

Open Water Testing of a Surface Piercing Propeller with Varying Submergence, Yaw
Angle and Inclination Angle

by

Justin M. Lorio

A Thesis Submitted to the Faculty of
The College of Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degree of
Master of Science

Florida Atlantic University

Boca Raton, Florida

May 2010

UMI Number: 1484581

All rights reserved

INFORMATION TO ALL USERS

The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if material had to be removed, a note will indicate the deletion.



UMI 1484581

Copyright 2010 by ProQuest LLC.

All rights reserved. This edition of the work is protected against unauthorized copying under Title 17, United States Code.



ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

Open Water Testing of a Surface Piercing Propeller with Varying Submergence, Yaw
Angle and Inclination Angle

by

Justin Lorio

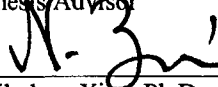
This thesis was prepared under the direction of the candidate's thesis advisor, Dr. Karl von Ellenrieder, Department of Ocean and Mechanical Engineering, and has been approved by the members of his supervisory committee. It was submitted to the faculty of the College of Engineering and Computer Science and was accepted in partial fulfillment of the requirements for the degree of Master of Science.

SUPERVISORY COMMITTEE:



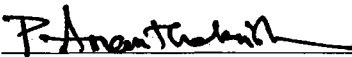
Karl von Ellenrieder, Ph.D.

Thesis Advisor



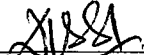
Nikolaos Xiros Ph.D.

Assistant Professor



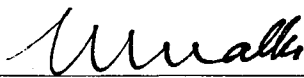
Palaniswamy Ananthakrishnan, Ph.D.

Associate Professor



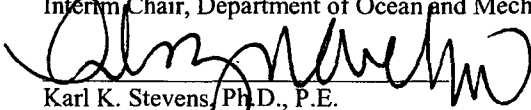
Raju Datla, Ph.D.

Research Associate Professor,
Stevens Institute of Technology



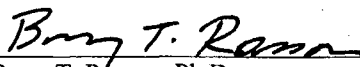
Mohammad Ilyas, Ph.D.

Interim Chair, Department of Ocean and Mechanical Engineering



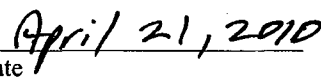
Karl K. Stevens, Ph.D., P.E.

Dean, College of Engineering and Computer Science



Barry T. Rosson, Ph.D.

Dean, Graduate College


Date

ABSTRACT

Author: Justin M. Lorio
Title: Open Water Testing of a Surface Piercing Propeller with Varying Submergence, Yaw Angle and Inclination Angle
Institution: Florida Atlantic University
Thesis Advisor: Dr. Karl vonEllenrieder
Degree: Master of Science in Ocean Engineering
Year: 2010

The use of surface piercing propellers (SPPs) shows promise for high speed operation by virtually eliminating appendage drag, which can be as much as 30 percent of the total drag on a vehicle at high speeds. The scarcity of available systematic test data has made reliable performance prediction difficult. The primary objective of this research is to obtain experimental performance prediction data that can be used in SPP design. In a series of open water tests in a non-pressurized towing tank facility, force transducer measurements were taken at tip immersion ratios from 0.5 to .33, yaw angles from 0° to 30° and inclination angles from 0° to 15° over a range of advance ratios from 0.8 to 1.8. Force transducer measurements were taken for thrust, torque, side forces and moments. These results will help develop a baseline for the verification of SPP performance prediction.

DEDICATION

To my family, without their support, this would not have been possible

CONTENTS

1.0 BACKGROUND.....1

 1.1 Introduction.....1

 1.2 Propeller Theory2

 1.3 Fundamentals of Surface Piercing Propellers6

2.0 EXPERIMENTAL APPROACH AND APPARATUS12

 2.1 Propeller.....12

 2.2 Facility13

 2.3 Apparatus Design.....16

 2.3.1 Design Load Estimation19

 2.3.2 Resonant Frequency Analysis20

 2.3.3 Mechanical System Design21

 2.3.4 Control System Design29

 2.3.5 Data Collection System.....35

 2.4 Experiments39

3.0 DATA ANALYSIS41

4.0 DISCUSSION45

 4.1 Uncertainty Analysis.....60

5.0 CONCLUSION61

 5.1 Future Work.....62

REFERENCES.....	63
APPENDIX A: PROPELLER MEASUREMENT.....	64
APPENDIX B: RESONANT FREQUENCY CALCULATIONS.....	65
Appendix B.1: Strut Natural Frequency Calculation	65
Appendix B.2: Propeller Shaft Natural Frequency Calculation	68
APPENDIX C: TEST CONDITIONS.....	70
APPENDIX D: RAW DATA AND POWER SPECTRAL DENSITY PLOTS	81
APPENDIX E: FILTERED DATA FOR ROTATIONAL RATE, THRUST AND TORQUE	84
APPENDIX F: MATLAB DATA ANALYSIS CODE.....	117
Appendix F.1: Function Condns	131
Appendix F.2: Function Condns_Cross	133
Appendix F.3: Function CoordTrans	135
Appendix F.4: Function FerrandPlot.....	136
Appendix F.5: Function Filter.....	146
Appendix F.6: Function PSD	149
Appendix F.7: Function txtreader	151
Appendix F.8: Function V2F	152
Appendix F.9: Function V2F_Cross	153
APPENDIX G: MATRICES FOR VOLTAGE TO FORCE CONVERSION	154

TABLE OF TABLES

Table 1. Propeller characteristics	13
Table 2. U_A required to meet critical F_n	14
Table 3. Motion control system holding torques and gear ratios.....	25
Table 4. Channels of Data	41
Table 5. Filter Properties	42
Table 6. Independent uncertainty values	60

TABLE OF FIGURES

Figure 1. Propeller with cavitation damage (Olds Engineering)	5
Figure 2. Propeller blade sections (van Manen and van Oossanen 1988)	7
Figure 3. Predicted trends for K_T , (13), K_Q (14) and η (15), for a general case with $\Psi = 0^\circ$, $\gamma = 0^\circ$ and $\left(\frac{P}{D}\right) = 1.9$	10
Figure 4. Test Propeller.....	12
Figure 5. Stevens Institute of Technology's Davidson Laboratory	15
Figure 6. Precision rail and overhead beam in Davidson Laboratory.....	16
Figure 7. Assembling for functional testing at Seatech.....	17
Figure 8. Traditional propeller boat used at Webb Institute of Naval Architecture	18
Figure 9. Ratio of dynamic load to time average load versus advance ratio	19
Figure 10. Propeller and shaft assembly.....	20
Figure 11. Inclination mechanical system	21
Figure 12. Yaw mechanical system.....	22
Figure 13. Immersion mechanical system	22
Figure 14. Immersion system and support structure.....	23
Figure 15. Yaw drive, exploded view	24
Figure 16. Inclination drive, exploded view.....	25
Figure 17. Right angle drive and mounting assembly	26
Figure 18. Lower assembly, still in assembly process.....	26
Figure 19. Upper part of propeller drive.....	27
Figure 20. Lower part of propeller drive	27
Figure 21. Full assembly rendering.....	28
Figure 22. Prototype assembled, excluding propeller, at test site.....	29

Figure 23. Carriage mounted electronics.....	30
Figure 24. Yaw control system diagram.....	30
Figure 25. Inclination control system diagram.....	31
Figure 26. Bottom assembly before fairings	32
Figure 27. Propeller motor control system diagram	33
Figure 28. Generic velocity profile for test run	34
Figure 29. Linear encoder on carriage support beam	35
Figure 30. Data collection system diagram	35
Figure 31. Coordinate axes with respect to the force transducer.....	36
Figure 32. Force transducer.....	36
Figure 33. Slip ring side of right angle drive with connector and wire exposed	37
Figure 34. Inside of slip ring coupler	38
Figure 35. Slip ring and watertight connector assembly	38
Figure 36. Compass/inclinometer watertight box.....	39
Figure 37. Unfiltered data for blade position versus time	42
Figure 38. Representative sample of biased data.....	43
Figure 39. Test run observations	45
Figure 40. K_T , K_Q and η for predicted values versus experimental values.....	46
Figure 41. K_T , K_Q and η for predicted values versus experimental values.....	47
Figure 42. K_T , K_Q and η for predicted values versus experimental values.....	48
Figure 43. K_T , K_Q and η for predicted values versus experimental values.....	49
Figure 44. K_T , K_Q and η for predicted values versus experimental values.....	50
Figure 45. K_T , K_Q and η for predicted values versus experimental values.....	51
Figure 46. K_T , K_Q and η for predicted values versus experimental values.....	52
Figure 47. K_T , K_Q and η for predicted values versus experimental values.....	53
Figure 48. K_T , K_Q and η for predicted values versus experimental values.....	54
Figure 49. K_T , K_Q and η for predicted values versus experimental values.....	55
Figure 50. K_T and K_Q holding angles of inclination and yaw constant while varying depth of immersion ...	56

Figure 51. K_T holding angle of yaw and depth of immersion constant while varying inclination angle57

Figure 52. K_Q holding angle of yaw and depth of immersion constant while varying inclination angle58

Figure 53. K_T holding angle of inclination and depth of immersion constant while varying yaw angle59

Figure 54. K_T holding angle of inclination and depth of immersion constant while varying yaw angle60

NOMENCLATURE

A	Planform Area	a_c	Carriage Acceleration (Eq 19)
C_L	Lift Coefficient	a_p	Propeller Acceleration (Eq 21)
D	Propeller Diameter	d_c	Acceleration Distance
F_n	Froude Number Based on Propeller Diameter (Eq 7)	g	Gravity
I	Centerline Immersion Ratio (Eq 4)	h	Distance from Centerline of Propeller to Free Surface
I_T	Tip Immersion Ratio (Eq 5)	h_c	Distance from Bottom Tip of Propeller to Free Surface
J	Advance Ratio (Eq 3)	n	Rotational Rate of Propeller
J_{SCALED}	Advance Ratio Scaled (Eq 23)	p_a	Atmospheric Pressure
K_T	Non-Dimensional Thrust Coefficient (Eq 1)	p_v	Vapor Pressure
K_Q	Non-Dimensional Torque Coefficient (Eq 2)	p_0	Total Pressure (Eq 11)
L	Lift Force (Eq 9)	r	Distance from Centerline of Propeller to Some Radius
$\frac{P}{D}$	Pitch to Diameter Ratio	t	Time
Q	Torque	t_a	Time period of Acceleration (Eq 20)
T	Thrust Force	t_d	Time period of Deceleration
R_n	Reynolds Number (Eq 6)	t_{ss}	Time period of Acceleration
U	Resultant Velocity (Eq 8)	γ	Angle of Shaft Inclination
U_A	Speed of Forward Advance	η	Propeller Efficiency (Eq 17 and 24)
$U_{ASCALED}$	Speed of Forward Advance Scaled for Shaft Angle	ν	Dynamic Viscosity
V_{OUT}	Voltage Out to Propeller Motor (Eq 18)	ρ	Density
W_n	Weber Number Based on Propeller Diameter (Eq 13)		

σ

Cavitation Number (Eq 10 and
12)

σ_{κ}

Kinematic Capillarity

Ψ

Angle of Shaft Yaw

1.0 BACKGROUND

The U.S. Navy is currently working to develop the Transformable Craft (T-Craft). One of the missions of this vessel is to transport wheeled and tracked vehicles at speeds of 20 m/s from a loading point offshore, through the shallow surf zone and onto the beach.

The requirement of a high speed shallow water mode makes surface piercing propellers (SPPs) a promising candidate for the method of propulsion for the T-Craft. SPPs have been used by the marine racing community for decades, and have been used on vehicles which travel in excess of 50 m/s. Also, the Navy has used them for the propulsion of surface effects ships and other small high speed vessels, some of which have attained speeds in excess of 40 m/s, such as the SES 100B (Clark 2004). The design of a partially submerged propeller drive system allows for most of the drive assembly (shafts, struts, stern tubes, etc.), as well as half of the propeller, to be elevated above the water surface when the vehicle is running at speed. This reduction in the operational profile gives a shallow draft, as well as a considerable drop in the total drag on the vessel, considering that appendage drag can be as much as 30% of the total vehicle drag at high speeds (Ferrando 2001).

1.1 Introduction

The nature of SPP operation presents many problems for designers. The interaction of the blade of the propeller with the air/water interface is a very difficult physical problem to model mathematically, which is why physical experiments are still a very important part of design process. This makes the preliminary costs for incorporating SPPs into a marine vehicle design very high.

In order to help advance the state of the art in the performance prediction of surface piercing propellers, we a towing tank facility was used to evaluate the open water performance of a SPP. Force

transducer data was recorded to measure the steady and unsteady thrust torque and side forces produced by a partially submerged propeller. Our performance data and analysis, along with any important flow observations, concurrently conducted by Altamirano (2010), can be used to develop numerical models, providing a baseline for comparison of computational and experimental results.

1.2 Propeller Theory

In order to explain the fundamentals of SPPs we must first refer to conventional propeller theory. The common non-dimensional numbers used in characterizing the properties of a conventional propeller can be found in equations 1 through 7. These properties allow us to scale our model results.

$$K_T = \frac{T}{\rho n^2 D^4}. \quad (1)$$

$$K_Q = \frac{Q}{\rho n^2 D^5}. \quad (2)$$

$$J = \frac{U_A}{nD}. \quad (3)$$

$$I = \frac{h}{D}. \quad (4)$$

$$I_T = \frac{h_T}{D}. \quad (5)$$

$$R_n = \frac{nD^2}{\nu}. \quad (6)$$

$$F_n = \frac{nD}{\sqrt{gD}}. \quad (7)$$

Where:

D = Diameter of Propeller

F_n = Froude Number

I = Propeller Centerline Immersion Ratio

I_T = Propeller Tip Immersion Ratio

J = Advance Ratio

K_T = Non-Dimensional Thrust Coefficient

K_Q = Non-Dimensional Torque Coefficient

Q = Torque

R_n = Reynolds Number

T = Thrust Force

U_A = Advance Velocity

g = Gravitational Acceleration

h = Distance from Free Surface to Propeller Centerline

h_T = Distance from Bottom Tip of Propeller to Free Surface

n = Rotational Rate

ν = Kinematic Viscosity

ρ = Density

The Momentum Theory of Propeller Action developed in Rankine (1865) and Froude (1887) develops the ideal conception of the propeller as a “disk” or mechanism capable of imparting a sudden increase in pressure to the fluid passing through it. The assumptions upon which the theory is based are the following:

- 1) The propeller works in an ideal fluid and therefore, does not experience energy losses due to frictional drag.
- 2) The propeller can be replaced by an actuator disk, and this is equivalent to saying that the propeller has an infinite number of blades.
- 3) The propeller can produce thrust without causing rotation in the slipstream.

This theory uses the disk to absorb all of the power of the engine and dissipate the power into the fluid by causing a pressure jump, and hence an increase in total head of the fluid, across the two faces of

the disc (Carlton 2007). In the case of the marine propeller, the suction side (upstream side) generates a low pressure and the pressure side (downstream side) produces a high pressure. The greater the pressure jump across the disc, the greater the thrust produced by the propeller.

The mechanism by which the difference in pressure is produced can be easily seen through 2-D foil theory, since a propeller section taken at any radius is simply a 2-D foil section. When the propeller is rotating at some rate n , we can calculate an instantaneous velocity at any radius r by $2\pi nr$. When resolved with some inflow into the propeller U_A due to the forward motion of the ship resultant inflow velocity U is obtained. As a result, if the rotational rate of the propeller, or the forward speed of the ship increases, the inflow velocity into the propeller increases.

$$U = \sqrt{U_A^2 + (2\pi nr)^2} . \quad (8)$$

Now, from thin foil theory, the lift force L delivered is:

$$L = \frac{1}{2}\rho U^2 C_L A . \quad (9)$$

Where:

A = Planform Area

C_L = Lift Coefficient

From Bernoulli's principle we know for lift to increase, the pressure difference across the foil has to increase, and from equation 9 we see that lift increases as U^2 . Therefore we can state that for a propeller, as rotational rate increases, U increases, causing lift to increase as long as the forward velocity of the ship increases or stays the same. However, this mechanism does reach a limit. We eventually reach the point where the pressure on the suction side of the blade drops too far and cavitation develops.

Cavitation can be described as when the pressure somewhere on the suction side of the blade becomes equal to the vapor pressure, p_v , and an air cavity forms. When we have partial cavitation, a portion of the suction side of the blade is unwetted. This can cause damage to the foil structure when the cavitation bubbles or sheets implode on the blade. This causes shockwaves to slam into the blade, which

causes pitting, leading to erosion of the propeller. Figure 1 shows how detrimental the effects of cavitation can be on a propeller blade.

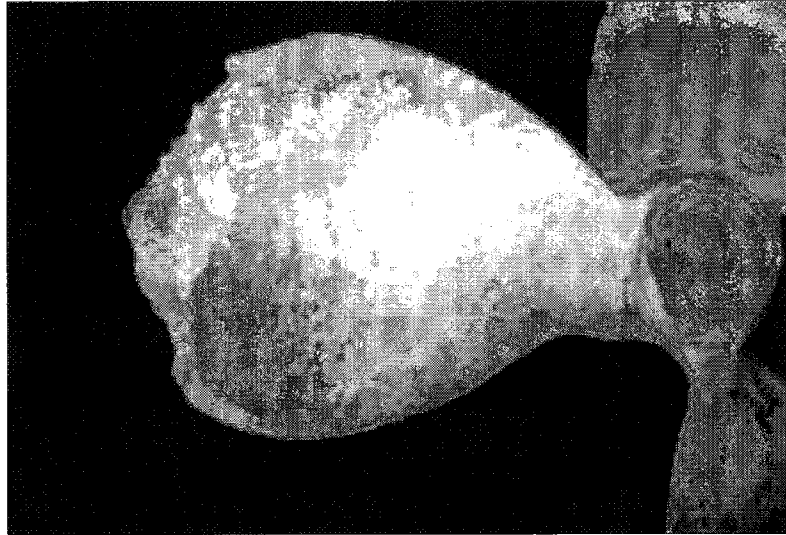


Figure 1. Propeller with cavitation damage (Olds Engineering)

To avoid cavitation the cavitation number must be in the safe range, as described below. The cavitation number, σ , is defined as:

$$\sigma = \frac{p_0 - p_v}{\frac{1}{2}\rho U^2} . \quad (10)$$

The total static pressure, p_0 , on a propeller is defined as the sum of atmospheric pressure, p_a and hydrostatic pressure, $\rho g(h \pm r)$, at a given instantaneous position on the propeller.

$$p_0 = p_a + \rho g(h \pm r) . \quad (11)$$

So we can re-write cavitation number as:

$$\sigma = \frac{p_a + \rho g(h \pm r) - p_v}{\frac{1}{2}\rho U^2} . \quad (12)$$

With the cavitation number, the lower the value, the higher the possibility of cavitation. When the total pressure becomes equal to vapor pressure ($\sigma=0$), which is about 0.012 atmospheres in fresh water or as high as 0.17 atmospheres in salt water cavitation occurs (van Manen and van Oossanen 1988). Even

though we know cavitation occurs at $\sigma=0$, it can occur at cavitation numbers greater than 0. In order to find a propeller that will operate without cavitation we can use propeller performance curves. For more information on this refer to (van Manen and van Oossanen 1988).

Cavitation is very difficult to avoid with a conventional, fully submerged propeller at high speeds. Hydrodynamicists describe a high speed, submerged hull, vessel as one which operates at $F_n \geq 0.4$. When discussing marine vehicles, a vehicle that operates at speed above approximately 15 m/s is often used to define high speed (Faltinsen 2005). At speeds of about 25 m/s conventional, fully submerged propellers have difficulty operating without cavitation.

1.3 Fundamentals of Surface Piercing Propellers

When a propeller surpasses the partially cavitating phase it becomes fully- or super-cavitating. In supercavitation the entire suction side of the propeller blade is unwetted, leaving the cavity to collapse in the trailing wake of the propeller blade. This allows the backside of the blade to be protected from the implosion of cavitation bubbles. When the propeller operates in this regime the cavity is reasonably steady state, reducing vibration from partial cavitation operation (van Manen and van Oossanen 1988).

Once the propeller has reached the supercavitating regime, the pressure on the suction side of the blade cannot be reduced anymore; therefore no additional lift can be generated by the backside. Only the face of the blade continues to create lift by continuing to build pressure as rotational speed of the propeller increases. Therefore, the propeller will still continue to increase thrust, but it will be at a slower rate than before cavitation began (van Manen and van Oossanen 1988).

When aiming to design a propeller that operates efficiently in fully cavitating regime, the section shape must provide clean flow separation at the leading and trailing edges. To do this, blades with wedge shaped cross sections are commonly used, on SPPs and supercavitating propellers. The differences between the shapes of subcavitating and supercavitating sections can be seen in figure 2.

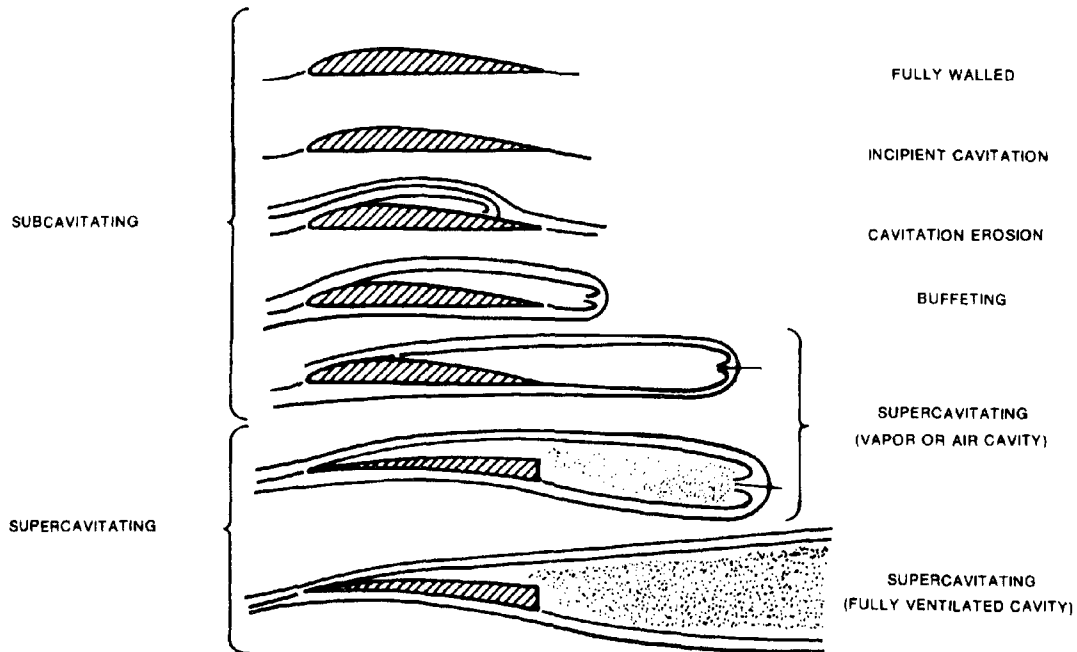


Figure 2. Propeller blade sections (van Manen and van Oossanen 1988)

Supercavitating propellers have been found to achieve their highest efficiencies at partial submergence (van Manen and van Oossanen 1988), which leads to the development of SPPs. SPPs are basically a type of supercavitating propeller that operates at partially submerged conditions allowing the cavity formed to ventilate to the free surface. SPPs have been found to be more efficient than the fully submerged supercavitating propeller for the following reasons (Ghiasi 2009):

- 1) Elevated shafts, struts, stern tube, etc. give a reduction in appendage drag.
- 2) The propeller diameter is not limited by the blade time clearance from the hull or the maximum vessel draft, which allows for a larger propeller diameter.
- 3) The blade surface friction and erosion are reduced since cavitation is replaced by ventilation.
- 4) Vertical force is generated by the propeller to obtain the optimum running trim.
- 5) The torque required decreases, giving a higher efficiency

With all of their benefits, SPPs are not without their problems. One major problem is designing a propeller that has good supercavitating qualities hydrodynamically as well as structurally. The extremely fine leading edge and thick trailing edge needed to ensure clean separation can cause structural instabilities.

When the blade breaks through the free surface of the water it generates a very high impact pressure. As a result, the thin leading edge, where most of the impact pressure is absorbed, is particularly susceptible to strength and fatigue issues. Bending and/or torsional oscillations may also occur due to these high impact pressures that generate a load eccentric to the centroid of the blade. Also resonant vibration may occur if the frequency of the cyclic loading comes close to the natural frequency of the blade (Young and Kinnas 2004).

The performance prediction of surface piercing propellers has been a problem troubling researchers for decades. Even though significant advancements have been made in this area, there is still much room for improvement. One of the drawbacks to the numerical prediction methods being developed today is that systematic test data of SPPs are affected by Froude number, Weber number, and immersion ratio, in addition to the usual nondimensional parameters (e.g. cavitation number, advance coefficient, Reynolds number), making validation of the numerical results difficult (Young and Kinnas 2004). This causes the reliability of the force estimations for SPPs to be uncertain, whereas the numerical methods developed for conventional propellers and waterjets have a well defined baseline to compare their results.

Full scale trial and error tests as well as model experiments are still a commonly used design approach for SPPs. Model tests performed in towing tanks or free surface cavitation tunnels make flow observations and measurements of hydrodynamic forces much easier than in full scale trials, because the environment is much more controllable. Also, with every new set of experimental data that becomes available, the baseline for the validation of numerical methods becomes more accurate. However, conducting model tests for the prediction of full scale performance is not without flaw. Scale effects exist that are not fully understood, causing discrepancies between model and full scale, however it is still our most accurate method of performance prediction (Olofsson 1996).

Two of the more recent experimental investigations were those of Olofsson (1996) and Ferrando (2001). Olofsson used a strain gauge system mounted inside the hub of the propeller to measure the dynamic and time averaged forces on each individual blade of the propeller. He conducted his experiments in a cavitation tunnel where he varied the angle of inclination, angle of yaw and advance ratio of the

propeller, while keeping immersion constant at $I_T = 0.33$. His data is one of the most complete sets publically available to date. Oloffson concluded that, efficiency “increased with increasing shaft yaw angle until reaching a point of maximum efficiency. At this point the resultant horizontal side force of the propeller is identically zero.” (Olofsson 1996). Also, at this yaw angle of maximum efficiency it apperat that all of the velocities induced into the wake are travelling in the free stream velocity direction, meaning all of the momentum or energy losses are going into thrust, not vertical or side force (Olofsson 1996).

Marco Ferrando (2001) measured the time averaged forces while varying immersion, angle of inclination and advance ratio, and keeping the angle of yaw fixed at 0° . These experiments were also conducted in a free surface cavitation tunnel. Based on these tests he developed polynomial regressions (eqns 13 and 14) to account for immersion and inclination in preliminary estimations of thrust and torque. He also found that by non-dimensionalizing K_T and K_Q using submerged blade area, the K_T and K_Q curves for different submergences and shaft inclinations tend to collapse into one unique curve.

Using equations 13 and 14, the expected trends of K_T and K_Q were developed for the test propeller, described in section 2.0, and are shown in figure 3. For this propeller, it was determined that the values of K_T , K_Q and η should all reach their max values between advance ratios of $1.0 \leq J \leq 1.6$.

$$K_T = -0.691625 * J + 0.794973 * \frac{P}{D} + 0.870696 * J * \frac{P}{D} - 0.395012 * J^2 - 0.515183 * \left(\frac{P}{D}\right)^2 \quad (13)$$

$$K_Q = \frac{\left(-0.300453 * J + 0.543738 * \frac{P}{D} + 0.877638 * J * \frac{P}{D} - 0.649314 * J^2 - 0.208974 * \left(\frac{P}{D}\right)^2\right)}{10} \quad (14)$$

$$\eta = \frac{K_T}{K_Q} * \frac{J}{2\pi} \quad (15)$$

Where:

η = Propeller Efficiency

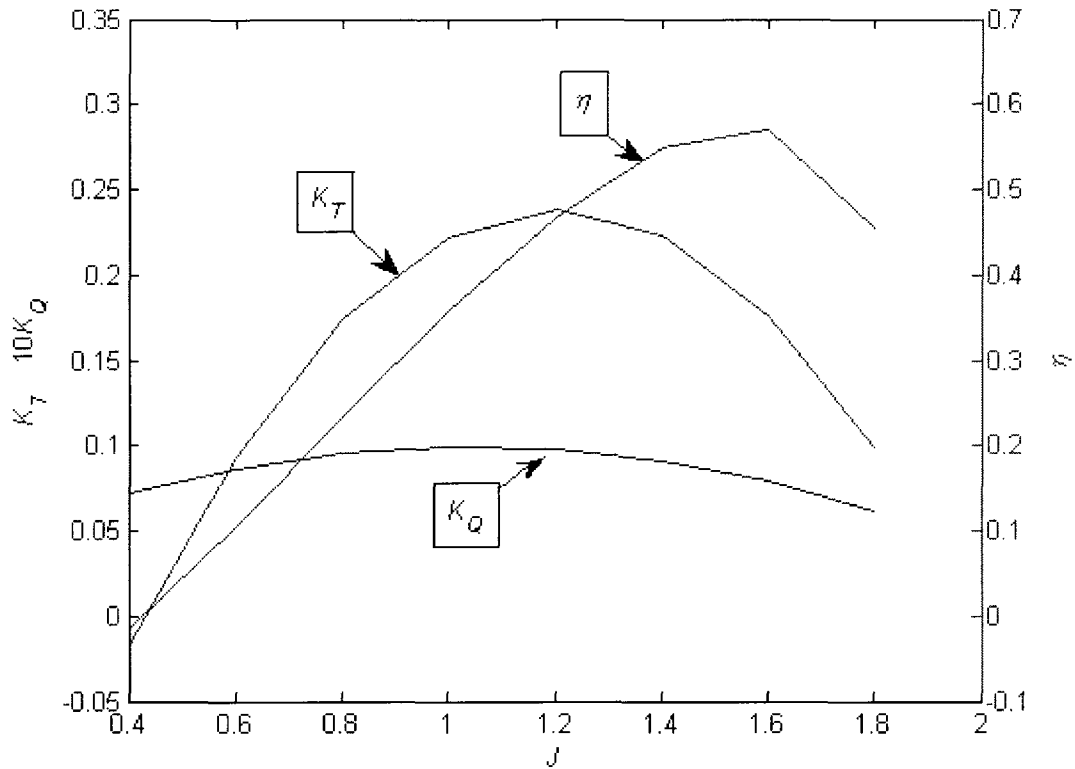


Figure 3. Predicted trends for K_T , (13), K_Q (14) and η (15), for a general case with $\Psi = 0^\circ$, $\gamma = 0^\circ$ and $\left(\frac{P}{D}\right) = 1.9$

For the project presented, time averaged and dynamic forces on the propeller were measured while the depth of submergence, angle of yaw and angle of inclination are systematically varied over a range of advance ratios. While Olofsson and Ferrando each varied the setup of their propeller in 2 degrees of freedom (i.e. yaw and inclination or submergence and inclination), this plan of research involves varying the propeller setup in 3 degrees of freedom. There is not a set of publically available data to date, in which all three degrees of freedom are explored in the same experiment. This will give new insight into the relationships between the performance of SPPs and varying angles of yaw and inclination as well as different depths of submergence.

The characterization of the performance of this propeller through systematic testing and analysis of the results will contribute to the development of methods for performance prediction of SPPs by providing more data for verification as well as possible new observations of flow phenomenon.

Since the experiments were conducted in a non-pressurized towing tank facility, the atmospheric pressure cannot be varied in the test section; therefore the cavitation number cannot be completely controlled. The effects of the cavitation number are primarily seen in the partially cavitating regime. Kruppa (1972) states that when the propeller becomes completely ventilated, the effects of the cavitation number are minimized, and Shiba (1953) has found that the fully ventilated regime has been found to occur at:

$$F_n > 3$$

After this Froude number is exceeded the ventilated cavities have assumed their ultimate form and the role of gravity influenced flows have asymptotically approached final values (Olofsson 1996). Another parameter that that can affect the performance of SPPs is the Weber number.

$$W_n = \frac{nD}{\sqrt{\frac{\sigma \kappa}{\rho D}}} \quad (16)$$

Shiba (1953) shows that Weber number dependency becomes negligible at values of:

$$W_n > 1.8 \times 10^2$$

For Weber number, Froude number and cavitation number effects to be negligible, their respective values must exceed the previously stated critical values. Based on these preliminary parameters, an experimental approach was developed

Since these experiments were unique, and tested propellers in a different way than previous researchers, a new test apparatus needed to be designed. The details of its design and design process, along with a description of the test facility, test propeller, and experimental approach are described in section 2. The methods used to analyze the data collected from the SPP testing can be found in section 3, while a discussion of the results obtained from this analysis can be found in section 4. Conclusions drawn from the experiments as well as suggested future work can be found in section 5.

2.0 EXPERIMENTAL APPROACH AND APPARATUS

2.1 Propeller

The design of the test apparatus started with propeller selection. Preliminary cost analysis was conducted to determine the feasibility of contracting the fabrication of a propeller specifically for research purposes. This cost was on the order of fifteen to twenty thousand dollars, which was out of the scope of budget for the project.

After contacting many surface piercing propeller designers and manufacturers, searching for an affordable alternative to custom fabrication, a propeller was donated by Dewald Propellers. The propeller, which can be seen in figure 4, is a cleaver style SPP, designed and built by Dewald, for Formula 1 outboard racing boats. It has a blade section typical to that of a surface piercing propellers, having a very sharp leading edge and a thick blunt trailing edge.

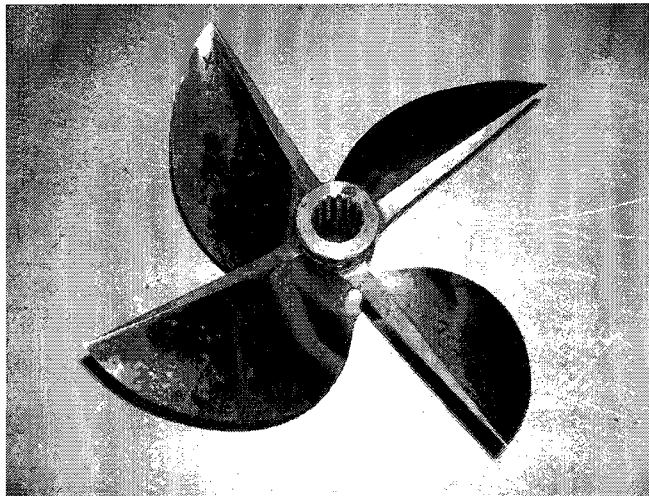


Figure 4. Test Propeller

The principal characteristics of the propeller, seen in table 1, were found using a Hale MRI propeller measurement system, the detailed measurements can be found in appendix A. It is important to

note that the pitch to diameter ratio $\left(\frac{P}{D}\right)$ is about 1.9, for the model propeller. This value is normal for small, racing powerboats, with extremely high power to weight ratios, but quite high as compared to the propellers tested in the research of Olofsson (1996), and Ferrando (2001). Even though the design of this propeller does not scale well for applications to marine vehicles like T-Craft, the large forces produced by the Dewald propeller serve as a good tool for designing a robust test apparatus.

Table 1. Propeller characteristics

Rotation: Left Handed	Diameter: 24.64 cm
Pitch: 46.5 cm	Blade #: 4
Bore: Splined	Material: Stainless

2.2 Facility

After a test propeller was obtained, test facilities were researched to determine a proper test site for the SPP experiments. Due to the cost constraints of the project, options were limited to the Florida Atlantic University (FAU) towing tank and the Stevens Institute of Technology's Davison Laboratory, who kindly offered their laboratory and facility personnel services to FAU's SPP research team as an ONR National Naval Responsibility for Naval Engineering (NRRNE) partner institution.

The factors, other than cost, that affect the facility selection are logistics, size of the tank as well as speed and load carrying ability of the towing carriage. For logistical reasons, the researchers would have liked to conduct experiments on the FAU-Seatech site, so a thorough investigation was conducted to determine the feasibility of using the Seatech Hydrodynamics Laboratory for the intended experiments.

Since testing was to be conducted in a non-pressurized facility, the cavitation number cannot be completely controlled. However, the effects of the cavitation number are seen primarily when the propeller is in the partially ventilated condition. When the propeller becomes fully ventilated at Froude numbers greater than 3 (Shiba 1953), the cavitation number becomes the Froude number (Kruppa 1972), which can be used to scale the results of the model experiment. To determine the minimum carriage speed required for testing, the critical propeller rotational rate was calculated based on a Froude number of 3, which is required to maintain the fully ventilated condition. Then, based on the rotational rate, speed of advance

could be calculated base on the range of advance ratios required. The results of this calculation of equation 17, can be seen in table 2. Weber number (eqn 16) was also checked to make sure it exceeded a value of 180, to minimize scale effects.

$$U_A = F_n J \sqrt{gD} \quad (17)$$

Table 2. U_A required to meet critical F_n

J	U_A (m/s)
0.8	3.73
1.0	4.66
1.2	5.60
1.4	6.53
1.6	7.46
1.8	8.39

The range of advance ratios, used in the calculation of required speed of advance, was found by calculating K_T and K_Q using equations 13 and 14, the points of max efficiency can then be estimated using equation 15. From this prediction, a proper range of advance ratio values can be determined.

Based on these results, it was decided that the modifications required for the FAU towing carriage to obtain the required speed were out of the scope of this project. Due to its 18 m tank length, even if the desired speeds were achieved, the steady state portion of the run would be too small for effective data capture. Based on this analysis, it was apparent that the facility at the Stevens Institute of Technology, seen in figure 5, was the better option for the SPP test site.

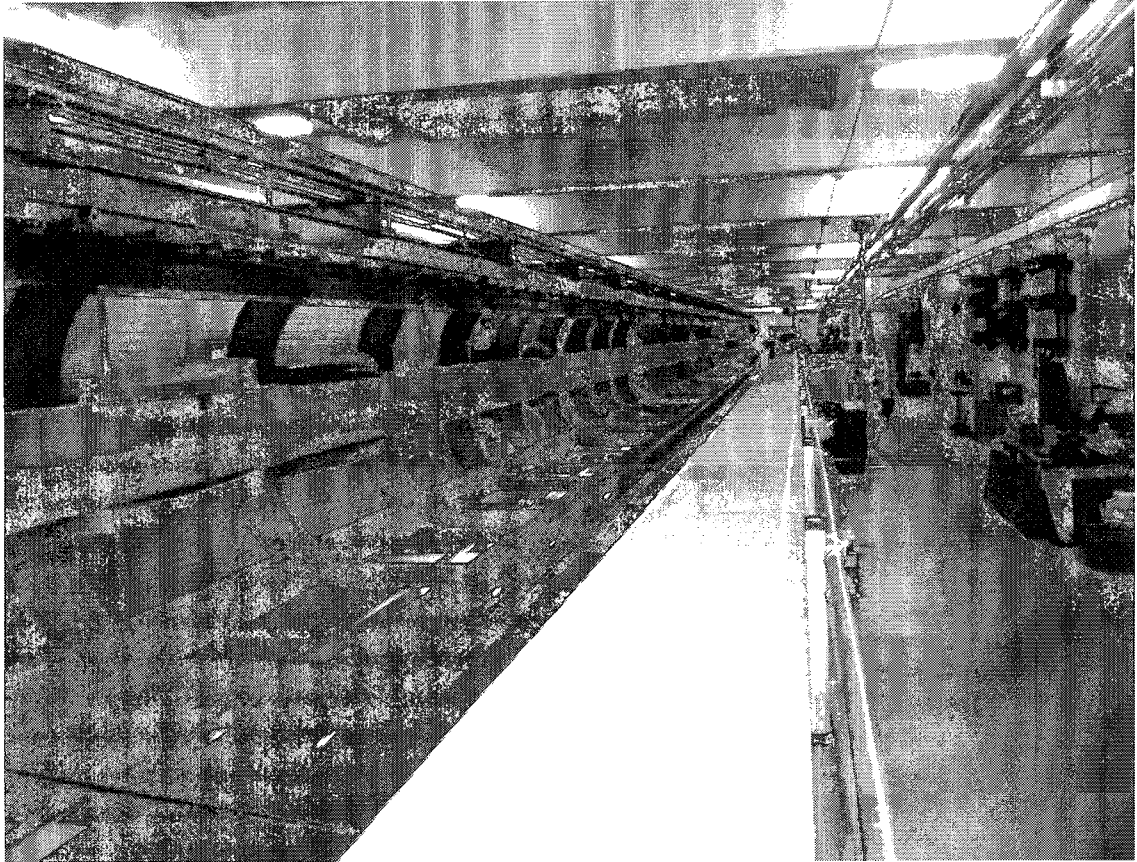


Figure 5. Stevens Institute of Technology's Davidson Laboratory

The high speed towing tank in the Davidson Laboratory at the Stevens Institute of Technology is 4.9 m wide, 2.0 m deep and 95.4 m long. The towing carriage is one of the fastest in the world, capable of speeds up to 30.5 m/s with a speed control of 0.003 m/s. It can also hold approximately 225 kg of mass.

The towing carriage wheels sit atop the precision rail, which is supported on an overhead steel beam as seen in figure 6. The carriage is connected, by a steel cable, to a 112 kW electric motor. The motor sits at the far end of the tank, pulling the carriage down the length of the basin.

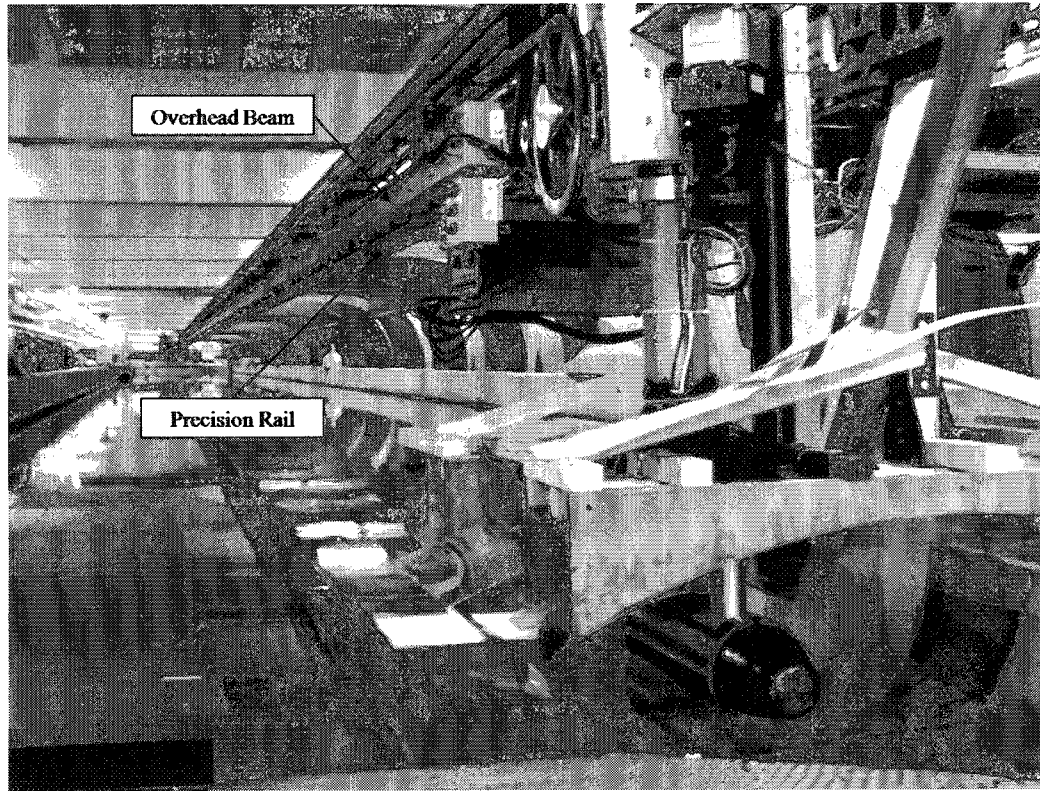


Figure 6. Precision rail and overhead beam in Davidson Laboratory

The laboratory also has free surface cameras as well as underwater cameras that are triggered by a voltage signal that is sent when the carriage reaches the section of the tank, about half the distance down, that is equipped with an underwater viewing window. For more information on the camera setup refer to Altamirano (2010).

2.3 Apparatus Design

The test stand was designed and fabricated at Florida Atlantic University's Seatech campus, in Dania Beach, Florida. Assembly and functional testing of the apparatus was also conducted at Seatech (figure 7), to ensure proper basic operation before disassembling the device and transporting it to the test facility in Hoboken, New Jersey, for reassembly and experimentation.

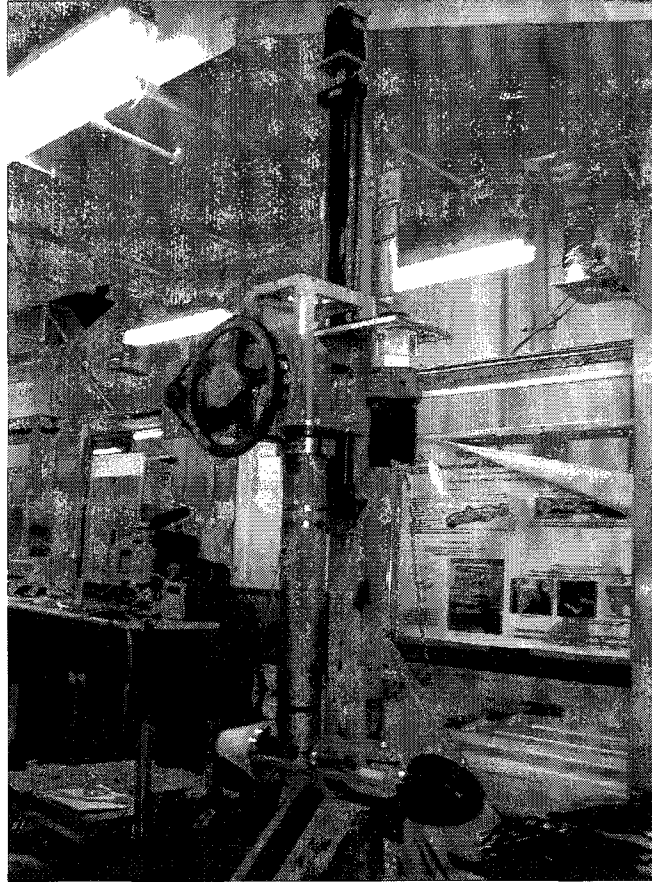


Figure 7. Assembling for functional testing at Seatech

The investigation of surface piercing propellers desired by the Office Naval Research calls for the variation of advance ratio (J), shaft yaw angles (Ψ) from 0° to 30° , shaft inclination angles (γ) from 0° to 15° and tip immersion ratio values (I_T) up to 1.5. To develop open water performance curves, traditional propeller testing usually consists of having the propeller set at a fixed depth (normally $I_T = 1.5$, which is considered deeply submerged), with angles of shaft yaw and inclination set at 0° . The experiments are then conducted over a range of advance ratios suitable to the performance of the particular propeller.

The design of a typical propeller boat reflects the type of testing required by traditional open water experiments. The only adjustments usually conducted are depth of submergence and rotation rate. If the yaw angle were to be increased, the foil shaped strut, commonly found on traditional propeller test apparatus (figure 8), would generate very unacceptable side lift for the advance speeds and yaw angles required in the present investigation.

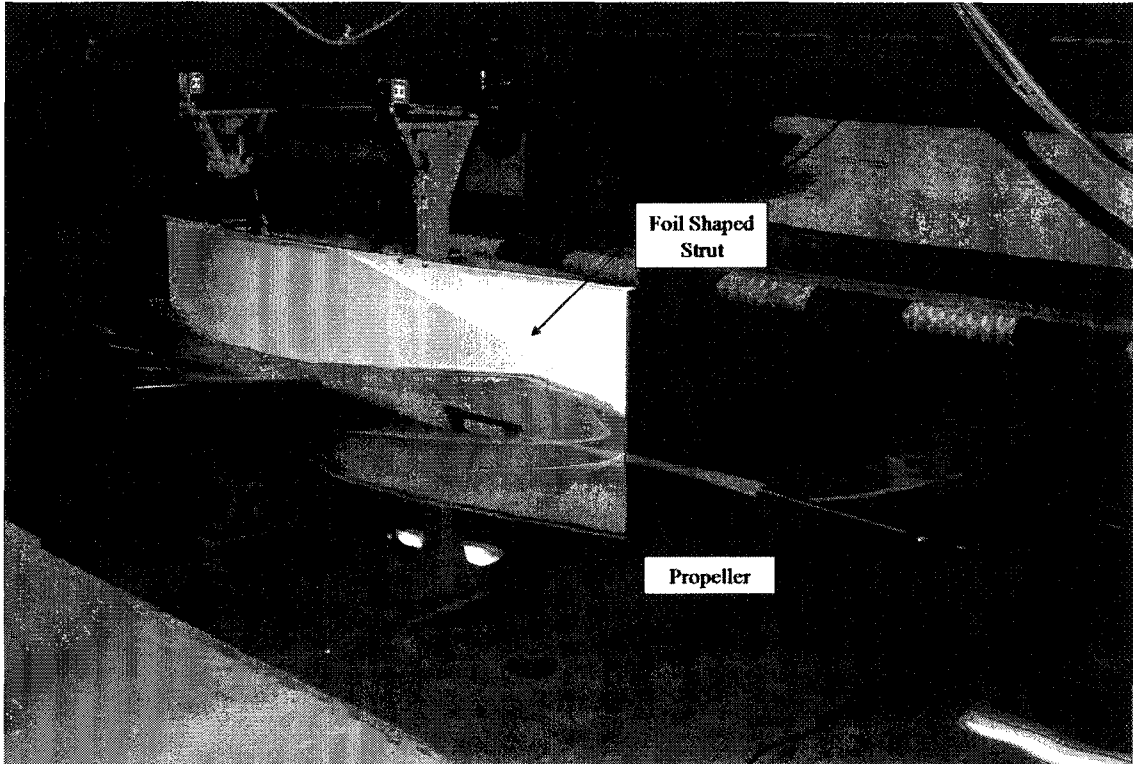


Figure 8. Traditional propeller boat used at Webb Institute of Naval Architecture

Since deeply submerged propellers do not produce significant side loads as compared to thrust and torque loads, traditional propeller boats have fairly low yaw moment stiffness. SPPs, on the contrary, produce large side forces, which would require a test apparatus to be much more rigid in comparison to traditional propeller boats. To accommodate the ability to change shaft yaw angle, a new propeller boat had to be designed.

The objective of the new test stand was to control the required parameters (γ , Ψ , I_T and n) throughout the range of motion desired by the Office of Naval Research. The apparatus was also required to accommodate the high thrust, torque and side loads produced by the aggressive design of the Dewald propeller. In order to increase setup speed and safety between runs, the researchers decided to design a propeller boat that used automatic control to change the variable experimental parameters, reducing manual manipulation of the test apparatus.

2.3.1 Design Load Estimation

The calculations of the time average forces of thrust and torque were completed using a regression for 4 bladed SPPs, developed by Marco Ferrando (2001). Using the results of this work, the maximum thrust and torque forces that were calculated were at the fully submerged condition. To calculate the dynamic forces of thrust, torque and side force, an analysis of the results of Oloffson (1996) was conducted. By plotting the ratio of dynamic forces to time average forces for each test condition, as seen in figure 9, it could be seen that the large majority of the dynamic forces of thrust, torque and side forces were between 3 and 4 times the average load. Using this approximation, the time average forces calculated using equations 15 and 16 [ref Ferrando] was multiplied by 5 for a conservative estimate on the maximum dynamic load.

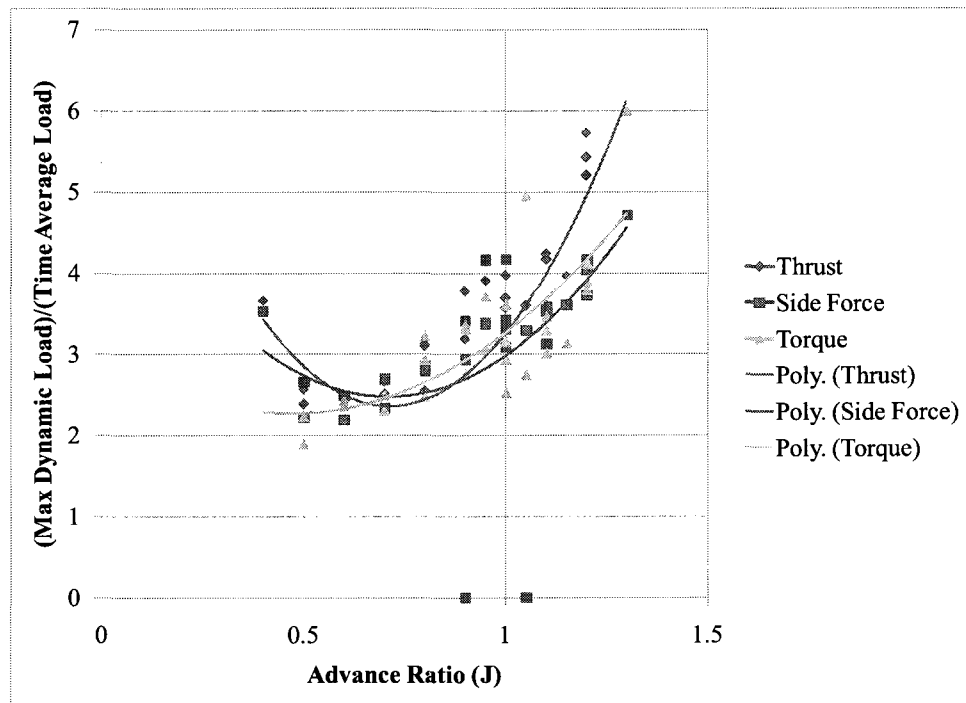


Figure 9. Ratio of dynamic load to time average load versus advance ratio

When the propeller is fully submerged, the torque and thrust loads are very high. When the propeller is partially submerged, the dynamic loads are much higher than the fully submerged condition, while the overall torque and thrust are much lower. In the experiments of Oloffson (1996), the propeller is partially submerged, with a tip immersion ratio of 0.33. So by coupling the dynamic load ratio taken from

the data in Oloffson (1996), with the force and torque estimates for a fully submerged propeller from Ferrando (2001), the design loads were conservative.

2.3.2 Resonant Frequency Analysis

The high dynamic loads of SPPs from the propeller entering and exiting the air/water interface can cause serious vibration problems, making a vibration analysis imperative for the design. After sizing the strut and propeller shaft based on the loads analysis, the natural frequencies of both components were checked for resonance with the first 4 harmonics of propeller frequencies: 25, 50, 60 and 80 Hz. It is important to note that the hydrodynamic added mass of both of these components were taken into account for the calculations.

For the propeller shaft assembly, the unsupported length of the shaft and the weight of the end mass (propeller, spline and fairing cone seen in figure 10), were fixed, the only parameters that could be varied to change the resonant frequency for this assembly were prop shaft diameter and material. The unsupported length of the propeller shaft was fixed due to the ITTC requirement of having 1.5 propeller diameters (International Towing Tank Conference 2002) between the propeller, and the start of the propeller boat housing. This is to ensure the pressure field created by the fairing does not approach the propeller, affecting its open water performance.

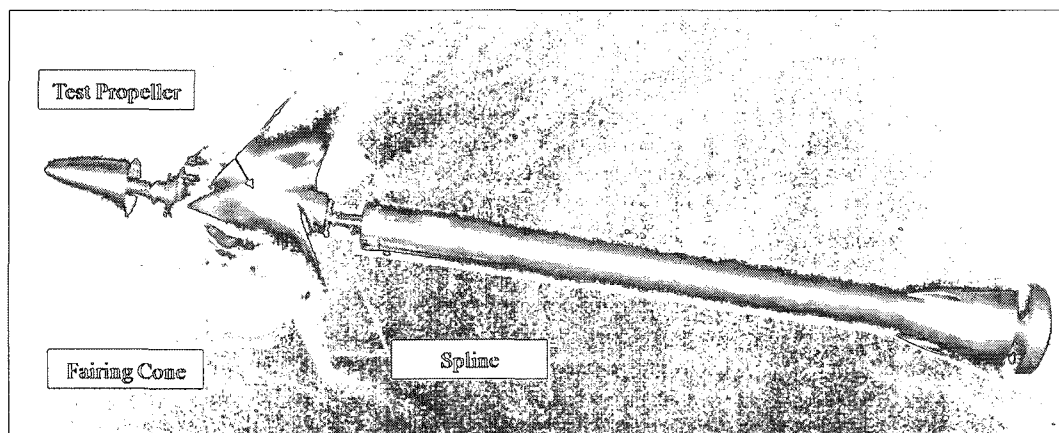


Figure 10. Propeller and shaft assembly

After the propeller shaft diameter and material were finalized, the strut was analyzed. The length of the strut was controlled by geometry so that material, diameter and wall thickness were the variable parameters affecting natural frequency. The strut had to be long enough to ensure a depth of 1.5 propeller diameters when the test stand was at the bottom of the linear translation stage. The diameter of the strut was desired to be as small as possible, reducing its drag, spray and wake.

Upon completion of the analysis, it was determined that the natural frequencies of both aluminum propeller shaft and aluminum strut, as sized for the loads analysis, were about 263.6 Hz and 218.9 Hz respectively. This was determined to be too close to the propeller frequencies to be considered safe from resonance. To correct this, the propeller shaft and strut material were changed to stainless steel, giving them natural frequencies of 380.8 Hz, and 333.6 Hz, respectively. This put the natural frequency of both components in an acceptable range of about three times the resonant frequency (300Hz). The system was then checked to make sure that there was no resonance with the vortex shedding frequencies of the strut. The Matlab code used to perform these calculations can be found in appendix B.

2.3.3 Mechanical System Design

The mechanical system, which drives the propeller position can be separated into three subsystems; depth of immersion control, angle of inclination control (figure 11), and angle of yaw control (figure 12).

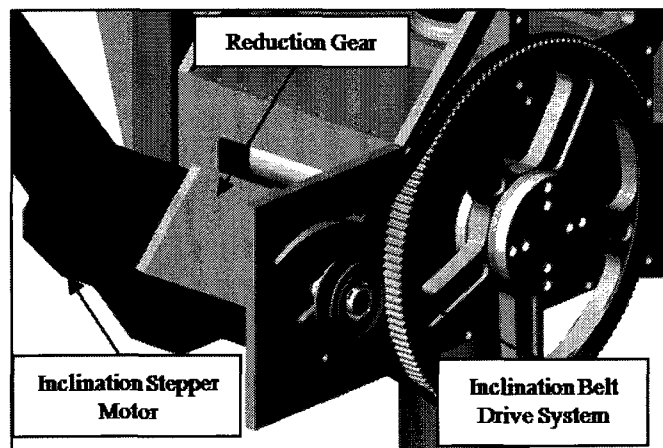


Figure 11. Inclination mechanical system

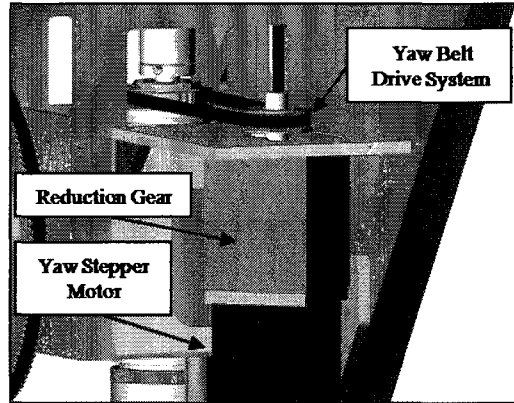


Figure 12. Yaw mechanical system

The depth of immersion is driven by a stepper motor that is coupled to a linear translation stage (figure 13). The linear translation stage is a precision ball bearing type with a 5 mm pitch, and the coupler was fabricated for this specific application. Before any of the test apparatus was attached to the mounting block, a vertical load, static holding test was conducted. The assembly successfully held 225 kg of mass, which was 90 kg more than what was calculated for the preliminary load estimate. To stiffen the linear translation stage, once it was mounted to the carriage, a support structure, seen in figure 14, was designed to distribute the eccentric loading.

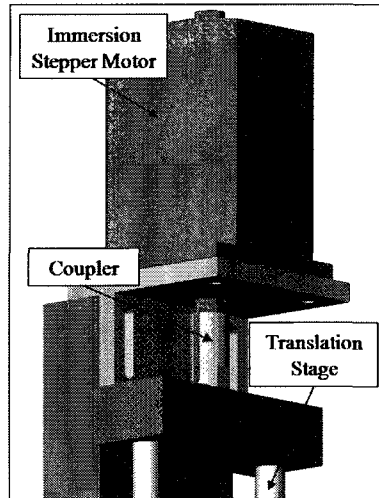


Figure 13. Immersion mechanical system

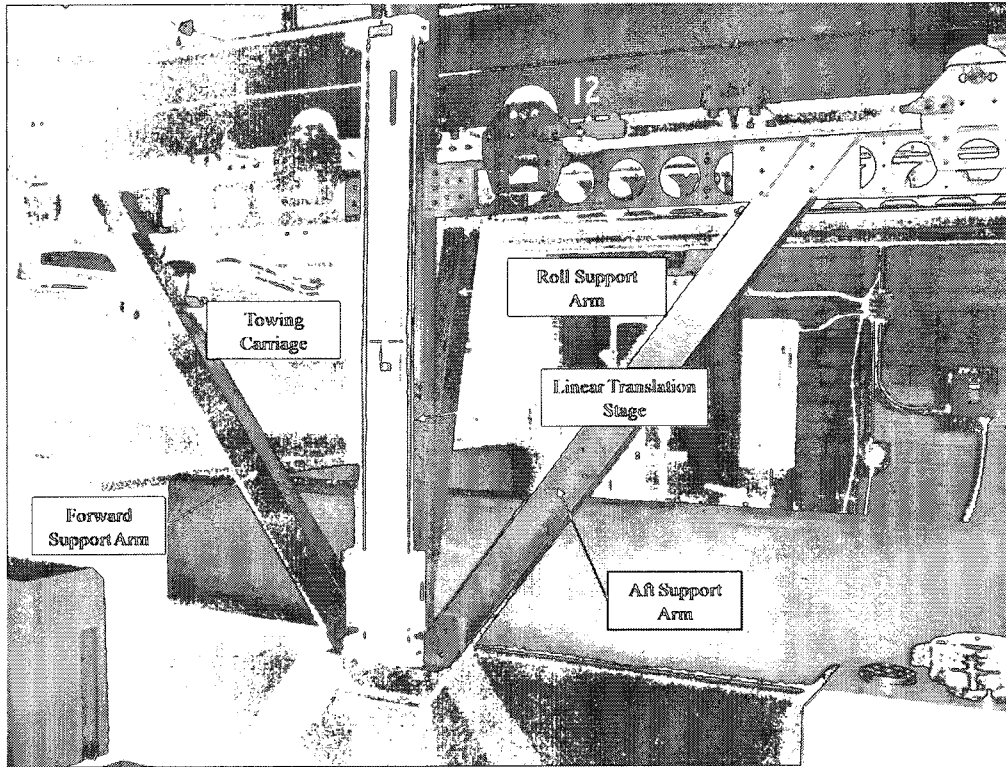


Figure 14. Immersion system and support structure

The propeller test apparatus yaws about a 73 mm diameter pipe, which runs vertical. Tapered roller bearings are fit around the yaw pipe and allow the strut and lower assembly to rotate together. The races of the bearings are press fit into the upper assembly plates, while the cages fit over the ends of the pipe and are held into the race when the assembly is bolted in together as seen in figure 15.

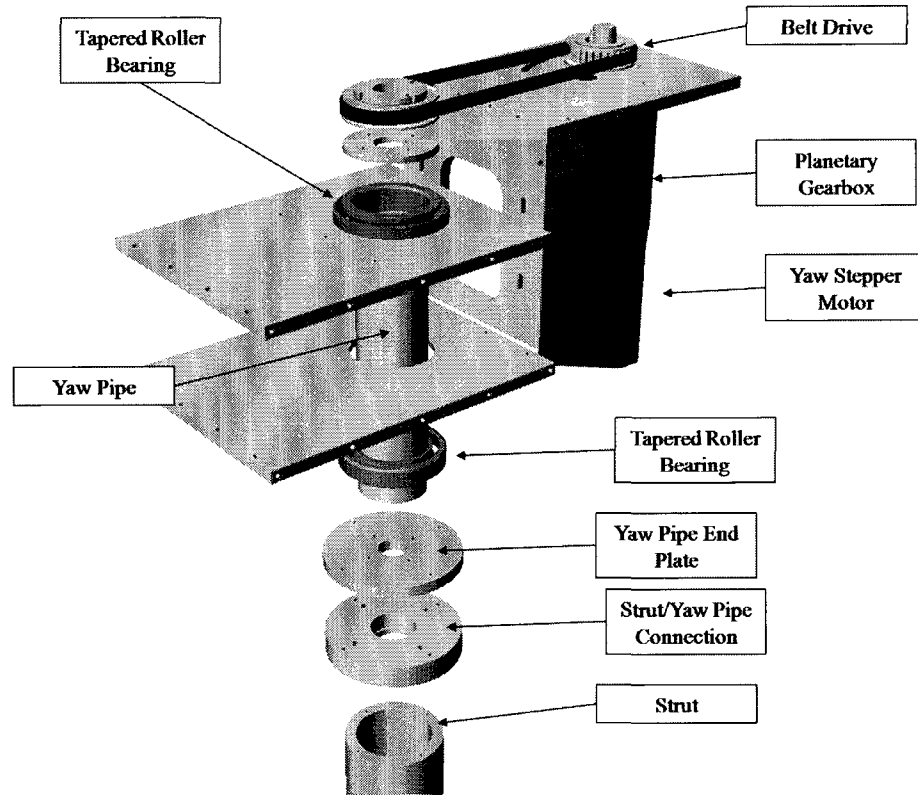


Figure 15. Yaw drive, exploded view

The yaw rotation is powered by a stepper motor, run through an inline planetary reduction gear. The output shaft of the reduction gear is fitted with a 25 tooth, 8mm pitch sprocket, driving a 32 tooth 8mm pitch sprocket via a 12mm wide carbon fiber reinforced belt. The 32 tooth sprocket is bolted directly into the yaw pipe, driving the yaw positioning. The gear ratios and holding torques of the yaw system can be found in table 3.

Angle of inclination is driven similarly to the yaw system. A 73 mm diameter pipe runs horizontally, bolted into the linear translation stage mounting block on one end and the inclination sprocket assembly on the other. The races are press fit into the upper assembly plates and the cages are held into the race when the outer sprocket is bolted in place.

Inclination rotation is powered by a stepper motor, run through an inline planetary reduction gear. The output shaft of the reduction gear is fitted with a 25 tooth, 8mm pitch sprocket, driving a 140 tooth 8mm pitch sprocket via a 12mm wide carbon fiber reinforced belt. The 140 tooth sprocket is bolted into

the inclination pipe driving the inclination positioning. An exploded view of the inclination drive is shown in figure 16. The gear ratios and holding torques of the inclination system can be found in table 3.

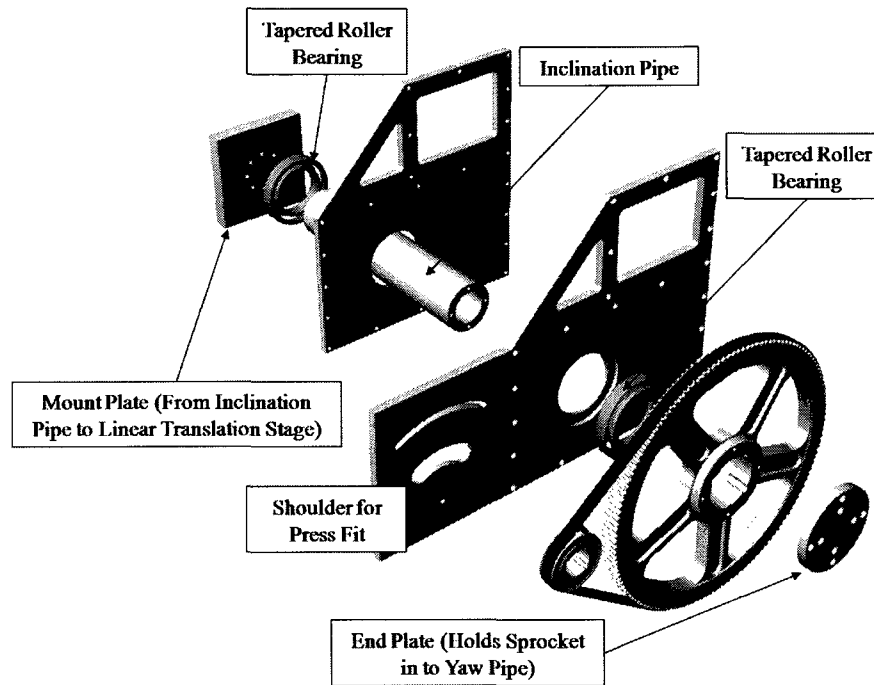


Figure 16. Inclination drive, exploded view

Table 3. Motion control system holding torques and gear ratios

Motion	Stepper Motor Torque (N-m)	Gear Ratio	Belt Drive Ratio	Total Holding Torque (N-m)
Pitch	20.99	21:1	5.6:1	2468.20
Yaw	20.99	21:1	1.28:1	564.16
Immersion	20.99	5:1	N/A	20.99

The motor for the propeller is a 4.5 kilowatt, 24 Volt, brushed DC motor. This motor was chosen because of its low weight (133 Newtons), and cost for the amount of power delivered. The DC propulsion motor is coupled to a right angle drive unit, seen in figure 17, with dual shaft output. This connection is via a 25.4 mm diameter shaft that runs inside of the yaw pipe and strut, supported near the center by a tapered roller bearing. The shaft is connected to the motor and right angle drive via flexible couplers for ease of

alignment. A custom fabricated coupler is threaded onto the right angle drive and then secured with three cone tipped set screws positioned 120° apart along the coupler's circumference, also serving for rotational alignment of the propeller shaft. The coupler is held by a sealed bearing, and bolted into the force transducer/prop shaft assembly, which can be seen in figure 18. An illustration of the upper part of the propeller drive line can be seen in figure 19, and the lower part in figure 20.

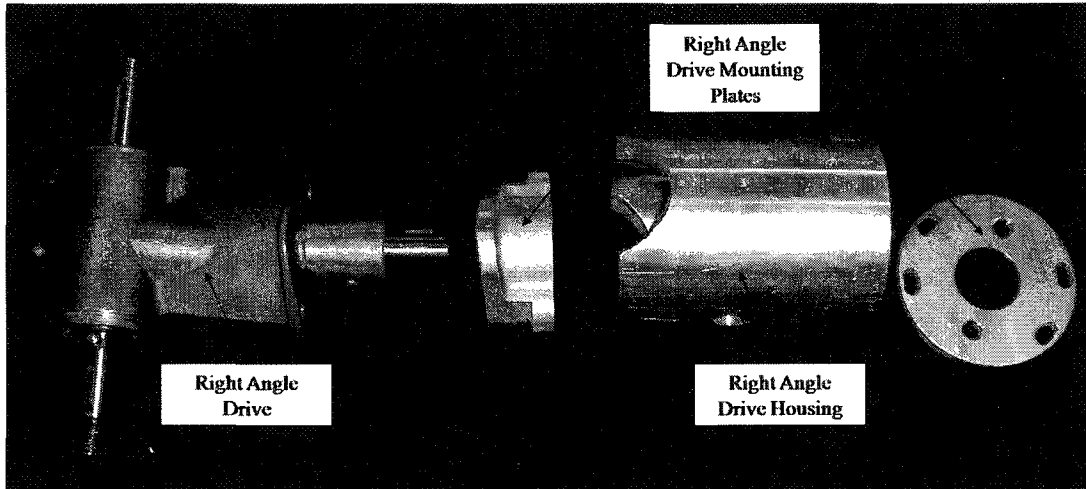


Figure 17. Right angle drive and mounting assembly

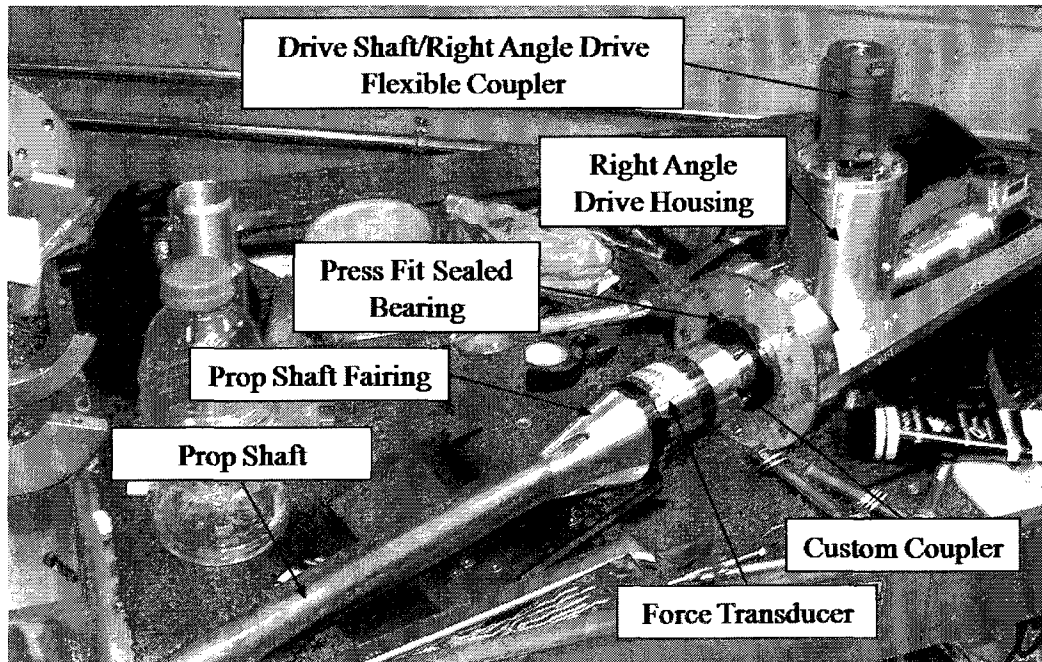


Figure 18. Lower assembly, still in assembly process

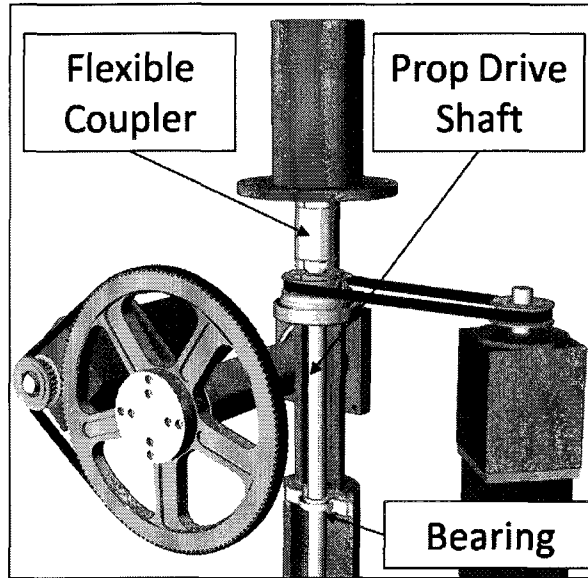


Figure 19. Upper part of propeller drive

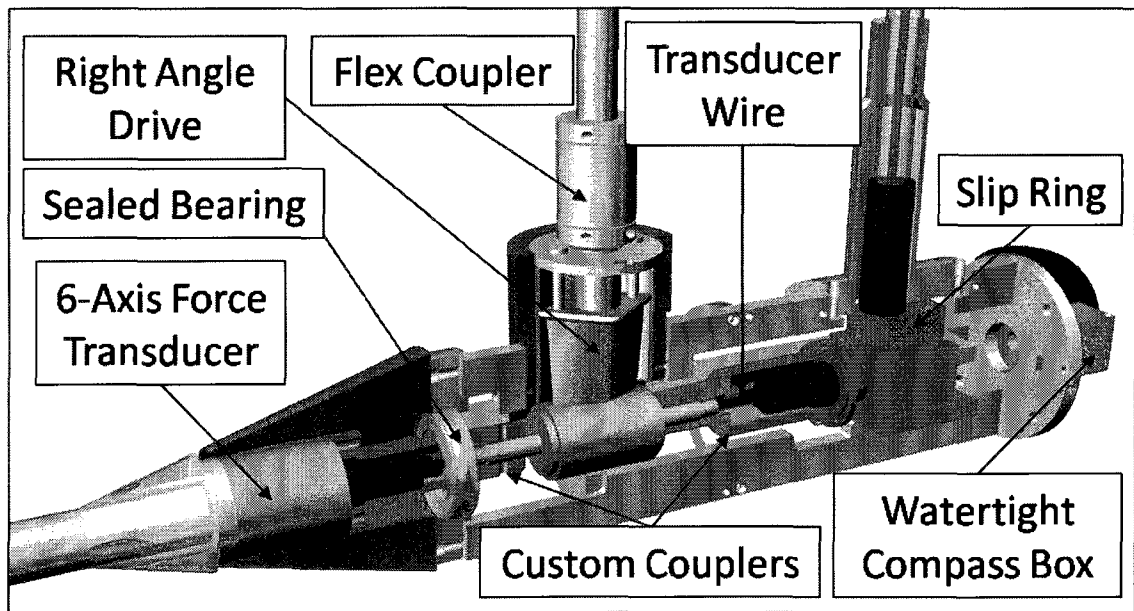


Figure 20. Lower part of propeller drive

The loads previously stated were used to design the structural and mechanical systems of the SPP test rig. It was determined that the structure should be mechanically fastened for ease of assembly and disassembly, since the fabrication site and the test site are not located in the same facility. Also mechanical fastening will eliminate distortion caused by welding, which can lead to alignment issues.

Figure 21 shows the a rendering of the completed design, while figure 22 shows the actual prototype.

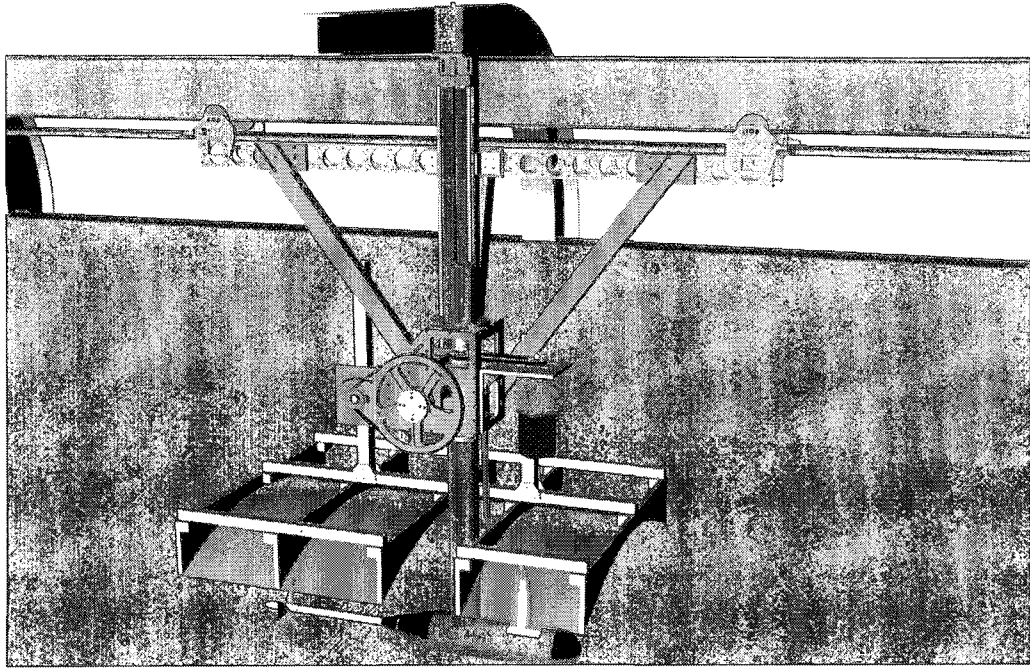


Figure 21. Full assembly rendering

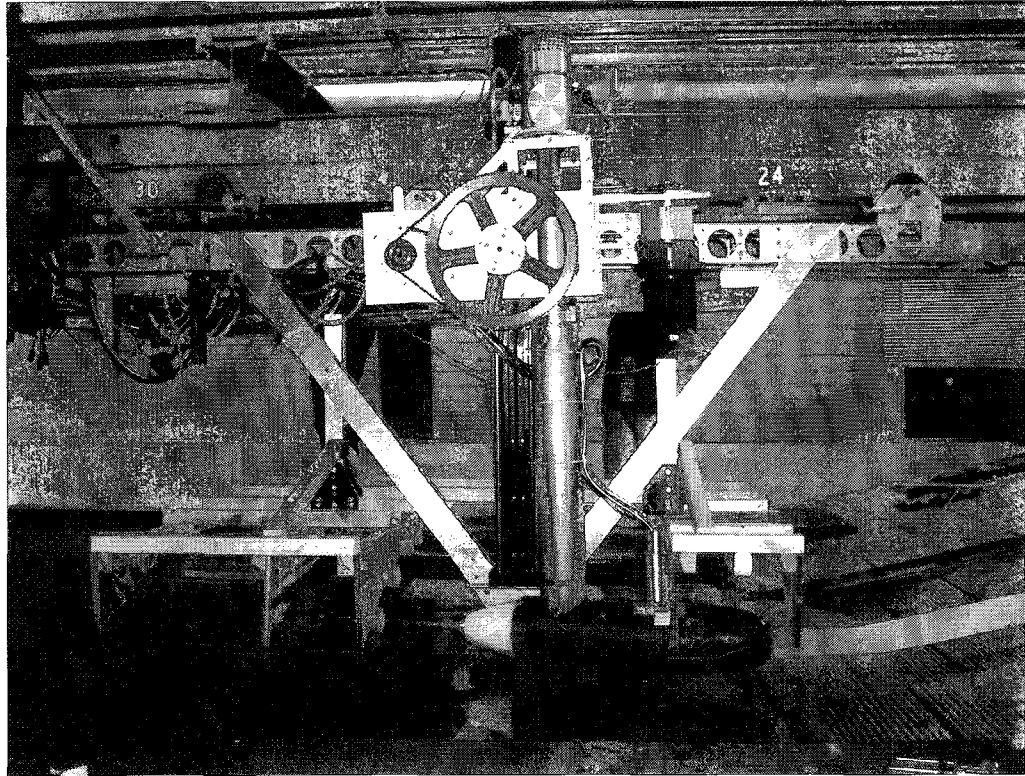


Figure 22. Prototype assembled, excluding propeller, at test site

2.3.4 Control System Design

The four variables that are being controlled by the test apparatus are propeller speed, yaw angle, pitch angle and tip immersion ratio. These parameters were controlled open loop, using a visual interface on a shore based laptop, which used Remote Desktop to wirelessly connect to the carriage mounted laptop. The carriage mounted laptop was connected to the drivers and controllers of the four previously mentioned systems using USB connections. The arrangement of the carriage mounted electronics can be seen in figure 23. It is noteworthy that the carriage laptop hard drive was replaced with a solid state hard drive, so that the accelerations of the carriage would not corrupt the data collection.

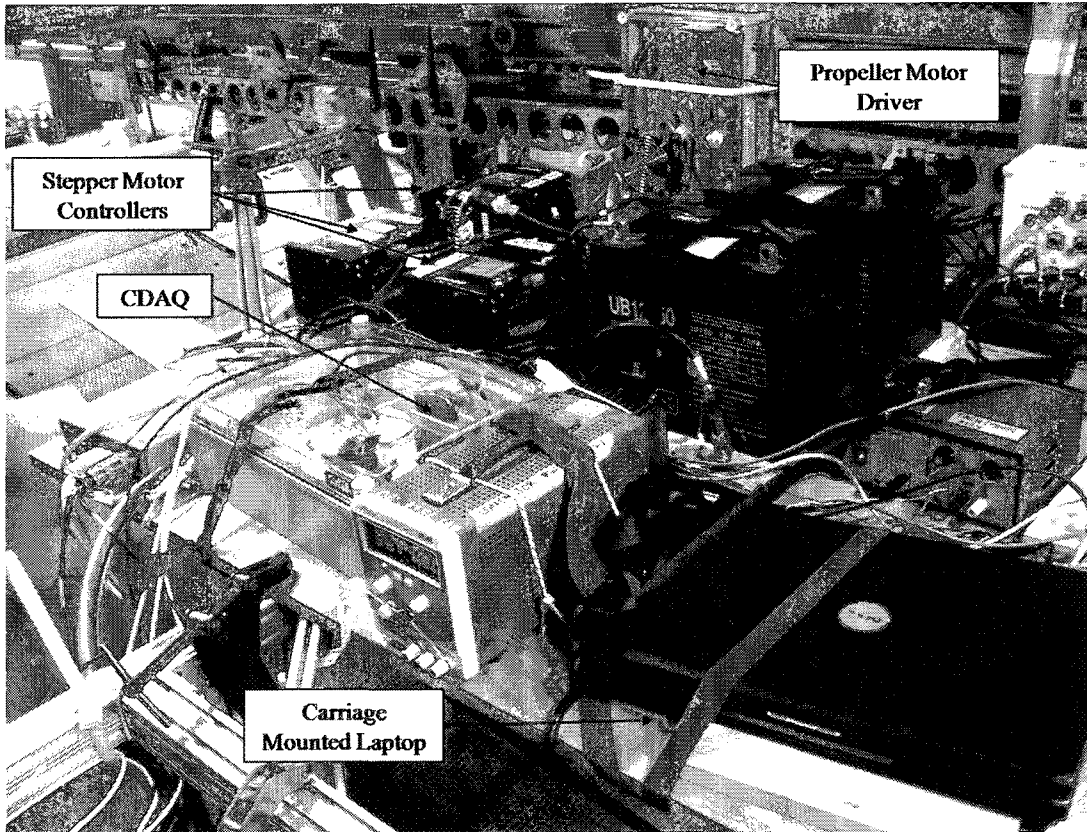


Figure 23. Carriage mounted electronics

The angle of yaw was open loop controlled, as seen in figure 24, using a protractor to measure the rotation of the yaw sprocket. Since the precise gear ratio of the yaw belt drive system was known to be 1.28:1, it was a matter of simple multiplication to calculate the angle of rotation needed at the sprocket, to give the desired yaw angle at the propeller shaft. To rotate the sprocket, the user would input the number of steps the motor was desired to rotate, where 21000 steps is 360° of shaft rotation. After the movement was made, the operator would check the sprocket mark versus the desired angle on the protractor to ensure the propeller shaft was rotated to the proper angle.

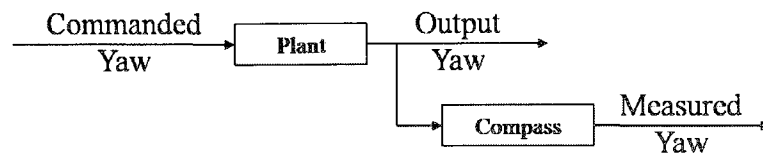


Figure 24. Yaw control system diagram

Inclination angle of the shaft was also open loop control, as shown by figure 25. The operator uses feedback from the compass/inclinometer unit, to choose the number of steps the inclination stepper motor needs to rotate. The feedback is a visual interface that displays heading, inclination and roll in decimal degrees.

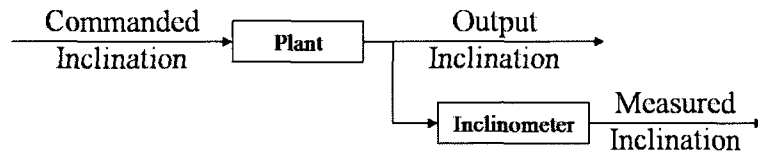


Figure 25. Inclination control system diagram

The compass and inclinometer are in a single unit, the TCM 5 by PNI Sensor Corporation. It has an accuracy of 0.3° in heading with a resolution of 0.1° and a repeatability of 0.05° . The tilt and roll use an inertial measurement unit, and the accuracy is 0.2° with a resolution of 0.01° and a repeatability of 0.01° . The unit is mounted inside the lower fairing, as seen in figure 26, in line with the propeller shaft.

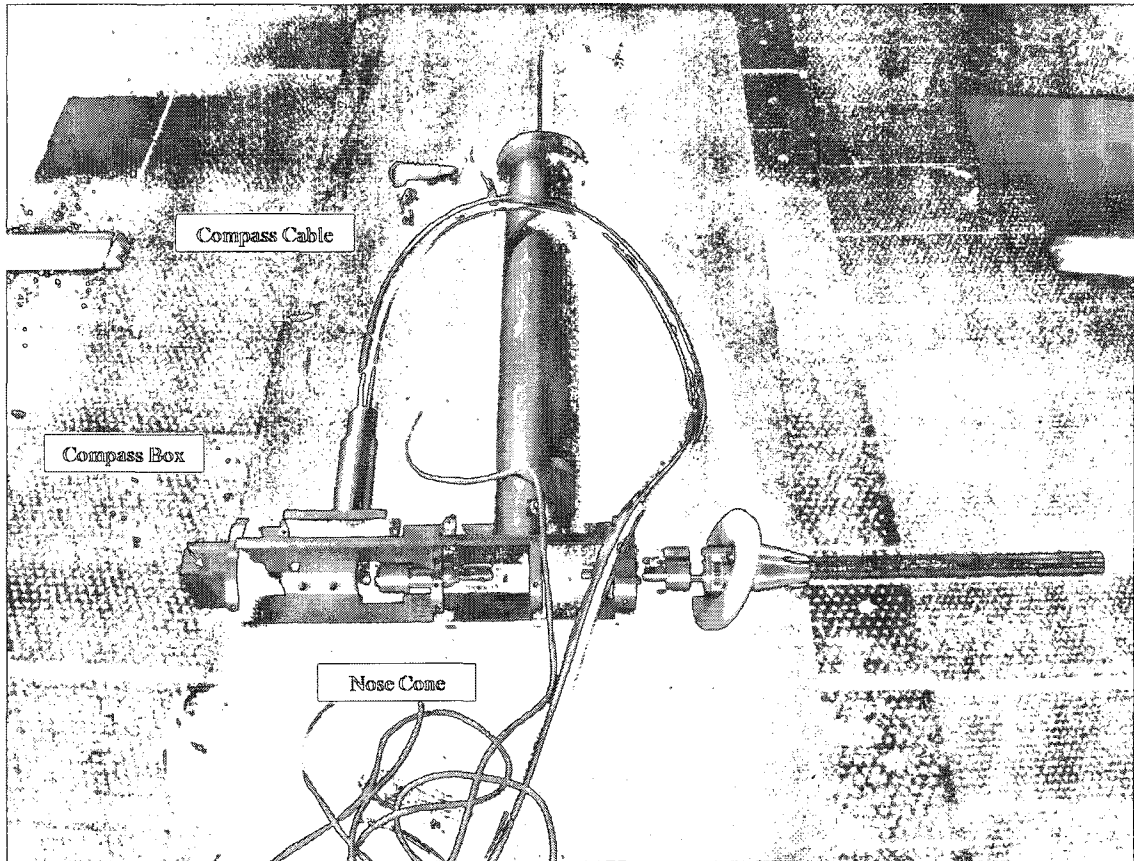


Figure 26. Bottom assembly before fairings

Depth of immersion was also controlled open loop using an interface that required the operator to input the number of steps the stepper motor was desired to turn. Since the linear translation stage was pitched at 5 turns per 25.4 mm, and the stepper motor controller was set to 1000 steps to 360° of shaft rotation, 5000 steps corresponded to 25.4 mm of vertical travel. To verify the submergence of the propeller, the operator used a caliper to measure from the tip of the propeller (with the blades at 0°, 90°, 180° and 270°), to the free surface of the water.

The propeller motor is open loop controlled, as shown in figure 27, with the tachometer, which is built into the slip ring, providing a measurement of the actual shaft rotational rate. The slip ring used is a Michigan Scientific Model SR20AW/T512/AX weatherproof slip ring and encoder assembly. This unit has 20 slip ring connections, a 512 pulse/revolution encoder and built-in encoder electronics with analog outputs for shaft velocity and angular position.

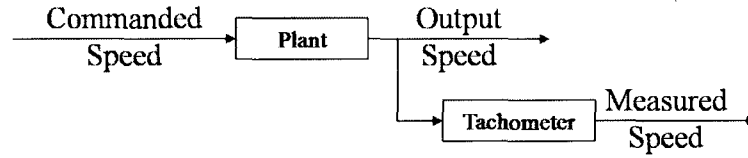


Figure 27. Propeller motor control system diagram

Using a Graphical User Interface (GUI), built in National Instruments Labview, the operator inputs a desired rpm. Using the desired rpm, the GUI commands an output analog voltage (V_{OUT}), where $0 \leq V_{OUT} \leq 5$, using equation 18, from the National Instruments Compact Data Acquisition System (CDAQ). The signal output from the CDAQ is then delivered to the driver, which outputs a pulse width modulated voltage to the motor proportional to the voltage input.

$$V_{OUT} = \left(\frac{5 \cdot (n \cdot 60)}{2000} \right) * 1.4 \quad (18)$$

The driver, manufactured by Minarik, also controlled the ramp up acceleration of the propeller, which was calculated by the GUI. Since the acceleration distance of the carriage was constant, the acceleration of the carriage is calculated by equation 19. Based on the acceleration of the carriage, the propeller acceleration was calculated such that the propeller and the towing carriage would reach their desired speeds coincidentally. Then, knowing the time of acceleration, equation 20, we get propeller acceleration as equation 21.

$$a_c = \frac{U_A^2}{d_c} \quad (19)$$

$$t_a = \frac{U_A}{a_c} \quad (20)$$

$$a_p = \frac{t_a}{n} \quad (21)$$

Where:

a_c = carriage acceleration

d_c = acceleration distance

t_a = time period of acceleration

a_p = propeller acceleration

The forward speed of the carriage was controlled by the test facility, using a velocity profile as seen in figure 28, which uses a linear encoder for closed loop control of forward speed. The encoder, seen in figure 29, is mounted on the overhead beam that holds the carriage rail.

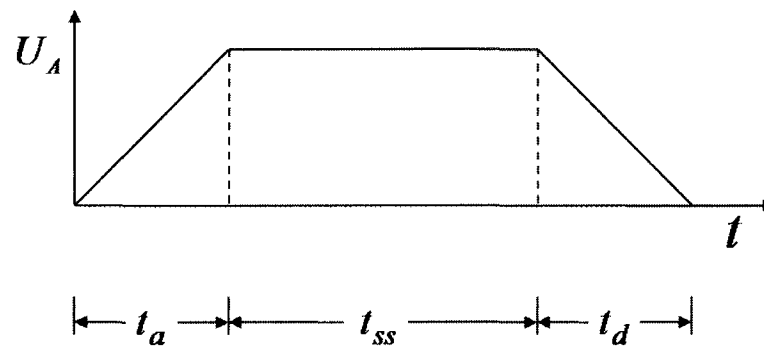


Figure 28. Generic velocity profile for test run

Where:

t = Time

t_{ss} = Time of steady state operation

t_d = Time of deceleration

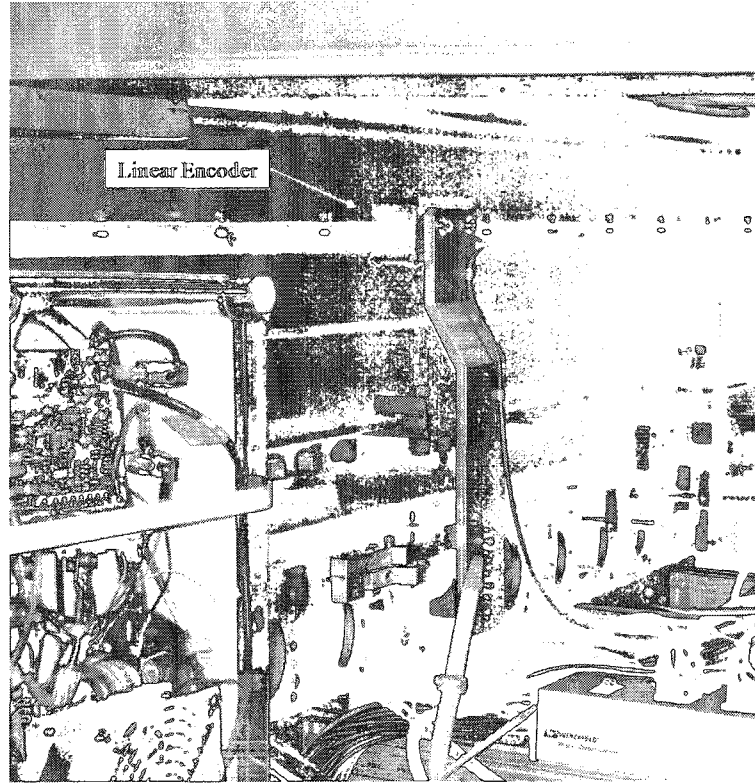


Figure 29. Linear encoder on carriage support beam

2.3.5 Data Collection System

The purpose of the data collection system is to collect and store force transducer, tachometer, shaft position, compass and inclinometer data. The diagram in figure 30, illustrates the data acquisition process.

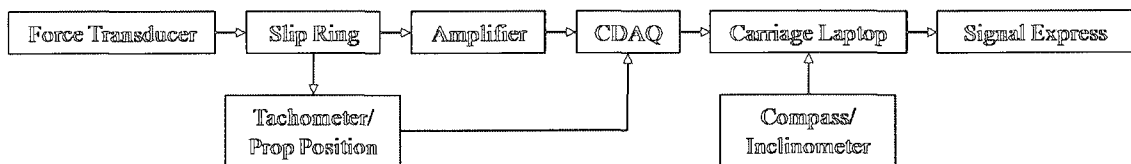


Figure 30. Data collection system diagram

The force transducer used is the AMTI MC 2.5-2K-SP and can be seen in figure 32. It is a stainless steel, cylindrical, potted, waterproof six component transducer with a vertical capacity of 8896 Newtons. It measures forces in the X, Y and Z directions as well as the moments about all 3 axes using a strain gauge bridge system, reference figure 31. Since the propeller boat was to be flooded, a waterproof

connector was used. The output wire from the force transducer passed through the tubular shaft of the right angle gear unit where it is connected to the slip ring, as seen in figure 33.

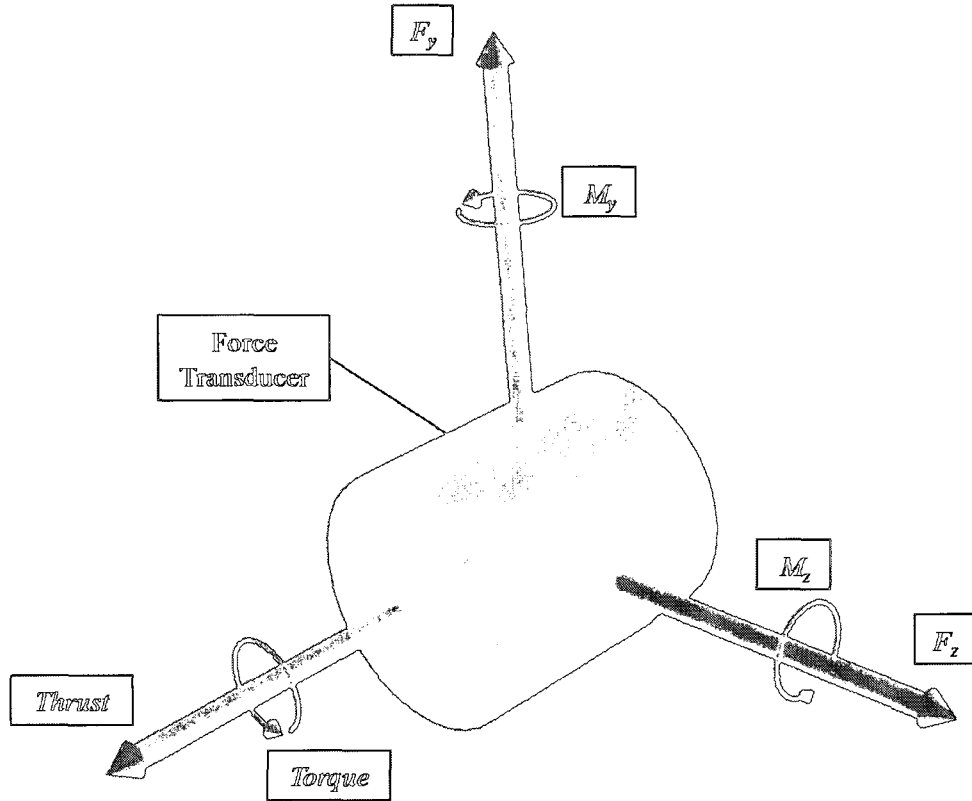


Figure 31. Coordinate axes with respect to the force transducer

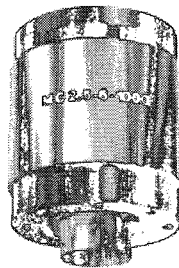


Figure 32. Force transducer

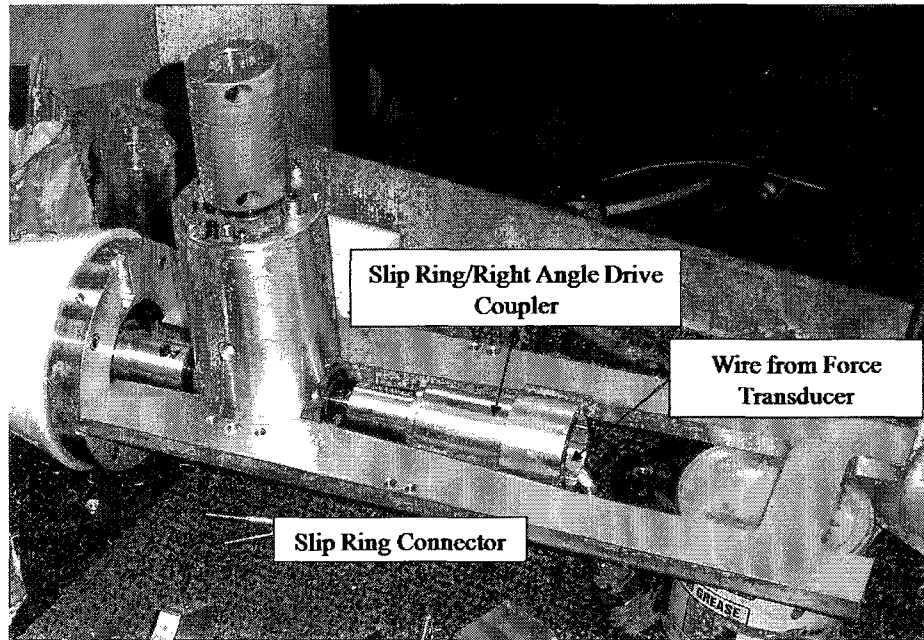


Figure 33. Slip ring side of right angle drive with connector and wire exposed

Since the force transducer output cable will be rotating with the shaft, a slip ring was needed to transfer the output from the rotating cable to a stationary one. The connection on the rotating side of the slip ring is flange coupled with an o-ring inset into the face of the flange. For a watertight connection to be made, a custom coupler, seen in figure 33 and figure 34 was fabricated. The coupler has a precisely machined face, in order to ensure a proper fit with the o-ring. Also, a watertight cord grip was fit into the coupler preventing water intrusion from the shaft side.

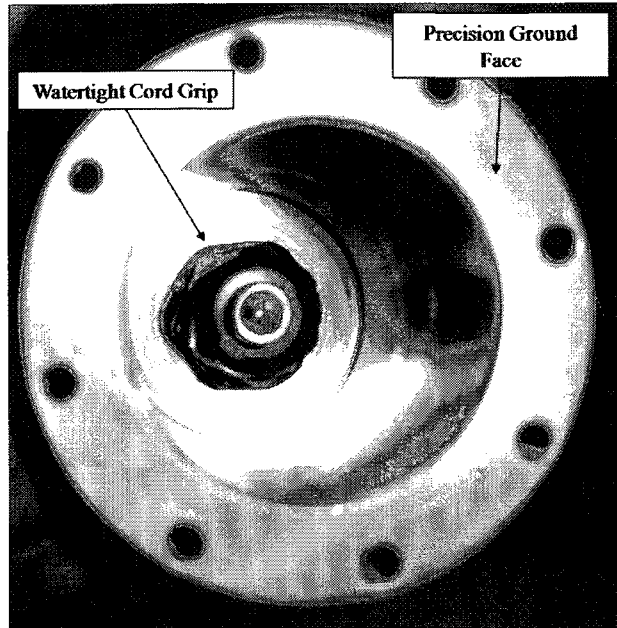


Figure 34. Inside of slip ring coupler

The stationary connection going from the slip ring to the force/torque amplifier has a watertight seal. This was accomplished with a watertight connector, with Tygon tubing around the connector and travelling the length of the output wiring. Figure 35 shows both the slip ring and the sealed connector. The vendor conducted testing on the product to ensure water-tightness of the stator seals, at the depths and rotational speeds expected in the experiment.

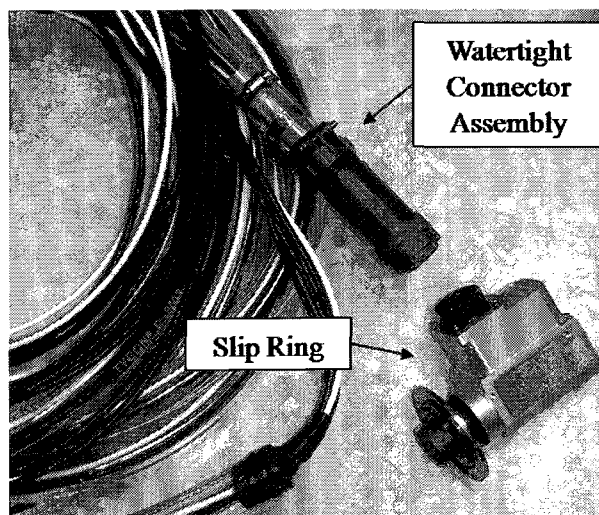


Figure 35. Slip ring and watertight connector assembly

The analog signals, containing the force transducer output, as well as the tachometer and rotational position signal, were connected to the AMTI MSA-6 MiniAmp amplifier. The amplifier has a built in low pass filter with a 1000 Hz cutoff frequency, as well as adjustable gains. Since a sampling frequency of about 180Hz is needed to satisfy the Nyquist theory, the cutoff frequency of the amplifier is sufficient.

The analog output of the amplifier is then converted to a digital signal using the CDAQ. The output of the CDAQ is directly connected via a USB cable to the data acquisition laptop, where the signals are collected and stored using National Instruments Signal Express software, using a 1000 Hz sample rate.

The compass/inclinometer unit is housed in a custom fabricated, watertight box (figure 36). The cord passes through a watertight cord grip, and goes directly into the data acquisition laptop, via a RS232 to USB converter.

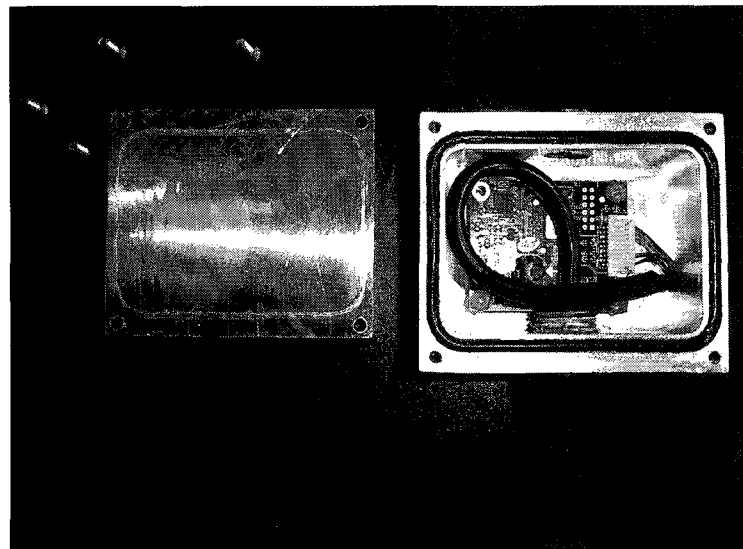


Figure 36. Compass/inclinometer watertight box

2.4 Experiments

Due to time constraints, the immersion ratios explored were limited to values of 0.33 and 0.5. The immersion ratio value of 0.33 was chosen because the extensive set of experiments conducted in Olofsson (1996), was at immersion ratios values of 0.33. This was thought to be a good basis of comparison to validate the data obtained by the new test apparatus. The immersion ratio of 0.5 was also chosen because it

is the value most common in practical operation of SPPs. A table of every test condition can be found in appendix C (table C.1 through table C.10).

The process in which the experiments were conducted started first with $I_T = 0.33$, $\gamma = 0^\circ$ and $\Psi = 0^\circ$. This condition was then tested for a range of J values from approximately $0.8 \leq J \leq 1.8$. The yaw was then changed to $\Psi = 15^\circ$ and was tested over the range of previously stated J values. The yaw was then changed to $\Psi = 30^\circ$ and was tested over the range of previously stated J values. After the range of yaw angles were covered, inclination angle was changed to $\gamma = 15^\circ$, then the range of J values and Ψ were covered. Then inclination angle was changed to $\gamma = 15^\circ$, and the process was repeated. Next, immersion was changed to $I_T = 0.5$, and the inclination and yaw angles were set back to 0° . Then J values of approximately $0.8 \leq J \leq 1.8$ were covered. Additional inclination angles and yaw angles were not covered due to time constraints.

It is important to note that the test apparatus successfully achieved the maximum values of angles of shaft yaw and inclination. The maximum value of immersion ratio was not attempted for several reasons. It was thought by the facility staff that the test apparatus would generate a volume of spray that could not be contained by the spray shield, and would damage sensitive laboratory equipment. Also, the loads on the towing carriage would be very high, increasing the risk of operation into a marginally unsafe range of operation.

The critical value of $F_n \geq 3.0$ was maintained for all test conditions except where $J \cong 1.8$ (figure 40 through figure 49). For this condition the carriage speed was kept at the same value as for $J = 1.6$ (7.62 m/s), and the rotational rate was reduced to increase the J value. This was done because it was thought that increasing carriage speed further could create an unnecessary increase in risk to the laboratory equipment and the safety of the researchers.

3.0 DATA ANALYSIS

The raw data collected in Signal Express was first converted to ASCII files, then imported into Matlab. The separate channels of data collected are listed in table 4. Their physical representation of the forces and moments can be seen in figure 31. After the raw data were compiled into a database, the time histories were plotted for each separate channel for visual inspection.

Table 4. Channels of Data

Rotational Rate (n)
Propeller Position (ϕ)
Thrust (F_x)
F_y
F_z
Torque (M_x)
M_y
M_z
Carriage Speed (U_A)

After a preliminary inspection of the raw data, a power spectral density analysis was conducted. When plotting the results, found in appendix D (figures D. 1 through D. 3), it was found that the largest power spikes were around 60 Hz and 120 Hz for the thrust and torque forces which could be caused by noise due to AC line voltage. However, since these frequencies fall within the range of the fourth harmonic of the propeller frequency, it was thought that filtering out these signals might attenuate the signals and distort the data. In order to filter the thrust and torque data, a low-pass, Chebyshev filter was used. The parameters used in the filter can be found in table 5. The signal for propeller position was not successfully filtered because the noise was at approximately the propeller frequency. An example of the typical signal for angular position can be seen in figure 37.

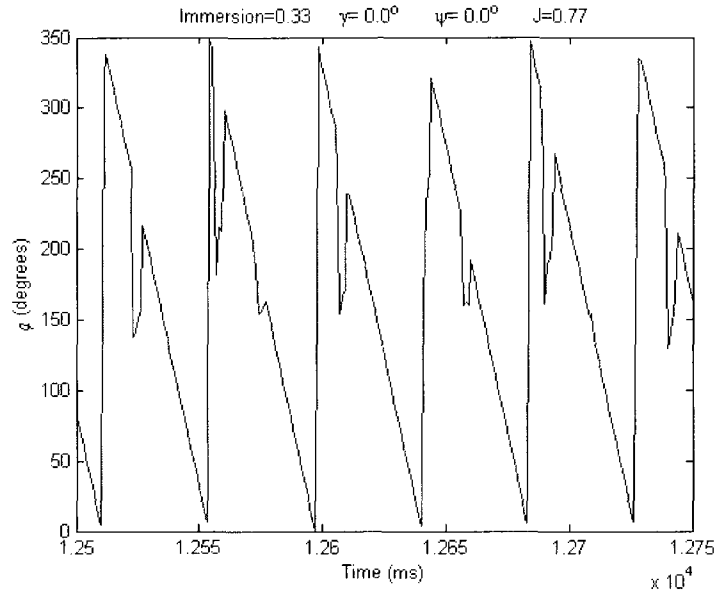


Figure 37. Unfiltered data for blade position versus time

Table 5. Filter Properties

Signal	Filter Type	Filter Name	Harmonics of Prop to Include	Filter Order	Ripple
Rotational Rate (n)	Low Pass	Chebyshev	1	9	0.5
Propeller Position (ϕ)	Low Pass	Chebyshev	N/A	N/A	N/A
F_x	Low Pass	Chebyshev	6	9	0.5
F_y	Low Pass	Chebyshev	6	9	0.5
Thrust	Low Pass	Chebyshev	4	9	0.5
M_x	Low Pass	Chebyshev	6	9	0.5
M_y	Low Pass	Chebyshev	N/A	N/A	N/A
Torque	Low Pass	Chebyshev	4	9	0.5
Carriage Speed (U_A)	Low Pass	Chebyshev	1	2	0.5

After filtering the data, the voltages recorded from the force transducer were converted to metric forces and moments using the cross talk matrix found in appendix G (table G. 1 and table G. 2). The Matlab code that does this calculation can be found in appendix F. The cross talk matrix, which cancels out signal cross talk between channels, was used due to the multiaxis nature of the resulting propeller loads.

The filtered data was then plotted over the time scale. This data was then plotted so the researcher could determine the sample range where the propeller was in steady state operation. Also it was determined that the force data, in many cases, was biased.

To calculate the time average forces for thrust and torque, the mean of the steady state section of each test run was taken. To offset the bias, a data sample was taken where the towing carriage and propeller were stationary (figure 38). The section of data was taken either at the beginning or end of each run, depending on which side had more samples. K_T and K_Q were then calculated from the time average thrust, torque and rpm obtained from the experimental data.

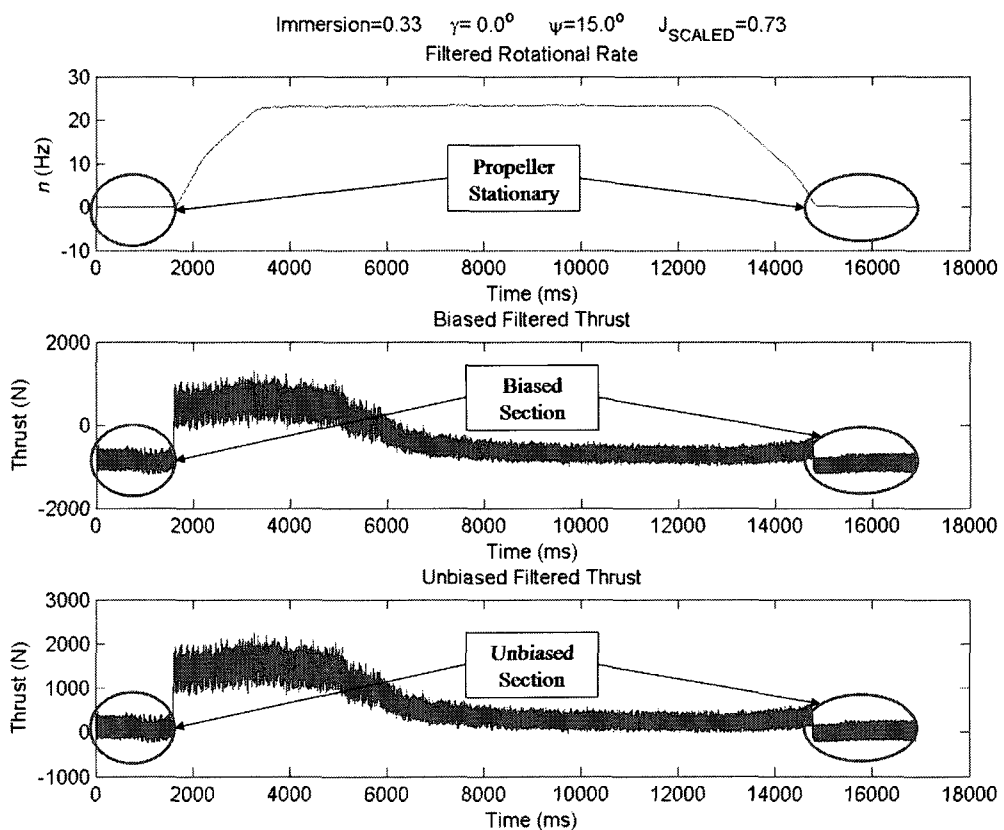


Figure 38. Representative sample of biased data

The efficiency was calculated using a scaled advance ratio. It has been shown that scaling the advance ratio, according to the velocity vector of inflow to the propeller, will cause the performance curves to collapse on one another, for different immersion ratios and angles of shaft inclination (Ferrando 2001). The

following equations show the method used to scale the advance ratio shaft inclination and yaw. First, the scaled speed of advance is calculated using a coordinate transformation. Equation 22 shows the transformation with 0° roll assumed. Using the scaled speed of advance, which has scaled the incoming flow vector based on the angle of shaft inclination and yaw, we determine the scaled advance ratio, which is used to calculate the scaled efficiency.

$$U_{SCALED} = \begin{bmatrix} \cos \gamma \cos \Psi & -\sin \Psi & \sin \gamma \cos \Psi \\ \cos \gamma \sin \Psi & \cos \Psi & \sin \gamma \sin \Psi \\ -\sin \gamma & 0 & \cos \gamma \end{bmatrix} \begin{bmatrix} U_A \\ 0 \\ 0 \end{bmatrix} \quad (22)$$

$$J_{SCALED} = \frac{U_{SCALED}}{nD} \quad (23)$$

$$\eta = \frac{K_T}{K_Q} * \frac{J_{SCALED}}{2\pi} \quad (24)$$

Where:

U_{SCALED} = Scaled Speed of Advance

J_{SCALED} = Scaled Advance Ratio

η = Propeller Efficiency

4.0 DISCUSSION

For the results of these experiments, one can easily see the values of J are not always round numbers (0.8, 1.0, 1.2, etc.). The values obtained for advance ratio are scattered because of the open loop control of the propeller motor, which made it difficult to obtain a precise rotational rate.

It could be seen, upon visual inspection, in each case the data had an initial static phase, which was the period of time before the propeller and carriage started motion. The static section was followed immediately by an almost instantaneous jump in thrust, but a more gradual increase in torque. Thrust immediately increases to its maximum value for the run due to the propeller accelerating faster than the carriage, which causes the SPP to “drag” the carriage. As the carriage and propeller speeds reach steady state, Thrust drops gradually, then levels off, while torque reaches its maximum value as propeller rotational rate reaches its peak value (figure 39).

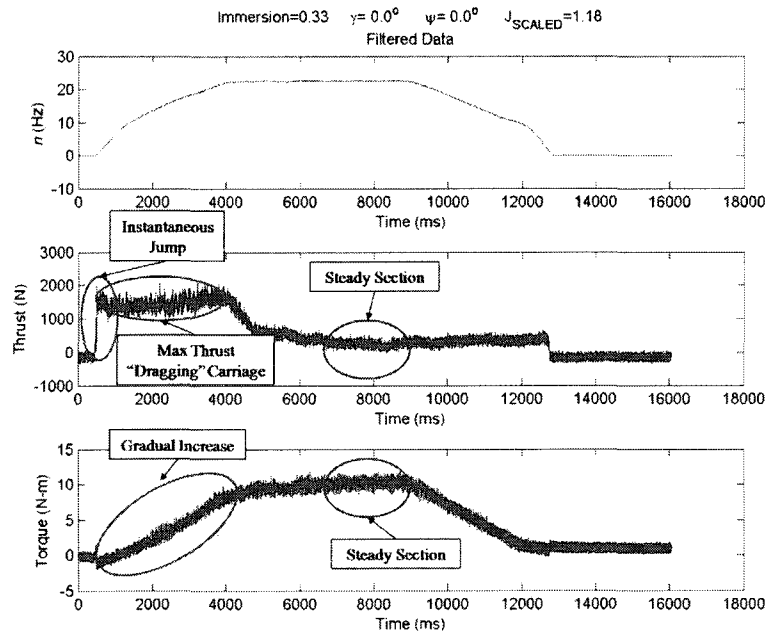


Figure 39. Test run observations

When comparing the results of the experiments to that of the predictions of K_T and K_Q , the experimental values of K_T are very close to the experimental values, while the K_Q values appear to be lower than predicted by a factor of about 10 (figure 40 through figure 49). Due to the low experimental K_Q values, efficiencies over 100% were calculated, which is unrealistic. More investigation of the test apparatus and data analysis will be needed to determine the source of this error.

When comparing the trends of experimental values versus those values predicted using the Ferrando (2001) regressions, it appears that the efficiencies measured at lower values of J_{SCALED} match more closely to the predicted values than the efficiencies measured at higher values of J_{SCALED} . This could be because the regressions used in the prediction were developed using propellers with lower $\left(\frac{P}{D}\right)$ values, which reach peak efficiencies at lower J values. Therefore, higher values of J were not thoroughly investigated in the development of the Ferrando (2001) regressions.

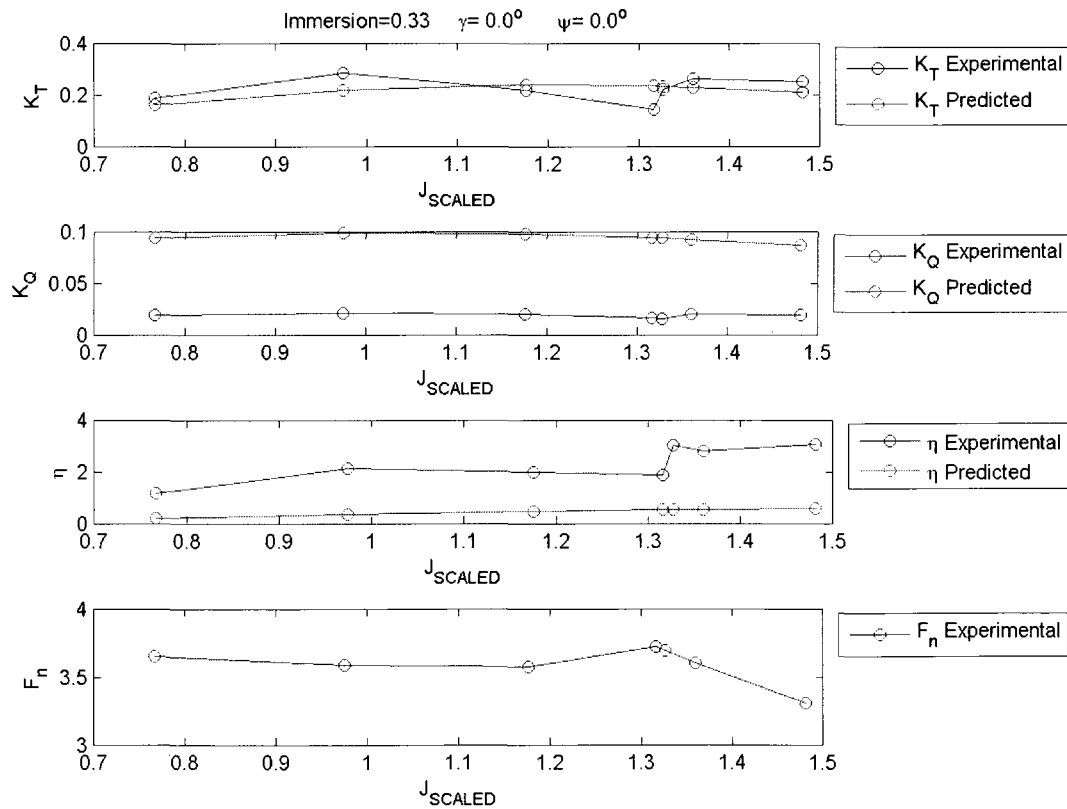


Figure 40. K_T , K_Q and η for predicted values versus experimental values

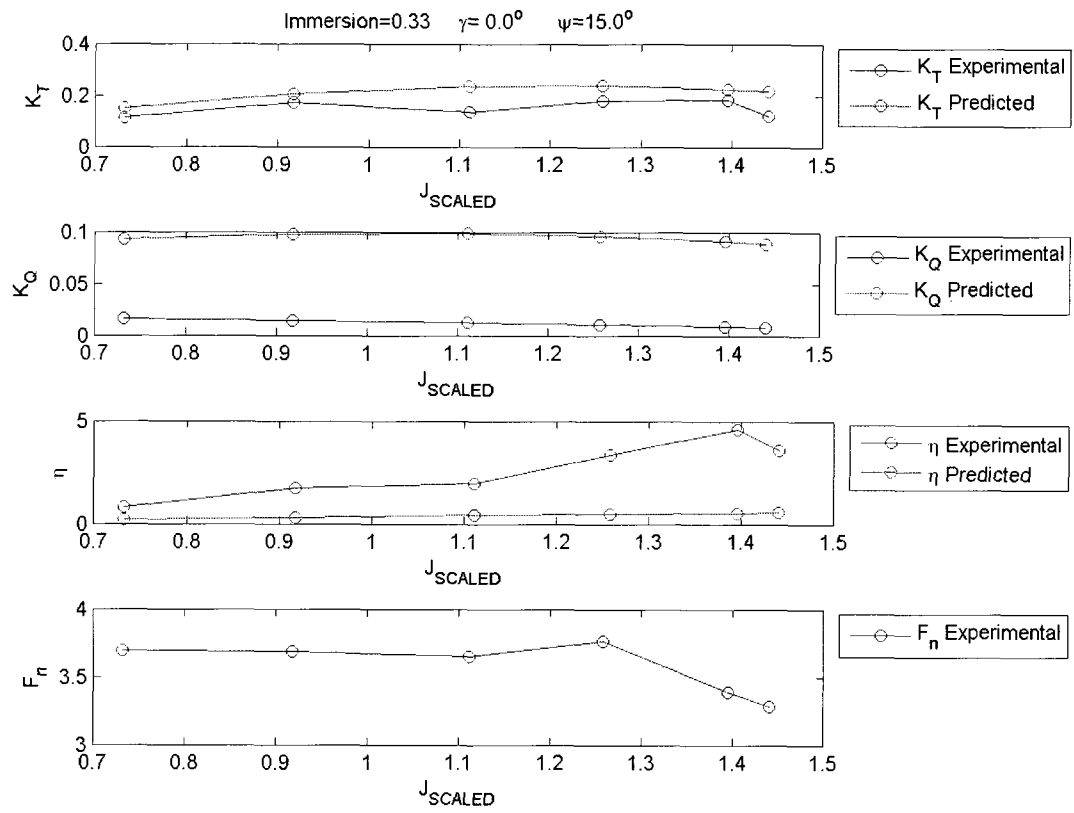


Figure 41. K_T , K_Q and η for predicted values versus experimental values

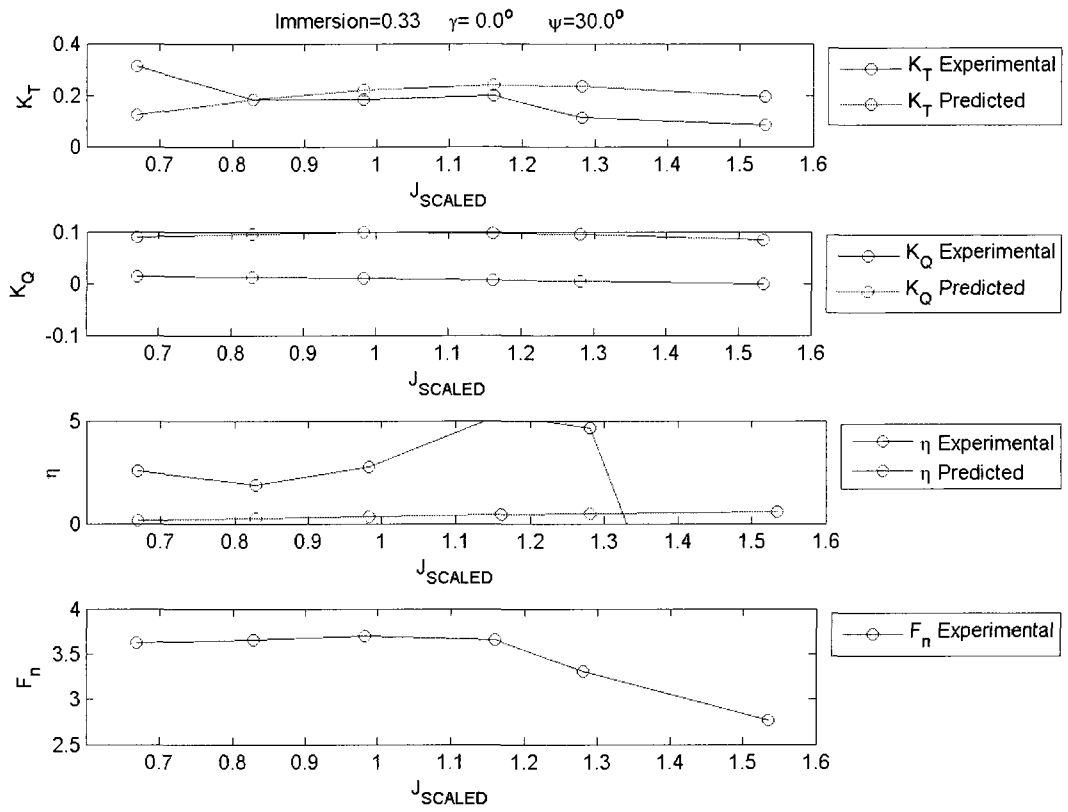


Figure 42. K_T , K_Q and η for predicted values versus experimental values

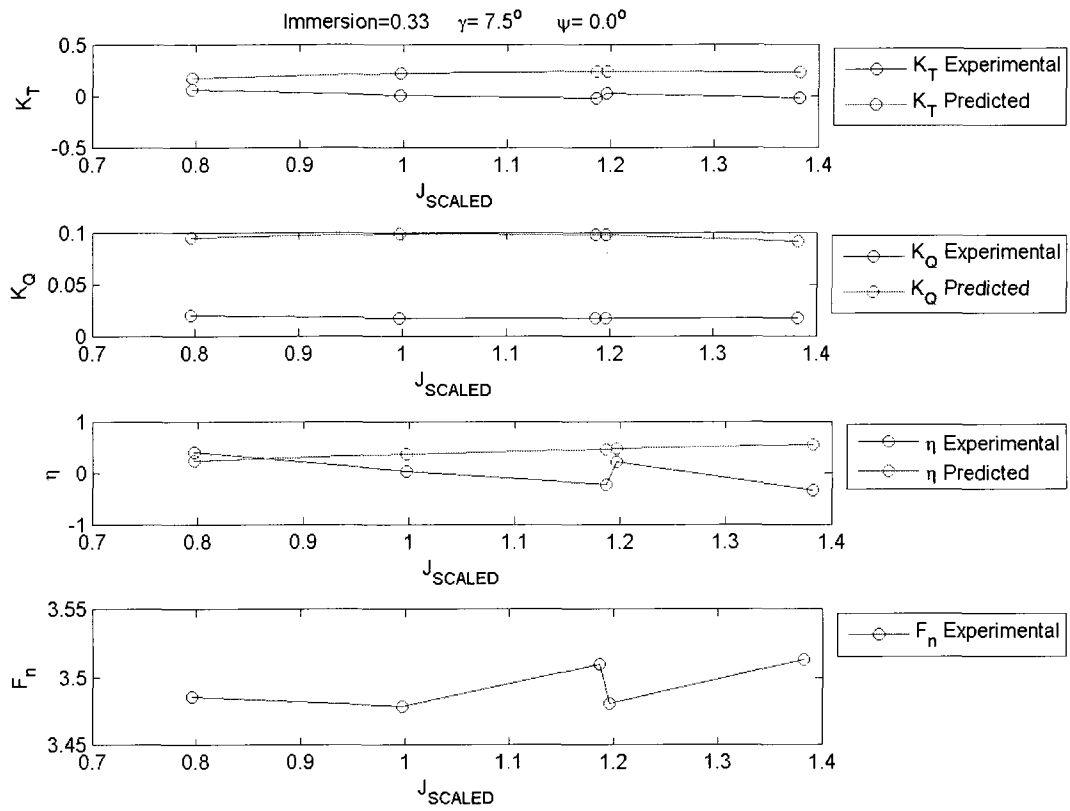


Figure 43. K_T , K_Q and η for predicted values versus experimental values

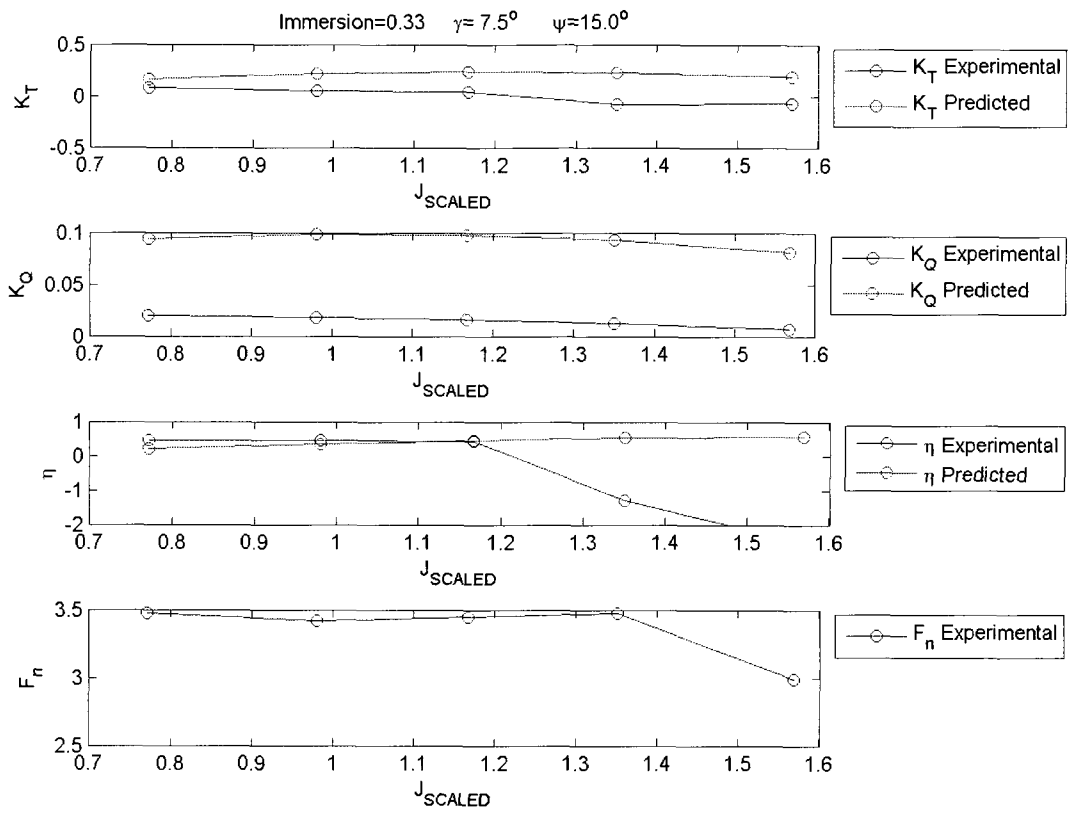


Figure 44. K_T , K_Q and η for predicted values versus experimental values

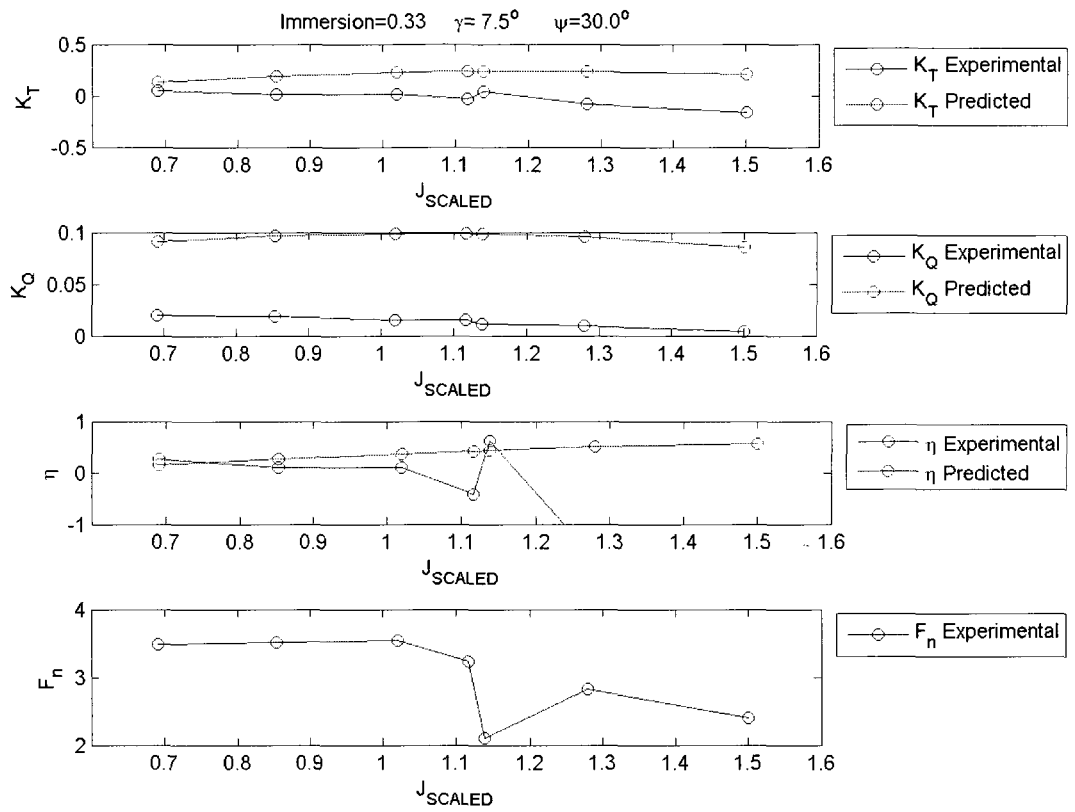


Figure 45. K_T , K_Q and η for predicted values versus experimental values

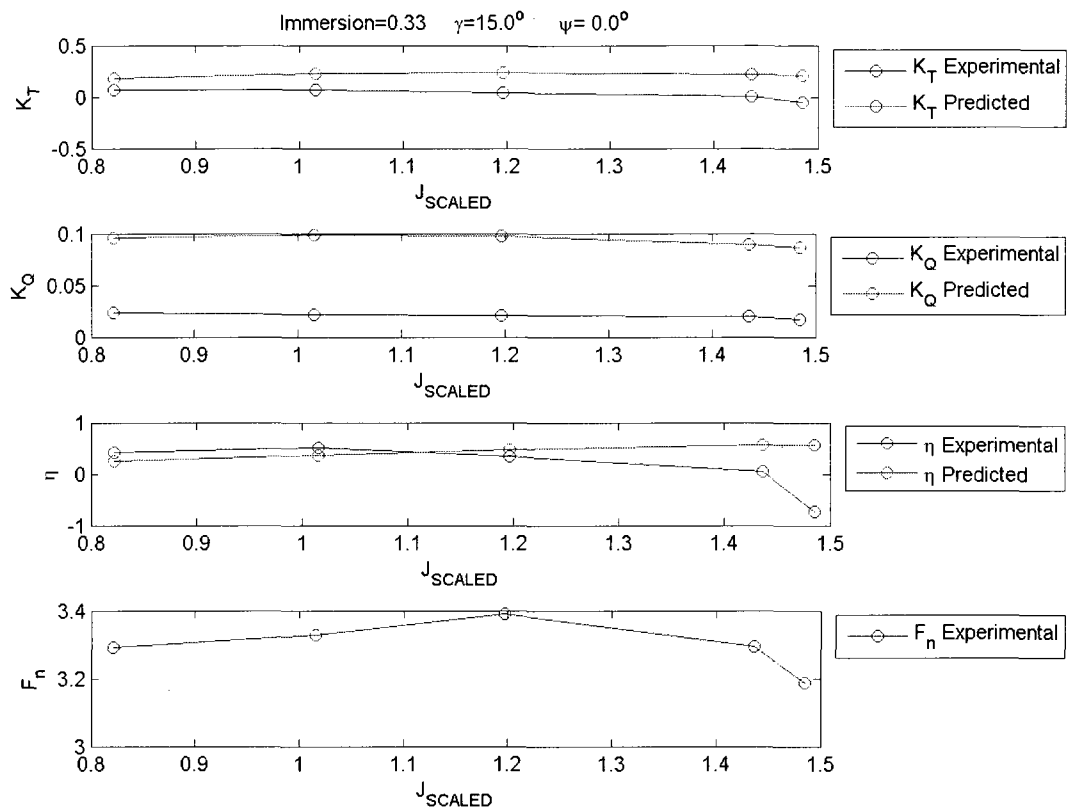


Figure 46. K_T , K_Q and η for predicted values versus experimental values

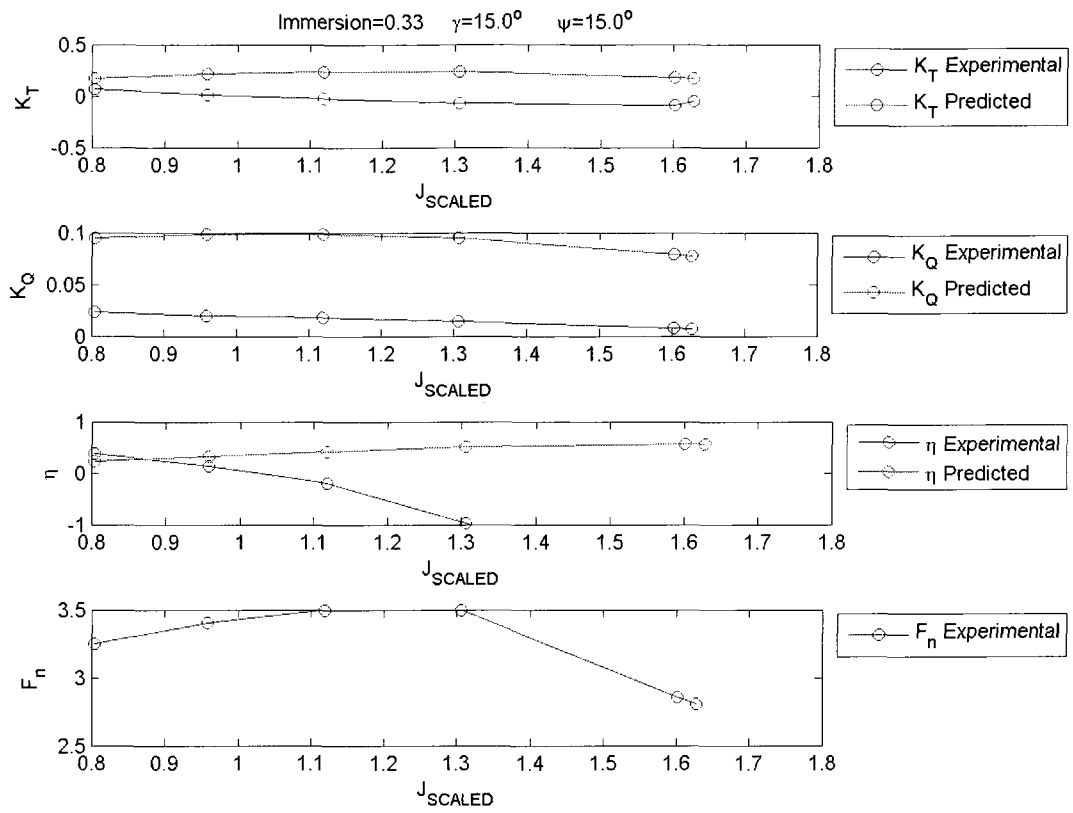


Figure 47. K_T , K_Q and η for predicted values versus experimental values

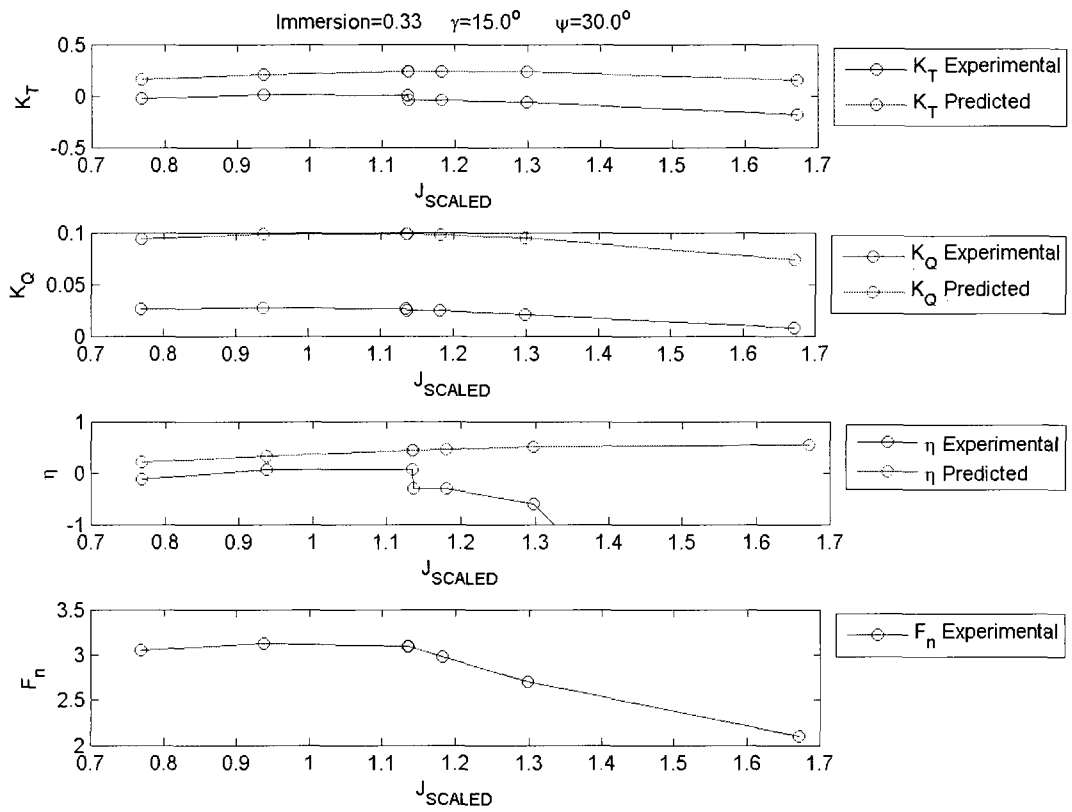


Figure 48. K_T , K_Q and η for predicted values versus experimental values

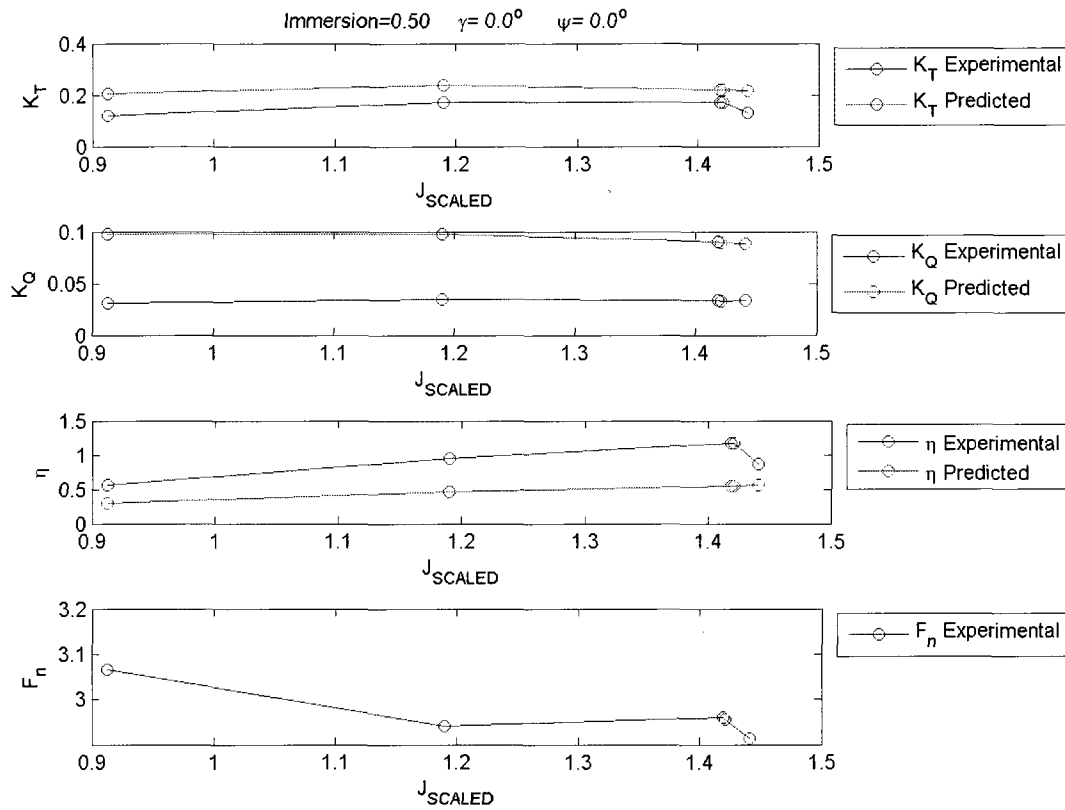


Figure 49. K_T , K_Q and η for predicted values versus experimental values

For figure 50 through figure 54, K_T and K_Q values for a range of J_{SCALED} were compared by keeping 2 parameters (i.e. I_T and γ , or I_T and Ψ) constant while the other was varied. The effects of varying immersion are not apparent in K_T (figure 50). However for K_Q , figure 50 shows that an increase in I_T corresponds to an increase in K_Q .

When varying angle of inclination while keeping yaw angle constant and immersion constant, K_T is higher at $\gamma = 0^\circ$, while K_T stays about the same for $\gamma = 7.5^\circ$ and $\gamma = 15^\circ$ (figure 51). Also, the gap between the K_T curve for $\gamma = 0^\circ$ and the K_T curves for the other inclination angles grows with increasing yaw angle (figure 51). The K_Q value increases as inclination increases when $\Psi = 15^\circ$ and $\Psi = 30^\circ$, while staying about the same when $\Psi = 0^\circ$ (figure 52). Also, the gap between the K_Q curves increase, as the yaw angle increases (figure 52).

Holding immersion and angle of inclination constant, while varying the yaw angle, does not reveal significant patterns in K_T (figure 53), the curves generally follow close to one another. The K_Q values (figure 54) tend to decrease with increasing yaw angle when $\gamma = 0^\circ$. Since the propeller shaft is yawed in a direction such that the free stream introduces an incoming flow vector that assists the rotation of the propeller, it is logical that K_Q would decrease as yaw angle increases, increasing the flow vector assisting the rotation of the propeller. For an inclination angle of $\gamma = 7.5^\circ$, it still appears that K_Q decreases with increasing yaw angle, however it is not as apparent as with inclination angles of $\gamma = 0$ (figure 54). At the maximum inclination value of $\gamma = 15^\circ$, there are no obvious trends (figure 54).

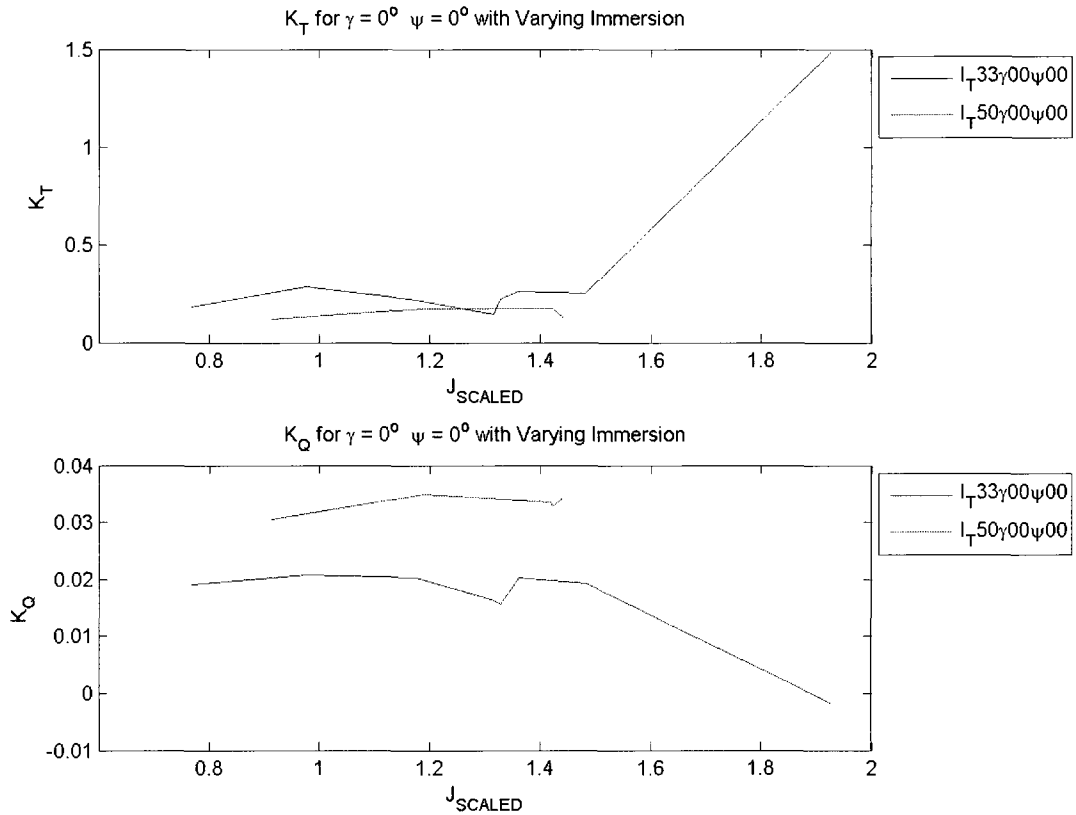


Figure 50. K_T and K_Q holding angles of inclination and yaw constant while varying depth of immersion

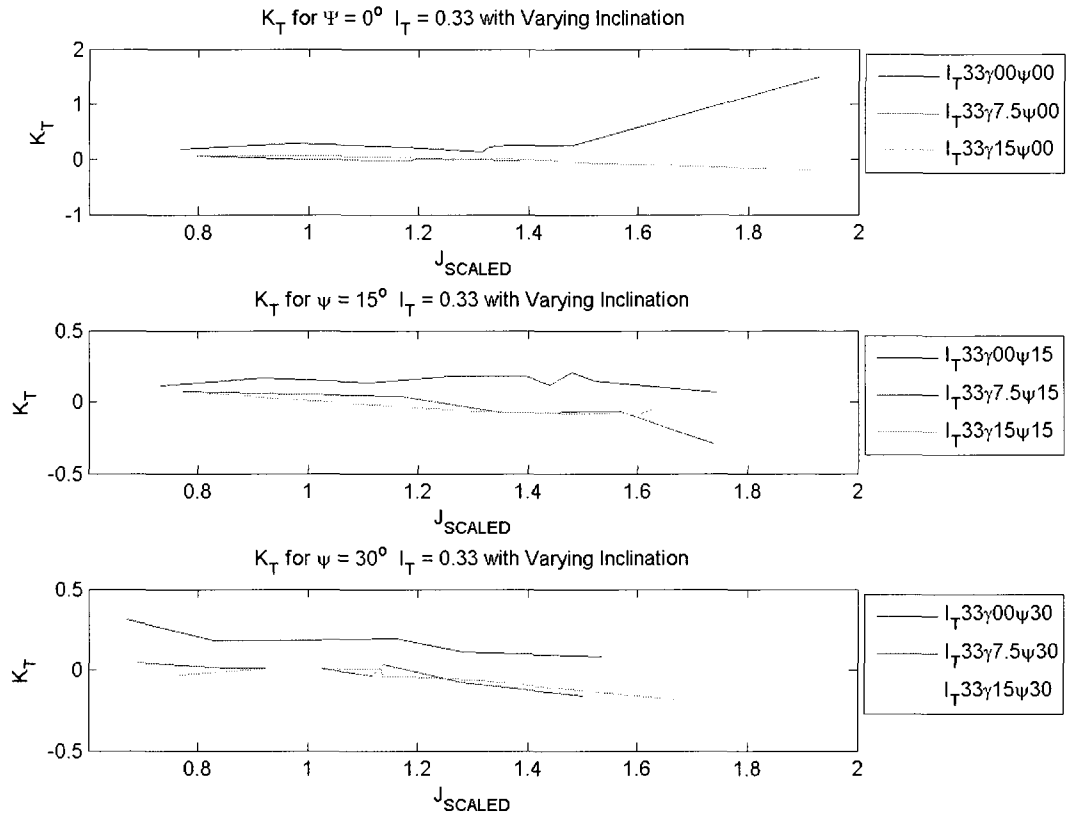


Figure 51. K_T holding angle of yaw and depth of immersion constant while varying inclination angle

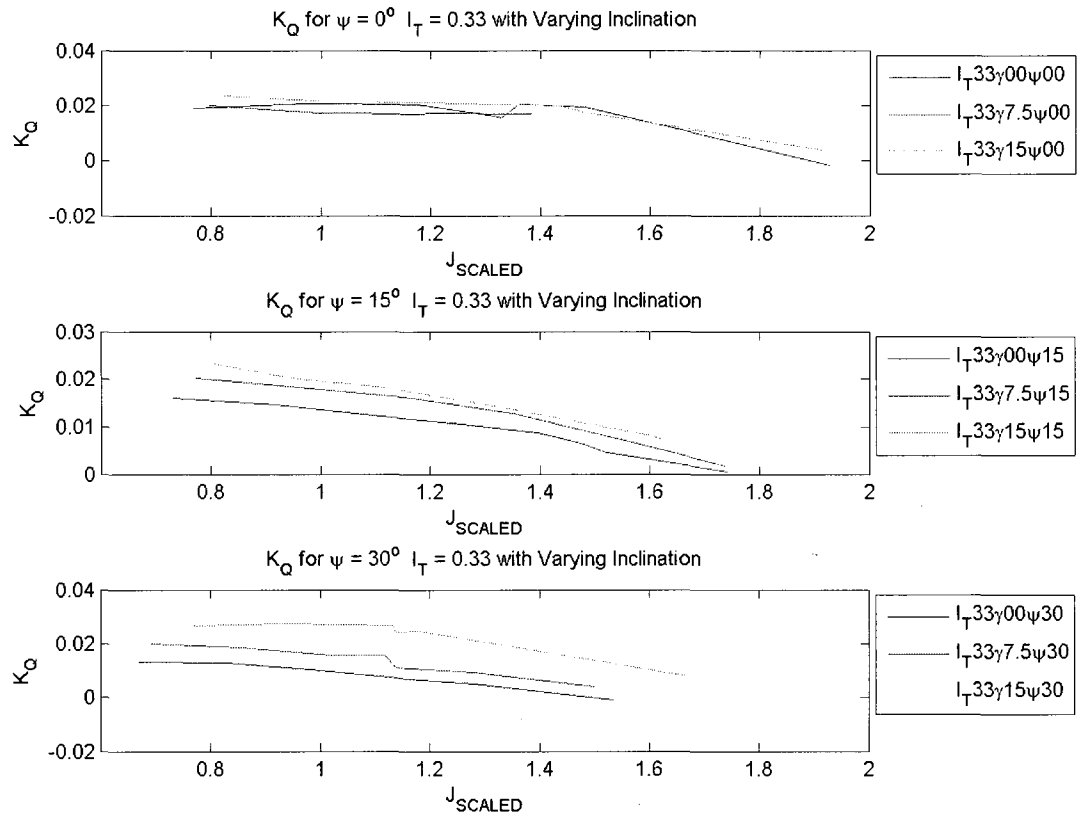


Figure 52. K_Q holding angle of yaw and depth of immersion constant while varying inclination angle

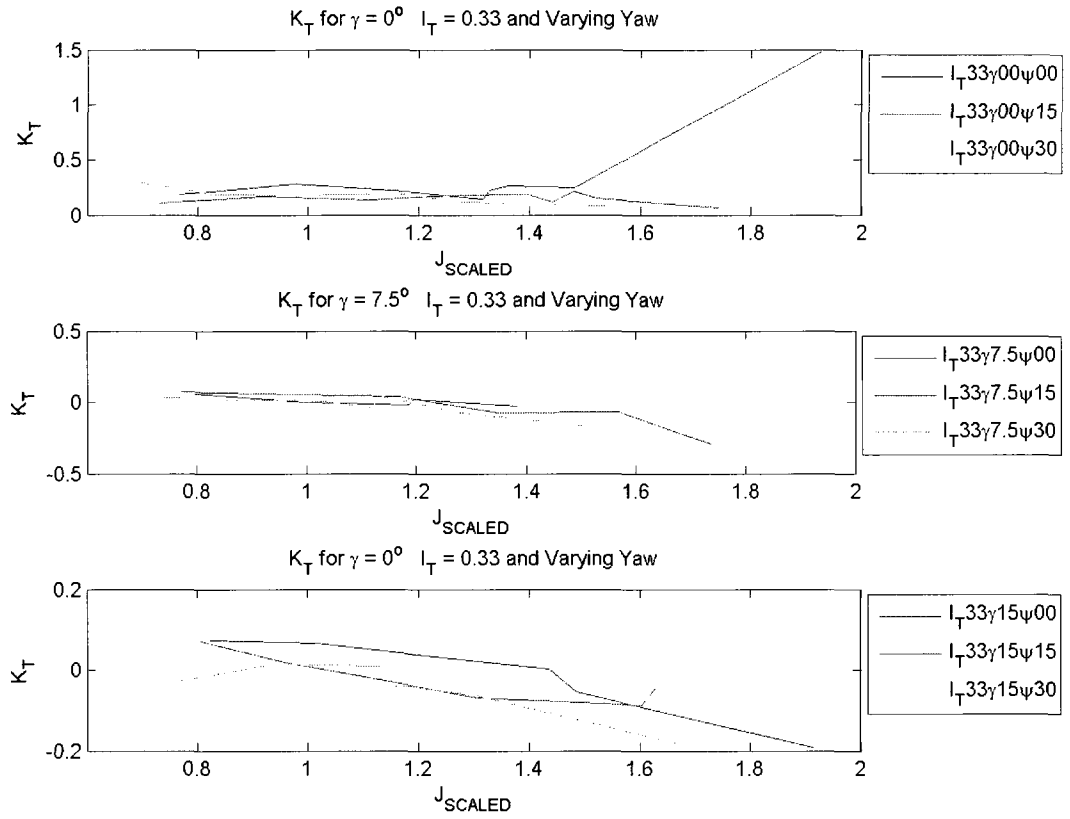


Figure 53. K_T holding angle of inclination and depth of immersion constant while varying yaw angle

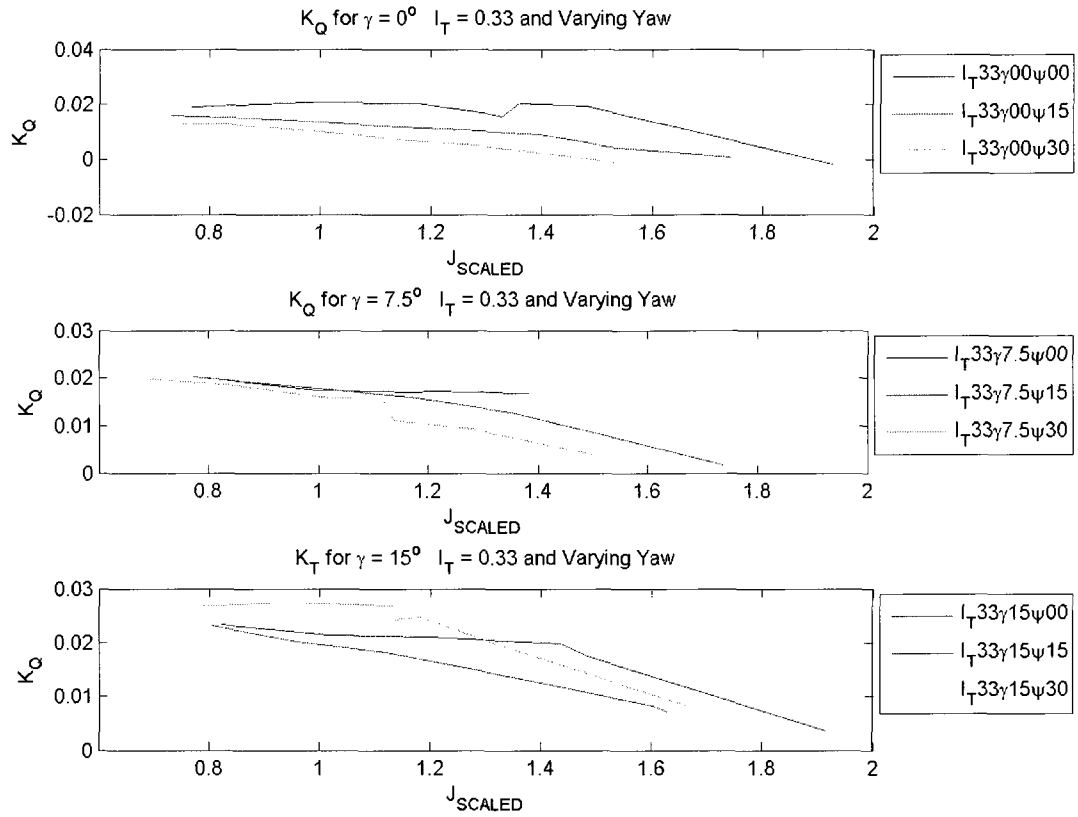


Figure 54. K_T holding angle of inclination and depth of immersion constant while varying yaw angle

4.1 Uncertainty Analysis

A complete uncertainty analysis has not been completed, due to time constraints, however some preliminary estimates have been done. Using equation 25, with the estimated uncertainty values found in table 6, the uncertainty in the data was found to be between 5% and 10%.

$$dK_T = \frac{\partial K_T}{\partial T} dT + \frac{\partial K_T}{\partial \rho} d\rho + \frac{\partial K_T}{\partial n} dn + \frac{\partial K_T}{\partial D} dD. \quad (25)$$

Table 6. Independent uncertainty values

$dT =$	0.019 N
$d\rho =$	0.00006 kg/m ³
$dn =$	0.1 Hz
$dD =$	0.0032 m

5.0 CONCLUSION

From a functionality standpoint, the test apparatus accomplished the mission set for by the Office of Naval Research. The model propeller was tested with varying depths of immersion, as well as varying angles shaft inclination and yaw. This is the first time, to the researcher's knowledge, that this has been done. This prototype has provided proof of concept in the ability to test surface piercing propeller with angles of yaw up to 30° as well as angles of shaft inclination up to 15° . The apparatus also successfully completed tests and different depths of immersion.

When comparing the results of the experiments to that of the predictions of K_T and K_Q , the experimental values of K_T are very close to the experimental values, while the K_Q values appear to be lower than predicted by a factor of about 10 (figure 40 through figure 49). Due to the low experimental K_Q values, efficiencies over 100% were calculated, which is unrealistic. More investigation of the test apparatus and data analysis will be needed to determine the source of this error.

The following conclusions were drawn from the comparison of K_T and K_Q values for a range of J_{SCALED} while keeping 2 parameters (i.e. I_T and γ , or I_T and Ψ) constant while the other parameter was varied.

- 1) Increasing I_T causes an increase in K_Q while K_T stayed approximately consistent
- 2) For constant yaw angle and immersion:
 - a) K_T is higher for $\gamma = 0^\circ$ than $\gamma = 7.5^\circ$ and $\gamma = 15^\circ$ which have approximately the same K_T curve
 - b) K_Q increases as inclination angle increases for $\gamma = 7.5^\circ$ and $\gamma = 15^\circ$, however this characteristic is minimized for $\gamma = 7.5^\circ$
 - c) The gap between K_Q curves increases as yaw angle increases
- 3) For constant inclination and immersion,
 - a) Yaw angle variation has no significant impact on K_T

- b) K_Q decreases with increasing yaw angle for $\gamma = 0^\circ$. This effect minimizes with increasing inclination angle

5.1 Future Work

The next phase of the project would be to fine tune the instrumentation system. The test apparatus as it exists is capable of operating on a closed loop, but it requires more work to get all the components working together. To aid in closing the loop for propeller speed control, a permanent magnet DC motor would make design of the control system more straightforward, where the rotational rate output is linearly proportional to the voltage input.

The problems with the data collection system need to be investigated. The saturation of the M_y channel needs to be resolved. Also, gains of the torque and thrust channels need to be tuned for optimum resolution in the proper data range.

Finally, more tests need to be performed on the propeller, investigating more shaft angles, inclination angles, depths of immersion and a wider range of advance ratios. Better resolution would be gained in the propeller performance curves, making it possible to more accurately identify points of maximum K_T , K_Q and η , allowing the researcher to more fully characterize the performance of the propeller.

REFERENCES

- 1) Altamirano, Luis M. "Flow Visualization of the Ventilated Cavities Generated by a Surface Piercing Propeller." *Master Thesis*. Florida Atlantic University, 2010.
- 2) Carlton, John. *Marine Propellers and Propulsion*. Great Britain: MPG Books Ltd, 2007.
- 3) Clark, Dennis J, William M. Ellsworth, and John R. Meyer. "The Quest for Speed at Sea." *Technical Digest*. Carderock Division, NSWC, April 2004.
- 4) Faltinsen, Odd M. *Hydrodynamics of High-Speed Marine Vehicles*. New York: Cambridge University Press, 2005.
- 5) Ferrando, Marco. *Performance of a Family of Surface Piercing Propellers*. London: Royal Institute of Naval Architects, 2001.
- 6) Ghiasi, Mahmoud. "Hydrodynamic Characteristics of the Surface Piercing Propeller (SPP) by Using Special Practical and Numerical Approach." *Journal of Marine Science and Application*, 2009: 267-274.
- 7) International Towing Tank Conference. "ITTC - Recommended Procedures; Testing and Extrapolation Methods Propulsion, Propulsor Open Water Test." *23rd International Towing Tank Conference*. 2002. 7.5-02-03-02.1.
- 8) Kruppa, C. "Testing of Partially Submerged Propellers." *13th International Towing Tank Conference*. 1972.
- 9) Lorio, J. M., L. M. Altamirano, M. H. Tall & K. D. vonEllenrieder. "Design of a Next Generation Surface-Piercing Propeller Test Stand." *2009 MTS/IEEE OCEANS Conference*. Biloxi, MS, 2009.
- 10) Olds Engineering. *Olds Marine Engineering, Marine Division, Boat Propeller and Propulsion Terminology*. <http://www.olds.com.au/marine/terminology.html> (accessed April 1, 2010).
- 11) Olofsson, Niclas. *Force and Flow Characteristics of a Partially Submerged Propeller*. Goteborg: Department of Naval Architecture and Ocean Engineering, Division of Hydromechanics, Chalmers University of Technology, 1996.
- 12) Shiba, H. *Air Drawing of Marine Propellers*. Japan: Transport Technical Research Institute, 1953.
- 13) van Manen, J. D., and P. van Oossanen. *Principles of Naval Architecture*. 2nd Edition. Edited by Edward V. Lewis. Vol. II. III vols. Jersey City: The Society of Naval Architects and Marine Engineers, 1988.
- 14) Young, Yin L., and Spyros A. Kinnas. "Performance Prediction of Surface-Piercing Propellers." *Journal of Ship Research*, 2004: 288-304.

APPENDIX A: PROPELLER MEASUREMENT

MRI
Report 1 - ISO Class II 8/4/2008 11:43:35 AM - Initial

Customer	FLA ATLANTIC UNIVERSITY	Job ID		FileName	ScanData758.txt
Vessel		Class	I	ScanDate	8/4/2008 11:43:35 AM
Job Number		Repair Status	Initial		
Brand	DEWALD	Style	4 Blade	Rotation	L
Part #		Material	Stainless	Marked Dia	10
S/N		Bore	SPLINED	Measured Dia	9.70
Stamped		DAR		Mk'd Pitch	18.25
Inspector	RMS	Cupping	#2 LIGHT	Pitch of Wheel	18.312

Radii Averages:

r/R	Bld 1	Bld 2	Bld 3	Bld 4
50.0 %	16.68	16.50	16.58	16.79
70.0 %	17.58	17.80	17.63	17.85
90.0 %	20.32	20.36	20.46	21.20

BladeAverages

Avg.	18.193	18.220	18.223	18.613
------	--------	--------	--------	--------

Blade 1	18.19
Blade 2	18.22
Blade 3	18.22
Blade 4	18.61

APPENDIX B: RESONANT FREQUENCY CALCULATIONS

Appendix B.1: Strut Natural Frequency Calculation

The following calculation was used to estimate the natural frequency of the test apparatus strut. Since the propeller frequency was to be around 20 Hz to 25 Hz, the harmonics of blade water impact would make the forcing frequency around 80 Hz to 100 Hz. Therefore a strut natural frequency of at least 300 Hz was required to be conservative.

```
clear all
clear
clc

%Conversion Constants
in2m = 2.54/100;
kg2lbs = 2.205;
lbs2kg = 1/kg2lbs;
lbs2N = 4.448;
N2lbs = 1/lbs2N;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%Physical Constants
%Using 304 Stainless
%Using 6061 Aluminum
E_SS = 200*10^9;    % Young's Modulus SS [Pa]
E_Al = 70*10^9;    % Young's Modulus Al [Pa]

rho_h2o = 998;    % kg/m^3

rho_CRS = 7.87; %g/cc
rho_CRS = rho_CRS*(1/1000)*(100^3/1); %kg/m^3

rho_SS = 8.03; %g/cc
rho_SS = rho_SS*(1/1000)*(100^3/1); %kg/m^3

rho_Al = 2.7; %g/cc
rho_Al = rho_Al*(1/1000)*(100^3/1); %kg/m^3

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
```

%Mass of lower body

%prop

prop_wgt = 4; %lbs
prop_mass = prop_wgt*lbs2kg; %kg
prop_wgt = prop_wgt*lbs2N; %N

%prop coupling assembly

spline_vol = 5.8; %cubic inch
spline_vol = spline_vol*(in2m)^3; %m^3

spline_mass_SS = spline_vol*rho_SS; %kg
spline_mass_Al = spline_vol*rho_Al; %kg

%shafting

%transducer to prop shaft

shft_lngth = 17.25*in2m; %m
shft_dia = 1.375*in2m; %m
shft_area = pi()*shft_dia^2/4; %m^2

shft_mass_CRS = rho_CRS*shft_lngth*shft_area; %kg
shft_mass_SS = rho_SS*shft_lngth*shft_area; %kg
shft_mass_Al = rho_Al*shft_lngth*shft_area; %kg

%gearleg

gearleg_wgt = 4.5; %lbs
gearleg_mass = gearleg_wgt*lbs2kg; %kg

%force transducer

ft_wgt = 2.5; %lbs
ft_mass = ft_wgt*lbs2kg; %kg

%slip ring

sr_wgt = 1; %lbs
sr_mass = sr_wgt*lbs2kg; %kg

%couplers

cplr_wgt = 5; %lbs
cplr_mass = cplr_wgt *lbs2kg; %kg

%fairing

Sw_fairing = 352*in2m^2; % Wetted surface area of fairing [m^2]
t_fairing = 4e-3; % Thickness of fairing in meters
rho_fg = 1.5e3; % Density of typical marine composites [kg/m^3] (30% glass fiber/70% resin)
fairing_mass = Sw_fairing*t_fairing*rho_fg; %kg

%Mass of strut

%4" sch 160 pipe

strut_OD = 4.5; %in
strut_thk = .531; %in

strut_area = pi()*((strut_OD/2)^2-(strut_OD/2-(strut_thk))^2); %in^2
strut_area = strut_area*in2m^2; %m^2

strut_L = 40; %in

```

strut_L = strut_L*in2m; %m

strut_vol = strut_area*strut_L; %m^3

strut_mass_Al = rho_Al*strut_vol; %kg
strut_mass_SS = rho_SS*strut_vol; %kg

%Natural Frequency
%added mass term Newman p.144 "the added mass of the cylinder is precisely
%equal to the displaced mass of fluid
%%%%NEED SUBMERGED LENGTH OF STRUT
strut_L_subm = 25.5; %in
strut_L_subm = strut_L_subm*in2m; %m

madd = strut_L_subm*(pi)*(strut_OD*in2m)^2/4)*rho_h2o;

%Moment of Inertia
strut_I = pi()*((strut_OD*in2m)^4-((strut_OD-(2*strut_thk))*in2m)^4)/64;

%stiffness
k_SS = 3*E_SS*strut_I/(strut_L^3);
k_Al = 3*E_Al*strut_I/(strut_L^3);

%equivolent mass
meq_SS = prop_mass+spline_mass_SS+
shft_mass_SS+gearleg_mass+ft_mass+sr_wgt+cplr_mass+fairing_mass+(.23*strut_mass_SS)+madd; %
meq_Al = prop_mass+spline_mass_SS+
shft_mass_SS+gearleg_mass+ft_mass+sr_wgt+cplr_mass+fairing_mass+(.23*strut_mass_Al)+madd; %

%nat freq
Properties.freq_SS = sqrt(k_SS/meq_SS);
Properties.freq_Al = sqrt(k_Al/meq_Al)

Properties =

freq_SS: 333.5716
freq_Al: 218.9105

```


Appendix B.2: Propeller Shaft Natural Frequency Calculation

The following calculation was used to estimate the natural frequency of the test apparatus strut. Since the propeller frequency was to be around 20 Hz to 25 Hz, the harmonics of blade water impact would make the forcing frequency around 80 Hz to 100 Hz. Therefore a propeller shaft natural frequency of at least 300 Hz was required to be conservative.

clear all

%Conversion Constants

in2m = 2.54/100;

%Physical Constants

E_SS = 200*10^9; % Young's Modulus SS [Pa]

%E_SS = E_SS*0.000145037738 %Young's Modulus SS [psi]

E_Al = 70*10^9;

rho_h2o = 998; %kg/m^3

%Prop Shaft properties

rho_SS = 8.03; %g/cc

rho_SS = rho_SS*(1/1000)*(100^3/1); %kg/m^3

rho_Al = 2.7; %g/cc

rho_Al = rho_Al*(1/1000)*(100^3/1); %kg/m^3

shft_lngth = 17*in2m; %m

shft_dia = 1.375*in2m; %m

shft_area = pi()*shft_dia^2/4; %m^2

shft_wgt_SS = rho_SS*shft_lngth*shft_area; %kg

shft_wgt_Al = rho_Al*shft_lngth*shft_area; %kg

%Inertia of Shaft

I_shft = pi()/4*(shft_dia/2)^4; %m^4

%Propeller

prop_wgt = 4; %lbs

prop_wgt = prop_wgt*.45359; %kg

spline_vol = 5.8; %cubic inch

spline_vol = spline_vol*(in2m)^3; %m^3

```

spline_wgt_SS = spline_vol*rho_SS; %kg
spline_wgt_Al = spline_vol*rho_Al; %kg

%added mass term Newman p.144 "the added mass of the cylinder is precisely
%equal to the displaced mass of fluid
madd = shft_lngth*shft_area*rho_h2o;

%stiffness
k_SS = 3*E_SS*I_shft/(shft_lngth^3);
k_Al = 3*E_Al*I_shft/(shft_lngth^3);

%equivalent mass
meq_SS = prop_wgt+spline_wgt_SS+(.23*shft_wgt_SS)+madd; %
meq_Al = prop_wgt+spline_wgt_Al+(.23*shft_wgt_Al)+madd; %

%nat freq
freq_SS = sqrt(k_SS/meq_SS)
freq_Al = sqrt(k_Al/meq_Al)

%rcm = (shft_lngth/2*shft_wgt + prop_wgt*shft_lngth)/(shft_wgt+prop_wgt)
%freq = sqrt(3*E_SS*I_shft/(rcm*(shft_wgt+prop_wgt+madd)*2*pi))

freq_SS: 380.75
freq_Al: 263.64

```

APPENDIX C: TEST CONDITIONS

These tables show the parameters of each test condition as well as the time average results for each condition. The rows labeled with “_Cross” as a suffix are the data that was generated using the matrix for cancelling channel cross talk when converting voltages from the force transducer to forces.

Table C. 1

It	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33
It_Cross	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33
Pitch (deg)	0	0	0	0	0	0	0	0
Pitch_Cross (deg)	0	0	0	0	0	0	0	0
Yaw (deg)	0	0	0	0	0	0	0	0
Yaw_Cross (deg)	0	0	0	0	0	0	0	0
Ucom (m/s)	14.28	17.85	21.42	24.99	24.99	24.99	24.99	24.99
Ucom_Cross (m/s)	14.28	17.85	21.42	24.99	24.99	24.99	24.99	24.99
Uout (m/s)	14.28	17.85	21.42	24.99	24.99	24.99	24.99	24.99
Uout_Cross (m/s)	14.28	17.85	21.42	24.99	24.99	24.99	24.99	24.99
Mean_n (Hz)	23.03	22.65	22.54	22.73	23.29	23.49	20.88	16.05
Mean_n_Cross (Hz)	23.03	22.65	22.54	22.73	23.29	23.49	20.88	16.05
J	0.77	0.97	1.18	1.36	1.33	1.32	1.48	1.93
J_Cross	0.77	0.97	1.18	1.36	1.33	1.32	1.48	1.93
Jscaled	0.77	0.97	1.18	1.36	1.33	1.32	1.48	1.93
Jscaled_Cross	0.77	0.97	1.18	1.36	1.33	1.32	1.48	1.93
Fn	3.65	3.59	3.57	3.60	3.69	3.72	3.31	2.54
Fn_Cross	3.65	3.59	3.57	3.60	3.69	3.72	3.31	2.54
Thrust (N)	361.63	537.39	402.95	496.31	440.56	292.91	399.71	1403.28
Thrust_Cross (N)	361.63	537.39	402.95	496.31	440.56	292.91	399.71	1403.28
Torque (N-m)	9.20	9.65	9.37	9.48	7.63	8.15	7.61	-0.38
Torque_Cross (N-m)	9.20	9.65	9.37	9.48	7.63	8.15	7.61	-0.38
KT	0.19	0.28	0.22	0.26	0.22	0.14	0.25	1.48
KT_Cross	0.19	0.28	0.22	0.26	0.22	0.14	0.25	1.48
KQ	0.0191	0.0208	0.0204	0.0203	0.0155	0.0163	0.0193	-0.0016
KQ_Cross	0.0191	0.0208	0.0204	0.0203	0.0155	0.0163	0.0193	-0.0016
eta	1.18	2.13	1.98	2.79	3.00	1.85	3.05	-278.82
eta_Cross	1.18	2.13	1.98	2.79	3.00	1.85	3.05	-278.82
TorqueBias (N-m)	0.82	0.40	0.93	0.60	1.24	0.77	0.60	-0.21
TorqueBias_Cross (N-m)	0.82	0.40	0.93	0.60	1.24	0.77	0.60	-0.21
ThrustBias (N)	27.69	-106.45	-145.73	-163.47	-27.11	96.02	-123.71	-874.95
ThrustBias_Cross (N)	27.69	-106.45	-145.73	-163.47	-27.11	96.02	-123.71	-874.95
KTFerr	0.16	0.22	0.24	0.23	0.23	0.23	0.21	0.03
KTFerr_Cross	0.16	0.22	0.24	0.23	0.23	0.23	0.21	0.03
KQFerr	0.0940	0.0987	0.0979	0.0925	0.0938	0.0942	0.0866	0.0485
KQFerr_Cross	0.0940	0.0987	0.0979	0.0925	0.0938	0.0942	0.0866	0.0485
etaFerr	0.21	0.34	0.46	0.54	0.52	0.52	0.57	0.21
etaFerr_Cross	0.21	0.34	0.46	0.54	0.52	0.52	0.57	0.21

Table C. 2

It	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33
It_Cross	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33
Pitch (deg)	0	0	0	0	0	0	0	0	0
Pitch_Cross (deg)	0	0	0	0	0	0	0	0	0
Yaw (deg)	15	15	15	15	15	15	15	15	15
Yaw_Cross (deg)	15	15	15	15	15	15	15	15	15
Ucom (m/s)	14.28	17.85	21.42	24.99	24.99	24.99	24.99	24.99	24.99
Ucom_Cross (m/s)	14.28	17.85	21.42	24.99	24.99	24.99	24.99	24.99	24.99
Uout (m/s)	13.79	17.24	20.69	24.14	24.14	24.14	24.14	24.14	24.14
Uout_Cross (m/s)	13.79	17.24	20.69	24.14	24.14	24.14	24.14	24.14	24.14
Mean_n (Hz)	23.32	23.24	23.04	23.75	21.40	20.73	20.19	19.64	17.14
Mean_n_Cross (Hz)	23.32	23.24	23.04	23.75	21.40	20.73	20.19	19.64	17.14
J	0.76	0.95	1.15	1.30	1.44	1.49	1.53	1.57	1.80
J_Cross	0.76	0.95	1.15	1.30	1.44	1.49	1.53	1.57	1.80
Jscaled	0.73	0.92	1.11	1.26	1.40	1.44	1.48	1.52	1.74
Jscaled_Cross	0.73	0.92	1.11	1.26	1.40	1.44	1.48	1.52	1.74
Fn	3.70	3.68	3.65	3.76	3.39	3.29	3.20	3.11	2.72
Fn_Cross	3.70	3.68	3.65	3.76	3.39	3.29	3.20	3.11	2.72
Thrust (N)	224.02	338.77	265.56	370.26	309.33	189.75	313.78	209.76	72.29
Thrust_Cross (N)	224.02	338.77	265.56	370.26	309.33	189.75	313.78	209.76	72.29
Torque (N-m)	7.89	7.10	5.85	5.45	3.66	2.97	2.31	1.60	0.12
Torque_Cross (N-m)	7.89	7.10	5.85	5.45	3.66	2.97	2.31	1.60	0.12
KT	0.11	0.17	0.14	0.18	0.18	0.12	0.21	0.15	0.07
KT_Cross	0.11	0.17	0.14	0.18	0.18	0.12	0.21	0.15	0.07
KQ	0.0160	0.0145	0.0122	0.0107	0.0088	0.0076	0.0063	0.0046	0.0005
KQ_Cross	0.0160	0.0145	0.0122	0.0107	0.0088	0.0076	0.0063	0.0046	0.0005
eta	0.82	1.72	1.98	3.35	4.62	3.61	7.88	7.80	39.91
eta_Cross	0.82	1.72	1.98	3.35	4.62	3.61	7.88	7.80	39.91
TorqueBias (N-m)	-0.62	-0.72	-0.27	-0.62	-0.45	-0.44	-0.44	-0.23	-0.41
TorqueBias_Cross (N-m)	-0.62	-0.72	-0.27	-0.62	-0.45	-0.44	-0.44	-0.23	-0.41
ThrustBias (N)	-951.90	-1011.12	-945.83	-982.36	-922.27	-855.26	-928.98	-885.66	-886.37
ThrustBias_Cross (N-m)	-951.90	-1011.12	-945.83	-982.36	-922.27	-855.26	-928.98	-885.66	-886.37
KTFerr	0.15	0.21	0.23	0.24	0.22	0.22	0.21	0.20	0.12
KTFerr_Cross	0.15	0.21	0.23	0.24	0.22	0.22	0.21	0.20	0.12
KQFerr	0.0927	0.0980	0.0987	0.0961	0.0910	0.0888	0.0867	0.0843	0.0674
KQFerr_Cross	0.0927	0.0980	0.0987	0.0961	0.0910	0.0888	0.0867	0.0843	0.0674
etaFerr	0.19	0.31	0.42	0.49	0.55	0.56	0.57	0.57	0.51
etaFerr_Cross	0.19	0.31	0.42	0.49	0.55	0.56	0.57	0.57	0.51

Table C. 3

It	0.33	0.33	0.33	0.33	0.33	0.33
It_Cross	0.33	0.33	0.33	0.33	0.33	0.33
Pitch (deg)	0	0	0	0	0	0
Pitch_Cross (deg)	0	0	0	0	0	0
Yaw (deg)	30	30	30	30	30	30
Yaw_Cross (deg)	30	30	30	30	30	30
Ucom (m/s)	14.28	17.85	21.42	24.99	24.99	24.99
Ucom_Cross (m/s)	14.28	17.85	21.42	24.99	24.99	24.99
Uout (m/s)	12.37	15.46	18.55	21.64	21.64	21.64
Uout_Cross (m/s)	12.37	15.46	18.55	21.64	21.64	21.64
Mean_n (Hz)	22.85	23.08	23.36	23.07	20.90	17.45
Mean_n_Cross (Hz)	22.85	23.08	23.36	23.07	20.90	17.45
J	0.77	0.96	1.13	1.34	1.48	1.77
J_Cross	0.77	0.96	1.13	1.34	1.48	1.77
Jscaled	0.67	0.83	0.98	1.16	1.28	1.53
Jscaled_Cross	0.67	0.83	0.98	1.16	1.28	1.53
Fn	3.62	3.66	3.70	3.66	3.31	2.76
Fn_Cross	3.62	3.66	3.70	3.66	3.31	2.76
Thrust (N)	602.20	357.40	365.07	384.61	184.17	92.30
Thrust_Cross (N)	602.20	357.40	365.07	384.61	184.17	92.30
Torque (N-m)	6.16	6.17	5.10	3.31	1.99	-0.29
Torque_Cross (N-m)	6.16	6.17	5.10	3.31	1.99	-0.29
KT	0.31	0.18	0.18	0.20	0.11	0.08
KT_Cross	0.31	0.18	0.18	0.20	0.11	0.08
KQ	0.0130	0.0128	0.0103	0.0069	0.0050	-0.0010
KQ_Cross	0.0130	0.0128	0.0103	0.0069	0.0050	-0.0010
eta	2.57	1.88	2.76	5.28	4.64	-19.39
eta_Cross	2.57	1.88	2.76	5.28	4.64	-19.39
TorqueBias (N-m)	-0.68	-0.64	-0.36	-0.16	-0.36	-0.12
TorqueBias_Cross (N-m)	-0.68	-0.64	-0.36	-0.16	-0.36	-0.12
ThrustBias (N)	-995.64	-963.75	-947.92	-912.39	-861.98	-801.45
ThrustBias_Cross (N-m)	-995.64	-963.75	-947.92	-912.39	-861.98	-801.45
KTFerr	0.13	0.18	0.22	0.24	0.24	0.20
KTFerr_Cross	0.13	0.18	0.22	0.24	0.24	0.20
KQFerr	0.0899	0.0960	0.0988	0.0981	0.0954	0.0834
KQFerr_Cross	0.0899	0.0960	0.0988	0.0981	0.0954	0.0834
etaFerr	0.1484	0.2506	0.3464	0.4473	0.5048	0.5726
etaFerr_Cross	0.1484	0.2506	0.3464	0.4473	0.5048	0.5726

Table C. 4

It	0.33	0.33	0.33	0.33	0.33	0.33
It_Cross	0.33	0.33	0.33	0.33	0.33	0.33
Pitch (deg)	15	15	15	15	15	15
Pitch_Cross (deg)	15	15	15	15	15	15
Yaw (deg)	0	0	0	0	0	0
Yaw_Cross (deg)	0	0	0	0	0	0
Ucom (m/s)	14.28	17.85	21.42	24.99	24.99	24.99
Ucom_Cross (m/s)	14.28	17.85	21.42	24.99	24.99	24.99
Uout (m/s)	13.79	17.24	20.69	24.14	24.14	24.14
Uout_Cross (m/s)	13.79	17.24	20.69	24.14	24.14	24.14
Mean_n (Hz)	20.77	21.00	21.39	20.79	20.11	15.60
Mean_n_Cross (Hz)	20.77	21.00	21.39	20.79	20.11	15.60
J	0.85	1.05	1.24	1.49	1.54	1.98
J_Cross	0.85	1.05	1.24	1.49	1.54	1.98
Jscaled	0.82	1.02	1.20	1.44	1.49	1.91
Jscaled_Cross	0.82	1.02	1.20	1.44	1.49	1.91
Fn	3.29	3.33	3.39	3.30	3.19	2.47
Fn_Cross	3.29	3.33	3.39	3.30	3.19	2.47
Thrust (N)	116.88	109.37	63.02	5.48	-81.28	-171.99
Thrust_Cross (N)	116.88	109.37	63.02	5.48	-81.28	-171.99
Torque (N-m)	9.16	8.54	8.73	7.78	6.39	0.81
Torque_Cross (N-m)	9.16	8.54	8.73	7.78	6.39	0.81
KT	0.07	0.07	0.04	0.00	-0.05	-0.19
KT_Cross	0.07	0.07	0.04	0.00	-0.05	-0.19
KQ	0.0234	0.0214	0.0211	0.0199	0.0174	0.0037
KQ_Cross	0.0234	0.0214	0.0211	0.0199	0.0174	0.0037
eta	0.41	0.51	0.34	0.04	-0.74	-15.84
eta_Cross	0.41	0.51	0.34	0.04	-0.74	-15.84
TorqueBias (N-m)	0.91	1.27	1.07	1.09	1.38	1.10
TorqueBias_Cross (N-m)	0.91	1.27	1.07	1.09	1.38	1.10
ThrustBias (N)	-71.14	-89.88	-47.93	-87.92	30.12	-20.47
ThrustBias_Cross (N-m)	-71.14	-89.88	-47.93	-87.92	30.12	-20.47
KTFerr	0.1803	0.2242	0.2384	0.2174	0.2075	0.0398
KTFerr_Cross	0.1803	0.2242	0.2384	0.2174	0.2075	0.0398
KQFerr	0.0958	0.0990	0.0975	0.0890	0.0864	0.0499
KQFerr_Cross	0.0958	0.0990	0.0975	0.0890	0.0864	0.0499
etaFerr	0.25	0.37	0.47	0.56	0.57	0.24
etaFerr_Cross	0.25	0.37	0.47	0.56	0.57	0.24

Table C. 5

It	0.33	0.33	0.33	0.33	0.33	0.33
It_Cross	0.33	0.33	0.33	0.33	0.33	0.33
Pitch (deg)	15	15	15	15	15	15
Pitch_Cross (deg)	15	15	15	15	15	15
Yaw (deg)	15	15	15	15	15	15
Yaw_Cross (deg)	15	15	15	15	15	15
Ucom (m/s)	14.28	17.85	21.42	24.99	24.99	24.99
Ucom_Cross (m/s)	14.28	17.85	21.42	24.99	24.99	24.99
Uout (m/s)	13.32	16.65	19.99	23.32	23.32	23.32
Uout_Cross (m/s)	13.32	16.65	19.99	23.32	23.32	23.32
Mean_n (Hz)	20.49	21.50	22.08	22.08	17.72	18.00
Mean_n_Cross (Hz)	20.49	21.50	22.08	22.08	17.72	18.00
J	0.86	1.03	1.20	1.40	1.75	1.72
J_Cross	0.86	1.03	1.20	1.40	1.75	1.72
Jscaled	0.80	0.96	1.12	1.31	1.63	1.60
Jscaled_Cross	0.80	0.96	1.12	1.31	1.63	1.60
Fn	3.25	3.41	3.50	3.50	2.81	2.85
Fn_Cross	3.25	3.41	3.50	3.50	2.81	2.85
Thrust (N)	107.98	30.47	-37.06	-121.35	-54.13	-105.18
Thrust_Cross (N)	107.98	30.47	-37.06	-121.35	-54.13	-105.18
Torque (N-m)	8.81	8.41	8.08	6.43	2.02	2.42
Torque_Cross (N-m)	8.81	8.41	8.08	6.43	2.02	2.42
KT	0.07	0.02	-0.02	-0.07	-0.05	-0.09
KT_Cross	0.07	0.02	-0.02	-0.07	-0.05	-0.09
KQ	0.0232	0.0201	0.0183	0.0146	0.0071	0.0082
KQ_Cross	0.0232	0.0201	0.0183	0.0146	0.0071	0.0082
eta	0.39	0.14	-0.20	-0.97	-1.71	-2.73
eta_Cross	0.39	0.14	-0.20	-0.97	-1.71	-2.73
TorqueBias (N-m)	1.22	1.33	1.69	1.59	1.31	1.40
TorqueBias_Cross (N-m)	1.22	1.33	1.69	1.59	1.31	1.40
ThrustBias (N)	-113.92	-102.97	-61.73	-101.69	-247.54	-150.62
ThrustBias_Cross (N-m)	-113.92	-102.97	-61.73	-101.69	-247.54	-150.62
KTFerr	0.18	0.21	0.24	0.23	0.17	0.18
KTFerr_Cross	0.18	0.21	0.24	0.23	0.17	0.18
KQFerr	0.0953	0.0985	0.0986	0.0946	0.0769	0.0788
KQFerr_Cross	0.0953	0.0985	0.0986	0.0946	0.0769	0.0788
etaFerr	0.24	0.33	0.43	0.52	0.57	0.57
etaFerr_Cross	0.24	0.33	0.43	0.52	0.57	0.57

Table C. 6

It	0.33	0.33	0.33	0.33	0.33	0.33	0.33
It_Cross	0.33	0.33	0.33	0.33	0.33	0.33	0.33
Pitch (deg)	15	15	15	15	15	15	15
Pitch_Cross (deg)	15	15	15	15	15	15	15
Yaw (deg)	30	30	30	30	30	30	30
Yaw_Cross (deg)	30	30	30	30	30	30	30
Ucom (m/s)	14.28	17.85	21.42	21.42	21.42	21.42	21.42
Ucom_Cross (m/s)	14.28	17.85	21.42	21.42	21.42	21.42	21.42
Uout (m/s)	11.95	14.93	17.92	17.92	17.92	17.92	17.92
Uout_Cross (m/s)	11.95	14.93	17.92	17.92	17.92	17.92	17.92
Mean_n (Hz)	19.24	19.71	19.54	19.51	18.76	17.07	13.26
Mean_n_Cross (Hz)	19.24	19.71	19.54	19.51	18.76	17.07	13.26
J	0.92	1.12	1.36	1.36	1.41	1.55	2.00
J_Cross	0.92	1.12	1.36	1.36	1.41	1.55	2.00
Jscaled	0.77	0.94	1.13	1.14	1.18	1.30	1.67
Jscaled_Cross	0.77	0.94	1.13	1.14	1.18	1.30	1.67
Fn	3.05	3.12	3.10	3.09	2.97	2.71	2.10
Fn_Cross	3.05	3.12	3.10	3.09	2.97	2.71	2.10
Thrust (N)	-37.00	18.89	11.96	-58.89	-53.46	-65.09	-118.45
Thrust_Cross (N)	-37.00	18.89	11.96	-58.89	-53.46	-65.09	-118.45
Torque (N-m)	8.99	9.69	9.29	8.38	7.89	5.43	1.28
Torque_Cross (N-m)	8.99	9.69	9.29	8.38	7.89	5.43	1.28
KT	-0.03	0.01	0.01	-0.04	-0.04	-0.06	-0.18
KT_Cross	-0.03	0.01	0.01	-0.04	-0.04	-0.06	-0.18
KQ	0.0268	0.0275	0.0269	0.0243	0.0247	0.0206	0.0080
KQ_Cross	0.0268	0.0275	0.0269	0.0243	0.0247	0.0206	0.0080
eta	-0.12	0.07	0.06	-0.31	-0.31	-0.61	-6.07
eta_Cross	-0.12	0.07	0.06	-0.31	-0.31	-0.61	-6.07
TorqueBias (N-m)	1.43	1.38	1.34	1.52	1.49	1.50	1.15
TorqueBias_Cross (N-m)	1.43	1.38	1.34	1.52	1.49	1.50	1.15
ThrustBias (N)	-55.70	-162.04	-158.42	-97.61	-126.18	-139.67	-169.34
ThrustBias_Cross (N-m)	-55.70	-162.04	-158.42	-97.61	-126.18	-139.67	-169.34
KTFerr	0.16	0.21	0.24	0.24	0.24	0.24	0.15
KTFerr_Cross	0.16	0.21	0.24	0.24	0.24	0.24	0.15
KQFerr	0.0940	0.0983	0.0985	0.0985	0.0978	0.0948	0.0735
KQFerr_Cross	0.0940	0.0983	0.0985	0.0985	0.0978	0.0948	0.0735
etaFerr	0.21	0.32	0.43	0.43	0.46	0.51	0.55
etaFerr_Cross	0.21	0.32	0.43	0.43	0.46	0.51	0.55

Table C. 7

It	0.33	0.33	0.33	0.33	0.33
It_Cross	0.33	0.33	0.33	0.33	0.33
Pitch (deg)	7.5	7.5	7.5	7.5	7.5
Pitch_Cross (deg)	7.5	7.5	7.5	7.5	7.5
Yaw (deg)	0	0	0	0	0
Yaw_Cross (deg)	0	0	0	0	0
Ucom (m/s)	14.28	17.85	21.42	21.42	24.99
Ucom_Cross (m/s)	14.28	17.85	21.42	21.42	24.99
Uout (m/s)	14.16	17.70	21.24	21.24	24.78
Uout_Cross (m/s)	14.16	17.70	21.24	21.24	24.78
Mean_n (Hz)	21.99	21.95	21.96	22.15	22.17
Mean_n_Cross (Hz)	21.99	21.95	21.96	22.15	22.17
J	0.80	1.01	1.21	1.20	1.39
J_Cross	0.80	1.01	1.21	1.20	1.39
Jscaled	0.80	1.00	1.20	1.19	1.38
Jscaled_Cross	0.80	1.00	1.20	1.19	1.38
Fn	3.49	3.48	3.48	3.51	3.51
Fn_Cross	3.49	3.48	3.48	3.51	3.51
Thrust (N)	109.69	5.36	32.90	-36.60	-46.75
Thrust_Cross (N)	109.69	5.36	32.90	-36.60	-46.75
Torque (N-m)	8.75	7.57	7.52	7.58	7.52
Torque_Cross (N-m)	8.75	7.57	7.52	7.58	7.52
KT	0.06	0.00	0.02	-0.02	-0.03
KT_Cross	0.06	0.00	0.02	-0.02	-0.03
KQ	0.0200	0.0173	0.0172	0.0170	0.0169
KQ_Cross	0.0200	0.0173	0.0172	0.0170	0.0169
eta	0.39	0.03	0.21	-0.22	-0.34
eta_Cross	0.39	0.03	0.21	-0.22	-0.34
TorqueBias (N-m)	2.01	2.13	2.01	1.85	1.83
TorqueBias_Cross (N-m)	2.01	2.13	2.01	1.85	1.83
ThrustBias (N)	-24.76	54.32	22.81	103.51	124.98
ThrustBias_Cross (N-m)	-24.76	54.32	22.81	103.51	124.98
KTFerr	0.17	0.22	0.24	0.24	0.23
KTFerr_Cross	0.17	0.22	0.24	0.24	0.23
KQFerr	0.0950	0.0989	0.0975	0.0977	0.0916
KQFerr_Cross	0.0950	0.0989	0.0975	0.0977	0.0916
etaFerr	0.23	0.36	0.47	0.46	0.54
etaFerr_Cross	0.23	0.36	0.47	0.46	0.54

Table C. 8

It	0.33	0.33	0.33	0.33	0.33	0.33
It_Cross	0.33	0.33	0.33	0.33	0.33	0.33
Pitch (deg)	7.5	7.5	7.5	7.5	7.5	7.5
Pitch_Cross (deg)	7.5	7.5	7.5	7.5	7.5	7.5
Yaw (deg)	15	15	15	15	15	15
Yaw_Cross (deg)	15	15	15	15	15	15
Ucom (m/s)	14.28	17.85	21.42	24.99	24.99	24.99
Ucom_Cross (m/s)	14.28	17.85	21.42	24.99	24.99	24.99
Uout (m/s)	13.68	17.09	20.51	23.93	23.93	23.93
Uout_Cross (m/s)	13.68	17.09	20.51	23.93	23.93	23.93
Mean_n (Hz)	21.92	21.58	21.73	21.92	18.88	17.04
Mean_n_Cross (Hz)	21.92	21.58	21.73	21.92	18.88	17.04
J	0.81	1.02	1.22	1.41	1.64	1.81
J_Cross	0.81	1.02	1.22	1.41	1.64	1.81
Jscaled	0.77	0.98	1.17	1.35	1.57	1.74
Jscaled_Cross	0.77	0.98	1.17	1.35	1.57	1.74
Fn	3.47	3.42	3.44	3.47	2.99	2.70
Fn_Cross	3.47	3.42	3.44	3.47	2.99	2.70
Thrust (N)	135.29	89.94	66.36	-135.32	-89.11	-307.63
Thrust_Cross (N)	135.29	89.94	66.36	-135.32	-89.11	-307.63
Torque (N-m)	8.77	7.61	6.86	5.55	2.17	0.46
Torque_Cross (N-m)	8.77	7.61	6.86	5.55	2.17	0.46
KT	0.08	0.05	0.04	-0.08	-0.07	-0.29
KT_Cross	0.08	0.05	0.04	-0.08	-0.07	-0.29
KQ	0.0202	0.0180	0.0160	0.0128	0.0067	0.0017
KQ_Cross	0.0202	0.0180	0.0160	0.0128	0.0067	0.0017
eta	0.47	0.45	0.44	-1.29	-2.52	-45.90
eta_Cross	0.47	0.45	0.44	-1.29	-2.52	-45.90
TorqueBias (N-m)	1.23	1.23	1.45	1.65	1.32	1.19
TorqueBias_Cross (N-m)	1.23	1.23	1.45	1.65	1.32	1.19
ThrustBias (N)	-25.78	-54.73	-11.94	135.78	57.75	136.59
ThrustBias_Cross (N-m)	-25.78	-54.73	-11.94	135.78	57.75	136.59
KTFerr	0.16	0.22	0.24	0.23	0.19	0.13
KTFerr_Cross	0.16	0.22	0.24	0.23	0.19	0.13
KQFerr	0.0942	0.0987	0.0980	0.0929	0.0812	0.0678
KQFerr_Cross	0.0942	0.0987	0.0980	0.0929	0.0812	0.0678
etaFerr	0.21	0.34	0.45	0.53	0.57	0.52
etaFerr_Cross	0.21	0.34	0.45	0.53	0.57	0.52

Table C. 9

It	0.33	0.33	0.33	0.33	0.33	0.33	0.33
It_Cross	0.33	0.33	0.33	0.33	0.33	0.33	0.33
Pitch (deg)	7.5	7.5	7.5	7.5	7.5	7.5	7.5
Pitch_Cross (deg)	7.5	7.5	7.5	7.5	7.5	7.5	7.5
Yaw (deg)	30	30	30	30	30	30	30
Yaw_Cross (deg)	30	30	30	30	30	30	30
Ucom (m/s)	14.28	14.28	17.85	21.42	21.42	21.42	21.42
Ucom_Cross (m/s)	14.28	14.28	17.85	21.42	21.42	21.42	21.42
Uout (m/s)	12.26	12.26	15.33	18.39	18.39	18.39	18.39
Uout_Cross (m/s)	12.26	12.26	15.33	18.39	18.39	18.39	18.39
Mean_n (Hz)	13.32	21.96	22.22	22.32	20.38	17.78	15.16
Mean_n_Cross (Hz)	13.32	21.96	22.22	22.32	20.38	17.78	15.16
J	1.33	0.80	0.99	1.19	1.30	1.49	1.75
J_Cross	1.33	0.80	0.99	1.19	1.30	1.49	1.75
Jscaled	1.14	0.69	0.85	1.02	1.12	1.28	1.50
Jscaled_Cross	1.14	0.69	0.85	1.02	1.12	1.28	1.50
Fn	2.11	3.48	3.52	3.54	3.23	2.82	2.40
Fn_Cross	2.11	3.48	3.52	3.54	3.23	2.82	2.40
Thrust (N)	24.72	86.06	26.34	19.10	-56.44	-89.21	-135.08
Thrust_Cross (N)	24.72	86.06	26.34	19.10	-56.44	-89.21	-135.08
Torque (N-m)	1.81	8.67	8.28	7.10	5.90	2.67	0.81
Torque_Cross (N-m)	1.81	8.67	8.28	7.10	5.90	2.67	0.81
KT	0.04	0.05	0.01	0.01	-0.04	-0.08	-0.16
KT_Cross	0.04	0.05	0.01	0.01	-0.04	-0.08	-0.16
KQ	0.0112	0.0198	0.0185	0.0157	0.0157	0.0093	0.0039
KQ_Cross	0.0112	0.0198	0.0185	0.0157	0.0157	0.0093	0.0039
eta	0.61	0.27	0.11	0.11	-0.42	-1.68	-9.77
eta_Cross	0.61	0.27	0.11	0.11	-0.42	-1.68	-9.77
TorqueBias (N-m)	1.46	1.37	1.25	1.63	1.37	1.48	1.32
TorqueBias_Cross (N-m)	1.46	1.37	1.25	1.63	1.37	1.48	1.32
ThrustBias (N)	-164.00	-114.21	-164.88	-166.74	-140.64	-143.99	-94.53
ThrustBias_Cross (N-m)	-164.00	-114.21	-164.88	-166.74	-140.64	-143.99	-94.53
KTFerr	0.24	0.13	0.19	0.22	0.24	0.24	0.20
KTFerr_Cross	0.24	0.13	0.19	0.22	0.24	0.24	0.20
KQFerr	0.0984	0.0909	0.0966	0.0990	0.0987	0.0954	0.0855
KQFerr_Cross	0.0984	0.0909	0.0966	0.0990	0.0987	0.0954	0.0855
etaFerr	0.44	0.16	0.27	0.37	0.42	0.50	0.57
etaFerr_Cross	0.44	0.16	0.27	0.37	0.42	0.50	0.57

Table C. 10

It	0.5	0.5	0.5	0.5	0.5
It_Cross	0.5	0.5	0.5	0.5	0.5
Pitch (deg)	0	0	0	0	0
Pitch_Cross (deg)	0	0	0	0	0
Yaw (deg)	0	0	0	0	0
Yaw_Cross (deg)	0	0	0	0	0
Ucom (m/s)	14.28	17.85	21.42	21.42	21.42
Ucom_Cross (m/s)	14.28	17.85	21.42	21.42	21.42
Uout (m/s)	14.28	17.85	21.42	21.42	21.42
Uout_Cross (m/s)	14.28	17.85	21.42	21.42	21.42
Mean_n (Hz)	19.35	18.55	18.68	18.65	18.39
Mean_n_Cross (Hz)	19.35	18.55	18.68	18.65	18.39
J	0.91	1.19	1.42	1.42	1.44
J_Cross	0.91	1.19	1.42	1.42	1.44
Jscaled	0.91	1.19	1.42	1.42	1.44
Jscaled_Cross	0.91	1.19	1.42	1.42	1.44
Fn	3.07	2.94	2.96	2.96	2.91
Fn_Cross	3.07	2.94	2.96	2.96	2.91
Thrust (N)	163.94	219.81	222.94	218.84	162.17
Thrust_Cross (N)	163.94	219.81	222.94	218.84	162.17
Torque (N-m)	10.37	10.85	10.61	10.36	10.47
Torque_Cross (N-m)	10.37	10.85	10.61	10.36	10.47
KT	0.12	0.17	0.17	0.17	0.13
KT_Cross	0.12	0.17	0.17	0.17	0.13
KQ	0.0306	0.0348	0.0336	0.0329	0.0342
KQ_Cross	0.0306	0.0348	0.0336	0.0329	0.0342
eta	0.57	0.95	1.17	1.18	0.87
eta_Cross	0.57	0.95	1.17	1.18	0.87
TorqueBias (N-m)	0.33	-0.17	-0.07	0.16	-0.34
TorqueBias_Cross (N-m)	0.33	-0.17	-0.07	0.16	-0.34
ThrustBias (N)	-224.82	-307.04	-331.00	-296.55	-278.93
ThrustBias_Cross (N-m)	-224.82	-307.04	-331.00	-296.55	-278.93
KTFerr	0.20	0.24	0.22	0.22	0.22
KTFerr_Cross	0.20	0.24	0.22	0.22	0.22
KQFerr	0.0979	0.0976	0.0899	0.0898	0.0888
KQFerr_Cross	0.0979	0.0976	0.0899	0.0898	0.0888
etaFerr	0.30	0.46	0.55	0.55	0.56
etaFerr_Cross	0.30	0.46	0.55	0.55	0.56

APPENDIX D: RAW DATA AND POWER SPECTRAL DENSITY PLOTS

The following representative plots show the raw data collected in the left column and their respective power spectral densities in the right column.

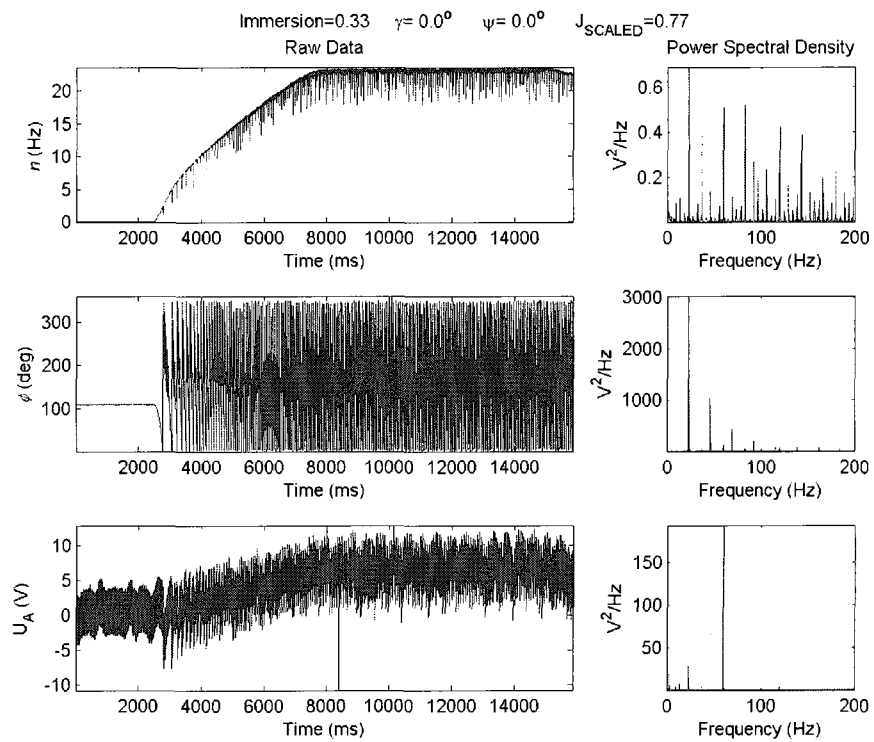


Figure D. 1

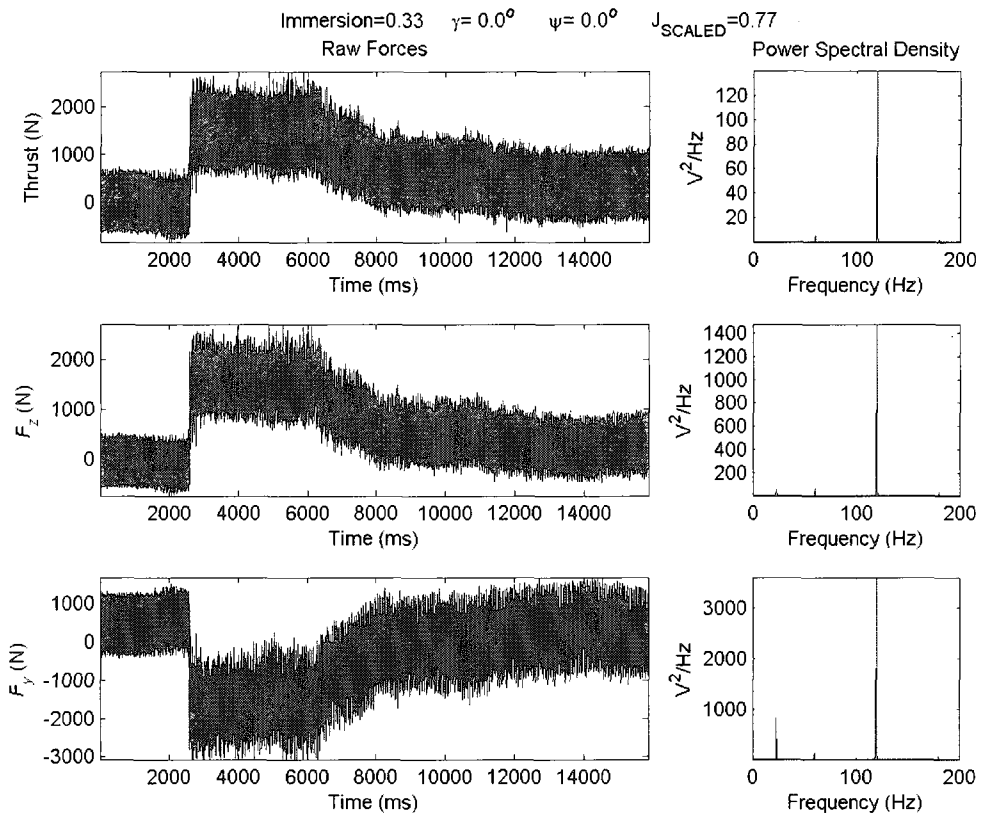


Figure D. 2

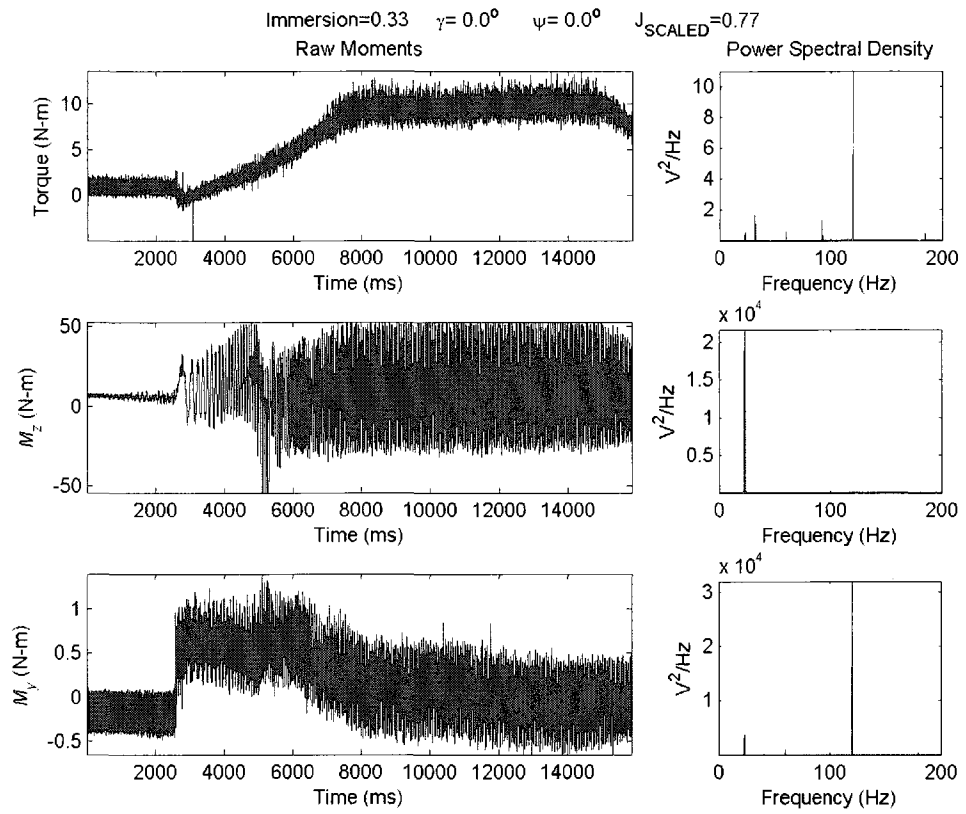


Figure D. 3

APPENDIX E: FILTERED DATA FOR ROTATIONAL RATE, THRUST AND TORQUE

The following plots show the rotational rate, thrust and torque for the steady state section of the respective experiment.

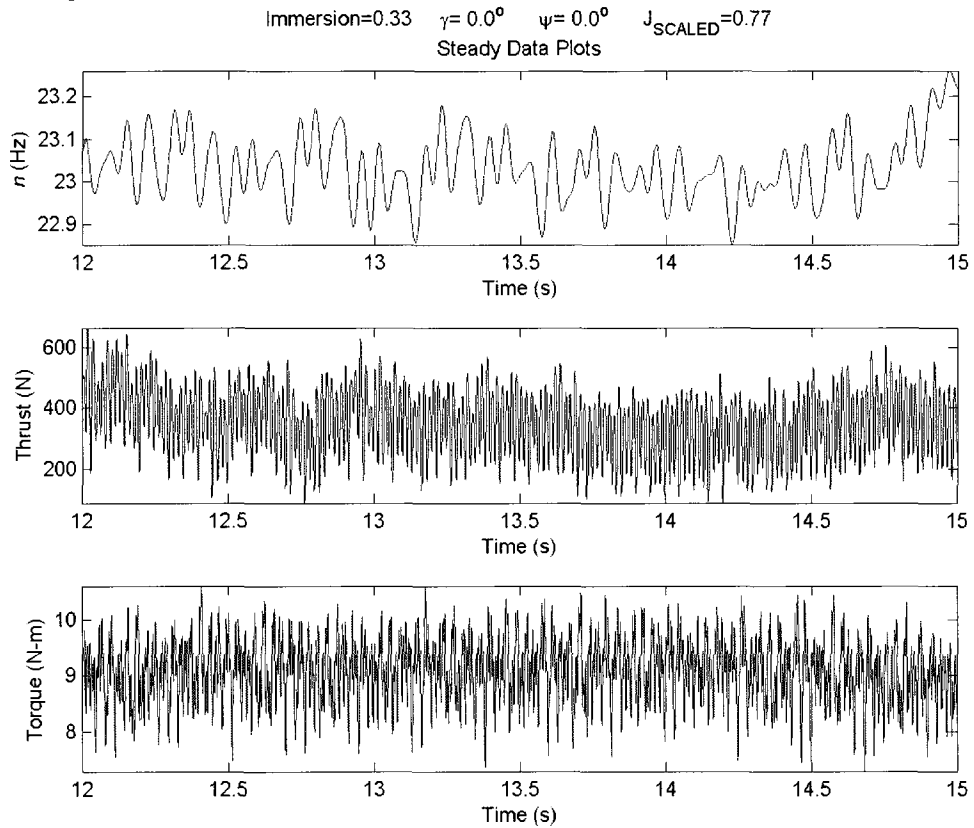


Figure E. 1

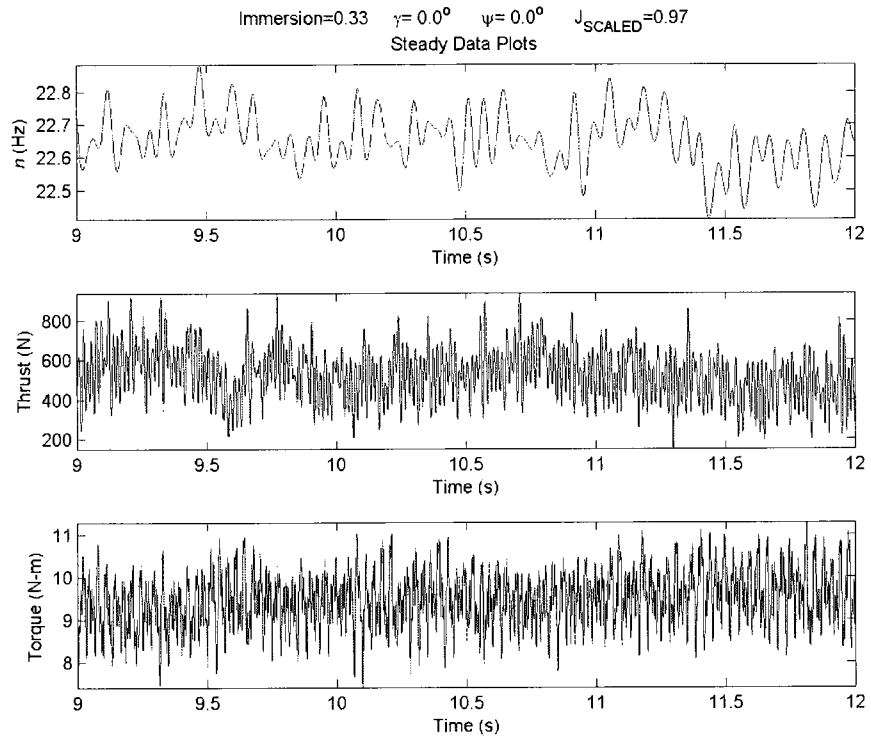


Figure E. 2

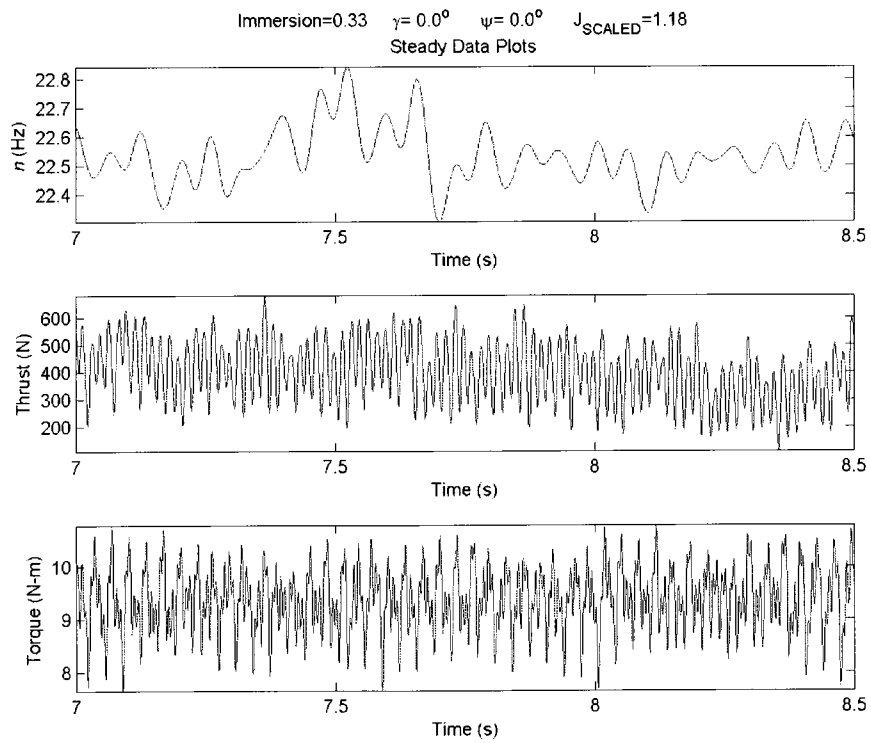


Figure E. 3

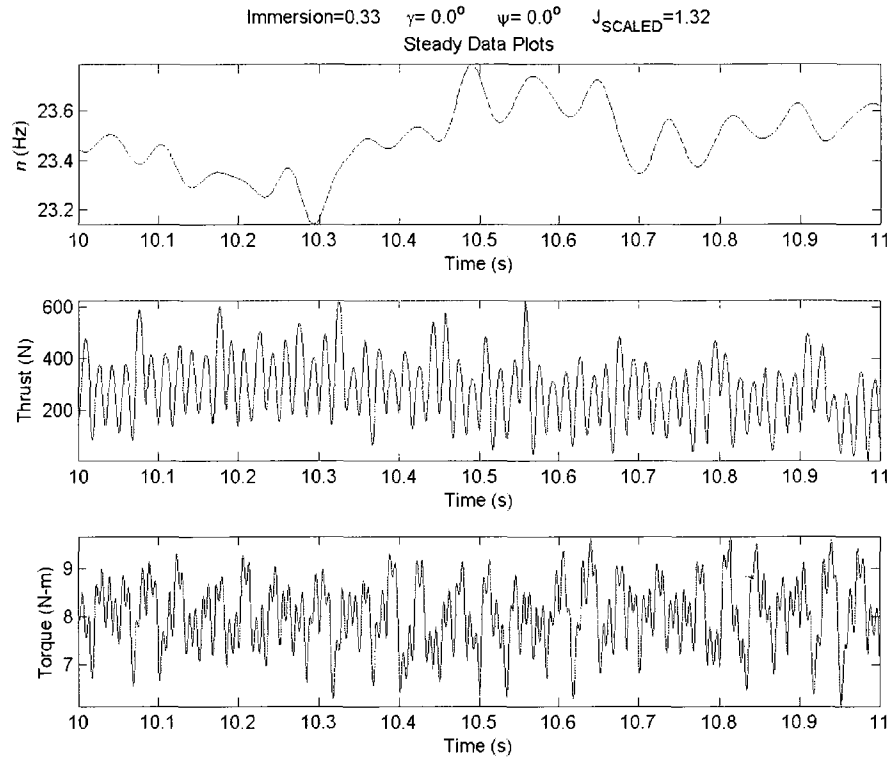


Figure E. 4

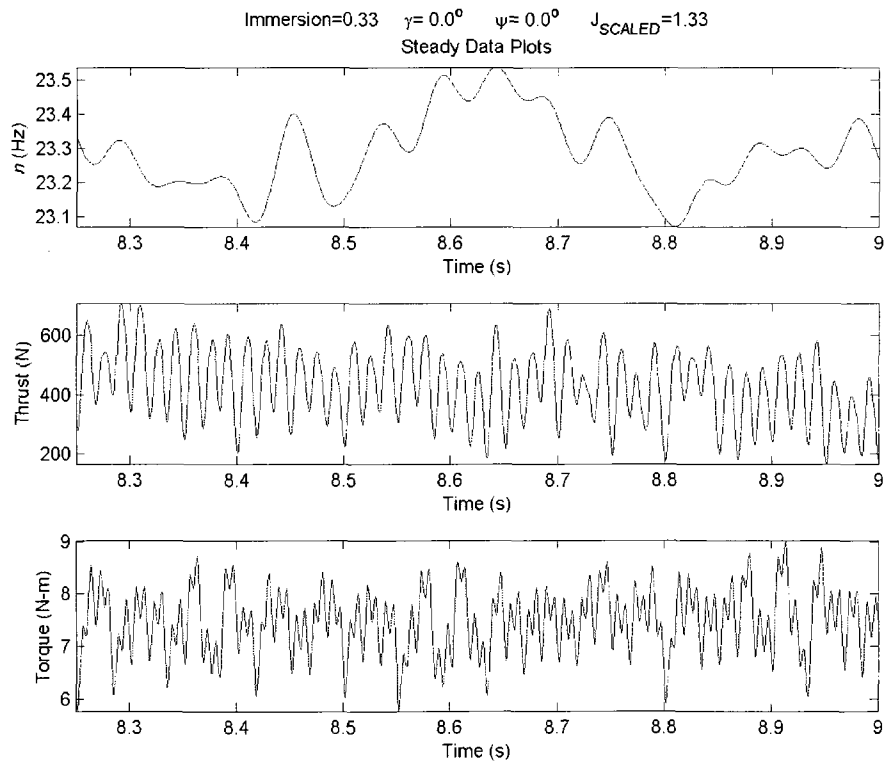


Figure E. 5

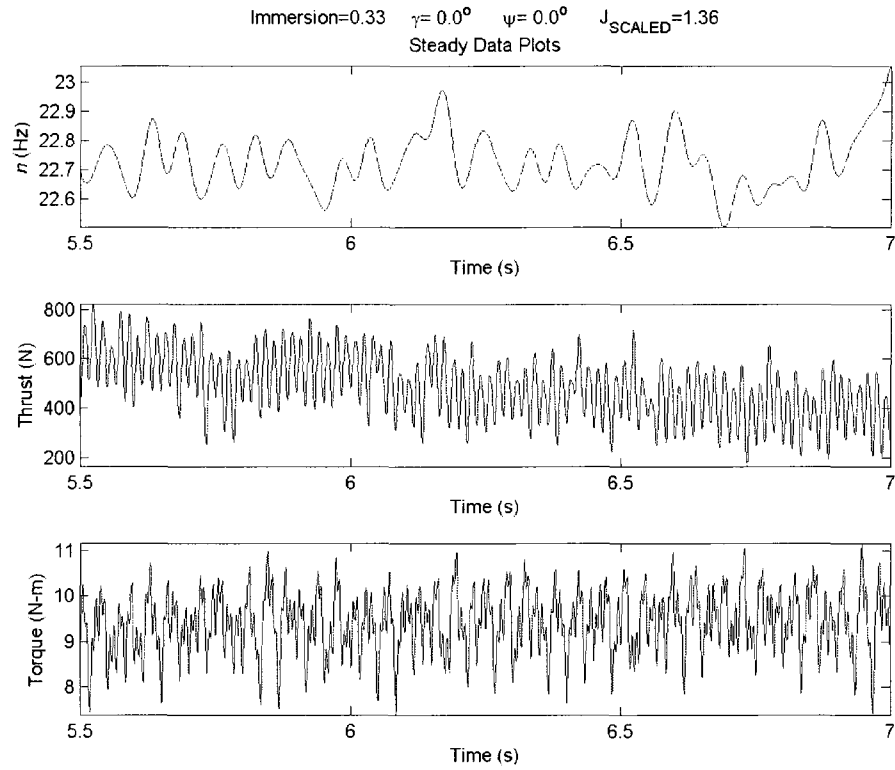


Figure E. 6

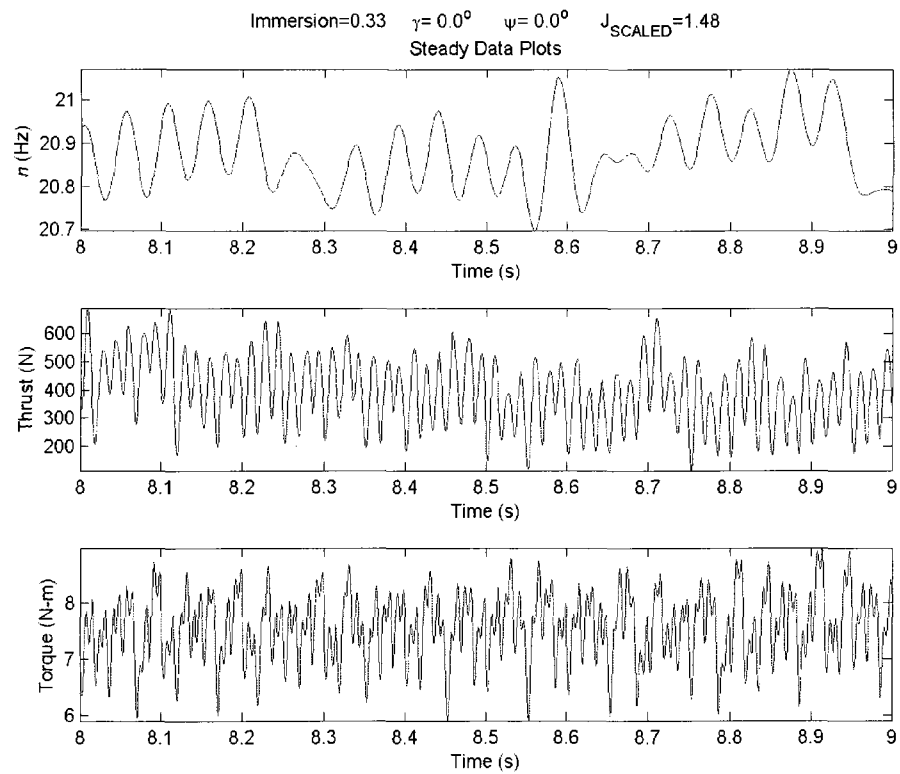


Figure E. 7

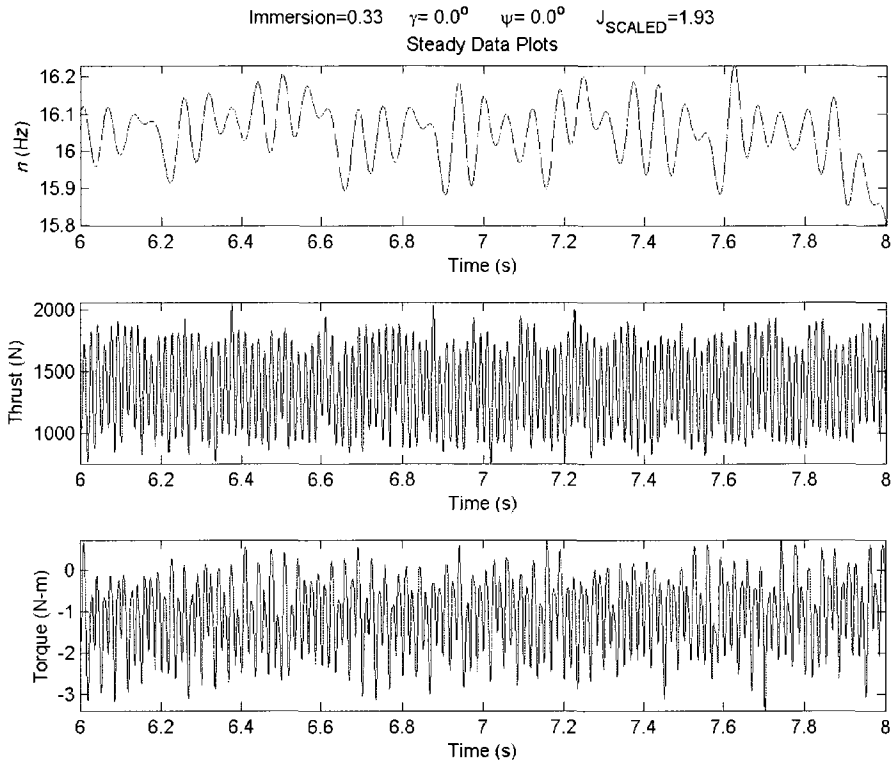


Figure E. 8

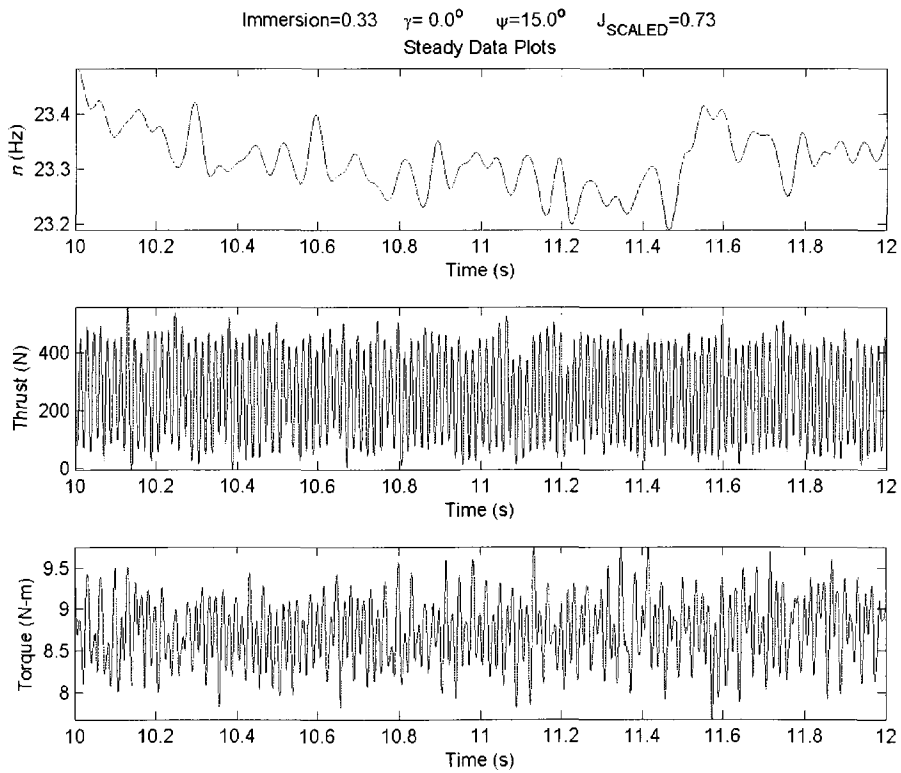


Figure E. 9

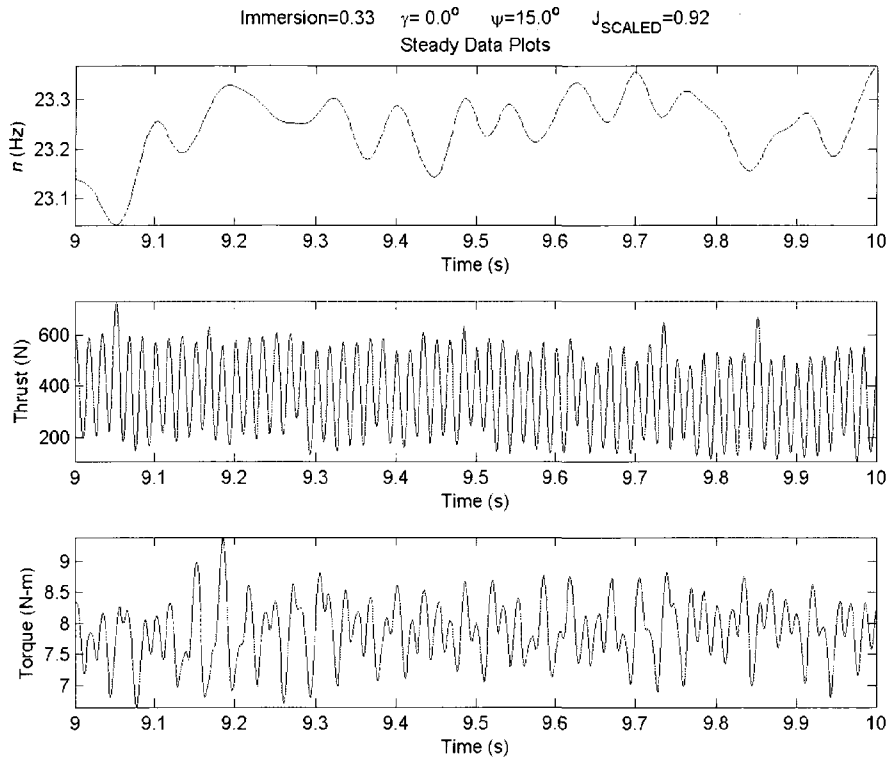


Figure E. 10

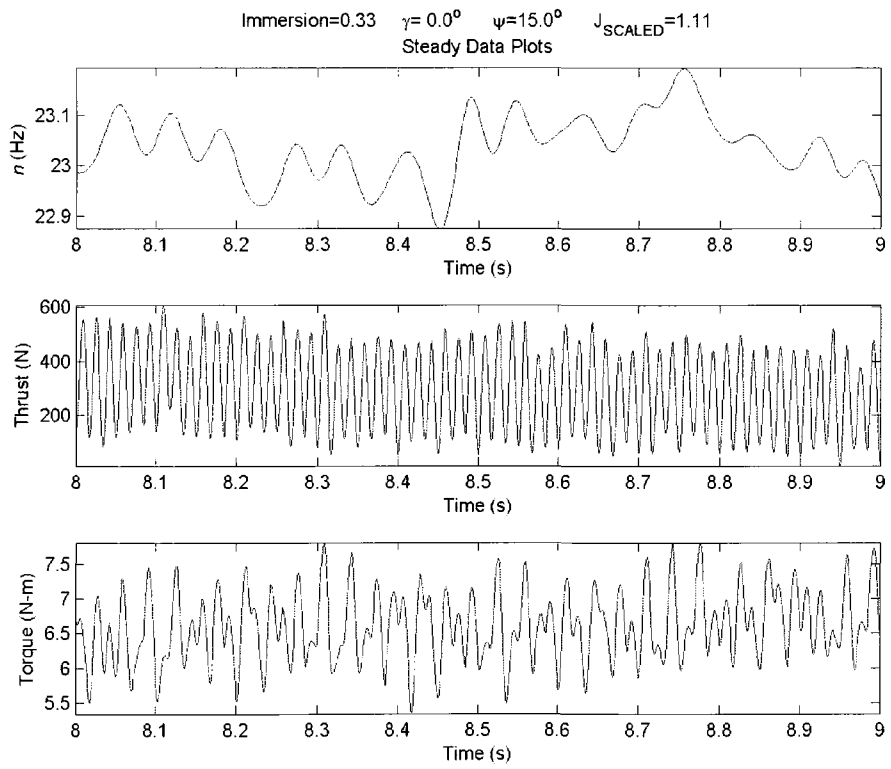


Figure E. 11

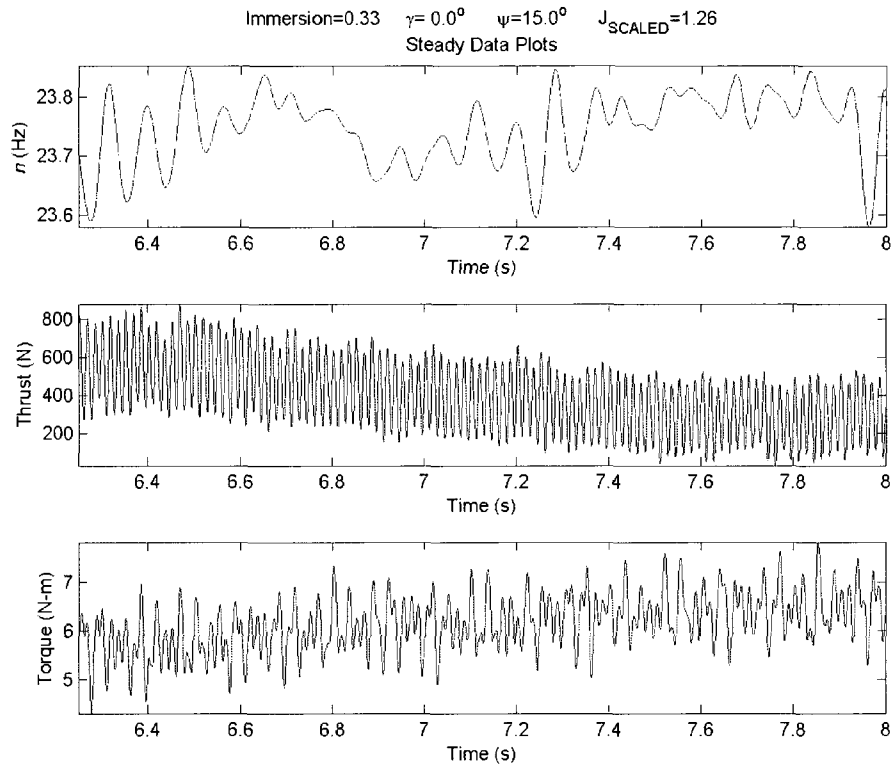


Figure E. 12

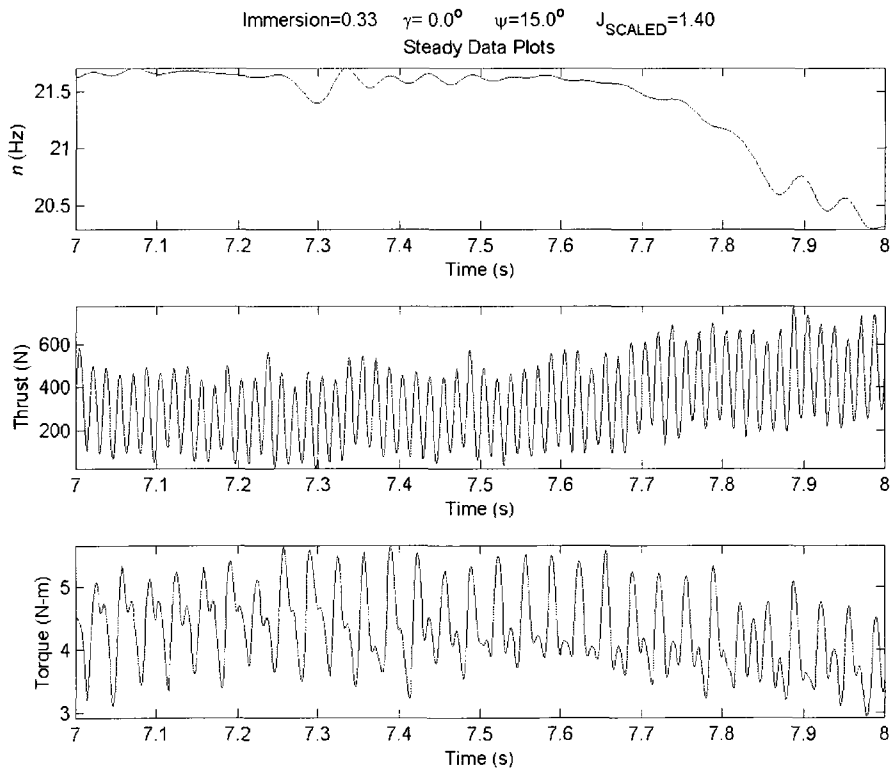


Figure E. 13

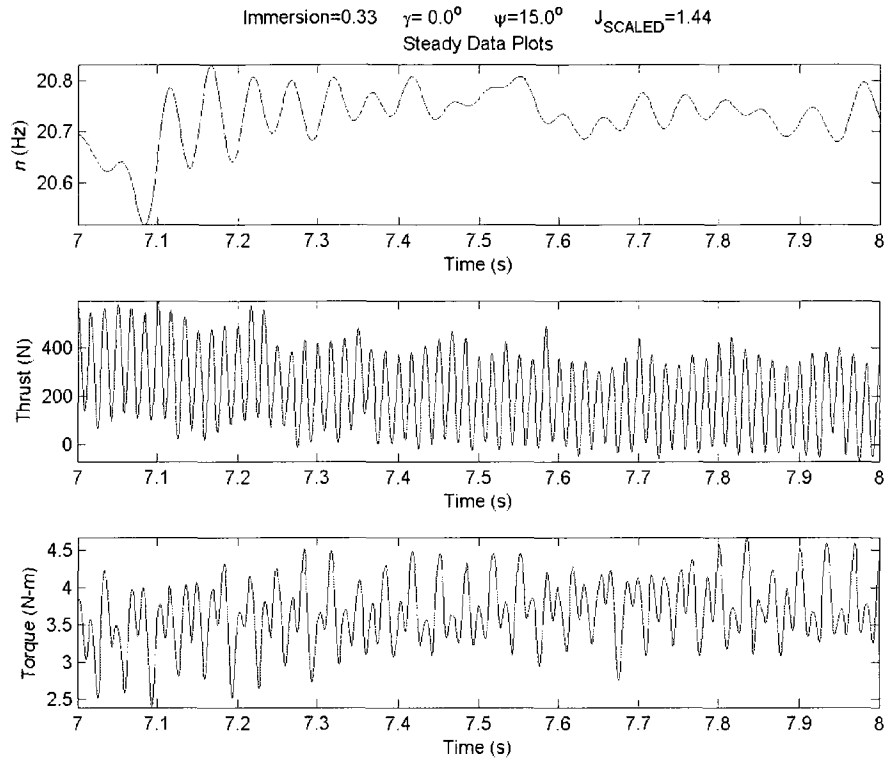


Figure E. 14

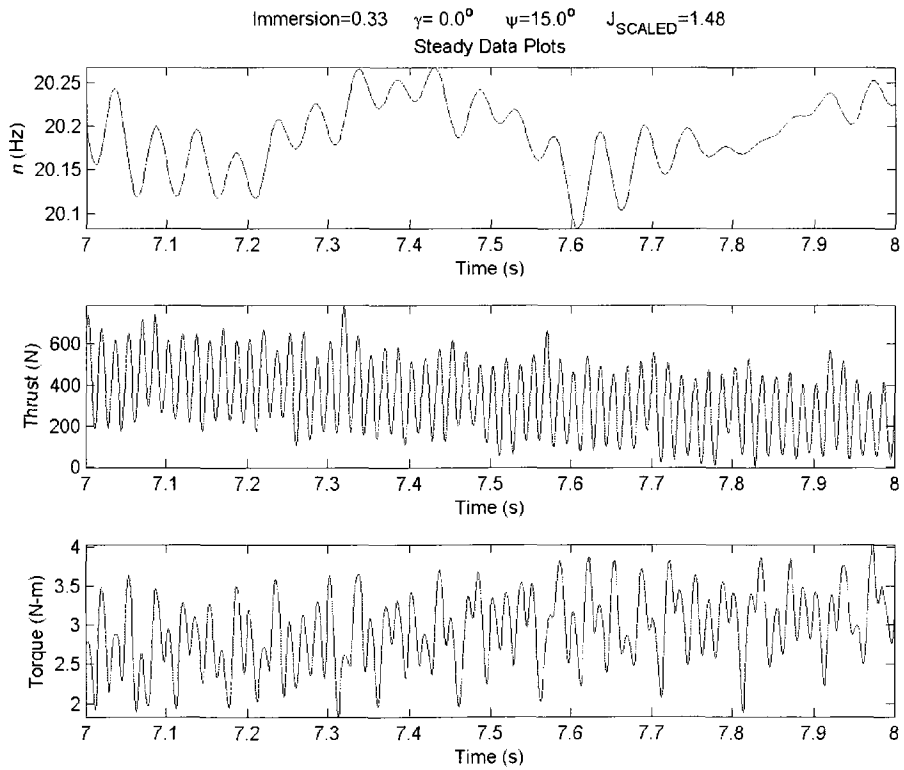


Figure E. 15

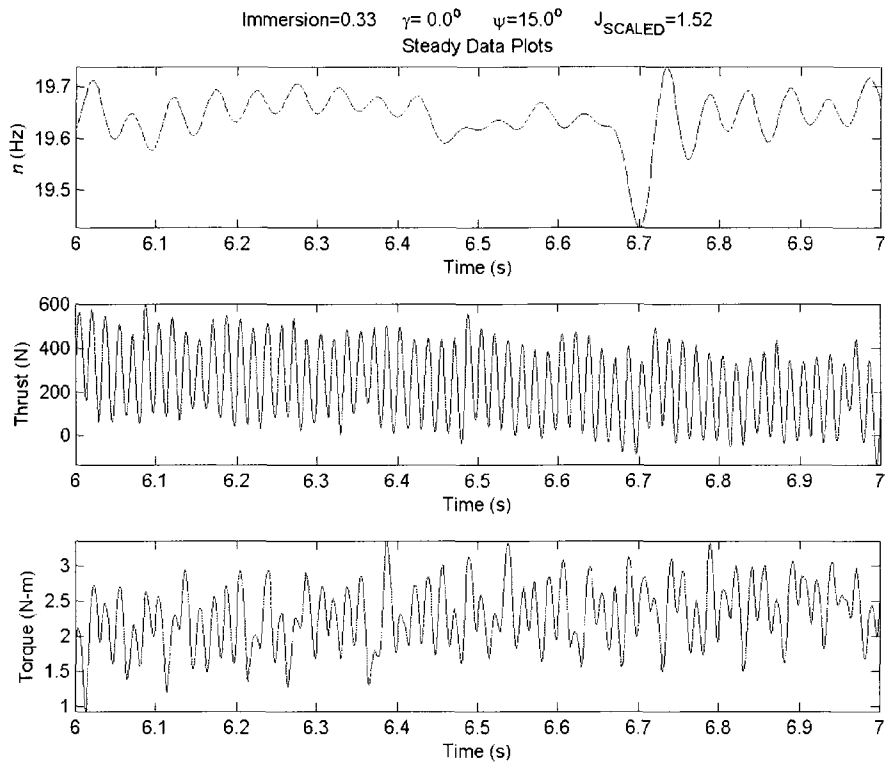


Figure E. 16

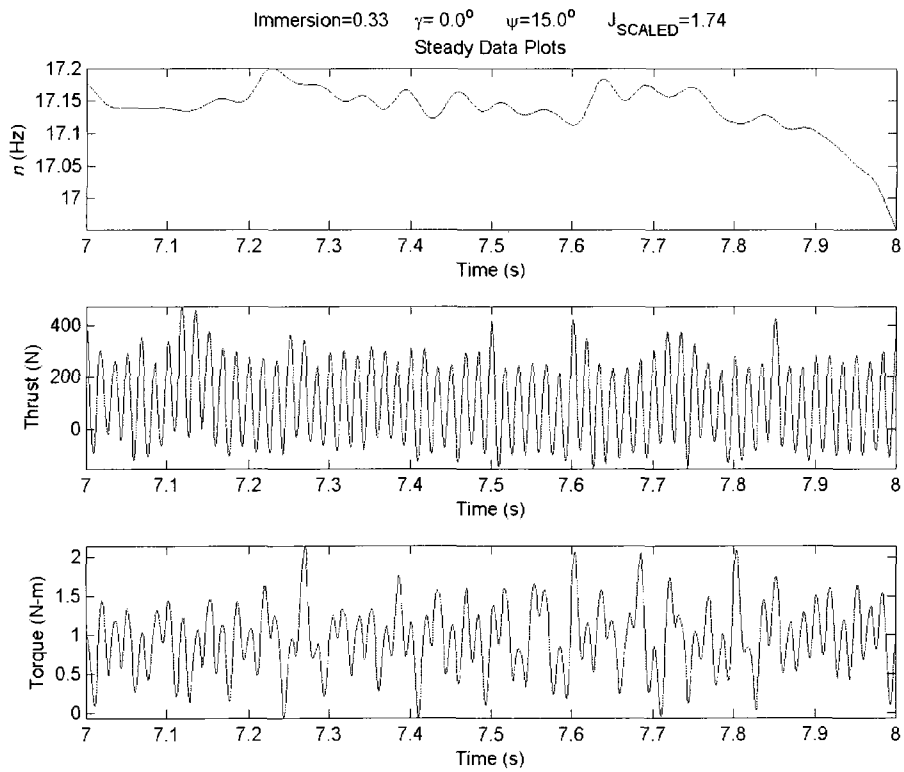


Figure E. 17

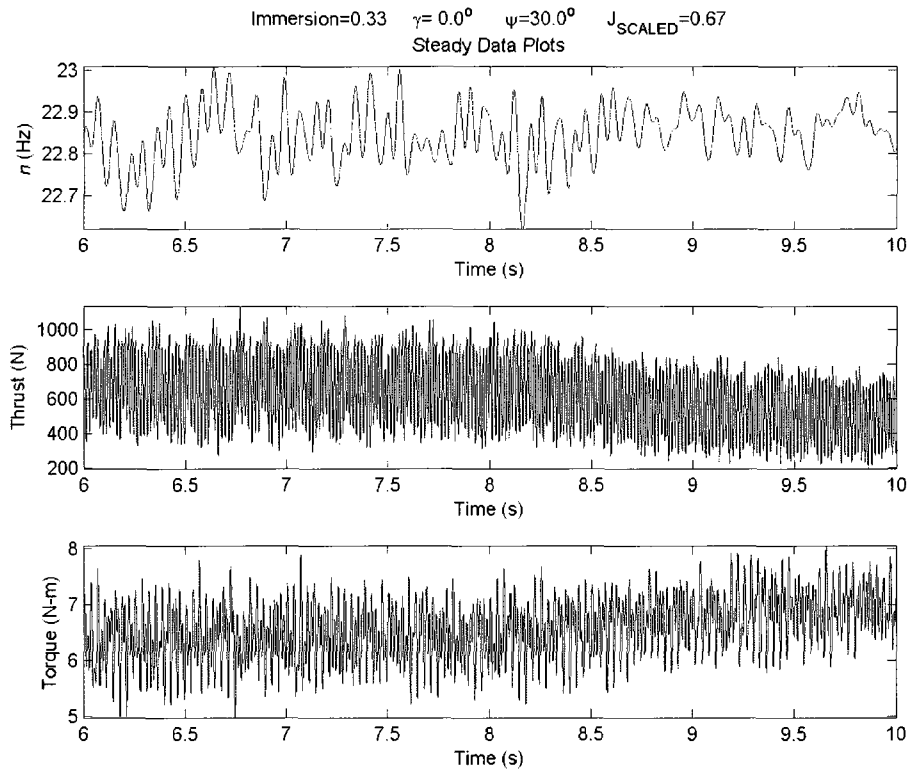


Figure E. 18

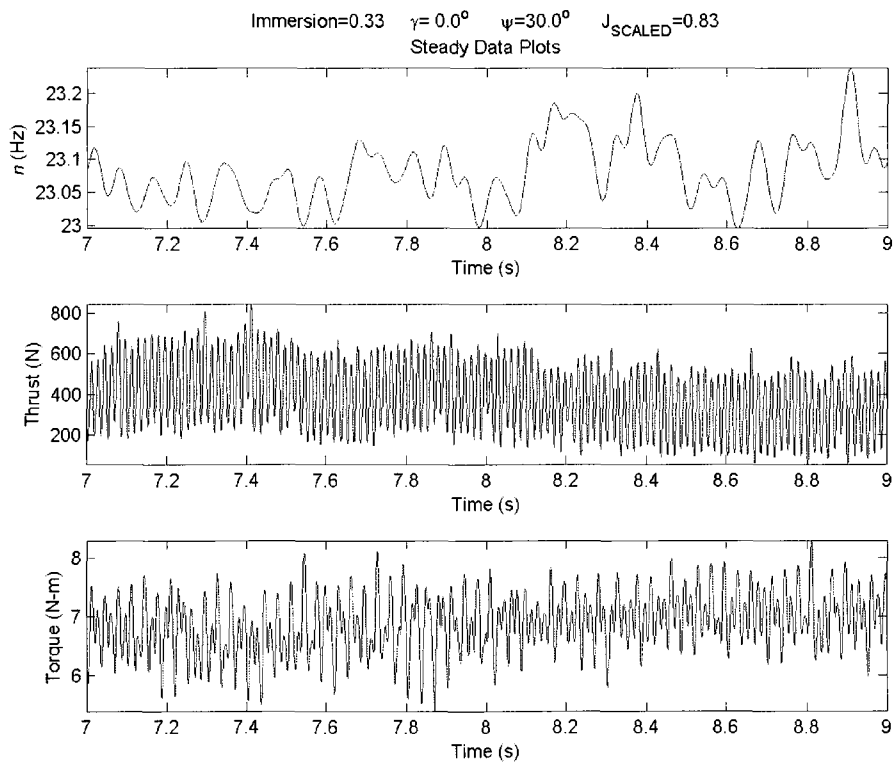


Figure E. 19

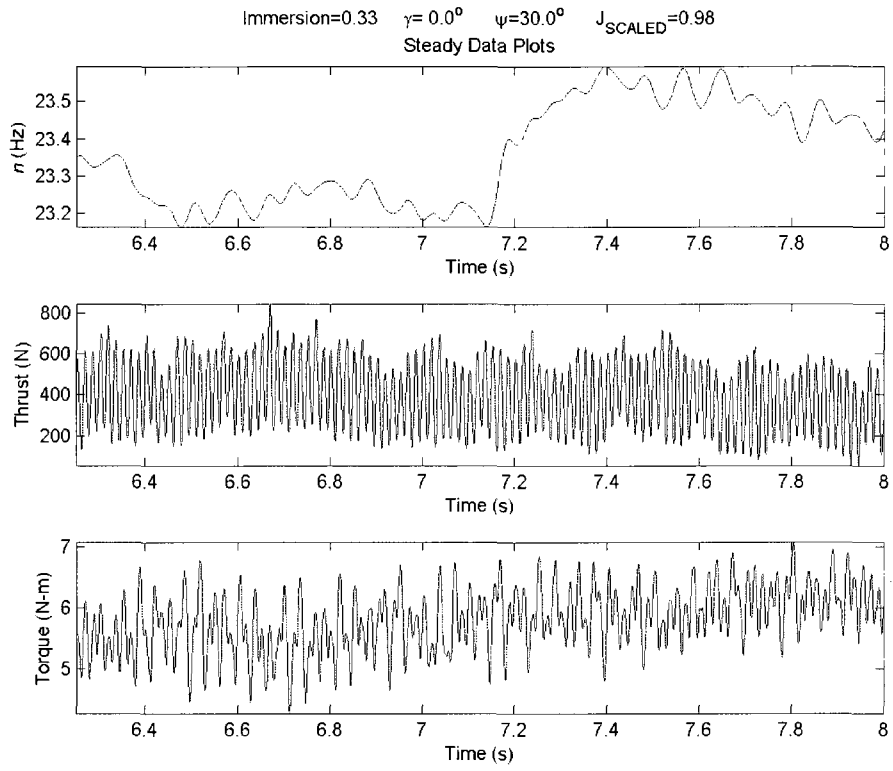


Figure E. 20

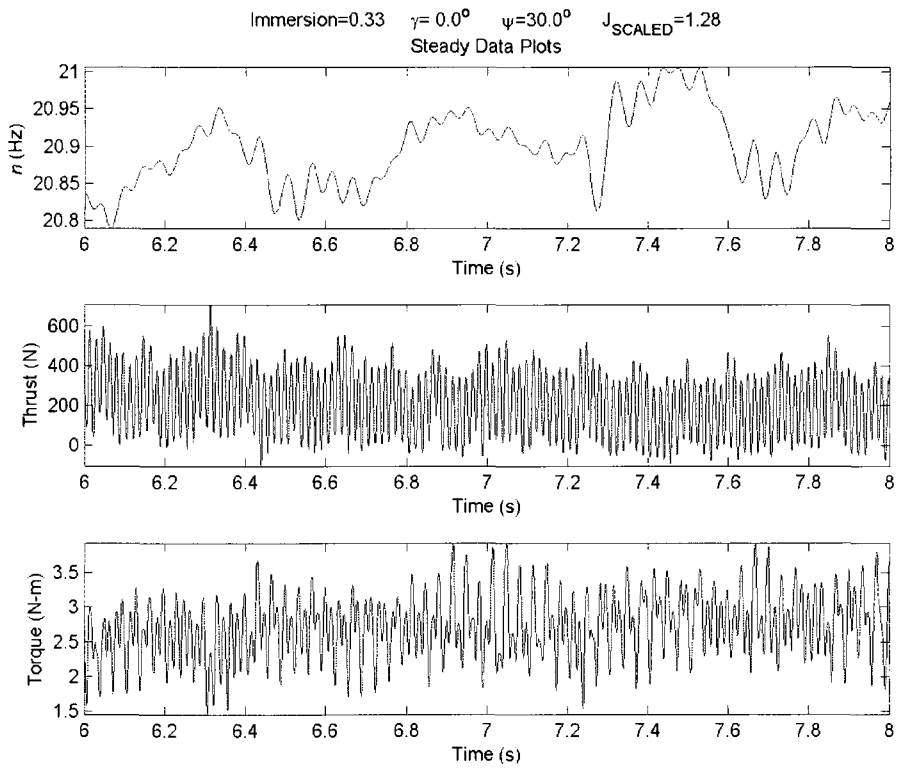


Figure E. 21

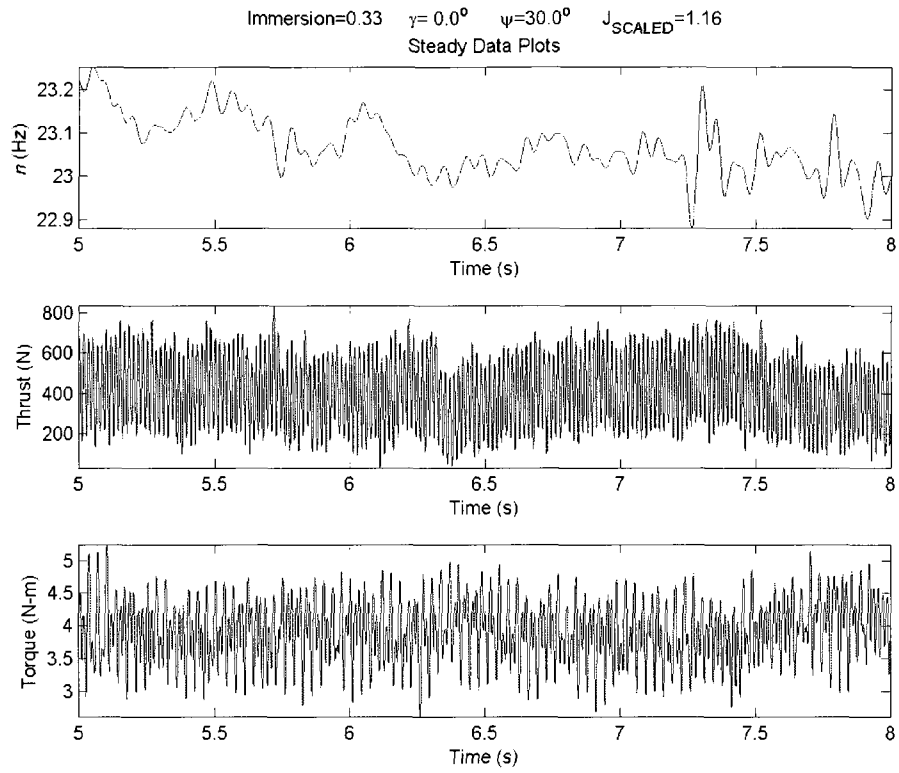


Figure E. 22

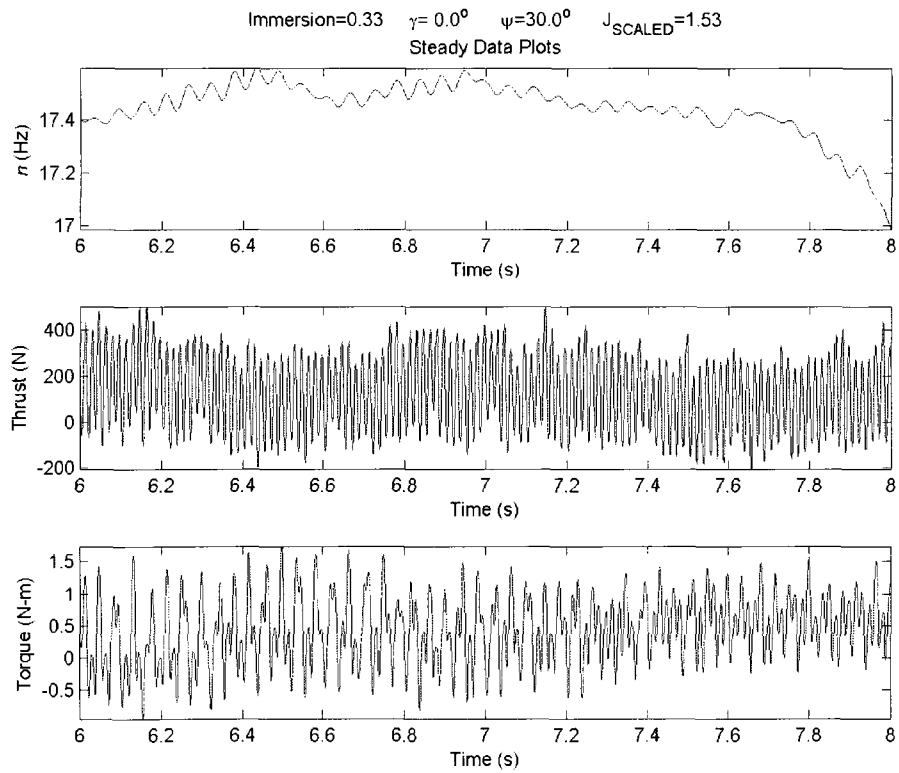


Figure E. 23

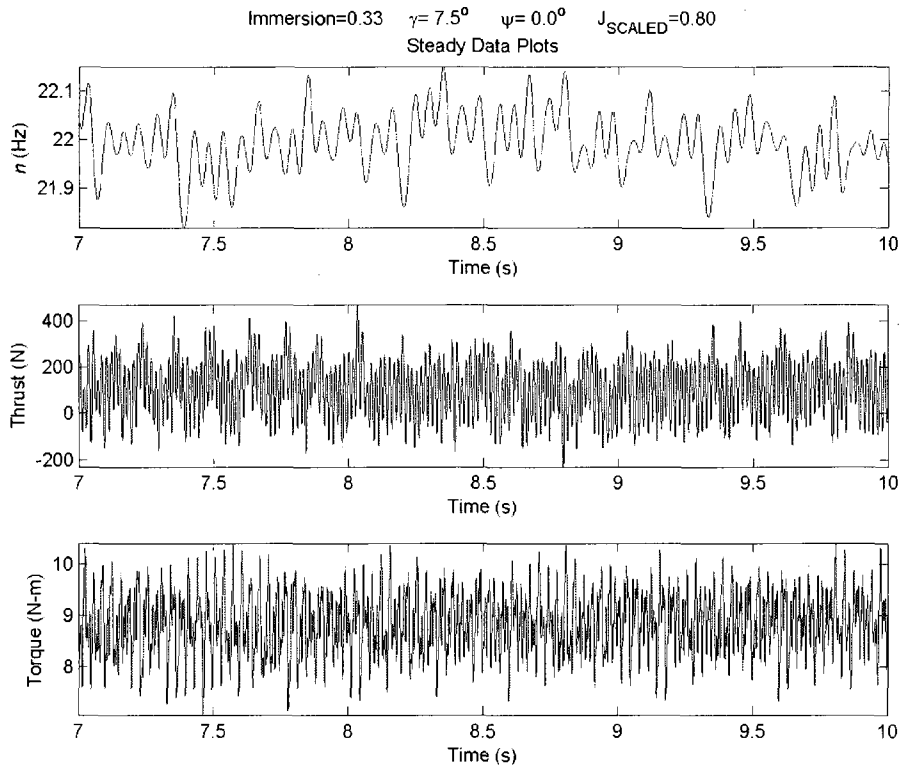


Figure E. 24

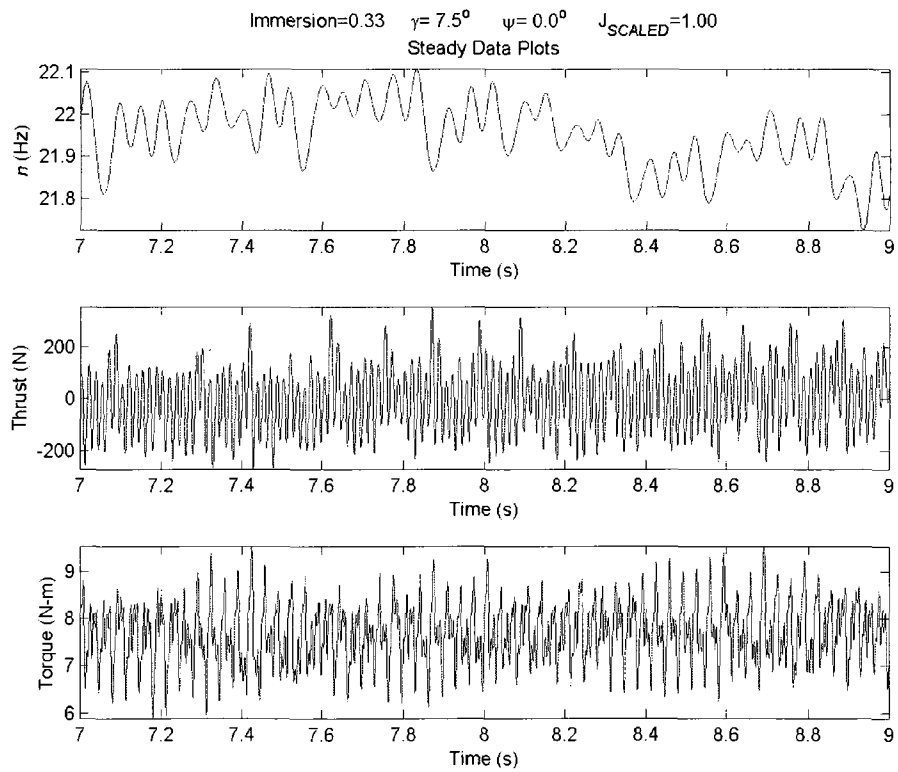


Figure E. 25

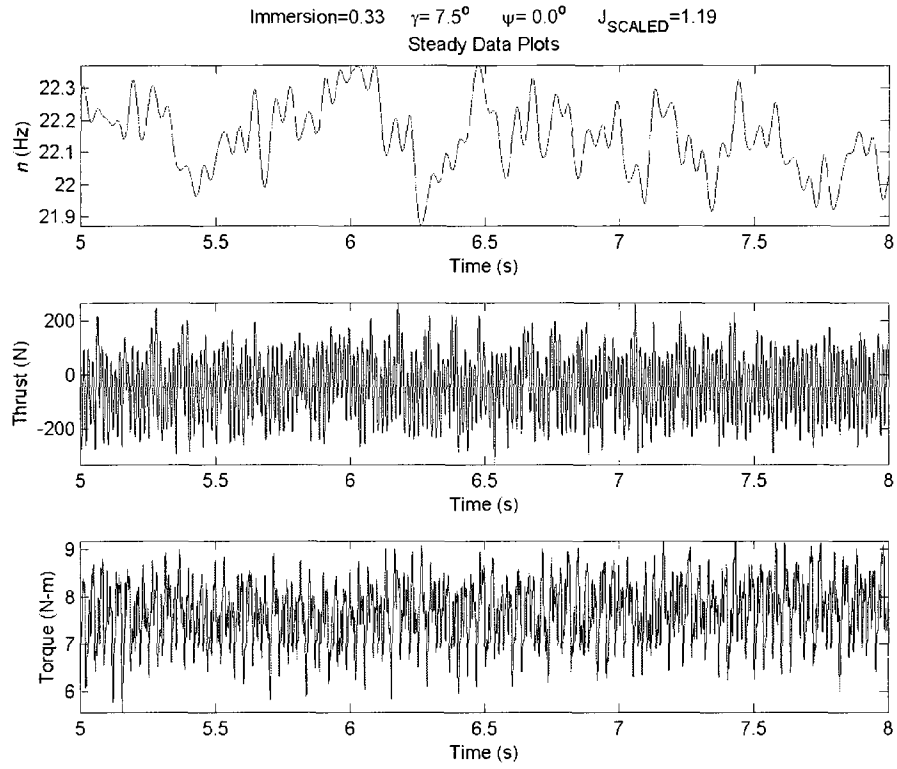


Figure E. 26

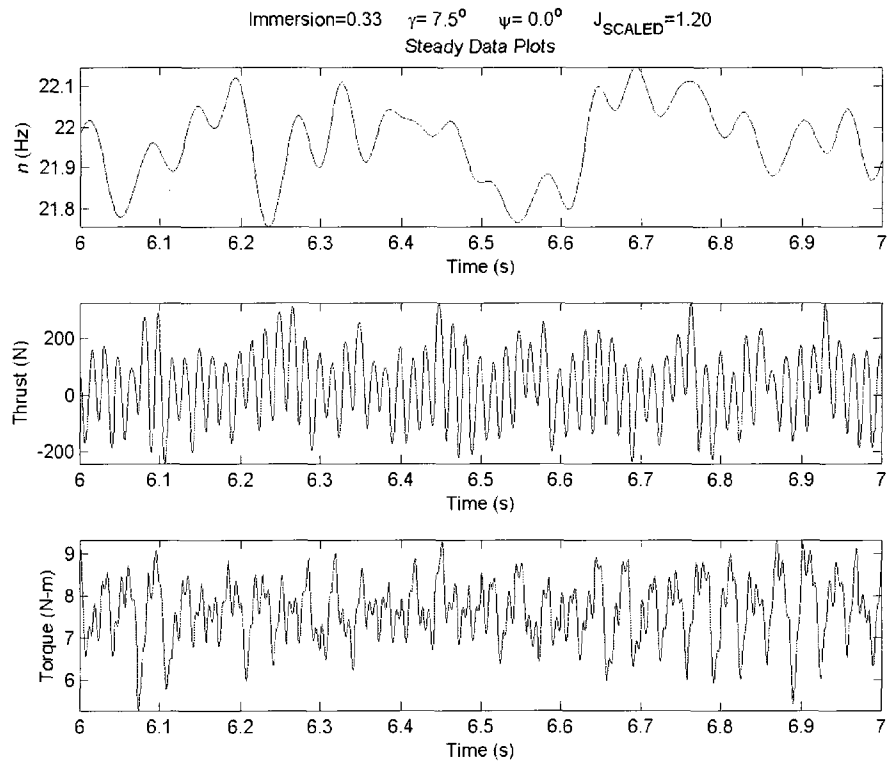


Figure E. 27

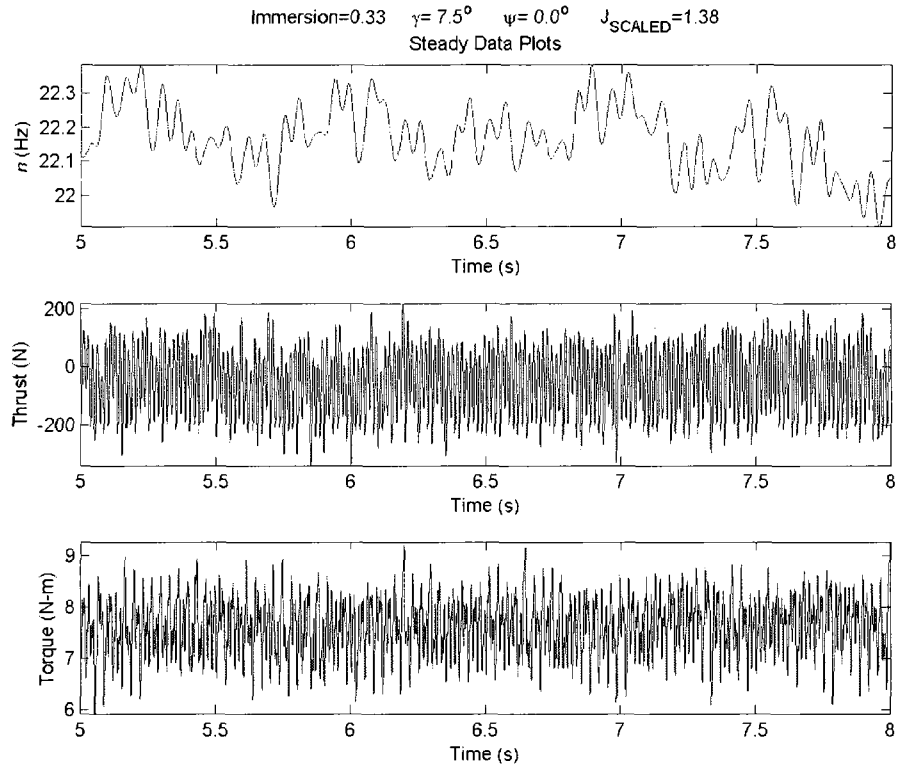


Figure E. 28

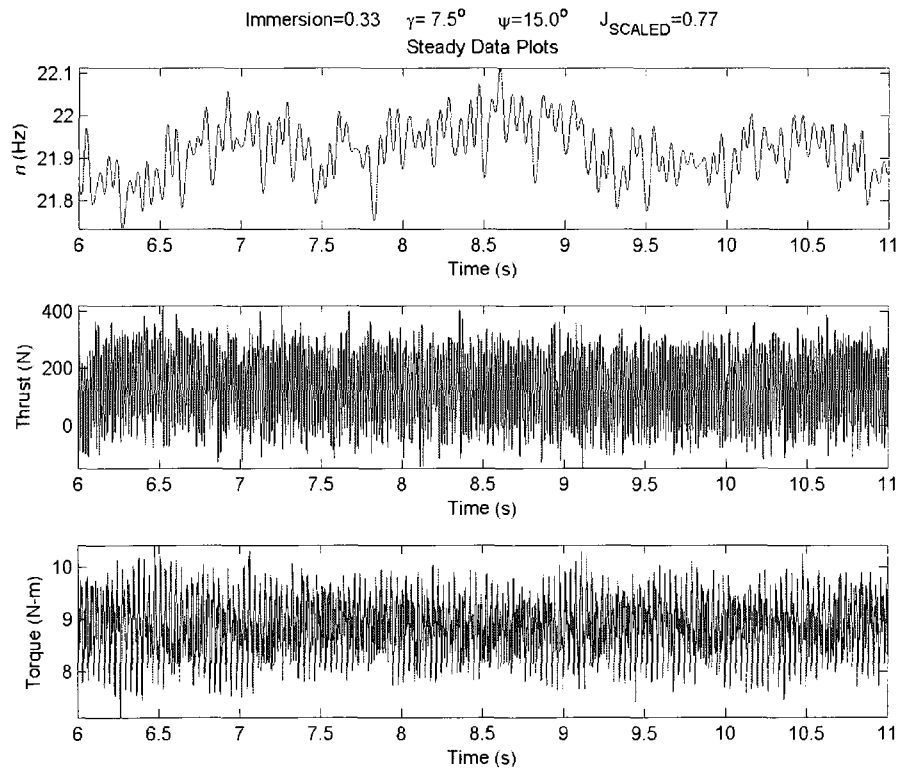


Figure E. 29

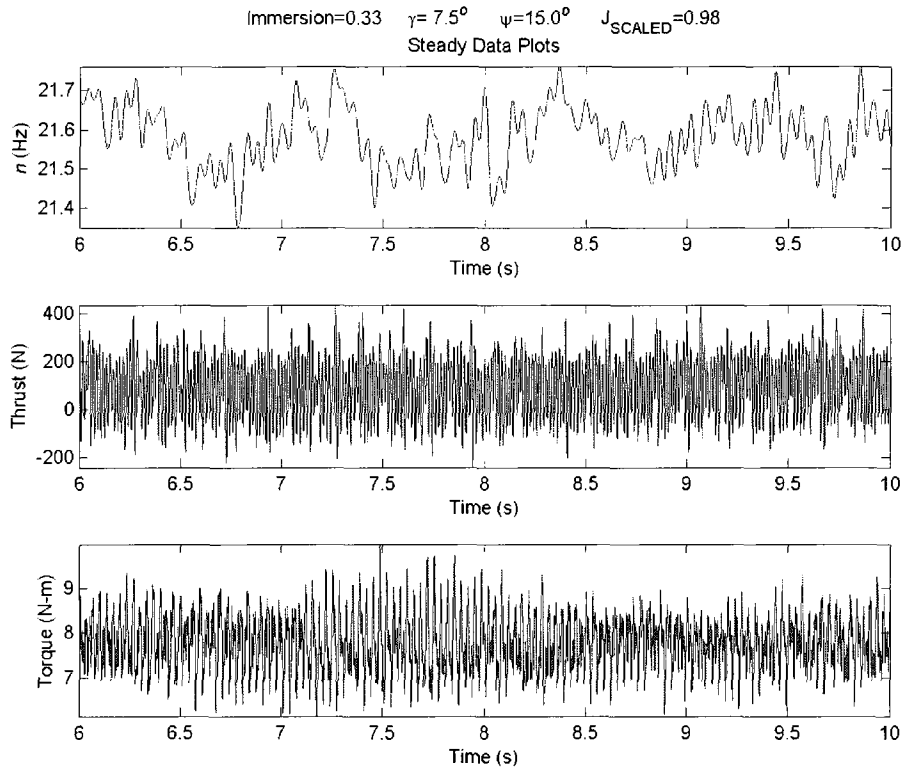


Figure E. 30

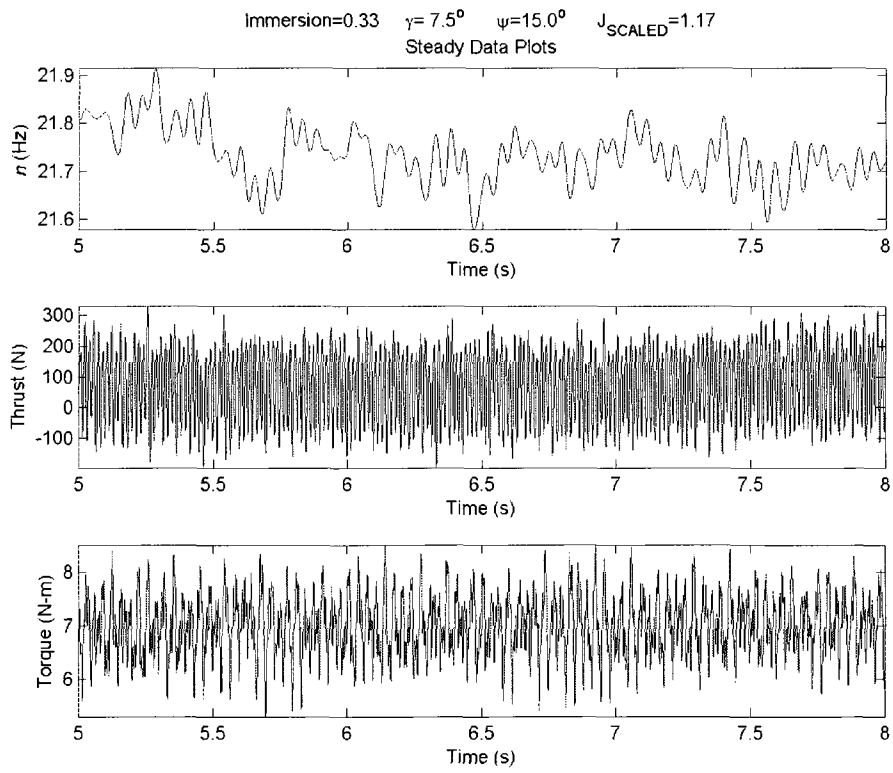


Figure E. 31

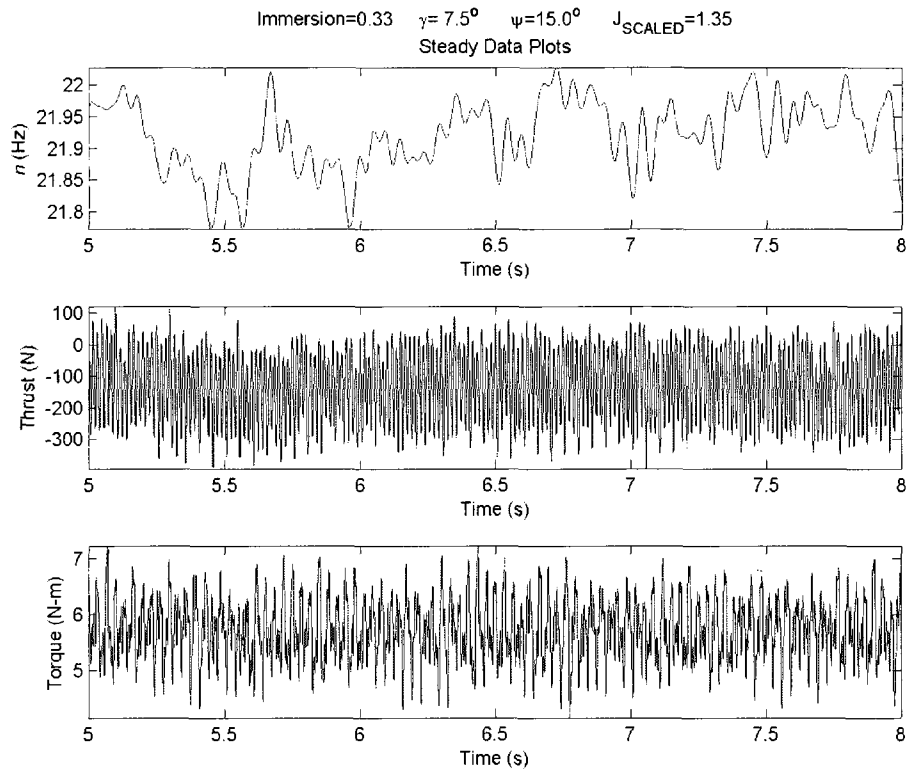


Figure E. 32

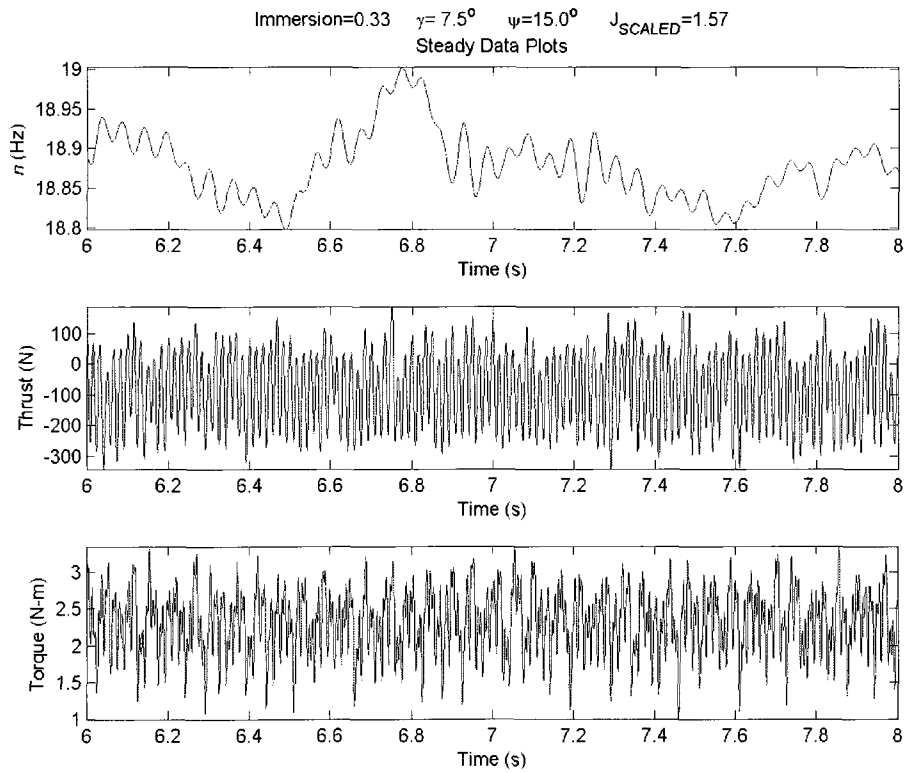


Figure E. 33

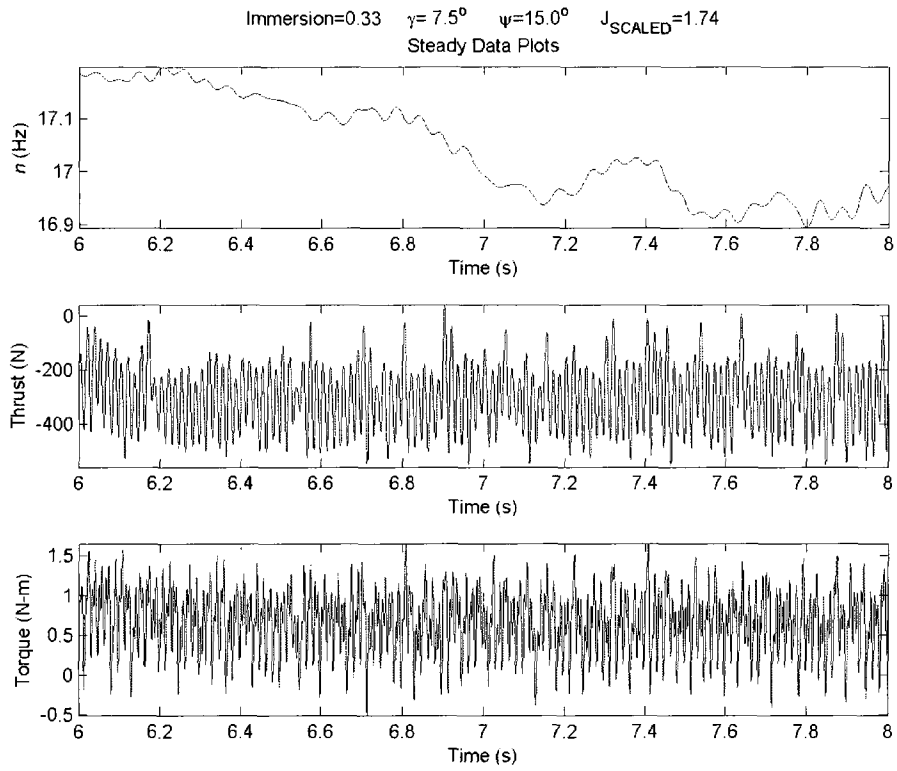


Figure E. 34

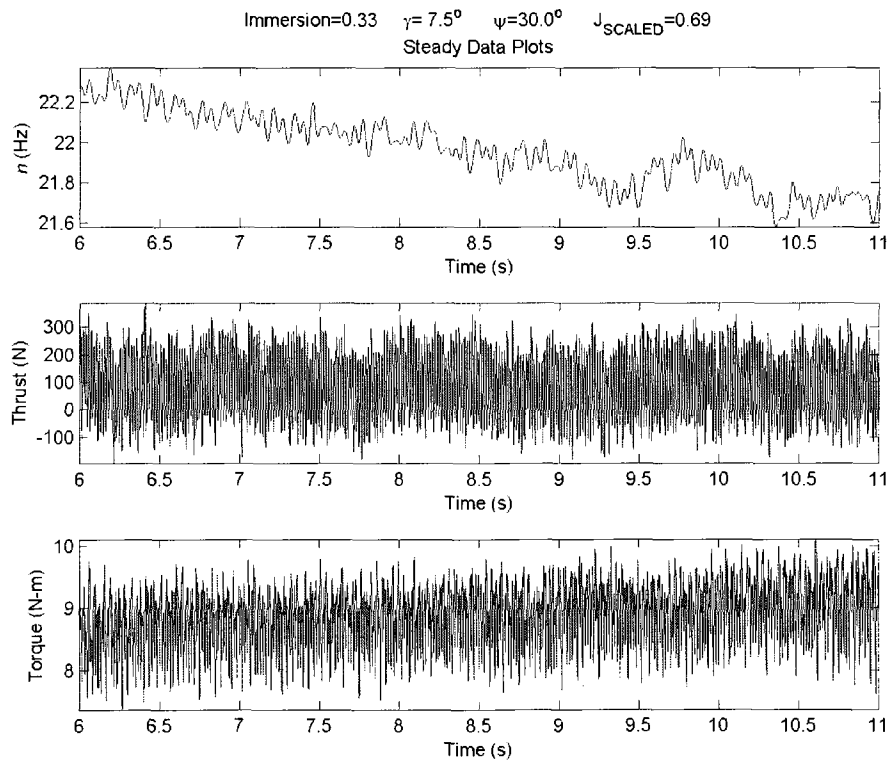


Figure E. 35

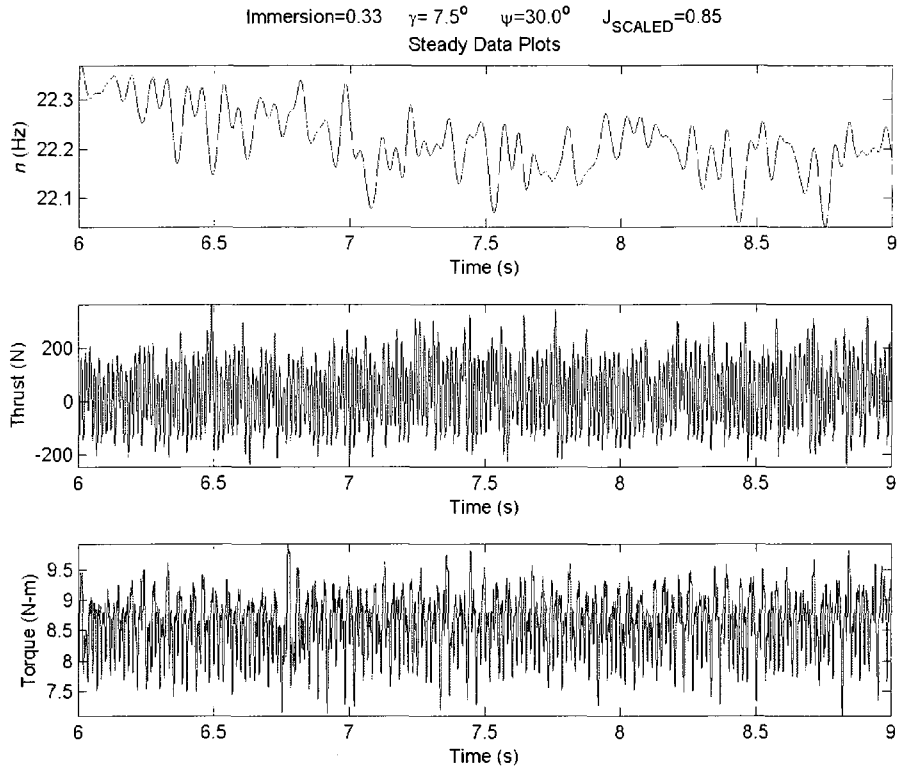


Figure E. 36

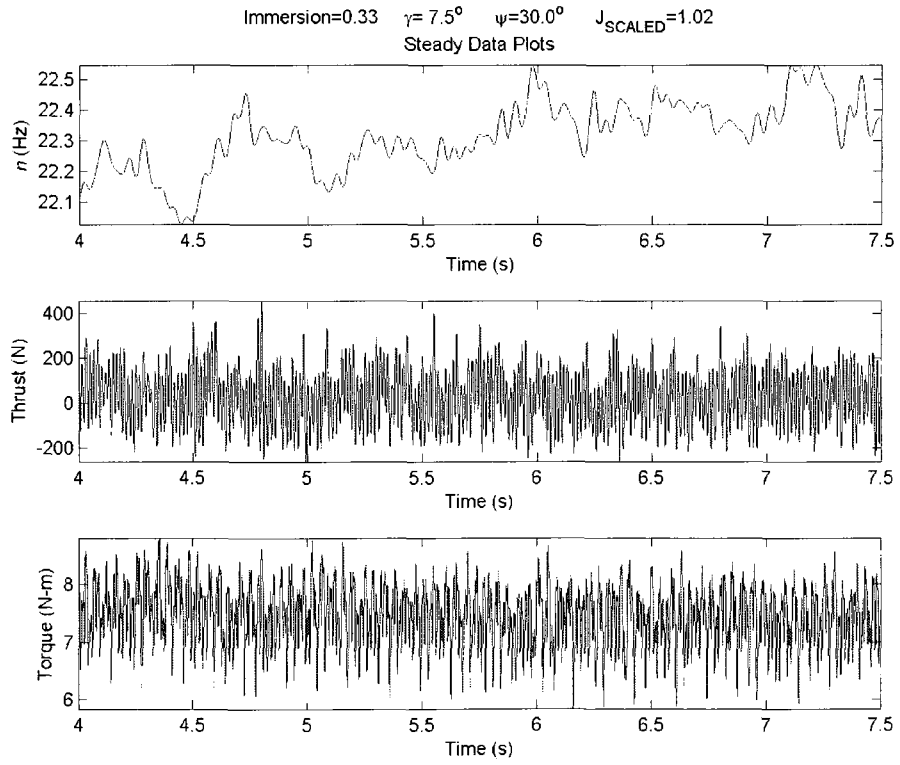


Figure E. 37

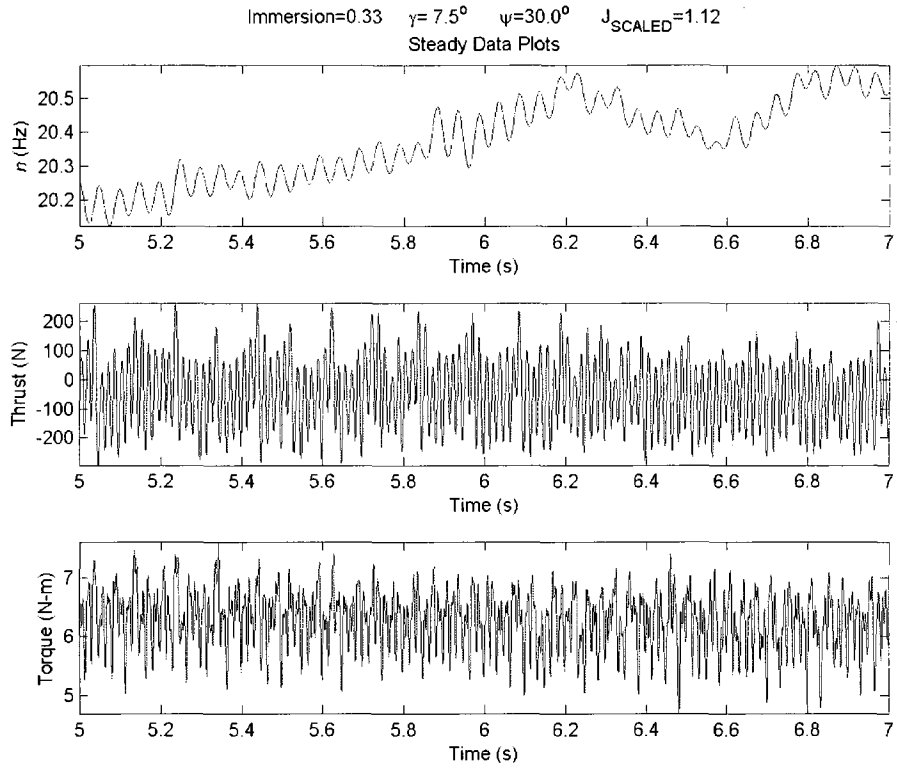


Figure E. 38

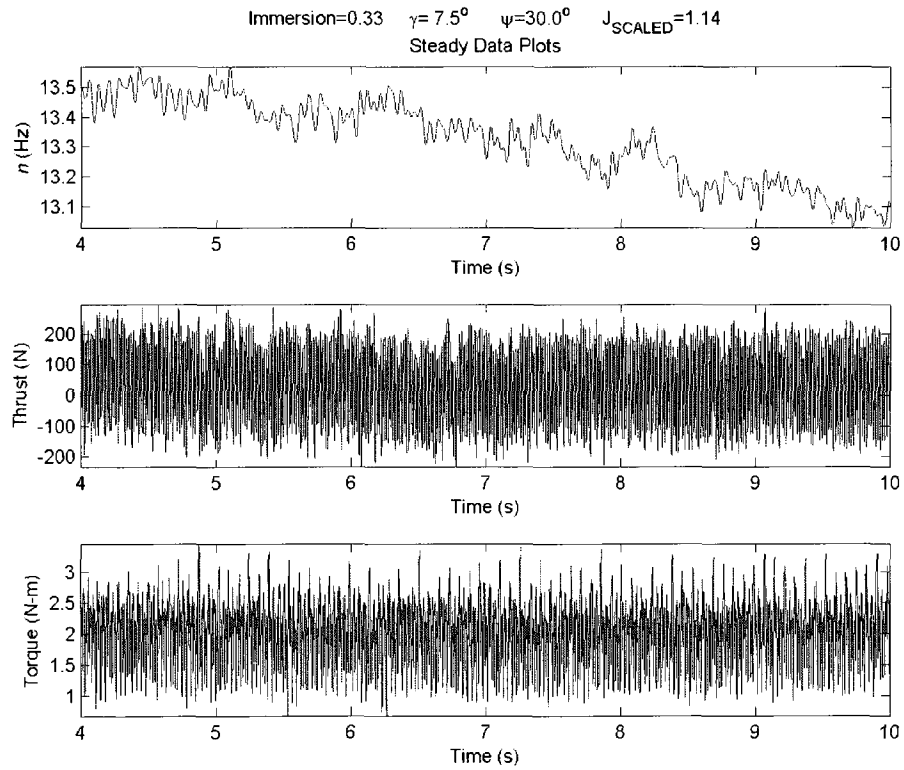


Figure E. 39

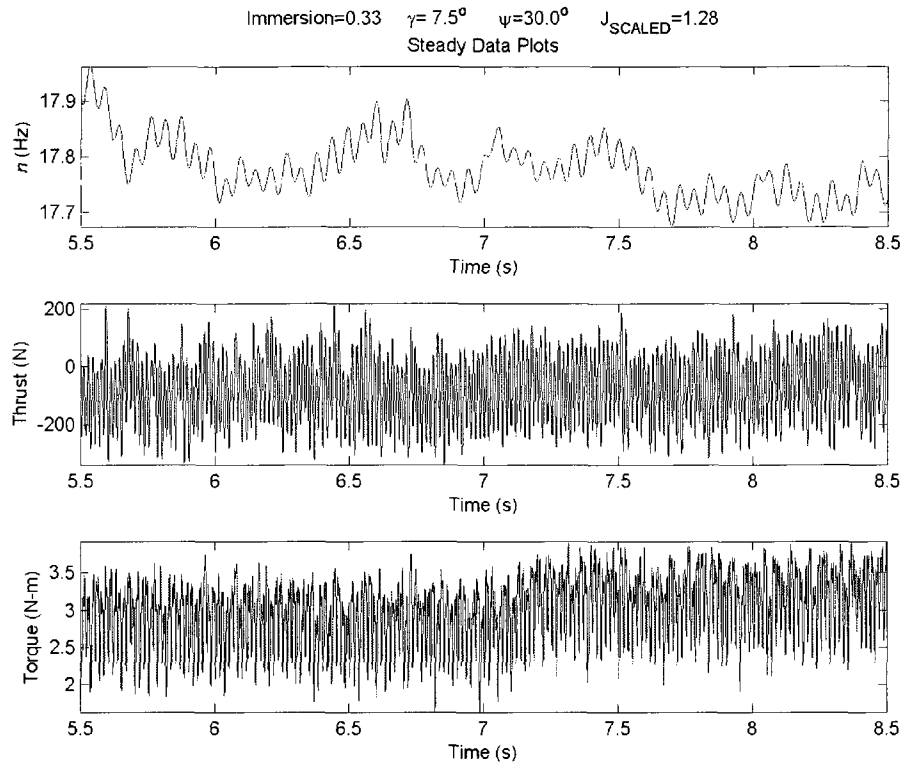


Figure E. 40

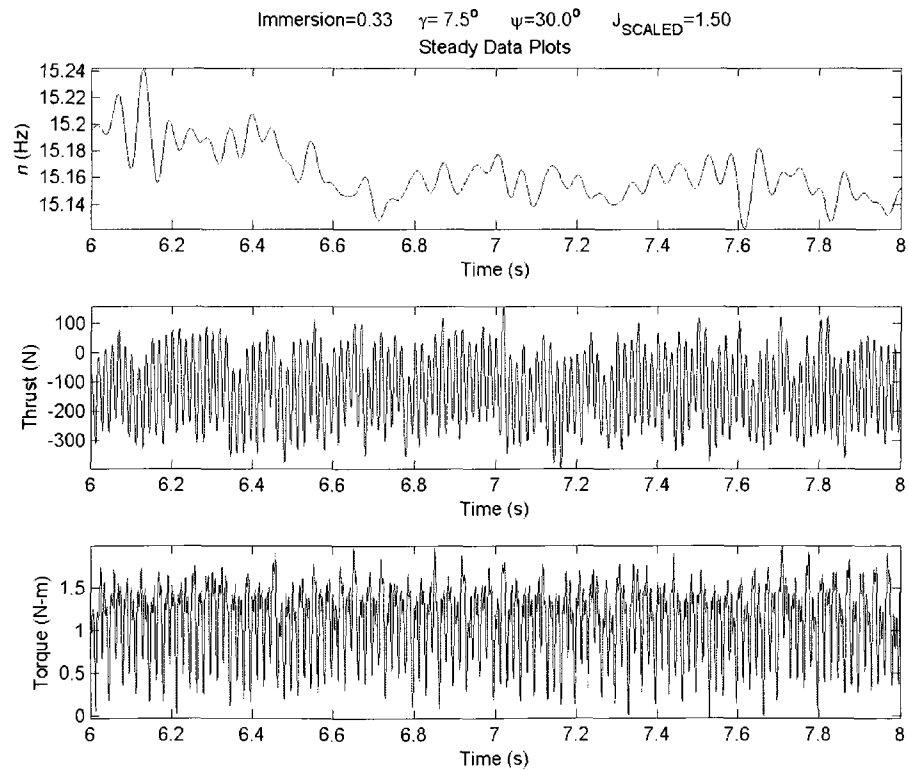


Figure E. 41

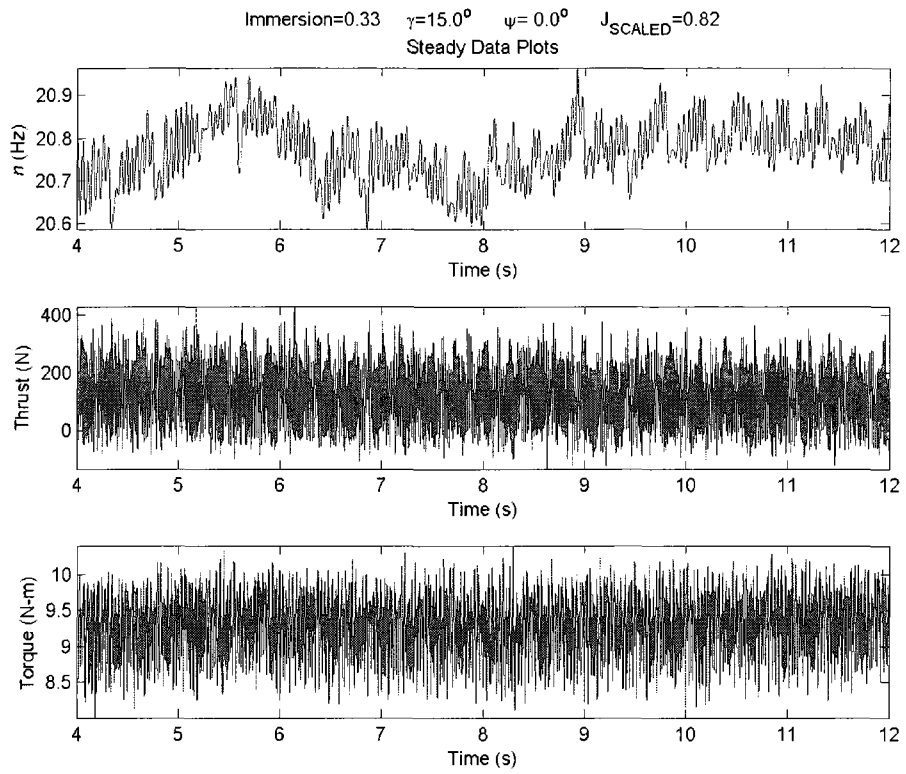


Figure E. 42

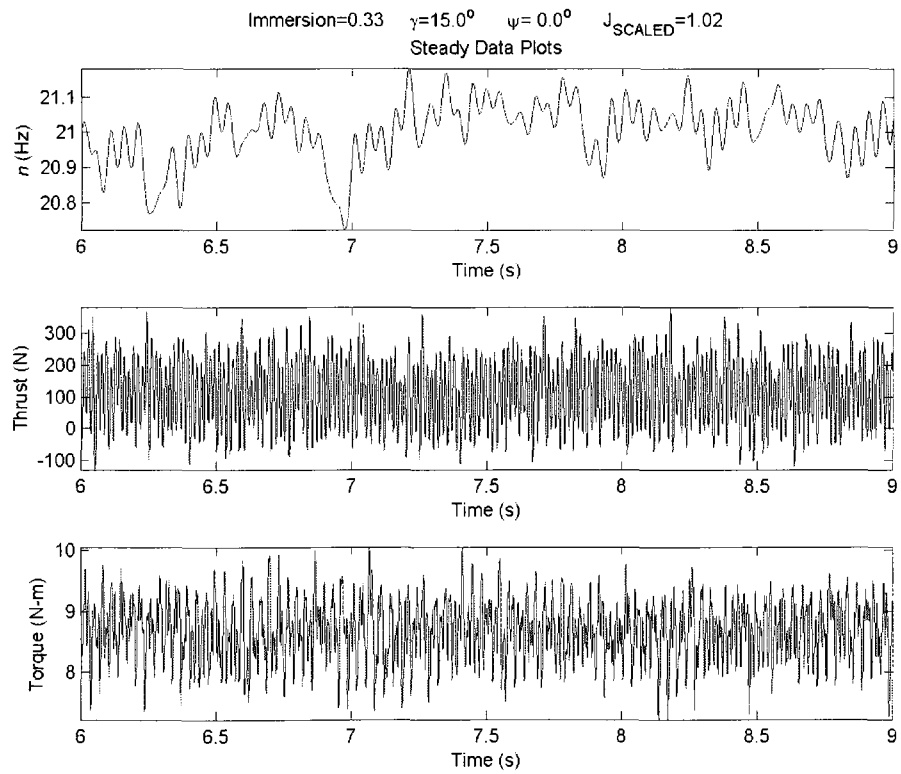


Figure E. 43

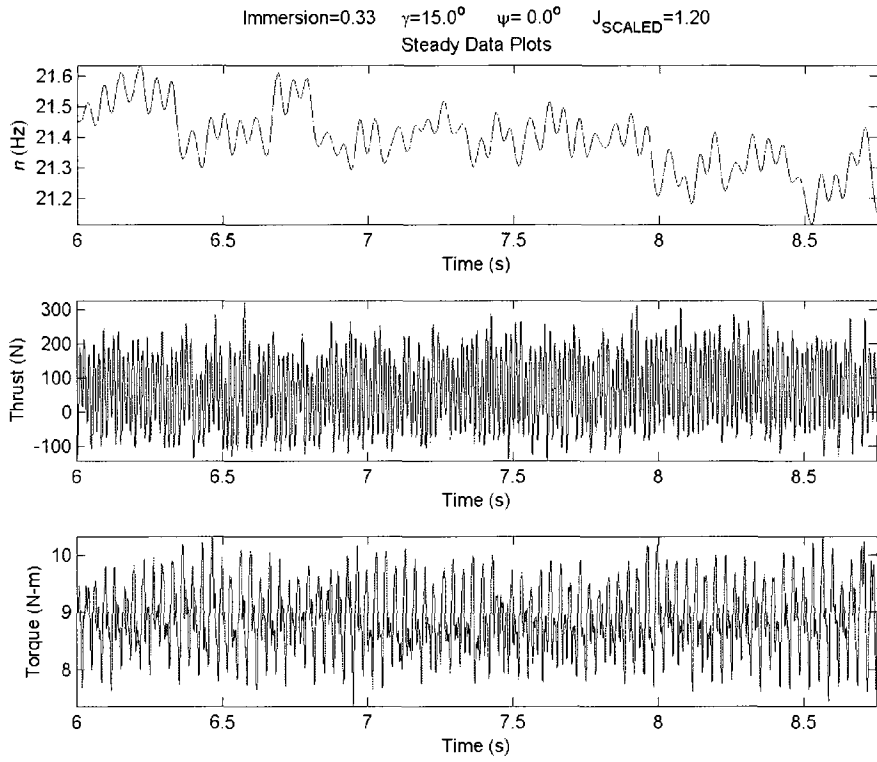


Figure E. 44

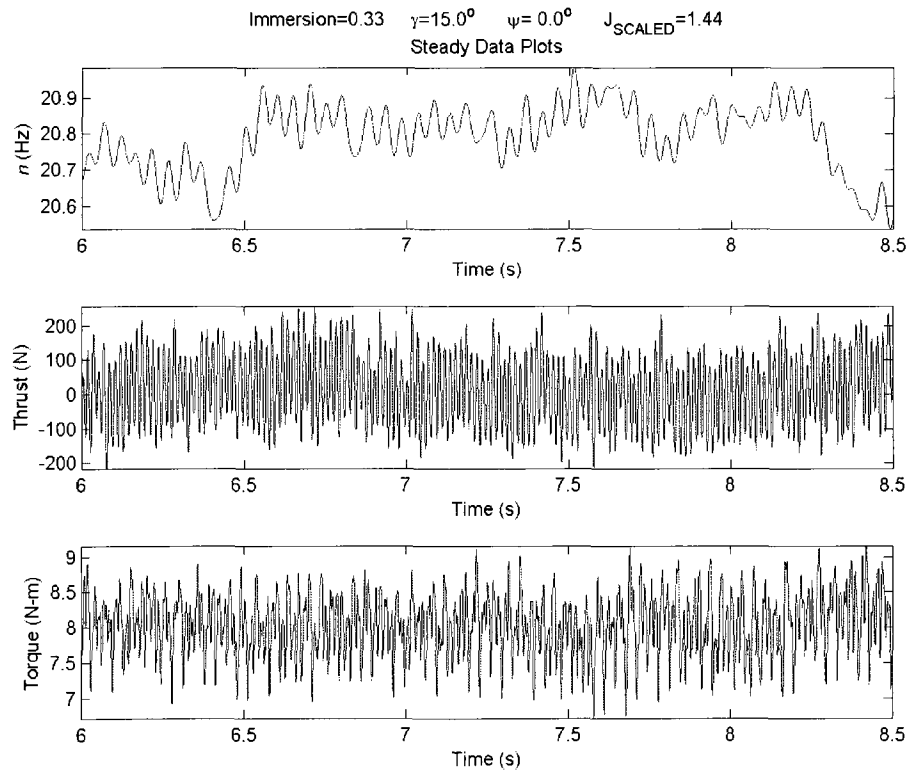


Figure E. 45

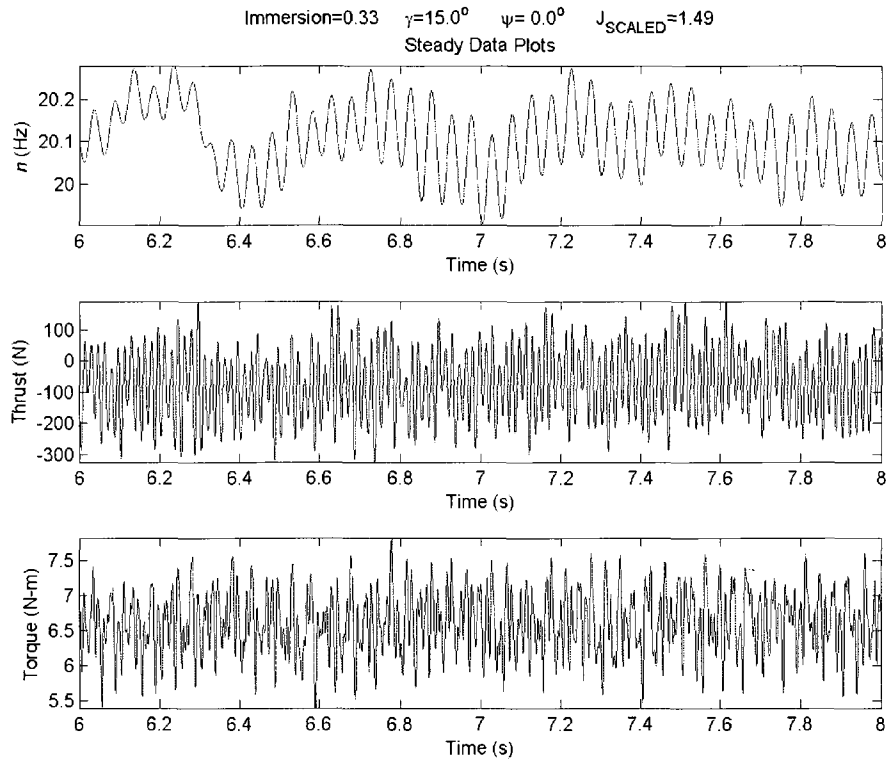


Figure E. 46

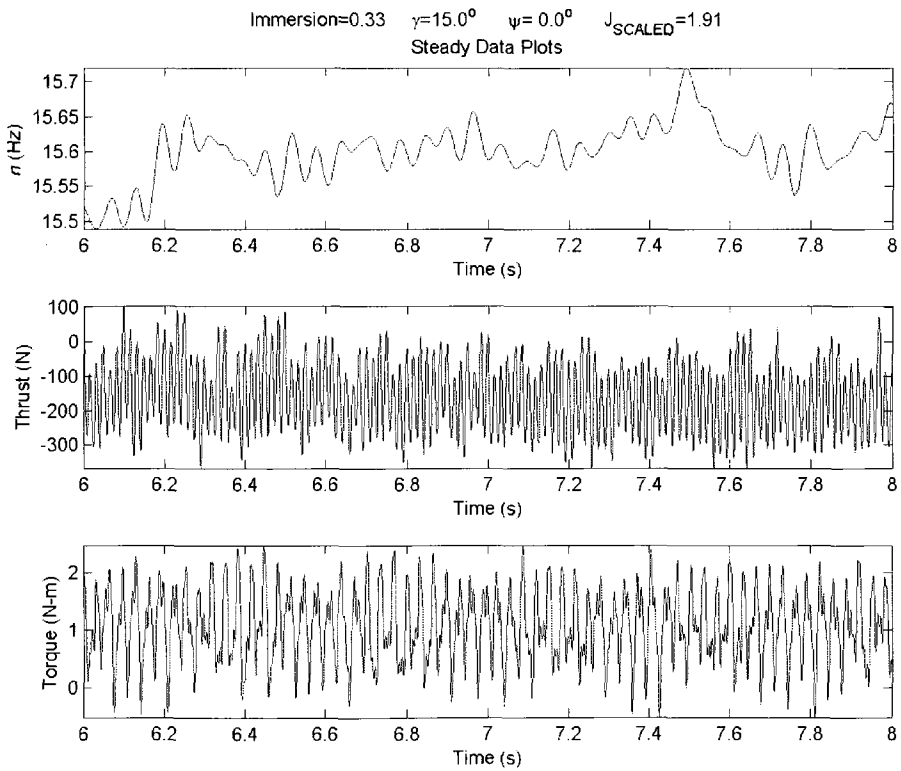


Figure E. 47

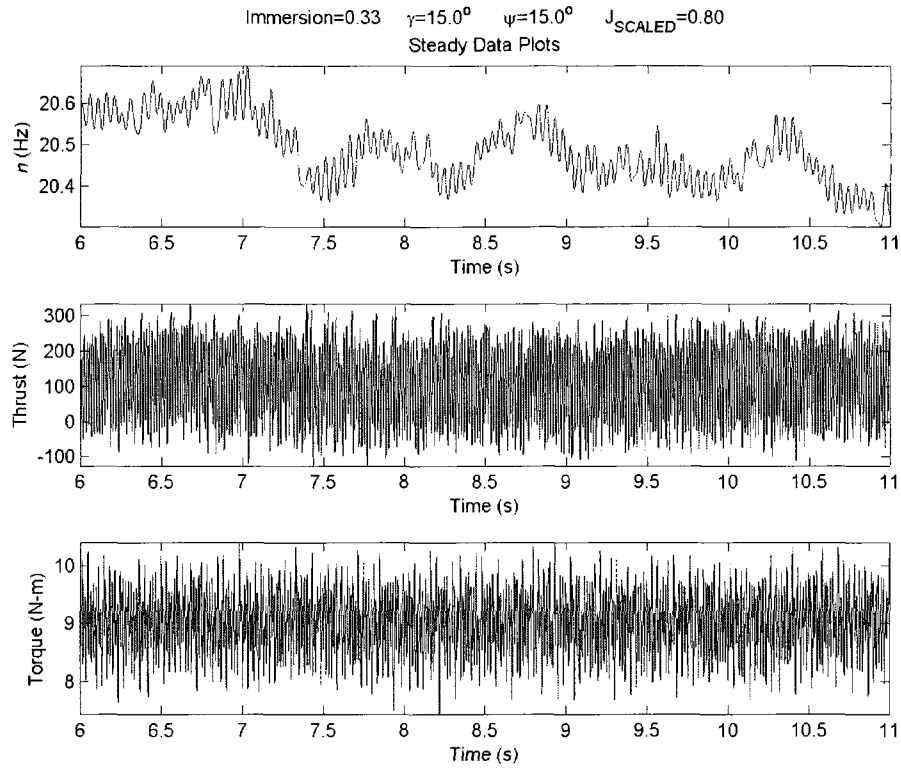


Figure E. 48

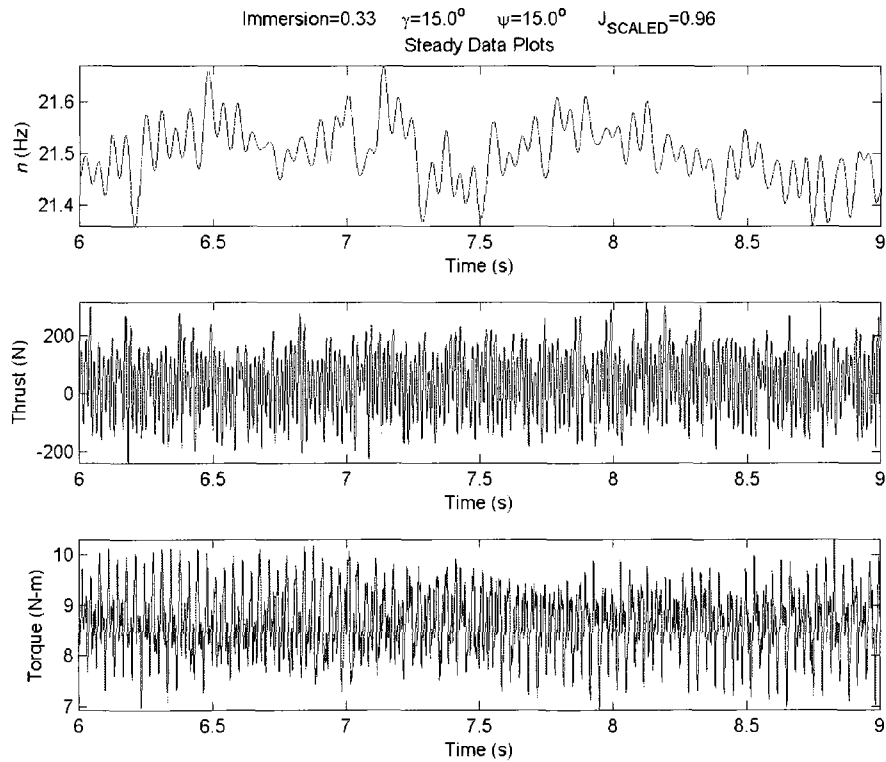


Figure E. 49

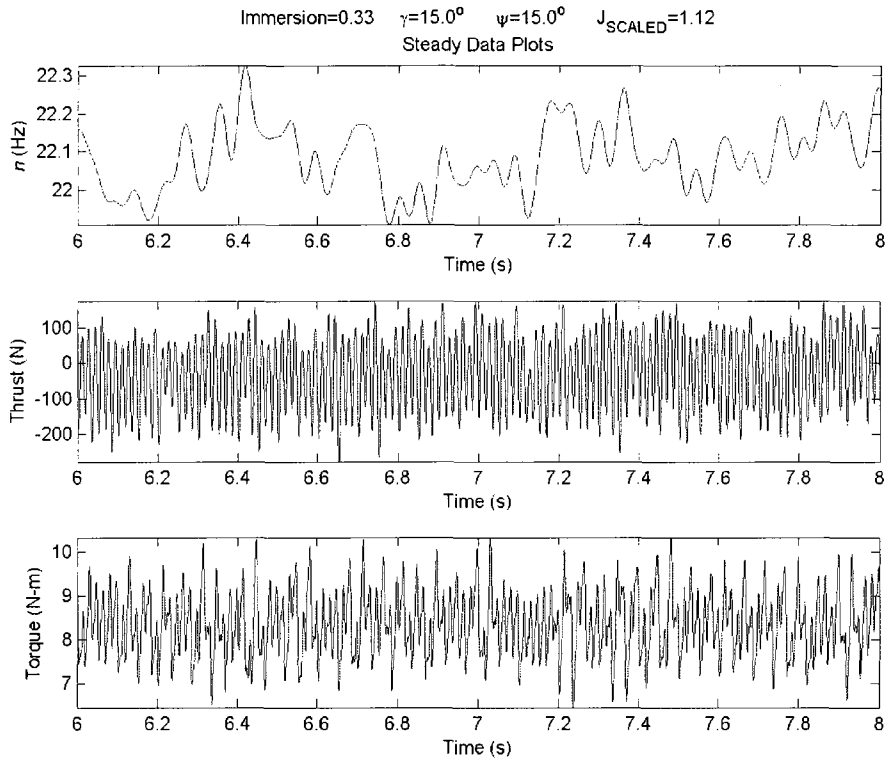


Figure E. 50

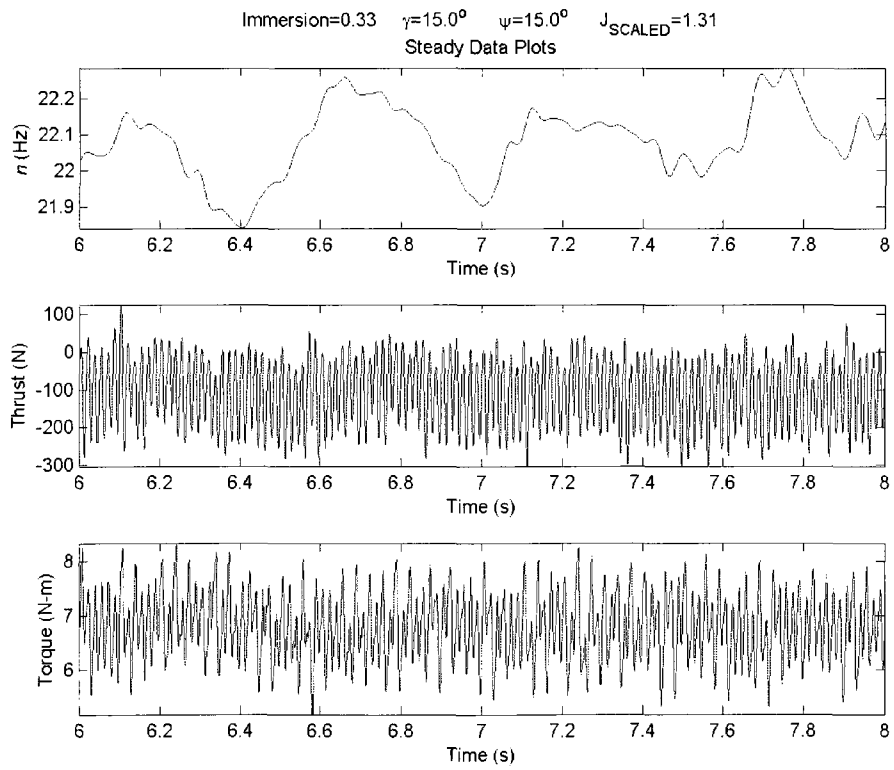


Figure E. 51

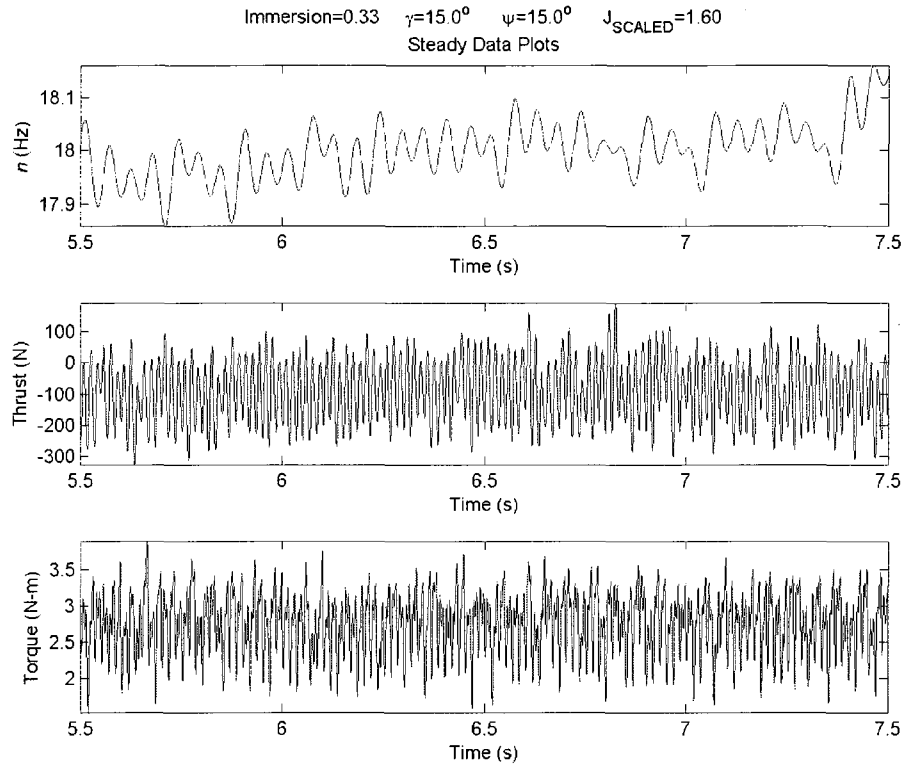


Figure E. 52

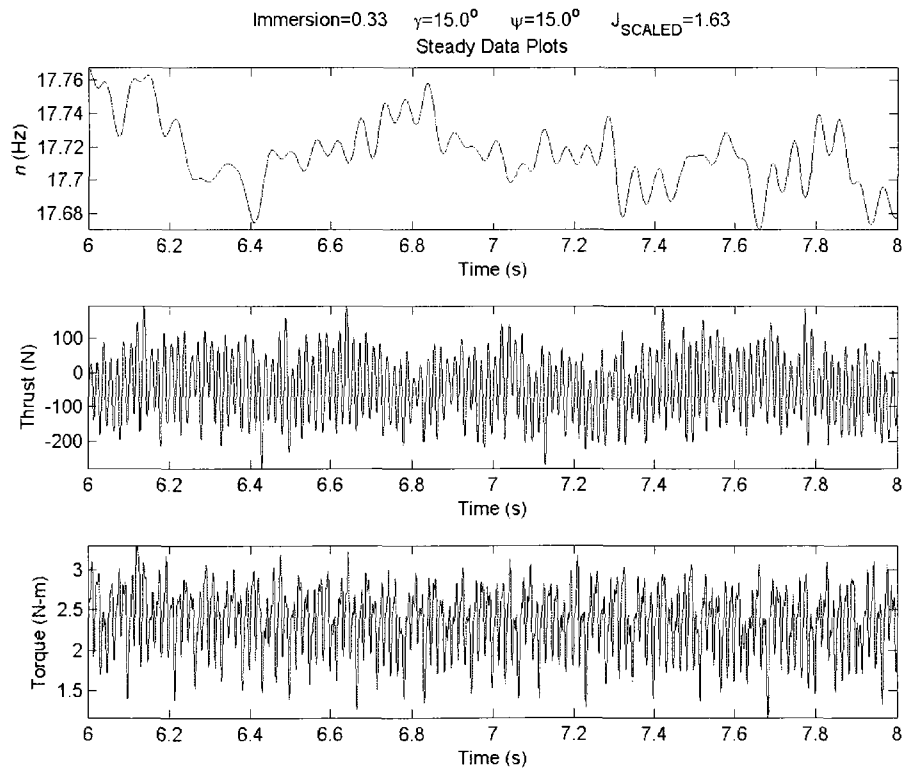


Figure E. 53

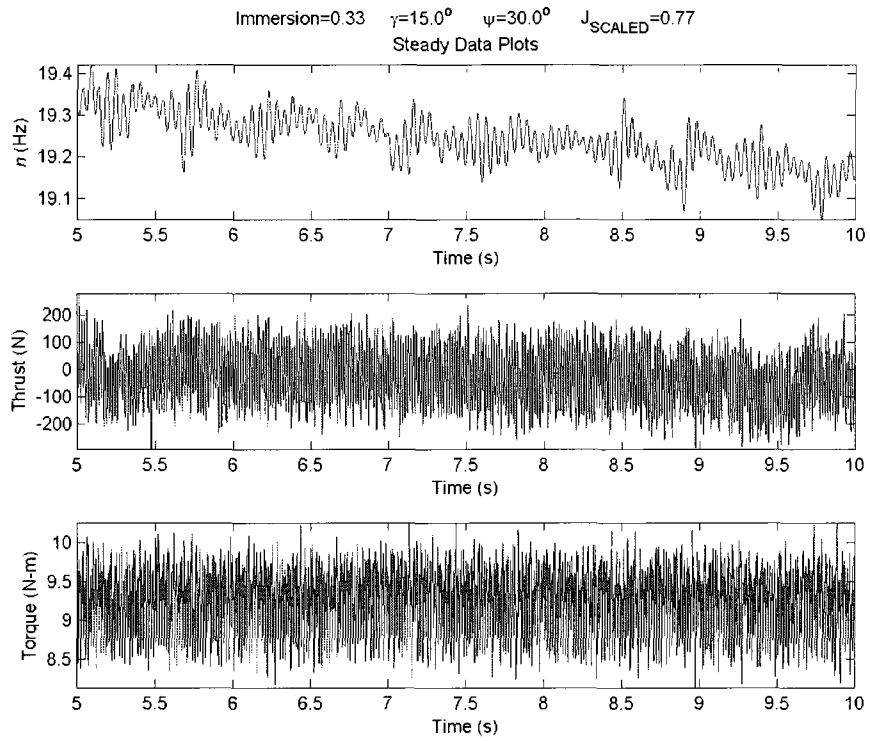


Figure E. 54

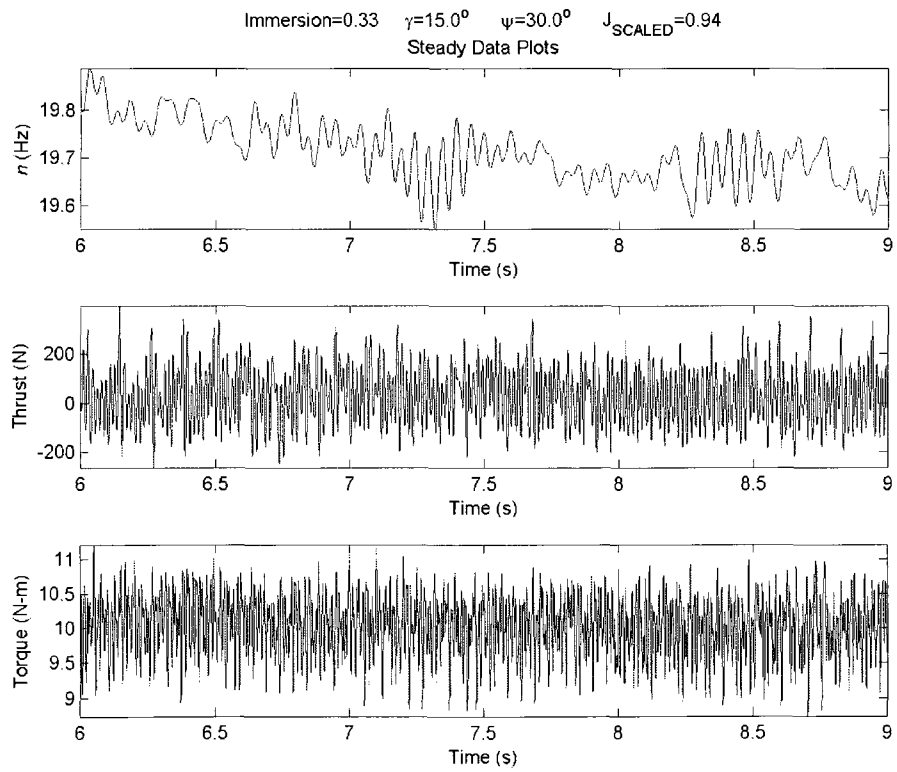


Figure E. 55

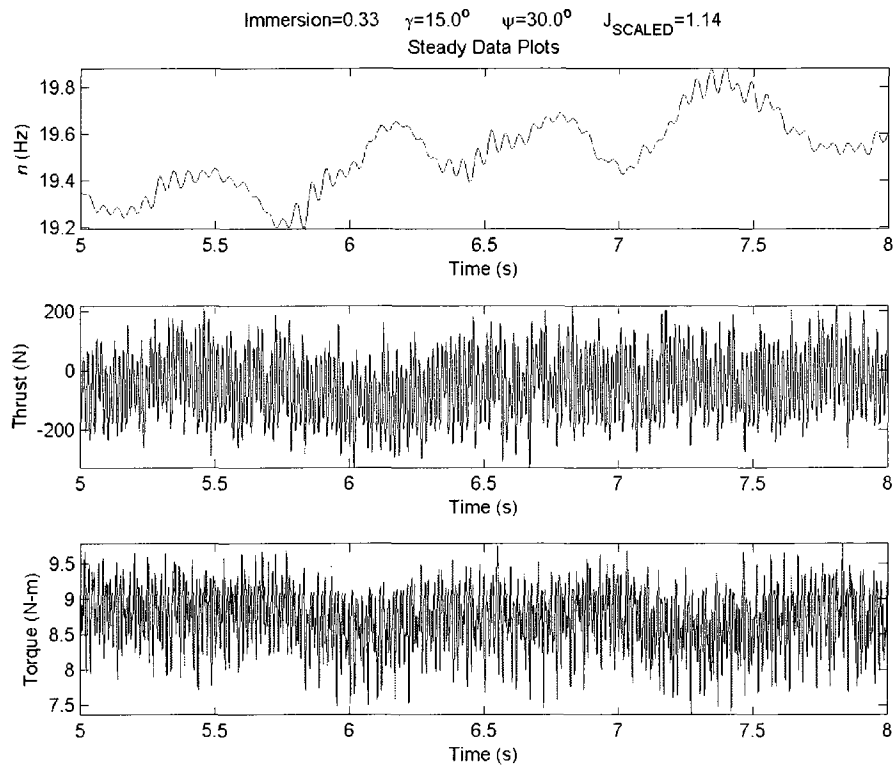


Figure E. 56

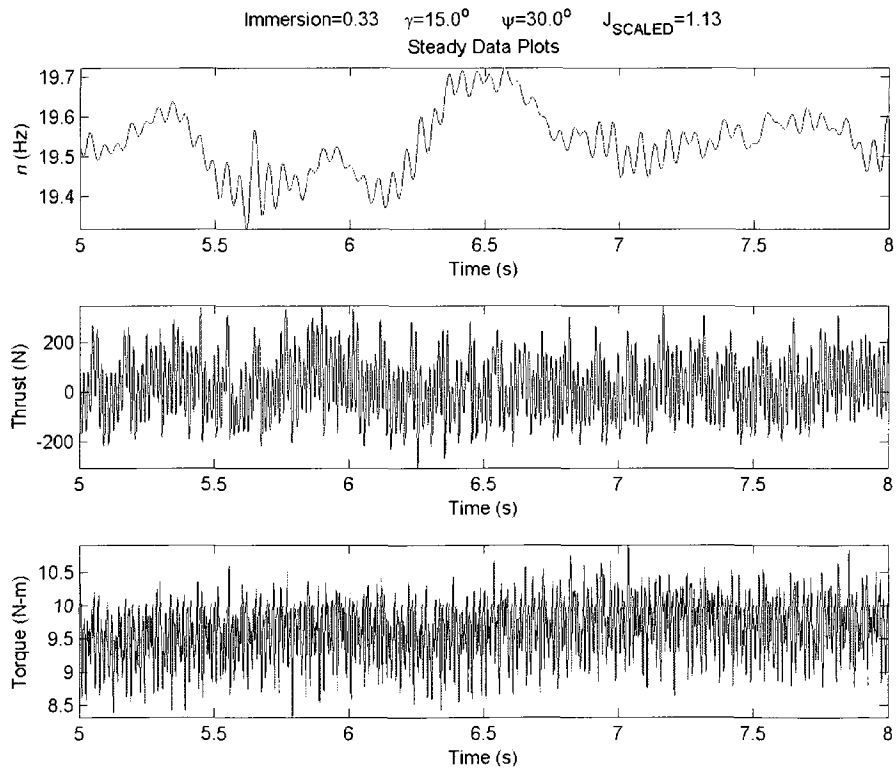


Figure E. 57

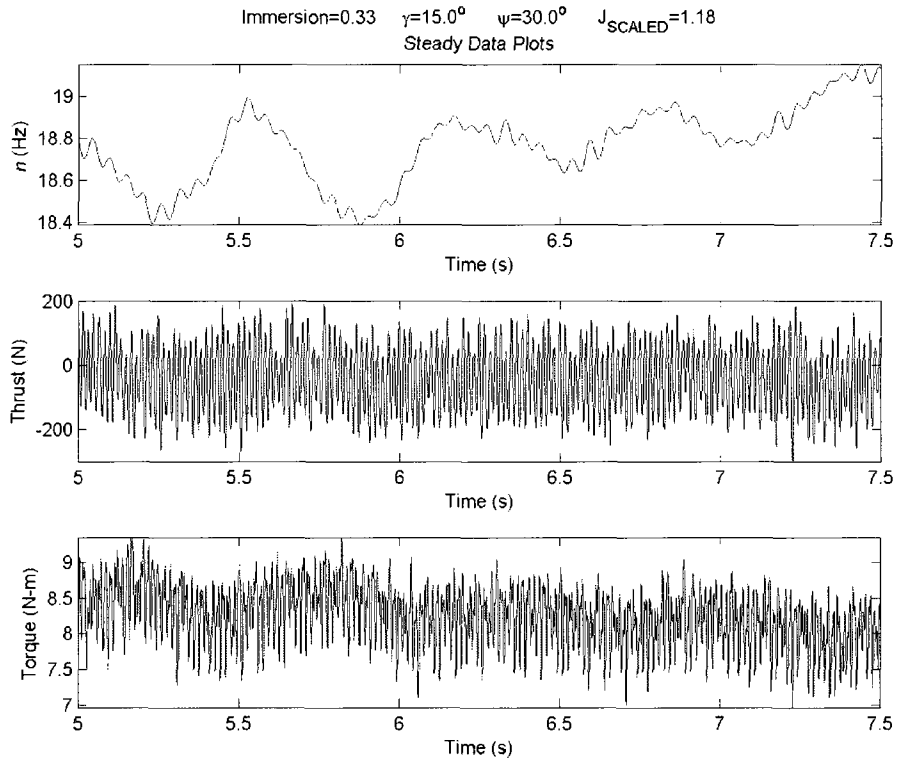


Figure E. 58

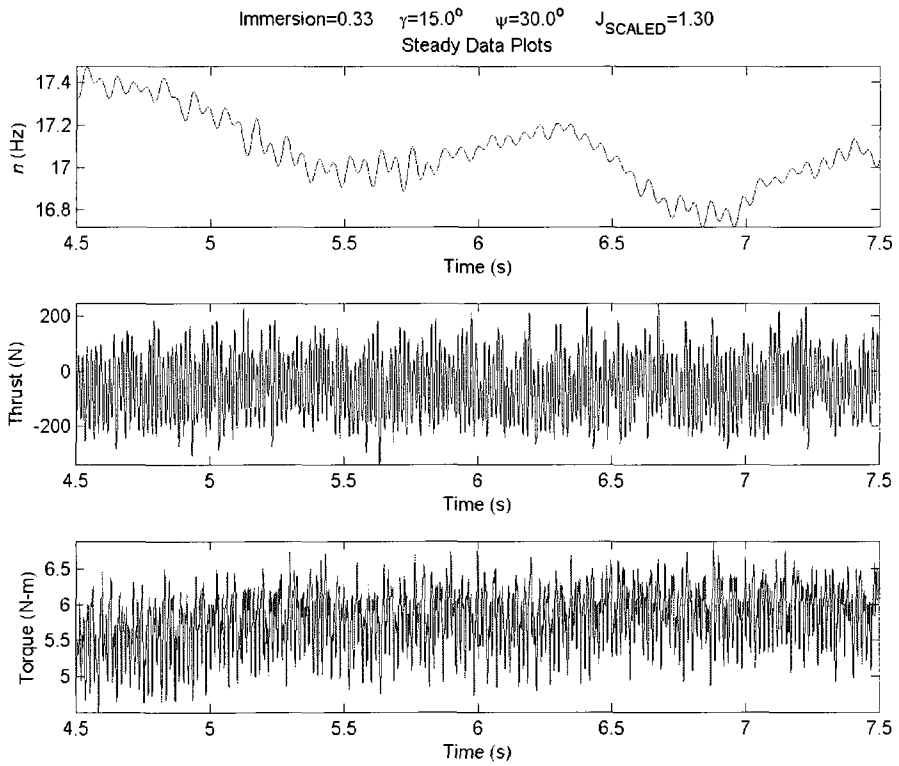


Figure E. 59

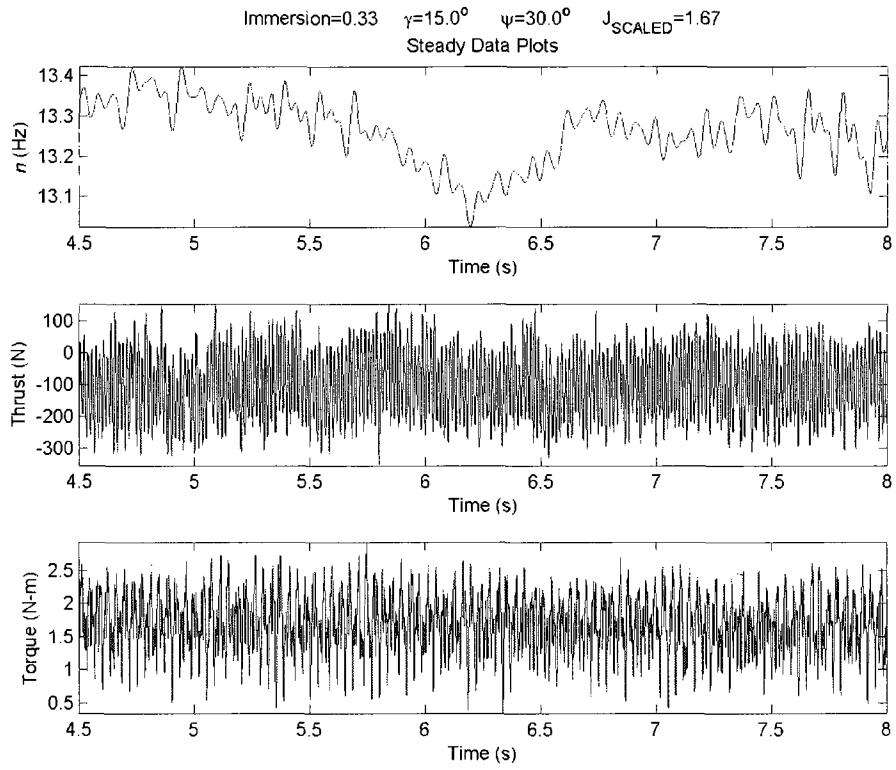


Figure E. 60

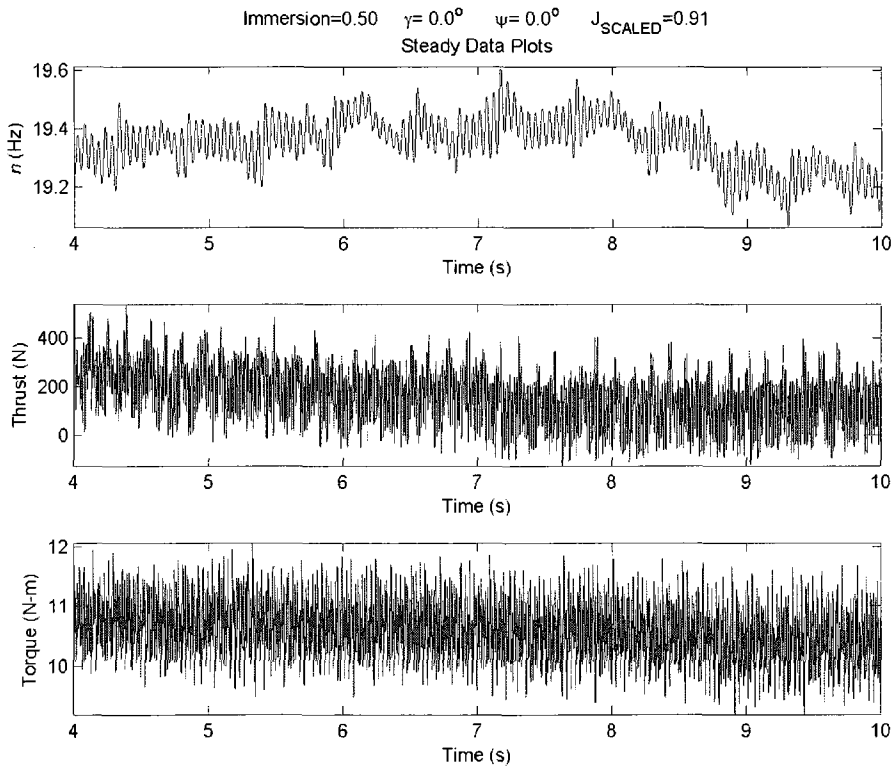


Figure E. 61

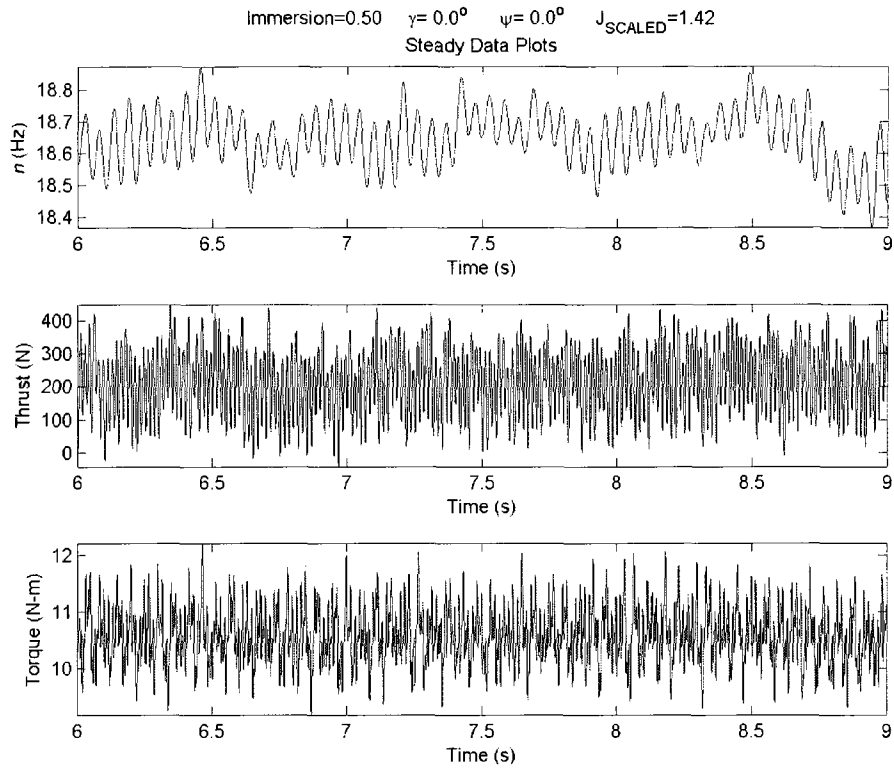


Figure E. 62

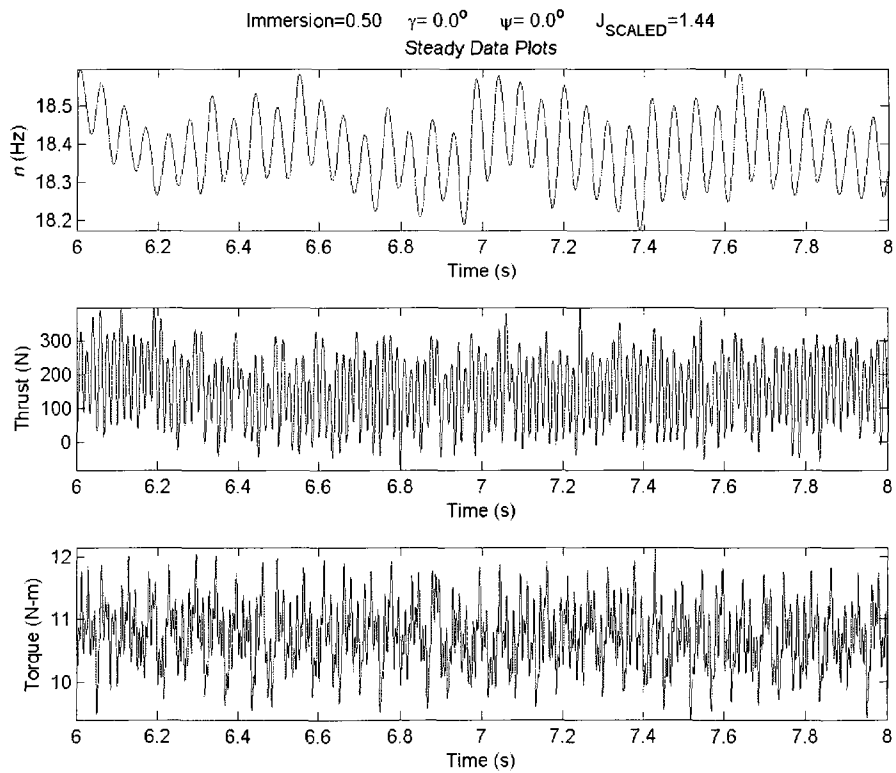


Figure E. 63

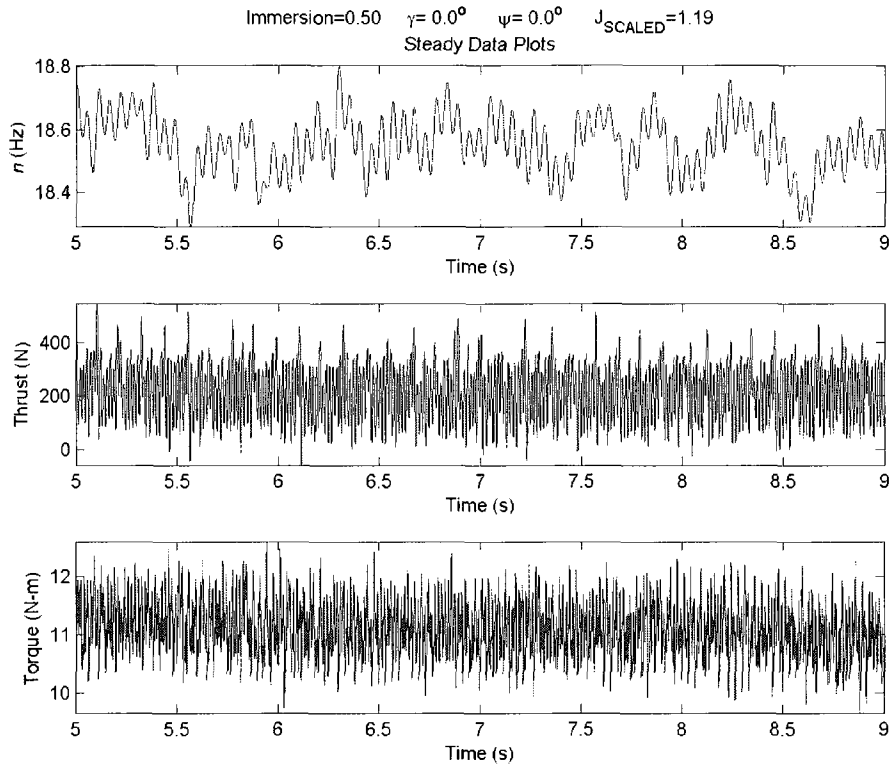


Figure E. 64

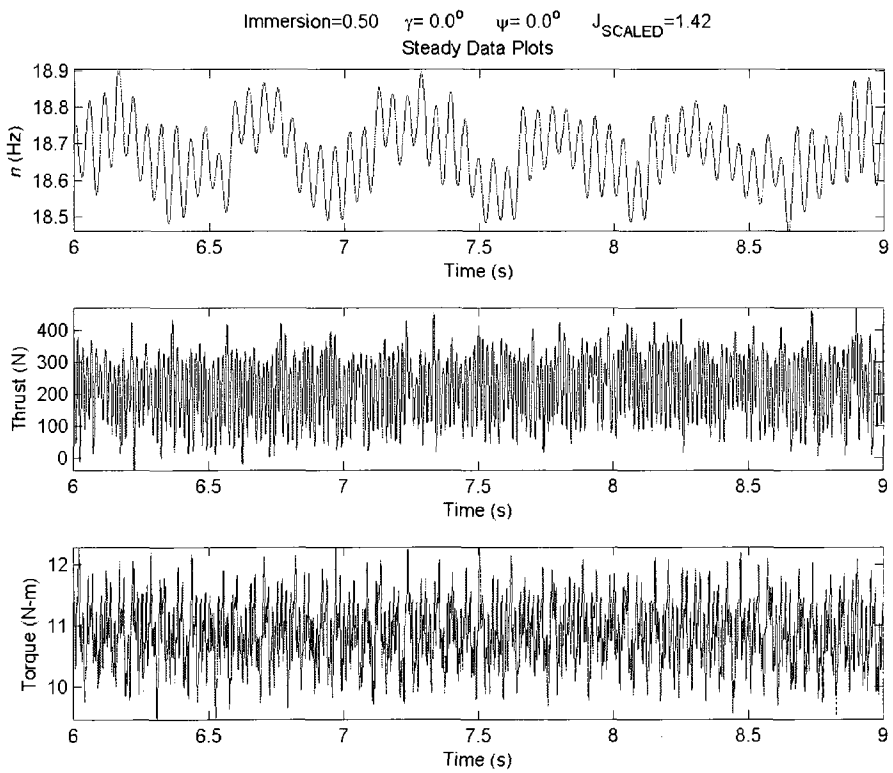


Figure E. 65

APPENDIX F: MATLAB DATA ANALYSIS CODE

The code presented here is the code which runs the data analysis for this report. The functions used in this code may be found at the end of the code.

```
clear
clc

master_dir = 'C:\Users\jlorio\Desktop\Master Data Analysis\';
%%
%This cell conatins all of the constants used throughout master_script and
%the functions and creates structure under Vars.(parameter)

%Pitch to diameter ratio
P_D = 18.312/9.70;

% Physical Constants
g = 9.81;          % gravity [m/s^2]

% Physical Properties
rho = 998.1;      % Density Freshwater [kg/m^3]

% Multiplicative Conversion Factors
lbf2N = 4.4482216;
ft2m = 0.30480;
in2m = 0.0254;

% Prop Dimensions
D = 9.7*in2m;    % Prop Diameter [m]

% Data measurement system constants
Fs = 1000; %sampling frequency in Hz set in labview

%Towing Carriage Conversions
AdvSpeed = 15; %15ft/s per Volt

%Slip Ring Conversions
PropSpeed = 2000/10/60; %2000RPM at 10V (/60 for Hz)
Position = 36; %10V = 360 degrees

% Force Transducer Conversions
Vexcite = ones(1,6)*10; % Excitation Voltage 10 [Volts] (All Channels)
Gain = ones(1,6)*1.0e3; % Gains (1000 All Channels)
CF = Gain.*Vexcite*1e-6;

S_metric_cross = xlsread((strcat(master_dir, 'Force Trans Properties\M49401.xls')), 'Cross', 'I14:N19');
%sensitivity matrix to go from forces to volts cancels crosstalk
```

```

S_english_cross = xlsread((strcat(master_dir, 'Force Trans Properties\M49401.xls')), 'Cross', 'B14:G19');
% sensitivity matrix to go from forces to volts cancels crosstalk
B_metric_cross = xlsread((strcat(master_dir, 'Force Trans Properties\M49401.xls')), 'Cross', 'I25:N30');
% inverted sensitivity matrix to go from volts to forces cancels crosstalk
B_english_cross = xlsread((strcat(master_dir, 'Force Trans Properties\M49401.xls')), 'Cross', 'B14:G19');
% inverted sensitivity matrix to go from volts to forces cancels crosstalk
S_metric = xlsread((strcat(master_dir, 'Force Trans Properties\M49401.xls')), 'Main', 'J13:N18'); % main
sensitivity matrix to go from forces to volts 6x1
S_english = xlsread((strcat(master_dir, 'Force Trans Properties\M49401.xls')), 'Main', 'D13:D18'); % main
sensitivity matrix to go from forces to volts 6x1

```

```

%% %% Create Structure Referencing all of the variables

```

```

Vars.P_D = P_D;
Vars.g = g;
Vars.rho = rho;
Vars.lbf2N = lbf2N;
Vars.ft2m = ft2m;
Vars.in2m = in2m;
Vars.D = D;
Vars.AdvSpeed = AdvSpeed;
Vars.PropSpeed = PropSpeed;
Vars.Position = Position;
Vars.Fs = Fs;
Vars.Vexcite = Vexcite;
Vars.Gain = Gain;
Vars.CF = CF;
Vars.S_metric_cross = S_metric_cross;
Vars.S_english_cross = S_english_cross;
Vars.B_metric_cross = B_metric_cross;
Vars.B_english_cross = B_english_cross;
Vars.S_metric = S_metric;
Vars.S_english = S_english;

```

```

save((strcat(master_dir, 'Structures\', 'Vars')), 'Vars');

```

```

clear P_D g rho lbf2N ft2m in2m D Fs AdvSpeed PropSpeed Position Vexcite Gain CF S_metric S_english
S_metric_cross S_english_cross B_metric_cross B_english_cross Vars

```

```

%%
%% %% functions called **txtreader**
%% %% This cell takes the directory dir_str and reads in all of the individual files
%% %% from each test run, which are in .txt format and converts them to .mat
%% %% files. The inputs from the slip ring are stored in the save directory
%% %% in a structure as follows
%% %% Raw.Phi = propeller position in raw voltage
%% %% Raw.RPM = propeller rpm in raw voltage
%% %% Raw.Fx = force in the x direction in raw voltage
%% %% Raw.Fy = force in the y direction in raw voltage
%% %% Raw.Thrust = force in the z direction in raw voltage
%% %% Raw.Mx = moment in the x direction in raw voltage
%% %% Raw.My = moment in the y direction in raw voltage
%% %% Raw.Torque = moment in the z direction in raw voltage
%% %% Raw.Speed = carriage speed in raw voltage
%% %% Raw.Filename = filename

```

```

dir_str = strcat(master_dir, 'Text Files\'); %read directory
sdir_str = strcat(master_dir, 'Raw Data mat Files\'); %save directory

txtreader(dir_str, sdir_str); %calling the function

clear dir_str sdir_str

%%
%%%%This cell loads in Files2Process.mat file which has 2 variables
%%%%data variable has 3 columns the first 2 columns are the numbers for min
%%%%and max sample number which bracket in the range of good data
%%%%textdata variable has 1 column that contains the names of the files
%%%%that have good data
    load(strcat(master_dir, 'Files2Process'));
    L = length(textdata);
%   filename = cell2struct(textdata,'name',2);    % Gives the filenames to convert
    for i = 1:L;

        file = textdata{i};
        Min = data(i,1);    % Min good sample value
        Max = data(i,2);    % Max good sample value
        Ucom = data(i,3);    % Commanded Velocity

        GoodRuns(i).Filename = file;
        GoodRuns(i).Min = Min;
        GoodRuns(i).Max = Max;
        GoodRuns(i).Ucom = Ucom;
    end
    save((strcat(master_dir, 'Structures\', 'GoodRuns')), 'GoodRuns')
    clear file Min Max Ucom data textdata GoodRuns Files2Process L i
%%
%%%%functions called **PSD(input)**
%%%%This cell takes the created directory of .mat files and calculates power
%%%%spectral density of all the channels. Produces elements for the
%%%%building of power spectral density plots in the following structure.
    %%%PYY.NFFT=NFFT;
    %%%PYY.f=f;
    %%%PYY.RPM
    %%%PYY.Thrust
    %%%PYY.Torque
    %%%PYY.Fx
    %%%PYY.Fy
    %%%PYY.Fx
    %%%PYY.Filename

clear PYY
dir_str = strcat(master_dir, 'Raw Data mat Files\'); %calls the file where the raw data is stored
load(strcat(master_dir, 'Structures\GoodRuns'))
load(strcat(master_dir, 'Structures\Vars'))
L = length(GoodRuns);
for i=1:L
    load(strcat(dir_str, GoodRuns(i).Filename, '.mat'));
    Min = GoodRuns(i).Min;
    Max = GoodRuns(i).Max;

```

```

    PYY(i) = PSD(Raw,Vars,Min,Max);
    clear Min Max
    clear(strcat(dir_str, GoodRuns(i).Filename, '.mat'))
end
save((strcat(master_dir, 'Structures\', 'PYY')), 'PYY')
clear dir_str GoodRuns PYY Vars Raw i L
%%
%%%%This program looks at the PSD when the carriage is stopped so we can
%%%%see if there is line noise

dir_str = strcat(master_dir, 'Raw Data mat Files\'); %calls the file where the raw data is stored
load(strcat(master_dir, 'Structures\GoodRuns'))
load(strcat(master_dir, 'Structures\Vars'))
L = length(GoodRuns);

for i=1:L
    load(strcat(dir_str, GoodRuns(i).Filename, '.mat'));
    filename = strcat(GoodRuns(i).Filename, '.mat');
    if(strcmp(filename, 'H33P00Y00JOP8_Run7_BalancedCarriage.mat')),
        PYY2(i) = PSD2(Raw,Vars,1,500)
    elseif(strcmp(filename, 'H33P00Y00J1P0_Run1.mat')),
        PYY2(i) = PSD2(Raw,Vars,1,500)
    elseif(strcmp(filename, 'H33P00Y00J1P4_Run5.mat')),
        PYY2(i) = PSD2(Raw,Vars,1,500)
    else
        Ln = length(Raw.Thrust);
        n = Ln-500;
        PYY2(i) = PSD2(Raw,Vars,n,Ln)

    end
    clear(strcat(dir_str, GoodRuns(i).Filename, '.mat'))
    clear Ln filename i n
end
save((strcat(master_dir, 'Structures\', 'PYY2')), 'PYY2')
clear dir_str Raw Vars
for i = 1:L
    subplot(211)
    plot(PYY2(i).f,PYY2(i).Thrust(1:PYY2(i).NFFT/2+1));
    subplot(212)
    plot(PYY2(i).f,PYY2(i).Torque(1:PYY2(i).NFFT/2+1));
    print ([strcat(master_dir, 'Plots\PSD2\', GoodRuns(i).Filename), '.jpg'], '-djpeg')
end
clear PYY2 GoodRuns i L

%%
%%%%Create Unfiltered Force Structure
clear ForceFiltered_Cross ForceFiltered Filtered Vars GoodRuns
dir_str = strcat(master_dir, 'Raw Data mat Files\');
load(strcat(master_dir, 'Structures\Filtered'));
load(strcat(master_dir, 'Structures\Vars'));
load(strcat(master_dir, 'Structures\GoodRuns'));

L = length(GoodRuns);
for i=1:L;
    load(strcat(dir_str, GoodRuns(i).Filename, '.mat'));
    fname = Raw.Filename;

```

```

ForceUnfiltered_Cross(i) = V2F_Cross(Raw,Vars,fname);
ForceUnfiltered_Cross(i).n = (-Raw.RPM)'.*Vars.PropSpeed;
ForceUnfiltered_Cross(i).Phi = (Raw.Phi)'.*Vars.Position;
ForceUnfiltered_Cross(i).Speed = (-Raw.Speed)'.*Vars.AdvSpeed*Vars.ft2m;
clear fname
end
clear i
for i=1:L
clear fname
load(strcat(dir_str, GoodRuns(i).Filename, '.mat'));
fname = Raw.Filename;
ForceUnfiltered(i) = V2F(Raw,Vars,fname);
ForceUnfiltered(i).n = (-Raw.RPM)'.*Vars.PropSpeed;
ForceUnfiltered(i).Phi = (Raw.Phi)'.*Vars.Position;
ForceUnfiltered(i).Speed = (-Raw.Speed)'.*Vars.AdvSpeed*Vars.ft2m;
clear fname
end
save((strcat(master_dir, 'Structures\', 'ForceUnfiltered')), 'ForceUnfiltered')
save((strcat(master_dir, 'Structures\', 'ForceUnfiltered_Cross')), 'ForceUnfiltered_Cross')
clear L ForceUnfiltered ForceUnfiltered_Cross Vars GoodRuns Raw i

%%
%%%%This function filters the raw data
clear Filtered
dir_str = strcat(master_dir, 'Raw Data mat Files\'); %calls the file where the raw data is stored
load(strcat(master_dir, 'Structures\GoodRuns'))
load(strcat(master_dir, 'Structures\Vars'))
for i=1:length(GoodRuns)
load(strcat(dir_str, GoodRuns(i).Filename, '.mat'));
Min = GoodRuns(i).Min;
Max = GoodRuns(i).Max;
Filtered(i) = Filter(Raw,Min,Max,Vars);
clear(strcat(dir_str, GoodRuns(i).Filename, '.mat'))
clear Min Max
end
save((strcat(master_dir, 'Structures\', 'Filtered')), 'Filtered')
clear dir_str GoodRuns Filtered Raw Vars i

%%
%%%%Convert voltage to forces
%%%%Use Filtered() Voltages to convert to Forces
clear ForceFiltered_Cross ForceFiltered Filtered Vars GoodRuns
load(strcat(master_dir, 'Structures\Filtered'));
load(strcat(master_dir, 'Structures\Vars'));
load(strcat(master_dir, 'Structures\GoodRuns'));

L = length(GoodRuns);
for i=1:L;
fname = Filtered(i).Filename;
ForceFiltered_Cross(i) = V2F_Cross(Filtered(i),Vars,fname);
ForceFiltered_Cross(i).n = (-Filtered(i).RPM)'.*Vars.PropSpeed;
ForceFiltered_Cross(i).Phi = (Filtered(i).Phi)'.*Vars.Position;
ForceFiltered_Cross(i).Speed = (-Filtered(i).Speed)'.*Vars.AdvSpeed*Vars.ft2m;
clear fname
end
clear i
for i=1:L

```

```

clear fname
fname = Filtered(i).Filename;
ForceFiltered(i) = V2F(Filtered(i), Vars, fname);
ForceFiltered(i).n = (-Filtered(i).RPM)' * Vars.PropSpeed;
ForceFiltered(i).Phi = (Filtered(i).Phi)' * Vars.Position;
ForceFiltered(i).Speed = (-Filtered(i).Speed)' * Vars.AdvSpeed * Vars.ft2m;
clear fname
end
save((strcat(master_dir, 'Structures\', 'ForceFiltered')), 'ForceFiltered')
save((strcat(master_dir, 'Structures\', 'ForceFiltered_Cross')), 'ForceFiltered_Cross')
clear L ForceFiltered ForceFiltered_Cross Vars GoodRuns Filtered i
%%
%%%%Make a Structure of run conditions
clear Conditions Conditions_Cross
load(strcat(master_dir, 'Structures\GoodRuns'));
load(strcat(master_dir, 'Structures\ForceFiltered'));
load(strcat(master_dir, 'Structures\Vars'));
L = length(GoodRuns);
for i = 1:L;
    Conditions(i) =
Condns(GoodRuns(i).Filename, GoodRuns(i).Min, GoodRuns(i).Max, GoodRuns(i).Ucom, ForceFiltered(i),
Vars);
    Conditions_Cross(i) =
Condns_Cross(GoodRuns(i).Filename, GoodRuns(i).Min, GoodRuns(i).Max, GoodRuns(i).Ucom, ForceFiltered(i), Vars);
end
for i = 1:L
    It(i) = Conditions(i).Immersion;
    It_Cross(i) = Conditions_Cross(i).Immersion;
    Pitch(i) = Conditions(i).Pitch;
    Pitch_Cross(i) = Conditions_Cross(i).Pitch;
    Yaw(i) = Conditions(i).Yaw;
    Yaw_Cross(i) = Conditions_Cross(i).Yaw;
    Ucom(i) = Conditions(i).Ucom;
    Ucom_Cross(i) = Conditions_Cross(i).Ucom;
    Uout(i) = Conditions(i).Uout;
    Uout_Cross(i) = Conditions_Cross(i).Uout;
    Mean_n(i) = Conditions(i).Mean_n;
    Mean_n_Cross(i) = Conditions_Cross(i).Mean_n;
    J(i) = Conditions(i).J;
    J_Cross(i) = Conditions_Cross(i).J;
    Jscaled(i) = Conditions(i).Jscaled;
    Jscaled_Cross(i) = Conditions_Cross(i).Jscaled;
    Thrust(i) = Conditions(i).Thrust;
    Thrust_Cross(i) = Conditions_Cross(i).Thrust;
    Torque(i) = Conditions(i).Torque;
    Torque_Cross(i) = Conditions_Cross(i).Torque;
    KT(i) = Conditions(i).KT;
    KT_Cross(i) = Conditions_Cross(i).KT;
    KQ(i) = Conditions(i).KQ;
    KQ_Cross(i) = Conditions_Cross(i).KQ;
    eta(i) = Conditions(i).eta;
    eta_Cross(i) = Conditions_Cross(i).eta;
    TorqueBias(i) = Conditions(i).TorqueBias;
    TorqueBias_Cross(i) = Conditions_Cross(i).TorqueBias;
    ThrustBias(i) = Conditions(i).ThrustBias;

```

```

    ThrustBias_Cross(i) =Conditions_Cross(i).ThrustBias
end

A =
[It;It_Cross;Pitch;Pitch_Cross;Yaw;Yaw_Cross;Ucom;Ucom_Cross;Uout;Uout_Cross;Mean_n;Mean_n_C
ross;J;J_Cross;Jscaled;Jscaled_Cross;Thrust;Thrust_Cross;Torque;Torque_Cross;KT;KT_Cross;KQ;KQ_C
ross;eta;eta_Cross;TorqueBias;TorqueBias_Cross;ThrustBias;ThrustBias_Cross];
    Title =
{'It';'It_Cross';'Pitch';'Pitch_Cross';'Yaw';'Yaw_Cross';'Ucom';'Ucom_Cross';'Uout';'Uout_Cross';'Mean_n';
'Mean_n_Cross';'J';'J_Cross';'Jscaled';'Jscaled_Cross';'Thrust';'Thrust_Cross';'Torque';'Torque_Cross';'KT';'K
T_Cross';'KQ';'KQ_Cross';'eta';'eta_Cross';'TorqueBias';'TorqueBias_Cross';'ThrustBias';'ThrustBias_Cross'
};
    clear It It_Cross Pitch Pitch_Cross Yaw Yaw_Cross Ucom Ucom_Cross Uout Uout_Cross Mean_n
Mean_n_Cross J J_Cross Jscaled Jscaled_Cross Thrust Thrust_Cross Torque Torque_Cross KT KT_Cross
KQ KQ_Cross eta eta_Cross
    B = {Title,A};
    xlswrite('C:\Users\jlorio\Desktop\Master Data Analysis\TestMatrix',Title,'Matrix','A1')
    xlswrite('C:\Users\jlorio\Desktop\Master Data Analysis\TestMatrix',A,'Matrix','B1')
    save((strcat(master_dir, 'Structures\','Conditions')), 'Conditions')
    save((strcat(master_dir, 'Structures\','Conditions_Cross')), 'Conditions_Cross')
    clear Conditions ForceFiltered i GoodRuns L Vars Conditions_Cross A Title B

%%
%%Plots filtered forces vs. raw forces
dir_str = strcat(master_dir, 'Raw Data mat Files\');
load(strcat(master_dir, 'Structures\ForceFiltered'));
load(strcat(master_dir, 'Structures\GoodRuns'));
load(strcat(master_dir, 'Structures\Conditions'));
load(strcat(master_dir, 'Structures\ForceUnfiltered'));
for i = 1:8
figure(i)
load(strcat(dir_str, GoodRuns(i).Filename, '.mat'));
subplot(211)
plot(ForceUnfiltered(i).Thrust)
title(sprintf('Immersion=%4.2f \ \delta=%4.1f^o \ \psi=%4.1f^o J=%4.2f, Conditions(i).Immersion,
Conditions(i).Pitch, Conditions(i).Yaw, Conditions(i).Jscaled))
hold on
plot(ForceFiltered(i).Thrust,'r')
hold off
subplot(212)
plot(ForceUnfiltered(i).Torque)
hold on
plot(ForceFiltered(i).Torque,'r')
hold off
end
clear Raw ForceFiltered ForceUnfiltered GoodRuns Conditions ThrustBias ThrustBias_Cross TorqueBias
TorqueBias_Cross dir_str
%%
%%Plots the section of the Filtered Data to use in the calculation of the
%%bias offset
clc
load(strcat(master_dir, 'Structures\ForceFiltered'));
load(strcat(master_dir, 'Structures\Conditions'));
L=length(ForceFiltered);

```



```

for i = 1:L
    if(strcmp(ForceFiltered(i).Filename,'H33P00Y00J0P8_Run7_BalancedCarriage.mat')),
        % Remove mean offsets from force and torque data
        Tmean = mean(ForceFiltered(i).Thrust(1:500));
        Qmean = mean(ForceFiltered(i).Torque(1:500));
        Ty = ones(size(ForceFiltered(i).Thrust))*Tmean;
        Qy = ones(size(ForceFiltered(i).Thrust))*Qmean;

        subplot(211)
        plot(ForceFiltered(i).Thrust)
        title(sprintf('Immersion=%4.2f \delta=%4.1f^o \psi=%4.1f^o J=%4.2f,
Conditions(i).Immersion, Conditions(i).Pitch, Conditions(i).Yaw, Conditions(i).Jscaled))
        hold on
        plot(Ty, 'g')
        hold off
        xlim([1 500])
        subplot(212)
        plot(ForceFiltered(i).Torque)
        hold on
        plot(Qy, 'g')
        hold off
        xlim([1 500])

    elseif(strcmp(ForceFiltered(i).Filename, 'H33P00Y00J1P0_Run1.mat')),
        % Remove mean offsets from force and torque data
        Tmean = mean(ForceFiltered(i).Thrust(1:500));
        Qmean = mean(ForceFiltered(i).Torque(1:500));
        Ty = ones(size(ForceFiltered(i).Thrust))*Tmean;
        Qy = ones(size(ForceFiltered(i).Thrust))*Qmean;

        subplot(211)
        plot(ForceFiltered(i).Thrust)
        title(sprintf('Immersion=%4.2f \delta=%4.1f^o \psi=%4.1f^o J=%4.2f,
Conditions(i).Immersion, Conditions(i).Pitch, Conditions(i).Yaw, Conditions(i).Jscaled))
        hold on
        plot(Ty, 'g')
        hold off
        xlim([1 500])
        subplot(212)
        plot(ForceFiltered(i).Torque)
        hold on
        plot(Qy,'g')
        hold off
        xlim([1 500])

    elseif(strcmp(ForceFiltered(i).Filename, 'H33P00Y00J1P4_Run5.mat')),
        % Remove mean offsets from force and torque data
        Tmean = mean(ForceFiltered(i).Thrust(1:500));
        Qmean = mean(ForceFiltered(i).Torque(1:500));
        Ty = ones(size(ForceFiltered(i).Thrust))*Tmean;
        Qy = ones(size(ForceFiltered(i).Thrust))*Qmean;

        subplot(211)
        plot(ForceFiltered(i).Thrust)
        title(sprintf('Immersion=%4.2f \delta=%4.1f^o \psi=%4.1f^o J=%4.2f,
Conditions(i).Immersion, Conditions(i).Pitch, Conditions(i).Yaw, Conditions(i).Jscaled))

```

```

    hold on
    plot(Ty, 'g')
    hold off
    xlim([1 500])
    subplot(212)
    plot(ForceFiltered(i).Torque)
    hold on
    plot(Qy, 'g')
    hold off
    xlim([1 500])

else
    % Remove mean offsets from force and torque data
    Ln = length(ForceFiltered(i).Thrust);
    n = Ln-500;
    Tmean = mean(ForceFiltered(i).Thrust(n:Ln))
    Qmean = mean(ForceFiltered(i).Torque(n:Ln))
    Ty = ones(size(ForceFiltered(i).Thrust))*Tmean;
    Qy = ones(size(ForceFiltered(i).Thrust))*Qmean;

    subplot(211)
    plot(ForceFiltered(i).Thrust)
    title(sprintf('Immersion=%4.2f \delta=%4.1f^o \psi=%4.1f^o J=%4.2f,
Conditions(i).Immersion, Conditions(i).Pitch, Conditions(i).Yaw, Conditions(i).Jscaled))
    hold on
    plot(Ty, 'g')
    hold off
    xlim([n Ln])
    subplot(212)
    plot(ForceFiltered(i).Torque)
    hold on
    plot(Qy, 'g')
    hold off
    xlim([n Ln])

end
print ([strcat(master_dir, 'Plots\Mean Offset\',ForceFiltered(i).Filename), '.jpg'], '-djpeg')

clear x
end
clear Conditions ForceFiltered L Ln Qmean Qy Tmean Ty i n
clf
%%
%%%% This section plots out Ferrando Regressions vs. actual test results

load(strcat(master_dir, 'Structures\Conditions'));
load(strcat(master_dir, 'Structures\Vars'));
L = length(Conditions);

for i = 1:L
    KQFerr(i) = (-0.300453*Conditions(i).Jscaled + 0.543738*Vars.P_D +
0.877638*Conditions(i).Jscaled*Vars.P_D-.649314*Conditions(i).Jscaled^2-.208974*Vars.P_D^2)/10;
    KTFerr(i) = -0.691625*Conditions(i).Jscaled + 0.794973*Vars.P_D +
0.870696*Conditions(i).Jscaled*Vars.P_D-.395012*Conditions(i).Jscaled^2-.515183*Vars.P_D^2;
    etaFerr(i) = Conditions(i).Jscaled*KTFerr(i)/KQFerr(i)/2/pi;
    KTscaled(i) = Conditions(i).KT;

```

```

KQscaled(i) = Conditions(i).KQ;
etascaled(i) = Conditions(i).eta;
Jscaled(i) = Conditions(i).Jscaled;
Pitch(i) = Conditions(i).Pitch;
Immersion(i) = Conditions(i).Immersion;
Yaw(i) = Conditions(i).Yaw;
end

```

```

FerrandPlot(KQFerr, KTFerr, etaFerr, KQscaled, KTscaled, etascaled, Jscaled, Pitch, Immersion, Yaw,
master_dir);

```

```

clear Conditions Jscaled KQFerr KQscaled KTFerr KTscaled L Vars etaFerr etascaled i Immersion Pitch
Yaw
%%
%%%%This section plots raw data for RPM POS SPEED / Fx Fy Fz/ Mx My Mz with
%%%%PSD on the side of the raw data plot
load(strcat(master_dir, 'Structures\ForceUnfiltered_Cross'));
load(strcat(master_dir, 'Structures\GoodRuns'));
load(strcat(master_dir, 'Structures\Conditions'));
load(strcat(master_dir, 'Structures\Vars'));
load(strcat(master_dir, 'Structures\PYY'));

L = length(GoodRuns);
for i=1:L;
%   dir_str = strcat(master_dir, 'Raw Data mat Files\');
%   load(strcat(dir_str, GoodRuns(i).Filename, '.mat'));

%   figure(1)
%   subplot(3,3,[1 2])

%   plot(ForceUnfiltered_Cross(i).n)
%   title('Raw Data')
%   xlabel('Time (ms)')
%   ylabel('\itn \rm(Hz)')
%   axis tight
%   subplot(3,3,3)

%   plot(PYY(i).f,PYY(i).RPM(1:PYY(i).NFFT/2+1));
%   title('Power Spectral Density')
%   xlabel('Frequency (Hz)')
%   ylabel('V^2/Hz')
%   axis([0 200 min(PYY(i).RPM) max(PYY(i).RPM)])

%   subplot(3,3,[4 5])

%   plot(ForceUnfiltered_Cross(i).Phi)
%   xlabel('Time (ms)')
%   ylabel('\it\phi \rm(deg)')
%   axis tight
%   subplot(3,3,6)

%   plot(PYY(i).f,PYY(i).Phi(1:PYY(i).NFFT/2+1));

```

```

xlabel('Frequency (Hz)')
ylabel('V^2/Hz')
axis([0 200 min(PYY(i).Phi) max(PYY(i).Phi)])

subplot(3,3,[7 8])

plot(ForceUnfiltered_Cross(i).Speed)
xlabel('Time (ms)')
ylabel('U_A (V)')
axis tight
subplot(3,3,9)

plot(PYY(i).f,PYY(i).Speed(1:PYY(i).NFFT/2+1));
xlabel('Frequency (Hz)')
ylabel('V^2/Hz')
axis([0 200 min(PYY(i).Speed) max(PYY(i).Speed)])

mtit(sprintf('Immersion=%4.2f \ \delta=%4.1f^o \ \psi=%4.1f^o J_{SCALED}=%4.2f,
Conditions(i).Immersion, Conditions(i).Pitch, Conditions(i).Yaw,
Conditions(i).Jscaled), 'xoff,0.0,'yoff,.0255)
% print ([strcat(master_dir, 'Plots\Raw Data and
PSD\Conditions','It',num2str(Conditions(i).Immersion),'Pitch',num2str(Conditions(i).Pitch),'Yaw',num2str(
Conditions(i).Yaw),'J',num2str(Conditions(i).Jscaled)), '.jpg'], '-djpeg')
print ([strcat(master_dir, 'Plots\Raw Data and
PSD\Conditions','It',num2str(Conditions(i).Immersion),'Pitch',num2str(Conditions(i).Pitch),'Yaw',num2str(
Conditions(i).Yaw),'J',num2str(Conditions(i).Jscaled)), '.jpg'], '-djpeg')

% figure(2)
subplot(3,3,[1 2])
plot(ForceUnfiltered_Cross(i).Thrust)
title('Raw Forces')
xlabel('Time (ms)')
ylabel('Thrust (N)')
axis tight
subplot(3,3,3)
plot(PYY(i).f,PYY(i).Thrust(1:PYY(i).NFFT/2+1));
title('Power Spectral Density')
xlabel('Frequency (Hz)')
ylabel('V^2/Hz')
axis([0 200 min(PYY(i).Thrust) max(PYY(i).Thrust)])

subplot(3,3,[4 5])
plot(ForceUnfiltered_Cross(i).Fx)
xlabel('Time (ms)')
ylabel('\itF_z \rm(N)')
axis tight
subplot(3,3,6)
plot(PYY(i).f,PYY(i).Fx(1:PYY(i).NFFT/2+1));
xlabel('Frequency (Hz)')
ylabel('V^2/Hz')
axis([0 200 min(PYY(i).Fx) max(PYY(i).Fx)])

subplot(3,3,[7 8])
plot(ForceUnfiltered_Cross(i).Fy)
xlabel('Time (ms)')
ylabel('\itF_y \rm(N)')

```

```

axis tight
subplot(3,3,9)
plot(PYY(i).f,PYY(i).Fy(1:PYY(i).NFFT/2+1));
xlabel('Frequency (Hz)')
ylabel('V^2/Hz')
axis([0 200 min(PYY(i).Fy) max(PYY(i).Fy)])

mtit(sprintf('Immersion=%4.2f \ \delta=%4.1f^o \ \psi=%4.1f^o \ J_{SCALED}=%4.2f',
Conditions(i).Immersion, Conditions(i).Pitch, Conditions(i).Yaw,
Conditions(i).Jscaled),'xoff,0.0,'yoff,.0255)
% print ([strcat(master_dir, 'Plots\Raw Data and
PSD\Forces',It',num2str(Conditions(i).Immersion),'Pitch',num2str(Conditions(i).Pitch),'Yaw',num2str(Con
ditions(i).Yaw),'J',num2str(Conditions(i).Jscaled)),'.jpg'],'-djpeg')
print ([strcat(master_dir, 'Plots\Raw Data and
PSD\Forces',It',num2str(Conditions(i).Immersion),'Pitch',num2str(Conditions(i).Pitch),'Yaw',num2str(Con
ditions(i).Yaw),'J',num2str(Conditions(i).Jscaled)),'.jpg'],'-djpeg')

figure(3)
subplot(3,3,[1 2])
plot(ForceUnfiltered_Cross(i).Torque)
title('Raw Moments')
xlabel('Time (ms)')
ylabel('Torque (N-m)')
axis tight
subplot(3,3,3)
plot(PYY(i).f,PYY(i).Torque(1:PYY(i).NFFT/2+1));
title('Power Spectral Density')
xlabel('Frequency (Hz)')
ylabel('V^2/Hz')
axis([0 200 min(PYY(i).Torque) max(PYY(i).Torque)])

subplot(3,3,[4 5])
plot(ForceUnfiltered_Cross(i).Mx)
xlabel('Time (ms)')
ylabel('\itM_z \rm(N-m)')
axis tight
subplot(3,3,6)
plot(PYY(i).f,PYY(i).Mx(1:PYY(i).NFFT/2+1));
xlabel('Frequency (Hz)')
ylabel('V^2/Hz')
axis([0 200 min(PYY(i).Mx) max(PYY(i).Mx)])

subplot(3,3,[7 8])
plot(ForceUnfiltered_Cross(i).My)
xlabel('Time (ms)')
ylabel('\itM_y \rm(N-m)')
axis tight
subplot(3,3,9)
plot(PYY(i).f,PYY(i).My(1:PYY(i).NFFT/2+1));
xlabel('Frequency (Hz)')
ylabel('V^2/Hz')
axis([0 200 min(PYY(i).My) max(PYY(i).My)])

```

```

        mtit(sprintf('Immersion=%4.2f \ \delta=%4.1f^o \ \psi=%4.1f^o J_{SCALED}=%4.2f,
Conditions(i).Immersion, Conditions(i).Pitch, Conditions(i).Yaw,
Conditions(i).Jscaled),'xoff,0.0,'yoff,.0255)
% print ([strcat(master_dir, 'Plots\Raw Data and
PSD\Moments','It',num2str(Conditions(i).Immersion),'Pitch',num2str(Conditions(i).Pitch),'Yaw',num2str(C
onditions(i).Yaw),'J',num2str(Conditions(i).Jscaled)), '.jpg'], '-djpeg')
print ([strcat(master_dir, 'Plots\Raw Data and
PSD\Moments','It',num2str(Conditions(i).Immersion),'Pitch',num2str(Conditions(i).Pitch),'Yaw',num2str(C
onditions(i).Yaw),'J',num2str(Conditions(i).Jscaled)), '.jpg'], '-djpeg')
end
clear Conditions ForceUnfiltered_Cross GoodRuns L PYY Vars i
%%
%%%% this section creates a plot of the steady section of data for RPM
%%%% Thrust and Torque
load(strcat(master_dir, 'Structures\ForceFiltered_Cross'));
load(strcat(master_dir, 'Structures\Conditions'));
load(strcat(master_dir, 'Structures\GoodRuns'));

L = length(GoodRuns);
for i = 1:L

    subplot(3,1,1)

    plot((GoodRuns(i).Min:GoodRuns(i).Max)/1000,ForceFiltered_Cross(i).n(GoodRuns(i).Min:GoodRuns(i).
Max))
        xlabel('Time (s)')
        ylabel('\itn \rm(Hz)')
        title('Steady Data Plots')
        axis tight

    subplot(3,1,2)

    plot((GoodRuns(i).Min:GoodRuns(i).Max)/1000,ForceFiltered_Cross(i).Thrust(GoodRuns(i).Min:GoodRu
ns(i).Max)-Conditions(i).ThrustBias)
        xlabel('Time (s)')
        ylabel('Thrust (N)')
        axis tight

    subplot(3,1,3)

    plot((GoodRuns(i).Min:GoodRuns(i).Max)/1000,ForceFiltered_Cross(i).Torque(GoodRuns(i).Min:GoodRu
ns(i).Max)-Conditions(i).TorqueBias)
        xlabel('Time (s)')
        ylabel('Torque (N-m)')
        axis tight

        mtit(sprintf('Immersion=%4.2f \ \delta=%4.1f^o \ \psi=%4.1f^o J_{SCALED}=%4.2f,
Conditions(i).Immersion, Conditions(i).Pitch, Conditions(i).Yaw,
Conditions(i).Jscaled),'xoff,0.0,'yoff,.0255)
        print ([strcat(master_dir, 'Plots\n Thrust Torque
Steady','It',num2str(Conditions(i).Immersion),'Pitch',num2str(Conditions(i).Pitch),'Yaw',num2str(Conditio
ns(i).Yaw),'J',num2str(Conditions(i).Jscaled)), '.jpg'], '-djpeg')
end

%%
%%%% This section creates a Kt Kq Eff prediction

```

```

load(strcat(master_dir, 'Structures\Vars'));

J = [0.4 0.6 0.8 1.0 1.2 1.4 1.6 1.8];
L = length(J);

for i = 1:L
    KQFerr(i) = (-0.300453*J(i) + 0.543738*Vars.P_D + 0.877638*J(i)*Vars.P_D-.649314*J(i)^2-
.208974*Vars.P_D^2)/10;
    KTFerr(i) = -0.691625*J(i) + 0.794973*Vars.P_D + 0.870696*J(i)*Vars.P_D-.395012*J(i)^2-
.515183*Vars.P_D^2;
    etaFerr(i) = J(i)*KTFerr(i)/KQFerr(i)/2/pi;
end

plot(J,KTFerr, '-r')
hold on
[AX,H1,H2] = plotyy(J,KQFerr,J,etaFerr,'plot');

set(get(AX(1),'Ylabel'),'String','\itK_T\rm 10\itK_Q')
set(get(AX(2),'Ylabel'),'String','\iteta')
hold off

clear AX H1 H2 J KQFerr KTFerr L Vars etaFerr i

%%
%% Create a small Plot for position to show noise in data at prop freq
%% Create a plot for My to show channel saturation
load(strcat(master_dir, 'Structures\ForceUnfiltered_Cross'));
load(strcat(master_dir, 'Structures\Vars'));
load(strcat(master_dir, 'Structures\GoodRuns'));
load(strcat(master_dir, 'Structures\Conditions'));

for i = 1
    figure(1)
    plot(12500:12750,ForceUnfiltered_Cross(i).Phi(12500:12750))
    xlabel('Time (ms)')
    ylabel('\it\phi \rm(degrees)')
    title(sprintf('Immersion=%4.2f \ \delta=%4.1f^o \ \psi=%4.1f^o J=%4.2f,
Conditions(i).Immersion, Conditions(i).Pitch, Conditions(i).Yaw, Conditions(i).Jscaled))
    print('C:\Users\jlorio\Pictures\Thesis Figures\NoisyPosition', '-djpeg')
end
%%

```

Appendix F.1: Function Condns

```
function Conditions = Condns(file,Min,Max,Ucom,ForceFiltered,Vars)

% .file,GoodRuns.Min,GoodRuns.Max,GoodRuns.Ucom
filename = strcat(file,'.mat');

% filename = strcat(data_dir, file(i).name,'.mat');
% load([filename]);

% Perform the velocity conversion
temp = regexp(filename,'H\d\d','match');
temp2 = cell2struct(temp,'text',1);
Immersion = temp2.text;

if(Immersion == 'H33'),
    Immersion = 0.33;
else Immersion = 0.5;
end
Conditions.Immersion = Immersion;

temp = regexp(filename,'P\d\d','match');
temp2 = cell2struct(temp,'text',1);
Pitch = temp2.text;

if(Pitch == 'P00'),
    Pitch = 0.0;
elseif Pitch == 'P75', Pitch = 7.5;
else Pitch = 15.0;
end
Conditions.Pitch = Pitch;

temp = regexp(filename,'Y\d\d','match');
temp2 = cell2struct(temp,'text',1);
Yaw = temp2.text;

if(Yaw == 'Y00'),
    Yaw = 0.0;
elseif Yaw == 'Y15', Yaw = 15;
else Yaw = 30.0;
end
Conditions.Yaw = Yaw;
Conditions.Ucom = Ucom;
Conditions.Uout = [];
Conditions.Uout = CoordTrans(Conditions.Ucom, Conditions.Pitch, Conditions.Yaw);
Conditions.Mean_n = mean(ForceFiltered.n(Min:Max));
Conditions.J = (Conditions.Ucom*Vars.ft2m)/Conditions.Mean_n/Vars.D;
Conditions.Jscaled = (Conditions.Uout*Vars.ft2m)/Conditions.Mean_n/Vars.D;

if(strcmp(filename,'H33P00Y00J0P8_Run7_BalancedCarriage.mat')),
    % Remove mean offsets from force and torque data
    Conditions.TorqueBias = mean(ForceFiltered.Torque(1:500));
    Conditions.ThrustBias = mean(ForceFiltered.Thrust(1:500));
    Conditions.Thrust = mean(ForceFiltered.Thrust(Min:Max))-Conditions.ThrustBias;
    Conditions.Torque = mean(ForceFiltered.Torque(Min:Max))-Conditions.TorqueBias;
```



```

elseif(strcmp(filename, 'H33P00Y00J1P0_Run1.mat')),
    % Remove mean offsets from force and torque data
    Conditions.TorqueBias = mean(ForceFiltered.Torque(1:500));
    Conditions.ThrustBias = mean(ForceFiltered.Thrust(1:500));
    Conditions.Thrust = mean(ForceFiltered.Thrust(Min:Max))-Conditions.ThrustBias;
    Conditions.Torque = mean(ForceFiltered.Torque(Min:Max))-Conditions.TorqueBias;
elseif(strcmp(filename, 'H33P00Y00J1P4_Run5.mat')),
    % Remove mean offsets from force and torque data
    Conditions.TorqueBias = mean(ForceFiltered.Torque(1:500));
    Conditions.ThrustBias = mean(ForceFiltered.Thrust(1:500));
    Conditions.Thrust = mean(ForceFiltered.Thrust(Min:Max))-Conditions.ThrustBias;
    Conditions.Torque = mean(ForceFiltered.Torque(Min:Max))-Conditions.TorqueBias;
else
    % Remove mean offsets from force and torque data
    Ln = length(ForceFiltered.Thrust);
    n = Ln-500;
    Conditions.TorqueBias = mean(ForceFiltered.Torque(n:Ln));
    Conditions.ThrustBias = mean(ForceFiltered.Thrust(n:Ln));
    Conditions.Thrust = mean(ForceFiltered.Thrust(Min:Max))-Conditions.ThrustBias;
    Conditions.Torque = mean(ForceFiltered.Torque(Min:Max))-Conditions.TorqueBias;
end
Conditions.KT    = Conditions.Thrust/Vars.rho/Conditions.Mean_n^2/Vars.D^4;
Conditions.KQ    = Conditions.Torque/Vars.rho/Conditions.Mean_n^2/Vars.D^5;
% Conditions.eta = Conditions.Jscaled*Conditions.KT/2/pi/Conditions.KQ;
Conditions.eta  =
Conditions.Thrust*Conditions.Uout*Vars.ft2m/(2*pi*Conditions.Mean_n*Conditions.Torque);
Conditions.Filename = file;

```

Appendix F.2: Function Condns_Cross

```
function Conditions_Cross = Condns_Cross(file,Min,Max,Ucom,ForceFiltered,Vars)

% .file,GoodRuns.Min,GoodRuns.Max,GoodRuns.Ucom
filename = strcat(file,'.mat');

% filename = strcat(data_dir, file(i).name,'.mat');
% load([filename]);

% Perform the velocity conversion
temp = regexp(filename,'H\d\d','match');
temp2 = cell2struct(temp,'text',1);
Immersion = temp2.text;

if(Immersion == 'H33'),
    Immersion = 0.33;
else Immersion =0.5;
end
Conditions_Cross.Immersion = Immersion;

temp = regexp(filename,'P\d\d','match');
temp2 = cell2struct(temp,'text',1);
Pitch = temp2.text;

if(Pitch == 'P00'),
    Pitch = 0.0;
elseif Pitch == 'P75', Pitch = 7.5;
else Pitch = 15.0;
end
Conditions_Cross.Pitch = Pitch;

temp = regexp(filename,'Y\d\d','match');
temp2 = cell2struct(temp,'text',1);
Yaw = temp2.text;

if(Yaw == 'Y00'),
    Yaw = 0.0;
elseif Yaw == 'Y15', Yaw = 15;
else Yaw = 30.0;
end
Conditions_Cross.Yaw = Yaw;
Conditions_Cross.Ucom = Ucom;
Conditions_Cross.Uout = [];
Conditions_Cross.Uout = CoordTrans(Conditions_Cross.Ucom, Conditions_Cross.Pitch,
Conditions_Cross.Yaw);
Conditions_Cross.Mean_n = mean(ForceFiltered.n(Min:Max));
Conditions_Cross.J = (Conditions_Cross.Ucom*Vars.ft2m)/Conditions_Cross.Mean_n/Vars.D;
Conditions_Cross.Jscaled = (Conditions_Cross.Uout*Vars.ft2m)/Conditions_Cross.Mean_n/Vars.D;

if(strcmp(filename,H33P00Y00J0P8_Run7_BalancedCarriage.mat')),
    % Remove mean offsets from force and torque data
    Conditions_Cross.TorqueBias = mean(ForceFiltered.Torque(1:500));
    Conditions_Cross.ThrustBias = mean(ForceFiltered.Thrust(1:500));
```

```

    Conditions_Cross.Thrust = mean(ForceFiltered.Thrust(Min:Max))-Conditions_Cross.ThrustBias;
    Conditions_Cross.Torque = mean(ForceFiltered.Torque(Min:Max))-Conditions_Cross.TorqueBias;
elseif(strcmp(filename, 'H33P00Y00J1P0_Run1.mat')),
    % Remove mean offsets from force and torque data
    Conditions_Cross.TorqueBias = mean(ForceFiltered.Torque(1:500));
    Conditions_Cross.ThrustBias = mean(ForceFiltered.Thrust(1:500));
    Conditions_Cross.Thrust = mean(ForceFiltered.Thrust(Min:Max))-Conditions_Cross.ThrustBias;;
    Conditions_Cross.Torque = mean(ForceFiltered.Torque(Min:Max))-Conditions_Cross.TorqueBias;;
elseif(strcmp(filename, 'H33P00Y00J1P4_Run5.mat')),
    % Remove mean offsets from force and torque data
    Conditions_Cross.TorqueBias = mean(ForceFiltered.Torque(1:500));
    Conditions_Cross.ThrustBias = mean(ForceFiltered.Thrust(1:500));
    Conditions_Cross.Thrust = mean(ForceFiltered.Thrust(Min:Max))-Conditions_Cross.ThrustBias;;
    Conditions_Cross.Torque = mean(ForceFiltered.Torque(Min:Max))-Conditions_Cross.TorqueBias;;
else
    % Remove mean offsets from force and torque data
    Ln = length(ForceFiltered.Thrust);
    n = Ln-500;
    Conditions_Cross.TorqueBias = mean(ForceFiltered.Torque(n:Ln));
    Conditions_Cross.ThrustBias = mean(ForceFiltered.Thrust(n:Ln));
    Conditions_Cross.Thrust = mean(ForceFiltered.Thrust(Min:Max))-Conditions_Cross.ThrustBias;;
    Conditions_Cross.Torque = mean(ForceFiltered.Torque(Min:Max))-Conditions_Cross.TorqueBias;;
end
Conditions_Cross.KT = Conditions_Cross.Thrust/Vars.rho/Conditions_Cross.Mean_n^2/Vars.D^4;
Conditions_Cross.KQ = Conditions_Cross.Torque/Vars.rho/Conditions_Cross.Mean_n^2/Vars.D^5;
% Conditions_Cross.eta =
Conditions_Cross.Jscaled*Conditions_Cross.KT/2/pi/Conditions_Cross.KQ;
Conditions_Cross.eta =
Conditions_Cross.Thrust*Conditions_Cross.Uout*Vars.ft2m/(2*pi*Conditions_Cross.Mean_n*Conditions
_Cross.Torque);
Conditions_Cross.FileName = file;

```

Appendix F.3: Function CoordTrans

```
function Uout = CoordTrans(Ucom, gamma, psi)
% Ucom = Commanded Towing Velocity
% Pitch = Shaft Inclination Angle [deg]
% Yaw = Yaw Angle [deg]

% gamma = GoodRuns.Pitch;
% psi = GoodRuns.Yaw;

p1 = 0.0;      % Roll angle [deg]
p2 = gamma;    % Shaft Inclination Angle [deg]
p3 = psi;      % Yaw Angle [deg]

% Rotation from Inertial Coordinates to Body-Fixed Coordinates
Rib = [
    cosd(p2)*cosd(p3) -cosd(p1)*sind(p3)+sind(p1)*sind(p2)*cosd(p3)
    sind(p1)*sind(p3)+cosd(p1)*sind(p2)*cosd(p3)

    cosd(p2)*sind(p3) cosd(p1)*cosd(p3)+sind(p1)*sind(p2)*sind(p3) -
    sind(p1)*cosd(p3)+cosd(p1)*sind(p2)*sind(p3)

    -sind(p2)      sind(p1)*cosd(p2)          cosd(p1)*cosd(p2)
    ];

V = [Ucom 0 0]';
VA = Rib*V;

Uout = VA(1);

return
```

Appendix F.4: Function FerrandPlot

```
function FerrandPlot(KQFerr, KTFerr, etaFerr, KQscaled, KTscaled, etascaled, Jscaled, Pitch, Immersion,
Yaw, master_dir)
```

```

figure(1)
subplot(3,1,1);
[Jscaled_P, IX] = sort(Jscaled(1:8));
KTFerr_P = KTFerr(IX);
plot(Jscaled_P,KTFerr_P(1:8));
title(sprintf('Immersion=%4.2f \ \delta=%4.1f^o \ \psi=%4.1f^o', Immersion(1), Pitch(1),
Yaw(1)))
xlabel('J_{SCALED}');
ylabel('K_T');
% axis tight
% axis([.7 2 -.25 1]);
hold on;
KTscaled_P = KTscaled(IX);
plot(Jscaled_P(1:8),KTscaled_P(1:8), 'r');
% axis([.7 2 -.25 1]);
legend('K_T Predicted','K_T Experimental','Location','NorthEastOutside')
hold off;

subplot(3,1,2);
KQFerr_P = KQFerr(IX);
plot(Jscaled_P(1:8),KQFerr_P(1:8));
xlabel('J_{SCALED}');
ylabel('K_Q');
% axis([.7 2 0 .15]);
hold on;
KQscaled_P = KQscaled(IX);
plot(Jscaled_P(1:8),KQscaled_P(1:8),'r');
% axis([.7 2 0 .15]);
legend('K_Q Predicted','K_Q Experimental','Location','NorthEastOutside')
hold off;

subplot(3,1,3);
etaFerr_P = etaFerr(IX);
plot(Jscaled_P(1:8),etaFerr_P(1:8));
xlabel('J_{SCALED}');
ylabel('\eta_{SCALED}')
% axis tight
% axis([.7 2 0 3]);
hold on;
etascaled_P = etascaled(IX);
plot(Jscaled_P(1:8),etascaled_P(1:8), 'r');
% axis tight
% axis([.7 2 0 3]);
legend('\eta Predicted','\eta Experimental','Location','NorthEastOutside')

hold off;
print ([strcat(master_dir, 'Plots\KT KQ
EFF\,'It',num2str(Immersion(1)), 'Pitch',num2str(Pitch(1)), 'Yaw',num2str(Yaw(1))), '.jpg'], '-djpeg')
```

```

figure(2)
    subplot(3,1,1);
    [Jscaled_P, IX] = sort(Jscaled(9:17));
    KTFerr_P = KTFerr(IX);
    plot(Jscaled_P,KTFerr_P);
    title(sprintf('Immersion=%4.2f   \delta=%4.1f^o   \psi=%4.1f^o', Immersion(9), Pitch(9),
Yaw(9)));
    xlabel('J_{SCALED}');
    ylabel('K_T');
    % axis tight
    % axis([.7 2 -.25 1]);
    hold on;
    KTscaled_P = KTscaled(IX);
    plot(Jscaled_P,KTscaled_P, 'r');
    % axis tight
    % axis([.7 2 -.25 1]);
    legend('K_T Predicted','K_T Experimental','Location','NorthEastOutside')
    hold off;

    subplot(3,1,2);
    KQFerr_P = KQFerr(IX);
    plot(Jscaled_P,KQFerr_P);
    xlabel('J_{SCALED}');
    ylabel('K_Q');
    % axis tight
    % axis([.7 2 0 .15]);
    hold on;
    KQscaled_P = KQscaled(IX);
    plot(Jscaled_P,KQscaled_P,'r');
    % axis tight
    % axis([.7 2 0 .15]);
    legend('K_Q Predicted','K_Q Experimental','Location','NorthEastOutside')
    hold off;

    subplot(3,1,3);
    etaFerr_P = etaFerr(IX);
    plot(Jscaled_P,etaFerr_P);
    xlabel('J_{SCALED}');
    ylabel('\eta_{SCALED}');
    % axis tight
    % axis([.7 2 0 3]);
    hold on;
    etascaled_P = etascaled(IX);
    plot(Jscaled_P,etascaled_P, 'r');
    % axis tight
    % axis([.7 2 0 3]);
    legend('\eta Predicted','\eta Experimental','Location','NorthEastOutside')
    hold off;
    print ([strcat(master_dir, 'Plots\KT KQ
EFF\,'It',num2str(Immersion(9)),'Pitch',num2str(Pitch(9)),'Yaw',num2str(Yaw(9))),'.jpg'], '-djpeg')

```

```

figure(3)
    subplot(3,1,1);
    [Jscaled_P, IX] = sort(Jscaled(18:23));
    KTFerr_P = KTFerr(IX);
    plot(Jscaled_P,KTFerr_P);

```

```

    title(sprintf('Immersion=%4.2f  \delta=%4.1f^o  \psi=%4.1f^o', Immersion(18), Pitch(18),
Yaw(18)))
    xlabel('J_{SCALED}');
    ylabel('K_T');
%    axis tight
%    axis([.7 2 -.25 1]);
    hold on;
    KTscaled_P = KTscaled(IX);
    plot(Jscaled_P,KTscaled_P, 'r');
%    axis tight
%    axis([.7 2 -.25 1]);
    legend('K_T Predicted','K_T Experimental','Location','NorthEastOutside')
    hold off;

    subplot(3,1,2);
    KQFerr_P = KQFerr(IX);
    plot(Jscaled_P,KQFerr_P);
    xlabel('J_{SCALED}');
    ylabel('K_Q');
%    axis tight
%    axis([.7 2 0 .15]);
    hold on;
    KQscaled_P = KQscaled(IX);
    plot(Jscaled_P,KQscaled_P,'r');
%    axis tight
%    axis([.7 2 0 .15]);
    legend('K_Q Predicted','K_Q Experimental','Location','NorthEastOutside')
    hold off;

    subplot(3,1,3);
    etaFerr_P = etaFerr(IX);
    plot(Jscaled_P,etaFerr_P);
    xlabel('J_{SCALED}');
    ylabel('\eta_{SCALED}');
%    axis tight
%    axis([.7 2 0 3]);
    hold on;
    etascaled_P = etascaled(IX);
    plot(Jscaled_P,etascaled_P, 'r');
%    axis tight
%    axis([.7 2 0 3]);
    legend('\eta Predicted','\eta Experimental','Location','NorthEastOutside')
    hold off;
    print ([strcat(master_dir, 'Plots\KT KQ
EFF\','It',num2str(Immersion(18)), 'Pitch',num2str(Pitch(18)), 'Yaw',num2str(Yaw(18))), '.jpg'], '-djpeg')

figure(4)
subplot(3,1,1);
[Jscaled_P, IX] = sort(Jscaled(24:29));
KTFerr_P = KTFerr(IX);
plot(Jscaled_P,KTFerr_P);
title(sprintf('Immersion=%4.2f  \delta=%4.1f^o  \psi=%4.1f^o', Immersion(24), Pitch(24),
Yaw(24)));
xlabel('J_{SCALED}');
ylabel('K_T');
%    axis tight

```

```

%   axis([.7 2 -.25 1]);
hold on;
KTscaled_P = KTscaled(IX);
plot(Jscaled_P,KTscaled_P, 'r');
%   axis tight
%   axis([.7 2 -.25 1]);
legend('K_T Predicted','K_T Experimental','Location','NorthEastOutside')
hold off;

subplot(3,1,2);
KQFerr_P = KQFerr(IX);
plot(Jscaled_P,KQFerr_P);
xlabel('J_{SCALED}');
ylabel('K_Q');
%   axis tight
%   axis([.7 2 0 .15]);
hold on;
KQscaled_P = KQscaled(IX);
plot(Jscaled_P,KQscaled_P,'r');
%   axis tight
%   axis([.7 2 0 .15]);
legend('K_Q Predicted','K_Q Experimental','Location','NorthEastOutside')
hold off;

subplot(3,1,3);
etaFerr_P = etaFerr(IX);
plot(Jscaled_P,etaFerr_P);
xlabel('J_{SCALED}');
ylabel('\eta_{SCALED}');
%   axis tight
%   axis([.7 2 0 3]);
hold on;
etascaled_P = etascaled(IX);
plot(Jscaled_P,etascaled_P, 'r');
%   axis tight
%   axis([.7 2 0 3]);
legend('\eta Predicted','\eta Experimental','Location','NorthEastOutside')
hold off;
print ([strcat(master_dir, 'Plots\KT KQ
EFF\,'It',num2str(Immersion(24)), 'Pitch',num2str(Pitch(24)), 'Yaw',num2str(Yaw(24))), '.jpg'], '-djpeg')

figure(5)
subplot(3,1,1);
[Jscaled_P, IX] = sort(Jscaled(30:35));
KTFerr_P = KTFerr(IX);
plot(Jscaled_P,KTFerr_P);
title(sprintf('Immersion=%4.2f \ \delta=%4.1f^o \ \psi=%4.1f^o', Immersion(30), Pitch(30),
Yaw(30)));
xlabel('J_{SCALED}');
ylabel('K_T');
%   axis tight
%   axis([.7 2 -.25 1]);
hold on;
KTscaled_P = KTscaled(IX);
plot(Jscaled_P,KTscaled_P, 'r');
%   axis tight

```



```

% axis([.7 2 -.25 1]);
legend('K_T Predicted','K_T Experimental','Location','NorthEastOutside')
hold off;

subplot(3,1,2);
KQFerr_P = KQFerr(IX);
plot(Jscaled_P,KQFerr_P);
xlabel('J_{SCALED}');
ylabel('K_Q');
% axis tight
% axis([.7 2 0 .15]);
hold on;
KQscaled_P = KQscaled(IX);
plot(Jscaled_P,KQscaled_P,'r');
% axis tight
% axis([.7 2 0 .15]);
legend('K_Q Predicted','K_Q Experimental','Location','NorthEastOutside')
hold off;

subplot(3,1,3);
etaFerr_P = etaFerr(IX);
plot(Jscaled_P,etaFerr_P);
xlabel('J_{SCALED}');
ylabel('\eta_{SCALED}');
% axis tight
% axis([.7 2 0 3]);
hold on;
etascaled_P = etascaled(IX);
plot(Jscaled_P,etascaled_P, 'r');
% axis tight
% axis([.7 2 0 3]);
legend('\eta Predicted','\eta Experimental','Location','NorthEastOutside')
hold off;
print ([strcat(master_dir, 'Plots\KT KQ
EFF\','It',num2str(Immersion(30)), 'Pitch',num2str(Pitch(30)), 'Yaw',num2str(Yaw(30))),'.jpg'], '-djpeg')

figure(6)
subplot(3,1,1);
[Jscaled_P, IX] = sort(Jscaled(36:42));
KTFerr_P = KTFerr(IX);
plot(Jscaled_P,KTFerr_P);
title(sprintf('Immersion=%4.2f \ \delta=%4.1f^o \ \psi=%4.1f^o', Immersion(36), Pitch(36),
Yaw(36)));
xlabel('J_{SCALED}');
ylabel('K_T');
% axis tight
% axis([.7 2 -.25 1]);
hold on;
KTscaled_P = KTscaled(IX);
plot(Jscaled_P,KTscaled_P, 'r');
% axis tight
% axis([.7 2 -.25 1]);
legend('K_T Predicted','K_T Experimental','Location','NorthEastOutside')
hold off;

subplot(3,1,2);

```

```

KQFerr_P = KQFerr(IX);
plot(Jscaled_P,KQFerr_P);
xlabel('J_{SCALED}');
ylabel('K_Q');
% axis tight
% axis([.7 2 0 .15]);
hold on;
KQscaled_P = KQscaled(IX);
plot(Jscaled_P,KQscaled_P,'r');
% axis tight
% axis([.7 2 0 .15]);
legend('K_Q Predicted','K_Q Experimental','Location','NorthEastOutside')
hold off;

subplot(3,1,3);
etaFerr_P = etaFerr(IX);
plot(Jscaled_P,etaFerr_P);
xlabel('J_{SCALED}');
ylabel('\eta_{SCALED}');
% axis tight
% axis([.7 2 0 3]);
hold on;
etascaled_P = etascaled(IX);
plot(Jscaled_P,etascaled_P, 'r');
% axis tight
% axis([.7 2 0 3]);
legend('\eta Predicted','\eta Experimental','Location','NorthEastOutside')
hold off;
print ([strcat(master_dir, 'Plots\KT KQ
EFF\, 'It',num2str(Immersion(36)), 'Pitch',num2str(Pitch(36)), 'Yaw',num2str(Yaw(36))), '.jpg'], '-djpeg')

```

```

figure(7)
subplot(3,1,1);
[Jscaled_P, IX] = sort(Jscaled(43:47));
KTFerr_P = KTFerr(IX);
plot(Jscaled_P,KTFerr_P);
title(sprintf('Immersion=%4.2f \ \delta=%4.1f^o \ \psi=%4.1f^o', Immersion(43), Pitch(43),
Yaw(43)));
xlabel('J_{SCALED}');
ylabel('K_T');
% axis tight
% axis([.7 2 -.25 1]);
hold on;
KTscaled_P = KTscaled(IX);
plot(Jscaled_P,KTscaled_P, 'r');
% axis tight
% axis([.7 2 -.25 1]);
legend('K_T Predicted','K_T Experimental','Location','NorthEastOutside')
hold off;

subplot(3,1,2);
KQFerr_P = KQFerr(IX);
plot(Jscaled_P,KQFerr_P);
xlabel('J_{SCALED}');
ylabel('K_Q');

```

```

% axis tight
% axis([.7 2 0 .15]);
hold on;
KQscaled_P = KQscaled(IX) ;
plot(Jscaled_P,KQscaled_P,'r');
% axis tight
% axis([.7 2 0 .15]);
legend('K_Q Predicted','K_Q Experimental','Location','NorthEastOutside')
hold off;

subplot(3,1,3);
etaFerr_P = etaFerr(IX);
plot(Jscaled_P,etaFerr_P);
xlabel('J_{SCALED}');
ylabel('\eta_{SCALED}');
% axis tight
% axis([.7 2 0 3]);
hold on;
etascaled_P = etascaled(IX);
plot(Jscaled_P,etascaled_P, 'r');
% axis tight
% axis([.7 2 0 3]);
legend('\eta Predicted','\eta Experimental','Location','NorthEastOutside')
hold off;
print ([strcat(master_dir, 'Plots\KT KQ
EFF\','It',num2str(Immersion(43)), 'Pitch',num2str(Pitch(43)), 'Yaw',num2str(Yaw(43))), '.jpg'], '-djpeg')

figure(8)
subplot(3,1,1);
[Jscaled_P, IX] = sort(Jscaled(48:53));
KTFerr_P = KTFerr(IX);
plot(Jscaled_P,KTFerr_P);
title(sprintf('Immersion=%4.2f \delta=%4.1f^o \psi=%4.1f^o', Immersion(48), Pitch(48),
Yaw(48)));
xlabel('J_{SCALED}');
ylabel('K_T');
% axis tight
% axis([.7 2 -.25 1]);
hold on;
KTscaled_P = KTscaled(IX) ;
plot(Jscaled_P,KTscaled_P, 'r');
% axis tight
% axis([.7 2 -.25 1]);
legend('K_T Predicted','K_T Experimental','Location','NorthEastOutside')
hold off;

subplot(3,1,2);
KQFerr_P = KQFerr(IX);
plot(Jscaled_P,KQFerr_P);
xlabel('J_{SCALED}');
ylabel('K_T');
% axis tight
% axis([.7 2 0 .15]);
hold on;
KQscaled_P = KQscaled(IX);

```

```

    plot(Jscaled_P,KQscaled_P,'r');
%    axis tight
%    axis([.7 2 0 .15]);
    legend('K_Q Predicted','K_Q Experimental','Location','NorthEastOutside')
    hold off;

    subplot(3,1,3);
    etaFerr_P = etaFerr(IX);
    plot(Jscaled_P,etaFerr_P);
    xlabel('J_{SCALED}');
    ylabel('\eta_{SCALED}');
%    axis tight
%    axis([.7 2 0 3]);
    hold on;
    etascaled_P = etascaled(IX);
    plot(Jscaled_P,etascaled_P, 'r');
%    axis tight
%    axis([.7 2 0 3]);
    legend('\eta Predicted','\eta Experimental','Location','NorthEastOutside')
    hold off;
    print ([strcat(master_dir, 'Plots\KT KQ
EFF\','t',num2str(Immersion(48)), 'Pitch',num2str(Pitch(48)), 'Yaw',num2str(Yaw(48))), '.jpg'], '-djpeg')

    figure(9)
    subplot(3,1,1);
    [Jscaled_P, IX] = sort(Jscaled(54:60));
    KTFerr_P = KTFerr(IX);
    plot(Jscaled_P,KTFerr_P);
    title(sprintf('Immersion=%4.2f \ \delta=%4.1f^o \ \psi=%4.1f^o', Immersion(54), Pitch(54),
Yaw(54)));
    xlabel('J_{SCALED}');
    ylabel('K_T');
%    axis tight
%    axis([.7 2 -.25 1]);
    hold on;
    KTscaled_P = KTscaled(IX);
    plot(Jscaled_P,KTscaled_P, 'r');
%    axis tight
%    axis([.7 2 -.25 1]);
    legend('K_T Predicted','K_T Experimental','Location','NorthEastOutside')
    hold off;

    subplot(3,1,2);
    KQFerr_P = KQFerr(IX);
    plot(Jscaled_P,KQFerr_P);
    xlabel('J_{SCALED}');
    ylabel('K_Q');
%    axis tight
%    axis([.7 2 0 .15]);
    hold on;
    KQscaled_P = KQscaled(IX);
    plot(Jscaled_P,KQscaled_P,'r');
%    axis tight
%    axis([.7 2 0 .15]);
    legend('K_Q Predicted','K_Q Experimental','Location','NorthEastOutside')
    hold off;

```

```

subplot(3,1,3);
etaFerr_P = etaFerr(IX);
plot(Jscaled_P,etaFerr_P);
xlabel('J_{SCALED}');
ylabel('\eta_{SCALED}');
% axis tight
% axis([.7 2 0 3]);
hold on;
etascaled_P = etascaled(IX);
plot(Jscaled_P,etascaled_P, 'r');
% axis tight
% axis([.7 2 0 3]);
legend('\eta Predicted', '\eta Experimental', 'Location', 'NorthEastOutside')
hold off;
print ([strcat(master_dir, 'Plots\KT KQ
EFF\, 'It', num2str(Immersion(54)), 'Pitch', num2str(Pitch(54)), 'Yaw', num2str(Yaw(54))), '.jpg'], '-djpeg')

```

```

figure(10)
subplot(3,1,1);
[Jscaled_P, IX] = sort(Jscaled(61:65));
KTFerr_P = KTFerr(IX);
plot(Jscaled_P, KTFerr_P);
title(sprintf('Immersion=%4.2f \delta=%4.1f^o \psi=%4.1f^o', Immersion(61), Pitch(61),
Yaw(61)));
xlabel('J_{SCALED}');
ylabel('K_T');
% axis tight
% axis([.7 2 -.25 1]);
hold on;
KTscaled_P = KTscaled(IX);
plot(Jscaled_P, KTscaled_P, 'r');
% axis tight
% axis([.7 2 -.25 1]);
legend('K_T Predicted', 'K_T Experimental', 'Location', 'NorthEastOutside')
hold off;

```

```

subplot(3,1,2);
KQFerr_P = KQFerr(IX);
plot(Jscaled_P, KQFerr_P);
xlabel('J_{SCALED}');
ylabel('K_Q');
% axis tight
% axis([.7 2 0 .15]);
hold on;
KQscaled_P = KQscaled(IX);
plot(Jscaled_P, KQscaled_P, 'r');
% axis tight
% axis([.7 2 0 .15]);
legend('K_Q Predicted', 'K_Q Experimental', 'Location', 'NorthEastOutside')
hold off;

```

```

subplot(3,1,3);
etaFerr_P = etaFerr(IX);
plot(Jscaled_P, etaFerr_P);
xlabel('J_{SCALED}');

```

```

    ylabel('\eta_{SCALED}');
%   axis tight
%   axis([.7 2 0 3]);
    hold on;
    etascaled_P = etascaled(IX);
    plot(Jscaled_P,etascaled_P, 'r');
%   axis tight
%   axis([.7 2 0 3]);
    legend('\eta Predicted', '\eta Experimental', 'Location', 'NorthEastOutside')
    print ([strcat(master_dir, 'Plots\KT KQ
EFF\, 'It', num2str(Immersion(61)), 'Pitch', num2str(Pitch(61)), 'Yaw', num2str(Yaw(61))), '.jpg'], '-djpeg')
    hold off;

```

Appendix F.5: Function Filter

```

function Filtered = Filter(Raw,Min,Max,Vars)

N_Low = Min;
N_High = Max;

L = length(Raw.Thrust(N_Low:N_High)); %Length of Signal

NFFT = 2^nextpow2(L); %Creates the number of points for the fft
%fprop = 20; %prop freq in hz
%fprop = mean (RPM(N_Low:N_High))/60; %use to get value from RPM data

% Filters
fney = Vars.Fs/2;
Wney = 2*pi*fney;
fprop = 20; %approximate prop freq in hz

%use to get value from RPM
%fprop = mean (RPM(N_Low:N_High))*2000/10/60; %2000/10/60 is
%conversion from voltage to prop frequency

%This section of code can be used to plot the filter shape
%[bcoeff, acoeff] = cheby1(n,Rp, Wn,'low'); %gives b and a coeffs for filter
%figure(3)
%clf
%freqz(bcoeff,acoef,NFFT,Fs);

% Filter RPM
%filter inputs
Harmonic = 1; % Harmonics to keep
Wn = Harmonic*2*pi*fprop/Wney; % Cutoff Frequency
% Chebyshev
n = 9; % Filter Order
Rp = 0.5; % Ripple
% Implement Filter
[bcoeff, acoeff] = cheby1(n,Rp, Wn,'low'); %originally harmon=4 and Wn/4
Filtered.RPM = filter(bcoeff,acoef,Raw.RPM);%((N_Low:N_High))

% Filter Speed
% filter inputs
Harmonic = 1; % Harmonics to keep
Wn = Harmonic*2*pi*.5/Wney; % Cutoff Frequency, use .5 instead of fprop
% Chebyshev
n = 2; % Filter Order
Rp = 0.5; % Ripple
% Implement Filter
[bcoeff, acoeff] = cheby1(n,Rp, Wn,'low');
Filtered.Speed = filter(bcoeff,acoef,Raw.Speed);%((N_Low:N_High))

% Following 2 lines produced acceptable filtered data
% [bcoeff, acoeff] = cheby1(2, Rp, 2*pi*0.5/Wney,'low'); %
% Y_out_Speed = filter(bcoeff,acoef,Speed);%((N_Low:N_High))

```

```

% Filter Thrust
%filter inputs
    Harmonic = 4;          % Harmonics to keep

    Wn = Harmonic*2*pi*fprop/Wney; % Cutoff Frequency
% Chebyshev
    n = 9;                % Filter Order
    Rp= 0.5;              % Ripple
% Implement Filter
    [bcoeff, acoeff] = cheby1(n,Rp, Wn,'low');
    Filtered.Thrust = filter(bcoeff,acoef,Raw.Thrust);%((N_Low:N_High))

% Filter Torque
%filter inputs
    Harmonic = 4;          % Harmonics to keep

    Wn = Harmonic*2*pi*fprop/Wney; % Cutoff Frequency
% Chebyshev
    n = 9;                % Filter Order
    Rp= 0.5;              % Ripple
% Implement Filter
    [bcoeff, acoeff] = cheby1(n,Rp, Wn,'low');
    Filtered.Torque = filter(bcoeff,acoef,Raw.Torque);%((N_Low:N_High))

% Filter Fx
%filter inputs
    Harmonic = 6;          % Harmonics to keep
    Wn = Harmonic*2*pi*fprop/Wney; % Cutoff Frequency
% Chebyshev
    n = 9;                % Filter Order
    Rp= 0.5;              % Ripple
% Implement Filter
    [bcoeff, acoeff] = cheby1(n,Rp, Wn,'low');
    Filtered.Fx = filter(bcoeff,acoef,Raw.Fx);%((N_Low:N_High))

% Filter Fy
%filter inputs
    Harmonic = 6;          % Harmonics to keep
    Wn = Harmonic*2*pi*fprop/Wney; % Cutoff Frequency
% Chebyshev
    n = 9;                % Filter Order
    Rp= 0.5;              % Ripple
% Implement Filter
    [bcoeff, acoeff] = cheby1(n,Rp, Wn,'low');
    Filtered.Fy = filter(bcoeff,acoef,Raw.Fy);%((N_Low:N_High))

% Filter Mx
%filter inputs
    Harmonic = 6;          % Harmonics to keep
    Wn = Harmonic*2*pi*fprop/Wney; % Cutoff Frequency
% Chebyshev
    n = 9;                % Filter Order
    Rp= 0.5;              % Ripple
% Implement Filter

```



```
[bcoeff, acoeff] = cheby1(n,Rp, Wn,'low');  
Filtered.Mx = filter(bcoeff,acoeff,Raw.Mx);%((N_Low:N_High))
```

```
%Adding Parts to the Filtered. structure  
Filtered.Phi = Raw.Phi;  
Filtered.FileName = Raw.FileName;
```

```
return
```

Appendix F.6: Function PSD

```
function PYY = PSD(Raw,Vars,Min,Max)
%%%%This function computes the power spectral density of the raw data

%%%%Data coming in is in the following structures
%%%% Raw.Phi = propeller position in raw voltage
%%%% Raw.RPM = propeller rpm in raw voltage
%%%% Raw.Fx = force in the x direction in raw voltage
%%%% Raw.Fy = force in the y direction in raw voltage
%%%% Raw.Thrust = force in the z direction in raw voltage
%%%% Raw.Mx = moment in the x direction in raw voltage
%%%% Raw.My = moment in the y direction in raw voltage
%%%% Raw.Torque = moment in the z direction in raw voltage
%%%% Raw.Speed = carriage speed in raw voltage
%%%% Raw.Filename = filename
%%%% Vars.g = g;
%%%% Vars.rho = rho;
%%%% Vars.lbf2N = lbf2N;
%%%% Vars.ft2m = ft2m;
%%%% Vars.in2m = inwm;
%%%% Vars.D = D;
%%%% Vars.Fs = Fs;
%%%% Vars.Vexcite = Vexcite;
%%%% Vars.Gain = Gain;
%%%% Vars.CF = CF;
%%%% Vars.S_metric = S_metric;
%%%% Vars.S_english = S_english;
%%%% Vars.B_metric = B_metric;
%%%% Vars.B_english = B_english;

N_Low = Min;%to select specific area of data
N_High = Max;
Fs = Vars.Fs; %Sampling Frequency
T = 1/Fs; %Period

L = length(Raw.Thrust(N_Low:N_High)); %Length of Signal

%Computing the number of points to use for the FFT
NFFT = 2^nextpow2(L); %Create the number of points to use in the fast fourier transform calc
f = Fs/2*linspace(0,1,NFFT/2+1); %calculating the frequency

%Computing the mean of the selected portion of data
MeanRPM = mean (Raw.RPM(N_Low:N_High));
MeanSpeed = mean (Raw.Speed(N_Low:N_High));
MeanPhi = mean (Raw.Phi(N_Low:N_High));
MeanThrust = mean(Raw.Thrust(N_Low:N_High));
MeanTorque = mean(Raw.Torque(N_Low:N_High));
MeanFx = mean(Raw.Fx(N_Low:N_High));
MeanFy = mean(Raw.Fy(N_Low:N_High));
MeanMx = mean(Raw.Mx(N_Low:N_High));
MeanMy = mean(Raw.My(N_Low:N_High));
```

```

%PSD for Speed Data
Yspeed= fft(Raw.Speed(N_Low:N_High)-MeanSpeed, NFFT); %not sure what this does either
PYYspeed = Yspeed.*conj(Yspeed)/NFFT;

%PSD for Position Data
Yphi= fft(Raw.Phi(N_Low:N_High)-MeanPhi, NFFT); %not sure what this does either
PYYphi = Yphi.*conj(Yphi)/NFFT;

%PSD for RPM Data
Yrpm= fft(Raw.RPM(N_Low:N_High)-MeanRPM, NFFT); %not sure what this does either
PYYrpm = Yrpm.*conj(Yrpm)/NFFT;

%PSD for Thrust Data
Ythrust = fft(Raw.Thrust(N_Low:N_High)-MeanThrust, NFFT); %not sure what this does either
PYYthrust = Ythrust.*conj(Ythrust)/NFFT;

%PSD for Torque Data
Ytorque = fft(Raw.Torque(N_Low:N_High)-MeanTorque, NFFT); %not sure what this does either
PYYtorque = Ytorque.*conj(Ytorque)/NFFT;

%PSD for Fx Data
Yfx= fft(Raw.Fx(N_Low:N_High)-MeanFx, NFFT); %not sure what this does either
PYYfx = Yfx.*conj(Yfx)/NFFT;

%PSD for Fy Data
Yfy = fft(Raw.Fy(N_Low:N_High)-MeanFy, NFFT); %not sure what this does either
PYYfy = Yfy.*conj(Yfy)/NFFT;

%PSD for Mx Data
Ymx = fft(Raw.Mx(N_Low:N_High)-MeanMx, NFFT); %not sure what this does either
PYYmx = Ymx.*conj(Ymx)/NFFT;

%PSD for My Data
Ymy = fft(Raw.My(N_Low:N_High)-MeanMy, NFFT); %not sure what this does either
PYYmy = Ymy.*conj(Ymy)/NFFT;

%Create a Structure for Data COming out

PYY.NFFT=NFFT;
PYY.f=f;
PYY.RPM=PYYrpm;
PYY.Thrust=PYYthrust;
PYY.Torque=PYYtorque;
PYY.Fx=PYYfx;
PYY.Fy=PYYfy;
PYY.Mx=PYYmx;
PYY.My=PYYmy;
PYY.FileName = Raw.FileName;
PYY.Speed=PYYspeed;
PYY.Phi=PYYphi;

```

Appendix F.7: Function txtreader

```
function Raw = txtreader(dir_str, sdir_str)
    %This function reads in the directory stored in the master script. That
    %directory contains the .txt files that were converted from signal express
    %and is in the form of voltage output

files = dir(sprintf('%s*', dir_str));

%Use size of return to specify how many files will be processed.
num = size(files,1);

for i = 1:num-2,
    filename = files(i+2).name;
    fid = fopen([dir_str filename], 'r');
    filename = strrep(filename, '.', '');
    fdata = textscan(fid, '%f %f %f %f %f %f %f %f %f', 'headerlines', 7);

    fclose(fid);

    Raw.Phi = fdata{1};
    Raw.RPM = fdata{2};
    Raw.Fx = fdata{3};
    Raw.Fy = fdata{4};
    Raw.Thrust = fdata{5};
    Raw.Mx = fdata{6};
    Raw.My = fdata{7};
    Raw.Torque = fdata{8};
    Raw.Speed = fdata{9};
    Raw.Filename = filename;

    save([sdir_str filename], 'Raw')
end
```

Appendix F.8: Function V2F

```
function ForceFiltered = V2F(Filtered,Vars,fname)

% Output forces are in [N] and output torque is in [N-m]

% Inverse Sensitivity Matrix - for cancelling cross talk
S = Vars.S_metric;
S = S';
%Since My Channel is getting bad, Substitute with 0's
L = length(Filtered.Fx);
Filtered.My = zeros(L,1);
CF = Vars.CF;

% Method 1 - single channel calibrations
% Sensitivities [microVolts/Voltexc-N]

ForceFiltered.Fx = Filtered.Fx/S(1)/CF(1);
ForceFiltered.Fy = Filtered.Fy/S(2)/CF(2);
ForceFiltered.Thrust = -Filtered.Thrust/S(3)/CF(3);
ForceFiltered.Mx = Filtered.Mx/S(4)/CF(4);
ForceFiltered.My = Filtered.My/S(5)/CF(5);
ForceFiltered.Torque = -Filtered.Torque/S(6)/CF(6);
ForceFiltered.Filename = fname;

ForceFiltered.Fx = ForceFiltered.Fx';
ForceFiltered.Fy = ForceFiltered.Fy';
ForceFiltered.Thrust = ForceFiltered.Thrust';
ForceFiltered.Mx = ForceFiltered.Mx';
ForceFiltered.My = ForceFiltered.My';
ForceFiltered.Torque = ForceFiltered.Torque';
ForceFiltered.n = [];
ForceFiltered.Phi = [];
ForceFiltered.Speed = [];

return
```

Appendix F.9: Function V2F_Cross

```
function ForceFiltered_Cross = V2F_Cross(Filtered,Vars,fname)

% Output forces are in [N] and output torque is in [N-m]

% Inverse Sensitivity Matrix - for cancelling cross talk
B = Vars.B_metric_cross;

%Since My Channel is getting bad, Substitute with 0's
L = length(Filtered.Fx);
Filtered.My = zeros(L,1);

for i=1:L;

    vect = [Filtered.Fx(i) Filtered.Fy(i) Filtered.Thrust(i) Filtered.Mx(i) Filtered.My(i) Filtered.Torque(i)];

    FM = B*(vect./Vars.CF)';
    % FM = B*(vect.*Vars.CF)'; %seeing if multiplying by CF makes difference

    ForceFiltered_Cross.Fx(i) = FM(1);
    ForceFiltered_Cross.Fy(i) = FM(2);
    ForceFiltered_Cross.Thrust(i) = -FM(3);
    ForceFiltered_Cross.Mx(i) = FM(4);
    ForceFiltered_Cross.My(i) = FM(5);
    ForceFiltered_Cross.Torque(i) = -FM(6);
    ForceFiltered_Cross.n = [];
    ForceFiltered_Cross.Phi = [];
    ForceFiltered_Cross.Speed = [];
end
ForceFiltered_Cross.FileName = fname;
return
```

APPENDIX G: MATRICES FOR VOLTAGE TO FORCE CONVERSION

Table G. 1. Main sensitivity matrices

English Units	Metric Units
<u>Main Sensitivity</u>	<u>Main Sensitivity</u>
Fx 1.5818 microVolt/Voltexc-lb	Fx 0.3556 microVolt/Voltexc-N
Fy 1.5848 microVolt/Voltexc-lb	Fy 0.3563 microVolt/Voltexc-N
Fz 0.3912 microVolt/Voltexc-lb	Fz 0.0880 microVolt/Voltexc-N
Mx 2.3309 microVolt/Voltexc-in-lb	Mx 20.6308 microVolt/Voltexc-N-m
My 2.3375 microVolt/Voltexc-in-lb	My 20.6893 microVolt/Voltexc-N-m
Mz 1.5793 microVolt/Voltexc-in-lb	Mz 13.9781 microVolt/Voltexc-N-m

Table G. 2. Cross talk matrices

English Units	Metric Unit
<u>Sensitivity matrix S(l,j)</u>	<u>Sensitivity matrix S(l,j)</u>
1.5771 -0.0082 0.0095 -0.0064 -0.0046 -0.0019	0.3546 -0.0018 0.0021 -0.0564 -0.0410 -0.0164
0.0207 1.5796 0.0028 0.0051 -0.0101 -0.0066	0.0046 0.3551 0.0006 0.0452 -0.0893 -0.0582
-0.0056 0.0066 0.3912 -0.0013 -0.0031 0.0048	-0.0013 0.0015 0.0880 -0.0119 -0.0278 0.0428
0.0047 -0.0569 -0.0090 2.3320 -0.0054 0.0115	0.0011 -0.0128 -0.0020 20.6406 -0.0480 0.1022
-0.0571 0.0213 -0.0417 0.0129 2.3391 0.0190	-0.0128 0.0048 -0.0094 0.1140 20.7036 0.1684
-0.0298 0.0230 -0.0019 -0.0149 -0.0148 1.5793	-0.0067 0.0052 -0.0004 -0.1316 -0.1310 13.9781
<u>Inverted Sensitivity Matrix (B(l,i))</u>	<u>Inverted Sensitivity Matrix (B(l,i))</u>
0.6340 0.0034 -0.0153 0.0017 0.0013 0.0008	2.8201 0.0151 -0.0681 0.0076 0.0056 0.0035
-0.0082 0.6329 -0.0040 -0.0014 0.0027 0.0026	-0.0363 2.8152 -0.0180 -0.0063 0.0121 0.0116
0.0092 -0.0105 2.5564 0.0015 0.0034 -0.0079	0.0409 -0.0468 11.3704 0.0065 0.0149 -0.0352
-0.0015 0.0154 0.0099 0.4288 0.0011 -0.0031	-0.0002 0.0017 0.0011 0.0484 0.0001 -0.0004
0.0156 -0.0059 0.0451 -0.0023 0.4275 -0.0053	0.0018 -0.0007 0.0051 -0.0003 0.0483 -0.0006
0.0122 -0.0091 0.0033 0.0041 0.0040 0.6331	0.0014 -0.0010 0.0004 0.0005 0.0005 0.0715