FULL-POSE ESTIMATION AND TRACKING CONTROL FOR A MULTI-ROTOR AIRCRAFT PACKAGE EXCHANGE

by

Trent P. Smith

A thesis submitted in partial fulfillment
of the requirements for the degree

of

MASTER OF SCIENCE

in

Electrical Engineering

Approved:

Randall Christensen, Ph.D. Major Professor	Greg Droge, Ph.D. Committee Member
Jonathan Phillips, Ph.D. Committee Member	Richard S. Inouye, Ph.D. Vice Provost for Graduate Studies

UTAH STATE UNIVERSITY
Logan, Utah

2019

Copyright © Trent P. Smith 2019

All Rights Reserved

ABSTRACT

Full-Pose Estimation and Tracking Control for a Multi-Rotor Aircraft Package Exchange

by

Trent P. Smith, Master of Science

Utah State University, 2019

Major Professor: Randall Christensen, Ph.D.
Department: Electrical and Computer Engineering

In this work, research to develop guidance, navigation, and control algorithms for a package exchange maneuver of two quad-rotor aircraft is presented. First, research platforms and tool development is discussed. Second, a differential flatness and LQR trajectory tracking controller is designed for full pose synchronization of two quad-rotor robots. The controller is used to guide a designated follower quad-rotor aircraft to track a leader aircraft's position and attitude. The follower aircraft is equipped with a 2-DOF manipulator to compensate for under-actuation of the vehicle. Finally, a sensor architecture study for relative navigation of Unmanned Aerial Vehicles (UAV) is presented. The architecture study develops multiple navigation solutions, considers each solution's appropriateness for close-proximity missions, and compares performance.

(129 pages)

PUBLIC ABSTRACT

Full-Pose Estimation and Tracking Control for a Multi-Rotor Aircraft Package Exchange

Trent P. Smith

In this work, research to develop algorithms for a package exchange maneuver between two quad-rotor aircraft is presented. First, the development of tools used for this research is discussed. Second, a controller is designed that synchronizes the flight paths and motion of two quad-rotor robots. The controller is used to guide a designated follower quad-rotor to follow a leader aircraft's position and orientation. The follower aircraft is equipped with a simple mechanical manipulator to compensate for limitations in the aircrafts maneuverability. finally, a sensor architecture study for relative navigation of Unmanned Aerial Vehicles (UAV) is presented. The architecture study presents typical navigation solutions, considers each solution's appropriateness for close-proximity missions, and compares performance.

ACKNOWLEDGMENTS

I'd like to thank my advisor Dr. Randall Christensen for his mentorship, teachings, and support which made this work possible. I'd also like to thank the members of my committee Dr. Greg Droge and Dr. Jonathan Phillips for their insight and support.

Finally, I'd like to thank my wife Morgan for her patience and support. I cannot thank you enough for your sacrifices that made my achievements possible.

Trent P. Smith

CONTENTS

Page

ABSTRACT . . . iii

PUBLIC ABSTRACT . . . iv

ACKNOWLEDGMENTS . . . v

LIST OF TABLES . . . ix

LIST OF FIGURES . . . x

ACRONYMS . . . xiii

1 INTRODUCTION . . . 1
1.1 Contributions . . . 2
1.2 Overview . . . 3

2 BACKGROUND AND LITERATURE REVIEW . . . 4
2.1 Aerial Robotics . . . 4
2.2 Trajectory Tracking . . . 5
2.3 Relative Navigation . . . 6
2.3.1 Vision Only Navigation . . . 6
2.3.2 Other Relative Navigation Architectures . . . 7

3 RESEARCH TOOLS DEVELOPMENT . . . 9
3.1 Notation . . . 9
3.2 States and Coordinate Frames . . . 10
3.2.1 Coordinate Frames . . . 10
3.2.2 Controller State Variables . . . 12
3.2.3 Navigation State Variables . . . 13
3.3 Multi-Rotor Simulation . . . 14
3.3.1 Quad-rotor Kinematics . . . 15
3.3.2 Rigid Body Dynamics . . . 15
3.4 Synthesized Sensor Measurements . . . 17

4 FULL-POSE TRACKING CONTROLLER . . . 19
4.1 Controller Model . . . 19
4.1.1 Pan-Tilt Device Model . . . 19
4.1.2 Nonlinear State Space Representation . . . 22
4.1.3 Linear Model . . . 24
4.2 Controller Design . . . 27
4.2.1 Differential Flatness . . . 28
4.2.2 LQR . . . 29
4.3 Implementation and Simulation . . . 29

5 NAVIGATION ARCHITECTURE STUDY . . . 31
5.1 Study Structure . . . 31
5.2 Kalman Filter Overview . . . 32
5.3 System Models . . . 33
5.3.1 Full System Models . . . 33
5.3.2 Linear Error Models . . . 39
5.4 Measurement Models . . . 44
5.4.1 GPS Position Measurements . . . 44
5.4.2 LOS Measurement Model . . . 45
5.4.3 Linear Measurement Models for Sensor Architectures . . . 51
5.5 Kalman Update and State Estimate Correction . . . 52
5.6 Validation and Simulation . . . 54
5.6.1 Simulation Parameters . . . 54
5.6.2 Sensor Values . . . 57
5.6.3 Key Results . . . 59
5.6.4 Filter Validation . . . 61

6 CONTROLLER SIMULATION RESULTS . . . 62
6.1 Position and Heading Tracking without Manipulator . . . 62
6.2 Full-Pose Tracking with 2-DOF Manipulator . . . 64
6.3 Discussion . . . 66
6.3.1 Position and Heading Tracking without Manipulator . . . 66
6.3.2 Full-Pose Tracking with 2-DOF Manipulator . . . 66

7 NAVIGATION SIMULATION RESULTS . . . 68
7.1 Filter Validation . . . 68
7.2 Steady State Relative State Uncertainties . . . 75
7.3 Relative State Uncertainties with Respect to Distance . . . 75
7.4 Discussion . . . 80
7.4.1 Steady State Uncertainty Discussion . . . 80
7.4.2 Relative State Uncertainties with Respect to Distance Discussion . . . 82

8 CONCLUSION AND FUTURE WORK . . . 83
8.1 Conclusion . . . 83
8.2 Future Work . . . 84

REFERENCES . . . 85

APPENDICES . . . 88
A Derivations . . . 89
A.1 Velocity and Attitude Quaternion Kinematics Linearization . . . 89
A.2 LOS Measurement Model Linearization . . . 92
B Additional Plots . . . 94
B.1 Monte Carlo Hair Plots for Architecture 2 with Medium-Grade IMU 94
B.2 Monte Carlo Hair Plots for Architecture 3 with Medium-Grade IMU 98
B.3 Monte Carlo Hair Plots for Architecture 4 with Medium-Grade IMU 102
B.4 Monte Carlo Hair Plots for Architecture 5 with Medium-Grade IMU 106
B.5 Monte Carlo Hair Plots for Architecture 6 with Medium-Grade IMU 110

B.6 Monte Carlo Hair Plots for Architecture 7 with Medium-Grade IMU 113

LIST OF TABLES

Table Page

5.1 Sensors Included in Each Architecture. 33

5.2 Beacon Positions in the Leader Body Frame. 56

5.3 1st Order Markov Parameters 57

5.4 Simulation Parameters 57

5.5 Low Grade IMU Error Model Values 58

5.6 Medium Grade IMU Error Model Values 58

5.7 High Grade IMU Error Model Values 58

7.1 North 3-Sigma Relative Position Uncertainty. 75

7.2 East 3-Sigma Relative Position Uncertainty. 75

7.3 Down 3-Sigma Relative Position Uncertainty. 75

7.4 X Axis 3-Sigma Relative Attitude Uncertainty. 76

7.5 Y Axis 3-Sigma Relative Attitude Uncertainty. 76

7.6 Z Axis 3-Sigma Relative Attitude Uncertainty. 76

LIST OF FIGURES

Figure Page

3.1 Body Frame to Manipulator 1 Frame. 11

3.2 Manipulator 1 Frame to Manipulator 2 Frame. 11

4.1 Pan-Tilt Device. 20

4.2 Tracking Controller Overview. 30

5.1 Position Vectors Geometry. 47

5.2 True Relative North Position. 55

5.3 True Relative East Position. 55

5.4 True Relative Down Position. 56

6.1 NED Position Errors 63

6.2 Angle Errors without Manipulator. 63

6.3 Angle Errors without Manipulator for First 10 Seconds and Steady State Error. 63

6.4 Distance Between Aircraft. 64

6.5 Angle Errors with Manipulator. 65

6.6 Angle Errors with Manipulator for First 10 Seconds and Steady State Error. 65

7.1 Architecture 1 Monte Carlo Analysis of Relative North Position. 69

7.2 Architecture 1 Monte Carlo Analysis of Relative East Position. 69

7.3 Architecture 1 Monte Carlo Analysis of Relative Down Position. 70

7.4 Architecture 1 Monte Carlo Analysis of Relative X Axis Attitude. 70

7.5 Architecture 1 Monte Carlo Analysis of Relative Y Axis Attitude. 71

7.6 Architecture 1 Monte Carlo Analysis of Relative Z Axis Attitude. 71

7.7 Residual and 3-sigma Residual Covariance for GPS North Measurement. . . 72

7.8 Residual and 3-sigma Residual Covariance for GPS East Measurement. . . . 73

7.9 Residual and 3-sigma Residual Covariance for GPS South Measurement. . . 73

7.10 Residual and 3-sigma Residual Covariance for LOS X Measurement for Beacon 1. 74

7.11 Residual and 3-sigma Residual Covariance for LOS Y Measurement for Beacon 1. 74

7.12 North Relative Position 3-Sigma Values Over True Distance. 77

7.13 East Relative Position 3-Sigma Values Over True Distance. 77

7.14 Down Relative Position 3-Sigma Values Over True Distance. 78

7.15 X Axis Relative Attitude 3-Sigma Values Over True Distance. 78

7.16 Y Axis Relative Attitude 3-Sigma Values Over True Distance. 79

7.17 Z Axis 3-Sigma Values Over True Distance. 79

B.1 Architecture 2 Monte Carlo Analysis of Relative North Position. 94

B.2 Architecture 2 Monte Carlo Analysis of Relative East Position. 95

B.3 Architecture 2 Monte Carlo Analysis of Relative Down Position. 95

B.4 Architecture 2 Monte Carlo Analysis of Relative X Axis Attitude. 96

B.5 Architecture 2 Monte Carlo Analysis of Relative Y Axis Attitude. 97

B.6 Architecture 2 Monte Carlo Analysis of Relative Z Axis Attitude. 97

B.7 Architecture 3 Monte Carlo Analysis of Relative North Position. 98

B.8 Architecture 3 Monte Carlo Analysis of Relative East Position. 99

B.9 Architecture 3 Monte Carlo Analysis of Relative Down Position. 99

B.10 Architecture 3 Monte Carlo Analysis of Relative X Axis Attitude. 100

B.11 Architecture 3 Monte Carlo Analysis of Relative Y Axis Attitude. 101

B.12 Architecture 3 Monte Carlo Analysis of Relative Z Axis Attitude. 101

B.13 Architecture 4 Monte Carlo Analysis of Relative North Position. 102

B.14 Architecture 4 Monte Carlo Analysis of Relative East Position. 103

B.15 Architecture 4 Monte Carlo Analysis of Relative Down Position. 103

B.16 Architecture 4 Monte Carlo Analysis of Relative X Axis Attitude. 104

B.17 Architecture 4 Monte Carlo Analysis of Relative Y Axis Attitude. 105

B.18 Architecture 4 Monte Carlo Analysis of Relative Z Axis Attitude. 105

B.19 Architecture 5 Monte Carlo Analysis of Relative North Position. 106

B.20 Architecture 5 Monte Carlo Analysis of Relative East Position. 107

B.21 Architecture 5 Monte Carlo Analysis of Relative Down Position. 107

B.22 Architecture 5 Monte Carlo Analysis of Relative X Axis Attitude. 108

B.23 Architecture 5 Monte Carlo Analysis of Relative Y Axis Attitude. 109

B.24 Architecture 5 Monte Carlo Analysis of Relative Z Axis Attitude. 109

B.25 Architecture 6 Monte Carlo Analysis of Relative North Position. 110

B.26 Architecture 6 Monte Carlo Analysis of Relative East Position. 111

B.27 Architecture 6 Monte Carlo Analysis of Relative Down Position. 111

B.28 Architecture 6 Monte Carlo Analysis of Relative X Axis Attitude. 112

B.29 Architecture 6 Monte Carlo Analysis of Relative Y Axis Attitude. 112

B.30 Architecture 6 Monte Carlo Analysis of Relative Z Axis Attitude. 113

B.31 Architecture 7 Monte Carlo Analysis of Relative North Position. 114

B.32 Architecture 7 Monte Carlo Analysis of Relative East Position. 114

B.33 Architecture 7 Monte Carlo Analysis of Relative Down Position. 115

B.34 Architecture 7 Monte Carlo Analysis of Relative X Axis Attitude. 115

B.35 Architecture 7 Monte Carlo Analysis of Relative Y Axis Attitude. 116

B.36 Architecture 7 Monte Carlo Analysis of Relative Z Axis Attitude. 116

ACRONYMS

COTS	commercial of the shelf
DOF	degrees of freedom
ECEF	earth centered earth fixed
EKF	extended Kalman filter
FOV	field of vision
GPS	global positioning system
IMU	inertial measurement unit
LOS	line of sight
LQR	linear quadratic regulator
MPC	model predictive control
PSD	power spectral density
UAV	unmanned aerial vehicle
VTOL	vertical take-off and landing

CHAPTER 1

INTRODUCTION

Interest in autonomous capabilities for Unmanned Aerial Vehicles (UAV) has been increasing significantly in recent years for both military and commercial sectors. Autonomous UAVs are successfully used in applications such as surveillance, reconnaissance, precision agriculture, and remote sensing. Academia and industry are actively researching additional autonomous UAV capabilities such as aircraft docking, autonomous refueling, package exchanges and deliveries, and tool manipulation using UAVs [1–3]. These aerial robotic capabilities have the potential to significantly expand the UAV market and solve many difficult existing problems. The capability to exchange packages between drones, for example, has the potential to increase the efficiency and practicality of drone delivery systems. This work presents control and navigation algorithms that can be used for a package exchange between two multi-rotor aircraft.

A package exchange maneuver is of great interest to the aerial robotics research community and is the aerial robotic problem this research will focus on. Delivery drones are already being used to deliver packages in various parts of the world. Airborne package delivery systems are much faster than ground based but small battery powered UAVs that are used for delivery systems suffer from significant down-time and limited range due to their limited flight time. If UAVs had the ability to exchange packages while still flying, a package relay system could be implemented to overcome these limitations. It could be even possible to exchange the UAVs battery while flying. Even though this research focuses on this specific problem of exchanging packages, the resulting navigation and control system designs could be used for other aerial robotic tasks.

Aerial tasks that involve physical interaction and manipulation are much more difficult than the typical UAV applications because of the close proximity of other objects and vehicles. These tasks require UAVs to intentionally approach other objects, greatly increasing

the possibility of a collision. Not only are the aerial robots required to not collide with other objects, but they must be able to execute precise movements to interact with an object and accomplish a task. Consequently, navigation and control systems for aerial robotics must meet strict accuracy, speed, and robustness requirements.

Multi-rotor VTOL aircraft specifically is a great candidate for an aerial platform with robotic manipulation capabilities. Their small size, affordability, and hovering capabilities make multi-rotor aircraft ideal for aerial robotic applications. For these reasons, the multi-rotor platform was chosen for this work.

1.1 Contributions

The work presented in this thesis includes the design of a full-pose synchronization controller for multi-rotor UAVs and a relative navigation architecture study. The presented algorithms guides a multi-rotor aircraft in close proximity and synchronizes its attitude and position with another aircraft to allow for a package exchange. The work does not include the development of specific higher-level guidance algorithms, hardware, and mechanical mechanisms for a package exchange, so the algorithms in this work could be used for other aircraft-to-aircraft aerial robotic tasks. Even though the objective of this research is not to solve the entire package exchange problem, the successful design of the synchronization controller and the completion of the navigation study provides a foundation of algorithms from which a package exchange system can be built.

The navigation study ensures sufficient knowledge of relative position and attitude states can be obtained for close-proximity missions. The synchronization controller minimizes changes in relative attitude and position creating a more ideal situation for robotic manipulation between multiple aircraft. Since this is the first iteration of this project, the algorithms are tested in simulation only and proximity effects are neglected. This allows the complex challenges associated with hardware implementation and proximity effect disturbances to be more fully addressed in future work.

This work is significant in that it provides design insights into relative state estimation for aircraft in close-proximity. The results will be presented and analyzed in such a way

to help the system designer determine which architecture is appropriate for their specific mission requirements and objectives. Special attention will be given to the appropriateness of the presented navigation solutions for close-proximity missions.

The contributions of this work are the following:

- The design of relative navigation solutions for seven distinct sensor architectures.
- Performance comparisons between each of the proposed architectures in the context of a close-proximity maneuver.
- A nonlinear controller design for full-pose synchronization of quad-rotor UAVs.

1.2 Overview

The rest of the paper is organized as follows. First in chapter 2, relevant literature is reviewed. Chapter 3 develops a simulation used for this research and presents other tools and conventions used in this work. Chapter 4 explains the design of a full-pose tracking controller for the synchronization of multi-rotor drones. In chapter 5, the navigation architecture study is presented for relative navigation. The study includes the development of an Extended Kalman Filter (EKF) for relative state estimation, the implementation of that EKF for seven separate navigation architectures with varying grades of IMUs, and methods for validating the estimator's results. Chapter 6 gives simulation results of the synchronization controller implementation. Chapter 7 presents and compares simulation results for each of the sensor architectures. This chapter also presents data that supports the proper functioning of the EKFs. Finally, in chapter 8 concluding remarks are given and future work is discussed.

CHAPTER 2
BACKGROUND AND LITERATURE REVIEW

To help the reader see how this works contributes to the aerial robotics field and builds upon previous work, it is necessary to review relevant and existing literature. First, literature focused specifically on aerial robotics will be discussed. Second, existing literature on control techniques for trajectory tracking will be presented. Third, background on relative navigation will be given. Fourth, vision-only navigation solutions for relative navigation will be discussed. Lastly, other navigation architectures commonly found in state-of-the-art navigation systems will be reviewed.

2.1 Aerial Robotics

Research focused on aerial robots interacting with surrounding objects and environments is an emerging topic that can drastically change how a variety of tasks are solved. Augugliaro et al [1] constructed a simple but large structure with flying robots. In [2], the researchers were able to get an autonomous multi-rotor aircraft to pick up and place objects, position pegs into small holes, and to turn a valve. Even though much of the recent research is able to consistently accomplish these tasks [1–5], the targets for the aerial robotic tasks presented in these works are stationary. An aircraft-to-aircraft robotic interaction has disturbances and other challenges that have not been addressed in previous literature.

Tasks where aerial robots interact with moving objects are comprised of aerial refueling, aerial docking, and landing on a moving platform. The research on these topics are dominated by fixed-wing platforms and drogues attached to extended cables or hoses [6–8]. A drogue is a device, typically conical or funnel-shaped with open ends, towed behind the aircraft at the end of a hose or cable. While there is a plethora of research on both fixed-wing aircraft-to-aircraft interactions using a variety of manipulators and multi-rotor aircraft-to-aircraft interactions using drogue manipulators, to the authors' knowledge, there

has not been significant work done in package exchange by two rotor aircraft using a non-drogue-based manipulator.

The type of manipulator used for aerial robotics can be split into three categories: rigid graspers, multi-degree of freedom robotic arms, and cables. A rigid grasper is a popular manipulator option due to its simplicity and light-weight nature. These systems, however, must rely solely on the maneuverability of the aircraft which is usually under-actuated [3]. Multi-degree of freedom manipulators can compensate for under-actuation and even provide higher robustness due to redundant maneuverability, but these manipulators significantly increase the complexity of hardware and control design. For manipulation, cable systems are a common solution in practice because these systems allow the aircraft to stay farther away from obstacles and other aircraft, reducing the possibility of collisions. These cable manipulator systems will often use a drogue or hook at the end of the cable. Cable systems have been successfully used for aerial refueling and object towing [7, 9]. Cable manipulators are unable to perform more complex tasks due to the lack of dexterity and can add a significant amount of weight.

2.2 Trajectory Tracking

No matter the manipulator used, it is critical for the aircraft to be able to execute a desired trajectory precisely with respect to the target object. Trajectory-tracking for autonomous vehicles is the problem where a vehicle follows a desired position and/or attitude profile which is not only parameterized spatially but also over time [10]. Following a time-parameterized trajectory is crucial for avoiding non-stationary obstacles such as other aircraft.

Trajectory-tracking for fully-actuated systems is well understood but is substantially more difficult for under-actuated systems [10]. Both linear and nonlinear methods have been proposed and researched for trajectory-tracking of under-actuated systems which include techniques such as local linearization and state decoupling, trajectory trimming, feed-forward linearization, integrator back-stepping, and careful trajectory shaping. Although several Degrees of Freedom (DOF) can be fully tracked with many of the proposed system

designs, there will be remaining DOF that can only be partially tracked by under-actuated systems for generic full-pose trajectories. A typical multi-rotor aircraft, for example, is only capable of tracking four DOF for a generic full-pose trajectory [11]. A simple 2-DOF manipulator is proposed for this research to fully-actuate the multi-rotor platform.

To achieve increased performance, state of the art trajectory tracking controllers typically implement some type of feed-forward technique. Since the applications focused on in this research involve multiple aircraft maneuvering in close-proximity, the improved performance from a feed-forward technique will be crucial to avoid collisions. [12] used a differential flatness-based feed-forward control system to achieve tracking of aggressive maneuvers.

2.3 Relative Navigation

Proposed relative navigation solutions found in existing literature can be categorized by defining characteristics such as the sensor suite, application, states estimated, and algorithm type. Of particular interest is literature that presents relative attitude estimation in addition to position estimation. The literature review will first examine vision only solutions. Afterwards, available literature that provides full-relative state estimates using additional sensors will be discussed.

2.3.1 Vision Only Navigation

Vision only navigation is an attractive solution that is frequently discussed in recent literature [13–15]. A navigation system that depends only on an on-board vision sensor does not depend on external systems, potentially making the system more robust. GPS failures due to natural phenomenon or spoofing is a common problem for autonomous systems but is not a concern for vision only systems. Other advantages are that on-board vision sensors are passive sensors and are useful for non-cooperative targets and environments. Two of the more common vision sensor types include monocular cameras and stereo cameras. Monocular cameras are simpler and cheaper but cannot measure range like the stereo camera.

Even though a monocular camera cannot measure range, if the sensor is used to make

Line of Sight (LOS) measurements to markers placed on a target aircraft and a sufficient number of those measurements are available, relative position and attitude can be estimated [15, 16]. Image processing algorithms are available to estimate states for targets without markers [13] but the computations are much larger and they don't estimate relative attitude. Another vision only estimation solution that uses a monocular camera was proposed by Tournier et al [14]. A combination of Moir patterns and red-light markers are used in the research to obtain position and attitude estimates. The red-light markers are used to create LOS of measurements to estimate the relative altitude and yaw while the Moir patterns are used to estimate the remaining position and attitude states. Even though this solution can provide highly accurate estimates, it requires more complex marker and image processing systems.

Since relative position and attitude estimation solutions using LOS measurements from a monocular camera are widely found in literature, and the sensor systems are simple and affordable, this sensor will be the only vision sensor analyzed in this paper. These systems are comprised of markers located on a target and a monocular camera located on an observer platform. The markers can be anything from light beacons to printed images. The camera sensor can range from any Commercial Off the Shelf (COTS) camera to a single position-sensing diode with a lens. The Field of Vision (FOV), sensor size, resolution, and error models can vary depending on the specific sensor type, but the measurement geometry is the same. Open source software is readily available that can process images to generate LOS measurements, making this system more accessible [17].

2.3.2 Other Relative Navigation Architectures

An attractive approach to relative navigation is to process both GPS and IMU data from each aircraft. It is well known that GPS and IMU data have complimentary bandwidths, so when combined they provide a complete and practical navigation solution. A modern concern with these systems is that GPS is susceptible to natural interference as well as intentional jamming and spoofing. Especially for close-proximity aerial missions, any of these GPS failures or performance degradation can easily cause a mid-air collision. There

are also concerns that close-proximity missions may require such high accuracy and update rates that only expensive, high-end GPS and IMU sensors will suffice.

Adding a vision sensor to the GPS/IMU navigation system adds redundancy to guard against common GPS fail points and can increase the performance of the system so that cheaper sensors can be used. A couple examples of these systems were presented by Wang et al and Williamson et al [18,19]. Including all of these sensors into the navigation system's estimator will provide the best results, but is such a system necessary? If the navigation performance is sufficient with a smaller subset of the proposed sensor suit, the system can be greatly reduced in cost, complexity, and computational load.

The final relative navigation solution architecture that is readily found in literature and that will be discussed in this review is a vision and IMU system [20, 21]. An advantage of using adding an IMU over a vision-only system is that the IMU can be used to propagate states rather than having to rely on dynamic models which for UAV's contain high levels of uncertainty. It is expected that the IMU will increase the bandwidth and accuracy of a vision-only navigation system. However, if the vision only solution is sufficient, the system again can be greatly reduced in cost, complexity, and computational load. For example, if the follower aircraft does not need GPS or IMU data from the leader, the follower could estimate the relative states without any communication with the other aircraft.

Many practical relative navigation solutions are available, but a potentially difficult question to answer is which one should be used for which application. This paper aims to formally analyze which solutions are sufficient for a close-proximity relative navigation system and provide insights on which solution is most appropriate according to specific mission requirements.

CHAPTER 3

RESEARCH TOOLS DEVELOPMENT

This chapter develops the research tools used in this research, mentions existing tools that are also used, and explains the various state variables, coordinate frames, and notation used throughout this work. First, comments on the notation is provided. Second, state variables and coordinate frames are discussed. Third, the development of a quad-rotor simulation is presented. Finally, this chapter concludes with mentioning other tools used in the work.

3.1 Notation

In an attempt to avoid confusion, this section explicitly presents the notation that will be used for vectors, Direction Cosine Matrices (DCM), estimated values, measured values, quaternions, and other conventions.

Vectors in this paper are denoted with an under-bar i.e. $\underline{\mathrm{x}}$. Vectors are assumed to be column vectors, and so row vectors will be written as a transposed vector $\underline{\mathrm{x}}^T$.

Direction cosine matrices are denoted with a capitol R. DCMs can be used to project $3x3$ matrices into alternate frames. The subscript of a DCM in this paper denotes the originating frame and the superscript denotes the new frame. For example, R_b^{NED} is a DCM that projects a vector from the body b frame to the inertial North East Down (NED) frame. Quaternions follow the same script conventions and are denoted with a lower-case q i.e. q_b^{NED}. The superscript of a vector or scalar variable show what frame the quantity is projected into. For example, $\underline{\mathrm{x}}^{NED}$ is a vector projected into the inertial NED frame.

Variables in this paper will commonly be accented with a dot symbol, a hat symbol, or a tilde i.e. $\dot{x}$, $\hat{x}$, and $\tilde{x}$ respectively. Each of these symbols represent the time derivative of a variable, an estimate of a variable, or a measurement of a variable respectively.

3.2 States and Coordinate Frames

3.2.1 Coordinate Frames

One of the most common frames used in this work is an inertial NED frame. This frame is a right-handed coordinate frame fixed at an arbitrary position in the air with the x axis pointed north, the y axis pointed east, and the z axis pointed down into the earth. Since this research only uses small UAVs that follow trajectories that cover a distance well under a kilometer, there is no need for a frame that is truly inertial such as the Earth Centered Earth Fixed Frame (ECEF).

Another common frame is the vehicle frame, which has the same orientation as the NED frame but is translated so that its origin is that a center of the corresponding aircraft. The body frame is a right-handed coordinate frame where the x axis points out of the designated front of the vehicle, z axis points down out of the belly of the aircraft, and the y axis completes the right-handed coordinate frame. There will be distinct vehicle and body frames for two separate aircraft. One aircraft will be designated the follower aircraft and the other will be designated as the leader.

The body frame orientation will often be described using an intrinsic ZYX Euler angle rotation sequence with angles ϕ, θ, and ψ. The intermediate frames created after the ψ and θ angle rotations will be referred to the vehicle 1 and 2 frames respectively. A manipulator 1 and a manipulator 2 frame will be used to keep track of the angular positions of the two joints in the manipulator. The manipulator 1 frame is obtained by rotating the body frame about the y^b axis by an angle α_m. The manipulator 2 frame is obtained by rotating the manipulator 1 frame about the x^{m1} axis by an angle β. These frames are illustrated in Fig. 3.1 and Fig. 3.2. It should be noted that the linkage distances between each manipulator actuator and between the actuators and the aircraft are ignored, so the manipulator frames are quiescent with the body frame. The manipulator 2 frame is the frame that is aligned with the final rigid arm of the manipulator.

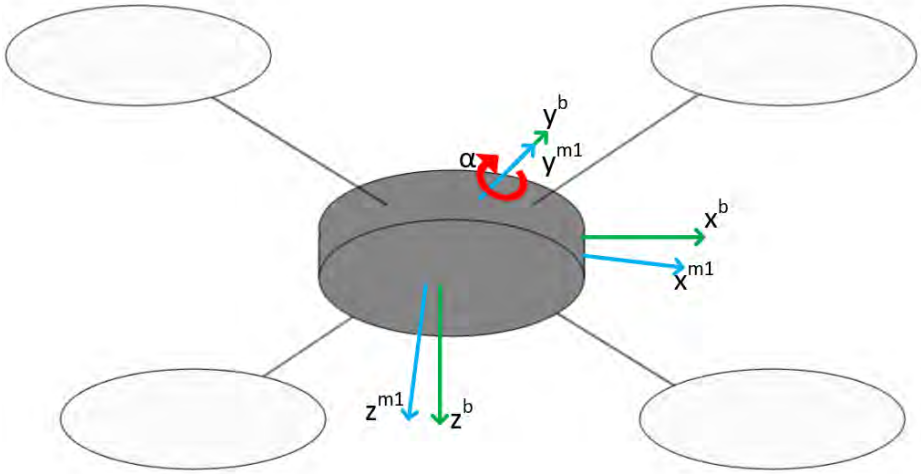

Fig. 3.1: Body Frame to Manipulator 1 Frame.

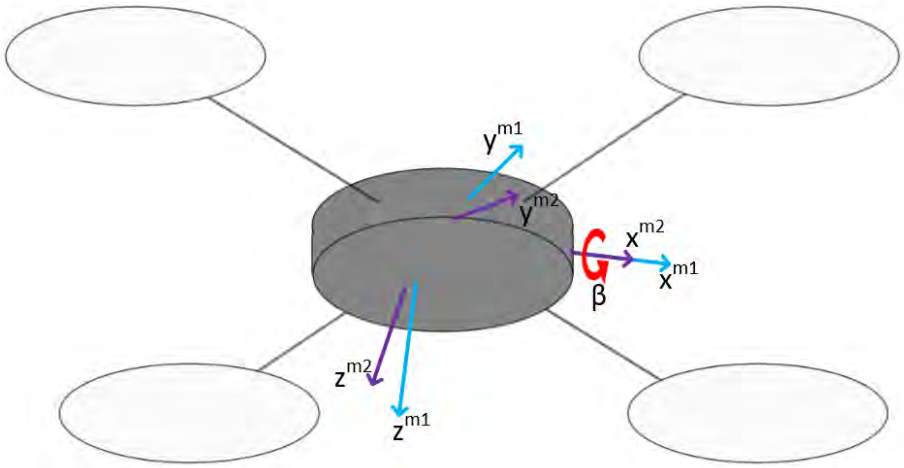

Fig. 3.2: Manipulator 1 Frame to Manipulator 2 Frame.

3.2.2 Controller State Variables

Let the state variables for a single aircraft be the following:

p_n**:** the inertial north position of the quad-rotor in the inertial frame

p_e**:** the inertial east position of the quad-rotor in the inertial frame

p_d**:** the inertial down position of the quad-rotor in the inertial frame

u**:** the body frame velocity measured along x^b in the body frame

v**:** the body frame velocity measured along y^b in the body frame

w**:** the body frame velocity measured along z^b in the body frame

ϕ**:** the roll angle defined with respect to the vehicle 2 frame

θ**:** the pitch angle defined with respect to the vehicle 1 frame

ψ**:** the yaw angle defined with respect to the vehicle frame

p**:** the roll rate measured along x^b in the body frame

q**:** the pitch rate measured along y^b in the body frame

r**:** the yaw rate measured along z^b in the body frame

α_m**:** angle of the first servo in the pan-tilt device with respect to the body frame

β**:** angle of the second servo in the pan-tilt device with respect to the manipulator 1 frame

$\dot{\alpha_m}$**:** time derivative of the angle α

$\dot{\beta}$**:** time derivative of the angle β

ϕ_m**:** the roll angle of the manipulator defined with respect to the vehicle 2 frame

θ_m**:** the pitch angle of the manipulator defined with respect to the vehicle 1 frame

ψ_m**:** the pitch angle of the manipulator defined with respect to the vehicle 1 frame

Let the inputs of the system be the following:

F**:** the upward force along z^b in the body frame due to the thrust of the propellers

τ_ϕ**:** the torque along x^b in the body frame

τ_θ**:** the torque along y^b in the body frame

τ_ψ**:** the torque along z^b in the body frame

α_c**:** the commanded servo 1 angle of the manipulator

β_c**:** the commanded servo 2 angle of the manipulator

3.2.3 Navigation State Variables

Let the state vector of the navigation system be

$$\underline{\mathrm{x}}_{nav} = \begin{bmatrix} \underline{\mathrm{p}}_L^{ned} \\ \underline{\mathrm{v}}_L^{ned} \\ q_{ned}^{b_L} \\ \underline{\mathrm{b}}_{accel,L} \\ \underline{\mathrm{b}}_{gyro,L} \\ \underline{\mathrm{p}}_F^{ned} \\ \underline{\mathrm{v}}_F^{ned} \\ q_{ned}^{b_F} \\ \underline{\mathrm{b}}_{accel,F} \\ \underline{\mathrm{b}}_{gyro,F} \end{bmatrix}$$

where

$$q_{ned}^{b} = \begin{bmatrix} \underline{\mathrm{q}} \\ q_0 \end{bmatrix}$$

$$\underline{\mathrm{q}} = \begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix}^T$$

and the L and F subscripts specify states for the leader or follower aircraft respectively. The vectors $\underline{\mathrm{p}}^{ned}$, $\underline{\mathrm{v}}^{ned}$, $\underline{\mathrm{b}}_{accel}$, $\underline{\mathrm{b}}_{gyro}$ and q^b_{ned} are the position, velocity, bias of the accelerometers, bias of the gyros, and the attitude quaternions respectively.

One of the navigation system architectures that will be analyzed will not have access to the IMU data of the leader aircraft. Without the inertial sensors used in the other architectures, it is necessary to provide a system model that is not replaced by the inertial sensors. A further explanation of these state vectors is found in chapter 5. Let the state vector for this single architecture be

$$\underline{\mathrm{x}}^{'}_{nav} = \begin{bmatrix} \underline{\mathrm{p}}^{ned}_L \\ \underline{\mathrm{v}}^{ned}_L \\ \underline{\mathrm{a}}^{ned}_L \\ q^{b_L}_{ned} \\ \underline{\omega}^b_L \\ \underline{\alpha}^b_L \\ \underline{\mathrm{p}}^{ned}_F \\ \underline{\mathrm{v}}^{ned}_F \\ q^{b_F}_{ned} \\ \underline{\mathrm{b}}_{accel,F} \\ \underline{\mathrm{b}}_{gyro,F} \end{bmatrix}$$

Where vectors $\underline{\mathrm{a}}^{ned}$, $\underline{\omega}^b$, and $\underline{\alpha}^b$, are the linear acceleration, angular velocity, and the angular acceleration of the leader aircraft.

3.3 Multi-Rotor Simulation

In this section, a simulation capable of simulating two quad-rotor aircraft and the

corresponding dynamics will be developed. The simulation is implemented using dynamic and kinematic equations from [22] which neglect ground and proximity effects.

3.3.1 Quad-rotor Kinematics

Simple rotations can be used to define the following state variables:

$$\begin{bmatrix} \dot{p_n} \\ \dot{p_e} \\ \dot{p_d} \end{bmatrix} = R_b^v \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{3.1}$$

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = R_b^{v1v2v} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{3.2}$$

where

$$R_b^v = \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix} \tag{3.3}$$

and

$$R_b^{v1v2v} = \begin{bmatrix} 1 & s\phi t\theta & c\phi t\theta \\ 0 & c\phi & -s\phi \\ 0 & s\phi sec\theta & c\phi sec\theta \end{bmatrix} \tag{3.4}$$

3.3.2 Rigid Body Dynamics

This section derives the rest of the nonlinear functions used to model the dynamics of the chosen state variables.

Use Newton's laws to find the differential equation for the velocity variables:

$$m\frac{d\underline{\mathrm{v}}}{dt_i} = \underline{\mathrm{f}} = m\left(\frac{d\underline{\mathrm{v}}}{dt_b} + \underline{\mathrm{w}}_{b/i} \times \underline{\mathrm{v}}\right) \tag{3.5}$$

.

Solve for $\frac{d\underline{\mathrm{v}}}{dt_b}$:

$$\frac{d\underline{\mathrm{v}}}{dt_b} = \frac{1}{m}\left(-\underline{\mathrm{w}}_{b/i} \times \underline{\mathrm{v}} + \underline{\mathrm{f}}\right) \tag{3.6}$$

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \begin{bmatrix} rv - qw \\ pw - ru \\ qu - pv \end{bmatrix} + \frac{1}{m}\begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix} \tag{3.7}$$

where

$$\frac{1}{m}\begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix} = \begin{bmatrix} -g\sin(\theta) \\ g\cos(\theta)\sin(\phi) \\ g\cos(\theta)\cos(\phi) - \frac{1}{m}F \end{bmatrix} \tag{3.8}$$

due to gravity and thrust. To derive the final equations, use Newton's second law:

$$m = \frac{d\underline{\mathrm{h}}^b}{dt_i} = \frac{d\underline{\mathrm{h}}}{dt_b} + \underline{\mathrm{w}}_{b/i} \times \underline{\mathrm{h}} \tag{3.9}$$

The angular momentum can be expressed as $\underline{\mathrm{h}} = J\underline{\mathrm{w}}^b_{b/i}$:

$$\underline{\mathrm{m}}^b = J\frac{d\underline{\mathrm{w}}^b_{b/i}}{dt_b} + \underline{\mathrm{w}}^b_{b/i} \times \left(J\underline{\mathrm{w}}^b_{b/i}\right) \tag{3.10}$$

where

$$J = \begin{bmatrix} J_x & -J_{xy} & -J_{xz} \\ -J_{xy} & J_y & -J_{yz} \\ -J_{xz} & -J_{yz} & J_z \end{bmatrix} \tag{3.11}$$

Solving for $\frac{d\underline{\mathrm{w}}^b_{b/i}}{dt_b}$:

$$\frac{d\underline{\mathrm{w}}^b_{b/i}}{dt_b} = J^{-1}\left\{\underline{\mathrm{m}}^b - \underline{\mathrm{w}}^b_{b/i} \times \left(J\underline{\mathrm{w}}^b_{b/i}\right)\right\} \tag{3.12}$$

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = J^{-1} \left\{ \begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} - \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \left(J \begin{bmatrix} p \\ q \\ r \end{bmatrix} \right) \right\} \tag{3.13}$$

Assuming the quad-rotor is symmetric:

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{J_y - J_z}{J_x} qr \\ \frac{J_z - J_x}{J_y} pr \\ \frac{J_x - J_y}{J_z} pq \end{bmatrix} + \begin{bmatrix} \frac{1}{J_x}\tau_\phi \\ \frac{1}{J_y}\tau_\theta \\ \frac{1}{J_z}\tau_\psi \end{bmatrix} \tag{3.14}$$

The inertia and mass values used for the simulated aircraft were chosen to match the measured values of an actual quad-rotor aircraft. The values were measured using a scale and a bifilar pendulum.

The equations presented in this section were implemented in Simulink and Matlab to simulate a quad-rotor drone as a rigid-body. The controller designed in chapter 4 will be used to control the simulated quad-rotor aircraft.

3.4 Synthesized Sensor Measurements

Flight data used for the navigation simulation is generated from the aircraft simulation given in this chapter. IMU data without noise is generated with the following equations:

$$\underline{\nu}^b = R^b_{ned} \left(\frac{\underline{\mathrm{f}}^{ned}}{m} - \mathrm{g}^{ned} \right)$$

$$\underline{\omega}^b = \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

where g^{ned} is the gravity acceleration vector in the NED frame, $\underline{\nu}^b$ is the simulated, noiseless accelerometer output vector and $\underline{\omega}^b$ is the simulated, noiseless gyro output vector.

These values will be used in chapter 5 for the navigation simulation.

CHAPTER 4

FULL-POSE TRACKING CONTROLLER

The work presented in this chapter includes the design and implementation of a differential flatness and LQR hybrid trajectory tracking controller for full-pose synchronization of two quad-rotor robots. The problem is presented as a leader and follower scenario of two quad-rotor aircraft. The leader is assumed to be flying any arbitrary trajectory while the follower attempts to synchronize its attitude and position with the leader. The leader aircraft states will be available to the follower and the follower will use those states as an input trajectory to track. A slow-moving offset will be added to the input trajectory so that that the follower can approach the leader and then interact with the leader from a desired position.

Section 4.1 presents the manipulator mathematical model, the full nonlinear state-space model, and then develops the linear control models used for controller design. Section 4.2 discusses the entire controller design. That discussion includes a high-level overview of the controller, a feed-forward component, and a linear feedback component. The chapter ends with section 4.3, which discusses specifics of the implementation and the trajectory being simulated.

4.1 Controller Model

Simplified models of the quad-rotor and 2 DOF manipulator that are used for the controller design are developed in this section. This section first develops the manipulator mathematical models. The non-linear state-space models of the entire system are then presented. Finally, the simplified linear models used for controller design are found.

4.1.1 Pan-Tilt Device Model

The quad-rotor aircraft is an under-actuated vehicle that can only track 4 DOF at a

time [11]. To compensate for this limitation, the quad-rotor system will be designed to only track the position and heading of the desired trajectory while a 2-DOF manipulator will be used to track the roll and pitch of the leader. The 2-DOF manipulator is modeled after a two-servo, pan-tilt device as shown in Fig. 4.1 where a grasper or other interaction device can be mounted.

As mentioned in chapter 3, the linkage distances between each manipulator actuator and between the manipulator actuators and the aircraft are ignored. An offset that is a function of the manipulator angles, aircraft attitude, and linkage distances will need to be applied to the aircraft position trajectory so that the end of the manipulator instead of the follower quad-rotor is tracking the full-pose trajectory. This would allow the end of the manipulator to be fully synchronized with the leader aircraft so that it can then interact safely with the leader aircraft. This manipulator offset will be left to future work, and so this work only studies the effect of the manipulator on attitude tracking.

The reaction torques on the aircraft caused by the servo torques are also neglected for this study. This is appropriate for a manipulator and payload that have a sufficiently small combined mass as well as sufficiently slow angular rates.

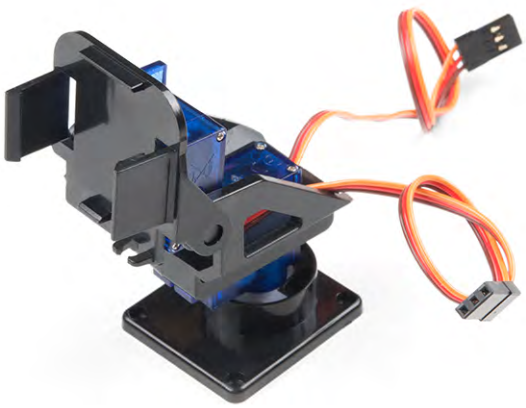

Fig. 4.1: Pan-Tilt Device.

The servos in the pan-tilt manipulator are approximated with a second order linear system model. The models are

$$\frac{\alpha}{\alpha_d} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{4.1}$$

$$\frac{\beta}{\beta_d} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{4.2}$$

where

$$\zeta = 0.707$$

$$\omega_n = 2\pi(8.5).$$

In the time domain, the dynamics of the servos can be written as

$$\dot{\alpha} = \dot{\alpha} \tag{4.3}$$

$$\ddot{\alpha} = \omega_n^2 \left(\alpha_c - \alpha\right) - 2\zeta\omega_n\dot{\alpha} \tag{4.4}$$

$$\dot{\beta} = \dot{\beta} \tag{4.5}$$

$$\ddot{\beta} = \omega_n^2 \left(\beta_c - \beta\right) - 2\zeta\omega_n\dot{\beta}. \tag{4.6}$$

To derive the attitude kinematics of the manipulator end arm, the angular rates of the manipulator arm in the body frame can be expressed as

$$\begin{bmatrix} p_m \\ q_m \\ r_m \end{bmatrix} = \begin{bmatrix} p \\ q \\ r \end{bmatrix} + \begin{bmatrix} 0 \\ \dot{\alpha} \\ 0 \end{bmatrix} + R^b_{m1} \begin{bmatrix} \dot{\beta} \\ 0 \\ 0 \end{bmatrix} \tag{4.7}$$

where p_m, q_m, and r_m are the angular rates of the manipulator in the aircraft body frame, and

$$R^b_{m1} = \begin{bmatrix} cos(\alpha) & 0 & -sin(\alpha) \\ 0 & 1 & 0 \\ -sin(\alpha) & 0 & cos(\alpha) \end{bmatrix} \tag{4.8}$$

which is the rotation matrix from the manipulator 1 frame to the body frame. The ZYX sequence Euler angles of the manipulator is then found by projecting the manipulator angular rates from the body frame to the Euler sequence frames:

$$\begin{bmatrix} \dot{\theta}_m \\ \dot{\phi}_m \\ \dot{\psi}_m \end{bmatrix} = R^{v1v2v}_b \begin{bmatrix} p_m \\ q_m \\ r_m \end{bmatrix}. \tag{4.9}$$

4.1.2 Nonlinear State Space Representation

We can now represent the entire system in the state space representation:

$$\begin{bmatrix} \dot{p_n} \\ \dot{p_e} \\ \dot{p_d} \end{bmatrix} = R^v_b \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{4.10}$$

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \begin{bmatrix} rv - qw \\ pw - ru \\ qu - pv \end{bmatrix} + \begin{bmatrix} -g\sin\theta \\ g\cos\theta\sin\phi \\ g\cos\theta\cos\phi - \frac{1}{m}F \end{bmatrix} \tag{4.11}$$

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = R_b^{v1v2v} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{4.12}$$

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{J_y - J_z}{J_x} qr \\ \frac{J_z - J_x}{J_y} pr \\ \frac{J_x - J_y}{J_z} pq \end{bmatrix} + \begin{bmatrix} \frac{1}{J_x} \tau_\phi \\ \frac{1}{J_y} \tau_\theta \\ \frac{1}{J_z} \tau_\psi \end{bmatrix} \tag{4.13}$$

$$\begin{bmatrix} \dot{\alpha} \\ \ddot{\alpha} \\ \dot{\beta} \\ \ddot{\beta} \end{bmatrix} = \begin{bmatrix} \dot{\alpha} \\ \omega_n^2 \left(\alpha_c - \alpha\right) - 2\zeta\omega_n \dot{\alpha} \\ \dot{\beta} \\ \omega_n^2 \left(\beta_c - \beta\right) - 2\zeta\omega_n \dot{\beta} \end{bmatrix} \tag{4.14}$$

$$\begin{bmatrix} \dot{\theta}_m \\ \dot{\phi}_m \\ \dot{\psi}_m \end{bmatrix} = \begin{bmatrix} p + \dot{\beta} cos(\alpha) + \dot{\alpha} sin(\phi) tan(\theta) + \left(r - \dot{\beta} sin(\alpha)\right) cos(\phi) tan(\theta) \\ \dot{\alpha} cos(\phi) - \left(r - \dot{\beta} sin(\alpha)\right) sin(\phi) \\ \dot{\alpha} sin(\phi) sec(\theta) + \left(r - \dot{\beta} sin(\alpha)\right) cos(\phi) sec(\theta) \end{bmatrix} \tag{4.15}$$

Stacking them into a single vector:

$$\dot{\underline{x}} = \begin{bmatrix} \dot{p}_n \\ \dot{p}_e \\ \dot{p}_d \\ \dot{u} \\ \dot{v} \\ \dot{w} \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ \dot{p} \\ \dot{q} \\ \dot{r} \\ \dot{\alpha} \\ \ddot{\alpha} \\ \dot{\beta} \\ \ddot{\beta} \\ \dot{\theta}_m \\ \dot{\phi}_m \\ \dot{\psi}_m \end{bmatrix} = \vec{f}(\underline{x}, \underline{u}). \tag{4.16}$$

4.1.3 Linear Model

Similar to [12], the controller being designed in this paper is an outer-loop guidance controller. It is assumed that the multi-rotor platform has a low-level attitude controller already implemented that takes thrust, roll, pitch, and yaw rate input commands (T^c, ϕ^c,θ^c, and r^c). Since the inner-loop attitude controller is typically much faster than the outer-loop guidance controller and is assumed to achieve zero error, it can be assumed that the commands to the inner-loop attitude controller are successfully tracked, i.e., $T = T^c$, $\phi = \phi^c$,$\theta = \theta^c$, and $r = r^c$. The availability of multi-rotor attitude control research and

COTS multi-rotor platforms that achieve high performance attitude control make this design method a practical and feasible method for an initial design of the outer-loop guidance controller.

With the inner-loop controller closing the loop on the aircraft attitude states, the state vector now becomes

$$\underline{\mathrm{x}} = \begin{bmatrix} p_n \\ p_e \\ p_d \\ \dot{p}_n \\ \dot{p}_e \\ \dot{p}_d \\ \psi \\ \alpha \\ \dot{\alpha} \\ \beta \\ \dot{\beta} \end{bmatrix} \tag{4.17}$$

with inputs

$$\nu = \begin{bmatrix} T_c \\ \phi_c \\ \theta_c \\ r_c \\ \alpha_c \\ \beta_c \end{bmatrix} \tag{4.18}$$

Note that the linear velocities are projected into the NED frame for the outer-loop guidance controller design. To finish the linearization of this model, a non-linear mapping function f is used to map a new input vector of accelerations

$$u = \begin{bmatrix} u_n \\ u_e \\ u_d \\ u_\psi \end{bmatrix} \tag{4.19}$$

to the inputs of the inner-loop controller $\nu_{1:4}$. The non-linear mapping is an approximation of the inverse of the true inner-loop controller and physical system mapping between the inputs $\nu_{1:4}$ and the true accelerations, i.e., $\ddot{p_n}$, $\ddot{p_e}$, $\ddot{p_d}$, and $\ddot{\psi}$. Since the model for the servo dynamics is already linear, there is no need to map the corresponding input commands. The non-linear mapping is defined as

$$f = \begin{bmatrix} f_p \\ f_\psi \end{bmatrix} = \begin{bmatrix} R_b^{ned} \begin{bmatrix} 0 \\ 0 \\ -T \end{bmatrix} \frac{1}{m} \\ q\frac{sin\phi}{cos\theta} + r\frac{cos\phi}{cos\theta} \end{bmatrix} \tag{4.20}$$

.

The state-space model of the final system is given as

$$\underline{\mathrm{x}} = \begin{bmatrix} p_n \\ p_e \\ p_d \\ \dot{p}_n \\ \dot{p}_e \\ \dot{p}_d \\ \psi \\ \alpha \\ \dot{\alpha} \\ \beta \\ \dot{\beta} \end{bmatrix} \tag{4.21}$$

$$u = \begin{bmatrix} u_n \\ u_e \\ u_d \\ u_\psi \\ \alpha_c \\ \beta_c \end{bmatrix} \tag{4.22}$$

$$A = \begin{bmatrix} 0_{3x3} & I_{3x3} & 0_{3x1} & 0_{3x1} & 0_{3x1} & 0_{3x1} & 0_{3x1} \\ 0_{3x3} & 0_{3x3} & 0_{3x1} & 0_{3x1} & 0_{3x1} & 0_{3x1} & 0_{3x1} \\ 0_{1x3} & 0_{1x3} & 0 & 0 & 0 & 0 & 0 \\ 0_{1x3} & 0_{1x3} & 0 & 0 & 1 & 0 & 0 \\ 0_{1x3} & 0_{1x3} & 0 & -\omega_n^2 & -2\zeta\omega_n & 0 & 0 \\ 0_{1x3} & 0_{1x3} & 0 & 0 & 0 & 0 & 1 \\ 0_{1x3} & 0_{1x3} & 0 & 0 & 0 & -\omega_n^2 & -2\zeta\omega_n \end{bmatrix} \tag{4.23}$$

$$B = \begin{bmatrix} 0_{3x3} & 0_{3x1} & 0_{3x1} & 0_{3x1} \\ I_{3x3} & 0_{3x1} & 0_{3x1} & 0_{3x1} \\ 0_{1x3} & 1 & 0 & 0 \\ 0_{1x3} & 0 & 0 & 0 \\ 0_{1x3} & 0 & \omega_n^2 & 0 \\ 0_{1x3} & 0 & 0 & 0 \\ 0_{1x3} & 0 & 0 & \omega_n^2 \end{bmatrix} \tag{4.24}$$

4.2 Controller Design

An LQR feedback controller is designed to reject disturbances and achieve a desired performance for the tracking of the position and heading states. Differential flatness is used to generate a feed-forward input to increase the speed of the trajectory tracking [12]. Since the model provided for the servos is a model of the servo motors with an internal controller, a controller will not be designed for the servo angles. The inputs to the servo systems will

be defined as

$$\alpha_c = \theta^{ref} - \theta_m \tag{4.25}$$

$$\beta = \phi^{ref} - \phi_m \tag{4.26}$$

where the reference angles are the angles of the leader aircraft.

4.2.1 Differential Flatness

Differential flatness of a system is where the state and control inputs can be expressed as functions of the output and its time derivatives [23, 24]. This property permits the generation of feed-forward inputs given a trajectory. Let a given trajectory be defined as

$$y_{traj} = \begin{bmatrix} p_n^{ref} \\ p_e^{ref} \\ p_d^{ref} \\ \psi^{ref} \end{bmatrix} \tag{4.27}$$

where the reference states are the states of the leader aircraft plus a position offset. Due to differential flatness, a reference input can be computed:

$$u^{ref} = \begin{bmatrix} u_n^{ref} \\ u_e^{ref} \\ u_d^{ref} \\ u_\psi^{ref} \end{bmatrix} = \begin{bmatrix} \ddot{p}_n^{ref} \\ \ddot{p}_e^{ref} \\ \ddot{p}_d^{ref} - g \\ \dot{\psi}^{ref} \end{bmatrix} \tag{4.28}$$

Let the reference states be

$$\underline{x}^{ref} = \begin{bmatrix} p_n^{ref} \\ p_e^{ref} \\ p_d^{ref} \\ \dot{p}_n^{ref} \\ \dot{p}_e^{ref} \\ \dot{p}_d^{ref} \\ \psi^{ref} \end{bmatrix} \tag{4.29}$$

.

4.2.2 LQR

The LQR controller is designed using the linear model derived in section 4.1.3. The Q and R gains were originally weighted using Bryson's rule from [25]. The Q and R gains were selected to be

$$Q = \begin{bmatrix} I_{7x7} \end{bmatrix} \tag{4.30}$$

$$R = \begin{bmatrix} 0.4 & 0 & 0 & 0 \\ 0 & 0.8 & 0 & 0 \\ 0 & 0 & 0.8 & 0 \\ 0 & 0 & 0 & 10 \end{bmatrix} \tag{4.31}$$

.

The final controller and system are illustrated in Fig. 4.2.

4.3 Implementation and Simulation

To test the tracking controller performance, the controller designed in the previous section is implemented in Simulink and Matlab with the simulation developed in chapter 3. The leader aircraft is first simulated following a constant-altitude, circular orbit with a radius of 12 meters and a rate of 30 seconds per orbit. The leader is initialized to a point

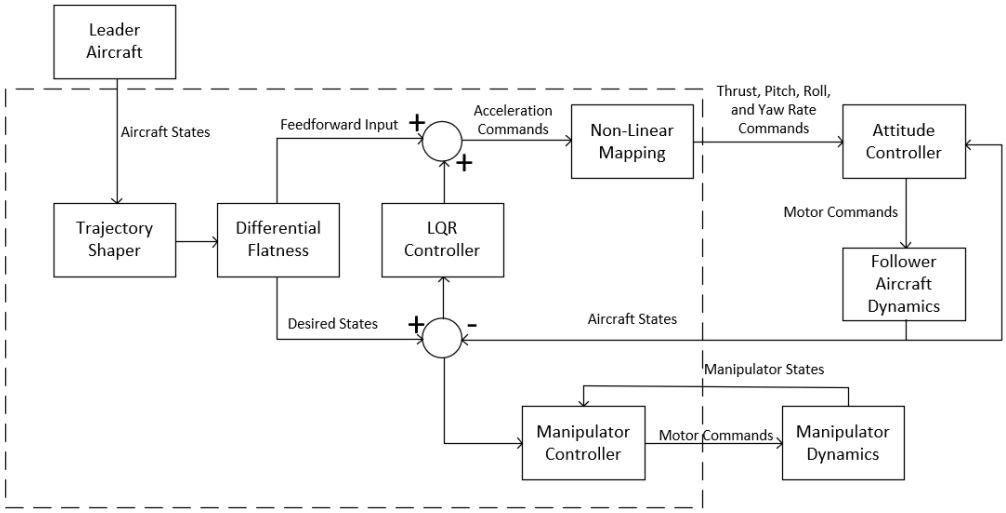

Fig. 4.2: Tracking Controller Overview.

holding condition, and then follows the orbit trajectory. The follower aircraft uses the tracking controller designed in the previous section to synchronize with the leader aircraft's position and attitude. An offset is applied in the up direction so that that the follower aircraft is above the leader. The offset starts at 10 meters in the up direction and then ramps down to a 2-meter offset.

The simulation is first ran without the manipulator implemented. The follower aircraft is only commanded to track the leader's position and heading. Since both the aircraft are similar to each other, it is expected that the attitude difference between the two aircraft will not be extremely large. The simulation is ran again with the manipulator implemented onto the follower aircraft. With the manipulator working to track the roll and pitch of the leader aircraft, the differences in attitude between the two aircraft is expected to decrease.

The plots provided in chapter 6 will include the error between the follower aircraft's actual position and attitude and the commanded trajectory. The commanded trajectory is generated by adding the position offset mentioned previously to the true states of the leader aircraft.

CHAPTER 5

NAVIGATION ARCHITECTURE STUDY

The principal objective of the navigation architecture study described in this chapter is to compare the state estimation performance of different navigation architectures with emphasis on a UAV close-proximity scenario. To work towards this objective, the study identifies seven architectures to study, designs a navigation system for each architecture, simulates each filter over identical trajectories, and then compares performance. A trajectory is generated that keeps a designated leader aircraft approximately stationary while the other aircraft, designated the follower aircraft, executes a maneuver to approach the leader aircraft from above. The leader aircraft is not completely stationary because it is subject to disturbances. The follower aircraft is the platform where the relative state estimation is implemented.

The first section of this chapter explains each of the seven architectures. The second section gives an overview of an EKF algorithm which is the navigation algorithm used for each architecture. The following section presents the system models used for the propagation step of the EKF. Afterwards, the measurement models for sensors used in the architectures are given. Section 5.5 then explains the Kalman update and state estimate correction steps. The concluding section of this chapter explains the methods used to validate the proper functioning of the navigation filters and provides the methods used to generate the final results of the study.

5.1 Study Structure

Each of the seven architectures comprise of different combinations of GPS, IMU, and camera sensors. The camera sensors are used to generate LOS measurements to multiple beacons located on the leader aircraft. These sensors were chosen because they are common sensors found on UAVs and in similar research.

The first architecture employs a centralized Kalman filter, with maximal information sharing. It is assumed that each UAV is equipped with a GPS and IMU, and a communication link exists between the UAVs. In addition, the follower vehicle contains a monocular camera. An EKF onboard the follower utilizes the IMUs to propagate the state of both the leader and follower. For updating the states of each vehicle, both GPS measurements as well as LOS measurements from the camera are processed.

The second architecture considered during this study is similar to the first, but with the absence of LOS measurements. This is intended to quantify the benefit of LOS measurements for relative state estimation. The third and fourth architectures are also similar to the first, but where the GPS measurements of the leader and the follower, respectively, are deactivated. Note that LOS measurements are still active. This architecture is applicable to partially-denied GPS scenarios. The fifth architecture deactivates GPS on both UAVs, and is intended to determine the feasibility of relative state estimation in a fully-GPS-denied scenario.

The sixth architecture considered in this study assumes no information sharing between UAVs. The follower UAV is equipped with an IMU and GPS. The dynamics of the leader are modeled with a first order Markov process, and the corresponding state is estimated via LOS measurements. There is no information about the leader UAV states between LOS measurements, and so the first order Markov processes are used to propagate the state uncertainty limited to an appropriate strength and bandwidth.

The final architecture is the same as the sixth, but GPS on the follower aircraft is deactivated. This architecture assumes a fully GPS denied environment and no information sharing between UAVs. The sensors used in each architecture are summarized in table 5.1

5.2 Kalman Filter Overview

An Extended Kalman Filter (EKF) for each of the sensor architectures is derived to estimate relative position, relative velocity, relative attitude, and sensor biases. Major steps of designing the EKFs include developing navigation models to propagate the estimated states, defining the process and sensor noise, and defining the measurement models to update the

Table 5.1: Sensors Included in Each Architecture.

Architecture	Leader IMU	Leader GPS	Follower IMU	Follower GPS	LOS
1	X	X	X	X	X
2	X	X	X	X	
3	X		X	X	X
4	X	X	X		X
5	X		X		X
6			X	X	X
7			X		X

states and covariance. The process noise, sensor noise, and time constant parameters are selected using sensor datasheets, existing research, and empirical data.

Each EKF is a an indirect Kalman filter with continuous dynamics and discrete measurements. The vehicle dynamics models for the majority of the Kalman filters are replaced with IMU sensor measurement models. The indirect EKF algorithm is provided in algorithm 5.1.

5.3 System Models

This section provides the system models used to propagate the state vector and state covariance matrix between update steps. First the functions $\underline{\mathrm{f}}$ used to calculate $\underline{\dot{\hat{x}}}^{-}$ in algorithm 5.1 are presented. Linearized error models are then derived which are denoted as F in algorithm 5.1.

5.3.1 Full System Models

In this subsection, the functions $\underline{\mathrm{f}}$ used to calculate $\underline{\dot{\hat{x}}}^{-}$ in algorithm 5.1 are presented. Two versions of these state dynamics functions are provided: one uses IMU measurement model replacement and the other uses first order Markov processes.

Model Replacement

As chapter 3 states, the state vector of this navigation system is

Algorithm 5.1 Indirect EKF

Input:

Initial state vector $\underline{\mathrm{x}}_0$,
Initial state covariance matrix P_0,
Measurement vector $\underline{\tilde{\mathrm{z}}}$,

Output:

Estimated state vector, $\underline{\hat{x}}$
State covariance matrix, P

Begin

$\dot{\underline{\hat{x}}}^- = \underline{\mathrm{f}}(\underline{\hat{\mathrm{x}}}, \underline{\mathrm{w}})$
$\dot{P}^- = FP^- + P^-F^T + Q$
$\underline{\hat{x}}^- = \text{FourthOrderRunge-Kutta}(\dot{\underline{\hat{x}}}^-)$
$P^- = \text{FourthOrderRunge-Kutta}(\dot{P}^-)$

If Measurement Available

$K = P^-H^T(HP^-H^T + R)^{-1}$
$\delta x^+ = K(\tilde{z} - h(\hat{x}))$
$P^+ = (I - KH)P^-(I - KH)^T + KRK^T$
$\hat{x}^+ = g(\hat{x}^-, \delta)x^+$

End

$$\underline{\mathrm{x}}_{nav} = \begin{bmatrix} \underline{\mathrm{p}}_L^{ned} \\ \underline{\mathrm{v}}_L^{ned} \\ q_{ned}^{b_L} \\ \underline{\mathrm{b}}_{accel,L} \\ \underline{\mathrm{b}}_{gyro,L} \\ \underline{\mathrm{p}}_F^{ned} \\ \underline{\mathrm{v}}_F^{ned} \\ q_{ned}^{b_F} \\ \underline{\mathrm{b}}_{accel,F} \\ \underline{\mathrm{b}}_{gyro,F} \end{bmatrix}$$

where

$$q^b_{ned} = \begin{bmatrix} \underline{\mathrm{q}} \\ q_0 \end{bmatrix}$$

$$\underline{\mathrm{q}} = \begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix}^T$$

and the L and F scripts specify the leader or follower aircraft respectively. The position and velocity states, $\underline{\mathrm{p}}$ and $\underline{\mathrm{v}}$, are all projected into the North East Down (NED) frame.

Since the dynamics of each aircraft and the corresponding inertial sensors are defined by the same equations, the dynamics of a single arbitrary aircraft will be presented. The dynamics of the state vector for a single aircraft are

$$\dot{\underline{x}} = \underline{\mathrm{f}}(\underline{\mathrm{x}}, \tilde{\underline{\mathrm{y}}}, \underline{\mathrm{w}})$$

where

$$\dot{\underline{p}} = \underline{\mathrm{v}}$$

$$\dot{\underline{\mathrm{v}}} = R^{ned}_b(\tilde{\underline{\nu}} - \underline{\mathrm{b}}_{accel} + \underline{\mathrm{n}}_\nu) + \underline{\mathrm{g}}^{ned}$$

$$\dot{q}^b_{ned} = \frac{1}{2} \begin{bmatrix} \tilde{\underline{\omega}}^b - \underline{\mathrm{b}}_{gyro} + \underline{\mathrm{n}}_\omega \\ 0 \end{bmatrix} \otimes q^b_{ned}$$

$$\dot{\underline{b}}_{accel} = -\frac{1}{\tau_{accel}} \underline{\mathrm{b}}_{accel} + \underline{\mathrm{w}}_{accel}$$

$$\dot{\underline{b}}_{gyro} = -\frac{1}{\tau_{gyro}}\underline{\mathrm{b}}_{gyro} + \underline{\mathrm{w}}_{gyro}$$

and

g^{ned} : acceleration due to gravity

τ_{accel} : 1st order Markov process time constant of the accelerometers

τ_{gyro} : 1st order Markov process time constant of the gyros.

The inertial measurement vector is defined as

$$\underline{\tilde{\mathrm{y}}} = \begin{bmatrix} \underline{\tilde{\nu}}^b \\ \underline{\tilde{\omega}}^b \end{bmatrix} = \begin{bmatrix} \underline{\nu}^b + \underline{\mathrm{b}}_{accel} \\ \underline{\omega}^b_{b/ned} + \underline{\mathrm{b}}_{gyro} \end{bmatrix}$$

where $\underline{\nu}^b$ is the true specific force measured by the accelerometers and $\underline{\omega}^b$ is the true angular rate measured by the gyros.

The process noise vector is defined as

$$\underline{\mathrm{w}} = \begin{bmatrix} \underline{\mathrm{n}}_{\nu} \\ \underline{\mathrm{n}}_{\omega} \\ \underline{\mathrm{w}}_{accel} \\ \underline{\mathrm{w}}_{gyro} \end{bmatrix}$$

whose power spectral density is defined as

$$E\left[\underline{\mathrm{w}}(t)\underline{\mathrm{w}}(t')^T\right] = \hat{Q}_w\delta(t-t')$$

where

$$\hat{Q}_w = diag\left(Q_\nu, Q_\omega, \frac{2\sigma^2_{accel,ss}}{\tau_{accel}}I_{3x3}, \frac{2\sigma^2_{gyro,ss}}{\tau_{gyro}}I_{3x3}\right)$$

and

$\sigma^2_{accel,ss}$: steady state covariance of accelerometer bias

$\sigma^2_{gyro,ss}$: steady state covariance of gyro bias.

System Without Model Replacement

Two of the navigation system architectures that will be analyzed cannot use IMU measurement model replacement for the propagation of the leader states. The follower aircraft will still have an IMU but the leader IMU data will not be available. Without the inertial sensors used in the other architectures, it is necessary to provide a system model that is not replaced by the inertial sensors. Let the state vector of the system be

$$\underline{\mathrm{x}}' = \begin{bmatrix} \underline{\mathrm{p}}_L^{ned} \\ \underline{\mathrm{v}}_L^{ned} \\ \underline{\mathrm{a}}_L^{ned} \\ q_{ned}^{b_L} \\ \underline{\omega}_L^b \\ \underline{\alpha}_L^b \\ \underline{\mathrm{p}}_F^{ned} \\ \underline{\mathrm{v}}_F^{ned} \\ q_{ned}^{b_F} \\ \underline{\mathrm{b}}_{accel,F} \\ \underline{\mathrm{b}}_{gyro,F} \end{bmatrix}$$

The dynamics of the translational and rotational accelerations of the leader aircraft will be estimated by a 1st order Markov process. The dynamics are

$$\dot{\underline{p}}_L^{ned} = \underline{\mathrm{v}}_L^{ned}$$

$$\dot{\underline{v}}_L^{ned} = \underline{\mathrm{a}}_L^{ned}$$

$$\dot{\underline{\mathrm{a}}}_L^{ned} = -\frac{1}{\tau_a}\underline{\mathrm{a}}_L^{ned} + \underline{\mathrm{w}}_{a,L}$$

$$\dot{q}_{ned}^{b_L} = \frac{1}{2}\begin{bmatrix} \underline{\omega}_L^b \\ 0 \end{bmatrix} \otimes q_{ned}^{b_L}$$

$$\dot{\underline{\omega}}_L^b = \underline{\alpha}_L^b$$

$$\dot{\underline{\alpha}}_L^b = -\frac{1}{\tau_\alpha}\underline{\alpha}_L^b + \underline{\mathrm{w}}_{\alpha,L}$$

where

τ_a : 1st order Markov process time constant of the linear acceleration

τ_α : 1st order Markov process time constant of the angular acceleration.

The process noise vector is defined as

$$\underline{\mathrm{w}} = \begin{bmatrix} \underline{\mathrm{w}}_{a,L} \\ \underline{\mathrm{w}}_{\alpha,L} \end{bmatrix}$$

whose power spectral density is defined as

$$E\left[\underline{\mathrm{w}}(t)\underline{\mathrm{w}}(t')^T\right] = \hat{Q}_w\delta(t-t')$$

where

$$\hat{Q}_w = diag\left(\frac{2\sigma_{a,ss}^2}{\tau_a} I_{3x3}, \frac{2\sigma_{\alpha,ss}^2}{\tau_\alpha} I_{3x3}\right)$$

and

$\sigma_{a,ss}^2$: steady state covariance of linear acceleration

$\sigma_{\alpha,ss}^2$: steady state covariance of angular acceleration.

5.3.2 Linear Error Models

The linear estimation error models are explicitly defined in this section. The estimation errors are defined as small perturbation from a nominal value. Since the best estimates will be used as the nominal values in the EKF, the nominal value will be denoted with the same notation as a state estimate.

$$\underline{\mathrm{p}}^{ned} = \underline{\hat{p}}^{ned} + \delta\underline{\mathrm{p}}^{ned}$$

$$\underline{\mathrm{v}}^{ned} = \underline{\hat{v}}^{ned} + \delta\underline{\mathrm{v}}^{ned}$$

$$\underline{\mathrm{q}}_{ned}^{b} = \delta\underline{\mathrm{q}}_{\hat{b}}^{b} \otimes \underline{\mathrm{q}}_{ned}^{\hat{b}}$$

$$\underline{\mathrm{b}}_{accel} = \underline{\hat{b}}_{accel} + \delta\underline{\mathrm{b}}_{accel}$$

$$\underline{\mathrm{b}}_{gyro} = \underline{\hat{b}}_{gyro} + \delta\underline{\mathrm{b}}_{gyro}$$

where the attitude estimation error is defined as

$$\delta q_{\hat{b}}^{b} = \begin{bmatrix} \frac{\delta \underline{\theta}_b}{2} \\ 1 \end{bmatrix}.$$

The attitude estimation error can also be defined in terms of the DCM:

$$R_{ned}^{b} = R_{\hat{b}}^{b} R_{ned}^{\hat{b}}$$

$$R_{b}^{ned} = R_{\hat{b}}^{ned} \left(R_{\hat{b}}^{b} \right)^{T}$$

$$R_{\hat{b}}^{b} = \left[I - (\delta \underline{\theta}_b \times) \right].$$

Substituting these defined error models into the original design model equations yields

$$\dot{\underline{p}}^{ned} + \delta \dot{\underline{p}}^{ned} = \underline{\hat{v}}^{ned} + \delta \underline{\mathrm{v}}^{ned} \tag{5.1}$$

$$\dot{\underline{\hat{v}}}^{ned} + \delta \dot{\underline{v}}^{ned} = R_{b}^{ned} \left([I - (\delta \underline{\theta}_b \times)] \right)^{T} \left(\underline{\tilde{\nu}}^{b} - \underline{\hat{b}}_{accel} - \delta \underline{\mathrm{b}}_{accel} + \underline{\mathrm{n}}_{\nu} \right) + \mathrm{g}^{ned} \tag{5.2}$$

$$\frac{d}{dt} \left(\delta q_{\hat{b}}^{b} \otimes q_{ned}^{\hat{b}} \right) = \delta \dot{q}_{\hat{b}}^{b} \otimes q_{ned}^{\hat{b}} + \delta q_{\hat{b}}^{b} \otimes \dot{q}_{ned}^{\hat{b}} \tag{5.3}$$

$$\dot{\underline{\hat{b}}}_{accel} + \delta \dot{\underline{b}}_{accel} = -\frac{1}{\tau_{accel}} \left(\underline{\hat{b}}_{acce5l} + \delta \underline{\mathrm{b}}_{accel} \right) + \underline{\mathrm{w}}_{accel} \tag{5.4}$$

$$\dot{\underline{\hat{b}}}_{gyro} + \delta \dot{\underline{b}}_{gyro} = -\frac{1}{\tau_{gyro}} \left(\underline{\hat{b}}_{gyro} + \delta \underline{\mathrm{b}}_{gyro} \right) + \underline{\mathrm{w}}_{gyro}. \tag{5.5}$$

Solving for the derivatives of the perturbations and eliminating like and higher order terms, the final error state vector differential equations are derived:

$$\delta\dot{\underline{p}}^{ned} = \delta\underline{\mathrm{v}}^{ned} \tag{5.6}$$

$$\delta\dot{\underline{\nu}}^{ned} = -R_b^{ned}(\tilde{\underline{\nu}}^b - \delta\underline{\mathrm{b}}_{accel}) \times \delta\underline{\theta}_b - R_b^{ned}\delta\underline{\mathrm{b}}_{accel} + R_b^{ned}\underline{\mathrm{n}}_{\nu} \tag{5.7}$$

$$\delta\dot{\underline{\theta}}_b = -\left(\tilde{\underline{\omega}} - \hat{\underline{b}}_{gyro}\right) \times \delta\underline{\theta}_b - \delta\underline{\mathrm{b}}_{gyro} + \underline{\mathrm{n}}_{\omega} \tag{5.8}$$

$$\delta\dot{\underline{b}}_{accel} = -\frac{1}{\tau_{accel}}\delta\underline{\mathrm{b}}_{accel} + \underline{\mathrm{w}}_{accel} \tag{5.9}$$

$$\delta\dot{\underline{b}}_{gyro} = -\frac{1}{\tau_{gyro}}\delta\underline{\mathrm{b}}_{gyro} + \underline{\mathrm{w}}_{gyro}. \tag{5.10}$$

The full derivations for the linear velocity and attitude error differential equations can be found in appendix A.1. The differential equations can be put into the state space form

$$\delta\dot{\underline{x}} = \hat{F}\delta\underline{\mathrm{x}} + \hat{B}\underline{\mathrm{w}} \tag{5.11}$$

where

$$\delta\underline{\mathrm{x}} = \begin{bmatrix} \delta\underline{\mathrm{p}}^{ned} \\ \delta\underline{\mathrm{v}}^{ned} \\ \delta\underline{\theta}_b \\ \delta\underline{\mathrm{b}}_{accel} \\ \delta\underline{\mathrm{b}}_{gyro} \end{bmatrix}$$

$$\hat{F} = \begin{bmatrix} 0_{3x3} & I_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & -\hat{R}_b^{ned}(\tilde{\underline{\nu}}^b - \hat{\underline{b}}_{accel})\times & -\hat{R}_b^{ned} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & -\left(\tilde{\underline{\omega}} - \hat{\underline{b}}_{gyro}\right)\times & 0_{3x3} & -I_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & -\frac{1}{\tau_{accel}} I_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & -\frac{1}{\tau_{gyro}} I_{3x3} \end{bmatrix}$$

$$\hat{B} = \begin{bmatrix} 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & R_b^{ned} \\ 0_{3x3} & I_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & I_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & I_{3x3} \end{bmatrix}.$$

The linear system presented is used to derive the covariance propagation

$$\dot{\hat{P}} = \hat{F}\hat{P} + \hat{P}\hat{F}^T + \hat{B}\hat{S}_{\dot{\omega}}\hat{B}^T. \tag{5.12}$$

Following a similar pattern for the non-IMU replacement models, the linear differential equations are found to be

$$\delta\dot{\underline{p}}_L^{ned} = \delta\underline{\mathrm{v}}_L^{ned} \tag{5.13}$$

$$\delta\dot{\underline{v}}_L^{ned} = \underline{\delta\mathrm{a}}_L^{ned} \tag{5.14}$$

$$\delta\dot{\underline{\mathrm{a}}}_L^{ned} = -A_a\delta\underline{\mathrm{a}}_L^{ned} + \underline{\mathrm{w}}_{a,L} \tag{5.15}$$

$$\delta\dot{\underline{\theta}}_b = -\left(\hat{\underline{\omega^b}}\right)\times\delta\underline{\theta}_b + \delta\underline{\dot{\omega}}_L^b \tag{5.16}$$

$$\delta\dot{\underline{\omega}}_L^b = \delta\underline{\alpha}_L^b \tag{5.17}$$

$$\delta\dot{\underline{\alpha}}_L^b = -A_\alpha \delta\underline{\alpha}_L^b + \underline{\mathrm{w}}_{\alpha,L}. \tag{5.18}$$

The differential equations can be put into the state space form

$$\delta\dot{\underline{x}} = \hat{F}\delta\underline{\mathrm{x}} + \hat{B}\underline{\mathrm{w}} \tag{5.19}$$

where

$$\delta\underline{\mathrm{x}} = \begin{bmatrix} \delta\underline{\mathrm{p}}^{ned} \\ \delta\underline{\mathrm{v}}^{ned} \\ \delta\underline{\mathrm{a}}^{ned} \\ \delta\underline{\theta}_b \\ \delta\dot{\underline{\omega}}_L^b \\ \delta\dot{\underline{\alpha}}_L^b \end{bmatrix}$$

$$\hat{F} = \begin{bmatrix} 0_{3x3} & I_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & I_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & -A_\alpha & 0_{3x3} & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & -(\hat{\underline{\omega}})\times & I_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & I_{3x3} \\ 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x3} & -A_\alpha \end{bmatrix}$$

$$\hat{B} = \begin{bmatrix} 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} \\ I_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} \\ 0_{3x3} & 0_{3x3} \\ 0_{3x3} & I_{3x3} \end{bmatrix}.$$

5.4 Measurement Models

In this section, the measurement models for both the GPS and LOS measurements are developed. The models are then linearized to form the measurement geometry matrices that are used in the Kalman filters. The models are determined in discrete time. The GPS model is derived for the states of a single aircraft. The complete models for each sensor architecture are then presented.

5.4.1 GPS Position Measurements

The model that will be used for the GPS measurements is a position measurement model with added noise. The discrete-time noise is assumed to be white, Gaussian, and zero-mean. The model for this measurement is

$$\underline{z}_{gps}(t_i) = \underline{p}^{ned}(t_i) + \underline{\eta}_{gps}(t_i). \tag{5.20}$$

where $\underline{\eta}_{gps}$ is a 3-element column vector with strength

$$E\{\underline{\eta}_{gps}(t_i)\underline{\eta}_{gps}(t_i)\} = R_{gps}(t_i).$$

Since all of the following analysis for the measurement model is done at a single moment in time, the t_i notation is dropped. The system is already linear, and so the linear state space error model and corresponding measurement matrix is found to be

$$\delta \underline{z}_{gps} = H_{gps} \delta \underline{x} + \underline{\eta}_{gps} \tag{5.21}$$

where

$$H_{gps} = \begin{bmatrix} I_{3x3} & 0_{3x12} \end{bmatrix} \tag{5.22}$$

$$\delta \underline{x} = \begin{bmatrix} \delta \underline{p}^{ned} \\ \delta \underline{v}^{ned} \\ \delta \underline{\theta}_b \\ \delta \underline{b}_{accel} \\ \delta \underline{b}_{gyro} \end{bmatrix}.$$

5.4.2 LOS Measurement Model

A focal plane measurement model with added noise is used for the LOS measurements. The discrete-time noise is again assumed to be white, Gaussian. and zero-mean. The measurement model is

$$\underline{z}_{LOS}(t_i, j) = \underline{h}(t_i, \underline{x}, j) + \underline{\eta}_{LOS}(t_i) = \begin{bmatrix} \frac{l_x(t_i, \underline{X}, j)}{l_z(t_i, \underline{X}, j)} \\ \frac{l_y(t_i, \underline{X}, j)}{l_z(t_i, \underline{X}, j)} \end{bmatrix} + \underline{\eta}_{LOS}(t_i) \tag{5.23}$$

where $\underline{\eta}_{LOS}$ is a 2-element column vector with strength

$$E\{\underline{\eta}_{LOS}(t_i) \underline{\eta}_{LOS}(t_i)\} = R_{LOS}(t_i)$$

and $\underline{l}(j)$ is the position vector of the jth marker located on the leader aircraft projected to the body frame of the follower aircraft:

$$\underline{\mathrm{l}}(t_i, \underline{\mathrm{x}}, j) = \begin{bmatrix} l_x(t_i, \underline{\mathrm{x}}, j) \\ l_y(t_i, \underline{\mathrm{x}}, j) \\ l_z(t_i, \underline{\mathrm{x}}, j) \end{bmatrix}.$$

Again, all of the following analysis for the measurement model is done at a single moment in time, so the t_i notation is dropped. Since the following analysis is performed for the measurement of a single marker, much of the j notation will also be dropped. The nonlinear function h is also a function of the marker position vector, which allows the measurement model to be written as

$$\underline{\mathrm{z}}_{LOS} = \underline{\mathrm{h}}(\underline{\mathrm{l}}(\underline{\mathrm{x}})) + \underline{\eta}_{LOS} \tag{5.24}$$

Let the z and l vectors be equal to nominal values plus a perturbation. The nominal value is chosen to be the best estimate of the Kalman filters:

$$\underline{\mathrm{z}}_{LOS} = \underline{\hat{z}}_{LOS} + \delta\underline{\mathrm{z}}_{LOS} \tag{5.25}$$

$$\underline{\mathrm{l}} = \underline{\hat{\mathrm{l}}} + \delta\underline{\mathrm{l}} \tag{5.26}$$

where

$$\underline{\hat{z}}_{LOS} = \underline{\mathrm{h}}(\underline{\hat{\mathrm{l}}}). \tag{5.27}$$

Substituting the perturbation definitions into the measurement model and using a Taylor Series expansion, we obtain

$$\underline{\hat{z}}_{LOS} + \delta\underline{\mathrm{z}}_{LOS} = \underline{\mathrm{h}}(\underline{\hat{\mathrm{l}}}) + \left.\frac{\delta\underline{\mathrm{h}}}{\delta\underline{\mathrm{l}}}\right|_{\underline{\hat{\mathrm{l}}}} \delta l + \underline{\eta}_{LOS} + ... \tag{5.28}$$

The higher order terms of the expansion are discarded and the nominal measurements are canceled to find the linear LOS measurement model:

$$\delta \underline{z}_{LOS} = \left.\frac{\delta \underline{h}}{\delta \underline{l}}\right|_{\hat{\underline{l}}} \delta l + \underline{\eta}_{LOS}. \tag{5.29}$$

The model is only linear if δl is also linear. To solve for δl, the marker position vector must first be written as a function of the states $\underline{x}$. Fig. 5.1 is an illustration of the vector $\underline{l}$. The geometry in Fig. 5.1 shows that the marker position vector projected into the NED frame can be written as

$$\underline{l}^{ned} = \underline{p}_L^{ned} - \underline{p}_F^{ned} + \underline{p}_{j,L}^{ned}. \tag{5.30}$$

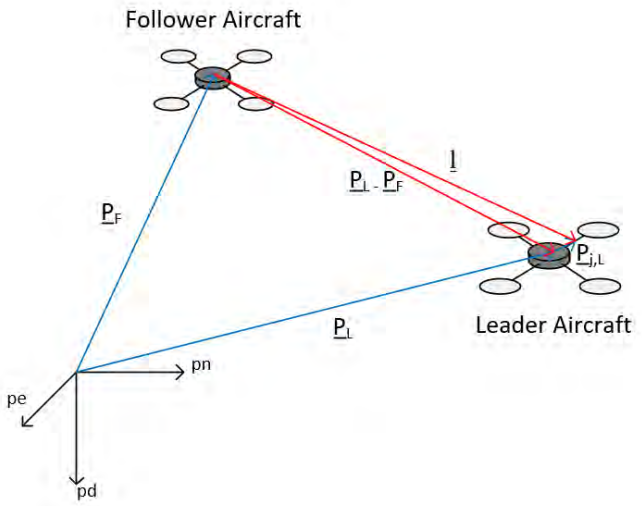

Fig. 5.1: Position Vectors Geometry.

The marker position vector must be projected into the follower aircraft body frame because that is where the measurement is taken:

$$\underline{l} = R_{ned}^{b_F}\left(\underline{p}_L^{ned} - \underline{p}_F^{ned}\right) + R_{ned}^{b_F} R_{b_L}^{ned} \underline{p}_{j,L}^{b_L}. \tag{5.31}$$

where the subscripts and superscripts b_F and b_L denote the body frames of the follower and leader aircraft respectively and $\underline{\mathrm{p}}_{j,L}^{b_L}$ is the position vector of the jth marker with respect to the origin of the leader aircraft projected into the leader body frame. The marker is placed on the body of the leader aircraft and the body of the aircraft is assumed to be rigid, therefore $\underline{\mathrm{p}}_{j,L}^{b_L}$is constant. It is also assumed that the $\underline{\mathrm{p}}_{j,L}^{b_L}$is known. With the leader to marker position vector constant, equation 5.31 is now only a function of the states $\underline{\mathrm{x}}$. To linearize equation 5.31, the following perturbations are defined:

$$\underline{\mathrm{p}}_F^{ned} = \hat{\underline{p}}_F^{ned} + \delta\underline{\mathrm{p}}_F^{ned} \tag{5.32}$$

$$\underline{\mathrm{p}}_L^{ned} = \hat{\underline{p}}_L^{ned} + \delta\underline{\mathrm{p}}_L^{ned} \tag{5.33}$$

$$\underline{\mathrm{l}} = \hat{\underline{\mathrm{l}}} + \delta\underline{\mathrm{l}} \tag{5.34}$$

$$R_{ned}^{b_F} = R_{\hat{b}_F}^{b_F} R_{ned}^{\hat{b}_F} = \left[I_{3x3} - (\delta\theta_{b_F}\times)\right] R_{ned}^{\hat{b}_F} \tag{5.35}$$

$$R_{b_L}^{ned} = R_{\hat{b}_L}^{ned}\left(R_{\hat{b}_L}^{b_L}\right)^T = R_{\hat{b}_L}^{ned}\left[I_{3x3} - (\delta\theta_{b_L}\times)\right]^T. \tag{5.36}$$

Substituting equations 5.32 to 5.36 into 5.31 yields

$$\hat{\underline{\mathrm{l}}} + \delta\underline{\mathrm{l}} = \left[I_{3x3} - (\delta\theta_{b_F}\times)\right] R_{ned}^{\hat{b}_F}\left(\hat{\underline{p}}_L^{ned} + \delta\underline{\mathrm{p}}_L^{ned} - \hat{\underline{p}}_F^{ned} - \delta\underline{\mathrm{p}}_F^{ned}\right)$$

$$+ \left[I_{3x3} - (\delta\theta_{b_F}\times)\right] R_{ned}^{\hat{b}_F} R_{\hat{b}_L}^{ned}\left[I_{3x3} - (\delta\theta_{b_L}\times)\right]^T \underline{\mathrm{p}}_{j,L}^{b_L}. \tag{5.37}$$

After much algebra, the linear equation for the marker position vector is

$$\delta\underline{\mathrm{l}} = R_{ned}^{\hat{b}_F}\delta\underline{\mathrm{p}}_L^{ned} - R_{ned}^{\hat{b}_F}\delta\underline{\mathrm{p}}_F^{ned} + \left(\left[R_{ned}^{\hat{b}_F}\left(\hat{\underline{p}}_L^{ned} - \hat{\underline{p}}_F^{ned}\right)\times\right] + \left[R_{ned}^{\hat{b}_F} R_{\hat{b}_L}^{ned}\underline{\mathrm{p}}_{j,L}^{b_L}\times\right]\right)\delta\theta_{b_F}$$

$$-R_{ned}^{\hat{b}_F} R_{\hat{b}_L}^{ned} \left(\underline{\mathrm{p}}_{j,L}^{b_L} \times \right) \delta\theta_{b_L} \tag{5.38}$$

A full derivation of $\delta\underline{\mathrm{l}}$ with more detail is located in appendix A.2. To finish the derivation of the linear measurement model that is in the form of equation 5.29, the partial derivatives of the nonlinear equations $\underline{\mathrm{h}}$ with respect to the position vector $\underline{\mathrm{l}}$ must be evaluated. The two nonlinear equations of $\underline{\mathrm{h}}$ are found in equation 5.23. The partial derivative matrix is found to be

$$\left.\frac{\delta \underline{\mathrm{h}}}{\delta \underline{\mathrm{l}}}\right|_{\hat{\underline{\mathrm{l}}}} = \left[\begin{array}{ccc} \frac{\delta h_1}{\delta l_x} & \frac{\delta h_1}{\delta l_y} & \frac{\delta h_1}{\delta l_z} \\ \frac{\delta h_2}{\delta l_x} & \frac{\delta h_2}{\delta l_y} & \frac{\delta h_2}{\delta l_z} \end{array} \right] \Bigg|_{\hat{\underline{\mathrm{l}}}} = \left[\begin{array}{ccc} \frac{1}{l_z} & 0 & \frac{-l_x}{l_z^2} \\ 0 & \frac{1}{l_z} & \frac{-l_y}{l_z^2} \end{array} \right] \Bigg|_{\hat{\underline{\mathrm{l}}}} = \left[\begin{array}{ccc} \frac{1}{\hat{l}_z} & 0 & \frac{-\hat{l}_x}{\hat{l}_z^2} \\ 0 & \frac{1}{\hat{l}_z} & \frac{-l_y}{\hat{l}_z^2} \end{array} \right] \tag{5.39}$$

Substituting equations 5.38 and 5.39 into equation 5.29 yields the linear model of the LOS measurement model:

$$\delta \underline{\mathrm{z}}_{LOS} = \left[\begin{array}{ccc} \frac{1}{\hat{l}_z} & 0 & \frac{-\hat{l}_x}{\hat{l}_z^2} \\ 0 & \frac{1}{\hat{l}_z} & \frac{-l_y}{\hat{l}_z^2} \end{array} \right] \Big[R_{ned}^{\hat{b}_F} \delta \underline{\mathrm{p}}_L^{ned} - R_{ned}^{\hat{b}_F} \delta \underline{\mathrm{p}}_F^{ned} +$$

$$\left(\left[R_{ned}^{\hat{b}_F} \left(\hat{\underline{p}}_L^{ned} - \hat{\underline{p}}_F^{ned} \right) \times \right] + \left[R_{ned}^{\hat{b}_F} R_{\hat{b}_L}^{ned} \underline{\mathrm{p}}_{j,L}^{b_L} \times \right] \right) \delta\theta_{b_F} - R_{ned}^{\hat{b}_F} R_{\hat{b}_L}^{ned} \left(\underline{\mathrm{p}}_{j,L}^{b_L} \times \right) \delta\theta_{b_L} \Big] + \underline{\eta}_{LOS} \tag{5.40}$$

where the elements of the nominal marker position vector projected into the follower aircraft body frame are calculated using the following equation:

$$\hat{\underline{\mathrm{l}}} = R_{ned}^{\hat{b}_F} \left(\hat{\underline{p}}_L^{ned} - \hat{\underline{p}}_F^{ned} \right) + R_{ned}^{\hat{b}_F} R_{\hat{b}_L}^{ned} \hat{\underline{p}}_{j,L}^{\hat{b}_L}. \tag{5.41}$$

Putting the linear model in state space form:

$$\delta \underline{\mathrm{z}}_{LOS} = H_{LOS} \delta \underline{\mathrm{x}} + \underline{\eta}_{LOS} \tag{5.42}$$

where

$$H_{LOS} = \begin{bmatrix} \frac{1}{\hat{l}_z} & 0 & \frac{-\hat{l}_x}{\hat{l}_z^2} \\ 0 & \frac{1}{\hat{l}_z} & \frac{-l_y}{\hat{l}_z^2} \end{bmatrix} \begin{bmatrix} R_{ned}^{\hat{b}_F} & 0_{3x3} & \left(-R_{ned}^{\hat{b}_F} R_{\hat{b}_L}^{ned} \left[\underline{\mathrm{p}}_{j,L}^{b_L} \times\right]\right) & 0_{3x6} \end{bmatrix}$$

$$\left(-R_{ned}^{\hat{b}_F}\right) \quad 0_{3x3} \quad \left(\left[R_{ned}^{\hat{b}_F}\left(\hat{\underline{p}}_L^{ned} - \hat{\underline{p}}_F^{ned}\right)\times\right] + \left[R_{ned}^{\hat{b}_F} R_{\hat{b}_L}^{ned} \underline{\mathrm{p}}_{j,L}^{b_L} \times\right]\right) \quad 0_{3x6} \Bigg] \tag{5.43}$$

$$\delta\underline{\mathrm{x}} = \begin{bmatrix} \delta\underline{\mathrm{p}}_L^{ned} \\ \delta\underline{\mathrm{v}}_L^{ned} \\ \delta\underline{\theta}_{b_L} \\ \delta\underline{\mathrm{b}}_{accel,L} \\ \delta\underline{\mathrm{b}}_{gyro,L} \\ \delta\underline{\mathrm{p}}_F^{ned} \\ \delta\underline{\mathrm{v}}_F^{ned} \\ \delta\underline{\theta}_{b_F} \\ \delta\underline{\mathrm{b}}_{accel,F} \\ \delta\underline{\mathrm{b}}_{gyro,F} \end{bmatrix}.$$

For the Kalman filter that only updates using LOS measurements, the linear measurement model is written as

$$\delta\underline{\mathrm{z}}_{LOS}^{'} = H_{LOS}^{'}\delta\underline{\mathrm{x}}^{'} + \underline{\eta}_{LOS} \tag{5.44}$$

where

$$H_{LOS}^{'} = \begin{bmatrix} \frac{1}{\hat{l}_z} & 0 & \frac{-\hat{l}_x}{\hat{l}_z^2} \\ 0 & \frac{1}{\hat{l}_z} & \frac{-l_y}{\hat{l}_z^2} \end{bmatrix} \begin{bmatrix} R_{ned}^{\hat{b}_F} & 0_{3x6} & \left(-R_{ned}^{\hat{b}_F} R_{\hat{b}_L}^{ned} \left[\underline{\mathrm{p}}_{j,L}^{b_L} \times\right]\right) & 0_{3x6} \end{bmatrix}$$

$$\begin{bmatrix} \left(-R_{ned}^{\hat{b}_F}\right) & 0_{3x3} & \left(\left[R_{ned}^{\hat{b}_F}\left(\hat{\underline{p}}_L^{ned} - \hat{\underline{p}}_F^{ned}\right)\times\right] + \left[R_{ned}^{\hat{b}_F} R_{\hat{b}_L}^{ned} \underline{\mathrm{p}}_{j,L}^{b_L}\times\right]\right) & 0_{3x6} \end{bmatrix} \tag{5.45}$$

$$\delta\underline{\mathrm{x}}' = \begin{bmatrix} \delta\underline{\mathrm{p}}_L^{ned} \\ \delta\underline{\mathrm{v}}_L^{ned} \\ \delta\underline{\mathrm{a}}_L^{ned} \\ \delta\underline{\theta}_{b_L} \\ \delta\dot{\underline{\omega}}_L^b \\ \delta\dot{\underline{\alpha}}_L^b \\ \delta\underline{\mathrm{p}}_F^{ned} \\ \delta\underline{\mathrm{v}}_F^{ned} \\ \delta\underline{\theta}_{b_F} \\ \delta\underline{\mathrm{b}}_{accel,F} \\ \delta\underline{\mathrm{b}}_{gyro,F} \end{bmatrix}.$$

5.4.3 Linear Measurement Models for Sensor Architectures

The total linear measurement models for architectures 1 through 7 are given in equations 5.46 through 5.52 respectively. Note that the measurement matrices and error vectors are created from combinations of the measurement models derived in the previous two sections. It is important to note that the LOS and GPS measurements will not always be available at the same time. Discussion on how this is handled in the Kalman filter takes place later in this paper.

$$\delta\underline{\mathrm{z}}_1(t_i, j) = \begin{bmatrix} \begin{bmatrix} H_{gps}(t_i) & 0_{3x15} \\ 0_{3x15} & H_{gps}(t_i) \end{bmatrix} \\ H_{LOS}(t_i, \hat{\underline{x}}(t_i), j) \end{bmatrix} \delta\underline{\mathrm{x}}(t_i) + \begin{bmatrix} \underline{\eta}_{gps}(t_i) \\ \underline{\eta}_{gps}(t_i) \\ \underline{\eta}_{LOS}(t_i) \end{bmatrix} \tag{5.46}$$

$$\delta \underline{z}_2(t_i) = \begin{bmatrix} H_{gps}(t_i) & 0_{3x15} \\ 0_{3x15} & H_{gps}(t_i) \end{bmatrix} \delta \underline{x}(t_i) + \begin{bmatrix} \underline{\eta}_{gps}(t_i) \\ \underline{\eta}_{gps}(t_i) \end{bmatrix} \tag{5.47}$$

$$\delta \underline{z}_3(t_i, j) = \begin{bmatrix} \begin{bmatrix} 0_{3x15} & H_{gps}(t_i) \end{bmatrix} \\ H_{LOS}(t_i, \underline{\hat{x}}(t_i), j) \end{bmatrix} \delta \underline{x}(t_i) + \begin{bmatrix} \underline{\eta}_{gps}(t_i) \\ \underline{\eta}_{LOS}(t_i) \end{bmatrix} \tag{5.48}$$

$$\delta \underline{z}_4(t_i, j) = \begin{bmatrix} \begin{bmatrix} H_{gps}(t_i) & 0_{3x15} \end{bmatrix} \\ H_{LOS}(t_i, \underline{\hat{x}}(t), j) \end{bmatrix} \delta \underline{x}(t_i) + \begin{bmatrix} \underline{\eta}_{gps}(t_i) \\ \underline{\eta}_{LOS}(t_i) \end{bmatrix} \tag{5.49}$$

$$\delta \underline{z}_5(t_i, j) = H_{LOS}(t_i, \underline{\hat{x}}(t_i), j) \delta \underline{x}(t_i) + \underline{\eta}_{LOS}(t_i) \tag{5.50}$$

$$\delta \underline{z}_6(t_i, j) = \begin{bmatrix} \begin{bmatrix} 0_{3x18} & H_{gps}(t_i) \end{bmatrix} \\ H'_{LOS}(t_i, \underline{\hat{x}}'(t_i), j) \end{bmatrix} \delta \underline{x}'(t_i) + \begin{bmatrix} \underline{\eta}_{gps}(t_i) \\ \underline{\eta}_{LOS}(t_i) \end{bmatrix} \tag{5.51}$$

$$\delta \underline{z}_7(t_i, j) = H'_{LOS}(t_i, \underline{\hat{x}}'(t_i), j) \delta \underline{x}(t_i)' + \underline{\eta}_{LOS}(t_i) \tag{5.52}$$

5.5 Kalman Update and State Estimate Correction

The final piece of the Kalman filter to be designed is the Kalman update and error correction stages. The actual measurements from the discrete time sensors and the linear measurement models are used to generate an error for each of the navigation states. The errors are then used to correct the navigation states. The residuals and residual covariances are given by

$$r_i = \tilde{z}_i - \hat{z}_i \tag{5.53}$$

$$\Re_i = H_i P_i^- H_i^T + R \tag{5.54}$$

where $\hat{z}_i$ is given by equations 5.20 or 5.23 for GPS measurement or LOS measurement

respectively evaluated at the ith a priori state estimates and H_i is given by equations 5.22, 5.43, or 5.45 depending on measurement type and architecture evaluated at the ith a priori state estimates. P_i^- is the state covariance propagated to the ith time moment.

The Kalman gain is computed by

$$K_i = P_i^- H_i^T \Re_i^{-1} \tag{5.55}$$

and the navigation state error is estimated to be

$$\delta\hat{x}^+ = K_i r_i. \tag{5.56}$$

The state covariance is updated using the Josephs form update equation for a discrete measurement update:

$$P_i^+ = (I - K_i H_i) P_i^- (I - K_i H_i)^T + K_i R_{gps} K_i^T. \tag{5.57}$$

The final step of the EKF algorithms is to update the estimated states. The navigation states are corrected with the error estimates generated from equation 5.56 as follows:

$$\underline{\mathrm{p}}^{ned} = \underline{\hat{p}}^{ned} + \delta\underline{\mathrm{p}}^{ned+}$$

$$\underline{\mathrm{v}}^{ned} = \underline{\hat{v}}^{ned} + \delta\underline{\mathrm{v}}^{ned+}$$

$$\underline{\mathrm{q}}^{b}_{ned} = \delta\underline{\mathrm{q}}^{b+}_{\hat{b}} \otimes \underline{\mathrm{q}}^{\hat{b}}_{ned}$$

$$\underline{\mathrm{b}}_{accel} = \underline{\hat{b}}_{accel} + \delta\underline{\mathrm{b}}^{+}_{accel}$$

$$\underline{\mathrm{b}}_{gyro} = \hat{\underline{b}}_{gyro} + \delta\underline{\mathrm{b}}^{+}_{gyro}$$

where the attitude estimation error is defined as

$$\delta q^{b}_{\hat{b}} = \begin{bmatrix} \frac{\delta\underline{\theta}^{+}_{b}}{2} \\ 1 \end{bmatrix}.$$

5.6 Validation and Simulation

Each of the navigation architectures derived in this chapter were implemented in a custom Matlab simulation. IMU data generated with the simulation developed in chapter 3 was used for the navigation simulations. For the architecture study simulation trajectory, the leader aircraft remains at a constant position while the follower aircraft approaches from above. The follower aircraft starts at a position that is 10 meters to the east and 20 meters above the leader. The follower executes a glide-slope maneuver over a one-minute interval to approach the leader aircraft finishing the maneuver at a position directly above the leader and at one meter away. Fig. 5.2, Fig. 5.3, and Fig. 5.4 illustrate the described relative trajectory used for this study. The rest of this section presents parameters and values used in the simulation and describes methods for obtaining the final results.

5.6.1 Simulation Parameters

Simulation parameters needed to simulate the navigation solutions presented in this chapter include initial state uncertainties, the number of beacons for LOS measurements, beacon positions for LOS measurements, sensor error model parameters, and model parameters for the 1st order Markov processes used in architecture 6.

It has been determined by other researchers that six beacons for LOS measurements should be used in practice to get full-pose information [15]. For this reason, six beacons were used in the simulations performed in this study. It also has been found in previous research that the markers should not be placed on a single two-dimensional plane for the

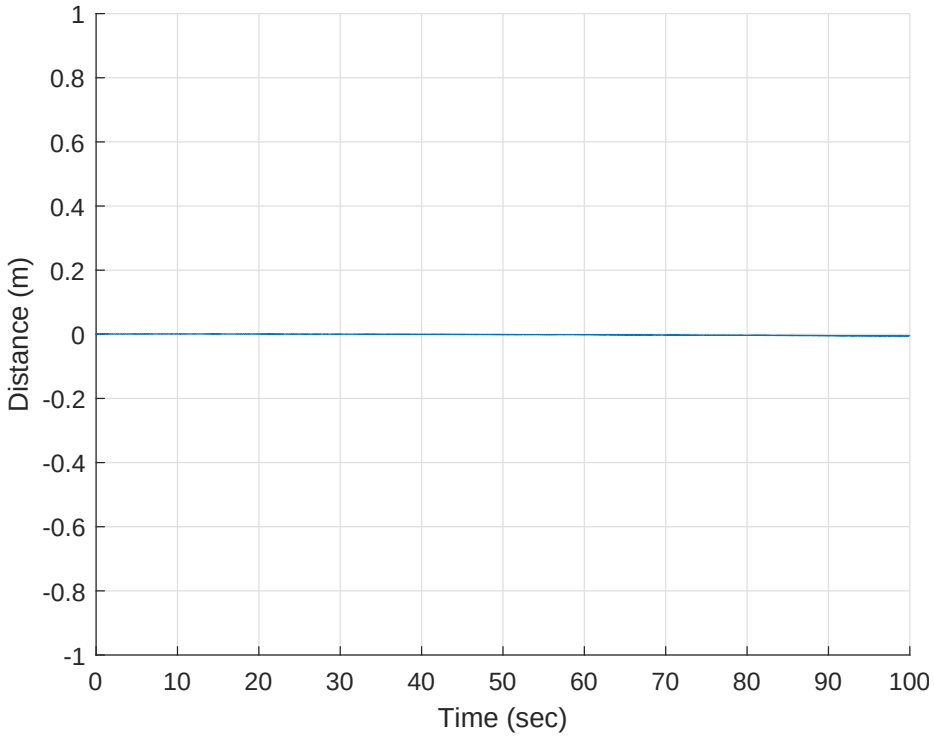

Fig. 5.2: True Relative North Position.

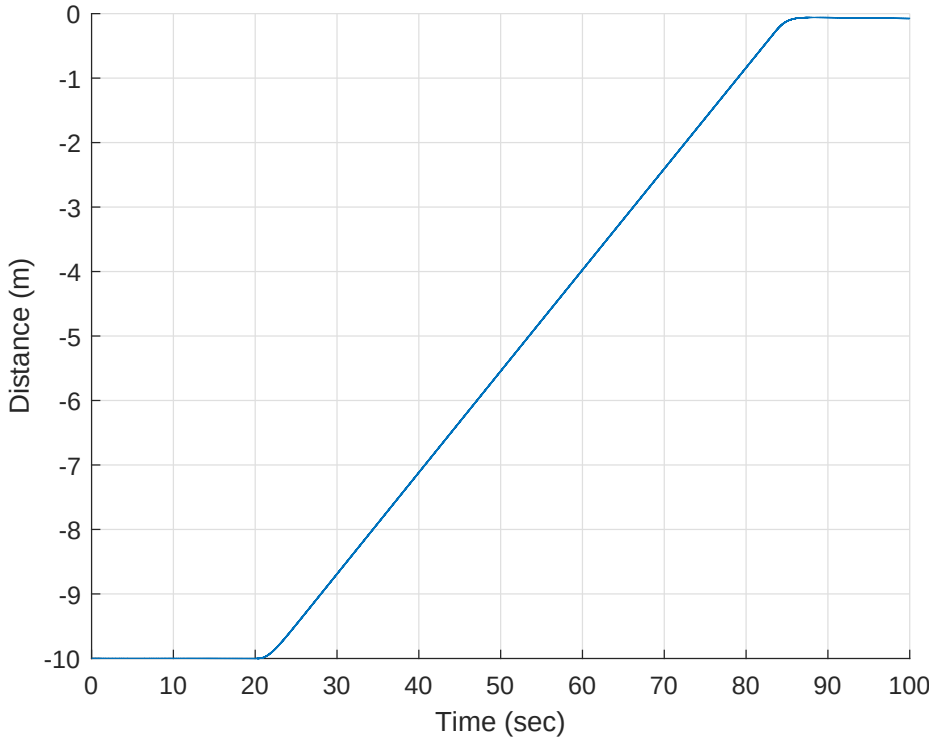

Fig. 5.3: True Relative East Position.

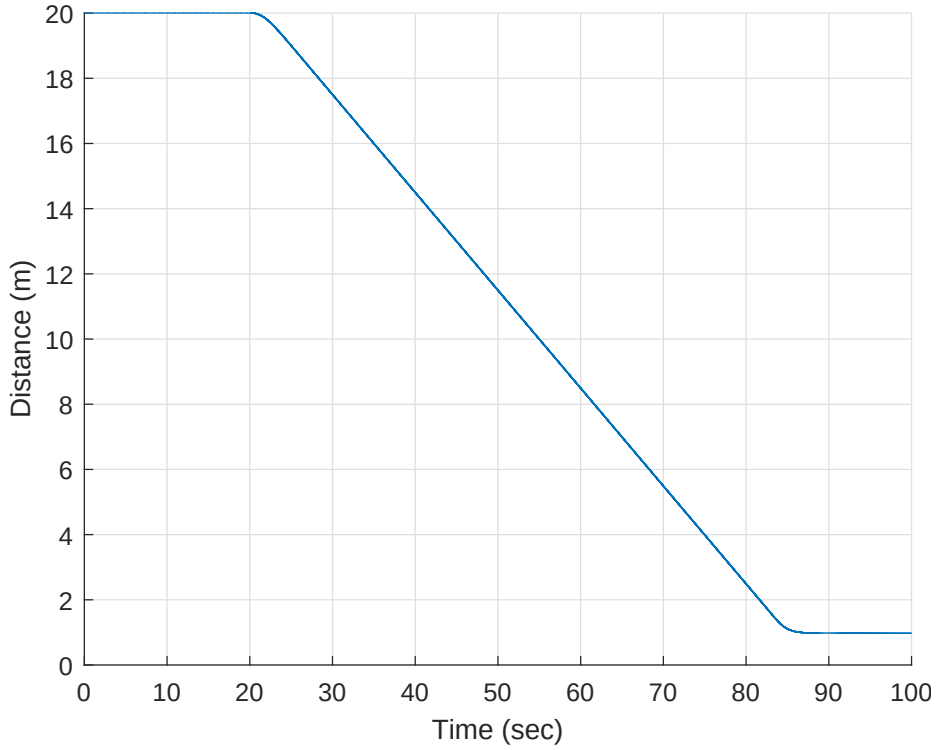

Fig. 5.4: True Relative Down Position.

Table 5.2: Beacon Positions in the Leader Body Frame.

Beacon	X Position (m)	Y Position (m)	Z Position (m)
1	0	0	0
2	0.2	0	0
3	-0.2	0	0
4	0	0.2	0
5	0	-0.2	0
6	0	0	0.2

best results [15]. The beacon positions were chosen to take advantage of this fact and to place the beacons at reasonable locations for a small sized UAV. The beacon positions can be found in table 5.2.

Two parameters need to be specified for each of the first order Markov processes: a time constant and a noise strength. Since these processes are used to model the dynamics of the leader aircraft which is holding a single position, the accelerations are only due

to disturbances and are attenuated by the quad-rotor controller. Reasonable disturbance values were extracted from an existing simulation [26] and then injected into the custom simulation developed in chapter 3. The power spectral density (PSD) was then calculated for the linear and angular accelerations of the simulated aircraft. A PSD plot for a first order Markov process was then fitted to the PSD of the acceleration data to determine appropriate model parameters. The 1st order Markov process model parameters can be found in table 5.3.

Table 5.3: 1st Order Markov Parameters

Parameter	Value	Units
linear acceleration ECRV time constant	0.0796	sec
angular acceleration ECRV time constant	0.0398	sec
3-sigma steady-state linear acceleration	9.92e-5	m/s^2
3-sigma steady-state angular acceleration	5.57e-4	rad/s^2

The values selected for the initial state uncertainties are listed in table 5.4.

Table 5.4: Simulation Parameters

Parameter	Value	Units
3-sigma initial position uncertainty	3	m
3-sigma initial position uncertainty	3	m
3-sigma initial position uncertainty	10	m
3-sigma initial velocity uncertainty	0.1	m/sec
3-sigma initial velocity uncertainty	0.1	m/sec
3-sigma initial velocity uncertainty	0.1	m/sec
3-sigma initial orientation uncertainty	0.1	rad
3-sigma initial orientation uncertainty	0.1	rad
3-sigma initial orientation uncertainty	0.1	rad

5.6.2 Sensor Values

The final simulation parameters that need to be specified are the parameters for the sensor error models for the GPS, IMU, and camera sensors. A single error model will be selected for both the GPS and camera sensors, but different IMU parameters sets will be

used for three different IMU grades. The low, medium, and high grade IMU parameters are taken from the data sheets of the Sensonor STIM300, Northrop Grumman LN200, and Honeywell HG9900 respectively [27–29]. The actual values used in the simulation are found in tables 5.5 to 5.7.

Table 5.5: Low Grade IMU Error Model Values

Parameter	Value	Units
Accel. bias ECRV time constant	600	sec
Gyro bias ECRV time constant	600	sec
3-sigma steady-state accel bias	2.136	mg
3-sigma velocity random walk	0.07	m/s/sqrt(hr)
3-sigma steady-state gyro bias	14.3	deg/hr
3-sigma angular random walk	0.15	deg/sqrt(hr)

Table 5.6: Medium Grade IMU Error Model Values

Parameter	Value	Units
Accel bias ECRV time constant	3600	sec
Gyro bias ECRV time constant	3600	sec
3-sigma steady-state accel bias	0.3	mg
3-sigma velocity random walk	0.06	m/s/sqrt(hr)
3-sigma steady-state gyro bias	1.0	deg/hr
3-sigma angular random walk	0.07	deg/sqrt(hr)

Table 5.7: High Grade IMU Error Model Values

Parameter	Value	Units
Accel bias ECRV time constant	3600	sec
Gyro bias ECRV time constant	3600	sec
3-sigma steady-state accel bias	0.025	mg
3-sigma velocity random walk	0.00225	m/s/sqrt(hr)
3-sigma steady-state gyro bias	0.003	deg/hr
3-sigma angular random walk	0.002	deg/sqrt(hr)

To calculate the GPS sensor noise, equation 2.39 from [30] was used. The satellite geometry provided in table 2.1 of [30] and the receiver noise of table 5.4 from [31] were

chosen as parameters to calculate the final measurement noise matrix:

$$R = \begin{bmatrix} 0.168 & 0 & 0 \\ 0 & 0.168 & 0 \\ 0 & 0 & 11.01 \end{bmatrix}. \tag{5.58}$$

The LOS measurement noise strength is found by calculating

$$\sigma_{LOS} = radians(FOV)/Resolution * PixelError. \tag{5.59}$$

A camera with a resolution of 4096 pixels and a FOV of 70 degrees is chosen. It is assumed that the fiducials can be detected within 8 pixels.

The final note on sensor parameters in the navigation simulations is that the sample rates for each sensor are chosen to be 1 Hz, 10 Hz, and 100 Hz for the GPS, LOS, and IMU respectively.

5.6.3 Key Results

To compare the sensor architectures quantitatively, a covariance analysis is used to determine performance and sensitivities of each architecture over the same trajectory. The covariance analysis is done simply by analyzing the elements from the state covariance matrix P extracted from a simulation run. The diagonal elements of a state covariance matrix contain the estimated state variable variances at the time the state covariance matrix was calculated. For example, the value in the 3rd row and 3rd column of the state covariance matrix for the navigation state vector is the variance of the leader aircraft down position estimate.

Relative position and attitude are the values of interest for this study, but the filters designed in this chapter estimate each aircraft's individual position and attitude. The relative position, attitude, and corresponding covariances are calculated as follows:

$$\underline{\mathrm{p}}_{rel}^{ned} = \underline{\mathrm{p}}_{L}^{ned} - \underline{\mathrm{p}}_{F}^{ned} \tag{5.60}$$

$$q_{b_F}^{b_L} = q_{ned}^{b_L} \otimes q_{b_F}^{ned} \tag{5.61}$$

$$P_{rel} = APA^T \tag{5.62}$$

where

$$A = \begin{bmatrix} I_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x6} & -I_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x6} \\ 0_{3x3} & 0_{3x3} & I_{3x3} & 0_{3x6} & 0_{3x3} & 0_{3x3} & -I_{3x3} & 0_{3x6} \end{bmatrix} \tag{5.63}$$

for architectures 1 through 5 and

$$A = \begin{bmatrix} I_{3x3} & 0_{3x6} & 0_{3x3} & 0_{3x6} & -I_{3x3} & 0_{3x3} & 0_{3x3} & 0_{3x6} \\ 0_{3x3} & 0_{3x6} & I_{3x3} & 0_{3x6} & 0_{3x3} & 0_{3x3} & -I_{3x3} & 0_{3x6} \end{bmatrix} \tag{5.64}$$

for the 6th and 7th architecture.

Since the distribution of the process and sensor noise errors used in the simulations are selected to be Gaussian, it is expected that the distribution of states estimate errors are also Gaussian. With a Gaussian distribution of state estimate errors, it can be assumed that about 99% of the state estimate errors should be within three standard deviations of the true state. So, calculating the three standard deviation values, or 3-sigma values, for each state of interest shows how close 99% of state estimates will be to the true states. The final 3-sigma values for relative position and attitude are chosen to be key results from this navigation study because they show how well each navigation solution estimates the relative position and attitude from 1 meter away and at steady state. These results are tabulated by architecture and IMU grade in chapter 7.

Other key results are obtained by plotting the 3-sigma uncertainties of each relative state as a function of distance. The LOS measurement geometry is sensitive to where markers are located in its FOV, so as the distance between the two aircraft decreases the markers span a greater are of the cameras FOV resulting in stronger measurements. Since

the flight path was planned so that each beacon stays within the FOV of the follower aircrafts camera, state uncertainties are expected to decrease as distance decreases. The plots generated as explained in this paragraph will also help to give insight into what the relative state uncertainties will be at distances greater than 1 meter.

5.6.4 Filter Validation

To verify that the EKF of each architecture is functioning properly, a few methods can be used to validate the filter's results. A Monte Carlo analysis is commonly used to validate the results of the covariance analysis. Monte Carlo analyses are completed by running the navigation solution multiple times with varying initial state and sensor noise errors. About 99% of the simulated filter estimates should stay within the 3-sigma bounds found in the covariance analysis. If this is not the case, it is an indication that either the simulated navigation solution is implemented incorrectly or the designed filter is not capable of estimating the states for the given system or trajectory.

The implemented filters from this chapter were tested using the Monte Carlo analysis described in this section. Plots of the analyses is given in chapter 7 and appendix B.

Residual monitoring is an additional tool that can not only be used for validating the proper functioning of a Kalman filter, but can be used to detect malfunctioning sensors. A residual monitoring is similar to the previously mentioned covariance analysis, in that the residual of sensor measurements should lie within the 3-sigma value of the residual covariance. The residual and residual covariance are each calculated using equations 5.53 and 5.54 respectively. Plots of residual monitoring are also found in chapter 7.

CHAPTER 6

CONTROLLER SIMULATION RESULTS

Results from the two different simulated flights described in chapter 4 are presented in this chapter. First, simulation results without the two DOF manipulator are presented. Afterwards, simulation results with the manipulator added to the system are given. Finally, this chapter ends with a discussion on the provided results including conclusions and the significance of those conclusions.

6.1 Position and Heading Tracking without Manipulator

This section provides the position and angle errors between the provided leader UAV states plus an offset and the follower UAV states when the position and heading tracker provided in chapter 4 is implemented without the manipulator. It was expected that the position and heading errors would be small because those states are tracked by the controller and that the roll and pitch angles errors would be slightly higher. The roll and pitch angles are not expected to be extremely large due to the fact that the two UAVs have the same exact dynamics. The results over the whole simulated flight are provided in Fig. 6.1 and Fig. 6.2. Fig. 6.3 gives a closer look at the errors for the first seconds of the flight and at the steady state error found after the transient phase.

Recall that the aircraft start at a point-hold condition so that the initial velocities are zero, the leader aircraft immediately starts a constant attitude circular orbit maneuver, and the follower aircraft is constantly trying to synchronize with the leader aircraft but with a position offset in the up direction. Note that the position offset stops changing at about 55 seconds which is considered the rendezvous point. At this point, the follower aircraft attempts to maintain a constant 2-meter offset.

It can be seen from the provided plots that the majority of the error occurs during the initial seconds of the simulation when the leader aircraft accelerates from a point hold

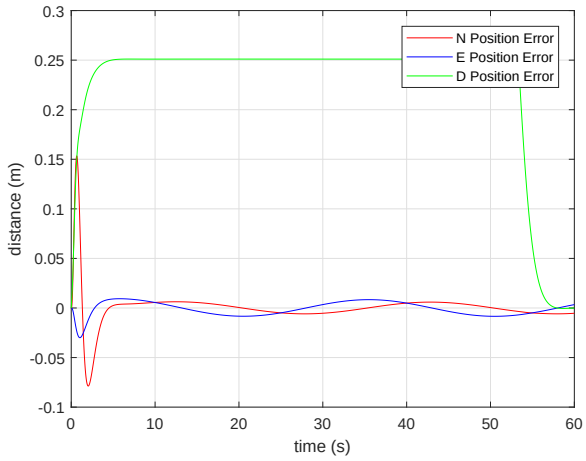

Fig. 6.1: NED Position Errors

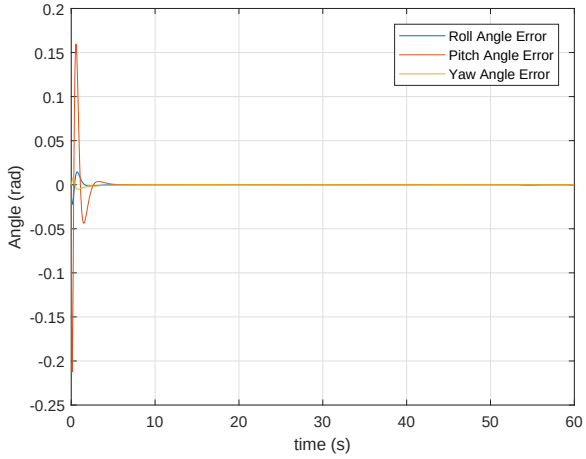

Fig. 6.2: Angle Errors without Manipulator.

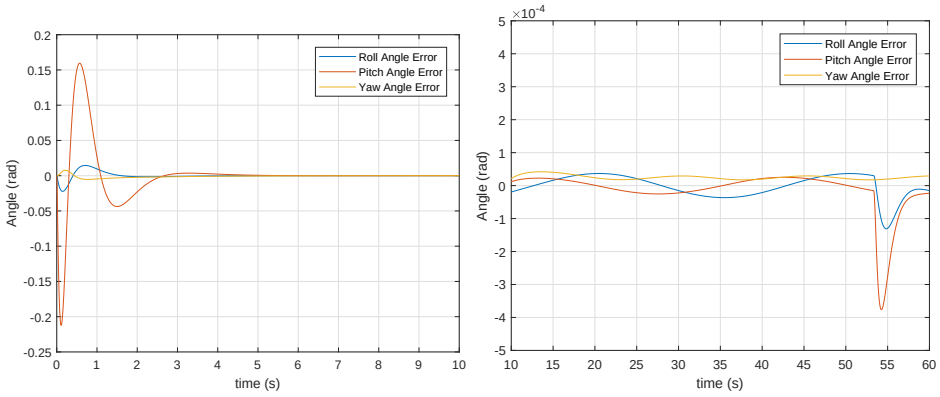

Fig. 6.3: Angle Errors without Manipulator for First 10 Seconds and Steady State Error.

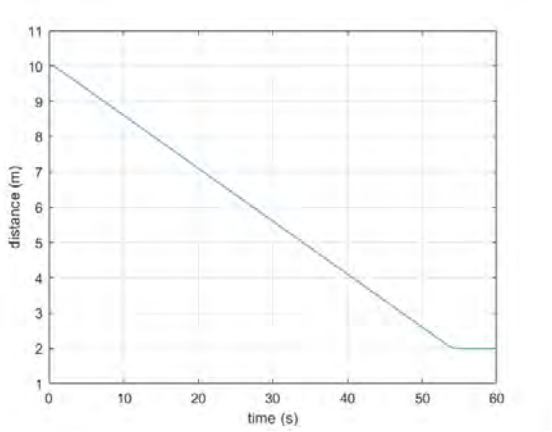

Fig. 6.4: Distance Between Aircraft.

flight mode to a circular orbit maneuver. The errors of the north and down positions are significantly smaller with only centimeters of error for the first few seconds, but the down position error is more significant in that it has a full 10 cm of error for the majority of the flight.

The roll and pitch angles were higher than the yaw angle error as expected. The worst error for the simulated trajectory was found to be 0.22 radians or 12.6 degrees. The highest yaw angle error was 0.021 radians or 1.2 degrees.

6.2 Full-Pose Tracking with 2-DOF Manipulator

Since the linkage distances for the manipulator are neglected, there is no difference in the position errors from the previous section. The angle errors, however, are significantly reduced. The new angle errors are plotted in Fig. 6.5 and Fig. 6.6.

The provided plot shows that the roll and pitch angle errors were significantly reduced. The max error from the simulation was reduced to 0.084 radians or 4.8 degrees. The servos in the simulation were modeled with consumer grade performance parameters. Models for higher grade servos or alternative actuators could be used and better performance would be expected. The bandwidth of the servos used for this work was set to be 8.5 Hz.

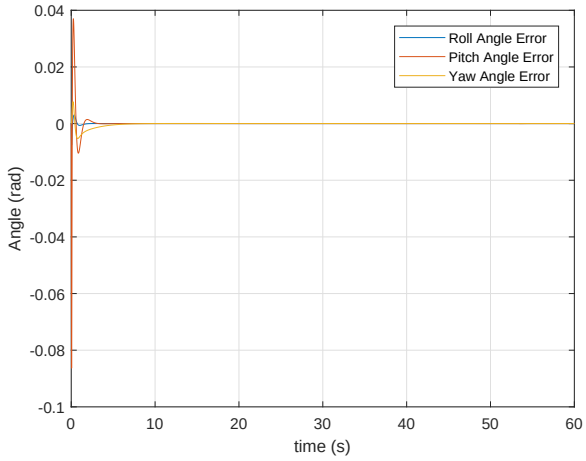

Fig. 6.5: Angle Errors with Manipulator.

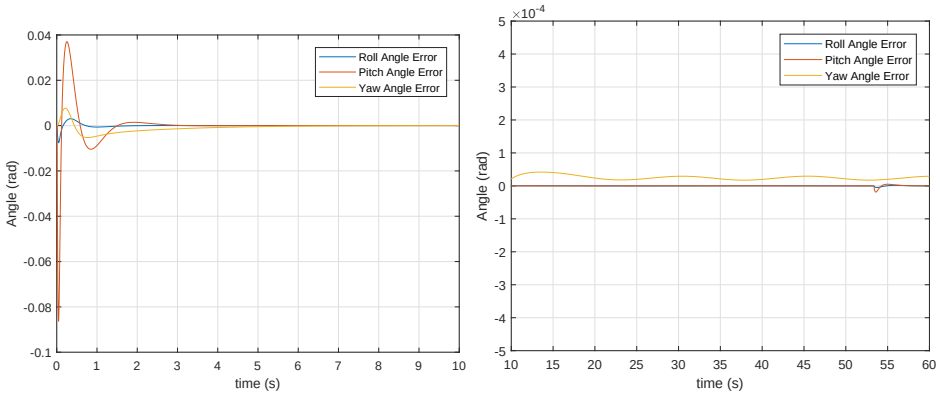

Fig. 6.6: Angle Errors with Manipulator for First 10 Seconds and Steady State Error.

6.3 Discussion

6.3.1 Position and Heading Tracking without Manipulator

The significant error for the down position tracking merits an explanation. The controller designed in chapter 4 uses the leader aircrafts position, linear velocities, and linear accelerations to calculate feed-forward inputs that are sent to the follower aircraft's actuators. The tracking controller also adds a moving position offset to the leader aircraft's position input but the corresponding linear velocities and accelerations needed to follow that moving offset are not calculated. This lack of anticipation for the offset movement causes the significant error seen in Fig. 6.1.

If the analytical derivatives of the offset movement can be calculated, a simple solution would be to calculate the velocity and acceleration of the offset movement and add them to the feed-forward input calculations in the differential flatness algorithm. This solution is left to future work. This error could also be treated as a disturbance on the system. The controller could be redesigned to better reject disturbances by adding an integrator to the LQR controller.

The error values presented in this chapter for the first simulation may be acceptable for certain aerial robotics applications. For the presented simulated flight, the angle errors at the time of rendezvous are below 10 mrads. The small errors without the manipulator show that the tracking controller designed in this work is a good initial design for achieving full-pose synchronization. The effect of wake disturbances was ignored in this study and introducing these disturbances will likely degrade performance, but that disturbance analysis is left to future work. If the errors of the first system presented are sufficiently low for a specific aerial-robotic mission, the complexity of adding a manipulator can be avoided.

6.3.2 Full-Pose Tracking with 2-DOF Manipulator

For aerial-robotic missions that require smaller angle errors, adding a 2-DOF manipulator caused a 60% improvement on the maximum angle errors seen for the simulated flights.

Other simulation runs using different servo models caused an even greater performance improvement, thus a desired angle error can be achieved through choosing higher performing servos. The steady state error for the roll and pitch angles are also reduced by more than 90%. The steady state error for the yaw angle is not reduced as expected.

Further work is needed to improve position tracking performance and to verify the designs presented in this work can handle wake disturbances and other typical disturbances, but the performance found in the simulation results show that the given designs have a great potential to achieve full-pose synchronization.

CHAPTER 7

NAVIGATION SIMULATION RESULTS

This chapter presents the results of the navigation architecture study. First, section 7.1 presents results from the Monte Carlo analysis and residual monitoring explained in 5.6.4 to help support that the implemented filters function properly. Next, the following section provides tabulated data for the steady state relative state uncertainties for each architecture and for each IMU grade at a 1-meter distance. Section 7.3 plots the relative state uncertainties over the entire glide-slope maneuver simulation using low-grade IMUs only. Finally, the concluding section discusses the presented results and gives conclusions from those results.

7.1 Filter Validation

This section provides results from the validation methods given in chapter 5. The plots in this section are generated using a medium grade IMU and architecture 1. Similar plots were generated for each combination of architectures and IMU grades to verify proper functioning filters. Since each set of plots had similar results, only one set of plots are presented in this chapter. Plots for each architecture using a medium-grade IMU is provided in appendix B.

The figures from Fig. 7.1 to Fig. 7.6 are the results from a Monte Carlo analysis. The red dashed lines are the 3-sigma uncertainty values calculated from values in the state covariance matrix. Each gray solid line is the error between the state estimate of the navigation filter for a single Monte Carlo run and the true aircraft state. One-hundred Monte Carlo runs were used for this analysis.

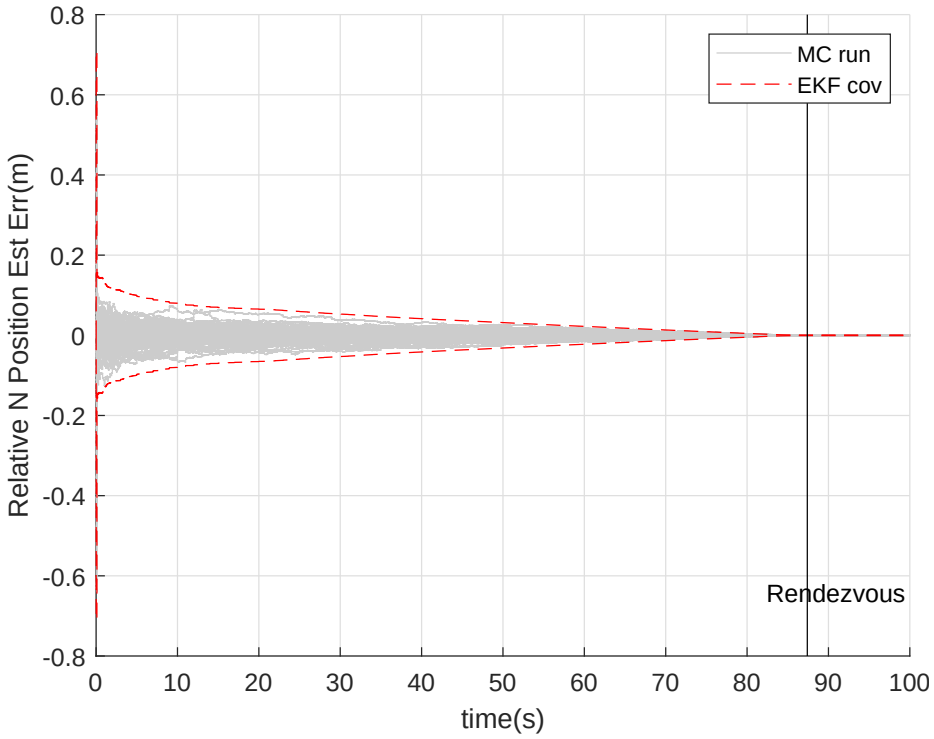

Fig. 7.1: Architecture 1 Monte Carlo Analysis of Relative North Position.

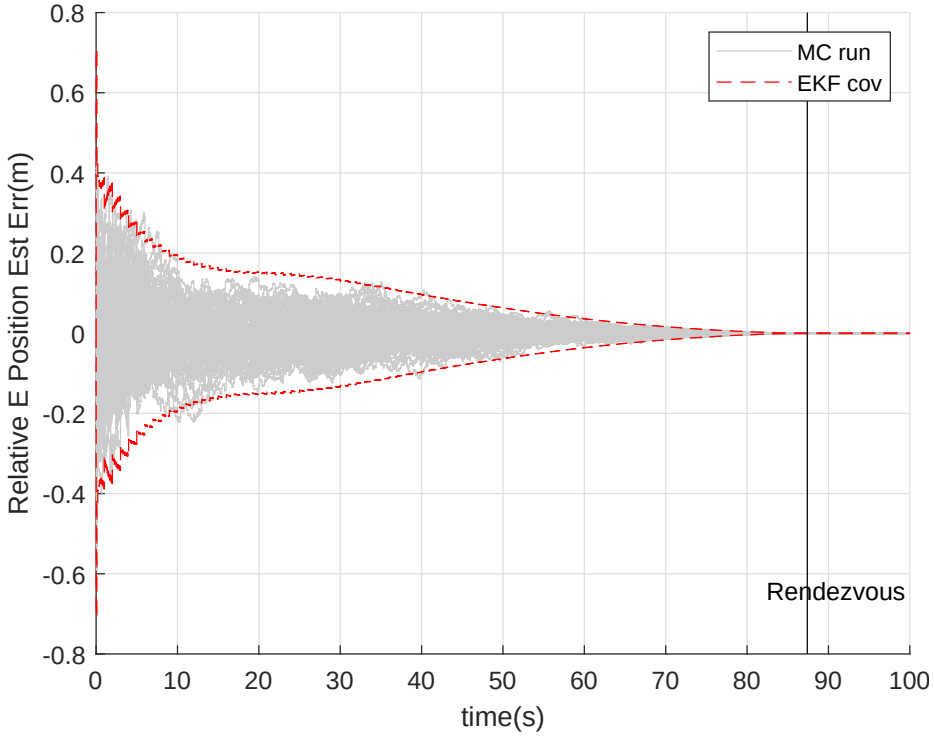

Fig. 7.2: Architecture 1 Monte Carlo Analysis of Relative East Position.

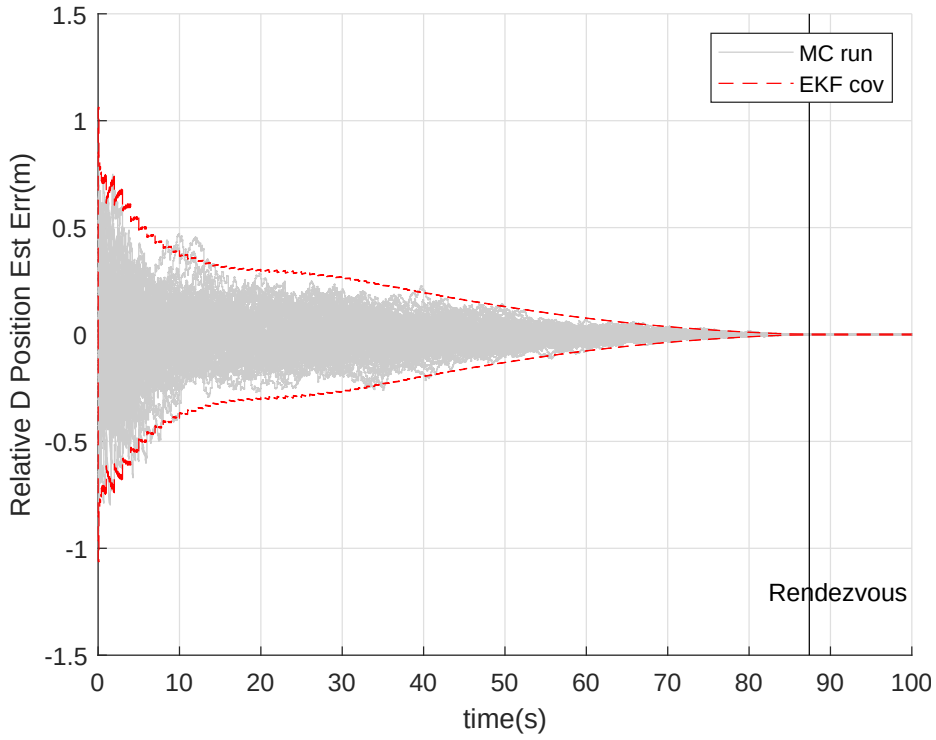

Fig. 7.3: Architecture 1 Monte Carlo Analysis of Relative Down Position.

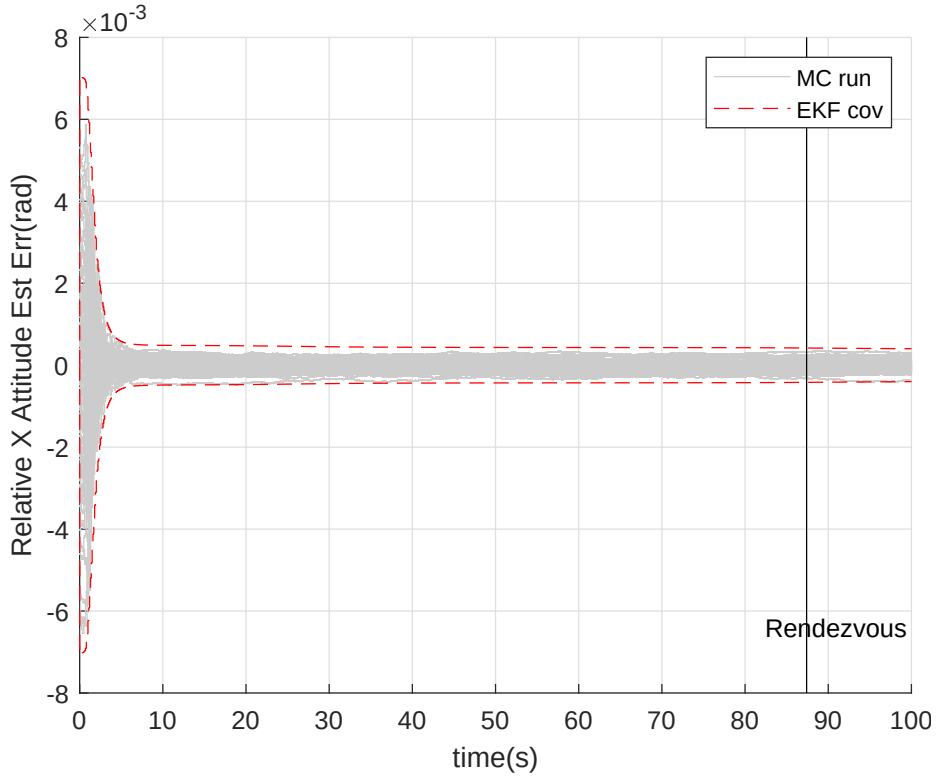

Fig. 7.4: Architecture 1 Monte Carlo Analysis of Relative X Axis Attitude.

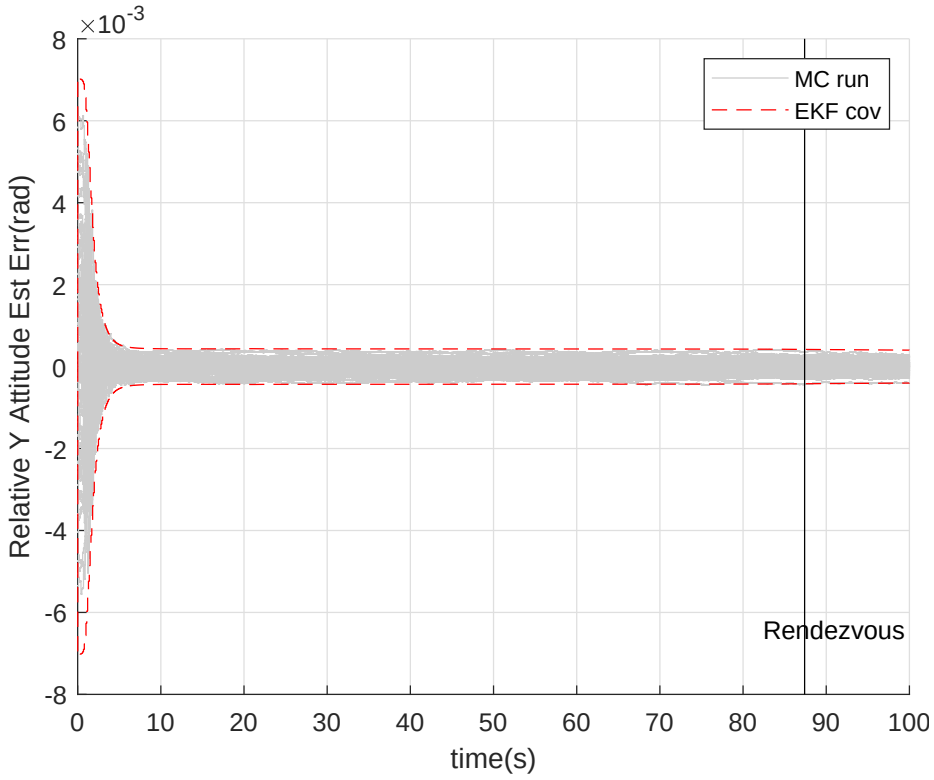

Fig. 7.5: Architecture 1 Monte Carlo Analysis of Relative Y Axis Attitude.

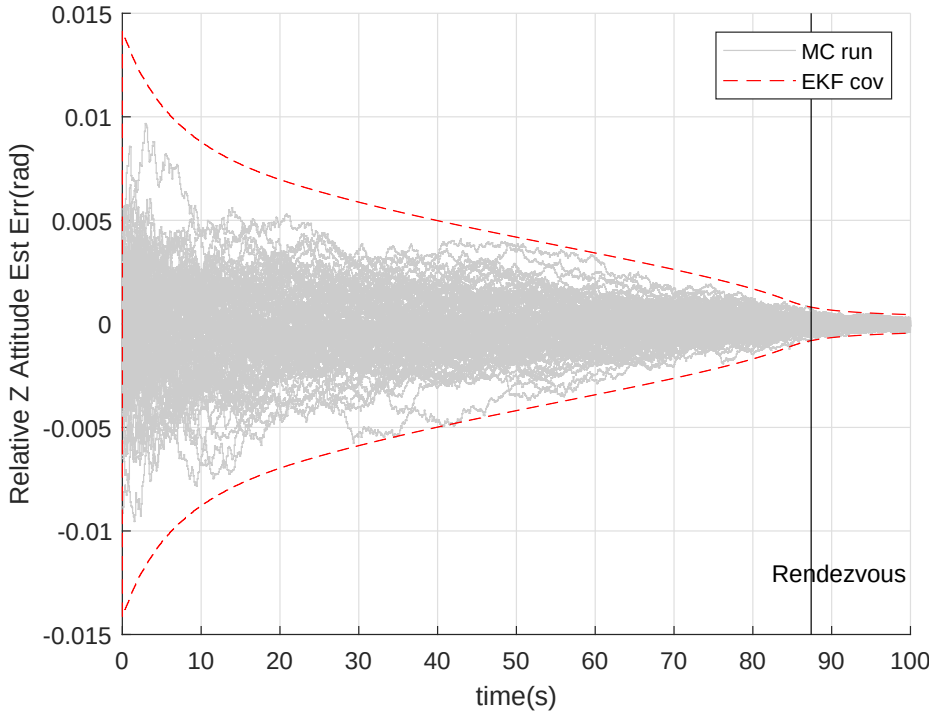

Fig. 7.6: Architecture 1 Monte Carlo Analysis of Relative Z Axis Attitude.

The figures from Fig. 7.7 to Fig. 7.11 are the results from a residual monitoring analysis. Only the follower position GPS measurement and the two LOS measurements for a single beacon are given. Plots were generated for each measurement in the filter but the results are extremely similar. The red dashed lines are the 3-sigma uncertainty values calculated from values in the residual covariance matrix. The blue solid lines are the residuals of the measurements.

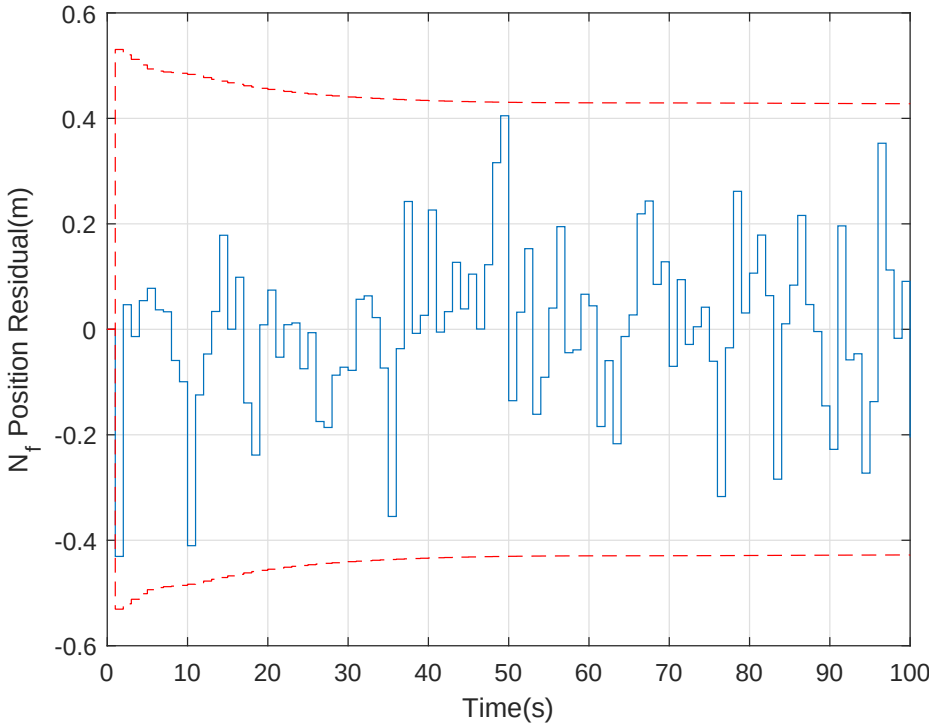

Fig. 7.7: Residual and 3-sigma Residual Covariance for GPS North Measurement.

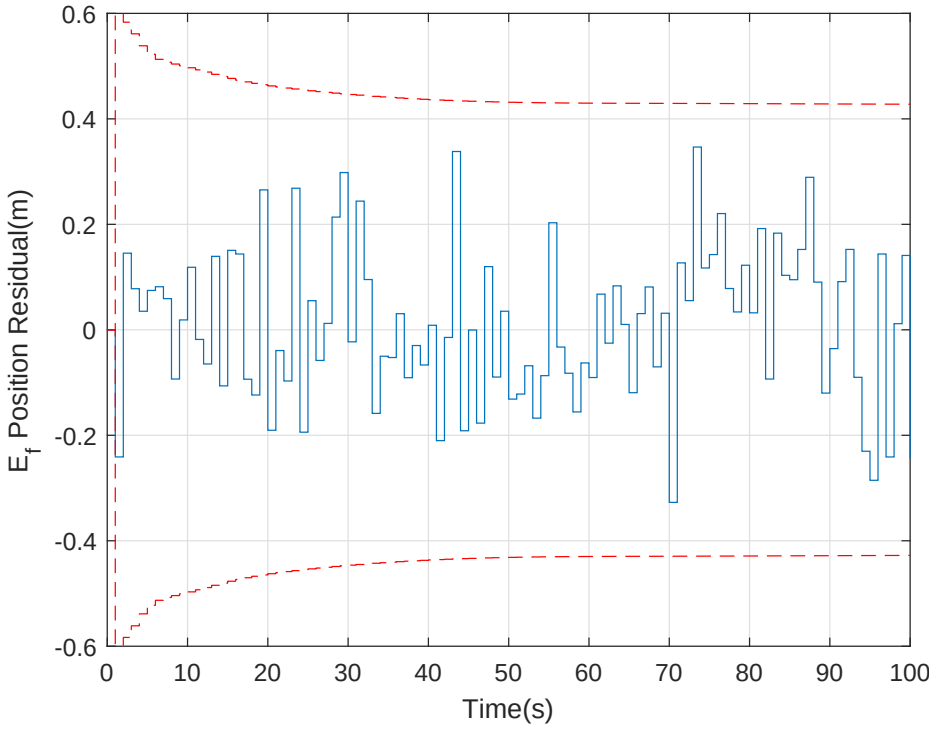

Fig. 7.8: Residual and 3-sigma Residual Covariance for GPS East Measurement.

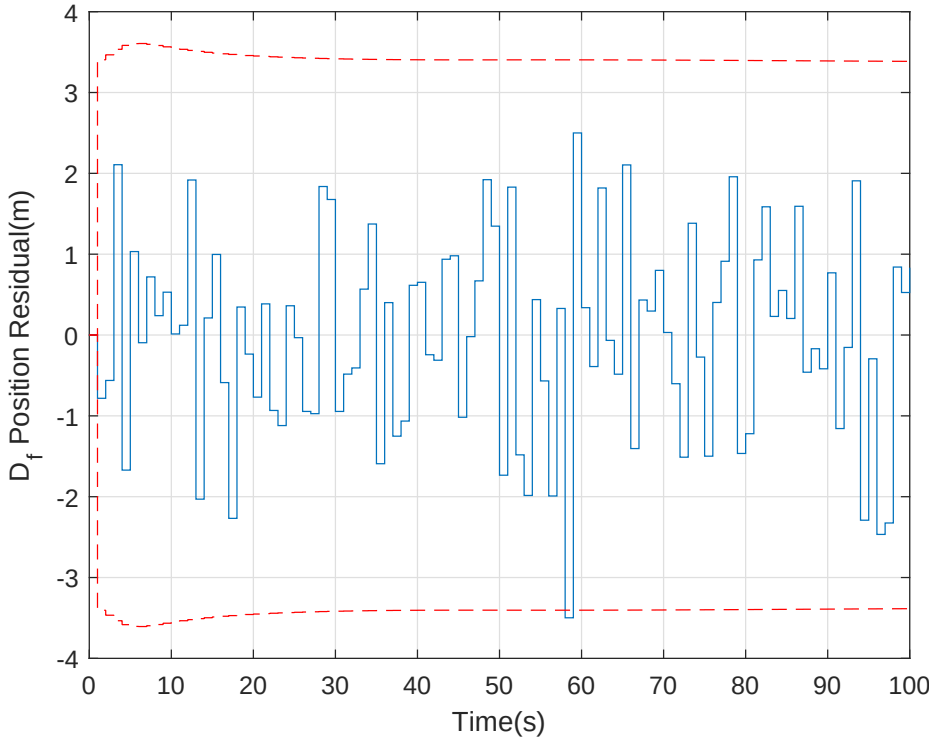

Fig. 7.9: Residual and 3-sigma Residual Covariance for GPS South Measurement.

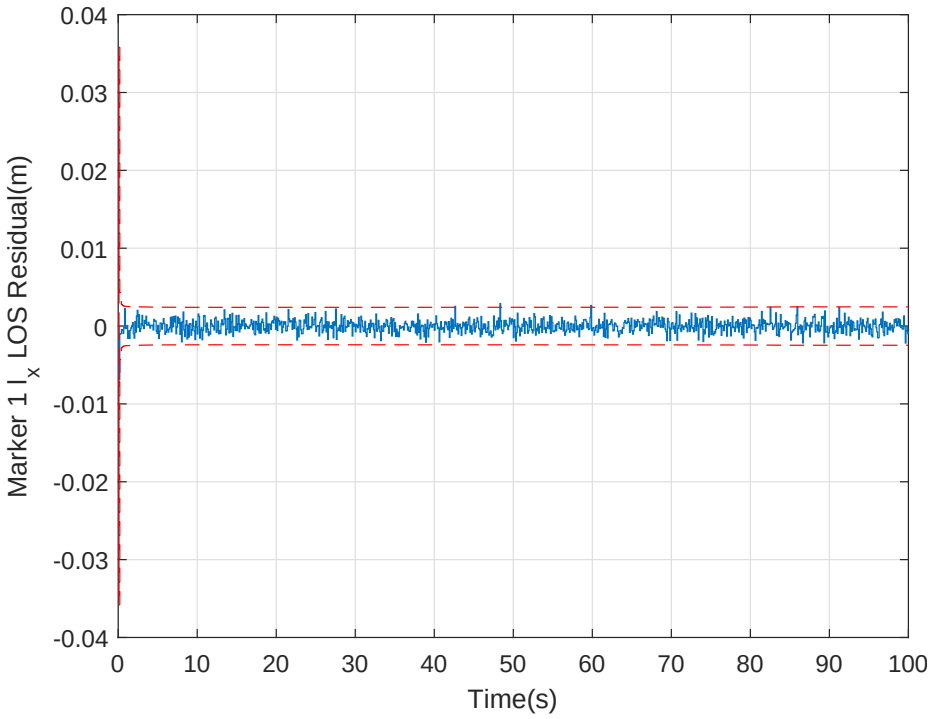

Fig. 7.10: Residual and 3-sigma Residual Covariance for LOS X Measurement for Beacon 1.

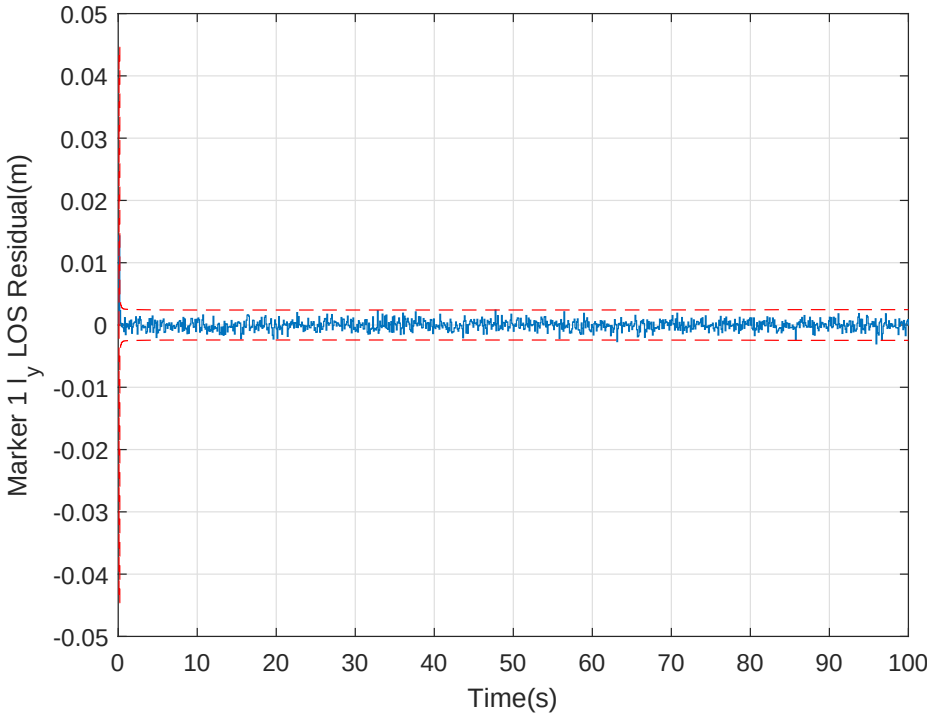

Fig. 7.11: Residual and 3-sigma Residual Covariance for LOS Y Measurement for Beacon 1.

7.2 Steady State Relative State Uncertainties

This section provides the steady state 3-sigma uncertainty values for each architecture and IMU grade combination at the final rendezvous position of 1-meter distance. Tables 7.1 through 7.3 gives the north, east, and down 3-sigma relative position values. Tables 7.4 through 7.6 gives the x, y, and z 3-sigma relative attitude values. These values are taken from the state covariance matrix at the end of the simulated flight, so these are the uncertainty values of the relative states when the follower aircraft has remained at a 1-meter distance for a significant amount of time.

Table 7.1: North 3-Sigma Relative Position Uncertainty.

IMU Grade	Arch 1	Arch 2	Arch 3	Arch 4	Arch 5	Arch 6	Arch 7
Low	0.00658	0.304	0.008	0.00807	0.0141	0.00914	0.00921
Medium	0.00472	0.241	0.00578	0.00587	0.00982	0.00665	0.00673
High	0.00447	0.189	0.00537	0.00544	0.00789	0.00629	0.00634

Table 7.2: East 3-Sigma Relative Position Uncertainty.

IMU Grade	Arch 1	Arch 2	Arch 3	Arch 4	Arch 5	Arch 6	Arch 7
Low	0.00636	0.304	0.00649	0.0065	0.0135	0.00672	0.00672
Medium	0.00383	0.241	0.00383	0.00384	0.00904	0.00371	0.00371
High	0.00317	0.189	0.00318	0.00318	0.00705	0.00321	0.00322

Table 7.3: Down 3-Sigma Relative Position Uncertainty.

IMU Grade	Arch 1	Arch 2	Arch 3	Arch 4	Arch 5	Arch 6	Arch 7
Low	0.0135	1.7	0.0135	0.0135	0.0141	0.0131	0.0131
Medium	0.0106	1.41	0.0106	0.0106	0.011	0.0104	0.0104
High	0.00874	0.997	0.00876	0.00876	0.00899	0.00898	0.00901

7.3 Relative State Uncertainties with Respect to Distance

Table 7.4: X Axis 3-Sigma Relative Attitude Uncertainty.

IMU Grade	Arch 1	Arch 2	Arch 3	Arch 4	Arch 5	Arch 6	Arch 7
Low	0.124	0.168	0.125	0.125	0.125	0.363	0.363
Medium	0.0244	0.0266	0.0244	0.0244	0.0244	0.363	0.363
High	0.00203	0.00256	0.00204	0.00204	0.00204	0.361	0.361

Table 7.5: Y Axis 3-Sigma Relative Attitude Uncertainty.

IMU Grade	Arch 1	Arch 2	Arch 3	Arch 4	Arch 5	Arch 6	Arch 7
Low	0.126	0.168	0.128	0.128	0.128	0.375	0.375
Medium	0.0244	0.0266	0.0244	0.0244	0.0244	0.376	0.375
High	0.00203	0.00257	0.00203	0.00203	0.00203	0.375	0.375

This section plots the relative state uncertainties with respect to the distance between both aircraft. Assuming that navigation architectures using low-grade IMUs will have the most difficulty in close-proximity environments, the in-depth look at each navigation architectures performance provided in this section is limited to only using low-grade IMUs.

Table 7.6: Z Axis 3-Sigma Relative Attitude Uncertainty.

IMU Grade	Arch 1	Arch 2	Arch 3	Arch 4	Arch 5	Arch 6	Arch 7
Low	0.115	0.922	0.115	0.115	0.115	0.209	0.209
Medium	0.0967	0.811	0.0969	0.0969	0.0969	0.209	0.209
High	0.0962	0.809	0.0963	0.0963	0.0963	0.209	0.209

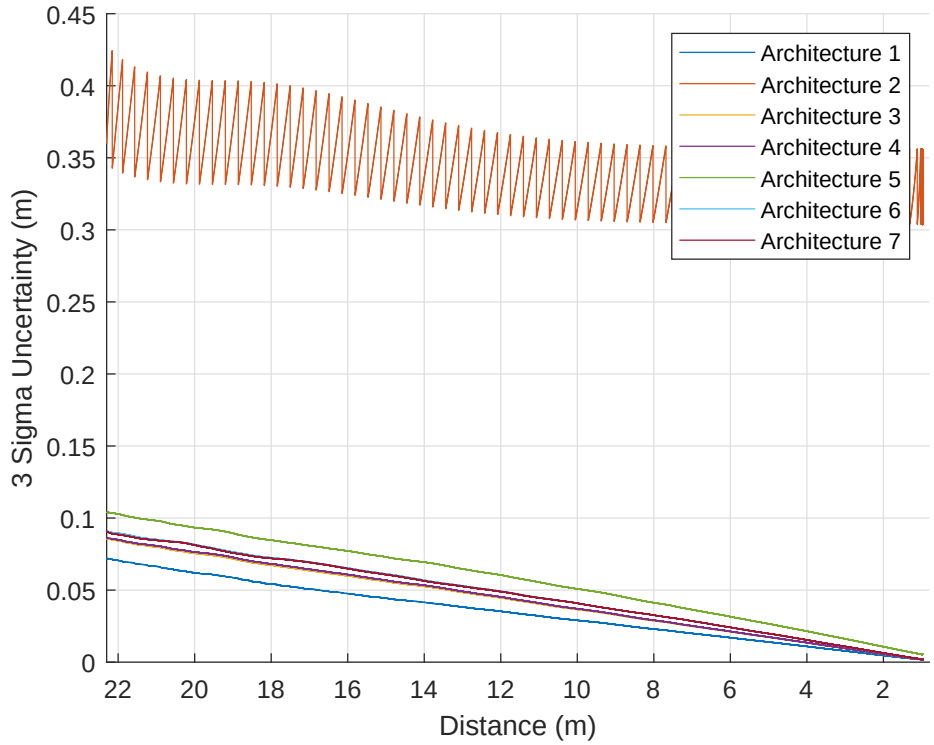

Fig. 7.12: North Relative Position 3-Sigma Values Over True Distance.

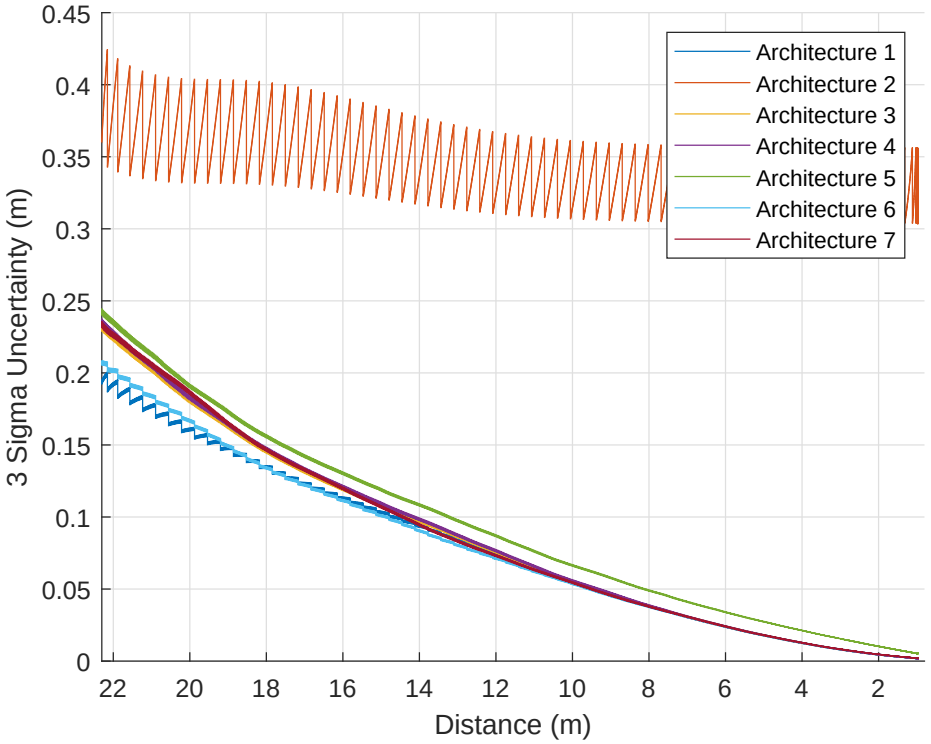

Fig. 7.13: East Relative Position 3-Sigma Values Over True Distance.

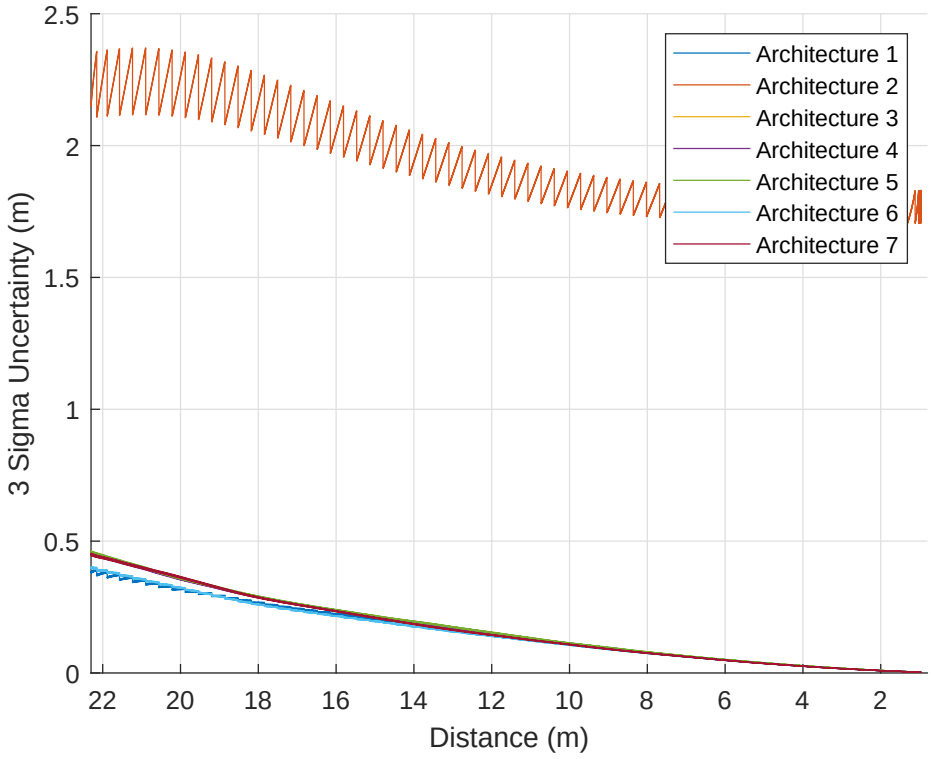

Fig. 7.14: Down Relative Position 3-Sigma Values Over True Distance.

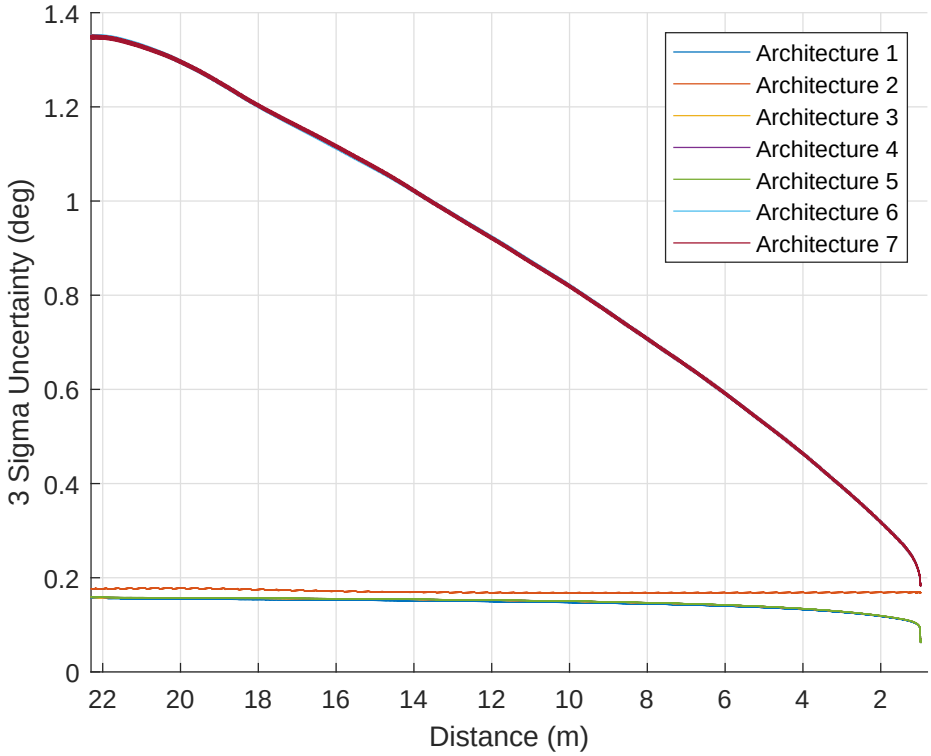

Fig. 7.15: X Axis Relative Attitude 3-Sigma Values Over True Distance.

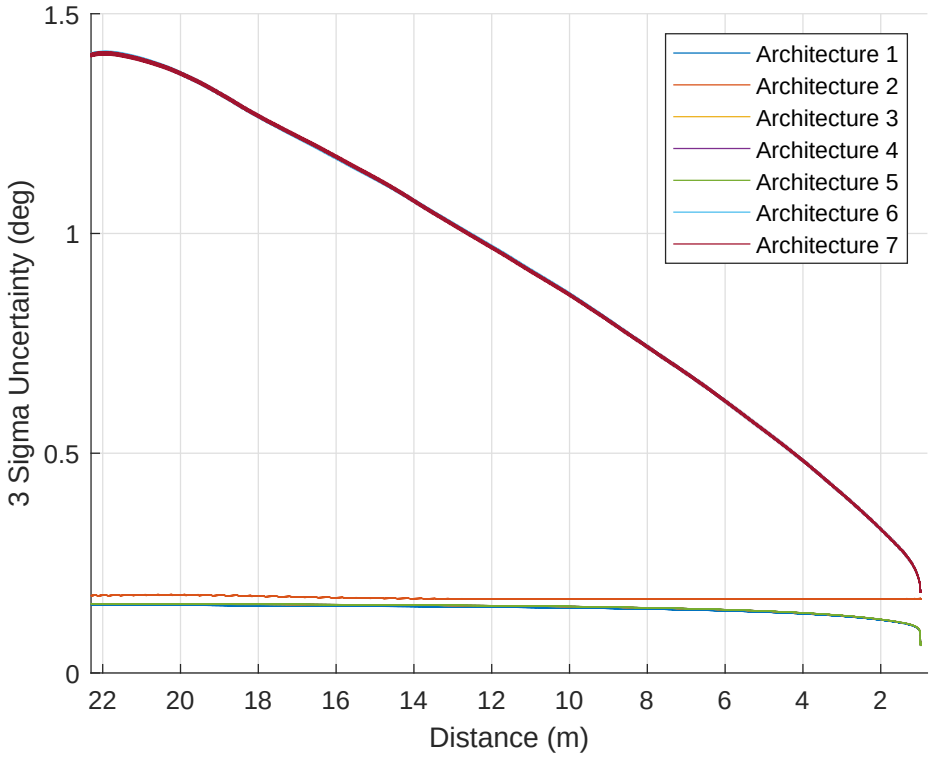

Fig. 7.16: Y Axis Relative Attitude 3-Sigma Values Over True Distance.

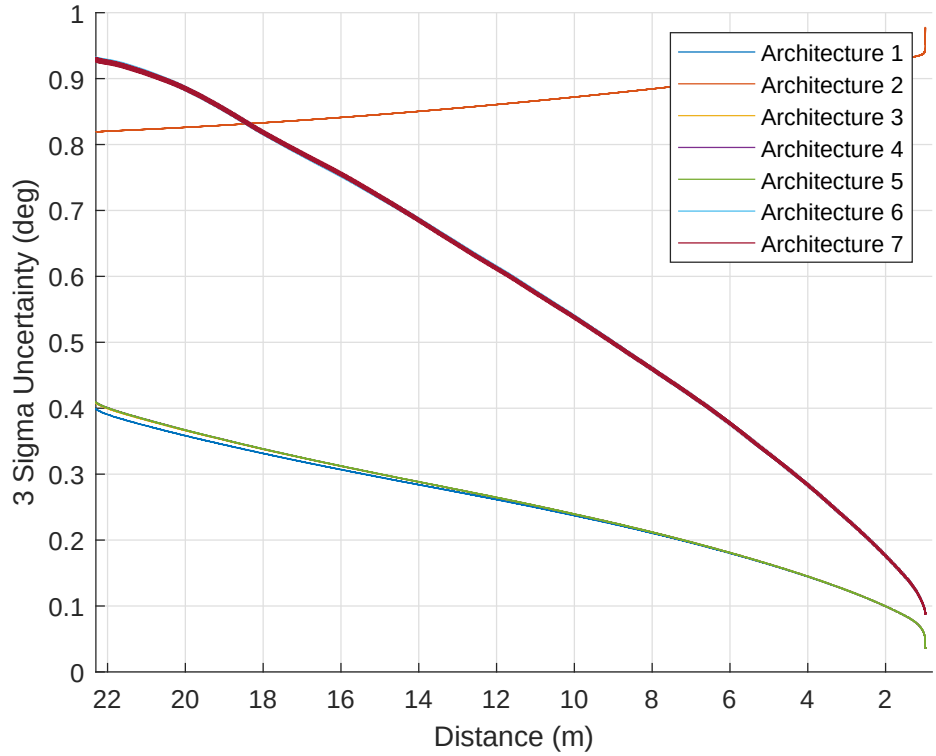

Fig. 7.17: Z Axis 3-Sigma Values Over True Distance.

7.4 Discussion

Looking first at section 7.1 to validate the implemented navigation filters, the Monte Carlo and residual plots look as expected. It appears that the vast majority of each of the relative estimation errors remained within the three standard deviation bounds throughout the whole simulation time. This is what the plots are expected to look like because about 99% of the estimation errors should remain within those bounds. The measurement residuals also remained within the 3-sigma residual covariance bounds. The distribution of measurement residuals also appeared to be zero-mean, white, and Gaussian as expected. These results support the claim that the implemented navigation filters used for the architecture study were implemented correctly.

7.4.1 Steady State Uncertainty Discussion

Section 7.2 arguably presents the most important results of this study. The tabulated results in this section give steady state 3-sigma uncertainty values for each architecture and IMU grade at the point where the two aircraft are the closest. Since this study is attempting to focus on close-proximity relative navigation, these results are important because they show how well relative attitude and position can be estimated at a close 1-meter distance.

As expected, architecture 1 that uses all of the sensors on both aircraft has the best performance according to these metrics. For the lowest IMU grade, architecture 1 is capable of estimating 99% of the time the relative north, east, and down positions to within 6.58 mm, 6.36 mm, and 13.5 mm respectively. With the highest grade IMU, architecture 1 is capable of estimating the relative north, east, and down positions to within 4.49 mm, 3.17 mm, and 8.74 mm respectively. Attitude estimates using architecture 1 with the lowest grade IMU were found to be within 0.124, 0.126, and 0.115 degrees respectively.

Deciding whether any of these architectures are appropriate for close-proximity missions depends on the specific mission requirements, but the fact that each architecture has attitude estimate errors well below 1 degree supports the idea that each architecture can satisfy the majority of attitude estimate performance requirements. The position estimate performance is more likely to disqualify some architectures for certain close-proximity, such

as architecture 2 with maximum north, east, and down position errors of 30.4 cm, 30.4 cm, and 1.7 m.

Architecture 2 is by far the lowest performing architecture. The GPS error model implemented in this study was chosen to be analogous with a low-quality, hobby-grade GPS with only 4 satellites available at a time. Further studies could be performed to explore how these navigation solutions would perform with higher grade GPS sensors, but the GPS model used in this study highlights how well systems with LOS measurements perform even with low grade GPS sensors or even in GPS-denied environments. Architecture two's low performance is attributed to the fact that it is the only architecture that does not process LOS measurements and so it must rely on the low-quality GPS sensor for measurement corrections.

Architecture 6 and 7 are particularly attractive architectures because they do not rely on any communication between the two aircraft. This reduces the complexity of the navigation system tremendously when the system is implemented on actual hardware. These two architectures have an extremely similar performance with architecture 6 outperforming architecture 7 on a scale of sub-millimeters for position and sub-millidegrees for attitude. Architectures 6 and 7 outperform architectures 2 and 5, and perform at a similar level to architectures 1, 3, and 4. Since architecture 6 performs on such a similar level to the other highest-performing architectures but with significantly less system complexity, architecture 6 seems like a good choice for the close-proximity mission simulated in this work. The performance achieved for this architecture is dependent on the time constant and steady-state covariance parameters of the first order Markov process chosen to propagate the leader's states. Future studies should explore how this architecture's performance varies with these parameters.

Conclusions can also be drawn form how the uncertainty metrics improve when the IMU grade varies. The uncertainty values for each relative state does indeed improve as better performing IMUs are selected as is expected, except for relative attitude with architectures 6 and 7. The provided tables show no performance improvement within the

significant figures provided. Relative attitude is insensitive to IMU noise parameters for architectures 6 and 7 because there is no IMU information from the leader aircraft. The IMU located on the follower provides absolute attitude information for that aircraft, but relative attitude information is not available. Relative information for these architectures can only be obtained from the LOS measurements. Architectures 1 through 5, however, have IMU sensors on both aircraft and so are able to obtain relative attitude information from the IMUs and the LOS measurements. The relative attitude for architectures 1 through 5 improves by about an order of magnitude as the IMU grade is improved.

7.4.2 Relative State Uncertainties with Respect to Distance Discussion

The plots in section 7.3 presents the estimate uncertainty over a range of distances obtained from the glide-slope maneuver simulation. The conclusions and observations of the previous subsection seem to mostly hold over the entire set of distances in this study. For example, architecture 1 has the lowest estimation error over the entire set of distances. Also, architecture 2 is the worst performing by far and the other architectures each perform at a similar level to each other.

An insight given by these plots that was not apparent from the tabulated results is that the attitude uncertainty for architecture 7 is significantly higher than the other architectures.

CHAPTER 8
CONCLUSION AND FUTURE WORK

8.1 Conclusion

In this thesis, research to develop guidance, navigation, and control algorithms for a package exchange maneuver of two quad-rotor aircraft was proposed. Benefits and challenges of the chosen topic were discussed. Literature for aerial robotics, trajectory tracking, and relative navigation were then reviewed. Research tool development was also discussed.

A full pose tracking controller was successfully designed and implemented in simulation to achieve synchronization of two multi-rotor robots. A simulator for multi-rotor drones was first created to simulate the dynamics and kinematics of two drones. The non-linear equations were presented in this paper. Linear models were derived for the system and used to tune an LQR controller. The LQR controller was used with a differential flatness command feed-forward feature to achieve desired performance. It was found that synchronization error was low enough for the full pose states that the design could be suitable for some applications even without a manipulator. This is expected to only hold for systems with similar dynamics. Adding a 2-DOF manipulator greatly reduced the roll and pitch angle errors.

A navigation architecture study for close-proximity missions was performed. EKFs were designed for each architecture and each filter was simulated using an identical trajectory. It was found that the architecture with the best performance had access to GPS sensors on each aircraft, IMU sensors on each aircraft, and LOS measurements from a camera on the follower aircraft. It was also found that each architecture that used LOS measurements performed at a similar, high performance level. Finally, it was discovered that the only two architectures that do not require a communication link between the two aircraft estimated states with a very low error. These two architectures' state estimate errors were not significantly different

to the highest performing architectures. Without the need for a communication link, these two architectures are much simpler to implement and are therefore attractive architecture options.

8.2 Future Work

When aircraft fly in close-proximity, the wake of an aircraft can cause powerful disturbances on the other. This wake disturbance was ignored in this work, but this large disturbance could make it more difficult for two aircraft to be synchronized. Before implementing the controller designed in chapter 4 onto hardware, disturbance rejection analysis should first be done. In order to analyze the wake disturbances, work would need to be done in modeling those disturbances as well.

The disturbance rejection analysis may show that improvements need to be made on the tracking controller design, such as adding an integrator to the LQR controller. Developing and adding a disturbance observer to the system would most likely improve performance in the presence of wake disturbance and may even be necessary to avoid collisions.

Once disturbances have been considered, the tracking controller could be implemented on hardware. Experimental test flights would be used to validate the synchronization control design.

Implementing the navigation systems on a hardware platform and generating flight data would validate this study for real world applications. Since this study found architectures that work well in simulation, a hardware implementation could further validate these architectures so that they could be used in close-proximity missions. The architecture study in this work could also be expanded to other architectures and GPS sensor grades.

Other studies that could contribute to this work include alternative controller designs such as Model Predictive Control (MPC) based control design that leverages propeller theory, on-line estimation of Markov model parameters, and modern linear covariance analysis for go/no go autonomous decision making.

REFERENCES

[1] F. Augugliaro, S. Lupashin, M. Hamer, C. Male, M. Hehn, M. W. Mueller, J. S. Willmann, F. Gramazio, M. Kohler, and R. D'Andrea, "The flight assembled architecture installation: Cooperative construction with flying machines," *IEEE Control Syst. Mag.*, vol. CS-34, no. 4, pp. 46–64, 2014. [Online]. Available: https://ieeexplore.ieee.org/document/8059875

[2] M. Orsag, C. Korpela, S. Bogdan, and P. Oh, "Dexterous aerial robots – mobile manipulation using unmanned aerial systems," *IEEE Trans. Robot.*, vol. RO-33, no. 6, pp. 1453–1466, 2017. [Online]. Available: https://ieeexplore.ieee.org/document/8059875/references#references

[3] D. Mellinger, Q. Lindsey, M. Shomin, and V. Kumar, "Design, modeling, estimation and control for aerial grasping and manipulation," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, Dec. 2011.

[4] K. Kondak, K. Krieger, , A. ALbu-Schaeffer, marc Schwarzbach, M. Laiacker, I. Maza, A. Rodriquez-Castano, and A. Ollero, "Closed-loop behavior of an autonomous helicopter equipped with a robotic arm for aerial manipulation tasks," *International Journal of Advanced Robotic Systems*, vol. 10, no. 2, 2013.

[5] S. Kim, S. Choi, and H. J. Kim, "Aerial manipulation using a quadrotor with a two dof robotic arm," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov. 2013.

[6] D. B. Wilson, A. Goktogan, and S. Sukkarieh, "Guidance and navigation for uav airborne docking," *Robotics: Science and Systems*, 2015.

[7] W. Mao and F. O. Eke, "A survey of the dynamics and control of aircraft during aerial refueling," *Nonlinear Dynamics and Systems Theory*, vol. 8, no. 4, pp. 375–388, 2008.

[8] M. Colton, L. Sun, D. Carlson, and R. Beard, "Multi-vehicle dynamics and control for aerial recovery of micro air vehicles," *International Journal of Vehicle Autonomous Systems*, vol. 9, no. 1/2, 2011.

[9] M. Bernard and K. Kondak, "Generic slung load transportation system using small size helicopters," in *Proc. IEEE/RSJ International Conference on Robotics and Automation*, May 2009.

[10] A. P. Aguiar and J. P. Hespanha, "Trajectory-tracking and path-following of underactuated autonomous vehicles with parametric modling uncertainty," *IEEE Trans. Autom. Control*, vol. 52, no. 8, pp. 1362–1379, 2007.

[11] A. Franchi, R. Carli, D. Bicego, and M. Ryll, "Full-pose tracking control for aerial robotic systems with laterally bounded input force," *IEEE Trans. Robot.*, vol. RO-34, no. 2, pp. 534–541, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8291488

[12] J. Ferrin, R. Leishman, R. Beard, and T. McLain, "Differential flatness based control of a rotorcraft for aggressive maneuvers," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, Dec. 2011.

[13] H. Choi and Y. Kim, "Uav guidance using a monocular-vision sensor for aerial target tracking," *Control Engineering Practice*, vol. 22, pp. 10–19, 2014.

[14] G. P. Tournier, M. Valenti, and J. P. How, "Estimation and control of a quadrotor vehicle using monocular vision and moire patterns," in *Proc. AIAA/Guidance, Navigation, and Control Conference*, Aug. 2006.

[15] J. L. Crassidis, R. Alonso, and J. L. Junkins, "Optimal attitude and position determination from line-of-sight measurements," *Journal of the Astronautical Sciences*, vol. 48, no. 2, 2000.

[16] S. B. Robinson and J. A. Christian, "Pattern design for 3d point matching," *Journal of the Institute of Navigation*, vol. 62, no. 3, pp. 189–203, 2015.

[17] OpenCV team. (2019) Opencv. [Online]. Available: https://opencv.org/

[18] X. Wang, N. Cui, and J. Guo, "Ins/visnav/gps relative navigation system for uav," *Aerospace Science and Technology*, vol. 28, no. 1, pp. 242–248, 2013.

[19] W. R. Williamson, G. J. Glenn, V. T. Dang, J. Speyer, S. M. Stecko, and J. M. Takacs, "Sensor fusion applied to autonomous aerial refueling," *Journal of Guidance, Control, and Dynamics*, vol. 32, no. 1, pp. 262–275, 2009.

[20] A. M. Fosbury and J. L. Crassidis, "Relative navigation of air vehicles," *Journal of Guidance, Control, and Dynamics*, vol. 31, no. 4, pp. 824–834, 2008.

[21] S.-G. Kim, J. L. Crassidis, Y. Cheng, A. M. Fosbury, and J. L. Junkins, "Kalman filtering for relative spacecraft attitude and position estimation," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 1, pp. 133–143, 2007.

[22] R. W. Beard and T. W. McLain, *Small Unmanned Aircraft Theory and Practice*. Princeton, NJ: Princeton University Press, 2012, ch. 3–4.

[23] P. Martin, R. Murray, and P. Rouchon. (1997) Flat systems. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00472051/document

[24] R. Murray, M. Rathinam, and W. Sluis, "Differential flatness of mechanical control systems," in *ASME Int'l Mech Eng Congress and Expo.*, Nov. 1995.

[25] J. Hespanha, *Linear Systems Theory*. Princeton, NJ: Princeton University Press, 2009, ch. 3–4.

[26] Mathworks. (2019) quadcopter project. [Online]. Available: https://www.mathworks.com/help/aeroblks/quadcopter-project.html?searchHighlight=quadcopter%20simulation&s_tid=doc_srchtitle

[27] Sensonor. (2018) Stim300 inertia measurement unit. [Online]. Available: https://www.sensonor.com/products/inertial-measurement-units/stim300/

[28] Northrop Grumman. (2013) Ln-200 fog family. [Online]. Available: https://www.northropgrumman.com/Capabilities/LN200FOG/Documents/ln200.pdf

[29] Honeywell. (2016) Hg9900 inertial measurement unit. [Online]. Available: https://aerospace.honeywell.com/en/~/media/aerospace/files/brochures/n61-1638-000-000-hg9900inertialmeasurementunit-bro.pdf

[30] M. S. Grewal, A. P. Andrews, and C. G. Bartone, *Global Navigation Satellite Systems, Inertial Navigaion, and Integration.* Hoboken, NJ: Wiley, 2013, ch. 2, pp. 43–46.

[31] P. Misra and P. Enge, *Global Positioning System.* Lincoln, MA: Ganga-Jamuna Press, 2011, ch. 5, p. 187.

APPENDICES

APPENDIX A

Derivations

A.1 Velocity and Attitude Quaternion Kinematics Linearization

This appendix documents the linearization of the velocity kinematics and attitude quaternion kinematics presented in chapter 5.

Velocity Kinematics Linearization

Beginning with equation 5.2:

$$\dot{\underline{\hat{v}}}^{ned} + \delta\dot{\underline{v}}^{ned} = R_b^{ned}\left([I - (\delta\underline{\theta}_b\times)]\right)^T\left(\underline{\tilde{\nu}}^b - \underline{\hat{b}}_{accel} - \delta\underline{\mathrm{b}}_{accel} + \underline{\mathrm{n}}_\nu\right) + \mathrm{g}^{ned}. \tag{A.1}$$

Distributing the transpose yields

$$\dot{\underline{\hat{v}}}^{ned} + \delta\dot{\underline{v}}^{ned} = R_b^{ned}\left([I + (\delta\underline{\theta}_b\times)]\right)^T\left(\underline{\tilde{\nu}}^b - \underline{\hat{b}}_{accel} - \delta\underline{\mathrm{b}}_{accel} + \underline{\mathrm{n}}_\nu\right) + \mathrm{g}^{ned}. \tag{A.2}$$

Expanding yields

$$\dot{\underline{\hat{v}}}^{ned} + \delta\dot{\underline{v}}^{ned} = R_b^{ned}(\underline{\tilde{\nu}}^b - \underline{\hat{b}}_{accel} - \delta\underline{\mathrm{b}}_{accel} + \underline{\mathrm{n}}_\nu) + R_b^{ned}\left(\delta\underline{\theta}_b\times\right)\left(\underline{\tilde{\nu}}^b - \underline{\hat{b}}_{accel} - \delta\underline{\mathrm{b}}_{accel} + \underline{\mathrm{n}}_\nu\right) + \mathrm{g}^{ned}. \tag{A.3}$$

Subtracting the estimated $\dot{\underline{\hat{v}}}^{ned}$ from both sides yields

$$\delta\dot{\underline{v}}^{ned} = R_b^{ned}(-\delta\underline{\mathrm{b}}_{accel} + \underline{\mathrm{n}}_\nu) + R_b^{ned}\left(\delta\underline{\theta}_b\times\right)\left(\underline{\tilde{\nu}}^b - \underline{\hat{b}}_{accel} - \delta\underline{\mathrm{b}}_{accel} + \underline{\mathrm{n}}_\nu\right). \tag{A.4}$$

Reversing the cross product yields

$$\delta\dot{\underline{v}}^{ned} = R_b^{ned}(-\delta\underline{\mathrm{b}}_{accel} + \underline{\mathrm{n}}_\nu) - R_b^{ned}(\underline{\tilde{\nu}}^b - \underline{\hat{b}}_{accel} - \delta\underline{\mathrm{b}}_{accel} + \underline{\mathrm{n}}_\nu)\times\delta\underline{\theta}_b. \tag{A.5}$$

Neglecting second order terms yields

$$\delta\dot{\underline{v}}^{ned} = R_b^{ned}(-\delta\underline{\mathrm{b}}_{accel} + \underline{\mathrm{n}}_{\nu}) - R_b^{ned}(\tilde{\underline{\nu}}^b - \hat{\underline{b}}_{accel}) \times \delta\underline{\theta}_b. \tag{A.6}$$

Rearranging yields the final equation:

$$\delta\dot{\underline{v}}^{ned} = -R_b^{ned}(\tilde{\underline{\nu}}^b - \hat{\underline{b}}_{accel}) \times \delta\underline{\theta}_b - R_b^{ned}\delta\underline{\mathrm{b}}_{accel} + R_b^{ned}\underline{\mathrm{n}}_{\nu}. \tag{A.7}$$

Attitude Quaternion Kinematics Linearization

Substituting the definition of the estimated attitude quaternion into equation 5.3 yields

$$\dot{q}_{ned}^b = \delta\dot{q}_{\hat{b}}^b \otimes q_{ned}^{\hat{b}} + \delta q_{\hat{b}}^b \otimes \frac{1}{2}\begin{bmatrix} \tilde{\underline{\omega}}^b - \hat{\underline{b}}_{gyro} \\ 0 \end{bmatrix} \otimes \dot{q}_{ned}^{\hat{b}}. \tag{A.8}$$

The right-hand-side of equation 5.3.1 can also be expanded about the current estimate by substituting equations 5.3.2 and 5.3.2 to yield

$$\dot{q}_{ned}^b = \frac{1}{2}\begin{bmatrix} \tilde{\underline{\omega}}^b - \hat{\underline{b}}_{gyro} - \delta\underline{\mathrm{b}}_{gyro} + \underline{\mathrm{n}}_{\omega} \\ 0 \end{bmatrix} \otimes \delta q_{\hat{b}}^b \otimes q_{ned}^{\hat{b}}. \tag{A.9}$$

The objective of the rest of this section is to isolate the $\delta\dot{q}_{\hat{b}}^b$. Setting the previous two equations equal to each other and right-quaternion-multiplying each side by $\left(q_{ned}^{\hat{b}}\right)^*$ yields

$$\delta\dot{q}_{\hat{b}}^b + \delta q_{\hat{b}}^b \otimes \frac{1}{2}\begin{bmatrix} \tilde{\underline{\omega}}^b - \hat{\underline{b}}_{gyro} \\ 0 \end{bmatrix} = \frac{1}{2}\begin{bmatrix} \tilde{\underline{\omega}}^b - \hat{\underline{b}}_{gyro} - \delta\underline{\mathrm{b}}_{gyro} + \underline{\mathrm{n}}_{\omega} \\ 0 \end{bmatrix} \otimes \delta q_{\hat{b}}^b. \tag{A.10}$$

Substituting the definition for the angular error from equation 5.3.2 yields

$$\begin{bmatrix} \frac{\delta\dot{\underline{\theta}}_b}{2} \\ 1 \end{bmatrix} + \begin{bmatrix} \frac{\delta\underline{\theta}_b}{2} \\ 1 \end{bmatrix} \otimes \frac{1}{2} \begin{bmatrix} \tilde{\underline{\omega}}^b - \hat{\underline{b}}_{gyro} \\ 0 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} \tilde{\underline{\omega}}^b - \hat{\underline{b}}_{gyro} - \delta\underline{\mathrm{b}}_{gyro} + \underline{\mathrm{n}}_\omega \\ 0 \end{bmatrix} \otimes \begin{bmatrix} \frac{\delta\underline{\theta}_b}{2} \\ 1 \end{bmatrix}. \tag{A.11}$$

Evaluating the quaternion multiplications yields

$$\frac{1}{2} \begin{bmatrix} \delta\dot{\underline{\theta}}_b + \tilde{\underline{\omega}}^b - \hat{\underline{b}}_{gyro} + \left(\tilde{\underline{\omega}}^b - \hat{\underline{b}}_{gyro}\right) \times \frac{\delta\underline{\theta}_b}{2} \\ -\left(\tilde{\underline{\omega}}^b - \hat{\underline{b}}_{gyro}\right) \cdot \frac{\delta\underline{\theta}_b}{2} \end{bmatrix} =$$

$$\frac{1}{2} \begin{bmatrix} \tilde{\underline{\omega}}^b - \hat{\underline{b}}_{gyro} - \delta\underline{b}_{gyro} - \underline{\mathrm{n}}_\omega + \frac{\delta\underline{\theta}_b}{2} \times \left(\tilde{\underline{\omega}}^b - \hat{\underline{b}}_{gyro} - \delta\underline{b}_{gyro} + \underline{\mathrm{n}}_\omega\right) \\ -\frac{\delta\underline{\theta}_b}{2} \cdot \left(\tilde{\underline{\omega}}^b - \hat{\underline{b}}_{gyro} - \delta\underline{b}_{gyro} + \underline{\mathrm{n}}_\omega\right) \end{bmatrix}. \tag{A.12}$$

Eliminating like terms and second-order terms yields the simpler form

$$\begin{bmatrix} \delta\dot{\underline{\theta}}_b + \left(\tilde{\underline{\omega}}^b - \hat{\underline{b}}_{gyro}\right) \times \frac{\delta\underline{\theta}_b}{2} \\ 0 \end{bmatrix} = \begin{bmatrix} -\delta\underline{b}_{gyro} + \underline{\mathrm{n}}_\omega + \frac{\delta\underline{\theta}_b}{2} \times \left(\tilde{\underline{\omega}}^b - \hat{\underline{b}}_{gyro}\right) \\ 0 \end{bmatrix} \tag{A.13}$$

Equating the vector portion of the quaternions and rearranging yields the final form

$$\delta\dot{\underline{\theta}}_b + \left(\tilde{\underline{\omega}}^b - \hat{\underline{b}}_{gyro}\right) \times \frac{\delta\underline{\theta}_b}{2} = -\delta\underline{b}_{gyro} + \underline{\mathrm{n}}_\omega + \frac{\delta\underline{\theta}_b}{2} \times \left(\tilde{\underline{\omega}}^b - \hat{\underline{b}}_{gyro}\right) \tag{A.14}$$

$$\delta\dot{\underline{\theta}}_b - \frac{\delta\underline{\theta}_b}{2} \times \left(\tilde{\underline{\omega}}^b - \hat{\underline{b}}_{gyro}\right) = -\delta\underline{b}_{gyro} + \underline{\mathrm{n}}_\omega + \frac{\delta\underline{\theta}_b}{2} \times \left(\tilde{\underline{\omega}}^b - \hat{\underline{b}}_{gyro}\right) \tag{A.15}$$

$$\delta\dot{\underline{\theta}}_b = -\delta\underline{b}_{gyro} + \underline{\mathrm{n}}_\omega + \frac{\delta\underline{\theta}_b}{2} \times \left(\tilde{\underline{\omega}}^b - \hat{\underline{b}}_{gyro}\right) + \frac{\delta\underline{\theta}_b}{2} \times \left(\tilde{\underline{\omega}}^b - \hat{\underline{b}}_{gyro}\right) \tag{A.16}$$

$$\delta\dot{\underline{\theta}}_b = -\delta\underline{b}_{gyro} + \underline{\mathrm{n}}_\omega + \delta\underline{\theta}_b \times \left(\tilde{\underline{\omega}}^b - \hat{\underline{b}}_{gyro}\right) \tag{A.17}$$

$$\delta\dot{\underline{\theta}}_b = -\left(\underline{\tilde{\omega}}^b - \underline{\hat{b}}_{gyro}\right) \times \delta\underline{\theta}_b - \delta\underline{b}_{gyro} + \underline{\mathrm{n}}_\omega \tag{A.18}$$

A.2 LOS Measurement Model Linearization

This appendix gives the full derivation of the LOS measurement error vector δl as a function of the error state vector $\delta\underline{\mathrm{x}}$. Recall that the nonlinear LOS measurement equations with perturbations is given by

$$\hat{\underline{\mathrm{l}}} + \delta\underline{\mathrm{l}} = \left[I_{3x3} - (\delta\theta_{b_F}\times)\right] R_{ned}^{\hat{b}_F}\left(\hat{\underline{p}}_L^{ned} + \delta\underline{\mathrm{p}}_L^{ned} - \hat{\underline{p}}_F^{ned} - \delta\underline{\mathrm{p}}_F^{ned}\right)$$

$$+ \left[I_{3x3} - (\delta\theta_{b_F}\times)\right] R_{ned}^{\hat{b}_F} R_{\hat{b}_L}^{ned}\left[I_{3x3} - (\delta\theta_{b_L}\times)\right]^T \underline{\mathrm{p}}_{j,L}^{b_L} \tag{A.19}$$

Distributing the transpose, rearranging terms, and expanding terms, the equation obtains a new form

$$\hat{\underline{\mathrm{l}}} + \delta\underline{\mathrm{l}} = R_{ned}^{\hat{b}_F}\left(\hat{\underline{p}}_L^{ned} - \hat{\underline{p}}_F^{ned}\right) + R_{ned}^{\hat{b}_F}\left(\delta\underline{\mathrm{p}}_L^{ned} - \delta\underline{\mathrm{p}}_F^{ned}\right) - (\delta\theta_{b_F}\times) R_{ned}^{\hat{b}_F}\left(\hat{\underline{p}}_L^{ned} - \hat{\underline{p}}_F^{ned}\right) -$$

$$(\delta\theta_{b_F}\times) R_{ned}^{\hat{b}_F}\left(\delta\underline{\mathrm{p}}_L^{ned} - \delta\underline{\mathrm{p}}_F^{ned}\right) + \left[R_{ned}^{\hat{b}_F} R_{\hat{b}_L}^{ned} - (\delta\theta_{b_F}\times) R_{ned}^{\hat{b}_F} R_{\hat{b}_L}^{ned}\right]\left[I_{3x3} + (\delta\theta_{b_L}\times)\right]\underline{\mathrm{p}}_{j,L}^{b_L} \tag{A.20}$$

Eliminating higher order terms and further rearranging yields

$$\hat{\underline{\mathrm{l}}} + \delta\underline{\mathrm{l}} = R_{ned}^{\hat{b}_F}\left(\hat{\underline{p}}_L^{ned} - \hat{\underline{p}}_F^{ned}\right) + R_{ned}^{\hat{b}_F}\left(\delta\underline{\mathrm{p}}_L^{ned} - \delta\underline{\mathrm{p}}_F^{ned}\right) - (\delta\theta_{b_F}\times) R_{ned}^{\hat{b}_F}\left(\hat{\underline{p}}_L^{ned} - \hat{\underline{p}}_F^{ned}\right)$$

$$+ R_{ned}^{\hat{b}_F} R_{\hat{b}_L}^{ned}\underline{\mathrm{p}}_{j,L}^{b_L} - (\delta\theta_{b_F}\times) R_{ned}^{\hat{b}_F} R_{\hat{b}_L}^{ned}\underline{\mathrm{p}}_{j,L}^{b_L} + R_{ned}^{\hat{b}_F} R_{\hat{b}_L}^{ned}(\delta\theta_{b_L}\times)\underline{\mathrm{p}}_{j,L}^{b_L} \tag{A.21}$$

Using properties of cross products

$$\hat{\underline{\mathrm{l}}} + \delta\underline{\mathrm{l}} = R_{ned}^{\hat{b}_F}\left(\hat{\underline{p}}_L^{ned} - \hat{\underline{p}}_F^{ned}\right) + R_{ned}^{\hat{b}_F}\left(\delta\underline{\mathrm{p}}_L^{ned} - \delta\underline{\mathrm{p}}_F^{ned}\right) + \left[R_{ned}^{\hat{b}_F}\left(\hat{\underline{p}}_L^{ned} - \hat{\underline{p}}_F^{ned}\right)\times\right]\delta\theta_{b_F}$$

$$+R_{ned}^{\hat{b}_F}R_{\hat{b}_L}^{ned}\underline{\mathrm{p}}_{j,L}^{b_L} + \left[R_{ned}^{\hat{b}_F}R_{\hat{b}_L}^{ned}\underline{\mathrm{p}}_{j,L}^{b_L}\times\right]\delta\theta_{b_F} - R_{ned}^{\hat{b}_F}R_{\hat{b}_L}^{ned}\left(\underline{\mathrm{p}}_{j,L}^{b_L}\times\right)\delta\theta_{b_L} \tag{A.22}$$

Let the nominal marker position vector be defined as

$$\hat{\underline{\mathrm{l}}} = R_{ned}^{\hat{b}_F}\left(\hat{\underline{p}}_L^{ned} - \hat{\underline{p}}_F^{ned}\right) + R_{ned}^{\hat{b}_F}R_{\hat{b}_L}^{ned}\hat{\underline{p}}_{j,L}^{\hat{b}_L} \tag{A.23}$$

Canceling and combining like terms gives the final equation

$$\delta\underline{\mathrm{l}} = R_{ned}^{\hat{b}_F}\delta\underline{\mathrm{p}}_L^{ned} - R_{ned}^{\hat{b}_F}\delta\underline{\mathrm{p}}_F^{ned} + \left(\left[R_{ned}^{\hat{b}_F}\left(\hat{\underline{p}}_L^{ned} - \hat{\underline{p}}_F^{ned}\right)\times\right] + \left[R_{ned}^{\hat{b}_F}R_{\hat{b}_L}^{ned}\underline{\mathrm{p}}_{j,L}^{b_L}\times\right]\right)\delta\theta_{b_F}$$

$$-R_{ned}^{\hat{b}_F}R_{\hat{b}_L}^{ned}\left(\underline{\mathrm{p}}_{j,L}^{b_L}\times\right)\delta\theta_{b_L} \tag{A.24}$$

APPENDIX B

Additional Plots

B.1 Monte Carlo Hair Plots for Architecture 2 with Medium-Grade IMU

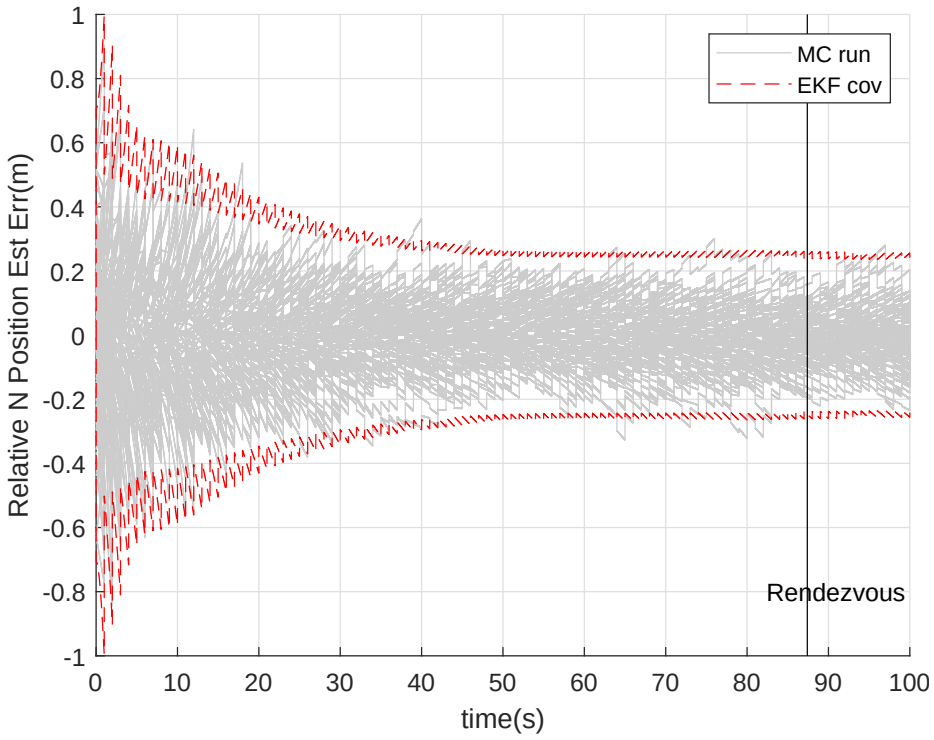

Fig. B.1: Architecture 2 Monte Carlo Analysis of Relative North Position.

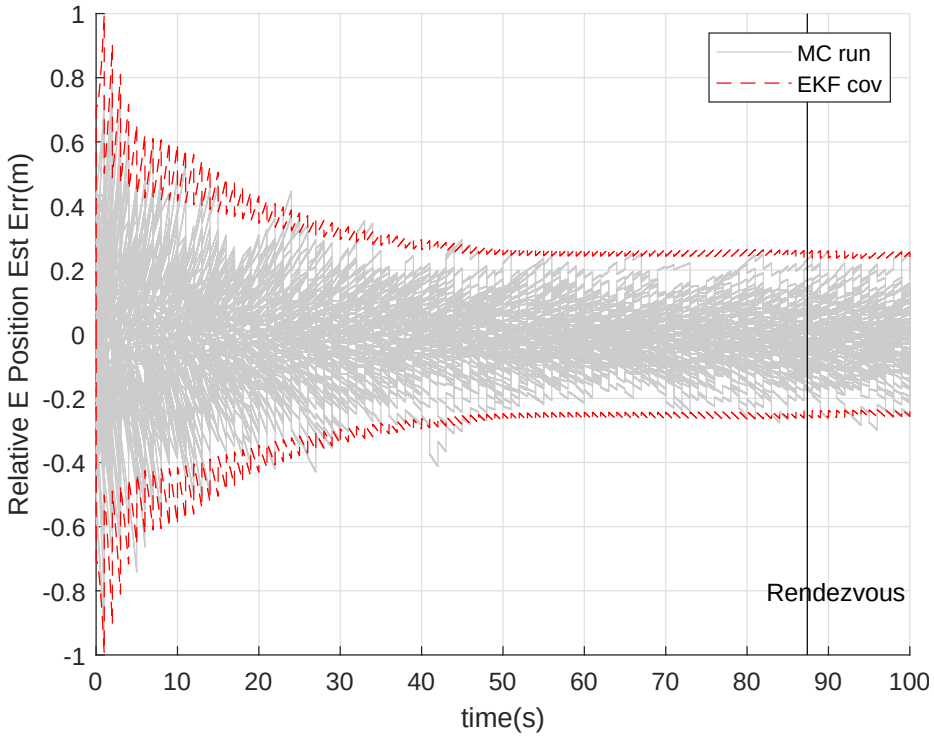

Fig. B.2: Architecture 2 Monte Carlo Analysis of Relative East Position.

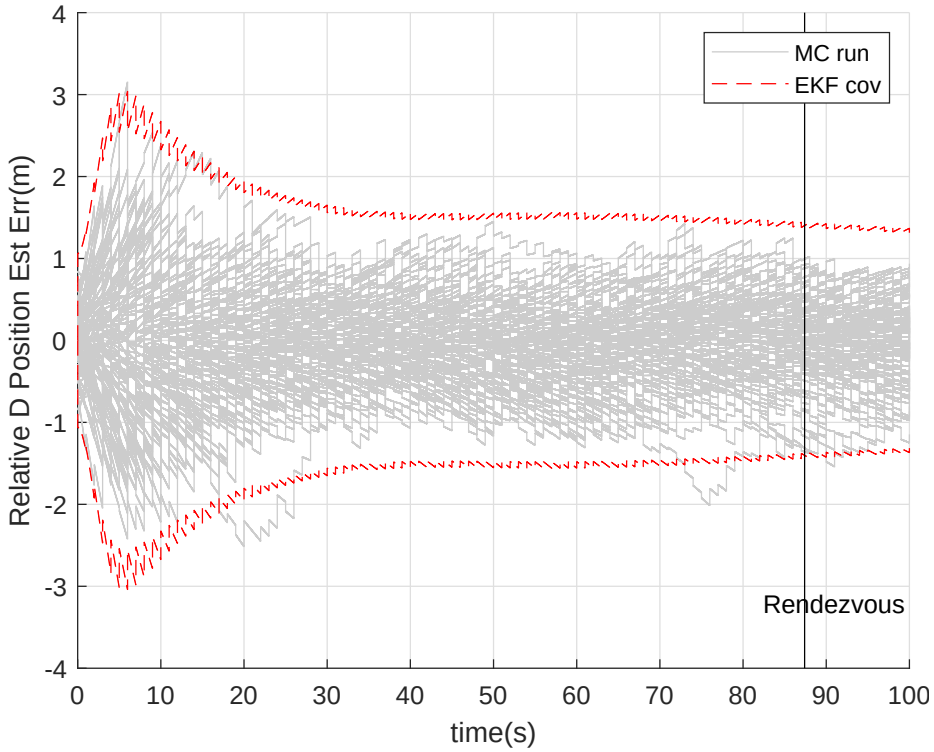

Fig. B.3: Architecture 2 Monte Carlo Analysis of Relative Down Position.

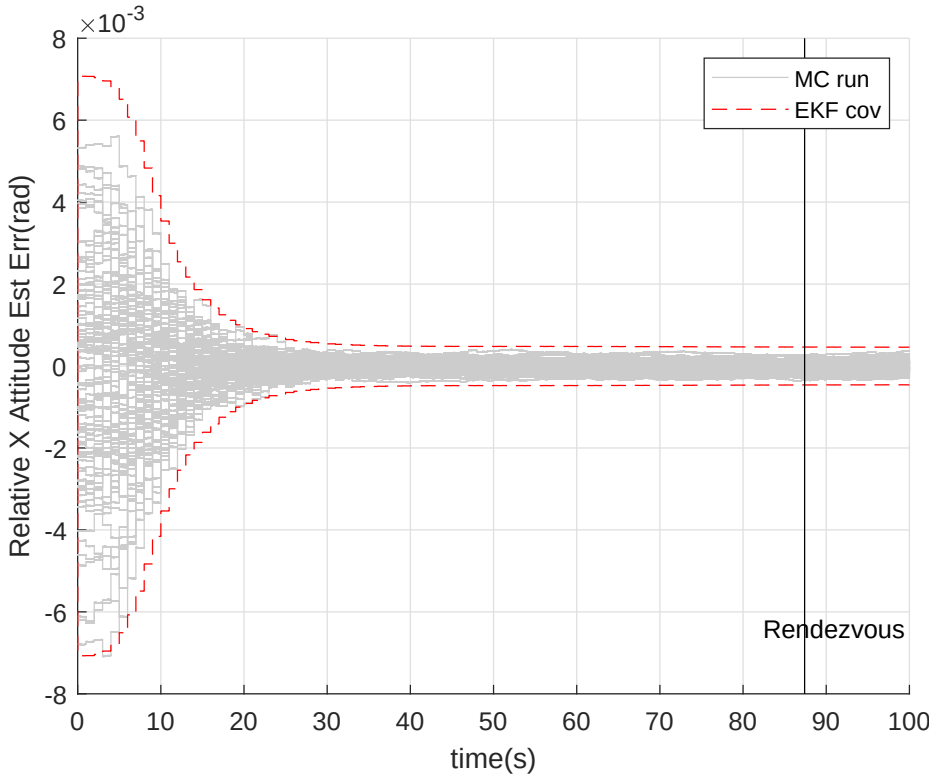

Fig. B.4: Architecture 2 Monte Carlo Analysis of Relative X Axis Attitude.

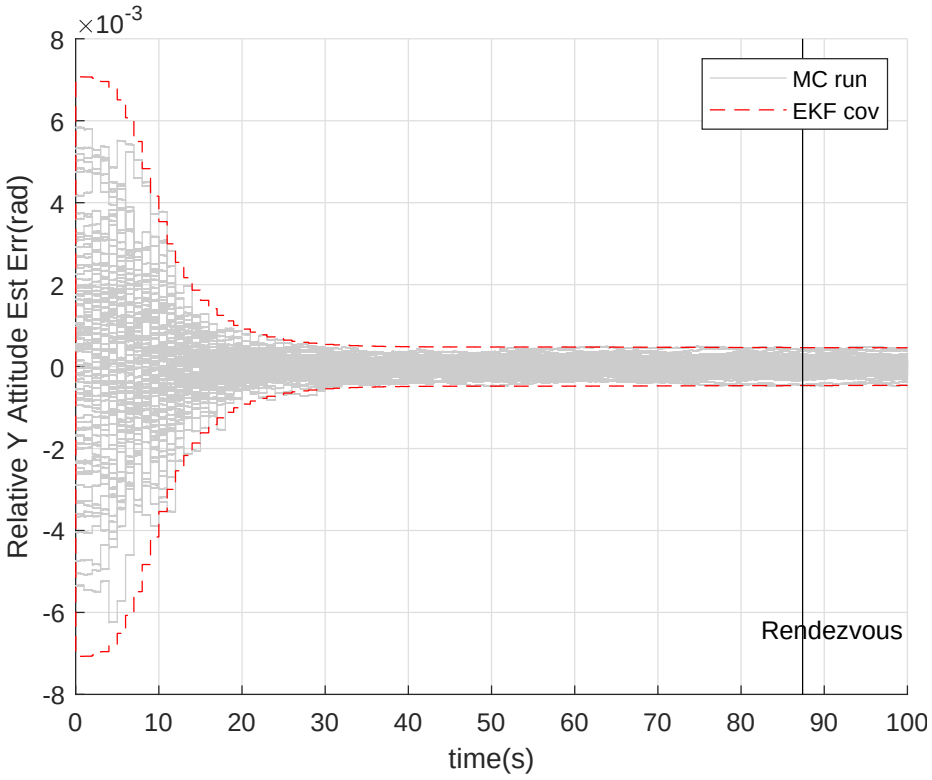

Fig. B.5: Architecture 2 Monte Carlo Analysis of Relative Y Axis Attitude.

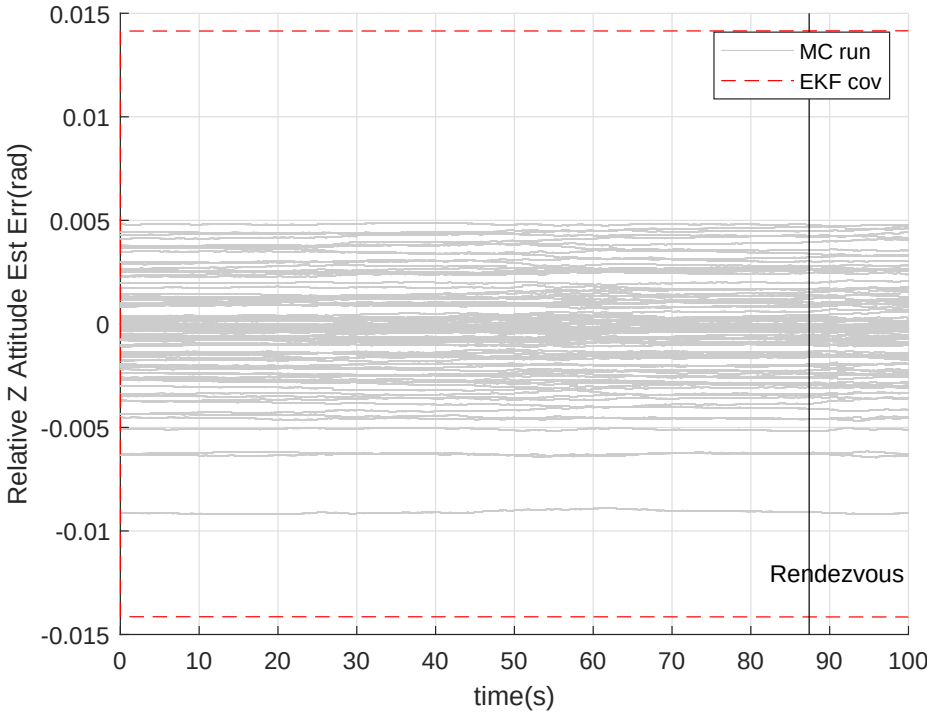

Fig. B.6: Architecture 2 Monte Carlo Analysis of Relative Z Axis Attitude.

B.2 Monte Carlo Hair Plots for Architecture 3 with Medium-Grade IMU

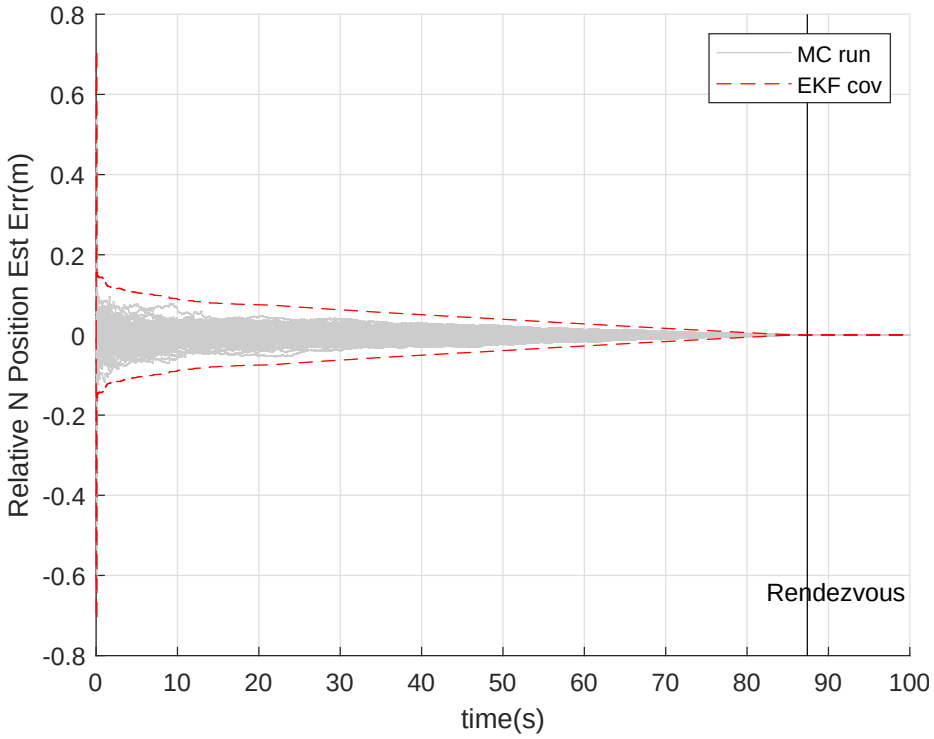

Fig. B.7: Architecture 3 Monte Carlo Analysis of Relative North Position.

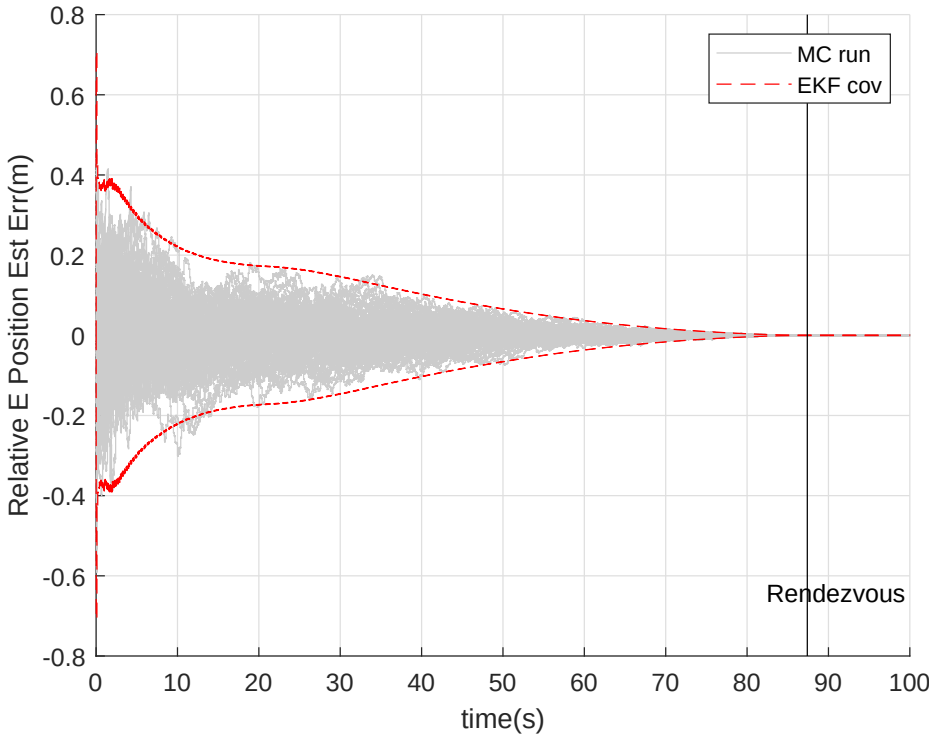

Fig. B.8: Architecture 3 Monte Carlo Analysis of Relative East Position.

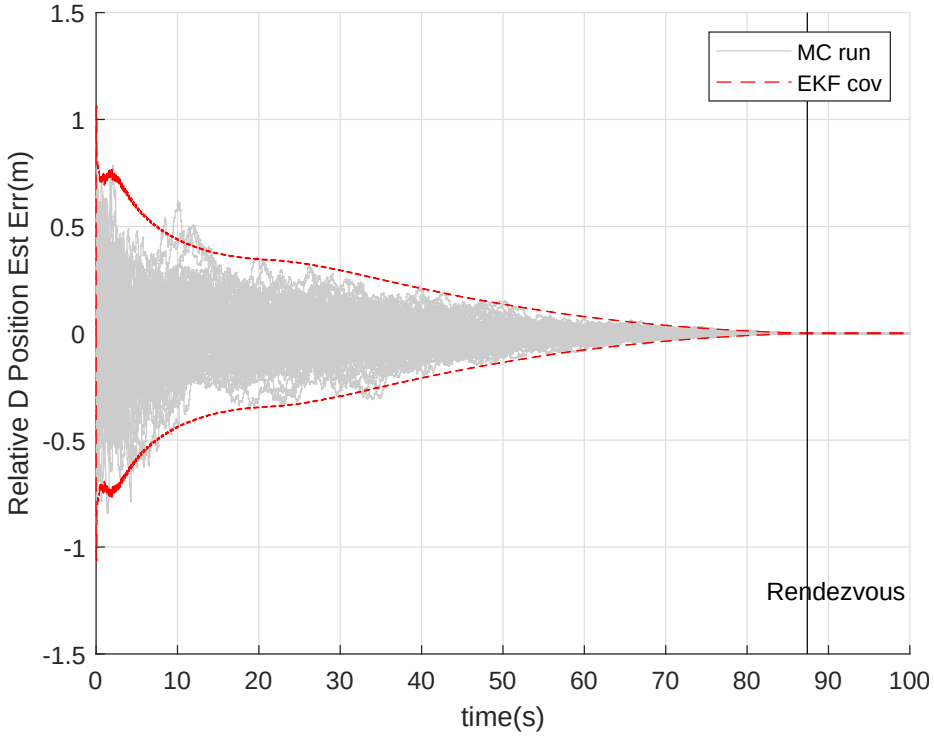

Fig. B.9: Architecture 3 Monte Carlo Analysis of Relative Down Position.

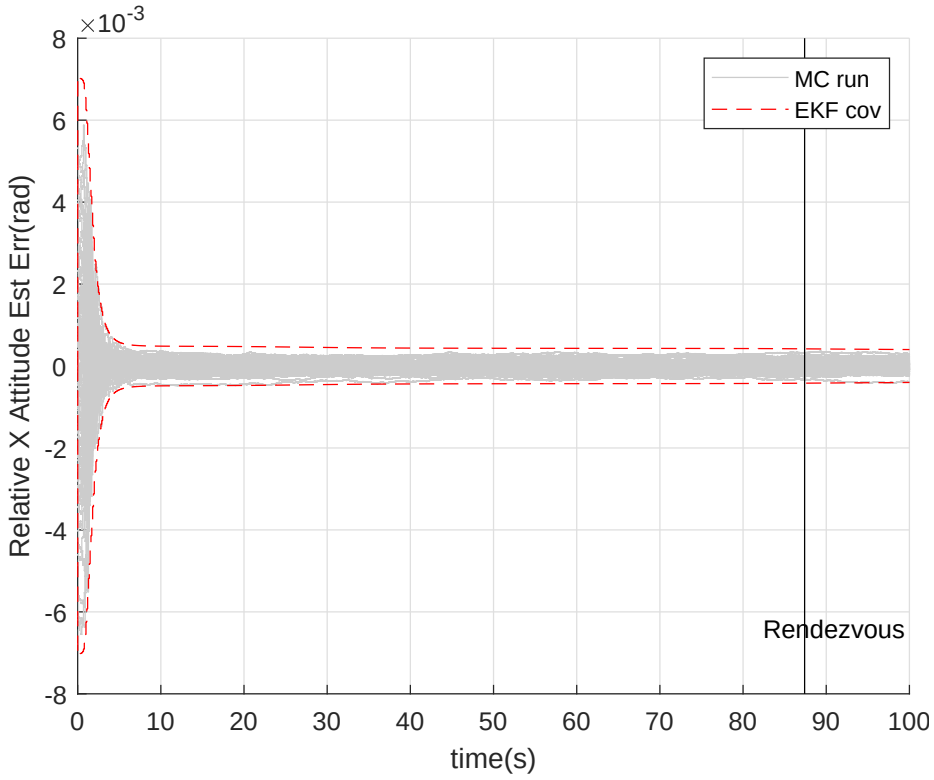

Fig. B.10: Architecture 3 Monte Carlo Analysis of Relative X Axis Attitude.

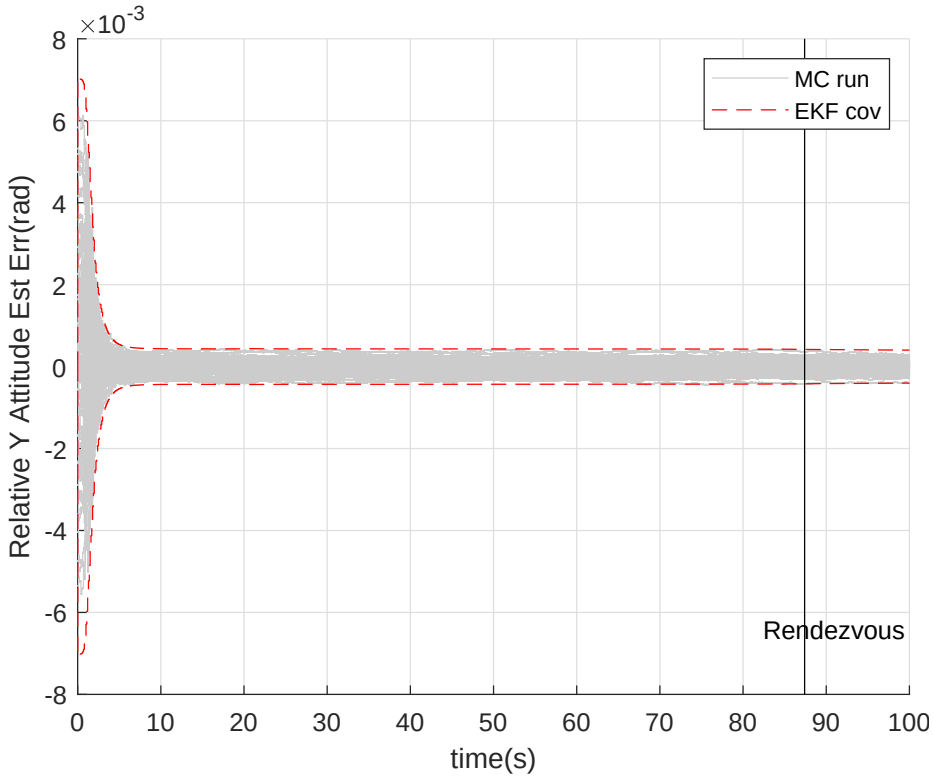

Fig. B.11: Architecture 3 Monte Carlo Analysis of Relative Y Axis Attitude.

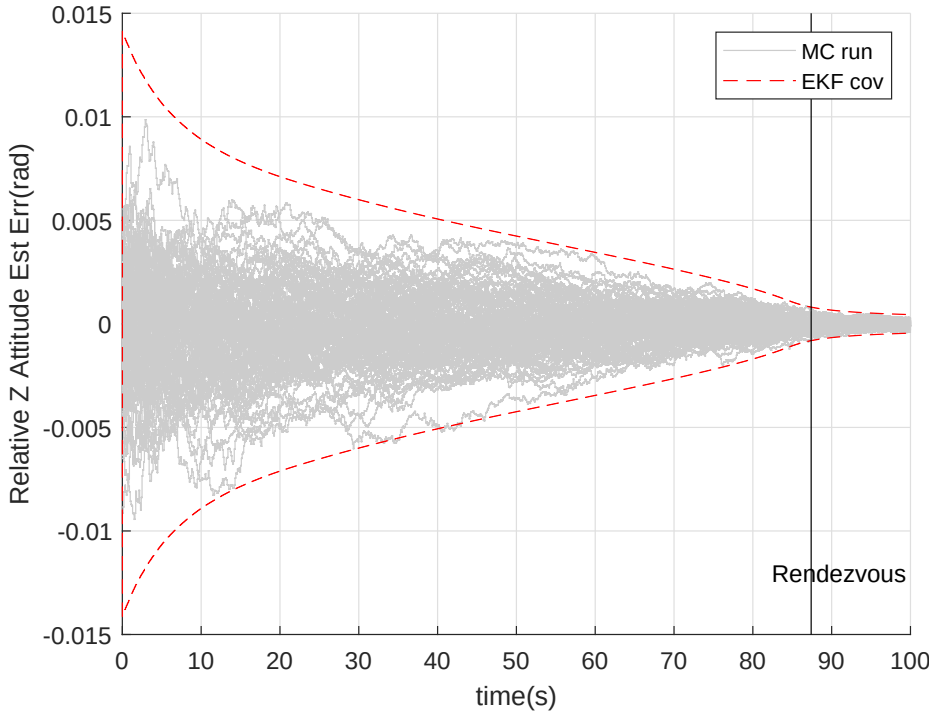

Fig. B.12: Architecture 3 Monte Carlo Analysis of Relative Z Axis Attitude.

B.3 Monte Carlo Hair Plots for Architecture 4 with Medium-Grade IMU

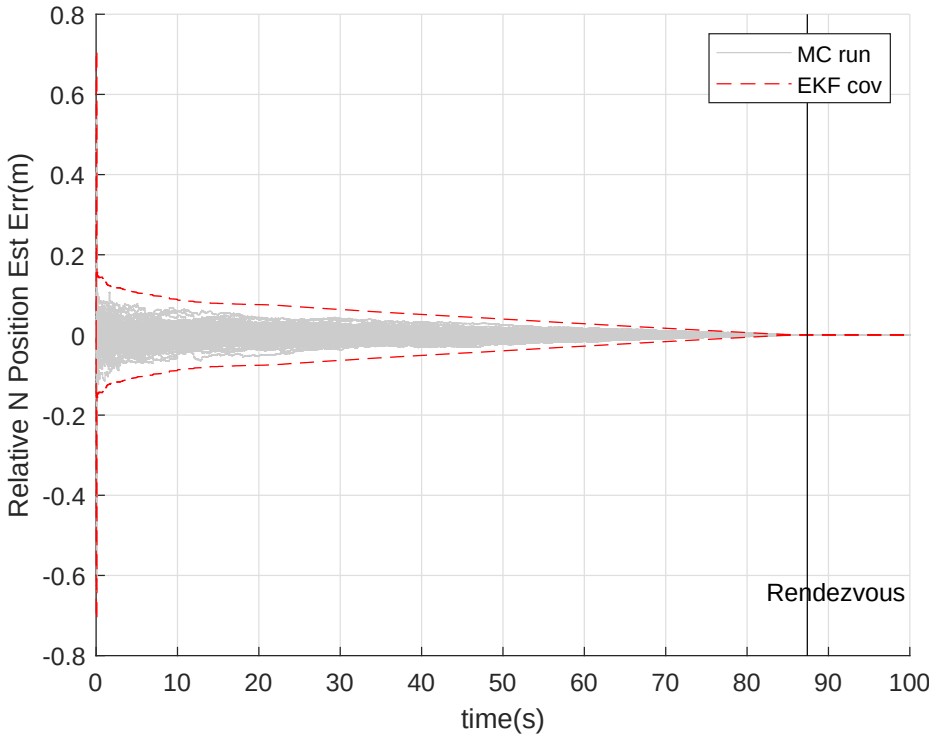

Fig. B.13: Architecture 4 Monte Carlo Analysis of Relative North Position.

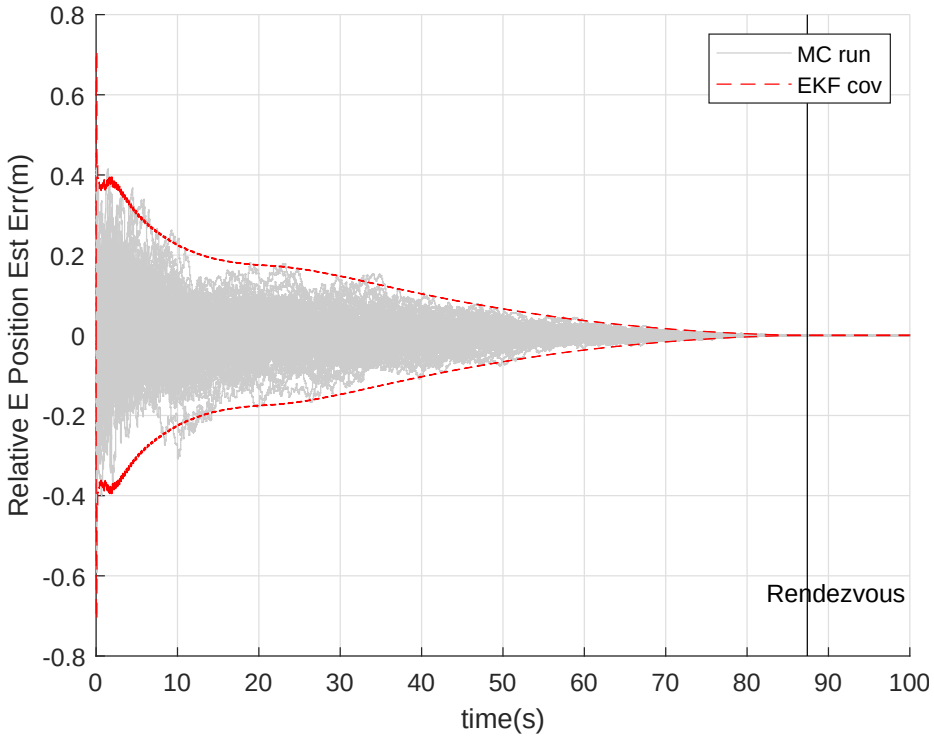

Fig. B.14: Architecture 4 Monte Carlo Analysis of Relative East Position.

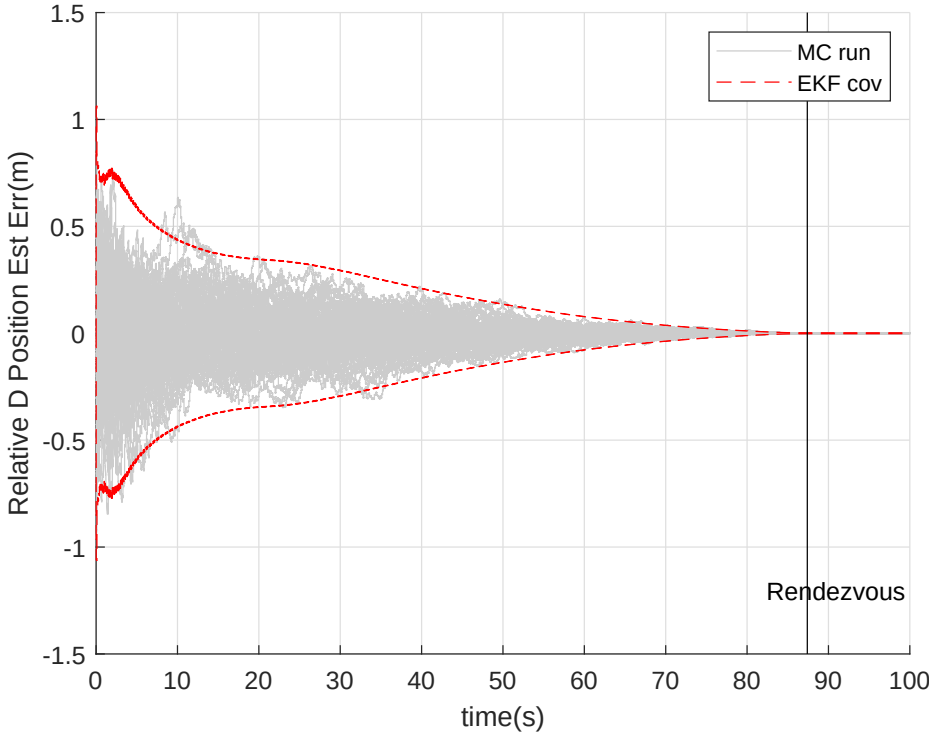

Fig. B.15: Architecture 4 Monte Carlo Analysis of Relative Down Position.

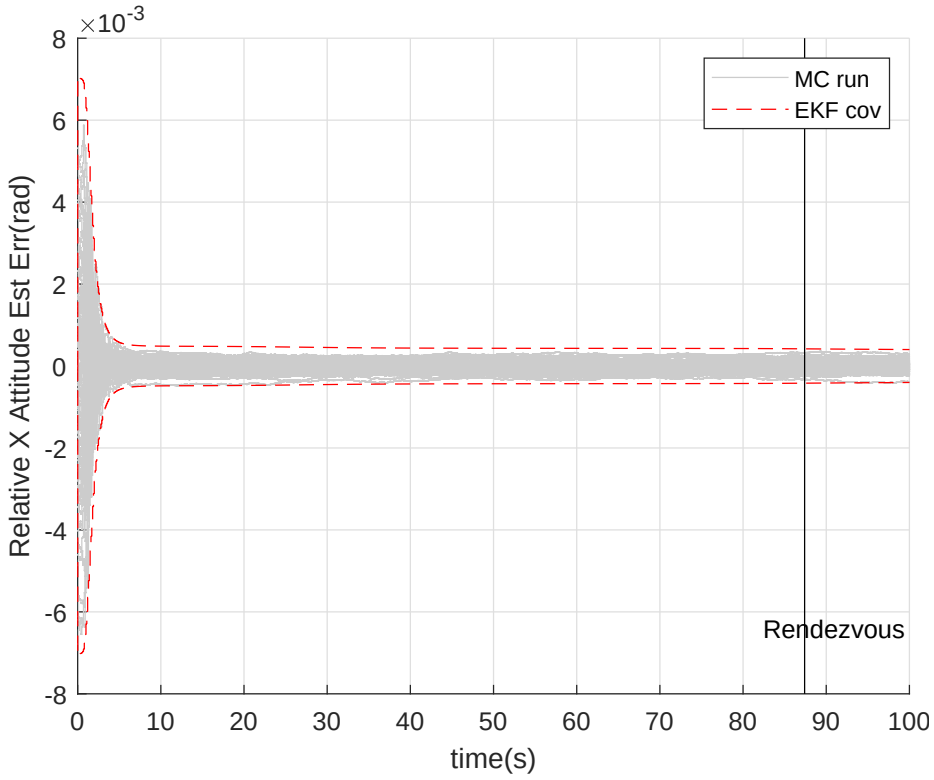

Fig. B.16: Architecture 4 Monte Carlo Analysis of Relative X Axis Attitude.

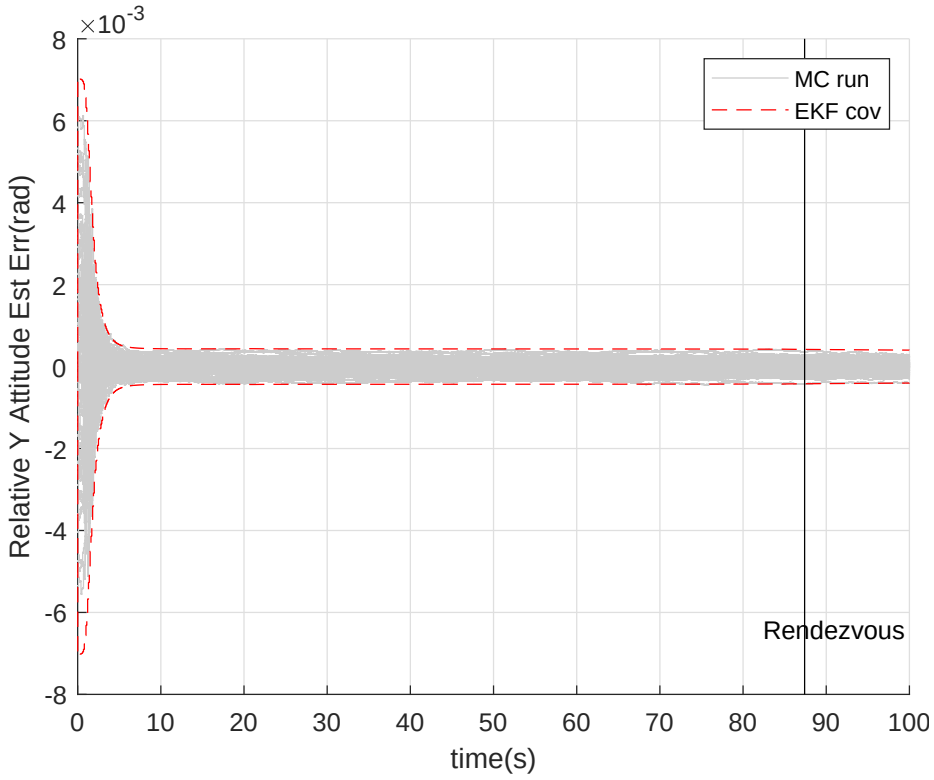

Fig. B.17: Architecture 4 Monte Carlo Analysis of Relative Y Axis Attitude.

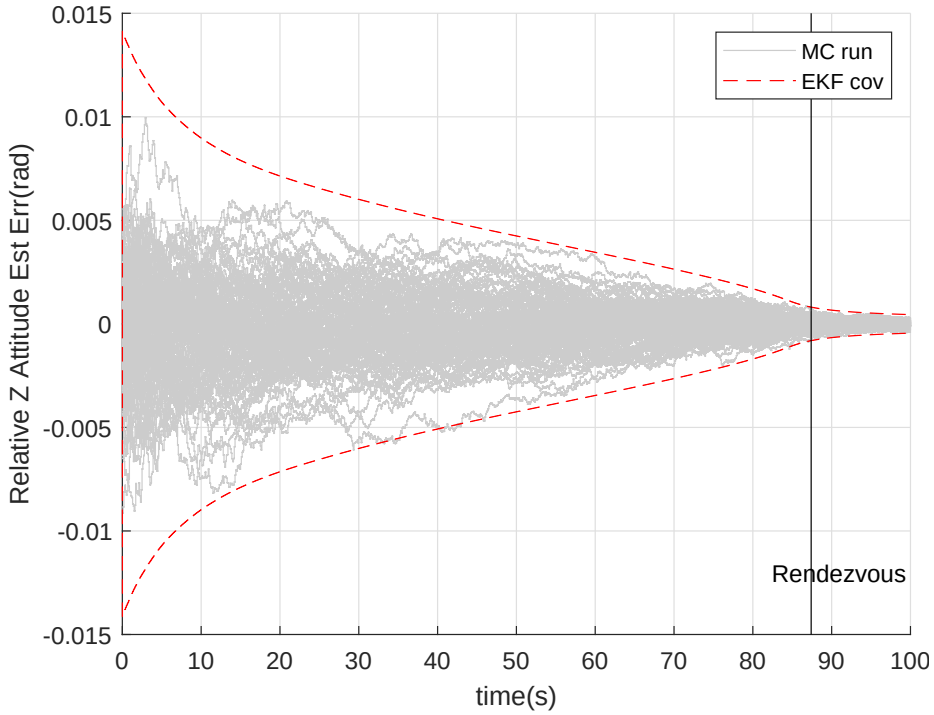

Fig. B.18: Architecture 4 Monte Carlo Analysis of Relative Z Axis Attitude.

B.4 Monte Carlo Hair Plots for Architecture 5 with Medium-Grade IMU

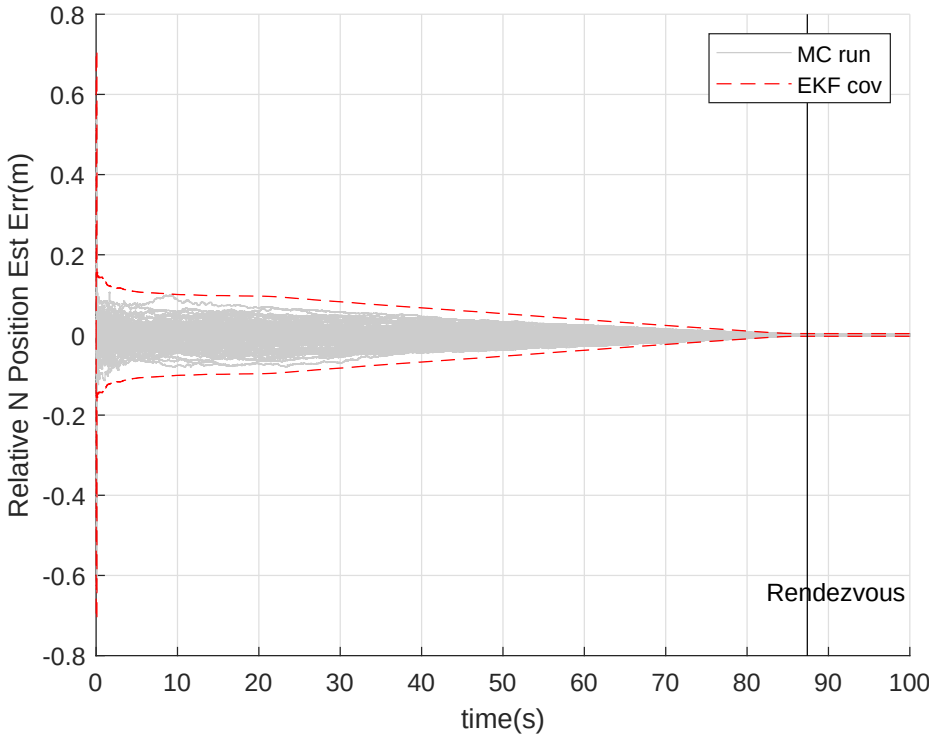

Fig. B.19: Architecture 5 Monte Carlo Analysis of Relative North Position.

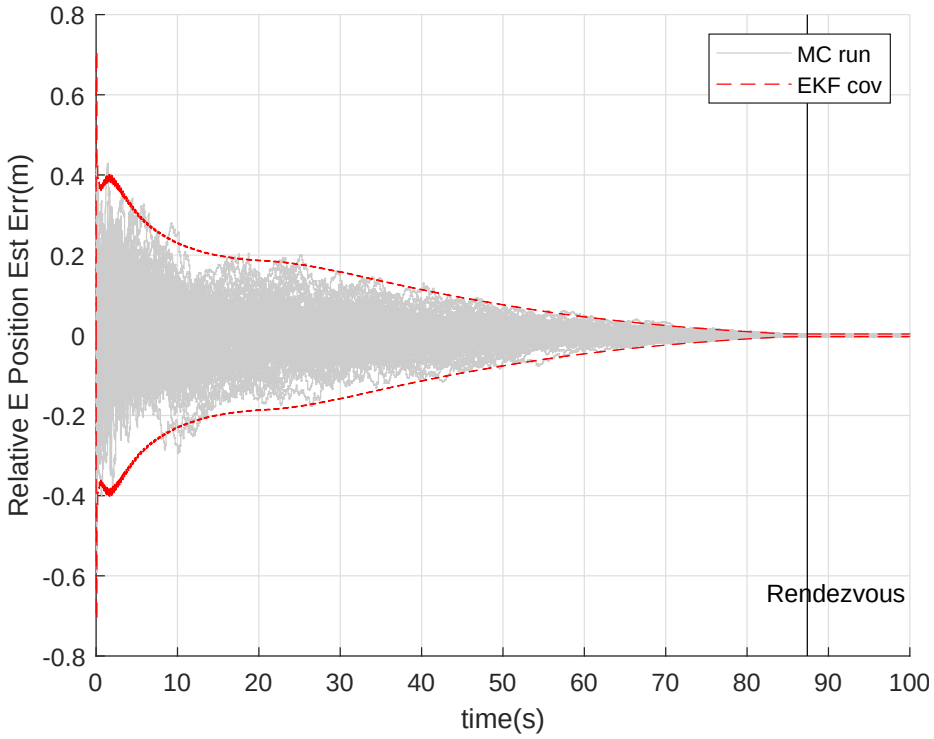

Fig. B.20: Architecture 5 Monte Carlo Analysis of Relative East Position.

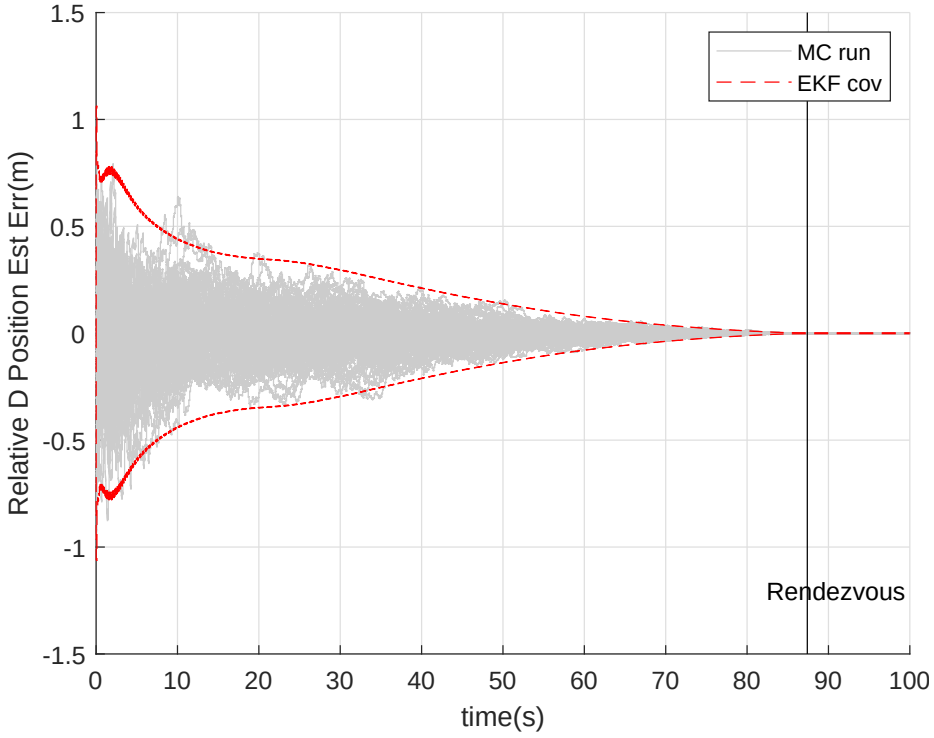

Fig. B.21: Architecture 5 Monte Carlo Analysis of Relative Down Position.

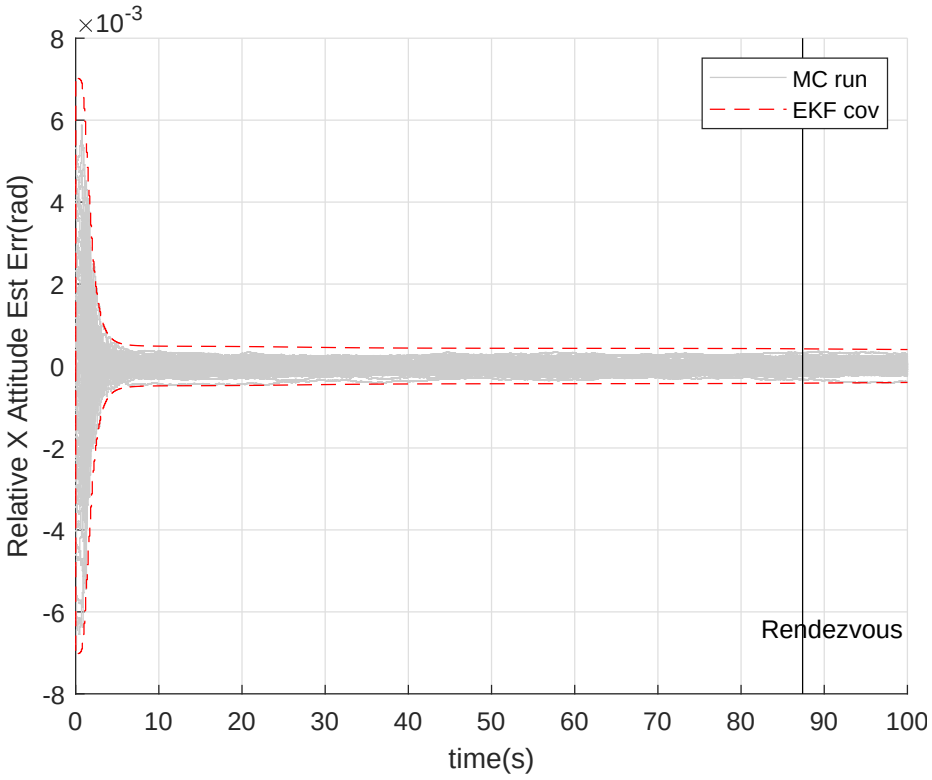

Fig. B.22: Architecture 5 Monte Carlo Analysis of Relative X Axis Attitude.

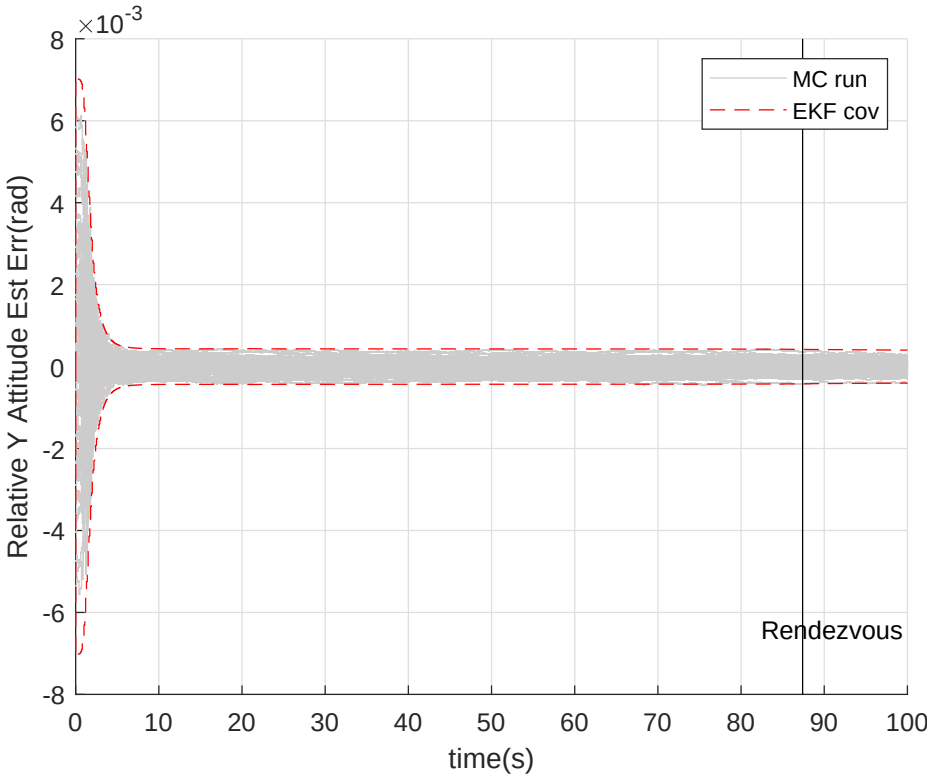

Fig. B.23: Architecture 5 Monte Carlo Analysis of Relative Y Axis Attitude.

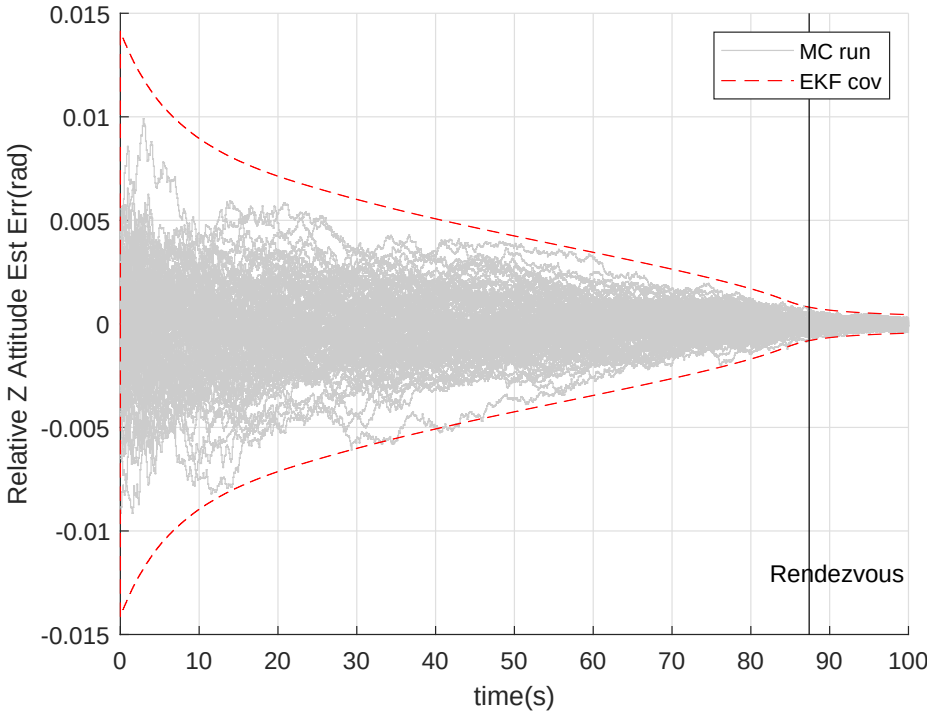

Fig. B.24: Architecture 5 Monte Carlo Analysis of Relative Z Axis Attitude.

B.5 Monte Carlo Hair Plots for Architecture 6 with Medium-Grade IMU

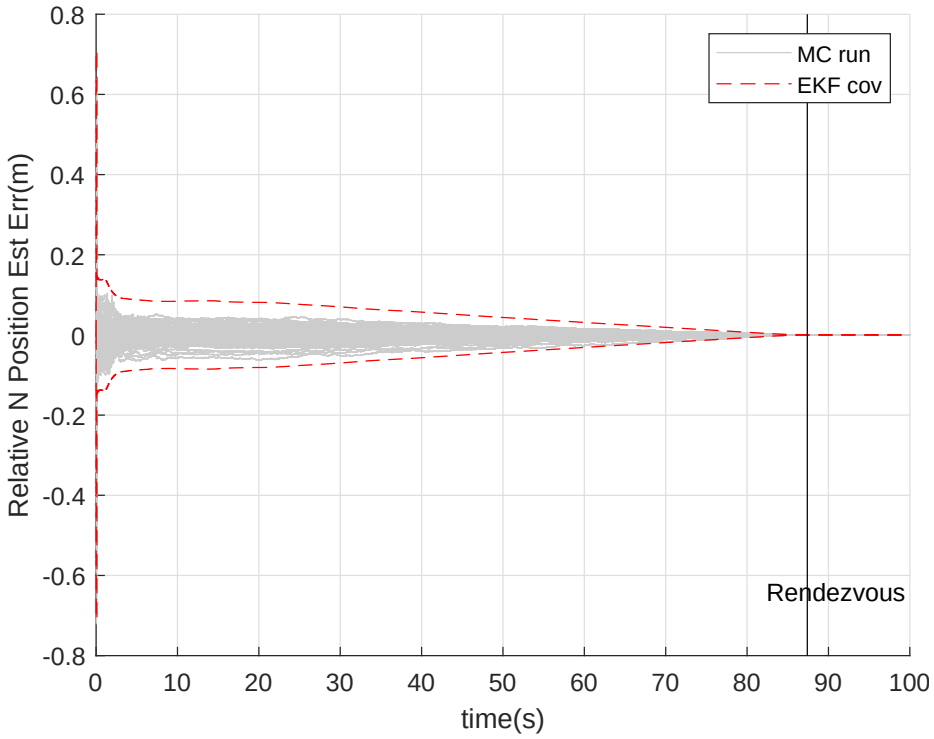

Fig. B.25: Architecture 6 Monte Carlo Analysis of Relative North Position.

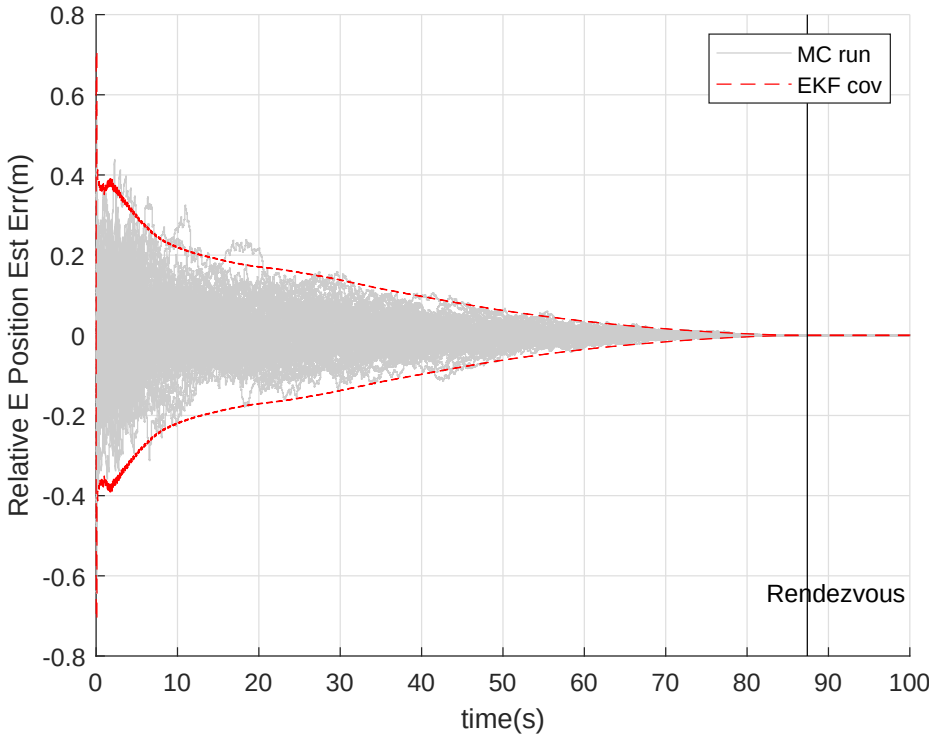

Fig. B.26: Architecture 6 Monte Carlo Analysis of Relative East Position.

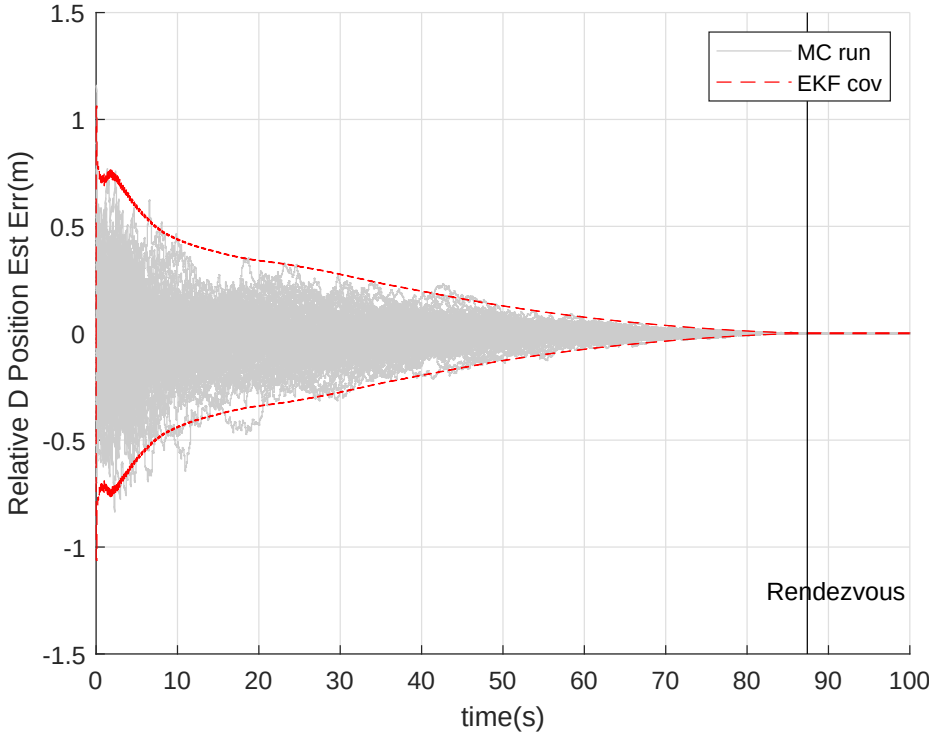

Fig. B.27: Architecture 6 Monte Carlo Analysis of Relative Down Position.

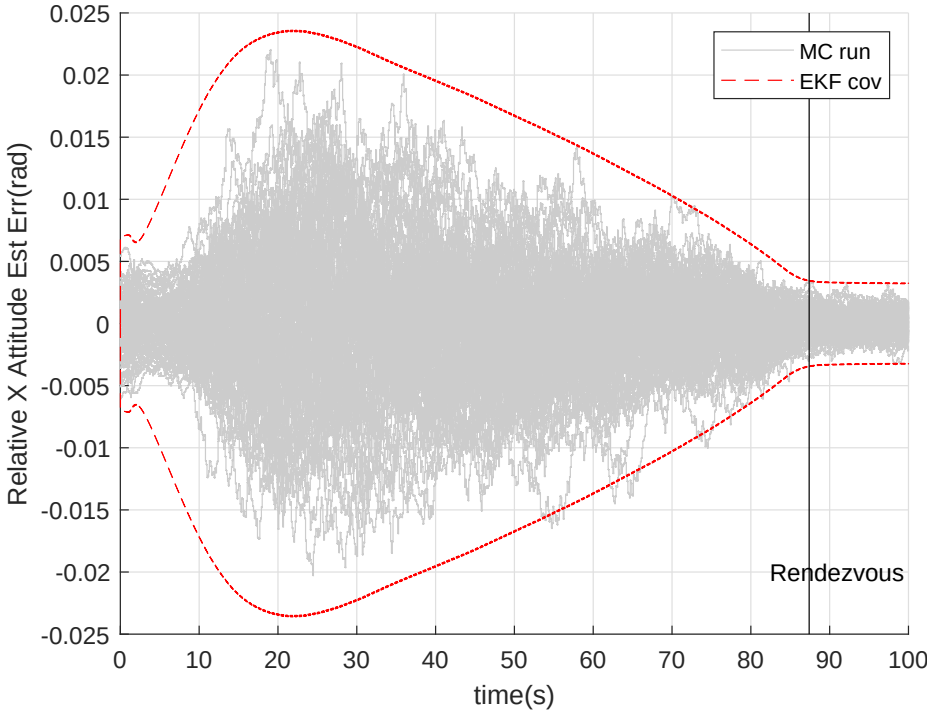

Fig. B.28: Architecture 6 Monte Carlo Analysis of Relative X Axis Attitude.

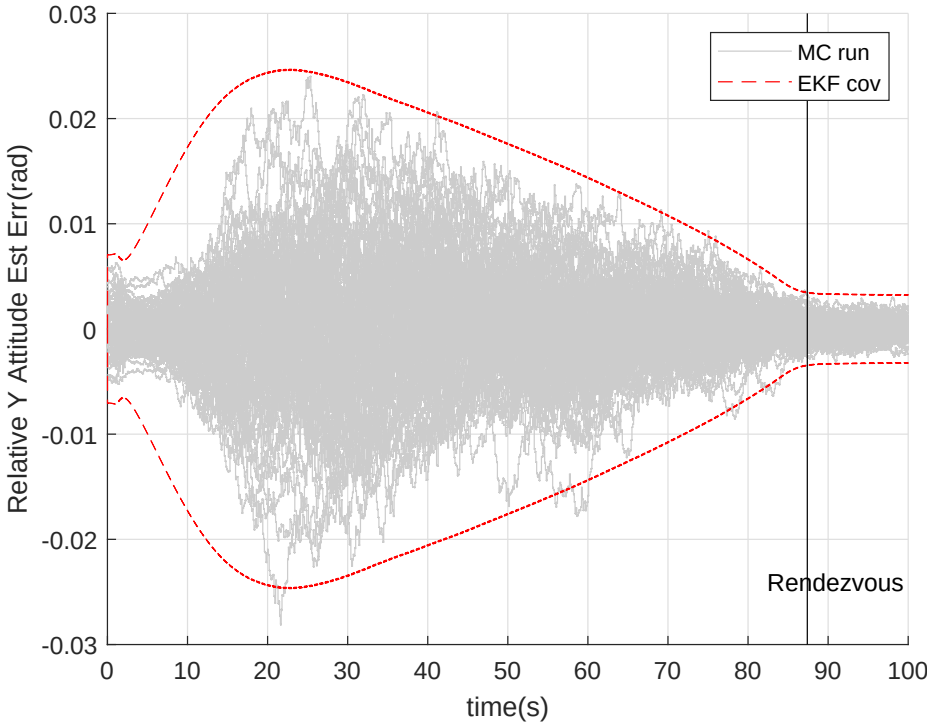

Fig. B.29: Architecture 6 Monte Carlo Analysis of Relative Y Axis Attitude.

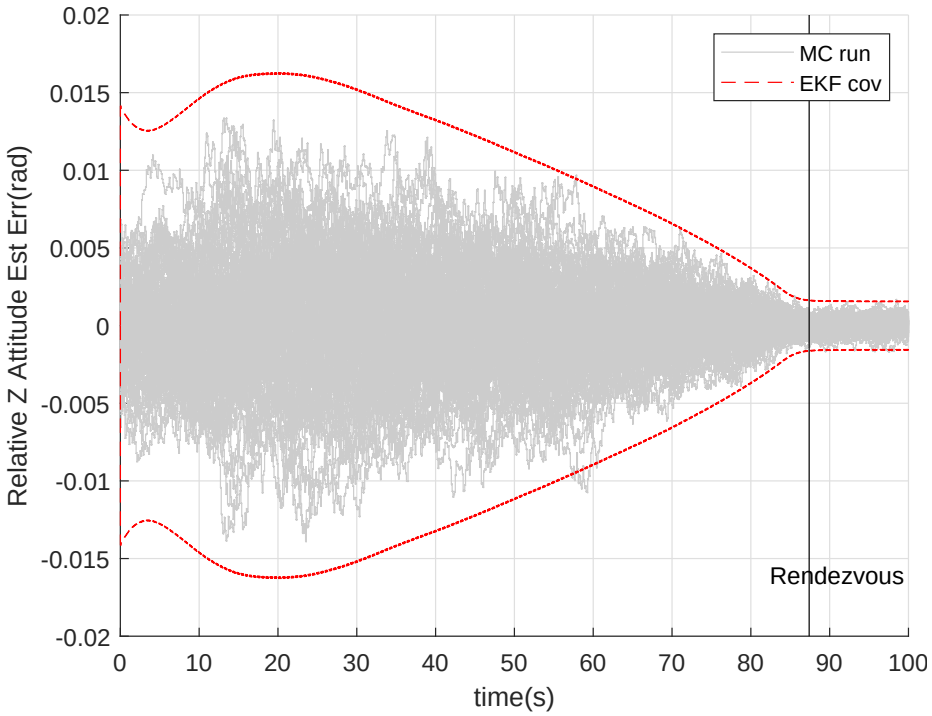

Fig. B.30: Architecture 6 Monte Carlo Analysis of Relative Z Axis Attitude.

B.6 Monte Carlo Hair Plots for Architecture 7 with Medium-Grade IMU

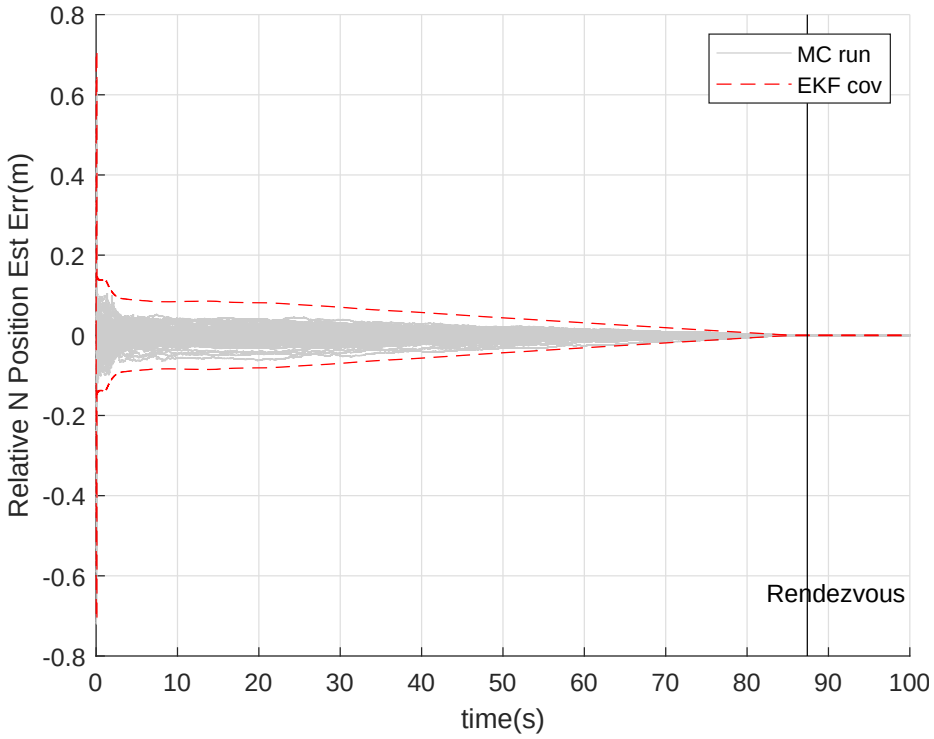

Fig. B.31: Architecture 7 Monte Carlo Analysis of Relative North Position.

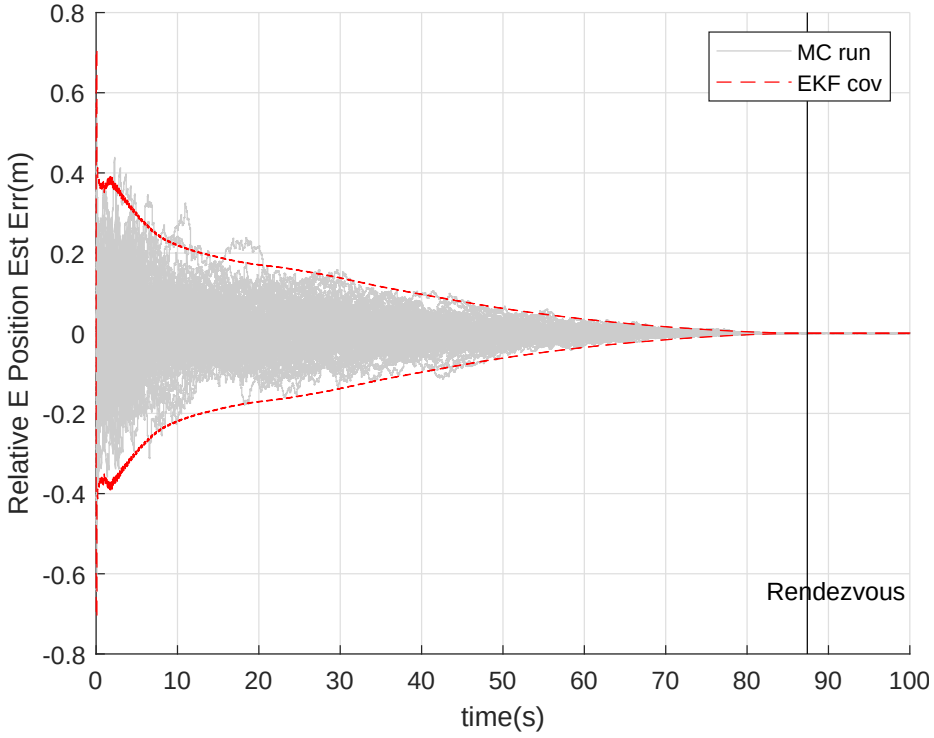

Fig. B.32: Architecture 7 Monte Carlo Analysis of Relative East Position.

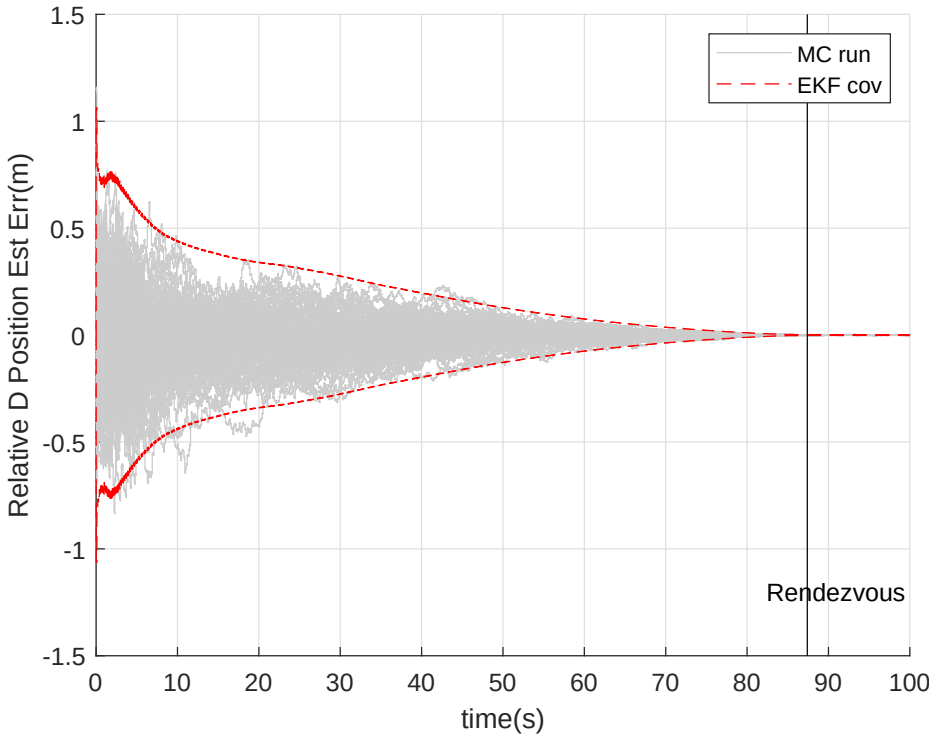

Fig. B.33: Architecture 7 Monte Carlo Analysis of Relative Down Position.

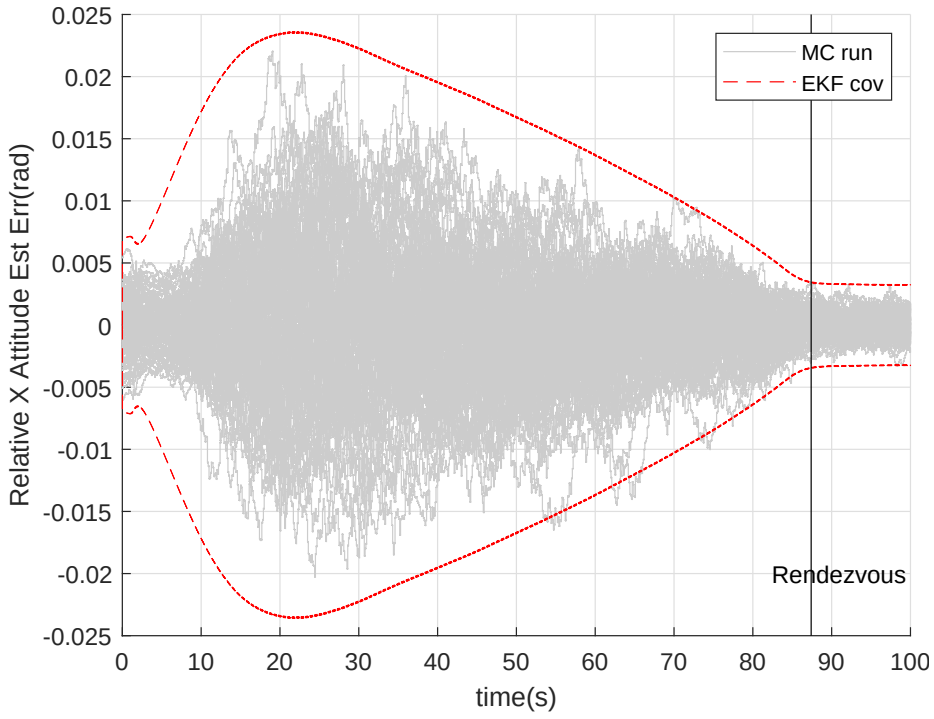

Fig. B.34: Architecture 7 Monte Carlo Analysis of Relative X Axis Attitude.

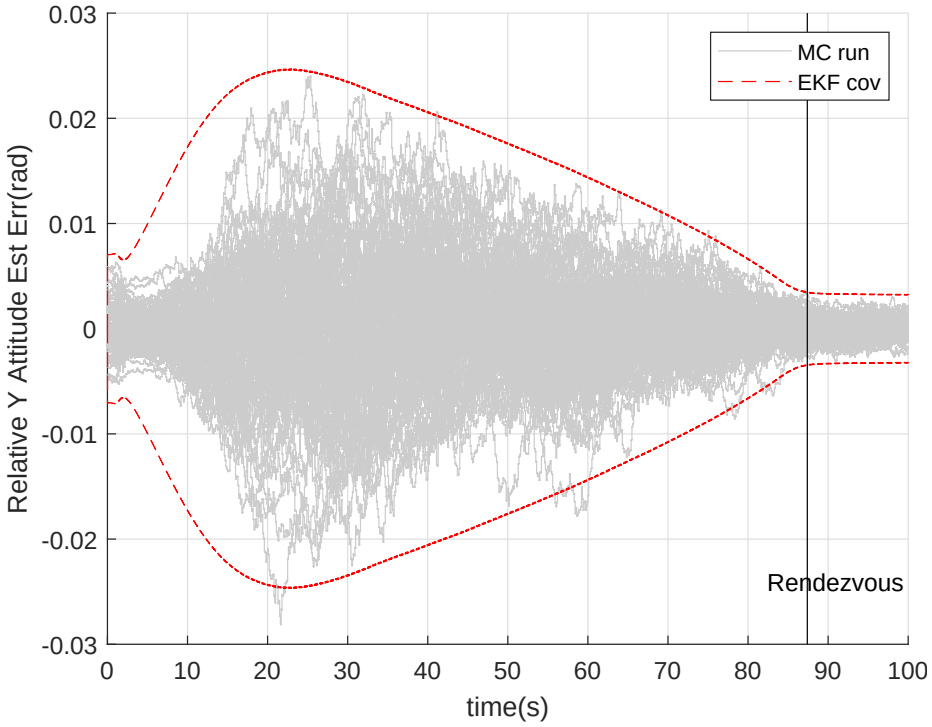

Fig. B.35: Architecture 7 Monte Carlo Analysis of Relative Y Axis Attitude.

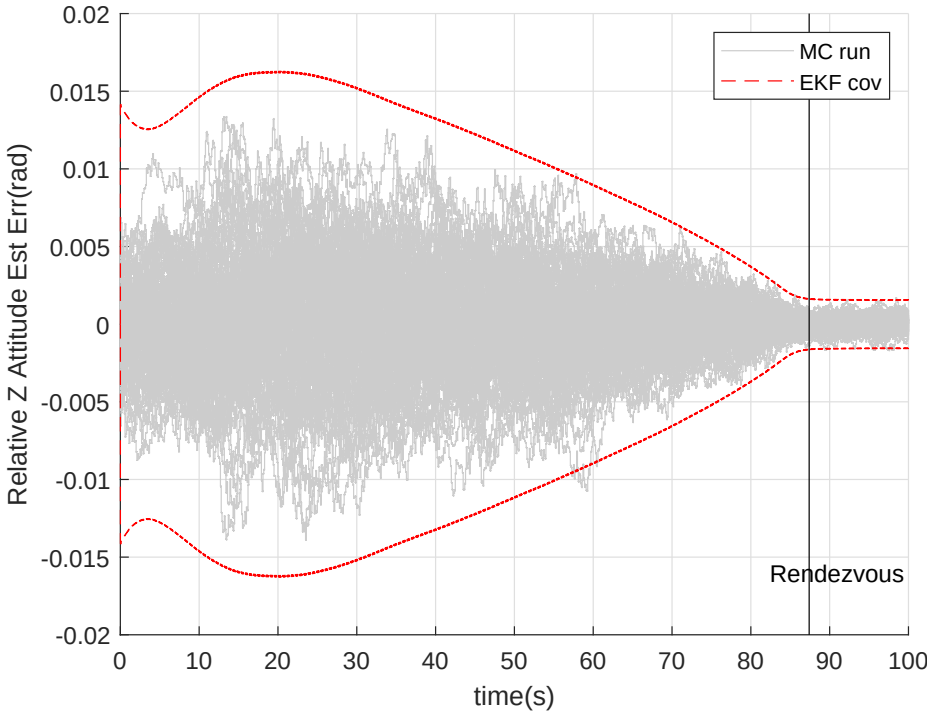

Fig. B.36: Architecture 7 Monte Carlo Analysis of Relative Z Axis Attitude.