EINDHOVEN UNIVERSITY OF TECHNOLOGY

Department of Mechanical Engineering
Dynamics and Control Group

Fault Diagnostics and Estimation of a Spacecraft Attitude Determination System

Project Report

4DM110 – Fault Diagnostics and Security for Control Systems

Tobias Wejbora (1785486)
[redacted]

Lecturer:
dr. M.S. Chong
Dynamics & Control group, Department of Mechanical Engineering, TU/e

Teachers Assistant:
Zeyad Manaa
Dynamics & Control group, Department of Mechanical Engineering, TU/e

Eindhoven, June 2026

Abstract

This report designs and evaluates fault diagnostic and estimation methods for the three-axis spacecraft attitude system of Pirmoradi et al. The system is linearised to decoupled double integrators and stabilised by a PD feedback controller. For the fault diagnostics, an unknown-input observer (UIO) is designed for pure sensor faults on the closed-loop plant. The UIO design is simplified to a fault-free model simulation, where faults are read from the output residual. For secure state estimation under sensor attacks, a bank of Luenberger observers is run over all observable sensor subsets and the decoder selects the least-disagreeing estimate. Because of the block-diagonal plant structure and lack of cross-coupling, observable subsets need to keep all three angle sensors. Attacks on gyroscope rate sensors are correctly decoded, but attacks on angle sensors corrupt the estimate. A method to restore observability by the addition of redundant sensors is presented. For resilient control, the nonlinear-encoding of Joo, Qu and Namerikawa (2021) is implemented in continuous time. The control signal is scaled by the encoding factor before crossing the network and being divided out at the plant using a slave circuit that synchronises. A perfectly stealthy integrity attack signal is multiplied by an uncancellable chaotic signal in the residual and is therefore detectable.

Acknowledgments

The authors thank the lecturer, dr. M.S. Chong, for her guidance and feedback throughout the project and Zeyad Manaa assisting with exercises and feedback.

Contents

Nomenclature

Roman symbols

t, k	Continuous and discrete time
T_s	Sampling period (0.1 s)
x	State vector $[p\ q\ r\ \phi\ \theta\ \psi]^\top$
u	Control input (reaction-wheel torques)
u_{tx}	Control input transmitted by the controller
u_{rx}	Control input received by the plant
y, y_i	Output / measurement of sensor i
a, a_i	Fault or attack signal on sensor i
p, q, r	Roll, pitch, yaw angular rates
I_x, I_y, I_z	Principal moments of inertia
A, B, C	System, input, output matrices
K	PD output-feedback gain matrix
E, F, G	Unknown-input observer gain matrices
L	Luenberger observer gain matrix
N	Number of sensors
M	Number of attacked sensors
S, P	Level-1 and level-2 sensor subsets
v_1, v_2, I	Chua circuit state (voltage, voltage, current)
l_c	Chua slave coupling gain

Greek symbols

ϕ, θ, ψ	Roll, pitch, yaw Euler angles
ω_n	Closed-loop natural frequency (0.1 rad/s)
ζ	Closed-loop damping ratio (0.8)
ε	Encoding depth (0.15)
ξ, $\hat{\xi}$	Chua encoding signal at plant / controller
π_S	Subset disagreement score
σ	Selected subset index
α_c	Chua synchronisation decay rate
τ	Communication delay

Subscripts and superscripts

$\dot{(\cdot)}$	First time derivative
$\hat{(\cdot)}$	Estimated quantity
$(\cdot)_c$	Continuous-time
$(\cdot)_d$	Discrete-time
$(\cdot)_{cl}$	Closed-loop
$(\cdot)_a$	Attack-related quantity

Abbreviations

LTI	Linear time-invariant
UIO	Unknown-input observer
SSE	Secure state estimator
PD	Proportional–derivative
ZOH	Zero-order hold
ODE	Ordinary differential equation

Chapter 1

Introduction

Modern spacecrafts make use of attitude determination systems to keep the vehicle pointing in the correct orientation. This is important for use cases like communication, imaging or even generating power from the sun. The control loop is sent over a communication link between the spacecraft and the control station where humans monitor the system. This means that the sensor measurement and actuator comments travel (usually) by means of radio signals across a channel that is shared and open to eavesdropping. There are two threats that must be considered, faults and cyber-attacks. Faults occur when a rate gyroscope for example develops a bias or a reaction wheel that delivers a wrong amount of torque. Cyber-attacks are deliberate attacks that corrupt the data crossing the network. Therefore, both unintentional events like faults and intentional events like cyber-attacks which can be sneaky so that it is not detected in a standard residual-based detector are considered in this report.

Throughout this report, the three-axis spacecraft attitude determination system of Pirmoradi et al. [2] is the chosen system where faults and cyber-attacks are modeled on. The spacecraft considered in [2] has three rate gyroscopes, which measure the roll, pitch and yaw angular velocities, and three vector sensors, measuring the corresponding attitude angles. The spacecraft is actuated by reaction-wheel torques and implements an output-feedback controller. Regarding the rigid-body dynamics, they are nonlinear because of the gyroscopic coupling between the axes, but around the nominal (stationary) operating point they linearize into decoupled double integrators. By doing so, the system becomes an LTI system required for the fault-diagnosis and secure-estimation theorems.

In the following sections of the report, the first implementation is an unknown-input observer to detect and estimate sensor faults. Then secure state estimation for reconstructing the spacecraft state when a subset of the sensors are under attack. This is done through running multiple Luenberger observers over the healthy sensor subsets. The final implementation in this report is a nonlinear encoding method to detect stealthy attacks on the system.

Chapter 2

System and model

This chapter works through the non-linear rigid-body equations and how they are linearized as well as the stabilizing controller and discrete-time form of the system.

2.1 Nonlinear rigid-body model

There are two distinct reference frames that must be clarified and introduced regarding the non-linear motion of the spacecraft. Firstly, the rotational dynamics are defined in the spacecraft's body-fixed frame with Euler's equations, where (p, q, r) represent the internal body rates (roll, pitch, and yaw angular velocities) which are actuated by the reaction-wheel torques. Secondly, the the global orientation of the spacecraft is defined using an external inertial frame with the Euler angles (ϕ, θ, ψ). The following kinematic equations relate the local body rates to the rates of change of the global Euler angles:

$$\dot{p} = \tfrac{I_y - I_z}{I_x}\, qr + \tfrac{1}{I_x} T_x, \qquad \dot{\phi} = p + \sin\phi \tan\theta\, q + \cos\phi \tan\theta\, r, \tag{2.1a}$$

$$\dot{q} = \tfrac{I_z - I_x}{I_y}\, pr + \tfrac{1}{I_y} T_y, \qquad \dot{\theta} = \cos\phi\, q - \sin\phi\, r, \tag{2.1b}$$

$$\dot{r} = \tfrac{I_x - I_y}{I_z}\, pq + \tfrac{1}{I_z} T_z, \qquad \dot{\psi} = \tfrac{\sin\phi}{\cos\theta}\, q + \tfrac{\cos\phi}{\cos\theta}\, r, \tag{2.1c}$$

with state $x = [\, p\ q\ r\ \phi\ \theta\ \psi \,]^\top$, control input $u = [\, T_x\ T_y\ T_z \,]^\top$ (the reaction-wheel torques), and moments of inertia $I_x = 10,\ I_y = 12,\ I_z = 2\ \mathrm{kg\,m^2}$ [2].

2.2 Linearised continuous-time model

When linearising the nonlinear equations (2.1) around the equilibrium point $x = 0, u = 0$, there are some simplifications to be made. Firstly, it is assumed that the states are small near the stationary equilibrium and therefore, the gyroscopic cross-products (e.g., qr, pr, pq) are approximately zero. Secondly, the trigonometric terms in the kinematic equations disappear when using the small-angle approximation. By doing so, the body rates now directly relate to the rates of change of the global Euler angles (e.g., $\dot{\phi} \approx p$). This decouples the roll, pitch, yaw axes into double integrators. The linearised dynamics are writen in state-space form with $\dot{x} = A_c x + B_c u$:

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p \\ q \\ r \\ \phi \\ \theta \\ \psi \end{bmatrix} + \begin{bmatrix} \frac{1}{I_x} & 0 & 0 \\ 0 & \frac{1}{I_y} & 0 \\ 0 & 0 & \frac{1}{I_z} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \tag{2.2}$$

With the given moments of inertia $(I_x = 10,\ I_y = 12,\ I_z = 2)$ from [2], the matrices are can also be written in block notation:

$$A_c = \begin{bmatrix} 0_{3\times3} & 0_{3\times3} \\ I_3 & 0_{3\times3} \end{bmatrix}, \qquad B_c = \begin{bmatrix} 0.1 & 0 & 0 \\ 0 & \frac{1}{12} & 0 \\ 0 & 0 & 0.5 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{2.3}$$

Here it is also assumed that the full state is directly measured since the system has three rate gyroscopes (p, q, r) and three vector sensors (ϕ, θ, ψ). Therefore, $y = C_c x$ is simply the identity matrix:

$$y = \begin{bmatrix} p \\ q \\ r \\ \phi \\ \theta \\ \psi \end{bmatrix} \quad \Rightarrow \quad C_c = I_6 \tag{2.4}$$

2.3 Output-feedback PD controller

Looking at the pure double integrator plant from equation (2.3) there are some challenges. Firstly, the double integrators have two poles at the origin of the complex plane, thus the open-loop system is not asymptotically stable. In space, there is no natural damping thus any disturbance will make the states drift. A feedback controller is made to stabilize the spacecraft.

A PD control law $u = -Kx$ is implemented and since the state vector is made with the body rates first, then the angles ($x = [p, q, r, \phi, \theta, \psi]^\top$), the controller matrix is made so that: $K = \begin{bmatrix} K_d & K_p \end{bmatrix}$. Therefore, the K_p gains act on the angles for a restoring stiffness, while the K_d gains act on the body rates for damping.

The full gain matrix K is made so that each gain term scales with its moment of inertia (I_x, I_y, I_z). To explicitly demonstrate how this mathematically enforces decentralization, the full control law $u = -Kx$ is expanded in equation (2.5). The off-diagonal terms are zero for decentralization, the derivative gains map to the first three states (body rates), and the proportional gains map to the last three states (Euler angles):

$$\begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} = - \begin{bmatrix} 2\zeta\omega_n I_x & 0 & 0 & \omega_n^2 I_x & 0 & 0 \\ 0 & 2\zeta\omega_n I_y & 0 & 0 & \omega_n^2 I_y & 0 \\ 0 & 0 & 2\zeta\omega_n I_z & 0 & 0 & \omega_n^2 I_z \end{bmatrix} \begin{bmatrix} p \\ q \\ r \\ \phi \\ \theta \\ \psi \end{bmatrix} \tag{2.5}$$

The benefit of this structure of the matrix K is that when multipled, the off-diagonals eliminate all cross-axis interactions. For example, the roll torque $T_x = -(2\zeta\omega_n I_x)p - (\omega_n^2 I_x)\phi$.

To determine the values for these gains, the characteristic equation for a double integrator with PD control was used for a single axis with inertia I:

$$s^2 + \frac{K_d}{I} s + \frac{K_p}{I} = 0 \tag{2.6}$$

This is then compared to the characteristic equaiton for a second order system with a natural frequency ω_n and damping ratio ζ:

$$s^2 + 2\zeta\omega_n s + \omega_n^2 = 0 \tag{2.7}$$

Then by matching the coefficients of the two equations, where the constant (s^0) terms give the proportional gain, and the s^1 terms give the derivative gain:

$$\frac{K_p}{I} = \omega_n^2 \quad \Rightarrow \quad K_p = \omega_n^2 I \tag{2.8a}$$

$$\frac{K_d}{I} = 2\zeta\omega_n \quad \Rightarrow \quad K_d = 2\zeta\omega_n I \tag{2.8b}$$

What has occurred is that the gains have been scaled by the moments of inertia. This cancels the $1/I$ in the B matrix seen in (2.2). Now ω_n and ζ can made designed to be identical for all three axes, no matter the difference in inertias.

Since the axis are decoupled the design parameters are ω and ζ which can be designed independently. $\omega_n = 0.1$ rad/s and $\zeta = 0.8$ were chosen by testing a combinations of ω and ζ and comparing settling time and overshoot percentages(`stepinfo`). Furthermore, the overshoot of a second-order step response is $M_p = e^{-\pi\zeta/\sqrt{1-\zeta^2}}$, and with $\zeta = 0.8$ is only 1.5%. Regarding the transient decays from the start,$e^{-\zeta\omega_n t} = e^{-0.08t}$, meaning the error is about $0.4°$ by $t = 40\,\text{s}$ which is when faults are injected. Figure 2.1 shows the step response for each axis of the closed-loop.

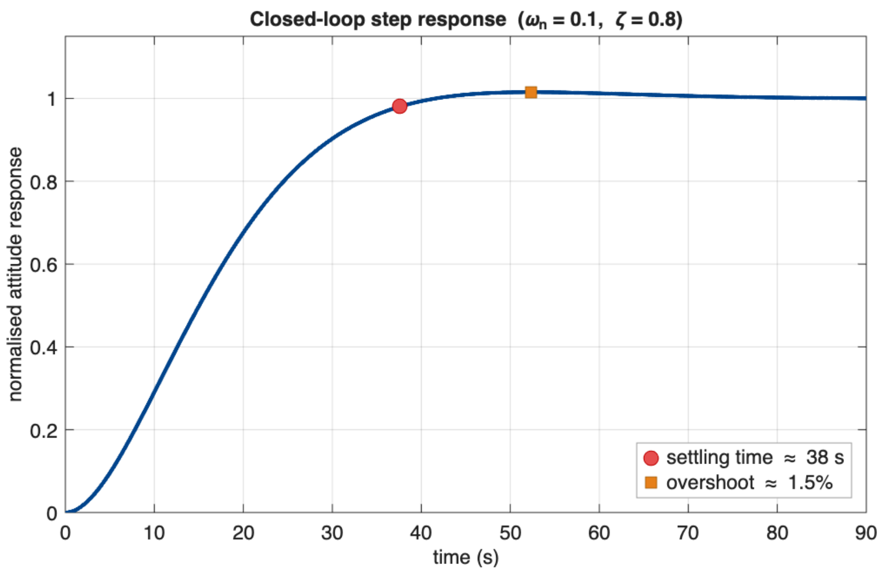

Figure 2.1: Closed-loop step response for the chosen gains(per-axis) $(\omega_n, \zeta) = (0.1, 0.8)$. The response settles to within $\pm 2\%$ in $\approx 38\,\text{s}$ with $\approx 1.5\%$ overshoot.

Substituting these parameters and the inertias gives the following K gain matrix:

$$K = \begin{bmatrix} 1.60 & 0 & 0 & 0.10 & 0 & 0 \\ 0 & 1.92 & 0 & 0 & 0.12 & 0 \\ 0 & 0 & 0.32 & 0 & 0 & 0.02 \end{bmatrix}. \tag{2.9}$$

2.4 Discrete-time model

The controller and estimators run on a digital computer, so the continuous model is discretised with a zero-order hold at $T_s = 0.1$ s.

There are two simplifications made during the discretisation. The A_c matrix found in equation (2.3) is nilpotent, meaning $A_c^2 = 0$. Thus when finding A_d, the Taylor series stops after the second term:

$$A_d = e^{A_c T_s} = I_6 + A_c T_s + \frac{1}{2!} A_c^2 T_s^2 + \dots \quad \Rightarrow \quad A_d = I_6 + A_c T_s \tag{2.10}$$

The nilpotent property also simplifies the ZOH needed for matrix B_d as the two term Taylor series gives an exact integration.

$$B_d = \left(\int_0^{T_s} e^{A_c \tau} d\tau \right) B_c = \left(I_6 T_s + \frac{1}{2} A_c T_s^2 \right) B_c \tag{2.11}$$

Substituting $T_s = 0.1$ and evaluating gives the discrete-time LTI system $x_{k+1} = A_d x_k + B_d u_k$, $y_k = C_d x_k$, with:

$$A_d = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0.1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0.1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0.1 & 0 & 0 & 1 \end{bmatrix}, \quad B_d = \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & \frac{1}{120} & 0 \\ 0 & 0 & 0.05 \\ \frac{1}{2000} & 0 & 0 \\ 0 & \frac{1}{2400} & 0 \\ 0 & 0 & \frac{1}{400} \end{bmatrix}, \quad C_d = I_6 \tag{2.12}$$

By substituting $u_k = -K x_k$ into the plant, the closed-loop dynamics matrix are:

$$A_{cl} = A_d - B_d K \tag{2.13}$$

The maximum magnitude of the eigenvalues of A_{cl} is $|z| \approx 0.992$ which are strictly within the unit circle ($|z| < 1$). Thus, the discrete closed-loop system is Schur stable.

Chapter 3

Fault diagnostics

In this section, an unknown-input observer is applied to the spacecraft of Chapter 2 and is used to detect and isolate a faulty sensor.

3.1 Fault model

A fault is modeled as an unknown input a acting on the discrete plant,

$$x(k+1) = A\,x(k) + B\,a(k), \qquad y(k) = C\,x(k) + D\,a(k). \tag{3.1}$$

Here only sensor faults are considered, such as a bias or drift, which can corrupt one or more measurements without changing the physical motion of the spacecraft. Since the physical dynamics are unaffected, the closed-loop matrix stays $A = A_{cl}$ and the fault does not physically actuate the system, $B_{\text{fault}} = 0$. Since the full state is still measured, $C = I_6$ and the D_{fault} matrix has columns which correspond to the unit vectors e_i of the measurement channels affected by the fault. For example, a fault on the yaw-gyro (r) and the pitch-angle (θ) sensor simultaneously makes $D_{\text{fault}} = [e_3\ e_5]$. Therefore, the fault model simplifies to:

$$x(k+1) = A_{cl}\,x(k), \qquad y(k) = x(k) + D_{\text{fault}}\,a(k). \tag{3.2}$$

The unknown fault vector $a(k)$ reconstructed from the available data, by setting up a system of linear equations and isolate the unknown fault terms on the right-hand side:

$$\begin{aligned} x(k+1) - A_{cl}\,x(k) &= B_{\text{fault}}\,a(k) \\ y(k) - x(k) &= D_{\text{fault}}\,a(k) \end{aligned}$$

Then, to solve simultaneously, they can be stacked vertically:

$$\begin{bmatrix} x(k+1) - A_{cl}\,x(k) \\ y(k) - x(k) \end{bmatrix} = \begin{bmatrix} B_{\text{fault}} \\ D_{\text{fault}} \end{bmatrix} a(k) \tag{3.3}$$

In order to isolate $a(k)$, the stacked matrix has to be inverted. To do so, it must have full column rank. If the matrix dropped rank, there would be multiple combinations of faults that would produce the same output residuals, making it impossible to see which specific sensor has failed.

For the sensor fault scenarios here, the full column rank holds, because they don't affect the rigid-body dynamics, B_{fault} and the stacked matrix simplifies to $\left[\begin{smallmatrix} 0 \\ D_{\text{fault}} \end{smallmatrix}\right]$.

Looking at the example from earlier of a simultaneous yaw-gyro (e_3) and pitch-angle (e_5)

fault, the resulting stacked matrix is:

$$\left[\begin{array}{c} B_{\text{fault}} \\ \hline D_{\text{fault}} \end{array}\right] = \left[\begin{array}{cc} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \hline 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{array}\right] \tag{3.4}$$

These are clearly independent and any combination of broken sensors simply adds independent columns into D_{fault}. Thus for any combination of simultaneous sensor faults, the columns hold full rank.

3.2 Unknown-input observer

The goal of the UIO is to reconstruct the state without knowing a by looking at L sequential output steps simultaneously,

$$\hat{x}(k+1) = E\,\hat{x}(k) + F\,y(k : k+L), \tag{3.5}$$

where $y(k : k+L)$ stacks $y(k), \dots, y(k+L)$. The observer can then recover the fault from the residual difference between the predicted and actual dynamics:

$$\hat{a}(k) = \begin{bmatrix} B_{\text{fault}} \\ D_{\text{fault}} \end{bmatrix}^{+} \begin{bmatrix} \hat{x}(k+1) - A_{cl}\hat{x}(k) \\ y(k) - C\hat{x}(k) \end{bmatrix}. \tag{3.6}$$

The standard procedure for determining the UIO gains was done by solving $SJ_L = [\,B_{\text{fault}}\ 0\ \cdots\ 0\,]$ (where O_L and J_L are the the stacked observability and input matrices). Then N is taken in the left nullspace of J_L, then by choosing a gain matrix G so that $E = (A_{cl} - SO_L) - GNO_L$ is Schur stable, and then setting $F = S + GN$.

However, in the specific case of the spacecraft sensor-fault setup, there are some simplifications to be made. Firstly, since the faults considered are only sensor faults, they show up instantly in the measurement. Since there are no actuator fault that would require the computer to wait and watch the sensors for a few timesteps in order to determine a good state estimate. Therefore, since D_{fault} is full rank, the step zero input matrix $J_0 = D_{\text{fault}}$ is also full column rank, not needing measurement look-ahead ($L = 0$).

The matrix S is the decoupling matrix which essentially acts as a filter that subtracts out any physical disturbance affecting the observer. However, since only sensor faults are considered, there is nothing to subtract, hence $S = 0$.

Finally, the observer feedback gain G is there to correct the observer to the true state, however since the PD control loop has made the true plant A_{cl} Schur stable, there is no need for an observer feedback. Hence $G = 0$ and this creates an observer that is blind to the sensors:

$$F = S + GN = 0, \qquad E = A_{cl}, \tag{3.7}$$

This results in a state estimator that runs a simulation based on physics equations of the closed-loop model: $\hat{x}(k+1) = A_{cl}\hat{x}(k)$. Equation (3.6) is therefore simplified to determining the fault purely from the output residual $y(k) - \hat{x}(k)$. This fault diagnostic implementation thus considered any divergence in the residual to a fault on its corresponding measurement channel.

3.3 Inertia uncertainty

The fault diagnostics has been modeled based on the model A_{cl}, which depends on the moments of inertia I_x, I_y, I_z. These are not exactly known at any given moment in time due to anomalies that could shift mass properties over time. Thus there is a possibility that there is a mismatch between the true inertia and the value that the controller reads. This could cause the fault model to affect control performance.

The error can be modeled as $I^{\text{true}} = (1+\delta)\,I^{\text{nom}}$ on the three axes. The gain K and the model A_{cl} are ran with I^{nom}, while the plant runs on I^{true}, so the real closed loop is $A_{cl}^{\text{true}} = A_d - B_d(I^{\text{true}})\,K(I^{\text{nom}})$, however, the diagnostic is ran on $A_{cl}^{\text{nom}} = A_d B_d(I^{\text{nom}})\,K(I^{\text{nom}})$. Since $B_d \propto 1/I$, a term can be factored out, $B_d(I^{\text{true}}) = B_d(I^{\text{nom}})/(1+\delta)$, so the real closed loop is

$$A_{cl}^{\text{true}} = A_d - \frac{1}{1+\delta} B_d(I^{\text{nom}})K(I^{\text{nom}}),$$

Because $K \propto I$ while $B_d \propto 1/I$, the product $B_d K$ is inertia-independent (nominally), so the closed loop barely changes. We sweep δ from -60% to $+60\%$, to cover more than enough of the realistic mass-property error range. Then the worst-case fault-free residual which is essentially the false-alarm risk is measured.

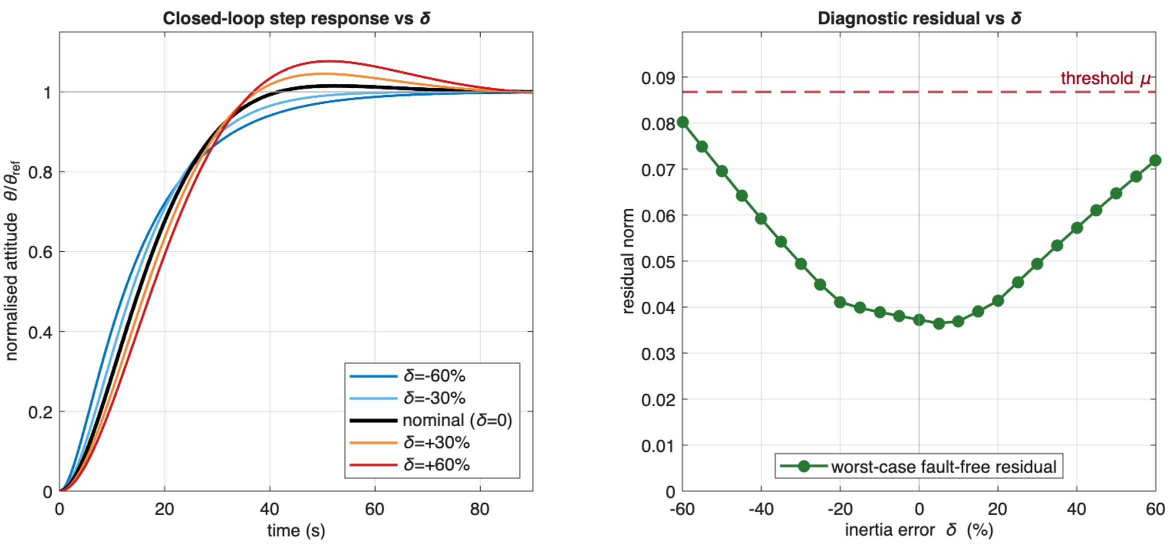

Figure 3.1: Robustness to an inertia error δ swept to $\pm 60\%$. Left: The closed-loop step response is stable and a heavier spacecraft ($\delta > 0$) overshoots more ($\approx 8\%$ at $+60\%$), a lighter spacecraft is overdamped, but they all settle to the reference. Right: The worst-case fault-free residual increases with $|\delta|$ but never goes above the threshold μ, so inertia uncertainty doesn't raises a false alarm.

3.4 Simulations

The fault diagnostic implementation is tested on the closed-loop spacecraft with measurement noise ($\sigma_\omega = 0.05°/\text{s}$ on the gyros, $\sigma_\theta = 0.5°$ on the angle sensors). Each fault is injected as a step bias at $t = 40\,\text{s}$.

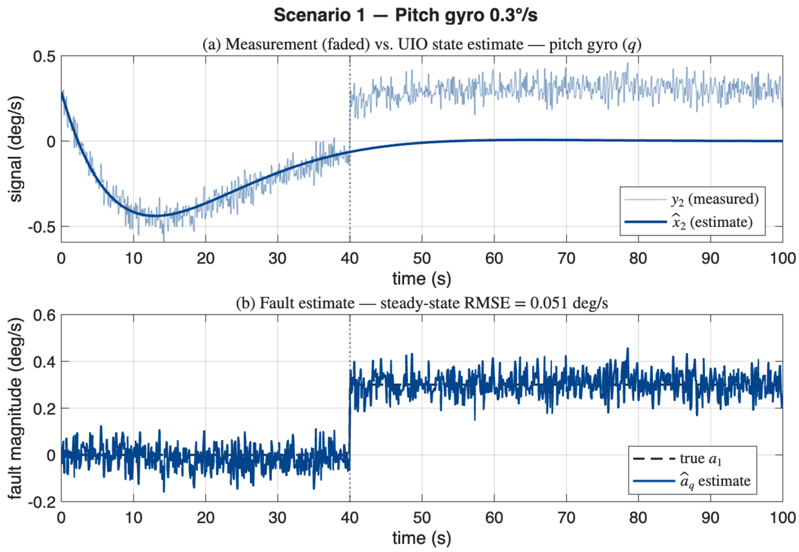

Figure 3.2: Scenario 1: pitch-gyro fault ($+0.3°/\text{s}$ on sensor 2). Top: noisy measurement and UIO state estimate of the pitch rate q. Bottom: the reconstructed fault $\hat{a}$ follows the true bias with a steady-state RMSE of $0.051°/\text{s}$, equal to the sensor noise floor.

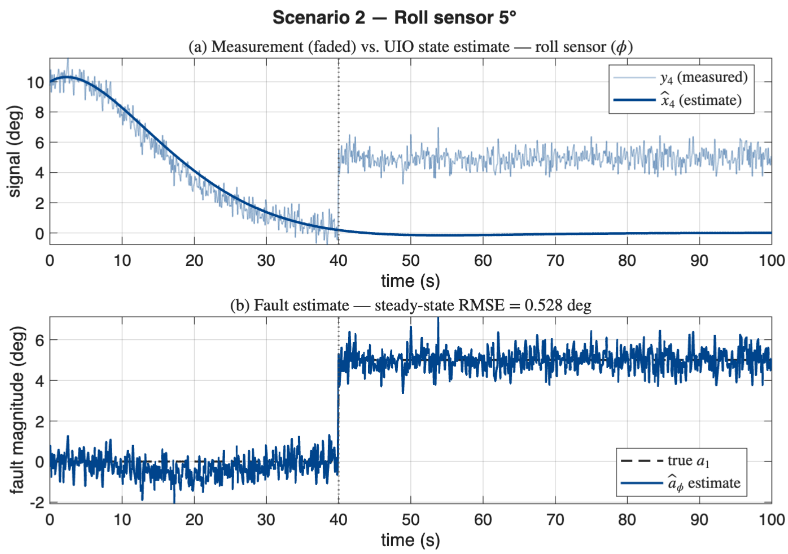

Figure 3.3: Scenario 2: roll-angle-sensor fault (sensor 4). The estimator doesn't follow the corrupted measurement and the fault estimate locks onto the bias once injected at $t = 40\,\text{s}$.

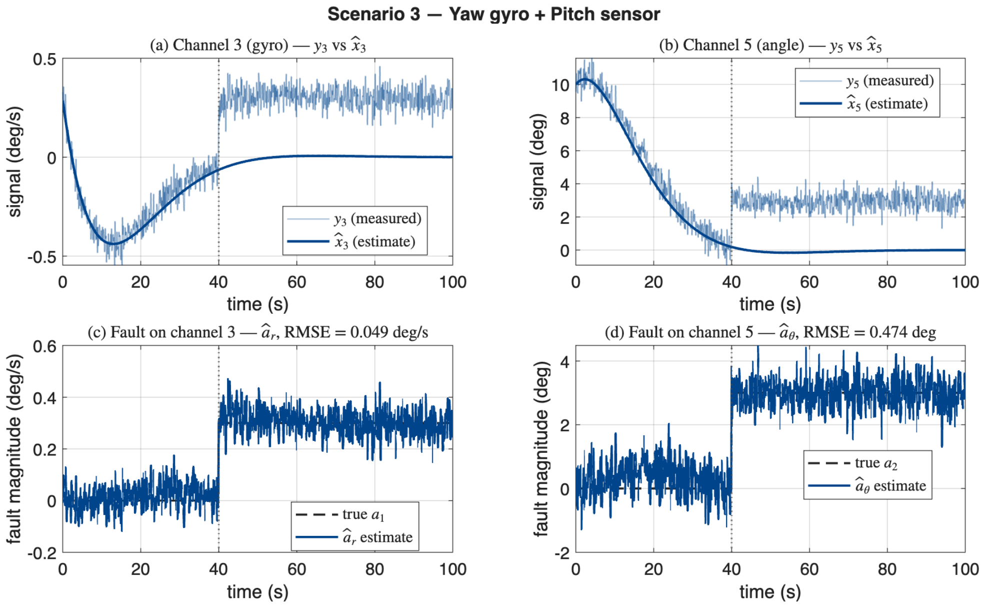

Figure 3.4: Scenario 3: two simultaneous faults, yaw gyro (sensor 3) and pitch sensor (sensor 5).

In each scenario, the residual stays close to the measurement noise level until the fault is turned on. The reconstructed $\hat{a}$ follows the injected bias within 1-2 samples, creating a good estimate. The simultaneous scenario (Figure 3.4) is the most difficult scenario, however, the diagnostic still manages to isolate which two sensors are faulty and estimate both biases separately. Each scenario also has steady-state RMSE which are at the sensor noise floor, meaning that the estimators do not add bias of their own.

Chapter 4

Secure state estimation

In this chapter, a secure state estimation algorithm is implemented. The restrictions of the initial 6-sensor system alongside a method to restore observability while one sensor is under attack are presented.

4.1 Attack model and sensors

The estimator assumes an LTI system of the form

$$x(k+1) = Ax(k) + Bu(k), \qquad y_i(k) = C_i x(k) + a_i(k), \quad i \in \{1, \dots, N\}, \tag{4.1}$$

with a separate attack signal a_i on each sensor. It is clear how many sensors can be corrupted but not which ones. The spacecraft has $N = 6$ measurements coming from three rate gyros (sensors 1–3) and three angle sensors (sensors 4–6). For this implementation, the estimator is designed to defend against a single sensor attack with $M = 1$. This choice is explained in the following section.

4.2 Observability check

A requirement of SSE is that the true state has to be fully reconstructible even after removing the compromised sensors. The system needs $N > 2M$ sensors (which in this case is satisfied, as $6 > 2(1)$), and that the pair (A_d, C_J) is fully observable for every subset J made of $N - 2M = 4$ healthy sensors.

This condition was calculated for $\binom{6}{4} = 15$ possible sensor subsets by checking the rank of the observability matrix in MATLAB with $\mathrm{obsv}(A_d, C_J)$. This showed that only three of the 15 subsets are observable. This comes from the decoupled double-integrator structure of the spacecraft. Each rotational axis has a state vector of $[\mathrm{rate}, \mathrm{angle}]^\top$. It is possible to find the rate from an angle, however, it is not possible vice versa as the initial condition of the angle would need to be known when integrating the rate. Thus if an angle sensor gets dropped, then its respective axis is unobservable. A subset is thus observable if and only if it keeps all three angle sensors $\{4, 5, 6\}$. For $|J| = 4$, this leaves exactly three viable subsets:

- $\{1, 4, 5, 6\}$ — All angle sensors plus the roll gyro.
- $\{2, 4, 5, 6\}$ — All angle sensors plus the pitch gyro.
- $\{3, 4, 5, 6\}$ — All angle sensors plus the yaw gyro.

Since the other 12 subsets drop at least one angle sensor, they fail the rank test. The current state of the open-loop system is not globally 1-fault observable, but the estimator is still implemented.

4.3 Observer bank and decoder

For every set S of $N - M = 5$ healthy sensors a Luenberger observer is ran, and for every set P of $N - 2M = 4$ sensors a second-level observer is ran. This works out to 6 S observers and 15 P observers and only 3 and 3 are observable and the rest are skipped:

$$\begin{aligned} \hat{x}_S(k+1) &= A\hat{x}_S(k) + Bu(k) + L_S\big(C_S\hat{x}_S(k) - y_S(k)\big), \\ \hat{x}_P(k+1) &= A\hat{x}_P(k) + Bu(k) + L_P\big(C_P\hat{x}_P(k) - y_P(k)\big), \end{aligned} \tag{4.2}$$

where C_S is made by stacking vertically the measurement rows corresponding to the sensors in subset S. We placed each gain L_S with MATLAB's place function so that $A + L_S C_S$ is Schur and that the clean subset's estimate converges to the true state. It is not known which subset is clean so a decoder is introduced to isolate the uncorrupted estimate via evaluating the self consistency of the observer bank:

$$\pi_S(k) = \max_{P\subset S:\,|P|=N-2M} \big\|\hat{x}_S(k) - \hat{x}_P(k)\big\|, \qquad \sigma(k) = \arg\min_{|S|=N-M} \pi_S(k), \qquad \hat{x}(k) = \hat{x}_{\sigma(k)}(k). \tag{4.3}$$

The decoder works as follows: if a clean subset agrees with all of its sub-observers, it scores $\pi_S \approx 0$, while an attacked subset scores high. The output should thus be the estimate of the lowest-scoring subset. Because the observer is stable, the estimation error of a clean subset shrinks to zero over time ($c\lambda^k$). The clean subset also excludes the hacked sensor, it is blind to the attack, meaning the attacker can inject a lot of noise and the clean observer will still converge ($\pi_S \approx 0$). At the same time, the attackers data makes the compromised observers disagree, increasing their scores. The clean subset settles as the global minimum, meaning the decoder always selects the true state.

The discrete-time observer bank is implemented by evaluating the viable subsets and skipping the 12 unobservable combinations. This observability limit means that if an attacker compromises a rate gyro (sensors 1–3), that attack is isolated, since the corrupted gyro can be dropped and still reconstruct the state. However, if an attacker compromises an angle sensor (sensors 4–6), the decoder fails. Because all viable observers are forced to use all three angle sensors, a single corrupted angle sensor reading corrupts all observers in the bank.

4.4 Restoring one-fault observability

To satisfy the requirement for global 1-fault observability, every possible sensor subset P of size $N-2M$ must yield a fully observable pair (A_d, C_P). In the initial architecture, removing a single angle sensor compromised the observability of its corresponding rotational axis. The addition of redundant measurements can eliminate this restriction. Specifically, the sensor set must be expanded such that any subset of $N - 2M$ sensors retains sufficient independent linear combinations to reconstruct the angle states (ϕ, θ, ψ) across all three axes.

By augmenting the original 6-sensor configuration with three additional cross-axis sensors, the system is expanded to $N = 9$ total sensor outputs:

$$C_{\text{aug}} = \begin{bmatrix} I_6 \\ C_7 \\ C_8 \\ C_9 \end{bmatrix}, \tag{4.4}$$

where each supplementary measurement matrix C_i links multiple state dimensions. Under this $N = 9, M = 1$ framework, the decoder evaluates $\binom{9}{7} = 36$ sensor subsets. Two distinct architectural approaches for these supplementary sensors are presented below.

State summation sensor architecture

This configuration utilizes hardware or software components that measure a linear combination of two distinct angular orientations simultaneously. The measurement vectors for the supplementary channels are defined as:

$$C_7 = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}, \quad C_8 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}, \quad C_9 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}. \tag{4.5}$$

This yields the scalar outputs $y_7 = \phi + \theta$, $y_8 = \theta + \psi$, and $y_9 = \phi + \psi$. Because these measurements form a linearly independent cycle over the attitude angles, dropping any single angle sensor $\{4, 5, 6\}$ leaves the system capable of recovering the missing state via the state summation sensors.

Compact state summation sensor

As an alternative to expanding the network to 9 sensors, full 1-fault observability can also be achieved by redesigning the primary 6-sensor suite into a highly coupled, compact measurement framework. By mapping rate-angle combinations and cross-axis sums within a single 6×6 matrix, we define:

$$C_{\text{compact}} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}. \tag{4.6}$$

Here, the first three channels act as combined rate-and-angle trackers $(r_i + \theta_i)$, while the latter three handle attitude cross-summation. This satisfies the rank condition for all $\binom{6}{4} = 15$ combinations without increasing the physical sensor count.

While the compact structure reduces the total number of sensors ($N = 6$) and subsequently lowers the computational effort of the observer bank, it introduces the following drawbacks. By coupling rate and angle states within the same measurement channel (e.g., $r_i + \theta_i$), a single sensor attack simultaneously corrupts both rate and attitude information for that channel. Additionally, losing direct physical state outputs forces the flight software to continuously process coupled matrices to extract the satellites states.

Combo Sensor Architecture

Instead of summing states into a single scalar channel, the system can alternatively incorporate multi-channel "combo" sensors (e.g., dual-axis star trackers or IMUs) that output independent vector components simultaneously. The supplementary matrices are defined as

$$C_7 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \quad C_8 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \; and \quad C_9 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.7}$$

This directly duplicates the angle sensor channels, ensuring that even if a primary angle sensor is corrupted, a clean, redundant channel for that exact axis remains active in the observer bank.

4.5 Simulations

For the simulations, the loop was closed with sensor noise and a bias was injected at $t = 40\,\mathrm{s}$. In the figures, the disagreement scores π_S are shown, the decoded subset $\sigma(k)$, and the true state vs. the secure estimate on the sensor being attacked. The state summation sensor architecture was chosen to validate the 1-fault observability and for comparison with with the initial 6-sensor approach.

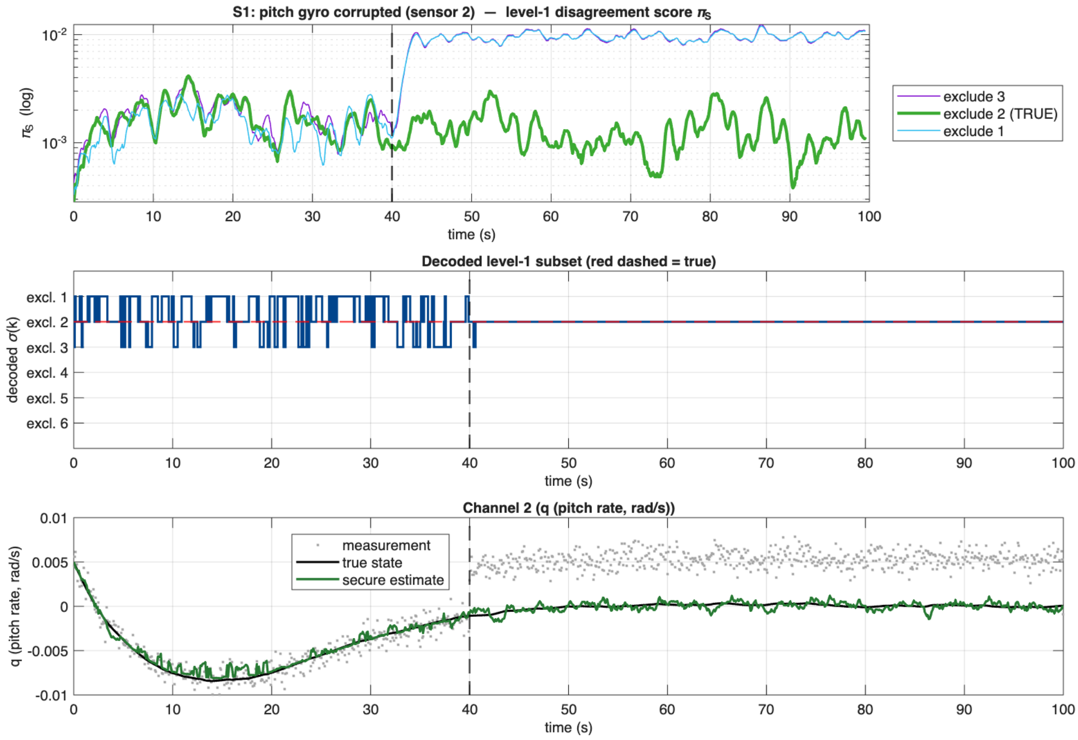

Figure 4.1: Attack on the pitch gyro (sensor 2). After $t = 40\,\mathrm{s}$ the decoder locks onto the healthy/true subset (exclude sensor 2 (since sensor 2 is corrupt)) and the secure estimate tracks the true value.

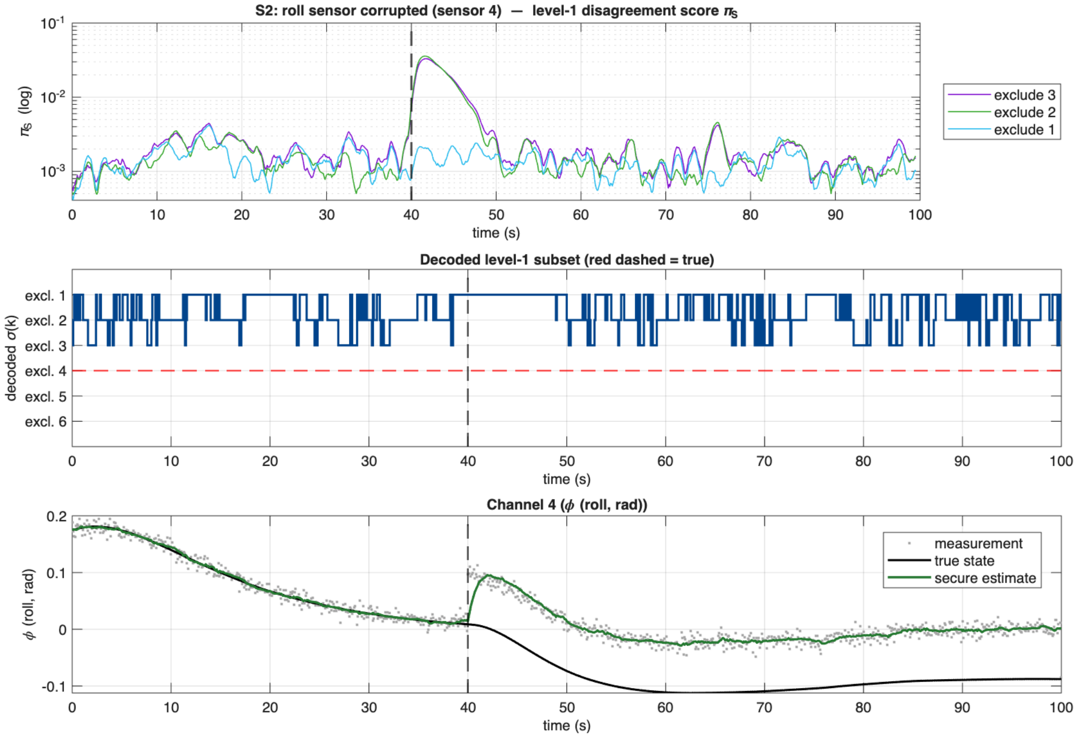

Figure 4.2: Attack on the roll angle sensor (sensor 4). Because the observable subsets have to keep all three angle sensors, the decoder chooses the wrong subset and the secure estimate is corrupted by the attack. This scenario fails.

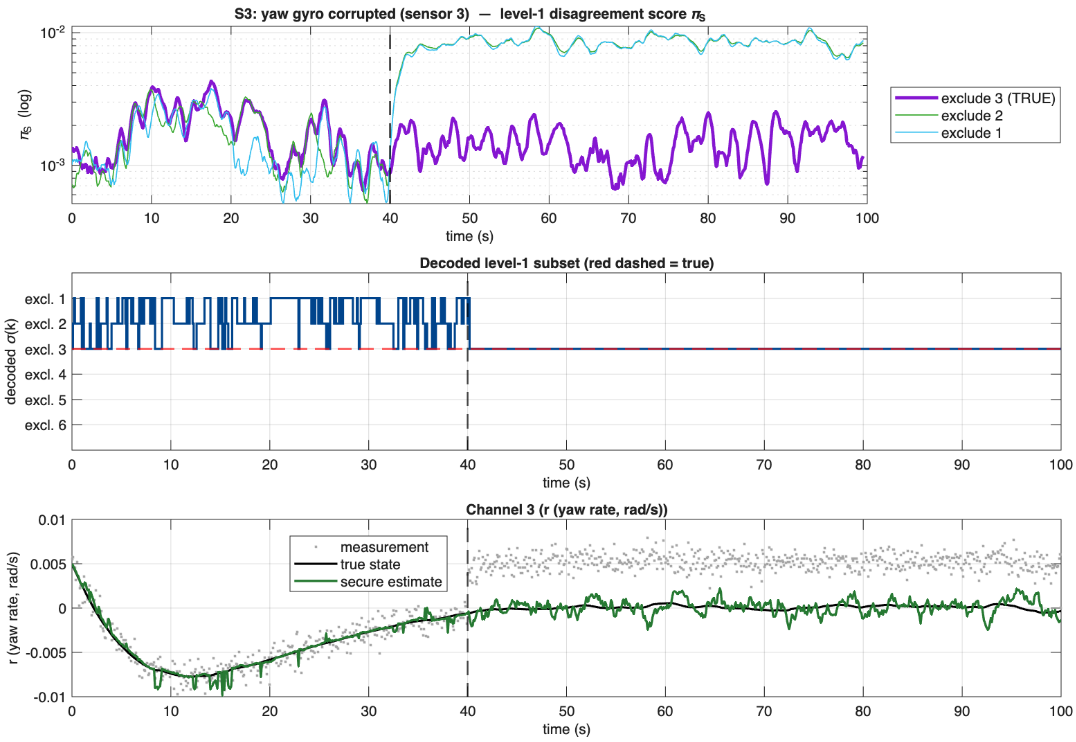

Figure 4.3: Attack on the yaw gyro (sensor 3). The decoder finds the true subset and the estimate is unaffected.

For the two rate-gyro attacks (Figures 4.1 and 4.3) the score π_S of the true subset goes to zero while the others increase, $\sigma(k)$ chooses the correct subset, and the secure estimate is the same as the true state. For the angle-sensor attack (Figure 4.2) all observable subsets are corrupted, thus no π_S stays small, and the decoder chooses the wrong subset. This is the structural limitation given only six sensors on the spacecraft.

State summation sensor simulation

To validate the proposed state summation architecture ($N = 9$), the same set of sensor attacks was executed under identical noise conditions. The resulting performance is illustrated in Figures 4.4, 4.5, and 4.6.

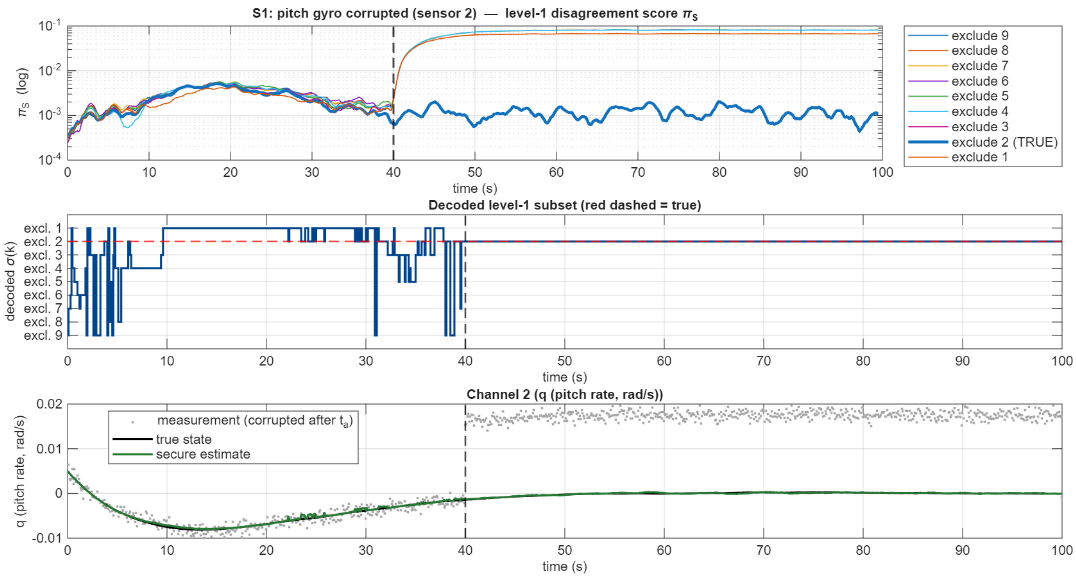

Figure 4.4: Attack on the pitch gyro (sensor 2). After $t = 40$ s the decoder follows the true fault-free subset (exclude sensor 2) and the secure estimate follows the true pitch rate.

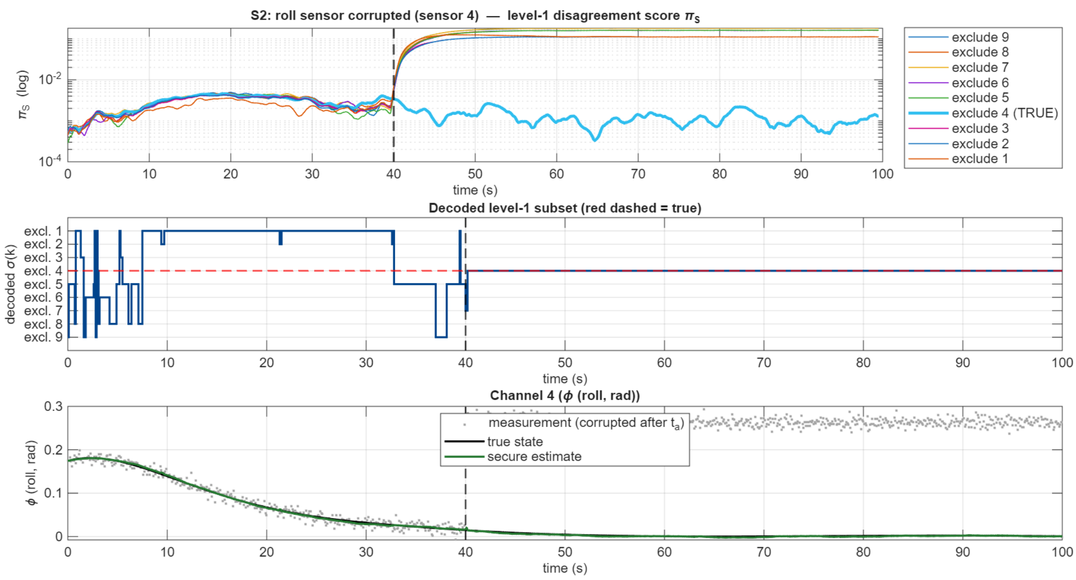

Figure 4.5: Attack on the roll angle sensor (sensor 4). As in the pitch-gyro case the decoder recovers the true subset and the estimate is unaffected.

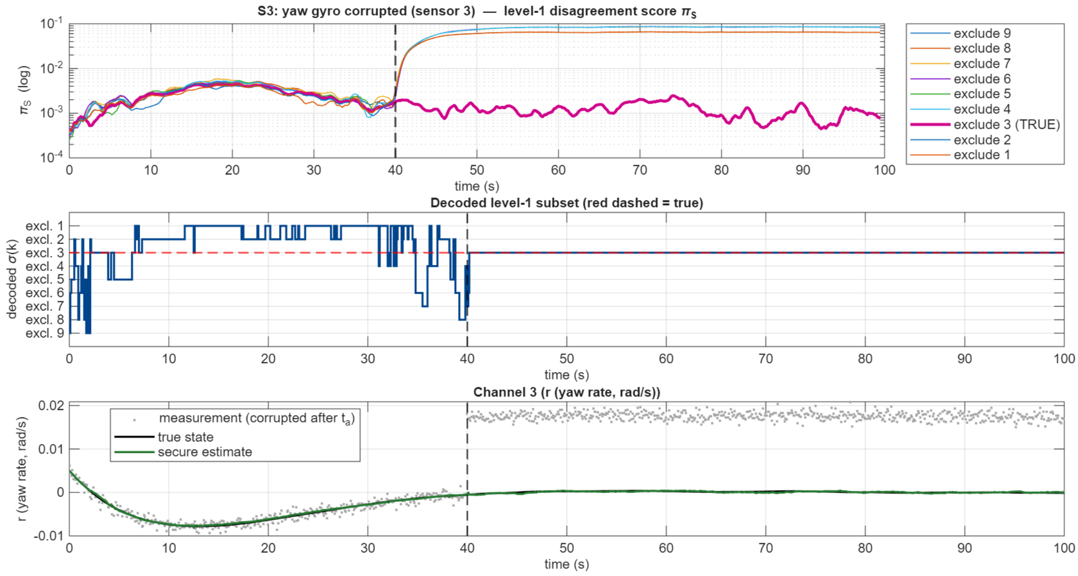

Figure 4.6: Attack on the yaw gyro (sensor 3). As in the pitch-gyro case the decoder recovers the true subset and the estimate is unaffected.

In contrast to the original 6-sensor configuration, the augmented architecture successfully maintains state estimation integrity across all scenarios. When an attack is directed at the pitch gyro (Figure 4.4) or the yaw gyro (Figure 4.6), the decoder rapidly isolates the faulted sensor, dropping its corresponding subsets and allowing the secure estimate to seamlessly track the true state.

Figure 4.5 demonstrates that the restriction regarding angle measurements has been completely resolved. When an attack is injected into the roll angle sensor (sensor 4) at $t = 40\,\text{s}$, the summation sensors (y_7, y_8, y_9) preserve the strict rank conditions for the remaining healthy subsets. The decoder successfully identifies and rejects the compromised roll sensor, ensuring the secure state estimate remains entirely uncorrupted by the attack. This confirms that the augmented system is globally 1-fault observable.

Chapter 5

Resilient control against integrity attacks

Here the nonlinear-encoding of Joo, Qu and Namerikawa [1] is implemented on the spacecraft. The aim here is to take an attack that is perfectly invisible to the detector and make it show up, with zero performance cost when there is no attack. The plant and controller are from Chapter 2. The paper was based on a continuous time system, thus the continuous time system of the spacecraft is used in this section.

5.1 Stealthy attack model

To execute a perfectly stealthy attack, the attacker must have knowledge of the system dynamics. When setting the feedback gains $K_a = K$ and zero feedthrough $H = 0$(same as D matrix), the attack dynamics are:

$$u_a = -Kx_a + r_a, \qquad \dot{x}_a = A_c x_a + B_c u_a, \qquad y_a = -C_c x_a, \tag{5.1}$$

By setting $K_a = K$, the attacker makes sure that their internal model mirrors the real closed-loop dynamics. Now the attacker can predict the exact control signal from the plant in response to the attack. Then, a sensor signal y_a can be made that cancels the physical attack in the residual.

5.2 Nonlinear encoding/decoding

The defense mechanism against a system integrity attack is outlined as a chaotic, rapidly varying non linear modulation factor that gets multiplied by the control signal at the controller side of the network. On the plant side, the modulation factor gets divided, where ξ is the plant's generated chaotic signal and $\hat{\xi}$ as the controller's synchronized copy. The encoding structure is:

$$u_{tx} = (1 + \varepsilon\hat{\xi})(-K\hat{x}), \qquad u_{rx} = \frac{u_{tx} + u_a}{1 + \varepsilon\xi}, \qquad \varepsilon|\hat{\xi}| \leq 0.5, \tag{5.2}$$

The encoding depth is set as $\varepsilon = 0.15$. Looking at (5.2), the received control signal is divided by the term $(1 + \varepsilon\xi)$ and to make sure that the system never gets divided by zero (when $\varepsilon|\hat{\xi}| = 1$). This is done with the constraint $\varepsilon|\hat{\xi}| \leq 0.5$, which makes sure the denominator stays in the range of $[0.5, 1.5]$. At the same time, the encoding depth has to be large enough to ensure that an injected attack is clearly visible in the output residual. This is further evaluated in Section 5.6. When the two Chua circuits are perfectly synchronized ($\hat{\xi} = \xi$) and there is no attack ($u_a = 0$), then the modulation factors cancel out. Thus, the plant receives the exact control effort $-K\hat{x}$. This cancellation also allows for zero effect on performance.

Since the true state x is stored on the plant side, the controller needs a Luenberger observer to reconstruct the state estimate $\hat{x}$ and to compute the control law $u = -K\hat{x}$.

This observer has the transmitted control signal u_{tx} and the received sensor measurements y available. The complete observer dynamics are defined as:

$$\dot{\hat{x}} = A_c\hat{x} + B_c \frac{u_{tx}}{1+\varepsilon\hat{\xi}} + L_c\,(y - C_c\hat{x}),$$
$$L_c = \begin{bmatrix} 0.780 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.672 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.619 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0.566 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0.513 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0.460 \end{bmatrix}. \tag{5.3}$$

The observer has to accurately model the physical dynamics of the plant to prevent estimation drift. Normally the term $B_c u$ is used to simulate the input control, but $u_{tx} = (1+\varepsilon\hat{\xi})(-K\hat{x})$ is chaotic. The controller can use its local $\hat{\xi}$ and the observer reverses the encoding:

$$\frac{u_{tx}}{1+\varepsilon\hat{\xi}} = \frac{(1+\varepsilon\hat{\xi})(-K\hat{x})}{1+\varepsilon\hat{\xi}} = -K\hat{x} \tag{5.4}$$

The correction term $L_c(y - C_c\hat{x})$ brings the state estimation error $x - \hat{x}$ to zero whenever the sensor readings are clean. The gain L_c was found with pole placement, where the observer error poles are faster than the plan poles so that $\hat{x}$ converges before estimation errors affect control performance. Furthermore, $y - C_c\hat{x}$ acts as the detector residual z. Under under no attack it goes to zero as the observer converges, but under attack it cannot since the attacker does not know ξ and therefore cannot cancel the chaotic encoding factor.

5.3 Chaotic signal generator: Chua's circuit

To make the encoded signal safe there has to be unpredictability, thus the signal is generated from a chaotic Chua circuit. Two Chua circuits are ran which act as copies. A master circuit that is set as $\xi = v_1$, and a slave on the controller side that sees only the scalar $y_c = v_1$ and locks onto the master through the coupling term, giving $\hat{\xi} = \hat{v}_1$,

$$\begin{bmatrix} \dot{v}_1 \\ \dot{v}_2 \\ \dot{I} \end{bmatrix} = \begin{bmatrix} \frac{1}{C_1}\big(\frac{1}{R_1}(v_2 - v_1) - g(v_1)\big) \\ \frac{1}{C_2}\big(\frac{1}{R_1}(v_1 - v_2) + I\big) \\ \frac{1}{L_1}(-v_2 - R_2 I) \end{bmatrix}, \quad \begin{bmatrix} \dot{\hat{v}}_1 \\ \dot{\hat{v}}_2 \\ \dot{\hat{I}} \end{bmatrix} = \begin{bmatrix} \frac{1}{C_1}\big(\frac{1}{R_1}(\hat{v}_2 - \hat{v}_1) - g(\hat{v}_1) + l_c(y_c - \hat{y}_c)\big) \\ \frac{1}{C_2}\big(\frac{1}{R_1}(\hat{v}_1 - \hat{v}_2) + \hat{I}\big) \\ \frac{1}{L_1}(-\hat{v}_2 - R_2\hat{I}) \end{bmatrix}, \tag{5.5}$$

with the piecewise-linear diode

$$g(v) = m_1 v + \tfrac{1}{2}(m_0 - m_1)(|v + B_p| - |v - B_p|), \qquad m_0 = -\tfrac{8}{7},\ m_1 = -\tfrac{5}{7},\ B_p = 1. \tag{5.6}$$

We used the canonical double-scroll values $C_1 = \frac{1}{9},\ C_2 = 1,\ R_1 = 1,\ R_2 = 0,\ L_1 = 0.07,\ l_c = 8$ taken from the simulation example in [1]. The coupling gain $l_c = 8$ is addressed separately in the next section.

5.4 Boundedness and synchronization (Lemma 2)

There are two things to consider before relying on ξ, and [1] Lemma 2 details this. The circuit state $x_c(t)$ stays bounded by a radius, $\|x_c(t)\| \leq M_c$, and the slave's tracking error decays when the coupling gain l_c is made large enough,

$$l_c \geq \frac{1}{R_1} - d + \beta \ \ (\beta > 0), \qquad \|\tilde{x}_c(t)\| \leq M_1 \|\tilde{x}_c(0)\| \, e^{-\alpha_c t}, \quad \alpha_c = \min\Big(\frac{2\beta}{C_1}, \frac{1}{R_1 C_2}, \frac{2R_2}{L_1}\Big). \tag{5.7}$$

The diode $g(v)$ is a piecewise-linear negative resistor with breakpoint voltage (separating the two linear regions) $B_p = 1$. It has a conductance (current per unit voltage) of $m_1 = -\frac{5}{7}$ in the inner region of the piecewise function $|v| \leq B_p$ and the steeper $m_0 = -\frac{8}{7}$ in the outer region $|v| > B_p$. Both the slopes are negative mean that the diode brings energy into the circuit, which is needed for chaotic-ness. The Lemma 2 proof in [1] takes the steeper outer slope as the worst-case divergence rate between master and slave, setting $d = m_0 = -\frac{8}{7}$.

The coupling gain l_c has to be large enough to converge the slave and the master faster than the divergence pushing them apart. This gives the threshold $\frac{1}{R_1} - d = 1 + \frac{8}{7} = \frac{15}{7} \approx 2.14$. Here $l_c = 8$ which is well above the threshold. Making l_c arbitrarily large should not be done because, as mentioned in Theorem 3, the tolerable communication delay shrinks as l_c grows.

As mentioned in 5.3, $R_2 = 0$ thus the third term in α_c gives $\frac{2R_2}{L_1} = 0$, so the minimum equals to zero. Thus the mathematical formula cannot give us a sync rate, meaning that this was done in simulation. Figure 5.2 shows the error $v_1 - \hat{v}_1$ decaying to zero.

5.5 Attack detection (Theorem 2)

With the encoding switched on, the attack u_a does not go through un-noticed. It gets scaled by the factor $1/(1 + \varepsilon\xi)$ which is not reproducible by the attacker. The residual is therefore only flat when there is no attack,

$$z_a = 0 \iff u_a = 0, \tag{5.8}$$

The residual gets larger the more ε is raised, until the limit of $\varepsilon|\hat{\xi}| \leq 0.5$.

5.6 Simulations

As per the paper [1], the simulations and results were conducted on the continuous-time system. The stealthy attack of (5.1) is ran on the closed loop, first with the encoding off and then on, and a parameter sweep was done on the encoding depth ε and the attack magnitude r_a.

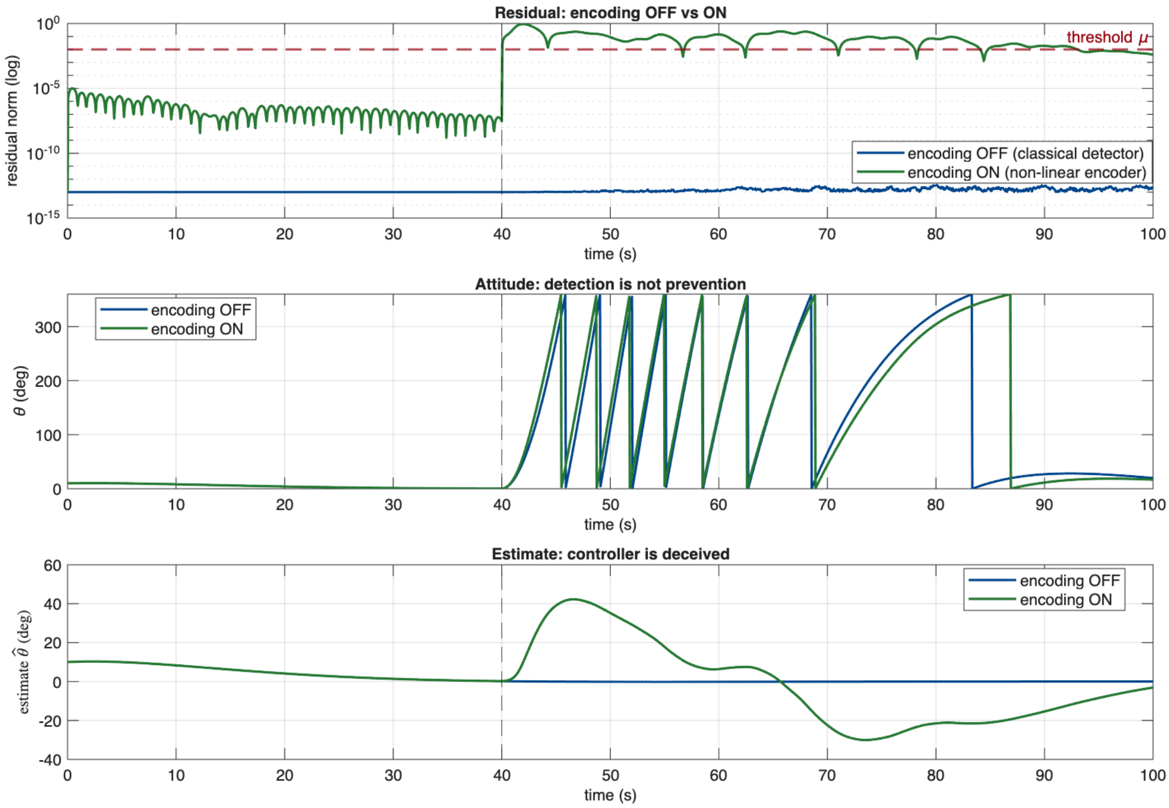

Figure 5.1: With the encoding off the residual stays at noise level before and after the attack switches on at $t = 40\,\mathrm{s}$. With the encoding turned on the residual jumps above the threshold μ. In the middle graph, it can be seen that the spacecraft still drifts off course either way(detection is not prevention). With no attack the encoded and unencoded responses are identical, so there is no performance costs. In the bottom graph: the controller's observer estimate of the pitch angle and the corrupted measurement it receives during the attack is shown.

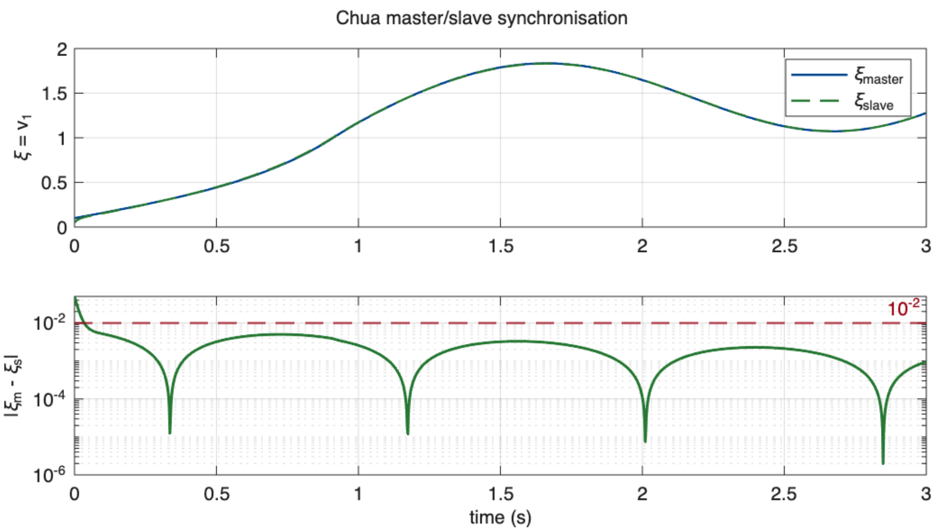

Figure 5.2: Master & slave synchronisation of the two Chua circuits.

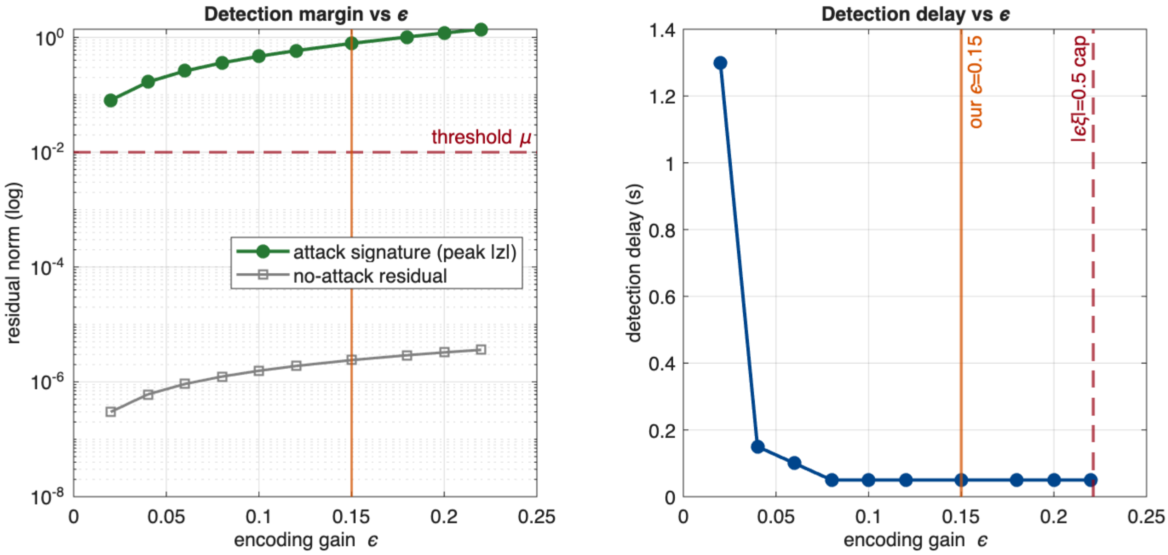

Figure 5.3: Effect of the encoding depth ε. In the left graph: the peak residual grows with ε and stays above the no attack residual and the alarm μ. In the right graph: the detection delay falls rapidly to a $\approx 0.05\,\mathrm{s}$ and the choice of $\varepsilon = 0.15$ sits in the fast detection region.

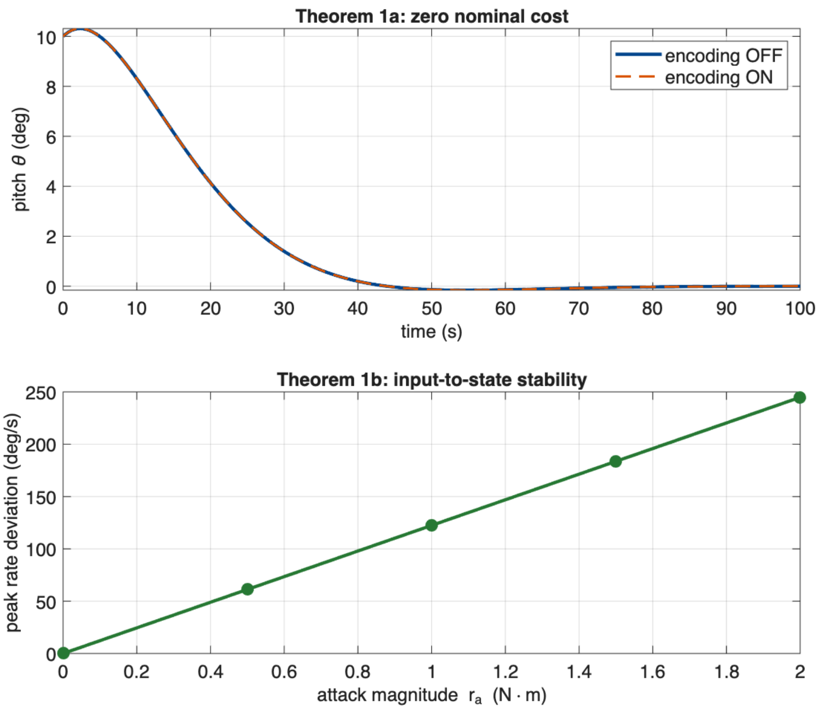

Figure 5.4: Top: With the non-linear encoding on vs. off, the attitudes are the same, there is no performance cost. Bottom: Shows input-to-state stable, where the peak rate deviation grows linearly with the attack magnitude r_a. The closed loop's response to the attack is bounded and proportional. This allows the operator to have a predictable amount of time to respond to the attack.

5.7 Future implementation

There were two areas which were explored in [1] which were out of the scope of this report. The first is implementing the non-linear encoding in discrete-time. The continuous loop could be sampled on the discrete model (2.12). This would match what was done in fault diagnostics (Chapter 3) and secure state estimation (Chapter 4), and would also expose the encoding to the measurement noise that the deterministic continuous model used here does not capture. The second area is robustness to communication delay, where Theorem 3 of [1] shows a bound for the tolerable delay τ as

$$\tau < \min\left\{\tfrac{\alpha_c}{\alpha_d},\ \tfrac{\lambda_d}{\alpha_1+\alpha_2+14}\right\}, \tag{5.9}$$

with α_c as in (5.7) and α_d growing with l_c. Thus, as mentioned earlier faster synchronisation shrinks the delay that is tolerable.

Chapter 6

Conclusion and recommendations

6.1 Conclusion

The objective of this project was to design, evaluate and diagnose faults for a spacecraft attitude system against both sensor faults and system integrity attacks. Firstly, in Chapter 2, the nonlinear dynamics of the spacecraft were linearized and then a feedback (PD) controller was designed to stabilize the system. Furthermore, the linearized system was discretized using ZOH with a sampling frequency of 10Hz. For secure state estimation (Chapter 4), the multi-observer estimator reconstructs the state by choosing the least-disagreeing sensor subset. In the spacecraft implementation, it succeeds for attacks on gyro rate sensors but, using only the six original sensors, fails for attacks on angle sensors, since every observable subset must retain all three angle measurements. Adding redundant summation sensors (y_7, y_8, y_9) restores global one-fault observability, so the augmented configuration also resolves angle-sensor attacks. For resilient control (Chapter 5), the nonlinear-encoding of [1] was implemented. As soon as the two Chua circuits synchronized, the encoding cancels exactly under no attack with no effect on performance cost. While during an attempted stealthy attack, the signal is multiplied by an uncancellable chaotic factor and becomes detectable $(z_a = 0 \Leftrightarrow u_a = 0)$. Detection does not mean prevention as the attack is revealed to the operator but not prevented from driving the spacecraft off course.

6.2 Recommendations for future work

There are several areas of improvement and future work in this implementation. The first is continuing the fault diagnostics of Chapter 3 from sensor faults to actuator faults, most notably a reaction-wheel failure or a loss of effectiveness over time. These faults are seen in the dynamics ($B_{\text{fault}} \neq 0$) and not just in the measurement, so the full unknown-input-observer design would have to be used instead of the collapsed sensor-fault form used in Section 3.2. The current implementation of the non-linear encoding of [1] has two further possible extensions. A discrete-time implementation and robustness to communication delay, which are discussed in Chapter 5. It would be a great step to combine the fault diagnosis, secure estimation and encoding into a single integrated system.

Statement on the use of AI

The following AI tools were used in the preparation of the code:

- Claude was used for graph visualization and formatting (IEEE).
- Copilot was used inside of MATLAB to help with coding errors while writing the MATLAB scripts.

Bibliography

[1] Youngjun Joo, Zhihua Qu, and Toru Namerikawa. Resilient control of cyber-physical system using nonlinear encoding signal against system integrity attacks. *IEEE Transactions on Automatic Control*, 66(9):4334–4341, 2021.

[2] F.N. Pirmoradi, F. Sassani, and C.W. de Silva. Fault detection and diagnosis in a spacecraft attitude determination system. *Acta Astronautica*, 65(5–6):710–729, 2009.