Fault-Tolerant Control of Unmanned Underwater Vehicles

Lingli Ni

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Mechanical Engineering

Chris R. Fuller, Chair
Harley H. Cudney
Daniel J. Inman
Harry H. Robertshaw
William R. Saunders

June 29, 2001
Blacksburg, Virginia

Keywords: Fault Detection and Identification, Control Reconfiguration, Unmanned Underwater Vehicles

Fault-Tolerant Control of Unmanned Underwater Vehicles

Lingli Ni

(ABSTRACT)

Unmanned Underwater Vehicles (UUVs) are widely used in commercial, scientific, and military missions for various purposes. What makes this technology challenging is the increasing mission duration and unknown environment. It is necessary to embed fault-tolerant control paradigms into UUVs to increase the reliability of the vehicles and enable them to execute and finalize complex missions. Specifically, fault-tolerant control (FTC) comprises fault detection, identification, and control reconfiguration for fault compensation. Literature review shows that there have been no systematic methods for fault-tolerant control of UUVs in earlier investigations. This study presents a hierarchical methodology of fault detection, identification and compensation (HFDIC) that integrates these functions systematically in different levels. The method uses adaptive finite-impulse-response (FIR) modeling and analysis in its first level to detect failure occurrences. Specifically, it incorporates a FIR filter for on-line adaptive modeling, and a least-mean-squares (LMS) algorithm to minimize the output error between the monitored system and the filter in the modeling process. By analyzing the resulting adaptive filter coefficients, we extract the information on the fault occurrence. The HFDIC also includes a two-stage design of parallel Kalman filters in levels two and three for fault identification using the multiple-model adaptive estimation (MMAE). The algorithm activates latter levels only when the failure is detected, and can return back to the monitoring loop in case of false failures. On the basis of MMAE, we use multiple sliding-mode controllers and reconfigure the control law with a probability-weighted average of all the elemental control signals, in order to compensate for the fault.

We validate the HFDIC on the steering and diving subsystems of Naval Postgraduate School (NPS) UUVs for various simulated actuator and/or sensor failures, and test the hier-

archical fault detection and identification (HFDI) with realistic data from at-sea experiment of the Florida Atlantic University (FAU) Autonomous Underwater Vehicles (AUVs). For both occasions, we model actuator and sensor failures as additive parameter changes in the observation matrix and the output equation, respectively.

Simulation results demonstrate the ability of the HFDIC to detect failures in real time, identify failures accurately with a low computational overhead, and compensate actuator and sensor failures with control reconfiguration. In particular, verification of HFDI with FAU data confirms the performance of the fault detection and identification methodology, and provides important information on the vehicle performance.

Acknowledgements

I would like to express my sincere appreciation to Dr. C. R. Fuller for his supervision and assistance in my research and preparation of this dissertation. His permanent support and interest enabled the presentation of this work. I also wish to thank Dr. H. H. Cudney, Dr. D. J. Inman, Dr. H. H. Robertshaw, and Dr. W. R. Saunders for serving on my advisory committee.

I would like to thank Dr. Y. A. Liu in the Department of Chemical Engineering at Virginia Tech for his patience in examining the grammar and valuable opinions in improving the readability of this dissertation. I am indebted to the Office of Naval Research for funding this work. I want to thank Dr. Kam Ng, the technical monitor of this project, for his concern in my research. I also wish to thank Dr. Anthony Healey and Dr. David Marco at Naval Postgraduate School and Dr. Edgar An at Florida Atlantic University for their support and advice during the research.

Many thanks to all my colleagues in the Vibration and Acoustics Laboratories at Virginia Tech whose friendship I always enjoy. I also wish to thank Ms. Sally Shrader, Ms. Lynda King, and Ms. Dawn Bennett for their administrative support.

Finally, I want to thank my parents, my grandmother, and my whole family for the invaluable support during my life and studies. I especially want to express my heartfelt gratitude to my husband Dongyu Wang, for his understanding and support during all these years.

Contents

List of Figures

List of Tables

Nomenclature

Dynamics and Control of UUVs

A	State transition matrix in the state space
B	Control input matrix in the state space
C	Observation matrix in the state space, or
	matrix of Coriolis and centrifugal terms in equations of motion
D	Matrix of hydrodynamic damping terms in equations of motion
$f(x)$	Nonlinear function representing disturbances and un-modeled dynamics
$\hat{f}(x)$	Estimate of $f(x)$
g	Vector of gravity and buoyant forces
H	Discretized observation matrix (same as its continuous version)
I_c	Inertia tensor referring to the body-fixed frame with origin at the center of gravity
I_{xc}, I_{yc}, I_{zc}	Moments of inertia relating to I_c
I_o	Inertia tensor referring to the body-fixed coordinate frame with origin O
I_x, I_y, I_z	Moments of inertia about X_o, Y_o, Z_o axes
K	Vector of feedback gains
M	Matrix of inertia and added inertia in equations of motion
K, M, N	Moment of external forces about the origin in the body-fixed frame
p, q, r	Rotational motion (angular velocity) about X_o, Y_o, Z_o axes
$r_G(x_G, y_G, z_G)$	Position of the center of gravity in the body-fixed frame

r_d	Desired value for the angular velocity in yaw (yaw rate)
$\dot{r}_d$	Derivative of r_d
S	Vector of coefficients defining the sliding surface
u	Control input in the state space
u_0	Constant forward speed of UUVs
$\hat{u}$	Linear feedback control law of the control input u
$\tilde{u}$	Nonlinear feedback control law of the control input u
u, v, w	Translational motion along X_o, Y_o, Z_o axes
X	Vector of state variables
X_d	Desired values of state variables
$\tilde{X}$	Vector of state tracking errors from desired
$\dot{X}$	Derivative of state variables
$\hat{X}$	Estimation of state variables
x_d	Desired value for the state variable x
$\tilde{x}$	Tracking error from its desired value
Y	Vector of observable variables
X, Y, Z	External forces in the body-fixed frame
XYZ	Earth-fixed coordinate system
$X_oY_oZ_o$	Body-fixed coordinate system
x, y, z	Coordinates of the vehicle position in the inertial frame
Φ	Transition matrix in discretized form
Γ	Control input matrix in discretized form
ϕ, θ, ψ	Attitude of the vehicle in the inertial frame
ψ_d	Desired value for the heading angle
ϕ	Thickness of sliding surface boundary layer
τ	Control input vector in equations of motion
δ_r	Rudder deflection
δ_s	Stern plane deflection

$\hat{r}, \hat{\psi}, \hat{v}$	Estimation of the yaw rate, the heading angle, and the sway velocity
$\tilde{r}, \tilde{\psi}, \tilde{v}$	Error from their desired values
$\tilde{q}, \tilde{\theta}, \tilde{z}$	Error of pitch rate, pitch angle, and depth from desired values
q_d	Desired value for the pitch rate, or the angular velocity in pitch
θ_d	Desired value for the pitch angle
$\sigma(\cdot), s$	Scalar function representing the sliding surface
λ	Control bandwidth in the sliding-mode control law
η	Tunable parameter in the sliding-mode control law

Fault Detection and Identification and Control Reconfiguration

B_F	Additive parameter changes in the control input matrix
C_0	FIR model of the normally operated system
C_j	Adaptive models represented by vectors of FIR coefficients
C_F	Additive parameter changes in the output equation
c	Normalizing constant
D_j	Difference vectors between adaptive models and the normal system
D	Vector of summations of elements in each D_j
d_j	Summation of elements in D_j
f_d	System disturbances or modeling uncertainties
$G(z)$	Function describing input-output relationship of FIR filter
$K(k), K_i(k)$	Gain matrix of the Kalman filtering (for the ith filter)
$k, k+1$	The kth sampling period, and its one-step forward time
$k+1\|k$	One-step forward prediction from time k
P	Desired-to-input cross-correlation vector
$P(k)(P_k)$	System state estimation error-covariance matrix
$P_i(k)$	Error-covariance matrix for the ith filter
$p(\theta_i\|Z_k)$	The *a posteriori* probability for the ith model up to time k

Q_k, Q_k^i	Designed covariance matrix of the system disturbances, for the ith model
R	Input autocorrelation matrix
R_k, R_k^i	Designed covariance matrix of the system measurement noises, for the ith model
$\hat{r}_i$	Residual for the ith model (equal to $\tilde{z}_{k\|\theta_i}$)
SNR	Signal to noise ratio
U_k	Vector of control inputs up to time k
u_k	Input signal at time k
V	Normalized difference vector
V_k	White noise corrupting the measurement signals
W	Vector of FIR adjustable weights
W_k	White noise disturbance to the system behavior
W_k, W_{k+1}	Old and new weights of the adaptive FIR filter
w_n	Adjustable weights of the adaptive FIR filter
X_k	Vector of state variables at time k
$\widehat{X}(k), \widehat{X}_i(k)$	State estimation at time k, for the ith hypothesized model
y_k	Output signal from the FIR filter
Z_k	System measurement history up to time k
z_k	System measurement at time k
$\hat{z}_{k\|\theta_i}, \tilde{z}_{k\|\theta_i}$	Estimation and residual of the system measurement at time k for the ith model
θ_k	Uncertain parameters in the modeling at time k
θ_i	Parameter of the ith hypothesized failure mode
ϵ_k	Error signal between the FIR output and the desired response
$\xi(k)$	Mean-square error
$\nabla\xi(k)$	Gradient vector in the mean-square-error performance surface
$\nabla\hat{\xi}(k)$	Estimate of the gradient vector $\nabla\xi(k)$
μ	Step-size parameter that controls the stability and the rate of decent
$\Omega_{k\|\theta_i}$	Covariance of the innovation, or residual

Chapter 1

Introduction

The deep blue sea has always been a place that arouses curiosity and imagination. Mankind has been trying to explore and exploit this mysterious part of the world for decades. The advent of underwater vehicles improves our ability to understand the undersea world. Unmanned Underwater Vehicles (UUVs) fall between two extremes of underwater vehicles: the Remotely Operated Vehicles (ROVs) and the torpedo. They are being widely used in commercial, scientific and military missions for search and survey, decoy and outboard sensors, ocean engineering work service, swimmer support, test and evaluations. With increasing mission durations in these applications, one of the primary concerns is the fault occurrence in actuators, sensors, or components. When failures occur and result in abnormal operations, the only present solution is to abort the mission, and use a damage control to make UUVs surface. Therefore, the problem of reliability and security of UUVs, especially their ability of fault tolerance, has become a major concern. Even though most UUVs use adaptive control systems, the response of the controller is reactive, and no consideration is given to the source or extent of the failures. It is desirable to incorporate a function of fault detection and identification into the control system, so that we can detect and identify actuator and/or sensor failures, and design compensation measures. This is the so-called fault-tolerant control (FTC) [1,2].

Fault tolerance in dynamic systems is traditionally achieved through the use of parallel redundancy, or hardware redundancy. It uses sensors and actuators in a triplex or quadruplicate redundancy configuration and compares redundant outputs or measurements for consistency. We can apply a voting logic to select a signal with a middle value, so that a single channel or a double-failed channel never affects the plant. We can also declare that a sensor is faulty if its signal deviates too far from the average value of others, assuming that the others remain within a small difference from one another. Parallel redundancy protects against control system component failures or sensor failures in a passive way, since the system remains insensitive to failures. This approach to fault tolerance is straightforward to apply, and it is essential in the control of aircraft, space vehicles, and certain process plants such as nuclear power plants that are safety critical.

The major problems encountered with the hardware redundancy are the extra cost of redundant hardware, weight penalties, and additional space required to accommodate the equipments. With advances in digital computers, fault detection and identification (FDI) methods using analytical redundancy have emerged as practical alternatives to hardware redundancy. Analytical redundancy is basically a signal processing technique employing state estimation, parameter estimation, adaptive filtering, variable threshold logic, statistical decision theory, and various combinatorial and logical operations. Analytical redundancy improves the capability of fault tolerance with fewer additional components and increased computation. This study develops new techniques of FTC in the realm of analytical redundancy.

1.1 Literature Review

A typical FTC system includes a FDI system and an on-line control reconfiguration and fault compensation system. Some literature focuses on the problem of FDI, while other literature investigates control reconfiguration. This section briefly reviews both aspects.

1.1.1 Fault Detection and Identification Methods

FDI methods using analytical redundancy have been investigated extensively in complex dynamic systems, such as the aviation and aerospace fields, nuclear process plants, chemical process plants, and transport technology [3]. The approaches fall into two major groups, model-free and model-based methods [4].

The model-free FDI method does not use a dynamic model of the plant. It is well-suited for large scale systems where the development of a model is too time-consuming. Stengel [2] gives a good review of the fault-tolerant control system using the expert system and artificial neural network.

The expert system offers a useful formalism for fault-tolerant control because it can consider diverse data sources and sub-problem abstractions. Furthermore, the expert system can combine qualitative and quantitative reasoning, heuristics, and statistics. An expert system performs deduction using knowledge and beliefs expressed as parameters and rules. It can be implemented as a production system or a rule-based system consisting of a database, a rule base, and a rule interpreter (or an inference engine). A rule-based fault-tolerant control system contains an expert system performing fault detection, identification and reconfiguration logic as an adjunct to the nominal control structure. However, generating the parameters and rules and teaching them to the expert system is a major concern.

Artificial neural networks consist of nodes that simulate the neurons and of weighting factors that simulate the synapses of a living nervous system. They are intrinsically nonlinear. They can address problems of high dimension, and learn from experiences. Neural networks can be applied to fault detection and identification by mapping data patterns, or feature vectors associated with failures onto detector/identification vectors. For m failure modes, either m neural networks with scalar outputs are employed, or a single neural network with an m-dimension vector output is used.

Neural networks trained to detect failures would learn little from monitoring normally operating plants. A training set must contain desired outputs as well as available inputs, and either use off-line learning [5–8], or allow on-line learning [9,10]. In any case, the training of neural networks can be quite time-consuming. An immediate application of neural networks in fault-tolerant control systems is to approximate other FDI approaches.

The model-based method utilizes the idea of generating residuals, that reflect the inconsistency between the actual and estimated behavior [4]. Incorporating a model for the system dynamics facilitates early FDI and control reconfiguration in complex systems. To generate residuals, we may use the techniques of parity equations [11], eigenstructure-assignment method [12], state estimation, and parameter estimation.

State-estimation method uses a state estimator to generate residuals, and processes state estimations and residuals by testing methods to identify the failure [13–15]. Among various state-estimation methods, the multiple-model method using a bank of parallel Kalman filters has been widely used for fault detection, identification in various dynamic systems [16–19]. de Benito [20] presents an on-board and real-time failure detection and diagnosis (FDD) technique for sensor failures of an active suspension system. He represents sensor failures by abrupt changes at random instants of time in the parameters of the output equation of the system. Menke and Maybeck [21] apply multiple-model adaptive estimation (MMAE) to the Variable In-flight Stability Test Aircraft (VISTA) F-16 flight control system. They embed the MMAE algorithm into the front end of VISTA F-16 control system to provide fault-detection capacity, and use the resulting state estimates in the control system. The multiple-model method is that it not only triggers fault alarms, but also determines the specific failure among a hypothesized set of all possible failure modes. However, the method involves heavy computational load, especially when the number of hypothesized models is large.

The category of parameter estimation [3,22] uses system identification techniques to adaptively estimate model parameters based on measurements of input and output signals of the system, and analyzes the resulting parameters to extract distinct features for fault detection. We can monitor the condition of various components of the plant using estimated parameters related to specific subsystems, and detect failure status with the help of decision functions. Previous studies [23,24] apply parameter estimation methods for fault detection in UUVs. These studies employ an auto-regressive model to represent the dynamic relationship between input and output signals, different ways to minimize the square error of the parameter estimation, and different decision functions to detect failures.

Healey and Marco [23] use a Kalman filter to identify the key parameters indicating UUV subsystem's operational health, and train a neural network to detect and isolate any degraded operational mode. They point out that the most important part of the neural network design is to specify the input training data, and define the varying ranges of operating parameters for normal as well as abnormal operations. Using neural network requires extensive off-line training of network parameters.

Rae [24] develops fast and slow models to monitor the UUV using a recursive least-square algorithm. By comparing constantly updated models in fast and slow modeling processes, he obtains coefficient differences of the two models to analyze changes in the vehicle performance. He later correlates these changes with the physical causes of the problem. Since it is hard to determine the number of time steps to give a reasonable estimate of the parameter vector for a constantly updated model, the determination of a suitable time interval in-between two adjacent models is empirical. Apparently, large time intervals may cause false alarms, or delays in fault detection, while small time intervals may increase the computational load. Furthermore, the algorithm requires the number of fast models in-between two adjacent slow models.

The state-estimation method monitors the system status based on an analytical model, whereas the parameter estimation method determines the system status based on system identification, where the residuals refer to plant parameters, rather than state variables. The parameter-estimation method does not require the analytical model in advance. This is especially helpful in those areas where analytical models are not easy to develop.

1.1.2 Control Reconfiguration Methods

Control reconfiguration is to modify the control system's structure, laws, and parameters according to the result of on-line FDI.

One approach to control reconfiguration is to use multiple-model adaptive control (MMAC) [25–28]. Maybeck and Pogoda [29] apply an MMAC algorithm with four elemental controllers to sensor/actuator failure detection and control reconfiguration in a STOL F-15 aircraft. They design four elemental controllers for a healthy aircraft, the failed pitch rate sensor, the failed stabilator, or the failed pseudo surface and trailing edge flaps. Maybeck [30] extends this application of MMAC algorithm with seven elemental controllers for a healthy aircraft, three actuator failures, or three sensor failures. These researchers design PI controllers for each of the elemental controllers via linear-quadratic-gaussian(LQG) synthesis of the feedback gains. The MMAC adaptation mechanism is a selection among these elemental controllers according to the relative possibilities assigned to them. The key point of MMAC algorithm is to design each elemental controller providing desirable vehicle behavior for a particular failure status of sensors and actuators.

A second approach to control reconfiguration is to alter the control system for specific failures. Noura et al. [31] use this approach in the fault-tolerant control of a winding machine. By estimating magnitudes of the failure, they add an additional control law to the nominal one to thwart the fault effect on the system. They design different control laws for different failures. Therefore, the method needs to identify actuator failures from sensor failures. The paper also describes the concept of FTC, and classifies fault-tolerant controllers.

Cheng and Leonard [32] design a control strategy to compensate for a stuck-fin failure on the 21UUV of Naval Undersea Warfare Center (NUWC). The strategy combines a sliding-mode control with a modified fin mixing and a method for handling fin deflection saturation. With the knowledge of which fin has failed and at what deflection angle, the fin mixing scheme map the effective fin deflections resulting from the sliding-mode control to the remaining three physical fin deflections. When any of the remaining fins saturates, the control strategy amend the control input with a dynamic controller that drives the fins away from saturation while keeping the vehicle under stable control.

Yang, Yuh, and Choi [33] present an experiment study of fault-tolerant systems for Omni-Directional Intelligent Navigator (ODIN), a six-degree-of-freedom autonomous underwater vehicle designed at the University of Hawaii. They design a fault-tolerant system for thrusters and sensors, respectively. The fault-tolerant system for thrusters consists of a thruster control matrix (TCM) that relates the input force, or the effective thruster force from the controller to the physical thruster force. Once a faulty thruster is detected, the system accommodates the failure by eliminating the corresponding column in TCM. For thruster failure detection, they equip a Hall-effect sensor to each thruster to measure output voltages from thruster motors. The input voltage to the thruster motor is the desired voltage calculated through the controller and TCM. If the error between the input and output voltages do not stay within a tolerable limit, then the thruster is concluded to be faulty. The fault-tolerant system for sensors involves redundant sensors and an analytical sensor model. Sensor failures are detected through a voting technique. In the case of a sensor failure, the system simply switches from the faulty sensor to the good one. The fault-tolerant system design in this paper needs thruster and sensor redundancy. The thruster failure accommodation for ODIN is based on the reconfiguration of the TCM. The idea is similar to Cheng's fin failure compensation [32] in that both methods accommodate actuator failures by mapping the effective deflections or forces from the controller to the remaining physical actuators.

Another approach to control reconfiguration is to use adaptive and learning control. It employs neural networks or fuzzy logic [25] to develop a nonlinear model, and updates the model as new information is received to adjust the control. An example of this approach is a fuzzy model reference learning controller [34–37].

Fault-tolerant control systems can benefit from blending the perspectives of intelligent control. Passino [37] describes some of the limitations of current approaches, and states that a combination of intelligent and conventional control methods may be the best way to implement autonomous control.

The above review shows many theoretical results and application cases in fault detection, identification, and compensation. However, from the application viewpoint, the main area of these publications is in flight-control, only a few studies have been devoted to UUVs. Rae's work [24] applies the parameter estimation to UUVs for damage detection. It does not cover how to identify the nature of failures through examining coefficient differences between fast and slow models, nor mentions how to compensate the failures. Cheng and Leonard [32] assume that an identification package is available to locate the specific stuck fin prior to developing their fin failure compensation approach. Yang, Yuh, and Choi [33] design fault-tolerant systems for thrusters and sensors, respectively. In summary, the literature survey suggests that there have been no systematic methods for fault-tolerant control of UUVs. Our research aims at extending this field beyond its current boundary.

1.2 Scope of the Work

The major objective of this work is to develop a robust and computationally efficient method of fault detection, identification and compensation for UUVs. The algorithm should be a real-time method that would process data from sensory returns, including gyro rates, positions, and control surface inputs, to determine the current operational status of vehicles' major subsystems, and that would identify actuator and/or sensor failures and reconfigure the control algorithm to allow UUVs to carry out preprogrammed missions even in the event of partial actuator and/or sensor failures.

We formulate the problem base upon Naval Postgraduate School (NPS) UUVs. First, we develop sliding-mode controllers for decoupled UUV subsystems. Then, we propose a fault detection method of adaptive LMS modeling and analysis to detect failures, and design a hierarchical methodology for fault detection and identification (HFDI). Lastly, we reconfigure the control system according to FDI results to compensate for actuator and/or sensor failures.

In addition to designing the FTC system and investigating the improvement of the algorithm, we test the HFDI and the adaptive control reconfiguration method through numerical simulations of the steering and the diving subsystems of NPS UUVs in different fault scenarios. We also test the HFDI approach with realistic at-sea test data of the Ocean Explorer (OEX) series AUVs developed at the Ocean Engineering Department of Florida Atlantic University (FAU). Both cases demonstrate the effectiveness of the developed methods.

1.3 Dissertation Outline

The organization of the dissertation is as follows:

Chapter 2 describes the dynamic system and the control. First, it introduces the dynamics of UUVs, and presents simplified rigid-body equations of motion for decoupled NPS UUV steering and diving subsystems. These equations provide the model on which the fault-tolerant control will be applied. As mentioned above, we are concerned about actuator and sensor status in this work. Then, the chapter presents the concepts of sliding-mode control, and sliding-mode controllers designed for the steering and the diving subsystems, respectively. On the basis of these discussions, we can formulate the FDI problem of these subsystems, and develop FDI techniques successively.

Chapter 3 presents a new method for fault detection. In particular, it addresses the adaptive least-mean-square (LMS) modeling and the adaptive analysis of the model history. In this chapter, we first describe the modeling of actuator and sensor failures, and the formulation of FDI problem. Then, after developing the LMS method we apply it to detect simulated actuator and/or sensor failures in NPS UUV subsystems.

Chapter 4 presents a hierarchical approach to fault detection and identification (HFDI) on the basis of Chapter 3 and the method of MMAE. It reviews the component methods and describes the hierarchical methodology. Finally, it presents simulation results of applying the HFDI to NPS UUV subsystems.

Chapter 5 discusses the re-configurable control scheme of the FTC system. It presents a multiple-model adaptive sliding-mode control based on the HFDI developed in Chapter 4. The discussion covers the overall results of FDI and fault compensation obtained on simulated NPS UUV subsystems.

Chapter 6 verifies the HFDI method with real data of Florida Atlantic University (FAU) AUVs. We formulate the FDI problem according to subsystem models of FAU AUV, and use the actual control signal and measurements in the HFDI. We present verification results at the end of this chapter.

Finally, Chapter 7 summarizes the main techniques and results of the hierarchical fault detection, identification, and compensation methodology, and recalls important conclusions of the study. The chapter also outlines several directions for future

Chapter 2

Decoupled Sliding-Mode Control of NPS UUVs

The first step towards fault-tolerant control of UUVs involves modeling and control. A complete discussion of the theory of UUV modeling is beyond the scope of the present analysis. It is useful, however, to present the fundamental concepts of modeling issues. The first section of this chapter introduces the derivation of equations of motion of UUVs. Section 2.2 describes the simplified, decoupled models of NPS UUV subsystems. After introducing the concepts of sliding-mode control, Section 2.3 discusses designing sliding-mode controllers for each subsystem. Section 2.4 presents simulation results of sliding-mode control of NPS UUV subsystems.

2.1 Underwater Vehicle Dynamics

Modeling of UUVs, like the modeling of all other marine vehicles, discusses the motion in six degrees of freedom (DOF). Figure 2.1 is a picture of the NPS UUV. The six motion components of marine vehicles are: surge, sway, heave, roll, pitch and yaw. Table 2.1 shows the notations.

Figure 2.1: Picture of NPS UUV

Table 2.1: Notation Used for Marine Vehicles

6 motion components	forces and moments	linear and angular velocities	positions and Euler angles
surge (motions in the x-direction)	X	u	x
sway (motions in the y-direction)	Y	v	y
heave (motions in the z-direction)	Z	w	z
roll (rotation about the x-axis)	K	p	ϕ
pitch (rotation about the y-axis)	M	q	θ
yaw (rotation about the z-axis)	N	r	ψ

Six independent coordinates determine the position and the orientation of the vehicle. The first three coordinates (x, y, z) represent the position. Their time derivatives (u, v, w)

describe the translational motion along the x-, y-, and z- axes. The last three coordinates (ϕ, θ, ψ) are Euler angles representing the orientation. Their time derivatives (p, q, r) describe the rotational motion. These quantities are defined according to the Society of Naval Architects and Marine Engineers (SNAME) notation [38].

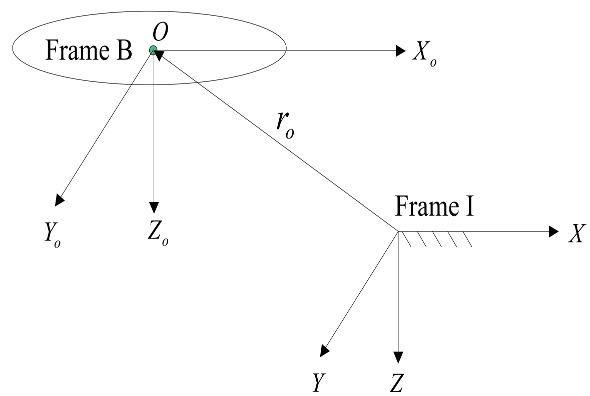

Figure 2.2: Body-Fixed (Frame B) and Inertia-Fixed (Frame I) Reference Frames

Figure 2.2 indicates two coordinate frames: the body-fixed reference frame B $(X_oY_oZ_o)$, and the inertia-fixed frame I (XYZ). We describe the motion of the body-fixed frame relative to the inertial reference frame. Usually, we choose the origin O of frame B at the center of gravity (CG) when the CG is in the principal plane of symmetry. Otherwise, we choose another convenient point for O. The body-fixed axes X_0, Y_0, Z_0 coincide with the principal axes of inertia with the longitudinal axis X_0 pointing from aft to fore, the transverse axis Y_0 to starboard, and the normal axis Z_0 from top to bottom. We describe the position and orientation of the vehicle $(x, y, z, \phi, \theta, \psi)$ relative to frame I, while expressing the linear and angular velocities of the vehicle (u, v, w, p, q, r) in the body-fixed coordinate system.

Let ν be the vector of linear and angular velocity of the vehicle in the body-fixed frame, and η be the vector of position and attitude of the vehicle in the inertial frame.

$$\nu = [\nu_1^T, \nu_2^T]^T = [u, v, w, p, q, r]^T \tag{2.1}$$

$$\eta = [\eta_1^T, \eta_2^T]^T = [x, y, z, \phi, \theta, \psi]^T \tag{2.2}$$

The Euler angular transformation in Eq. (2.3) relates these two vectors [39]. It gives the kinematic transformation between frame I and frame B.

$$\dot{\eta} = J(\eta)\nu = \begin{bmatrix} J_1(\eta_2) & 0 \\ 0 & J_2(\eta_2) \end{bmatrix} \nu \tag{2.3}$$

where

$$J_1(\eta_2) = \begin{bmatrix} c\psi c\theta & -s\psi c\phi + c\psi s\theta s\phi & s\psi s\phi + c\psi c\phi s\theta \\ s\psi c\theta & c\psi c\phi + s\psi s\theta s\phi & -c\psi s\phi + s\psi c\phi s\theta \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix} \tag{2.4}$$

and

$$J_2(\eta_2) = \begin{bmatrix} 1 & s\phi t\theta & c\phi t\theta \\ 0 & c\phi & -s\phi \\ 0 & s\phi/c\theta & c\phi/c\theta \end{bmatrix} \tag{2.5}$$

2.1.1 Equations of Motion for UUVs

Fossen [39] presents the 6-DOF nonlinear dynamic equations of motion in a compact form:

$$M\dot{\nu} + C(\nu)\nu + D(\nu)\nu + g(\eta) = \tau \tag{2.6}$$

where M is the matrix of inertia and added inertia, C is the matrix of Coriolis and centrifugal terms, D is the matrix of hydrodynamic damping terms, g is the vector of gravity and buoyant forces, and τ is the control-input vector describing the forces and moments acting on the vehicle in the body-fixed frame.

Figure 2.3 illustrates the position of the origin relative to the center of gravity in frame B. We can simplify the general rigid-body equations of motion by choosing the origin of the body-fixed coordinate system [39].

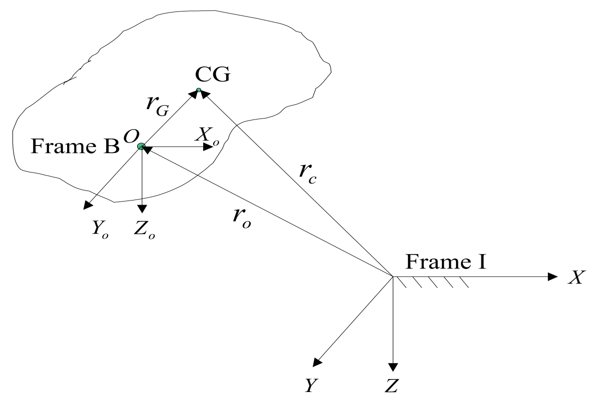

Figure 2.3: Origin and Center of Gravity in Body-Fixed Frame

Let us define the body's inertia tensor corresponding to the body-fixed coordinate system X_o, Y_o, Z_o with origin O as:

$$I_o = \begin{bmatrix} I_x & -I_{xy} & -I_{xz} \\ -I_{yx} & I_y & -I_{yz} \\ -I_{zx} & -I_{zy} & I_z \end{bmatrix} \tag{2.7}$$

Here I_x, I_y, and I_z are the moments of inertia about the X_o, Y_o, and Z_o axes, respectively. $I_{xy} = I_{yx}, I_{xz} = I_{zx}$, and $I_{yz} = I_{zy}$ are products of inertia.

When the body axes coincide with the principal axes of inertia, or the longitudinal, lateral and normal symmetry axes of the vehicle, we can choose the origin of the body-fixed coordinate system such that the inertia tensor I_o is diagonal. We can perform a principal axis transformation if the body axes do not coincide with the principal axes of inertia. In such an occasion, Fossen [39] simplifies the rigid-body equations of motion as:

$$m[\dot{u} - vr + wq - x_G(q^2 + r^2) + y_G(pq - \dot{r}) + z_G(pr + \dot{q})] = X \tag{2.8}$$

$$m[\dot{v} - wp + ur - y_G(r^2 + p^2) + z_G(qr - \dot{p}) + x_G(qp + \dot{r})] = Y \tag{2.9}$$

$$
\begin{aligned}
m[\dot{w} - uq + vp - z_G(p^2 + q^2) + x_G(rp - \dot{q}) + y_G(rq + \dot{p})] &= Z \qquad (2.10)\\
I_x\dot{p} + (I_z - I_y)qr + m[y_G(\dot{w} - uq + vp) - z_G(\dot{v} - wp + ur)] &= K \qquad (2.11)\\
I_y\dot{q} + (I_x - I_z)rp + m[z_G(\dot{u} - vr + wq) - x_G(\dot{w} - uq + vp)] &= M \qquad (2.12)\\
I_z\dot{r} + (I_y - I_x)pq + m[x_G(\dot{v} - wp + ur) - y_G(\dot{u} - vr + wq)] &= N \qquad (2.13)
\end{aligned}
$$

Here m is the mass of the body, x_G, y_G and z_G are the coordinate components of r_G.

Another case is to choose the center of gravity (CG) as the origin O such that $r_G = [0, 0, 0]$. It is usually the case for UUVs due to the symmetric shape. When the body axes coincide with the principal axes of inertia, it implies that the inertial tensor about the body's center of gravity is diagonal, i.e. $I_C = diag(I_{x_C}, I_{y_C}, I_{z_C})$. This approach yields the following simple representation [39]:

$$
\begin{aligned}
m(\dot{u} - vr + wq) &= X; \qquad I_{x_C}\dot{p} + (I_{z_C} - I_{y_C})qr = K \qquad (2.14)\\
m(\dot{v} - wp + ur) &= Y; \qquad I_{y_C}\dot{q} + (I_{x_C} - I_{z_C})rp = M \qquad (2.15)\\
m(\dot{w} - uq + vp) &= Z; \qquad I_{z_C}\dot{r} + (I_{y_C} - I_{x_C})pq = N \qquad (2.16)
\end{aligned}
$$

With the above rigid body equations of motion as the modeling background of underwater vehicles, we discuss control concepts in the following sections.

2.2 Decoupled Subsystems of NPS UUVs

A controller designed for UUVs first needs to be robust enough to deal with modeling uncertainties. On the other hand, it needs to be relatively simple so that the computational time required to calculate the control signal is short, and we can implement the control algorithm using a high sampling rate. In this study, we use the sliding-mode control to design autopilots for NPS UUVs.

A control strategy is to decouple the six-DOF equations of motion into non-interacting or lightly interacting subsystems, and design one controller for each subsystem. For decoupled control system design, we group together related equations of motion for separate functions of steering, diving, and forward speed [40,41]. We can find similar decomposition in the work of Jalving [42], Rodriguez and Dobeck [43], Russel and Bugge [44]. This decomposition leads to simplified models of UUVs and separate designs of control system for the steering, the diving, and the speed subsystem. This decoupled control philosophy maps with the current practice in naval submarines. While the control of all six degrees of freedom may be important for future UUV operation, and particularly in the transition from cruise to hover mode, it is not the primary focus in this research due to its complexity. Table 2.2 summarizes control and state variables for specific subsystems. Each subsystem has a single control element and multiple states. The steering subsystem controls heading errors, the diving subsystem controls depth and pitch errors, and the speed subsystem controls the propeller dc motor. We leave the rolling mode passive in this control. In many UUV applications, it is reasonable to assume that the vehicle is in level flight experiencing no roll.

Table 2.2: Control and State Variables of UUV Subsystems

	State Variables	Control Inputs
Speed Subsystem	$u(t)$	$n(t)$
Steering Subsystem	$v(t), r(t), \psi(t)$	$\delta_r(t)$
Diving Subsystem	$w(t), q(t), \theta(t), z(t)$	$\delta_s(t)$

In the following sections, we will describe dynamic models for the steering and the diving subsystems of NPS UUVs, and design separate sliding-mode autopilots based on their dynamic models.

2.2.1 Steering Subsystem of NPS UUVs

Figure 2.4 presents the model of NPS UUV steering subsystem. The input to the steering subsystem is a control command for deflection of bow rudders and stern rudders. The output of the steering subsystem is the yaw rate of turn.

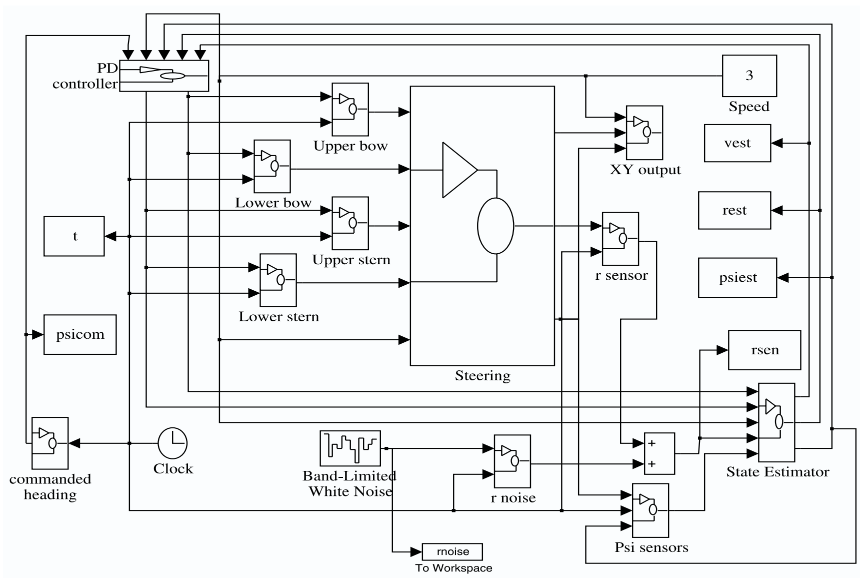

Figure 2.4: Steering Subsystem of NPS UUVs

There are two bow rudders: the upper-bow rudder and the lower-bow rudder, and two stern rudders: the upper-stern rudder and the lower-stern rudder. It is assumed that bow rudders and stern rudders have identical size and shape, and take the deflection command equally, but in opposite direction. Figure 2.5 illustrates the opposite direction of bow and stern rudders.

There are three state variables for the steering subsystem: r, ψ, and v. r represents the yaw rate of turn, which is obtained by a rate gyro or a rate sensor; ψ represents the heading angle, which is measured by a compass; and v represents the sway velocity, which can be obtained from state estimation.

Fossen [39] expresses the linear steering equations of motion as:

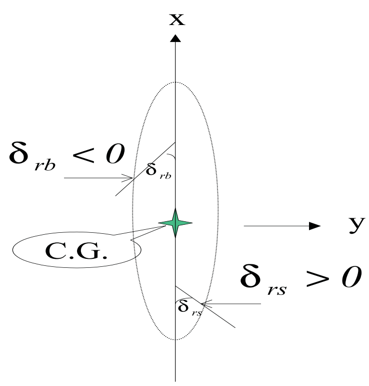

Figure 2.5: Deflections of Bow and Stern Rudders in the Steering Subsystem

$$\begin{bmatrix} m - Y_{\dot{v}} & mx_G - Y_{\dot{r}} & 0 \\ mx_G - N_{\dot{v}} & I_z - N_{\dot{r}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{v} \\ \dot{r} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} Y_v & Y_r - mu_0 & 0 \\ N_v & N_r - mx_G u_0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} v \\ r \\ \psi \end{bmatrix} + \begin{bmatrix} Y_\delta \\ N_\delta \\ 0 \end{bmatrix} \delta_r \quad (2.17)$$

Re-writing (2.17) based on the above assumptions gives:

$$\begin{bmatrix} m - Y_{\dot{v}} & mx_G - Y_{\dot{r}} & 0 \\ mx_G - N_{\dot{v}} & I_z - N_{\dot{r}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{v} \\ \dot{r} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} Y_v u_0 & (Y_r - m)u_0 & 0 \\ N_v u_0 & (N_r - mx_G)u_0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} v \\ r \\ \psi \end{bmatrix} +$$

$$0.5u_0^2 Y_\delta \begin{bmatrix} 1 & 1 \\ 0.283L & -0.377L \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_{rb} \\ \delta_{rs} \end{bmatrix} \quad (2.18)$$

NPS's work determines the values of constants and coefficients in (2.18) as applied to their Phoenix AUVs. Rearranging it into a state-space form, we have:

$$
\begin{bmatrix} \dot{v} \\ \dot{r} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} -0.6383 & -1.4439 & 0 \\ 0.0591 & -0.4659 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} v \\ r \\ \psi \end{bmatrix} + \begin{bmatrix} 0.1708 & -0.2650 \\ 0.1924 & 0.302 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_{rb} \\ \delta_{rs} \end{bmatrix} \tag{2.19}
$$

$$
Y = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ r \\ \psi \end{bmatrix} \tag{2.20}
$$

2.2.2 Diving Subsystem of NPS UUVs

The diving subsystem involves mainly the heave velocity w, the angular velocity in pitch q, the pitch angle θ, the depth z and the stern plane deflection δ_s. It neglects the primary horizontal motions of the sway velocity v, the yaw rate of turn r, the heading angle ψ, the rolling mode (p, ϕ), and the horizontal plane positions X and Y. Assuming that the vehicle is already in forward motion with a constant forward speed u_0 relative to the water column, and a zero pitch, we may drop products of small motions and horizontal-plane motions coupled to the vertical-plane equations. The linearized equations of motion in heave and pitch from [39] are as follows.

$$
\begin{bmatrix} m - Z_{\dot{w}} & mx_G - Z_{\dot{q}} & 0 & 0 \\ mx_G - M_{\dot{w}} & I_y - M_{\dot{q}} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{w} \\ \dot{q} \\ \dot{\theta} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} Z_w & Z_q - mu & 0 & 0 \\ M_w & M_q - mx_G u_0 & -\overline{BG_z}W & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & -u_0 & 0 \end{bmatrix} \begin{bmatrix} w \\ q \\ \theta \\ z \end{bmatrix} + \begin{bmatrix} Z_\delta \\ M_\delta \\ 0 \\ 0 \end{bmatrix} \delta_s \tag{2.21}
$$

Equation (2.21) uses the hydrodynamic added mass, the linear damping, and the effects of stern plane deflection to describe the external forces and moments. In addition, it models the moment from $\overline{BG_z}$, the vertical distance between the center of gravity z_G and the center of buoyancy z_B.

Since the heave velocity w is very small in diving, less than $0.05 m/s$ [42], Equation (2.21) reduces to:

$$\begin{bmatrix} \dot{q} \\ \dot{\theta} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} \frac{M_q}{I_y - M_{\dot{q}}} & -\frac{\overline{BG_z}W}{I_y - M_{\dot{q}}} & 0 \\ 1 & 0 & 0 \\ 0 & -u_0 & 0 \end{bmatrix} \begin{bmatrix} q \\ \theta \\ z \end{bmatrix} + \begin{bmatrix} \frac{M_\delta}{I_y - M_{\dot{q}}} \\ 0 \\ 0 \end{bmatrix} \delta_s \tag{2.22}$$

With numerical values from NPS Phoenix AUVs for the constants and coefficients in (2.22), we have:

$$\begin{bmatrix} \dot{q} \\ \dot{\theta} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} -0.7589 & -0.1086 & 0 \\ 1 & 0 & 0 \\ 0 & -3 & 0 \end{bmatrix} \begin{bmatrix} q \\ \theta \\ z \end{bmatrix} + \begin{bmatrix} -0.0697 & -0.0929 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_{bp} \\ \delta_{sp} \end{bmatrix} \tag{2.23}$$

where δ_{bp} and δ_{sp} represent the bow plane and the stern plane deflection in the diving subsystem, respectively. We can measure the depth z by a pressure meter, the pitch angle θ by an inclinometer, and the pitch rate q by a rate gyro or a rate sensor.

2.3 Decoupled Sliding-Mode Autopilot Design

Sliding-mode control (SMC) has been proposed for advanced UUV control and is very promising because of its robustness, stability and disturbance rejection properties. Early in 1980's, Yoerger and Soltine [45,46] proposed a series of single-input-single-output (SISO) continuous time controllers and used sliding surfaces in the control of underwater vehicles. They discuss the application of adaptive sliding-mode control to underwater vehicles later in [47]. Cristi, Papoulias and Healey [48] design an adaptive sliding-mode controller for the UUV in dive plane. There are also many other successful implementations of SMC on different underwater vehicles. For examples, Yoerger, Newman and Slotine [49] apply SMC to the supervisory control system of JASON ROV; Dougherty and co-workers [50,51] design a flight control system using sliding-mode control for Autonomous Underwater Vehicles (AUVs), and test the sliding-mode control autopilot on the MUST lab; and Healey and Lienard [41] propose a multivariable sliding-mode autopilot and apply it to the NPS AUV II for the combined control of speed, steering, and diving.

2.3.1 Sliding-Mode Control Concepts

The typical structure of a sliding-mode controller is composed of a nominal part and additional terms to deal with model uncertainty. The way SMC deals with uncertainty is to drive the plant's state error trajectory onto a sliding surface, and maintain the error trajectory on this surface for all subsequent times. The significant advantage of SMC is that the controlled system becomes insensitive to system disturbances, such as sinusoidal signals and random noises, and variations of state variables while on the sliding surface.

Let $\tilde{x} = x - x_d$ be the tracking error of a state variable x. We can define a time varying surface $s(t)$ in the state space by a scalar equation $\sigma[\tilde{x}(t)] = 0$. The sliding surface is defined such that the state tracking error converges to zero.

Yoerger and co-authors [45,46] define a sliding surface for single-input and single-output affine systems by:

$$s = \dot{\tilde{x}} + \lambda \tilde{x} = 0 \tag{2.24}$$

where $\lambda > 0$ is the control bandwidth. On the sliding surface, we have:

$$\dot{\tilde{x}} + \lambda \tilde{x} = 0 \qquad \Longrightarrow \qquad \tilde{x}(t) = \exp[-\lambda(t - t_0)\tilde{x}(t_0)] \tag{2.25}$$

The exponential dynamics in (2.25) ensures that the tracking error $\tilde{x}$ will converge to zero in finite time for any initial $\tilde{x}(t_0)$. This definition of sliding surface is good for single-input single-output systems.

For coupled-mode multiple-state system, Cristi, et al. [48] use an alternative definition of the sliding surface (2.26), which is based on state variables rather than output errors:

$$\sigma(\widetilde{X}) = S^T \widetilde{X} = 0 \tag{2.26}$$

where $\widetilde{X} = X - X_d$ is a vector of state tracking errors from desired state variables, and S is a vector of known coefficients to be designed based on the linear model of the system. The coefficients in the vector S completely determine the sliding surface. This definition of sliding surface ensures the convergence of the state tracking error to zero. The following discussion and SMC design are based on the latter definition.

Healey and co-authors have shown that the dynamic model of an underwater vehicle can be linearized around nominal flight-operating conditions, and we can separate the dynamics of underwater vehicles into linear approximate model and non-linear disturbances, as expressed in Eq.(2.27).

$$\dot{X} = AX + Bu + f(x) \tag{2.27}$$

where $f(x)$ represents a non-linear function describing the deviation from linearity in terms of disturbances and un-modeled dynamics. Their experiments show that this model can be used to describe a large number of underwater flight conditions. A well-established SMC design [48] is based on this linear model in Eq.(2.27).

The control objective now turns to find a control law to drive $\sigma(\widetilde{X})$ towards zero based on model (2.27). By defining a Lyapunov function:

$$V(\sigma) = 1/2[\sigma]^2 \tag{2.28}$$

we can guarantee that the sliding surface $\sigma(\widetilde{X}) = 0$ is reached in finite time by choosing equation (2.29) to ensure that $\dot{V} = \sigma\dot{\sigma} \leq 0$.

$$\sigma\dot{\sigma} = -\eta^2|\sigma| \qquad \text{or} \qquad \dot{\sigma} = -\eta^2 sgn(\sigma) \tag{2.29}$$

where η^2 is a tunable parameter. From Eq. (2.26), we have:

$$\begin{aligned} \dot{\sigma}(\widetilde{X}) &= S^T\dot{\widetilde{X}} \\ &= S^T(\dot{X} - \dot{X}_d) \\ &= S^T(AX + Bu + f(x)) - S^T\dot{X}_d \end{aligned} \tag{2.30}$$

Let (2.29) equal to (2.30), we have:

$$S^T(AX + Bu + f(x)) - S^T\dot{X}_d = -\eta^2 sgn(\sigma) \tag{2.31}$$

Therefore, we can justify the function of (2.29) by choosing a control input as:

$$\begin{aligned} u &= \hat{u} + \bar{u} \\ &= (S^TB)^{-1}(-S^TAX) + (S^TB)^{-1}(S^T\dot{X}_d - S^Tf - \eta^2 sgn(\sigma)) \end{aligned} \tag{2.32}$$

The control law u in (2.32) consists of two parts. The first part $\hat{u}$ is a linear feedback control law based on the nominal model in Eq.(2.27) with a gain vector $K^T = (S^TB)^{-1}S^TA$ computed from pole placement. The second part $\bar{u}$ is a nonlinear feedback control with the term $-\eta^2 sgn(\sigma)$ switching its sign between plus and minus, depending on which side of the sliding plane the state tracking error is located. The latter part is for driving and keeping the system onto the sliding surface $\sigma(\widetilde{X}) = 0$. If η is chosen large enough, $\bar{u}$ can provide required robustness to the effects of momentary disturbances and un-modeled dynamics. However, the discontinuous function sgn in the control law can lead to control chattering. As Slotine and Li [52] point out, the chattering involves high control activity, and furthermore may excite high-frequency dynamics neglected in the modeling. Therefore, it is desirable to smooth the control activity near the sliding surface. In practice, we can modify the sliding-mode control law by replacing the sgn function with a continuous function of $tanh$ [39]:

$$\begin{aligned} u &= \hat{u} + \bar{u} \\ &= -K^T X + (S^T B)^{-1}[S^T \dot{X}_d - S^T \hat{f}(x) - \eta^2 tanh(\sigma/\phi)] \end{aligned} \tag{2.33}$$

where $\hat{f}(x)$ is an estimate of $f(x)$, and ϕ is the thickness of sliding-surface boundary layer.

From $K^T = (S^T B)^{-1} S^T A$, we have:

$$S^T(A - BK^T) = 0 \qquad A - BK^T = A_c \tag{2.34}$$

Therefore, S^T is a left annihilator of A_c, or S is a right eigenvector of A_c^T corresponding to a zero eigenvalue. With this choice of S, we can completely determine the sliding surface $S^T \widetilde{X} = 0$ and the sliding-mode control law in (2.33).

In summary, the design procedure of the sliding-mode controller consists of two steps: first, determine the feedback control law $\hat{u}$ on the basis of the nominal model; and secondly, design a control law $\bar{u}$ to compensate for deviations from ideal performances due to uncertainties.

2.3.2 Autopilot Design for Steering and Diving Subsystems

Considering the linear steering equations in (2.19) and (2.20), which is controllable and observable, we define the sliding surface as:

$$\sigma = s_1 \tilde{v} + s_2 \tilde{r} + s_3 \tilde{\psi} = 0 \tag{2.35}$$

Here $\tilde{v}$, $\tilde{r}$ and $\tilde{\psi}$ are three components of $\widetilde{X}$. From a practical point of view, we let the desired sway velocity v_d in steering be zero. The values of ψ_d and r_d in course keeping are different from those in path tracking. For course keeping, ψ_d is a constant, and therefore r_d=0. For path tracking, ψ_d and r_d usually follow certain dynamics.

Due to the pure integration in the yaw channel of (2.19), there is an eigenvalue of zero for the linear steering subsystem. By placing steering poles at $(-0.5, -0.34, 0)$, we find a linear feedback gain $K = [0.0673, -0.5215, 0]^T$. Therefore,

$$A_c = A - BK^T = \begin{bmatrix} -0.632 & -1.493 & 0 \\ 0.0258 & -0.208 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{2.36}$$

We can determine the sliding surface coefficients as:

$$S = \begin{bmatrix} 0.0394 & 0.9649 & 0.2596 \end{bmatrix}^T \tag{2.37}$$

For course keeping, we have a sliding surface of:

$$\sigma_1 = 0.0394v + 0.9649r + 0.2596(\psi - \psi_d) \tag{2.38}$$

and a steering control law of:

$$u = -0.0673v + 0.5215r + \frac{1}{S^T B}[-\eta^2 tanh(\sigma_1/\phi)] \tag{2.39}$$

The linear feedback term in (2.39) does not include the ψ because of a zero feedback gain corresponding to the ψ. Only v and r act to stabilize the sway-yaw dynamics.

If it is designed for path tracking, then the sliding surface is:

$$\sigma_2 = 0.0394v + 0.9649(r - r_d) + 0.2596(\psi - \psi_d) \tag{2.40}$$

and the steering control law becomes:

$$u = -0.0673v + 0.5215r + \frac{1}{S^T B}[0.9649\dot{r}_d + 0.2596r_d - \eta^2 tanh(\sigma_2/\phi)] \tag{2.41}$$

For the diving subsystem, we define the sliding surface as:

$$\sigma = s_1\tilde{q} + s_2\tilde{\theta} + s_3\tilde{z} \tag{2.42}$$

where $\tilde{q}$, $\tilde{\theta}$ and $\tilde{z}$ are errors from their desired values. Usually, the value of z_d is a constant, therefore $\dot{z}_d = 0$.

By placing poles at (0 -0.25 -0.26), we find a linear feedback gain $K = [1.5303, 0.2681, 0]^T$ for the diving subsystem. Therefore, we have:

$$A_c = A - BK^T = \begin{bmatrix} -0.51 & -0.065 & 0 \\ 1 & 0 & 0 \\ 0 & -3 & 0 \end{bmatrix} \tag{2.43}$$

and the sliding surface coefficients are:

$$S = \begin{bmatrix} -0.8907 & -0.4542 & 0.0193 \end{bmatrix}^T \tag{2.44}$$

The control law for the diving subsystem becomes:

$$u = -1.5303q - 0.2681\theta + \frac{1}{S^TB}[-0.8907\dot{q}_d - 0.4542q_d - \eta^2 tanh(\sigma/\phi)] \tag{2.45}$$

In actual implementation, we often have to use estimated states instead of direct measurement due to restricted set of signals obtained or noisy corrupted sensor data. Cristi[48] investigates sliding-mode control with estimated states, and shows that the control system is still stable provided that perturbations in the dynamics are small enough in their L_∞ norm. Also, we can achieve greater degrees of robustness by using robust observers. We consider using Kalman filter for state estimation from sensor measurements. The discrete-time dynamic model of the steering subsystem at a sample rate of 10Hz is:

$$\begin{cases} X_{k+1} = \Phi X_k + \Gamma u_k + W_k \\ Y_k = HX_k + V_k \end{cases} \tag{2.46}$$

where

$$\Phi = \begin{bmatrix} 0.9378 & -0.1366 & 0 \\ 0.0056 & 0.9541 & 0 \\ 0.0003 & 0.0977 & 1 \end{bmatrix} \qquad \Gamma = \begin{bmatrix} -0.0126 \\ 0.0483 \\ 0.0024 \end{bmatrix} \tag{2.47}$$

W_k is a zero-mean white-noise disturbance, causing unpredictable uncertainty in the system behavior, and V_k is zero mean white noise corrupting the measurement signals. Furthermore, W_k and V_k are not correlated. Based on the model in (2.46), the Kalman filter generates a "best" estimation of X_k, with a least squared error between X_k and its estimation $\widehat{X_k}$.

We use Matlab code for simulating the control based on linearized subsystem models, and producing all the figures. Taking depth control as an example, we simulate SMC for two occasions: one is disturbance-free; another is corrupted with random noises $0.01rand(1)$ in each state channel. We set the desired pitch angle θ_d and pitch rate q_d as zero, and a depth command of 10 meters. Figures 2.6 and 2.7 show the control result without disturbances and with disturbances, respectively. In the case of clear environment, we see accurate control of each state. In the case of disturbances, the control system manages to keep the depth at $10m$, only with a small steady state error.

For the steering subsystem, we simulate sliding-mode control for both course keeping and path tracking. For comparison, we also design a PD controller for the steering subsystem to compare the performance of SMC with that of PD control in the same situation.

The PD control law simply is:

$$\begin{aligned} u &= K_1\hat{v} + K_2(\hat{r} - r_d) + K_3(\hat{\psi} - \psi_d) \\ &= -0.1196\hat{v} - 1.43(\hat{r} - r_d) - 0.6949(\hat{\psi} - \psi_d) \end{aligned} \tag{2.48}$$

Jalving [42] uses a similar form of PD control law for NDRE-AUV, and has demonstrated satisfactory performance.

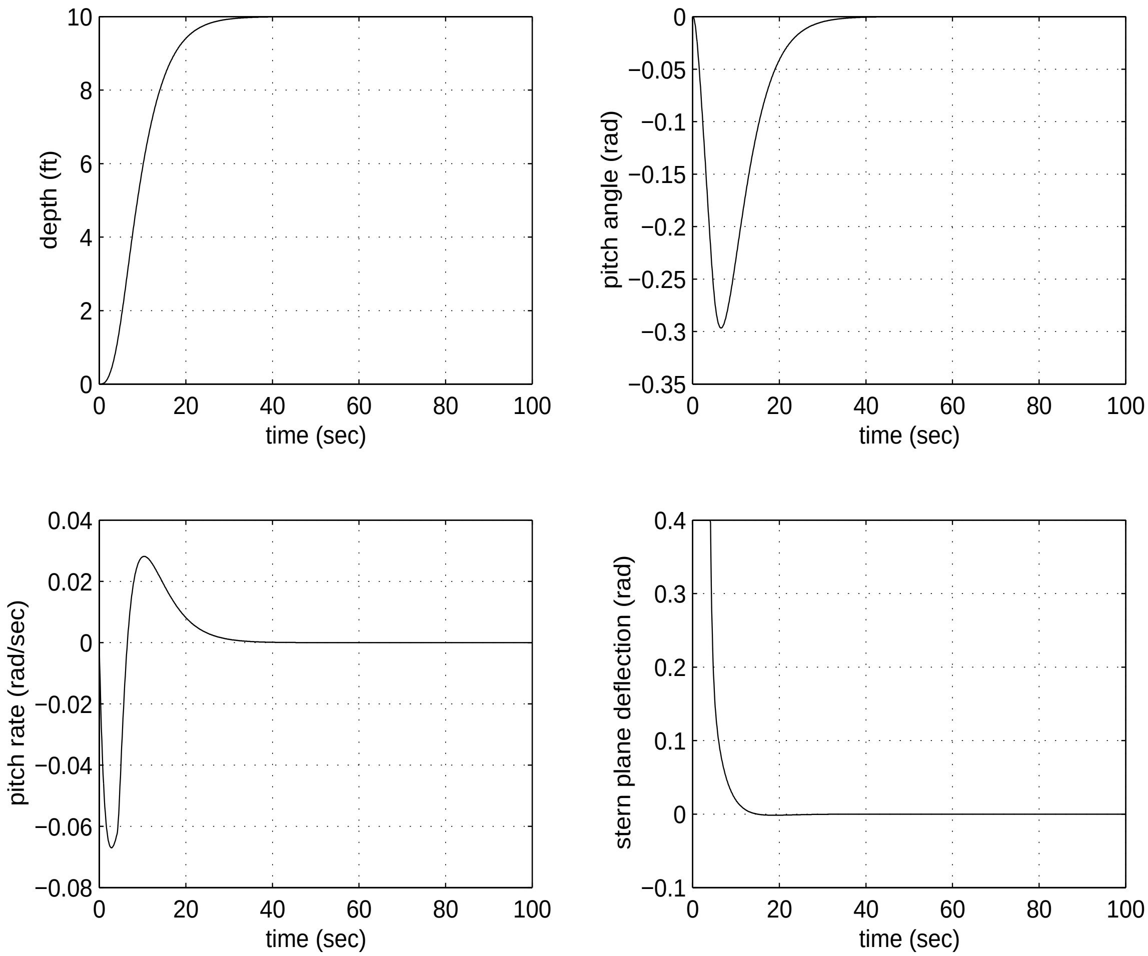

Figure 2.6: Depth Control without Disturbances

In course keeping, we try to keep a constant heading angle. For the example here, we set $\psi_d = 90^0$. Figure 2.8 presents the results of an example course keeping. Both SMC and PD control give satisfactory performance in course keeping. Only SMC is a little slower than PD control in controlling the heading angle to reach 90-degree. However, SMC does not use as much rudder control as PD, and sway velocity and yaw rate with SMC are relatively smaller than with PD in making the turn.

In simulating path tracking, we make the steering path follow a reference model in Eq. (2.49). Figure 2.9 shows the plane trajectory of the vehicle when both ψ and r follow the

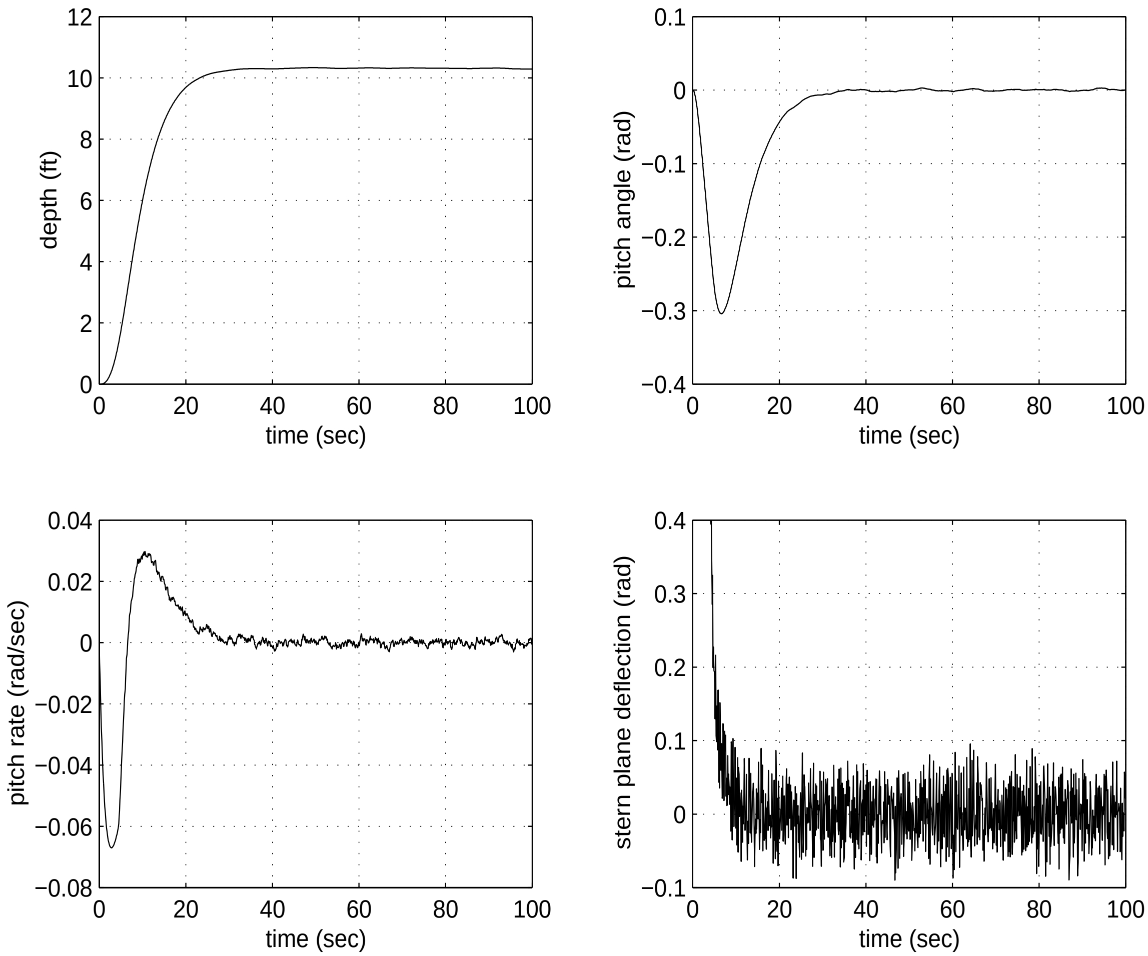

Figure 2.7: Depth Control with Disturbances

dynamic model.

$$\begin{bmatrix} \dot{\psi}_d \\ \dot{r}_d \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -0.01 & -0.16 \end{bmatrix} \begin{bmatrix} \psi_d \\ r_d \end{bmatrix} + \begin{bmatrix} 0 \\ 0.1 \end{bmatrix} 20 \tag{2.49}$$

Figure 2.10 simulates the path tracking using SMC and PD control separately. It demonstrates that SMC is superior to PD in tracking the desired yaw rate and heading angle.

Considering a sinusoidal disturbance $sin(t)$ in the channel of states v and r, we simulate

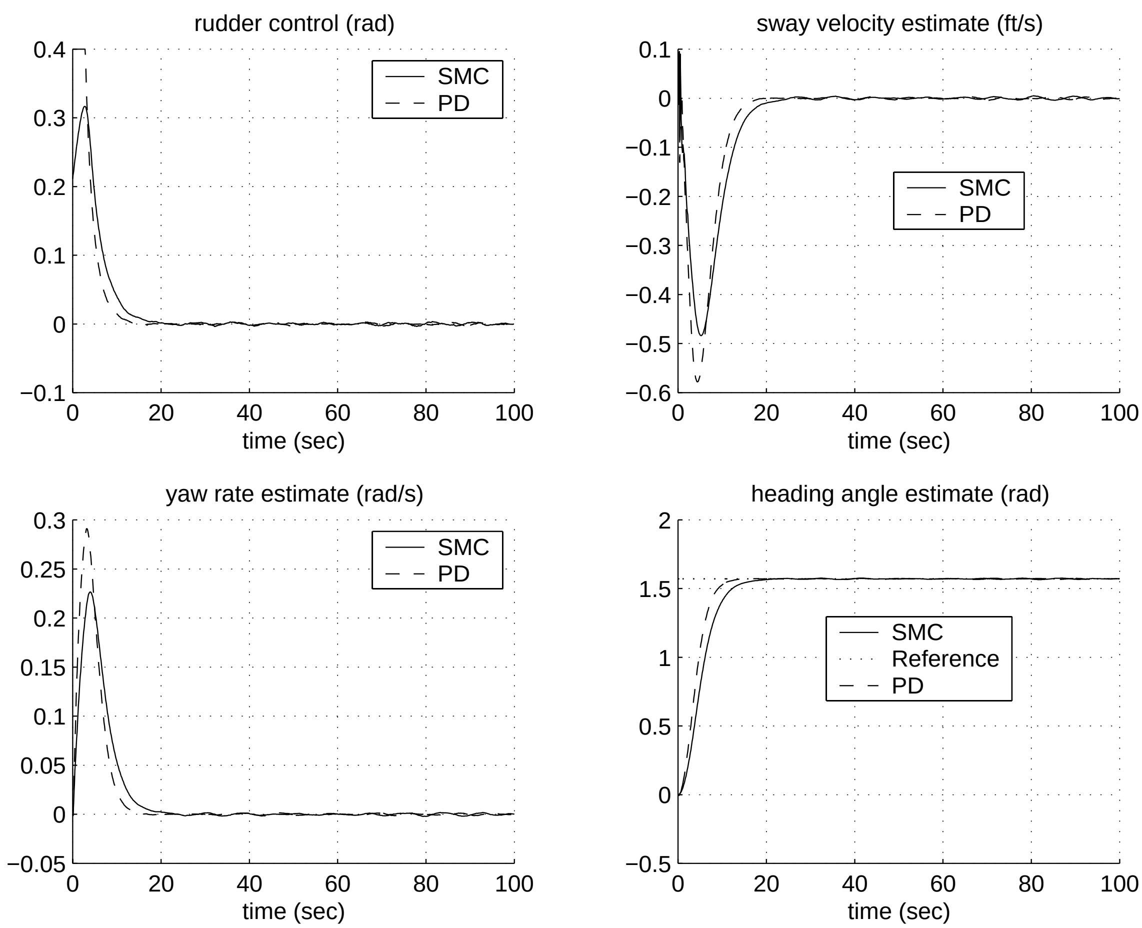

Figure 2.8: SMC and PD control for Course Keeping

the sliding-mode control and the PD control again for the same course keeping and path tracking. Figure 2.11 and Figure 2.12 show the results.

Figures 2.11 and 2.12 show that the sinusoidal disturbance causes the rudder control to oscillate, and makes sway velocity, yaw rate, and heading angle oscillate stably around the desired value in the course keeping. However, the tracking performance under sinusoidal disturbance is still satisfactory.

Figure 2.13 compares the steering trajectories from different control, together with the reference one. The trajectory resulting from SMC is closer to the reference trajectory than

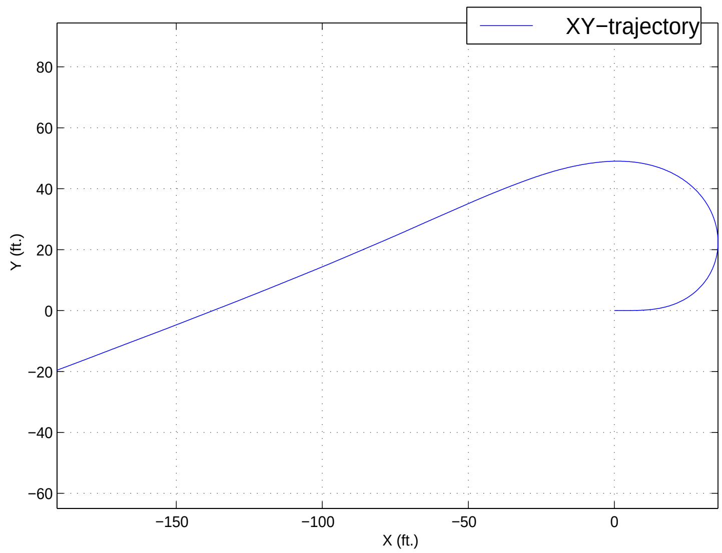

Figure 2.9: Trajectory of the Steering Path

that of the PD control.

2.4 Summary

By grouping together related control and state variables, we decompose the dynamic model of UUVs into three separate non-interacting subsystems for decoupled control of forward speed, steering and diving. We derive autopilot designs for the steering and the diving subsystems separately based on the sliding-mode control. We also compare the performances of the SMC and PD control with simulated maneuvering and disturbances, and demonstrate that SMC is superior to PD control in this application. By operating the vehicle properly, we can observe abnormal information by comparing with what we have been expecting. In the following chapters, we will discuss in details the design and implementation of fault detection, identification and compensation for UUVs.

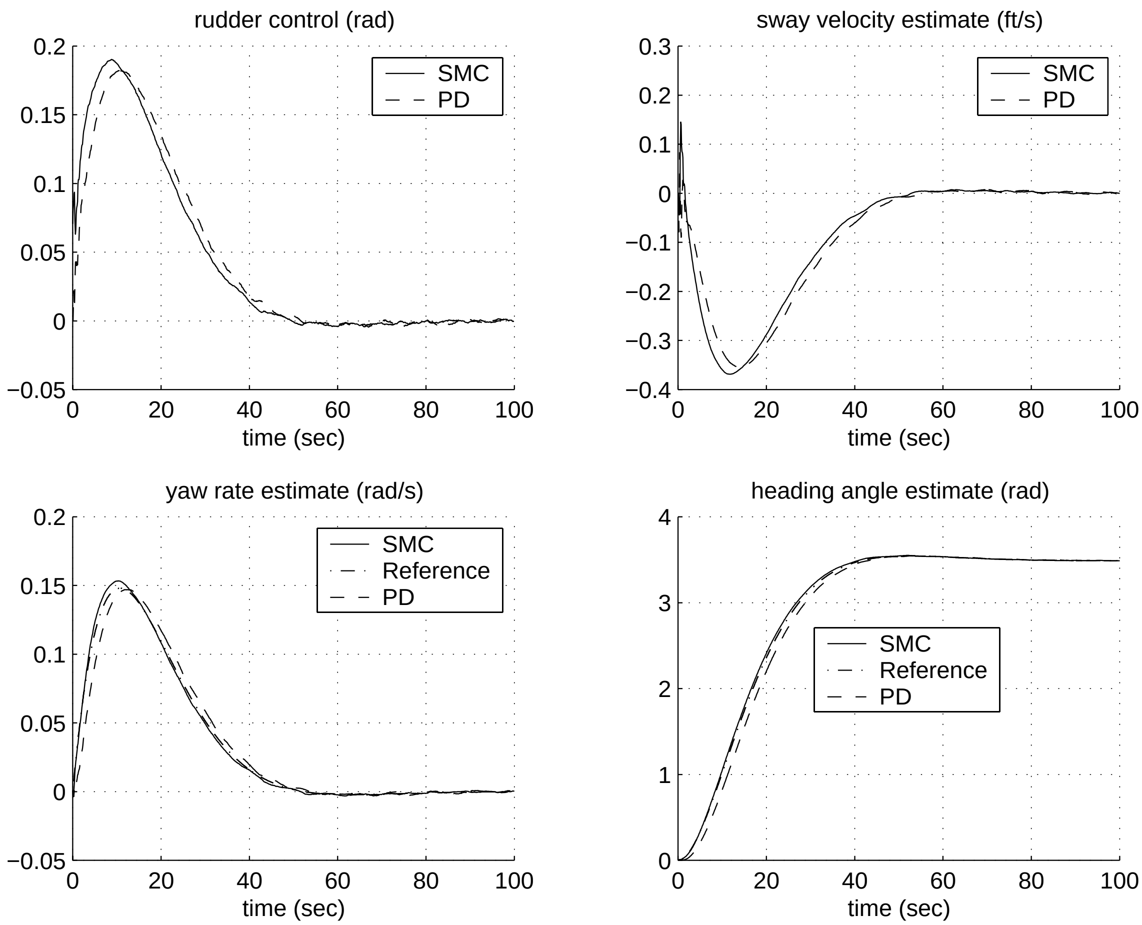

Figure 2.10: SMC and PD Control for Path Tracking

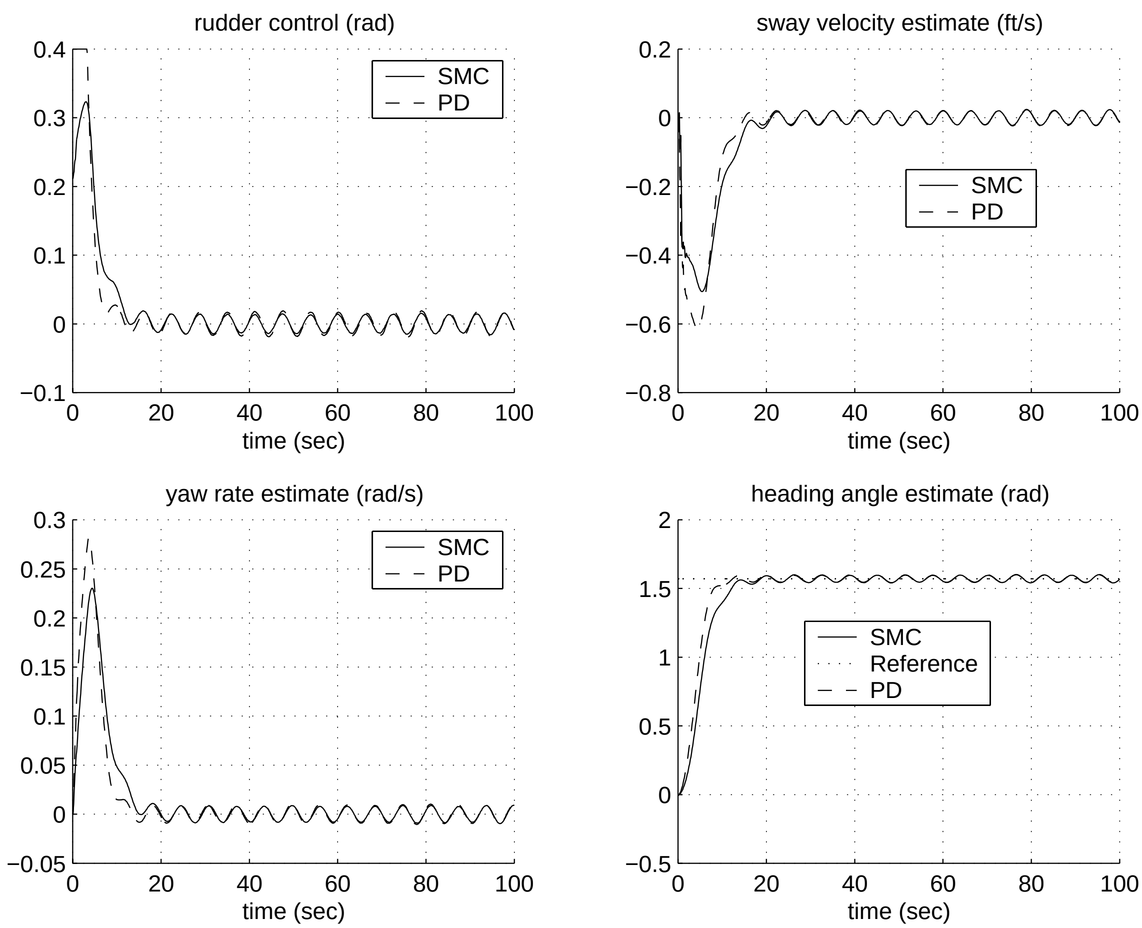

Figure 2.11: Course Keeping with Sinusoidal Disturbance

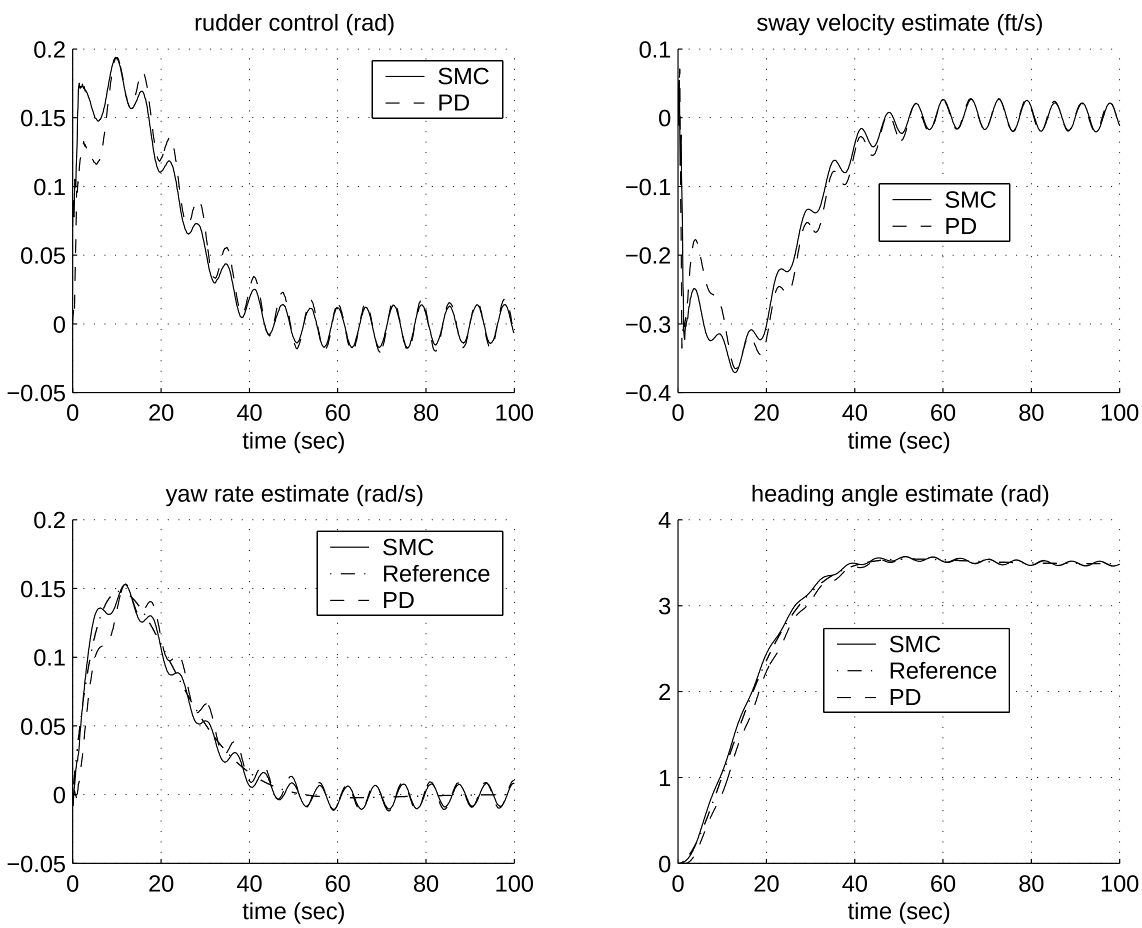

Figure 2.12: Path Tracking with Sinusoidal Disturbance

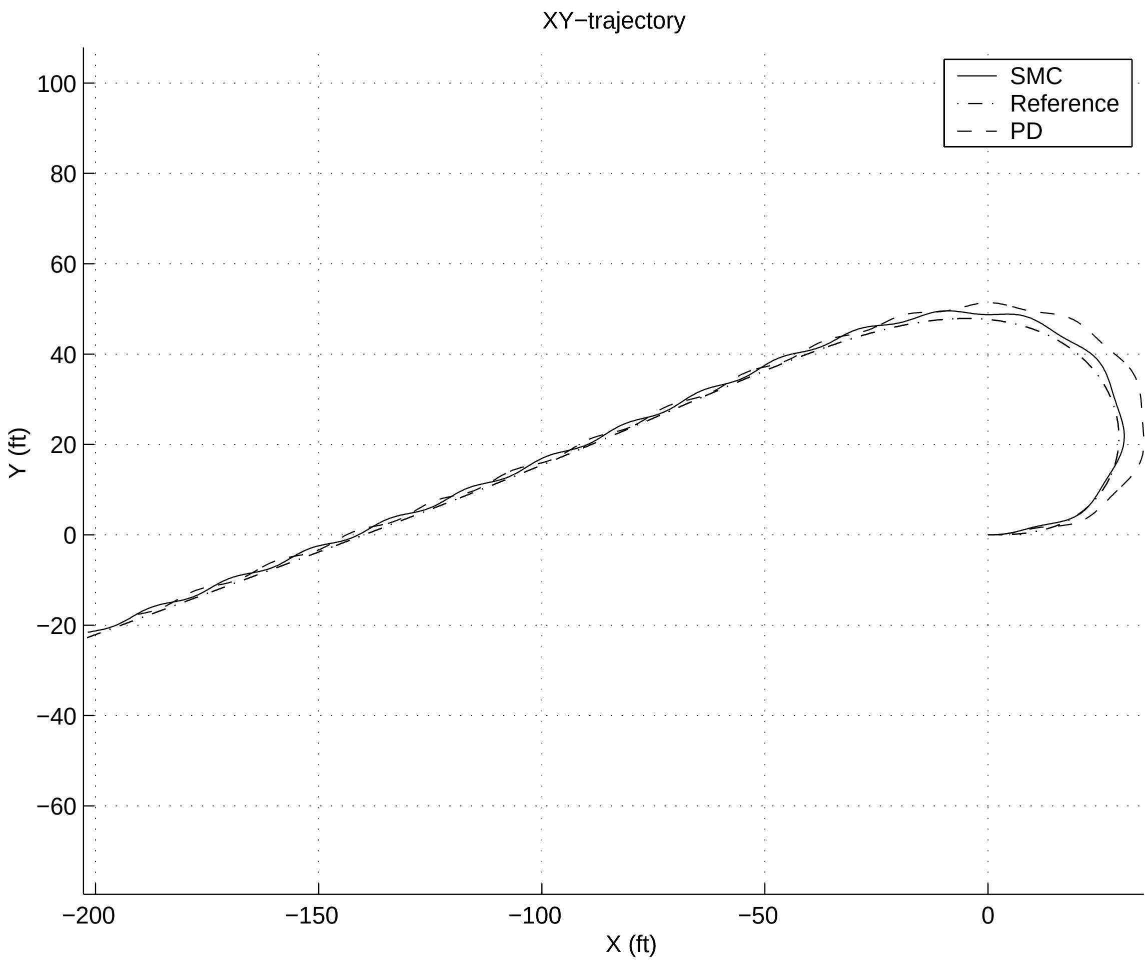

Figure 2.13: Comparison of Path Tracking with Different Control Schemes

Chapter 3

Fault Detection Based on Adaptive LMS Modeling and Analysis

Chapter 2 introduced modeling and control of UUV subsystems to establish a controlled system for the purpose of fault detection, identification, and compensation. This chapter develops fault detection techniques from a practical point of view. Implementation of fault detection techniques in real time involves system state estimation, information processing, and decision logic.

The chapter first briefly discusses actuator and sensor failures and fault modeling. The remaining sections focus on the design and implementation of the fault detection technique. We use a finite-impulse-response (FIR) filter for on-line adaptive modeling and parameter estimation. We apply a least-mean-squares (LMS) algorithm to minimize the output error between the monitored system and the filter in the modeling process. Then, we analyze the resulting adaptive filter coefficients to extract information about fault occurrence and design a decision function based on this information. Finally, we implement the technique to the simulated steering and diving subsystems of NPS UUV to detect actuator and sensor failures.

3.1 Modeling of Actuator and Sensor Failures

As mentioned before, non-linear models for UUVs are usually decoupled into main subsystems for autopilot design. We restrict our fault-detection problem to the same subsystems, and are concerned about actuator and sensor failures only for this work. Figure 3.1 is the schematic drawing of the NPS UUV [41].

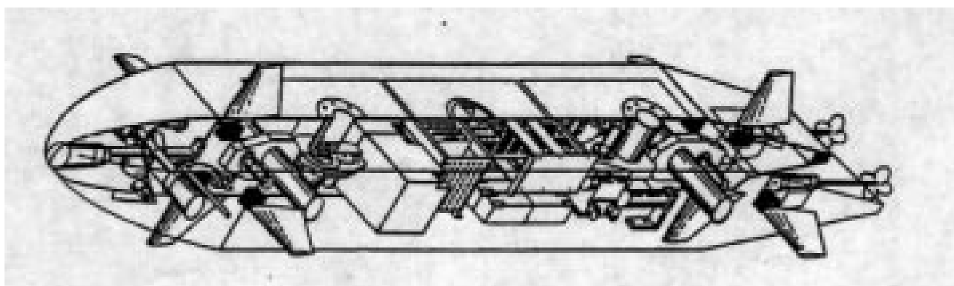

Figure 3.1: Schematic Drawing of NPS UUV

According to the description in Section 2.2, the steering subsystem is responsible for control of heading errors. The input to the steering system is the control command for deflection of the bow rudders and stern rudders, while the output from the steering system is the yaw rate of turn. There are four rudders and two sensors for the steering subsystem of NPS UUV. The four rudders are the upper and lower bow rudders, and the upper and lower stern rudders (see Figure 3.1). The two sensors are the yaw rate sensor and the heading angle sensor. The diving subsystem controls the depth and pitch errors. It has two planes at both bow and stern (see Figure 3.1), and three sensors for the pitch angle, the pitch rate, and the depth.

Table 3.1 lists main possible rudder and sensor failure modes relating to the steering subsystem. Similar failures happen to planes and sensors of the diving subsystem. By examining these failures, we can classify them into two categories: total and partial failures [29,30]. Total failures refer to those failures caused by a complete functional loss of the actuator or the sensor, such as no rudder response, or loss of sensor input and output, while partial failures represent the other type of failures such as the stroke limited rudder and the sensor bias.

Table 3.1: Failure Modes of Actuators and Sensors

rudder failure	no rudder response, stroke limited rudder loose rudder, stuck rudder, hard rudder
sensor failure	loss of input, loss of output sensor bias, increased noise level

Partial actuator or sensor failures are not necessary less harmful than total failures. We separate failures into total and partial categories because they are different in modeling, and this classification facilitates a more detailed description of failures in fault-identification results.

In Chapter 2, we have represented dynamic subsystems with linearized state space models. To formulate the problem of fault detection and identification, we need to describe the fault model. We model actuator failures as additive parameter changes in the control input matrix, and sensor failures as additive parameter changes in the output equation of the system.

$$\begin{cases} \dot{X} = AX + (B + B_F)u + Ef_d \\ Y = (C + C_F)X + v \end{cases} \tag{3.1}$$

Equation (3.1) includes the nominal model, the disturbance, and the failure. An equivalent discrete form of Eq. (3.1) is as follows.

$$\begin{cases} X_{k+1} = \Phi X_k + \Gamma(\theta_k)u_k + W_k \\ Y_k = H(\theta_k)X_k + V_k(\theta_k) \end{cases} \tag{3.2}$$

where Φ, Γ, and H are the state transition matrix, the control input matrix, and the observation matrix in their discretized form, respectively; θ_k denotes the uncertain parameters that affect the matrices defining the structure of the model or depicting the statistics of the noises entering it at time k. Furthermore, we assume that θ_k takes on discrete values and a first-order Markov chain governs the mode transition, for example, from a fully functional mode θ_1 to any other failure modes θ_i. W_k is the system disturbances or modeling error. V_k is the measurement noise. We assume both W_k and V_k are independent Gaussian random sequences with the following mean values and covariances:

$$\begin{aligned} E[W_k] &= 0; \qquad E[V_k] = 0 \\ E[W_k W_k^T] &= Q_k; \qquad E[V_k V_k^T] = R_k \end{aligned}$$

We also assume W_k and V_k to be mutually independent, or at least not correlated, so that:

$$E[W_k V_k] = 0 \tag{3.3}$$

With these assumptions, we can use Kalman filters for estimating states [56].

There are different ways to describe total and partial actuator and sensor failures. Let the ith component of the output vector Y be the measurement from the ith sensor, and let the jth column of the control input matrix Γ correspond to the jth actuator. For total sensor failures, such as sensor disconnection, the elements of the ith row of the observation matrix H are zero. For partial sensor failures, the value of the ith component of the output vector is nonzero, and the covariance of the ith component of the output noise vector increases by the amount representing the noise increase. For total failure of the jth actuator, the jth column of the control input matrix Γ is zero. For partial actuator failures, we multiply the corresponding columns by an effectiveness factor, instead of zero. Table 3.2 summarizes the fault modeling, where p means an arbitrary number of percent. We simulate actuator and sensor failures according to this fault model.

Table 3.2: Modeling of Actuator and Sensor Failures

	ith sensor	jth actuator
Partial failures	$H(i,:) = H(i,:) \times p\%$	$\Gamma(:,j) = \Gamma(:,j) \times p\%$
Total failures	$H(i,:) = H(i,:) \times 0$	$\Gamma(:,j) = \Gamma(:,j) \times 0$

3.2 Adaptive Modeling and Analysis

The basic idea of our fault detection method is to monitor the health status of the dynamic system through LMS adaptive system identification, and to detect fault occurrence through analysis of the history of adaptive model coefficients.

3.2.1 Adaptive Modeling

Adaptive plant modeling or identification is important for adaptive control, since the dynamics of the plant to be controlled may be poorly known. A discrete-time adaptive modeling system automatically adjusts its internal parameters to produce a sampled output closely matching the samples of the plant output, when the adaptive modeling use the same input signal as the plant. In this study, we use adaptive modeling for the purpose of fault detection. We build an adaptive filter to model the unknown faulty system, and to identify any actuator and sensor failures.

Figure 3.2 illustrates the adaptive system-identification process with a least-mean-squares (LMS) algorithm. It uses a finite-impulse-response (FIR) filter to adaptively model the plant, which is the steering subsystem here, since a linear approximation is applicable to the UUV subsystems [39]. The FIR filter can build a linear approximation to the system even if it is non-linear. The same input is applied to the adaptive filter as to the monitored system. The output of the unknown system provides the desired response for the adaptive filter.

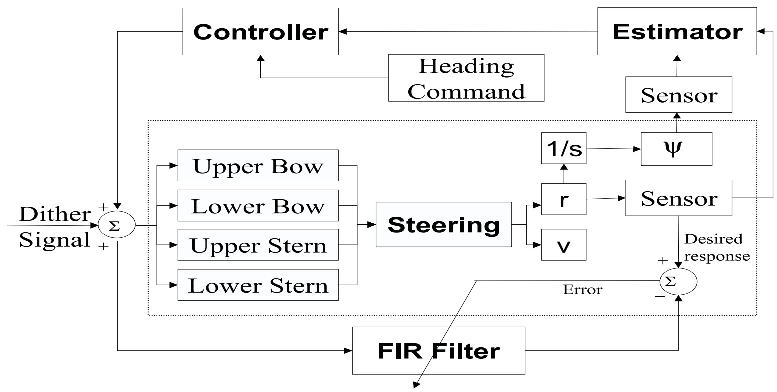

Figure 3.2: Adaptive System Identification with LMS Algorithm

Figure 3.3 illustrates the adaptive FIR filter. It consists of a tapped delay line, adjustable weights, and a summer [53]. The adaptation process uses an error signal ε_k, which is the difference between the output of the adaptive filter and the desired response of the UUV dynamic subsystem.

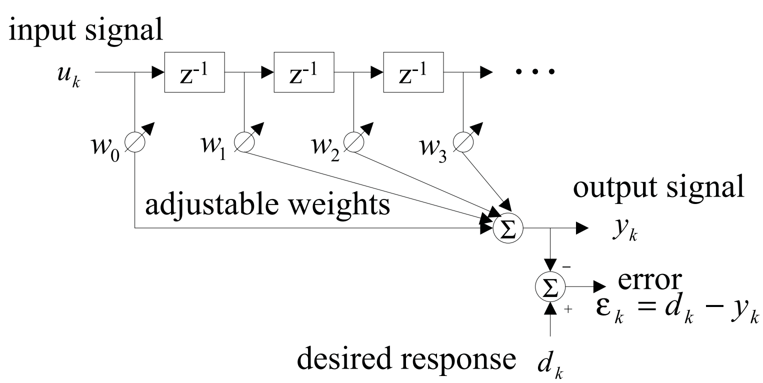

Figure 3.3: Adaptive FIR Filter

Equation 3.4 represents the output from the adaptive model:

$$y_k = W^T U_k \tag{3.4}$$

where $W^T = [w_0, w_1, w_2, ..., w_n]$ is a set of adjustable filter weights, and $U_k = [u_k, u_{k-1}, u_{k-2}, ..., u_{k-n}]^T$ is a vector of control inputs.

Furthermore, we can express the input and the output relationship in Eq. (3.4) as:

$$y_k = G(z)u_k = \sum_{i=0}^{n} w(i)u(k-i) \tag{3.5}$$

where:

$$G(z) = \sum_{i=0}^{n} w(i)z^{-i} \tag{3.6}$$

When the weights converge and the error signal becomes small, the output of the adaptive filter tends to match the desired response. In this case, the impulse response of the model becomes a good approximation to the impulse response of the monitored system. The coefficients of the FIR model then represent a discrete approximation of the impulse response of the system and characterize its dynamics. When sensor or actuator failures occur, the resulting changes in the system dynamics are reflected in the changes of impulse response as expressed in the FIR coefficients (weights). We use the LMS algorithm [53] to drive the adaptation process to update FIR coefficients at each sample of input signal.

The LMS Algorithm

The LMS algorithm differs from other adaptive algorithms in that the implementation of the algorithm is relatively simple, computationally efficient, and can be used in real time. To better understand the LMS algorithm, it is necessary to introduce the concept of mean-square-error (MSE) performance surface and the method of steepest descent [54].

From Figure 3.3, we see that the error ε_k between the desired response and the output signal from adaptive filter at the kth sampling time is:

$$\varepsilon(k) = d_k - y_k = d_k - W^T U_k = d_k - U_k^T W \tag{3.7}$$

The mean-square error ξ is defined as the expected value of the squared error $\varepsilon^2(k)$:

$$\begin{aligned} \xi(k) &= E[\varepsilon^2(k)] = E[d_k^2 - 2d_k U_k^T W + W^T U_k U_k^T W] \\ &= E[d_k^2] - 2E[d_k U_k^T]W + W^T E[U_k U_k^T]W \end{aligned} \tag{3.8}$$

where U_k is the input vector at time k, $E[d_k U_k] = P$ is the desired-to-input cross-correlation vector, and $E[U_k U_k^T] = R$ is the input autocorrelation matrix.

The MSE performance function in Eq. (3.8) is a quadratic function of weights (filter coefficients) because the weights appear only to the first and second degrees. In the case of two weights, MSE values form a bow-shaped surface. The adaptive process adjusts the weights to seek the bottom of the bowl, or the minimum mean-square error.

The method of steepest descent is used to minimize the MSE. It is an iterative technique that searches for the minimum by following the direction in which the performance surface has the greatest rate of decrease, specifically, following the negative gradient of the performance surface. With successive corrections of the weight vector in the direction of the steepest descent on the performance surface, the MSE will arrive at its minimum value, where the weight vector components take on their optimum values.

To implement the steepest descent method, we make the change of the weight vector proportional to the negative of the gradient vector. In other words, we iteratively progress in the direction of the gradient.

$$W_{k+1} = W_k + \mu(-\nabla\xi(k)) \tag{3.9}$$

where,

$$\nabla\xi(k) = W_k + \mu(-2P + 2RW_k) \tag{3.10}$$

μ is a step-size parameter that controls stability and the rate of descent, and W_k and W_{k+1} are old and new weights, respectively. With Eq. (3.9), we can find the optimum solution of the weight vector. However, P and R are unknown in practical applications. Therefore, the exact gradient of the performance surface is unknown.

The LMS algorithm is an implementation of the steepest descent method using measured or estimated gradients. Widrow [53] uses an instantaneous squared error ε_k^2 to estimate the MSE $\xi(k)$, thus the gradient estimate used by the LMS algorithm is

$$\begin{aligned}\nabla\hat{\xi}(k) &= 2[\nabla\varepsilon_k]\varepsilon_k \\ &= -2U_k\varepsilon_k\end{aligned} \tag{3.11}$$

We express the LMS algorithm as follows [53]:

$$W_{k+1} = W_k + 2\mu\varepsilon_k U_k \tag{3.12}$$

The principal factors that affect the response and performance of the LMS algorithm are the step-size parameter μ, the number of tapped delays, and the eigenvalues of the correlation matrix R of the input vectors [53].

The LMS algorithm only involves multiplications and additions, and does not require squaring, averaging, or differentiating. When the number of weights is determined, the number of multiplications and additions at each iteration is also fixed.

Modeling Mismatch

The adaptive FIR model is a discrete model of the impulse response of the plant. When the plant and its model produce similar output signals, the adaptive impulse response is a good representation of the plant impulse response. However, the differences between the adaptive model and the plant is always existing. This is so-called modeling mismatch. Before we apply the adaptive modeling, we need to consider the problem of mismatch.

There are three sources of mismatch [53]. The first comes from the use of FIR models to represent the plant, since the plant usually has an infinite impulse response (IIR). Theoretically, we should make sure that the FIR model match that part of plant impulse response within the duration of finite response. To obtain good modeling, the finite response should be long enough to cover the significant part of the plant's impulse response. A practical way is to examine the impulse response of the system and observe the statistics of the input signal in order to choose the number of adaptive weights.

It is good to have a large number of weights for FIR model to reduce the mismatch. However, misadjustments are also proportional to the number. Weight noise is another source of mismatch. The weight vector becomes noisy when the adaptive process is too fast. Therefore, how to trade off between the adaptation speed and the number of weights is another problem. We achieve the trade-off through trial and error in the simulation.

The third source of mismatch is due to the inadequate excitation over the interested range of frequencies as expressed by a transform of the impulse response. This also results in the mismatch between the adaptive model and the plant. When the vehicle is not maneuvering, the plant input usually fails to have adequate spectral density within the frequencies of interest. One way to circumvent this problem is to use a "persistently exciting" random dither signal as an input to the plant and the adaptive filter in the modeling process.

In Figure 3.2, we add white noise into the control signal to provide the plant input as well as the modeling signals. Widrow and Walach [53] present three schemes of using dither signal in the modeling process. Among these schemes, scheme A is in the same way as we described in Figure 3.2. It is very straightforward, and effective as long as the independent random dither signal is added. In contrast to scheme A, schemes B and C (see Figures 3.4 and 3.5) use only the dither signal as the modeling input. The minimum mean square error with scheme B is greater than with scheme A. The higher value of minimum mean square error will cause increased noise in the adaptive model's weights. Scheme C is an improvement on scheme B. It reduces the weight noise while adapting with the same rate of convergence and with the same dither amplitude as with scheme B. However, scheme C make the system more sophisticated.

We use white noise as the dither signal in the adaptive modeling process as described in Figure 3.2. Figure 3.6 shows the adaptive model of the normally operated steering subsystem.

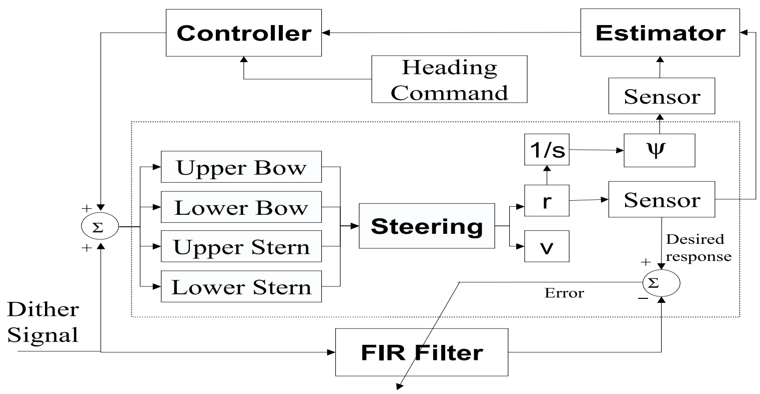

Figure 3.4: Scheme B for Using Dither Signal

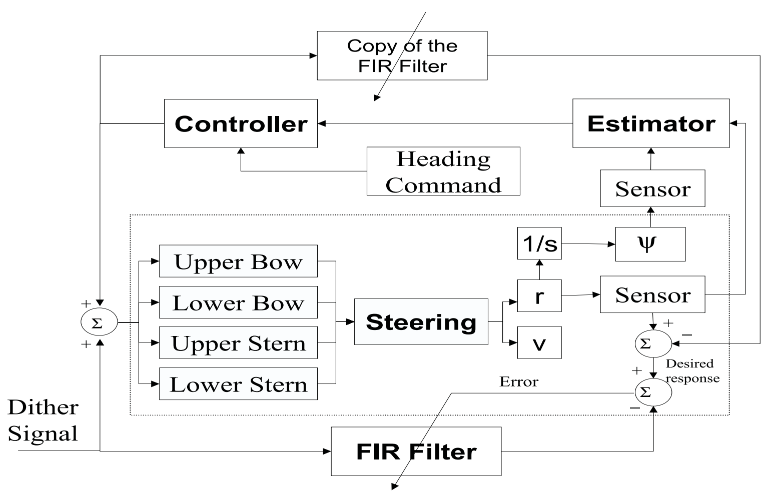

Figure 3.5: Scheme C for Using Dither Signal

We obtain the model by building a FIR filter adaptively using the LMS algorithm until the FIR coefficients converge. The solid curve is the impulse response of the monitored system without any disturbance or sensor noise, while the circle curve is the impulse response obtained from the adaptive filter in the presence of system noises and measurement noises. We see that the impulse response of the model almost matches that of the monitored system, except for some noise.

Although the dither signal is necessary in many cases to achieve the desired spectral

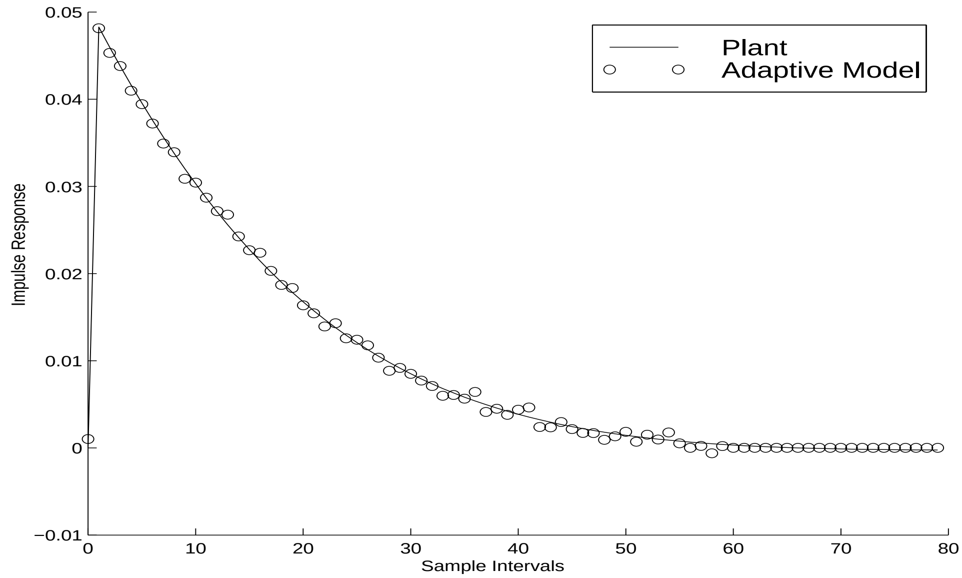

Figure 3.6: Impulse Response of the Adaptive Model Using Dither Signal

characteristics of the modeling signal, it introduces extra disturbances to the control process and causes noisy output signals.

Another way to reduce the mismatch due to low excitation inputs is to use a purposeful command that makes the vehicle perform a specific maneuver, for example, continual step input signals. Figure 3.7 is the result of adaptive modeling using the purposeful command of a series of 90-degree turns for the heading angle of steering. We see that the impulse response of the model closely matches that of the monitored system.

We apply both methods in our simulations to improve the result of adaptive modeling. The purpose of adaptive modeling is to track the on-line system performance so that we can detect the failure occurrence timely. Beginning with the next section, we analyze the time-varying filter coefficients for fault detection.

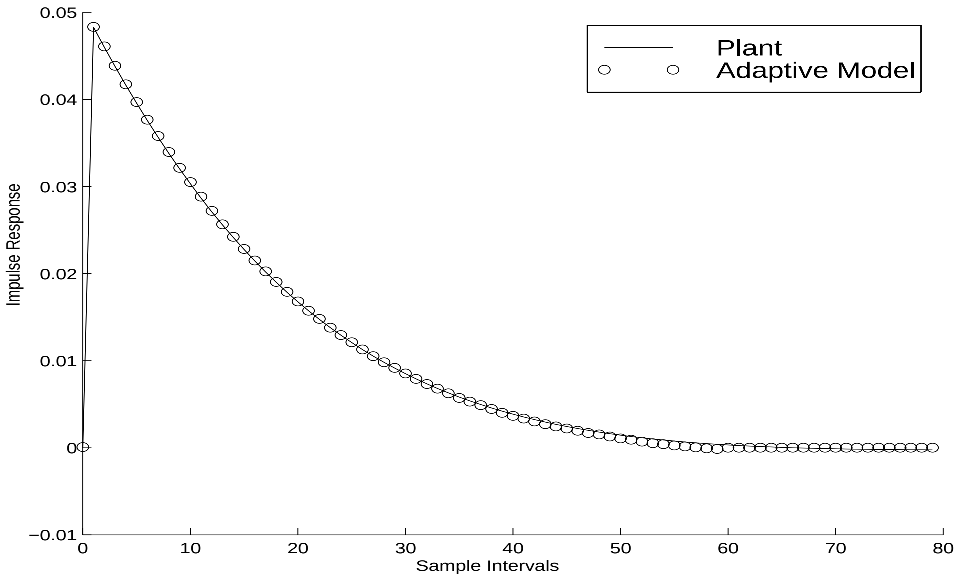

Figure 3.7: Impulse Response of the Adaptive Model and Plant

3.2.2 Adaptive Analysis

The purpose of adaptive analysis is to extract useful information from the time-history of coefficients, so that we can detect the failure occurrence by checking the significant rate of change.

Assuming that we have already obtained a matched model of a normally operated UUV subsystem (see Figure 3.7), we represent it with a vector of FIR coefficients $\vec{C}_0$. We can simply make $\vec{C}_0$ equal to the finite impulse response of the plant, or we can obtain $\vec{C}_0$ through adaptive modeling. We begin the adaptation process using this matched model as a starting point.

Figure 3.8 illustrates the adaptive model history. Specifically, we represent the adaptive model at every sampling period by a vector of FIR coefficients $\vec{C}_j, (j = 1, 2, ..., n)$, where n is the current sampling point, and N is the number of coefficients of each adaptive FIR filter. Each $\vec{C}_j$ approximates the impulse response of the plant at that moment. It carries the information of plant dynamics and thus reflects any change in the dynamics.

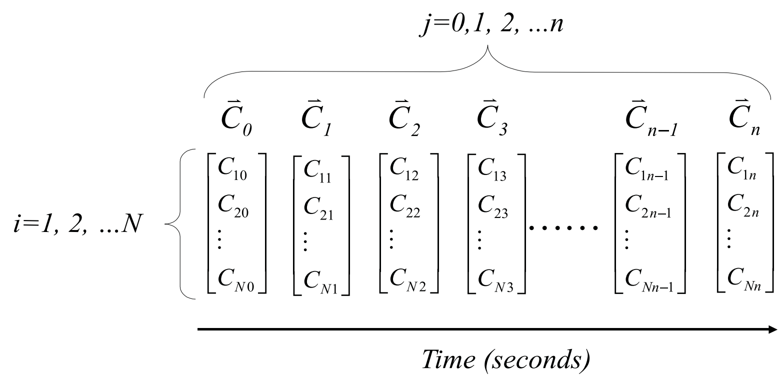

Figure 3.8: Adaptive Model History as Represented by Vectors of FIR Coefficients

Suppose that we are dealing with the information in an off-line case, in which we post-process all the data of the mission duration, first, we subtract $\vec{C_0}$ from all the other $\vec{C_j}, (j = 1, 2, ..., n)$, resulting in difference vectors $\vec{D_j}$(see Figure 3.9), where $\vec{D_j} = \vec{C_j} - \vec{C_0}, (j = 1, 2, ..., n)$. $\vec{D_j}, (j = 1, 2, ..., n)$ represent the deviations of each adaptive model from the normal system. Since all the $\vec{C_j}, (j = 1, 2, ..., n)$ are adapted from $\vec{C_0}$, elements of $\vec{D_j}$ will approach zero if there is no fault occurrence. Otherwise, some elements of $\vec{D_j}$ may have large magnitudes. The points where $\vec{D_j}$ have high rate of change indicate the change in the dynamics of the plant.

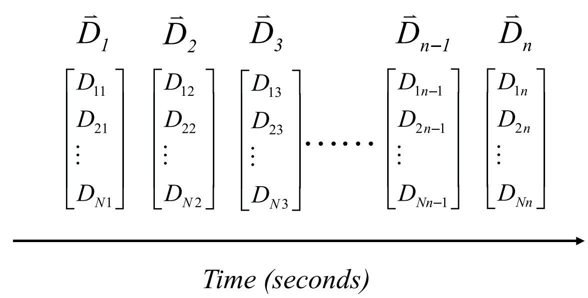

Figure 3.9: Difference Vectors Between the Adaptive Model and the Normal System

Next, we sum over the elements D_{ij} of each column vector $\vec{D_j}$, resulting in scalars $d_j, (j = 1, 2, ..., n)$. In other words, $d_j, (j = 1, 2, ..., n)$ sum up the deviations in $\vec{D_j}, (j =$

$1, 2, ..., n$). We define a vector $\vec{D}$ consisting of all the d_j as follows.

$$d_j = \sum_{i=1}^{N} D_{ij} \qquad \vec{D} = [d_1, d_2, ...d_n] \tag{3.13}$$

The vector $\vec{D}$ represents the deviations of each adaptive model from the normal system. Finally, we normalize the $\vec{D}$ in Eq. (3.14) by subtracting the mean value of all elements $d_j, (j = 1, 2, ..., n)$ from d_j themselves, and dividing them by their standard deviation (*std*). We take their absolute values in Eq. (3.14).

$$\vec{V} = |\frac{\vec{D} - mean(\vec{D})}{std(\vec{D})}| \tag{3.14}$$

Now, we have a normalized difference vector $\vec{V}$. The scalar elements of the vector $\vec{V}$ are of a similar order of magnitude because of the scaling. To emphasize abnormal changes among adaptive model deviations and reduce influences from noisy signals, we calculate the signal-to-noise ratio (SNR) [24] for each element of $\vec{V}$ as follows.

$$S\vec{N}R = \frac{diag([\vec{V} - mean(\vec{V})][\vec{V} - mean(\vec{V})]^T)}{[std(\vec{V})]^2} \tag{3.15}$$

The $S\vec{N}R$ is a vector of signal-to-noise ratios corresponding to each element of $\vec{V}$. By observing the $S\vec{N}R$, we can tell if there is any significant jumps in the coefficient history. To implement fault detection, we will design a decision function based on the $S\vec{N}R$ to determine if the jumps are fault occurrences or not. The elements of $S\vec{N}R$, as well as those of $\vec{D}$ and $\vec{V}$, are corresponding to a sampling point j on the time axis, with their first element relates to $j = 1$.

We develop the preceding adaptive analysis in an off-line manner, in which all the FIR coefficients in the adaptive model history from $j = 1$ to n (see Figure 3.8) are available and can be processed simultaneously. However, in real-time adaptive analysis, we do not have all the data in advance. Therefore, we calculate the $S\vec{N}R$ at each sampling period based upon the existing data up to that point.

Assuming that at a sampling period $j = m$, we obtain a vector $S\vec{N}R_m = [r_1, r_2, ..., r_m]$, we have a series of vectors $S\vec{N}R_m, (m = 1, 2, ..., n)$ as moving on the time axis.

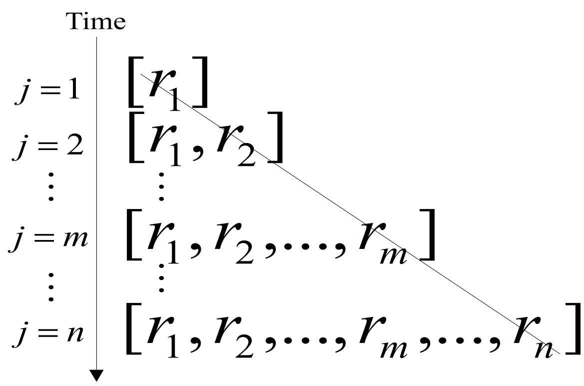

Figure 3.10: On-line $S\vec{N}R$

In each $S\vec{N}R_m$, only its last element r_m is concurrent with time. We define the on-line $S\vec{N}R$ as a vector consisting of all the elements $r_m, (m = 1, 2, ..., n)$ from different sampling periods.

The on-line $S\vec{N}R$ differs from the off-line $S\vec{N}R$ in two aspects. First, the on-line $S\vec{N}R$ is concurrent with time, while the off-line $S\vec{N}R$ is retroactive. Second, the mean values and standard deviations used in calculating the on-line $S\vec{N}R$ are in a moving form along the time axis due to the varying number of data available, while all the mean values and standard deviations in the off-line case are constants.

We compare the on-line $S\vec{N}R$ and the off-line $S\vec{N}R$ in Figure 3.11. We simulate a failure of no upper-stern-rudder response at $t = 35s$ and no response of both stern rudders at $t = 50s$. The on-line $S\vec{N}R$ becomes large quickly after $35s$, and reaches its highest value around $40s$. The off-line $S\vec{N}R$ increases gradually after $35s$, and becomes very large around $60s$. Apparently, the on-line $S\vec{N}R$ signals the failure earlier and has a higher confidence level than the off-line $S\vec{N}R$. Due to varying mean values and standard deviations, the on-line

$S\vec{N}R$ fluctuates markedly at the beginning. However, it does not trigger any alarm.

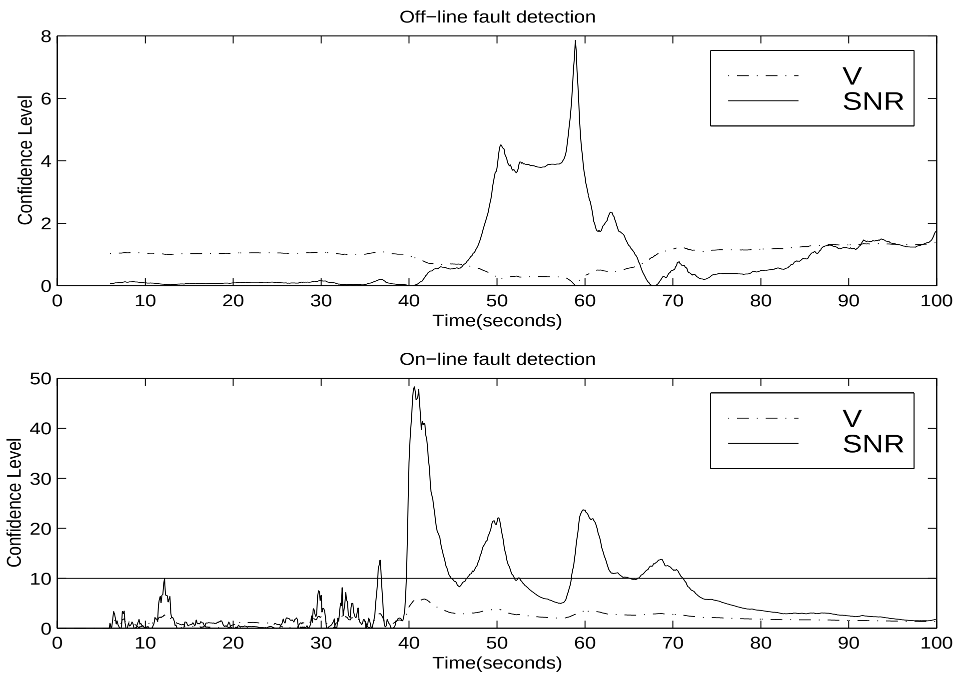

Figure 3.11: $\vec{V}$ and $S\vec{N}R$ Versus Sampling Time: On-line and Off-line

3.3 Algorithm Performance

In this section, we will demonstrate the performance of our fault-detection method by applying it to the steering and diving subsystems of the NPS UUVs with simulated actuator and/or sensor failures. The failures are those described earlier, namely, total or partial failures of actuators and sensors. In the cases given below, we also simulate dual failures as well as single failures, where dual failures are modeled as two failures occurring some time apart, with one following the other.

To perform adaptive modeling, we need to decide the sampling rate and the order of FIR adaptive filter first to ensure a good representation of the plant. We choose the sampling rate at $T = 0.1s$. We determine the number of coefficients by choosing a sufficiently large number of coefficients to completely describe the significant part of the plant's impulse response, that is the plant's impulse response up to the point where the latter decays to a "negligible" level. Figure 3.7 plots the impulse response for the linearized model of the NPS steering subsystem. The first 60 samples constitute the significant part and the impulse response decays to close to zero after that. Therefore, we choose the number of coefficients as $N = 60$.

For the diving subsystem, we set the number of FIR coefficients as $N = 300$, since the impulse response of the diving model in Figure 3.12 decays to zero after 300 samples. We fix the number of FIR coefficients in the adaptive modeling process as described in the method.

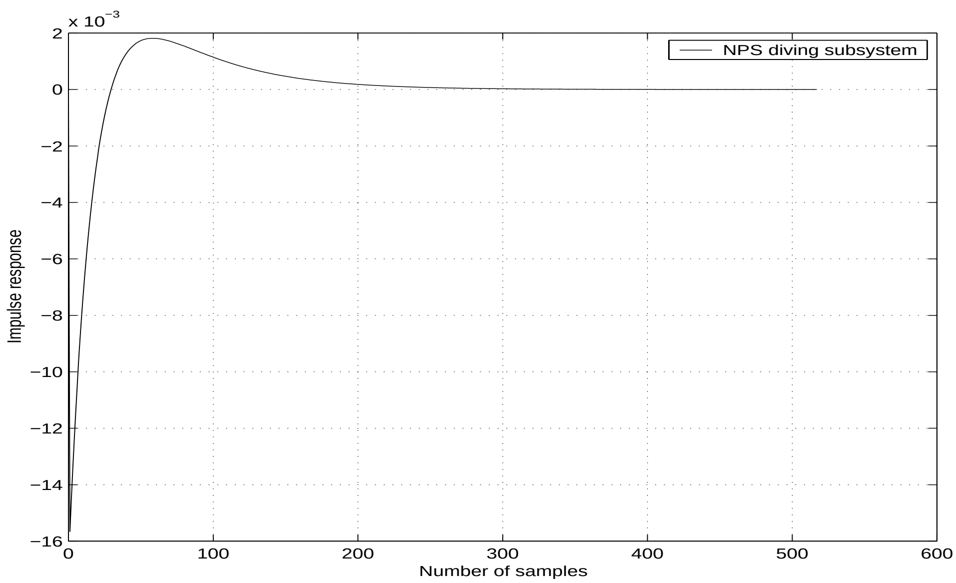

Figure 3.12: Impulse Response of the Adaptive Model of NPS Diving Subsystem

To obtain a good step-size parameter μ for the LMS algorithm, we use a trial and error method. In the simulation, we found that $\mu = 0.01$ provides good convergence and note that it is a commonly used value.

In implementing the on-line adaptive analysis, the number of coefficient vectors in the adaptive model history (see Figure 3.8) increases rapidly as time goes on, and the computational load increases with it. To prevent rapidly increasing computation loads, we set up a sliding window of a fixed length, for example, $L = 1000$.

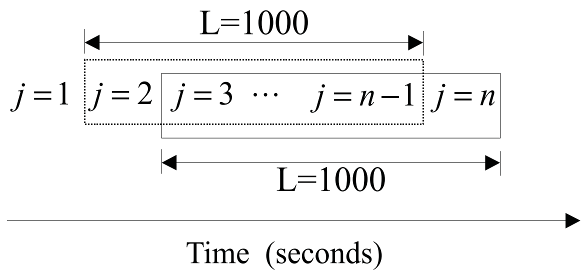

Figure 3.13: Fixed Length Sliding Window

At every sampling period, we only process the data within the window. As the window slides along the time axis, we gradually discard old data beyond the window. As a result, we limit the amount of data to its maximum of 1000.

In the following simulations, the UUV steering is designated to follow a square-track path (see Figure 3.14), and the diving subsystem follows a multiple-step path in Figure 3.15 for depth.

Figure 3.16 to 3.18 show fault-detection results in the steering subsystem. When a failure occurs to the subsystems, we see significant changes in the SNR and in the output errors. When the SNR is above a certain level, we begin to consider the possibility of fault occurrence, since the method matches the FIR filter model and the monitored system at the beginning of the simulation. Specifically, if the SNR surpass a certain confidence level continuously for 10 samples, then the method identifies the fault occurrence. The method chooses a threshold at a confidence level of 10, which is a normalized non-dimensional value. We determine this threshold by examining the chances of false alarms for various fault types.

In Figure 3.16, we simulate a single failure of no upper-bow-rudder response at $t = 50s$ for on-line fault detection. Figure 3.16 compares the yaw-rate state of the steering model and the output from the adaptive FIR filter, shows the square error between outputs from

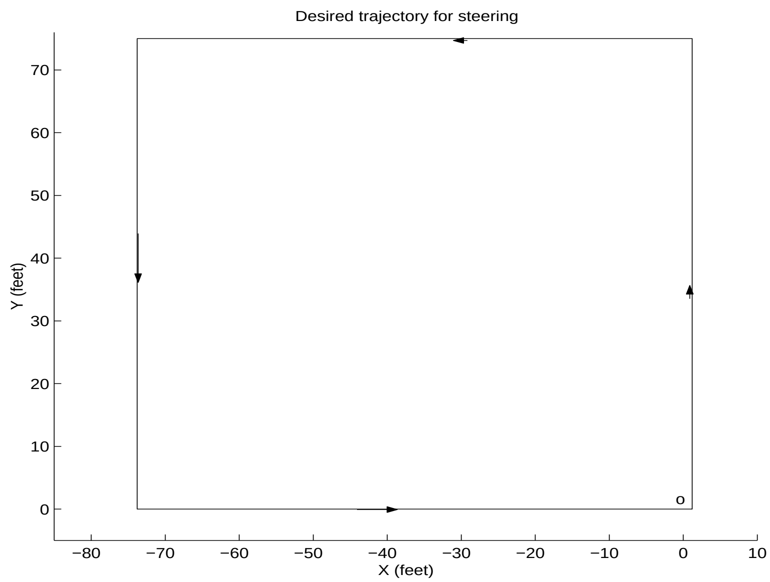

Figure 3.14: Steering Trajectory for UUVs

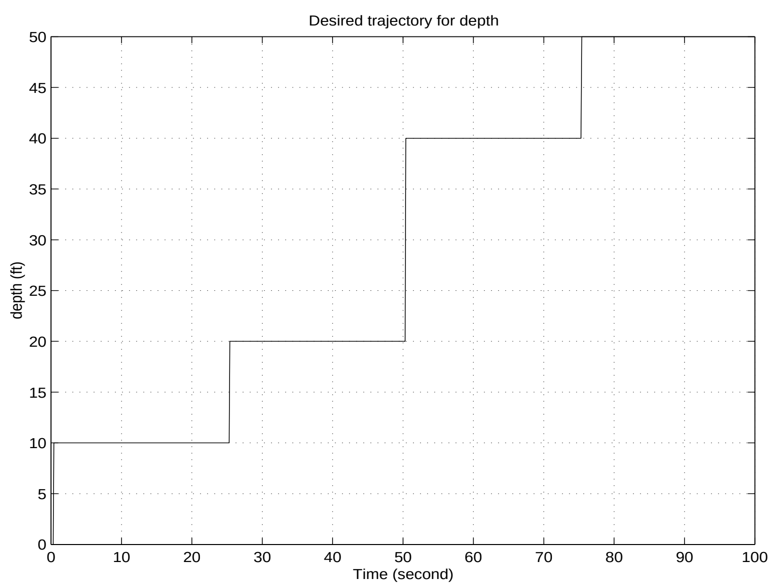

Figure 3.15: Desired Depth for Diving Subsystem

the plant and the FIR filter, and gives the result of on-line fault detection. Before the fault occurrence, the output from the adaptive FIR filter matches the yaw-rate state of the

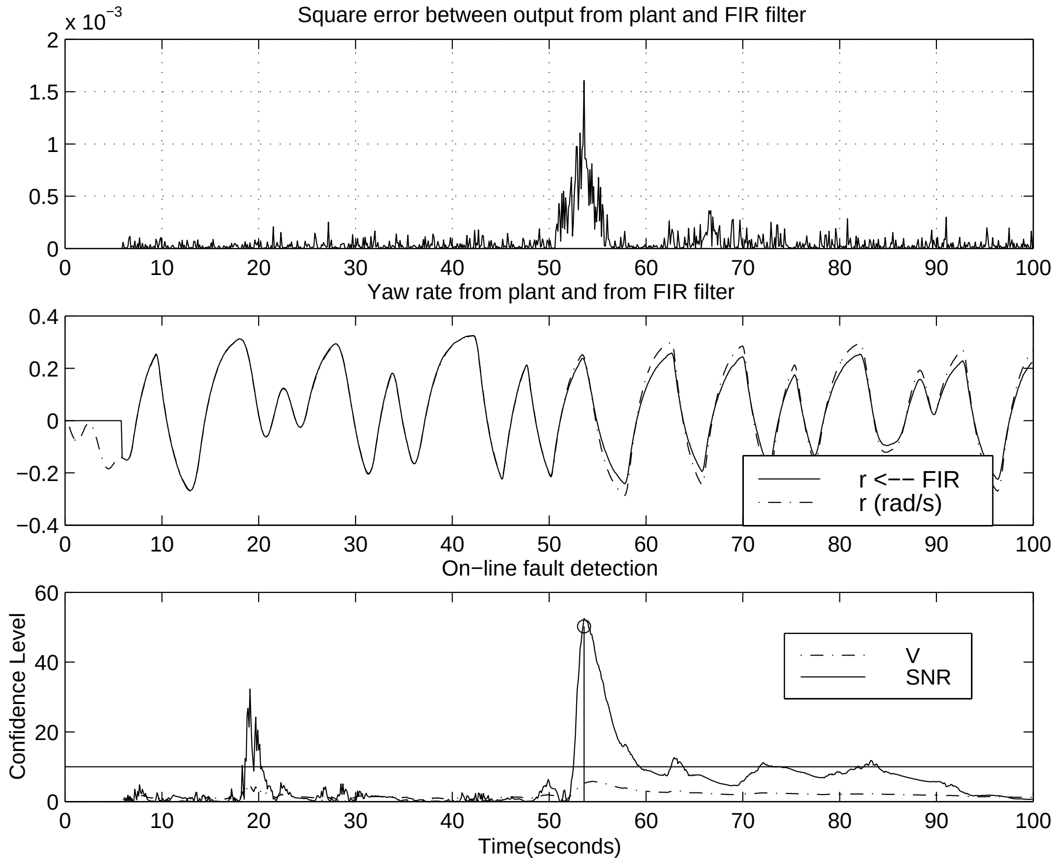

Figure 3.16: Single-Fault Detection in Steering Subsystem

steering model. After $t = 50s$, the two yaw rates split from each other. The rudder failure results in differences between them. There is a jump in the SNR shortly after the failure. There is also a large square error between the outputs after the rudder failure occurs. The curves have zero values at the beginning, since the FIR filter needs some sampling periods to initialize.

According to our threshold decision logic, the method detects the fault occurrence at $t = 53.6s$, 3.6 seconds after introducing the failure. The vehicle is moving with a longitudinal speed of $3ft/s$. It travels 10.8 feet, or 36 sampling periods before the failure is detected. However, considering that the vehicle is 6 feet long and moving with a slow speed, we regard this detection rate as reasonable.

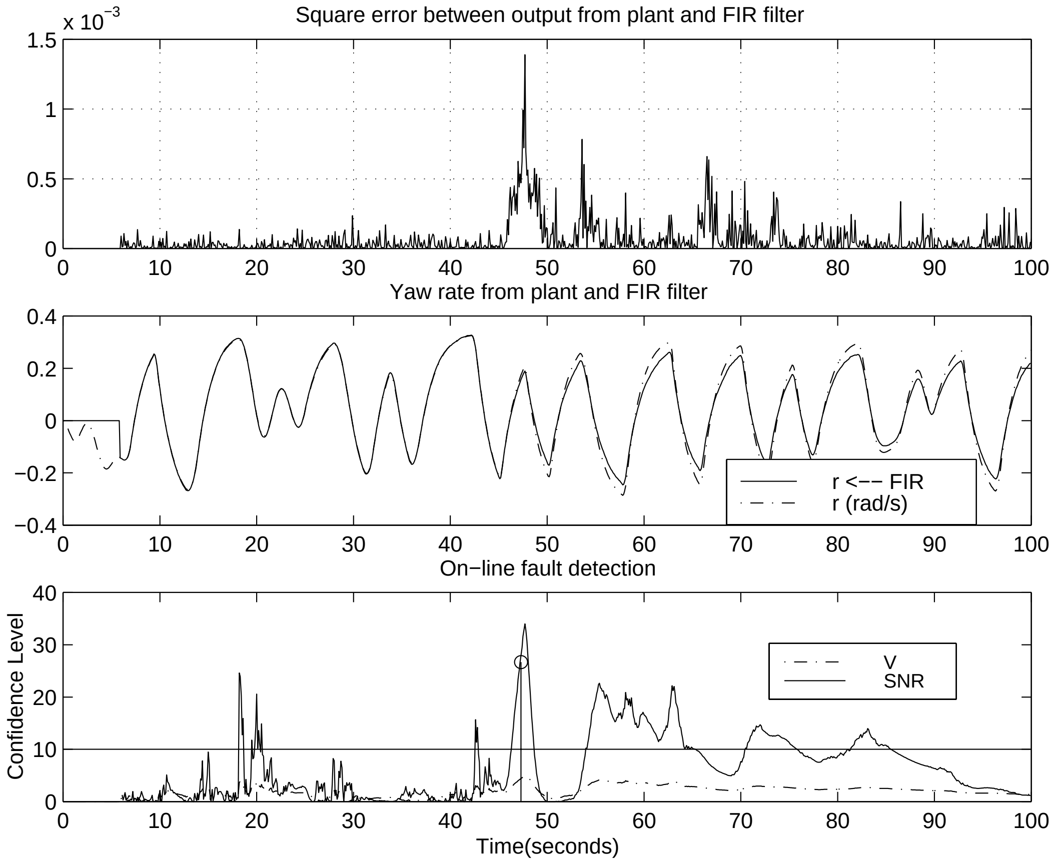

Figure 3.17: Dual-Fault Detection in Steering Subsystem

Figure 3.16 also shows several spikes around $t = 20s$. However, these spikes do not trigger alarms, because there are only a few points of significant values, and they later drop below the threshold value. Besides relying on the SNR for supervising system health, we also monitor the output error between the monitored system and the FIR filter. A significant output error usually denotes sudden changes in the monitored system. It provides additional information for fault detection.

Figure 3.17 simulates a dual total failure of no response of upper-bow and lower-bow rudder. The first failure occurs at $t = 45s$, and the second failure appears at $t = 50s$. The SNR begins to rise shortly after $t = 45s$, and the method detects the failure at $t = 47.6s$. For the second failure, however, it is hard to detect the occurrence through decision logic,

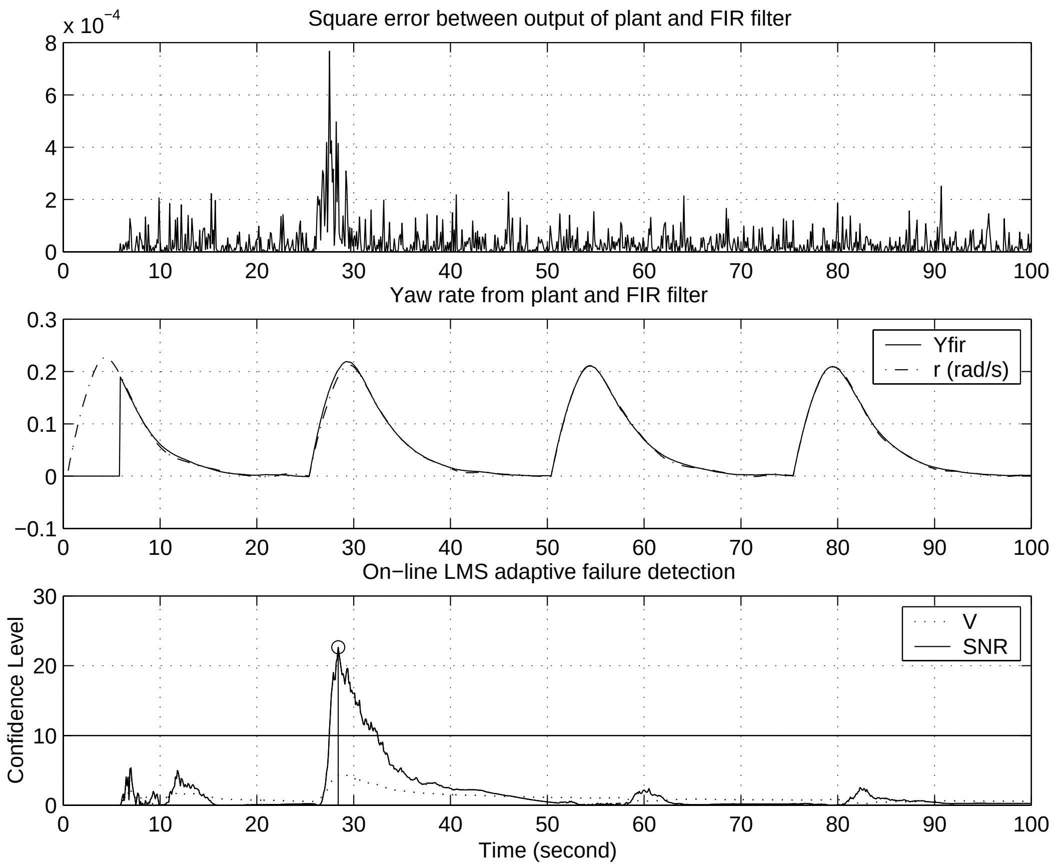

Figure 3.18: Partial-Failure Detection in Steering Subsystem

because the system will remain faulty after the first failure being introduced. The method needs an additional logic to identify the second failure's occurrence.

The adaptive analysis algorithm focuses on the detection of the initial fault occurrence, which is very important in the applications in UUVs, because the UUV is no longer in normal operation after the initial failure. We need to apply fault identification and compensation methods for continued operation of the vehicle after detecting the initial failure.

Figure 3.18 simulates a partial failure of upper-bow rudder, which is reduced by 50% in its effectiveness. The reduction of rudder effectiveness is equivalent to scaling the rudder command by 50%. We simulate the failure at $t = 25s$, and the method detects it at $t = 28.4s$.

We can observe a large spike of square error around that time.

Figure 3.19 to 3.21 show fault-detection results of the diving subsystem. Figure 3.19 simulates a total failure of no stern-plane reponse at $t = 45s$. It compares the pitch-rate from the plant and the output of the adaptive filter, and shows a detection of the failure at $t = 51.8s$.

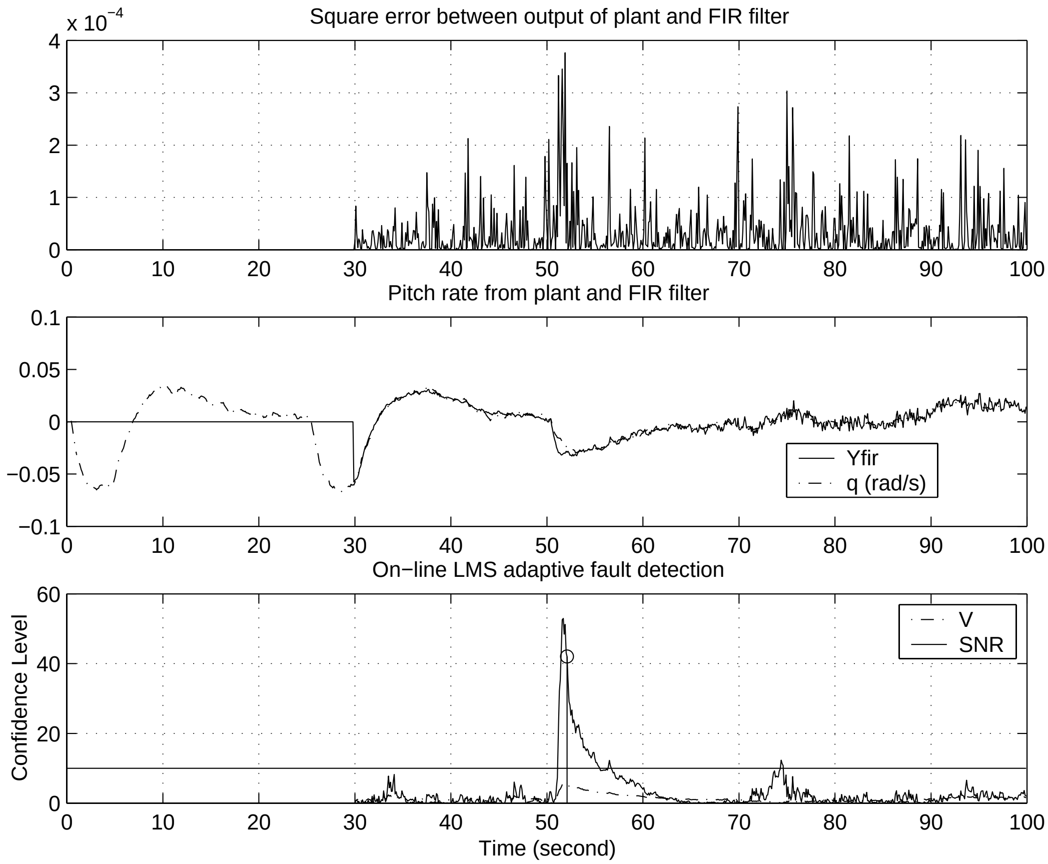

Figure 3.19: Single-Fault Detection in Diving Subsystem

Figure 3.20 simulates a total stern-plane failure at $t = 45s$, and a total pitch-rate failure at $t = 55s$. We detect the first failure at $t = 51.6s$. We can also see the occurrence of the second failure, although we did not announce it.

Figure 3.21 demonstrates the fault-detection result for a partial failure of pitch-rate

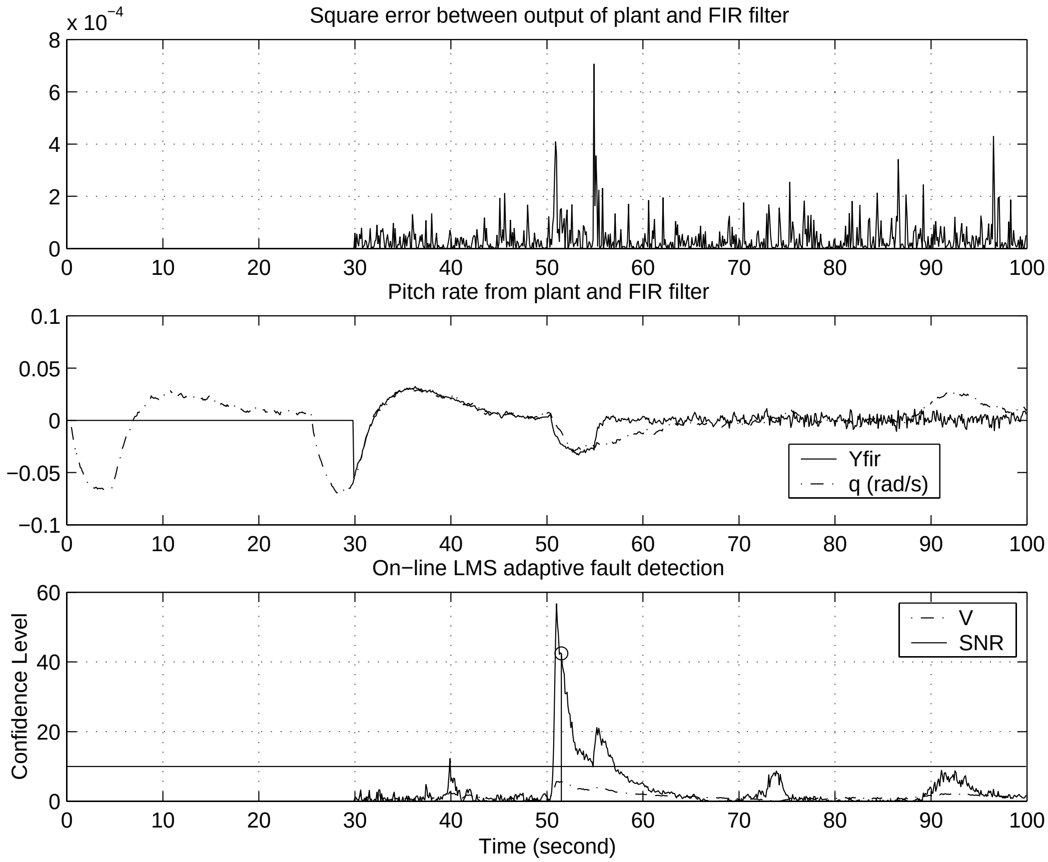

Figure 3.20: Dual-Fault Detection in Diving Subsystem

sensor at $t = 45s$. The sensor is only half effective, or the measurements from the sensor are only 50% of their full scale. Although it is hard to observe the fault occurrence from the output square error, our method easily detects the failure at $t = 52.2s$.

3.4 Summary

This Chapter develops a new method of fault detection for UUV based on LMS adaptive filtering and on-line information extraction from the resulting adaptive FIR coefficient history. We show that the method is effective and robust in system supervision and fault detection in simulated operation of NPS UUV steering and diving subsystems. Our method is very

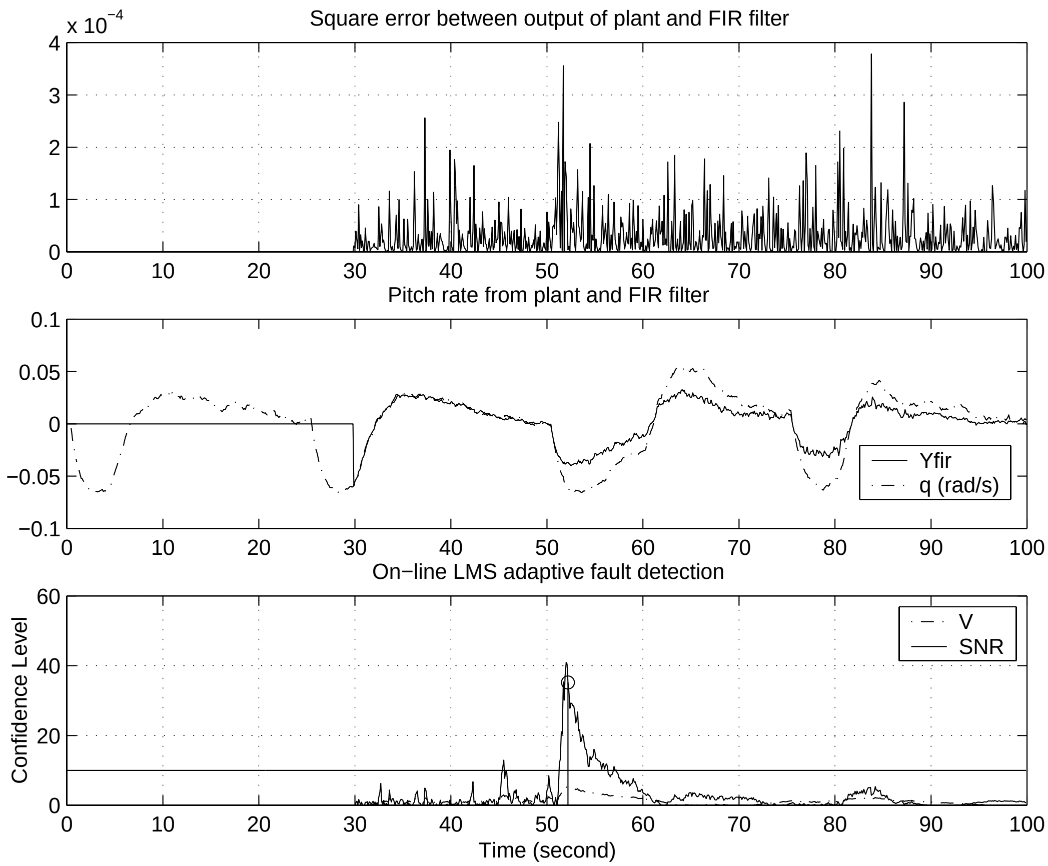

Figure 3.21: Partial-Failure Detection in Diving Subsystem

simple to implement, requiring relatively small computational or processor overhead. It thus can be applied on-line during the UUV missions without taking much of the computing resources. The method is quite general and is applicable to other related fault-detection problems. It can also be used in conjunction or sequentially with fault-identification methods for efficiently and automatically identifying the nature of a failure after detecting its occurrence.

Chapter 4

Hierarchical Fault Detection and Identification (HFDI)

4.1 Introduction

To develop a fault-tolerant control system for UUVs, it is desirable to identify the type of failure as well as to detect it. The fault detection method we developed in Chapter 3 is effective in fault detection and simple in implementation. Like other methods in the category of parameter estimation, it needs additional logic to identify the failure type.

We mentioned in the literature review that the multiple-model adaptive estimation (MMAE) method in the category of state estimation is widely used for fault detection and identification in various dynamic systems [20,21,30]. The advantage of this method is that it not only triggers fault alarm, but also identifies specific failures among a hypothesized set of all possible failure modes. However, we also mentioned the computational load of the method is very high, since it uses a bank of parallel Kalman filters, especially when the number of hypothesized models is large. Also, the system may not experience failures during the mission. Therefore, it is a waste of computational resources to run the MMAE all the time, but only detect normal operation.

An immediate question following the above discussion is: can we find a method that is effective in fault detection and identification, but needs less computation overhead? To solve this problem, we propose a Hierarchical Fault Detection and Identification (HFDI) method with applications to NPS UUV subsystems. The HFDI method combines advantages of the above two methods by using the relative simple and computationally efficient method of adaptive LMS modeling and analysis for fault detection, and implementing the MMAE algorithm for fault identification at a latter stage. The HFDI method implements MMAE only after failures have been detected, thus avoiding unnecessary computation overhead from using MMAE continuously.

In this chapter, we first introduce the Kalman filtering theory and the MMAE algorithm for fault identification. Then, we describe the hierarchical methodology for FDI, and discuss techniques that influence the performance of the algorithm. Finally, we present simulation results and conclusions.

4.2 MMAE for Fault Identification

The Kalman filtering [55] is a method of optimal linear recursive filtering, which provides the solution to a class of minimum mean-square estimation problems. The Kalman filter is formulated as in the following.

Assume that there is a linear perturbation stochastic state model of a system in Eq. (4.1):

$$\begin{cases} X_{k+1} = \Phi X_k + \Gamma u_k + W_k \\ z_k = H X_k + V_k \end{cases} \tag{4.1}$$

where W_k and V_k are white-noise sequences with the following statistic characteristics:

$$E[W_k] \;=\; E[V_k] = 0 \tag{4.2}$$

$$E[W_k W_j^T] = Q_k \delta_{kj} \tag{4.3}$$

$$E[V_k V_j^T] = R_k \delta_{kj} \tag{4.4}$$

$$E[W_k V_j^T] = 0 \tag{4.5}$$

In the above equations, Q_k is a non-negative matrix, and R_k is a positive matrix. δ_{kj} is a Kroneker δ function as:

$$\delta_{kj} = \begin{cases} 1 & k = j \\ 0 & k \neq j \end{cases} \tag{4.6}$$

Assume that the initial value of the state variable obeys a normal distribution, and has the following characteristics.

$$E[X(0)] = m_0 \tag{4.7}$$

$$E[(X(0) - m_0)(X(0) - m_0)^T] = P_0 \tag{4.8}$$

The well-known Kalman filtering equations for each cycle are as follows:

Updating

$$K(k) = P(k|k-1)H^T[HP(k|k-1)H^T + R_k]^{-1} \tag{4.9}$$

$$\widehat{X}(k) = \widehat{X}(k|k-1) + K(k)[z(k) - H\widehat{X}(k|k-1)] \tag{4.10}$$

$$P(k) = [1 - K(k)H]P(k|k-1) \tag{4.11}$$

Prediction

$$\widehat{X}(k+1|k) = \Phi\widehat{X}(k) + \Gamma u_k \tag{4.12}$$

$$P(k+1|k) = \Phi P(k)\Phi^T + \Gamma Q_k \Gamma^T \tag{4.13}$$

The algorithm of multiple-model adaptive estimation consists of a bank of parallel Kalman filters. Figure 4.1 shows the schematic diagram of the MMAE.

Each of the Kalman filters assumes a different linear stochastic model for the dynamic system, taking a particular parameter sequence from a discrete set $[\theta_1, \theta_2, ...\theta_n]$. The inputs

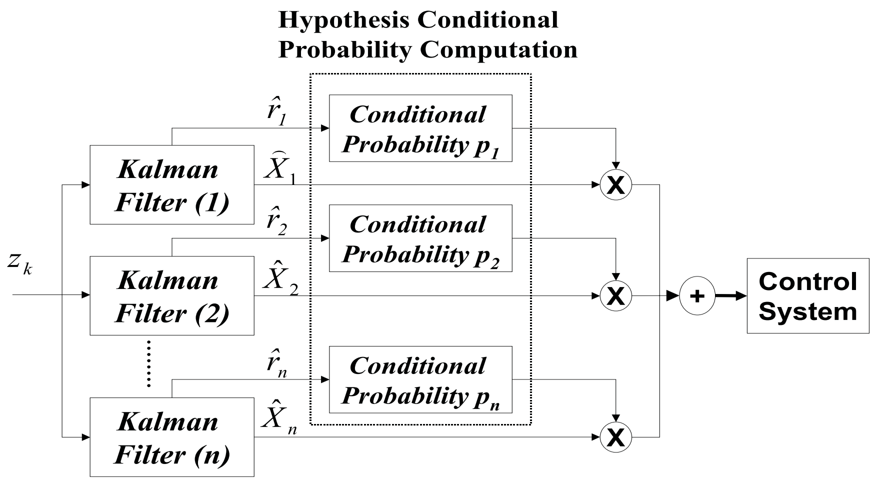

Figure 4.1: Multiple Model Adaptive Estimation

to the bank of Kalman filters are the discrete observation vector and control inputs. A hypothesis testing algorithm uses the residuals from each Kalman Filter to assign conditional probability to each of the hypothesis $\theta = \theta_i$. These probabilities are the so-called *a posteriori* probabilities which denote how correct each hypothesis is, and act as the weighting coefficients for each state estimate in the combined state estimation. We also obtain the weighted sum of the corresponding state-estimation error-covariance matrix. The algorithm feeds back the combined state estimation to the controller for generating the next step control signal.

The computation of the *a posteriori* probabilities is the key point of MMAE, and it is also very crucial for the purpose of fault identification. By monitoring the *a posteriori* probabilities of elemental filters, we make decisions on whether the system has failed or not, and how. We derive the recursive method for updating *a posteriori* probabilities in the following [56].

Suppose that the dynamic model is operating with an unknown parameter sequence θ from a discrete set $[\theta_1, \theta_2, ..., \theta_n]$. An application of Bayes' rule yields:

$$p(\theta_i|Z_k) = \frac{p(Z_k, \theta_i)}{p(Z_k)} = \frac{p(Z_k|\theta_i)p(\theta_i)}{\sum_{i=1}^{n} p(Z_k|\theta_i)p(\theta_i)} \tag{4.14}$$

where Z_k denotes a sequence of measurements $[z_0, z_1, ..., z_k]$ up to time k, $p(\theta_i|Z_k)$ is the *a posteriori* probability, that is, a conditional probability when $\theta = \theta_i$, and $p(Z_k|\theta_i)$ is the likelihood function.

Furthermore, we have:

$$\begin{aligned} p(\theta_i|Z_k) &= \frac{p(z_k, Z_{k-1}, \theta_i)}{p(z_k, Z_{k-1})} = \frac{p(z_k, \theta_i|Z_{k-1})p(Z_{k-1})}{p(z_k|Z_{k-1})p(Z_{k-1})} \\ &= \frac{p(z_k, \theta_i|Z_{k-1})}{p(z_k|Z_{k-1})} = \frac{p(z_k|Z_{k-1}, \theta_i)p(\theta_i|Z_{k-1})}{\sum_{i=1}^{n} p(z_k|Z_{k-1}, \theta_i)p(\theta_i|Z_{k-1})} \end{aligned} \tag{4.15}$$

The denominators of both (4.14) and (4.15) are normalizing constants. If the conditional probability density $p(z_k|Z_{k-1}, \theta_i)$ is Gaussian with a mean value $\hat{z}_{k|\theta_i}$ and a covariance $\Omega_{k|\theta_i} = E[\tilde{z}_{k|\theta_i}\tilde{z}_{k|\theta_i}^T]$, then we can calculate the *a posteriori* probability recursively for the assumption $\theta = \theta_i$, up to the kth sampling interval with Eq. (4.16).

$$p(\theta_i|Z_k) = \frac{1}{c}|\Omega_{k|\theta_i}|^{-1/2} \exp(-\frac{1}{2}\tilde{z}_{k|\theta_i}^T \Omega_{k|\theta_i}^{-1} \tilde{z}_{k|\theta_i})p(\theta_i|Z_{k-1}) \tag{4.16}$$

where c is a normalizing constant chosen to ensure $\sum_{i=1}^{n} p(\theta_i|Z_k) = 1$, and $\tilde{z}_{k|\theta_i}$ is the residual defined by $z_k - \hat{z}_{k|\theta_i}$.

We derive the covariance $\Omega_{k|\theta_i}$ in the following. For simplicity, we omit θ_i in the expressions.

$$\begin{aligned} \Omega_{k|\theta_i} &= E[\tilde{z}\tilde{z}^T] = E[(z_k - \hat{z}_k)(z_k - \hat{z}_k)^T] \\ &= E[(H\widetilde{X}_k + V_k)(H\widetilde{X}_k + V_k)^T] \\ &= E[(H\widetilde{X}_k + V_k)(\widetilde{X}_k^T H^T + V_k^T)] \\ &= E[H\widetilde{X}_k\widetilde{X}_k^T H^T + H\widetilde{X}_k V_k^T + V_k\widetilde{X}_k^T H^T + V_k V_k^T] \\ &= HP_kH^T + R_k \end{aligned} \tag{4.17}$$

We assume $E[\widetilde{X}_k V_k^T] = 0$ in the above calculation. This assumption holds if the model and measurement are reasonably good, or if both model and measurement are biased. P_k is the state-error covariance matrix defined by:

$$P_k = E[\widetilde{X}_k \widetilde{X}_k^T] \tag{4.18}$$

We re-write the *a posteriori* probability in Eq. (4.16) as:

$$\begin{aligned} p(\theta_i|Z_k) &= \frac{|\Omega_{k|\theta_i}|^{-1/2}\exp(-\frac{1}{2}\tilde{z}^T_{k|\theta_i}\Omega^{-1}_{k|\theta_i}\tilde{z}_{k|\theta_i})p(\theta_i|Z_{k-1})}{\sum_{i=1}^{n}|\Omega_{k|\theta_i}|^{-1/2}\exp(-\frac{1}{2}\tilde{z}^T_{k|\theta_i}\Omega^{-1}_{k|\theta_i}\tilde{z}_{k|\theta_i})p(\theta_i|Z_{k-1})} && (4.19) \\ \Omega_{k|\theta_i} &= H_i P_i(k) H_i^T + R_k^i && (4.20) \end{aligned}$$

Suppose that the dynamic model is operating with the parameter θ_1, i.e., $\theta = \theta_1$, we expect that *a posteriori* probability for θ_1 converges to one, while all the other probabilities for $\theta \neq \theta_1$ converge to zero as k goes to infinity (see Eq. 4.21). This follows because the residuals of the matched filter are supposed to have a mean-squared value consistent with its own computed $\Omega_{k|\theta_1}$, while residuals of other filter are larger than anticipated due to the mismatch of θ with the real value.

$$\begin{cases} p(\theta_1|Z_k) \to 1 & \text{as } k \to \infty \\ p(\theta_j|Z_k) \to 0 & j \neq 1, k \to \infty \end{cases} \tag{4.21}$$

For any two different θ values, θ_i and θ_j, if the difference between residuals z_i and z_j does not approach zero as k goes to infinity, or if their covariance matrices are different, or if both conditions are satisfied, then the convergence of the probabilities is guaranteed [56].

The MMAE during one cycle is as follows:

Filtering

$$\begin{aligned} \widehat{X}_i(k+1|k) &= \Phi\widehat{X}_i(k|k-1) + \Gamma_i u_k + K_i(k)[z_k - H_i\widehat{X}_i(k|k-1)] && (4.22) \\ K_i(k) &= \Phi P_i(k|k-1)H_i^T[H_iP_i(k|k-1)H_i^T + R_k^i]^{-1} && (4.23) \\ P_i(k+1|k) &= \Phi P_i(k|k-1)\Phi^T - K_i(k)H_iP_i(k|k-1)\Phi^T + \Gamma Q_k^i \Gamma^T && (4.24) \end{aligned}$$

Combination

$$\widehat{X}(k) = \sum_{i=1}^{n} p(\theta_i|Z_k)\widehat{X}_i(k) \tag{4.25}$$

$$P(k) = \sum_{i=1}^{n} p(\theta_i|Z_k)[P_i(k) + (\widehat{X}_i(k) - \widehat{X}(k))(\widehat{X}_i(k) - \widehat{X}(k))^T] \tag{4.26}$$

For the purpose of fault detection and identification, the parameter set of $[\theta_1, \theta_2, ...\theta_n]$ represent different failure scenarios. We tune the best performance of each Kalman filter for a particular sensor or actuator failure, assuming that it is the true scenario. The *a posteriori* probability for each filter denotes the possibility of the occurrence for each scenario. If the above convergence criterion is satisfied, then the *a posteriori* probabilities from the matched filter will have a dominant value over others. Therefore, by examining the *a posteriori* probabilities, we identify the type of failure occurrence.

4.3 Hierarchical Fault Detection and Identification

We propose a hierarchical fault detection and identification method for applications to the NPS UUV subsystems. Figure 4.2 is the block diagram of the hierarchical method. It has three levels. In level one, we monitor the system through adaptive LMS modeling, and detect sensor and/or actuator failures with the help of adaptive analysis. In levels two and three, we identify single and/or dual failures with MMAE algorithm.

Level one uses a single Kalman filter for state estimation, assuming that the system is in the mode of normal operation. If there is no failure detected in level one, then it continues; otherwise, the algorithm loads a bank of Kalman filters at level two to identify the failure detected from the previous level. If level two identifies a normal mode, then the algorithm will return to level one; otherwise, it starts level three with a second bank of Kalman filters to identify double failures, and compensate failures. Again, if only normal operation mode is identified, then the algorithm will return to level one, since no failure actually happened.

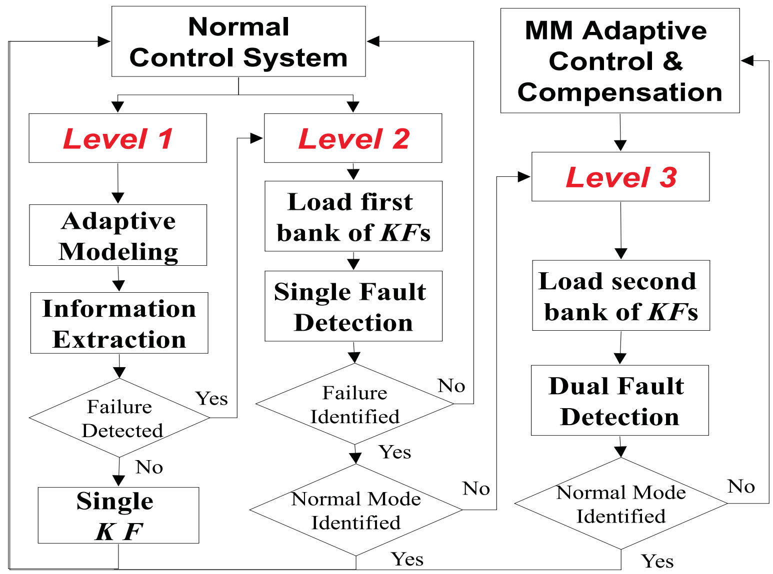

Figure 4.2: Schematic Diagram of Hierarchical Fault Detection and Identification

Figure 4.3 describes the HFDI with an illustration of the first and the second bank of Kalman filters. $F_i, (i = 1, 2, ..., n)$ represent Kalman filters based on the single fault model. $F_i + F_j, (j = 1, 2, ..., n)$ represent Kalman filters based on the dual fault model. "normal" represents the filter relating to the normal operational model. Basically, the HFDI algorithm has two main loops: the monitoring loop (level one), and the identification loop (levels two and three). In the monitoring loop, we build up a series of adaptive FIR models to monitor the system status, and extract the SNR by adaptive analysis. We design a decision logic based on the SNR. Once the monitoring loop detects any failure according to the decision logic, the algorithm will jump out of the monitoring loop and move into the identification loop; otherwise, the monitoring loop continues.

In the identification loop, we design two-stage Kalman filters for single and/or dual failure identification. We design the first bank of Kalman filters, assuming that only single failures occurred. Therefore, the first bank of Kalman filters only identifies single failures. The elemental filters in the second bank of Kalman filters are based on a different set of hypothesized models, each representing a second hypothesized failure, while assuming the existence of a first failure. Therefore, the identification results from the second bank of Kalman filters denote a dual failure. The first failure is identified in the first bank, and the second failure could be the same failure as the first one, or could be any one of the other failures (referring to Figure 4.3). We assume that there is some time apart between the two fault occurrences of the dual failure. By examining the *a posteriori* probabilities in levels two and three, we can identify the failure model that has a dominant probability. Each bank of Kalman filters includes a fully functional mode, so that the algorithm can quit any time from the identification loop and return back to the monitoring loop in case of false alarm due to disturbances.

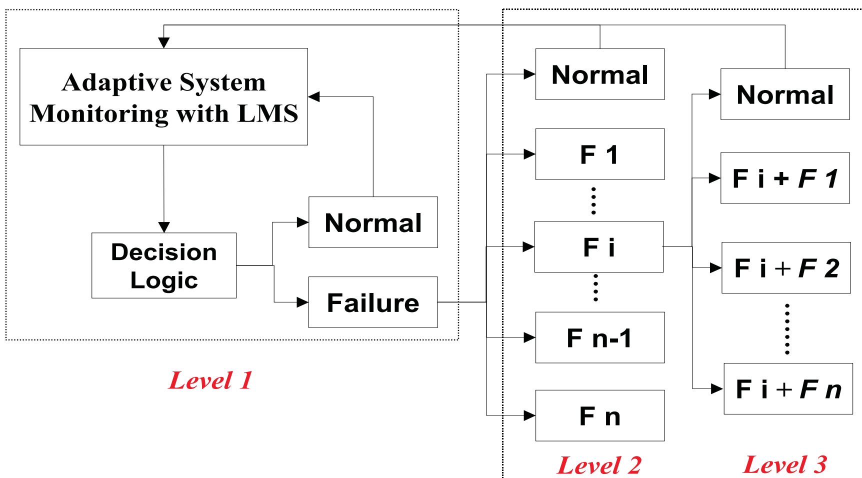

Figure 4.3: Hierarchical Structure

In the HFDI method, we use a single Kalman filter in level one, and banks of parallel Kalman filters in levels two and three. Figure 4.4 illustrates the hierarchical structure of Kalman filters.

To start Kalman filtering (see equations 4.9 to 4.11), we need initial values of the error-covariance matrix P, and the state estimation $\widehat{X}$. In the hierarchical structure, we also need to consider the switching of Kalman filters from level to level. There are three different ways of switching Kalman filters (see Figure 4.4). (1a), We may initiate $\widehat{X}$ and P

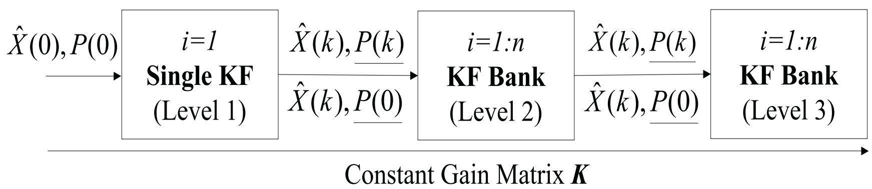

Figure 4.4: Hierarchical Kalman Filters

at the beginning, and then initiate parallel Kalman filters in levels two and three with values carried from the previous level. Or (1b), we may initiate them at the beginning, and then re-initiate P every time we switch Kalman filters. The third way is to use the steady-state value of P from the beginning.

Figure 4.5 shows the norm of two-step differences of P and K corresponding to logic (1b). We see three spikes, each relating to the initiation of P with one at the beginning and two at the switching. If it were for logic (1a), we would only see the spike at the beginning.

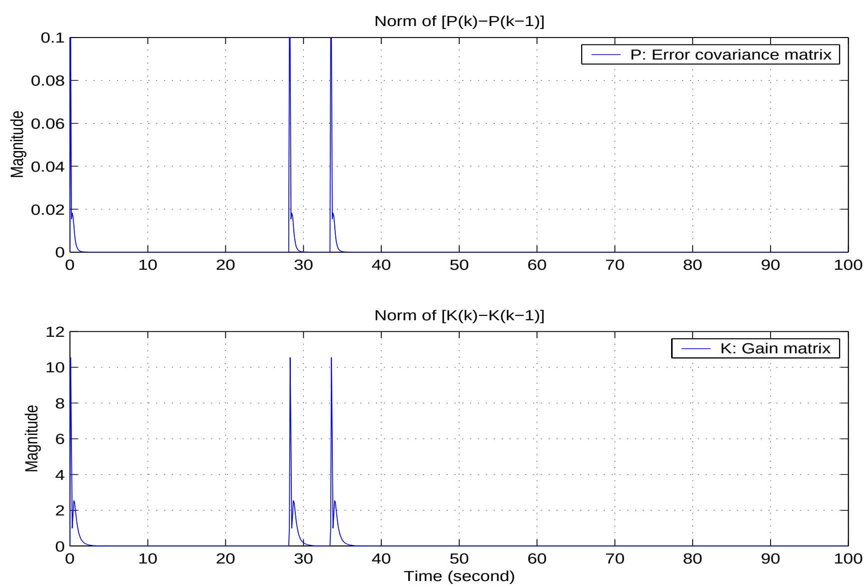

Figure 4.5: Transition Processes of Error-covariance and Gain Matrixes

The first two switching logics belong to the category of changing-gain Kalman filters, in which we calculate the error-covariance matrix P and the gain matrix K recursively. The third way uses constant-gain Kalman filters, in which the error covariance $P(k|k-1)$ converges to a steady-state value P. It results from equations (4.9), (4.11), and (4.13) that P is a solution of the so-called algebraic Riccati equation:

$$P - \Phi P \Phi^T + \Phi P H^T (HPH^T + R_k)^{-1} HP\Phi^T - \Gamma Q_k \Gamma^T = 0 \tag{4.27}$$

As a result, the Kalman gain $K(k)$ also converges to its steady-state value given by:

$$K = PH^T (HPH^T + R_k)^{-1} \tag{4.28}$$

By using the steady-sate value of K, the constant-gain Kalman filter only calculates the state-estimation recursively. Apparently, it saves a lot of computation.

The next section shows the performance of the HFDI algorithm in different fault situations, and compares fault-identification results with different switching logics.

4.4 Simulation Results

We simulate different failure situations according to the fault model in Chapter 3. We assume that failures occur to actuators and sensors randomly, and the dynamic system switches to a failure status after staying at the normal mode for a certain period of time. We describe the failure process by state transitions in the Markov chain, in which discrete states represent the parameter space $\theta_1, \theta_2, ..., \theta_n$. The *a priori* transition-probability matrix is:

$$\begin{array}{c} \\ r_1 \\ r_2 \\ \vdots \\ r_n \end{array} \begin{array}{c} \begin{array}{cccc} r_1 & r_2 & \cdots & r_n \end{array} \\ \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1n} \\ p_{21} & p_{22} & & \vdots \\ \vdots & & \ddots & \\ p_{n1} & \cdots & & p_{nn} \end{bmatrix} \end{array}$$

p_{ij} denotes the transition probability from state r_i and r_j. Apparently, each element in the matrix is non-negative, and the sum of each row is one. We simply assign each element equal values in the simulation, and use them as initial values of the *a posteriori* probability in Eq.(4.19). This assumption is reasonable, because in reality, we have no *a priori* knowledge which scenario would happen.

For adaptive modeling and analysis, we apply same parameters as those used in Chapter 3 for fault detection. The only thing different is that we need to consider switches between the monitoring loop and the identification loop. To implement MMAE, we need to design two-stage Kalman filters first. We consider total failures in the design of filters, so that residuals show significantly different characteristics in correct versus mismatched filters. We initialize Kalman filters with estimated states brought from the previous level to continue the state estimation. The decision logic is very simple: if the *a posteriori* probability of any of the elemental filters in level two is equal to or above 0.49 for ten sequential sample periods, then the algorithm identifies the failure in the first bank. If any of the *a posteriori* probabilities in level three is above 0.9 for ten sequential sample periods, then the dual failure is identified.

For the steering subsystem, we design seven elemental filters in the first bank for models of a normal operation mode (fully functional) and total failures of the upper-bow rudder, the lower-bow rudder, the upper-stern rudder, the lower-stern rudder, the yaw-rate sensor, and the heading-angle sensor. We design seven elemental Kalman filters in the 2nd bank based on dual-failure models of the identified failure plus each one of the above failures.

Figures 4.6 to 4.8 show fault-identification results of HFDI with logic (1a), logic (1b), and constant-gain Kalman filters, respectively, in which we simulate a total failure of upper-bow rudder in the steering subsystem at $t = 25s$. Blue curves represent results from the 1st bank of Kalman filters in level two, and we use left labels to interpret them. Left labels from top to bottom represent the normal operation mode, and total failures of the upper-bow rudder, the lower-bow rudder, the upper-stern rudder, the lower-stern rudder, the yaw-rate

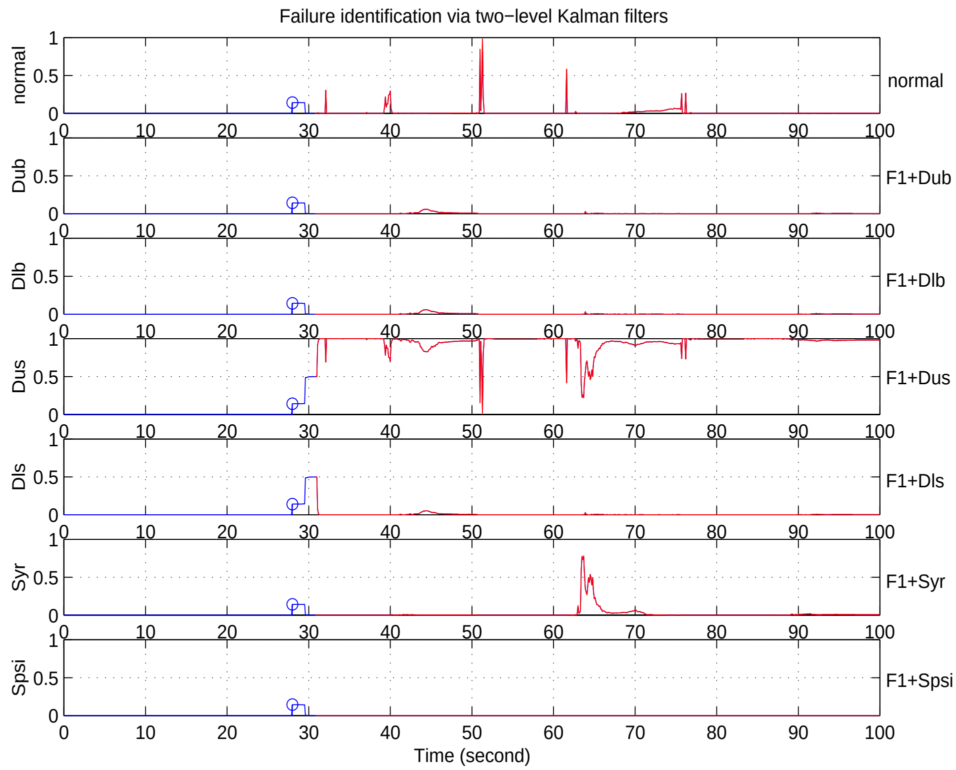

Figure 4.6: Identification Result with Logic (1a)

sensor, and the heading-angle sensor. Red curves are results from bank-two Kalman filters in level three, and we use right labels to explain them. Right labels mean dual failures of F1 plus a second failure. F1 is the failure identified in the first bank. We can regard the single failure as a special dual failure with the second failure the same as F1. "t1" represent the time when the monitoring loop detects the failure, and the identification loop starts. "t2" represents the time when the algorithm identifies a single failure in the first bank, and loads the second bank of Kalman filters. With logic (1b), the algorithm gives equal probabilities to the two bow rudders, and identifies the failure in the first bank as either total upper-bow-rudder failure, or total lower-bow-rudder failure at $t2 = 32.9s$ according to the threshold logic. Due to the same modeling of the function of upper and lower bow rudders in the steering subsystem, bank one Kalman filters could not distinguish them. Since the algorithm assumes that only single failure is identified in the first bank, we choose the number one failure in

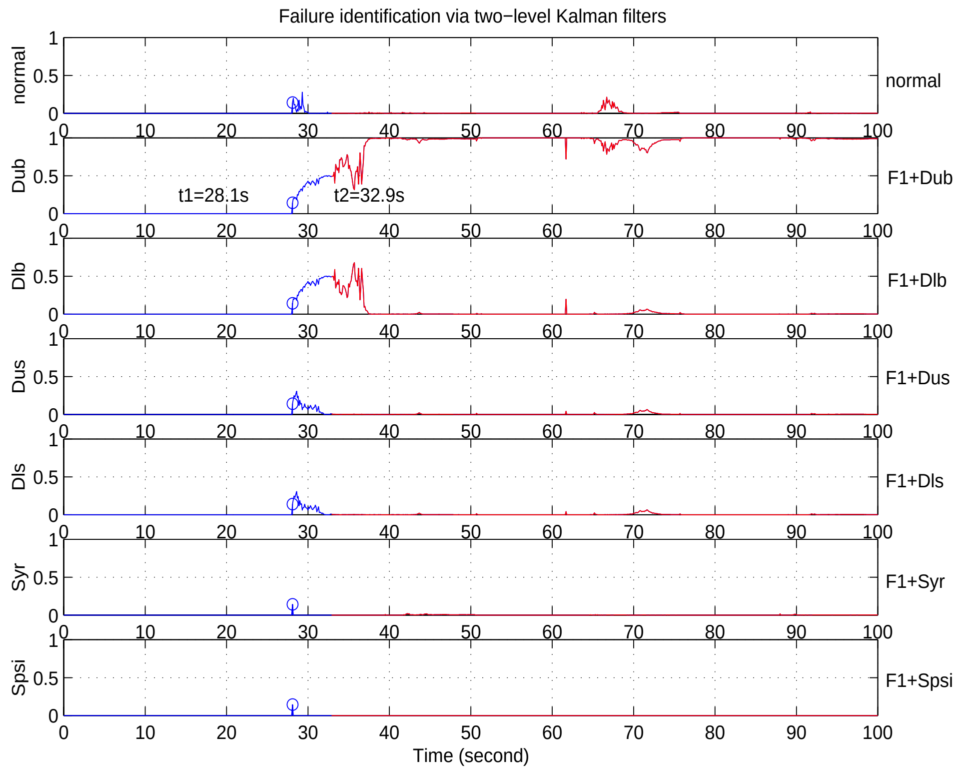

Figure 4.7: Identification Result with Logic (1b)

the sequence of identified models as F1. We choose the upper-bow-rudder failure in this occasion. Finally, the algorithm identifies the failure as a single upper-bow-rudder failure in the second bank. Looking at the identification results with logic (1a), the algorithm identifies stern-rudder failures in the first bank, and the upper-stern-rudder failure in the second bank. Apparently, there is a misidentification with logic (1a). If we compare logic (1b) with the constant-gain case, both identify the failure correctly. But with a constant-gain Kalman filter, the identification result is more clear-cut.

We change the scenario to compare the identification results with different logics again. In Figure 4.9 to 4.11, we simulate a dual failure, including a total upper-stern-rudder failure at $t = 25s$ and a total lower-stern-rudder failure at $t = 30s$. For dual failures, the algorithm tends to identify the first failure in bank one to determine the multiple models for the

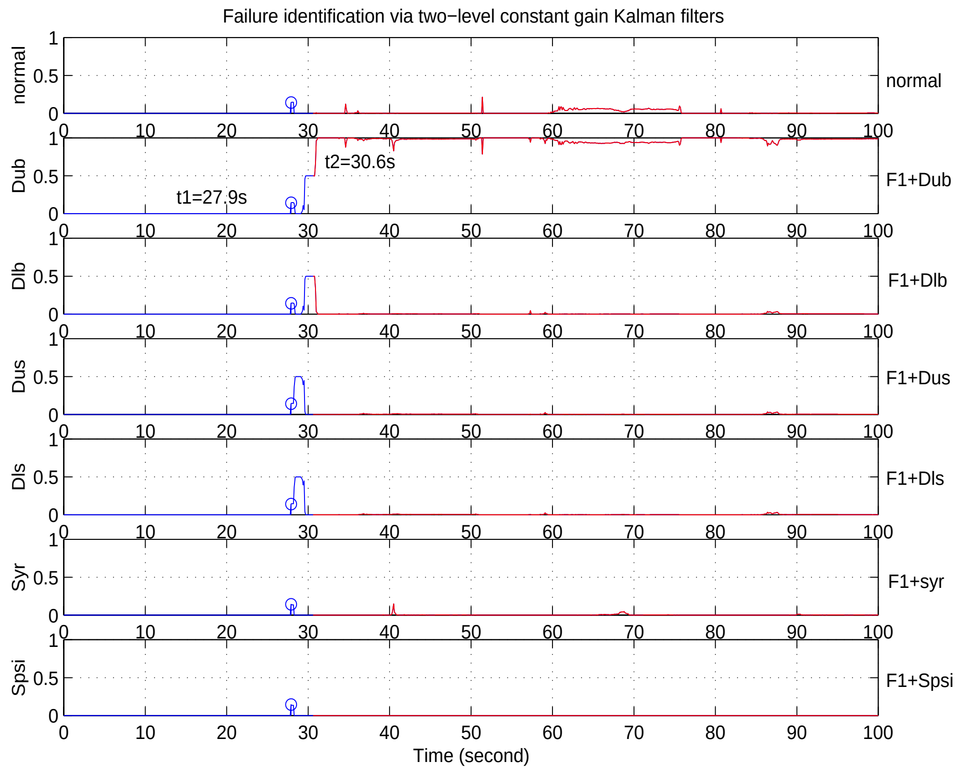

Figure 4.8: Identification Result with Constant Gain

second bank of Kalman filters, and then identify the dual failure in bank two. With logic (1a), the HFDI algorithm identifies the upper-stern-rudder failure at $t2 = 45.2s$, and the dual failure right after that. Like bow rudders, the upper and lower stern rudders have the same modeling, and thus take equal probabilities in bank-one Kalman filters. With logic (1b), the algorithm identifies the upper-stern-rudder failure and the dual failure at $t2 = 31.2s$, with 14 seconds earlier than the $t2$ with logic (1a). However, there are some jumps of probabilities with logic 1(b). With a constant-gain Kalman filter, the algorithm identifies the first failure before the second one occurs. Then, after $t = 30s$, it identifies the second failure and thus the dual failure at $t = 32s$. The identification result with a constant-gain Kalman filter is more focused than with logic (1b). The dispersion of probabilities will influence the state estimation, since the estimated states from parallel Kalman filters is a probability-weighted sum of all the elemental estimations.

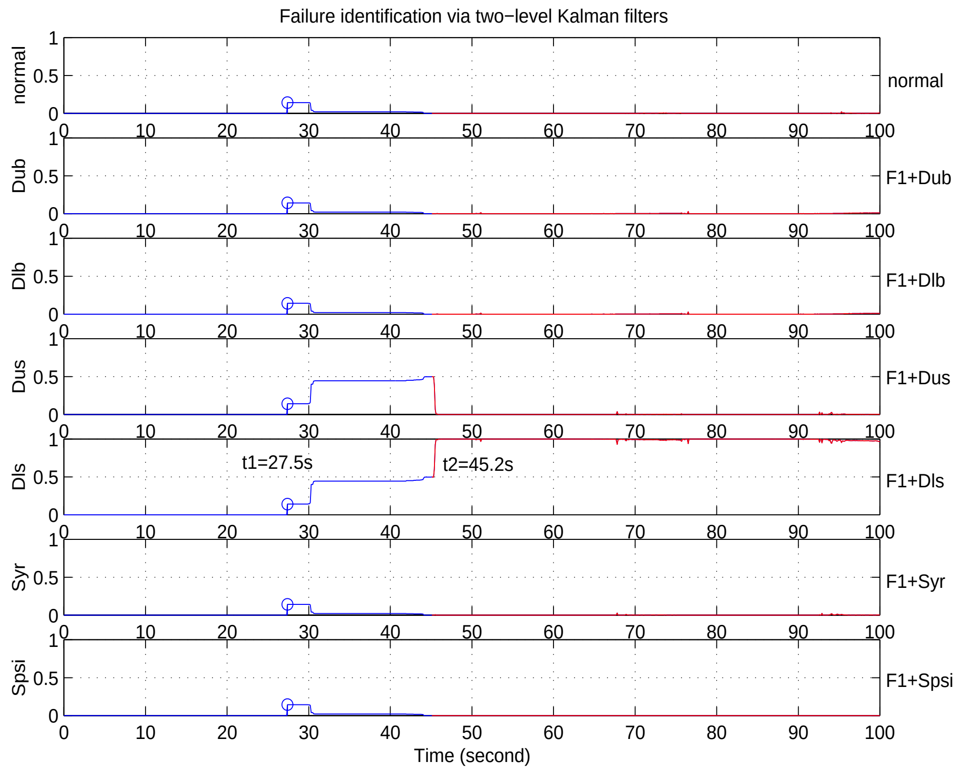

Figure 4.9: Identification Result with Logic (1a)

Through comparison, we find that the HFDI with constant-gain Kalman filters has better performance than with changing-gain Kalman filters, and also saves a lot of computation overhead. Therefore, we use constant-gain Kalman filters in the HFDI algorithm.

In Figure 4.12, we simulate a total yaw-rate-sensor failure at $t = 25s$ and a total heading-angle-sensor failure at $t = 30s$. The monitoring loop detects the failure at $t1 = 27.1s$. The algorithm identifies the total yaw-rate-sensor failure at $t2 = 28.7s$ in the first bank, and identifies the same failure in the second bank Kalman filters. After the heading-angle-sensor failure occurred at $t = 30s$, the algorithm attributes all the probabilities in bank two to the dual failure of yaw-rate sensor and heading-angle sensor.

In Figure 4.13, we simulate a partial failure of the upper-bow rudder with an effectiveness factor 0.5 at $t = 25s$. The HFDI algorithm detects the failure at $t = 28.3s$ and identifies

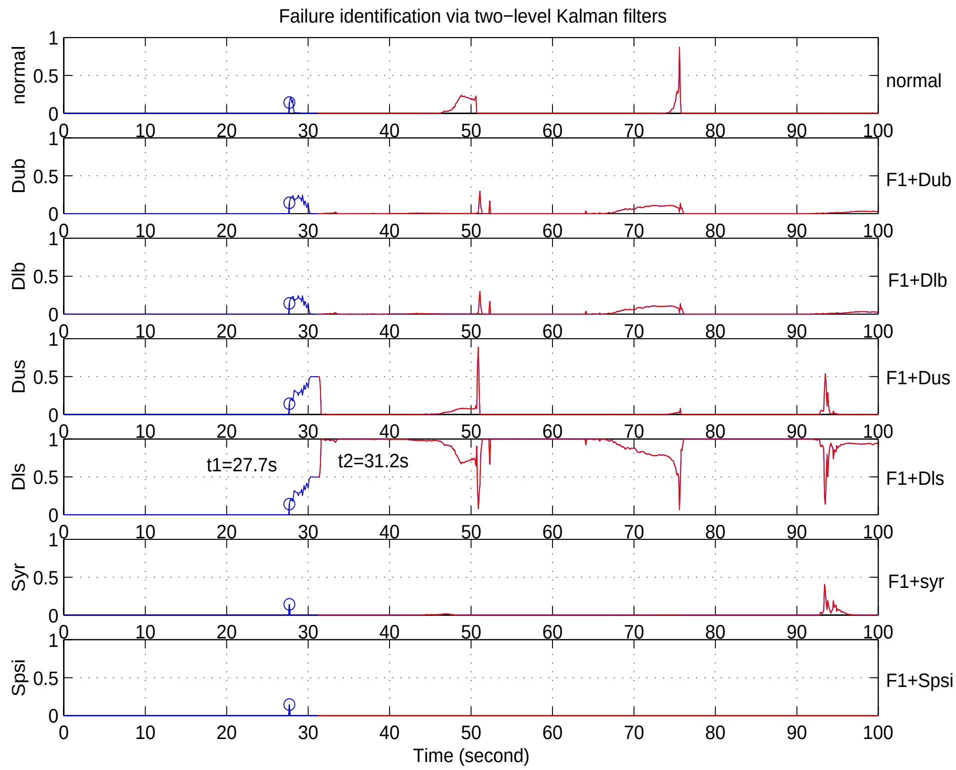

Figure 4.10: Identification Result with Logic (1b)

it as bow-rudder failure in bank one. Then in the second bank, the normal operational mode shares part of the probabilities with the total bow-rudder failure.

In Figure 4.14, we simulate a partial yaw-rate sensor failure with an effectiveness factor of 0.5 at $t = 25s$. The algorithm detects the failure at $t1 = 27.2s$. It identifies the yaw-rate-sensor failure and then shares the probabilities between the normal mode and the total yaw-rate-sensor failure, with an identification of the normal mode. At $t = 51.6s$, the algorithm detects the failure again, and then switches the identification to the normal mode after identifying the yaw-rate-sensor failure for some time. Examining the results in Figure 4.14, they indicate a total yaw-rate-sensor failure at one time, and a normal condition at another time. Figures 4.13 and 4.14 show that a partial-failure identification involves both the normal mode and the failure mode.

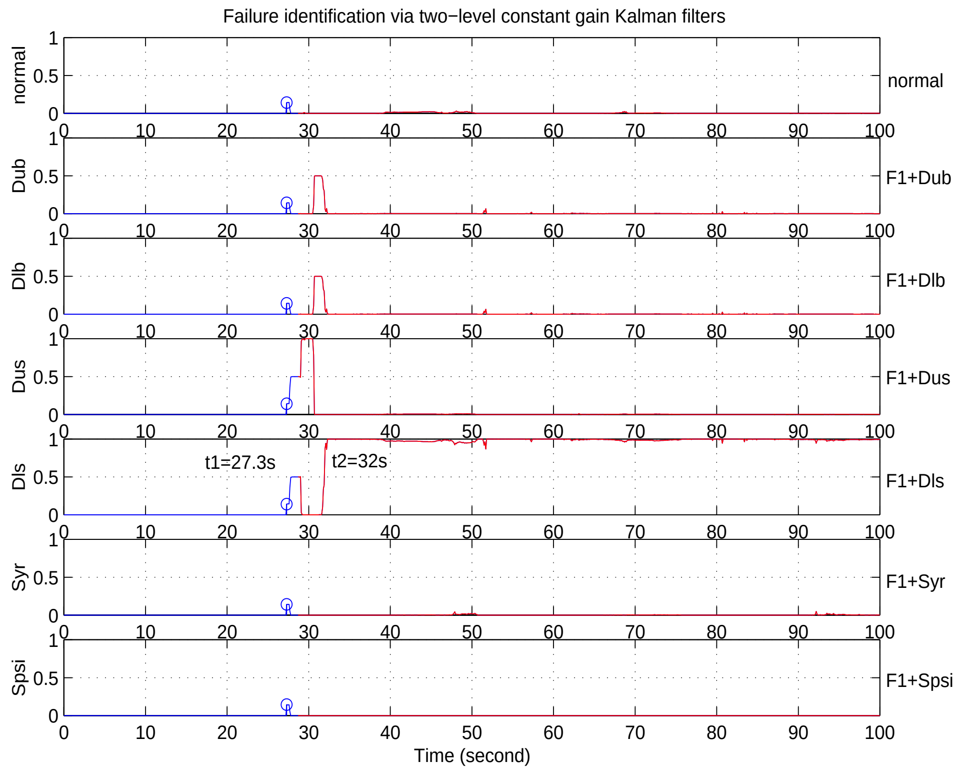

Figure 4.11: Identification Result with Constant Gain

Now we consider an extreme case of dual failure, in which two failures occur simultaneously. This situation is a challenge to the algorithm, since it is against our assumption of having some time apart between the two failures.

First, we simulate a simultaneous dual failure of upper and lower stern rudders in Figure 4.15. Suppose that they are total failures occurring at $t = 25s$, the algorithm detects them at $t1 = 27.2s$. The first bank of Kalman filters does not identify the failure until $t2 = 34.9s$. Then, the algorithm chooses the identification of upper-stern rudder in the first bank and identifies the dual failure in the second bank of Kalman filters. We see some delay in fault identification for simultaneous rudder failures. Although it is not designed for simultaneous failures, the HFDI algorithm identifies the failure correctly.

Again, we simulate total failures of both yaw-rate sensor and heading-angle sensor at

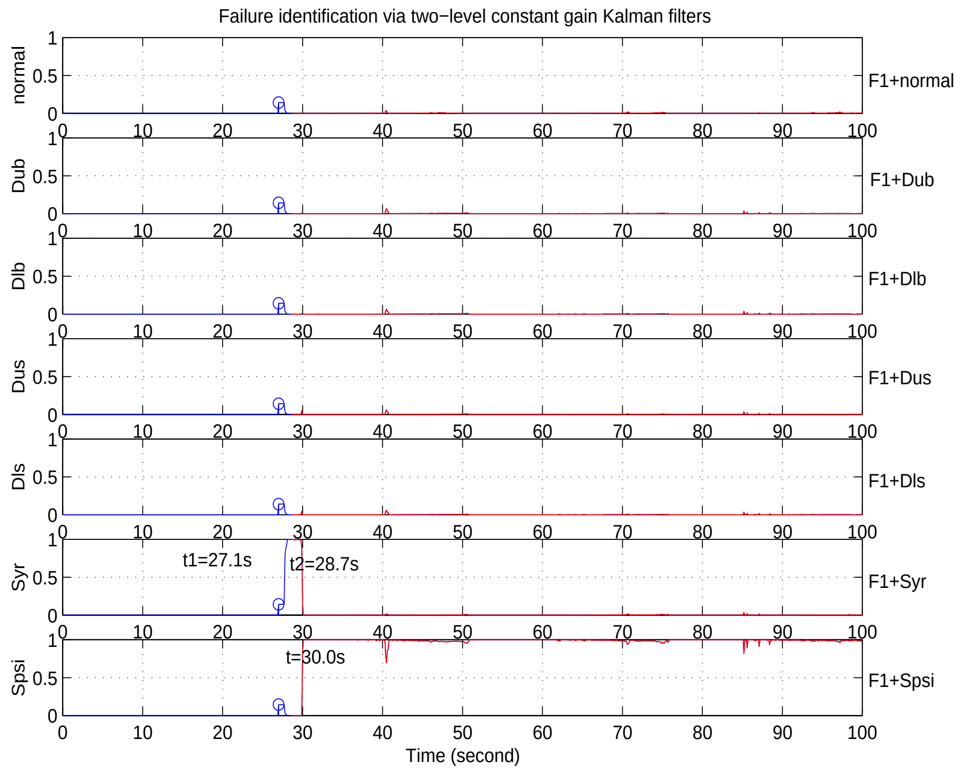

Figure 4.12: Dual Failure Identification via Two-Level Kalman Filters

$t = 25s$ in Figure 4.16. The algorithm detects failures at $t1 = 27.1s$, and identifies the yaw-rate-sensor failure at $t2 = 29.4s$ first. Right after that, the algorithm identifies the dual-sensor failure in the second bank of Kalman filters.

Comparing the results in Figure 4.16 and Figure 4.12, we find that apart from the different occurrence of the heading-angle-sensor failure, the performance of the algorithm does not even degrade for simultaneous sensor failures.

Figures 4.17 and 4.18 are the results of the monitoring loop and the identification loop, respectively. They show the switching between the two loops. Figure 4.17 shows that the HFDI algorithm detects failures at $t = 16.3s$, $t = 68.7s$, and $t = 93.1s$. However, the algorithm returns to the monitoring loop again every time, since it identifies normal modes only in the first bank of Kalman filters. There is actually no failure occurrence. The fault

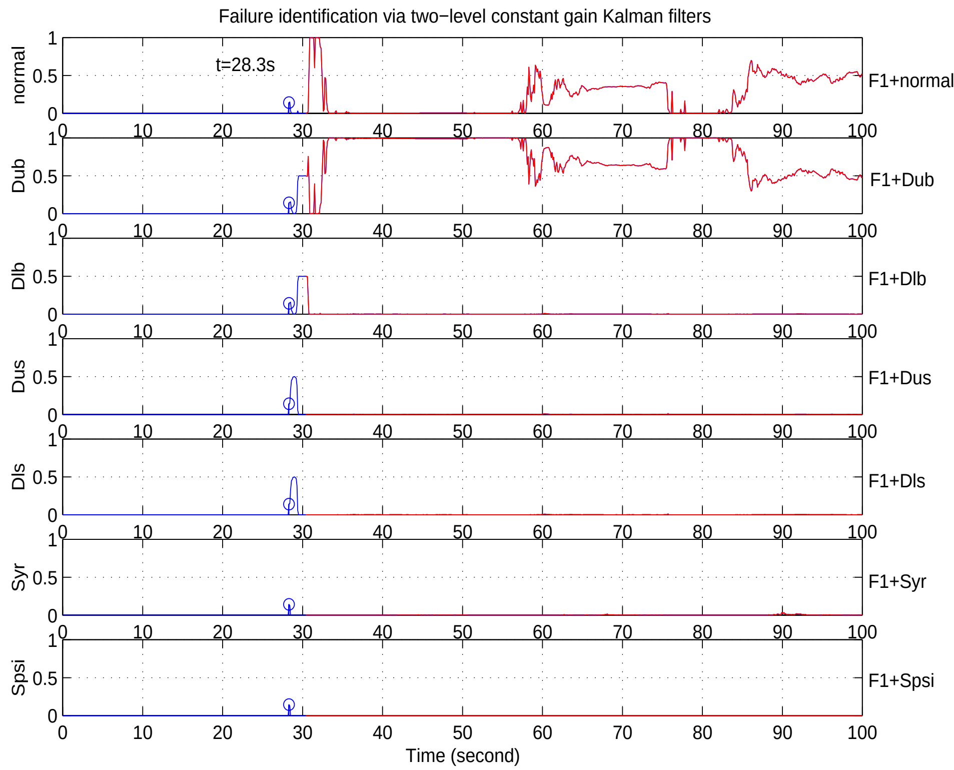

Figure 4.13: Partial Rudder Failure Identification in Steering Subsystem

detection is due to disturbances. It demonstrates the ability of the algorithm to remove false alarms and to identify the failure correctly.

To apply the HFDI algorithm to the diving subsystem, we design six Kalman filters in the first bank based on models of a normal operational mode, and total failures of bow plane, stern plane, pitch-rate sensor, pitch-angle sensor, and depth sensor. In the second bank, we also design six Kalman filters for each of the above situations plus an identified failure from the previous level.

In Figure 4.19, we simulate a total bow-plane failure at $t = 45s$. The algorithm detects the failure at $t1 = 51.5s$, and activates bank-one Kalman filters to identify the bow-plane failure at $t2 = 53.5s$. In bank-two Kalman filters, the algorithm confirms the identification of the single total bow-plane failure.

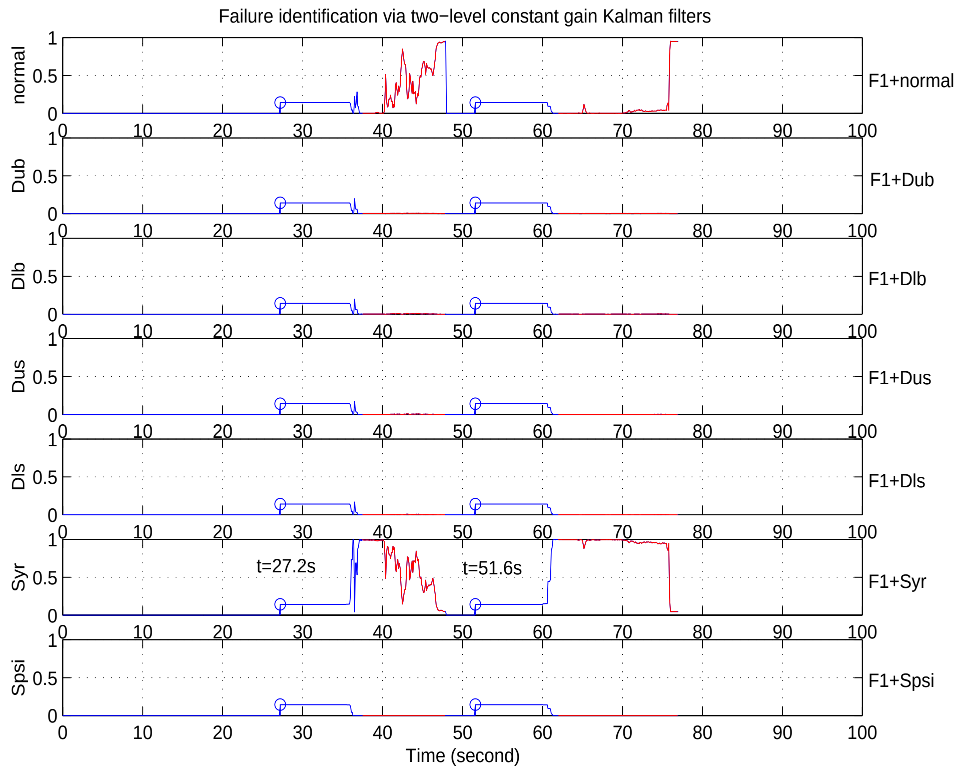

Figure 4.14: Partial Sensor Failure Identification in Steering Subsystem

Figure 4.20 shows the identification result for a dual failure of bow plane at $t = 45s$, and pitch rate at $t = 55s$. The algorithm detects the fault occurrence at $t1 = 52s$, and identifies "F1" as a total bow-plane failure at $t2 = 53.5s$ in the first bank of Kalman filters. In the second bank, the algorithm still confirms the identification until the occurrence of the total pitch-rate failure. At $t = 55.1s$, with a probability transition from "F1+bowP" to "F1+q", the algorithm identifies the dual failure.

Figure 4.21 simulates a partial bow-plane failure at $t = 45s$. We reduce its effectiveness by 50%. The algorithm first detects a failure and identifies it as a bow-plane failure. Later on, the algorithm attributes part of the probabilities to the total bow-plane failure, and part of the probabilities to the normal mode, while none of the other failure modes are involved.

Finally, we simulate two failures occurring simultaneously in the diving subsystem to

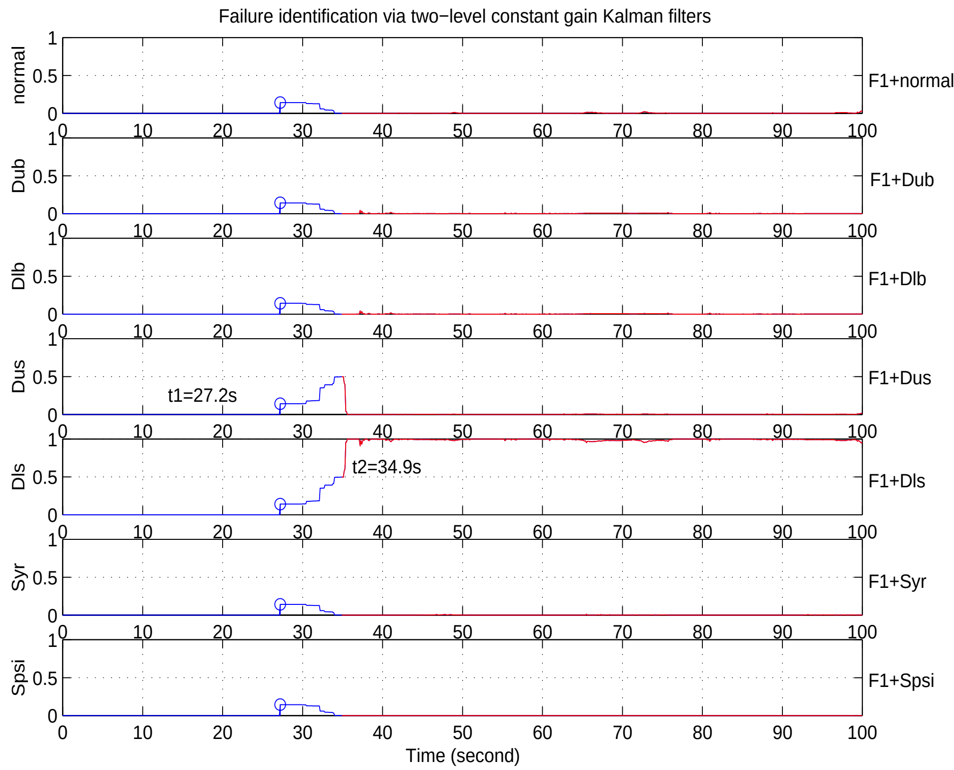

Figure 4.15: Simultaneous-Rudder-Failure Identification in Steering Subsystem

test the algorithm. We simulate both a total stern-plane failure and a total failure of pitch-rate sensor at $t = 45s$. Figure 4.22 shows that the algorithm detects the dual failure at $t1 = 52.1s$, and identifies it as a pitch-rate-sensor failure first at $t = 53.1s$. Then, around $t = 56.2s$, there is a probability transition and the algorithm identifies the dual failure.

4.5 Summary

In this chapter, we have developed a hierarchical methodology for fault detection and identification based on the LMS adaptive filtering and analysis, and the multiple-model adaptive estimation (MMAE) algorithm. The LMS adaptive filtering and analysis is a method of on-line information extraction from resultant adaptive FIR coefficients. It is effective and

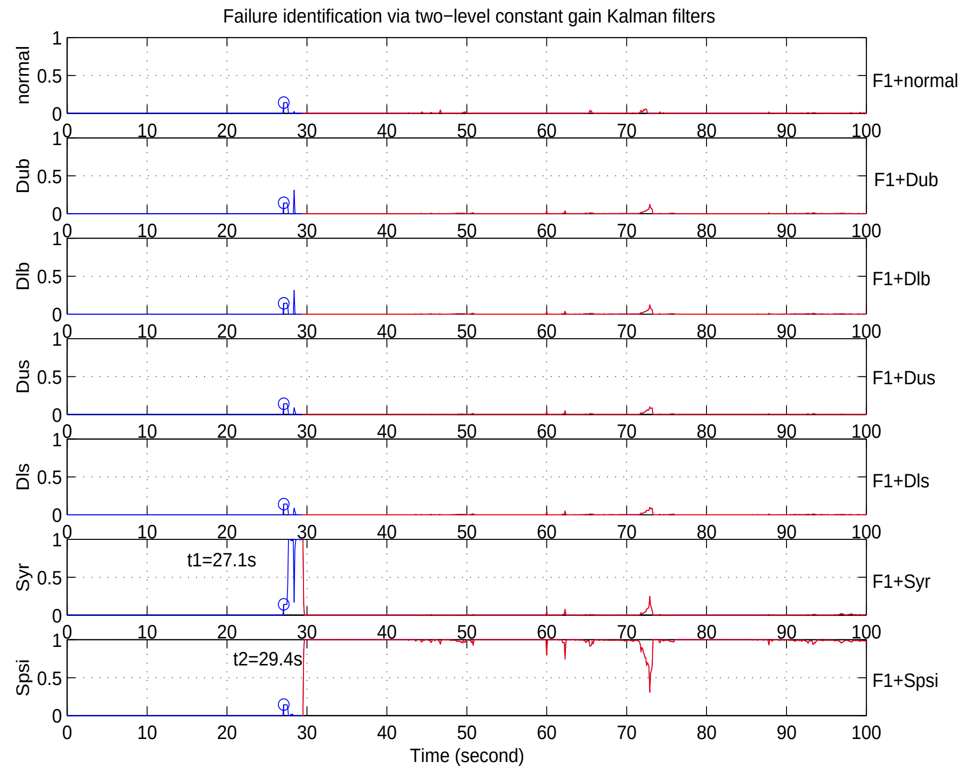

Figure 4.16: Simultaneous-Sensor-Failure Identification in Steering Subsystem

reliable for the purpose of fault supervision and detection, and it is very simple in implementation. The hierarchical methodology uses the adaptive filtering and analysis in its first level for fault detection. If there is no fault occurrence, the algorithm only performs routine supervision. Otherwise, the algorithm applies the MMAE in a two-level design to identify the nature of failures detected. The identification result provides information to the UUV control system for fault compensation. In addition, the MMAE tends to give almost correct state estimations when the actual mode has dominant contributions in the weight averaged state estimations.

The hierarchical methodology combines the two methods together for unified fault detection and identification. It has the advantage of both methods and enables the implementation of a very computationally efficient fault detection and identification procedure.

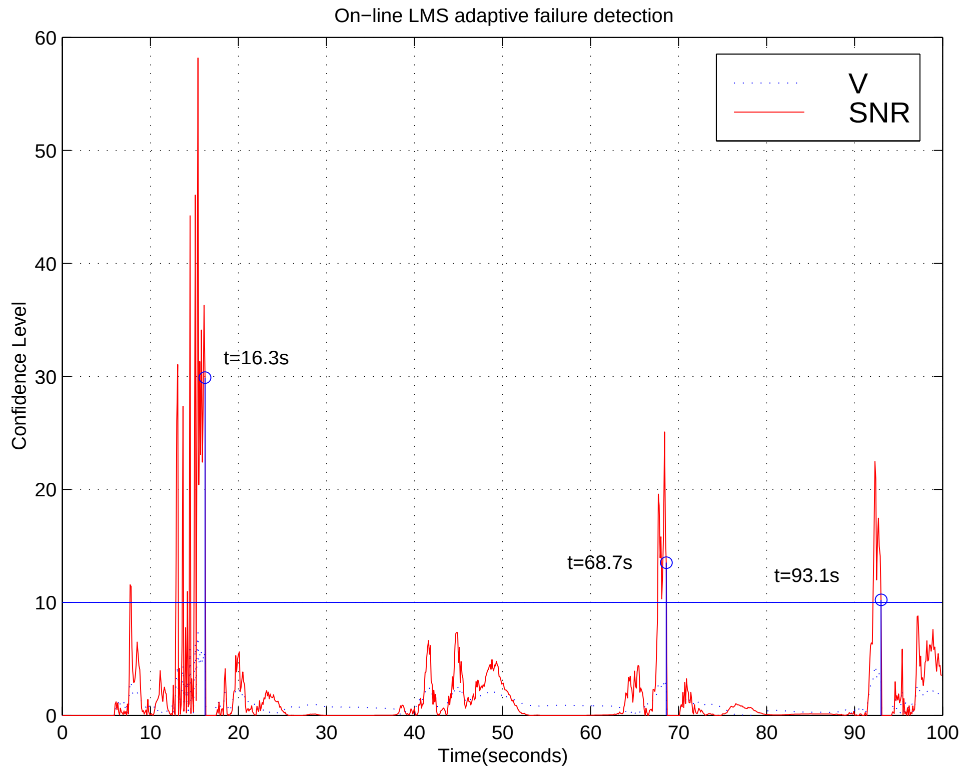

Figure 4.17: Fault Detection

We simulate different failure situations of actuators and sensors in the steering and the diving subsystems, including single, dual, and even simultaneous failures. Simulation results show that the HFDI detects failures and identifies them correctly. The algorithm is effective not only for total failures, but also for partial failures. It also demonstrates its ability to distinguish false alarms from real fault occurrences. We develop the hierarchical method for FDI problems of UUVs. However, it is not restricted to UUV applications only. We can apply the methodology to other applications with similar fault detection and identification problems. We can also develop failure-compensation measures based on the identification result for fault-tolerant control.

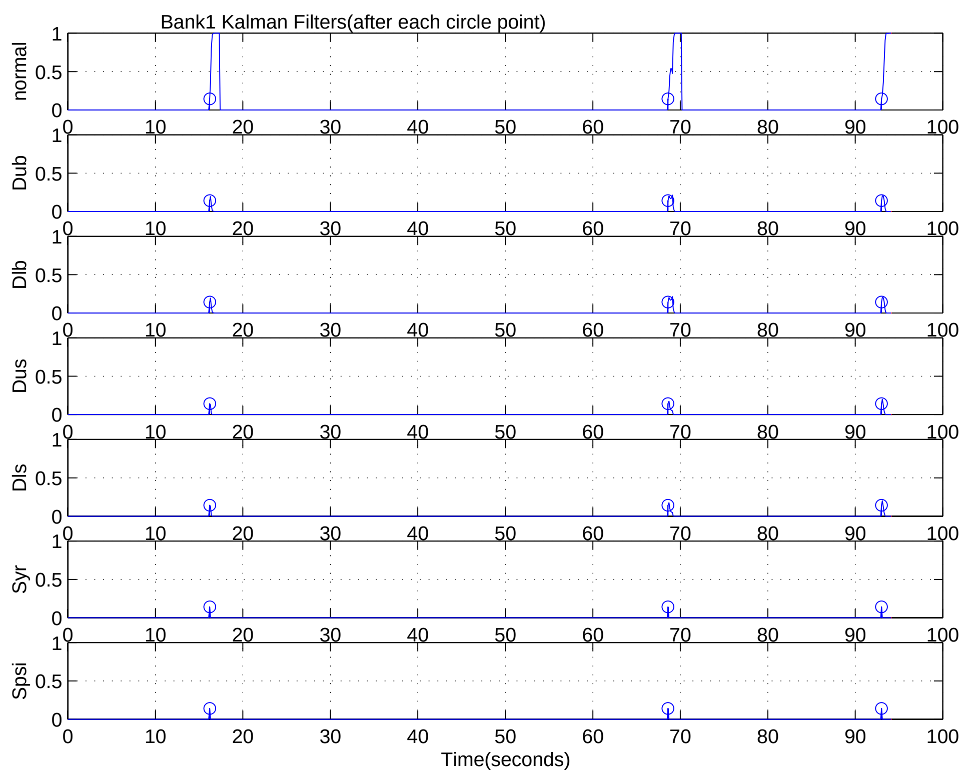

Figure 4.18: Fault Identification

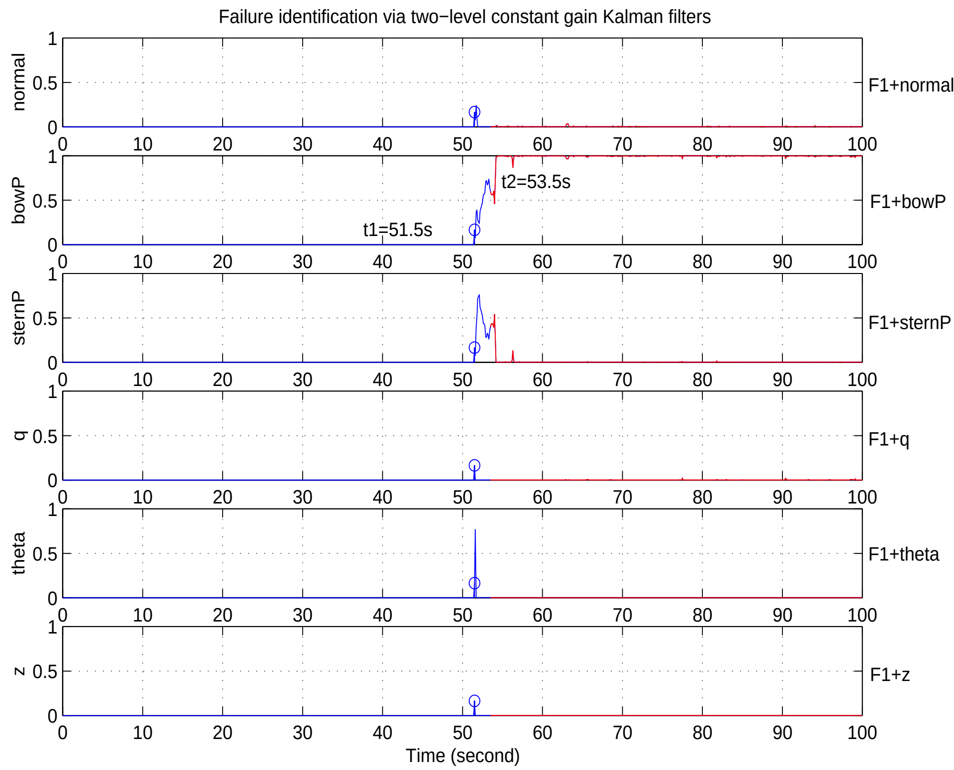

Figure 4.19: Single-Fault Identification in Diving Subsystem

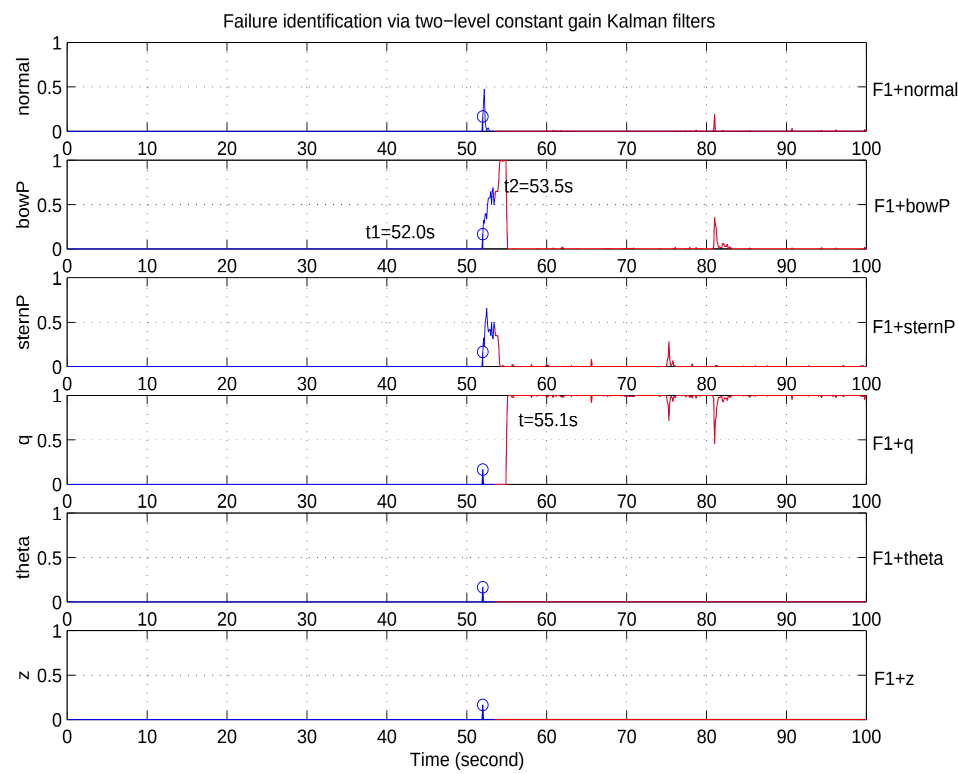

Figure 4.20: Dual-Fault Identification in Diving Subsystem

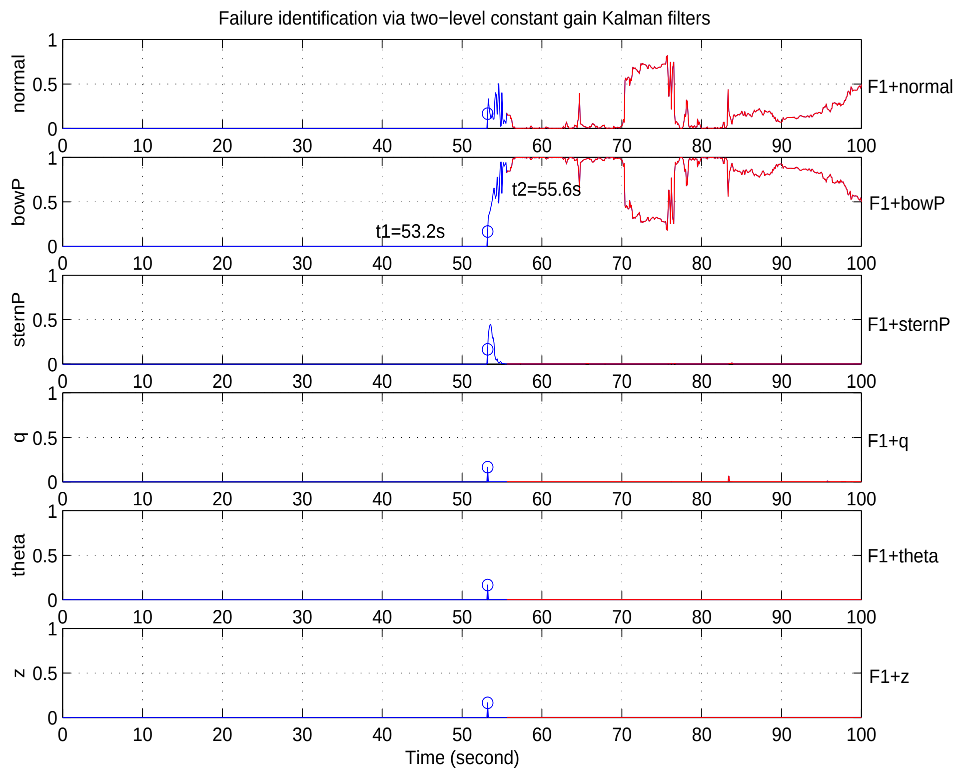

Figure 4.21: Partial-Failure Identification in Diving Subsystem

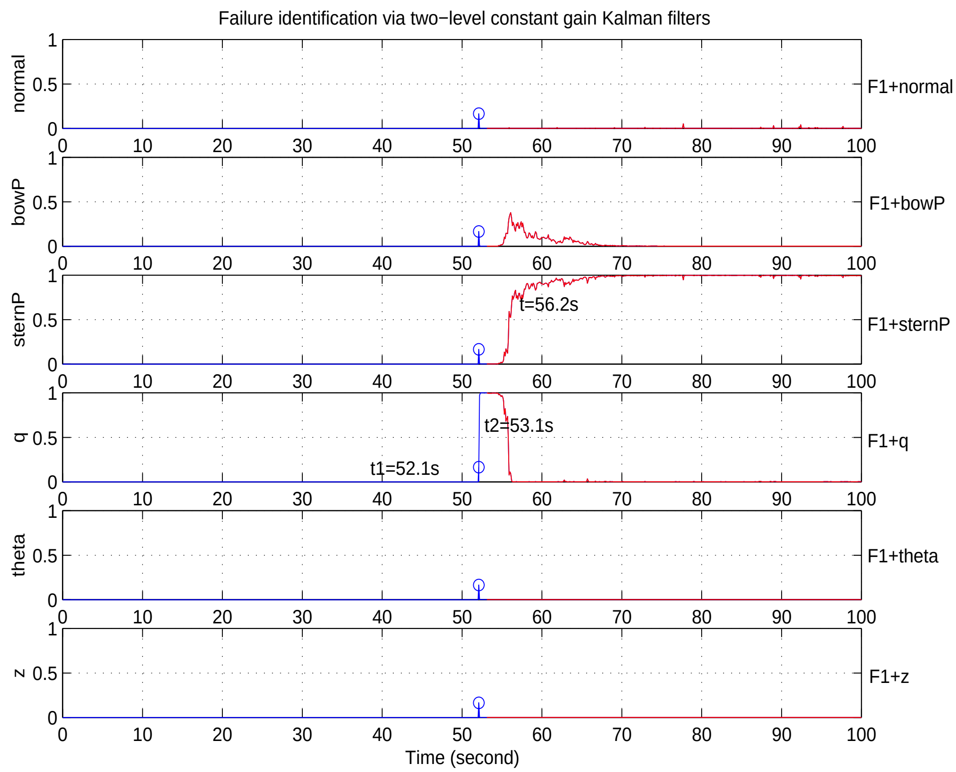

Figure 4.22: Simultaneous-Failure Identification in Diving Subsystem

Chapter 5

Control Reconfiguration Based on HFDI

5.1 Introduction

As we mentioned earlier in Chapter One, when failures occur to UUVs and result in abnormal operations, the only solution is to abort the mission due to lack of fault tolerance. For UUVs to continue operating acceptably following failures in actuators and/or sensors, the control system must be able to autonomously detect, identify, and modify the control law to accommodate failures. We have developed a hierarchical fault detection and identification methodology in previous chapters. Now we begin to consider fault-compensation measures.

We have been applying the sliding-mode control (SMC) to UUVs for fault detection and identification. It is a robust control strategy that retains satisfactory performance in the presence of variations from the model without changes in the control system's structure or parameters. However, a fixed control structure only accommodates a restricted degree of variation. In this chapter, we will investigate control reconfiguration based on the result of HFDI. Specifically, we will alter control system's parameters in response to the fault-identification results.

This chapter is organized as follows. First, we talk about the structure of the fault-tolerant control system. Then, we discuss the adaptive multiple-model sliding-mode control (MMSMC) for control reconfiguration. Finally, we simulate the fault-tolerant control in the steering subsystem of NPS UUVs, and summarize the results.

5.2 Fault-Tolerant Control System

Figure 5.1 illustrates an architecture for fault-tolerant control [1]:

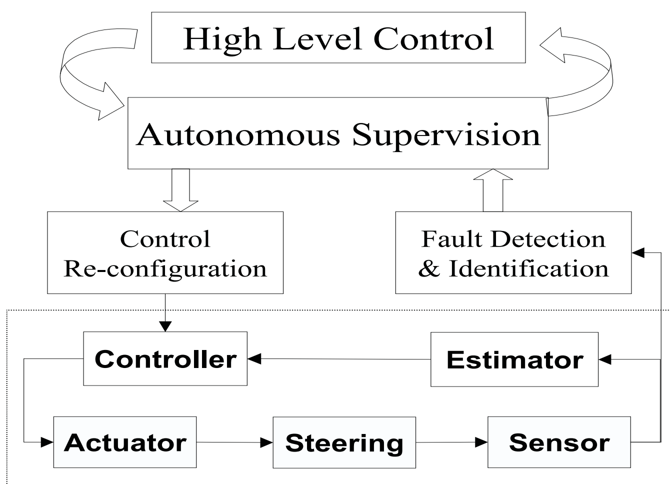

Figure 5.1: Fault-Tolerant Control Structure

The structure consists of a low-level control, an autonomous supervision level, and a high-level control. The low-level control consists of actuators, sensors, a controller and an estimator. It is responsible for controlling the vehicle maneuver according to commands. The autonomous supervision level is to implement fault detection and identification, and control reconfiguration. The high-level control involves strategy control and human interventions. High-level control is important in the control of unmanned underwater vehicles for complex missions, but it is beyond the consideration of this work. We achieve the fault-tolerant control by incorporating the function of autonomous supervision into the low-level control.

5.3 Control Reconfiguration with Multiple Sliding-Mode Controllers

Among different ways of control reconfiguration described in the literature review, it is natural to choose the multiple-model adaptive control (MMAC), since we have been using the multiple-model adaptive estimation (MMAE) for FDI. Based on the MMAE, we develop a method of multiple-model sliding-mode control (MMSMC), in which we design sliding-mode controllers for each hypothesized failure model, and synthesize the control signal according to the *a posteriori* probability of each hypothesis. It is different from Cheng and Leonard's UUV fin failure compensation approach [32] and Yang, Yuh, and Choi's thruster failure accommodation method [33] in that, they deal with failures in the actuator control matrix by mapping the control signal to the remaining physical actuators, instead of in the control-law design.

Figure 5.2 illustrates the MMSMC algorithm.

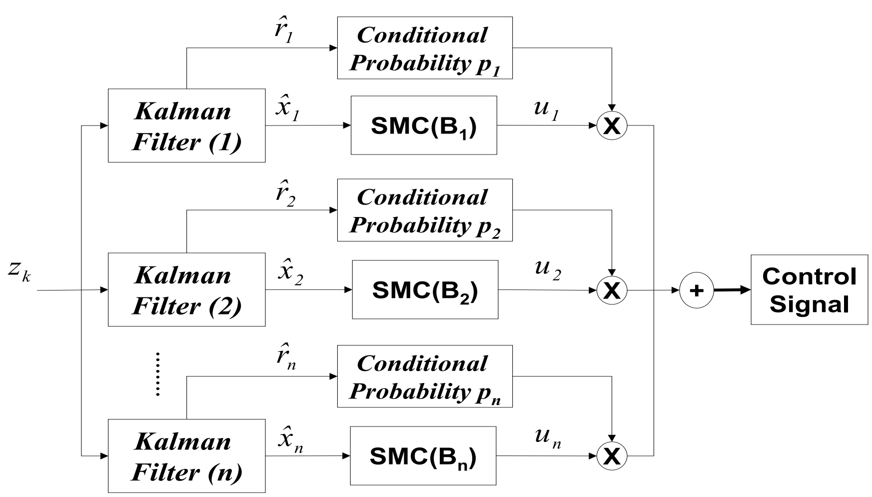

Figure 5.2: Multiple-Model Sliding-Mode Control

Basically, the MMSMC algorithm comprises a bank of parallel Kalman filters and the same number of sliding-mode controllers, corresponding to a set of hypothesized failure models. We apply a hypothesis conditional-probability computation to calculate the *a posteriori* probability for each model, using the residual from each Kalman filter. Suppose that θ_i represents an uncertain parameter from the hypothesized set $[\theta_1, \theta_2, ..., \theta_n]$, that depicts the failure status of sensors and actuators. According to the MMAE, we can calculate the hypothesis conditional probability for the assumption $\theta = \theta_i$ $(i = 1, 2, ..., n)$ up to the kth sampling interval from Eq.(4.19). We express it here again:

$$p(\theta_i|Z_k) = \frac{|\Omega_{k|\theta_i}|^{-1/2}\exp(-\frac{1}{2}\tilde{z}^T_{k|\theta_i}\Omega^{-1}_{k|\theta_i}\tilde{z}_{k|\theta_i})p(\theta_i|Z_{k-1})}{\sum_{i=1}^{n}|\Omega_{k|\theta_i}|^{-1/2}\exp(-\frac{1}{2}\tilde{z}^T_{k|\theta_i}\Omega^{-1}_{k|\theta_i}\tilde{z}_{k|\theta_i})p(\theta_i|Z_{k-1})} \tag{5.1}$$

$$\Omega_{k|\theta_i} = H_i P_i(k) H_i^T + R_k^i \tag{5.2}$$

where $Z_k = [z_1, z_2, ..., z_k]$ is the observation history up to time k, and $\tilde{z}_{k|\theta_i}$ is the residual, or innovation represented by $\hat{r}_i$ in Figure 5.2.

If θ_i is the actual operating mode, then for a single failure scenario, the *a posteriori* probability of that mode tends to approach one, while all the other probabilities go to zero. These probabilities denote how correct each hypothesis is. The MMAE combines the filter state estimations, with their *a posteriori* probabilities as weighting coefficients. In the HFDI found in Chapter 4, we design a sliding-mode controller based on this combined state estimation. With MMSMC, we design n sliding-mode controllers associating with each state estimation, and sum over all the elemental sliding-mode control signals weighted by their probabilities. In other words, we reconfigure the control with a probability-weighted average of the elemental control signals, instead of using a single sliding-mode controller at the end. Equation (5.3) illustrates the summation.

$$u_{MMSMC}(k) = \sum_{i=1}^{n} u_i p(\theta_i|Z_k) \tag{5.3}$$

where u_i is the elemental control law based on the corresponding state estimate, $p(\theta_i|Z_k)$ is the *a posteriori* probability serving as the weighting coefficient. If $p(\theta_i|Z_k)$ is very high, then

u_i will contribute more in the summation. As a result, the reconfigured control law is tailed for $\theta = \theta_i$.

We apply the MMSMC in the third level of HFDI for control reconfiguration, based on the second bank of Kalman filters. After the fault occurrence has been confirmed in the second level by the first bank of Kalman filters, we start to compensate the failure with the MMSMC.

Figure 5.3 shows the structure of the ith elemental sliding-mode controller, designed solely for the ith model.

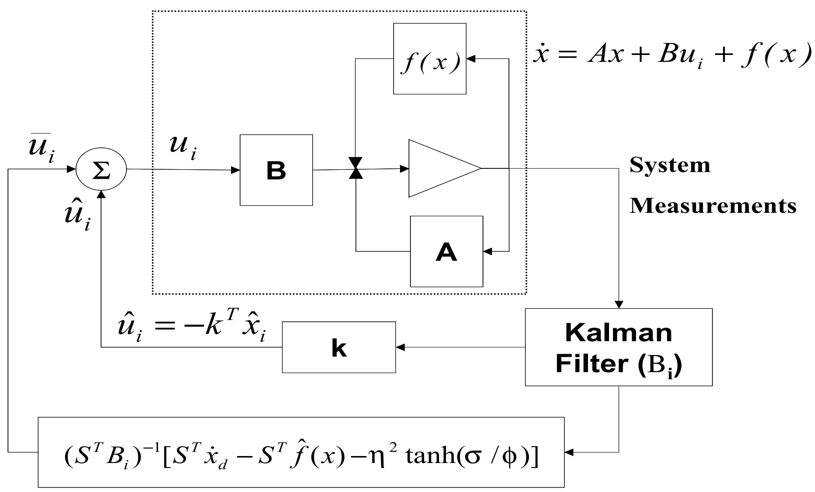

Figure 5.3: Elemental Controller Design

The elemental controller accepts system measurements from the actual system, and drives the actual system with a weighted control law. We design the elemental sliding-mode control law based on the state estimate and the system parameter of the hypothesized fault model. As we described in Chapter 2, it consists of two parts: a linear feedback control law and a switching control law.

$$\begin{aligned} u_i &= \hat{u}_i + \bar{u}_i \\ &= -k^T \hat{x}_i + (S^T B_i)^{-1}[S^T \dot{x}_d - S^T \hat{f}(x) - \eta^2 tanh(\sigma/\phi)] \end{aligned} \tag{5.4}$$

The re-configured control signal in Eq.(5.3) now becomes:

$$\begin{aligned} u_{MMSMC}(k) &= \sum_{i=1}^{n} u_i p(\theta_i|Z_k) \\ &= \sum_{i=1}^{n} [\hat{u}_i + \bar{u}_i] p(\theta_i|Z_k) \end{aligned} \tag{5.5}$$

Alternatively, we can use the maximum elemental control as the output of control reconfiguration if it is dominant.

$$u_{MMSMC}(k) = [u_j, j|p_j = \max_{1\leq i\leq n} p_i] \tag{5.6}$$

5.4 Algorithm Performance

This section applies the fault-tolerant control to the subsystems of NPS UUVs. We use the same simulation setting for fault detection and identification as in Chapter 3 and 4. As for MMSMC, we design sliding-mode controllers relating to each fault model, including the full functional operation. The controllers work in parallel and constitute the re-configurable control. We use the steering subsystem as an example, since it is easier to exemplify the compensation results with the steering trajectory of the vehicle.

In Figures 5.4, 5.5, 5.6, and 5.7, we simulate a total yaw-rate-sensor failure at $t = 25s$ followed by a total upper-stern-rudder failure at $t = 30s$ in the steering subsystem. The results consist of two parts: fault detection and identification results in Figures 5.4 and 5.5, and compensation results in Figures 5.6, and 5.7. We interpret the fault detection and identification results in the same way as we did in Chapter 3 and 4. In Figure 5.4, the SNR begins to increase after $t = 25s$. According to the threshold decision logic, the algorithm detects the yaw-rate-sensor failure at $t = 27.2s$. There are seven filters in the first bank corresponding to models of the normal operation mode, and total failures of the upper-bow rudder, the lower-bow rudder, the upper-stern rudder, the lower-stern rudder, the yaw-rate sensor, and the heading-angle sensor. The *a posteriori* probability of the yaw-rate-sensor

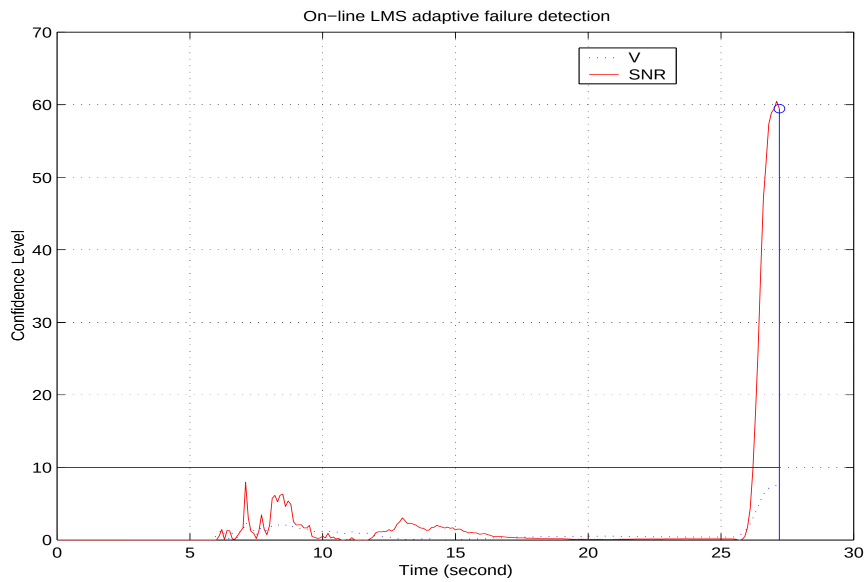

Figure 5.4: Fault Detection in FTC System

failure is close to one, while others are close to zero. The algorithm identifies the single yaw-rate-sensor failure in the first bank of Kalman filters at $t = 29.2s$.

The second bank of Kalman filters assumes the occurrence of the yaw-rate-sensor failure, and still identifies the single failure until the occurrence of the rudder failure. After $t = 30s$, the algorithm identifies the dual failure of the yaw-rate sensor and stern rudders. The upper-stern and the lower-stern rudders take equal probabilities in bank two because of the same modeling. The yaw-rate-sensor failure appears dominant in the dual failure at some of time.

The algorithm begins to perform the MMSMC in the third level of HFDI. Figures 5.7 and 5.6 illustrate results with control reconfiguration. Figure 5.6 shows the rudder control signal and the heading angle estimation, and Figure 5.7 shows the steering track of the vehicle. In both figures, we compare the compensated situation and the failure situation, using the normal situation as a reference. In Figure 5.6, the rudder control signal is different from the normal situation due to the occurrence of failures. As a result, the heading angle deteri-

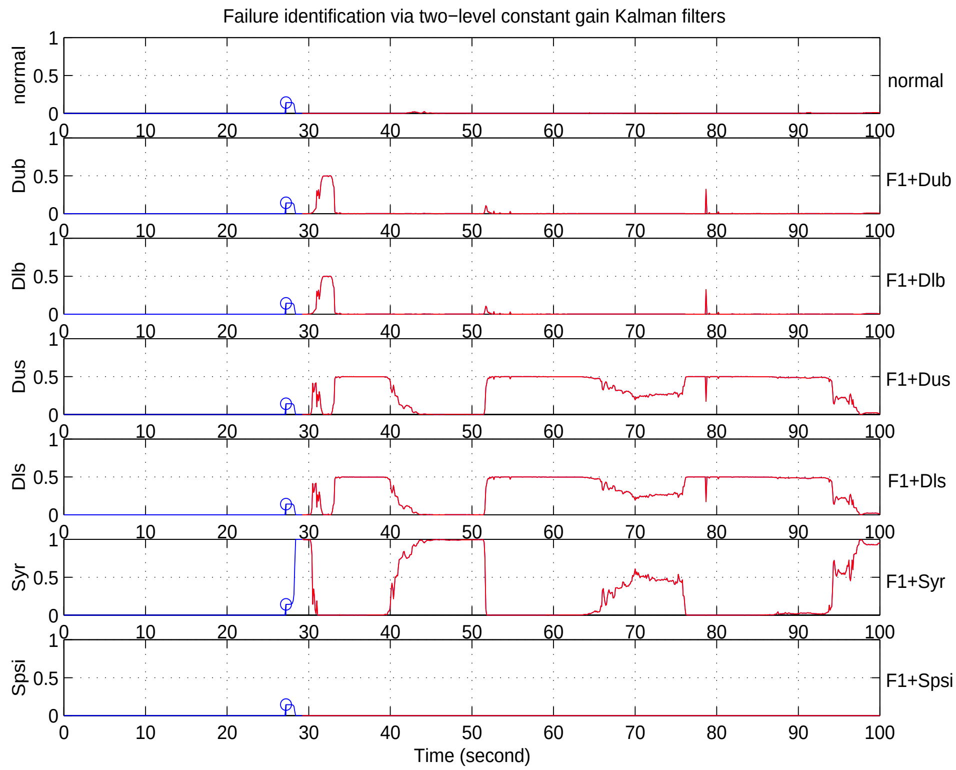

Figure 5.5: Fault Identification

orates, and the steering track deviates from the normal track. With control reconfiguration, the rudder control signal becomes larger than normal to compensate for the failure. We can see that the heading angle and the trajectory of the vehicle almost come back to the normal situation after control compensation.

From Figures 5.8 to 5.11, we simulate a total upper-stern-rudder failure at $t = 25s$ followed by a total lower-stern-rudder failure at $t = 30s$. Figures 5.8 and 5.9 show that the algorithm detects the upper-stern-rudder failure at $t = 27.5s$ and identifies it at $t = 28.9s$ before the occurrence of the second failure at $t = 30s$. In bank two Kalman filters, the algorithm identifies the dual failure of both stern rudders besides some probability drifts.

Again, we compare the compensated rudder control, heading angle and steering trajectory with their failure situations, with respect to normal ones. From Figures 5.10 and 5.11,

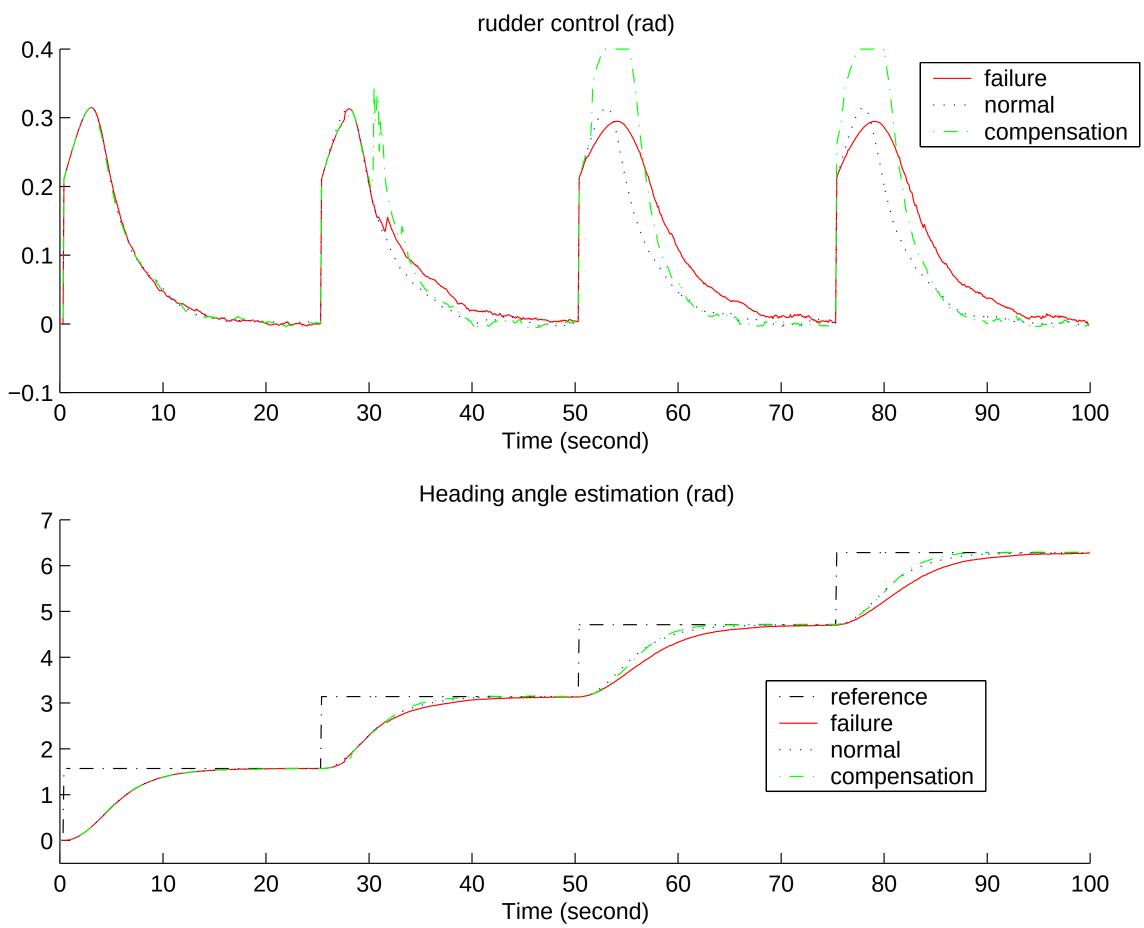

Figure 5.6: Rudder Control and Heading Angle

we see that the heading angle and the trajectory greatly degrade due to the dual stern-rudder failure. After control compensation, the rudder control becomes saturated, and drives the heading and the steering trajectory towards their normal values. Although it still deviates from the normal path, the trajectory has been greatly improved comparing to that without control compensation.

Comparing the previous two examples, we find sensor failures are relatively easier to compensate than actuator failures. Furthermore, we try to test the control compensation with simultaneous failures.

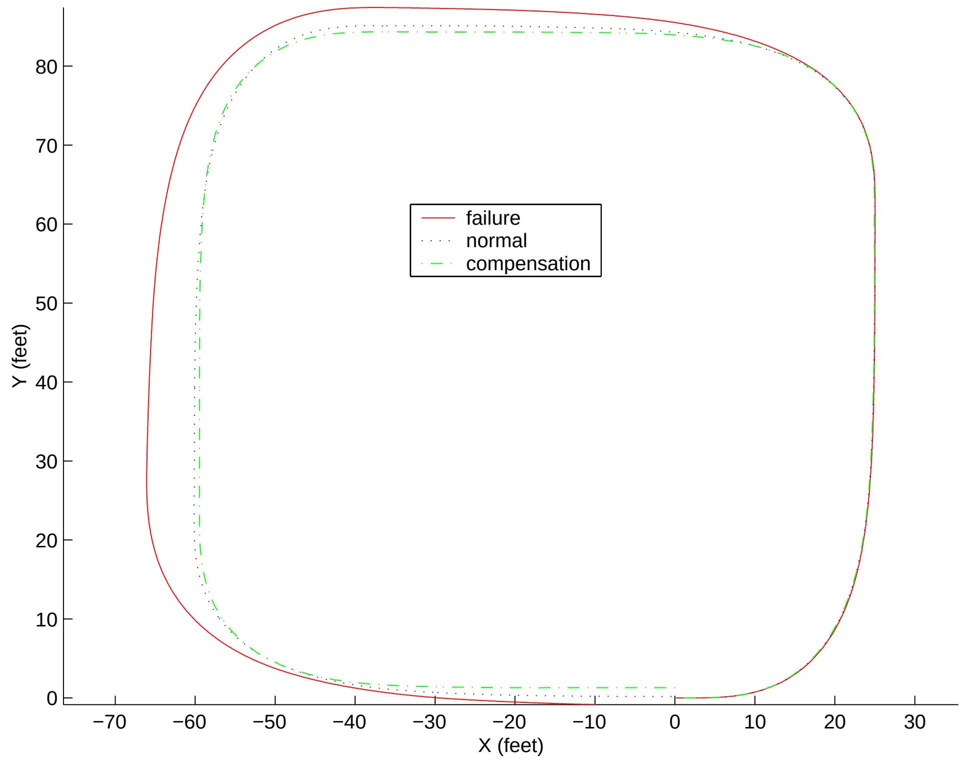

Figure 5.7: Steering Trajectory

We simulate both total stern-rudder failures at $t = 25s$, and apply the fault-tolerant control. Figures 5.12 to 5.14 show the results. The algorithm detects the failure at $t = 27.3s$ in Figure 5.12, and identifies the dual failure at $t = 33.7s$ in Figure 5.13. We have shown in the previous chapter that although the algorithm is not designed for simultaneous failures, it identifies the failure correctly.

Figures 5.14 and 5.15 present the result of fault compensation in this occasion. We see some improvement of the heading angle and trajectory, but compared to stern-rudder failures occurring at different time, the compensation result is not satisfying. The proposed control reconfiguration method is limited in its ability to compensate for simultaneous rudder failures.

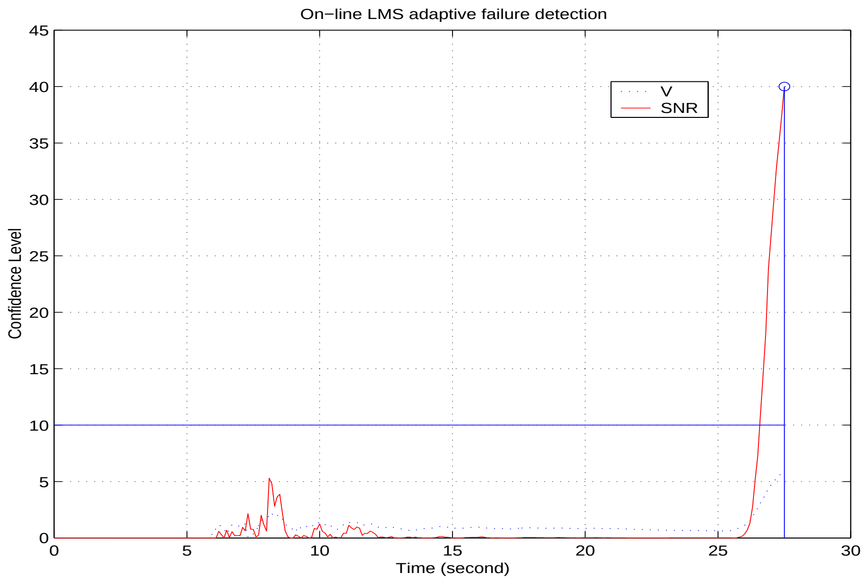

Figure 5.8: Fault Detection

5.5 Summary

This chapter presents a study of the fault-tolerant control of UUVs. First, we identify actuator and sensor failures through the approach of HFDI. Based on the results of HFDI, we develop a re-configurable control with multiple sliding-mode controllers for each of the hypothesized failure modes, and reconfigure the control signal for identified failures with a probability-weighted average of all the elemental control signals. We apply this fault-tolerant control method to the steering subsystem of NPS UUV. We simulate dual-failure situations, such as total failures of a yaw-rate sensor and a stern rudder, total failures of both upper and lower stern rudders. Simulation results illustrate effective compensations in the heading angle and the steering track. Therefore, it is possible to avoid unnecessary mission abortions by control reconfiguration. The results also show limitation of the method in severe failure situations, such as simultaneous rudder failures. In practice, fault-tolerant control is a very complex problem depending on the control system, mechanical system, and many other factors. It is necessary to investigate the compensation measures considering these factors.

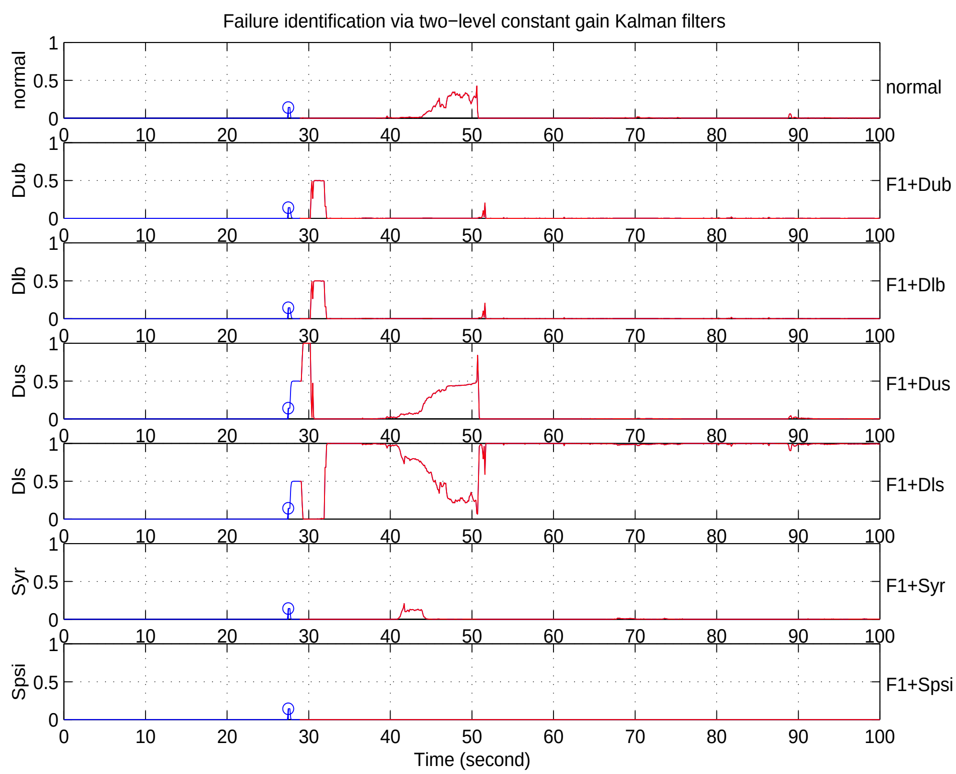

Figure 5.9: Fault Identification

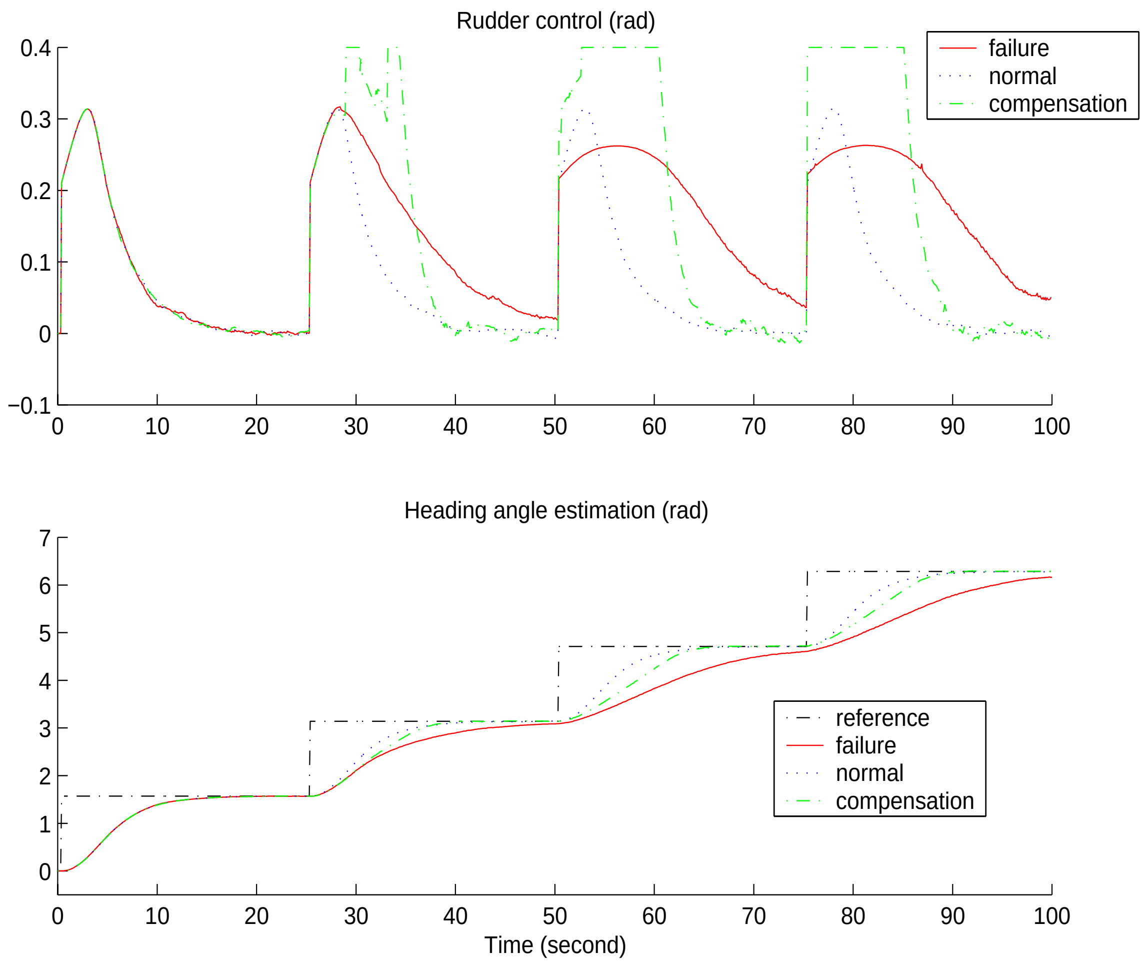

Figure 5.10: Rudder Control and Heading Angle

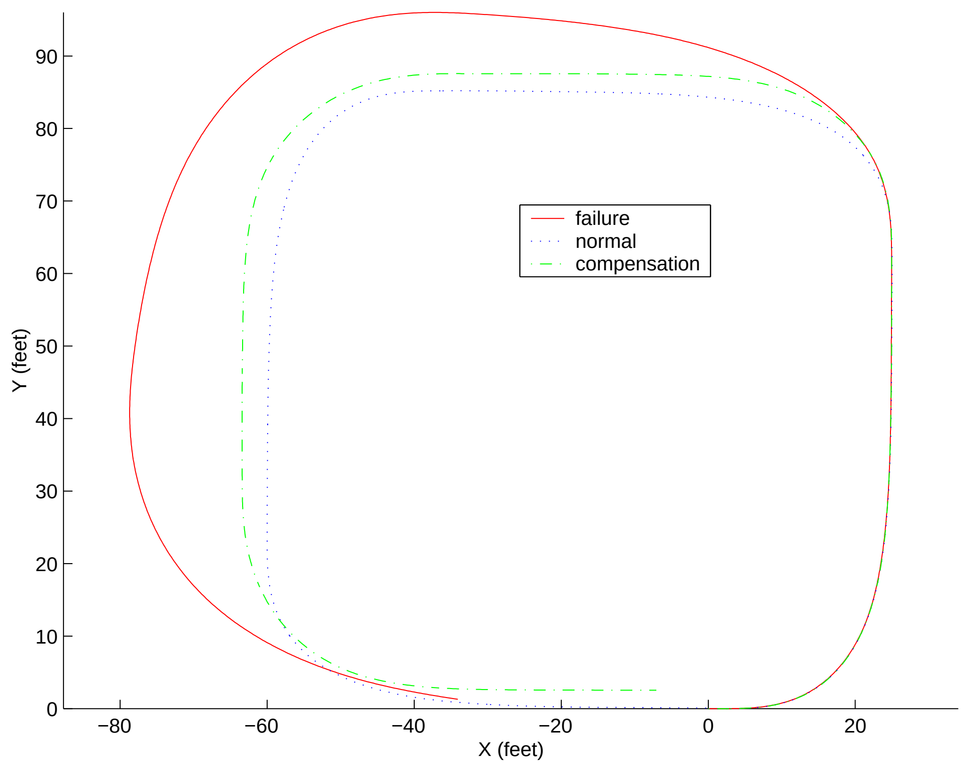

Figure 5.11: Steering Trajectory

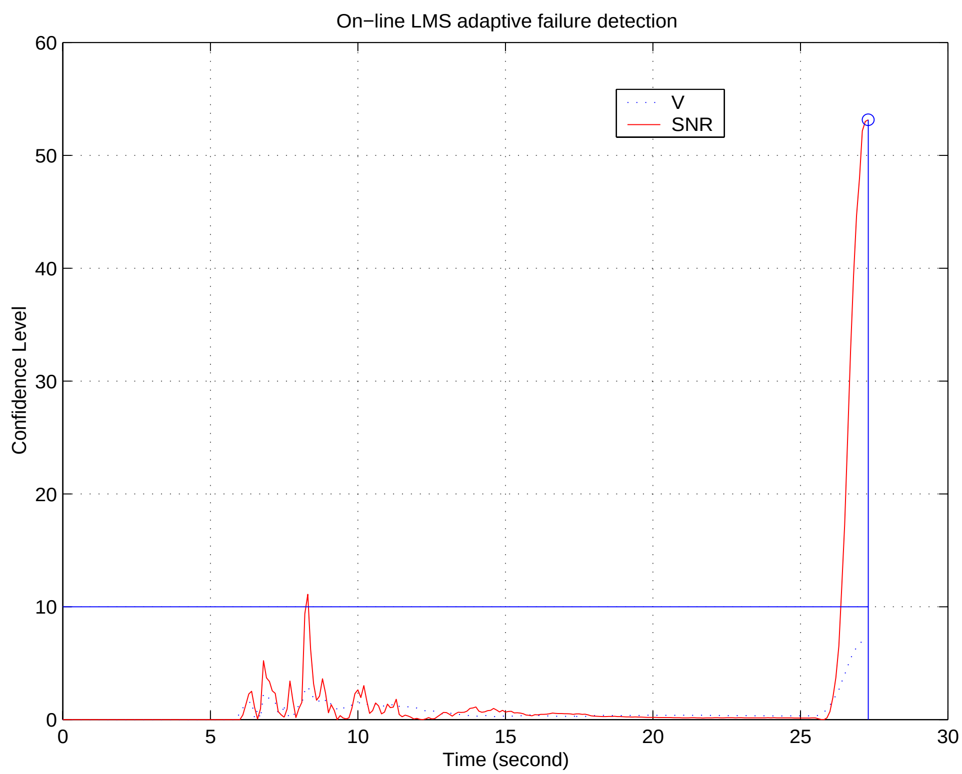

Figure 5.12: Fault Detection

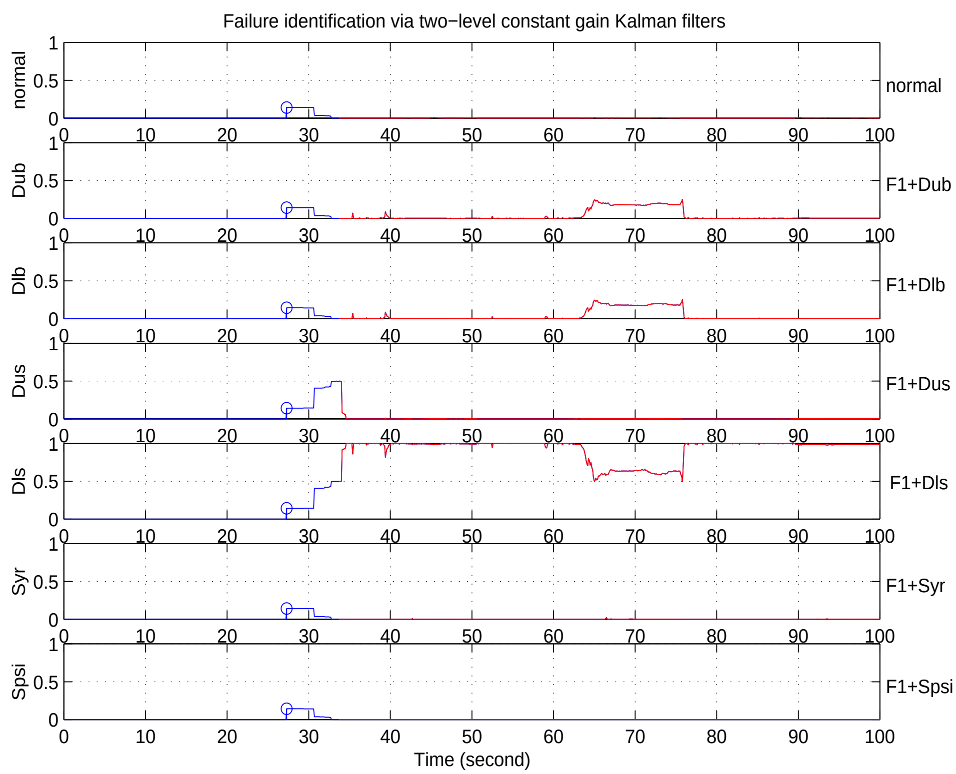

Figure 5.13: Fault Identification

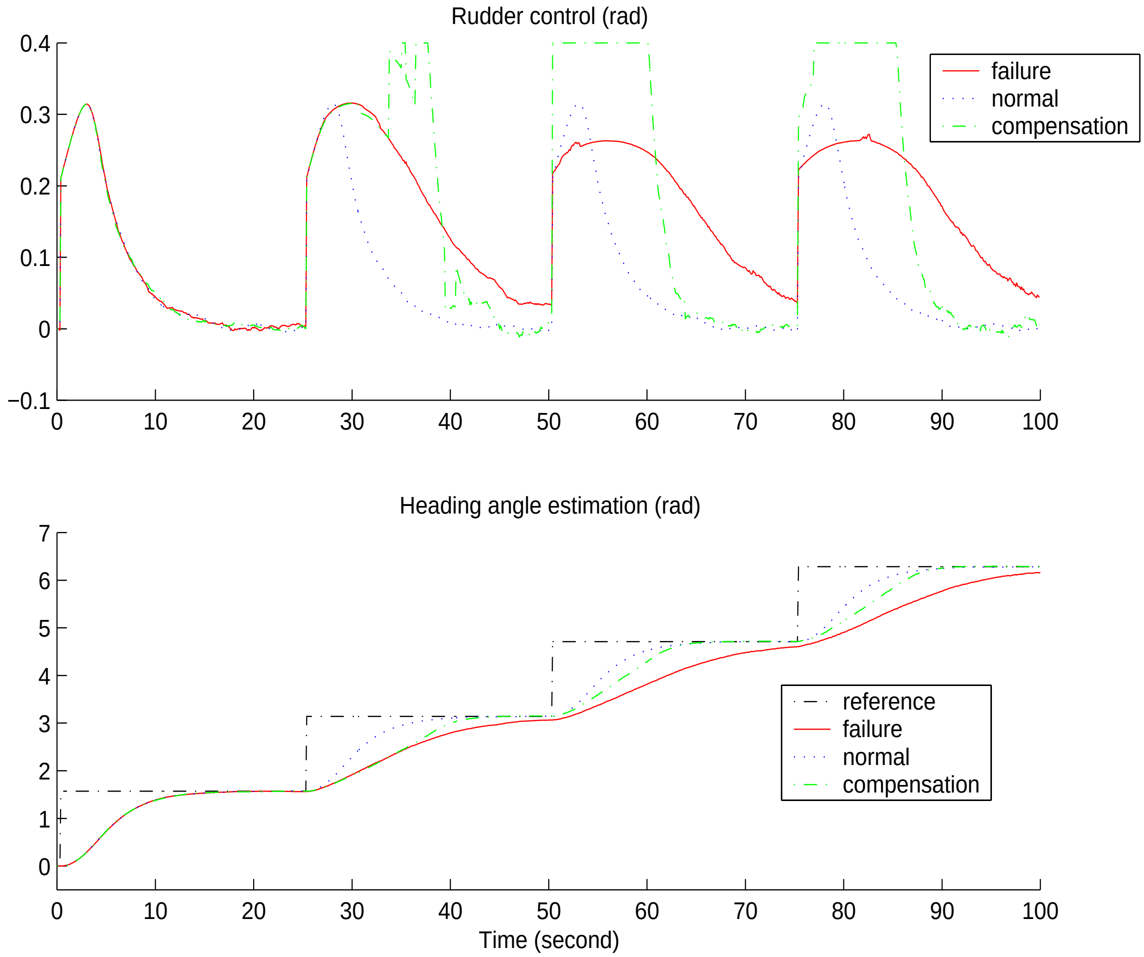

Figure 5.14: Rudder Control and Heading Angle

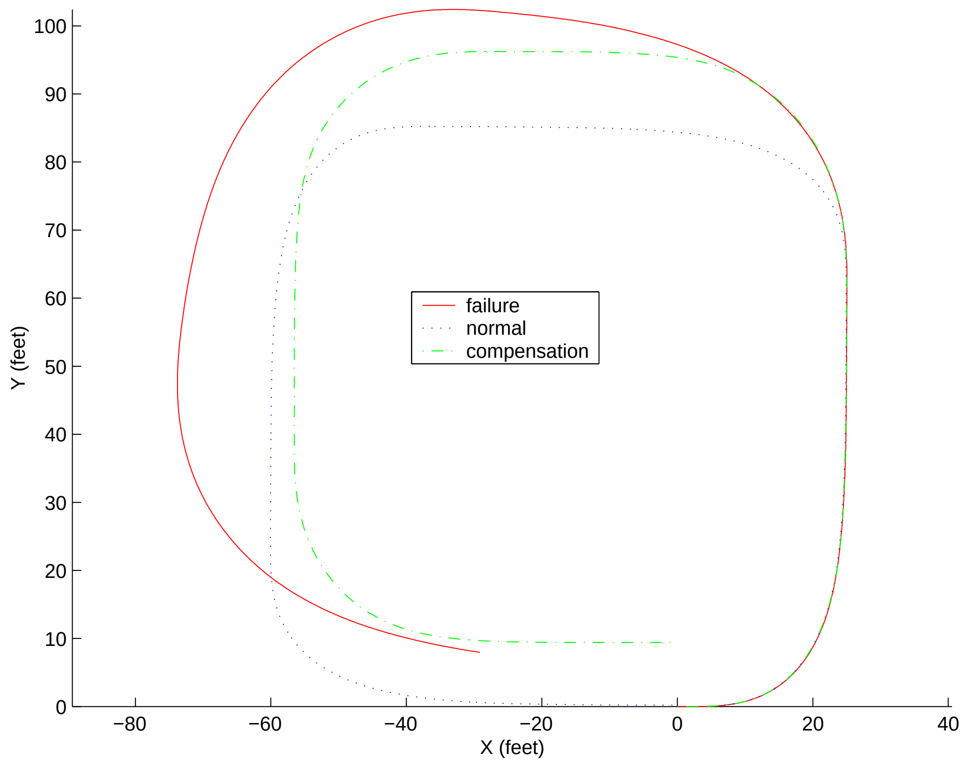

Figure 5.15: Steering Trajectory

Chapter 6

Verification of HFDI with Realistic Data of FAU AUVs

6.1 Introduction

We have proposed a hierarchical methodology of fault detection and identification, and demonstrated its effectiveness with simulated actuator and sensor failures in subsystems of NPS UUVs. In this chapter, we will test our HFDI algorithm with recorded data from at-sea test run of the Florida Atlantic University (FAU) autonomous underwater vehicles (AUVs). The development of HFDI is completely independent of the FAU AUVs and their test run. First, we will briefly introduce FAU AUV subsystem models and the background of the data. Then, we will apply the HFDI to the subsystem of FAU AUVs and show verification results.

6.2 FAU AUV Subsystem Models

The FAU Ocean Explorer (OEX) series AUVs are developed in the Ocean Engineering Department. Generally, we use six degree-of-freedom equations of motion to describe the vehicle's linear and angular velocities with respect to a body-fixed coordinate, and its orientations and positions with respect to the global reference coordinate. In OEX series AUVs, we achieve control of the heading and the pitch through adjustment of the rudder and the stern plane.

6.2.1 Steering Model of FAU AUVs

Equation (6.1) is the model used to describe the steering subsystem of FAU AUVs.

$$\begin{cases} m(\dot{v}_e + r_e u_0) = Y_{\dot{v}_e}\dot{v}_e + Y_{v_e}v_e + Y_{\dot{r}_e}\dot{r}_e + Y_{r_e}r_e + Y_{\delta_{rudder}}\delta_{rudder} \\ \dot{r}_e I_z = N_{\dot{v}_e}\dot{v}_e + N_{v_e}v_e + N_{\dot{r}_e}\dot{r}_e + N_{r_e}r_e + N_{\delta_{rudder}}\delta_{rudder} \end{cases} \tag{6.1}$$

where $\dot{v}_e$ is the side acceleration (lateral acceleration in the body-fixed frame), v_e is the side velocity, $\dot{r}_e$ is the yaw acceleration, r_e is the yaw rate, δ_{rudder} is the angle of the rudder (positive when the angle is so that the vehicle turns left).

The coefficients in Eq. (6.1) have the following values:

$$m = \frac{716.8}{32.2}\frac{1}{2}\frac{1}{\rho l^2 u_0^2} = 0.0188 slug \tag{6.2}$$

$$u_0 = 4.92 feet.s^{-1} \tag{6.3}$$

$$Y_{\dot{v}_e} = -19.01 slug \tag{6.4}$$

$$Y_{v_e} = -23 slug.s^{-1} \tag{6.5}$$

$$Y_{\dot{r}_e} = 3.303 slug.feet \tag{6.6}$$

$$Y_{r_e} = 52.32 slug.feet.s^{-1} \tag{6.7}$$

$$Y_{\delta_{rudder}} = 18.09 slug.feet.s^{-2} \tag{6.8}$$

$$I_z = 57.14 slug.feet^2 \tag{6.9}$$

$$N_{\dot{v}_e} = 3.303 slug.feet \tag{6.10}$$

$$N_{v_e} = -21.36 slug \tag{6.11}$$

$$N_{\dot{r}_e} = -43.23 slug.feet^2 \tag{6.12}$$

$$N_{r_e} = -202.55 slug.feet^2.s^{-1} \quad (6.13)$$

$$N_{\delta_{rudder}} = -55.87 slug.feet^2.s^{-2} \quad (6.14)$$

Equation (6.15) expresses the state-space model of the steering subsystem including the state of heading:

$$\begin{bmatrix} \dot{v}_e \\ \dot{r}_e \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} -1.2528 & 2.4081 & 0 \\ -0.2540 & -1.9388 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} v_e \\ r_e \\ \psi \end{bmatrix} + \begin{bmatrix} 0.8589 \\ -0.5284 \\ 0 \end{bmatrix} \delta_{rudder} \quad (6.15)$$

The steering subsystem has state variables of the sway velocity, the yaw rate, and the heading angle, and sensors for the yaw rate and the heading angle. It is controlled by the rudder deflection.

6.2.2 Diving Model of FAU AUVs

In the diving model, we only consider angular velocity in pitch q_e and pitch angle θ_e, while assuming that the heave velocity and its derivative are zero, that is, $w_e = \dot{w}_e = 0$. This results in a diving model in Eq. 6.16.

$$I_y \dot{q}_e = M_{\dot{q}_e} \dot{q}_e + M_{q_e} q_e + M_{\delta_{sternplane}} \delta_{sternplane} + Z_B B \theta_e \quad (6.16)$$

Since we are only dealing with small angles, we have $\dot{\theta}_e \approx q_e$. It leads to:

$$I_y \ddot{\theta}_e = M_{\dot{q}_e} \ddot{\theta}_e + M_{q_e} \dot{\theta}_e + M_{\delta_{sternplane}} \delta_{sternplane} + Z_B B \theta_e \quad (6.17)$$

where $\ddot{\theta}_e$ is the pitch acceleration of the vehicle, $\dot{\theta}_e$ is the pitch rate of the vehicle, θ_e is the pitch of the vehicle, $\delta_{sternplane}$ is the angle of the stern plane (negative upward).

The coefficients in Eq. (6.16) have the following values.

$$I_y = 57.14 slug.feet^2 \quad (6.18)$$

$$M_{\dot{q}_e} = -43.15 slug.feet^2 \quad (6.19)$$

$$B = 716 lbf \quad (6.20)$$

$$Z_B = -0.0762 feet \quad (6.21)$$

$$M_{q_e} = -8.41 slug.feet^2.s^{-1} \quad (6.22)$$

$$M_{\delta_{sternplane}} = -70.35 slug.feet^2.s^{-2} \quad (6.23)$$

Considering the state of depth and substituting coefficient values into Eq. 6.16, we have a diving model in the form of:

$$\begin{bmatrix} \dot{q}_e \\ \dot{\theta}_e \\ \dot{z} \end{bmatrix} = \begin{bmatrix} -10.0509 & -0.5438 & 0 \\ 1 & 0 & 0 \\ 0 & -4.92 & 0 \end{bmatrix} \begin{bmatrix} q_e \\ \theta_e \\ z \end{bmatrix} + \begin{bmatrix} -0.7015 \\ 0 \\ 0 \end{bmatrix} \delta_{sternplane} \quad (6.24)$$

The diving subsystem has sensors for the pitch rate and the pitch angle, and it is controlled by the stern plane.

To apply the HFDI to FAU AUV subsystems, we need to model failures of actuators and sensors. In spite of the difference in the vehicle design, failures occurring to actuators and sensors basically have the same characteristics. We use the same technique of fault modeling as in Chapter 3. In other words, we model actuator failures as additive parameter changes in the control input matrix, and sensor failures as additive parameter changes in the output equation of the system.

6.3 Background of FAU Data

Before we test our HFDI with FAU AUV real data, we review the background of the data. The Ocean Department tests their AUVs at sea many times through the year. Taking year 1999 as an example, they recorded about 257 sets of data in logger files. We need to extract logger files to obtain data. Each set of data consists of hundreds of variables. We are only concerned about those variables describing AUV motions in the steering and diving subsystems, such as the AUV heading, the yaw rate, the rudder command, the pitch angle, the pitch rate, and the stern-plane command. Figure 6.1 is an example plot of the AUV heading, the AUV motion in yaw rate, and the rudder command of the steering subsystem.

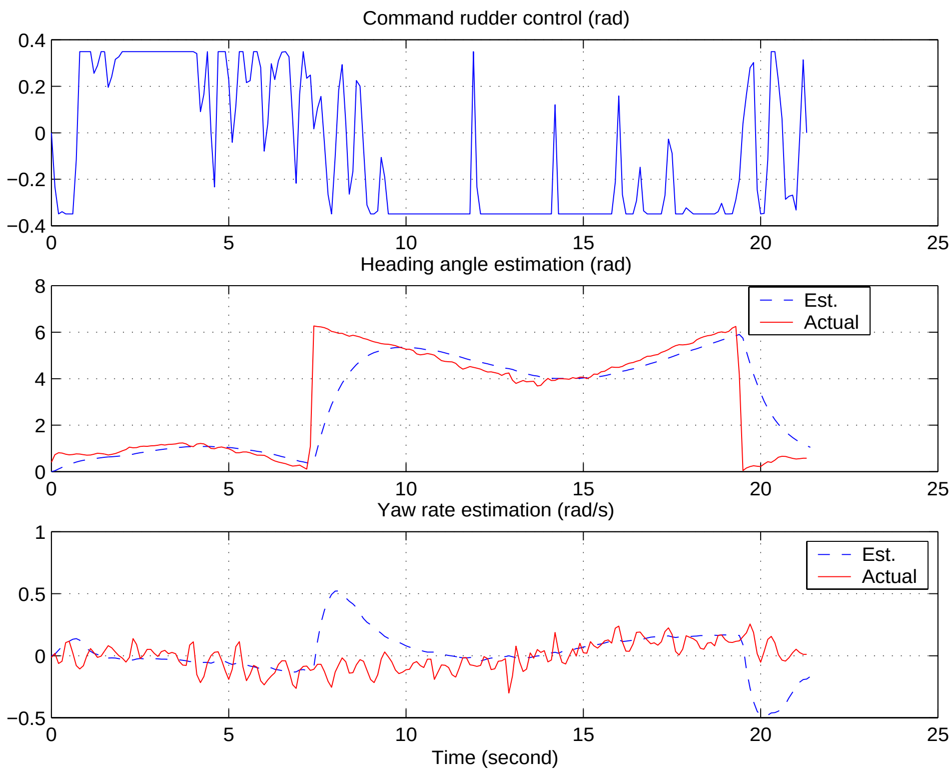

Figure 6.1: Example Data of AUV Steering Motions

The rudder command and the heading angle are recorded in "degree", and the yaw rate is in "degree/second". We transform all the units from "degree" to "radian" for the convenience of calculation. The sampling interval of the data is 1/60 second. To reduce noise influence, we resample it by taking an average of every six samples of data. This results in a new set of data at a sampling rate of 1/10 second.

We find in the data that when the heading angle decreases to 0 degree, it becomes 360 degrees and continues to decrease from that. Similarly, when the heading angle increases to

360 degrees, it becomes 0 degree, and then continues to increase from that. In other words, the data is "wrapped". As a result of this, we see some jumping in the figure of the heading angle.

From the data plot, we can see how the heading angle changes corresponding to the rudder command. In terms of the trend, when the rudder command increases towards the positive limit ($+0.4$ *rad*), the heading angle increases with it; when the command decreases towards the negative limit (-0.4 *rad*), the heading angle decreases also. Generally speaking, if there is no failure in the operation, then the heading angle, the yaw rate, and the rudder command should correspond to each other. However, it is difficult to see the discord from the data directly. With the help of the HFDI, we can check to see if there is any failure or not, and what are those failures.

6.4 Verification Results of HFDI

The OEX AUV is operated with fuzzy logic control. Currently, there is no *a priori* knowledge about the occurrence of the failure, and there is no measure to tell whether the data contain a failure or not. Usually, when commanders observe the failure occurring to the vehicle, they terminate the test mission and order the vehicle to surface.

For fault detection and identification, we use the actual control signal and measurements in the adaptive FIR modeling and state estimation. We set the number of FIR coefficients as $N = 60$ for the steering subsystem, and $N = 250$ for the diving subsystem, since they cover the most significant part of the impulse response, respectively. Figure 6.2 shows the impulse response of the steering subsystem, and 6.3 shows the impulse response of the diving subsystem and its zoom in picture.

We keep other parameters in the adaptive modeling and analysis process the same as we used previously. We design four parallel Kalman filters in the first bank based on hypothesized models of total actuator and sensor failures, and four Kalman filters in the

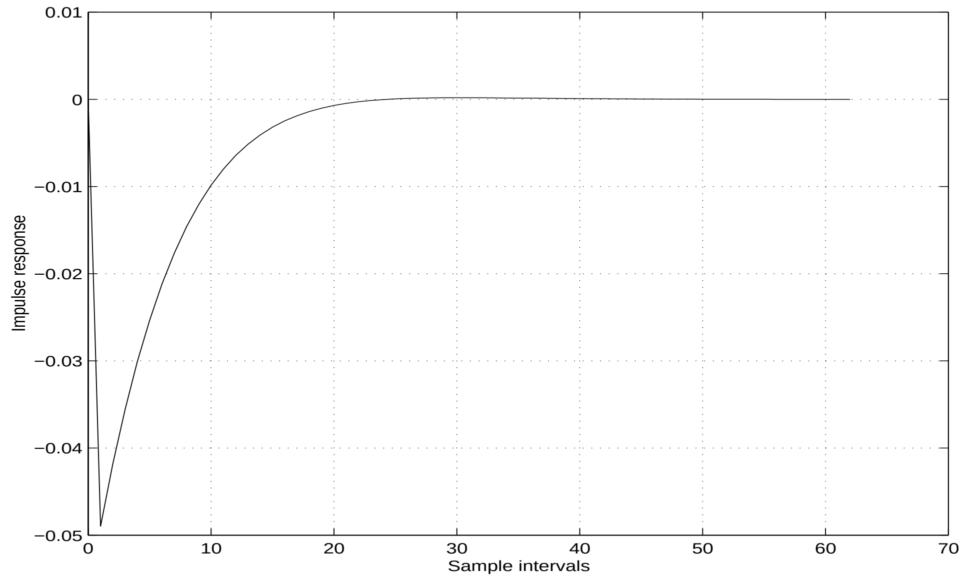

Figure 6.2: Impulse Response of the FAU AUV Steering Subsystem

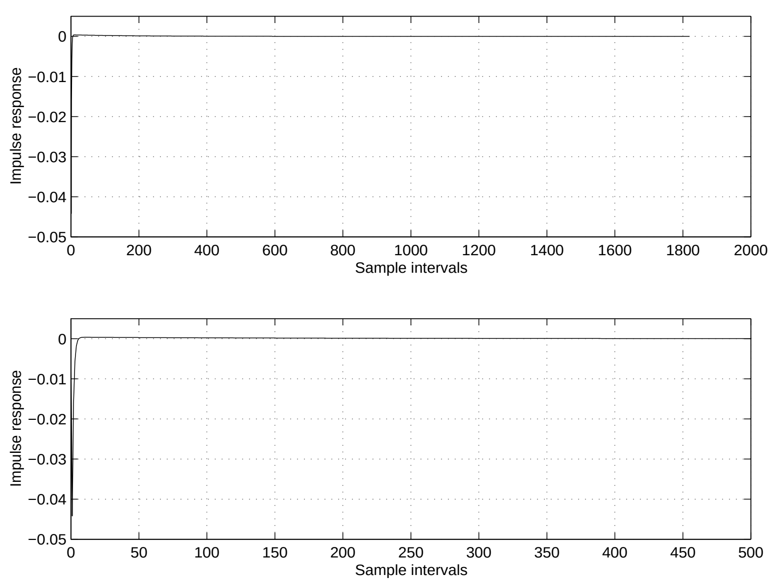

Figure 6.3: Impulse Response of the FAU AUV Diving Subsystem

second bank based on dual-failure models depending on the identification result of the first bank.

We apply the HFDI to the steering subsystem first. Figure 6.4 shows AUV motions of the heading and the yaw rate, and their estimations corresponding to the rudder command in data set number one. We do not have the exact date when the data are recorded, however, according to the FAU staff a failure was observed during the test run.

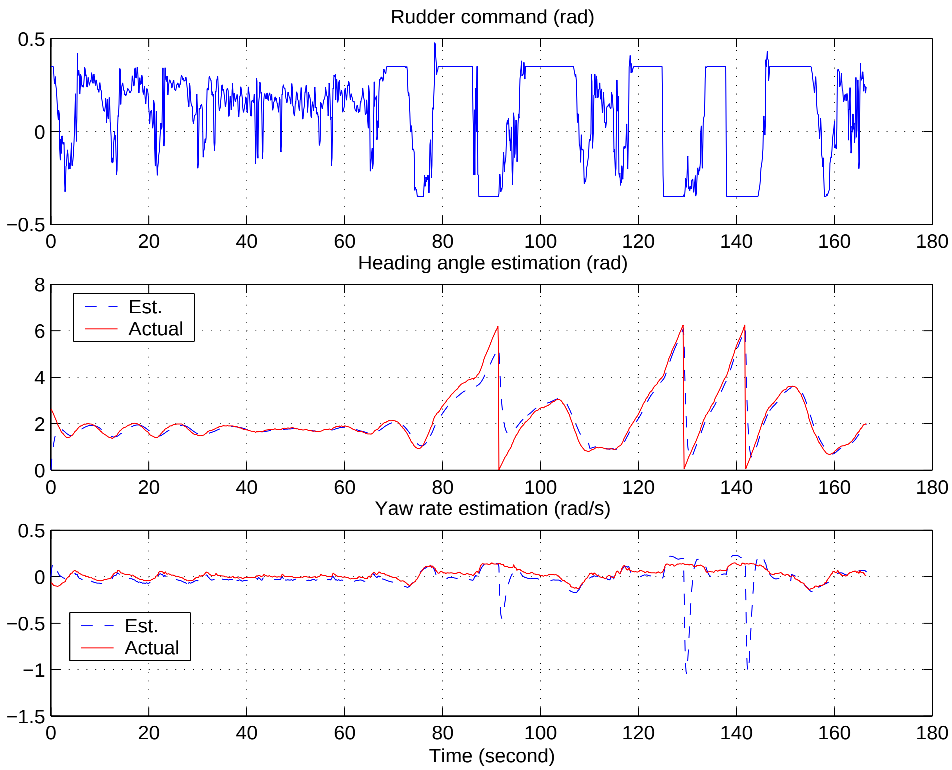

Figure 6.4: AUV Motions in Data No. 1

From Figure 6.4, we see discontinuous heading angles around $t = 90s, 130s$, and $140s$. The jump is because when the heading angle reaches 2π, it actually returns to the zero-degree position. Figure 6.5 shows the steering trajectory of the vehicle according to its heading angle.

The vehicle heading is around 106 degrees first. It increases to 2π at about $t = 90s$. After passing the zero-degree position, it continues to increase until π. Then, the heading decreases to $\pi/3$ around $t = 110s$. The vehicle starts turning from $t = 120s$.

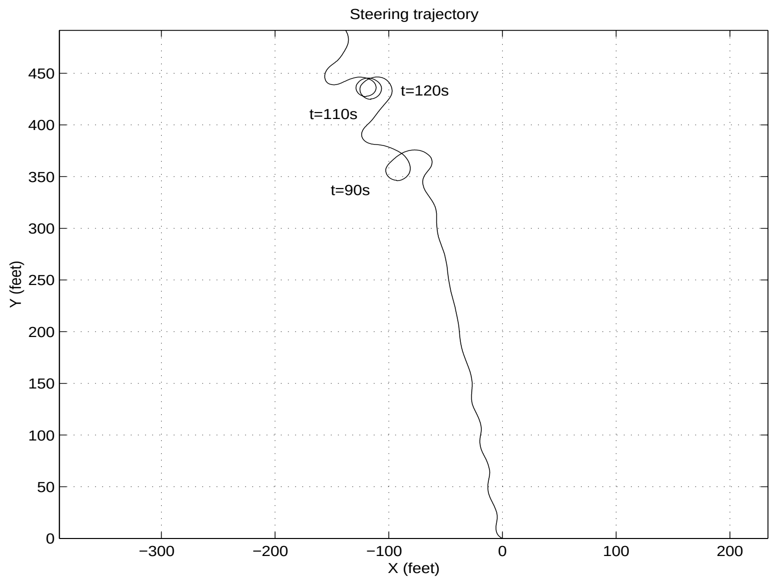

Figure 6.5: Steering Trajectory of Data No. 1

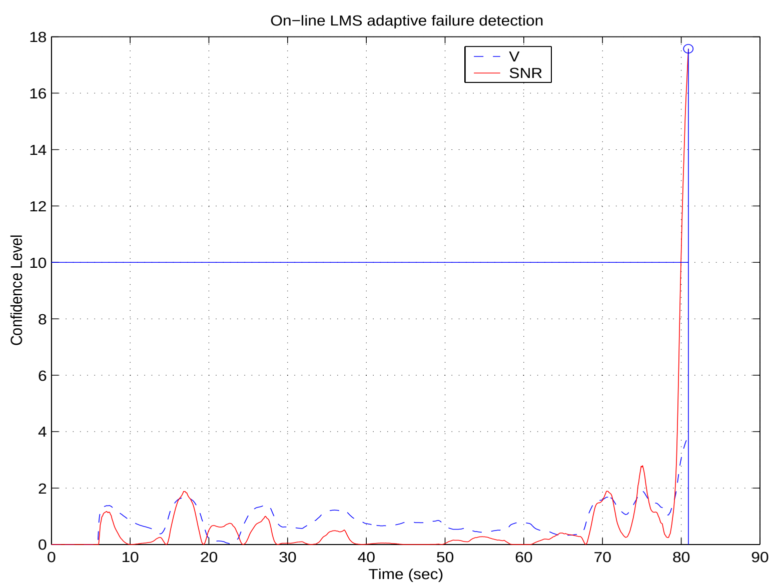

Figure 6.6: Fault Detection in Steering with Data No. 1

Figures 6.6 and 6.7 show FDI results with data number one. The HFDI detects a failure at $t = 80.9s$. In Figure 6.7, the labels on the left represent the normal operation

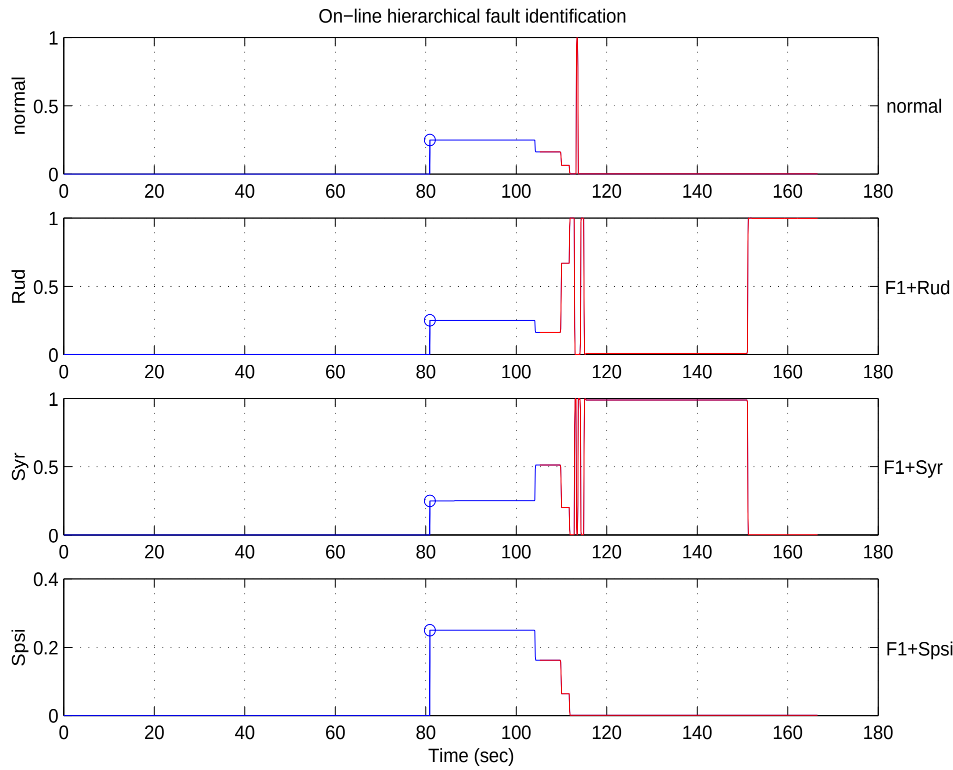

Figure 6.7: Fault Identification in Steering with Data No. 1

mode, the single rudder failure, the single yaw-rate-sensor failure, and the single heading-angle-sensor failure. They explain the result of the first bank of Kalman filters (blue curves). The labels on the right are for bank-two Kalman filters. They represent dual failures of the first identified failure (F1) plus four single scenarios. Here, F1 is identified as a single yaw-rate-sensor failure. The second bank of Kalman filters confirms the identification, except some identification of the dual failure of yaw-rate sensor and rudder at the beginning and at the end. The result suggests a total yaw-rate-sensor failure.

In data set number two, which is recorded on Feb. 3, 1999, the heading is first varying around the position of $0(2\pi)$. Then, the heading angle increases and stays around π. Figure 6.8 shows the heading angle and the yaw rate changing with the rudder command. Figure 6.9 compares the output from the FIR filter with the actual yaw rate, shows the square error

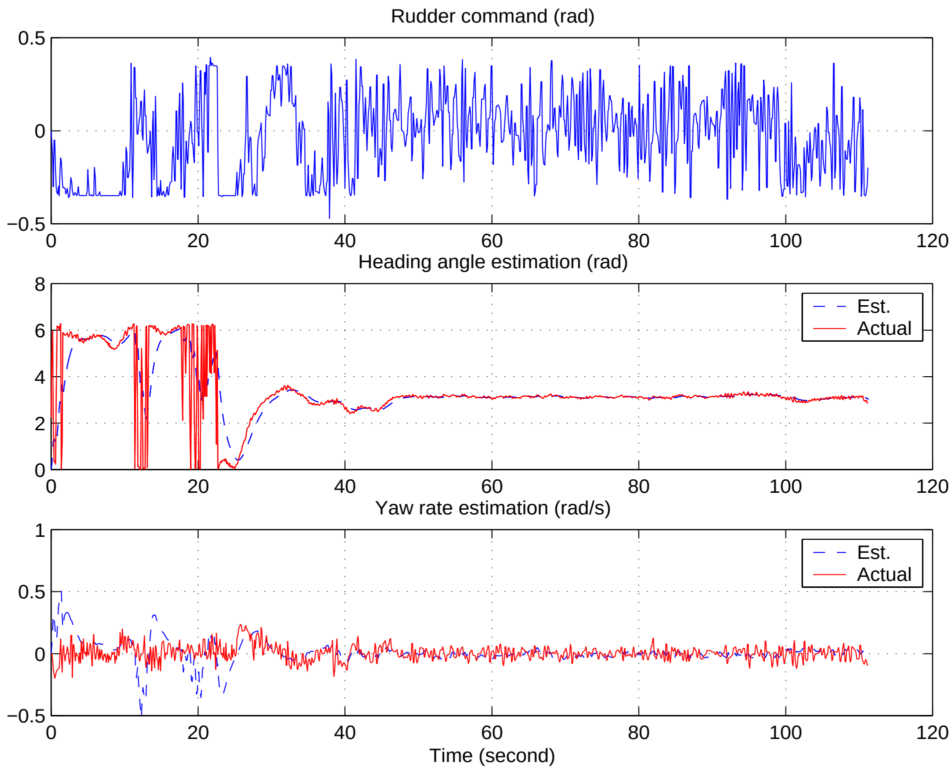

Figure 6.8: AUV Motions in Data No. 2

between them, and gives fault detection results. In this case, there is no failure detected according to the threshold logic.

The third set of data was recorded on Oct. 24, 1999. Figure 6.10 shows its AUV motions. The heading angle is around $0(2\pi)$ position at first. Then, it changes to π and keeps the direction for about $150s$. Finally, it returns to $0(2\pi)$ position again. Figure 6.11 is the diagrammatic sketch of the vehicle trajectory.

Figures 6.12 and 6.13 show FDI results for data set number three. The HFDI detects failures at $t = 12.8s, 178.5s$, and $326.1s$, respectively. The first detection occurs at the beginning, while the other two are coincident with the two turnings of the vehicle. For the first fault detection, the algorithm identifies it as a single yaw-rate-sensor failure. Then it moves to the identification of a dual failure of yaw-rate sensor and rudder. But finally, it

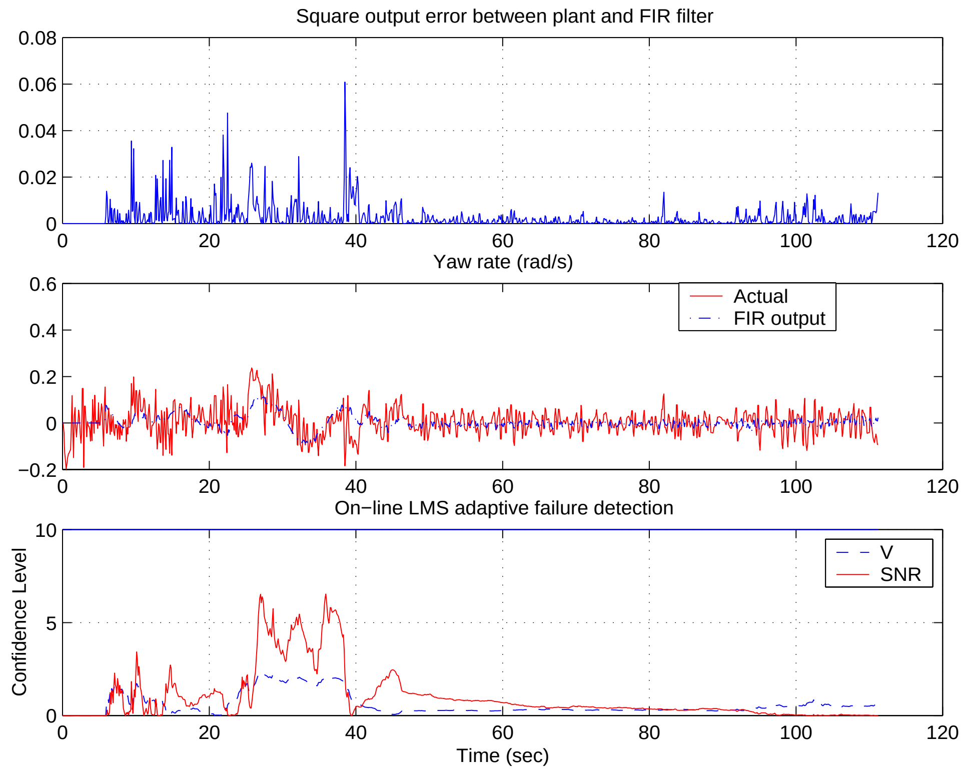

Figure 6.9: Fault Detection with Data No. 2

identifies the normal operation mode and returns to the monitoring loop. For the second detection, again, the algorithm begins with the identification of the single yaw-rate-sensor failure, but ends with the normal mode. For the third detection, the algorithm identifies the normal mode directly. The FDI result suggests no actual failure occurs besides wave disturbances at the turnings.

Data No. 4 was recorded on Feb. 4, 1999. Figure 6.14 shows corresponding AUV motions. The vehicle was moving with the heading around $0(2\pi)$ position. There is some increase in the heading at the end. The algorithm does not detect any failure until $t = 308.9s$. It identifies a single yaw-rate-sensor failure at first. In addition to that, the algorithm also identifies partial failures of both rudder and yaw-rate sensor. It is hard to make any conclusion in this case, since the data end very soon.

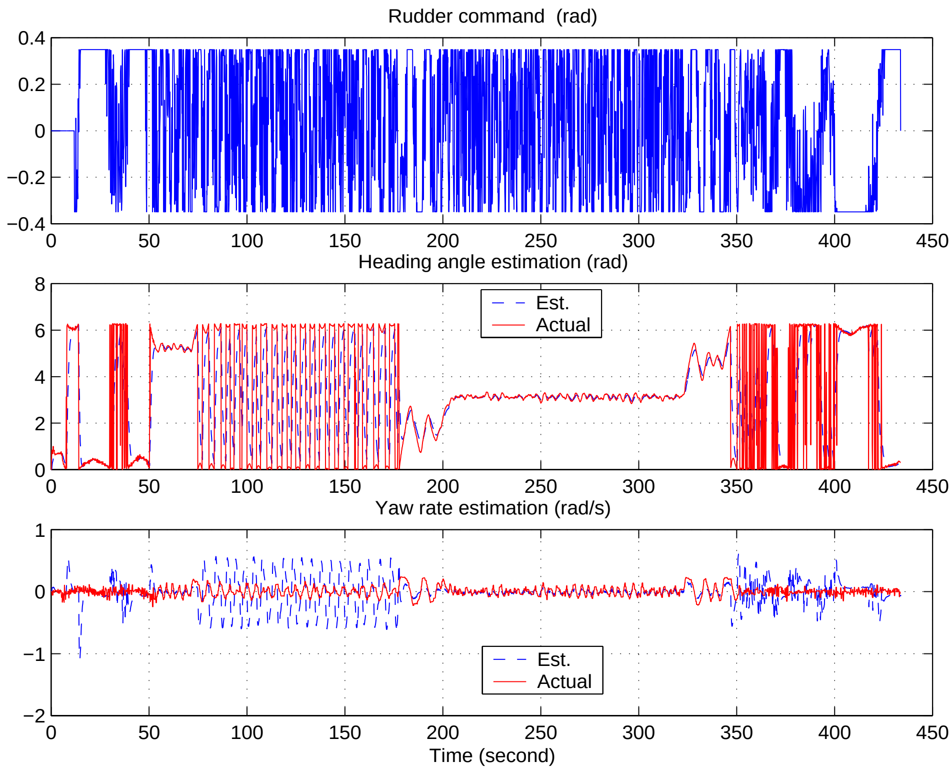

Figure 6.10: AUV Motions in Data No. 3

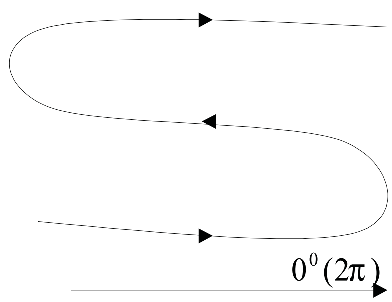

Figure 6.11: Vehicle Trajectory in Data No. 3

We also verify the HFDI in the diving subsystem. Figure 6.17 shows AUV motions of the pitch rate and the pitch angle in data set No. 1. We see actions in the diving subsystem are relatively small. Figures 6.18 and 6.19 show corresponding FDI results. Now the labels represent the normal mode, the stern plane, the pitch-rate sensor, and the pitch-angle sensor.

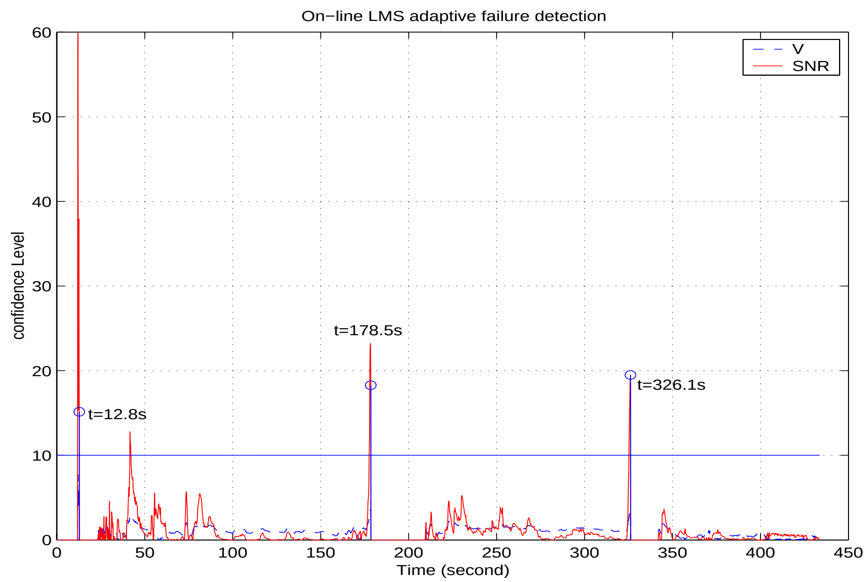

Figure 6.12: Fault Detection with Data No. 3

The algorithm detects a failure around $t = 90s$, and identifies it as a single failure of pitch-rate sensor. After that, the algorithm identifies partial failures of both stern plane and pitch-rate sensor. The algorithm also distributes some probabilities to the normal mode. It even identifies the normal operation mode towards the end and continues fault detection. The result suggests partial failures of the stern plane and the pitch-rate sensor.

6.5 Summary

This chapter tests the HFDI approach with realistic at-sea test data of the OEX series AUVs developed at the Ocean Engineering Department of FAU. We formulate the FDI problem in the steering and the diving subsystems of FAU AUVs, and apply the HFDI using actual actuator command and measurements. Our HFDI algorithm has demonstrated its effectiveness in detecting failures in real scenarios, and its ability to identify failures according to our fault modeling. The results will provide useful information to the analysis of FAU test data concerning the performance of the vehicle. Furthermore, incorporating

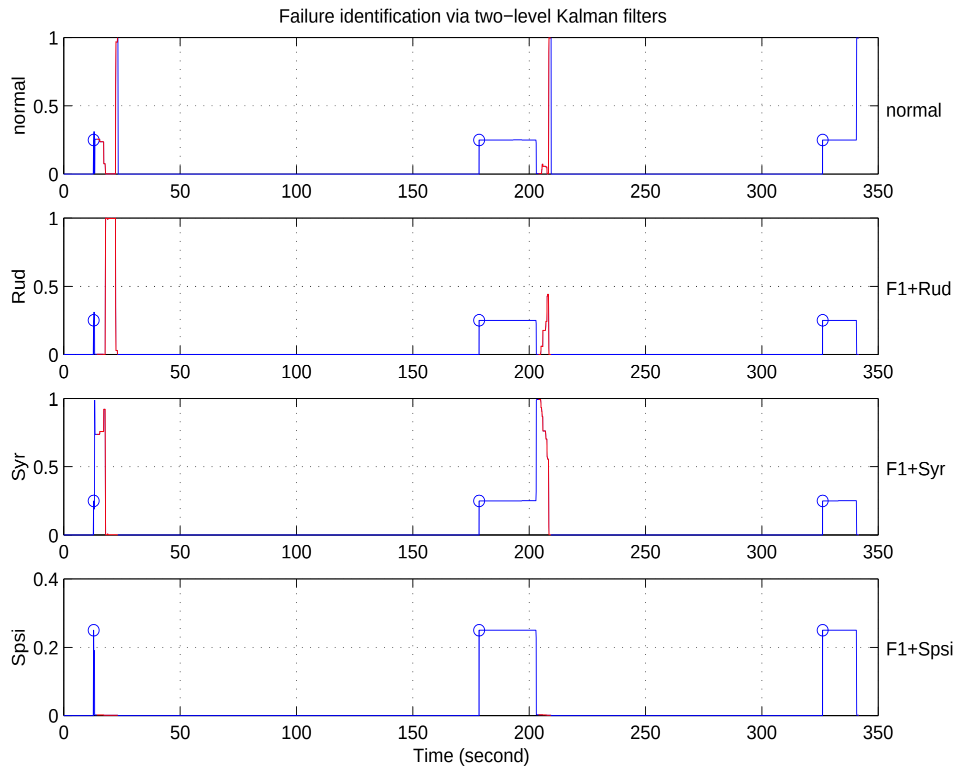

Figure 6.13: Fault Identification with Data No. 3

the HFDI algorithm into FAU AUVs and purposeful commands in the test run can produce better results of fault detection and identification, and benefits the control of the vehicle ultimately. The experiments results can also help improving the HFDI algorithm.

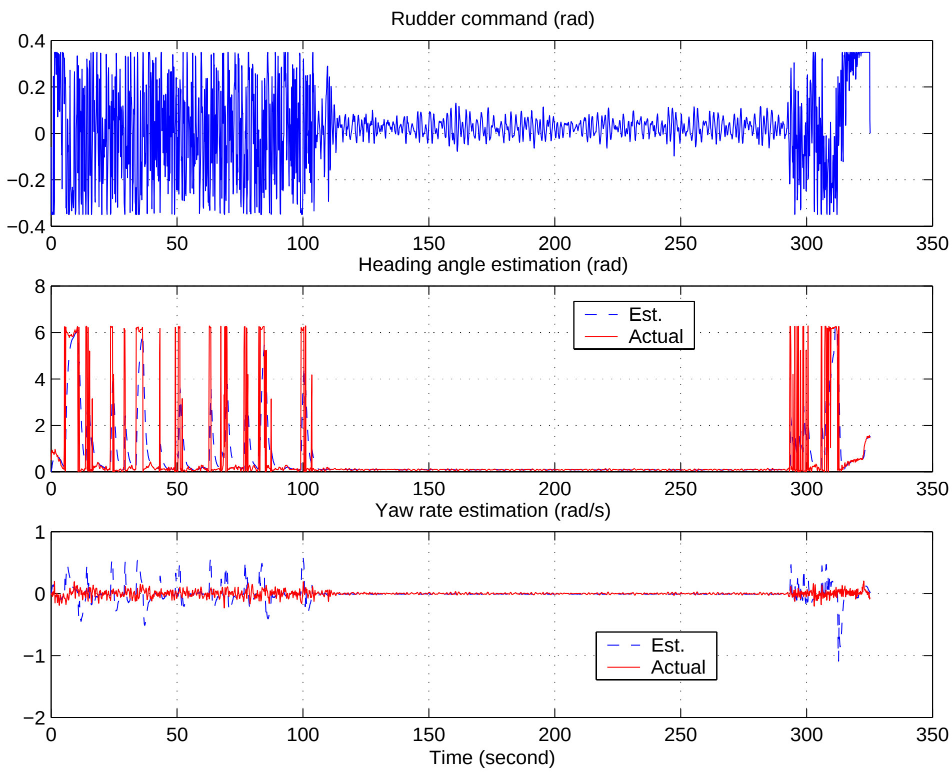

Figure 6.14: AUV Motions in Data No. 4

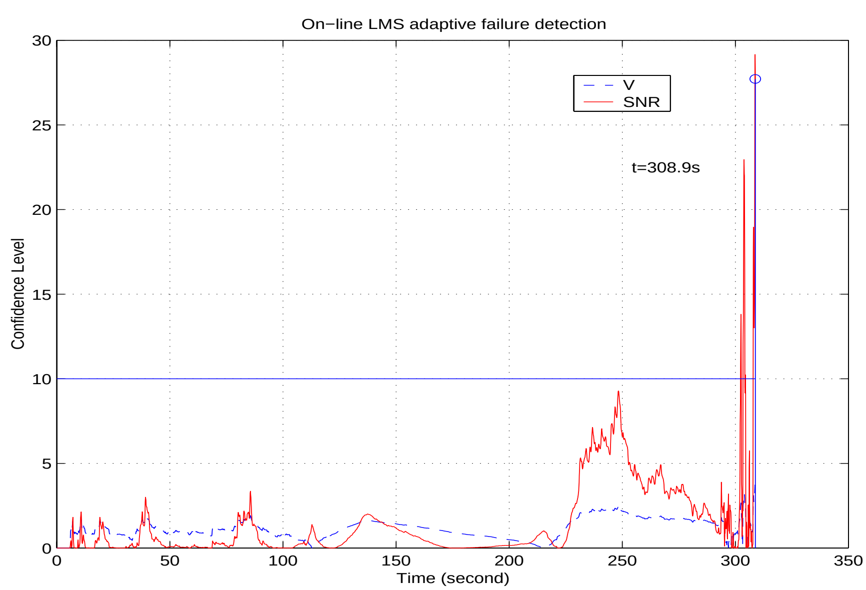

Figure 6.15: Fault Detection with Data No. 4

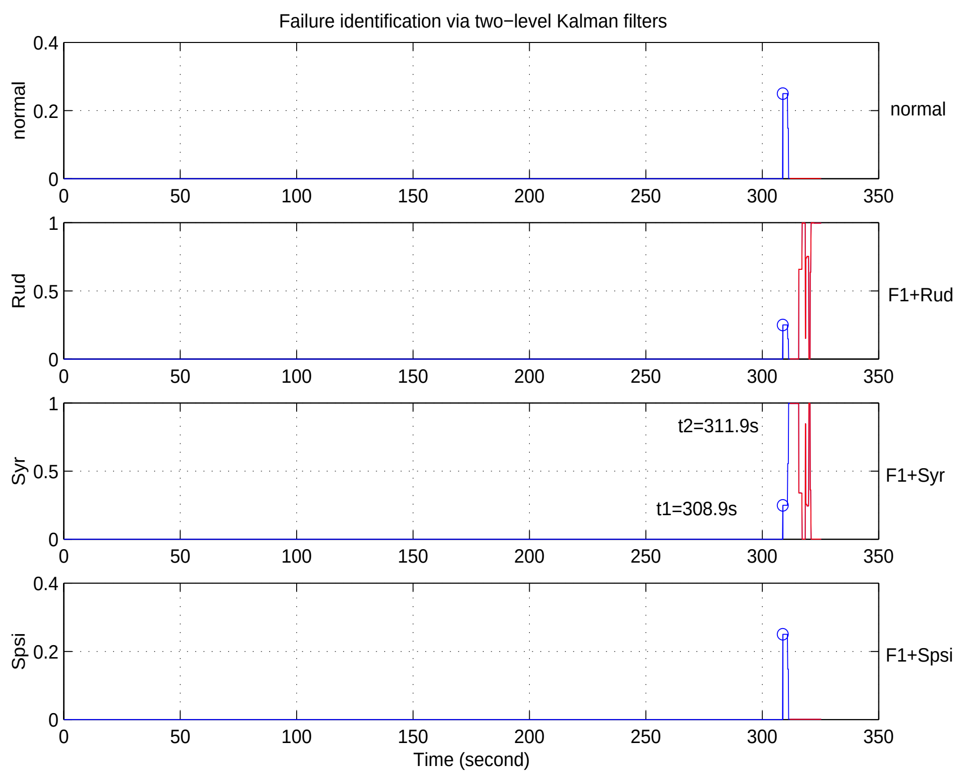

Figure 6.16: Fault Identification with Data No. 4

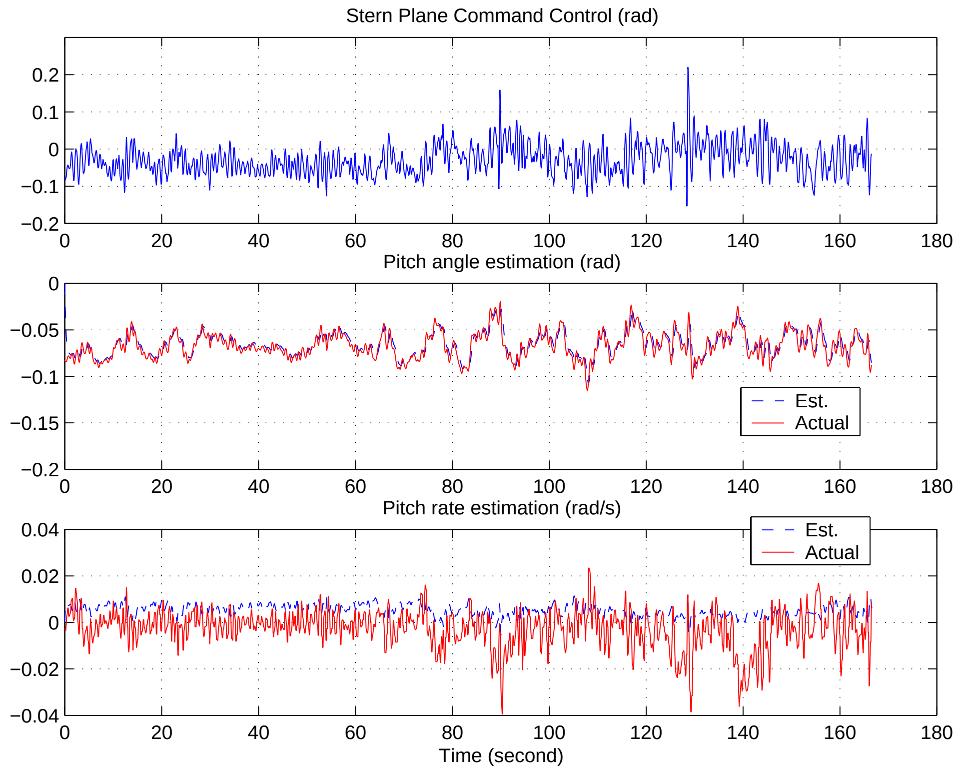

Figure 6.17: AUV Diving Motions in Data No. 1

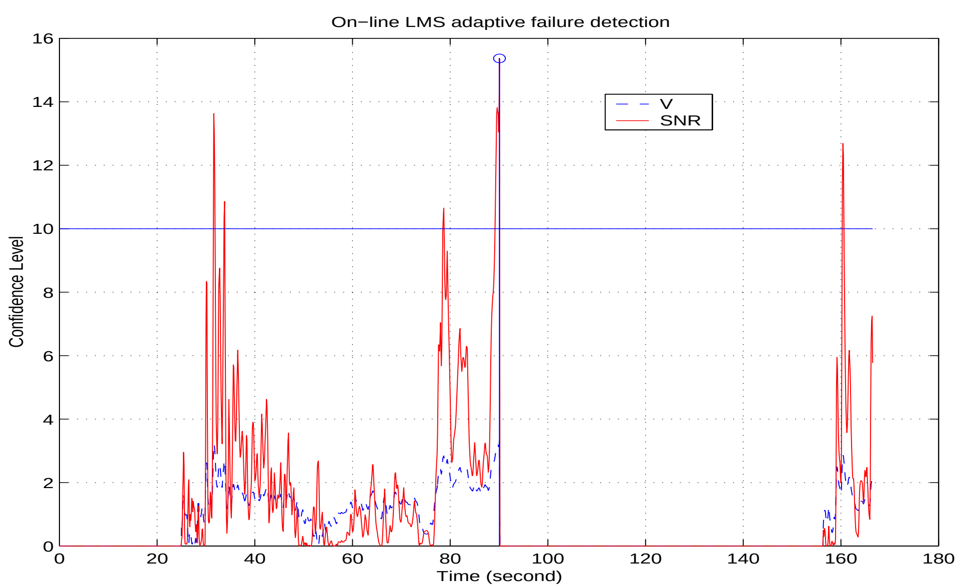

Figure 6.18: Fault Detection with Data No. 1 in Diving Subsystem

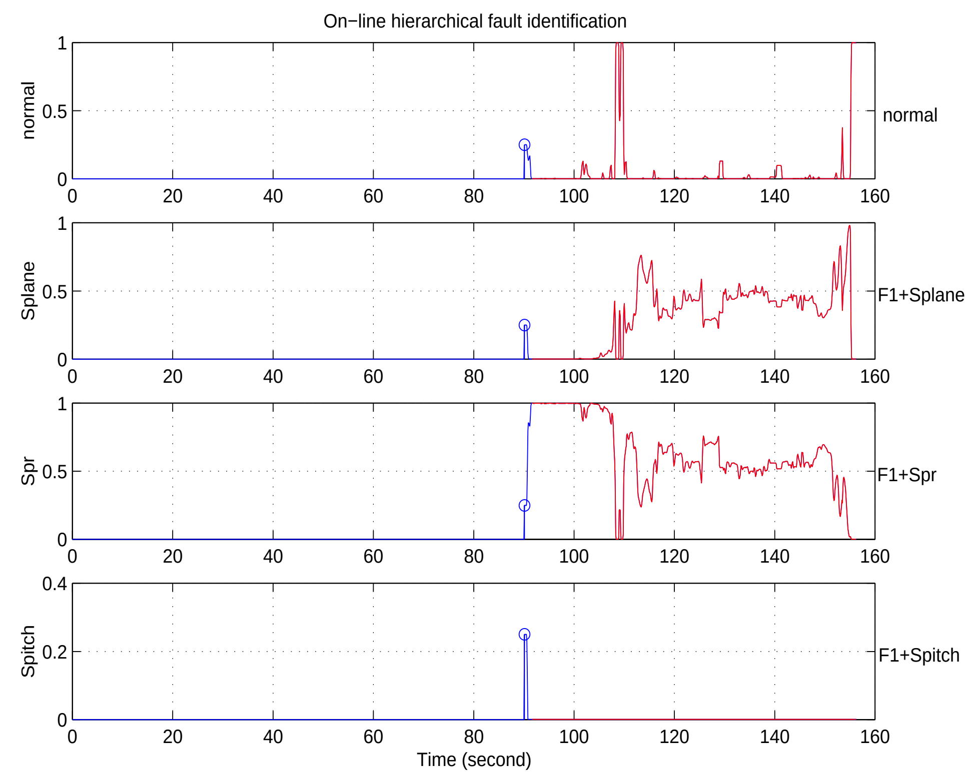

Figure 6.19: Fault Identification with Data No. 1 in Diving Subsystem

Chapter 7

Concluding Discussions and Recommendations for Future Work

This final chapter summarizes the work and discusses the main conclusions of the study. It also presents a number of recommendations for possible future work.

7.1 Summary of the Work

The principle of fault-tolerant control is to monitor the system in real time for failures, identify the failure when it is detected, and compensate for failures to maintain the system performance based on the low-level control and sensory measurements. The technique is important in many complex and critical dynamic systems. This study presents a hierarchical methodology of fault detection, identification and compensation for UUVs. We consider actuator and sensor failures in the decoupled steering and diving subsystems of the Naval Postgraduate School (NPS) UUV. While the control of all six degrees of freedom may be important for future UUV operation, and particularly in the transition from cruise to hover mode, it is not the primary focus in this research.

We first develop sliding-mode controllers for NPS UUV steering and diving subsystems, and perform sliding-mode control of simulated subsystem maneuvers. We also compare the performance of the sliding-mode control with that of the PD control for two kinds of steering maneuvers with and without disturbances: course keeping and path tracking. Simulation results show that the sliding-mode control provides effective and robust ways of control, and is superior to the traditional PD control. We formulate the fault detection and identification problem based on this modeling and control.

Secondly, we propose a fault-detection method based on adaptive LMS modeling and analysis to detect actuator and/or sensor failures. Basically, we use a finite-impulse-response (FIR) filter to model the subsystem adaptively, and use the LMS algorithm to minimize the output difference between the FIR filter and the monitored system. We use the FIR model to approximate the monitored system and analyze the adaptive model history to extract information on fault occurrences.

Thirdly, we design a hierarchical approach of fault detection and identification (HFDI) by combining our fault-detection method with a multiple-model adaptive estimation (MMAE). We use the adaptive modeling and analysis method in its first level for fault detection, and a two-stage parallel Kalman filters in levels two and three for fault identification using the MMAE. We activate latter levels only when the failure is detected.

Lastly, we present a multiple-model adaptive sliding-mode control based on the HFDI to compensate failures. First, we identify actuator and sensor failures through the HFDI. Then, we design multiple sliding-mode controllers associated with each of the hypothesized failure modes, and reconfigure the control signal with a probability-weighted average of all the elemental control signals.

We model actuator failures as additive parameter changes in the control input matrix, and sensor failures as additive parameter changes in the output equation of the system. We consider total failures as well as partial failures, so as to have a comprehensive representation of actuator and sensor failures. In implementation, we simulate single failures, dual failures, and simultaneous failures during the maneuvering. Simulation results have demonstrated effective detection, identification, and compensation of these failures.

Besides validating HFDIC on the steering and diving subsystems of NPS UUV for various simulated actuator and/or sensor failures, we also verify the HFDI method with realistic data from Florida Atlantic University (FAU). We formulate the FDI problem according to subsystem models of FAU AUV, and use the actual control command and measurements in the HFDI.

7.2 Main Conclusions

The main conclusions of this study are as follows.

- The fault-detection method based on the adaptive LMS modeling and analysis is easy to implement, and suitable for on-line application. The adaptive modeling involves simple FIR filters and uses the LMS algorithm to minimize the output difference between the FIR filter and the monitored system. This FIR filter is well suited to represent the impulse responses of UUV subsystems. The adaptive modeling process begins with an FIR model that approximates the impulse response of the dynamic system. When sensor and/or actuator failures change the impulse response of the system, they express themselves as changes in FIR model coefficients. By analyzing adaptive model history, we can extract information on fault occurrence.

- The adaptive analysis of the fault-detection method obtains difference vectors by comparing the FIR model at each sampling period with the first one. Since the initial FIR model is an approximation to the impulse response of the normal operated system, the difference vector represents the deviation of each FIR model from the normal situation. In this way, we observe the variation of the model relative to the normal one. This differs from other techniques that obtain the difference by comparing with the one-step-backward information. In comparison, our difference vectors provide more robust error information.

- The hierarchical methodology integrates the fault detection, identification, and failure compensation functions systematically in three different levels, and enables a computationally efficient implementation of fault-tolerant control. It applies adaptive modeling and analysis in its first level for system identification and fault detection, and multiple-model adaptive estimation with parallel Kalman filters in levels two and three for fault identification. It performs control compensation in level three according to fault identification results. The algorithm does not transfer to fault identification until the failure has been detected in the first level. Also, the algorithm does not proceed to level three for fault compensation until failure occurrence has been confirmed in level two.

- The hierarchical methodology has the ability to remove false alarms. The algorithm also can be separated into two main loops: a monitoring loop including level one for fault detection, and an identification loop including levels two and three. In the identification loop, each level includes the normal operation mode in its multiple-model design. The monitoring loop may mistakenly detect a failure due to disturbances. But the algorithm can quit from the identification loop and return to the monitoring loop by identifying the normal operation mode.

- The fault-tolerant control system autonomously reconfigures the control law to compensate for actuator and sensor failures so that the UUV can continue to operate acceptably. By taking the probability-weighted average of all the elemental control signals, the sliding-mode control corresponding to the dominant failure mode plays an important role in the output. Therefore, the reconfigured control is tailored for the identified failures.

- Applying the HFDI method to FAU real data leads to similar results to those of NPS UUV simulations. We interpret the results in the same way and they are meaningful FDI results relating to AUV motions. The verification results conform the performance of the HFDI, and provide important information on the FAU AUV performance in the test run.

7.3 Recommendations for Future Work

Some suggestions for future work are as follows:

- To incorporate the fault detection, identification, and compensation function into UUV design, we need to investigate hardware/software requirements and limitations on installing the algorithm, and evaluate the performance of the algorithm in the test run. We may also need to investigate enhancement measures if necessary.

- Control reconfiguration is a critical technique given the results of fault detection and identification. We have applied multiple sliding-mode controllers to compensate for the failure with a weighted-sum of the elemental control signals. We can investigate further the form of control algorithm for specific failures. We can also consider intelligent techniques for fault compensation, such as fuzzy logics and neural networks, in which we can include human interventions.

- We have developed the hierarchical fault detection and identification method with applications in UUVs. However, the method is quite general and can be applied to other similar fault detection and identification problems. In these cases, we need to formulate the particular problem of fault detection and identification. We need to investigate the optimal configuration of the approach, such as the form of the dither signal and the purposeful command, the number of FIR coefficients, hypothesized failures and fault modeling, the efficient representation in designing the bank of Kalman filters, and the threshold for alarm announcement. We may also need to deal with specific problems in those applications.

Bibliography

[1] Blanke, M., R. Izadi-Zamanabadi, S.A. Bogh, and C. P. Lunau "Fault-Tolerant Control Systems - A Holistic View," Control Eng. Practice, Vol. 5, pp. 693-702, 1997.

[2] Stengel, R.F. "Intelligent Failure-Tolerant Control," IEEE Control Systems, pp. 14-23, June 1991.

[3] Patton, R. P. Frank, and R. Clark "Fault Diagnosis in Dynamic Systems: Theory and Application," Prentice Hall, Englewood Cliffs, N.J., 1989.

[4] Gertler, J. "Survey of Model-Based Failure Detection and Isolation in Complex Plants," IEEE Control Systems Magazine, Vol. 9, No. 1, pp. 3-11, December 1988.

[5] Nguyen, D.H., and B. Widrow, "Neural Networks for Self-Learning Control Systems," IEEE Control System Magazine, Vol. 10, pp. 18-23, April 1990.

[6] Fadali, M. et al. "Minimum-Time Control of Robotic Manipulators Using a Back Propagation Neural Network," Proceedings of American Control Conference, San Diego, CA, pp. 2997-3000, May 1990.

[7] Passino, K. M., M. Sartori, and P. Antsaklis "Neural Computing for Numeric-to-Symbolic Conversion in Control Systems," IEEE Control System Magazine, Vol. 9, pp. 44-52, April 1989.

[8] Naidu, S., E. Zafiriou, and T. McAvoy "Use of Neural Networks for Sensor Failure Detection in a Control System," IEEE Control System Magazine, Vol. 10, pp. 49-55, April 1990.

[9] Stengel, R.F. and D. J. Linse "System Identification for Nonlinear Control Using Neural Networks," Proceedings of Conference on Information Sciences and Systems, Princeton, NJ, March 1990.

[10] Linse, D. J. and R.F. Stengel "A System Identification Model for Adaptive Nonlinear Control," American Control Conference, Boston, MA, June 1991.

[11] Gertler, J. and D. Singer "A New Structural Framework for Parity Equation-Based Failure Detection and Isolation," Automatica, Vol. 26, No. 2, pp. 381-388, 1990.

[12] White, J. E. and J. L. Speyer "Detection Filter Design: Spectral Theory and Algorithms," IEEE Trans. on Automatic Control, Vol. 32, No. 7, pp. 593-603, July 1987.

[13] Mangoubi, R., B. Appleby, G. Verghese, and W. VanderVelde "A Robust Failure Detection and Isolation Algorithm," Proceedings of the 34th Conference on Decision & Control, New Orleans, LA, pp. 2377-2382, December 1995.

[14] Frank, P.M. "Fault Diagnosis in Dynamic System Using Analytical and Knowledge-Based Redundancy - A Survey and Some New Results," Automatica, Vol. 26, No. 3, pp. 459-474, 1990.

[15] Desai, M., and R. Ray "A Fault Detection and Isolation Methodology," Proceedings of the 20th Conference on Decision and Control, pp. 1363-1369, 1981.

[16] Maybeck, P.S. "Stochastic Models, Estimation and Control," Vol. 2, New York: Academic Press, 1982.

[17] Lainiotis, D.G. "Partitioning: A Unifying Framework for Adaptive Systems, I. Estimation," Proceedings of the IEEE, 64, pp. 1126-1143, 1976.

[18] Athans, M. and C. B. Chang "Adaptive Estimation and Parameter Identification Using Multiple Model Estimation Algorithm," Technical Note 1976-28, ESD-TR-76-184, Lincoln Laboratory, Lexington, MA, June 1976.

[19] Magill, D.T. "Optimal Adaptive Estimation of Sampled Stochastic Processes," IEEE Trans. on Automatic Control, AC-10, No. 4, pp. 434-439, October 1965.

[20] de Benito, C.D. "On-Board Real-Time Failure Detection and Diagnosis of Automotive Systems," Trans. of ASME, Journal of Dynamic Systems, Measurement, and Control, Vol. 112, No.4, pp. 769-773, December 1990.

[21] Menke, T. E. and P. S. Maybeck "Sensor/Actuator Failure Detection in the Vista F-16 by Multiple Model Adaptive Estimation," IEEE Transactions on Aerospace & Electronic Systems, Vol. 31, No. 4, pp. 1218-1228, October 1995.

[22] Isermann, R. "Process Fault Detection Based on Modeling and Estimation Methods - A Survey," Automatica, Vol. 20, No. 4, pp. 387-404, 1984.

[23] Healey, A.J. and D. B. Marco "A Neural Network Approach to Failure Diagnostics for Underwater Vehicles," Proceedings of IEEE Oceanic Engineering Society Symposium on Autonomous Underwater Vehicles, AUV-92, Washington D.C., pp. 131-135, June 2-3 1992.

[24] Rae, G. J. S. "Damage Detection for Autonomous Underwater Vehicles," Ph.D. Dissertation, Dept. of Ocean Engineering, Florida Atlantic University, Boca Raton, Florida, 1993.

[25] Rauch, H. E. "Autonomous Control Reconfiguration," IEEE Control Systems, pp. 37-48, December 1995.

[26] Maybeck, P. S. "Stochastic Models, Estimation and Control," Vol. 3, New York: Academic Press, 1982.

[27] Greene, C. S. and A. S. Willsky "An Analysis of the Multiple Model Adaptive Control Algorithm," Proceedings of the IEEE Conference on Decision and Control, Albuquerque, NM, pp. 1142-1145, December 1980.

[28] Athans, M., et al. "The Stochastic Control of the F-8C Aircraft Using a Multiple Model Adaptive Control (MMAC) Method - Part 1: Equilibrium Flight," IEEE Trans. on Automatic Control, AC-22, No. 5, pp. 768-780, October 1977.

[29] Maybeck, P. S. and D. L., Pogoda "Multiple Model Adaptive Controller for the STOL F-15 with Sensor/Actuator Failures," Proceedings of the IEEE Conference on Decision and Control, Tampa, FL, pp. 1566-1572, December 1989.

[30] Maybeck, P. S. and R. D., Stevens "Reconfigurable Flight Control Via Multiple Model Adaptive Control Methods," IEEE Trans. on Aerospace and Electronic Systems, Vol. 27, No. 3, pp. 470-479, May 1991.

[31] Noura, H., D. Sauter, F. Hamelin, and D. Theilliol "Fault-Tolerant Control in Dynamic Systems: Application to a Winding Machine," IEEE Control Systems Magazine, pp. 33-49, February 2000.

[32] Cheng A. S.-F. and N. E. Leonard "Fin Failure Compensation for an Unmanned Underwater Vehicle," Proceedings of the 11th International Symposium on Unmanned Untethered Submersible Technology, pp. 342-351, August 1999.

[33] Yang, K. C. H., J. Yuh, and S. K. Choi "Fault-Tolerant System Design of an Autonomous Underwater Vehicle - ODIN: an Experimental Study," International Journal of Systems Science, Vol. 30, No. 9, pp. 1011-1019, 1999.

[34] Kwong, W. A., K. M. Passino, E.G. Laukonen, and S. Yurkovich "Expert Supervision of Fuzzy Learning Systems for Fault Tolerant Aircraft Control," Proceedings of the IEEE, Vol. 83, No. 3, pp. 342-351, March 1995.

[35] Layne, J. R. and K. M. Passino "Fuzzy Model Reference Learning Control for Cargo Ship Steering," IEEE Control Systems Magazine, December 1993.

[36] Moudgal, V., W. A. Kwong, K. M. Passino, and S. Yurkovich "Fuzzy Learning Control for a Flexible-Link Robot," IEEE Trans. on Fuzzy Systems, Vol. 3, No. 2, May 1995.

[37] Passino, K. M. "Intelligent Control for Autonomous Systems," IEEE Spectrum, June 1995.

[38] Society of Naval Architects and Marine Engineers (SNAME), "Nomenclature for Treating the Motion of a Submerged Body through a Fluid," Tech. Res. Bull., Vol. 1-5, 1950.

[39] Fossen, T.I. "Guidance and Control of Ocean Vehicles," John Wiley & Sons, New York, 1994.

[40] Healey, A.J. and D. B. Marco "Slow Speed Flight Control of Autonomous Underwater Vehicles: Experimental Results with the NPS AUV II," Proceedings of the 2nd International Offshore and Polar Engineering Conference(ISOPE), San Francisco, CA, pp. 523-532, June 14-19,1992.

[41] Healey, A. J. and D. Lienard "Multivariable Sliding Mode Control for Autonomous Diving and Steering of Unmanned Underwater Vehicles," IEEE Journal of Oceanic Engineering, OE-18(3), pp. 327-339, July 1993.

[42] Jalving, B. "The NDRE-AUV Flight Control System," IEEE Journal of Oceanic Engineering Vol. 19, No. 4, pp. 497-501, October 1994.

[43] Rodriguez, R. R. and G. J. Dobeck, "Guidance and Control System of the Large Scale Vehicle," Proceedings of the 6th International Symposium on Unmanned Untethered Submersible Technology, Marine Systems Engineering Laboratory, University of New Hampshire, pp. 434-451, June 1989.

[44] Russel, G. T. and J. Bugge, "An Integrated Guidance and Control System for an Unmanned Tethered Submersible," Proceedings of IEE Conference on Control and Its Applications, Coventry, Publication 194, pp. 281-285, March 23-25, 1981.

[45] Yoerger, D.R. and J.J.E. Slotine "Nonlinear Trajectory Control of Autonomous Underwater Vehicles using the Sliding Methodology," Proceedings of the ROV'84 Conference, pp. 245-251, 1984.

[46] Yoerger, D.R. and J.J.E. Slotine "Robust Trajectory Control of Underwater Vehicles," IEEE Journal of Oceanic Engineering OE-10(4), pp. 462-470, 1985.

[47] Yoerger, D.R. and J.J.E. Slotine "Adaptive Sliding Control of an Experimental Underwater Vehicle," Proceedings of IEEE International Conference on Robotics and Automation, Sacramento, California, pp. 2746-2751, 1991.

[48] Cristi, R., F.A., Papoulias, and A.J., Healey "Adaptive Sliding Mode Control of Autonomous Underwater Vehicles in the Dive Plane," IEEE Journal of Oceanic Engineering OE-15(3), pp. 152-160, 1990.

[49] Yoerger, D.R., J.B. Newman, and J.J.E., Slotine "Supervisory Control System for the Jason ROV," IEEE Oceanic Engineering, OE-11(3), pp. 392-400, 1986.

[50] Dougherty, F., T. Sherman, G. Woolweaver and G. Lovell "An Autonomous Underwater Vehicle (AUV) Flight Control System Using Sliding Mode Control," Ocean'88, Baltimore, MD, pp. 1265-1270, 1988.

[51] Dougherty, F. and G. Woolweaver, "At-Sea Testing of an Unmanned Underwater Vehicle Flight Control System," Symposium on Autonomous Underwater Technology, Washington, D.C., pp. 65-68, 1990.

[52] Slotine, J.J.E. and W. Li "Applied Nonlinear Control," Prentice Hall, 1991.

[53] Widrow, B. and E. Walach "Adaptive Inverse Control," Prentice Hall, Englewood Cliffs, N.J., 1996.

[54] Kuo, S. M. and D. R. Morgan "Active Noise Control Systems: Algorithms and DSP Implementations," John Wiley & Sons, Inc., New York, 1996.

[55] Kalman, R.E. "A New Approach to Linear Filtering and Prediction Problems," ASME Journal of Basic Engineering, Vol. 86, pp. 35-45, 1960.

[56] Anderson, B. D. O. and J. B. Moore "Adaptive Filtering," Prentice Hall, Englewood Cliffs, N.J., 1979.

[57] Ni, L. and C.R. Fuller "On-line Fault Detection and Diagnosis for Unmanned Underwater Vehicles (UUVs)," Second International Workshop on Fault Detection and Compensation, Denver, CO, August 11-12, 1998.

[58] Ni, L. and C.R. Fuller "Fault Detection Based on Adaptive LMS Modeling and Analysis for Unmanned Underwater Vehicles (UUVs)," Proceedings of the 1999 ASME - Applied Mechanics and Materials Conference, Blacksburg, VA, June 27-30, 1999.

[59] Ni, L. and C.R. Fuller "An LMS Method for On-line Fault Detection in Unmanned Underwater Vehicles (UUVs)," Proceedings of the Symposium on Underwater Robotic Technology, World Automation Congress 2000, Wailea, Maui, Hawaii, June 11-16, 2000.

[60] Ni, L. and C.R. Fuller "Re-configurable Control of Unmanned Underwater Vehicles (UUVs) Based on Hierarchical Fault Detection and Identification (HFDI)," Special Edition in Nonlinear Dynamics: "Active Control of Nonlinear Dynamical Systems," 8th Conference on Nonlinear Vibrations, Stability, and Dynamics of Structures, Blacksburg, VA, July 23-27, 2000.

[61] Ni, L. and C.R. Fuller "Hierarchical Fault Detection and Identification for Unmanned Underwater Vehicles (UUVs)," submitted to ASME Journal of Dynamic Systems, Measurement, and Control, 2000.

Vita

Lingli Ni was born on April 24, 1970 in Longyan, China. She received both her B.S. and M.S. degrees in Electrical Engineering in 1991 and 1994, respectively, from the Department of Automatic Control of Northwestern Polytechnical University, Xi'an, China. She came to the Unites States for Ph.D studies in the Mechanical Engineering Department of Virginia Polytechnic Institute and State University in January 1997, under the supervision of Professor Chris Fuller. She completed her Doctoral degree in June 2001.