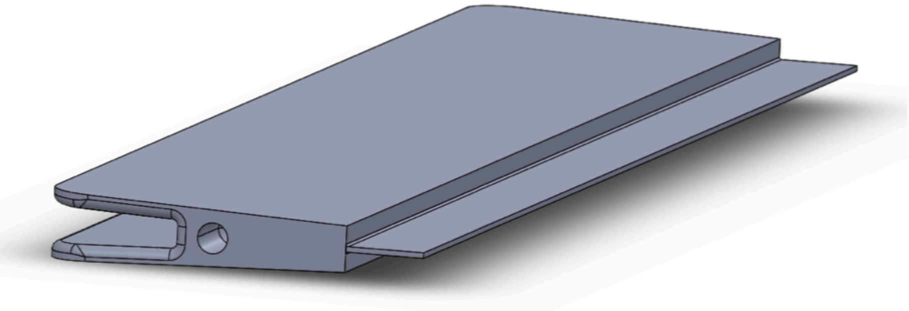
Master’s thesis

NTNU
Norwegian University of Science and Technology
Faculty of Engineering
Department of Energy and Process Engineering

Carsten Cleve-Hansen

Splitter plate’s effect on trailing edge vortex shedding and fluid structure interaction

Master’s thesis in MTPROD
Supervisor: Pål-Tore Storli
June 2020



NTNU
Kunnskap for en bedre verden

Carsten Cleve-Hansen

Splitter plate's effect on trailing edge vortex shedding and fluid structure interaction

Master's thesis in MTPROD
Supervisor: Pål-Tore Storli
June 2020

Norwegian University of Science and Technology
Faculty of Engineering
Department of Energy and Process Engineering

NTNU
Norwegian University of
Science and Technology

Abstract

Oscillatory fluid structure interactions due to vortex shedding is a well known source of vibrations experienced by components in hydro machinery. This phenomena is called vortex induced vibrations (VIVs), and are known to cause damage and early fatigue of components like turbine blades and guide vanes. Being able to mitigate these VIVs of such components is very interesting in an industrial perspective, given that this extends the life time of vital components in hydro machinery. In light of this, a research campaign investigating how different trailing edge designs can reduce VIVs on a hydrofoil is being carried out at the Waterpower laboratory at the Norwegian School of Science and Technology (NTNU). Work has been performed to prepare a test rig at the Waterpower laboratory for doing Particle Image Velocimetry (PIV) measurements of the wake flow behind a hydrofoil.

As a part of this research campaign, this master thesis is focusing on how the use of a splitter plate attached at the trailing edge of a hydrofoil can mitigate VIVs. Splitter plates have shown to be dampen vortex shedding when used on cylinders, and it is therefore interesting to investigate if they have the same effect on hydrofoils for use in hydro machinery.

The initial plan was to conclude a PIV measurement campaign of the hydrofoil with splitter plate. Yet, due to the outbreak of the global pandemic, Covid-19, things did not go exactly after the plan. The shut down of the university caused delays on the laboratory work, which resulted in taking the hydrofoil with splitter plate measurements off the program. PIV measurements of a hydrofoil with another trailing edge modification have been carried out, and the results from these measurements are presented in this thesis. This trailing edge modification got prioritized above the splitter plate due to higher relevance to the over all research campaign. The outcome off this thesis will be divided in two parts, with the first part building up to a measurement campaign. using splitter plate and the second part presenting results from another measurement campaign. In presentation of the results, the fluid mechanics behind will not be discussed, as the used trailing edge modification is not investigated earlier in the thesis. The results will be discussed more on how the experimental method of PIV works, and a conclusion will be drawn on how well PIV would work for the initial case with splitter plates.

Keywords: Vortex Induced Vibrations (VIVs), Vortex Shedding, Particle Image Velocimetry (PIV), Splitter Plate, Lock-In

Sammendrag

Oscillerende fluid-struktur interaksjoner grunnet virvelavløsninger er en godt kjent kilde til vibrasjoner i komponenter tilhørende hydromaskineri. Disse vibrasjonene kan føre til skader og tidlig utmattelse av komponenter som turbinblader og ledeskovler. Derfor er det veldig interessant fra et industrielt perspektiv å kunne dempe virvelavløsninger, og følgelig dempe vibrasjoner i disse komponentene, for å øke levetiden. Det har det blitt satt i gang en forskningskampanje på Vannkraftlaboratoriet ved Norges Teknisk-Naturvitenskapelige Universitet (NTNU), hvor effekten av hvordan forskjellig endedesign på en hydrofoil kan redusere virvelavløsning. Det har blitt gjort arbeid for å forberede en test rig til å utføre PIV- (Particle Image Velocimetry) målinger av vaken bak en hydrofoil.

Som en del av denne forskningskampanjen skal denne masteroppgaven fokusere på effekten av å bruke en splitterplate festet på enden av en hydrofoil, for reduksjon av virvelavløsninger. Splitterplater har vist god effekt for demping av virvelavløsninger på sylindere, og det er derfor interessant å undersøke om effekten er like god ved bruk på en hydrofoil.

I utgangspunktet var planen å utføre PIV målinger av hydrofoilen med splitterplate, men på bakgrunn av utbruddet av Covid-19 viste dette seg å bli vanskelig. Nedstengningen av universitetet førte til utsettelser i laboratoriearbeidet som var planlagt for våren 2020, hvilket resulterte i at PIV målingene for hydrofoil med splitterplate ble tatt av programmet. Dette medfører at det ikke vil bli presentert noen resultater for det foreslåtte designet. Som en erstatning vil det bli presentert noen resultater for PIV målinger utført på en hydrofoil med et annet endedesign. Dette endedesignet ble prioritert over splitter plate, da det ble vurdert mer relevant til hele forskningskampanjen sett under ett. Fluidmekanikken bak disse resultatene vil ikke bli diskutert i særlig grad, da dette designet ikke blir omtalt gjennom oppgaven. Resultatene vil bli diskutert mer på bakgrunn av fremgangsmåten for å komme frem til dem, hvor også usikkerheter rundt målingene blir diskutert. Konklusjonen trekkes ut fra hvor vidt måleteknikken, PIV, ville egnet seg for målinger av det planlagte designet med splitterplate.

Nøkkelord: Virvelindusert vibrasjon, Virvelavløsning, Particle Image Velocimetry (PIV), Splitter Plate, Lock-In

Acknowledgement

The work presented in this master thesis is performed at the Waterpower laboratory, Department of Energy and Process Engineering at the Norwegian school of Science and Technology.

During the process of working with this master thesis, there have been several people that have helped me getting a finished result. First of all I would like to thank my supervisor, Pål-Tore Storli, for great guidance through the process. Pål-Tore have been helping me a lot with the structure of the final report, and general logistics around the project. Special thanks to my co-supervisor Kristian Sagmo for letting me be a part of his project and teaching me about PIV in theory and practice. Working with state-of-the-art equipment in this project have been very educating and fun. I would also like to share gratitude to the guys working at the work-shop at the Waterpower laboratory, for helping out with different tasks. Finally I would like to thank my student colleague Hasan Besirovic. We have been working on similar projects, and helping each other to a finished thesis. Working with Hasan have been enjoyable, and his help is greatly appreciated.

Carsten Cleve-Hansen

Carsten Cleve-Hansen

Trondheim, 06 2020

Table of Contents

List of Tables

List of Figures

Abbreviations

Symbol	=	**Definition**
CCD	=	Charge Coupled Devices
CMOS	=	Complementary Metal-Oxide Semiconductor
CFD	=	Computational Fluid Dynamics
FFT	=	Fast Fourier Transform
FOV	=	Field of View
FSI	=	Fluid-Structure Interaction
IA	=	Interrogation Area
PIV	=	Particle Image Velocimetry
TE	=	Trailing Edge
VIVs	=	Vortex Induced Vibrations

Chapter 1

Introduction

In a world where the energy demand is constantly increasing while the focus on climate changes due to human pollutants and emissions is higher than ever, the importance of renewable energy sources is immense. Being able to saturate the energy demand, while at the same time reducing the emissions, is what drives all the research on renewable energy sources these days. Hydropower is a renewable energy source and the main source of electricity in Norway. At the Waterpower laboratory at the Norwegian School of Science and Technology it have been done research on the topic for the past 100 years. While the efficiency of the hydropower turbines is very high, a big part of the research on the topic is now on how to extend the lifetime of the different components in a hydropower plant. This is where the topic of this thesis comes in. In hydro machinery, there are guide vanes, stay vanes and runner blades with the shape of a hydrofoil. To avoid unnecessary fatigue or failure of these components, it is crucial to keep the shedding frequency away from the natural frequency of the blades during range of operation.

In the early 1960s, the relationship between trailing edge geometry and vortex shedding was observed (Blake, 1984). Since then, trailing edge modifications have been used as a solution to flow-induced vibration and fatigue of turbine blades. Using splitter plates for this purpose is the topic of this thesis, and the idea is that the splitter plates will mitigate the vortex shedding and hence the vortex induced vibrations. To investigate the effect of splitter plates on a hydrofoil, a measurement campaign using PIV will be carried out. By using PIV to calculate the instantaneous velocity field of the wake behind the hydrofoil, combined with sensors to log the vibrations that are experienced by the hydrofoil, the effect of the splitter plates regarding vortex induced vibrations will be well established. The PIV experimental test setup that is used in this project is delivered by LaVision. To post process the raw data from the measurements, a PIV software delivered by LaVision, called DaVis, is used. The other sensors on the test rig is connected through a logging program by National Instruments, called LabView.

1.1 Objective

As mentioned, the original objective of this study was to carry out a PIV measurement campaign of a hydrofoil fitted with a splitter plate. The goal was to investigate how the splitter plate would work for mitigation of vortex shedding and flow induced vibrations. However, due to the outbreak of Covid-19 and the shut down of the university, things did not go according to the plan with the laboratory work. Changes in the activity plan at the Waterpower laboratory resulted in taking the measurement campaign for hydrofoil with splitter plates off the program. Accordingly, there will not be presented any results for this case in the thesis. Nevertheless, PIV measurements were preformed on a hydrofoil with another trailing edge design. This design got prioritized above the splitter plates, due to higher relevance to the research campaign on mitigation of VIVs on a hydrofoil. The acquaintance from these measurements, and the analysis of the results will be discussed in this thesis. Finally there will be concluding remarks on how well the experimental method would work for a similar measurement campaign with splitter plates. The physics and fluid dynamics behind the results that are acquired will not be discussed, as this thesis is not investigating the effect of the trailing edge modifications used in the experiments. More over, the attainments and discussions presented here will be helpful information for a possible future study to carry out the measurement campaign of a hydrofoil with splitter plate, that initially was expected from this thesis.

1.2 Outline

The thesis ended up being rather divided. Chapter two through five will be building up to the measurement campaign with splitter plates, and by chapter six the results from a measurement campaign with another trailing edge modification will be discussed.

- Chapter two contains basic theory on the phenomenons that are investigated in this study.
- Chapter three is a literature review on using splitter plates as a passive flow controller.
- The fourth chapter is on basic theory in the PIV measurement technique.
- Chapter five is a description of the measurement campaign, which contains a description of the hydrofoil design, the test rig and how the experiments are executed.
- Chapter six is presenting results from the measurement campaign on the other trailing edge design. The discussion in this chapter will not be on the physics behind the results, but more on how the method works and a presentation of uncertainty of the obtained results.
- Chapter seven is a conclusion, where it will be concluded on weather the measurement campaign will work for the planned design with splitter plates.
- Finally, chapter eight will present further work to proceed with on the topic.

Chapter 2

Basic Theory

2.1 Fluid Structure Interaction

Fluid structure interactions (FSI) is a combination of fluid dynamics and structural mechanics that appears when a fluid flow meets a structure. These interactions will either be stable or oscillatory, and forces are converted from the fluid to the structure by shear stresses and pressure differences as the fluid flows over or through the structure. Stable FSI are when the fluid is interacting on a structure with constant force, without the structure deforming considerably, for example hydro static pressure from water on a dam. Oscillatory FSI happens when fluid interacts with a structure leading to induced strain which causes the structure to move. The strain will then be reduced, and the structure returns to original state only for the process to continue. These effects are important to consider in many different engineering applications, e.g. bridges, aircrafts, subsea constructions and pipelines, and turbo machinery, to name a few. Failing to recognize oscillatory FSI in a design can give catastrophic outcomes. A good example of this is the collapse of Tacoma bridge, shown in figure 2.1. High winds resulting in vortex shedding frequency overlapping with the natural frequency of the bridge caused this collapse.

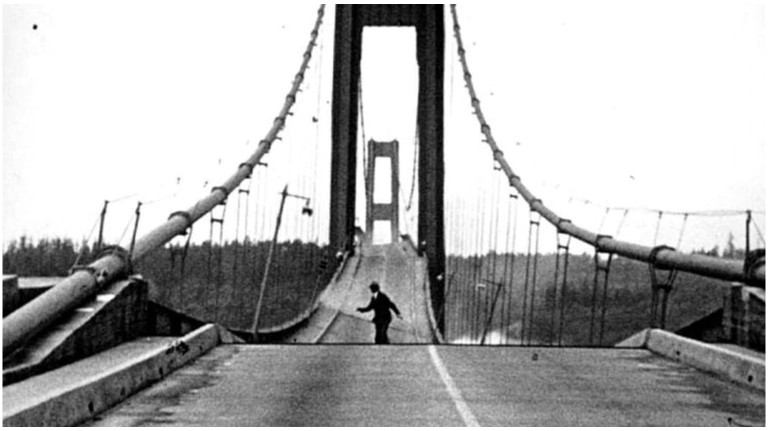


Figure 2.1: Tacoma bridge collapse, November 7, 1940. (https://kiro7.com/)

2.2 Vortex shedding and Vortex Induced Vibrations

In the case of this thesis, the fluid structure interactions that are dealt with comes from fluid flowing over a structure, i.e. water flowing over a hydrofoil. The main source of interactions on the hydrofoil in this case comes from a flow phenomena called vortex shedding. Vortex shedding is oscillating vortices that are developing downstream of a bluff body, when a fluid is flowing over it. The vertices are caused by pressure differences on the top and bottom section of the bluff body, due to flow separation. The vortex shedding creates a pressure field, which initiates force in horizontal direction pulling the hydrofoil with the stream, i.e. drag forces. Many studies that are focusing on mitigation of vortex shedding does this for the purpose of drag reduction. The fluctuating pressure differences on top and bottom of the hydrofoil will induce oscillating lift forces, force in vertical direction, causing the hydrofoil to vibrate. This is called vortex induced vibrations or VIVs for short.

The frequency of the vibrations in the hydrofoil will follow the vortex shedding frequency, which can be found with the Strouhal number ($f_s = St\frac{U_\infty}{L}$), where f_s is the shedding frequency, L is the characteristic length of the body, U_∞ is the free stream velocity, and St is the Strouhal number. This non-dimensional relationship can help us predict a theoretical shedding frequency, by setting the other values based on former experiments and the geometry of the bluff body. For a smooth cylinder, the Strouhal number is about 0.20 (Sarpkaya, 1979). The Strouhal number was found by the Czech physicist Vincenc Strouhal, in 1878.

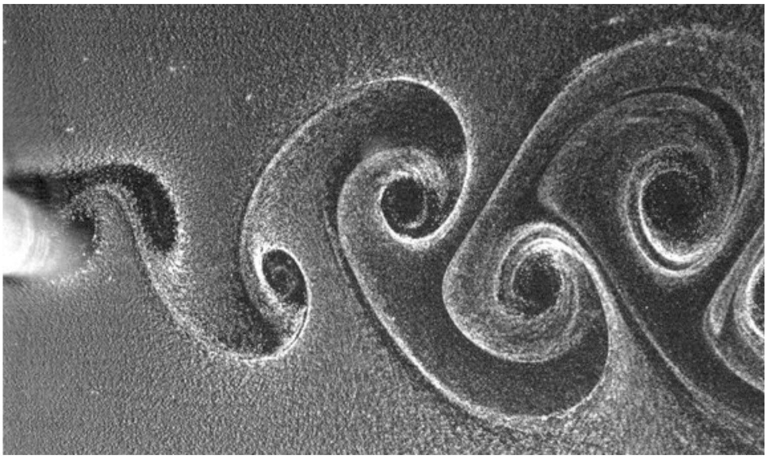


Figure 2.2: Vortex shedding behind a cylinder (https://steemit.com/)

2.2.1 Lock-in

The main goal of this study is to mitigate the vortex shedding and hence the VIVs in the hydrofoil, to avoid a state called lock-in. Lock-in is reached when the shedding frequency is overlapping with the natural frequency of the hydrofoil, which causes the vortex shedding frequency to latch on to the natural frequency. This will cause intense vibrations in the hydrofoil, which again can cause unnecessary fatigue and failure of components. A study by Sagmo et al. (2019) investigated where the lock-in state is reached for a hydrofoil using PIV- and CFD measurements. The study stated that when the shedding frequency reached natural frequency at $U_\infty \approx 11$ m/s, the frequencies were latched on to each other for an interval of $\Delta U_\infty \approx 1$ m/s. In this region, lock-in was reached. This is shown in Figure 2.3 (Sagmo et al., 2019).

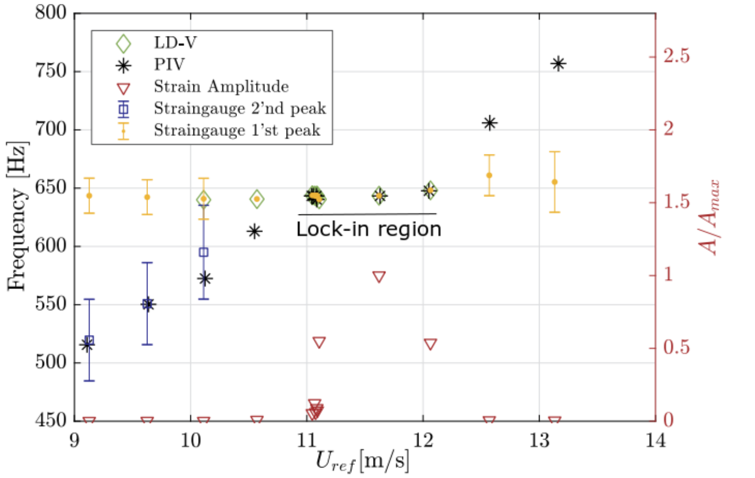


Figure 2.3: Hydrofoil vibration frequencies and shedding frequencies measured by PIV, strain-gauges and LD-Vibrometer. The relative hydrofoil vibrational amplitude is plotted along the right y-axis. (Sagmo et al., 2019)

2.3 Splitter Plates

Splitter plates are a passive device used for flow control. The plate is either attached to the trailing edge of the body, or placed some length down stream. The function of the plate is to reduce the strength of the vortex shedding, which again can give drag reduction or a reduction of vibrations experienced by the body. Studies have shown that if the splitter plate is long enough, about 5 times trailing edge thickness (Kwon and Choi, 1996), the vortex shedding will be fully suppressed. This is also depending on the Reynolds number, Re[1], of the flow. Figure 2.4 shows an example of a splitter plate attached to a cylinder.

[1] The Reynolds number is a non-dimensional quantity in fluid mechanics of high relevance. The value of the Reynolds number gives the ratio between the inertial forces to viscous forces within the fluid, subjected to the velocity of the fluid. The value helps determine if the fluid flow is laminar or turbulent, generally low Reynolds number gives laminar flow and high Reynolds number gives turbulence.

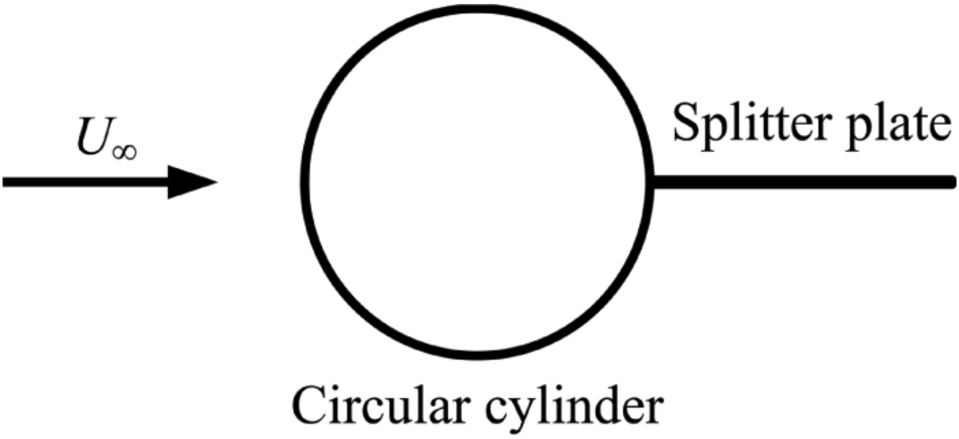


Figure 2.4: Splitter plate attached to a cylinder.

Chapter 3

Literature Review

The effect of using splitter plates are mostly studied on flow over cylinders, however they have shown to dampen the vortex shedding and hence reducing the drag forces acting on the cylinder. The main goal of this study is to see how the vortex shedding and vortex induced vibrations can be reduced by using splitter plates, and some earlier studies of this topic are presented in the following section.

3.1 Passive Flow Control with Splitter Plates

In a study by Kwon and Choi (1996), the use of splitter plates in a flow over a cylinder was investigated by simulations. The study is looking in to laminar vortex shedding, and thus the Reynolds numbers used are fairly low ($50 < Re = u_\infty d/\nu < 200$) where d is the cylinder diameter, ν is the kinematic viscosity and u_∞ is the free stream velocity. Different splitter plate lengths are also investigated in the range of Reynolds numbers. The lengths are varying between $1 < l/d < 5$, where l/d is the ratio of plate length and cylinder diameter. The critical plate length, l_c/d, is also found for the different Reynolds numbers. This is the plate length where the down stream vortex shedding is fully suppressed by the splitter plate. The critical plate length was shown to be $l_c = 3d$ at $Re = 100$ and $l_c = 5d$ at $Re = 160$. Given this, it seems that as the Reynolds number increase the plate length must also increase to achieve full suppression of the vortex shedding. The drag reduction detected were also decreasing as the plate length increased, giving a minimum skin friction drag at $l/d \approx 4.5$.

More recently Dai et al. (2018) carried out a CFD study on a cylinder fitted with splitter plate at high Reynolds number of $Re = 1.8 * 10^5$. The splitter plate lengths used were between $0 < l/D < 1.25$, where l and D are plate length and cylinder diameter respectively. Experimental results was used to validate the results. By looking at the forces acting on the cylinder, one can determine weather the vortex shedding is decreasing or not. Since the vortex shedding will produce fluctuating lift forces on the cylinder, a reduction of this is a result of damped vortex shedding. The study found that the ideal plate length was $l/D = 0.66$, which yielded about 30% reduction of the fluctuating lift force.

A study that might be even more transferable to this thesis are the one by Bearman (1965), which looked in to the use of splitter plates fitted on a two-dimensional model with a blunt trailing edge. Since the hydrofoil that is investigated in this thesis have a blunt trailing edge, this paper is highly relevant. Looking into plate lengths varying from $0 < l/h < 4$ (1 = plate length and h = trailing edge thickness) at a Reynolds number between $1.4 * 10^5$ and $2.56 * 10^5$. The experiments were preformed in a wind tunnel, and to detect the results the base pressure coefficient were measured at the surface. An oil film on the surface was used to find any re-attachment points, and a hot wire anemometer was used to find the velocity fluctuations. Results showed that the flow re-attached at $l/h \approx 2.9$ on the splitter plate of length $4.0h$. This means that the vortex shedding is fully suppressed on trailing edge scale at this plate length. The study also showed that the fully formed vortices moved downstream as the plate length increased.

The effect of using splitter plates for drag reduction on a blunt trailing-edge airfoil was investigated both numerically and experimentally in a study by Baker and van Dam (2008). Different edge treatments on the splitter plates were tested, at a chord based Reynolds number of 666,000. For the case with splitter plate, the experimental and computational results matched very well in both lift and drag forces. For the baseline airfoil the drag forces did not agree very well when comparing experimental and computational. According to Baker and van Dam (2008), this is likely because of over-predicted strength from the vortices due to artificial restrictions of the flow in two-dimensions for the computational case. Looking away from the computational results of the baseline airfoil, both the experimental and computational results with slitter plate show a 50% reduction in drag, compared to the experimental results of the baseline airfoil at zero angle of attack. The drag reduction is likely to come from absence of the vortex shedding once the splitter plate is added to the airfoil.

From the papers presented above, it is noted that most of the studies on splitter plates are focusing on the effect of drag reduction. Drag reduction can be accomplished from mitigation of vortex shedding, which is the function of a splitter plate. In this thesis the purpose of the splitter plate is to reduce the VIVs, by mitigation of vortex shedding. Therefore, the drag reduction as a result of the splitter plate is not studied in particular later in the thesis.

Chapter 4

Basic Theory: Particle Image Velocimetry

The method that is used to collect results in this thesis is called Particle Image Velocimetry (PIV). This is an effective and accurate way of calculating the flow field when doing experiments in a laboratory. In short terms, the method works by sending tracer particles through the flow, and then taking pictures of the particles to follow their movement. By doing this, the flow field can be calculated from the movement of the tracer particles. An explanation of important factors in PIV will be described in the following sections. Figure 4.1 shows an example of a PIV setup.

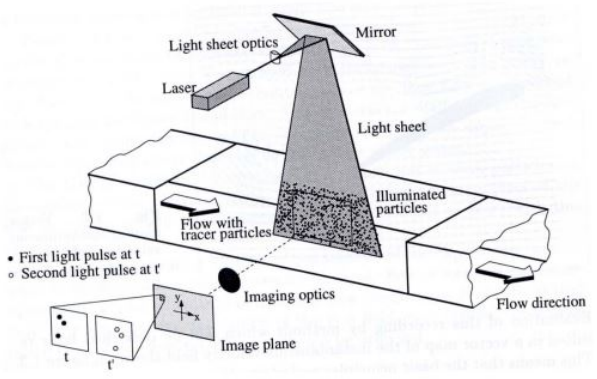


Figure 4.1: Example of experimental arrangement for planar 2D PIV in a wind tunnel (Raffel et al., 2018)

4.1 Tracer Particles

The tracer particles used in PIV must follow the flow perfectly, without interrupting the flow pattern or changing it in any way. The particles should have similar density as the fluid it is flowing through, to make them naturally buoyant. Another important tracer particle feature is efficient scattering of light, to easily be seen and recorded by the camera.

4.2 Camera and Illumination Source

To be able to catch all the particles in a picture, both a camera and an illumination source is needed. The illumination source are usually provided by a laser, but other methods, like LED lights, can also be used. The reason why laser is convenient for use in PIV, is because many laser have the advantage of a pulsed output with a duration and repetition rate that can be coordinated with the closing time of the camera lens (Raffel et al., 2018). The illumination beam must be shaped in to a planar sheet. The illuminated sheet must be exactly perpendicular to the camera direction to avoid bias errors in the recordings.

4.3 Field of View and Interrogation Area

The field of view (FOV) is the area that are investigated in the PIV recordings, i.e. the area of the flow that is recorded by the camera. The field of view must then be divided in smaller areas to be able to analyse the pictures, these areas are called interrogation areas (IAs) or interrogation windows and are show in figure 4.2. The area must be big enough to fit several particles inside, so no information is lost. If the IA is too small, particles can travel through without being recorded. A rule of thumb is that the IA shall be big enough to show 10-25 illuminated tracer particles (https://www.dantecdynamics.com/). On the other hand, you want the IAs as small as possible to achieve as god spatial resolution as possible. The size of the IAs is chosen based on tracer particle size and feeding density, and is decided after doing a post processing test of the recorded PIV data.

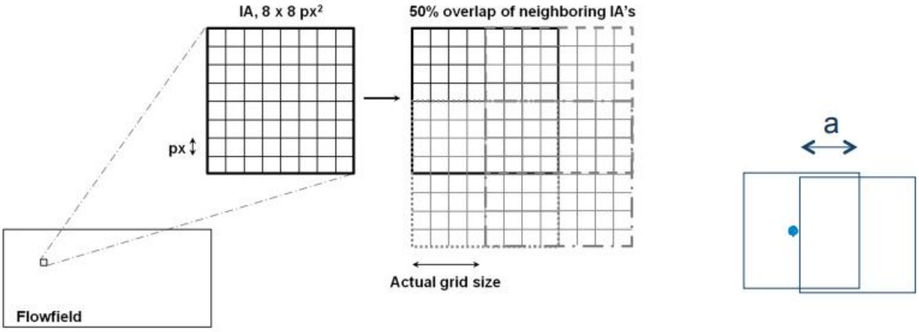


Figure 4.2: Left: Example of how the flow field is divided in interrogation area (Vergine and Maddalena, 2014). Right: Overlapping of two IAs.

To avoid that information from the tracer particles at the edge of the IA is lost, overlapping of the areas is used. This is shown in figure 4.2. Using an overlap of about 50% is common, but this will depend on the set up. The best overlap is found by doing tests and analysing the results. If interrogation area overlapping exceeds 50%, the velocity data can be increasingly correlated due to over sampling of the tracer particles (Raffel et al., 2018).

4.4 Digital Camera Sensors

These days, most of the imaging is done with electronic recording techniques and not photographic. The big advantage with electronic imaging is the immediate availability and feedback during recording. When it comes to digital camera sensors, there are mainly two types that are used. These being CCD (charge coupled devices) and CMOS (complementary metal-oxide semiconductor), where CMOS is the stat-of-the-art technology.

A CCD-sensor works by converting light (photons) in to an electric charge (electrons). A CCD sensor can be referred to as an array of many CCD-elements, where each element produces one pixel. Putting all the pixels together will result in an image.

A CMOS-sensor is built up by pixels, like the CCD-sensor. The difference is that each pixel can be controlled separately by a transistor. This gives the sensor an advantage as it is able to focus on one special region of interest (RoI), and will therefore be able to produce images at a higher rate. This makes the CMOS-sensor more convenient to use, especially when it comes to high speed PIV (Raffel et al., 2018).

4.4.1 Recording Techniques

There are different techniques that are used in PIV recording. These techniques can generally be divided in two groups; methods that capture two or more illumination pulses to a single frame (Single-frame/multi-exposure) and methods that capture one image per illumination pulse (multi-frame/single-exposure).

4.5 Image Analysing Methods

In PIV there are several different methods for analysing the imagines. Young's fringe analysis method is a way of following ”Young's fringes” to see their direction, and hence obtain velocity. The fringes can be found by illuminating image pairs so that they act as interfering point sources, and the transmitted light is then forming Young's fringes. Correlation methods are another way of analysing PIV imagines. Different correlation methods are *auto-correlation* and *cross-correlation*.

An auto-correlation equation is obtained from Fourier transform of the fringe pattern, and produces three peaks that gives the displacement of particles between imagines. There is one peak at the origin, which is zero-order (self-correlation), and two first-order displacement peaks of plus and minus the displacements between particles (Grant, 1997). The velocity can then be measured from the distance between the center of the first- and second-order peaks. This gives a volume average of the velocity in the examined cell or interrogation area.

Cross-correlation can be used in analysing single/double-famed, double- or multiple-exposure PIV imagines. The scattered light from the particles from first and second exposure is recorded in separate images, and then these pictures are divided in to interrogation areas. By tracking the movement of the particles inside the IA between the two exposures, a local velocity vector for this IA is calculated. An illustration of the cross-correlation can be seen in figure 4.3

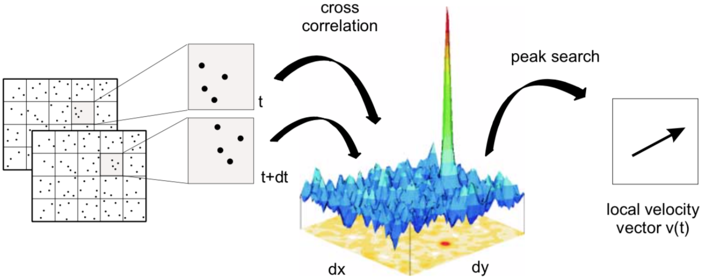


Figure 4.3: Evaluation of PIV recordings using cross-correlation (LaVision, 2017)

4.6 Peak Locking

Peak locking, or pixel locking, is a well known source of error in PIV. It occurs when the tracer particle diameter is less than the size of one pixel. Since one are not able two track movement inside one pixel, only between pixels, having particles of too small diameters will give errors. The single traceable displacement will then be from pixel to pixel, and as a result all particle displacements will fall under an integer amount of pixels. Therefore, it is recommended that the tracer particles have the size of at least two pixels according to Raffel et al. (2018).

Chapter 5

Measurement Campaign

Based on the project work last semester, where a measurement campaign was planned, the campaign have been decided. In the following sections, different aspects to the measurement campaign will be described in detail.

5.1 Experimental Method

The experimental method of choice in this thesis is PIV. This method is a reliable and effective way to procure what is needed for the study. An other method that could have been used is computational fluid dynamics (CFD). If CFD were to be used in this study, it would require a lot of computational power to complete in reasonable time. The test section would have to be simulated in three dimensions, and with a mesh fine enough to solve the small scale motions in the flow. In the study of the similar case done by Sagmo et al. (2019), a CFD study was also used as a tool. The mesh in that study contained 13 million hexahedral elements. The simulations have to be done transient as well, with a time step small enough to at least capture two samples for each period of the vortex shedding. This is to be able to read out the shedding frequency. Taking these factors in to consideration it is evident that a CFD study of this case could be as time consuming as doing a PIV. And, on top of this, a CFD study does not stand good as a result by it self, and should be validated with experiments. Given this the most efficient method to use is probably PIV, which is why this is the method that is applied.

The Waterpower laboratory at NTNU hosts a suitable test rig for the experiment, which contains all the necessary equipment to preform a PIV measurement campaign of the hydrofoil with splitter plate.

5.2 Hydrofoil Design

The hydrofoil design is similar to the one used by Sagmo et al. (2019), but with trailing edge modifications. The Geometry that is used in this thesis can be found in Figure 5.1, which shows a blunt trailing edge hydrofoil fitted with a splitter plate. Figure 5.1 shows a splitter plate length $l_s = t_{TE}$ = 4.2mm, where t_{TE} describes the trailing edge thickness. The trailing edge is designed to be able to change the splitter plate, without taking off the entire trailing edge part. This makes it faster to change between different plate lengths. Initially, the plate lengths that will be tested are: $l_s = t_{TE}$ = 4.2mm and $l_s = 5 * t_{TE}$ = 21mm. The splitter plate design is based on previous studies on the topic, and is more carefully discussed in the project thesis, found in attachments.

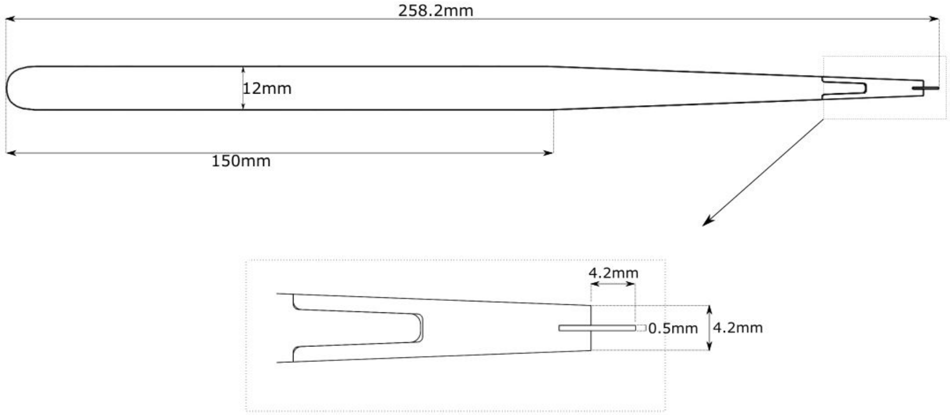


Figure 5.1: Hydrofoil geometry and trailing edge design

After the trailing edge tip and the splitter plate is attached to the foil body, the leftover glue needs to be brushed off to make the surface completely smooth. To ensure that the surface is smooth and does not reflect light, which can disturb the PIV recordings, it is coated in matte black paint.

The trailing edge tip is manufactured without the slit as a standard part that can be machined in to different trailing edge modifications. This manufacturing is done by a company independent of the university. The slit must be machined by the workshop employees at the Waterpower laboratory. The get good effect of the splitter plate, it is beneficial that it is remarkably thinner compared to the trailing edge thickness. With a trailing edge thickness of t_{TE} = 4.2mm, it follows that the splitter plate must be very thin. This again means that the slit in the trailing edge tip must be as thin, which is challenging to manufacture. The slit is made with a 0.5mm milling tool. Milling in aluminium can be difficult, since the material is ductile compared to e.g. steel. When milling in aluminum, the material can easily stick to the tool, causing the tool to break. This is an even higher risk when using very thin tool of 0.5 mm, in Figure 5.2 a picture of the milling process is shown. The process of milling out the slit must be done with ease, and each cut must be small.

The slit was made in the trailing edge tip with success. The milling was concluded by using high rotational speed of the milling tool, about 1500 rev, and a lot of cutting fluid to reduce friction and effectively remove shavings. With small cuts and low feeding speed, the whole process took about 6 hours.

The splitter plate that is attached to the trailing edge tip must be cut out of an aluminium plate of thickness 0.5 mm. It is beneficial that the alloy is the same for the plate and the hydrofoil. By using the same aluminum alloy, the natural frequency of the hydrofoil will not be affected as much. This is advantageous when comparing with a reference hydrofoil.

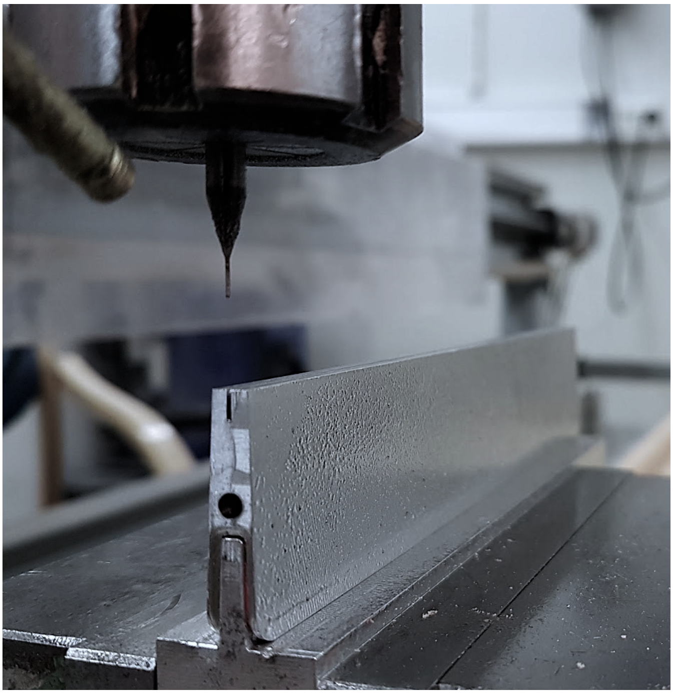


Figure 5.2: Trailing edge tip: Milling slit

5.3 Test Rig

The Waterpower laboratory at NTNU hosts a suitable test rig for PIV measurements of a hydrofoil. In figure 5.3, a side view of the test section can be seen. In addition to the PIV instruments, the test rig also contains various other sensors. These are sensors to log pressure, volume flow, strain and temperature. The PIV set up and the other sensors that are used in the measurement campaign will be more carefully described in the following sub sections.

5.3.1 PIV Recording Equipment

The PIV recording system that is used in the experiment is delivered by LaVision. The system is using a Nd:YFL dual cavity laser to illuminate the flow following particles. The camera is operating with a CMOS camera sensor, which is described in section 4.4. A PIV-pulse sensor is establishing the communication between the camera and the laser. This is to make sure that the pictures are taken at the same time as the laser pulse is fired. The PIV-pulse sensor are also giving signals to LabVIEW, which makes it possible to extract data from the other sensors at the same time as the PIV recordings were taken. This can be helpful when analysing the data.

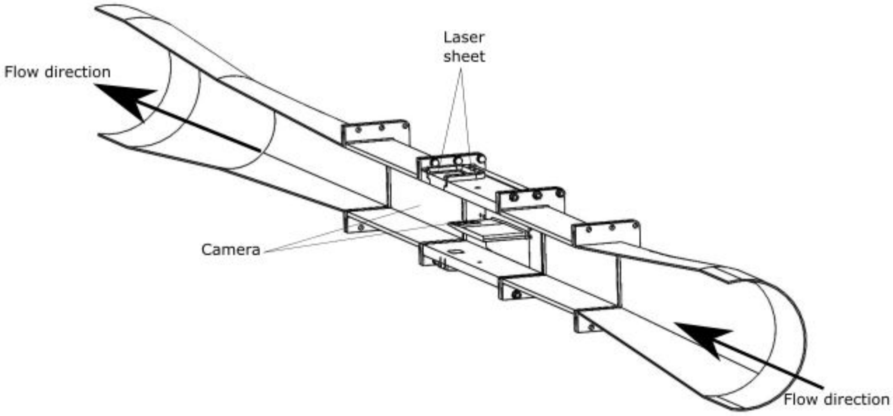


Figure 5.3: Experimental test section

PIV Calibration

The PIV camera and laser must be adjusted and calibrated to get results that can be used. This is a process that should be executed with high accuracy, to make sure measurements are reliable. One of the most important things to make sure is that neither the camera nor the laser is moving during recording. A small movement will change the focus or the coordinate system, and the outcome will be useless recordings. By placing the camera and laser on solid stands, it is easier to keep the position of these during experiments. With this done, the calibration of the laser and camera can proceed.

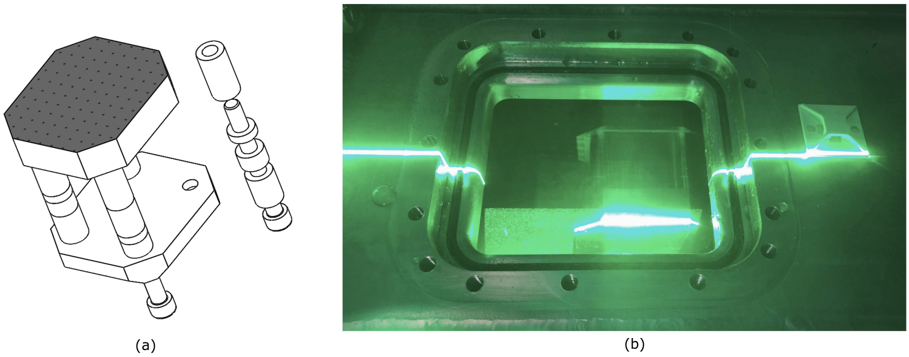


Figure 5.4: a) PIV calibration device with calibration grid on front plate b) Laser alignment with calibration device

The laser is creating two beams that are reflected in two different sheets. These laser sheets must overlap, to illuminate the same set of particles. This is because the recording technique used is double frame-double exposure, which means that each picture is a combination of two frames. The two frames is captured on each exposure of the two laser sheets, and the velocity field is calculated from cross-correlation between the two frames. By that reason it is important that the laser sheets are overlapping to illuminate the same set of particles, or else the cross-correlation will contain a lot of noise and spurious vectors will be unavoidable. The laser sheet overlap is inspected visually by placing a sheet of paper underneath the laser, and change between the two laser sheets. The overlap will also be calculated by DaVis when taking pictures of the flow subsequently to the particle feeding (section 5.4.1). A correlation value between 0 and 1 for the two exposures are given, with 0 being anti-correlation, 0.5 being no correlation and 1 being complete correlation. A correlation value above 0.9 is preferred to avoid noise. When the overlap is controlled, the laser sheets must be aligned to the plane that is recorded. This is done with using a calibration device, shown in Figure 5.4 (a). This device is machined with high accuracy, so it reliably aligning with the desired field. When the calibration device is in place, the laser sheet is adjusted to align with the device. An example of good laser sheet alignment can be found in Figure 5.4 (b).

After the laser sheet is in sufficient position, the camera must be placed so that it is recording orthogonal to the laser sheet. Positioning of the camera is also done with help from the calibration grid. The ideal positioning of this case will give a field of view (FOV) where the wake from the hydrofoil is more or less centered. To get correct length unit in the PIV processing program, DaVis, values based on the calibration grid must be set. DaVis will recognize the crosses on the calibration grid, and by entering the distance between them, DaVis will know the length scale of the FOV. Also, with the positioning of the calibration grid, the distance from trailing edge and bottom of the test section is known. This is also entered in DaVis, so the position down stream can easily be extracted when post processing data.

5.3.2 Pressure Sensors

The test section is equipped with four different sets of pressure sensors. These are sensors to measure absolute pressure in the test rig, pressure distribution over the hydrofoil, differential pressure between inlet and outlet of test section, and dynamic pressure placed over and below the hydrofoil. All of the sensors are connected to a DAQ station, where the signals is read out through LabVIEW[1].

The sensor that measures the absolute pressure is placed upstream of the test section. This sensor is delivered by DRUCK, and is called PTX UNIK 5000 pressure sensor. The measuring range is from 0 to 10 bar, with output signal from 7 to 32 Ampere. The output signal is then converted to voltage by sending it through a resistance. This is because LabVIEW reads signal in voltage.

The pressure distribution over the hydrofoil is measured with 16 different pressure sensors, all of type DRUCK PMP UNIK 5000. The output signal from these sensors are ranging from 0 to 10 Volt, with a measurement range from 0 to 8 bar. The hydrofoil is machined with small holes on the surface, which are the pressure outlets. These holes are connected to the sensors by tubes. This setup is shown in figure 5.5.

[1] LabVIEW is a sensor logging program delivered by National Instruments. LabVIEW connects all the sensors on the test rig, and creates a file where all the data from the measurements is stored.

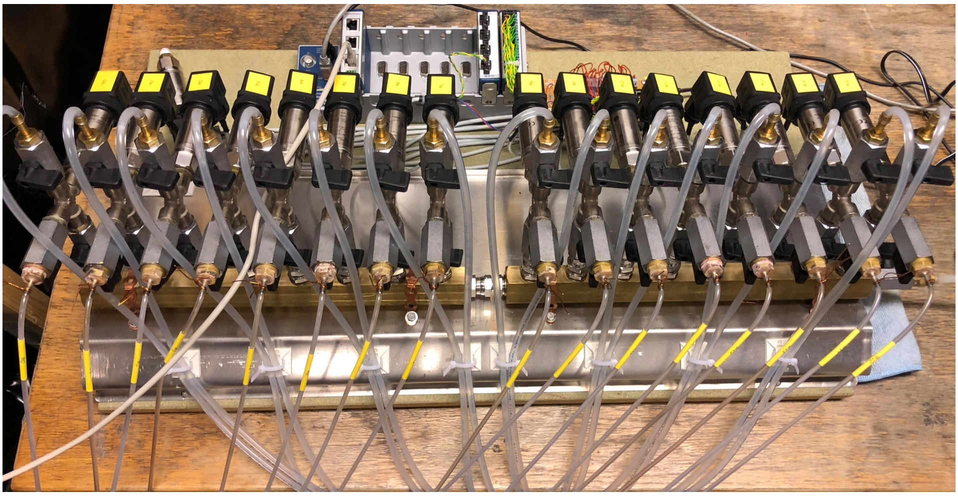


Figure 5.5: Differential pressure sensor rack

To measure the differential pressure between upstream and downstream of the test section, a pressure sensor from Fuji Electric is used. The sensor can measure a pressure difference ranging from 0 to 320 mbar, with an uncertainty of about 1 mbar. The output signal from this sensor are 4-20 mA. The differential pressure sensor is used to investigate how different trailing edge designs are comparing in energy loss.

The dynamic pressure sensor is there to give redundancy on frequency measurements from strain gauge. The sensor is measuring pressure fluctuations in the flow past the hydrofoil, which is associated with the vibrations of the hydrofoil.

Pressure Calibration

Each pressure sensor is calibrated with using a hydraulic deadweight tester. This pressure calibrator gives an accuracy of $\pm 0.008\%$.

The calibration process works by setting pressures starting at 1 bar, with 1 bar increments, up to the full range of the sensor and down again to 1 bar. At each increment, the signal from the sensor is written down and entered in LabVIEW. Finally, a calibration constant is achieved, which makes LabVIEW give the right output value from the sensor input. A calibration sheet with uncertainty for all the pressure sensors can be found in appendix A.1 and A.2.

This process is repeated for all the pressure sensors on the test rig before starting the experiments, and preferably after to investigate if the calibration constants have drifted during the experiments.

5.3.3 Flowmeter

To calculate the velocity of the flow in the test section, an electromagnetic flowmeter is used. The flowmeter is delivered by ABB, and the model is FSM 4000. This is a robust flowmeter designed for heavy duty applications, according to their own website (https://new.abb.com/). The flowmeter is placed some length down stream of the test section, and is connected to the DAQ station like the other sensors.

Flowmeter Calibration

The flowmeter is calibrated by using a weighing tank system. By running water through the rig past the flowmeter for a given time in to a weighing tank, the flow can be calculated from the wright change in the weighing tank. The flowmeter should be calibrated through the full velocity range that is used while doing the experiments. Starting at a low flow rate and increasing with increments of $\approx 0.05m^3/s$ up to the highest velocities that is reached during experiments, and then the flow rate is decreased back to the starting point with same increment size. At each increment, the water is sent in to the weighing tank for a given amount of time, between 120 s and 30 s (depending on the flow rate, the change of weight in the tank should be at least 5 tonnes). The calculated flow rate for each increment is then used to find a calibration constant, which is entered in LabVIEW. A calibration chart can be found in Appendix A.3.

An important note that is important to watch out for during flowmeter calibration is cavitation. When the velocity of the water increases, there can be cavitation around the corners of the test rig, which will disturb the signal from the flow meter. To avoid this, the pressure in the test rig is set to 6 bar.

5.3.4 Strain Gauge

The strain gauge is produced by Kulite, and the signal sent out from it is given in voltage. The strain gauge is used to measure natural frequency of the hydrofoil, and the shedding frequency prior to lock-in state. From the strain gauge, it is seen at which velocity the hydrofoil reaches lock-in, or if it reaches it at all.

5.3.5 Temperature Sensor

Down stream of the test section, there is a temperature sensor to measure the water temperature. To have an exact temperature comes handy when the Reynolds number is calculated, since the viscosity changes with the fluid temperature.

Temperature Calibration

The calibration of the temperature sensor is an easier process. To obtain the calibration constant, the only thing needed is two known temperatures. These two temperatures are found in ice water at 0 degrees Celsius, and boiling water at 100 degrees Celsius. There will be some margin of error in this method of calibration, but it is acceptable in this case.

5.4 Execution of Experiment

When all the sensors are calibrated and the PIV set up is ready, then the measurements can be executed. In this section the process of doing PIV measurements will be explained, and notes that are important to take in to account will be conferred. Table 5.1 shows some of the values for the PIV setup that is used, these will also be discussed in more detail in the following section.

Table 5.1: PIV recording parameters for the hydrofoil wake flow measurements

Field of view (FOV) / Area of interest	21.1 mm x 16.9 mm / 1280 px x 1024 px (x-y)
Interrogation volume / Interrogation area	1.06 mm x 1.06 mm x 0.5 mm / 64 px x 64 px (x-y)
Experimental velocity range	(4 - 16) m/s
Observation distance & Lens F-number	215 mm & 5.4
Recording method & Camera sensor	Double frame/Double exposure & CMOS
Exposure time & image acquisition rate	250 μs & 4.882 kHz
Image processing mode	cross-correlation
Mean tracer particle diameter d_p	13 μm
Tracer particle density δ_p	$1.1 g/cm^3$
Illumination source	Nd:YFL dual cavity laser, 527 nm wavelength

5.4.1 Tracer Particles

The tracer particles used is of diameter d_p = 13 μm and density δ_p = $1.1 g/cm^3$, which satisfy the requirements explained in section 4.1. Before the measurements can start, the particle saturation must be satisfactory. From the LaVision manual (LaVision, 2017), it is said that a particle saturation of ≈ 0.05 particles per pixel is desired. To fill the test rig with tracer particles, a tank with particles is connected to it. While the water is running through the test rig, the particle feeding tank is opened for a couple of seconds at a time. The water must take a couple of revolutions through the whole rig for the particles to mix homogeneously. This process is repeated until the saturation is acceptable.

5.4.2 Area of Interest and Sampling

The full resolution of the camera gives 1280 px x 1024 px, which gives a sampling rate, or image acquisition rate, at 2441 Hz. The amount of images per measuring is limited by the camera RAM, which is 16 GB. When it is wanted to solve the smallest turbulent eddies, Kolmogorov scale[2], the sampling rate must be higher to have satisfactory time resolution. To achieve this, the resolution of the camera must be reduced. Therefore are some of the measurements done with reduced resolution, 1280 px x 702 px, which gives a sampling rate at 3551 Hz. This is only needed if one wants to calculate the turbulence intensity correctly. Also, to make sure that the total time of the measurements are sufficient, it can be useful to do some long samples at lower sampling rate (200 Hz).

5.4.3 Measurement Points

Table 5.2 shows the measurement points that is used for the completed measurement campaign. All of the points have been measured two times, starting at the lowest and going up

[2] The Kolmogorov scale is representing the smallest turbulent motions there is, based on viscosity and dissipation of turbulent kinetic energy.

to highest (Ramp Up), and down again (Ramp Down). This is to check for repeatability in the measurements. For ramp down measurements, the PIV recording is only done with full resolution and normal sampling time. Some of the measurement points have been taken without using PIV, only by using the sensors on the test rig. It is also beneficial to have smaller increments at velocities near lock-in, provided that this is the phenomena investigated in this study.

Table 5.2: Measurement velocities. RD = Only ramp down PIV measurements.

Velocity [m/s]	Measurement
4	PIV
6	PIV
8	PIV
9.1	PIV
9.6	PIV (RD)
10.1	PIV
11.1	PIV (RD)
11.6	Only sensors
12.1	PIV
12.6	Only sensors
13.1	PIV
14	Only sensors
15	PIV
16	Only sensors

5.4.4 Important Considerations While Recording

While doing the measurements, there are some considerations that must be taken in to account to make sure things go as planned. Constantly checking the signals from the sensors in LabVIEW to observe if there are any peculiar signals. This is to confirm that the sensors are working properly. If anything is seems wrong, it should be inspected and noted in the log file. Frequently examine the test rig for any leakages, these are likely to accrue at pressure outputs.

After each PIV recording is done, all the images must be transferred from the camera to a computer. This takes about 10 minutes. To save time, it is therefore a good idea to check through the images before storing them on the computer. If, for example, the laser is shut off, or the laser - camera timing is not matching, the pictures will be all black. This is easy to identify when checking through the images prior to storing. Before starting the PIV recording, one must also double check if the resolution, sampling frequency and number of images is correct.

5.5 Post Processing

The post processing of the PIV images that are conducted during the measurements is done with the PIV software delivered by LaVision, called DaVis. The software gives a wide range of options on how to process the data. The post processing algorithm that is used is chosen prior to starting the experiment, since this is based on how the data is collected. The method of choice in this study is cross-correlation, with using double-frame/double-exposure recording technique. Cross-correlation is further explained in Section 4.5.

Before calculating the vector fields from the images, it is convenient to pre-process the images to eliminate background noise. DaVis does this by calculating the minimum, average or Gaussian average intensity of the light reflections, and subtracting this from the images. This will remove background noise, as illustrated in Figure 5.6. When background noise is removed, the software will be able to calculate the vector fields with lower uncertainty.

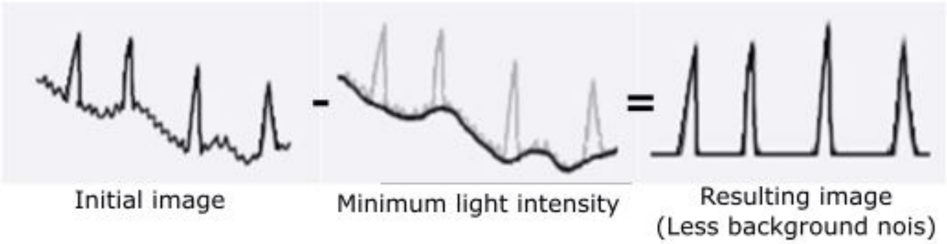


Figure 5.6: Illustration of light intensity after subtracting minimum light intensity (background noise).

When starting the vector field calculations, there are a few parameters that must be chosen based on the result that is wanted. The most important are interrogation area size and shape, and if the correlation should be done in a single step or iteratively via multi-pass. What it all comes down to is a trade of between computation time and accuracy.

To get god spatial resolution of the vector field, the interrogation areas must be small. But the limitation here is numbers of illuminated particles per area, which should be at least 10 (Section 4.3). Since the PIV algorithm calculates the vector field by tracking the displacement of a group of particles (inside an IA), there will be high uncertainty if there are to few particles in each group.

A way of getting better spatial resolution, smaller IAs, without increasing the amount of particles is by using multi pass instead of single pass. When using single pass the vector field is calculated out of a single iteration, this is the fastest way in terms of computational time. By using regular multi pass, the vector field is calculated with an arbitrary number of iterations for each image with a constant interrogation area size. The computed vector field is used as reference for the next pass. This helps to correlate the right particles and improves the signal-to-noise (LaVision, 2017), but at a higher computational coast. And the last option, which is giving better spatial resolution without going on compromise with accuracy, is multi pass with decreasing window size. The vector field is calculated with an arbitrary number of iterations and with a decreasing window size. The initial and final window size is chosen, and the calculation starts with the initial size. In the next iteration the window size is halved and the vector calculated in the previous pass is used as best-choice window shift. By doing this the the vectors in the smaller interrogation windows are calculated more accurately and more reliable. This allows to use much smaller window size, which improves spatial resolution, but again at a higher computational coast (LaVision, 2017).

The shape (weighting) of the interrogation windows can either be square, round, elliptical or adaptive. Square windows gives the fastest computation time, but will not give the best result depending on the quality of the raw data. Round weighting improves the accuracy, as the calculation will be done for twice the interrogation size but centered around the current position. This way, many outer pixels will be taken in to account. But it comes at a price of higher calculation time. Elliptical weighting is used if there exist high gradients at a given direction, e.g. in a laminar flow through a pipe. Adaptive PIV is constantly optimizing the size and weighing of the interrogation window based on local condition. This gives the best accuracy, but will also give the highest calculation time.

By testing several different set ups for post processing for a reduced set of raw data from the measurements, the selection on final set up was made based on uncertainty and number of spurious vectors in the fields. The selected set up was multi pass with decreasing window size with round weighing. The initial and final window size was set to 64x64 px and 16x16 px, respectively, with a 50% overlap (Section 4.3).

Chapter 6

Results and Discussion

Due to Covid-19 and the shut down of the university, there was not sufficient time to preform the experiments on the planned hydrofoil design with splitter plates. Therefore, it will not be presented any results for this. As a replacement, there have been done experiments on the same hydrofoil body with an other trailing edge modification, which got prioritized above the splitter plate study in a time with restricted laboratory access. The modification on the tested hydrofoil is vortex generators close to trailing edge. Vortex generators and their effect on VIVs and FSI have not been discussed earlier in this thesis, and this chapter will not be an exception. The focus in this chapter will be more on how to use the data from the measurements to obtain results that is wanted, and how to validate them.

6.1 Velocity Fields and Plots

After the post processing is finished, which is described in section 5.5, a velocity field of each image is calculated. From this, the data that is needed can be exported. Figure 6.1 shows the velocity field of one image at an inlet velocity of U_∞ = 9.1 m/s. The image shows the wake behind the hydrofoil, and the shape of the vortex shedding can be seen. By looking through the calculated velocity fields, one can see if the calculations done in DaVis implies correct in terms of what is expected.

By calculating the average of the images in one recording, the result will be the average velocity field shown in Figure 6.2 a). From this, the velocity distribution in the wake becomes clear. The standard deviation is also calculated for the average velocity field, this is shown in Figure 6.2 b). As the Figure shows, the standard deviation is higher inside the wake. This is likely due to the turbulent mechanisms which appear with higher intensity in this region. To reduce the standard deviation, the turbulent structures must be solved. This requires higher resolution in both time and space, as described in section 5.5. The average velocity field will also tell if the sampling time for each recording is adequate. If the wake is not approximately symmetrical, the chances are that the sampling time is to short and there is not captured enough periods of the vortex shedding. Figure 6.2 a) shows a symmetrical wake, which means that the flow is recorded long enough. The average velocity fields for long sample and normal sample should also show the same if the sampling period is long enough for normal sampling.

Plotting the normalized time-averaged velocity distribution at a given length down stream of the hydrofoil trailing are also giving information of the flow. In the study of a hydrofoil at lock-in by Sagmo et al. (2019), it was noted that the velocity distributions at different free stream velocities varied considerably more during lock-in. In Figure 6.3

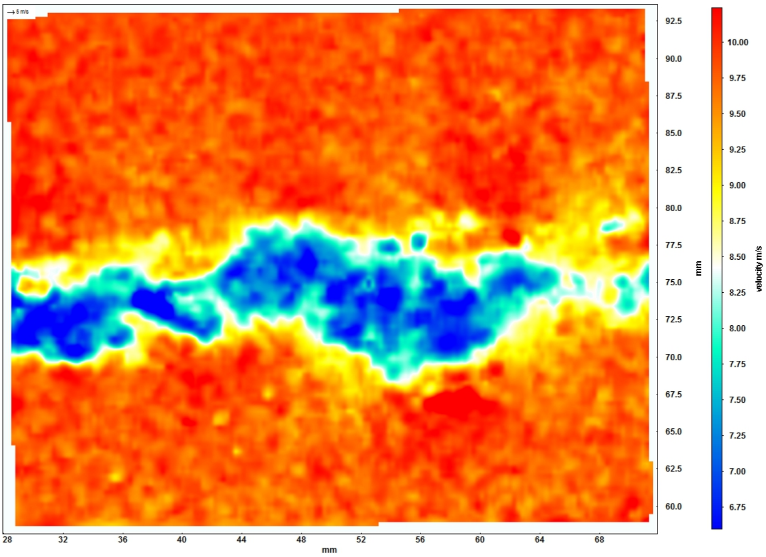
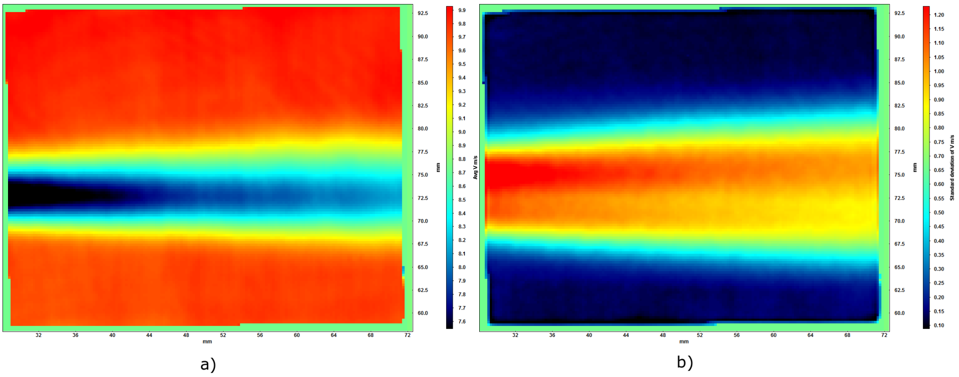


Figure 6.1: Calculated velocity field of one image at U_∞ = 9.1 m/s



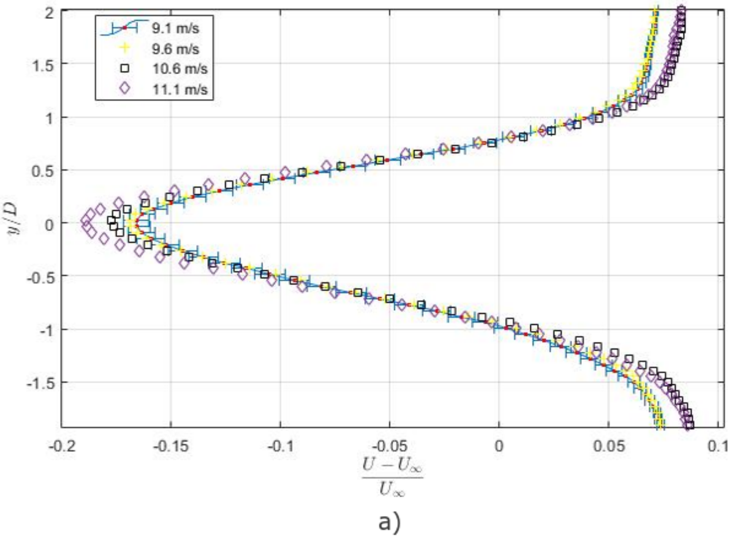
a)

b)

Figure 6.2: Average velocity field (a) and standard deviation (b) at $U_\infty = 9.1m/s$

the time-averaged velocity distributions from the PIV measurements at x = 40mm and x = 60mm down stream are presented. Based on the findings of Sagmo et al. (2019), it can be a sign that lock-in is avoided if the normalized velocity distributions are more or less the same through the entire velocity range measured.

When doing CFD analysis together with PIV, these time- averaged velocity distributions can be helpful. By plotting the time-average of the CFD results at the same points, and investigating how good it aligns with the experimental result, it gives a good remark on how reliable the CFD calculations are. Comparing different turbulence models with the experimental results will tell which method is best choice, by considering which is closest to the experimental result. It can also be used as validation of the CFD results.



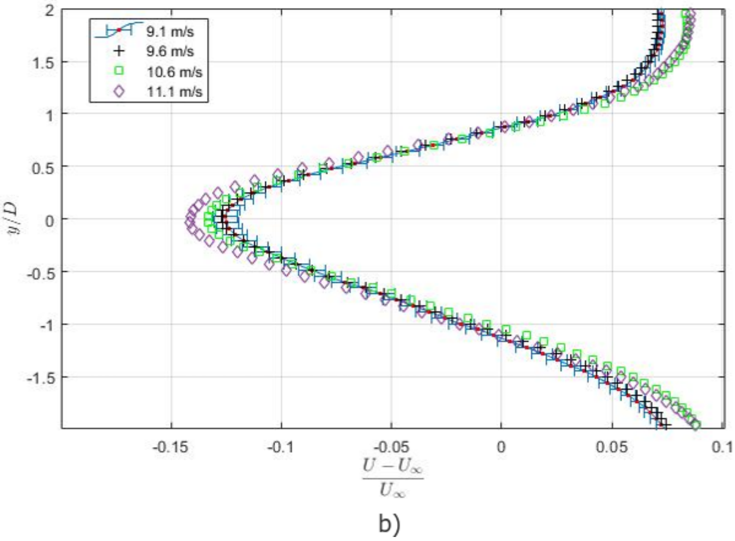


Figure 6.3: Time-averaged PIV measured velocity distributions normalised with respect to the mean channel velocity for different downstream positions. Height normalised by trailing edge thickness, D, with y = 0 set at the trailing edge tip. In (a) x = 8.3D = 40mm. In (b) x = 12.5D = 60mm. Uncertainty error bar is only plotted for reference velocity of 9.1 m/s for clarity.

6.2 Frequency Analysis

In a study on vortex induced vibrations, it is evident that the frequencies are investigated. Both the frequency of the vortex shedding, and the natural frequency of the hydrofoil are values that must be treated in the results. As mentioned, from the PIV results the shedding frequency can be found, and the strain gauge is granting the natural frequency. The shedding frequency can also be found from the strain gauge in lock-off conditions, i.e. velocities below lock-in velocity.

The velocity field is already provided from the post processing of the PIV data, described in section 5.5. The shedding frequency can be calculated from velocity data in y-direction, U_y (vertical direction), obtained from a point inside the wake behind the hydrofoil. The frequency of the velocity fluctuations in y-direction inside the wake will naturally be the same as the shedding frequency. This frequency is calculated by using Fast Fourier Transform (FFT) on the velocity data. The FFT deconstructs the signal from from a time domain representation in to a frequency domain representation (https://ni.com/), this is to analyse the different frequencies. The FFT was preformed by using a built in function in MatLab, and the code used is included in Appendix B.

The results from the FFT of the velocity data from PIV can be seen in Figure 6.4. The results are obtained from a set of velocities, yielding U_∞ = [8, 13.6] m/s. From the Figures, the shedding frequency are marked at each velocity, and as seen it is increasing with increasing velocities. The frequency is found at the peak of the FFT, plot. It is not always evident where the exact peak is, but by inspection of the Figures it can be found within a reasonable range.

The point in the wake where the velocity data is extracted is 40 mm down stream of the trailing edge. If this point is selected to far down stream, it is more difficult to pick up any fluctuations from the shedding. But the point chosen at 40mm down stream seems to be tolerable. On a side note, the vortex generators, which are used in this measurements, are disturbing the flow to hopefully reduce continual fluctuations of U_y. Therefore it is reason to believe that the peak would be clearer for measurements of a hydrofoil without the vortex generators.

The measured data from the sensors on the test rig are stored in a .tdms file created in LabView while recording. This is a file type that National Instruments (NI), the company behind LabView, have made to store the data in an effective way. When doing measurement campaigns like this, there will be a huge amount of data that is collected from all the sensors and the .tdms files gets quite big (about 2-4 GB). Different approaches can be used to access and process these data. In this case, a MatLab function (Robert , 2020) was used to convert the .tdms file to a .MAT file, and the needed data was reached from there. This function can be found in Appendix B.3 along with the script used to plot the FFT of the signal from the sensors.

Figure 6.5 shows the FFT-plots of measurements done with strain gauge, and top and bottom wall dynamic pressure sensors. As seen from the Figure the frequencies observed agrees well for all sensors. The peak is showing the natural frequency of the hydrofoil. If the vortex shedding was stronger, i.e. producing more vibrations, a second peak would be showing on the strain gauge FFT-plots. The vortex generators are possibly reducing the force of the vortex shedding and vortex induced vibrations, and that could be the reason why it is not detected by the strain gauge. The FFT-plot from the strain gauge in Figure

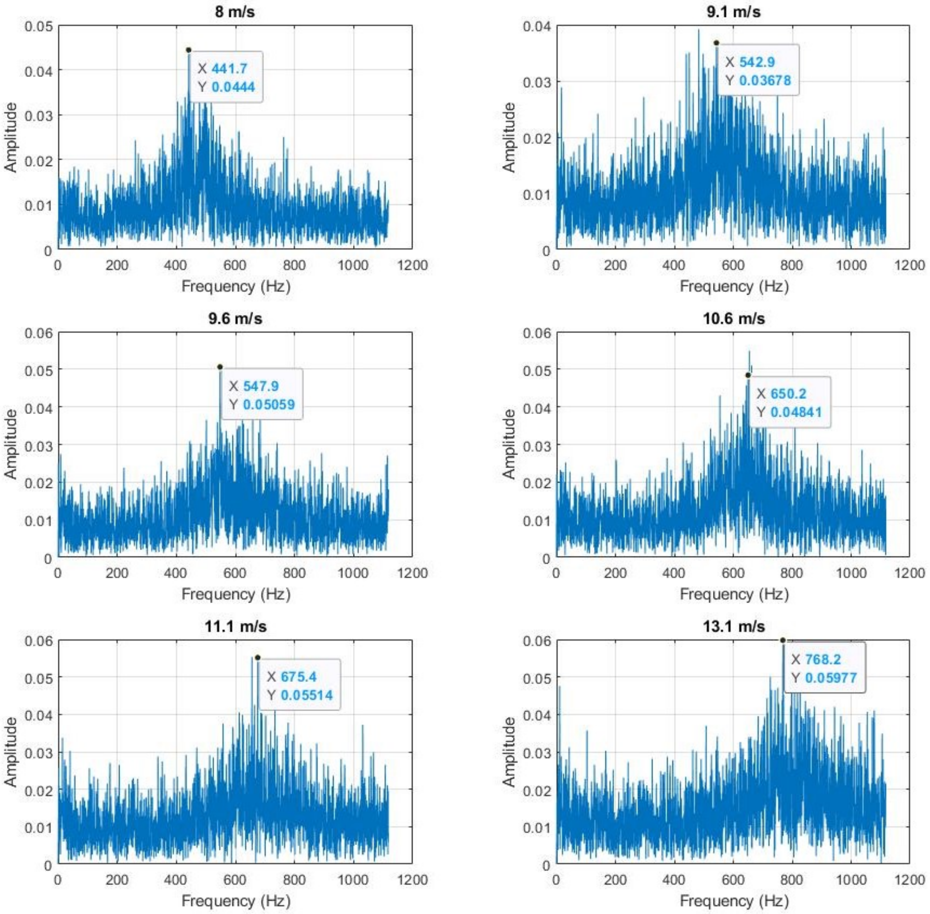


Figure 6.4: Frequency plots of velocity component in y-direction inside the wake behind the hydrofoil.

6.6 are showing two peaks. This is for a lower free stream velocity, 6 m/s, where the shedding frequency is further away from the natural frequency. This can be the reason that the frequency of the vortex shedding is detected better in this case. It is also known that the shedding frequency is not observed by the strain gauge when it is higher than the natural frequency, which also can be the case. Nevertheless, it may seem like the vortex generators are mitigating the vortex shedding by reducing continual fluctuations, without going any further with analysing the reason.

The natural frequency that is detected from the sensors are agreeing well, as both Figure 6.5 and 6.6. This agreement of the sensor gives a good verification for the measured

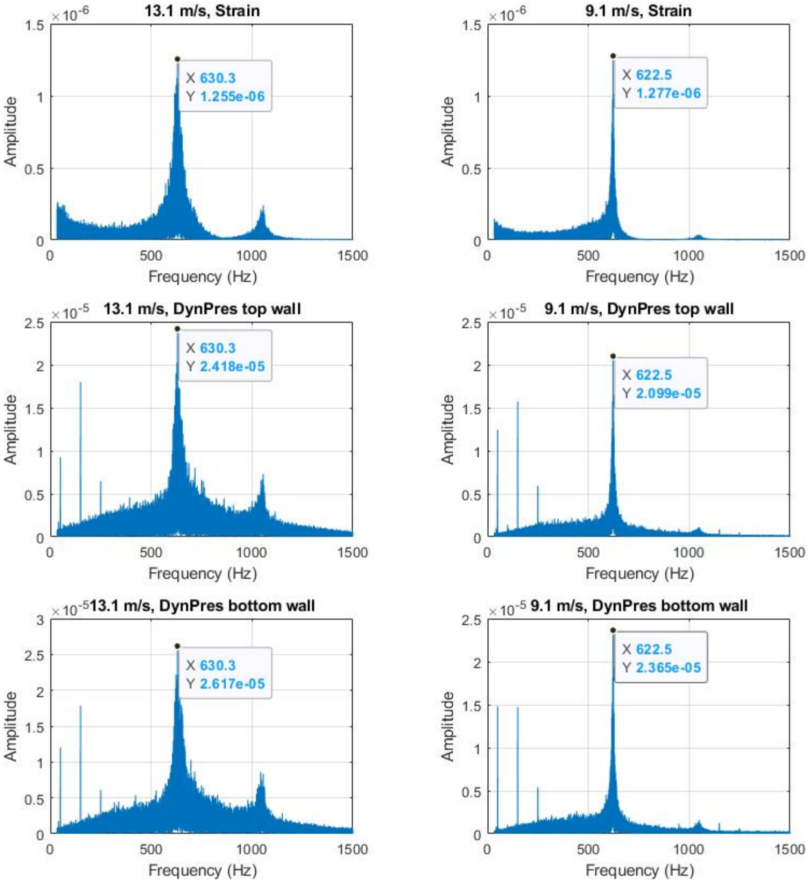


Figure 6.5: Frequency plots of sensor measurements for U_∞ = 9.1 m/s and 13.1 m/s. Top row: Strain gauge. Middle row: Top wall dynamic pressure. Bottom row: Bottom wall dynamic pressure.

data. As well as the verification, using two or more sensors that are measuring the same things give redundancy, which can be convenient if one of the sensors are failing during the measurements.

6.3 Uncertainty Analysis

When looking at the uncertainty of PIV results, there are several techniques that can be used. In this section the ones used in this thesis will be described. Initially, when looking at the calculated vector fields, vectors that are clearly wrong can be identified. These

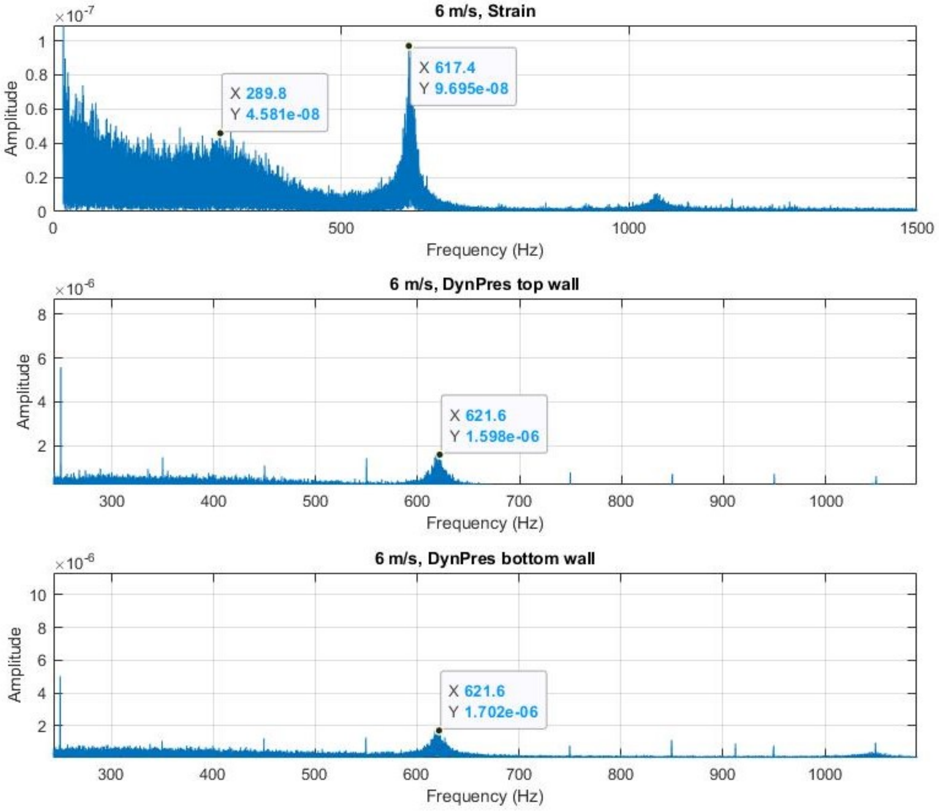
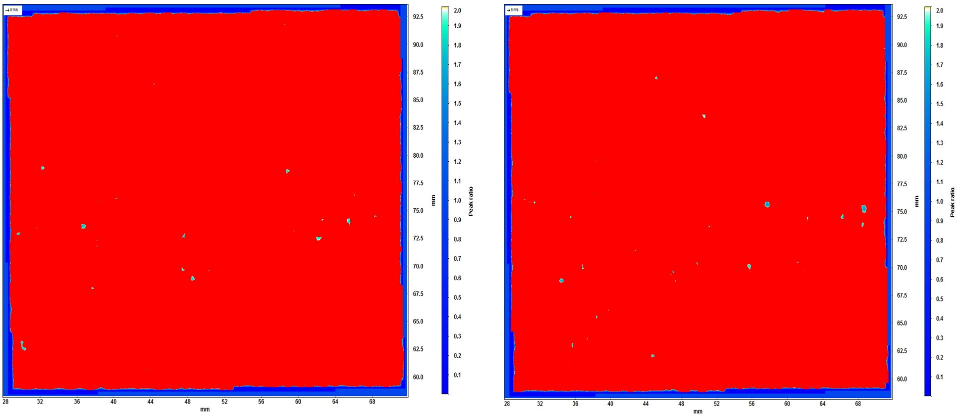


Figure 6.6: Frequency plots of sensor measurements for $U_\infty = 6m/s$. Top row: Strain gauge. Middle row: Top wall dynamic pressure. Bottom row: Bottom wall dynamic pressure.

are vectors with magnitude and direction that differ considerably from their surrounding vectors. Often, these type of spurious vectors appear at edges of the data field as single incorrect vectors (Raffel et al., 2018). Human observation of these spurious vectors is very effective as long as it is a small number of PIV recordings, but for a large number of recordings it is not longer possible. It is a good technique to use when testing different post processing setups, but when validating all the results it will be too time consuming. If the post-processing setup is good, there will be few such erroneous vectors that is easily identified. For further uncertainty analysis of the PIV results, the methods that have been used are peak ratio and uncertainty calculations done by DaVis.

The peak ratio, also known as signal-to-noise ratio, is the ratio of the correlation value of the highest and the second highest correlation peak (LaVision, 2017). The displacement of the highest peak is the one that the algorithm uses to calculate the particle displacement inside the interrogation area. The other peaks are looked at as noise. If the ratio between these two peaks are low, meaning that the ”noise”- peak and the displacement-peak are almost of similar correlation value, it is highly likely that the displacement vector is calculated based on noise and not on actual particle displacement. This again results in a high likelihood of getting spurious vectors. According to Hain and Kähler (2007), numerical investigation shows that a peak ratio of about 2.0 will reliably avoid spurious vectors. This is used as benchmark when looking at the peak ratio of the calculated vector fields.

Figure 6.7 are showing peak ratio calculations for the velocity fields that have been used to present results in this chapter. As seen from the Figures, there are not many spurious vectors in the fields. Most of the images are fully red, peak ratio above 2.0, with small dots of low signal-to-noise ratio. This shows that the results are reliable in terms of spurious vectors. At the edges, one can see that the peak ratio is very low. This is not very surprising, as particles are traveling in and out of the interrogation areas without being recorded at the edge of the frame.



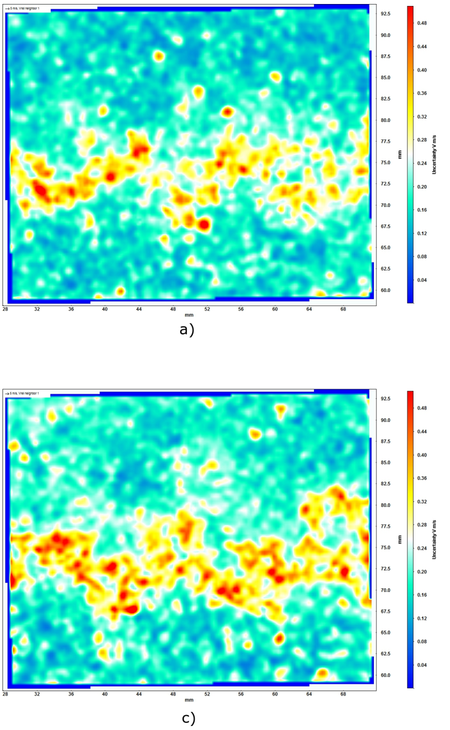
a)

b)

Figure 6.7: Peak ratio calculated for an arbitrary image at two different velocities, a) U_∞ = 11.1 m/s and b) U_∞ = 13.1 m/s. Red field equals peak ratio above 2.0, other colors equals peak ratio below 2.0

When calculating the vector field with using multi-pass, DaVis will calculate the uncertainty of the vector field. DaVis estimates the uncertainty of each vector based on a method presented by Wieneke (2015). The method works by matching the two corresponding particle images onto each other. In an ideal noise-free case, these images would match perfectly onto each other creating a symmetrical correlation peak. Most of the time, this is not the case and such paired images does not match perfectly, creating a non-symmetrical correlation peak. The uncertainty of the displacement vector are then derived by using statistical analysis on how each pixel contribute to the cross-correlation peak shape. This method have been tested for various types of noise and experimental conditions, and have shown to provide an accurate estimate of true error.

In Figure 6.8, uncertainty fields for velocities of 9.1 to 13.1 m/s are shown. For all cases, the uncertainty is highest inside the wake. This is most likely due to higher turbulence intensity inside the wake. This also connects with the standard deviation of the



a)

b)

c)

d)

Figure 6.8: Estimated uncertainty of displacement vectors for an arbitrary velocity field at four different velocities (color bar on right side of the figures ranging from 0 to 0.48 m/s), a) U_∞ = 9.1 m/s, b) U_∞ = 9.6 m/s, c) U_∞ = 11.1 m/s and d) U_∞ = 13.1 m/s.

average velocity field, discussed earlier in section 6.1. Since the uncertainty of the displacement is calculated based on matching particle image pairs, it is no surprise that the uncertainty will be higher in regions of high turbulence. It is also noted that the uncertainty increases with increasing velocity, probably due to increasing turbulence. Figure 6.8 a) shows a maximum uncertainty of about 5.2%, which was considered adequate. Since all the uncertainty fields in Figure 6.8 have the same scale in velocity, it should also be mentioned that the areas of maximum uncertainties display lower values (in percentage) for increasing velocity. With maximum displayed uncertainty of 5%, 4.32% and 3.66% for b), c) and d) respectively in the deep read fields. But this maximum uncertainty are showing in an increasing share of the vector field, and there is no guarantee that the uncertainty is not higher in these areas of highest uncertainty. Never the less, the over all uncertainty is considered not to give any remarkable impact on the results.

Chapter 7

Conclusion

A measurement campaign for investigating a splitter plates effect on mitigation of vortex induced vibrations with using PIV have been planed and prepared, but not carried out due to the outbreak of Covid-19. Instead, the results from a similar measurement campaign have been presented, with the focus moved more over on to the experimental method instead of the mitigation of VIVs. The reason being that the used trailing edge design have not been discussed earlier in the thesis and therefore there is not sufficient theoretical support to investigate how this trailing edge design is performing, and if the flow behaves as expected.

Since there are no results on how well splitter plates preform on mitigating vortex shedding and VIVs, it is not possible to draw a conclusion on weather the prepared design would work as imagined. Instead, a conclusion on how well the experimental method of choice will function for measurements on the prepared design.

The test rig at the Waterpower laboratory with the PIV setup and sensors is clearly a good tool for investigation of fluid structure interactions, and the effect of different trailing edge modifications. Being able to obtain the shedding frequency of the wake flow combined with the natural frequency of the hydrofoil gives a good base to obtain the data that should be investigated. This would also be the case for a trailing edge design with splitter plates. By comparing different plate lengths and materials with a reference case, it can easily be deducted from the results how effective the different designs are in terms of mitigating VIVs. One possible challenge could be if the splitter plate is to long, so it is visible in the PIV field of view. Having a vibrating plate inside the field of view could disturb the cross-correlation when post processing. This can result in velocity fields of higher uncertainty. Another downside is the time consumption of the process, like with most laboratory work in general. Because of this, good planning of the measurement campaign is required, and a flexible time perspective if any unexpected incidents should occur. For future work on this project, a tentative plan of time perspective when doing PIV measurements are presented in section 8.2.

All in all, the method of using PIV combined with the strain gauge is a reliable and good way of conducting results for this study. Even if the process is time consuming, there are not many methods that are as reliable. Using CFD is an option, but the results would have to be validated with experimental results to have any scientific value. More on this in section 8.1.

Chapter 8

Further Work

The planned measurement campaign of the hydrofoil fitted with splitter plate was not concluded, due to the circumstances. Therefore, it is obvious that further work for this project is to execute the measurement campaign that was the initial plan for this thesis. By using the design proposal and experiences from PIV measurements that is discussed in this thesis, it gives a good starting point for future work on the same topic. In addition to completing the planned measurement campaign presented in this thesis, using CFD combined with the PIV is also something that would strengthen the study. This is discussed in the next section. The last section presents an evaluation of time usage when doing PIV, which can be used as an aid for future work on this project.

8.1 Computational Fluid Dynamics

It is an established fact that a CFD study does not always stand by it self as a good and reliable result when used in science. Reliable CFD results can be achieved by using Direct Numerical Simulation (DNS), which fully resolves the unsteady Navier-Stokes equations. Using DNS to study the case of this thesis would require a tremendous amount of computational power, given that it is a high Reynolds number flow. A flow with Reynolds number of $Re_L = 96,000$ would take about 5000 CPU years to calculate with DNS according to Pope (2000). And with a chord based Reynolds number of $Re \approx 2.6 * 10^6$ at U_∞ = 11.1 m/s, it is obvious that DNS is not possible to use for this study. Therefore, if CFD is to use, it must be done with Reynolds Averaged Navier-Stokes (RANS) methods, with turbulence modelling. By using these methods and the experimental results to validate the CFD, it can be very helpful. Modifications on the hydrofoil, e.g. change splitter plate length, are done quickly with CFD, and the results can be used to optimize the design that is tested with PIV.

With the PIV setup and all the sensors at the rig, things are well suited for validation of CFD results. The measurements of the pressure distribution around the hydrofoil are very useful to combine with CFD, as it helps to quantify the wall-model chosen for turbulence closure. Comparing results of velocity distributions inside the wake with PIV measurement will also give good validation of CFD results.

All in all, doing CFD simulations of this study combined with the PIV will complement the measurement campaign very well. By helping to find ideal designs and hence reduce the number of required PIV measurements, valuable time can be saved, given that lab work always requires more time than first assumed.

8.2 Evaluation of Time Perspective

When doing experimental laboratory work, it is important to set time frames to keep track of progress. With that said, it is not always easy to predict how much time each process will take, since there are usually unanticipated challenges appearing. Figure 8.1 shows an overview of the whole measurement campaign from start to finish, and advised time frames for each of the processes. As mentioned, it is hard to give exact time frames on this kind of work, but figure 8.1 gives an idea of what can be expected.

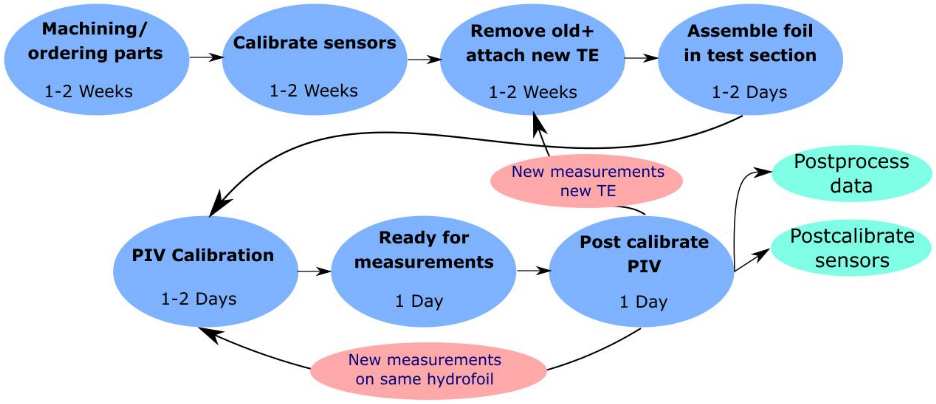


Figure 8.1: Overview of processes and suggested time frames for each process when doing PIV measurements with hydrofoil at the Waterpower laboratory, NTNU

When doing a PIV measurement campaign, it is likely that there will be carried out several different measurements. Both on different hydrofoil configurations, or in different planes on the same hydrofoil. Figure 8.1 is also showing where to jump back in the time frame when doing new measurements of any kind.

Bibliography

Baker, J.P., van Dam, C.P., 2008. Drag reduction of blunt trailing-edge airfoils, in: BBAA VI International Colloquium on: Bluff Bodies Aerodynamics & Applications, Milano, Italy.

Bearman, P.W., 1965. Investigation of the flow behind a two-dimensional model with a blunt trailing edge and fitted with splitter plates. Journal of Fluid Mechanics 21, 241–255.

Blake, W.K., 1984. Excitation of plates and hydrofoils by trailing edge flows. Journal of Vibration and Acoustics 106, 351–363.

Dai, S., Younis, B.A., Zhang, H., Guo, G., 2018. Prediction of vortex shedding suppression from circular cylinders at high reynolds number using base splitter plates. Journal of Wind Engineering & Industrial Aerodynamics 182, 115–127.

Grant, I., 1997. Particle image velocimetry: A review. Journal of Mechanical Engineering Science 211, 55–76.

Hain, R., Kähler, C.J., 2007. Fundamentals of multiframe particle image velocimetry (piv). Experiments in Fluids 42, 575–587.

https://kiro7.com/, . Tacoma bridge. `https://www.kiro7.com/news/tacoma-narrows-bridge-collapses-on-november-7-1940/1006176170/`. Accessed: 2020-05-28.

https://new.abb.com/, . Robust flowmeter. `https://new.abb.com/products/measurement-products/flow/electromagnetic-flowmeters/process-industry/fsm4000`. Accessed: 2020-05-19.

https://ni.com/, . Understanding ffts and windowing. `https://www.ni.com/en-no/innovations/white-papers/06/understanding-ffts-and-windowing.html`. Accessed: 2020-06-01.

https://steemit.com/, . Vortex shedding and how it works. `https://steemit.com/science/@frostfractal/the-sound-of-wind-vortex-shedding-and-how-it-works/`. Accessed: 2020-05-28.

https://www.dantecdynamics.com/, . Measurement principles of piv. `https://www.dantecdynamics.com/solutions-applications/solutions/fluid-mechanics/particle-image-velocimetry-piv/measurement-principles-of-piv/`. Accessed: 2020-04-01.

Kwon, K., Choi, H., 1996. Control of laminar vortex shedding behind a circular cylinder using splitter plates. Physics of Fluids 8, 479–486.

LaVision, 2017. FlowMaster Product-Manual. 84 ed. LaVision GmbH. Anna-Vandenhoeck-Ring 19, D-37081 Goettingen. Document name: 1003005_FlowMaster_D84.pdf.

Pope, S.B., 2000. Turbulent Flows. Cambridge University Press.

Raffel, M., Willert, C.E., Fulivo, S., Kähler, C.J., Wereley, A.T., Kompenhans, J., 2018. Particle Image Velocimetry. Springer.

Robert (2020), . Converttdms (v9), matlab central file exchange. `https://www.mathworks.com/matlabcentral/fileexchange/28771-converttdms-v91`. Accessed: 2020-06-01.

Sagmo, K.F., Tengs, E.O., Bergan, C.W., Storli, P.T., 2019. PIV measurements and CFD simulations of a hydrofoil at lock-in. IOP Conference Series: Earth and Environmental Science 240, 1755–1315.

Sarpkaya, T., 1979. Vortex-induced oscillations: A selective review. JOURNAL OF APPLIED MECHANICS 46, 241–258.

Vergine, F., Maddalena, L., 2014. Stereoscopic particle image velocimetry measurements of supersonic, turbulent, and interacting streamwise vortices: challenges and application. Progress in Aerospace Sciences 66, 1–16.

Wieneke, B., 2015. Piv uncertainty quantification from correlation statistics. Meas. Sci. Technol 26.

Appendices

Appendix A

Calibration Reports

A.1 Absolute pressure sensor

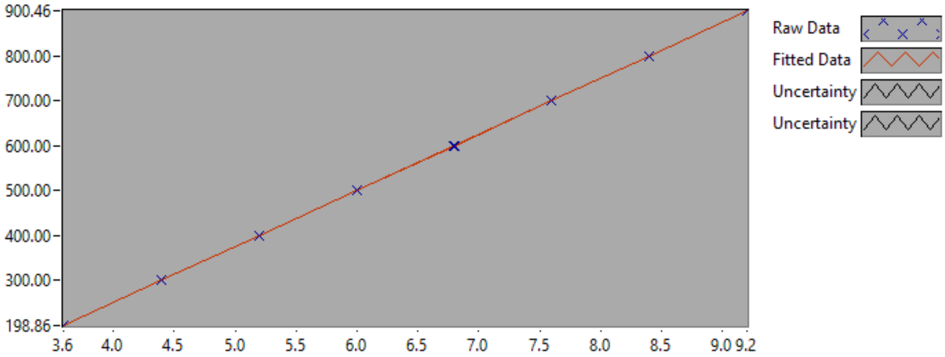


Figure A.1: Calibration chart (The uncertainty band is multiplied by 10)

CALIBRATION REPORT

CALIBRATION PROPERTIES

Calibrated by: Kristian Sagmo
Type/Producer: Druck Unik 5000
SN: 3689526
Range: 0-10 bar
Unit: Pa

CALIBRATION SOURCE PROPERTIES

Type/Producer: Pressurements deadweight tester P3223-1
SN: 66256
Uncertainty [%]: 0,008

POLY FIT EQUATION:

Y= -249.80057817E+0X^0 + 124.98392463E+0X^1

CALIBRATION SUMARY:

Max Uncertainty : 0.011920 [%]
Max Uncertainty : 0.025019 [Pa]
RSQ : 1.000000
Calibration points : 29

Figure 1 : Calibration chart (The uncertainty band is multiplied by 10)

____
Kristian Sagmo

CALIBRATION VALUES

Value [Pa]	Voltage [V]	Best Poly Fit [Pa]	Deviation [Pa]	Uncertainty [%]	Uncertainty [Pa]
199.163306	3.591638	199.096398	0.066908	0.011920	0.023741
299.324377	4.393599	299.328618	-0.004241	0.006334	0.018960
399.475448	5.195136	399.507881	-0.032433	0.003786	0.015125
499.626519	5.996774	499.699714	-0.073196	0.002586	0.012920
599.777590	6.798075	599.849544	-0.071955	0.002214	0.013279
699.938661	7.599195	699.976640	-0.037979	0.002289	0.016019
800.089732	8.400281	800.099567	-0.009835	0.002511	0.020088
900.250802	9.201236	900.206047	0.044755	0.002776	0.024995
800.099732	8.400195	800.088718	0.011014	0.002521	0.020168
699.948661	7.599100	699.964733	-0.016073	0.002285	0.015992
599.797590	6.797891	599.826554	-0.028964	0.002217	0.013296
499.656519	5.996677	499.687674	-0.031155	0.002590	0.012943
399.505448	5.195180	399.513359	-0.007912	0.003790	0.015143
299.354377	4.393808	299.354850	-0.000474	0.006333	0.018958
199.203306	3.592295	199.178513	0.024792	0.011895	0.023695
299.354377	4.393733	299.345412	0.008965	0.006338	0.018974
399.505448	5.195021	399.493503	0.011945	0.003787	0.015130
499.656519	5.996515	499.667382	-0.010864	0.002593	0.012957
599.817590	6.797880	599.825124	-0.007534	0.002217	0.013300
699.968661	7.599049	699.958340	0.010320	0.002284	0.015984
800.119732	8.400171	800.085768	0.033964	0.002516	0.020132
900.270802	9.201230	900.205253	0.065549	0.002779	0.025019
800.119732	8.400237	800.094044	0.025687	0.002515	0.020120
699.968661	7.599088	699.963233	0.005428	0.002283	0.015982
599.817590	6.797950	599.833848	-0.016258	0.002210	0.013255
499.666519	5.996728	499.693974	-0.027455	0.002583	0.012907
399.525448	5.195207	399.516744	0.008704	0.003778	0.015096
299.374377	4.393831	299.357716	0.016661	0.006328	0.018943
199.233306	3.592400	199.191672	0.041634	0.011888	0.023684

COMMENTS:

The uncertainty is calculated with 95% confidence. The uncertainty includes the randomness in the calibrated instrument during the calibration, systematic uncertainty in the instrument or property which the instrument under calibration is compared with (dead weight manometer, calibrated weights etc.), and due to regression analysis to fit the calibration points to a linear calibration equation.The calculated uncertainty can be used as the total systematic uncertianty of the calibrated instrument with the given calibration equation.

A.2 Pressure sensor rack

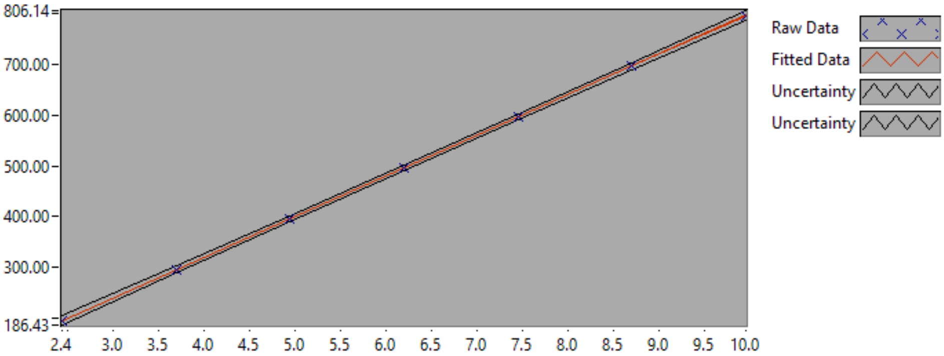


Figure A.2: Calibration chart (The uncertainty band is multiplied by 1000)

CALIBRATION REPORT

CALIBRATION PROPERTIES

Calibrated by: Kristian Sagmo
Type/Producer: GE Druck PMP Unik 5000
SN: 5453978
Range: 0-8 bar
Unit: kPa

CALIBRATION SOURCE PROPERTIES

Type/Producer: Pressurements deadweight tester P3223-1
SN: 66256
Uncertainty [%]: 0,008

POLY FIT EQUATION:

Y= + 509.80516779E-3X^0 + 79.94623299E+0X^1

CALIBRATION SUMARY:

Max Uncertainty : 0.004731 [%]
Max Uncertainty : 0.009539 [kPa]
RSQ : 1.000000
Calibration points : 28

Figure 1 : Calibration chart (The uncertainty band is multiplied by 1000)

____
Kristian Sagmo

CALIBRATION VALUES

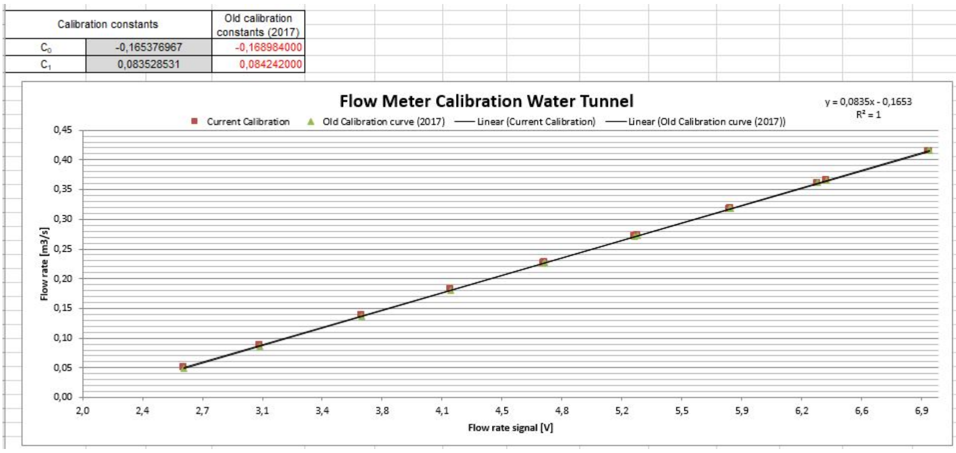
Value [kPa]	Voltage [V]	Best Poly Fit [kPa]	Deviation [kPa]	Uncertainty [%]	Uncertainty [kPa]
195.756654	2.442216	195.755745	0.000909	0.004725	0.009250
295.887725	3.694643	295.882628	0.005097	0.002454	0.007260
396.028796	4.947576	396.049831	-0.021035	0.001538	0.006089
496.189867	6.200319	496.201951	-0.012084	0.001049	0.005203
596.340938	7.452942	596.344458	-0.003521	0.000977	0.005825
696.472009	8.705699	696.497647	-0.025638	0.001060	0.007385
796.623080	9.958202	796.630571	-0.007492	0.001175	0.009363
796.623080	9.957874	796.604324	0.018756	0.001197	0.009539
696.462009	8.705303	696.465950	-0.003941	0.001050	0.007311
596.300938	7.452480	596.307530	-0.006592	0.000968	0.005769
496.159867	6.199698	496.152285	0.007582	0.001043	0.005175
396.008796	4.946934	395.998524	0.010272	0.001456	0.005765
295.857725	3.693970	295.828820	0.028905	0.002456	0.007268
195.676654	2.441474	195.696450	-0.019796	0.004731	0.009258
195.676654	2.441355	195.686937	-0.010283	0.004729	0.009254
295.837725	3.693908	295.823870	0.013855	0.002456	0.007265
395.978796	4.946583	395.970499	0.008297	0.001461	0.005787
496.139867	6.199424	496.130409	0.009458	0.001050	0.005207
596.290938	7.452201	596.285204	0.005733	0.000966	0.005761
696.442009	8.705143	696.453171	-0.011162	0.001050	0.007311
796.593080	9.957466	796.571701	0.021378	0.001172	0.009334
796.603080	9.957662	796.587363	0.015716	0.001167	0.009294
696.452009	8.705239	696.460867	-0.008858	0.001052	0.007327
596.300938	7.452515	596.310282	-0.009344	0.000965	0.005753
496.149867	6.199843	496.163907	-0.014040	0.001038	0.005152
396.008796	4.947057	396.008343	0.000452	0.001466	0.005807
295.857725	3.694379	295.861508	-0.003783	0.002453	0.007259
195.716654	2.441587	195.705496	0.011158	0.004725	0.009248

COMMENTS:

Reference sensor for 16 pressure channel pressure rack.

The uncertainty is calculated with 95% confidence. The uncertainty includes the randomness in the calibrated instrument during the calibration, systematic uncertainty in the instrument or property which the instrument under calibration is compared with (dead weight manometer, calibrated weights etc.), and due to regression analysis to fit the calibration points to a linear calibration equation.The calculated uncertainty can be used as the total systematic uncertianty of the calibrated instrument with the given calibration equation.

A.3 Flowmeter



Calibration constants		Old calibration constants (2017)
C_0	-0,165376967	-0,168984000
C_1	0,083528531	0,084242000

Figure A.3: Flowmeter calibration chart

Appendix B

MatLab Scripts and Functions

B.1 FFT plot from PIV data

```
clear, clc

A = importdata('8ms_Vy.dat');
B = importdata('9_1ms_Vy.dat');
C = importdata('9_6ms_Vy.dat');
D = importdata('10_6ms_Vy.dat');
E = importdata('11_1ms_Vy.dat');
F = importdata('13_1ms_Vy.dat');

time = (1/2441)*A.data(:,1);      %[s]

vy_A = A.data(:,3);
vy_B = B.data(:,3);
vy_C = C.data(:,3);
vy_D = D.data(:,3);
vy_E = E.data(:,3);
vy_F = F.data(:,3);


Vy_vect = [vy_A, vy_B, vy_C, vy_D, vy_E, vy_F];
n = 6;
%b = zeros(n:6);
%fd = zeros(n);


Fs = 1/(time(2)-time(1));          % Sampling Frequency
Fn = Fs/2;                         % Nyquist Frequency
Ts = 1/Fs;                         % Sampling Interval

for i = 1:n
Ld = size(Vy_vect(:,i),1);                 % Length Of Data Vectors
t = linspace(0,Ld/Fs,Ld);          % Time Vector
d = bsxfun(@minus, Vy_vect(:,i), mean(Vy_vect(:,i))); % Subtract Means From Each Vector
fd = fft(Vy_vect(:,i))/Ld;                 % Calculate FFT
b(:,i) = abs(fd);                       % Absolute Magnitudes
Fv = linspace(0, 1, Ld/2+1)*Fn;    % Frequency Vector
%Iv = 1:length(Fv);                 % Index Vector
end

Fv_plot = Fv(1:2000);
%plot
```

```
t=tiledlayout(3,2);

nexttile
plot(Fv_plot, b((1:2000),1))                  % Plot FFTs (2D)
title('8 m/s')
grid
xlabel('Frequency (Hz)')
ylabel('Amplitude')


nexttile
plot(Fv_plot, b((1:2000),2))                  % Plot FFTs (2D)
title('9.1 m/s')
grid
xlabel('Frequency (Hz)')
ylabel('Amplitude')


nexttile
plot(Fv_plot, b((1:2000),3))                  % Plot FFTs (2D)
title('9.6 m/s')
grid
xlabel('Frequency (Hz)')
ylabel('Amplitude')


nexttile
plot(Fv_plot, b((1:2000),4))                  % Plot FFTs (2D)
title('10.6 m/s')
grid
xlabel('Frequency (Hz)')
ylabel('Amplitude')


nexttile
plot(Fv_plot, b((1:2000),5))                  % Plot FFTs (2D)
title('11.1 m/s')
grid
xlabel('Frequency (Hz)')
ylabel('Amplitude')



nexttile
plot(Fv_plot, b((1:2000),6))                  % Plot FFTs (2D)
title('13.1 m/s')
grid
xlabel('Frequency (Hz)')
ylabel('Amplitude')
```

B.2 FFT plot from sensor measurements

```
clear, clc

A = convertTDMS(true, 'RampDown.tdms');
```

```
%B = convertTDMS(true, 'RampUp.tdms');


Strain_A = A.Data.MeasuredData(29).Data;        %Collecting strain data from .MAT file
Strain_B = A.Data.MeasuredData(211).Data;       %Collecting strain data from .MAT file
DynPres_A = A.Data.MeasuredData(31).Data;       %Collecting dynamic pressure data from .MA
DynPres_B = A.Data.MeasuredData(213).Data;      %Collecting dynamic data from .MAT file
DynPres_C = A.Data.MeasuredData(32).Data;       %Collecting dynamic pressure data from .MA
DynPres_D = A.Data.MeasuredData(214).Data;      %Collecting dynamic data from .MAT file


%time =       %[s]


Vy_vect = [Strain_A, Strain_B, DynPres_A, DynPres_B, DynPres_C, DynPres_D];
n = 6;
%b = zeros(n:6);
%fd = zeros(n);


%Fs = 1/(time(2)-time(1));          % Sampling Frequency
Fs = length(Strain_A)/60;           %Sampling Frequency (60 second long measurement)
Fn = Fs/2;                       % Nyquist Frequency
Ts = 1/Fs;                         % Sampling Interval

for i = 1:n
Ld = size(Vy_vect(:,i),1);                 % Length Of Data Vectors
t = linspace(0,Ld/Fs,Ld);          % Time Vector
d = bsxfun(@minus, Vy_vect(:,i), mean(Vy_vect(:,i))); % Subtract Means From Each Vector
fd = fft(Vy_vect(:,i))/Ld;                 % Calculate FFT
b(:,i) = abs(fd);                       % Absolute Magnitudes
Fv = linspace(0, 1, Ld/2+1)*Fn;    % Frequency Vector
%Iv = 1:length(Fv);                 % Index Vector
end


%plot
t=tiledlayout(3,2);

nexttile
plot(Fv (2000:90000), b((2000:90000),1))               % Plot FFTs (2D)
title('13.1 m/s, Strain')
grid
xlabel('Frequency (Hz)')
ylabel('Amplitude')


nexttile
plot(Fv (2000:90000), b((2000:90000),2))               % Plot FFTs (2D)
title('9.1 m/s, Strain')
grid
xlabel('Frequency (Hz)')
ylabel('Amplitude')

nexttile
plot(Fv (2000:90000), b((2000:90000),3))               % Plot FFTs (2D)
title('13.1 m/s, DynPres top wall')
```

```
grid
xlabel('Frequency (Hz)')
ylabel('Amplitude')

nexttile
plot(Fv (2000:90000), b((2000:90000),4))               % Plot FFTs (2D)
title('9.1 m/s, DynPres top wall')
grid
xlabel('Frequency (Hz)')
ylabel('Amplitude')

nexttile
plot(Fv (2000:90000), b((2000:90000),5))               % Plot FFTs (2D)
title('13.1 m/s, DynPres bottom wall')
grid
xlabel('Frequency (Hz)')
ylabel('Amplitude')

nexttile
plot(Fv (2000:90000), b((2000:90000),6))               % Plot FFTs (2D)
title('9.1 m/s, DynPres bottom wall')
grid
xlabel('Frequency (Hz)')
ylabel('Amplitude')
```

B.3 TDMS converter (Robert , 2020)

```
function [ConvertedData,ConvertVer,ChanNames,GroupNames,ci] = convertTDMS(varargin)
%Function to load LabView TDMS data file(s) into variables in the MATLAB workspace.
%An *.MAT file can also be created.  If called with one input, the user selects
%a data file.
%
%   TDMS format is based on information provided by National Instruments at:
%   http://zone.ni.com/devzone/cda/tut/p/id/5696
%
% [ConvertedData,ConvertVer,ChanNames]=convertTDMS(SaveConvertedFile,filename)
%
%       Inputs:
%               SaveConvertedFile (required) - Logical flag (true/false) that
%                 determines whether a MAT file is created.  The MAT file's name
%                 is the same as 'filename' except that the 'TDMS' file extension is
%                 replaced with 'MAT'.  The MAT file is saved in the same folder
%                 and will overwrite an existing file without warning.  The
%                 MAT file contains all the output variables.
%
%               filename (optional) - Filename (fully defined) to be converted.
%                 If not supplied, the user is provided a 'File Open' dialog box
%                 to navigate to a file.  Can be a cell array of files for bulk
%                 conversion.
%
%       Outputs:
%               ConvertedData (required) - Structure of all of the data objects.
%               ConvertVer (optional) - Version number of this function.
%               ChanNames (optional) - Cell array of channel names
%               GroupNames (optional) - Cell array of group names
```

```
%
%
%'ConvertedData' is a structure with 'FileName', 'FileFolder', 'SegTDMSVerNum',
%'NumOfSegments' and 'Data' fields'. The 'Data' field is a structure.
%
%'ConvertedData.SegTDMSVerNum' is a vector of the TDMS version number for each
%segment.
%
%'ConvertedData.Data' is a structure with 'Root' and 'MeasuredData' fields.
%
%'ConvertedData.Data.Root' is a structure with 'Name' and 'Property' fields.
%The 'Property' field is also a structure; it contains all the specified properties
%(1 entry for each 'Property) for the 'Root' group. For each 'Property' there are
%'Name' and 'Value' fields. To display a list of all the property names, input
%'{ConvertedData.Data.Root.Property.Name}'' in the Command Window.
%
%'ConvertedData.Data.MeasuredData' is a structure containing all the channel/group
%information. For each index (for example, 'ConvertedData.Data.MeasuredData(1)'),
%there are 'Name', 'Data' and 'Property' fields.  The list of channel names can
%be displayed by typing 'ChanNames' in the Command Window.  Similarly, the list
%of group names can be displayed by typing 'GroupNames' in the Command Window.
%The 'Property' field is also a structure; it contains all the specified properties
%for that index (1 entry in the structure for each 'Property'). Any LabView waveform
%attributes ('wf_start_time', 'wf_start_offset', 'wf_increment' and 'wf_samples') that
%may exist are also included in the properties. For each 'Property' there are 'Name'
%and 'Value' fields.  To display a list of all the property names, input
%'{ConvertedData.Data.MeasuredData(#).Property.Name}'' in the Command Window
%where '#' is the index of interest.
%

%-------------------------------------------------------------------------
%Brad Humphreys - v1.0 2008-04-23
%ZIN Technologies
%-------------------------------------------------------------------------

%-------------------------------------------------------------------------
%Brad Humphreys - v1.1 2008-07-03
%ZIN Technologies
%-Added abilty for timestamp to be a raw data type, not just meta data.
%-Addressed an issue with having a default nsmaples entry for new objects.
%-Added Error trap if file name not found.
%-Corrected significant problem where it was assumed that once an object
%    existsed, it would in in every subsequent segement.  This is not true.
%-------------------------------------------------------------------------

%-------------------------------------------------------------------------
%Grant Lohsen - v1.2 2009-11-15
%Georgia Tech Research Institute
%-Converts TDMS v2 files
%Folks, it's not pretty but I don't have time to make it pretty. Enjoy.
%-------------------------------------------------------------------------

%-------------------------------------------------------------------------
%Jeff Sitterle - v1.3 2010-01-10
%Georgia Tech Research Institute
%Modified to return all information stored in the TDMS file to inlcude
%name, start time, start time offset, samples per read, total samples, unit
```

```
%description, and unit string.  Also provides event time and event
%description in text form
%Vast speed improvement as save was the previous longest task
%--------------------------------------------------------------------------

%--------------------------------------------------------------------------
%Grant Lohsen - v1.4 2009-04-15
%Georgia Tech Research Institute
%Reads file header info and stores in the Root Structure.
%--------------------------------------------------------------------------

%--------------------------------------------------------------------------
%Robert Seltzer - v1.5 2010-07-14
%BorgWarner Morse TEC
%-Tested in MATLAB 2007b and 2010a.
%-APPEARS to now be compatible with TDMS version 1.1 (a.k.a 4712) files;
%   although, this has not been extensively tested.  For some unknown
%   reason, the version 1.2 (4713) files process noticeably faster. I think
%   that it may be related to the 'TDSm' tag.
%-"Time Stamp" data type was not tested.
%-"Waveform" fields was not tested.
%-Fixed an error in the 'LV2MatlabDataType' function where LabView data type
%   'tdsTypeSingleFloat' was defined as MATLAB data type 'float64' .  Changed
%   to 'float32'.
%-Added error trapping.
%-Added feature to count the number of segments for pre-allocation as
%   opposed to estimating the number of segments.
%-Added option to save the data in a MAT file.
%-Fixed "invalid field name" error caused by excessive string lengths.
%--------------------------------------------------------------------------




%Initialize outputs
ConvertVer='1.95';    %Version number of this conversion function
ConvertedData=[];


p=inputParser();

p.addRequired('SaveConvertedFile',@(x) islogical(x)||(ismember(x,[0,1])));
p.addOptional('filename','',@(x) iscell(x)||exist(x,'file'));
p.parse(varargin{:});

filename=p.Results.filename;
SaveConvertedFile=p.Results.SaveConvertedFile;

if isempty(filename)

    %Prompt the user for the file
    [filename,pathname]=uigetfile({'*.tdms','All Files (*.tdms)'},'Choose a TDMS File');
    if filename==0
        return
    end
    filename=fullfile(pathname,filename);
```

```
end


if iscell(filename)
    %For a list of files
    infilename=filename;
else
    infilename=cellstr(filename);
end

for fnum=1:numel(infilename)

    if ~exist(infilename{fnum},'file')
        e=errordlg(sprintf('File ''%s'' not found.',infilename{fnum}),'File Not Found');
        uiwait(e)
        return
    end

    FileNameLong=infilename{fnum};
    [pathstr,name,ext]=fileparts(FileNameLong);
    FileNameShort=sprintf('%s%s',name,ext);
    FileNameNoExt=name;
    FileFolder=pathstr;

    if fnum==1
        fprintf('\n\n')
    end
    fprintf('Converting ''%s''...',FileNameShort)

    fid=fopen(FileNameLong);

    if fid==-1
        e=errordlg(sprintf('Could not open ''%s''.',FileNameLong),'File Cannot Be Opened')
        uiwait(e)
        fprintf('\n\n')
        return
    end

    [SegInfo,NumOfSeg]=getSegInfo(fid);
    channelinfo=getChannelInfo(fid,SegInfo,NumOfSeg);
    ob=getData(fid,channelinfo);
    fclose(fid);

    %Assign the outputs
    ConvertedData(fnum).FileName=FileNameShort;
    ConvertedData(fnum).FileFolder=FileFolder;

    ConvertedData(fnum).SegTDMSVerNum=SegInfo.vernum;
    ConvertedData(fnum).NumOfSegments=NumOfSeg;
    [ConvertedData(fnum).Data,CurrGroupNames]=postProcess(ob,channelinfo);

    GroupNames(fnum)={CurrGroupNames};

    TempChanNames={ConvertedData(fnum).Data.MeasuredData.Name};
    TempChanNames(strcmpi(TempChanNames,'Root'))=[];
    ChanNames(fnum)={sort(setdiff(TempChanNames',CurrGroupNames))};
    if SaveConvertedFile
```

```
        MATFileNameShort=sprintf('%s.mat',FileNameNoExt);
        MATFileNameLong=fullfile(FileFolder,MATFileNameShort);
        try
            save(MATFileNameLong,'ConvertedData','ConvertVer','ChanNames')
            fprintf('\n\nConversion complete (saved in ''%s'').\n\n',MATFileNameShort)
        catch exception
            fprintf('\n\nConversion complete (could not save ''%s'').\n\t%s: %s\n\n',MATFi
                exception.message)
        end
    else
        fprintf('\n\nConversion complete.\n\n')
    end
end
ci=channelinfo;
end



function [SegInfo,NumOfSeg]=getSegInfo(fid)
%Count the number of segments.  While doing the count, also include error trapping.

%Find the end of the file
fseek(fid,0,'eof');
eoff=ftell(fid);
frewind(fid);

segCnt=0;
CurrPosn=0;
LeadInByteCount=28; %From the National Instruments web page (http://zone.ni.com/devzone/cc
%the 'Lead In' description on page 2: Counted the bytes shown in the table.
while (ftell(fid) ~= eoff)

    Ttag=fread(fid,1,'uint8');
    Dtag=fread(fid,1,'uint8');
    Stag=fread(fid,1,'uint8');
    mtag=fread(fid,1,'uint8');

    if Ttag==84 && Dtag==68 && Stag==83 && mtag==109
        %Apparently, this sequence of numbers identifies the start of a new segment.

        segCnt=segCnt+1;

        %ToC Field
        ToC=fread(fid,1,'uint32');

        %TDMS format version number
        vernum=fread(fid,1,'uint32');

        %From the National Instruments web page (http://zone.ni.com/devzone/cda/tut/p/id/5
        %description on page 2:
        %The next eight bytes (64-bit unsigned integer) describe the length of the remaini
        %segment minus length of the lead in). If further segments are appended to the fil
        %locate the starting point of the following segment. If an application encountered
        %to a TDMS file (crash, power outage), all bytes of this integer can be 0xFF. This
        %segment in a file.
        nlen=fread(fid,1,'uint64');
        if (nlen>2^63)
```

```
            break;
        else

            segLength=nlen;
        end
        TotalLength=segLength+LeadInByteCount;
        CurrPosn=CurrPosn+TotalLength;

        status=fseek(fid,CurrPosn,'bof');       %Move to the beginning position of the nex
        if (status<0)
            warning('file glitch');
            break;
        end
    end

end

frewind(fid);

CurrPosn=0;
SegInfo.SegStartPosn=zeros(segCnt,1);
SegInfo.MetaStartPosn=zeros(segCnt,1);
SegInfo.DataStartPosn=zeros(segCnt,1);
SegInfo.vernum=zeros(segCnt,1);
SegInfo.DataLength=zeros(segCnt,1);
segCnt=0;
while (ftell(fid) ~= eoff)

    Ttag=fread(fid,1,'uint8');
    Dtag=fread(fid,1,'uint8');
    Stag=fread(fid,1,'uint8');
    mtag=fread(fid,1,'uint8');

    if Ttag==84 && Dtag==68 && Stag==83 && mtag==109
        %Apparently, this sequence of numbers identifies the start of a new segment.

        segCnt=segCnt+1;

        if segCnt==1
            StartPosn=0;
        else
            StartPosn=CurrPosn;
        end

        %ToC Field
        ToC=fread(fid,1,'uint32');
        kTocMetaData=bitget(ToC,2);
        kTocNewObject=bitget(ToC,3);
        kTocRawData=bitget(ToC,4);
        kTocInterleavedData=bitget(ToC,6);
        kTocBigEndian=bitget(ToC,7);

        if kTocInterleavedData
            e=errordlg(sprintf(['Seqment %.0f within ''%s'' has interleaved data which is
                'function (%s.m).'],segCnt,TDMSFileNameShort,mfilename),'Interleaved Data
            fclose(fid);
            uiwait(e)
        end

        if kTocBigEndian
            e=errordlg(sprintf(['Segment %.0f within ''%s'' uses the big-endian data forma
                'with this function (%s.m).'],segCnt,TDMSFileNameShort,mfilename),'Big-Enc
            fclose(fid);
            uiwait(e)
        end

        %TDMS format version number
        vernum=fread(fid,1,'uint32');
        if ~ismember(vernum,[4712,4713])
            e=errordlg(sprintf(['Segment %.0f within ''%s'' used LabView TDMS file format
                'supported with this function (%s.m).'],segCnt,TDMSFileNameShort,vernum,mf
                'TDMS File Format Not Supported');
            fclose(fid);
            uiwait(e)
        end

        %From the National Instruments web page (http://zone.ni.com/devzone/cda/tut/p/id/5
        %description on page 2:
        %The next eight bytes (64-bit unsigned integer) describe the length of the remaini
        %segment minus length of the lead in). If further segments are appended to the fil
        %locate the starting point of the following segment. If an application encountered
        %to a TDMS file (crash, power outage), all bytes of this integer can be 0xFF. This
        %segment in a file.
        segLength=fread(fid,1,'uint64');
        metaLength=fread(fid,1,'uint64');
        if (segLength>2^63)
            fseek(fid,0,'eof');
            flen=ftell(fid);
            segLength=flen-LeadInByteCount-TotalLength;
            TotalLength=segLength+LeadInByteCount;
        else
            TotalLength=segLength+LeadInByteCount;
            CurrPosn=CurrPosn+TotalLength;
            fseek(fid,CurrPosn,'bof');      %Move to the beginning position of the next se
        end


        SegInfo.SegStartPosn(segCnt)=StartPosn;
        SegInfo.MetaStartPosn(segCnt)=StartPosn+LeadInByteCount;
        SegInfo.DataStartPosn(segCnt)=SegInfo.MetaStartPosn(segCnt)+metaLength;
        SegInfo.DataLength(segCnt)=segLength-metaLength;
        SegInfo.vernum(segCnt)=vernum;

    end

end
NumOfSeg=segCnt;
end



function index=getChannelInfo(fid,SegInfo,NumOfSeg)
%Initialize variables for the file conversion
index=struct();
```

```
objOrderList={};
for segCnt=1:NumOfSeg

    fseek(fid,SegInfo.SegStartPosn(segCnt)+4,'bof');

    %Ttag=fread(fid,1,'uint8');
    %Dtag=fread(fid,1,'uint8');
    %Stag=fread(fid,1,'uint8');
    %mtag=fread(fid,1,'uint8');

    %ToC Field
    ToC=fread(fid,1,'uint32');
    kTocMetaData=bitget(ToC,2);
    kTocNewObjectList=bitget(ToC,3);
    kTocRawData=bitget(ToC,4);
    %kTocInterleavedData=bitget(ToC,6);
    %kTocBigEndian=bitget(ToC,7);

    segVersionNum=fread(fid,1,'uint32');                            %TDMS format version r

    segLength=fread(fid,1,'uint64');

    metaLength=fread(fid,1,'uint64');
    offset=0;
    %Process Meta Data
    if (kTocNewObjectList==0) %use the object list from the previous segment
        fnm=fieldnames(index);
        for kk=1:length(fnm)
            ccnt=index.(fnm{kk}).rawdatacount;
            if (ccnt>0)
                if (index.(fnm{kk}).index(ccnt)==segCnt-1)
                    ccnt=ccnt+1;
                    index.(fnm{kk}).rawdatacount=ccnt;
                    index.(fnm{kk}).datastartindex(ccnt)=SegInfo.DataStartPosn(segCnt);
                    index.(fnm{kk}).arrayDim(ccnt)=index.(fnm{kk}).arrayDim(ccnt-1);
                    index.(fnm{kk}).nValues(ccnt)=index.(fnm{kk}).nValues(ccnt-1);
                    index.(fnm{kk}).byteSize(ccnt)=index.(fnm{kk}).byteSize(ccnt-1);
                    index.(fnm{kk}).index(ccnt)=segCnt;
                    index.(fnm{kk}).rawdataoffset(ccnt)=index.(fnm{kk}).rawdataoffset(ccnt
                end
            end
        end
    end

    if kTocMetaData
        numObjInSeg=fread(fid,1,'uint32');
        if (kTocNewObjectList)
            objOrderList=cell(numObjInSeg,1);
        end
        for q=1:numObjInSeg

            obLength=fread(fid,1,'uint32');                                 %Get the length of
            ObjName=convertToText(fread(fid,obLength,'uint8'))';        %Get the objects name

            if strcmp(ObjName,'/')
                long_obname='Root';
            else
                long_obname=ObjName;

                %Delete any apostrophes.  If the first character is a slash (forward or ba
                long_obname(strfind(long_obname,''''))=[];
                if strcmpi(long_obname(1),'/') || strcmpi(long_obname(1),'\')
                    long_obname(1)=[];
                end
            end
            newob=0;
            %Create object's name.  Use a generic field name to avoid issues with strings
            %characters that cannot be used in MATLAB variable names.  The actual channel
            %output structure.
            if exist('ObjNameList','var')
                %Check to see if the object already exists
                NameIndex=find(strcmpi({ObjNameList.LongName},long_obname)==1,1,'first');
                if isempty(NameIndex)
                    newob=1;
                    %It does not exist, so create the generic name field name
                    ObjNameList(end+1).FieldName=sprintf('Object%.0f',numel(ObjNameList)+1
                    ObjNameList(end).LongName=long_obname;
                    NameIndex=numel(ObjNameList);
                end
            else
                %No objects exist, so create the first one using a generic name field name
                ObjNameList.FieldName='Object1';
                ObjNameList.LongName=long_obname;
                NameIndex=1;
                newob=1;
            end
            %Assign the generic field name
            obname=ObjNameList(NameIndex).FieldName;

            %Create the 'index' structure
            if (~isfield(index,obname))
                index.(obname).name=obname;
                index.(obname).long_name=long_obname;
                index.(obname).rawdatacount=0;
                index.(obname).datastartindex=zeros(NumOfSeg,1);
                index.(obname).arrayDim=zeros(NumOfSeg,1);
                index.(obname).nValues=zeros(NumOfSeg,1);
                index.(obname).byteSize=zeros(NumOfSeg,1);
                index.(obname).index=zeros(NumOfSeg,1);
                index.(obname).rawdataoffset=zeros(NumOfSeg,1);
                index.(obname).multiplier=ones(NumOfSeg,1);
                index.(obname).skip=zeros(NumOfSeg,1);
            end
            if (kTocNewObjectList)
                objOrderList{q}=obname;
            else
                if ~ismember(obname,objOrderList)
                    objOrderList{end+1}=obname;
                end
            end
            %Get the raw data Index
            rawdataindex=fread(fid,1,'uint32');

            if rawdataindex==0
```

```
                if segCnt==0
                    e=errordlg(sprintf('Seqment %.0f within ''%s'' has ''rawdataindex'' va
                        TDMSFileNameShort,mfilename),'Incorrect ''rawdataindex''');
                    uiwait(e)
                end
                if kTocRawData
                    if (kTocNewObjectList)
                        ccnt=index.(obname).rawdatacount+1;
                    else
                        ccnt=index.(obname).rawdatacount;
                    end
                    index.(obname).rawdatacount=ccnt;
                    index.(obname).datastartindex(ccnt)=SegInfo.DataStartPosn(segCnt);
                    index.(obname).arrayDim(ccnt)=index.(obname).arrayDim(ccnt-1);
                    index.(obname).nValues(ccnt)=index.(obname).nValues(ccnt-1);
                    index.(obname).byteSize(ccnt)=index.(obname).byteSize(ccnt-1);
                    index.(obname).index(ccnt)=segCnt;

                end
            elseif rawdataindex+1==2^32
                %Objects raw data index matches previous index - no changes.
The root object will always have an
                %'FFFFFFFF' entry
                if strcmpi(index.(obname).long_name,'Root')
                    index.(obname).rawdataindex=0;
                else
                    %Need to account for the case where an object (besides the 'root') is
                    %using previous.
                    if newob
                        index.(obname).rawdataindex=0;
                    else
                        if kTocRawData
                            if (kTocNewObjectList)
                                ccnt=index.(obname).rawdatacount+1;
                            else
                                ccnt=index.(obname).rawdatacount;
                            end
                            index.(obname).rawdatacount=ccnt;
                            index.(obname).datastartindex(ccnt)=SegInfo.DataStartPosn(seg
                            index.(obname).arrayDim(ccnt)=index.(obname).arrayDim(ccnt-1);
                            index.(obname).nValues(ccnt)=index.(obname).nValues(ccnt-1);
                            index.(obname).byteSize(ccnt)=index.(obname).byteSize(ccnt-1);
                            index.(obname).index(ccnt)=segCnt;

                        end
                    end
                end
            else
                %Get new object information
                if (kTocNewObjectList)
                    ccnt=index.(obname).rawdatacount+1;
                else
                    ccnt=index.(obname).rawdatacount;
                    if (ccnt==0)
                        ccnt=1;
                    end
                end
                index.(obname).rawdatacount=ccnt;
                index.(obname).datastartindex(ccnt)=SegInfo.DataStartPosn(segCnt);
                %index(end).lenOfIndexInfo=fread(fid,1,'uint32');

                index.(obname).dataType=fread(fid,1,'uint32');
                if (index.(obname).dataType~=32)
                    index.(obname).datasize=getDataSize(index.(obname).dataType);
                end
                index.(obname).arrayDim(ccnt)=fread(fid,1,'uint32');
                index.(obname).nValues(ccnt)=fread(fid,1,'uint64');
                index.(obname).index(ccnt)=segCnt;
                if index.(obname).dataType==32
                    %Datatype is a string
                    index.(obname).byteSize(ccnt)=fread(fid,1,'uint64');
                else
                    index.(obname).byteSize(ccnt)=0;
                end

            end

            %Get the properties
            numProps=fread(fid,1,'uint32');
            if numProps>0

                if isfield(index.(obname),'PropertyInfo')
                    PropertyInfo=index.(obname).PropertyInfo;
                else
                    clear PropertyInfo
                end
                for p=1:numProps
                    propNameLength=fread(fid,1,'uint32');
                    switch 1
                        case 1
                            PropName=fread(fid,propNameLength,'*uint8')';
                            PropName=native2unicode(PropName,'UTF-8');
                        case 2
                            PropName=fread(fid,propNameLength,'uint8=>char')';
                        otherwise
                    end
                    propsDataType=fread(fid,1,'uint32');

                    %Create property's name.  Use a generic field name to avoid issues wit
                    %characters that cannot be used in MATLAB variable names.
The actual property name is retained for the
                    %final output structure.
                    if exist('PropertyInfo','var')
                        %Check to see if the property already exists for this object.
Need to get the existing 'PropertyInfo'
                        %structure for this object.  The 'PropertyInfo' structure is not r
                        %object in the data file.
                        PropIndex=find(strcmpi({PropertyInfo.Name},PropName));
                        if isempty(PropIndex)
                            %Is does not exist, so create the generic name field name
                            propExists=false;
                            PropIndex=numel(PropertyInfo)+1;
                            propsName=sprintf('Property%.0f',PropIndex);
                            PropertyInfo(PropIndex).Name=PropName;
                            PropertyInfo(PropIndex).FieldName=propsName;
                        else
                            %Assign the generic field name
                            propExists=true;
                            propsName=PropertyInfo(PropIndex).FieldName;
                        end
                    else
                        %No properties exist for this object, so create the first one usin
                        propExists=false;
                        PropIndex=p;
                        propsName=sprintf('Property%.0f',PropIndex);
                        PropertyInfo(PropIndex).Name=PropName;
                        PropertyInfo(PropIndex).FieldName=propsName;
                    end
                    dataExists=isfield(index.(obname),'data');

                    if dataExists
                        %Get number of data samples for the object in this segment
                        nsamps=index.(obname).nsamples+1;
                    else
                        nsamps=0;
                    end

                    if propsDataType==32
                        %String data type
                        PropertyInfo(PropIndex).DataType='String';
                        propsValueLength=fread(fid,1,'uint32');
                        propsValue=convertToText(fread(fid,propsValueLength,'uint8=>char')
                        if propExists
                            if isfield(index.(obname).(propsName),'cnt')
                                cnt=index.(obname).(propsName).cnt+1;
                            else
                                cnt=1;
                            end
                            index.(obname).(propsName).cnt=cnt;
                            index.(obname).(propsName).value{cnt}=propsValue;
                            index.(obname).(propsName).samples(cnt)=nsamps;
                        else
                            if strcmp(index.(obname).long_name,'Root')
                                %Header data
                                index.(obname).(propsName).name=index.(obname).long_name;
                                index.(obname).(propsName).value={propsValue};
                                index.(obname).(propsName).cnt=1;
                            else
                                index.(obname).(propsName).name=PropertyInfo(PropIndex).Na
                                index.(obname).(propsName).datatype=PropertyInfo(PropIndex
                                index.(obname).(propsName).cnt=1;
                                index.(obname).(propsName).value=cell(nsamps,1);        %P
                                index.(obname).(propsName).samples=zeros(nsamps,1); %Pre-a
                                if iscell(propsValue)
                                    index.(obname).(propsName).value(1)=propsValue;
                                else
                                    index.(obname).(propsName).value(1)={propsValue};
                                end
                                index.(obname).(propsName).samples(1)=nsamps;
                            end
                        end
                    else
                        %Numeric data type
                        if propsDataType==68
                            PropertyInfo(PropIndex).DataType='Time';
                            %Timestamp data type
                            tsec=fread(fid,1,'uint64')/2^64+fread(fid,1,'uint64');  %time
                            %R. Seltzer: Not sure why '5/24' (5 hours) is subtracted from
That's how it was
                            %coded in the original function I downloaded from MATLAB Centr
But I found it to be 1 hour too
                            %much.  So, I changed it to '4/24'.
                            %propsValue=tsec/86400+695422-5/24; %/864000 convert to days;
                            propsValue=tsec/86400+695422-4/24;  %/864000 convert to days;
                        else
                            PropertyInfo(PropIndex).DataType='Numeric';
                            matType=LV2MatlabDataType(propsDataType);
                            if strcmp(matType,'Undefined')
                                e=errordlg(sprintf('No MATLAB data type defined for a ''Pr
                                    propsDataType),'Undefined Property Data Type');
                                uiwait(e)
                                fclose(fid);
                                return
                            end
                            if strcmp(matType,'uint8=>char')
                                propsValue=convertToText(fread(fid,1,'uint8'));
                            else
                                propsValue=fread(fid,1,matType);
                            end
                        end
                        if propExists
                            cnt=index.(obname).(propsName).cnt+1;
                            index.(obname).(propsName).cnt=cnt;
                            index.(obname).(propsName).value(cnt)=propsValue;
                            index.(obname).(propsName).samples(cnt)=nsamps;
                        else
                            index.(obname).(propsName).name=PropertyInfo(PropIndex).Name;
                            index.(obname).(propsName).datatype=PropertyInfo(PropIndex).Da
                            index.(obname).(propsName).cnt=1;
                            index.(obname).(propsName).value=NaN(nsamps,1);             %P
                            index.(obname).(propsName).samples=zeros(nsamps,1);     %Pre-a
                            index.(obname).(propsName).value(1)=propsValue;
                            index.(obname).(propsName).samples(1)=nsamps;
                        end
                    end

                end %'end' for the 'Property' loop
                index.(obname).PropertyInfo=PropertyInfo;

            end

        end %'end' for the 'Objects' loop
    end

    %Move the offset calculation to the end to account for added channels and other optimi
    if (kTocRawData) %only do the check if there was raw data in the segment
        offset=0;
        for kk=1:numel(objOrderList)
```

```
            obname=objOrderList{kk};
            ccnt=index.(obname).rawdatacount;
            if (ccnt>0)
                index.(obname).rawdataoffset(ccnt)=offset;
                if index.(obname).dataType==32
                    %Datatype is a string
                    offset=offset+index.(obname).byteSize(ccnt);
                else
                    offset=offset+index.(obname).nValues(ccnt)*index.(obname).datasize;
                end
            end
        end

        %Don't know why but sometimes the 'nValues' parameter is sometimes incorrect. Eith
        %someone who wrote the drivers was lazy. Seems to happen with waveform files. Chec
        %offset value matches the segment's size.  If it doesn't, then check if the size i
If
        %it is, then multiply all appropriate parameters in the index structure.
If not, then generate a warning.
        if (offset~=SegInfo.DataLength(segCnt))
            %if (mod(SegInfo.DataLength(segCnt),offset)==0)
            multiplier=floor(SegInfo.DataLength(segCnt)/offset);
            for kk=1:numel(objOrderList)
                obname=objOrderList{kk};
                ccnt=index.(obname).rawdatacount;
                if (ccnt>0)&&(index.(obname).index(ccnt)==segCnt);
                    index.(obname).multiplier(ccnt)=multiplier;
                    if index.(obname).dataType==32
                        %Datatype is a string
                        index.(obname).skip(ccnt)=offset-index.(obname).byteSize(ccnt);
                    else
                        index.(obname).skip(ccnt)=offset-index.(obname).nValues(ccnt)*inde
                    end
                end
            end
            % else
            %    warning('segment %d error: offset=%d, dataLength=%d\n',segCnt,offset,SegI
            % end
        end
    end

end
%clean up the index if it has to much data
fnm=fieldnames(index);
for kk=1:numel(fnm)
    ccnt=index.(fnm{kk}).rawdatacount+1;

    index.(fnm{kk}).datastartindex(ccnt:end)=[];
    index.(fnm{kk}).arrayDim(ccnt:end)=[];
    index.(fnm{kk}).nValues(ccnt:end)=[];
    index.(fnm{kk}).byteSize(ccnt:end)=[];
    index.(fnm{kk}).index(ccnt:end)=[];
    index.(fnm{kk}).rawdataoffset(ccnt:end)=[];
    index.(fnm{kk}).multiplier(ccnt:end)=[];
    index.(fnm{kk}).skip(ccnt:end)=[];

end
```

```
end



function ob=getData(fid,index)
ob=[];
fnm=fieldnames(index);
for kk=1:length(fnm)
    id=index.(fnm{kk});
    nsamples=sum(id.nValues.*id.multiplier);
    if id.rawdatacount>0
        cname=id.name;
        ob.(cname).nsamples=0;
        if id.dataType==32
            ob.(cname).data=cell(nsamples,1);
        else
            ob.(cname).data=zeros(nsamples,1);
        end
        for rr=1:id.rawdatacount
            %Loop through each of the groups/channels and read the raw data
            fseek(fid,id.datastartindex(rr)+id.rawdataoffset(rr),'bof');


            nvals=id.nValues(rr);

            if nvals>0

                switch id.dataType

                    case 32     %String
                        %From the National Instruments web page (http://zone.ni.com/devzon
                        %'Raw Data' description on page 4:
                        %String type channels are preprocessed for fast random access. All
                        %contiguous piece of memory. The offset of the first character of
                        %of memory is stored to an array of unsigned 32-bit integers. This
                        %first, followed by the concatenated string values. This layout al
                        %any string value from anywhere in the file by repositioning the f
                        %and without reading any data that is not needed by the client.
                        data=cell(1,nvals*id.multiplier(rr));   %Pre-allocation
                        for mm=1:id.multiplier(rr)
                            StrOffsetArray=fread(fid,nvals,'uint32');
                            for dcnt=1:nvals
                                if dcnt==1
                                    StrLength=StrOffsetArray(dcnt);
                                else
                                    StrLength=StrOffsetArray(dcnt)-StrOffsetArray(dcnt-1);
                                end
                                data{1,dcnt+(mm-1)*nvals}=char(convertToText(fread(fid,Str
                            end
                            if (id.multiplier(rr)>1)&&(id.skip(rr)>0)
                                fseek(fid,id.skip(rr),'cof');
                            end
                        end
                        cnt=nvals*id.multiplier(rr);

                    case 68     %Timestamp
                        %data=NaN(1,nvals); %Pre-allocation
                        data=NaN(1,nvals*id.multiplier(rr));
                        for mm=1:id.multiplier(rr)
                            dn=fread(fid,2*nvals,'uint64');
                            tsec=dn(1:2:end)/2^64+dn(2:2:end);
                            data((mm-1)*nvals+1:(mm)*nvals)=tsec/86400+695422-4/24;
                            fseek(fid,id.skip(rr),'cof');
                        end
                        %{
                        for dcnt=1:nvals
                            tsec=fread(fid,1,'uint64')/2^64+fread(fid,1,'uint64');
%time since Jan-1-1904 in seconds
                            %R. Seltzer: Not sure why '5/24' (5 hours) is subtracted from
That's how it was
                            %coded in the original function I downloaded from MATLAB Centr
But I found it to be 1 hour too
                            %much.  So, I changed it to '4/24'.
                            data(1,dcnt)=tsec/86400+695422-5/24;    %/864000 convert to da
                            data(1,dcnt)=tsec/86400+695422-4/24;    %/864000 convert to da
                        end
                        %}
                        cnt=nvals*id.multiplier(rr);

                    otherwise   %Numeric
                        matType=LV2MatlabDataType(id.dataType);
                        if strcmp(matType,'Undefined')
                            e=errordlg(sprintf('No MATLAB data type defined for a ''Raw Da
                                id.dataType),'Undefined Raw Data Type');
                            uiwait(e)
                            fclose(fid);
                            return
                        end
                        if (id.skip(rr)>0)
                            ntype=sprintf('%d*%s',nvals,matType);
                            if strcmp(matType,'uint8=>char')
                                [data,cnt]=fread(fid,nvals*id.multiplier(rr),ntype,id.skip
                                data=convertToText(data);
                            else
                                [data,cnt]=fread(fid,nvals*id.multiplier(rr),ntype,id.skip
                            end
                        else
                            [data,cnt]=fread(fid,nvals*id.multiplier(rr),matType);
                        end
                end

                if isfield(ob.(cname),'nsamples')
                    ssamples=ob.(cname).nsamples;
                else
                    ssamples=0;
                end
                if (cnt>0)
                    ob.(cname).data(ssamples+1:ssamples+cnt,1)=data;
                    ob.(cname).nsamples=ssamples+cnt;
                end
            end
        end

    end
```

```
end
end



function [DataStructure,GroupNames]=postProcess(ob,index)
%Re-organize the 'ob' structure into a more user friendly format for output.


DataStructure.Root=[];
DataStructure.MeasuredData.Name=[];
DataStructure.MeasuredData.Data=[];

obFieldNames=fieldnames(index);

cntData=1;

for i=1:numel(obFieldNames)

    cname=obFieldNames{i};

    if strcmp(index.(cname).long_name,'Root')

        DataStructure.Root.Name=index.(cname).long_name;

        %Assign all the 'Property' values
        if isfield(index.(cname),'PropertyInfo')
            for p=1:numel(index.(cname).PropertyInfo)
                cfield=index.(cname).PropertyInfo(p).FieldName;
                if isfield(index.(cname).(cfield),'datatype')
                    DataType=index.(cname).(cfield).datatype;
                else
                    %ASSUME a 'string' data type
                    DataType='String';
                end
                DataStructure.Root.Property(p).Name=index.(cname).PropertyInfo(p).Name;

                switch DataType
                    case 'String'
                        if iscell(index.(cname).(cfield).value)
                            Value=index.(cname).(cfield).value';
                        else
                            Value=cellstr(index.(cname).(cfield).value);
                        end

                    case 'Time'
                        clear Value
                        if index.(cname).(cfield).cnt==1
                            if iscell(index.(cname).(cfield).value)
                                Value=datestr(cell2mat(index.(cname).(cfield).value),'dd-m
                            else
                                Value=datestr(index.(cname).(cfield).value,'dd-mmm-yyyy HH
                            end
                        else
                            Value=cell(index.(cname).(cfield).cnt,1);
                            for c=1:index.(cname).(cfield).cnt
                                if iscell(index.(cname).(cfield).value)
                                    Value(c)={datestr(cell2mat(index.(cname).(cfield).valu
                                else
                                    Value(c)={datestr(index.(cname).(cfield).value,'dd-mmm
                                end
                            end
                        end

                    case 'Numeric'
                        if isfield(index.(cname).(cfield),'cnt')
                            Value=NaN(index.(cname).(cfield).cnt,1);
                        else
                            if iscell(index.(cname).(cfield).value)
                                Value=NaN(numel(cell2mat(index.(cname).(cfield).value)),1)
                            else
                                Value=NaN(numel(index.(cname).(cfield).value),1);
                            end
                        end
                        for c=1:numel(Value)
                            if iscell(index.(cname).(cfield).value)
                                Value(c)=index.(cname).(cfield).value{c};
                            else
                                Value(c)=index.(cname).(cfield).value(c);
                            end
                        end
                    otherwise
                        e=errordlg(sprintf(['No format defined for Data Type ''%s'' in the
                            'within %s.m.'],index.(cname).(cfield).datatype,mfilename),'Un
                        uiwait(e)
                        return
                end
                if isempty(Value)
                    DataStructure.Root.Property(p).Value=[];
                else
                    DataStructure.Root.Property(p).Value=Value;
                end
            end
        end


    end

    DataStructure.MeasuredData(cntData).Name=index.(cname).long_name;
    %Should only need the 'ShortName' for debugging the function
    %DataStructure.MeasuredData(cntData).ShortName=cname;
    if (isfield(ob,cname))
        if isfield(ob.(cname),'data')
            DataStructure.MeasuredData(cntData).Data=ob.(cname).data;
            %The following field is redundant because the information can be obtained from
            DataStructure.MeasuredData(cntData).Total_Samples=ob.(cname).nsamples;
        else
            DataStructure.MeasuredData(cntData).Data=[];
            DataStructure.MeasuredData(cntData).Total_Samples=0;
        end
    else
        DataStructure.MeasuredData(cntData).Data=[];
```

```
        DataStructure.MeasuredData(cntData).Total_Samples=0;
    end

    %Assign all the 'Property' values
    if isfield(index.(cname),'PropertyInfo')
        for p=1:numel(index.(cname).PropertyInfo)
            cfield=index.(cname).PropertyInfo(p).FieldName;
            DataStructure.MeasuredData(cntData).Property(p).Name=index.(cname).(cfield).na

            if strcmpi(DataStructure.MeasuredData(cntData).Property(p).Name,'Root')
                Value=index.(cname).(cfield).value;
            else

                switch index.(cname).(cfield).datatype
                    case 'String'
                        clear Value
                        if index.(cname).(cfield).cnt==1
                            if iscell(index.(cname).(cfield).value)
                                Value=char(index.(cname).(cfield).value);
                            else
                                Value=index.(cname).(cfield).value;
                            end
                        else
                            Value=cell(index.(cname).(cfield).cnt,1);
                            for c=1:index.(cname).(cfield).cnt
                                if iscell(index.(cname).(cfield).value)
                                    Value(c)=index.(cname).(cfield).value;
                                else
                                    Value(c)={index.(cname).(cfield).value};
                                end
                            end
                        end

                    case 'Time'
                        clear Value
                        if index.(cname).(cfield).cnt==1
                            if iscell(index.(cname).(cfield).value)
                                Value=datestr(cell2mat(index.(cname).(cfield).value),'dd-
                            else
                                Value=datestr(index.(cname).(cfield).value,'dd-mmm-yyyy H
                            end
                        else
                            Value=cell(index.(cname).(cfield).cnt,1);
                            for c=1:index.(cname).(cfield).cnt
                                if iscell(index.(cname).(cfield).value)
                                    Value(c)={datestr(cell2mat(index.(cname).(cfield).valu
                                else
                                    Value(c)={datestr(index.(cname).(cfield).value,'dd-mmm
                                end
                            end
                        end

                    case 'Numeric'
                        if isfield(index.(cname).(cfield),'cnt')
                            Value=NaN(index.(cname).(cfield).cnt,1);
                        else
                            if iscell(index.(cname).(cfield).value)
```

```
                                Value=NaN(numel(cell2mat(index.(cname).(cfield).value)),1)
                            else
                                Value=NaN(numel(index.(cname).(cfield).value),1);
                            end
                        end
                        for c=1:numel(Value)
                            if iscell(index.(cname).(cfield).value)
                                Value(c)=index.(cname).(cfield).value{c};
                            else
                                Value(c)=index.(cname).(cfield).value(c);
                            end
                        end

                    otherwise
                        e=errordlg(sprintf(['No format defined for Data Type ''%s'' in the
                            'within %s.m.'],index.(cname).(cfield).datatype,mfilename),'Ur
                        uiwait(e)
                        return
                end
            end
            if isempty(Value)
                DataStructure.MeasuredData(cntData).Property(p).Value=[];
            else
                DataStructure.MeasuredData(cntData).Property(p).Value=Value;
            end
        end
    else
        DataStructure.MeasuredData(cntData).Property=[];
    end

    cntData = cntData + 1;
end %'end' for the 'groups/channels' loop

%Extract the Group names
GroupIndices=false(numel(DataStructure.MeasuredData),1);
for d=1:numel(DataStructure.MeasuredData)

    if ~strcmpi(DataStructure.MeasuredData(d).Name,'Root')
        if (DataStructure.MeasuredData(d).Total_Samples==0)
            fs=strfind(DataStructure.MeasuredData(d).Name,'/');
            if (isempty(fs))
                GroupIndices(d)=true;
            end
        end
    end

end
if any(GroupIndices)
    GroupNames=sort({DataStructure.MeasuredData(GroupIndices).Name})';
else
    GroupNames=[];
end

end

function sz=getDataSize(LVType)
switch(LVType)
```

```
    case 0
        sz=0;
    case {1,5,33}
        sz=1;
    case 68
        sz=16;
    case {8,10}
        sz=8;
    case {3,7,9}
        sz=4;
    case {2,6}
        sz=2;
    case 32
        e=errordlg('Do not call the getDataSize function for strings.  Their size is writt
        uiwait(e)
        sz=NaN;
    case 11
        sz=10;
end
end




function matType=LV2MatlabDataType(LVType)
%Cross Refernce Labview TDMS Data type to MATLAB
switch LVType
    case 0   %tdsTypeVoid
        matType='';
    case 1   %tdsTypeI8
        matType='int8';
    case 2   %tdsTypeI16
        matType='int16';
    case 3   %tdsTypeI32
        matType='int32';
    case 4   %tdsTypeI64
        matType='int64';
    case 5   %tdsTypeU8
        matType='uint8';
    case 6   %tdsTypeU16
        matType='uint16';
    case 7   %tdsTypeU32
        matType='uint32';
    case 8   %tdsTypeU64
        matType='uint64';
    case 9  %tdsTypeSingleFloat
        matType='single';
    case 10  %tdsTypeDoubleFloat
        matType='double';
    case 11  %tdsTypeExtendedFloat
        matType='10*char';
    case 25 %tdsTypeSingleFloat with units
        matType='Undefined';
    case 26 %tdsTypeDoubleFloat with units
        matType='Undefined';
    case 27 %tdsTypeextendedFloat with units
        matType='Undefined';
    case 32  %tdsTypeString
        matType='uint8=>char';
    case 33  %tdsTypeBoolean
        matType='bit1';
    case 68  %tdsTypeTimeStamp
        matType='2*int64';
    otherwise
        matType='Undefined';
end

end


function text=convertToText(bytes)
%Convert numeric bytes to the character encoding localy set in MATLAB (TDMS uses UTF-8)

text=native2unicode(bytes,'UTF-8');
end
```

Appendix C

Description of Master Thesis

Norwegian University of Science and Technology

Faculty of Engineering
Department of Energy and Process Engineering

EPT-M-2020

MASTER WORK

for
student Carsten Cleve-Hansen

Spring 2020

”Splitter plate’s effect on trailing edge vortex shedding and fluid structure interaction”
“Effekten av splitter plater på en hydrofoils virvelavløsning og fluid struktur interaksjon»

Background and objective

In tests on cylinders, splitter plates have been shown to dampen vortex induced vibrations (VIVs). Such VIVs can be observed in several components in hydropower plants, and effective ways to mitigate VIVs of such components are interesting from an industrial perspective.

Work has been performed to prepare a test rig at the Waterpower laboratory for experimental investigations of the effect of splitter plates. The work of the candidate is to finalize this work and to execute a measurement campaign including particle image velocimetry (PIV) measurement technique to quantify the effect of splitter plates.

If for some reason the experimental investigation can’t be performed, CFD investigation of the setup should be performed if time allows.

The following tasks are to be considered:

1. A literature review of passive flow controllers for mitigation of vortex induced vibrations with focus on splitter plates.
2. Decide on a measurement campaign on the test rig using splitter plates at the waterpower laboratory
3. Complete the installation, instrumentation and calibration needed to perform the measurement campaign
4. Execute the measurement campaign
5. If the student will go to Nepal for an excursion, earlier and further work will be presented as a publication and presented at the conference; 10th *International symposium on Current Research in Hydropower Technologies (CRHT-IX)* at Kathmandu University

-- “ --

The master work comprises 30 ECTS credits.

The work shall be edited as a scientific report, including a table of contents, a summary in Norwegian, conclusion, an index of literature etc. When writing the report, the candidate must emphasise a clearly arranged and well-written text. To facilitate the reading of the report, it is important that references for corresponding text, tables and figures are clearly stated both places.
By the evaluation of the work the following will be greatly emphasised: The results should be thoroughly treated, presented in clearly arranged tables and/or graphics and discussed in detail.

The candidate is responsible for keeping contact with the subject teacher and teaching supervisors.

Risk assessment of the candidate's work shall be carried out according to the department's procedures. The risk assessment must be documented and included as part of the final report. Events related to the candidate's work adversely affecting the health, safety or security, must be documented and included as part of the final report. If the documentation on risk assessment represents a large number of pages, the full version is to be submitted electronically to the supervisor and an excerpt is included in the report.

According to "Utfyllende regler til studieforskriften for teknologistudiet/sivilingeniørstudiet ved NTNU" § 20, the Department of Energy and Process Engineering reserves all rights to use the results and data for lectures, research and future publications.

Submission deadline: To be found in Inspera.

☒ Work to be done in lab (Water power lab, Fluids engineering lab, Thermal engineering lab)
□ Field work

Department for Energy and Process Engineering 4/1 2020

Pål-Tore Storli
Supervisor

Co-Supervisor(s): Kristian Sagmo

Appendix D

Risk Assessment Report

NTNU SINTEF

Risk Assessment Report

HiFrancis Blade Cascade

Project name	PIV measurements on blade cascade for HiFrancis project
Apparatus	Double-Cavity High speed laser, 2*10mJ, 1000Hz, 527nm
Unit	EPT
Equipment manager	Bård Brandåstrø
Project leader	Pål Tore Storli
HSE coordinator	Morten Grønli
HSE responsible (linjeleder)	Therese Løvås
Location	Waterpower Laboratory
Room number	11
Risk assessment performed by	Magne Tveit Bolstad

Approval:

Apparatur kort (UNIT CARD) valid for:	12 months
Forsøk pågår kort (EXPERIMENT IN PROGRESS) valid for:	12 months

Rolle	Navn	Dato	Signatur
Project leader	Pål-Tore Storli	24/8 2017	
HSE coordinator	Morten Grønli		
HMS responsible (linjeleder)	Therese Løvås		

TABLE OF CONTENTS

1 INTRODUCTION

PIV measurements will be performed on a Site Acceptance Test-rig in accordance with the new PIV equipment at the waterpower laboratory. Measurements will be performed in May-June 2017.

2 ORGANISATION

Role	
Project leader	Pål Tore Storli
Equipment manager	Bård Brandåstrø
Room manager	Bård Brandåstrø
HSE coordinator	Morten Grønli
HSE responsible (linjeleder):	Therese Løvås

3 RISK MANAGEMENT IN THE PROJECT

Hovedaktiviteter risikostyring	Nødvendige tiltak, dokumentasjon	DATE
Prosjekt initiering Project initiation	Prosjekt initiering mal	X
Veiledningsmøte Guidance Meeting	Skjema for Veiledningsmøte med pre-risikovurdering	x
Innledende risikovurdering Initial Assessment	Fareidentifikasjon – HAZID Skjema grovanalyse	x
Vurdering av teknisk sikkerhet Evaluation of technical security	Prosess-HAZOP Tekniske dokumentasjoner	x
Vurdering av operasjonell sikkerhet Evaluation of operational safety	Prosedyre-HAZOP Opplæringsplan for operatører	x
Sluttvurdering, kvalitetssikring Final assessment, quality assurance	Uavhengig kontroll Utstedelse av apparaturkort Utstedelse av forsøk pågår kort	x

4 DESCRIPTIONS OF EXPERIMENTAL SETUP

- *Drawings and photos describing the setup.*
- *Process and Instrumentation Diagram (PID) with list of components*
- *Location of the operator, gas bottles, shutdown valves for water / air.*

5 EVACUATION FROM THE EXPERIMENTAL AREA

Evacuate at signal from the alarm system or local gas alarms with its own local alert with sound and light outside the room in question, see 6.2

Evacuation from the rigging area takes place through the marked emergency exits to the assembly point, (corner of Old Chemistry Kjelhuset or parking 1a-b.)

Action on rig before evacuation:
Describe in which condition the rig should be left in case of evacuation (emergency shutdown procedure, water, gas, electric supply, etc.)

6 WARNING

6.1 Before experiments

Send an e-mail with information about the planned experiment to:
iept-experiments@ivt.ntnu.no

The e-mail must include the following information:

- Name of responsible person:
- Experimental setup/rig:
- Start Experiments: (date and time)
- Stop Experiments: (date and time)

You must get the approval back from the laboratory management before start up. All running experiments are notified in the activity calendar for the lab to be sure they are coordinated with other activity.

6.2 Non-conformance

FIRE
If you are NOT able to extinguish the fire, activate the nearest fire alarm and evacuate area. Be then available for fire brigade and building caretaker to detect fire place.
If possible, notify:

NTNU	SINTEF
Morten Grønli, Mob: 918 97 515	Harald Mæhlum, Mob: 930 14 986
Therese Løvås, Mob: 91897007	Anne Karin T. Hemmingsen Mob: 930 19 669
NTNU – SINTEF Beredskapstelefon	800 80 388

GAS ALARM

If a gas alarm occurs, close gas bottles immediately and ventilate the area. If the level of the gas concentration does not decrease within a reasonable time, activate the fire alarm and evacuate the lab. Designated personnel or fire department checks the leak to determine whether it is possible to seal the leak and ventilate the area in a responsible manner.

PERSONAL INJURY

- First aid kit in the fire / first aid stations
- Shout for help
- Start life-saving first aid
- CALL 113 if there is any doubt whether there is a serious injury

OTHER NON-CONFORMANCE (AVVIK)

NTNU:
You will find the reporting form for non-conformance on:
https://innsida.ntnu.no/wiki/-/wiki/Norsk/Melde+avvik

SINTEF:
Synergi

7 ASSESSMENT OF TECHNICAL SAFETY

7.1 HAZOP

See Chapter 13 "Guide to the report template".

The experiment set up is divided into the following nodes:

Node 1	Blade cascade
Node 2	Laser class IV
Node 3	Tent/enclosure

Attachments, Form: Hazop_mal

Conclusion

Node 1:

- **Pressure is supervised during operation**

Node 2:

- **Radiation area shielded**
- **Appropriate signalling and lights in place, light active during operation**

Node 3:

- **Appropriate signalling and lights in place, light active during operation**

7.2 Flammable, reactive and pressurized substances and gas

See Chapter 13 "Guide to the report template".

NO	

7.3 Pressurized equipment

See Chapter 13 "Guide to the report template".

YES	

7.4 Effects on the environment (emissions, noise, temperature, vibration, smell)

See Chapter 13 "Guide to the report template".

NO	

7.5 Radiation

See Chapter 13 "Guide to the report template".

YES	Radiation Sources need to have an own risk assessment

Attachments: Radiation risk assessment
Conclusion:

7.6 Chemicals

See Chapter 13 "Guide to the report template".

NO	

7.7 Electricity safety (deviations from the norms/standards)

See Chapter 13 "Guide to the report template".

NO	

8 ASSESSMENT OF OPERATIONAL SAFETY

Ensure that the procedures cover all identified risk factors that must be taken care of. Ensure that the operators and technical performance have sufficient expertise.

8.1 Procedure HAZOP

See Chapter 13 "Guide to the report template".
The method is a procedure to identify causes and sources of danger to operational problems.

Attachments:: HAZOP_MAL_Prosedyre

8.2 Operation procedure and emergency shutdown procedure

See Chapter 13 "Guide to the report template".
The operating procedure is a checklist that must be filled out for each experiment.

Emergency procedure should attempt to set the experiment set up in a harmless state by unforeseen events.

Attachments: Procedure for running experiments
Emergency shutdown procedure:

8.3 Training of operators

8.4 Technical modifications

8.5 Personal protective equipment

- *It is mandatory use of appropriate eye protection in the rig zone*

8.6 General Safety

- *The area around the staging attempts shielded.*
- *Operator has to be present during experiments.*

8.7 Safety equipment

- *Warning signs and worning light, see the Regulations on Safety signs and signalling in the workplace*

8.8 Special predations

9 QUANTIFYING OF RISK - RISK MATRIX

See Chapter 13 "Guide to the report template".

The risk matrix will provide visualization and an overview of activity risks so that management and users get the most complete picture of risk factors.

IDnr	**Aktivitet-hendelse**	**Frekv-Sans**	**Kons**	**RV**
1	*Unintentional rarefaction/reflection of laser beam*	1	A	1A
2	*People without protective goggles entering radiation area*	1	C	1C
3	*Damaging lab equipment*	2	B	2B
4	*Water-damage on lab equipment*	3	B	3C
5	*Structural failure due to high pressure*	1	C	1C

Conclusion: There is little remaining risk. The most prominent risk is that people unintentionally wander into the radiation area without protective goggles, but proper signalling and blocking should prevent this. The risk is therefore acceptable.

10 REGULATIONS AND GUIDELINES

Se http://www.arbeidstilsynet.no/regelverk/index.html

- Lov om tilsyn med elektriske anlegg og elektrisk utstyr (1929)
- Arbeidsmiljøloven
- Forskrift om systematisk helse-, miljø- og sikkerhetsarbeid (HMS Internkontrollforskrift)
- Forskrift om sikkerhet ved arbeid og drift av elektriske anlegg (FSE 2006)
- Forskrift om elektriske forsyningsanlegg (FEF 2006)
- Forskrift om utstyr og sikkerhetssystem til bruk i eksplosjonsfarlig område NEK 420
- Forskrift om håndtering av brannfarlig, reaksjonsfarlig og trykksatt stoff samt utstyr og anlegg som benyttes ved håndteringen
- Forskrift om Håndtering av eksplosjonsfarlig stoff
- Forskrift om bruk av arbeidsutstyr.
- Forskrift om Arbeidsplasser og arbeidslokaler
- Forskrift om Bruk av personlig verneutstyr på arbeidsplassen
- Forskrift om Helse og sikkerhet i eksplosjonsfarlige atmosfærer
- Forskrift om Høytrykksspyling
- Forskrift om Maskiner
- Forskrift om Sikkerhetsskilting og signalgivning på arbeidsplassen
- Forskrift om Stillaser, stiger og arbeid på tak m.m.
- Forskrift om Sveising, termisk skjæring, termisk sprøyting, kullbuemeisling, lodding og sliping (varmt arbeid)
- Forskrift om Tekniske innretninger
- Forskrift om Tungt og ensformig arbeid
- Forskrift om Vern mot eksponering for kjemikalier på arbeidsplassen (Kjemikalieforskriften)
- Forskrift om Vern mot kunstig optisk stråling på arbeidsplassen
- Forskrift om Vern mot mekaniske vibrasjoner
- Forskrift om Vern mot støy på arbeidsplassen

Veiledninger fra arbeidstilsynet
se: http://www.arbeidstilsynet.no/regelverk/veiledninger.html

11 DOCUMENTATION

- Tegninger, foto, beskrivelser av forsøksoppsetningen
- Hazop_mal
- Sikker bruk av LASERE, retningslinje
- HAZOP_MAL_Prosedyre
- Forsøksprosedyre
- Opplæringsplan for operatører
- Skjema for sikker jobb analyse, (SJA)
- Apparaturkortet
- Forsøk pågår kort

12 GUIDANCE TO RISK ASSESSMENT TEMPLATE

Chapter 7 Assessment of technical safety.

Ensure that the design of the experiment set up is optimized in terms of technical safety. Identifying risk factors related to the selected design, and possibly to initiate re-design to ensure that risk is eliminated as much as possible through technical security.
This should describe what the experimental setup actually are able to manage and acceptance for emission.

7.1 HAZOP

The experimental set up is divided into nodes (eg motor unit, pump unit, cooling unit.). By using guidewords to identify causes, consequences and safeguards, recommendations and conclusions are made according to if necessary safety is obtained. When actions are performed the HAZOP is completed.
(e.g. "No flow", cause: the pipe is deformed, consequence: pump runs hot, precaution: measurement of flow with a link to the emergency or if the consequence is not critical used manual monitoring and are written into the operational procedure.)

7.2 Flammable, reactive and pressurized substances and gas.

According to the Regulations for handling of flammable, reactive and pressurized substances and equipment and facilities used for this:

Flammable material: Solid, liquid or gaseous substance, preparation, and substance with occurrence or combination of these conditions, by its flash point, contact with other substances, pressure, temperature or other chemical properties represent a danger of fire.

Reactive substances: Solid, liquid, or gaseous substances, preparations and substances that occur in combinations of these conditions, which on contact with water, by its pressure, temperature or chemical conditions, represents a potentially dangerous reaction, explosion or release of hazardous gas, steam, dust or fog.

Pressurized : Other solid, liquid or gaseous substance or mixes having fire or hazardous material response, when under pressure, and thus may represent a risk of uncontrolled emissions

Further criteria for the classification of flammable, reactive and pressurized substances are set out in Annex 1 of the Guide to the Regulations "Flammable, reactive and pressurized substances"
http://www.dsb.no/Global/Publikasjoner/2009/Veiledning/Generell%20veiledning.pdf
http://www.dsb.no/Global/Publikasjoner/2010/Tema/Temaveiledning_bruk_av_farlig_stoff_Del_1.pdf

Experiment setup area should be reviewed with respect to the assessment of Ex zone

- Zone 0: Always explosive atmosphere, such as inside the tank with gas, flammable liquid.
- Zone 1: Primary zone, sometimes explosive atmosphere such as a complete drain point
- Zone 2: secondary discharge could cause an explosive atmosphere by accident, such as flanges, valves and connection points

7.4 Effects on the environment

With pollution means: bringing solids, liquid or gas to air, water or ground, noise and vibrations, influence of temperature that may cause damage or inconvenience effect to the environment.
Regulations: http://www.lovdata.no/all/hl-19810313-006.html#6
NTNU guidance to handling of waste:http://www.ntnu.no/hms/retningslinjer/HMSR18B.pdf

7.5 Radiation

Definition of radiation

Ionizing radiation: Electromagnetic radiation (in radiation issues with wawelength <100 nm) or rapid atomic particles (e.g. alpha and beta particles) with the ability to stream ionized atoms or molecules.

Non ionizing radiation: Electromagnetic radiation (wavelength >100 nm), og ultrasound[1] with small or no capability to ionize.

Radiation sources: All ionizing and powerful non-ionizing radiation sources.

Ionizing radiation sources: Sources giving ionizing radiation e.g. all types of radiation sources, x-ray, and electron microscopes.

Powerful non ionizing radiation sources: Sources giving powerful non ionizing radiation which can harm health and/or environment, e.g. class 3B and 4. MR[2] systems, UVC[3] sources, powerful IR sources[4].

[1] Ultrasound is an acoustic radiation ("sound") over the audible frequency range (> 20 kHz). In radiation protection regulations are referred to ultrasound with electromagnetic non-ionizing radiation.
[2] MR (e.g. NMR) - nuclear magnetic resonance method that is used to "depict" inner structures of different materials.
[3] UVC is electromagnetic radiation in the wavelength range 100-280 nm.
[4] IR is electromagnetic radiation in the wavelength range 700 nm - 1 mm.

For each laser there should be an information binder (HMSRV3404B) which shall include:

- General information
- Name of the instrument manager, deputy, and local radiation protection coordinator
- Key data on the apparatus
- Instrument-specific documentation
- References to (or copies of) data sheets, radiation protection regulations, etc.
- Assessments of risk factors
- Instructions for users
- Instructions for practical use, startup, operation, shutdown, safety precautions, logging, locking, or use of radiation sensor, etc.
- Emergency procedures
- See NTNU for laser: http://www.ntnu.no/hms/retningslinjer/HMSR34B.pdf

7.6 The use and handling of chemicals.

In the meaning chemicals, a element that can pose a danger to employee safety and health
See: http://www.lovdata.no/cgi-wift/ldles?doc=/sf/sf/sf-20010430-0443.html
Safety datasheet is to be kept in the HSE binder for the experiment set up and registered in the database for chemicals.

Chapter 8 Assessment of operational procedures.

Ensures that established procedures meet all identified risk factors that must be taken care of through operational barriers and that the operators and technical performance have sufficient expertise.

8.1 Procedure Hazop

Procedural HAZOP is a systematic review of the current procedure, using the fixed HAZOP methodology and defined guidewords. The procedure is broken into individual operations (nodes) and analyzed using guidewords to identify possible nonconformity, confusion or sources of inadequate performance and failure.

8.2 Procedure for running experiments and emergency shutdown.

Have to be prepared for all experiment setups.

The operating procedure has to describe stepwise preparation, startup, during and ending conditions of an experiment. The procedure should describe the assumptions and conditions for starting, operating parameters with the deviation allowed before aborting the experiment and the condition of the rig to be abandoned.

Emergency procedure describes how an emergency shutdown have to be done, (conducted by the uninitiated),

what happens when emergency shutdown, is activated. (electricity / gas supply) and which events will activate the emergency shutdown (fire, leakage).

Chapter 9 Quantifying of RISK

Quantifying of the residue hazards, Risk matrix

To illustrate the overall risk, compared to the risk assessment, each activity is plotted with values for the probability and consequence into the matrix. Use task IDnr.

Example: If activity IDnr. 1 has been given a probability 3 and D for consequence the risk value become D3, red. This is done for all activities giving them risk values.

In the matrix are different degrees of risk highlighted in red, yellow or green. When an activity ends up on a red risk (= unacceptable risk), risk reducing action has to be taken

CONSEQUENSES	Catastrophic	**E1**	**E2**	**E3**	**E4**	**E5**
	Major	**D1**	**D2**	**D3**	**D4**	**D5**
	Moderate	**C1**	**C2**	**C3**	**C4**	**C5**
	Minor	**B1**	**B2**	**B3**	**B4**	**B5**
	Insignificant	**A1**	**A2**	**A3**	**A4**	**A5**
		Rare	Unlikely	Possible	Likely	Almost
		PROBABILITY				

Table 8. Risk's Matrix

Table 9. The principle of the acceptance criterion. Explanation of the colors used in the matrix

COLOUR		DESCRIPTION
Red	🟥	Unacceptable risk Action has to be taken to reduce risk
Yellow	🟨	Assessment area. Actions has to be considered
Green	🟩	Acceptable risk. Action can be taken based on other criteria

NTNU SINTEF

Attachment to Risk Assessment report

HiFrancis Blade Cascade PIV LDY300

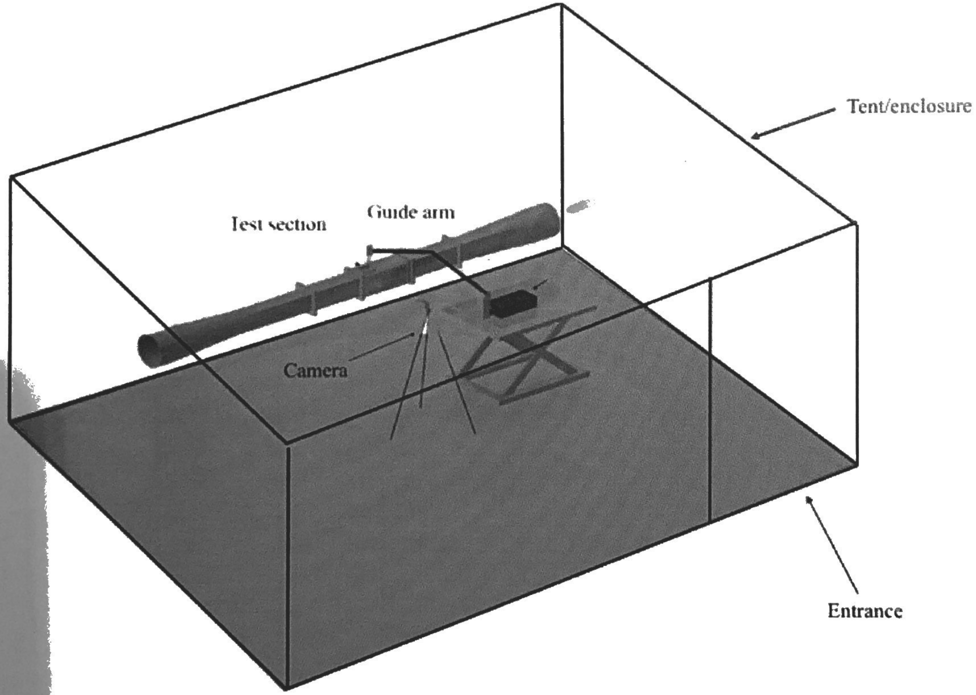
Project name	PIV measurements on blade cascade for HiFrancis project
Apparatus	Double-Cavity High speed laser, 2*10mJ, 1000Hz, 527nm
Unit	EPT
Equipment manager	Bård Brandåstrø
Project leader	Pål Tore Storli
HSE coordinator	Morten Grønli
HSE responsible (linjeleder)	Therese Løvås
Location	Waterpower Laboratory
Room number	11
Risk assessment performed by	Magne Tveit Bolstad

TABLE OF CONTENTS

ATTACHMENT A: PROCESS AND INSTRUMENTATION DIAGRAM

Radiation area is blocked on all sides by light-blocking fabric (this includes a roof), working as a sealed tent. The entrance is overlapping fabric.

A warning light by the entrance will activate when the laser is on.



ATTACHMENT B: HAZOP TEMPLATE

Project: **Node: Laser**	**Page**

Ref	Guideword	Causes	Consequences	Safeguards	Recommendations	Action	Date/Sign
	Unintentional rarefaction of beam	Poor alignment of laser beam, unintentional beam obstruction	Damage to personnel, damage to equipment	Laser emitter latched in place, radiation area enclosed	Careful instrument handling		
	Laser generation unit temperature rise	Cooling system failure	Equipment will overheat and break	-	Be aware of equipment condition		
	Unprotected personnel entering radiation area	Insufficient signalling, radiation area too accessible	Possibility of severe damage to personnel	Signalling and warning light in place, radiation area enclosed with physical obstruction	Always run experiments with warning light on		
	Electrical failures in apparatuses	Wear, poor construction	Equipment failure				
	Fire hazard	Laser beam hitting flammable material	Fire	Enclosing fabric approved for shielding laser experiments.	Make sure a fire extinguisher is easily available		
	Beam interception by operator	Careless equipment handling, obstructed work area	Damage to personnel, severe if lacking protective gear	Goggles approved for laser in use worn by operator	Careful instrument handling		

NTNU SINTEF

APPARATURKORT / UNITCARD

Dette kortet SKAL henges godt synlig på apparaturen!
This card MUST be posted on a visible place on the unit!

Apparatur (Unit) PIV measurements on blade cascade for HiFrancis project	
Prosjektleder (Project Leader) Pål Tore Storli	**Telefon mobil/privat** (Phone no. mobile/private) 73592518
Apparaturansvarlig (Unit Responsible) Bård Brandåstrø	**Telefon mobil/privat** (Phone no. mobile/private) 918 97 257
Sikkerhetsrisikoer (Safety hazards) Flashing laser light	
Sikkerhetsregler (Safety rules) -Wear appropriate safety goggles for laser wavelength -no shiny objects (eg. Jewellery) worn	
Nødstopp prosedyre (Emergency shutdown) Turn off laser using emergency stop or turn off warning light to the left of the tent-entrance.	

Her finner du (Here you will find):

Prosedyrer (Procedures)	Apparaturperm ved laser
Bruksanvisning (Users manual)	Apparaturperm ved laser

Nærmeste (Nearest)

Brannslukningsapparat (fire extinguisher)	På veggen vis a vis teltinngangen utenfor teltet.
Førstehjelpsskap (first aid cabinet)	Ved inngangen til laboratoriet i 2.etg.

NTNU
Institutt for energi og prosessteknikk

Dato

____

Signert

____

SINTEF Energi
Avdeling energiprosesser

Dato

____

Signert

____

9

NTNU

Norwegian University of
Science and Technology