bmatrix} x_1 \mathbf{q}_1^{3T} - \mathbf{q}_1^{1T} & 0,\ 0,\ 0 \\ y_1 \mathbf{q}_1^{3T} - \mathbf{q}_1^{2T} & 0,\ 0,\ 0 \\ u_1 \mathbf{q}_1^{3T} - \mathbf{q}_1^{1T} & x_1 \mathbf{q}_1^{3T} - \mathbf{q}_1^{1T} \\ v_1 \mathbf{q}_1^{3T} - \mathbf{q}_1^{2T} & y_1 \mathbf{q}_1^{3T} - \mathbf{q}_1^{2T} \\ x_2 \mathbf{q}_2^{3T} - \mathbf{q}_2^{1T} & 0,\ 0,\ 0 \\ y_2 \mathbf{q}_2^{3T} - \mathbf{q}_2^{2T} & 0,\ 0,\ 0 \\ u_2 \mathbf{q}_2^{3T} - \mathbf{q}_2^{1T} & x_2 \mathbf{q}_2^{3T} - \mathbf{q}_2^{1T} \\ v_2 \mathbf{q}_2^{3T} - \mathbf{q}_2^{2T} & y_2 \mathbf{q}_2^{3T} - \mathbf{q}_2^{2T} \end{bmatrix} \tag{B.30}$$

$$\mathbf{J_x} = \begin{bmatrix} z_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & z_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ w_1 & 0 & z_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & w_1 & 0 & z_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & z_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & z_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & w_2 & 0 & z_2 & 0 \\ 0 & 0 & 0 & 0 & 0 & w_2 & 0 & z_2 \end{bmatrix} \tag{B.31}$$

with $z_m = \mathbf{p}_m^{3T}\mathbf{X}$, $w_m = \mathbf{p}_m^{3T}\mathbf{U}$ for $m = (1,\ 2)$.

Appendix C

Calibration

C.1. Rodrigues formula

A rotation in $\mathbb{R}^3$ has 3 degrees of freedom but is described by a 3×3 matrix with 9 entries (the rotation matrix $\mathbf{R}$) in the framework of projective transformation. The Rodrigues formula provides a method to describe the rotation of a vector $\mathbf{v}$ around a unit vector $\mathbf{k}$ about the angle θ (in the right hand sense). This formulation allows the conversion of the rotation matrix $\mathbf{R}$ into a rotation vector ω with 3 entries (and vice versa). This conversion is useful for calibration as the unknown entries of the rotation vector are equal to the degrees of freedom and no additional constraints have to provided (as would be the case if using the rotation matrix). The rotation is described as follows.

$$\mathbf{v}_{rot} = \mathbf{v}\cos(\theta) + (\mathbf{k} \times \mathbf{v})\sin(\theta) + \mathbf{k}(\mathbf{k} \cdot \mathbf{v})(1 - \cos(\theta)) \tag{C.1}$$

The rotation is summarized with the rotation vector $\omega = \theta\mathbf{k}$. The rotation can be given in matrix notation as $\mathbf{v}_{rot} = \mathbf{R}\mathbf{v}$. The rotation matrix is calculated from the rotation vector with following equation,

$$\mathbf{R} = \mathbf{I} + \sin(\theta)[\mathbf{k}]_\times + (1 - \cos(\theta))(\mathbf{k}\mathbf{k}^T - \mathbf{I}). \tag{C.2}$$

C.2. Skew–symmetric matrix

The cross product of vectors $\mathbf{t}$ and $\mathbf{x}$ can be rewritten as a matrix multiplication with

$$\mathbf{t} \times \mathbf{x} = [\mathbf{t}]_\times \mathbf{x}, \tag{C.3}$$

where the operator $[\cdot]_\times$ represents the skew–symmetric matrix,

$$[\mathbf{t}]_\times = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix}. \tag{C.4}$$

C.3. Solution for homogeneous equation system

The following derivation is based on Hartley & Zisserman (2003), page 563. Reconstruction seeks a solution to the homogeneous system $\mathbf{Ax} = 0$ other than the trivial solution $\mathbf{x} = \mathbf{0}$. Since the solution $\mathbf{x}$ is only defined up to a scale factor ($k\mathbf{x}$ is also a solution), a possible approach would be to set one component of $\mathbf{x}$ to 1 and solve the resulting inhomogeneous system (e.g. via least squares, since the number of equations is larger than the number of variables). The problem with this approach arises when the real solution of the prescribed component is near or equal to zero, leading to unstable results.

The solution of $\mathbf{Ax} = 0$ can be reformulated as a minimization of the norm $||\mathbf{Ax}||$ subject to $||\mathbf{x}|| = 1$ to ensure a non-trivial solution.

Using the singular value decomposition we rewrite $\mathbf{A} = \mathbf{UDV}'$, therefore having to minimize $||\mathbf{UDV}'\mathbf{x}||$. By multiplication with the orthogonal matrices $\mathbf{U}$ and $\mathbf{V}'$ the norm is unchanged and the following relations hold: $||\mathbf{UDV}'\mathbf{x}|| = ||\mathbf{DV}'\mathbf{x}||$ and $||\mathbf{x}|| = ||\mathbf{V}'\mathbf{x}||$. Substitution with $\mathbf{y} = \mathbf{V}'\mathbf{x}$ allows to restate the problem as a minimization of $||\mathbf{Dy}||$ subject to $||\mathbf{y}|| = 1$. $\mathbf{D}$ is a diagonal matrix with diagonal entries in descending order, thus the norm is minimized with $\mathbf{y} = [0\ 0\ ...\ 0\ 1]^\mathbf{T}$ and therefore $\mathbf{x} = \mathbf{Vy}$ is equal to the last column of $\mathbf{V}$.

A more optimal approach is achieved by minimizing the geometrical error, for more information see Hartley & Zisserman (2003) page 304.

C.4. Calibration code

This Section will give an overview of the calibration code and the equations to calculate the Jacobian matrices for the optimization. The calibration code is implemented in Matlab and uses Matlab routines for optimization as well as functions from the Caltech calibration toolbox. The function is called by

`[opt,args]=auto_calibration(fc,cc,kc,om,t,args,options)`

The last input `options` is equal to the options from `optimset` for the optimization function `nonlsqlin`.

C.4.1. Input

The code accepts initial values of the intrinsic and extrinsic parameters. for the optimization in `fc,cc,kc,om,t`. The values might be omitted and

will then be initialized by linear methods.

The main input arguments are part of the structure `args`. Following is a list of the main inputs necessary for the code to work.

- args.M: Number of cameras.
- args.K: Number of views.
- args.x{k,m}: Image coordinates of the target for each view and camera.
- args.X{k}: 3D coordinates of the target. These required when the target is known (e.g. the LED board).

The following list gives the additional options available in the code. The options are to be defined in the structure `args` if required.

- args.fix_fc: Fix focal lengths to the input f_c.
- args.fix_cc: Fix principal points to the input c_c.
- args.fix_kc: Fix distortion coefficients to k_c (the distortion is defined by a 1×5 vector of the distortion coefficients k_{1-5}).
- args.copy_cam: Copy intrinsic parameters. The format is a cell matrix with entries for each associated camera pair, e.g.: args.copy_cam={[1 2]} for camera 1 and camera 2 sharing the same intrinsic parameters.
- args.fix_str: Do not optimize for structure. If this is set to false, the code will also optimize the coordinate points of the target. This can be useful if the target is not accurately manufactured.
- args.fix_str_point: Fix certain points of the structure. Only used if args.fix_str is set to false. The array fix_str_point contains the indices of the target points to fix.
- args.fix_cam: Cameras are regarded as fixed relative to each other, the output will be $om_c\{m\}$, $t_c\{m\}$ for the camera positions and $om_t\{k,m\}$, $t_t\{k,m\}$ for the positions of the target relative to the cameras for each view.
- args.max_rpe: The maximum reprojection error for a point to be considered valid. If this parameter is provided, the code performs a second optimization run without the image points with a higher reprojection error than args.max_rpe.

C.4.2. Results

All results are saved in the structure `opt` as cell matrices where the index m corresponds to the camera and the index k corresponds to the respective view. `args` can be requested as output, in which case the reprojection errors are saved in args.rpe and information on the validity of image points is saved in args.val. An image point is considered invalid if its reprojection error is higher than args.max_rpe. The following list gives an overview of the output collected in `opt`.

- opt.fc{m}, opt.cc{m}, opt.kcm: Intrinsic parameters for each camera.
- opt.omc{m}, opt.tc{m}: Extrinsic parameters of cameras (if not specified, opt.omc{1}= [0, 0, 0], opt.tc{1}= [0, 0, 0])
- opt.omt{k, m}, opt.tt{k, m}: Pose of cameras in target centric coordinate system (for each view k).
- opt.x{k}{m}: Image coordinates in homogeneous coordinates ($3 \times N$ matrix, where N is the number of points).
- opt.X{k}: 3D coordinates in homogeneous coordinates ($4 \times N$ matrix, where N is the number of points).

C.4.3. Jacobi matrices

The optimization requires the Jacobi matrices of the projection equation for an efficient calculation. Following is an overview of the computation of the Jacobi matrices.

The intrinsic parameters are collected in vector form as

$$\mathbf{\Phi}_{in} = \begin{bmatrix} f_{c,x} & f_{c,y} & c_{c,x} & c_{c,y} & k_{c,1} & k_{c,2} & k_{c,3} & k_{c,4} & k_{c,5} \end{bmatrix}^T . \quad \text{(C.5)}$$

The corresponding Jacobi matrix is

$$\frac{\partial \mathbf{x}_p}{\partial \mathbf{\Phi}_{in}} = \begin{bmatrix} x_d & 0 & 1 & 0 & r^2 x_n & r^4 x_n & 2x_n y_n & r^2 + 2x_n^2 & r^6 x_n \\ 0 & y_d & 0 & 1 & r^2 y_n & r^4 y_n & r^2 + 2y_n^2 & 2x_n y_n & r^6 y_n \end{bmatrix} . \quad \text{(C.6)}$$

The extrinsic parameters are divided in the following way, with subscript $(\cdot)_c$ marking a camera centric coordinate system and subscript $(\cdot)_t$ marking a target coordinate system. $\mathbf{X}_t$ are the predefined coordinates of the target in the target coordinate system.

The optimization solves for the following extrinsic parameters:

- $\mathbf{R}_{c,m}$, $\mathbf{t}_{c,m}$: The pose of the cameras in the world coordinate system. The pose of the first camera $\mathbf{R}_{c,0}$, $\mathbf{t}_{c,0}$ is prescribed and not solved for.
- $\mathbf{R}_{t,k}$, $\mathbf{t}_{t,k}$: The pose of the first camera in the target coordinate system for each view of the target.

The extrinsic parameters are collected in vector form for each camera as

$$\mathbf{\Phi}_c = [r_{c,11},\ r_{c,21},\ r_{c,31},\ r_{c,12},\ r_{c,22},\ r_{c,32},\ r_{c,13},\ r_{c,23},\ r_{c,33},\ t_{c,1},\ t_{c,2},\ t_{c,3}]^T, \tag{C.7}$$

and for each view of the target as

$$\mathbf{\Phi}_t = [r_{t,11},\ r_{t,21},\ r_{t,31},\ r_{t,12},\ r_{t,22},\ r_{t,32},\ r_{t,13},\ r_{t,23},\ r_{t,33},\ t_{t,1},\ t_{t,2},\ t_{t,3}]^T, \tag{C.8}$$

where $r_{t,ij}$ and $r_{c,ij}$ are the elements of the rotation matrices $\mathbf{R}_t$ and $\mathbf{R}_c$ and $t_{t,ij}$ and $t_{c,ij}$ are the elements of the vectors $\mathbf{t}_t$ and $\mathbf{t}_c$.

The following equations calculate the Jacobi matrices of the individual steps of the projection equation. The definitions of the variables are found in Section 4.2.2.

$$\frac{\partial \mathbf{x}_p}{\partial \mathbf{x}_d} = \begin{bmatrix} f_{c,x} & 0 \\ 0 & f_{c,y} \end{bmatrix} \tag{C.9}$$

$$\frac{\partial \mathbf{x}_d}{\partial \mathbf{x}_n} = \begin{bmatrix} \gamma + \gamma_r x_n^2 + 2k_{c,3}y_n + 6k_{c,4}x_n & \gamma_r x_n y_n + 2k_{c,3}x_n + 2k_{c,4}y_n \\ \gamma_r x_n y_n + 2k_{c,3}x_n + 2k_{c,4}y_n & \gamma + \gamma_r y_n^2 + 6k_{c,3}y_n + 2k_{c,4}x_n \end{bmatrix} \tag{C.10}$$

with

$$\gamma = 1 + k_{c,1}r^2 + k_{c,2}r^4 + k_{c,5}r^6, \tag{C.11}$$

$$\gamma_r = 2k_{c,1} + 4k_{c,2}r^2 + 6k_{c,5}r^4. \tag{C.12}$$

The homogeneous formulation requires an intermediate Jacobian to account for division by the homogeneous coordinate.

$$\frac{\partial \mathbf{x}_n}{\partial \mathbf{X}_c} = \begin{bmatrix} \frac{1}{Z_c} & 0 & -\frac{X_c}{Z_c^2} \\ 0 & \frac{1}{Z_c} & -\frac{Y_c}{Z_c^2} \end{bmatrix} \tag{C.13}$$

The target centric coordinates are related to the camera centric coordinates by Equation 4.34. Following is a summary of the equation in shortened form.

$$\mathbf{X}_c = \mathbf{R}_{c,m}\mathbf{R}_{c,1}^T(\mathbf{R}_{t,k}\mathbf{X}_t + \mathbf{t}_{t,k} - \mathbf{t}_{c,1}) + \mathbf{t}_{c,m} \tag{C.14}$$

$$\frac{\partial \mathbf{X}_c}{\partial \mathbf{\Phi}_t} = \mathbf{R}_{c,m}\mathbf{R}_{c,1}^T \begin{bmatrix} X_t & 0 & 0 & Y_t & 0 & 0 & Z_t & 0 & 0 & 1 & 0 & 0 \\ 0 & X_t & 0 & 0 & Y_t & 0 & 0 & Z_t & 0 & 0 & 1 & 0 \\ 0 & 0 & X_t & 0 & 0 & Y_t & 0 & 0 & Z_t & 0 & 0 & 1 \end{bmatrix} \tag{C.15}$$

The Jacobian with respect to $\mathbf{\Phi}_c$ can be calculated with

$$\frac{\partial \mathbf{X}_c}{\partial \mathbf{\Phi}_c} = \begin{bmatrix} X_t^* & 0 & 0 & Y_t^* & 0 & 0 & Z_t^* & 0 & 0 & 1 & 0 & 0 \\ 0 & X_t^* & 0 & 0 & Y_t^* & 0 & 0 & Z_t^* & 0 & 0 & 1 & 0 \\ 0 & 0 & X_t^* & 0 & 0 & Y_t^* & 0 & 0 & Z_t^* & 0 & 0 & 1 \end{bmatrix}, \tag{C.16}$$

where

$$\mathbf{X}_t^* = [X_t^*,\ Y_t^*,\ Z_t^*]^T = \mathbf{R}_{c,1}^T(\mathbf{R}_{t,k}\mathbf{X}_t + \mathbf{t}_{t,k} - \mathbf{t}_{c,1}). \tag{C.17}$$

The structure may be included in the optimization with the respective Jacobi matrix given as

$$\frac{\partial \mathbf{X}_c}{\partial \mathbf{\Phi}_\mathbf{X}} = \mathbf{R}_{c,m}\mathbf{R}_{c,1}^T\mathbf{R}_{t,k}, \tag{C.18}$$

where

$$\mathbf{\Phi}_\mathbf{X} = [X_t,\ Y_t,\ Z_t]^T. \tag{C.19}$$

With the chain rule the overall Jacobi matrix can be calculated for $\mathbf{\Phi}_c$, $\mathbf{\Phi}_t$ and $\mathbf{\Phi}_\mathbf{X}$.

$$\frac{\partial \mathbf{x}_p}{\partial \mathbf{\Phi}_{(\cdot)}} = \frac{\partial \mathbf{x}_p}{\partial \mathbf{x}_d}\frac{\partial \mathbf{x}_d}{\partial \mathbf{x}_n}\frac{\partial \mathbf{x}_n}{\partial \mathbf{X}_c}\frac{\partial \mathbf{X}_c}{\partial \mathbf{\Phi}_{(\cdot)}} \tag{C.20}$$

Appendix D

Tracking

D.1. Simulated data

The tracking codes were tested on simulated data. A random trajectory in $\mathbb{R}^3$ was generated in Matlab and projected onto a set of virtual DVS cameras. The projected trajectory had to be converted to AER to simulate the DVS.

The image point is assumed to be a Gaussian distribution given by

$$I = I_{bg} + I_{max} e^{-\frac{||\mathbf{x}(t)-\mathbf{x}_p||^2}{2\sigma^2}} \tag{D.1}$$

where $\mathbf{x}_p$ is the coordinate of a pixel of the DVS and $\mathbf{x}(t)$ is the time dependent path of the particle's image. An event is generated at integer values of $\log(I)/C$ where C is some unknown constant, it is basically an analog to $C_{threshold}$. C is set accordingly to reproduce a realistic event count per pixel as observed in the experiments. The following equation is solved for the time stamp t for every pixel at prescribed values of $\log(I)/C$.

$$2\sigma^2 \frac{e^{\log(I)/C} - I_{bg}}{I_{max}} = -||\mathbf{x}(t) - \mathbf{x}_p||^2 \tag{D.2}$$

The prescribed levels of $\log(I)$ are distributed uniformly in the interval between the minimum level of $\log(I_{bg})$ and the maximum level of $\log(I_{max} + I_{bg})$.

D.2. Ricatti equation

The Riccati equation describes the steady state solution for the a priori covariance matrix $\mathbf{P}_{k+1|k}$. This case is equal to the converged state with

constant matrices (by assuming a constant timestep Δt).

$$\mathbf{P}_\infty = \mathbf{F}\left(\mathbf{P}_\infty - \mathbf{P}_\infty\mathbf{H}^T\left(\mathbf{H}\mathbf{P}_\infty\mathbf{H}^T + \mathbf{R}\right)^{-1}\mathbf{H}\mathbf{P}_\infty\right)\mathbf{F}^T + \mathbf{Q} \tag{D.3}$$

An analytic solution for the model by Reid can be derived. It is the solution of a quartic equation. The covariance matrix $\mathbf{P}_\infty$ for the 1–dimensional case is given by:

$$\mathbf{P}_\infty = \begin{bmatrix} \sigma_x^2 & \sigma_{x,u} \\ \sigma_{x,u} & \sigma_u^2 \end{bmatrix} \tag{D.4}$$

with

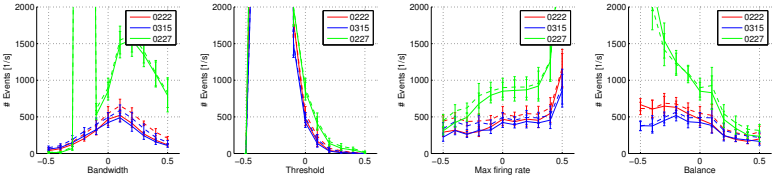
$$\begin{aligned} \sigma_x^2 &= \tfrac{\sigma_{x,u}^2}{q\Delta t} - r \\ \sigma_u^2 &= q\Delta t + \tfrac{\sigma_{x,u}}{\Delta t} - \tfrac{qr}{\sigma_{x,u}} \\ \sigma_{x,u} &= \tfrac{q\Delta t^2}{4}\left(D + \sqrt{2D}\right), \text{ with } D = 1 + \sqrt{1 + 16\tfrac{r}{q\Delta t^3}}. \end{aligned} \tag{D.5}$$

For typical values of the parameters $q = 10^7$, $r = 1$ and $\Delta t = 0.5\ ms$ the solution for the covariances is $\sigma_x = 0.55$, $\sigma_{x,u} = 80.78$, $\sigma_u = 206$. This result is the range achieved after processing real data (i.e. it is in the same order as the converged covariance matrix) and will be used for initialization.

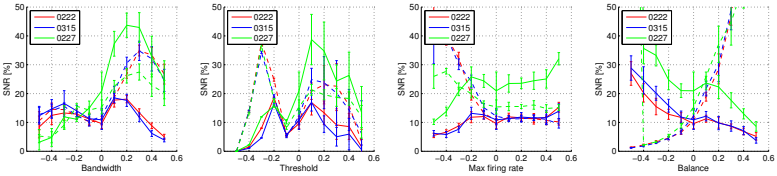
Appendix E

Bias parameter study

Figures E.1 and E.2 show the results of a parameter study of the bias settings with an aperture of respectively $f/\#2.8$ and $f/\#5.6$. The DVS cameras are recording the flow of bubbles in the wind tunnel at a low speed of $2m/s$ and all cameras are situated at the wind tunnel entry perpendicular to the free stream. The illumination uses LED pairs A, C, D and E at full power; all LED pairs are situated in the wind tunnel. The fast bias setting represents the baseline. For each parameter a study is performed in the range of -0.5 to 0.5 while the other parameters are set to nominal values at 0. The results show no clear improvement for both apertures. The baseline already presents a good operating point for the measurements.

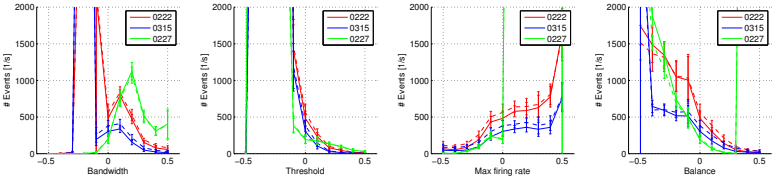


(a) Bandwidth (# events) **(b)** Threshold (# events) **(c)** Max firing rate (# events) **(d)** Balance (# events)

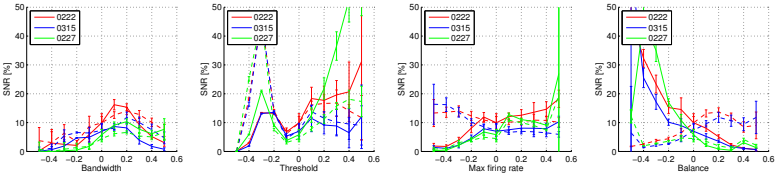


(e) Bandwidth (SNR) **(f)** Threshold (SNR) **(g)** Max firing rate (SNR) **(h)** Balance (SNR)

Figure E.1.: Variation of bias settings with $f/2.8$, full lines are ON events, the dashed lines are OFF events.



(a) Bandwidth (# events) **(b)** Threshold (# events) **(c)** Max firing rate (# events) **(d)** Balance (# events)



(e) Bandwidth (SNR) **(f)** Threshold (SNR) **(g)** Max firing rate (SNR) **(h)** Balance (SNR)

Figure E.2.: Variation of bias settings with $f/5.6$, full lines are ON events, the dashed lines are OFF events.

Bibliography

Anderson, B. & Moore, J. 2012 *Optimal Filtering*. Dover Publications.

Bar-Shalom, Y. & Fortmann, T. E. 1988 *Tracking and Data Association*. Academic Press, Orlando, Florida.

Berner, R. 2011 *Building-Blocks for Event-Based Vision Sensors*. Diss. ETH Nr. 19884, Institute of Neuroinformatics, University of Zürich.

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

Blackman, S. S. 1986 *Multiple-target tracking with radar applications*. Dedham, MA, Artech House, Inc.

Borter, G. 2013 *Erzeugung von Neatraldichten Seifenblasen aus einer Zweiphasenströmung in Mikrokanälen*. MSc. Thesis, Institute of Fluid Dynamics, ETH Zürich.

Bosbach, J., Kühn, M. & Wagner, C. 2009 Large scale particle image velocimetry with helium filled soap bubbles. *Experiments in Fluids* **46** (3), pp. 539–547.

Bouguet, J.-Y. 2008 Camera Calibration Toolbox for Matlab. Available online at `http://www.vision.caltech.edu/bouguetj/calib_doc/`.

Brown, D. C. 1966 Decentering Distortion of Lenses. *Photometric Engineering* **32** (3), pp. 444–462.

Cao, X., Liu, J., Jiang, N. & Chen, Q. 2014 Particle image velocimetry measurement of indoor airflow field: A review of the technologies and applications. *Energy and Buildings* **69** (0), pp. 367 – 380.

Christina Politz, R. G. & Ranasinghe, S. 2013 Ground Based Large Scale Wake Vortex Investigations by Means of Particle Image Velocimetry: A Feasibility Study. In *Advanced In-Flight Measurement Techniques*, pp. 291–304. Springer London, Limited.

Conradt, J., Cook, M., Berner, R., Lichtsteiner, P., Douglas, R. & Delbruck, T. 2009 A pencil balancing robot using a pair of AER dynamic vision sensors. In *IEEE International Symposium on Circuits and Systems, 2009*, pp. pp. 781–784.

Doh, D. H., Lee, C. J., Cho, G. R. & Moon, K. R. 2012 Performances of Volume-PTV and Tomo-PIV. *Open Journal of Fluid Dynamics* **2**, pp. 368.

Drazen, D., Lichtsteiner, P., Häfliger, P., Delbrück, T. & Jensen, A. 2011 Toward real-time particle tracking using an event-based dynamic vision sensor. *Experiments in Fluids* **51** (5), pp. 1465–1469.

Faugeras, O., Luong, Q.-T. & Papadopoulo, T. 2001 *The Geometry of Multiple Images*. MIT Press.

Hanimann, L. 2011 *3D Flow Visualization with a Stereo DVS–System*. Sem. Thesis, Institute of Fluid Dynamics, ETH Zürich.

Hartley, R. & Zisserman, A. 2003 *Multiple View Geometry in Computer Vision*. Cambridge University Press.

Högbom, J. A. 1974 Aperture Synthesis with a Non-Regular Distribution of Interferometer Baselines. *Astronomy and Astrophysics Supplement* **15**, pp. 417.

Honkanen, M. & Nobach, H. 2005 Two-dimensional Gaussian regression for sub-pixel displacement estimation in particle image velocimetry or particle position estimation in particle tracking velocimetry. *Experiments in Fluids* **38** (4), pp. 511–515.

Kalman, R. E. 1960 A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME, Journal of Basic Engineering* (82 (Series D)), pp. 35–45.

Lichtsteiner, P. 2006 *An AER Temporal Contrast Vision Sensor*. Diss. ETH Nr. 16898, Institute of Neuroinformatics, University of Zürich.

Lichtsteiner, P., Posch, C. & Delbruck, T. 2008 A 128 × 128 120 dB 15 μs Latency Asynchronous Temporal Contrast Vision Sensor. *IEEE Journal of Solid-State Circuits* **43** (2), pp. 566–576.

Litzenberger, M., Posch, C., Bauer, D., Belbachir, A., Schon, P., Kohn, B. & Garn, H. 2006 Embedded Vision System for Real-Time Object Tracking using an Asynchronous Transient Vision Sensor. In *12th Digital Signal Processing Workshop, - 4th Signal Processing Education Workshop*, pp. 173–178.

Lobutova, E., Resagk, C., Rank, R. & Müller, D. 2009 Extended Three Dimensional Particle Tracking Velocimetry for Large Enclosures. In *Imaging Measurement Methods for Flow Analysis*, *Notes on Numerical Fluid Mechanics and Multidisciplinary Design*, vol. 106, pp. 113–124. Springer Berlin Heidelberg.

Maas, H.-G. 1993 Complexity analysis for the establishment of image correspondences of dense spatial target fields. *International Archives of Photogrammetry and Remote Sensing* **29**, pp. 102.

Macháček, M. 2002 *A Quantitative Visualization Tool for Large Wind Tunnel Experiments*. Diss. ETH Nr. 14957, Institute of Fluid Dynamics, ETH Zürich.

Matisko, P. & Havlena, V. 2012 Optimality tests and adaptive Kalman filter. *System Identification* **16** (1), pp. 1523–1528.

Meier, A. 2012 *Imaging Laser Doppler Velocimetry*. Diss. ETH Nr. 20498, Institute of Fluid Dynamics, ETH Zürich.

Müller, R. H. G., Scherer, T., Rotger, T., Schaumann, O. & Markwart, M. 1997 Large Body Aircraft Cabin A/C Flow Measurement by Helium Bubble Tracking. *Journal of Flow Visualization and Image Processing* **4** (3), pp. 295–306.

Müller, T. J., ed. 1996 *Fluid Mechanics Measurements*, 2nd edn., p. 409. Hemisphere Publishing Corporation.

Okuno, Y., Fukuda, T., Miwata, Y. & Kobayashi, T. 1993 Development of three- dimensional air flow measuring method using soap bubbles. *JSAE Review* **14** (4), pp. 5055.

Piatkowska, E., Belbachir, A. N., Schraml, S. & Gelautz, M. 2012 Spatiotemporal Multiple Persons Tracking Using Dynamic Vision Sensor. In *The Eighth IEEE Workshop on Embedded Vision*.

Rago, C., Willett, P. & Streit, R. 1995 A comparison of the JPDAF and PMHT tracking algorithms. *IEEE International Conference on Acoustics, Speech, and Signal Processing* **5**, pp. 3571–3574.

Ramm, G. & Faltin, G. 1984 *Some Salient Features of the Time-averaged Ground Vehicle Wake*. Society of Automotive Engineers.

Reid, D. 1979 An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control* **24** (6), pp. 843–854.

Reid, I. & Term, H. 2001 Lecture on Estimation. Department of Engineering Science, University of Oxford.

Rosenstiel, M. & Grigat, R.-R. 2010 Segmentation and classification of streaks in a large-scale particle streak tracking system. *Flow Measurement and Instrumentation* **21** (1), pp. 1–7.

Schraml, S., Belbachir, A., Milosevic, N. & Schn, P. 2010 Dynamic stereo vision system for real-time tracking. In *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1409–1412.

Shelley, T. 2009 Bubble tracking. `http://www.worksmanagement.co.uk/information-technology/features/bubble-tracking/17510/`.

Svoboda, T., Martinec, D. & Pajdla, T. 2005 A Convenient Multi-Camera Self-Calibration for Virtual Environments. *PRESENCE: Teleoperators and Virtual Environments* **14** (4), pp. 407–422.

Tsai, R. Y. 1986 An efficient and accurate camera calibration technique for 3D machine vision. *Proc. IEEE Conference on Computer Vision and Pattern Recognition, 1986* .

Wang, A., Qiu, T. & Shao, L. 2009 A Simple Method of Radial Distortion Correction with Centre of Distortion Estimation. *Journal of Mathematical Imaging and Vision* **35** (3), pp. 165–172.

Publications

D. Borer, T. Rösgen 2009a Forced Alveolar Flows and Mixing in the Lung. *Proc. Appl. Math. Mech.* 9.

D. Borer, T. Rösgen 2009b Forcing of low Reynolds number alveolar flows in the lung. *CMBE09, 1st International Conference on Computational & Mathematical Biomedical Engineering*, Swansea, UK.

D. Borer, T. Rösgen 2010 Mixing processes in the lung based on particle tracking simulations. *17th Congress of the European Society of Biomechanics*, Edinburgh, Scotland.

A. Landolt, D. Borer, A. Meier, and T. Rösgen 2010 Quantitative flow visualization applied to a passive wake control problem. *Aerodynamics of Heavy Vehicles III: Trucks, Buses and Trains*, Potsdam, Germany.

D. Borer, T. Rösgen 2011 Experimental Investigation of Particle–laden Flow in an In–vitro Model of a Single Alveolus. *Euromech Colloquium 521, Biomedical Flows at Low Reynolds Numbers*, Zürich, Switzerland.

D. Borer, S. Yammine, F. Singer, T. Rösgen, P. Latzin, D Obrist 2013 A computer model for single–breath washout with double–tracer gas to help understanding ventilation inhomogeneity. *Dreiländertagung der Deutschen, Schweizerischen und Österreichischen Gesellschaft für Biomedizinische Technik*, Graz, Austria.

D. Borer, T. Rösgen 2014 Large-scale Particle Tracking with Dynamic Vision Sensors *ISFV16 -16th International Symposium on Flow Visualization*, Okinawa, Japan.

Curriculum Vitae

Name	David Johannes Borer
Date of birth	May 5, 1979
Place of birth	Johannesburg, South Africa
Nationality	Swiss
Since Jan. 2008	PhD student, research and teaching assistant under the the supervision of Prof. Dr. T. Rösgen at the Institue of Fluid Dynamics, ETH Zürich, Switzerland
Oct. 2005 to Nov. 2007	M.Sc. in Mechanical Engineering, ETH Zürich, Switzerland
Oct. 2001 to Jul. 2005	B.Sc. in Mechanical Engineering, ETH Zürich, Switzerland
Oct. 2000 to Jan. 2001	Studies in Computer Science, ETH Zürich, Switzerland
May 2000	Matura, Kollegium St. Fidelis, Stans, Switzerland