

Naval Free Space Optical communications

A simulation & control design of optical terminals operated in naval environments

Supervisors: M. Noppen, R. Saathof & G-J Netjes
Mateo T. Kuiken Mendoza

Naval Free Space Optical communications

A simulation & control design of optical
terminals operated in naval environments

by

Mateo T. Kuiken Mendoza

To obtain the degree of Master of Science at the Delft University of Technology, to be defended
publicly on Tuesday 17th of September 2024.

Organization Supervisors:	M. Noppen, G-J Netjes
University supervisor:	R. Saathof
Project Duration:	01, 2024 - 08, 2024
Faculty:	Faculty of Aerospace Engineering, Delft

The Defence Space Security Centre part of the Royal Netherlands Airforce (Koninklijke Luchtmacht)
has supported the work in this thesis.



Preface

Optical communications is one of those elusive and intriguing technologies, that currently receives plenty of investigation investments, to make it a practical addition to existing radio frequency communications. The first time I heard about it, was during the optical instrumentation lectures, as part of my journey through the space engineering master courses. While I immediately found the topic captivating, I believed at the time that there were far too many challenges still to be solved for the technology to come to anything remotely close to practical applications.

As I advanced through my master's, I was searching for an internship and a recommendation came through by my girlfriend, to apply at the Space Defense Security Center. When I reached out, they asked me to create a list of topics that I was interested in performing work on. While I was creating that list, classic research topics such as micro-satellite engineering, space debris, etc, came first to mind. When the first version of that list was complete, I wasn't satisfied. I then proceeded to iterate, rethinking what I was truly passionate about and then I remembered optical communications. Then I decided to add it to the list and it turned out to be the only topic of interest for which an internship was available.

During the internship, I finally started to appreciate how the technology readiness level of the technology was so much further ahead of what I originally thought. I then began to think about possible novel applications of the technology in my spare time, therefore when the opportunity arose to perform my master's thesis on the expansion of practical applications, I was sold into the topic from day one.

I would like to thank M. Noppen and G-J Neetjes, for giving me the opportunity to dive into the world of optical communications at the Defence Space Security Centre, as well as for helping me to troubleshoot the many problems I have faced during the research. I would like to thank R. Saathof for his time, patience and guidance throughout the research process, it has proven critical for achieving the results presented in this work. I would like to thank my parents, Ana Elena Mendoza Gonzales and Klaas Kuiken, for providing me with the care, tools and means to complete my studies and have shaped the person I am. I would like to thank my oldest sister Fernanda Bernadette Kuiken Mendoza and my girlfriend Valerie Wendela Schindeler, for their continuous support and encouragement, which helped me to get through the difficult parts of this research and life.

*Mateo T. Kuiken Mendoza
Delft, September 2024*

Summary

Wireless Optical Communication (WOC) has recently become a viable complementary technology in addition to existing Radio Frequency Communication (RFC). One of its critical advantages over existing RFC is its significant increase in carrier frequency, since "it is a well-known fact that an increase in carrier frequency increases the information-carrying capacity of a communication system" [24]. WOC has several critical advantages over existing RFC, namely tariff-free bandwidth allocation, low on-board power requirements and improved channel security. Due to the rise in global tensions, the last advantage can prove a crucial asset for military applications.

There are also several critical challenges of WOC, namely its high dependability on stable atmospheric conditions. Recent advances in Vertical-Cavity Surface-Emitting Lasers (VCSEL's), Avalanche Photodiodes (APD's), Adaptive Optics, Real-Time Control, Active Optics, etc, all have allowed addressing the challenges of WOC, increasing the feasibility of implementing such technology as a practical communication system. The application of WOC for Ground to Low-Earth-Orbit communication is referred to as Free Space Optical Communication (FSOC). This application remains the focus of this work.

To perform FSOC, 3 main elements are required, these include the transmitter, the transmission channel and the receiver. The fundamental working principle of this technology is to modulate the information into an optical carrier, typically performed at the transmitter in simplex mode. As a section of the transmission channel passes through Earth's atmosphere, this results in the introduction of disturbances, due to the presence of turbulence in the different layers of Earth's atmosphere.

Therefore a functional FSOC system must possess precise and efficient pointing capabilities at both receiver and transmitter ends, such that the communication channel has the capacity to compensate for the atmospheric disturbances. Therefore the Pointing Acquisition and Tracking (PAT) system must comply with stringent angular and angular-rate pointing requirements.

While there is plenty of contemporary research regarding the design and testing of ground-based Optical Communications Terminals (OCT's) meant for FSOC, currently, there is a gap in knowledge regarding mobile OCT's. Therefore the focus of this work has centered on the expansion of knowledge, by investigating the achievable performance of a mobile OCT placed on board a Royal Netherlands Navy frigate-size ship. To summarize, this thesis has centered on answering: What is the strategy to simulate and control the effects that ship-wave disturbances have on a mobile Dual-Stage Optical Communications Terminal to determine if the achievable pointing precision is sufficient for performing Free Space Optical Communications in naval environments?

To answer this a simulation strategy has been employed to divide the work required to answer this question into 3 main steps. Besides this strategy, literature research has been performed to determine suitable system and sub-system requirements in close cooperation with the customer and research partner the Royal Netherlands Airforce (Koninklijke Luchtmacht). The 3 key requirements are as follows: **PATR-01:** The total angular (static) and angular-rate (dynamic) Root Mean Square (RMS) pointing jitter error, shall not exceed 50 [μrad] / [$\mu\text{rad}/\text{sec}$]. **PATR-01-01:** The precise pointing actuator model shall approximate the dynamics and performance characteristics of the Optics in Motion OIM5001 Fast Steering Mirror (FSM). **PATR-01-02:** The coarse actuator model shall approximate the dynamics and performance characteristics of the PlaneWave instruments L-350 direct drive telescope mount.

The first step regarding the implementation of the simulation strategy has been the incorporation of the recordings from an inertial measurement system (sampling rate of 200 Hz) on board a Royal Netherlands frigate-sized ship, into the performance analysis. These measurements provide the angular and angular-rate disturbances, which the dual-stage communications terminal would require to compensate for. These disturbances are of a stochastic nature, since (as of writing) there are limited

time-based simulation tools that can accept signals/disturbances of a stochastic nature. The decision was made to perform the analysis in the frequency spectrum, where the disturbance signals are represented as Power Spectral Density (PSD) functions.

The second step involved the development of a simulation that is capable of approximating the performance characteristics of a dual-stage mobile OCT. To complete this step, the focus centered on creating 3 separate simulation models, each allowing for the expansion of the capabilities of the previous models. These include the Precise Pointing Actuator System (PPAS) that centered on recreating the closed-loop system dynamics of the FSM OIM5001, as an LTI spring damper system, paired with a PID controller and a Quadcell Photodiode (QPD) tracking sensor sampling delay. The QPD tracking sensor errors are included in the analysis in the form of an additional PSD disturbance signal.

The Coarse Pointing Actuator System (CPS) centered on modeling the L-350 gimbal telescope mount, as an LTI spring damper system, paired with a PI controller and the same QPD sampling delay. Lastly, the 3rd model (DSAS) centered on combining the previous models into a combined dual-stage chief-deputy controller configuration, where the chief stage (PPAS) is being actively off-loaded by the deputy stage (CPAS).

The last step of the simulation strategy has been the determination of the achievable performance of the 3 simulation models. To achieve this the Dynamic Error Budgeting (DEB) design analysis has been utilized, as it allows for the determination of the performance of a system, to which stochastic disturbance signals are applied.

From the performance results, it is possible to argue that individual actuator stages are unsuitable for FSOC in naval environments. While the performance results of the PPAS simulation model result in pointing angle and angle rate errors of $\approx 24.6\%$ and $\approx 43.6\%$ below PATR-01. The angular range requirements go beyond those achievable by the FSM actuator standing at $\approx \pm 9.06$ degrees, while the actuator specification states a maximum range of ± 3 degrees [38].

While the performance results of the CPAS indicated achievable actuator effort and controller torque requirements, the system is incapable of complying with the pointing requirements remaining $\approx 234.6\%$ above the allowable pointing angle error requirement and $\approx 234.6\%$ above the allowable pointing angle-rate error.

With an understanding of the performance limitations of individual stages, it is now possible to motivate the need for a dual-stage optical communications system. Therefore it is now relevant to discuss the performance results of the DSAS, which are comprised of multiple controller bandwidth configurations. Notable from these results are the highest performance configurations with chief and deputy controller configurations of 398/500 Hz and 765/500 Hz. Both of these are capable of complying with the pointing requirements each remaining $\approx 5.5\%$ and $\approx 46.3\%$ below the pointing angle rate, while both achieving to remain $> 90\%$ under the pointing angle requirement. At the highest two performance configurations, it can be seen that the deputy controller requires $\approx 12 N * m$ and has a power consumption of $\approx 94 W$, which is an increase from the base configuration of $\approx 66.2\%$ and $\approx 412.8\%$ both still falling within the actuator specifications. Another relevant outcome is the significant reduction in angular range requirements of the chief actuator, where for the highest performance configurations it is only required $1.2 - 1.4 mdegrees$ of range.

With the DSAS performance results, it is of relevance to verify and back up such findings, this end 3 verification steps have been performed. The first step involved the demonstration of the correct functioning of the frequency analysis, this has been obtained by passing a test signal with known properties through the entire frequency analysis pipeline. It was then possible to determine that the estimated properties differ from their original properties by only an average of 2%. The second step involved verifying the control design of the DSAS, to achieve this the transfer functions developed under this work, were compared against equivalent transfer functions from research papers, which were found to be equivalent to each other. The last step involved verifying the performance results obtained from the DEB design approach, to achieve this an alternative time analysis simulation has been utilized to

reproduce the performance results. The difference between the results obtained from the alternative simulation methods and DEB analysis remains at a margin of 1.3 – 2.8 %.

In conclusion, it is possible to argue that for that a Dual-Stage optical communications terminal with a chief-deputy controller configuration of 398/500 Hz or 765/500 Hz can be capable of performing FSOC in naval environments by fulfilling **PATR-01** pointing requirements and **PATR-01-01**, **PATR-01-02** actuator specification sub-requirements, with a preference for the higher-bandwidth configuration.

Regarding future work, it is possible to recommend extending the sampling frequency to 10-20 KHz for the inertial measurements, as this could allow to recreate phenomena that could be occurring at higher frequencies. Another crucial recommendation would be the expansion of the simulation models into specific models obtained from system identification, as these could potentially improve the estimates regarding torque and power requirements of the deputy stage of the DSAS, which has been approximated via simple geometry.

Contents

Preface	i
Summary	ii
1 Introduction	1
1.1 Background	1
1.1.1 Pointing, Acquisition & Tracking subsystem	3
1.2 Research objective	4
1.2.1 The challenge	4
1.2.2 Goal of the research	5
1.3 Scope of work	6
1.4 Structure of the report	6
2 State of the art	8
2.1 PAT system design and mission requirements	8
2.1.1 Types of fine PAT mechanisms	11
2.1.2 PAT system requirements	15
2.2 PAT modeling sub-system requirements	16
2.3 Ship-wave motion & relevant previous research	17
2.3.1 Relevant previous research	18
3 Ship-wave motion frequency analysis	21
3.1 Discrete Fourier Transform algorithm	21
3.2 Power Spectral Density functions from sub-data sets	23
3.3 Cumulative Power Spectrum function	24
4 Actuator control loop models	26
4.1 Precise Pointing Actuator System control model	26
4.1.1 FSM actuator model	27
4.1.2 QPD- Tracking sensor and sampling board	29
4.1.3 PID controller design and tuning	32
4.2 Coarse Pointing Actuator System simulation model	35
4.2.1 Coarse Actuator model	36
4.2.2 PID controller design	38
4.3 Dual-Stage Actuator System simulation model	40
4.3.1 Dual-Stage Actuator System model	40
4.3.2 Augmented Open loop and Closed loop response	41
4.3.3 Augmented Open Loop and Loop sensitivity response	42
4.3.4 Comparison of loop sensitivity functions	43
4.3.5 Simunlink model for actuator off-loading verification	44
5 Dynamic Error Budgeting analysis of the simulation models	46
5.1 PPAS and CPAS Dynamic Error Budgeting analysis summary of results	46
5.2 DSAS Dynamic Error Budgeting analysis	47
5.2.1 DSAS DEB equations	48
5.3 Summary of Dynamic Error Budgeting analysis results and requirements verification . .	49
6 Verification of results	51
6.1 Frequency processing pipeline	51
6.2 Dual-stage closed loop and sensitivity transfer functions	53
6.3 Dynamic Error Budgeting frequency response	54
7 Conclusion	56

8 Recommendations for future work	58
References	59
A Ship-wave inconsistencies with the data set	63
A.1 Inconsistencies with the original data set	63
A.1.1 Linear interpolation as a possible fix to sampling time artifacts	65
A.1.2 Narrowing the original total data set into smaller sets	67
A.2 Creation of data sub-data sets	67
A.3 Windowing and Zero padding for pre-processing of the data subsets	68
A.3.1 Compensation for decrease in resulting magnitude as a result of applying a win- dowing function	68
B Average Ship-Wave motion/interactions frequency plots	72
C Consistent sampling step-size search algorithm	76
D Pre and post-processing Digital Signal Processing algorithms and main script	78
E Additional information, plots and code from actuator loop models chapter	87
E.1 Precise Pointing Actuator System accompanying plots, descriptions and code	87
E.1.1 QPD tracking error estimation model description	87
E.1.2 QPD tracking error estimation model code	88
E.1.3 Augmented Open Loop and Closed-Loop response	90
E.1.4 Augmented Open loop and loop sensitivity function	91
E.2 Coarse Pointing Actuator System accompanying plots and descriptions	92
E.2.1 Estimation of rotational moments of inertia of the Coarse Pointing Actuator System	92
E.2.2 Elevation open loop response	94
E.2.3 Augmented Open loop and Closed loop response	94
E.2.4 Augmented open loop and Loop sensitivity response	95
F System identification attempt for approximating a L-350 actuator transfer function	97
F.1 Attempt to perform system identification on the Planewave L-350	97
F.2 Input/Output Planewave API code	99
G Dynamic Budget Error Analysis supplementary information	104
G.1 Brief introduction to Dynamic Error Budgeting analysis	104
G.2 PPAS Dynamic Error Budgeting analysis	106
G.2.1 PPAS DEB equations	106
G.2.2 PPAS DEB pointing angle error, controller torque and actuator angular effort results	107
G.2.3 PPAS DEB pointing angle-rate error, controller power and actuator angular-rate effort results	113
G.3 CPAS Dynamic Error Budgeting analysis	118
G.3.1 CPAS DEB equations	118
G.3.2 CPAS DEB pointing angle error, controller torque and actuator angular effort results	119
G.3.3 CPAS DEB pointing angle-rate error, controller power and actuator angular-rate effort results	124
G.4 DSAS DEB analysis results and figures	129
G.4.1 DSAS DEB pointing angle error, controller torque and actuator angular effort results	129
G.4.2 DSAS DEB pointing angle-rate error, controller power and actuator angular-rate effort results	133
H Dynamic Budget Error Analysis code	136
I Verification results full-size plots and Code	141
I.1 Frequency pipeline verification full-size plots	141
I.1.1 Frequency pipeline verification code	146
I.2 Dual-stage transfer function control loop models	148
I.3 Simulink block simulation diagram, time analysis plots and DEB frequency response plots	149
I.3.1 DEB frequency response verification code	152

List of Figures

1.1	Possible communication link applications of FSO by [45]	1
1.2	Earth surface spot size comparison between radio frequency Ka-band and optical communications for Geo-stationary and Low-earth orbit satellites, images taken from [9].	3
1.3	Acquisition modes sequence approach for a beaconless Master-Slave algorithm [58]	4
1.4	Classification of different types of waves based on their respective periods. Note that the relative amplitude is indicated by the curve [40]	5
1.5	Report structure in which the cascaded relationship between chapters is highlighted by blue arrows.	6
2.1	Main contributors to initial Uncertainty Area [24]	9
2.2	Two main approaches for spiral scans, a) corresponds to a continuous spiral approach, while b) is a step spiral approach [24].	9
2.3	Total single-scan acquisition time on the vertical axis, while the horizontal axis corresponds to the FOU, for different pointing errors [24]	10
2.4	Differences in pointing errors [54].	11
2.5	2 degrees of freedom typical gimbal configuration, mounted to a vertical base frame [28]	12
2.6	A FSM capable of multiple Degrees Of Freedom (DOF) by mechanical mirror rotation utilizing voice-coil actuators [56]	12
2.7	Schematic figure of a MEMS DM, AO mechanism. The silicon membrane is actuated by an array of PZT actuators [67]	13
2.8	LCs in a transmissive configuration, which allows for spatial light modulation of an incident beam [25]	14
2.9	Standard notation and sign convention of ship-wave interaction/motion [47]	17
2.10	Pitch rate motion with an amplitude in degrees plotted over sampled time in seconds (Total sample duration of 9 Hours and 37 minutes)	18
2.11	Illustration/rendering of the LLCD inertially stabilized telescoped by [3]	19
2.12	FSO OCT terminal onboard the USS Bonhomme Richard for the Trident Warrior, courtesy of [51].	20
3.1	Result of applying the DFT algorithm to the pitch rate Ship-Wave motion data set of an individual interval	22
3.2	Plot of the resulting average PSD of 10 intervals each with durations of 180-200 seconds, obtained from the pitch-rate ship motion data set. N.B. the amplitude on the vertical axis is in $[degrees/sec/Hz]^2$	24
3.3	Result of the cumulative integral function of the average PSD function of the pitch rate S.W. motion	25
4.1	PPAS control loop logic architecture	27
4.2	OIM5001 FSM [39]	28
4.3	FSM actuator model frequency response Bode plot. Note that the first natural frequency is at $27 Hz$	29
4.4	Example of a pair of Quadrant PhotoDiode InGaAs tracking sensors [13]. The sensor on the left has a 2mm quadrant element while the right sensor has a 1mm quadrant element	29
4.5	Frequency response of the QPD-tracking sensor time delay function	31
4.6	PSD estimate of the tracking error, plotted for the same frequencies as the ones of the Ship-Wave IMS data set	32
4.7	Simulink model of the PPAS including main control elements loaded as transfer functions in a separate Matlab script	33

4.8	Tuned step response of the simulated PPAS system with tuned controller. Settling time is plotted with a 2% of overshoot	34
4.9	Resulting open loop system (with controller along the forward path) response Bode plot	35
4.10	Overview of the control design architecture of CPAS	36
4.11	PlaneWave instruments L-350 Direct Drive mount approximation via rectangular boxes	37
4.12	Bode plot of the frequency response of the CPAS azimuth actuator transfer function	38
4.13	Simulink model of the CPAS control system for gain tuning	39
4.14	Tuned step response of the CPAS actuator control system. Note that the settling time is based on reaching 98% of the final desired value	39
4.15	Dual-Stage Actuator System Control Logic architecture	40
4.16	Augmented open loop and closed loop frequency responses of the DSAS shown as Bode plots	42
4.17	Augmented open loop and loop sensitivity frequency responses of the DSAS shown as Bode plots	43
4.18	Bode plot of the comparison between the multiple sensitivity functions of the PPAS, CPAS and DDAS	43
4.19	Simulink model of the DSAS incorporating a step response and tracking sensor disturbances	44
4.20	Step response of the DSAS simulink model response, shown at the top plot is the total displacement of the system, middle plot represents the chief actuator displacement and lastly, the bottom plot represents the deputy actuator displacement	45
5.1	DSAS DEB analysis loop logic	47
A.1	Artifacts in the sampled time data of the Pitch Rate motion	64
A.2	Variance in time step size over the total sampling number	65
A.3	Constant time-step via linear interpolation of the pitch rate data series	66
A.4	Comparison between raw (un-modified) and "clean" (linearly interpolated) data sets in the frequency domain, of the pitch rate ship motion	66
A.5	Terminal output of the constant_sampling_intervals function	67
A.6	Signal with an interval duration of ≈ 47.1 seconds, with constant sampling rate, utilizing the pitch-rate ship motion data. The left side plots display the time-varying data before and after the application of the Hamming window functions, while the right side plots show the data in the frequency domain.	69
A.7	Signal with an interval duration of ≈ 47.1 seconds, with constant sampling rate, obtained from the pitch-rate ship motion data set. The plots on the left side correspond to time domain data, while plots on the right side correspond to frequency domain data.	71
B.1	Roll Ship-Wave motion IMU frequency data set in logarithmic scale for both axes	72
B.2	Pitch Ship-Wave motion IMU frequency data set in logarithmic scale for both axes	73
B.3	Yaw Ship-Wave motion IMU frequency data set in logarithmic scale for both axes	73
B.4	Roll rate Ship-Wave motion IMU frequency data set in logarithmic scale for both axes	74
B.5	Pitch rate Ship-Wave motion IMU frequency data set in logarithmic scale for both axes	74
B.6	Yaw rate Ship-Wave motion IMU frequency data set in logarithmic scale for both axes	75
E.1	Open and Closed loop Bode plot of the frequency response of the designed PPAS	91
E.2	Augmented open loop and Loop Sensitivity of the PPAS	92
E.3	Locations of the approximated distances of the center of mass of the 3 main blocks representing the I-350 mount	93
E.4	Bode plot of the frequency response of the CPAS elevation actuator transfer function	94
E.5	Augmented Open loop and Closed loop frequency responses of the CPAS	95
E.6	Plot of the frequency response of the Loop Gain and Loop sensitivity transfer functions	96
F.1	Third verification test, where the lines in colour represent fitted LTI transfer function models, while the black line indicates the original output data. The bottom plot showcases the original input commands	99

G.1	Schematic of a closed loop system with various disturbances acting in it [36]. Note the blue-dashed boxes correspond to the PSD of a signal.	105
G.2	PPAS DEB analysis loop logic	106
G.3	DEB analysis results of the PPAS system pointing angle error ASD and CAS	108
G.4	DEB analysis results of the PPAS controller torque ASD and CAS	110
G.5	DEB analysis results of the PPAS actuator effort for angular range ASD and CAS	112
G.6	DEB analysis results of the PPAS pointing angle rate error ASD and CAS	114
G.7	DEB analysis results of the PPAS controller power ASD and CAS	115
G.8	DEB analysis results of the PPAS actuator angle rate effort ASD and CAS	117
G.9	CPAS DEB analysis loop logic	118
G.10	DEB analysis results of the CPAS actuator pointing angle error ASD and CAS plots	120
G.11	DEB analysis results of the CPAS controller torque ASD and CAS plots	122
G.12	DEB analysis results of the CPAS actuator angular effort ASD and CAS plots	123
G.13	DEB analysis results of the CPAS pointing angle rate ASD and CAS plots	125
G.14	DEB analysis results of the CPAS controller power ASD and CAS plots	126
G.15	DEB analysis results of the CPAS actuator angular rate effort ASD and CAS plots	128
G.16	DEB analysis results of the DSAS total actuator pointing angle error ASD and CAS plots	130
G.17	DEB analysis results of the DSAS chief and deputy controller torque ASD and CAS plots. Top plots correspond to the chief controller, while bottom ones correspond to the deputy controller	131
G.18	DEB analysis results of the DSAS chief and deputy actuator angular effort ASD and CAS plots. The plots at the top side of the figure correspond to those of the chief actuator, while those at the bottom correspond to those of the deputy actuator.	132
G.19	DEB analysis results of the DSAS total pointing angle rate ASD and CAS plots	133
G.20	DEB analysis results of the DSAS Chief and Deputy controller power ASD and CAS plots	134
G.21	DEB analysis results of the DSAS chief and deputy actuator angular rate effort ASD and CAS plots	135
I.1	Input test time domain signal at the top plot, while the bottom plot corresponds to the output power spectral density function of the input test signal. Test signal with a total duration of 1 second.	141
I.2	Input test signal after applying a hamming window function and zero padding, with a signal duration of 1 second.	142
I.3	Input test time domain signal at the top plot, while the bottom plot corresponds to the output power spectral density function of the input test signal. Test signal with a total duration of 10 seconds.	142
I.4	Input test signal after applying a hamming window function and zero padding, with a signal duration of 10 seconds.	143
I.5	Input test time domain signal at the top plot, while the bottom plot corresponds to the output power spectral density function of the input test signal. Test signal with a total duration of 50 seconds.	143
I.6	Input test signal after applying a hamming window function and zero padding, with a signal duration of 50 seconds.	144
I.7	Input test time domain signal at the top plot, while the bottom plot corresponds to the output power spectral density function of the input test signal. Test signal with a total duration of 100 seconds.	144
I.8	Input test signal after applying a hamming window function and zero padding, with a signal duration of 100 seconds.	145
I.9	Input test time domain signal at the top plot, while the bottom plot corresponds to the output power spectral density function of the input test signal. Test signal with a total duration of 200 seconds.	145
I.10	Input test signal after applying a hamming window function and zero padding, with a signal duration of 200 seconds.	146
I.11	Dual Stage control design diagram by [6]	148
I.12	Sensitivity block diagram by [32]	148
I.13	Simulink model utilized to capture the time analysis response to disturbance spectra	149

I.14	Time plots of the output signals from the Simulink model angle analysis	149
I.15	Time plots of the output signals from the Simulink model angle-rate analysis	150
I.16	Power Spectral Density resulting from the pointing angle time analysis	150
I.17	Power Spectral Density resulting from the pointing angle-rate time analysis	151
I.18	Power Spectral Density resulting from the pointing angle frequency response	151
I.19	Power Spectral Density resulting from the pointing angle rate frequency response . . .	152

List of Tables

2.1	Mission requirements	11
2.2	Comparison of the main specifications of PAT mechanisms	15
2.3	PAT system requirements	15
2.4	PAT modeling sub-system requirements	17
5.1	PPAS and CPAS DEB analysis performance results	47
5.2	DSAS DEB analysis performance results for multiple controller bandwidths	50
5.3	Verficiation against critical requirements	50
6.1	Frequency processing pipeline test signal variance estimate results	52
6.2	Results summary of the performance results of the time analysis and DEB analysis	55
E.1	Specification of the distances utilized to determine the rotational moments of inertia	93
G.1	PPAS total system pointing error DEB analysis performance results	109
G.2	PPAS Controller torque DEB analysis performance results	110
G.3	PPAS Actuator effort angular range DEB analysis performance results	113
G.4	PPAS Pointing Angle Rate error DEB analysis performance results	114
G.5	PPAS Controller Power DEB analysis performance results	116
G.6	PPAS Actuator Angle Rate effort DEB analysis performance results	117
G.7	CPAS actuator pointing angle error DEB analysis performance results	121
G.8	CPAS Controller torque DEB analysis performance results	122
G.9	CPAS actuator angular effort DEB analysis performance results	124
G.10	CPAS Pointing Angle Rate DEB analysis performance results	125
G.11	CPAS Controller Power DEB analysis performance results	127
G.12	CPAS Actuator Angular Rate effort DEB analysis performance results	128
G.13	CPAS actuator pointing angle error DEB analysis performance results	130
G.14	DSAS Chief and Deputy Controller torque DEB analysis performance results	132
G.15	DSAS chief and deputy actuators angular efforts DEB analysis performance results	133
G.16	DSAS Total Pointing Angle Rate DEB analysis performance results	133
G.17	DSAS Chief and Deputy Controller Power DEB analysis performance results	134
G.18	CPAS Actuator Angular Rate effort DEB analysis performance results	135

Introduction

1.1. Background

Wireless Optical Communication (WOC) has recently become a viable complementary technology in addition to existing Radio Frequency Communication (RFC). One of its critical advantages over existing RFC is its significant increase in carrier frequency, since "it is a well-known fact that an increase in carrier frequency increases the information-carrying capacity of a communication system" [24]. WOC has several critical advantages over existing RFC, namely tariff-free bandwidth allocation, low on-board power requirements and improved channel security. Due to the rise in global tensions, the last advantage can prove a crucial asset for military applications.

Recent advances in Vertical-Cavity Surface-Emitting Lasers (VCSEL's), Avalanche Photodiodes (APD's), Adaptive Optics, Real-Time Control, Active Optics, etc, all have allowed addressing the challenges of WOC, increasing the Technology Readiness Level (TRL), thus driving the increase in practical applications of such technology. With such developments, WOC has been identified to hold many commercial and military applications, within such a variety of applications, Ground to Low-Earth-Orbit (LEO) applications are of special interest for this work. WOC for ground-to-LEO or LEO-to-ground applications is often referred to as Free Space Optical Communication (FSOC). To visually represent some of the potential applications offered by FSOC, see Figure 1.1.

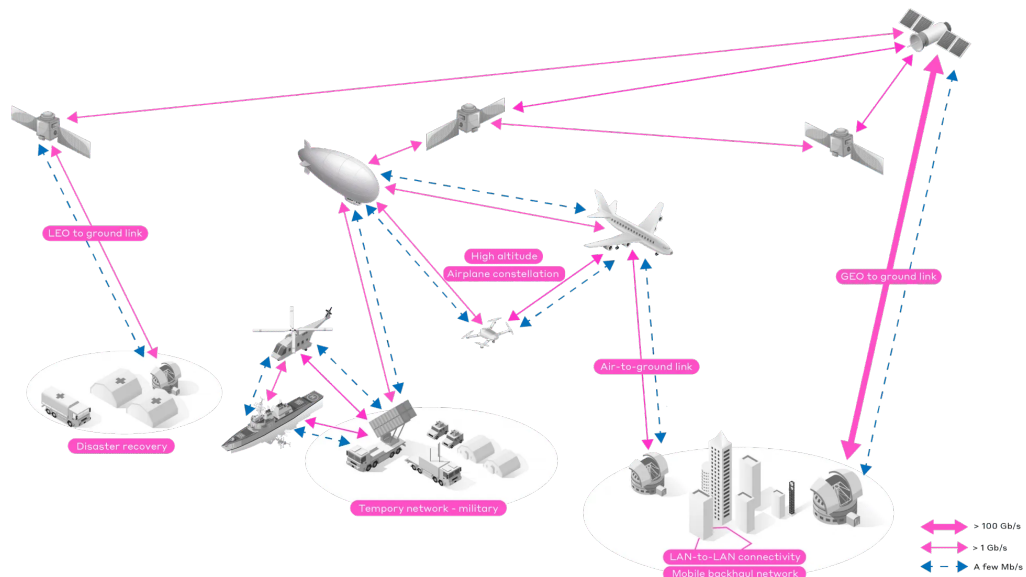


Figure 1.1: Possible communication link applications of FSO by [45]

FSOC systems are typically made up of 3 main functional elements these are; the transmitter, the transmission channel -and the receiver. The fundamental working principle is based on modulating information of the signal into an optical carrier, this is performed at the transmitter end. Once the signal is modulated it is then propagated through space and Earth's atmosphere, serving as the transmission channel. The communication is complete once the signal is received and demodulated at the receiver end, with the help of an Optical Communications Terminal (OCT).

Typical challenges of performing FSO communications are related to disturbances introduced by Earth's atmosphere, as well as strict pointing requirements of both receiver and transmitter ends. The strict pointing requirements are a result of working with optical carrier waves with high frequencies (i.e. low wavelength sizes). The International Commission on Illumination [18] has classified optical radiations into three main categories, these include IR-A (700 to 1400 nm), IR-B (1400-3000 nm) and IR-C (3000 nm to 1 mm). For further information about the classifications of the optical wavelength spectrum, readers are referred to the references contained in the paragraph.

To be able to perform FSOC it "requires a Line Of Sight (LOS) connection between transmitter and receiver for the propagation of information from one point to another" [24]. Achieving LOS for an FSOC system is one of the most challenging aspects, since "near-infrared laser wavelength's narrow beam divergence leads to the requirements for precision beam pointing by both ends of the link, often on the order of a few micro-radians"[35].

To provide the reader with a physical representation of the pointing requirements, it is possible to compare the estimates of the relative spot size (at Earth's surface) taken by a radio frequency signal at 30 GHz (Ka-band) to the one of an optical signal at 1550 nm (193.4 THz). It is possible to estimate the apparent spot size via Equation 1.1, which performs an estimate over the Half-Power Beam Width (HPBW), which is a measure of the angular width of the main lobe of an antenna radiation pattern, further information about the derivation of the equation and its assumption can be found on [2].

$$\theta_{HPBW} \approx \frac{70 * \lambda}{D_{antenna}} \quad (1.1)$$

Where λ corresponds to the optical wavelength size of the signal and $D_{antenna}$ corresponds to the diameter of the antenna. Next by utilizing trigonometry and a small angle approximation, it is possible to relate the HPBW to an estimated guess of the relative spot size of the radio/optical signal at the earth's surface via Equation 1.2

$$W_{spot\ size} = 2 * R_{sat} * \tan\left(\frac{\theta_{HPBW}}{2}\right) \quad (1.2)$$

Where R_{sat} is the distance from Earth's surface to the satellite. Making use of both equations it is then possible to roughly estimate the relative spot size for radio-frequency and optical signals from satellites placed at different orbits. Visible in Figure 1.2, are three images relating three different spot sizes, the left figure corresponds to a radio-frequency signal (at 30 GHz) sent from a Geostationary orbit at an approximate distance of 36000 Km and with an antenna size of 2 $meters$. The center image corresponds to the same Geostationary satellite but for an optical signal at a 1550 nm wavelength and assuming a smaller satellite antenna of 0.1 $meters$. The right figure corresponds to a low-earth orbit satellite at an approximate distance of 1000 Km from the surface sending an optical signal at 1550 nm wavelength and as well assuming a smaller satellite antenna of 0.1 $meters$

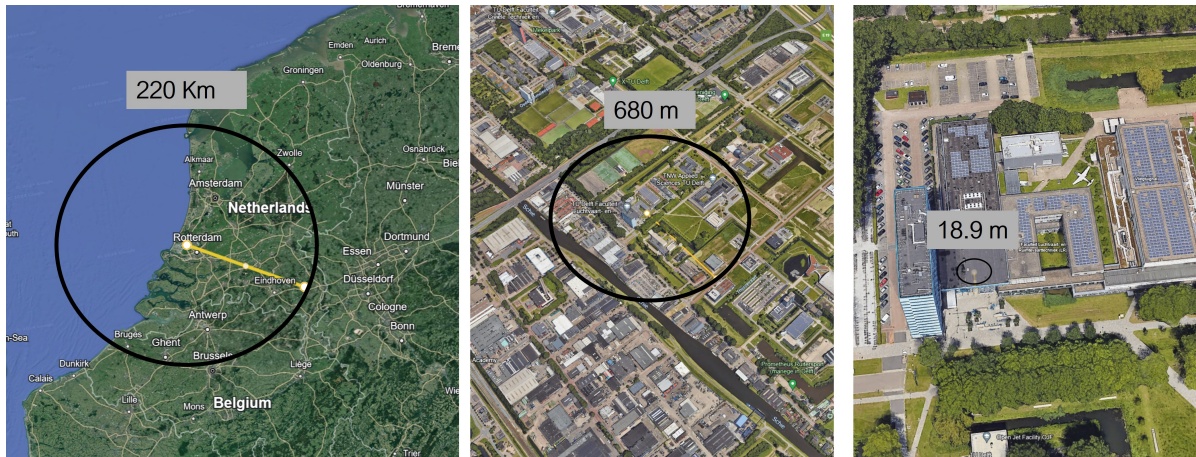


Figure 1.2: Earth surface spot size comparison between radio frequency Ka-band and optical communications for Geo-stationary and Low-earth orbit satellites, images taken from [9].

1.1.1. Pointing, Acquisition & Tracking subsystem

A functional FSOC system must possess precise and efficient pointing of narrow optical beams at both receiver and transmitter communication ends. To achieve this, the Pointing, Acquisition and Tracking (PAT) subsystem follows a sequential approach, which targets the gradual reduction in pointing errors via the employment of interconnected coarse and fine pointing mechanisms, as "Implementing an optical link involves an orderly process that reduces the initial pointing error and enables efficient use of the narrow beam" [35].

As of this writing, there are two main approaches regarding the type of terminal/receiver design of the PAT subsystem, these include a beacon and a beaconless design. For this work, the efforts are concentrated on a beaconless design, in fulfillment of the requirement OCT-10 outlined in [1], which defines interoperability specifications, for optical communications systems employed by the Space Development Agency (SDA) and its partners. Complying with this set of standards is necessary, as it conforms to part of the mission requirements of this research. The mission requirements are discussed in further detail in chapter 2.

The process of acquisition starts with the co-alignment of the LOS of both the receiver and transmitter. Utilizing precise positional information of both terminals, coalignment can be achieved by direct open-loop pointing. The estimated pointing error between terminals is referred to as an Uncertainty Cone (UC). After both terminals point in the estimated direction of their respective counter terminal, the spatial acquisition process starts by engaging the master-slave mode. During this mode, the master terminal emits light and spirals the UC with its low divergent transmit beam, while the slave terminal should receive a light pulse once per scan.

The receiving (slave) terminal will successively reduce its UC with each light pulse detection, this allows for correcting the pointing resulting in a reduction in the step size. After a sufficient number of scans, the master terminal stops scanning and transmitting, at this point the slave terminal has been correctly aligned to the master terminal, then the orders are reversed where the former slave terminal takes the role of the master terminal. This allows for re-scanning and re-alignment between terminals, sequentially decreasing the size of the UC, until a certain alignment threshold is reached, at this point, spatial tracking has been accomplished. Figure 1.3 provides a visual representation of this process.

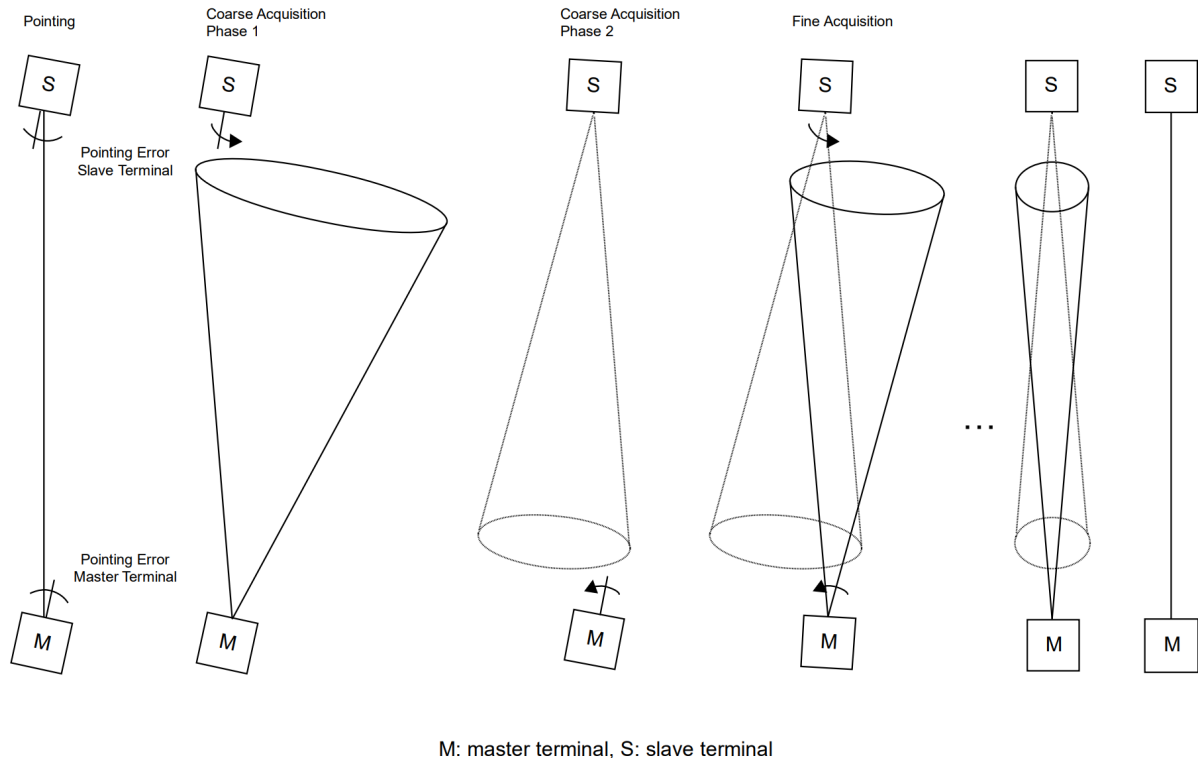


Figure 1.3: Acquisition modes sequence approach for a beaconless Master-Slave algorithm [58]

Now with the initial acquisition complete, the next step involves the synchronization of each platform clock, often achieved by relaying predetermined bit patterns. Next is the continuous correction of the boresight (i.e. alignment between transit and receiver laser terminals), typically achieved with end-to-end power-on-target algorithms. The last step involves continuous maintenance of the communication link, to achieve this continuous corrections for the relative velocity between terminals and the finite velocity of light need to be performed.

1.2. Research objective

1.2.1. The challenge

As discussed in the previous section, one of the main challenges regarding effective FSOC is complying with strict pointing and tracking requirements. While the focus of this research lies entirely on the OCT receiver communication end, the difficulty of complying with such requirements is further increased, by operating in a dynamic environment such as the one on board a ship. What could be expected is that significant disturbances are introduced into the OCT, the extent to which those disturbances affect the accurate and precise pointing of the PAT subsystem is not yet known, thus critical to determine during this research.

To better understand the challenge posed by introducing ship-wave interactions, it is then useful to delve into the expected frequency spectrum of such interactions. The absolute wave spectrum has been thoroughly investigated, where it has been possible to classify the types of waves based on their frequency range "Virtually all the energy of the spectrum is contained in two of the period ranges: the ordinary gravity waves and the ordinary tides. These are the ones that can be noticed with the naked eye, the ones that have affected mankind since the dawn of history"[40]. To provide a clear overview of what type of frequency ranges, Figure 1.4 provides an overview of the frequency and energy spectrum of ocean waves.

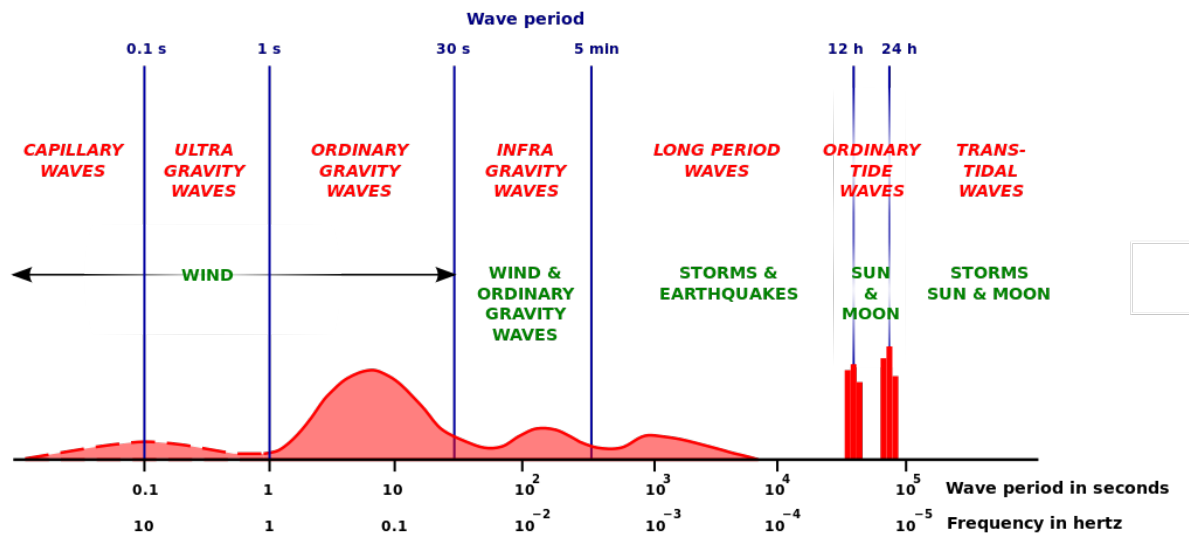


Figure 1.4: Classification of different types of waves based on their respective periods. Note that the relative amplitude is indicated by the curve [40]

As visible in Figure 1.4, it is possible to remark that the great majority of energy (amplitude) lies between 0.01 to 1 Hz frequency range. For this frequency range, only the lower frequency range would be relevant as "Basically, any ship is a low-pass wave filter and, like in this case, the high-frequency tail may be "cut off" since the particular vessel does not respond to waves with a frequency higher than about 0.20 Hz " [43].

1.2.2. Goal of the research

Now having covered the general spectrum of the frequency range at which the ship-wave interactions are expected to take place, it is then possible to propose a way forward for the overall direction of the research. Due to the suspected low-frequency range of the main disturbances, which are of large amplitude but long periods, a dual-stage PAT system is proposed. The dual stage would consist of a Coarse Pointing Sub-system (CPS) and a Precise Pointing Sub-system (PPS), the aim of splitting the system into dual stages is for the CPS to mainly take care of large amplitude long period disturbances, while letting the PPS take care of compensating for the very short period atmospheric disturbances.

While significant technological progress has been made to mitigate the challenges introduced by Earth's atmosphere, as of the time of writing, progress is still required to address the challenges introduced by operating FSOC (receiver end) in dynamic environments. Such environments are prone to introducing disturbances which tend to hamper the pointing accuracy and precision of PAT systems.

Therefore, the aim of the research is to develop a simulation of a dual-stage control system that best represents the expected behavior of the CPS and PPS. Once the dynamics of the expected behavior are in the simulation, the actual ship-wave interactions/disturbances can be introduced, which will then allow to determine the performance of the PAT system under such dynamic environments. To focus the research in the desired direction, the following main and sub-research questions are posed:

What is the strategy to simulate and control the effects that ship-wave disturbances have on a mobile Dual-Stage Optical Communications Terminal as to determine if the achievable pointing precision is sufficient for performing Free Space Optical Communications in naval environments?

- How to incorporate into the analysis the recorded inertial measurements of the ship-wave motions as a means to incorporate realistic disturbances resulting from operating in naval environments?
- How to develop a simulation that captures the dynamics of the dual-stage mobile optical communications terminal suitable for performing Free Space Optical Communications?

- What are the performance requirements to achieve the required pointing precision for performing Free Space Optical Communications in naval environments?

1.3. Scope of work

The focus of this research lies in recreating the dynamics of a receiver-end OCT intended for FSOC in a simulation. The focus covers as well the incorporation of ship-wave interactions/disturbances measurements into the simulation, as a means to recreate a realistic naval environment in which the receiver-end would be operational. The ship-wave interactions/disturbances are made available due to previous works by [53]. The following aspects are considered as topics falling out of the scope of research:

- Physical testing via the manufacturing of an equivalent/representative receiver-end prototype terminal. The analysis remains contained to analytic means via actuator simulation in MATLAB.
- Recording inertial measurements of ship-wave motions. The research utilizes the aforementioned data set gathered by [53], with a total duration of 9 Hours and 37 min.
- Computer-Aided Design (CAD) drawings or 3d models of the receiver end. The analysis remains concentrated on the creation of closed-loop control simulation models, which utilize transfer functions to capture the actuator dynamics of the receiver-end terminal.
- Correlation between inertial measurements and sea depth. Since the measurements do not contain information on sea depth, the reconstruction of the path of the boat to determine the sea depth falls outside the envisioned work.
- Simulation of the optical path for the precise pointing system. Performing such simulation would require ray propagation methods, which lie outside the scope of investigation.
- Determination of the sources of frequency peaks found in the ship-wave measurements. Since the ship-wave data set has been obtained from a previous project and currently no additional measurements are being performed, determining the sources of the frequency peaks lies outside of the reach of this project.

1.4. Structure of the report

The aim of this section is to provide the reader with an overview of how each chapter is interconnected to each other, as well as to provide a brief summary of the contents that can be found in each chapter.

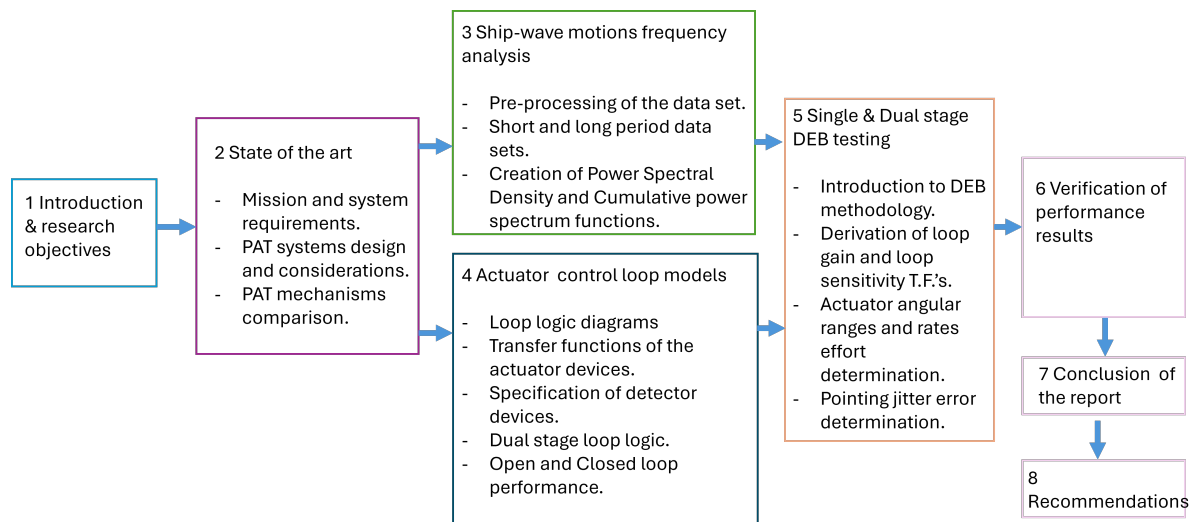


Figure 1.5: Report structure in which the cascaded relationship between chapters is highlighted by blue arrows.

As visible in Figure 1.5, there is a cascaded relationship between chapters, the relationship between chapters is also visible in the figure. Starting with the first chapter, the topic in general terms is introduced, providing general knowledge, that is required to understand the rest of the research, the

research objectives as well as the scope of research are discussed as well in this chapter. Advancing towards the second chapter, it is then possible to arrive at the state of the art, which focuses on discussing general PAT system design and considerations. This chapter also discusses the level of achievable performance with current (as of writing) technology and relevant previous research performed on this subject. Building upon that knowledge the mission and system requirements are introduced as well as motivated. Lastly, differences in PAT mechanism are explored and a selection of them is provided as input for the dual-stage controller design.

Next are the third and fourth chapters which provide critical information for the understanding of the subsequent fifth chapter. The former chapter centers on the frequency analysis of the inertial measurements of the ship-wave motions. In this chapter aspects such as the steps taken for pre-processing of the data set, creation of separate short and long-period data subsets as well as the methods utilized to perform the calculation of the Power Spectral Density and Cumulative Power Spectrum functions are covered. In the latter chapter, the focus lies on the determination of transfer functions that represent the actuator dynamics, specification of the detector performance, derivation of the state space model for the dual-stage controller design. The chapter concludes with a discussion of the closed-loop controller path logic.

The fifth chapter builds on the information covered by the two previous chapters, to focus on the diverse aspects of Dynamic Error Budget (DEB) analysis. Critical to understanding the analysis, the chapter starts by providing a brief introduction to the methodology utilized to perform the analysis. It then covers the Loop Gain and Loop Sensitivity functions of the different control loop logics, providing crucial information about the performance of the different control loops. Following the same logic the Open and Closed loop are subsequently discussed, to provide an idea of the achievable performance of the different control loops and the difference in response of the different actuator systems. With that covered the chapter then goes on to present the results of performing the DEB analysis, as to determine the required actuator angular ranges as well as the rotational rates, as well as the controller effort involved. By utilizing the outcomes of these results, estimates are then given regarding the pointing jitter error achieved by the different setups. This chapter concludes with a summary of the achieved results and a summary of the performance requirements to achieve multiple levels of pointing precision.

The sixth chapter covers the methodology utilized to verify the performance results acquired in previous chapters. It covers the frequency processing pipeline, closed-loop and sensitivity transfer functions as well as the frequency responses acquired via Dynamic Error budgeting. The seventh chapter provides a conclusion to this research by attempting to provide clear and conclusive answers to the research questions posed earlier. Lastly, the seventh chapter provides recommendations for future research work on this topic.

2

State of the art

This chapter aims to provide general information to the reader, as to better understand the rest of the work performed during this research, as well as to provide an overview of the current level of development of relevant technologies, as well as discussing relevant research performed in this subject. The chapter starts by introducing general PAT system design considerations, covering aspects such as the sources of uncertainty, pointing errors, it then concludes this section by outlining the mission requirements. Next, the different types of PAT mechanisms suitable for FSOC are presented and compared to each other, a table with current achievable levels of performance is also provided. The section then motivates briefly the chosen dual-stage configuration, then it proceeds to introduce and motivate the PAT system requirements. The last section of this chapter concludes by discussing the type of ship-wave motions that are to be expected, the initial results of the frequency analysis, and building on the initial results an overview is also provided with respect to relevant research performed in this subject.

2.1. PAT system design and mission requirements

In chapter 1 the challenge of achieving initial acquisition was introduced, where an open-loop pointing procedure is to be utilized by beaconless Optical Communication Terminals (OCT) to systematically reduce the size of the Uncertainty Cone (UC). This section intends to explore further the details involved with initial target acquisition for effective FSOC, as well as outlining the requirements regarding pointing accuracies of OCTs, it then concludes by providing a brief overview of the common mechanism employed in PAT systems design. **Note:** the tracking phase part of PAT systems is considered outside the scope of work.

A critical aspect to consider for PAT subsystems is their initial Uncertainty Area (UA), which directly contributes to the size of the UC. The UA is a critical parameter to consider, since a large UA paired with a single narrow communications beam (characteristic of beaconless PAT designs), can result in impractical acquisition times. Two main sources contribute to the size of the initial UA, the former encompasses the attitude -and ephemeris data that corresponds to the orbital trajectory of the LEO satellite, while the latter corresponds to the gimbal pointing errors of the OCT. To provide an organized overview of the different sources and the magnitude of their pointing error contributions, Figure 2.1 aims to provide an overview of them.

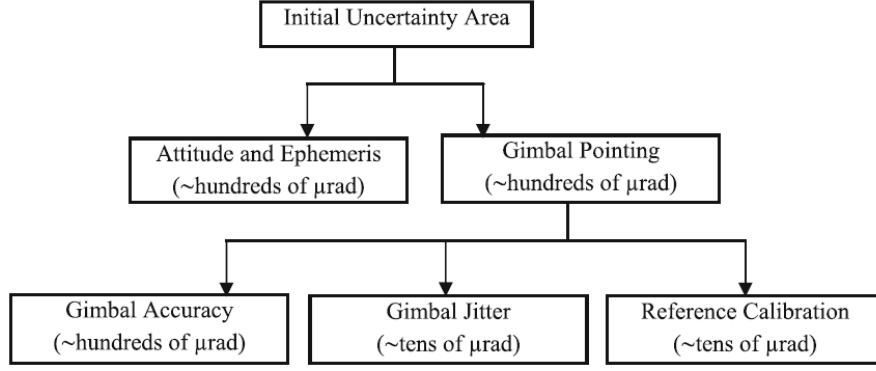


Figure 2.1: Main contributors to initial Uncertainty Area [24]

With an understanding of the multiple sources of pointing errors, next is to discuss the scanning technique, utilized for initial target acquisition. In contemporary literature, is possible to find several acquisition approaches. For this work, spiral scanning is of relevance not only because it is the primary scanning mode required by [1], but also is "the most efficient spatial acquisition technique as it is very easy to implement" [24]. The main characteristics, as outlined in the works by [35], include benefits such as a common laser beam (utilized for both acquisition and communication purposes), a less expensive sensor, no need for defocusing optics, etc. While the downsides include aspects such as the increase in difficulty for eliminating intensity dropouts, rapid scanning commands can place significant stresses on steerable optical elements, etc. Typically there are 2 types of implementations of spiral scanning these include continuous and step, for clarity purposes, Figure 2.2 provides an overview of the differences between them.

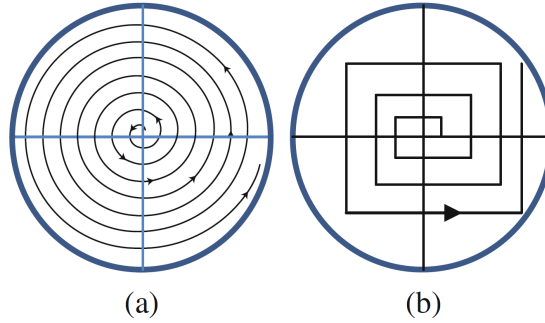


Figure 2.2: Two main approaches for spiral scans, a) corresponds to a continuous spiral approach, while b) is a step spiral approach [24].

Using the approach described in [24], it is possible to describe in a compact form, the trace of the spiral scan, utilizing polar coordinates, as shown in Equation 2.1.

$$r_s = \frac{L_\theta}{2\pi} \theta_s \quad (2.1)$$

Where L_θ represents the step length which is related to the beam divergence angle (θ_{div}) and the overlap factor (F_0), as described in Equation 2.2. Note that θ_s corresponds to the current pointing position.

$$L_\theta = \theta_{div}(1 - F_0) \quad (2.2)$$

From Equation 2.2, is possible to derive 2 main conclusions, the former relates to the fact that increasing θ_{div} , results in a direct linear increase in the trace of the spiral scan, while minimizing the overlap factor, provides an alternative for increasing the trace of the spiral scan, as increasing θ_{div} often requires increases significantly the available transmission power. Next, in Equation 2.3 the scan time from an

initial point at $(0,0)$ to (r_s, θ_s) is presented.

$$T(r_s) \approx \frac{\pi \theta_s^2}{L_\theta^2} \left(T_R + 2 \frac{R}{c} \right) \quad (2.3)$$

Where T_R corresponds to the response time of the receiver acquisition system, R is the link range and C corresponds to the velocity of light in vacuum. By averaging the single-scan acquisition time with respect to the Probability Density Function (PDF) of the satellite position and integrating the area under the curve, it is then possible to arrive at Equation 2.4.

$$T_{SS} = \frac{2\pi\sigma_\theta^2}{L_\theta^2} \left[1 - \left(\frac{\theta_U^2}{2\sigma_\theta^2} + 1 \right) \exp \left(-\frac{\theta_U^2}{2\sigma_\theta^2} \right) \right] \left(T_R + 2 \frac{R}{c} \right) \quad (2.4)$$

In Equation 2.4, the relationship between the multiple components that contribute to the total single-scan acquisition mean-time can be seen, two parameters are of interest, these include the pointing error (σ_θ), and the transmitter Field Of Uncertainty (FOU) (θ_U). To provide an overview of how these parameters impact the total acquisition time Figure 2.3

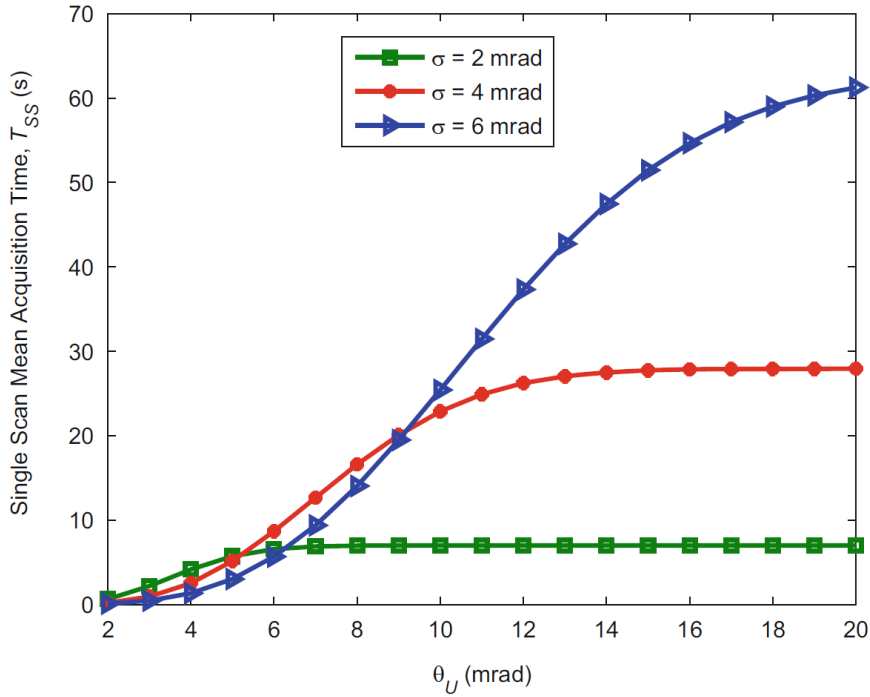


Figure 2.3: Total single-scan acquisition time on the vertical axis, while the horizontal axis corresponds to the FOU, for different pointing errors [24]

Now from Equation 2.4 it is possible to see the importance of a low pointing error paired with a low FOU, as to minimize the total acquisition time and guarantee effective means of communication. Referring back to Figure 2.1, so far the discussion has been centered around static pointing errors and how they impact the acquisition process. However static pointing errors are only half of the story regarding FSOC, the other pointing errors that must be accounted for are regarding dynamic angular pointing jitter errors. These refer "to the uncertainty of the instantaneous direction of the optic axis with respect to the direction of the desired line of sight" [64]. Sources of these types of pointing errors include vibrations introduced by the OCT, scintillation due to atmospheric turbulence, etc. To provide an overview of the differences between static pointing error and dynamic pointing jitter, Figure 2.4 provides a visual representation.



Figure 2.4: Differences in pointing errors [54].

Now with an overview of the general design drivers of PAT systems as well as an understanding of the differences in pointing errors, next is to introduce the mission requirements. As visible in Table 2.1 there are 2 main mission requirements, which outline the objectives to be reached by the system designed in this work. These requirements are the result of performing this work in collaboration with the Royal Netherlands Air Force (Koninklijke Luchtmacht) as well as the overall objective of the thesis.

Table 2.1: Mission requirements

Requirement identifier	Description
MSR-01	The mobile OCT system shall be compatible with the Space Development Agency OCT standard.
MSR-02	The mobile OCT system shall have the capacity to be fitted into and operated from the Royal Dutch Navy (Koninklijke Marine) frigate-sized ships.

2.1.1. Types of fine PAT mechanisms

Fine PAT mechanisms play a crucial role in both static and mobile FSOC systems, especially when they are designed to utilize a narrow communication beam. The fundamental objectives of such mechanisms, are to increase the angular steering speed and accuracy. While there are several types of PAT mechanisms, the focus of this sub-section will be specifically on the ones relevant for mobile FSOC applications, as well as being mature enough such that performance quantities such as their achievable pointing resolution are known. NOTE in compliance with [1] Requirement OCT-007, the scope of this subsection also does not include hybrid RFC-FSOC PAT mechanisms, as OCT terminals shall perform the PAT process without access to real-time side channels for communication and coordination.

The first mechanism to be looked at is gimbal-based PAT mechanisms, which are characterized by mechanically rotary platforms that are controlled by motors. The mechanical rotary joints provide two or three degrees of freedom for the gimbal. "These types of systems are preferred specially fit for transceivers that require a wide angular range of motion" [34]. However, the wide angular range of motion comes at the price of a relatively coarse pointing resolution (large step size). "The large mass of a gimbal is another limitation" [27]. Because of the large-sized servo/stepper motors required to rotate the gimbal, it is typically not suitable for mobile stations with stringent mass limits. To provide a visual overview of such mechanisms, Figure 2.5 provides an overview of a typical 2 degrees of freedom gimbal mechanism.

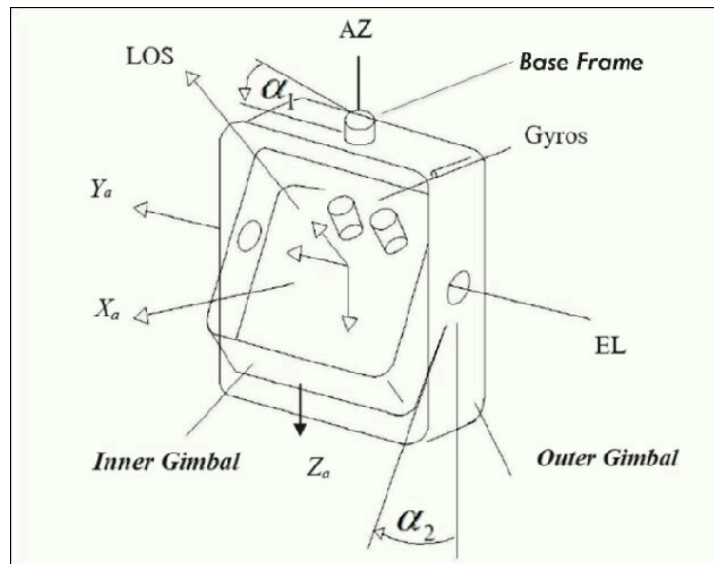


Figure 2.5: 2 degrees of freedom typical gimbal configuration, mounted to a vertical base frame [28]

Next is the discussion of mirror-based PAT mechanisms, these mechanisms are characterized by the employment of Fast Steering Mirrors (FSMs) to achieve stabilization, pointing and tracking. The fundamental working principle is the deflection of the incident beam into the location of the detector. Such mechanisms are characterized by their compactness which results in low power consumption and low mass. "FSMs are light of weight because the mirror they move have a small mass"[25]. The small mirror mass allows for "FSMs to have a high steering speed and fine pointing resolution (e.g., sub- μ rad)" [66]. However there is a major drawback of FSM, it is related to "their limited angular range of motion that may restrict their field-of-view for data acquisition/tracking" [66]. To provide an overview of how such a mechanism could look like Figure 2.6 provides an overview of a voice coil-activated 2 Degrees Of Freedom (DOF) FSM.

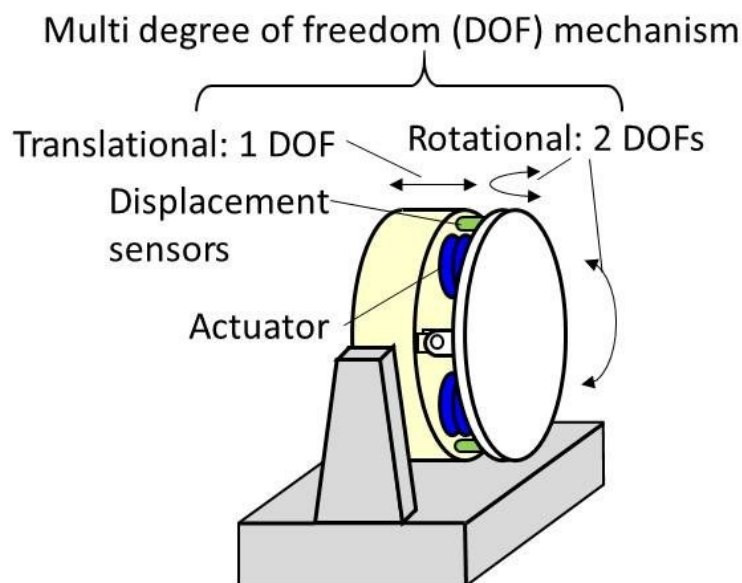


Figure 2.6: A FSM capable of multiple Degrees Of Freedom (DOF) by mechanical mirror rotation utilizing voice-coil actuators [56]

The next PAT mechanism involves Adaptive Optics (AO). AO has an extensive heritage derived from astronomical telescopes, as these mechanisms can compensate "wave-front distortions of optical waves through atmospheric turbulence by using Deformable Mirrors (DMs) and wave-front sensors"[59]

[61]. The main characteristics of AO involve the capability of low & high order beam aberration correction as well as nano-scale beam steering. While this system is typically primarily utilized for the improvement of detector coupling, its nano-scale beam steering capabilities can still form part of PAT system with limited angular range.

The fundamental working principle of AO is the actuation of several individual members along vertical and horizontal arrays to deform a membrane in 3-dimensional space. The nano-scale beam steering capabilities are of special interest, as AO are able to obtain one of the highest position resolutions among all PAT subsystems, they are capable of an "Average Step Size <1 nm (equivalent to a $<n$ rad resolution)" [63]. However, such a high positional resolution, results in an even more restricted angular range when compared to FSMs, as typically "the maximum actuator displacement is in the range of a few μm " [63]. To see an overview of the layout of a typical AO with a Micro-Electro-Mechanical System (MEMS) actuating a DM, refer to Figure 2.7.

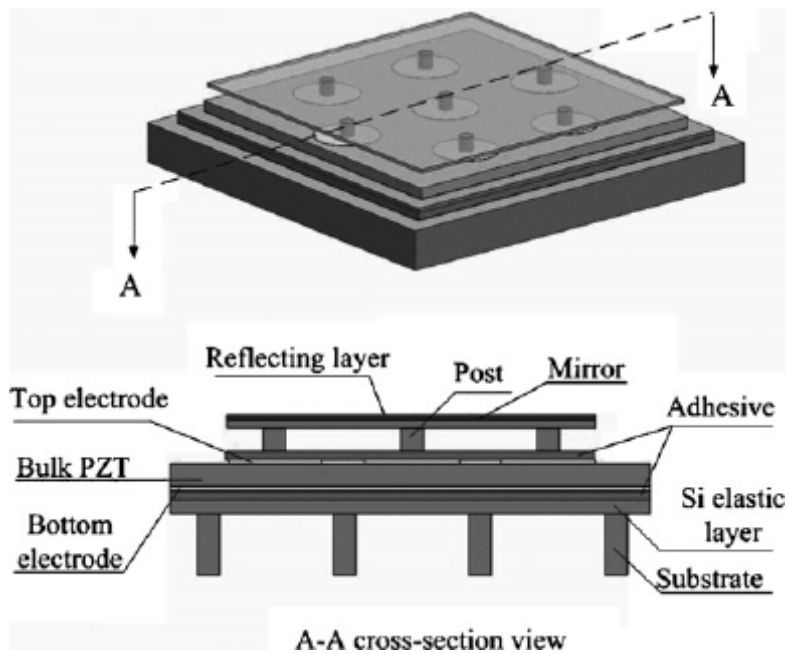


Figure 2.7: Schematic figure of a MEMS DM, AO mechanism. The silicon membrane is actuated by an array of PZT actuators [67]

The last PAT mechanism to be covered includes Liquid Crystals (LCs) based PAT mechanisms. These mechanisms are non-mechanical as they rely on thin electrodes to manipulate the direction of an incident beam. The fundamental principle lies in manipulating a "one-dimensional array of tens of thousands of long thin electrodes to manipulate the amplitude or phase pattern of a light beam" [65]. By applying specific voltages to each of the electrically addressable pixels of the device the refractive index of each of the LCs can be modified, as to produce a modulated wavefront or a diffractive grating which in turn allows for the deflection of the incident beam into a specific direction.

The main advantages of LCs include component affordability, low power consumption, low mass and "the ability of agile redirection of the steering elements, over the mechanical light modulators and conventional DMs" [65], [11]. However, the greatest drawback is their low angular motion "typically in the range of ± 4 mrad" [49]. In Figure 2.8, an overview of a LCs in a transmissive configuration for spatial light modulation of incident light, is provided.

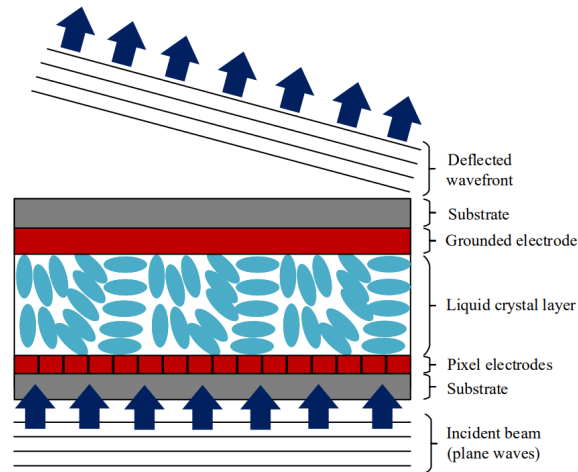


Figure 2.8: LCs in a transmissive configuration, which allows for spatial light modulation of an incident beam [25]

With the general specifications of the relevant PAT mechanism having been discussed, it is now of relevance to discuss how each of these mechanisms compares to each other. Table 2.2, presents the main performance parameters applicable to the PAT mechanisms under consideration. A total of five performance parameters allow us to discern quickly the differences between mechanisms.

The first parameter is regarding **angular steering acceleration**, which indicates the responsiveness of the mechanism to turning itself, to align the incident beam into the desired direction. The second parameter centers on **Pointing resolution**, which indicates the maximum amount of precision to which the mechanism can direct the incident beam. The third parameter focuses on **Angular motion**, which points out the maximum displacement range of the mechanism, such a parameter is relevant for determining the achievable Field Of Regard (FOR).

The fourth is dedicated to outlining the **Steering mechanism** which the PAT mechanism relies on to provide steering motion, such a parameter allows us to determine if there is the possibility to further tune the mechanism to better match the desired specifications. Lastly, the fifth one is dedicated to **Technology Readiness Level (TRL)** as a measurement system to assess the level of maturity of the different PAT mechanisms. This parameter is crucial to designing a system that can be practically implemented in the foreseeable future. The scaling for this parameter follows the guidelines outlined by [42].

Table 2.2: Comparison of the main specifications of PAT mechanisms

PAT mechanism	Angular steering acceleration	Pointing resolution	Angular motion	Steering mechanism	TRL
Gimbal	Low (0.4 - 150 Hz) [46]	$mrad$ [25]	Wide (inner $\pm 20^\circ$ & outer $\pm 180^\circ$) [7]	Mechanical [25]	Level 9 "flight proven" [55], [16] **
FSM	High (300-750 Hz) [62], [38]	sub- μrad [25], [62]	Narrow ($\pm 1.5-6^\circ$) [62], [38]	Mechanical, piezo-electric and electromechanical [25]	Level 9 "Flight proven" [16] [55]
AO	High to Extremely high (0.1 - 10 kHz) [63]	$nrad$ [25], [63]	Extremely narrow ($\pm 3.5 \mu m$ actuator stroke) [25], [63]	Electro-mechanical [25]	TRL 8 "flight qualified" through test and demonstration [25], [41]
LCs	High (600-900 Hz) [15]	sub- μrad [25], [15]	Extremely narrow ($\pm 4 mrad$) [25], [49]	Electrical [25]	TRL 4 validation in laboratory environment [68] [25]

Before continuing to the next section, it is necessary to comment on an aspect related to classifying the TRL level for Gimbal systems. The high TRL level of Gimbal mechanisms is due to the aspect that the coarse pointing system of the LCT in [16] is being considered a variant of gimbal mechanisms, thus placed in the "flight proven" level. With the differences in performance of PAT mechanisms presented, it is now relevant to center on describing the level of performance that the PAT system needs to be capable of meeting.

2.1.2. PAT system requirements

With an overview of the type of performance differences expected from PAT mechanisms, it is now relevant to center on describing the performance level that is required to meet mission requirements. To achieve this, Table 2.3 provides an overview of the main PAT system requirements with a detailed description. Motivation for the requirements can be found underneath the table.

Table 2.3: PAT system requirements

Requirement identifier	Description
PATR-01	The total angular (static) and angular-rate (dynamic) Root Mean Square (RMS) pointing jitter error, shall not exceed 50 μrad / $\mu rad/sec$.
PATR-02	The total angle of the Field Of Uncertainty (FOU) shall not surpass 7.2 $mrad$.
PATR-03	Total acquisition time shall not surpass the 66 <i>second</i> threshold mark.

The following section provides an overview of the motivation behind the requirements outlined in Table 2.3. **PATR-01**, has been derived utilizing a combination of 2 key customer design drivers. The first design driver is the envisioned divergence angle to be utilized by the mobile OCT. The divergence angle allows us to determine the spread of the laser beam, as it propagates through a medium. The second design driver is the allowable Bit-Error-Rate (BER) for a given output laser power. The BER can be utilized to quantify the number of bit errors occurring during transmission, relative to the total number of bits transmitted.

Combining these 2 design drivers, as well as utilizing the optimum ratio between the divergence angle and pointing jitter error method proposed by [64], it is then possible to define the allowable angular pointing jitter error. For a system that is envisioned to employ a divergence angle (θ_{div}) of $330 \mu rad$, with an allowable BER of 10^{-6} , the allowable total angular pointing jitter-error is $50 \mu rad$ (RMS). **Note:** The divergence angle and pointing jitter-error have been selected in collaboration with the customer research organization.

The next is requirement **PATR-02**, which has been derived utilizing the expected receiver terminal pointing errors and an accurate positional error estimate of Two-Line Elements (TLE) orbital positional information by [52]. TLE's were selected as a worst-case (conservative) scenario, in case more accurate orbital position and determination systems are not available. The expected customer terminal total pointing error (host uncertainty error) has been set at $1 mrad$, while the positional error has been defined as $\pm 2.6 Km$ (combining both Along track and Cross-track satellite position errors).

The envisioned OCT satellites (at the receiver end) have an altitude of $1000 Km$, with the positional error of TLE's, which then leads to a positional error half angle of $2.6 mrad$. Adding both half angles to derive the complete positional error cone, as well as adding the host uncertainty error, it is then possible to arrive at the $6.2 mrad$ of the total angle of FOU.

Lastly is requirement **PATR-03** which has been derived as a direct result of **MSR-01**, under requirement OCT-016 in [1]. It outlines a limit on the total maximum acquisition time of under 100 seconds. To guarantee that the maximum threshold of acquisition time is not reached, the maximum acquisition time has been divided by a safety margin of 1.5, leading to the effective acquisition time requirement of 66 seconds.

For a PAT system to achieve fast-tracking capabilities, it requires employing mechanisms with the capacity to achieve high angular steering accelerations. Thus PAT mechanisms such as FSMs, AOs or LCs are particularly suited for such tasks. However, the fact that it is envisioned to be a mobile FSOC link also places a requirement of a wide FOV, which in turn requires mechanisms with a wide angular motion. As visible from the summary provided by Table 2.2, there are no individual mechanisms with both a wide angular motion and high steering accelerations. Thus for mobile FSOC systems often combinations of several PAT mechanisms are employed, typically a Gimbal mechanism is paired with an FSM and/or AO mechanism to obtain both objectives, such combinations have been utilized in the works by [60], [25], [50], [51], [26], [29], [31].

2.2. PAT modeling sub-system requirements

With the system requirements presented and motivated, the next step involves the presentation of the modeling sub-system requirements. This set of sub-requirements aims to specify the type of actuators that are to be represented via simulation of the Dual-stage OCT. These have been defined in consultation with the Royal Netherlands Air Force (Koninklijke Luchtmacht) and represent Commercial-Off-The-Shelf (COTS) components.

Table 2.4: PAT modeling sub-system requirements

Requirement identifier	Description
PATR-01-01	The precise pointing actuator model shall approximate the dynamics and performance characteristics of the Optics in Motion OIM5001 FSM.
PATR-01-02	The coarse pointing actuator model shall approximate the dynamics and performance characteristics of the PlaneWave instruments L-350 direct drive telescope mount.

2.3. Ship-wave motion & relevant previous research

In the previous section, an overview of PAT systems and mission design has been provided. Typically such overviews in the context of FSOC are envisioned primarily for ground stations. This is expected, as currently the majority of contemporary research targets such applications. However, for the scope of this work and as part of **MSR-02**, the focus lies on a mobile OCT that can be fitted and operated from frigate-sized ships.

Therefore it is now of relevance to discuss the type of dynamic environment to which a Naval OCT terminal would be exposed. To facilitate this discussion, Figure 2.9 provides an overview of the standard notation and sign convention of the motion of a ship. The ship-wave motion can be described by a 6 DOF rotational joint relative to a fixed frame of reference, these can be further split into; three rotations (roll, pitch & yaw) and three translations (surge, sway & heave).

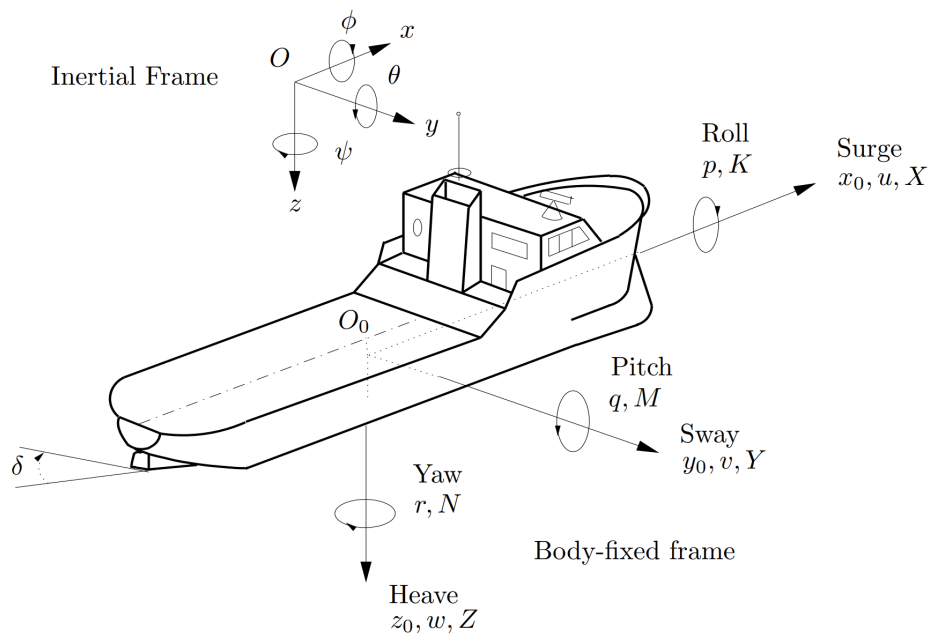


Figure 2.9: Standard notation and sign convention of ship-wave interaction/motion [47]

With the sign convention of the interaction between ship and waves covered, it is now relevant to point out that modeling such perturbances requires stochastic models as "ocean waves are random in terms of both time and space"[47]. Since the attention of the work is on the stabilization and precise pointing aspects of mobile FSOC, the development of such stochastic models is left out of the scope of work.

Instead, the focus lies on utilizing the data sets gathered by [53]. During this research, measurements were performed utilizing an onboard Inertial Measurement System (IMS) with a sampling bandwidth of 200 Hz , onboard Royal Netherlands Navy (Koninklijke Marine) frigate-sized vessels. The research aimed to capture accurate rotational angles and rotational rates of the vessels, to record the ship-wave interactions as well as other disturbances, introduced onboard an operational vessel. A plot of the complete time-varying data set, of the pitch rate motion, can be seen in Figure 2.10.

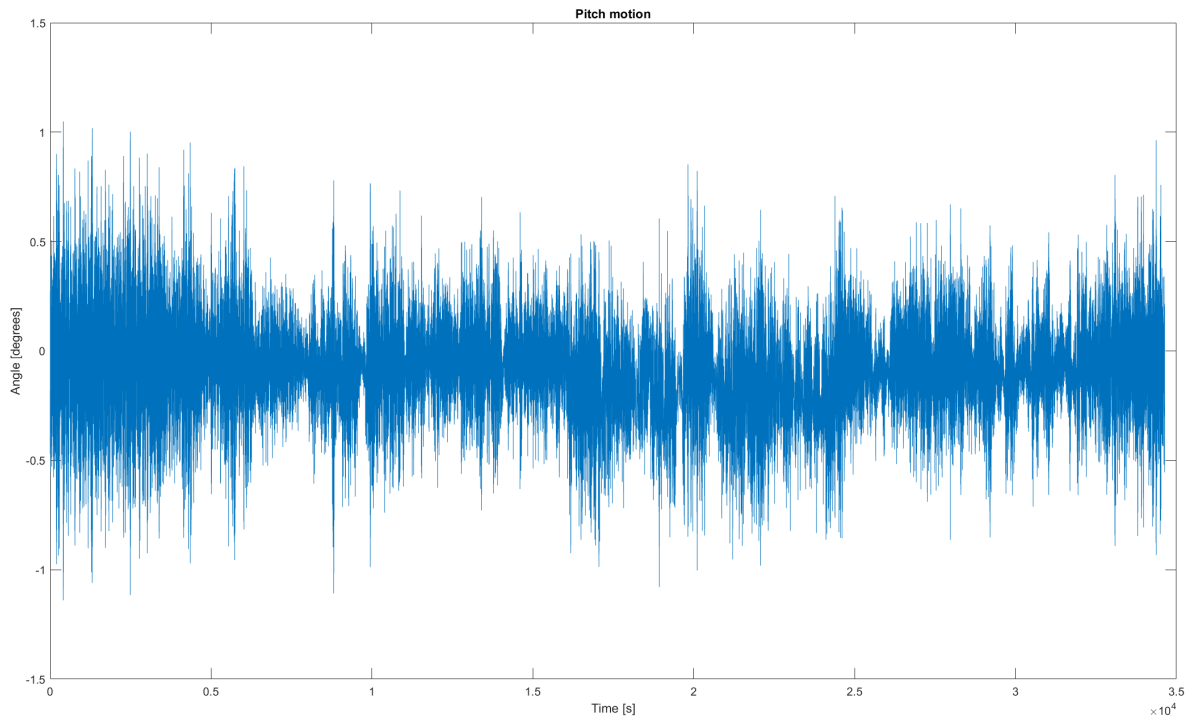


Figure 2.10: Pitch rate motion with an amplitude in degrees plotted over sampled time in seconds (Total sample duration of 9 Hours and 37 minutes)

Drawing detailed conclusions from data in the time domain can prove challenging, as the different interactions can be intertwined with each other. An effective method to separate the different interactions, to assess their impact on the performance of the mobile OCT, is to instead perform the analyses in the frequency domain. This is covered in further detail in chapter 3, where the methods to convert time-varying data to frequency data, as well as different functions to perform different types of analyses, are covered in further detail.

2.3.1. Relevant previous research

The aim of this subsection is to provide a brief overview to the reader, of the previous research which has been performed on mobile FSOC OCT's. The objective is to outline what has been achieved until now (as of writing) as to identify the scientific gap that this work aims to fulfill.

In the last decades, massive advancements have been achieved regarding FSOC. A clear highlight of such advancements is the National Aeronautics and Space Administration (NASA) Deep Space Optical Communications (DSOC) experiment onboard the PSYCHE spacecraft. The DSOC aims to demonstrate the capacity to perform optical communications at extreme distances as "The optical communication link would be demonstrated during the Cruise Phase of the Psyche mission over Earth-probe distances ranging from 0.1 to over 2 astronomical units (AU)" [14] [n.b. $1\text{ AU} \sim 149.6 \times 10^6\text{ Km}$].

To achieve effective communication at such extreme distances an accurate and precise PAT system is required. While is difficult (as of writing) to find publicly available information about the specific performance specifications of the DSOC, it is possible to compare it to the Lunar Laser Communication

System (LLCD), which exemplifies an OCT intended for extra long range communications. The LLCD has been developed jointly by the Massachusetts Institute of Technology (MIT) Lincoln Laboratory and NASA. "The LLCD successfully demonstrated a 622 Mbps lasercom link from lunar orbit" [21].

To achieve this, the PAT system contains 5 stages, while body pointing is not considered directly part of the PAT system it has a relatively strict pointing requirement of " $100 \mu\text{rad}$ RMS in each axis to support compatibility with the LLCD" [3]. From that starting line the first stage includes a two-axis gimbal capable of executing coarse adjustments. Adding to that are the transmitter and receiver fibers, which have two-axis linear stages, for the receiver fiber this enables nutation (fine) tracking of the up-link signal, while for the transmitter fiber, it implements the point ahead angle required for the downlink signal. "The overall pointing accuracy is within $2.5 \mu\text{rad}$ " [3]. Figure 2.11 provides a visual overview of the LLCD PAT system.

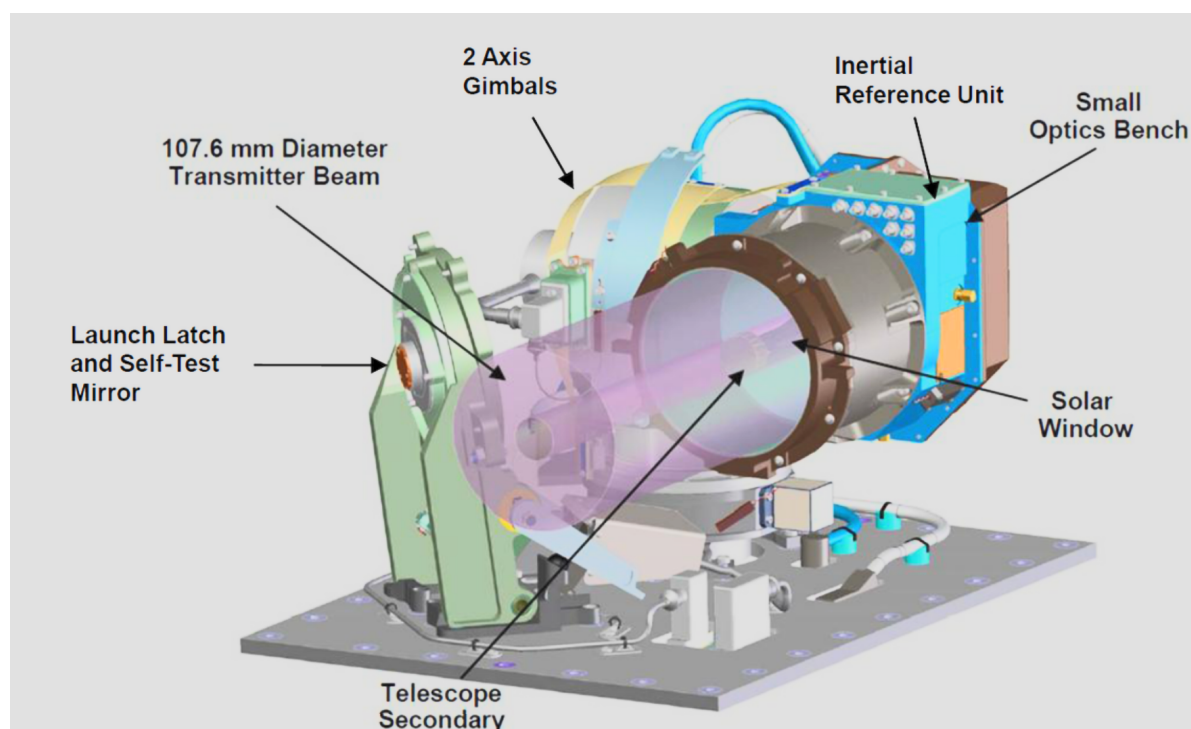


Figure 2.11: Illustration/rendering of the LLCD inertially stabilized telescoped by [3]

Now having discussed some of the extreme long-range FSOC systems, the focus now returns to ground-based OCT meant for mobile applications. Currently (as of writing) there is a significant gap in publicly available information regarding the performance and design of mobile ground OCT stations. One of the few projects that closely resembles the naval setting of this work, is the Trident Warrior (2006) exercise conducted by the United States Naval Research Laboratory (NRL) in which a dual-stage PAT system was effectively utilized for short optical communications, "Over 10 hours of link data was recorded at ship-to-ship ranges up to 17.5 km" [51].

Before delving into the details of this system it is relevant to point out the fact that this communications link utilizes a Modulating Retro-Reflector (MRR) communications link, the key difference being the fact that the transmitter emits a continuous optical beam towards the receiver, which contains both a retro-reflector and a modulator system. By varying the reflective properties of the retro-reflector allows to encode data into the reflected beam. One of the key advantages of MRR are the relative loose pointing requirements "the pointing requirements for the MRR are very loose. A gimbal-mounted MRR only needs a pointing accuracy better than the retroreflector's field of view. For glass corner-cube retro reflectors, this is $\pm 15^\circ$ " [51].

Since a laser with an angular divergence of 1 mrad was utilized, it required a dual-stage PAT system in order to cope with both atmospheric turbulence, ship roll and relative bearing changes between transmitter and receiver ends. The first stage of the PAT system consisted of a Sagebrush gimbal utilized for coarse tracking "The gimbals were pointed manually using sighting cameras and joysticks to within $\sim 1^\circ$ of the opposite terminal" [51]. It is relevant to note that the gimbals have been stabilized via the employment of an inertial measurement unit, neglecting the need for compensation of the ship-wave motions themselves. For the second stage of the PAT system it consisted of the fast steering mirrors, which had an accuracy up to " $\pm 100 \mu\text{rad}$ " [51]. To provide a visual overview of how the dual-stage PAT system, Figure 2.12 provides an overview of the system mounted into a vessel.



Figure 2.12: FSO OCT terminal onboard the USS Bonhomme Richard for the Trident Warrior, courtesy of [51].

While there is still a significant amount of ongoing research in increasing the ranges for which optical communications can be achieved, as well as into increasing the achievable bandwidth during communications, as of writing there is a significant gap when it comes to research about ground stations placed in dynamic environments. While the previous (Trident Warrior exercise) study case highlighted that in principle, if adequate compensation is provided into an OCT terminal, optical communications remain achievable in dynamic environments for short distances (20-30 Km). However, questions regarding the level of performance required for a PAT system to achieve long-distance optical communications remain open, thus presenting an opportunity to expand the available research in the topic.

3

Ship-wave motion frequency analysis

This chapter aims to present the steps taken to perform the frequency analysis on the ship-wave motion data set. The objective is to present the main steps taken to process the data, which are used as inputs for subsiding chapters. It is relevant to highlight the fact that the initial data set, contained sampling inconsistencies. These were present over sections across the entire data set. The pre-processing steps taken to address the issues arising from sampling inconsistencies are covered in Appendix A. The main steps involved dissecting the data set into smaller pieces and the application of both windowing and zero padding functions. The result from taking these steps is an averaged data set, which aims to conserve the original information and is utilized as an input for the functions utilized to convert data from the time domain into the frequency domain.

The first section of this chapter covers the Discrete Fourier Transform algorithm, which is utilized to convert the time-varying data into frequency-varying data. Aspects such as the analytical expression of the algorithm as well as the results of implementing the algorithm are presented. The next section discusses the Power Spectral Density function, which is utilized to determine the amount of energy contained in the ship-wave data per frequency. Aspects such as the analytical expression and the type of achievable results are covered in this section. The chapter concludes by discussing the Cumulative Power Spectrum function. This function aims to provide information regarding the cumulative sum of power across the relevant frequency ranges. The analytical derivation and results of implementing it are part of the discussion.

3.1. Discrete Fourier Transform algorithm

The aim of this section is to cover the fundamentals of the Discrete Fourier Transform (DFT) algorithm, which is tasked with converting the time-domain data into frequency-domain data. While there are many approaches to perform the conversion from time to frequency domain, the approach to be discussed is specific to the implementation provided by the in-built functionality in Matlab (MathWorks Inc software) [19]. The algorithm is based on the library FFTW by [8]. The DFT algorithm is characterized by requiring finite sequences of equally spaced input samples. The DFT can be defined as in Equation 3.1.

$$Y(k) = \sum_{j=1}^n X(j) W_n^{(j-1)(k-1)} \quad (3.1)$$

Where $Y(k)$ is a sequence of complex numbers, $X(j)$ is the input vector with discrete time-varying information and $W_n = \exp(-j2\pi/n)$ which is required for cross-correlation of the input sequence, it correlates both cosines and imaginary sines. Equation 3.1 completely describes the DFT which only comprises frequency components. Further information about the implementation of the DFT into the FFTW library can be found in [8].

By utilizing the DFT algorithm, the time domain data can be converted into the frequency domain. In the frequency domain, it is possible to determine the frequencies at which the greatest amount of

amplitude of the ship-wave interactions and disturbances occur. However, the vector containing the frequency information from the DFT is complex, containing phase information of the original signal. Thus for the signal processing applications of this work, the one sided section of the DFT is tacking and its amplitude is normalized.

To perform amplitude normalization it is possible to make use of the Parsevals relation, which links the total amount of energy contained between time signals and their corresponding frequency signals. Parseval's relation can be expressed as in Equation 3.2.

$$\sum_{i=0}^{N-1} x[i]^2 = \frac{2}{N} \sum_{k=0}^{N/2} \text{Mag } X[k]^2 \quad (3.2)$$

Where $x[i]$ contains the time domain signal with i running from zero to $N - 1$. $X[k]$ contains the frequency spectrum of the time domain signal, where K runs from zero to $N/2$. The left side of this equation lets us know that to obtain the total energy contained in the time domain signal, it is required to sum all the energies contained in the N individual samples. The same holds for the other side, where the total energy is found by summing the energies up to the $N/2 + 1$ samples. Note that the sum is extending up to $N/2 + 1$ because the original DFT contains both positive and negative frequencies. Since for this analysis only the positive signals are of interest, it is only required to sum one of the two sides.

Parsevals theorem concentrates on the conservation of energy, however, the goal is to perform normalization of amplitude in the frequency domain. To achieve this, it is possible to apply a slight modification to the relation. The change is for the magnitude of $X[k]$ not to be squared since "Energy is proportional to the amplitude squared" [57].

By converting the entire data set from the time domain to the frequency domain and then normalizing the amplitude to represent the amplitude of the time signal. The code developed to implement this into the Matlab environment, which is utilized for the rest of the simulation, can be found as "normalize_amplitude_dft_and_psd_creation" in Appendix D. By running this code it is possible to obtain Figure 3.1.

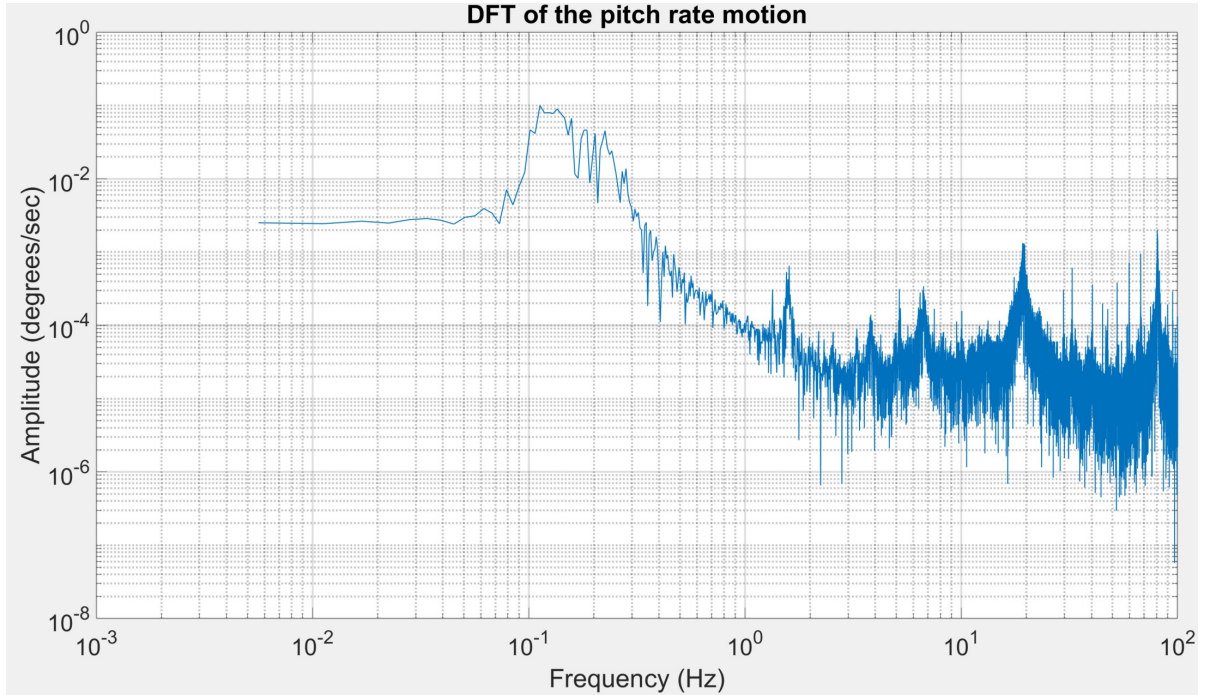


Figure 3.1: Result of applying the DFT algorithm to the pitch rate Ship-Wave motion data set of an individual interval

Multiple aspects can be recognized from the information conveyed on Figure 3.1. The first aspect is the fact that for the recorded signal, the greatest amount of energy in terms of amplitude, can be found close to the frequency range of wind-induced waves, that being in the 0.01 to 1 Hz frequencies (as gravity-induced waves occur at much lower frequencies). However, there are several peaks in amplitude of disturbances/interactions occurring at much higher frequencies than those of wind-induced waves. Since these peaks could have multiple sources of origin (as broadband noise/disturbances are under consideration), such as rotary engines with slight variations in the mass of their turning disks or shafts, actuators, hydraulic pumps, rotary radars, walking personnel, etc, identifying the origin of specific amplitude peaks is considered outside the scope of the investigation.

3.2. Power Spectral Density functions from sub-data sets

The next step in the frequency analysis of the data set is the calculation of the Power Spectral Densities (PSD's) of the averaged intervals. The PSD provides an approach to describe the amount of power/energy contained per frequency of a signal. This is of special relevance for this work, as the disturbances/interactions recorded in the data set are of a stochastic nature, that is the signals are composed of random or unpredictable components, which are not straightforward (if possible) to describe in statistical terms.

"The name power spectral density comes from observing how the average power of a random process is distributed in the frequency domain" [10], thus the PSD provides a suitable means to characterize and represent the energy contained in the stochastic signals contained in the data set. To provide additional information regarding how the PSD is calculated, it is possible to start by defining a given stationary stochastic signal $x(t)$, the autocorrelation function, $R_{xx}(\tau)$, can then be defined as:

$$R_{xx}(\tau) = \lim_{T \rightarrow \infty} \frac{1}{T} \int_{-T/2}^{T/2} x(t) \cdot x(t + \tau) dt \quad (3.3)$$

This function represents the similarity between the signals at different time lags (τ). The PSD is derived by taking the Fourier transform of the autocorrelation function. According to the Wiener-Khinchin theorem, the PSD, $S_{xx}(f)$, is given by:

$$S_{xx}(f) = \int_{-\infty}^{\infty} R_{xx}(\tau) \cdot e^{-j2\pi f \cdot \tau} d\tau \quad (3.4)$$

This relation shows that the PSD provides a frequency-domain representation of the power contained in a stochastic signal. In the discrete case, given a signal $x[n]$ with length N , the power spectral density can be derived from the Discrete Fourier Transform (DFT). The DFT of $x[n]$ is:

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{-j2\pi k \cdot n/N} \quad (3.5)$$

The PSD estimate, $S_{xx}[k]$, is then given by:

$$S_{xx}[k] = \frac{1}{N} \cdot |X[k]|^2 \quad (3.6)$$

This discrete PSD represents the energy or power content of the signal at various frequency bins. By normalizing by N , it accounts for the length of the signal and provides a consistent measure of power distribution across the frequency spectrum. As part of the pre-processing steps mentioned at the introduction of this chapter, the data set is provided as a set of averaged intervals meeting consistent sampling criteria. From these, it is now possible to create a more accurate estimate of the underlying power distribution of the real disturbances/interactions signals.

To determine the average PSD of the multiple input intervals, it is possible to sum the resulting PSD for each signal and then divide by the total number of intervals, this can be described analytically as

follows:

$$S_{xx,avg}[k] = \frac{1}{M} \sum_{m=1}^M S_{xx,m}[k] \quad (3.7)$$

Where M corresponds to the total number of intervals and $S_{xx,m}[k]$ to the value of the PSD of each interval. By averaging the PSD it is possible to provide an estimate of the underlying energy of the real signal, which is likely to minimize the influence of outliers or unusual variations in a single interval. Since the original dataset contains operational data of the ship at different sea depths and operating modes, it is then critical to obtain a consistent and representative measure of the signal characteristics, thus motivating the approach of utilizing the average PSD from multiple intervals.

The results of implementing the average PSD calculation into the high-frequency interval set can be observed in Figure 3.2. The code developed to implement this into the Matlab environment, which is utilized for the rest of the simulation, can be found as "normalize_amplitude_dft_and_psd_creation" and "calculate_average_psd" in Appendix D. Note that the total 6 DOF Ship-Wave motions plots can be found in Appendix B.

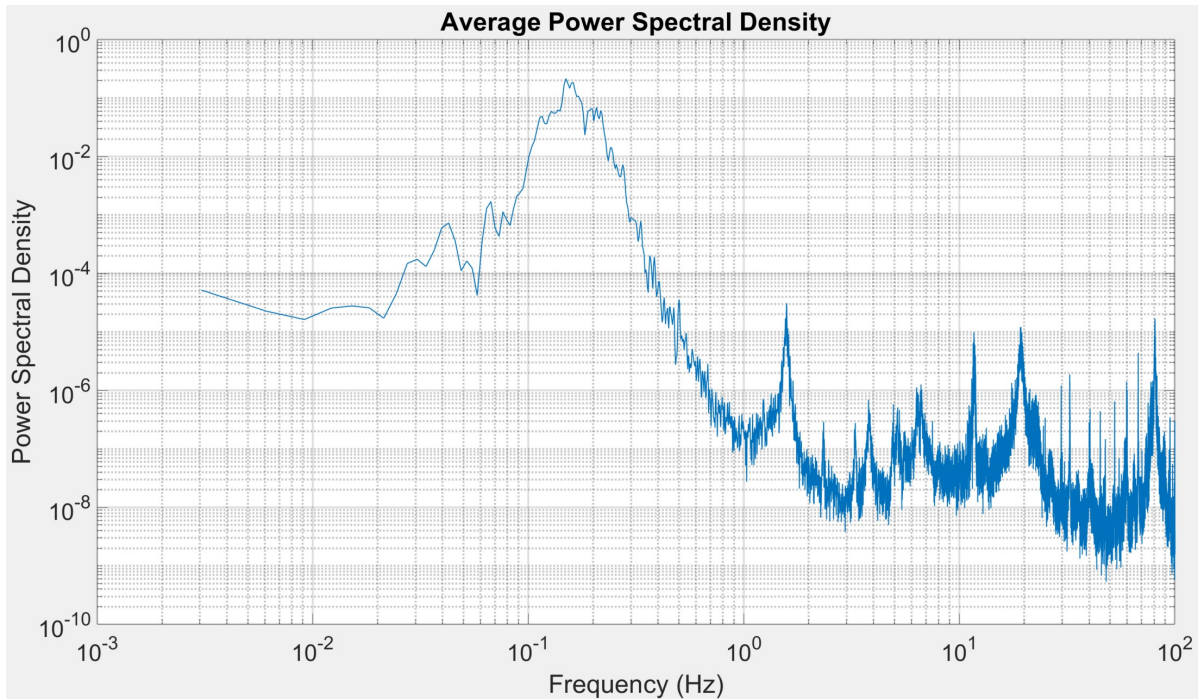


Figure 3.2: Plot of the resulting average PSD of 10 intervals each with durations of 180-200 seconds, obtained from the pitch-rate ship motion data set. N.B. the amplitude on the vertical axis is in $[degrees/sec/Hz]^2$

3.3. Cumulative Power Spectrum function

This section aims to provide a conclusion to this chapter by presenting the Cumulative Power Spectrum (CPS) function. This function differentiates from the PSD, as it provides a visual representation of the accumulation of energy per frequency. During analysis of the frequency responses of the closed-loop systems designed in the subsequent chapter, the CPS function can provide visual information of the accumulation in disturbances affecting performance characteristics such as angular or angular-rate pointing errors.

The CPS function is directly related to the PSD function, as both are typically employed as means of presenting the power over frequency information of a signal. However, the key difference between the PSD function and CPS function, is that the former provides information about the amount of power available per frequency while the latter accumulates the sum of power from the starting frequency (in this case 0 Hz) up to the cut-off frequency. Therefore the main objective of this function is to under-

stand how the total power accumulates as frequency increases.

The CPS function as it is directly related to the PSD function, can be derived by building upon the previous derivation. Picking up from Equation 3.7, the average CPS function can be determined as follows:

$$CPS(f) = \int_0^{f_e} S_{xx,avg}(f) df \quad (3.8)$$

Where the CPS is defined as the cumulative (monotone increasing) integral of the PSD function ranging from 0 Hz all the way up to f_e (in this case is up to 100 Hz), which represents the cutoff frequency. The result of implementing this function can be seen in Figure 3.3. The code developed to implement this into the Matlab environment, which is utilized for the rest of the simulation, can be found in ???. Visible from Figure 3.3 is the CPS function plot resulting from the Pitch Rate disturbance PSD. From this figure, it is possible to argue that the greatest concentration in disturbances lies in the frequency range between $\approx 0.01 Hz$ til $\approx 0.2 Hz$, as this represents the section at which the CPS is growing the fastest, these frequencies match the frequency ranges at which wind-induced waves have their largest amplitudes.

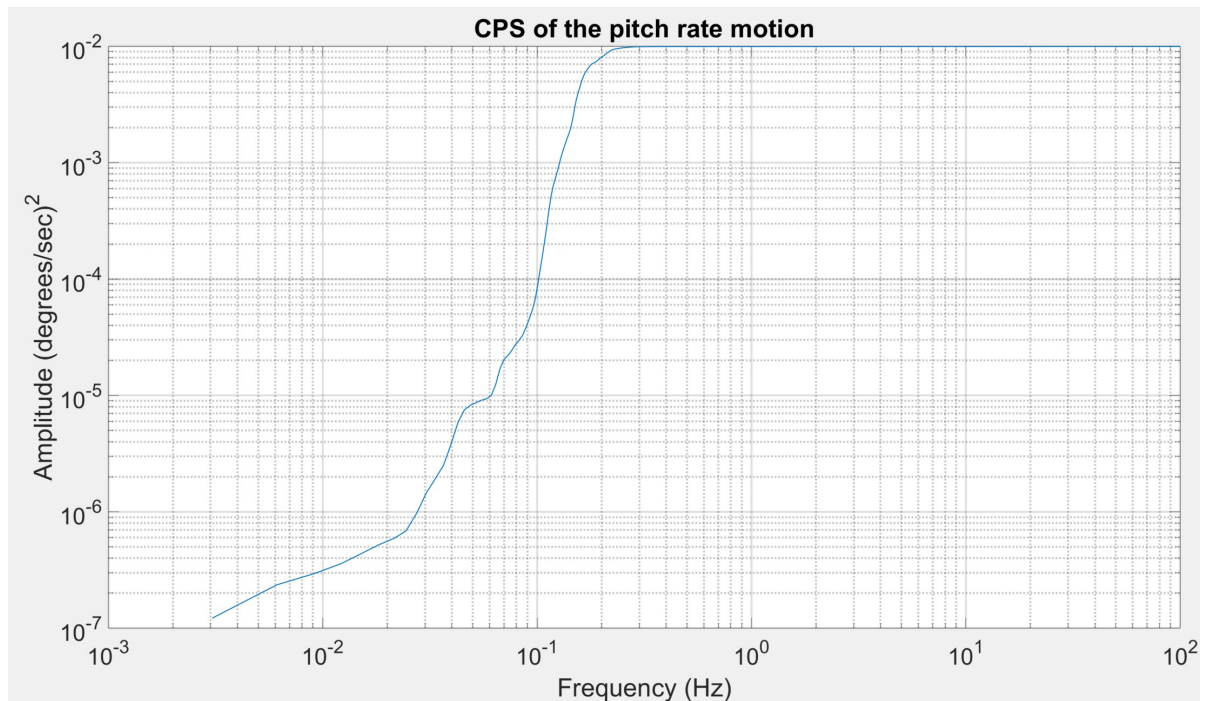


Figure 3.3: Result of the cumulative integral function of the average PSD function of the pitch rate S.W. motion

4

Actuator control loop models

This chapter aims to provide insight into the control logic of the different actuator control models, these include the CPAS, PPAS as well as the Dual-Stage Actuator System. In the upcoming three sections, the control system design for each system is covered separately. The topics covered include loop logic as to outline the reference and command signals across the system, the transfer functions that compose the individual blocks of the system, gain tuning of the controllers, open and closed loop responses of the system, loop gain and sensitivity function of the system, etc.

Before heading into the dedicated sections, it is relevant to specify which type of systems will be discussed and what type of assumptions are related to them. The 3 control models are chosen to be represented as Linear Time Invariant (LTI) systems, which not only simplify the modeling approach required to approximate the expected/desired behavior, but as well are compatible with the Dynamic Error Budget (DEB) analysis. This type of analysis will be discussed in further detail in the subsiding chapter, the main benefit of this type of analysis is the capability of predicting the systems response to the introduction of a stochastic disturbance signal, which in this case would be the Ship-Wave interactions/disturbances.

To represent the Mobile OCT as an LTI system, several assumptions are required to be made. Linearity is one of the main assumptions as it requires that the output of the system to be proportional to the input. It also carries the assumption regarding additivity, which is critical for superposition, in which the response of the system is the same as if all the different inputs that contribute to the analysis are supplied all at once or as if each is supplied separately and the results are then added.

Another critical assumption is regarding the time-invariance of the system, it relates to the system properties remaining constant over time. Such that if a system produces a certain output, if the same input is delayed, the time-shifted response of the system will be equal to the non-time-shifted input. While there are many other properties and assumptions regarding LTI systems, the multiple properties proofs and detailed discussions are considered outside the scope of this work and readers are referred to the works by [30] and [17].

4.1. Precise Pointing Actuator System control model

The focus of this section is to cover in further detail the multiple elements that compose the PPAS control model. The section starts by discussing the overall control logic loop as well as the routing of the signals. Then the section advances to cover the individual elements that compose the system, covering design choices and citing performance specifications. With the elements of the system covered next are the Bode plots of the system response at different frequencies, which allow for a better understanding of the achievable performance of the actuator and control loop. A comparison between Open loop and Closed loop functional plots, as well as the capability to reject disturbances in the system visible in the Loop Gain and Loop Sensitivity function plots can be found in Appendix E under the augmented

open loop, closed loop and loop sensitivity subsections part of the Precise Pointing Actuator System accompanying plots, descriptions and code section.

Starting with the control loop logic design visible in Figure 4.1. The control logic is depicted as a cascaded signal flow, in which the signals propagate through the system starting from the left, at which the system receives a reference input signal (Y_{ref}) command. As the signal propagates through the system, the previous measurement regarding the actuator position is subtracted to the reference input signal, yielding the current loop error, in the position of the actuator. This error signal serves as an input to the controller, which then provides an input torque force to the actuator. Then the actuator converts the torque force into a rotational displacement with a certain amount of disturbance, arising from errors propagating through the system. In this simplified case the main simulated source of errors is centered on the tracking error of the tracking sensor.

While there are other sources of errors such as errors arising from Digital to Analog and vice versa conversion, in this simplified case scenario, the focus lies fully on the errors arising from the positional tracking errors. Thus the signal traveling along the feedback path passes through the tracking sensor model of a Quad-cell Photo Diode (QPD) laser beam tracking sensor, where eventually gaussian noise is added to the signal. Sequentially the feedback signal concludes by being subtracted to the upcoming reference position signal, while the main output of the system is the current loop position of the system plus disturbances propagating in the system.

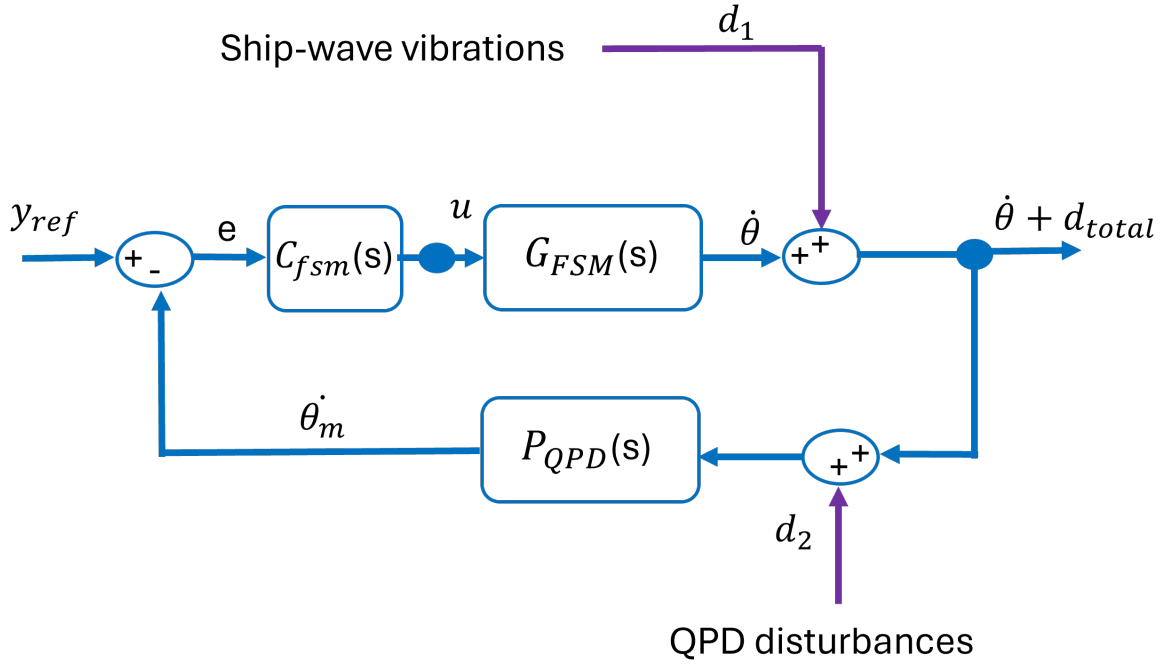


Figure 4.1: PPAS control loop logic architecture

The following subsections cover in further detail the elements depicted in the blocks part of the PPAS shown in Figure 4.1. The order in which these are covered differs from the cascaded loop logic represented in the figure, to improve readability of the chapter.

4.1.1. FSM actuator model

To create a model of the FSM actuator, as discussed in [4], typically there are 3 main approaches, these include; "knowledge-based modeling (white box modeling), data-driven modeling (black box modeling), and process-based modeling (grey box modeling)". For modeling the FSM, the "white box" modeling approach has been selected, due to its high accuracy and being commonly researched, as this approach "has been followed by most of the researchers" [4], which in this case is of relevance, as it is

desirable to obtain a model that is as close as possible to the specific dynamics of the Optics In Motion OIM5001 FSM, in compliance to **PATR-01-01**.

The OIM5001 is a commercial state-of-the-art FSM, equipped with four voice coil actuators with moving magnets and static coils, which operate in a "push-pull" configuration. The main performance characteristics as mentioned in [37], include the following, an industry-standard 1-inch glass mirror with the availability of custom coating. Angular range up to ± 3.0 degrees (mechanical). Dynamic performance of 3dB Bandwidth above 850 Hz and a step response time for an input of 1 mrad under 5 ms. In Figure 4.2 a picture of a fully assembled OIM5001 can be seen.



Figure 4.2: OIM5001 FSM [39]

Having discussed the specific type of actuator that is desired to be modeled, next is the discussion of the approach taken to recreate an LTI system, to get a representative actuator model. What is being sought after is to end up with a transfer function, that can map the relationship between an input torque and a deflection angle accurately enough. To achieve this, the approach taken by [5] has been selected. This research provides an accurate representation of an Optics in Motion FSM via a transfer function. This was achieved via system identification with the help of the 3562A, Hewlett-Packard system analyzer.

The system analyzer maps a transfer function to a series of input and output data, for further information the reader is referred to the original work. **Note**, the following are considered the main assumptions of this model. The actuator model can be effectively represented via a single actuator transfer function model, in which the responses of the current amplifier, the mirror voice coil and the internal position sensors are all modeled together. The actuator model of the FSM can be effectively be represented by a 2nd order spring-damper transfer function. Furthermore, it is assumed that there is no crosstalk between axes, such that a single-axis model is sufficient to model the FSM, this is justified by the fact that "The crosstalk magnitude between the axes at the resonance frequency is 38 dB lower than the magnitude of the single axis transfer functions (TFs)" [5]. The transfer function of the FSM is described in Equation 4.1.

$$G_{fsm} = K * \frac{\omega_0^2}{s^2 + 2\omega_0\zeta s + \omega_0^2} \quad (4.1)$$

Where the DC gain $K = 11$ [–], the first natural frequency (suspension mode) $\omega_0 = 169.6$ [rad/s] (27 Hz) and the damping ratio $\zeta = 0.04$. **Note** The DC gain is part of the actuator model, the K shown in the triangular block in Figure 4.1, is an additional gain utilized to convert the output of the actuator transfer function which is in radians, to degrees, as the rest of the analysis has been performed in degrees. To achieve this, the main loop gain multiplies the output in radians to the conversion ratio of $180/\pi$. The Bode plot of the frequency response of the plant model can be seen in Figure 4.3.

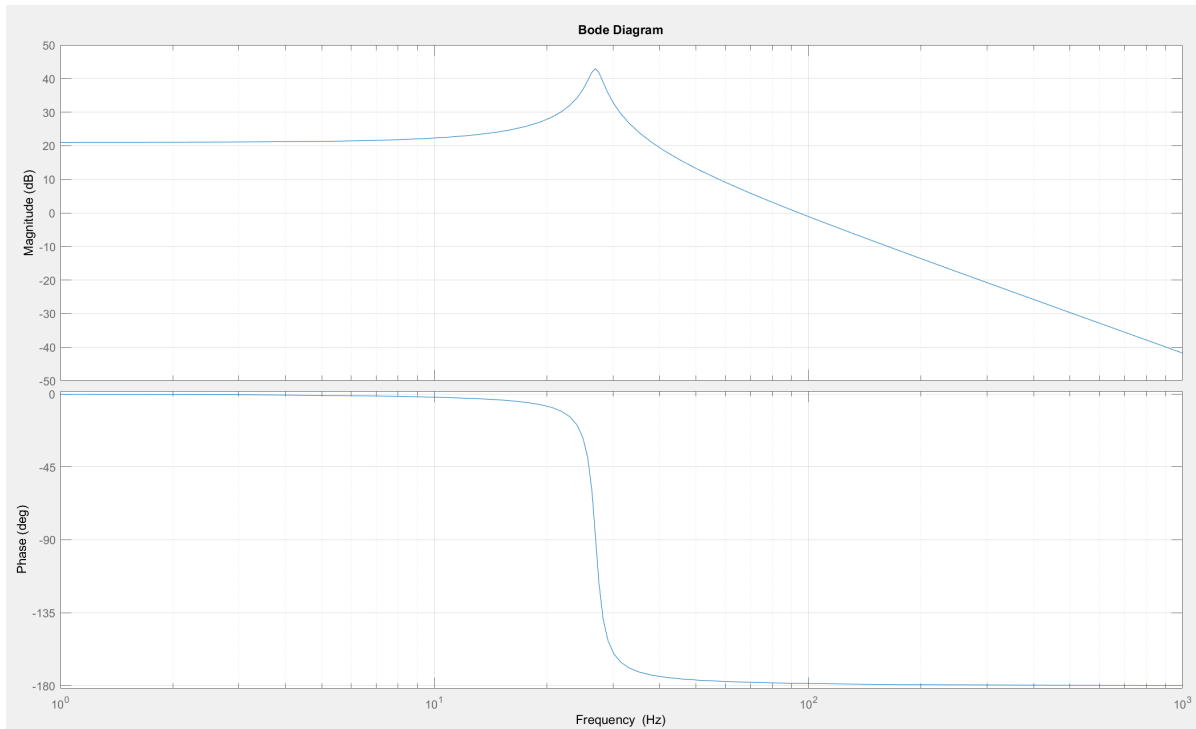


Figure 4.3: FSM actuator model frequency response Bode plot. Note that the first natural frequency is at 27 Hz

4.1.2. QPD- Tracking sensor and sampling board

This subsection aims to cover briefly the model utilized to estimate the QPD laser beam tracking sensor. Modeling the dynamics of a QPD tracking sensor can be challenging, as these require the simulation of the dynamics involved with converting information from the incident light reaching the detector to electric signals, which the controller board can interpret. QPD typically consists of four photodiodes arranged in a quadrant, each photodiode can be modeled individually as a low pass filter with a time delay. Figure 4.4 provides an overview of an exemplary QPD sensor. Furthermore, modeling the relationship between the four quadrants can prove difficult due to the possible existence of non-linearities in the response of each photodiode.

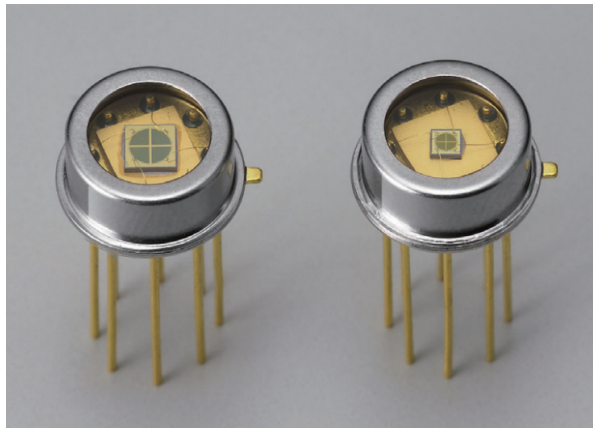


Figure 4.4: Example of a pair of Quadrant PhotoDiode InGaAs tracking sensors [13]. The sensor on the left has a 2mm quadrant element while the right sensor has a 1mm quadrant element

This work focuses on the control behavior and achievable performance of the system, therefore modeling the incident light sensitivity of the QPD sensor is considered outside the scope of work. Thus the chosen transfer function centers purely on the time delay, introduced by the sampling frequency of both the tracking sensor and controller board. Placing the focus on only modeling the time delay can

be motivated by the fact that from a control-theory perspective, what is of greatest interest is how a sensor affects the relative phase gain of the total system as a whole. This directly influences not only the achievable performance at higher frequencies but also the gain tuning procedure of the controller.

It is possible to model the time delay (T) in the Laplace domain as in Equation 4.2.

$$T(s) = \exp^{-sT} \quad (4.2)$$

Where the S stands for the complex frequency variable. Since it is of interest to determine the effects at higher frequencies, it is then suitable to represent the time delay in the frequency domain as in Equation 4.3.

$$T(j\omega) = \exp^{-j\omega T} \quad (4.3)$$

The impact the time delay has on the phase margin of the system is clear whenever the exponential term in Equation 4.2 is expressed in polar form as $\cos(\omega T) - j\sin(\omega T)$, which has a unit magnitude and a phase of $-\omega T$ radians. As the frequency scales linearly with frequency (ω), the effects of time delay can become severe at higher frequencies, as these could result in reaching the critical phase lag of -180 degrees, potentially leading to instability in the system.

With an understanding of the effects of time delay, next is the chosen representation of the time delay transfer function. While the original expression of time delay is straightforward mainly consisting of an exponential term, this results in an irrational transfer function. Irrational transfer functions increase the difficulty of performing analysis on the frequency response of a system, as in the case of time delay the linear phase lag increases indefinitely with frequency, thus increasing the difficulty of assessing phase margin and stability.

As the subsiding chapter centers primarily on the integration of the frequency response analysis of the systems covered in this chapter, the choice was made to opt for a Pade approximation of the time delay, which is a rational transfer function. As the actuator model is 2nd order, it was decided to opt for a 2nd order Pade approximation, the n-th order expression is given in Equation 4.4.

$$P_{QPD} = \frac{(1 - \frac{\theta}{2n}s)^n}{(1 + \frac{\theta}{2n}s)^n} \quad (4.4)$$

Where θ represents the time delay in seconds. The chosen time delay is related to the chosen controller board and tracking sensor. For this work, the controller board is considered a generic option with a sampling speed of $20 \text{ kHz} / 50 \mu s$ (Typical sampling speed among COTS Real-time controller boards, such as the one utilized in the research by [5]), while the QPD tracking sensor is specific to the Hamamatsu G6849-01 InGaAs PIN Photodiode with a typical sampling speed of $120 \text{ MHz} / 8.3 \text{ ns}$.

As the time delay introduced by the tracking sensor is 4 orders of magnitude smaller than the sampling board speed, it is then possible to neglect the delay of the tracking sensor and model purely based on the effects of the time delay of the control board. Thus the modeled sampling speed for the effective time delay is $50 \mu s$. The Bode plot of the 2nd order Pade time delay approximation can be seen in Figure 4.5, it is possible to denote that the phase loss at 1 KHz remains manageable at -18 Degrees . However, as the frequency increases the sampling delay starts playing a more dominant factor, leading to a complete phase margin loss (-180 Degrees) at the 12.7 KHz mark.

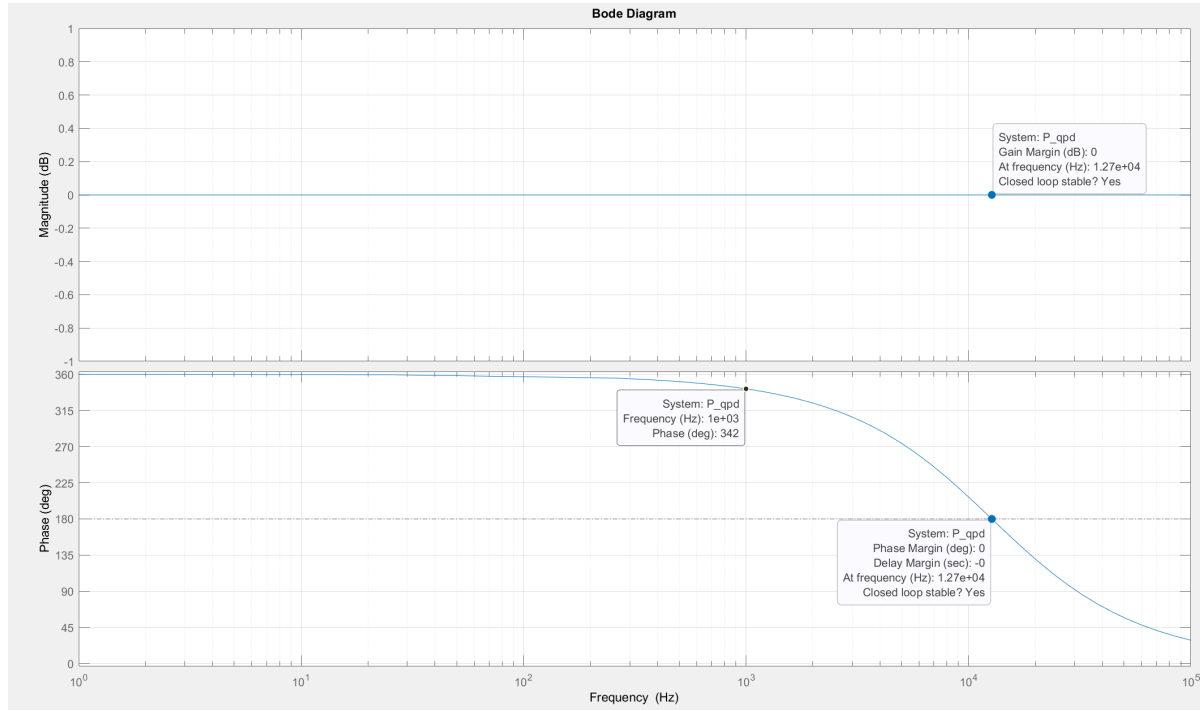


Figure 4.5: Frequency response of the QPD-tracking sensor time delay function

Tracking error

Although the tracking performance of a realistic QPD sensor can be quite accurate and precise, especially if it is receiving a sufficient amount of incident light, it will be limited at a certain point. This in turn can result in the introduction of tracking errors into the control system, which at higher frequencies could result in undesirable/unstable behavior.

Therefore to model the tracking error, a set of disturbances are introduced into the tracking loop, which are based on the performance of an exemplary QPD sensor specifications sheet. The key assumptions as well as the procedure utilized to perform this estimation can be found in Appendix E under the QPD tracking error estimation model description subsection. Implementing the equations discussed in this subsection and making use of the tool presented in chapter 3 and Appendix A, it is then possible to reconstruct the PSD of the tracking error, based on the specifications of the G6849-01 InGaAs PIN Quad-Photodiode from [13], as well with an assumption of 858.7 nW of incident laser power as well as focal length of 1.5 m . The results are plotted in Figure 4.6, for a standard deviation of 88.6 $\mu\text{degrees}$ ($\simeq 1.55 \mu\text{rads}$).

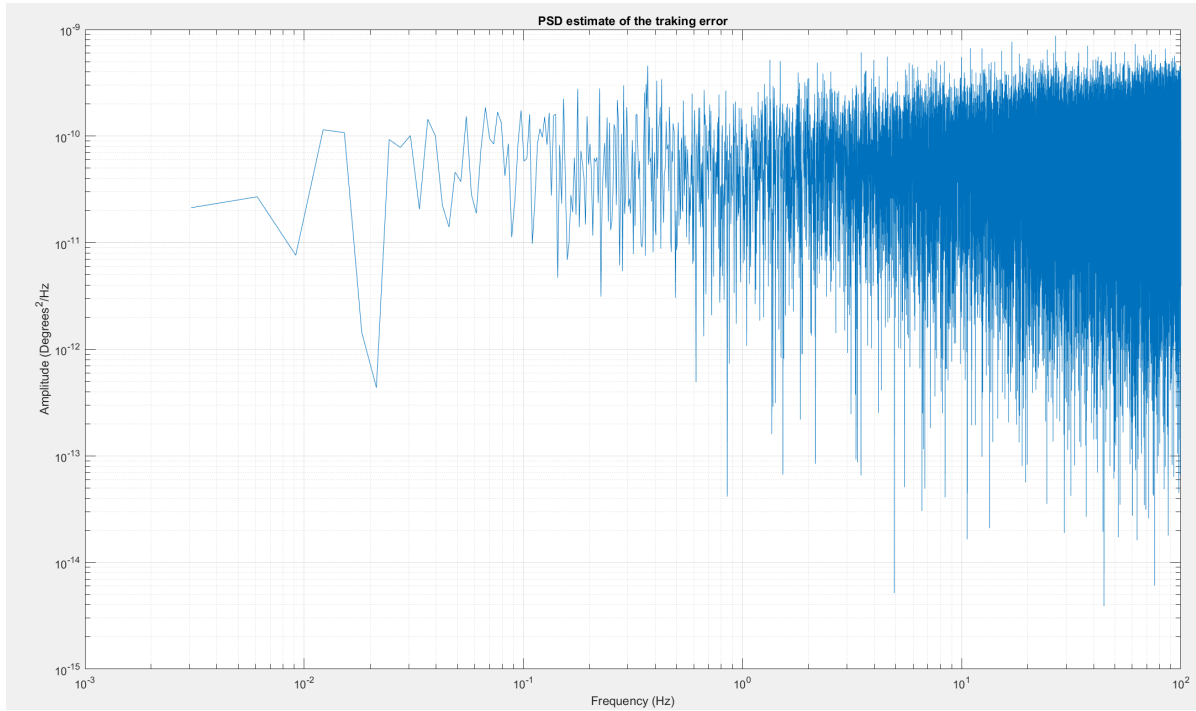


Figure 4.6: PSD estimate of the tracking error, plotted for the same frequencies as the ones of the Ship-Wave IMS data set

From Figure 4.6 it is possible to denote that due to the high amount of fluctuation visible at higher frequencies, this could in turn increase the relative contribution of the QPD-tracking error disturbances to the overall sum of all disturbances affecting the closed loop system.

4.1.3. PID controller design and tuning

To conclude the discussion regarding the control system logic architecture, it is then relevant to discuss the design of the controller. In this case, a Proportional Integral Derivative (PID) controller has been chosen, based on the characteristics of the 2nd order spring damper actuator model of the PPAS. As visible in Figure 4.3, the actuator model has a steep phase loss at higher frequencies, therefore the controller design would account for this by prioritizing increasing the phase gain at higher frequencies.

There are 3 main components to a PID controller. Starting with the proportional part, this is the section intended for minimizing the present error in the system, however by itself, due to the possible introduction of oscillatory behavior (particularly in spring-damper systems), it could result in difficulties with completely removing steady-state error. To remove the accumulation of errors, it is then useful to incorporate the integral term, capable of summing up the error over time, thus effective at eliminating the steady-state error (at least over a sufficient lapse of time). Lastly, the derivative term helps to counteract rapid changes in the system error. This is based on the prediction of possible future errors by considering the current rate of change of the errors, allowing for a dynamic compensation effect.

One of the main aspects motivating the choice of including the Derivative term of the controller, is the fact that in terms of phase, the derivative term allows for a $+90$ *degrees* phase lead. This proves particularly useful for spring-damper systems as they typically exhibit a phase lag due to inertia and damping factors, which as frequency increases, result in instabilities or poor transient responses.

However, the Derivative term comes with its own set of drawbacks, mainly the fact that it could make the system too sensitive to disturbances, especially those occurring at higher frequencies. Therefore as a mitigation strategy, a low pass filter has been implemented, to ensure overall stability even at higher frequencies. The transfer function of the PID controller is fully described by Equation 4.5.

$$PID(s) = K_p + K_i\left(\frac{1}{s}\right) + K_d\left(\frac{N}{1 + N * (1/s)}\right) \quad (4.5)$$

The next step is to discuss the procedure employed for gain tuning of the controller gains. Gain tuning of a PID controller can be a challenging procedure, especially if it is performed manually, as the relationship between benefits/downsides of increasing/decreasing a particular gain is non-trivial to decipher. To facilitate this procedure, the control architecture has been recreated in a Simulink (Matlab, Inc.) model, visible in Figure 4.7. It can be seen that the control logic follows a similar signal flow as the initial loop logic diagram in Figure 4.1. The main differences are found with respect to the input/output components of the system. The input of the system is a step response function, which has been chosen as this type of input covers a broad spectrum of frequency components as well as being capable of showing the transient and steady-state behavior of the system. The output of the system is directly fed to the scope function, which also feeds the reference angle feed into the system and the error feed into the controller (for plotting purposes).

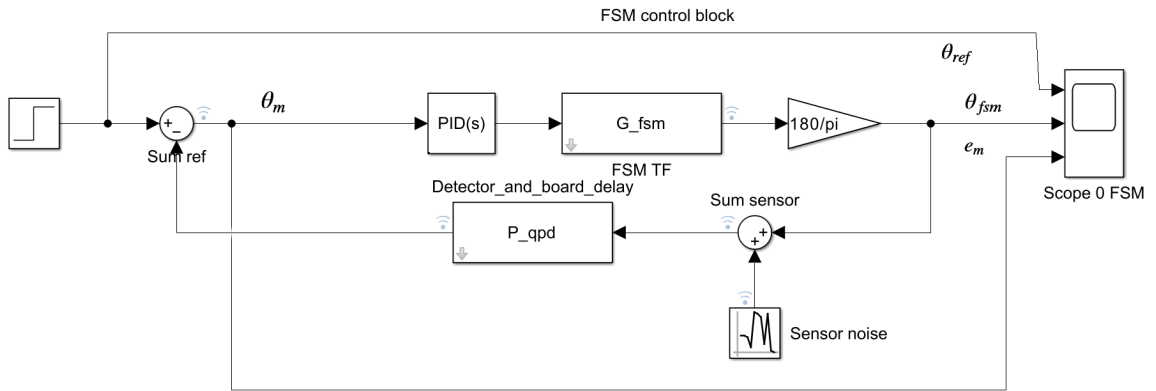


Figure 4.7: Simulink model of the PPAS including main control elements loaded as transfer functions in a separate Matlab script

Now with an understanding of the different elements of the Simulink model, it is now relevant to discuss the strategy employed to perform gain tuning of the PID controller. Individual gain tuning of a PID controller can be a challenging task, as the relationship between adjusting individual gains can have an impact on the non-adjusted gains, which can lead to undesirable/unstable control behavior.

To simplify this process the Matlab PID tuner application has been employed. The fundamental working logic of the tuner application is to start analyzing the system by identifying key characteristics of the plant model, these include aspects such as number of poles, zeros, time delays, etc. Next is the initial gain tuning process, it can involve multiple initial tuning methods, some of which include the Ziegler-Nichols, Cohen-Coon and AMIGO methods. For further information about the MATLAB PID tuner application, readers are referred to [20].

While the initial tuning procedure of the Gain tuner allows for setting some initial gains that provide an initial working point for the controller design, rarely do they already provide the precise system response sought after. Therefore the next step involves adjusting the gains, to end up with the desired frequency responsiveness and controller robustness.

The main design driving criteria has been to closely match the system's performance to closely match the performance specifications of the FSM actuator. In particular, the specification of a step response time (1 mrad) under 5 ms, is of interest. While the real-world performance of the FSM actuator likely could depend on the amplitude of the step response, the focus of tuning the linear model (which is invariant to different step input amplitudes) still lies in matching the response time. This is motivated by the fact that the critical aspect is to match the controller bandwidth to the actuator capacity limit.

Taking the performance specifications as a guiding reference point and adjusting the controller gains results in the following controller gains $K_P = 0.06111$ [-], $K_i = 6.8820$ [-], $K_d = 0.00013$ [-], while the filter has been manually hard set at $N = 1450$ Hz. The choice for the filter has been based on the research performed with respect to system identification of the FSM plant model [5], in which the remaining structural (resonant) nodes of the FSM have been observed to occur at frequencies above 1500 Hz. In Figure 4.8, the unit step response of the system can be seen.

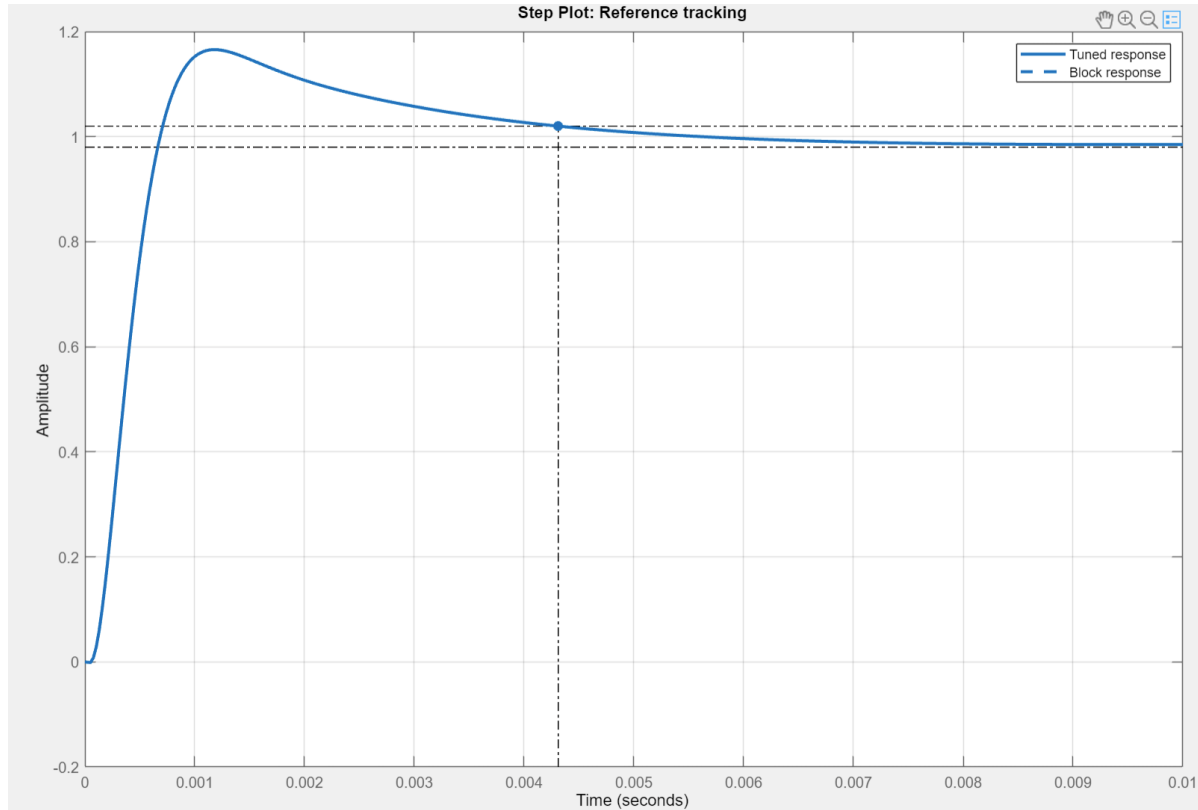


Figure 4.8: Tuned step response of the simulated PPAS system with tuned controller. Settling time is plotted with a 2% of overshoot

Another relevant plot to understand the effects of the controller on the system is the open-loop response (with the controller in the forward path) of the plant model. Visible in Figure 4.9, the open loop response of the complete system including time delays can be seen. The controller bandwidth has been set at 398 Hz. From the plot it is possible to recognize the critical stability points of the system, these indicate a phase margin and a gain margin of 61.2 Degrees and 16.5 dB, respectively. This points out that the system requires a sufficient amount of phase lag or decreases in gain, for it to become unstable, albeit remaining highly sensitive to systems delays with a delay margin of only 0.4 msec. **Note:** While the phase margin approaches the -180 degrees delay point, the combination of the positive gain margin and the soon lift in phase due to the derivative term help to keep the system closed loop stable.

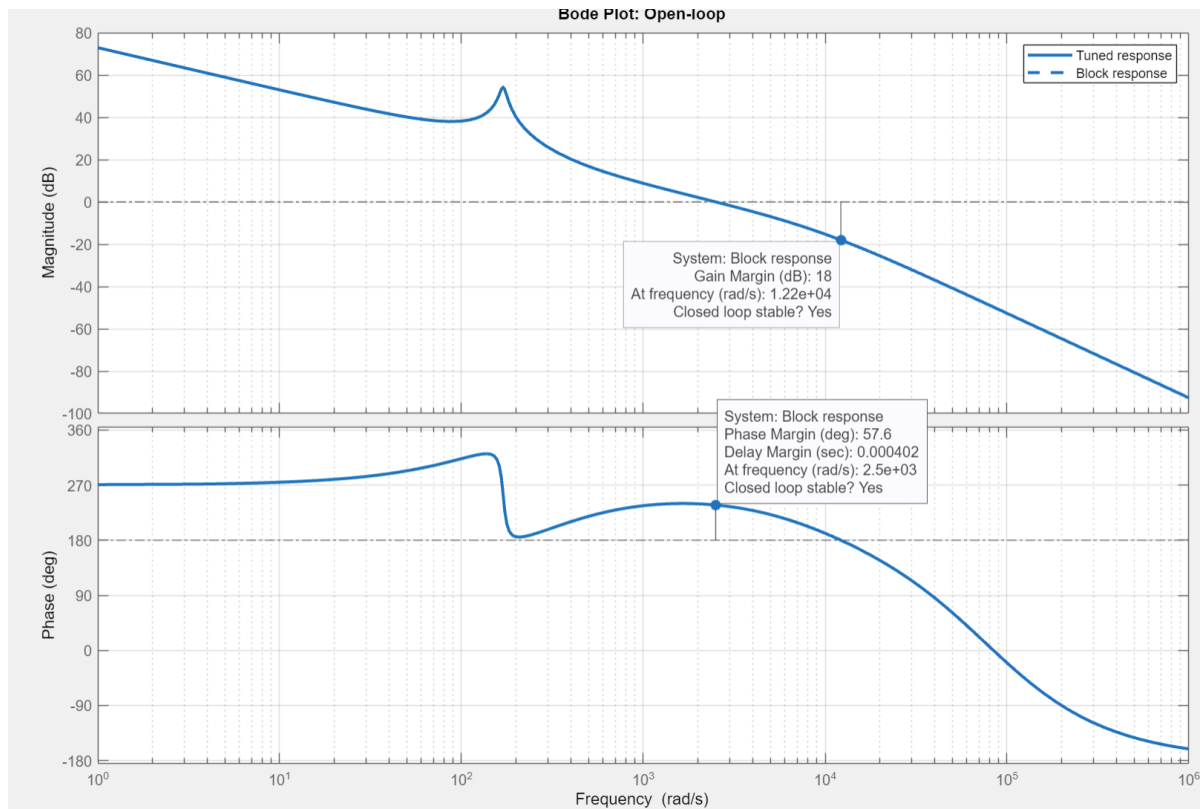


Figure 4.9: Resulting open loop system (with controller along the forward path) response Bode plot

This section concludes with the previous discussion of the open-loop response of the system, however for readers who are interested in gaining an understanding of the achievable noise suppression and closed loop response of the system a discussion with further information and Bode plots can be found in Appendix E under the sub-sections Augmented Open Loop -and Closed-Loop response and Augmented Open Loop -and Loop Sensitivity function part of the **Precise Pointing Actuator System accompanying plots, descriptions and code** main section.

4.2. Coarse Pointing Actuator System simulation model

This section aims to cover in detail the thought process behind the CPAS control design, to give insights and motivations for the different choices involved with the systems design. The section starts by covering the control loop architecture, it then covers the procedure employed for approximating the Coarse Actuator model. Next is the motivation and design of the PI controller, to which a Simulink model was created to aid with the gain tuning of the controller. Lastly, the last two subsections center on presenting the performance of the system by covering the Augmented Open loop, closed loop, Loop Gain and Loop sensitivity transfer functions and their frequency response. **Note** for the design and development of the CPAS control system it is assumed that there is no cross-correlation between actuator mount axis, as such a 1-DOF analysis is considered as sufficient for representing the overall behavior of the system.

Kicking off the discussion of the CPAS systems design is the control architecture, this is portrayed in Figure 4.10. The control logic follows a similar approach to the one taken by the PPAS, where there are 3 main subsystems paired with a gain for converting the output of the actuator plant model into *Degrees*. The loop logic starts with the reference angle, to this signal the previous measurement on the position of the actuator model gets subtracted.

This allows to determine the current error between the desired position and the current position, which allows the controller to either send a control command or if the error is negligible to keep the system in its current position. Lastly, the control input in the form of torque is applied to the actuator system which has been tuned to provide a representative response of the modeled system, resulting

in a displacement that can be measured in the feedback path.

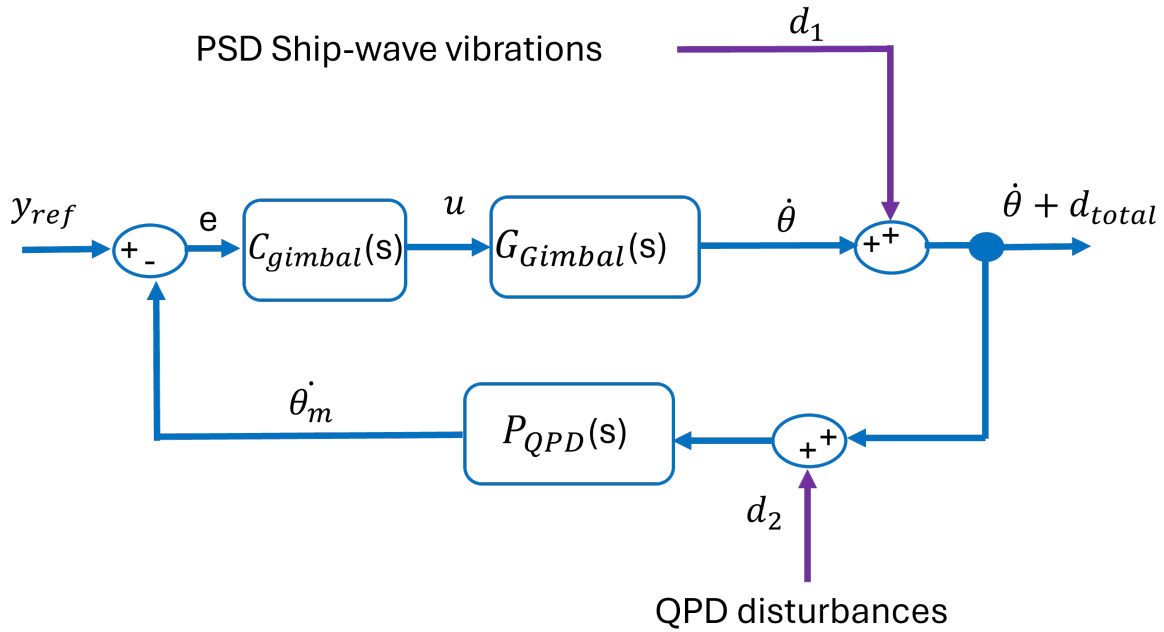


Figure 4.10: Overview of the control design architecture of CPAS

Note The transfer function of the QPD is identical to the one covered in the PPAS section, as it represents the main measurement system for both systems. Therefore the discussion of the QPD transfer function is omitted in this section.

4.2.1. Coarse Actuator model

Now with an intuitive understanding of the general CPAS control logic architecture next is to cover the methodology utilized to derive the Coarse Actuator model. It is relevant to highlight the fact that the Coarse Actuator model has been centered on recreating the dynamics of the L-350 direct drive telescope, as in compliance with requirement **PATR-01-02**. However, as of writing, limited information could be found regarding system-identified models that recreate the dynamics of the L-350 direct drive telescope.

At first, it was attempted the creation of a transfer function that could approximate the dynamics of the L-350 direct drive telescope, via system identification mapping input to output data. However, as the input/output data was recorded via a high-level software command interface, rather than the direct application of the voltages into the direct drive motors of the L-350, it was then determined that this procedure could involve a high degree of uncertainty. Further information regarding this approach can be found in Appendix F

Since system identification has been determined as an unsuitable option for this work, the decision was made to perform a simple approximation of the L-350 telescope mount by estimating the rotational inertia of the mount. The fundamental idea is to determine the masses acting on the mount and their corresponding moments of inertia, to estimate the rotational moments of inertia, which then allows mapping torque inputs to angular displacements via an LTI transfer function. The main assumptions include; the rotational moment of inertia of the elevation axis is fully described by the dimensions and mass of a dummy telescope with an aperture of 30 cm and a mass of 20 kg, the telescope is mounted at the center of the elevation rotational axis, the telescope and L-350 mount can be represented by simple geometrical shapes, both the telescope and mount behave as a rigid body which is stiff enough to not deflect during operation, uniform material density such that the center of mass is equal to the geometrical center point and the center of mass of the lower mount base is located at the center of the

azimuth rotational axis.

This process starts by searching for the official mass specifications of the mount, which can be found on [48], in which the fork arm and elevation motor are specified to have a mass of 70 *Lbs* (≈ 31.75 *kg*) and the base and azimuth motor 40 *Lbs* (≈ 18.1 *kg*), in addition to these masses a dummy telescope payload with a mass of 20 *kg* is also accounted in the rotational moments of inertia.

The next step is to approximate the shape of the L-350 telescope mount, to reduce the complexity of the calculation of the the rotational moments of inertia. As visible in the right section of Figure 4.11, the telescope mount has been approximated by 2 rectangles, while the dummy payload telescope is by a circular cylinder. In the bottom right corner, it is possible to see an assumption made by the approximation, in which the red rectangle corresponding to the azimuth base is assumed that its center of mass is located in the center of the azimuth axis of rotation. Similar to the azimuth base, the green cylinder representing the dummy payload its center of mass is assumed to align with the elevation axis.

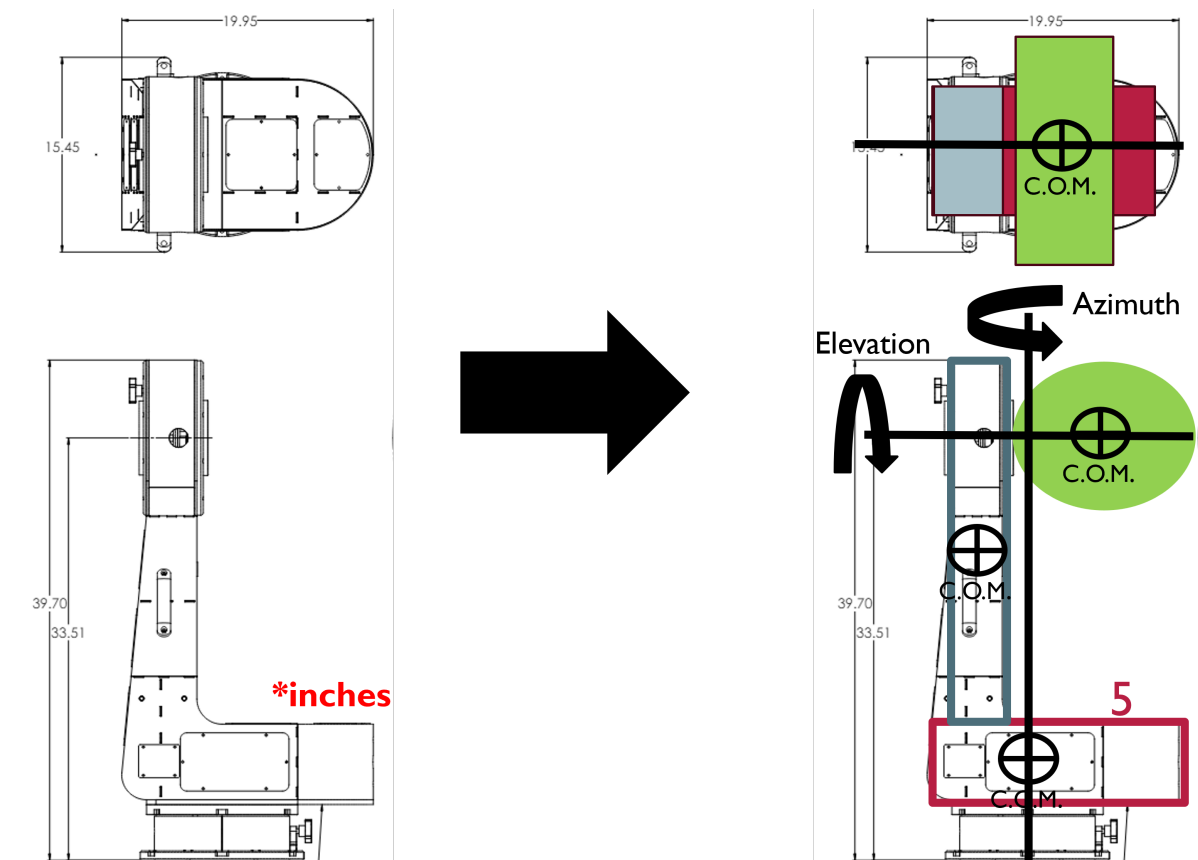


Figure 4.11: PlaneWave instruments L-350 Direct Drive mount approximation via rectangular boxes

Now with the approximation of the L-350 mount in place next is the determination of the individual and total contributions of the multiple blocks, to obtain the rotational moment of inertia in the elevation and azimuth directions, a description of this procedure can be found in Appendix E under the Estimation of rotational moments of inertia of the Coarse Pointing Actuator System subsection part of the Coarse Pointing Actuator System accompanying plots and descriptions main section. Following the outcomes discussed in the appendix, Equation 4.6 and Equation 4.6 represent the estimated rotational moments of inertia and their corresponding transfer functions determining the rotational displacement of the L-350 mount in the azimuth and elevation directions. Note the expression of the LTI transfer functions has been inspired by the work of [12], in which modeling and simulation of an inertially stabilized gimbal system has been investigated.

$$G_{gimbal} = \frac{1}{J_{az} * s} \quad (4.6)$$

$$G_{gimbal} = \frac{1}{J_{el} * s} \quad (4.7)$$

Where $J_{az} = 2.264 \text{ kg} * \text{m}^2$ and $J_{el} = 0.930 \text{ kg} * \text{m}^2$. To obtain a better understanding of the frequency response of each of the actuator models it is possible to refer the readers to Figure 4.12 and Figure E.4 (found in Appendix E in the Elevation open loop response subsection), where the Bode plots of both actuator models can be seen. Notable from these plots are the effects of the system being approximated by a rotational inertia and a pure integrator, this is visible in the phase margin which remains at a constant -90degrees , this is also reflected in the gain margin where there is a linear constant decrease at an expected rate of -20 dB/decade . The main notable difference across these plots is the fact that each crosses the negative gain axis at different points, which is likely caused by the difference in rotational inertia across the models.

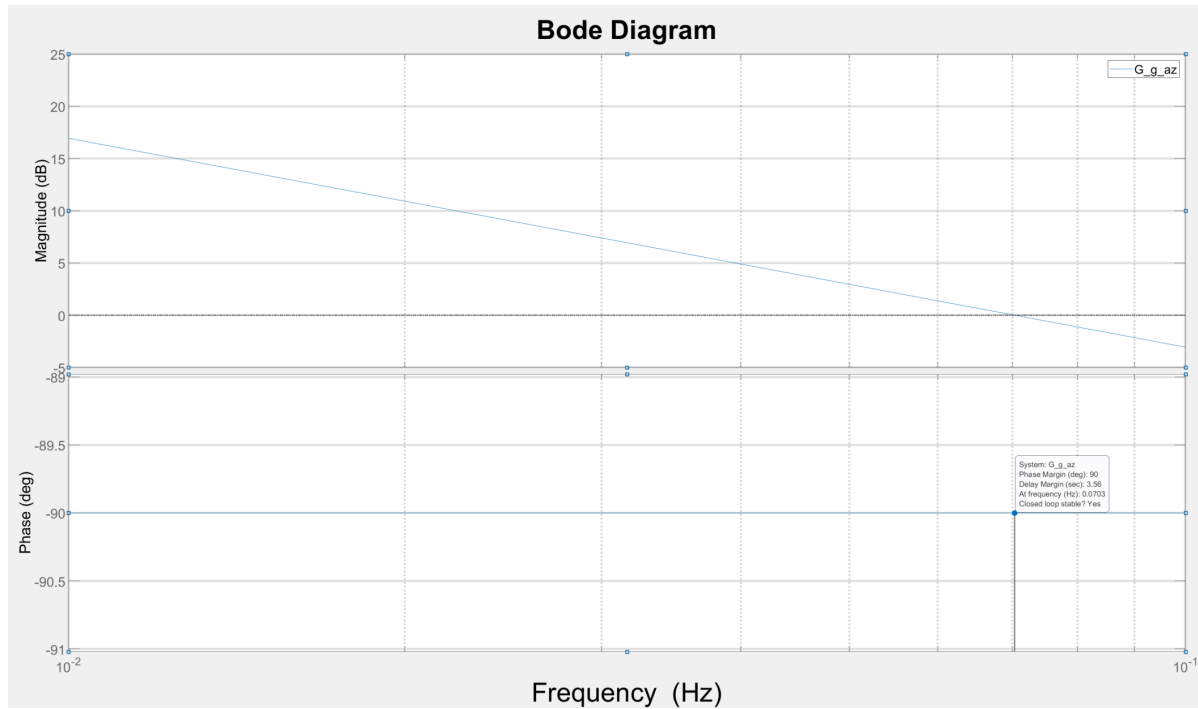


Figure 4.12: Bode plot of the frequency response of the CPAS azimuth actuator transfer function

4.2.2. PID controller design

With both actuator transfer functions having been discussed, the next step is to discuss the design of the PID controller for the CPAS. Since the PID transfer function is identical to the one of the PPAS, it is possible to start discussing directly the tuning of the controller. Utilizing the same approach as in the previous section, a Simulink model was created to facilitate the gain-tuning procedure. As visible in Figure 4.13, the control logic follows very closely to the original control logic diagram found at the beginning of this section, with the main difference being the step input and scope functions. **Note:** In the figure two identical models can be seen, as each one is dedicated to tuning independently the gains of the azimuth and elevation axis.

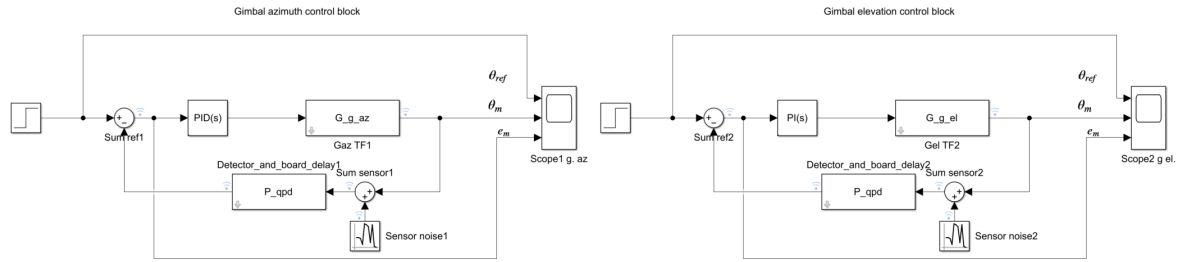


Figure 4.13: Simulink model of the CPAS control system for gain tuning

With the Simulink model and utilizing the MATLAB PID tuner application, it was then possible to tune the response of the system. As a performance baseline, the controller bandwidth has been adjusted to 10 Hz , which represents a common operational bandwidth for commercial telescope gimbals, moreover to smooth out the controller output torques, the controller has been adjusted to a phase margin of 90 degrees . The combination of the controller bandwidth and phase margin results in a proportional, integral and derivative gain for the azimuth axis of 136.63, 1525.59 and 0.346, while the elevation axis results in gains of 55.91, 622.19 and 0.156, with the corresponding high-disturbance filter of 229.13 (rad/sec) for both controllers. The step response to these gains can be seen in Figure 4.14, it can be seen that the system is capable of reaching a settling time of $\approx 0.22\text{ sec}$ with a marginal amount of overshoot, which could be the result of the high phase margin built into the controller.

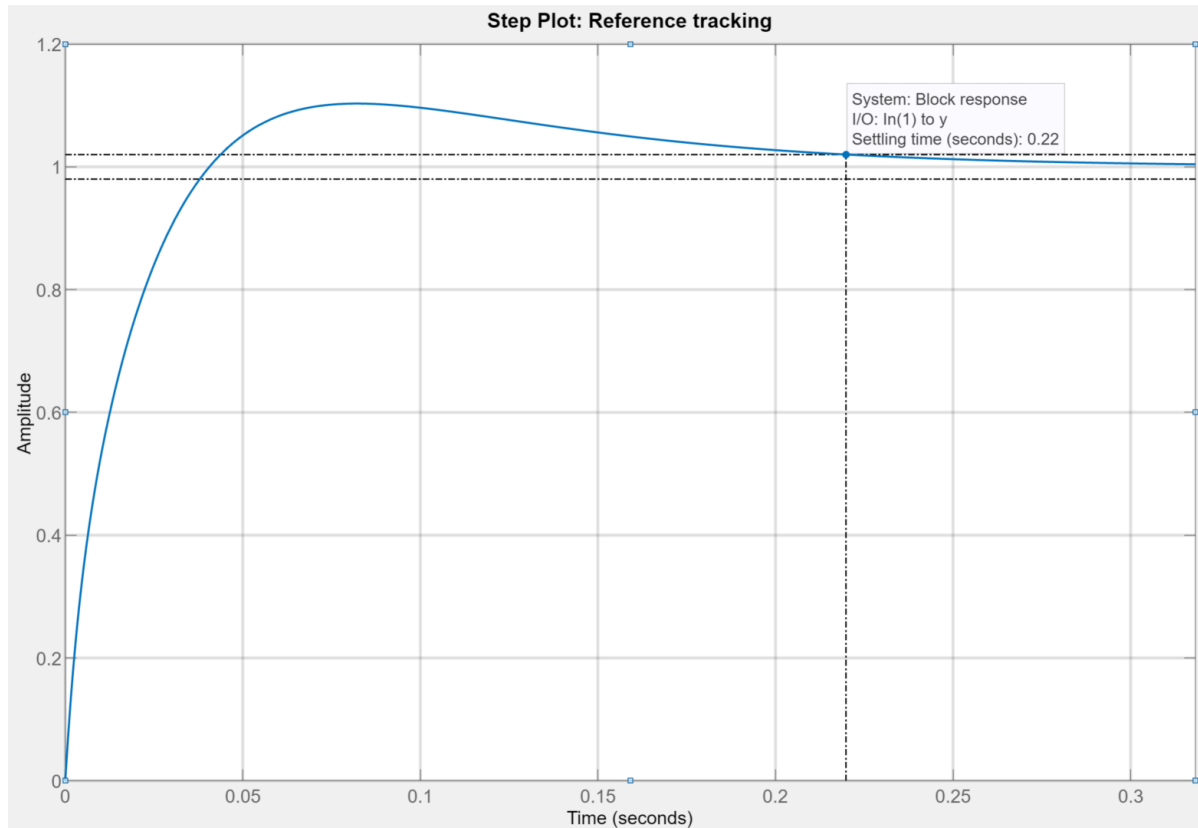


Figure 4.14: Tuned step response of the CPAS actuator control system. Note that the settling time is based on reaching 98% of the final desired value

By providing information regarding the open-loop response of the system this section aims to conclude, however for readers who are interested in gaining an understanding of the achievable noise suppression and closed loop response of the CPAS a discussion with further information and Bode plots can be found in Appendix E under the sub-sections Augmented Open Loop -and Closed-Loop

response and Augmented Open Loop -and Loop Sensitivity function part of the **Coarse Pointing Actuator System accompanying plots and descriptions** main section.

4.3. Dual-Stage Actuator System simulation model

The aim of this section is to describe the design choices and control loop logic of the Dual-Stage Actuator System (DSAS). The section starts by introducing the loop logic, including the motivation for the chosen control methodology. By covering the design choices and due to the re-utilization of tuned controllers, the next two subsections center on the discussion of the derivations of the Augmented Open loop, Closed loop and Loop sensitivity system response functions. These provide insights into the system response and disturbance rejection capabilities. Next is the Loop sensitivity subsection which centers on distinguishing the differences and similarities between the loop sensitivity functions of the PPAS, CPAS and DSAS. Lastly, the chapter ends with a discussion of the Simulink simulation model of the DSAS, to investigate the step response of the system as well as the off-loading capacity of this control configuration.

4.3.1. Dual-Stage Actuator System model

The objective of this subsection is to cover in further detail the design methodology and assumptions of the Dual-Stage Actuator System (DSAS) control architecture. The fundamental idea of the DSAS is part of the chief-deputy control architecture inspired by the works of [6], [44] and [32], which center on the control architecture of Dual-stage controller systems for Hard Disk Drives. The commonality across these Dual-Stage control systems is that they all tackle the limitations imposed by actuators with high actuator bandwidth (slew rate) but low angular range, such that the end system has both a high actuator bandwidth and high angular range. The way they tackle this is via the introduction of a chief-deputy control architecture, which pairs the high bandwidth actuator with a low bandwidth actuator with a high angular range.

The fundamental assumption built into the chief-deputy control architecture is the accurate prediction of the high-bandwidth actuator position. For this work, the prediction is only linear and fully based on the same transfer function as the one of the high-bandwidth actuator. Basing the prediction on a fully linear prediction can be justified based on the fact that the whole control logic of the chief-deputy system is designed to keep the high-bandwidth actuator as close as possible to the zero displacement position, this aims to avoid the non-linearities typically encountered at the limits of the angular range. The designed control logic path of the DSAS can be seen in Figure 4.15.

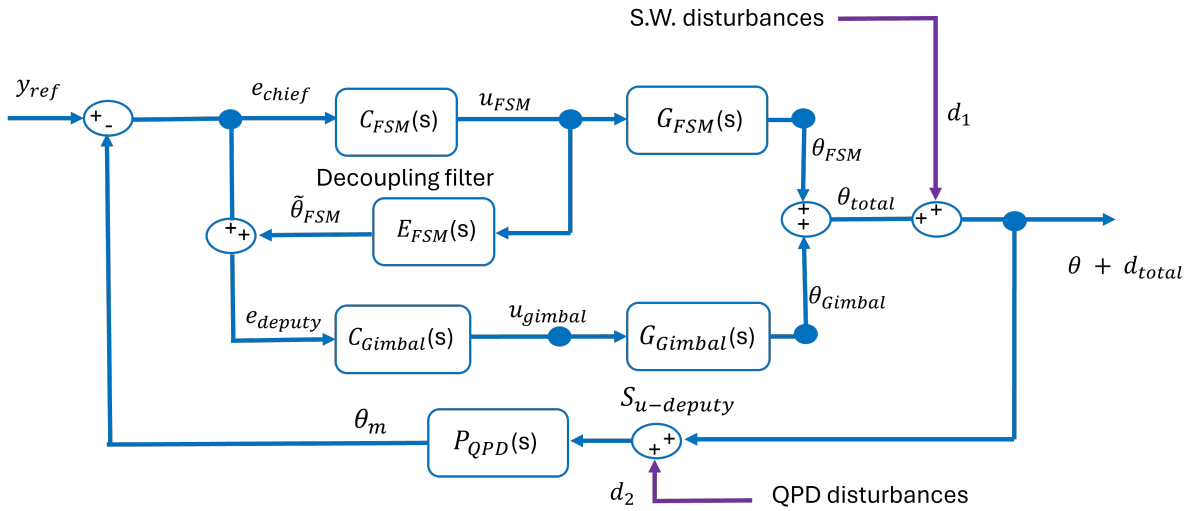


Figure 4.15: Dual-Stage Actuator System Control Logic architecture

As visible in Figure 4.15, the working principle of the dual-stage control logic lies in the combination of both outputs of the high-bandwidth actuator (G_{fsm}) and the low-bandwidth actuator (G_{Gimbal}).

Starting from the reference angle the path gets split into the "Chief" path and the "Deputy" path. In the Chief path, the total error in the system gets directly fed into the chief controller which then places a control input for the high-bandwidth actuator. The objective of this path is to minimize the tracking error by making use of the higher bandwidth of the high-bandwidth actuator. For the controller logic of this work, the OIM5001 FSM has been selected as the chief actuator due to its high angular-rate bandwidth, which allows for faster response times when compared to the L-350 telescope gimbal (coarse actuator).

Next is the Deputy control path, which at its input receives the total tracking error of the system together with the estimate of the high-bandwidth actuator position. The aim is to combine both inputs, such that the low-bandwidth system can start working on removing both the total tracking error of the system and start off-loading the high-bandwidth actuator, by taking into account its position. To achieve this the total error in the system (e_{chief}) as well as the estimate of the high-bandwidth actuator position (E_{fsm}) are both added up and fed as the deputy error (e_{deputy}) into the deputy controller (C_{fsm}). Next, the control input of the deputy controller is sent to the low-bandwidth actuator (G_{gimbal}) and its output (θ_{fsm}) is added together with the output of the high-bandwidth actuator (θ_{fsm}), to acquire the total output of the system (θ_{total}). Lastly is the feedback path which passes through the QPD tracking sensor (P_{QPD}) which represents the sampling time delay of the system and ultimately its feedback to compare the current system position to the desired reference position.

Equation 4.8 and Equation 4.9 are introduced as short-hand notations of the chief and deputy controller sections, as to limit the amount of space required for the subsiding systems response transfer functions. **Note:** to save space G_{gimbal} can represent either elevation and azimuth actuator models, this is motivated by the fact that it is relevant for this section to investigate the overall system performance, since both elevation and azimuth models are tuned to the same controller bandwidth and phase margin, they both produce the same system frequency responses.

$$P_{fsm}(s) = C_{fsm}(s) * G_{fsm}(s) \quad (4.8)$$

$$P_{gimbal}(s) = C_{gimbal}(s) * G_{gimbal}(s) \quad (4.9)$$

4.3.2. Augmented Open loop and Closed loop response

This subsection aims to discuss the equations utilized to determine the Augmented Open Loop response and the Closed loop response of the DSAS. These equations are critical to determine the type of performance which can be expected of the DSAS. Equation 4.10 describes the Augmented Open Loop response of the system. **Note:** These transfer functions make use of the in-built assumption of the controller being capable of accurately predicting the precise pointing actuator position. This assumption is carried over for the determination of Equation 4.11 which describes the Closed loop response and for the rest of the transfer functions part of the DSAS control design.

$$\begin{aligned} A.O.L.(s) &= [P_{fsm}(s) + P_{gimbal}(s) * (1 + C_{fsm} * E_{fsm})] * P_{qpd} \\ &= [P_{fsm}(s) + P_{gimbal}(s) + P_{fsm}(s) * P_{gimbal}(s)] * P_{pqd} \end{aligned} \quad (4.10)$$

$$C.L.(s) = \frac{P_{fsm}(s) + P_{gimbal}(s) + P_{fsm}(s) * P_{gimbal}(s)}{1 + [P_{fsm}(s) + P_{gimbal}(s) + P_{fsm}(s) * P_{gimbal}(s)] * P_{pqd}} \quad (4.11)$$

By making use of the Augmented Open loop and Closed loop functions it is then possible to determine the frequency response of the DSAS. As visible in Figure 4.16, the augmented open loop response is quite similar to the response of the PPAS, which is the dominant (Chief) controller. The effective bandwidth displays a similar behavior of a large gain margin at lower frequencies, indicating adequate performance of the system at lower frequencies. The Closed Loop response has a bandwidth close to the Chief controller bandwidth close to 377 Hz.

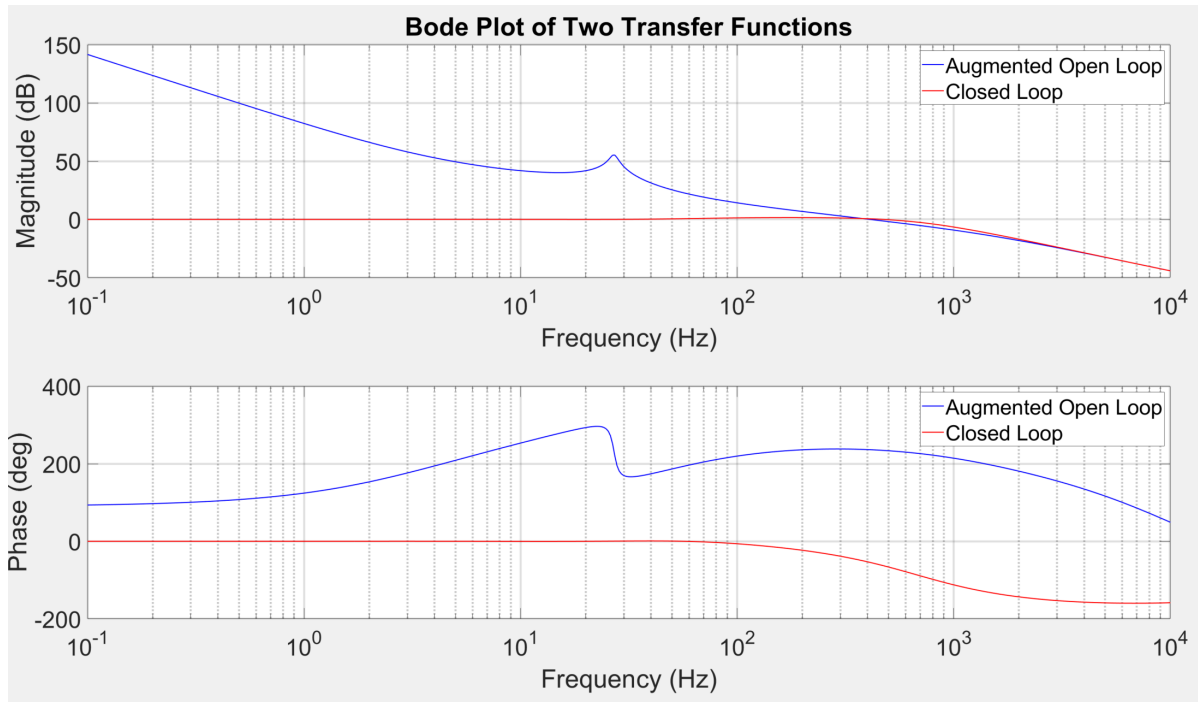


Figure 4.16: Augmented open loop and closed loop frequency responses of the DSAS shown as Bode plots

4.3.3. Augmented Open Loop and Loop sensitivity response

The next step is to determine the disturbance rejection capabilities of the DSSAS, to determine this, it is then critical to determine the Loop sensitivity function of the system. This is fully described on Equation 4.12, where the simplification due to accurate prediction of the high-bandwidth actuator is utilized to reduce the terms present in the equation.

$$L.S.(s) = \frac{1}{1 + [P_{fsm}(s) + P_{gimbal}(s) + P_{fsm}(s) * P_{gimbal}(s)] * P_{pqd}} \quad (4.12)$$

As visible in Figure 4.17, the Loop Sensitivity function response of the DSSAS follows closely the response of the Augment Open Loop function, with similar performance to the one of the PPAS at higher frequencies. While at lower frequencies the combined performance of both actuators allows for a higher gain margin relative to the one achievable by either the CPAS or PPAS on their own. From the information provided by these plots, one could argue that the DSAS potentially has adequate command processing capabilities at lower frequencies and decent noise attenuation at higher frequencies.

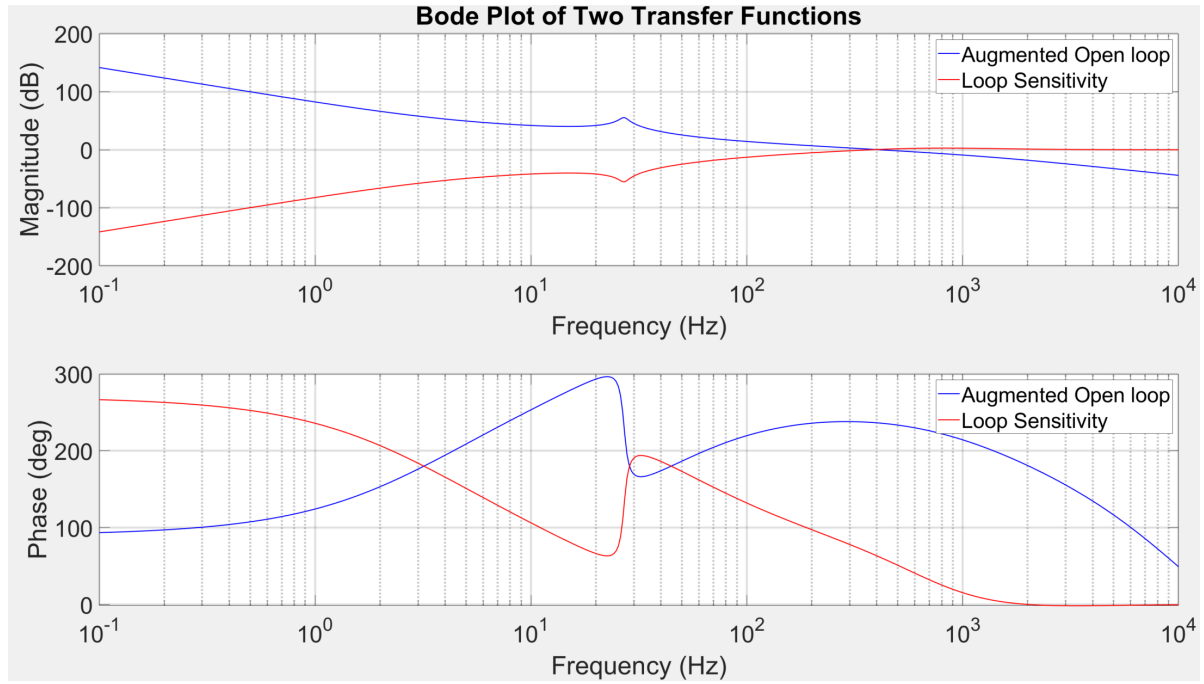


Figure 4.17: Augmented open loop and loop sensitivity frequency responses of the DSAS shown as Bode plots

4.3.4. Comparison of loop sensitivity functions

Due to similarities in the performance of the DSAS loop sensitivity function and the PPAS loop sensitivity function, it is then relevant to investigate relative to each system the level of disturbance rejection capabilities of each system. This is visible in Figure 4.18, where bode plots the loop sensitivity functions of the PPAS, CPAS and DSAS are visible. It is possible to argue that due to the much higher operational bandwidth of the chief controller, it is to be expected for the loop sensitivity function of the DSAS to approximate the one of the PPAS. However it is relevant to point out that the noise disturbance bandwidth at lower frequencies of the CPAS is added to the DSAS, increasing its capacity to neglect disturbances at lower frequencies.

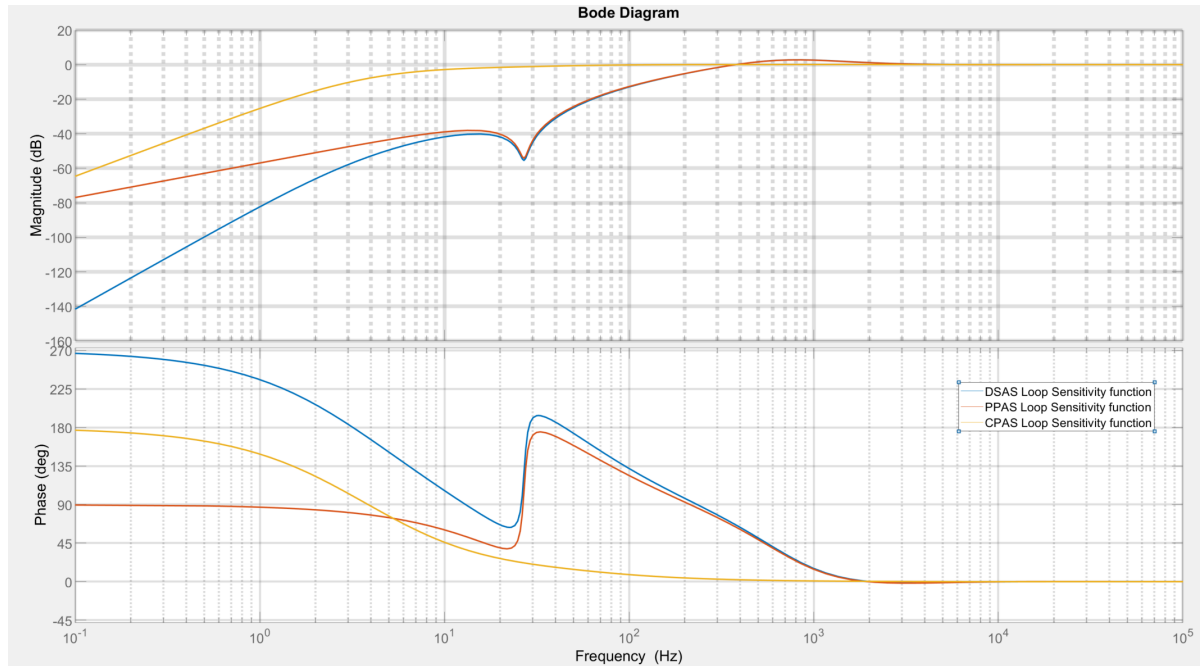


Figure 4.18: Bode plot of the comparison between the multiple sensitivity functions of the PPAS, CPAS and DDAS

4.3.5. Simunlink model for actuator off-loading verification

Next is the determination of the Chief and Deputy actuator performance, to assess if the control logic design of the DSAS properly off-loads the high-bandwidth actuator, the Simulink model shown in Figure 4.19 has been developed. It incorporates a similar loop logic as the one shown in Figure 4.15, with the addition of the step reference input (unit amplitude) and the sensor noise representing a tracking error of $88.6 \mu\text{degrees}$. **Note:** In this case the model is performed with a single actuator model for the gimbal actuator, as it is relevant to determine the controller response rather than the controller output torques (which are axis dependent).

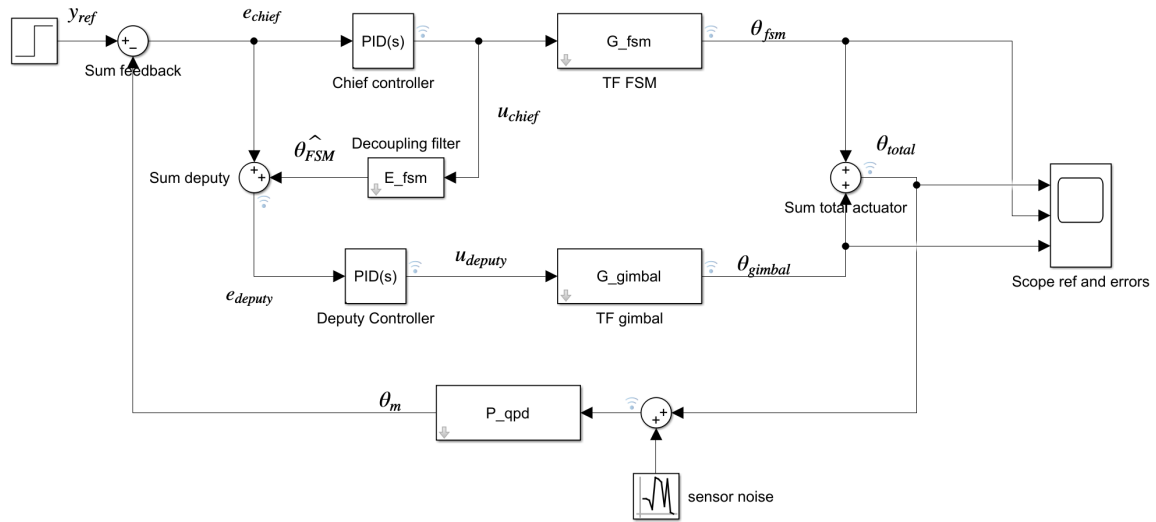


Figure 4.19: Simulink model of the DSAS incorporating a step response and tracking sensor disturbances

Next is the determination of the actuator response of the total system as well as the ones of the chief and deputy actuators. This is shown in Figure 4.20, where the step response of the DSAS can be seen. It is possible to remark that the total displacement of the system follows closely into the pure response of the PPAS, however as time progresses, the deputy actuator slowly is able to respond to the reference command and start offloading the displacement of the chief actuator. It is noteworthy that to enhance the transition the controller bandwidth of the deputy controller could be increased, to potentially minimize the possibility of the chief actuator reaching its angular range limits.

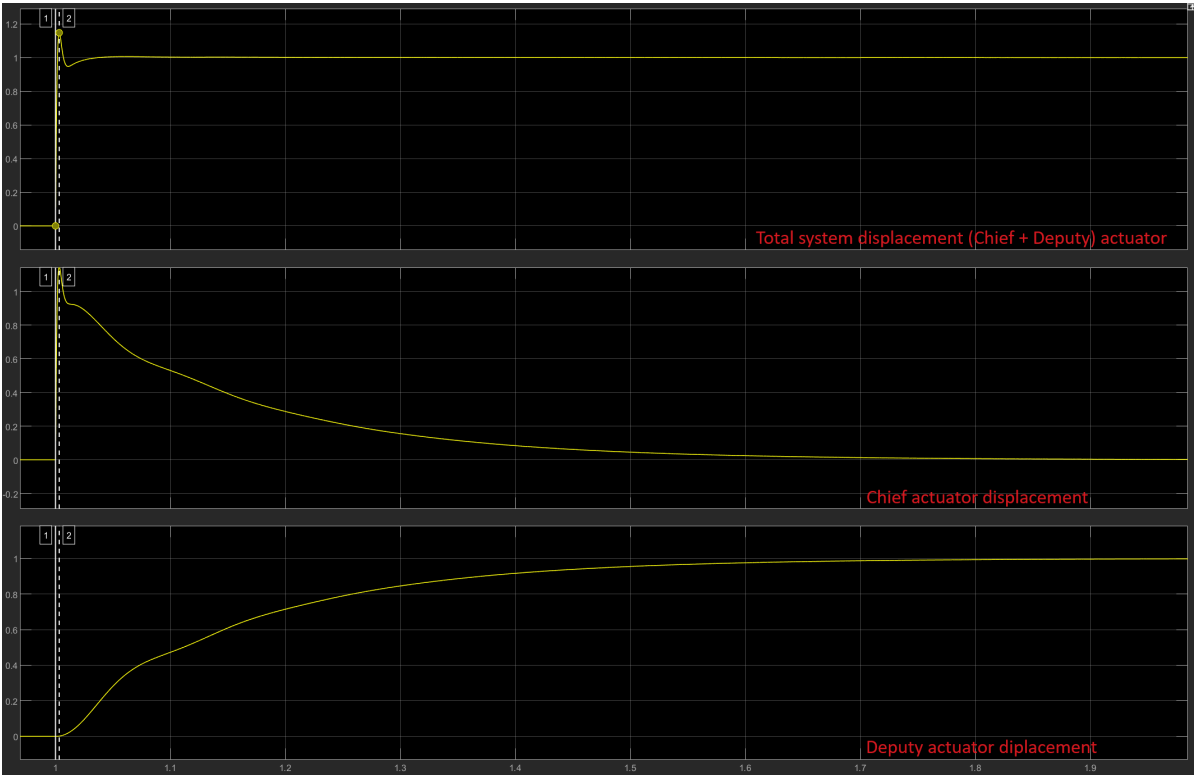


Figure 4.20: Step response of the DSAS simulink model response, shown at the top plot is the total displacement of the system, middle plot represents the chief actuator displacement and lastly, the bottom plot represents the deputy actuator displacement

5

Dynamic Error Budgeting analysis of the simulation models

This section aims to discuss the results obtained via the Dynamic Error Budgeting (DEB) design approach. For readers unacquainted with the DEB design approach, a brief introduction to the method and its key assumptions is provided in Appendix G under the **Brief introduction to Dynamic Error Budgeting analysis** section. Moreover, to center the core of the report, the results obtained from performing DEB analysis on the PPAS and CPAS are summarized in the **PPAS and CPAS Dynamic Error Budgeting analysis summary of results** section. While the detailed discussion of the DEB loop logic, equations, angle and angle-rate results are provided in Appendix G, under the **PPAS Dynamic Error Budgeting analysis** and **CPAS Dynamic Error Budgeting analysis** sections.

With the main research outcomes of the individual stages presented, the chapter advances to the DSAS Dynamic Error Budget analysis section, where the DEB test logic, DEB performance-specific functions are discussed. To limit the length of this section, the DEB frequency response plots and individual results are provided in Appendix G under the **DSAS DEB analysis performance and plots** section.

5.1. PPAS and CPAS Dynamic Error Budgeting analysis summary of results

Before discussing the results of the DSAS, it is relevant to justify the need for a dual-stage system, therefore in this section aims to present the performance results of the PPAS and CPAS, to provide the reader with information, as to why each system individually is not capable of fulfilling the performance requirements to perform FSOC on board a frigate-sized ship. Note that for the DEB results of the CPAS analysis, an additional assumption has been performed, that is a small angle approximation with respect to the pitch, roll and yaw angles of the boat remaining low enough such that the disturbances caused by the pitch and roll motions can be compensated purely by the elevation actuator model, while the yaw motions purely by the azimuth actuator model.

Visible in Table 5.1 are the performance results from the DEB analysis, notable from these results are the pointing angle and pointing angle-rate errors, which are crucial to comply with PATR-01, here the PPAS is capable of delivering adequate performance to stay well below the $50 [\mu rad - \frac{\mu rad}{sec}]$ pointing requirement, moreover here is where the CPAS encounters severe challenges, as it is unable of fulfilling these pointing requirements, even for the higher bandwidth controller setting of the CPAS.

The next performance requirement that is relevant to motivate the need for a dual-stage system is the angular and angular-rate efforts, which in this case are well within the specifications of the CPAS, however the angular effort required for the PPAS is beyond the angular range achievable by the actuator which tops at $\pm 3 \text{ degrees}$ [38]. Since both systems are incapable of both meeting pointing

requirements as well as actuator performance specifications, the individual systems can be considered on their own as unsuitable for performing the task at hand, therefore motivating the choice of combining these systems into the dual-stage system. **Note:** There is missing information regarding torque and power limitations for the PPAS actuator, therefore it would be recommended in future works to determine if the performance requirements fall within the specifications of the actuator. The controller torque and power requirements are within the specifications of the CPAS actuator, being rated for a maximum-continuous torque of $20 \text{ N} * \text{m}$ and having a maximum power supply of 1800 W (120 VAC 15A IEC Type B Regulated Power Adapter [48]).

Table 5.1: PPAS and CPAS DEB analysis performance results

Total RMS (3σ) error/performance req.	PPAS (398 Hz)	CPAS (10 Hz)	CPAS (500 Hz)
Pointing angle error [μrads]	37.62	493.20	20.49
Pointing angler effort [$d\mu\text{rad}/\text{sec}$]	28.21	2531.80	280.88
Torque [$\text{N} * \text{m}$]	0.014	7.225	12.085
Power [W]	0.165	18.37	94.267
Angular effort [degrees]	9.06	9.06	9.06
Angular-rate effort [$\text{degrees}/\text{sec}$]	6.693	6.696	6.696

5.2. DSAS Dynamic Error Budgeting analysis

This section aims to cover the DEB analysis design and results of the DSAS. It starts by presenting the DEB loop logic, where the disturbance inputs and output performance locations are covered, next are the DEB DSAS equations, in which the total actuator performance, individual (chief & deputy) controller/actuator performance and loop-specific sensitivity functions are covered.

The DSAS DEB loop logic diagram is shown in Figure 5.1, it is relevant to note that the sources of disturbances are introduced once into the system, contrary to adding it to each of the chief/deputy actuator paths, the same is applicable to the tracking sensor error. Visible in the figure are the 5 different system performance outputs, these are in place to determine the individual performance of specific elements part of the DSAS.

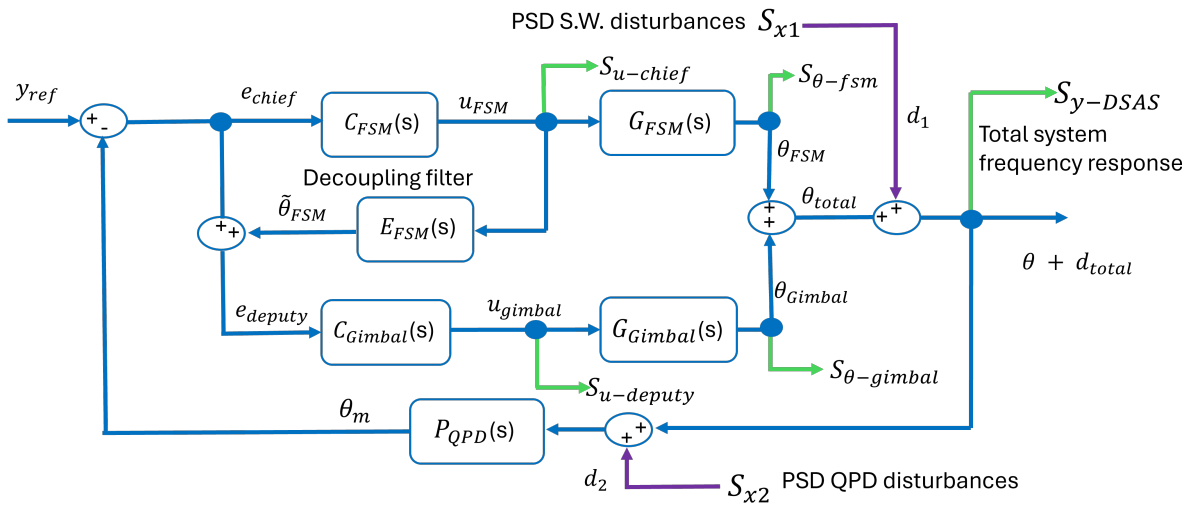


Figure 5.1: DSAS DEB analysis loop logic

5.2.1. DSAS DEB equations

With the DSAS loop logic covered, the multiple DEB closed-loop performance equations are determined next. As the DEB equations of the DSAS can get lengthy, Equation 5.1 and Equation 5.2 provide a short-hand notation for the combination of the chief/deputy controller and actuator paths.

$$P_{fsm} = C_{fsm} * G_{fsm} \quad (5.1)$$

$$P_{gimbal} = C_{gimbal} * G_{gimbal} \quad (5.2)$$

The main DEB closed-loop output performance for the entire combined system is described by Equation 5.3. Note that due to the fact that the disturbance inputs are placed within the total combined forward and feedback paths only two loop-specific sensitivity functions are required.

$$S_{Y_{DSAS}}(\omega) = |H_{dsas}(j2\pi\omega)|^2 * S_{x1}(\omega) + |H_{dsas-ts}(j2\pi\omega)|^2 * S_{x2}(\omega) \quad (5.3)$$

It is relevant to highlight that for Equation 5.4, the total complete loop sensitivity function is described first as a direct translation from the DEB DSAS loop logic diagram. However the equations can be grouped into a simpler representation, via the assumption that the chief actuator output can be estimated accurately, that is $E_{fsm} = G_{fsm}$. This assumption is applied with the rest of the DEB equations to simplify notation.

$$\begin{aligned} H_{dsas}(j2\pi\omega) &= \frac{1}{1 + [P_{fsm} + (1 + E_{fsm} * C_{fsm}) * P_{gimbal}] * P_{QPD}} \\ &= \frac{1}{1 + [P_{fsm} + P_{gimbal} + P_{fsm} * P_{gimbal}] * P_{QPD}} \end{aligned} \quad (5.4)$$

$$H_{dsas-ts}(j2\pi\omega) = H_{dsas} * [P_{fsm} + P_{gimbal} + P_{fsm} * P_{gimbal}] * P_{QPD} \quad (5.5)$$

Next is the determination of the closed-loop output performance of the chief and deputy actuators, these are described by Equation 5.6 and Equation 5.8. Note that loop-specific sensitivity functions of the chief/deputy actuators Equation 5.7 and Equation 5.9 only modify the total loop sensitivity function by the logic path, tracing back to the disturbance inputs, therefore for the case of the chief actuator is a single logic direction, while the deputy actuator passes through the combination of the direct total system error path and the chief actuator estimation path.

$$S_{Y_{DSAS-\theta_{chief}}}(\omega) = |H_{dsas-\theta_{chief}}(j2\pi\omega)|^2 * (S_{x1}(\omega) + S_{x2}(\omega)) \quad (5.6)$$

$$H_{dsas-\theta_{chief}}(j2\pi\omega) = H_{dsas} * P_{fsm} * P_{QPD} \quad (5.7)$$

$$S_{Y_{DSAS-\theta_{deputy}}}(\omega) = |H_{dsas-\theta_{deputy}}(j2\pi\omega)|^2 * (S_{x1}(\omega) + S_{x2}(\omega)) \quad (5.8)$$

$$H_{dsas-\theta_{deputy}}(j2\pi\omega) = H_{dsas} * [(1 + P_{fsm}) * P_{gimbal}] * P_{QPD} \quad (5.9)$$

Lastly are the DEB closed-loop performance equations of the chief/deputy controllers, these are fully described by Equation 5.10 and Equation 5.12. While the loop-specific sensitivity functions are described by Equation 5.11 and Equation 5.13.

$$S_{Y_{DSAS-U_{chief}}}(\omega) = |H_{dsas-u_{chief}}(j2\pi\omega)|^2 * (S_{x1}(\omega) + S_{x2}(\omega)) \quad (5.10)$$

$$H_{dsas-U_{chief}}(j2\pi\omega) = H_{dsas} * C_{fsm} * P_{QPD} \quad (5.11)$$

$$S_{Y_{DSAS-U_{deputy}}}(\omega) = |H_{dsas-u_{deputy}}(j2\pi\omega)|^2 * (S_{x1}(\omega) + S_{x2}(\omega)) \quad (5.12)$$

$$H_{dsas-U_{deputy}}(j2\pi\omega) = H_{dsas} * [(1 + P_{fsm}) * C_{gimbal}] * P_{QPD} \quad (5.13)$$

Now with the DEB equations in place, it is then possible to perform the DEB analysis for the DSAS, a dedicated section to discussing the base performance configuration and frequency response plots of the DSAS with a chief controller working at 398 Hz and a deputy controller working at 10 Hz can be found in Appendix G under the **DSAS DEB analysis results and figures** section.

5.3. Summary of Dynamic Error Budgeting analysis results and requirements verification

This section aims to conclude the chapter by summarizing key performance results obtained via the DEB analysis design approach. The results center on the discussion of several control bandwidth configurations of the DSAS, to provide information regarding the performance specifications required to comply with **PATR-01**.

There are two main performance configurations, the base configuration utilizes a chief controller operating at a bandwidth of 398 Hz (this is the bandwidth fitted to 1 mrad per 1 ms , discussed in chapter 4 under the PPAS control model section), while the deputy controller bandwidth ranges from 10 Hz to 500 Hz . The second configuration utilizes a chief controller operating much closer to its bandwidth limit at 765 Hz which is under 10% of the maximum bandwidth (3 dB bandwidth $>850\text{ Hz}$ [38]) with the deputy controller bandwidth varying along the same frequency ranges as the base configuration.

In Table 5.2, the total RMS (3σ) pointing errors and performance requirements for the multiple controller configurations can be seen. Notable from these results are the higher bandwidth configurations namely $398/500\text{ Hz}$ and $765/500\text{ Hz}$ which both are capable of complying with **PATR-01**, while the pointing angle error requirement can be met with the base configuration of $398/10\text{ Hz}$, it is the pointing angle-rate the one which appears from the results presented, the harder requirement to comply with.

By comparing the results with an increase in deputy controller bandwidth it can be seen that ultimately the pointing angle-rate requirement can be met. However, this results in an increase in the torque and power required by the deputy controller. At the highest two performance configurations, it can be seen that the deputy controller requires $\approx 12\text{ N} \cdot \text{m}$ and has a power consumption of $\approx 94\text{ W}$, which is an increase from the base configuration of $\approx 66.2\%$ and $\approx 412.8\%$, both still falling under the specifications of the deputy actuator, thus possible to argue that both configurations appear to successfully comply with **PATR-01** and by incorporating both precise and coarse actuators also comply with **PATR-01-01** and **PATR-01-02**. Table 5.3 aims to provide a clear overview of the compliance against the requirements of the multiple systems.

Another relevant remark from the acquired results is the fact that increasing the chief controller bandwidth allows for a significant performance improvement, for all deputy controller bandwidths. From the results presented, it can be noted an $\approx 30\%$ improvement for the pointing angle error and a $\approx 40\%$ improvement over the pointing angle-rate error.

Table 5.2: DSAS DEB analysis performance results for multiple controller bandwidths

Total RMS (3σ)	DSAS (398/10)	DSAS (398/100)	DSAS (398/300)	DSAS (398/500)	DSAS (765/10)	DSAS (765/100)	DSAS (765/300)	DSAS (765/500)
Pointing angle error [μrads]	21.76	14.67	7.06	4.25	15.69	8.82	4.41	2.83
Pointing angle-rate error [$\mu\text{rads/sec}$]	308.00	207.01	90.01	47.22	176.87	116.87	50.84	26.85
Chief Controller Torque [$\mu\text{N} * \text{m}$]	107.76	67.22	30.36	16.23	99.60	61.40	27.64	14.75
Deputy Controller Torque [$\text{N} * \text{m}$]	7.222	10.996	12.179	12.067	7.222	10.999	12.182	12.069
Chief controller Power [mW]	1.47	0.994	0.432	0.225	1.353	0.91	0.39	0.20
Deputy controller Power [W]	18.336	71.12	95.05	93.972	18.336	71.17	95.10	94.00
Chief Actuator effort [mdegrees]	28.49	7.015	2.435	1.229	27.98	6.655	2.260	1.35
Deputy Actuator effort [degrees]	9.066	9.061	9.061	9.061	9.066	9.061	9.061	9.061
Chief Actuator-rate effort [mdegrees/sec]	154.43	84.40	34.77	17.90	145.96	77.99	31.93	16.42
Deputy Actuator-rate effort [degrees/sec]	6.696	6.694	6.695	6.695	6.696	6.694	6.694	6.695

Table 5.3: Verification against critical requirements

Requirement ID	PPAS (398)	CPAS (500)	DSAS (398/500)	DSAS (765/500)
PATR-01 [50 μrads – $\mu\text{rads/sec}$]	Compliant (37.62 / 28.21)	Non-compliant (20.50 / 280.88)	Compliant (4.25 / 47.22)	Compliant (2.83 / 26.85)
PATR-01-01	Non-compliant (Actuator range beyond $\pm 3 \text{ degrees}$)	Non-compliant (No precise actuator)	Compliant (Controller and actuator are under specifications)	Compliant (Controller and actuator are under specifications)
PATR-01-02	Non-compliant (No coarse actuator)	Compliant (Controller and actuator are under specifications)	Compliant (Controller and actuator are under specifications)	Compliant (Controller and actuator are under specifications)

6

Verification of results

This chapter aims to provide the reader with information regarding the steps taken to verify the results obtained from the performed research. The objective is to provide information about different alternative methods that can be utilized to compare the obtained research results.

To achieve this the chapter is divided into three sections, the first section centers on verifying the frequency processing pipeline, which has been discussed in chapter 3 and Appendix A. The next section centers on comparing the obtained Dual-stage closed loop and sensitivity transfer functions discussed in chapter 4, to representative transfer functions which can be found in currently available research. The chapter concludes with the comparison of Dynamic Error Budgeting frequency response analysis discussed in chapter 5, to system response from alternative time-analysis methods.

6.1. Frequency processing pipeline

This section aims to provide the reader with information regarding the procedure utilized to determine the frequency processing pipeline's reliability and verify the consistency of the analysis results of the ship-wave motions dataset. **Note:** this section focuses on the discussion of the accuracy of the frequency processing pipeline, for further information about the specific procedures utilized in the frequency processing pipeline readers are referred to chapter 3 and Appendix A.

The procedure to determine the accuracy of the frequency processing pipeline starts by defining the characteristics of a known signal. This contrasts with the data set utilized in previous chapters, where the ship-wave interactions arise from stochastic processes, thus making it challenging to determine the specific characteristics of those signals. For the procedure in this section, Equation 6.1 provides an analytical description of the test signal which is a white (gaussian) noise signal with a zero mean and a definable standard deviation.

$$W_{in}(t) = \mu + \sigma * \xi(t) \quad (6.1)$$

Where μ represents the mean, which in this case is equal to zero, σ is the standard deviation, for this test is chosen as unit-less unit value and ξ represents a stochastic process of a set of independent identically distributed variables which are obtained from a normal standard distribution, sized accordingly to the simulation duration.

Now with the test signal discussed, next is to discuss the logic of the test procedure. The fundamental idea is to apply all the processing steps that have been utilized as a part of the frequency analysis, including (hamming) windowing, zero padding, Discrete Fast Fourier Transform, conversion of the two-sided spectrum to a single-sided spectrum, power spectral density computation and normalization of amplitude. These steps are applied to the test signal with the help of two custom-made functions 'pre-process_for_dft' and 'normalize_amplitude_dft_and_psd_creation'. The script where the functions are implemented together with the rest of the testing procedure can be found in Appendix I.

Next is to discuss the procedure utilized to estimate the original standard deviation of the signal, since after applying the frequency processing pipeline one can end up with the Power Spectral Density (PSD) function of the signal, it is then possible to utilize the relationship between the PSD and the variance of the signal. The relationship between the two is related to the total power of the signal since the original test signal has a zero mean, the variance of the signal quantifies as its total power. Thus by integrating the PSD function over all the relevant frequencies (since the signal is discrete in this test case), one can determine the total power of the signal, which in this case equates to the variance of the signal, therefore by taking the square root of the variance one can estimate the standard deviation, this is described by Equation 6.2. For readers who are interested in the derivations related to this analytical outcome and are looking for further information about signal processing, are referred to [57].

$$\sigma_{Estimate} = \sqrt{\int_0^{f_s/2} S(f) df} \quad (6.2)$$

Where $S(f)$ represents the output PSD obtained via the frequency processing pipeline, note that since it is being worked with a single-sided power spectral density function the limits of the integral range from the zero frequency to the Nyquist frequency (half of the sampling frequency f_s). For this simulation, the sampling frequency is left at 200 Hz, just as in the original data set utilized in this work. With the method to estimate the variance of the test signal discussed next is to discuss the results. In Appendix I, the full-size plots of the time signal before processing, the resulting power spectral density plot of the signal after the processing steps and the time signal plot after the window and zero padding can be found.

In Table 6.1, the summary of results achieved via the verification script reference earlier can be seen. Note that all the simulations have been performed for a unit standard deviation, as it is of interest to determine the effects of simulation time rather than changes in standard deviation values. It is relevant to note that the presented results are the resulting average estimated resulting from running 10 simulations with equal time, margins ($\pm$) are provided as additional information regarding the difference between obtained results.

Table 6.1: Frequency processing pipeline test signal variance estimate results

Duration of simulation	Estimated variance	Difference to actual [%]
1 second	0.947 $\pm$ 0, 12	-5.30
10 seconds	0.975 $\pm$ 0, 07	-2.50
50 seconds	0.986 $\pm$ 0, 05	-1.40
100 seconds	0.994 $\pm$ 0, 05	-0.60
200 seconds	0.998 $\pm$ 0, 05	-0.20
Total average		-2.00

From the results visible in Table 6.1, it is possible to argue that the frequency processing pipeline could estimate the original standard deviation of a signal within a given margin of error. It could also be argued that increasing the simulation time could help decrease the margin of error, as the simulation results for longer simulation times appear to approximate the original standard deviation value better. Via this verification check it is possible to argue that the frequency processing pipeline can be effective at determining the standard deviation of known test signals, however in the scope of work of this research, the frequency processing pipeline has been utilized for processing stochastic signals arising from ship-wave interactions, whose properties are unknown.

Therefore the results provided in this research are specific to the data set provided, which corresponds to a specific ship under a specific environment with continuous signal durations of 180 *seconds*. To extend these results to different ships, operating in different environments, requires performing additional measurements, thus being considered outside the scope of work of this research. To further verify the validity of the presented results of this research, it would be required to extend the continuous signal duration, which requires performing additional measurements, which in turn could create the opportunity to perform the measurements at a higher sampling rate which could allow to recreate higher-frequency stochastic phenomena potentially occurring onboard a ship.

6.2. Dual-stage closed loop and sensitivity transfer functions

The next section of this chapter centers on the verification of the transfer functions which have been utilized to evaluate the performance of the Dual-Stage Actuator System (DSAS), as to verify the correct control logic design of such transfer functions. To achieve this the closed loop and sensitivity transfer functions obtained in chapter 5 at the DSAS DEB equations section, are compared to transfer functions in research papers where dual-stage controller design has been utilized.

Starting with the closed loop dual-stage transfer function which can be expressed as in Equation 6.3, which in this case is equivalent to the sensitivity function specific to the tracking sensor error, as the position where the disturbances are entering the dual-stage control logic is in the feedback path, as it has passed through the entire system at this point is then possible to equate it to the closed loop transfer function.

$$H_{closed\ loop} = H_{dsas-ts} = \frac{[P_{fsm} + P_{gimbal} + P_{fsm} * P_{gimbal}] * P_{QPD}}{1 + [P_{fsm} + P_{gimbal} + P_{fsm} * P_{gimbal}] * P_{QPD}} \quad (6.3)$$

Now with the closed loop transfer function of the dual-stage controller described, it is possible to compare it to the closed loop transfer function by [6], in which research over a dual-stage piezoelectric-actuated micropositioner is performed, the closed loop transfer function proposed by this research paper is described in Equation 6.4. The control logic diagram utilized in this research is visible in Figure I.11.

$$H_{cl} = \frac{y}{y_r} = \frac{G_1 + G_2 + G_1 * G_2}{(1 + G_1)(1 + G_2)} = \frac{G_1 + G_2 + G_1 * G_2}{1 + G_2 + G_1 + G_2 * G_1} \quad (6.4)$$

Where $G_1 = P_1 * C_1$ and $G_2 = P_2 * C_2$ each represent the coarse and fine pointing stages, respectively. Moreover, the assumption that $\hat{P}_2 = P_2$, refers to the same assumption utilized in this research which mentions that the control system is capable of estimating the fine pointing displacement position, as to equate the actuator estimate to the actual actuator displacement. By comparing the two transfer functions presented so far one could argue that regarding the main contents of the transfer functions, each is equivalent to the other, therefore, it is possible to argue that the closed-loop transfer function is verified against other closed-loop transfer functions from equivalent research. Note that the delay transfer function (P_{qpd}) is missing in the second closed-loop transfer function as time delays are not accounted for in most research articles, however, removing this term from the former closed-loop transfer function results in an equivalent expression.

Next is the verification of the dual-stage sensitivity transfer functions, Equation 6.5 provides the expression of the dual-stage sensitivity transfer function utilized in this work.

$$H_{dsas} = \frac{1}{1 + [P_{fsm} + P_{gimbal} + P_{fsm} * P_{gimbal}] * P_{QPD}} \quad (6.5)$$

Next is to compare the utilized sensitivity transfer function to other research articles, in this case, [32] provides a comparable sensitivity transfer function, as the research performed in that article focuses on investigating the closed-loop sensitivity response of a dual-stage servo system for compute disk drives. The sensitivity function proposed in this article is described by Equation 6.6, while Figure I.12 provides a figure of the sensitivity block diagram.

$$S_T = \frac{1}{1 + K_v * G_v} * \frac{1}{1 + K_m * G_r} = \frac{1}{(1 + P_v)(1 + P_m)} = \frac{1}{1 + P_m + P_v + P_m * P_v} \quad (6.6)$$

Where $P_v = K_v * G_v$ and $P_m = +K_m * G_r$. Now with both expressions fully expanded for ease of comparison, it can be argued that by omitting the time delay in the first sensitivity function, one can argue that they are equivalent to each other, thus verifying the validity of the sensitivity function.

6.3. Dynamic Error Budgeting frequency response

This chapter concludes with a discussion of the verification process utilized to compare the analysis results from the Dynamic Error Budgeting approach. To perform the verification, it was chosen to perform a time-based simulation in Simulink, in which the time signals of the disturbances are fed into the DSAS, and its response to such disturbances is recorded into arrays, which allow for further processing in a regular Matlab script.

The Simulink model of the DSAS angle rate can be seen in Figure I.13, the blocks to the left side correspond to the input time signals, for which windowing and zero padding have been applied. The blocks in the center correspond to the specific sensitivity function and the blocks on the right side perform measurements at a consistent sample rate of 5 ms or 200 Hz to match the original signal sample rate, since the simulink auto solver can perform the simulation at smaller time-steps. **Note** : the application of a hamming window function and zero padding to the input disturbance signals has been based on the choice of attempting to recreate similar conditions to the ones occurring at the frequency analysis. Moreover, the hamming window function could be stabilizing the time-based results at the initial sections of the simulation, as when the time-based signal was fed directly into the Simulink model, it was seen to result in large amplitude spikes for the initial seconds of simulation, which could affect the analysis results.

With the output of the DSAS via the Simulink model, next is the interpretation of output arrays, this has been performed via two different methods, the first method directly determines the standard deviation in the output array, while the second method applies the rest of the frequency processing pipeline to obtain the PSD of the output signals, then the PSD function gets integrated across the relevant frequencies, to obtain the variance (which its square root value is equivalent to the standard deviation). The results from performing both analyses can be seen in Table 6.2. Figures of the output time simulation analysis, the output power spectral densities and the code utilized to process the output results can be found in Appendix I.

Table 6.2: Results summary of the performance results of the time analysis and DEB analysis

Disturbance	DEB $[\mu\text{rad-}\mu\text{rad/sec}]$	Time STD. $[\mu\text{rad-}\mu\text{rad/sec}]$	Diff. To DEB [%]	Time frequency $[\mu\text{rad-}\mu\text{rad/sec}]$	Diff. To DEB [%]
Pitch angle	2.98	3.06	2.68	3.05	2.34
Roll angle	0.18	0.20	11.11	0.19	5.55
Yaw angle	3.70	3.80	2.70	3.78	2.16
QPD angle	0.38	0.36	-5.26	0.39	2.63
Total angle	7.25	7.35	2.38	7.41	2.20
Pitch angle rate	39.25	38.46	-2.01	39.50	0.63
Roll angle rate	21.97	22.06	0.40	22.31	1.54
Yaw angle rate	41.05	43.50	5.96	41.16	0.26
QPD angle rate	0.38	0.37	-2.60	0.36	-5.26
Total angle rate	102.66	104.41	1.70	103.35	0.66
Total average			2.74		1.27

As visible from Table 6.2, the results obtained from the time analysis remain close to the original results from the complete frequency analysis approach, while the standard deviation results have a slightly higher total average when compared to the results acquired from part-time simulation and part frequency analysis, it could be argued that the latter could be closer to the results acquired from the DEB approach as these undergo similarly frequency processing steps, which could potentially affect the averaged results. Moreover, one aspect that has been seen throughout the time analysis plots (visible in Appendix I) is the fact that the time-based simulation appears to have difficulties with representing the response of the system at the lowest frequencies, this could perhaps be caused by the time analysis requiring longer simulation times, when compared to the frequency response of the DEB analysis.

This can be motivated based on the type of response that is expected of the DSAS, as in the Bode plots of the closed loop and sensitivity transfer functions (seen in Figure 4.17), the system shows an increasing capacity of dampening disturbances occurring at frequencies lower than 0.1 Hz , which is also seen in the DEB frequency response plots (Figure G.21). However, the contribution of the system response of the lowest frequencies appears to be low enough such that the results from integrating all the system responses remain low enough such that the total average remains within the $3 - 2 \%$ mark when compared to the DEB results, therefore it is possible to argue that the obtained results are comparable to each other, which allows for the argumentation of the verification of the DEB frequency response results.

7

Conclusion

This work has aimed to answer the main research question regarding whether a dual-stage optical communications terminal could be capable of achieving sufficient pointing precision, which allows for performing Free Space Optical Communications (FSOC) in naval environments.

To achieve this a simulation strategy consisting of 3 main steps has been employed. The first step involved the frequency analysis of the ship-wave motions, as a method to obtain the disturbance spectra representing the naval environment. The second step involved the creation of simulation models which started by representing each of the actuator stages individually (namely the Precise Pointing Actuator Stage PPAS and the Coarse Pointing Actuator Stage CPAS), as to later combine them into the Dual-Stage Actuator System (DSAS). The last step involved the disturbance spectra and the 3 simulation models, these are utilized as inputs for the Dynamic Error Budgeting (DEB) analysis, which approximates the achievable performance of the simulation models.

Before discussing the results of the DSAS, it is relevant to motivate the need for a dual-stage setup for performing FSOC in naval environments. To this end, the DEB performance results of the PPAS and CPAS can be utilized. Critical in determining the suitability of an optical communications terminal for FSOC is its pointing angle and pointing angle-rate errors, for this work **PATR-01** provides the requirement for the setup to reach a pointing angle and pointing angle-rate errors below $50 \mu\text{rads} - \frac{\mu\text{rads}}{\text{sec}}$. While the pointing errors of the PPAS meet the pointing error requirements (total RMS (3σ) of $37.62 \mu\text{rads}$ and $28.21 \frac{\mu\text{rads}}{\text{sec}}$, respectively), the required actuator angular range is at 9.06 degrees , which lies beyond the performance specifications of the precise actuator (namely $\pm 3 \text{ degrees}$ [38]).

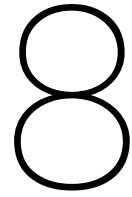
Next are the performance results of the CPAS, which is capable of meeting all actuator and controller performance requirements, however incapable of meeting the pointing error requirements (with a total RMS (3σ) of $493.2 \mu\text{rads}$ and $2531.80 \frac{\mu\text{rads}}{\text{sec}}$, respectively). From both of these results, it is possible to argue that individual actuator stages are not suitable for FSOC in naval environments.

Next are the DSAS results, which are comprised of multiple controller bandwidth configurations, which aim to identify the controller configuration which is capable of complying with the pointing error requirements. Notable from these results are the highest performance configurations with chief and deputy controller configurations of $398/500 \text{ Hz}$ and $765/500 \text{ Hz}$. Both of these are capable of complying with the pointing requirements each remaining $\approx 5.5\%$ and $\approx 46.3\%$ below the pointing angle rate, while both achieving to remain $> 90\%$ under the pointing angle requirement. Furthermore looking into the performance requirements for each actuator it is notable that what is demanded for each stage is well within their respective specification limits, for example, the angular range for the chief (precise) stage is only required $1.2 - 1.4 \text{ mdegrees}$ of range.

It is of great relevance to support the performance claims under this work, to achieve this 3 verification steps have been performed. The first step involved the demonstration of the correct functioning of the frequency analysis, this has been obtained by passing a test signal with known properties through

the entire frequency analysis pipeline. It was then possible to determine that the estimated properties differ from their original properties by only an average of 2%. The second step involved verifying the control design of the DSAS, to achieve this the transfer functions developed under this work, were compared against equivalent transfer functions from research papers, which were found to be equivalent to each other. The last step involved verifying the performance results obtained from the DEB design approach, to achieve this an alternative time analysis simulation has been utilized to reproduce the performance results. The difference between the results obtained from the alternative simulation methods and DEB analysis remains at a margin of 1.3 – 2.8 %.

Based on the research outcomes of this work, it is then possible to argue that a Dual-Stage optical communications terminal with a chief-deputy controller configuration of 398/500 Hz or 765/500 Hz can be capable of performing FSOC in naval environments by fulfilling **PATR-01** pointing requirements and **PATR-01-01**, **PATR-01-02** actuator specification sub-requirements, with a preference for the higher-bandwidth configuration. Moreover given the significant reduction in the required angular range of the chief actuator, if further performance improvements over the pointing errors are desired, changing the OIM5001 for an actuator with a smaller angular range but higher control bandwidth, could potentially further increase the pointing precision of the DSAS setup.



Recommendations for future work

This chapter aims to provide the reader with ideas/recommendations regarding actionable items for future work, these are included in the following points. Note that the placement order (ranging from top to bottom), aims to build upon the previous recommendations, as to further mature the project readiness level.

- **Performing direct system identification of the actuator models:** The analysis performed in this work has relied on the research performed by other research to re-create the dynamics for the FSM actuator, while a rough estimation has been performed regarding the gimbal actuator. This could over or under estimate controller torque and power requirements. By determining the system and its characteristics more thoroughly it would be possible to improve such estimates, to gain a better understanding of the required actuator performances.
- **Expansion of the frequency range of the IMS measurements:** The current data set utilized for this work, has been sampled at 200 Hz , however interesting phenomena such as sudden frequency spikes have been observed close to the Nyquist frequency (100 Hz). As these phenomena start playing a significant role in the pointing error contribution, it would be recommended to investigate the accumulation of pointing errors with data sets with much higher sampling rates.
- **Introduction of external (passive/active) dampening into the performance analysis:** As the build-up in pointing errors for most of the analysis picks up close to the highest frequencies, it would be recommendable to investigate the effects of a passive/active vibration isolation devices, such as a spring damper, as it is likely that they could help in decreasing the transmitted vibrations, thus capable of improving the achievable pointing error.
- **Performing real-world testing of the dual-stage controller setup:** The DEB analysis is typically utilized during early design stages to provide early feedback with respect to design elements that require change, however, it would be advisable to eventually test a prototype in realistic conditions, as to determine not only determine the precision of the DEB analysis but verify in realistic conditions the actual operability of the system.

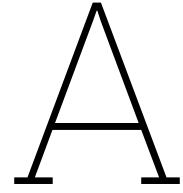
References

- [1] Space Development Agency. *Optical Communications Terminal (OCT) Standard*. Version 3.1.0. <https://www.sda.mil/home/work-with-us/resources/>: United States Space Force, 2023.
- [2] Constantine A. Balanis. *ANTENNA THEORY, ANALYSIS AND DESIGN*. JOHN WILEY & SONS, INC, 2005.
- [3] Jamie W. Burnside et al. "Design of an inertially stabilized telescope for the LLCD". In: *Proceedings of SPIE 7923*. 2011.
- [4] Nanhu Chen et al. "Modeling and control of a fast steering mirror in imaging applications". In: *IEEE Conference on Automation Science and Engineering* (2010).
- [5] Ernst Csencsics and Georg Schitter. "System Design and Control of a Resonant Fast Steering Mirror for Lissajous-Based Scanning". In: *IEEE/ASME Transactions on Mechatronics* 22.5 (2017), pp. 1963–1972.
- [6] Romina Zarrabi Ekbatani et al. "Design and Control of a Flexure-Based Dual Stage Piezoelectric Micropositioner". In: *International Journal of Precision Engineering and Manufacturing* (2024).
- [7] M Elsepee et al. "Design, modeling, manufacturing, and control of dual axis stabilized system". In: *Journal of Physics: Conference Series* 2616 (Nov. 2023), p. 012030.
- [8] M. Frigo and S.G. Johnson. "FFTW: an adaptive software architecture for the FFT". In: *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP '98 (Cat. No.98CH36181)* 3 (1998), pp. 1381–1384.
- [9] Google. *Google earth web*. 2024. URL: <https://earth.google.com/web> (visited on 09/06/2024).
- [10] Robert M. Gray and Lee D. Davisson. *Introduction to Statistical Signal Processing*. Cambridge University Press, 1990.
- [11] Mark T. Gruneisen et al. "Holographic compensation of severe dynamic aberrations in membrane-mirror-based telescope systems". In: *SPIE, International Society for Optics and Photonics* 3760 (1999), pp. 142–152.
- [12] Gary Haggart. *Modeling and Simulation of an Inertially Stabilized Gimbal System for System Performance Evaluation Based on Sensor Selection*. CALIFORNIA STATE UNIVERSITY, NORTHRIDGE, 2015.
- [13] HAMAMATSU. "InGaAs PIN Photodiodes, G6849 series". In: (2024). URL: https://www.hamamatsu.com/content/dam/hamamatsu-photonics/sites/documents/99_SALES_LIBRARY/ssd/g6849_series_kird1042e.pdf (visited on 07/23/2024).
- [14] William Hart et al. "Overview of the spacecraft design for the Psyche mission concept". In: *IEEE Aerospace Conference* (2018), pp. 1–20.
- [15] Ziqian He et al. "Liquid Crystal Beam Steering Devices: Principles, Recent Advances, and Future Developments". In: *Crystals* 9.6 (2019). ISSN: 2073-4352. URL: <https://www.mdpi.com/2073-4352/9/6/292>.
- [16] Frank Heine et al. "In orbit performance of tesat LCTs". In: *SPIE LASE* (2019).
- [17] João P. Hespanha. *Linear Systems Theory: Second Edition*. NED - New edition, 2. Princeton University Press, 2018. ISBN: 9780691179575. (Visited on 06/28/2024).
- [18] International Commission on Illumination. 2024. URL: <https://cie.co.at/> (visited on 01/16/2024).
- [19] Matlab Inc. *Fast Fourier Transform*. 2024. URL: <https://nl.mathworks.com/help/matlab/ref/fft.html> (visited on 02/16/2024).
- [20] Matlab Inc. "PID Controller tuning". In: (2024). URL: https://nl.mathworks.com/help/control/pid-controller-design.html?category=pid-controller-design&s_tid=CRUX_topnav (visited on 07/03/2024).

- [21] Siegfried W. Janson and Richard P. Welle. "The NASA Optical Communication and Sensor Demonstration Program: An Update". In: *Proceedings of the 28th Annual AIAA/USU Conference on Small Satellites*. 2014.
- [22] Andrew H. Jazwinski. "Stochastics Processes and Filtering Theory". In: *Dover Publications* (2007).
- [23] Ir. A. Kamp. "Space Instrumentation Engineering, Lecture Notes ae4-880". In: *Faculty of Aerospace Engineering, TU Delft* (2007).
- [24] Hemani Kaushal, V.K. Jain, and Subrat Kar. *Free Space Optical Communication*. 7th Floor, Vijaya Building, 17 Barakhamba Road, New Delhi 110 001, India: Springer, 2017.
- [25] Yagiz Kaymak et al. "A Survey on Acquisition, Tracking, and Pointing Mechanisms for Mobile Free-Space Optical Communications". In: *IEEE Communications Surveys and Tutorials* 20.2 (2018), pp. 1104–1123.
- [26] Chenghu Ke et al. "Design and Implementation of a Non-Common-View Axis Alignment System for Airborne Laser Communication". In: *Photonics* 10.9 (2023).
- [27] Mahmudur Khan and Murat Yuksel. "Maintaining a free-space-optical communication link between two autonomous mobiles". In: (2014), pp. 3154–3159.
- [28] Hamed Khodadadi, MRJ Motlagh, and Mohammad Gorji Sefidmazgi. "Robust control and modeling a 2-DOF Inertial Stabilized Platform". In: (June 2011). DOI: 10.1109/INECCE.2011.5953880.
- [29] Li L. et al. "High-Capacity Free-Space Optical Communications Between a Ground Transmitter and a Ground Receiver via a UAV Using Multiplexing of Multiple Orbital-Angular-Momentum Beams." In: *Sci Rep* 7 (2017).
- [30] B. P. Lathi. *Linear Systems and Signals*. 2nd. USA: Oxford University Press, Inc., 2009. ISBN: 0195392566.
- [31] Qing Li et al. "Development of Multitarget Acquisition, Pointing, and Tracking System for Airborne Laser Communication". In: *IEEE Transactions on Industrial Informatics* 15.3 (2019), pp. 1720–1729.
- [32] Yunfeng Li and Roberto Horowitz. "Mechatronics of Electrostatic Microactuators for Computer Disk Drive Dual-Stage Servo Systems". In: *IEEE/ASME TRANSACTIONS ON MECHATRONICS, VOL. 6* (2001).
- [33] Richard G. Lyons. "Understanding Digital Signal Processing, 3rd Edition". In: *Pearson Education, Inc.* (2011).
- [34] A. K. Majumdar. "Advanced Free Space Optics (FSO): A Systems Approach". In: (2014).
- [35] Robert G. Marshalek. *Pointing, Acquisition, and tracking in Near-Earth Laser communications*. Second Edition. CRC Press, 2020.
- [36] Wouter Monkhorst. "Dynamic Error Budgeting, a design approach". In: *TU Delft, Systems and Control* (2004).
- [37] Optics in Motion. - *OIM5001-small one inch FSM*. 2024. URL: <https://opticsinmotion.org/wp-content/uploads/2019/09/OIM5001-Specification-Sheet-12-23-2013.pdf> (visited on 07/01/2024).
- [38] Optics in Motion. *OIM102-two inch FSM*. 2024. URL: <https://opticsinmotion.org/wp-content/uploads/2019/09/OIM102-Specification-sheet-11-25-08.pdf> (visited on 02/13/2024).
- [39] Optics In Motion. *Standard mirrors*. 2024. URL: <https://opticsinmotion.org/standardmirrors/> (visited on 05/27/2024).
- [40] Walter H. Munk. "ORIGIN AND GENERATION OF WAVES". In: *Coastal Engineering Proceedings* 1 (1950). DOI: 10.9753/icce.v1.1. URL: <https://icce-ojs-tamu.tdl.org/icce/article/view/904>.
- [41] Vijay Nafria, Xiao Han, and Ivan B. Djordjevic. "Improving free-space optical communication with adaptive optics for higher order modulation". In: 11509 (2020), 115090K.
- [42] NASA. *Technology Readiness Levels*. 2024. URL: <https://www.nasa.gov/directorates/somd/space-communications-navigation-program/technology-readiness-levels/> (visited on 02/26/2024).

- [43] Ulrik D. Nielsen and Jesper Dietz. "Ocean wave spectrum estimation using measured vessel motions from an in-service container ship". In: *Marine Structures, ELSEVIER* (2020).
- [44] Hidehiko Numasato and Masayoshi Tomizuka. "Setting Control and Performance of Dual-Actuator Systems for Hard Disk Drives". In: *IEEE/ASME Transactions on Mechatronics* (2004).
- [45] Axiom Optics. *Free-Space Optical Communications (FSO)*. 2024. URL: <https://www.axiomoptics.com/application/free-space-optical-communications/> (visited on 08/09/2024).
- [46] Mert Özçelik, Burak Kürkçü, and Zeki Yağız Bayraktaroğlu. "Modelling and Simulation of 2 DOF Gimbal System with Experimental System Identification". In: *International Conference on Systems and Control (ICSC)* 10th (2022), pp. 508–515.
- [47] Tristan Pérez and Mogens Blanke. "Simulation of Ship Motion in Seaway". In: *Department of Electrical Engineering, Technical University of Denmark* (2002).
- [48] Inc. PlaneWave instruments. "L-350 Direct Drive telescope mount". In: (2024). URL: <https://planewave.com/products/l-350-telescope-mount/> (visited on 07/08/2024).
- [49] J. Pouch et al. "Liquid crystal-based beam steering technologies for NASA applications". In: *Great Lakes Photonics Symposium, Antenna, Microwave and Optical Systems Branch* (2006).
- [50] Abdelfatah R., Alshaer N., and Ismail T. "A review on pointing, acquisition, and tracking approaches in UAV-based fso communication systems". In: *Opt Quant Electron* 54 (2022).
- [51] W. S. Rabinovich et al. "Free-space optical communications research and demonstrations at the U.S. Naval Research Laboratory". In: *Appl. Opt.* 54.31 (Oct. 2015), F189–F200.
- [52] Danielle Racelis and Mathieu Joerger. "High-Integrity TLE Error Models for MEO and GEO Satellites". In: *AIAA SPACE and Astronautics Forum and Exposition* (2018).
- [53] Distrubution limited but can be provided upon request. *Feasibility study for naval communications*. Breda, The Netherlands: Dutch Ministry Of Defence (Ministerie van Defensie), 2021.
- [54] Dr.ir. Rudolf Saathof. "Laser Satellite Communications (Lecture 2)". In: *AE4880 Space instrumentation course at the Technical Univerisity of Delft* (2023).
- [55] Karen Saucke et al. "The Tesat transportable adaptive optical ground station". In: *SPIE LASE* (2016).
- [56] Tadahiko Shinshi, Kazuhide KODEKI, and Kazuhiko FUKUSHIMA. "Fast Steering Mirrors for Observation Satellites". In: *Journal of the Japan Society for Precision Engineering* 86 (Dec. 2020), pp. 930–933.
- [57] Steven W. Smith. "The Scientist and Engineer's guide to Digital Signal Processing, 2nd edition". In: *California Technical Publishing* (1999).
- [58] The Consultative Committee for Space Data systems. *Optical High Data Rate (HDR) communication-1064 NM*. ORANGE BOOK, 2018.
- [59] Weyrauch T. and Vorontsov M. "Free-space laser communications with adaptive optics: Atmospheric compensation experiments". In: *Optic Comm Rep* 1 (2004).
- [60] Gil Talmor, Harvard Harding, and Chien-Chung Chen. "Two-axis gimbal for air-to-air and air-to-ground laser communications". In: (Mar. 2016), 97390G.
- [61] Charles Thompson et al. "Free Space Optical Communications Utilizing MEMS Adaptive Optics Correction". In: *Proceedings of SPIE - The International Society for Optical Engineering* 4821 (Dec. 2002).
- [62] Inc. Thorlabs. *Fast steering mirror*. 2024. URL: https://www.thorlabs.com/newgrouppage9.cfm?objectgroup_id=15347 (visited on 02/13/2024).
- [63] Inc. Thorlabs. *TMEMS-Based Deformable Mirrors*. 2024. URL: https://www.thorlabs.com/newgrouppage9.cfm?objectgroup_id=3258&gad_source=1 (visited on 02/13/2024).
- [64] Morio Toyoshima et al. "Optimum divergence angle of a Gaussian beam wave in the presence of random jitter in free-space laser communication systems". In: *J. Opt. Soc. Am. A* 19.3 (2002), pp. 567–571.

- [65] Xinghua Wang et al. "Liquid Crystal on Silicon (LCOS) Wavefront Corrector and Beam Steerer". In: *Proceedings of SPIE - The International Society for Optical Engineering* 5162 (Dec. 2003).
- [66] Heinz A. Willebrand and Baksheesh S. Ghuman. "Free Space Optics: Enabling Optical Connectivity in Today's Networks". In: *Sams, Imprint of Simon and Schuster* (2001). URL: <https://api.semanticscholar.org/CorpusID:61136110>.
- [67] Xiao-Hui Xu et al. "Design, fabrication and characterization of a bulk-PZT-actuated MEMS deformable mirror". In: *Journal of Micromechanics and Microengineering* 17 (Oct. 2007), p. 2439.
- [68] He Ziqian et al. "Liquid Crystal Beam Steering Devices: Principles, Recent Advances, and Future Developments". In: *Crystals* 9 (June 2019), p. 292.



Ship-wave inconsistencies with the data set

The aim of this appendix is to provide a thorough discussion of the challenges introduced by the sampling rate inconsistencies found in the entire ship-wave motion IMU data. The first section is devoted to discussing in further detail the type of inconsistencies found in the data set. The next sub-section covers an attempt to change the data set via linear interpolation, as a method for resampling the entire data set, as well as its drawbacks, which ultimately led to discarding this method.

The following sub-section covers the approach taken in this work, which involves splitting the data set into smaller data sets, that meet certain sampling requirements. The next section covers the pre-processing steps regarding the application of windowing and zero padding functions, which are employed to prepare the smaller data subsets for being incorporated into the average disturbance analysis in chapter 3.

A.1. Inconsistencies with the original data set

While examining the data set of the ship-wave interactions recorded by the IMS (sampling frequency of 200 Hz) first discussed in chapter 2, several artifacts and inconsistencies were discovered. While a certain amount of artifacts is expected when dealing with real-sampling devices, a relatively large presence of artifacts in the data set could introduce artificial amplitude peaks when looking at the higher frequency spectrum of the DFT's, thus inquiring further into such artifacts was necessary before utilizing the data for further analysis.

The identified artifacts are related to an inconsistent sampling time of the IMS. The reason behind such sampling inconsistencies is unknown and since it affects the time vector, the recordings of the 6 DOF are all affected by it. To provide an idea of what such artifacts look like, see Figure A.1, which provides a zoomed-in overview of the time-varying data series of the Pitch Rate motion.

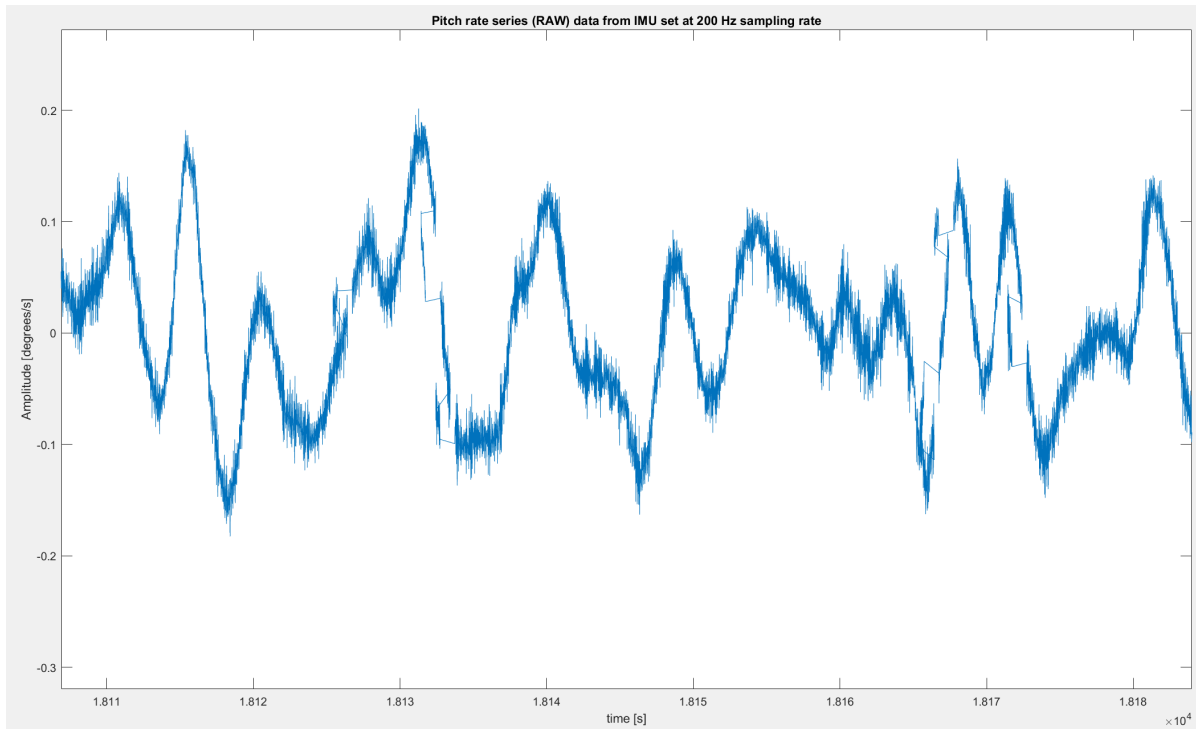


Figure A.1: Artifacts in the sampled time data of the Pitch Rate motion

As visible in Figure A.1, the sampling artifacts introduce sharp changes in the data series, which could potentially introduce high-to-mid frequency peaks. Since it would be difficult to determine the difference between real peaks and those caused/induced by the sampling artifacts, further analysis has been performed. It was critical to determine if the sampling artifacts were limited to only certain sections of the data set, to further investigate upon this, the variance in time step over the data set was investigated. To achieve this the difference between the nominal step size of 5ms (for a sampling rate of 200 Hz) and the current timestep was calculated for each sample and plotted as visible in Figure A.2.

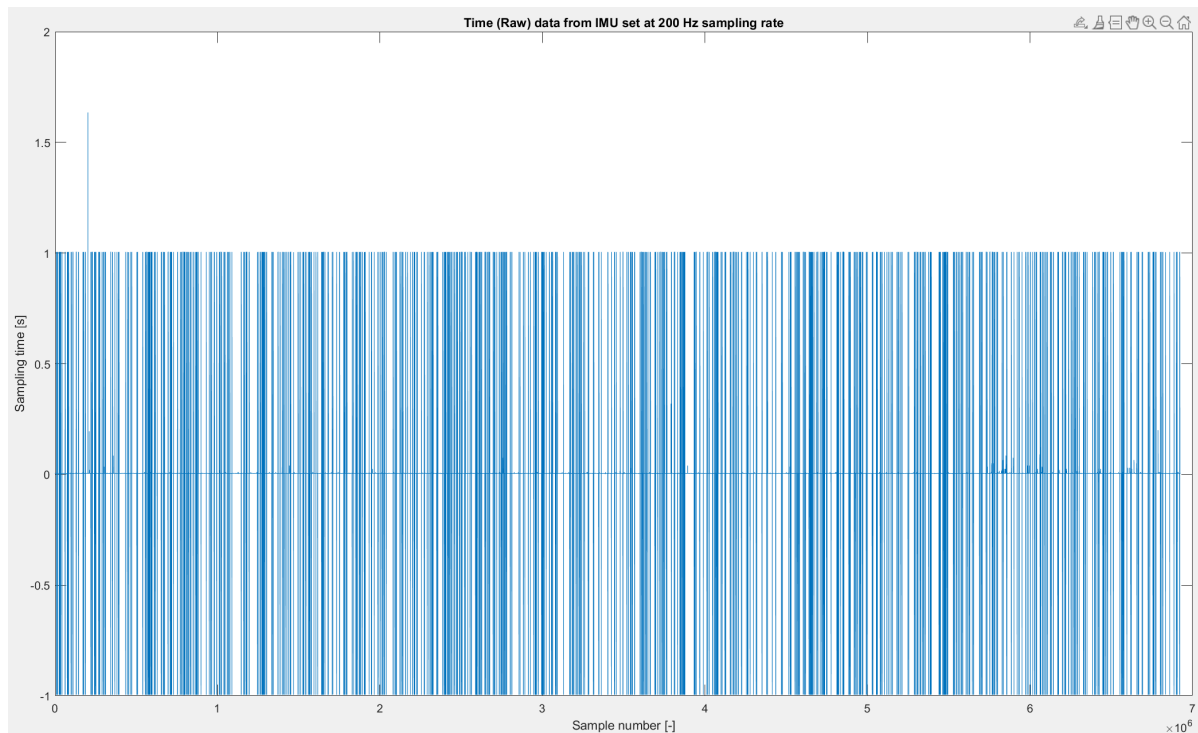


Figure A.2: Variance in time step size over the total sampling number

As visible in Figure A.2, the sampling artifacts are present all across the data set, albeit with certain regions showcasing local regions of constant sampling time. This matches well with the information portrayed in Figure A.1, where it can be observed that while there are multiple sets of data where the sampling times are inconsistent, there are also sections of data that seem to appear as having adequate sampling times.

A.1.1. Linear interpolation as a possible fix to sampling time artifacts

As discussed in the previous paragraph, there is a significant number of sections that showcase constant sampling times. Moreover, since the sections with inconsistent sampling times are brief and overlap between sections with constant sampling times, it was originally thought that by linearly interpolating the mean value at the sections with inconsistent time steps, the set could be re-formatted into a set with constant time steps. Performing the linear interpolation utilizing mean values as fill-in values for data missing in between the inconsistent sampling can be quickly achieved via the "retime" MATLAB function. In Figure A.3, the results of applying this function to the data set are shown.

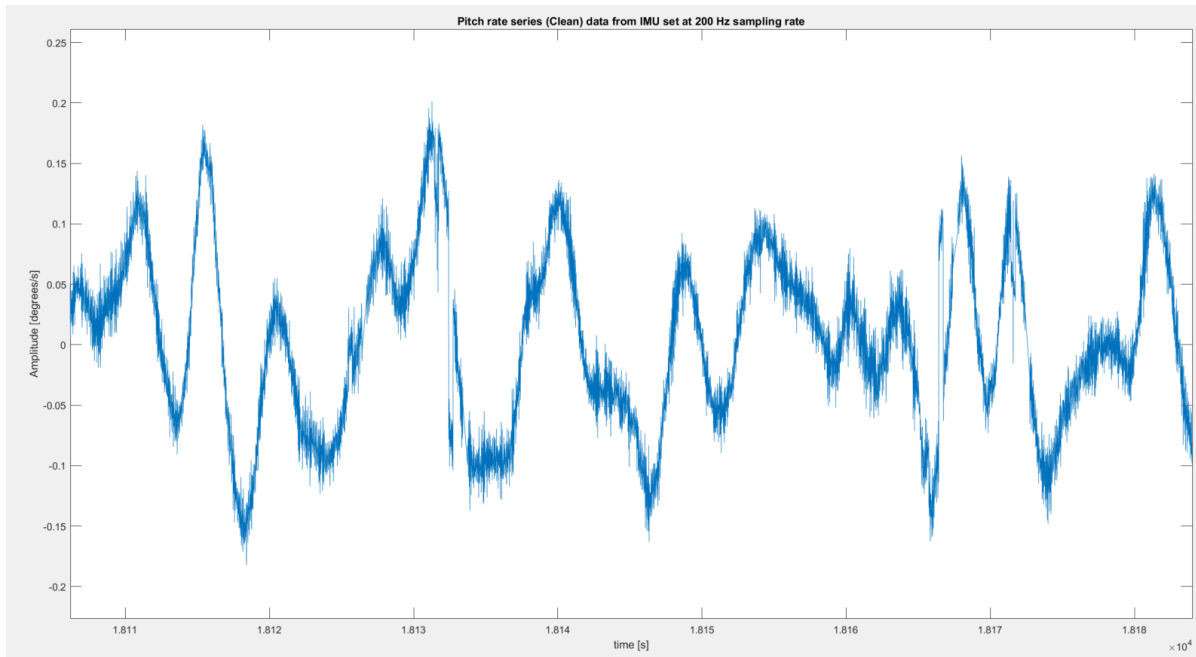


Figure A.3: Constant time-step via linear interpolation of the pitch rate data series

While it is possible to determine from Figure A.3, that a certain amount of artifacts remain in the modified data set, the rest of the data points appear to follow a consistent path. However, to determine with clarity the effects of performing linear interpolation it is best to examine the results in the frequency domain. Applying the DFT algorithm discussed in chapter 3 allows us to convert the data set from the time domain into the frequency domain. Figure A.4 provides an overview of the differences between the original (raw) and the modified (clean) data sets in the frequency domain.

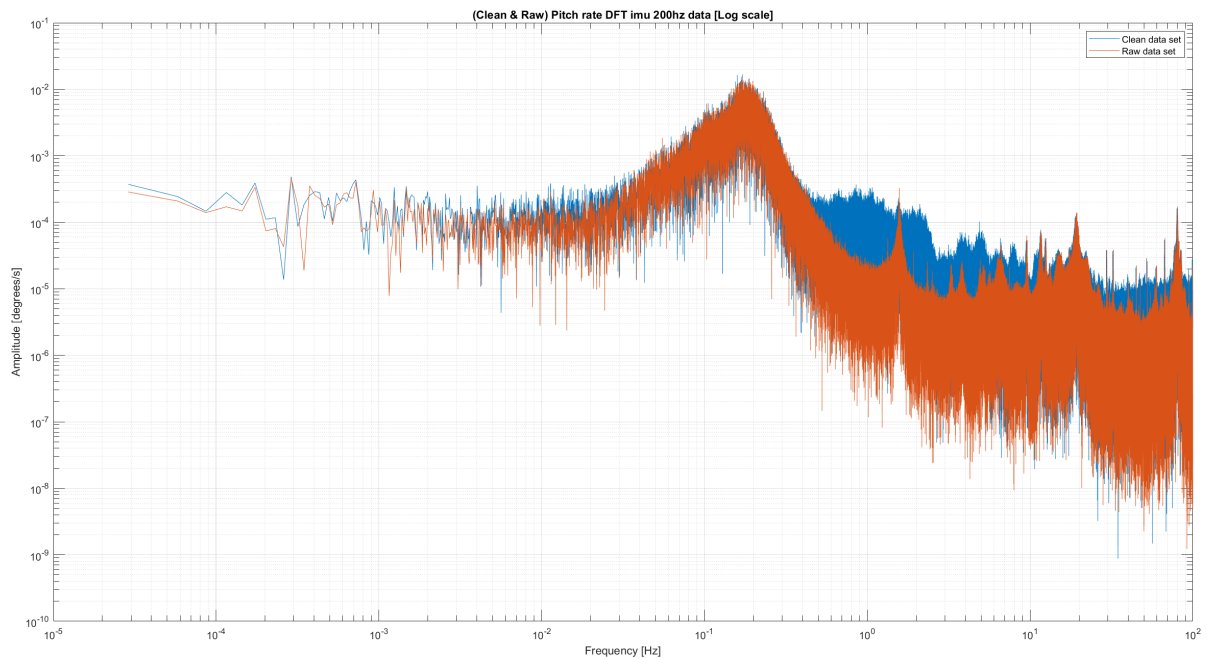


Figure A.4: Comparison between raw (un-modified) and "clean" (linearly interpolated) data sets in the frequency domain, of the pitch rate ship motion

One of the main remarks that can be drawn from Figure A.4 is the apparent dampening of peaks in amplitude found at higher frequencies. It appears that manipulating the data set via linear interpolation

has introduced dampening at higher frequencies. This could be the result of removing the majority of the sharp changes in the data set due to the removal of sampling rate artifacts. By looking into the re-sampled data is possible to notice that the variance in time steps for the modified data set is almost completely removed and the remaining variance is limited to $< X10^{-9}$ seconds.

However the dampening could be affecting naturally occurring phenomena in the data sets, as some of the peaks in amplitude in the frequency plots, could be arising from actual disturbances taking place in the ship. Therefore further analysis of the data set is required to determine if the peaks in amplitude are naturally occurring or artifacts from inconsistent sampling rates.

A.1.2. Narrowing the original total data set into smaller sets

Since the data set contains 9 hours and 37min of recorded data, another approach to determine if peaks in amplitude seen in the raw data set (orange color) Figure A.4 are naturally occurring, is to split the data set into smaller sets. Since we are interested in the behavior at higher frequencies (>0.01 - 0.1 Hz), selecting signals with durations larger than 30 seconds should contain sufficient information to adequately characterize the higher-frequency phenomena. Moreover as visible in Figure A.2, there are sections where the time-steps variations are kept to a minimal amount, thus confirming the existence of such smaller data sets with constant sampling rates.

Since the data set is large and looking for such data sets could be time intensive if not as well prone to human error (missing sections), an algorithm has been developed in MATLAB to search in the data set for smaller regions with constant sampling rates. This algorithm/function can be found in Appendix C. The function ("constant_sampling_intervals") requires as inputs an array with only the difference between the nominal sampling rate (this case being 5ms) and the actual sampling rate. It also requires the minimum duration for which the signals should last with a continuous sampling rate, to be added to the selection list. The step size i.e. nominal sampling rate and lastly the allowed difference between nominal sampling rate and actual sampling rate. The last one is required to be greater than zero, since there is numerical noise involved with these calculations, thus a variance lower than $< X10^{-6}$ seconds is being considered as a "constant" sampling rate, for this work.

Now with the right set of inputs, the function can then output the total number of intervals that fulfill the requirements, the starting and finishing sample numbers of each interval and the duration of each interval in sample number and seconds. To highlight the results from utilizing this function Figure A.5, provides an overview of the terminal output for showcasing results matching the requirements of intervals with a duration longer than 40 seconds and a variance between timesteps less than $X10^{-8}$ seconds.

```
>> identification_of_continuous_sampling
Interval 1: Start index = 51377, End index = 59714, Duration = 8338 samples / 4.169000e+01 secs
Interval 2: Start index = 1181584, End index = 1190550, Duration = 8967 samples / 4.483500e+01 secs
Interval 3: Start index = 1993340, End index = 2001784, Duration = 8445 samples / 4.222500e+01 secs
Interval 4: Start index = 2445771, End index = 2454231, Duration = 8461 samples / 4.230500e+01 secs
```

Figure A.5: Terminal output of the constant_sampling_intervals function

A.2. Creation of data sub-data sets

Using the function described in the previous sub-section it was then possible to identify the longest intervals in the data set which meet certain sampling requirements. Originally a variance in the nominal sampling rate (that is the difference between the actual and the nominal 5ms sampling rates) of $< X10^{-8}$ seconds, was sought after. However the duration of each of those sets were below the ≈ 47 seconds mark, thus phenomena happening close to the 0.01 to 0.1 Hz could be under-represented or entirely missing. As a compromised solution, the requirement in nominal sampling rate was decreased to $< X10^{-5}$ to create larger data sets. This is aimed to capture the slower occurring phenomena, while frequencies higher than 10 Hz are represented by the data sets of shorter duration but with a variance in nominal sampling rate $< X10^{-8}$.

A.3. Windowing and Zero padding for pre-processing of the data subsets

This section aims to provide additional information, regarding the procedures employed for pre-processing of the shortened signals. The aim is to utilize the shortened signals as the basis for creating high and low frequency data sets, which approximate as close as possible to the real (continuous) disturbance signals, by minimizing the introduction of artifacts during signal processing. To achieve this, two commonly applied methods have been employed, these including function windowing and zero padding.

The former procedure involves the multiplication of the shortened signal by a window function, such that the edges of the original signal end up tapered. The aim is to reduce the amount of spectral leakage, which could arise from discontinuities being introduced, by abruptly cutting a section of the original signal. The undesirable effect of spectral leakage is the spreading of energy across frequencies instead of remaining confined to specific frequencies.

The fundamental working principle of the window is to "force the amplitude of the input time sequence at both the beginning and the end of the sample interval to go smoothly toward a single common amplitude value"[33]. There are several window functions available for implementation, however for this research, a Hamming window function has been selected.

The decision is based on the capacity of this function to reduce spectral leakage while maintaining a reasonably narrow main lobe. While it is not capable of achieving the ultra-narrow main lobe of a rectangular window, it is still capable of providing higher frequency resolution when compared to other window functions with lower side lobes. Paired with the fact that such window function is provided with the in-built functionality in MATLAB, as well as not requiring specific tuning such as the Kaiser window, are the main reasons for choosing this window function over others. The Hamming window function can be expressed analytically utilizing:

$$w(n) = 0.54 - 0.46 \cos\left(\frac{2\pi n}{N}\right) \quad (\text{A.1})$$

Valid for $n = 0, 1, 2, \dots, N - 1$.

A.3.1. Compensation for decrease in resulting magnitude as a result of applying a windowing function

Before presenting the results of applying the windowing function and comparing them to the original signal, it is relevant to address the issue of reduction in amplitude as a by-product of the windowing function. Since the signal is being tapered at the beginning and end of the interval of the signal duration, the total energy contained in the signal is affected as well.

To address this, it is possible to create a compensation ratio that is capable of making up for the difference in the windowed amplitude and the original amplitude. To understand how this work is useful to define the total energy of a discrete-time signal $x[n]$ is defined as:

$$E_x = \sum_{n=0}^{N-1} |x[n]|^2 \quad (\text{A.2})$$

When a windowing function $w[n]$ is applied to the signal, the resulting windowed signal $y[n]$ is equal to the following:

$$y[n] = x[n] \cdot w[n] \quad (\text{A.3})$$

Therefore the energy of the windowed signal $y[n]$ can be then described as follows:

$$E_y = \sum_{n=0}^{N-1} |y[n]|^2 = \sum_{n=0}^{N-1} |x[n] \cdot w[n]|^2 = \sum_{n=0}^{N-1} |x[n]|^2 \cdot |w[n]|^2 \quad (\text{A.4})$$

This can be rewritten in a more convenient format as follows:

$$E_y = E_x \cdot \sum_{n=0}^{N-1} |w[n]|^2 \quad (\text{A.5})$$

To compensate for the amplitude loss due to applying the windowing function, it is then necessary to adjust the amplitude of the windowed signal. The aim is that its energy matches the energy of the original signal. The ratio of the energies gives the following scaling factor. Let W denote the sum of the squared window values:

$$W = \sum_{n=0}^{N-1} |w[n]|^2 \quad (\text{A.6})$$

From the energy relation, it is possible to set the following equality:

$$E_y = E_x \cdot W \quad (\text{A.7})$$

To compensate for the amplitude loss, it is possible to scale the windowed signal by a factor that adjusts its energy back to its original value E_x . This scaling factor can be described as follows:

$$\sqrt{\frac{E_x}{E_y}} = \sqrt{\frac{E_x}{E_x \cdot W}} = \sqrt{\frac{1}{W}} \quad (\text{A.8})$$

Note that the ratio is the square root since the original relationship has been derived in terms of energy, which is proportional to the square of amplitude. The developed code to implement this into the Matlab environment which is utilized for the rest of the simulation can be found as "preprocess_for_dft" and "normalize_amplitude_dft_and_psd_creation" in Appendix D. The result before and after applying this function to an exemplary signal with a constant sampling rate can be seen in Figure A.6.

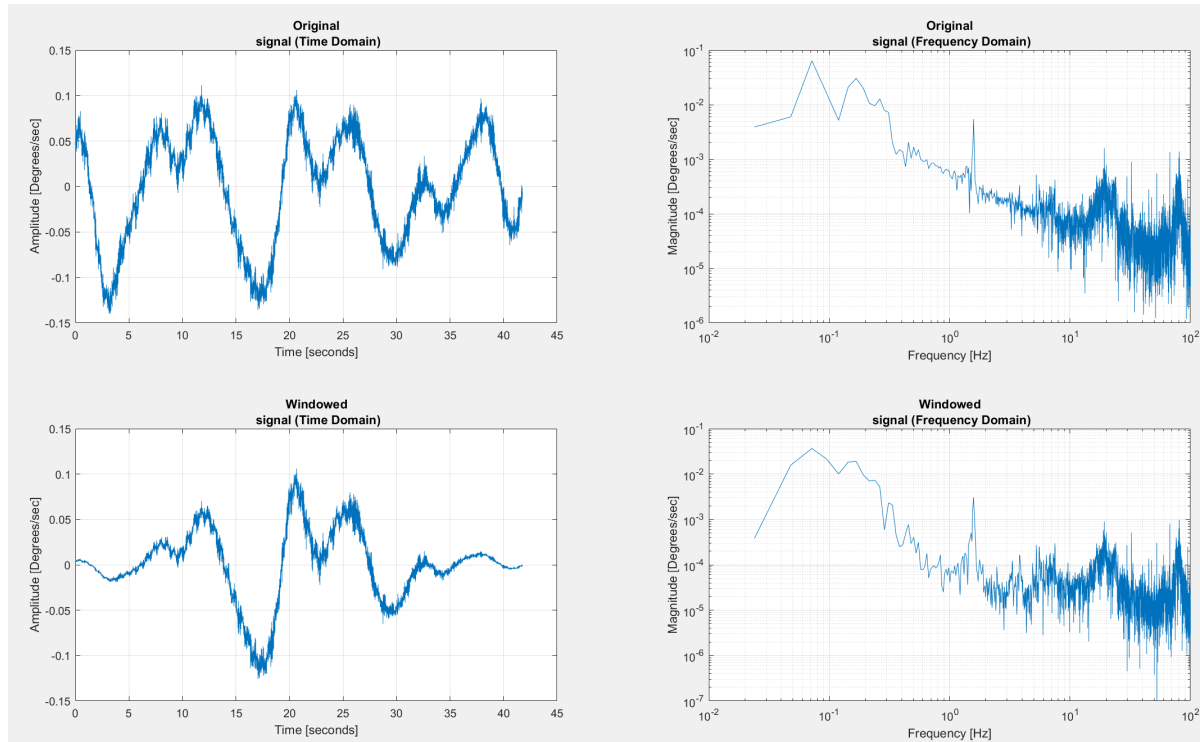


Figure A.6: Signal with an interval duration of ≈ 47.1 seconds, with constant sampling rate, utilizing the pitch-rate ship motion data. The left side plots display the time-varying data before and after the application of the Hamming window functions, while the right side plots show the data in the frequency domain.

As visible in Figure A.6, it is quite noticeable the effects of applying the Hamming window function as seen in the bottom left sub-plot, the abrupt changes at the edges of the signal have been smoothed

out. However, what is of significant relevance are the results provided in the bottom frequency plots, especially for the frequencies higher than 1 Hz, where the peaks in amplitude for the disturbances have smaller tails when comparing between windowed and un-windowed signal, indicating what could be described as a significant decrease in spectral leakage.

Continuing to the latter procedure applied for pre-processing includes zero padding. The fundamental principle of zero padding is the addition of zeros to a signal, to increase its duration, thereby increasing the total number of input data samples.

By increasing the sample number, the DFT is able to not only increase the total number of frequency bins but as well decrease the spacing between frequency bins, which results in a higher frequency resolution. This can be described analytically as follows; if a signal $x[n]$ of length N is considered, when zero padding is applied, the new signal ($x_p[n]$) will then result in a new length M (where $M > N$), the new signal can be described as follows:

- $x_p[n] = x[n]$ for $n < N$,
- $x_p[n] = 0$ for $n \geq M$,

The DFT of the zero padded signal $x_p[n]$ is given by :

$$X_p[k] = \sum_{n=0}^{M-1} x_p[n] \cdot e^{-j \cdot 2\pi \cdot k \cdot n / M}$$

In contrast, the DFT of the original signal $x[n]$ with length N is:

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{-j \cdot 2\pi \cdot k \cdot n / N}$$

Zero padding changes the frequency resolution (spacing between DFT bins). For the original signal with length N :

$$\Delta f = \frac{f_s}{N},$$

Where f_s is the sampling rate. For the zero-padded signal with length M :

$$\Delta f_p = \frac{f_s}{M},$$

This results in a higher frequency resolution because the spacing between the frequency bins of the DFT is smaller. While the increase in frequency allows for improvement in the approximation of the Continuous Fourier Transform (CFT) of the real signal, there is another critical advantage for applying this pre-processing step, that is the fact that intervals of the signal with different durations, can be modified into signals with equal duration.

This allows for the combination and averaging of the frequency information found in such intervals, which further increases the accuracy of the DFT approximation of the original CFT of the real signal. To observe the results of applying zero padding to the previously discussed signal, Figure A.7 provides an overview of the results of applying zero padding. The developed code to implement this into the Matlab environment which is utilized for the rest of the simulation can be found as "preprocess_for_dft" and "normalize_compen_padd_amplitude_dft_and_psd_creation" in Appendix D.

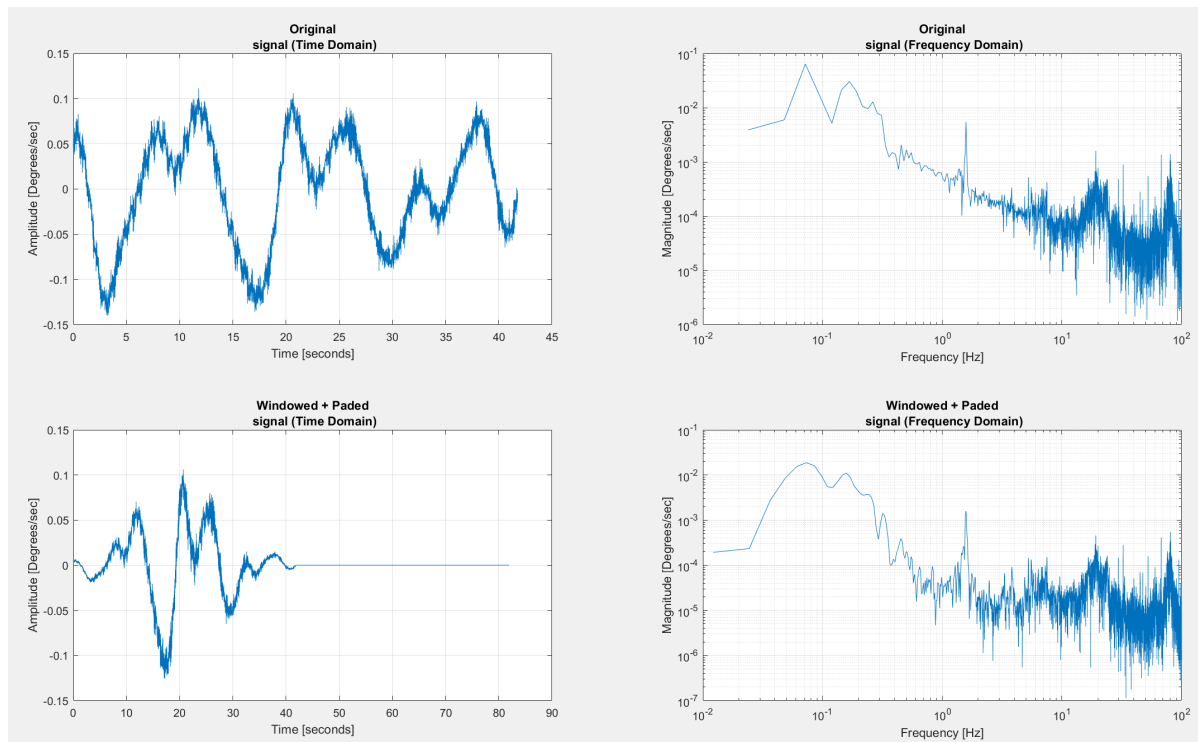


Figure A.7: Signal with an interval duration of ≈ 47.1 seconds, with constant sampling rate, obtained from the pitch-rate ship motion data set. The plots on the left side correspond to time domain data, while plots on the right side correspond to frequency domain data.

B

Average Ship-Wave motion/interactions frequency plots

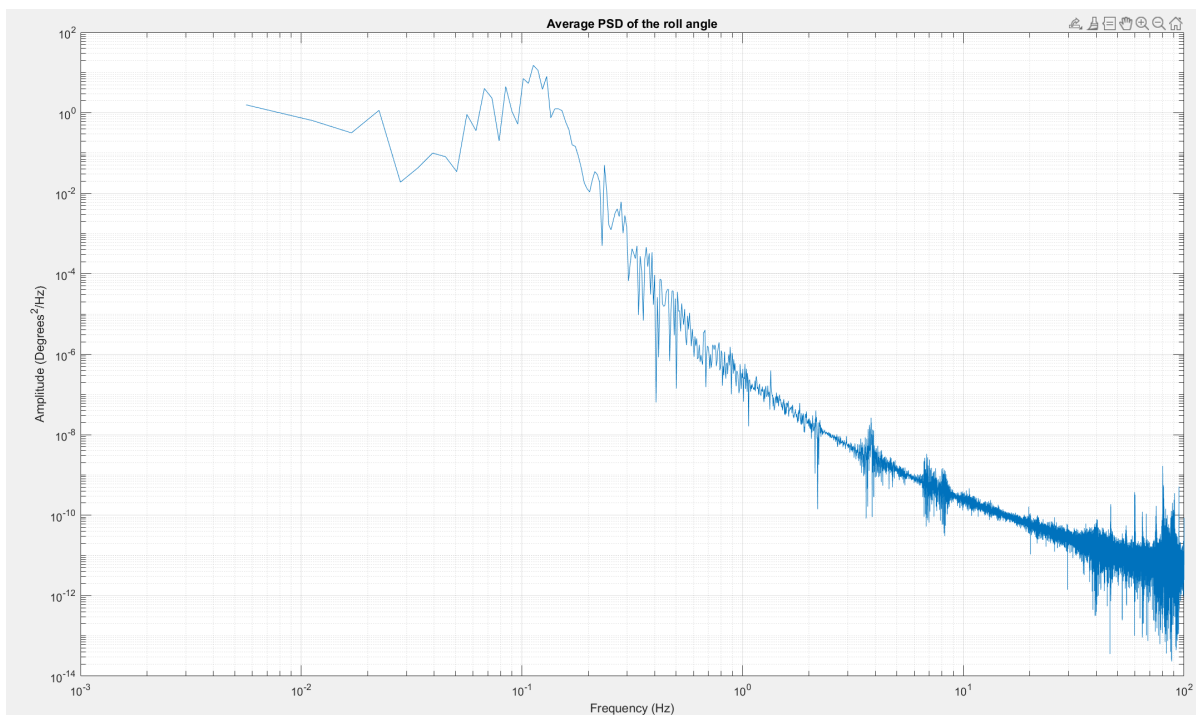


Figure B.1: Roll Ship-Wave motion IMU frequency data set in logarithmic scale for both axes

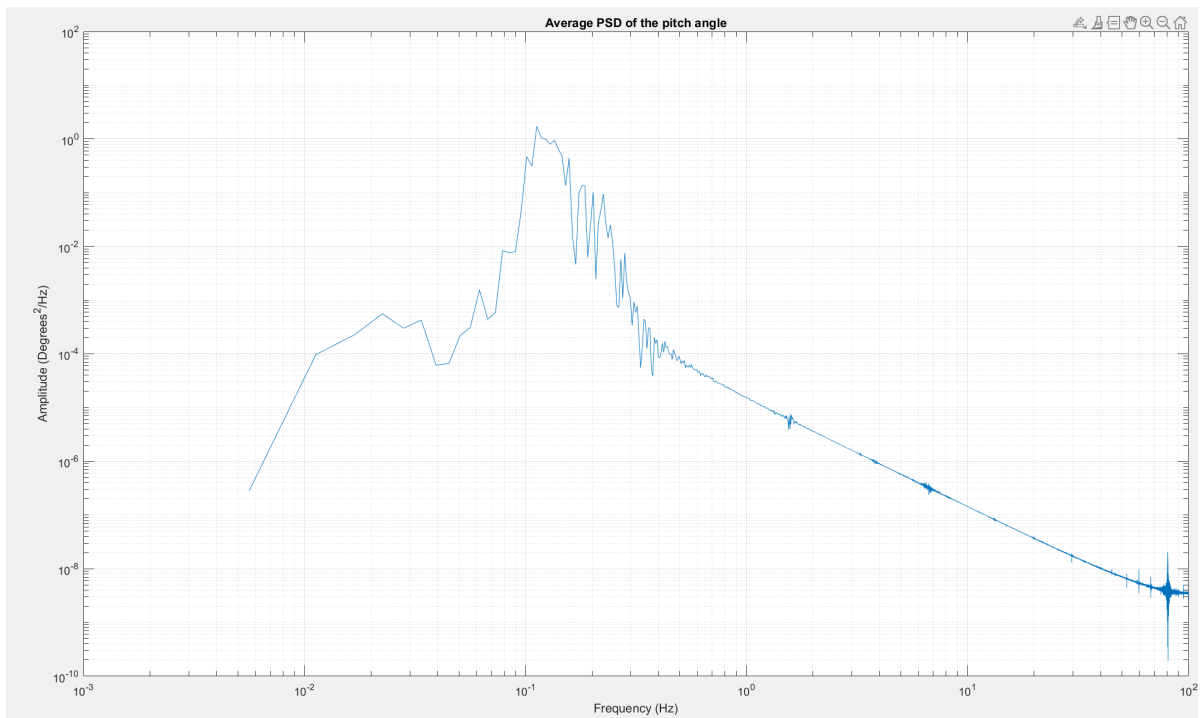


Figure B.2: Pitch Ship-Wave motion IMU frequency data set in logarithmic scale for both axes

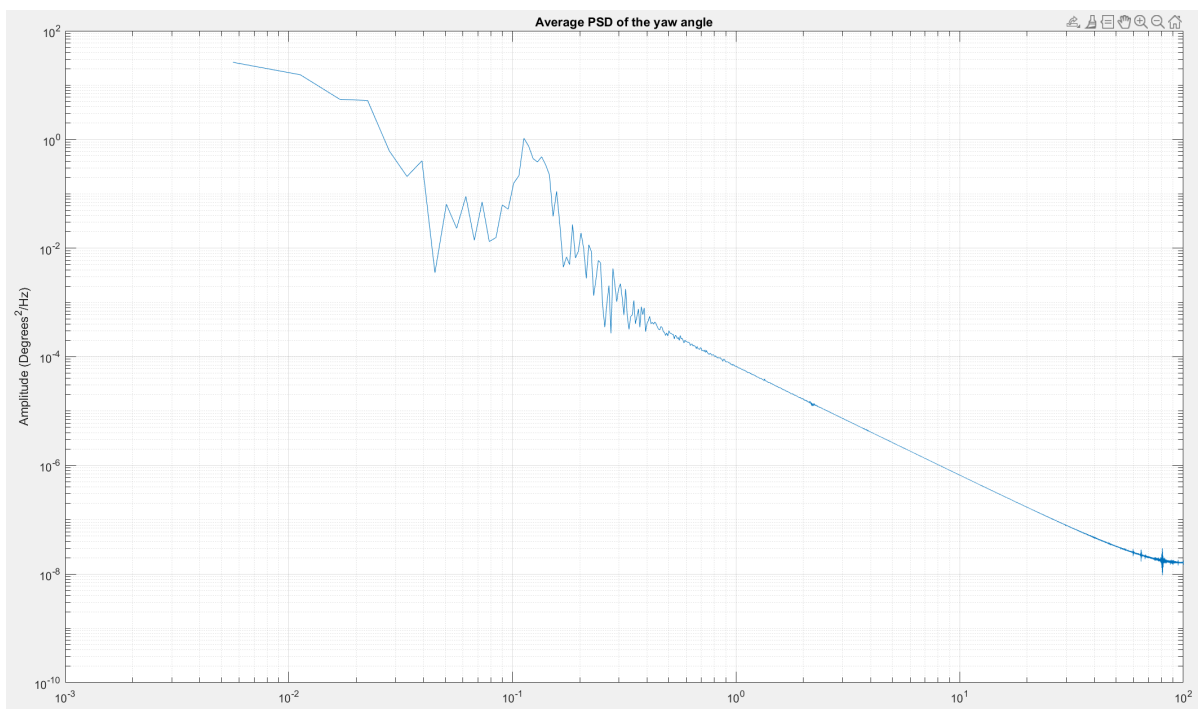


Figure B.3: Yaw Ship-Wave motion IMU frequency data set in logarithmic scale for both axes

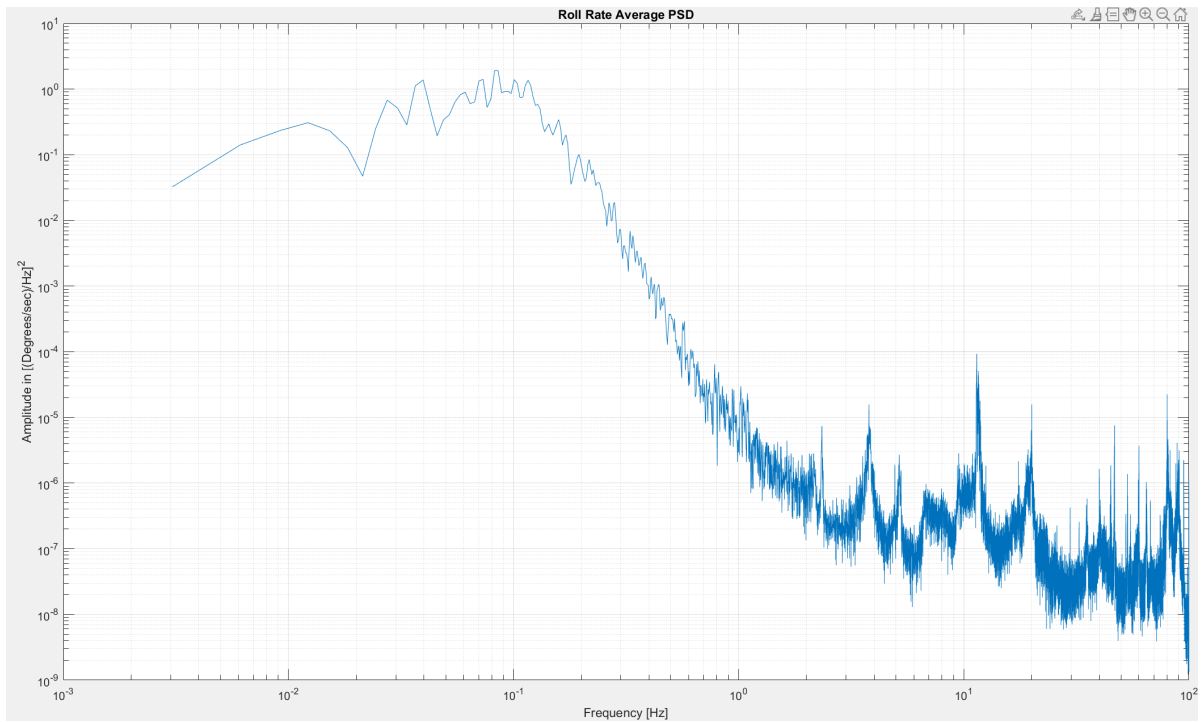


Figure B.4: Roll rate Ship-Wave motion IMU frequency data set in logarithmic scale for both axes

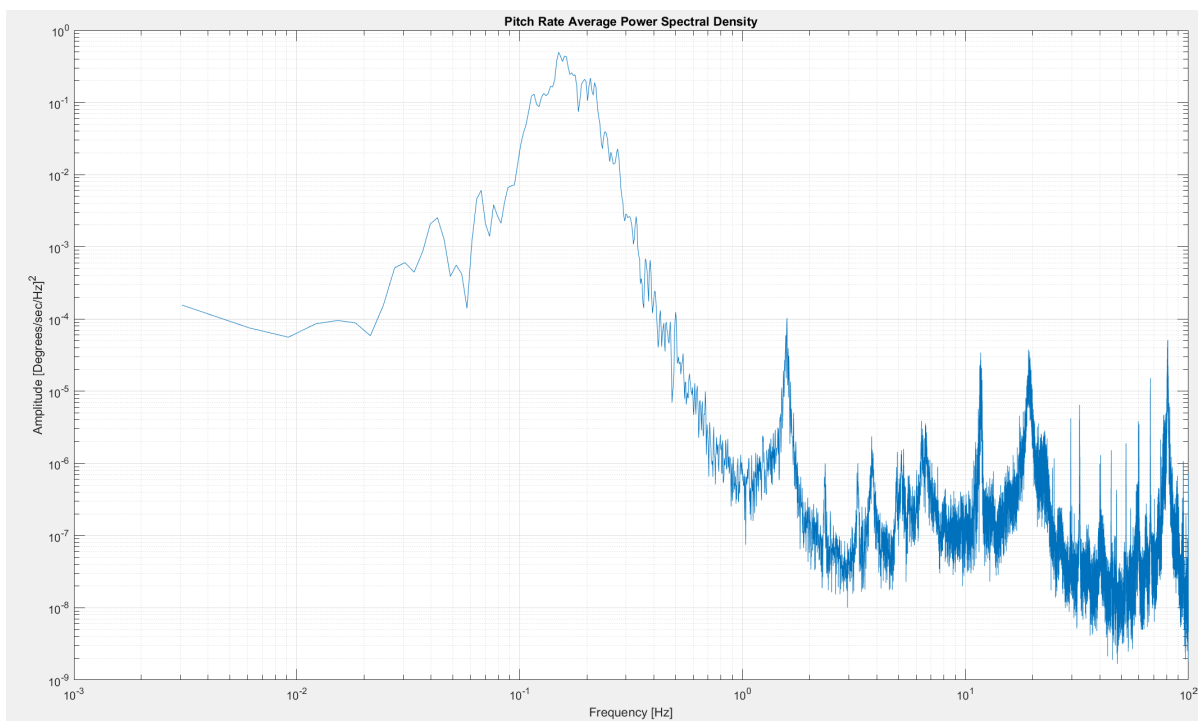


Figure B.5: Pitch rate Ship-Wave motion IMU frequency data set in logarithmic scale for both axes

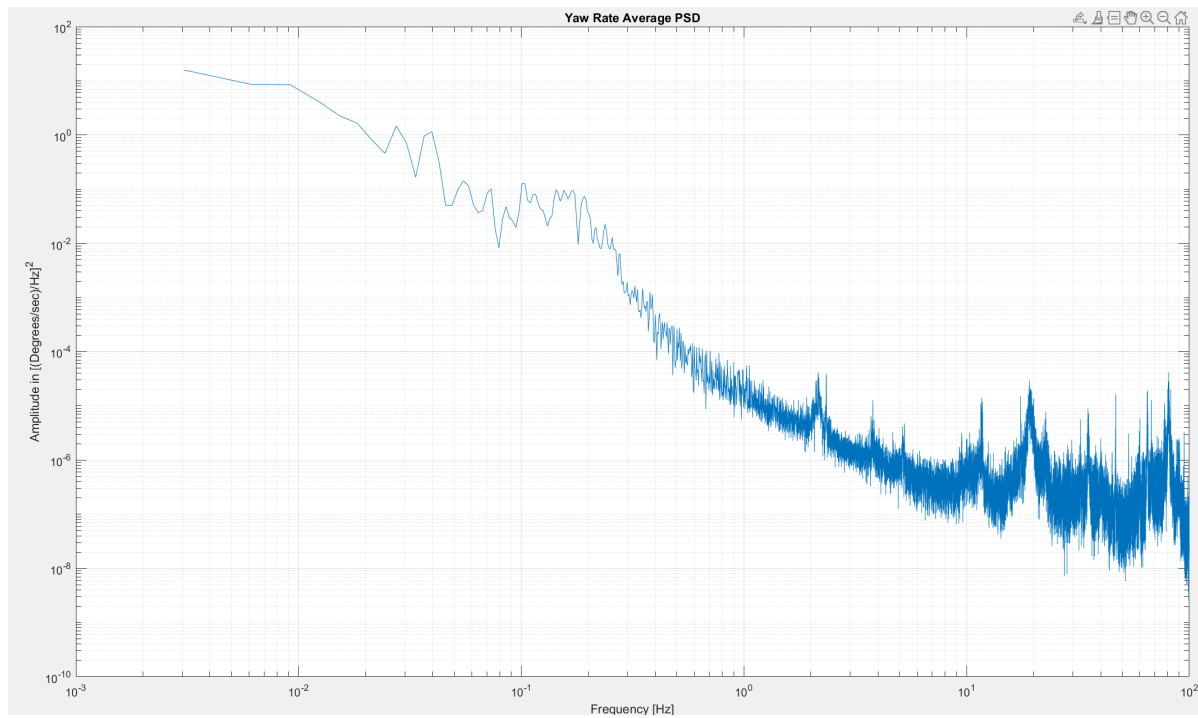
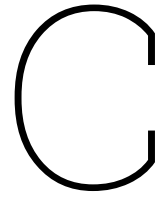


Figure B.6: Yaw rate Ship-Wave motion IMU frequency data set in logarithmic scale for both axes



Consistent sampling step-size search algorithm

```
1 % run chopping_data_set.m script to obtain the time data and angle data (RAW)
2 % input the raw time vector
3 % output the duration (above a threshold value) of continuous sampled data
4 % output as well the start and ending sampling number of the continuous sampled data
5 % function is located at the bottom
6
7 % threshold value = we are probably interested into continuous signals larger than t_min
  seconds
8 t_min = 43;
9 stepsize = 0.005; % for 200 Hz of sampling frequency
10 fixed_duration = t_min/stepsize; % minimum number of samples overpassing the t_min
  seconds threshold value
11
12 % total sample size
13 difference_timesteps = diff(time_vector_raw);
14 n_total = length(difference_timesteps);
15
16 % obtaining only the differences with respect to the nominal stepsize
17 time_diff = abs(difference_timesteps-stepsize);
18
19 % allowed variation in timesteps
20 dif_allowed = 1E-08; % for constant 5ms data
21 %dif_allowed = 1.1E-03; % for slight inconsistent 5ms (+/-1ms) data
22
23 % run the function
24 constant_sampling_intervals(time_diff,fixed_duration,stepsize,dif_allowed)
25
26 function constant_sampling_intervals(time_diff, fixed_duration,stepsize,dif_allowed)
27 % Function to find intervals of constant sampling times
28
29 % Initialize variables to store information
30 start_index = [];
31 end_index = [];
32 duration = [];
33
34 % Iterate through the time differences array
35 i = 1;
36 while i <= length(time_diff)
37 % Check if the current time difference is zero
38 if time_diff(i) <= dif_allowed
39 % Record the starting index
40 start_index = [start_index, i];
41 % Find the ending index and calculate the duration
42 j = i;
43 while j <= length(time_diff) && time_diff(j) <= dif_allowed
44 j = j + 1;
45 end
```

```

46         end_index = [end_index, j - 1];
47         duration = [duration, j - i];
48         % Move to the next index after the constant sampling time
49         i = j;
50     else
51         % Move to the next index
52         i = i + 1;
53     end
54 end
55
56 % Find intervals with duration greater than fixed_duration
57 long_intervals = find(duration > fixed_duration);
58
59 % Display information about long intervals
60 for k = 1:length(long_intervals)
61     fprintf('Interval%d: Start index=%d, End index=%d, Duration=%d samples / %d\n', k, ...
62           start_index(long_intervals(k)), end_index(long_intervals(k)), ...
63           duration(long_intervals(k)), (duration(long_intervals(k))*stepsize);
64 end
65 end

```

D

Pre and post-processing Digital Signal Processing algorithms and main script

```
1 % pre-processing of low and high frequency intervals (run chopping data sets to get the
   original data set loaded into the working space)
2 % then run the identification of continuous sampling file to get indexes of starting and
   ending indexes of the high and low frequencies data sets
3
4 % input imu data (change this variable to obtain PSD's of different angles)
5 input_imu_data = yaw;
6 % Change as well lines 54 (main average psd), 68 and 69 for filtered versions
7
8 % starting with the high frequency data sets:
9 sampling_frequency = 200; % assumed a constant sampling frequency of 200 Hz
10 time_vector1 = [];
11 time_vector2 = [];
12 time_vector3 = [];
13 psd_data = [];
14
15 psd_matrix = [];
16
17 for index = 1:length(total_start_indexes)
18     index_number = index, start_value = total_start_indexes(index), end_value =
        total_end_indexes(index)
19
20     % chopping the data set into smaller subsets
21     [chopped_time_vector, chopped_imu_data, chopped_duration] = chop_imu_data (
        total_start_indexes(index), total_end_indexes(index), input_imu_data, time_vector_raw,
        sampling_frequency );
22     % pre-processing of the data sets into windowed and padded signals
23     [padded_signal, windowed_signal, time_vector_padded] = preprocess_for_dft(
        chopped_imu_data, sampling_frequency);
24
25     % performing the DFT to the original chopped signal (convert time-varying data into the
        frequency domain as well as normalizing the amplitude)
26     [frequency_original, amplitude_original, psd_original, dft_original] =
        normalize_amplitude_dft_and_psd_creation (chopped_imu_data, sampling_frequency,
        chopped_time_vector);
27
28     % performing the DFT to the windowed chopped signal
29     [frequency_windowed, amplitude_windowed, psd_windowed, dft_windowed, comp_window] =
        normalize_compensation_amplitude_dft_and_psd_creation (windowed_signal,
        sampling_frequency, chopped_time_vector, dft_original);
30
31     % performing the DFT to the padded chopped signal
32     [frequency_padded, amplitude_padded, psd_padded, dft_padded] =
        normalize_compen_padd_amplitude_dft_and_psd_creation (padded_signal,
```

```

        sampling_frequency,time_vector_padded,dft_original,chopped_time_vector,comp_window);
33
34 % matrix with all psd's
35 psd_matrix(index,:)= psd_padded;
36
37 % % performing plots of the original, windowed and padded signals in time and frequency
    domains
38 % plot_signal_frequency_domains(chopped_imu_data, windowed_signal, padded_signal, ...
39 %                               chopped_time_vector, chopped_time_vector,
        time_vector_padded, ...
40 %                               amplitude_original, amplitude_windowed, amplitude_padded
        , ...
41 %                               frequency_original, frequency_windowed, frequency_padded
        )
42
43 % plot_signal_and_psd(frequency_original, amplitude_original, frequency_windowed,...
44 %                     amplitude_windowed, frequency_padded, amplitude_padded, psd_original, psd_windowed,
        psd_padded)
45
46 end
47
48 plot_signals(chopped_time_vector,chopped_imu_data,frequency_original,amplitude_original,"
    Original",chopped_time_vector,windowed_signal,frequency_windowed,amplitude_windowed,"
    Windowed")
49
50 plot_signals(chopped_time_vector,chopped_imu_data,frequency_original,amplitude_original,"
    Original",time_vector_padded,padded_signal,frequency_padded,amplitude_padded,"Windowed +
    Paded")
51
52 % calculate average of the psd
53 avg_psd = calculate_average_psd(psd_matrix);
54 avg_psd_yaw = avg_psd;
55
56 % Plot the average PSD
57 figure;
58 loglog(frequency_padded, avg_psd);
59 grid on
60 xlabel('Frequency(Hz)');
61 ylabel('PowerSpectralDensity');
62 title('AveragePowerSpectralDensity');
63
64 % Post processing settings for High frequency PSD data
65 % Post processing of the psd with HIGH pass filter
66 cut_off_freq_h = 2; % cut off frequency of the high pass filter [Hz]
67 [filtered_freq, filtered_psd] = high_pass_filter(frequency_padded,avg_psd,cut_off_freq_h);
68 filt_freq_psd_roll = filtered_freq;
69 filt_psd_roll = filtered_psd;
70
71 % % Post processing settings for Low to very low frequency PSD data
72 % % Post processing of the psd with HIGH pass filter
73 % cut_off_freq_h = 0.01; % cut off frequency of the high pass filter [Hz]
74 % [filtered_freq, filtered_psd] = high_pass_filter(frequency_padded,avg_psd,cut_off_freq_h);
75 %
76 % % Post processing of the psd with LOW pass filter
77 % cut_off_freq_l = 1; % cut off frequency of the low pass filter [Hz]
78 % [filtered_freq, filtered_psd] = low_pass_filter(filtered_freq, filtered_psd,cut_off_freq_l)
    ;
79
80 % Plot the post-processed average PSD
81 figure;
82 loglog(filtered_freq, filtered_psd);
83 grid on
84 xlabel('Frequency(Hz)');
85 ylabel('PowerSpectralDensity');
86 title('PostprocessedAveragePowerSpectralDensity');
87
88
89 function [chopped_time_vector, chopped_imu_data, chopped_duration] = chop_imu_data (
    start_index,end_index, imu_data, time_vector, sampling_frequency )
90 % function to chop the original data set into smaller slices
91

```

```

92 % creation of start till end indexes vector
93 start_and_end_indexes = start_index:end_index;
94
95 % cutting of the actual data
96 chopped_imu_data = imu_data(start_and_end_indexes,:);
97 chopped_time_vector = time_vector(start_and_end_indexes,:);
98 % restart the time vector so the data set start from zero seconds
99 chopped_time_vector = chopped_time_vector - chopped_time_vector(1);
100
101 % get chopped signal duration
102 chopped_duration = chopped_time_vector(end)+1/sampling_frequency;
103
104 end
105
106 function [padded_signal, windowed_signal, time_vector_padded] = preprocess_for_dft(signal,
107     sampling_frequency)
108     % Determine the number of samples
109     num_samples = length(signal);
110
111     % Apply windowing to the signal (here, using Hamming window)
112     % note that the signal of the real imu data needs to be transposed to be in the right
113     % shape
114     windowed_signal = signal' .* hamming(num_samples)';
115
116     % Find the next power of 2
117     next_power_of_2 = 2^nextpow2(num_samples);
118     %next_power_of_2 = next_power_of_2*2;
119
120     % Calculate the number of zeros to pad
121     num_zeros_to_pad = next_power_of_2 - num_samples;
122
123     % Pad zeros at the end of the signal
124     padded_signal = [windowed_signal, zeros(1, num_zeros_to_pad)];
125
126     % Generate the new time vector for the padded signal
127     time_vector_padded = (0:next_power_of_2-1) * (1/sampling_frequency);
128 end
129
130 function [frequency_vector, y_norm_amplitude, psd_imu_data, dft_data] =
131     normalize_amplitude_dft_and_psd_creation(imu_data, sampling_frequency, time_vector)
132 % determine sample size i.e. the length of the signal
133 N = length(time_vector);
134
135 % % (old procedure) creation of the frequency vector
136 % start = 0; % first frequency (DC) at zero Hz
137 % freq_time_step = 1/N; % frequency time step (resolution) or total number of frequency bins
138 % last_value = sampling_frequency/2-(freq_time_step); % last sampling frequency value before the Nyquist frequency
139 % frequency_vector = start:freq_time_step:last_value; % creation of the frequency vector (which contains all the frequency bins)
140
141 % % perform DFT of the time-varying data
142 % y = fft(imu_data);
143 %
144 % % absolute values (magnitude of the complex values) of the DFT
145 % y_norm_amplitude = abs(y);
146 %
147 % % sized to fit frequency vector (only one side of the DFT i.e. the real side containing the positive frequencies)
148 % y_norm_amplitude = y_norm_amplitude(1:length(frequency_vector));
149 %
150 % % normalization of the amplitude via division of the one sided DFT by the total sample size and double to account for the one-sided DFT
151 % y_norm_amplitude = 2*y_norm_amplitude/N;
152
153 % perform calculation of DFT with a two sided amplitude
154 y_norm_amplitude = abs(fft(imu_data))/N;
155

```

```

154 % sizing to one sided amplitude
155 y_norm_amplitude = y_norm_amplitude(1:N/2+1);
156
157 % double values except for DC and Nyquist frequencies
158 y_norm_amplitude(2:end-1) = 2*y_norm_amplitude(2:end-1);
159
160 % get dft for plotting purposes
161 dft_data = y_norm_amplitude;
162
163 % create a dedicated frequency vector
164 frequency_vector = 0:sampling_frequency/length(time_vector):sampling_frequency-1/
    sampling_frequency;
165
166 % adjust frequency vector
167 frequency_vector = frequency_vector(1:N/2+1);
168
169 % perform calculation of PSD with two sided power
170 psd_imu_data = (1/(sampling_frequency*N))*abs(fft(imu_data)).^2;
171
172 % sizing to one sided power
173 psd_imu_data = psd_imu_data(1:N/2+1);
174
175 % double values except for DC and Nyquist frequencies
176 psd_imu_data(2:end-1) = 2*psd_imu_data(2:end-1);
177
178 end
179
180 function [frequency_vector, y_norm_amplitude, psd_imu_data, dft_data, comp_window] =
    normalize_compensation_amplitude_dft_and_psd_creation (imu_data, sampling_frequency,
    time_vector, original_amplitude)
181 % determine sample size i.e. the length of the signal
182 N = length(time_vector);
183
184 % % (old procedure) creation of the frequency vector
185 % start = 0; % first frequency (DC) at
    zero Hz
186 % freq_time_step = 1/N; % frequency time step (resolution) or
    total number of frequency bins
187 % last_value = sampling_frequency/2-(freq_time_step); % last sampling frequency
    value before the Nyquist frequency
188 % frequency_vector = start:freq_time_step:last_value; % creation of the frequency
    vector (which contains all the frequency bins)
189 %
190 % % perform DFT of the time-varying data
191 % y = fft(imu_data);
192 %
193 % % absolute values (magnitude of the complex values) of the DFT
194 % y_norm_amplitude = abs(y);
195 %
196 % % sized to fit frequency vector (only one side of the DFT i.e. the real side containing
    the positive frequencies)
197 % y_norm_amplitude = y_norm_amplitude(1:length(frequency_vector));
198 %
199 % % normalization of the amplitude via division of the one sided DFT by the total sample
    size and double to account for the one-sided DFT
200 % y_norm_amplitude = 2*y_norm_amplitude/N;
201
202 % perform calculation of DFT with a two sided amplitude
203 y_norm_amplitude = abs(fft(imu_data))/N;
204
205 % sizing to one sided amplitude
206 y_norm_amplitude = y_norm_amplitude(1:N/2+1);
207
208 % double values except for DC and Nyquist frequencies
209 y_norm_amplitude(2:end-1) = 2*y_norm_amplitude(2:end-1);
210
211 % get dft for plotting purposes
212 dft_data = y_norm_amplitude;
213
214 % Calculate the total energy of the original signal
215 original_energy = sum(original_amplitude.^2);

```



```

216 % Calculate the total energy of the windowed signal
217 windowed_energy = sum(dft_data .^ 2);
218
219 % Calculate the compensation factor
220 comp_factor = sqrt(original_energy / windowed_energy);
221 fprintf('compensation_factor=%d',comp_factor)
222 comp_window = comp_factor;
223
224 % compensated dft_data
225 dft_data = dft_data*comp_factor;
226
227 % create a dedicated frequency vector
228 frequency_vector = 0:sampling_frequency/length(time_vector):sampling_frequency-1/
    sampling_frequency;
229
230 % adjust frequency vector
231 frequency_vector = frequency_vector(1:N/2+1);
232
233 % perform calculation of PSD with two sided power
234 psd_imu_data = (1/(sampling_frequency*N))*abs((fft(imu_data))*comp_factor).^2;
235
236 % sizing to one sided power
237 psd_imu_data = psd_imu_data(1:N/2+1);
238
239 % double values except for DC and Nyquist frequencies
240 psd_imu_data(2:end-1) = 2*psd_imu_data(2:end-1);
241
242 end
243
244 function [frequency_vector, y_norm_amplitude, psd_imu_data, dft_data] =
    normalize_compen_padd_amplitude_dft_and_psd_creation (imu_data, sampling_frequency, ...
245 time_vector,original_amplitude, original_time_vector, comp_window)
246 % determine sample size i.e. the length of the signal
247 N = length(time_vector);
248 N Og = length(original_time_vector);
249
250 % % (old procedure)creation of the frequency vector
251 % start = 0; % first frequency (DC) at
    zero Hz
252 % freq_time_step = 1/N; % frequency time step (resolution) or
    total number of frequency bins
253 % last_value = sampling_frequency/2-(freq_time_step); % last sampling frequency
    value before the Nyquist frequency
254 % frequency_vector = start:freq_time_step:last_value; % creation of the frequency
    vector (which contains all the frequency bins)
255
256 %
257 % % perform DFT of the time-varying data
258 % y = fft(imu_data);
259 %
260 % % absolute values (magnitude of the complex values) of the DFT
261 % y_norm_amplitude = abs(y);
262 %
263 % % sized to fit frequency vector (only one side of the DFT i.e. the real side containing
    the positive frequencies)
264 % y_norm_amplitude = y_norm_amplitude(1:length(frequency_vector));
265 %
266 % % normalization of the amplitude via division of the one sided DFT by the total sample
    size and double to account for the one-sided DFT
267 % y_norm_amplitude = 2*y_norm_amplitude/N;
268
269 % perform calculation of DFT with a two sided amplitude
270 y_norm_amplitude = abs(fft(imu_data))/N;
271
272 % sizing to one sided amplitude
273 y_norm_amplitude = y_norm_amplitude(1:N/2+1);
274
275 % double values except for DC and Nyquist frequencies
276 y_norm_amplitude(2:end-1) = 2*y_norm_amplitude(2:end-1);
277
278 % get dft for plotting purposes

```

```

279     dft_data = y_norm_amplitude;
280
281     % Calculate the compensation factor
282     comp_factor = N/N_og;
283     % total compensation factor
284     comp_total = comp_factor*comp_window;
285     fprintf('compensation_factor=%d', comp_factor)
286
287     % compensated dft_data
288     dft_data = dft_data*comp_window;
289
290     % create a dedicated frequency vector
291     frequency_vector = 0:sampling_frequency/length(time_vector):sampling_frequency-1/
        sampling_frequency;
292
293     % adjust frequency vector
294     frequency_vector = frequency_vector(1:N/2+1);
295
296     % perform calculation of PSD with two sided power
297     psd_imu_data = (1/(sampling_frequency*N))*abs((fft(imu_data))*comp_window).^2;
298
299     % sizing to one sided power
300     psd_imu_data = psd_imu_data(1:N/2+1);
301
302     % double values except for DC and Nyquist frequencies
303     psd_imu_data(2:end-1) = 2*psd_imu_data(2:end-1);
304
305 end
306
307 function avg_psd = calculate_average_psd(psd_matrix)
308     % Number of intervals (rows) in the PSD matrix
309     num_intervals = size(psd_matrix, 1);
310
311     % Sum all PSDs in the matrix
312     psd_sum = sum(psd_matrix, 1); % Sum across rows (axis 1)
313
314     % Calculate the average by dividing by the number of intervals
315     avg_psd = psd_sum / num_intervals;
316 end
317
318 function [new_freq, filtered_psd] = low_pass_filter(freq, psd, cut_off_freq)
319     % Validate inputs
320     if length(freq) ~= length(psd)
321         error('Frequency_vector_and_PSD_must_have_the_same_length. ');
322     end
323
324     % Find indices of frequencies that are below the cut-off frequency
325     low_pass_indices = find(freq <= cut_off_freq);
326
327     % Create a new frequency vector with only the frequencies below the cut-off
328     new_freq = freq(low_pass_indices);
329
330     % Create the filtered PSD by keeping only the values below the cut-off frequency
331     filtered_psd = psd(low_pass_indices);
332 end
333
334 function [new_freq, filtered_psd] = high_pass_filter(freq, psd, cut_off_freq)
335     % Validate inputs
336     if length(freq) ~= length(psd)
337         error('Frequency_vector_and_PSD_must_have_the_same_length. ');
338     end
339
340     % Find indices of frequencies that are above the cut-off frequency
341     high_pass_indices = find(freq >= cut_off_freq);
342
343     % Create a new frequency vector with only the frequencies above the cut-off
344     new_freq = freq(high_pass_indices);
345
346     % Create the filtered PSD by keeping only the values above the cut-off frequency
347     filtered_psd = psd(high_pass_indices);
348 end

```

```

349
350 function plot_signal_frequency_domains(original_signal, windowed_signal, padded_signal, ...
351                                     time_vector_original, time_vector_windowed,
352                                     time_vector_padded, ...
353                                     amplitude_original, amplitude_windowed,
354                                     amplitude_padded, ...
355                                     frequency_original, frequency_windowed,
356                                     frequency_padded)
357
358 % Create a figure
359 figure;
360
361 % Plot original signal in time domain
362 subplot(2,3,1);
363 plot(time_vector_original, original_signal);
364 title('Original_Signal_(Time_Domain)');
365 xlabel('Time');
366 ylabel('Amplitude');
367
368 % Plot windowed signal in time domain
369 subplot(2,3,2);
370 plot(time_vector_windowed, windowed_signal);
371 title('Windowed_Signal_(Time_Domain)');
372 xlabel('Time');
373 ylabel('Amplitude');
374
375 % Plot padded signal in time domain
376 subplot(2,3,3);
377 plot(time_vector_padded, padded_signal);
378 title('Padded_Signal_(Time_Domain)');
379 xlabel('Time');
380 ylabel('Amplitude');
381
382 % Plot original signal in frequency domain
383 subplot(2,3,4);
384 loglog(frequency_original, amplitude_original);
385 grid on
386 title('Original_Signal_(Frequency_Domain)');
387 xlabel('Frequency');
388 ylabel('Amplitude');
389
390 % Plot windowed signal in frequency domain
391 subplot(2,3,5);
392 loglog(frequency_windowed, amplitude_windowed);
393 grid on
394 title('Windowed_Signal_(Frequency_Domain)');
395 xlabel('Frequency');
396 ylabel('Amplitude');
397
398 % Plot padded signal in frequency domain
399 subplot(2,3,6);
400 loglog(frequency_padded, amplitude_padded);
401 grid on
402 title('Padded_Signal_(Frequency_Domain)');
403 xlabel('Frequency');
404 ylabel('Amplitude');
405
406 end
407
408 function plot_signal_and_psd(freq_original, amp_original, freq_windowed, amp_windowed,
409                             freq_padded, amp_padded, psd_original, psd_windowed, psd_padded)
410 % Create figure
411 figure;
412
413 % Plot signals in frequency domain
414 subplot(3,2,1);
415 plot(freq_original, amp_original);
416 title('Original_Signal_(Frequency_Domain)');
417 xlabel('Frequency');
418 ylabel('Amplitude');
419 set(gca, 'XScale', 'log');
420 set(gca, 'YScale', 'log');
421 grid on;

```

```

416
417 subplot(3,2,3);
418 plot(freq_windowed, amp_windowed);
419 title('Windowed_Signal_(Frequency_Domain)');
420 xlabel('Frequency');
421 ylabel('Amplitude');
422 set(gca, 'XScale', 'log');
423 set(gca, 'YScale', 'log');
424 grid on;
425
426 subplot(3,2,5);
427 plot(freq_padded, amp_padded);
428 title('Padded+Windowed_Signal_(Frequency_Domain)');
429 xlabel('Frequency');
430 ylabel('Amplitude');
431 set(gca, 'XScale', 'log');
432 set(gca, 'YScale', 'log');
433 grid on;
434
435 % Plot PSDs
436 subplot(3,2,2);
437 plot(freq_original, psd_original);
438 title('Original_Signal_(PSD)');
439 xlabel('Frequency');
440 ylabel('Power_Spectral_Density');
441 set(gca, 'XScale', 'log');
442 set(gca, 'YScale', 'log');
443 grid on;
444
445 subplot(3,2,4);
446 plot(freq_windowed, psd_windowed);
447 title('Windowed_Signal_(PSD)');
448 xlabel('Frequency');
449 ylabel('Power_Spectral_Density');
450 set(gca, 'XScale', 'log');
451 set(gca, 'YScale', 'log');
452 grid on;
453
454 subplot(3,2,6);
455 plot(freq_padded, psd_padded);
456 title('Padded+Windowed_Signal_(PSD)');
457 xlabel('Frequency');
458 ylabel('Power_Spectral_Density');
459 set(gca, 'XScale', 'log');
460 set(gca, 'YScale', 'log');
461 grid on;
462 end
463
464
465 function plot_signals(time1, signal1_time, freq1, signal1_freq, name1, time2, signal2_time,
466     freq2, signal2_freq, name2)
467     % Create a figure
468     figure;
469
470     % Subplot 1: Signal 1 Time Domain
471     subplot(2, 2, 1);
472     plot(time1, signal1_time);
473     grid on
474     title([name1 'signal_(Time_Domain)']);
475     xlabel('Time_[seconds]');
476     ylabel('Amplitude_[Degrees/sec]');
477
478     % Subplot 2: Signal 1 Frequency Domain
479     subplot(2, 2, 2);
480     loglog(freq1, signal1_freq);
481     grid on
482     title([name1 'signal_(Frequency_Domain)']);
483     xlabel('Frequency_[Hz]');
484     ylabel('Magnitude_[Degrees/sec]');
485
486     % Subplot 3: Signal 2 Time Domain

```

```
486 subplot(2, 2, 3);
487 plot(time2, signal2_time);
488 grid on
489 title([name2 'signal_(Time_Domain)']);
490 xlabel('Time_[seconds]');
491 ylabel('Amplitude_[Degrees/sec]');
492
493 % Subplot 4: Signal 2 Frequency Domain
494 subplot(2, 2, 4);
495 loglog(freq2, signal2_freq);
496 grid on
497 title([name2 'signal_(Frequency_Domain)']);
498 xlabel('Frequency_[Hz]');
499 ylabel('Magnitude_[Degrees/sec]');
500 end
```

E

Additional information, plots and code from actuator loop models chapter

E.1. Precise Pointing Actuator System accompanying plots, descriptions and code

E.1.1. QPD tracking error estimation model description

This subsection aims to provide additional information regarding the estimation procedure utilized to determine the tracking error. Since direct measurements from the tracking sensor are not available, the approach taken to recreate the disturbances has been based on modeling them as white Gaussian noise. Moreover to simplify the process of relating the Signal-to-Noise-Ratio (SNR) to a tracking error typical engineering approximations have been utilized. The following list includes the key assumptions employed to determine the tracking error:

- **Gaussian distribution:** The tracking error disturbances follow a Gaussian (normal) distribution, characterized by a zero mean and a standard deviation.
- **Stationarity:** The statistical properties of the noise signal are constant over time, that is the noise signal is invariant with time.
- **Uncorrelated Samples:** The value of the noise signal at a specific point, does not influence the value of the noise signal at another time point.
- **Independence from signal:** The noise is independent of the signal that is being measured. The true signal and the noise signal are uncorrelated and do not influence each other.
- The measurement error of the sensor is assumed to be linearly related to the inverse of the SNR
- The noise in the sensor is assumed to be Additive White Gaussian Noise (AWGN)
- The approximation assumes the SNR level to be in the high sensitivity regime of the QPD sensor, such that the signal power is much higher than the noise power.
- Small angle approximation to relate the measurement error in meters to degrees.
- Ideal measurement system with no other sources of errors such as calibration errors, quantization noise or other systematic errors.

For further information regarding the assumptions and background theory regarding modeling stochastic processes, the readers are referred to the works by [22]. With the assumptions outlined it is possible to discuss the equations utilized to determine the tracking error. The following noise equations have been extracted from [23] part of the Space Instrumentation Engineering lecture notes (AE4-880). The calculations start with Equation E.1, Equation E.2 and Equation E.3. These equations represent the main sources of disturbances for sensors with photodiodes.

$$I_{sd}^2 = 2 * e * I_d * \Delta f \quad (E.1)$$

Where e = electron elementary charge [e], I_d represents the sensor dark current [A] and Δf is the sensor electrical bandwidth [Hz].

$$I_j^2 = \frac{4 * k * T * \Delta f}{R_{sh}} \quad (E.2)$$

Where k represents the Boltzmann constant [J/K], T absolute operational temperature [K] and R_{sh} is the shunt resistance [Ω].

$$I_{ss}^2 = 2 * e * I_s * \Delta f \quad (E.3)$$

Where I_s is the photocurrent created in the detector when the light is converted via the photoelectric effect into electrons [A]. Adding up the differences sources of noise and utilizing this to divide over available signal at the detector it is then possible to arrive at the linear SNR relation, as described by Equation E.4.

$$SNR = \frac{P_{laser-in} * R}{\sqrt{I_{sd}^2 + I_j^2 + I_{ss}^2}} \quad (E.4)$$

Where P_{laser} is the incident laser beam power at the detector [W] and R is the sensor responsivity (Photo-sensitivity) [A/W]. Next is the approximation of the standard deviation, to determine this parameter the assumption of being in the high sensitivity regime of the QPD sensor is utilized to approximate the inverse relationship described by Equation E.5.

$$\sigma \propto \frac{1}{SNR} \quad (E.5)$$

Now with a relation that can be utilized to determine the standard deviation, the next step involves mapping the standard deviation to physical units, in this case, a direct proportionality relation equal to the side length of the QPD sensor is utilized, where the gap between the photodiodes is assumed to be small enough to be neglect, resulting on a side length equaling the total diameter of the QPD sensor. This relation allows us to map the Standard deviation to meters, next is mapping the relation to degrees. A triangular small angle approximation is utilized, where the vertical side is equal to the standard deviation in meters, while the bottom side is the focal length of the optical setup, by making use of the small angle approximation, the tangent component can be then removed. Lastly, as the small angle approximation yields the value in radians, the value is then multiplied with a conversion ratio to end up with a value in degrees. The calculation to achieve this is outlined by Equation E.6.

$$\theta_{error} = \frac{\sigma * K_{direct}}{D_{focal\ length}} * \frac{180}{\pi} \quad (E.6)$$

Where K_{direct} is the direct length proportionality relation [m] and $D_{focal\ length}$ is the optical assembly focal length [m].

E.1.2. QPD tracking error estimation model code

```

1 % parameters based on InGaAs PIN photodiode G6849-01 from Hamamatsu
2
3 NEP = 1e-14; % W/sqrt(Hz)
4 Idark = 15; % nA
5 responsivity = 0.85; % A/W
6 bandwidth = 12e+06; % Hz
7 P_laser = (0.97^5)*1e-6; % W 5 optical components in between
8 area = 1e-6; % m^2 (1 mm^2)
9 distance = 1.5; % meter
10 T = 25; % °C
11 temp_coeff_Id = 1.09; % 1/°C
12 terminal_cap = 40e-12; % F
13 shunt_resistance = 80e6; % Ohms
14 detectivity = 1e12; % cm*(sqrt(Hz))/W
15 wavelength = 1550E-09; % m
16
17 [sigma_deg, sigma] = improvedQuadCellNoiseAnalysis(Idark, responsivity, bandwidth, P_laser,
18     area, distance, ...
19     T, shunt_resistance, wavelength);

```

```

20
21 % creation of the QPD psd errors
22 % Statistical error properties
23 mean = 0; % Mean of the positional error
24 std_dev = sigma_deg; % Standard deviation of the positional error (square root of the
    variance) 0.1 arcseconds for QPD
25
26 % sizing properties to match the original reference signal
27 sampling_rate = 200; % Sampling rate in Hz
28 num_samples = length(padded_signal); % Number of data points required to obtain an estimate
    of the PSD SW function
29
30 % Generate simulated positional error data
31 positional_error = mean + std_dev * randn(1, num_samples);
32
33 % convert positional error vector into PSD error vector
34 [frequency_vector, y_norm_amplitude, psd_error_data] =
    normalize_amplitude_dft_and_psd_creation (positional_error, sampling_rate);
35
36 sigma_dot_x = sqrt(trapz(frequency_vector, psd_error_data));
37
38 fprintf('Standard deviation of the rate of tracking error is: %.2e degrees/second\n',
    sigma_dot_x);
39
40
41 function [sigma_deg, sigma] = improvedQuadCellNoiseAnalysis(Idark, responsivity, bandwidth,
    P_laser, area, distance, ...
42     T, shunt_resistance, wavelength)
43 % Constants
44 q = 1.60217662e-19; % Charge of an electron (Coulombs)
45 k = 1.380649e-23; % Boltzmann constant (J/K)
46 h = 6.62607e-34; % planck's constant (m^2*kg/s)
47 v = 299792458; % Speed of light (m/s)
48
49 % Convert dark current to Amps
50 Idark = Idark * 1e-9;
51
52 % Shot noise current due to dark current (A/sqrt(Hz))
53 I_shot_dark = 2 * q * Idark * bandwidth;
54
55 % Thermal noise current due to shunt resistance
56 I_johnson = (4 * k * (T + 273.15)*bandwidth) / shunt_resistance;
57
58 % Determine quantum efficiency
59 QE = (responsivity * h*(v/wavelength))/(q);
60 % Photon-current in the QPD
61 I_s = (QE*wavelength*q*P_laser)/(h*v);
62
63 % Statistical photon noise
64 I_ss = 2*q*I_s*bandwidth;
65
66 % Total noise current (A)
67 I_noise = sqrt(I_ss + I_shot_dark + I_johnson);
68
69 % Total photocurrent (A) assuming given laser power
70 I_total = P_laser * responsivity;
71
72 % Standard deviation of the position estimate (normalized units)
73 sigma_norm = I_total / I_noise;
74
75 % Convert normalized position error to physical units
76 quad_side = sqrt(area); % Side length of the entire detector
77 sigma = (1/sigma_norm) * quad_side;
78
79 % Convert position error in meters to angular error in radians
80 sigma_theta_rad = sigma / distance;
81
82 % Convert angular error from radians to degrees
83 sigma_deg = sigma_theta_rad * (180 / pi);
84
85 % Display the results

```



```

86     fprintf('Standard deviation of the position estimate: %.2e meters\n', sigma);
87     fprintf('Standard deviation of the angular estimate: %.2e degrees\n', sigma_deg);
88 end
89
90
91
92 function [frequency_vector, y_norm_amplitude, psd_imu_data] =
    normalize_amplitude_dft_and_psd_creation (imu_data, sampling_frequency)
93 % determine sample size i.e. the length of the signal
94 N = length(imu_data);
95
96 % perform calculation of DFT with a two sided amplitude
97 y_norm_amplitude = abs(fft(imu_data))/N;
98
99 % sizing to one sided amplitude
100 y_norm_amplitude = y_norm_amplitude(1:N/2+1);
101
102 % double values except for DC and Nyquist frequencies
103 y_norm_amplitude(2:end-1) = 2*y_norm_amplitude(2:end-1);
104
105 % create a dedicated frequency vector
106 frequency_vector = 0:sampling_frequency/N:sampling_frequency-1/sampling_frequency;
107
108 % adjust frequency vector
109 frequency_vector = frequency_vector(1:N/2+1);
110
111 % perform calculation of PSD with two sided power
112 psd_imu_data = (1/(sampling_frequency*N))*abs(fft(imu_data)).^2;
113
114 % sizing to one sided power
115 psd_imu_data = psd_imu_data(1:N/2+1);
116
117 % double values except for DC and Nyquist frequencies
118 psd_imu_data(2:end-1) = 2*psd_imu_data(2:end-1);
119
120 end

```

E.1.3. Augmented Open Loop and Closed-Loop response

To understand the achievable performance of the system, it is relevant to compare the difference between the Augmented Open-loop and Closed-loop system frequency responses. First, It is beneficial to introduce the transfer function utilized to determine such frequency responses. Starting with Equation E.7 which denotes the transfer function of the Augmented Open-loop frequency response. While Equation E.8 determines the Closed Loop system response.

$$A.O.L. = C_{fsm} * G_{fsm} * P_{QPD} \quad (E.7)$$

$$C.L. = \frac{C_{fsm} * G_{fsm}}{1 + C_{fsm} * G_{fsm} * P_{QPD}} \quad (E.8)$$

Having defined the Open and Closed loop transfer functions, it is then possible to combine their frequency responses into a single plot, as shown in Figure E.1. It is relevant to highlight the aspect that at lower frequencies, the gain margin of the Augmented Open-Loop has a significant positive margin, indicating an adequate system performance across that frequency range.

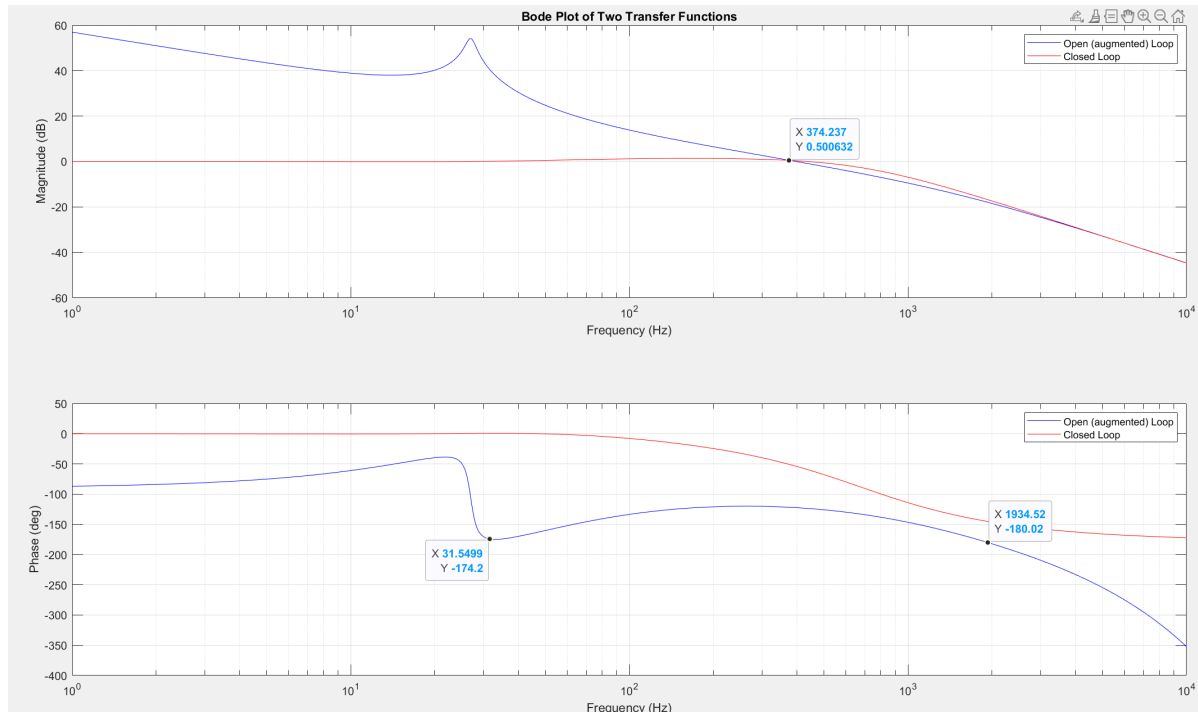


Figure E.1: Open and Closed loop Bode plot of the frequency response of the designed PPAS

It is also relevant to highlight that prior to the natural resonant frequency, the open-loop behavior approximates integral (control) behavior with a negative slope. Moreover across the first natural resonant frequency, as expected the phase margin quickly drops, for which the point at which the derivative part of the controller has been adjusted, with the intention to lift the phase and prevent it from going over the -180 (degrees) point. It can also be seen from the Closed loop response that the system starts losing gain margin close to the maximum controller bandwidth occurring close to 374 Hz (Note controller bandwidth is set at 398 Hz), indicating possible disturbance rejection close to the controller bandwidth limit.

E.1.4. Augmented Open loop and loop sensitivity function

The next step to improve the understanding of the performance of the system, is to determine the capacity of the system to reject disturbances. To achieve this a plot with the Augmented Open loop and Loop Sensitivity transfer functions, proves insightful as their frequency response can provide information about the robustness of the control design of the system. The Augmented Open loop function is included as it helps to determine the response and bandwidth of the system to reference signals.

Next is the Loop Sensitivity function, which can be utilized to determine the disturbance rejection performance of the control system. The fundamental idea of this transfer function is to measure how variations in controller and plant design affect the closed-loop transfer function. The sensitivity function can be described as in Equation E.9.

$$L.S. = \frac{1}{1 + C_{fsm} * G_{fsm} * P_{QPD}} \quad (\text{E.9})$$

Utilizing the Augmented Open Loop and Loop Sensitivity functions, it is then possible to recreate the frequency response of the system according to those functions, this is shown in Figure E.2. It is possible to denote that the large gain margin in the Augmented Open Loop function at lower frequencies, is equally visible in the loop sensitivity function, as a large negative gain margin. It is of interest the point at which both the Augment Open Loop and Loop sensitivity functions cross the zero dB gain margin, indicating the performance threshold of the system, the loop sensitivity remains close to the $\sim 2.7 \text{ dB}$ indicating a slight sensitivity of the system across these frequencies, gradually returning to the $\sim 0 \text{ dB}$ gain margin close to 4.5 KHz . This could indicate that the system could have a slight sensitivity to

disturbances at higher frequencies. This could also be a result of the relatively low damping coefficient of the FSM model, which could make the system prone to be sensitive to oscillatory behavior.

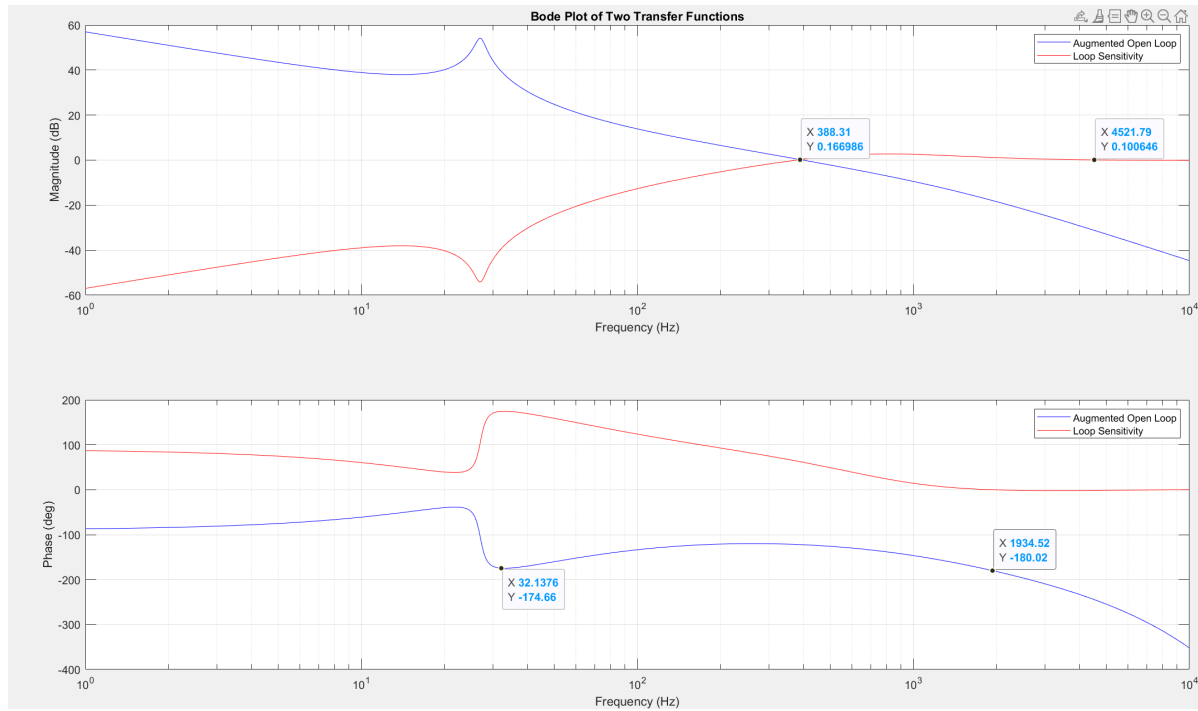


Figure E.2: Augmented open loop and Loop Sensitivity of the PPAS

E.2. Coarse Pointing Actuator System accompanying plots and descriptions

E.2.1. Estimation of rotational moments of inertia of the Coarse Pointing Actuator System

This section aims to provide additional information regarding the estimation process utilized to estimate the rotational moments of inertia of the L-350 planewave direct drive telescope mount. The process starts by performing measurements of the distance from the rotational axis to the locations of the center of mass of the 3 estimation blocks representing the L-350 mount. To facilitate this process Figure E.3, provides an overview of the naming convention utilized to denote the location of the distances utilized to estimate the rotational moments of inertia. With the overview of the locations of the distances, Table E.1 provides an overview of the estimated distance of such locations to the azimuth and elevation axis.

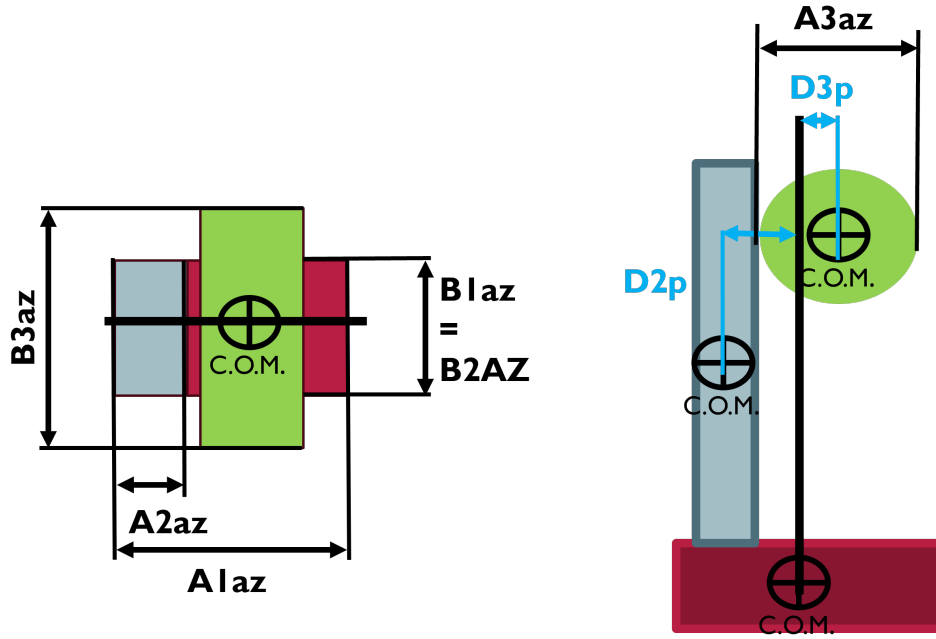


Figure E.3: Locations of the approximated distances of the center of mass of the 3 main blocks representing the I-350 mount

Table E.1: Specification of the distances utilized to determine the rotational moments of inertia

Specification	Estimated distance [cm]	Specification	Estimated distance [cm]
A1az	50	B1az	30
A2az	12	B2az	30
A3az	30	B3az	70
A1el	30	B1el	70
D2p	12	D3p	3

Next is to define the equations utilized to estimate the individual contributions of each of the blocks to each of the rotational axis, note that the parallel axis is utilized to estimate the rotational contribution for the center of mass located away from the rotational axis. Note that M1 corresponds to the azimuth base ($\approx 31.75 \text{ kg}$), m2 corresponds to the elevation arm ($\approx 18.10 \text{ kg}$) and m3 corresponds to the dummy telescope payload (20 kg).

$$J_{az1} = \frac{1}{12} * m_1 * (A_{1az}^2 + B_{1az}^2) = 0.513 \text{ kg} * m^2 \quad (\text{E.10})$$

$$J'_{az2} = \frac{1}{12} * m_1 * (A_{2az}^2 + B_{2az}^2) = 0.276 \text{ kg} * m^2 \quad (\text{E.11})$$

$$J_{az2} = J'_{az2} + m_2 * D_{2p}^2 = 0.813 \text{ kg} * m^2 \quad (\text{E.12})$$

$$J'_{az3} = \frac{1}{12} * m_3 * (3 * (\frac{A_{3az}}{2})^2 + B_{3el}^2) = 0.930 \text{ kg} * m^2 \quad (\text{E.13})$$

$$J_{az3} = J'_{az3} + m_3 * D_{3p}^2 = 0.938 \text{ kg} * m^2 \quad (\text{E.14})$$

Now with the previous equations it is possible to then determine the total contribution per axis, this is described by the following two equations, note that the elevation rotational moment of inertia only contains the contribution of the mass of the dummy telescope, as it is located in the elevation axis,

however, it is off center from the azimuth axis, thus the contribution from the parallel axis theorem is applied only to the azimuth total rotational moment of inertia.

$$J_{azt} = J_{az1} + J_{az2} + J_{az3} = 2.264 \text{ kg} * \text{m}^2 \quad (\text{E.15})$$

$$J_{elt} = J'_{az3} = 0.930 \text{ kg} * \text{m}^2 \quad (\text{E.16})$$

E.2.2. Elevation open loop response

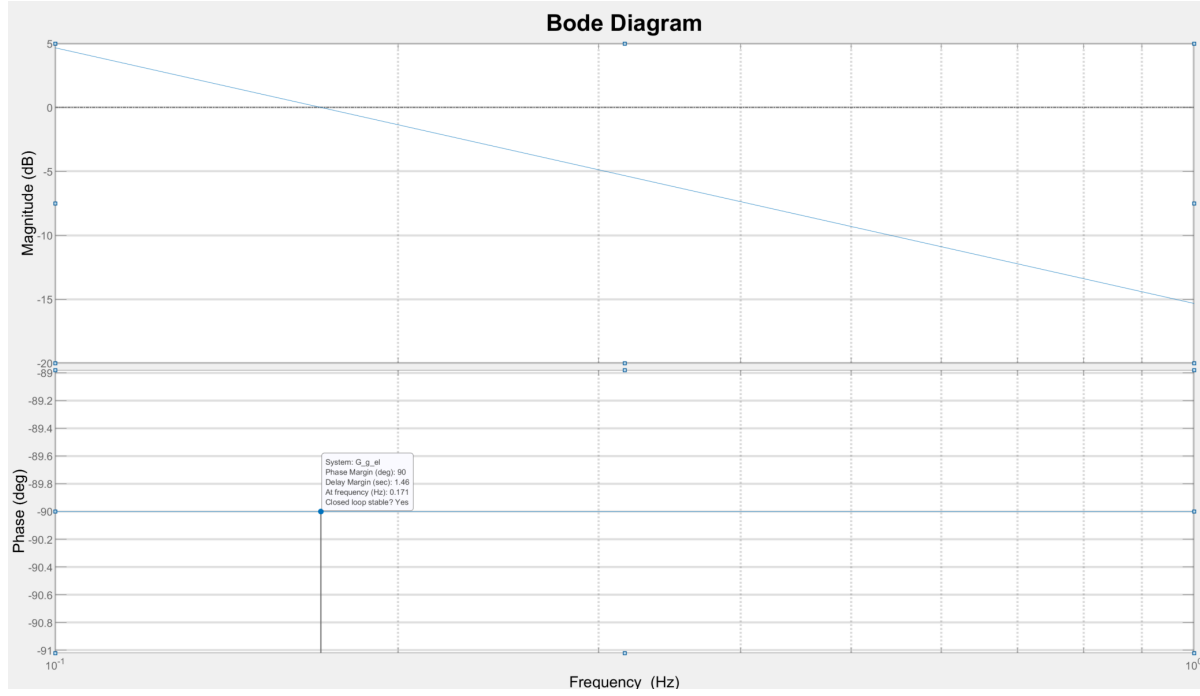


Figure E.4: Bode plot of the frequency response of the CPAS elevation actuator transfer function

E.2.3. Augmented Open loop and Closed loop response

To understand the difference between the achievable performance by incorporating feedback into the system and operating with no feedback, it is then useful to determine the frequency response to the augmented open loop and closed loop responses. In Equation E.17 and Equation E.18 the transfer functions of the augmented open loop and closed loop of the CPAS are fully described. **Note:** The following equations are derived for the generic G_{gimbal} actuator transfer function, which is representative of both azimuth and elevation actuators.

$$(A.O.L.)_{cpas} = C_{gimbal} * G_{gimbal} * P_{QPD} \quad (\text{E.17})$$

$$(O.L.)_{cpas} = \frac{C_{gimbal} * G_{gimbal}}{1 + C_{gimbal} * G_{gimbal} * P_{QPD}} \quad (\text{E.18})$$

With the Augmented Open loop and Closed loop transfer functions, it is then possible to create Figure E.5. Distinguishable from these plots are the effects of the controller bandwidth as both the augmented open-loop and closed-loop responses quickly loose gain margin beyond the controller bandwidth frequency of 10 Hz. Another relevant remark is regarding the fact that the Closed Loop response remains at zero dB for lower frequencies, highlighting the fact that there could be effective disturbance rejection at lower frequencies.

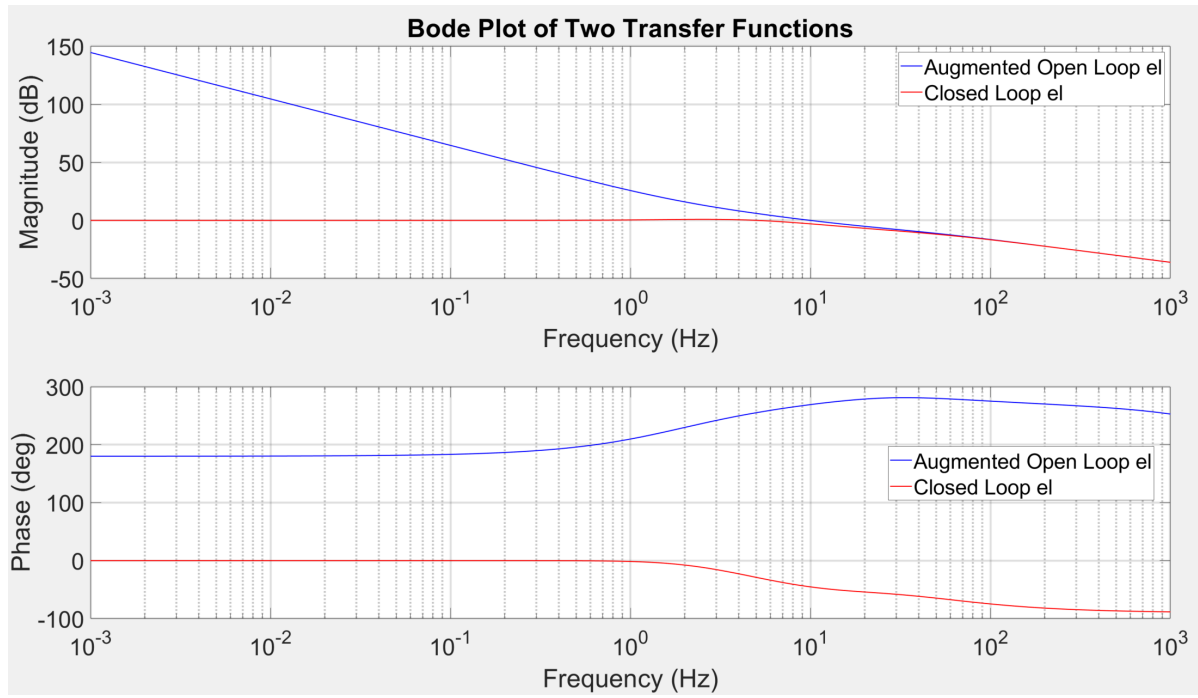


Figure E.5: Augmented Open loop and Closed loop frequency responses of the CPAS

E.2.4. Augmented open loop and Loop sensitivity response

In the previous sub-section, it was possible to highlight at higher frequencies the capacity of the system to reject noise. To further expand this the frequency response Loop Sensitivity transfer functions, provide additional information regarding the performance of the system. The loop sensitivity function of the CPAS is fully described by Equation E.19.

$$(L.S.)_{cpas} = \frac{1}{1 + C_{gimbal} * G_{gimbal} * P_{qpd}} \quad (E.19)$$

Now with the Loop Sensitivity function defined, it is then possible to map the frequency responses of the system as visible in Figure E.6. **Note:** A single plot is provided since the controller response has been tuned to the same specifications for both azimuth and elevation actuator models, resulting in identical frequency responses.

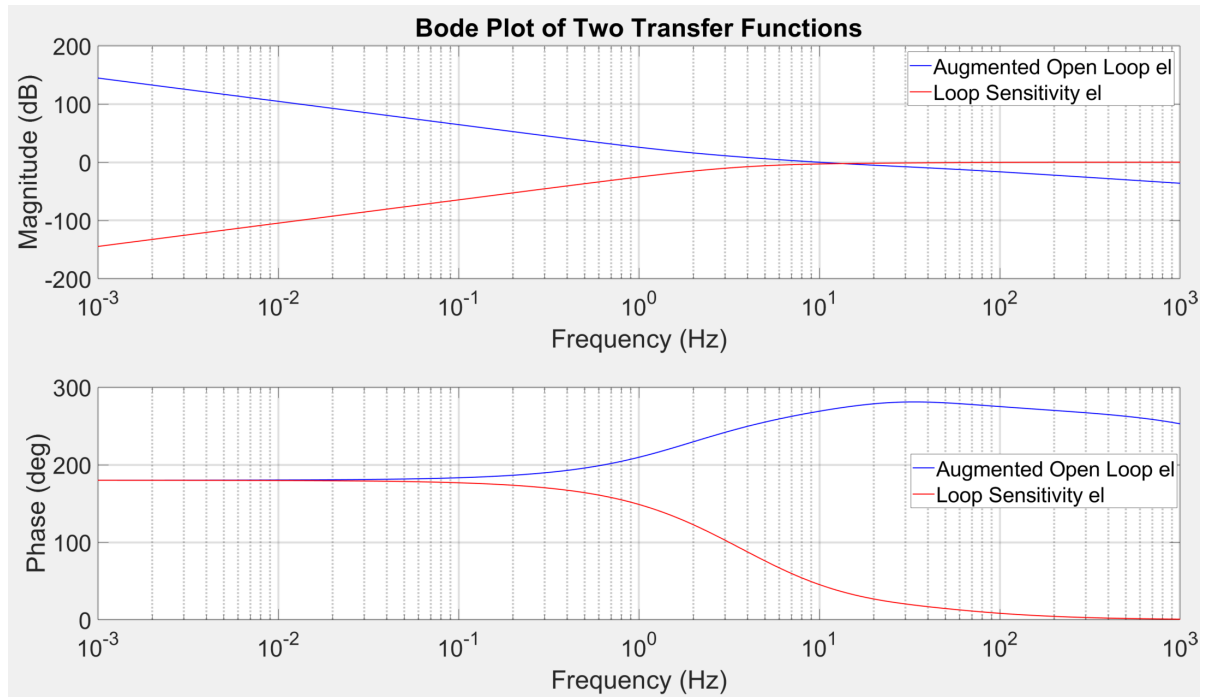
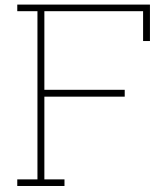


Figure E.6: Plot of the frequency response of the Loop Gain and Loop sensitivity transfer functions

From Figure E.6 it is possible to remark from the plot that the large gain margin of the augmented open loop function at low frequencies, showcases the adequate performance of the system at lower frequencies, however, the gain diminishes quickly, approaching zero at 10 Hz, around these frequencies the Loop Sensitivity function meets up with the Augment Open Loop function, indicating the performance threshold of the system, beyond this point the gain of the Augmented Open Loop function quickly decreases indicating an increasing incapacity of the controller to keep up with managing the system. After that point, the sensitivity function stays for the majority at zero dB showcasing adequate noise rejection at higher frequencies. In contrast to Figure E.2, the Loop Sensitivity function of the CPAS quickly passes from a negative gain margin to zero gain margin, which in contrast to the transition zone of the PPAS, where the sensitivity function hovers close to positive gain margin.



System identification attempt for approximating a L-350 actuator transfer function

The aim of this appendix is to provide the reader with a brief overview of the steps taken in an attempt to perform system identification of the L-350 direct drive mount transfer. The first section discusses the attempted approach, as well as the reasoning why this approach has failed to provide an accurate transfer function fully representative of the L-350 direct drive mount. The second section provides the code developed in Python, which utilizes the Planewave instruments API, for communications to the L-350 direct drive mount via a USB connection. **Note:** This appendix is a brief description of the attempts performed for system identification, it does not aim to provide extensive theoretical background information regarding the fundamental mathematics or algorithms behind this process.

F.1. Attempt to perform system identification on the Planewave L-350

The fundamental idea behind system identification is the measurement of input/output data, to characterize the dynamics of a device and create from that characterization an estimated model which approximates the dynamics found in the input/output data set.

System identification can be a complex procedure as the process of building mathematical models of dynamic systems from measured data, involves several assumptions and design choices. System identification is capable of creating Linear Time-Invariant (LTI) models, represented by transfer functions, state-space representations or differential equations. It is also capable of creating Non-linear models such as nonlinear autoregressive models (NARX), etc.

For this work it is of interest the creation of models that are compatible with the DEB analysis design approach, therefore the focus was on the creation of a linear LTI model represented as a transfer function. To achieve this two pieces of software have been utilized, the first one is the system identification toolbox application from Matlab, while the second one is the PlaneWave Interface Version 4.0, which is a Python-based API that allows for communication between a laptop and the L-350 direct drive mount.

The first step required for system identification is the creation of a data set with both input and output data. The input is classified as the desired/reference angle position, while the output is the measured angle position of the gimbal. When dealing with LTI transfer function models of the data set it is possible to increase the order of the transfer function, this has the advantage of increasing the accuracy of how the model represents the system's behavior.

However, this can also result in the model overfitting the behavior of a particular training data set. Whenever the conditions deviate from the training data set, the overfitted model could struggle to capture the behavior of the original system. As a measure against this, it is possible to create multiple data sets, where a single data set is utilized to fit a particular model, then the results of that fitted model can be compared to the results from the other data sets. To achieve this the code included in the second section of this appendix has been utilized to obtain 6 data sets, stored as Excel files.

The next step involves the system identification toolbox from Matlab. The toolbox allows to import directly from Excel files the input/output data files directly. After importing the data files, a total of 9 LTI transfer function models were fitted to the training data set, the transfer models start from a 1st order (1 pole, 0 zeros) to a 9th order (9poles, 8 zeros).

After fitting the models to the training data set, then the results were compared to the verification data sets. From these results, 2 outcomes became apparent which in-validated this approach as a sound option for representing the dynamics of the L-350 direct drive mount. The first option is that the success rate for the models was linearly increasing with respect to model order, which could indicate that the data sets were too similar to each other. Thus could be argued that the models are increasingly being overfitted into the training data, as their order increases.

While some degree of overfitting is difficult to prevent when performing system identification, what makes this approach unsuitable is the fact that due to how the communications line is set between the laptop recording the data and the L-350 gimbal following the commands, there are no direct measurements of the input/output data. Rather the suspicion is that an independent controller could be receiving the reference positions and then creating a smoothed trajectory, that the gimbal ends up following. This suspicion could be confirmed by looking into the results from the fitted models as well as the original output visible on Figure F.1. From this figure, it can be seen that the output of the system follows a smooth line trajectory which prevents sudden spikes when sudden commands are given to the system.

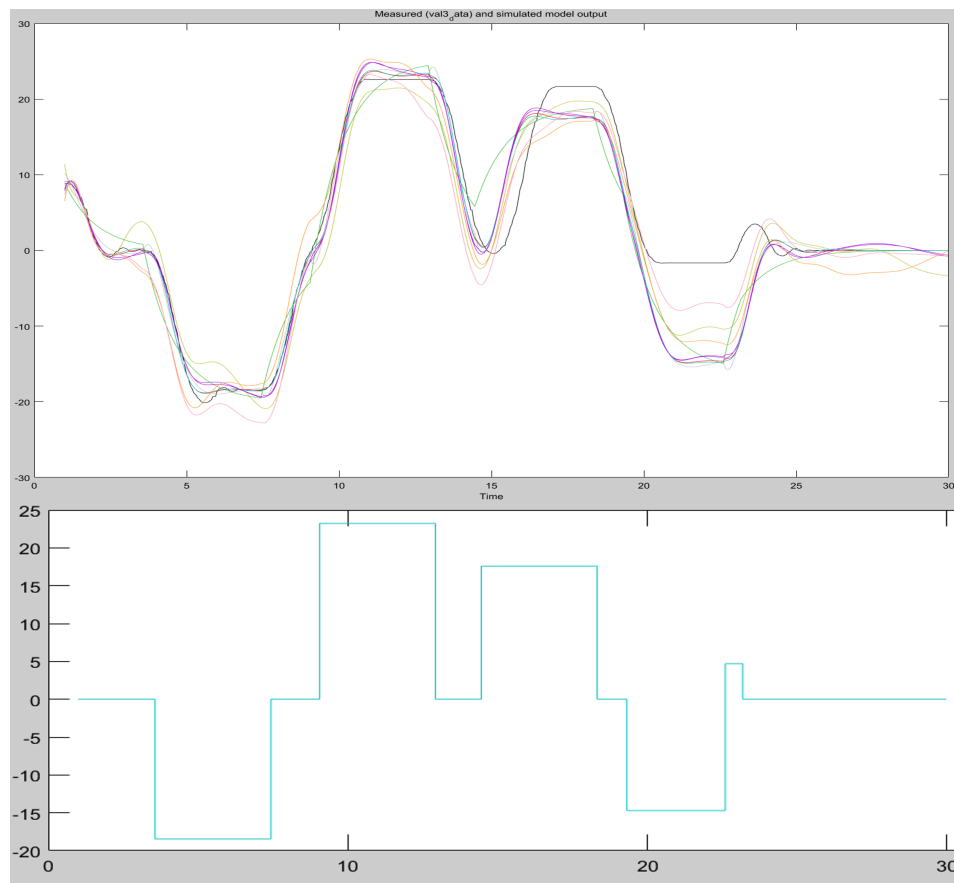


Figure F.1: Third verification test, where the lines in colour represent fitted LTI transfer function models, while the black line indicates the original output data. The bottom plot showcases the original input commands

F.2. Input/Output Planewave API code

```

1 import time
2 import matplotlib.pyplot as plt
3 import pandas as pd
4 from datetime import datetime
5 from pwi4_client import PWI4
6 import numpy as np
7 import os
8
9
10
11 # Function to generate step inputs
12 def generate_step_inputs(num_steps=5, min_amplitude=0.01, max_amplitude=25, overlap_time=3):
13     amplitudes = np.random.uniform(min_amplitude, max_amplitude, num_steps)
14     # Generate random signs (-1 or 1) to make amplitudes positive or negative
15     signs = np.random.choice([-1, 1], num_steps)
16     # Apply the signs to the amplitudes
17     amplitudes *= signs
18     times = np.zeros(num_steps)
19
20     # Generate the first time
21     times[0] = np.random.uniform(0, overlap_time)
22
23     # Generate subsequent times with the required overlap
24     for i in range(1, num_steps):
25         start_time = times[i - 1] + overlap_time
26         times[i] = np.random.uniform(start_time, start_time + overlap_time)
27
28     # Round amplitudes and times to two decimal places
29     amplitudes = np.round(amplitudes, 2)

```

```

30     times = np.round(times, 1)
31
32     return amplitudes, times
33
34
35 # Generate the step inputs
36 amplitudes, times = generate_step_inputs()
37
38 # assigning amplitudes and times to individual variables
39
40 time_for_step_input_1, time_for_step_input_2, time_for_step_input_3, time_for_step_input_4,
    time_for_step_input_5 = times
41 step_degrees_1, step_degrees_2, step_degrees_3, step_degrees_4, step_degrees_5 = amplitudes
42
43
44 # Connect to the telescope mount
45 host = "localhost" # Change if necessary
46 port = 8220
47 pwi4 = PWI4(host, port)
48
49 # Attempt to connect to the mount
50 try:
51     pwi4.mount_connect()
52     print("Successfully connected to the mount.")
53 except Exception as e:
54     print(f"Error connecting to the mount: {e}")
55     exit(1)
56 # attempt to enable the motors
57 try:
58     pwi4.mount_enable(0)
59     pwi4.mount_enable(1)
60     print("Successfully enabled the mount motors.")
61 except Exception as e:
62     print(f"Error enabling the mount motors: {e}")
63     exit(1)
64
65 # Step 1: Move to the starting position (user-specified)
66 start_azimuth = 135 # Example azimuth in degrees
67 start_altitude = 45 # Example altitude in degrees
68
69 print(f"Moving mount to start position: Azimuth {start_azimuth}, Altitude {start_altitude}")
70 pwi4.mount_goto_alt_az(start_altitude, start_azimuth)
71
72 # Wait until the mount is in position
73 while pwi4.status().mount.is_slewing:
74     print('Moving mount to starting position')
75     time.sleep(0.1) # Check every 100 ms
76 print('Mount in starting position')
77
78 # Print the results
79 for i in range(len(amplitudes)):
80     print(f"Step {i+1}: Amplitude = {amplitudes[i]:.2f} degrees, Time = {times[i]:.2f} seconds")
81
82 # Step 2: Record azimuth, altitude, and time for 30 seconds at 20Hz
83 recording_duration = 30 # in seconds
84 sampling_frequency = 20 # in Hz
85 sampling_interval = 1 / sampling_frequency
86 nominal_interval_duration = 0.038 # in seconds
87 step_response_delay = 4 # delay in seconds to return from step input to nominal value
88 # time_for_step_input_1 = 2 # in seconds
89 # time_for_step_input_2 = 12 # in seconds
90 # time_for_step_input_3 = 17 # in seconds
91 # time_for_step_input_4 = 34 # in seconds
92 # time_for_step_input_5 = 41 # in seconds
93 # step_degrees_1 = 0.01 # First step input in degrees
94 # step_degrees_2 = 5 # First step input in degrees
95 # step_degrees_3 = 1 # First step input in degrees
96 # step_degrees_4 = 0.1 # First step input in degrees
97 # step_degrees_5 = 4 # First step input in degrees
98

```

```

99 time_vector = []
100 input_vector = []
101 azimuth_vector = []
102 altitude_vector = []
103 dynamic_delay_vector = []
104
105 current_input = 0
106 start_time = time.time()
107
108 while time.time() - start_time < recording_duration:
109     start_loop = time.time()
110     current_time = time.time() - start_time
111     status = pwi4.status()
112
113     time_vector.append(current_time)
114     azimuth_vector.append(status.mount.azimuth_degs)
115     altitude_vector.append(status.mount.altitude_degs)
116     input_vector.append(current_input)
117     # introduction of 1st step input response
118     difference_for_step_input_1 = abs(current_time - time_for_step_input_1)
119     if difference_for_step_input_1 <= 0.025: # Apply step input at the desired step input
120         time
121         print(f"Applying step input of {step_degrees_1} degrees to azimuth and altitude")
122         pwi4.mount_goto_alt_az(status.mount.altitude_degs + step_degrees_1, status.mount.
123             azimuth_degs + step_degrees_1)
124         current_input = step_degrees_1
125     # return to nominal position
126     difference_for_step_input_1 = abs(current_time - (time_for_step_input_1 +
127         step_response_delay))
128     if difference_for_step_input_1 <= 0.025: # return from step input at desired step input
129         time + step response delay
130         print(f"Returning from step input of {step_degrees_1} degrees to azimuth and altitude")
131         pwi4.mount_goto_alt_az(start_altitude, start_azimuth)
132         current_input = 0
133
134     # introduction of 2nd step input response
135     difference_for_step_input_2 = abs(current_time - time_for_step_input_2)
136     if difference_for_step_input_2 <= 0.025: # Apply step input at the desired step input
137         time
138         print(f"Applying step input of {step_degrees_2} degrees to azimuth and altitude")
139         pwi4.mount_goto_alt_az(status.mount.altitude_degs + step_degrees_2, status.mount.
140             azimuth_degs + step_degrees_2)
141         current_input = step_degrees_2
142     # return to nominal position
143     difference_for_step_input_2 = abs(current_time - (time_for_step_input_2 +
144         step_response_delay))
145     if difference_for_step_input_2 <= 0.025: # return from step input at desired step input
146         time + step response delay
147         print(f"Returning from step input of {step_degrees_2} degrees to azimuth and altitude")
148         pwi4.mount_goto_alt_az(start_altitude, start_azimuth)
149         current_input = 0
150
151     # introduction of 3rd step input response
152     difference_for_step_input_3 = abs(current_time - time_for_step_input_3)
153     if difference_for_step_input_3 <= 0.025: # Apply step input at the desired step input
154         time
155         print(f"Applying step input of {step_degrees_3} degrees to azimuth and altitude")
156         pwi4.mount_goto_alt_az(status.mount.altitude_degs + step_degrees_3, status.mount.
157             azimuth_degs + step_degrees_3)
158         current_input = step_degrees_3
159     # return to nominal position
160     difference_for_step_input_3 = abs(current_time - (time_for_step_input_3 +
161         step_response_delay))
162     if difference_for_step_input_3 <= 0.025: # return from step input at desired step input
163         time + step response delay
164         print(f"Returning from step input of {step_degrees_3} degrees to azimuth and altitude")
165         pwi4.mount_goto_alt_az(start_altitude, start_azimuth)
166         current_input = 0

```

```

155
156 # introduction of 4th step input response
157 difference_for_step_input_4 = abs(current_time - time_for_step_input_4)
158 if difference_for_step_input_4 <= 0.025: # Apply step input at the desired step input
159     time
160     print(f"Applying step input of {step_degrees_4} degrees to azimuth and altitude")
161     pwi4.mount_goto_alt_az(status.mount.altitude_degs + step_degrees_4, status.mount.
162         azimuth_degs + step_degrees_4)
163     current_input = step_degrees_4
164 # return to nominal position
165 difference_for_step_input_4 = abs(current_time - (time_for_step_input_4 +
166     step_response_delay))
167 if difference_for_step_input_4 <= 0.025: # return from step input at desired step input
168     time + step_response_delay
169     print(f"Returning from step input of {step_degrees_4} degrees to azimuth and altitude")
170     pwi4.mount_goto_alt_az(start_altitude, start_azimuth)
171     current_input = 0
172
173 # introduction of 5th step input response
174 difference_for_step_input_5 = abs(current_time - time_for_step_input_5)
175 if difference_for_step_input_5 <= 0.025: # Apply step input at the desired step input
176     time
177     print(f"Applying step input of {step_degrees_5} degrees to azimuth and altitude")
178     pwi4.mount_goto_alt_az(status.mount.altitude_degs + step_degrees_5, status.mount.
179         azimuth_degs + step_degrees_5)
180     current_input = step_degrees_5
181 # return to nominal position
182 difference_for_step_input_5 = abs(current_time - (time_for_step_input_5 +
183     step_response_delay))
184 if difference_for_step_input_5 <= 0.025: # return from step input at desired step input
185     time + step_response_delay
186     print(f"Returning from step input of {step_degrees_5} degrees to azimuth and altitude")
187     pwi4.mount_goto_alt_az(start_altitude, start_azimuth)
188     current_input = 0
189
190 # dynamic sampling interval delay for constant sampling rate
191 dynamic_delay = nominal_interval_duration - (time.time() - start_loop)
192 dynamic_delay_vector.append(dynamic_delay)
193 time.sleep(dynamic_delay)
194
195 # Step 3: Plot the recorded data
196 plt.figure()
197 plt.plot(time_vector, azimuth_vector, label='Azimuth')
198 plt.plot(time_vector, altitude_vector, label='Altitude')
199 plt.xlabel("Time(s)")
200 plt.ylabel("Degrees")
201 plt.title("Mount Position Over Time")
202 plt.legend()
203 plt.show()
204
205 # Step 4: Save the data to an Excel file
206 # Filename includes the test number and step input value
207 test_number = 4 # Change as needed
208 excel_filename = f"test_{test_number}_steps_{step_degrees_1,step_degrees_2}.xlsx"
209
210 # Create a DataFrame to store the data
211 data = {
212     "Time": time_vector,
213     "Input_vector": input_vector,
214     "Azimuth": azimuth_vector,
215     "Altitude": altitude_vector,
216     "Dynamic_delay": dynamic_delay_vector
217 }
218
219 # data frame
220 df = pd.DataFrame(data)
221
222 # Define the directory and file name
223 directory = r"C:\Users\mateo\Documents\TU_Space_master\Thesis\matlab_files\Gymbal_simulation\
224     test_data_p_135az_45alt\3_seconds_overlap_25_max_ampli"

```

```
215 file_path = os.path.join(directory, excel_filename)
216
217 # Save the DataFrame to an Excel file
218 df.to_excel(file_path, index=False)
219
220
221 print(f"Data recorded and saved to {excel_filename}")
```



Dynamic Budget Error Analysis supplementary information

G.1. Brief introduction to Dynamic Error Budgeting analysis

This section aims to provide a brief background into the DEB analysis design approach, it is based on the master thesis of W. Monkhurst [36]. The objective is to provide the main assumptions and analytical equations utilized to perform this type of analysis, for more in-depth information readers are referred to the cited work.

DEB analysis has been chosen based on the fact that it allows for the determination of the possible achievable performance of a system, in the early stages of the design process. During the early design stages often initial models of the actuators can be determined and paired with the main sources of disturbances. The main benefit is that this type of analysis is compatible with signals that are of stochastic nature, such as the ones introduced by Ship-Wave motions, Sensor tracking noise, analog-to-digital converters, etc. The main assumptions required for DEB analysis include the following:

- The system for which the analysis is to be performed has to be fully modeled as a Linear Time Invariant system. Non-linear behavior can be added as an additional source of disturbances in the analysis.
- Disturbances acting on the system must be stationary (i.e. their statistical properties do not change over time).
- Disturbances must be uncorrelated to each other.
- Disturbance sources must have a defined PSD, therefore only stochastic disturbances can be utilized. However, no assumption is made regarding the distribution of the distribution functions of the disturbances.

The fundamental idea of DEB analysis is to model how disturbances in different sections of a Closed Loop system affect the end performance of the system. Instrumental to this, is the correct modeling of the propagation of disturbances that act on the Closed Loop system. As visible in Figure G.1, the sources of disturbances can occur at different sections of the closed loop system. Since these disturbances act in different locations of the closed-loop system, the transfer functions to which these are applied, then have to be fitted specifically to the section to which these are applied.

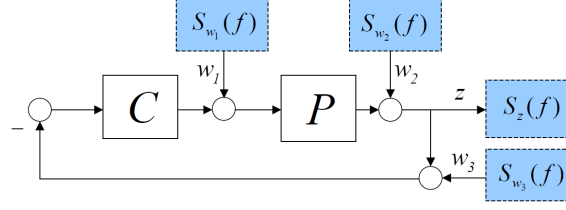


Figure G.1: Schematic of a closed loop system with various disturbances acting in it [36]. Note the blue-dashed boxes correspond to the PSD of a signal.

As described earlier in the assumptions section, the disturbance signal needs to be modeled as a PSD, as this allows for the determination of the end closed system performance.

The fundamental idea to perform this type of analysis is to fit the Loop Sensitivity function to the location where the PSD disturbance signal is being applied. Since the Loop Sensitivity function contains the frequency response of the system per frequency (i.e. the degree of noise suppression/transmissivity per frequency), then Equation G.1 can be utilized to determine the output performance of the closed loop system (S_{out}).

$$S_{out}(f) = |H(j2\pi f)|^2 * S_{in}(f) \quad (G.1)$$

Where $H(j2\pi f)$ is the specific loop sensitivity frequency response and $S_{in}(f)$ is the input PSD of the location-specific disturbance signal.

Next is possible to integrate across the relevant frequency range, since the PSD describes the power density over frequency [$\frac{SI^2}{Hz}$], then the integral of the output PSD equals the power of the performance signal, however since all disturbance signals are assumed to have zero mean, taking the integral equates to the variance of the output performance as shown in Equation G.2.

$$\sigma_z^2 = \int_0^\infty S_{out}(f) df \quad (G.2)$$

Where the discrete Root-Mean-Square error of the system's performance can be then defined as in Equation G.3.

$$Y_{rms} = \sigma_z = \sqrt{\int_0^{f_c} S_{out}(f) df} \quad (G.3)$$

Next is the determination of the Cumulative Power Spectrum (CPS) of the system's performance output. As described earlier in chapter 3 at the Cumulative Power Spectrum function subsection, the aim of this function is to describe how the variance in the system is stacking up across frequencies (i.e. how the build-up in errors scales across the relevant frequency range). Equation G.4 can be utilized to determine CPS of the system performance output.

$$CPS(f) = \int_0^{f_c} S_{out}(f) df \quad (G.4)$$

Where the CPS is defined from 0 Hz to the cut-off frequency (f_c), while the integral is a monotone increasing function, with its magnitude increasing from zero to the end value of $CPS(f_c) = \sigma_z^2$. This result is due to the fact that the final value of the CPS system performance value is equal to the variance of the signal.

Lastly are the equations utilized for ease of visualization/understanding purposes, since the system performance output of S_{out} is in [$\frac{SI^2}{Hz}$] and the CPS is in [SI^2]. It is simpler to plot the square root versions of these, these are called the Amplitude Spectral Density (ASD) and the Cumulative Amplitude Spectrum (CAS) and are defined as in Equation G.5 and Equation G.6.

$$ASD(f) = \sqrt{S_{out}(f)} \quad (G.5)$$

$$CAS(f) = \sqrt{CPS(f)} \quad (G.6)$$

G.2. PPAS Dynamic Error Budgeting analysis

This section aims to present the PPAS DEB analysis results, covering aspects such as DEB loop logic, loop-specific equations for actuator pointing error, controller effort and conclude with a discussion of the ASD & CAS performance analysis results.

Figure G.2 provides the PPAS DEB analysis loop logic diagram, in which the input PSD disturbances are marked with purple arrows, the core of the system Closed Loop logic is indicated by blue arrows & boxes, the controller & actuator outputs are marked by green arrows.

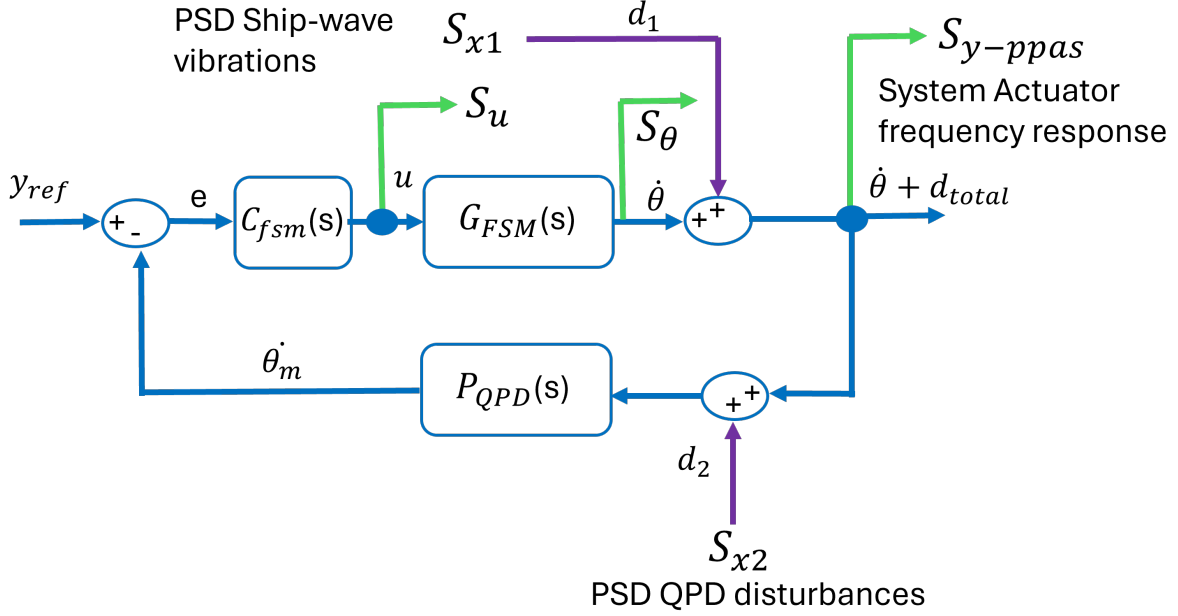


Figure G.2: PPAS DEB analysis loop logic

G.2.1. PPAS DEB equations

With an understanding of the PPAS control loop logic as well as the location at which the disturbances are entering the closed loop system, it is then possible to outline the loop-specific output performance equations, starting with the total closed loop system pointing error performance described by Equation G.10.

$$S_{Y_{PPAS}}(\omega) = |H_{ppas}(j2\pi\omega)|^2 * S_{x1}(\omega) + |H_{ppas-ts}(j2\pi\omega)|^2 * S_{x2}(\omega) \quad (G.7)$$

Where S_{x1} and S_{x2} are the input PSD of the ship-wave disturbances and QPD disturbances, respectively. While Equation G.8 and Equation G.9, describe the Loop Specific Sensitivity functions. Note that the frequency-specific response ($j2\pi\omega$) of the individual transfer functions (i.e. C_{fsm} or G_{fsm}) has been skipped to shorten the notation space.

$$H_{ppas}(j2\pi\omega) = \frac{1}{1 + C_{fsm} * G_{fsm} * P_{QPD}} \quad (G.8)$$

$$H_{ppas-ts}(j2\pi\omega) = H_{ppas} * G_{fsm} * C_{fsm} * P_{QPD} \quad (G.9)$$

Next is the determination of the controller effort DEB equations. The controller effort is measured right after the controller transfer function, which sends the control torques to the actuator. The controller effort for the PPAS can be determined utilizing Equation G.10 and Equation G.11.

$$S_{u_{PPAS}}(\omega) = |H_{ppas-u_t}(j2\pi\omega)|^2 * (S_{x1}(\omega) + S_{x2}(\omega)) \quad (G.10)$$

Where S_{x1} and S_{x2} are the input disturbances PSD are combined utilizing the assumption of superposition (applicable to linear systems and signals).

$$H_{ppas-u_t}(j2\pi\omega) = H_{ppas} * C_{fsm} * P_{QPD} \quad (G.11)$$

Note that there is a single loop-specific sensitivity response function for the DEB analysis of the PPAS, as relative to this configuration of input/output, the input disturbance PSDs enter at the same location in the closed-loop feedback path.

Lastly, the actuator effort DEB equations are determined. The actuator effort is measured right after the actuator displacement transfer function, it receives the control torques and converts them into an angular displacement. The actuator effort for the PPAS can be determined utilizing Equation G.12 and Equation G.13.

$$S_{\theta_{PPAS}}(\omega) = |H_{ppas-\theta}(j2\pi\omega)|^2 * (S_{x1}(\omega) + S_{x2}(\omega)) \quad (G.12)$$

$$H_{ppas-\theta}(j2\pi\omega) = H_{ppas} * C_{fsm} * G_{fsm} * P_{QPD} \quad (G.13)$$

Now that the PPAS DEB analysis equations have been discussed, next is the interpretation of the performance analysis results. Since the IMU data set provides information on both recorded angles and angle rates, the following two subsections center on discussing the individual results for each of the two measurements. Note the code utilized to perform the numerical analysis can be found in Appendix H.

G.2.2. PPAS DEB pointing angle error, controller torque and actuator angular effort results

This section centers on discussing the PPAS DEB analysis performance results with respect to the angle disturbances of the ship-wave motions data set.

Visible in Figure G.3 is the distribution of how the disturbances contribute to the pointing error, what is notable from the ASD plot is that the main contribution of the Ship-Wave disturbances occurs at lower frequencies ($\approx 0.1 : Hz$), this can be further confirmed by the CAS plot in which the cumulative gain in error, for the Ship-Wave disturbances, all display a sharply increasing phenomena before reaching this frequency. After that, the cumulative sum of errors quickly stagnates afterwards, until reaching frequencies close to the cutoff frequency. The cumulative sum of tracking errors is steadily increasing, starting at a low value at lower frequencies and primarily contributing to the pointing errors at higher frequencies.

Notable from the outcomes presented in Table G.1, is the total RMS error of $3\sigma \approx 37.62 \mu\text{rads}$ which would be sufficient to comply with the pointing error requirement PATR-01. Note that 3σ is taken as it "gives a good approximation of the maximum amplitude of a stochastic signal" (p14,[36]).

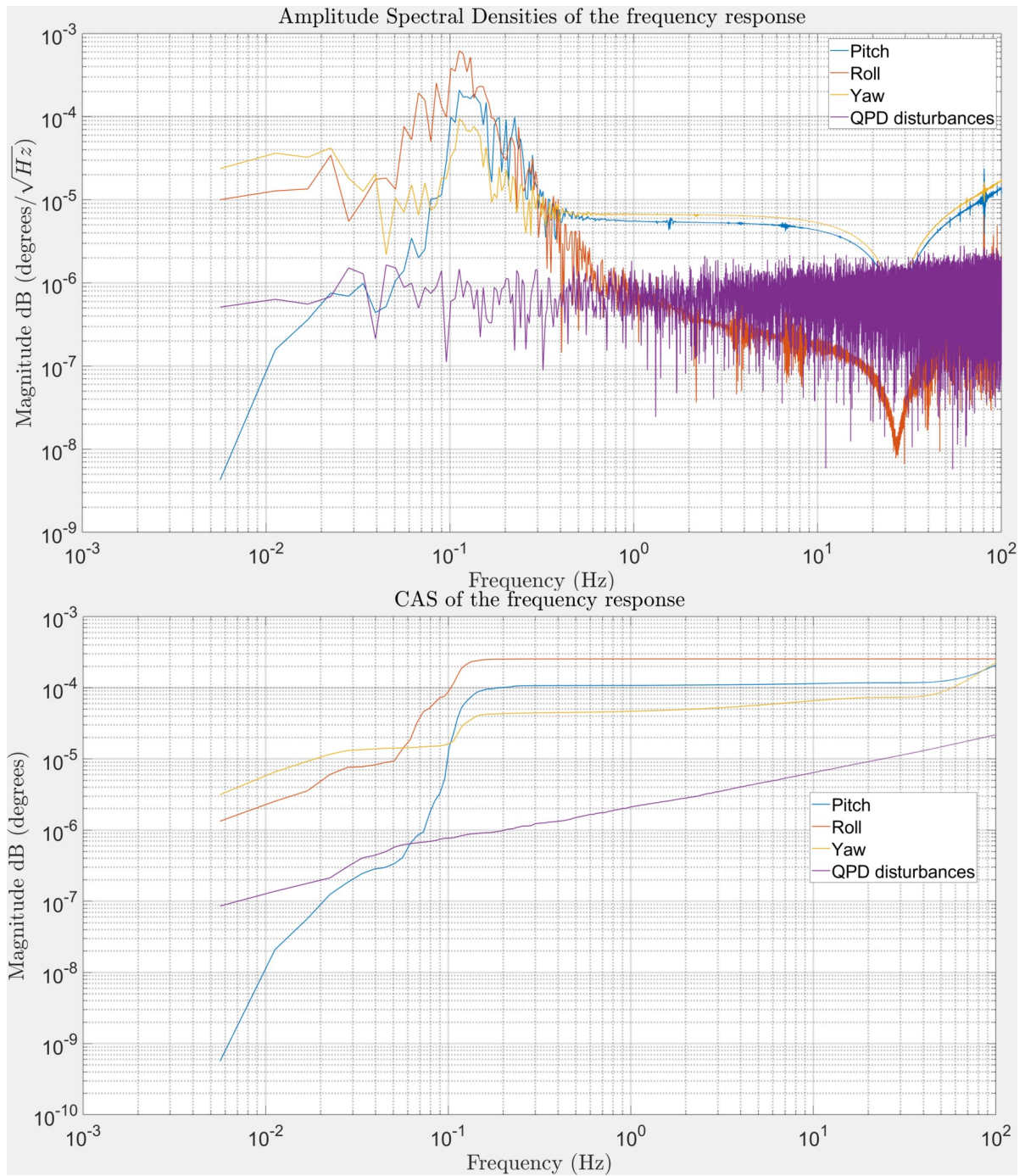


Figure G.3: DEB analysis results of the PPAS system pointing angle error ASD and CAS

Table G.1: PPAS total system pointing error DEB analysis performance results

RMS error contribution	Unit [$\mu\text{degrees}$]	Unit [μrads]	Unit [Arcsec]
Pitch Angle	212.078	3.701	0.763
Roll Angle	254.452	4.441	0.916
Yaw Angle	230.109	4.016	0.828
QPD sensor	21.972	0.383	0.079
Total RMS error (1σ)	718.612	12.542	2.587

Notable from the ASD plot in Figure G.4 is the fact that the contribution to the controller torque demands occurs primarily at lower frequencies. This could be driven by the fact that the disturbance spectra of the ship-wave interactions have a higher amplitude at lower frequencies. This is also visible from the CAS plot, in which the disturbance torque requirements of the ship-wave disturbance flat out beyond $\approx 2 \text{ Hz}$. Notably, the controller efforts required by the QPD-tracking sensor error continue to increase, this could be the result of the oscillatory behavior increasing torque demands at higher frequencies, this could potentially be mitigated by lowering the bandwidth of the controller, since potentially at frequencies beyond the sampling rate, it appears like it could become a dominant source of torque requirements.

From Table G.2, it is possible to conclude that RMS torque is of only $3 \sigma \approx 14.094 \text{ mN} * m$, which should be a manageable torque effort for the precise actuator i.e. FSM, albeit the precise torque specifications are not publicly available in [38], as to fully confirm this.

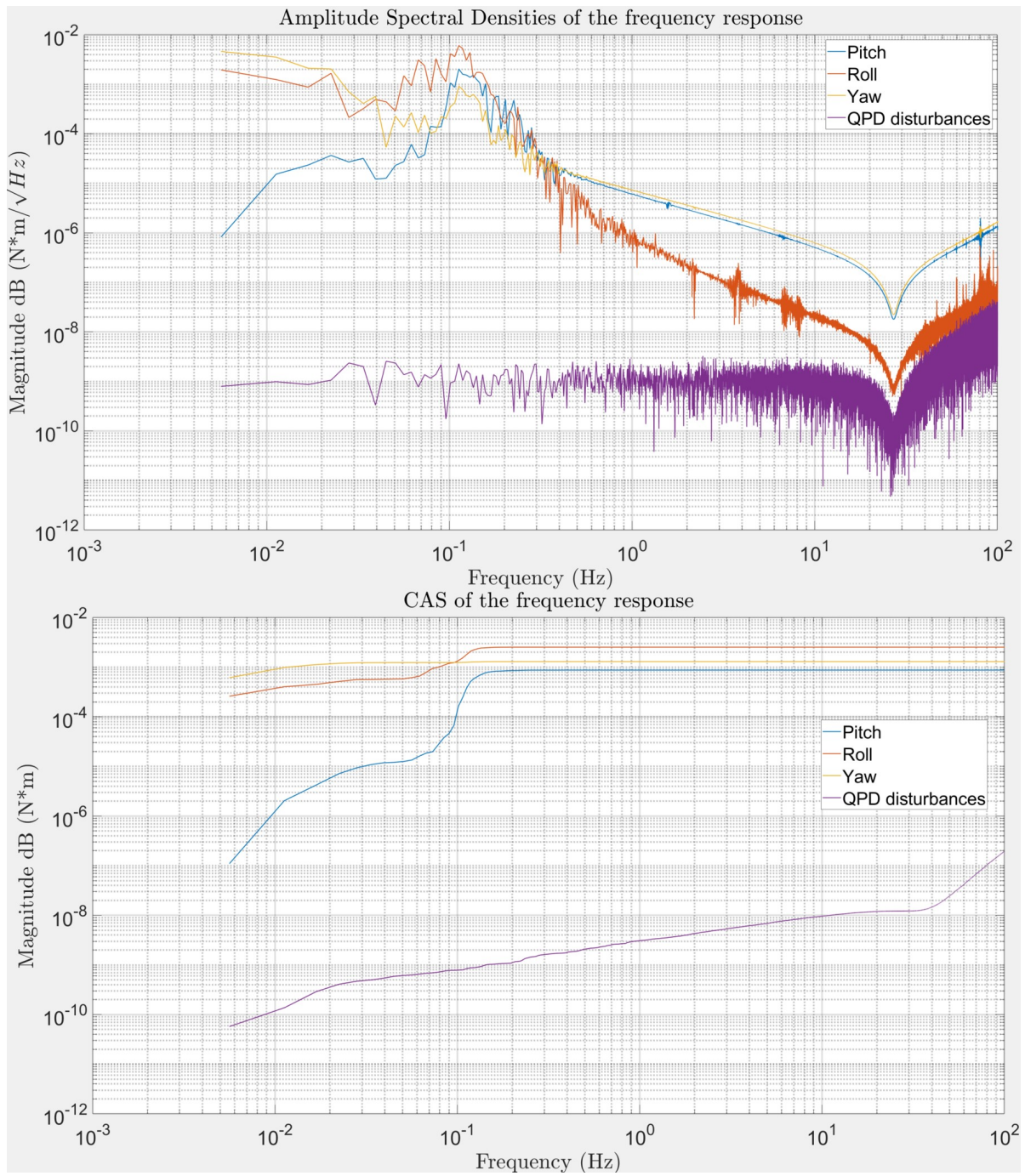


Figure G.4: DEB analysis results of the PPAS controller torque ASD and CAS

Table G.2: PPAS Controller torque DEB analysis performance results

RMS torque contribution	Unit [$mN \cdot m$]
Pitch Angle	0.877
Roll Angle	2.565
Yaw Angle	1.293
QPD sensor	0.197×10^{-3}
Total RMS torque (1σ)	4.698

Notable from Figure G.5, is the fact that from the ASD plot, it is possible to recognize that the disturbances at frequencies below 1 Hz contribute significantly to the required actuator effort. This could be motivated based on the fact that it would be likely to expect that high-frequency disturbance spectra occurring in a frigate, would have a minor contribution to angular displacement requirements, albeit still playing a dominant role with respect to pointing errors, as seen at the CAS plot when focussing on the contributions at the highest frequencies in Figure G.3.

From Table G.3, it is possible to determine the total angular RMS effort of $3\sigma \approx 9.06\text{ degrees}$, which would be beyond the achievable displacement range of the FSM, which tops-out at 3 degrees [38]. **Note:** It is being assumed that it is possible to directly add up the angular contributions, to obtain an initial estimate of the total angular displacement range. It would be advisable for future research to verify such results via a time-based simulation, as this falls outside the scope of work of the current research.

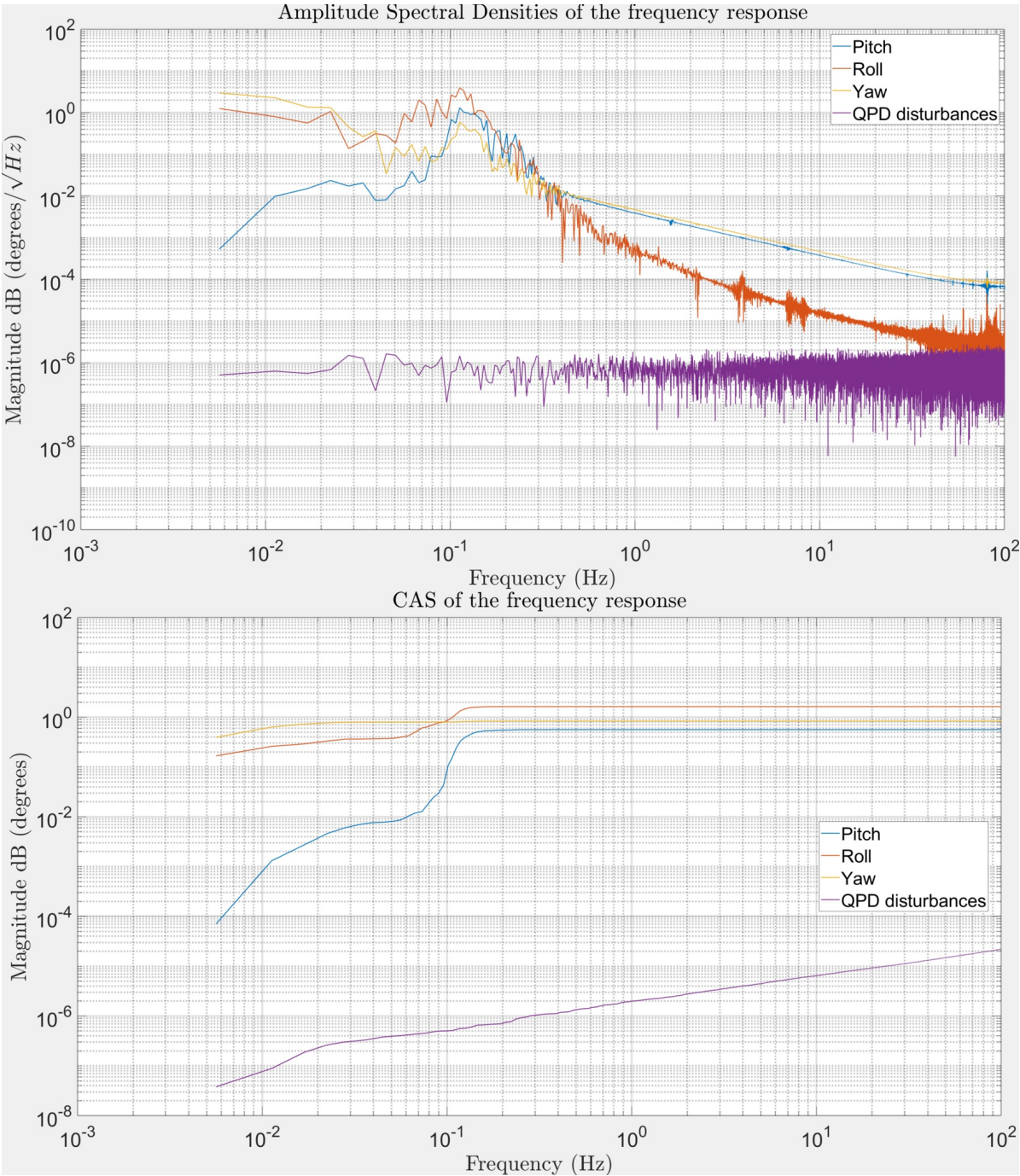


Figure G.5: DEB analysis results of the PPAS actuator effort for angular range ASD and CAS

Table G.3: PPAS Actuator effort angular range DEB analysis performance results

RMs error contribution	Unit [<i>mdegrees</i>]	Unit [<i>mrads</i>]	Unit [<i>Arcsec</i>]
Pitch Angle	560.494	9.782	2017.778
Roll Angle	1628.235	28.418	5861.647
Yaw Angle	831.664	14.515	2993.990
QPD sensor	0.021	0.382×10^{-3}	0.078
Total RMS error (1σ)	3020.415	52.716	10873.494

G.2.3. PPAS DEB pointing angle-rate error, controller power and actuator angular-rate effort results

Next, are the DEB analysis results of the angle rate pointing error of the actuator, the controller supply power, and actuator angle rate effort (slew-rate).

From the ASD plot shown in Figure G.6, it is possible to recognize that at higher frequencies, the disturbances introduced by tracking errors start compiling, playing an ever more dominant role at higher frequencies. Since the bandwidth of the PPAS controller is set higher than the cut-off frequency of the DEB analysis, it is not yet possible to see the attenuation introduced by limiting the controller output, however, this remains an alternative to decrease the compiling of pointing angle-rate errors. From the CAS plot, it is possible to argue that even as the tracking sensor error disturbances are dominant at higher frequencies, the initial contributions of the ship-wave disturbances at lower frequencies are still the main contributing factors to the overall pointing angle rate error.

From the results summarized on Table G.4, it is possible to see that the yaw-angle rate has the highest contribution to the pointing rate error. The total pointing rate RMS error approximates $3\sigma \approx 28.209 \mu\text{rads}$, which would be compliant with the **PATR-01** pointing requirement.

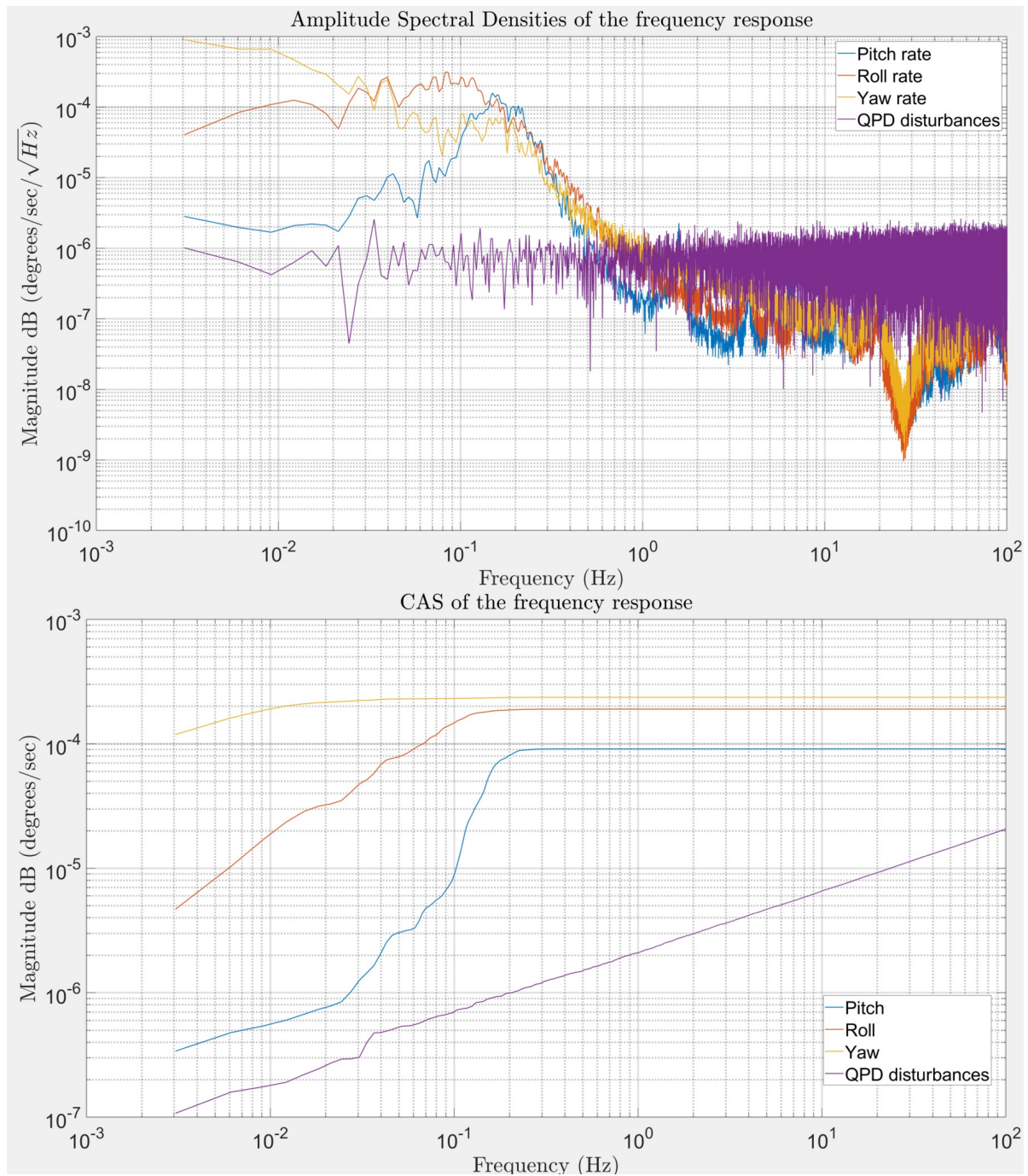


Figure G.6: DEB analysis results of the PPAS pointing angle rate error ASD and CAS

Table G.4: PPAS Pointing Angle Rate error DEB analysis performance results

RMS error contribution	Unit [$\mu\text{degrees/sec}$]	Unit [$\mu\text{rads/sec}$]	Unit [Arcsec/sec]
Pitch Angle-rate	91.144	1.590	0.328
Roll Angle-rate	190.210	3.319	0.684
Yaw Angle-rate	236.743	4.132	0.852
QPD sensor-rate	20.669	0.360	0.074
Total RMS error (1σ)	538.768	9.403	1.939

It is possible to recognize from the ASD plot found in Figure G.7, that the controller power draw is the highest at lower frequencies and sequentially reducing significantly at higher frequencies. From the summary of results visible on Table G.5 it is possible to note that once again the yaw rate disturbances have a dominant effect on the power draw, being almost an order of magnitude larger than the second main source of power draw.

It can be argued that the total RMS power draw of $3\sigma \approx 165.72 \text{ mW}$ should remain manageable even for a small actuator such as the one of the FSM. The relatively low power demands could be explained by the precise angular rate corrections, occurring at higher frequencies, which could require less power, since they appear to be of smaller displacement amplitude, thus helping to explain the relative stagnation at higher frequencies seen in the CPS plot of Figure G.4.

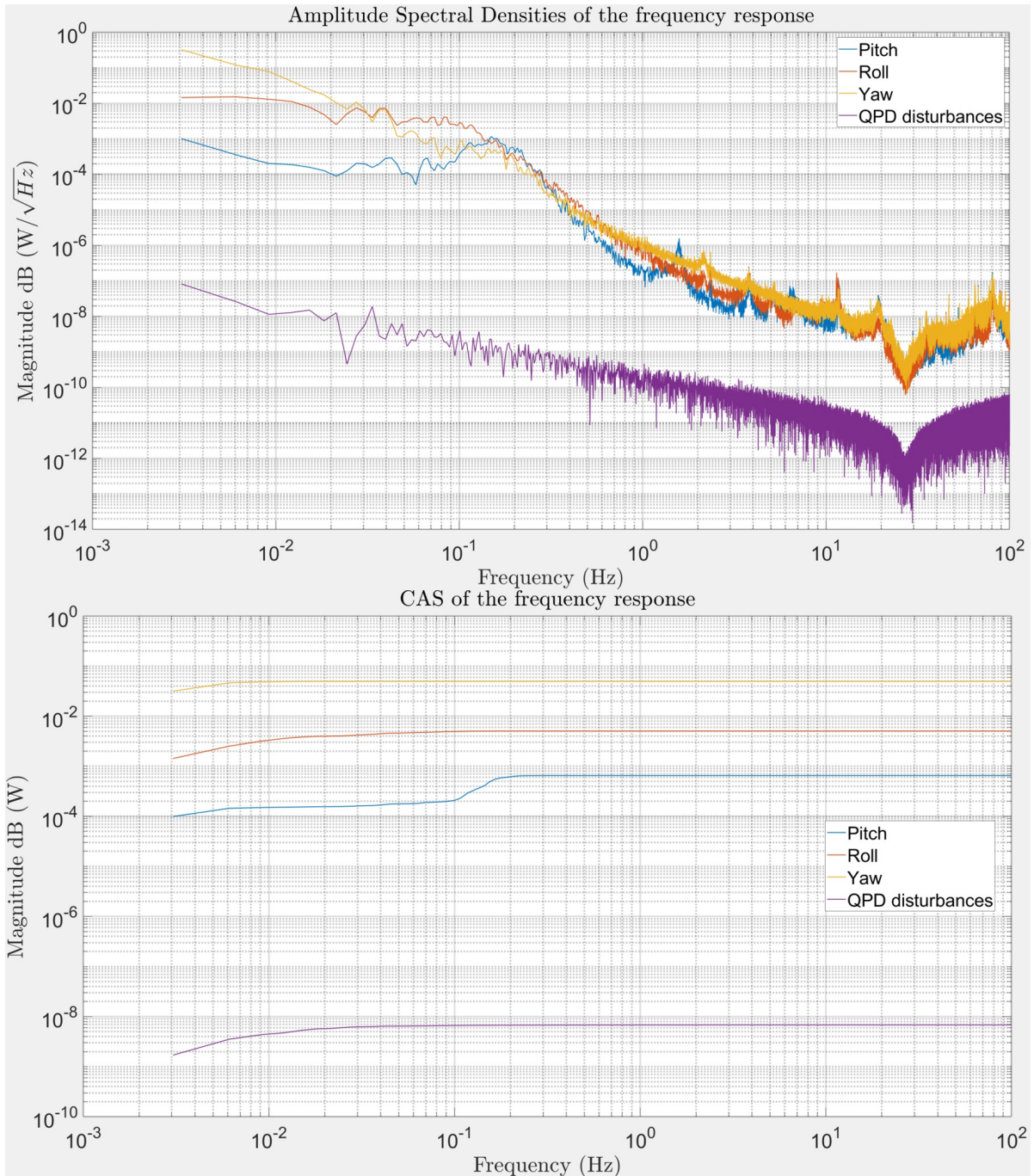


Figure G.7: DEB analysis results of the PPAS controller power ASD and CAS

Table G.5: PPAS Controller Power DEB analysis performance results

RMS power contribution	Unit [mW]
Pitch Angle	0.651
Roll Angle	5.057
Yaw Angle	49.532
QPD sensor	5.3×10^{-6}
Total RMS power (1σ)	55.24

Visible from the CAS plot in Figure G.8, is the fact that the angle-rate effort tops out quickly after reaching $\approx 0.2 \text{ Hz}$, notable from these results is the fact that for both actuator angle and angle rate effort, the main contributions appear to occur at lower frequencies.

From the results summarized on Table G.6, it is possible to determine the total actuator pointing rate effort/slew rate, by taking the total RMS actuator rate effort it approximates a value of $3\sigma \approx 6.69 \text{ degrees/sec}$, which is certainly achievable given the allowable closed-loop bandwidth of the FSM actuator.

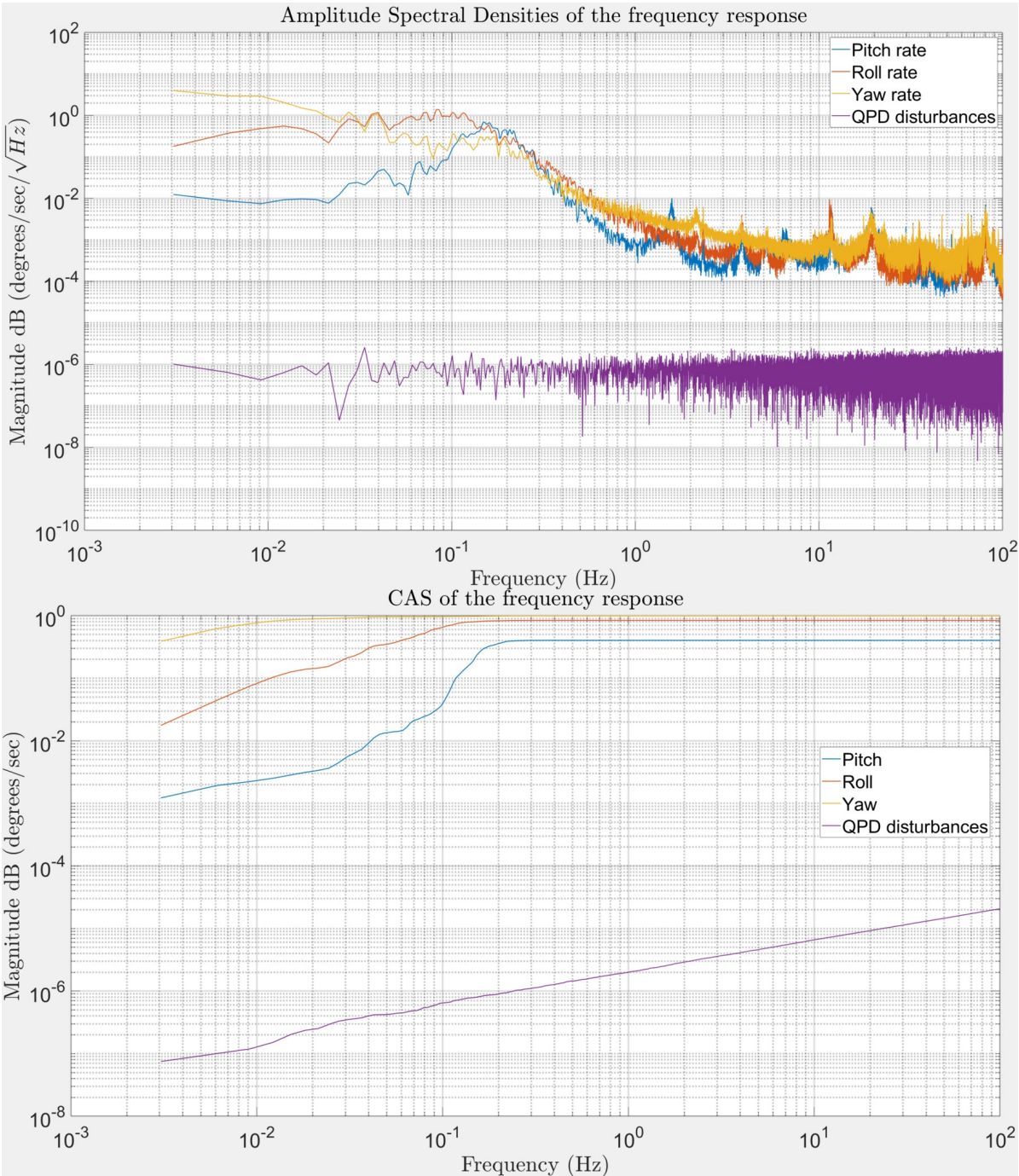


Figure G.8: DEB analysis results of the PPAS actuator angle rate effort ASD and CAS

Table G.6: PPAS Actuator Angle Rate effort DEB analysis performance results

RMs error contribution	Unit [<i>mdegrees/sec</i>]	Unit [<i>mrads/sec</i>]	Unit [<i>Arcsec/sec</i>]
Pitch Angle-rate	403.270	7.038	1451.773
Roll Angle-rate	841.473	14.686	3029.305
Yaw Angle-rate	986.142	17.211	3550.112
QPD sensor-rate	0.020	0.362×10^{-3}	0.074
Total RMS error (1σ)	2230.907	38.936	8031.266

G.3. CPAS Dynamic Error Budgeting analysis

The next section in this chapter is the CPAS DEB analysis. This section aims to cover the CPAS DEB loop logic, derive the output performance DEB equations including the loop specific sensitivity functions and conclude with a discussion of the performance results.

The section starts by discussing the CPAS DEB loop logic, which is clearly outlined on Figure G.9. From the figure it is possible to see that the purple arrows introduce the ship-wave PSD's and the QPD PSD disturbances at the forward and feedback paths, respectively. While the output of performance is measured at three points by the green arrows, these being the total system output measuring pointing errors at the end of the forward path, the controller system performance right after the controller transfer function, lastly the actuator effort is measured right after the actuator transfer function.

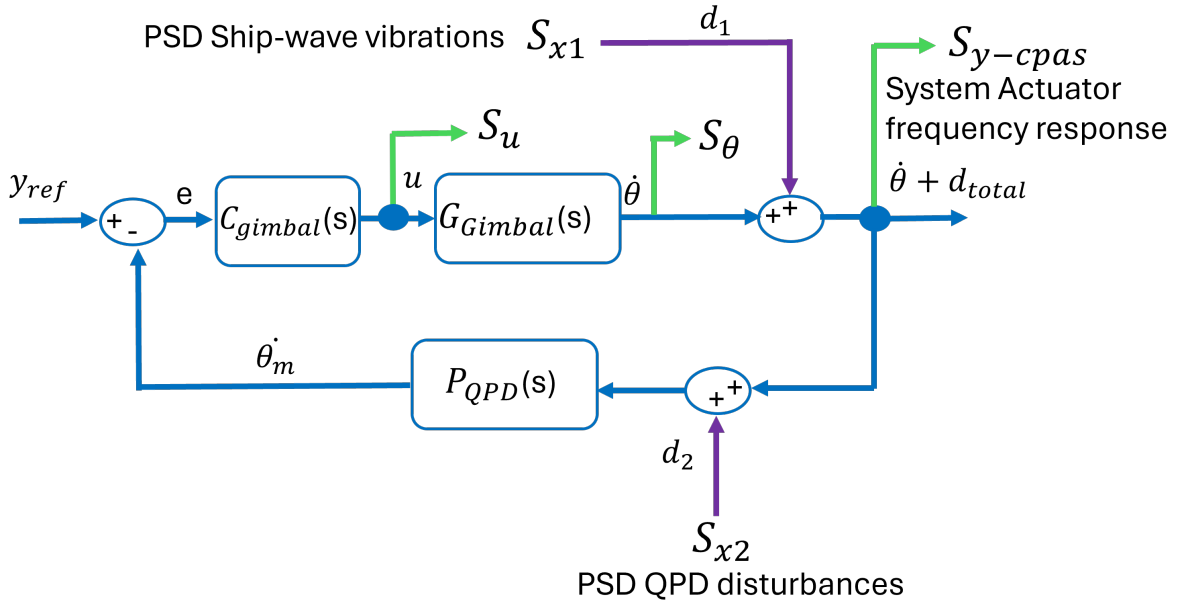


Figure G.9: CPAS DEB analysis loop logic

G.3.1. CPAS DEB equations

With the DEB loop logic of the CPAS discussed, next is the discussion of the DEB system performance and loop specific sensitivity functions. It is then possible to start describing the DEB actuator total output performance system, described by Equation G.14. What can be seen from this equation is that there are two loop specific sensitivity functions, which correspond to the two specific places at which the Ship-Wave and QPD disturbances PSD are introduced into the closed loop.

The first loop specific sensitivity function corresponds to the complete sensitivity of the total closed-loop performance and is fully described by Equation G.15. The second loop specific sensitive function corresponds to the sensitivity function relative to the location where the QPD disturbances are introduced, as the actuator output is located at the forward path and is required to include the transfer functions located at the end of the forward path, to then proceed backward until reaching the location at which the QPD disturbances enter the closed loop system, this is described on Equation G.16.

$$S_{Y_{CPAS}}(\omega) = |H_{cpas}(j2\pi\omega)|^2 * S_{x1}(\omega) + |H_{cpas-ts}(j2\pi\omega)|^2 * S_{x2}(\omega) \quad (G.14)$$

$$H_{cpas}(j2\pi\omega) = \frac{1}{1 + C_{gimbal} * G_{gimbal} * P_{QPD}} \quad (G.15)$$

$$H_{cpas-ts}(j2\pi\omega) = H_{cpas} * G_{gimbal} * C_{gimbal} * P_{QPD} \quad (G.16)$$

With the CPAS DEB output performance equations of the actuator having been discussed, next is to discuss the output performance equations of the CPAS controller, which are utilized to determine the controller effort. For this configuration a single DEB output performance and a single loop specific sensitivity function are required, this is due to the fact that relative to the output of the controller, the locations where the PSDs of the Ship-Wave and sensor disturbances are introduced, they both pass through the same closed loop path. The total controller performance output is described by Equation G.17, while the controller loop sensitivity function is covered by Equation G.18.

$$S_{u_{CPAS}}(\omega) = |H_{cpas-u_t}(j2\pi\omega)|^2 * (S_{x1}(\omega) + S_{x2}(\omega)) \quad (G.17)$$

$$H_{cpas-u_t}(j2\pi\omega) = H_{cpas} * C_{gimbal} * P_{QPD} \quad (G.18)$$

The last DEB equations left to discuss are the output performance equations of the actuator effort. The actuator effort performance output is described by Equation G.19, while the actuator loop sensitivity function is covered by Equation G.20.

$$S_{\theta_{CPAS}}(\omega) = |H_{cpas-\theta}(j2\pi\omega)|^2 * (S_{x1}(\omega) + S_{x2}(\omega)) \quad (G.19)$$

$$H_{cpas-\theta}(j2\pi\omega) = H_{cpas} * C_{gimbal} * G_{gimbal} * P_{QPD} \quad (G.20)$$

G.3.2. CPAS DEB pointing angle error, controller torque and actuator angular effort results

The next is to present the results of performing the DEB analysis with respect to the angle PSD disturbances. From Figure G.10 it is possible to denote from the CAS plot, that the pointing angle errors while initially damped, they pointing angle errors start to accumulate as the frequency increases, after reaching the controller bandwidth limit, they appear to decrease the accumulation of errors, which could be the result of the system rejecting inputs occurring at frequencies much higher than the controller bandwidth.

Next are the results provided on Table G.7, which highlight a total RMS pointing error of $3 \sigma \approx 493 \mu\text{rads}$, that is clearly insufficient to comply with **PATR-01** pointing jitter requirement. Another remarkable outcome of this analysis is the fact that due to the early cut-off frequency of the gimbal controller, the tracking sensor error disturbances are minimized and their impact contribution to the overall pointing error is almost negligible when compared to the magnitude of the errors caused by the Ship-Wave disturbances.

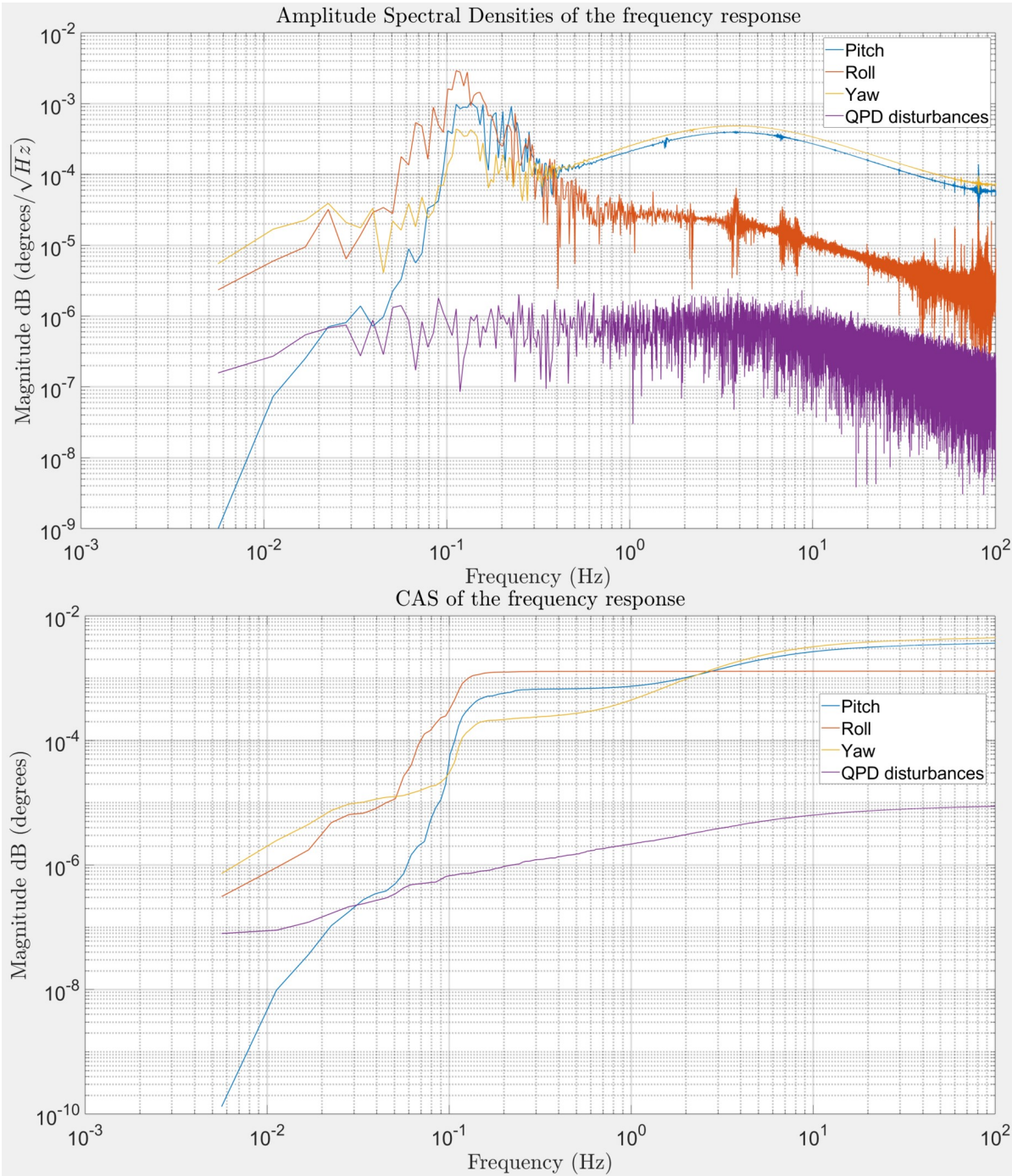


Figure G.10: DEB analysis results of the CPAS actuator pointing angle error ASD and CAS plots

Table G.7: CPAS actuator pointing angle error DEB analysis performance results

RMs error contribution	Unit [<i>mdegrees</i>]	Unit [<i>mrads</i>]	Unit [<i>Arcsec</i>]
Pitch Angle	3.649	0.063	13.139
Roll Angle	1.300	0.022	4.683
Yaw Angle	4.459	0.077	16.056
QPD sensor	0.008	0.153×10^{-3}	0.031
Total RMS error (1σ)	9.419	0.153	33.910

As visible from Figure G.11, the main contributions to the controller torque effort occur at lower frequencies, this is visible on the ASD plot, where the disturbance-induced controller torques dominate in amplitude for frequencies below 1 *Hz*. This could be the result of lower frequency disturbances being large in amplitude and due to the relatively larger damping coefficient of the CPAS (when compared to the PPAS), this combination could be the result of the relatively high torque demand at lower frequencies.

From the results shown in Table G.8, it is possible to determine that the total RMS torque is of $3\sigma \approx 7.225 \text{ N} \cdot \text{m}$, which is ≈ 3.8 lower than the official maximum continuous torque rating of $27 \text{ N} \cdot \text{m}$ of the L-350 direct drive gimbal [48]. Thus fully capable of fulfilling these torque effort requirements.

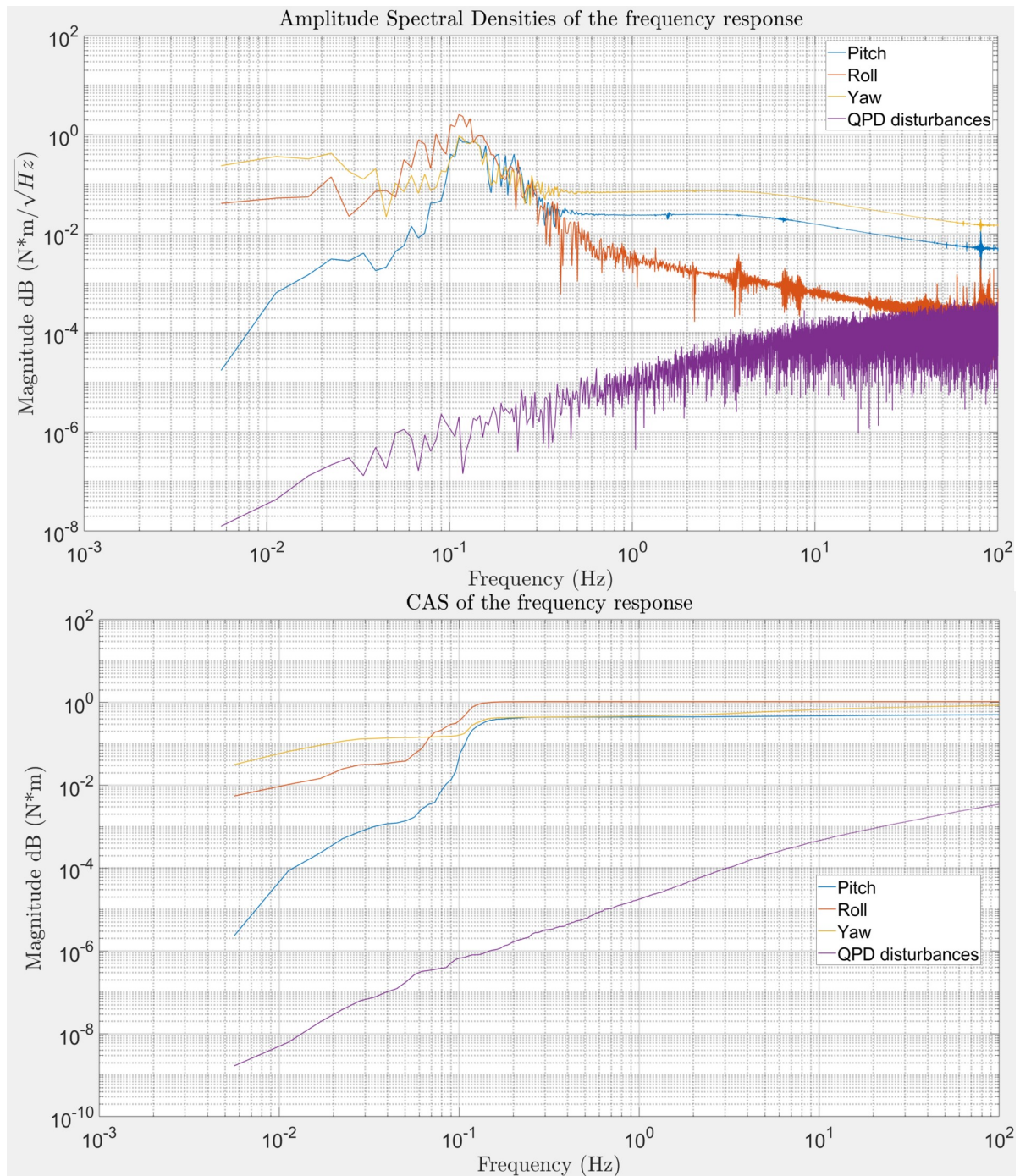


Figure G.11: DEB analysis results of the CPAS controller torque ASD and CAS plots

Table G.8: CPAS Controller torque DEB analysis performance results

RMS torque contribution	Unit [$mN \cdot m$]
Pitch Angle	504.98
Roll Angle	1046.97
Yaw Angle	853.09
QPD sensor	3.47
Total RMS torque (1σ)	2408.53

The summary of results provided on Table G.9 are relatively similar to those provided earlier in the PPAS actuator angular effort (Table G.3). This could be explained based on the fact that both systems are attempting to offset the same disturbances with the same amplitude, albeit there is a minor difference in effort with respect to the displacements performed by each actuator. The total RMS angular effort is of $3\sigma \approx 9.06 \text{ degrees}$, which is compatible with the angular displacement ranges of the L-350 (namely being 0-90 *Degrees* for northern/southern hemispheres [48]).

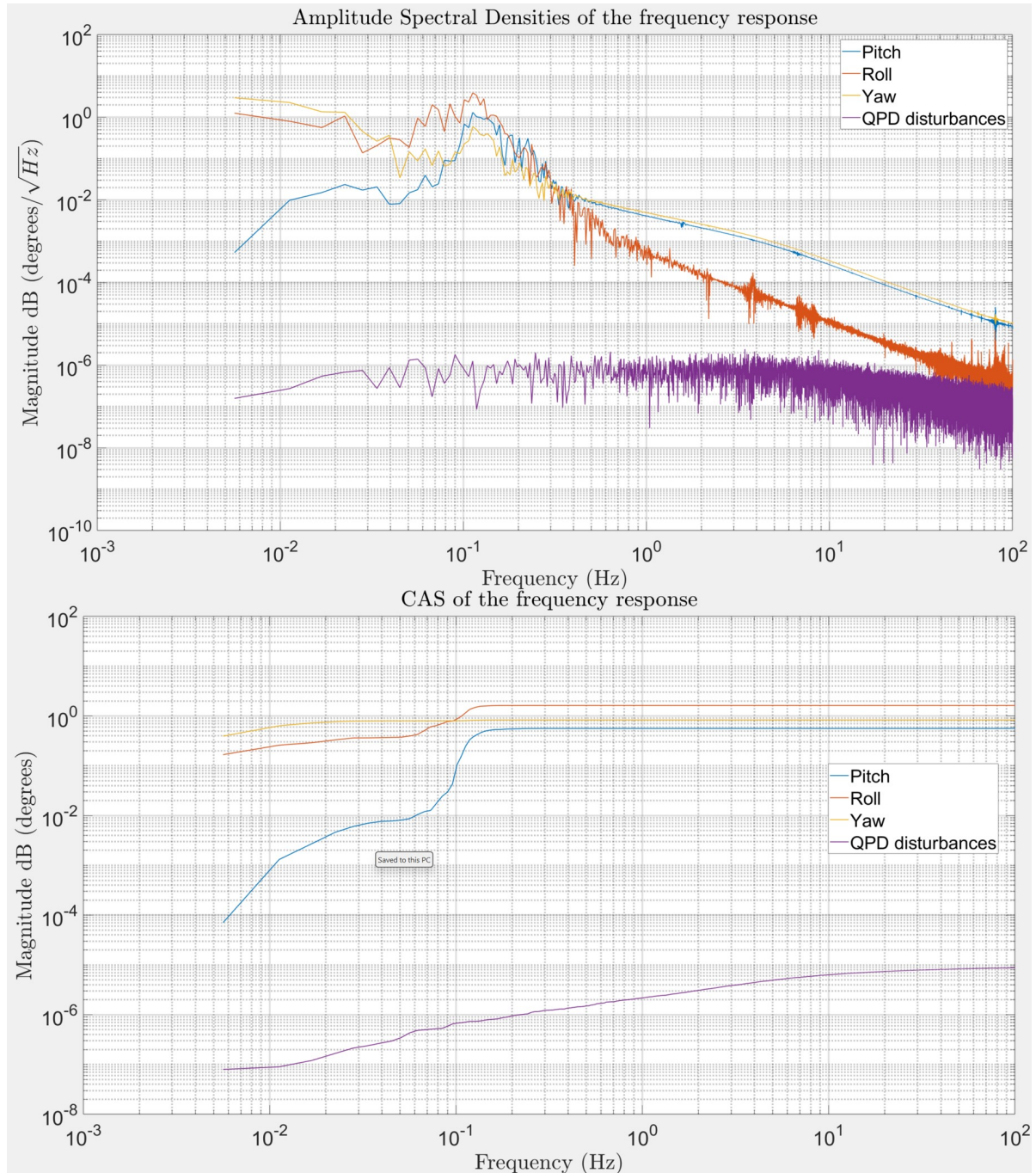


Figure G.12: DEB analysis results of the CPAS actuator angular effort ASD and CAS plots

Table G.9: CPAS actuator angular effort DEB analysis performance results

RMs error contribution	Unit [<i>mdegrees</i>]	Unit [<i>mrads</i>]	Unit [<i>Arcsec</i>]
Pitch Angle	561.10	9.79	2019.97
Roll Angle	1629.39	28.43	5865.81
Yaw Angle	831.74	14.51	2994.27
QPD sensor	0.008	0.15×10^{-3}	0.031
Total RMS error (1σ)	3022.24	52.74	10880.09

G.3.3. CPAS DEB pointing angle-rate error, controller power and actuator angular-rate effort results

From the results shown in Figure G.13, it is possible to denote from the ASD plot that the contributions to the pointing angle rate error of the Ship-Wave disturbances peak at frequencies above 10 Hz , staying consistently high, until reaching the cut-off frequency. From these results is possible to argue that the CPAS is encountering difficulties with offsetting/counteracting disturbances occurring at higher frequencies, even as those are of lower amplitude when compared to those occurring at lower frequencies.

This outcome is also visible in the CAS plot, where as expected, the errors increase quickly at lower frequencies, then stagnate, and then significantly increase thereafter. This once again, as in the angle pointing error analysis, highlights the limits imposed by the relatively low controller bandwidth, leading to an incapacity to comply with requirement **PATR-01**, as the total RMS pointing angle-rate error of $3\sigma \approx 2.531\text{ mrads/sec}$.

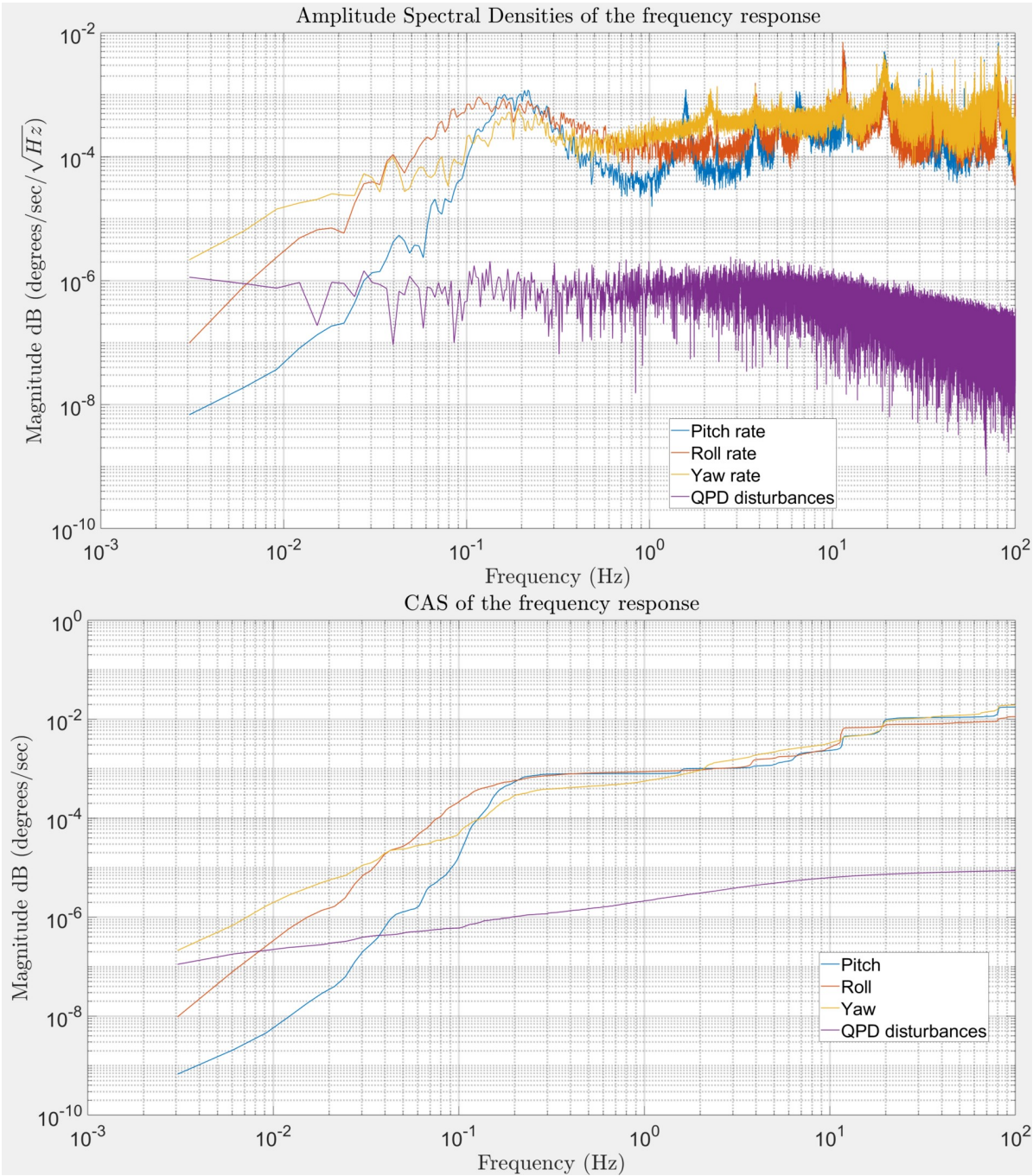


Figure G.13: DEB analysis results of the CPAS pointing angle rate ASD and CAS plots

Table G.10: CPAS Pointing Angle Rate DEB analysis performance results

RMS error contribution	Unit [<i>mdegrees/sec</i>]	Unit [<i>mrads/sec</i>]	Unit [<i>Arcsec/sec</i>]
Pitch Angle	17.64	0.30	63.51
Roll Angle	11.30	0.19	40.70
Yaw Angle	19.39	0.33	69.82
QPD sensor	0.008	0.152×10^{-3}	0.031
Total RMS error (1σ)	48.35	0.843	174.07

From Figure G.14 it is possible to highlight from the ASD plot, that the contribution to the controller power requirements concentrates at two different sections, the first section corresponds to the power required to off-set low-frequency disturbances with a high amplitude, while the second section corresponds to high-frequency disturbances, which are characterized to have a lower amplitude but their high-frequency content could be requiring an increasing amount of power. The total RMS power requirement is $3\sigma \approx 18.37\text{ W}$, which should be achievable as the standard power supply system of the L-350 should be capable of supplying 1800 W (120 VAC 15A IEC Type B Regulated Power Adapter [48]).

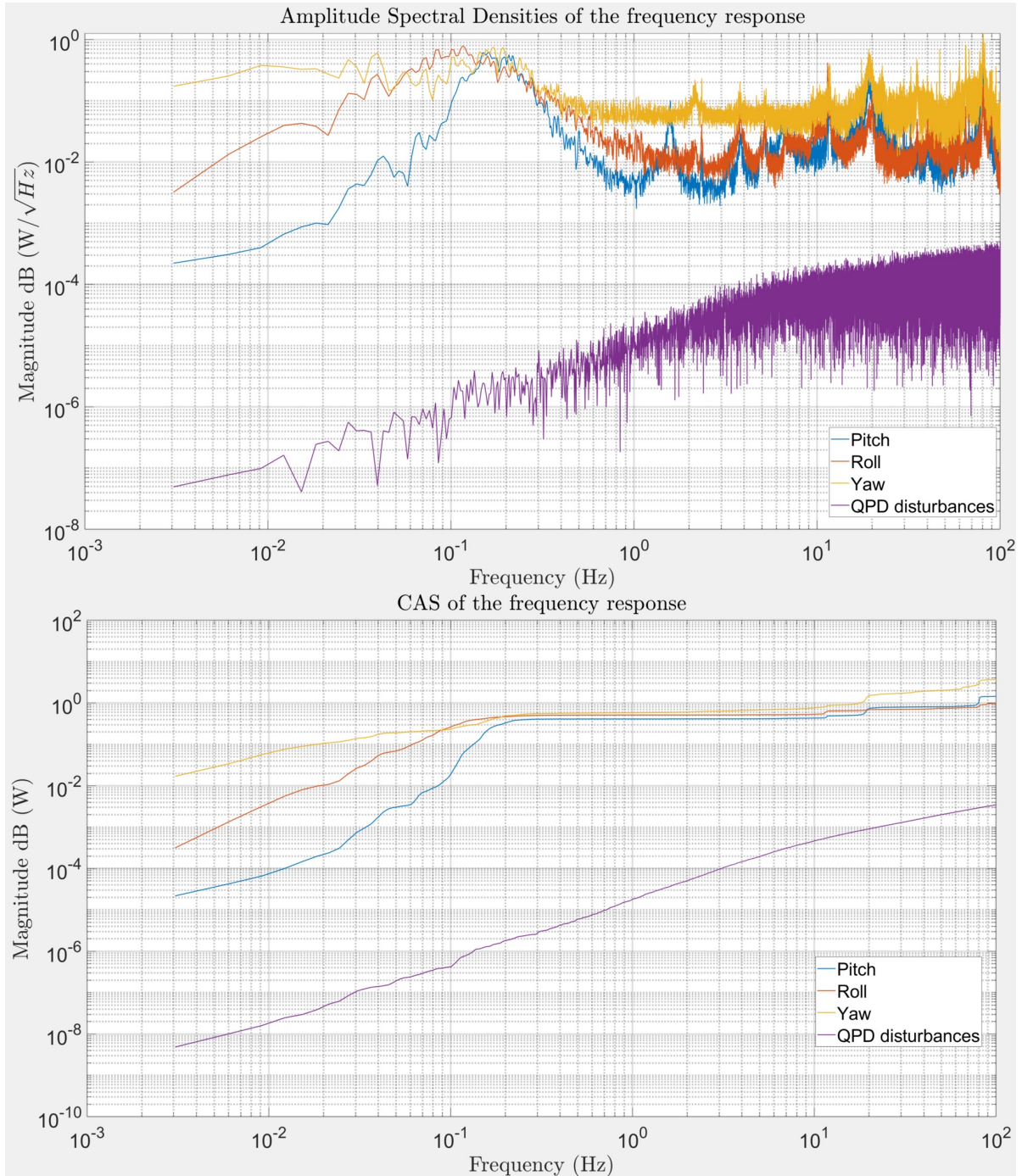


Figure G.14: DEB analysis results of the CPAS controller power ASD and CAS plots

Table G.11: CPAS Controller Power DEB analysis performance results

RMS power contribution	Unit [mW]
Pitch Angle	1445.25
Roll Angle	981.80
Yaw Angle	3693.46
QPD sensor	3.474
Total RMS power (1σ)	6124.00

Is possible to conclude from the CAS plot shown in Figure G.15, that the great majority of the angular rate effort requirements build up before reaching $1Hz$, which could be explained, either by the majority of the amplitude of the disturbances occurring before this frequency or due to the limitations imposed by low controller bandwidth limiting the capacity of requiring larger angular rate effort. The total RMS angular rate effort is $3\sigma \approx 6.69 \text{ degrees/sec}$. Which falls well in line with the specifications of the L-350 mount capable of continuous 20 degrees/sec slew rate [48].

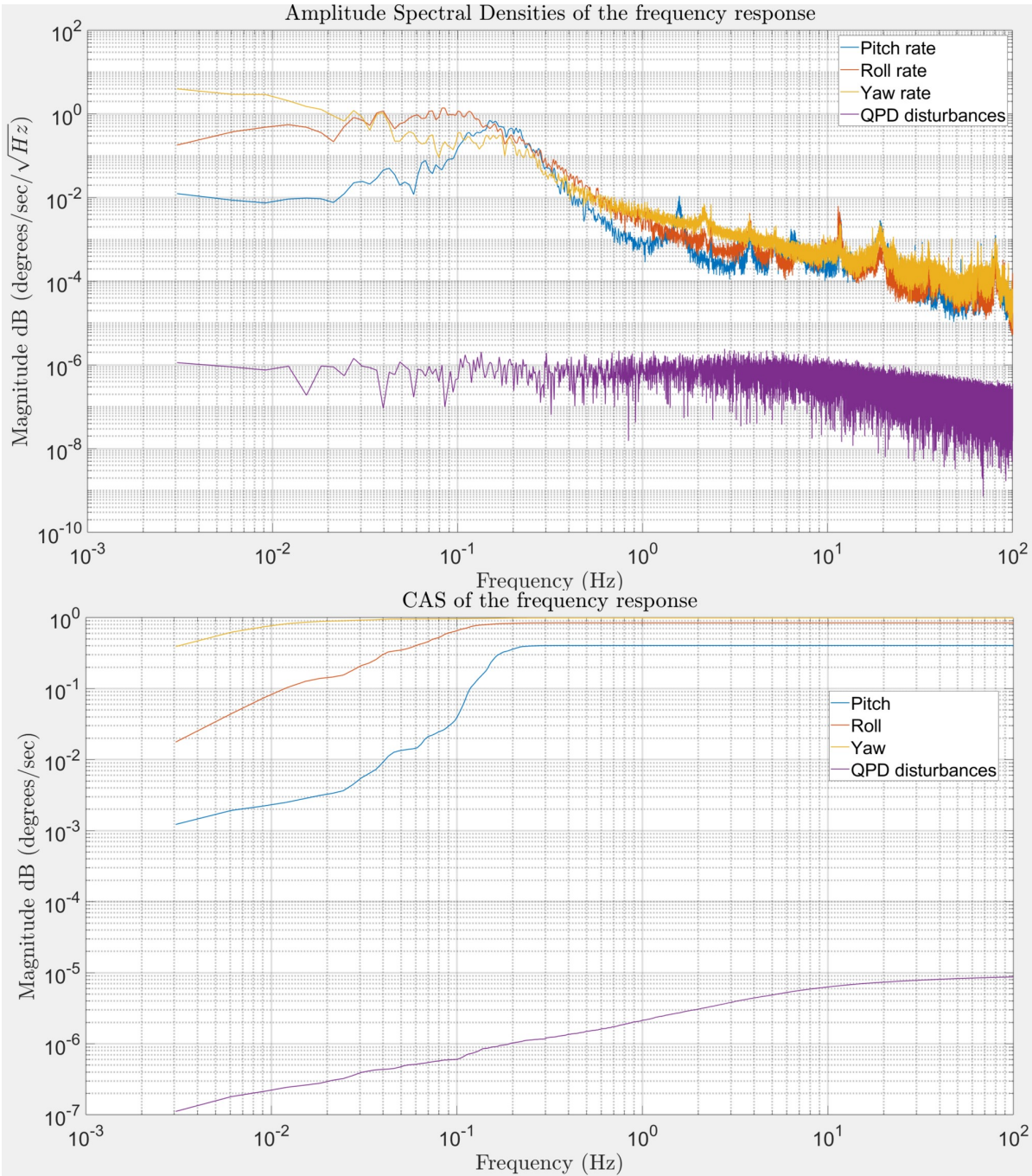


Figure G.15: DEB analysis results of the CPAS actuator angular rate effort ASD and CAS plots

Table G.12: CPAS Actuator Angular Rate effort DEB analysis performance results

RMs error contribution	Unit [<i>mdegrees/sec</i>]	Unit [<i>mrads/sec</i>]	Unit [<i>Arcsec/sec</i>]
Pitch Angle	403.67	7.04	1453.21
Roll Angle	842.12	14.69	3031.65
Yaw Angle	986.27	17.21	3550.57
QPD sensor	0.008	0.152X10 ⁻³	0.031
Total RMS error (1σ)	2232.07	38.95	8035.48

G.4. DSAS DEB analysis results and figures

G.4.1. DSAS DEB pointing angle error, controller torque and actuator angular effort results

With the DSAS DEB equations covered, next is to discuss the multiple results acquired by performing the DEB analysis. Starting with the results from the total system pointing angle error analysis, are the ASD and CAS plots, visible on Figure G.16. Notable from these results is the significant attenuation of the ship-wave disturbances at lower frequencies. At such frequency ranges the sensor disturbances play the dominant role until reaching $\approx 12 \text{ Hz}$, where from that point-onwards the pitch and yaw then become dominant. This is also visible on the CAS plot where the contribution of the QPD sensor error dominates at lower frequencies and its total contribution stays above the Roll disturbances.

The outcomes visible from both plots are directly visible as well on the total pointing error summary provided on Table G.13, where the pitch, yaw and QPD-sensor errors are the 3 main contributors. The total RMS error is set at $3\sigma \approx 21.75 \text{ } \mu\text{rads}$, which is $\approx 56 \%$ below the maximum PATR-01 RMS pointing jitter requirement.

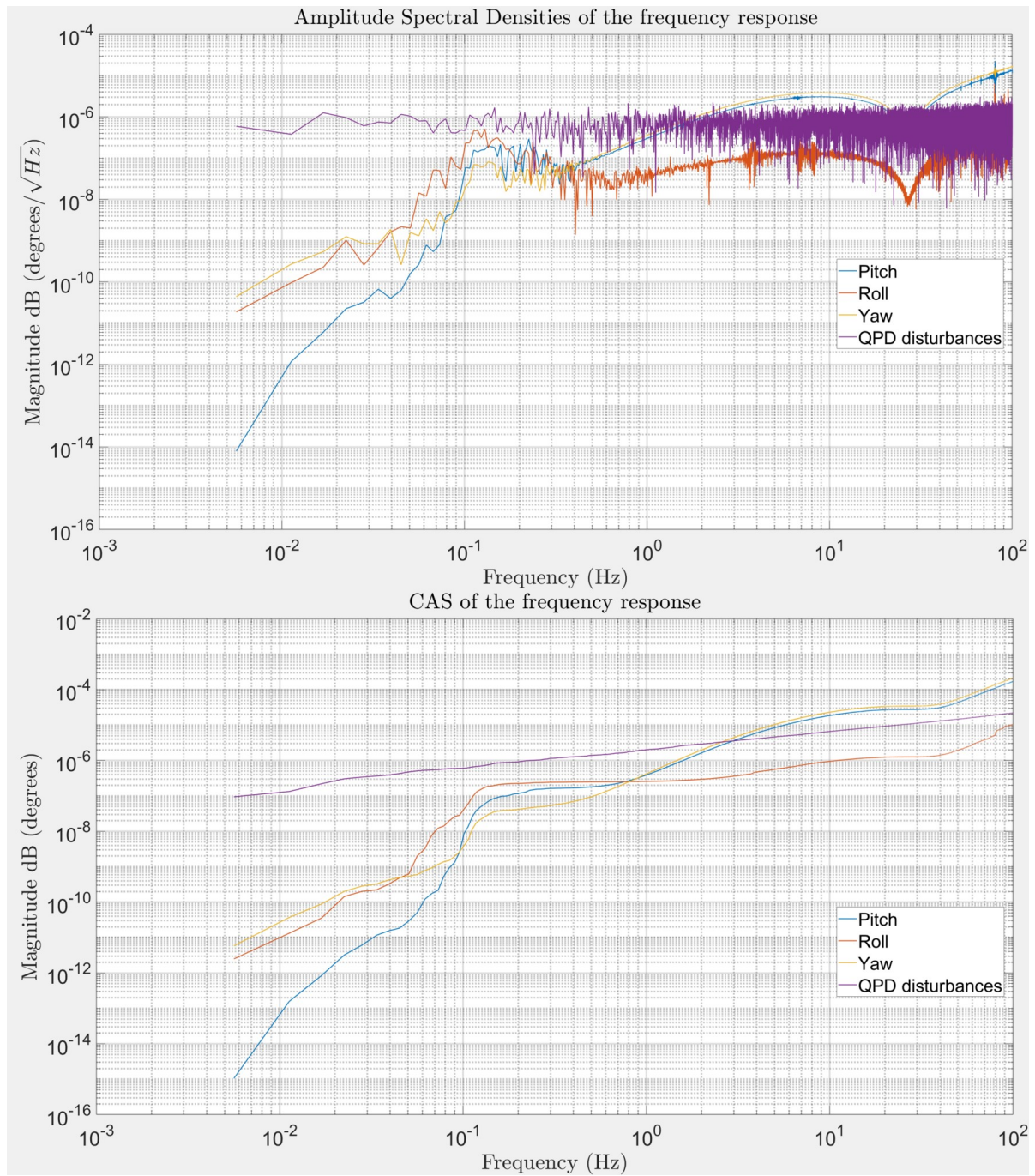


Figure G.16: DEB analysis results of the DSAS total actuator pointing angle error ASD and CAS plots

Table G.13: CPAS actuator pointing angle error DEB analysis performance results

RMs error contribution	Unit [$\mu\text{degrees}$]	Unit [μrads]	Unit [Arcsec]
Pitch Angle	171.09	2.98	0.61
Roll Angle	10.31	0.18	0.03
Yaw Angle	212.05	3.70	0.76
QPD sensor	22.09	0.38	0.07
Total RMS error (1σ)	415.55	7.25	1.49

Next is the interpretation of the chief and deputy controller torque results. Visible from the ASD plots shown in Figure G.17, is the aspect that the ship-wave disturbances require the highest amount of torque at lower frequencies, starting to decline at frequencies higher than $\approx 0.15 \text{ Hz}$. Beyond these frequencies, the behavior of the chief and deputy controllers deviate from each other.

The torque requirements increase once again when looking at the contributions beyond $\approx 10 \text{ Hz}$ which is to be expected as it is the first natural frequency of the gimbal actuator, from there onwards the behavior changes, whereas, for the deputy controller, the torque efforts eventually decline and settle out, the controller demands the chief controller increase significantly thereafter. This effect is also visible in the CAS plot, where the cumulative torque effort increases significantly towards the last sampled frequencies. This is to be expected as the disturbances occurring at higher frequencies, stay within the chief controller bandwidth, thus the chief stage is capable of still offsetting these errors coming at the cost of higher torques, albeit the torque requirements of the chief actuator remain marginal.

From the summary of results provided on Table G.14, it is possible to conclude that the total RMS controller torque for the chief controller is $3\sigma \approx 107.76 \mu\text{N} * m$, while for the deputy controller is $3\sigma \approx 7222.21 \text{ mN} * m$. While the controller torque effort required for both chief and deputy controllers remains manageable, it is noteworthy to observe that the dual-stage controller system has reduced the torque requirements of the chief actuator by at least an order of magnitude, when compared to the chief actuator working alone as in the PPAS.

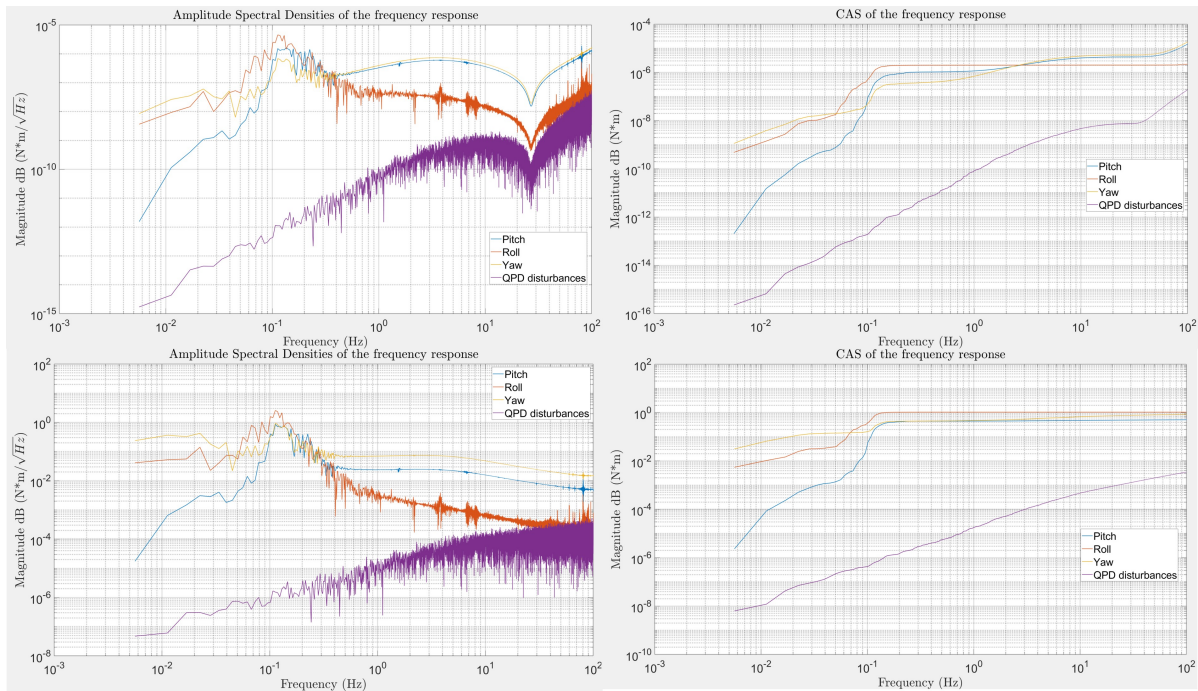


Figure G.17: DEB analysis results of the DSAS chief and deputy controller torque ASD and CAS plots. Top plots correspond to the chief controller, while bottom ones correspond to the deputy controller

Table G.14: DSAS Chief and Deputy Controller torque DEB analysis performance results

Chief controller torque		Deputy controller torque
RMS torque contribution	Unit [$\mu N * m$]	Unit [$mN * m$]
Pitch Angle	14.99	504.80
Roll Angle	2.19	1046.97
Yaw Angle	18.53	852.14
QPD sensor	0.19	3.47
Total RMS torque (1σ)	35.92	2407.40

Lastly to discuss, are the results from chief and deputy actuator effort results. Visible in Figure G.18 are the ASD and CPAS actuator angular effort results. A similar story to the one of the controller effort results, is visible here, where the actuator efforts for both chief and deputy actuators are higher at lower frequencies.

From Table G.15 it is possible to determine the total RMS actuator effort of the chief actuator of $3\sigma \approx 28.49 \text{ mdegrees}$, which falls well within the FSM (chief actuator) capabilities, highlighting the fact that the dual-stage configuration has helped to offload the chief actuator. Next is the total RMS actuator effort of the deputy actuator with a value of $3\sigma \approx 9.06 \text{ degrees}$, which is well within the range of the deputy actuator. From both of these results, it is possible to conclude that both chief and deputy actuators show the capacity to remain within their respective angular ranges.

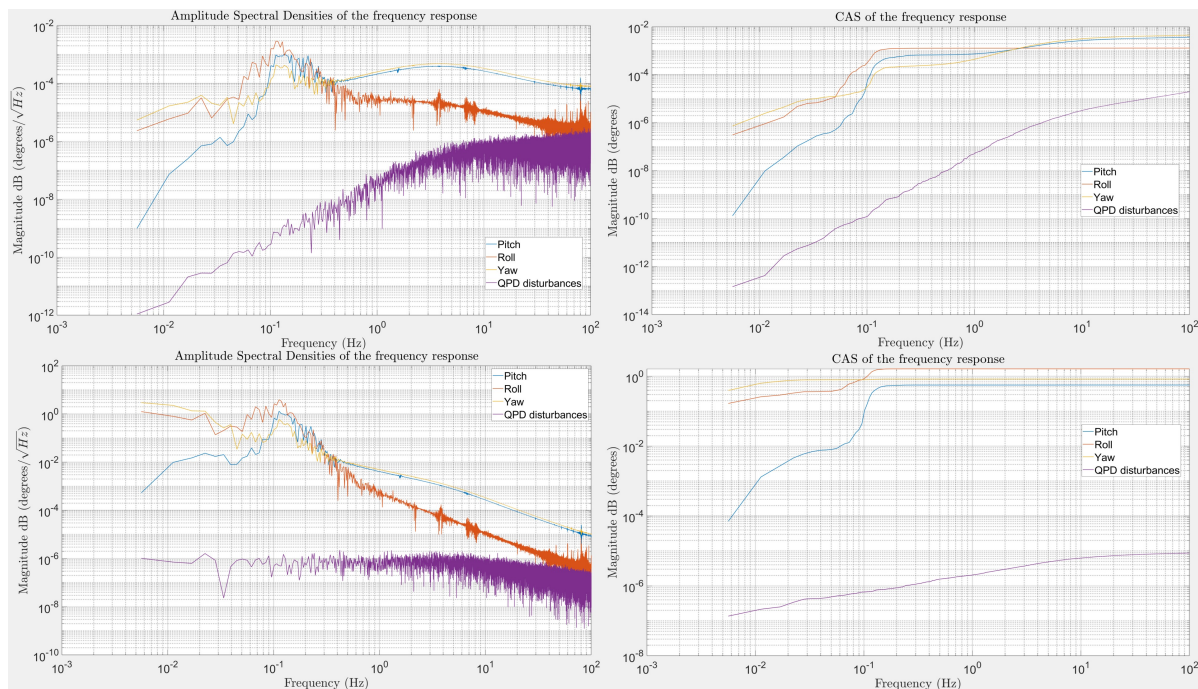


Figure G.18: DEB analysis results of the DSAS chief and deputy actuator angular effort ASD and CAS plots. The plots at the top side of the figure correspond to those of the chief actuator, while those at the bottom correspond to those of the deputy actuator.

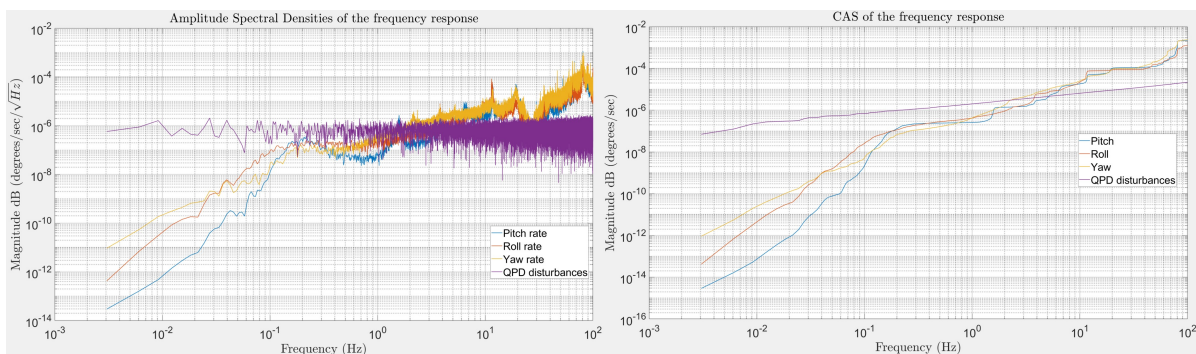
Table G.15: DSAS chief and deputy actuators angular efforts DEB analysis performance results

Chief actuator effort			Deputy actuator effort			
RMs error contribution	Unit [mdegrees]	Unit [mrads]	Unit [Arcsec]	Unit [mdegrees]	Unit [mrads]	Unit [Arcsec]
Pitch Angle	3.68	0.064	13.24	560.509	9.782	2017.832
Roll Angle	1.30	0.022	4.68	1628.248	4.49	5861.690
Yaw Angle	4.49	0.078	16.19	831.673	14.515	2994.023
QPD sensor	0.019	0.34×10^{-3}	0.07	0.015	0.275×10^{-3}	0.056
Total RMS effort (1σ)	9.49	0.165	34.19	3022.24	52.74	10880.09

G.4.2. DSAS DEB pointing angle-rate error, controller power and actuator angular-rate effort results

As visible from the ASD plot in Figure G.19, for lower frequencies, the contribution of the QPD sensor errors heavily influences the contribution towards pointing angle-rate errors. This holds until reaching frequencies above ≈ 10 Hz, after this, the ship-wave disturbances overtake to become the prominent sources of errors.

These sources of errors add up quickly at the frequency closest to the cut-off frequency, as visible from the CAS plot. From Table G.16 it is possible to determine the total RMS pointing angle-rate errors of $3\sigma \approx 308 \mu\text{rad/sec}$. This highlights the inability of the base performance (398 Hz for the chief actuator and 10 Hz for the deputy actuator) DSAS to fully comply with **PATR-01** pointing jitter error requirement.

**Figure G.19:** DEB analysis results of the DSAS total pointing angle rate ASD and CAS plots**Table G.16:** DSAS Total Pointing Angle Rate DEB analysis performance results

RMs error contribution	Unit [mdegrees/sec]	Unit [mrads/sec]	Unit [Arcsec/sec]
Pitch Angle	2.25	0.039	8.09
Roll Angle	1.25	0.021	4.53
Yaw Angle	2.35	0.041	8.46
QPD sensor	0.022	0.384×10^{-3}	0.079
Total RMS error (1σ)	5.882	0.102	21.17

Next is to discuss the power requirement results shown in Figure G.20. As visible from the ASD and CAS plots, the power requirements follow along the controller bandwidths, in the case of the chief controller, having a larger bandwidth, it is capable of requiring consistently higher amounts of power, as more energy is required for off-setting the disturbances occurring at higher frequencies.

The opposite is true for the deputy controller, where the highest contribution to power requirements occurs at lower frequencies. From Table G.17, it is possible to determine the total RMS of the chief controller power draw is at $3\sigma \approx 1.47 \text{ mW}$ while the deputy controller is at $3\sigma \approx 18.33 \text{ W}$, both remaining manageable for the selected setups.

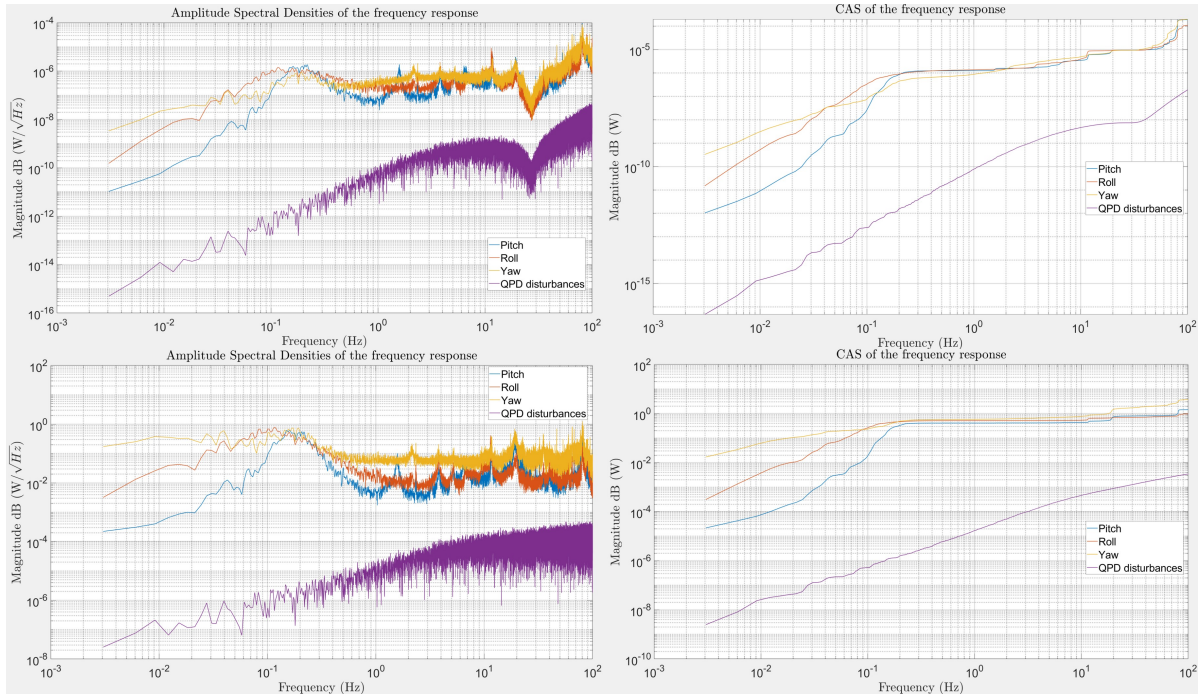


Figure G.20: DEB analysis results of the DSAS Chief and Deputy controller power ASD and CAS plots

Table G.17: DSAS Chief and Deputy Controller Power DEB analysis performance results

Chief controller power		Deputy controller power
RMS torque contribution	Unit [μW]	Unit [mW]
Pitch Angle	188	1442.59
Roll Angle	108	980.47
Yaw Angle	195	3685.70
QPD sensor	0.18	3.45
Total RMS power (1σ)	492	6112.23

Lastly are the actuator angle-rate effort requirements, the ASD and CAS plots are shown in Figure G.21. Visible from the ASD plots are the contributions per frequency to the actuator angle-rate requirements, it is possible to observe that for the chief actuator, the actuator requirements remain consistent, only peaking at higher frequencies. The pointing requirements for the deputy actuator are highest at the lower frequencies gradually reducing at higher frequencies.

From Table G.18, it is possible to determine the total RMS angular-rate effort requirement for the chief actuator of $3\sigma \approx 154.42 \text{ mdegrees/sec}$ while the deputy actuator is at $3\sigma \approx 6.88 \text{ degrees/sec}$, which both actuators are capable of achieving.

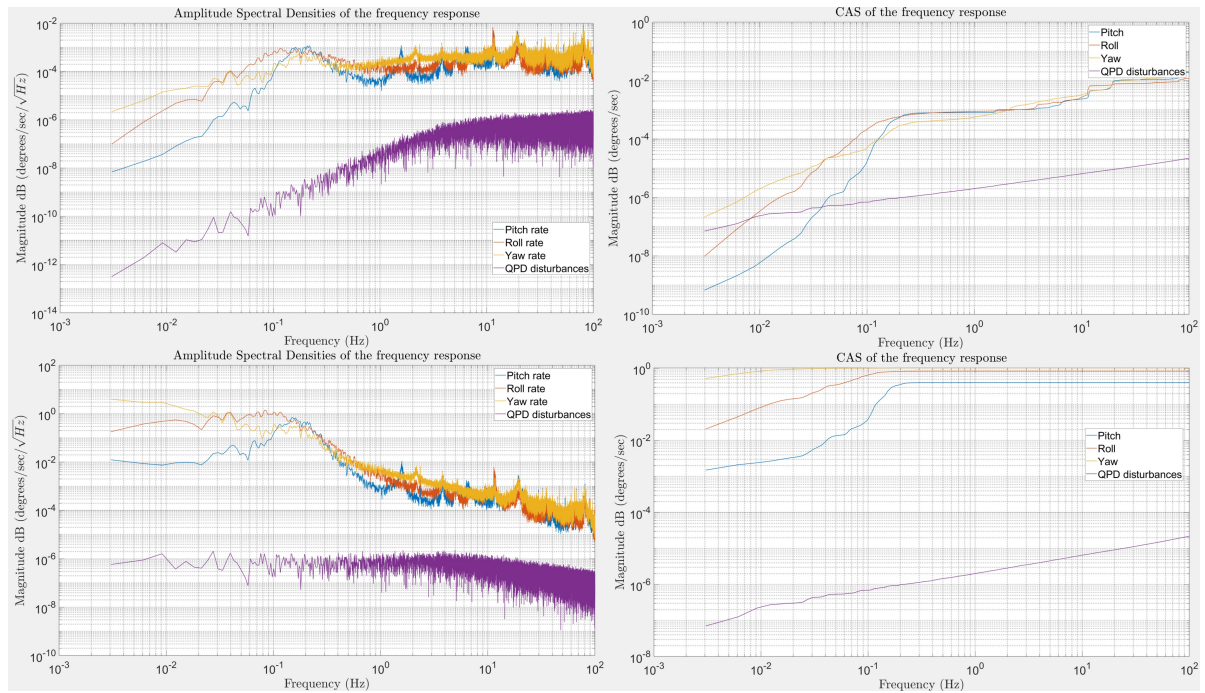
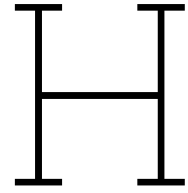


Figure G.21: DEB analysis results of the DSAS chief and deputy actuator angular rate effort ASD and CAS plots

Table G.18: CPAS Actuator Angular Rate effort DEB analysis performance results

Chief actuator angular rate				Deputy actuator angular rate		
RMS error contribution	Unit [mdeg/sec]	Unit [mrads/sec]	Unit [Arcsec/sec]	Unit [mdeg/sec]	Unit [mrads/sec]	Unit [Arcsec/sec]
Pitch rate	18.86	0.329	67.91	403.67	7.04	1453.22
Roll rate	11.87	0.207	42.73	842.19	14.69	3031.91
Yaw rate	20.72	0.361	74.59	1047.54	18.283	3771.14
QPD sensor-rate	0.022	0.384×10^{-3}	0.079	0.022	0.384×10^{-3}	0.079
Total RMS effort (1σ)	51.47	0.89	185.31	2293.43	40.02	8256.35



Dynamic Budget Error Analysis code

```
1 % Updated script utilized to perform DEB analysis of the PPAS stage
2 load('deb_analysis_angle_files.mat')
3 % import the general data set into individual vectors
4 avg_psd1 = reg_psd_pitch';
5 avg_psd2 = reg_psd_roll';
6 avg_psd3 = reg_psd_yaw';
7
8 % creation of the QPD psd errors
9 % Statistical error properties
10 mean = 0; % Mean of the positional error
11 std_dev = 8.27E-06; % Standard deviation of the positional error (square root of the variance
    ) QPD G6849-01
12 %std_dev = 2.78E-05;
13 % sizing properties to match the original reference signal
14 sampling_rate = 200; % Sampling rate in Hz
15 num_samples = length(original_signal); % Number of data points required to obtain an estimate
    of the PSD SW function
16
17 % Generate simulated positional error data
18 positional_error = mean + std_dev * randn(1, num_samples);
19
20 % convert positional error vector into PSD error vector
21 [frequency_vector, y_norm_amplitude, psd_error_data] =
    normalize_amplitude_dft_and_psd_creation(positional_error, sampling_rate);
22
23
24 % Discrete transform
25 s = tf('s');
26
27 % step 2 creation of the LTI models
28 % second order plant model parameters:
29 k = 11.22; % gain
30 k_rad_deg = 180/(pi); % Gain for conversion from rad to degrees [-]
31 w_0 = 169.6; % rad/s
32 C = 0.04; % damping ratio fitted to the measured data of the system
33 % measured sampling delay of the digital system
34 T_s = 50.04E-06; % Controller board running at 20khz with quadcell running at a
    sampling rate of 25 MHz
35 %T_s = 500.4E-06; % time delay increase by a factor of 10 to account for delays in
    the system
36 % FSM controller gain parameters
37 k_p = 0.06111;
38 k_i = 6.88191;
39 k_d = 0.0001317;
40 N = 9117.06; % rad/sec which is roughly a cutt-off frequency around 1451 Hz
41
42 % 2nd order pade approximation for time delay
43 P_qpd = ((1-(T_s/(2*2))*s)^2)/((1+(T_s/(2*2))*s)^2);
44 %P_qpd = exp(-s*T_s)
45 % FSM plant model
```



```

46 G_fsm = k_rad_deg*k*((w_0^2)/(s^2+2*w_0*C*s+w_0^2));
47 % Controller T.F.
48 C_fsm = k_p+k_i*(1/s)+k_d*(N/(1+N*(1/s)));
49
50
51 % step 3 obtain the loop and sensitivity function and perform sensitivity analysis
52
53 % first the frequency vector needs to be converted into rad/sec
54 frequency_padded_rad = 2*pi*frequency_original;
55
56 % loop function
57 L_s = minreal(G_fsm*C_fsm*P_qpd);
58 % sensitivity function
59 S_x = minreal(1/(1+L_s));
60 % modified sensitivity function for T.S. disturbances
61 S_x_ts = L_s/(1+L_s);
62 S_x_ts = minreal(S_x_ts);
63
64 % frequency response of the total sensitivity function
65 [S_x_mag,S_x_phase, wout] = bode(S_x,frequency_padded_rad);
66 S_x_mag = squeeze(S_x_mag);
67 % Convert magnitude to array and square it to get |S(jw)|^2
68 S_x_mag_squared = S_x_mag.^2';
69 % convert phase array
70 S_x_phase = squeeze(S_x_phase);
71
72 % frequency response of the T.S. sensitivity function
73 [S_x_ts_mag,S_x_ts_phase, wout2] = bode(S_x_ts,frequency_padded_rad);
74 S_x_ts_mag = squeeze(S_x_ts_mag);
75 % Convert magnitude to array and square it to get |S(jw)|^2
76 S_x_ts_mag_squared = S_x_ts_mag.^2';
77 % convert phase array
78 S_x_ts_phase = squeeze(S_x_ts_phase);
79
80
81 % PSD output of the multiplication between sensitivity function and input PSD
82 S_y_1 = avg_psd1.*S_x_mag_squared;
83 S_y_2 = avg_psd2.*S_x_mag_squared;
84 S_y_3 = avg_psd3.*S_x_mag_squared;
85
86 % output of the sensitivity function
87 S_y_ts = psd_error_data.*S_x_ts_mag_squared;
88 % total sum of disturbances PSD output
89 S_y_total = S_y_1+S_y_2+S_y_3+S_y_ts;
90
91 % Amplitude Spectral Density of the output PSD
92 Asd_1 = sqrt(S_y_1);
93 Asd_2 = sqrt(S_y_2);
94 Asd_3 = sqrt(S_y_3);
95 Asd_4 = sqrt(S_y_ts);
96 Asd_total = sqrt(S_y_total);
97
98
99 figure;
100 loglog(frequency_vector,Asd_1)
101 hold on
102 loglog(frequency_vector,Asd_2)
103 hold on
104 loglog(frequency_vector,Asd_3)
105 hold on
106 loglog(frequency_vector,Asd_4)
107 % hold on
108 % semilogx(frequency_padded,20*log10(Asd_total))
109 grid on
110 legend('Pitch', 'Roll','Yaw','QPD_disturbances')
111 xlabel('Frequency(Hz)','Interpreter','latex');
112 ylabel('Magnitude_dB(degrees/$\sqrt{Hz}$)','Interpreter','latex');
113 title('Amplitude Spectral Densities of the frequency response','Interpreter','latex')
114
115 % Integrate over the frequency range to find total effect
116 total_effect_1 = sqrt(trapz(wout, S_y_1));

```

```

117 total_effect_2 = sqrt(trapz(wout, S_y_2));
118 total_effect_3 = sqrt(trapz(wout, S_y_3));
119 total_effect_4 = sqrt(trapz(wout, S_y_ts));
120 total_effect_total = total_effect_1+total_effect_2+total_effect_3+total_effect_4;
121
122 % Display Standard deviations
123 disp(['\sigma{rms}_Pitch_rate_Disturbance:', num2str(1e+6*total_effect_1), '\u00b5micro-degrees,',
    , num2str(1e+6*(pi/180)*total_effect_1), '\u00b5micro-rads,', ...
    , num2str(3600*total_effect_1), '\u00b5arcsec']);
124 disp(['\sigma{rms}_Roll_rate_Disturbance:', num2str(1e+6*total_effect_2), '\u00b5micro-degrees,',
    , num2str(1e+6*(pi/180)*total_effect_2), '\u00b5micro-rads,', ...
    , num2str(3600*total_effect_2), '\u00b5arcsec']);
125 disp(['\sigma{rms}_Yaw_rate_Disturbance:', num2str(1e+6*total_effect_3), '\u00b5micro-degrees,',
    , num2str(1e+6*(pi/180)*total_effect_3), '\u00b5micro-rads,', ...
    , num2str(3600*total_effect_3), '\u00b5arcsec']);
126 disp(['\sigma{rms}_QPD_Disturbance:', num2str(1e+6*total_effect_4), '\u00b5micro-degrees,',
    , num2str(1e+6*(pi/180)*total_effect_4), '\u00b5micro-rads,', ...
    , num2str(3600*total_effect_4), '\u00b5arcsec']);
127
128 % Display total standard deviation
129 disp(['\sigma{rms}_of_Disturbances:', num2str(1e+6*total_effect_total), '\u00b5micro-degrees,',
    , num2str(1e+6*(pi/180)*total_effect_total), '\u00b5micro-rads,', ...
    , num2str(3600*total_effect_total), '\u00b5arcsec']);
130
131 % Classic approach
132 cps_1 = cumtrapz(wout, S_y_1);
133 cps_2 = cumtrapz(wout, S_y_2);
134 cps_3 = cumtrapz(wout, S_y_3);
135 cps_4 = cumtrapz(wout, S_y_ts);
136 cps_total = cps_1+cps_2+cps_3+cps_4;
137
138 % plot to see results
139 figure;
140 loglog(frequency_vector, sqrt(cps_1))
141 hold on
142 loglog(frequency_vector, sqrt(cps_2))
143 hold on
144 loglog(frequency_vector, sqrt(cps_3))
145 hold on
146 loglog(frequency_vector, sqrt(cps_4))
147 grid on
148 legend('Pitch', 'Roll', 'Yaw', 'QPD_disturbances')
149 xlabel('Frequency (Hz)', 'Interpreter', 'latex');
150 ylabel('Magnitude dB (degrees)', 'Interpreter', 'latex');
151 title('CAS of the frequency response', 'Interpreter', 'latex')
152
153 % get final results
154 total_cps_effect_1 = sqrt(cps_1(end));
155 total_cps_effect_2 = sqrt(cps_2(end));
156 total_cps_effect_3 = sqrt(cps_3(end));
157 total_cps_effect_4 = sqrt(cps_4(end));
158 total_cps_effect_total = total_cps_effect_1+total_cps_effect_2+total_cps_effect_3+
    total_cps_effect_4;
159
160 % Display individual effects
161 disp(['Total_CPS_Effect_Pitch_rate_Disturbance_on_System_Output:', num2str(1000*
    total_cps_effect_1), '\u00b5mdegrees,', num2str(1000*(pi/180)*total_cps_effect_1), '\u00b5mrads,',
    ...
    , num2str(3600*total_cps_effect_1), '\u00b5arcsec']);
162 disp(['Total_CPS_Effect_Roll_rate_Disturbance_on_System_Output:', num2str(1000*
    total_cps_effect_2), '\u00b5mdegrees,', num2str(1000*(pi/180)*total_cps_effect_2), '\u00b5mrads,',
    ...
    , num2str(3600*total_cps_effect_2), '\u00b5arcsec']);
163 disp(['Total_CPS_Effect_Yaw_rate_Disturbance_on_System_Output:', num2str(1000*
    total_cps_effect_3), '\u00b5mdegrees,', num2str(1000*(pi/180)*total_cps_effect_3), '\u00b5mrads,',
    ...
    , num2str(3600*total_cps_effect_3), '\u00b5arcsec']);
164 disp(['Total_CPS_Effect_QPD_Disturbance_on_System_Output:', num2str(1000*total_cps_effect_4)
    , '\u00b5mdegrees,', num2str(1000*(pi/180)*total_cps_effect_4), '\u00b5mrads,', ...
    , num2str(3600*total_cps_effect_4), '\u00b5arcsec']);
165
166 % Display total effect
167 disp(['Total_Effect_CPS (S.W.+T.S.) of Disturbance on System Output:', num2str(1000*
    total_cps_effect_total), '\u00b5mdegrees,', num2str(1000*(pi/180)*total_cps_effect_total), '\u00b5mrads,',
    ...
    , num2str(3600*total_cps_effect_total), '\u00b5arcsec']);
168

```

```

    mrad_s, ...
    num2str(3600*total_cps_effect_total), '\uarcsec']);
174
175
176
177 % DEB controller effort
178
179 %controller specific transfer function
180 S_ut = minreal(S_x*C_fsm*P_qpd);
181 % frequency response of the controller specific sensitivity function
182 [S_ut_mag,S_ut_phase, wout_ut] = bode(S_ut,frequency_padded_rad);
183 S_ut_mag = squeeze(S_ut_mag);
184 % Convert magnitude to array and square it to get |S(jw)|^2
185 S_ut_mag_squared = S_ut_mag.^2;
186
187 % PSD output of the multiplication between sensitivity function and input PSD
188 S_y_1 = avg_psd1.*S_ut_mag_squared;
189 S_y_2 = avg_psd2.*S_ut_mag_squared;
190 S_y_3 = avg_psd3.*S_ut_mag_squared;
191
192 % output of the sensitivity function
193 S_y_4 = psd_error_data.*S_ut_mag_squared;
194 % total sum of disturbances PSD output
195 S_y_total = S_y_1+S_y_2+S_y_3+S_y_4;
196
197 % Amplitude Spectral Density of the output PSD
198 Asd_1 = sqrt(S_y_1);
199 Asd_2 = sqrt(S_y_2);
200 Asd_3 = sqrt(S_y_3);
201 Asd_4 = sqrt(S_y_4);
202 Asd_total = sqrt(S_y_total);
203
204
205 figure;
206 loglog(frequency_vector,Asd_1)
207 hold on
208 loglog(frequency_vector,Asd_2)
209 hold on
210 loglog(frequency_vector,Asd_3)
211 hold on
212 loglog(frequency_vector,Asd_4)
213 grid on
214 legend('Pitch', 'Roll','Yaw','QPD_disturbances')
215 xlabel('Frequency\u(Hz)','Interpreter','latex');
216 ylabel('Magnitude\uB\u(N*m/\u\sqrt{Hz}\u)','Interpreter','latex');
217 title('Amplitude\uSpectral\uDensities\uof\uthe\ufrequency\uresponse','Interpreter','latex')
218
219
220
221 % Integrate over the frequency range to find total effect
222 total_effect_1 = sqrt(trapz(wout, S_y_1));
223 total_effect_2 = sqrt(trapz(wout, S_y_2));
224 total_effect_3 = sqrt(trapz(wout, S_y_3));
225 total_effect_4 = sqrt(trapz(wout, S_y_4));
226 total_effect_total = total_effect_1+total_effect_2+total_effect_3+total_effect_4;
227
228 % Display Standard deviations
229 disp(['\sigma{rms}\uPitch\uRate\uDisturbance:\u', num2str(1e+3*total_effect_1), '\u mN*m']);
230 disp(['\sigma{rms}\uRoll\uRate\uDisturbance:\u', num2str(1e+3*total_effect_2), '\u mN*m']);
231 disp(['\sigma{rms}\uYaw\uRate\uDisturbance:\u', num2str(1e+3*total_effect_3), '\u mN*m']);
232 disp(['\sigma{rms}\uQPD\uDisturbance:\u', num2str(1e+3*total_effect_4), '\u mN*m']);
233 % Display total standard deviation
234 disp(['\sigma{rms}\uof\uDisturbances:\u', num2str(1e+3*total_effect_total), '\u mN*m']);
235
236 % Classic approach
237 cps_1 = cumtrapz(wout,S_y_1);
238 cps_2 = cumtrapz(wout,S_y_2);
239 cps_3 = cumtrapz(wout,S_y_3);
240 cps_4 = cumtrapz(wout,S_y_4);
241 cps_total = cps_1+cps_2+cps_3+cps_4;
242 % plot to see results
243 figure;

```

```

244 loglog(frequency_vector,sqrt(cps_1))
245 hold on
246 loglog(frequency_vector,sqrt(cps_2))
247 hold on
248 loglog(frequency_vector,sqrt(cps_3))
249 hold on
250 loglog(frequency_vector,sqrt(cps_4))
251 grid on
252 legend('Pitch', 'Roll','Yaw','QPD_disturbances')
253 xlabel('Frequency(Hz)','Interpreter','latex');
254 ylabel('Magnitude_dB(N*m)','Interpreter','latex');
255 title('CAS_of_the_frequency_response','Interpreter','latex')
256 % get final results
257 total_cps_effect_1 = sqrt(cps_1(end));
258 total_cps_effect_2 = sqrt(cps_2(end));
259 total_cps_effect_3 = sqrt(cps_3(end));
260 total_cps_effect_4 = sqrt(cps_4(end));
261 total_cps_effect_total = total_cps_effect_1+total_cps_effect_2+total_cps_effect_3+
    total_cps_effect_4;
262 % Display individual effects
263 disp(['Total_CPS_Effect_Pitch_rate_Disturbance_on_System_Output:', num2str(
    total_cps_effect_1),'Nm']);
264 disp(['Total_CPS_Effect_Roll_rate_Disturbance_on_System_Output:', num2str(total_cps_effect_2
    ),'Nm']);
265 disp(['Total_CPS_Effect_Yaw_rate_Disturbance_on_System_Output:', num2str(total_cps_effect_3)
    , 'Nm']);
266 disp(['Total_CPS_Effect_QPD_Disturbance_on_System_Output:', num2str(total_cps_effect_4),'Nm
    ']);
267 % Display total effect
268 disp(['Total_Effect_CPS(S.W.+T.S.)of_Disturbance_on_System_Output:', num2str(
    total_cps_effect_total),'Nm']);
269
270
271
272
273 function [frequency_vector, y_norm_amplitude, psd_imu_data ] =
    normalize_amplitude_dft_and_psd_creation(imu_data, sampling_frequency)
274 % determine sample size i.e. the length of the signal
275 N = length(imu_data);
276
277 % perform calculation of DFT with a two sided amplitude
278 y_norm_amplitude = abs(fft(imu_data))/N;
279
280 % sizing to one sided amplitude
281 y_norm_amplitude = y_norm_amplitude(1:N/2+1);
282
283 % double values except for DC and Nyquist frequencies
284 y_norm_amplitude(2:end-1) = 2*y_norm_amplitude(2:end-1);
285
286 % create a dedicated frequency vector
287 frequency_vector = 0:sampling_frequency/N:sampling_frequency-1/sampling_frequency;
288
289 % adjust frequency vector
290 frequency_vector = frequency_vector(1:N/2+1);
291
292 % perform calculation of PSD with two sided power
293 psd_imu_data = (1/(sampling_frequency*N))*abs(fft(imu_data)).^2;
294
295 % sizing to one sided power
296 psd_imu_data = psd_imu_data(1:N/2+1);
297
298 % double values except for DC and Nyquist frequencies
299 psd_imu_data(2:end-1) = 2*psd_imu_data(2:end-1);
300
301 end

```

Verification results full-size plots and Code

I.1. Frequency pipeline verification full-size plots

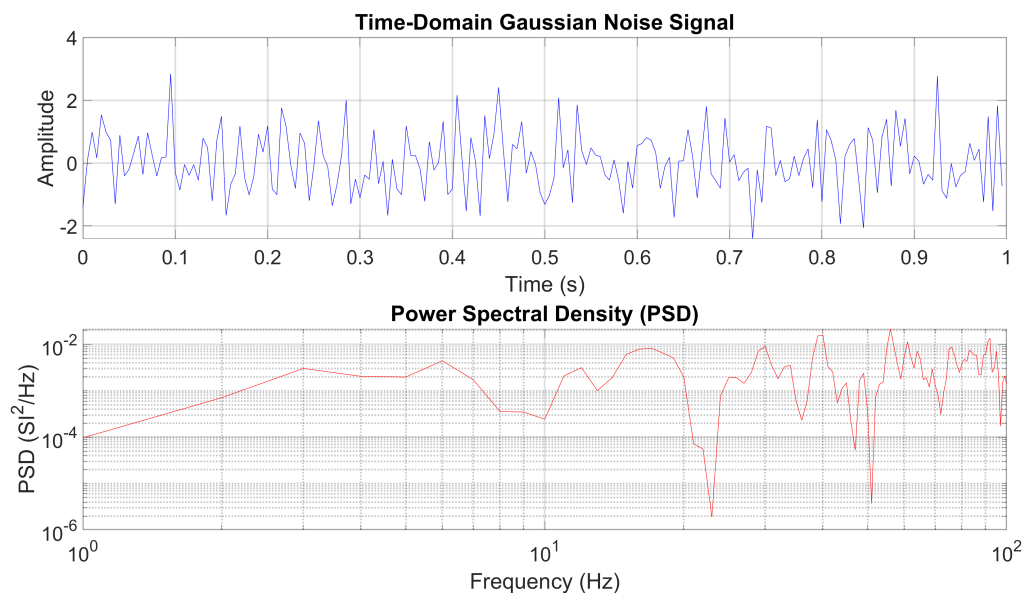


Figure I.1: Input test time domain signal at the top plot, while the bottom plot corresponds to the output power spectral density function of the input test signal. Test signal with a total duration of 1 second.

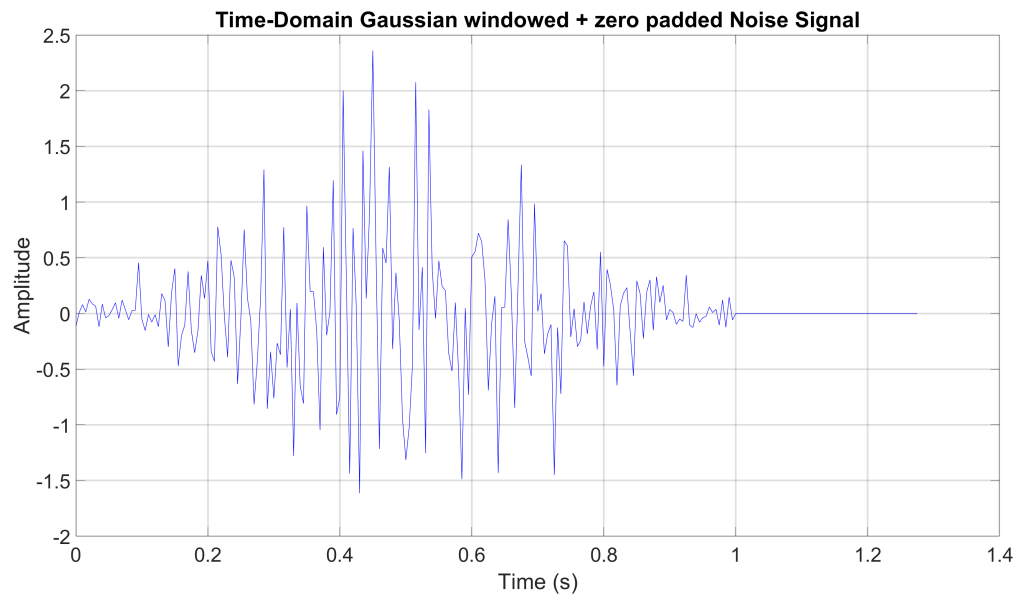


Figure I.2: Input test signal after applying a hamming window function and zero padding, with a signal duration of 1 second.

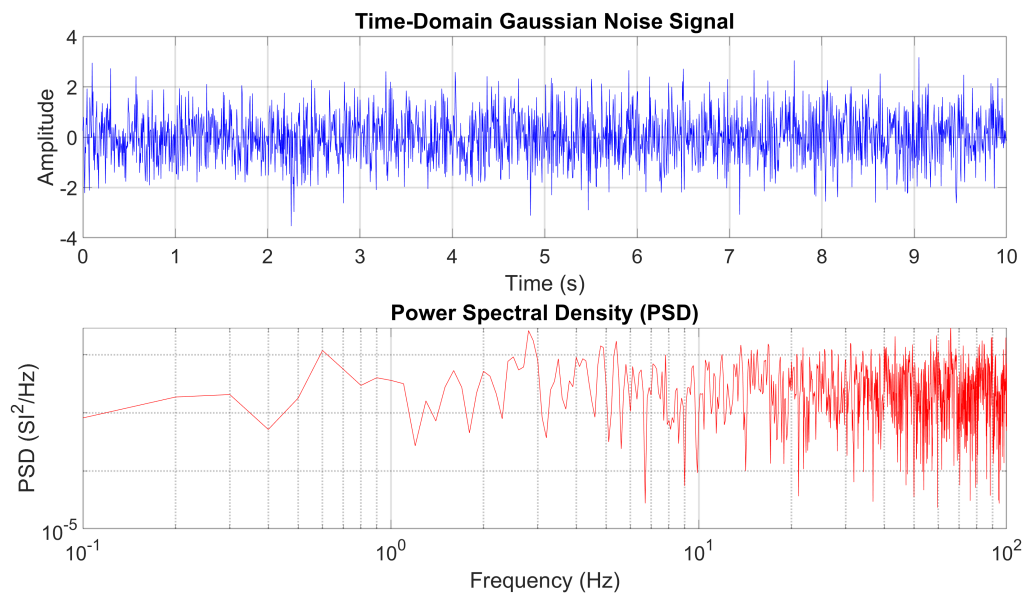


Figure I.3: Input test time domain signal at the top plot, while the bottom plot corresponds to the output power spectral density function of the input test signal. Test signal with a total duration of 10 seconds.

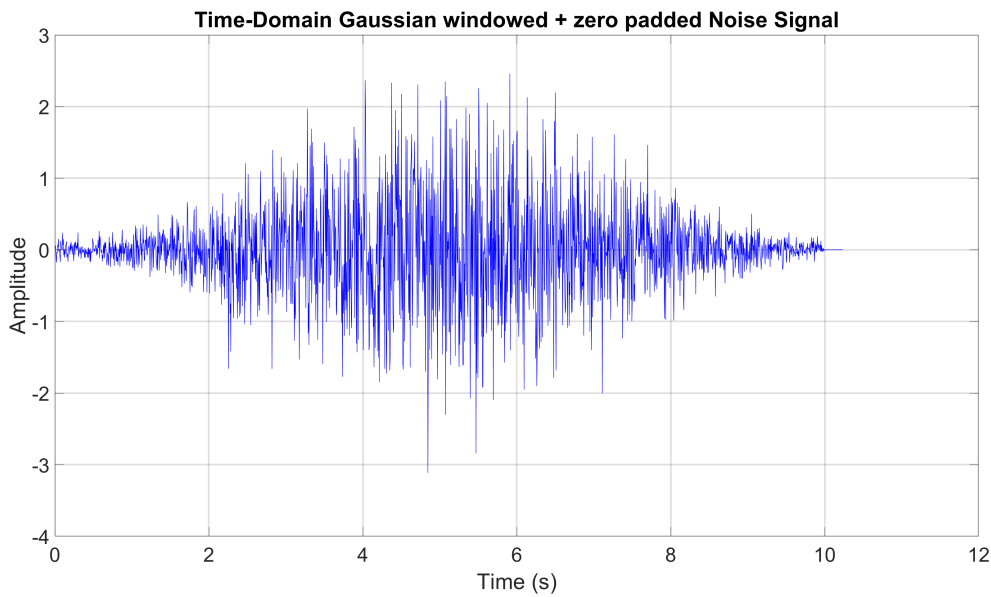


Figure I.4: Input test signal after applying a hamming window function and zero padding, with a signal duration of 10 seconds.

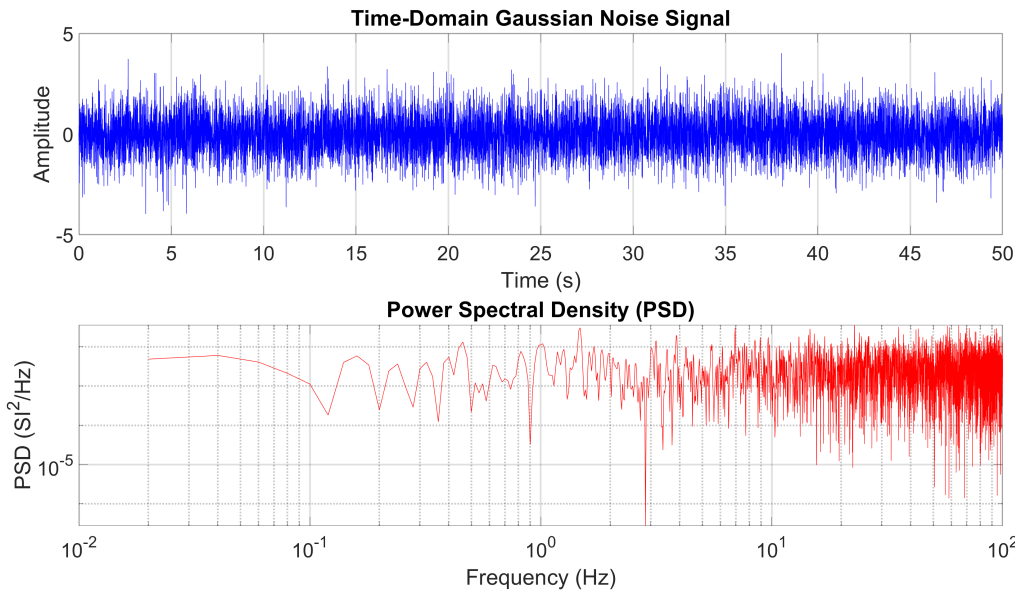


Figure I.5: Input test time domain signal at the top plot, while the bottom plot corresponds to the output power spectral density function of the input test signal. Test signal with a total duration of 50 seconds.

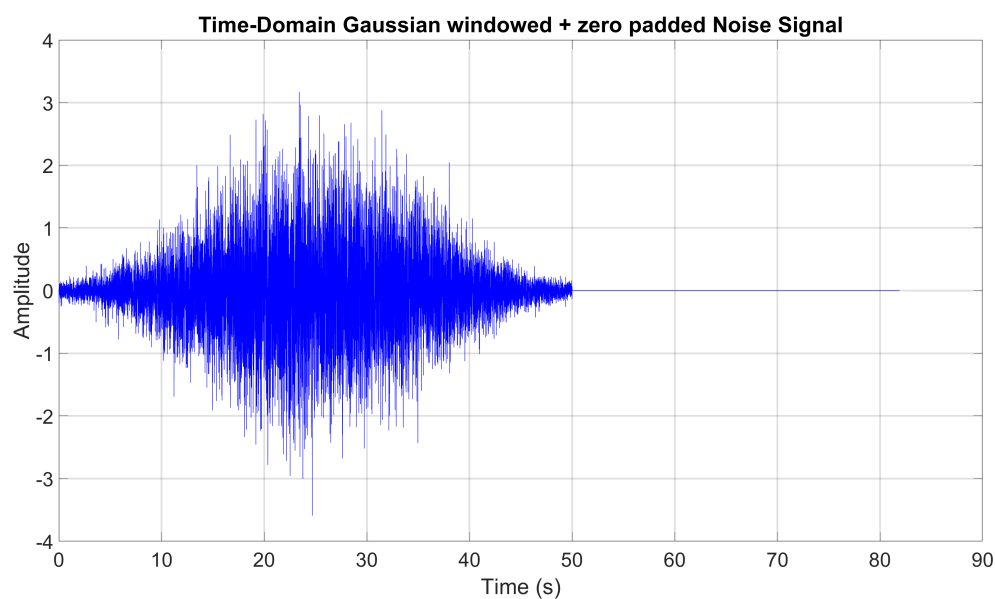


Figure I.6: Input test signal after applying a hamming window function and zero padding, with a signal duration of 50 seconds.

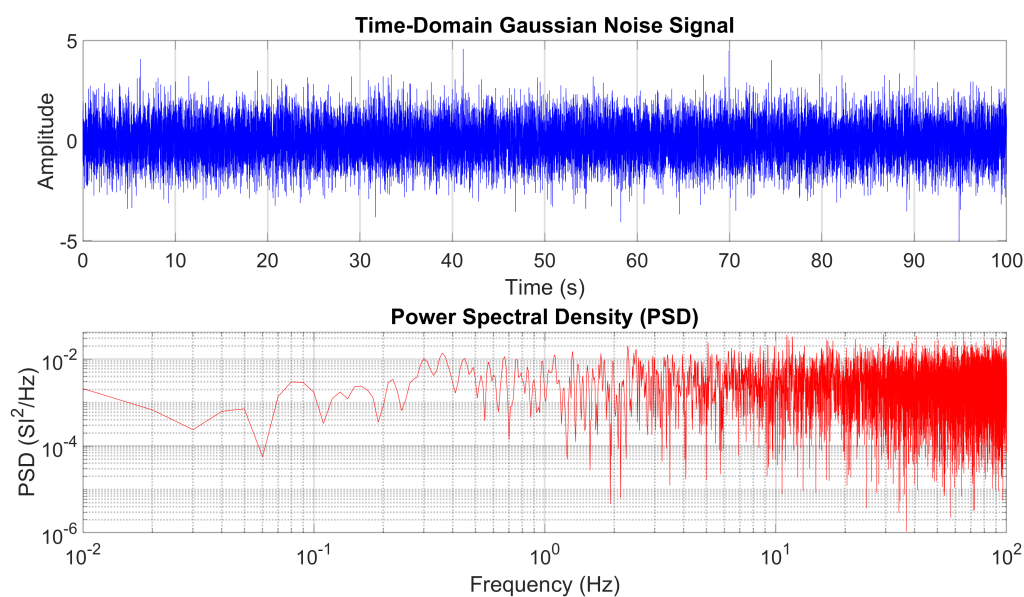


Figure I.7: Input test time domain signal at the top plot, while the bottom plot corresponds to the output power spectral density function of the input test signal. Test signal with a total duration of 100 seconds.

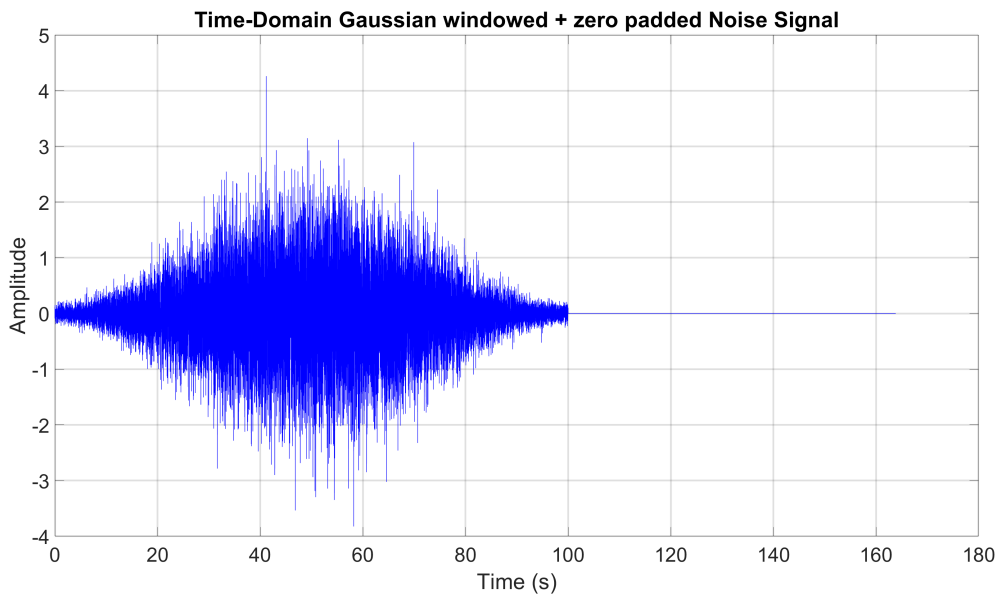


Figure I.8: Input test signal after applying a hamming window function and zero padding, with a signal duration of 100 seconds.

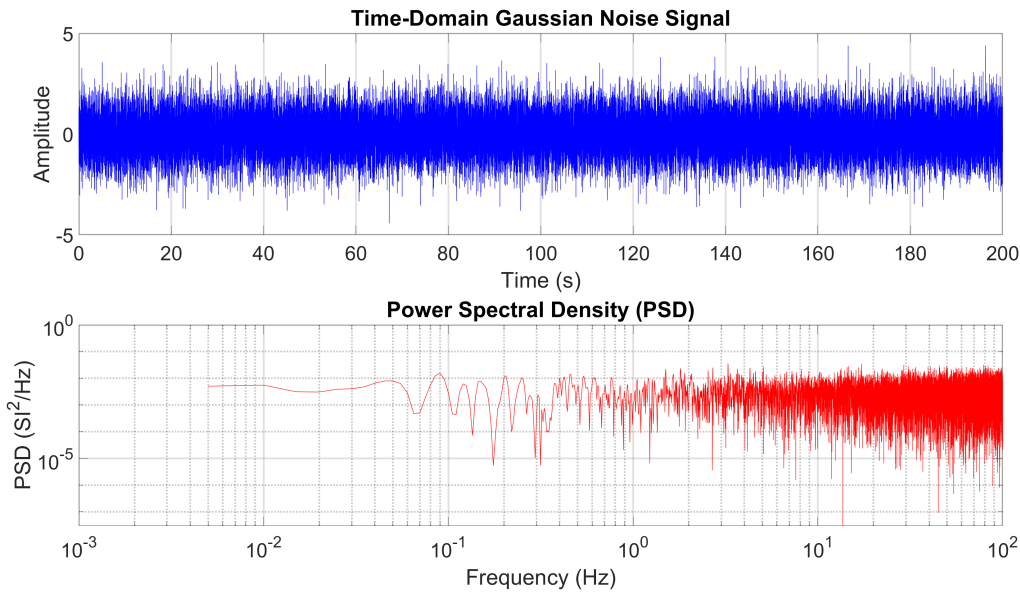


Figure I.9: Input test time domain signal at the top plot, while the bottom plot corresponds to the output power spectral density function of the input test signal. Test signal with a total duration of 200 seconds.

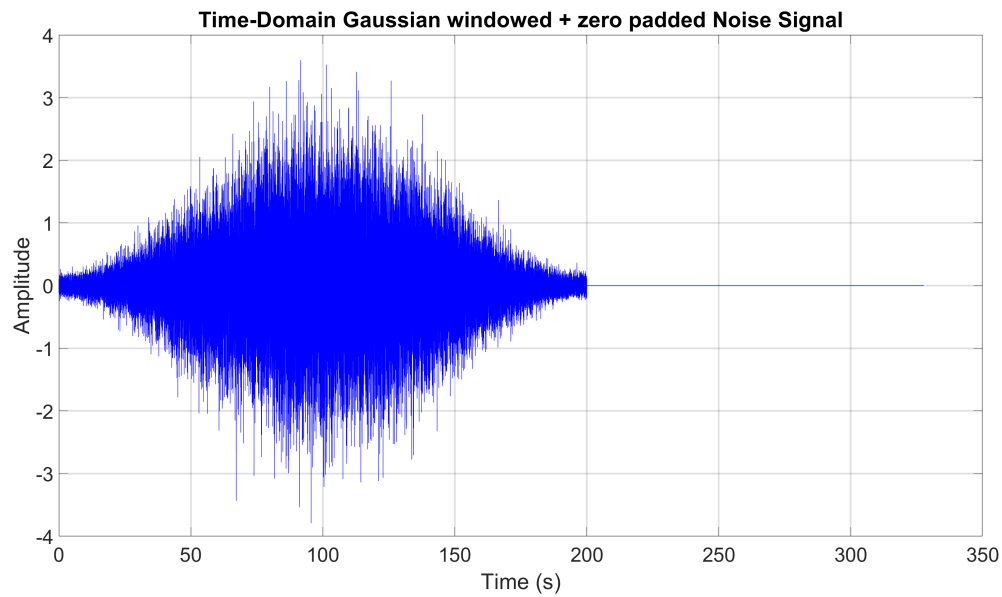


Figure I.10: Input test signal after applying a hamming window function and zero padding, with a signal duration of 200 seconds.

I.1.1. Frequency pipeline verification code

```

1 % This script generates a Gaussian noise signal, computes its Power Spectral Density (PSD),
2 % plots the time-domain signal and PSD, and estimates the original standard deviation
3 % from the PSD.
4
5 close all;
6 set(groot, 'DefaultAxesFontSize', 32);
7 set(groot, 'DefaultAxesGridLineWidth', 2.8); set(groot, 'DefaultAxesGridLineWidthMode', '
    manual')
8 set(groot, 'DefaultLineLineWidth', 0.8)
9
10
11 % Parameters
12 fs = 200; % Sampling frequency in Hz
13 t_end = 200; % duration of simulation in sec
14 disp(['Simulation duration = ', num2str(t_end), ' sec'])
15 T = 0:1/fs:t_end-1/fs; % time vector in seconds
16 N = length(T); % Number of samples
17 N_original = N;
18 sigma = 1; % Desired standard deviation of Gaussian noise [-]
19
20 % Step 1: Generate Gaussian Noise Signal
21 % Generate white Gaussian noise with zero mean and specified standard deviation
22 time_signal = sigma * randn(N,1);
23
24 % Step 2: Apply Windowing and Zero-Padding
25 [processed_signal, windowed_signal, time_vector_padded] = preprocess_for_dft(time_signal, fs);
26
27 % Step 3: Perform Discrete Fourier Transform (DFT)
28 [frequency_vector, psd_norm_amplitude, psd_imu_data, dft_data] =
    normalize_amplitude_dft_and_psd_creation(processed_signal, fs, N_original);
29
30 % Step 4: Plot Time-Domain Signal and PSD
31 figure;
32 % Subplot 1: Time-Domain Signal
33 subplot(2,1,1);
34 plot(T, time_signal, 'b');
35 xlabel('Time(s)');
36 ylabel('Amplitude');
37 title('Time-Domain Gaussian Noise Signal');
38 grid on;

```

```

39
40 % Subplot 2: Power Spectral Density (Logarithmic Scale)
41 subplot(2,1,2);
42 loglog(frequency_vector, psd_imu_data, 'r');
43 xlabel('Frequency(Hz)');
44 ylabel('PSD(SI^2/Hz)');
45 title('PowerSpectralDensity(PSD)');
46 grid on;
47
48 % plot windowed and zero padded signal
49 figure;
50 plot(time_vector_padded, processed_signal, 'b');
51 xlabel('Time(s)');
52 ylabel('Amplitude');
53 title('Time-DomainGaussianwindowed+zero paddedNoiseSignal');
54 grid on;
55
56 % Step 5: Estimate Standard Deviation from PSD
57 % The variance of the signal can be estimated by integrating the PSD over all frequencies
58 % first convert the frequency vector to rad/sec
59 frequency_rad = 2*pi*frequency_vector;
60 % integrate to find the variance
61 alt_std_trapez = sqrt(trapz(frequency_rad, psd_norm_amplitude));
62
63 % Display Results
64 fprintf('OriginalStandardDeviation:%f\n', sigma);
65 fprintf('EstimatedStandardDeviationfromPSD(TrapezoidalIntegration):%f\n',
66         alt_std_trapez);
67
68 function [processed_signal,windowed_signal, time_vector_padded] = preprocess_for_dft(signal,
69     sampling_frequency)
70     % Windowing the signal (using Hamming window as an example)
71     N = length(signal);
72     window = hamming(N);
73     windowed_signal = signal .* window;
74     N_pad = 2^nextpow2(N);
75     % Zero-padding to length N_pad
76     processed_signal = [windowed_signal; zeros(N_pad-N,1)];
77     time_vector_padded = (0:length(processed_signal)-1) * (1/sampling_frequency);
78 end
79
80 function [frequency_vector, psd_norm_amplitude, psd_imu_data, dft_data ] =
81     normalize_amplitude_dft_and_psd_creation (imu_data, sampling_frequency, N_original)
82     % Compute the FFT of the signal
83     N = N_original;
84     Y = fft(imu_data);
85     dft_data = Y;
86
87     % Calculate the two-sided spectrum P2 and the single-sided spectrum P1
88     P2 = (abs(Y).^2)/(N*sampling_frequency);
89     P1 = P2(1:N/2+1);
90     P1(2:end-1) = 2*P1(2:end-1);
91
92     % Compute the power spectral density
93     psd_imu_data = P1;
94     frequency_vector = sampling_frequency*(0:(N/2))/N;
95
96     % apply normalization of amplitude
97     % Correct amplitude estimation
98     % Amplitude correction factor for windowing (Hamming window)
99     % The correction factor is computed as the inverse of the window's RMS value
100    correction_factor = sum(hamming(N).^2) / N; % Energy correction factor for Hamming
101    window
102    psd_norm_amplitude = psd_imu_data*correction_factor;
103 end

```

I.2. Dual-stage transfer function control loop models

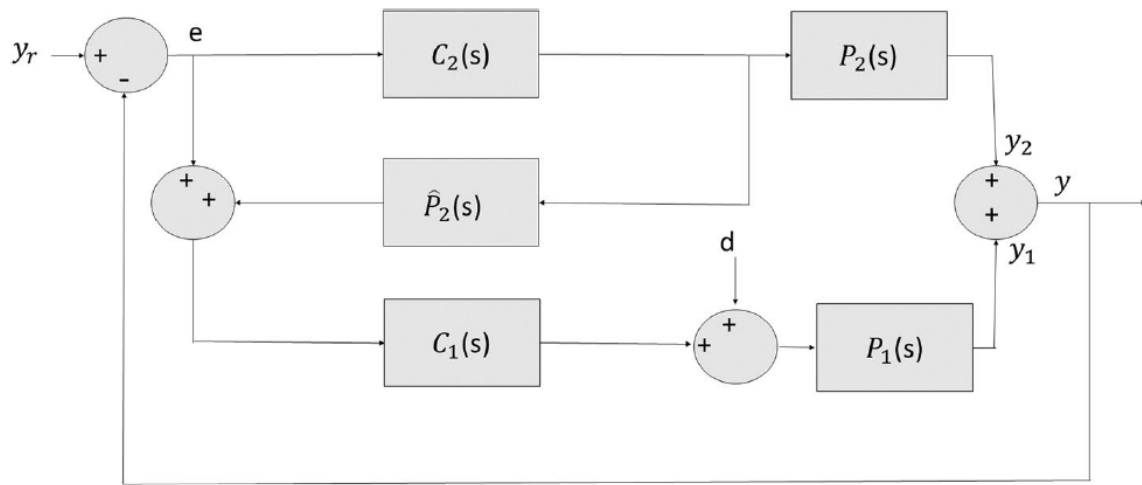


Figure I.11: Dual Stage control design diagram by [6]

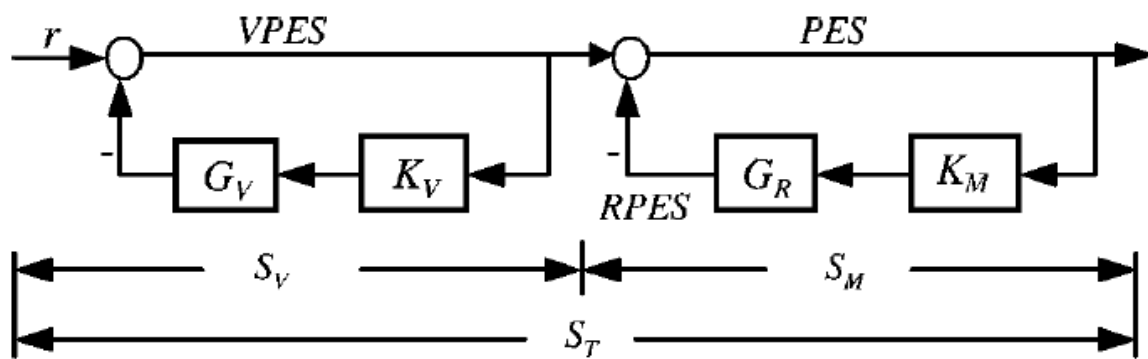


Figure I.12: Sensitivity block diagram by [32]

I.3. Simulink block simulation diagram, time analysis plots and DEB frequency response plots

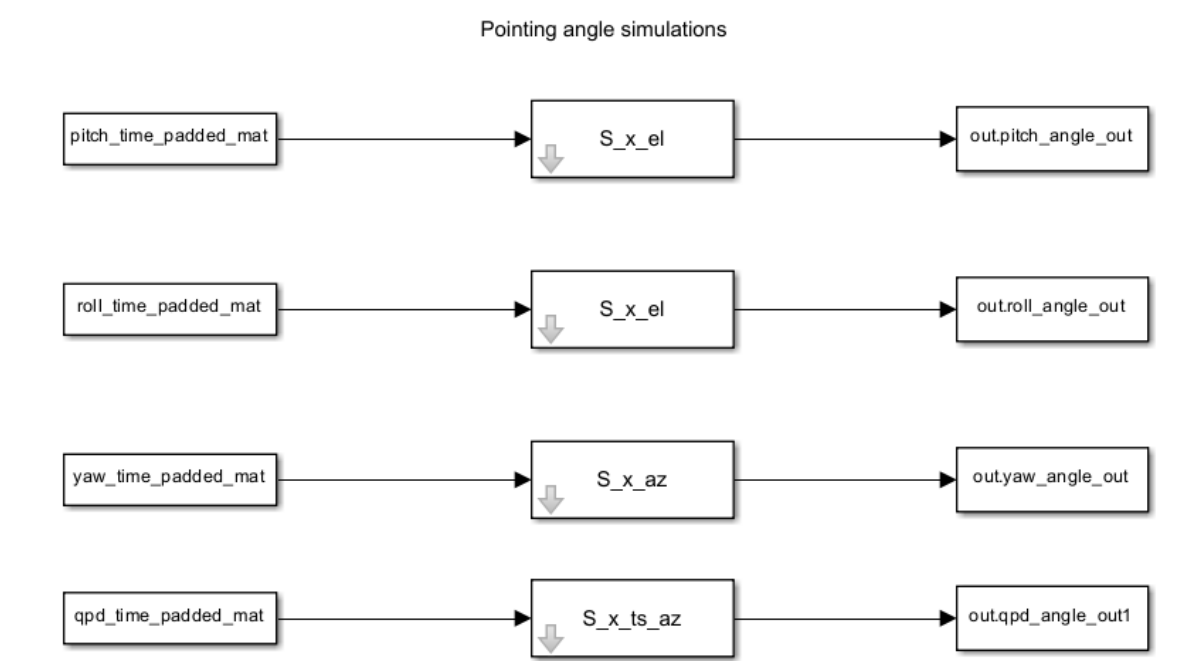


Figure I.13: Simulink model utilized to capture the time analysis response to disturbance spectra

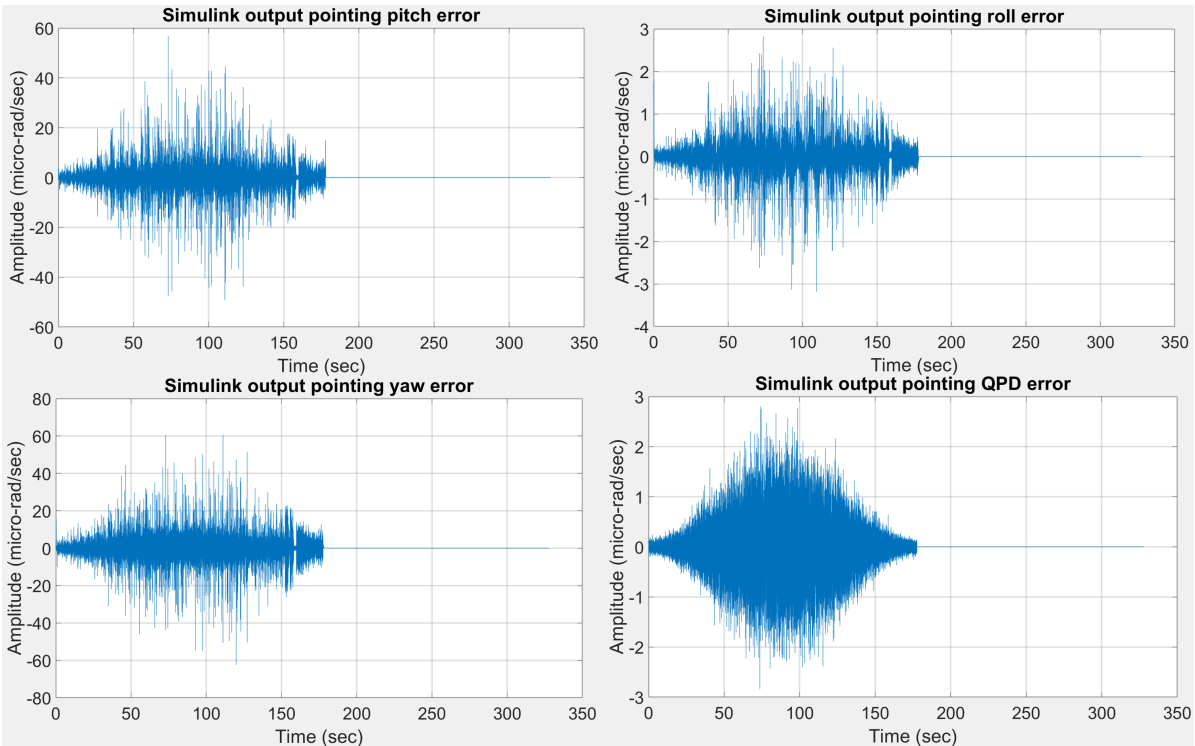


Figure I.14: Time plots of the output signals from the Simulink model angle analysis

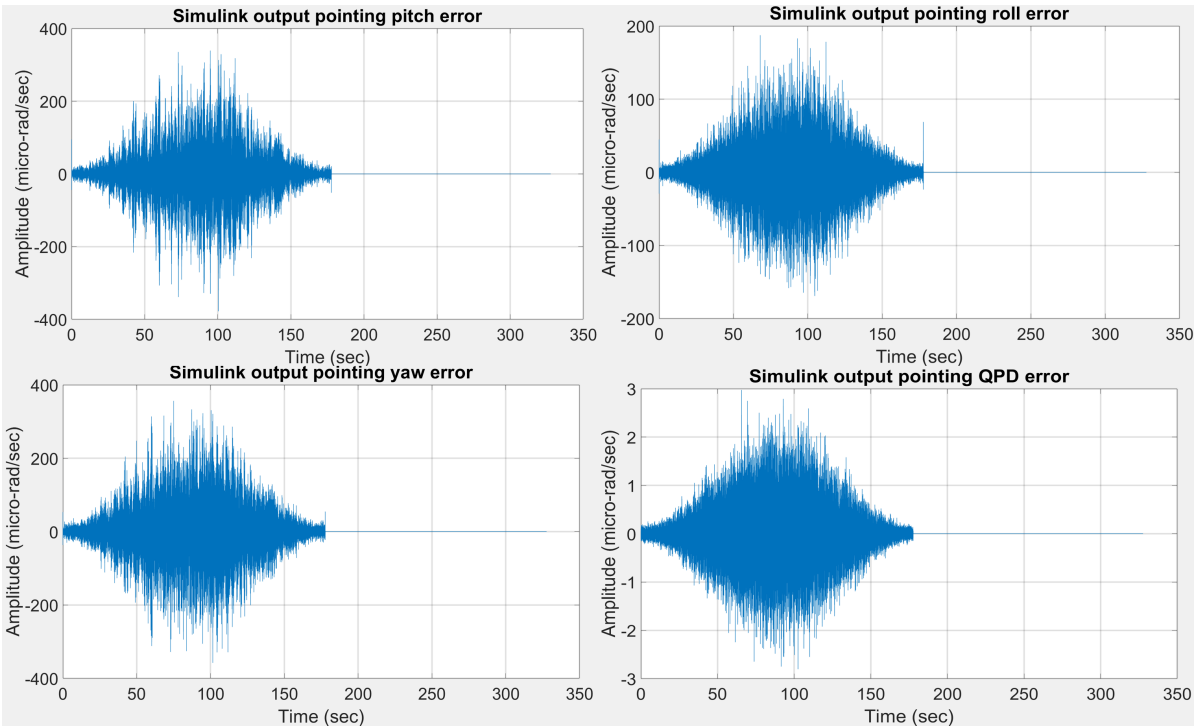


Figure I.15: Time plots of the output signals from the Simulink model angle-rate analysis

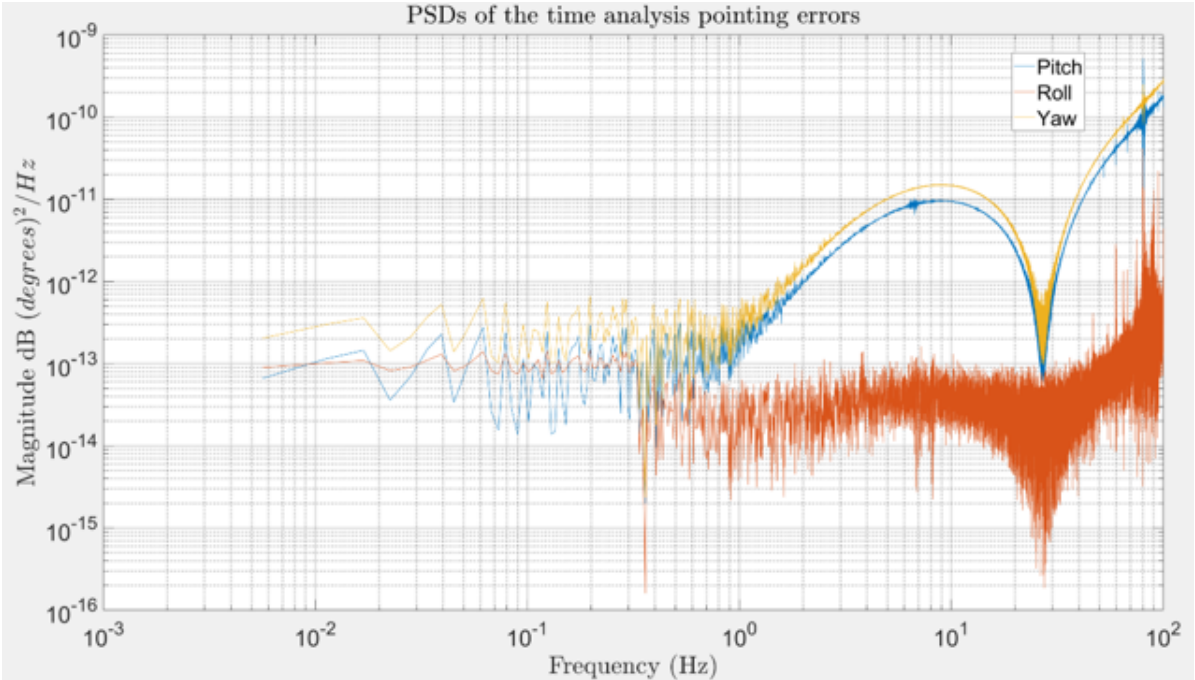


Figure I.16: Power Spectral Density resulting from the pointing angle time analysis

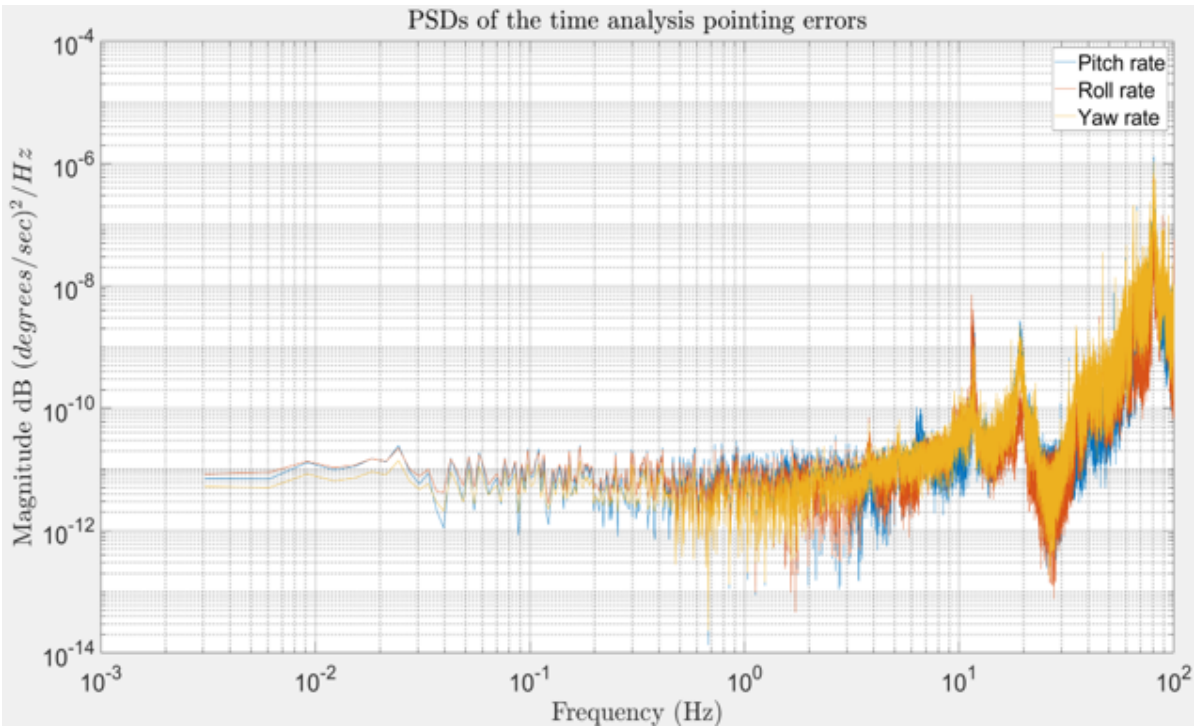


Figure I.17: Power Spectral Density resulting from the pointing angle-rate time analysis

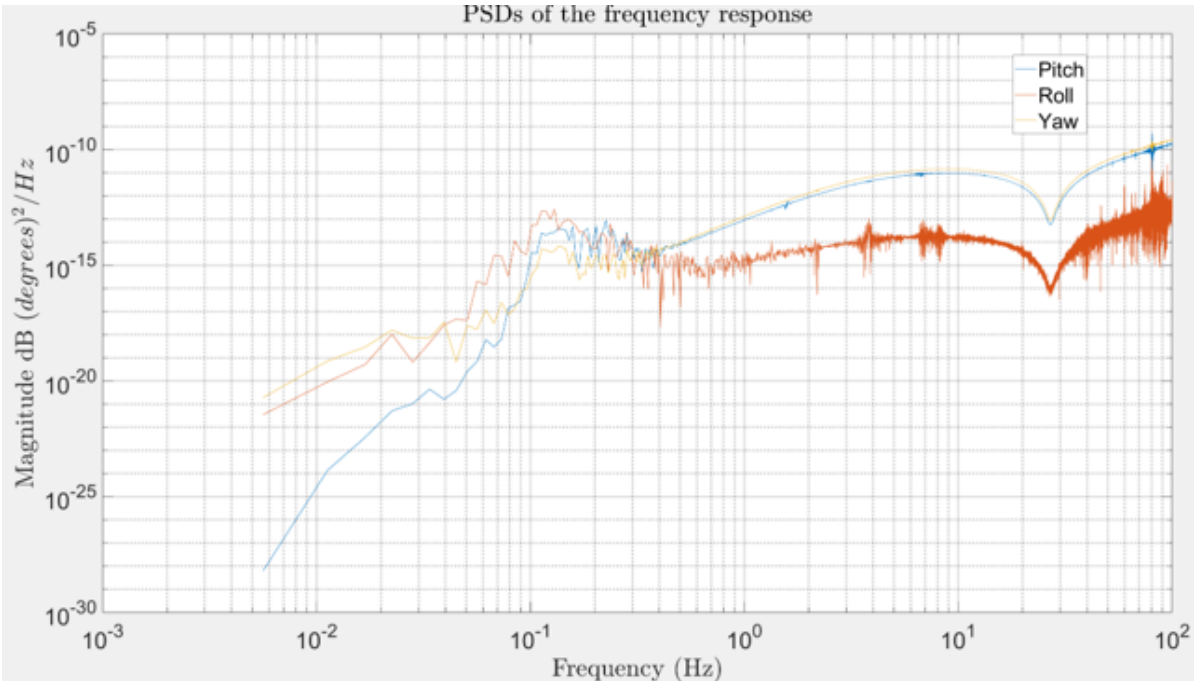


Figure I.18: Power Spectral Density resulting from the pointing angle frequency response

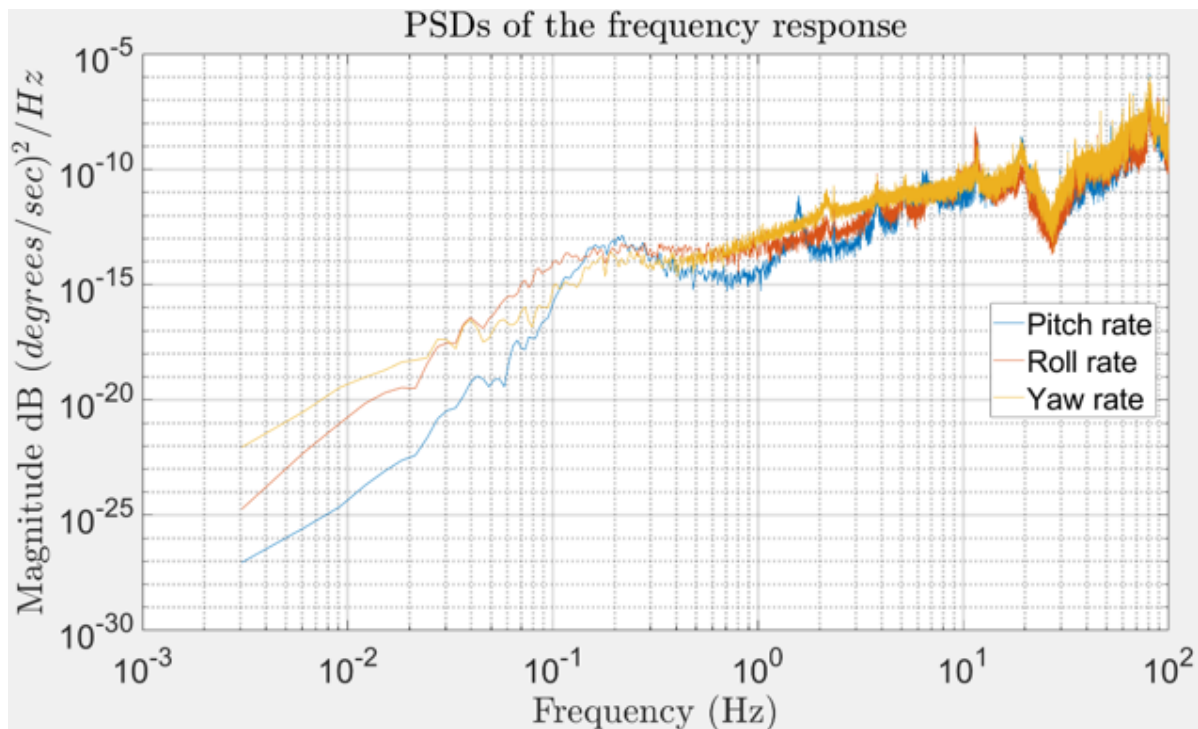


Figure I.19: Power Spectral Density resulting from the pointing angle rate frequency response

I.3.1. DEB frequency response verification code

```

1 % script to convert data from the simulink model into psd's for evaluation of the system
  performance
2 % angle rate simulink data ! angle has it dedicated matlab and simulink model:
  convert_sim_data_to_plots.m
3 % Note run first loading_tf_models.m to get the transfer functions into simulink and as well
  for the DEB analysis
4 set(groot, 'DefaultAxesFontSize', 32);
5 set(groot, 'DefaultAxesGridLineWidth', 2.8); set(groot, 'DefaultAxesGridLineWidthMode', '
  manual')
6 set(groot, 'DefaultLineLineWidth', 0.8)
7
8 close all
9
10 load('deb_analysis_angle_rate_files.mat')
11 % import the general data set into individual vectors
12 avg_psd1 = avg_psd_pitch_rate;
13 avg_psd2 = avg_psd_roll_rate;
14 avg_psd3 = avg_psd_yaw_rate;
15
16 % Frequency analysis modelling parameters
17 step_size = 0.05;
18 sampling_frequency = 200;
19
20 % frequency analysis data and parameters
21 time_vector = out.pitch_angle_rate_out.time;
22 % get the 4 different output pointing errors
23 pointing_error_pitch = out.pitch_angle_rate_out.Data;
24 pointing_error_roll = out.roll_angle_rate_out.Data;
25 pointing_error_yaw = out.yaw_angle_rate_out.Data;
26 pointing_error_qpd = out.qpd_angle_out1.Data;
27
28
29 % creation directly of the psd vector
30 [frequency_vector, y_norm_amplitude, psd_pitch_error, dft_pitch_data] =
  normalize_amplitude_dft_and_psd_creation (pointing_error_pitch, sampling_frequency,
  time_vector);
31 [frequency_vector, y_norm_amplitude, psd_roll_error, dft_roll_data] =

```

```

        normalize_amplitude_dft_and_psd_creation (pointing_error_roll, sampling_frequency,
        time_vector);
32 [frequency_vector, y_norm_amplitude, psd_yaw_error, dft_yaw_data ] =
        normalize_amplitude_dft_and_psd_creation (pointing_error_yaw, sampling_frequency,
        time_vector);
33 [frequency_vector, y_norm_amplitude, psd_qpd_error, dft_qpd_data ] =
        normalize_amplitude_dft_and_psd_creation (pointing_error_qpd, sampling_frequency,
        time_vector);
34
35
36 % conversion of the frequency vector into radians
37 frequency_rad = 2*pi*frequency_vector;
38
39
40 % feed the same input signal into the DEB of the PPAS
41
42 psd_pitch_in = avg_psd1;
43 psd_roll_in = avg_psd2;
44 psd_yaw_in = avg_psd3;
45 % create updated qpd vector:
46 % Statistical error properties
47 mean = 0; % Mean of the positional error
48 std_dev = 8.27E-06; % Standard deviation of the positional error (square root of the variance
    ) QPD G6849-01
49 % sizing properties to match the original reference signal
50 num_samples = length(padded_signal); % Number of data points required to obtain an estimate
    of the PSD SW function
51 % Generate simulated positional error data
52 positional_error = mean + std_dev * randn(1, num_samples);
53 % convert positional error vector into PSD error vector
54 [frequency_vector_qpd, y_norm_amplitude, psd_error_data ] =
        normalize_amplitude_dft_and_psd_creation (positional_error, sampling_frequency,
        padded_signal);
55 % first the frequency vector needs to be converted into rad/sec
56 frequency_padded_rad = frequency_padded;
57 frequency_padded_rad = 2*pi*frequency_padded;
58 psd_qpd_in = psd_error_data;
59
60
61
62 % azimuth response
63 % frequency response of the total sensitivity function
64 [S_x_mag_az, S_x_phase, wout] = bode(S_x_az, frequency_padded_rad);
65 S_x_mag_az = squeeze(S_x_mag_az);
66 % Convert magnitude to array and square it to get |S(jw)|^2
67 S_x_mag_squared_az = S_x_mag_az.^2';
68
69 % tracking sensor response
70 % frequency response of the T.S. sensitivity function
71 [S_x_ts_mag_az, S_x_ts_phase, wout2] = bode(S_x_ts_az, frequency_padded_rad);
72 S_x_ts_mag_az = squeeze(S_x_ts_mag_az);
73 % Convert magnitude to array and square it to get |S(jw)|^2
74 S_x_ts_mag_squared_az = S_x_ts_mag_az.^2';
75
76 % elevation response
77 % frequency response of the total sensitivity function
78 [S_x_mag_el, S_x_phase, wout] = bode(S_x_el, frequency_padded_rad);
79 S_x_mag_el = squeeze(S_x_mag_el);
80 % Convert magnitude to array and square it to get |S(jw)|^2
81 S_x_mag_squared_el = S_x_mag_el.^2';
82
83
84
85 % PSD output of the multiplication between sensitivity function and input PSD
86 S_y_1 = psd_pitch_in.*S_x_mag_squared_el;
87 S_y_2 = psd_roll_in.*S_x_mag_squared_el;
88 S_y_3 = psd_yaw_in.*S_x_mag_squared_az;
89 S_y_4 = psd_qpd_in.*S_x_ts_mag_squared_az;
90
91 % Integrate over the frequency range to find total effect
92 total_effect_1 = sqrt(trapz(wout, S_y_1));

```

```

93 total_effect_2 = sqrt(trapz(wout, S_y_2));
94 total_effect_3 = sqrt(trapz(wout, S_y_3));
95 total_effect_4 = sqrt(trapz(wout, S_y_4));
96 total_effect_total = total_effect_1+total_effect_2+total_effect_3+total_effect_4;
97
98 % Display Standard deviations
99 disp(['\sigma{rms}_DEB_Pitch_rate_Disturbance:', num2str(1e+6*total_effect_1), '\u00b5micro-
    degrees/sec,', num2str(1e+6*(pi/180)*total_effect_1), '\u00b5micro-rads/sec,', ...
    num2str(3600*total_effect_1), '\u00b5arcsec/sec']);
100 disp(['\sigma{rms}_DEB_Roll_rate_Disturbance:', num2str(1e+6*total_effect_2), '\u00b5micro-degrees
    /sec,', num2str(1e+6*(pi/180)*total_effect_2), '\u00b5micro-rads/sec,', ...
    num2str(3600*total_effect_2), '\u00b5arcsec/sec']);
101 disp(['\sigma{rms}_DEB_Yaw_rate_Disturbance:', num2str(1e+6*total_effect_3), '\u00b5micro-degrees/
    sec,', num2str(1e+6*(pi/180)*total_effect_3), '\u00b5micro-rads/sec,', ...
    num2str(3600*total_effect_3), '\u00b5arcsec/sec']);
102 disp(['\sigma{rms}_DEB_QPD_Disturbance:', num2str(1e+6*total_effect_4), '\u00b5micro-degrees/sec,',
    num2str(1e+6*(pi/180)*total_effect_4), '\u00b5micro-rads/sec,', ...
    num2str(3600*total_effect_4), '\u00b5arcsec/sec']);
103 disp(['\sigma{rms}_DEB_total_Disturbance:', num2str(1e+6*total_effect_total), '\u00b5micro-degrees
    /sec,', num2str(1e+6*(pi/180)*total_effect_total), '\u00b5micro-rads/sec,', ...
    num2str(3600*total_effect_total), '\u00b5arcsec/sec']);
104
105 % % plot with all 4 disturbance sources
106 % figure;
107 % loglog(frequency_pad_dist, S_y_1)
108 % hold on
109 % loglog(frequency_pad_dist, S_y_2)
110 % hold on
111 % loglog(frequency_pad_dist, S_y_3)
112 % hold on
113 % loglog(frequency_pad_dist, S_y_4)
114 % hold on
115 % grid on
116 % legend('Pitch', 'Roll', 'Yaw', 'QPD disturbances')
117 % xlabel('Frequency (Hz)', 'Interpreter', 'latex');
118 % ylabel('Magnitude dB $(degrees/sec)^2/Hz$', 'Interpreter', 'latex');
119 % title('PSDs of the frequency response', 'Interpreter', 'latex')
120
121 % plot with only ship-wave disturbance sources
122 figure;
123 loglog(frequency_pad_dist, S_y_1)
124 hold on
125 loglog(frequency_pad_dist, S_y_2)
126 hold on
127 loglog(frequency_pad_dist, S_y_3)
128 hold on
129 loglog(frequency_pad_dist, S_y_4)
130 hold on
131 grid on
132 legend('Pitch_rate', 'Roll_rate', 'Yaw_rate')
133 xlabel('Frequency(Hz)', 'Interpreter', 'latex');
134 ylabel('Magnitude dB $(degrees/sec)^2/Hz$', 'Interpreter', 'latex');
135 title('PSDs of the frequency response', 'Interpreter', 'latex')
136
137
138
139 % integration for determining total error in the setup
140 total_effect_1 = sqrt(trapz(frequency_rad, psd_pitch_error));
141 total_effect_2 = sqrt(trapz(frequency_rad, psd_roll_error));
142 total_effect_3 = sqrt(trapz(frequency_rad, psd_yaw_error));
143 total_effect_4 = sqrt(trapz(frequency_rad, psd_qpd_error));
144 total_effect_total = total_effect_1+total_effect_2+total_effect_3+total_effect_4;
145
146 % Display Standard deviations
147 disp(['time_Pitch_rate_error:', num2str(1e+6*total_effect_1), '\u00b5micro-degrees/sec,', num2str(
    1e+6*(pi/180)*total_effect_1), '\u00b5micro-rads/sec,', ...
    num2str(3600*total_effect_1), '\u00b5arcsec/sec']);
148 disp(['time_Roll_rate_error:', num2str(1e+6*total_effect_2), '\u00b5micro-degrees/sec,', num2str(1
    e+6*(pi/180)*total_effect_2), '\u00b5micro-rads/sec,', ...
    num2str(3600*total_effect_2), '\u00b5arcsec/sec']);
149 disp(['time_Yaw_rate_error:', num2str(1e+6*total_effect_3), '\u00b5micro-degrees/sec,', num2str(1e
    +6*(pi/180)*total_effect_3), '\u00b5micro-rads/sec,', ...
    num2str(3600*total_effect_3), '\u00b5arcsec/sec']);
150 disp(['time_QPD_rate_error:', num2str(1e+6*total_effect_4), '\u00b5micro-degrees/sec,', num2str(1e
    +6*(pi/180)*total_effect_4), '\u00b5micro-rads/sec,', ...
    num2str(3600*total_effect_4), '\u00b5arcsec/sec']);
151

```

```

155 disp(['time_total_rate_error:', num2str(1e+6*total_effect_total), 'micro-degrees/sec,',
156       num2str(1e+6*(pi/180)*total_effect_total), 'micro-rads/sec,', ...
157       num2str(3600*total_effect_total), 'arcsec/sec']);
158 figure;
159 loglog(frequency_vector, psd_pitch_error)
160 hold on
161 loglog(frequency_vector, psd_roll_error)
162 hold on
163 loglog(frequency_vector, psd_yaw_error)
164 grid on
165 legend('Pitch_rate', 'Roll_rate', 'Yaw_rate')
166 xlabel('Frequency (Hz)', 'Interpreter', 'latex');
167 ylabel('Magnitude dB $(degrees/sec)^2/Hz$', 'Interpreter', 'latex');
168 title('PSDs of the time analysis pointing errors', 'Interpreter', 'latex')

```