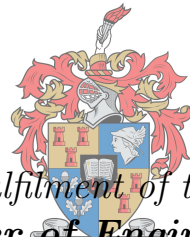


Infrared Horizon Sensor for CubeSat Implementation

by

Jurie Hendrik Wessels



Thesis presented in partial fulfilment of the requirements for the degree

***Master of Engineering
(Electronic)***

in the Faculty of Engineering at Stellenbosch University.

UNIVERSITEIT
STELLENBOSCH
UNIVERSITY

100
1918 · 2018

Advisor: Prof. W.H. Steyn

March 2018

Plagiaatverklaring / Plagiarism Declaration

1. Plagiaat is die oorneem en gebruik van die idees, materiaal en ander intellektuele eiendom van ander persone asof dit jou eie werk is. *Plagiarism is the use of ideas, material and other intellectual property of another's work and to present it as my own.*
2. Ek erken dat die pleeg van plagiaat 'n strafbare oortreding is aangesien dit 'n vorm van diefstal is. *I agree that plagiarism is a punishable offence because it constitutes theft.*
3. Ek verstaan ook dat direkte vertalings plagiaat is. *I also understand that direct translations are plagiarism.*
4. Dienooreenkomstig is alle aanhalings en bydraes vanuit enige bron (ingesluit die internet) volledig verwys (erken). Ek erken dat die woordelike aanhaal van teks sonder aanhalingstekens (selfs al word die bron volledig erken) plagiaat is. *Accordingly all quotations and contributions from any source whatsoever (including the internet) have been cited fully. I understand that the reproduction of text without quotation marks (even when the source is cited) is plagiarism.*
5. Ek verklaar dat die werk in hierdie skryfstuk vervat, behalwe waar anders aangedui, my eie oorspronklike werk is en dat ek dit nie vantevore in die geheel of gedeeltelik ingehandig het vir bepunting in hierdie module/werkstuk of 'n ander module/werkstuk nie. *I declare that the work contained in this assignment, except where otherwise stated, is my original work and that I have not previously (in its entirety or in part) submitted it for grading in this module/assignment or another module/assignment.*

March 2018

Abstract

Attitude determination is essential in satellite design, as it directly affects the pointing ability of the satellite. In the CubeSat industry there exists a need for high accuracy attitude sensors that are low-power and low-cost. Currently, horizon sensors are a desirable option, but with recent growth in thermopile technology, it is possible to create horizon sensors that operate in the infrared spectrum offering tremendous benefits.

This study focusses on the design, development, and evaluation of such an infrared horizon sensor. This includes the circuit and PCB design, software development and embedded implementation, as well as the creation of simulation/emulation environments for sensor calibration and evaluation. The critical limitation of this study is the extremely low resolution of the infrared camera (32×31 pixels) from which the horizon location should be determined to a sub-pixel accuracy. This limitation is overcome by calculating the gradient image with use of a Sobel Operator, after which the sub-pixel local extrema is determined by approximating a parabola shape on the horizon edge.

In conclusion, a robust, low-power and low-cost sensor were developed, that is implementable on a CubeSat. This sensor delivers a worst case elevation accuracy of 0.075° with added noise of $\sigma = 0.023^\circ$. Similarly, the rotation measurement delivers a worst case accuracy of 0.39° with added noise of $\sigma = 0.14^\circ$. This satisfies the initial goal of reaching 0.1° elevation measurement accuracy.

Uittreksel

Oriëntasiekennis is belangrik in satellietontwerp, aangesien dit sy oriëntasiebeheer direk beïnvloed. In die CubeSat-industrie bestaan daar 'n behoefte aan hoë akkuraatheid oriëntasiesensors wat lae krag en lae koste is. Tans is horison sensors 'n wenslike opsie, maar met 'n onlangse groei in termopiel tegnologie is dit moontlik om horison sensors te ontwerp wat in die infrarooi spektrum funksioneer, wat enorme voordele bied.

Hierdie studie fokus op die ontwerp, ontwikkeling en evaluering van so 'n infrarooi horison sensor. Dit sluit in die volledige stroombaan ontwerp, sagteware-ontwikkeling en mikroverwerker implementering, asook die skep van simulasië- en emulasieomgewings vir sensorkalibrasie en evaluering. Die kritiese beperking van hierdie studie is die uiters lae resolusie van die infrarooi kamera (32×31 beeldpunte (Engels: pixels)) waarvan die horison-lokasie bepaal moet word vir 'n sub-beeldpunt-akkuraatheid. Hierdie beperking word oorkom deur die gradientbeeld te bereken deur gebruik te maak van 'n Sobel Operator, waarna die sub-beeldpunt plaaslike ekstrem bepaal word deur 'n paraboolvorm aan die horison rand te pas.

Ten slotte is 'n robuuste, laekrag- en laekostesensor ontwikkel wat op 'n CubeSat implementeerbaar is. Hierdie sensor lewer 'n slegste geval elevasie meting akkuraatheid van 0.075° met bygevoegde geraas van $\sigma = 0.023^\circ$. Net so lewer die rotasie meting 'n slegste geval akkuraatheid van 0.39° met bygevoegde geraas van $\sigma = 0.14^\circ$. Dit voldoen aan die aanvanklike akkuraatheid doelwit om 'n 0.1° elevasiehoek te meet.

Acknowledgements

I would like to acknowledge and thank the following persons and institutes for their support and guidance:

- Prof. W.H. Steyn for the guidance, insight, mentorship and helping me develop my own ideas.
- Mike-Alec Kearney and his team at CubeSpace for answering all my curiosities with zealous.
- My parents, Abrie and Aletta Wessels, for their constant input of wisdom, motivation, inspiration and love.
- The woman I love, Jodi Moore, for tireless help in editing the language of this document. Your endless love and support is unfathomable.

Contents

Plagiaatverklaring / Plagiarism Declaration	i
Abstract	ii
Uittreksel	ii
Acknowledgements	iii
Contents	iv
List of Figures	vii
List of Tables	x
Nomenclature	xi
1 Introduction	1
1.1 Background	1
1.2 Research Question	1
1.3 Scope of this study	2
1.4 Document Outline	2
2 Literature Review	4
2.1 Attitude Sensors on CubeSats	4
2.2 Infrared Horizon Sensor Technology	4
2.3 Previous Research on Infrared Horizon Sensors	6
2.3.1 Infrared Horizon Sensor Modeling for Attitude Determination and Control[1]	6
2.3.2 Attitude Control on the Pico Satellite Solar Cell Testbed-2[2]	8
2.3.3 Attitude Determination using Infrared Earth Horizon Sensors[3, 4]	8
2.4 Possible Improvements and Additions	9
3 Imaging Environment Investigation	10
3.1 Observable Earth Shape	10
3.2 Effect of Earth Oblateness and Infrared Horizon	12
3.3 Signal to Noise Ratio	14
3.3.1 Worst Case Theoretical SNR	15
3.3.2 Measured SNR	16
3.3.3 Effect of Averaging on SNR	19
3.4 Motion Blur Quantification	20
3.5 Ground Test Environment	21
3.5.1 Required Testing Environment	21
3.5.2 Detailed Camera and Earth Disc Setup	22
3.5.3 Impact of Misalignments	23
3.5.4 Non-Uniform Steel Plate Heating	24
3.5.5 Computer Interface and Logging Software	25
3.6 Horizon Simulation Software	26
3.6.1 Determining Disc Parameters	27
3.6.2 Intersection Determination between Rectangle and Disc	28
3.6.3 Determining Edge Pixel Intensity	28
3.6.4 Edge Smoothing	29
3.6.5 Adding Noise	29
3.7 Discussion	30
4 Hardware Design	31
4.1 Requirements and Overview	31
4.2 Component Selection	32
4.2.1 Infrared Sensor	32
4.2.2 Microprocessor	33
4.2.3 External Memory	33
4.3 Circuit Design	33
4.3.1 Infrared Camera Interface	33

4.3.2	Analogue Low Pass Filter Design	34
4.3.3	Power Planes and PCB Design	36
4.3.4	FRAM Interface Design	36
4.4	Power Consumption	37
4.5	Disussion	38
5	Thermal Pixel Calibration	39
5.1	Detailed Explanation	39
5.2	Calibration Steps	40
5.2.1	Determining Offset Voltage	41
5.2.2	Determining Housing Temperature	41
5.2.3	Determining Object Temperature	42
5.3	Results	42
5.3.1	Determining Offset Voltage	42
5.3.2	Determining Housing Temperature	43
5.3.3	Determining Object Temperature	44
5.4	Discussion	45
6	Horizon Location Extraction	46
6.1	Available Edge Detection Techniques	46
6.1.1	Super Resolution	47
6.1.2	Linear Approximation	47
6.1.3	Neighbouring Pixel Utilisation	48
6.1.4	Center of Mass	49
6.1.5	Local Extrema	50
6.2	Comparing Edge Detection Techniques	51
6.2.1	Accuracy	51
6.2.2	Robustness Against Noise	54
6.2.3	Software Implementation	54
6.2.4	Summary	54
6.3	Available Scanning Techniques	56
6.3.1	Rotation Pre-Estimation using Center of Intensities	57
6.3.2	Simple Column Scanning	58
6.3.3	Total Scanning	59
6.3.4	Predictive Scanning	59
6.3.5	Dynamic Scanning	60
6.4	Comparing Scanning Techniques	60
6.4.1	Accuracy	61
6.4.2	Edge Coordinate Quantity and Range	63
6.4.3	Implementation Simplicity and Robustness	63
6.4.4	Summary	65
6.5	Shape Fitting Techniques	65
6.5.1	Straight Line Fit	66
6.5.2	Polynomial Fit	68
6.5.3	Circular Fit	69
6.6	Comparing Shape Extraction Techniques	71
6.6.1	Accuracy	71
6.6.2	Robustness Against Noise	73
6.6.3	Implementation Simplicity	74
6.6.4	Summary	75
6.7	Discussion	75
7	Lens Distortion Correction	76
7.1	Overview	76
7.2	Modeling Lens Distortion	77
7.2.1	Overview	77
7.2.2	Methodology	77
7.2.3	Calculating Polynomial Intersections	78
7.2.4	Results	79
7.3	Lens Distortion Correction	82
7.3.1	Lens Distortion Investigation	83

7.3.2	Different Distortion Correction Models	83
7.3.3	Fitting Lens Distortion Model	85
7.3.4	Comparing Distortion Correction Models	85
7.4	Discussion	88
8	Software Implementation	89
8.1	Software Overview	89
8.2	Detailed Descriptions	89
8.2.1	Storing of Variables	89
8.2.2	Communication Protocol	90
8.2.3	Pixel Acquisition and Calibration	91
8.2.4	Sobel Edge Filter	92
8.2.5	Predictive Edge Detection	94
8.2.6	Least Squares Circle Fit	94
8.2.7	Lens Distortion Correction and Post Calibration	95
8.3	Program Timing	96
8.4	Discussion	96
9	Ground Test Results	97
9.1	Methodology	97
9.2	Misalignment Investigation and Correction	98
9.2.1	Calculating Misalignments	99
9.2.2	Correcting Misalignments	100
9.3	MATLAB Results	101
9.3.1	Initial Results	101
9.3.2	Post Calibration	103
9.3.3	Calibrated Results	104
9.3.4	Robustness Against Noise	106
9.4	Full Onboard Results	107
9.4.1	Initial Results	107
9.4.2	Calibrated Results	109
9.4.3	Robustness Against Noise	111
9.4.4	Robustness Against Changing Environment	112
9.5	Discussion	113
10	Conclusion & Recommendations	115
10.1	Conclusion	115
10.2	Recommendations and Future Work	115
10.2.1	Better Ground Test Setup	115
10.2.2	Moon & Sun Rejection	116
10.2.3	utilise FRAM for Image Storing	116
10.2.4	Increase Camera Resolution	116
	Appendix A Sobel Operator	118
	Appendix B Telecommands and Telemetry Requests	120
B.1	Telecommands	121
B.2	Telemetry Requests	122
	References	123

List of Figures

2.1	Earth viewed from space in the visible spectrum (left) and the infrared spectrum (right)[5] . . .	5
2.2	Earth's infrared spectrum for two extreme conditions, namely Sahara and the Antarctic[1] . . .	6
2.3	Example of seasonal variation of measured horizon radiance profiles for various latitudes[1]	7
2.4	Visualisation of the infrared nadir sensor on the PSSCT-2[2]	8
2.5	Visualisation of the MAI-SES infrared nadir sensor[3, 4]	9
3.1	Camera mounting and pointing direction	10
3.2	Earth disc definition	11
3.3	Example of expected horizon with a camera elevation of 20° and rotation of 30°	12
3.4	Illustration of the ellipsoid Earth (solid line) relative to the spherical Earth (dotted line) shown with an approximation of a horizon sensor's maximum measurement error (ϵ)	13
3.5	Expected view of the mean horizon profile of the summer in 1966 from a low-resolution camera	14
3.6	Definition of signal to noise ratio	14
3.7	Example of measured edge between <i>warm</i> and <i>cold</i> pixels	17
3.8	Example of a fitted polynomial to <i>cold</i> and <i>warm</i> pixel's signal over time (amplification = 3)	18
3.9	Histogram of noise present for <i>warm</i> and <i>cold</i> pixels for both amplification settings	18
3.10	Drop-off in SNR due to non-uniform heating of steel plate	19
3.11	Horizon shift vs. satellite angular rate over a 108 <i>ms</i> time period	21
3.12	Overview of ground test setup	22
3.13	Detailed drawings showing ground testing setup	23
3.14	Temperature distribution on heated steel plate	24
3.15	Example of ground test image before, and after, post processing. ($\phi = 45^\circ, \theta = 15^\circ$)	25
3.16	Screenshot of Computer Interface	26
3.17	Flowchart depicting steps used to create simulated horizon	27
3.18	Calculating pixel area covered by the disc utilising the straight line approximation	28
3.19	Examples of high and low contrast simulated edges	29
3.20	Example of simulated image of Earth disc section with added noise during ground test conditions	30
4.1	Overview of hardware design	31
4.2	Modified 2 nd -order Low Pass Filter	34
4.3	Results of simulated Low Pass Filter	35
4.4	Measured performance of the low pass filter. The dashed lines (V_{SAM}) represents the start of the serial analogue output	36
4.5	Image of designed hardware with essential components shown shown in red	38
5.1	Camera setup to determine V_{off}	41
5.2	Temperature response of thermal vacuum chamber, as well as the times sampled	42
5.3	The average pixel values ($V_{AOTu} - V_{ref}$) for varying V_{AORt} values aquired by using the Vacuum Chamber	43
5.4	Calculated V_{obj} values using the approximated V_{off} model over the expected V_{AORt} range	43
5.5	Approximation of V_{AORt} vs. T_S	44
5.6	Temperature response of thermal vacuum chamber, as well as the times sampled	45
5.7	Calculated and approximated k value for pixel (14,14)	45
6.1	Illustration of the superresolution process	47
6.2	Linear approximation pixel definitions	48
6.3	Illustration of 4 pixels with their corresponding created sub-pixel matrices	49
6.4	Center of Mass pixel definitions	49
6.5	Local extrema method pixel definitions	50
6.6	Example of simulated edge for space conditions with an vertical translation of 0.25 pixels and rotation of 30°	52
6.7	Comparing measurement errors of different feature extraction techniques (solid line shows space conditions and dashed shows ground test conditions)	53
6.8	Measurement error statistics due to different noise levels and horizon rotations (solid line shows space conditions and dashed ground test conditions)	55
6.9	Representation of angle of incidence between scanning direction and the horizon	56
6.10	Rotation Pre-Estimation using Center of Intensities	57

6.11	Performance of different variants of the horizon pre-estimation using CoI	58
6.12	Example of simple column scanning's direction on an Earth disc with a rotation of 45°	58
6.13	Example of the total scanning techniques's direction on an Earth disc with a rotation of 45°	59
6.14	Example of the predictive scanning techniques's direction on an Earth disc with a rotation of 45° . The ✖ represents the CoI	60
6.15	Mean (solid) and maximum (dashed) scanning error observed using different scanning techniques	62
6.16	Contour plots representing the edge coordinate quantity as determined with various rotation and elevation angles using different scanning techniques	64
6.17	Determining camera boresight attitude relative to the horizon on the image plane	66
6.18	Straight Line Fit to Horizon	67
6.19	Example of straight line fitting refinement using coordinate distances to line OA . Blue crosses represents all determined edge coordinates, and red circles the masked coordinates	67
6.20	Polynomial Line Fit to Horizon	68
6.21	Straight Line Fit to Horizon	70
6.22	Comparing measurement errors of different shape fitting techniques (solid line shows space conditions and dashed ground test conditions)	72
6.23	Example images used for noise investigation representing the expected ground testing environment with a SNR of 30 dB	73
6.24	Statistics of rotation measurement errors made while estimating a simulated horizon's location under different SNR levels (solid line shows space conditions and dashed ground test conditions)	74
7.1	Examples of pincushion (left) and barrel (right) distortion	76
7.2	Simulating a single square location utilising multiple images	77
7.3	Example showing the original edge (solid) inverted edge (dotted), and the measured edge locations (circles)	78
7.4	Example of four images required to estimate one point on the checkerboard	80
7.5	Example of the fitted polynomials on the imaged edges	81
7.6	Example of combining non-inverted (green) and inverted (blue) points to calculate the checkerboard point location (red)	81
7.7	Simulated checkerboard with fitted polynomials through middle line	82
7.8	Simulated - and corrected - checkerboard with fitted polynomials through middle line	82
7.9	Investigation of lens distortion errors	83
7.10	Comparing performance of distortion models vs. the amount of coefficients, with the (5,3)-order Trapezium Model encircled	86
7.11	Lens distortion corrected coordinates from checkerboard	87
7.12	Heatmap showing error of corrected coordinates	87
8.1	Sample timing of pixel transmission (1 MCLK=1 μs)[6]	91
8.2	Expansion of pixel acquisition and storage timings according to oscilloscope readings	91
8.3	Approximation of magnitude calculation	92
8.4	Comparison between the radial response of normal Sobel Filter and the proposed approximation	93
8.5	Visualisation of dynamic buffer when calculating the Sobel Filter showing only pixel rows	93
8.6	Visualisation of scanning pattern determination	94
9.1	Flow diagram showing the attitude estimation process	97
9.2	Fitted circle centre distribution from multiple images	99
9.3	Method to determine the Z_T misalignment (or offset)	99
9.4	Method to determine the Y_T misalignment (or offset)	100
9.5	Example of applied misalignment correction for horizon with a rotation of $\theta = -15^\circ$. (ϕ and θ represents the horizon elevation and rotation errors respectively)	101
9.6	Attitude estimation obtained through an emulated horizon and MATLAB	102
9.7	Heatmap of attitude estimation errors (in degrees) obtained through the emulated horizon and MATLAB	103
9.8	Attitude estimation obtained through an emulated horizon and MATLAB after post calibration	105
9.9	Sensor operational range's limiting factors over an elevation sweep	106
9.10	Variation in the measurement error for two possible attitudes. (Solid lines show the elevation error and dashed lines the rotation error)	106
9.11	Measurements obtained through the sensor's onboard MCU	108
9.12	Measurement results obtained through processing the sensor's data in MATLAB	109
9.13	Calibrated measurements obtained through the sensor's onboard MCU	110
9.14	Proof of small oscillation on sensor's elevation response	111

9.15	Elevation measurement error over time	112
9.16	Signal magnitude error over time	113
9.17	Example images to show the difference in environment over time (both images are scaled to (a)'s intensity range)	113
10.1	Performance compared to change in camera resolution	117
A.1	Example of Sobel Operator being applied to a grayscale image	119

List of Tables

2.1	Comparison between available CubeSat attitude sensors. Data from [7],[8],[9],[10],[3],[11] respectively, as taken on 5/10/2017.	4
3.1	Worst case theoretical SNR calculation values	16
3.2	Best Case Measured SNR calculation values	19
3.3	Mounting tolerances of steel plate around the different axes	24
3.4	Mounting tolerances of camera on the rotation stage around the different axes	24
4.1	Comparison between available infrared sensors. Data from [6],[12],[13],[14],[15] respectively, as taken on 17/11/2017.	32
4.2	Maximum power usage of different power supplies	37
4.3	Subsystems used by each power supply during measurement	37
4.4	Possible power usage comparison to MAI-SES[3]	37
6.1	Ranking of Feature Extraction Techniques	56
6.2	Ranking of Scanning Techniques	65
6.3	Ranking of Shape Fitting Techniques	75
7.1	Infrared camera's effective FOV	88
8.1	UART transmission decoding	90
8.2	Worst case timing for the entire process over 1 second	96
9.1	Main misalignments present in ground test setup	100
9.2	Error statistics of post calibration results when averaging five images	107
9.3	Error statistics of sensor's the performance	112

Nomenclature

Abbreviations

ADC	Analogue to Digital Converter
ADCU	Analogue to Digital Converter Units
CMOS	Complementary Metal-Oxide-Semiconductor
CoI	Center of Intensities
CoM	Center of Mass
COTS	Commercial of the Shelf
FOV	Field of View
FRAM	Ferroelectric Random Access Memory
HRDB	Horizon Radiance Data Base
IC	Integrated Circuit
LEO	Low Earth Orbit
LPF	Low Pass Filter
MATLAB	Matrix Laboratory
MCU	Microcontroller
MODTRAN	Moderate Resolution Atmospheric Transmission
NASA	National Aeronautics and Space Administration
PCB	Printed Circuit Board
PMP	Parallel Master Port
RAM	Random Access Memory
SNR	Signal to Noise Ratio
SPI	Serial Peripheral Interface
SSP	Sub-Satellite Point
WGS84	World Geodetic System 1984

Camera Specific Variables

T_S	Housing Temperature	Kelvin
T_{amb}	Ambient Temperature	Kelvin
T_{obj}	Object Temperature	Kelvin
V_{AORt}	Housing Temperature Output	Volt
V_{AOTu}	Uncompensated sensor output	Volt
V_{obj}	Thermopile Output Voltage	Volt
V_{off}	Calibration Value	Volt
V_{ref}	Reference Voltage	Volt

Variables

α	Sensor Elevation	pixels
ϕ	Sensor Elevation	degrees
θ	Sensor Rotation	degrees
d	Distance from camera to horizon	Application Dependant
f	Focal Length	meters
H	Satellite Altitude	meters
r	Earth Disc Radius	meters
R_E	Earth Radius	meters
r_p	Earth Disc Radius	pixels

1 Introduction

1.1 Background

Attitude determination is very critical in satellite design. It describes the satellite's ability to determine its orientation in space through onboard sensors, which directly influences the satellite's attitude control capabilities. Attitude determination on modern-day satellites has reached technological maturity, which has led these satellites to accomplish great feats, such as: orbiting and observing our sun[16], landing on comets[17], and observing distant galaxies with large telescopes[18]. These feats are only possible because the technology has grown to a high level of sophistication and maturity since the launch of Sputnik 60 years ago.

With the advent of the CubeSat in 1999, by a collaboration between California Polytechnic State University and Stanford University, a new era of satellite design started. This era envisioned small 10 cm^3 satellites, with a commercial off-the-shelf (COTS) design philosophy, to bring the ability to build and launch satellites to the masses[19]. This new design philosophy created room for innovation and experimentation in all aspects of satellite design, including attitude determination.

Options for attitude determination on CubeSats are currently fairly limited. To achieve high attitude knowledge star trackers are required, but they are big and have high power usage[11]. The less-accurate alternative is the use of visible spectrum horizon sensors, which calculate the satellite's attitude relative to the observed horizon. These devices deliver good accuracies with relatively low power usage but have the significant limitation of not being operable during eclipse. For this reason, infrared horizon sensors were developed to detect the difference in thermal radiation between cold space and the warm Earth. However, the technology for small infrared cameras that follow the COTS design philosophy is still in its infancy.

Typically CubeSat infrared horizon sensors make use of multiple single-element thermal sensors (called thermopiles) to measure the horizon's location[2, 3]. At the time, this was the only infrared sensors viable for CubeSat design and resulted in large and sub-optimal designs. With the aid of recent technological advances COTS infrared sensors have improved. It is now possible to fit multiple thermopiles on a single Microelectromechanical System (MEMS) device to create infrared cameras, albeit only with low resolutions of up to 80×64 pixels[12]. This creates the opportunity for a new trend of infrared sensors by utilising these infrared cameras.

This thesis will focus on utilising this novel infrared camera technology to design an infrared horizon sensor.

1.2 Research Question

The research question that will be answered in this study is:

Is it feasible to develop a low power infrared horizon sensor that can determine a satellite's attitude to within 0.1° from a 500 km orbit?

1.3 Scope of this study

This study focussed on the design and development of an infrared horizon sensor. This includes all aspects of the design, namely: theoretical design, development, testing, and evaluation. The specific tasks focussed on during the course of this study are:

- **Literature Review:** Past work done on infrared horizon sensors. This includes the various sensor's utilized, the spectral windows these devices used, and the shape such an infrared horizon will assume.
- **Horizon Detection Strategy:** Development of a sufficient horizon detection strategy to overcome the limitations associated with typical satellite design and limited hardware.
- **Hardware:** Development of a prototype sensor to test the algorithms with real data. This development will include calibration of the different aspects of the hardware design, namely: lens distortion, pixel behaviour, and measurement offsets.
- **Simulated and Emulated Environments:** Building of a simulation environment to rapidly develop and evaluate various horizon detection techniques. Developing an emulation environment that will test these techniques in a real and non-ideal environment.
- **Fully functioning Horizon Sensor:** Developing a fully functioning sensor by utilising the developed techniques, following typical satellite design philosophies.

A few design constraints are applied to this study, or considered as out-of-scope. These includes:

- **Maximum Operational Range:** Typical CubeSat missions only require fine pointing accuracy when in an Earth-pointing configuration. Therefore it is decided to only estimate the sensor's elevation and rotation angle up to a maximum of 45° .
- **Accurate Infrared Horizon Modelling:** In reality the infrared horizon varies in height with climate, season, location, etc. The modelling and correcting of this variation is not included in this study but is considered during the design process.

1.4 Document Outline

The rest of this document consists of 9 chapters which are followed by appendices. The flow of this document follows the design process used. First it starts with the necessary research and testing environments, followed by the design of the sensor and subsequent algorithms, and finally the testing of the sensor and a conclusion section. This outline and section contents will be discussed in more detail below.

Section 2: Literature Review

Current CubeSat attitude sensors will be investigated with an emphasis on current infrared horizon sensors. Research is also done on the horizon observed in the infrared spectrum. This will serve as a broad overview of the current state of infrared horizon sensing technologies and will create a foundation on which the design is built.

Section 3: Imaging Environment Investigation

The horizon visible from space will be investigated in more detail, and subsequently recreated in software and an emulated environment. The expected signal strength and noise conditions will also be investigated. This will serve as a baseline on which further work will be evaluated.

Section 4: Hardware Design

Hardware is designed and developed, that will test the sensor in an emulated environment. The choice of

components, circuit design and PCB design philosophies will be discussed.

Section 5: Thermal Pixel Calibration

The images captured by the infrared camera have to be calibrated to ensure uniform images under varying temperature conditions. This section will cover a detailed explanation of the calibration process followed, as well as the results obtained.

Section 6: Horizon Location Extraction

Algorithms are developed and evaluated to determine the horizon's location on an image to a sub-pixel accuracy. For each step of the horizon detection method (edge detection, scanning pattern, and shape fitting) multiple techniques will be investigated and compared. This will be the main focus of this study and will, therefore, be thoroughly investigated.

Section 7: Lens Distortion Correction

The infrared camera's lens distortion will be calibrated and corrected in this section. It will discuss the innovative method used to model the distortion, as well as compare various ways to correct for this distortion. Once the distortion is corrected, the final results will be shown, effectively removing lens distortion.

Section 8: Software Implementation

The algorithms developed in Section 6 are implemented on the hardware described in Section 4. The various innovations, optimisations and embedded implementation will be discussed. This will be the final design required to finish the infrared horizon sensor.

Section 9: Ground Test Results

This will be the final results of the designed and built, infrared horizon sensor. It will show the performance of the developed algorithms on emulated data through MATLAB. Here the sensor will be tested as a stand-alone product, and its performance evaluated in an emulated environment.

Section 10: Conclusion & Recommendations

In this section, a summary of the accomplishments and limitations of this study, as well as additional suggestions for future work, will be discussed.

Appendices:

The appendices will include information that will aid in the understanding of certain parts of the design but will not be crucial in the operation of the design.

2 Literature Review

This section will cover relevant literature to create a strong understanding of the current state of CubeSat attitude estimation, infrared sensors, and the infrared horizon. This includes:

- Commercial CubeSat attitude sensors and their limitations.
- COTS infrared sensors limitations and recent improvements.
- Analysis of the infrared horizon done by the National Aeronautics and Space Administration (NASA) from mission experience.

2.1 Attitude Sensors on CubeSats

Attitude estimation is a vital aspect of a CubeSat's design as it directly affects its attitude control ability which is required to point the satellite's science instruments or antennas. This attitude estimation is determined by using a combination of various sensors, such as magnetometers and star trackers. Their design follows the same low-cost and COTS trends used in conventional CubeSat design. These sensors also vary significantly in complexity and price as their designs and goals change, for example only a coarse attitude estimation is required in communication satellites ($< 5^\circ$), whereas Earth imaging requires fine attitude knowledge ($< 0.5^\circ$). Examples of these sensors and their performance are shown in Table 2.1, additional to their respective power requirements, volume, price, etc. As in all nanosatellite design small, low power devices are favoured.

Table 2.1: Comparison between available CubeSat attitude sensors. Data from [7],[8],[9],[10],[3],[11] respectively, as taken on 5/10/2017.

Type	Accuracy	Power	Volume	Weight	Cost
Coarse Sun	5°	None	1.45 cm^3	14 g	Negligible
Magnetometer	1°	500 mW	143 cm^3	120 g	\$ 17 000
Fine Sun	0.5°	50 mW	2.2 cm^3	5 g	\$ 3 300
IR Horizon ¹	0.25°	264 mW	88 cm^3	66 g	\$ 14 900
Horizon ²	0.2°	150 mW	23.6 cm^3	80 g	\$ 5 600
Star Tracker	0.004°	1.5 W	218 cm^3	282 g	\$ 32 500

¹ Requires two modules to yield nadir angle (statistics shown accordingly).

² Power, weight, and price includes sun sensor and supporting hardware.

Is it clear from Table 2.1 that a horizon and/or star tracker is required to achieve high accuracy attitude knowledge ($< 0.5^\circ$). A star tracker delivers very high attitude knowledge but is very expensive regarding power and cost. Therefore, if $< 0.3^\circ$ is sufficient, a less expensive alternative would be the horizon sensors. Therefore, this thesis focussed on developing such an infrared horizon sensor.

2.2 Infrared Horizon Sensor Technology

Horizon sensors are devices that measure the direction to the center of the Earth from a satellite position; this direction is called the nadir angle. This is done by utilising a camera to locate the horizon relative to the

satellite's body axis. If the horizon's location is known, the nadir angle can be calculated, which is then used to determine the satellite's pitch and roll (to determine yaw angle additional attitude knowledge is required).

Initially, such horizon sensors operated in the visible spectral band which mainly utilises sunlight reflected from the Earth's surface (called albedo)[10, 20]. The main advantage of using the visible spectrum is the large intensity difference between the Earth and space. However, visible spectrum sensors are not operational during orbit eclipse, whereas infrared sensors are operational during the entire orbit because they rely on the constant heat radiated from the warm Earth. Figure 2.1 shows an example of Earth viewed from space in the visible and infrared spectrum respectively. The advantages of using the infrared spectrum for horizon sensing is not limited to only full orbit operation, but includes [20, 21]:

- Invisible terminator (boundary between illuminated and eclipsed Earth).
- Seasonal variations in infrared radiation are roughly 3%, which is significantly lower than the 10% variation in albedo radiation.
- Local radiation fluctuations are significantly lower in infrared than in the visible spectrum.
- Solar interference is roughly two orders of magnitude more intensive in the visible spectrum than in the infrared spectrum.

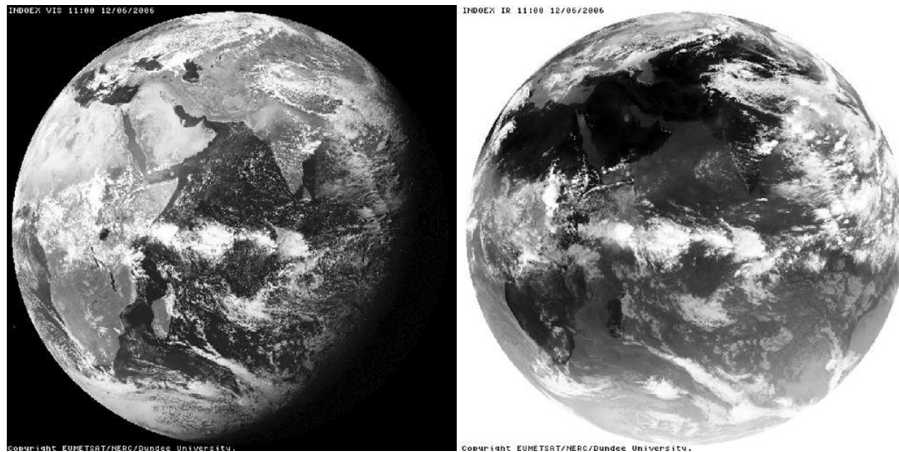


Figure 2.1: Earth viewed from space in the visible spectrum (left) and the infrared spectrum (right)[5]

However, the main disadvantage of using the infrared spectrum is the limited technology available for nano- and picosatellites. Originally large and complex cryogenic bolometers were used for infrared horizon sensing by NASA[1], which offer high-resolution designs. However, as miniaturisation developed with semiconductor technology, such as the CMOS process and micromachining, alternative infrared sensing methods became available. This miniaturisation enabled thermopile technology to use semiconductor technology (instead of previously used metal thermocouples), which offers four main advantages[22]:

- Seebeck coefficients are one or two orders of magnitude higher, which increases the induced thermoelectric voltage in response to a temperature difference.
- Semiconducting thermopile characteristics can be finely tuned by doping processes.
- Thermal capacity of sensors can be reduced efficiently.
- High detectivity in thermopiles using conventional IC processes (such as the CMOS process).

This enables the design of small thermopile sensors, and even thermopile arrays, to be commercially available at affordable prices. For example Heimann Sensor, who produces thermopile sensors with resolutions up to 80×64 by utilising a combination of CMOS processes and micromachining. Although these sensors do not reach the resolutions possible with bolometers, or the responsiveness of pyroelectrics, thermopile technology delivers simple and reliable sensors with an excellent cost to performance ratio[22].

2.3 Previous Research on Infrared Horizon Sensors

This section investigates some of the previous work done on the design and development of horizon sensors. This includes:

- Analysis of the infrared horizon, as well as the performance of infrared horizon sensors, based on flight data from 12 satellite missions.
- Two CubeSat infrared horizon sensors with flight heritage.

2.3.1 Infrared Horizon Sensor Modeling for Attitude Determination and Control[1]

In 1985 NASA compiled a survey on the work done by their Attitude Determination and Control Section on infrared horizon sensors by analysing and evaluating the performance of satellites from before 1970 to 1984. This included up to 12 different satellite missions attempting to determine the state-of-the-art in infrared horizon sensing. These missions investigated the effects of the infrared horizon over different climates and locales using various triggering methods, and their effects on the satellite's attitude control systems. This section will focus on a few applicable and important aspects of the survey, which is shown below.

Infrared Spectrum Passband: All infrared sensors on the surveyed satellites viewed the Earth's infrared spectrum in a passband centered around $15 \mu\text{m}$, mostly with a passband width of roughly $4 \mu\text{m}$. This portion of the spectrum is dominated by the CO_2 absorption band above the troposphere. This band is chosen because the intensity stays fairly constant regardless of climate or locale-specific conditions. This is illustrated in Figure 2.2 where the radiance intensity difference is plotted as a function of latitude (the latitude is differentiated as either Sahara or Antarctic), and the contour lines represent the temperature in Kelvin. The stability of the infrared horizon location directly correlates to the potential accuracy of an infrared horizon sensor, and this figure shows that infrared sensors susceptible to below $14 \mu\text{m}$ or above $16 \mu\text{m}$ could suffer from large signal variations between different climates.

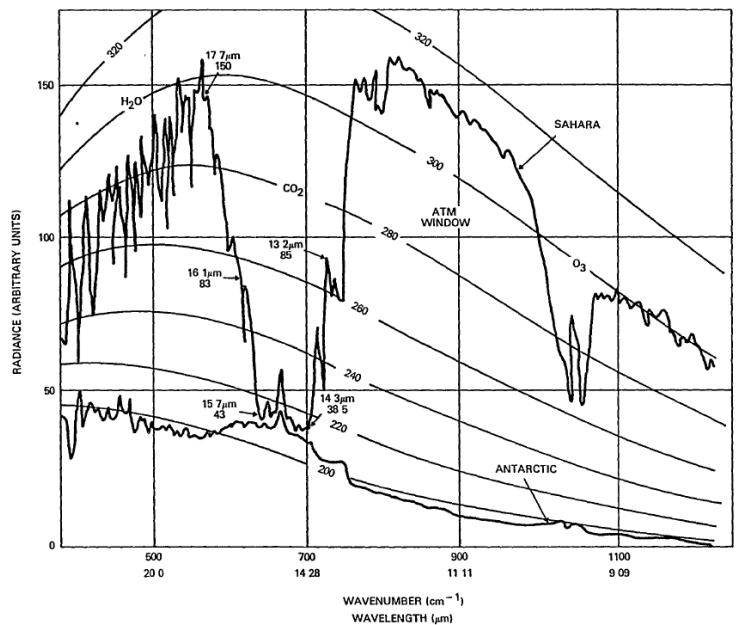


Figure 2.2: Earth's infrared spectrum for two extreme conditions, namely Sahara and the Antarctic[1]

Infrared Horizon Profile: The profile of the infrared horizon edge is an important factor in determining the horizon edge accurately. Therefore NASA launched a mission, codename Project Scanner, which took measurements of the infrared horizon profile with numerous single rocket flights in 1966. As climate conditions will influence the horizon, measurements were taken over different parts of the year over a variety of latitudes. The averaged results of this investigation are shown in Figure 2.3.

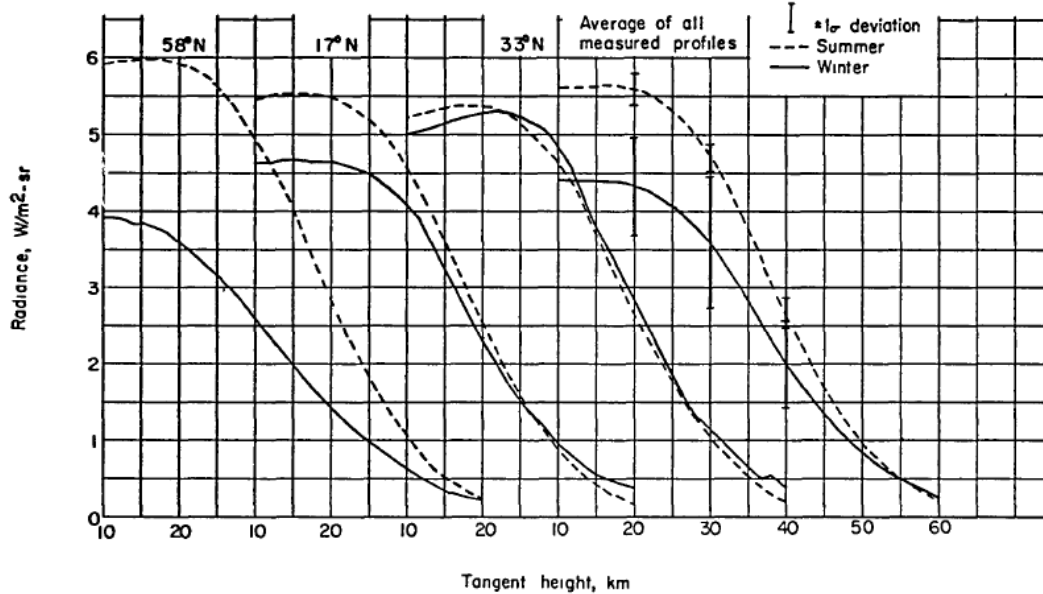


Figure 2.3: Example of seasonal variation of measured horizon radiance profiles for various latitudes[1]

It is clear that the horizon profile varies significantly due to the climate (or season), with the variation emphasized closer to the Arctic regions. This variation could result in large attitude estimation errors, as high as 0.25° . This correlates with data captured by a later satellite's horizon sensor (TOMS-EP, 1996), which recorded a variation in the horizon location of roughly 0.2° from a 500 km orbit[23]. This variation in horizon location lead NASA to start developing atmospheric radiance models to improve their sensors' accuracy.

Infrared Radiance Modeling: The satellites surveyed utilised different infrared radiance models which varied in complexity and purpose. Generally, these models performed well during summer and near the equator, but during winter and in the Arctic regions the day-to-day variation in radiation increased significantly. This is due to the stochastic nature of winter weather. Several models were created by various research centers, with the best performing model being the HRDB (Horizon Radiance Data Base) developed by Computer Sciences Corporation. The HRDB model returned the expected horizon profile relative to the latitude, spectral band, and month. This was fairly advanced for the time, as most other models had a model for each season (instead of each month), and utilised simplification techniques such as season mirroring. The Landsat-4 showed that using an adaptation of this model (which includes the Earth's oblateness), a $3\sigma = 0.1^\circ$ attitude estimation accuracy could be achieved during summer at an orbit height of 710 km , but during winter the accuracy lowered to roughly $3\sigma = 0.2^\circ$.

Triggering Methods: Triggering references the method the various sensors utilised to determine the horizon location from the horizon profile. The most successful triggering methods, according to the survey, included a normalized threshold trigger (e.g. Seasat-1) and derivative trigger (e.g. Landsat-4). The normalized threshold trigger estimated the horizon at a certain percentage of the mean Earth radiance value (e.g. 40%) on the horizon profile. This method is only justified if the horizon profile varies in amplitude and has no change in shape. This was not the case for Seasat-1, and it experienced errors attitude estimation up to 0.3° due to

its wide spectral passband. The derivative trigger method relied on a dual conical scanning system, where the horizon was estimated as the middle of the ascending and descending points (i.e. of the 2nd derivative of the horizon profile). This triggering method delivered accurate attitude knowledge (mentioned earlier) with resilience to the horizon profile shape.

2.3.2 Attitude Control on the Pico Satellite Solar Cell Testbed-2[2]

The Pico Satellite Solar Testbed-2 (PSSCT-2) was built by The Aerospace Corporation and launched from Atlantis on the final U.S. Space Shuttle mission in 2011. Included in this satellite's attitude control system was one of the first infrared horizon sensors flown on a CubeSat. This 25×25 cm sensor utilised nine COTS thermopile sensors developed by Melexis (MLX90615) which were mounted in different orientations, as shown in the schematic drawing in Figure 2.4a. This resulted in extensive Earth coverage, as seen in Figure 2.4b. By comparing the measured nadir Earth radiance with the Earth-plus-space radiance, the nadir angle could be estimated up to 0.5° along two axis (pitch and roll). It appears that no Earth radiance models were used, which also contributes to the low accuracy observed. Flight data of this sensor is not available at time of writing.

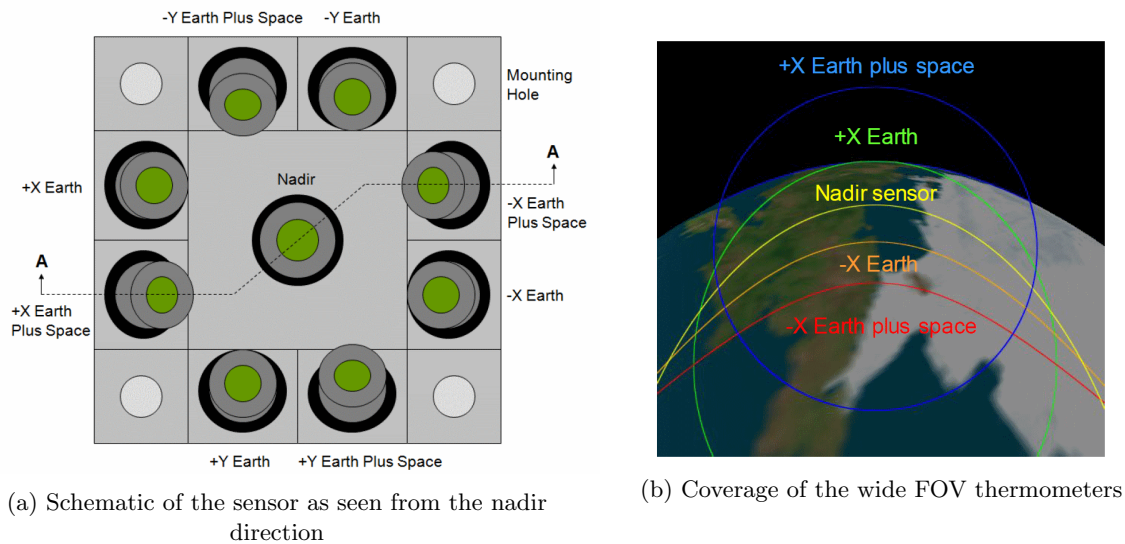


Figure 2.4: Visualisation of the infrared nadir sensor on the PSSCT-2[2]

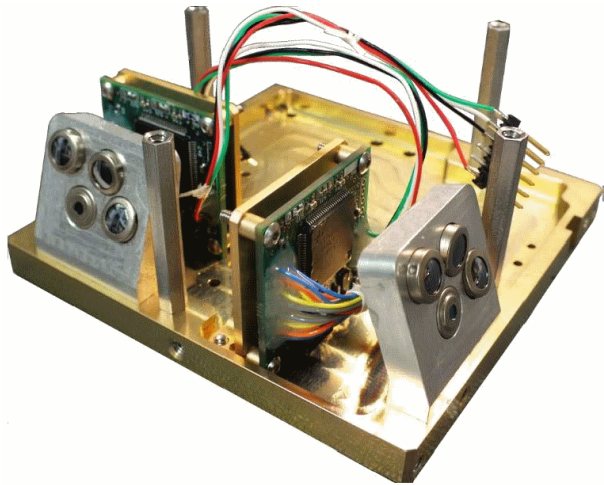
Initial testing showed that leaking occurred from some of the sensors under prolonged exposure in vacuum chambers. This caused inaccurate temperature readings due to reduced conductive and convective heat transfer within the sensor housing. Even though sensors were selected that did not appear to leak in ground tests, leaking still occurred in space. Therefore The Aerospace Corporation's sensors are now vented during development by drilling holes in the housing to force uniform behaviour under vacuum conditions.

This nadir sensor does not seem to be commercially available.

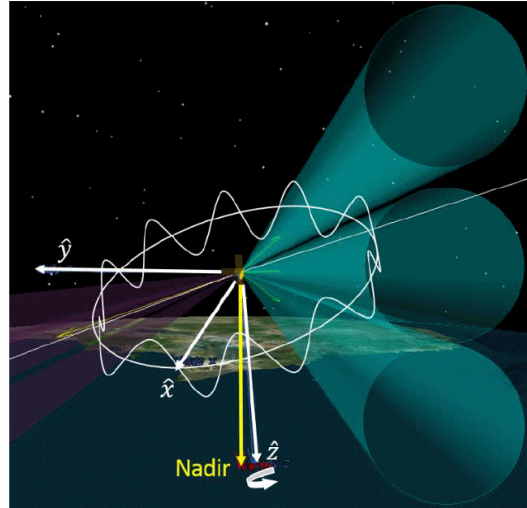
2.3.3 Attitude Determination using Infrared Earth Horizon Sensors[3, 4]

The Massachusetts Institute of Technology's (MIT) Department of Aeronautics and Astronautics (AeroAstro) developed an infrared horizon sensor called the MAI-SES which was implemented on the MicroMAS satellite

(launched in 2014). This sensor is available for purchase from Maryland Aerospace, and also referenced in Table 2.1. Each sensor utilises four COTS TPS334 thermopile detectors (developed by Excelitas). Three of the four thermopiles have a narrow FOV of 6° , with the final coarse sensor having a FOV of 60° . The three fine sensors view the Earth, Earth-plus-space, and space respectively to estimate an accurate nadir angle as shown in Figure 2.5b, with the coarse sensor solving the possible ambiguity. utilising two MAI-SES sensors, as shown in Figure 2.5a, the satellite's pitch and roll can be determined.



(a) Image of two mounted MAI-SES sensors



(b) Coverage of the three fine thermopile sensors represented by the blue circles

Figure 2.5: Visualisation of the MAI-SES infrared nadir sensor[3, 4]

The development of this sensor simulated accuracies as low as 0.18° when taking into account the Earth's oblateness. However, this prediction was based on the assumption that the infrared radiance observed by the sensor is uniform in its FOV, and appears to ignore atmospheric effects on the infrared horizon. Flight data of this sensor is not available at the time of writing.

2.4 Possible Improvements and Additions

According to Phenneger[1] the theoretical limit in accuracy of infrared horizon sensors is approximately $3\sigma = 0.1^\circ$. Current CubeSat horizon sensors are approaching this limit (albeit only in simulation), but future advancements can result in even higher accuracies. Current devices are also large and power hungry, mainly due to the use of several individual thermopile sensors, which offers significant room for improvement. Therefore, improvements in accuracy, size, and power usage can be achieved by:

- utilisation of improving thermopile technology to focus on miniaturization and low power devices, such as the using the small COTS single-package thermopile arrays created by Heimann Sensor[6].
- Developing novel software to ensure high accuracies with limited sensor technology.
- Accurate and efficient modelling of the varying horizon location due to the varying infrared radiance profile and Earth oblateness.

This study will not implement the modelling of the horizon, but instead focus on developing a low-power, low-cost sensor that accurately determines the current horizon's location.

3 Imaging Environment Investigation

This section will investigate what the infrared camera is expected to observe when mounted on a satellite in space. The size and shape of the horizon were determined, as well as possible noise expected on the system. Additionally, a horizon simulation program was designed in MATLAB to aid the development of horizon detection algorithms, and a ground testing environment was built to test said algorithms.

3.1 Observable Earth Shape

This section will investigate the expected shape of the horizon viewed from space to efficiently develop and evaluate horizon detecting software. It is assumed that the camera is pointing towards the horizon, with an elevation offset ϕ , as shown in Figure 3.1.

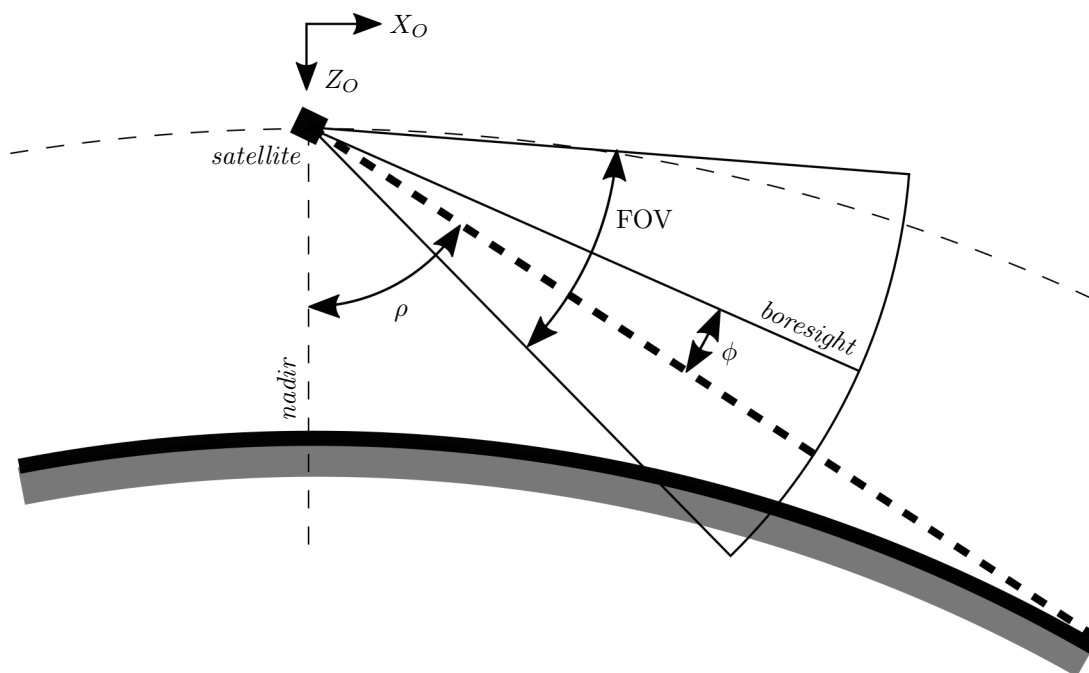


Figure 3.1: Camera mounting and pointing direction

If approximating the Earth as a perfect sphere the Earth always creates a segment of a perfect circle on the camera's image plane. This circle is called the Earth disc[24] and is described in Figure 3.2. Therefore, the Earth disc's radius is only dependant on the satellite altitude H . Approximating the Earth as a sphere is sufficient, as the difference between an oblate Earth disc and circular Earth disc is negligible (see Section 3.2).

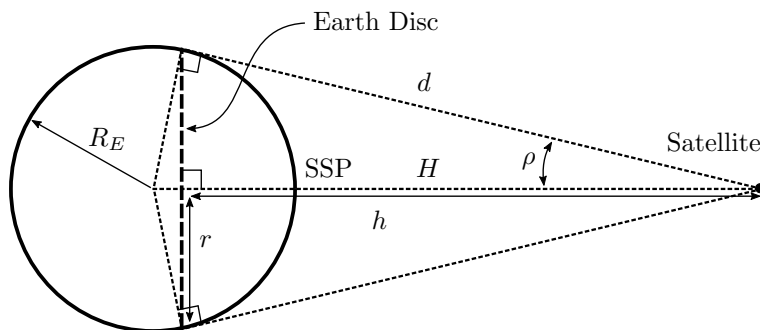


Figure 3.2: Earth disc definition

It was assumed that the satellite is at an altitude (H) of 500 km, which is a typical altitude for CubeSats. This results in the Earth disc radius (r) being calculated by Eq. 3.2.

$$\rho = \arcsin\left(\frac{R_E}{H + R_E}\right) \quad (3.1)$$

$$r = R_E \cos \rho \quad (3.2)$$

With ρ calculated as 68.02° the Earth disc radius r is equal to 2387.4 km. The size of this Earth disc on the camera's image plane is dependant on the camera's focal length (f) of 5.8 mm [6]. Using a typical pinhole approximation of the infrared camera lens the size of the Earth disc on the image plane r_p is equal to:

$$r_p = f \tan \rho = 0.0144 \text{ m} \quad (3.3)$$

The pixel pitch of the camera is 220 μm which means the Earth disc radius is 65.31 pixels. This is larger than the 32×31 pixels sized imager, and therefore the full Earth disc will not fit in a single image. However, the location of the Earth disc on the image plane will vary with different camera attitudes. To calculate this location it was assumed that with an elevation of zero the camera boresight is pointed at the horizon. Therefore, with an elevation angle ϕ a small segment of the disc will vertically translate a distance of α pixels, as described by Eq. 3.4.

$$\alpha = f \tan \phi \quad (3.4)$$

Assuming an elevation angle ϕ and zero rotation angle, the Earth disc's center is described using Eq. 3.5 and 3.6. These equations assume that the coordinate system origin is in the center of the image plane.

$$x_0 = 0 \quad (3.5)$$

$$y_0 = -(\alpha + r_p) \quad (3.6)$$

A rotation θ is then applied to the camera which rotates the Earth disc around the center of the image plane (the origin). Due to the fact that the Earth disc is approximated by a circle only the circle center has to be rotated. This will change Eq. 3.5 and 3.6 to:

$$x_0 = (\alpha + r_p) \sin \theta \quad (3.7)$$

$$y_0 = -(\alpha + r_p)\cos\theta \quad (3.8)$$

Therefore, the Earth disc's center will be at the point (x_0, y_0) on the image plane with a radius of r_p pixels. An example case is shown in Figure 3.3 where an elevation of 20° and rotation of 30° is applied to the camera and the horizon shifted appropriately.

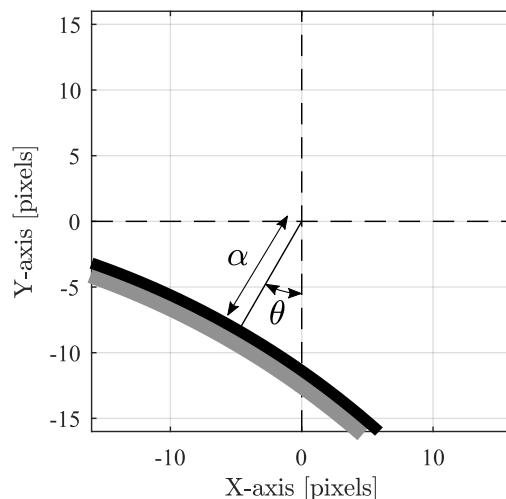


Figure 3.3: Example of expected horizon with a camera elevation of 20° and rotation of 30°

3.2 Effect of Earth Oblateness and Infrared Horizon

As discussed in Section 2.3.1 by Phenneger[1] the accuracy of a horizon sensor is severely influenced by the Earth oblateness and, more importantly, the infrared horizon profile. These effects change the measured location of the horizon relative to the satellite's body axis, which results in an erroneous attitude estimation.

Effect of Earth Oblateness: The simpler of these two effects is the Earth's oblateness. Most Earth models, which is also used in this project, assume the Earth is a perfect sphere for simplicity, but in reality, the Earth is oblate and therefore ellipsoid shaped. The spherical assumption is sufficient in most use cases, but it is essential to investigate the possible errors such an assumption might induce.

A common model for the Earth's ellipsoid shape is the World Geodetic System established in 1984, or simply called the WGS84[25]. According to the WGS84, the Earth's flattening factor is approximately $1/f = 298.3$, which can be used to calculate the difference between the Earth radius at the equator and the poles as $\Delta R_E = 21.39 \text{ km}$. Using this difference, an approximation of the maximum attitude error (ϵ) is calculated with Eq. 3.9 and using Figure 3.4.

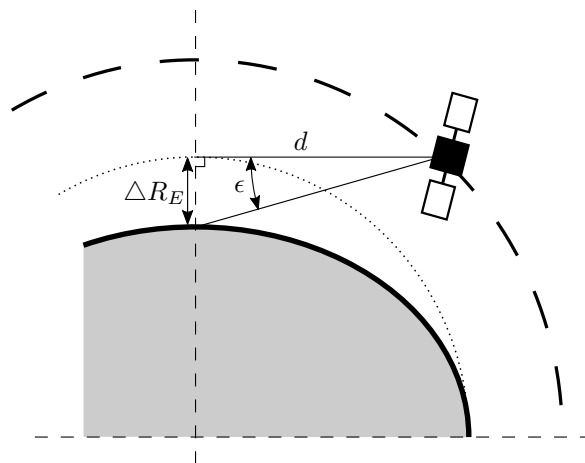


Figure 3.4: Illustration of the ellipsoid Earth (solid line) relative to the spherical Earth (dotted line) shown with an approximation of a horizon sensor's maximum measurement error (ϵ)

$$\epsilon = \tan^{-1} \frac{\Delta R_E}{d} \quad (3.9)$$

This shows a maximum error of roughly $\epsilon = 0.48^\circ$ from an altitude of 500 km . This severely limits the potential accuracy of a horizon sensor, and therefore needs to be accounted for in a final implementation.

Effect of Infrared Horizon: As discussed in Section 2.3.1 the ideal spectral window for infrared horizon sensors are around $15 \mu\text{m}$. The infrared camera used for this project's spectral response is between $8 \mu\text{m}$ and $11.5 \mu\text{m}$. However, it is assumed that a future infrared horizon sensor implementation will ensure that the infrared camera has a narrow spectral passband at $15 \mu\text{m}$. This is a sufficient assumption as a thermopile's spectral response is fairly independent of wavelength and mostly influenced by the transmittance characteristics of the window and lens, therefore making it simple to manufacture[26].

The correct choice of spectral pass band does completely eliminate fluctuations in the horizon location, but only minimizes it. Phenneger[1] measured horizon location variations of up to 10 km in this pass band, which corresponds to errors up to 0.22° from a 500 km altitude, if adapting Eq. 3.9. This can be accounted for pre-flight with readily available software that models atmospheric propagation of electromagnetic radiation, and then determining the sensor's measurement errors for different conditions. The most accurate atmospheric radiance model is called MODTRAN (Moderate Resolution Atmospheric Transmission) which is a FORTRAN-based simulation developed by Spectral Sciences, Inc. and the U.S. Air Force Research Laboratory[27]. The Satellite Development Kit (STK) also includes a simplified MODTRAN-4 based atmospheric model[28].

For future reference Figure 3.5 shows the expected horizon profile (see Figure 2.3) as seen by the supplied infrared camera[6], assuming it has the ideal $15 \mu\text{m}$ spectral response. It shows the horizon profile in voltage, additional to temperature, as the profile shape will change due to the 4th-order nature of Eq. 5.1.

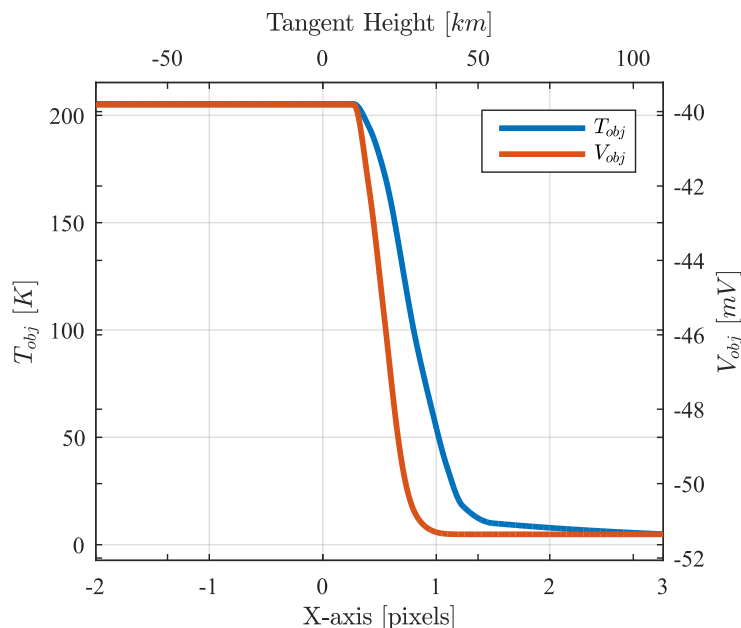


Figure 3.5: Expected view of the mean horizon profile of the summer in 1966 from a low-resolution camera

Influence on Project: The effects of the Earth oblateness and infrared horizon will not be taken into account during this project as the project focusses only on the design of the sensor. However, it will be taken into account during development for simpler easier development.

3.3 Signal to Noise Ratio

The signal to noise ratio (SNR) of a system measures the ratio of signal power to noise power. For this project the signal is defined as the difference in pixel intensity (measured in Volts) between the *warm* Earth pixels and the *cold* outer space pixels. The noise is defined as the standard deviation of the measured voltage. This is depicted in Figure 3.6, and described by Eq. 3.10 in *dB*.

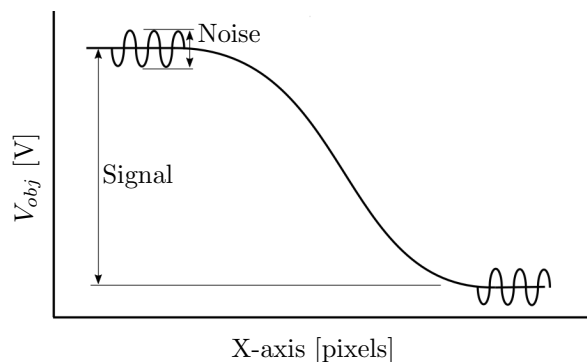


Figure 3.6: Definition of signal to noise ratio

$$\text{SNR} = \frac{V_{\text{signal}}^2}{\sigma^2} \quad (3.10)$$

The theoretical SNR is higher than the measured SNR of the final system, due to non-ideal components and approximated formulas. This section calculates the theoretical and measured SNR values and investigates how this can be improved. The SNR also changes due to a varying environment, and therefore only the worst case is investigated. The internal camera amplification also influences the SNR, and thus the SNR is calculated for both amplification settings (gain equal to 1 and 3).

3.3.1 Worst Case Theoretical SNR

The worst case theoretical SNR is calculated by investigating the radiance received by the infrared camera (Section 3.2), the change in pixel voltage due to this received radiance (Section 5.1), as well as the expected pixel noise according to the camera's datasheet [6]. External noise sources such as non-ideal power supplies, quantization errors and electromagnetic interference will not be investigated only the performance of the camera is quantified.

Signal Power: The signal power is calculated by first determining the change in temperature between the *warm* Earth pixels and the *cold* outer space pixels, on which Eq. 5.1 is then applied. Section 3.2 shows that the lowest Earth temperature can be roughly 200 K, and according to Phenneger[1] the Earth approximates a blackbody. Also, according to experiments run by a NASA satellite[29] the cosmic background spectrum of outer space is that of a nearly perfect blackbody with a temperature of roughly 2.7 K. For these tests, the camera housing temperature is assumed to be 25°C.

This results with the *warm* pixels having a minimum temperature of 200 K, and the *cold* pixels 2.7 K. The emissivity of both the Earth and outer space resembles that of a blackbody and is therefore assumed to be unity. Using Eq. 5.1 the change in pixel voltage is then calculated with the results shown in Table 3.1.

Noise Power: The infrared camera datasheet[6] specifies the inherent system noise by quantifying the noise present before the internal amplification ($30 \text{ nV}/\sqrt{\text{Hz}}$), as well as the noise created by the amplification ($50 \text{ nV}/\sqrt{\text{Hz}}$). Therefore, it is assumed that the pre-amplification noise will be increased by the amplification stage (gain = 1 or 3), which means the resultant total noise is either $30 + 50 = 80 \text{ nV}/\sqrt{\text{Hz}}$ or $3 \times 30 + 50 = 140 \text{ nV}/\sqrt{\text{Hz}}$. The voltage level of this noise is then calculated by using the system bandwidth. As described in Section 4.3.2 the system bandwidth is limited by an external low pass filter (LPF) with a cut-off frequency of 7.2 kHz. This LPF is assumed to have an ideal frequency response and unity gain. This results in the system noise with both amplification levels being 6.79 μV and 11.88 μV respectively.

Signal To Noise Ratio: The SNR is calculated by dividing the calculated power ($P = V^2/R$, assuming $R = 1 \Omega$) in the signal (P_{signal}) by the noise power (P_{noise}). The values used to calculate the SNR, as well as the final result, is shown in Table 3.1.

Table 3.1: Worst case theoretical SNR calculation values

	Cold Pixels	Warm Pixels
Temperature	2.7 K	200 K
Amplification = 1		
V_{obj}	-45.8 mV	-36.6 mV
V_{signal}	9.2 mV	
P_{signal}	-40.6 dBW	
V_{noise}	$6.79\text{ }\mu\text{V}$	
P_{noise}	-103.4 dBW	
SNR_{dB}	62.8 dB	
Amplification = 3		
V_{obj}	-116.9 mV	-93.3 mV
V_{signal}	23.68 mV	
P_{signal}	-32.51 dBW	
V_{noise}	$11.88\text{ }\mu\text{V}$	
P_{noise}	-98.5 dBW	
SNR_{dB}	66 dB	

These calculated SNR values are sufficiently high, as Section 6.6.2 shows that a SNR of at least 40 dB is required for sufficiently accurate horizon location estimation. However, the low P_{signal} values are caused by the low temperatures measured (2.7 K and 200 K), which decreases the power of the signal exponentially. This is observed in Eq. 5.1 where the voltage is proportional to the 4th power of the measured temperature. This is also visible in Planck's Law (Eq. 3.11) where the radiance power B_λ increases exponentially with the temperature (T).

$$B_\lambda(\lambda, T) = \frac{2hc^2}{\lambda^5} \frac{1}{e^{\frac{hc}{\lambda k_B T}} - 1} \quad (3.11)$$

However, higher *warm* pixel temperatures, such as 250 K in the Arctic winter, will increase the theoretical SNR values by roughly 10 dB for both amplification settings. This is approximately a 1000% increase in SNR with only a 25% increase in signal temperature difference. This is also shown by the measured results in Section 3.3.2 where a small signal temperature difference of roughly 75°C has an high P_{signal} of up to -14 dB . This means that measurements taken over cold environments will severely influence the SNR of the system. However, the SNR can still be increased by averaging the input images, which will reduce the image noise. This is explained in more detail in Section 3.3.3; and in Section 3.4 it is shown that no significant image distortion will occur while taking multiple images over a short time period.

3.3.2 Measured SNR

The actual SNR of the camera's test setup (described in Section 3.5) is calculated to quantify results relative to the systems SNR. This is done using actual measurements, and not the expected theoretical values, although it is compared to the expected values. Tests for the different amplification settings was run at different times. Therefore the camera temperature T_S and *cold* pixel values might vary. However, this does not influence results, as only a change in object temperature is required which essentially removes T_S from Eq. 5.1.

As discussed in Section 3.5.4 the used test setup do not create ideal conditions as the steel plate is not heated uniformly (Figure 3.10). Therefore, the values used for the SNR's signal calculations is only calculated over a small window covering the *warm* pixels, i.e. only the best case SNR. However, the drop-off in SNR as the

steel plate's temperature decays is shown and discussed in Figure 3.10.

Signal Power: The signal power is calculated by emulating the Earth horizon using a heated steel plate (described in Section 3.5), and quantifying the difference in measured object voltage V_{obj} . An example of the measured edge between the *warm* and *cold* pixels is shown in Figure 3.7 for both amplification settings. From this figure it is clear that the measured V_{signal} is 71.31 mV and 193 mV for both amplification settings respectively. This figure also shows that a typical edge between *warm* and *cold* pixels approximates a sigmoid function.

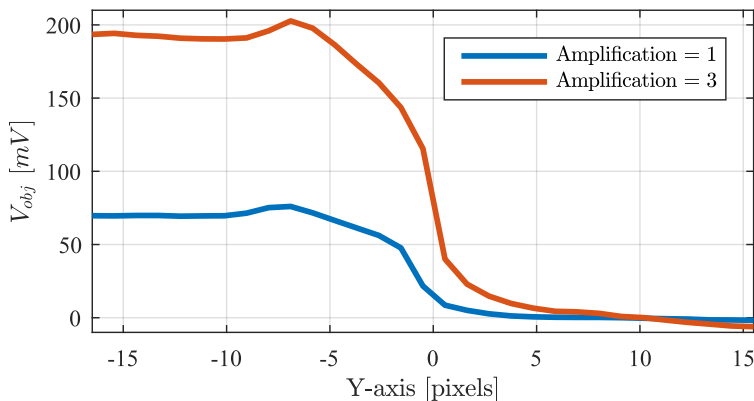


Figure 3.7: Example of measured edge between *warm* and *cold* pixels

This correlates roughly with the expected signal voltages of 67 mV and 170 mV calculated using Eq. 5.1. This verifies that the SNR values determined in Section 3.3.1 are accurate.

However, it should be noted that signal strength decreases to the sides of the steel plate with the drop in temperature. This is because the plate's heating elements were non-ideal which resulted in the temperatures being non-uniform. This decreases the performance of the camera, but only during ground testing. A depiction of this is shown in Figure 3.14, and an improved test setup is proposed in Section 10.2.

Noise Power: The noise power was calculated by taking large amounts of images in the expected conditions. The tests were run in non-ideal conditions where the plate and background temperature varied during testing. This was due to the change in room temperature and sunlight resulting in the signal's mean value changing over time. This change in mean value was removed by fitting a polynomial through the signal over time for each pixel, and subtracting it from the original signal. It was found that a polynomial of the 5th order follows the mean value sufficiently without following the noise. The noise was also expected to increase as the imaged object's temperature increased, and therefore the *warm* and *cold* pixels were calculated separately. An example of such approximations is shown in Figure 3.8 where the change in signal mean value is clearly shown, additional to the noise and the fitted polynomial.

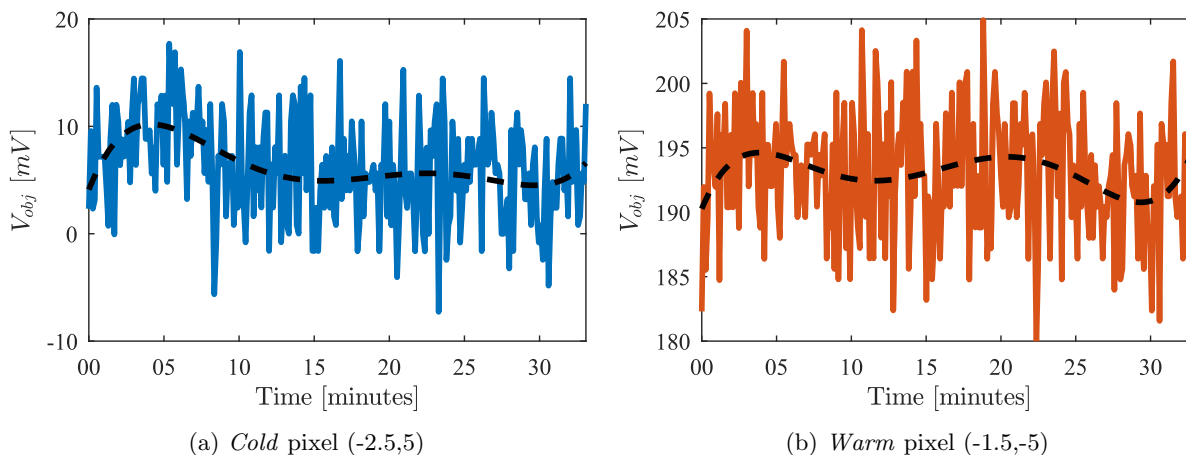


Figure 3.8: Example of a fitted polynomial to *cold* and *warm* pixel's signal over time (amplification = 3)

Once all the pixel signals' mean values were removed, all the *warm* pixel's datasets are concatenated to create one large dataset showing the noise distribution. This was also done for the *cold* pixels and for both amplification settings. The noise power P_{noise} is equal to the noise variance (σ^2). A histogram of this concatenated data is shown in Figure 3.9, with the results shown in Table 3.2.

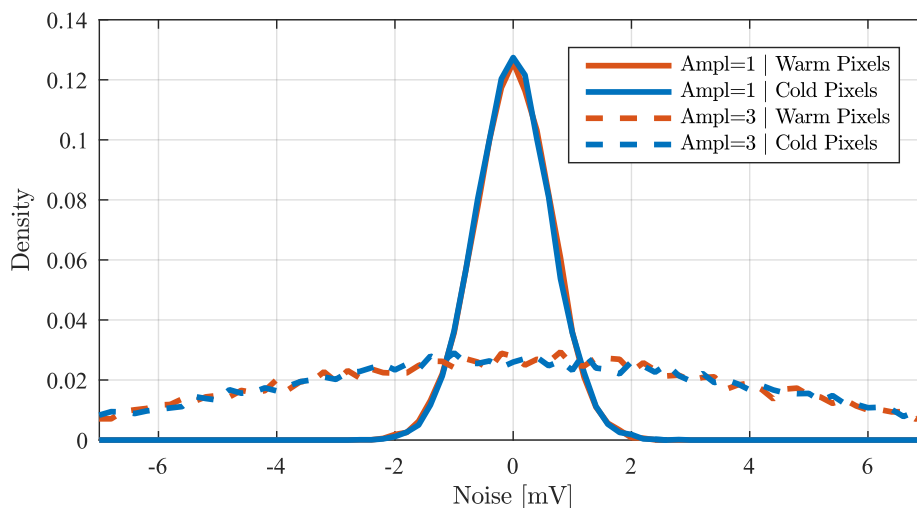


Figure 3.9: Histogram of noise present for *warm* and *cold* pixels for both amplification settings

This noise is higher than the theoretical noise levels described in Section 3.3.1 due to: but not limited to, non-ideal power supplies, quantisation errors and electromagnetic interference on a non-ideal PCB layout. It is evident that there is no significant difference in noise levels between measured *warm* and *cold* pixels.

Signal To Noise Ratio: In Table 3.2 the values required to calculate the measured system's SNR is shown. The measured SNR is lower than the theoretical values due to increased noise levels in a real implementation. However, the higher temperatures measured does result in a higher signal power.

Table 3.2: Best Case Measured SNR calculation values

	Cold Pixels	Warm Pixels
Temperature	298.15 K	373.15 K
Amplification = 1		
V_{obj}	80 μV	71.4 mV
V_{signal}	71.3 mV	
P_{signal}	-22.9 dBW	
V_{noise}	0.63 mV	0.64 mV
P_{noise}	-64 dBW	
SNR_{dB}	41.1 dB	
Amplification = 3		
V_{obj}	1.25 mV	193 mV
V_{signal}	191.8 mV	
P_{signal}	-14.3 dBW	
V_{noise}	4.49 mV	4.51 mV
P_{noise}	-46.9 dBW	
SNR_{dB}	32.6 dB	

As discussed previously, these results were calculated using only pixels pointed directly at the hot plate. Figure 3.10 shows the drop in SNR as the steel plate's temperature dissipates for both amplification settings.

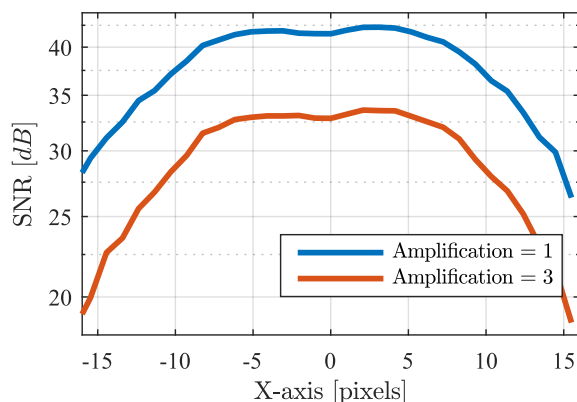


Figure 3.10: Drop-off in SNR due to non-uniform heating of steel plate

It is clear that the ground test setup does not accurately emulate the theoretical conditions relative to the expected SNR, although the signal power is significantly higher. This is due to the large noise power (P_{noise}) present on the system, which was mainly caused by the non-ideal power supply and electronic components. The SNR of the system was further increased by averaging input images, as explained in Section 3.3.3, but still did not accurately emulate the expected conditions. However, this SNR was high enough for sufficient testing.

3.3.3 Effect of Averaging on SNR

Averaging the measured signal increases the SNR of the signal because the noise is uncorrelated and the signal will stay constant during averaging. This increase in SNR will be quantified in this section. Averaging

a constant signal sampled n times results in the same value as $\frac{\sum^n c}{n} = c$. However, the noise does not stay constant but decrease as it is uncorrelated. This decrease is calculated using the Bienaymé formula[30], which is described in Eq. 3.12.

$$\text{Var} \left(\sum_{i=1}^n X_i \right) = \frac{1}{n} \sum_{i=1}^n \text{Var}(X_i) \quad (3.12)$$

It is also assumed that all the samples have equal variances σ^2 , which means that a division by n is a linear transformation. This implies that the variance of the averaged sample is:

$$\text{Var}(\bar{X}) = \text{Var} \left(\frac{1}{n} \sum_{i=1}^n X_i \right) = \frac{1}{n^2} \sum_{i=1}^n \text{Var}(X_i) = \frac{\sigma^2}{n} \quad (3.13)$$

Therefore Eq. 3.10 can be rewritten as:

$$\text{SNR} = \frac{nV_{\text{signal}}^2}{\sigma^2} \quad (3.14)$$

This shows that the SNR increase due to averaging α_{avg} , in dB , is described as:

$$\alpha_{\text{avg}} = 10 \log(n) \quad (3.15)$$

This equation shows that the effective SNR can be significantly increased using image averaging. The absolute maximum number of images that can be taken within 1 second is nine images (see Section 8.3), which corresponds to a $\alpha_{\text{avg}} = 9.5 \text{ dB}$ increase in SNR. This proves that the low theoretical SNR values (Section 3.3.1) can be sufficiently increased using averaging.

3.4 Motion Blur Quantification

If the horizon's position relative to the camera boresight does not stay constant during an image acquisition, it can result in a blurred image. This is a similar effect as to when a moving object is imaged by a camera with a slow shutter speed. This will happen if the camera has a non-zero angular rate around the Y_O -axis during the 108 ms it takes to capture a single image. Additionally, blurring can also be induced while capturing and averaging multiple images over a 1 second period maximum, as discussed in Section 3.3.3. Therefore, an investigation was done on the magnitude of the horizon shift during different angular rates.

In a circular orbit, the horizon shift is directly proportional to the angular rate around the Y_O -axis. A simulation was run to determine the horizon's angular shift over a single image capture with a duration of 108 ms . The results are shown in Figure 3.11. Typical control stability rates for 3-axis stable CubeSats are approximately $\omega_{oy} = 0.1^\circ/s$, and therefore the expected horizon shift was investigated for rotation rates up to $\omega_{oy} = 0.2^\circ/s$ (to include possible worse conditions).

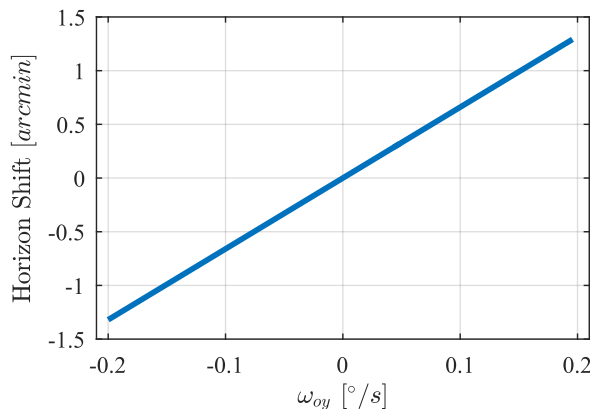


Figure 3.11: Horizon shift vs. satellite angular rate over a 108 *ms* time period

From this figure, it is clear that during a single image acquisition while rotating at a typical $\omega_{oy} = 0.1^{\circ}/s$, the horizon will shift 0.6 *arcmin* or 0.01° which is within the accuracy margins. If multiple images are taken over 1 second (maximum acquisition time) and averaged the horizon shift increases to 0.1° which will slightly influence the results. However, this is still an extreme case as images will only be taken for approximately 500 *ms*, as described in Section 8.3. Therefore, the motion blur was not investigated further.

3.5 Ground Test Environment

This section will describe the ground testing environment created to emulate the expected Earth disc which is used to test the infrared camera as a horizon sensor.

3.5.1 Required Testing Environment

The ground test environment was designed to only roughly emulate the visible Earth disc and satellite attitude, as a perfect emulation is not easily realised. Therefore, other expected factors such as vacuum, solar radiation and weightlessness were ignored as it is not expected to interfere with the results significantly. The ground test environment consists of three main parts, namely:

- Stationary heated object to emulate warm Earth disc.
- Stationary cold background to emulate cold space.
- Camera mounted on 2-axis high precision rotation stage to emulate the camera's pointing elevation and rotation.

The heated object was realized by heating a curved black anodized steel plate which ensured maximum emissivity. This plate was heated to a maximum of approximately $100^{\circ}C$ using a conventional dual kitchen hot plate fastened to the steel plate. The cold background was a glass window behind the steel plate. Initially a second Aluminium plate was used as a cold background, but it was found that this plate heated up significantly (roughly $30^{\circ}C$) due to the heated steel plate's thermal radiation. Alternatively, the glass window was not severely affected by the thermal radiation and dropped to as low as $20^{\circ}C$. This resulted in the time of day and outside temperature having a varying effect on camera readings. Therefore, it was attempted to collect different datasets during similar conditions, such as at night or a certain time of day.

The rotation stage was created by combining two separate rotation stages, both provided by Stellenbosch

University. Using these devices the elevation angle could be measured to 10^{-5} degrees resolution, and the rotation stadium up to approximately 0.017° , which is significantly better than the required accuracy margins. Unfortunately, these stages were not able to actuate autonomously due to defective actuators or lack thereof. This meant that it had to be rotated by hand and in turn severely increased testing durations.

The assembly of this setup was done with the following criteria:

- The infrared camera used is designed to focus at infinity, and therefore the steel plate should be far away from the camera. However, the steel plate must still be close enough to fill the entire FOV of the camera.
- The camera must be mounted in the center of the entire rotation stage. This is to simplify testing algorithms, as a rotation in each rotation stage will result in a pure elevation or rotation only.
- The steel plate must have a circularly curved edge to emulate the Earth disc, as well as a straight edge to aid the lens calibration.

An overview of the final ground test setup is shown in Figure 3.12.

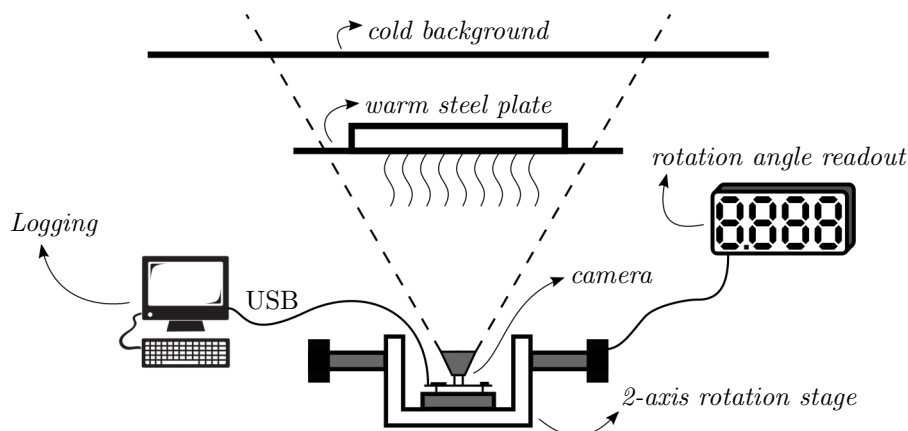
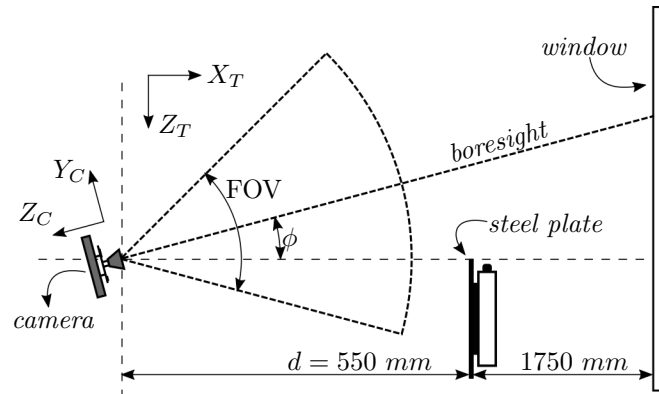


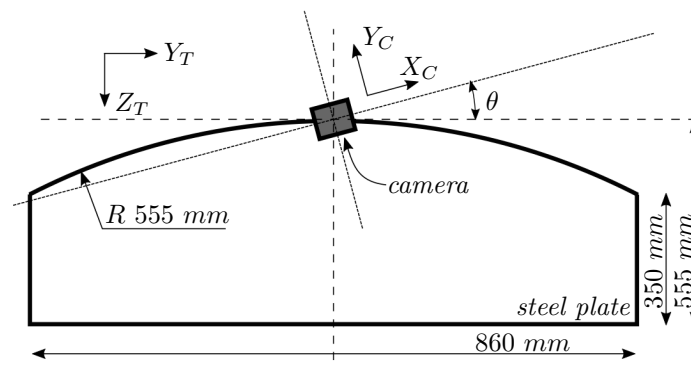
Figure 3.12: Overview of ground test setup

3.5.2 Detailed Camera and Earth Disc Setup

In Figure 3.13 the detailed ground testing setup is shown. Although the steel plate should ideally be placed more than 6 m away from the camera, the steel plate was placed only 550 mm away from the camera. This was done to meet the requirement of the steel plate filling the FOV. As will be discussed in Section 7.3.4, the effective FOV of the camera is roughly 70° . This means that the steel plate should be placed a maximum of 600 mm from the camera, therefore 550 mm was sufficient.



(a) View from side showing elevation stage



(b) View from back showing rotation stage

Figure 3.13: Detailed drawings showing ground testing setup

This steel plate did not create a perfect circle on the image plane, even though the plate itself contained a segment of a perfect circle. This was because the camera always images the flat disc from a slight angle, and therefore slightly warped the circle to an elliptical shape. However, this effect is negligible as the angles and distances used are small enough. This steel plate also created a smaller visible disc than would be expected from a 500 km orbit (described in Section 3.1). The plate's radius of 555 mm converts to 26.6 pixels on the image plane, instead of 65.3 pixels. This relates more to a satellite in a higher orbit, or a camera with a larger FOV.

It should be noted that the infrared camera was not in focus at distance d from the steel plate, which caused the visible edge to be blurred. This effect was also amplified by heated air visible by the infrared camera. To minimize this effect a desk fan was used to circulate the air, but the effect was not eliminated entirely. An example of the visible edge profile is shown in Figure 3.7, and a better solution will be discussed in Section 10.2.

3.5.3 Impact of Misalignments

Care was given to ensure that all the ground testing components were perfectly aligned, but due to the nature of the setup slight misalignments were inevitable. Therefore, the possible effects of these misalignments were investigated.

Steel Plate Misalignments: If the steel plate is only misaligned in the Z_T -axis, it will result in a elevation measurement error where a 10 mm offset will bias the measured elevation by 1.04° . Such a misalignment can be expected in space conditions and needs to be corrected during runtime by adding an offset to the measured elevation angle. A misalignment in the Y_T -axis will induce an error in the elevation and rotation measurements. Such a disc shift is not expected (or possible) in space conditions. For example, when measuring a zero elevation and rotation disc with a 10 mm offset in the Y_T -axis, it will result in an elevation error of 0.01° and a rotation error of 1.03° . During ground testing, this was corrected by translating the approximated disc center appropriately (see Section 9.2). A rotational misalignment in the X_T -axis will directly translate to a rotation error. Any other misalignments of the steel plate is negligible as it will not significantly influence the measured ϕ or θ angles. In Table 3.3 the mounting tolerances for the steel plate is shown.

Table 3.3: Mounting tolerances of steel plate around the different axes

	X_T	Y_T	Z_T
Translational	10 mm	5 mm	5 mm
Rotational	0.5°	3°	1°

Camera misalignments: The camera might have slight rotational misalignments due to the mounting method on the PCB, as well as the mounting on the rotation stages. However, these rotational misalignments correspond to possible misalignments in space applications, and can easily be corrected in runtime by using a constant offset. Any translational offset on the camera, i.e. distance from the rotation stage's center, can significantly influence results. For example, if the camera is misaligned in the X_c direction it will behave similar to a steel plate misalignment in the Y_T direction.

Table 3.4: Mounting tolerances of camera on the rotation stage around the different axes

	X_C	Y_C	Z_C
Translational	1 mm	1 mm	1 mm
Rotational	3°	3°	3°

3.5.4 Non-Uniform Steel Plate Heating

Ideally, the steel plate should be heated uniformly to emulate the Earth disc and fill most of the camera's FOV. However, due to the localized heat source (hot plates) and a combination of the steel's low thermal conductivity and high emissivity, the steel plate was not uniformly heated. The measured temperatures of the heated steel plate is shown in Figure 3.14 (assuming both stoves were turned to 80% power).

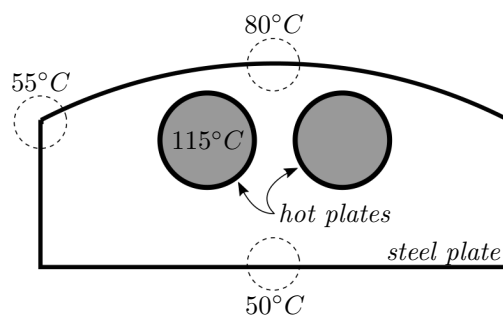


Figure 3.14: Temperature distribution on heated steel plate

Additionally, in some elevation/rotation configurations, the steel plate did not fill the entirety of the camera's FOV, as seen in Figure 3.15a. This, in combination with the non-uniform heat distribution, caused detection of false edges, additional to interfering with the Rotation Pre-Estimation (Section 6.3.1). Therefore, during the algorithm development and testing on the PC, the images used were first processed to emulate the expected Earth disc more accurately, as shown in Figure 3.15b.

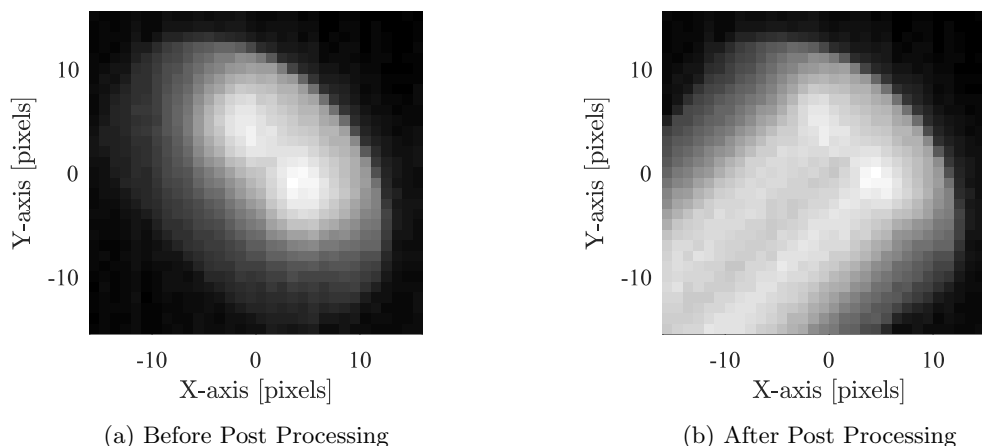


Figure 3.15: Example of ground test image before, and after, post processing. ($\phi = 45^\circ, \theta = 15^\circ$)

This processing was specifically designed not to alter the edge of the disc, but only to ensure the disc fills the FOV as uniformly as possible. The pixel intensity with which the missing disc is filled, was chosen dynamically, as the pixel intensity varies over the disc's edge (see Figure 3.10). The filling value was chosen as 90% of the maximum pixel value in the scanning direction. The scanning direction was limited to horizontal, vertical, or diagonal ($\pm 45^\circ$), as it is enough to sufficiently fill the FOV. To ensure the edge stayed unaltered, the filling was only enabled three pixels distance from the pixel with the maximum intensity, which is visible in Figure 3.15b. Three pixels distance was chosen as the limit because the combination of Sobel Filter (Appendix A) and the Local Extrema Method (Section 6.1.5) uses 2 pixels around the edge's middle. Therefore, a 3 pixel gap from the edge's maximum ensured that the Post Processing did not interfere with any feature extraction. This was sufficient as the filling was only required to aid the Rotation Pre-Estimation, which calculates an approximate rotation angle, and therefore a precise disc was not required. The orientation of the scanning lines was chosen to match the logged rotation of the image.

3.5.5 Computer Interface and Logging Software

All images taken by the camera is transmitted to the logging computer through a USB connection, which is also used to control the camera. This software has complete control of the device and has the ability to save information during onboard attitude estimation or image capture. However, during development of the device only the raw pixel values ($V_{AOTu} - V_{ref}$) are transmitted and stored to be thermally calibrated (Section 5) on the computer. This is done for ease of development and testing of calibration algorithms. A screenshot of this software (written in Python) is shown in Figure 3.16.

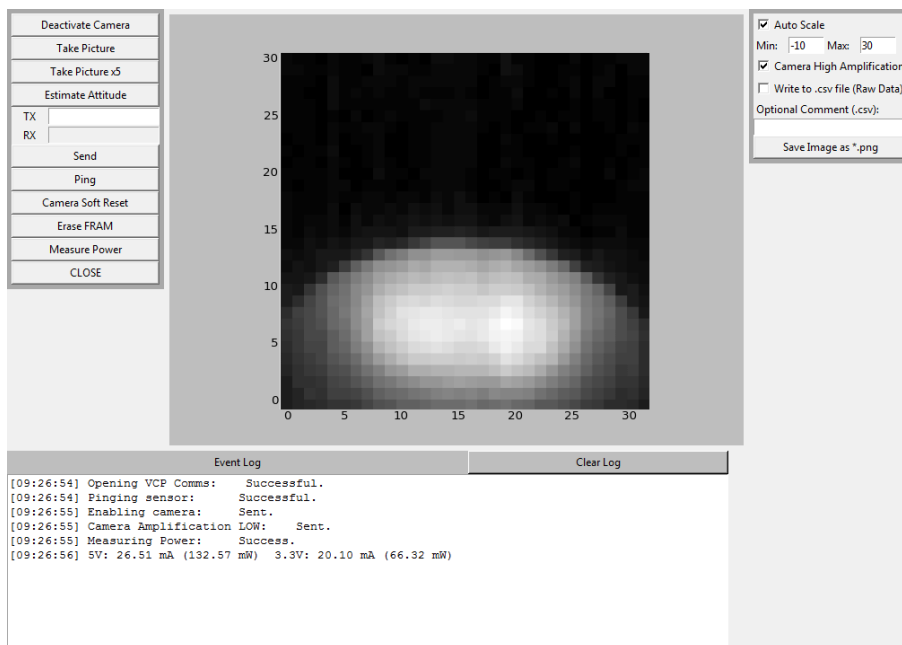


Figure 3.16: Screenshot of Computer Interface

3.6 Horizon Simulation Software

To thoroughly investigate image processing algorithms, such as in Section 6, a simulation program was written in MATLAB to create low-resolution images of the horizon. These images accurately represent the location and shape of the horizon while maintaining the ability to simulate different noise levels. The profile of the simulated horizon has two settings in order to simulate the expected high contrast horizon edge in space (see Figure 3.5), or the low contrast edge during ground testing setup (see Figure 3.7). This simulation does not take lens distortion into account as it is only intended to evaluate feature extraction techniques. The steps taken to create the simulated horizon are shown in Figure 3.17.

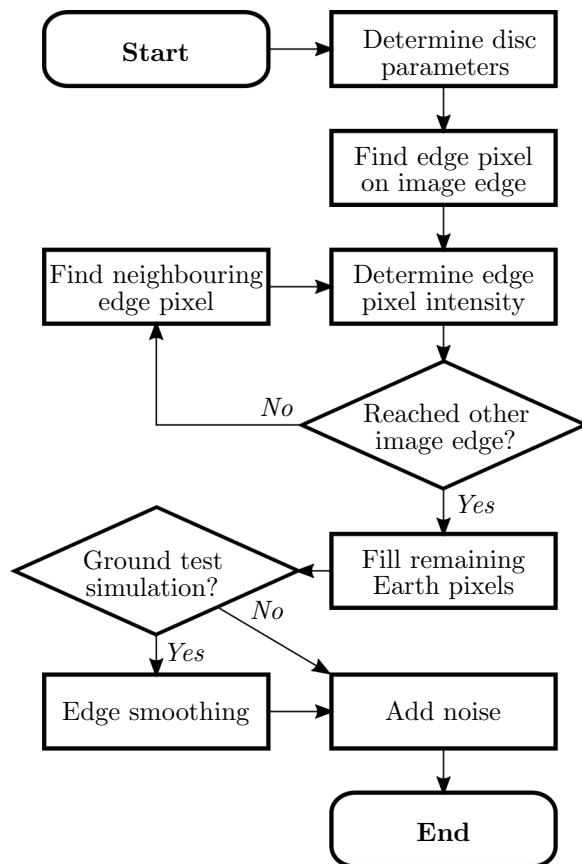


Figure 3.17: Flowchart depicting steps used to create simulated horizon

The simulated image is created by first calculating an expression for the expected Earth disc shape (Section 3.6.1). The intersection between this shape and the edge of the image plane is then determined (Section 3.6.2), and on this location the first edge pixel's intensity is calculated (Section 3.6.3). This pixel's neighbouring edge pixel is then found by following the line created by the Earth disc, after which the neighbouring pixel's intensity is calculated. This process is repeated dynamically until the opposing image plane limit is reached. The created low resolution image is then blurred to smooth the edge (Section 3.6.4) when simulating a ground test environment, and lastly noise is added (Section 3.6.5).

3.6.1 Determining Disc Parameters

As stated in Section 3.1, the Earth is assumed to be a perfect sphere and therefore the visible Earth disc will always be circular in shape. The visible Earth disc, at an elevation ϕ and rotation θ , is described by:

$$(x - x_0)^2 + (y - y_0)^2 = r_p^2 \quad (3.16)$$

where x_0 and y_0 are determined by utilising Eq. 3.7 and 3.8 respectively.

3.6.2 Intersection Determination between Rectangle and Disc

When determining the Earth disc's intersection with the image plane limits (or individual pixel borders), the problem is simplified by determining the intersection between a circle and a rectangle. This is done by attempting to find the circle's intersection on each of the rectangle's four sides. With each of the rectangle's sides, the x or y value is known thus eliminating an unknown from Eq 3.16.

Assuming the rectangle's top side is described by a line $y = y_r$, the intersection (x_i, y_i) between the disc and this line is determined by solving the quadratic equation as shown in Eq. 3.17.

$$x_i = \pm \sqrt{r_p^2 - (y_r - y_0)^2} + x_0 \quad (3.17)$$

The same is done for the rectangle's right side, for example. Assuming this side is described by $x = x_r$, the intersection (x_i, y_i) is determined by solving Eq. 3.18

$$y_i = \pm \sqrt{r_p^2 - (x_r - x_0)^2} + y_0 \quad (3.18)$$

However, Eq. 3.17 and 3.18 will result in two possible solutions. Therefore, the assumption is made that only one intersection will occur within the rectangle's range as the disc is significantly larger than the rectangle.

3.6.3 Determining Edge Pixel Intensity

Determining the edge pixel intensity P was achieved by determining the ratio between the pixel area covered by the disc and the total pixel area, as shown in Eq. 3.19.

$$P = \frac{A_{disc}}{A_{pixel}} \quad (3.19)$$

The area covered by the disc (A_{disc}) is calculated by approximating the disc's intersection with a straight line. This is done by first determining the disc's intersections with the pixel by utilising the method described in Section 3.6.2, and then creating a straight line between the two intersections, as depicted in Figure 3.18.

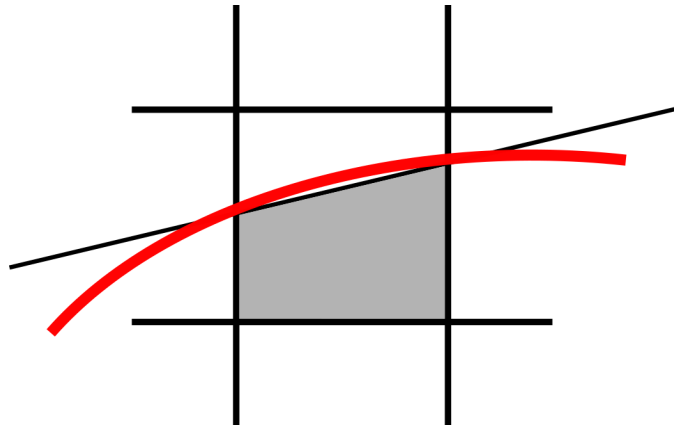


Figure 3.18: Calculating pixel area covered by the disc utilising the straight line approximation

This straight line approximation is sufficient, as the disc is significantly larger than the pixel meaning that the intersecting line resembles a straight line. The area covered by the disc (A_{disc}) is then calculated using conventional area formulas for rectangles and trapezoids. This results in each pixel in the simulated image will having an intensity between 0 and 1.

3.6.4 Edge Smoothing

Figure 3.19a shows the high contrast edge which sufficiently simulates the expected space conditions. Figure 3.7 in Section 3.3.2, shows that ground test conditions created a lower contrast edge, and therefore the simulated edge needs to be modified to fit this behaviour. The edges shown in Figure 3.19 are of an Earth disc at a zero rotation and elevation depicting the high contrast and low contrast image. It is also clear that the disc only covers half of the middle pixel which is expected as there is an uneven number of pixel rows.

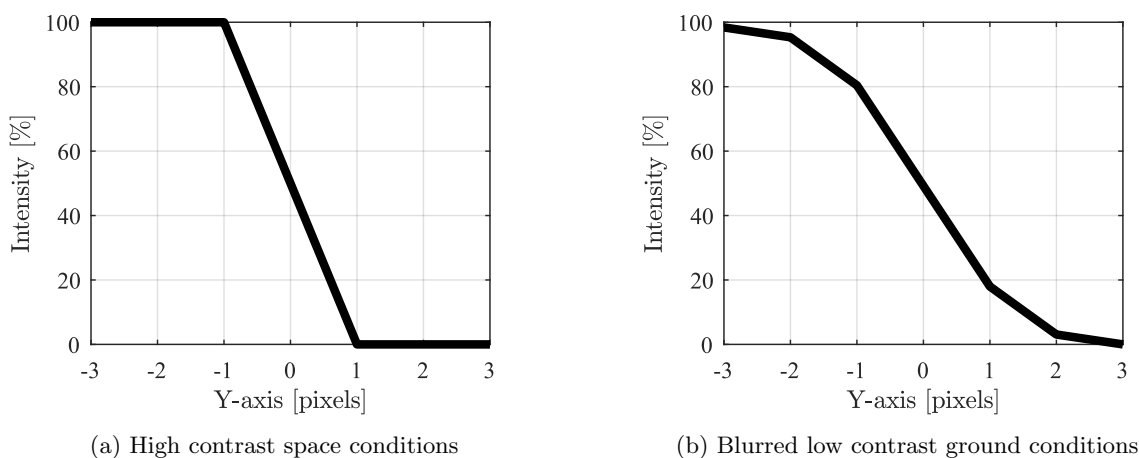


Figure 3.19: Examples of high and low contrast simulated edges

Lowering the contrast of the edge is achieved by blurring the simulated image, which lowers the contrast of the edge as shown in Figure 3.19b. A normalised 5x5 Gaussian Blur kernel is used, which is derived using Eq. 3.20 with a unit standard deviation ($\sigma = 1$). This smooths and extends the simulated edge to more accurately fit the expected image during ground testing conditions.

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \quad (3.20)$$

3.6.5 Adding Noise

Gaussian noise is added to the images relative to the SNR specified. Gaussian noise is chosen as it accurately simulates the noise on the pixels, as seen in Section 3.3.2. The added noise is created with a zero mean and a standard deviation (σ) relative to the SNR (in dB), as shown in Eq. 3.21.

$$\sigma = 1/10^{\frac{SNR}{20}} \quad (3.21)$$

This results in a simulated image as shown in Figure 3.20, where a zero rotation and elevation is assumed and an SNR of 30 *dB*.

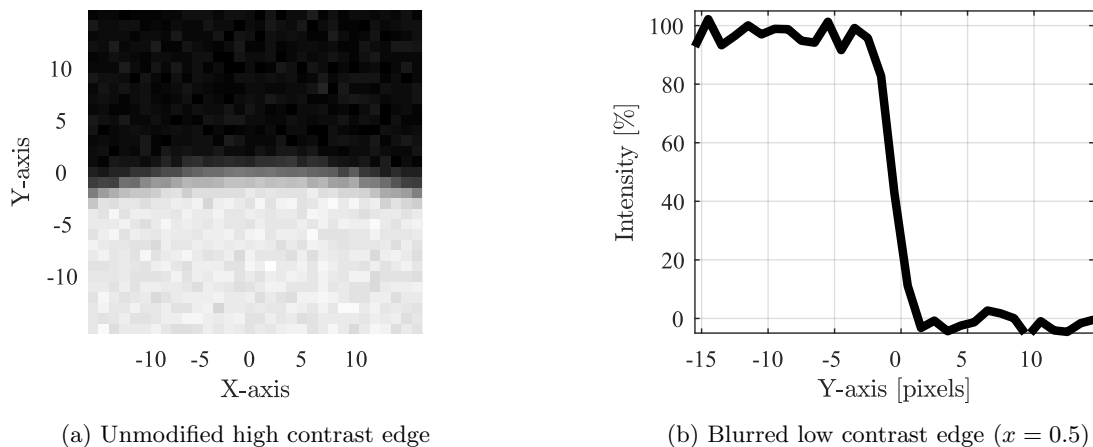


Figure 3.20: Example of simulated image of Earth disc section with added noise during ground test conditions

3.7 Discussion

This section accurately described the imaging environment that a horizon sensor's infrared camera would encounter in space. The shape of the visible Earth is investigated, as well as the properties of the horizon edge itself. After this the expected SNR is calculated, as well as the SNR observed during ground testing. Finally, a method to emulate and simulate the expected horizon is developed which is used for development and evaluation of throughout this study.

4 Hardware Design

This section will investigate the required hardware to build a horizon sensor. It covers the required components, the design of filters and buffer, the PCB layout, as well as the performance of the created circuit. The entire design process will be not described here, but rather the final design used.

4.1 Requirements and Overview

A horizon sensor prototype was built to test the feature extraction algorithms in real-world conditions. This prototype's goal was to prove that such a sensor is viable given the rough conditions of space electronics. Therefore, this prototype is designed with the following main design constraints:

- Low Power
- Space worthy components and design
- Small volume

The circuit was designed to run and control the supplied HTPA32x31 infrared camera[6], retrieve the images, and subsequently calculate the estimated attitude based on the horizon's location. Additionally it was designed to be near space worthy, which requires switches and buffer stages to isolate the different subsystems. An overview of the design is shown in Figure 4.1.

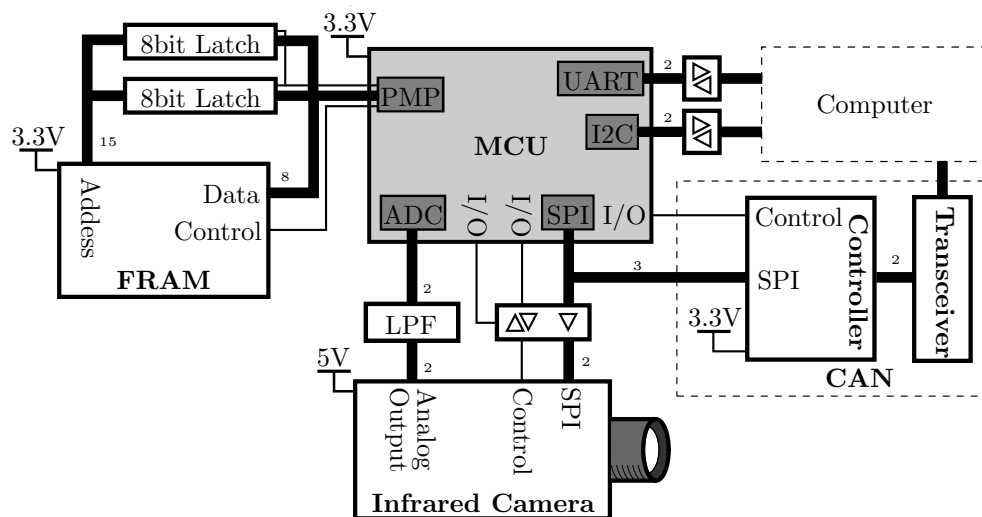


Figure 4.1: Overview of hardware design

The design consists out of three main parts, namely:

- **Microprocessor (MCU):** Controls the sensor's subsystems and does the necessary calculations to determine an attitude estimation.
- **Infrared Camera:** Images the horizon and returns the raw pixel data to the MCU.

- **Communication:** Communication with the master controlling the sensor. Different protocols are implemented as typical space systems requires communication redundancy.
- **External Memory (FRAM):** External memory to potentially store images, as typical low power processors is not equipped enough storage.

Additional circuitry was also designed to aid the development process, which includes (not shown in Figure 4.1):

- Onboard power supplies (3.3V and 5V).
- Input power protection (Reverse Voltage, Over/Under Voltage, Overcurrent).
- Current sensing circuit to measure power consumed by sensor.
- Status LED's and solder jumpers for simple reconfigurability.

The components in the main design were chosen to have significant space heritage. The MCU, communications, and supporting circuitry have flown in satellite subsystems built by CubeSpace. For external memory FRAM [31] is chosen, as it is significantly more tolerant to radiation than conventional memories such as DRAM or SRAM.

4.2 Component Selection

The component selection in this project is essential as it severely influences the possible performance, power usage, and robustness in space conditions. Therefore, great care was given to the choice of components. This section will briefly explain some of the important component choices.

4.2.1 Infrared Sensor

The infrared camera utilised is called the HTPA32x31L5.8k1.0HiS and was developed by Heimann Sensor. This sensor was used mainly due to its availability. However, it does have competitive specifications relative to current thermopile products on the market, as shown in Table 2.1.

Table 4.1: Comparison between available infrared sensors. Data from [6],[12],[13],[14],[15] respectively, as taken on 17/11/2017.

Device	Resolution	Spectral Range	Sensitivity	Frame Rate	Power	Price
Heimann HTPA32x31	32×31	8 → 11.5 μm	< 150 mK	10 Hz	37 mW	N/A
Heimann HTPA80x64d	80×64	8 → 11.5 μm	< 50 mK	20 Hz	82 mW	N/A
FLIR Lepton	80×60	8 → 14 μm	< 50 mK	< 9 Hz	150 mW	\$ 183
Panasonic GRID-EYE	8 × 8	10 μm	2.5 K	10 Hz	22 mW	\$ 22
Melexis MLX90640	32×24	> 15 μm	100 mK	1 Hz	50 mW	N/A

This table shows that the HTPA32x31 has good performance compared to other available products and excels with low power consumption, which then comes with a decrease in sensitivity. However, the sensitivity was not the limiting factor for this specific implementation and low power devices were preferred, which

makes the HTPA32x31 a good choice. The main limitation of this project was the low-resolution, and larger resolutions than shown in this table is not currently available commercially. For future implementations, the HTPA80x63d could be considered as it has 550% the resolution of the HTPA32x31 with only double the power consumption. None of the mentioned devices has known space heritage.

4.2.2 Microprocessor

The MCU is the main workhorse of the design and directly affects the possible performance of the sensor. Therefore, the processor family was chosen to have good space heritage and have sufficient resources for a flexible software implementation. Additionally, to follow typical satellite design, it should have low power capabilities.

It was decided to use the PIC18F47J13, which is an 8-bit processor made by Microchip with nanoWatt XLP Technology[32]. This processor has very low-power states ($< 660 \text{ nW}$) and sufficient memory, timers, ADC channels, etc. Importantly, it has space heritage as it has flown on multiple flights as part of CubeSpace's ADCS devices[10].

4.2.3 External Memory

It is decided to add external memory to the circuit. The MCU itself has 3670 bytes of RAM[32] which proved sufficient, but external memory was added to ensure development flexibility. Therefore, a ferromagnetic random access memory (FRAM) was added because it has excellent robustness against radiation. The critical radiation levels required to degrade the FRAM are beyond those for other memory types[33]. Another great advantage of FRAM is that it has non-volatile memory. The FRAM implemented was an FM28V020 developed by Cypress[34] which has 32kB of memory and a 70 ns access time.

4.3 Circuit Design

This section will discuss the detailed design or design process of various parts of the circuit.

4.3.1 Infrared Camera Interface

The HTPA32x31L5.8/1.0HiM infrared camera[6] communicates with the master MCU (PIC18F47J13) mainly through two-wire SPI and two analogue lines.

Two-Wire SPI: This is used to write the camera's internal register, and in turn, control's the camera's behaviour, i.e. amplification and soft reset. In order to save pins on the MCU, this protocol is emulated by utilising its SPI interface. However, to reduce circuit complexity only the SPI's data out line is connected to the camera, and not the MCU's SPI data in. This means that the camera's internal register can only be overwritten, and not read. This is sufficient, as no valuable information is stored in the camera's internal register.

Analogue Lines: The camera transfers data to the MCU serially through two analogue lines, namely the values off V_{AOTu} , V_{ref} and V_{AORt} . These analogue lines are filtered and the voltage reduced to ensure a clean signal is available for the MCU, as described in Section 4.3.2.

4.3.2 Analogue Low Pass Filter Design

The camera transmits its captured data (V_{AOTu} , V_{off} and V_{AORt}) serially over two analogue lines, of which each has a superimposed ripple voltage caused by the camera hardware[6], as well as inherent system noise. Additionally, the output voltage ranges between $0V$ and $5V$, while the MCU can only measure analogue values up to $3.3V$. Therefore, an active low pass filter was designed to remove this ripple and noise, as well as scaling down the voltage to $3.3V$. A typical output of the camera output is shown in Figure 4.4, where it is represented by V_I .

It was decided to design a modified 2nd-order low pass Sallen-Key filter[35], as shown in Figure 4.2. It is the combination of a typical 2nd-order LPF and an initial voltage divider stage. This was done to save space on the PCB by not having a separate filter and amplification stages. First, the filter was designed by setting R_2 as an open circuit, after which the final gain is adjusted by adding R_2 . Although the resistor R_2 slightly alters the LPF's frequency response, it did influence the design significantly and was considered negligible.

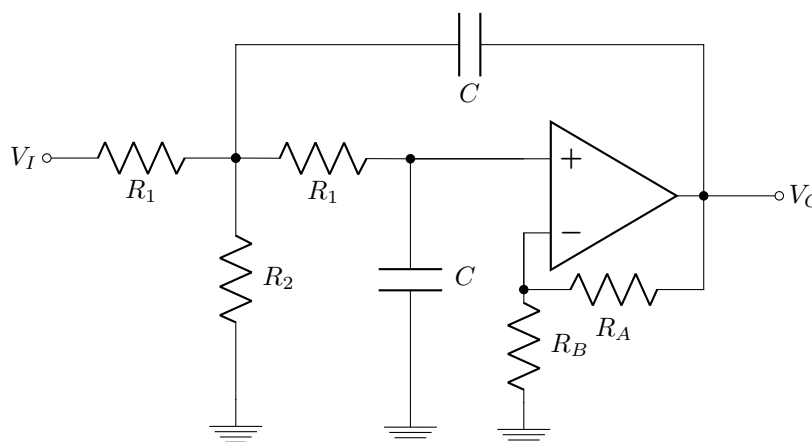


Figure 4.2: Modified 2nd-order Low Pass Filter

The camera datasheet specifies that the amplitude of this ripple is $37mV$, and according to Figure 4.4 the noise has a frequency of roughly $62kHz$. The cutoff frequency (f_c) of the filter was therefore designed to be low enough to eliminate the noise on the system, but high enough to ensure the induced rise time fits in the $200\mu s$ acquisition period[6]. It was decided that the filter should be optimally damped with $\zeta = 0.707$. This results in Eq. 4.1, which determined the filter's quality factor as $Q = 0.707$.

$$Q = \frac{1}{2\zeta} \quad (4.1)$$

The required filter stage's gain A was then calculated with formula Eq. 4.2 as $A = 1.585$.

$$A = 3 - 2\zeta \quad (4.2)$$

However, the gain A is determined by discrete resistors R_A and R_B , which was chosen as $R_A = 33k\Omega$ and $R_B = 56k\Omega$. This results in the true gain A being equal to $A = 1.589$, according to Eq. 4.3

$$A = 1 + \frac{R_A}{R_B} \quad (4.3)$$

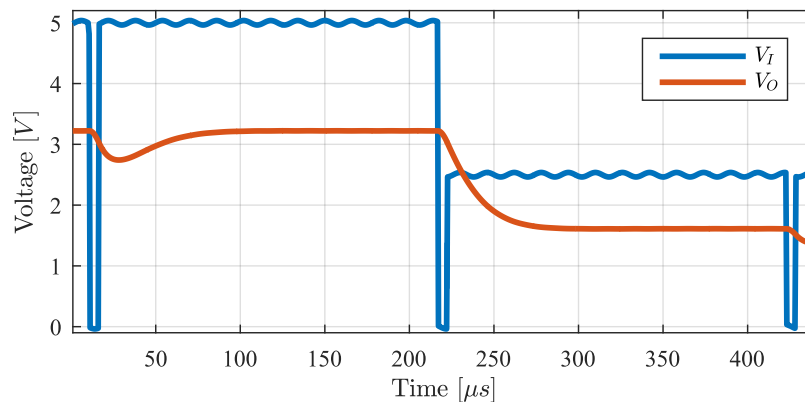
The total gain (G) of the LPF is required to convert a $5V$ swing to a $3V$ swing, and which results in a gain of $G = 3.3/5 = 0.66$. Taking the voltage division between R_1 and R_2 into account the gain total gain of the LPF (G) can be approximated with Eq. 4.4.

$$G = \frac{AR_2}{R_1 + R_2} \quad (4.4)$$

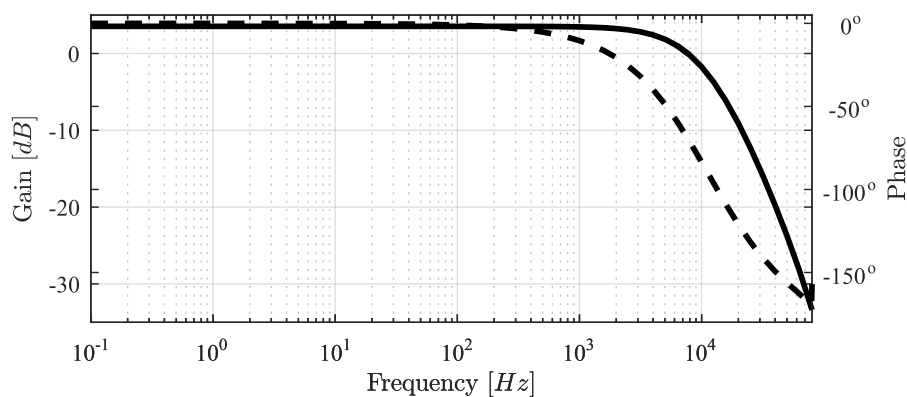
Assuming that $C = 1 \text{ nF}$, it was decided to choose $R_1 = 22 \text{ k}\Omega$ and $R_2 = 15 \text{ k}\Omega$. utilising Eq. 4.5 this resulted in $f_c = 7.2 \text{ kHz}$ which was low enough to ensure sufficient noise cancellation while the rise time is fast enough, as proved in Figure 4.3a.

$$f_c = \frac{1}{2\pi R_1 C} \quad (4.5)$$

Therefore, utilising Eq. 4.4 the total gain was determined to be $G = 0.644$, which is sufficiently close to the required 0.66 . The LPF was simulated in LTSpice to ensure it behaves as expected. During camera operation, the typical change between two serially transmitted values (V_I) does not exceed 500 mV , but to ensure working behaviour under all conditions a change of 2.5 V was simulated. The results are shown in Figure 4.3a, where it is clear that a 2.5 V change in V_I is followed with ease.



(a) Time Response



(b) Frequency Response

Figure 4.3: Results of simulated Low Pass Filter

Figure 4.3b shows that the expected cutoff frequency is at 6.3 kHz , which is lower than the designed cutoff frequency of $f_c = 7.2\text{ kHz}$ because of the initial voltage divider affecting the filter stage. However, this is still sufficient as the ripple is eliminated while the filter's rise time is inside the acquisition period. This behaviour is also mimicked on the implemented circuitry as shown in Figure 4.4. It is also clear that the gain G of the system is approximate $G = V_O/V_I = 0.64$, which corresponds to the expected value according to Eq. 4.4. Therefore, the designed low pass filter performs as designed.

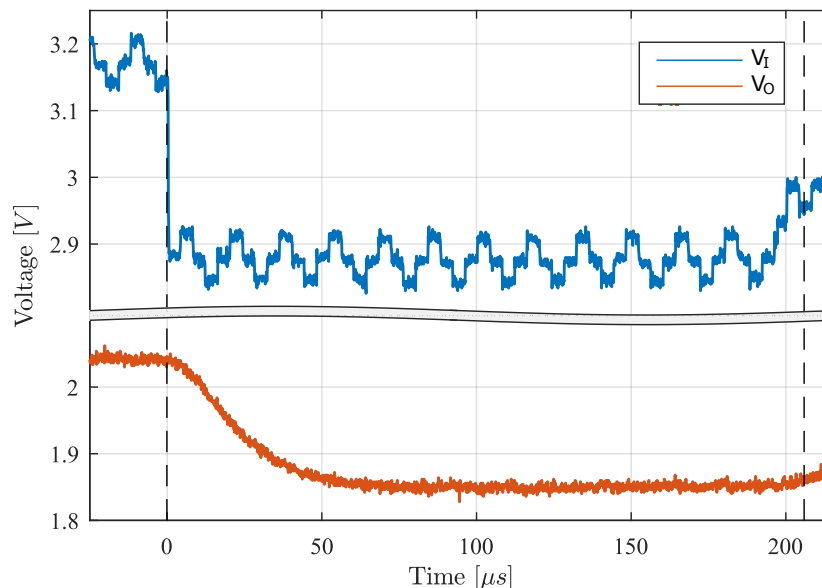


Figure 4.4: Measured performance of the low pass filter. The dashed lines (V_{SAM}) represents the start of the serial analogue output

4.3.3 Power Planes and PCB Design

Noise on the power or data lines between the camera and MCU will alter the SNR of the system, and in turn, decrease the available accuracy. Therefore, steps were taken in the circuit and PCB design to minimize noise the system, which included:

- **Isolated Power Rails:** The infrared camera's V_{ADD} was isolated from the MCU's V_{DD} by a ferrite bead and multiple capacitors. This was done to eliminate digital noise (induced by the MCU) on the power lines connected to analogues devices.
- **Clean Return Paths:** The current return paths (through ground) of the analogue lines were designed to be kept clear to reduce travel length.
- **Large Power and Ground Rails:** Power and ground plains were designed to occupy maximum space and ensure return paths for the analogue line, as well as not creating large isolated 'islands'.
- **Subsystem Sectioning:** Subsystems (e.g. power regulation, digital and analogue) were grouped to reduce noise cross-contamination.

4.3.4 FRAM Interface Design

External memory was added to the system to ensure sufficient storage space is available to store the required image data. An external 32kB FRAM was added which communicates in parallel, which means that a 15-bit

address line and an 8-bit data line is required between the MCU and the FRAM. The MCU is equipped with a Parallel Master Port (PMP) which can transmit all the required data through 8 data lines and temporarily store the address in two octal latches, as shown in Figure 4.1. The resultant read/write speed is roughly $5 \mu s$ per byte, which is significantly slower than the using the MCU's internal RAM. However, as described in Section 8.1 the external memory was not required for the attitude estimation algorithms.

4.4 Power Consumption

As previously mentioned the sensor is required to low power, and therefore the power consumption of the device was investigated by measuring the current draw of the 3.3 V and 5 V power rails separately using onboard current sensors. Although the processor was not actively estimating the attitude during this current measurement, it delivered a sufficient estimation of normal operation power usage. This is because the MCU was running at full pelt with no power saving features enabled. Also, the infrared camera continually transmits images when turned on. The results of these current measurements are shown in Table 4.2, while the different subsystems using each power supply are listed in Table 4.3.

Table 4.2: Maximum power usage of different power supplies

	3.3 V	5 V	Total
Current Draw	20.4 <i>mA</i>	11.2 <i>mA</i>	31.6 <i>mA</i>
Power Draw	67.3 <i>mW</i>	56 <i>mW</i>	123.3 <i>mW</i>

Table 4.3: Subsystems used by each power supply during measurement

3.3 V	5 V
MCU	Camera
FRAM	LPF
Communications	
Current Sensors	

It was found that the camera's power draw increases by roughly $500 \mu A$ (or $2.5 mW$) as the internal circuitry reaches thermal equilibrium. This is due to the camera's internal resistances increasing with temperature during ground tests (e.g Sections 5 and 7), which is also affected by the radiating heated steel plate. However, the MCU's power draw remains constant during operation.

In the final product the power consumption could be decreased further, as the camera is only required for roughly 55% of the time, as discussed in Section 8.3. When the camera is not required it can be either turned off through an external switch, or be set to a soft reset state. The processor can also be put into a low-power state when not in active use, but the this will only be roughly 10% of the total time. Assuming the camera and LPF is disabled while not in use (45% of the time), it will reduce the total average power draw to $92 mW$. The only infrared horizon sensor currently available, called the MAI-SES (discussed in Section 2.1 and 2.3.3), has a power usage of roughly $264 mW$. This is roughly 280% more than the device being developed, as shown in Table 4.4.

Table 4.4: Possible power usage comparison to MAI-SES[3]

This Project	MAI-SES
92 <i>mW</i>	264 <i>mW</i>

4.5 Disussion

The hardware designed for this project was successful. It fulfilled all the design requirements and performed consistently under various conditions such as under vacuum or while being heated or cooled. The noise present on the system didn't severely influence results, but could still be improved on future iterations. Below in Figure 4.5 the final circuit is shown. Only components selected in red is necessary for a final space implementation (unselected components are not used or typically supplied by satellite bus).

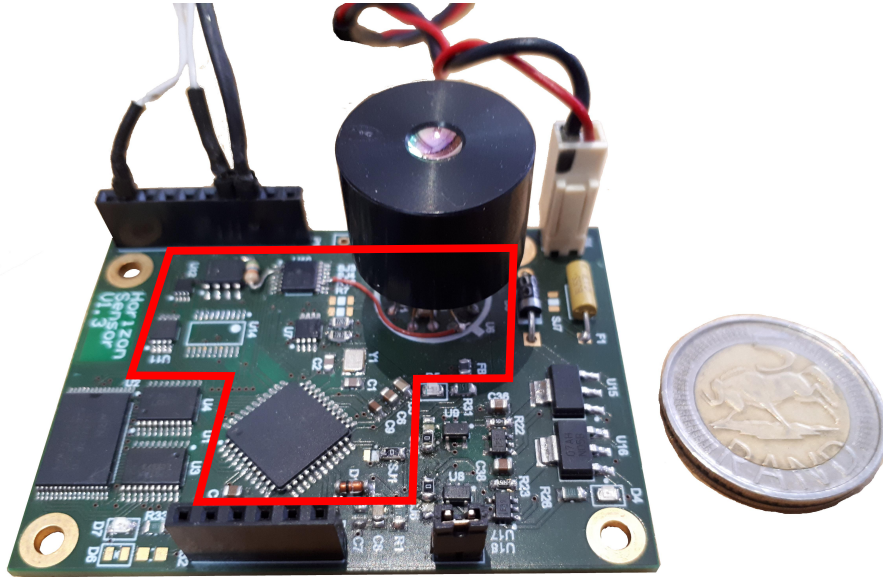


Figure 4.5: Image of designed hardware with essential components shown shown in red

5 Thermal Pixel Calibration

The HTPA32x31L5.8/1.0HiM infrared camera, manufactured by Heimann Sensor GmbH[6], uses a 32x31 array of thermopile elements that convert the temperature radiation of an object surface to an electrical voltage using thermocouples. This process is called the Seebeck effect[36]. Each of the 992 thermopiles is run at a slightly different reference voltage, and therefore each one has a slightly different response to temperature change which is caused by non-ideal manufacturing. The camera is equipped with a lens which might not be uniformly transparent, which could also affect the behaviour of different pixels. These irregularities need to be accounted for with calibration techniques to ensure uniform pixel behaviour. Heimann proposes a method of thermal calibration which will ensure uniform pixel behaviour and translate this information into temperatures.

5.1 Detailed Explanation

This section will show the relationship between each pixel's output and the imaged object's temperature, as well as techniques (given by Heimann[6]) on how to model this relationship through calibration. Each pixel's output voltage is a function of the object temperature, emissivity, the sensor's temperature and the surrounding temperature. This is described by the following equation:

$$V_{obj} = \epsilon k (T_{obj}^n - T_S^n) \quad (5.1)$$

where,

- V_{obj} = Thermopile Output voltage
- k = Constant apparatus factor
- ϵ = Object emissivity
- T_{obj} = Object temperature
- T_{amb} = Ambient (Surrounding) temperature
- T_S = Sensor (Housing) Temperature
- n = Exponent to describe the temperature dependency of the signal voltage

The value of n is theoretically 4, but in practice, it has been empirically determined to be between 3 and 4. However, using a value of 4 is still sufficient in most applications as it does not significantly impacting the measurement tolerances. It is also assumed that $T_S = T_{amb}$ as the camera housing reaches a thermal equilibrium with the ambient temperature after a certain amount of time. Converting the sensor output voltage (V_{obj}) to a temperature in Kelvin (T_{obj}) the following equation is used:

$$T_{obj} = \sqrt[n]{\frac{V_{obj}}{\epsilon k} + T_S^n} \quad (5.2)$$

where V_{obj} is determined from the camera's analogue output voltage V_{AOTu} using the following formula:

$$V_{obj} = V_{AOTu} - V_{ref} - V_{off} \quad (5.3)$$

where V_{AOTu} is the uncompensated sensor output voltage and V_{ref} the reference voltage (also given by camera). V_{off} (also known as V_{th} in some literature) is an additional offset created by internal temperature gradients (application specific), and is empirically determined (discussed in Section 5.2.1).

In order to determine T_{obj} the housing temperature (T_S) is required for which the camera has an embedded module to measure T_S and convert it to an output voltage V_{AORt} (also called PTAT in some literature). The conversion between voltage and temperature is experimentally determined for each camera. The following equation describes this module's behaviour:

$$T_S = \frac{1}{S_T}(V_{AORt} - V_{AORt@25}) + 298.15 \quad (5.4)$$

where S_T is the temperature sensitivity of the internal temperature reference, and $V_{AORt@25}$ is the measured temperature output V_{AORt} at $25^\circ C$.

5.2 Calibration Steps

This section shows the calibration steps as described by Heimann. All tests were run in a thermal controllable vacuum chamber which was made available by Stellenbosch University. The vacuum chamber was chosen to simulate space conditions by minimising the effects of air particles which might interfere with measurements. An additional reason is that water vapour might condensate on the camera electronics during thermal cooling, which could interfere - or damage - the circuit behaviour.

The following guidelines were also taken into account during calibration:

- The camera has two operating modes: low and high pixel voltage amplification. All tests were done for both instances, except for determining the housing temperature which is not affected by the pixel amplification.
- The camera electronics produce heat when active. Therefore, the camera was activated and capturing dummy images for at least 1 hour before the sample recording was initiated. It was tested beforehand that the camera reaches thermal equilibrium in the atmosphere after roughly 30 minutes.
- Each image sample recorded was created by averaging 15 images taken by the camera as a way to eliminate noise.
- A thermal control system for the Thermal Vacuum chamber's heating and cooling elements, which could handle a small device, was not available at the time of testing. Therefore, a Bang-Bang Thermal Controller was written for the chamber (not discussed in this study).

The camera was mounted in the thermal vacuum chamber pointing downwards as shown in Figure 5.1. In this configuration the plates on either side of the camera could be heated to change the temperature of the camera. The bottom plate could be either heated or cooled. This means the object in view could be heated with the camera during the test described in Section 5.2.1, and then cooled separate of the camera as described in Section 5.2.3.

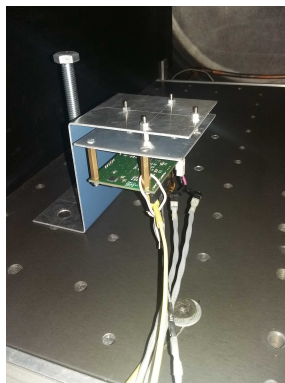


Figure 5.1: Camera setup to determine V_{off}

5.2.1 Determining Offset Voltage

The offset voltage V_{off} quantizes the voltage offsets on individual pixels, which is generated by internal temperature gradients and dependent on application specific influences. The value is different for each pixel and varies with the housing temperature. According to Heimann when determining V_{off} the housing and object temperatures should be kept equal ($T_S = T_{amb}$), and the object should ideally have an emission coefficient close to unity. This results in $V_{obj} = 0 V$ according to Eq. 5.1 which simplifies Eq. 5.3 to:

$$V_{off} = V_{AOTu} - V_{ref} \quad (5.5)$$

Heimann advises mounting the camera in a tempered fitting, i.e. a blind hole. The end plane of the blind hole should have a high emission coefficient, ideally a black anodised surface with a pyramid structure. Such a blind hole was not available during testing. Therefore it was decided to place the camera in a vacuum chamber and pointed at the black surface of a heating element. Such a setup has an emission coefficient of nearing unity but is sufficient for this application as absolute temperatures do not need to be calculated, only differences in temperature.

The thermal control system of the vacuum chamber was set to start at $25^{\circ}C$, and when the temperatures of both the camera and the object in view were equal, a set of 15 images were taken and averaged. Ideally, a lower starting temperature should be used to ensure the camera is calibrated over the $0-25^{\circ}C$ operating temperature, but the test setup did not allow sufficient cooling. Both the chamber's and camera's temperatures would then be incremented by $5^{\circ}C$, and a set of images retaken as the temperatures stabilise. This process was repeated until the chamber reached $50^{\circ}C$.

5.2.2 Determining Housing Temperature

As discussed previously, the housing temperature was required to determine an object's temperature accurately. The camera has an embedded module that measures the housing's temperature and converts it to a voltage, which can then be read by the external microcontroller. The conversion from object voltage to temperature is completed either empirically or by using Eq. 5.4. For this test, the camera was heated in the vacuum chamber. During the heating process temperature samples were taken of the camera V_{AORt} , as well as the housing temperature measured by an external thermometer. These results were used to find a linear approximation of the ambient temperature.

5.2.3 Determining Object Temperature

To determine the object temperature T_{obj} , the values for ϵ and k are required, as shown in Eq. 5.2. It was assumed that the emissivity $\epsilon = 1$, which is inaccurate as the imaged object does not have perfect emissivity. However, this assumption only scales the final value for T_{obj} , which is adequate as only a temperature difference is detected in the final product, and not an absolute temperature. The value for k is then determined for each pixel by capturing images of objects at different temperatures. Using the theoretical value of $n = 4$, Eq. 5.2 is then be used to calculate k as it is then the only unknown variable.

For this test, the camera was placed in the vacuum chamber as shown in Figure 5.1. Heating plates on either side of the camera were activated to heat the camera, while the bottom plate viewed by the camera was cooled. This ensured a high temperature difference between the camera and the plate.

5.3 Results

5.3.1 Determining Offset Voltage

The offset voltage V_{off} was determined by using a thermal vacuum chamber as described in Section 5.2.1. The thermal vacuum chamber's temperature did not follow the desired temperatures exactly but provided sufficient data to determine a rough estimate. The temperature curve of the camera and the imaged object is shown in Figure 5.2, as well as when the pixels were sampled.

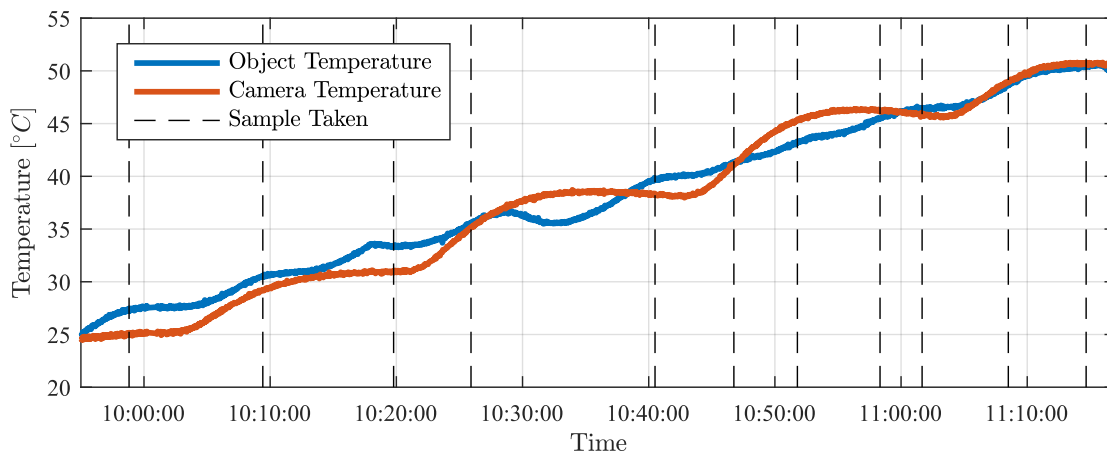


Figure 5.2: Temperature response of thermal vacuum chamber, as well as the times sampled

The results obtained for the tests are shown in Figure 5.3, which shows the average value of all the pixels ($V_{AOTu} - V_{ref}$) over a change in temperature (represented by V_{AORt}). A clear inverse proportionality is observed, albeit noisy. This is due to the temperatures of the camera and imaged object not being exactly the same; as well as sampling not being instantaneous as it was sampled for roughly 30 seconds for each data point.

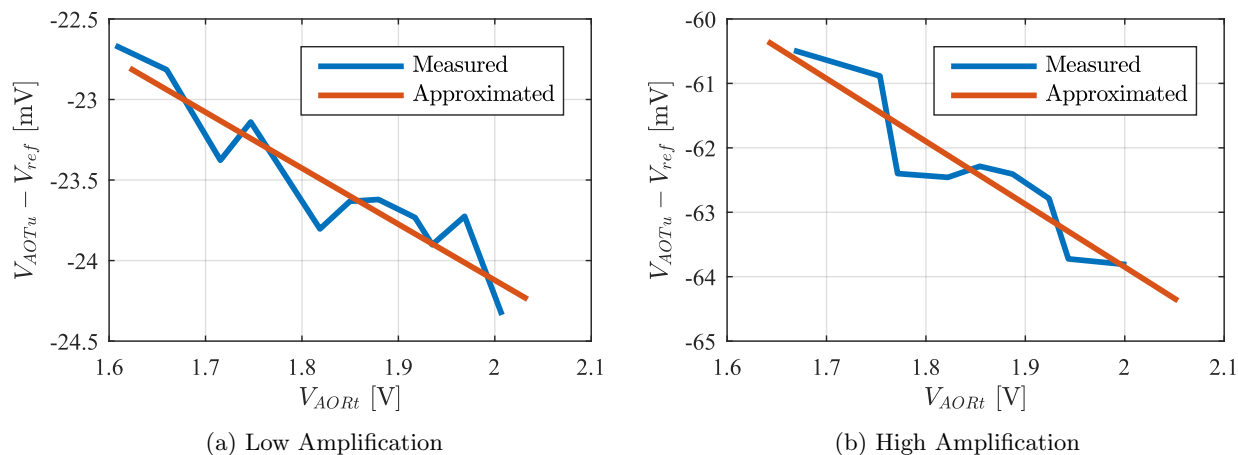


Figure 5.3: The average pixel values ($V_{AOTu} - V_{ref}$) for varying V_{AORt} values acquired by using the Vacuum Chamber

According to Eq. 5.5 the offset voltage V_{off} is equal to the measured value for $V_{AOTu} - V_{ref}$ over different values for V_{AORt} . Therefore, a V_{off} model for each pixel is created with a linear approximation over the V_{AORt} range, as indicated by the red line in Figure 5.3. This V_{off} model's performance was then tested by determining V_{obj} of all the pixels in data used, using Eq. 5.1. This was done for each V_{AORt} , and is shown in Figure 5.4. The calculated V_{obj} values are all close to zero as expected, with the deviations in the mean value correlating with the deviations in Figure 5.3.

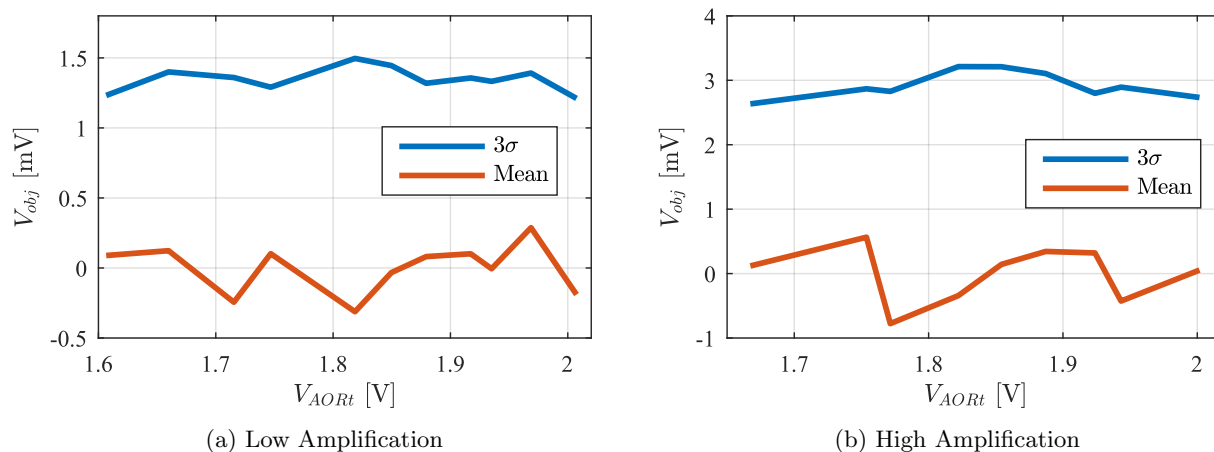


Figure 5.4: Calculated V_{obj} values using the approximated V_{off} model over the expected V_{AORt} range

5.3.2 Determining Housing Temperature

To determine the camera housing temperature T_S , as described in Section 5.2.2, the data used for determining the V_{off} model was used in conjunction with Eq. 5.4. Data captured for thermal calibration tests was used, because the camera housing temperature was also logged externally, as well camera's embedded temperature module output V_{AORt} . The housing temperature measured at different V_{AORt} values is shown in Figure 5.5. The small deviations in the measured data is caused by small inaccuracies when logging the temperature by hand.

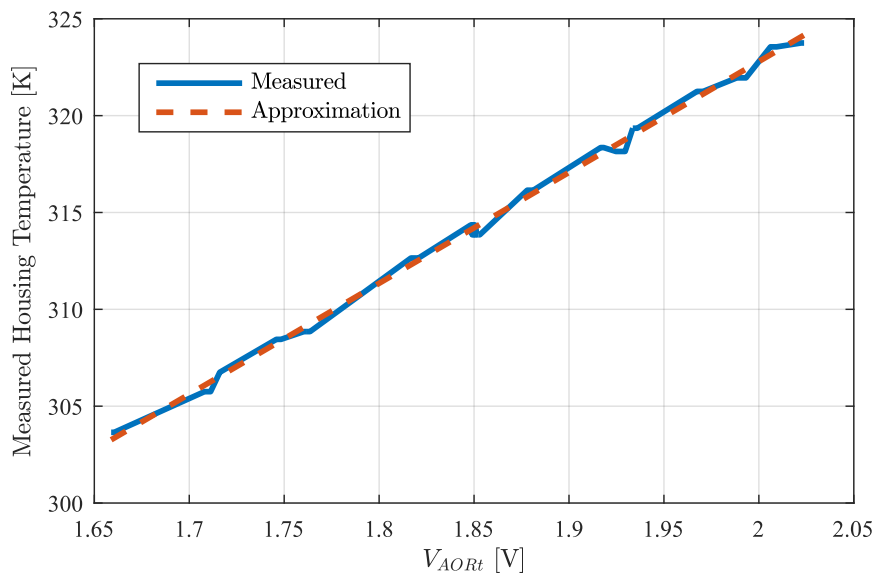


Figure 5.5: Approximation of V_{AORt} vs. T_S

It is clear from the figure that a linear relationship exists between the voltage V_{AORt} and the camera temperature T_S . This relationship is quantified by implementing a Least Squares straight line fitting of the points, which is plotted in red in Figure 5.5. This results in the housing temperature being calculated by Eq 5.6, which is a manipulation of Eq. 5.4.

$$T_S = mV_{AORt} + c \quad (5.6)$$

with $m = 57.25 \text{ K/V}$ and $c = 209.29 \text{ K}$ as determined by the Least Squares method. According to Eq. 5.4 this fitted line's gradient is equal to the reciprocal of the temperature sensitivity S_T , which results in $S_T = 17.46 \text{ mV/K}$, and in turn $V_{AORt@25} = 1.57 \text{ V}$. Therefore, there is sufficient information to determine T_S when V_{AORt} is known.

5.3.3 Determining Object Temperature

The value k was determined for each pixel as described in Section 5.2.3. These tests started with both the camera and object at 50°C at which the bottom cooling plate was activated to lower the object's temperature. This is shown in Figure 5.6, as well as the times when samples was taken.

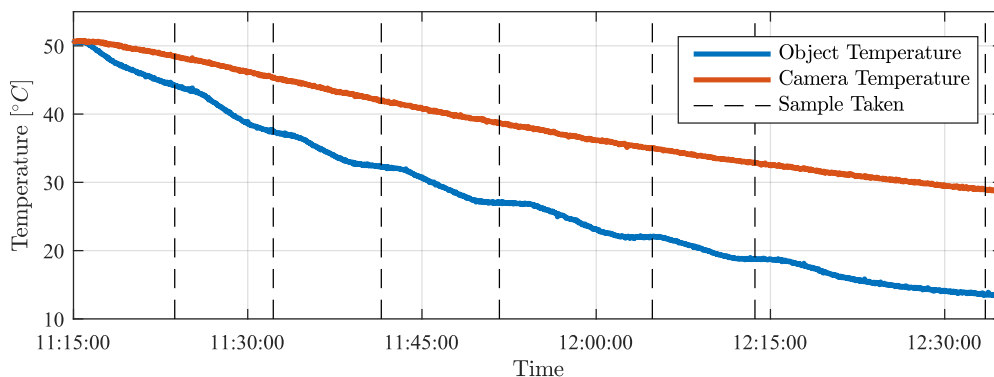


Figure 5.6: Temperature response of thermal vacuum chamber, as well as the times sampled

The value for k was then calculated for every pixel in every captured sample using Eq. 5.4. All the samples' k values were then averaged for each pixel, resulting in a 32×31 matrix of k values. Figure 5.7 shows an example of the calculated k value of all the samples (blue), as well as the approximated k value (red) for one pixel.

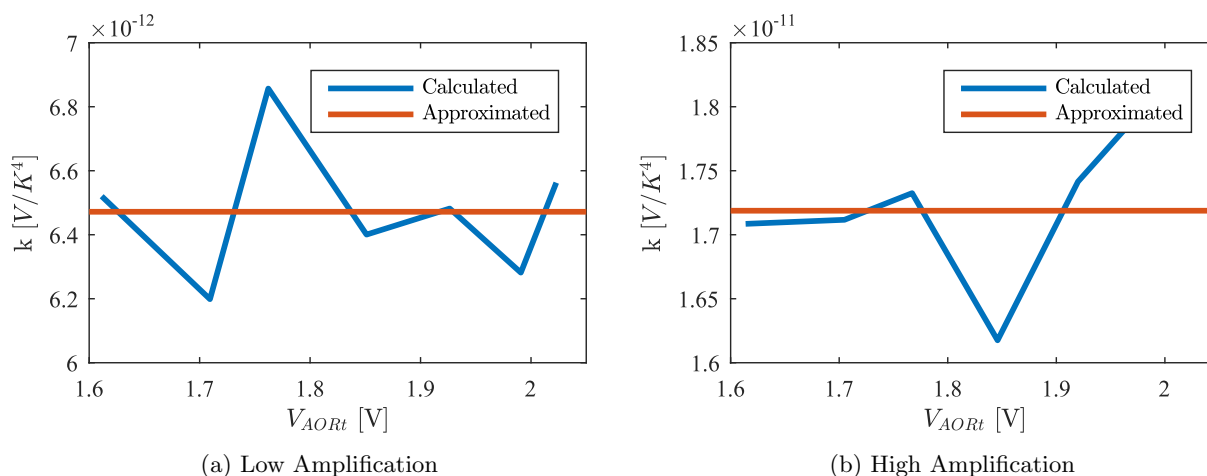


Figure 5.7: Calculated and approximated k value for pixel (14,14)

It is clear from this graph that the k value is 2.6 times higher when using the camera's high amplification setting, which is close to specified amplifier gain of 3 according to the camera datasheet[6].

5.4 Discussion

This section showed the successful thermal calibration of the infrared camera to ensure a uniform response from all the pixels. This was done by empirically determining each pixel's V_{off} which results in a uniform image in Volts. It was decided not to calculate the image relative to the object temperature as this requires an unnecessary complex calculation when only a difference in pixel intensity is required. This also has the advantage of creating higher contrast horizon edge (see Figure 3.5) which will result in more accurate measurements in space.

6 Horizon Location Extraction

This section will discuss and evaluate various methods with which to determine the location of the Earth in an image using feature extraction techniques. Included in this investigation is the accuracy achievable with different SNR levels (expected SNR discussed in Section 3.3). The investigation was done in three stages, namely:

- **Edge Detection Techniques:** Locating an edge in a straight line of pixels to an sub-pixel accuracy. (Section 6.1 and 6.2)
- **Scanning Techniques:** The order and direction in which to scan the images, using an edge detection technique, to find edge points on the visible horizon. (Section 6.3 and 6.4)
- **Shape Fitting Techniques:** Fitting the horizon shape to the edge points which is created using the previous two stages. (Section 6.5 and 6.6)

For each of these stages different methods were proposed, compared, and the best technique chosen. The techniques were tested with ground testing conditions, and with space conditions, to sufficiently evaluate the algorithms in a physical setup.

6.1 Available Edge Detection Techniques

The accuracy of this horizon sensor is primarily limited to how accurately the horizon edge can be detected on the image taken by the infrared camera, which is directly proportional to the camera's resolution. A single pixel represents approximately 2° which would, in turn, limit the accuracy to a minimum of 2° . However, the goal of this sensor was to have an accuracy of roughly 0.1° , which meant that extra steps had to be taken to improve the accuracy.

There are two main ways of increasing the accuracy limitation caused by a camera's resolution:

- Increasing the image resolution using software.
- Extracting features from an image with sub-pixel accuracy.

There are however several limiting factors to consider when investigating and choosing a method to increase the accuracy:

- **Limited Performance.** The chosen method will need to be implemented on the chosen 8-bit processor and be executed in less than a second. This means there is a severe limitation in computational complexity.
- **Non-uniform horizon edges.** As described in Section 3.2 the temperature between space and the Earth will vary with seasons, region, and altitude. This will result in the edge created between the warm and cold pixels varying, and therefore the system should preferably be designed to be independent of the created edge and temperature difference.
- **Robustness to noise.** The inherent system noise should not significantly influence results.

The problem statement can be simplified by determining what the infrared camera is expected to observe while in space. As described in Section 3.1 this is best described by a circle representing the Earth disc. The pixels covered by the circle will have a high intensity (warm Earth), while the rest of the pixels will have a low intensity (cold space). Therefore only the location of the edge between *warm* pixels and the *cold* pixels

is required.

6.1.1 Super Resolution

This method is used to create high resolution images by using multiple low resolution images on aircraft[37]. The movement and vibration of the camera mounted on the aircraft ensures that multiple images taken of the same target is taken from slightly different locations. The registration between the images is determined, and the images are stitched together to create a reconstructed high resolution image. This is illustrated in Figure 6.1. This method only increases the available resolution, and will need to be used in conjunction with another edge detection technique described in this section.

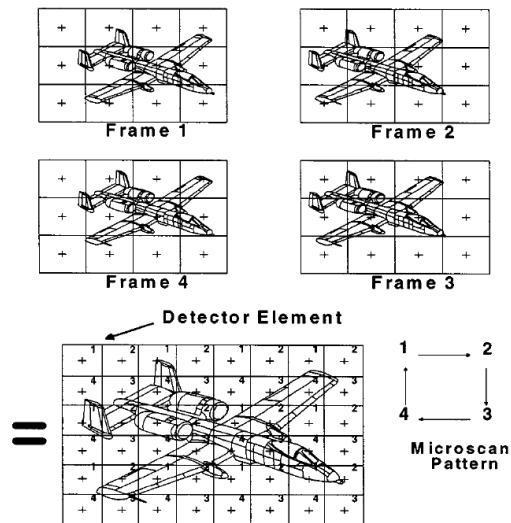


Figure 6.1: Illustration of the superresolution process

There are no significant movements (vibrations, rotations) on typical CubeSats which limits the image registration and therefore the maximum achievable resolution. More importantly, reconstructing a high resolution images is severely processing intensive, and is not viable on a low power 8-bit processor. This method will therefore not be investigated further.

6.1.2 Linear Approximation

In this method the edge between the *warm* and the *cold* pixels are approximated linearly by using multiple sequential pixels to determine x , where the edge crosses $\Delta p/2$ in intensity, as shown in Figure 6.2. This is done by calculating the intensity difference between pixel p_0 (located in the center of the edge) and half the pixel intensity range ($\Delta p/2$), and then scaling it relative to the edge length d .

This is done by locating the pixel in the center of the edge, which in the figure is assumed to be at location $x = 0$. This can be done in multiple ways, but the most robust way is by using a gradient image (e.g. by applying a Sobel Filter, described in Appendix A, to the original image) and locating the pixel in the selected pixel range with the maximum gradient. This pixel's intensity will be called p_0 .

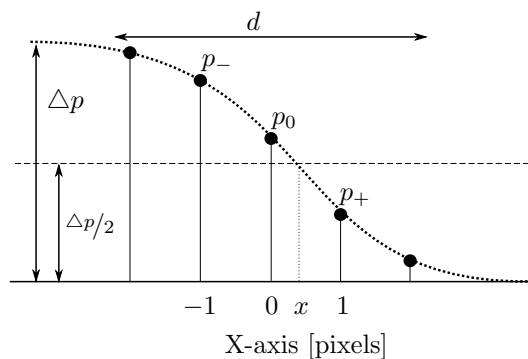


Figure 6.2: Linear approximation pixel definitions

The length of the edge d is approximated by linearly scaling the distance between pixels p_- and p_+ (2 pixels) with a ratio defined by the intensity range (Δp) divided by their difference in pixel intensities ($p_- - p_+$). This is shown in Eq. 6.1.

$$d = 2 \left(\frac{\Delta p}{p_- - p_+} \right) \quad (6.1)$$

The offset x is determined by scaling the edge length d by the ratio defined between the measured error in pixel intensity ($p_0 - \Delta p/2$) and intensity range Δp . This results in Eq. 6.2.

$$x = d \left(\frac{p_0 - \Delta p/2}{\Delta p} \right) \quad (6.2)$$

As described in Section 3.2 the expected edge will be sigmoid shaped, and not linear, which could introduce an estimation error in calculating d and x . This method requires the intensity range (Δp) which is complex to quantify and might result in ambiguous results, and inaccurate estimations can drastically bias results. This formula is straightforward to implement, could provide sufficient accuracy and will therefore be investigated further.

6.1.3 Neighbouring Pixel Utilisation

This is a method developed by Jansen Van Rensburg [38] while prototyping an infrared Horizon Sensor. In this method each pixel (or parent pixel) is subdivided into several smaller sub-pixels, for example a 10×10 sub-pixel matrix. The sub-pixels which are located on the horizon edge are then identified by comparing the parent pixel's neighbouring pixels and approximating a line with the equation $y = mx + c$. The pixels on the left and right of each parent pixel influences the gradient of the line (m), and the top and bottom pixels influences the height of the line (c). This is illustrated in Figure 6.3.

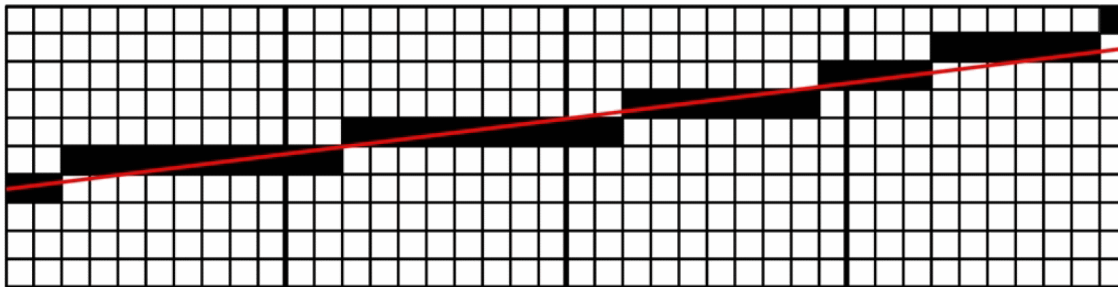


Figure 6.3: Illustration of 4 pixels with their corresponding created sub-pixel matrices

This method is complex while only mimicking a linear approximation (described in Section 6.1.2). It also limits the accuracy to 0.1 pixels and increases the required program memory significantly by storing redundant information. This method also assumes the horizon can be approximated with a straight line, which results in a non-ideal fit. This method will therefore not be investigated further.

6.1.4 Center of Mass

In this method the location of the feature is approximated by determining the center of mass (referenced by CoM) of the edge. This is done by identifying the sequence of pixels present in the edge, and calculating the center of mass of underneath this edge, as shown in Figure 6.4.

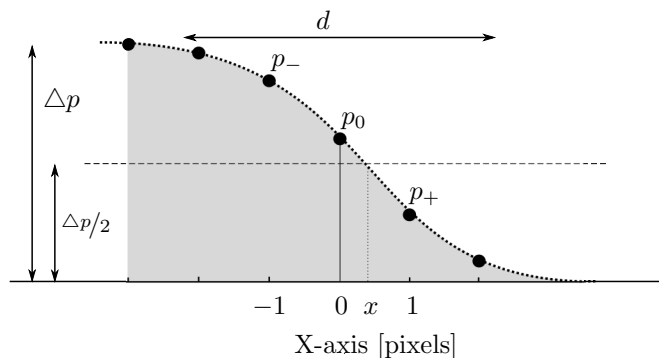


Figure 6.4: Center of Mass pixel definitions

First the edge is located, which is done by finding the pixel in the center of the edge, similar as described in Section 6.1.2. After this the edge length d is approximated using Eq. 6.1. All pixels within $d/2$ pixels distance of pixel p_0 are then selected as edge pixels. Erratic behaviour is present when one pixel is on the limit of being within the range d , and it was found that padding d with 0.5 pixels on either side minimizes the erratic behaviour.

These pixels' intensities ($p_i = [..., p_-, p_0, p_+, ...]$) and their respective indices ($x_i = [..., -1, 0, 1, ...]$) are then used to calculate the CoM using Eq. 6.3.

$$x = \frac{\sum p_i x_i}{\sum p_i} \quad (6.3)$$

This method does not return the required edge center, but rather the edge's CoM, and therefore results in a

measurement error and will have to be corrected and calibrated. Erratic behaviour might still occur as pixels move over edge defined by d . It is however simple to implement and robust to varying edge shapes and noise, therefore this method will be investigated further.

6.1.5 Local Extrema

This method, described by Baily [39], locates the center of an edge by fitting a model to the edge, and finding the local extrema of the fitted model. Baily proposes that the parabolic model is best for edges larger than 1 pixel in length. However, the edge investigated is not parabolic and more sigmoid shaped, but it can be manipulated to be parabolic by calculating the gradient image. The edge gradient will be parabolic shaped at the peak, which location corresponds to the middle of the original edge. This is shown in Figure 6.5, with the dotted line representing the edge, the solid line the edge's gradient, and x the center of the edge.

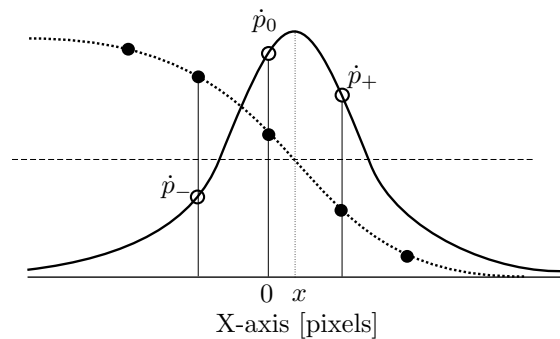


Figure 6.5: Local extrema method pixel definitions

The gradient image is calculated by using a Sobel Filter (see Appendix A) which is described in Appendix A. The maximum pixel value (\dot{p}_0) on the edge's gradient is then found, which represents the pixel nearest to the middle of the edge. A parabolic model is then fitted to the center three pixels ($\dot{p}_-, \dot{p}_0, \dot{p}_+$), and the parabola's local extrema location is then calculated by finding where the edge's gradient is zero, using Eq. 6.4.

$$x = \frac{\dot{p}_+ - \dot{p}_-}{4\dot{p}_0 - 2(\dot{p}_+ + \dot{p}_-)} \quad (6.4)$$

This method is very robust, as it only requires finding the highest point on a gradient image, irrespective of the temperature difference between the *warm* and *cold* pixels. The Sobel Filter is relatively complex but can be accurately approximated (discussed in Section 8.2.4), and the rest of the calculations are simple. Additionally, this gradient image is created by incorporating diagonal gradients, which results that this method will be more resilient against rotations than the other methods. The downside of this method is that it detects the maximum gradient, which doesn't necessarily correspond directly to the edges' center, but rather slightly offset. However, the magnitude of this offset can be determined empirically if the imaging conditions are known. This method will be investigated further.

6.2 Comparing Edge Detection Techniques

In this section, the chosen methods described in Section 6.1 will be compared in order to select the best method for this specific application. The following critical and limiting factors of the different methods will be compared:

- Accuracy
- Robustness to noise
- Robustness to different environments
- Implementation simplicity

The desired accuracy of the horizon location is ± 0.1 . In the final product the mentioned methods will be run on each captured image multiple times, e.g. once in each column, to find different coordinates on the horizon edge. Combining all these coordinates to find a single expression for the horizon will result in a greater accuracy than each individual coordinate has. Therefore the mentioned methods' accuracy does not necessarily have to be less than ± 0.1 , but rather as close as possible.

The different methods that was tested are:

- Sub Pixel Estimation using Linear Approximation (Section 6.1.2)
- Sub Pixel Estimation Using Center of Mass (Section 6.1.4)
- Sub-Pixel Estimation Using Local Extrema (Section 6.1.5)

The different methods were compared by their ability to determine the location of an edge on the image, while the edge might be in different locations or in different rotations. Therefore a simple image was generated with a circular edge across it, as shown in Figure 6.6a, with the edge translated and rotated. The effect of noise will also be discussed. The input data will be a created by the simulation software discussed in Section 3.6.

6.2.1 Accuracy

First, the different algorithms was used to estimate the location of the edge y during an ideal elevation (or rotation) only sweep with no noise. An example of the simulated edge in space conditions is shown in Figure 6.6a, where the pixels between the highlighted blue lines are searched for the red horizon line. Figure 6.6b shows the highlighted pixels' intensities over the y -axis, as well as the location of the horizon y .

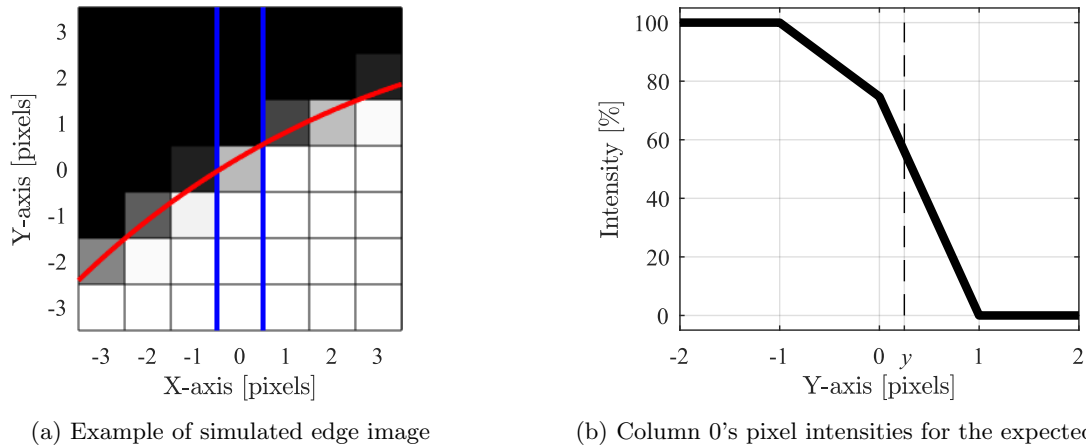
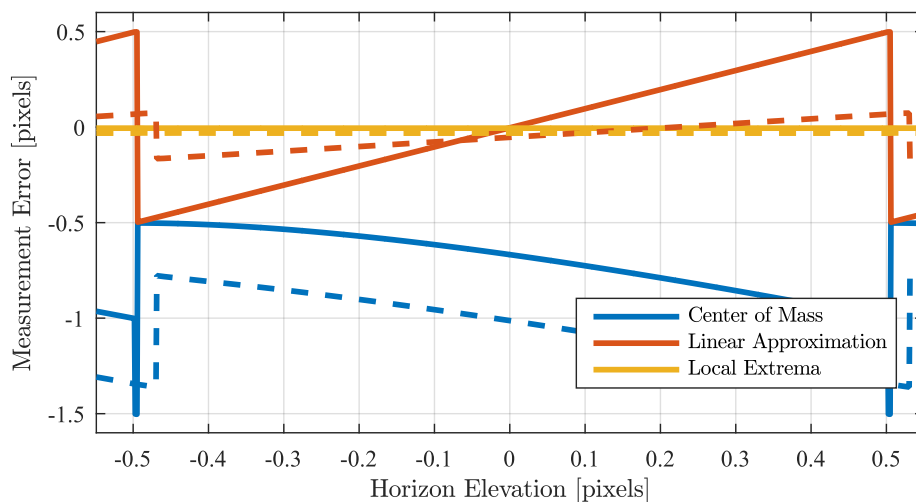
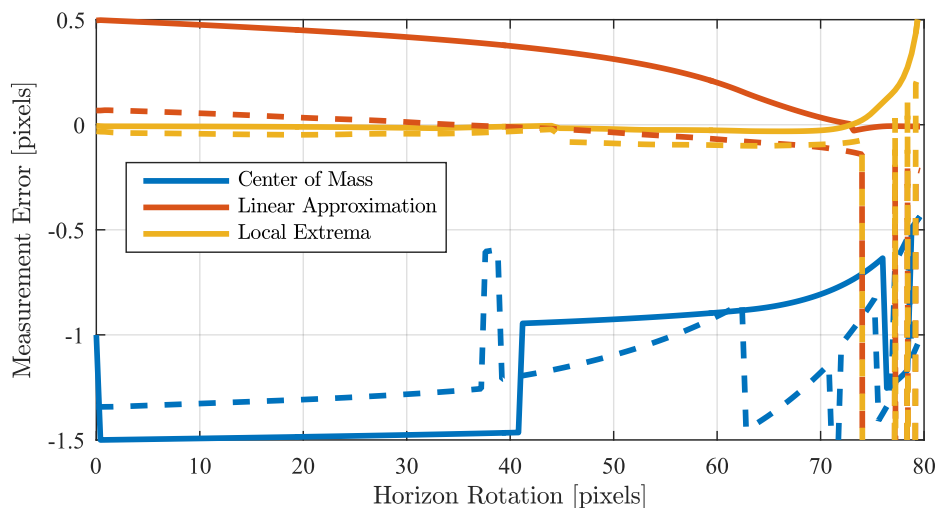


Figure 6.6: Example of simulated edge for space conditions with an vertical translation of 0.25 pixels and rotation of 30°

The horizon line might cross the pixel in any orientation, and therefore all possible situations was tested. First a zero rotation was tested by translating the horizon line vertically over pixel (0,0) from -0.5 to 0.5 pixels. Each algorithm was then applied to find y , with the results plotted in Figure 6.7a. An investigation was also done on the change in measurement error if the horizon line is rotated, which is shown in Figure 6.7b. A constant elevation of 0.5 pixels was used for this test, as the rotated measurement error is influenced by the horizon line vertical translation as well.



(a) Measurement error over a translation sweep with zero rotation



(b) Rotation sweep with an elevation of 0.5 pixels

Figure 6.7: Comparing measurement errors of different feature extraction techniques (solid line shows space conditions and dashed shows ground test conditions)

The Center of Mass method (Section 6.1.4) results in a large measurement error as expected, because the center of mass is offset from the slope center. Correction of this offset during runtime will be complex as the offset changes as the horizon moves over a pixel. However, the main problem arises when rotating the horizon, as the measurement becomes erratic as the slope width changes with the horizon rotation. This is because only a discrete number of pixels can be used to determine the CoM.

As expected the Linear Approximation method (Section 6.1.2) creates an error because of linearly approximating a sigmoid shaped line, as shown in Figure 6.7a. This is very complex to correct during runtime as the offset is dependant on the horizon's rotation and the location on the individual pixels.

The Local Extrema method (Section 6.1.5) outperforms both the other methods in both cases. It shows near perfect behaviour during an elevation sweep, and this behaviour is kept fairly consistent during horizon rotations. The zero error in space conditions is due to the ideal simulation, and the $< 0.01^\circ$ error in ground

test conditions is due to the increased slope width. However, it is found that above a 45° rotation the measurement error increases, and diverges at above 70° (regardless of the technique used). This is mostly due to the non-symmetrical approximation of the Sobel Filter described in Section 8.2.4. This error is very small and relatively constant, which therefore only has a small impact on the final accuracy. It is also found that the high contrast edge during space conditions is more accurately measured than the low contrast edge, which is due to the slimmer parabola to be estimated[39].

6.2.2 Robustness Against Noise

The robustness of the different methods under different noise conditions was also investigated. As described in Section 3.3 an SNR of between 30 dB and 70 dB is expected from a single image. The simulation discussed in Section 6.2.1 was used, but noise was added to the system while keeping the horizon location constant (vertical translation and rotation set to zero). Different horizon rotations were investigated as the inflected noise might produce a larger error under different environments. Additionally only the 3σ measurement error is plotted, as the mean error is discussed in Section 6.2.1 and assumed to be corrected. The results of these tests are shown in Figures 6.8a to 6.8c.

In these figures, the CoM technique's results are not plotted for some conditions because it results in erratic non-Gaussian behaviour by oscillating with $> 0.5^\circ$ variations. In other conditions the performance still varies wildly. The Local Extrema Method and Linear Approximation both deliver good robustness against noise, but it is clear that the Local Extrema method is superior at the majority of expected environments. These results also enforce the observation made in Section 6.2.1 that the accuracy decreases as the horizon rotation (or angle of incidence) increases.

6.2.3 Software Implementation

The mentioned methods have to be implemented on a 8-bit processor which limits the potential complexity of the feature detection method. The two main limiting factors in the software implementation is the memory required to store all the variables (RAM) and the number of instruction cycles required to execute the algorithms.

The CoM and Linear Approximation methods are both fairly simple to execute, only requiring a few additions and singular divisions. Implementing these methods requires the pixel in the middle of the slope, which can be acquired by running a mask through each column, but can be proved to be unreliable under varying conditions. Alternatively, the middle pixel can be found by calculating the gradient image and finding the peak in the column (used in simulations), but this would require significant memory and processing time. The Local Extrema method also requires the gradient image, but is fairly simple to implement after the gradient image is calculated.

Therefore the Local Extrema method would be the most complex method to use, while the two alternative methods would be less complex.

6.2.4 Summary

In Table 6.1 all the mentioned feature extraction techniques are ranked according to different criteria creating a summary of the comparisons in the previous sections.

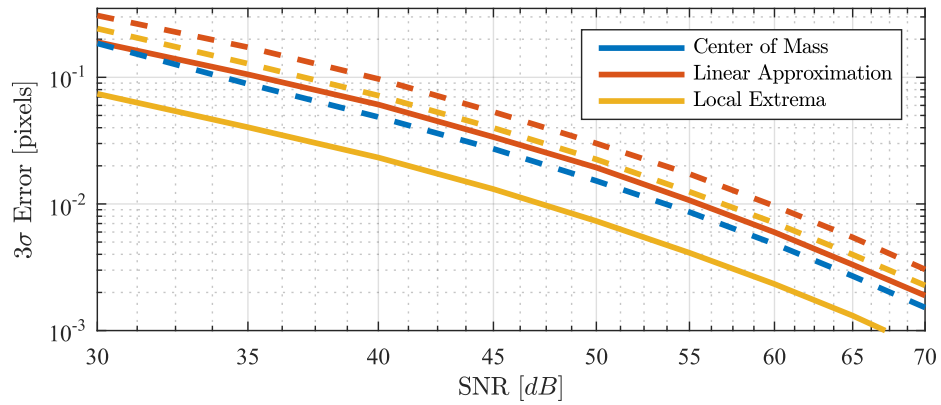
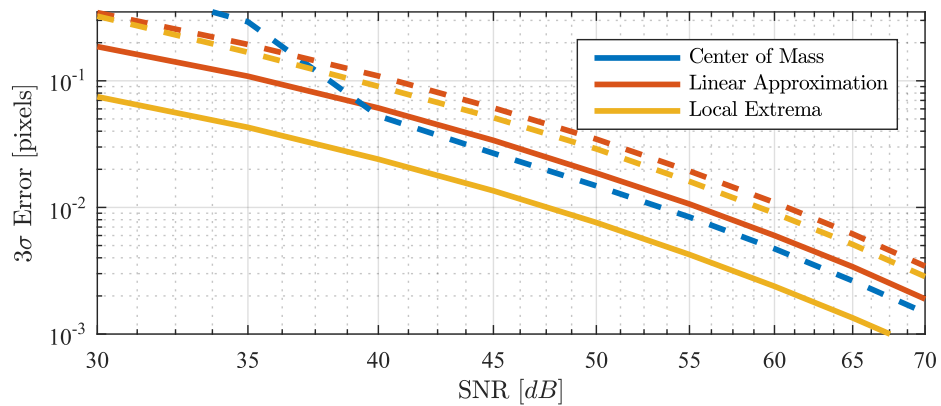
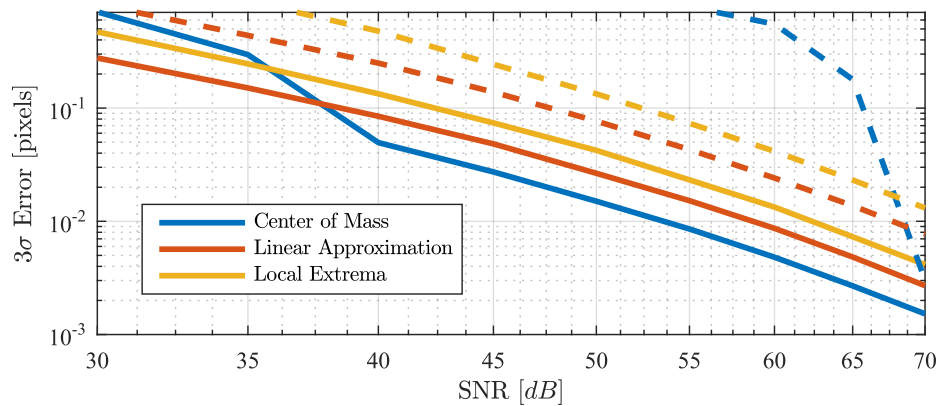
(a) Horizon rotation of 0° (b) Horizon rotation of 35° (c) Horizon rotation of 70°

Figure 6.8: Measurement error statistics due to different noise levels and horizon rotations (solid line shows space conditions and dashed line shows ground test conditions)

Table 6.1: Ranking of Feature Extraction Techniques

	Center of Mass	Linear Approximation	Local Extrema
Accuracy	3	2	1
Robustness to Noise	3	1	2
Robustness to Environment	3	2	1
Implementation Simplicity	2	1	3

(1 and 3 corresponds to the best and worst performances respectively.)

The Local Extrema method was chosen as it is the most accurate, but most importantly it remains robust to noise and a changing environment. The Linear Approximation method might be slightly more robust towards this noise, but this method produces measurement offsets that change with the environment which is complex to correct during runtime. Additionally, the Local Extrema is relatively processing intensive, but it is still viable to implement on an 8-bit processor and a worthy trade-off to ensure robustness.

6.3 Available Scanning Techniques

This section will investigate in which order and direction scanning should be done on an image, as it can influence the processing time and the achievable accuracy.

The Local Extrema technique locates an edge in a row of pixels down to a sub-pixel accuracy, as described in Section 6.1.5. As shown in Figure 6.7b, the accuracy varies as the angle between the edge and edge detection implementation changes - also called the angle of incidence (β in Figure 6.9). Therefore it is important to keep the angle of incidence constant during horizon detection for predictable results, and ideally as small as possible.

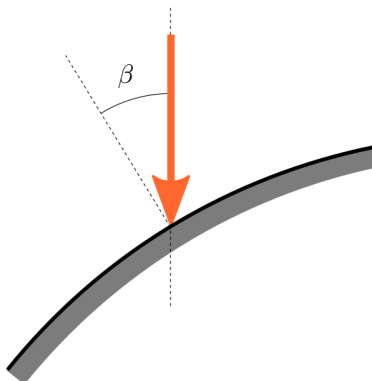


Figure 6.9: Representation of angle of incidence between scanning direction and the horizon

The Local Extrema technique also limits the pixel region in which possible edge coordinates can be found, as the Sobel Filter uses a 3×3 kernel and the Local Extrema itself uses a 3×1 pixel range. Therefore from the original 32×31 pixels, only approximately 28×27 pixels can be used to determine edge coordinates, depending on the scanning method. Additionally, it is essential to determine as many edge coordinates as possible, as each additional point will theoretically result in a more accurate horizon location estimation.

Also discussed in this section is a simple way to determine an approximate horizon rotation before scanning. This is important as it enables the scanning techniques to adapt to different horizon orientations and maximise

its effectiveness. This is described in Section 6.3.1.

6.3.1 Rotation Pre-Estimation using Center of Intensities

A simple way to determine a rough approximate horizon rotation is by determining the center of pixel intensities (CoI) of the image. This is done using conventional center of mass equations, as shown by Eq. 6.5 and 6.6, where n_x and n_y represents the number of pixels in a row and column respectively, and $p(x, y)$ the pixel intensity at coordinate (x, y) .

$$\text{CoI}_x = \frac{\sum_{x=0}^{n_x} \sum_{y=0}^{n_y} p(x, y)x}{\sum_{x=0}^{n_x} \sum_{y=0}^{n_y} p(x, y)} \quad (6.5)$$

$$\text{CoI}_y = \frac{\sum_{x=0}^{n_x} \sum_{y=0}^{n_y} p(x, y)y}{\sum_{x=0}^{n_x} \sum_{y=0}^{n_y} p(x, y)} \quad (6.6)$$

However, this method alone will not create a sufficiently accurate rotation approximation (Figure 6.11). This is because only a section of the Earth disc is visible which results in the discrete integration not being applied uniformly around the image center. Therefore integration is only enabled inside a circular mask, as shown in Figure 6.10, where the dark grey represents the integration range, \times the CoI, and $\hat{\theta}$ the approximated horizon rotation. This ensures that the integration occurs uniformly around the image center.

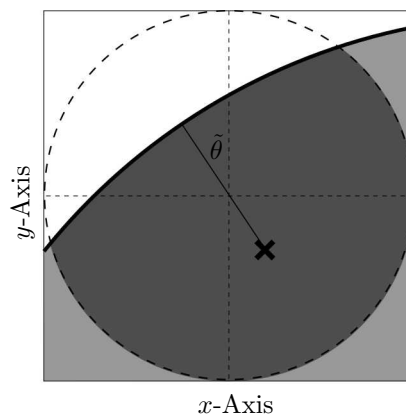


Figure 6.10: Rotation Pre-Estimation using Center of Intensities

This discrete integration can be simplified by assuming that the *warm* Earth pixels and *cold* space pixels will be uniform in their respective intensities. Therefore pixel intensities above a certain threshold can be approximated as 1 while ignoring pixel values below the threshold. This simplifies the denominator of Eq. 6.5 and 6.6 to be equal to the number of *warm* pixels within the mask, as well as removing a large amount of multiplications and additions without severely altering the accuracy. This threshold is empirically determined to be effective at 75% of the maximum *warm* pixel value. A disadvantage of this estimation is that horizons will only be detectable if it is located within the mask, although this will only occur in the rare combination of a large elevation and rotation angle.

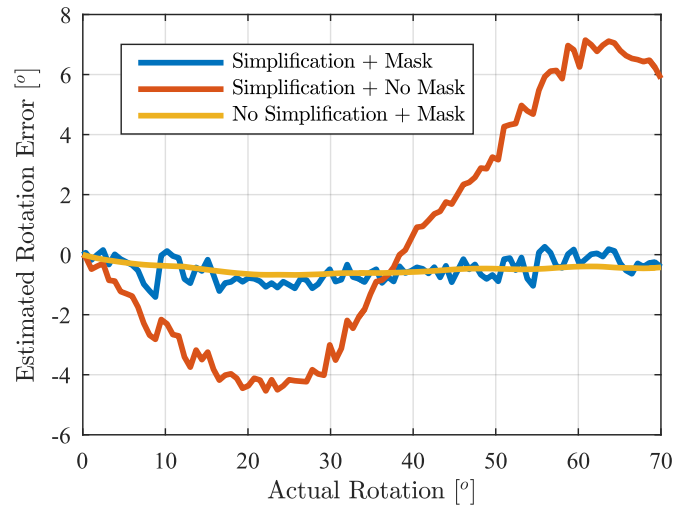


Figure 6.11: Performance of different variants of the horizon pre-estimation using CoI

6.3.2 Simple Column Scanning

The simplest scanning technique would be only scanning column-by-column (vertically). As the sensor's operating range is between $\pm 45^\circ$, the angle of incidence should stay relatively small as the horizon roughly resembles a straight line. However, this will change as the rotation nears 45° . An example is shown in Figure 6.12, where the right arrow has a sufficient angle of incidence of 33° , but the left arrow only has an angle of 57° which might result in inaccurate measurements.

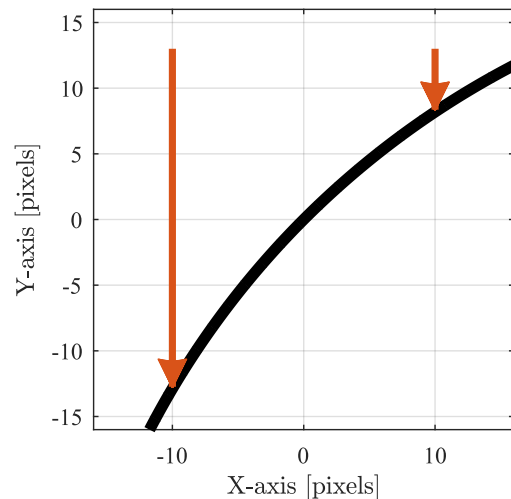


Figure 6.12: Example of simple column scanning's direction on an Earth disc with a rotation of 45°

This scanning method also ensures that the Local Extrema technique's required 3×1 pixel range is always vertical. This adds two more columns to be scanned which increases the maximum amount of potential edge coordinates to 30.

The advantage of this method is that it is straightforward to implement, and therefore very robust in different environments. However, a possible disadvantage is that some edge coordinates can be calculated with a high angle of incidence, which could introduce a measurement error (as described in Section 6.2.2).

6.3.3 Total Scanning

The Simple Column scanning technique (Section 6.3.2) can be expanded by scanning row-by-row (horizontal), additional to the column-by-column scanning, resulting in two sets of edge coordinates. This ensures that the edge will be scanned everywhere with an accurate low angle of incidence, while still preserving the simple and robust implementation, as well as maximising the determined edge coordinates. The downside of this technique is that additional to the accurate low angle of incidence, an inaccurate high angle of incidence measurement will be made. This is depicted in Figure 6.13 where the right hand scan results in an accurate measurement (red) with the $\beta = 33^\circ$, additional to an inevitable inaccurate measurement (blue) with $\beta = 58^\circ$. These inaccurate measurements can potentially offset the final fitted horizon location.

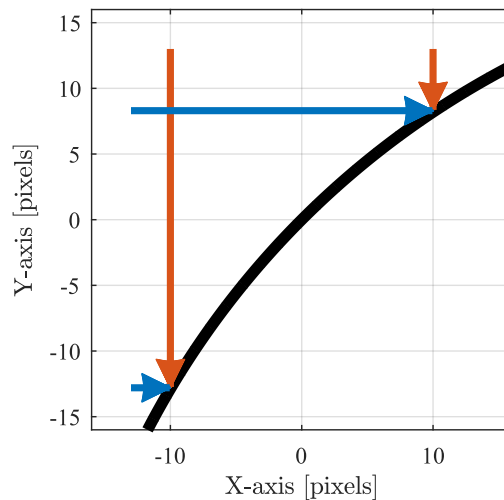


Figure 6.13: Example of the total scanning techniques's direction on an Earth disc with a rotation of 45°

This effect is diminished by only enabling the row-by-row scanning if the expected rotation (determined by the CoI discussed in Section 6.3.1) is large enough. This ensures that the row-by-row scanning is only active when a segment of the horizon line is rotated more than 45° . It is empirically determined that the endpoints of the horizon, such as in Figure 6.13, differ by up to 15° from the true horizon rotation. Therefore the row-by-row scanning is only activated when the expected rotation $\tilde{\theta}$ is larger than $45^\circ - 15^\circ = 30^\circ$.

6.3.4 Predictive Scanning

The shape of the Earth disc will always be circular, as discussed in Section 3.1, and a rough approximation of the Earth disc's location can be determined using the CoI (discussed in Section 6.3.1). Using this information, it is possible to predict where to scan horizontally or vertically roughly. This is done by creating a straight boundary line going through the center image and the CoI approximation of the Earth disc. Combining a horizontal and vertical scan, and only scanning until the boundary line, will then eliminate high angles of incidence while preserving incident angle measurements. This is shown in Figure 6.14.

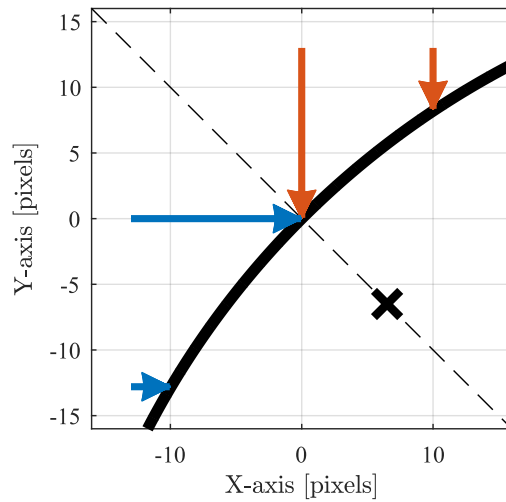


Figure 6.14: Example of the predictive scanning techniques's direction on an Earth disc with a rotation of 45° . The \times represents the CoI

This method can be simplified further by always creating the boundary line at a 45° angle, and the sign dependant on the sign of the CoM x -value. This simplifies the software implementation significantly, but this simplification will not handle zero horizon rotations well. Therefore the predictive Scanning is then set only to activate when the horizon rotation $\tilde{\theta}$ is expected to be larger than $45 - 15 = 30^\circ$ (similar to Total Scanning), and during smaller rotation angles the Simple Column scanning is implemented.

The disadvantage of this method is that it will limit the number of coordinates determined. Additionally, under certain conditions such as $\tilde{\theta} \approx 30^\circ$, high incidence angles might still occur. It will, however, ensure that most of the coordinates will be determined with small incidence angles, and in turn increase the robustness towards noise and not significantly influence results.

6.3.5 Dynamic Scanning

Part of the Local Extrema technique includes the use of a Sobel Filter which calculates the pixel gradients in both the x - and y -directions (called \dot{x} and \dot{y}). The magnitude of this vector is then calculated to create the Sobel Filtered image. If \dot{x} and \dot{y} is known during the edge scanning process, the scanning direction can be dynamically determined for each pixel individually by using the direction with the largest gradient. This would ensure that the angle of incidence is always smaller than 45° .

The downside of this method is the software complexity. It doubles the required programming memory for the Sobel Filter, as two stored variables are required for the direction vector. The more complex software also has a higher risk of failure and increased development time. Therefore this method will not be investigated further.

6.4 Comparing Scanning Techniques

In this section the various scanning techniques described in Section 6.3 are evaluated and compared. The evaluation parameters were as follows:

- Accuracy consistent over entire elevation/rotation range
- Maximizing edge coordinate quantity
- Elevation and rotation angle operating range
- Implementation Simplicity
- Robustness

These techniques were evaluated by implementing them, in conjunction with the Local Extrema edge detection method (Section 6.1.5), to detect edge coordinates on a simulated horizon. However, to minimize the detection of false edges, the edge is only seen as valid when above a certain contrast level. This is done by comparing the scanned pixel's gradient (\dot{p}_0) to the maximum measured gradient (\dot{p}_{max}). It was empirically determined that a threshold of $> 0.65\dot{p}_{max}$ provided sufficient contrast edges.

The different techniques compared are:

- Simple Scanning (Section 6.3.2)
- Total Scanning (Section 6.3.3)
- Predictive Scanning (Section 6.3.4)

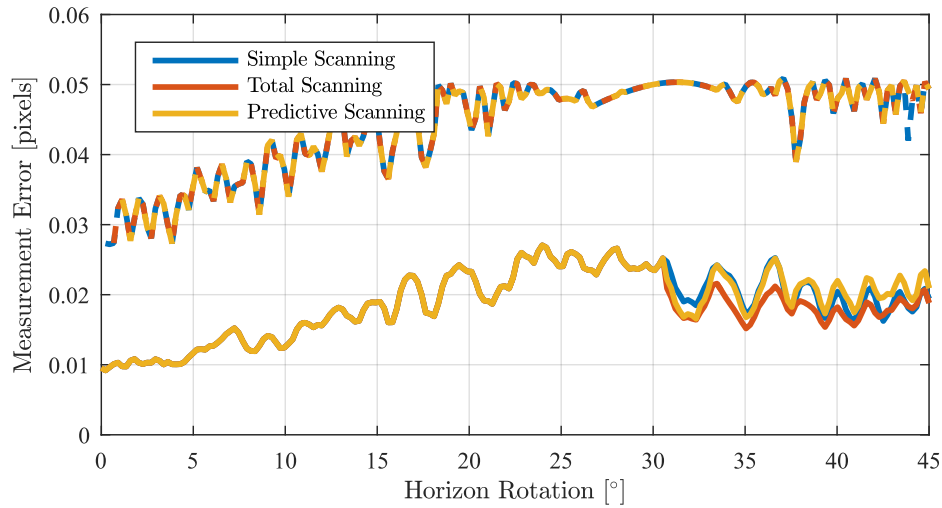
The MATLAB simulations used is discussed in Section 3.6.

6.4.1 Accuracy

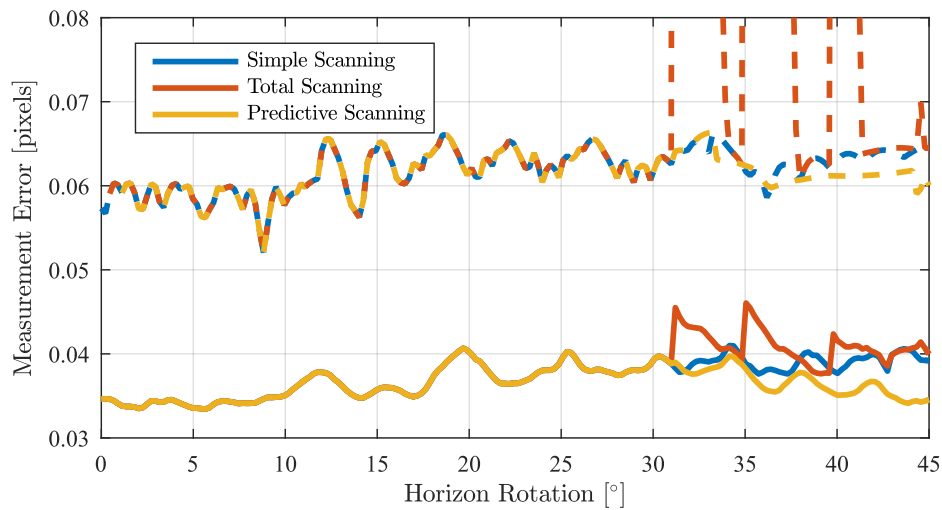
The accuracy of the scanning method varies significantly on the resultant angle of incidence, and will, therefore, vary with the location of the horizon on the image plane. However, this is mainly influenced by the horizon's rotation as it has the most significant impact on the angle of incidence while the elevation angle mostly influences the number of edge coordinates. Additionally, an infinite SNR was assumed, as any noise will cause the edge detection error (Section 6.2.2) to overshadow any possible improvements in different scanning techniques.

To quantify and compare the different scanning methods' accuracy, they were implemented over a variety of horizon rotations as it directly influences the angle of incidence. The erroneous distance between each detected edge coordinate and the true horizon was then compared by determining the mean absolute error as well as the maximum error. The horizon was only rotated between 0° and 45° , as the results should be mirrored between -45° and 0° , all while keeping the elevation angle zero. This is shown in Figure 6.15a.

The discussed techniques will be implemented on the ground tests as well (setup discussed in Section 3.5), which does not perfectly fit the expected Earth disc as discussed in Section 3.1. The ground test's disc radius is roughly 26.6 pixels instead of 65.3 pixels. The smaller disc will create larger incident angles, and therefore a smaller Earth disc is also investigated, as shown in Figure 6.15b.



(a) Expected conditions in outer space



(b) Expected conditions in ground tests

Figure 6.15: Mean (solid) and maximum (dashed) scanning error observed using different scanning techniques

Figure 6.15a shows that all the proposed techniques have a similar performance in the expected space environment. The attempt to lower the angle of incidence through horizontal and predictive scanning is redundant, as the effects are negligible. It is clear where the horizontal scanning is activated at 30° (described in Section 6.3.3 and 6.3.4), but there is no significant improvement from a simple vertical scan (Section 6.3.2).

The advantages of the Predictive Scanning is however prominent in Figure 6.15b at larger rotation angles, as its performance stays constant over the entire rotation range. As expected, the Total Scanning technique performs poorly because it still calculates coordinates using high incident angles. This effect is also visible with the Simple Scanning technique. The decrease in performance in Figure 6.15b is due to the smaller Earth disc, which creates higher incidence angles during scanning.

The improved performance in space conditions over ground test conditions is due to the nature of the edge detection technique, as shown in Figure 6.7, additional to the larger Earth disc.

6.4.2 Edge Coordinate Quantity and Range

The number of acquired edge coordinates should, in theory, increase the accuracy of the measured horizon location, and thus should be maximised. However, this quantity of coordinates is not only dependant on the horizon's rotation but is also dependant on the elevation angle. Therefore, the coordinate quantity acquired was determined for both horizon elevation and rotation angles. The ground test setup's smaller disc size was also investigated, as the disc size plays a significant role in the number of coordinates acquired. The results are shown in Figure 6.16 with the expected space conditions placed on the left and the test setup conditions placed on the right.

These tests were run in the expected operating range with rotation angles between 0° and 45° . This is because of the negative rotation behaviour mirroring the positive rotation behaviour. The elevation angle was tested until $\pm 40^\circ$ which is slightly larger than the camera's FOV of approximately 60° . Therefore, the maximum measurable elevation angle could also be determined, and in turn the available operating range of the sensor.

The results of these tests are shown in Figure 6.16. In these images it is clear that the Total and Predictive Scanning is only activated after a 30° rotation, as designed. Additionally, the number of coordinates is not symmetrical around the $\phi = 0^\circ$ -axis, which is expected as the imaged horizon is not vertically symmetrical. At elevation angles larger than 30° and smaller than -30° , the coordinate quantities rapidly decrease as expected due to the Earth disc moving outside the camera's FOV.

Figures 6.16a/b show that the Simple Scanning method will always result in at least five coordinates under the expected conditions and during ground tests. The coordinate quantity rapidly decays for high elevation or rotation angles, which shows that Simple Scanning results in the least amount of coordinates. This small amount of acquired coordinates is undesirable as it can decrease the accuracy.

In Figures 6.16c/d it shows that Total Scanning results in the largest quantity of coordinates acquired. It is also clear that this technique mimics the Simple scanning technique until 30° rotation at which row-by-row scanning is activated, as designed.

Figures 6.16e/f shows that the Predictive Scanning mimics the Simple scanning as well, but unlike Simple Scanning, the quantity does not decay after 30° . Therefore it detects more coordinates than the Simple scanning, but less than the Total scanning.

6.4.3 Implementation Simplicity and Robustness

The Simple scanning technique is the simplest to implement in software as the scanning process is always constant. The Total scanning technique is still fairly simple to implement, but an initial rotation estimate is required. The Predictive Scanning technique is the most complex due to an initial rotation estimate, as well as only scanning a certain section of the image. Although some techniques are more complex than others, all are relatively simplistic to implement and would not take significant processing time or resources.

The chosen method should also be robust against changing environments and possible extreme angular conditions. It is clear from Figure 6.15 that the Predictive scanning's performance stays constant with most rotation angles. Additionally edge coordinate quantity is sufficient and consistent with different elevation and rotation angles, although it has less than the Total Scanning technique. Total Scanning is not robust as the measurement errors increase significantly after 30° which will severely influence results. Simple Scanning's error stays fairly constant while the edge coordinate quantity decays. Therefore, Predictive Scanning was chosen as the most robust.

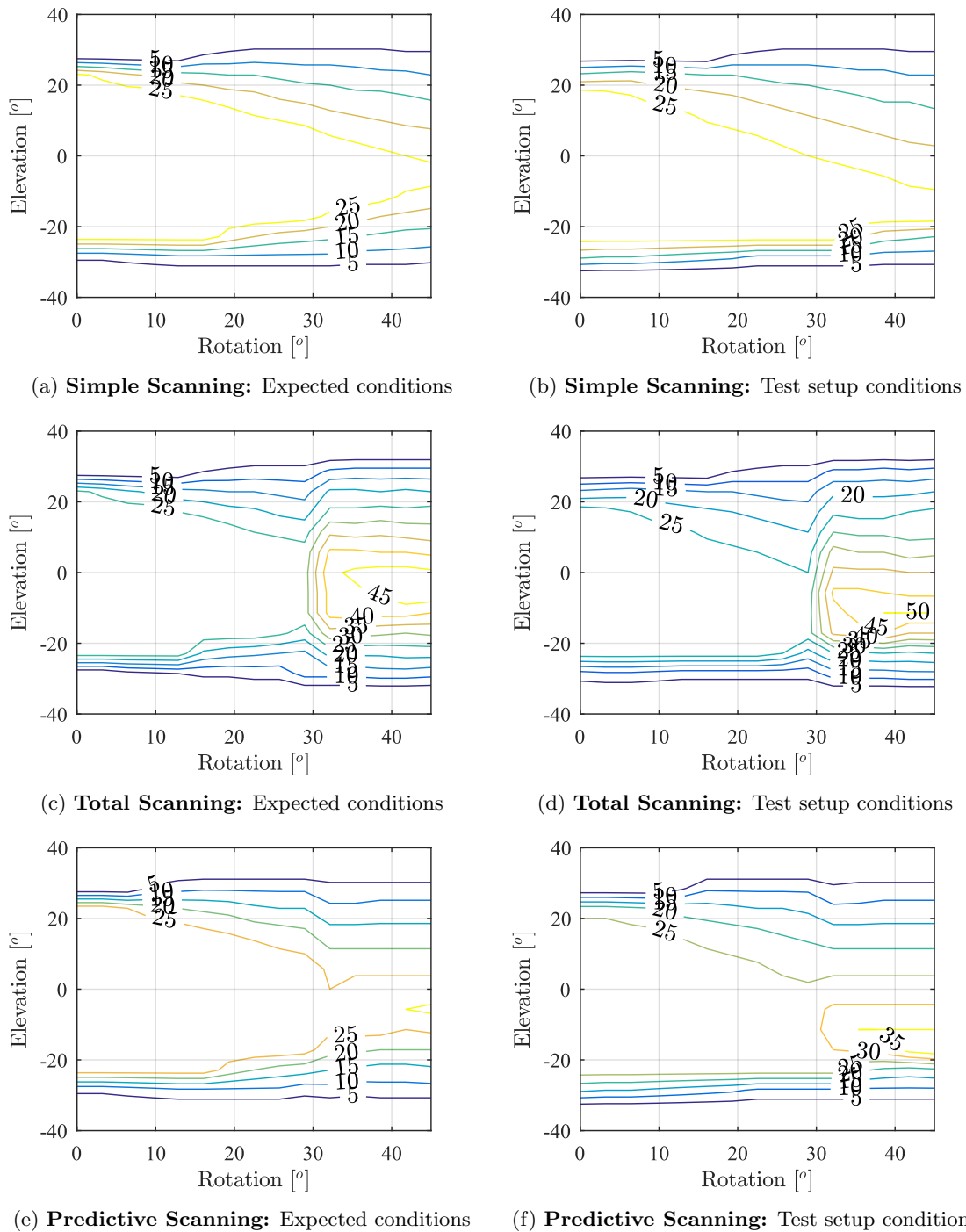


Figure 6.16: Contour plots representing the edge coordinate quantity as determined with various rotation and elevation angles using different scanning techniques

6.4.4 Summary

In Table 6.2 the different scanning techniques were ranked according to the criteria discussed and the results acquired in this section.

Table 6.2: Ranking of Scanning Techniques

	Simple Scanning	Total Scanning	Predictive Scanning
Accuracy	2	3	1
Edge Coordinate Quantity	3	1	2
Simplicity	1	2	3
Robustness	2	3	1

(1 and 3 corresponds to the best and worst performances respectively.)

Predictive scanning was chosen as it is the most robust. In the expected Earth disc conditions and operating range it performs similar to Simple scanning, but unlike Simple Scanning, this behaviour is consistent with higher rotation angles. Total scanning results in more edge coordinates, but at higher rotation angles its behaviour becomes more erratic and unreliable. Therefore, Predictive Scanning was chosen as the most elegant solution.

6.5 Shape Fitting Techniques

This section will describe how predetermined edge coordinates are used to determine the location of the horizon on an image plane (the determining of these coordinates are discussed in Sections 6.1 - 6.4). The location of the horizon is required to determine the pointing direction of the camera boresight, and in turn, determine the device's attitude. Therefore this location should be accurately estimated.

The horizon's location is estimated by fitting a shape to the edge coordinates and then determining the location of the shape. This is more accurate than merely using the discrete edge coordinates, as the shape fitting will filter out slight variations in edge coordinate locations induced by noise and image artefacts. Therefore, the shape chosen to fit the edge coordinates was an important design choice and thus investigated thoroughly.

The horizon shape's location was used to determine the elevation and rotation angle as described by Figure 6.17.

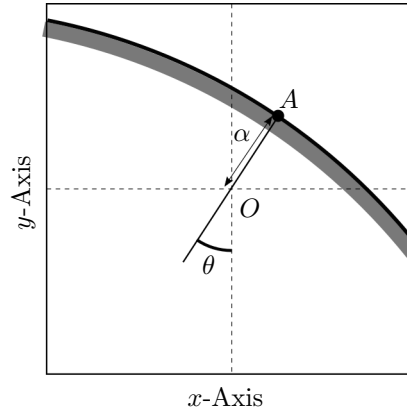


Figure 6.17: Determining camera boresight attitude relative to the horizon on the image plane

In Figure 6.17 the variable A is defined as the horizon's closest point to the origin O . The horizon's rotation is referenced by θ . The horizon elevation angle ϕ is calculated by converting α to a distance on the image sensor, and then applying Eq. 6.7 (which is a manipulation of Eq. 3.4).

$$\phi = -\tan^{-1} \frac{\alpha}{f} \quad (6.7)$$

For perseverance of generality, during this investigation it was assumed that n edge coordinates were available with x -values of $x_i = x_0, x_1, x_2 \dots x_n$ and y -values of $y_i = y_0, y_1, y_2 \dots y_n$.

6.5.1 Straight Line Fit

The most straightforward shape to fit to the horizon is a straight line approximation as the expected horizon closely resembles a straight line (reference Figure 3.3). However, in reality, the horizon is not perfectly straight, which causes an imperfect fit. This means that the fitted line's rotation can be lopsided if the edge coordinates are not equally distributed around the point A . This effect is minimised by applying a circular mask and only using edge coordinates inside the mask, as shown in Figure 3.3.

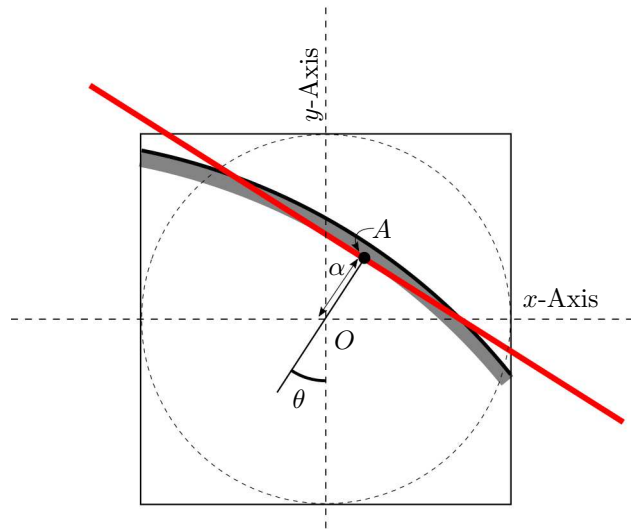


Figure 6.18: Straight Line Fit to Horizon

However, edge coordinates are still discrete and might cause erratic behaviour if the furthest masked edge coordinates are unequal distances from line OA . These coordinates can be refined further by determining each masked coordinate's distance from line OA , as depicted in Figure 6.19. The difference in distance (ϵ) between the two outer most masked coordinates (B and C) are then compared to determine if C 's distance more closely resembles coordinate D 's distance. In the example shown, coordinate B is dismissed and therefore not used in the Least Squares fitting. It is empirically determined to dismiss coordinate B or C when $\epsilon > 0.7$, which significantly diminished the straight line fitting's erratic behaviour. This refinement process is complex to implement but is only to serve as proof of concept and can be simplified further if required.

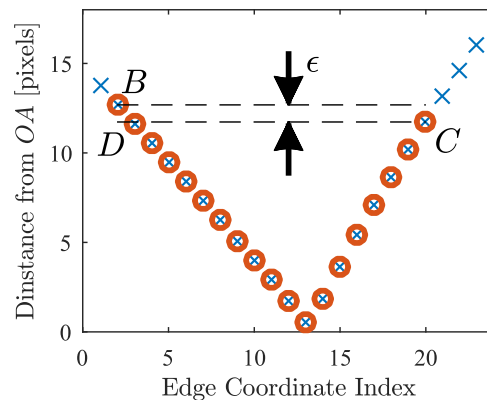


Figure 6.19: Example of straight line fitting refinement using coordinate distances to line OA . Blue crosses represents all determined edge coordinates, and red circles the masked coordinates

Once the coordinates are masked and refined a straight line is approximated. This line is described by the formula $y = mx + c$, and is approximated by utilizing simple linear regression[40]. This is achieved by first determining the gradient m by using Eq. 6.8.

$$m = \frac{\sum^n (x_i - \bar{x})(y_i - \bar{y})}{\sum^n (x_i - \bar{x})^2} \quad (6.8)$$

where \bar{x} and \bar{y} represents the mean x and y values of the coordinates respectively, which in turn is used to calculate c by using Eq. 6.9

$$c = \bar{y} - m\bar{x} \quad (6.9)$$

The variables m and c are then translated to the required θ and α using Eq 6.10 and 6.11 respectively.

$$\theta = -\tan^{-1} m \quad (6.10)$$

$$\alpha = c \cdot \cos \theta \quad (6.11)$$

The advantage of this method is its simplicity. The Least Squares method itself does not require any complex operations and consists mostly of adding, subtracting and multiplying. As mentioned, the refinement process needs simplifying to be implemented on a processor. Another significant disadvantage is that a straight line will not fit the slightly curved horizon perfectly. The calculated α , and in turn the elevation ϕ , will always vary with from the actual elevation as the exact position of the straight line relative to the true horizon is biased. However, this bias error can be roughly corrected during runtime.

6.5.2 Polynomial Fit

A polynomial fit to the horizon will be able to follow the curved nature of the horizon. The polynomial's order (k) should be at least $k = 2$ as the horizon is circular. Additionally the order k should be even, because the horizon's two end points will always point in the same direction along the y -axis. However, a order k too large can potentially fit to the noise present on the edge, which can cause erratic results. It was empirically determined that $k = 4$ follows the Earth disc the most accurately without causing erratic results due to noise.

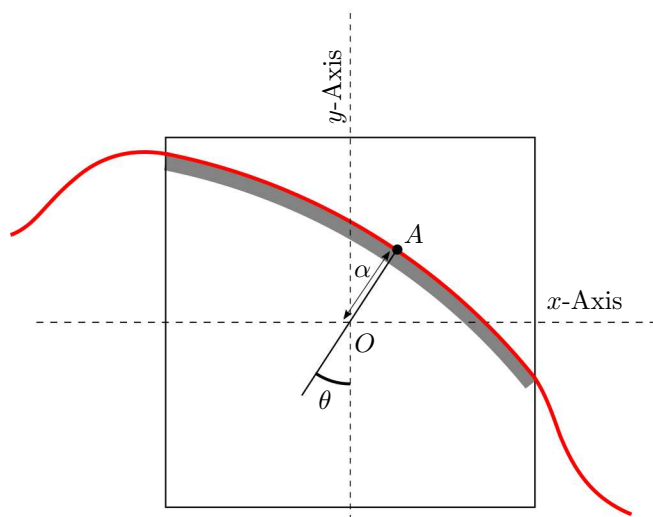


Figure 6.20: Polynomial Line Fit to Horizon

The polynomial's coefficients are calculated by determining the general polynomial model, which is determined by using the Least Squares method. The coefficients of a k^{th} order model is determined by solving a set of linear equations ($\mathbf{M}\mathbf{a} = \mathbf{b}$) described in Eq. 6.12.

$$\begin{bmatrix} n & \sum^n x_i & \cdots & \sum^n x_i^k \\ \sum^n x_i & \sum^n x_i^2 & \cdots & \sum^n x_i^{k+1} \\ \vdots & \vdots & \vdots & \vdots \\ \sum^n x_i^k & \sum^n x_i^{k+1} & \cdots & \sum^n x_i^{2k} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_k \end{bmatrix} = \begin{bmatrix} \sum^n y_i \\ \sum^n x_i y_i \\ \vdots \\ \sum^n x_i^k y_i \end{bmatrix} \quad (6.12)$$

Calculating the coefficients using the normal method of $\mathbf{a} = (\mathbf{M}^T\mathbf{M})^{-1}(\mathbf{M}^T\mathbf{b})$ is not used, because the inverse of an large matrix is severely processing intensive. Therefore \mathbf{a} is calculated by utilizing Cramer's Rule [41] which is described in Eq. 6.13.

$$a_k = \frac{\det(\mathbf{M}_i)}{\det(\mathbf{M})} \quad (6.13)$$

where \mathbf{M}_i is the matrix \mathbf{M} (described in Eq. 6.12) with the i^{th} column replaced by the column vector \mathbf{b} . This fitting process is complex, and will be more complex with higher k values, but it will result in a more accurate representation of the Earth disc.

The attitude of the fitted polynomial is determined by calculating A , and then calculating α and θ by analysing line OA . Calculating A 's location will be complex with the processing time directly proportional to the accuracy of the calculation. Calculating A can be done in various optimized ways as the distance from the polynomial to O will create a local minima around A , which can be utilized. However, this is not investigated, as only a proof of concept is required.

The disadvantage of this method is that the horizon is required to resemble a function, i.e. have one y -value for every x value. As the horizon rotation nears $\pm 45^\circ$ the horizon end points might not resemble a function by becoming vertical. This effect will be emphasized with smaller Earth discs.

6.5.3 Circular Fit

As the horizon is circular shaped (Section 3.1) a circular model will fit the horizon perfectly. This means that each additional coordinate acquired will improve this model, unlike the straight line fit (Section 6.5.1). Additionally small variations in edge coordinate locations will not severely influence estimated horizon location, like the polynomial fit (Section 6.5.2).

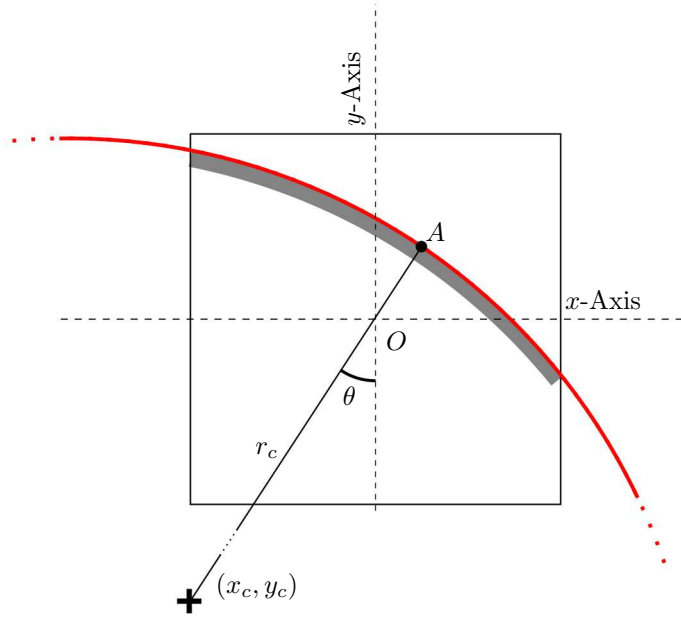


Figure 6.21: Straight Line Fit to Horizon

The circle is fit to the data using a Least Squares method by solving the equation $\mathbf{M}\mathbf{a} = \mathbf{b}$. In order to derive \mathbf{M} and \mathbf{b} a circle's equation is rewritten as:

$$ax + by + c = x^2 + y^2 \quad (6.14)$$

with

$$\begin{aligned} a &= 2x_c \\ b &= 2y_c \\ c &= r_c^2 - \sqrt{x_c^2 + y_c^2} \end{aligned} \quad (6.15)$$

The equation $\mathbf{M}\mathbf{a} = \mathbf{b}$ is rewritten as:

$$\begin{bmatrix} x_0 & y_0 & 1 \\ x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots \\ x_n & y_n & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} x_0^2 + y_0^2 \\ x_1^2 + y_1^2 \\ \vdots \\ x_n^2 + y_n^2 \end{bmatrix} \quad (6.16)$$

The vector \mathbf{a} representing the fitted circle's properties is calculated by Eq. 6.17. An inversion of a symmetrical 3×3 matrix is still applicable for a 8-Bit processor and is therefore sufficient for this implementation.

$$\mathbf{a} = (\mathbf{M}^T \mathbf{M})^{-1} (\mathbf{M}^T \mathbf{b}) \quad (6.17)$$

of which the expressions $\mathbf{M}^T \mathbf{M}$ and $\mathbf{M}^T \mathbf{b}$ is simplified to:

$$\mathbf{M}^T \mathbf{M} = \begin{bmatrix} \sum^n x_i^2 & \sum^n x_i y_i & \sum^n x_i \\ \sum^n x_i y_i & \sum^n y_i^2 & \sum^n y_i \\ \sum^n x_i & \sum^n y_i & n \end{bmatrix} \quad (6.18)$$

$$\mathbf{M}^T \mathbf{b} = \begin{bmatrix} \sum^n x_i (x_i^2 + y_i^2) \\ \sum^n y_i (x_i^2 + y_i^2) \\ \sum^n (x_i^2 + y_i^2) \end{bmatrix} \quad (6.19)$$

Once \mathbf{a} is determined, the circle's properties is calculated using Eq. 6.15. Using the circle parameters x_c, y_c and r_c the device's attitude is then determined by calculating α and θ using Eq. 6.20

$$\begin{aligned} \alpha &= r_c - \sqrt{x_c^2 + y_c^2} \\ \theta &= -\tan^{-1} \frac{x_c}{y_c} \end{aligned} \quad (6.20)$$

This method is relatively complex to implement, but this is diminished by simplifying the matrix computations as discussed in Section 8.2.6. Additionally, each additional coordinate acquired will only improve the measurement. This method also creates in a simple way to determine if the fitted model is accurate as the radius r_c should be equal to the expected Earth disc radius (Section 3.1).

6.6 Comparing Shape Extraction Techniques

In this section, the various shape fitting techniques will be evaluated and compared in their ability to estimate the horizon location from pre-determined edge coordinates. These techniques were evaluated according to the following criteria:

- Uniform behaviour
- Robustness towards noise
- Implementation simplicity

The different horizon models compared are:

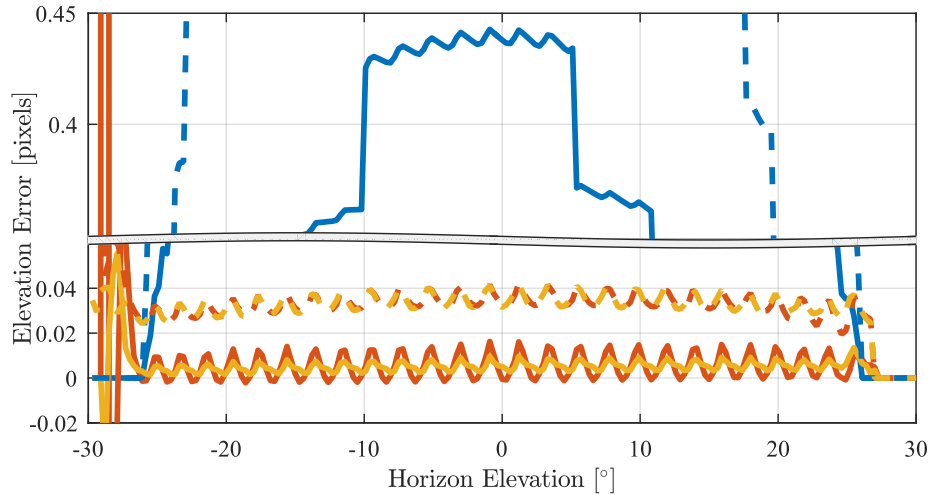
- Straight Line Fit (Section 6.5.1)
- Polynomial Fit Model (Section 6.5.2)
- Circular Fit (Section 6.5.3)

The chosen shape fitting algorithms were implemented during the ground testing and therefore required the ability to fit smaller Earth discs (setup discussed in Section 3.5). Therefore, both conditions are evaluated, i.e. the expected space and ground test conditions. The simulations were done using the MATLAB simulation discussed in Section 3.6.

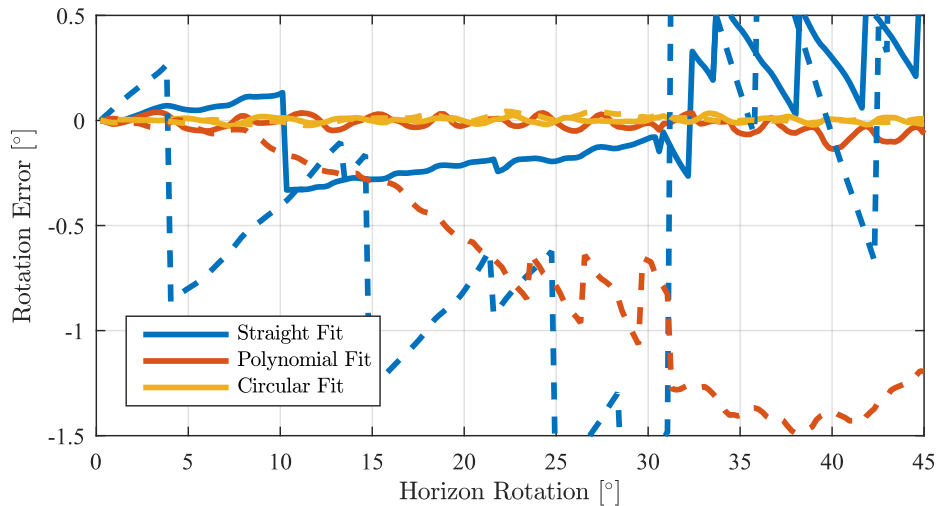
6.6.1 Accuracy

In this section, the general accuracy of the different shape fitting techniques will be investigated, i.e. how accurately the different techniques follows an ideal horizon rotation and elevation. Since only the performance of the different techniques is tested, the SNR is set to infinity, thus eliminating the effect of noise. As mentioned in Section 6.4 the created ground tests (Section 3.5) creates a smaller disc than the expected from

an altitude of 500 km, therefore both disc sizes were investigated. The results of a horizon rotation sweep with zero elevation is shown in Figure 6.22a, and in Figure 6.22b an elevation sweep is shown with a zero rotation. The elevation error and rotation error represents the values α and θ respectively, as seen in Figure 3.3.



(a) Elevation measurement errors over a horizon elevation sweep



(b) Rotation measurement errors over a horizon rotation sweep

Figure 6.22: Comparing measurement errors of different shape fitting techniques (solid line shows space conditions and dashed ground test conditions)

In Figure 6.22b it is clear that the fitting techniques only provide sufficient elevation estimations until roughly $\pm 25^\circ$. This is because the techniques require a sufficient number of coordinates to estimate the horizon shape accurately. However, this range of accurate estimations will be larger in a real application as this simulation does not include lens distortion which will allow edge coordinates farther from the camera boresight to be detected.

The Straight Fit shows erratic behaviour during the rotation sweep as discrete coordinates move in and out of the mask. During the elevation sweep, this technique also induces a severe offset due to non-ideal fit which

is complex to correct during runtime. The Polynomial Fit performs on par with the Circular Fit during an elevation sweep, but shows in increasing error during a rotation sweep, especially with the smaller disc size in the ground test conditions (Figure 6.22a). This is due to the Earth disc section not approximating a function, as described in Section 6.5.2. It is found that higher order values k mitigate this error but results in making the method significantly more susceptible to noise.

Figures 6.22a and 6.22b clearly shows that the Circular Fit outperforms the other two methods in both rotation and elevation sweeps, as expected. This method is able to follow the horizon location to within a margin of 0.007 pixels in space conditions, and 0.035 pixels in the ground tests conditions. The small inevitable error is caused by the edge detection technique, as discussed in Section 6.2.1. This also causes a slight oscillation in the elevation measurement response, which is mainly due to high incident angle measurements. The error these large incident angle measurements create is proportional to the horizon's rotation and location on the pixel.

6.6.2 Robustness Against Noise

The robustness against noise of the different techniques was investigated to ensure that the shape fitting works in the inevitable noisy conditions. Therefore, the various techniques were tested by being utilized to determine a simulated horizon's location under different noise levels, and quantifying the errors made, as shown in Figures ?? and 6.24. The location of the horizon on the image plane also affects the accuracy, as under some conditions fewer coordinates are detected (Section 6.4.2). Therefore, the optimal location was tested ($\theta = \phi = 0$) as well as an arbitrary sub-optimal location ($\theta = \phi = 20^\circ$). It was chosen to use the horizon rotation estimation error to evaluate the different techniques because it is the most susceptible to noise. Additionally only the 3σ measurement error was plotted, as the mean error is discussed in Section 6.6.1 and assumed to be corrected.

An example of the simulated images are shown in Figure 6.23.

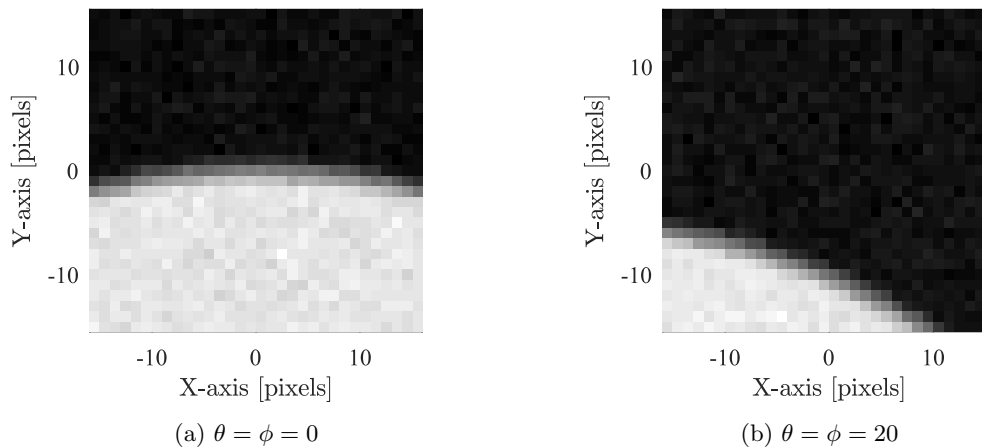


Figure 6.23: Example images used for noise investigation representing the expected ground testing environment with a SNR of 30 dB

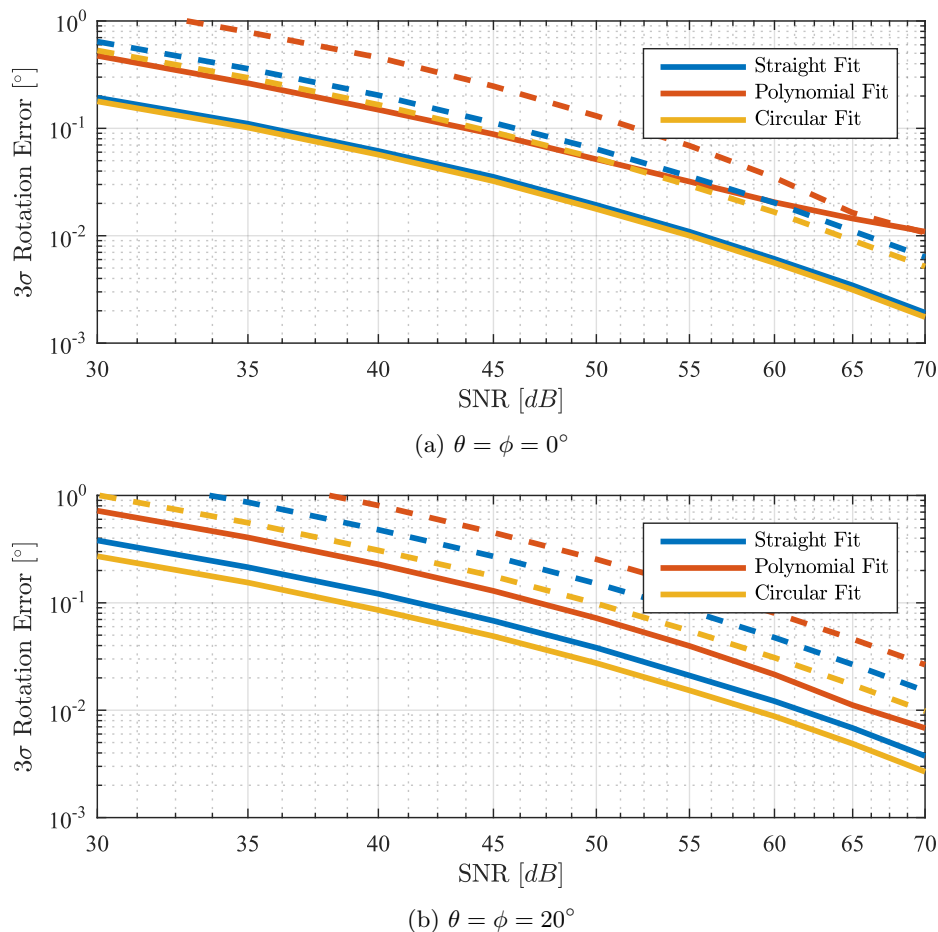


Figure 6.24: Statistics of rotation measurement errors made while estimating a simulated horizon's location under different SNR levels (solid line shows space conditions and dashed ground test conditions)

Although all the techniques' accuracies are sufficient at high SNRs, it is clear that the Circular Fit outperforms both the other methods under all conditions as expected. It delivered high accuracies, as well as being consistent under various conditions.

6.6.3 Implementation Simplicity

The simplest technique to implement is the Circular Fit. The horizon estimation of both other methods might prove simpler, but they require additional complex calculations to remain accurate. The Straight Line Fit involves edge coordinate masking and refinement, which severely increases the computational complexity. The Polynomial Fit requires finding point A on the fitted polynomial closest to the image center, which is also very complex to calculate. The Circular Fit provides an accurate representation directly using the circle's parameters and is simple to determine.

6.6.4 Summary

In Table 6.3 all the mentioned shape fitting techniques were ranked according to different criteria creating a summary of the comparisons in the previous sections.

Table 6.3: Ranking of Shape Fitting Techniques

	Straight Line Fit	Polynomial Fit	Circular Fit
Uniform Behaviour	3	2	1
Robustness to Noise	2	3	1
Implementation Simplicity	2	3	1

(1 and 3 corresponds to the best and worst performances respectively.)

From this table, it is clear that the Circular Fit is the best choice. It has a uniform behaviour over a large attitude range, has the largest range in which accurate estimations are made, and is the most robust towards noise. Additionally. It is the simplest to implement. Therefore, the Circular Fit technique was chosen.

6.7 Discussion

Various strategies were considered and evaluated to determine the horizon's location accurately, and the most robust and elegant solution was chosen to implement and use for the rest of this study. The chosen strategy consists of determining the edge coordinates using the Local Extrema method (Section 6.1.5) while using the Predictive Scanning pattern (Section 6.3.4). To model the location of the Earth, the Circular Fit (Section 6.5.3) is then fitted to the determined edge coordinates. This resulted in an accurate, robust, simple and elegant solution, and was used for the rest of this study.

7 Lens Distortion Correction

This section will investigate the inevitable lens distortion present on the camera, and how to correct this distortion effectively and efficiently.

7.1 Overview

Lens distortion (or lens aberrations) is an image distortion caused by the lens design, which includes astigmatism, spherical aberrations, field curvature and barrel/pincushion distortion. These aberrations distort the captured image from the ideal pinhole camera model. To accurately determine feature locations in images, this distortion needs to be modelled and then corrected. The most prominent form of lens distortion is the barrel/pincushion distortion, as shown in Figure 7.1, and therefore only this distortion is focused on.

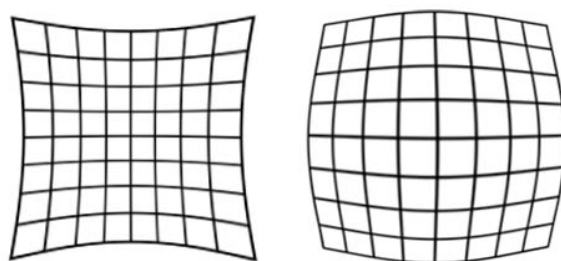


Figure 7.1: Examples of pincushion (left) and barrel (right) distortion

Typical lens distortion correction implementations focus on high-resolution images where individual pixels are shifted to their corrected locations. However, to implement this on low-resolution images, where feature locations are required to sub-pixel accuracy, would be severely processing intensive and complex. Therefore, edge coordinates will be determined on the distorted image (using techniques discussed in Section 6.1 - 6.4). Only these coordinates will be lens distortion corrected which will then be used for the horizon shape fitting (Section 6.5 - 6.6). This ensures that no unnecessary information is distortion corrected.

This calibration process also assumes most of the infrared camera's intrinsic parameters[42] are as provided in it is datasheet [6], which is shown in Eq 7.1. However, the principle point is determined through the distortion correction process discussed in this section.

$$\mathbf{K} = \begin{bmatrix} \alpha_x & \gamma & u_0 \\ 0 & \alpha_y & v_0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 26.364 & 0 & 0.246 \\ 0 & 26.364 & 0.370 \\ 0 & 0 & 0 \end{bmatrix} \quad (7.1)$$

The parameters $\alpha_x = f \cdot m_x$ and $\alpha_y = f \cdot m_y$ represents the focal length in pixels, where m_x and m_y are the pixel size, and f the focal length. The skew coefficient is represented by γ , and the principle point by (u_0, v_0) .

7.2 Modeling Lens Distortion

This section will describe how to model the lens distortion using a set of images captured by the camera. Once this model is determined Section 7.3 will describe the distortion correction methods used.

7.2.1 Overview

In literature, the most common way to model lens distortion is by imaging a checkerboard [43, 44], sometimes in different configurations. These images are then processed and the checkerboard square locations extracted, resulting in a two-dimensional matrix of distorted coordinates. These coordinates are then compared to their expected locations, given the pinhole camera model, from which the lens distortion can be modelled and then corrected as described in Section 7.3. If the exact location of the checkerboard is not known the lens distortion can still be modelled by utilising iterative algorithms [44]. However, this method requires a large number of checkerboard images, and given the long testing durations (Section 3.5.2) this method is not feasible, and therefore not considered.

7.2.2 Methodology

As described in Section 7.2.1 an image of a checkerboard is required to model the lens distortion. However, imaging a checkerboard with a low-resolution infrared camera result in two problems, namely:

- **Creating a thermal checkerboard:** This can be achieved by heating a specialised checkerboard, as black has a higher emissivity than white. This difference in emissivity can be emphasised by coating the black squares in specialised high emissivity paint, and low emissivity paint for the white squares. However, heating such a checkerboard without damaging it could prove problematic, and will not result in a high contrast image due to the small difference in temperatures.
- **Aliasing due to Low Resolution:** The low-resolution camera's Nyquist spatial frequency limits the size and complexity of the imaged checkerboard, additional to limiting the available contrast. This severely inhibits the ability to determine the squares' locations accurately.

Therefore, a more elegant solution is proposed. Low-resolution images result in poor performance when extracting complex features (e.g. checkerboards), but simple feature locations can still be estimated to a high accuracy. A single edge's location, for example, can be accurately determined to within 0.1 pixels using the Local Extrema method (Section 6.1.5). Therefore, instead of imaging an entire checkerboard instantaneously, a checkerboard can be built in software using multiple images to simulate the straight lines present in the checkerboard. An example is shown in Figure 7.2 where a single point on the image plane is located by utilising two images.

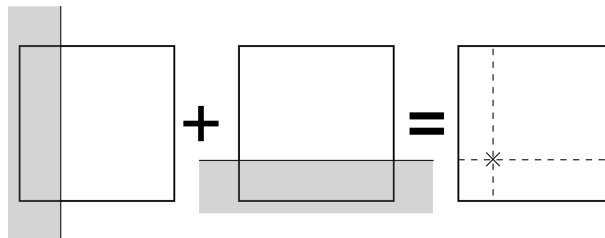


Figure 7.2: Simulating a single square location utilising multiple images

The two lines (right Figure 7.2) are determined by fitting a polynomial through multiple arbitrary edge points

on each of the left two images' edges, similar as described in Section 6.5.2. A polynomial line is fitted, instead of a straight line, because the imaged edge will be curved due to the lens distortion (as seen in Figure 7.1 on the right). Finding the intersection of the two polynomials is described in Section 7.2.3.

The edge points (x_e) are determined using the Local Extrema method, as discussed in Section 6.1.5, which finds the highest gradient on the edge. This edge point does not necessarily correspond to the exact location of the imaged plate, but rather at a slight offset (ϵ) from x_e . The magnitude of this offset has to be empirically determined, which in turn requires complete lens distortion knowledge. To solve this causality dilemma the inevitable edge offset is removed from the equation by imaging an additional set of inverted images, from which an additional set of points are calculated. These inverted images are created by rotating the imaged steel plate 180° from the original location for both configurations shown in Figure 7.2. This will cause the mentioned offset to be in the opposite direction of the first set of images, as shown in Figure 7.3. Combining the original and inverted edge points the precise location of the imaged plate (x_e) can be determined without knowledge about the edge offset's magnitude (ϵ).

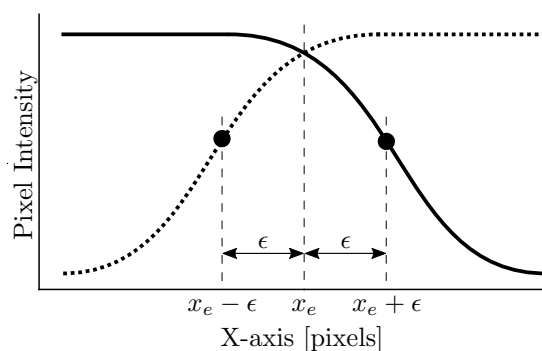


Figure 7.3: Example showing the original edge (solid) inverted edge (dotted), and the measured edge locations (circles)

This means that four images are required to simulate a single point on the checkerboard, or $4N$ images to create a $N \times N$ checkerboard matrix. It was chosen to create a 25×25 checkerboard which is slightly larger than the camera's effective FOV, which is empirically determined to be approximately 70° . The effective FOV is larger than the theoretical FOV due to the lens distortion. The equipment discussed in Section 3.5 was used with the steel plate's straight side facing upwards. To ensure a high SNR for accurate measurements, each image was sampled 15 times and then averaged, which improved the SNR by 11.76 dB (Section 3.3.3).

In order to create the images as seen in Figure 7.2 the rotation angles used was $\theta = (0^\circ, 90^\circ, 180^\circ, 270^\circ)$. The elevation angle ϕ was varied which translated the rectangle across the image to simulate the checkerboard. As the plate is $d = 550 \text{ mm}$ from the camera and assuming the effective 70° FOV, the height of the simulated checkerboard was calculated as 770 mm . This resulted in 24 squares of $(a \times a)$ in size with $a = 32.1 \text{ mm}$. The required elevation angles are, therefore, be determined by:

$$\phi_i = \tan^{-1} \frac{ai}{d}, \quad \text{where} \quad i = (-12, -11, \dots, 11, 12) \quad (7.2)$$

7.2.3 Calculating Polynomial Intersections

In order to find the different points on the checkerboard, the intersection ($x_{\text{int}}, y_{\text{int}}$) between two polynomials is required. It was found that 3rd-order polynomials fit the distorted edges sufficiently. To ensure the vertical

lines remain a function they were transposed. Therefore, the two fitted polynomials for the horizontal (y_H) and vertical (x_V) lines is expressed as Eq. 7.3 and 7.4 respectively.

$$y_H = ax^3 + bx^2 + cx + d \quad (7.3)$$

$$x_V = ey^3 + fy^2 + gy + h \quad (7.4)$$

The intersection ($x_{\text{int}}, y_{\text{int}}$) of these polynomials is determined by substituting y_H into Eq. 7.4, which results in Eq. 7.5, where $x = x_{\text{int}}$.

$$\begin{aligned} x = & e(ax^3 + bx^2 + cx + d)^3 \\ & + f(ax^3 + bx^2 + cx + d)^2 \\ & + g(ax^3 + bx^2 + cx + d) \\ & + h \end{aligned} \quad (7.5)$$

Eq. 7.5 is then expanded, and then simplified to create Eq. 7.6.

$$0 = Ax^9 + Bx^8 + \dots + Ix + J \quad (7.6)$$

Using MATLAB the roots of Eq. 7.6 are found. Due to this specific application it was assumed that only one rational intersection exists, and therefore the only rational root is equal to x_{int} . The value for y_{int} was then calculated by substituting x_{int} into Eq. 7.3.

7.2.4 Results

In Figure 7.4 the four images required to determine one checkerboard point are shown. The top two figures (a and b) are used to determine the first point, and the inverted set of images (c and d) are used to determine the inverted point.

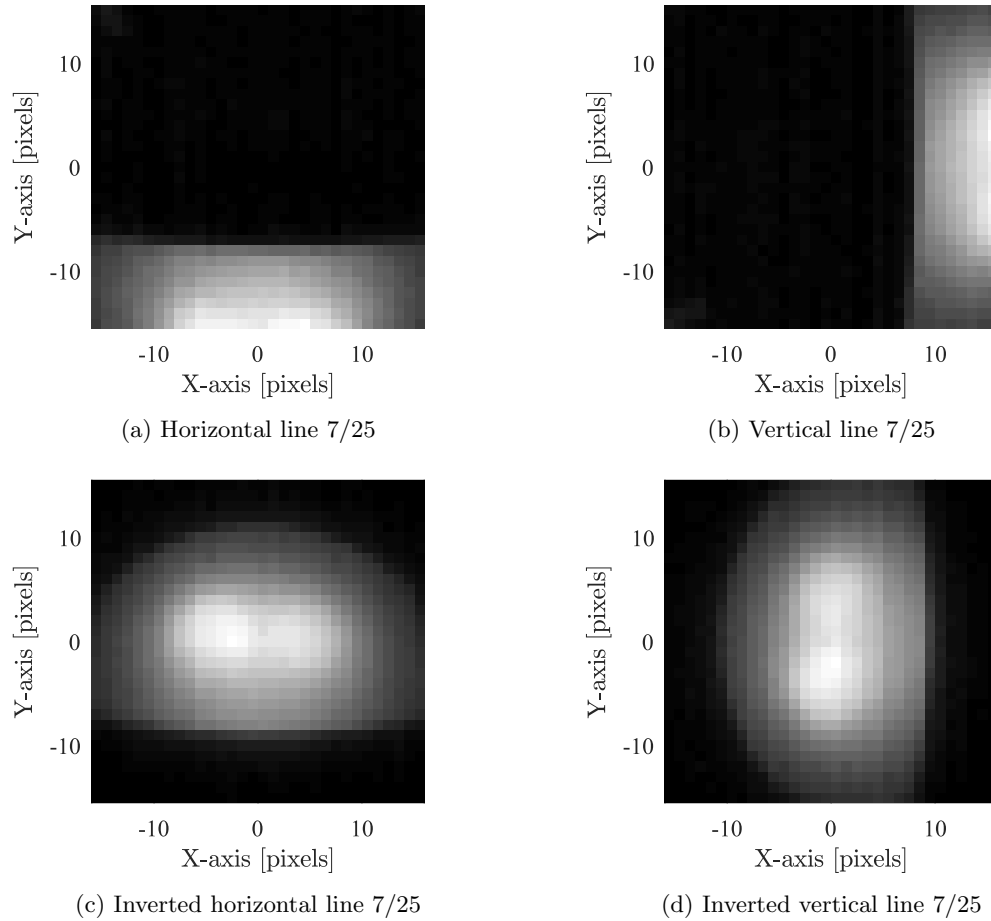


Figure 7.4: Example of four images required to estimate one point on the checkerboard

Figure 7.5 shows the fitted polynomials on the edges shown in the non-inverted images. These polynomials are still affected by the offset ϵ .

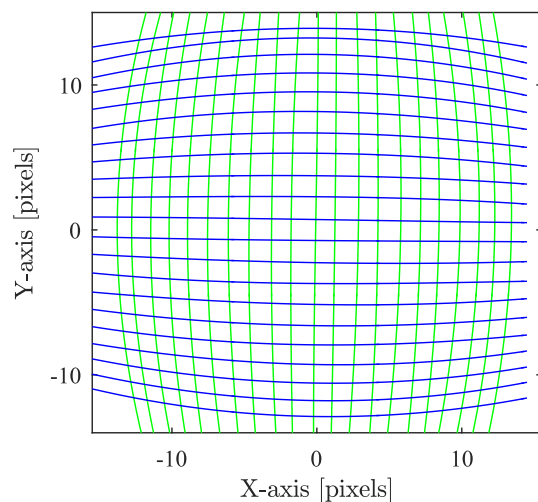


Figure 7.5: Example of the fitted polynomials on the imaged edges

This offset ϵ was then removed from the equation by finding the middle between the non-inverted and inverted sets of images. The middle point (x_e) of the measured coordinates were then calculated. This is shown in Figure 7.6, where the circles represent the measured intersections (blue=inverted), and the red plus the combined point. It is clear that the offset ($\epsilon \approx 0.42$) is present and in the direction of the expected offset.

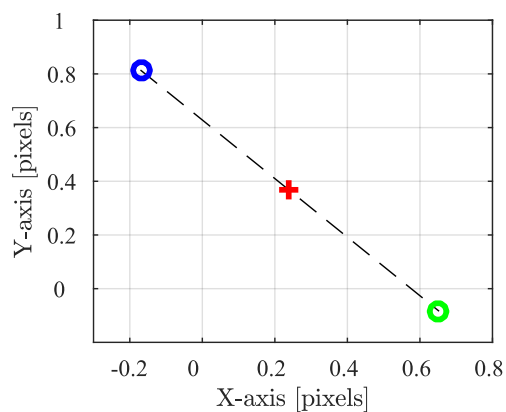


Figure 7.6: Example of combining non-inverted (green) and inverted (blue) points to calculate the checkerboard point location (red)

Figure 7.7 shows the combination of all the combined points which created the simulated checkerboard. The middle of the checkerboard (c_0) was determined as $c_0 = (0.2464, 0.3698)$, which is assumed to be the principle point (u_0, v_0). There was a slight rotation present on the checkerboard (approximately 0.78°) which was caused by mounting errors, as lens aberrations do not cause rotation around the principle point.

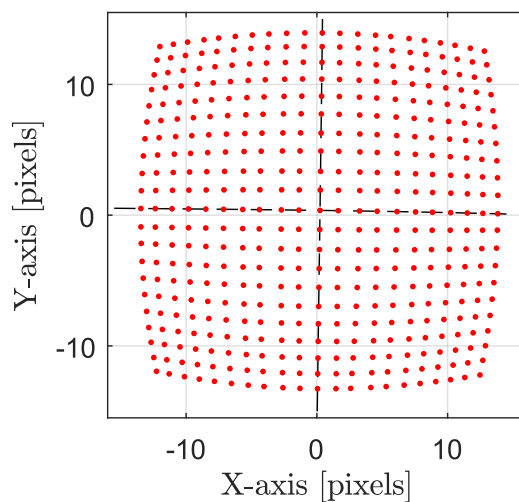


Figure 7.7: Simulated checkerboard with fitted polynomials through middle line

This rotation was corrected using software by rotating all the determined points in Figure 7.7 around the principle point c_0 . This manipulation did not affect the lens distortion model but ensured that future ground tests were aligned accurately. The corrected simulated checkerboard is shown in Figure 7.8.

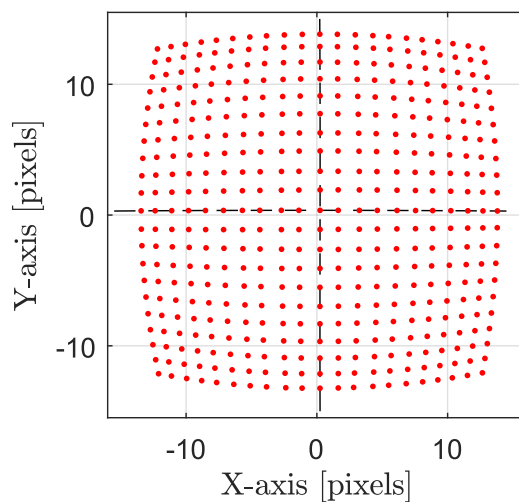


Figure 7.8: Simulated - and corrected - checkerboard with fitted polynomials through middle line

7.3 Lens Distortion Correction

This section will describe how lens distortion is corrected, once the lens distortion model is known and investigated. The approximation of this model is defined in Section 7.2.

7.3.1 Lens Distortion Investigation

The distortion correction is required to sufficiently correct the lens distortion without being too complex to implement on a small processor. Therefore, the present lens distortion was investigated to ensure an efficient and effective distortion model. This was done by comparing the distorted coordinates with their respective expected locations.

In Figure 7.9a the distortion error is plotted relative to the distorted coordinate's distance from the center c_0 . This shows that the distortion has a radial component as the error increases further from the center. However, pure radial distortion creates a thin line as the radius increases, while the Figure 7.9a shows a broad line, which means the distortion is not purely radial.

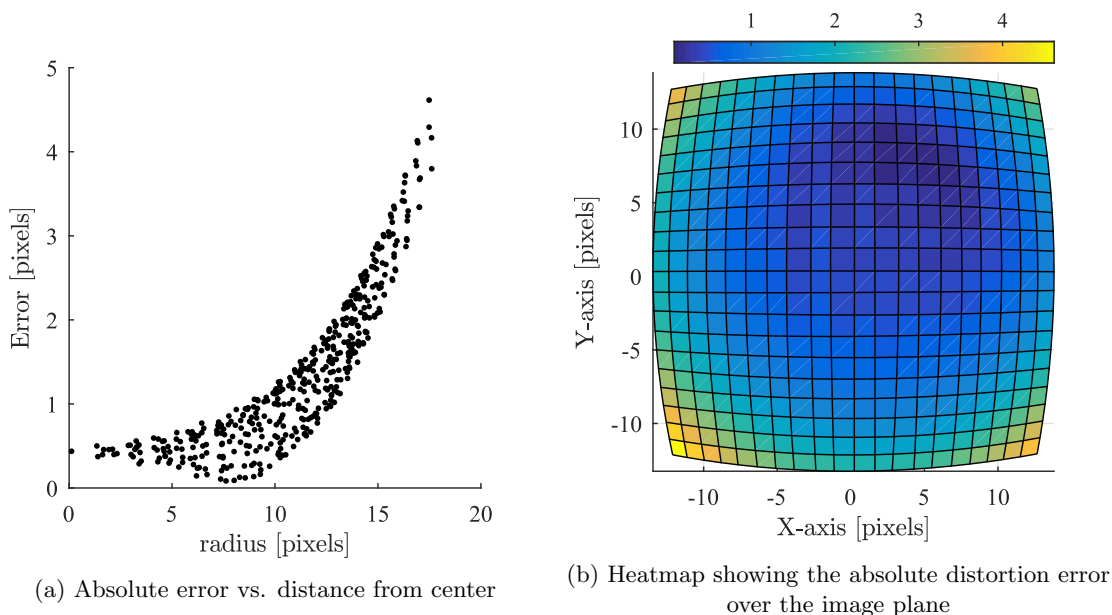


Figure 7.9: Investigation of lens distortion errors

Figure 7.9 also proves that the lens distortion does not consist of pure radial distortion, as a lens artefact is visible around the point (5, 5). Therefore, a more complex distortion model might be required to correct the lens distortion.

7.3.2 Different Distortion Correction Models

The lens distortion correction model defines how distorted coordinates are corrected. Different models vary in size, complexity, and accuracy, and it is important to define a model that describes the camera's lens sufficiently. Therefore, different models are proposed and compared. There are complex models used in literature which relate the models directly to the camera's intrinsic parameters (see Zhang[44]). These methods are, however, unnecessarily complex. Therefore simpler models are proposed and written in the format:

$$x_c = x + f(x, y) \quad (7.7)$$

where the distorted coordinates are represented by (x, y) , the lens distortion corrected coordinates by (x_c, y_c) ,

and $f(x, y)$ the lens distortion correction model.

2D Rectangular Model: The simplest model is a 2 dimensional model which determines x_c from x by translating the coordinate in the x and y -directions individually, depending on the original coordinates location, as depicted by Eq. 7.8 and 7.9. This creates a rectangular coefficient matrix.

$$x_c = x - u_0 + \sum_{j=0}^{R_y} \sum_{i=0}^{R_x} c_{1ij} x^i y^j \quad (7.8)$$

$$y_c = y - v_0 + \sum_{j=0}^{R_y} \sum_{i=0}^{R_x} c_{2ij} x^i y^j \quad (7.9)$$

The values R_x and R_y defines the size of the model, and the coefficients c_{1ij} and c_{2ij} represent the model's behaviour. This has the largest set of coefficients ($2(R_x + 1)(R_y + 1)$) which results in the most accurate correction. However, this does directly translate to longer processing times and more required memory. The high order coefficients, as $j \rightarrow R_y$ and $i \rightarrow R_x$, are close to zero and could potentially be ignored.

2D Trapezium Model: This model is similar to the 2D Rectangular Model, except it reduces the number of required coefficients. This is done by creating a trapezium-shaped coefficient matrix, instead of a rectangular matrix, as depicted by Eq. 7.10 and 7.11. This is to remove the need for the redundant high order coefficients present in the 2D Rectangular Model.

$$x_c = x - u_0 + \sum_{j=0}^{R_j} \sum_{i=0}^{R_i-j} c_{1ij} x^i y^j \quad (7.10)$$

$$y_c = y - v_0 + \sum_{j=0}^{R_j} \sum_{i=0}^{R_i-j} c_{2ij} x^j y^i \quad (7.11)$$

where

$$R_i \geq R_j \quad (7.12)$$

Again the values R_i and R_j defines the size of the model, and the coefficients c_{1ij} and c_{2ij} represent the model's behaviour. Note that the model's longest dimension (R_i) is in the direction of the variable (x or y) being corrected. The main advantage of this model is the decreased number coefficients, which reduces the number of computations without a significant increase in complexity.

It was also investigated to create a strictly triangular model by replacing the sum limit ($R_i - j$) with ($R_i - \frac{jR_i}{R_j}$) to limit the coefficient quantity further. However, this method showed no distinct difference to the Trapezium Model with an unnecessary increase in complexity and was therefore ignored.

Radial Model: This model assumes that the distortion present is strictly radial, i.e. only the radius around the boresight $c_0 = (u_0, v_0)$ is distorted. The coordinate (x, y) is then corrected by only manipulating the coordinate's distance from c_0 , as described by Eq 7.13 and 7.13. In Section 7.3.1 it is shown that the distortion measured isn't pure radial distortion, but a radial model might result in sufficient distortion correction.

$$r_c = r + \sum_{i=0}^R c_i r^i \quad (7.13)$$

$$r = \sqrt{(x - u_0)^2 + (x - v_0)^2} \quad (7.14)$$

The value R defines the size of the model, and c_i represents the model's behaviour. This model drastically lowers the number of coefficients to R , but it does require a complex computation for Eq. 7.14. The downside of this model is that it could introduce a significant error if the lens distortion is not strictly radial.

7.3.3 Fitting Lens Distortion Model

The different distortion correction models in Section 7.3.2 are fitted to the measured lens distortion using Least Squares algorithms, which is achieved by comparing the distorted coordinates with the expected coordinates. The general equation for Least Squares algorithms are in the form:

$$\mathbf{Ax} = \mathbf{b} \quad (7.15)$$

where \mathbf{A} describes the input variables combined with the chosen conversion formula, \mathbf{b} the required output variables, and \mathbf{x} the unknown coefficients of the conversion formula that should be calculated. Therefore, the 2D Rectangular Model's x -coefficients, for example, can be calculated using Eq. 7.16.

$$\begin{bmatrix} 1 & x_0 & \dots & x_0^{R_x} & y_0 & x_0 y_0 & \dots & x_0^i y_0^j & \dots & x_0^{R_x} y_0^{R_y} \\ 1 & x_1 & \dots & x_1^{R_x} & y_1 & x_1 y_1 & \dots & x_1^i y_1^j & \dots & x_1^{R_x} y_1^{R_y} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & x_m & \dots & x_m^{R_x} & y_m & x_m y_m & \dots & x_m^i y_m^j & \dots & x_m^{R_x} y_m^{R_y} \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \dots \\ c_n \end{bmatrix} = \begin{bmatrix} x_{c0} - x_0 + u_0 \\ x_{c1} - x_1 + u_0 \\ \dots \\ x_{cm} - x_m + u_0 \end{bmatrix} \quad (7.16)$$

where m is the quantity of available coordinates (or equations), c_n the model coefficients ($n = R_x R_y$), (x, y) the distorted coordinates, and (x_c, y_c) the corrected coordinates. The values for c_n are then solved using Eq. 7.17.

$$\mathbf{c} = (\mathbf{A}^T \mathbf{A})^{-1} (\mathbf{A}^T \mathbf{b}) \quad (7.17)$$

This method is easily expanded to solve all the mentioned models in Section 7.3.2.

7.3.4 Comparing Distortion Correction Models

The models discussed in Section 7.3.2 were compared by fitting the models to the distortion (Section 7.3.3), and in turn evaluating the residual error after the distortion correction. The accuracy of the models are heavily dependant on their respective orders (e.g. R_x and R_y), but the required processing time increases with the model order due to the increasing number of coefficients. Therefore, each model's performance is compared relative to the coefficient quantity required, as shown in Figure 7.10.

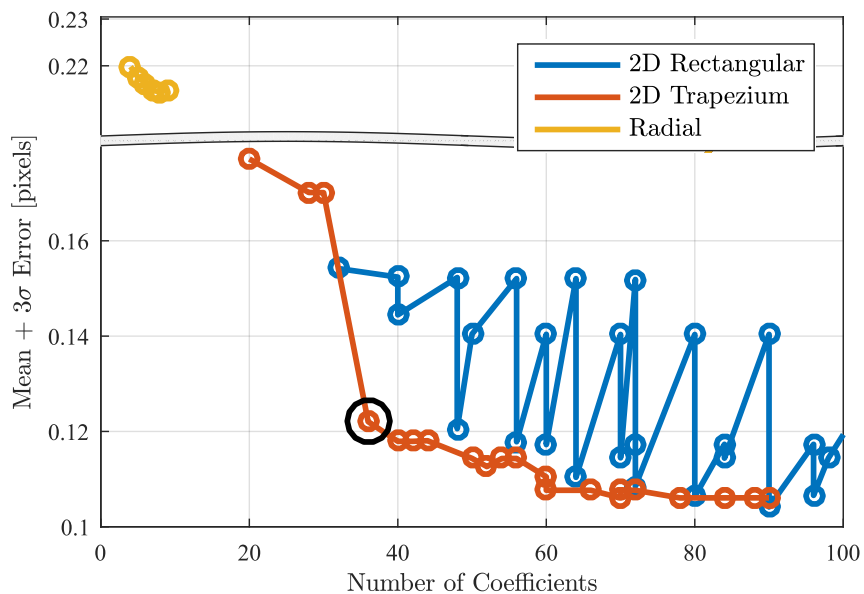


Figure 7.10: Comparing performance of distortion models vs. the amount of coefficients, with the (5,3)-order Trapezium Model encircled

It is clear that a radial model is not sufficient to correct the distortion, which is predicted in Section 7.3.1. The 2D Rectangular model's performance varies wildly with the increase of coefficients. This is due to the coefficient matrix shape varying over long rectangles, extended in the x or y directions (e.g. $R_x \gg R_y$); and squares ($R_x = R_y$) with an equal amount of coordinates. This effect is diminished by the 2D Trapezium Model where the model only extends in the direction of the coordinate being corrected.

It is also clear that 2D Trapezium Model performs the best providing the least amount of coefficients. This is enforced by the fact that the extra coefficients that the Rectangular Model utilises (e.g. $cx^{R_x}x^{R_y}$) are mostly very close to zero and therefore redundant. The Trapezium efficiently uses this amount of coefficients, and therefore this method was chosen. The order of the model was chosen as ($R_i = 5, R_j = 3$), because it has only 36 coefficients, while the performance increase with higher orders is negligible. This model is encircled in Figure 7.10.

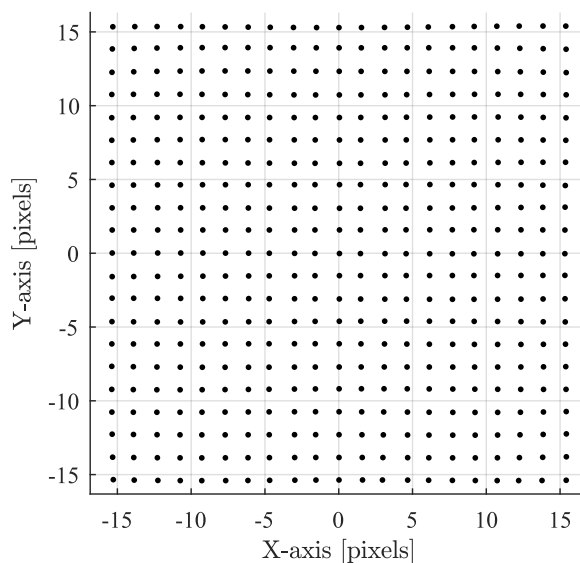


Figure 7.11: Lens distortion corrected coordinates from checkerboard

Figure 7.11 shows an example of distortion corrected coordinates from the simulated checkerboard. There were still slight offsets to their locations, as shown in Figure 7.12. These errors are relatively small and does not significantly influence results. This error is also mainly localised around the column $x = -3$, which only interferes with a few of potential edge coordinates under most conditions.

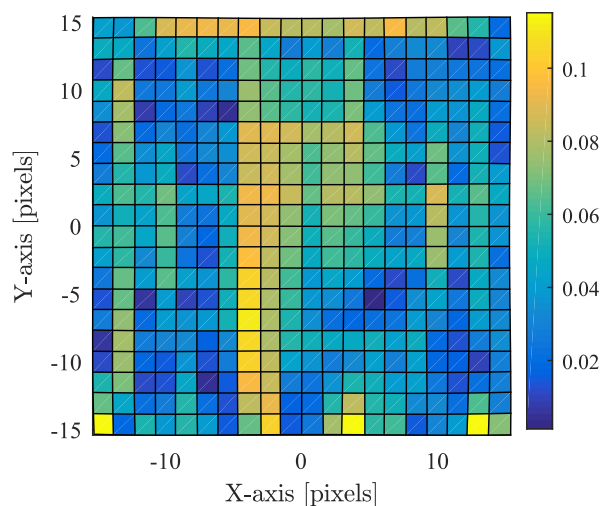


Figure 7.12: Heatmap showing error of corrected coordinates

Since the lens distortion is known the effective FOV of the camera was calculated. The horizontal FOV, for example, was calculated by correcting the image plane's original width (32 pixels) for distortion. Using this image plane's effective width, together with the pixel pitch ($220 \mu\text{m}$) and focal length (5.8 mm), the effective horizontal FOV is calculated as 71.32° . This was repeated for the vertical FOV, with the results are shown in Table 7.1.

Table 7.1: Infrared camera's effective FOV

	Horizontal	Vertical
Original Size	32 pixels	31 pixels
Original FOV	62.5°	60.9°
Effective Size	37.83 pixels	36.87 pixels
Effective FOV	71.32°	69.91°

7.4 Discussion

This section shows the modelling and correction of the inevitable lens distortion present on an infrared camera. The distortion modelling required an innovative new method as conventional methods were not viable for low-resolution infrared sensors. Therefore, modelling was achieved by imaging a straight heated steel plate in different configurations, from which a calibration checkerboard was created in software. This distortion was then corrected by correcting individual coordinates (instead of the entire image). The correction method utilises a 2D Trapezium shaped coefficient matrix which delivers accurate results while maintaining a fast execution time.

8 Software Implementation

This section discusses the software implemented on the MCU, the optimisations utilised, as well as the timing performance of the software. Descriptions of the software written for the PC interface to the MCU, or the MATLAB simulations, are not included.

8.1 Software Overview

The goal of the software on the MCU is to determine the satellite's attitude from images captured by the infrared camera. Additionally, the device should adhere to the design philosophy of conventional CubeSat design. For this reason, the software's design guidelines are:

- **Configurable:** It should be configurable (to a certain extent) during runtime to promote improved operation. Therefore different functionalities of the MCU should have the option to be enabled, disabled or modified, during operation.
- **Efficient:** Typical CubeSat attitude control systems require a sensor measurement at 1 Hz . Subsequently this device should have a measurement available every second. This enables the low power requirement of satellite design, as the MCU can enter a low power state when not in use.
- **Robust:** If the sensor is required to run with minimal external intervention it should be robust, as crashing during operation will be complex to analyse and correct, especially in space conditions.

However, using a low power 8-bit processor results in several limitations which have to be overcome, namely:

- **Storage:** The processor only has 3760 bytes of SRAM, which severely limits the software. Innovative ways had to be found to implement communications and attitude measurements within the limited storage.
- **Timing:** All calculations done by the processor is done through 8-bit numbers, which results in floating point operations (32 bits), or even integers (16 bits), is relatively time-consuming. This limits the amount of time available for calculations. To relieve this problem slightly for prototyping a 32 MHz clock speed is used.

8.2 Detailed Descriptions

This section describes the essential aspects of the software used to measure the satellite's attitude by only using the MCU. All the steps used for this measurement were controlled through an onboard finite state machine and executed automatically, but can also be controlled remotely through the use of telecommands (see Section 8.2.2).

8.2.1 Storing of Variables

The choice of different variable types for different algorithms severely influences the accuracy and processing time of the sensor. Floating point variables are very accurate, but require a substantial amount of memory and are slow to process. Integers, for example, are fast to compute and require little memory, but the accuracy

is limited. Therefore, well-designed software finds the correct balance between the accuracy and processing time. This sensor only to use floating point variables where necessary, and integers (or characters) elsewhere. The main variables are stored as follows:

- **Pixels:** The pixel values are stored in a 32×31 array as 16-bit integers in the MCU's RAM. Each value represents its corresponding pixel's V_{obj} (Section 5.1). These pixels are stored in Analogue to Digital Converter units (ADCU) instead of Volts to save processing time as only relative magnitude is required. The system noise on each pixel is $3\sigma = 2.3$ ADCU, which renders floating point variables unnecessary. Although a range of roughly -100 to 100 ADCU is required for each pixel (which only requires an 8-bit variable), the pixel values are accumulated for an averaging effect. Therefore 16-bit variables are used to eliminate the possibility of integer overflow.
- **Coordinates:** Roughly half of the processing is done on pixel coordinates, e.g. Lens Distortion Corrections and circle fitting through Least Squares. These methods require high accuracy variables as it utilises high order polynomials, and therefore these variables are stored as floats in the MCU's RAM. The number of detected coordinates are limited to 32 to ensure deterministic timing, which is slightly more than the maximum amount expected according to Figure 6.16e.
- **Calibration Values:** There are three main calibration values to be stored in the MCU's ROM. The Lens Distortion (Section 7) and Post Calibration (Section 9.3.2) both require coefficients for high order polynomials (typically smaller than 1.0), which is therefore stored as floats. The V_{off} -values (Section 5.1) should also be stored, and since it is simply subtracted from integer pixel values, and in the range of 20 to 40 ADCU, these values are stored as 8-bit integers.

8.2.2 Communication Protocol

It is important that the sensor has a robust communications system to maximise ease of use while having the functionality of typical satellite subsystems. Therefore, the communication protocol used by the company CubeSpace is implemented on this sensor[45]. Although the hardware is available on the sensor to implement the UART, I2C and CAN interfaces simultaneously, only the UART is implemented.

The UART operates at a baud rate of 115200 *bps*, uses 8 bits, 1 stop bit and no parity. The protocol itself makes use of three identifiers. A start-of-message (SOM=0x1F) and end-of-message (EOM=0xFF) to mark the start and end of any transmission. The first two identifiers are always preceded by an escape character (SC=0x1F). If the transmission contains a message, it is situated between the SOM and EOM. If any of the message data bytes match the escape character, that data byte is replaced by two escape characters (0x1F,0x1F). A typical transmission can then be expanded to have an ID byte as follows:

Table 8.1: UART transmission decoding

Byte	Meaning
0	SC
1	SOM
2	ID
3	message*
...	message*
n-2	message*
n-1	SC
n	EOM

* Optional

The ID byte of the transmission communicates to the sensor if it is receiving a telecommand (TC) or a telemetry request (TLM). This is done by setting the most significant bit (MSB) of the ID if transmitting a

TLM to the sensor, or clearing it for a TC. This means, if observing the full ID byte, a TLM will have an ID byte equal to 128 or larger, while smaller ID bytes signify a TC. The sensor is also equipped with internal flags to denote errors occurring within the transmission, as well as other parts of the software. Once set these flags stay set until read through a TLM. A list of all TC and TLM commands are given in Appendix B.

8.2.3 Pixel Acquisition and Calibration

The infrared camera sequentially transmits the image data to the MCU through two analogue lines [6], which the MCU in turn stores in its internal RAM after the analogue to digital conversion. The timing of this transmission is shown in Figure 8.1, where OUT_A1 and OUT_A2 represent the camera's two output analogue lines, and VSAM (referenced from here forth by V_{SAM}), a signal showing the MCU when the sample is ready.

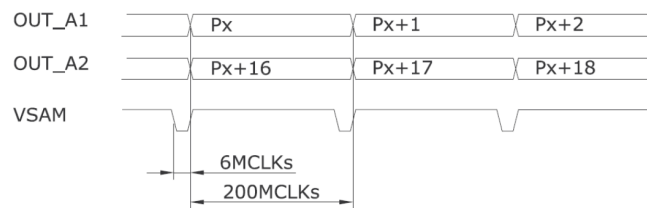


Figure 8.1: Sample timing of pixel transmission (1 MCLK=1 μ s)[6]

Figure 8.1 shows that 200 μ s are available to sample each pixel's analogue signal. However, due to the low pass filter described in Section 4.3.2, the analogue signal will only have reached its steady state after roughly 80 μ s. Therefore, a 100 μ s delay (using timers and interrupts) is added after the V_{SAM} -trigger to ensure that a steady state signal is read. This is shown in Figure 8.2, which depicts how the pixel acquisition and storage processes are only initiated after 100 μ s.

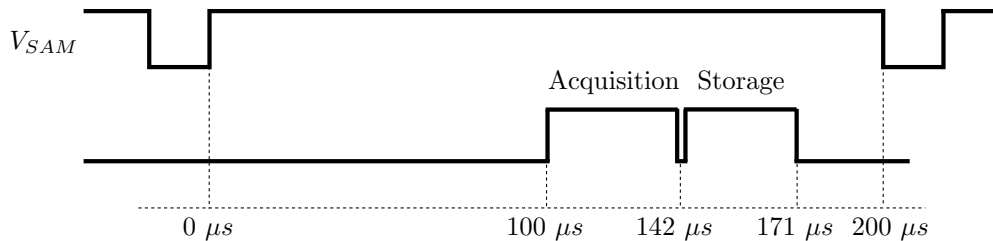


Figure 8.2: Expansion of pixel acquisition and storage timings according to oscilloscope readings

For each image, a total of 1056 packets are transmitted over the two transmission lines, and therefore a single image takes $(1056/2) \times 206 \mu$ s = 108 ms to transmit. When images are averaged (for an increased SNR, see Section 3.3.3), the pixel values are accumulated, and not divided by the total amount of averaged images. This increases the resolution through not losing bits during the division operation. After all relevant image information is stored, in the MCU's internal RAM, the pixels are thermally calibrated to ensure a uniform response, as discussed in Section 5.2.1.

Each pixel's calibration value V_{off} is dependant on the camera's housing temperature, which means that the correct calibration value has to be calculated relative to V_{AORt} (an indication of the temperature).

The dependency between V_{AORt} and V_{off} is linear in the operating range (see Figure 5.3), and therefore an interpolation between two points is used to determine the required V_{off} for each pixel. Interpolation calculations require a division calculation which is processing intensive on an 8-bit MCU. To minimise this processing time, the two calibration values stored are designed to differ by a V_{AORt} of 512 $ADCU$. This enables the possibility of an integer division by 2^n which the compiler can implement as a bit shift.

8.2.4 Sobel Edge Filter

The Sobel Edge Filter (explained in Appendix A) is required to calculate the gradient image for the edge detection algorithms (see Section 6.1.5). Although the Sobel Operator is optimised for fast execution, running it on an 8-bit MCU can still be slow due to the square root calculation, which is required to calculate the magnitude of the gradient vector. Each square root takes roughly 837 μs , which means calculating the gradient for 992 pixels will require an extra 830 ms , which is not a viable solution. However, the magnitude calculation can be approximated with multiplication and addition which is less computationally expensive. Assume a triangle depicted in Figure 8.3, where a and b represents the gradient vector, and c the absolute magnitude:

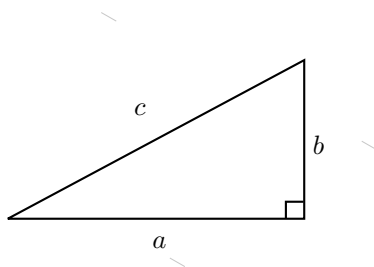


Figure 8.3: Approximation of magnitude calculation

Conventionally, the magnitude is calculated using the Pythagoras Theorem, as shown in Eq. 8.1 which is very computationally expensive.

$$c = \sqrt{a^2 + b^2} \quad (8.1)$$

However, the value for c can be approximated, as seen in Eq. 8.2, where a is longer than b . This calculation is very computationally inexpensive.

$$c \approx a + \frac{b}{2} \propto 2a + b \quad (8.2)$$

The approximation of $c = 2c$ is sufficient, as only a relative gradient is required, and not the absolute gradient. This approximation will change the behaviour of the Sobel operator and therefore the behavioural change was investigated. A small simulation was run (results in Figure 8.4) to determine the gradient for a constant edge at different rotations which should ideally result in a uniform response. This simulation showed that the approximation is not perfectly radial symmetrical and only varies by 10% more than the normal Sobel Filter. This is sufficient for this sensor's design goals, and therefore this approximation is utilised.

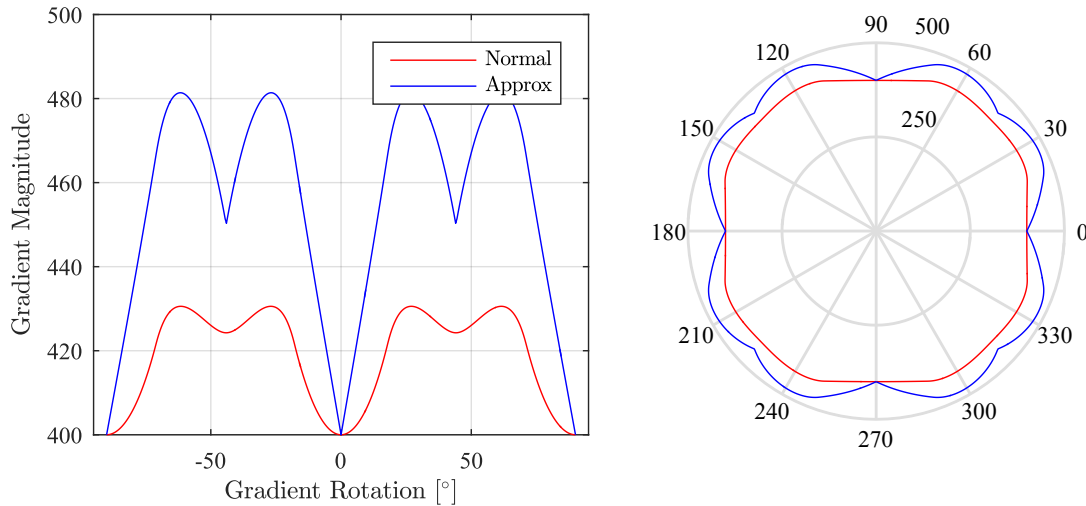


Figure 8.4: Comparison between the radial response of normal Sobel Filter and the proposed approximation

An alternative to using this rough approximation of magnitude calculation would be to substitute only the square root in Eq. 8.1 with the Fast Inverse Square Root (FISR)[46]. This will potentially decrease the total time spent on square root calculations from 830 *ms* to 167 *ms*, which is viable. However, this will still limit the number of images that can be averaged and is therefore not implemented.

Another limiting factor of the Sobel Filter is that each gradient pixel is dependant on multiple original pixels. This means that when calculating the gradient pixels, it cannot simply overwrite the original image in the RAM. Ideally, both the original and gradient images should simultaneously be stored in memory, but the selected MCU does not have enough RAM. 992 pixels using two bytes each requires 1984 bytes of memory, which means two complete images will not fit in the 3760 bytes of available RAM. Therefore, as the original image is not entirely necessary, the original image is overwritten by the gradient image with the assistance of a small buffer.

The gradient image is calculated row-by-row, where original pixel knowledge is required for the two neighbouring rows. Therefore a buffer to store only the current and previous row of original pixels are required. This is shown in Figure 8.5 where the third row's gradients are being calculated using row 4 and the buffer's rows (*A* and *B*) which contains the original pixels of rows 2 and 3 respectively. After each the row's gradients is calculated, the next row is copied into the buffer. In this example row 4 would be copied into buffer *A*, as buffer *B*'s data is still required for the gradient calculation of row 4.

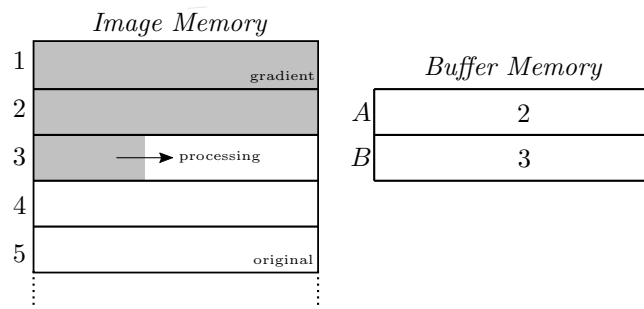


Figure 8.5: Visualisation of dynamic buffer when calculating the Sobel Filter showing only pixel rows

8.2.5 Predictive Edge Detection

This function detects the maxima of the gradient image (corresponding to the Earth disc edge) and finds corresponding coordinates on the maximum of this edge. This is done by implementing a Local Extrema method (Section 6.1.5) in conjunction with a predictive scanning technique (Section 6.3.4).

The Predictive Scanning is executed in two main parts, first the column-by-column scanning, and (if required) the row-by-row scanning. To determine which scanning pattern to execute the CoI rotation estimate is used (see Section 6.3.1). The row-by-row scanning is only activated if $|\tilde{\theta}| < 30^\circ$. However, the determination of the CoI angle $\tilde{\theta}$ is an expensive trigonometric calculation, and therefore the CoI coordinate itself is used to select the scanning pattern. This is done by drawing two mirrored diagonal lines and determining if the CoI is in the area between these two lines. This shown in Figure 8.6 where the lines are shown in red, and the subsequent area is shaded red.

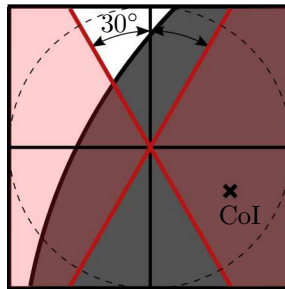


Figure 8.6: Visualisation of scanning pattern determination

Determining the scanning pattern using this method eliminates the need to calculate the angle $\tilde{\theta}$, and in turn only requires a conditional statement as follows (assuming (x, y) represents the CoI):

```

Require:  $m = \tan^{-1} 60^\circ$ 
if  $(y > mx \text{ AND } y < -mx)$  OR  $(y < mx \text{ AND } y > -mx)$  then
    Scan columns and rows
else
    Scan only columns
end if

```

This significantly decreases the execution time of this function without a loss in accuracy, because it is geometrically equivalent to using the angle $\tilde{\theta}$. Once the scanning pattern is determined the pixels are scanned accordingly. This is implemented efficiently by not scanning pixels outside the chosen scanning pattern.

The pixels and coordinates are stored as integers and floats respectively, which means that a conversion is required. Integers are less computationally expensive and should be utilised as long as possible (without a loss in accuracy). Therefore, the values are only converted during the calculation of the coordinates using Eq. 6.4. This equation's numerator and denominator are still calculated as integers and are then converted to floats for the division.

8.2.6 Least Squares Circle Fit

When fitting a circle to the various edge coordinates, using the method discussed in Section 6.5.3, a Least Squares algorithm is implemented on the MCU. This calculation requires high accuracies and therefore all

calculations are done in 32-bit floating point operations. This requires a large amount of memory and processing time. Hence the number of variables to be stored and processed is minimised.

The $\mathbf{M}^T\mathbf{M}$ -matrix (see Eq. 6.18) is populated by iterating through the acquired edge coordinates. This matrix is symmetrical which means it only has six unique elements, and hence the 3×3 matrix can be stored in an array of only six elements. The array stores the matrix elements as follows:

$$[0 \ 1 \ 2 \ 3 \ 4 \ 5] = \begin{bmatrix} 0 & 1 & 2 \\ 1 & 3 & 4 \\ 2 & 4 & 5 \end{bmatrix} \quad (8.3)$$

Furthermore, to reduce for-loop overhead and unnecessary complexity, all matrix calculations are implicitly defined element-by-element. To avoid unnecessary division calculations during determinant calculation, the division coefficient is calculated (the reciprocal of the value to divide by) and all the elements are simply multiplied by this value.

It was attempted to increase the speed of the function by substituting the supplied square root function with the FISR[46]. However, the square root operation is not the limiting factor of this function regarding processing time, which renders this optimisation redundant and therefore the FISR is not implemented.

8.2.7 Lens Distortion Correction and Post Calibration

The lens distortion correction (Section 7.3.2) and Post Calibration (Section 9.3.2) processes are very similar processing-wise, and therefore these two functions were designed similarly. For the sake of simplicity, only the Post Calibration function is described here, as the same techniques are used for the lens distortion correction. As this is a complex calculation using processing intensive floats, the amount of calculation is minimised to maximise efficiency.

In Eq. 9.2 the offset itself is calculated by multiplying two vectors: one containing variations of ϕ and θ values, and the other a set of predetermined coefficients. The vector containing ϕ and θ can be rewritten in matrix format as:

$$v(\phi, \theta) = \begin{bmatrix} 1 & \phi & \dots & \phi^{R_\phi} \\ \theta & \phi\theta & \dots & \phi^{R_\phi}\theta \\ \vdots & & & \vdots \\ \theta^{R_\theta} & \phi\theta^{R_\theta} & \dots & \phi^{R_\phi}\theta^{R_\theta} \end{bmatrix} \quad (8.4)$$

As each element in this matrix follows the same pattern for each row (or column), it was decided to calculate the elements row-by-row. For each row, the first number in the row is stored (*b_mult*) after which the different column's values (*mult*) are calculated by simply multiplying by ϕ . To calculate the next row's value *b_mult* is multiplied by θ , after which the same process is followed. This requires the storage of only two floating point variables (instead of the entire matrix), as well as minimising the number of multiplications to $2N - 1$ for N elements. This process is shown below in pseudocode to calculate the offset ϵ , where $c(i, j)$ represents each element's corresponding c_{1ij} .

```

Require:  $\phi, \theta$ 
 $mult = b\_mult = 1$ 
 $\epsilon = 0$ 
for each row  $j$  do
   $mult = b\_mult$ 
  for each column  $i$  do
     $\epsilon = \epsilon + mult \times c(i, j)$ 
     $mult = mult \times \phi$ 
  end for
   $b\_mult = b\_mult \times \theta$ 
end for

```

8.3 Program Timing

The total processing time to measure the satellite's attitude should be under 1 second. Therefore all the applicable functions' execution times were measured using an oscilloscope with the final results shown in Table 8.2. It was decided to maximise the number of images to capture (and average) for each measurement to minimise noise. To find the maximum number of images that can be averaged, it was determined that the image processing and attitude measurement requires 380 *ms*, which leaves 620 *ms* to capture and average images. As each image capture takes 108 *ms* (see Section 8.2.3), it was determined that five images can easily be averaged for each attitude measurement. This leaves ample a 90 *ms* for communications, or putting the processor in a low power state to save power.

Table 8.2: Worst case timing for the entire process over 1 second

Process	Timing over 1 second
Image Acquisition	540 <i>ms</i>
Pixel Calibration	113 <i>ms</i>
Sobel Filter	109 <i>ms</i>
Edge Detection	15 <i>ms</i>
Lens Dist. Corr.	112 <i>ms</i>
Shape Fitting	15 <i>ms</i>
Att. Est. & Calib.	5 <i>ms</i>

It should be noted that the timings in Table 8.2 correspond to the worst case scenario. The only condition that changes this timing is the number of coordinates found (and processed) during a measurement. To limit the required processing time the number of edge coordinates are limited to 32, which is just above the maximum amount expected (see Figure 6.16). Furthermore, all software is written to have deterministic timing, where possible, to ensure consistent execution times.

8.4 Discussion

This software implemented on this device was successful. It met all the criteria, especially regarding robustness. The communications were sufficient to send and receive all required data reliably, and the timings were consistent. This also shows that the MCU's properties were sufficient, as only 77% of the available 3760 bytes of RAM were used, and only 28% of the program memory, without the requirement of the external FRAM. Additionally, the attitude could be measured within the designated time span of 1 second.

9 Ground Test Results

This section will investigate the performance of the designed horizon sensor in emulated space conditions. It is important to evaluate the sensor in real-world conditions to ensure operation in a non-ideal environment.

9.1 Methodology

The evaluation of the horizon sensor includes all of the algorithms and techniques discussed in this project. The general process of determining the attitude from a raw infrared image is shown in Figure 9.1 followed by a brief description of each step.

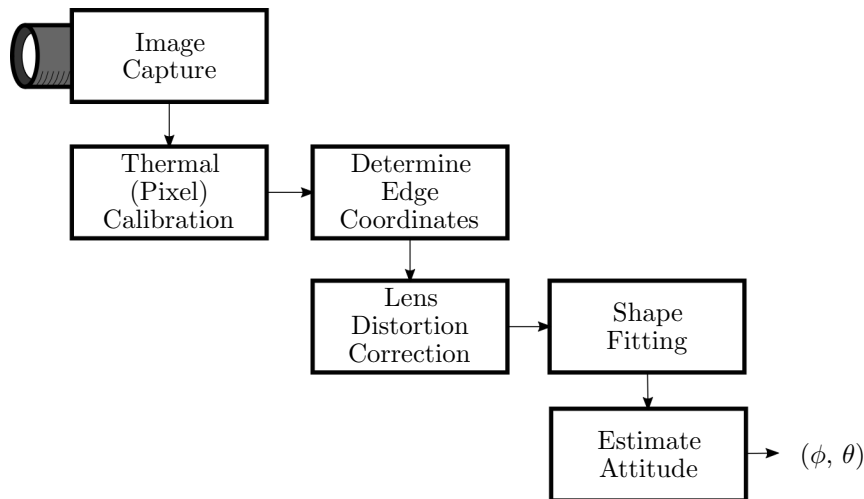


Figure 9.1: Flow diagram showing the attitude estimation process

A brief description of each step:

1. **Image Capture:** Capturing infrared image of the heated steel plate discussed in Section 3.5.
2. **Thermal (Pixel) Calibration:** Thermally calibrating each pixel to ensure uniform pixel behaviour over the entire image as discussed in Section 5.
3. **Determine Edge Coordinates:** Determining edge coordinates on the image using the Local Extrema method (Section 6.1.5) in conjunction with Predictive Scanning (Section 6.3.4).
4. **Lens Distortion Correction:** Correcting each individual edge coordinates for lens distortion as discussed in Section 7.
5. **Shape Fitting:** Fitting a circular shape to the edge coordinate to represent the visible Earth disc as discussed in Section 6.5.3.
6. **Estimate Attitude:** Estimate the sensor's attitude (also discussed in Section 6.5.3), and implementing sufficient calibration techniques (investigated in this section).

Initially, data is only captured by the infrared camera (Step 1) and processed in MATLAB (Step 2 to 6)

to retrieve the attitude, as shown in Section 9.3.1-9.3.2. After the operation of the various algorithms are confirmed the attitude is determined by using the sensor's microprocessor. This is done to confirm the working of the microprocessor's firmware and will be discussed in Section 9.4. The performance of the algorithms is evaluated by imaging the ground test setup as described previously in Section 3.5.

9.2 Misalignment Investigation and Correction

As described in Section 3.5.3, inevitable misalignments in the ground test setup severely limits the available performance of the horizon sensor. Therefore steps were taken to minimise this effect. The limited hardware available and the nature of the physical setup resulted in the setup not being perfectly aligned. To promote accuracy, it was decided to calibrate the horizon sensor in software, as this would mimic the actual alignment calibration process of a typical satellite sensor.

The three main misalignments are as follows:

- **Z_T-Offset:** The heated steel plate translated in the Z_T -axis (up or down). This will cause a constant offset in the measured elevation angle but is easily correctable.
- **Y_T-Offset:** The heated steel plate is translated in the Y_T -axis (left or right). This will cause large varying offsets in the measured elevation and rotation angles and is therefore complex to calibrate. (this misalignment is not possible in space conditions)
- **Rotation Offset:** The heated steel plate rotated around the X_T -axis. This will cause a constant offset in the measured rotation angle but is easily correctable.

Calibrating the setup yields the same problems as seen during lens distortion correction. The radiance of the heated air, in conjunction with an unfocused lens, results in the measured edge being offset from the actual edge location. Therefore a similar approach was used as discussed in Section 7. However, this approach had to be modified to work with a circular edge instead of a straight edge. This was done by utilising the techniques investigated in Section 6 to fit a circle to the observed disc. To create a sufficient calibration image, four straight lines were created, each representing the Z_T direction observed from a different camera rotation ($\theta = [0^\circ, 90^\circ, 180^\circ, 270^\circ]$). Each straight line was estimated by observing the disc from 7 different elevation angles ($\phi = [0^\circ, \pm 10^\circ, \pm 20^\circ, \pm 25^\circ]$) and subsequently fitting a straight line through the fitted circle centers. This is shown in Figure 9.2, where the plotted small circles represent the various fitted circle centers (fitted discs shown for *solid* small circles). In contrast to Section 7, this image includes setup misalignments. For each fitted circle, a set of 25 images were taken and averaged to remove noise from the calibration process effectively.

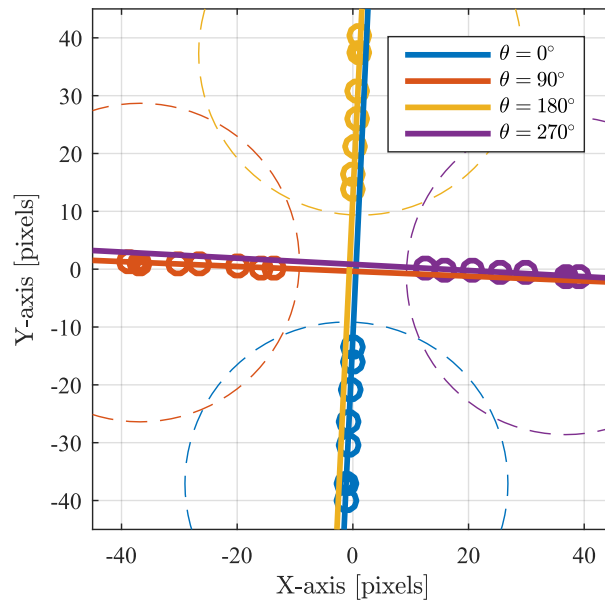


Figure 9.2: Fitted circle centre distribution from multiple images

9.2.1 Calculating Misalignments

From Figure 9.2, the previously mentioned misalignments were calculated with the results shown in Table 9.1. They were calculated as follows:

Z_T -Offset: This was calculated by utilising the average measured circle center for the rotations 0° and 180° , as these were the only points containing information about the possible Z_T offset. The difference in Z_T offset for both of the rotations was then averaged, which resulted in the effective elevation offset. This is shown in Figure 9.3.

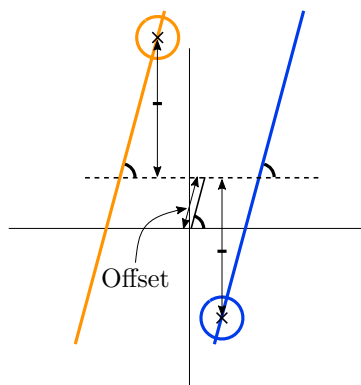


Figure 9.3: Method to determine the Z_T misalignment (or offset)

Y_T -Offset: This was calculated by determining the absolute distance between the two vertical lines (rotations of 0° and 180°) and dividing the result by two. This is illustrated in Figure 9.4.

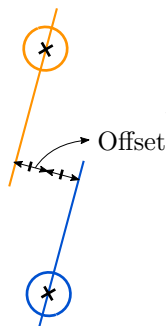


Figure 9.4: Method to determine the Y_T misalignment (or offset)

Rotation Offset: The rotation offset was calculated by averaging the gradients for rotations 90° and 270° and the inverted gradients for rotations 0° and 180° . The rotation offset was then equal to the average of all the calculated gradients.

Resulting Misalignments: The misalignments were calculated using Figure 9.2 in conjunction with the previously mentioned methods. The calculated misalignments are shown in Table 9.1.

Table 9.1: Main misalignments present in ground test setup

Misalignment	Magnitude
Z_T Offset	0.1 pixels
Y_T Offset	0.58 pixels
Rotation Offset	-2.7°

9.2.2 Correcting Misalignments

Z_T - and Y_T -Offset: These misalignments essentially translates the emulated Earth disc in the (Y_T, Z_T) -plane. This translation is projected onto the (X_C, Y_C) -plane and corrected during runtime. This projection requires knowledge of the horizon rotation which is available through the initial rotation estimate using the CoI (see Section 6.3.1). The correction is done by appropriately translating the already-fitted circle's center (x, y) on the (X_C, Y_C) -plane using Eq. 9.1.

$$\begin{bmatrix} x_c \\ y_c \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} -\cos \tilde{\theta} & -\sin \tilde{\theta} \\ \sin \tilde{\theta} & -\cos \tilde{\theta} \end{bmatrix} \begin{bmatrix} \tilde{\epsilon}_x \\ \tilde{\epsilon}_y \end{bmatrix} \quad (9.1)$$

where $\tilde{\theta}$ is the initial horizon rotation estimate, and $\tilde{\epsilon}_x$ and $\tilde{\epsilon}_y$ are the misalignments in the Y_T and Z_T directions respectively.

Rotation Offset: This misalignment is simply corrected by subtracting the rotation offset from the final measured rotation angle.

Example of Misalignment Correction: Figure 9.5a shows the estimated attitude with misalignments for various horizon elevation angles and a constant rotation angle. Apart from constant bias errors, the rotation

error (θ) varies wildly. This is due to the Z_T - and Y_T -misalignments. Using the calibration methods discussed in this section, the misalignments were corrected and a uniform response was acquired as seen in Figure 9.5b.

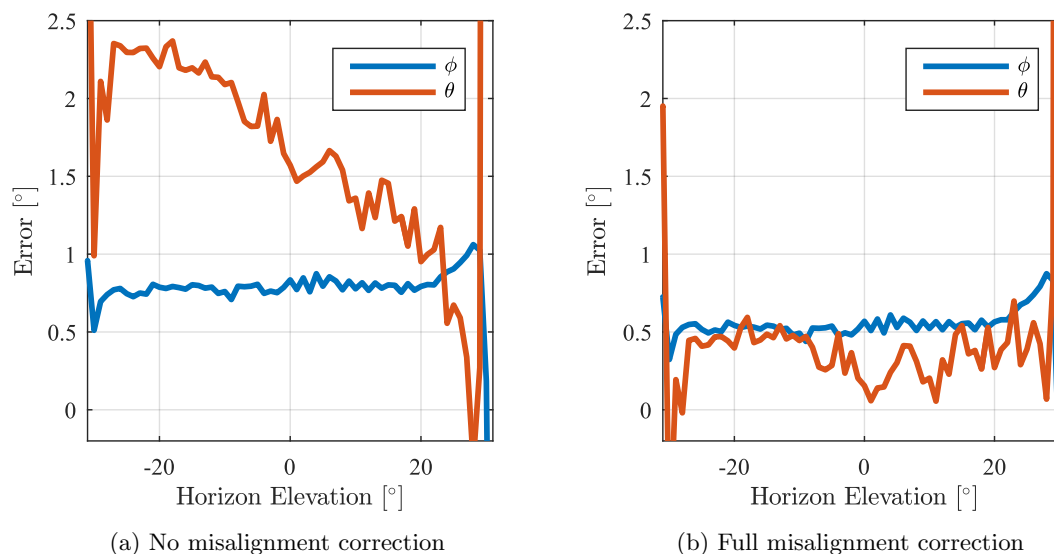


Figure 9.5: Example of applied misalignment correction for horizon with a rotation of $\theta = -15^\circ$. (ϕ and θ represents the horizon elevation and rotation errors respectively)

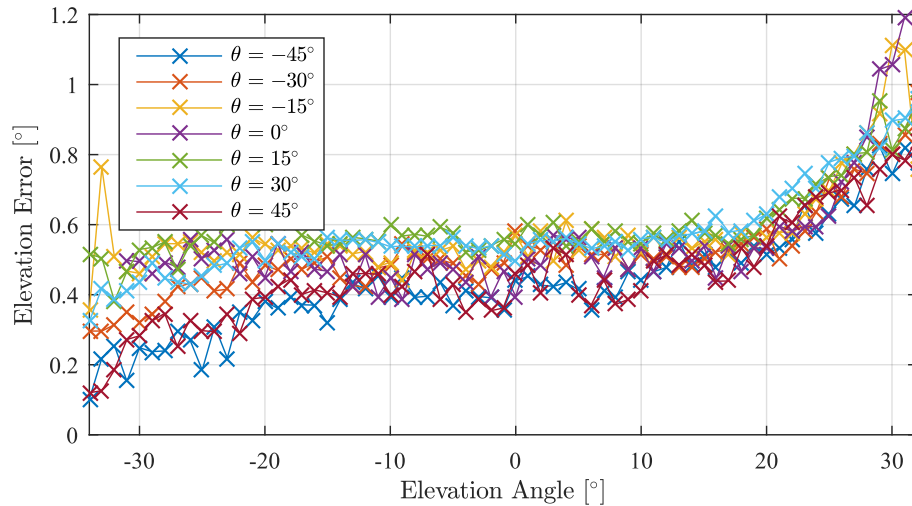
Constant bias errors are still present, but this is due to other effects which will be discussed in more detail later in this section. It should also be noted that for this misalignment correction illustration, the CoI estimate was not used but rather a hardcoded rotation angle. This was because the ground test setup's CoI was not an accurate estimation solely due to the non-uniform heating described in Section 3.5.4. This temporary simplification was sufficient because in space conditions the CoI estimate would be accurate, as shown in Section 6.3.1.

9.3 MATLAB Results

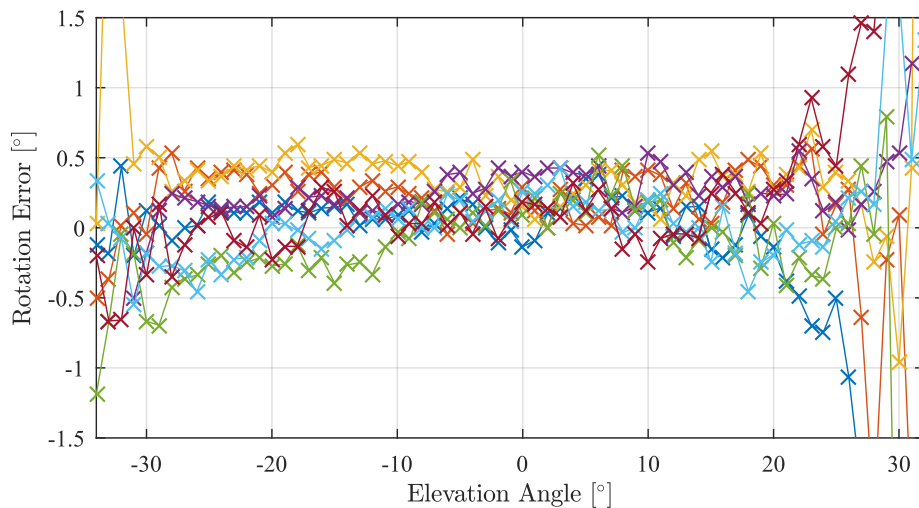
This section will evaluate the performance of the sensor's algorithms using real data processed in MATLAB. It will also describe a Post Calibration process which maximises the sensor's accuracy.

9.3.1 Initial Results

This section shows attitude estimation performance in the case where emulated data (imaged by the infrared camera) is processed in MATLAB. Due to the low throughput of the ground test setup (Section 3.5) only a limited number of samples were acquired. A collection of eight tests were run at various horizon rotation angles, with each test measuring the horizon at varying elevation angles. For each elevation angle, a combination of 5 images was taken and averaged to emulate a typical sensor implementation. These tests included the misalignment correction discussed in Section 9.2, and the results are shown in Figure 9.6.



(a) Measured elevation error over an elevation sweep



(b) Measured rotation error over an elevation sweep

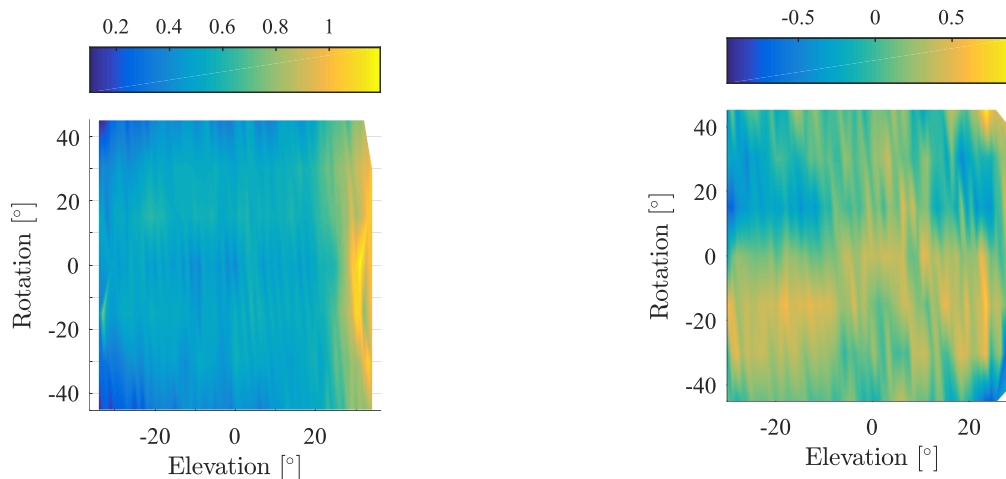
Figure 9.6: Attitude estimation obtained through an emulated horizon and MATLAB

These results show fairly large residual bias errors of up to 0.5° . These bias errors and the general sensor response is described by the following attributes:

- Accuracy drops significantly below -30° and above 26° horizon elevation. This was mainly due to the drop in edge coordinate quantity (explained further in Section 9.3.3).
- There is a varying bias error in the elevation measurement, which is proportional to the elevation angle. This slightly resembles a low contrast sigmoid function.
- There is a varying bias error in both the elevation and rotation measurements which is proportional to the horizon rotation, and has clear peaks and troughs.
- Both elevation and rotation errors have inherit noise of roughly 0.1° and 0.3° in magnitude.
- The elevation error is fairly symmetrical around the $\theta = 0^\circ$ axis, while the rotation error seems to be symmetrically inverted around the $\theta = 0^\circ$ axis.

These bias errors are simpler to visualise in absolute error heatmaps, as shown in Figure 9.7. Although a large

bias error exists, the variation in elevation error is fairly close to the desired 0.1° . Similarly, the variation in rotation error is slightly larger at roughly 0.3° . This means that if the bias error is dynamically removed, high accuracies of up to 0.1° and 0.3° respectively are possible.



(a) Measured elevation error over various elevation and rotation angles

(b) Measured rotation error over various elevation and rotation angles

Figure 9.7: Heatmap of attitude estimation errors (in degrees) obtained through the emulated horizon and MATLAB

There are various possible sources of these bias errors. These include:

- Slight ground setup misalignments which are not accounted for. For example a camera mounting error. Such misalignments are complex to quantify due to the camera's limited resolution and limited hardware.
- Non-uniform heating of the steel plate (discussed in Section 3.5.4). This will possibly cause non-uniform edge coordinate acquisition. Additionally it will cause uneven heating of the air which will amplify this effect.
- Varying amount of edge coordinates. As the horizon moves over the image plane the amount of edge coordinates detected varies which influences the detection algorithms.
- Non-uniform thermopile sensitivity. Commercial thermopiles don't have a uniform response over the entire FOV. This entails that the sensitivity decreases as the incoming radiation's angle of incidence near the FOV's limit[47]. This will also result in non-uniform edge coordinate acquisition

These effects are either quantifiable or not present in space conditions, and most importantly these effects are deterministic which makes them correctable. Therefore a post-calibration process was utilised to improve the accuracy and will be discussed in the next section.

9.3.2 Post Calibration

As discussed in Section 9.3.1, there were several factors that caused bias errors in the measured attitude. This includes non-uniform sensor sensitivity, non-uniform heating and slight uncorrected misalignments. Because these errors were deterministic (and not stochastic) it was post calibrated by removing the expected bias error relative to the measured elevation and rotation angle. This was done by utilising the Least Squares method, similar to the lens distortion correction in Section 7.3.

The model with which to describe the bias error was chosen as a simple square coefficient matrix, depicted by Eq. 9.2. This is similar to the 2D Rectangular Model in Section 7.3.2. This model was chosen as it was simple to implement. In this case the size was not a limiting factor as the processor had enough code storage memory, and the processing time was small because it was only be executed twice every second.

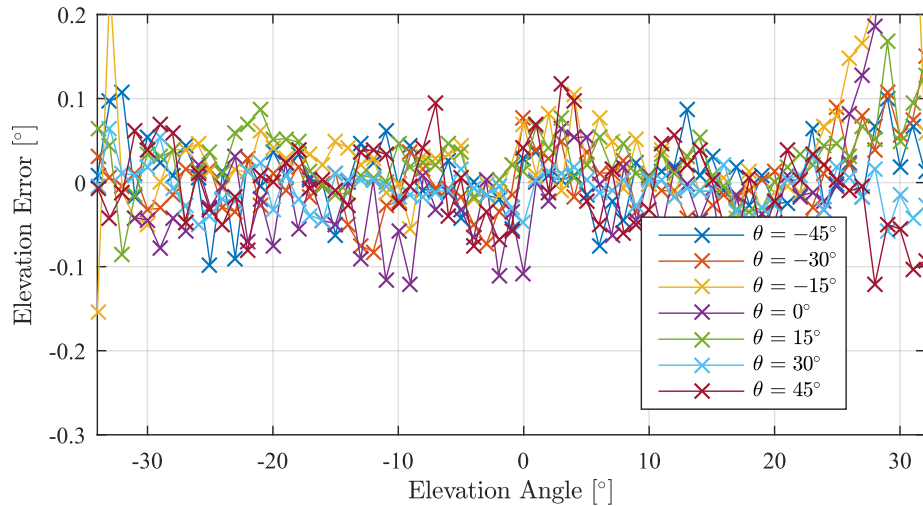
$$\begin{aligned}\phi_c &= \phi - \sum_{j=0}^{R_\theta} \sum_{i=0}^{R_\phi} c_{1ij} \phi^i \theta^j \\ \theta_c &= \theta - \sum_{j=0}^{R_\theta} \sum_{i=0}^{R_\phi} c_{2ij} \phi^i \theta^j\end{aligned}\tag{9.2}$$

In Eq. 9.2 the measured elevation and rotation angles are represented by ϕ and θ respectively and the coefficients c_{1ij} and c_{2ij} represent the model's behaviour. R_ϕ and R_θ represent the order of the model in its respective dimensions. In Figures 9.7a it is clear that the rotation bias error has two peaks and three troughs, and therefore the model order of $R_\theta=4$ was initially chosen to follow this curve. However, it was found that $R_\theta=4$ was not sufficient, and therefore $R_\theta=5$ was chosen as the final value. Similarly, the order R_ϕ was chosen as 3.

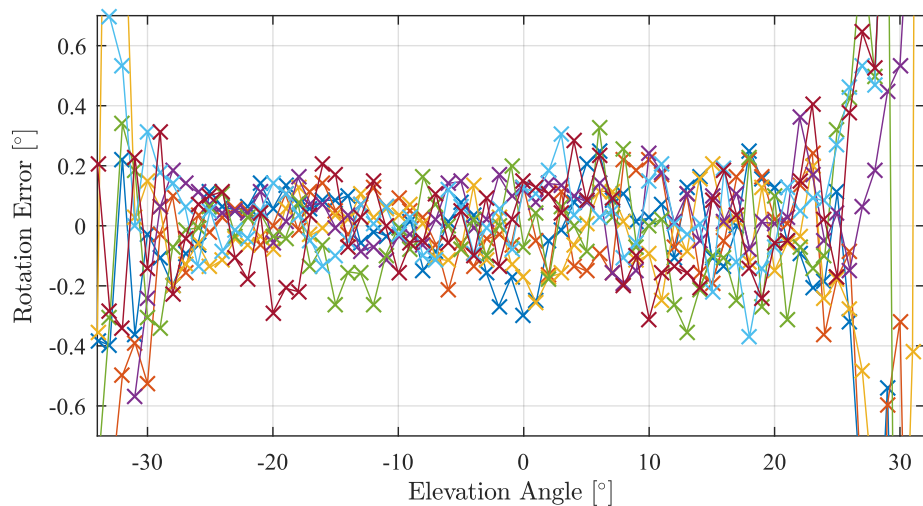
From the figures in Section 9.3.1 it is clear that measurements below -30° and above 26° cause erratic results. Therefore these elevation angles were chosen as the operational limits. The post-calibration results are shown in Figure 9.8 in Section 9.3.3 as a proof of concept using MATLAB. This post calibration process effectively removed the varying bias offsets from the measurements without attempting to remove the noise, and therefore this calibration process was successful.

9.3.3 Calibrated Results

The expected performance of the sensor in its operational range with the added Post Calibration is shown in Figure 9.8. It shows that within the sensor's operational range (between -30° and 26° elevation), the elevation and rotation angles are accurately measured to within approximately 0.1° or 0.4° respectively. In this operational range, there was no bias error as expected after the Post Calibration, whereas outside this range the accuracy dropped significantly.



(a) Measured elevation error over an elevation sweep



(b) Measured rotation error over an elevation sweep

Figure 9.8: Attitude estimation obtained through an emulated horizon and MATLAB after post calibration

It is clear that significant errors are still present in the signal. Additional to the inevitable system noise, these errors include an aliased version of the small sinusoidal-shaped effect present in simulated results (see Figure 6.22b). This effect will be discussed further in Section 9.4.2 where a larger data set is used to analyse the performance of the algorithms.

The operational range is influenced mainly by the number of edge coordinates acquired which severely limits the achievable accuracy. In Figure 9.9a it is clear that this number of coordinates decrease significantly below -30° and above 26° elevation and stays relatively constant between these two angles. This is reflected in the measured disc radius (shown in Figure 9.9b) which is a good indication of the measurement accuracy. The expected disc radius (calculated with the orbit height and camera FOV) and the measured disc radius should be approximately equal. In the figure, the measured disc radius is lower than expected which is due to the unfocussed lens and heated air changing the shape of the disc. However, it is clear that the measured disc radius is only uniform in the operational range, which confirms the selected operational limits. The two figures in Figure 9.9 are not symmetrical around $\phi = 0^\circ$ due to the observed disc not being vertically

symmetrical.

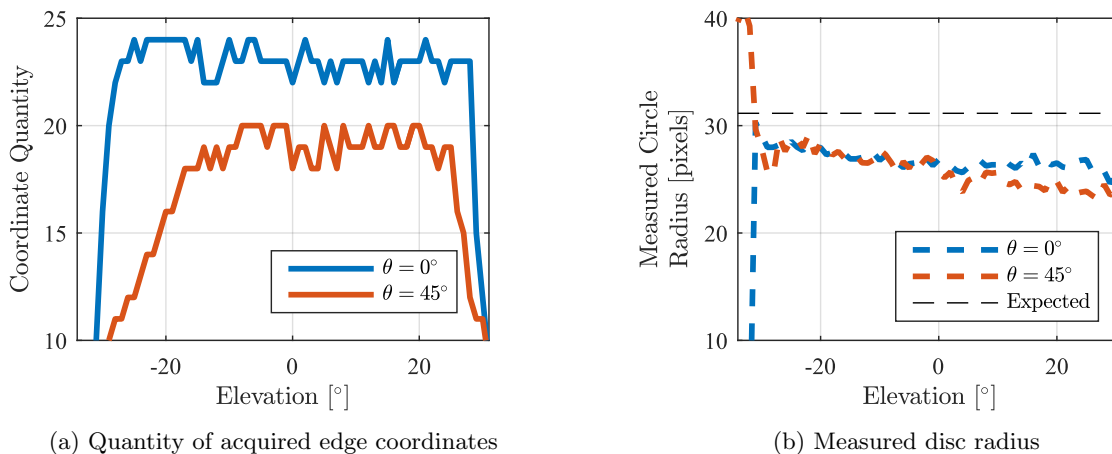


Figure 9.9: Sensor operational range's limiting factors over an elevation sweep

9.3.4 Robustness Against Noise

In the previous section (9.3.3) it was proved that the sensor has uniform behaviour over its operating range and successfully eliminates deterministic bias errors. However, the observed images also have a stochastic component which has to be investigated to ensure that the sensor can reliably measure the correct attitude. Therefore the sensor was set to measure a single attitude from multiple images to quantify its performance. To follow the trend in Section 6.6.2 this was done for a attitude of $\phi = \theta = 0^\circ$ and $\phi = \theta = 20^\circ$. The effect of averaging the images (before the feature extraction process) was also investigated, which effectively changed the measurement's SNR. This measurement's SNR was approximated by using the expected SNR shown in Table 3.2, in conjunction with Eq. 3.15, to model the effect of averaging. The result of these test are shown in Figure 9.10.

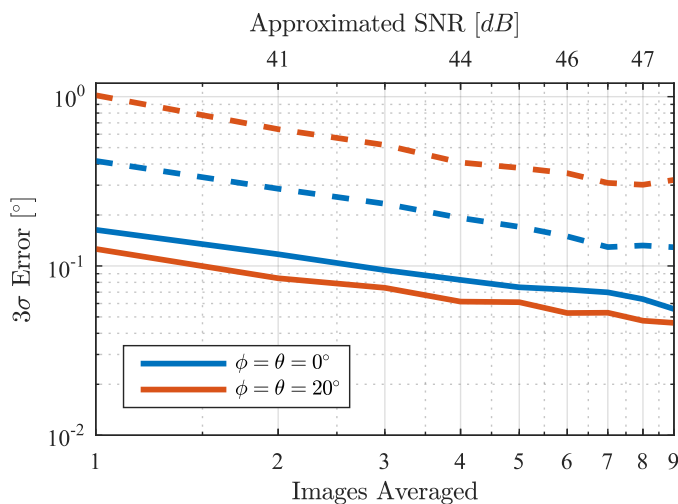


Figure 9.10: Variation in the measurement error for two possible attitudes. (Solid lines show the elevation error and dashed lines the rotation error)

Figure 9.10 shows that the design requirement of 0.1° accuracy is reachable if at least three images are averaged before the feature extraction process. This was easily reachable, as Section 8.3 showed that up five images could easily be averaged during the time limitation of 1 second, while still leaving ample room for the image processing. However, this accuracy was still an order of magnitude worse than expected through simulation (see Section 6.6.2). This was mainly due to the steel's temperature loss (see Figure 3.10) and other external influences such as heated air and unfocused images.

Table 9.2: Error statistics of post calibration results when averaging five images

	Worst Case Error	
	Elevation	Rotation
σ	0.025°	0.11°
3σ	0.075°	0.36°
Mean	0.074°	0.38°

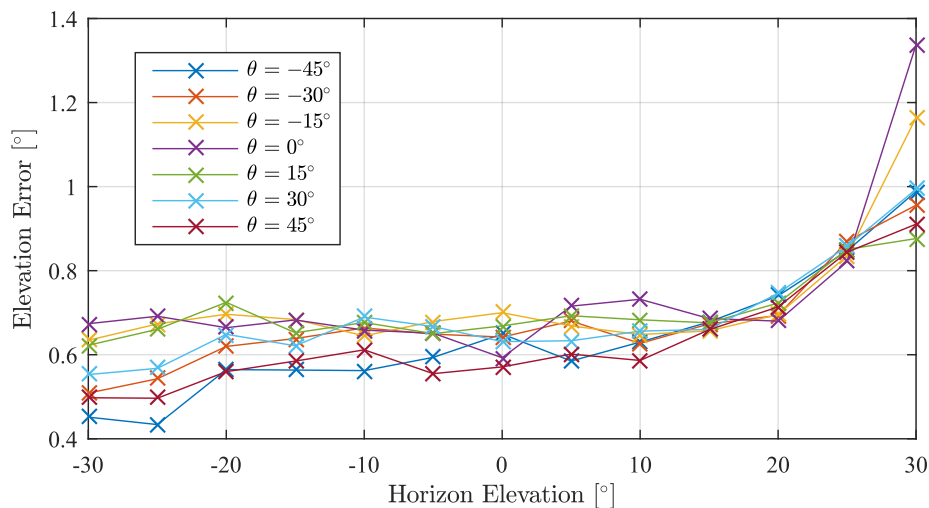
Table 9.2 shows the error statistics when averaging five images. This shows that a mean error (read bias error) is still present in the measurement, which means that the Post Calibration process was not entirely successful in eliminating the deterministic bias error. This large bias error is due to two factors: first, the small dataset used for the Post Calibration which did not sufficiently remove noise. Secondly, the slight oscillation in the elevation response during simulation (see Figure 6.22) is not accounted for. These influences will be investigated in further detail in Section 9.4. However, the results acquired through MATLAB proves that, with sufficient calibration, sufficient accuracies should be achievable by the MCU.

9.4 Full Onboard Results

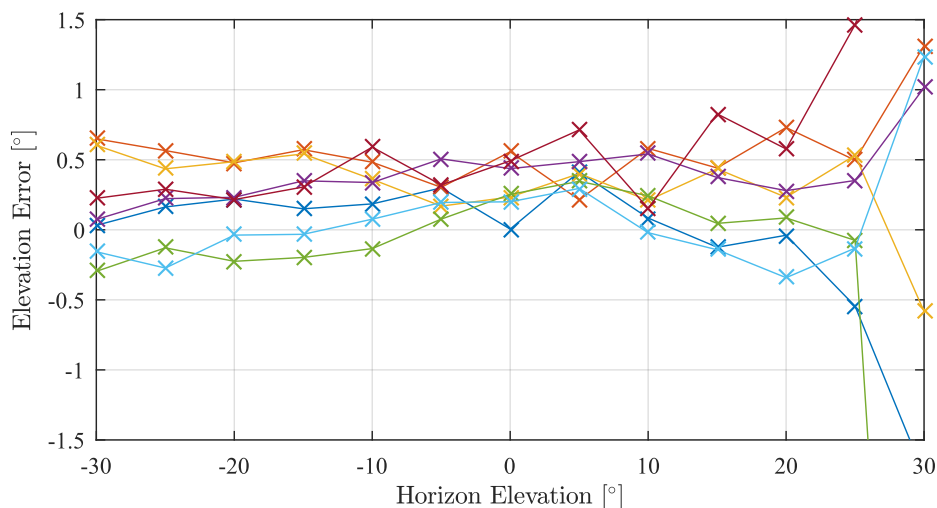
This section will investigate the performance of a stand-alone sensor, i.e. where all the processing is done on the sensor's onboard MCU while capturing data from the ground test setup discussed in Section 3.5. First, the raw performance of the sensor without the Post Calibration (Section 9.3.2) will be investigated. The sensor will then be recalibrated using its own captured data, and the final sensor performance evaluated.

9.4.1 Initial Results

The initial results without Post Calibration were acquired in similar ways as discussed in Section 9.3.1. In this test each acquired sample included the five images averaged on the MCU itself. Ten samples were then taken for each data-point to remove noise from the measurement effectively. This was done to increase the accuracy and effectiveness of the Post Calibration process. To accommodate the low throughput of the testing process, the elevation step was increased to 5° , as it was found the bias offsets are gradual enough to still be fitted sufficiently (see Figure 9.6). The results of these tests are shown in Figure 9.11.



(a) Measured elevation error over an elevation sweep

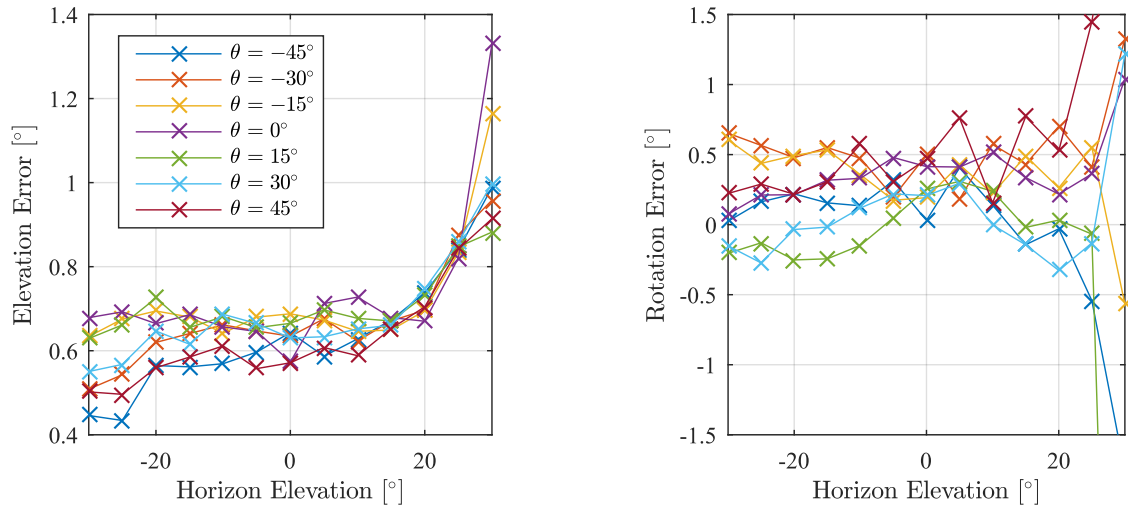


(b) Measured rotation error over an elevation sweep

Figure 9.11: Measurements obtained through the sensor's onboard MCU

This data returned by the sensor mimics the data captured in Figure 9.6 with operational range and shape, albeit with a larger bias error (roughly 0.6° , instead of 0.4°). This could be due to the camera mounting being shifted during the mounting (and demounting) of the sensor. Regardless, it is clear that a bias error exists on the measured data which will be calibrated in the next section, using the same process discussed in Section 9.3.2.

To ensure that the MCU's software works as designed, the images used for each measurement were also returned to the PC and in turn processed in MATLAB. These results are shown in Figure 9.12, and it is clear that the MCU's software is working as expected, as the results are identical.



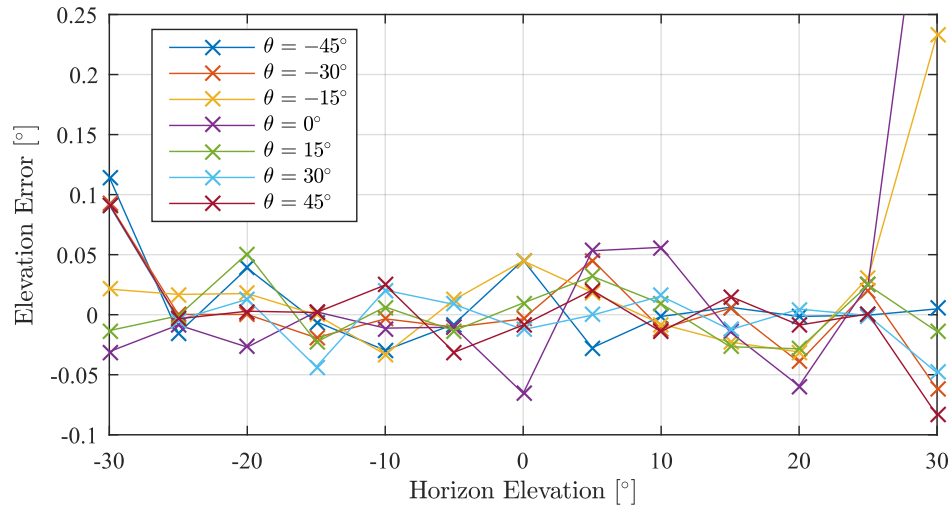
(a) Measured elevation error over an elevation sweep

(b) Measured rotation error over an elevation sweep

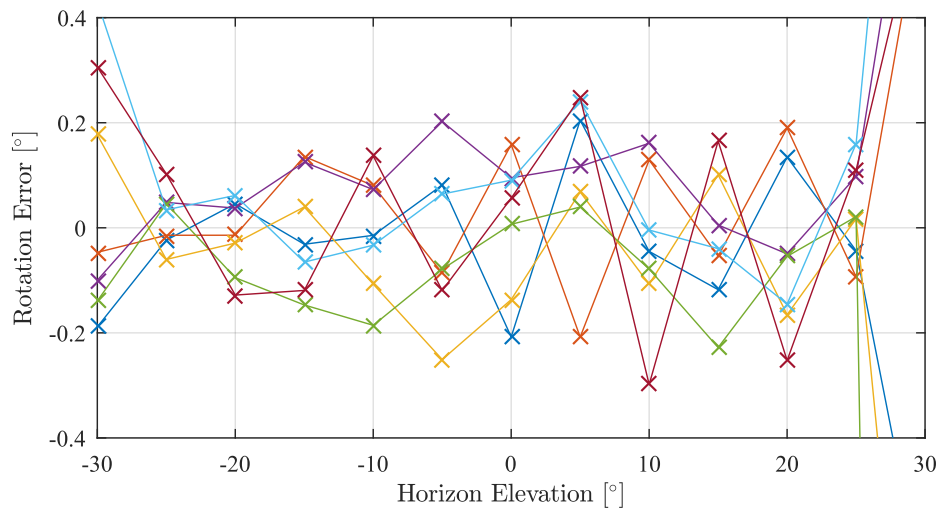
Figure 9.12: Measurement results obtained through processing the sensor's data in MATLAB

9.4.2 Calibrated Results

The data shown in the previous section was calibrated with the Post Calibration process to ensure a uniform response. As discussed in Section 9.3.2 this was a viable process, as the bias offset is deterministic and not stochastic. The calibrated results are shown in Figure 9.13, where an identical model was used with the MATLAB data.



(a) Measured elevation error over an elevation sweep



(b) Measured rotation error over an elevation sweep

Figure 9.13: Calibrated measurements obtained through the sensor's onboard MCU

Figure 9.13 shows that the bias errors are eliminated from the measurement, but a slight variation still exists in the sensor's response which is caused by a deterministic effect. This effect mimics white noise but should be smaller with the effective SNR of these measurements. Rather, this effect is an aliased version of the slight oscillations visible in the sensor's response due to Sobel Filter approximation (see Section 8.2.4). This is clearly visible during simulation (see Figure 6.22b in Section 6.6.1). To prove this, a fine step elevation sweep test was utilised on the ground test setup, with the results shown in Figure 9.14. This figure shows the mean measurement error of ten samples, additional to the maximum and minimum errors observed (dataset is too small to calculate the standard deviation), which shows the oscillation in the sensor's response.

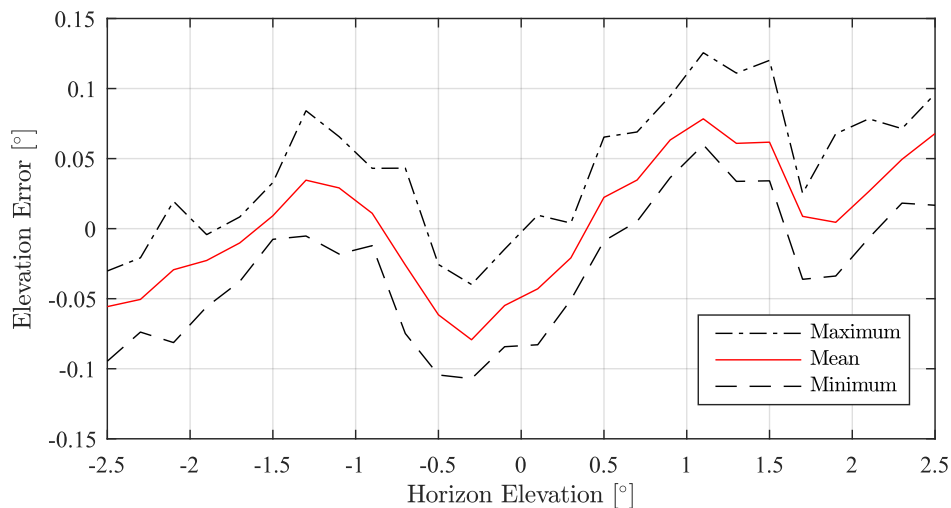


Figure 9.14: Proof of small oscillation on sensor's elevation response

This oscillating effect is an order of magnitude larger than in simulation (0.15° instead of 0.01°), which is due to the more gradual transition between *warm* and *cold* pixels. This effect is directly proportional to the slope width. This means that for a perfect Post Calibration, a very fine elevation sweep must be utilised to ensure that the mean values are obtained for each horizon elevation angle (and not a peak or trough). This is because the effect itself is not correctable, and will therefore still be present on the sensor's current configuration. This effect's phase is extremely sensitive to the horizon's location over the pixels, and consequently a sufficiently fine Post Calibration is not implementable for this relatively noisy sensor.

The simulated results show that this effect might be larger in space conditions due to the high contrast edge even though the visible horizon will be flatter (resulting in less high-incident angle measurements). However, this effect can be diminished by removing the Sobel Filter approximation or determining a gradient calculation to ensure the gradient calculation is more radially symmetrical.

9.4.3 Robustness Against Noise

The sensor's robustness towards noise is important, as noise is inevitable in any system. According to Section 3.3.2, the noise present on each pixel is roughly $\sigma = 0.64 \text{ mV}$ in magnitude which can impact the sensor's performance. Therefore, to follow the previously set convention, two tests are run, namely: one at ideal conditions with $\phi = \theta = 0^\circ$, and one at $\phi = \theta = 20^\circ$. Due to the sensor's limited functionality, the performance is only measured when averaging five images for each measurement. The measured results followed a clear Gaussian response, and are summarised in Table 9.3.

Table 9.3: Error statistics of sensor's the performance

	$\phi = \theta = 0^\circ$	
	Elevation	Rotation
σ	0.023°	0.0614°
3σ	0.069°	0.184°
Mean	-0.024°	-0.16°
	$\phi = \theta = 20^\circ$	
	Elevation	Rotation
σ	0.018°	0.136°
3σ	0.054°	0.409°
Mean	-0.005°	-0.389°

These statistics coincide with what was observed when processing the images with MATLAB in Table (9.2), except for the change in mean error. This shows that the accuracy did not decrease as the sensor's attitude changes, but rather that the calibration is not perfect. As discussed in Section 9.4.2 the calibration was not perfectly fitted due to the slight oscillation present, which explains the drift in bias error at larger attitude angles. However, the deviation in error magnitude is well within the design goals of this project and the mean error can easily be corrected with a finer calibration dataset.

9.4.4 Robustness Against Changing Environment

As this sensor should be fairly robust to a changing environment (see Section 3.2), the robustness was investigated by emulating scene temperature changes. This was done by simply switching off the hot plate, which heats the emulated Earth disc, while constantly attempting to measure the attitude. This emulated a change in Earth temperature, changed the edge profile, and in turn gave a good indication of the robustness. The measured elevation angles over time are shown in Figure 9.15, where the dashed line shows the switching-off of the hot plate at 15:11, and the black circle shows the first erratic measurement at 15:21.

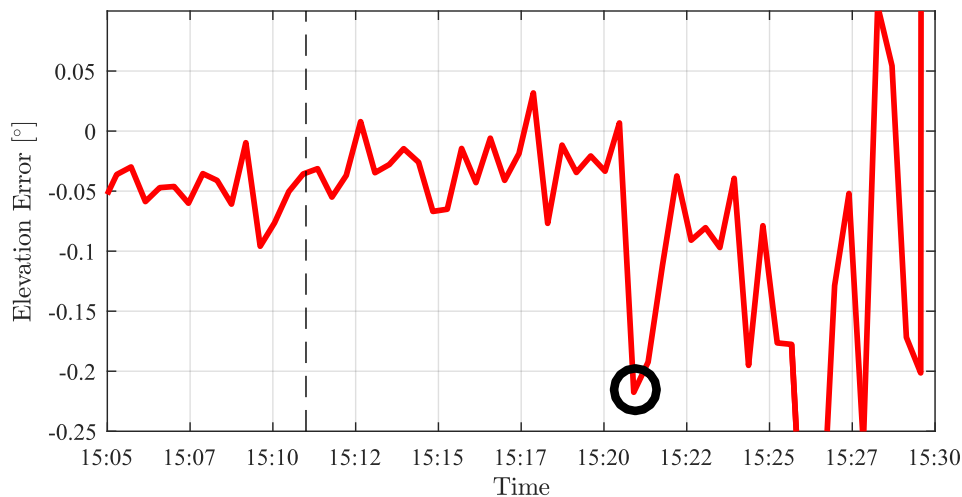


Figure 9.15: Elevation measurement error over time

The drop in signal strength, or the difference between the *warm* and *cold* pixels, is shown over time in Figure 9.16, with the same indications for the hot plate switching off and the first erratic measurement. This proves

that the steel temperature changed with over 40°C before the slope profile was altered enough for an erratic measurement. This shows extreme robustness towards a changing environment.

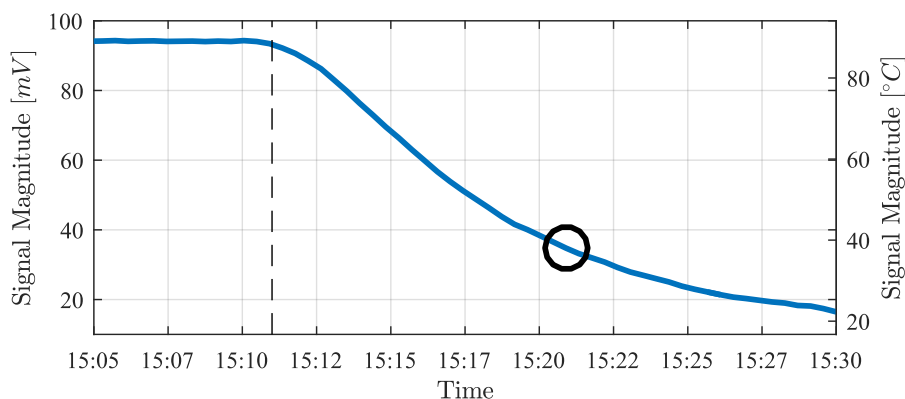


Figure 9.16: Signal magnitude error over time

For a simpler visualisation of the environment difference, two images are shown in Figure 9.17, where the left image is taken at 15:10, and the right at 15:21. It shows that the edge profile is significantly flatter, or has a decreased contrast.

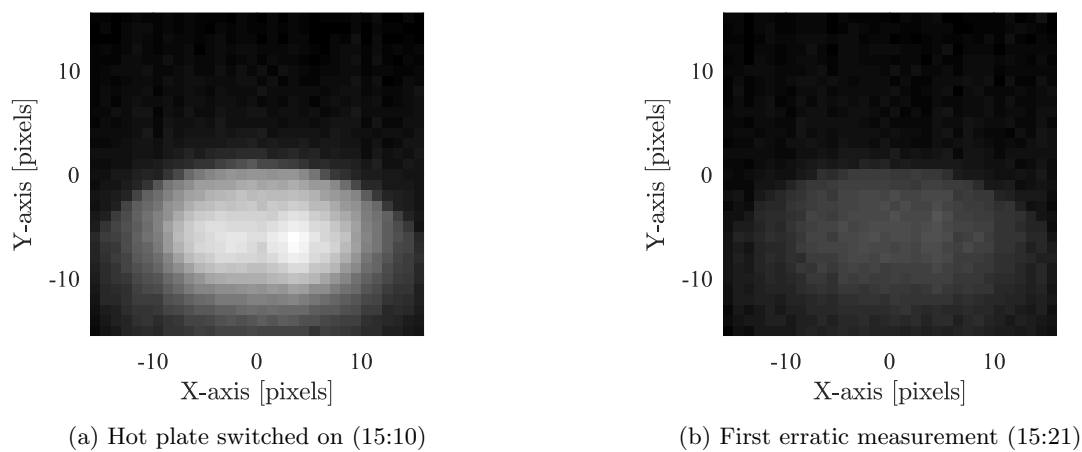


Figure 9.17: Example images to show the difference in environment over time (both images are scaled to (a)'s intensity range)

9.5 Discussion

This section showed that a worst-case elevation accuracy of up to 0.075° with added noise of $\sigma = 0.23^{\circ}$ is possible. Similarly, 0.39° with added noise of $\sigma = 0.14^{\circ}$ is achieved for the rotation angle accuracy. To achieve this accuracy, first, the ground test setup had to be calibrated for misalignments, which was done successfully (Section 9.2). Secondly, deterministic bias errors had to be removed by utilising a Least Squares method. This calibration was successful, but only to a certain extent (Section 9.3.2). For a perfect calibration, the dataset should be created with a fine elevation sweep to ensure that the slight oscillation in the elevation response is quantified correctly. With the current hardware, such a test would take weeks and therefore this

oscillation was ignored for the Post Calibration. It was found that that the accuracy would still be within 0.1° regardless, as seen in Figure 9.14. This means that the elevation accuracy can still be improved to roughly 0.07° (Table 9.3) if the bias errors are removed successfully through a finer calibration, or by removing the approximation causing the slight oscillation (Section 8.2.4). It was also shown that the entire process is very robust to a change in scene temperature, where constant measurements were made with the disc temperature varying by over $40^\circ C$ (Section 9.4.4).

The main limitation of these results was the low-resolution of the camera, as it directly affected the amplitude of the elevation oscillation and limits the robustness towards noise. In Section 10.2.4 the expected performance of doubling the resolution will briefly be discussed.

10 Conclusion & Recommendations

10.1 Conclusion

The development strategy of this sensor was successful. The pixel acquisition and calibration process can store images in the MCU's memory in real time for later processing, and ensure a uniform pixel response. The edge detection and scanning techniques accurately locate edge coordinates to within sub-pixel accuracies. These coordinates are corrected for the inevitable lens distortion to create a geometrically correct list of coordinates. A circle is then fitted on these coordinates to resemble the Earth disc from which the satellite attitude is determined. To ensure uniform behaviour across the sensor's entire operating range, these measurements are calibrated resulting in a very accurate reading. This is all executed on a small 8-bit processor within 1 second, while only using an average 92 *mW* of power.

This is possible because of the innovative detection strategies developed for this sensor. These algorithms, with an emphasis on the Local Extrema edge detection method (see Section 6.1.5), are extremely robust while retaining its accuracy. It requires no prior knowledge of the image characteristics and dynamically adapts to different edge profiles. Additionally, these algorithms were significantly optimised to require fewer resources, creating fast execution times. This leaves more available processing time to increase the image SNR through averaging, as well as implementing post calibration to improve the final measurement.

However, the main limitation of the developed sensor is the camera's low resolution. The low resolution is the main cause of a slight oscillation in the sensor's elevation response (see Figures 6.22a and 9.14) severely limiting the possible accuracy. The variation in elevation measurement due to noise is only about $\sigma = 0.023^\circ$, while the bias error due to low resolution alone is as large as 0.075° in amplitude. Although steps were taken to maximise the accuracy achievable by this sensor, the limitations caused by the low resolution can only be circumvented to a certain extent. Without a significant increase in processing power and horizon detection strategy complexity, a more accurate measurement is improbable.

In conclusion, this study was a success. An operational infrared horizon sensor was developed that can measure its altitude within the timespan of 1 second. It can determine its elevation and rotation angles with a worst case accuracy of up to 0.14° and 0.4° respectively (with added noise of $\sigma = 0.023^\circ$ and $\sigma = 0.14^\circ$), within. This sensor is low power, small volume and implementable on a CubeSat, satisfying all the criteria for this study.

10.2 Recommendations and Future Work

10.2.1 Better Ground Test Setup

A large limitation in the evaluation of the developed sensor was the testing setup. It had various flaws, all of which could be improved in further study. This included:

- **Uniform disc edge:** The disc edge created in the used setup had a non-uniform temperature distribution. Due to the localized heating by the steel plate, the middle of the steel was warmer than the left and right edges. Additionally the air around the steel plate was heated, which was also picked up by the infrared camera. These two effects created a non-uniform response in the sensor which will

not be present in a space application. The steel plate can be heated uniformly by custom shaping a heating element to fill the back of the plate. It was attempted to mitigate the effect of the heated air by utilising a desk fan, but a more direct solution can be applied by sucking the warm air to the bottom of the steel plate with more powerful fans. This will ideally result in all the warm air being pulled down, instead of moving up and warping the edge of the disc. Alternatively the disc could be mounted upside down, and the natural draft of warm air would mitigate the heated air effect.

- **Actuated rotation stages:** Another severe limitation of calibration techniques and sensor evaluation was the lack of rotation stage actuation, as each rotation had to be made by hand. This was extremely tedious, and limited resolution of the angles to be imaged, and in turn limits the observation of high-frequency artefacts (see Figure 9.14). Future research should ideally incorporate actuated rotation stages to aid the development of such a sensor.

10.2.2 Moon & Sun Rejection

During orbit, it is inevitable that the sensor will occasionally have either the moon or sun in view. Although this was not discussed in this study, it should be accounted for as it will influence results. If these celestial bodies are in view, the edge detection algorithm has a chance of erroneously locating an edge of this body. In this event, it will only affect the shape of the fitted circle (or Earth disc). It will be clear that the fitted disc's radius is much larger (or smaller) than expected, which will show that the measurement can be ignored. If handled correctly, it is impossible for it to cause any error conditions in the software.

It is assumed that the sun will oversaturate the infrared camera which renders an accurate measurement impossible. Alternatively, an accurate measurement can be attempted while the moon is in view. It is possible to obtain an accurate measurement while the moon is in view above the horizon, and not overlapping the horizon. The scanning pattern can then be adapted to only scan upwards from the bottom of the image (the *warm* Earth) until it detects the first sufficient edge. This will result in the edge detection technique never reaching the moon, and therefore it will not influence the measurement. If overlapping occurs, this method could still be used if the moon's thermal radiance is significantly less than the Earth's. If the moon's radiance is roughly equal to the Earth's, then the moon's edge could be erroneously detected which will produce an unusable measurement (can be ignored using the fitted disc's radius).

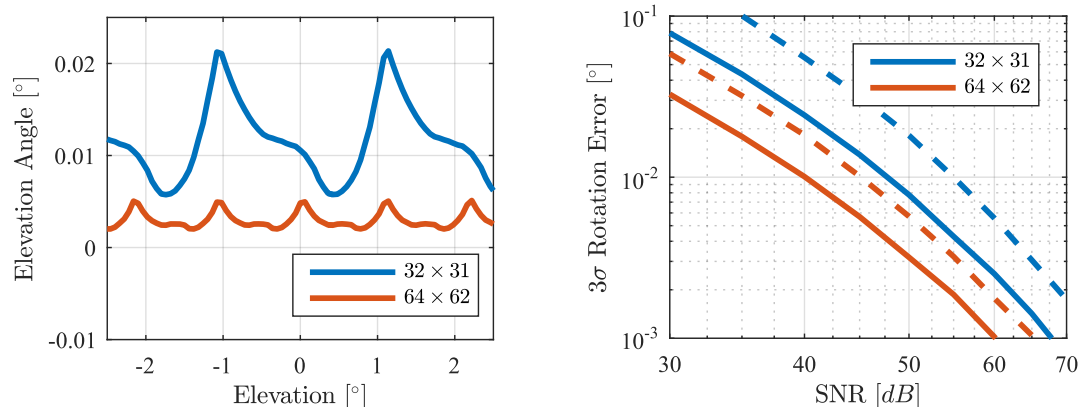
10.2.3 utilise FRAM for Image Storing

The FRAM discussed in Section 4.2.3 was not utilised in the final iteration of the developed sensor as the MCU's internal RAM storage was sufficient, although the FRAM was in full working condition. However, utilising this FRAM for primary image storage in a future implementation would be beneficial, as it is very robust against radiation and would, therefore, be more reliable in space conditions. The FRAM's read/write speed through the PMP-module (see Section 4.3.4) is roughly $4.5 \mu\text{s}$ per byte, which is on par with the read/write speed of the internal RAM of $4.75 \mu\text{s}$ per byte. Therefore, with only a slight increase in software complexity, it is possible to significantly increase the sensor's robustness against radiation by utilising the FRAM.

10.2.4 Increase Camera Resolution

As discussed previously, the main limitation of the sensor is the low resolution of 32×31 pixels. As shown in Table 4.1, there are alternative cameras commercially available with higher resolutions of up to 80×64 pixels. Utilising these higher resolutions will significantly increase the accuracy of this sensor, as well as effectively removing the oscillation in the elevation response. A simulation was run to visualise the effect of doubling

the resolution (with $4\times$ the pixels). The elevation oscillation and robustness towards noise on the system are shown in Figure 10.1.



(a) Reduction in elevation oscillation with increase of resolution

(b) Increased robustness towards noise with increased resolution for the elevation (solid) and rotation error (dashed)

Figure 10.1: Performance compared to change in camera resolution

It is clear that the performance of the sensor increases significantly. The simulated elevation oscillation amplitude decreases by 80%. Assuming an SNR of 50 dB (achieved during ground testing), the elevation error standard deviation decreases by 60%. Although the actual accuracies will be worse in a real implementation, it can be expected that the ratio of increase in accuracy relative to resolution will remain consistent. Following this assumption, it can be assumed that with a 64×62 resolution camera, the current sensor implementation can achieve an absolute elevation accuracy of $< 0.045^\circ$.

Appendices

A Sobel Operator

The Sobel operator, or alternatively the Sobel-Feldman operator, is a computationally efficient, gradient operator developed in 1968 by Irwin Sobel and Gary Feldman[48]. It was developed to achieve an estimated gradient efficiently which is more isotropic than the then popular Roberts Cross Operator[49], and optimized for use in integer arithmetic. It was designed to determine the gradient in a digitized image at a single point, taking four different gradient directions in a 3x3 pixel neighbourhood into account.

Assume point e on a Cartesian grid and its eight neighbours as shown below.

a	b	c
d	e	f
g	h	i

The gradients between different pixel pairs can be expressed as:

$$|\Delta| = \frac{\text{Intensity Difference}}{\text{Distance to Neighbour}} \quad (\text{A.1})$$

The gradient vector Δ can then be estimated using Eq. A.2. Note the gradient between the corner pixel pairs is calculated using an approximated distance of four pixels, instead of the actual distance of $\sqrt{2^2 + 2^2} = 2.83$ pixels. It will simplify equations further down, and is acceptable as only an estimation of the gradient is required.

$$\begin{aligned} \Delta &= (c - g)/4 \times [1, 1] \\ &+ (a - i)/4 \times [-1, 1] \\ &+ (b - h)/2 \times [0, 1] \\ &+ (f - d)/2 \times [1, 0] \end{aligned} \quad (\text{A.2})$$

This can be simplified to:

$$\Delta = [(c + i - a - g)/4 + (f - d)/2, (a + c - i - g)/4 + (b - h)/2] \quad (\text{A.3})$$

Notice that the final result Δ should be divided by four to determine the average gradient to be metrically correct. Since this operator is designed for integer arithmetic, and is only an estimate of the gradient, the vector Δ is rather multiplied by four. This results in a gradient estimate that is 16 times as large as the average gradient, but with no divisions and rather multiplications that can be done using a left bit-shift to preserve low order bits. The gradient estimate can therefore be expressed as:

$$\Delta' = [(c + i - a - g) + 2(f - d), (a + c - i - g) + 2(b - h)] \quad (\text{A.4})$$

This can then be expressed as weighted intensity summations by the following weighting functions in the x and y directions:

-1	0	1
-2	0	2
-1	0	1

x-component

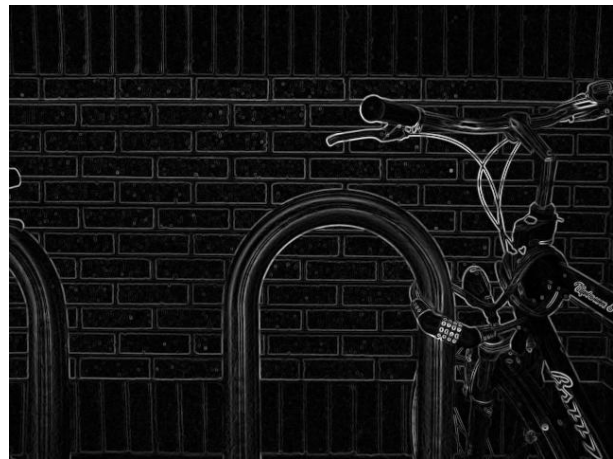
1	2	1
0	0	0
-1	-2	-1

y-component

The absolute gradient of any pixel can therefore be estimated by calculating the gradient vector using the two weighting functions, and then determining the absolute length of the calculated vector. An example of such an image is shown in Figure A.1.



(a) Grayscale image of brick wall and bike rack



(b) Normalized gradient image using Sobel Operator

Figure A.1: Example of Sobel Operator being applied to a grayscale image

B Telecommands and Telemetry Requests

B.1 Telecommands

ID	Command	Length	COffset	CLength	CDescription	Type
1	Turns on the camera	0				
2	Turns off the camera	0				
3	Turns on LED 2	0				
4	Turns off LED 2	0				
5	Write a byte into the FRAM	3	0	16	address	unsigned int
6	Reads a byte from the FRAM (use TLM to access the read byte)	2	16	8	data	unsigned char
9	Implements a soft reset	0	0	16	address	unsigned int
10	Initiates image capture	0				
11	Sets the camera amplification as high	0				
12	Sets the camera amplification as low	0				
13	Writes 0 to all FRAM locations	0				
14	Send 1 if the MCU should do automatic post processing	1	0	8	enable	unsigned char
16	Returns edge coordinates	0				
17	Runs the Sobel Edge algorithm	0				
18	Runs the Edge Detection Algorithm	0				
19	Streams the pixel values through the UART	0				
20	Runs the circlefitting algorithm on the processor	0				
21	Runs the lense distortion correction algorithm	0				
22	Estimates the attitude	0				
23	Enables/Disables post calibration	1	0	8	enable	unsigned char
24	Sets the measured rotation for Post Calib (in degrees)	1	0	8	rotation	signed char
25	Enables/Disables rig calibration	1	0	8	enable	unsigned char

B.2 Telemetry Requests

ID	Command Description	Length	Coffset	Clength	Channel Description	Channel Type
128	Device Information	4	0	8	Device ID	unsigned char
			8	8	Firmware Version	unsigned char
129	Potential expansion space	1	16	16	Runtime (Seconds)	unsigned int
			0	0	Anything	unsigned char
130	All set device error flags	1	0	1	UART Error	bit
			1	1	Communications Error	bit
131	All errors flags set by uart	1	0	1	Framing Error	bit
			1	1	Overrun Error	bit
			2	1	Parity Error	bit
			3	1	SC not sent	bit
			4	1	SOM not sent	bit
			5	1	Invalid command	bit
			6	1	Invalid Data	bit
			7	1	TX buffer overflow	bit
132	All error flags set by comms	1	0	1	TLC buffer overflow	bit
			1	1	Not Used	bit
			2	1	Not Used	bit
			3	1	Not Used	bit
			4	1	Not Used	bit
			5	1	Not Used	bit
			6	1	Not Used	bit
			7	1	Not Used	bit
133	LED Status	1	0	1	LED 2 Turned on	bit
			1	1	LED 1 Turned on	bit
134	Returns the last byte the FRAM read	1	0	8	The byte	unsigned char
135	Returns the last value measured by tick-tock timer	3	0	24	Value (1 = 250ns)	unsigned short long
136	Returns the last PTAT value from camera	2	0	16	PTAT	unsigned int
137	Returns the number of images to average for each sample	1	0	8	Number to average	unsigned char
138	Returns the current pulled from the 5V and 3.3V regulators	4	0	16	Current 5V	unsigned int
			16	16	Current 3.3V	unsigned int
139	Returns the fitted circle to the detected points	12	0	32	Circle Center x	float
			32	32	Circle Center y	float
			64	32	Circle radius	float
140	Returns the sensor's attitude	8	0	32	Elevation	float
			32	32	Roll	float
142	Return rotation (override of CoI) in degrees	8	8	8	Rotation	signed char

References

- [1] M. Phenneger, S. Singhal, T. Lee, and T. Stengle, “Infrared horizon sensor modeling for attitude determination and control: analysis and mission experience,” 1985.
- [2] S. Janson, B. Hardy, A. Chin, D. Rumsey, D. Ehrlich, and D. Hinkley, “Attitude control on the pico satellite solar cell testbed-2,” in *26th Annual AIAA/USU Conference on Small Satellites*, 2012.
- [3] Mai-ses. Maryland Aerospace. [Online]. Available: <http://maiaero.com/datasheets/MAI-SES-Specifications-20150827.pdf>
- [4] T. Nguyen, “Attitude determination using infrared earth horizon sensors,” in *26th Annual AIAA/USU Conference on Small Satellites*, 2014.
- [5] (2007) Eumetsat image gallery. EUMETSAT. [Online]. Available: www.eumetsat.com
- [6] H. Sensor, “Htpa32x31110/1.0his,” 2013.
- [7] Css-01,02 coarse sun sensors. Space Micro. [Online]. Available: <http://www.spacemicro.com/assets/datasheets/guidance-and-nav/CSS.pdf>
- [8] Magnetometer. NewSpace Systems. [Online]. Available: <http://www.newspacesystems.com/wp-content/uploads/2016/05/NewSpace-DS-Magnetometer-1.pdf>
- [9] Cubesat sun sensor. NewSpace Systems. [Online]. Available: <http://www.newspacesystems.com/wp-content/uploads/2016/05/NewSpace-DS-Cubesat-Sun-Sensor-1.pdf>
- [10] Cubesense. CubeSpace. [Online]. Available: <https://www.cubespace.co.za/componentscubesense>
- [11] Mai-ss. Maryland Aerospace. [Online]. Available: <http://maiaero.com/datasheets/MAI-SS%20Space%20Sextant%20Datashet.pdf>
- [12] Htpa80x64d infrared thermopile array sensor. Heimann Sensor. [Online]. Available: http://www.heimansensor.com/Datasheets/Overview-HTPA80x64d_Rev8.pdf
- [13] (2016, March) Flir lepton long wave infrared (lwir) datasheet. FLIR. [Online]. Available: <http://www.flir.com/uploadedFiles/OEM/Products/LWIR-Cameras/Lepton/Lepton%20Engineering%20Datashet%20-%20without%20Radiometry.pdf>
- [14] Thermopile arrays enable new generation of automation systems. Panasonic. [Online]. Available: http://www.mouser.com/pdfdocs/panasonic_grid-eye_wp_v8.pdf
- [15] Far infrared thermal sensor array (32x24 res). Melexis. [Online]. Available: <https://www.melexis.com/en/product/MLX90640/Far-Infrared-Thermal-Sensor-Array>
- [16] Soho (solar and heliospheric observatory). Sharing Earth Observation Resources. [Online]. Available: <https://directory.eoportal.org/web/eoportal/satellite-missions/s/soho>
- [17] Rosetta. European Space Agency. [Online]. Available: <http://rosetta.esa.int/>
- [18] About the hubble space telescope. National Aeronautics and Space Administration. [Online]. Available: https://www.nasa.gov/mission_pages/hubble/story/index.html
- [19] (2015, May) Tiny ‘cubesats’ gaining bigger role in space. Space.com. [Online]. Available: <https://www.space.com/29464-cubesats-space-science-missions.html>

- [20] A. Wicks and C. Underwood, "A Novel Two-Axis Earth Horizon Sensor," in *Spacecraft Guidance, Navigation and Control Systems*, ser. ESA Special Publication, K. Fletcher and R. A. Harris, Eds., vol. 516, Feb. 2003, p. 65.
- [21] J. R. Wertz, *Spacecraft attitude determination and control*. Springer Science & Business Media, 2012, vol. 73.
- [22] A. Graf, M. Arndt, M. Sauer, and G. Gerlach, "Review of micromachined thermopiles for infrared detection," *Measurement Science and Technology*, vol. 18, no. 7, p. R59, 2007.
- [23] J. A. Hashmall, J. Sedlak, D. Andrews, and R. Luquette, "Empirical correction for earth sensor horizon radiance variation," 1998.
- [24] J. R. Wertz, D. F. Everet, and J. J. Puschell, *Space Mission Engineering: The New SMAD*, 2011.
- [25] J. Verheijen. (2016, July) World geodetic system 1984 (wgs84). Hydrographic and Marine Software Solutions. [Online]. Available: <https://confluence.qps.nl/pages/viewpage.action?pageId=29855173>
- [26] Hamamatsu, *Thermal Detectors*, ch. 7. [Online]. Available: https://www.hamamatsu.com/resources/pdf/ssd/e07_handbook.Thermal_detectors.pdf
- [27] Modtran. [Online]. Available: <http://modtran.spectral.com/>
- [28] Stk eoir atmosphere model. [Online]. Available: http://help.agi.com/stk/index.htm#eoir/eoir_atmosphere.htm
- [29] (2008, June) Cosmic background explorer. [Online]. Available: <https://lambda.gsfc.nasa.gov/product/cobe/>
- [30] M. Loeve, *Probability Theory*, 4th ed. Springer-Verslag, 1977, vol. 45.
- [31] R. Sayyah, T. C. Macleod, and F. D. Ho, "Radiation-hardened electronics and ferroelectric memory for space flight systems," *Ferroelectrics*, vol. 413, no. 1, pp. 170–175, jan 2011. [Online]. Available: <https://doi.org/10.1080/00150193.2011.554145>
- [32] Pic18f47j13. Microchip. [Online]. Available: <http://www.microchip.com/wwwproducts/en/PIC18F47J13>
- [33] S. Gerardin and A. Paccagnella, "Present and future non-volatile memories for space," vol. 57, no. 6, December 2010.
- [34] (2015, August) Fm28v020. Cypress. [Online]. Available: <http://www.cypress.com/file/136591/download>
- [35] Second order filters. Electronics Tutorials. [Online]. Available: <http://www.electronics-tutorials.ws/filter/second-order-filters.html>
- [36] H. Sensor, "Application note," October 2011.
- [37] M. S. Alam *et al.*, "Infrared image registration and high-resolution reconstruction using multiple translationally shifted aliased video frames," vol. 49, no. 5, 2000.
- [38] H. M. Van Rensburg, "An infrared earth horizon sensor for a leo satellite," 2008.
- [39] D. G. Bailey, "Sub-pixel estimation of local extrema," in *Proceeding of Image and Vision Computing New Zealand*, 2003, pp. 414–419.
- [40] J. F. Kenney and E. S. Keeping, *Mathematics of Statistics*, 3rd ed., 1962.
- [41] S. M. Robinson, "A short proof of Cramer's rule," *Mathematics Magazine*, vol. 43, no. 2, pp. 94–95, 1970. [Online]. Available: <http://www.jstor.org/stable/2688979>

-
- [42] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004.
- [43] J. Wang and F. Shi, “A new calibration model of camera lens distortion,” 2006.
- [44] Z. Zhang, “A flexible new technique for camera calibration,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000.
- [45] (2016, August) Cubesense. [Online]. Available: <http://41.185.8.177/~cubespac/ClientDownloads/CubeSense/CubeSense%20User%20Manual%20V1.6.pdf>
- [46] C. Lomont, “Fast inverse square root,” 2 2003.
- [47] “Infrared sensing solutions,” April 2015. [Online]. Available: http://www.excelitas.com/Downloads/CAT_SensorsAndEmittersInfraredSensing.pdf
- [48] I. Sobel, “History and definition of the so-called ”sobel operator”, more appropriately named the sobel-feldman operator.”
- [49] L. G. Roberts, “Machine perception of three-dimensional solids,” pp. 159–197, 1965.