


Summer 2017

Investigation, Modeling And Validation Of Digital Bridge For A New Generation Hot-Wire Anemometer

Karthik Kamalakar Joshi
Old Dominion University

Follow this and additional works at: https://digitalcommons.odu.edu/mae_etds

 Part of the [Aerospace Engineering Commons](#), and the [Heat Transfer, Combustion Commons](#)

Recommended Citation

Joshi, Karthik K.. "Investigation, Modeling And Validation Of Digital Bridge For A New Generation Hot-Wire Anemometer" (2017).
Master of Science (MS), thesis, Mechanical & Aerospace Engineering, Old Dominion University, DOI: 10.25777/frqv-p817
https://digitalcommons.odu.edu/mae_etds/29

This Thesis is brought to you for free and open access by the Mechanical & Aerospace Engineering at ODU Digital Commons. It has been accepted for inclusion in Mechanical & Aerospace Engineering Theses & Dissertations by an authorized administrator of ODU Digital Commons. For more information, please contact digitalcommons@odu.edu.

**INVESTIGATION, MODELING AND VALIDATION OF DIGITAL BRIDGE
FOR A NEW GENERATION HOT-WIRE ANEMOMETER**

by

Karthik Kamalakar Joshi

B.Tech, May 2011, Kakaitya Institute of Technology and Science, Warangal, India

A Thesis Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

MECHANICAL ENGINEERING

OLD DOMINION UNIVERSITY

August 2017

Approved by:

Dr. Colin P. Britcher (Advisor)

Dr. Robert L. Ash (Member)

Dr. Thomas Alberts (Member)

ABSTRACT

INVESTIGATION, MODELING AND VALIDATION OF DIGITAL BRIDGE FOR A NEW GENERATION HOT-WIRE ANEMOMETER

Karthik Kamalakar Joshi
Old Dominion University, 2017
Director: Dr. Colin P. Britcher

The Digital Bridge Thermal Anemometer (DBTA) is a new generation anemometer that uses advanced electronics and a modified half-Wheatstone bridge configuration, specifically a sensor and a shunt resistor in series. This allows the miniaturization of the anemometer and the communication between host computer and anemometer is carried out using serial or ethernet which eliminates the noise due to the use of long cables in conventional anemometer and the digital data sent to host computer is immune to electrical noise. In the new configuration the potential drop across a shunt resistor is used to control the bridge.

This thesis is confined to the anemometer used in constant temperature (CT) mode. The heat transfer relations are studied and new expressions are developed based on the new configuration of the bridge using perturbation analysis. The theoretical plant model of a commercially available sensor and a custom built sensor are derived and quantified. The plant model is used to design a controller to control the plant in closed-loop using feedback. To test the performance of the modified sensor used with a "generation-I" bridge and DAQ, an experiment was conducted. The controller was implemented in a user interface in LabVIEW. The test is to compare the results between a conventional TSI sensor with an IFA 300 anemometer and the setup describe above, in the wake behind a circular cylinder. Performance of the DBTA is satisfactory at low frequencies. A user interface capable of communicating with the anemometer to control the operation and collect data generated by anemometer is developed in LabVIEW.

Copyright, 2017, by Karthik Kamalakar Joshi, All Rights Reserved.

ACKNOWLEDGMENTS

I would like to thank all the people who have helped me in achieving this endeavor. First of all, I would like to extend my gratitude to my advisor, Dr. Colin P. Britcher, without whose guidance and support this dissertation was not possible. I would like to thank my committee members, Dr. Bob Ash and Dr. Thomas Alberts, for their time in serving on my dissertation committee. A special thanks to Dr. Han P. Bao, our former graduate program director, Dr. Sebastian Bawab, our Department Chair, and Ms. Diane Mitchell.

I would also like to thank the management and staff of ViGYAN inc. for their continuous cooperation and support. I'd like to thank Mr. John Bledsoe and Mr. Richard White for the support and encouragement during the research, development, and testing.

On a personal note, I'd like to thank my parents and my brother, for their support and encouragement throughout this endeavor. I would like to take this opportunity to thank all my friends who have made my stay at ODU such a memorable time of my life.

NOMENCLATURE

| | |
|------------|---|
| c | Constant associated with the thermal capacity of wire $\left(\frac{KJ}{\Omega}\right)$ |
| C | Temperature coefficient of electrical resistivity $\left(\frac{1}{K}\right)$ |
| C_p | Specific heat of fluid at constant pressure (if fluid is gas) $\left(\frac{KJ}{kgK}\right)$ |
| C_w | Specific heat of wire $\left(\frac{KJ}{kgK}\right)$ |
| d | Characteristic dimension (m) |
| E_b | Bridge voltage (V) |
| E_s | Shunt voltage (V) |
| h | Convective heat transfer coefficient of the flow $\left(\frac{W}{m^2K}\right)$ |
| k | Thermal conductivity $\left(\frac{W}{m * K}\right)$ |
| L | Characteristic linear dimension (m) |
| OHR | Overheat Ratio |
| R_g | Resistance of sensing element at ambient temperature (Ω) |
| R_w | Instantaneous resistance of the sensor (Ω) |
| R_s | Shunt resistance (Ω) |
| θ_g | Ambient temperature fluid temperature (K) |
| θ_w | Operating temperature (K) |
| U | Velocity of the flow $\left(\frac{m}{s}\right)$ |
| μ | Absolute or dynamic viscosity $\left(\frac{kg}{ms}\right)$ |
| ν | Kinematic viscosity $\left(\frac{m^2}{s}\right)$ |

TABLE OF CONTENTS

| | Page |
|--|------|
| LIST OF TABLES | viii |
| LIST OF TABLES | viii |
| LIST OF FIGURES | ix |
| Chapter | |
| 1 Introduction to Hot-Wire Anemometry | 1 |
| 2 Heat Transfer Relations of Hot-Wires | 7 |
| 2.1 Types of single wire sensors | 7 |
| 2.2 King's Law | 11 |
| 2.3 Plant model identification using experimentation | 17 |
| 2.4 Open-loop plant identification | 20 |
| 2.5 Open-loop plant model using step response | 23 |
| 3 Controller Design | 29 |
| 3.1 Introduction | 29 |
| 3.2 Plant identification | 30 |
| 3.3 Controller design | 33 |
| 3.4 P-Controller | 34 |
| 3.5 PI controller | 36 |
| 3.6 Converting the controller to discrete domain | 39 |
| 4 Graphical User Interface | 42 |
| 4.1 Introduction | 42 |

| | |
|---|-----|
| | vii |
| 4.2 Important concepts of GUI for DBTA | 43 |
| 4.3 Development of the GUI for the DBTA | 47 |
| 5 Concept Validation and Results | 56 |
| 5.1 Experiment | 56 |
| 5.2 Discussion of results | 58 |
| 6 Discussion and Conclusion | 64 |
| 6.1 Discussion | 64 |
| 6.2 Conclusion | 66 |
| APPENDIX A Theoretical Calculation | 67 |
| APPENDIX B Experimental Calibration | 70 |
| APPENDIX C Validation of Results | 78 |
| REFERENCES | 82 |
| VITA | 84 |

LIST OF TABLES

| Table | Page |
|--|------|
| 2.1 Different sensors used in the following experiments | 19 |
| 2.2 Open-loop plant model gain and time constant using various methods | 23 |

LIST OF FIGURES

| Figure | Page |
|---|------|
| 1.1 Traditional hot wire anemometer circuit | 3 |
| 1.2 DBTA with digital data link | 4 |
| 1.3 DBTA with digital data link without controller | 5 |
| 1.4 DBTA with digital data link with on-board controller | 5 |
| 2.1 Traditional hot-wire sensor (DANTEC sensors catalog) | 8 |
| 2.2 Scheme of the shear stress hot film sensor[1] | 9 |
| 2.3 Schematic diagram of the shear stress hot film sensor[1] | 9 |
| 2.4 Traditional hot-film sensor (DANTEC sensors catalog) | 10 |
| 2.5 Section of hot-wire | 12 |
| 2.6 Non-linear voltage-current characteristics of a hot-wire filament | 14 |
| 2.7 Generation-I bridge | 18 |
| 2.8 TSI T1.5 and modified Sensor 2 | 20 |
| 2.9 Sensor voltage map for different velocities and overheat ratios | 24 |
| 2.10 Bridge voltage map for different velocities and overheat ratios | 25 |
| 2.11 Sensor resistance map for different velocities and overheat ratios | 26 |
| 2.12 X coefficient map for different velocities and overheat ratios | 27 |
| 2.13 Y coefficient map for different velocities and overheat ratios | 28 |
| 3.1 Open loop and closed loop feedback system representation | 29 |
| 3.2 Identification of plant model using <i>pidtool</i> | 31 |
| 3.3 Step response plot of open-loop plant | 32 |
| 3.4 Bode plot of open-loop plant | 32 |
| 3.5 Unity feedback system | 33 |
| 3.6 Root locus plot | 36 |

| | |
|--|----|
| | x |
| 3.7 Closed loop step response | 37 |
| 3.8 Root locus of the plant cascaded with PI controller | 38 |
| 3.9 Step response and control effort (green) of controller cascaded with plant . . | 38 |
| 3.10 Bode plot of controller cascaded with plant | 39 |
| 3.11 Discrete time controller line diagram | 41 |
| | |
| 4.1 Example of front panel | 43 |
| 4.2 Example of block diagram | 43 |
| 4.3 While loop diagram | 45 |
| 4.4 State machine line diagram | 46 |
| 4.5 Event structure with while loop | 47 |
| 4.6 Line diagram | 47 |
| 4.7 Queued message handler command flow | 48 |
| 4.8 Example of event handling loop | 49 |
| 4.9 Enqueue VI terminal description | 49 |
| 4.10 Example of message handling loop | 52 |
| 4.11 Acquisition loop | 54 |
| 4.12 Communications loop | 55 |
| | |
| 5.1 Top view of test setup | 57 |
| 5.2 Velocity distribution along-wind | 59 |
| 5.3 Turbulence intensity distribution along-wind | 60 |
| 5.4 Power-spectral density (log-scale) @ -4in | 60 |
| 5.5 Power-spectral density (linear-scale) @ -4in | 61 |
| 5.6 Power-spectral density @ 0in | 62 |
| 5.7 Power-spectral density @ 0in | 62 |
| 5.8 Power-spectral density | 63 |
| 5.9 Power-spectral density | 63 |

CHAPTER 1

Introduction to Hot-Wire Anemometry

An anemometer is a device used for measuring wind characteristics such as speed, direction or consistency of flow which relates to turbulence etc. The name is derived from Greek word **anemos**, which means wind. There are different types of anemometers used for different situations. Cup anemometers, ultrasonic anemometers, vane anemometers, hot-wire anemometers, and ping-pong ball anemometers are some of the velocity-based intrusive anemometers. Laser doppler velocimeter, particle image velocimeter are examples of velocity-based non-intrusive type anemometers. The choice of anemometers is based on the type of measurement and the level of precision required.

The desire to understand the mechanism of turbulence led to an early recognition of the need for accurate measurement of the instantaneous fluid velocity [2]. Turbulence is one of the important factors that is studied during wind tunnel testing. It plays a major role from construction of a wind tunnel and to the testing carried out in the wind tunnel. One of the most robust and effective ways to measure turbulence is by using a hot-wire anemometer.

Thermal anemometers (a.k.a hot-wire anemometers) have been a key experimental tool used in fluid mechanics for many decades. The use of heated sensor in the study of turbulent flow is so common due to its small physical size, high frequency response and high level of precision in the output. The most commonly used heated sensor types, hot-wire and hot-film, are essentially electrically heated metallic elements whose temperature is uniquely influenced by the instantaneous flow of fluid past them [3]. The exact origin of hot-wire anemometry is difficult to determine but according to L. V. King [4] preliminary experiments were carried out by Shakespeare at Birmingham as early as 1902. It is now over 11 decades since publication of the seminal study of heat transfer from cylinders in cross-flow [5] that

underpins the method. The limitations in technology in electronics of the era caused the early prevalence of the constant current mode of operation, often with analog frequency response compensation. In the 1960's, reliable constant temperature anemometer circuitry became commercially available and the vast majority of applications now utilize this approach.

In the 1980's, anemometers were adapted to rely on computerized control of settings, such as the TSI IFA-300 and the DANTEC StreamLine anemometers. It is important to realize that the basic configuration of the hot-wire anemometer has changed little in the last 50 years (i.e. using Constant Temperature (CT) operation) and indeed exhibits many similarities to the configurations used 100 years ago. That is to say, the small electrically-heated sensor is connected via a long cable (up to 150 feet long in some large wind tunnels) to a Wheatstone bridge circuit in the anemometer itself. The characteristics of the cable (sensor cable) connecting sensor to the rest of the circuit has a profound influence on system operation with particularly acute problems where the fluid temperature is a strong variable. Figure 1.1 illustrates the conventional bridge configuration. The fact is that the long cable has its own resistance, capacitance, inductance and is even sensitive to external electromagnetic interference. The compensation to the change in resistance is calculated based on the imbalance in the bridge and the disturbance in imbalance due to the above mentioned factors that cannot be accounted for, as some cannot even be quantified. Thus, there is a chance that the sensor may be over- or under-compensated due to disturbances in the circuitry other than sensor itself.

The Wheatstone Bridge configuration uses an indirect way to maintain the temperature of the sensor by measuring the voltage fluctuations in the bridge to estimate the correction voltage. For every operating resistance of the wire there should be a balancing resistance on the passive side of the bridge shown as the R_3 resistor in Figure 1.1. This limits the range of practical operating resistances. This configuration is not fundamental to the operation of the device, rather, it serves two main purposes:

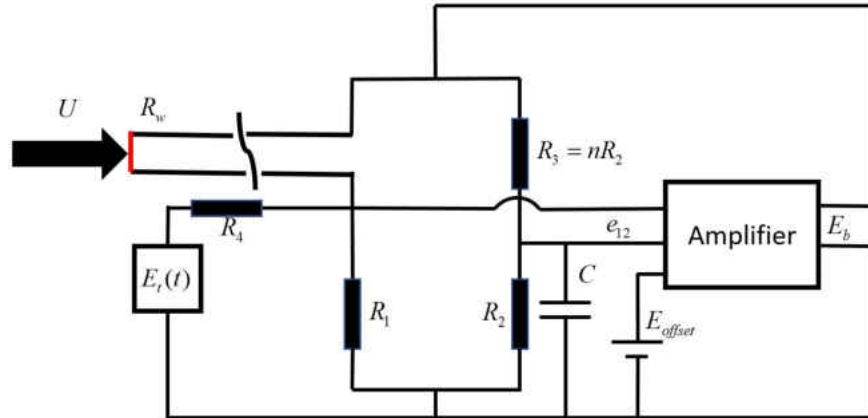


Figure 1.1: Traditional hot wire anemometer circuit

1. As a convenient way to establish a sensor operating point by "balancing" the active side of the bridge against a passive side with known resistor values.
2. As a convenient way to isolate the small fluctuating voltage associated with low-level turbulence from the steady-state signal.

Further, two modes of operation of thermal anemometers are commonly employed [6], [7], that is Constant Temperature (CT) and Constant Current (CC), with a third used mainly for qualitative measurements, namely Constant Voltage (CV) [8]. Pulse-width modulated (PWM) anemometers have been investigated [9]. However, this approach does not resolve the cable length issue as the basic structure of the circuit remains much the same as Figure 1.1, with limited bandwidth. What is now being proposed is a fundamental change to the anemometer configuration, with two related aspects.

1. The essential circuitry to power the sensor and establish its operating point will be packaged immediately adjacent to the sensor, i.e. in the typical probe holder, removing the effects of the cable connecting the sensor to an external anemometer.
2. Modern analog-digital conversion hardware (analog-to-digital - A/D; digital-to-analog - D/A) will be employed to the maximum extent possible, ideally including directly

driving the sensor from a D/A converter. Data transmission to the "anemometer" can then be largely digital, immune to environmental variations and electrical noise.

It should be noted that direct excitation via a D/A converter should permit the choice of operating modes (i.e. CC, CT, CV) as there is no need to change the circuit. Only changes in the feedback control algorithm (i.e. firmware) of the controller are needed. The ultimate objective of the research is therefore referred to as a "Digital Bridge". Figure 1.2 shows the concept in its purest form. Emphasis is placed on the Constant Temperature (CT) mode. There are two possible variations of the concept

1. With a high-speed digital communication link and feedback control via a remote processor. However, this may affect the compensation process speed. Figure 1.3 illustrates this configuration.
2. With the feedback hardware packaged adjacent to the sensor, with a digital communication link to permit remote control and data recovery. Figure 1.4 illustrates this configuration.



Figure 1.2: DBTA with digital data link

The sensor is directly driven from a high-speed, high precision D/A converter. The available power is increased using a power amplifier. The hot-wire and the shunt resistor are essentially like the left half of the Wheatstone bridge in Figure 1.1. The A/D converter has

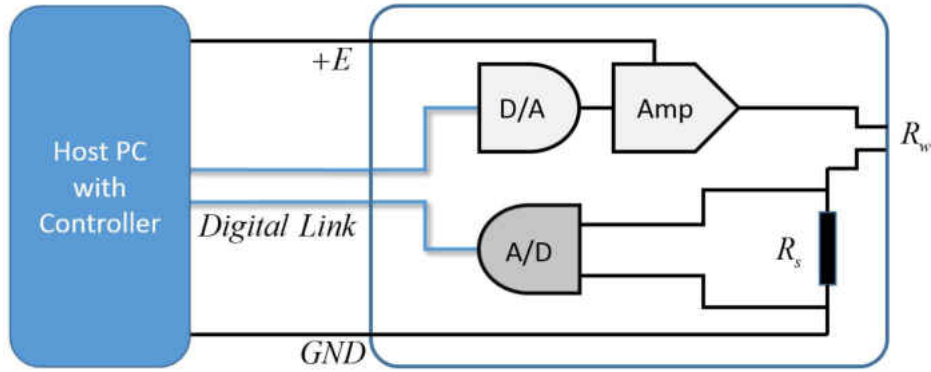


Figure 1.3: DBTA with digital data link without controller

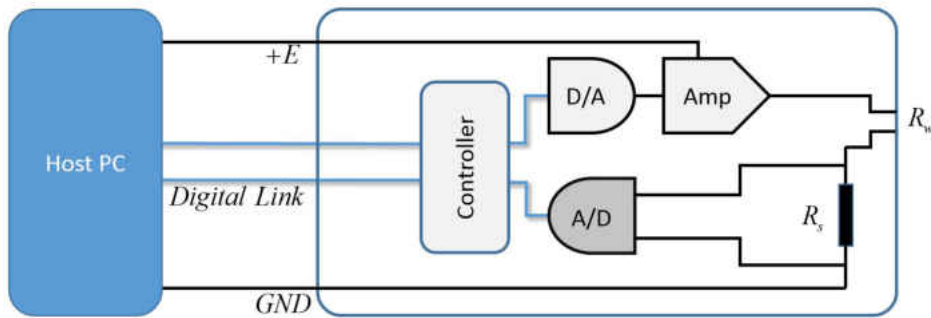


Figure 1.4: DBTA with digital data link with on-board controller

high impedance so a very tiny amount of current is consumed which will not effect the bridge current. The voltage drop across the shunt resistor will be the measure of the current drawn from the amplifier. The bridge voltage, shunt voltage and the current passing through the circuit are therefore known. The resistance of the sensor can then simply be calculated based on Equation 1.1.

$$R_w = \frac{(E_b - E_s)R_s}{E_s} \quad (1.1)$$

where

E_b = Input voltage to the bridge

E_s = Measured voltage across shunt resistor

R_s = Resistance of the shunt resistor

R_w = Resistance of wire

Even though the voltage/current characteristics are non-linear for the sensor, because of small perturbation assumptions, the relation can be linearized about a given operating point. Frequency response and other practical issues are discussed in more detail in later chapters. Various considerations such as range/resolution and other factors might lead to modify the configuration of the circuit.

CHAPTER 2

Heat Transfer Relations of Hot-Wires

The heat transfer from a heated sensor placed in a fluid flow depends on the properties of the ambient fluid (density (ρ), viscosity (μ), thermal conductivity (k), etc.), the properties of the sensor (length of sensing element (l), area of cross-section (A), ratio of surface area to cross-sectional area (A_s/A), density of wire (ρ_w), specific heat of sensor material per unit mass (c_w)) and the parameters of the flow (velocity vector (V), fluid temperature (T_a), pressure (P_a)), etc. The following discussion is derived from [6] and [7] and reproduced to give theoretical insight of the heated sensor relation with the medium.

2.1 Types of single wire sensors

There are different types of hot wire anemometer sensors available namely cylindrical, wedge, cone and flush mounted etc. The choice of type of sensor is based on the application. In this paper the topic is limited to **wire and film** type sensors.

2.1.1 Hot-wire sensor

The traditional sensor for this research has been a fine wire. The wire sensor is very useful for very low turbulence intensities as the smaller the wire, the faster the response. The wire sensor is typically a $25\mu m$ diameter silver wire which has a thin thread of platinum $4 - 5\mu m$ diameter down its central core as shown in Figure 2.1. This is known as **platinum Wollaston wire** [6]. Silver wire provides structural support to the sensing element and is used to solder the sensing element to the prongs. Core is the actual sensing element and is exposed by removing silver using acid etching. The length of the sensing element is determined by l/d (length to diameter) ratio which should be roughly 200. Platinum wire withstands high temperatures in an oxidizing atmosphere but is susceptible to damage from

debris. To avoid this problem platinum-coated tungsten wire $4 - 5\mu m$ diameter can be used as the core. Tungsten is very strong and has a high temperature coefficient of resistance. It will, however, deteriorate at high temperatures in oxidizing atmospheres (air) so the platinum coating of about $50 - 100nm$ thickness is used as protective layer. Platinum-iridium alloy also is used if more strength is needed at high temperature.

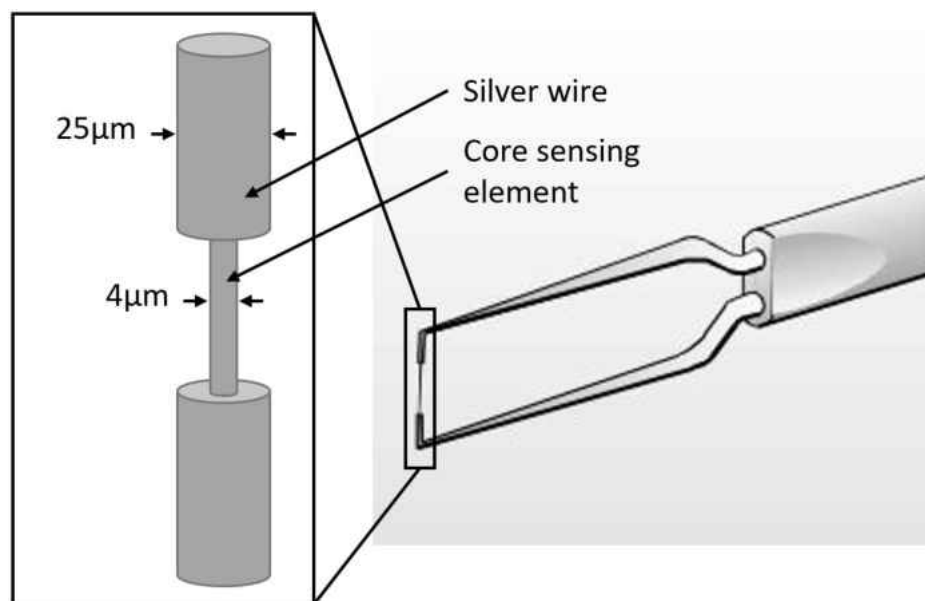


Figure 2.1: Traditional hot-wire sensor (DANTEC sensors catalog)

2.1.2 Hot-film sensor

The hot film sensor is essentially a conducting film laid on an insulating substrate such as **ceramic or quartz** (usually **quartz**). These hot film sensors can be sub classified into following.

- Surface film sensors
- Cylindrical film sensors

Surface film sensors

These type of film sensors are essentially laid on a flat surface. The sensor consists of two flat conductive films[1]: one that will perform as a sensing element and the other that will serve as a thermal shield between the sensor and the wall. Figure 2.2 shows the arrangement of the sensor on a surface and Figure 2.3 shows the schematic of the surface film sensor.

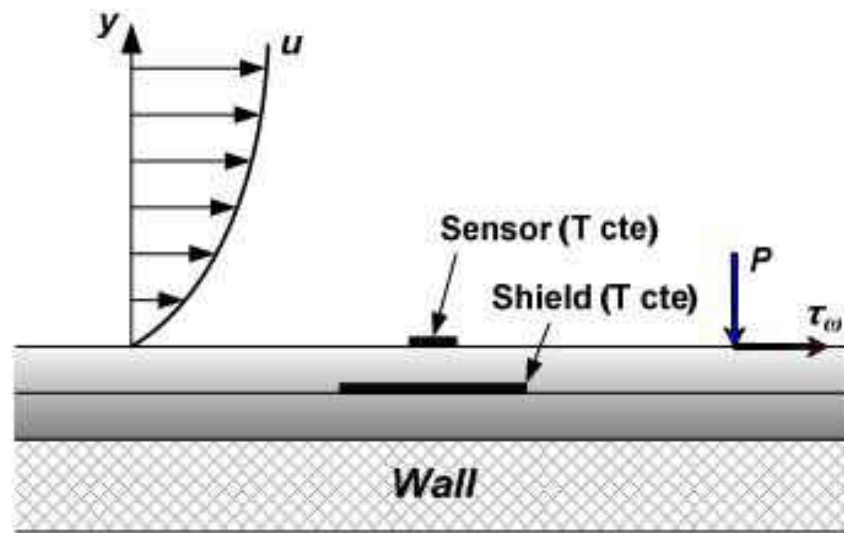


Figure 2.2: Scheme of the shear stress hot film sensor[1]

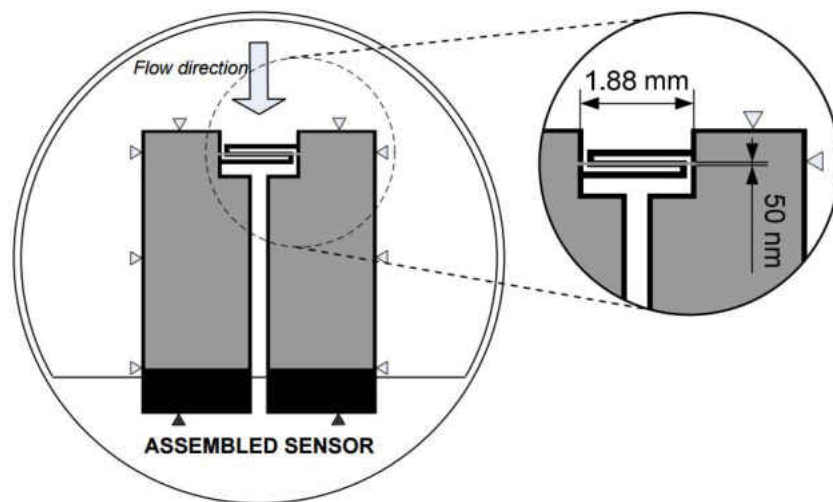


Figure 2.3: Schematic diagram of the shear stress hot film sensor[1]

Cylindrical film sensors

The sensor is a cylindrical quartz filament coated with platinum film using cathode sputtering to ensure a uniform thickness of sensing element [7]. The film is usually $0.1\mu m$ thick and the overall diameter of the sensing element is $25 - 70\mu m$ and the length is determined by $l/d = 200$. When compared with hot wires, the cylindrical hot film sensor has many advantages such as its resistance to damage due to particles in the flow and is easier to clean. However, the double structure (film plus substrate) makes its frequency response more complex. The metal film thickness on a typical film sensor is less than $0.1\mu m$. Thus, the mechanical strength and the effective thermal conductivity of the sensor are determined almost entirely by the substrate material. Figure 2.4 shows a typical hot-film.

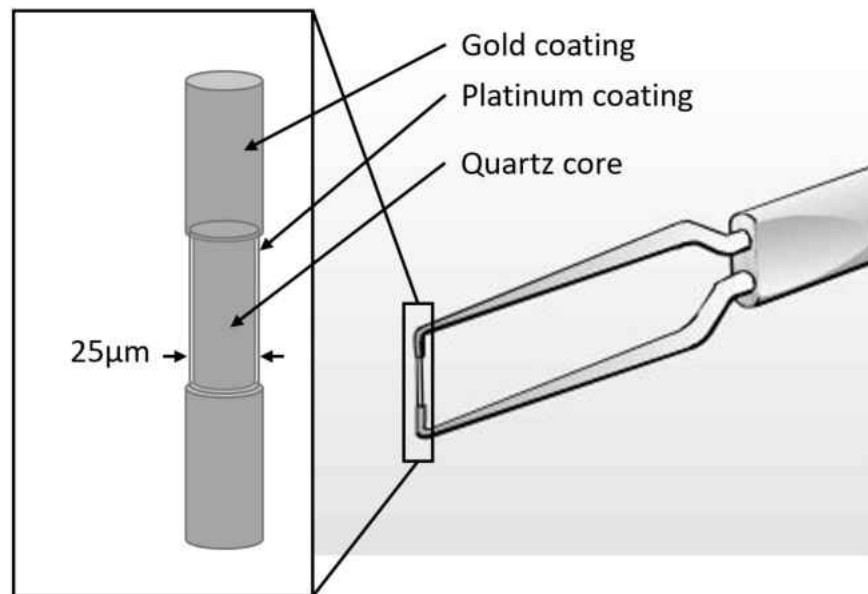


Figure 2.4: Traditional hot-film sensor (DANTEC sensors catalog)

During the actual measurement of heat transfer between the sensor and its environment, the thermal anemometer will respond to changes in parameters other than velocity, such as temperature, pressure and fluid composition. While this adds to versatility, it also means that when more than one parameter is changing, special techniques must be used to extract

velocity. Temperature compensation for the changes in ambient temperature is done by constant monitoring of ambient temperature.

2.2 King's Law

The heat-transfer relationship is expressed in terms of the non-dimensional groups of Nusselt (Nu), Reynolds (Re), and Prandtl (Pr) numbers. The heat-transfer from the hot-wire is expressed in these terms by King [10] as:

$$Nu = A + B\sqrt{Re} \quad (2.1)$$

where A, B are empirical calibration constants for the fluid. A more general form of the equation is proposed by Kramers [11] considering the temperature of the film (θ_f) as the average of ambient temperature (θ_a) and wire temperature (θ_w) for the fluid properties as:

$$Nu = 0.42Pr^{0.2} + 0.57Pr^{0.33}Re^{0.5} \quad (2.2)$$

The Prandtl Number is a dimensionless number approximating the ratio of momentum diffusivity (kinematic viscosity) to the thermal diffusivity and can be expressed as:

$$Pr = \frac{\mu C_p}{k} \quad (2.3)$$

The Reynolds Number is a dimensionless number approximating the ratio of inertial forces to viscous forces and can be expressed as:

$$Re = \frac{vL}{\nu} \quad (2.4)$$

The Nusselt number is the ratio of convective to conductive heat transfer across the boundary

layer.

$$Nu = \frac{hd}{k} \quad (2.5)$$

At a given instant the heat supplied to the hot-wire transfers to the surroundings mainly in three ways – conduction, convection and radiation. Any remaining heat is stored in the wire. This energy equation for the wire is expressed analytically as:

$$d\dot{Q}_e = d\dot{Q}_{fc} + d\dot{Q}_c + d\dot{Q}_r + d\dot{Q}_s \quad (2.6)$$

$d\dot{Q}_e = \frac{I^2 \chi_w}{A_w} dx$ is the electrical heat-generation rate

$d\dot{Q}_{fc} = \pi dh(\theta_w - \theta_g) dx$ is the forced-convective heat-transfer rate

$d\dot{Q}_c = k_w A_w \frac{\partial^2 \theta_w}{\partial x^2} dx$ is the conductive heat-transfer rate

$d\dot{Q}_r = \pi d \sigma \varepsilon (\theta_w^4 - \theta_g^4) dx$ is the radiation heat-transfer rate

$d\dot{Q}_s = \rho_w c_w A_w \frac{\partial \theta_w}{\partial t} dx$ is the heat storage rate (i.e. rate of change of internal energy).

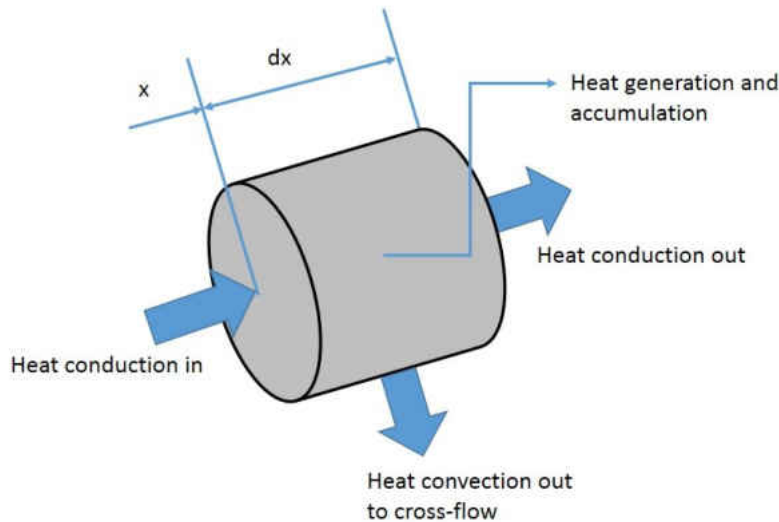


Figure 2.5: Section of hot-wire

By inserting these relations in Equation 2.6, the following equation is obtained:

$$\frac{I^2 \chi_w}{A_w} dx - \pi dh(\theta_w - \theta_g) dx - k_w A_w \frac{\partial^2 \theta_w}{\partial x^2} dx - \pi d \sigma \varepsilon (\theta_w^4 - \theta_g^4) dx - \rho_w c_w A_w \frac{\partial \theta_w}{\partial t} dx = 0 \quad (2.7)$$

For a wire of unit length the fluid temperature θ_a , is approximately constant along the wire and the radiation effect is much less than the continuous forced convection, so both conduction and radiation terms can be neglected. When resistivity is expressed in-terms of resistance for a given length and area of cross-section of the wire, Equation 2.7 can be written as

$$I^2 R_w = (R_w - R_g)(X + Y\sqrt{U}) + c \frac{dR_w}{dt} \quad (2.8)$$

The X, Y are analytically calculated according to Perry [6] as:

$$X = \frac{0.42kA_s}{R_g C d} \left(\frac{\mu C_p}{k} \right)^{0.2}; Y = \frac{0.57kA_s}{R_g C d} \left(\frac{\mu C_p}{k} \right)^{0.33} \left(\frac{\rho d}{\mu} \right)^{0.5}$$

The thermal capacity of the wire, $c = \frac{C_w v}{R_g C}$.

The voltage drop across the probe is related to the current in a steady state at constant velocity as:

$$E_w = IR_w = \frac{R_a F(\bar{U}) I}{F(\bar{U}) - I^2} \quad (2.9)$$

The relation between voltage drop and current across the probe is as shown in Figure 2.6. It is clear that there is no linear relationship to establish a transfer function to be used as a plant model, so small d.c. perturbation analysis is used to linearize Equation 2.8. Since the input to the bridge is the bridge voltage E_b , the current I is substituted as:

$$I = \frac{E_b}{R_w + R_s}$$

Equation 2.8 becomes:

$$\left(\frac{E_b}{R_w + R_s}\right)^2 R_w = (R_w - R_g) (X + Y\sqrt{U}) + c \frac{dR_w}{dt} \quad (2.10)$$

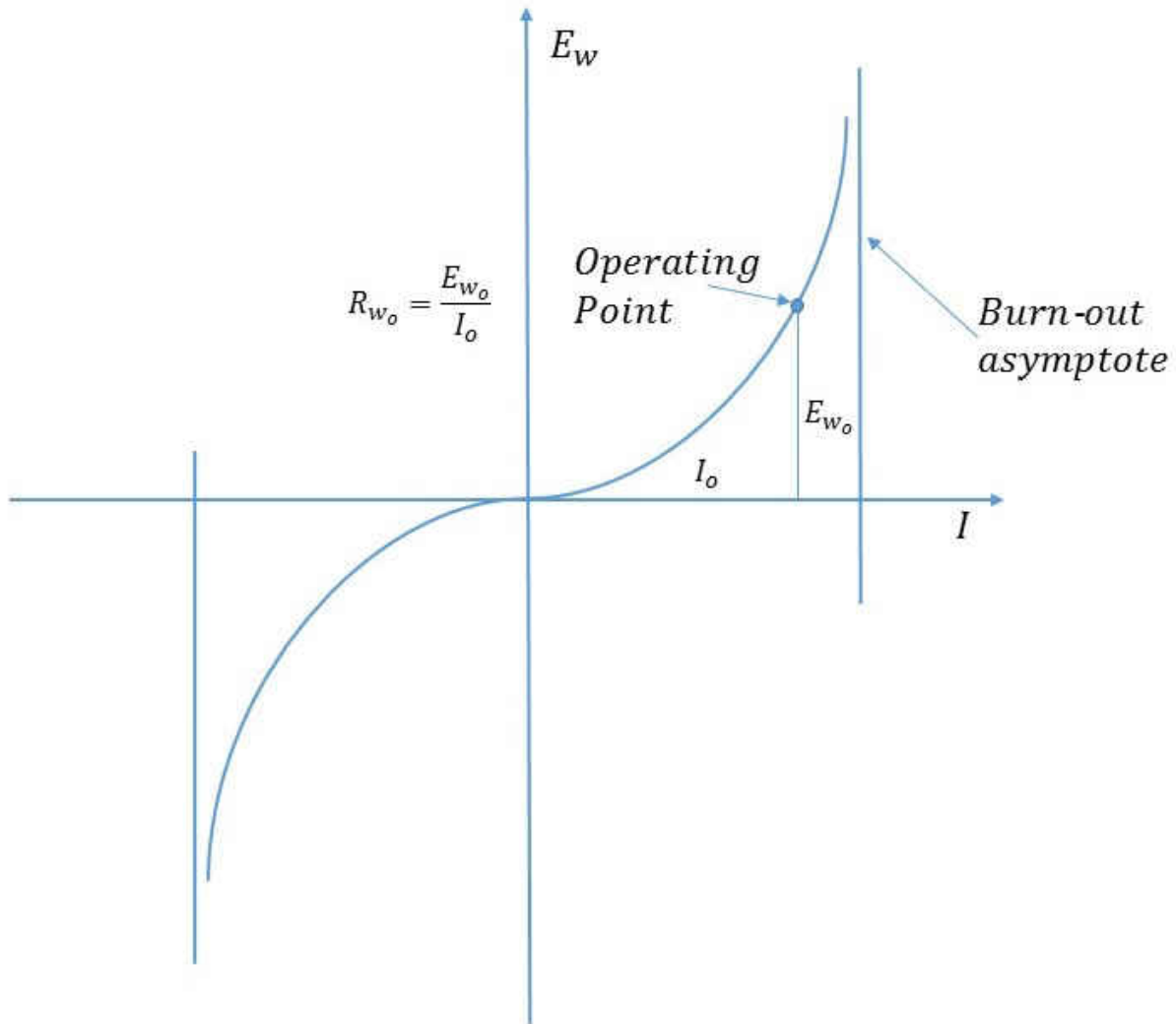


Figure 2.6: Non-linear voltage-current characteristics of a hot-wire filament

Rearranging the terms in Equation 2.10 becomes:

$$E_b^2 R_w = (R_w + R_s)^2 (R_w - R_g) (X + Y\sqrt{U}) + c (R_w + R_s)^2 \frac{dR_w}{dt} \quad (2.11)$$

In order to obtain a linear system equation, the above non-linear equation is linearized using perturbation analysis about an operating point. Let lower case letters denote small perturbations:

$$E_b = \bar{E}_b + e$$

$$R_b = \bar{R}_b + r$$

$$U = \bar{U} + u$$

Equation 2.11 becomes:

$$\begin{aligned} (\bar{E}_b + e)^2 (\bar{R}_w + r) &= (\bar{R}_w + r + R_s)^2 (\bar{R}_w + r - R_g) \left(X + Y\sqrt{\bar{U} + u} \right) \\ &+ c (\bar{R}_w + R_s + r)^2 \frac{dr}{dt} \end{aligned} \quad (2.12)$$

Expanding the terms considering:

$$M = \bar{R}_w + R_s$$

$$N = \bar{R}_w - R_g$$

the LHS becomes:

$$\bar{E}_b^2 \bar{R}_w + 2\bar{E}_b \bar{R}_w e + \bar{R}_w e^2 + \bar{E}_b^2 r + 2\bar{E}_b e r + e^2 r$$

the RHS becomes:

$$\begin{aligned} (M^2 N + 2MNr + Nr^2 + M^2 r + 2Mr^2 + r^3) &\left(X + Y\sqrt{\bar{U}} \left(1 + \frac{u}{2\bar{U}} \right) \right) + c(M + r)^2 \frac{dr}{dt} \end{aligned} \quad (2.13)$$

Expanding the RHS and eliminating higher order perturbations from both LHS and RHS gives:

$$\bar{E}_b^2 \bar{R}_w + 2\bar{E}_b \bar{R}_w e + \bar{E}_b^2 r = (M^2 N + 2MNr + M^2 r)(X + Y\sqrt{\bar{U}}) + \left(\frac{Y}{2\sqrt{\bar{U}}} \right) M^2 N u + cM^2 \frac{dr}{dt} \quad (2.14)$$

Taking a time average of Equation 2.14 gives:

$$\bar{E}_b^2 \bar{R}_w = M^2 N \left(X + Y \sqrt{\bar{U}} \right) \quad (2.15)$$

Subtracting Equation 2.15 from Equation 2.14 gives:

$$2\bar{E}_b \bar{R}_w e + \bar{E}_b^2 r = \left(2MNF(\bar{U}) + M^2 F(\bar{U}) \right) r + M^2 N \left(\frac{Y}{2\sqrt{\bar{U}}} \right) u + cM^2 \frac{dr}{dt} \quad (2.16)$$

$$2\bar{E}_b \bar{R}_w e = \left(2MNF(\bar{U}) + M^2 F(\bar{U}) - \bar{E}_b^2 \right) r + M^2 N \left(\frac{Y}{2\sqrt{\bar{U}}} \right) u + cM^2 \frac{dr}{dt} \quad (2.17)$$

– where $F(\bar{U}) = X + Y\sqrt{\bar{U}}$

To simplify the Equation 2.17, let

$$D_1 = \left(2MNF(\bar{U}) + M^2 F(\bar{U}) - \bar{E}_b^2 \right)$$

$$D_2 = 2\bar{E}_b \bar{R}_w$$

$$D_3 = M^2 N \left(\frac{Y}{2\sqrt{\bar{U}}} \right)$$

therefore Equation 2.17 becomes

$$D_2 e = D_1 r + D_3 u + cM^2 \frac{dr}{dt} \quad (2.18)$$

Converting the above time domain equation into a frequency domain:

$$r = \left(\frac{\alpha}{T_w s + 1} \right) e + \left(\frac{\beta}{T_w s + 1} \right) u \quad (2.19)$$

where

$$\alpha = \frac{D_2}{D_1}$$

$$\beta = \frac{-D_3}{D_1}$$

$$T_w = \frac{cM^2}{D_1}$$

Equation 2.19 gives the plant model with two inputs i.e. velocity and bridge voltage perturbation. Since velocity cannot be controlled it is considered as an external disturbance. Therefore the plant model to be considered is

$$\frac{r}{e} = \left(\frac{\alpha}{T_w s + 1} \right) \left[\frac{\Omega}{V} \right] \quad (2.20)$$

The derivation is carried out considering 'lumped' for the sensing element thus the constant T_w is referred as 'bulk heating' time constant the units are 'sec'. From the values of T_w and α it is clear that the plant model is strongly dependent on velocity (\bar{U}), bridge offset voltage (\bar{E}_b) and operating Over Heat Ratio (OHR). Over heat ratio is defined as the ratio of operating resistance to the ambient resistance of the sensor.

$$OHR = \frac{R_w}{R_g} \quad (2.21)$$

2.3 Plant model identification using experimentation

The closed-loop time response of a commercially available anemometer is tested using **electronic square wave test** on a TSI Model 1076 anemometer which is a conventional anemometer with configuration shown in Figure 1.1. The sensor used in this experiment is TSI T1.5 wire which is 0.00015in diameter and 0.05in length. The $-3dB$ point for the sine wave test is 138KHz which equates to the time response of 5.5 μ sec [12]. Since the actual DBTA is under development, validation of the DBTA cannot be done to compare the

result to conventional anemometer. So a generation-I bridge is used instead. This bridge has a power amplifier, sensor and shunt resistance connected in series is used for concept validation. The configuration is illustrated in Figure 2.7. The output is obtained in analog form using a NI 9205 data acquisition system and converted to digital form and sent to the LabVIEW UI (User Interface) developed separately to run this setup.

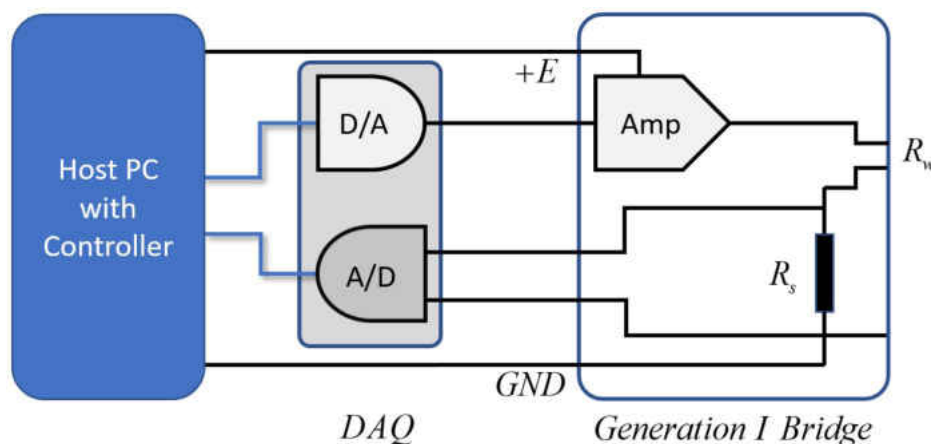


Figure 2.7: Generation-I bridge

Generation-I can run at a maximum loop rate of 1kHz loop-rate. So the maximum frequency that can be observed is limited to $500Hz$. the closed loop system cannot handle the fast fluctuations as the corrected value of bridge voltage may over or under compensate given the state of the sensor i.e. the resistance would've changed. To address this there is a need for a sensor with similar characteristics but with slow response time. Slow response can only be achieved if the thermal capacity of the sensor is high. The idea is to design a sensor with time response in units of *msecs*.

The sensors used in the following experiment are modified to suit the available hardware to operate it. Many metals and alloys of metals such as Nichrome (NiCr), Kanthal (Fe-CrAl), Tungsten and Platinum were examined to select a candidate for the modified sensor. Nichrome and Kanthal have low temperature coefficients of resistivity i.e. the resistance

change with the temperature change is very low which is not suitable for operation in constant temperature mode as the quantification of the change in temperature is given in terms of resistance. Even though Tungsten has high value of temperature coefficient of resistivity as platinum it is highly reactive to oxygen at elevated temperatures. The equipment used in fabricating the modified sensor cannot be used to make sensors using platinum coated tungsten wire as the platinum coating will be lost during soldering of sensing to the prongs and might cause oxidation of tungsten. The last option is pure platinum. Pure platinum is also a material commonly used in hot-wire sensors. The diameter and the length of the sensor are chosen such that the thermal capacity of the wire is increased. A compromise between thermal capacity and cold resistance is to be achieved. Table 2.1 gives a comparison between the new modified and commercially available wire type sensor and Figure 2.8 shows the two sensors

An open-loop step response is analyzed for the TSI wire and Sensor 2 from Table 2.1 and the time taken for both the sensors are roughly calculated as 0.001sec and 0.5sec respectively. The sensing element dimensions for the modified sensor is chosen such that the closed loop to open loop response ratio is consistent.

Table 2.1: Different sensors used in the following experiments

| Modified sensor | | |
|--------------------------|---------------|----------|
| Property | TSI T1.5 wire | Sensor 2 |
| Material | Platinum | Platinum |
| Diameter (μm) | 5.10 | 25.40 |
| Length (mm) | 1.27 | 10.90 |
| Cross-section (m^2) | 2.04e-11 | 5.07e-10 |
| Surface area (m^2) | 2.04e-08 | 8.70e-07 |
| Cold resistance | 5.47 | 2.60 |
| Thermal capacity (c) | 4.57e-06 | 2.8e-03 |

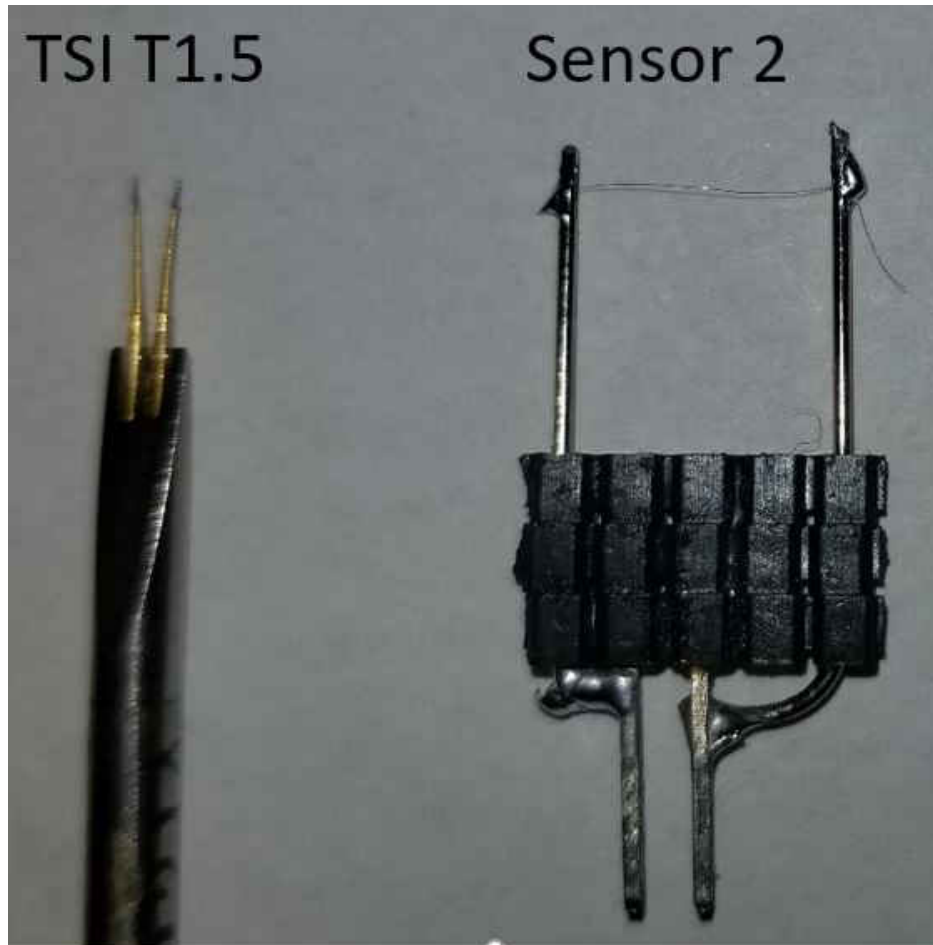


Figure 2.8: TSI T1.5 and modified Sensor 2

2.4 Open-loop plant identification

The open-loop plant model for a given sensor (i.e. α , T_w) are dependent on the values of X , Y , \bar{E}_b and \bar{R}_w at a given flow velocity as seen in Equation 2.20. The α and T_w values can be obtained in two ways.

1. Theoretical calculation using derived equations.
2. Experimental calibration using open loop plant in steady state.

2.4.1 Theoretical calculation

The theoretical transfer function is obtained based on the perturbation analysis about an operating point given by Equation 2.20. α and T_w are dependent on \bar{E}_b , \bar{R}_w , X and Y values. \bar{E}_b and \bar{R}_w are inter-related to each other in steady state. For a given sensor at a given flow the open-loop transfer function depends on the operating temperature of the sensor. The X and Y values are calculated based on the sensing element characteristics and the flow parameters so they remain constant. It is to be noted that the values of X , Y and R_o are very sensitive to the dimensions of sensing element and simple measurement cannot be done to validate the sensor dimensions, so experimental calibration is an effective way to determine the constants for a given sensor. The theoretical calculations are done using the MATLAB code given in Appendix A.

2.4.2 Experimental calibration

It is not always easy to obtain the sensing element characteristics such as the composition of the material and accurate dimensions which can vary results when computing theoretical calculations. Thus, experimental determination of the open-loop transfer function is preferred. This method does not require the sensor characteristics to determine X and Y values. The following procedure is used for the calibration.

1. Determination of cold resistance of the sensor.
2. Determination of operating resistance based on the desired overheat ratio (not to exceed $300^{\circ}C$).
3. The value of X is calculated using the sensor voltage required to maintain the operating resistance of the sensor at no flow. From the derivation, X addresses heat loss due to all other factors except forced convection. The value remains constant at any flow velocity at a given OHR.

$$E_w = E_b - E_s \quad (2.22)$$

$$X = \frac{E_w^2}{\bar{R}_w(\bar{R}_w - R_g)} \quad (2.23)$$

4. The value of Y is calculated by measuring the voltage across the sensor that is required to maintain the operating resistance at $> 0m/sec$ flow.

$$Y = \frac{\frac{E_w^2}{\bar{R}_w(\bar{R}_w - R_g)} - X}{\sqrt{U}} \quad (2.24)$$

From the above procedure experiments were conducted on TSI T1.5 wire and Sensor 2 to identify the plant model. The OHR ranges from 1.4 – 1.8 and the velocity range is 0 – 35m/sec for TSI T1.5 sensor and the OHR ranges from 1.6 – 1.9 and the velocity range is 0 – 35m/sec for Sensor 2 and this is because when the sensor tested at lower OHR did not work properly. The probable cause might be the higher heat capacity and imperfections in the build of the sensor. The following figures show the results.

From Figure 2.10 and Figure 2.9 it can be seen that the bridge voltage and sensor voltage maps for both TSI T1.5 and Sensor 2 look similar at different velocities and overheat ratios but the magnitude of voltages differ because of the fact that Sensor 2 has higher volume thus higher heat capacity so current required to maintain the sensor at a given resistance is high and so is the voltage. The resistance maps look slightly different as seen in Figure 2.11 and it is due to the fact that the cold resistance is different for both sensors.

The X coefficient maps for the two sensors is similar but the magnitudes differ because of the different dimensions of the wires. The value of X coefficient for a given OHR remains constant throughout the velocity range as X is not affected with flow velocity. The Y coefficient maps on the other hand look different. The probable cause might be the imperfect build of the sensor that allows it to behave differently.

From the above results all the required constants to find the open-loop plant model are known. From Equation 2.20 the values of α and T_w are calculated as

$$\alpha = \frac{2\bar{E}_b \bar{R}_w}{(2MNF(\bar{U}) + M^2F(\bar{U}) - \bar{E}_b^2)} \left[\frac{\Omega}{V} \right] \quad (2.25)$$

$$T_w = \frac{cM^2}{(2MNF(\bar{U}) + M^2F(\bar{U}) - \bar{E}_b^2)} [sec] \quad (2.26)$$

Table 2.2: Open-loop plant model gain and time constant using various methods

| | TSI T1.5 | | Sensor 2 | |
|---------------|-------------------|-------------------------|-------------------|-------------------------|
| | Gain (α) | Time constant (T_w) | Gain (α) | Time constant (T_w) |
| Theoretical | 11.44 | 0.0012 | 1.57 | 0.0375 |
| Experimental | 6.943 | 0.00041 | 2.15 | 0.041 |
| Step response | 6.608 | 0.00052 | 0.67 | 0.043 |

2.5 Open-loop plant model using step response

There is a simple way to find the open-loop plant model using the step response of the sensor. The sensor is supplied with a step voltage and the response is analyzed to identify the model. This is explained in detail in the next chapter.

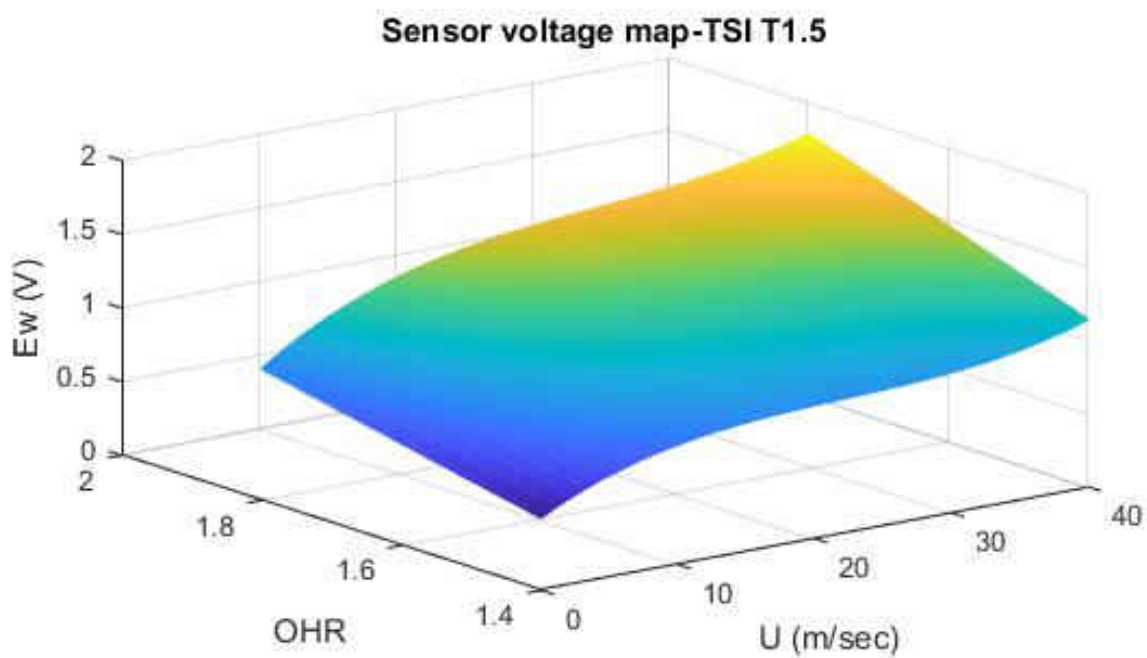


Figure 2.9: Sensor voltage map for different velocities and overheat ratios

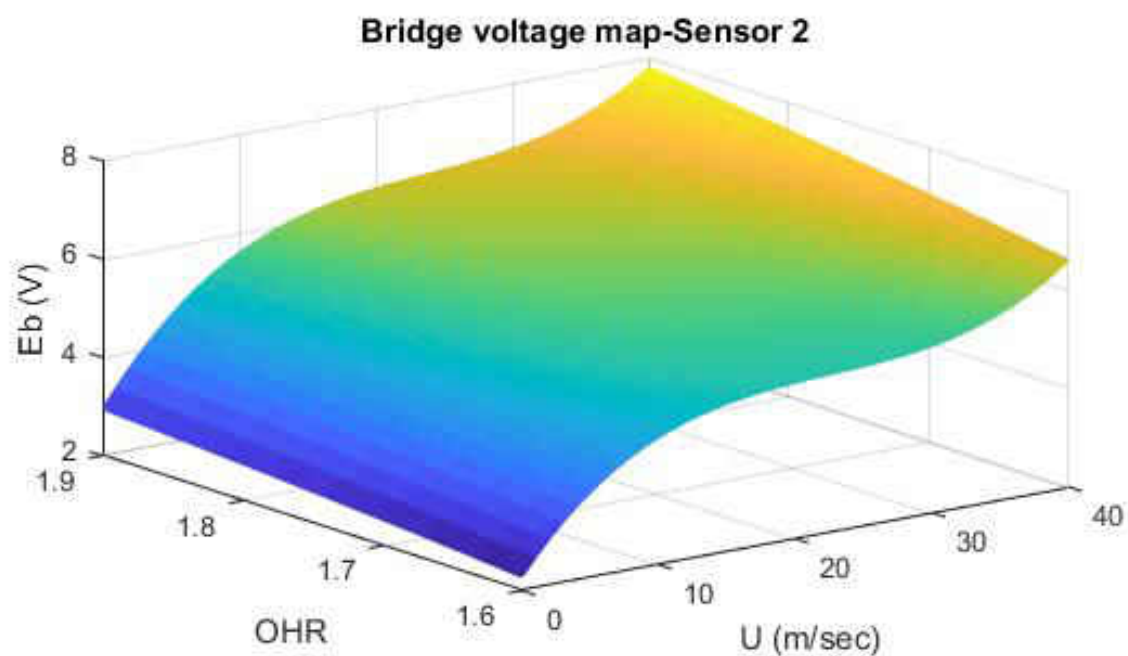
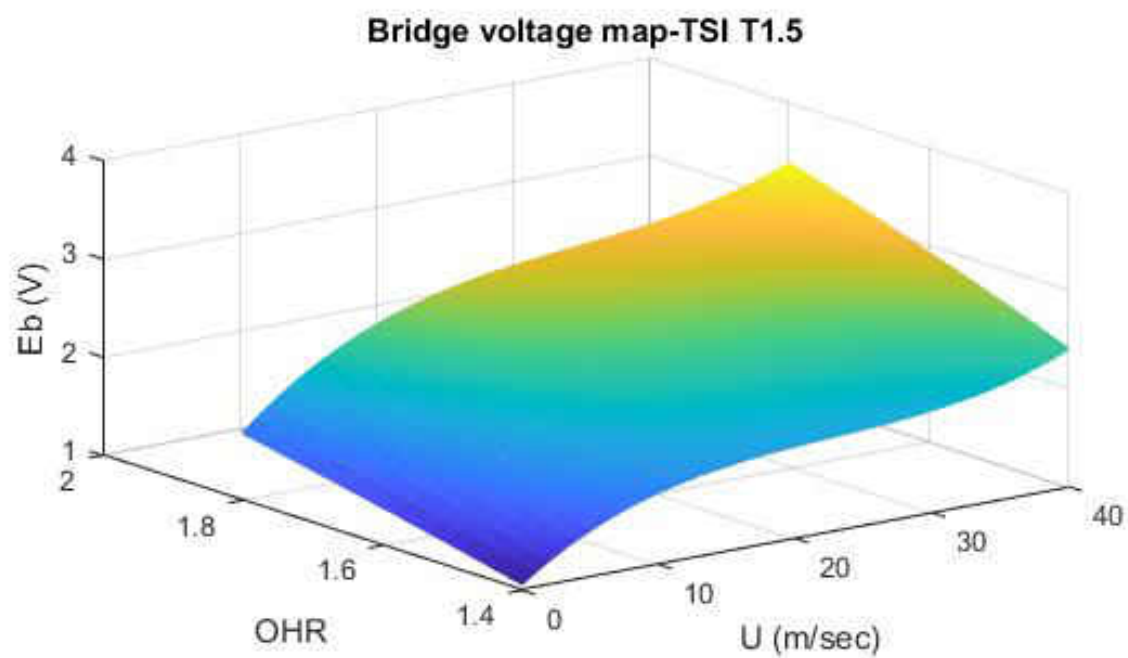


Figure 2.10: Bridge voltage map for different velocities and overheat ratios

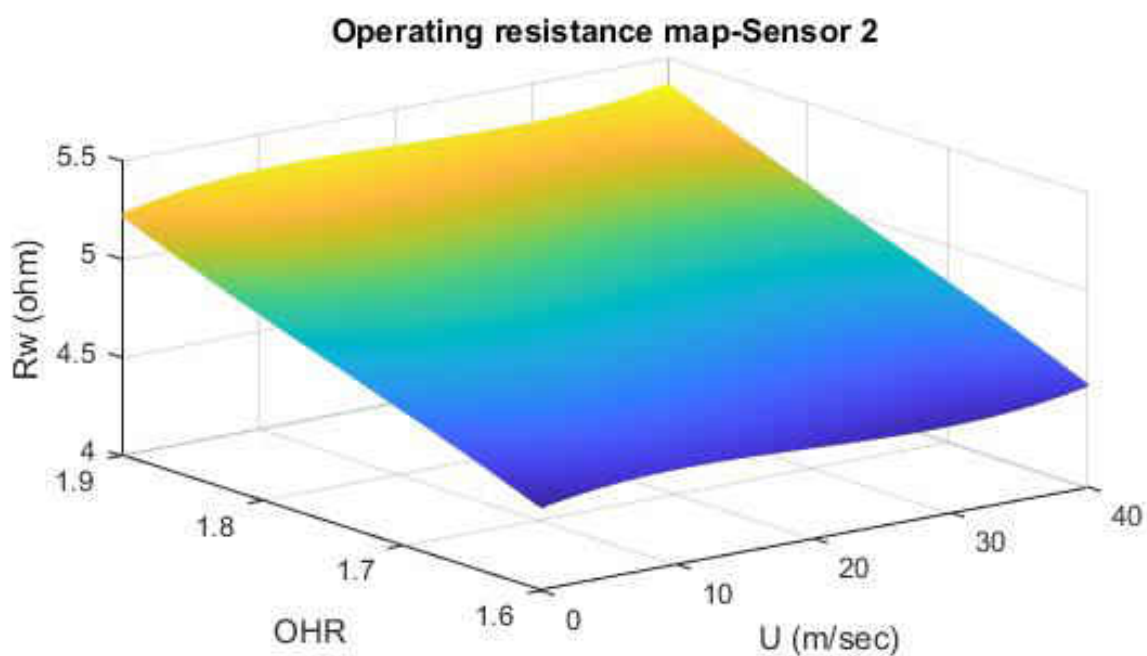
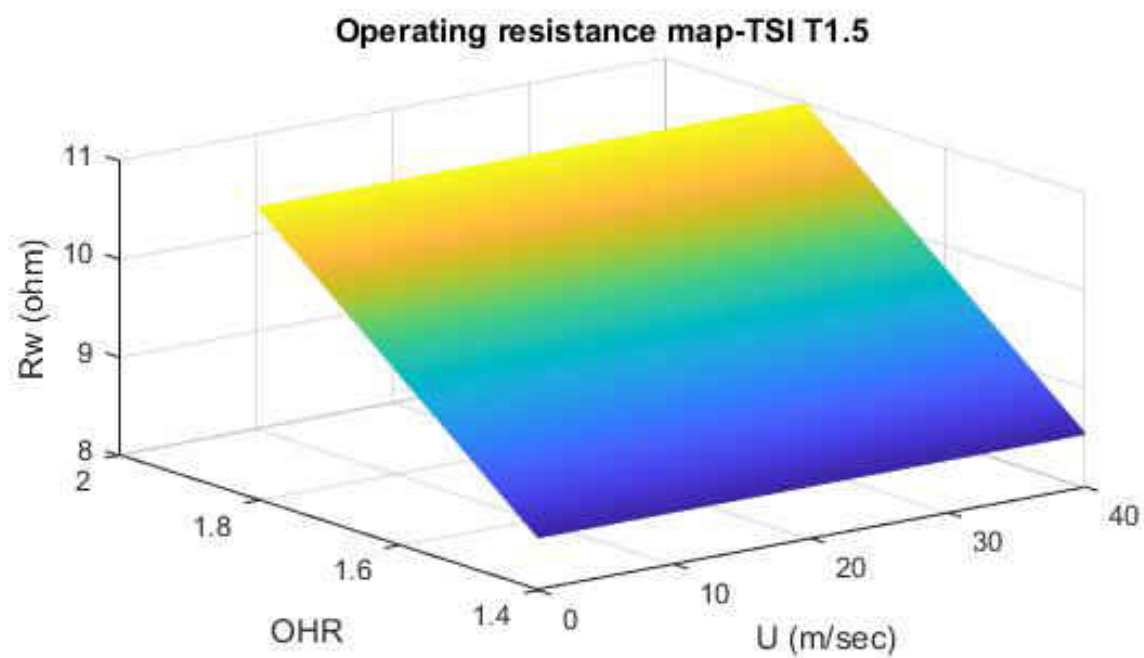


Figure 2.11: Sensor resistance map for different velocities and overheat ratios

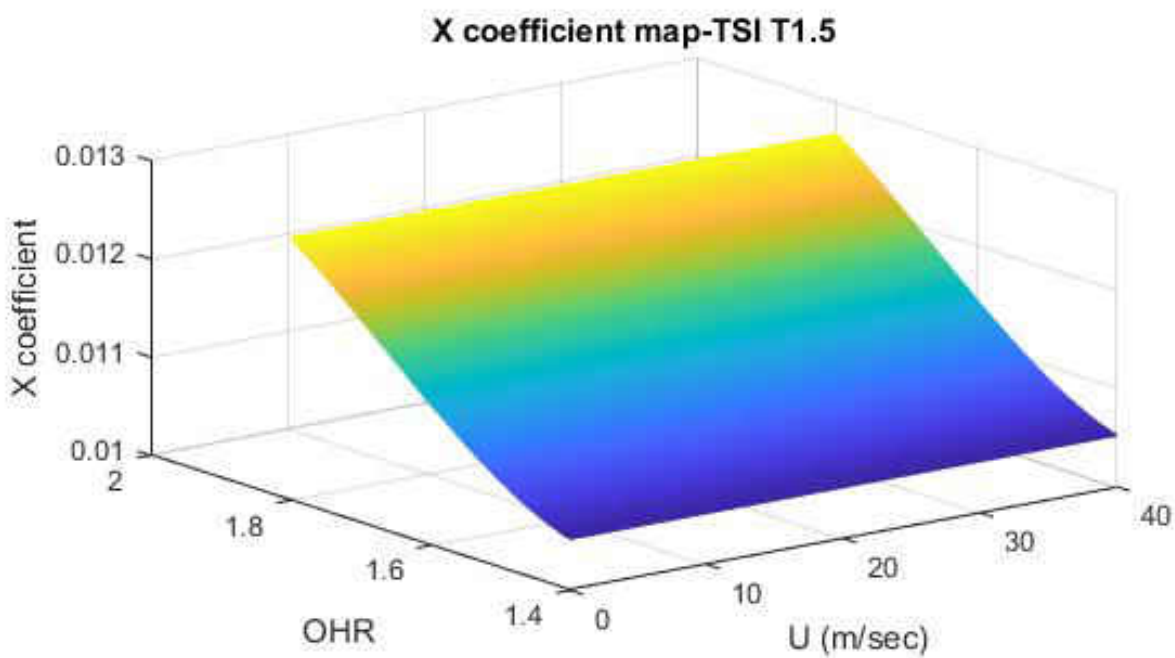


Figure 2.12: X coefficient map for different velocities and overhear ratios

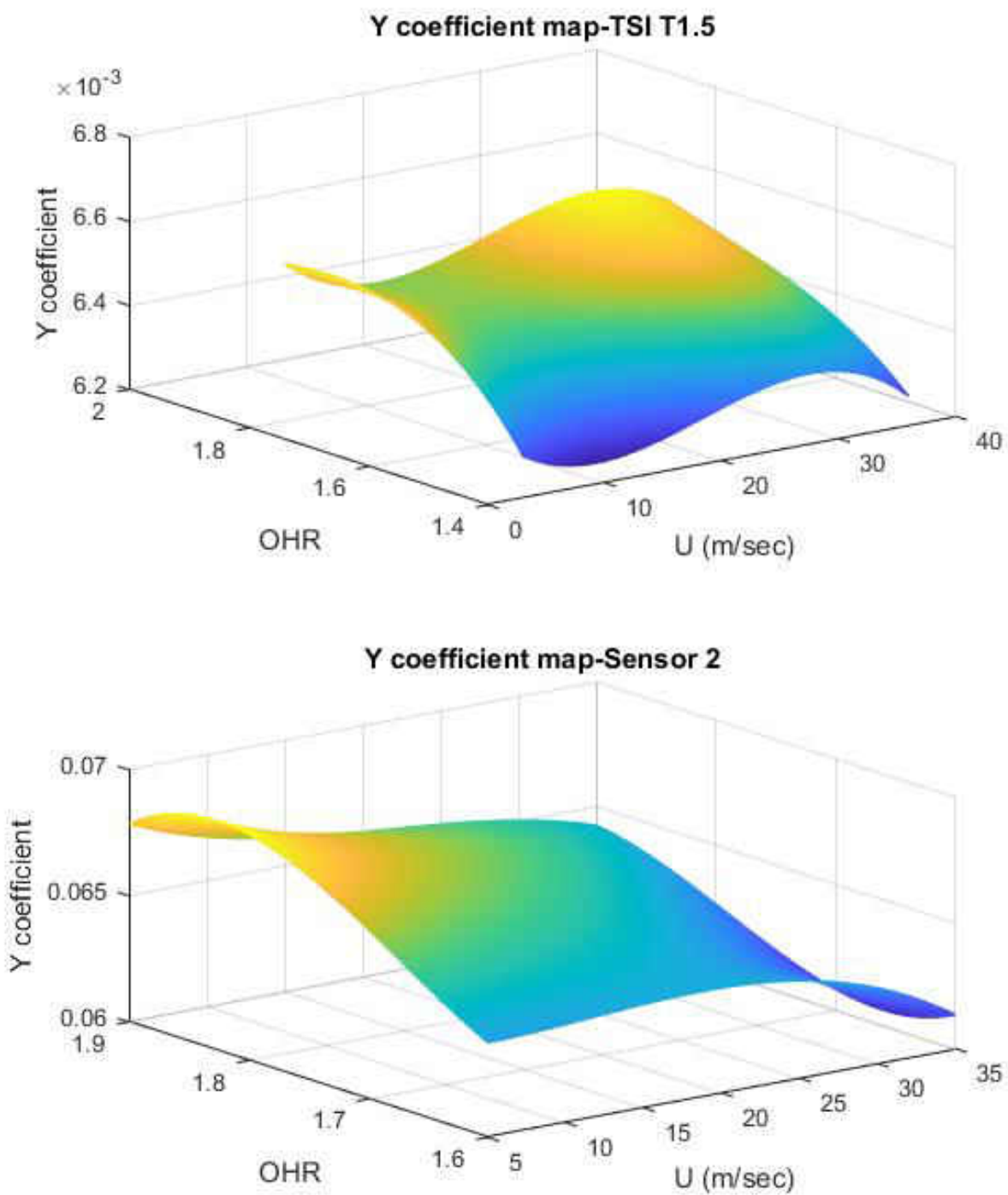


Figure 2.13: Y coefficient map for different velocities and overheat ratios

CHAPTER 3

Controller Design

3.1 Introduction

Constant corrections are needed to the bridge voltage being supplied in order to maintain the sensor resistance constant. This can be done by closed-loop feedback from the sensor to a controller which calculates and sends the corrected voltage to the bridge. A **Closed-loop Control system** also known as *feedback control system* is defined as a system with feedback of the output from the plant to the controller so that necessary corrections can be made to the input signal to the plant to obtain the desired output. Figure 3.1 [13] shows the block diagram representation of an open loop system and a closed loop Feedback system.

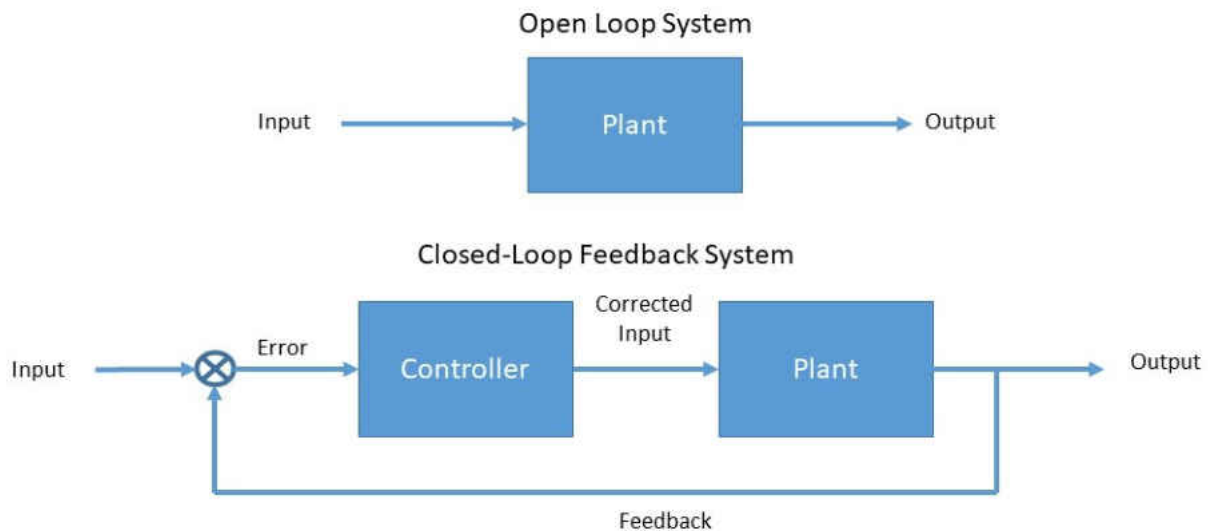


Figure 3.1: Open loop and closed loop feedback system representation

The open-loop plant model behaves differently for different circumstances, so the controller designed should be able to work for a wide range of operating points. The following design

is based on the identification of the plant using open loop step responses of the plant model. Design of controller is carried out in the continuous–time domain and then converted into the discrete–time domain using a bilinear transform.

3.2 Plant identification

The Open–loop plant identification is a process to identify the plant model i.e. the plant transfer function. This gives an insight of how the sensor responds to the given input step signal. This identification can be done in different ways: **step response, impulse response, wide pulse response and arbitrary I/O data.**

1. **Step response**, also called a **Bump test**, is a process where input to the plant is varied in fixed steps and the output is recorded as a response.
2. **Impulse response**, also called an **Impact test**, is a process where input to plant is varied as sudden but fixed impulse for a very short period and the response is recorded.
3. **Wide pulse** is an arbitrary rising and falling amplitude signal given as input to the plant with the response recorded.
4. **Arbitrary I/O data** is also called as an arbitrary signal given to plant with the response recorded.

Step response is the most effective and simplest of all the above–mentioned ways to find the plant model. The step response of the open loop plant i.e. Sensor 2 with properties given in Table 2.1 is taken in the continuous time domain and the data is imported into Matlab. Using *pidtool* in Matlab the plant is identified as shown in Figure 3.2. The plant is a first order model as given by the transfer function in Equation 2.19. Since the velocity cannot be controlled, the first part of the RHS of Equation 2.19 is considered as the plant model and the rest is considered as an external disturbance.

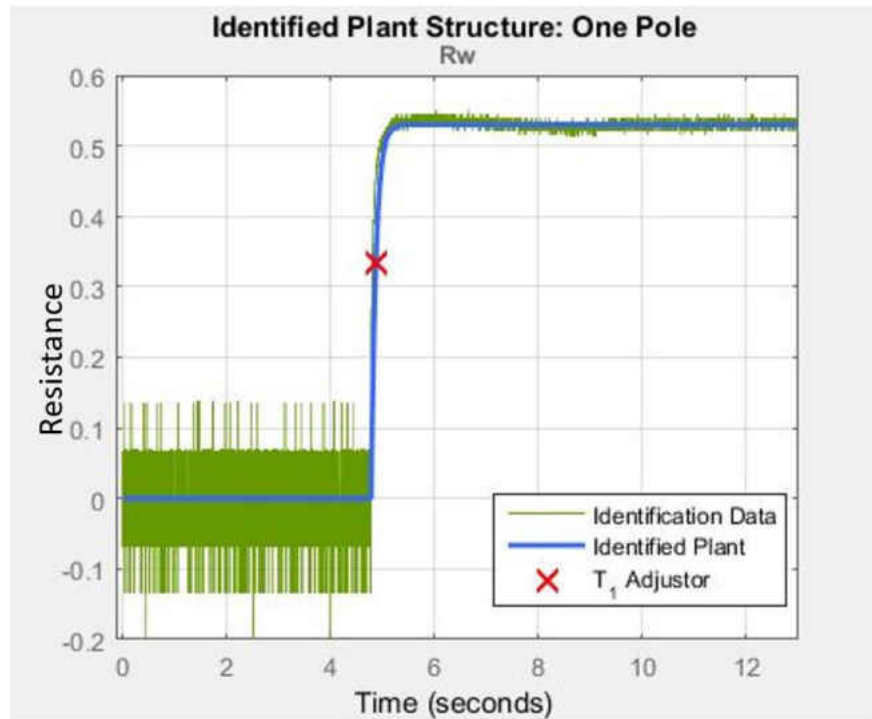


Figure 3.2: Identification of plant model using *pidtool*

The open loop plant transfer function is given as Equation 3.1

$$G(s) = \frac{0.67}{0.043s + 1} \quad (3.1)$$

The step response and bode plot of the open-loop transfer function of the plant model is shown in Figure 3.3 and Figure 3.4, respectively.

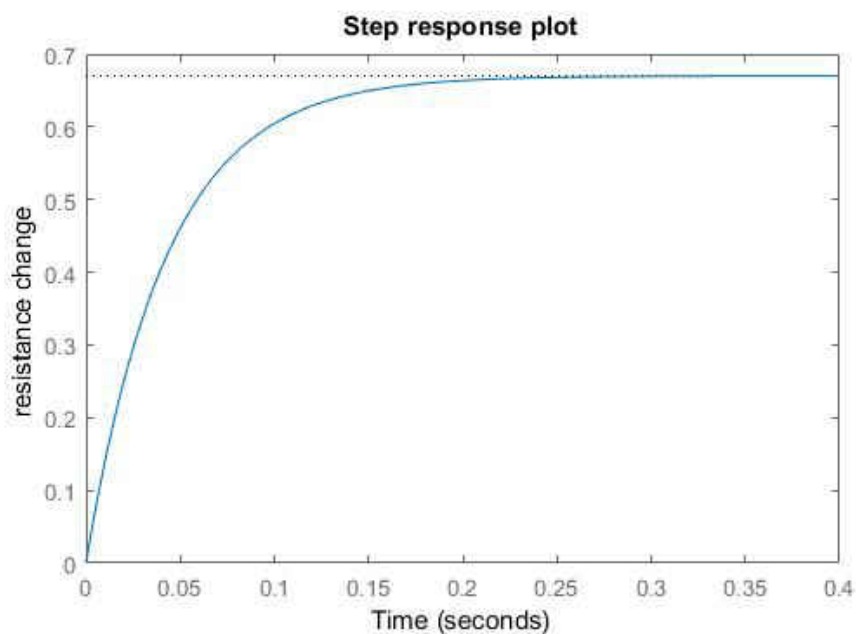


Figure 3.3: Step response plot of open-loop plant

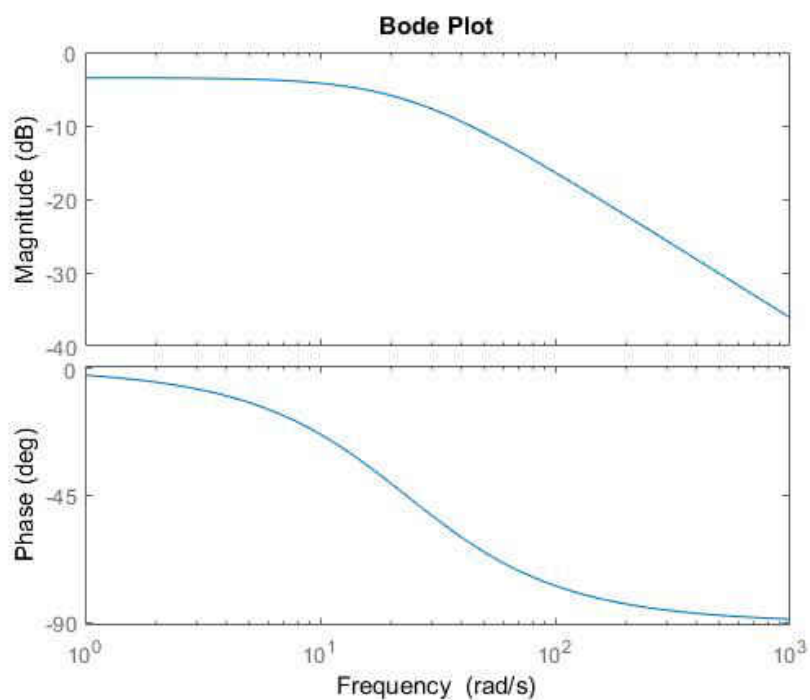


Figure 3.4: Bode plot of open-loop plant

3.3 Controller design

The objective of controller to be used in the DBTA is to control the input voltage to the bridge in order to keep the resistance of the sensor constant. So the job of a controller is to take the feedback of the shunt voltage and calculate the instantaneous resistance of the sensor using Equation 1.1, then compare the value to the target resistance. The difference is taken as error and a correction voltage is added to the previous voltage value and sent to the bridge. Different types of controllers can be employed based on the plant model. Since the present plant model is a first order system, a PID controller should be sufficient. The output of a PID controller, equal to control input to the plant, in the time-domain is given by Equation 3.2 and the typical unity feedback system is shown in Figure 3.5

$$E_b(t) = K_p e(t) + K_i \int e(t) dt + K_d \frac{de(t)}{dt} \quad (3.2)$$

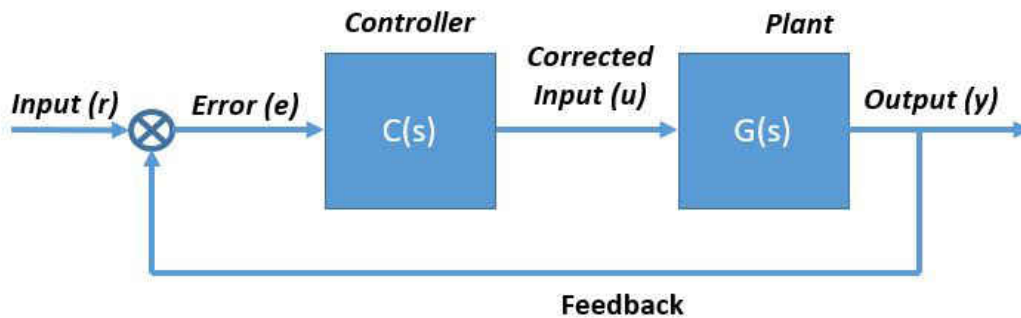


Figure 3.5: Unity feedback system

The variable (e) represents the tracking error, the difference between desired input and actual output. This error signal (e) will be sent to the PID controller and the controller computes both the derivative and integral of this signal. The control signal to the plant is equal to the proportional gain (K_p) times the magnitude of the error plus the integral gain (K_i) times the integral of the error plus the derivative gain (K_d) times the derivative of the

error. The transfer function of the PID controller is obtained using Laplace transform of Equation 3.2

$$K_p + \frac{K_i}{s} + K_d s = \frac{K_p s + K_i + K_d s^2}{s} \quad (3.3)$$

The chosen performance specifications of the closed loop system are:

1. *Rise time* ≤ 0.05 sec
2. *Settling time* ≤ 0.1 sec
3. *Maximum overshoot* $< 10\%$
4. *Steady state error* $< 2\%$

The PID controller can be sub-classified into 4 types

1. P-Controller or Proportional controller.
2. PI-Controller or Proportional-Integral controller.
3. PD-Controller or Proportional-Derivative controller.
4. PID-Controller or Proportional-Integral-Derivative controller.

3.4 P-Controller

The first thing to do is to find a closed-loop transfer function with a proportional control ($C(s) = K_p$) added. By reducing the unity feedback block diagram, the closed-loop transfer function with a proportional controller becomes:

$$Cl(s) = \frac{G(s)C(s)}{1 + G(s)C(s)} \quad (3.4)$$

i.e. from Equation 2.20:

$$Cl(s) = \frac{\alpha K_p}{T_w s + 1 + \alpha K_p} \quad (3.5)$$

From the above mentioned performance specifications the damping ratio and natural frequency are determined:

$$\omega_n \geq \frac{1.8}{T_r} \quad (3.6)$$

$$\zeta \geq \sqrt{\frac{\ln^2(M_p)}{\pi^2 + \ln^2(M_p)}} \quad (3.7)$$

where

- ω_n = Natural Frequency [rad/sec]
- ζ = Damping Ratio
- T_r = Rise time [sec]
- M_p = Maximum Overshoot

Therefore,

$$\omega_n = \frac{1.8}{0.05} = 36Hz \quad (3.8)$$

$$\zeta = \sqrt{\frac{\ln^2(10)}{\pi^2 + \ln^2(10)}} = 5.3 \quad (3.9)$$

The root locus of the plant with ω & ζ is shown in Figure 3.6

The two dotted straight lines in an angle to the real axis indicate the locations of constant damping ratio ($\zeta = 0.59$); the damping ratio is greater than 0.59 in between these lines and less than 0.59 outside the lines. The semi-ellipse indicates the locations of constant natural frequency ($\omega_n = 36$); the natural frequency is greater than 36 outside the semi-ellipse, and smaller than 36 inside. We can then find a gain to place the closed-loop poles in the desired region using *rlocfind*. The *red plus sign* in Figure 3.6 indicates this choice. The obtained point and loop gain are

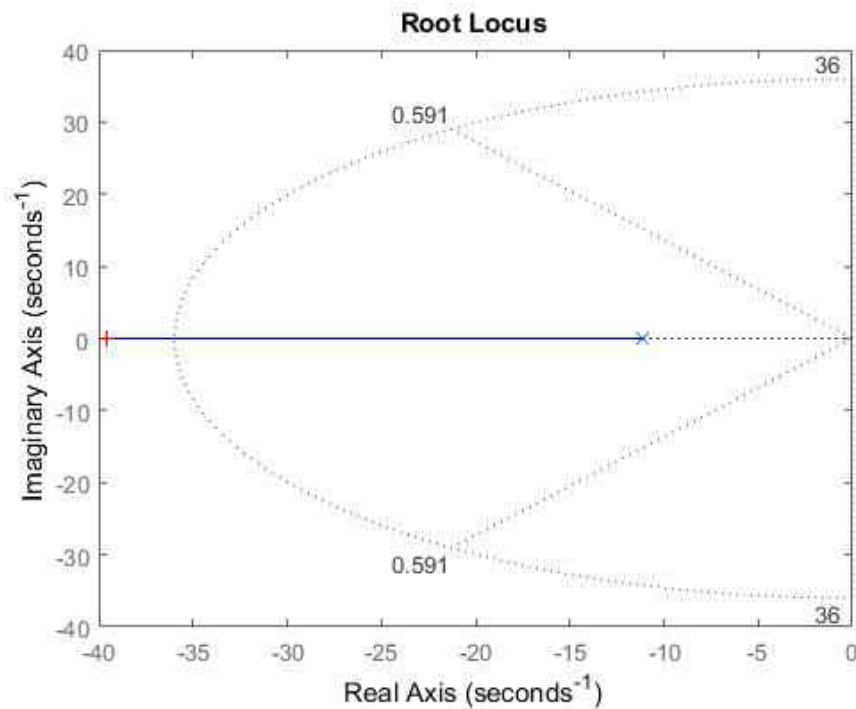


Figure 3.6: Root locus plot

- Selected point = -39.218
- $K_p = 1.018$

The closed-loop transfer function is given as

$$Cl_{plant} = \frac{0.6824}{0.043s + 1.682} \quad (3.10)$$

The step response of the closed loop system with the obtained K_p and step input of 0.001 is shown in Figure 3.7. Further examining the response by increasing the loop gain it is seen that regardless of the loop gain a steady state error always persists.

3.5 PI controller

It is evident that a proportional-only (P) controller cannot fulfill the requirements so PI (proportional-integral) controller is tested. The values of proportional and integral coeffi-

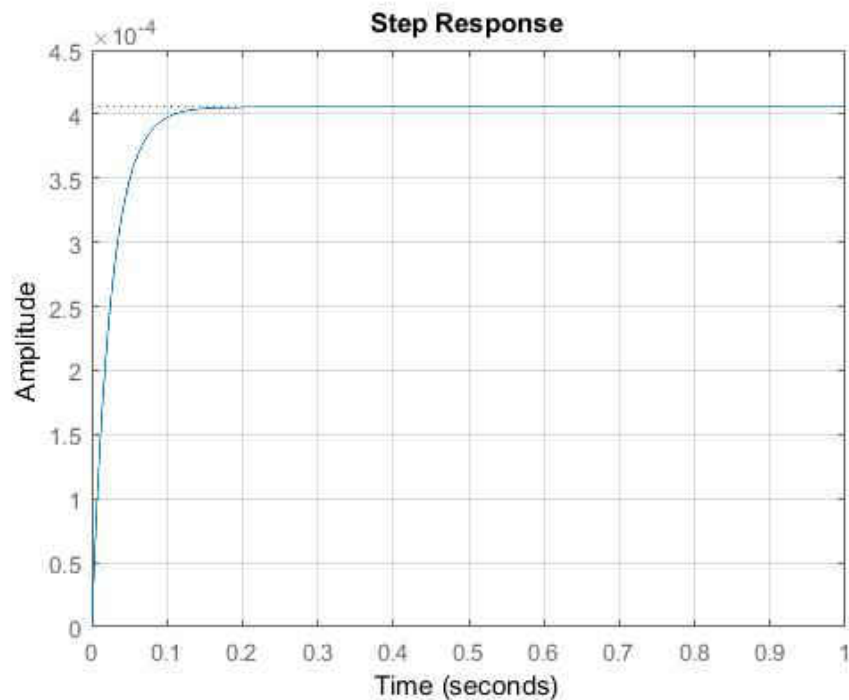


Figure 3.7: Closed loop step response

coefficients are optimized using **sisotool** in matlab. The root locus plot has design specifications such as percentage overshoot and settling time is shown in Figure 3.8. The angled lines represent the region with percentage OS of 5% and the vertical line indicates the settling time of 0.1sec. The zero must be placed somewhere on the negative x-axis since that is the region which yields the required specifications.

The step response and bode plot of the controller cascaded with the open-loop plant are shown in Figure 3.9 and Figure 3.10. The maximum control effort initially is less than the maximum limit of D/A i.e. 5V.

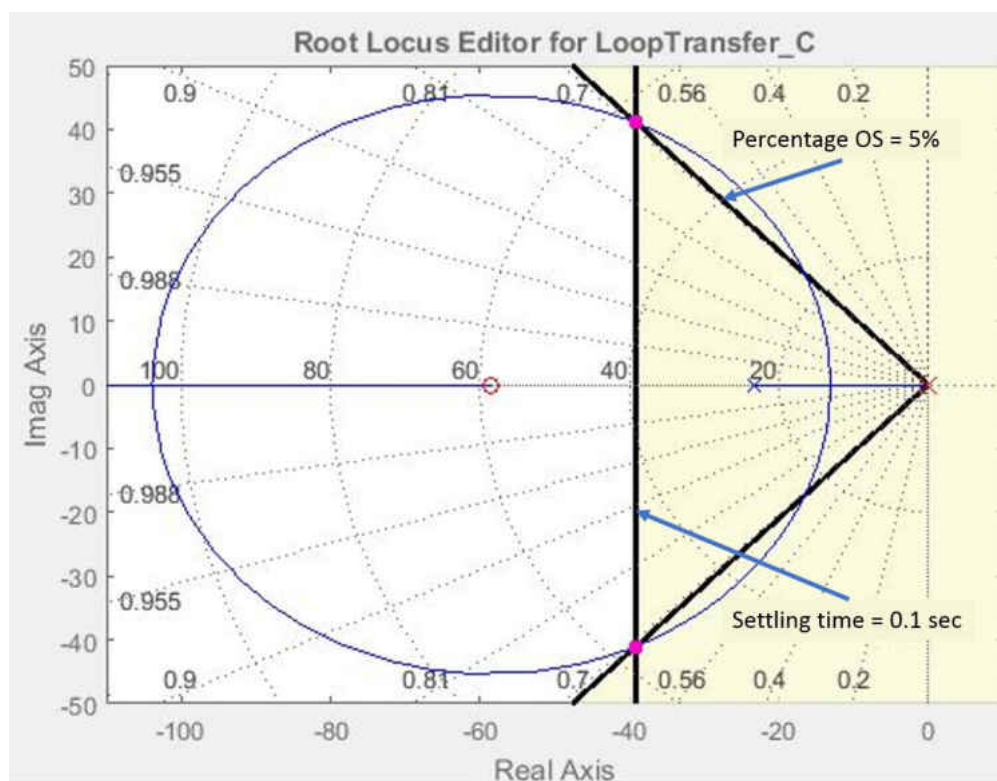


Figure 3.8: Root locus of the plant cascaded with PI controller

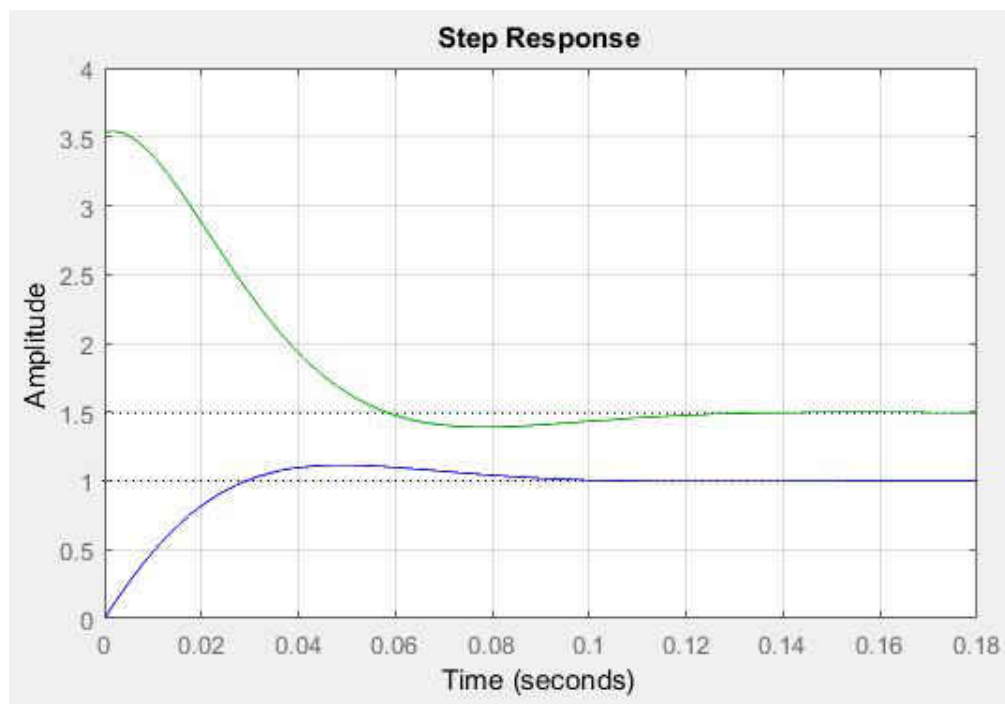


Figure 3.9: Step response and control effort (green) of controller cascaded with plant

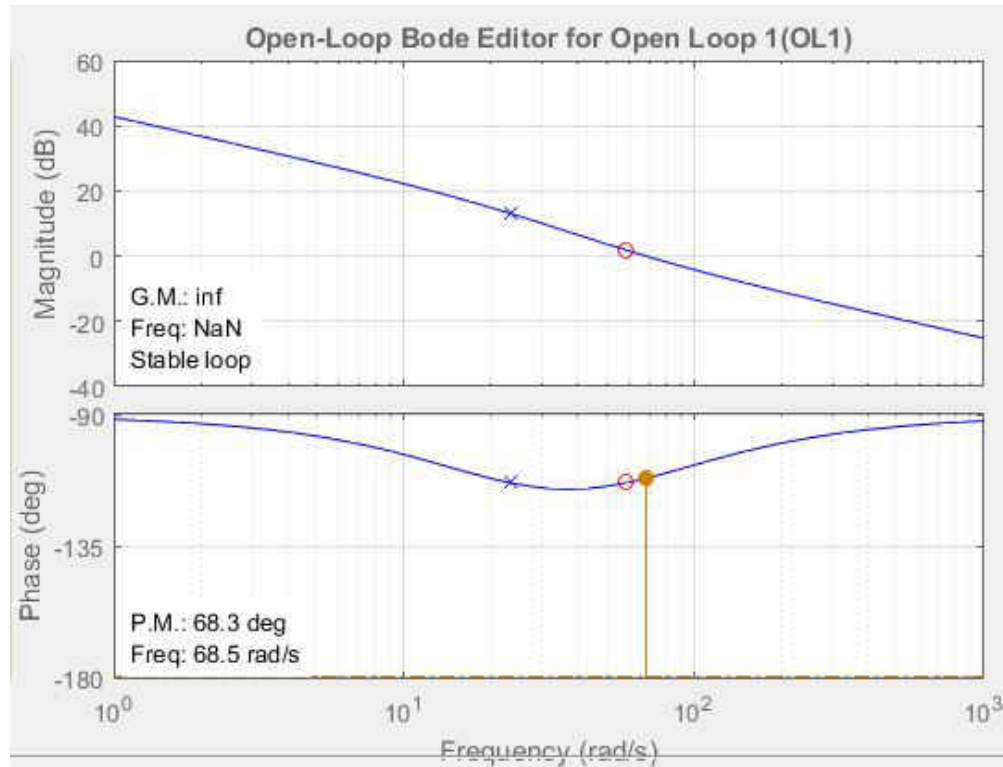


Figure 3.10: Bode plot of controller cascaded with plant

The PI type controller transfer function is given as:

$$C(s) = \frac{3.53s + 206.51}{s} \quad (3.11)$$

Since the PI type controller gives satisfactory results, it is not necessary to investigate PD and full PID controllers but is an option.

3.6 Converting the controller to discrete domain

The controller is converted from continuous time domain to discrete time domain using Bilinear Transform i.e. substituting

$$s = \frac{2z - 1}{Tz + 1}$$

where T is the sampling time

Considering $T = 0.001$

$$s = 2000 \frac{z-1}{z+1} \quad (3.12)$$

The controller in z domain is given as

$$C(z) = \frac{3.53 * 2000 \left(\frac{z-1}{z+1} \right) + 206.51}{2000 \left(\frac{z-1}{z+1} \right)} \quad (3.13)$$

Therefore,

$$C(z) = \frac{3.63z - 3.43}{z - 1} \quad (3.14)$$

The controller transfer function can be written as:

$$C(z) = \frac{e}{r} = \frac{3.63z - 3.43}{z - 1} \quad (3.15)$$

Therefore,

$$e = \frac{3.63z - 3.43}{z - 1} r \quad (3.16)$$

Solving the above equation:

$$ez - e = (3.63z - 3.43)r \quad (3.17)$$

The output voltage of the controller is the sum of the RHS terms in Equation 3.18.

$$e = (3.63 - 3.43z^{-1})r + ez^{-1} \quad (3.18)$$

The line diagram of the discrete time controller is therefore,

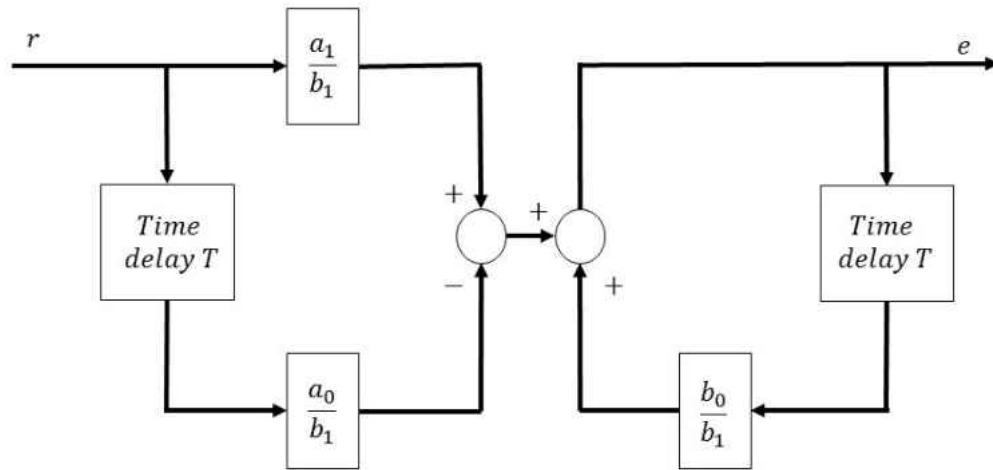


Figure 3.11: Discrete time controller line diagram

where the constants from Equation 3.14 are $a_1 = 3.63$; $a_0 = 3.43$; $b_0 = b_1 = 1$

CHAPTER 4

Graphical User Interface

4.1 Introduction

Effective user controls and the ability to view acquired data are fundamental requirements for any test, measurement, or control system such as a thermal anemometer. A user interface capable of communicating with the instrument in order to send commands and then collect and display the data to be analyzed by the user is a critical tool. The communication with the DBTA is done using a Graphical User Interface (*GUI* or *UI*), which has been developed using LabVIEW. Laboratory Virtual Instrument Engineering Workbench (LabVIEW) is a system-design platform and development environment for a visual programming language from National Instruments. There are two main windows which comprise the GUI i.e. the Front Panel and the Block Diagram.

4.1.1 Front Panel

The front panel window is the user interface for the VI (Virtual Instrument). In this window all the controls, indicators, and graphs are present through which the user can operate the VI. This panel of the VI is equivalent to a physical instrument in which the user can see and operate knobs, switches, and displays, etc. Figure 4.1 shows a typical front panel.

4.1.2 Block Diagram

The block diagram is where all the logic required to operate according to the programming and the front panel controls and display is present. The diagram consists of blocks that are executed according to control logic, following a data flow architecture. This is the process diagram not visible to the user when the VI is converted to an executable. It is equivalent

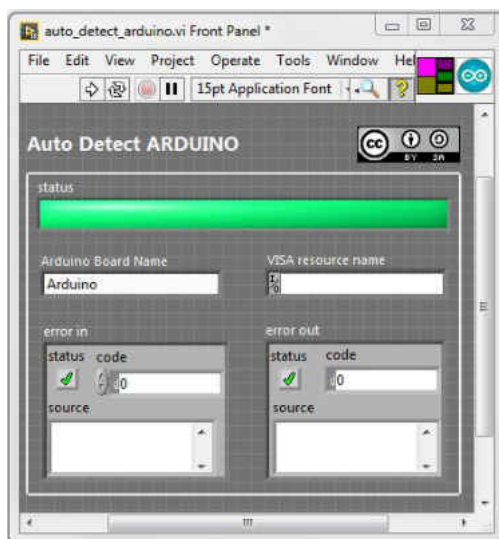


Figure 4.1: Example of front panel

to the circuits and electronic components in the physical instrument. Figure 4.2 shows a typical block diagram.

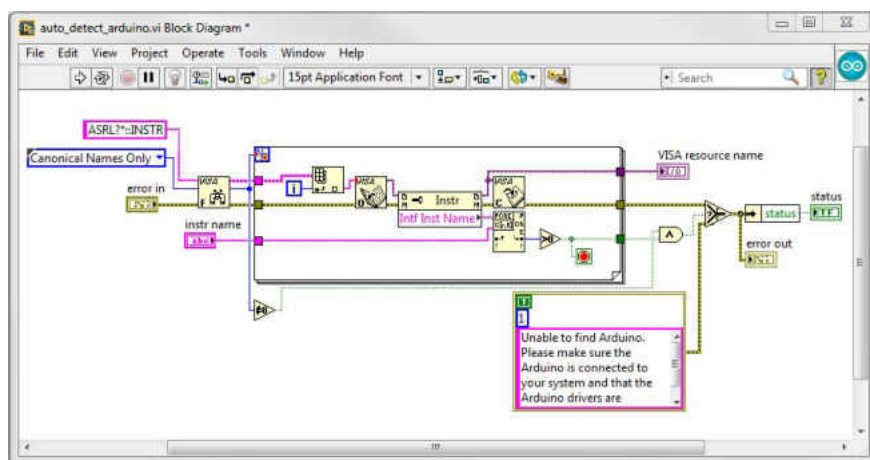


Figure 4.2: Example of block diagram

4.2 Important concepts of GUI for DBTA

The development of the GUI for DBTA is based on the TSI IFA300 Calibrator which is just a driver to start the IFA-300 [14]. The requirement of the GUI for the DBTA is to add

all the functionality to make it a stand-alone system to operate the DBTA, collect data, store, analyze, and display it. Initially the idea was to develop a State Machine based GUI, but due to the limitations of series operation of this method it was decided that it would not provide all the required functionality. The next option was to attempt a Queued Message Handler (QMH) based GUI. This provides parallel execution of the sub-VI's that can work independently to send commands, receive data, and display data. The following concepts are used in the QMH design:

1. While Loop
2. State Machine
3. Event Structure with While Loop

4.2.1 While loop

A while loop repeats the code within its subdiagram until a specific condition occurs i.e. to stop the loop. It executes at least once. It is similar to Do Loop or Repeat-Until Loop in text-based programming language. This loop plays a major role in the GUI design. This loop is used in all the other sub-VIs to keep them running. The iteration terminal provides the iteration count. The conditional terminal is the decision-making terminal which takes in a boolean input value (i.e. true or false) to control the loop execution. This input value is checked every iteration. The feedback of the previous loop is optionally carried to the present loop via shift register terminals.

In the present GUI design all major loops and sub-VI's utilize this structure. While loop is the driver of the state machines and event handing loop. Figure 4.3 shows the while loop with the components.

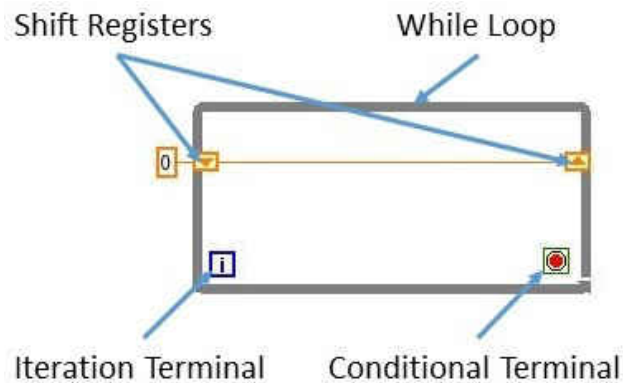


Figure 4.3: While loop diagram

4.2.2 State machine

A state machine is one of the fundamental architectures in LabVIEW. It is used in places where distinguishable states exist. Each state can lead to one or more states and can end the process flow. The operation of the state machine is based on user input or in-state calculation to determine the flow of execution of states in required order. The state machine starts with an *initialize* state followed by a default state i.e. *wait*. The *wait* state is the center of all states. All operations including the exit from the application state are channeled through the *wait* state are streamlined better. Finally the *exit* state is where all the cleanup actions are done to restore the default values to ensure a smooth start during next initiation.

A basic state machine can be created using a case structure inside a while loop. Shift registers along with the case statement are used to navigate along the cases. The while loop is the main program loop, which executes until a *true* command to stop the loop is executed. In each loop the input for the case structure, i.e. states, is checked. The default state is *wait* which is the center and in an idle condition it is executed. The following logic diagram shows the procedure.

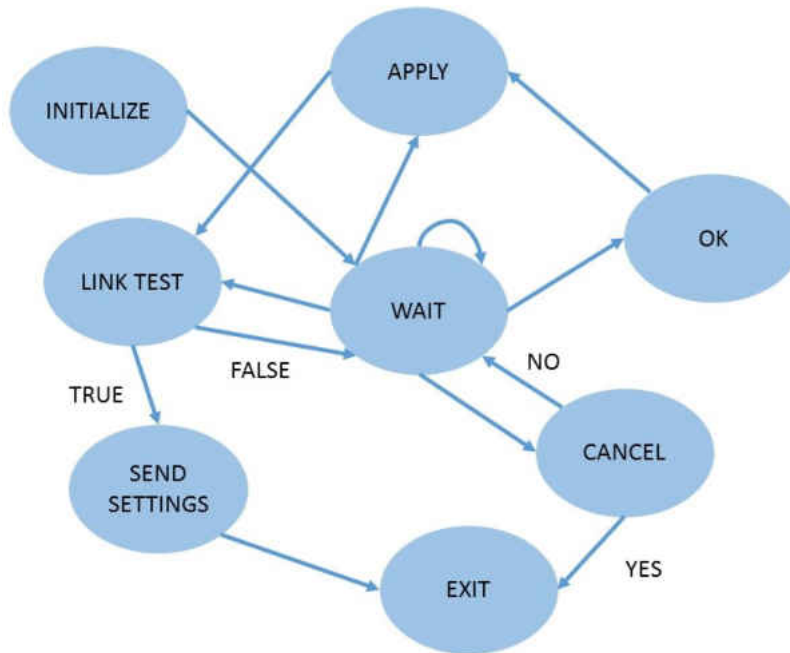


Figure 4.4: State machine line diagram

4.2.3 Event structure with while loop

Event structure with while loop is similar to the state machine but instead of a case structure there is an event structure inside the while loop. It is used where there is no dependency between two states. Thus when the user prompts for something using a switch or value change of an input, that particular event is triggered. The *idle* case is like the *timeout* where, in every iteration of the while loop, the event structure waits for the user input for a specified amount of time. If nothing happens then the while loop runs again. This structure greatly increases the efficiency of the program by implementing *timeout* so there is no need to go to a state and wait for the command. The event structure can take data and process it. This allows not only boolean values but actual change in values of numerical controls. This structure can keep track of the new value and old value, its source, the type of value and the time when it was changed last.

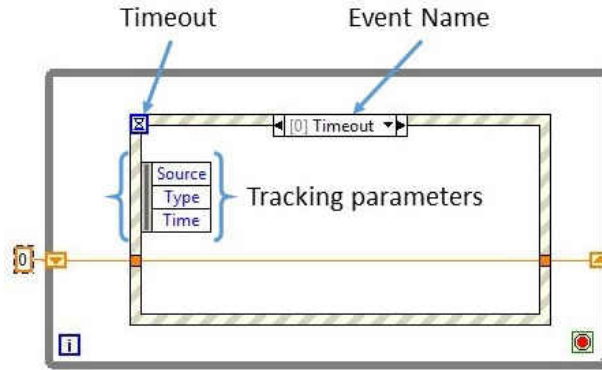


Figure 4.5: Event structure with while loop

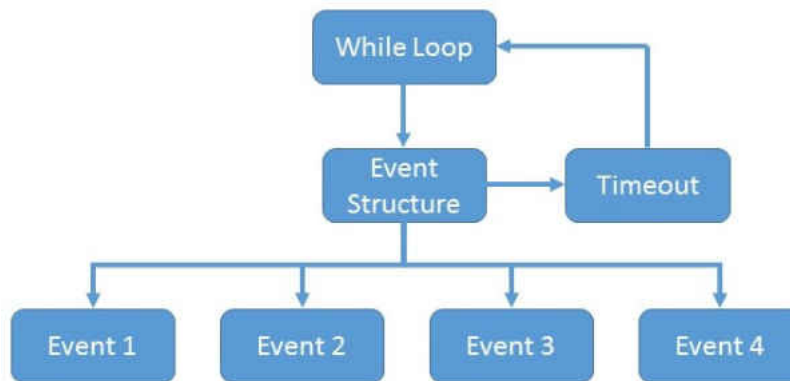


Figure 4.6: Line diagram

4.3 Development of the GUI for the DBTA

As stated earlier the graphical user interface for the DBTA is based on the Queued Message Handler (QMH). QMH is a parallel programming architecture where different Virtual Instruments (VI's) run in parallel. The instructions to these VI's are passed as messages. The DBTA UI consists mainly of an Event Handling Loop, a Message Handling Loop, and four Sub-VI's in the main VI i.e:

1. Calibration Loop
2. Acquisition Loop

3. Communication Loop

4. Data Logging and Display Loop

All the loops in the QMH follow the message flow sequence as shown in the Figure 4.7.

MAIN VI FLOW CHART

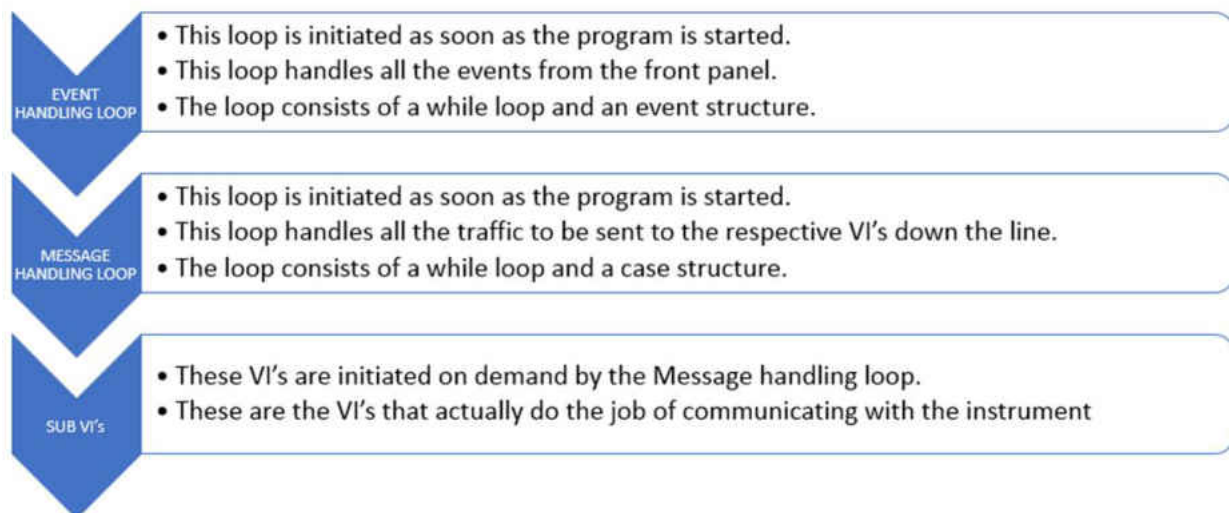


Figure 4.7: Queued message handler command flow

4.3.1 Event handling loop (EHL)

This loop is the first to respond to any changes in the front panel. Initially, the UI queue is loaded into the loop and is used to carry messages in the VI. The event structure waits for a specified amount of time and if no command is received the iteration is terminated and the next iteration of the while loop starts. Once there is any change of state or the value of any control in the front panel of the GUI, the respective event is triggered and the message is added to the UI queue. Figure 4.8 represents a typical event.

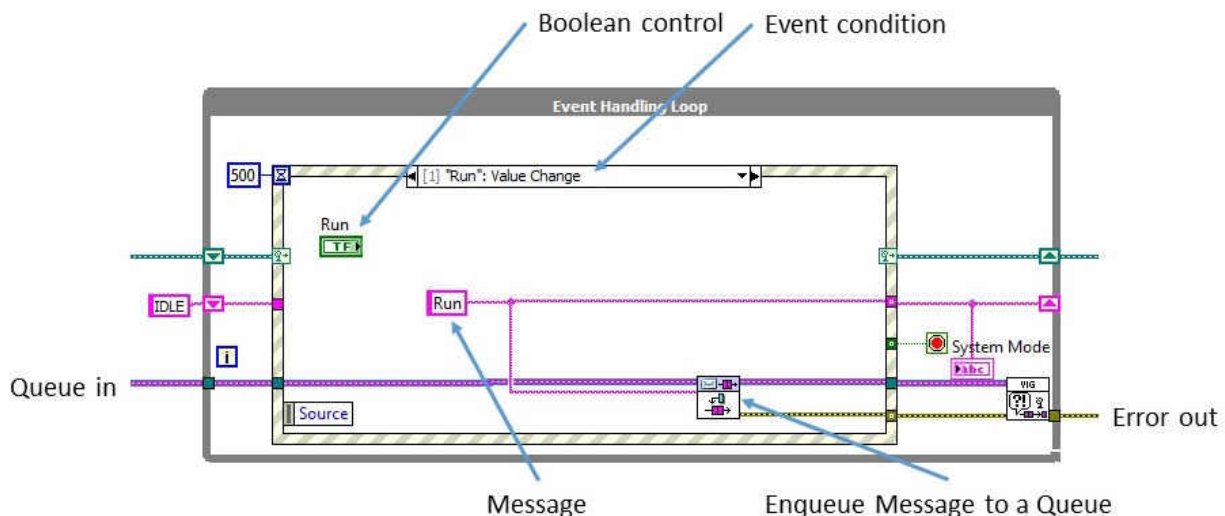


Figure 4.8: Example of event handling loop

The Enqueue Message VI queues the message of the event to the queue. The terminal description of the Enqueue VI is shown in Figure 4.9. The 'error out' terminal provides information about the functioning of the queue and VI so if there is any problem with queuing messages the error becomes positive and actions will be taken to either shutdown the program if either the error is potentially harmful in operation of the device or if it is an error that can be fixed without having to shutdown the program it will be fixed and error in will be reset.

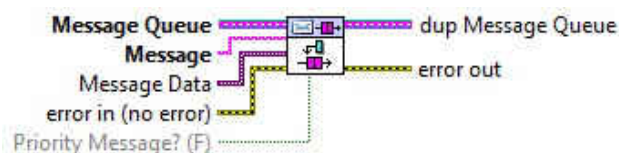


Figure 4.9: Enqueue VI terminal description

4.3.2 List of events

All the events are triggered by a value change in the front panel, some of them are boolean controls and some are actual value inputs. The following are the list of events handled by

the EHL.

Timeout : This event is executed when there is no input from the user in every loop.

Run : This event is used to send the command to DBTA to go to 'RUN' mode.

Start Cal : This event is used in calibration mode and to set the DBTA for calibration and start calibration loop.

Take Data Point : This event is used to take a data point during calibration mode.

Record : Even though the DBTA is running and streaming data real time, the data is recorded when this event is executed.

Load Cal File : This event is used to load a calibration file that was created and stored earlier.

New Cal File : This event is used to create a new calibration file. It is usually triggered when using a new sensor or an old sensor to recalibrate.

Standby : This event is used to command the DBTA to go into standby mode where the bridge circuit stops.

Reset : This event is used to reset the GUI and DBTA to default values. Its a failsafe as if something is entered wrong and that wrong entry cannot be found this boolean operator is used to reset everything.

Calibration Complete : During calibration mode user has option to go with fixed number of calibration points or can do calibration with limitless points so in the second case once the user is satisfied with the number of points this boolean operator is used to end the calibration.

Measure R : This event is used to command the DBTA to measure the cold resistance value of the sensor.

Settings : This event is used to activate 'settings.vi' to change settings for DBTA.

Exit : This event is used to exit the application this stops the DBTA and stops all the sub-VI's and exits the application.

Application Instance Close : This event is triggered when the user accidentally or intentionally clicks the close button. It initiates exit procedures described above and closes application.

4.3.3 Message handling loop (MHL)

This loop is the central command for receiving and sending messages. This is a 'State Machine'. All the messages generated by the EHL arrive and are dequeued by the 'Dequeue Message.vi'. The messages act as the input to the case structure. If the change in the front panel is not a boolean operator but is involved with a numerical value change, the new value is sent as 'message data' i.e. message followed by the new value. During timeout in the EHL a continuous 'wait' message is sent to the MHL to execute a wait state. All the other queues arrive here to collect the messages from the EHL. The failsafe system is similar to the EHL. This loop is the home for the 'Settings.vi' and initially at the start of the program default settings are loaded and when settings are changed they are updated. The state machine operation is similar to Figure 4.4 but with many states. The Message Handling Loop (MHL) with one of the states is shown in Figure 4.10

The queues are initiated at the start of the program using the 'Create Message Queues.vi'. Since all the queues are bundled, they are unbundled to individual queues as shown in Figure 4.10. All queues except the UI queue are used to send messages out of the MHL and the UI queue is the only queue that is used to collect messages from the EHL. 'Settings in' is a 'type def' that has all the predefined default settings. There are many states in the MHL but only those requiring detailed explanation are discussed here.

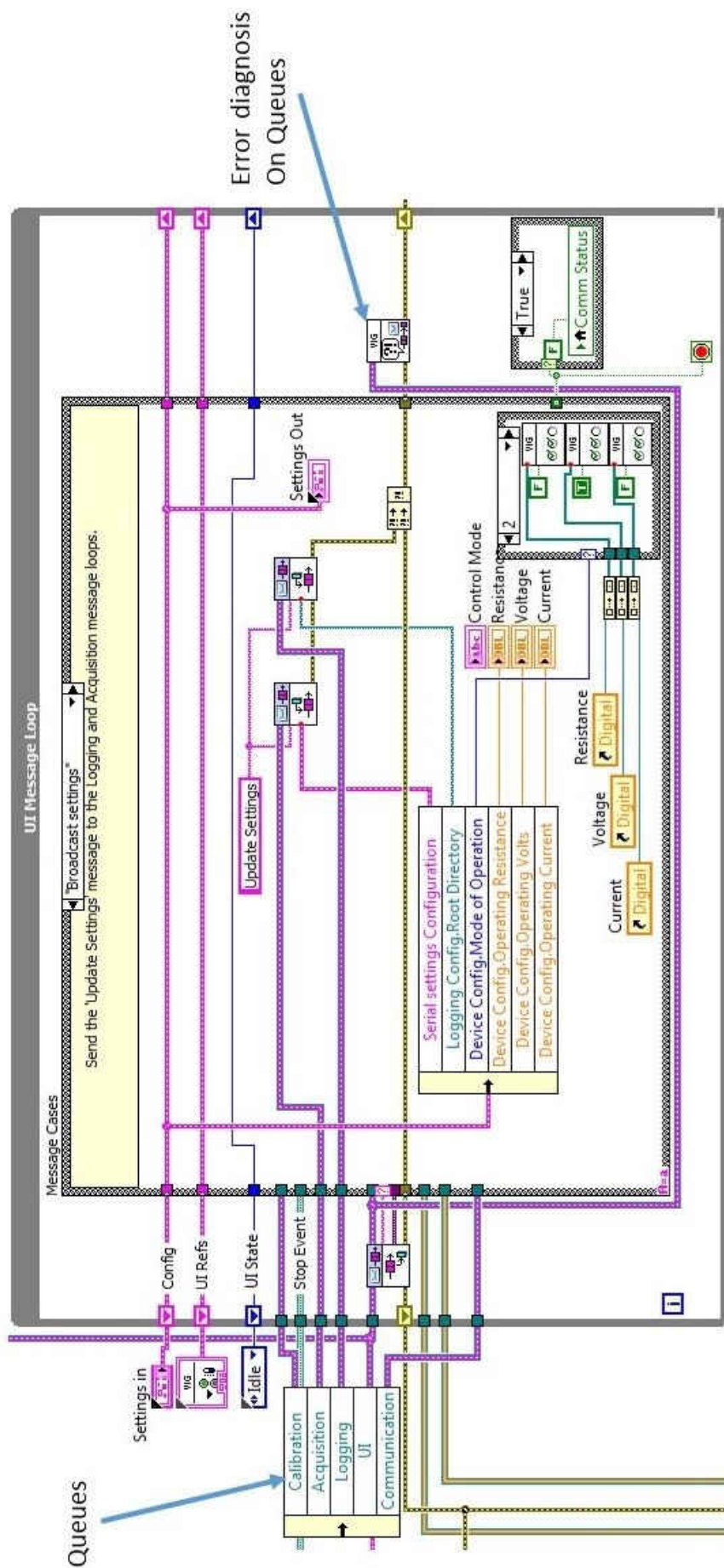


Figure 4.10: Example of message handling loop

4.3.4 Message handling loop states

Initialize : This state is the first to start when the application is initiated. It initially defaults everything for a safe operation with the default values for settings and ensures the DBTA is in standby.

Run : This state is initiated with the message 'Run' from the EHL. The message is passed to Comms.vi via Communication queue to send command to the DBTA to go into Run mode and is also passed to Acquisition.vi via Acquisition queue to start acquiring the data. Similar process is executed with Standby and Calibration messages except in Calibration state an additional message is sent to Calibration.vi via calibration queue.

Launch Settings : This state is initiated with the message received from the EHL where the value change in boolean settings control. This launches Settings.vi where user can change settings. More about this VI is discussed later.

Broadcast Settings : The new settings are obtained and are broad casted and stored in temporary memory to run the VI using these new settings.

Error : This state is initiated when there is any error occurred anywhere in the VI. The errors are compared to list of common errors and it will be fixed without stopping the GUI if its non-fetal.

Confirm Quit : This state is initiated when user tries to quit the program intentionally or accidentally. Its the final layer of protection from accidental quitting of program.

New Cal File : This state initiates a new calibration file setup in desired location. This pops a dialog to select the location and name of the new calibration file.

Load Cal File : This state is initiated with the user prompting to load a existing calibration file. This pops a dialog to select the desired calibration file.

4.3.5 Acquisition loop

This sub-VI is used to acquire data streamed by the DBTA for processing and real-time display. It operates via two modes i.e. Serial and Ethernet. There are two case structures in the while loop and the outer case structure is temporary which is used to select the mode of communication. Since the final mode of operation of the DBTA is not finalized (which would be via Ethernet), the Serial mode offers reduced speed to test the prototype firmware of the DBTA. This is a state machine. Messages are received through an acquisition queue. The data is transferred to the data display loop using a data queue and data notifier. The settings are updated in real-time through the broadcast settings state in the MHL. Figure 4.11 shows the block diagram of the acquisition loop.

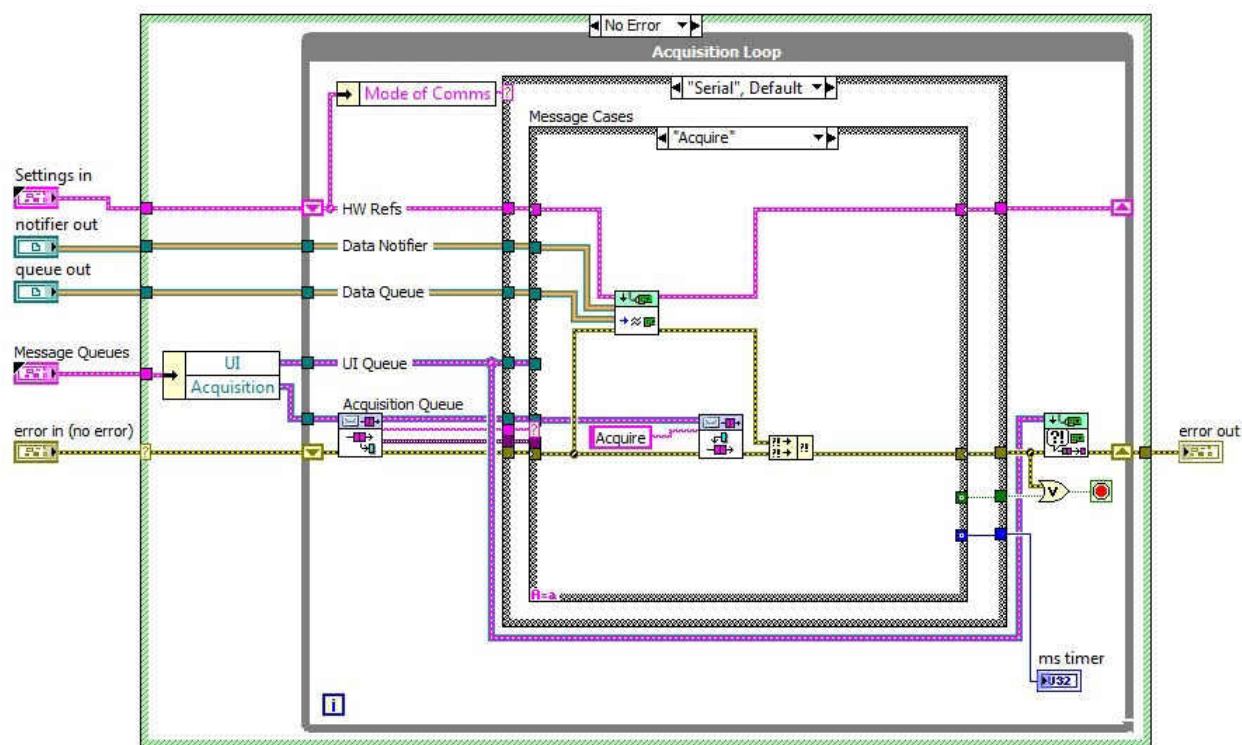


Figure 4.11: Acquisition loop

4.3.6 Communications loop

This sub-VI is used to send commands to DBTA and settings to DBTA. As the DBTA is planned to work in three different modes i.e. Constant Voltage (CV), Constant Current (CC) and Constant Temperature (CT), it is important that GUI supports the variations. Figure 4.12 shows the block diagram of the communications loop.

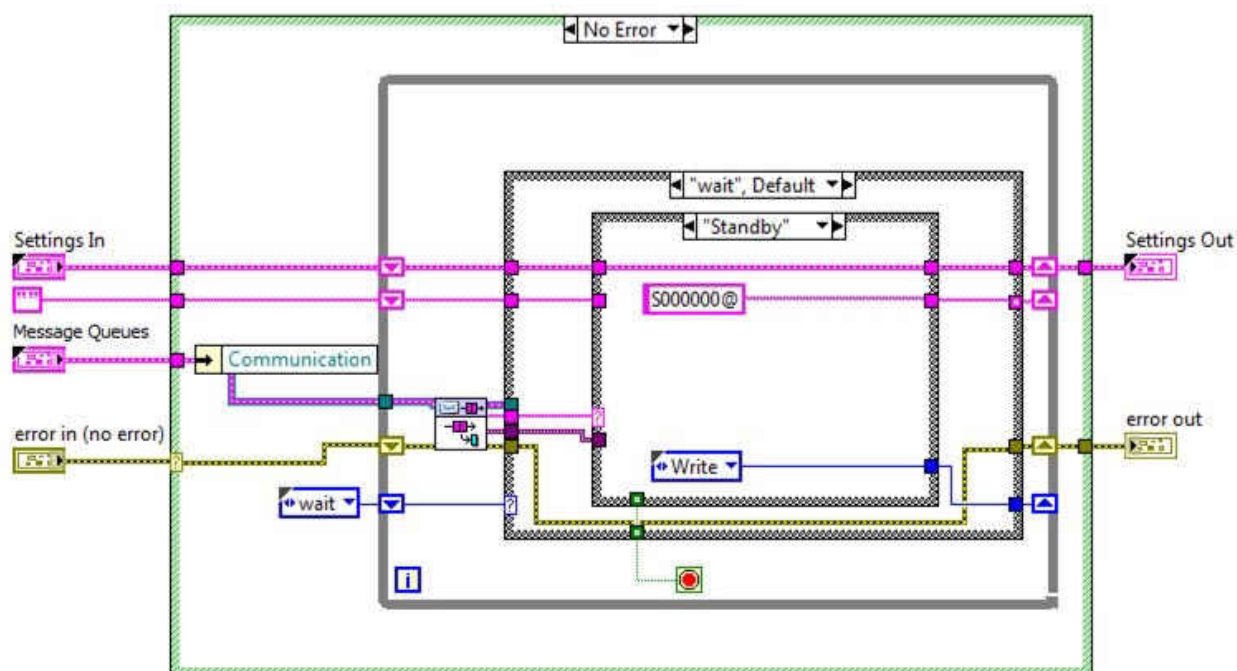


Figure 4.12: Communications loop

CHAPTER 5

Concept Validation and Results

The theory behind the digital control of the DBTA was validated by investigating the unsteady flow field and comparing the results with a commercial anemometer such as TSI IFA-300. Since the main purpose of the hot-wire anemometer is to measure turbulence in the flow, experiments such as the wake behind a cylinder and grid turbulence can be used for experiments. Given the limitation of speed in acquisition, experiment was chosen that give good results for frequencies below $500Hz$.

5.1 Experiment

The purpose of this experiment is to investigate the unsteady flow field behind a vertically-mounted cylinder. The intent is to acquire unsteady velocity measurements streamwise direction. Unsteady time history measurements, turbulence intensity and Reynolds stress can also be computed.

5.1.1 Test Setup

The test equipment consists of the high-speed test section of the ODU low-speed wind tunnel, NI LabVIEW data collecting system developed for Gen-I bridge, associated computers and probe-mounting hardware. A $3.5inch$ diameter cylinder was mounted vertically in the center of the test section upstream of a probe traverse system. The sensor was placed in such a way that the sensing element was approximately 5–6 diameters downstream of the cylinder. Additionally, wind-tunnel instrumentation was utilized to control and measure flow parameters. A schematic top view of the test setup is shown in Figure 5.1.

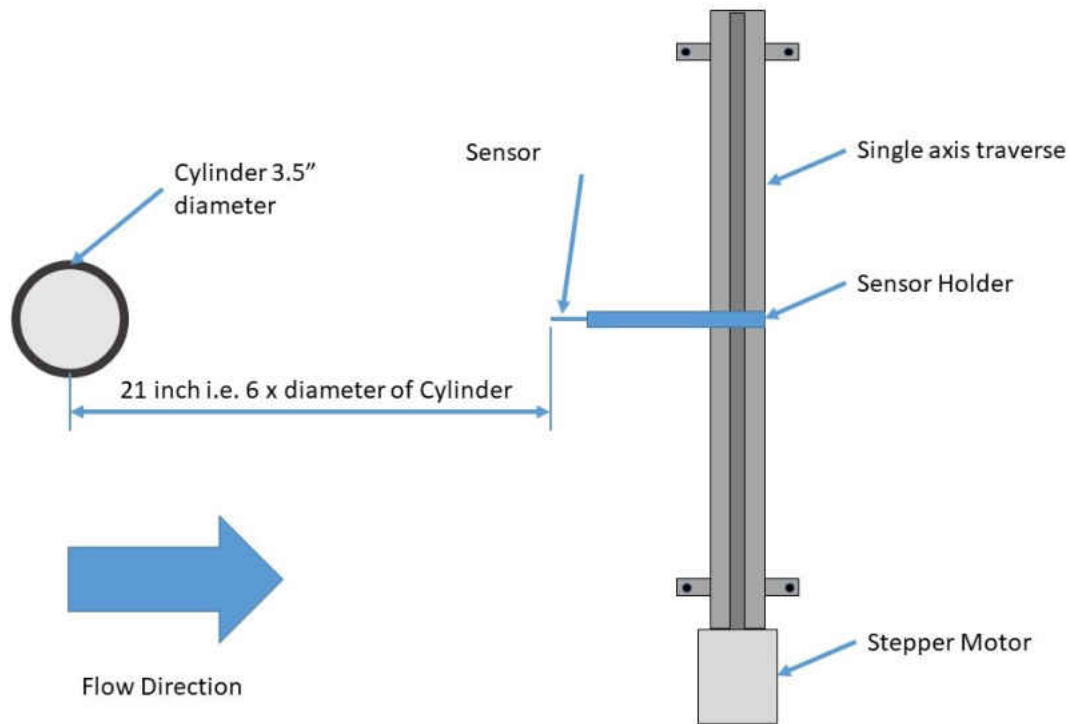


Figure 5.1: Top view of test setup

Data acquisition was done using TSI Thermal Pro for IFA-300 and program developed to run the Gen-I bridge was used for the DBTA. The frequency range of interest is up to 100Hz . Therefore, the hot-wire probe data were filtered using a low-pass 300Hz filter and sampled at a rate of 1kHz . The details of filtering in the IFA-300 are unknown (number of poles and the type of filter such as: Butterworth, Bessel etc...)

5.1.2 Procedure

The following steps are followed to repeat the data collection for a set of points:

1. The wind tunnel was brought to a steady condition of approximately 10.0 m/s .
2. The sensor was positioned to the left side of the cylinder at approximately 4 inches left of centerline.

3. Data were acquired at a rate of 1000 Hz for a period of approximately 10 seconds.
4. The probe was displaced in increments of 0.25 inches.
5. Upon reaching 4 inches of displacement to the right of centerline, the points are repeated and data is acquired until returning to 4 inches left of centerline.

Tunnel velocity was noted visually from the displayed instrumentation and typically fluctuated from 9.7 to 10.3 m/s with an average value of 10.0 m/s. Tunnel temperature was manually recorded for each acquisition period and fluctuated from 21.2 to 22.8 degrees Celsius with an average value of 22 degrees. Data were processed using the TSI system to create time history files with parameters of time, velocity for IFA-300 and microsoft excel was used to collect the time history files for DBTA. These data files were converted into Excel files for easy import to work in Matlab. Additional parameters such as dynamic pressure and Mach number were computed; and a statistical analysis was conducted to determine mean, standard deviation, and power-spectral density results.

5.2 Discussion of results

The mean velocity in the streamwise direction at both extremes of the sensor position is almost equal to the tunnel flow velocity and then starts decreasing towards the center and is lowest exactly behind the cylinder. The mean velocity versus the horizontal position is shown in Figure 5.2. In these figures negative horizontal position is left of the centerline when facing upstream and positive horizontal position is right of centerline when facing upstream.

Turbulence intensity can be calculated as the standard deviation of the velocity component normalized by the mean velocity in the streamwise direction at the point of measurement. MATLAB was used to determine the standard deviation of each velocity component, and the result was then normalized by the mean value of velocity at each point. The results are shown in Figure 5.3.

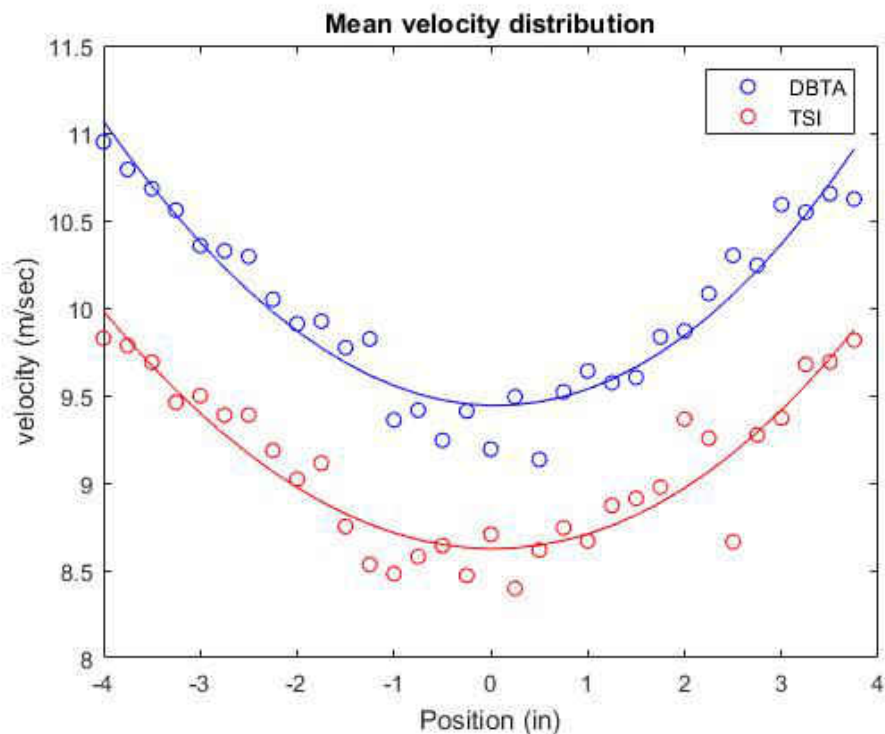


Figure 5.2: Velocity distribution along-wind

Power-spectral densities (PSD) were computed using the `pwelch` function in MATLAB. A `fft` size of 1,000 and window size of 1,000 were chosen. A Hamming window was used with 50% overlap and mean stripped prior to computing the PSD. The PSD in log-scale is shown in Figure 5.4. The frequency ranges from $10Hz$ to $100Hz$ and a linear scale was chosen to highlight the dominant frequency. The PSD values of the velocity for the dataset (4in left of center) is shown in Figure 5.5.

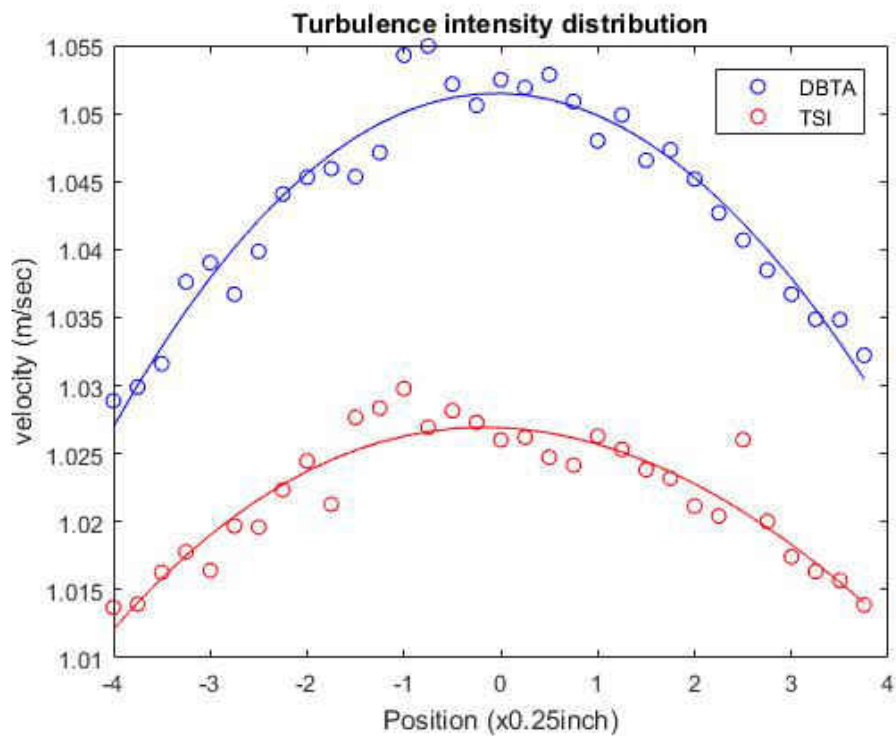


Figure 5.3: Turbulence intensity distribution along-wind

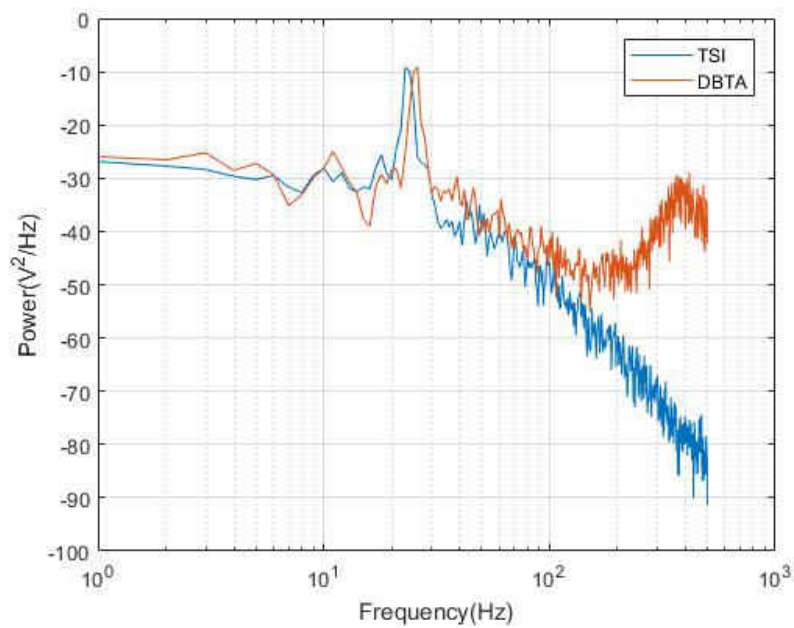


Figure 5.4: Power-spectral density (log-scale) @ -4in

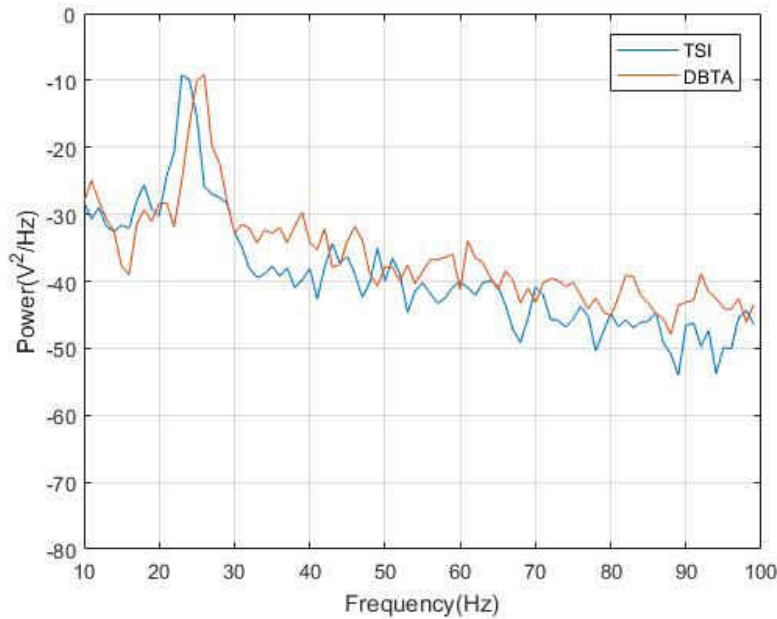


Figure 5.5: Power-spectral density (linear-scale) @ -4in

From inspection of Figure 5.5 it can be observed that the dominant peak PSD is roughly between $24 - 26\text{Hz}$ for both TSI and DBTA. At this sub-critical Reynolds number, the nominal Strouhal number characterizing von Karman vortex shedding to be 0.2. The equation of Strouhal number is expressed as

$$St = \frac{fD}{V} \quad (5.1)$$

where

- f = frequency (Hz)
- D = Diameter of cylinder (m)
- V = Local flow velocity (m/sec)

The velocity fluctuation frequency directly behind the cylinder should be double the vortex shedding frequency. Figure 5.7 shows the PSD results for both TSI and DBTA at the centerline position of sensor which is directly behind the cylinder in log-scale and a zoomed image in the frequency of interest is shown in Figure 5.7.

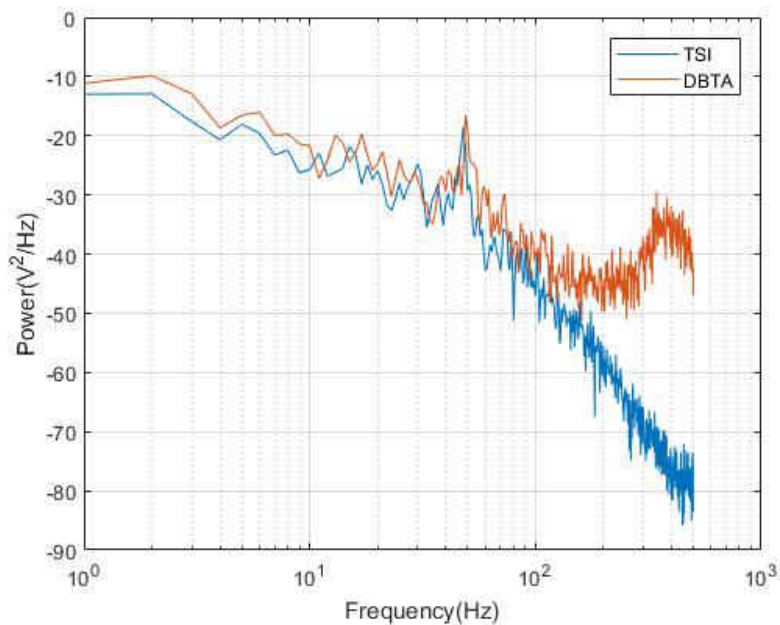


Figure 5.6: Power-spectral density @ 0in

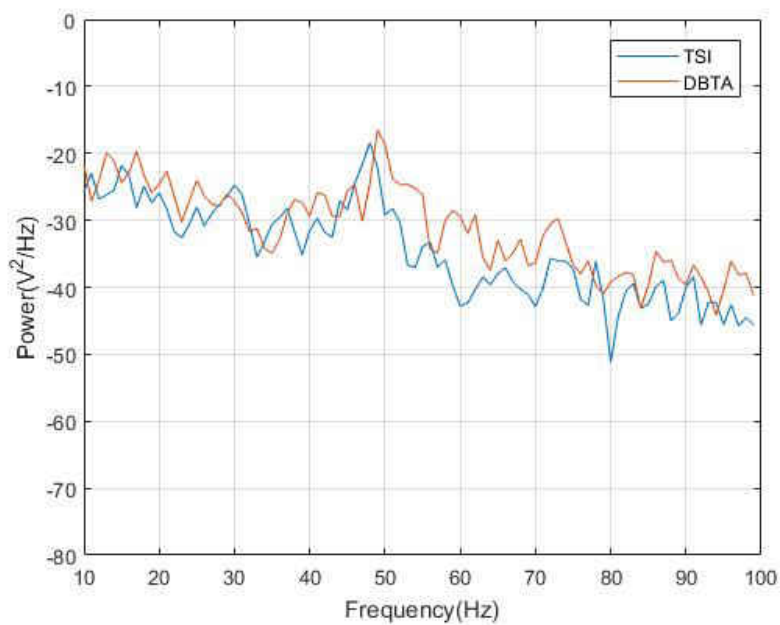


Figure 5.7: Power-spectral density @ 0in

The PSD across the length for both TSI and DBTA are shown in Figure 5.8 and Figure 5.9

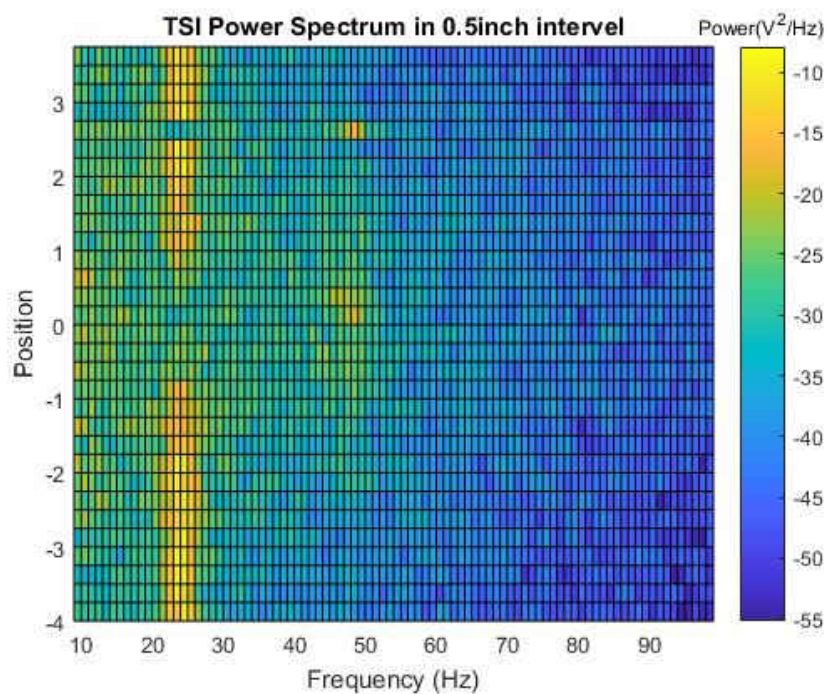


Figure 5.8: Power-spectral density

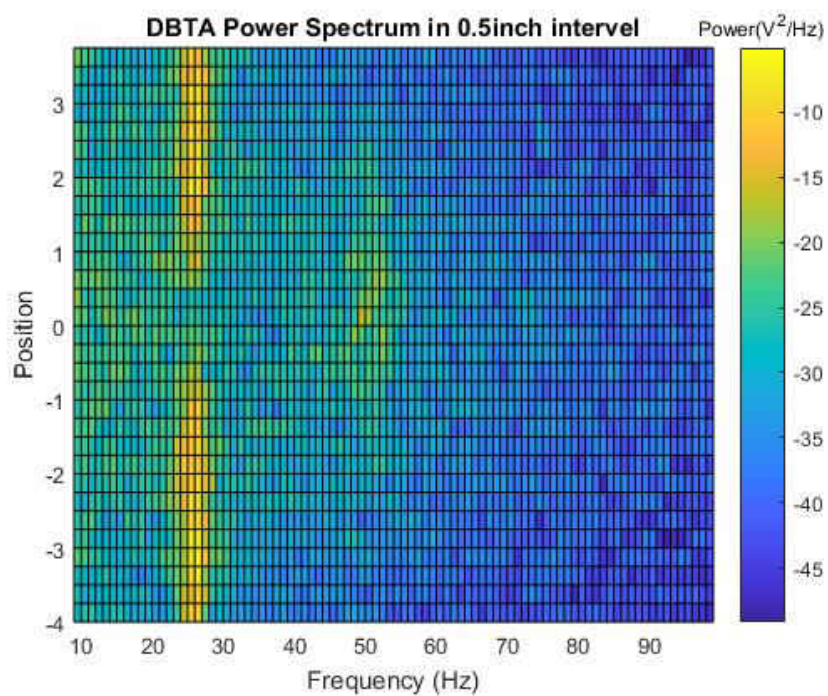


Figure 5.9: Power-spectral density

CHAPTER 6

Discussion and Conclusion

6.1 Discussion

The Digital Bridge Thermal Anemometer (DBTA) is a new generation anemometer that is constructed using a half-Wheatstone bridge configuration (i.e. the sensing element and a shunt resistor connected in series). A conventional hot wire anemometer takes feedback from the imbalance in the bridge to maintain the resistance of the sensing element constant, the DBTA takes the feedback of the current flowing through the bridge which can be used to calculate the instantaneous resistance of the sensor and supply the required voltage needed to maintain the resistance of the sensing element. Unlike a conventional anemometer where the basic closed-loop configuration of the circuit needs to be changed to change the working mode of the anemometer, in the DBTA, adjusting the feedback algorithm is sufficient to operate the anemometer with different sensors and in different operating modes.

In Chapter 1 an overview of the anemometers, the use of hot-wire anemometers in quantifying the flow characteristics using different modes of operation were discussed, along with the advantages and disadvantages of conventional hot-wire anemometer. The need for a new generation anemometer to overcome the disadvantages is addressed. Chapter 2 discusses different types of sensing elements that are used. Using King's Law, a heat transfer relation was established and new transfer function for the open-loop bridge was derived theoretically for the new bridge configuration. Quantifying the open-loop plant model both theoretically and experimentally. Due to the limitation to work with DBTA as it's in development, generation-I bridge which is just the bridge and a power amplifier is used with external data acquisition system for experimenting. This setup was tested with traditional sensors, but due to low heat capacity of the sensors which are designed to run at high speeds and the system

being limited in terms of loop speed, the sensors burnt up with over and under compensation of the correction voltage. The system often went to oscillations between the limits of A/D converter. A custom-built sensor was designed that runs with the available test setup. This sensor was thicker diameter and longer length thus having higher heat capacity. A modified version of the user interface was also developed to work with the test setup. Controller design was based on the open-loop plant model from Chapter 3 using PID techniques. The controller was then converted to discrete time domain and implemented. The concept of digital control of the DBTA was proved by investigating the unsteady flow field using both a generation-I bridge with a custom sensor and an IFA-300 with a TSI T1.5 wire type sensor. The task was to analyze the unsteady flow field behind a vertically mounted cylinder in the flow. Further development of the feedback algorithms will be needed, in order to improve the performance at higher sample rates and to incorporate different operating modes, such as CCA and CVA.

The new anemometer requires a user interface (UI) to control the anemometer and to analyze data discussed in Chapter 4. The user interface was developed in LabVIEW using the concept of queued message handler (QMH) and state machine. This user interface was capable of handling the incoming data and outgoing commands simultaneously while processing, presenting and storing the data obtained. This UI was successfully tested with DBTA.

6.2 Conclusion

The concept of the digital bridge for the anemometer is truly ground breaking given the fact that very little has changed in the past 11 decades. Even though the concept of digital anemometer such as pulse-width modulated (PWM) anemometers is new, the basic bridge configuration has changed very little. The redesigned bridge with integrated electronics has the capability to run in all the modes with wide range of sensors selection. The modified dynamic model of the DBTA configuration was adequately validated by simulation and experiment. The proof-of-concept anemometer functioned satisfactorily, although opportunities for testing were limited. The reduced sample rate and extended time constant inevitably limited the achievable bandwidth. The power spectra of the flow in the wake of a 2D circular cylinder showed adequate agreement between the DBTA and a commercial anemometer/sensor. The user interface is functioning but will need further debugging and development as the functionalities of the DBTA are improved.

Overall, the results indicate that the DBTA concept is valid and shows potential for a new-generation anemometer with improved practicality, versatility and overall performance, especially in extreme environments.

APPENDIX A

Theoretical Calculation

```

1 clear all
2 close all
3 clc
4
5 %%%%%%%%%%%%%%% Temperature Related Variables %%%%%%%%%%%%%%%
6 data=[ -150      2.793    1.026    0.0116    3.08     8.21     0.760;
7        -100      1.980    1.009    0.016     5.95     5.82     0.740;
8         -50      1.534    1.005    0.0204    9.55     4.51     0.725;
9          0       1.293    1.005    0.0243   13.30     3.67     0.715;
10         20       1.205    1.005    0.0257   15.11     3.43     0.713;
11         40       1.127    1.005    0.0271   16.97     3.20     0.711;
12         60       1.067    1.009    0.0285   18.90     3.00     0.709;
13         80       1.000    1.009    0.0299   20.94     2.83     0.708;
14        100       0.946    1.009    0.0314   23.06     2.68     0.703;
15        120       0.898    1.013    0.0328   25.23     2.55     0.700;
16        140       0.854    1.013    0.0343   27.55     2.43     0.695;
17        160       0.815    1.017    0.0358   29.85     2.32     0.690;
18        180       0.779    1.022    0.0372   32.29     2.21     0.690;
19        200       0.746    1.026    0.0386   34.63     2.11     0.685;
20        250       0.675    1.034    0.0421   41.17     1.91     0.680;
21        300       0.616    1.047    0.0454   47.85     1.75     0.680;
22        350       0.566    1.055    0.0485   55.05     1.61     0.680;
23        400       0.524    1.068    0.0515   62.53     1.49     0.680];
24 %%%%%%%%%%%%%%%
25
26
27 Tair=data(:,1);           % Temperature of air degC
28 TairK=Tair+273.15;       % Temperature of air K
29 rhoair=data(:,2);        % Density of air kg/m^3
30 Cpair=data(:,3);         % Specific heat kJ/(kgK)

```

```

31 kair=data(:,4);           % Thermal conductivity W/(mK)
32 U=0;                     % Flow velocity m/sec
33
34 %%%%%%%%%%% Conditions setup %%%%%%%%%%%
35 Tamb=22;                 % Ambient temperature [degC]
36 OHR=1.8;                 % Overheat ratio
37 %%%%%%%%%%%
38
39 %%%%%%%%%%% Wire Dimensions %%%%%%%%%%%
40 dw = 25.4e-6;            % Diameter of wire [m]
41 lw = 10.90e-3;          % Length of wire [m]
42 A = pi*dw^2/4;          % Cross sectional area base [m^2]
43 As=pi*dw*lw;            % Surface area [m^2]
44 vol=A*lw;                % Volume of wire [m^3]
45 %%%%%%%%%%%
46
47 %%%%%%%%%%% Sensor Properties %%%%%%%%%%%
48 gammaP20=1.06e-07;      % Platinum Resistivity ohm-m [@20degC]
49 CP=0.0024;               % Platinum Resistivity coefficient [1/K]
50 % Source http://www.endmemo.com/physics/resistt.php
51 Cw=0.13e+03;            % Platinum & Tungsten specific heat [kJ/kg-K]
52 % Source http://www.engineeringtoolbox.com/specific-heat-metals-d\_152.html
53 rhos=21450;              % Density of wire [kg/m^3]
54 % Source http://hypertextbook.com/facts/2004/OliviaTai.shtml
55 Rs=10;                   % Shunt Resistance
56 %%%%%%%%%%%
57
58 %%%%%%%%%%% Calculations %%%%%%%%%%%
59 Rw20=gammaP20*lw/A
60 Rwopt=OHR*Rw20;          % Operating resistance of wire [ohm]
61 Topt=(Rwopt-Rw20)/(CP*Rw20)+Tamb; % Operating temperature [degC]
62 Tfilm=(Tamb+Topt)/2+273.15; % Film temperature [degC]
63 %%%%%%%%%%%
64
65 %%%%%%%%%%% Variables related to air %%%%%%%%%%%
66 % Thermal conductivity of air

```

```

67 k=polyfit(TairK,kair,3);
68 ko=polyval(k,Tfilm);           % [W/(m-K)]
69
70 % Specific heat of air at constant pressure
71 Cp=polyfit(TairK,Cpair,3);
72 Cpo=10^3*polyval(Cp,Tfilm); % [J/(kg-K)]
73
74 % Density of air
75 rhoair=polyfit(TairK,rhoair,3);
76 rhoairo=polyval(rhoair,Tfilm); % [kg/m^3]
77
78 % Dynamic viscosity calculation
79 b=1.458e-6;                   % Constant [kg/m-s-K^0.5]
80 s=110.4;                     % Constant [K]
81 mu=b*Tfilm^(1.5)/(Tfilm+s); % Viscosity [kg/m-s]
82 %Source http://www-mdp.eng.cam.ac.uk/web/library/enginfo/
83 %aerothermal_dvd_only/aero/fprops/propsoffluids/node5.html
84 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
85
86 X=0.42*ko*As/(CP*dw*Rw20)*(mu*Cpo/ko)^0.2           % X value termed as A in Eq
      2.34 Brunn
87 Y=0.57*As*ko/(CP*dw*Rw20)*(mu*Cpo/ko)^0.33*(rhoairo*dw/mu)^0.5 % Y value termed
      as B in Eq 2.34 Brunn
88 c=Cw*vol*rhos/(Rw20*CP)           % Thermal capacity of wire
89 Ew=sqrt(Rwopt*(Rwopt-Rw20)*(X+Y*sqrt(U)))
90 Eb=(Rwopt+Rs)*sqrt((Rwopt-Rw20)*(X+Y*sqrt(U))/Rwopt)
91
92 M=Rwopt+Rs;
93 N=Rwopt-Rw20;
94 F=X+Y*sqrt(U);
95
96 D1=(2*M*N*F+M^2*F-Eb^2)
97 D2= 2*Eb*Rwopt
98
99 alpha=D2/D1
100 Tw=c*M^2/D1

```


APPENDIX B

Experimental Calibration

```

1 clear all
2 close all
3 clc
4
5 Cw=130; % Table 2.1 Bruun 0.14-Tungsten; 0.13-Platinum
6 % 0.15-Platinum-rhodium, 0.13-Platinum-iridium
7 % (90-10%) (80-20%)
8 C=0.0024; % Perry Page 11
9 rho=21450; % Platinum
10 gammaP20=1.06e-07; % Platinum Resistivity ohm-m [@20degC]
11 U0=0; % Flow velocity
12 OHR0=1.8; % OHR
13 Rs=10; % Shunt resistance
14 %% TSI T1.5 %%
15 dw = 5.1e-6; % Diameter of wire [m]
16 lw = 1.27e-3; % Length of wire [m]
17 A = pi*dw^2/4; % Cross sectional area base [m^2]
18 vol=A*lw; % Volume of wire [m^3]
19 % Rw20=gammaP20*lw/A; % Cold resistance of the sensor
20 Rw20=6.1;
21 Caldata=xlsread('C:\Users\kjos004\Google_Drive\Thesis_work\Defence_related\Thesis
    _report\Karthik_Report_Final\Matlab_files\Experimental_calibration','TSI_T1.5')
    ;
22 U=Caldata(:,1);
23 OHR=Caldata(:,2);
24 Ew=Caldata(:,3);
25 Eb=Caldata(:,4);
26 Rw=Caldata(:,5);
27
28

```

```

29 for i=1:5
30     X(i)=Ew(i)^2/(Rw(i)*(Rw(i)-Rw20));
31 end
32
33 EwY=Ew(6:length(Ew));
34 RwY=Rw(6:length(Rw));
35
36 for j=1:12
37     for i=1:5
38         K(i+5*j)=(Ew(i+5*j)^2/(Rw(i+5*j)*(Rw(i+5*j)-Rw20))-X(i));
39         Y(i+5*j)=K(i+5*j)/sqrt(U(i+5*j));
40         X(i+5*j)=X(i);
41     end
42 end
43
44 Y=Y';
45 X=X';
46
47 pEb=fit([U OHR],Eb,'poly33');
48 pEw=fit([U OHR],Ew,'poly33');
49 pRw=fit([U OHR],Rw,'poly33');
50 pX=fit([U OHR],X,'poly33');
51 pY=fit([U(6:65) OHR(6:65)],Y(6:65),'poly33');
52
53 [x,y]=meshgrid(0:1:40,1.4:0.001:1.8);
54 z1=pEb.p00 + pEb.p10*x + pEb.p01*y + pEb.p20*x.^2 + pEb.p11*x.*y + pEb.p02*y.^2 +
    pEb.p30*x.^3 + pEb.p21*x.^2.*y+pEb.p12*x.*y.^2+pEb.p03*y.^3;
55 z2=pEw.p00 + pEw.p10*x + pEw.p01*y + pEw.p20*x.^2 + pEw.p11*x.*y + pEw.p02*y.^2 +
    pEw.p30*x.^3 + pEw.p21*x.^2.*y+pEw.p12*x.*y.^2+pEw.p03*y.^3;
56 z3=pRw.p00 + pRw.p10*x + pRw.p01*y + pRw.p20*x.^2 + pRw.p11*x.*y + pRw.p02*y.^2 +
    pRw.p30*x.^3 + pRw.p21*x.^2.*y+pRw.p12*x.*y.^2+pRw.p03*y.^3;
57 zX=pX.p00 + pX.p10*x + pX.p01*y + pX.p20*x.^2 + pX.p11*x.*y + pX.p02*y.^2 + pX.p30
    *x.^3 + pX.p21*x.^2.*y+pX.p12*x.*y.^2+pX.p03*y.^3;
58 zY=pY.p00 + pY.p10*x + pY.p01*y + pY.p20*x.^2 + pY.p11*x.*y + pY.p02*y.^2 + pY.p30
    *x.^3 + pY.p21*x.^2.*y+pY.p12*x.*y.^2+pY.p03*y.^3;
59

```

```

60
61
62
63 Eb0=pEb.p00 + pEb.p10*U0 + pEb.p01*OHR0 + pEb.p20*U0^2 + pEb.p11*U0*OHR0 + pEb.p02
    *OHR0^2 + pEb.p30*U0^3 + pEb.p21*U0^2*OHR0+pEb.p12*U0*OHR0^2+pEb.p03*OHR0^3;
64 Ew0=pEw.p00 + pEw.p10*U0 + pEw.p01*OHR0 + pEw.p20*U0^2 + pEw.p11*U0*OHR0 + pEw.p02
    *OHR0^2 + pEw.p30*U0^3 + pEw.p21*U0^2*OHR0+pEw.p12*U0*OHR0^2+pEw.p03*OHR0^3;
65 Rw0=pRw.p00 + pRw.p10*U0 + pRw.p01*OHR0 + pRw.p20*U0^2 + pRw.p11*U0*OHR0 + pRw.p02
    *OHR0^2 + pRw.p30*U0^3 + pRw.p21*U0^2*OHR0+pRw.p12*U0*OHR0^2+pRw.p03*OHR0^3;
66 zX0=pX.p00 + pX.p10*U0 + pX.p01*OHR0 + pX.p20*U0.^2 + pX.p11*U0.*OHR0 + pX.p02*
    OHR0.^2 + pX.p30*U0.^3 + pX.p21*U0.^2.*OHR0+pX.p12*U0.*OHR0.^2+pX.p03*OHR0.^3;
67 zY0=pY.p00 + pY.p10*U0 + pY.p01*OHR0 + pY.p20*U0.^2 + pY.p11*U0.*OHR0 + pY.p02*
    OHR0.^2 + pY.p30*U0.^3 + pY.p21*U0.^2.*OHR0+pY.p12*U0.*OHR0.^2+pY.p03*OHR0.^3;
68
69 c=rho*Cw*vol/(Rw20*C);
70 M=Rw0+Rs;
71 N=Rw0-Rw20;
72 FUbar=zX0+zY0*sqrt(U0);
73 D1=2*M*N*FUbar+M^2*FUbar-Eb0^2;
74 D2=2*Eb0*Rw0;
75
76 alpha=D2/D1;
77 Tw=c*M^2/D1;
78
79 ss=tf(alpha,[Tw 1])
80 %%                               Sensor 2                               %%
81 dws = 25.4e-6;                   % Diameter of wire [m]
82 lws = 10.90e-3;                  % Length of wire [m]
83 As = pi*dws^2/4;                 % Cross sectional area base [m^2]
84 vols=As*lws;                     % Volume of wire [m^3]
85 % Rw20s=gammaP20*lws/As;         % Cold resistance of the sensor
86 Rw20s=2.75;
87 Caldatas=xlswread('C:\Users\kjosh004\Google Drive\Thesis work\Defence related\
    Thesis report\Karthik Report Final\Matlab files\Experimental_calibration',
    'Sensor_2');
88 Us=Caldatas(:,1);

```

```

89 OHRs=Caldatas(:,2);
90 Ews=Caldatas(:,3);
91 Ebs=Caldatas(:,4);
92 Rws=Caldatas(:,5);
93
94
95 for i=1:4
96     Xs(i)=Ews(i)^2/(Rws(i)*(Rws(i)-Rw20s));
97 end
98
99 for j=1:7
100     for i=1:4
101         %           L=[U(i+5*j) Ew(i+5*j) Rw(i+5*j) X(i)];
102             Ks(i+4*j)=(Ews(i+4*j)^2/(Rws(i+4*j)*(Rws(i+4*j)-Rw20s))-Xs(i));
103             Ys(i+4*j)=Ks(i+4*j)/sqrt(Us(i+4*j));
104             Xs(i+4*j)=Xs(i);
105         end
106     end
107
108 Ys=Ys';
109 Xs=Xs';
110
111 pEbs=fit([Us OHRs],Ebs,'poly33');
112 pEws=fit([Us OHRs],Ews,'poly33');
113 pRws=fit([Us OHRs],Rws,'poly33');
114 pXs=fit([Us OHRs],Xs,'poly33');
115 pYs=fit([Us(5:32) OHRs(5:32)],Ys(5:32),'poly33');
116
117 [xs,ys]=meshgrid(0:1:40,1.6:0.001:1.9);
118 z1s=pEbs.p00 + pEbs.p10*xs + pEbs.p01*ys + pEbs.p20*xs.^2 + pEbs.p11*xs.*ys + pEbs
        .p02*ys.^2 + pEbs.p30*xs.^3 + pEbs.p21*xs.^2.*ys+pEbs.p12*xs.*ys.^2+pEbs.p03*ys
        .^3;
119 z2s=pEws.p00 + pEws.p10*xs + pEws.p01*ys + pEws.p20*xs.^2 + pEws.p11*xs.*ys + pEws
        .p02*ys.^2 + pEws.p30*xs.^3 + pEws.p21*xs.^2.*ys+pEws.p12*xs.*ys.^2+pEws.p03*ys
        .^3;
120 z3s=pRws.p00 + pRws.p10*xs + pRws.p01*ys + pRws.p20*xs.^2 + pRws.p11*xs.*ys + pRws

```

```

.p02*ys.^2 + pRws.p30*xs.^3 + pRws.p21*xs.^2.*ys+pRws.p12*xs.*ys.^2+pRws.p03*ys
.^3;
121 zXs=pXs.p00 + pXs.p10*xs + pXs.p01*ys + pXs.p20*xs.^2 + pXs.p11*xs.*ys + pXs.p02*
ys.^2 + pXs.p30*xs.^3 + pXs.p21*xs.^2.*ys+pXs.p12*xs.*ys.^2+pXs.p03*ys.^3;
122 zYs=pYs.p00 + pYs.p10*xs + pYs.p01*ys + pYs.p20*xs.^2 + pYs.p11*xs.*ys + pYs.p02*
ys.^2 + pYs.p30*xs.^3 + pYs.p21*xs.^2.*ys+pYs.p12*xs.*ys.^2+pYs.p03*ys.^3;
123
124
125 Eb0s=pEbs.p00 + pEbs.p10*U0 + pEbs.p01*OHR0 + pEbs.p20*U0^2 + pEbs.p11*U0*OHR0 +
pEbs.p02*OHR0^2 + pEbs.p30*U0^3 + pEbs.p21*U0^2*OHR0+pEbs.p12*U0*OHR0^2+pEbs.
p03*OHR0^3;
126 Ew0s=pEws.p00 + pEws.p10*U0 + pEws.p01*OHR0 + pEws.p20*U0^2 + pEws.p11*U0*OHR0 +
pEws.p02*OHR0^2 + pEws.p30*U0^3 + pEws.p21*U0^2*OHR0+pEws.p12*U0*OHR0^2+pEws.
p03*OHR0^3;
127 Rw0s=pRws.p00 + pRws.p10*U0 + pRws.p01*OHR0 + pRws.p20*U0^2 + pRws.p11*U0*OHR0 +
pRws.p02*OHR0^2 + pRws.p30*U0^3 + pRws.p21*U0^2*OHR0+pRws.p12*U0*OHR0^2+pRws.
p03*OHR0^3;
128 zX0s=pXs.p00 + pXs.p10*U0 + pXs.p01*OHR0 + pXs.p20*U0.^2 + pXs.p11*U0.*OHR0 + pXs.
p02*OHR0.^2 + pXs.p30*U0.^3 + pXs.p21*U0.^2.*OHR0+pXs.p12*U0.*OHR0.^2+pXs.p03*
OHR0.^3;
129 zY0s=pYs.p00 + pYs.p10*U0 + pYs.p01*OHR0 + pYs.p20*U0.^2 + pYs.p11*U0.*OHR0 + pYs.
p02*OHR0.^2 + pYs.p30*U0.^3 + pYs.p21*U0.^2.*OHR0+pYs.p12*U0.*OHR0.^2+pYs.p03*
OHR0.^3;
130
131 cs=rho*Cw*vols/(Rw20s*C);
132 Ms=Rw0s+Rs;
133 Ns=Rw0s-Rw20s;
134 FUbars=zX0s+zY0s*sqrt(U0);
135 D1s=2*Ms*Ns*FUbars+Ms^2*FUbars-Eb0s^2;
136 D2s=2*Eb0s*Rw0s;
137
138 alphas=D2s/D1s;
139 Tws=cs*Ms^2/D1s;
140
141 sss=tf(alphas,[Tws 1])
142 %% Figures %%

```

```

143
144 %-----%
145 figure('units','inches','pos',[0 0 6 7.5])
146 ax1 = subplot(2,1,1);
147 mesh(x,y,z1)
148 xlabel('U(m/sec)')
149 ylabel('OHR')
150 zlabel('Eb(V)')
151 title('Bridge_voltage_map-TSI_T1.5')
152
153 ax2 = subplot(2,1,2);
154 mesh(xs,ys,z1s)
155 xlabel('U(m/sec)')
156 ylabel('OHR')
157 zlabel('Eb(V)')
158 title('Bridge_voltage_map-Sensor_2')
159 % scatter3(U0,OHR0,Eb0)
160 % hold on
161 %-----%
162 figure('units','inches','pos',[0 0 6 7.5])
163 ax1 = subplot(2,1,1);
164 mesh(x,y,z2)
165 xlabel('U(m/sec)')
166 ylabel('OHR')
167 zlabel('Ew(V)')
168 title('Sensor_voltage_map-TSI_T1.5')
169
170 ax1 = subplot(2,1,2);
171 mesh(xs,ys,z2s)
172 xlabel('U(m/sec)')
173 ylabel('OHR')
174 zlabel('Ew(V)')
175 title('Sensor_voltage_map-Sensor_2')
176 %-----%
177 figure('units','inches','pos',[0 0 6 7.5])
178 ax1 = subplot(2,1,1);

```

```

179 mesh(x,y,z3)
180 xlabel('U(m/sec)')
181 ylabel('OHR')
182 zlabel('Rw(ohm)')
183 title('Operating_resistance_map-TSI_T1.5')
184
185 ax2 = subplot(2,1,2);
186 mesh(xs,ys,z3s)
187 xlabel('U(m/sec)')
188 ylabel('OHR')
189 zlabel('Rw(ohm)')
190 title('Operating_resistance_map-Sensor_2')
191 %-----%
192 figure('units','inches','pos',[0 0 6 7.5])
193 ax1 = subplot(2,1,1);
194 mesh(x,y,zX)
195 xlabel('U(m/sec)')
196 ylabel('OHR')
197 zlabel('Xcoefficient')
198 title('Xcoefficient_map-TSI_T1.5')
199
200 ax2 = subplot(2,1,2);
201 mesh(xs,ys,zXs)
202 xlabel('U(m/sec)')
203 ylabel('OHR')
204 zlabel('Xcoefficient')
205 title('Xcoefficient_map-Sensor_2')
206 %-----%
207 [x,y]=meshgrid(3:1:36,1.4:0.001:1.8);
208 [xs,ys]=meshgrid(5:1:35,1.6:0.001:1.9);
209 figure('units','inches','pos',[0 0 6 7.5])
210 ax1 = subplot(2,1,1);
211 mesh(x,y,zY(:,8:41))
212 xlabel('U(m/sec)')
213 ylabel('OHR')
214 zlabel('Ycoefficient')

```

```
215 title('Y_coefficient_map-TSI_T1.5')
216
217 ax2 = subplot(2,1,2);
218 mesh(xs,ys,zYs(:,11:41))
219 xlabel('U(m/sec)')
220 ylabel('OHR')
221 zlabel('Y_coefficient')
222 title('Y_coefficient_map-Sensor_2')
```


APPENDIX C

Validation of Results

```

1 % Karthik's cylinder wake
2 clc,clear all ,close all
3 [datatsie]=xlsread('C:\Users\Karthik\Google_Drive\Thesis_work\Defence_related\
    Platinum_related_Experiments\Validation_experiments\WAKE_CONSOLIDATED_Exp2',
    'TSIerror');
4 [datadbtae]=xlsread('C:\Users\Karthik\Google_Drive\Thesis_work\Defence_related\
    Platinum_related_Experiments\Validation_experiments\WAKE_CONSOLIDATED_Exp2',
    'DBTAerror');
5 [datatsi]=xlsread('C:\Users\Karthik\Google_Drive\Thesis_work\Defence_related\
    Platinum_related_Experiments\Validation_experiments\WAKE_CONSOLIDATED_Exp2',
    'TSI');
6 [datadbta]=xlsread('C:\Users\Karthik\Google_Drive\Thesis_work\Defence_related\
    Platinum_related_Experiments\Validation_experiments\WAKE_CONSOLIDATED_Exp2',
    'DBTA');
7 t=datatsie(1:7001,1);
8 xtsi=datatsie(1:7001,2:33);
9 xdbta=datadbtae(1:7001,2:33);
10 b=2*pi*110;
11 a=[1 b];
12 % LPtf=tf(wc,[1 wc]);
13 for n=1:32
14 mtsi(n)=mean(datatsi(:,n));
15 mdbta(n)=mean(datadbta(:,n));
16 rtsi(n)=rms(datatsi(:,n));
17 rdbta(n)=rms(datadbta(:,n));
18 TItsi(n)=rtsi(n)/mtsi(n);
19 TIdbta(n)=rdbta(n)/mdbta(n);
20 [ptsi(:,n),f(:,n)]=pwelch(xtsi(:,n),1000,500,1000,1000);
21 pdbta(:,n)=pwelch(xdbta(:,n),1000,500,1000,1000);
22 dbtsi(:,n)=20*log10(ptsi(:,n));

```

```

23 dbdbta(:,n)=20*log10(pdbta(:,n));
24 figure
25 plot(f(10:100,n),dbtsi(10:100,n),f(10:100,n),dbdbta(10:100,n));
26 grid;xlabel('Frequency(Hz)'),ylabel('Power(V^2/Hz)');
27 axis([10 100 -80 0]);
28 legend('TSI','DBTA');
29 end
30 %%
31 pos=-4:0.25:3.75;
32 polyMtsi=polyfit(pos,mtsi,3);
33 polyMdbta=polyfit(pos,mdbta,3);
34 pvalMtsi=polyval(polyMtsi,pos);
35 pvalMdbta=polyval(polyMdbta,pos);
36 figure
37 plot(pos,mdbta,'bo',pos,mtsi,'ro')
38 hold on
39 plot(pos,pvalMdbta,'b',pos,pvalMtsi,'r')
40 title('Mean_velocity_distribution')
41 xlabel('Position(in)')
42 ylabel('velocity(m/sec)')
43 legend('DBTA','TSI')
44 hold off
45
46 polyTtsi=polyfit(pos,Titsi,3);
47 polyTdbta=polyfit(pos,Tidbta,3);
48 pvalTtsi=polyval(polyTtsi,pos);
49 pvalTdbta=polyval(polyTdbta,pos);
50 figure
51 plot(pos,Tidbta,'bo',pos,Titsi,'ro')
52 hold on
53 plot(pos,pvalTdbta,'b',pos,pvalTtsi,'r')
54 title('Turbulence_intensity_distribution')
55 xlabel('Position(x0.25inch)')
56 ylabel('velocity(m/sec)')
57 legend('DBTA','TSI')
58 hold off

```

```

59
60
61 m=log(f(10:100,1))';
62 [xx,yy]=meshgrid(f(10:100),pos);
63 %-----
64 figure
65 surfc(xx,yy,dbtsi(10:100,:))'
66 % shading interp
67 h = colorbar;
68 set(get(h,'title'),'string','Power(V^2/Hz)');
69 title('TSI Power Spectrum in 0.5inch interval')
70 ylabel('Position')
71 xlabel('Frequency(Hz)')
72 % set(gca,'XScale','log')
73 zlabel('Power(V^2/Hz)')
74 view(0,90)
75 %-----
76 % figure
77 % contourf(xx,yy,dbtsi(1:100,:))',20)
78 % % shading interp
79 % colormap(hot)
80 % colorbar
81 % title('TSI Power Spectrum in 0.5inch interval')
82 % ylabel('Position')
83 % set(gca,'XScale','log')
84 % xlabel('Frequency (Hz)')
85 % %-----
86 figure
87 surfc(xx,yy,dbdbta(10:100,:))'
88 % shading interp
89 h = colorbar;
90 set(get(h,'title'),'string','Power(V^2/Hz)');
91 title('DBTA Power Spectrum in 0.5inch interval')
92 ylabel('Position')
93 xlabel('Frequency(Hz)')
94 % set(gca,'XScale','log')

```

```
95 xlabel('Power (V2/Hz)')
96 view(0,90)
97 %-----
98 % figure
99 % contourf(xx,yy,dbdbta(1:100,:)',20)
100 % % shading interp
101 % colormap(hot)
102 % colorbar
103 % title('TSI Power Spectrum in 0.5inch interval')
104 % ylabel('Position')
105 % set(gca,'XScale','log')
106 % xlabel('Frequency (Hz)')
107 %%
108 St=0.2;
109 D=0.0889;
110 V=10;
111 f=St*V/D
```

REFERENCES

- [1] O. Osorio and N. Silin, "Wall shear stress hot film sensor for use in gases," in *Journal of Physics: Conference Series*, vol. 296, no. 1. IOP Publishing, 2011, p. 012002.
- [2] J. Wyngaard and J. Lumley, "A constant temperature hot-wire anemometer," *Journal of Scientific Instruments*, vol. 44, no. 5, p. 363, 1967.
- [3] I. Kidron, "Measurement of the transfer function of hot-wire and hot-film turbulence transducers," *IEEE Transactions on Instrumentation and Measurement*, vol. 15, no. 3, pp. 76–81, 1966.
- [4] L. V. King, "On the precision measurement of air velocity by means of the linear hot-wire anemometer," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 29, no. 172, pp. 556–577, 1915.
- [5] A. Perry and G. Morrison, "A study of the constant-temperature hot-wire anemometer," *J. Fluid Mech*, vol. 47, no. 3, pp. 577–599, 1971.
- [6] A. Perry, "Hot-wire anemometry," Report, 1982.
- [7] H. H. Bruun, "Hot-wire anemometry," 1995.
- [8] T. R. Moes, G. R. Sarma, and S. M. Mangalam, "Flight demonstration of a shock location sensor using constant voltage hot-film anemometry," 1997.
- [9] J. F. Foss, D. G. Bohl, and T. J. Hicks, "The pulse width modulated-constant temperature anemometer," *Measurement Science and Technology*, vol. 7, no. 10, p. 1388, 1996.
- [10] L. V. King, "On the convection of heat from small cylinders in a stream of fluid: determination of the convection constants of small platinum wires with applications to hot-wire anemometry," *Philosophical Transactions of the Royal Society of London. Series*

A, Containing Papers of a Mathematical or Physical Character, vol. 214, pp. 373–432, 1914.

- [11] H. Kramers, “Heat transfer from spheres to flowing media,” *Physica*, vol. 12, no. 2-3, pp. 61–80, 1946.
- [12] R. Goldstein, *Fluid mechanics measurements*. CRC Press, 1996.
- [13] B. Friedland, *Control system design: An introduction to state-space methods*. Courier Corporation, 2012.
- [14] A. L. Favaregh, C. P. Britcher, and D. Landman, “A new approach for calibration of hot-wires for use in uncertain environments,” in *25th AIAA Aerodynamic Measurement Technology and Ground Testing Conference*, 2006, p. 2809.

VITA

Karthik Kamalakar Joshi
Department of Mechanical and Aerospace Engineering
115 Kaufman Hall
Norfolk, VA 23529-0247

Education and Experience:

Bachelor of Science (Mechanical Engineering), Kayatiya Institute of Technology and Science, May 2011

Master of Science (Mechanical Engineering), Old Dominion University, December 2017
Option Area: Design and Manufacturing

Worked for Hyundai Motor India Engineering for three years as Design Engineer. My role was to design interior plastic parts such as pillar trims, dash board, headlining. I did a summer internship at Volvo Construction Equipment, Shippensburg PA as Design Intern. In VCE I was involved in redesign of generator coupler in paver.