Design and implementation of resilient attitude estimation algorithms for aerospace applications

XIANLIANG CHEN

M.Eng

THE UNIVERSITY OF SYDNEY

Supervisor: Xiaofeng Wu
Associate Supervisor: KC Wong

A thesis submitted in fulfilment of
the requirements for the degree of
Doctor of Philosophy

School of Aerospace, Mechanical and Mechatronic Engineering
Faculty of Engineering
The University of Sydney
Australia

2 July 2023

Declaration

I declare that this thesis is my original work and has not been submitted for any other academic degree or diploma at any institution. This work represents my own research and is based on my personal findings, analyses, and interpretations.

I confirm that all the sources used in this thesis have been properly acknowledged and cited according to the guidelines set forth by my academic institution. I have taken due care to ensure that all the information presented in this thesis is accurate, reliable, and unbiased. Any errors or omissions are unintentional and solely my responsibility.

Furthermore, I declare that I have complied with all ethical standards, including obtaining necessary permissions and consent from research participants, and that my research has been conducted in accordance with the relevant codes of conduct and guidelines.

In conclusion, I affirm that this thesis is a true representation of my academic achievements and research capabilities. I hope that it will contribute to the advancement of knowledge in the field and inspire future research endeavors.

Name: Xianliang Chen

Authorship attribution statement

Chapter 4 of this thesis is published in Xianliang Chen et al. (2023a). 'An Observation Model From Linear Interpolation for Quaternion-Based Attitude Estimation'. In: *IEEE Transactions on Instrumentation and Measurement* 72, pp. 1–12. I designed the research, conducted the experiments, analyzed the data, and wrote this paper.

Chapter 5 of this thesis is submitted to the Xianliang Chen et al. (2023b). 'Kalman filter and neural network fusion for fault detection and recovery in satellite attitude estimation'. Manuscript submitted for publication and is under review condition. In addition, the QUEST and RBF part of Chapter 5 is published in Xianliang Chen et al. (2022). 'Feasibility Study of Neural Network in Satellite Attitude Determination'. In: *6th International Technical Conference on Advances in Computing, Control and Industrial Engineering (CCIE 2021)*. Springer, pp. 264–271. In these 2 papers, I designed the research, conducted the experiments, analyzed the data, and wrote the paper.

Chapter 6 of this thesis is finished and prepared to submit to the journal "IEEE Sensor Journal". I designed the research, conducted the experiments, analyzed the data, and wrote the paper.

Chapter 7 of this thesis is prepared to be submitted to the Xianliang Chen et al. (2023c). 'Optimized FPGA Implementation of Fault Detection, Isolation and Recovery System for Satellite Attitude Estimation'. Manuscript submitted for publication. I designed the research, conducted the experiments, analyzed the data, and wrote this paper.

In addition to the statements above, in cases where I am not the corresponding author of a published item, permission to include the published material has been granted by the corresponding author.

Student Name: XIANLIANG CHEN

Signature:

Date: 15 March 2023

As supervisor for the candidature upon which this thesis is based, I can confirm that the authorship attribution statements above are correct.

Supervisor Name: Xiaofeng Wu

Signature:

Date:

Abstract

Satellite attitude estimation is a critical component of satellite attitude determination and control systems, relying on highly accurate sensors such as Inertial Measurement Units (IMUs), star trackers, and sun sensors. However, the complex space environment can cause sensor performance degradation or even failure. To address this issue, fault detection, isolation, and recovery (FDIR) systems are necessary.

This thesis presents a novel approach to satellite attitude estimation that utilizes an Inertial Navigation System (INS) to achieve high accuracy with the low computational load. The algorithm is based on a two-layer Kalman filter, which incorporates the quaternion estimator (QUEST) algorithm, Factored Quaternion Algorithm (FQA), Linear interpolation (LERP) algorithms, and kalman filter (KF).

Moreover, the thesis proposes an FDIR system for the INS that can detect and isolate faults and recover the system to a safe state. This system includes two-layer fault detection with isolation and two-layered recovery, which utilizes an Adaptive Unscented Kalman Filter (AUKF), QUEST algorithm, residual generators, Radial Basis Function (RBF) neural networks, and an adaptive complementary filter (ACF). These two fault detection layers aim to isolate and identify faults while decreasing the rate of false alarms. An FPGA-based FDIR system is also designed and implemented to reduce latency while maintaining normal resource consumption in this thesis.

Finally, a Fault Tolerance Federated Kalman Filter (FTFKF) is proposed to fuse the output from INS and the Celestial Navigation System (CNS) to achieve high precision and robust attitude estimation.

The findings of this study provide a solid foundation for the development of FDIR systems for various applications such as robotics, autonomous vehicles, and unmanned aerial vehicles, particularly for satellite attitude estimation. The proposed INS-based approach with the FDIR system has demonstrated high accuracy, fault tolerance, and low computational load, making it a promising solution for satellite attitude estimation in harsh space environments.

Acknowledgements

I would like to express my deepest appreciation to my supervisor Dr. Xiaofeng Wu, for his invaluable advice, continuous support, and patience during my PhD study. I am also grateful to my colleague and friends, Mr. Zhicheng Xie, Dr. Zihao Wang for their inspiration. I would like to thank all the members of our lab. Their help and support made my study and life at the University of Sydney a wonderful journey. Finally, I would like to express my gratitude to my parents. It would be impossible for me to complete my study without their support and encouragement.

Contents

List of Figures

List of Tables

Abbreviations & Non-standard Terms

ADCS Attitude, Determination, and Control System

INS Inertial Navigation System

CNS Celestial Navigation System

GNSS Global navigation satellite systems

IMU inertial measurement unit

FDIR Fault, Isolation, and recovery system

FPGAs Field-Programmable Gate Arrays

LERP Linear interpolation

ACF daptive complementary filter

AUKF Adaptive Unscented Kalman filter

PL Programmable Logic

PS Processing System

AHRS Attitude and Heading Reference System

MARG Magnetic, Angular Rate, and Gravity

TRIAD Three-axis attitude determination

QUEST The quaternion estimator algorithm

FQA The factored quaternion algorithm

KF Kalman Filter

Q-AKF Quaternion-based Adaptive Kalman Filter

AGSD Adaptive-step Gradient Descent algorithm

O2OQ Two-observation quaternion estimation method

TGIC The two-step geometrically intuitive correction

AQUA Algebraic Quaternion Algorithm

APST Arcsecond Pico Star Tracker

SPIT Star-Pair Identification Technique

SMIT The Reference-Star Star-Matching Identification Technique

FOV Field-of-view

LAMBDA Least-squares Ambiguity Decorrelation Adjustmen

LSAST Least-Squares Ambiguity Search Technique

EKF Extend Kalman filter-based

UKF Unscented Kalman filter

DD Double-difference

NHC Non-holonomic constraint

AFKF Adaptive fading Kalman filter

CAFCKF A chi-square test-based adaptive federated cubature Kalman filter

MFKF Modified federated Kalman filter

HCV hypersonic cruise vehicle

ADS Attitude Determination System

FKF Federated Kalman Filters

FTFKF	Fault Tolerance FKF
MUKF	modified unscented Kalman Filter
PE	Parity Equation
FF	formation fly
SST	Satellite-to-Satellite tracking
RN	relative navigation
DOF	degrees-of-freedom
SPN	Spacecraft Pose Network
PAL	Programmable Array Logic
GAL	Generic Array Logic
CPLD	omplex Programmable Logic Device
CLB	Configurable Logic Block
CNN	Convolutional Neural Network
IoT	Internet of Things
CDFG	Control and Data Flow Graph
GNC	Guidance, Navigation, and Control
UAVs	Unmanned aerial vehicle
FSM	Finite-State Machine
DNN	Deep Learning Neural Networks
ASIC	Application-Specific Integrated Circuit
RN	relative navigatio
IAE	nnovation-based adaptive estimatio
DCM	The Direction Cosine Matrix
CPLD	Complex Programmable Logic Device
CPU	Central Processing Unit
DDR	Double Data Rate
DMA	Direct Memory Aceess
DSP	Digital Signal Processing
EDA	Electronic Design Automation
SVM	Support Vector Machines
SURF	Speeded-Up Robust Feature
OTG	USB On-The-Go port
FA	Full Adder
FF	Flip-Flop
FPGA	Field-Programmable Gate Array
FPS	Frames Per Second
API	Application Programming Interface
GP	General-Purpose
HDL	Hardware Description Languages
AXI	Advanced extensible Interface
HLS	High-level Synthesis
LSTM	Long short-term memory
HW/SW	Hardware/Software
I/O	Input/Output
IC	Integrated Circuit
IOB	Input Output Block
IP	Intellectual Property
Bp	Backpropagation Neural Networks
QNN	Quantum Neural Networks
LUT	Look-Up Table
MUX	Multiplexers
GAN	Generative adversarial networ
NN	Neural Network
PAL	Programmable Array Logic
GPIO	General Purpose Input/Outpu
MIMO	Memory Mapped IO
ECI	Earth Centered Interial
RBF	Radial Basis Function

RAM Random Access Memory

ECEF Earth-centered Earth-fixed reference frame,

NED North East Down

RTL Register-Transfer-Level

SDR Software Defined-Radio

BRAM Block Random Access Memory

SIFT Scale-Invariant Feature Transform

SLAM Simultaneous Localisation And Mapping

SoC System-on-Chip

MRP Modified Rodriguez Parameter

CCD Conversion circuit

GPU Graphics processing unit

VHDL VHSIC Hardware Description Language

UART Universal asynchronous receiver / transmitter

DPRAM Dual port random axis memory

Nomenclature

ω	Angular Rate
b_g	The gyro bias vector
η_g	zero mean Gaussian white noise
a	]Acceleration
R	Rotation matrix
ϕ	Roll Angle
θ	Pitch Angle
ψ	Yaw Angle
Θ	Euler Angles
g	The acceleration of gravity
H_1,H_2,H_3	The magnetic field tensor vector of X, Y, Z axis
V_s	The measurement noise vector of the star tracker
A	The attitude matrix
$I_{3\times3}$	3×3 identify matrix
e$\times$	The cross matrix
$\Omega(\omega)$	The skew matrix
H	Observation Matrix
L(A)	The loss function
g(A)	The gain function
B	The attitude profile matrix
e	The anti-symmetric,
K	The Kalman Gain
Y	Gibb's vector
λ	Eigenvalues
X_k	The state vector
Z_k	The observation vector
T_s	The sample time
Q_k	The process noise covariance matrix
R_k	The measurement noise covariance matrix
q	Quaternion
P^-	The prior error covariance
σ_q^2	Averaged quaternion variance
$\hat{x}_k^+$	The prior state vector
κ	a constant secondary scaling paramete
h	measurement function
f	system function
v	Residual vector
$\widetilde{y}$	measured values from the sensors
$\hat{y}$	The predicted value
prob	probability
J	Threshold
G	The Gaussian function
d_j	The width of the neuron
Z_j	The radial basis function
β	The information-sharing factor coefficien

γ	Fault detection factor
γ_0	Threshold of fault detection factor
χ	Sigma points
P^{xy}	The cross-correlation matrix
σ_{mag}	The variance of the magnetometer noise
H_q	he transform function
R^*	The updated covariance matrix
W	Watt

CHAPTER 1

Introduction

Satellite technology has become an indispensable tool for modern society. Satellites are used for various applications including communication, navigation, earth observation, and scientific research. To perform these applications, satellites need to be accurately controlled in their attitude, which is the orientation of the satellite in space. The Attitude , Determination, and Control System (ADCS) in satellites is essential for pointing the payloads in the right direction, maintaining communication links, and ensuring accurate earth observations. There are an increasing large number of research about the satellite attitude, including the Inertial Navigation System (INS), Celestial Navigation System (CNS), Global navigation satellite systems (GNSS), etc.

The INS consisting of an IMU and a processing unit utilizes a gyroscope, accelerometer, and magnetometer to determine the position, velocity, and attitude of the satellite in real time, without any external assistance. The gyroscope establishes the reference coordinate system (inertial coordinate system) and measures the rotational motion of the satellite, while the accelerometer and magnetometer measure the acceleration of motion. The processing unit decodes the velocity, position, and attitude data based on the measured signals and outputs the satellite's navigation information. With its high short-term accuracy, continuous output, and strong anti-interference ability, the inertial navigation system is capable of performing the navigation task autonomously and efficiently. However, the major drawback of the INS is the accumulation of drift errors from the gyroscope with time, leading to a large error in long-term attitude estimation.

The CNS employs a range of sensors, including Earth sensors, Sun sensors, and star trackers, to acquire celestial orientation information. The CNS comprises an Astrometry Measurement section and a navigation solution component, the latter of which typically encompasses algorithms for determining orbit, attitude, and position. While the CNS can furnish exceptionally precise attitude information, the output is subject to discontinuity and environmental perturbations.

GNSS, which stands for Global Navigation Satellite System, is a cutting-edge and highly advanced navigation system that relies on a network of orbiting satellites to provide accurate and reliable location and attitude information. By utilizing a sophisticated array of sensors and receivers, GNSS is able to calculate the precise position, velocity, and orientation of objects with unparalleled accuracy. The technology behind GNSS is truly groundbreaking. With a constellation of satellites in orbit around the Earth, GNSS is able to triangulate the location of any object with a compatible receiver. This allows for a wide range of applications, such as transportation, mapping, military operations, and surveying. Because of its high level of accuracy and reliability, GNSS has become an essential tool in a wide variety of fields.

However, attitude estimation is still challenging, especially in the presence of external disturbances, such as solar radiation pressure, magnetic field, and atmospheric drag. These disturbances can cause the satellite to deviate from its desired attitude, which can lead to reduced performance, reduced lifetime, or even a mission failure. Therefore the Fault, Isolation, and recovery system (FDIR) is necessary for the ADCS in the satellite. The FDIR involves fault detection, isolation, and recovery, which can recover the system safely. The goal of isolation and recovery is to minimize the impact of the fault on the satellite's performance and to ensure that the satellite can continue to perform its mission. Furthermore, the objective of the fault detection system is to identify the fault in a timely manner, with the aim of minimizing false alarms.

Implementing FDIR on Field-Programmable Gate Arrays (FPGAs) is a promising approach to improve the performance and reliability of ADCS in the satellite. FPGAs are integrated circuits that can be reconfigured in space, making them ideal for implementing complex algorithms and control systems. Implementing FDIR on FPGAs allows for real-time processing of the data from the sensors, keeping the accuracy of the attitude estimation and enabling rapid response to faults and recovery to a safe state. Furthermore, the utilization of FPGAs has been proven to significantly enhance programming efficiency and minimize latency through the implementation of optimized algorithms and the utilization of optimized input data. These advantages make FPGAs an attractive choice for various demanding applications, particularly those in the aerospace industry. In addition to their programming capabilities, FPGAs possess remarkable reliability with low power consumption characteristics, making them well-suited for deployment in space-based systems. These attributes make FPGAs an ideal solution for meeting the stringent requirements of space-based applications, where harsh conditions and limited power availability characterize the operational environment. The combination of programming efficiency, reliability, and low power consumption provides an attractive solution for space applications, where performance and reliability are of utmost importance.

1.1 Thesis Motivation

The field of satellite attitude estimation is crucial to the success of the ADCS. To accurately determine the attitude of a satellite, a combination of sensors such as star trackers, sun sensors, and gyroscopes are utilized to gather measurements. The task of attitude estimation requires high levels of accuracy while also being mindful of limited computational resources on the satellite. However, the accuracy of the attitude estimation is highly dependent on the sensors' accuracy, each of which has its own inherent limitations and measurement errors. For example, the IMU may suffer from bias problems, and the Star tracker can easily be impacted by environmental factors leading to discontinuity issues. Therefore, it is necessary to have a novel algorithm to improve the attitude accuracy of INS and keep a low computational load. There is also a need for a novel algorithm that can maintain high levels of attitude accuracy while also being computationally efficient. Such an algorithm would not only enhance the overall performance of the ADCS but also address the limitations of the current systems in use.

The FDIR system in Satellite attitude estimation is an important aspect of satellite design that ensures that the satellite can continue to perform its mission in the presence of faults. A fault in the ADCS, such as a failure of a sensor, can cause the satellite to deviate from its desired attitude. Fault tolerance mechanisms are used to detect and isolate the faulty component and recover the system to a safe state. This is commonly achieved through redundancy. However, the limited space and power resources on satellites, particularly for Cube-Sats, make the implementation of redundant components challenging. Therefore, the FDIR algorithms play a crucial role in ensuring accurate attitude estimation and avoiding false alarms. The algorithms is able to effectively address the problem of sensor outliers and provide a robust solution for fault detection, isolation, and recovery.

The FDIR system has stringent demands on programming speed, as early detection of faults is essential for ensuring a quicker recovery to a safe state. Optimization on the algorithm side of the FDIR system can enhance its running speed. However, the optimization at the hardware level proves to be more effective. An example of such optimization is implementing the FDIR system on an FPGA and further optimizing it through parallel pipelining. This approach reduces latency significantly while maintaining normal resource consumption.

Based on the above reasons, this thesis focuses on the development of satellite attitude estimation by INS, the FDIR system of the INS for the satellite attitude estimation, and the fusion method of INS and CNS. Another objective of this research is to design and implement an FPGA-based FDIR system that can accurately estimate the satellite attitude, detect and isolate faults, and recover the system to a safe state. The FDIR system will be tested on an air-bearing Table with the motion tracking system to evaluate its performance and validate its effectiveness. The results of this research will provide a basis for future work on the development of FDIR systems for other applications, such as robotics, autonomous vehicles, and unmanned aerial vehicles.

1.2 Scope and Objectives

The scope of this thesis is to propose and develop novel algorithms for satellite attitude estimation and FDIR within an onboard satellite ADCS. The thesis aims to address the challenges of achieving high accuracy, robustness, and fault tolerance in satellite attitude estimation and

FDIR in the presence of adverse conditions and faulty sensors. These algorithms will be optimized and implemented on the hardware. The objectives of this thesis are as follows:

(1) Develop a two-layer Kalman filter-based algorithm for satellite attitude estimation utilizing an INS. The algorithm integrates the QUEST and the FQ to achieve higher accuracy in attitude estimation. The algorithm should provide smooth outputs, avoid singularities, and reduce computational complexity.

(2) Propose a groundbreaking approach for the FDIR system within an IMU as part of the satellite ADCS. The approach should involve the fusion of outputs from the AUKF, QUEST, and two Radial Basis Function (RBF) neural networks. The objective is to ensure robustness and high accuracy in satellite attitude estimation, even in the presence of a faulty sensor.

(3) Implement and optimize the FDIR system on the FPGA platform. The optimization should focus on enhancing efficiency, minimizing latency, and maintaining normal resource consumption and temperature. The implementation should leverage pipelining and parallel techniques to achieve prompt fault detection and quick recovery.

(4) Evaluate the proposed algorithms and system implementation through simulations and experiments. The evaluation should assess the accuracy, robustness, fault tolerance, false alarm rate, and efficiency of the satellite attitude estimation and FDIR system. Comparative analysis with existing approaches should be performed to demonstrate the superiority of the proposed solutions.

(5) Provide insights into the significance of the proposed algorithms and system implementation in the field of satellite attitude determination and control. Highlight the contributions and potential applications of the novel, robust, and high-accuracy solutions for satellite ADCS, especially in challenging space environments.

1.3 Thesis Structure

This thesis is structured as follows. Chapter 2 provides a comprehensive review of the related fields. It covers the satellite attitude determination system, fault tolerance problems in satellites, and the implementation of Field Programmable Gate Arrays (FPGAs) for attitude estimation.

Chapter 3 introduces the background and basic knowledge of satellite attitude estimation, including coordinate reference, attitude representation, different sensor models in satellite attitude estimation and kinematic equation and low earth orbit dynamic equation.

Chapter 4 presents a novel algorithm for satellite attitude estimation utilizing an INS system. It introduces a two-layer Kalman filter-based approach for attitude estimation. The proposed algorithm integrates the results of the quaternion estimator algorithm (QUEST) and the factored quaternion algorithm (FQA) to achieve higher accuracy in attitude estimation compared to using each algorithm separately. The Kalman filter uses a quaternion matrix derived from the linear interpolation (LERP) of the observation and process models. The process model, which integrates the output from the gyroscope, provides a smooth output while avoiding singularities and reducing computational complexity. The two-layer architecture is robust against magnetic disturbances and other adverse conditions. Additionally, the second linear interpolation ensures high-precision attitude estimation for vehicles in both static and dynamic environments.

Chapter 5 presents a groundbreaking approach for the FDIR system within an IMU as a part of an onboard satellite Attitude ADCS. This approach involves the fusion of outputs from the AUKF, QUEST, and two RBF neural networks, which are based on an adaptive complementary filter(ACF) and hypothesis testing. This multi-layer approach ensures the robustness and high accuracy of the satellite's attitude estimation. In the preliminary phase of the recovery process, the AUKF is utilized for both fault detection and attitude recovery. This is followed by a secondary recovery phase that employs trained neural networks to estimate the attitude, providing a more comprehensive solution. This multi-level recovery strategy guarantees that the satellite ADCS system can maintain a reasonable level of accuracy even in the presence of a faulty sensor. Additionally, the proposed algorithm provides a lower false alarm rate, resulting in a more reliable satellite attitude estimation solution. The proposed approach represents a significant contribution to the field of satellite attitude determination and control, offering a novel, robust, and high-accuracy solution to the challenging problem of fault tolerance in INS.

Chapter 6 introduces a novel algorithm based on the federated kalman filter for fusion INs and CNS in satellite ADCS, and realizes fault detection, isolation, and recovery in a complicated space environment. The FTFKF includes two sub-filters and a master filter. When in a safe situation, compared with the single INS and CNS, the FTFKF can has higher accurate attitude information. When in failure, the FTFKF can detect which sub-filter is in the failure, and isolate this sub-filter, recover the normal satellite attitude. Furthermore, in case of a false alarm, when the fault detection factor exceeds the threshold constant steps, the sub-filter will be determined in the failure.

Chapter 7 introduces the implementation and optimization of the FDIR system on the FPGA in detail. The optimization of the system on an FPGA board, utilizing pipelining and parallel techniques, significantly enhances its efficiency while minimizing latency and maintaining normal resource consumption and temperature. The FDIR system demands high programming speed due to the need for prompt fault detection and quick recovery. The proposed implementation leverages the ZYNQ platform by integrating complex algorithms, such as AUKF and RBF, into the FPGA part (Programmable Logic (PL)) and integrating them with the ARM processor (Processing System (PS)) using ACF to maximize the utilization of resources available on the ZYNQ platform.

Chapter 8 will conclude the work that has been finished in this thesis and present the future research direction with the potential improvements.

CHAPTER 2

Literature review

2.1 Attitude Determination System

In the field of satellite attitude determination, there are various techniques utilized, with INS, CNS, and GNSS being the most well-known. This chapter will introduce these algorithms and how they are utilized in satellite ADCS.

2.1.1 Inertial Navigation System

Magnetic, Angular Rate, and Gravity (MARG) sensors have been used for real-time attitude estimation for applications such as human body motion tracking (Yun et al. 2005; Lee and Park 2009; Liem and Gavrila 2014) and orientation of quadrotor drones(Marins et al. 2001). MARG sensors include a 3-axis magnetometer, a 3-axis gyroscope, and a 3-axis accelerometer. MARG sensors can be used as an alternative to an IMU for attitude estimation as part of an Attitude and Heading Reference System (AHRS) (Collinson 2013). In addition to the 3-axis gyroscope, a 3-axis accelerometer is found in an IMU; a MARG sensor also contains a 3-axis magnetometer.

However, the sensors required by an AHRS have disadvantages that reduce the accuracy of attitude estimation. Specifically, the gyroscope has gyro bias drift and measurement noise. Gyroscopes are suited to predict an orientation in a short period but not a long steady-state period. The magnetometer can be easily affected by the nearby ferrous materials causing distortions in the local magnetic field. The accelerometer cannot give information on the yaw angle because of its inability to measure the rotation along a vertical axis (Wang et al. 2015b). Hence, sensor fusion is necessary for accurate attitude estimations.

Algorithms, such as the QUEST algorithm, the three-axis attitude determination (TRIAD) algorithm, and the q-Euler algorithm (Markley and Mortari 2000; Guo et al. 2017a; Markley 2002; Campos and Furtado 2017), can estimate attitude from two reference vectors (gravity and magnetic field) and an observation vector. Such algorithms can calculate the normalized quaternion by solving Wahba's problem (Wahba 1965). However, the uncertainty of the reference vectors is still problematic to the attitude determination algorithm due to measurement noises of the sensors, bias errors, and installation errors. For the QUEST (Yun et al. 2005; Crassidis et al. 2007) and TRIAD algorithms, the local magnetic field data affects the yaw angle, the roll angle, and the pitch angles. However, due to the deviation in the direction of the magnetic field vector between different locations, these algorithms should avoid using magnetic field data to calculate the pitch and roll angles. To solve this problem (Liu et al. 2012), presents the Factored Quaternion Algorithm (FQA) to obtain quaternions, meaning magnetic variation only affects the yaw angle. Both QUEST and TRIAD or FQA use current measurements only; this measurement is based on available information about the body and the initial frame. However, these algorithms can only be used for steady-state attitude determination and do not involve any predictive dynamic attitude information.

A method of attitude estimation that offers the ability to perform dynamic attitude determination in real-time uses relative measurements based on a gyroscope and Kalman Filter (KF) (Sun and Deng 2004; Zanetti et al. 2009). In addition, (Makni et al. 2014) uses a viable quaternion-based Adaptive Kalman Filter (q-AKF) to estimate the attitude. While it reduces computational loads, it only estimates two Euler angles. Quaternions serve as a singularity-free substitution for Euler Angles to represent attitude (Alaimo et al. 2013). The estimation scheme (Lefferts et al. 1982) demonstrates a seven-dimensional state matrix, including a group of 4-D quaternions and a 3-D gyro drift-rate bias matrix. (Sabatini 2006; Zhang et al. 2012; Foxlin 1996) introduces a 10-D state matrix (4-D quaternions, 3-D acceleration bias, and 3-D magnetic field bias), and (Sabatini 2011) presents a 9-D state matrix (3-axis acceleration, 3-axis angular velocity, 3-axis magnetometer output). However, the estimation schemes mentioned above require a large number of matrix operations to linearize the process model and the observation model by Jacobian Matrices. However, having multidimensional Jacobian matrix operations inside these algorithms means the computational load is high.

To solve the high dimension problem, a large number of researchers have focused on a two-layered architecture (Yun et al. 2005; Lee and Park 2009; Marins et al. 2001; Wang et al. 2015a; Crassidis et al. 2007; Feng et al. 2017; Lee and Choi 2017; Liu et al. 2014; Valenti et al. 2015; Sabatelli et al. 2012). The first layer uses an attitude estimation algorithm, such as the QUEST algorithm (Yun et al. 2005), adaptive-step Gradient Descent Algorithm (AGSD) (Wang et al. 2015a), or other algorithms (Lee and Park 2009; Feng et al. 2017; Liu et al. 2014)to calculate the computed quaternion that is then input to the Kalman filter. The second layer is the Kalman Filter.

Yun (Yun et al. 2005) uses the QUEST algorithm to compute a quaternion to describe an attitude. Still, the state matrix and observation matrix are 7-D, including 4-D computed quaternion and 3-D angular velocity, which will increase the computational load; however, compared with the traditional two-layered QUEST and Kalman filter approach, it improves the attitude accuracy while decreasing the computational load. Similarly, Seo (Seo et al. 2011) employs the FQA algorithm to calculate a quaternion as the observation model. However, the precision of the attitude is not good, and it is still superior to the traditional FQA and Kalman filter algorithm. Wang (Wang et al. 2015a) uses an adaptive-step Gradient Descent algorithm (AGSD) to get the computed quaternion from the accelerometer. It is necessary to use the Jacobin Matrix to get the step size during this step. Then the calculated quaternion and the angular velocity obtained from the gyroscope are input to the Kalman Filter to estimate a quaternion. However, the accelerometer cannot measure the rotation about the vertical axis; therefore, it is necessary to use a magnetometer to determine the yaw angle in AHRS. The calculation of the Jacobin Matrix to get the step size in AGSD leads to a large computational load and total running time. As an alternative to the above, Lee (Lee and Park 2009) utilizes a 4-D computed quaternion produced by the two-observation quaternion estimation method (O2OQ) with a vector selector used as the observation model. However, during the process, there are two steps to calculate the computed quaternion: O2OQ algorithm and vector selector, which will increase computational cost. Feng (Feng et al. 2017) also treats computed quaternions as the observation model in the Kalman Filter and only uses the two-step geometrically intuitive correction (TGIC) to get the computed quaternion. This algorithm can obtain correct attitude information immune to magnetic distortion and reduce 33.8% of root mean square error. As the accelerometer cannot measure the rotation of the vertical axis, additional sensors are required (Magnis and Petit 2014; Ni and Zhang 2011; Zhai 2017) to solve this problem.

Another method is the traditional one-layer Kalman Filter. Shuster (Shuster 1989) combines the QUSET algorithm and Kalman Filter in different ways. In (Shuster 1989) the Kalman filter is used to smooth the prediction, and the QUEST is used as a filter, which can avoid questionable subtractions and improve the computation speed because it implements the whole attitude calculation rather than incremental corrections. In (Shuster 1990), the QUEST algorithm is used as the data compressor to improve the calculation efficiency. The estimated attitude from vectors in a single frame can be described as Wahba's problem (Wertz 2012), and there have been many solutions to Wahba's problem, such as the TRIAD algorithm and QUEST algorithm (Shuster and Oh 1981). In these algorithms, the QUEST can find the quaternions by minimizing the loss function. In addition, Valenti (Valenti et al. 2015) presents how to combine Algebraic Quaternion Algorithm (AQUA) with the Kalman Filter, which computes the quaternion as the composition of two algebraic quaternions. Alternatively, the Complementary Filter and gradient descent algorithm[37] can deal with the inertial and magnetic sensors (Liang et al. 2011; Madgwick et al. 2011; Calusdian et al. 2011). Calusdian (Calusdian et al. 2011) presents a quaternion algorithm based on the adaptive gain Complementary Filter and FQA. The algorithm utilizes the Complementary filter to take advantage of low-frequency accelerometers, magnetometers, and high-frequency gyroscopes to achieve high accuracy. While QUEST does offer a less computationally intense alternative to FQA, the accuracy of the quaternions offered by the FQA is not as high as the QUEST algorithm.

This paper develops a quaternion-based algorithm with a two-layer architecture. The observation vector is obtained by the fusion of the QUEST algorithm and FQA using Linear interpolation (LERP). In this scheme, the first step is using the QUEST algorithm and the FQA to preprocess the data from the accelerometers and magnetometers to get two groups of computed quaternions. Then these computed quaternions go through LERP to get the measured quaternions to form an observation quaternion model before being input into a Kalman Filter. In addition, the angular rate from the gyroscope is used to get the approximated quaternion that will be combined with the quaternion from the LERP to obtain the optimal quaternion for the Kalman Filter. Then the second layer combines the measured quaternions and optimal quaternions in the Kalman Filter to get the estimated quaternions.

2.1.2 Celestial Navigation System

CNS is a method of determining the position of an object in space by using observations of celestial objects, such as the sun, moon, and stars. Nowadays, CNS is commonly used as a navigation system for many satellites, providing reliable and accurate navigation information. The celestial observation in CNS is based on different kinds of sensors, such as star trackers (Liebe 1995), sun sensors(Psiaki 1999), Horizon sensors(Tekawy et al. 1996).

Star trackers are essential devices for satellite attitude estimation, providing accurate orientation information based on the positions of stars. These devices have been extensively utilized in a variety of satellite missions due to their exceptional accuracy. In particular, Yelubayev et al. 2015 describes the development of star trackers for satellite attitude determination, focusing on the optical system of the tracker.

The German Aerospace Center (DLR) is currently designing and developing a novel star tracker for use as the primary attitude sensor in the SHEFEX mission. The developed star tracker is a low-cost and low-accuracy sensor that meets the attitude accuracy requirements of the mission (Samaan and Theil 2012). Additionally, Samaan et al. 2011 presents an open-loop and closed-loop hardware test bench to demonstrate the functionality of star trackers. The test bench consists of an optical star field simulator, a real-time simulation computer, and the star tracker being tested. Recently, an ultra-low-cost star tracker based on the Raspberry Pi has been introduced by Gutiérrez et al. 2020. This new design offers a cost-effective alternative to

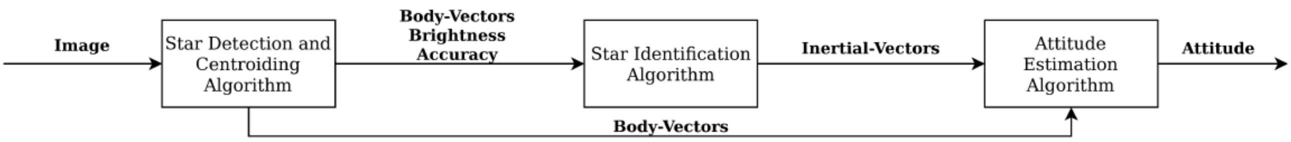


FIGURE 2.1: Basic processing flow of the star tracker (Rijlaarsdam et al. 2020)

conventional star trackers while maintaining the essential functionality required for satellite attitude estimation. In addition, Ju et al. 2000 has developed a micro star tracker suitable for small spacecraft, which combines a high quantum efficiency sensor, a high-speed dedicated microcomputer, and a lightweight, high aperture optical system. This design provides a practical solution for small spacecraft requiring high-precision attitude determination. Finally, Muruganandan 2018 has designed the Arcsecond Pico Star Tracker (APST), which has been optimized for use on nanosatellites. The APST is expected to provide pointing knowledge in the arcsecond range, which is a significant improvement over previous designs.

In the development of a star tracker, the algorithm is as crucial as the hardware in achieving high-precision attitude determination. Figure 2.1 illustrates the basic processing flow of a star tracker, which comprises three phases: star detection and centroiding, star identification, and attitude estimation. Various studies have been conducted on different phases of the star tracker, and algorithms can be classified into two main categories (Spratling IV and Mortari 2009). The first category is capable of autonomously identifying stars in a scene without prior attitude information, which addresses the lost-in-space problem. The second category involves a more efficient star identification algorithm that leverages some available attitude information. To tackle the star identification problem, algorithms of the first category treat stars as vertices in a subgraph, with angular distances between stars serving as edge weights. The task of identifying stars then involves finding an isomorphic subgraph in a database. Several algorithms fall under this category, including the triangle algorithm (Liebe 1993), polygon angular matching algorithm (Wertz 1978), group match algorithms (Kosik 1991), and pyramid algorithm (Mortari et al. 2004). Additionally, (Mortari 1997) proposed a fast and robust technique for identifying stars in a large catalog using only their angular separation, suitable for spacecraft equipped with wide field-of-view star trackers. This algorithm involves two identification processes: the K-vector Star-Pair Identification Technique (SPIT) and the Reference-Star Star-Matching Identification Technique (SMIT). In a similar vein, (Samaan et al. 2005) introduced two novel algorithms for recursive mode star identification. Based on the spherical polygon search (SP-search) algorithm, the first approach allows for accessing all the cataloged stars observed by the sensor field-of-view (FOV) and recursively adding or removing candidate cataloged stars based on predicted image motion induced by camera attitude dynamics. The second method relies on star neighborhood information and a catalog neighborhood pointer matrix to access the star catalog.

Algorithms in the second category assign a unique pattern to each star based on its relative position to nearby stars and then search for the closest match to the measured pattern in a pre-existing database. Examples of such algorithms include the grid algorithm (Padgett and Kreutz-Delgado 1997), the singular value method algorithm (Kim et al. 2003), the Log-Polar transform algorithm (Wei et al. 2009), the genetic algorithm-based identification algorithm (Paladugu et al. 2003), and the ordered set of points algorithm (Zhu et al. 2018). An efficient star pattern recognition algorithm was presented by (Lamy Au Rousseau et al. 2005), which ensures compatibility between the software and the imaging sensor's noise level. Another algorithm, proposed by (Kolomenkin et al. 2008), matches stars in an image taken with a camera to stars in a star catalog using a geometric voting scheme. In this method, a pair of stars in the catalog vote for a pair of stars in the image if the angular distance between the stars in both pairs is similar. (Rufino and Accardo 2003) introduced an analytical study of the centroiding algorithm's error and showed that both a systematic and a random contribution exist. From this approach, the position computation accuracy was improved from 0.01 to 0.005 pixels

Overall, the continuous development of star tracker technology has provided a wide range of options for satellite missions requiring accurate attitude determination. These advancements have made significant contributions to the field of satellite engineering and continue to offer new possibilities for future missions.

2.1.3 Global Navigation Satellite Systems

GNSS has become ubiquitous in the modern area and has been utilized in several applications, such as UAV and satellite attitude estimation, determining the attitude of any outdoor moving object (Raskaliyev et al. 2020).

The GNSS baseline model can be described in the following formula according to the Gauss-Markov modelTeunissen 2008:

$$E(y) = Az + Bb; D(y) = Q_y \tag{2.1}$$

where E(.) and D(.) denote the expectation and dispersion operator, y is the given GNSS data vector of order m, z and b are the unknown parameter vectors of order n and p, and where A and G are the given design matrices that link the data vector to the unknown parameters. The geometry matrix G contains the unit line-of-sight vectors. The variance matrix of y is given by the positive definite matrix Q_y, which is assumed to be known. The data vector y will usually consist of the observed minus computed' single- or multiple frequency double-difference (DD) phases and/or pseudo-range (code) observations accumulated overall observation epochs. The entries of vector z are then the DD carrier phase ambiguities, expressed in units of cycles rather than range. They are known to be integers, $z \in Z^n$. The entries of vector b will consist of the remaining unknown parameters, such as baseline components (coordinates) and possibly atmospheric delay parameters (troposphere, ionosphere). They are known to be real-valued,$b \in R^p$.

High-precision positioning, navigation, and attitude determination heavily rely on carrier phase ambiguity resolution, as emphasized in Teunissen's work (Teunissen 2007). The initial proposal for ambiguity resolution in GNSS attitude determination was presented by Peng et al. (Peng et al. 1999). Meibo et al. (Meibo et al. 2013) provide a concise overview of various GNSS-based models and methods for spacecraft attitude determination using phase measurements. In their work, Bing et al. (Bing et al. 2013) compare two different approaches, constrained Least-squares Ambiguity Decorrelation Adjustment (LAMBDA), and multivariate constrained LAMBDA, for GNSS-based attitude estimation. Baroni and

Kuga (Baroni and Kuga 2012) conduct both theoretical and experimental analyses of the Least-Squares Ambiguity Search Technique (LSAST) and LAMBDA algorithms using quaternion formulation for attitude determination. Their work aims to compare the two methods in terms of their efficiency and accuracy. Different algorithms, including Cohen's traditional method (Li 2000) and the vectorization method (Kuang and Tan 2002), are used to convert scalar carrier phase measurements into 3-D vector measurements, simulating a Wahba-like cost function. This conversion allows for conventional attitude determination algorithms, such as QUEST, to determine the vehicle's attitude. Li et al. (Li 2000) propose an algorithm that directly computes all nine elements in the attitude matrix based on the scalar SDCP cost function, classified as a non-Wahba-like problem.

The determination of attitude by GNSS can be classified into three categories: (1) baseline position-based estimation, (2) Wahba-like problem, and (3) non-Wahba-like problem. The baseline position-based method utilizes the estimated antenna position in the inertial reference frame instead of SDCP, and it includes the DCM attitude determination method, which is described in previous studies (Lu et al. 1993, Lin et al. 2004). Specifically, this method computes respective Euler angles based on each axis of the baseline position vector.

In recent years, there has been a growing interest in developing GNSS algorithms to address various error sources, such as multipath, ionospheric delay, and receiver noise (Goh and Low 2017). To mitigate these errors, attitude estimation based on GNSS determination has been proposed, which includes Extend Kalman filter-based (EKF)Weill 1994, unscented Kalman filter (UKF)Julier and Uhlmann 2004, and particle filter (PF) methods. Kalman filter-based methods estimate the satellite's attitude and GNSS measurements simultaneously, while UKF methods are more robust to nonlinearity in the system. PF methods, on the other hand, rely on a Monte Carlo-based approach to provide a probabilistic estimate of the satellite's attitude. Overall, these algorithms offer promising solutions for GNSS-based attitude estimation, which is a critical task for various space and navigation applications.

2.1.4 Multi-sensor Fusion

Multi-sensor fusion is a powerful approach to enhance the accuracy and reliability of satellite attitude estimation by combining the strengths of different sensors while compensating for their individual weaknesses. Among various combinations of sensors, the integration of INS and CNS has been widely studied and applied in various aerospace missions. INS provides accurate and low-drift attitude and position measurements by utilizing gyroscopes, magnetometers, and accelerometers to measure the inertial forces and movements of a satellite. CNS, on the other hand, allows for the monitoring of external forces and environmental conditions to provide feedback on the satellite's position and orientation. The INS/CNS system combines the complementary information from these two sensors to create a more robust and accurate navigation system.

2.1.4.1 Inertial Navigation System/Celestial Navigation System

To further improve the performance of INS/CNS, researchers have proposed various innovative approaches. For example, Ning et al. 2013 applied INS/CNS in lunar rovers, while (Qu et al. 2010) used it in spacecraft. In (Wu et al. 2013), an INS/CNS integrated system was developed to eliminate the accumulated errors of the INS for a moving object by using CNS output. (Wu et al. 2022) proposed an adaptive main Kalman filter for ambiguity, eliminated GNSS/INS tightly coupled integrated systems, and developed a robust adaptive subfilter for GNSS individually. Another innovative approach is presented in Gou et al. 2019, where a novel INS/CNS integrated navigation system based on multi-star pseudo measurements was developed to solve the problem of inaccurate navigation parameter estimations caused by small stellar angular distances in a single field-of-view star sensor. Finally, (Yang et al. 2022) proposed a SINS/CNS integrated navigation scheme based on a novel mathematical horizon reference determination method. This approach utilizes the inertial coordinate system to construct the mathematical horizon reference, which can decouple and compensate for attitude and position errors, resulting in more accurate navigation results. To address the problem that traditional CNS takes a long time to identify the star map, which limits the improvement of the dynamic response-ability, (Mu et al. 2020) developed an INS/CNS deeply integrated navigation method that includes a deeply integrated model and a second-order state augmented H-infinity filter. This approach achieves better dynamic response-ability by optimizing the identification of the star map in CNS.

In conclusion, the integration of INS and CNS is a promising approach for improving satellite attitude estimation, and innovative approaches such as multi-star pseudo measurements and deeply integrated models have been proposed to further enhance the performance of INS/CNS systems. These advances in multi-sensor fusion can lead to more reliable and accurate navigation systems for aerospace missions.

2.1.4.2 Inertial Navigation System/Global navigation satellite systems

The integration of INS and GNSS systems is a common approach used to enhance navigation accuracy and reliability. By combining these systems, the GNSS system can correct the errors in the INS system, resulting in improved overall accuracy and robustness (Jing et al. 2022). The limitations of GNSS accuracy, caused by factors such as the number of available satellites, signal-to-noise ratio, and multipath effect, can be reduced by the use of an INS system that is not affected by these factors (Arribas Lázaro 2012). In addition, the combination of INS and GNSS can result in more accurate velocity estimates by utilizing the inertial measurements to mitigate the effects of GNSS system drift (Liu et al. 2018).

Recent research has proposed innovative approaches to further improve the performance of INS/GNSS integrated systems. For instance, Ding et al. (Ding et al. 2022) proposed a data fusion scheme that leverages the complementary advantages of a MARG sensor and a low-cost GNSS receiver. The resulting approach offers enhanced navigation performance with improved accuracy. Moreover, researchers have proposed robust INS/GNSS integration approaches that can effectively compensate for errors in MEMS-SINS systems. For example, Wang et al. 2020 proposed a tightly-coupled navigation approach aided by non-holonomic constraint (NHC) that achieves accuracy improvements of about 46% (position), 35% (velocity), and 15% (attitude), compared to traditional approaches. Another challenge faced by INS/GNSS integration is filter divergence caused by unknown or variable noise statistical characteristics in a dynamic environment. To address this, Sun et al. 2022 proposed a fusion adaptive filtering scheme combining innovation-based adaptive estimation (IAE) and the adaptive fading Kalman filter (AFKF) to prevent filter divergence and maintain optimal performance. Finally, Xu et al. 2023 proposed a motion-constrained GNSS/INS integrated navigation method based on a BP neural network (MC-BP method). This method fuses a BP neural network with motion constraints to predict the pseudo-measurement of GNSS, resulting in improved navigation performance.

In summary, the integration of INS and GNSS systems offers improved accuracy and robustness, and recent research has proposed innovative approaches to further enhance the performance of integrated systems, including the use of complementary sensors, robust approaches to compensate for errors, and adaptive filtering schemes.

2.1.4.3 Global navigation satellite systems/Celestial Navigation System

To enhance the accuracy, reliability, availability, and integrity of positioning data, a combination of GNSS and CNS has been proposed. While GNSS provides accurate positioning data, its effectiveness can be limited by environmental factors and interference, as noted in previous research (Feng et al. 2019). However, CNS can complement GNSS by providing additional information, such as ground-based aids, to improve accuracy and mitigate the impact of jamming and spoofing attacks. By integrating these two systems, the overall accuracy and reliability of the navigation system can be improved, resulting in more precise and trustworthy positioning information. Moreover, combining GNSS and CNS can also enhance the availability and integrity of the navigation system. GNSS can be vulnerable to interference and satellite outages, but the integration of CNS provides a backup system and redundancy that can ensure the navigation services are still available even in the absence of GNSS. This CNS/GNSS hybrid system provides a more robust and resilient navigation solution that can effectively operate even when the GNSS is in a failure situation, as stated in recent research (Dhahbane et al. 2021).

In summary, integrating GNSS and CNS offers significant advantages, including improved accuracy, reliability, availability, and integrity. By combining these two systems, the navigation system can overcome the limitations of each and provide a more comprehensive and reliable solution.

2.1.4.4 Inertial Navigation System/Global Navigation Satellite Systems/Celestial Navigation System

The combination of INS, GNSS, and CNS systems is widely used for high-accuracy navigation and positioning applications due to its ability to provide reliable and accurate information in a range of environments. The INS system utilizes gyroscopes and accelerometers to provide precise navigation information, while the CNS system employs a map-matching algorithm to offer dependable navigation information even in areas with obstructions or interference. The GNSS system utilizes satellites to provide highly accurate and reliable positioning information, even in areas without a line of sight. Combining these three systems results in a robust and accurate navigation and positioning system that is capable of operating in a wide range of environments.

However, the performance of data fusion algorithms based on the CKF can be degraded when there are non-Gaussian noise and process-modeling errors in the system model. To address this issue, (Liu and Chen 2022) proposed the use of the AFCCKF-ODF algorithm for optimal data fusion. The AFCCKF-ODF algorithm is based on the Adaptive Fading maximum Correntropy generalized high-degree CKF and is capable of handling non-Gaussian noise and process-modeling errors in the system model. In addition, (Gao et al. 2018) presented an unscented Kalman filter (UKF) based multi-sensor optimal data fusion methodology for INS/GNSS/CNS integration. This methodology is based on a nonlinear system model and is capable of providing highly accurate navigation and positioning information in a range of environments. Furthermore, (Hu et al. 2016) proposed a modified version of the federated Kalman filter (MFKF) for INS/GNSS/CNS integration. The MFKF improves the computational efficiency of the FKF's master filter and is capable of providing highly accurate navigation and positioning information in a range of environments. Finally, (Xu et al. 2022) presented a chi-square test-based adaptive federated cubature Kalman filter (CAFCKF) to improve the stability of navigation in hypersonic cruise vehicles (HCVs). The CAFCKF algorithm is capable of handling the high levels of noise and uncertainties that are present in HCV navigation systems and is capable of providing highly accurate and reliable navigation and positioning information.

2.2 Fault Tolerance Scheme

The satellite Attitude Determination System (ADS) is crucial for satellite operations and requires high reliability and safety. The ADS reliability is directly dependent on the sensors, whose failure is the main reason for system breakdowns (Balaban et al. 2009; Nasrolahi and Abdollahi 2018). The ADS failure onboard a satellite is a high risk of mission failure and can even lead to the loss of the satellite. Therefore, Fault Detection, Isolation, and Reconstruction (FDIR) in ADS is essential for the reliability and safety of the satellite (Yuan et al. 2021; Tipaldi and Bruenjes 2015; Hasan et al. 2022; Carvajal-Godinez et al. 2017). The methods for detecting faults are divided into hardware redundancy(Nasrolahi et al. 2012; Liang and Jia 2015; Hwang et al. 2009) and analytical redundancy (Guerrier et al. 2012; Wang et al. 2015b; Venkateswaran et al. 2002; Yoon et al. 2011; Nasrolahi and Abdollahi 2018; Hajiyev 2014; Guo et al. 2017b; Xiong et al. 2013). For the hardware redundancy, multiple sources of sensor information for attitude estimation can be collected from Inertial Measurement Units (IMU), sun sensors, star trackers or infrared Earth sensors and compared with each other to detect the faults. However, the redundant hardware leads to extra mass and size, which infers a higher satellite cost that is undesirable for small satellites like CubeSats (Tipaldi and Bruenjes 2015; Patton et al. 2010; Scharnagl et al. 2022). In comparison, the analytical redundancy is mostly a model-based method using the system's dynamic and kinematic models. These model-based methods produce residuals using analytical relations and output an attitude estimate without extra hardware requirements (Nasrolahi et al. 2014). Another approach to fault tolerance is the model-free method (Li et al. 2020), which is based on prior known data, such as fault diagnostic trees (Barua et al. 2009), neural networks (Cai et al. 2007; Sheng et al. 2018; Xinyuan et al. 2012), fuzzy sets(Mei et al. 2022; Gao et al. 2021; Hou et al. 2022), possibility theory (Cayrac et al. 1996), and telemetry data (Nalepa et al. 2022). Some researchers use neural networks and genetic algorithms for fault detection and isolation of control moment gyroscopes onboard satellites(Muthusamy and Kumar 2021). However, these methods mainly depend on large datasets, and the limited data available from the space environment means that these methods cannot cope with real-world fault scenarios. Furthermore, some researchers believe the working process of the neural network, similar to a black box, is unreliable in satellite attitude estimation (Li et al. 2020).

Attitude estimation for satellites cannot be defined entirely by a linear system, which means traditional Kalman Filters (KF) have limited capability. Therefore, Extended EKF(Lim and Park 2014; Pirmoradi et al. 2009; Mehra et al. 1995), Unscented Kalman Filters (UKF) (Uhlmann and Durrant-Whyte 1995; Le and Matunaga 2014; Soken and Hajiyev 2010; Xiong et al. 2007; Pourtakdoust et al. 2022), and Federated Kalman Filters (FKF)(Zhou et al. 2016; Xu et al. 2022; Hu et al. 2016; Bae et al. 2011; Ushaq et al. 2013) are used. The EKF uses hypothesis testing for effectively detecting and isolating a faulty sensor; however, the nonlinear system's first-order linearization can cause increasing errors in the mean and covariance of the state vector (Crassidis and Markley 2003). For an adaptive unscented Kalman Filter (AUKF), the main deficiency is the heavy and complex calculation load from the residual generator, which leads to delays in the fault isolation process (Le and Matunaga 2014), but it has a recovery function because of the adaptive covariance matrix. In (Pourtakdoust et al. 2022), the modified unscented Kalman Filter (MUKF) is presented to estimate the gyro=less satellite under faulty sensor conditions. In (Zhou et al. 2016), the Fault Tolerance FKF (FTFKF) can detect a faulty sensor by comparing and analyzing dimensionless fault detection factors and then selectively fusing the sub-filter outputs to improve the satellite attitude estimation accuracy. However, this process is computational-intensive. Furthermore, (Hu et al. 2016) presents a modified version of the FKF for INS, GNS, and CNS, this integration improves the computational efficiency involved in the master filter of FKF but requires hardware redundancy.

In the field of fault-tolerant navigation systems, the conventional approach entails the integration of three fundamental filter types with a residual processor, as depicted in Figure 2.2 (Williamson et al. 2009). The overarching objective of this system is to detect and isolate any fault signals that may be present in the measurements and actuator commands received from

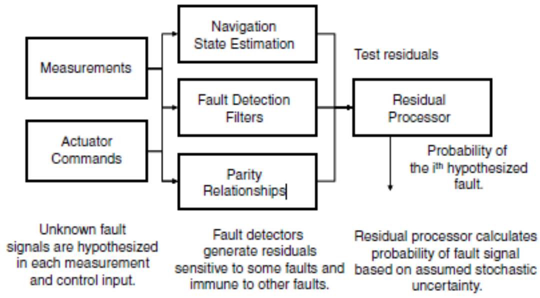


FIGURE 2.2: The traditional fault-tolerant navigation system (Williamson et al. 2009)

various sensors. To achieve this objective, the measurements and actuator commands undergo processing through a sequence of filters. These include a standard state estimator filter, a bank of fault detection filters, and a bank of parity relationships. The residuals obtained from each of these filters are subsequently fed into the residual processor, which comprises a hypothesis-testing scheme. In the residual processor, separate hypotheses are formulated for each potential fault signal. The processor then generates an estimate of the likelihood of each fault signal, aiding in the determination of the presence of any faults within the system.

This thesis presents a novel FDIR subsystem in chapter 5, made up of three important stages. The first stage acquires two groups of quaternions from the AUKF, and the QUEST algorithms (Shuster and Oh 1981). Then two residual generators produce the residual used for fault detection and fault isolation. The QUEST algorithm can achieve a high accuracy attitude estimation and avoid the gyro bias and noise of the magnetic field data. Compared with (Le and Matunaga 2014) using estimated errors and gyro bias as the input of the residual generators, the proposed scheme employs the quaternion as the input of the residual generator. In normal operation, the residuals are zero-mean with white noise when each sensor is healthy without fault or failure. In contrast, a biased residual means there is a failure or fault in the sensor. The second stage has fault detection, which employs the residuals to determine whether a fault has occurred, and in fault isolation, the specific faulty sensor within the IMU is identified. This phase uses statistical methods to determine whether the residuals have strayed considerably from zero. However, if the threshold is not accurate, the residuals will be not only sensitive to faults but also noise, distributions, and model uncertainties, and this will cause a false alarm.

To decrease the rate of false alarms the Parity Equation (PE) (Jin and Zhang 1997 Du et al. 2019)and Chi-square (Wang et al. 2016 Kottath et al. 2017)approaches are used with residual generators for detecting gyro and quaternion faults, respectively. In the proposed scheme, the hypothesis testing algorithm is employed to reduce false alarms Kottath et al. 2017. Compared with traditional statistical methods Yuan et al. 2021, this scheme decreases the false alarm rate and avoids constant weightings of AUKF, QUEST, and RBF. At the preliminary recovery phase, this scheme also gives a quaternion obtained from the AUKF to update the measurement noise of the covariance matrix within the AUKF.

2.3 Relative Attitude

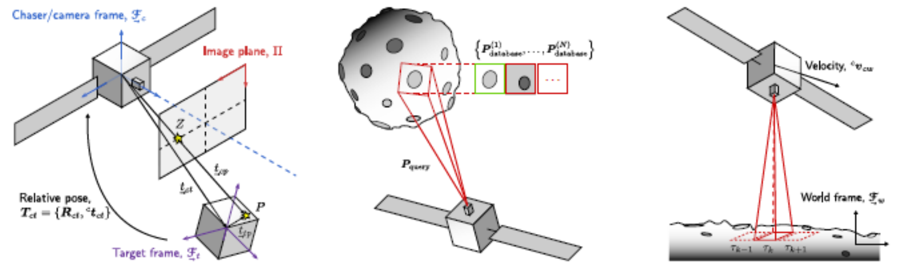
Relative attitude is a critical aspect of satellite attitude estimation, especially for rendezvous and docking in formation fly (FF) and the fault isolation and recovery in deep space (Williamson et al. 2009). Maintaining precise knowledge of relative position and attitude is essential for the success of many space missions, and it will play an important role in future applications of formation missions and space exploration.

For instance, in the context of rendezvous, proximity operations, and docking (RvD) of two identical 3U CubeSats, the final translation phase and proximity operations are critical to mission safety since multiple spacecraft are involved in the process. To address this challenge, researchers have proposed various onboard relative position and attitude estimation and control methods.

Philip et al. (Philip and Ananthasayanam 2003) describe a scheme for autonomous space rendezvous and docking systems that includes homing, closing, final translation, and proximity operations. They highlight the importance of the final translation phase and proximity operations from a mission safety perspective and present an onboard relative position and attitude estimation and control method for these phases. Torisaka et al. (Torisaka et al. 2013) propose a method for controlling relative position and attitude using only magnetic force with multi-dipole for formation-flying spacecraft. Their approach has potential applications for FF missions that require precise formation control. Qiao et al. (Qiao et al. 2013) address the challenge of vision-based relative position and attitude estimation for spacecraft RvD by proposing a dual quaternion-based algorithm for the final phase of the process. Yu et al. (Yu et al. 2014) present a stereo vision-based method for estimating relative pose during the final phase of rendezvous and docking of noncooperative satellites. Their approach utilizes sparse stereo vision algorithms, which could have significant advantages over dense stereo algorithms in terms of computational efficiency. Shakouri and colleagues (2018) proposed an innovative algorithm for fault detection in satellite formation flying. Their approach leverages the relative attitude between two satellites and is able to operate effectively even in the presence of time-varying faults. Importantly, the algorithm does not require the addition of any extra subsystems to the satellites themselves, making it a practical and cost-effective solution for real-world applications. In a related study, Kim and colleagues (2000) also explored the use of Satellite-to-Satellite tracking (SST) methods for fault detection and recovery in satellite networks. Through simulations, they demonstrated the feasibility of their approach and highlighted the potential benefits of leveraging SST in future spacecraft missions. To accurately describe the relative motion of spacecraft formation flying, Baoyin and colleagues (2002) presented a novel method based on relative orbital elements. This approach is particularly well-suited to elliptical orbits with arbitrary eccentricity and provides a more accurate and detailed understanding of the dynamics at play in satellite formations.

There are various approaches to relative navigation (RN) for formation flying (FF) satellites, each with its advantages and limitations. Low-orbit satellites can utilize the Global Positioning System (GPS) for RN, as described in Montenbruck and Gill's work (Montenbruck et al. 2002), while deep space missions may require autonomous methods, such as those presented by Purcell and Davis in their study (Purcell et al. 1998). Another novel approach for RN is presented by Tweddle et al. (Tweddle and Saenz-Otero 2015), who developed a design for a relative state estimator that employs a small fiducial target and a single monochrome camera, effectively solving the exterior orientation problem (Horn et al. 1986).

In addition, Kim and colleagues (Kim et al. 2007) developed a new method for relative navigation and attitude estimation of spacecraft in formation flying, which involves coupling line-of-sight measurements with gyro measurements and dynamical models in an extended



(a) Rendezvous with non-cooperative spacecraft (b) Asteroid pinpointing via patch classification (c) Terrain navigation

FIGURE 2.3: Identification of potential relative navigation scenarios for the application (Song et al. 2022)

Kalman filter (EKF) to determine the relative attitude, position, and gyro biases. Xing and colleagues (Xing et al. 2010) proposed an approach for estimating relative position and attitude for satellite formation flying using an EKF and 6 degrees-of-freedom (DOF) relative motion models. These models include two parts: the relative translational dynamics of the center of mass (c.m) and the rotational dynamics of two spacecraft. This is an improvement over traditional three DOF point mass models (Alfriend 2002), which ignore the influence of the angular motion of the spacecraft body with respect to the Earth or other spacecraft.

The relative satellite attitude estimation also can be done by Artificial Intelligence (AI), deep Learning (DL), Machine Learning (ML), and neural network(Song et al. 2022) in recent years. The relative attitude estimation based on neural network is often employed in the following scenarios which is shown in Figure 2.3:

(1) non-cooperative rendezvous with a spacecraft
(2) terrain navigation for descent and landing;
(3) asteroid explorations and asteroid patch pinpoint localization

(Sharma et al. 2018) proposed a deep CNN for relative pose classification of non-cooperative spacecraft to address two issues: robustness to illumination conditions due to a lack of reliable visual features and scarcity of image datasets required for training and benchmarking. (Sharma and D'Amico 2020) also presented the Spacecraft Pose Network (SPN), the first neural network-based method for on-board estimation of the relative position and attitude, of a known noncooperative spacecraft using monocular vision. (Harl et al. 2013) developed an NN-based state observer, which is a modified state observer to estimate gravitational uncertainties that spacecraft experience in an asteroid orbiting scenario. (Ren et al. 2015) employed Faster Region-based Convolutional Neural Network (R-CNN) architecture to detect the 2D bounding box of the target in the input image. Mathematically, the Spacecraft Pose Network utilizes a Gauss-Newton algorithm to solve a minimization problem for the estimate of relative position, for which the required initial guess is obtained from the bounding box (Kehl et al. 2017).

In summary, relative attitude estimation is a critical component of satellite operations, particularly in RvD processes for FF missions and fault isolation and recovery in deep space exploration. Various methods have been developed for onboard relative position and attitude estimation and control, RN, and fault detection and recovery, utilizing techniques such as magnetic force, vision-based estimation, and GPS. These approaches have the potential to significantly enhance the safety and efficiency of space missions in the future.

2.4 Field-programmable Gate Arrays

Field-programmable gate arrays (FPGAs) are a type of programmable hardware device (Programmable Array Logic (PAL), Generic Array Logic (GAL), or Complex Programmable Logic Device (CPLD)) that can be configured to perform a wide variety of digital logic functions. They are utilized in a variety of applications, including digital signal processing, image processing, and communication systems. When the FPGA is utilized as a semi-custom circuit in an Application-Specific Integrated Circuit (ASIC), it has 4 main advantages:

(1) FPGAs can address the issue of the lack of full-custom circuits. Unlike full-custom circuits, which require specialized design and manufacturing processes, FPGAs can be programmed to perform a wide range of functions, making them highly versatile.
(2) FPGAs solve the problem of the limited number of gate circuits available in traditional programmable logic circuit devices, such as Programmable Array Logic (PAL) and Generic Array Logic (GAL). With FPGAs, the number of gates can be tailored to specific application needs, allowing for greater flexibility in design.

(3) FPGAs have the ability to be reprogrammed, making them suitable for applications that require frequent changes or updates. This feature also makes them suitable for prototyping and testing new designs.
(4) FPGAs can be used for prototyping and testing digital circuits before committing to the more expensive process of creating an ASIC. This can reduce the risk of costly errors or design flaws and speed up the development process.

Field-Programmable Gate Arrays (FPGAs) are versatile devices that can be used to implement a wide range of digital circuits, from simple gate circuits to complex designs. FPGAs are programmable hardware devices that allow for the logical allocation of resources, including logic units, Random Access Memory (RAM), and Digital Signal Processing (DSP) units. The ability to reprogram and load new designs onto FPGAs quickly and inexpensively makes them an attractive choice for a variety of applications.

While FPGAs are highly adaptable devices, many high-level languages cannot be directly implemented on them. Instead, hardware description languages (HDLs), such as VHDL and Verilog, are commonly used to design FPGA circuits. However, the Xilinx Company has developed a new FPGA board, the PYNQ-Z2, which allows for the generation of register transfer level (RTL) designs using system C code. The PYNQ-Z2 board is based on the ZYNQ-7020 system-on-chip (SoC), which includes ARM Cortex-A9 processors. This allows for the integration of a Processing System (PS) within the FPGA, which is separate from the Programmable Logic (PL). In our tasks, we have successfully implemented algorithms on the PYNQ-Z2 using system C code to generate RTL designs. This has allowed us to take advantage of the versatility of FPGAs while using a high-level language. The integration of a Processing System within the FPGA has also allowed us to perform tasks that would otherwise require external hardware or software. The ability to rapidly prototype and test designs on the PYNQ-Z2 has reduced development costs and accelerated our research.

FPGAs are complex digital circuits that are composed of an array of configurable logic blocks (CLBs), input-output blocks (IOBs), and programmable interconnects. These components can be configured by the user to implement a desired digital circuit. Programmable interconnects are responsible for routing signals between the CLBs and input/output (I/O) blocks, allowing

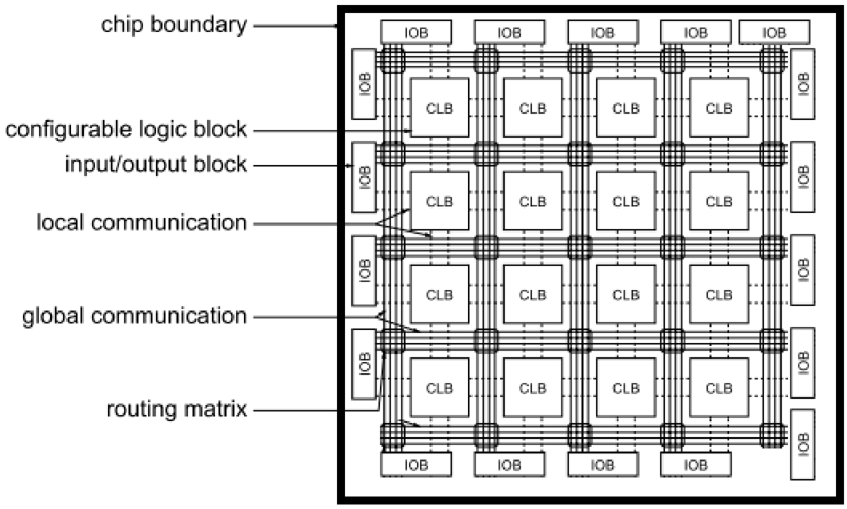


FIGURE 2.4: Structure of the FPGA (Souissi et al. 2012)

for the transmission of information throughout the FPGA. The structure of FPGA is shown in figure 2.4.

The CLBs are the heart of the FPGA and contain logic gates, flip-flops, Look-Up Tables (LUTs), and other digital building blocks. These elements can be utilized to construct sophisticated digital circuits. Each LUT is connected to the input of a Flip-Flop (FF) which drives other logic components such as Full Adders (FAs), Multiplexers (MUXs), and in-block Input/Output (I/O). All CLBs modules are connected to each other or the IOB by metal wiring, creating a cohesive and integrated system.

To program the FPGA, logic synthesis, layout, and routing tools are utilized. This enables the rapid creation and testing of a designed logic circuit, making FPGAs a powerful technology for modern Integrated Circuit (IC) design verification. The memory unit of the FPGA stores information about the connections between modules and I/O, the logic functions of the CLBs, and ultimately establishes the general features of the FPGA device. This high-speed programmability and versatility make FPGAs an important tool in modern digital design.

FPGAs are electronic devices that integrate various basic components to create complex circuits. This integration leads to a reduction in the required area, an increase in the speed of operation, and the ability to perform a wider range of functions. These functions include general-purpose DSP blocks, multipliers, embedded processors, high-speed I/O logic, and embedded memory. One of the key characteristics of FPGAs is their ability to be programmed during running time, which enables high efficiency in reprogrammable computing or re-programmable systems. This means that the onboard Central Processing Unit (CPU) can reconfigure itself to suit real-time changing tasks. A complete System-on-Chip (SoC) includes the logic blocks and interconnects of FPGAs and embedded microprocessors. One example of this technology is the PYNQ-z2 (Xilinx Zynq-7020), shown in Figure 2.5. This device features a dual-core ARM Cortex-A9 processor with a 28nm-based processing system (PS) and connections to the programmable logic (PL).

2.4.1 PYNQ-Z2 Board

The PYNQ-Z2 board, as illustrated in Figure 2.6, has been purposefully designed to facilitate cutting-edge research in the areas of embedded systems, DSP, and computer vision. This powerful board offers users the ability to customize their hardware acceleration for a variety of tasks, such as image processing and machine learning while boasting a considerable 512MB of DDR3 memory. In addition to its impressive memory capabilities, the PYNQ-Z2 board also features a USB-UART interface, a USB On-The-Go (OTG) port, an Ethernet port, a micro SD card slot, and an array of peripheral connectors. These features make it an extremely versatile tool, well-suited to a wide range of research applications.

Python programming language is a prominent feature of the PYNQ-Z2. Python is a high-level language that is easy to understand, making it an excellent choice for both novice and expert developers. The language's readability enables developers to write, test, and debug code quickly, which results in reduced development time and increased efficiency. Moreover, Python's flexibility makes it an excellent fit for a diverse range of applications, including embedded systems, Internet of Things (IoT) devices, data analysis, and machine learning. As a result, developers can utilize the same language across various projects, reducing the learning curve and enhancing productivity.

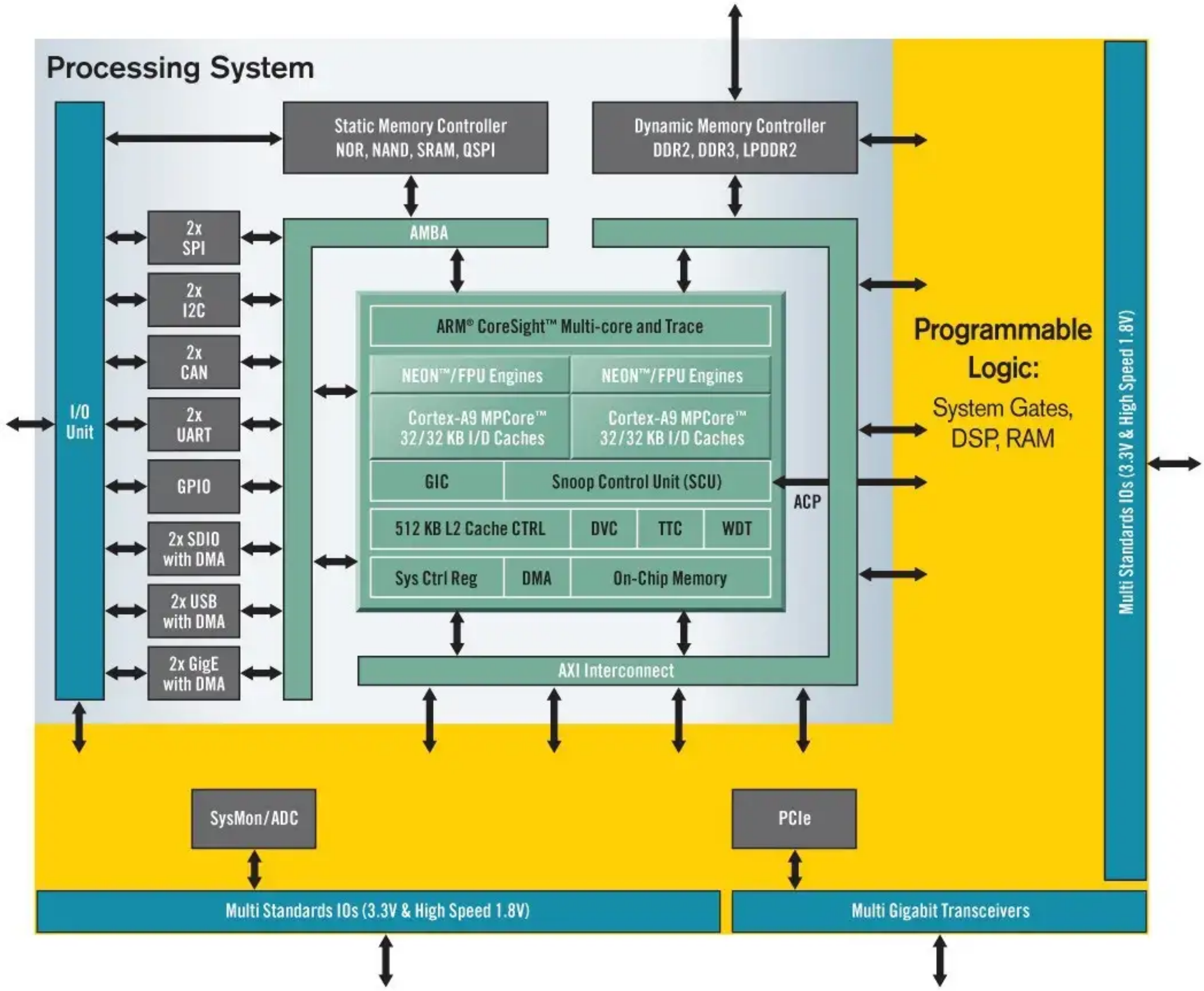


FIGURE 2.5: The architecture of ZYNQ-Z2 (Xilinx 2022a)

Another essential feature of the PYNQ-Z2 is its ability to perform high-speed, low-latency data processing. The FPGA fabric of the PYNQ-Z2 is designed for low-power, high-performance computing, which makes it an ideal platform for various applications such as signal processing, image and video processing, and machine learning. Furthermore, the PYNQ-Z2 supports a wide range of high-speed interfaces such as USB 3.0, Ethernet, and HDMI, allowing easy connectivity to various peripherals and external devices.

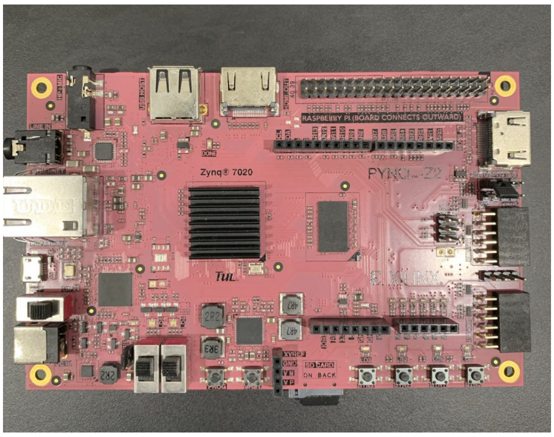


FIGURE 2.6: PYNQ-Z2

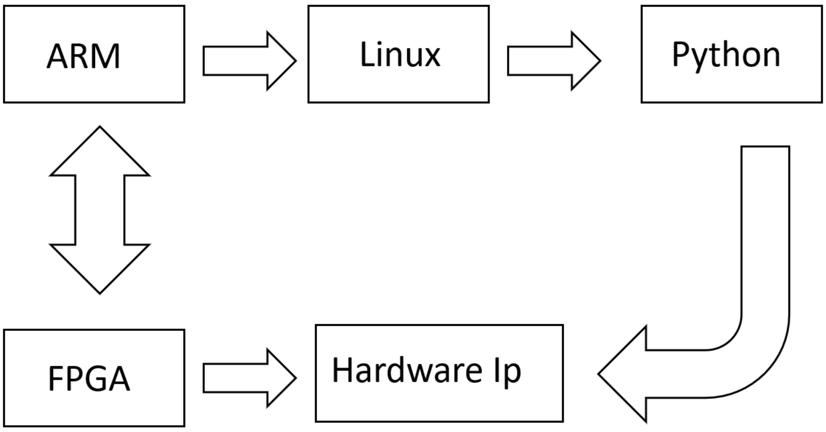


FIGURE 2.7: The Framework of PYNQ-Z2

The PYNQ is divided into four layers: the hardware layer, the Linux kernel, the python software layer, and the application layer. The framework of the PYNQ-Z2 is shown in figure 2.7. The hardware layer has PS and PL, which include the Arm and the FPGA respectively. It can generate the different bitstreams, and dynamically switch different functions of the FPGA through software API. The software is in the PS and it consists of Linux and Python. The application layer consists of Jupyter Notebook based on Python and IPython. Jupyter Notebook provides an environment to record code, run code and view the results, visualize data analysis, and view the output. These features make it a convenient tool for data cleaning, statistical modeling, building and training machine learning models, and visualizing data.

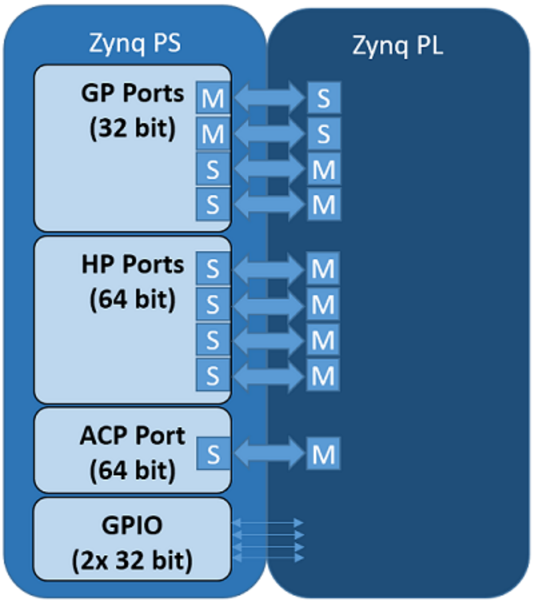


FIGURE 2.8: The Communication between PS and PL (Xilinx 2022b)

The ZYNQ platform has 9 AXI interfaces facilitating communication between the Programmable Logic (PL) and the Processing System (PS). Specifically, the PL side has 4 AXI Master HP (High Performance) ports, 2 AXI GP (General Purpose) ports, 2 AXI Slave GP ports, and 1 AXI Master ACP port. The PS side also features GPIO controllers that are connected to the PL.

To manage data movement between the Zynq PS and PL interfaces, four Zynq classes are utilized: GPIO (General Purpose Input/Output), MIMO (Memory Mapped IO), Xink (Memory allocation), and DMA (Direct Memory Access). The appropriate class for a given IP depends on the Zynq PS interface it is connected to and the interface of the IP. For instance, a Python program running on PYNQ can access an IP that is connected to an AXI Slave via a GP port using MIMO.

2.4.2 FPGA Development

Programming on the FPGA is divided into two methods: High-level Synthesis(HLs) and Hardware Description Language(HDL). The HLS includes C and C++. They provide compilation instructions that inform processors about the program execution process(javaTpoint

2022). The main function of it is instead of coding the Hardware register-transfer level (RTL). HLS is used as a hardware development approach to design digital systems at a higher level of abstraction. HLS can allow users to write source code in software languages and tools in HLS can compile the code, generate Control and Data Flow Graph (CDFG), perform the optimization, and finally output RTL for synthesis and implementation. The design flow by HLS and Vivado is shown in figure 2.9. All steps of the project are finished in three main parts: HLS, Vivado, and PYNQ-Z2 board. The project begins with system analysis and partitioning, followed by writing C++ code to meet the requirements specified in high-level synthesis (HLS). In HLS, the next steps involve simulation and synthesis, culminating in exporting the register transfer level (RTL) code.

Moving into the Vivado environment, the first step is importing the RTL code as an IP core and building a block diagram. The second step involves generating the bitstream and exporting it.

Finally, the PYNQ-Z2 board is used to execute the program. The bitstream is imported to the board, the overlay is loaded, and the program is run using Python code.

Vivado HLS provides designers with practical assistance by increasing the level of abstraction in system design in two methods:

(1) Using C/C++ as the programming language, taking full advantage of the high-level structures available in that language.
(2) Providing more data primitives for designers to use the underlying hardware building blocks (bit vectors, queues, etc.)

These two features help designers solve common protocol system design challenges more easily with Vivado HLS than with RTL. Another major advantage of HLS is the ease of architectural research and simulation.

The HDL is a critical tool for designing digital logic systems and describing digital circuits. Unlike traditional methods of manually drawing each individual component, the HDL allows designers to specify high-level functional behavior, making it particularly well-suited for handling large and complex structures. Two of the most widely used HDLs are Verilog and the

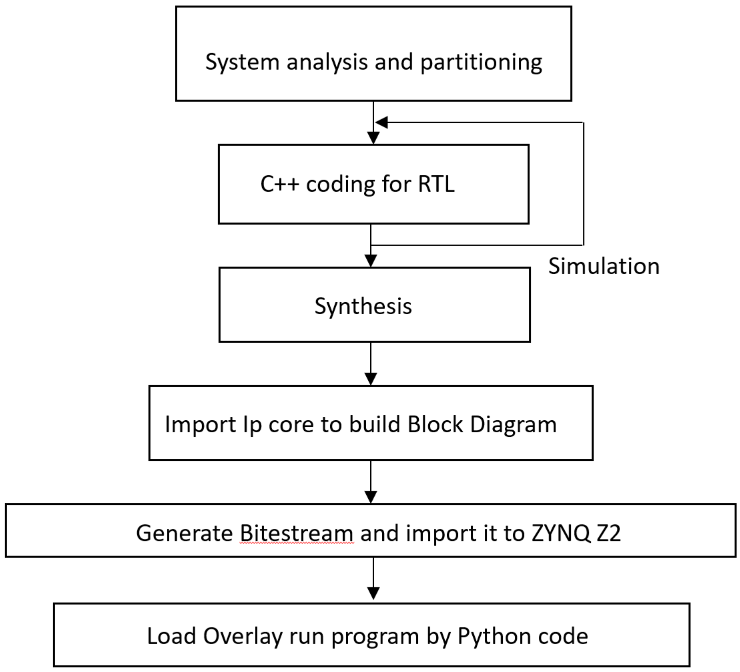


FIGURE 2.9: The design flow in HLS and Vivado

VHSIC Hardware Description Language (VHDL). Both of these languages focus on circuit description and tell the processor how the circuit diagrams are formed and connected.

Of the two, VHDL is a particularly high-level language for circuit design and is considered to be a universal HDL for Electronic Design Automation (EDA) technology. This is due to its precise syntax, clear hierarchical structure, and independence from device design when compared to other hardware description languages. However, despite its many advantages, designing with VHDL can be quite complex. Therefore, it is important to introduce alternate languages to reduce the complexity and streamline the design process. One such language is Verilog, which develops an abstraction level to hide the details of its implementation to simplify the process. This makes it more robust and flexible than other HDLs. Verilog has become one of the most popular HDLs in use today. As an example of the practical applications of HDLs, the development of a Guidance, Navigation, and Control (GNC) system for unmanned aerial vehicles (UAVs) is often implemented on the FPGA using Verilog

(Cadena et al. 2017). This underscores the critical role that HDLs play in the design and implementation of advanced electronic systems.

In FPGAs, because each logic block is independent, it can perform some unique design attributes such as parallel and style and pipelined architecture. Furthermore, the number of gates used for a certain process can also be optimized. All these technologies can be combined with serial processing to enhance the performance of applications.

(1) Serial Processing: Serial processing is a method of designing functions where each clock cycle executes one operation in sequential order. This approach is often utilized as the internal logic of a Finite-State Machine (FSM), where the FSM can only exist in one of a finite number of states at any given time. In FPGA programming, registers are vital resources used to store state variables, and the state transition is determined by a block of combinational logic. A second logic block is then required to produce the output of the FSM. The advantage of using a serial design is that it requires minimal hardware, thereby reducing both the area and power consumption. However, the downside is that the performance of the design is significantly slower.
(2) Parallel Processing: One of the significant advantages of FPGA design is its ability to perform parallel processing. Unlike sequential processing which is commonly used in other processors, FPGA can simultaneously execute multiple tasks or modules at the same time, provided they are not sequentially related. This feature enables FPGA to perform several operations in one clock cycle, resulting in a significant performance improvement, particularly for applications with strict processing time requirements. Although parallel design strategy requires more resources than the sequential processing type, the benefits outweigh the cost in applications that demand high performance.
(3) Pipelining is a powerful technique that enables the seamless execution of multiple operations continuously and efficiently. Unlike traditional approaches where each operation must complete before the next one can start, pipelining allows for overlapping and parallel processing of operations, resulting in faster and more efficient

performance. In the context of FPGA programming, pipelining is a critical design aspect that offers unique advantages for high-performance computing. It is essentially a hybrid approach that combines the benefits of both serial and parallel processing methods. Multiple tasks can be designed in parallel, and each task can receive input and output results in sequential logic, thereby maximizing efficiency and minimizing latency. Overall, pipelining represents a critical design element in modern FPGA programming, enabling developers to achieve unprecedented levels of performance and efficiency in a wide range of applications. By embracing this approach, researchers and practitioners can unlock the full potential of FPGA technology and drive innovation in a variety of fields, from aerospace engineering to medical imaging and beyond.

2.4.3 Kalman Filter and Neural Networks on FPGA

Satellite attitude estimation algorithms rely heavily on the accuracy and computational efficiency of the Kalman Filter. Unfortunately, the implementation of complex algorithms like the Kalman Filter on FPGA can be challenging due to hardware design complexities. However, recent advancements in FPGA technology have enabled the implementation of novel attitude-solving algorithms with IMU, such as the INS algorithm presented by Zhu et al. (Zhu et al. 2022). To further improve the performance of integrated INS-GPS systems, Agarwal et al. (Agarwal et al. 2009) proposed an improved design and fabrication approach, where the digital signal processor (DSP) is utilized for inertial navigation and Kalman filter computations. This approach reduces the total chip count, resulting in a compact system, which can be further implemented on the FPGA by creating a universal asynchronous receiver transmitter (UART) and dual port random axis memory (DPRAM). Another approach to address the scalability problem of IMU array sensor fusion is presented in Waheed et al. (Waheed and Elfadel 2018). The authors designed a specialized vector processor that achieves real-time, high-throughput IMU sensor array fusion based on the KF paradigm. The proposed design offers a more efficient and scalable solution compared to conventional FPGA-based implementations. Bhogadi et al. (Bhogadi et al. 2015) proposed an improved design of a loosely coupled GPS/INS integrated system that utilizes the MicroBlaze processor on the Virtex-6 FPGA for inertial navigation and Kalman filter computations. The approach offers a more efficient and scalable solution compared to conventional FPGA-based implementations, providing a promising solution for future INS-GPS integrated systems.

The Extended Kalman filter (EKF) is a critical algorithm in embedded systems. In (Carletta et al. 2020), the authors achieved Cubesat attitude estimation using a three-axis magnetometer and implemented the EKF algorithm on an FPGA. They rearranged complex matrix operations into the form of the Faddeev algorithm to enhance computational efficiency. Similarly, (Weimer et al. 2015) proposed a particle filter to estimate attitude and velocity on a small unmanned aerial system using GPS, gyro, and accelerometer measurements. However, particle filters have a large computational load, and the authors implemented them on an FPGA using pipeline techniques to solve this problem. In (Jew et al. 2010), the authors presented the implementation of real-time algorithms for an aided INS on a fabric processor-based FPGA. This approach allowed for the development of a hard real-time computational architecture tailored to the specific INS requirements while still preserving flexibility. Furthermore, (Xu et al. 2010) combined the Kalman filter and the least squares support vector machine (LS-SVM) to aid the GPS/INS integrated system, and they implemented the system on an FPGA. In (Akgün et al. 2020), various system identification techniques, such as the Kalman filter (EKF), recursive least square (RLS), and least mean square (LMS) filters, were used to estimate the parameters of linear (DC motor) and nonlinear systems (inverted pendulum and adaptive polynomial models). The authors used FPGAs for rapid prototyping, real-time processing, and high computational programs.

Another variant, the unscented Kalman filter (UKF) also has superior performance (Crassidis et al. 2007). (Soh and Wu 2017) presents a hardware/software co-design of the unscented Kalman filter with a five-stage pipeline on FPGA.(Soh and Wu 2016) also explored the feasibility of an FPGA-based UKF for a singular nanosatellite, as well as a generic, more portable variant with parallelized datapaths. The parallelism can increase the overall data throughput of the system by the hardware pipelines and increase the algorithm running speed (Xue et al. 2020). Hardware pipelines are also often used to accelerate complex algorithms that can be implemented in hardware and have a certain amount of data independence, such as encryption (Wang and Ha 2013) or image processing (Draper et al. 2003).

The Neural Network is also an important algorithm in satellite attitude algorithms, especially for the image processing in Star tracker. Numerous well-known neural networks are crucial to image processing, such as Convolutional Neural Networks (CNN), Deep Learning Neural Networks (DNN), Quantum Neural Networks (QNN), Backpropagation Neural Networks (Bp), and Support Vector Machines (SVM). (Liu and Wu 2022) developed a general-purpose feature detection hardware architecture based on the Speeded-Up Robust Features (SURF) algorithm and presents an FPGA-based implementation of a modified SURF algorithm. (Zhao et al. 2019) designed an embedded FPGA image recognition system on Convolutional Neural Network (CNN). By using parallelism and pipeline, and parallelization to realize multi-depth convolution operations. (Guo et al. 2017c) presented an overview of different neural network inference accelerators based on FPGA and summarizes the main techniques used, including CNN, RNN, and generative adversarial network(GAN). (Wang and Luo 2022) proposed a special processor for keypoint detection of aircraft that was based on FPGA with DNN to accelerate the detection process. The design used HLS high-level synthesis, fixed-point quantization, on-chip data buffering, and FIFO (first in first out) optimization methods.

The development of voice recognition systems on FPGA also has been a significant area of research in recent years. Recurrent neural networks (RNN) and their variant, Long short-term memory (LSTM), have emerged as the dominant models for speech recognition. Early research efforts, such as those by Ferreira et al. (2016), focused on implementing LSTM networks on FPGA hardware. This work sparked interest in exploring the hardware implementation of LSTM, rather than just relying on software implementation. Building upon this work, Zhang et al. (2017) developed an improved system that incorporated sparse LSTM layers, which occupied fewer resources and achieved better performance. In addition, Han et al. (2017) proposed an efficient speech recognition engine called ESE that is based on FPGA and utilizes sparse LSTM. The ESE engine is capable of compressing the size of the LSTM model by a factor of 20, resulting in a highly efficient and compact system.

2.5 Summary

This chapter presents a comprehensive review of various fields related to satellites at an advanced level. Specifically, it explores the satellite attitude determination system, encompassing INS, CNS , GNSS, and multi-system fusion methods. Additionally, it examines the fault tolerance of sensors employed in satellites, relative attitude estimation techniques, fundamental concepts of FPGAs, and methods for implementing FPGA-based satellite attitude estimation.

Numerous innovative algorithms for satellite attitude estimation are introduced, leveraging IMU, star tracker, and other sensors. Notable algorithms include the QUEST algorithm, FQA algorithm, O2OQ algorithm, SMIT algorithm, and LAMBDA algorithm. Furthermore, to ensure fault tolerance for each sensor on the satellite, several FDIR algorithms are presented, such as AUKF, FTFKF, and a hardware redundancy scheme.

The comprehension of FPGAs is fundamental for implementing these algorithms on the hardware, and this section also provides numerous optimization schemes aimed at reducing latency while maintaining optimal resource utilization.

CHAPTER 3

Attitude Determination System

To accurately describe attitude information, it is essential to possess knowledge of attitude representations and reference frames. Additionally, understanding the sensor model and kinematic model is crucial for precise attitude estimation. This chapter aims to introduce various reference frames and attitude representations, as well as important sensor models used in satellites.

The ability to accurately estimate the attitude of a satellite is vital for its proper functioning, and requires a solid understanding of various aspects. Reference frames, which provide a means for expressing the orientation of an object, are crucial for describing the attitude of a satellite. Different reference frames, such as the Earth-centered inertial frame, the Earth-centered Earth-fixed frame, and the body frame, are discussed in this chapter.

Furthermore, various attitude representations, such as Euler angles, quaternions, and rotation matrices, are introduced. Each representation has its own strengths and weaknesses, and the choice of representation often depends on the specific application.

In addition to reference frames and attitude representations, knowledge of sensor models is essential for accurate attitude estimation. Different sensor models, such as the gyro, magnetometer, and sun sensor, are discussed, along with their advantages and disadvantages.

Overall, this chapter provides a comprehensive overview of the various components necessary for precise attitude estimation in satellites. A thorough understanding of reference frames, attitude representations, and sensor models is essential for accurate attitude determination and the proper functioning of satellites.

3.1 Reference Frame

This section presents the definitions of various reference frames. These frames are essential as they yield distinct measurements and models. To ensure clarity and consistency in the analysis, it is crucial to establish unique and unambiguous definitions for the rotations between frames. The aim is to ensure that the resulting data and models are reliable and accurate.

3.1.1 The Earth Centered Interial

The Earth Centered Interial (ECI) is located in the center of the earth and fixed towards the stars. From figure 3.1(Popescu 2014), The X-axis corresponds to the vernal equinox and the Z-axis points towards the north celestial pole. The Y axis completes a right-hand Cartesian coordinate system. the Properties of the ECI reference frame are: first, it does not rotate with the Earth's rotation. Second, the ECI reference frame can be used as the inertial coordinate system for the spacecraft near the Earth.

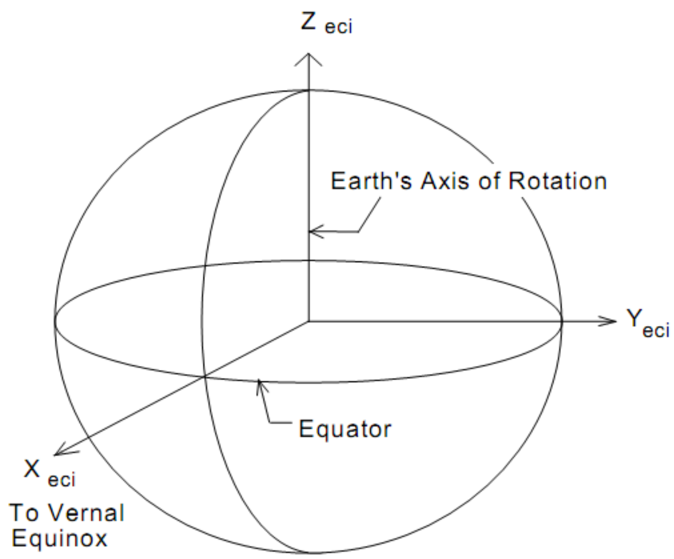


FIGURE 3.1: ECI reference frame (Popescu 2014)

3.1.2 Earth-centered Earth-fixed reference frame

Figure 3.2 depicts the Earth-centered Earth-fixed (ECEF) reference frame, which serves as a crucial coordinate system in modern geodesy and satellite navigation. In this reference frame, the Z axis points towards the north pole, while the X axis lies in the plane of the Greenwich meridian. Completing the right-hand set, the Y axis forms a perpendicular axis to both the X and Z axes.

One of the key properties of the ECEF reference frame is that it is fixed with respect to the Earth and rotates along with the planet. This feature distinguishes the ECEF reference frame from the Earth-centered inertial (ECI) reference frame, which remains fixed with respect to distant stars. Such a distinction is of paramount importance in many applications that require accurate positioning and tracking of objects in space.

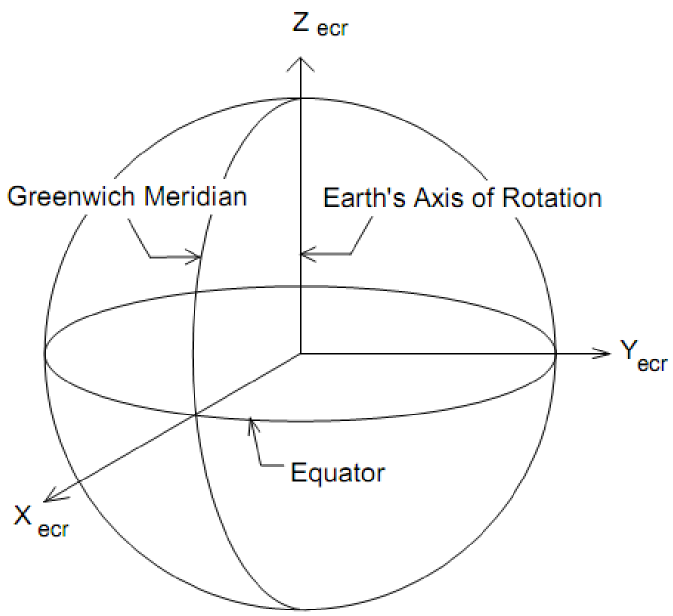


FIGURE 3.2: ECEF reference frame (Popescu 2014)

3.1.3 North East Down

The ECI and ECEF coordinate frames both utilize the center of the Earth as their origin point, presenting significant challenges for navigation implementation. Alternatively, the North East

Down (NED) frame establishes its origin at the Earth's surface and aligns with the carrier's orientation. This frame is derived by fitting the local ellipsoid shape onto a tangent plane at the current location, resulting in a coordinate system that is fixed relative to a point on the Earth's surface. As illustrated in Figure 3.3, the X axis points towards the North, the Z axis points downwards along the local ellipsoid normal, and the Y axis completes the right-hand rule by pointing East. Notably, the NED frame rotates with the Earth's rotation.

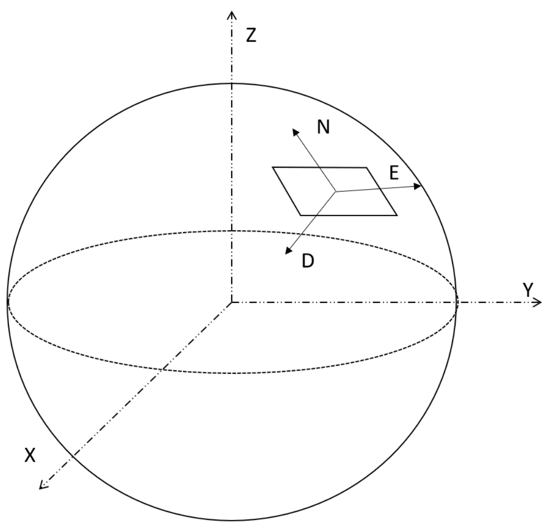


FIGURE 3.3: Representation of the NED reference frame

3.2 Attitude Representation

3.2.1 Rotation Matrix

The rotation matrix is one of the most popular methods to describe the attitude. The rotation matrix must be orthogonal (both frames with orthogonal axes) and orthonormal (all axes are orthonormal). It is a 3×3 matrix to describe the rotation of three axes (x, y, and z axes). The rotation matrix can be described as:

$$R_b^a = \{\overline{a_i} \bullet \overline{b_i}\} \tag{3.1}$$

It means the rotation matrix from a to b, the elements $r_{ij} = \overline{a_i} \bullet \overline{b_j}$ of the rotation matrix R_b^a are called the direction matrix. A simple rotation is a rotation about a fixed axis. There are three simple rotations when the vectors are expressed in Cartesian coordinate frames:

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \tag{3.2}$$

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{3.3}$$

$$R_z(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.4}$$

They represent three angles $\phi, \theta, and \psi$ rotation about three axis x, y z, respectively. The Rotation matrix has some important properties, the first is the rotation matrix is always orthogonal and it satisfies:

$$R_a^b = {R_a^b}^{-1} = {R_b^a}^T \tag{3.5}$$

There are two definitions of the rotation matrix. First is a coordinate transformation matrix when transforming the coordinate vector v_b to v_a. secondly as a rotation matrix when rotating the coordinate vector p^a to the coordinate vector q^a where $q^b = p^a$ by:

$$q^a = R_b^a p^a \tag{3.6}$$

The third is that the rotation matrix can describe a complicated rotation:

$$R_d^a = R_b^a R_c^b R_d^c \tag{3.7}$$

However, using the rotation matrix has some disadvantagesEgeland and Gravdahl 2002, the first is it is very difficult to interpolate rotation between two orientations, and the second is it is hard to keep the matrix orthonormalized.

3.2.2 Euler Angles

The Euler angles are also commonly used to describe attitude. It can be presented as $\Theta = [\phi\ \theta\ \psi]^T$, representing roll pitch yaw respectively. When the attitude of the satellite is determined by the rotation between the body frame and the orbit frame. The rotation from the body frame to orbit frame may be considered as a composite rotation consisting of a rotation ψ about the z_b, then a rotation θ about the current (rotated) y-axis, and finally a rotation ϕ about the current x-axis. The details are illustrated in figure 3.4(Sunde 2005). The resulting rotation matrix is:

$$R_b^o(\Theta) = \begin{bmatrix} \cos\theta\cos\psi & \sin\theta\sin\phi\sin\psi - \cos\phi\sin\psi & \sin\theta\cos\phi\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\theta\sin\phi\cos\psi + \cos\phi\cos\psi & \sin\theta\cos\phi\sin\psi - \sin\phi\cos\psi \\ \sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix} \tag{3.8}$$

It can be found that the matrix is singular at $\pm 90°$ from (3.8). The singularity is a not avoided problem When using the Euler angles.

3.2.3 Quaternion

The shortages of the rotation matrix and Euler angles are not intuitive, and exist singularity problems that could lead to the gimbal-lock problem at 90 degrees, respectively. The quaternion can also represent the attitudes in the b frame (body frame) relative to the r frame (reference frame). Using quaternion to represent the attitude has several advantages: First, it can Solve the Gimbal Lock problem. Second, the quaternion only needs to store four floating point numbers, which is lighter than the rotation matrix, and Euler angles. Third, When using some operations such as inversion or concatenation, the quaternion are more efficient than the rotation matrix and Euler angles. It can be described as:

$$q = \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix} \tag{3.9}$$

The quaternion consists of a scalar part q_0 and a vector part $e = [q_1\ q_2\ q_3]^T$. The conjugate of

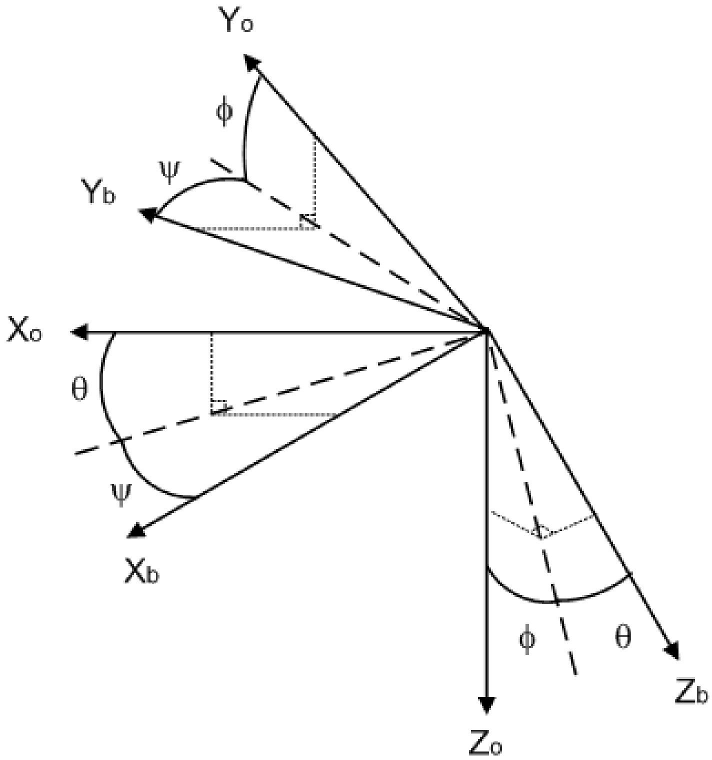


FIGURE 3.4: Attitude representation by Euler angles (Sunde 2005)

the quaternion q can be written as q^*:

$$q^* = \begin{bmatrix} q_0 & -q_1 & -q_2 - q_3 \end{bmatrix} \tag{3.10}$$

Quaternion should satisfy $q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1$. In addition, the multiplication of two quaternions $p \otimes q$ are:

$$p \otimes q = \begin{bmatrix} p_0q_0 - p_1q_1 - p_2q_2 - p_3q_3 \\ p_1q_0 + p_0q_1 + p_2q_3 - p_3q_2 \\ p_2q_0 + p_0q_2 + p_3q_1 - p_1q_3 \\ p_3q_0 + p_0q_3 + p_1q_2 - p_2q_1 \end{bmatrix} \tag{3.11}$$

The Direction Cosine Matrix (DCM) is also can be represented by quaternion:

$$C_n^b = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 + q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \tag{3.12}$$

Similarly, the Euler angles can be expressed in quaternations, and the sequence is ZYX and the coordinate system is NED (North-EAST-Down):

$$\begin{bmatrix} \theta \\ \phi \\ \psi \end{bmatrix} = \begin{bmatrix} atan2((2q_2q_3 + 2q_1q_0), (q_0^2 - q_1^2 - q_2^2 + q_3^2)) \\ asin(2q_0q_2 - 2q_1q_3) \\ atan2((2q_1q_2 + 2q_0q_3), (q_0^2 + q_1^2 - q_2^2 - q_3^2)) \end{bmatrix} \tag{3.13}$$

Therefore, there are 4 main advantages: No singularities, No trigonometric functions, a Convenient product rule for successive rotations, and the Kinematic equation of attitude is linear. On the other hand, the disadvantage is: No obvious physical interpretation.

3.2.4 Modified Rodriguez Parameter

Sometimes, using the 4-element quaternion vector causes ambiguity and unwinding phenomena(Song and Cai 2012). Therefore, the Modified Rodriguez Parameter (MRP), which were first created in 1962 by T. F Wiener(Markley and Crassidis 2014), is used to represent the satellite attitude deriving from the Eulers principle rotation theorem:

$$p = \tanh \theta / 4n \tag{3.14}$$

where, q, n,and θ are three-element vectors of MRP. They are the unit vector, Euler's rotation axes, and rotation angles, respectively(Forbes 2015). The vector of Modified Rodriguez parameters p is related to the quaternion q by the following relation:

$$p = \frac{q_{1:3}}{1 + q_0} \tag{3.15}$$

which has the inverse

$$q = \frac{1}{1 + ||p||^2} \begin{bmatrix} 2p \\ 1 - ||p||^2 \end{bmatrix} \tag{3.16}$$

However, there is a singularity when θ between $[0, 2\pi]$, and this singularity can be avoided by using the MRP shadow set (p^s):

$$p^s = -p/p^T p \tag{3.17}$$

These is the main disadvantages of the MRP, and the advantages of MPR are: No redundant parameters, No trigonometric functions, and a Convenient product rule for successive rotations.

Overall, the advantages (+) and disadvantages (-) of different attitude representations are summarized in Table 3.1:

TABLE 3.1: Characteristics of different attitude representations

Attitude representations	Characteristics
Euler Angles	(+) No redundant parameters (+) Clear physical interpretation (-) Presence of trigonometric functions (-) Singularity when $\theta = \pm\pi/2$ (-) No convenient product rule for successive rotations (-) The kinematic equation of attitude is nonlinear
DCM	(+) No singularities (+) No trigonometric functions (+) Convenient product rule for successive rotations (+) The kinematic equation of attitude is linear (-) complicated computation
quaternion	(+) No singularities (+) No trigonometric functions (+) Convenient product rule for successive rotations (+) The kinematic equation of attitude is linear (-) No obvious physical interpretation
MRP	(+) No redundant parameters (+) No trigonometric functions (+) Convenient product rule for successive rotations (-) No obvious physical interpretation Singularity

3.3 Sensor Model

3.3.1 IMU

3.3.1.1 Gyroscope

The gyroscope outputs the angular rate of the body frame with respect to the inertial frame. The model of the gyroscope can be described as follows:

$$\omega(t) = \omega_t(t) + b_g(t) + \eta_g(t) \tag{3.18}$$

Where η_g is the zero mean Gaussian white noise with a standard deviation σ_w and a covariance Q_w, ω_t is the true angular velocity, $b_g = [b_{gx}\ b_{gy}\ b_{gz}]^T$ is the gyro bias vector that can be described as:

$$\frac{d}{dt} b_g(t) = -b_g(t)/\tau + \eta_b \tag{3.19}$$

Where η_b is a zero-mean Gaussian white noise with standard deviation σ_b and a covariance Q_b, and τ is the sensor time constant. It is assumed that noise signals in the measurement and the process are uncorrelated

$$E = [\eta_g \eta_b^T] = 0 \tag{3.20}$$

3.3.1.2 Acceleometer

The output of the accelerometer are a_x, a_y, and a_z, which represent the acceleration of the X, Y, and Z axis, respectively. When the accelerometer is placed horizontally, the Z-axis is vertically up, the output of the Z-axis is 1g (g is the acceleration of gravity), and the output of the X-axis and Y-axis are 0. Therefore, the output of the accelerometer is $[0\ 0\ g]$. The rotation matrix of X, Y, and Z axis are M_x, M_y, and M_z, respectively. The Euler angles $\Theta = [\phi\ \theta\ \psi]^T$ represent the rotation angles about the X Y Z axis, and the rotation sequence is Z-Y-X. It follows:

$$\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = M_x \bullet M_y \bullet M_z \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \tag{3.21}$$

In this Equation:

$$M_x \bullet M_y \bullet M_z = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \bullet \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \bullet \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & cos\psi & 0 \\ 0 & 0 & 1` \end{bmatrix} \tag{3.22}$$

We can get the following:

$$\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \begin{bmatrix} \sin\theta \\ \cos\theta sin\phi \\ \cos\theta\cos\phi \end{bmatrix} \bullet g \tag{3.23}$$

Therefore, the roll angle and the pitch angle are:

$$roll = \arctan\frac{a_y}{g} \tag{3.24}$$

$$pitch = -\arctan\frac{a_x}{\sqrt{a_y^2 + a_z^2}} \tag{3.25}$$

The model of the accelerometer is shown:

$$a(t) = A * a_t(t) + \eta_a(t) \tag{3.26}$$

Where a_t is the true angular velocity, η_a is the zero mean Gaussian white noise, and A represents the direction cosine matrix.

The major drawback of the accelerometer is that it cannot obtain the yaw angle, which means It cannot be utilized uniquely. Accelerometers are often used with other sensors, such as magnetometers, gyroscopes, star trackers, to get a more accurate attitude.

3.3.1.3 Magnetometer

The magnetic field vector varies with the orbital parameters as the spacecraft moves throughout its orbit. When those parameters are known, the magnetic field tensor vector that influences the satellite can be presented analytically as a function of time Sekhavat et al. 2007:

$$\begin{aligned}
H_1(t) &= \frac{M_e}{r_0^3}(cos(\omega_0 t)[cos(\epsilon)sin(i) - sin(\epsilon)cos(i)cos(\omega_e)t)] - sin(\omega_0 t)sin(\epsilon)sin(\omega_e t)) \\
H_2(t) &= -\frac{M_e}{r_0^3}[cos(\epsilon)cos(i) + sin(\epsilon)sin(i)cos(\omega_e t))] \\
H_3(t) &= \frac{2M_e}{r_0^3}(sin(\omega_0 t)[cos(\epsilon)sin(i) - sin(\epsilon)cos(i)cos(\omega_e t)] - 2sin(\omega_0 t)sin(\epsilon)sin(\omega_e t))
\end{aligned} \tag{3.27}$$

Where ω_0 is the angular velocity of the orbit with respect to the inertial frame, which can be described as $\omega_0 = (u/r_0^3)^{\frac{1}{2}}$ where u is the Earth Gravitational constant, M_e is the distance between the center of mass of the satellite and Earth. The model of the magnetometer output can be presented as:

$$\begin{bmatrix} H_{mx}(t) \\ H_{my}(t) \\ H_{mz}(t) \end{bmatrix} = A * \begin{bmatrix} H_1(t) \\ H_2(t) \\ H_3(t) \end{bmatrix} + \eta_m \tag{3.28}$$

Where, $H_m = [H_{mx}\ H_{my}\ H_{mz}]^T$, it is the magnetic field vector from the magnetometer, and $H = H_1\ H_2\ H_3$ is the magnetic field vector in the orbit frame. The η_m is the zero-mean Gaussian white noise, and A represents the direction cosine matrix.

3.3.2 Star Tracker

Star tracker is an important sensor of celestial navigation systems. It uses the star as the reference source for attitude measurement and can output the vector direction in star-tracker coordinates, providing high-precision measurements for spacecraft attitude control and astronomical navigation systems (Eisenman and Liebe 1996).

Nowdays, the CCD star tracker is mature and widely used in the aerospace industry. The star tracker includes the hood, optical system, photoelectric conversion circuit (CCD components, timing circuit, driver circuit, acquisition, and amplification circuit), control circuit (cooler control, operating circuit), secondary power supply, a data processing module (star map preprocessing, star mass extraction, star map recognition and attitude estimation) and external interface. The structure of the CCD star tracker is shown in figure 3.5

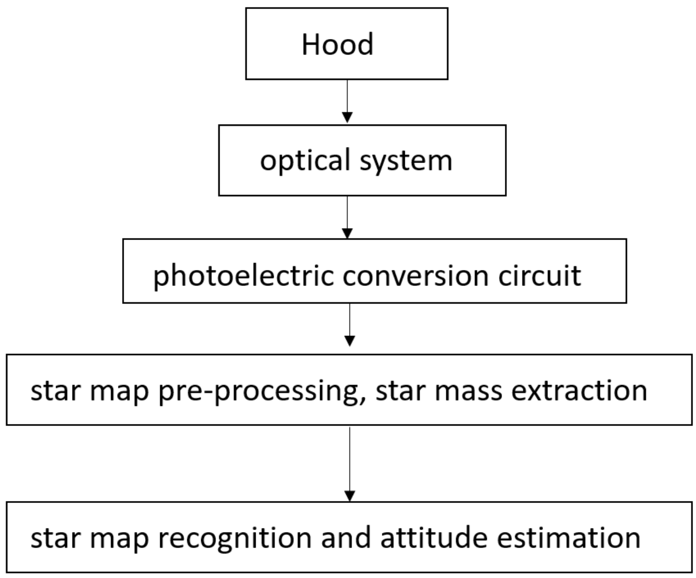


FIGURE 3.5: The basic structure of the CCD Star tracker

The theory of the Star tracker is shown in figure 3.6.$O_sX_SY_SZ_S$ is the star tracker coordinate system. $O\mu\omega\nu$ is the Image plane coordinate system. The Y_s is the same as the ω, and they have the same direction with the O_sO. The O_sO is the focal length of the optical lens f. P is the image of the star on the $O\mu\omega\nu$ plan, and the P_μ is the projection of PO_s on the $O\mu\omega\nu$ plan. The angle between the PO_s and O_sO is α. The angle between PO_s and $P_\mu O_s$ is δ. From 3.6 it can found:

$$\tan\alpha = \frac{OP_\mu}{f} \tag{3.29}$$

$$\tan\delta = \frac{\nu}{f/\cos\alpha} \tag{3.30}$$

The unit vector of PO_S is po, and it can be described in the star tracker coordinate system:

$$po = \begin{bmatrix} x_s \\ y_s \\ z_s \end{bmatrix} = \begin{bmatrix} -\sin\alpha\cos\delta \\ \cos\alpha\cos\delta \\ -\sin\delta \end{bmatrix} + V_s \tag{3.31}$$

where V_s is the measurement noise vector of the star tracker

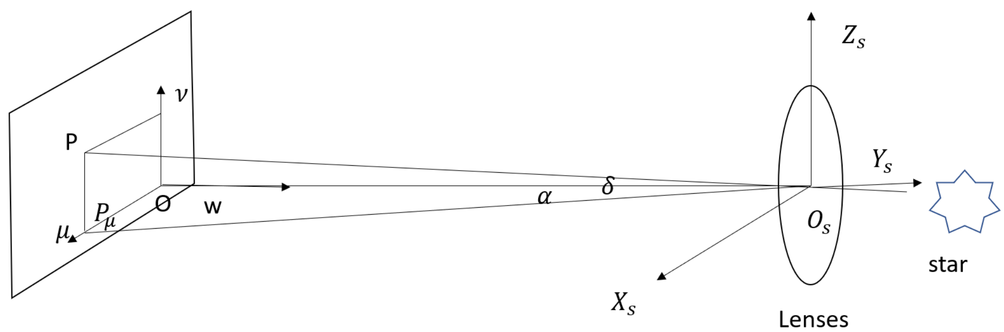


FIGURE 3.6: The Star tracker measurement diagram

3.4 Kinematic Equation

This part presents the deduction of the satellite's mathematical model. This model is the basis of the Kalman Filter and the nonlinear observer.

Quaternions can represent satellite attitudes without singularities as $q = [q_0\ q_1\ q_2\ q_3]^T$, the convention used for this work has the scalar q_0 and a three-axis vector as $e = [q_1\ q_2\ q_3]^T$. The attitude matrix A_q is calculated as a quadratic function of q:

$$A_q(q) = (q_0^2 - ||e||^2)I_{3\times3} + 2ee^T - 2q_0[e\times] \tag{3.32}$$

Where $I_{3\times3}$ is a 3×3 matrix and the $[e\times]$ is defined as the cross matrix:

$$[e\times] = \begin{bmatrix} 0 & -q_3 & q_2 \\ q_3 & 0 & -q_1 \\ -q_2 & q_1 & 0 \end{bmatrix} \tag{3.33}$$

The quaternion kinematics model can then be defined as:

$$\dot{q} = \frac{1}{2}\Xi(q)\omega = \frac{1}{2}\Omega(\omega)q \tag{3.34}$$

Where $\omega = [\omega_x\ \omega_y\ \omega_z]^T$ is the vector of the angular velocity of the body frame with respect to the inertial frame, the $\Omega(\omega)$ is the skew matrix:

$$\Omega(\omega) = \begin{bmatrix} -[\omega\times] & \omega \\ \omega^T & 0 \end{bmatrix} \tag{3.35}$$

3.5 Low Earth Orbit Dynamic Equation

The altitude of the Earth orbit is in the range of 160 km to 2000 km from sea level. The dynamics of the satellite is based on Newton's law of gravitation. However, the actual motion of the satellite deviates from the motion modeled because of the irregular shape of the Earth and the non-uniform density. To solve this problem, a Legendre polynomial with latitude and longitude as parameters is used to enlarge the Earth's gravitational potential (Kaula 2013). The state vector of the satellite motion can be described as:

$$X_{sat} = \begin{bmatrix} t \\ v \end{bmatrix} = \begin{bmatrix} x & y & z & v_x & v_y & v_z \end{bmatrix}^T \tag{3.36}$$

Where $r = [x\ y\ z]^T$ and $v = [v_x\ v_y\ v_z]^T$ represent the position and velocity of the satellite in the ECI frame, respectively.

3.6 Summary

This chapter provides a comprehensive overview of the fundamental concepts in satellite attitude determination systems. It covers essential aspects such as attitude representation, including quaternions and Euler angles, the navigation reference frame, models of various sensors, as well as kinematic and dynamic equations. This foundational knowledge serves as a basis for understanding the subsequent sections.

CHAPTER 4

An Observation Model from Linear Interpolation for Quaternion-based Attitude Estimation

---

In this chapter, the proposed observation model from linear interpolation for quaternion-based attitude estimation is described. The work was first presented by (Chen et al. 2023a). First, an introduction is given a brief flowchart of this algorithm. The following sections provide each module separately, including the QUEST algorithm, FQA algorithm, and Kalman filter. Finally, the simulation and experimental results are given to verify the validity of the algorithm.

4.1 Introduction

This paper aims to solve multiple problems associated with attitude estimation: the complexity of Kalman Filter calculations in the Attitude and Heading Reference System (AHRS), the interference sensitivity of Magnetic Angular Rate and Gravity (MARG) sensors, and the low accuracy of the Factored Quaternion Algorithm (FQA). It presents a two-layer Linear Kalman Filter using MARG sensors to obtain attitude estimation in quaternions. Firstly, data from a triaxial accelerometer and magnetometer is processed by a novel algorithm, which fuses the Quaternion Estimator (QUEST) algorithm and FQA by the Linear interpolation (LERP) to obtain an observation model. Secondly, the process model in the two-layer Kalman Filter was obtained by using LERP to fuse an optimum quaternion obtained from a gyroscope and FQA. The LERP can eliminate gyro bias drift, and integral error, and compensate for unexpected conditions, such as fast rotation and temporary strong magnetic disturbances. The proposed algorithm presents higher accuracy and lower computational load than QUEST or FQA used with a Kalman Filter. The performance of the proposed algorithm is verified through simulations and experiments.

4.2 Kalman Filter Design

The gyroscope is a basic sensor in the satellite attitude estimation system. The relationship between the angular velocity ω and quaternion derivation $\dot{q}$ is:

$$\dot{q} = \frac{1}{2} \times q \otimes \omega \tag{4.1}$$

Where $\otimes$ represents quaternion multiplication. The quaternions can be obtained by integrating the quaternion derivative $\dot{q}$ with the constant sampling time. However, the quaternions from the integrator are not unit quaternions, which cannot satisfy the attitude calculation requirement. Thus, it is necessary to normalize the resultant quaternions in the last step of the updating procedure. The last output of the integration is called the approximation quaternion q_{ap}. Equation (4.1)is employed as the process model in the last step. The flow chart of the process model is shown in figure 4.1, in which ω and ω^n denote the angular velocity and the measurement noise, respectively. The two-layer Kalman filter will be discussed in the following section.

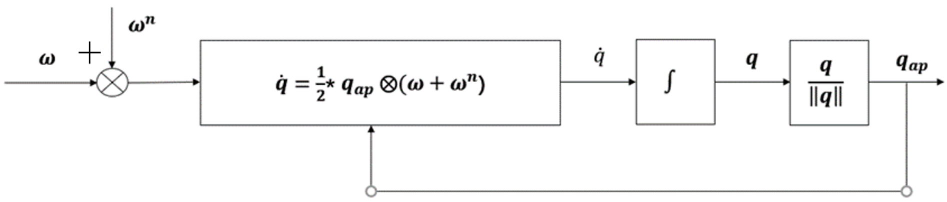


FIGURE 4.1: Kalman filter process model of a gyroscope

The Kalman Filter state vector is typically a 7-D vector (a 4-D quaternion vector with 3-D angular velocity or 3-D gyro bias drift). Hence, the classical Kalman filter in most real-time attitude estimation applications is more than four dimensions. To reduce the dimensions, some researchers also have combined a two-layer Kalman Filter with other algorithms, such as FQA as shown in Fig. 2, Gradient Descent Algorithm, and other algorithms (Yun et al. 2005; Lee and Park 2009; Wang et al. 2015a; Feng et al. 2017; Valenti et al. 2015; Seo et al. 2011).

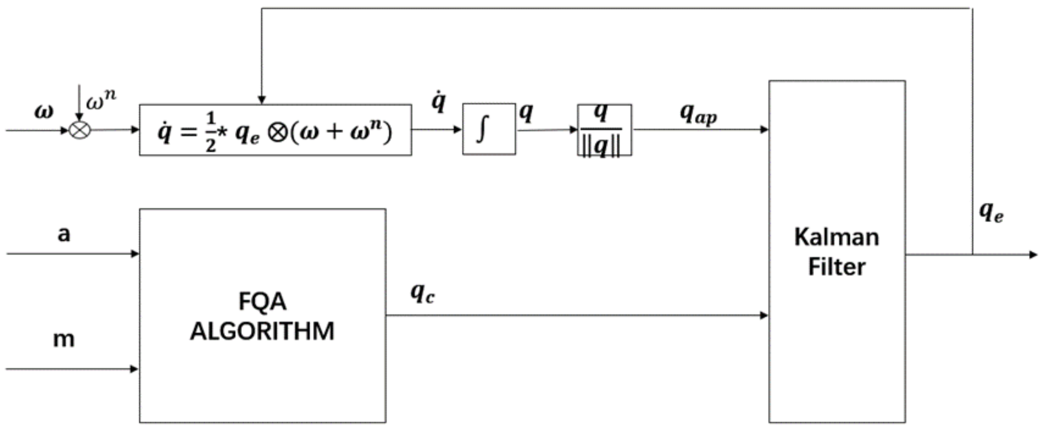


FIGURE 4.2: Block diagram of combing a two-layer Kalman filter and FQA

In figure 4.2, a and m are the acceleration and local magnetic field vectors, respectively and are input into the FQA to obtain the computed quaternion q_c. The angular velocity from the gyroscope can be integrated at the same time to yield the approximation quaternion q_ap. Then the Kalman Filter fuses the q_c with the q_ap to get the estimated quaternion q_e. The key distinction between this scheme and the others is that both the state and measurement vector are four-dimensional. The low-dimensional matrix has fewer processing demands than a high-dimensional matrix, reducing the running time. This technique, however, has limited performance in several aspects. The first is the noise of each sensor, e.g. the magnetometer output is easily contaminated by the ferromagnetic particles, while the accelerometer is affected by the linear motion; Second, when the angular rate is integrated, the measurement errors are likewise integrated, resulting in infinite attitude drift errors; Third, the observation model q_c can be produced by different algorithms such as FQA, QUEST, and Gradient Descent algorithm, which have their own shortages. For example, the low accuracy of the FQA the low efficiency of QUEST, and the weak dynamics performance of the Gradient Descent algorithm. This paper introduces a new observation model to solve these problems. More text.

4.3 New Two-Layer Kalman Filter

4.3.1 Traditional Observation Model

4.3.1.1 QUEST Algorithm

The deterministic algorithm is always popular in attitude determination, including the TRIAD algorithm, q-Euler algorithm, and QUEST algorithm (Liu et al. 2012; Shuster and Oh 1981). In (Wahba 1965), Wahba presented the famous Wahba problem to find the orthogonal matrix A, which satisfies:

$$A\hat{V_i} = \hat{W_i}(i = 1.......n) \tag{4.2}$$

Where $\hat{V_i}$ represents a set of unit reference vectors, and $\hat{W_i}$is a set of unit observation vectors. To find an orthogonal matrix, Wahba minimizes a loss function:

$$L(A) = \frac{1}{2}\sum_{i=1}^{n} a_i \left|\hat{W_i} - A\hat{V_i}\right|^2 \tag{4.3}$$

Where a_i are non-negative weights. At the same time, the gain function can be defined as:

$$g(A) = 1 - L(A) = \sum_{i=1}^{n} a_i \hat{W_i}^T A\hat{V_i} = tr[AB^T] \tag{4.4}$$

The gain function$g(A)$ is the maximum when the loss function $L(A)$ is the minimum. The goal is to find the suitable attitude matrix A, i.e. finding the maximum number of gain function g(A). In (8), the tr is the trace operation, and B is the attitude profile matrix, which is defined by:

$$B = \sum_{i=1}^{n} a_i \hat{W_i} A\hat{V_i}^T \tag{4.5}$$

The attitude matrix A can be given in terms of quaternion q:

$$A(q) = (q_0^2 e^T e)I + 2ee^T + 2q_0 e \tag{4.6}$$

where I is the identity matrix and e is anti-symmetric, and it denotes:

$$e = \begin{bmatrix} 0 & q_3 & -q_2 \\ -q_3 & 0 & q_1 \\ q_2 & -q_1 & 0 \end{bmatrix} \tag{3}$$

The gain function $g(A)$ also can be written in terms of quaternion q:

$$g(q) = (q_0^2 - e^T e)\varepsilon + e^T S e + 2q_0(e^T Z) \tag{4.7}$$

In this equation:

$$\begin{cases} \varepsilon = tr(B), \\ S = B + B^T, \\ Z = \sum_{i=1}^{n} a_i(\hat{W}_i)\hat{V}_i)) = B - B^T. \end{cases} \tag{4.8}$$

Where K is the maximum eigenvalue of the matrix:

$$K = \begin{bmatrix} S - \varepsilon I & Z \\ Z^T & \varepsilon \end{bmatrix} \tag{4.9}$$

From the Method of Lagrange multipliers, Equation 4.7 should be:

$$g(q) = q^T K q - \lambda q^T q \tag{4.10}$$

To get the optimal quaternion of the gain function, taking the first derivative of 4.10 for q^T, and setting this equation to zero, we have $\lambda q = Kq$. The quaternion is represented by the Gibb's vector Y:

$$\begin{cases} q = \dfrac{1}{1 + ||Y||^2} \begin{bmatrix} Y \\ 1 \end{bmatrix}, \\ Y = [(\lambda + \varepsilon)I - S]^{-1} Z, \\ \lambda = \varepsilon + Z * Y. \end{cases} \tag{4.11}$$

When $\lambda = \lambda_{max}$, the Y and q represent the optimal attitude solution, and the eigenvalues λ can be determined by:

$$\lambda = \sigma + Z^T \frac{1}{(\lambda + \varepsilon)I - S} Z \tag{4.12}$$

(4.12) is the characteristic equation for the eigenvalue K; however, the maximum eigenvalue of K is close to the unit when the gain function is maximized, which results in the singularity problem. To avoid this problem, the Cayley-Hamilton theorem can be applied to the characteristic of the matrix:

$$S^3 = 2\sigma S^2 - \kappa S + \Delta I \tag{4.13}$$

where

$$\begin{aligned} \sigma &= 1/2trS \\ \kappa &= tr(adjS) \\ \Delta &= detS \end{aligned} \tag{4.14}$$

From (4.13), the meromorphic function of S can be described by :

$$[(\omega + \sigma)I - S]^{-1} = \gamma^{-1}(\alpha I + \beta S + S^2) \tag{4.15}$$

where $\alpha = \omega^2 - \sigma^2 + \kappa$,$\beta = \omega - \sigma$, and $\gamma = (\omega + \sigma)\alpha - \Delta$ Inserting (4.15) to (4.12) and a new e characteristic equation is obtained:

$$\lambda^4 - (a + b)\lambda^2 - c\lambda + (ab + c\sigma - d) = 0 \tag{4.16}$$

Where $a = \sigma^2 - \kappa$, $b = \sigma^2 + Z^T Z c = \Delta + Z^T S Z$ $d = Z^T S^2 Z$

The maximum eigenvalue of (4.16) is close to unity, and Newton-Raphson Method is applied to (4.16) to get a high accuracy maximum eigenvalue of the matrix. In this research, there are two 3-axial sensors to be applied, and the maximum eigenvalue can be written:

$$\lambda_{max} = \sqrt{a_1^2 + 2a_1a_2 \cos\theta_V - \theta_W + a_2^2} \tag{4.17}$$

where

$$\cos\theta_V - \theta_W = (\hat{V}_1\hat{V}_2)(\hat{W}_1\hat{W}_2) + \left|\hat{V}_1\hat{V}_2\right| \left|\hat{W}_1\hat{W}_2\right| \tag{4.18}$$

Then for (4.16), assuming the value of ω is λ_{max}, we obtain:

$$q_{opt} = \frac{1}{\gamma^2 + \|X\|^2} \begin{bmatrix} X \\ Y \end{bmatrix} \tag{4.19}$$

4.3.1.2 FQA Algorithm

According to (Liu et al. 2012), FQA calculates the quaternion of each Euler angle separately. It first calculates the elevation orientation, i.e. the pitch angle θ. From the accelerometer readings, we obtain:

$$a = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} \tag{4.20}$$

It is necessary to normalize the vector of the accelerometer measurements and get $\overline{a}$. It is noted that when a rigid body moves at a constant velocity and is in a fixed orientation, the accelerometer only measures gravity. In addition, the x-axis accelerometer only detects the component of gravity along the x-axis, which is also described by the elevation angle (pitch angle). It can be seen in the following arguments. The x-axis accelerometer is perpendicular to gravity; therefore, it detects zero acceleration when the rigid body is in its reference orientation. The y-axis accelerometer is also zero, while the z-axis is $-g$. If the rigid body is pitched up through an angle θ, the relationship between the pitch angle θ and accelerometers measurements is:

$$a_x = g\sin\theta \tag{4.21}$$

Where $g = 9.81m/s^2$, is the acceleration gravity, and the θ can be expressed as:

$$\sin\theta = \overline{a}_x \tag{4.22}$$

To obtain the quaternion of the pitch angle, it is necessary to calculate the $\cos\frac{\theta}{2}$ and $\sin\frac{\theta}{2}$ by the trigonometric half-angle formulae:

$$\sin\frac{\theta}{2} = sgn(\sin\theta)\sqrt{(1-\cos\theta)/2} \tag{4.23}$$

$$\cos\frac{\theta}{2} = \sqrt{(1+\cos\theta)/2} \tag{4.24}$$

Therefore, the elevation quaternion can be represented as:

$$q_e = \cos\frac{\theta}{2}\begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix} + \sin\frac{\theta}{2}\begin{bmatrix}0 & 0 & 1 & 0\end{bmatrix} \tag{4.25}$$

The relationship between roll angle ϕ and accelerometer readings are:

$$\begin{aligned} a_y &= -g\cos\theta\sin\phi \\ a_z &= -g\cos\theta\cos\phi \end{aligned} \tag{4.26}$$

The a_y and a_z should be in the normalization form and (4.26) can be represented as:

$$\begin{aligned} \sin\phi &= -\overline{a}_y/\cos\theta \\ \cos\phi &= -\overline{a}_z/\cos\theta \end{aligned} \tag{4.27}$$

The next step is to obtain the half-angle sine and cosine for the roll angle in the same way as (4.23) and (4.24), and the roll quaternion is obtained by:

$$q_r = \cos\frac{\phi}{2}\begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix} + \sin\frac{\phi}{2}\begin{bmatrix}0 & 1 & 0 & 0\end{bmatrix} \tag{4.28}$$

The calculation of the yaw angle only requires the magnetometer readings, and the first step is normalization:

$$m^b = \begin{bmatrix} m_x^b \\ m_y^b \\ m_z^b \end{bmatrix} \tag{4.29}$$

And the magnetic in the intermediate Earth coordinate system:

$$m^e = (q_e \otimes q_r) \otimes m^b \otimes (q_r^{-1} \otimes q_e^{-1}) \tag{4.30}$$

In this formula, the m^b can be written in the pure vector $m^b = [0\ m_x^b\ m_y^b\ m_z^b]$, and q^{-1} is the inverse quaternion. m^e should be the same with the local normalized magnetic field vector $n = [(n_x\ n_y\ n_z)]$, which can be found in [41] when there is no measurement error.

$$\begin{bmatrix} n_x \\ n_y \end{bmatrix} = \begin{bmatrix} cos\psi & -sin\psi \\ sin\psi & cos\psi \end{bmatrix} \begin{bmatrix} m_x^e \\ m_y^e \end{bmatrix} \tag{4.31}$$

After the normalization, it can be written as:

$$N = \begin{bmatrix} N_x \\ N_y \end{bmatrix} = \frac{1}{\sqrt{{n_x}^2 + {n_y}^2}} = \begin{bmatrix} n_x \\ n_y \end{bmatrix} \tag{4.32}$$

And the corresponding data measured by the magnetometer are:

$$M = \begin{bmatrix} M_x \\ M_y \end{bmatrix} = \frac{1}{\sqrt{m_x^{e\,2} + m_y^{e\,2}}} \begin{bmatrix} m_x^e \\ m_y^e \end{bmatrix} \tag{4.33}$$

Then the value of $\sin(\Psi)$ and $\cos\psi$ can be presented as:

$$\begin{bmatrix} \cos\psi \\ \sin\psi \end{bmatrix} = \begin{bmatrix} M_x & M_y \\ -M_y & M_x \end{bmatrix} \begin{bmatrix} N_x \\ N_y \end{bmatrix} \tag{4.34}$$

The yaw angle quaternion is given by:

$$q_a = \cos\frac{\psi}{2} \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} + \sin\frac{\psi}{2} \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.35}$$

Finally, the computed quaternion can be obtained by:

$$q_c = q_a \otimes q_e \otimes q_r \tag{4.36}$$

4.3.2 New Observation Model

The LERP can be used to generate the new observation model from the FQA and QUEST algorithms. Figure 4.3 shows how the LERP is used to fuse FQA and QUEST, and the output quaternion can be the observation model. It enhances the accuracy and solves high-frequency noise-affected accelerometer data and low-frequency noise-affected gyroscope output (Valenti et al. 2015). Figure 4.3indicates the quaternion q_f and q_Q from FQA and QUEST, respectively, and Δq_f, Δq_Q are quaternion noise of FQA and QUEST, respectively. From LERP, the quaternion noise can be calculated by:

$$\begin{cases} \Delta\overline{q_f} = (1-\alpha)q_I + \alpha q_f, \\ \Delta\overline{q_Q} = (1-\alpha)q_I + \alpha q_Q. \end{cases} \tag{4.37}$$

Where $q_I = [1\ 0\ 0\ 0]$ and $\alpha \in [0,1]$ is the introduced in (De Franceschi and Zardi 2003). However, the outputs of the LERP are not unit quaternion, it is necessary to normalize the quaternion:

$$\begin{cases} \Delta\hat{q_f} = \dfrac{\Delta\overline{q_f}}{\|q_f\|} \\ \Delta\hat{q_Q} = \dfrac{\Delta\overline{q_Q}}{\|q_Q\|} \end{cases} \tag{4.38}$$

The first group of optimal quaternion q_oq from the fusion of the FQA and the QUEST by LERP requires the employment of a select vector:

$$q_{oq} = \begin{cases} q_f \otimes \Delta\hat{q_f} & if \Delta\overline{q_f} < \Delta\overline{q_Q} \\ q_Q \otimes \Delta\hat{q_Q} & if \Delta\overline{q_Q} < \Delta\overline{q_f} \end{cases} \tag{4.39}$$

Furthermore, the FQA and QUEST are suitable for static conditions, whilst the integrating angular rate is suitable for dynamic conditions, necessitating the use of a second LERP to extract the second group optimal quaternion q_{of} from Kalman Filter and FQA.

$$q_{of} = (1-G)q_\omega + Gq_f \tag{4.40}$$

Where G is the gain 0 to 1; q_ω is the quaternion by integrating the angular rate from the gyroscope, q_{of} is the optimal quaternion by combining q_f and q_ω.

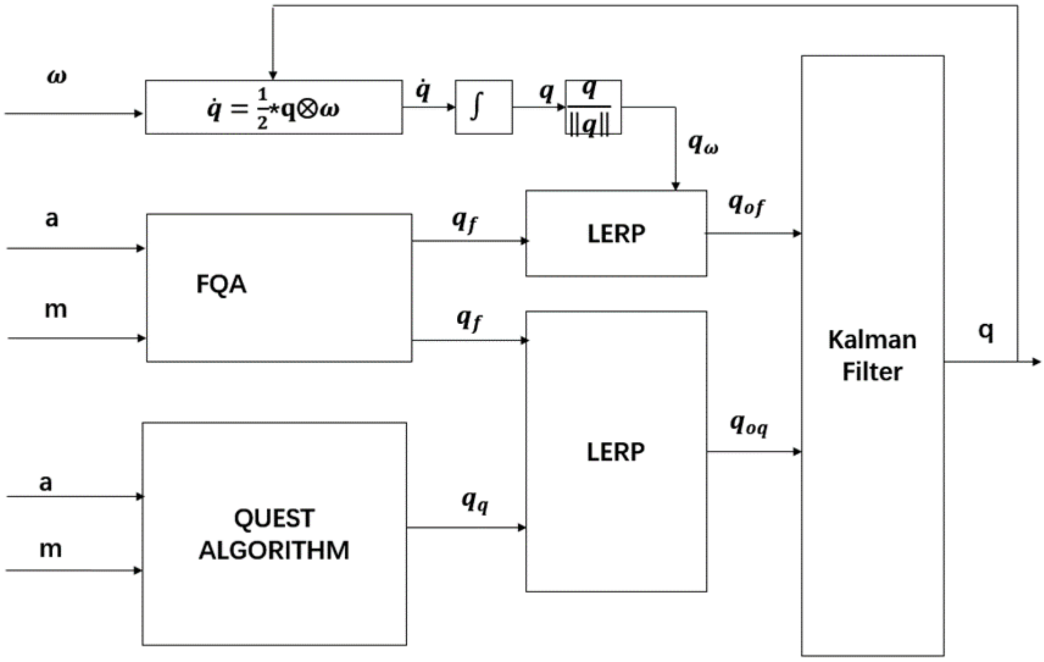


FIGURE 4.3: Block diagram of the proposed algorithm

4.3.2.1 Details of A Two-layer Kalman Filter

In the Kalman Filter, the most important two parts are the process model and observation model, which can be described by:

$$\begin{cases} X_k = \phi_{k-1} X_{k-1} + w_{k-1}, \\ Z_k = H X_k + v_k. \end{cases} \tag{4.41}$$

Where ϕ is the state transition matrix, H denotes the observation matrix, the identity matrix, and w and v are White Gauss process noise and measurement noise. In this scheme, the state vector is X_k and the observation vector is Z_k and they can be defined as:

$$\begin{cases} X_k = q_{of}, \\ Z_k = q_{oq}. \end{cases} \tag{4.42}$$

Therefore, the state vector can be rewritten in:

$$q_{k+1} = \phi_k(\Omega_k, T_s)q_k + w_k \tag{4.43}$$

where T_s in it represents the sample time, which should be quite small and

$$\Omega(\omega) = \frac{1}{2}\begin{bmatrix} 0 & -\omega^T \\ \omega & \omega[\times] \end{bmatrix} \tag{4.44}$$

ω is the angular velocity of the gyroscope and its size is 1×3. In addition, $\omega[\times]$ is a 3×3 skew-symmetric matrix described by:

$$\omega[\times] = \begin{bmatrix} 0 & -\omega_z & -\omega_y \\ -\omega_z & 0 & \omega_x \\ \omega_y & -\omega_x & 0 \end{bmatrix} \tag{4.45}$$

Then the Taylor series can be applied to (4.43), we can get:

$$q_{k+1} = [I_{4\times4}(1 - \frac{\Delta\theta^2}{8}) + \frac{1}{2}\Omega_k T_s]q_k \tag{4.46}$$

where

$$\Delta\theta^2 = (\omega_x T_s)^2 + (\omega_y T_s)^2 + (\omega_z T_s)^2 \tag{4.47}$$

the state vector(4.43) can be represented by:

$$q_{k+1} = [I_{4\times4}(1 - \frac{\Delta\theta^2}{8}) + \frac{1}{2}\Omega_k T_s]q_k + \omega_k^q \tag{4.48}$$

and

$$-\frac{T_s}{2}\Xi_k\omega^m = -\frac{T_s}{2}\begin{bmatrix} [e_k\times] - q_0 I_{3\times3} \\ -{e_k}^T \end{bmatrix}\omega^m = \omega_k^q \tag{4.49}$$

Where $[e_k]$ =$[q_{1k}\ q_{2k}\ q_{3k}]^T$, and ω^m is the gyroscope measurement noise. The Kalman Filter also considers the process noise covariance matrix Q_k:

$$Q_k = \frac{{T_s}^2}{2}\Xi_k{\sigma_g}^2 I{\Xi_k}^T \tag{4.50}$$

For the observation model, the measurement model can be given by:

$$Z = \begin{bmatrix} q_{oq0} & q_{oq1} & q_{oq2} & q_{oq3} \end{bmatrix}^T \tag{4.51}$$

During the process, it is essential to calculate the Kalman Gain K and the measurement noise covariance matrix R_k , which can be defined by:

$$\begin{aligned} K_{k+1} &= P_{k+1}^{-} H_{k+1}^{T} (H_{k+1} P_{k+1}^{-} H_{k+1}^{T} + R_{k+1})^{-1} \\ R_k &= E\left[V_k V_k^T\right] = \sigma_q^2 I \end{aligned} \tag{4.52}$$

The ${\sigma_q}^2$ is averaged quaternion variance, P_{k+1}^{-} is prior error covariance and I is the four-dimensional identity matrix.

Figure 4.4 depicts the whole structure of the Kalman Filter. It demonstrates the fluent diagram of the whole Kalman ole Kalman Filter and consists of three primary components: projection, Kalman gain, and update. The initial estimate of quaternion $\hat{X}_0$ or $\hat{q}_0$is defined as the matrix $[1\ 0\ 0\ 0]^T$ and P_0 is the initial covariance matrix equaling the matrix $[1\ 0\ 0\ 0]^T$. They are input to the projection step to compute a priori state estimate quaternion q_{k+1}^{-} and a priori error covariance P_{k+1}^{-}. Then calculating the Kalman Gain K_{k+1} be the basis of updating a posteriori state estimate quaternion $q_{k+1}$and a posteriori state error covariance P_{k+1}. Finally, the optimal quaternion q_{oq} from LERP input to the update step of Kalman Filter together to calculate a posteriori state estimate quaternion $q_{k+1}(q_{of})$and a posteriori state error covariance P_{k+1}.

4.4 Simulation and Experiments of New Two-Layer Kalman Filter

4.4.1 Simulation Results

We obtain the true attitude information in the simulation by integrating the angular rate without noise. The simulated angular velocity, acceleration, and magnetic field are affected by White Gaussian noise with a standard deviation of $\sigma_g = 0.0545 rad/s$, $\sigma_a = 0.19 m/s^2$, and $\sigma_m = 0.001 G$, respectively. In addition, the measurement noise of the accelerometer, gyroscope, and magnetometer is $0.0001972 m/s^2$, $9.1385 \times 10^{-5} rad/s$, and $0.1 \mu T$,and the step time is 0.01 seconds. The true attitude is compared with the results of the proposed

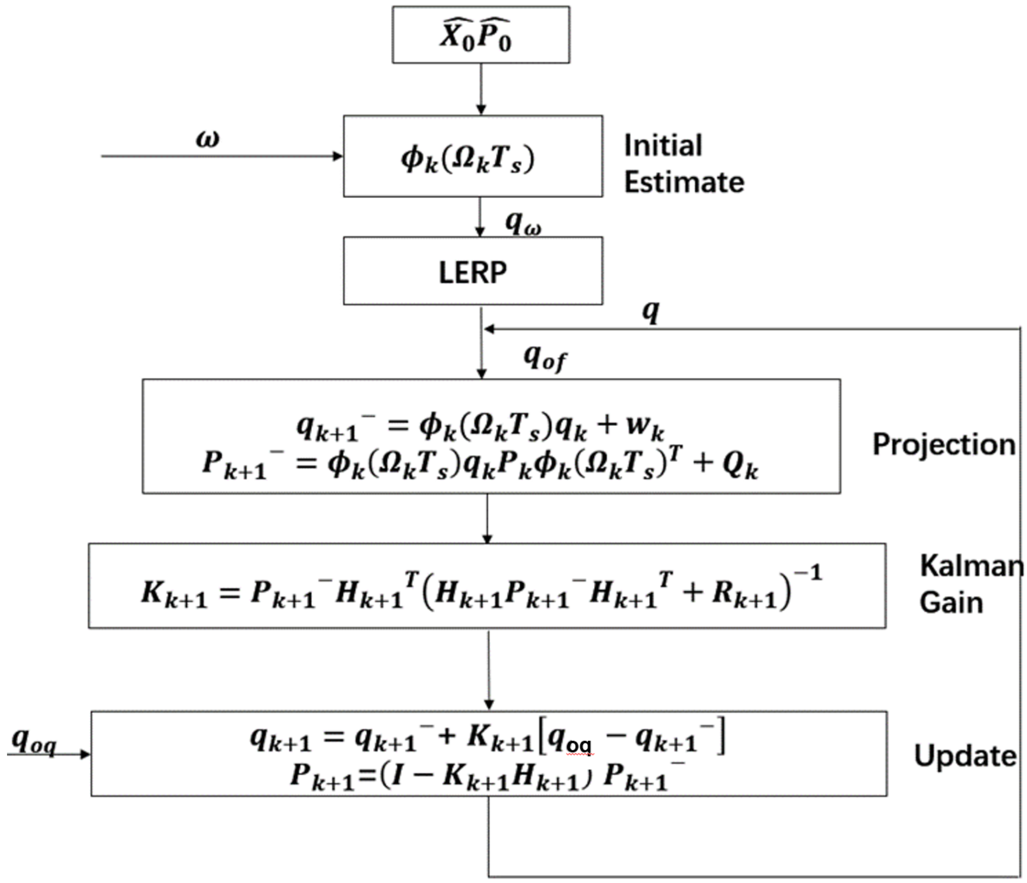


FIGURE 4.4: Block diagram of Kalman Filter

algorithm, QKF, and FQAKF in figure 4.5. The proposed algorithm outperforms the QKF and FQAKF in terms of reduced errors in all Euler angles, particularly the pitch angle.

Figure 4.6 depicts the detailed mistakes of the proposed approach and other algorithms. The average error of the proposed algorithm is 0.2669 deg. The average absolute error of QKF is 0.7912 deg, which is more than 3 times higher than that obtained by the proposed algorithm. The average error of FQAKF is 1.5818 deg, which is the largest error of the three algorithms. This indicates that the proposed algorithm outperforms the other comparison Kalman filters in simulation.

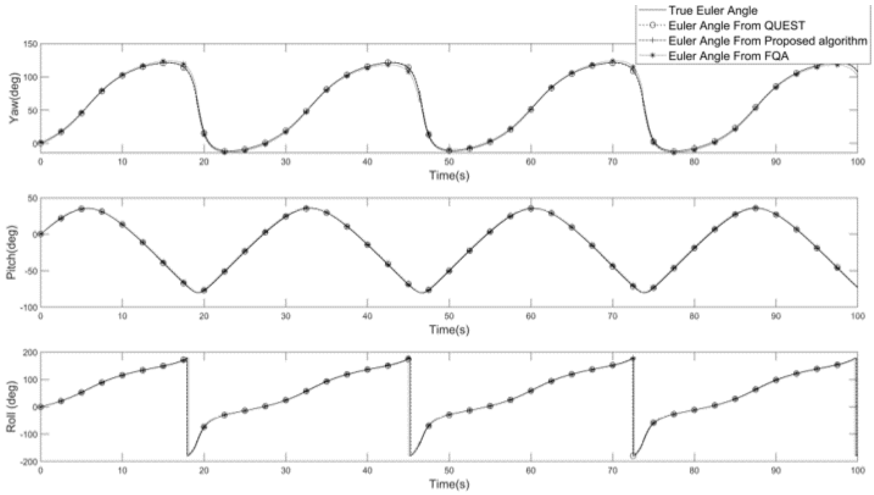


FIGURE 4.5: Comparison of Simulated Euler Angler Between Different Algorithms

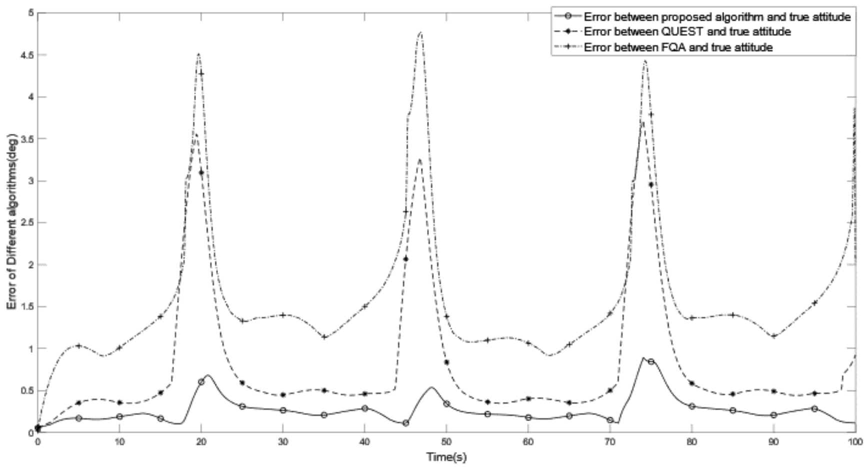


FIGURE 4.6: Comparison of the absolute error between Different Algorithms

4.4.2 Experimental Results

In this section, we evaluate the performances of the three algorithms using MARG inertial/magnetic sensors experimentally. The experiment setup on the air-bearing table as shown in figure 4.7. The air-bearing table offers torque-free motion with three degrees of freedom, excluding extreme angles due to the platform structure. The LSM9DS1 is a system-in-package featuring a 3D digital linear acceleration sensor, a 3D digital angular rate sensor, and a 3D digital magnetic sensor. The running step is around 0.036 seconds, and there are 4631 steps.

It takes about 166 seconds for the whole experiment. The initial quaternion $q = [1\ 0\ 0\ 0]^T$. Furthermore, the sensor module is assembled with its x, y, z-axis and aligned with NED (North, EAST, Down) direction.

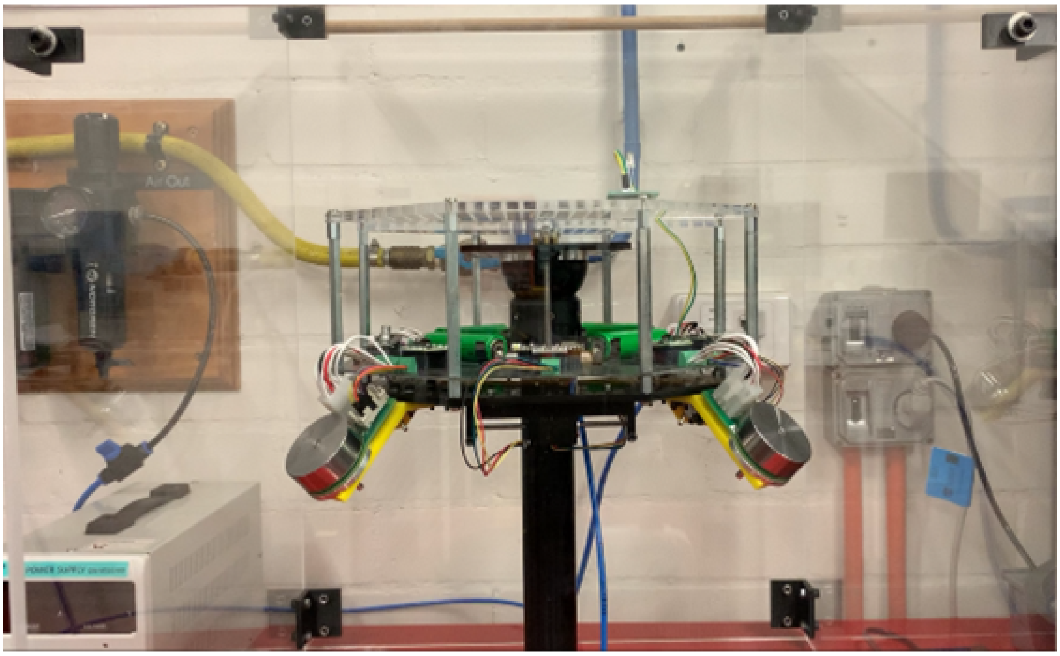


FIGURE 4.7: The experiment on air-bearing table

Figure 4.8 illustrates the quaternion results from the three algorithms. Each FQAKF quaternion has significant errors, particularly q_0 and q_3, while q_1 and q_2 have fewer errors. The estimated quaternion q from the proposed algorithm is the closest to the reference quaternion, followed by the quaternion from QKF. In conclusion, the proposed algorithm presents the best accuracy, followed by QKF while FQAKF shows the most inferior performance. Figure 4.9 depicts the quaternion errors from different algorithms. The FQAKF has the largest error, which is about 0.1614, while the error of the QKF is 0.0136. The error of the proposed algorithm is minimal (0.004). As a result, the proposed method outperforms QKF and FQAKF in terms of accuracy.

Table 4.1 shows the average quaternion errors of the three algorithms. The proposed algorithm has minimum errors for almost all the members of the quaternion except for the q_3. However, the difference of q_3, between the proposed algorithm and the FQAKF is negligible. Therefore,

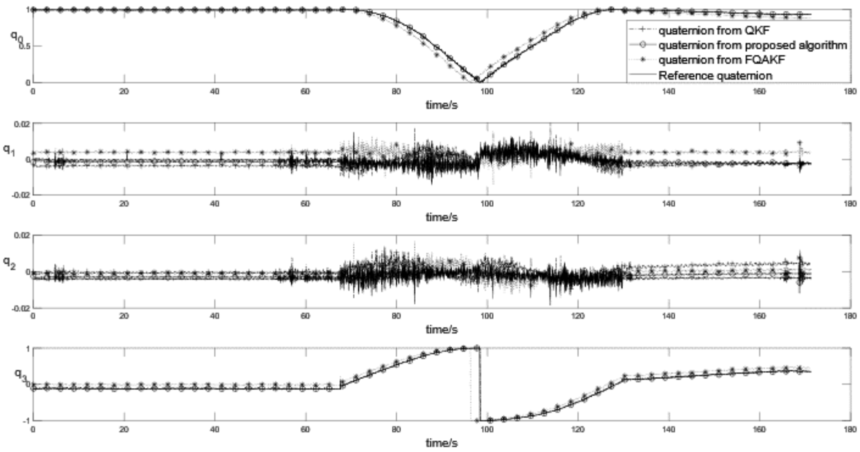


FIGURE 4.8: The Quaternion Produced By Different Algorithms

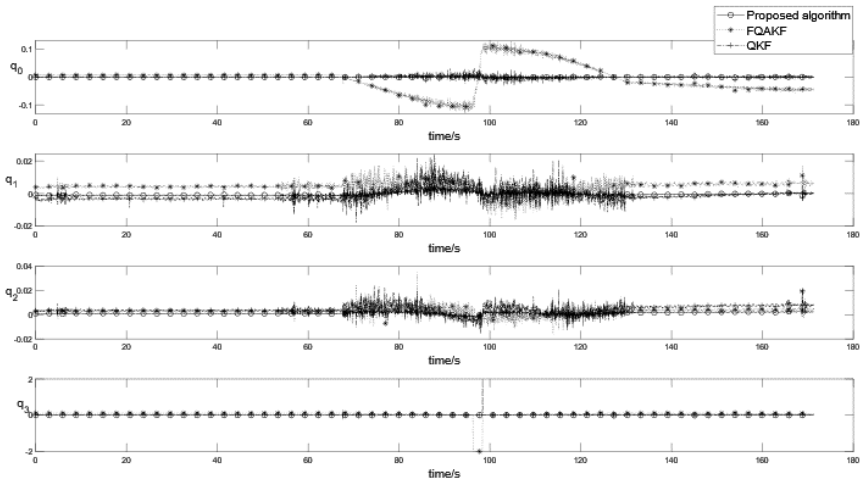


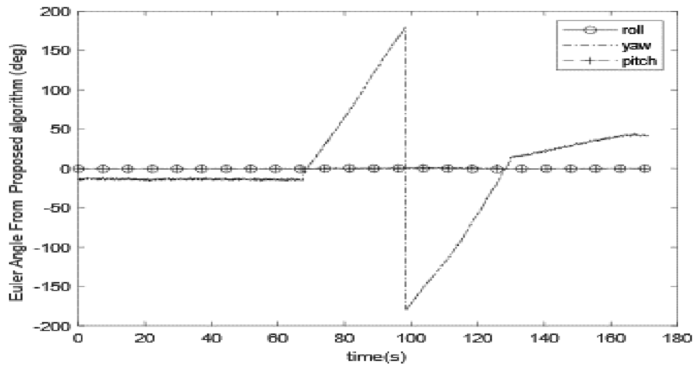
FIGURE 4.9: The Quaternion Error 1of Different Algorithms(Presented Algorithm, QKF, FQAKF)

the quaternion accuracy of the proposed algorithm is the highest, and the quaternion accuracy of the FQAKF is the lowest.

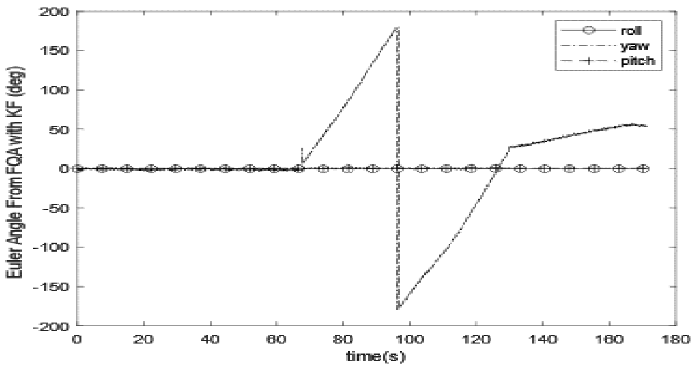
TABLE 4.1: DETAILED QUATERNION ERROR OF DIFFERENT ALGORITHMS

quaternion	Proposed algorithm	FQAKF	QKF
q_0	5.7639×10^{-4}	0.0346	0.0019
q_1	9.3341×10^{-4}	0.0051	0.0031
q_2	0.0013	0.0038	0.005
q_3	9.5106×10^{-4}	0.1179	0.0036

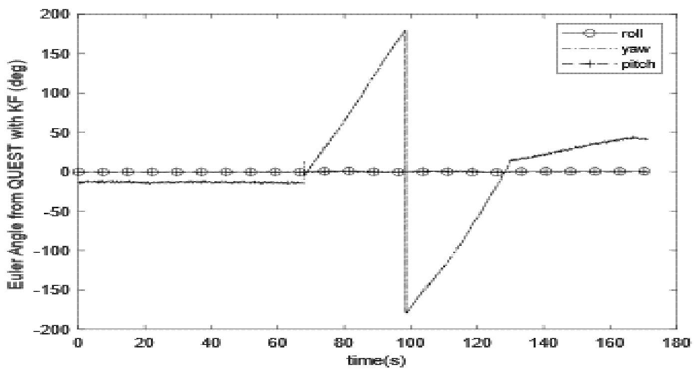
Figure 4.10 shows the Euler angles produced by the proposed algorithm (a), FQAKF (b), QKF (c), and the reference Euler angles (d), respectively. In this experiment, the air-bearing table only changes the yaw angle, which has a 360-degree rotation. In the proposed algorithm, the roll angle and pitch angle calculated by the proposed controller are robust to the magnetic distortion, which is the main advantage of the proposed algorithm. The yaw angle from FQAKF shows the significant delays from the other algorithms. The reference yaw angle is similar to the yaw angle from the proposed algorithm and the detailed average error is shown in Table 4.2.



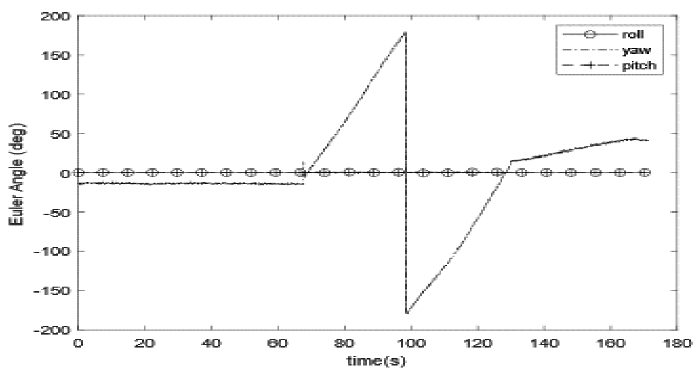
(a)



(b)



(c)



(d)

FIGURE 4.10: Euler angle from different algorithms

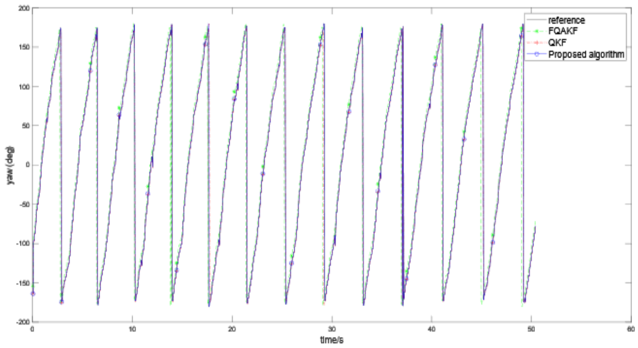
TABLE 4.2: ERROR OF YAW ANGLE BETWEEN DIFFERENT ALGORITHMS AND REFERENCE YAW ANGLE

quaternion	Proposed algorithm	FQAKF	QKF
Average error of yaw angle(deg)	0.1421	8.1971	0.6266
Average error of pitch angle(deg)	0.1606	0.3643	0.5353
Average error of roll angle(deg)	0.1194	0.5837	0.3977

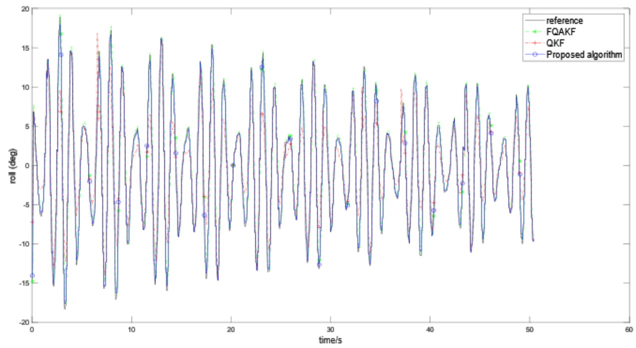
The average error of the yaw angle from the proposed algorithm is 0.1421, which is 97% lower than that of the FQAKF and around 23% lower than that of the QKF. For the pitch angle and the roll angle, the proposed algorithm also presents the minimum error.

According to the previous experiment, the proposed algorithm is more accurate on the yaw angle than other algorithms. The experiment results in figure 4.11 demonstrate that the proposed algorithm is not only more accurate on the yaw angle, but also the pitch angle and roll angle than other algorithms. Figure 4.11 shows that the proposed algorithm produces the best results for pitch and roll angle, followed by the QKF and the FQAKF with the lowest accuracy.

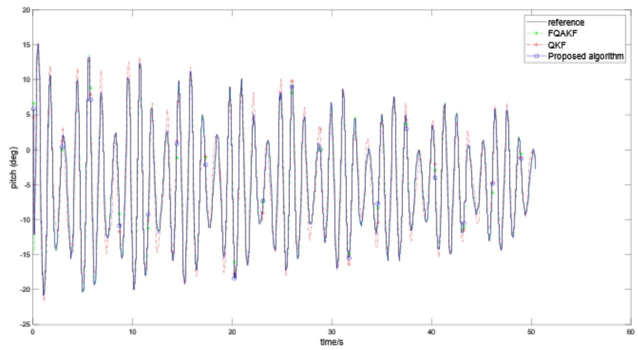
The process model and observation mode are 7×7 matrixes in the QKF. There are 6 steps of matrix calculation and 294 elements to estimate the attitudes in one loop. The FQAKF requires 4×4 matrixes and 9 matrix calculation steps. Therefore, there are 144 elements in one loop. The proposed algorithm employs a 4×4 matrix, and two matrix calculation steps less than FQAKF because of the LERP. Therefore, there are 112 elements in one loop. We conduct 10 repeat experiments and got the average running time for the algorithms. The three algorithms are implemented on the Raspberry PI 3B+ with Python. The Raspberry PI 3B+ has a 64-bit 1.4GHz quad-core ARM Cortex-A53 (CPU), 1 GB LPDDR2 SDRAM(RAM), and Broadcom Videncore-IV(GPU). The speed of 1K writes and 1K reads of memory are 632.27MB/S and 857.96MB/S, respectively. Table 4.3 exhibits the average one-loop running time of three algorithms on Raspberry PI 3B+. The running time of the proposed algorithm is 0.002459526 seconds less than that of the FQAKF (0.010483503 seconds) and lower than that of the QKF (0.011303639 seconds). In theory, the computational load of the FQAKF is 29% higher than the proposed algorithm. In the experiment, compared with the FQAKF, the proposed algorithm improves 16.9% computational efficiency. While the efficiency of the QKF algorithm is the worst.



(a)



(b)



(c)

FIGURE 4.11: Euler angle from different algorithms

TABLE 4.3: RUNNING TIME OF DIFFERENT ALGORITHMS

	TIME
Novel algorithm	0.006459526 seconds
QKF	0.011303639 seconds
FQAKF	0.010483503 seconds

4.5 Summery

This chapter presents a two-layer Kalman filter architecture for attitude estimation. It provides a novel algorithm to produce a quaternion observation model that is the result of the fusion of QUEST and FQA, offering a higher attitude accuracy than the attitudes obtained if these algorithms were used individually. The Kalman filter employs the quaternion matrix from the LERP of the observation and process models. The process model in the two-layer architecture is formed by integrating the gyroscope output, providing a smooth output while avoiding singularities and reducing computational complexity. The two-layer architecture is robust against magnetic disturbance and other undesirable conditions. Furthermore, the second LERP ensures the attitude estimation of vehicles can maintain a high level of accuracy in both static and dynamic motion conditions. The performance of the proposed algorithm is validated by both simulation and physical experiments on the air-bearing table. The experiment results indicate that the proposed algorithm can achieve highly accurate attitude estimation with low computational load. This part has the following highlights:

(1) Compared with traditional algorithms of QKF, and FQAKF, the fusion of QUEST and FQA by LERP can improve the attitude estimation accuracy.

(2) The quaternion-based Kalman Filter is in the minimum order, significantly reducing the computational load.

(3) The 2-layer architecture algorithm combines the measurement model and the Kalman Filter to form a feedback loop, which can deal with unexpected situations such as fast-moving and temporary magnetic disturbance. In addition, the LERP can reduce the quaternion error from QUEST, FQA, and integration of the gyroscope output.

(4) The second LERP step takes advantage of QUEST, FQA, and the gyroscope to achieve better performances on both the short-term and long-term attitude estimations.

CHAPTER 5

Kalman filter and neural network fusion for fault detection and recovery in satellite attitude estimation

This chapter provides a fault detection, isolation, and recovery scheme for satellite attitude estimation, which is presented by (Chen et al. 2023b), and the part of neural network has been published by (Chen et al. 2022). First, it shows a brief of the proposed scheme, and then the theory and structure of this algorithm are presented in the following parts. Finally, the simulation and experimental results are given to verify the validity of the algorithm.

5.1 Introduction

Most satellite missions have extremely stringent requirements for attitude reliability. However, the IMU, which is part of the ADS, is susceptible to performance degradation in the space environment and can lead to failure. The proposed fault tolerance scheme includes two-layer fault detection with isolation and two-layered recovery. AUKF, QUEST algorithm and residual generator constitute the first layer of fault detection. At the same time, RBF neural networks and an adaptive complementary filter (ACF) make up the second layer of fault detection. These two fault detection layers aim to isolate and identify faults while decreasing the rate of false alarms. The AUKF and FDIR residual generator comprise the two-layered attitude recovery system.

The proposed scheme builds two RBF neural networks from the AUKF and QUEST quaternion output, they offer a quaternion when a fault has been identified by the residual generator. The RBF activated is dependent on the sensor failure, triggering a chain response to disregard quaternions that are produced by either the AUKF or QUEST, depending on the relevant failed sensor.

Compared with the traditional fault tolerance system, this scheme reduces the calculation load in the residual generator. It not only solves the outlier problem of sensors but also has higher accuracy attitude estimation. When one of the IMU sensors fails and is detected, the proposed scheme can still maintain accurate attitude estimation by leveraging a trained neural network. In addition, the secondary fault detection and isolation layer can minimize the rate of false alarms, meaning more reliable ADS for satellites.

5.2 Fault Detection, Isolation, and Recovery System

In the proposed scheme, the system is divided into two main phases; the first phase is for fault detection and isolation to give preliminary recovery, and the second part is a secondary, more advanced recovery stage. The preliminary recovery is based on the AUKF and the QUEST algorithm, while the RBF is only used for training the model during this phase, as seen in figure 5.1.

Fig.5.1 includes two estimators: an AUKF, which has a fault recovery function, and QUEST. The AUKF estimator employs a gyroscope and a magnetometer to get the quaternion q_a based on the satellite attitude kinematics seen in (4.1). In normal operation, the AUKF result is the same as the UKF. If a fault occurs in the gyroscope or magnetometer, the AUKF will update the covariance matrix and produce an updated quaternion. The second estimator, QUEST, provides a quaternion q_q by minimizing the Wahba's loss function based on the outputs of the accelerometer and magnetometer in the inertial frame. The next are two residual generators, an ACF, and two RBF neural networks. When the fault is detected, firstly, the residual generators after the AUKF and QUEST algorithms will determine which sensor is broken. If the fault is in the gyroscope or magnetometer, AUKF will adjust the covariance matrix to achieve the new q_a. This is the preliminary recovery.

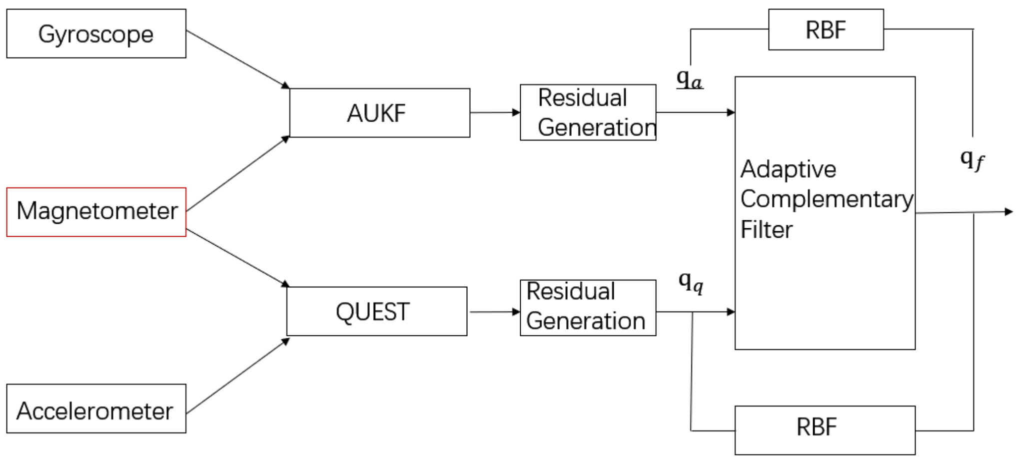


FIGURE 5.1: Flowchart of the fault detection, preliminary recovery, and the training process

During the secondary recovery phase, figure 5.2 shows the process of the secondary fault recovery by RBF and ACF. When the preliminary recovery is initiated, the two RBF neural networks stop training and start to produce two quaternion groups q_{r1} and q_{r2} at the same time.

The quaternion group to be used depends on the sensors without fault. When the gyroscope is at fault, the input data are q_a, q_q and q_{r1}. When the fault is at the magnetometer or the accelerometer, the input data are q_{r1} and q_a. If there is no fault, the input data are q_a and q_q. This quaternion is then used by ACF to get the final quaternion q_f. For example, the first RBF (RBF1) connects the AUKF outputs with final quaternion; The second RBF (RBF2) connects the QUEST outputs with the final quaternion. When the fault detection and fault isolation process determines that the gyroscope is in failure, then the quaternion from AUKF is incorrect. Hence, RBF2 will be used for the secondary fault recovery. The other quaternions used by ACF are from the QUEST and the primary recovery result from AUKF. This process increases the accuracy of the final attitude estimation.

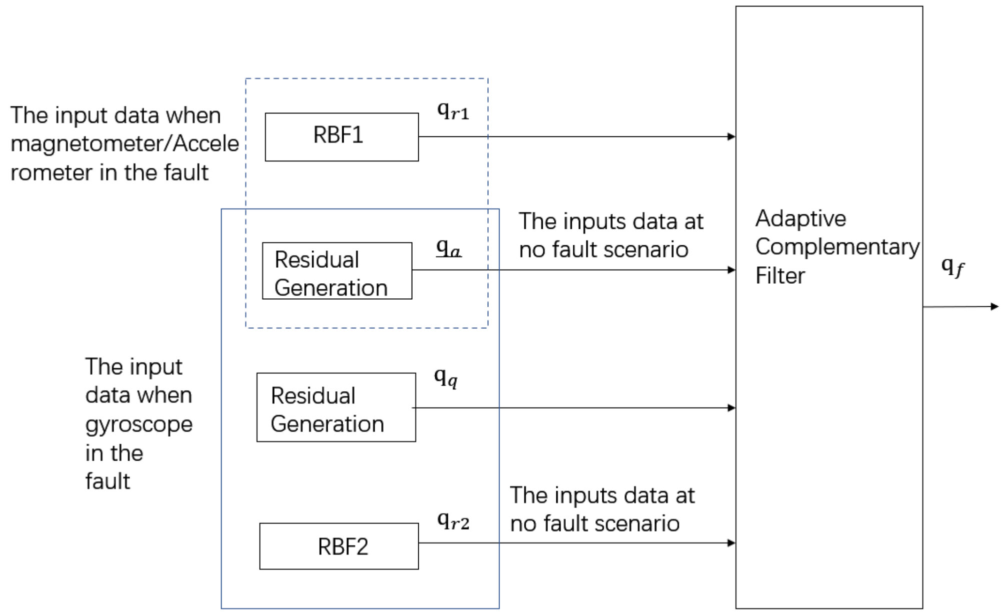


FIGURE 5.2: process of the secondary fault recovery by RBF and the Adaptive Complementary Filter.

5.2.1 The Theory of Fault Detection and Isolation System

The residual vectors of the filter/estimator will be monitored utilizing residual generation and statistical tests to find faults of failure in AUKF or QUEST. The hypothesis tests determine which sensor is in failure.

5.2.1.1 The Theory of UKF

UKF is used for nonlinear systems because traditional Kalman Filters can only be used to represent linear systems. The UKF can produce a set of the sigma points from a known prior mean and covariance of the current state, and the mean prediction is calculated from the transformed sigma points (Soken et al. 2014). The UKF model is given by:

$$\begin{aligned} x_{k+1} &= f(x_k, k) + w_k \\ \widetilde{y_k} &= h(x_k, k) + v_k \end{aligned} \tag{5.1}$$

Where, x_k is the state vector, $\widetilde{y_k}$ is the measurement vector, and w_k and v_k are stated process noise and measurement noise, respectively. Moreover, the noise vector w_k and $\widetilde{y_k}$ are assumed to be the Gaussian white noise with the covariance vector Q_k and R_k. f and h denote the system function and measurement function, respectively.

UKF starts with the calculation of the $2n+1$ sigma points with a mean of $\widetilde{x}(k|k)$ and a covariance of $P(k|k)$. For the n dimensional state vector, the sigma points are depicted by:

$$\begin{aligned} \chi_0^+ &= \widetilde{x}^+ \\ \chi_j^+ &= \widetilde{x} + \gamma(\sqrt{p_k^+ + Q_k})_j \\ \chi_{j+n}^+ &= \widetilde{x} - \gamma(\sqrt{p_k^+ + Q_k})_j \end{aligned} \tag{5.2}$$

Here, $j = 1, 2, 3...n$, $(\sqrt{p_k^+ + Q_k})_j$ is the j^{th} column of the square root matrix, $\gamma = \sqrt{n+\lambda}$, and $\lambda = \alpha^2(n+\kappa) - n$ are the scaling parameters. α is a constant number, which decides the spread of the sigma points around $\widetilde{x}(k|k)$ and normally is set as a small positive value. The constant κ is a secondary scaling parameter, which is always set to zero. The next step in UKF is using the system dynamics to convert each sigma point:

$$\chi_j^-(k+1) = f(\chi_k^+, k) \tag{5.3}$$

The prior mean and covariance are computed as:

$$\hat{x}_{k+1}^+ = \sum_{i=0}^{2n} W_i^m \chi_{k+1}^-(i) \tag{5.4}$$

$$P_{k+1}^- = \sum_{i=0}^{2n} W_i^c (\chi_{k+1}^-(i) - \hat{x}_{k+1}^+)(\chi_{k+1}^-(i) - \hat{x}_{k+1}^+)^T + Q_k \tag{5.5}$$

Where:

$$\begin{aligned} W_0^m &= \frac{\lambda}{n+\lambda} \\ W_0^c &= W_0^m + 1 - \alpha^2 + \beta \\ W_i^c = W_i^m &= \frac{1}{2(n+\lambda)} \end{aligned} \tag{5.6}$$

β is employed to include the prior information of the distribution of the state vector x. The mean measurement vector and output covariance matrix are shown by:

$$\hat{y}^{+}_{k+1} = \sum_{i=0}^{2n} W^{m}_{i} Y_{k+1}(i) \tag{5.7}$$

$$P^{yy}_{k+1} = \sum_{i=0}^{2n} W^{c}_{i} (Y_{k+1}(i) - \hat{y}^{+}_{k+1})(Y_{k+1}(i) - \hat{y}^{+}_{k+1})^{T} \tag{5.8}$$

Where the Y_{k+1} represents the predicted observation and it is given by

$$Y_{k+1}(i) = h(\chi^{-}_{k+1}(i), k) \tag{5.9}$$

Next the innovation covariance P^{vv} is calculated by

$$P^{vv}_{k+1} = P^{yy}_{k+1} + R_{k+1} \tag{5.10}$$

The cross-correlation matrix is represented by

$$P^{xy}_{k+1} = \sum_{i=0}^{2n} W^{c}_{i} (\chi_{k+1}(i) - \hat{x}^{+}_{k+1})(Y_{k+1}(i) - \hat{y}^{+}_{k+1})^{T} \tag{5.11}$$

The Kalman gain is calculated by P^{xy} and P^{vv}:

$$K_{k+1} = P^{xy}_{k+1} P^{vv}_{k+1} \tag{5.12}$$

Finally, the updated state and covariance are computed by the traditional Kalman Filter method:

$$\hat{x^{+}_{k+1}} = \hat{x^{-}_{k+1}} + K_{k+1}(\widetilde{y_{k+1}} - \hat{y}^{+}_{k+1})) \tag{5.13}$$

$$P^{+}_{k+1} = P^{-}_{k+1} - K_{k+1} P^{vv}_{k+1} K^{T}_{k+1} \tag{5.14}$$

5.2.1.2 Residual Generators

The critical step of the proposed algorithm detects the fault based on the residual, which is sensitive to the faults. The faults can be unpredictable in terms of time and magnitude. AUKF generates the first residual v_{r1}, which is the difference between the measured values from the sensors $\widetilde{y}$ and the predicted values from the AUKF $\hat{y}$.

$$v_{r1} = \widetilde{y} - \hat{y} \tag{5.15}$$

The first residual needs to be normalized to realize the state diagnosis:

$$\widetilde{v_{r1}} = \frac{v_{r1}(i)}{\sqrt{\sigma_{ii}^{vv}}}, i = 1 : n, \sqrt{\sigma_{ii}^{vv}} = p_{k+1}^{vv}(i, 1) \tag{5.16}$$

The second residual is based on the QUEST algorithm, which outputs the current satellite attitudes $q_q = [q_0\ q_1\ q_2\ q_3]^T$; however, it does not store past data or predict any attitude information. Therefore, the variance of the q_q can be monitored, rather than the output of the QUEST algorithm.

In order to reduce the false alarm rate, the hypothesis test is employed to decide the correct threshold. The statistical hypothesis is a method based on the data from scientific studies. The testing begins by expressing the value of a population mean in a null hypothesis, which is assumed to be true. For the first residual vector, it is important to find the threshold J_{th1},

$$prob\{|\overline{v}_{r1}| > J_{th1}|\theta = 0\} < \alpha, \overline{v}_{r1} = \frac{1}{M}\sum_{j=1}^{N} v_{r1} \tag{5.17}$$

Where, $prob\{|\overline{v_{r1}}| > J_{th1}|\theta = 0\}$ is the probability that $|\overline{v_{r1}}| > J_{th1}$ when the condition $\theta = 0$, it represents the false alarm rate. α is a constant number and also can be called the significant level. In this scheme, the following rules are adopted:

$$\begin{aligned} \hat{v_{r1}} \leq J_{th1} : \theta = 0(H_0, nullhypothesis) \\ \hat{v_{r1}} > J_{th1} : \theta = 0(H_1, alternativehypothesis) \end{aligned} \tag{5.18}$$

Where, the null hypothesis H_0 denotes the no-fault situation, the alternative hypothesis H_1 represents the presence of a fault. Then the next step is calculating the threshold J_{th1}, the detailed steps are shown as follows.

Step 1:

$$prob\{t > t_{\alpha/2}\} = \alpha/2 \tag{5.19}$$

$\alpha/2$ is calculated by the table of t distribution with the degree of freedom, which equals to $M-1$

Step 2:

$$J_{th1} = t_{\alpha/2}\frac{S_{r1}}{\sqrt{M}} \tag{5.20}$$

$$S_{r1}^2 = \frac{\sum_{j=1}^{M}(v_{r1j} - \overline{v}_{r1})}{M-1} \tag{5.21}$$

Where

$$t = \frac{\overline{v}_{r1}}{S_{r1}/\sqrt{M}} \tag{5.22}$$

For the second residual generator, it is necessary to find a statistic threshold I_{th2}, and the best method is monitoring the change in the variance. The statistic is depicted as:

$$S_{r2}^2 = \frac{1}{M-1}\sum_{j=1}^{M}(q_j - \overline{q})^2 \tag{5.23}$$

$$\overline{q} = \frac{1}{M}\sum_{j=1}^{M} q_j \tag{5.24}$$

In addition, the decision rule is denoted as:

$$\begin{aligned} S_{r2}^2 \leq J_{th2} &: (H_0, nullhypothesis) \\ S_{r2}^2 > J_{th2} &: (H_1, alternativehypothesis) \end{aligned} \tag{5.25}$$

When $S_{r2}^2 \leq J_{th2}$, it indicates there is no fault, while $S_{r2}^2 > J_{th2}$ represents the fault case.

5.2.2 Fault Isolation

After fault detection, the next step is fault isolation. This scheme allows for identifying single-sensor fault situations.

TABLE 5.1: Table 1 Fault isolation logic table

AUKF	×	×	×	✓
QUEST	✓	×	×	✓
Broken sensor	Gyro	Magn	Accel	No Fault

The first residual generator is based on AUKF, which receives data from the gyroscope and magnetometer. The second residual generator is based on QUEST, which is designed to obtain the measurement data from the accelerometer and the magnetometer. Table 5.1 shows the fault isolation logic in the FTD system, to determine which sensor is in fault. The $\times$ represents H_1 alternative hypothesis, which means fault occurs, and the $\checkmark$ represents H_0 null hypothesis, which means no fault. From Table 1, it can be seen when AUKF is detected as the fault and QUEST is no fault, it means the fault has only occurred in the gyroscope. Similarly, when AUKF and QUEST are both detected as faults, it signifies a fault has occurred in the magnetometer. If the fault occurs in the accelerometer, AUKF is in normal operation while the QUEST is detected as a fault.

5.2.3 The Preliminary Recovery: AUKF

The input of AUKF is the angular velocity and magnetic field data from the gyroscope and the magnetometer, respectively. From the first residual generator, the first residual vector v_{r1} can be computed by (5.13) and the average value of the first residual can be calculated by (5.16). The estimated value of the sensor noise can be calculated as:

$$\hat{\sigma}_R^2 = \frac{\sum_{j=1}^{M}(v_{r1j-\bar{v}_{r1}})^2}{M-1} \tag{5.26}$$

When either the gyroscope or magnetometer has a fault, the covariance matrix R^* is updated in real-time based on the measurement noise statistical estimator. As opposed to the traditional UKF, where the covariance matrix R^* is a constant number, R_0, and can not deal with the fault tolerance problem.

$$R_0 = \sigma_{mag}^2 I_{3\times3} \tag{5.27}$$

σ_{mag} is the variance of the magnetometer noise, and it is a constant number, which is the base of the updated covariance matrix R^*:

$$R^* = \gamma_R R_0 \tag{5.28}$$

Where γ is a positive scalar scaling factor:

$$\gamma = \frac{\hat{\sigma}_R^2}{\sigma_{mag}^2} \tag{5.29}$$

For the second residual generator, the input is the magnetic field data from the magnetometer and the acceleration from the accelerometer. When each sensor is in normal operation, the output of the QUEST q_q will go through the adaptive complementary filter with the result of the AUKF q_a to get the final quaternion q_f, as shown in figure 5.1. When one of the accelerometers or magnetometers has a fault, the gain of the QUEST in the adaptive complementary filter will be zero. The ACF is shown in figure 5.3, which fuses the quaternion q_q and q_a to provide the quaternion q_f. The q_f has higher attitude accuracy because ACF can adjust the weight of the q_q and q_a, respectively by the filter gain k. The q_q is compared with the most recent q_f to get the quaternion error: $e(t) = q_q(t) - q_f(t)$. The error $e(t)$ is multiplied by the Gain k, which is used to correct the quaternion q_a. The Laplace transform is used to analyze the adaptive complementary filter. When the input of the q_a is zero, the transform function from q_q $Q_q(s) = \mathcal{L}q_q(t)$ to the final quaternion: $Q_f(s) = \mathcal{L}q_f(t)$ is:

$$H_q(s) = \frac{Q_f(s)}{Q_q(s)} = \frac{k}{s+k} \tag{5.30}$$

When the input of the q_q is zero, the transform function is:

$$H_a(s) = \frac{Q_f(s)}{Q_a(s)} = \frac{k}{s+k} \tag{5.31}$$

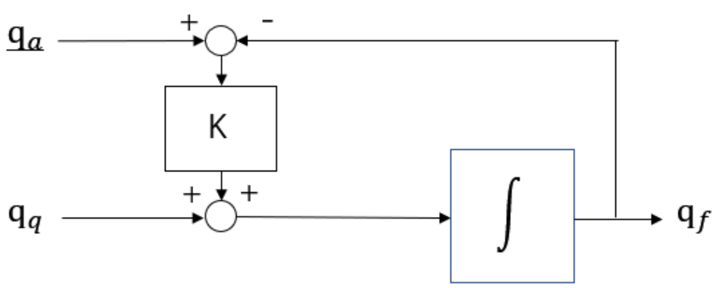


FIGURE 5.3: Block diagram of the adaptive complementary filter based on the quaternion

5.2.4 The Second Recovery System

After the first recovery system, there is the secondary recovery system, which is based on the t outputs of the RBF neural network and the output of the first recovery system. In the no-fault case, the RBF neural network is used to train the model. It has three layers: the input layer, the hidden layer, which applies a nonlinear transformation, and the output layer. Once a fault is detected, the RBF stops training the model and starts producing predicted results based on the trained model. The output of RBF can be described as:

$$y_k = \sum_{i=1}^{N} \omega_{ik} \varphi(x, c_i) \tag{5.32}$$

Where x is the m dimension input signals; ω_{ik} is the synaptic weights from the hidden layer to the output layer; $\phi(x, c_i)$ is the radial basis active function with the center c_i; N is the number of hidden layers. The radial basis active function is based on the Gaussian function:

$$\varphi(x, c_i) = G(x, c_i) \tag{5.33}$$

Where,

$$G(x, c_i) = exp(-\frac{1}{2\sigma_i^2} \|x - c_i\|^2) \tag{5.34}$$

Where σ is the width of the Gaussian function. In the proposed method, K is the clustering algorithm used for calculating the value of the center c_i. From the input layer to the hidden layer, the radial basis function is:

$$Z_j = exp(-\left\| \frac{x - c_j}{d_j} \right\|^2) \tag{5.35}$$

Where d_j is the width of the neuron, and it can be defined as:

$$d_j = \frac{c_{max}}{\sqrt{2h}} \tag{5.36}$$

The c_{max} is the maximum distance between centers, and h is the number of cluster centers. From the hidden layer to the output layer, the function is seen in (5.31) and ω_{ik} can be computed by the Gradient descent:

$$\omega_{ik}(t) = \omega_{ik}(t-1) - \eta \frac{\partial E}{\partial \omega_{ik}(t-1)} + \alpha[\omega_{ik}(t-1) - \omega_{ik}(t-2)] \tag{5.37}$$

where η is the learning rate and E is the loss function. Then the output of the RBF will go through the Hypothesis testing algorithm with the output of the AUKF and QUEST, respectively, which is shown in figure 5.4. Figure 5.4 shows the multiple model structure

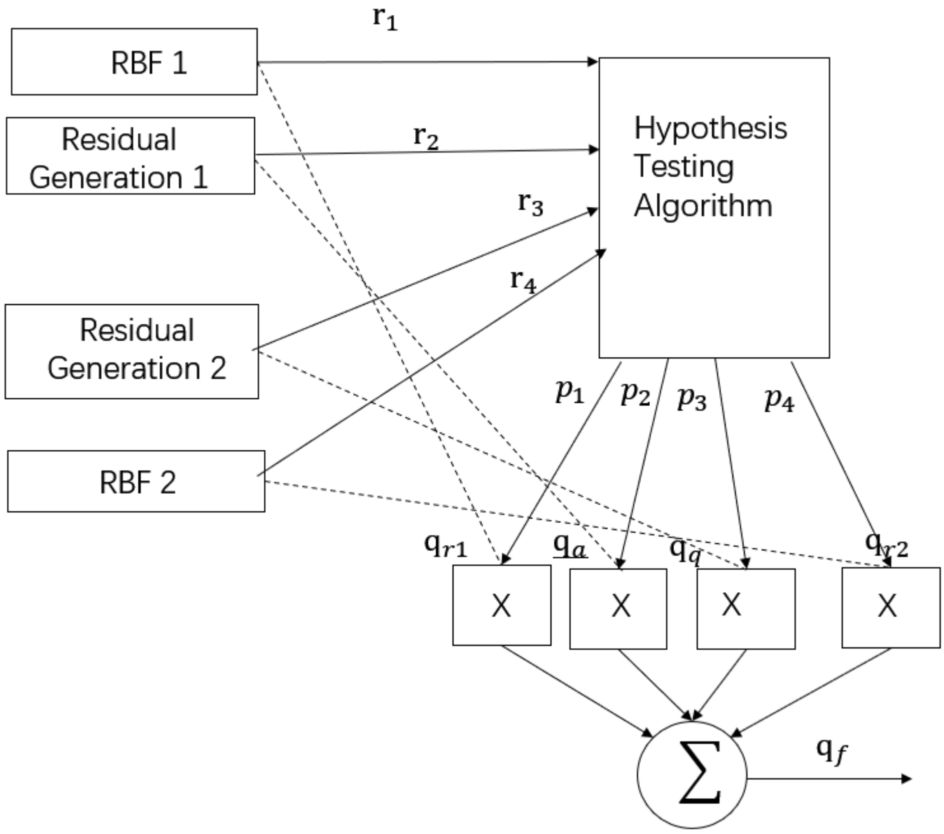


FIGURE 5.4: Multiple Model adaptive estimation structure with hypothesis testing algorithm

adaptive estimation with the hypothesis test. In figure 5.4 r_i is the residual error between the final quaternion q_f and the quaternion from different estimators, such as q_q, q_a,q_{r1}, and q_{r2}. The output of the estimator can be defined as:

$$\hat{q}_{fi} = \int_x xp(x|z_i^*)\mathrm{d}x \tag{5.38}$$

Where x is the system rate, which is q_a, q_f, q_{r1}, q_{r2}, z_i^* represents the measurements up to the time t_i and $p(x|z_i^*)\mathrm{d}x$ is the probability density function of x given z_i^*. According to the traditional probability, which is based on the measurements z_i^* and the filter parameter

α,(5.37) can be represented as:

$$\hat{q}_{fi} = \int_x x \int_\alpha p(x|z_i^*)\mathrm{d}\alpha\mathrm{d}x \tag{5.39}$$

According to the Bays theorem:

$$p(x, \alpha|z_i^*) = p(x|\alpha, z_i^*)p(\alpha|z_i^*) \tag{5.40}$$

Where $p(\alpha|z_i^*)$ is calculated by:

$$p(z_i^*|\alpha) = \beta e^{\{\bullet\}} \tag{5.41}$$

Where the $\beta = \frac{1}{(2\pi)^{\frac{1}{2}}|A_i|^{\frac{1}{2}}}$, the $\{\bullet\} = \{-\frac{1}{2}r_i^T(t_i).A_i^{-1}r_i(t_i)\}$ A_i being the covariance matrix of the computed residual r_i. Substituting (5.39) into (5.38), it gives:

$$\hat{q}_{fi} = \int_\alpha p(\alpha|z_i^*) \int_x p(x|\alpha, z_i^*)\mathrm{d}\alpha\mathrm{d}x = \int_x p(\alpha|z_i^*)\hat{q}_{fi}\alpha\mathrm{d}x \tag{5.42}$$

5.3 Experiment of Fault Isolation and Recovery

In this section, the proposed algorithm is implemented on an air-bearing table. The motion-tracking system is used to provide the reference attitude information, which is shown in figure 5.5.

In this experiment, a torque-free air-bearing table with three degrees of freedom is used, excluding extreme angles due to the platform constraints. The LSM9DS1 is a system-in-package featuring a 3D digital linear acceleration sensor, a 3D digital angular rate sensor, and a 3D digital magnetic sensor. The motion tracking system includes four OptiTrack cameras that capture the air-bearing table's attitude information. In this experiment, the results of the AUKF, QUEST algorithm, and the proposed algorithm are compared with the reference attitudes from the motion tracking system. In addition, there are three different scenarios: the gyroscope (scenario 1), magnetometer (scenario 2), and accelerometer (scenario 3), which are faulty individually. The failure point starts at the 700^{th} steps and finishes at the 1122^{th} steps. When each sensor is in the normal state, the Euler Angle and the quaternion from the AUKF, QUEST algorithm, and proposed algorithm are shown in figure 5.6(a) and figure 5.6(b),

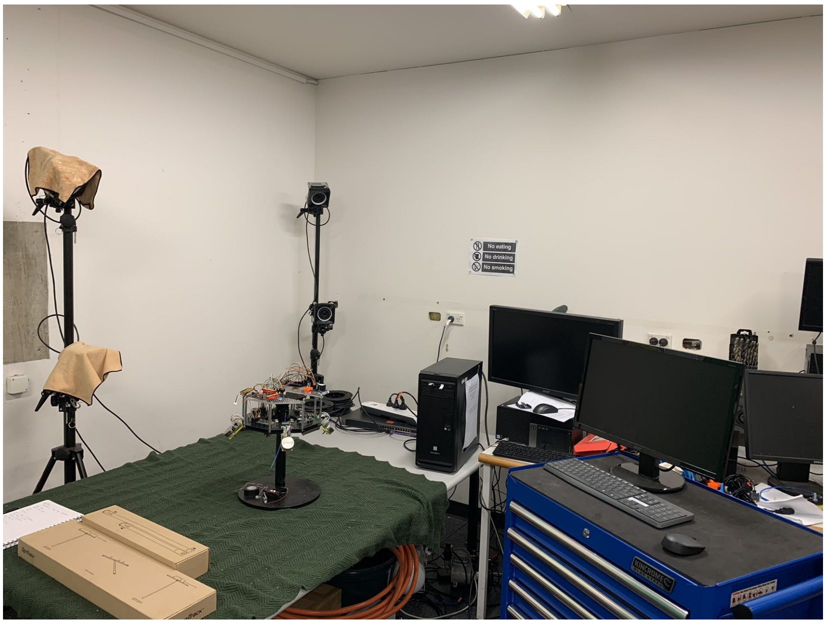


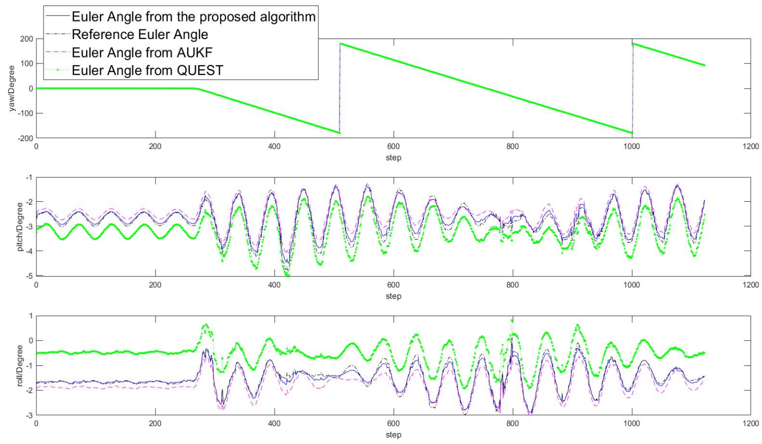
FIGURE 5.5: The Air-bearing Table and Motion Tracking System

respectively. According to figure 5.6, the proposed algorithm produces the best results for yaw, pitch, and roll angle, followed by the AUKF and the QUEST with the lowest accuracy, and the detailed data is presented in Table 5.2.

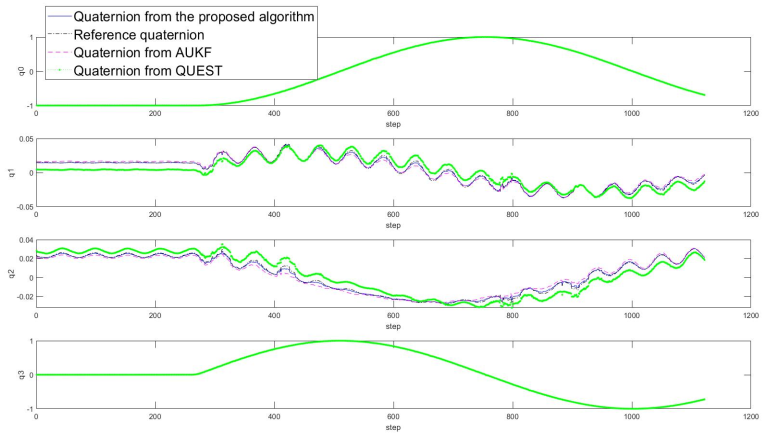
TABLE 5.2: The detailed absolute error of different algorithms in standard situation

Euler Angle/deg	Proposed Algorithm	AUKF	QUEST
Yaw	0.049562	0.063	0.094
Pitch	0.2269	0.2455	0.4635
Roll	0.15344	0.2457	0.2792

In scenario 1, the standard output and failure output of the gyroscope are both shown in figure 5.7. The failure starts from the 700^{th} point and then remains at zero until the experiment is completed. figure 5.8 (a) and (b) indicate the Euler Angles representations of the quaternion of this scenario. The recovery results of the AUKF, QUEST algorithm and the proposed algorithms are also shown in figure 5.8 (a) and (b). Figure 5.8(c) presents the errors of these algorithms. For this scenario, the fault is limited to the gyroscope. The QUEST output is the same as the output of the standard scenario because the input data are acceleration and magnetic field. From figure 5.8(a) and (b), the AUKF and the proposed algorithm can both



(a) The Euler Angle of Different Algorithms



(b) The Quaternion of Different Algorithms

FIGURE 5.6: The Euler Angle and Quaternion from Different Algorithms

recover the attitude from the 710^{th} step. However, for the yaw Angle, the recovery result of the AUKF shows a lag to the reference yaw angle.

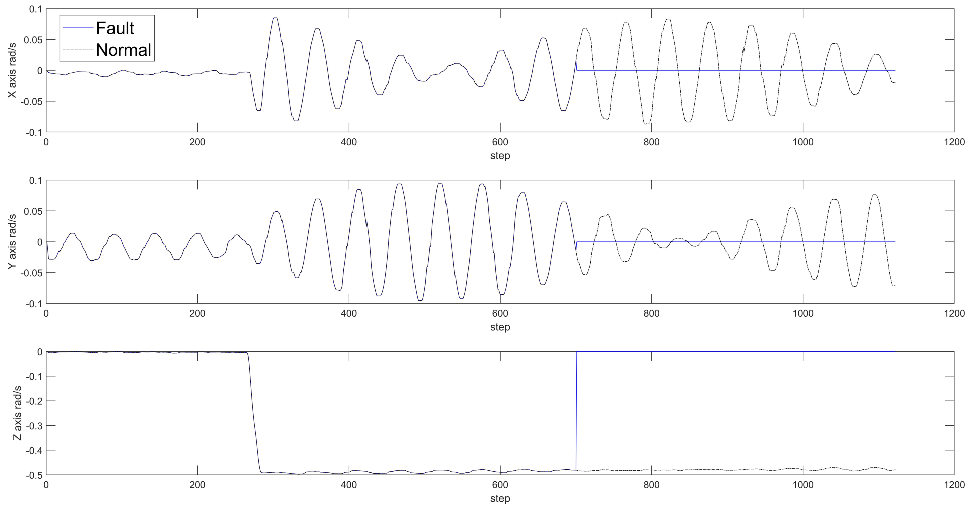
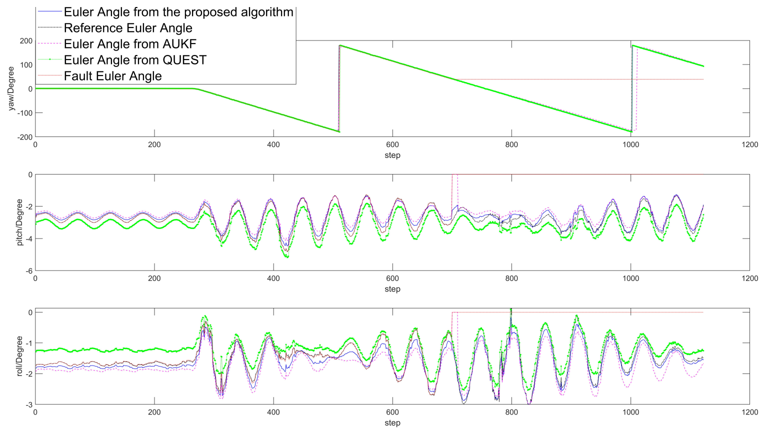
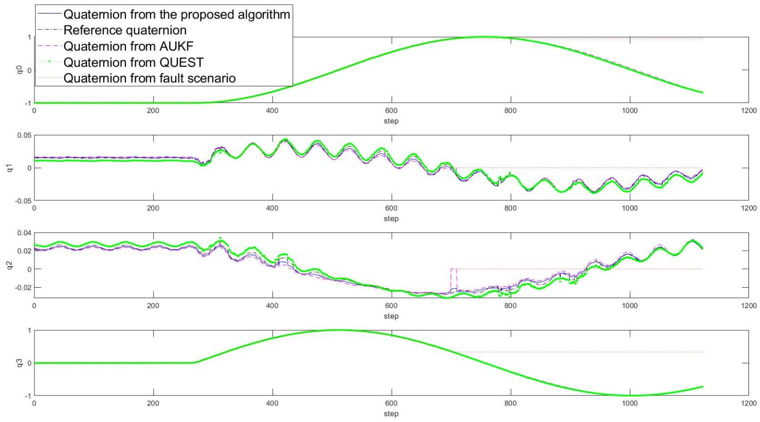


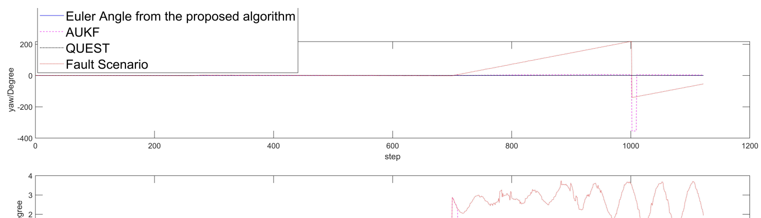
FIGURE 5.7: The output of standard and fault gyroscope



(a) The Euler Angle of Different Algorithms



(b) The Quaternion of Different Algorithms



The Euler angles from the AUKF, from the 700^{th} to the 710^{th} step, maintains normal operation as it is still within the bounds of the threshold. However, at the 710^{th} step the threshold of 0.06 is exceeded. For the proposed algorithm, the result fuses the quaternion from AUKF, QUEST, RBF1, and RBF2. Therefore, from the 700^{th} step to the 710^{th} step, the error at the pitch angle is around 0.5 degrees, and the error at the roll angle is about 0.1 degrees. Before the fault was detected, the accuracy of each algorithm is the same as the standard scenario, as shown in figure 5.6, and after the recovery (710^{th} step). The Euler Angles and quaternion from the proposed algorithm are still the most similar to the reference Euler Angle and quaternion from the reference attitude. The second is the AUKF, and QUEST has the lowest accuracy, although the result of AUKF has significant fault detected from the 700^{th} to the 710^{th} step, and the delay at the 1009^{th} step, this can also be seen in figure 5.8 (c). In addition, Table 5.3 shows the detailed error of different algorithms, the attitude error from the proposed algorithm is the minimum, the yaw angle is only 0.048 degrees, while the one from the QUEST algorithm is the maximum, especially for the pitch roll angles(0.4635 degree).

TABLE 5.3: The detailed absolute error of different algorithms in scenario 1

Euler Angle/deg	Proposed Algorithm	AUKF	QUEST
Yaw	0.048	0.9473	0.094
Pitch	0.1189	0.2787	0.4635
Roll	0.1123	0.2398	0.2792

Figure 5.9 shows the residual vectors of the fault, AUKF and QUEST algorithm. The residuals, $v_{r1} = [v_{r11}\ v_{r12}\ v_{r13}\ v_{r14}]$, $v_{r2}\ [v_{r21}\ v_{r22}\ v_{r23}\ V_{r24}$, and $v_{r3} = [v_{r31}\ v_{r32}\ v_{r33}\ v_{r34}]$, represent the vectors of the quaternion from the AUKF, the QUEST, and the fault scenario respectively. The vectors of v_{r1} and v_{r4} in the AUKF are over the threshold 0.06 from the 700_{th} step. After the 710^{th} step, the AUKF performs the real-time update of the covariance matrix R^{*} to recover the attitudes. The quaternion vector from the QUEST is in the nominal state because the input data of the QUEST algorithm are from the accelerometer and magnetometer. Therefore, according to the logic table, the fault is from the gyroscope.

In scenario 2 the magnetometer has a fault. From the 700^{th} step, the Euler angle will not change until the end of the experiment, which is shown in figure 5.10. Figure 5.11 (a), (b) and

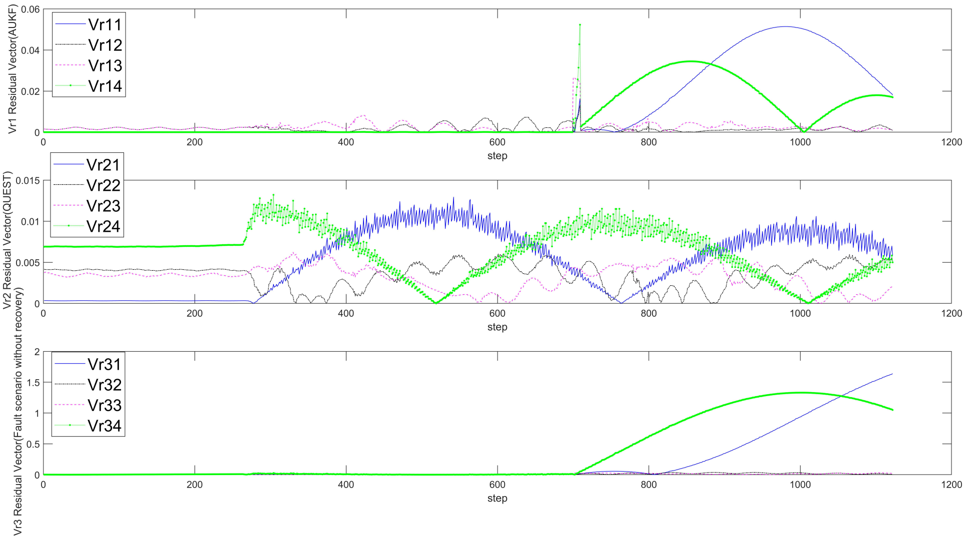


FIGURE 5.9: The Residual vector of the AUKF and QUEST algorithm

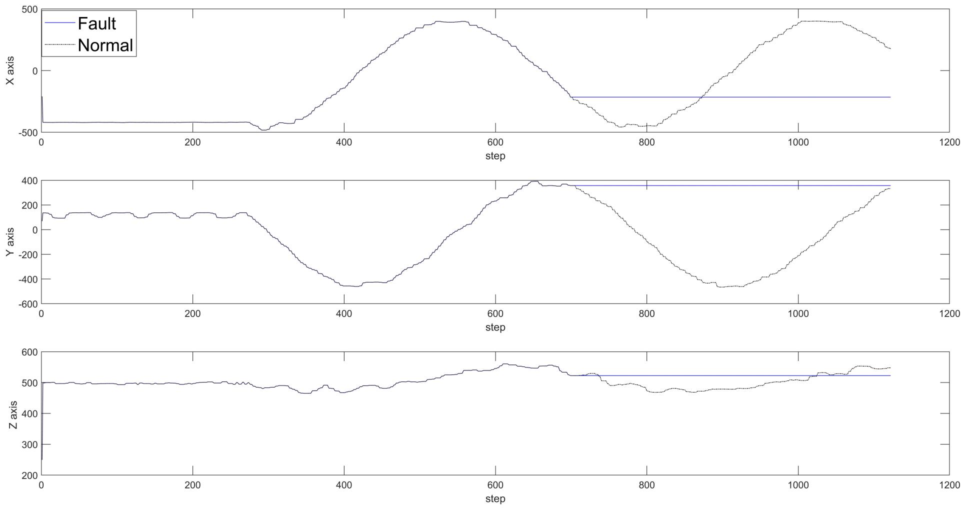


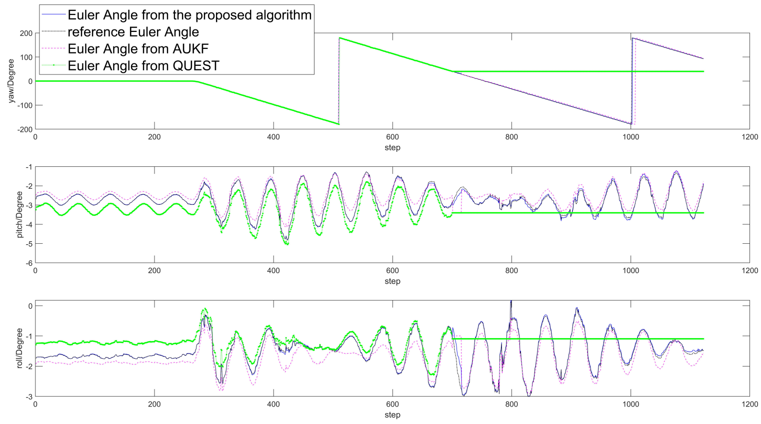
FIGURE 5.10: The Residual vector of the AUKF and QUEST algorithm

(c) are the Euler angles, quaternions, and errors of the algorithms when the magnetometer fails at the 700^{th} step. From figure 5.11 (a) and (b), the Euler angles and quaternions are the same as the result of the standard scenario. After the 700^{th} step, the result from the AUKF and the QUEST are always the same data because the magnetic field data are both the input of the AUKF and the QUEST. AUKF has the recovery function based on updating the covariance matrix R^{*}. However, QUEST does not have the capability, because it is based on real-time measurement data. Therefore, in figure 5.11 (a), the Euler angle of the attitude from the QUEST algorithm keeps the same data after the fault point (the 700^{th} step). However, the attitude of AUKF keeps the fault Euler angle from the 700_{th} step to the 716^{th} step because the residual exceeds the threshold 0.1 at the 716^{th} step, which is shown in figure 5.12. After the 716^{th} step, the result of the AUKF starts to recover to normal attitudes. However, the accuracy is lower than the proposed algorithm. figure 5.11 (c) shows the error of the Euler angle from different algorithms, and the detailed data is displayed in Table 5.4. It can be found that the error from QUEST is the largest the Yaw angle is up to 19.5250 degrees because the QUEST cannot recover fault. The nest is the AUKF, and the minimum error is from the proposed algorithm. Furthermore, because QUEST lacks a recovery mechanism, the error after the $700th$ step is rather substantial.

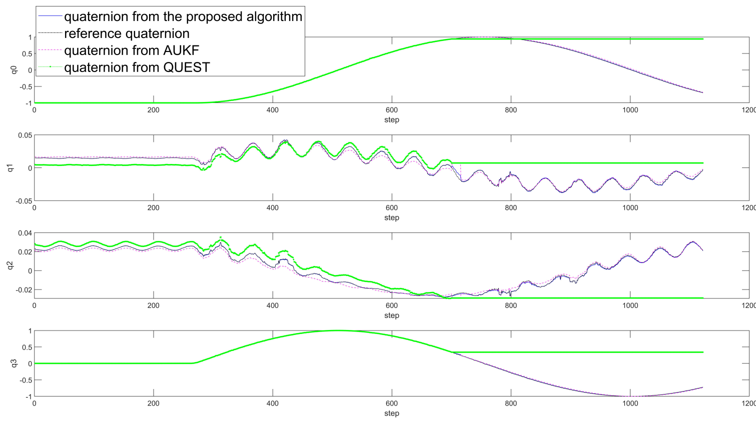
TABLE 5.4: The detailed absolute error of different algorithms in scenario 2

Euler Angle/deg	Proposed Algorithm	AUKF	QUEST
Yaw	0.0518	0.5185	19.5250
Pitch	0.0073	0.2285	0.5463
Roll	0.021	0.2031	0.3295

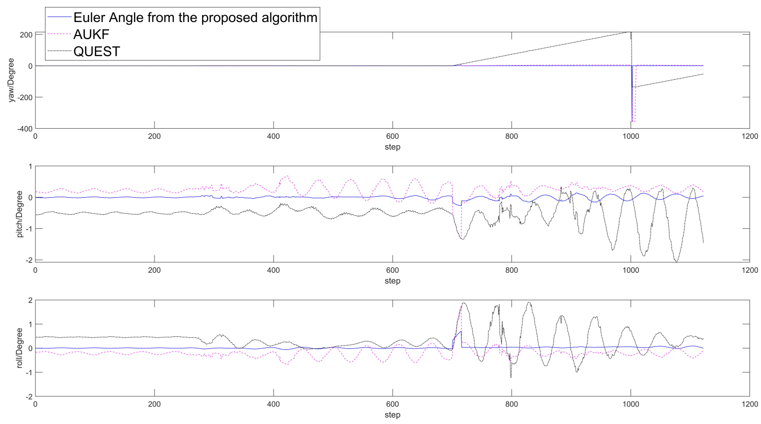
Figure 5.12 shows the residual vectors from the AUKF and QUEST algorithms. From figure 5.12, both the residual vector from the AUKF and the QUEST exceed the threshold of 0.1. Therefore, the AUKF and QUEST both have fault signals. From the logic table, the fault can therefore be determined to be the magnetometer.



(a) The Euler Angle of Different Algorithms



(b) The Quaternion of Different Algorithms



(c) The Error of Different Algorithms

FIGURE 5.11: The Euler Angle and Quaternion,and Error from Different Algorithms

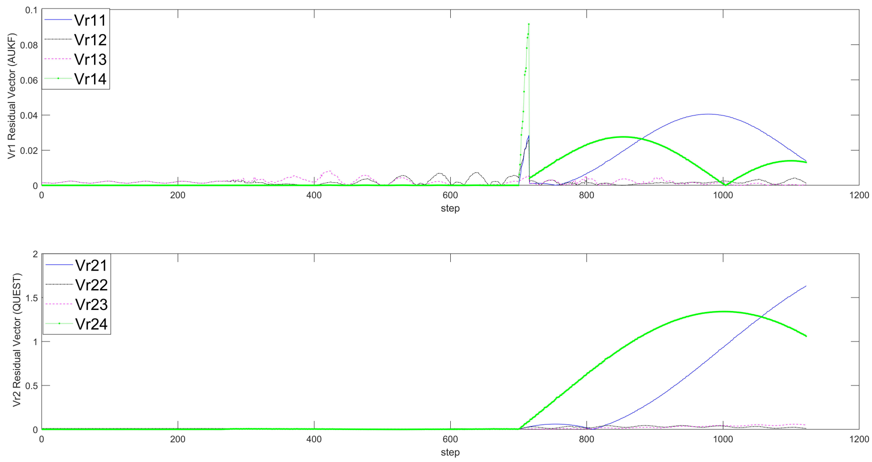


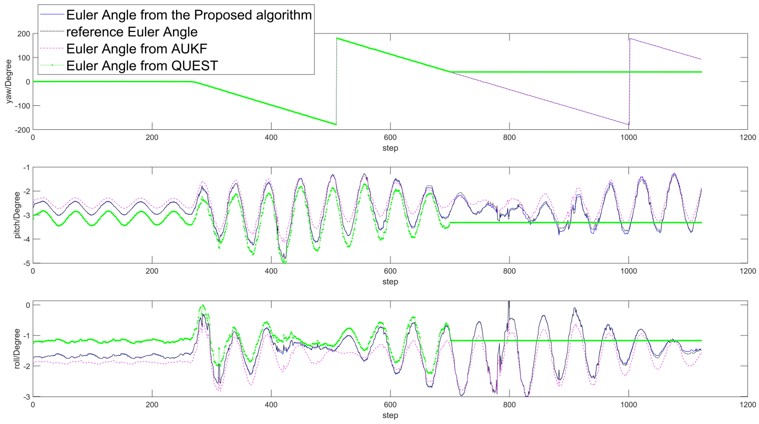
FIGURE 5.12: The Residual vector of the AUKF and QUEST algorithm

In scenario 3 the accelerometer is faulty from the 700^{th} step. figure 5.13 (a), (b), and (c) show the Euler angles, quaternions, and errors from different algorithms. From figure 5.13(a) and Fig.13 (b), the output of the AUKF is correct because the input of the AUKF is the angular rate from the gyroscope and the magnetic field from the magnetometer. The acceleration is only used as an input by the QUEST algorithm. Therefore, the attitude from QUEST keeps the same data from the 700 step. The proposed algorithm fuses the attitude from the AUKF, the RBF1 by the adaptive complementary filter to get the final attitude information, which has the highest accuracy among the algorithms, which is shown in figure 5.13 (c), and the detailed data is in the Table 5.5. It can be found, the error of the proposed algorithm is minimal, compared to the large error from the QUEST algorithm(the error of the yaw angle is 19.4561 degrees). From the experimental results, the proposed algorithm is proven for fault detection and recovery. Furthermore, compared with the AUKF, the proposed algorithm has higher accuracy, especially during the between after fault and finishing fault detection. The dynamic threshold also reduces the false alarm rate.

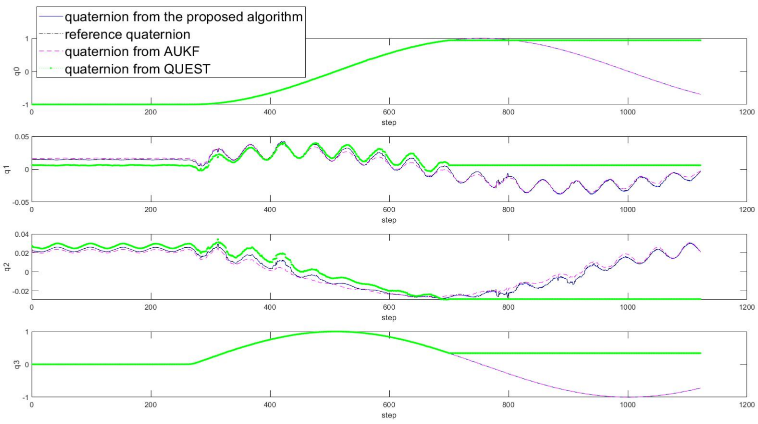
TABLE 5.5: The detailed error of different algorithms in scenario 3

Euler Angle/deg	Proposed Algorithm	AUKF	QUEST
Yaw	0.05109	0.063	19.4561
Pitch	-0.072	0.2455	-0.4619
Roll	0.0058	-0.2457	0.3454

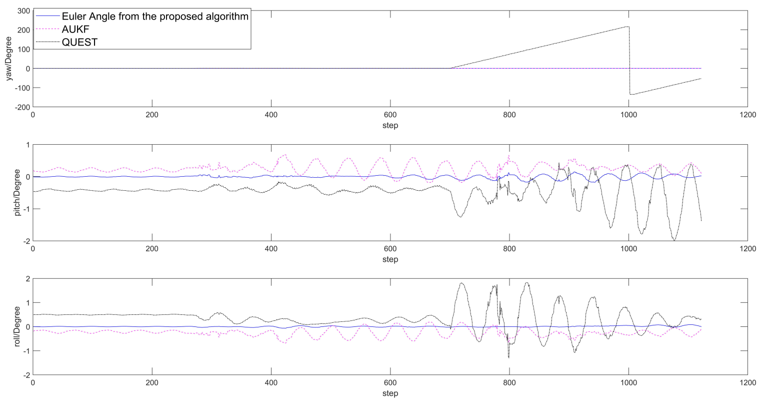
Figure 5.14 shows the residual vector from the AUKF and QUEST algorithms. The residual vector of the AUKF is always smaller than the threshold of 0.05. Therefore, there is no fault in AUKF. However, for the QUEST algorithm, the residual vector exceeds the threshold 0.05 at the 700^{th} step. From the logic table rule. the fault is at the accelerometer.



(a) The Euler Angle of Different Algorithms



(b) The Quaternion of Different Algorithms



(c) The Error of Different Algorithms

FIGURE 5.13: The Euler Angle and Quaternion,and Error from Different Algorithms

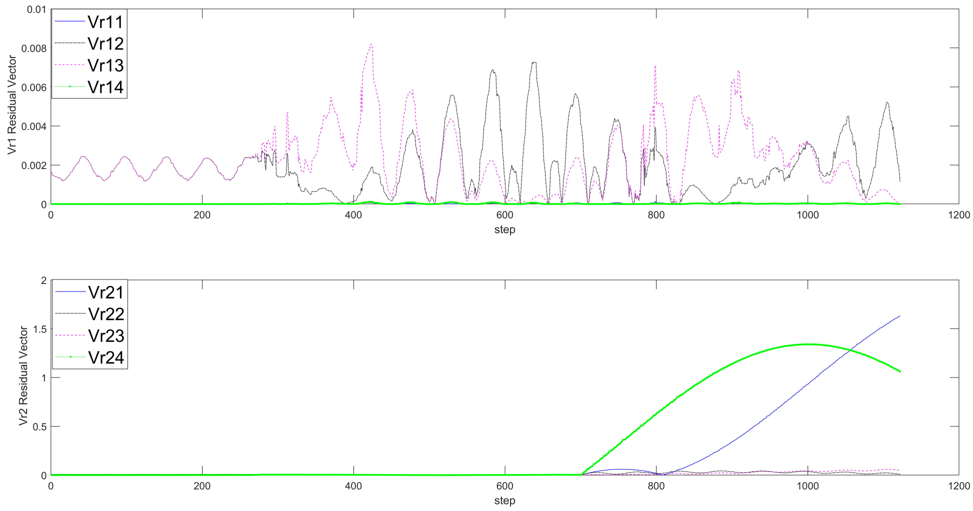


FIGURE 5.14: The Residual vector of the AUKF and QUEST algorithm

5.4 Summery

This chapter presented a novel method to detect, isolate and recover faults of an IMU as part of an onboard satellite ADCS system. By fusing the outputs of AUKF, QUEST and two RBF neural networks based on an adaptive complementary filter and hypothesis test to achieve the fault-tolerant and high-accurate attitude estimations. The preliminary recovery phase uses an AUKF as fault detection as well as providing the ability to recover attitudes. The secondary recovery phase uses trained neural networks to provide an estimated attitude. This multi-level recovery strategy allows the satellite ADCS system to maintain a reasonable accuracy even with a sensor fault. In addition, the proposed algorithm offers a higher accuracy when compared to methods such as the AUKF and QUEST algorithms for each sensor failure when tested on an experimental platform. Furthermore, the secondary fault detection and isolation layer provides lower false alarm rates, resulting in a more reliable satellite attitude estimation solution. In conclusion, the highlights of this part:

(1) This Chapter develops a novel algorithm, which is based on the AUKF and RBF neural network, to realize fault detection, isolation, and recovery for satellite attitude estimation.

(2) The hypothesis testing algorithm follows the residual generators to adjust the weight of the AUKF, QUEST, and RBF in order to decrease the false alarm rate.

(3) Compared with AUKF, the double fault recovery strategy has better precision for satellite attitude estimation.

.

CHAPTER 6

The Fault Recovery Based on Fault-tolerant Federated Kalman Filter for INS/CNS

This chapter describes a fault-tolerant federated kalman filter for INS/CNS. The work was finished and prepared to submit to the IEEE Sensor Journal. First, a novel scheme of fault-tolerant federated kalman filter is given to solve the failure in the INS/CNS. The following sections present the theory and details of this algorithm. Finally, the implementation analysis and test results are given to verify the validity of the algorithm.

6.1 Introduction

Chapter 5 of this thesis focuses on FDIR in the satellite attitude determination system (ADS), specifically in the FDIR of the INS. However, most satellites do not rely solely on the INS for high-precision attitude determination but also employ the CNS. To enable fault detection and recovery in both subsystems of the ADS, this chapter proposes a fault-tolerant federated Kalman filter (FTFKF) that integrates INS and CNS data.

The FTFKF scheme consists of two subsystems based on a common gyroscope model. The first subsystem uses the QUEST algorithm to fuse data from the accelerometer and magnetometer, while the second subsystem employs data from the star tracker. These subfilters operate in parallel, making them suitable for FPGA-based parallel optimization. By fusing attitude information from INS and CNS, the proposed scheme improves the accuracy of ADS attitude estimation and allows for online fault detection and timely fault tolerance. Under normal conditions, ADS attitude estimation relies on data from both INS (gyroscope, accelerometer, and magnetometer) and CNS (star tracker). In the event of a fault, the

FTFKF analyzes dimensionless fault factors to identify the faulty subsystem and selectively fuses the output of the unaffected subsystem to restore attitude estimation accuracy. This approach enhances the reliability of the ADS and provides more accurate and reliable attitude information compared to relying on only INS and CNS data. Additionally, in case of a false alarm, the FTFKF has a fault confirmation process. which can decrease the false alarm rate.

6.2 Traditional Satellite Attitude Determination System Model

Traditional satellite attitude determination relies on the gyroscope and star tracker for accurate orientation measurements. The gyroscope provides high-precision reference information for short-term measurements. However, over long periods, it suffers from gyro drift bias, leading to significant errors that accumulate over time. To correct for this, real-time correction from the star tracker is necessary. As illustrated in figure 6.1, the output of the star tracker, denoted by s, and the gyroscope data are used as input into the attitude calculation process to obtain the computed quaternion q_s. Simultaneously, the angular velocity from the gyroscope is integrated to yield the approximation quaternion $q_a p$. The Kalman Filter is then utilized to fuse the q_s and $q_a p$ to obtain the estimated quaternion q_e.

The state quaternion is based on the satellite kinematic model, providing a reliable and accurate orientation measurement for the satellite:

$$\dot{q} = \frac{1}{2}\Xi(q)\omega = \frac{1}{2}\Omega(\omega)q \tag{6.1}$$

The relationship between the angular velocity ω and quaternion derivative $\dot{q}$ is:

$$\dot{q} = \frac{1}{2} \times q \otimes (\omega + \omega^n) \tag{6.2}$$

The output quaternion, denoted as q_s, is obtained from the star tracker, a sophisticated device capable of measuring spacecraft attitude with high precision by utilizing stars as a reference source. The star tracker generates directional vectors in star-tracker coordinates, which serve

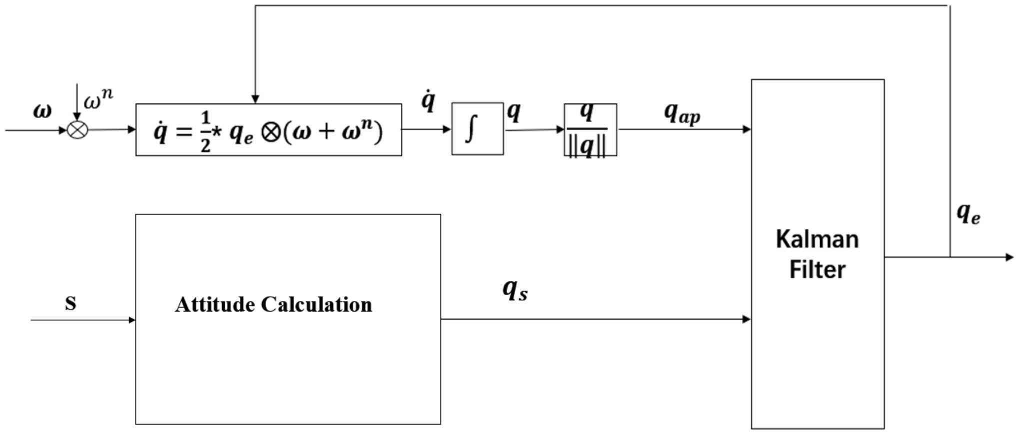


FIGURE 6.1: The traditional satellite determination system

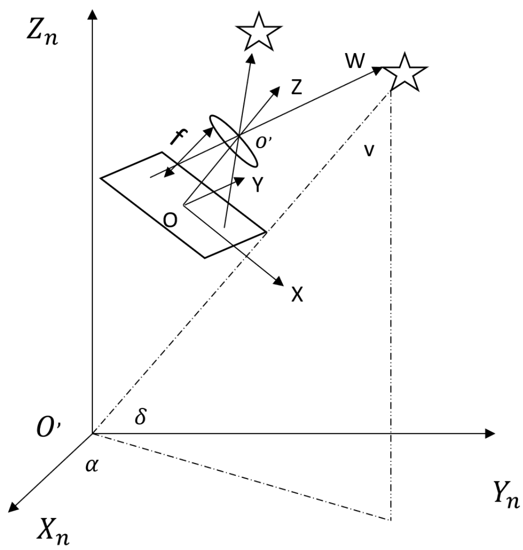


FIGURE 6.2: The measurement model of star tracker

as a fundamental component in the attitude determination systems of satellites. Specifically, the measurement model of the star tracker can be modeled as a pinhole imaging system, as illustrated in Figure 6.2, which enables accurate and reliable measurements. This traditional system has been widely adopted in spacecraft attitude determination.

Figure 6.2 presents the star's direction vector will be represented in the star tracker reference. The v and w represent cataloged vectors in the inertial frame $O' - X_nY_nZ_n$ and the direction vector in the star tracker frame $o^{'} - xyz$, respectively, and they can be described as:

$$v = \begin{bmatrix} \cos\alpha\cos\delta \\ \sin\alpha\cos\delta \\ \sin\delta \end{bmatrix} + V_s \tag{6.3}$$

and

$$w = \frac{1}{\sqrt{x^2 + y^2 + f^2}} \begin{bmatrix} -x \\ -y \\ f \end{bmatrix} \tag{6.4}$$

where α and δ are the right ascension and declination of the associated guide star on the celestial sphere, f represents the focal length of the camera in the star tracker. $xandy$ are star spot locations in the detector plane. In addition the direction vector w and v should follows:

$$w = H(q)v = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 + q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} v + V_s \tag{6.5}$$

where the $q = [q_0 \ q_1 \ q_2 \ q_3]$ are four components of quaternion, and the V_s is the sensor measurement noise.

6.3 Fault-tolerance Federated Kalman Filter

6.3.1 Traditional Federated Kalman Filter

The traditional federated Kalman filter (FKF) is able to fuse measurement information from different sensors reliably and effectively. The traditional architecture FKF is shown in figure 6.3. This system includes INC, CNS and GNSS. It is a partitioned estimation method with a two-level data processing architecture. On the first level, The INS is integrated with CNS and GNSS respectively, by two local filters, which are always Kalman Filter. These two standard Kalman filers follow;

$$\begin{cases} X_{k+1}^{-} = \phi_k X_k + w_k, \\ P_{k+1}^{-} = \phi_k X_k \phi_k^T + Q_k, \\ K_{k+1} = P_{k+1}^{-} H_{k+1}^T (H_{k+1} P_{k+1}^{-} H_{k+1}^T + R_{k+1})^{-1}, \\ X_{k+1} = K_{k+1}(X_{k+1}^{-} - X_k), \\ P_{k+1}^{-} = (I - K_{k+1}) P_{k+1} \end{cases} \tag{6.6}$$

The standard KF is comprised of three primary steps: Projection, Kalman Gain, and Update. In this context, ϕ represents the state transition matrix, H represents the observation matrix, P_{k+1}^{-} denotes the priori error covariance matrix, P_{k+1} represents the a posteriori state error covariance matrix, K represents the Kalman gain, and Q and R represent the process noise covariance matrix and the measurement noise covariance matrix, respectively.

In the traditional FKF, two local filters operate in parallel, which enhances calculation efficiency. The second filter utilizes the two local state estimations as input for the main filter, resulting in global state estimation. The master filter aggregates all estimation information from the different sub-filters to achieve a global result and then distributes the information to each sub-filter in accordance with the principle of information sharing. This process enables feedback of attitude information to each sub-filter, ultimately resetting their initial attitude estimation and covariance matrix. Therefore, the whole process can be summarized as two parts, information distribution, and information fusion.

In the figure 6.3, the $\hat{X}_G$ is the output information of the master filter, the $\hat{X}_1$ and $\hat{X}_2$represent the local state estimation from local filter 1 and 2 respectively. P_g is the error covariance matrix of the master filter, and the P_i represents the error covariance matrix from the different local filters.

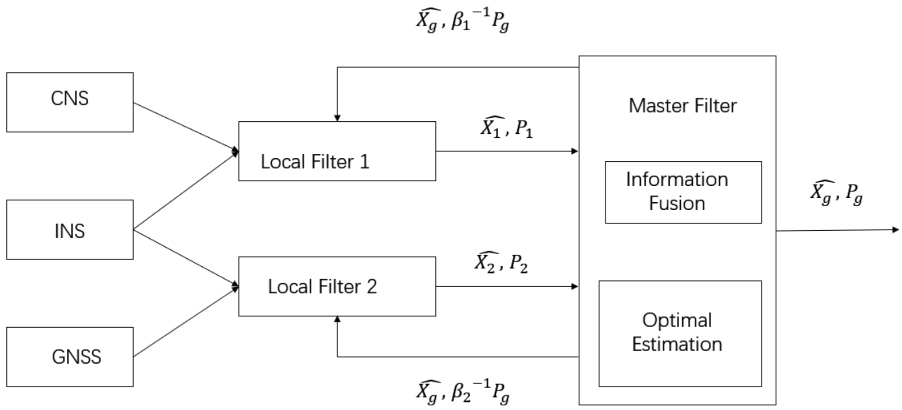


FIGURE 6.3: The architecture of FKF

6.3.1.1 Information Distribution Process

Information distribution can distribute the state estimation information P^{-1} and process noise information Q^{-1} to each sub-filter, and this process is followed by:

$$\begin{aligned} P_i^{-1} &= \beta_i P_g^{-1}(k) \\ Q_i^{-1} &= \beta_i Q_g^{-1}(k) \\ \hat{X}_i(k) &= \hat{X}_g(k) \end{aligned} \tag{6.7}$$

where β_i is the information-sharing factor coefficient. In the traditional FKF, the information-sharing factor coefficient follows the information distribution principle and is commonly determined by equal division:

$$\begin{aligned} \beta_1 + \beta_2 + \beta_n &= 1 \\ \beta_1 = \beta_2 &= \beta_n \end{aligned} \tag{6.8}$$

6.3.1.2 Information Fusion Process

The main task of the information fusion is to get the global optimal estimation through the estimation results of all sub-filters:

$$
\begin{aligned}
P_g &= (\sum_{i=1}^{n} P_i^{-1}(k))^{-1} \\
\hat{X}_g &= P_g(\sum_{i=1}^{n} P_i^{-1}(k)\hat{X}_i(k))
\end{aligned}
\tag{6.9}
$$

6.3.2 The Novel Fault-tolerant Federated Kalman filter

The traditional FKF poses challenges for satellites, particularly Cube-Sats, due to the redundancy sensors that consume space and power resources. Moreover, the traditional FKF does not provide a solution for sensor faults. To address these issues, we propose an FTFKF that only utilizes an IMU and a star tracker to determine the attitude of the satellite.

The FTFKF architecture, as shown in figure 6.4, facilitates fault tolerance in the event of an INS or CNS sensor failure. The angular velocity from the gyroscope is utilized in the public state equation of the two sub-filters. Chapter 4 describes the state equation from the gyroscope as follows:

$$q_{k+1} = \phi_k(\Omega_k, T_s)q_k + w_k \tag{6.10}$$

Where ϕ is the state transition matrix, H denotes the observation matrix, and w and v are White Gauss process noise and measurement noise. The T_s in it represents the sample time, which should be quite small, and Ω can be represented as:

$$\Omega(\omega) = \frac{1}{2}\begin{bmatrix} 0 & -\omega^T \\ \omega & \omega[\times] \end{bmatrix} \tag{6.11}$$

Where ω is the angular velocity of the gyroscope, and its size is 1×3. In addition, $\omega[\times]$ is a 3×3 skew-symmetric matrix described by:

$$\omega[\times] = \begin{bmatrix} 0 & \omega_z & -\omega_y \\ -\omega_z & 0 & \omega_x \\ \omega_y & -\omega_x & 0 \end{bmatrix} \tag{6.12}$$

Then the Taylor series can be applied to (6.10):

$$q_{k+1} = [I_{4\times4}(1 - \frac{\Delta\theta^2}{8}) + \frac{1}{2}\Omega_k T_s]q_k \tag{6.13}$$

where

$$\Delta\theta^2 = (\omega_x T_s)^2 + (\omega_y T_s)^2 + (\omega_z T_s)^2 \tag{6.14}$$

The two sub-filters both are standard KF. For the first sub-filter, the observation equation is the quaternion q_q from the QUEST algorithm. The input data of the QUEST algorithm are the acceleration a and the local magnetic field m. For the second sub-filter, the observation equation is the quaternion q_s from the star tracker directly. Then these two sub-filters output the attitude information $\hat{X}_i$, the covariance matrix P_i, and the fault detection factors of each sub-filters γ_1 and γ_2 are sent to the master filter to get the final attitude information $\hat{X}_g$ and final covariance matrix P_g. The $\hat{X}_g$ and P_f are distributed to each sun-filter according to the distribution factor β_i, which can be calculated by:

$$\beta_i(k+1) = \frac{||P_i(k)||}{\sum_{i=1}^{n}||P_i(k)||} \tag{6.15}$$

where $||P_i(k)||$ is Frobenius norm of P_i. The $||P_i(k)||$ reflects the absolute estimation error and the β_i reflects the relative estimation error.

For the fault detection of the proposed scheme, each fault detection factor γ_i is compared with the threshold γ_0. When γ_i does not exceed the γ_0, it means this sub-filter is in the safe scenario. On the other side, when the γ_i exceeds the γ_0, which means the sub-filter is in the failure scenario:

$$\gamma = \begin{cases} \leq \gamma_0 & Normal \\ > \gamma_0 & Fault \end{cases} \tag{6.16}$$

The fault detection factor γ is determined by:

$$\gamma(k+1) = \sqrt{\frac{1}{N}\sum_{i=k+2-N}^{k+1} \widetilde{Y_i}^T R^{-1} \widetilde{Y_i}/m} \tag{6.17}$$

Where $\widetilde{Y_i}$ is the residual of quaternion from each sub-filter, the calculation can be found in Eq.(5.15). When the sub-filter is in the safe scenario, the corresponding residual $\widetilde{Y_i}$ is small. While the abrupt fault appears or a serious failure accumulated by the slowly changing fault, the $\widetilde{Y_i}$ becomes significantly large. The N is the statistical number of measurements, m is the size of the quaternion and R is the sensor noise variance matrix. The γ is able to reflect the abrupt fault and slow fault. The threshold γ_0 is a critical element of fault detection. A smaller γ_0 is easier to detect the faults but also easier to lead the false alarm. On the other side, the larger γ_0 makes it harder to detect faults. The threshold γ_0 can be described as:

$$\gamma_0 = \sqrt{\frac{||\widetilde{Y}||}{\sigma^2}/4} \tag{6.18}$$

where σ is the quaternion error standard deviation of the corresponding sub-filter.The fault detection factor threshold γ_0 is associated with the σ. When the satellite is in a steady state, the residual of each sub-filter is similar to the white noise. The upper bouns is 3σ in statistical. However, the residual is also affected by the satellite hitters and the unmodelled dynamics, the upper bound is normally considered from 3σ to 5σ.

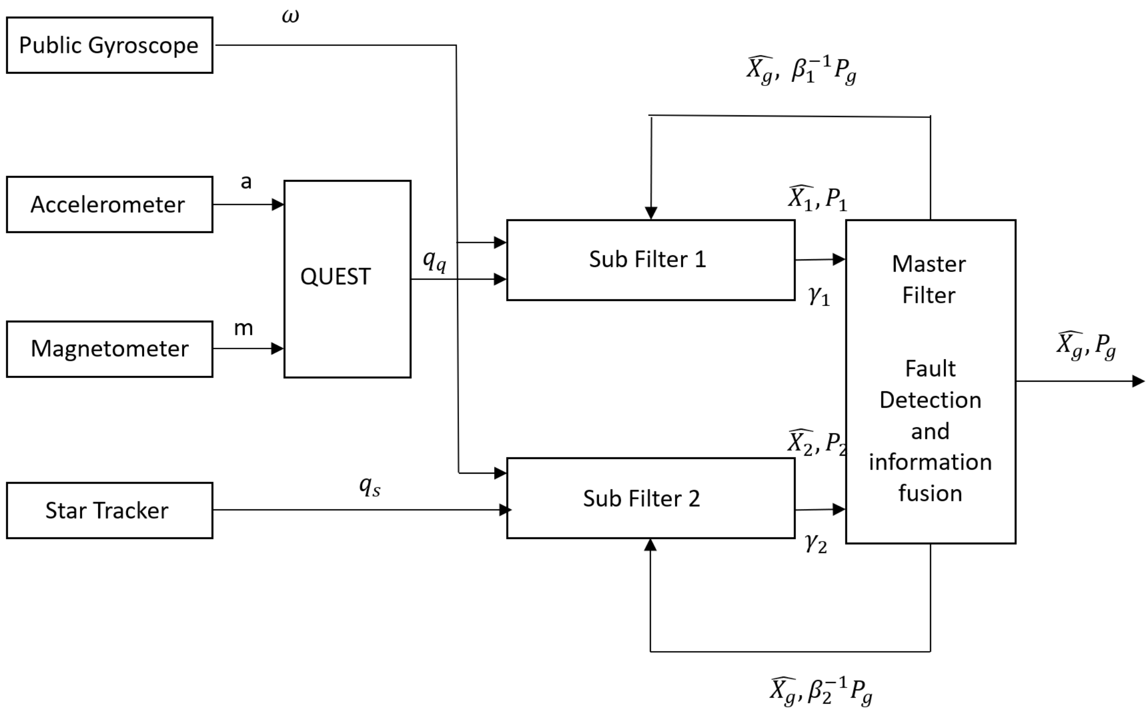


FIGURE 6.4: The architecture of the Fault tolerance federated Kalman filter

6.4 Experiment of Fault-tolerant Federated Kalman Filter

In this section, the experiment is set up on the air-bearing table with the IMU and star tracker. The air-bearing table offers torque-free motion with three degrees of freedom, excluding extreme angles due to the platform structure. The LSM9DS1 is an IMU, which is employed to track the motion of the system. The running step is around 0.036 seconds, and there are 1100 steps. It takes about 166 seconds for the whole experiment. The initial quaternion $q = [1\ 0\ 0\ 0]^T$. Furthermore, the sensor module is assembled with its x, y, and z-axis and aligned with NED (North, EAST, Down) direction. The star tracker is shown in figure 6.5, which is made by ourselves and will be assembled for the CAUVA-2. The update rate of the star tracker is 2 Hz, the field of view (FOV) is $20°$, and the image dimension is $2448 \times 2048px$. The accuracy of Roll, Pitch, and Yaw angles are $10.29^"$, $21.94^"$, and $26.24^"$, respectively.

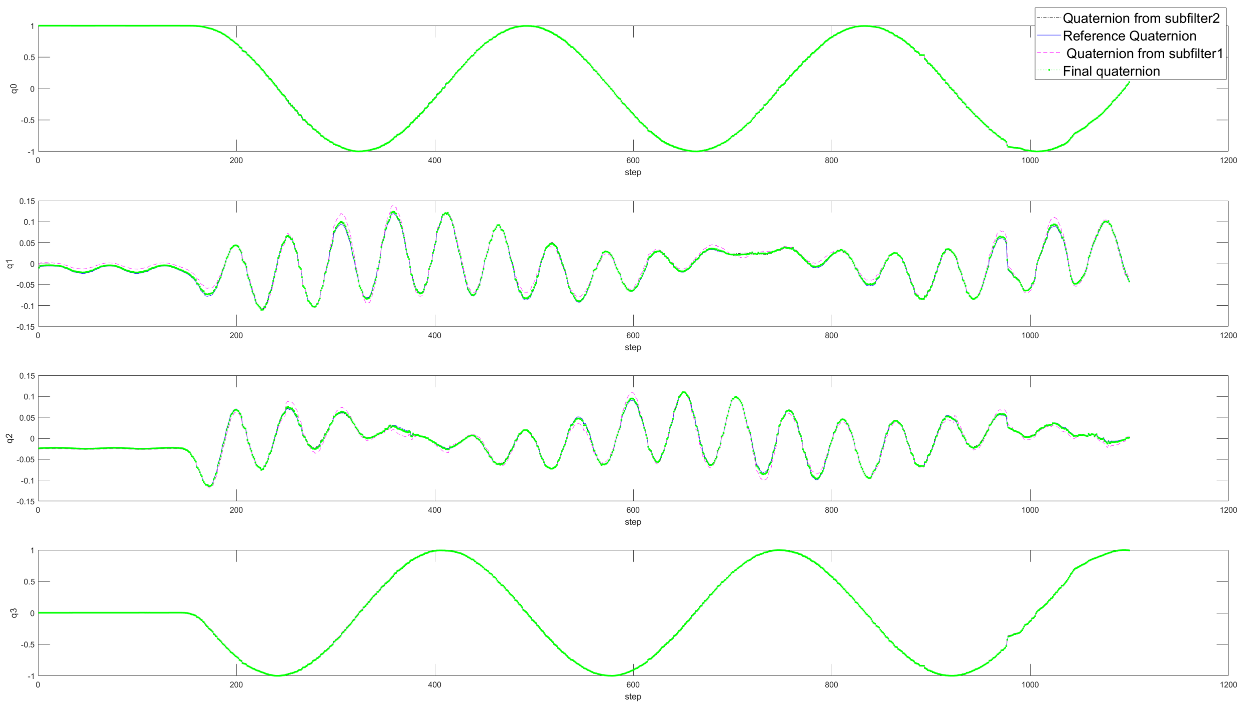


FIGURE 6.6: The quaternion of the Fault tolerance federated Kalman filter

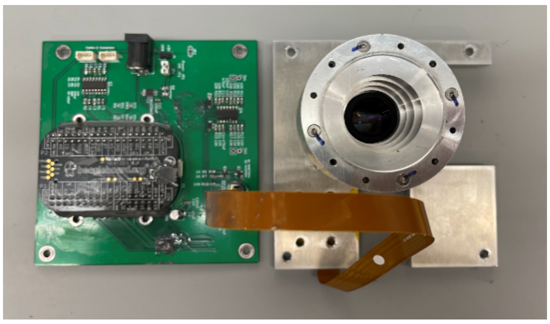


FIGURE 6.5: The Star Tracker

The different quaternions obtained from the first sub-filter, the second sub-filter, and the master filter are presented in Figure 6.6. The state equation for the first subfilter is the public gyroscope model, and the observation model is the quaternion q_q obtained from the QUEST algorithm. The state equation for the second sub-filter is also the public gyroscope model, and the measurement model is the quaternion q_s obtained directly from the star tracker. The final output of the master filter is represented by the quaternion q_f.

As shown in Figure 6.7, the quaternion from the second sub-filter q_s is the closest to the reference quaternion q_r obtained from the motion tracking system. The average errors for q_{s0}, q_{s1}, q_{s2}, and q_{s3} are 0.0012, 0.0013, 0.0013, and 0.0014, respectively. The average errors for the final quaternion (q_{f0}, q_{f1}, q_{f2}, and q_{f3}) are 0.0016, 0.0016, 0.0014, and 0.0019, respectively. They are similar to the quaternion q_s. The accuracy of the quaternion obtained from the first sub-filter is the worst, with average errors for q_{q0}, q_{q1}, q_{q2}, and q_{q3} of 0.0065, 0.0072, 0.0059, and 0.0078, respectively. Further details can be found in Table 6.1.

TABLE 6.1: The quaternion error from different filters

Quaternion	q_0	q_1	q_2	q_3
Subfilter 1(IMU)	0.0065	0.0072	0.0059	0.0078
Subfilter 2(Star Stracker)	0.0012	0.0013	0.0012	0.0014
Master Filter	0.0016	0.0016	0.0014	0.0019

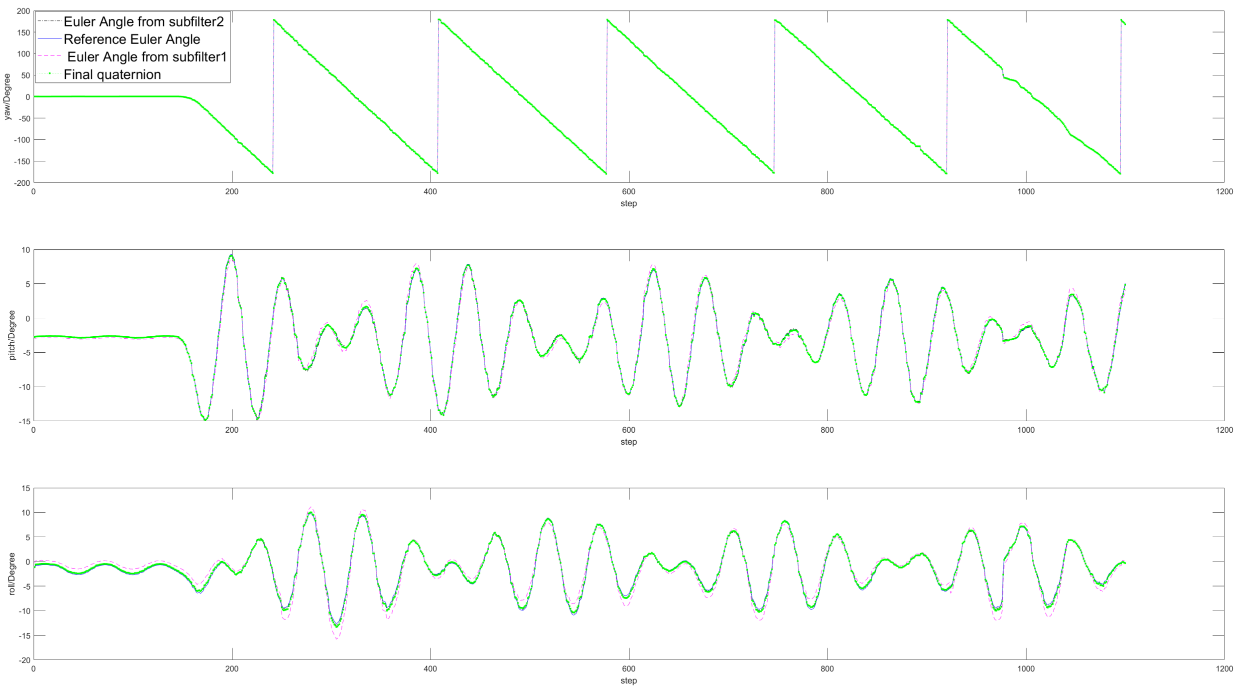


FIGURE 6.7: The Euler angle of the Fault tolerance federated Kalman filter

Figure 6.7 displays the Euler angles generated by different filters. The second sub-filter achieves the highest accuracy, followed by the Master filter, while the first sub-filter has the worst accuracy. These findings are further supported by the error analysis in Figure 6.8, which compares the errors of the Euler angles generated by different filters. Specifically, the Yaw and Roll angles exhibit the smallest errors when generated by the second sub-filter, with values of 0.2422 and 0.1679 degrees, respectively, followed by the errors of the Euler angles generated by the Master filter. In contrast, the smallest error for the pitch angle is obtained by the Master filter, with a value of 0.097 degrees, followed by the second sub-filter, which has an error of 0.1652 degrees. Furthermore, the Euler angles generated by the first sub-filter have the worst accuracy across all dimensions. The detailed error is shown in Table 6.2

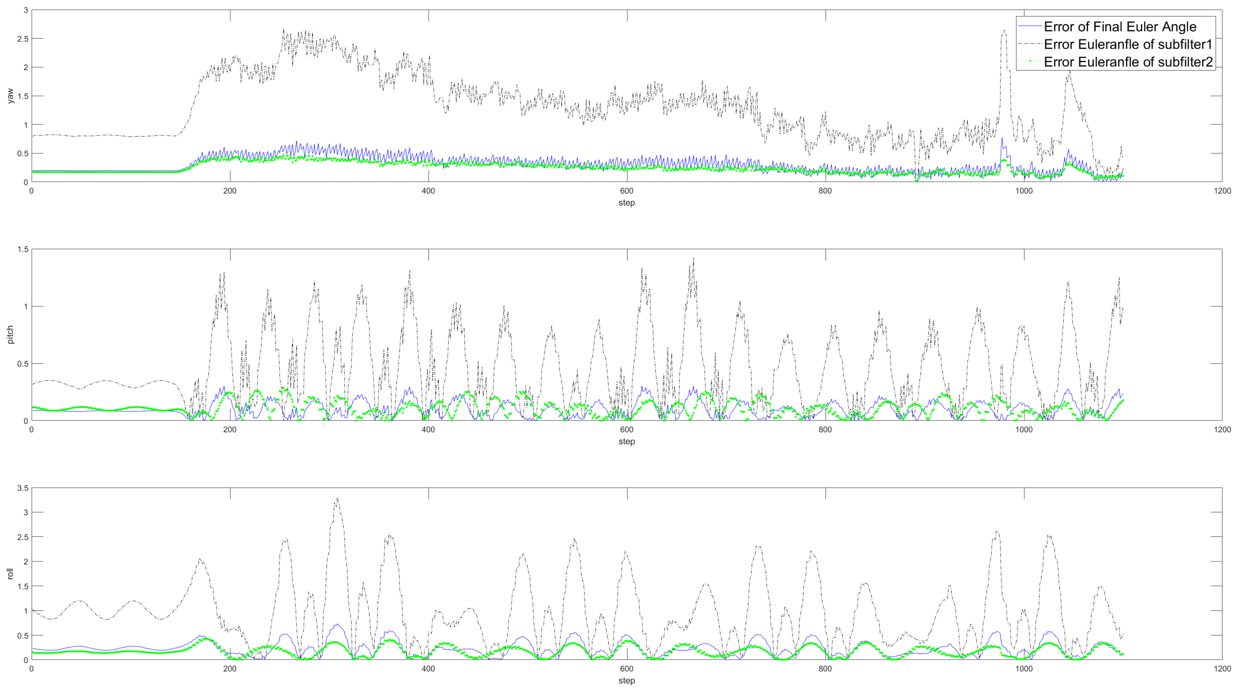


FIGURE 6.8: The Euler angle absolute error of the Fault tolerance federated Kalman filter

The process of the FDIR in INS has been introduced in Chapter 5, in the whole ADS, there are two scenarios, the fault occurs in the first sub-filter (scenario 1), and the fault is in the second sub-filter (scenario 2).

TABLE 6.2: The error of Euler angles from different filters

Euler angles/deg	Yaw	Pitch	Roll
Subfilter 1(IMU)	1.2796	0.4554	0.9894
Subfilter 2(Star tracker)	0.2422	0.1052	0.1679
Master Filter	0.3044	0.0970	0.2275

6.4.1 Scenario 1: When IMU Is in Fault

Figure 6.9 shows the quaternion of the FTFKF when the first sub-filter fails. The fault begins at the 500^{th} step and continues until the end of the experiment. The quaternions are safe from the first step to the 500^{th} step. From the 500^{th} step to the 520^{th} step is the fault confirmation process, which means the failure is only judged to have occurred if γ exceeds the threshold $gamma_0$ for 20 continuous steps. This process is for decreasing false alarms. If a failure is detected, the model switches to using the second sub-filter, and the quaternion from the master filter q_f is then the same as the quaternion from the second sub-filter q_s from the 520^{th} step to the last step.

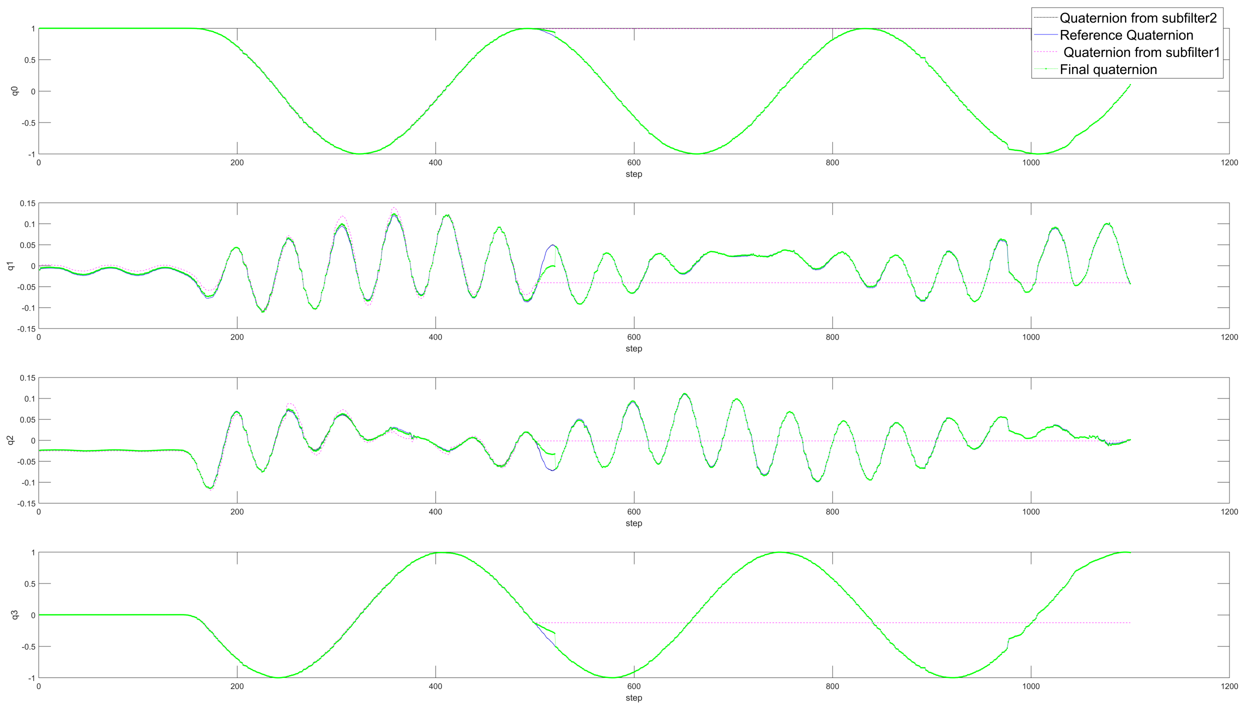


FIGURE 6.9: The quaternion of the FTFKF in scenario 1

Table 6.3 presents the detailed quaternion error from different filters in the first scenario. Compared with the quaternion values in Table 6.1 for the normal scenario, the quaternion error is the same for the second sub-filter. This indicates that while the failure is restricted to the first sub-filter, the second sub-filter remains functional and operates safely. Furthermore, in this scenario, the quaternion error of the master filter is greater than that of the second sub-filter but less than that of the first sub-filter.

TABLE 6.3: The quaternion error from different filters in scenario 1

Quaternion	q_0	q_1	q_2	q_3
Subfilter 1(IMU)	0.6011	0.0319	0.0238	0.3511
Subfilter 2(Star tracker)	0.0012	0.0013	0.0012	0.0014
Master Filter	0.019	0.0021	0.0017	0.0036

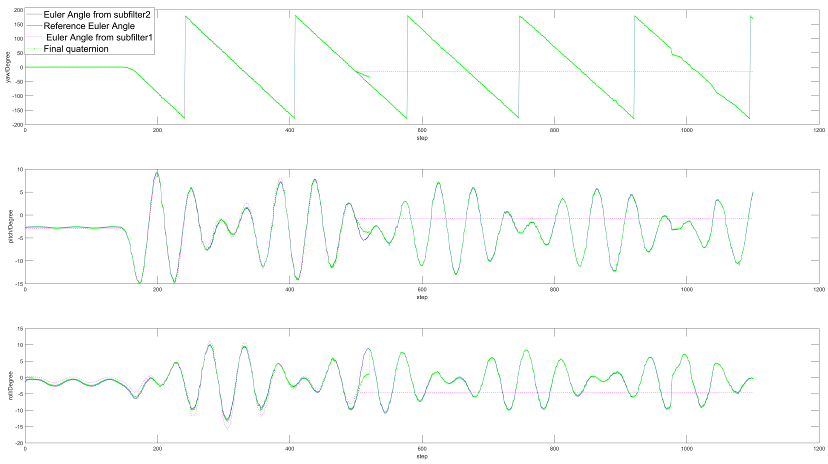


FIGURE 6.10: The Euler angle of the Fault tolerance federated Kalman filter in scenario 1

The figure 6.10 presents the Euler angles of the FTFKF when the first sub-filter is in failure. The highest accurate Euler angle is from the second sub-filter filer, followed by the Euler angle from the master filter. The Euler angles from the first sub-filter have the lowest accuracy because of the failure. This result is also proven in the figure 6.11, which shows the Euler angle absolute error of the FTFKF when the first sub-filter is in failure. In fugure 6.11, the smallest value of Euler angle error is from the second sub-filter, and the second is the master filter. The largest is the first sub-filter. The details are shown in Table 6.4.

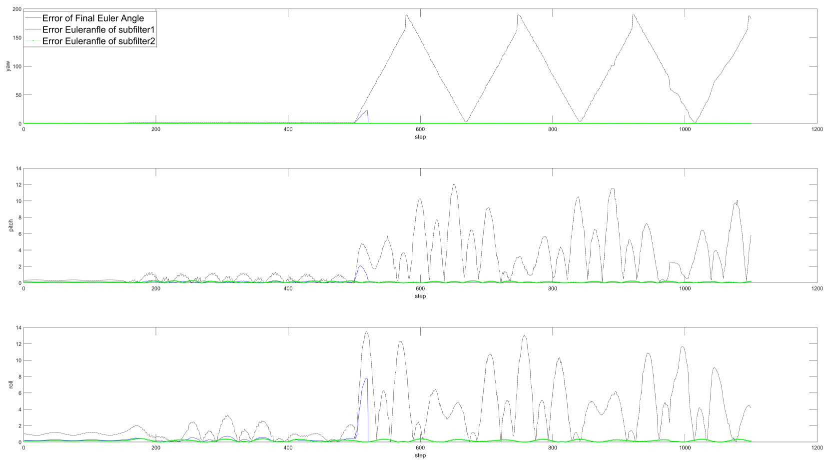


FIGURE 6.11: The Euler angle absolute error of the Fault tolerance federated Kalman filter when the first sub-filter in failure

In Table 6.4, it is evident that the largest Euler angle error is observed in the first sub-filter, which can be attributed to the presence of a fault in this particular sub-filter. The first sub-filter's yaw, pitch, and roll errors are quantified as $50.1267°$, $2.5238°$, and $3.2581°$, respectively. In contrast, the second sub-filter exhibits relatively lower errors, with yaw, pitch, and roll errors of $0.2422°$, $0.1052°$, and $0.1679°$, respectively. As for the master filter, the yaw, pitch, and roll errors are measured to be $0.5129°$, $0.1221°$, and $0.2854°$, respectively. Notably, the error of the master filter is not the smallest among the sub-filters since it is temporarily disabled during the fault confirmation process.

TABLE 6.4: The error of Euler angles from different filters in scenario 1

Euler angles/deg	Yaw	Pitch	Roll
Sub-filter 1(IMU)	50.1267	2.5238	3.2851
Sub-filter 2(Star tracker)	0.2422	0.1052	0.1679
Master Filter	0.5129	0.1221	0.2854

6.4.2 Scenario 2: When Star Tracker Is in Fault

For scenario 2, when the failure is in the second sub-filter, the quaternion of different filters is shown in figure 6.12. It can be found the fault is the second sub-filter, after the 500^{th}

step. From the first step to the 500^{th} step, the quaternion from different filters is the same as the quaternion from the normal situation. From the 500^{th} step to the 520^{th} step is the process of fault confirmation. During this process, the quaternion from the master q_f does not have high accuracy. After the 520^{th} step, the failure is determined in the second sub-filter, and FTFKF will switch to without the second sub-filter. Therefore, after the 520^{th} step, the quaternion from master q_f is the same as the quaternion from the first sub-filter q_q. The detailed quaternion error from different filters in scenario 2 is presented in Table 6.5

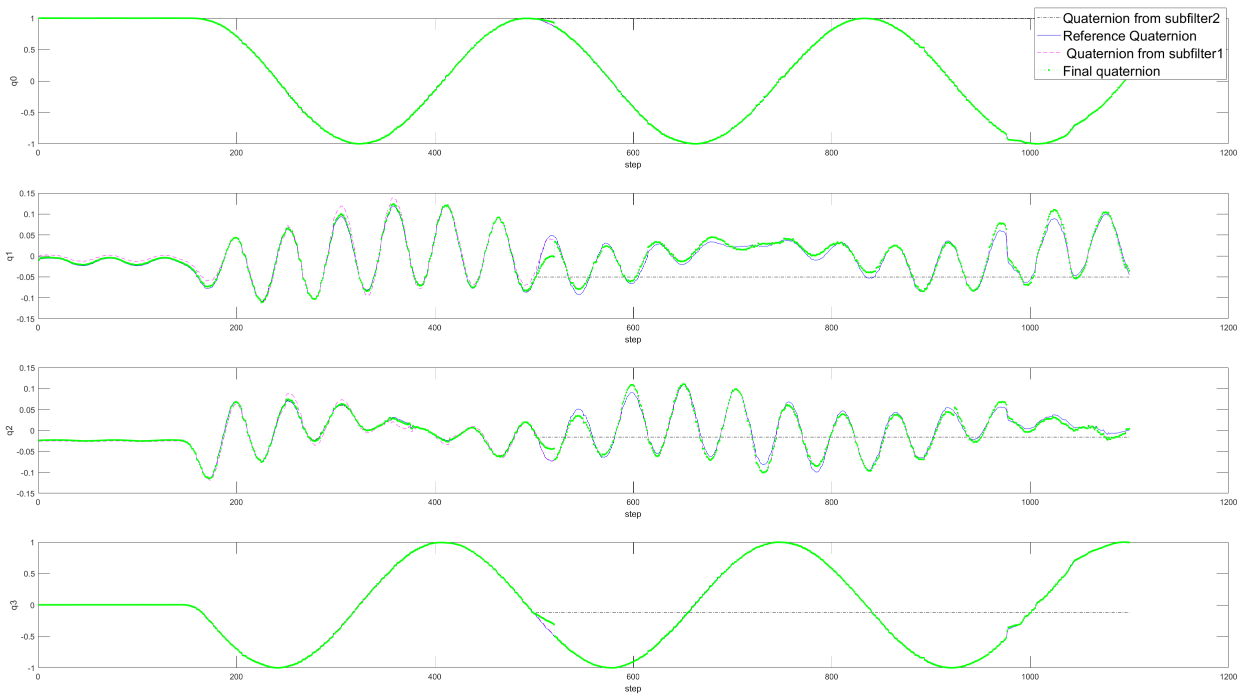


FIGURE 6.12: The quaternion of the Fault tolerance federated Kalman filter in scenario 2

TABLE 6.5: The quaternion error from different filters in scenario 2

Quaternion	q_0	q_1	q_2	q_3
Sub-filter 1(IMU)	0.0065	0.0072	0.0059	0.0078
Sub-filter 2(Star tracker)	0.5984	0.0313	0.0240	0.3475
Master Filter	0.0044	0.4048	0.044	0.0061

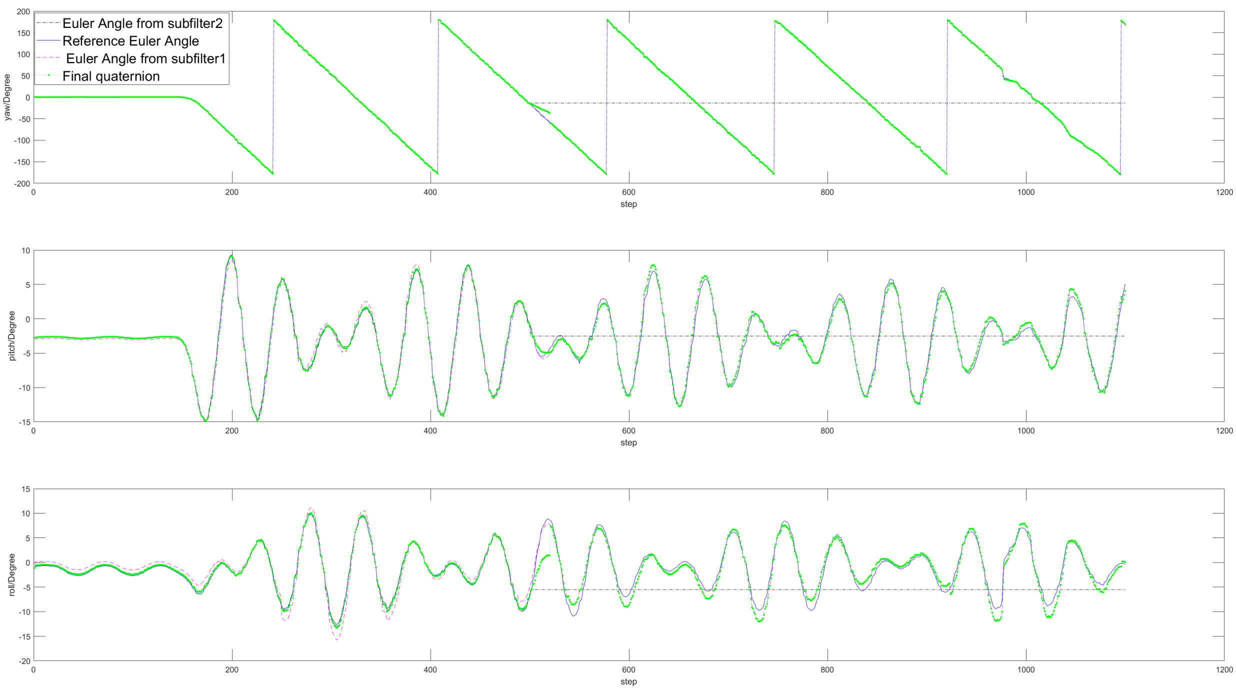


FIGURE 6.13: The Euler angle of the Fault tolerance federated Kalman filter when star tracker in failure

The Euler angle of the FTFKF in scenario 2 is presented in Figure 6.13, which illustrates the varying accuracies of the Euler angles obtained from different sub-filters and the master filter. As evidenced in Figure 6.14, which portrays the Euler angle error of the FTFKF when the star tracker experiences a fault, the highest accuracy is achieved by the Euler angles estimated by the master filter, followed by those obtained from the first sub-filter, while the second sub-filter produces the least accurate results due to the fault. These findings underscore the superiority of the master filter in achieving the most precise Euler angle estimations. At the same time, the accuracy of the sub-filters is constrained by the extent of the fault. Specifically, the Euler angle error in the second sub-filter is the largest, followed by the error in the first sub-filter, while the master filter exhibits the smallest error in the Euler angles. The figure 6.13 also shows, from the 500^{th} step to the 500^{th} step is the fault confirmation process, during this time, the error of the master filter is large, but it is a temporary. In addition, detailed error of Euler angles from different filters is shown in the table 6.6. It also can prove the highest accuracy is the master filter.

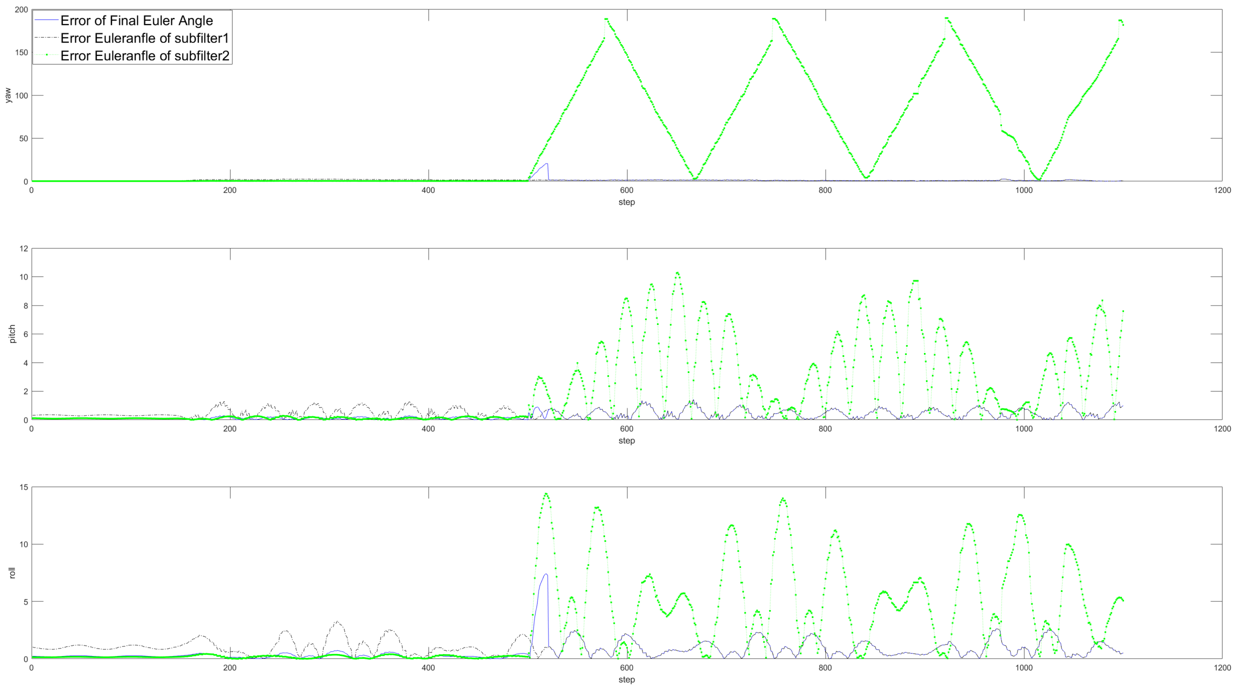


FIGURE 6.14: The Euler angle error of the Fault tolerance federated Kalman filter when star tracker in failure

TABLE 6.6: The error of Euler angles from different filters in scenario 2

Euler angles/deg	Yaw	Pitch	Roll
Subfilter 1(IMU)	1.2796	0.4554	0.9894
Subfilter 2(Star tracker)	49.571	2.0736	3.06
Master Filter	0.9429	0.2940	0.6925

6.5 Summery

This chapter presented a novel method to detect, isolate and recover faults of satellite ADCS by fusing the outputs of INS, and CNS based on FTFKF to achieve the fault-tolerant and high-accurate attitude information. The FTFKF employs INS and CNS as the sub-filter and the FKF is utilized to fuse the output from different sub-filters. The proposed algorithm offers a higher accuracy when compared to single INS and CNS. Furthermore, the fault confirmation process can provide lowers false alarm rates, resulting in a more reliable satellite attitude estimation solution. Finally, the highlights of this part:

(1) This paper develops a novel algorithm, which is based on FTFKF, to realize satellite attitude fault detection and recovery.

(2) The FTFKF-based algorithm offers reliable attitude estimation and decreases the false alarm rate.

(3) The parallel subsystem of fault detection, optimized for implementation on FPGA, enables real-time fault detection and fast recovery, demonstrating its potential in practical applications.

CHAPTER 7

Optimized FPGA Implementation of Fault Detection, Isolation and Recovery System for Satellite Attitude Estimation

---

This chapter introduces an optimized scheme for FPGA implementation of fault detection, isolation and recovery for satellite attitude estimation. The work was submitted by (Chen et al. 2023c). First, an overall design concept and framework are given, followed by a detailed discussion of the optimization process. Finally, the results of implementation on FPGA and the comparison with other devices, such as GPU and Raspberry Pi are presented.

7.1 Introduction

The attitude Determination and Control System (ADCS) of satellites is essential to have highly accurate attitude estimation. The Inertial Measurement Unit (IMU) is an important sensor for achieving accurate attitude estimation in ADCS. However, IMU sensors are vulnerable to degradation in space and may cause mission failure. To address this issue, a novel fault detection, isolation, and recovery system (FDIR) is necessary and presented in chapter 5. Traditional FDIR implementation on Field-Programmable Gate Array (FPGA) has a high computational load, taking up valuable hardware resources and slowing down processing speed. Despite this, it still delivers high attitude accuracy even if one sensor component of the IMU fails. In this chapter, the novel FDIR hardware design is divided into three stages: an Adaptive Kalman Filter (AUKF), two RBF neural networks, and an adaptive Complementary Filter (ACF). These stages are optimized through parallelism and pipeline processing on the FPGA, reducing latency while maintaining normal resource consumption and high attitude accuracy. Compared to the GPU (Jetxon TX2), the FDIR implementation on the PYNQ-Z2 (zynq 7000 clg400) offers faster processing speed while preserving excellent accuracy.

The following steps to the workflow are included in this Chapter: First, the RTL is generated in the HLS using C code. Secondly, the RTL is imported into Vivado, where a block design is generated and the HDL wrapper is completed this then provides a bitstream for exportation. Finally, the bitstream is imported to FPGA (PYNQ-Z2) board and the program uses Python to run the FDIR system.

7.2 Design Overview

7.2.1 Project Workflow

The FPGA project consists of three parts: programming with High-level Synthesis (HLs) using C++, designing the block in Vivado to generate the bitstream, and finally deploying the project on the PYNQ-Z2 board. The Whole project design flow is shown in figure 7.1

(1) The First part: The first step is writing the C++ code according to the requirement in the HLS, the next step in HLS is simulation and synthesis and the last step is exporting RTL.

(2) The second part: The Vivado toolchain comprises a two-step process to obtain the bitstream for a given design. First, the Register Transfer Level (RTL) and Block Diagram must be imported into the toolchain. Subsequently, a Hardware Description Language (HDL) wrapper is created to encapsulate the design hierarchy. Finally, the bitstream generation process is triggered and exported to complete the flow.

(3) The third part: It is in the PYNQ-Z2 board. Import the bitstream to the PYNQ-Z2 board by the Overlay and implement the ACF by Python.

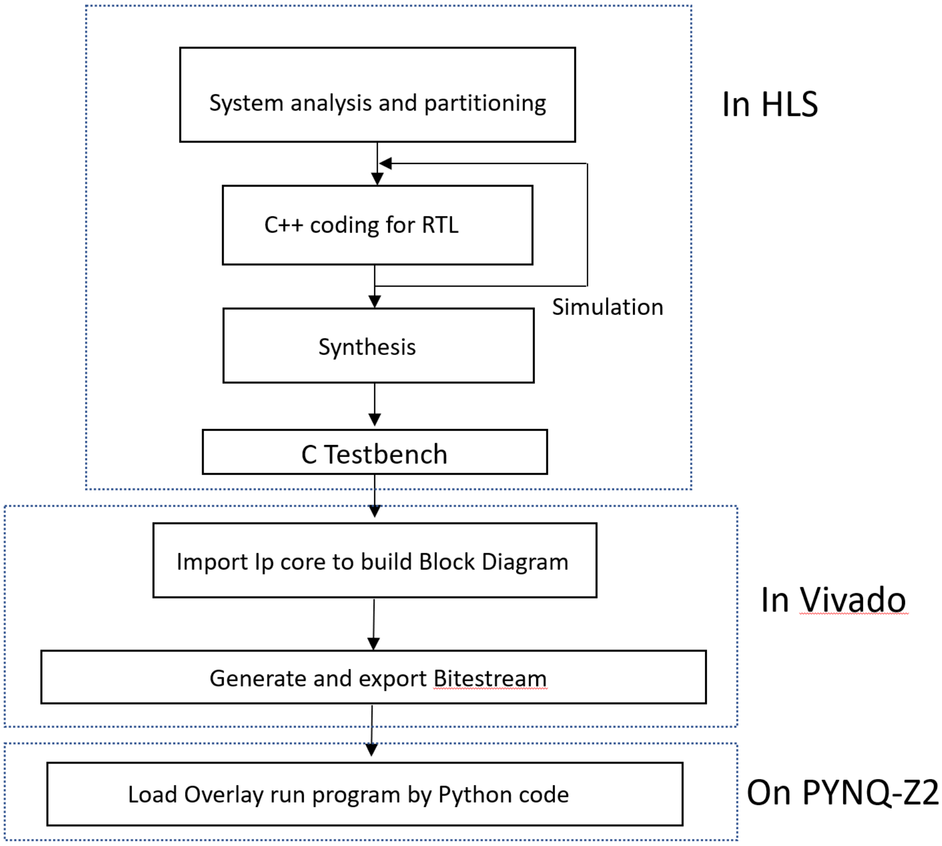


FIGURE 7.1: The design flow in HLS and Vivado

7.2.2 The Whole Project Process Flow

The FDIR system includes an AUKF, a QUEST algorithm, 2 RBF neural networks, and an ACF. In the hardware, the AUKF and the QUEST algorithm are in the first IP core, and two RBF neural networks belong to the second IP core. The ACF, due to its low computational demand, is implemented on the PS using Python. The hardware setup is depicted in figure 7.2. The IP cores communicate with the processor as slaves over a communication bus and are controlled by the software component. Memory buffers are in place between the modules of the AUKF and the processor for data and control information transfer. The IP cores start to work once valid data is present in the correct memory buffer, and the processor must store the necessary data beforehand. The two IP cores are designed for the parallel processing of new data, rather than serial processing. The working sequence on the FPGA board is as follows:

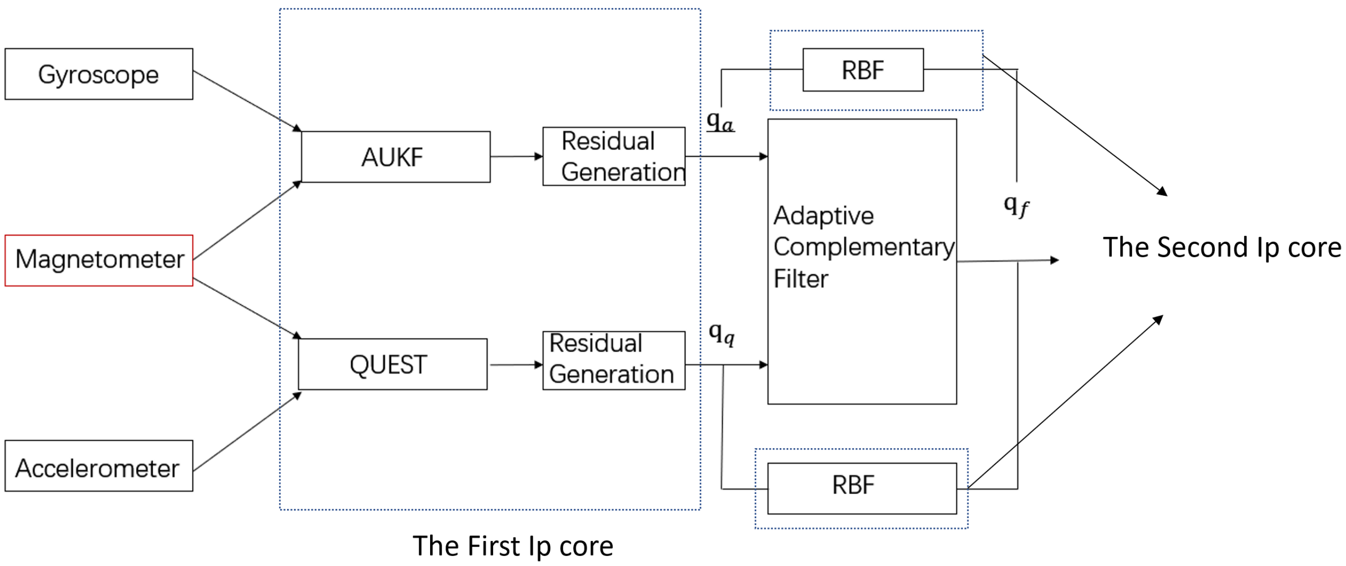


FIGURE 7.2: The details of IP core

the first IP core runs in a simultaneous pipeline with the second IP core, and finally, the results are fused by the ACF.

7.2.3 Traditional Hardware Design

The traditional hardware design follows a serial processing approach, where each algorithm is executed one after the other, as depicted in figure 7.3. The first step is the execution of the QUEST algorithm, followed by AUKF. Two residual generators are then used to determine if a failure has occurred based on the residuals. The subsequent steps involve running RBF1 and RBF2. The final stage involves implementing the ACF on the FPGA board.

Serial processing operates one operation per clock cycle and performs the operations in sequential order. This design strategy requires minimal hardware, reducing the area and power consumption. However, it has a slower performance and higher latency, leading to longer completion times for the program.

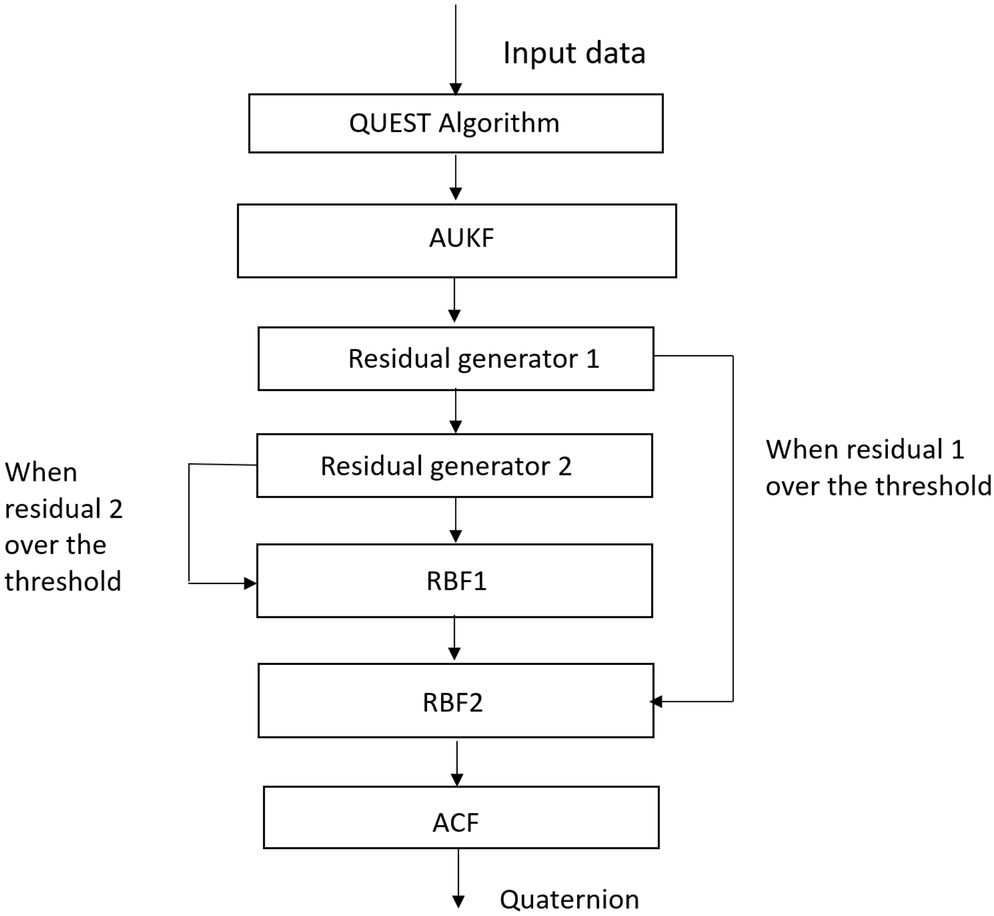


FIGURE 7.3: Traditional Design of FDIR

7.2.3.1 Traditional Design of the First IP Core

It is possible to set up the first IP core in this traditional hardware design; there are two main algorithms QUEST and the AUKF. The IMU data is input to the QUEST and AUKF, and these algorithms are processed serially as shown in figure 7.4. First, each step of the QUEST algorithm is calculated and then each step of the AUKF is computed through serial processing. The QUEST has 5 and AUKF has 6 important stages in traditional serial processing, this means 11 stages need to be complete in total, meaning 11 clock cycles must pass. This leads to high latency and long processing time. However, the stages in both algorithms can be

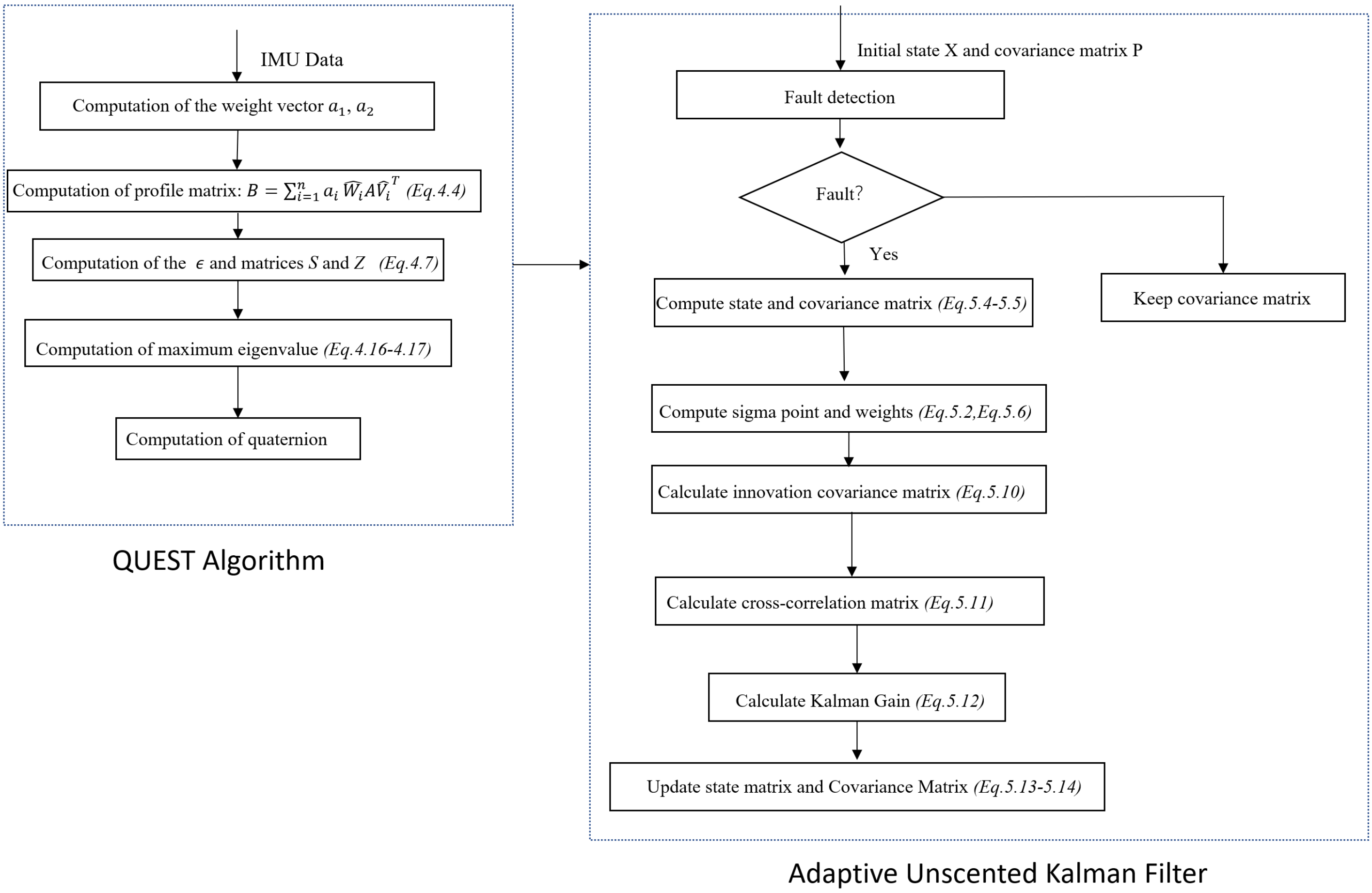


FIGURE 7.4: Flowchart of the Whole Project

optimized by pipeline processing and improved computational efficiency can be achieved by reshaping the input data through an array partition. The details are described in section 7.2.4.

7.2.3.2 Traditional Design of the Second IP Core

The basic structure of the RBF is shown in figure 7.5. The input data is the quaternion from the first IP, the activated function in the hidden layer is multivariate Gaussian function, which can be described as:

$$h_r = exp(-\frac{||X - c_r||^2}{2{\sigma_r}^2}), r = 1, 2.......n \tag{7.1}$$

where $c_r = [c_{r1}\ c_{r2}\ c_{r3}\ c_{r4}]$, n is the number of the neuron in the hidden layer,${\sigma_r}^2$ represents the node center and node variance of r^{th} neuron, and $||X - c_r||$ is the norm value which is measured by the inputs and the node center at each neuron. The output of the n-th neuron of

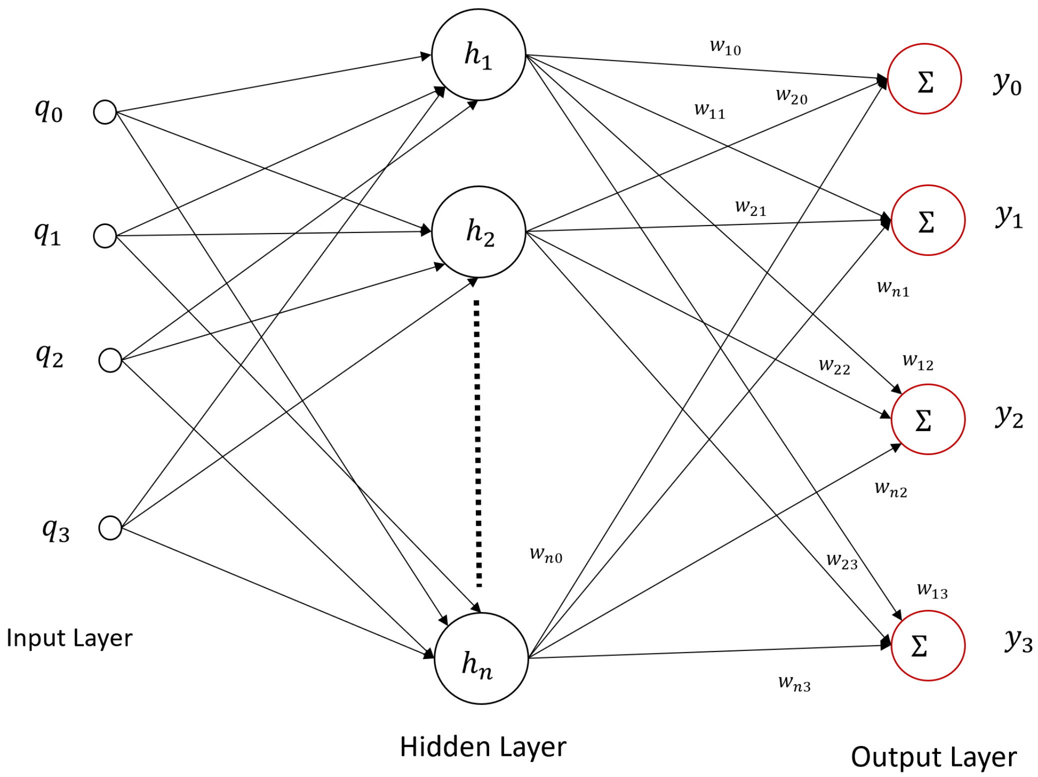


FIGURE 7.5: The basic architecture of RBF neural network

the output layer can be characterized as:

$$y_m = \sum_{h=0}^{4} w_{nm} h_r, m = 0, 1, 2, 3 \tag{7.2}$$

Substituting (7.1) in (7.2)gives:

$$y_m = \sum_{h=0}^{4} w_{nm} e^{(-\frac{||X - c_r||^2}{2{\sigma_r}^2})} \tag{7.3}$$

where w_{nm} is the synaptic weight between neuron n of the hidden layer and neuron m of the output layer.

7.2.4 Proposed Hardware Design

The proposed scheme of FDIR is depicted in figure 7.6. It depicts that the IMU data needs to be reshaped to be sent to the IP cores. Instead of sequential processing, the first and second IP cores are operated in pipelining. Additionally, the first IP core should run two algorithms(AUKF and QUEST) in parallel. The two RBF neural networks are also operated in parallel for the second IP core.

The optimization of the FPGA can be divided into 4 stages with the pipeline, which are presented in figure 7.7. these four stages are optimized through a pipeline system, meaning that stage 1 does not have to complete all its steps before stage 2 begins. This optimization improves the efficiency of the process and reduces the latency.

Furthermore, there are a number of loops and matrices computations in AUKF, QUEST, and RBF. In these algorithms, the loops can be optimized by pipelining, and the matrix multiplication can be optimized in parallel.

For the different stages, the algorithms in it can be optimized in Parallel. For example, in stage 2,the AUKF and QUEST can be operated in parallel. At stage 3, the residual generator 1 and residual generator 2 can be run in parallel as well. Similarly, stage 4 allows for the operation of RBF 1 and RBF 2 are in parallel.

These algorithms and networks are stored in the register, allowing the PS to directly access and calculate them using ACF with Python. To improve pipelining, the loops and matrix computations in AUKF and QUEST can be optimized by Partition Array, a pipelining method. The last step is still implementing the ACF on the PYNQ board.

The optimization scheme is based on three different methods: implementing two IP cores parallel, reshaping of the input data, and optimization of Loop and matrix calculation by pipeline and parallel.

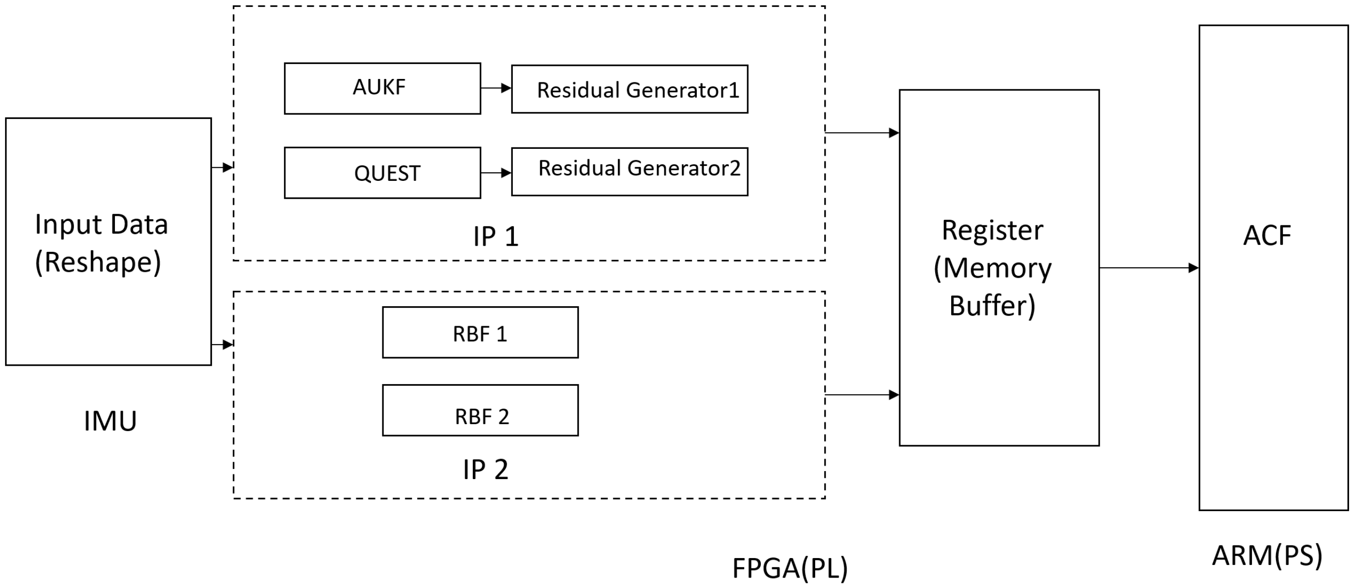


FIGURE 7.6: The proposed design of FDIR

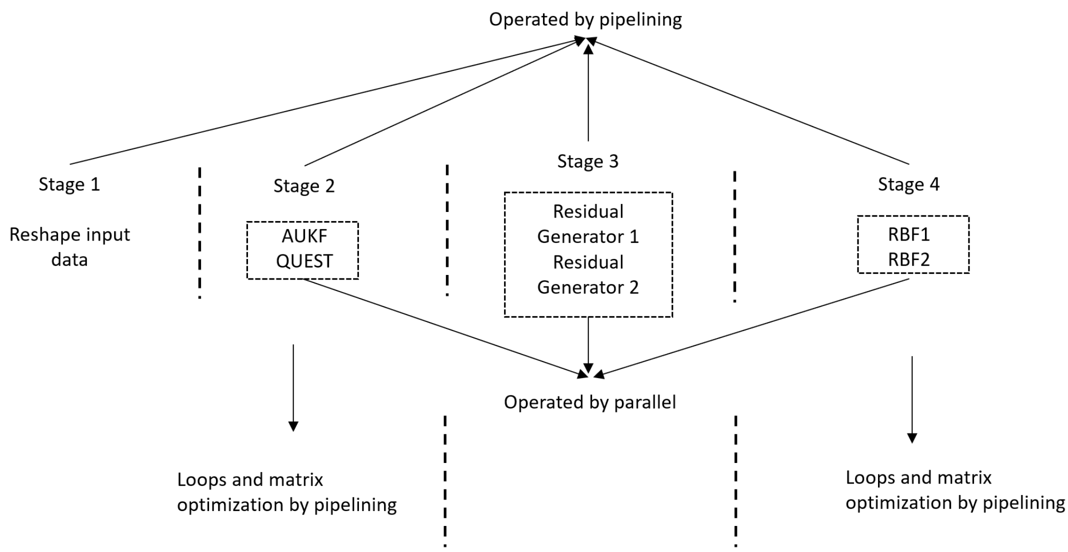


FIGURE 7.7: The stage of the proposed design

7.2.4.1 Reshape of the Input Data

The First method is reshaping the data size. The input data is from IMU, which includes 3-axis acceleration, 3-axis local magnetic field, and 3-axis angular rate. Therefore the size of one

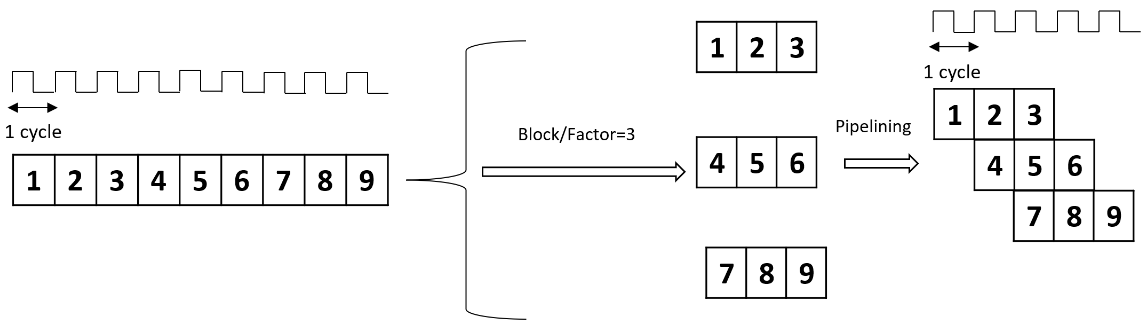


FIGURE 7.8: The Array partition and pipelining

group of the data is 3×3 , and there are total 9 elements. These elements will extract one by one because of serial processing. The optimization of the input data is array partition, which is shown in figure 7.8. In figure 7.8, the original size input data is 1×9, which is divided into 3 blocks, and each block has 3 elements. Then these three blocks can be extracted by pipelining. Traditional design requires 1 clock cycle to read a single data and 9 clock cycles to read a group of input data. The proposed design, however, only takes 5 clock cycles to extract a group of input data due to the array partition, which allows for more array elements to be read in a single clock cycle.

7.2.4.2 Implementing Two IP Cores Parallel

In different stages, many algorithms can be operated in parallel to improve computational efficiency. In stage 2, both AUKF and QUEST can be implemented in parallel. Similarly, in stage 3, the Residual generators can also be implemented in parallel. The two RBF neural networks in stage 4 are also designed for parallel processing. Compared with the traditional design of FDIR by serial processing in figure 7.3, the parallel processing in the proposed scheme is able to perform the IP core 1 and IP core 2 in one clock cycle because they are not sequentially related.

7.2.4.3 Optimization of Loop and Matrix Calculation by Pipeline

The last is the optimization of different Loops and matrices by pipelining and parallel. In AUKF, QUEST, and RBF, the matrix computations can be optimized by pipelining. Block

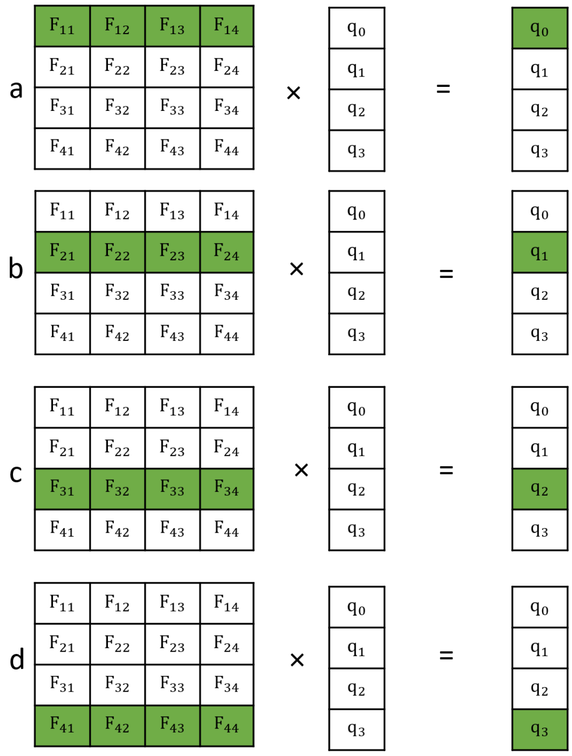


FIGURE 7.9: The optimization of matrix multiplication with parallel

matrices are sub-matrices partitioned out from the original matrix in the horizontal and vertical directions. The original matrix can be the set of block matrices. The multiplication of the original matrix can be divided into several multiplications of block matrices, and then these multiplication of block matrices can be optimized in parallel . These can increase the efficiency of computation speed with normal storage resources. For example, in the (5.1)in the AUKF, the size of the state matrix f is 4×4, and the input data is quaternion whose size is 4×1. The optimization process is shown in 7.9. The state matrix f is divided into 4 1×4 block matrices:a,b,c,d. These block matrices have matrix multiplication with quaternion, respectively at the same time. Then the 4 quaternion elements can be calculated at the time. Similarly, in QUEST and RBF, there are many this type of optimization that can reduce the latency.

Another optimization method in AUKF, QUESt, and RBF is the pipelining for the loops in the algorithms. The optimization of a loop is illustrated in Figure 7.10. Figure 7.10.A displays the loop without pipelining, while Figure 7.10. B shows the loop with pipelining. In the

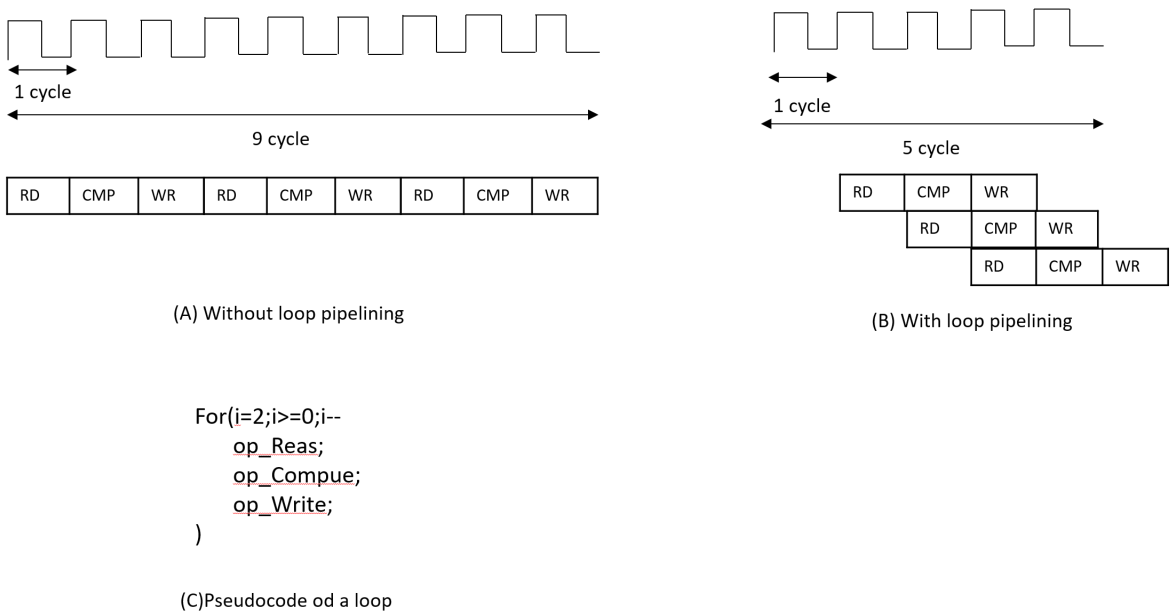


FIGURE 7.10: The optimization of matrix multiplication with parallel

figure, RD represents data reading, CMP signifies computing, and WR stands for data writing. Figure 7.10. C presents the pseudocode of the loop. It can be found in a loop. There are three main steps, reading data, computing, and writing data. If each step is one clock cycle, executing data needs three clock cycles. When executing three data it needs p clock cycles. However, with the use of loop pipelining, a 9-clock cycle loop can be reduced to 5 clock cycles.

Due to the strict processing time requirements of FDIR, it is crucial to determine any failures and respond immediately. By incorporating array partitioning, reshaping of data, and optimization in parallel can improve the efficiency of the work process.

7.3 Experiment

The simulation is completed in Cheaper 5. The FDIR system has been integrated on the FPGA (PYNQ-Z2) as illustrated in Figure 7.11. The PYNQ Z2 platform is equipped with a high-performance Zynq-7000 System on Chip (SoC) XC7Z020-1CLG400C, along with a 512 MB DDR3 memory controller boasting eight Direct Memory Access (DMA) channels and four high-speed AXI3 Slave ports. Additionally, it houses a Dual ARM Cortex-A9 MPCore, 256 KB On-Chip Memory, and 630 KB of fast block RAM, with an internal clock speed surpassing 450 MHz.

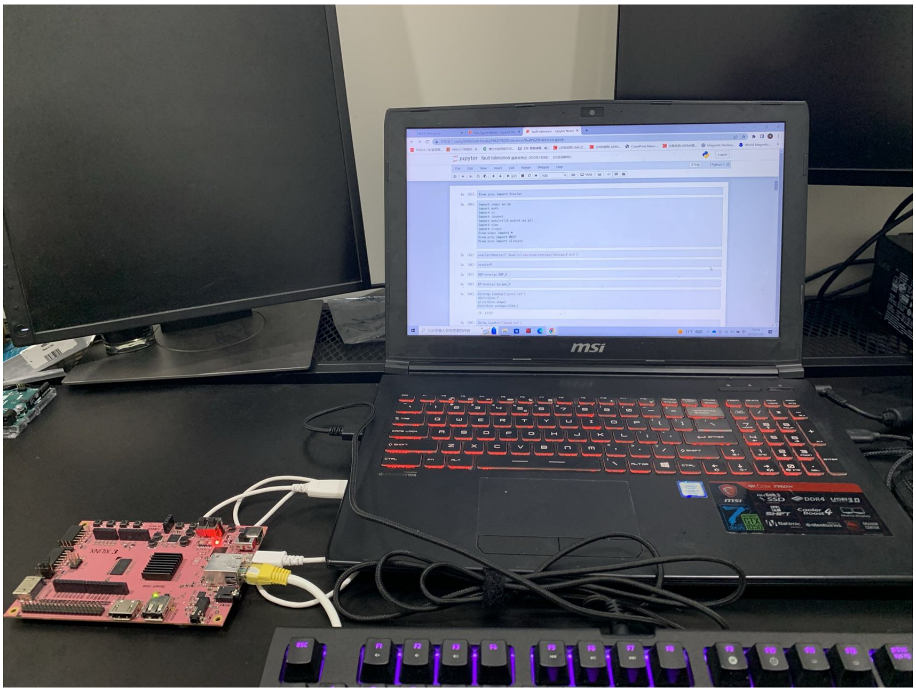
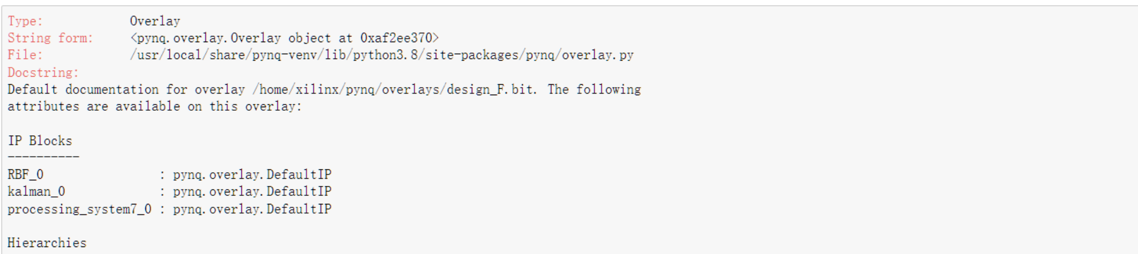


FIGURE 7.11: FDIR Implementation on PYNQ Z2

The FPGA of this project consists of two IP cores: AUKF and RBF, as depicted in figure 7.12. The illustration demonstrates that there are two IP blocks integrated into the FPGA, which can be managed through Python running in the PS. The PYNQ framework offers a Python-based interface for controlling the overlays in the PL from the PS.



```
Type:           Overlay
String form:    <pynq.overlay.Overlay object at 0xaf2ee370>
File:           /usr/local/share/pynq-venv/lib/python3.8/site-packages/pynq/overlay.py
Docstring:
Default documentation for overlay /home/xilinx/pynq/overlays/design_F.bit. The following
attributes are available on this overlay:

IP Blocks
----------
RBF_0                : pynq.overlay.DefaultIP
kalman_0             : pynq.overlay.DefaultIP
processing_system7_0 : pynq.overlay.DefaultIP

Hierarchies
-----------
```

FIGURE 7.12: Overlay of the FDIR

Figure 7.13 shows the quaternion from FDIR System in Matlab and Motion Tracking System, respectively. It can be found that the results in Python are the same as the simulation in

Matlab in Chapter 5. Figures 7.14, 7.15, and 7.16 depict the Euler angles for the gyroscope, magnetometer, and accelerometer, respectively, under fault conditions. Each sub-figure (a), (b), and (c) shows the yaw, pitch, and roll angle, respectively. The figures display three lines representing the Euler angles measured by the FDIR system in the FPGA, the reference Euler angles from the motion tracking system, and the Euler angles obtained when different sensors fail.

The results demonstrate that the FDIR system is capable of timely fault recovery, as evidenced by its ability to quickly correct for errors in the Euler angles measured by the faulty sensors. Specifically, the measurements of the FDIR system closely match those of the motion tracking system, indicating that the system is able to compensate effectively for sensor failures.

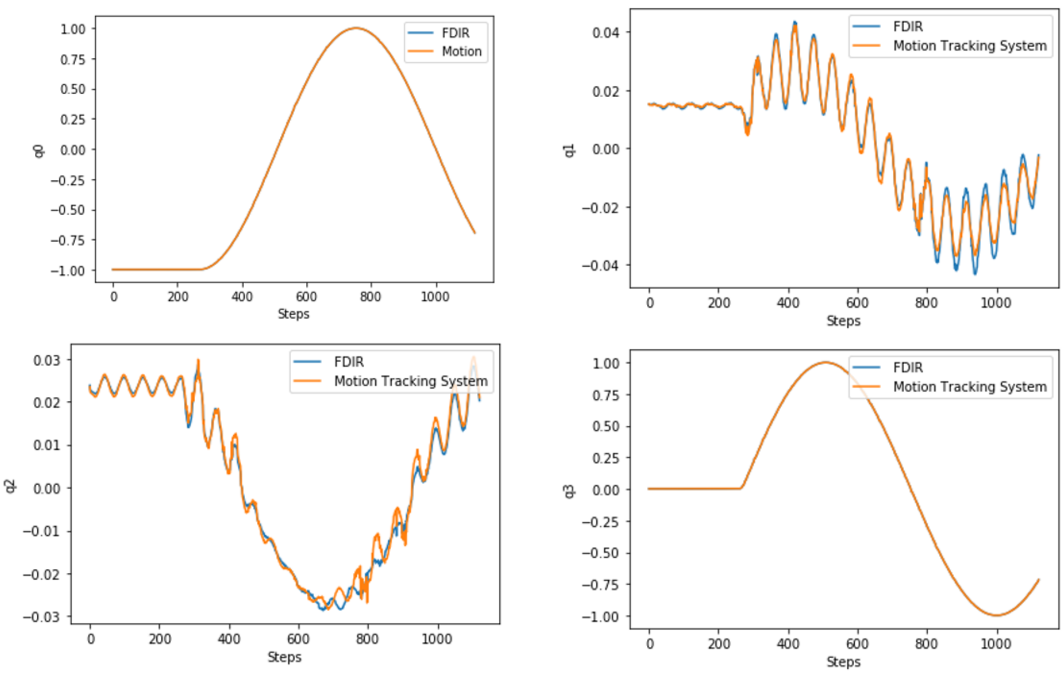
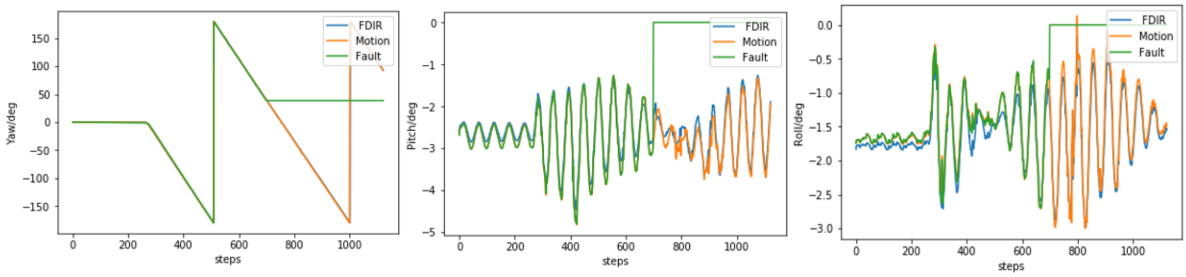


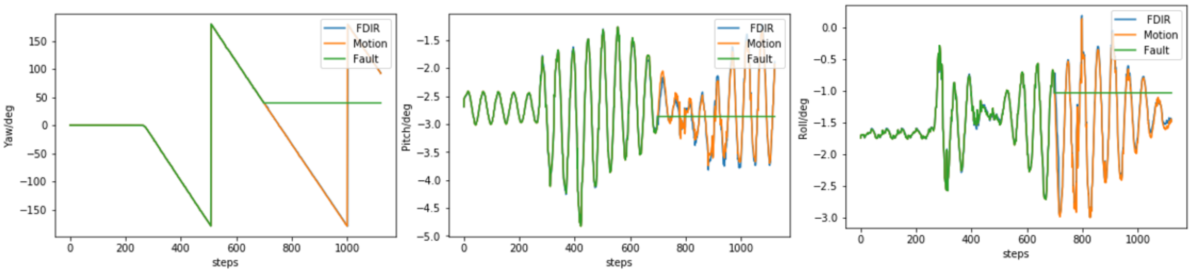
FIGURE 7.13: The Quaternion from FDIR System and Motion tracking System

The FDIR system has also been implemented on the GPU and the Raspberry Pi 3B+. For the GPU platform, the NVIDIA Jeston TX2 SoM has been utilized. This embedded system-on-module comprises a dual-core NVIDIA Denver2 and quad-core ARM Cortex-A57, 8GB 128-bit LPDDR4, and a 256-core Pascal GPU. It boasts a massive 58.3 GB Graphics Card Ram, a 1.2 GHz Tegra 4 processor, 32GB eMMC, and Dual ISPs. The memory speed of



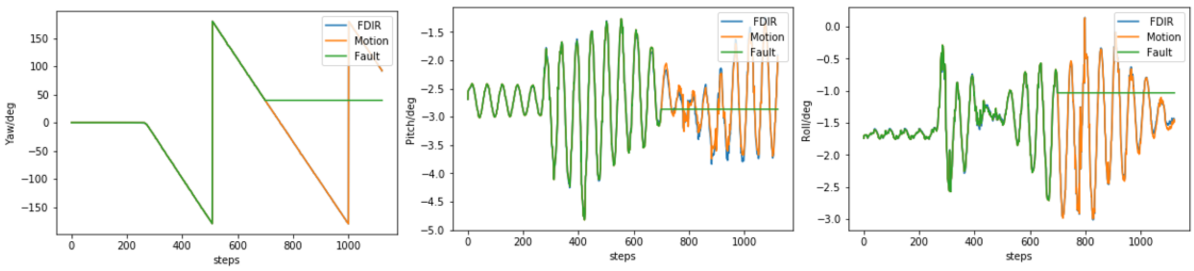
(a) (b) (c)

FIGURE 7.14: The Euler angle when gyroscope failure



(a) (b) (c)

FIGURE 7.15: The Euler angle when magnetometer failure



(a) (b) (c)

FIGURE 7.16: The Euler angle when accelerometer failure

the GPU is 60 MHz, while the hard drive rotational speed is 5400 RPM, and the memory bandwidth is 59.7GB/s. Furthermore, the power consumption of the GPU ranges from 7.5W to 15W.

On the other hand, the Raspberry Pi 3B+ features a 64-bit 1.4GHz quad-core ARM Cortex-A53 CPU, 1 GB LPDDR2 SDRAM, and Broadcom Videncore-IV GPU. The memory performance of the Raspberry Pi 3B+ is also noteworthy, with a write speed of 632.27MB/S and a read speed of 857.96MB/S.

Table 7.1 summarizes the FDIR system's running time performance on different hardware platforms, including FPGA, GPU, and Raspberry Pi 3B+. The FPGA implementation achieves the fastest running time, completing the entire project in 2.89578 seconds, followed by the GPU implementation, which takes 4.35827 seconds. The Raspberry Pi 3B+ implementation has the slowest running time performance, requiring 6.06531 seconds to complete the same task. Moreover, the power consumption of the processors varies significantly. The FPGA implementation is the most power-efficient, consuming only 1.7 watts, while the GPU implementation has the highest power consumption of 15 watts. This information is valuable for system designers to make informed decisions based on running time performance and power consumption requirements on hardware platform selection.

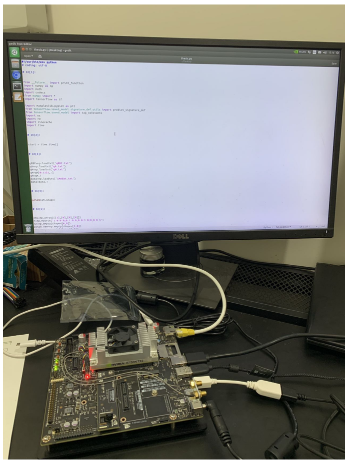


FIGURE 7.17: FDIR implementation on GPU

TABLE 7.1: The time and power consumption of different hardware

Processor	FPGA	GPU	Raspberry Pi 3B+
Time/second	2.89758	4.35827	6.06531
Power consumption/W	1.7	15	12.5

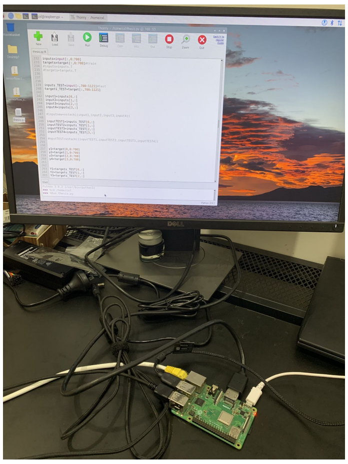


FIGURE 7.18: FDIR implementation on Raspberry Pi

7.3.1 Comparison Between Traditional Design and Proposed Optimization on FPGA

The conventional design adheres to a sequential processing approach. Optimization aims to strike a balance between computational speed and resource utilization. This section will compare the traditional FDIR System with the proposed FDIR system in terms of latency, resource consumption, and power consumption.

The initial step in the project is to create the RTL of two IP cores, AUKF and RBF, using HLS simulation. In HLS simulation, figure 7.19 displays the synthesis report of AUKF and RBF in the traditional design. It can be seen that the latency in Kalman and RBF are 2797 and 5828 clock cycles, which are relatively high. Figure 7.20 shows the synthesis report of AUKF and RBF in the proposed scheme. The latency in Kalman and RBF are 606 and 470 clock cycles. Both the AUKF and QUEST in the proposed scheme have lower latency than in the traditional design.

The resource consumption is shown in figure 7.21. The figure 7.21. (a) and (b) represent the AUKF and RBF without optimization, respectively. For the AUKF without optimization, the utilization of 4 different memories are: the utilization of BRAM is 13%, the utilization of DSP is 11%, the utilization of FF is 12%, and the utilization of LUT is 35%. For the RBF without optimization, the utilization of 4 different memories are: the utilization of BRAM is

FIGURE 7.19: The synthesis report in traditional design by HLS simulation

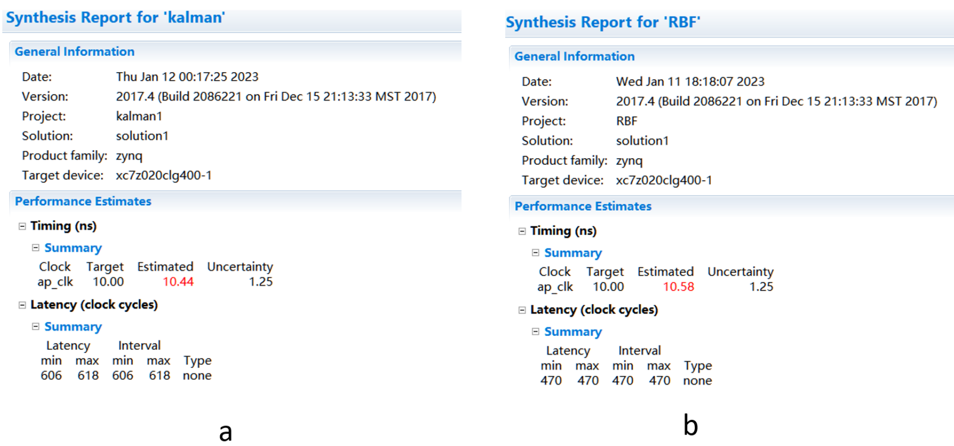


FIGURE 7.20: The synthesis report in the proposed scheme by HLS simulation

10%, the utilization of DSP is 11%, the utilization of FF is 12%, and the utilization of LUT is 30%.

In figure 7.21.(c) and (d) are the synthesis report of AUKF and RBF with the optimization scheme. For the AUKF with optimization, the utilization of BRAM is 9%, the utilization of DSP is 24%, the utilization of FF is 20%, and the utilization of LUT is 26%. For the RBF with optimization, the utilization of BRAM is 26%, the utilization of DSP is 50%, the utilization of FF is 22%, and the utilization of LUT is 47%. It can be found in AUKF, the utilization of BRAM in the proposed scheme is lower 1% of that in traditional design. However, other three

Utilization Estimates

Summary

Name	BRAM_18K	DSP48E	FF	LUT
DSP	-	-	-	-
Expression	-	-	0	3716
FIFO	-	-	-	-
Instance	16	25	8986	12232
Memory	22	-	387	33
Multiplexer	-	-	-	2657
Register	-	-	4314	-
Total	38	25	13687	18638
Available	280	220	106400	53200
Utilization (%)	13	11	12	35

(a) AUKF without optimization

Utilization Estimates

Summary

Name	BRAM_18K	DSP48E	FF	LUT
DSP	-	-	-	-
Expression	-	-	0	964
FIFO	-	-	-	-
Instance	28	25	10386	14014
Memory	2	-	0	0
Multiplexer	-	-	-	1401
Register	-	-	2441	-
Total	30	25	12827	16379
Available	280	220	106400	53200
Utilization (%)	10	11	12	30

(b) RBF without optimization

Utilization Estimates

Summary

Name	BRAM_18K	DSP48E	FF	LUT
DSP	-	-	-	-
Expression	-	-	0	7106
FIFO	-	-	-	-
Instance	4	53	9110	14222
Memory	18	-	389	33
Multiplexer	-	-	-	2463
Register	4	-	12685	675
Total	26	53	22184	24499
Available	280	220	106400	53200
Utilization (%)	9	24	20	46

(c) AUKF with optimization

Utilization Estimates

Summary

Name	BRAM_18K	DSP48E	FF	LUT
DSP	-	-	-	-
Expression	-	-	0	1192
FIFO	-	-	-	-
Instance	0	112	13977	21536
Memory	4	-	0	0
Multiplexer	-	-	-	2069
Register	70	-	10440	382
Total	74	112	24417	25179
Available	280	220	106400	53200
Utilization (%)	26	50	22	47

(d) RBF with optimization

FIGURE 7.21: The utilization report

memory registers, including DSP, FF, and LUT, the resource consumption is higher, especially for the DSP, the utilization is higher than 13%. Compared with the traditional design, the utilization of all memory registers has an increment in the RBF, especially for the DSP, the utilization increase from 11% to 50%. For the BRAM, FF, and LUT, the utilization increase by 16%,10%, and 17%.

The second step of the whole project is creating the Block design as the bitstream in Vivado. The Block design of the proposed scheme is shown in figure 7.22. It can be found there are two IP core: AUKF and RBF, a ZYNQ processor, and AXI SmartConnect. The AXI SmartConnect is a necessary part of matrix multiplication. The IP core AUKF and RBF are connected with AXI SmartConnect first, and then Axi Smartconnect connects with the ZYNQ processor.

In Vivado, the utilization of the whole project is shown in figure 7.23. Figure 7.23(a) is the resource consumption of FDIR in traditional design. Figure 7.23(b) is the resource consumption of FDIR in the proposed scheme. It can be found that the utilization of each

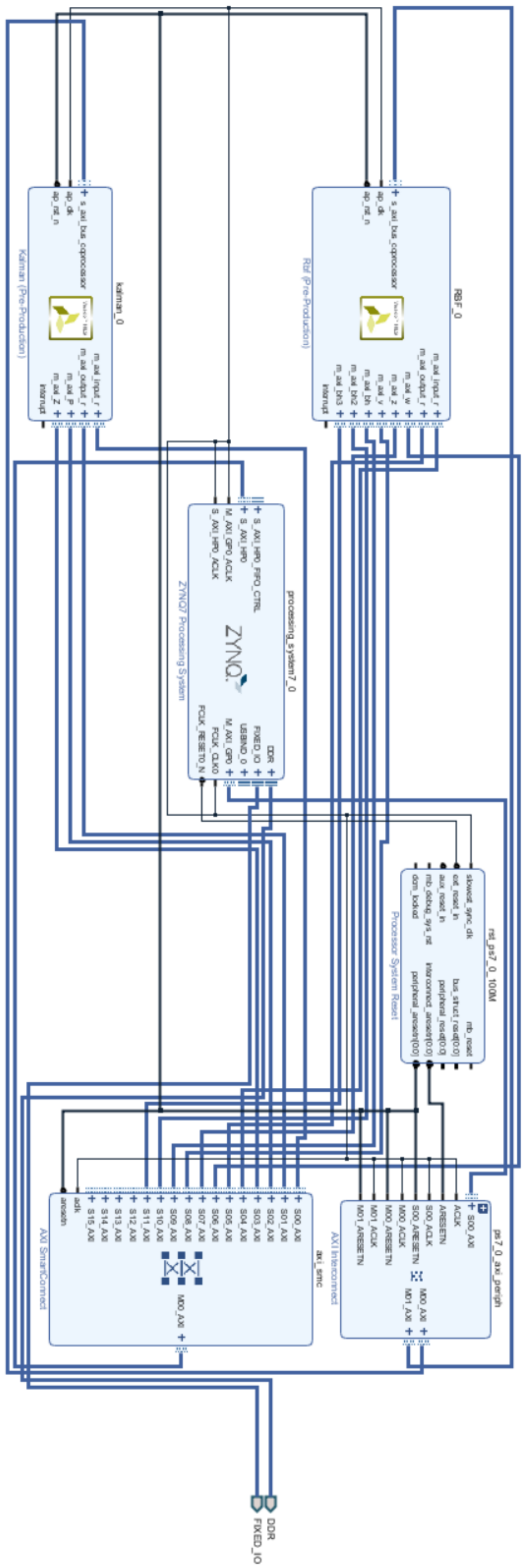


FIGURE 7.22: The Block Design

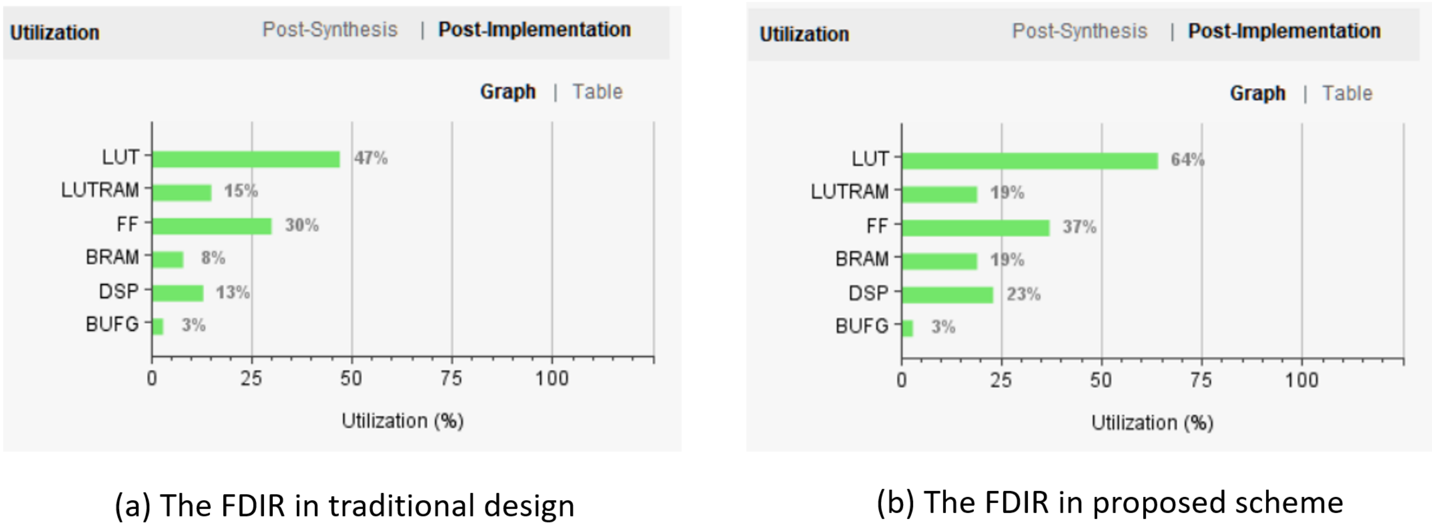


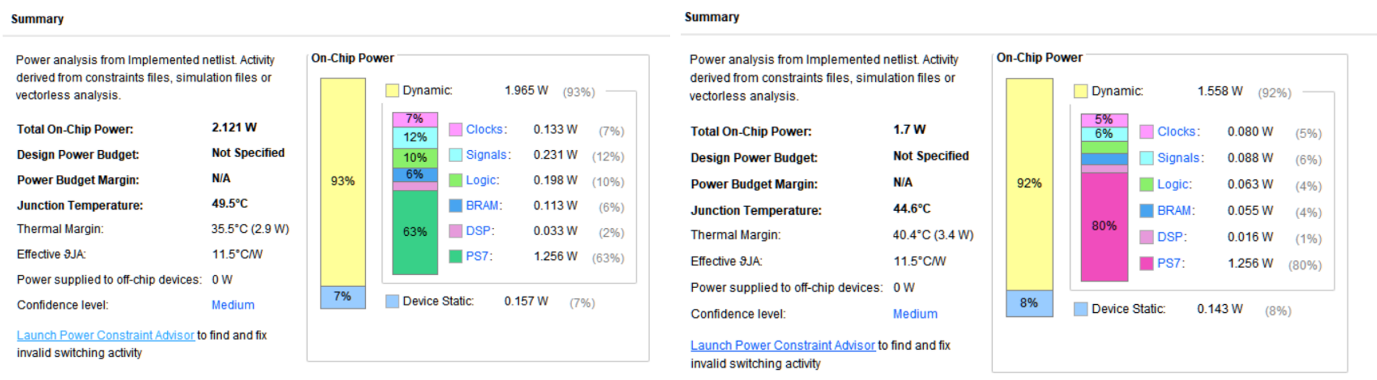
FIGURE 7.23: The Block Design of the whole project

memory registers in the proposed scheme is higher than that in traditional design. Compared with traditional design, the resource consumption of LUT FF, BRAM, and DSP increase by 17%, 7%, 9%, 10%. It should be noted that the total resource consumption $W_{utilization}$ does not simply equal the sum of the individual resource utilizations for AUKF and RBF ($AUKF_{utilization}$ and $RBF_{utilization}$, respectively):

$$W_{utilization} \neq AUKF_{utilization} + RBF_{utilization} \tag{7.4}$$

This is because both AUKF and RBF are optimized using parallel and pipeline techniques.

In Vivado, the power consumption is presented in figure 7.24. The traditional design consumes 2.12 watts, but the optimized design consumes only 1.7 watts, a decrease of 0.42 watts. The traditional design operates at $49.5°$C , while the proposed scheme operates at $44.6°$C. The optimized design outperforms the traditional design in terms of power consumption..



(a) Power consumption without optimization

(b) Power consumption with optimization

FIGURE 7.24: The power consumption

7.4 Summery

The present chapter presents the implementation of a novel FDIR system on various platforms, namely FPGA, GPU, and Raspberry Pi 3 B+. To enhance the efficiency and reduce the latency of the system on an FPGA board, the implementation employs advanced techniques such as pipelining and parallelization. The optimized implementation of the FDIR system on FPGA exhibits normal resource consumption and working temperature while significantly improving its efficiency. The FDIR system is subject to strict requirements for programming speed due to its critical role in detecting faults and enabling prompt recovery from them. The proposed approach utilizes the ZYNQ platform by integrating complex algorithms, including AUKF and RBF, into the FPGA part (PL) and fusing them with the ARM processor (PS) using the ACF. This approach maximizes the utilization of resources available on the ZYNQ platform. The implementation results of the proposed approach demonstrate high accuracy in attitude estimation, which is consistent with simulation results. Overall, the presented implementation significantly improves the efficiency and accuracy of the FDIR system on various platforms. This part has the several highlights:

(1) Novel implementation of the FDIR system on the FPGA board.

(2) Creating a novel hardware design, dividing the FDIR system into three stages, and generating the two IP cores. The two IP cores are implemented on the PL of PYNQ,

and the ACF will be implemented on the PS of PYNQ. Therefore we can utilize the resources of the PYNQ board efficiently.

(3) By implementing pipelining and parallelism for optimization, the latency can be reduced without sacrificing resource utilization, ensuring a high level of accuracy at all times and keeping a low working temperature.

CHAPTER 8

Conclusion

8.1 Summary of Research

This thesis presents three novel algorithms for satellite attitude estimation, along with the implementation of one of these algorithms on an FPGA board with optimization. The first algorithm is an attitude estimation algorithm for INS, which includes QUEST, FQA, Kalman filter, and LERP algorithm. The second algorithm is focused on FDIR for the INS navigation system. Finally, the third algorithm is a fault-tolerant federated Kalman filter for the ADCS in the satellite, which combines the output of the INS and CNS and includes fault detection and recovery mechanisms. In addition, the second algorithm has been implemented on the FPGA board to reduce latency and resource consumption with optimization.

Chapter 1 introduces the research motivation for developing a satellite attitude estimation algorithm in the INS and FDIR system for the ADCS. Additionally, the chapter presents the FTFKF method to integrate the INS and Celestial Navigation System (CNS) for better accuracy in attitude estimation.

Chapter 2 provides a comprehensive review of the related fields. It covers the satellite attitude determination system, satellite fault tolerance problems, and FPGA implementation for attitude estimation.

Chapter 3 introduces the background and basic knowledge of satellite attitude estimation. It explains the coordinate reference systems, different attitude representations, and sensor models used in satellite attitude estimation. It also presents the kinematic equation and low earth orbit dynamic equation.

Chapter 4 presents a novel two-layer Kalman filter-based algorithm for satellite attitude estimation using an INS system. The proposed algorithm integrates the outputs of the QUEST and FQA to achieve higher accuracy in attitude estimation. The Kalman filter uses a quaternion matrix derived from the LERP as the observation and process models. The process model integrates the output from the gyroscope to provide a smooth output while avoiding singularities and reducing computational complexity. The two-layer architecture is robust against magnetic disturbances and other adverse conditions. Furthermore, the second linear interpolation ensures high-precision attitude estimation for vehicles in both static and dynamic environments.

Chapter 5 presents a novel approach for the FDIR system within an Inertial Measurement Unit (IMU) as a part of an onboard satellite ADCS. The proposed approach is groundbreaking, involving the fusion of outputs from multiple algorithms, including AUKF, QUEST, and two RBF neural networks based on an ACF and hypothesis testing. This multi-layered approach ensures the robustness and high accuracy of the satellite's attitude estimation, even in the presence of faulty sensors. During the preliminary phase of the recovery process, the AUKF algorithm is utilized for both fault detection and attitude recovery. Following this, a secondary recovery phase employs trained neural networks to estimate the attitude, providing a more comprehensive solution. The multi-level recovery strategy guarantees that the satellite ADCS system can maintain a reasonable level of accuracy, ensuring that the satellite operates correctly even in the presence of a faulty sensor. Furthermore, the proposed algorithm offers a lower false alarm rate, which results in a more reliable satellite attitude estimation solution. This novel approach represents a significant contribution to satellite attitude determination and control, providing a robust, accurate, and comprehensive solution to the challenge of fault tolerance in INS. In summary, the proposed FDIR system based on a multi-layered approach involving the fusion of multiple algorithms offers an innovative solution to the challenging problem of fault tolerance in INS. The system ensures robustness, high accuracy, and reliability, making it an essential contribution to satellite attitude determination and control.

Chapter 6 presents a pioneering algorithm, the FTFKF, designed to fuse information from the INS and CNS in satellite Attitude Determination and Control Systems (ADCS). This algorithm enables fault detection, isolation, and recovery in complex space environments. The FTFKF is composed of a master filter and two sub-filters. In safe scenarios, the FTFKF provides more accurate attitude information than single INS and CNS solutions. However, in the event of a failure, the FTFKF can identify the sub-filter that has malfunctioned, isolate it, and recover the normal satellite attitude. Additionally, the algorithm incorporates a fault detection factor threshold that, when exceeded for a constant number of steps, identifies the sub-filter as the cause of a false alarm. The results of this study demonstrate the effectiveness of the FTFKF in improving the accuracy and reliability of satellite ADCS in complex space environments.

Chapter 7 presents a detailed account of implementing and optimizing the Fault Detection, Isolation, and Recovery (FDIR) system on an FPGA. The system is optimized on an FPGA board using pipelining and parallel techniques to achieve superior performance. This optimization significantly enhances the system's efficiency, minimizes latency, and maintains normal resource consumption and temperature. The FDIR system demands high programming speed for prompt fault detection and quick recovery. To this end, the proposed implementation leverages the ZYNQ platform by integrating complex algorithms such as AUKF and RBF into the FPGA part (PL). These algorithms are then integrated with the ARM processor (PS) using ACF to maximize the utilization of resources available on the ZYNQ platform. Integrating the advanced algorithms into the FPGA provides a powerful computing platform for the FDIR system, enabling it to detect and isolate faults rapidly and accurately. By utilizing the ZYNQ platform's available resources effectively, the proposed implementation ensures that the system operates at optimum performance levels while minimizing power consumption and heat generation. In conclusion, implementing and optimizing the FDIR system on an FPGA, as described in Chapter 7, presents a significant advancement in fault detection and recovery technology. This optimized system leverages advanced algorithms, pipelining, and parallel processing techniques to provide superior performance, making it a valuable contribution to the field of fault-tolerant systems.

8.2 Main Contribution

The main contributions of this thesis are three proposed algorithms and one algorithm implementation on FPGA. Two of them are designed for INS, and the other is the fusion of the INS and CNS in the satellite ADCS. Specific content and key details are listed below.

- The first algorithm is a novel two-layer Kalman filter architecture for attitude estimation, which outperforms existing algorithms by fusing QUEST and FQA to generate a more accurate quaternion observation model. The Kalman filter utilizes a quaternion matrix from the LERP as the observation and process models. The latter integrates the gyroscope output to achieve smooth results and avoid singularities, reducing computational complexity. The two-layer approach is robust against magnetic disturbances and other adverse conditions. The second LERP also ensures high accuracy in static and dynamic motion conditions. Simulations and physical experiments on an air-bearing table validate the algorithm; In the simulation, the average error of the proposed algorithm is 0.2669 deg. The average error of QKF is 0.7912 deg, and the average error of FQAKF is 1.5818 deg, the largest error of the three algorithms. In the experiment, the average error of the yaw angle from the proposed algorithm is 97% lower than that of the FQAKF and around 23% lower than that of the QKF. In addition, compared with the FQAKF, the proposed algorithm improves 16.9% computational efficiency.
- This thesis presents a novel algorithm for detecting, isolating, and recovering faults in INS, which is a critical component of an onboard satellite ADCS system. The algorithm combines the outputs of AUKF, QUEST, and two RBF neural networks with an adaptive complementary filter and hypothesis test to achieve fault-tolerant and high-accuracy attitude estimations. The preliminary recovery phase employs AUKF for fault detection and attitude recovery. The secondary recovery phase uses trained neural networks to estimate attitude. The multi-level recovery strategy ensures the satellite ADCS system maintains reasonable accuracy even with a sensor fault. Compared to other methods such as AUKF and QUEST algorithms, the proposed algorithm demonstrates higher accuracy for each sensor failure when

tested on an experimental platform. The absolute error of Yaw Pitch and roll when the gyroscope is in the fault are 0.0459562 deg/s, 0.2269 deg/s,and 0.15344 deg/s, respectively. The secondary fault detection and isolation layer also yields lower false alarm rates, making the satellite attitude estimation solution more reliable.

Overall, the algorithm offers a novel approach to fault detection, isolation, and recovery in IMUs that could potentially improve the accuracy and reliability of satellite ADCS systems.

- The third novel algorithm combines the INS and CNS of satellite ADCS. This algorithm addresses the unique challenges posed by satellite navigation systems, including the need for high accuracy and robustness to external disturbances. By fusing data from both the IMU and star tracker, this algorithm is able to provide highly accurate and robust estimation of the attitude, and attitude of the satellite, making it important for satellite ADCS.
- the implementation of the FDIR system on various platforms, including FPGA, GPU, and Raspberry Pi 3 B+, are presented. Optimizing the system on an FPGA board through pipelining and parallel techniques significantly improves its efficiency and reduces latency while keeping resource consumption and working temperature normal. This is a novel implementation of this FDIR system on an FPGA board. Fault Detection, The FDIR system is subject to stringent requirements for programming speed. This is because a faster programming speed enables more prompt detection of faults and quicker recovery from them. Furthermore, the proposed approach leverages the ZYNQ platform by integrating complex algorithms, such as AUKF and RBF, into the FPGA part (PL) and fusing them with the ARN processor (PS) using the ACF, thus maximizing the utilization of the resources available on the ZYNQ platform. The results of attitude estimation from the implementation are consistent with simulation results, demonstrating high accuracy.

8.3 Future Outlook

The current research presented in this thesis focuses on satellite attitude estimation, primarily based on the IMU and star tracker sensors. However, it is important to note that GPS, sun sensor, and other sensors are also commonly used in satellite attitude estimation. Therefore, future research can be advanced by investigating more algorithms based on other sensors to improve the accuracy and robustness of attitude estimation.

One approach for further research is to optimize and improve the algorithms used for attitude estimation, leveraging other common sensors such as GPS and sun sensors. Sophisticated sensor fusion algorithms like the extended Kalman Filter (EKF) or the Unscented Kalman Filter (UKF) can be employed to enhance the accuracy of attitude estimation. Moreover, more advanced fault detection algorithms such as the dynamic threshold algorithm, can be used to improve the system's fault detection and isolation capabilities.

Furthermore, the proposed algorithms can be extended to other applications like robotics and autonomous vehicle systems. The fault detection and recovery approach can be applied to other systems in the aerospace and automotive industries. The FPGA-based implementation of the FDIR system can also be optimized by utilizing more efficient data structures and design techniques.

Future research can also explore more advanced topics related to satellite attitude estimation and the FDIR system. For instance, a more comprehensive fault detection system can be developed to detect and diagnose a broader range of faults, such as permanent and transient faults. Additionally, machine learning and artificial intelligence can enhance the accuracy and robustness of attitude estimation and fault detection.

In conclusion, this thesis provides a comprehensive overview of the development of satellite attitude estimation and the FDIR system. Novel algorithms for attitude estimation and fault detection, along with a groundbreaking approach for the FDIR system have been developed and tested. This research forms the basis for future work on developing FDIR systems for other applications.

Bibliography

Agarwal, Vivek, Hemendra Arya and SHIVARAM Bhaktavatsala (2009). 'Design and development of a real-time DSP and FPGA-based integrated GPS-INS system for compact and low power applications'. In: *IEEE Transactions on Aerospace and Electronic Systems* 45.2, pp. 443–454.

Akgün, Gökhan et al. (2020). 'SysIDLib: a high-level synthesis FPGA library for online system identification'. In: *International Symposium on Applied Reconfigurable Computing*. Springer, pp. 97–107.

Alaimo, Andrea et al. (2013). 'Comparison between Euler and quaternion parametrization in UAV dynamics'. In: *AIP Conference Proceedings*. Vol. 1558. 1. American Institute of Physics, pp. 1228–1231.

Alfriend, Kyle (2002). 'Nonlinear considerations in satellite formation flying'. In: *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, p. 4741.

Arribas Lázaro, Javier (2012). 'GNSS array-based acquisition: theory and implementation'. In.

Bae, Jonghee, Seungho Yoon and Youdan Kim (2011). *Fault-tolerant attitude estimation for satellite using federated unscented Kalman filter*. InTech.

Balaban, Edward et al. (2009). 'Modeling, detection, and disambiguation of sensor faults for aerospace applications'. In: *IEEE Sensors Journal* 9.12, pp. 1907–1917.

Baoyin, Hexi, Li Junfeng and Gao Yunfeng (2002). 'Dynamical behaviors and relative trajectories of the spacecraft formation flying'. In: *Aerospace science and technology* 6.4, pp. 295–301.

Baroni, Leandro and Helio Koiti Kuga (2012). 'Analysis of attitude determination methods using GPS carrier phase measurements'. In: *Mathematical Problems in Engineering* 2012.

Barua, Amitabh, Purnendu Sinha and Khashayar Khorasani (2009). 'A diagnostic tree approach for fault cause identification in the attitude control subsystem of satellites'. In: *IEEE Transactions on Aerospace and Electronic Systems* 45.3, pp. 983–1002.

Bhogadi, Lokeswara Rao, Sasi Bhushana Rao Gottapu and VVS Konala (2015). 'MicroBlaze implementation of GPS/INS integrated system on Virtex-6 FPGA'. In: *SpringerPlus* 4.1, pp. 1–18.

Bing, Wang et al. (2013). 'Comparison of attitude determination approaches using multiple Global Positioning System (GPS) antennas'. In: *Geodesy and Geodynamics* 4.1, pp. 16–22.

Cadena, Arturo, Ronald Ponguillo and Daniel Ochoa (2017). 'Development of guidance, navigation and control system using fpga technology for an uav tricopter'. In: *Mechatronics and Robotics Engineering for Advanced and Intelligent Manufacturing*. Springer, pp. 363–375.

Cai, Lin et al. (2007). 'Using RBF neural network for fault diagnosis in satellite ADS'. In: *2007 IEEE International Conference on Control and Automation*. IEEE, pp. 1052–1055.

Calusdian, James, Xiaoping Yun and Eric Bachmann (2011). 'Adaptive-gain complementary filter of inertial and magnetic data for orientation estimation'. In: *2011 IEEE International Conference on Robotics and Automation*. IEEE, pp. 1916–1922.

Campos, Leandro José Evilásio and Edgar Campos Furtado (2017). 'Analysis of Quaternions Components in QUEST Algorithm-The Duality Problem'. In: *Proceedings of the XXXVIII Iberian Latin-American Congress on Computational Methods in Engineering, Florianópolis, Brazil*.

Carletta, Stefano, Paolo Teofilatto and M Salim Farissi (2020). 'A magnetometer-only attitude determination strategy for small satellites: Design of the algorithm and hardware-in-the-loop testing'. In: *Aerospace* 7.1, p. 3.

Carvajal-Godinez, Johan, Jian Guo and Eberhard Gill (2017). 'Agent-based algorithm for fault detection and recovery of gyroscope's drift in small satellite missions'. In: *Acta Astronautica* 139, pp. 181–188.

Cayrac, Didier, Didier Dubois and Henri Prade (1996). 'Handling uncertainty with possibility theory and fuzzy sets in a satellite fault diagnosis application'. In: *IEEE transactions on Fuzzy Systems* 4.3, pp. 251–269.

Chen, Xianliang, Youngho Eun and Xiaofeng Wu (2022). 'Feasibility Study of Neural Network in Satellite Attitude Determination'. In: *6th International Technical Conference on Advances in Computing, Control and Industrial Engineering (CCIE 2021)*. Springer, pp. 264–271.

Chen, Xianliang et al. (2023a). 'An Observation Model From Linear Interpolation for Quaternion-Based Attitude Estimation'. In: *IEEE Transactions on Instrumentation and Measurement* 72, pp. 1–12.

Chen, Xianliang et al. (2023b). 'Kalman filter and neural network fusion for fault detection and recovery in satellite attitude estimation'. Manuscript submitted for publication.

Chen, Xianliang et al. (2023c). 'Optimized FPGA Implementation of Fault Detection, Isolation and Recovery System for Satellite Attitude Estimation'. Manuscript submitted for publication.

Collinson, Richard PG (2013). *Introduction to avionics systems*. Springer Science & Business Media.

Crassidis, John L and F Landis Markley (2003). 'Unscented filtering for spacecraft attitude estimation'. In: *Journal of guidance, control, and dynamics* 26.4, pp. 536–542.

Crassidis, John L, F Landis Markley and Yang Cheng (2007). 'Survey of nonlinear attitude estimation methods'. In: *Journal of guidance, control, and dynamics* 30.1, pp. 12–28.

De Franceschi, M and D Zardi (2003). 'Evaluation of cut-off frequency and correction of filter-induced phase lag and attenuation in eddy covariance analysis of turbulence data'. In: *Boundary-layer meteorology* 108.2, pp. 289–303.

Dhahbane, Djamel, Abdelkrim Nemra and Samir Sakhi (2021). 'Attitude determination and attitude estimation in aircraft and spacecraft navigation. a survey'. In: *2020 2nd International Workshop on Human-Centric Smart Environments for Health and Well-being (IHSH)*. IEEE, pp. 187–192.

Ding, Wei et al. (2022). 'Improved attitude estimation accuracy by data fusion of a MEMS MARG sensor and a low-cost GNSS receiver'. In: *Measurement* 194, p. 111019.

Draper, Bruce A et al. (2003). 'Accelerated image processing on FPGAs'. In: *IEEE transactions on image processing* 12.12, pp. 1543–1551.

Du, Binhan, Jinlong Song and Zhiyong Shi (2019). 'An anomaly diagnosis method for redundant inertial measurement unit and its application with micro-electro-mechanical system sensors'. In: *Applied Sciences* 9.8, p. 1606.

Egeland, Olav and Jan Tommy Gravdahl (2002). *Modeling and simulation for automatic control*. Vol. 76. Marine Cybernetics Trondheim, Norway.

Eisenman, Allan and Carl Christian Liebe (1996). 'Operation and performance of a second generation, solid state, star tracker, the ASC'. In: *Acta astronautica* 39.9-12, pp. 697–705.

Feng, CHENG et al. (2019). 'Research on improvement of CNS+ GNSS+ INS ship-borne high precision real-time positioning and attitude determination algorithms'. In: *Bulletin of Surveying and Mapping* 5, p. 30.

Feng, Kaiqiang et al. (2017). 'A new quaternion-based Kalman filter for real-time attitude estimation using the two-step geometrically-intuitive correction algorithm'. In: *Sensors* 17.9, p. 2146.

Ferreira, Joao Canas and Jose Fonseca (2016). 'An FPGA implementation of a long short-term memory neural network'. In: *2016 International Conference on ReConFigurable Computing and FPGAs (ReConFig)*. IEEE, pp. 1–8.

Forbes, James Richard (2015). 'Attitude control with active actuator saturation prevention'. In: *Acta Astronautica* 107, pp. 187–195.

Foxlin, Eric (1996). 'Inertial head-tracker sensor fusion by a complementary separate-bias Kalman filter'. In: *Proceedings of the IEEE 1996 Virtual Reality Annual International Symposium*. IEEE, pp. 185–194.

Gao, Bingbing et al. (2018). 'Multi-sensor optimal data fusion for INS/GNSS/CNS integration based on unscented Kalman filter'. In: *International Journal of Control, Automation and Systems* 16, pp. 129–140.

Gao, Shihong et al. (2021). 'Adaptive fuzzy fault-tolerant control for the attitude tracking of spacecraft within finite time'. In: *Acta Astronautica* 189, pp. 166–180.

Goh, Shu Ting and Kay-Soon Low (2017). 'Survey of global-positioning-system-based attitude determination algorithms'. In: *Journal of Guidance, Control, and Dynamics* 40.6, pp. 1321–1335.

Gou, Bin, Yong-mei Cheng and Anton HJ de Ruiter (2019). 'INS/CNS navigation system based on multi-star pseudo measurements'. In: *Aerospace Science and Technology* 95, p. 105506.

Guerrier, Stephane et al. (2012). 'Fault detection and isolation in multiple MEMS-IMUs configurations'. In: *IEEE Transactions on Aerospace and Electronic Systems* 48.3, pp. 2015–2031.

Guo, Chengcheng et al. (2017a). 'HIGH-PRECISION ATTITUDE ESTIMATION METHOD OF STAR SENSORS AND GYRO BASED ON COMPLEMENTARY FILTER AND UNSCENTED KALMAN FILTER.' In: *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences* 42.

Guo, Dingfei, Maiying Zhong and Donghua Zhou (2017b). 'Multisensor data-fusion-based approach to airspeed measurement fault detection for unmanned aerial vehicles'. In: *IEEE Transactions on Instrumentation and Measurement* 67.2, pp. 317–327.

Guo, Kaiyuan et al. (2017c). 'A survey of FPGA-based neural network accelerator'. In: *arXiv preprint arXiv:1712.08934*.

Gutiérrez, Samuel T, César I Fuentes and Marcos A Dıaz (2020). 'Introducing sost: An ultra-low-cost star tracker concept based on a raspberry pi and open-source astronomy software'. In: *IEEE Access* 8, pp. 166320–166334.

Hajiyev, Chingiz (2014). 'Generalized Rayleigh quotient based innovation covariance testing applied to sensor/actuator fault detection'. In: *Measurement* 47, pp. 804–812.

Han, Song et al. (2017). 'Ese: Efficient speech recognition engine with sparse lstm on fpga'. In: *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, pp. 75–84.

Harl, Nathan, Karthikeyan Rajagopal and SN Balakrishnan (2013). 'Neural network based modified state observer for orbit uncertainty estimation'. In: *Journal of Guidance, Control, and Dynamics* 36.4, pp. 1194–1209.

Hasan, Muhammad Noman, Muhammad Haris and Shiyin Qin (2022). 'Flexible spacecraft's active fault-tolerant and anti-unwinding attitude control with vibration suppression'. In: *Aerospace Science and Technology* 122, p. 107397.

Horn, Berthold, Berthold Klaus and Paul Horn (1986). *Robot vision*. MIT press.

Hou, Yueqi et al. (2022). 'Fuzzy adaptive fixed-time fault-tolerant attitude tracking control for tailless flying wing aircrafts'. In: *Aerospace Science and Technology* 130, p. 107950.

Hu, Gaoge et al. (2016). 'Modified federated Kalman filter for INS/GNSS/CNS integration'. In: *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering* 230.1, pp. 30–44.

Hwang, Inseok et al. (2009). 'A survey of fault detection, isolation, and reconfiguration methods'. In: *IEEE transactions on control systems technology* 18.3, pp. 636–653.

javaTpoint (2022). *Flow of C Program*. URL: https://www.javatpoint.com/flow-of-c-program.

Jew, Michael, Aly El-Osery and Stephen Bruder (2010). 'Implementation of an FPGA-based aided IMU on a low-cost autonomous outdoor robot'. In: *IEEE/ION Position, Location and Navigation Symposium*. IEEE, pp. 1043–1051.

Jin, H and HY Zhang (1997). 'Configuration of redundant sensor system and its fault detection using parity vector method'. In: *IFAC Proceedings Volumes* 30.18, pp. 839–844.

Jing, Hao et al. (2022). 'Integrity Monitoring of GNSS/INS Based Positioning Systems for Autonomous Vehicles: State-of-the-Art and Open Challenges'. In: *IEEE Transactions on Intelligent Transportation Systems* 23.9, pp. 14166–14187. DOI: 10.1109/TITS.2022.3149373.

Ju, Gwanghyoek et al. (2000). 'Micro star tracker and attitude determination system'. In: *Reducing the Cost of Spacecraft Ground Systems and Operations*, pp. 209–216.

Julier, Simon J and Jeffrey K Uhlmann (2004). 'Unscented filtering and nonlinear estimation'. In: *Proceedings of the IEEE* 92.3, pp. 401–422.

Kaula, William M (2013). *Theory of satellite geodesy: applications of satellites to geodesy*. Courier Corporation.

Kehl, Wadim et al. (2017). 'Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again'. In: *Proceedings of the IEEE international conference on computer vision*, pp. 1521–1529.

Kim, HY, JL Junkins and JN Juang (2003). 'An efficient and robust singular value method for star pattern recognition and attitude determination'. In: *NASA, Tech Rep TM-2003-212142*.

Kim, Jeongrae (2000). *Simulation study of a low-low satellite-to-satellite tracking mission*. The University of Texas at Austin.

Kim, Son-Goo et al. (2007). 'Kalman filtering for relative spacecraft attitude and position estimation'. In: *Journal of Guidance, Control, and Dynamics* 30.1, pp. 133–143.

Kolomenkin, Michael et al. (2008). 'Geometric voting algorithm for star trackers'. In: *IEEE Transactions on Aerospace and Electronic Systems* 44.2, pp. 441–456. DOI: 10.1109/TAES.2008.4560198.

Kosik, Jean Claude (1991). 'Star pattern identification aboard an inertially stabilized aircraft'. In: *Journal of guidance, control, and dynamics* 14.2, pp. 230–235.

Kottath, Rahul et al. (2017). 'Multiple model adaptive complementary filter for attitude estimation'. In: *Aerospace Science and Technology* 69, pp. 574–581.

Kuang, Jinlu and Soonhie Tan (2002). 'GPS-based attitude determination of gyrostat satellite by quaternion estimation algorithms'. In: *Acta Astronautica* 51.11, pp. 743–759.

Lamy Au Rousseau, G., J. Bostel and B. Mazari (2005). 'Star recognition algorithm for APS star tracker: oriented triangles'. In: *IEEE Aerospace and Electronic Systems Magazine* 20.2, pp. 27–31. DOI: 10.1109/MAES.2005.1397146.

Le, Huy Xuan and Saburo Matunaga (2014). 'A residual based adaptive unscented Kalman filter for fault recovery in attitude determination system of microsatellites'. In: *Acta Astronautica* 105.1, pp. 30–39.

Lee, Jung Keun and Mi Jin Choi (2017). 'A sequential orientation kalman filter for AHRS limiting effects of magnetic disturbance to heading estimation'. In: *Journal of Electrical Engineering and Technology* 12.4, pp. 1675–1682.

Lee, Jung Keun and Edward J Park (2009). 'Minimum-order Kalman filter with vector selector for accurate estimation of human body orientation'. In: *IEEE Transactions on Robotics* 25.5, pp. 1196–1201.

Lefferts, Ern J, F Landis Markley and Malcolm D Shuster (1982). 'Kalman filtering for spacecraft attitude estimation'. In: *Journal of Guidance, Control, and Dynamics* 5.5, pp. 417–429.

Li, Yong (2000). 'A new approach for attitude determination using GPS carrier phase measurements'. In: *AIAA Guidance, Navigation, and Control Conference and Exhibit*, p. 4467.

Li, You et al. (2020). 'Inertial sensing meets artificial intelligence: Opportunity or challenge?' In: *arXiv preprint arXiv:2007.06727*.

Liang, YD et al. (2011). 'Attitude estimation of a quad-rotor aircraft based on complementary filter'. In: *Transducer and Microsystem Technologies* 30.11, pp. 56–58.

Liang, Yueqian and Yingmin Jia (2015). 'A nonlinear quaternion-based fault-tolerant SINS/GNSS integrated navigation method for autonomous UAVs'. In: *Aerospace Science and Technology* 40, pp. 191–199.

Liebe, Carl Ch (1995). 'Star trackers for attitude determination'. In: *IEEE Aerospace and Electronic Systems Magazine* 10.6, pp. 10–16.

Liebe, Carl Christian (1993). 'Pattern recognition of star constellations for spacecraft applications'. In: *IEEE Aerospace and Electronic Systems Magazine* 8.1, pp. 31–39.

Liem, Martijn C and Dariu M Gavrila (2014). 'Coupled person orientation estimation and appearance modeling using spherical harmonics'. In: *Image and Vision Computing* 32.10, pp. 728–738.

Lim, Jun Kyu and Chan Gook Park (2014). 'Satellite fault detection and isolation scheme with modified adaptive fading EKF'. In: *Journal of Electrical Engineering and Technology* 9.4, pp. 1401–1410.

Lin, D et al. (2004). 'GPS-based attitude determination for microsatellite using three-antenna technology'. In: *2004 IEEE Aerospace Conference Proceedings (IEEE Cat. No. 04TH8720)*. Vol. 2. IEEE, pp. 1024–1029.

Liu, Di and Xiyuan Chen (2022). 'An ANN-Based Data Fusion Algorithm for INS/CNS Integrated Navigation System'. In: *IEEE Sensors Journal* 22.8, pp. 7846–7854.

Liu, Fei et al. (2014). 'An improved quaternion Gauss–Newton algorithm for attitude determination using magnetometer and accelerometer'. In: *Chinese Journal of aeronautics* 27.4, pp. 986–993.

Liu, XC, S Zhang, LZ Li et al. (2012). ‘Quaternion-based algorithm for orientation estimation from MARG sensors’. In: *J. Tsinghua Univ.(Sci & Tech)* 52.5, pp. 627–631.

Liu, Yue et al. (2018). ‘Implementation and analysis of tightly coupled global navigation satellite system precise point positioning/inertial navigation system (GNSS PPP/INS) with insufficient satellites for land vehicle navigation’. In: *Sensors* 18.12, p. 4305.

Liu, Yunjie and Xiaofeng Wu (2022). ‘An FPGA-based General-purpose Feature Detection Algorithm for Space Applications’. In: *IEEE Transactions on Aerospace and Electronic Systems*.

Lu, Gang et al. (1993). ‘Attitude determination in a survey launch using multi-antenna GPS technologies’. In: *Proceedings of National Technical Meeting, The Institute of Navigation, Alexandria, VA*. Vol. 251. Citeseer, p. 260.

Madgwick, Sebastian OH, Andrew JL Harrison and Ravi Vaidyanathan (2011). ‘Estimation of IMU and MARG orientation using a gradient descent algorithm’. In: *2011 IEEE international conference on rehabilitation robotics*. IEEE, pp. 1–7.

Magnis, Lionel and Nicolas Petit (2014). ‘Estimation of 3D rotation for a satellite from Sun sensors’. In: *IFAC Proceedings Volumes* 47.3, pp. 10004–10011.

Makni, Aida, Hassen Fourati and Alain Y Kibangou (2014). ‘Adaptive Kalman filter for MEMS-IMU based attitude estimation under external acceleration and parsimonious use of gyroscopes’. In: *2014 European Control Conference (ECC)*. IEEE, pp. 1379–1384.

Marins, João Luıs et al. (2001). ‘An extended Kalman filter for quaternion-based orientation estimation using MARG sensors’. In: *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No. 01CH37180)*. Vol. 4. IEEE, pp. 2003–2011.

Markley, F Landis (2002). ‘Fast quaternion attitude estimation from two vector measurements’. In: *Journal of Guidance, Control, and Dynamics* 25.2, pp. 411–414.

Markley, F Landis and John L Crassidis (2014). *Fundamentals of spacecraft attitude determination and control*. Vol. 1286. Springer.

Markley, F Landis and Daniele Mortari (2000). ‘Quaternion attitude estimation using vector observations’. In: *The Journal of the Astronautical Sciences* 48.2, pp. 359–380.

Mehra, Raman et al. (1995). ‘Adaptive Kalman filtering, failure detection and identification for spacecraft attitude estimation’. In: *Proceedings of International Conference on Control Applications*. IEEE, pp. 176–181.

Mei, Yafei et al. (2022). ‘Fuzzy adaptive sliding mode fault estimation and fixed-time fault-tolerant control for coupled spacecraft based on SE (3)’. In: *Aerospace Science and Technology*, p. 107673.

Meibo, Lv, Naqvi Najam Abbas and Li YanJun (2013). ‘Exploring the ambiguity resolution in spacecraft attitude determination using GNSS phase measurement’. In: *International Journal of Humanities and Social Sciences* 7.7, pp. 1976–1980.

Montenbruck, Oliver et al. (2002). ‘A real-time kinematic GPS sensor for spacecraft relative navigation’. In: *Aerospace Science and Technology* 6.6, pp. 435–449.

Mortari, Daniele (1997). ‘Search-less algorithm for star pattern recognition’. In: *The Journal of the Astronautical Sciences* 45, pp. 179–194.

Mortari, Daniele et al. (2004). ‘The pyramid star identification technique’. In: *Navigation* 51.3, pp. 171–183.

Mu, Rongjun et al. (2020). ‘INS/CNS Deeply Integrated Navigation Method of Near Space Vehicles’. In: *Sensors* 20.20, p. 5885.

Muruganandan, VA (2018). ‘Hardware Design and Star Selection Algorithm for Arcsecond Pico Star Tracker’. In.

Muthusamy, Venkatesh and Krishna Dev Kumar (2021). ‘A novel data-driven method for fault detection and isolation of control moment gyroscopes onboard satellites’. In: *Acta Astronautica* 180, pp. 604–621.

Nalepa, Jakub et al. (2022). ‘Evaluating algorithms for anomaly detection in satellite telemetry data’. In: *Acta Astronautica* 198, pp. 689–701.

Nasrolahi, Seiied Saeed and Farzaneh Abdollahi (2018). ‘Sensor fault detection and recovery in satellite attitude control’. In: *Acta Astronautica* 145, pp. 275–283.

Nasrolahi, Seiied Saeed, Hossein Bolandi and Mostafa Abedi (2012). ‘An analytical fault detection algorithm using Euler angles-numerical and practical results’. In: *2012 12th International Conference on Control, Automation and Systems*. IEEE, pp. 1227–1232.

Nasrolahi, Seyyed Saeed, Hossein Bolandi and Mostafa Abedi (2014). 'Analytical fault detection and isolation algorithms based on rotation matrices for a three axis stabilized satellite'. In: *automatika* 55.3, pp. 330–342.

Ni, Shaobo and Cui Zhang (2011). 'Attitude determination of nano satellite based on gyroscope, sun sensor and magnetometer'. In: *Procedia Engineering* 15, pp. 959–963.

Ning, Xiaolin et al. (2013). 'Initial position and attitude determination of lunar rovers by INS/CNS integration'. In: *Aerospace Science and Technology* 30.1, pp. 323–332.

Padgett, Curtis and Kenneth Kreutz-Delgado (1997). 'A grid algorithm for autonomous star identification'. In: *IEEE Transactions on Aerospace and Electronic Systems* 33.1, pp. 202–213.

Paladugu, Lalitha, Brian Williams and Marco Schoen (2003). 'Star pattern recognition for attitude determination using genetic algorithms'. In.

Patton, Ron J et al. (2010). 'Robust FDI applied to thruster faults of a satellite system'. In: *Control Engineering Practice* 18.9, pp. 1093–1109.

Peng, HM, FR Chang and LS Wang (1999). 'Attitude determination using GPS carrier phase and compass data'. In: *Proceedings of the 1999 National Technical Meeting of The Institute of Navigation*, pp. 727–732.

Philip, N K and MR Ananthasayanam (2003). 'Relative position and attitude estimation and control schemes for the final phase of an autonomous docking mission of spacecraft'. In: *Acta Astronautica* 52.7, pp. 511–522.

Pirmoradi, FN, F Sassani and CW De Silva (2009). 'Fault detection and diagnosis in a spacecraft attitude determination system'. In: *Acta Astronautica* 65.5-6, pp. 710–729.

Popescu, Gabriel (2014). 'Pixel geolocation algorithm for satellite scanner data'. In: *Scientific Papers. Series E. Land Reclamation, Earth Observation & Surveying, Environmental Engineering* 3, pp. 127–136.

Pourtakdoust, Seid H, M Fakhari Mehrjardi and MH Hajkarim (2022). 'Attitude estimation and control based on modified unscented Kalman filter for gyro-less satellite with faulty sensors'. In: *Acta Astronautica* 191, pp. 134–147.

Psiaki, Mark L (1999). 'Autonomous low-earth-orbit determination from magnetometer and sun sensor data'. In: *Journal of Guidance, Control, and Dynamics* 22.2, pp. 296–304.

Purcell, George et al. (1998). ‘Autonomous formation flyer (AFF) sensor technology development’. In.

Qiao, Bing et al. (2013). ‘Relative position and attitude estimation of spacecrafts based on dual quaternion for rendezvous and docking’. In: *Acta Astronautica* 91, pp. 237–244.

Qu, Q et al. (2010). ‘Airborne SINS/CNS location integrated system’. In: *J. Nanjing Univ. Sci. Technol.(Nat. Sci.)* 34.6, pp. 729–732.

Raskaliyev, Almat et al. (2020). ‘GNSS-based attitude determination techniques—a comprehensive literature survey’. In: *IEEE Access* 8, pp. 24873–24886.

Ren, Shaoqing et al. (2015). ‘Faster r-cnn: Towards real-time object detection with region proposal networks’. In: *Advances in neural information processing systems* 28.

Rijlaarsdam, David et al. (2020). ‘A survey of lost-in-space star identification algorithms since 2009’. In: *Sensors* 20.9, p. 2579.

Rufino, Giancarlo and Domenico Accardo (2003). ‘Enhancement of the centroiding algorithm for star tracker measure refinement’. In: *Acta Astronautica* 53.2, pp. 135–147.

Sabatelli, Simone et al. (2012). ‘A double-stage Kalman filter for orientation tracking with an integrated processor in 9-D IMU’. In: *IEEE Transactions on Instrumentation and Measurement* 62.3, pp. 590–598.

Sabatini, Angelo M (2006). ‘Quaternion-based extended Kalman filter for determining orientation by inertial and magnetic sensing’. In: *IEEE transactions on Biomedical Engineering* 53.7, pp. 1346–1356.

Sabatini, Angelo Maria (2011). ‘Kalman-filter-based orientation determination using inertial/magnetic sensors: Observability analysis and performance evaluation’. In: *Sensors* 11.10, pp. 9182–9206.

Samaan, M.A., D. Mortari and J.L. Junkins (2005). ‘Recursive mode star identification algorithms’. In: *IEEE Transactions on Aerospace and Electronic Systems* 41.4, pp. 1246–1254. DOI: 10.1109/TAES.2005.1561885.

Samaan, Malak and Stephan Theil (2012). ‘Development of a low cost star tracker for the SHEFEX mission’. In: *Aerospace science and technology* 23.1, pp. 469–478.

Samaan, Malak A, Stephen R Steffes and Stephan Theil (2011). ‘Star tracker real-time hardware in the loop testing using optical star simulator’. In: *Spaceflight Mechanics* 140.

Scharnagl, Julian et al. (2022). ‘NetSat—Challenges and lessons learned of a formation of 4 nano-satellites’. In: *Acta Astronautica* 201, pp. 580–591.

Sekhavat, Pooya, Qi Gong and I Michael Ross (2007). ‘NPSAT1 parameter estimation using unscented kalman filtering’. In: *2007 American Control Conference*. IEEE, pp. 4445–4451.

Seo, Eun-Ho et al. (2011). ‘Quaternion-based orientation estimation with static error reduction’. In: *2011 IEEE International Conference on Mechatronics and Automation*. IEEE, pp. 1624–1629.

Shakouri, Amir and Nima Assadian (2018). ‘Fault detection and isolation of satellite gyroscopes using relative positions in formation flying’. In: *Aerospace Science and Technology* 78, pp. 403–417.

Sharma, Sumant, Connor Beierle and Simone D’Amico (2018). ‘Pose estimation for noncooperative spacecraft rendezvous using convolutional neural networks’. In: *2018 IEEE Aerospace Conference*. IEEE, pp. 1–12.

Sharma, Sumant and Simone D’Amico (2020). ‘Neural network-based pose estimation for noncooperative spacecraft rendezvous’. In: *IEEE Transactions on Aerospace and Electronic Systems* 56.6, pp. 4638–4658.

Sheng, Gao et al. (2018). ‘Neural network-based fault diagnosis scheme for satellite attitude control system’. In: *2018 Chinese Control And Decision Conference (CCDC)*. IEEE, pp. 3990–3995.

Shuster, Malcolm D (1990). ‘Kalman filtering of spacecraft attitude and the QUEST model’. In: *Journal of the Astronautical Sciences* 38, pp. 377–393.

Shuster, Malcolm David and S D_ Oh (1981). ‘Three-axis attitude determination from vector observations’. In: *Journal of guidance and Control* 4.1, pp. 70–77.

Shuster, Malcolmd (1989). ‘A simple Kalman filter and smoother for spacecraft attitude’. In: *Journal of the Astronautical Sciences* 37.1, pp. 89–106.

Soh, Jeremy and Xiaofeng Wu (2016). ‘An fpga-based unscented kalman filter for system-on-chip applications’. In: *IEEE Transactions on Circuits and Systems II: Express Briefs* 64.4, pp. 447–451.

— (2017). ‘A five-stage pipeline architecture of the unscented Kalman filter for system-on-chip applications’. In: *IEEE Transactions on Industrial Electronics* 65.3, pp. 2785–2794.

Soken, Halil Ersin and Chingiz Hajiyev (2010). ‘Pico satellite attitude estimation via robust unscented Kalman filter in the presence of measurement faults’. In: *ISA transactions* 49.3, pp. 249–256.

Soken, Halil Ersin, Chingiz Hajiyev and Shin-ichiro Sakai (2014). ‘Robust Kalman filtering for small satellite attitude estimation in the presence of measurement faults’. In: *European Journal of Control* 20.2, pp. 64–72.

Song, Jianing, Duarte Rondao and Nabil Aouf (2022). ‘Deep learning-based spacecraft relative navigation methods: A survey’. In: *Acta Astronautica* 191, pp. 22–40.

Song, Yong Duan and Wen Chuan Cai (2012). ‘New intermediate quaternion based control of spacecraft: part I—almost global attitude tracking’. In: *International Journal of Innovative Computing, Information and Control* 8.10, pp. 7307–7319.

Souissi, Omar et al. (2012). ‘Optimization of run-time mapping on heterogeneous cpu/fpga architectures’. In: *9th International Conference on Modeling, Optimization & SIMulation*.

Spratling IV, Benjamin B and Daniele Mortari (2009). ‘A survey on star identification algorithms’. In: *Algorithms* 2.1, pp. 93–107.

Sun, Shu-Li and Zi-Li Deng (2004). ‘Multi-sensor optimal information fusion Kalman filter’. In: *Automatica* 40.6, pp. 1017–1023.

Sun, Wei, Peilun Sun and Jiaji Wu (2022). ‘An Adaptive Fusion Attitude and Heading Measurement Method of MEMS/GNSS Based on Covariance Matching’. In: *Micromachines* 13.10, p. 1787.

Sunde, Bernt Ove (2005). ‘Sensor modelling and attitude determination for micro-satellite’. In: *MSc. thesis, Norwegian University of Science and Technology* 2005.

Tekawy, Jonathan A, Patrick Wang and Charles W Gray (1996). ‘Scanning horizon sensor attitude correction for Earth oblateness’. In: *Journal of guidance, control, and dynamics* 19.3, pp. 706–708.

Teunissen, Peter (2007). ‘A general multivariate formulation of the multi-antenna GNSS attitude determination problem’. In: *Artificial satellites* 42.2, pp. 97–111.

Teunissen, PJG (2008). 'GNSS ambiguity resolution for attitude determination: theory and method'. In: *Proceedings of the International Symposium on GPS/GNSS*.

Tipaldi, Massimo and Bernhard Bruenjes (2015). 'Survey on fault detection, isolation, and recovery strategies in the space domain'. In: *Journal of Aerospace Information Systems* 12.2, pp. 235–256.

Torisaka, Ayako et al. (2013). 'Control of electromagnetic current at final docking phase of small satellites'. In: *Journal of Space Engineering* 6.1, pp. 44–55.

Tweddle, Brent E and Alvar Saenz-Otero (2015). 'Relative computer vision-based navigation for small inspection spacecraft'. In: *Journal of guidance, control, and dynamics* 38.5, pp. 969–978.

Uhlmann, JK and H Durrant-Whyte (1995). 'A new approach for filtering nonlinear system'. In: *Proc. Amer. Control Conf.* Pp. 1628–1632.

Ushaq, Muhammad, Fang Jian Cheng and Ali Jamshaid (2013). 'A fault tolerant SINS/GPS/CNS integrated navigation scheme realized through federated Kalman filter'. In: *Applied Mechanics and Materials*. Vol. 332. Trans Tech Publ, pp. 104–110.

Valenti, Roberto G, Ivan Dryanovski and Jizhong Xiao (2015). 'A linear Kalman filter for MARG orientation estimation using the algebraic quaternion algorithm'. In: *IEEE Transactions on Instrumentation and Measurement* 65.2, pp. 467–481.

Venkateswaran, N, MS Siva and PS Goel (2002). 'Analytical redundancy based fault detection of gyroscopes in spacecraft applications'. In: *Acta Astronautica* 50.9, pp. 535–545.

Wahba, Grace (1965). 'A least squares estimate of satellite attitude'. In: *SIAM review* 7.3, pp. 409–409.

Waheed, Owais Talaat and Ibrahim M Elfadel (2018). 'FPGA sensor fusion system design for IMU arrays'. In: *2018 Symposium on Design, Test, Integration & Packaging of MEMS and MOEMS (DTIP)*. IEEE, pp. 1–5.

Wang, Chenghao and Zhongqiang Luo (2022). 'A Review of the Optimal Design of Neural Networks Based on FPGA'. In: *Applied Sciences* 12.21, p. 10771.

Wang, Dingjie et al. (2020). 'Constrained MEMS-Based GNSS/INS Tightly Coupled System With Robust Kalman Filter for Accurate Land Vehicular Navigation'. In: *IEEE Transactions on Instrumentation and Measurement* 69.7, pp. 5138–5148. DOI: 10.1109/TIM.2019.2955798.

Wang, Li, Zheng Zhang and Ping Sun (2015a). 'Quaternion-based Kalman filter for AHRS using an adaptive-step gradient descent algorithm'. In: *International Journal of Advanced Robotic Systems* 12.9, p. 131.

Wang, Rixin, Yao Cheng and Minqiang Xu (2015b). 'Analytical redundancy based fault diagnosis scheme for satellite attitude control systems'. In: *Journal of the Franklin Institute* 352.5, pp. 1906–1931.

Wang, Rong et al. (2016). 'Chi-square and SPRT combined fault detection for multisensor navigation'. In: *IEEE Transactions on Aerospace and Electronic Systems* 52.3, pp. 1352–1365.

Wang, Yi and Yajun Ha (2013). 'FPGA-based 40.9-Gbits/s masked AES with area optimization for storage area network'. In: *IEEE Transactions on Circuits and Systems II: Express Briefs* 60.1, pp. 36–40.

Wei, Xinguo, Guangjun Zhang and Jie Jiang (2009). 'Star identification algorithm based on log-polar transform'. In: *Journal of Aerospace Computing, Information, and Communication* 6.8, pp. 483–490.

Weill, Lawrence R (1994). 'Optimal GPS attitude determination without ambiguity resolution'. In: *Proceedings of the 1994 National Technical Meeting of The Institute of Navigation*, pp. 433–439.

Weimer, Florian, Michael Frangenberg and Walter Fichter (2015). 'Pipelined particle filter with nonobservability measure for attitude and velocity estimation'. In: *Journal of Guidance, Control, and Dynamics* 38.3, pp. 506–518.

Wertz, James R (1978). 'Mathematical Models of Attitude Hardware'. In: *Spacecraft Attitude Determination and Control*, pp. 217–277.

— (2012). *Spacecraft attitude determination and control*. Vol. 73. Springer Science & Business Media.

Williamson, Walton R et al. (2009). 'Fault detection and isolation for deep space satellites'. In: *Journal of guidance, control, and dynamics* 32.5, pp. 1570–1584.

Wu, Ling et al. (2022). 'A Novel Two-Stage Adaptive Filtering Model-Based GNSS/INS Tightly Coupled Precise Relative Navigation Algorithm for Autonomous Aerial Refueling'. In: *International Journal of Aerospace Engineering* 2022.

Wu, Weiren, Xiaolin Ning and Lingling Liu (2013). 'New celestial assisted INS initial alignment method for lunar explorer'. In: *Journal of Systems Engineering and Electronics* 24.1, pp. 108–117.

Xilinx (2022a). *Zynq-7000 SoC*. URL: https://www.xilinx.com/products/silicon-devices/soc/zynq-7000.html#productAdvantages.

— (2022b). *Zynq-7000 SoC*. URL: https://pynq.readthedocs.io/en/v2.3/overlay_design_methodology/pspl_interface.html.

Xing, Yanjun et al. (2010). 'Relative position and attitude estimation for satellite formation with coupled translational and rotational dynamics'. In: *Acta Astronautica* 67.3-4, pp. 455–467.

Xinyuan, Dong et al. (2012). 'An improved CDKF algorithm based on RBF neural network for satellite attitude determination'. In: *2012 International Conference on Image Analysis and Signal Processing*. IEEE, pp. 1–7.

Xiong, K, CW Chan and HY Zhang (2007). 'Detection of satellite attitude sensor faults using the UKF'. In: *IEEE Transactions on Aerospace and Electronic Systems* 43.2, pp. 480–491.

Xiong, Zhi et al. (2013). 'A new dynamic vector formed information sharing algorithm in federated filter'. In: *Aerospace Science and Technology* 29.1, pp. 37–46.

Xu, Fei, Guangle Gao and Longqiang Ni (2022). 'A New Adaptive Federated Cubature Kalman Filter Based on Chi-Square Test for SINS/GNSS/SRS/CNS Integration'. In: *Mathematical Problems in Engineering* 2022.

Xu, Ying et al. (2023). 'Motion-Constrained GNSS/INS Integrated Navigation Method Based on BP Neural Network'. In: *Remote Sensing* 15.1, p. 154.

Xu, Zhenkai et al. (2010). 'Novel hybrid of LS-SVM and Kalman filter for GPS/INS integration'. In: *The Journal of Navigation* 63.2, pp. 289–299.

Xue, Yali et al. (2020). ‘Pipeline Gaussian particle filter and hardware design for attitude estimation with UAV’. In: *Mathematical Problems in Engineering* 2020.

Yang, Shujie et al. (2022). ‘A SINS/CNS integrated navigation scheme with improved mathematical horizon reference’. In: *Measurement* 195, p. 111028.

Yelubayev, S et al. (2015). ‘Development of star tracker for satellite’. In: *Mech. Mechatron. Eng* 9, p. 23521.

Yoon, Seungho et al. (2011). ‘Experimental evaluation of fault diagnosis in a skew-configured UAV sensor system’. In: *Control Engineering Practice* 19.2, pp. 158–173.

Yu, Feng et al. (2014). ‘Stereo-vision-based relative pose estimation for the rendezvous and docking of noncooperative satellites’. In: *Mathematical Problems in Engineering* 2014.

Yuan, Zhengguo et al. (2021). ‘Fault detection, isolation, and reconstruction for satellite attitude sensors using an adaptive hybrid method’. In: *IEEE Transactions on Instrumentation and Measurement* 70, pp. 1–12.

Yun, Xiaoping et al. (2005). ‘Implementation and experimental results of a quaternion-based Kalman filter for human body motion tracking’. In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. IEEE, pp. 317–322.

Zanetti, Renato et al. (2009). ‘Norm-constrained Kalman filtering’. In: *Journal of guidance, control, and dynamics* 32.5, pp. 1458–1465.

Zhai, Kun (2017). ‘Angular velocity determination of spinning solar sails using only a sun sensor’. In: *Chinese Journal of Aeronautics* 30.1, pp. 412–418.

Zhang, Yiwei et al. (2017). ‘A power-efficient accelerator based on FPGAs for LSTM network’. In: *2017 IEEE International Conference on Cluster Computing (CLUSTER)*. IEEE, pp. 629–630.

Zhang, Zhi-Qiang, Xiao-Li Meng and Jian-Kang Wu (2012). ‘Quaternion-based Kalman filter with vector selection for accurate orientation tracking’. In: *IEEE Transactions on Instrumentation and Measurement* 61.10, pp. 2817–2824.

Zhao, Minghao et al. (2019). ‘Real-time underwater image recognition with FPGA embedded system for convolutional neural network’. In: *Sensors* 19.2, p. 350.

Zhou, Jun et al. (2016). ‘A scheme of satellite multi-sensor fault-tolerant attitude estimation’. In: *Transactions of the Institute of Measurement and Control* 38.9, pp. 1053–1063.

Zhu, Hailong, Bin Liang and Tao Zhang (2018). 'A robust and fast star identification algorithm based on an ordered set of points pattern'. In: *Acta Astronautica* 148, pp. 327–336.

Zhu, Yanping et al. (2022). 'Attitude Solving Algorithm and FPGA Implementation of Four-Rotor UAV Based on Improved Mahony Complementary Filter'. In: *Sensors* 22.17, p. 6411.