Summer 2018

Identification and Optimal Linear Tracking Control of ODU Autonomous Surface Vehicle

Nadeem Khan
Old Dominion University

IDENTIFICATION AND OPTIMAL LINEAR TRACKING CONTROL OF ODU AUTONOMOUS SURFACE VEHICLE

by

Nadeem Khan
B.Sc. September 2001, UET Peshawar, Pakistan
M.Sc. August 2009, UET Peshawar, Pakistan

A Thesis Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

MECHANICAL AND AEROSPACE ENGINEERING

OLD DOMINION UNIVERSITY
August 2018

Approved by:

Jen-Kuang Huang (Director)

Gene Hou (Member)

Han Bao (Member)

Duc Nguyen (Member)

ABSTRACT

IDENTIFICATION AND OPTIMAL LINEAR TRACKING CONTROL OF

ODU AUTONOMOUS SURFACE VEHICLE

Nadeem Khan
Old Dominion University, 2018
Director: Dr. Jen-Kuang Huang

Autonomous surface vehicles (ASVs) are being used for diverse applications of civilian and military importance such as: military reconnaissance, sea patrol, bathymetry, environmental monitoring, and oceanographic research. Currently, these unmanned tasks can accurately be accomplished by ASVs due to recent advancements in computing, sensing, and actuating systems. For this reason, researchers around the world have been taking interest in ASVs for the last decade. Due to the ever-changing surface of water and stochastic disturbances such as wind and tidal currents that greatly affect the path-following ability of ASVs, identification of an accurate model of inherently nonlinear and stochastic ASV system and then designing a viable control using that model for its planar motion is a challenging task. For planar motion control of ASV, the work done by researchers is mainly based on the theoretical modeling in which the nonlinear hydrodynamic terms are determined, while some work suggested the nonlinear control techniques and adhered to simulation results. Also, the majority of work is related to the mono- or twin-hull ASVs with a single rudder. The ODU-ASV used in present research is a twin-hull design having two DC trolling motors for path-following motion.

A novel approach of time-domain open-loop observer Kalman filter identifications (OKID) and state-feedback optimal linear tracking control of ODU-ASV is presented, in which a linear state-space model of ODU-ASV is obtained from the measured input and output data. The

accuracy of the identified model for ODU-ASV is confirmed by validation results of model output data reconstruction and benchmark residual analysis. Then, the OKID-identified model of the ODU-ASV is utilized to design the proposed controller for its planar motion such that a predefined cost function is minimized using state and control weighting matrices, which are determined by a multi-objective optimization genetic algorithm technique. The validation results of proposed controller using step inputs as well as sinusoidal and arc-like trajectories are presented to confirm the controller performance. Moreover, real-time water-trials were performed and their results confirm the validity of proposed controller in path-following motion of ODU-ASV.

This thesis is dedicated to my parents, siblings, wife, and kids.

ACKNOWLEDGMENTS

I am truly obliged to express my gratitude to my advisor, Dr. Jen-Kuang Huang from the core of my heart for his heedful motivation, encouragement, as well as technical guidance during the entire research. The members of my doctoral committee, Dr. Gene Hou, and Dr. Duc Nguyen, are appreciated for their inferential suggestions in completing my dissertation. I would especially thank to Dr. Gene Hou for his facilitative insight in the area of my research. He also fully supported me to have access the intelligent machine lab and the equipment/sensors therein, which I needed for this research. I would also never forget to thank Dr. Sebastian Bawab, the chair of the department of Mechanical and Aerospace Engineering, for his kind support in the form of fiscal matters that I needed earnestly in the course of completing my dissertation up to the very last moment. I express my appreciation to my friends: Mr. Theodore Teats, Mr. David, and Mr. Michael Polanco for their technical suggestions and help from time to time.

I also want to express my gratitude and respect to my parents and brothers for their continuous encouragement and everlasting prayers in my work as well as in my life. Moreover, I am grateful to and proud of my teenage kids for helping me in carrying the equipment to the outdoor experiment's site which is required for this research. Last but not least, I am very thankful to my wife for her companionship, struggle, and devotion in taking care of all business related to the household during my entire research.

ACRONYMS

ACF:	Auto-correlation Function
ANN:	Artificial Neural Network
ARE:	Algebraic Riccati Equation
ARX:	Auto Regressive with External Input
ASC:	Autonomous Surface Craft
ASV:	Autonomous Surface Vehicle
AUVSI:	Association for Unmanned Vehicle Systems International
BT:	Balanced Truncation
CG:	Center of Gravity
DAQ:	Data Acquisition
DOF:	Degree of Freedom
EAs:	Evolutionary Algorithms
ECEF:	Earth-centered, Earth-fixed
ERA:	Eigen-system Realization Algorithm
FRF:	Frequency Response Function
E_{SS}:	Steady-state Error
GA:	Genetic Algorithm
GCS:	Ground Control Station
GNSS:	Global Navigation Satellite System
GPS:	Global Positioning System
GUI:	Graphical User Interface

GWN:	Gaussian White Noise
HSV:	Hankel Singular Value
IMU:	Inertial Measurement Unit
LIDAR:	Light Detection and Ranging
LMBP:	Levenberg-Marquardt Backpropagation
LMI:	Linear Matrix Inequality
LOS:	Line of Sight
LPF:	Low Pass Filter
LQG:	Linear Quadratic Gaussian
LQR:	Linear Quadratic Regulator
LQT:	Linear Quadratic Tracker
LS:	Least Squares
LTI:	Linear Time Invariant
MAC:	Modal Amplitude Coherence
MFC:	Model Following Control
MIMO:	Multiple Inputs, Multiple Outputs
MLE:	Maximum Likely Estimation
MOEAs:	Multi Objective Evolutionary Algorithms
MOGA:	Multi Objective Genetic Algorithm
MOP:	Multiobjective Optimization Problem
MPs:	Markov Parameters
MPC:	Model Predictive Control
MSV:	Mode Singular Value

NARX:	Nonlinear Auto-Regressive with External Input
NED:	North East Down
NMEA:	National Marine Electronics Association
NN:	Neural Networks
NRMSE:	Normalized Root Mean Square Error
OKID:	Observer Kalman Filter Identification
OMPs:	Observer Markov Parameters
PI:	Performance Index
PN:	Proportional Navigation
PWM:	Pulse Width Modulation
RC:	Radio Control
SI:	System Identification
SISO:	Single Input, Single Output
SVD:	Singular Value decomposition
TTL:	Transistor-Transistor Logic
USV:	Unmanned Surface Vehicle
WGS-84:	World Geodetic System-1984

NOMENCLATURE

$\hat{A}, \hat{B}, \hat{C}, \hat{D}$:	Identified system matrices
$\mathcal{C}$:	Block controllability matrix
e:	Random Gaussian white noise error signal
F_s:	Sampling frequency
G:	Observer gain matrix
H(0):	Block Hankel matrix
$'\mathrm{H}(0)$:	Block correlation matrix
$H(1)$:	Shifted block Hankel matrix
I:	Identity matrix
J:	Quadratic cost function or performance index
K:	Kalman filter gain
K_∞:	Steady-state sub-optimal state feedback gain
$K(t)$:	Time-varying state feedback gain matrix
K_k^v:	Feedforward gains matrix at index *k*
k:	Time index
l:	Length of data
m:	Number of outputs
n:	Number of states
$\mathcal{O}$:	Block observability matrix
Q:	Symmetric positive semi-definite state weighting matrix
R:	Symmetric positive definite control weighting matrix

R:	Reachability matrix
R_f:	Goodness of fit
R_{hh}:	Data correlation matrix
r:	Number of inputs
u:	Control inputs vector
w:	Heave
x:	System states vector
x_d:	Decision variables vector
$\overline{\mathrm{Y}}$:	Observer Markov parameters
$\hat{Y}$:	Identified Markov parameters
Y_k^o:	Observer gain Markov parameters at index k
Y_k:	System Markov parameters at index k
y:	System outputs vector
†:	Moore-Penrose pseudo-inverse
$\hat{\Lambda}$:	Diagonal matrix of identified eigenvalues
ψ:	Nonsingular modal matrix
$\sum$:	Diagonal matrix of singular values
σ:	Singular values
λ_i:	i^{th} eigenvalue
θ:	Pitch angle
ϕ:	Roll angle
$\dot{\psi}$:	Yaw rate

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Motivation and Problem Statement

Intelligent robotic or autonomous vehicles are the unmanned air, land, surface, and underwater systems that integrate the computer system, microcontroller, guidance, navigation, and control system, as well as various other automation technologies to perform the various tasks of civilian and military use. For this very purpose, a significant level of autonomous functions in autonomous vehicles has been successfully incorporated. Autonomous surface vehicles (ASVs) or unmanned surface vehicles (USVs) are one of the categories of autonomous or unmanned systems and have prominent and diverse applications such as military reconnaissance, sea patrol, bathymetry, riverine or shallow water survey, environmental monitoring and oceanographic research. That is the reason that ASVs or USVs are gaining attention from the researchers around the world for the last decade [1], [2], [3]. The abovementioned various autonomous or unmanned tasks are effectively accomplished due to the recent advancements of computing, navigation, wireless communication, vision, laser measurement, and propulsion systems. These advancements have made the ASV more efficient and sophisticated in performing difficult and dangerous tasks. Hence, due to this fact, autonomous vehicles are, in general, most preferable to be used for their capabilities so that human operators are removed from the inhospitable and/or inaccessible marine or other shallow water working environments.

In order to accomplish these sophisticated tasks, an optimal controller is essentially required; optimal in a sense that the control inputs required for the ASV motion will be optimized

with respect to an already defined performance index (PI) while achieving the desired objective. In this scenario, to control the motion of ASV optimally is a challenging task due to the following facts. First, for optimal control, an accurate dynamical model of the ASV system under study is very essential; however, it is quite challenging and hard to incorporate the available theoretical formulations to achieve an accurate dynamical model of ASV because these techniques are normally nonlinear, time-varying, and coupled in nature [4] [5]. Second, water is the ever-changing surface, and the outdoor marine as well as estuarial tasks are stochastic in nature. Because most of the time these tasks performed by the light-weight and small size ASVs are accompanied by time-varying environmental disturbances such as wind and tidal currents, which are very unpredictable. These environmental disturbances are the main stochastic perturbations to the desired control system; that's why the ASV modeling is a stochastic process.

There are various theoretical formulations, which are basically based on first-principle modeling techniques available to model the dynamics of a powered ASV cruising on the water surface. These formulations, as mentioned above, are inherently nonlinear due to the nonlinear hydrodynamic and coriolis forces therein [4], [6]. There is some previous work done related to nonlinear control based on theoretical formulations for the motion control of ASVs such as backstepping control, sliding mode control, and adaptive control, but most of the work adhered to the simulation validations for their proposed models and controllers [7], [8], [9], [10], [11]. Some of the ASVs used in the previous work were mono-hull and rudder based systems which need one propeller and a thruster [12], [13]. A few studies that were performed were related to empirical modeling which is based on the experimental work, and the ASV used was catamaran type having two thrusters [14], [15]; however, the ASV is modeled as two inputs and single output system and also simulation validations are given. The work of Elkaim, G.H., [16] is based on the

observer Kalman filter identification (OKID) modeling; he a used sailing-type catamaran having one thruster and one rudder mechanism, whereas, the system configuration of the prototype ODU autonomous surface vehicle (ODU-ASV) is of twin-hull and two fixed DC trolling motors which is used for various path-following tasks. The ODU-ASV named *Big Blue* was used in this research in order to determine its linear time-invariant state-space model and later to use the identified model for the design of desirable controller for its motion control. Therefore, the acronyms ASV and ODU-ASV will sometimes be used interchangeably from now onward in this dissertation, unless the usage of acronym entails the ASV in general.

The system identification and control design for such type of inherently nonlinear and stochastic system is a challenging task. The OKID methodology is basically very suitable for a stochastic system such as the ODU-ASV due to the advancement in computing and navigation technology. Hence, the process of OKID is well applied with the help of incorporating a fast processing computer and accurate pose sensors of high sampling rates. After identifying the system, it is well utilized in the control design, which can easily be implemented with the help of computer and microcontroller boards. Therefore, in the present work, the OKID methodology is performed in order to identify the discrete linear time-invariant system and later on utilized the OKID-identified model to design optimal linear quadratic tracking controller for the ODU-ASV planar motion.

1.2 Survey of Previous Work

The scope of this dissertation pertains to the potential research areas such as system identification/modeling and control of ODU-ASV. Therefore, the relevant prior work related to the system identification/modeling and control of ASV dynamical motion is concisely reviewed and discussed in this section.

Elkaim, G.H., et al. [7] developed a nonlinear model of a modified Prindle-19 light catamaran called the Atlantis based on wind and water current models and subsequently designed PI feedback control for its motion control while traversing a series of user defined way-points as heading reference. This controller of the Atlantis makes it have station keeping functionality in the presence of water currents while motor powered. They performed Monte Carlo simulations of their proposed guidance and control system on the nonlinear model and showed that the Atlantis is capable of maintaining a cross-track error of less than one meter throughout the path; but large cross-track errors would be produced during the transition of way-point navigation. The time domain OKID has been applied by Elkaim, G.H., [16] for system identification of autonomous wind-propelled catamaran. The sea trials were performed for the system identification task, in which the catamaran was driven on the straight line paths. The rudder is used for those straight line maneuvers and the inputs were generated by the human pilot. In this work, the system of autonomous catamaran was considered to be the velocity invariant, so the yaw dynamics does not depend on velocity. A discrete time invariant model of order four was identified and tested, which showed acceptable prediction results for yaw dynamics.

Caccia, M., et al. [17], [18] used a small and low cost prototype catamaran autonomous surface craft (ASC) known as Charlie. In this study they assumed a reference theoretical nonlinear model of Blanke for speed and steering equations and proposed a simple experimental dynamical model for the ASC with limited instrumentation by identifying the hydrodynamic drag and actuator interaction effects through sea trials. Their model's steering system based on rudder and the sway speed was considered to be negligible. The application of their practical model based on theoretical model is to support the design of vessel automatic navigation, guidance and control system. The experiments showed that with the extended Kalman filter and simple PID

guidance and control laws, the ASC having only GPS and compass perform basic control tasks such as auto-heading, auto-speed, and straight line following. Gomes, P., et al. [11] used a two pontoon design autonomous surface craft having two propellers called Delfim for its path-following guidance and control. The error vector is driven to zero for the said controller, so for modeling of error dynamics, a poly-topic linear parameter varying (LPV) representation with piecewise affine dependence on the selected parameters is used. The formulation of control synthesis is done as a discrete-time H_2 control problem for LPV system and solved using linear matrix inequalities (LMIs). A preview controller design method is used for increasing path-following performance. The nonlinear controller is achieved and implemented with D-methodology under the gain-scheduling control domain. Simulations were given for the said control testing using full nonlinear model.

Sliding mode control law was proposed by Ashrafiuon, H., et al. to implement trajectory tracking control of under-actuated small autonomous surface vessel experimentally [10]. They also designed the control law by introducing the first-order sliding surface as surge motion tracking errors and the second-order sliding surface as the sway motion tracking errors. They performed experiments using a small surface vessel having two propellers in an indoor pool environment. A camera was used for the absolute position and orientation measurements of their vessel. The input voltage values to the motors were estimated from controller propeller forces and transmitted wirelessly to the motors. Straight-line and circular maneuvering experiments were performed for checking the validity of their proposed controller. A model-based multi-variable sliding mode control law for trajectory tracking of a ship was also suggested by Cheng, J., et al. [19]; in this work the positions and yaw angle of the ship motion were simultaneously tracked.

The work of Wei, M., et al. [20] is about sliding mode control of under-actuated surface

vehicles based on formation control scheme. The surface vehicle is considered to be under-actuated because the sway speed is not considered, so it is not directly actuated. The formation control scheme is achieved based on the leader-following strategy. Both Lyapunov's direct method and sliding mode control techniques have been utilized for the design of proposed controller. The surge and sway motion tracking errors were introduced as the first-order and second-order surfaces, respectively. Numerical simulations were performed to check the effectiveness of the designed sliding mode formation controller.

Shr, S. H., et al. [21] worked on robust nonlinear course keeping control of ship, having a rudder and single propeller, in presence of high wind and wave disturbances. He made use of two feedback loops, i.e. the inner-loop is H_∞ approximate input-output linearization and the outer-loop is related to μ-synthesis technique to deal with tracking, regulation, and robustness problems. He showed in his simulation that a robust nonlinear controller design performs better than linear robust control in presence of environmental high wind and wave conditions. Similarly, Jerzy, B., [13] worked on the design of robust nonlinear control system of the ship course angle based on a model following control (MFC) structure obtained through input-output linearization. In his work, the controller responds to deviation between the model output and real output signal of a mono-hull ship having a rudder and single propeller. In this simulation based work, surge speed and yaw angle are considered. So his proposed system can substitute the complex adaptive control system with MFC structure, which compensates for the difference of his linearized model and the nonlinear characteristics of the process.

Guerreiro, Bruno J., et al. [22] employed nonlinear model predictive controller (MPC) techniques for the problem of trajectory tracking control for ASC. In this work, a two-hull ASC is used, which is propelled by two propellers driven by electric motors. In their approach, a non-

linear dynamic model of the ASC is derived from first principles. Based on this derived model, a trajectory tracking error-space model is proposed. At each instant of time, the nonlinear MPC algorithm utilizes the nonlinear model of the ASC and the current state to predict the future states of the system within a predefined time horizon. Annamalai, A., et al. [23] also proposes nonlinear MPC control to improve the performance of traditional autopilot deterioration due to disturbances produced by an ever-changing marine environment. A line of sight (LOS) guidance system is used in this research. The proposed controller is compared with the LQG controller in simulation, and showed that the former has the ability to follow the desired trajectory smoothly by the catamaran-type USV called Springer, which has two fixed thrusters. Sharma et al. [24] also used nonlinear MPC control for the Springer USV having two fixed thrusters. In his work, an optimization problem was solved to find open-loop policy of the present state optimally.

A design framework named local control networks (LCNs) for the development of nonlinear autopilot is proposed by Sharma, et al. [25]. His autopilot is for controlling nonlinear yaw dynamics of a catamaran type unmanned surfaced vehicle having two fixed thrusters known as Springer. Simulation results are presented and the performance of nonlinear of autopilot is compared with that of an already existing Springer linear quadratic Gaussian (LQG) autopilot using standard system performance criteria, and it showed that the LCN autopilot perform well. The Springer USV is also studied and modeled as two inputs single output system by Naeem, et al. [15]. The suitable model structure, such as auto-regressive with exogenous input (ARX), auto-regressive moving average with exogenous input (ARMAX), output error, etc., was used for this purpose. The identified model is also applied to a general nonlinear USV model. The genetic algorithm based MPC autopilot was proposed, which utilizes the process model to search for the control moves that satisfy process constraints while optimizing the cost function. The simula-

tions were given, showing the autopilot perform in response to variation in the desired course of the USV. In another work, Naeem, et al. [14] proposed the LQG controller design for the motion control of the aforementioned Springer USV based on the state-space model obtained. In this control design, the LQR and Kalman filter were separately developed and then combined to construct the LQG control. The surge speed and yaw angle were considered in the design and simulations were shown.

Backstepping control technique was also used by Fossen, et al. [5] for station keeping control strategy of an over-actuated ship using nonlinear equations, in spite of the fact that he did not express the environmental disturbances explicitly in the problem formulation stage. Sonnenburg, et al. [8], [12] selected Nomoto's first-order steering planar motion model for the USV, which is a modified rigid hull inflatable boat having automated throttle and steering, at higher speed from different theoretical models after performing experiments over a wide range of speeds and planning conditions. While at low speed of ASV, a first-order lag model for sideslip is included. Their work focused on development, analysis, and experimental implementation of two trajectory tracking controllers, i.e. a cascade of proportional-derivative (PD) controller and a nonlinear controller obtained through backstepping approach. They suggested through experimental results that backstepping control is more effective at tracking trajectories with variable speed and course angle. The work of Sarda, E. I., et al. [9] is related to design and tests nonlinear proportional derivation (PD), backstepping, and sliding mode feedback controller with the addition of wind and without wind feedforward control for station keeping control of a USV in the presence of current and wind. They perform experiments on twin hull configuration of a USV having two azimuth thrusters and a theoretical planar dynamic model considering three DOF, i.e. surge, sway, and yaw, is used; so, for this model, all of the hydrodynamic coefficients are deter-

mined through their respective water tests. The experimental test results are given showing that sliding mode control was found to perform well overall. Dong, Z., et al. [26] proposed a state feedback based backstepping control law for the problem of trajectory tracking of an underactuated USV in horizontal plane. For this purpose, a nonlinear 3DOF dynamic model of USV is considered. Also, for steady state performance, an integral action is included in the proposed control. Lyapunov's theory is used in order to confirm the global stability of the overall system. Simulations of tracking straight line and curve trajectories were carried to demonstrate the effectiveness of the designed control law.

Aguiar, et al. [27], [28] showed an adaptive switching supervisory control along with a nonlinear Lyapunov-based two-dimensional (2D) position trajectory tracking and path-following control law for under-actuated autonomous vehicles so that the problem of global boundedness and convergence of the position tracking error is solved to a neighborhood of the origin. Peng, Z., et al. [29] proposed a robust adaptive steering law to track a heading reference signal with the respective measured signals and minimized the error. They incorporated predictor neural network and a modified dynamic surface controller technique in this work. In their control design, the theoretical nonlinear yaw dynamics are considered, in which the hydrodynamic coefficient are determined in the calm water trials. The USV used in these trials was mono-hull having three propellers and one rudder driven by servo motors. The stability of the closed loop steering law is determined by the Lyapunov's analysis. Simulation and experimental results are given to show the performance of their control in the presence of model uncertainty and measurement noise. The work of Ren, J., et al. [30] explained the application of adaptive backstepping theory to the automatic steering of ships. They suggested the course-keeping adaptive tracking fuzzy control strategy of the nonlinear system of mono-hull ship in strict feedback form. The system has one

propeller and one rudder, so surge speed and yaw angle are considered. The purpose of the control law is that the course of ship has to track output of the given reference model. The unknown system dynamics are approximated by fuzzy system. The simulation results given indicating that the boundedness of all the signals of the close-loop system is guaranteed by the proposed control technique.

Pan, C. Z., et al. [31] proposed neural network (NN) based real-time tracking control of an ASV with completely unknown dynamics and subject to bounded unknown disturbance such as the unstructured, unmolded dynamics. First, the control is derived at the kinematic level and then extended to the dynamic case. A single NN is suggested, and vehicle repressor dynamics formulation was utilized, so off-line training was not performed. The learning algorithm is derived from Lyapunov's stability analysis. Simulation results were given to show the proposed controller is capable of tracking the desired trajectory. Larrazabala, J., et al. [32] used both intelligent and conventional techniques for controlling rudder angle of their USV, which is a scale model of mono-hull marine vessel. The guidance law is also used for calculating the desired angle and then estimate trajectory based on the dynamic model of the said USV. The dynamic model of marine system of Moreno-Salinas was used and the physical and hydrodynamical parameters therein are determined by doing calm water-trials of the USV. An adaptive control law was suggested for any type of trajectory and then gain scheduling approach was used by incorporating PID controllers whose tuning parameters were optimized by genetic algorithm for various operation points. Moreover, a fuzzy logic controller was designed for tackling uncertainties of system dynamics. Simulations were given to depict the control performance and compared with the conventional control.

1.3 System Identification and Control of ODU Autonomous Surface Vehicle

The desired motion control of an ASV is generally obtained by three main motion control schemes, such as station-keeping, trajectory tracking, and path-following; their brief account will be discussed in Chapter 5. In the present research, path-following or way-points tracking control strategy is adopted for the ODU-ASV, which is the spatial tracking resulted basically from the decoupling of velocity and 2D path [4], [33]. The path-following problem of ODU-ASV can be affine formulated and solved as an optimal discrete linear quadratic tracking problem by incorporating full state feedback and the 2D reference path so that the state feedback and feedforward gains are obtained and stored for later use in the control real-time run [34],[35]. The path-following control strategy shows good performance while the position of ASV converges smoothly to the reference path, and less control effort is utilized in this process. As the ODU-ASV is an under-actuated system having two fixed DC trolling motors for its propulsion, so the challenge for under-actuation is to reject almost all the disturbances, which is quite difficult task with only two inputs in the form of two fixed thrusters. Therefore, for this purpose, the optimal discrete LQT control is designed and implemented in real-time for the path-following motion control of the ASV in this research. For the proposed optimal full-state feedback LQT control design and real-time implementation, it is necessary to have a mathematical model in the form of system matrices $A, B, C,$ and D for the ODU-ASV dynamical system. For this purpose, an open-loop system identification technique is used.

The system identification method is *empirical modeling* which has been used in the research of vast areas of engineering and other diverse fields of interest as a viable alternative approach to the traditional *modeling by first principles* for the last three decades. In the latter case of modeling, various laws of physics and energy methods are used to derive ordinary differential

equations (ODE) or partial differential equations (PDE) models for the dynamical or physical systems, while the system identification is a kind of black-box type of approach that pertains to developing models of the dynamical or physical systems from the observed input and output data [36]. System identification is comprised of a large number of linear or nonlinear techniques, which fall generally either in frequency or time domain response of dynamical systems [37], [38]. In this research, the open-loop linear state-space approach of identification called OKID will be used to identify the discrete linear time-invariant (LTI) system of ODU-ASV. The primary purpose of using OKID algorithm is to approximate the dynamics of ODU-ASV accurately with a less complex but more robust mathematical model so that the output residual is minimized substantially. For the very purpose of system identification, the OKID technique will be performed with the help of system observer controller identification toolbox (SOCIT) developed in MATLAB® at NASA Langley Research Center.

The first step in the OKID identification process is the real-time experimental input and output data collected from water-trials performed on the ODU-ASV. Afterwards, the observer Markov parameters (OMPs) will be obtained from the set of Markov parameters of the state-space observer equation. From OMPs, the system Markov parameters and the observer gain Markov parameters will be calculated. Then, the state space model of the ASV will be realized from the system Markov parameters using the Eigen-system realization algorithm with data correlation (ERA/DC), which starts with the formation of generalized block data matrix known as Hankel matrix and later on to factorize it by singular value decomposition. The true system order of ASV is identified by selecting the first significant non-zero singular values from HSV and MSV plots. Finally, the estimate of state-space system matrices of ASV such as $\hat{A}$, $\hat{B}$, and $\hat{C}$ will be obtained; the detailed methodology of both ERA/DC and OKID algorithms are elaborated in

Chapter 3 and Chapter 4, respectively. The accuracy of the identified model for the ASV dynamical will be confirmed by simulations of model output data reconstruction as well as the benchmark residual analysis tests.

The optimal discrete closed-loop LQT control methodology [34], [35] is proposed to be applied to the identified linear model of ODU-ASV so that it follows a desired known reference path over a time interval. The proposed control design methodology will address the problem of determining a control law for the identified dynamical system of ODU-ASV such that a predefined optimality criteria or cost function is attained. In this control strategy, optimal control is determined by combining linear full state variable feedback plus a term depending on reference track. The state variable feedback gains are obtained by minimizing a cost function utilizing weighting matrices, which are found by a multi-objective optimization genetic algorithm (MOGA) technique. The detailed methodology of optimal LQT control and its real-time implementation is given in Chapter 5. The results of simulations and, later on, real-time water-trials for validation will be performed in order to support the validity of the OKID identified model and its proposed control design for path-following motion of ODU-ASV.

1.4 Objective of the Dissertation

From review of the literature, it is concluded that the open-loop OKID method has not been used yet for the system identification of twin-hull ASV having two fixed DC trolling motors with no rudder mechanism using the outdoor experimental input and output data exclusively. Therefore, the key objectives of this research are elaborated as follows.

First, the system identification is performed using observer Kalman filter identification methodology from the experimental input and output data of the required sensors and actuators, which are obtained during outdoor water-trials of ODU-ASV in the presence of stochastic dis-

turbance. In this way, the discrete linear time-invariant state-space model of ODU-ASV is obtained, which is both accurate and conformable to the design of optimal state feedback linear quadratic tracking control. As mentioned earlier, the experimental data contains noise from various environmental disturbances such as wind and tidal currents as well as measurement noise from the sensors imperfections. So, it is the feature of OKID algorithm that the noisy data is discarded while the dynamical system is identified which is represented by the significant Hankel singular values. That's why the OKID technique is better than other identification methods for determining the stochastic systems or processes.

Second, the design and real-time implementation of a viable optimal discrete linear quadratic tracking control based on the identified model is accomplished afterwards. Consequently, the ODU-ASV could properly follow the predefined reference trajectory path described by the X, Y Cartesian coordinates, as way-point guidance coordinates, while coasting at low speed in the presence of varying and uncertain outdoor disturbances such as: wind and tidal currents. The aim is to achieve position tracking accuracy within 1 meter.

Finally, the goal of present research work would be to make the ODU-ASV as predecessor-prototype for the purpose of accurate system identification and control design of a similar kind of a full-scale autonomous or unmanned vehicle in the future research; hence, the civilian as well as military research community may take benefit from the present work.

1.5 Dissertation Structure

This dissertation is presented in accordance with the logical flow in order to describe and explain the various concepts, prior work, methodology, results, conclusion, and future work. So, the detail of present work is written as seven chapters, which is as follows.

Chapter 1 is the introduction which mainly contains the concepts about autonomous surface vehicles, the methodology of system identification and control, prior work, and a general overview of the dissertation.

Chapter 2 describes the overall ASV system configuration, various parts of the control console consisting of on-board computer, microcontroller boards, and different means of communications between on-board computer and the computer of ground control station (GCS). The layout of power supply required for the overall system as well as the ASV propulsion system is also given in the chapter. This chapter also gives a succinct description of the various pose measurement sensors that are vital for autonomous/unmanned control task.

Chapter 3 gives a general description of model, its main classification, and also the state-space form of the model. It then explains the systematic procedure of system identification and the concepts of observability and controllability in detail. The description of Markov parameters (MPs) and its role in the system realization are described. In this chapter, the Eigen-system Realization Algorithm (ERA) as well as the accuracy indicators for differentiating the true modes from noise is explained.

Chapter 4 initially describes some prominent types of model structures and a state-space linear observer. The observer Kalman filter identification (OKID) core-equation and computation of the observer Markov parameters are explained. The process of extraction of both system and observer gain Markov parameters from observer Markov parameters are elaborated; and then, the outline of OKID algorithm is given. The relationship of state-space observer and Kalman filter is also described. Finally, observable canonical-form realization.is explained.

Chapter 5 details the steady-state linear quadratic regulator (LQR) control, its design outline, and its salient properties. Then, the detail description, design outline, and implementation of

main control technique called the optimal linear quadratic tracking control are elaborated, which is used for the control of ASV motion in the present work.

Chapter 6 details the application of OKID methodology to identify the discrete linear time-invariant ASV system using experimental input and output data. The benchmark assessment tests, such as reconstruction of output data during simulation, residual analysis, and cross-validation, for the validity of OKID-identified ASV model, were explained. Afterwards, application of the optimal linear quadratic tracking control for the ASV motion along with the utilization of multi-objective genetic algorithm (MOGA) for the selection of weighting matrices of the said control's cost function is given. The results of controller performance regarding step inputs and various trajectories were also elaborated in this chapter. Finally, the results of implementation and real-time experimental validation of the proposed control along with the application of nonlinear autoregressive with exogenous input neural network (NARX) for the purpose of mapping the control signals are given.

In chapter 7, the conclusive summary of this dissertation along with the recommendations for future research is described.

CHAPTER 2

ODU-ASV SYSTEM CONFIGURATION

2.1 Introduction

The ODU-ASV named "Big Blue" was used in this research in order to perform system identification methodology for achieving the ASV system model and subsequently utilize the identified model to have optimal feedback control design and its validation. This prototype ASV system is a small-sized autonomous surface vehicle made by the ODU-ASV team for the purpose of the association for unmanned vehicle system international (AUVSI) RoboBoat competition. This chapter covers the overall structure of ASV, the required sensors, actuators, and other electronic control console necessary for achieving the above said aim of the research. The design of ASV is based on the flexibility of the modularity design concept, so that the custom-made changes regarding sensors, actuators, and other electrical/electronic components can easily be performed according to the specific needs. In section 2.2, the overall system configuration of ASV is described. The configuration of control console consisting of an on-board computer, microcontrollers, radio controller, and wireless modem is explained in section 2.3. The electrical power system is explained in section 2.4, while the section 2.5 elaborates pose measurement sensors. Lastly, the propulsion system of the ODU-ASV is given in section 2.6.

A pictorial view of Big Blue is given in Fig. 2.1 showing its top flat surface or deck, Inertial Measurement Unit (IMU) and Global Positioning System (GPS), Ubiquiti Bullet M2 networking modem, and control system console consisting of on-board computer, microcontroller boards, and radio controlled (RC) receiver. The MEMSense IMU is installed at CG point of ASV

inside the custom-made acrylic box which is located a little distance away from the electronic control console. The GPS receiver is mounted on a separate raised platform, which is fixed to the PVC pipe installed at the CG point above the acrylic box of IMU.

2.2 Overall ASV System Configuration

The design of the ODU-ASV is based on a custom-made catamaran type configuration made up of small-sized twin parallel hulls held together by a single flat platform or deck. The ASV is approximately 1.43 m long and 0.84 m wide. Its total height up to the IMU and GPS mast is 0.92 m and without mast it is 0.46 m. The deck is covered with a perforated polypropylene plastic sheet. The flat surface of deck is used for housing of electronic parts, computer, networking modem Bullet M2, and the navigational sensors such as IMU and GPS as shown in the Fig. 2.1, which elaborates the layout of all components of the ASV as below.

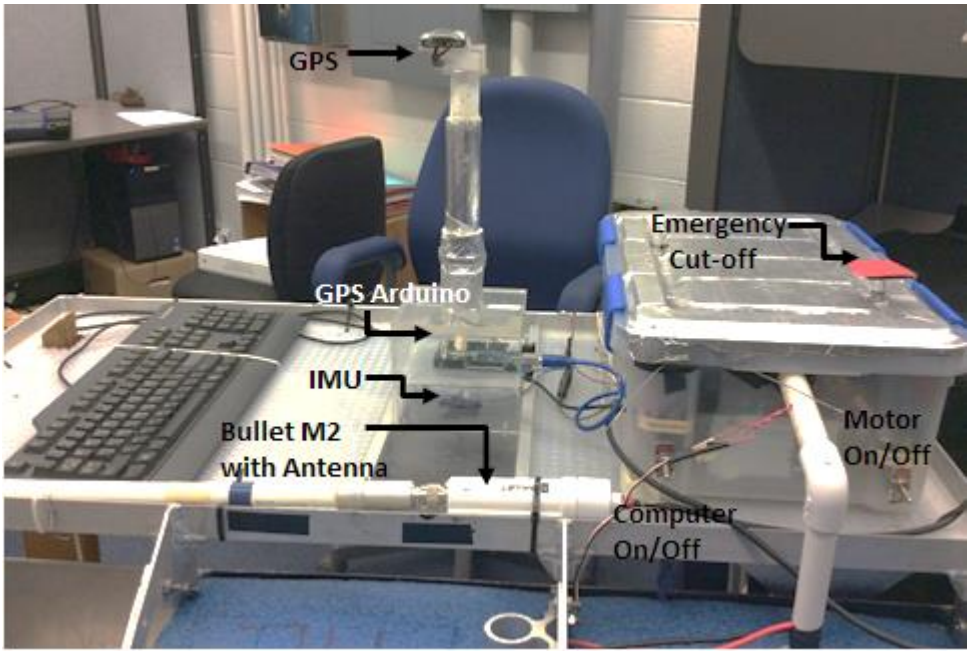

Figure 2.1 Pictorial view of the top flat surface or deck of ODU-ASV.

The required batteries are put inside the hollow body of the two parallel hulls which is covered with the blue Styrofoam sheet, so that the batteries heat can be decapitated with the surface of ASV's hulls, which are submerged externally in the water. The two hulls are streamlined design as demi-hull and built with 1/8" thick aluminum metal. During the motion of ASV, greater stability is attained on account of its low speed as well as dual hulls catamaran-type configuration. Moreover, due to distance between the two hulls, considerable space of the deck can be achieved for other coordinated autonomous tasks, for instance with quad-copter, in the future. One drawback of the catamaran-type design of the ASV, but not pronounced, is that it is less efficient at low speed than a mono-hull design due to more surface area.

2.3 The Configuration of Control Console

A custom-built control console is used for waypoints guidance, navigations and control of ASV. The control console is a $18'' \times 13'' \times 6''$ waterproof Ziploc plastic bin having enough space to have all the required electrical and electronic components such as: on-board computer and its necessary components, microcontroller boards, radio control receiver, electrical fuse blocks, power switches, Bullet M2 Ethernet port, and DAQ module of the MEMSense nIMU as shown in the Fig. 2.2. The control console is secured to the top of the flat surface or deck of ASV and it is interfaced with the required sensors and actuators that have been installed on the ASV. The components of control console are detailed as follows:

2.3.1 On-board Computer

In order for the ASV to accomplish navigation process during real-time water-trials, it is equipped with a stand-alone on-board computer that interfaces with all the Arduino microcontrollers, speed controllers, GPS, nIMU, RC receiver/transmitter, and Wi-Fi networking device. In the navigation process, all the inputs and outputs from sensors and actuators, respectively, were

collected, interpreted, and stored as navigational data into the large computer memory files, which are then used for system identification and, later on, for the control design. Moreover, an on-board computer is also necessary because the data requires processing to make executable control commands for the two fixed DC trolling motors during the water-trials, which involve computationally intensive calculations that couldn't be achieved by the Arduino controllers alone. The computer is custom made from reliable desktop computer components by the ODU-ASV team. The key components are as follows: ASUS motherboard, core-i5 processor with two 2.5 GHz processor, power supply, and hard drives as shown in Fig. 2.2 below.

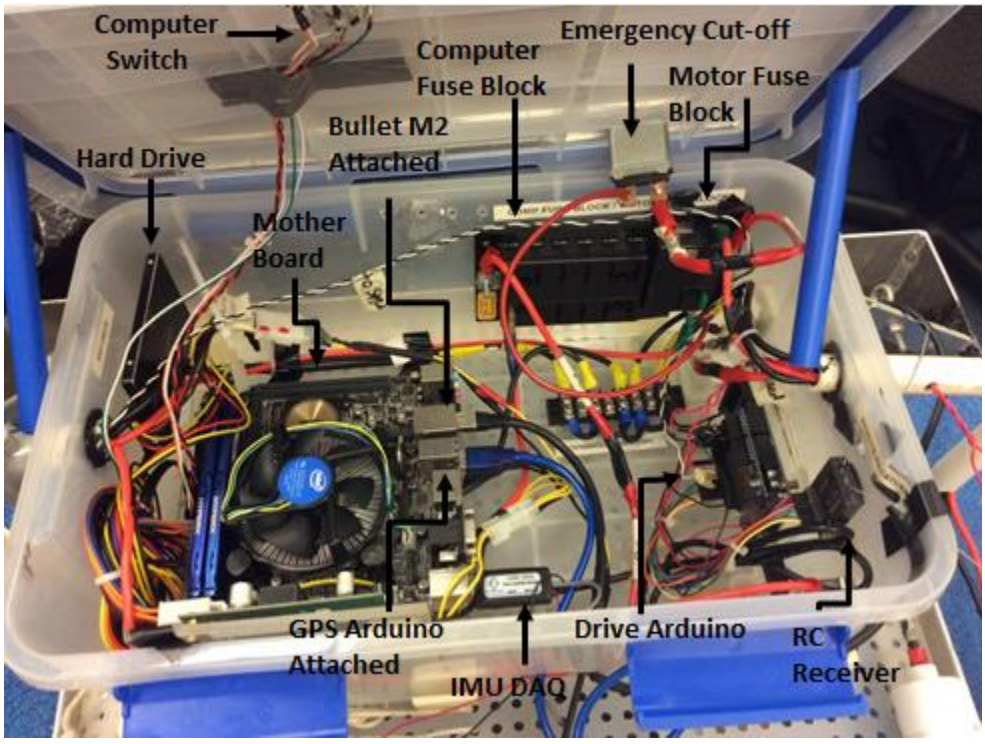

Figure 2.2 The control console of ODU-ASV.

The motherboard has USB ports, a VGA connector, and an Ethernet port, which handle all of the communications between input/output peripheral devices, storage medium, and the processor. The Windows-7 operating system is installed so that the system becomes user-friendly for performing various tasks. The on-board computer can also be remotely controlled from the shore, which means that monitor, mouse, and keyboard are not required for its normal operation. The keyboard is only required for entering the password of Windows operating system.

2.3.2 Microcontroller Boards

The microcontroller boards used in the present work were two Arduino Mega 2560s, which are based on the ATmega2560 and have 54 digital input/output pins for various purposes, a USB connection, a power jack, and a reset button as shown in Fig. 2.3. The Arduino Mega 2560 has everything needed to support the microcontroller for communicating with on-board computer or another microcontroller connected to it with a USB cable or though digital input/output pins. The open-source Arduino software, known as IDE, is used to write codes and upload them to the board for performing various tasks. The written code can run on the Windows, MAC OS X, and Linux operating systems. One of the Arduinos, called *GPS-Arduino*, and connected to GPS receiver, is used for the position data logging of ASV with the help of navigation code written in Arduino IDE. The GPS data in the form of latitude and longitude is transmitted to MATLAB® of the on-board computer by serial communication having baud rate of 38400 bps during the real-time control run; where it is converted to X and Y Cartesian coordinates expressed in metric units, i.e. meters, in order to be used as the position states of ASV. Another Arduino, called *Drive-Arduino*, is used for the actuation and controlling of the two Watersnake trolling motors. The Drive-Arduino is interfaced with both the control module on the main on-board computer as well as hand-held remote control unit so that it receives the required pulse

width modulation (PWM) commands from them and send those commands to the speed controller of two fixed DC trolling motors to rotate accordingly. Hence, the Drive-Arduino performs three main functions; first, it manages to toggle the ASV motor power between on and off. Second, it switches the ASV between autonomous and remote control modes. Third, it receives inputs PWM values from the on-board computer and sends them to the speed controllers of the two DC trolling motors for thrusting purpose.

Figure 2.3: A typical Arduino Mega-2560 microcontroller board.

2.3.3 Radio Controller

The hand-held radio controller (RC), consisting of a transmitter and a receiver, is required for controlling the boat manually from remote distance as well as for switching between the *manual* and *autonomous* modes. This functionality is achieved with the help of Futaba 6-channel transmitter and 7-channel receiver radio control system; each of them has frequency 2.4 GHz as shown in the Fig. 2.4 below. These RC channels consist of four toggle switches and two bi-directional joystick controls. The two out of four toggle switches were utilized for control

Figure 2.4 Futaba RC transmitter and receiver.

purpose; one toggle switch, which is connected wirelessly to one of the channel on the receiver, is programmed as on/off switch of the ASV motors. Another toggle switch is used for switching autonomous and remote control modes. The two bi-directional joysticks control the two fixed motors during RC mode in order to produced thrust levels required for both propellers attached to the motors. The receiver sends PWM signals to the speed controller according to the position of the respective channel on the transmitter. All the receiver's channels are attached to the Arduino microcontroller board which is already programmed to interpret the signals from the receiver and then control the motion of two DC trolling motors accordingly. At any point of the ASV's waypoint navigation, the RC system overrides the control values sent from the control module during autonomous mode so that emergency control of the ASV is achieved in case of unpredictable mishap. Also, the ASV motion can be terminated by pressing the emergency cut-off push-button on the control console.

2.3.4 Wireless Modem

The remote communication connection of the laptop computer of ground control station (GCS) and the on-board computer of ASV is an essential prerequisite so that the autonomous and remote control tasks can easily be performed and so that access to all of the required signals is achieved. For this purpose, a pair of wireless modem devices called Ubiquiti Bullet M2, as shown in Figure 2.5, is used to communicate the sensors and actuators data between the above said two computers. In this way, the maneuvering tasks of the ASV can be continuously monitored remotely from the site of GCS. The two Bullet M2, one at the GCS serving as router and another on the ASV working as receiver, with their two omni-directional outdoor 12dBi antennas offer a large Wi-Fi range. These antennas can communicate with up to a range of 1.6Km by utilizing its maximum power of 63 mW. The two powered antennas of the above said Bullets are configured to form a wireless bridge in order to connect with each other and hence this bridge setup acts as wired LAN connection. The recommended DC voltage range of this wireless system is 10-24V.

Figure 2.5 Ubiquiti Bullet M2 with antenna.

2.4 Electrical Power System

For all the electrical devices installed on ASV such as: on-board computer, DC trolling motors for two thrusters, speed controllers, RC receiver, GPS receiver, IMU, Ubiquiti Bullet M2, and Arduino microcontrollers to perform the necessary autonomous tasks, a reliable power source is required. This power source is in the form of two rechargeable lead-acid batteries of 12V & 8Ah which are placed in the hollow compartments of twin hulls of the ASV as well as a separate 7.4V Lipo battery. The first 12V lead-acid battery provides power to two thrusters only, because they consume more power for propelling the ASV on water surface as compared to all other electrical devices. The second lead-acid battery is used to power the on-board computer along with its attached peripheral devices.

The power used by the Nano IMU is provided by a third Lipo battery. The flow diagram of power distribution for the ASV system's basic functioning is shown in Fig. 2.6 below. The speed controllers as well as some of the electronics draw high current occasionally, so fuse blocks, as shown in Fig. 2.2 of section 2.3, were used in order to protect some of the circuits in case of power surge. One of the fuse blocks consists of two 30 amp fuses used for both speed controllers, while another fuse block has a 25 amp fuse used for the on-board computer. Both the lead-acid batteries share a common ground with the positive leads going to each fuse blocks.

2.5 Pose Measurement Sensors

The position and orientation of the autonomous vehicles with respect to the NED, Geo-detic, and/or body coordinate systems is known as its pose. Nowadays, with the advancement of Micro-Electro-Mechanical Systems (MEMS), robotic or autonomous vehicles are equipped with many state of the art and miniature sensing systems. A sensing system is comprised of one de-vice or a group of devices that senses the stimulus of a physical process or phenomenon and

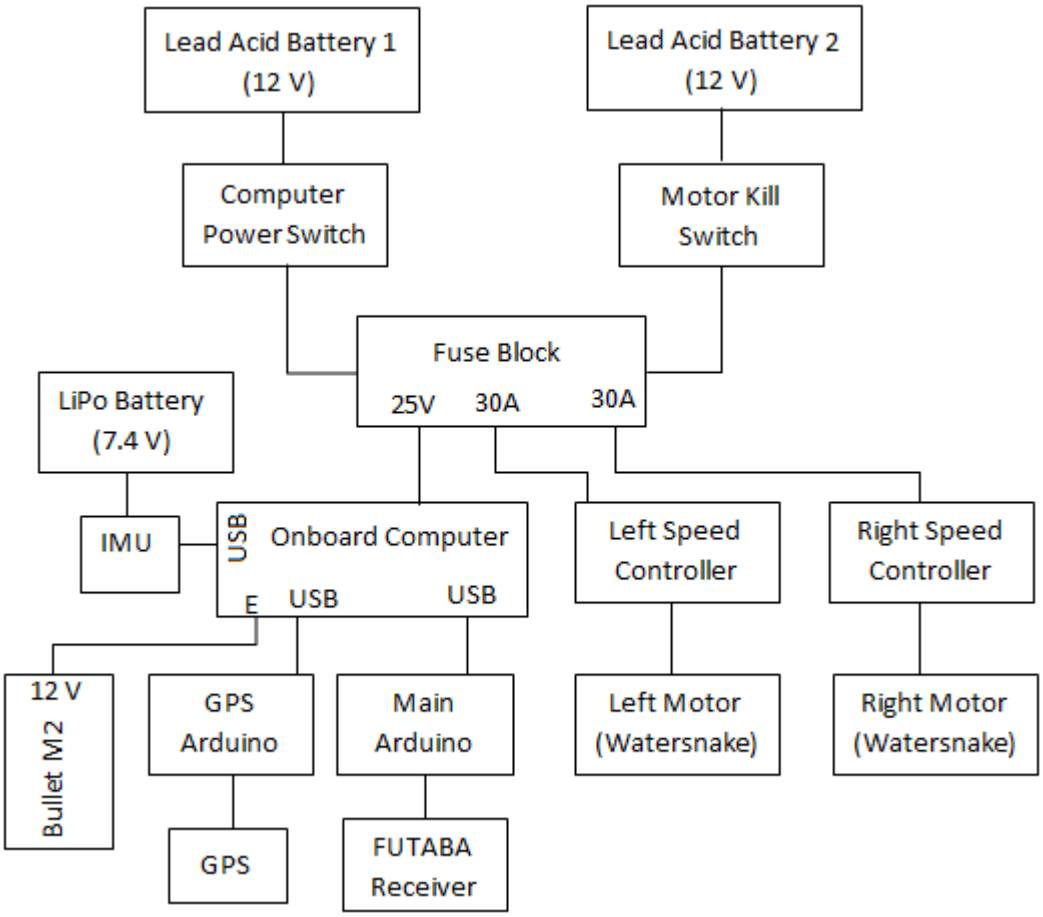

Figure 2.6 Flow diagram of electrical power distribution.

converts to signals having physical units. In order to accomplish the state estimation task of the autonomous vehicles, various common techniques are used for measuring raw data of the pose of these vehicles. These techniques are mainly based on the on-board sensors such as: laser sensors, accelerometers, gyroscopes, magnetometers, ultrasonic sensors, static/dynamic pressure sensors, odometers, and GPS receivers. Moreover, advanced actuator systems are being used for forward/reverse motion and steering maneuvers of these vehicles. In this section, various types of sensing and actuation systems that are installed and used for the ASV have been overviewed ra-

ther than in detailed study of the different underlying technologies incorporated in such systems.

2.5.1 Laser Measurement Device

Laser measurement devices, also known by the acronym LIDAR (Light Detection and Ranging), are generally the most robust sensors which are used for distance measurement, object detection, area monitoring, and mapping purposes. Laser scanners measurements are based on the time-of-flight principle. A beam of laser pulse with defined duration is directed towards a confronted object, and a fast counter is started. After being reflected by the object, the pulsed laser beam is deflected by an internal rotating mirror and then detected by a photodiode. The photodiode converts this beam into signals in an optoelectronic circuit. The counter is stopped at the time of the first return of a signal and this process continues. Hence, the time interval lapsed between transmission and reception of the reflected pulse of light is directly proportional to the distance between the laser scanner and the object that reflected the light. The light traverses approximately 15 cm from the source until it strikes upon the object and traverses the same distance back towards the receiver within 1 *nanosecond* [39].

The position of ODU-ASV in the NED coordinate system is determined using a laser scanner called the SICK laser Measurement Sensor (LMS) 200 as shown in the Figure 2.7. The position coordinates of ASV are calculated based on the distance and angle measurements of the reflected laser beam of SICK LMS 200. The angular resolution of SICK LMS 200 in the radial field of view is up to minimum of 0.25 degrees. Its typical accuracy is 35 mm for a range of 20 mm or more as far as 80 mm depending on the reflectivity of the target object and transmission strength of the SICK LMS 200 laser scanner. Due to good reflectivity of 110 to 150%, the aluminum foil was wrapped around a 1" PVC pipe and the pipe was installed on the center of gravity (CG) point of ASV during outdoor testing. With the help of this technique, the accurate position

of ASV is recorded within a circular area of 8 meters radius. Hence, the ASV dynamics could easily be tested and system identification runs are performed for a total length of 16 meters. The main parameters of SICK LMS 200 are highlighted in the Table 2.1 below.

Table 2.1 Specifications of SICK LMS 200

Measurement range	8 m, 16 m, 32 m, 80 m
Field of view	180° /100°
Resolution	10 mm
Angular resolution	1° /0.5° /0.25°
Response time	13ms /26ms/53ms
Update rate	75 scans of 180° sweep per second
Data transfer rate	9.6 /19.2 /38.4 /500 k Baud
Systematic error	±15 mm
Stochastic error at 1 σ	5 mm
Supply voltage	24 V ±15%

Figure 2.7 A typical SICK LMS 200 Lidar.

The main disadvantage of SICK LMS 200 is that its measurements are sometimes deteriorated by the adverse outdoor conditions such as fogginess and dazzling sunlight. Therefore, the SICK LMS 200 is best suited to use for indoor measurements, but it was used outdoors with good quality of data collection because on the test day the sky was partly overcast giving perfect daylight condition. The communication of LMS 200 with the connected computer is achieved through either RS-232 or RS-422 interface. The RS-422 being differential interface circuit employ the differential mode of operation that allows each signal to be split into two different wires, configured in opposite voltage states. Due to this interface configuration, substantially faster data transfer speeds can be achieved than that of the RS-232. For this purpose, an additional high-speed serial interface card was purchased and utilized.

2.5.2 Global Positioning System Receiver

The positioning and navigation technology is enormously versatile and being widely used in different kinds of applications around the world, namely guidance and navigation systems for autonomous vehicles, navigation systems for passenger airplanes and ground transport vehicles, surveying, agriculture, and military. The Global Positioning System (GPS) is a Global Navigation Satellite System (GNSS) and has been proving itself as a reliable and accurate method used for the positioning and timing information necessary for navigation applications over the last couple of decades. The low-cost 3DR uBlox LEA-6H GPS with compass module was used in the present work for logging the position data of the ASV in geodetic coordinates, i.e. latitude and longitude, only during the real-time validation water-trials of feedback optimal LQT control.

The 3DR GPS with compass module is in factory enclosed protective case. It is mounted on a custom-made small acrylic platform which is fixed to a PVC mast at about 40cm height from the deck of ASV with the arrow of GPS facing forward. The GPS mount is at a reasonable

distance with a clear view of the sky. So, in this way, the GPS module does not interfere with the magnetic fields of the motors, speed controllers, DC power wiring, batteries, and the nearby iron containing metallic objects to avoid possible magnetic interference as well as to ascertain the unobstructed view of the sky for maximum satellites constellation. The GPS mounting position is depicted in the pictorial view of Fig. 2.1 given in section 2.2 above. A custom design TTL (transistor-transistor logic) serial cable is made, which connects the GPS to the Arduino Mega microcontroller board with the communication baud rate of 38400 bps. Its specified data update rate is 5Hz, but when the position data is transferred from Arduino to the MATLAB® programming environment, then data sampling rate reduces to 3 Hz. This sampling is then set as the minimum sampling rate for the system identification of ASV and all other sampling rates of the sensors are adjusted to it accordingly. The voltage rating required for the GPS is 2.7V - 3.6V.

The 3DR GPS delivers position and status data in National Marine Electronics Association (NMEA), UBX binary, and RTCM protocols. An Arduino-C navigation code was used for the GPS data logging so that it decodes the position and status data of the ASV from the telemetric sentence defined within the binary protocol of the GPS receiver into the latitude and longitude values with the help of GPS Arduino. Then, the decoded data in the form of latitude and longitude position data is sent to the MATLAB® programming environment of the on-board computer; where it is transformed to X and Y Cartesian coordinates in metric units. The X and Y position data are then used by the designed feedback control module to compute the required control commands for controlling the two trolling motors of ASV. The 3DR GPS provides rather reasonable position accuracy in static and slow moving applications, and that can only be expected in an environment with unobstructed sky view. In outdoor experiments, all the water-trials were performed by the ODU-ASV in an open environment with an unobstructed sky view.

The GPS takes a little while in order to update all the satellite positions and then settle in on stable reading. The average position accuracy of 3DR GPS is observed to be approximately 0.8 to 2.5 m.

2.5.3 Inertial Measurement Unit

An Inertial Measurement Unit (IMU) is MEMS self-contained measurement system comprised of sensors' triad such as accelerometers, gyroscopes, and magnetometers that renders inertial measurement data consisting of angular velocity, linear acceleration, and magnetic field. The estimate of angular rates and the corresponding orientation of the ASV expressed as in Euler angles such as roll, pitch, & yaw are obtained by using the Nano IMU (nIMU) in this research. The nIMU is a small-size and light-weight MEMS measurement system which is manufactured by MEMSense [40] as shown in Fig. 2.8 below.

2.5.3.1 IMU Data Acquisition

The nIMU gives inertial data of 3D acceleration, 3D angular rate, and 3D magnetic field along with the temperature data through RS-422 serial communication protocol. The digital data outputs are compensated for the sensitivities of operating temperature to bias and scale factor. For the purpose of communication with nIMU, it is simply connect to its USB interface which is in turn connect to the computer. A manufacturer's USB RS-422 data acquisition (DAQ) module is used for the data communication to computer with the baud rate of 115200 bps. The RS-422 USB module incorporates virtual COM port drivers which are compatible with typical COM port communication protocol. The data format of RS-422 connection of IMU is factory configured as 8-bit UART with 1 start bit, 8 data bits, and 1 stop bit. The sampling rate of the inertial data provided by the IMU is 150 Hz, which is three times higher than its bandwidth i.e. 50 Hz. The software used for acquiring inertial data for the system identification task is known as Inertial Insight

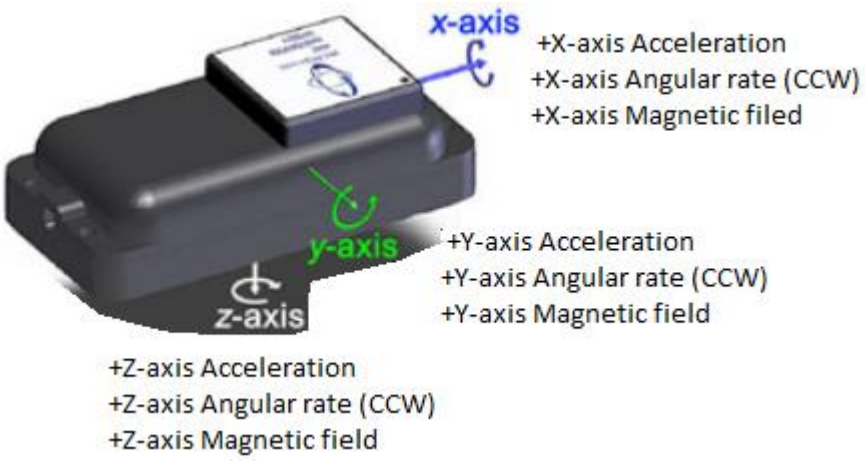

Figure 2.8 A typical MEMSense nIMU unit with the coordinate system shown.

software provided by the MEMSense which is the GUI-based menu-driven application; while a MATLAB® code is used for data logging from the nIMU during the real-time water-trials which are performed for the validation of the designed optimal discrete LQT controller. The main manufacturer specifications of nIMU are tabulated in Table 2.2.

2.5.3.2 Tilt Compensation

The heading or yaw angle ψ can be accurately determined through dual-axis magnetometer and tri-axial accelerometer sensors of the nIMU. In order to implement the compass system correctly, it must compensate for the effect of pitching and rolling angles, as well as try to calibrate out soft and hard iron effects [41], [42], [43]. The MEMSense IMU is installed at the CG point inside the custom-build acrylic box, which is rather far away of the electronic control console. Also, the top working platform of ASV is covered with the perforated polypropylene plastic sheet; so, it is assumed that there are no soft and hard iron effects which might distort the Earth's magnetic field and they don’t need to be calibrated out. Hence, according to right-hand rule,

Table 2.2 Specifications of MEMSense Nano IMU

Size	$1.84 \times 0.9 \times 0.56$ inches
Mass	20 gram
Angular rate dynamic range	$\pm 1200\ ^{\circ}/s$
Accelerometer dynamic range	± 10 g
Magnetic field dynamic range	± 1.9 gauss
Update rate	150 Hz
Input voltage	$5.4 - 85$ VDC
Supply Current	140 mA
Working temperature	$0 - 85\ ℃$

positive roll, pitch, and heading angles are counter-clockwise when looking along x-, y-, and z-axes of rotation, respectively, towards the origin as shown in Fig. 2.8 above. The orientation is assumed to be calculated relative to NED reference frame, where the reference frame axes are considered in uppercase having the subscript r, such as X_r. The details of the NED reference co-ordinate system are given in section 6.3.2. The gravity is incorporated in calculation of pitch and roll angles which exerts a constant acceleration of $1g$ downwards. Therefore, pitching angle is calculated following a negative rotation X-Z plane by the arc-tangent function as shown in the Fig. 2.9; the details for calculating pitch and roll angles are given in [41], [42], [43]. Gravity can't be used to calculate change in heading angle, so the magnetometer data is used instead. In this regard, the arc-tangent method is applied to the raw magnetometer data. However, errors are got into the heading data when the magnetometer's sensitivity decreases due to the increase in pitch and roll angles [42], [43]. The heading can be corrected by re-aligning the local z-axis with the Z axis of reference frame. Hence, the correction to the magnetometer's data is achieved by

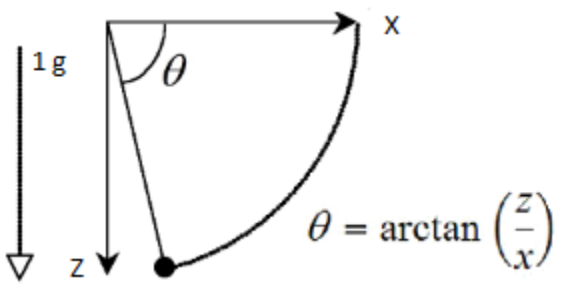

Figure 2.9 Calculation of elevation or pitch angle (θ).

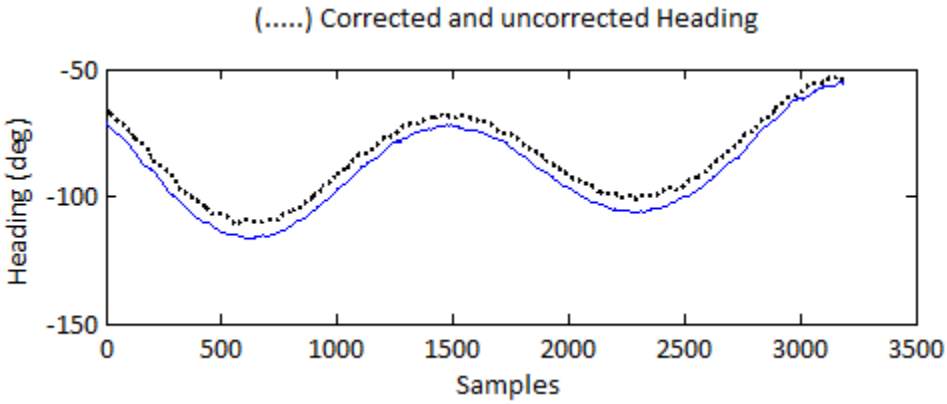

Figure 2.10 Tilt corrected heading angle data from a typical water-trial.

applying first a rotation, which removes the roll angle, and then a second rotation that removes the pitch angle, and consequently the local x-y plane will be re-aligned with the X-Y plane of the reference frame. The correction process of heading is done in MATLAB®, and the corrected and un-corrected heading angle data from one of the water-trials is shown in the Fig 2.10.

2.6 The Propulsion System

The propulsion system of the ODU-ASV consists of Kort nozzle system, i.e. a type of ducted propellers system, in which two propellers attached to the two trolling motors are sur-

rounded by lightweight and non-rotating shrouds made up of ABS plastic as shown in Fig. 2.11. The shroud is designed hydro-dynamically and wraps around the propeller blades, thereby increasing the low speed thruster performance like in the case of the ODU-ASV. The cross section of shroud towards the fore of the ASV is slightly wider than the one towards the stern side; therefore, more water is sucked into narrower space and passed faster through the propeller blades [44]. In this way, the overall efficiency of the propellers is improved that entails high thrust is achieved at very low speed, e.g. less than 5 knots. The ASV has two custom-designed propellers

Figure 2.11 The ASV propellers and shrouds system.

each of them has four blades. The actuators, based on their energy source, are mainly differentiated into electrical, mechanical, piezoelectric, and pneumatic or hydraulic actuators. The motion of ODU-ASV is attained by employing two Watersnake trolling motors as electrical actuators, each of them operates at variable speed and produces $18\,lb$ thrust in either forward or reverse direction.

As the ODU-ASV is rudderless; so, the heading is attained by the difference between two

motors' thrusts at any instant of time. The Watersnake trolling motors are installed below the twin-hulls and equally away from the midline at stern of the ASV. These actuators are excited by the commanded signals in the form of PWM through Arduino microcontroller board in order to produce thrust. The voltage and current rating required for each motor is $12\ V$ and $7 - 15\ Amps$. The Victor SP speed controllers, as shown in Fig. 2.12, having nominal voltage of 12V are placed inside the twin-hulls. They control the input voltage to motors at a specified time. The speed controllers read the PWM signals received from the microcontroller, which has been named as *Drive-Arduino*, connected to the on-board computer and output current to the motors thereby altering their speed and direction; hence, in this way, the required dynamics of ASV is achieved. The PWM values range from 40 to 160 duty cycles, 90 being the neutral value at which each trolling motor stops rotating. The range of PWM values above neutral value, i.e. 91-160 duty cycles, are used for forward motion; while the PWM values below neutral value, i.e. 40-89 duty cycles, are used for the reverse motion of ASV. The cruise speed of ASV is calculated to be approximately 1 m/s.

Figure 2.12 A typical Victor SP speed controller.

CHAPTER 3

TIME-DOMAIN IDENTIFICATION OF STATE-SPACE MODELS

3.1 Introduction

In today's fast growing digital computational era, dynamical systems are being analyzed and their inter-variable relationships are being dug out for the identification, prediction, control, and monitoring of these systems. For this purpose, the mathematical description known as *model* of the dynamical system under study is predominantly required to be developed. In order to develop a mathematical model for a physical or dynamical system, the system identification techniques which incorporate the observed/experimental input output data obtained from the system under study are used. Among the vast range of models for dynamical systems, *state-space* (SS) descriptions gradually getting attention for system identification, signal estimation, as well as multivariable control [45], [36], [46]. So in this research, state-space system identification will be incorporated to identify the system of small prototype ODU autonomous surface vehicle.

The organization of the topics covered in this chapter is as follows. First, the model definition and a brief and concise overview of the various types of models are given. Then, the concept of system identification and its systematic procedure is discussed. In section 3.4, a general linear discrete time-invariant state-space representation for a MIMO system and its response to general input are described. In section 3.5, the key concepts of state-space identification, i.e. observability and controllability, are briefly explained. Following in section 3.6, the building blocks of system realization called *Markov parameters* and the basic realization theory along with the *Eigen-system Realization Algorithm (ERA)* are presented. Some other accuracy indicators such

Modal Amplitude Coherence (MAC) and Mode Singular Value (MSV) for the distinguishing the true modes from the noise are also briefly explained to culminate the chapter.

3.2 Model of a Dynamical System and Its Classification

The model, a mathematical abstraction of the dynamical system or physical process, is elaborately defined as "the set of linear or non-linear, differential or algebraic- difference equations comprised of the system/process state variables, inputs, and outputs, which emulate the dynamic system or process in the course of time for a given set of operating conditions and the underlying assumptions". The model is both the final product and the central part of system identification practice. The criterion of "good" quality model is usually based on how well the model fulfills the purpose of fitting the measured data [36], [46].

Models are classified primarily on the method of identification used for modeling itself, i.e. models developed through *first-principle* vs. models developed through *empirical* methods. Fundamental Laws of Physics are used to obtain the first-principle models which are generally causal, continuous, non-linear differential-algebraic equations. Empirical models are obtained using the measured input output data, so minimal knowledge of system dynamics is needed in this regard. Further divisions within these two main classes are made based on system characteristics, available knowledge, domain of modeling, and response characteristics. The categorization of the models is by no means hierarchical; so any class of model could be chosen and further sub-divided into the other nomenclature of the model's types which are briefly described below.

The empirical models are further classified as *parametric* vs. *non-parametric* models as well as *black-box* vs. *grey-box* models. The parametric models have a specific structure, which is characterized by delay, order, and a set of parameters, while non-parametric models don't have specific structure to be assumed by the user but are usually described by responses. In black-box

modeling, there is no one-to-one correspondence between the structure and/or parameters of the model with the physical properties of the system. It is a good option to use the black-box models when the purpose of the model is only prediction. The grey-box modeling is the emerging area of identification which depends on the availability of a priori knowledge to the user. In grey-box models, the constraints are imposed on the structure of the model prior to and/or during the estimation process in order to interpret the physical laws involved in the system.

The classification of *linear* vs. *non-linear* model is based on the assumption of linearity conditions, i.e. homogeneity (scaling) and superposition, to be satisfied by the system. The system which fulfills those conditions is called linear system while the system which does not satisfy is called non-linear system. Next, the nature of domain of inputs, outputs, and states specify the model type that whether the model is built in *time-domain* (also known as raw-domain) or in *frequency-domain* (also known as transform-domain). The particular instance of domain-based modeling is the *continuous-time* or *discrete-time* case. All the physical quantities such as displacement, velocity, and acceleration are varying continuously in time and are measured by sensors to generate analog signals. These signals can be converted to discrete-time when sampled by A-D converter. Empirical models are generally constructed in discrete-time domain.

The *deterministic model* predicts the response of the system whose underlying physics is precisely known. The input profile is also known in this case. On the other hand, the *stochastic model,* which is also known as time-series model, is constructed on the basis of statistics and probability. The response of the stochastic model to the random or stochastic inputs can never be accurately predicted but rather described as a good fit to the measured data in statistical sense.

The *time-invariant* model relates to the system whose features do not vary with time. This means the system produces the same output by applying the same input irrespective of

when the input was applied. On the contrary, the output of the *time-varying* model changes with the passage of time even if the same input is applied. If the output variations of the model is just a function of one dimension such as time, then this model belongs to the family of *lumped parameters* models; while the variations of output of the *distributed parameters* model also depends on more than one dimension. The estimation of lumped parameters model is easier than the distributed parameters model because the former requires few parameters for their structure.

Additionally, there are *input-output models* and *state-space models* based on the mathematical abstraction of the dynamical system. State-space modeling is the most common class of representation in modern system and control theory because in this class of modeling approach the dynamics of a system is characterized by a set of either tangible or hidden variables known as *states*. The state-space models got some advantages over the input-output models in terms of numerical efficiency and order determination. In this research, however, the focus will be on the identification of the discrete linear time-invariant (LTI) dynamic state-space model of the ODU ASV system.

3.3 System Identification and Its Systematic Procedure

Development of a mathematical model with desired accuracy for a system is the main goal of system identification. So, system identification is defined as the process of developing a mathematical relation between the inputs and outputs of a system or process based on the measured data. The following objectives are generally the scope of the LTI system identification:

(i) Describe various deterministic and stochastic structures used for the identification of LTI systems.

(ii) Characterize the parameterization, bias, variance, and consistency of the dynamical process or system under study.

(iii) Discuss both classical and modern estimation methods for identifying optimal models, such as: Maximum Likely Estimation (MLE), Bayesian, and Least Squares (LS).

(iv) Using methods of prediction-error, observer based Kalman filter identification, and sub-space identification for identifying a model for a system at hand.

(v) The assessment of the goodness of estimated models by incorporating statistical techniques.

The process of system identification can be divided systematically into the following main steps, which are also depicted in Figure 3.1 below.

3.3.1 Data Generation and Acquisition

The basic requirement of system identification is the type of input excitation and its effect on the measured input. The input excitation should be such that its effect is more pronounced in the output rather than the effect caused by sensor noise and environment disturbances. Another important issue is to choose a suitable *sampling rate* so that there is no loss of information.

Informative data for system identification depends critically on the sampling rate of the data. If the sampling interval denoted by Δt is too much larger than the high-frequency content in the signal will look like low-frequency content; this phenomenon is known as *aliasing*. The frequency at which aliasing takes place is known as *Nyquist-frequency*. Hence, a continuous or discrete-time signal having frequency F should be sampled as fast as at least two times of its frequency in order to avoid the loss of information therein; i.e. $|F/F_s| \leq 1/2$ where F_s is the sampling frequency [38].

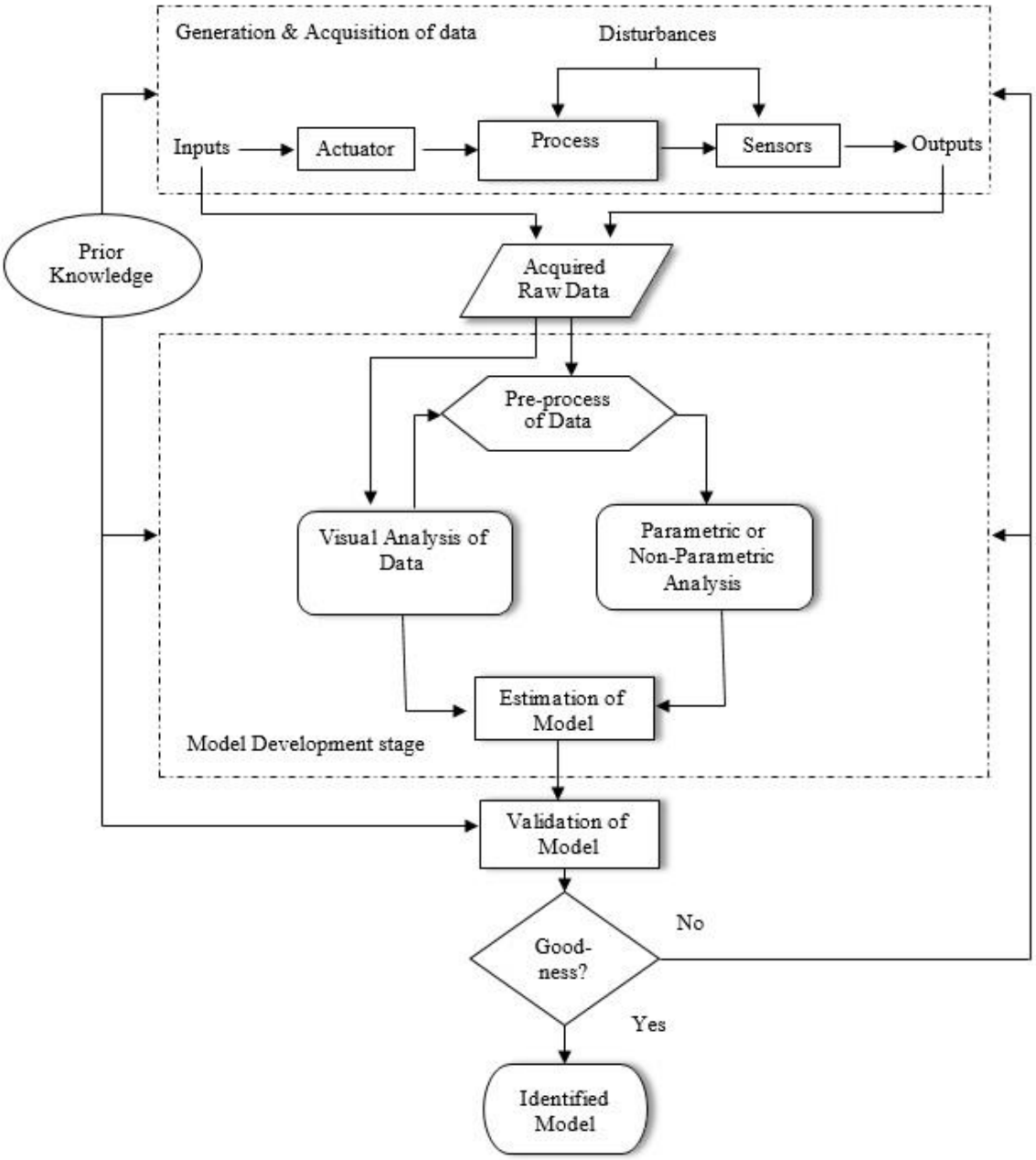

Figure 3.1 A generic layout of system identification procedure.

3.3.2 Pre-processing of Data

In addition to the noise, raw data is generally associated with outliers, missing data, and drifts that affect the data quality. Outliers are the data which don't conform to the remaining data chunk and they are come up from abrupt excursion in the process or sensor malfunctioning. Similarly, the missing data is resulted from the irregular sensor malfunctioning, non-uniform sampling, and losses in the data transfer. In order to perform the model estimation algorithm effectively, the raw data undergoes quality inspection and pre-processing by utilizing good statistical methods such as ML-based expectation maximization and the multiple imputation methods. The data pre-processing can also be facilitated by selecting reliable instrumentation and data acquisition system as well as careful design of experiment.

3.3.3 Visual Analysis of Data

Visual qualitative analysis of the data at every step of system identification process is the key requirement for information extraction. A visual scrutiny of the data helps in identifying the presence of outliers, drifts, and missing data before performing pre-processing step. Also, the visual examination of the data in the model development stage of the identification process is useful in the selection of model structure and its order as well as in model validation.

3.3.4 Model Development

Development of a useful working model of dynamic system is the main motive of the process of system identification. The step of model development is further categorizes into two steps. First, determination of suitable model structure and the order based on end use, prediction accuracy, prior knowledge, and physical insight. Second, estimation of the required parameters of the very model is done by minimizing the objective function of the prediction errors. Methods based on LS minimize the squared 2-norm, called the *Euclidean distance*, between the predicted

and observed values of data. On the other hand, objective functions of prediction errors are formulated based on maximum likelihood principle in the stochastic case of modeling.

3.3.5 Validation of the Identified Model

In order to have a good model, it is essential to perform the integral part of the system identification i.e. the quality assessment and validation of the identified model through the statistical analysis of the *residual* or *prediction error* and *cross-validation*. In the statistical analysis, one has to check any residual information left over for the model to capture; while in cross-validation, one has to determine that whether the identified model has captured the global characteristics of the dynamic system or process by investigating the response of the identified model on a fresh test data set.

3.4 Development of State-Space Model

As discussed previously, there are basically two ways to represent the dynamic systems. First, the fundamental physical laws lead the modeling of the dynamical systems to differential or difference equations which are parameterized by coefficients relating directly to the physical component of the system. Second, dynamical models are represented in transfer functions obtained through identification from input and output measurements either in the continuous *Laplace* domain or in the discrete *z-domain*. Yet, another more convenient form of representation of the model of dynamical system exists, which is known as the *state-space* model. In this type of modeling representation, the second-order equations of motion for a general multi-degree-of-freedom system are described by the first-order strictly causal matrix differential/difference equation. In essence, the state-space model relates the inputs and outputs of a system with the help of intermediate variables called the *state variables*. These are the minimum set of variables of a dynamical system equal in number to its order and whose present values together with the

known control input values completely predict the future response of the dynamical system. The multi input and multi output (MIMO) systems can be described with elegance and simplicity with the help of state-space descriptions. That's why they are usually used in modern control because it is very easily implemented with the help of modern-day digital computers having fast processing speed. For the sake of brevity, the MIMO discrete-time state-space model of a dynamical system, having r inputs and m outputs, is described below. However, the same approach can easily be used for single input and single output (SISO) systems in either discrete or continuous-time domain.

3.4.1 Discrete-Time Representation of State-Space Model

A generic finite-dimensional discrete LTI dynamical system can be described by the discrete-time state-space model as shown below [38].

$$x(k+1) = Ax(k) + Bu(k) \tag{3.1}$$

$$y(k) = Cx(k) + Bu(k) \tag{3.2}$$

where $x(k)$, $u(k)$, and $y(k)$ are the state $(n \times 1)$, control $(r \times 1)$, and output $(m \times 1)$ vectors respectively. The dimensions of constant matrices as in above equations i.e. $A, B, C,$ and D are $(n \times n)$, $(n \times r)$, $(m \times n)$, and $(m \times r)$ respectively; which map the inputs to outputs through a discrete-time state vector $x(k)$. In practice, the experimental data are discrete in nature due to sampling the analog signals such as: displacement, velocity, and acceleration into digital signals by analog-to-digital (A-D) converter. Therefore, the set of equations i.e. Eq. (3.1) called state equation and Eq. (3.2) called output equation constitute the foundation for the state-space system identification of a linear, time-invariant, dynamical systems of the form (3.1) and (3.2) directly from the observed data [38], [47]. For instance, the algorithm of some popular methods such as *observer Kalman filter identification (OKID)* and *subspace identification* known as the N4SID

are used for this purpose. The OKID algorithm is the subject of this research and will be discussed in the forthcoming Chapter 4.

3.4.2 Dynamic Response of a Discrete-time State-Space Model

Owing to the fact that integration action is already built into the discrete-time state-space model, it can be simply used to simulate the free response as well as the forced dynamic response to a general input $u(k)$. Let the initial conditions of the state $x(0)$ and the input profile are given, the state at each consecutive instant for $k > 0$ can be computed as follows:

$$\begin{aligned} x(1) &= Ax(0) + Bu(0) \\ x(2) &= Ax(1) + Bu(1) = A^2x(0) + ABu(0) + Bu(1) \\ &\vdots \qquad\qquad \vdots \qquad\qquad \vdots \\ x(k) &= A^kx(0) + \sum_{l=1}^{k} A^{l-1}Bu(k-l) \end{aligned} \tag{3.3}$$

The output response at any instant becomes

$$y(k) = CA^kx(0) + \sum_{l=1}^{k} CA^{l-1}Bu(k-l) + Du(k) \tag{3.4}$$

Eq. (3.4) indicates that stability of the dynamic system is associated with the behavior of the matrix exponential A^k. It means that a discrete LTI system is asymptotically stable if and only if the absolute value of the magnitude of all the eigenvalues of matrix A is less than unity, i.e., $|\lambda_i(A)| < 1 \quad \forall\ i$. Also, Eq. (3.4) is the starting point for deriving observability conditions which is the key concept of state-space system identification.

3.5 Observability and Controllability of Discrete-Time Systems

The application of system identification and hence optimal control rely equally on the concepts of *observability* and *controllability*; however the use of observability is more pro-

nounced due to its importance in identifying the stochastic models. The observability determines the ability to estimate the initial state $x(0)$ from the measurements and inputs over the finite time interval. Similar to observability, the controllability is a desirable property of the system to be identified, which states that a state-controllable system is the one for which there is an input sequence available to drive the states to the desired final values in a finite time interval. The dynamical system of ODU-ASV is modeled as discrete-time state-space system; therefore, these concepts are explained easily and elaborately in the discrete-time domain below. Though, the derivation of controllability and observability theories both in continuous-times as well as discrete-time domains are equivalent in the sense that continuous-time system can be converted to discrete-time after sampling process. Assuming the linear discrete-time state-space representation of a dynamical system as described in the Equations (3.1) and (3.2).

3.5.1 Observability

Observability means that a state $x(p)$ at time step $k = p$ can be determined from the measurement of input $u(k)$ & output $y(k)$ over the finite time interval $[0 \leq k \leq p-1]$. If all the system states can be determined in this way, then it is called completely observable or simply observable system [38], [45], [47]. Consider the zero input system where $u(k) = 0$:

$$y(0) = Cx(0) \tag{3.5a}$$

$$y(1) = Cx(1) = CAx(0) \tag{3.5b}$$

$$\vdots \qquad\qquad \vdots$$

$$y(p-1) = Cx(p-1) = CA^{p-1}x(0) \tag{3.5c}$$

In compact matrix form, Eq. (3.5c) is as follows:

$$Y_p = \mathcal{O}_p\, x(0) \tag{3.6}$$

where

$$Y_p = \begin{bmatrix} y(0) \\ y(1) \\ \vdots \\ y(p-1) \end{bmatrix}, \qquad \mathcal{O}_p = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{p-1} \end{bmatrix} \tag{3.7}$$

The matrix $\mathcal{O}_p$ of $mp \times r_p$ is called the observability matrix, which plays a very important role in state estimation and identification. The rank of observability matrix $\mathcal{O}_p$ is the order of a minimal realization with state space dimension *n*. In other words, a LTI state-space system (*A*, *B*, & *C*) is observable if the associated observability matrix $\mathcal{O}_p$ is of full rank. The full rank of observability matrix $\mathcal{O}_p$ guarantees that $x(0)$ can be uniquely solved as:

$$x(0) \; = \; \mathcal{O}_p^{\dagger} \, Y_{\boldsymbol{p}} \tag{3.8}$$

where † means the pseudo-inverse. The term minimal basically relates to the minimum number of states required for input-output relationship after the pole-zero cancellations get done, because for a given state-space system, there may be infinitely different realizations; if the realization is not minimal, then the redundant states are inevitably included in the model, which leads to unobservable and/or uncontrollable modes [38].

To better understand the concept of observability, assume the discrete-time system of Eqs. (3.1) & (3.2) with the constant matrix *A* having a full set of distinct eigenvalues λ_i and only one output; let the system be transformed to modal coordinates. This system would be observable if and only if the constant vector $C_m = \; c \, \psi$ has no zero elements, where ψ is modal matrix having eigenvalues of *A* as its column vectors [38], [47]. This fact entails that in a single sensor system, the sensor is located at a node of a mode, then that particular mode will be unobservable. After modal transformation, the discrete-time single output system of Eq. (3.1) gets the form as:

$$\begin{aligned} x_m(k+1) &= \; \psi^{-1} A \psi x_m(k) + \psi^{-1} B u(k) \\ x_m(k+1) &= \; \Lambda x_m(k) + B_m u(k) \end{aligned} \tag{3.9}$$

$$y(k) = C_m x(k) + du(k) \tag{3.10}$$

Here the subscript m indicates that the matrices are resulted from the modal transformation, hence $x = \psi\, x_m$, $B_m = \psi^{-1} B$, and the output influence and eigenvalues matrices are given as below:

$$C_m = c\,\psi = [c_1\, c_2 \ldots\ldots c_n] \tag{3.11}$$

$$\Lambda = \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{bmatrix} \tag{3.12}$$

and the observability matrix in modal coordinates is as follows:

$$\bar{\mathcal{O}}_p = \begin{bmatrix} C_m \\ C_m\Lambda \\ \vdots \\ C_m\Lambda^{n-1} \end{bmatrix} = \begin{bmatrix} c_1 & c_2 & \ldots & c_n \\ \lambda_1 c_1 & \lambda_2 c_2 & \ldots & \lambda_n c_n \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_1^{n-1} c_1 & \lambda_2^{n-1} c_2 & \ldots & \lambda_n^{n-1} c_n \end{bmatrix} \tag{3.13}$$

If any element of the output influence vector $C_m$is zero, for instance c_1 = 0, then observability matrix $\bar{\mathcal{O}}_p$ will be rank deficient. This means the corresponding modal coordinate x_{m1} will be unobserved because it is not included in the output of the system. Both observability and controllability have dual relationship; hence the realization of a system is minimal when the system is both observable and controllable.

3.5.2 Controllability

Controllability is also a requirement for state-space model identification; once you identify the minimal realization of the state-space model then it is necessary to be able to excite all the observable states by some control action. A state $x(k)$ at time step k will be controllable if this state can be reached by some input sequence from the origin or some initial state $x(0)$ of the system in a finite time interval $[0 \leq k \leq p-1]$. Strictly speaking, the system is completely control-

lable or simply controllable only if a set of controllable states constitute the entire state-space. Consider a scalar control input $u(k)$ which means $r = 1$ and constant matrix A has distinct eigenvalues λ_i. Also, some initial state $x(0)$ be driven by the control input to a desired final state at $x(p)$ at time step $k = p$ (p being an arbitrary number) is as follows:

$$x(1) = Ax(0) + Bu(0) \tag{3.14a}$$

$$x(2) = Ax(1) + Bu(1) = A^2x(0) + ABu(0) + Bu(1) \tag{3.14b}$$

$$\vdots \qquad\qquad \vdots \qquad\qquad \vdots$$

$$x(p) = A^p x(0) + A^{p-1}Bu(0) + \cdots + ABu(p-2) + Bu(p-1)$$

$$= A^p x(0) + [A^{p-1}B \ldots AB\ B] \begin{bmatrix} u(0) \\ \vdots \\ u(p-2) \\ u(p-1) \end{bmatrix} \tag{3.14c}$$

In compact matrix form, Eq. (3.14c) is written as follows:

$$x(p) - A^p x(0) = \mathcal{C}_p u_p \tag{3.15}$$

where, $\mathcal{C}_p$ is the controllability matrix of $n \times rp$ as:

$$\mathcal{C}_p = [A^{p-1}B \ldots AB\ \ B] \tag{3.16}$$

In order to find the control input time history for driving the initial state $x(0)$ to a desired final state $x(p)$, Eq. (3.15) is solved. The solution depends on the rank of the controllability matrix $\mathcal{C}_p$. Therefore, it turns out that a linear, finite-dimensional, discrete-time dynamical system is controllable if and only if the controllability matrix is of full row rank *n*, which is the order of the system and rp is larger than or equal to *n*. The addition of terms beyond $A^{p-1}B$ in the controllability matrix will not improve its rank.

The concept of controllability is easily understand by transforming the original state-space equation into modal coordinates that diagonalizes *A* containing the distinct eigenvalues as

shown in Eq. (3.12). The state is written as $x = \psi x_m$, where ψ is a nonsingular modal matrix with full rank *n* containing all the independent eigenvectors of *A* as its column vectors. After modal coordinate transformation, the elements of B_m and controllability matrix $\bar{\mathcal{C}}_n$ is denoted by:

$$B_m = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \tag{3.17}$$

$$\bar{\mathcal{C}}_n = \psi \left[\Lambda^{n-1} B_m \ldots \Lambda B_m \;\; B_m \right] = \begin{bmatrix} \lambda_1^{n-1} b_1 & \ldots & \lambda_1 b_1 & b_1 \\ \lambda_2^{n-1} b_2 & \ldots & \lambda_2 b_2 & b_2 \\ \vdots & \ddots & \vdots & \vdots \\ \lambda_n^{n-1} b_n & \ldots & \lambda_n b_n & b_n \end{bmatrix} \tag{3.18}$$

If any of the elements of B_m vector is zero, then the rank of controllability matrix $\bar{\mathcal{C}}_n$ will become $n - 1$. Hence, from Eq. (3.15) in modal coordinates, it is concluded that element of the state vector can't be controlled by any input and the system becomes uncontrollable.

3.6 System Realization & Role of Markov Parameters

The term system realization will be used in this research throughout for MIMO state-space system such as ODU-ASV, which is a technique of computing the system matrices *A, B, C* from the *Markov parameters* as described below. For a given input-output system, there are infinite state-space realizations possible that depict the same input-output behavior. Consequently, there is a chance for the state-space description to include more states than what is sufficiently required which may leads to the loss of observability and controllability. Therefore, the term *minimal realization* descriptions will be introduced without involving in the cancellation of unstable modes. Such type of minimal realization has the minimum number of state variables essential for the description of specific input-output behavior of the dynamic system.

3.6.1 Markov Parameters

The sequence of system Markov parameters or Markov parameters are actually the pulse response of a dynamical system and can be obtained from the experimental data in the time-domain or using frequency response function (FRF) in the frequency domain. These parameters are unique for a given linear dynamical system and hence are commonly utilized as the foundation for the identification of discrete-time model for the said linear system.

For simplicity, consider the discrete LTI stochastic dynamical system in state-space form having single input i.e. $r = 1$ as shown in Equations (3.1) and (3.2). Now, let the unit pulse $u(0) = 1$ and $u(k) = 0$ for time index $k = 1, 2, 3, \ldots$ be applied to this system and solve for the output response $y(k)$ with zero initial condition i.e. $x(k) = 0$ as follows:

$$
\begin{aligned}
x(0) &= 0 && \xrightarrow{yields} y(0) = Du(0), \\
x(1) &= Bu(0) && \xrightarrow{yields} y(1) = CBu(0) + Du(1), \\
x(2) &= ABu(0) + Bu(1) && \xrightarrow{yields} y(2) = CABu(0) + CBu(1) + Du(2), \\
&\vdots && \vdots \qquad\qquad \vdots \\
x(k) &= \sum_{l=1}^{k} A^{i-1}Bu(k-i) && \xrightarrow{yields} y(k) = \sum_{l=1}^{k} CA^{i-1}Bu(k-i) + Du(k)
\end{aligned}
\tag{3.19}
$$

The constant matrices in the sequence of Eq. (3.19) are called as Markov parameters Y_k having dimension $m \times r$ which are unique and coordinate-independent. They are given as:

$$
Y_0 = D, \quad Y_1 = CB, \quad Y_2 = CAB, \quad \ldots., \quad Y_k = CA^{k-1}B \tag{3.20}
$$

As it is apparent from the Eq. (3.20), the constant system matrices *A*, *B*, *C*, and *D* are embedded in the Markov parameter sequence. As $Y_0 = D$; therefore, only the matrices *A*, *B*, and *C* are required to be determined for the system identification.

3.6.2 The Eigen-system Realization Algorithm (ERA)

Although there are various methods to estimate the state-space realization $(A, B, \& C)$ from the system Markov parameters, but the Eigen-system realization algorithm (ERA) is commonly used in identification and control of linear dynamical system. This method is first developed by Ho and Kalman (1966) for the deterministic case and followed by Kung (1978) for the noisy case. Once the open loop Markov parameters as shown in Eq. (3.20) are obtained, then they are incorporated to construct the generalized block data matrix known as *Hankel matrix* of the order $pm \times \gamma r$ as shown:

$$H(0) = \begin{bmatrix} Y_1 & Y_2 & \cdots & Y_\gamma \\ Y_2 & Y_3 & \cdots & Y_{\gamma+1} \\ \vdots & \vdots & \ddots & \vdots \\ Y_p & Y_{p+1} & \ldots & Y_{p+\gamma-1} \end{bmatrix} \tag{3.21}$$

where m and r are the number of outputs and inputs respectively, and p and γ are arbitrary integers chosen such that $\gamma r \geq pm$. In order the system to be both observable and controllable, Hankel matrix $H(0)$ will be of full rank n which indicates the order of the system. For this purpose, the dimension $pm \geq n$ trivially. In order to prove this claim, let construct $H(0)$ from Eq. (3.20) and Eq. (3.21) and decompose subsequently into the following two matrices:

$$H(0) = \begin{bmatrix} CB & CAB & \cdots & CA^{\gamma-1}B \\ CAB & CA^2B & \cdots & CA^{\gamma}B \\ \vdots & \vdots & \ddots & \vdots \\ CA^{p-1}B & CA^pB & \ldots & CA^{p+\gamma-2}B \end{bmatrix} = \mathcal{O}_p A^0 \mathcal{C}_\gamma = \mathcal{O}_p \mathcal{C}_\gamma \tag{3.22}$$

where $\mathcal{O}_p$ and $\mathcal{C}_\gamma$ are block observability and controllability matrices as:

$$\mathcal{O}_p = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{p-1} \end{bmatrix}, \tag{3.23a}$$

$$\mathcal{C}_p = [B \;\; AB \;\; A^2B \;\; \ldots \; A^{\gamma-1}B] \tag{3.23b}$$

As discussed previously, the system will be controllable and observable for minimum realization of order n, which leads to the fact that the block observability matrix $\mathcal{O}_p$ and block controllability matrix $\mathcal{C}_p$ will be of rank n. Hence, it is inferred from Eq. (3.22) that rank of the Hankel matrix $H(0)$ is n. The order of Hankel matrix can be same as that of the constant trix A of the true system in that case when the data has sufficiently low noise. The ERA begins with the factorization of the Hankel matrix as given in Eq. (3.22) utilizing singular value decomposition (SVD) as follows:

$$H(0) = R \sum S^T \tag{3.24}$$

where both matrices R and S have orthonormal columns so that $R^T R = I$ and $S^T S = I$; and $\sum$ is a rectangular matrix of monotonically non-increasing singular values arranged along the diagonal as given below:

$$\sum = \begin{bmatrix} \sum_n & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}, \quad \sum_n = diag[\sigma_1,\ \sigma_2, \dots ,\ \sigma_i,\ \ \sigma_{i+1}, \dots ,\ \sigma_n] \tag{3.25}$$

where $\mathbf{0}$ are zero matrices with suitable dimensions and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_i \geq \sigma_{i+1} \geq \cdots \geq \sigma_n \geq 0$. In reality, the Hankel matrix $H(0)$ is usually of full rank due to a noisy set of measurements, process noise, and computer round-off; therefore, in this case the rank does not represent the true order of the system. Now to dig out the true inherent linear system which produces the smooth version of dynamical response, it is the purpose of realization to select the first n non-zero singular values in the diagonal matrix $\sum$ and accordingly to choose the first n columns of R and S as well. Hence the Hankel matrix $H(0)$ of rank n as well as its pseudo-inverse becomes:

$$H(0) = R_n \sum_n S_n^T \tag{3.26}$$

$$H^{\dagger} = S_n \sum_n^{-1} R_n^T \tag{3.27}$$

where $R_n^T R_n = I_n = S_n^T S_n$; From Eqs. (3.22) and (3.26) we have:

$$H(0) = [R_n \sum_n^{\frac{1}{2}}] [\sum_n^{\frac{1}{2}} S_n^T] \cong \mathcal{O}_p \mathcal{C}_\gamma$$

which implies that $\mathcal{O}_p = [R_n \sum_n^{\frac{1}{2}}]$ and $\mathcal{C}_\gamma = [\sum_n^{\frac{1}{2}} S_n^T]$ making both $\mathcal{O}_p$ and $\mathcal{C}_\gamma$ balanced as:

$$\mathcal{O}_p^T \mathcal{O}_p = \sum_n^{\frac{1}{2}} R_n^T R_n \sum_n^{\frac{1}{2}} = \sum_n \tag{3.28a}$$

$$\mathcal{C}_\gamma \mathcal{C}_\gamma^T = \sum_n^{\frac{1}{2}} S_n^T S_n \sum_n^{\frac{1}{2}} = \sum_n \tag{3.28b}$$

Both the controllability and observability grammians are equal and diagonal as given in Eqs. (3.28a) and (3.28b) which implies that the estimated realization is as controllable as it is observable. From Eqs. (3.23a) and (3.23b), the estimated values of matrices B and C are formed as:

$\hat{B}$ = the first r columns of $\mathcal{C}_\gamma$,

$\hat{C}$ = the first m rows of $\mathcal{O}_p$,

In order to estimate the $\hat{A}$, generalized shifted block Hankel matrix of dimension $pm \times \gamma r$ is used as follows:

$$H(1) = \begin{bmatrix} Y_2 & Y_3 & \cdots & Y_{\gamma+1} \\ Y_3 & Y_4 & \cdots & Y_{\gamma+2} \\ \vdots & \vdots & \ddots & \vdots \\ Y_{p+1} & Y_{p+2} & \ldots & Y_{p+\gamma} \end{bmatrix} \tag{3.29}$$

Similar to the block Hankel matrix $H(0)$, the shifted block Hankel matrix $H(1)$ can be written with the help of Markov parameters and split subsequently into two matrices as given:

$$H(1) = \begin{bmatrix} CAB & CA^2B & \cdots & CA^{\gamma}B \\ CA^2B & CA^3B & \cdots & CA^{\gamma+1}B \\ \vdots & \vdots & \ddots & \vdots \\ CA^pB & CA^{p+1}B & \ldots & CA^{p+\gamma-1}B \end{bmatrix} = \mathcal{O}_p A \mathcal{C}_\gamma \tag{3.30}$$

$$= R_n \sum_n^{\frac{1}{2}} A \sum_n^{\frac{1}{2}} S_n^T \tag{3.31}$$

The state matrix A is obtained from Eq. (3.31) as:

$$\hat{A} = \sum_n^{-\frac{1}{2}} R_n^T H(1) S_n \sum_n^{-\frac{1}{2}} \tag{3.32}$$

The detailed mathematical proof of above Eq. (3.32) is given in [38], [47]. The identified triplet $(\hat{A}, \hat{B}, \& \hat{C})$ is a minimum realization having order n of the discrete-time state-space system under consideration. The sign ^ over the system matrices represent their estimated quantities identified from the data having sufficiently low noise, and hence, they are differentiated from the true system matrices.

The computation method of ERA with data correlation (ERA/DC) algorithm is all similar to the ERA except that in ERA/DC a block correlation matrix $'\mathrm{H}(0)$ with the elements of a square matrix called data correlation matrix $R_{hh}(k+\tau)$ is built; where the $R_{hh}(k+\tau)$ is the product of a Hankel matrix $H(0)$ and a shifted Hankel matrix $H(k+\tau)$ and τ being integer selected to avoid substantial overlap of adjacent R blocks. The square data correlation matrix $R_{hh}(k+\tau)$ is smaller than Hankel matrix $H(0)$ used in ERA. Also, an additional step required for the ERA/DC algorithm is the computation of controllability matrix Q_β, from which the first r columns determine the estimate of input matrix i.e. $\hat{B}$.

3.6.3 Additional Accuracy Indicators for Discrimination of True Modes

As described in the previous section that the singular values are used to distinguish true modes from the noise modes, in this section two other methods including the Modal Amplitude Coherence (MAC) and the Mode Singular Value (MSV) are described to be used for quantifying the system & noise modes. For this purpose, the identified discrete-time model with r inputs and m outputs is described in modal coordinates as given below:

$$x_m(k+1) = \hat{\Lambda} x_m(k) + \hat{B}_m u(k) \tag{3.33a}$$

$$y(k) = \hat{C}_m x(k) + Du(k) \tag{3.33b}$$

where the diagonal matrices $\hat{\Lambda}$ are the identified eigenvalues; as well as $\hat{B}_m$ and $\hat{C}_m$ are the input

and output matrices in modal coordinates respectively, which are given as:

$$\hat{\Lambda} = \begin{bmatrix} \hat{\lambda}_1 & & & \\ & \hat{\lambda}_2 & & \\ & & \ddots & \\ & & & \hat{\lambda}_n \end{bmatrix}, \tag{3.34a}$$

$$\hat{B}_m = \begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \\ \vdots \\ \hat{b}_n \end{bmatrix}, \qquad \& \quad \hat{C}_m = [\hat{c}_1 \; \hat{c}_2 \; \dots \dots \hat{c}_n \;] \tag{3.34b \& 3.34c}$$

The measurement vector $y(k)$ is real, so the identified eigenvalues will be complex conjugate pairs. Also, each $\hat{b}_i$ is an input row-vector of length r and $\hat{c}_i$ is an output column-vector of length m for $i = 1,2, \dots n$; where n is the number of modal coordinates. The sequence of identified Markov parameters for a linear system is:

$$\hat{Y} = \left[\hat{D} \;\; \hat{C}_m\hat{B}_m \;\; \hat{C}_m\hat{\Lambda}\,\hat{B}_m \;\; \dots \;\; \hat{C}_m\hat{\Lambda}^{l-2}\,\hat{B}_m\right] \stackrel{\text{def}}{=} [\hat{Y}_0 \;\; \hat{Y}_1 \;\; \hat{Y}_2 \dots \; \hat{Y}_{l-1}] \tag{3.35}$$

where l is the number of Markov parameters. Each modal coordinate constitutes a sequence of Markov parameters given as below:

$$\left[\hat{c}_i\hat{b}_i \;\; \hat{c}_i\hat{\lambda}_i\,\hat{b}_i \;\; \dots \;\; \hat{c}_i\hat{\lambda}_i^{l-2}\hat{b}_i\right] \;\; \text{for} \;\; i = 1,2, \dots, n \tag{3.36}$$

Here, l represents the length of data. From the row vector $\hat{b}_i$ and identified eigenvalue $\hat{\lambda}_i$, the sequence of identified modal amplitude time-history $\hat{q}_i$ is reconstructed:

$$\hat{q}_i = \left[\hat{b}_i \;\; \hat{\lambda}_i\hat{b}_i \;\; \hat{\lambda}_i^2\hat{b}_i \;\; \dots \;\; \hat{\lambda}_i^{l-2}\hat{b}_i\right] \;\; \text{for} \;\; i = 1,2, \dots, n \tag{3.37}$$

The sequence of identified modal amplitude time-history $\hat{q}_i$ can also be computed by decomposing the Hankel matrix, as given in Eq. (3.21), through SVD [38], [47].

$$H(0) = \begin{bmatrix} Y_1 & Y_2 & \cdots & Y_{l-\alpha} \\ Y_2 & Y_3 & \cdots & Y_{l-\alpha+1} \\ \vdots & \vdots & \ddots & \vdots \\ Y_\alpha & Y_{\alpha+1} & \dots & Y_{l-1} \end{bmatrix} \tag{3.38}$$

$$= [R_n \sum_n^{\frac{1}{2}}] \left[\sum_n^{\frac{1}{2}} S_n^T\right] \tag{3.39}$$

$$= \; [R_n \sum_n^{\frac{1}{2}}] \psi \, \psi^{-1} \left[\sum_n^{\frac{1}{2}} S_n^T\right]$$

$$= [R_n \sum_n^{\frac{1}{2}} \psi] \, [\psi^{-1} \sum_n^{\frac{1}{2}} S_n^T] \; \triangleq \; \bar{\mathcal{O}} \bar{\mathcal{C}} \tag{3.40}$$

The arbitrary number α is so selected that $m\alpha \geq n$, n being the order of the system. If the modal coordinates are used, then ψ is the eigen-vector matrix composed from the estimated state trix $\hat{A}$ rather than the real system matrix A which is usually unknown; therefore, from Eqs. (3.35) and (3.40), we have:

$$H(0) \; \cong \begin{bmatrix} C_m B_m & C_m \Lambda B_m & \cdots & C_m \Lambda^{l-\alpha-1} B_m \\ C_m \Lambda B_m & C_m \Lambda^2 B_m & \cdots & C_m \Lambda^{l-\alpha} B_m \\ \vdots & \vdots & \ddots & \vdots \\ C_m \Lambda^{\alpha-1} B_m & C_m \Lambda^{\alpha} B_m & \cdots & C_m \Lambda^{l-2} B_m \end{bmatrix} \triangleq \; \bar{\mathcal{O}} \; \bar{\mathcal{C}} \tag{3.41}$$

$$= \begin{bmatrix} C_m \\ C_m \Lambda \\ C_m \Lambda^2 \\ \vdots \\ C_m \Lambda^{\alpha-1} \end{bmatrix} [B_m \;\; \Lambda B_m \;\; \Lambda^2 B_m \;\; \ldots \;\; \Lambda^{l-\alpha-1} B_m] \tag{3.42}$$

Here, the real system matrices i.e. Λ, B_m, and C_m are described in modal coordinates. Due to noise, some very small nonzero singular values are truncated after the singular value decomposition; hence from Eq. (3.42)

$$\bar{\mathcal{C}} = [B_m \;\; \Lambda B_m \;\; \Lambda^2 B_m \;\; \ldots \;\; \Lambda^{l-\alpha-1} B_m] \tag{3.43}$$

or $$\bar{\mathcal{C}} \; = \begin{bmatrix} \bar{q}_1 \\ \bar{q}_2 \\ \vdots \\ \bar{q}_n \end{bmatrix} = \begin{bmatrix} [b_1 \;\; \lambda_1 b_1 \;\; \lambda_1^2 b_1 \;\; \ldots \;\; \lambda_1^{l-\alpha-1} b_1] \\ [b_2 \;\; \lambda_2 b_2 \;\; \lambda_2^2 b_2 \;\; \ldots \;\; \lambda_2^{l-\alpha-1} b_2] \\ \vdots \\ [b_n \;\; \lambda_n b_n \;\; \lambda_n^2 b_n \;\; \ldots \;\; \lambda_n^{l-\alpha-1} b_n] \end{bmatrix} \tag{3.44}$$

In real practice, some nonzero singular values are discarded owing to the presence of noise, therefore identified modal amplitude time-history $\hat{q}_i$ is an approximation of $\bar{q}_i$; otherwise both

would be identical.

3.6.3.1 MAC Accuracy Indicator

The MAC is defined to be the dot product of the vector formed by the unit pulse response history up to the selected timestamps that is related to the mode of the identified model and the corresponding vector made by the pulse response data used in the identification process. It is generally represents the cosine of the angle between the vectors of the measured response time-history and the identified model's response history as denoted by the following relation:

$$MAC_i = \frac{|\bar{q}_i \hat{q}_i^*|}{(|\bar{q}_i \bar{q}_i^*| \, |\hat{q}_i \hat{q}_i^*|)^{1/2}} \tag{3.45}$$

where $i = 1,2, \dots, n$ and " $*$ " denotes the transpose and complex-conjugate of the term.

3.6.3.2 MSV Accuracy Indicator

This is a method of quantifying as well as specifying the contribution of each mode to the pulse response history of the identified model obtained through ERA. Mathematically, it is described by taking the maximum singular value of the sequence of Markov parameters of each identified modal coordinate as:

$$MSV_i = \sqrt{|\hat{c}_i| \left(1 + |\hat{\lambda}_i| + |\hat{\lambda}_i^2| + \cdots + |\hat{\lambda}_i^{l-2}|\right)|\hat{b}_i|}$$

$$\approx \sqrt{\frac{|\hat{c}_i| \, |\hat{b}_i|}{(1-|\hat{\lambda}_i|)}} \quad \text{(If } |\hat{\lambda}_i| < 1 \text{ for sufficiently large } l\text{)} \tag{3.46}$$

CHAPTER 4

OBSERVER KALMAN FILTER IDENTIFICATION

4.1 Introduction

In chapter 3, the state-space realization performed by some time-domain method namely ERA or ERA/DC is discussed; in which the Hankel matrix composed of the system Markov parameters are used to identify the state-space system matrices, i.e., $(A, B, C, \& D)$. The system Markov parameters for the lightly damped systems obtained through traditional identification methods may present very slow decay of the response to initial conditions, which may contribute to high computational cost in the identification algorithm owing to the larger Hankel matrix. Therefore, in order to identify the lightly damped systems in discrete time domain with multiple inputs and multiple outputs, an asymptotically stable observer model is introduced and is the main topic of this chapter. The purpose of using an *observer* allows to change the decay rate of the original system response as desired by compressing the number of identified Markov parameters leading to smaller Hankel matrix. Consequently, the data is compressed to achieve the computational efficiency. Section 4.1 is introduction to this chapter; while in section 4.2, some prominent types of discrete-time model structures are discussed. The concept of state-space linear observer is introduced in section 4.3. The backbone of system identification work done in this research is the core-equation of the observer Kalman filter identification (OKID) algorithm, i.e., Eq. (4.32), which is detailed in sections 4.4 and 4.5. Afterwards in section 4.6, the relationship of discrete-time state space observer and steady-state Kalman filter is discussed. Finally, observable canonical form realization, which is also used in this research, is elaborated.

4.2 Prominent Types of Model Structures

As discussed in the previous chapter, system identification pertained to the techniques of developing the mathematical models of dynamical systems from the set of input-output sampled-data. A quality model obtained through system identification is normally validated by its goodness of fit, i.e. how well the model attempt to fit the measured data. Therefore, a prior knowledge regarding the system at hand is pretty much supportive for the selection of model structure. Generally, the overall model represents the deterministic-plus-stochastic description of the input-output pair $\{u(k), y(k)\}$ for the combined deterministic as well as stochastic effects. Hence the output measurement $y(k)$ of the model is the superposition of true response $y_{true}(k)$ generated by the deterministic system and the disturbances/noise $v(k)$ in discrete domain as

$$y(k) = y_{true}(k) + v(k) \tag{4.1}$$

Let the deterministic and stochastic systems in Eq. (4.1) be discrete LTI systems denoted by the transfer functions G and H representing the plant and noise models, respectively; also, let the discrete input be $u(k)$ and the error to be Gaussian white noise (GWN); then, we have

$$y(k) = G\,u(k) + H\,e(k) \qquad G, H\text{: LTI} \;\;\&\;\; e(k)\text{: GWN} \tag{4.2}$$

As discussed in the previous chapter, there are two main categories of the model structures of discrete-time deterministic and stochastic systems, i.e. non-parametric and parametric, which may be cast either in time-domain or in frequency-domain. A brief summary of these model structures in conjunction with structured and unstructured state-space models are given below.

4.2.1 Non-parametric Model structure

In non-parametric models, both the system/process and noise models' transfer tions G and H, respectively, are represented in non-parametric form, such as convolution form or using spectral representations. Some of the important descriptions of the non-parametric model

structure are impulse, step, and frequency response models. The relevant features of the non-parametric models are that they require minimal assumptions about the process and that they don't need any substantial user intervention. The example of non-parametric time-domain model corresponding to convolution forms for G and H is

$$y(k) = \sum_{n=-\infty}^{\infty} g(n)u(k-n) + \sum_{m=-\infty}^{\infty} h(m)\, e(k-m) \tag{4.3}$$

Let's consider the usual conditions of causality, i.e. $g(n) = 0$, $h(m) = 0$, for $n < 0$ & $m < 0$ and uniqueness of noise model, i.e. $h(0) = 1$. Then, Eq. (4.3) can be expressed as

$$y(k) = \underbrace{\sum_{n=0}^{\infty} g(n)u(k-n) + \sum_{m=0}^{\infty} h(m)\, e(k-m)}_{\boldsymbol{predictable\ part}} + e(k) \tag{4.4}$$

4.2.2 Parametric Model structure

Parametric structures of the input-output system given in Eq. (4.2) result from the parameterizing the system and noise models $G(q^{-1})$ and $H(q^{-1})$ respectively. Parameterization of G and H is a very helpful representation of the system regarding compactness; but, using this approach, one has to confront the challenge of selecting the suitable model structure for a given application. There are many methods of parameterizing the transfer functions G and H, but the rational polynomial transfer function representations are mostly incorporated. Basically three main categories of parametric model structures exist, namely the equation-error (e.g. AR, ARX, MA, and ARMAX), output-error, and the Box-Jenkins; the equation-error and output-error models are briefly discussed below.

4.2.2.1 AR Model

The output of Auto-Regressive (AR) model is the autoregressive of itself. The structure of AR model is

$$y(k) = \sum_{i=1}^{q} \alpha_i y(k-i) + e(k) \tag{4.5}$$

where α_i are the parameters of AR model and $e(k)$ is random Gaussian white noise.

4.2.2.2 ARX Model

The parametric description knows as the Auto-Regressive eXogenous (ARX) model is the extension of the AR model which is commonly incorporated in the recursive technique of system identification. The structure of ARX model in the difference equation form is given as:

$$y(k) = \sum_{i=1}^{q} \alpha_i y(k-i) + \sum_{i=1}^{q} \beta_i u(k-i) + e(k) \tag{4.6}$$

where $y(k-i)$ represents the auto-regressive (AR) part, $u(k-i)$ represents the exogenous (X) part, α_i and β_i are the ARX model parameter coefficients. As the random Gaussian white noise gets into the difference equation, it is also called the *equation-error* model. A merit of the ARX model structure is that different orders of the system can be screen out in a computationally efficient way for their suitability.

4.2.2.3 MA Model

The moving average (MA) model structure represents the moving average term combined with the Gaussian white noise as follows:

$$y(k) = \sum_{i=1}^{q} \gamma_i e(k-i) + e(k) \tag{4.7}$$

where γ_i are the moving average parameters.

4.2.2.4 ARMAX Model

The model structure named Auto-Regressive Moving Average with eXogenous input (ARMAX) is an extension of both the ARX model because of the incorporated moving average

term in the noise model as well as the ARMA model due to the effects of exogenous inputs, which is given:

$$y(k) = \sum_{i=1}^{q} \alpha_i y(k-i) + \sum_{i=1}^{q} \beta_i u(k-i) + \sum_{i=1}^{q} \gamma_i e(k-i) + e(k) \tag{4.8}$$

where α_i, β_i, and γ_i are the ARMAX parameters.

4.2.2.5 Output-error Model

It is intuitive that for one-step ahead prediction of the measurement, prediction of both discrete-time deterministic system/process and stochastic disturbances/noise are required

$$\hat{y}(k) = \hat{x}(k) + \hat{e}(k) \tag{4.9}$$

Assuming the error in the measurement is absolutely unpredictable; it means that zero-mean Gaussian white noise error signal becomes $\hat{e}(k) = 0$. So from Eq. (4.9), we have

$$\hat{y}(k) = \hat{x}(k) = -\alpha_1 x(k-1) + \beta_1 u(k-1) \tag{4.10}$$

where α_1, β_1, and $x(k-1)$ are unknowns. Since GWN error signal directly enters the output, so the model constructing the output predictor of Eq. (4.10) is known as the output-error model.

4.2.3 Un-structured and Structured State-space Model

The third alternative of model structure is either the unstructured or structured state-space model. The model description which has no preference for specific structure for any or all of the system matrices or specific basis for the states is known as freely parameterized or unstructured state-space model. Therefore, the unstructured state-space model usually has a much larger number of unknowns to be estimated than those of a corresponding input-output model characterized by a transfer function. The OKID is not equipped with the ability of imposition of any constraints on the structure of sate-space matrices; the condition of constraints' imposition may generally arise whenever a state-space form corresponds to a transfer function form or a physical

representation is desired. On the other hand, the structured state-space model is the one in which the structural constraints are required to be imposed on system matrices corresponding to a particular input-output model structure, e.g. ARX, ARMAX that results in the matrices having non-zero entries only in specific locations. The benefit of the structured state-space model is parsimony, identifiability, and physically meaningful representation on the cost of excessive computation for its estimation. The structured state-space models are further sub-divided into parameterized black-box and grey-box state-space models due to the prior knowledge which causes to make a particular structure. The example of structured state-space model, in which non-zero entries of system matrices are basically dealt as parameters θ as

$$x(k+1) = A(\theta)x(k) + B(\theta)u(k) \tag{4.11a}$$

$$y(k) = C(\theta)x(k) + D(\theta)u(k) \tag{4.11b}$$

4.3 State-space Linear Observer

In most cases, the sate vector of the linear state-space model can't be accessed through direct real-time measurement but sometimes a subset or a linear combination of the state is available to be measured from the output y. Therefore, the state-space observer or *state estimator*, comprised of the linear state-space equations, is used to estimate the states and possibly the output of the original linear system by utilizing the measured input and output data so that those can adequately be used for the state-feedback or optimal trajectory tracking control of the dynamical system. Generally, for state estimation of the deterministic systems, the Luenberger observer is used, while for the stochastic systems, the Kalman filter formulated by Kalman (Kalman, 1960; Kalman & Bucy 1961) is a linear estimator used for estimation of the states optimally under the assumptions of Gaussian white noise both in the process and measurements [38]. The observer equation in discrete-time domain is derived from the generic discrete LTI multivariable state-

space model given as following [37], [45]:

$$x(k+1) = Ax(k) + Bu(k) \tag{4.12}$$

$$y(k) = Cx(k) + Du(k) \tag{4.13}$$

where $x(k) \in R^{n\times 1}$, $u(k) \in R^{r\times 1}$, and $y(k) \in R^{m\times 1}$ are the state, input, and the output vectors. Let the state-space matrices, i.e. $A, B, C,$ & D having suitable dimensions as well as the put $u(k)$ and the output $y(k)$ be known. Also, the zero initial conditions are to be assumed, i.e. $x(0) = 0$. The state and output response of the discrete LTI system as given in Eqn. (4.12) and (4.13) for the time instants $k = 0, 1, \ldots, l-1$ becomes

$$\begin{aligned} x(0) &= 0, & y(0) &= Du(0) \\ x(1) &= Bu(0), & y(1) &= CBu(0) + Du(1) \\ x(2) &= ABu(0) + Bu(1), & y(2) &= CABu(0) + CBu(1) + Du(2) \\ &\vdots & &\vdots \end{aligned}$$

$$x(l-1) = \sum_{i=1}^{l-1} A^{i-1} Bu(l-1-i) \tag{4.14}$$

$$y(l-1) = \sum_{i=1}^{l-1} CA^{i-1} Bu(l-1-i) + Du(l-1) \tag{4.15}$$

Eq. (4.15) can be written in matrix form

$$y_{m\times l} = Y_{m\times rl}\, U_{rl\times l} \tag{4.16}$$

where dimensions $m, r,$ and l are the number of outputs, inputs, and data length respectively. The terms $y, Y,$ and U represent the block matrices of the output data, unknown Markov parameters, and input data as given

$$y = [y(0) \;\; y(1) \;\; y(2) \;\; \ldots \;\; y(l-1)] \tag{4.17}$$

$$Y = [D \;\; CB \;\; CAB \;\; \ldots \;\; CA^{l-2}B] \tag{4.18}$$

$$U = \begin{bmatrix} u(0) & u(1) & u(2) & \dots & u(l-1) \\ & u(0) & u(1) & \dots & u(l-2) \\ & & u(0) & \dots & u(l-3) \\ & & & \ddots & \vdots \\ & & & & u(0) \end{bmatrix} \tag{4.19}$$

From Eq. (4.16), it is evident that the solution for matrix Y is not unique if $r > 1$. Additionally, when the input signals has zero initial value or not rich in frequency content adequately, or the data length l is too large; then the block upper triangular input matrix U turns out to be ill-conditioned and hence the solution of the matrix $Y = yU^{-1}$ can't be accurately found. In order for the system matrix A to be asymptotically stable, a time index p is arbitrarily selected to be sufficiently large to make $CA^kB \approx 0$ for $k \geq p$; so Eq. (4.16) can be expressed as

$$y_{m\times l} \approx \text{¥}_{m\times r(p+1)}\, \text{Ʉ}_{r(p+1)\times l} \tag{4.20}$$

where data samples l should be taken larger than $r(p+1)$. The block matrices $\text{¥}_{m\times r(p+1)}$ and $\text{Ʉ}_{r(p+1)\times l}$ are the truncated variants of the $Y_{m\times rl}$ and $U_{rl\times l}$ as given in Eq. (4.16). For state-space system of Eqs. (4.12) as well as (4.13), the first p Markov parameters approximate the solution of $\text{¥} = y\text{Ʉ}^{\dagger}$, where $\text{Ʉ}^{\dagger}$ is the Moore-Penrose pseudo-inverse of the block input matrix Ʉ. For lightly damped system, the values of integer p and hence data length l are to be increased so that Eq. (4.20) is satisfied. Consequently, the size of the block input matrix U also becomes excessively large which makes the calculation of its Moore-Penrose pseudo-inverse inadequate. For this purpose, damping of the system is increased artificially by introducing a term $Gy(k)$ with the arbitrary matrix $G \in R^{n\times m}$ to the state equation i.e., Eq. (4.12). As a result, the most general form of linear discrete-time state-space observer equation is derived as:

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + Gy(k) - Gy(k) \\ &= (A + GC)x(k) + (B + GD)u(k) - Gy(k) \end{aligned} \tag{4.21}$$

or

$$x(k+1) = \bar{A}x(k) + \bar{B}v(k) \tag{4.22}$$

where

$$\bar{A} = A + GC \tag{4.23a}$$

$$\bar{B} = [B + GD \;\; -G] \tag{4.23b}$$

$$v(k) = \begin{bmatrix} u(k) \\ y(k) \end{bmatrix} \tag{4.23c}$$

Eq. (4.22) is called a discrete-time state-space observer equation of dynamic system provided that the state $x(k)$ is taken as an observer sate vector. The observer sate vector is driven by the input $v(k)$ comprised of $u(k)$ and $y(k)$, which are actually the input and the output of original state-space system of Eqs. (4.12) and (4.13).

4.4 OKID Core-equation and Computation of Observer Markov Parameters

The set of Markov parameters of the state-space observer equation, i.e. Eq. (4.22), are known collectively as the observer Markov parameters (OMPs) denoted by $\overline{Y}$. These parameters are the blend of system Markov parameters and observer gain Markov parameters. The system matrices $A, B, C,$ and D are computed from the former while the observer gain matrix G is determined by the latter parameters. Similar to the Eqs. (4.14) and (4.15), the input-output mapping for Eq. (4.22) is described in matrix form

$$y_{m\times l} = \bar{Y}_{m\times[(m+r)(l-1)+r]} \quad V_{[(m+r)(l-1)+r]\times l} \tag{4.24}$$

where

$$\begin{aligned} y &= [y(0) \;\; y(1) \;\; y(2) \;\; \ldots \;\; y(l-1)] \\ \bar{Y} &= [D \;\; C\bar{B} \;\; C\bar{A}\bar{B} \;\; \ldots C\bar{A}^{p-1}\bar{B} \ldots \; C\bar{A}^{l-2}\bar{B}] \end{aligned} \tag{4.25}$$

$$V = \begin{bmatrix} u(0) & u(1) & u(2) & \dots & u(p) & \dots & u(l-1) \\ & v(0) & v(1) & \dots & v(p-1) & \dots & v(l-2) \\ & & v(0) & \dots & v(p-2) & \dots & v(l-3) \\ & & & \ddots & \vdots & \dots & \vdots \\ & & & & v(0) & \dots & v(l-p-1) \\ & & & & & \ddots & \vdots \\ & & & & & & v(0) \end{bmatrix} \tag{4.26}$$

The matrix $G \in R^{n\times m}$ can arbitrarily be selected in such a way that the observable system of Eq. (4.22) becomes a deadbeat observer, which implies that all the eigenvalues of the observer state matrix $\bar{A}$ are attempted to be placed at the origin. This causes the observer Markov parameters $C\bar{A}^k\bar{B} = 0$ for $k \geq p$ in the scenario of noise-free data. In case of real-time data having some noise, the OMPs may not decay asymptotically to zero; therefore, the eigenvalues of the observer state matrix $\bar{A}$ are placed in a way to make $C\bar{A}^k\bar{B} \approx 0$ for $k \geq p$, provided that the integer p is chosen to be sufficiently large. With this approach, the observer Markov parameters are obtained by solving the linear input-output mapping Eq. (4.27) as given below that is formed by the real noisy data set.

$$y_{m\times l} = \overline{\Upsilon}_{m\times[(m+r)p+r]} \ \forall_{[(m+r)p+r]\times l} \tag{4.27}$$

where

$$y = [y(0) \ \ y(1) \ \ y(2) \ \dots \ y(l-1)]$$

$$\overline{\Upsilon} = [D \ \ C\bar{B} \ \ C\bar{A}\bar{B} \ \dots \ C\bar{A}^{p-1}\bar{B}] \tag{4.28}$$

$$\forall = \begin{bmatrix} u(0) & u(1) & u(2) & \dots & u(p) & \dots & u(l-1) \\ & v(0) & v(1) & \dots & v(p-1) & \dots & v(l-2) \\ & & v(0) & \dots & v(p-2) & \dots & v(l-3) \\ & & & \ddots & \vdots & \dots & \vdots \\ & & & & v(0) & \dots & v(l-p-1) \end{bmatrix} \tag{4.29}$$

Similar to Eq. (4.20), the block matrices $\overline{\Upsilon}$ and $\forall$ of the observer equation are the truncated variants of the $\bar{Y}$ and V as given in Eqs. (4.25) and (4.26), respectively. For state-space realization of Eq. (4.22) which is the linear observer equation, the non-recursive (batch) least square solution

to equation $\overline{¥} = y\ \forall^{\dagger}$ for $\overline{¥}$ is approximately satisfied by the first p Markov parameters; where $\forall^{\dagger}$ is the Moore-Penrose pseudo-inverse of the block input matrix $\forall$ related to the observer equation. The unique solution of $\overline{¥}$ demands that all the rows of $\forall$ must be linearly independent. Therefore the integer p is chosen to be sufficiently large enough in order to maximize the number of independent rows $(m + r)p + r \leq l$ of $\forall$. It means that p is the upper bound of the order of the deadbeat observer. The non-recursive (batch) least square solution to Eq. (4.27) is given as follows. Considering the state equation of LTI observer system as given in Eq. (4.22) and let's assume zero initial conditions, i.e. $x(0) = 0$, then the sequence is

$$\begin{aligned}
x(k+1) &= \bar{A}x(k) + \bar{B}v(k) \\
x(k+2) &= \bar{A}x(k+1) + \bar{B}v(k+1) \\
&= \bar{A}^2x(k) + \bar{A}\bar{B}v(k) + \bar{B}v(k+1) \\
&\vdots \qquad\qquad\qquad \vdots \\
x(k+p) &= \bar{A}x(k+p-1) + \bar{B}v(k+p-1) \\
&= \bar{A}^p x(k) + \bar{A}^{p-1}\bar{B}v(k) + \bar{A}^{p-2}\bar{B}v(k+1) + \cdots \\
&\quad + \bar{B}v(k+p-1)
\end{aligned} \tag{4.30}$$

Incorporating Eq. (4.30) into Eq. (4.13) gives

$$\begin{aligned}
y(k+p) &= Cx(k+p) + Du(k+p) \\
&= C\bar{A}^p x(k) + C\bar{A}^{p-1}\bar{B}v(k) + C\bar{A}^{p-2}\bar{B}v(k+1) + \cdots \\
&\quad + C\bar{B}v(k+p-1) + Du(k+p)
\end{aligned} \tag{4.31}$$

The set of equations in compact matrix form for $k = 0,1, \ldots, l-1$ is the core equation of OKID given as below:

$$\bar{y} = C\bar{A}^p x + \overline{¥}\,\overline{\forall} \tag{4.32}$$

where $\bar{y} = [y(p)\ \ y(p+1)\ \ y(p+2)\ \ \ldots\ \ y(l-1)]$

$$x = [x(0) \;\; x(1) \;\; x(2) \;\; \dots \;\; x(l-p-2)]$$

$$\overline{\Upsilon} = [D \;\; C\bar{B} \;\; C\bar{A}\bar{B} \;\; \dots \; C\bar{A}^{p-1}\bar{B}]$$

$$\overline{\forall} = \begin{bmatrix} u(p) & u(p+1) & \dots & u(l-1) \\ v(p-1) & v(p) & \dots & v(l-2) \\ v(p-2) & v(p-1) & \dots & v(l-3) \\ \vdots & \vdots & \vdots & \ddots \\ v(0) & v(1) & \dots & v(l-p-1) \end{bmatrix} \tag{4.33}$$

In fact, the terms $\bar{y}$ and $\overline{\forall}$ of Eq. (4.32) are known measurements, while the OMPs denoted by $\overline{\Upsilon}$ are not known. The term $C\bar{A}^{p}x$ given in Eq. (4.32) can be discarded, provided that the observer sate matrix $\bar{A}^{p}$ is very small and all the states are bounded then Eq. (4.32) is written as:

$$\bar{y} = \overline{\Upsilon}\,\overline{\forall} \tag{4.34}$$

Eq. (4.34) can be solved for observer Markov parameters $\overline{\Upsilon}$ by non-recursive (batch) least-square method subject to the condition that the inverse of matrix $[\overline{\forall}\,\overline{\forall}^{T}]$ exists as given below:

$$\overline{\Upsilon} = \bar{y}\;\overline{\forall}^{T}\;[\overline{\forall}\,\overline{\forall}^{T}]^{-1} \tag{4.35}$$

otherwise

$$\overline{\Upsilon} = \bar{y}\;\;\overline{\forall}^{\dagger} \tag{4.36}$$

where $\overline{\forall}^{\dagger}$ is the Moore-Penrose pseudo-inverse of the block input matrix $\overline{\forall}$; moreover, the subset matrices $\bar{y}$ & $\overline{\forall}$ are obtained by discarding the first p columns of the matrices y and $\forall$ related to the linear input-output mapping Eq. (4.27).

4.4.1 Extraction of System Markov Parameters from OMPs

As already discussed, the OMPs of the observer model are combination of the system Markov parameters and the observer gain Markov parameters. The system matrices $A, B, C,$ and D are computed from the system Markov parameters through ERA algorithm as discussed in chapter 3, while the observer gain matrix G is determined from the observer gain Markov parameters. For obtaining the system Markov parameters, the identified observer Markov parameters in

matrix $\overline{\text{Ɣ}}$ is partitioned

$$\overline{\text{Ɣ}} = [\overline{\text{Ɣ}}_0 \;\; \overline{\text{Ɣ}}_1 \;\; \overline{\text{Ɣ}}_2 \ldots \overline{\text{Ɣ}}_p] \tag{4.37}$$

where

$$\overline{\text{Ɣ}}_0 = \text{Ɣ}_0 = D$$

$$\overline{\text{Ɣ}}_k = C\bar{A}^{k-1}\bar{B} \quad \text{for } k = 1,2,3, \ldots \tag{4.38a}$$

$$= [C((A+GC)^{k-1})(B+GD) \; - C(A+GC)^{k-1}G] \tag{4.38b}$$

$$\triangleq [\overline{\text{Ɣ}}_k^{(1)} \quad - \overline{\text{Ɣ}}_k^{(2)}] \tag{4.38c}$$

The system Markov parameter CB is recovered from the product term $\overline{\text{Ɣ}}_1^{(1)}$ in Eq. (4.38b):

$$\overline{\text{Ɣ}}_1^{(1)} = C(B+GD)$$

$$\overline{\text{Ɣ}}_1^{(1)} = CB - (CG)D$$

As the system Markov parameter $\text{Ɣ}_1 = CB$,

$$\overline{\text{Ɣ}}_1^{(1)} = \text{Ɣ}_1 - \overline{\text{Ɣ}}_1^{(2)}D$$

$$\text{Ɣ}_1 = \overline{\text{Ɣ}}_1^{(1)} - \overline{\text{Ɣ}}_1^{(2)}D \tag{4.39}$$

Similarly, the Markov parameter CAB is recovered from the product term $\overline{\text{Ɣ}}_2^{(1)}$ given in Eq. (4.38b):

$$\overline{\text{Ɣ}}_2^{(1)} = C\bar{A}\bar{B}$$

$$= C(A+GC)\,(B+GD)$$

$$= CAB + CGCB + CAGD + CCGGD$$

$$= CAB + CGCB + C(A+CG)GD$$

As the system Markov parameter $\text{Ɣ}_2 = CAB$, so

$$\overline{\text{Ɣ}}_2^{(1)} = \text{Ɣ}_2 + \overline{\text{Ɣ}}_1^{(2)}\text{Ɣ}_1 + \overline{\text{Ɣ}}_2^{(2)}D$$

$$\text{Ɣ}_2 = \overline{\text{Ɣ}}_2^{(1)} - \overline{\text{Ɣ}}_1^{(2)}\text{Ɣ}_1 - \overline{\text{Ɣ}}_2^{(2)}D \tag{4.40}$$

Likewise, the Markov parameter CA^2B is recovered from the product term $\overline{¥}_3^{(1)}$ given in Eq. (4.38b):

$$\begin{aligned}\overline{¥}_3^{(1)} &= C\bar{A}^2\bar{B} \\ &= C(A+GC)^2\,(B+GD) \\ &= C(A^2+CGA+ACG+CGCG+(B+GD) \\ &= CA^2B+CGCAB+C(A+GC)GCB+C(A+GC)^2GD\end{aligned}$$

As the system Markov parameter $¥_3 = CA^2B$,

$$\overline{¥}_3^{(1)} = ¥_3 + \overline{¥}_1^{(2)}¥_2 + \overline{¥}_2^{(2)}¥_1 + \overline{¥}_3^{(2)}D$$

$$¥_3 = \overline{¥}_3^{(1)} - \overline{¥}_1^{(2)}¥_2 - \overline{¥}_2^{(2)}¥_1 - \overline{¥}_3^{(2)}D \tag{4.41}$$

Therefore by induction, the system Markov parameters are generally described in terms of the observer Markov parameters as

$$¥_k = \overline{¥}_k^{(1)} - \sum_{i=1}^{k} \overline{¥}_i^{(2)}\,¥_{k-i} \qquad \text{for } k = 1,2,\dots,p \tag{4.42a}$$

$$¥_k = -\sum_{i=1}^{p} \overline{¥}_i^{(2)}\,¥_{k-i} \qquad \text{for } k = p+1,\dots,\infty \tag{4.42b}$$

From both parts of Eq. (4.42) as above, it is evident that the system Markov parameter $¥_k$ is the linear combination of its past arbitrary p system Markov parameters; which indicates one can choose only p independent system Markov parameters for a system of order n. The integer p is chosen in such a way that the terms $\overline{¥}_k^{(1)}$ and $\overline{¥}_k^{(2)}$ are assumed to be zero for the index $k > p$. The system Markov parameters can also be related to the observer Markov parameters with the help of generalized Hankel matrix H as

$$\breve{¥}^{(2)}H = \breve{¥} \tag{4.43}$$

where the integer N is selected to be sufficiently large and the matrices $\breve{¥}^{(2)}$, H and $\breve{¥}$ are:

$$\breve{\Upsilon}^{(2)} = [-\overline{\Upsilon}_p^{(2)} \ -\overline{\Upsilon}_{p-1}^{(2)} \ -\overline{\Upsilon}_{p-2}^{(2)} \ \dots \ -\overline{\Upsilon}_1^{(2)}] \tag{4.44}$$

$$H = \begin{bmatrix} \Upsilon_2 & \Upsilon_3 & \Upsilon_4 & \cdots & \Upsilon_{N+1} \\ \Upsilon_3 & \Upsilon_4 & \Upsilon_5 & \cdots & \Upsilon_{N+2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \Upsilon_{p+1} & \Upsilon_{p+1} & \Upsilon_{p+2} & \dots & \Upsilon_{p+N} \end{bmatrix} \tag{4.45}$$

$$\breve{\Upsilon} = [\ \Upsilon_{p+2} \ \Upsilon_{p+3} \ \Upsilon_{p+4} \ \dots \ \Upsilon_{p+N+1}] \tag{4.46}$$

As from Eq. (3.27), we know the Hankel matrix can be written

$$H = \mathcal{O}A\mathcal{C} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{p-1} \end{bmatrix} A\,[B \ AB \ A^2B \ \dots \ A^{N-1}B] \tag{4.47}$$

Hence, from Eqs. (4.43) and (4.47)

$$\breve{\Upsilon} = \breve{\Upsilon}^{(2)} H = \breve{\Upsilon}^{(2)}\, \mathcal{O}A\mathcal{C} \tag{4.48}$$

The number of observer Markov parameters p calculated must be selected in such a way so that $mp \geq n$; where m, p, and n are the number of outputs, arbitrary number, and order of the identified system model. As from Eq. (4.42), the number p specifies the maximum number of identified system Markov parameters. It indicates mp is the upper bound on the order of the identified system model [38], [47]. The set of equations i.e. Eq. (4.42) can also be written in the compact matrix form:

$$\begin{bmatrix} I & & & & \\ \overline{\Upsilon}_1^{(2)} & I & & & \\ \overline{\Upsilon}_2^{(2)} & \overline{\Upsilon}_1^{(2)} & I & & \\ \vdots & \vdots & \vdots & \ddots & \\ \overline{\Upsilon}_k^{(2)} & \overline{\Upsilon}_{k-1}^{(2)} & \overline{\Upsilon}_{k-2}^{(2)} & \dots & I \end{bmatrix} \begin{bmatrix} \Upsilon_1 \\ \Upsilon_2 \\ \Upsilon_3 \\ \vdots \\ \Upsilon_{k+1} \end{bmatrix} = \begin{bmatrix} \overline{\Upsilon}_1^{(1)} - \overline{\Upsilon}_1^{(2)} D \\ \overline{\Upsilon}_2^{(1)} - \overline{\Upsilon}_2^{(2)} D \\ \overline{\Upsilon}_3^{(1)} - \overline{\Upsilon}_3^{(2)} D \\ \vdots \\ \overline{\Upsilon}_{k+1}^{(1)} - \overline{\Upsilon}_{k+1}^{(2)} D \end{bmatrix} \tag{4.49}$$

where I and $\overline{\Upsilon}_i^{(2)}$ (for $i = 1, 2, \dots, k+1$) are $m \times m$ square matrices. It is noted from Eq. (4.49) that recursive back substitution for $\Upsilon_1, \Upsilon_2, \dots., \Upsilon_{k+1}$ gives Eq. (4.42).

4.4.2 Extraction of Observer-gain Markov Parameters from OMPs

The observer gain G is the main quantity used for relating the linear observer equation to the finite-difference equation. The quantity G is identified by computing the following sequence in terms of the OMP.

$$Y_k^o = CA^{k-1}G \quad \text{for } k = 1,2,3,\ldots \tag{4.50}$$

The sequence of Eq. (4.50) is usually known as observer gain Markov parameters. The realization of C and A in the sequence can be achieved by the identification procedure described in Chapter 3. The first, second, and third parameters of the sequence in Eq. (4.50) are obtained by considering the term $\overline{Y}_k^{(2)} = C(A+GC)^{k-1}G$ in Eq. (4.38) as:

$$Y_1^o = CG = \overline{Y}_1^{(2)} \tag{4.51}$$

$$\overline{Y}_2^{(2)} = C\bar{A}G = C(A+GC)G$$

$$= (CAG + CGCG) = Y_2^o + \overline{Y}_1^{(2)} Y_1^o$$

$$\xrightarrow{yields} \quad Y_2^o = \overline{Y}_2^{(2)} - \overline{Y}_1^{(2)} Y_1^o \tag{4.52}$$

$$\overline{Y}_3^{(2)} = C\bar{A}^2G = C(A+GC)^2G$$

$$= (CA^2G + CGCAG + C\bar{A}GCG)$$

$$\xrightarrow{yields} \quad Y_3^o = \overline{Y}_3^{(2)} - \overline{Y}_1^{(2)} Y_2^o - \overline{Y}_2^{(2)} Y_1^o \tag{4.53}$$

Hence, the sequence of observer gain Markov parameters are generally described in terms of OMP as:

$$Y_1^o = CG = \overline{Y}_1^{(2)} \tag{4.54a}$$

$$Y_k^o = \overline{Y}_k^{(2)} - \sum_{i=1}^{k-1} \overline{Y}_i^{(2)} Y_{k-i}^o \qquad \text{for } k = 2,3,\ldots,p \tag{4.54b}$$

$$Y_k^o = -\sum_{i=1}^{p} \overline{Y}_i^{(2)} Y_{k-i}^o \qquad \text{for } k = p+1, \dots, \infty \tag{4.54c}$$

The observer gain G is obtained by solving LS of the relation between the observability matrix of the system and the observer gain Markov parameters

$$G = (\mathcal{O}^T\mathcal{O})^{-1}\,\mathcal{O}^T Y^o \tag{4.55}$$

where

$$\mathcal{O} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^N \end{bmatrix}, \qquad Y^o = \begin{bmatrix} Y_1^o \\ Y_2^o \\ Y_3^o \\ \vdots \\ Y_{N+1}^o \end{bmatrix} = \begin{bmatrix} CG \\ CAG \\ CA^2G \\ \vdots \\ CA^NG \end{bmatrix} = \mathcal{O}G \tag{4.56}$$

Integer N is chosen sufficiently large in order to have the observability matrix $\mathcal{O}$ of rank equal to the order n of the identified system. The set of equations i.e. Eq. (4.54) can be written in compact matrix form:

$$\begin{bmatrix} I & & & & \\ \overline{Y}_1^{(2)} & I & & & \\ \overline{Y}_2^{(2)} & \overline{Y}_1^{(2)} & I & & \\ \vdots & \vdots & \vdots & \ddots & \\ \overline{Y}_k^{(2)} & \overline{Y}_{k-1}^{(2)} & \overline{Y}_{k-2}^{(2)} & \dots & I \end{bmatrix} \begin{bmatrix} Y_1^o \\ Y_2^o \\ Y_3^o \\ \vdots \\ Y_{k+1}^o \end{bmatrix} = \begin{bmatrix} \overline{Y}_1^{(2)} \\ \overline{Y}_2^{(2)} \\ \overline{Y}_3^{(2)} \\ \vdots \\ \overline{Y}_{k+1}^{(2)} \end{bmatrix} \tag{4.57}$$

where I and $\overline{Y}_i^{(2)}$ (for $i = 1, 2, \dots, k+1$) are $m \times m$ square matrices. It is noted from Eq. (4.57) that recursive back substitution for $Y_1^o, Y_2^o, \dots., Y_{k+1}^o$ gives Eq. (4.42). Hence as from Eq. (4.54), the number p specifies the maximum number of observer gain Markov parameters Y^o which are uniquely computed from the identified set of p observer Markov parameters. For computational purpose, either Eqs. (4.42) and (4.54) or Eqs. (4.49) and (4.57) can be written together in a compact matrix form in order to solve for both the system Markov parameters Y_k and the observer gain Markov parameters Y_k^o simultaneously as:

$$P_k = [\,Ɏ_k \;\; Y_k^o] = [CA^{k-1}B \quad CA^{k-1}G] = CA^{k-1}[B \quad G]$$

$$= \left[\overline{Ɏ}_k^{(1)} - \overline{Ɏ}_k^{(2)}D \quad \overline{Ɏ}_k^{(2)}\right] - \sum_{i=1}^{k-1} \overline{Ɏ}_i^{(2)}[\,Ɏ_{k-i} \quad Y_{k-i}^o] \qquad \text{for } k = 1, 2, \dots, l \quad (4.58)$$

4.5 The Outline of OKID Algorithm

The detailed steps of the observer Kalman filter identification (OKID) algorithm to implement are described below. The inputs to OKID algorithm are the input and output time histories like: $\{u(k)\}$ and $\{y(k)\}$ from the real-time experiment. The outputs from the algorithm are the system matrices $A, B, C, \& D$ and observer gain matrix G of the identified sate-space model.

1. Choose an arbitrary & sufficiently large integer p which indicates the number of identified observer Markov parameters as apparent from Eq. (4.27) or the maximum number of independent system Markov parameters as seen from Eq. (4.42). Generally the integer p is selected to be four or five times larger than the effective order of the system.
2. Observer Markov parameters $\overline{Ɏ}$ are obtained from the least-square solution of either Eq. (4.27) utilizing the block data matrices y and ∀ for zero initial conditions, or Eq. (4.34) utilizing $\bar{y}$ and $\overline{\forall}$ for nonzero initial conditions.
3. Both system Markov parameters $Ɏ_k$ as well as the observer gain Markov parameters Y_k^o are recovered from the identified observer Markov parameters using Eq. (4.58). In order to solve for more Markov parameters than the number of identified observer Markov parameters, the extra observer Markov parameters are set to zero.
4. The system matrices i.e., $A, B, C, \& D$ of the state-space system and its corresponding observer gain G are identified from already calculated sequence of the system and the observer gain Markov parameters by using Eigen-system Realization Algorithm (ERA) or ERA with data correlations (ERA/DC) method [38], [48].

5. Eventually, the modal parameters namely frequencies, dampings, and mode shapes (at the sensor points) of the realized model can be determined after having been transformed to modal coordinates.

4.6 Relationship of State-space Observer and Kalman Filter

The generic discrete-time Kalman filter (1960) is actually a linear estimator which finds optimal estimates of states under the assumptions that state variables and the input are Gaussian distributions with zero mean, and disturbances and measurement noise are uncorrelated and white in nature. A necessary prerequisite for computing steady-state Kalman filter gain for LTI dynamic system is that the state-space model as well as process noise and measurement noise covariances must be known a priori. In practice, the measurement noise co-variance may be estimated by analyzing the sensors outputs, while the process noise comprising the system uncertainties and input noise is obtained by some educated guess. In order to relate the identified state-space linear observer and Kalman filter, the identified state-space representation, i.e. Eqs. (4.12) and (4.13) can be written for both the deterministic plus stochastic linear dynamic system as:

$$x(k+1) = Ax(k) + Bu(k) + w(k) \tag{4.59}$$

$$y(k) = Cx(k) + Du(k) + v(k) \tag{4.60}$$

where $w(k)$ and $v(k)$ are assumed to be uncorrelated, Gaussian, zero-mean, and white random processes with covariance matrices Q and R respectively. The former is called state or process noise acting on the system states while the latter is known as measurement or output noise acting on the system outputs, respectively. The Kalman filter in innovation form can be written as:

$$\hat{x}(k+1) = A\hat{x}(k) + Bu(k) + Ke(k) \tag{4.61}$$

$$\hat{y}(k) = C\hat{x}(k) + Du(k) \tag{4.62}$$

$\hat{x}(k)$, $\hat{y}(k)$, and K represent the estimated state, estimated output, and the Kalman gain. The

term $e(k)$ is called the residual which is the usual Gaussian, zero-mean, and white noise process. The residual is actually gained from the difference between the real measurement $y(k)$ and the estimated measurement $\hat{y}(k)$ i.e., $e(k) = y(k) - \hat{y}(k)$ and is also known as *innovation* in the context of Kalman filtering. The Kalman filter in innovation form is a state-space model having inputs comprising of the input to the original system $u(k)$ as well as the sequence of the output residual $e(k)$. From Eqs. (4.61) and (4.62), the following estimated state equation is obtained

$$\begin{aligned}\hat{x}(k+1) &= A\hat{x}(k) + Bu(k) + K[y(k) - \hat{y}(k)] \\ &= A\hat{x}(k) + Bu(k) + K[y(k) - C\hat{x}(k) - Du(k)] \\ &= [A - KC]\hat{x}(k) + [B - KD]u(k) + Ky(k) \\ \hat{x}(k+1) &= \tilde{A}\hat{x}(k) + \tilde{B}v(k)\end{aligned} \tag{4.63}$$

where

$$\tilde{A} = [A - KC], \qquad \tilde{B} = [B - KD \quad K], \qquad v(k) = \begin{bmatrix} u(k) \\ y(k) \end{bmatrix}$$

From definition of residual and Eq. (4.62), the measurement equation can be written

$$y(k) = C\hat{x}(k) + Du(k) + e(k) \tag{4.64}$$

A comparison of Eq. (4.63) with the observer model i.e., Eq. (4.22) reveals that they are identical provided that $G = -K$ and $e(k) = 0$. The Markov parameters are also same for both these equations. Now, the conditions will be figured out for the situation when $K = -G$ as follows [37], [38], [45], [49]. So, we can write from Eqs. (4.63) and (4.64),

$$\begin{aligned}\hat{x}(k+1) &= \tilde{A}\hat{x}(k) + \tilde{B}v(k) \\ \hat{x}(k+2) &= \tilde{A}\hat{x}(k+1) + \tilde{B}v(k+1) \\ &= \tilde{A}^2\hat{x}(k) + \tilde{A}\tilde{B}v(k) + \tilde{B}v(k+1) \\ &\vdots \qquad\qquad\qquad \vdots \\ \hat{x}(k+p) &= \tilde{A}\hat{x}(k+p-1) + \tilde{B}v(k+p-1)\end{aligned}$$

$$= \tilde{A}^p \hat{x}(k) + \tilde{A}^{p-1}\tilde{B}v(k) + \tilde{A}^{p-2}\tilde{B}v(k+1) + \cdots$$

$$+\tilde{B}v(k+p-1) \tag{4.65}$$

Incorporating Eq. (4.65) into Eq. (4.64) gives

$$y(k+p) = C\hat{x}(k+p) + Du(k+p) + e(k+p)$$

$$= C\tilde{A}^p \hat{x}(k) + C\tilde{A}^{p-1}\tilde{B}v(k) + C\tilde{A}^{p-2}\tilde{B}v(k+1) + \cdots$$

$$+C\tilde{B}v(k+p-1) + Du(k+p) + e(k+p) \tag{4.66}$$

The set of equations for $k = 0,1, \ldots, l-1$ of Eq. (4.66) in compact matrix form is given as:

$$\bar{y} = C\tilde{A}^p \hat{x} + \tilde{\Upsilon}\,\bar{\forall} + e \tag{4.67}$$

where e is the residual matrix, l is the data length. Also, $\bar{y}$ and $\bar{\forall}$ given in Eqs. (4.32) & (4.33) are again written as below:

$$\bar{y} = [y(p) \;\; y(p+1) \;\; y(p+2) \;\; \ldots \;\; y(l-1)]$$

$$\hat{x} = [\hat{x}(0) \;\; \hat{x}(1) \;\; \hat{x}(2) \;\; \ldots \;\; \hat{x}(l-p-2)]$$

$$\tilde{\Upsilon} = [D \;\; C\tilde{B} \;\; C\tilde{A}\tilde{B} \;\; \ldots \; C\tilde{A}^{p-1}\tilde{B}]$$

$$\bar{\forall} = \begin{bmatrix} u(p) & u(p+1) & \ldots & u(l-1) \\ v(p-1) & v(p) & \ldots & v(l-2) \\ v(p-2) & v(p-1) & \ldots & v(l-3) \\ \vdots & \vdots & \vdots & \ddots \\ v(0) & v(1) & \ldots & v(l-p-1) \end{bmatrix}$$

$$e = [e(p) \;\; e(p+1) \;\; e(p+2) \;\; \ldots \;\; e(l-1)]$$

If the Eq. (4.67) is post-multiplied by $\bar{\forall}^T$, we have

$$\bar{y}\;\bar{\forall}^T = C\tilde{A}^p \hat{x}\,\bar{\forall}^T + \tilde{\Upsilon}\,\bar{\forall}\;\bar{\forall}^T + e\,\bar{\forall}^T \tag{4.68}$$

Eq. (4.68) can be written in a more elaborate way as

$$\left[\bar{y}\,v_p^T \;\; \bar{y}\,v_{p-1}^T \;\ldots\; \bar{y}\,v_0^T\right] - \tilde{\Upsilon}\begin{bmatrix} v_p v_p^T & v_p v_{p-1}^T & \cdots & v_p v_0^T \\ v_{p-1}v_p^T & v_{p-1}v_{p-1}^T & \cdots & v_{p-1}v_0^T \\ \vdots & \vdots & \ddots & \vdots \\ v_0 v_p^T & v_0 v_{p-1}^T & \cdots & v_0 v_0^T \end{bmatrix} =$$

$$\left[e\ v_p^T\ \ e\ v_{p-1}^T\ \ \ldots\ e\ v_0^T\right] + C\tilde{A}^p\left[\hat{x}\ v_p^T\ \ \hat{x}\ v_{p-1}^T\ \ \ldots\ \hat{x}\ v_0^T\right] \tag{4.69}$$

Considering the ergodicity of the abovementioned stationary random process, the conversion from the time average to an expected value is carried out for each term from the products i.e. $e\ \overline{\forall}^T,\ \bar{y}\ \overline{\forall}^T,\ \hat{x}\ \overline{\forall}^T$, and $\overline{\forall}\ \overline{\forall}^T$ for $i,\ j = 0,1,\ldots p$. Let's take a term from $e\ \overline{\forall}^T$ as:

$$e\ v_i^T = \sum_{j=0}^{l-p-1} e(p+j)\ v^T(i+j)$$

$$= \sum_{k=p}^{l-1} e(k)\ v^T(k-p+i) \qquad \text{for } i,j = 0,1,\ldots p \tag{4.70}$$

The term $e\ v_i^T$ of Eq. (4.70) can be written as its ensemble average provided $l \to \infty$ according to ergodicity property as:

$$E[e(k)\ v^T(k-p+i)] = \lim_{l\to\infty} \frac{1}{l-p} \sum_{k=p}^{l-1} e(k)\ v^T(k-p+i) \quad k > p \tag{4.71}$$

The data must be collected after the system transients are decayed by allowing sufficient time and integer p in Eq. (4.67) is selected sufficiently large. It shows that the transients of the Kalman filter are very small and can be discarded i.e. $C\tilde{A}^p\hat{x} \approx 0$; so Eq. (4.68) will have the form

$$\lim_{l\to\infty} \frac{1}{l-p} \left[\bar{y}\ \overline{\forall}^T - \widetilde{\Upsilon}\ \overline{\forall}\ \overline{\forall}^T\right] = E[e(k)v^T(k)\ \ e(k)v^T(k-1)\ldots e(k)v^T(k-p)] + \ldots$$

$$C\tilde{A}^p E[\hat{x}(k)v^T(k+p)\ \ \hat{x}(k)v^T(k+p-1)\ \ \ldots\ \hat{x}(k)v^T(k)] \quad \forall\ k > p \tag{4.72}$$

If the observer is chosen to be such it satisfy the following equation for the limit $l \to \infty$

$$\widetilde{\Upsilon} = \bar{y}\ \overline{\forall}^T\ [\overline{\forall}\ \overline{\forall}^T]^{-1} \tag{4.73}$$

From Eqs. (4.72) & (4.73)

$$E[e(k)v^T(k)\ \ e(k)v^T(k-1)\ldots e(k)v^T(k-p)]$$

$$= -\,C\tilde{A}^p E[\hat{x}(k)\ v^T(k+p)\ \ \hat{x}(k)\ v^T(k+p-1)\ \ \ldots\ \hat{x}(k)v^T(k)] \tag{4.74}$$

As matrix $\tilde{A}$ is asymptotically stable for an observer; therefore, the right-hand side of Eq. (4.74)

is disregarded for sufficiently large integer p, i.e.

$$E[e(k)v^T(k)\ \ e(k)v^T(k-1)\ldots e(k)v^T(k-p)] = 0 \qquad k > p \tag{4.75}$$

Eq. (4.75) implies that

$$E[e(k)\,u^T(k-i)] = 0; \qquad i = 0, 1, \ldots, p \quad \& \quad k > p \tag{4.76}$$

$$E[e(k)\,y^T(k-i)] = 0; \qquad j = 1, 2, \ldots, p \quad \& \quad k > p \tag{4.77}$$

Therefore, for an observer with the observer Markov parameters satisfying the least-squares i.e. Eq. (4.73) provided the inverse $[\bar{\forall}\ \bar{\forall}^T]^{-1}$ exists, the residual $e(k)$ is orthogonal to the given input and the measured output with the time delay. For the data obtained from the finite dimensional deterministic plus stochastic linear dynamic system as given in Eqs. (4.59) and (4.60), there is Kalman filter available having the property that the residual is zero-mean Gaussian white as described below:

$$E[e(k)] = 0 \qquad e(k)\,e^T(k) = 0 \qquad \text{for } j \neq k \tag{4.78}$$

the residual of the Kalman filter satisfies the principle of orthogonality

$$e(k)\,y^T(k-i) = 0 \qquad \text{for } i = 1,2,\ldots,k \tag{4.79}$$

The Kalman filter gain would be constant for data length $l \to \infty$, it means that the experimental process is assumed to be long enough, statistically stationary, and random. Hence, the Kalman filter Markov parameters would satisfy the least-squares in Eq. (4.73) provided the inverse $[\bar{\forall}\ \bar{\forall}^T]^{-1}$ exists. Thus any identified observer is like a Kalman filter if the data length l as well as the order of that observer is chosen to be sufficiently large. In conclusion, the observer gain G calculated from Eq. (4.58) yields the steady-state Kalman filter gain, i.e. $K = -G$. But in practice, the resultant identified observer is not the Kalman filter due to the disturbances, nonlinearity, and non-whiteness of the process and measurement noises.

4.7 Observable Canonical-form Realization

Besides the minimum realization technique that generates a controllable and observable model, the observable canonical form realization is also used for determining the system matrices $A, B, C,$ and D directly from observer Markov parameters without first calculating the system Markov parameters. Considering p be the number of available observer Markov parameters calculated from the input and output data, m being the number of outputs, and r being the number of inputs. Therefore, from Eqs. (4.37) & (4.38), we have $p + 1$ identified observer Markov parameters

$$\overline{Ɏ} = [D \quad \left[\overline{Ɏ}_1^{(1)} - \overline{Ɏ}_1^{(2)}\right] \quad \left[\overline{Ɏ}_2^{(1)} - \overline{Ɏ}_2^{(2)}\right] \ \ldots \ [\overline{Ɏ}_p^{(1)} - \overline{Ɏ}_p^{(2)}] \tag{4.80}$$

Eq. (4.27) can be expanded and shifted by p time steps that results in the finite-difference model like ARX model. Consequently it takes the form of input-output map of the dynamic system as

$$\begin{aligned} &y(k+p) + \overline{Ɏ}_1^{(2)} y(k+p-1) + \overline{Ɏ}_2^{(2)} y(k+p-2) + \cdots + \overline{Ɏ}_p^{(2)} y(k) \\ &= Du(k+p) + \overline{Ɏ}_1^{(1)} u(k+p-1) + \overline{Ɏ}_2^{(1)} u(k+p-2) + \cdots + \overline{Ɏ}_p^{(1)} u(k) \end{aligned} \tag{4.81}$$

Let's define the state variables as vectors of length m i.e. $x_i(k)$, for $i = 1,2, \ldots, p$ as given [38]:

$$\begin{aligned} x_p(k) &= y(k) - Du(k) \\ x_{p-1}(k) &= y(k+1) - Du(k+1) + \overline{Ɏ}_1^{(2)} y(k) - \overline{Ɏ}_1^{(1)} u(k) \\ x_{p-2}(k) &= y(k+2) - Du(k+2) + \overline{Ɏ}_1^{(2)} y(k+1) - \overline{Ɏ}_1^{(1)} u(k+1) + \overline{Ɏ}_2^{(2)} y(k) \\ &\quad - \overline{Ɏ}_2^{(1)} u(k) \\ &\vdots \qquad\qquad \vdots \qquad\qquad \vdots \\ x_1(k) &= y(k+p-1) - Du(k+p-1) + \overline{Ɏ}_1^{(2)} y(k+p-2) - \\ &\quad \overline{Ɏ}_1^{(1)} u(k+p-2) + \overline{Ɏ}_2^{(2)} y(k+p-3) - \overline{Ɏ}_2^{(1)} u(k+p-3) \\ &\quad \vdots \end{aligned} \tag{4.82}$$

$$\overline{\Upsilon}_{p-1}^{(2)} y(k) - \overline{\Upsilon}_{p-1}^{(1)} u(k)$$

The set of Eq. (4.82) gives the following equations:

$$\begin{aligned} y(k) &= x_p(k) + Du(k) \\ x_{p-1}(k) &= x_p(k+1) + \overline{\Upsilon}_1^{(2)} y(k) - \overline{\Upsilon}_1^{(1)} u(k) \\ x_{p-2}(k) &= x_{p-1}(k+1) + \overline{\Upsilon}_2^{(2)} y(k) - \overline{\Upsilon}_2^{(1)} u(k) \\ \vdots \quad & \quad \vdots \quad \quad \vdots \\ x_1(k) &= x_2(k+1) + \overline{\Upsilon}_{p-1}^{(2)} y(k) - \overline{\Upsilon}_{p-1}^{(1)} u(k) \end{aligned} \tag{4.83}$$

and with the incorporation of Eq. (4.82), the last equation in Eq. (4.81) yields the following:

$$x_1(k+1) = -\overline{\Upsilon}_p^{(2)} y(k) + \overline{\Upsilon}_p^{(1)} u(k)$$

Similar to the Eqs. (4.12) and (4.13), the above-mentioned equations are arranged in the compact matrix form which is called the canonical-form; for which the state vector $x \in \Re^{mp\times 1}$, the state matrix $A \in \Re^{mp\times mp}$, the input matrix $B \in \Re^{mp\times r}$, and the output matrix $C \in \Re^{m\times mp}$ are given in the set of Eqs. (4.84) as:

$$\left.\begin{aligned} x(k) &= \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \\ \vdots \\ x_{p-1}(k) \\ x_p(k) \end{bmatrix}, \qquad A = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 & -\overline{\Upsilon}_p^{(2)} \\ I & 0 & 0 & \dots & 0 & -\overline{\Upsilon}_{p-1}^{(2)} \\ 0 & I & 0 & \dots & 0 & -\overline{\Upsilon}_{p-2}^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & -\overline{\Upsilon}_2^{(2)} \\ 0 & 0 & 0 & \dots & I & -\overline{\Upsilon}_1^{(2)} \end{bmatrix}, \\ B &= \begin{bmatrix} \overline{\Upsilon}_p^{(1)} - \overline{\Upsilon}_p^{(2)} D \\ \overline{\Upsilon}_{p-1}^{(1)} - \overline{\Upsilon}_{p-1}^{(2)} D \\ \overline{\Upsilon}_{p-2}^{(1)} - \overline{\Upsilon}_{p-2}^{(2)} D \\ \vdots \\ \overline{\Upsilon}_2^{(1)} - \overline{\Upsilon}_2^{(2)} D \\ \overline{\Upsilon}_1^{(1)} - \overline{\Upsilon}_1^{(2)} D \end{bmatrix}, \qquad C = [0 \quad 0 \quad 0 \ \dots 0 \quad I], \quad D = \overline{\Upsilon}_0, \end{aligned}\right\} \tag{4.84}$$

The observability matrix $\mathcal{O}$ of the canonical-form given in Eq. (4.84) is as follows:

$$\mathcal{O} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{p-1} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 & 0 & I \\ 0 & 0 & 0 & \dots & 0 & I & -\overline{\Upsilon}_1^{(2)} \\ 0 & 0 & 0 & \dots & I & -\overline{\Upsilon}_1^{(2)} & -\overline{\Upsilon}_2^{(2)} + \overline{\Upsilon}_1^{(2)}\overline{\Upsilon}_1^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & I & \times & \dots & \times & \times & \times \\ I & \times & \times & \dots & \times & \times & \times \end{bmatrix} \tag{4.85}$$

where " $\times$ " shows some no-zero elements. The observability matrix $\mathcal{O}$ has full rank of mp, the dimension of realized sate matrix A, which means that it is non-singular for any $\overline{\Upsilon}_i^{(2)}$; and hence, all states are observable. A unique sequence of system Markov parameters can be computed from the canonical-form model given in Eq. (4.84) as follows:

$$\begin{aligned} \Upsilon_1 &= CB = \overline{\Upsilon}_1^{(1)} - \overline{\Upsilon}_1^{(2)} D \\ \Upsilon_2 &= CAB = \overline{\Upsilon}_2^{(1)} - \overline{\Upsilon}_2^{(2)} D - \overline{\Upsilon}_1^{(2)} \Upsilon_1 \\ \Upsilon_3 &= CA^2 B = \overline{\Upsilon}_3^{(1)} - \overline{\Upsilon}_3^{(2)} D - \overline{\Upsilon}_1^{(2)} \Upsilon_2 - \overline{\Upsilon}_2^{(2)} \Upsilon_1 \\ &\vdots \qquad\qquad \vdots \qquad\qquad \vdots \end{aligned} \tag{4.86}$$

The integer p is selected in a manner so that mp is less than or equal to the order of the system, then the triplet $[A, B, C]$ obtained through the above system Markov parameters in Eq. (4.86) is a minimum realization. Therefore, in this context the state-space model as given in Eq. (4.84) is thus called as the *observable canonical-form*. One usual drawback of this model is that the order of the system at hand is required to be known a priori, which is normally not known in real practice. Therefore, the best model order is determined by a trial and error technique in this scenario.

CHAPTER 5

LINEAR QUADRATIC TRACKING CONTROL OF ODU-ASV

5.1 Introduction

Nowadays autonomous vehicles such as ASVs have been gaining the significant attention for their ability to perform variety of autonomous tasks. For this purpose, modern control techniques are being used to design autopilots, stability augmentation systems (SASs), and other controller designs to achieve unmanned tasks of the autonomous vehicles. The main purpose of the control system design is the regulation as well as tracking control of the desired states of the dynamical system such as ASV in such a way that its desirable closed-loop response characteristics are obtained. This can be attained by placing the closed-loop poles at desirable locations on the pole-zero map without too much control input. In classical control theory, successive loop closure strategy has to be followed in which each loop is closed separately one at a time and as a result control gains are determined; so in this way, the multivariable control system can be designed. But pole placement technique for the MIMO control system design could usually be badly conditioned when unrealistic pole locations are considered. That is why the alternate modern state-space control designs techniques, such as linear quadratic regulator (LQR) control and the extended version of LQR, i.e. the optimal linear quadratic tracking (LQT) control, are used for optimal control of dynamical systems. The LQR control design technique in terms of mathematically precise performance index produces matrix equations that can be solved for computing the control gains by closing all the loops at the same time [35], [50]. The LQR and optimal LQT are

more powerful control design techniques for many linear MIMO as well as SISO systems. In the forthcoming sections, the design procedures for both the discrete-time LQR as well as optimal discrete LQT control techniques are discussed.

The desired motion of ASV is normally achieved by any of the following motion control strategies such as: station-keeping, trajectory-tracking, and path-following. The station-keeping permits the ASV system to keep the position and or heading constant over the time interval. In the trajectory tracking control strategy, full-state tracking of the given reference states is performed while path-following is the reduced-state tracking approach and particularly the spatial tracking. The path-following or way-points tracking is actually the decoupling between space and time which means the separation of velocity and two-dimensional (2D) path. The velocity, if considered in the design, is actually an additional DOF for time coordination [4], [7], [9], [33]. In the path-following control approach for the ASV, the predefined time-independent 2D spatial paths are followed with the help of designed control. As the ODU-ASV is an under-actuated system having twin thrusters for its propulsion, so the challenge for under-actuation is to reject almost all the disturbances which is quite difficult task with only two inputs in the form of two thrusters. Therefore for this purpose, the optimal discrete-time LQT control is designed and implemented for the path-following motion control of the ASV in the present work.

5.2 The Linear Quadratic Regulator Control

Due to the increase in hardware complexity and its ensuing costs as well as the decrease in reliability, the nonlinear models of mechanical/physical systems are either linearized about an operating point for having an approximate linear plant model or linear models are identified by various system identification techniques for those systems. Therefore, for the linear state-space plant models, traditional linear controller, such as LQR, could be designed in order to ensure the

plant state to be in the close vicinity of the operating point. The LQR or commonly known as optimal state-feedback control design is nothing more than the solution to a convex, least squares optimization problem that has some very attractive properties, namely the optimal controller automatically ensures a stable closed-loop system, which achieves guaranteed levels of stability robustness, and is simple to compute [35], [51].

5.2.1 Steady-State LQR

Consider the dynamics of continuous time-invariant multivariable linear system as given:

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{5.1}$$

with state vector $x(t) \in \mathbb{R}^n$ and control input $u(t) \in \mathbb{R}^r$. The control input is selected in such a manner that quadratic cost function is minimized, which is given as:

$$J = \frac{1}{2}\int_0^T \left[x^T(t)Qx(t) + u^T(t)Ru(t)\right] dt \tag{5.2}$$

with symmetric positive semi definite $Q \geq 0$ and symmetric positive definite $R > 0$ are the design parameter weight matrices to penalize the states and control inputs, respectively, of the linear system. These weight matrices can be tuned iteratively until the required performance is achieved [35]. It is assumed that all the states of linear system of Eq. (5.1) are available for feedback and $[A\ B]$ is stabilizable and $[A\ C]$ is detectable; then a unique, time-varying state-feedback control law is actually the linear quadratic controller [35] as:

$$u(t) = -Kx(t) \tag{5.3}$$

where

$$K(t) = R^{-1}B^T P(t) \tag{5.4}$$

is the time-varying gain matrix that minimizes the performance index (PI) or cost function J, subject to the dynamic constraints of the open-loop dynamics as given in Eq. (5.1). Here $P(t)$ is the solution of the Riccati differential equation (RDE) as:

$$A^T P(t) + P(t)A - P(t)BR^{-1}B^T P(t) + Q = -\frac{d}{dt}P(t) \tag{5.5}$$

If time horizon of the integration is indefinite i.e. $T \to \infty$, then Eq. (5.2) is called an infinite horizon performance index and an optimal solution exists; then $P(t)$ tends to be constant matrix$P$, where $\dot{P} = 0$. The RDE given in Eq. (5.5) is replaced by Algebraic Riccati Equation (ARE) as follows:

$$A^T P + PA - PBR^{-1}B^T P + Q = 0 \tag{5.6}$$

where symmetric positive semi-definite matrix $P \geq 0$ is the unique and steady-state solution to the algebraic Riccati equation (ARE). Then, time-varying gain matrix given in Eq. (5.4) is constant and is called as the steady-state sub-optimal gain given as:

$$K_\infty = R^{-1}B^T P_\infty \tag{5.7}$$

Hence, the constant state-variable feedback is the sub-optimal steady-state control as follows:

$$u(t) = -K_\infty x(t) \tag{5.8}$$

For matrix P, the corresponding closed-loop system under the influence of the steady-state LQR control law is asymptotically stable and has the time-invariant dynamics as:

$$\dot{x} = (A - BK_\infty)x \equiv A_c x \tag{5.9}$$

The block diagram of the linear quadratic regulator is given in Fig. 5.1 below.

5.2.2 Design Outline of LQR

The control system specifications are not directly achieved from the LQR formulation, but rather an iterative trial and error technique is required for the selection of the weighting matrices Q and R in the cost in order to have a satisfactory controller. So, the LQR design process is generally initiated by selecting values for the design weights, then synthesizing the control law, and later evaluating the designed control law for the desired performance and robustness. It is necessary for the design of LQR control that full-state feedback must be

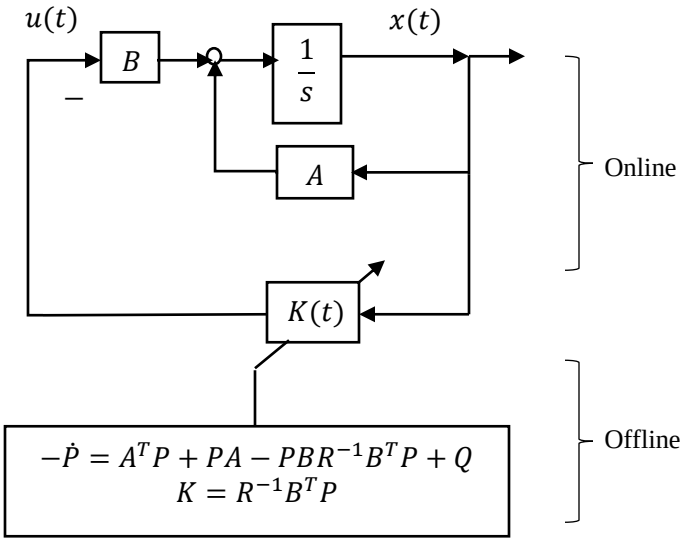

Figure 5.1 Schematic of linear quadratic regulator (LQR) control design.

available to be used, which is the main limitation of this methodology. The main design steps of the LQR control are as follows:

1. The optimal selection of the symmetric weighting matrices Q and R of the performance index (PI) has been the challenging topic for the control researchers. The selection process is an iterative process and so standard control system specifications are not directly achieved. But, there are different techniques, such as: trial & error technique, Bryson's approach, and evolutionary algorithm based methods i.e. the multi-objective genetic algorithm (MOGA), that are commonly used for determining the diagonal components of design weighting matrices Q and R in the performance index, so that the satisfactory controller performance can adequately be achieved. According to Bryson's rule, Q and R ma-

trices are determined using *maximum allowable deviations* in the component $x_i(t)$ of the state vector $x(t)$ as x_{iM} and the component $u_i(t)$ of the control input vector $u(t)$ as u_{iM}. Then the matrices Q and R are chosen as: $Q = diag\{q_i\}$ and $R = diag\{r_i\}$, such that

$$q_i = \frac{1}{(x_{i\,M})^2}, \qquad r_i = \frac{1}{(u_{i\,M})^2} \tag{5.10}$$

The state weighting matrix Q can also be selected as matrix $C^T C$, because this selection shows the weighting of the states in the output as: $y^T y = x^T C^T C x$. The diagonal elements of control weighting matrix R is selected to be equal weights on the control inputs. When $R = I$, the identity matrix, then a good compromise result can be achieved on the control input. Hence, we have as follows:

$$Q = \alpha\, C^T C \tag{5.11}$$

$$R = I \tag{5.12}$$

where α represents a scalar constant incorporated to decrease the states interaction. In this research, the MOGA technique is incorporated for the selection of Q and R matrices, the detail of which is given in Chapter-6.

2. Solve the ARE for symmetric positive semi-definite matrix P using Eq. (5.6)
3. Compute the sub-optimal state-feedback gain K_∞ using Eq. (5.7).
4. Then, find the required control input using Eq. (5.8) and by utilizing the gain obtained in the previous step.
5. Lastly, simulate the designed control system in order to check desired controller performance. If the controller performance is not satisfactory then repeat the design process from 1 to 4 until the desired controller performance is achieved.

5.2.3 Salient Properties of LQR Controller

There are some appealing properties of LQR controller which are briefly discussed as follows.

The steady-state LQR controller automatically guaranteed a stable close-loop system even if it is MIMO system, as long as the system satisfies some fundamental properties. Let the linear system of Eq. (5.1) have n states, then the system is said to be reachable only when the reachability matrix

$$\mathrm{R} = [B \;\; AB \;\; A^2B \; ... \; A^{n-1}B] \tag{5.13}$$

has rank n; which also implies to stabilizability. Also, the system is said to be observable only if the observability matrix

$$\mathcal{O} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{bmatrix} \tag{5.14}$$

has rank n, which implies detectability as well [35].

Theorem 5.1: If matrix C is a square root of Q i.e. $Q = C^TC$. Also, let (C, A) is detectable and (A, B) is stabilizable. Then:

(i) A unique symmetric positive-semi definite limiting solutionP_∞ exists to the ARE.

(ii) The closed-loop system $A_c = (A - BK_\infty)$ is asymptotically stable [35].

It is concluded from the above said Theorem 5.1 that the system of Eq. (5.1) and PI of Eq. (5.2) with $T \to \infty$ satisfy the fundamental controllability and observability requirements; hence, the steady-state LQR control will produce gains that stabilize the system. The theorem anticipates that closed-loop stability properties of the system in terms of open-loop system properties are determined utilizing matrix rank techniques. The detectability of matrix $(\sqrt{Q}, A)$ means

that all the unstable system modes should be weighted in the PI in terms of the weighting trix Q. If the $(\sqrt{Q}, A)$ is observable and the weighting matrices are selected as $Q \geq 0$ & $R > 0$, then the closed-loop poles will always be stable. Therefore, the suitable close-loop system performance is obtained through the selection of Q and R matrices during an interactive computer aided design of the control system and their diagonal elements may continuously be varied until the suitable closed-loop performance is achieved.

The LQR control needs simple computing steps for its design and then its implementation later on. Despite the robustness and easy to compute feature, there is main limiting fact of LQR controller that it requires full-states for feedback in order to work.

5.3 Design of Optimal Discrete Linear Quadratic Tracking Control

In this section, the discrete-time closed-loop linear quadratic optimal trajectory tracking control over the entire time interval $[0\ N]$ is designed for the motion control of inherently nonlinear time-invariant systems such as the ASV, robotic arm, and spacecraft etc. This kind of control design strategy is the extension of discrete linear quadratic regulator (LQR) control design technique and its results can also be generalized to the linear and time-varying case. The state-space model of the Eqs. (4.12) and (4.13), that characterizes the dynamical system as finite-dimensional discrete linear time-invariant system, is re-written:

$$x_{k+1} = Ax_k + Bu_k \tag{5.15}$$

where $x_k \in \mathbb{R}^n$, and some linear combination of the states are:

$$y_k = Cx_k \tag{5.16}$$

to follow already known reference trajectory r_k over the interval $k = [0\ N]$, while the performance index can be minimized as [34], [35], [50]:

$$J_0 = \frac{1}{2}(Cx_N - r_N)^T P\ (Cx_N - r_N) + \frac{1}{2}\sum_{k=0}^{N-1}[(Cx_k - r_k)^T Q\ (Cx_k - r_k) + u_k^T R u_k\ ] \tag{5.17}$$

where matrices $P \geq 0,\ Q \geq 0,\ R > 0$ and actual value of x_N is not constrained. The optimal control u can be achieved by solving the following state system as given in Eq. (5.15), co-state system, and stationarity condition given below. So the co-state system is given as:

$$\lambda_k = A^T\ \lambda_{k+1} + C^T Q C\ x_k - C^T Q\ r_k \tag{5.18}$$

where λ_k is the co-state and the stationarity condition:

$$0 = B^T\ \lambda_{k+1} + R\ u_k \tag{5.19}$$

with the boundary conditions as follows:

$$x_0 \qquad \text{(given)}$$

and $$\lambda_N = C^T P(Cx_N - r_N) \tag{5.20}$$

So the optimal control from stationarity condition of Eq. (5.19) is

$$u_k = -R^{-1}B^T\ \lambda_{k+1} \tag{5.21}$$

By eliminating u_k from Eq. (5.15), we can write discrete linear time-invariant system as:

$$x_{k+1} = Ax_k - BR^{-1}B^T\ \lambda_{k+1} \tag{5.22}$$

The coupled state and co-state equations, given as (5.18) and (5.22), can be written as single un-forced system known as nonhomogeneous discrete Hamiltonian system as:

$$\begin{bmatrix} x_{k+1} \\ \lambda_k \end{bmatrix} = \begin{bmatrix} A & -BR^{-1}B^T \\ C^T Q C & A^T \end{bmatrix} \begin{bmatrix} x_k \\ \lambda_{k+1} \end{bmatrix} + \begin{bmatrix} 0 \\ -C^T Q \end{bmatrix} r_k \tag{5.23}$$

The control law of Eq. (5.21) can't be practically implemented because the boundary conditions are divided between times $k = 0$ and $k = N$. Therefore, another practical approach is dug out. The control u_k can be described as combination of linear state feedback and a term depending on r_k by sweep method. From Eq. (5.20), we can assume that

$$\lambda_k = S_k x_k - v_k \tag{5.24}$$

where matrix $S_k \in \mathbb{R}^{n\times n}$ and vector $v_k \in \mathbb{R}^{n\times 1}$ are unknown auxiliary sequences. In order to find the consistent equation for S_k and v_k; let use Eq. (5.24) in the state equation portion of the

Eq. (5.23) as:

$$x_{k+1} = Ax_k - BR^{-1}B^T S_{k+1}x_{k+1} + BR^{-1}B^T v_{k+1} \tag{5.25}$$

Eq. (5.25) can be solved for x_{k+1} as:

$$x_{k+1} = (I + BR^{-1}B^T S_{k+1})^{-1}(Ax_k + BR^{-1}B^T v_{k+1}) \tag{5.26}$$

and from the co-state equation portion of Eq. (5.23) utilizing (5.24) and (5.26), we have:

$$S_k x_k - v_k = C^T QC\, x_k + A^T S_{k+1}\left(I + BR^{-1}B^T S_{k+1}\right)^{-1} \times \left(Ax_k + BR^{-1}B^T v_{k+1}\right) \ldots$$
$$-A^T v_{k+1} - C^T Q r_k \tag{5.27}$$

or

$$[-S_k + A^T S_{k+1}\left(I + BR^{-1}B^T S_{k+1}\right)^{-1} A + C^T QC]x_k + [v_{k+} A^T S_{k+1}\left(I + BR^{-1}B^T S_{k+1}\right)^{-1} \ldots$$
$$BR^{-1}B^T v_{k+1} - A^T v_{k+1} - C^T Q r_k] = 0 \tag{5.28}$$

Eq. (5.28) is valid for all state sequences x_k given any x_0, in order the terms in brackets must be vanished individually. From Eq. (5.28), the two sequences S_k and v_k can be written using the matrix inversion lemma as:

$$S_k = A^T[S_{k+1} - S_{k+1}B(B^T S_{k+1}B + R)^{-1}B^T S_{k+1}]A + C^T QC \tag{5.29}$$

and

$$v_k = [A^T - A^T S_{k+1}B(B^T S_{k+1}B + R)^{-1}B^T]v_{k+1} + C^T Q r_k \tag{5.30}$$

and if Eqs. (5.20) and (5.24) are compared, then the boundary conditions are given as:

$$S_N = C^T PC \tag{5.31}$$

$$v_N = C^T P\, r_N \tag{5.32}$$

So, the optimal control can be written as from Eq. (5.21)

$$u_k = -R^{-1}B^T\, \lambda_{k+1} = -R^{-1}B^T(S_{k+1}x_{k+1} - v_{k+1}) \tag{5.33}$$

From Eq. (5.33), it is evident that the control depends on the unknown x_{k+1} at time k; so state Eq. (5.15) is substituted into Eq. (5.33), and then pre-multiplied by R, then we have the required

control law as:

$$u_k = (B^T S_{k+1} B + R)^{-1} B^T (-S_{k+1} A x_k + v_{k+1}) \tag{5.34}$$

The affine formulation of control law given in Eq. (5.34) is more robust and can be written as:

$$u_k = -K_k x_k + K_k^v v_{k+1} \tag{5.35}$$

where K_k and K_k^v are state feedback and feedforward gains respectively as:

$$K_k = (B^T S_{k+1} B + R)^{-1} B^T S_{k+1} A \tag{5.36}$$

$$K_k^v = (B^T S_{k+1} B + R)^{-1} B^T \tag{5.37}$$

The diagram of optimal LQ tracker is given in Fig. 5.2 below; which is an affine state-feedback control law consisting of linear term of x_k and another term independent of x_k and whose gains are depend on the solution to the Riccati equation given in Eq. (5.29). The closed-loop system under the action of optimal LQT controller is nonhomogeneous time-varying system as [34]:

$$x_{k+1} = (A - BK_k) x_k + BK_k^v v_{k+1} \tag{5.38}$$

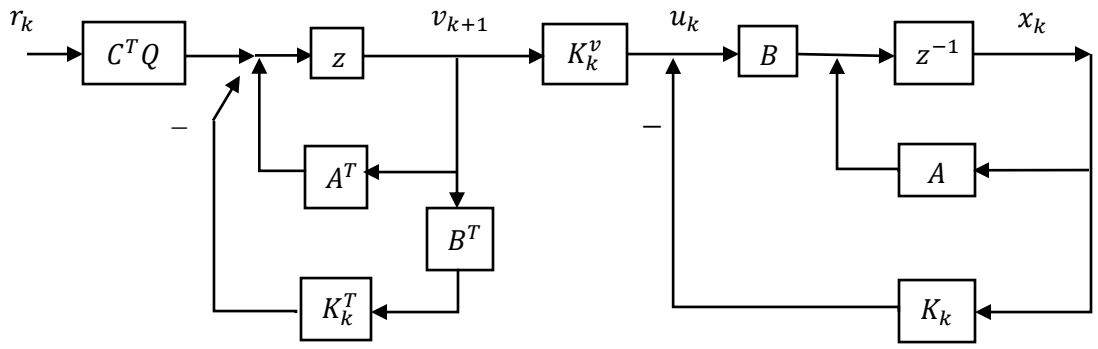

Figure 5.2 The schematic of linear quadratic tracking (LQT) control.

5.4 Outline of Optimal Discrete LQT Control

The outline of the formulation of optimal discrete LQT control is given as follows.

1. The linear time-invariant system identified as state-space, which is given in Eqs. (5.15) and (5.16), is considered, where the triplet of system matrices $A, B, C,$ is known.
2. The cost function is selected as given in Eq. (5.17), which needs to be minimized assuming the symmetric matrices as $P \geq 0, Q \geq 0, R > 0$. Where the symmetric matrices Q and R are called the weighting matrices and P is the solution to ARE given as Eq. (5.29). The techniques for selection of Q and R are briefly discussed in aforesaid section 5.2.2.
3. The feedback gain K_k is computed from Eq. (5.36) utilizing the boundary condition given in the Eq. (5.31).
4. The auxiliary sequence S_k is computed by the following relation:

$$S_k = A^T S_{k+1} (A - BK_k) + C^T Q C \tag{5.39}$$

5. The auxiliary sequence v_k is computed using boundary condition given in Eq. (5.32) as:

$$v_k = (A - BK_k)^T v_{k+1} + C^T Q r_k \tag{5.40}$$

6. The feedforward gain is determined as from Eq. (5.37).
7. Finally, the optimal affine state feedback control law is computed as given in Eq. (5.35).

5.5 Implementation of Optimal Linear Quadratic Tracker

The main steps of the optimal LQT control implementation in real-time are summarized as follows.

1. The desired track r_k is known in advance and stored in the computer's memory, where k is the index 0 to N, the total number of points of trajectory track.
2. The auxiliary sequence v_k is computed offline using Eq. (5.40) utilizing already known track r_k and boundary condition given in Eq. (5.32). The sequence v_k is stored in the

computer memory for later use.

3. As ARE does not depend on the state trajectory [34]; therefore, the sequence S_k as given in Eq. (5.39) as well as the LQT gains sequences K_k and K_k^v as given in Eqs. (5.36) and (5.37), respectively are computed offline and stored for later use during the control run.
4. During actual control run, the only work is to solve Eq. (5.35) for computing the optimal control u_k.

CHAPTER 6

EXPERIMENTAL RESULTS

6.1 Introduction

It is an essential requirement to develop a mathematical model for the ASV dynamical system in order to design and implement a feasible controller for its motion control. Likewise, the dynamics of complex systems such as aerial and subsurface vehicles, the motion of ASV is also very challenging to model because water is an ever-changing surface. The ASV is considered to be a rigid body, so its motion is actually six degrees of freedom (DOF): i.e. translation along three perpendicular axes and rotation about three perpendicular axes [4]. In this research, the vertical motion is not considered because the ASV is coasting on rather calm water and its motion is stable because of the twin-hull design, so it is assumed that its motion is characterized as 3DOF. For this purpose, experimental system identification technique known as the OKID is used to find the simplest state space model that will capture the dynamics of ASV system adequately. During the process of system identification, a trade-off between the model's fidelity and modeling effort is involved, which ensures the accuracy of the developed model, and the same practice has been exercised in this research. Once we identify the state space model for the ASV dynamics, the design of a controller for the motion of ASV and its implementation could be executable. Hence, the scope of this chapter encompasses important concepts for design of system identification and control for the ASV, which will be discussed onwards.

The System Observer Controller Identification Toolbox (SOCIT) developed at NASA by Jer-Nan Juang and Lucas G. Horta is a collection of MATLAB functions expressed in M-files,

which implements a variety of modern system identification techniques used for the identification of an open-loop system model and its corresponding forward and backward observers in the discrete time domain directly from the experimental input output data [48]. The identified model and the observer for state estimation can be utilized for controller design of linear systems as well as identification of modal parameters such as dampings, frequencies, and mode shapes. The optimal discrete closed-loop linear quadratic tracking (LQT) control methodology [34], [35] is applied to the identified linear model of ASV so that it follows a desired known reference path over a time interval. In this control strategy, optimal control is determined by combining linear state variable feedback plus a term depending on reference track. The state variable feedback gains are obtained by minimizing a cost function utilizing weighting matrices, which are found by genetic algorithm (GA) based multi-objective optimization technique. The results of simulations and later on validation water-trial support the validity of the OKID identified model and its proposed control design for path following motion of ASV. The experimental setup and the reference coordinate frames for system identification and control design processes are explained in sections 6.2 and 6.3, respectively. The process of open-loop OKID process and the subsequent results are detailed in section 6.4; in section 6.5, the benchmark assessment tests for the identified model are given and the concluding discussion regarding OKID identified model is given in section 6.6. The design of optimal LQT control and its simulation results are explained in sections 6.7 and 6.8, respectively. The discussion related to LQT control design is given in section 6.9 briefly. The software and hardware implementation process is discussed in section 6.10; in this section, mapping of control signal or manipulated variables obtained from the controller and the actual PWM required for the two outboard motors was elaborated. Then, the real-time validation testing of the proposed closed-loop controller while applying path-following control strategy

of ASV and its subsequent results are given. Finally, the concluding remarks are given in section 6.11.

6.2 Experimental Setup

As discussed in chapter 2, the ASV utilized in this research was already designed by ODU-ASV team and has comparatively small dimensions so that it could be operated in rather calm and shallow waters of low wave conditions such as river and estuarine environments. The open-loop system identification water-trials and stable autonomous maneuvers of ASV demand various kinds of sensors such as a magnetometer, accelerometer, gyroscope, LIDAR, and/or GPS that provide informative data regarding the vehicle motion. Hence, the necessary experimental layout for system identification and control design consists of on-board embedded control system console and ground control station (GCS) which is described in detail as following:

6.2.1 On-board Embedded Control System Console

A custom built on-board embedded control system console is made by the ODU-ASV team and installed on the deck of the ASV for controlling its motion either in remote control or unmanned/autonomous modes. This electronics console is comprised of the main processing computer for guidance, navigation, & control, Arduino Mega 2560 microcontroller boards, onboard pose sensors such as IMU & GPS receiver both installed at CG point, radio receiver for remote control, and Ubiquiti M_2 bullets as the outdoor wireless radio networking device. The IMU and GPS receiver are installed on the deck of the ASV in such a way that the GPS view is clear to the open sky and signals of IMU are not interfere with by the magnetic field of the adjacent wiring of the embedded control system console as shown in Fig. 6.1. The Memsense Nano inertial measurement unit (nIMU) was selected to be used for yaw and yaw rate data of the ASV because of its small footprint of 4.65 cm × 2.28 cm and its weight of about 20 g. The nIMU pro-

vides inertial data at sampling rate of 150 Hz from orthogonal triads of magnetometers, gyroscopes, and accelerometers which measure local magnetic fields $\boldsymbol{h}_{o_I}^{o}$, body-fixed rotational velocities $\boldsymbol{\omega}_{o_I}^{o}$, and local accelerations $\boldsymbol{a}_{o_I}^{o}$ respectively. From this data, the yaw and yaw rate of ASV are obtained through MATLAB® code.

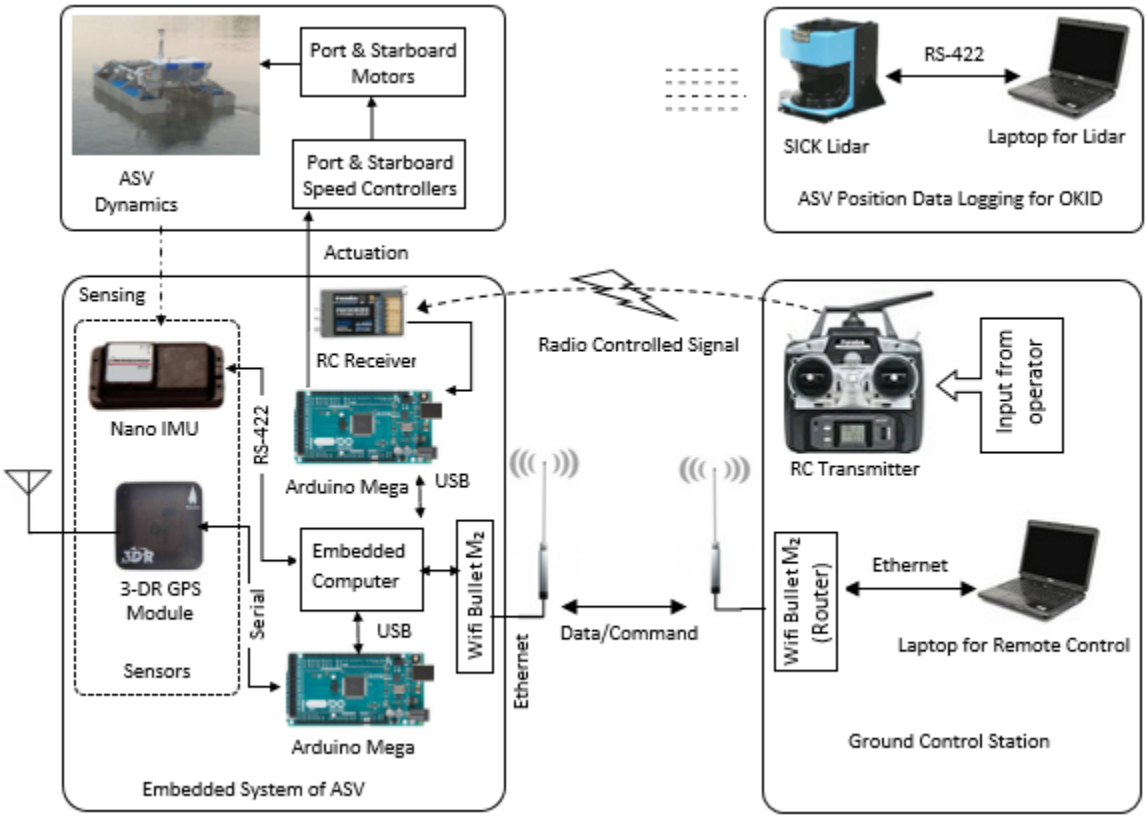

Figure 6.1 Schematic diagram illustrating experimental setup of ODU-ASV.

6.2.2 Ground Control Station

The ground control station (GCS) consists of a laptop computer having the Intel *core-i5*

microprocessor, an Ubiquiti Bullet M_2 as wireless radio networking router having LAN antenna, and R/C transmitter i.e. Futaba: 6-channel radio controlled system as shown in Figure 6.1. The laptop computer of GCS is remotely connected with the onboard main computer of ASV through the Bullet M_2 for its remote control as well as unmanned/autonomous maneuvers. Besides GCS, the ASV pose was continuously monitored by incorporating the Sick LMS-200 Lidar located alongside GCS so that the position data in Cartesian coordinates were logged to be used for open loop system identification tasks. The working range of Lidar is 16 m accurately, so the outdoor water-trials were performed within 16 m semicircular area. The ASV is navigated on the water surface with the help of Fatuba R/C transmitter remotely during manual mode. The ASV can be switched between the two modes of operation, i.e. autonomous and remote control, by using a reserved toggle switch channel on the R/C transmitter. Moreover, during the autonomous control mode, commands in the form of PWM voltage values are transmitted to the two outboard motors could be overridden instantly by a human operator through R/C transmitter remotely in order to switch the control of ASV back to the remote control mode. The same switching strategy is coded into the drive Arduino microcontroller board which is connected to the pins of R/C receiver. The provision of remotely override is very crucial for safe operation of the ASV which permits the human operator to either navigate the ASV in remote control mode or abruptly halt its motion altogether in the scenario of unpredicted maneuver.

6.2.3 Center of Gravity of ASV

As pose estimation sensors such as nIMU and GPS render information about the absolute or relative position and orientation of ASV, therefore it is necessary that they are installed at the center of gravity (CG) of the ASV. As the ODU-ASV is considered to be widthwise symmetrical regarding the distribution of mass, the term CG is used in a synonymous manner with the center

of mass (CM), regardless of the fact that CG point does not coincide with the CM point technically if the ASV does not have a symmetrical distribution of mass [4]. In the body-fixed reference frame as detailed in section 6.3.1 below, the CG is located at some point i.e. $r_{cg} = (x_{cg}\ 0\ z_{cg})^T$, which is calculated lengthwise for installation purpose of pose sensors as follows.

$$x_{cg} = (l_1 w_1 + l_2 w_2)/w_{cg} \tag{6.1}$$

where $l_1 = 4''$ and $l_2 = 44''$ are the distance measured along longitudinal x-axis from the aft vertical reference surface of the ASV's hull to the first and second hook-up point of the load cell, respectively; the weights w_1 and w_2 in pounds are measured by load cell at those particular points and $w_{cg} = w_1 + w_2$. The ASV is considered to be symmetrical widthwise; therefore all measurements are done amid of ASV along y-axis. So from Eq. 6.1, the CG is calculated to be located at $22.089''$ from the aft vertical reference surface of the ASV's hull.

6.3 Coordinate Systems and Transformations

Prior to performing real-time water-trials for open-loop system identification of the state-space model of the ASV and later on its real-time validation water tests for the fidelity of closed-loop control, it is necessary to specify the reference coordinate systems. For this very purpose, two main geographic reference coordinate systems, i.e. the body-fixed reference frame or local coordinate system and the Earth-fixed reference frame or global coordinate system, were considered. Moreover, GPS-based navigation used in the real-time validation water-trials for ASV control utilizes the geodetic coordinate system; they all discussed as follows.

6.3.1 Body-fixed Coordinate System

The body-fixed reference frame $\{b\}$ is a vehicle-carried moving coordinate frame, sometimes called the body coordinates comprised of three orthogonal axes x_b, y_b, and z_b having its origin

O_b, is usually chosen to be fixed at the center of the gravity (CG) of the ASV. The longitudinal axis x_b is directed from aft to fore, axis y_b is directed to starboard, and axis z_b is directed from CG to vertically downward of the ASV in order to comply with the right-hand rule. The body-fixed linear and angular velocities, i.e. $\boldsymbol{v}_{b/n}^{b} = [u,\ v,\ w]^T \in \mathbb{R}^3$ and $\boldsymbol{\omega}_{b/n}^{b} = [p,\ q,\ r]^T \in \mathbb{R}^3$, respectively, of the ASV are monitored by the vehicle's body-mounted sensors such as IMU and GPS as well as the dynamic constraints imposed by the ASV identified model are generally expressed in the body-fixed coordinate system. The graphical interpretation of the body-fixed coordinate system is given as in Fig. 6.2.

6.3.2 North-East-Down (NED) Coordinate System

NED is the geographic coordinate system $\{n\} = (X_n, Y_n, Z_n)$ having origin O_n arbitrarily fixed to a point on the Earth's geoid. This coordinate system, also known as flat Earth navigation coordinate system, is generally defined as tangent plane on the surface of the Earth moving with the ASV and considered to be the most common everyday-life local coordinate system. As its name suggests and according to the world geodetic system, 1984 (WGS-84), the x-axis X_n points towards the geodetic (true) north, the y-axis Y_n points towards the geodetic east while the z-axis Z_n completes the right-hand orthogonal coordinate system pointing downward along the normal to the Earth's reference ellipsoid. As the small autonomous surface vehicles navigate in a small region with quite low speed; therefore, being a small vessel, the navigation of the ASV used in this research is carried out within this frame of reference. To be precise, the vessel-carried NED as well as the local NED frames are not aligned exactly, but due to the nature of miniature ASV navigating in a small region with low speed, this directional difference is normally neglected entirely assuming them to be coinciding with each other. The NED coordinate frame is depicted in the Fig. 6.2 below. The pose $\boldsymbol{\eta}$ of ASV is described in this coordinate frame.

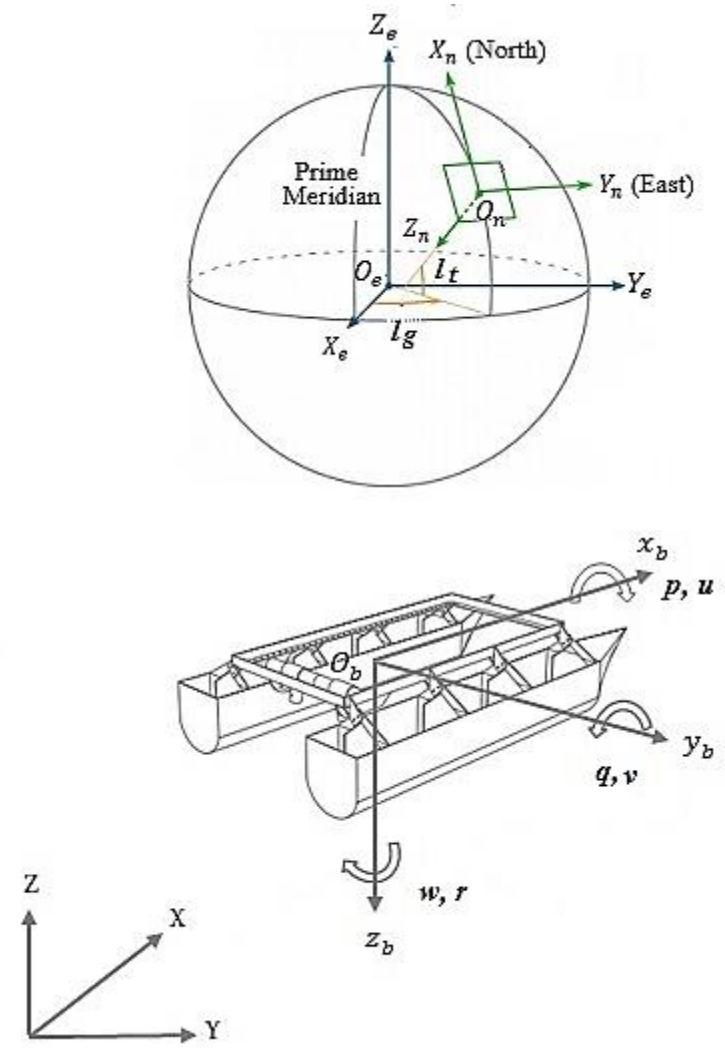

Figure 6.2 The coordinate systems of the ODU-ASV along with linear and angular velocities.

6.3.3 Geodetic Coordinate System

It is not a usual Cartesian coordinate system but a local geodetic coordinate system having origin coincide with the GPS frame. The GPS receiver utilizes the world geodetic system-1984 (WGS-84) as a global reference coordinate system originated by the U.S. Department of

Defense based on a reference ellipsoid model. This reference system is an Earth-centered, Earth-fixed (ECEF) terrestrial reference frame that consists of a reference ellipsoid, a standard coordinate system, altitude data, and a geoid. A coordinate point near the earth's surface is characterized in terms of longitude l_g, latitude l_t, and altitude h by this system. The longitude gives the position of measured point and the prime meridian in terms of rotational angle ranging from $-180°$ to $180°$. The latitude gives the angle ranging from $-90°$ to $90°$ between the normal of the reference ellipsoid that passes through the measured point and the equatorial plane. The altitude is local vertical distance the reference ellipsoid and the measured point. The position of ASV measured by GPS during the real-time validation water-trials for ASV control is transformed from WGS-84 domain to local Cartesian coordinates system i.e. NED frame using MATLAB® routines.

6.3.4 Homogenous Transformation

The position of ASV during the open-loop system identification water-trials was measured in 2-D Cartesian coordinates by LIDAR which was placed on the bank of dock. A local NED coordinate frame denoted as {A} is considered to be attached to LIDAR for the purpose of finding pose of ASV. In order to transform the local NED frame {A} to the arbitrary starting point of the ASV motion on the surface of water, and denoted as NED reference frame {B}, the 2-D homogenous transformation is performed on frame {A} as:

$$\begin{pmatrix} B_x \\ B_y \\ 1 \end{pmatrix} = \begin{pmatrix} R_A^B & t \\ 0_{1\times 2} & 1 \end{pmatrix} \begin{pmatrix} A_x \\ A_y \\ 1 \end{pmatrix}$$

where $t = (x, y)$ is the translation of the frame {A} to {B} and R_A^B is 2×2 orientation matrix of frame {A} with respect to frame {B} [52]. All other position data of ASV is related to the NED frame {B}.

6.4 Observer Kalman Filter Identification of ODU-ASV

The main purpose of OKID methodology is to best approximate the vehicle's dynamics with a less complex but more robust mathematical model so that the prediction error is reduced substantially. Once we have a sufficiently accurate mathematical model of the ASV, then the viable close-loop control is designed and later on implemented for its motion control. The identification of a useable model of the ASV dynamical system through the OKID algorithm requires that the open-loop input and output data measured from the motion of ASV must contain significant information regarding the persistently excited modes of system [37], [45]. This entails the OKID method starting from the point of exciting the relevant dynamic modes of ASV's system by some persistent discrete inputs $u(k)$ and measuring the resulting output responses $y(k)$ from all modes of interest by the required sensors. The detail step-wise process of OKID is given as follows.

6.4.1 Input and Output Data for OKID

The type of input $u(k)$ to the ASV system is selected in such a manner that it is persistent and can excite all relevant modes of the system. In order to get informative data from the ASV system, the thrusters are actuated by two outboard DC trolling motors installed on the stern of the ASV, which are in turn excited by the PWM voltage signals. The PWM is basically a modulation technique in which a digital signal is supplied in pulses of variable width and positions but maintaining the same frequency content; hence, this technique entails that the PWM signal is actually a kind of digital square wave consisting of two primary components: a duty cycle and a frequency. The variable speed control of the motors is attained with the help of the PWM technique applied to motor voltage, and the same process can be accomplished by the Arduino microcontroller and Victor SP speed controllers which have been utilized in this research. The

PWM values range from 40 to 160 duty cycles, with 90 being the neutral value at which each trolling motor stops rotating. The range of PWM values above the neutral value, i.e. 91-160 duty cycles, is used for forward motion, while the PWM values below neutral value, i.e. 40-89 duty cycles, are used for the reverse motion of ASV.

The ASV has no rudder mechanism, so the turning moment is induced by the differential thrust of the two astern outboard motors that are run by the PWM signals sent from the main onboard computer via Arduino microcontroller board. The turning moment in turn causes to change the yaw angle ψ of the ASV with respect to the z-axis which continuously varies with time and is then known as yaw rate r. So, in the present work, the motion of ASV is considered to be in horizontal plane because the water-trials of ASV having small size were performed in rather calm water as compare to the open sea. If only a few degrees of freedom at a time are considered for ASV dynamics then this assumption decouples the motion of the ASV and hence reduces the number of controllable degrees of freedom involved in control system design. As the thrusters are positioned in the ASV such that its vertical dynamics such as heave w, pitch angle θ, and roll angle ϕ are not dominant and hence not considered in the system identification and control of the ASV without considerable loss in accuracy during typical slow maneuvers [7], [16]. Therefore the ASV system, being six DOF, is under-actuated. However, the un-actuated roll and pitch motions are stable and can be left uncontrolled in the succeeding control design owing to twin-hull design and the slow motion of the ASV over the surface of water.

Rather than being truly stochastic, the motion of the ASV is better described in deterministic terms like: initial steady acceleration for a short period of time and then coasting of ASV with considerably low constant velocity which is calculated to be $1\ m/s$, so the dynamics of ASV is considered as velocity invariant [16]. According to general kinematic equations of ASV

motion in [4], position and velocity of the ASV in two-dimensional space are coupled, and also the states x, y, and yaw angle are seemed to be coupled [4]. From the Figure 6.2, surge is defined as the velocity vector u of the CG of ASV along the x-axis, and yaw angle defined as the angle of xz-plane with respect to z-axis of the ASV. So, the position of the ASV in x and y coordinates, yaw, and yaw rate are assumed to be instrumented which constitute the state vector as $states = [\, x \;\; y \;\; \psi \;\; \dot{\psi} \,]$.

6.4.2 Data Acquisition and Preprocessing

In order to achieve the rich set of input and output data for open-loop system identification tasks, a great deal of effort was made to carry out series of outdoor path-following water-trials in the Lafayette River in Norfolk, Virginia. It is actually a tidal estuary that empties into the Elizabeth River which in turn discharges into the Chesapeake Bay. The ASV used in this research is a miniature-design, so the water test location is selected in a way that it would provide moderate environmental disturbances by wind, current, and tidal wave. When the outdoor water-trials were being performed, there was partial sunny with slight overcast. The weather was rather calm having wind speed of 6.9 mph westbound and the temperature was 71° F. The tidal condition was average with a coefficient of 54; hence, there was not much disturbance to be brought forth by the tidal waves and the wind. The ODU-ASV is shown in Fig. 6.3 while cruising for the purpose of open-loop data acquisition for OKID algorithm.

There are many well-known paths that researchers usually propose for the ASV to follow for their autonomous/unmanned tasks but sinusoidal, zigzag, and arc-like are the most illustrious ones, and the same have been used in this research; in order to have a full range of ASV dynamics and validity of the resulting identified dynamic model for its control design could be assessed. The ASV was bounded to follow these paths within a 16 m range, which is the limit of

LIDAR for accurate measurements. During water trials, a human driver sent the control commands to the ASV in the form of PWM voltage values through R/C transmitter in manual mode, and thrusts were produced by the two outboard motors which were stabilized for a long enough time to ensure that the given thrusts impart an impact in the change of position and heading of the ASV.

Figure 6.3 ODU-ASV while cruising for the purpose of data acquisition for OKID identification.

Consequently, the raw input and output discrete data from the installed sensors was collected and saved as some large data files in the onboard computer memory for the OKID method to be applied later on. In the present work, fifteen water trials of sinusoidal, zigzag, and arc-like maneuvering paths were selected from various experiments performed at different occasions. For in-

stance, the raw data size of sinusoidal maneuvers was about 6000 sample points in total.

Prior to the OKID passes, the offline raw input and output data from the required actuators and sensors is refined and adjusted after visual inspection for missing data, outliers, and drift; for instance, it has been observed that LIDAR sometimes rendered position data of ASV disruptively, which were adjusted by interpolation technique. Then, filtered utilizing a low pass filter (LPF) so that high frequency disturbances caused by noise in the data are surfaced and discarded. The selection of the suitable and equal sampling frequency of each input and output signals is also assured to be fulfilled for the critical requirement of OKID algorithm. If the sampling rate of the input/output signals is too small as compared to the highest frequency content of the system, dynamics of the system is not actually identified well enough due to the resulting badly conditioned system equations. On the other hand, too high sample rate of the input/output signals

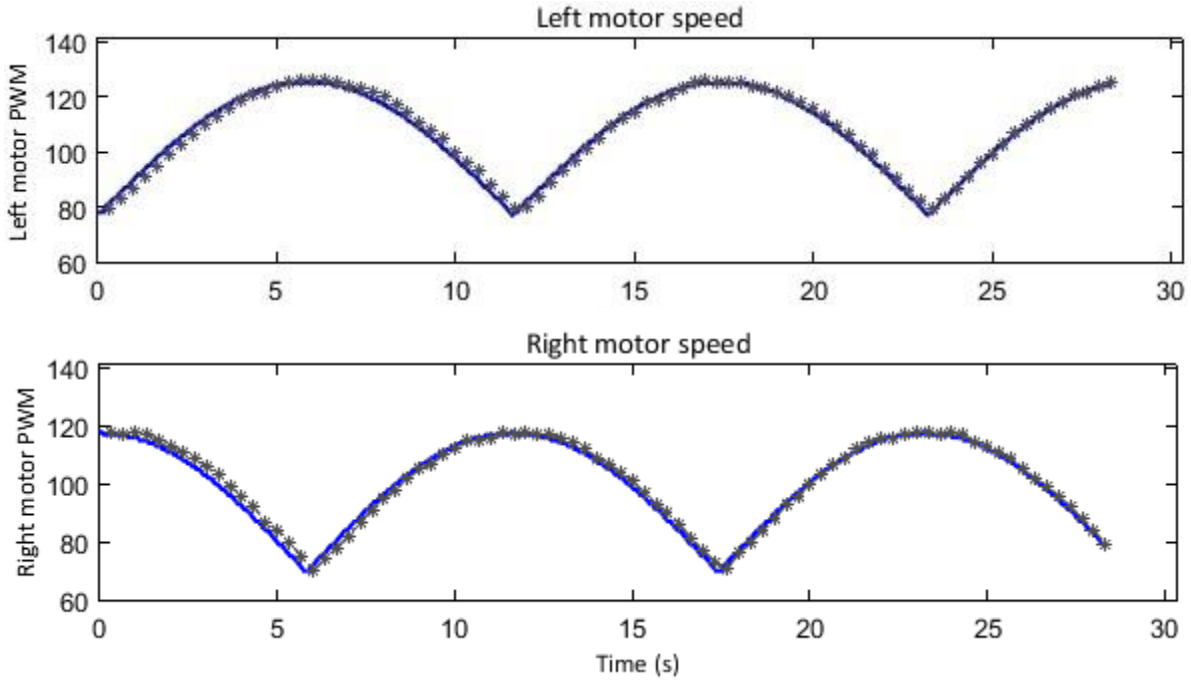

Figure 6.4 Raw (—) and their corresponding filtered input data (-*-) from an identification run.

results in bad numerical of the matrix $A^T A$ owing to almost linearly dependent rows; where A is the system dynamic matrix. Very high sample rate also bunching up the poles of a discrete-time system around $z = 1$ and cause more noise in the data [53]. The Raw input and output data with their corresponding filtered data from an illustrative identification run are shown in Figs. 6.4 and 6.5 respectively.

6.4.3 Illustrative Open-loop OKID Run and Results

As discussed in the preceding section, carrying out outdoor open-loop water tests of a dynamical system such as the ASV for the purpose of OKID algorithm to model the entire range of its required responses is a time consuming process; therefore, some engineering skill is required in the design of the experiments which are as informative as possible to minimize the unavoidable disturbances and hence to ensure the best mathematical model of the ASV which is workable for designing the optimal feedback control of ASV. The filtered experimental input and output data having same samples is retrieved from the computer memory and used by the OKID algorithm for identification of the ASV model.

An important and initial step necessary for starting the OKID method is to select an upper bound on the potential system order in such a way that the computation time would be reduced to a great extent. Hence, this step specifies blocks of the input and output data to be used. So a number p according to Eq. (4.27), representing number of the observer Markov parameters and also the time step for the observer to become deadbeat, was selected such that $m \times p > n$, where m and n are the number of outputs and the order of the system respectively. So, $m \times p$ represents the upper bound on the order of the identified system model. Apparently, the parameter p could be smaller than the true order of the system for multiple output system; so number p was chosen to be 20 in our work, when multiplied by $m = 4$ equals to 80 which would sufficiently

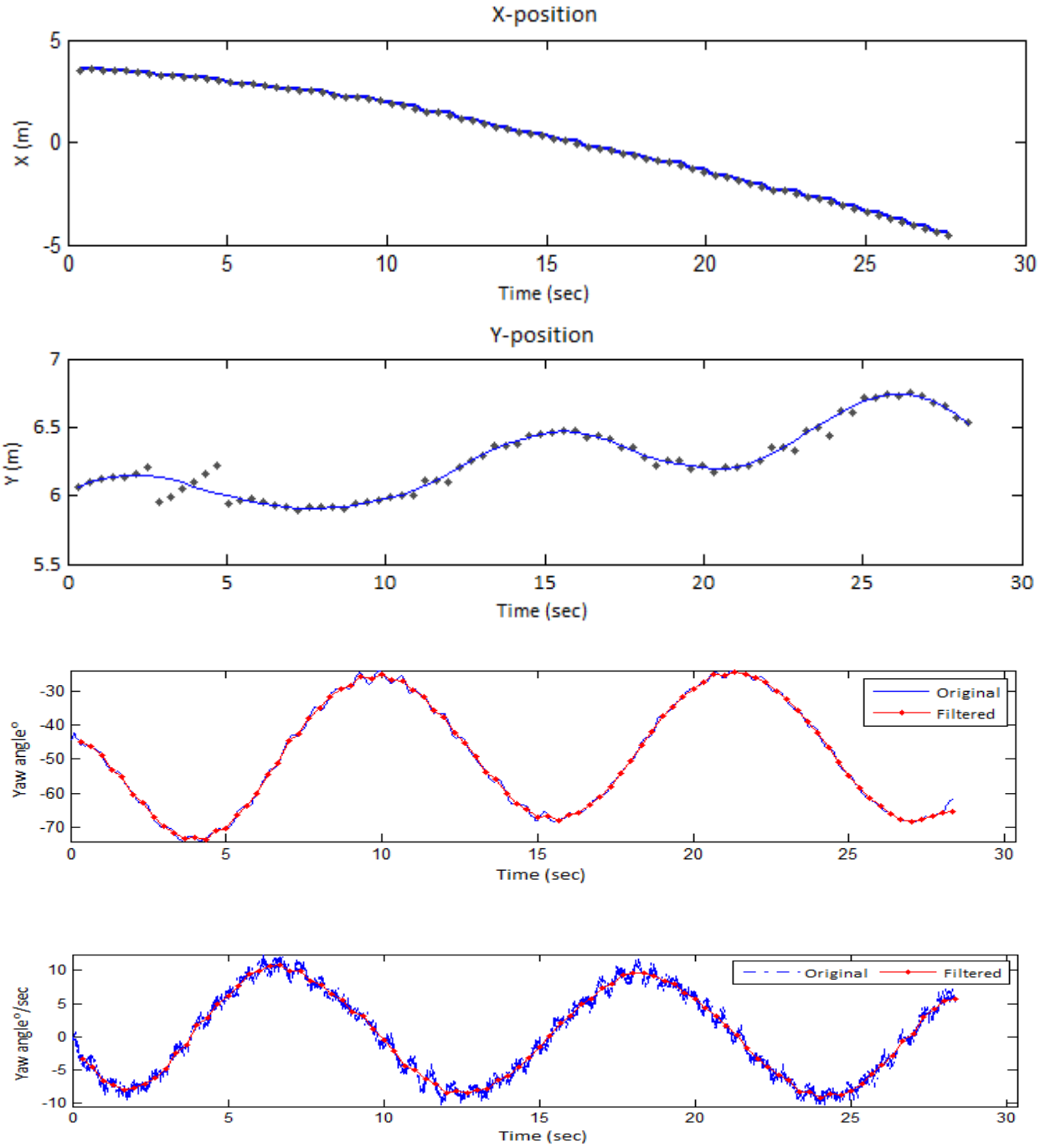

Figure 6.5 Output data from an illustrative identification run showing raw data (dotted line) and their corresponding filtered data (solid line).

larger than the maximum true system order for the ASV; hence, the guessed value for the number p is more than enough. The plot of Hankel matrix singular values (HSV) obtained through SVD and the modal singular values (MSV) from the OKID algorithm applied to the ASV dynamical system can be seen in Fig. 6.6 and 6.7, respectively, which can be used to select the correct system order. The number of non-zero significant singular values represents the system order [38] [48]. The system order, in essence, delineates overall system response with respect to the applied inputs.

Once we have HSV and MSV after SVD, then the balanced truncation (BT) method, which is one of the common reduction methods for selecting a model having suitable order of a linear time-invariant system, is used. In this technique, important modes of the ASV dynamical system represented by relatively much higher Hankel singular values arranged in descending order such as $\sigma_1 > \sigma_2 > \cdots > \sigma_n > 0$ of the diagonal matrix $\sum$ that were obtained through the SVD process. The significant singular values that evaluate the state contribution are kept while the insignificant ones that are relatively much smaller modes of the system are discarded which reflect the measurement noise affecting the system [38] [54]. The balanced truncation method can be performed either automatically in the code or interactively by the user. In this research, we follow the interactive approach of model order determination of the balanced truncation method in which the decision is made by the user as part of the offline identification process.

Due to the noisy input and output data of a real system, we can differentiate insightfully between the real and noise modes. Therefore, by examining the Hankel singular values shown in Fig. 6.6, a significant plunge in the singular values after the fourth one can be viewed clearly which made us to select the ASV system order to be four, whereupon majority of ASV system's output response was described by that order accurately. The identification of system having order

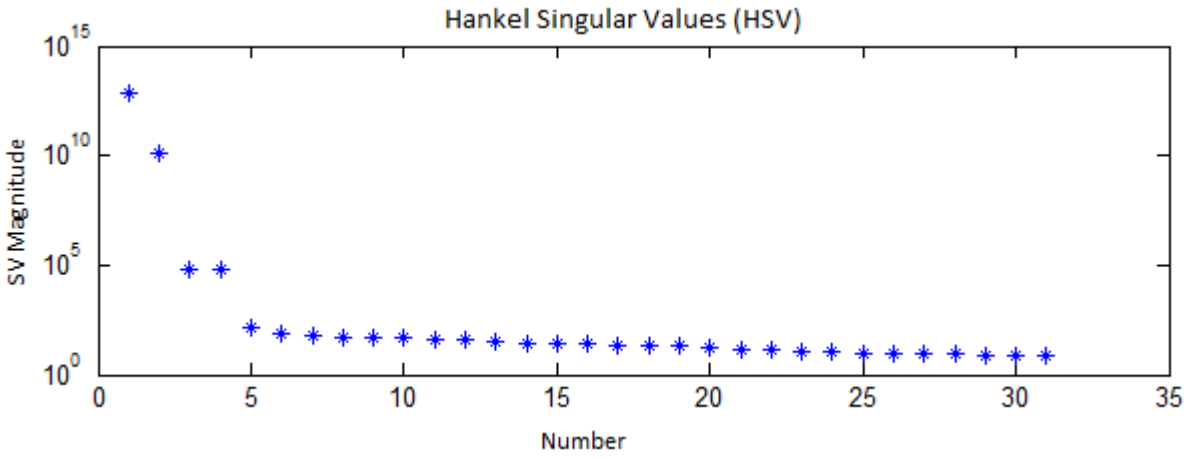

Figure 6.6 Hankel singular values from an illustrative identification run of ODU-ASV.

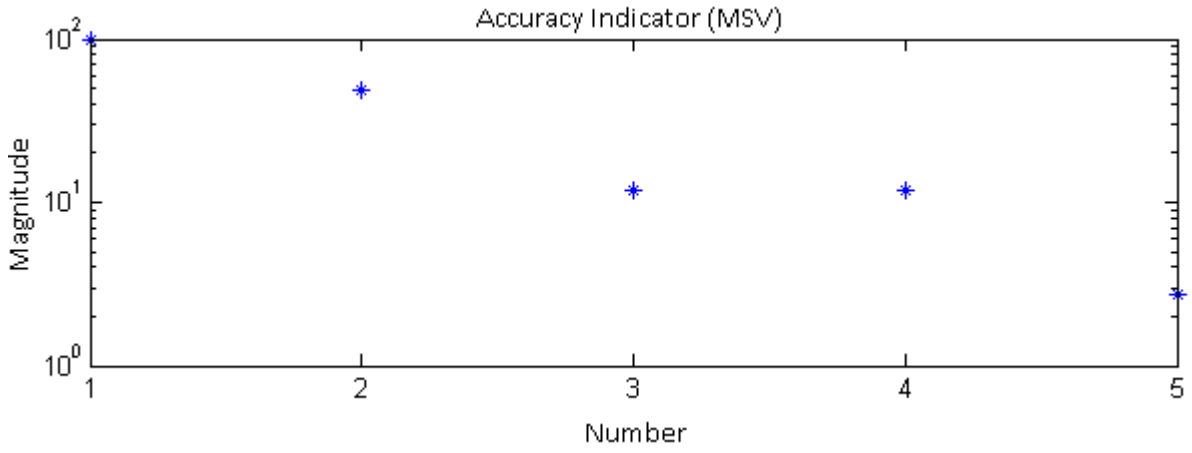

Figure 6.7 Modal singular values from an illustrative identification run of ODU-ASV.

fourth is also confirmed by the accuracy indicator i.e. modal singular values (MSV) as depicted in Fig. 6.7. The detail of accuracy indicator was given at the chapter 3; so the hypothetical reason for the fourth order model is that the fourth mode is considered to be a delay between the actuation of trolling motors rendering thrust and the ASV starts to rotate about the CG point. The delay would add another state and the fourth order is the acceptable choice for the ASV system. The truncated Hankel matrix is then used to calculate the system Markov parameters. Then, from utilizing system Markov parameters in the ERA/DC algorithm [38], balanced realization in the form of system state matrices i.e. $A, B, C, \& D$ of the ODU-ASV is obtained as:

$$\left.\begin{aligned}
A &= \begin{bmatrix} 1.0104 & 0 & 0 & 0 \\ 0 & -0.0303 & 0 & 0 \\ 0 & 0 & 0.9791 & 0.0121 \\ 0 & 0 & -0.0121 & 0.9791 \end{bmatrix}, \\
B &= \begin{bmatrix} -0.85000 & -0.85930 \\ -14.8151 & -23.9785 \\ 1.79070 & 1.91480 \\ -1.62140 & -1.67990 \end{bmatrix}, \\
C &= \begin{bmatrix} 0.1672 & -0.0429 & 0.1722 & 0.1150 \\ -0.1897 & 0.9573 & 0.3086 & 0.0000 \\ -0.0022 & 0.0328 & -0.0027 & 0.0015 \\ -0.000032124 & -0.0047 & 0.000032116 & -0.00023622 \end{bmatrix}, \\
D &= \begin{bmatrix} -0.0923 & -0.0907 \\ 0.1821 & 0.1790 \\ -0.0011 & -0.0011 \\ 0.00019659 & 0.00019323 \end{bmatrix}
\end{aligned}\right\} \quad (6.2)$$

As there are infinite realizations for a system that depict the same input/output behavior, but the realization that has least number of state variables required to describe the specific input/output behavior is called the minimal realization. The minimal realization is also the balanced realization which is achieved when the system is both completely state controllable and state observa-

ble. The controllability and observability of discrete-time ASV system is determined by computing the ranks of controllability and observability matrices given in equations (3.16) and (3.17), respectively; which are four in both cases and equal to the order of the ASV system.

6.5 Benchmark Assessment Tests of OKID-identified ASV Model

In order to assure the validity and performance evaluation of fourth order model identified by OKID algorithm from the experimental input and output data alone, the estimation and prediction results derived from observer and identified model respectively, should be verified. Afterwards, to verify the identification results for having the best identified model, its validation is ascertained by testing the residuals for whiteness, which is the process of calculating the autocorrelation function of inputs. Moreover, to check whether the identified model left behind any undetermined effects of inputs by computing the correlation between output residuals with lagged inputs [36], [55], [46].

6.5.1 Reconstruction of Output Data during Simulation

That OKID is the best suited method for the identification of MIMO systems, besides the SISO systems [45], [16]; a fourth order model was identified for the ASV forward motion by the OKID algorithm as elaborated in previous section. The identified model needs to be investigated for its viability; for this purpose, the identified ASV system model as well as its related identified observer was simulated in time to the known inputs. The predicted as well as estimated outputs in the form of $x\ y$ Cartesian coordinates, yaw angle ψ, and yaw rate $\dot{\psi}$ are achieved by simulating in time the identified model and its corresponding observer of the ASV system with the help of known inputs and are given in Fig. 6.8 and 6.9, respectively. This means the predicted outputs are the reconstructed output data from the identified system model only, while the estimated outputs are gained from the identified observer equation.

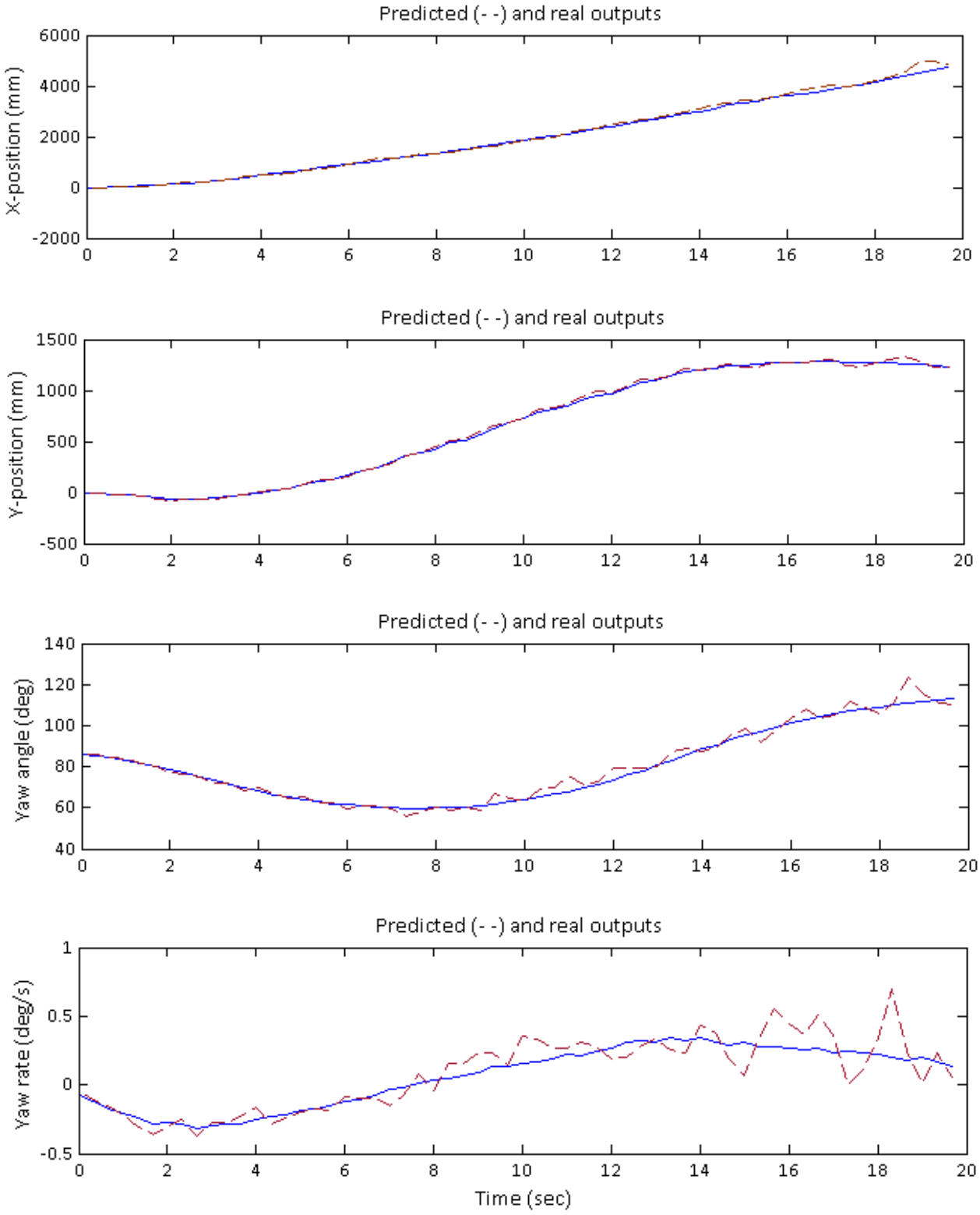

Figure 6.8 Observed and simulated outputs of reconstructed data of the identified forward model.

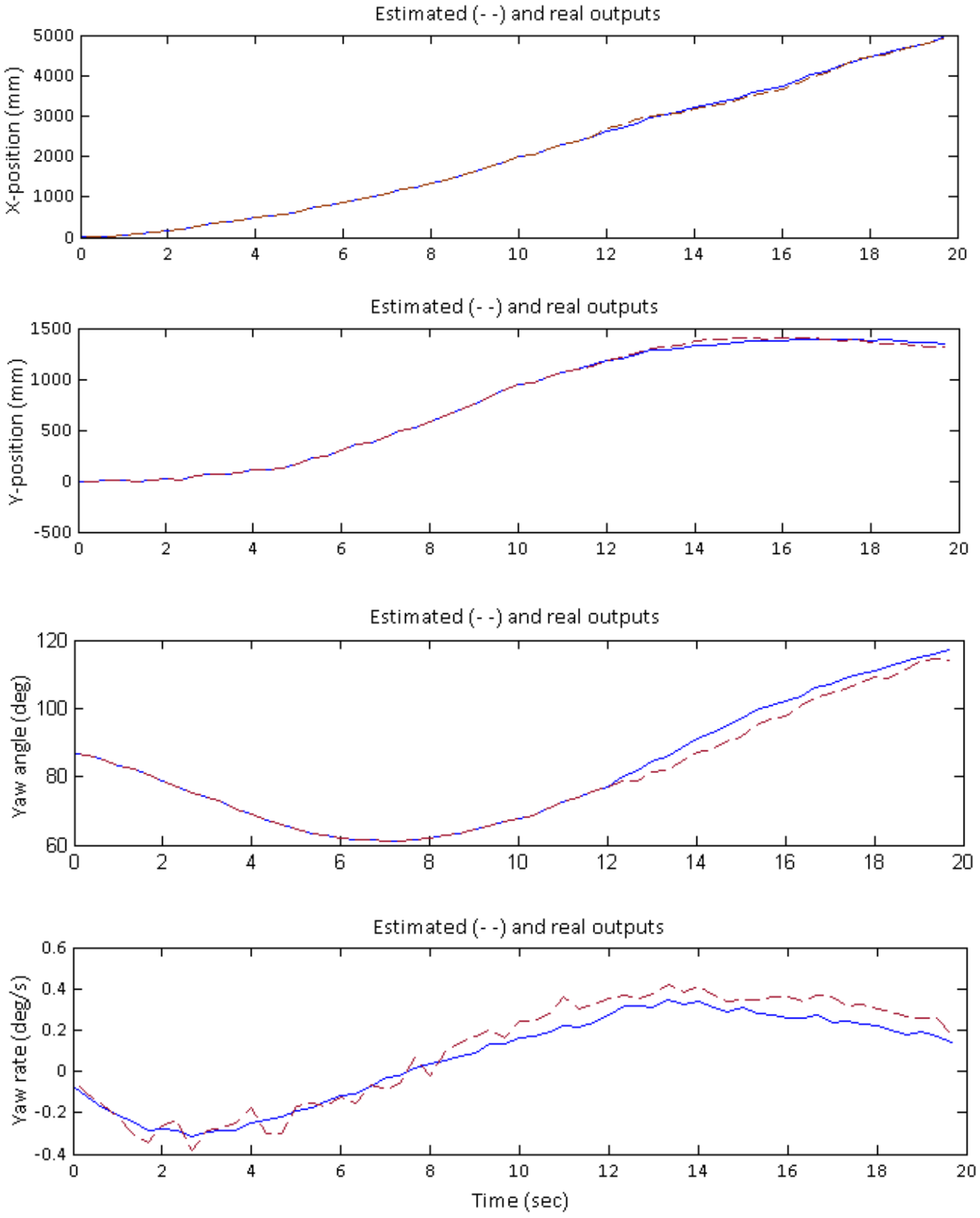

Figure 6.9 Observed and simulated outputs of reconstructed data of the identified observer.

The dashed lines in Figs. 6.8 and 6.9 shows the reconstructed data from simulation of the identified forward model and the observer, respectively, while the solid lines represent observed experimental output data. By examining these figures, it is evident that the prediction as well as estimation of Cartesian X, Y position coordinates, and yaw angle are in perfect agreement with their respective actual observed outputs of the ASV system. As a matter of fact, perfect tracking of the actual outputs is not required because the sensors might have high frequency noise which can be observable but definitely not controllable. Among the outputs, the Cartesian position data is critically important as they are used in waypoint trajectory following control design of the ASV. As far as the yaw rate is concerned, the reconstructed data from the identified forward observer matches the actual output data quite better than that from the identified system model. Therefore, overall performance of the identified ASV's model is good enough in predicting the ODU-ASV response to the known inputs. Simulation of the model performance can't completely clarify the deficiencies in the proposed state-space model. Therefore, model predictions error strategy can be used as residual analysis in the model validation step [36], as discussed ahead.

6.5.2 The Residual Analysis

As discussed in preceding section, model validation is also accomplished by comparing one-step ahead predicted response of the identified model with measured response of validation data set which is produced by the actual system and then evaluating their difference. The difference is technically known as residual or prediction error related to the dynamic model. The residual is the cardinal quantity of interest in the assessment tests of identified model and if not left behind by the said model then that is called the perfect model. This fact implies that ideally the residual should not depend on the inputs or its past residual values. If that requirement is not fulfilled, then it indicates that some of the validation data set is not described perfectly by the iden-

tified model but left over as residual or the prediction error; consequently, the model is inevitably to be improved [36], [55]. The residual $\epsilon(k)$ is defined in a formal manner as:

$$\epsilon(k) = y(k) - \hat{y}(k) \tag{6.2}$$

where $\hat{y}(k)$ is the predicted output, given by the identified system model, of the corresponding observed output $y(k)$ at time step k. Residual analysis is mainly performed with the help of two tests: the whiteness test which determines the predictability in residuals and the independence test which refers to correlating residuals with inputs; the details are presented as follows.

6.5.2.1 The Whiteness Test: Predictability in Residuals

According to the whiteness test of output residuals, the autocorrelation function (ACF) of residuals is used for finding how the prediction errors are related in time. Autocorrelation relates to the correlation of a time series data with its own past and/or future values. In order to have perfect prediction for an identified model, it is necessary to have only one nonzero value of the autocorrelation function occurring at zero lag. If the samples of output residuals separated in time by lag *l* have no statistically significant correlation among them, then there is no scope for predictability within the output residuals. This indicates that the residuals or prediction errors contain no obvious patterns, which means they are completely uncorrelated with each other, and hence, is a white noise sequence [36] [45] [55]. The ACF of each output residuals rendered by the OKID identified model of our ASV are shown in the Fig. 6.10. The figures show that the output correlations except for the one at zero lag fall approximately within 99% confidence interval around zero. Therefore, the identified model appears to be adequate and it is apparent from all the auto-correlation of output residuals that the model significantly the stochastic noise characteristics of the measured output.

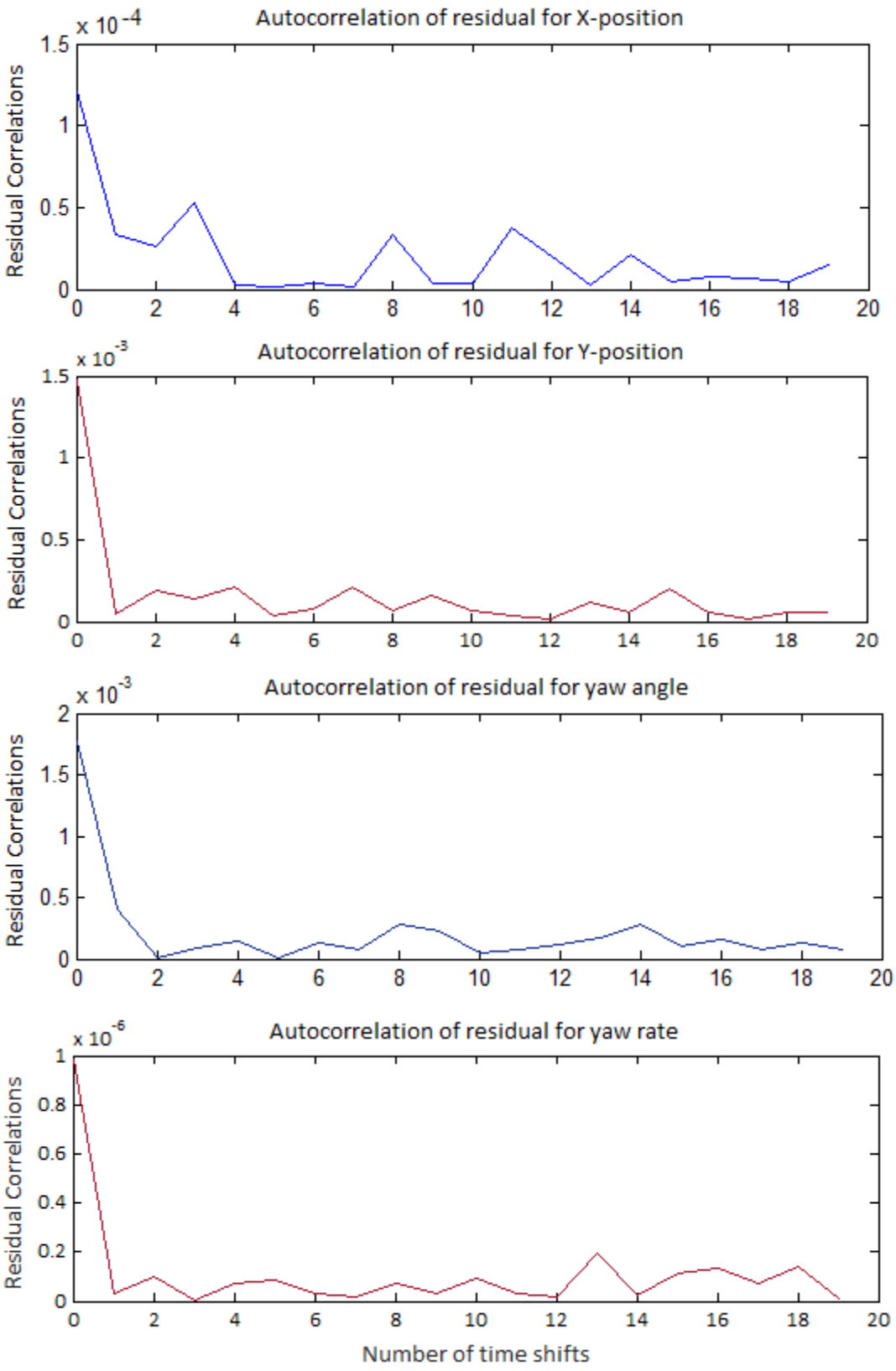

Figure 6.10 Auto-correlation of output residuals obtained from OKID identified model of ASV.

6.5.2.2 The Independent Test: Correlating Residuals with Inputs

Another test for the identified model to behave as a good deterministic model is to find out the independence of the model's residuals with respect to the past inputs of the system. This implies that a good identified model picks up the dynamics of the system adequately and never accumulates any unexplained leftovers originated by the inputs. For this purpose, the cross-correlation function of the residuals with the lagged (time-shifted) inputs of the system is computed and it should lie within the confidence region for this function. Mathematically the sample cross-correlation function $r_{u\in}(l)$ in discrete time domain is:

$$r_{u\in}(l) \;=\; lim_{N\to\infty}\frac{1}{2N+1}\sum_{i=-N}^{N} u(i+l) \;\in (i) \tag{6.3}$$

The cross-correlation function can also be calculated for the negative lag and are sometimes of interest, because the output does not depend on future inputs in causal systems. Therefore, the values of cross-correlation function for positive lags are of great importance and will be used. A significant correlation between $\in (l)$ and input $u(l)$ at positive lags directly implies that the effects of inputs on the process response have not been completely explained [36] [46]. The cross-correlation between inputs and the residuals for each input-output pair are given in Figs. 6.11 and 6.12. The correlation between the residual of each output of the identified model with each left and right motor time-lagged PWM inputs lie within the 95% confidence interval and do perform well.

6.5.3 Cross Validation

The technique of validating a model using an independent data set, which is not used in model identification process, is termed as *cross-validation.* Hence the best way of having insight into the examination of the identified model's quality is to simulate it with input from a new

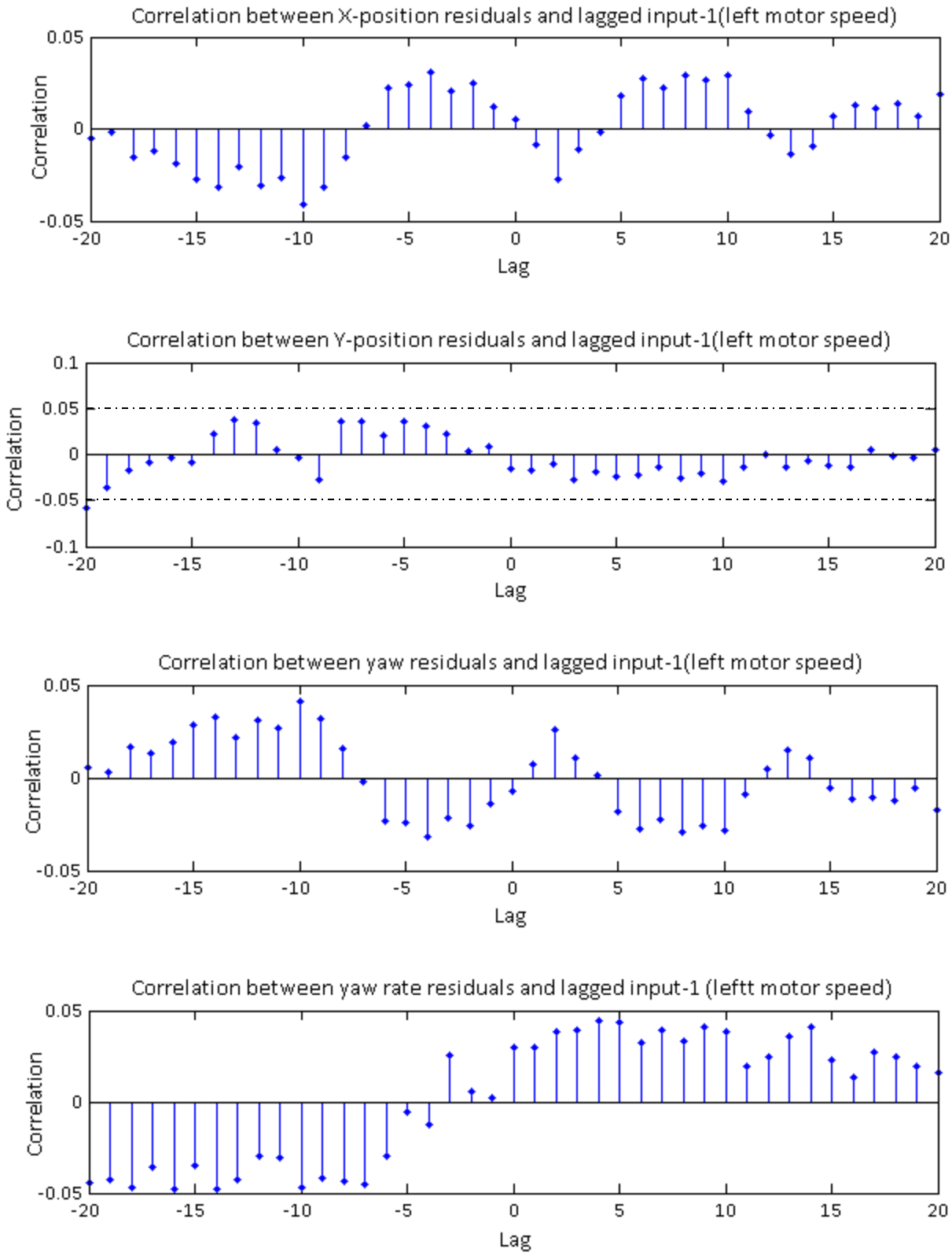

Figure 6.11 Cross-correlation between lagged values of input-1 and residuals for all output data.

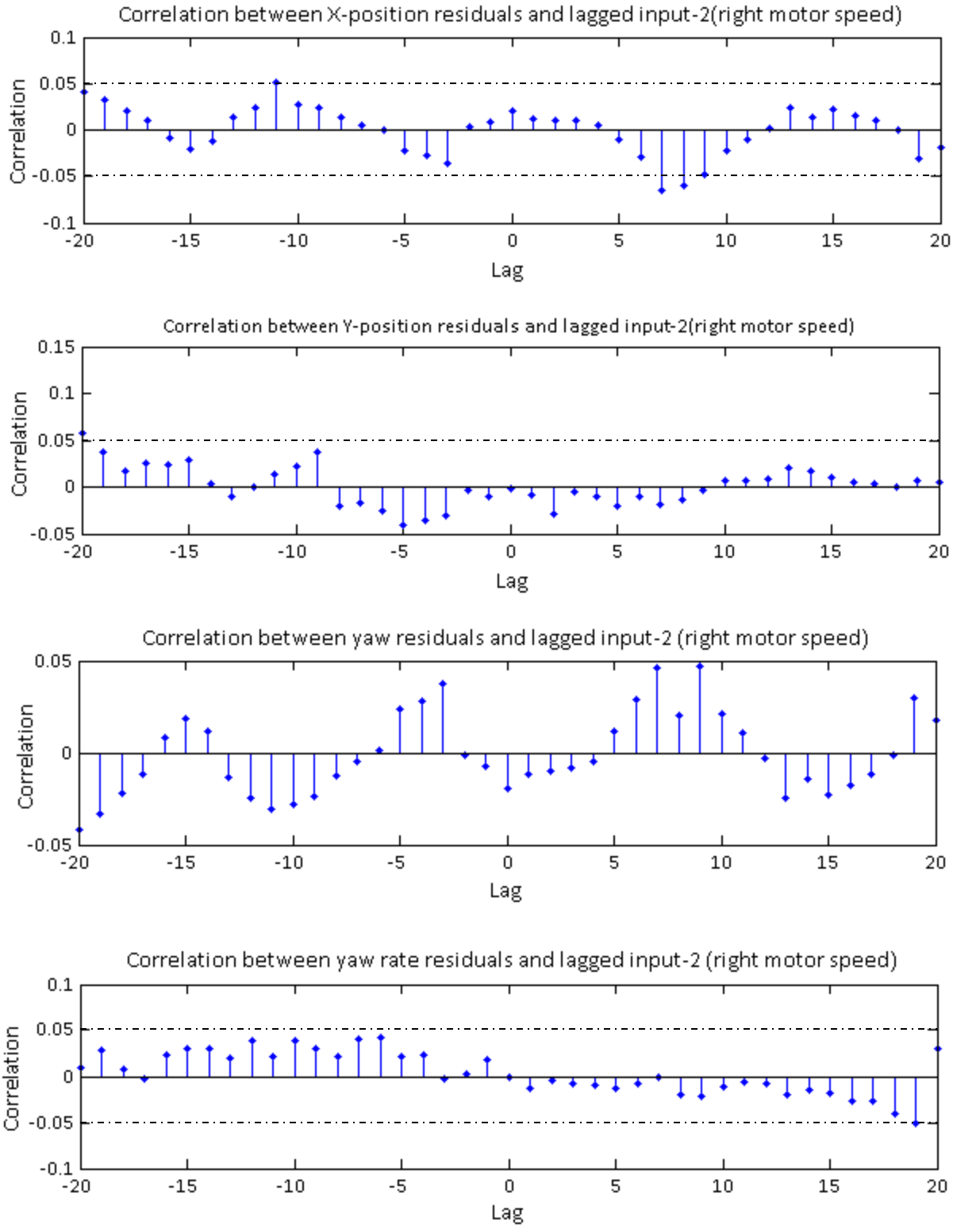

Figure 6.12 Cross-correlation between lagged values of input-2 and residuals for all output data.

validation data set that is not used in the identification process and then compare the simulated output thus obtained from the model with the measured one. A good model of the ASV system is the one which renders good predictions on the fresh data of validation set [37], [48], [56]. Therefore, the simulated output response of identified model of ASV dynamics is cross validated against the output data from the ASV measured during the water trials. A couple of the measured test data set was set aside for this purpose and was not used in system identification task. A goodness of fit is determined between the output data from the ASV dynamical system measured by the sensors during the real water tests and the simulated model output of ASV. The common measure used to determine the percentage of goodness of fit denoted by R_f is the normalized root mean square error (NRMSE) as given:

$$R_f = 1 - \frac{\|\boldsymbol{y} - \hat{\boldsymbol{y}}\|_2}{\|\boldsymbol{y} - mean[\boldsymbol{y}]\|_2} \times 100\ \% \tag{6.4}$$

where $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$ are vectors of measured and predicted outputs respectively; and the notation $\|.\|_2$ stands for the 2-norm of a vector. The fit explains that how much in percentage the output of ASV system behavior is explained by the simulated output of its identified model. The comparison between simulated responses of OKID identified dynamical model of ASV against true observed output data which is reserved for validation of X, Y position coordinates, and yaw angle is shown in the Fig. 6.13. The percentage goodness of fitness (NRMSE) values for predictions of X & Y position coordinates and yaw angle of the ASV are given in Table 6.1.

By examining the plots given in Fig. 6.13 and the NRMSE values given in Table 6.1, it is clearly revealed that simulated output response in case of Cartesian X, Y position data of ASV obtained through OKID identified model closely matches their respective experimental observed data, while yaw angle ψ is not a very good match. As we are mainly concerned with the X and Y

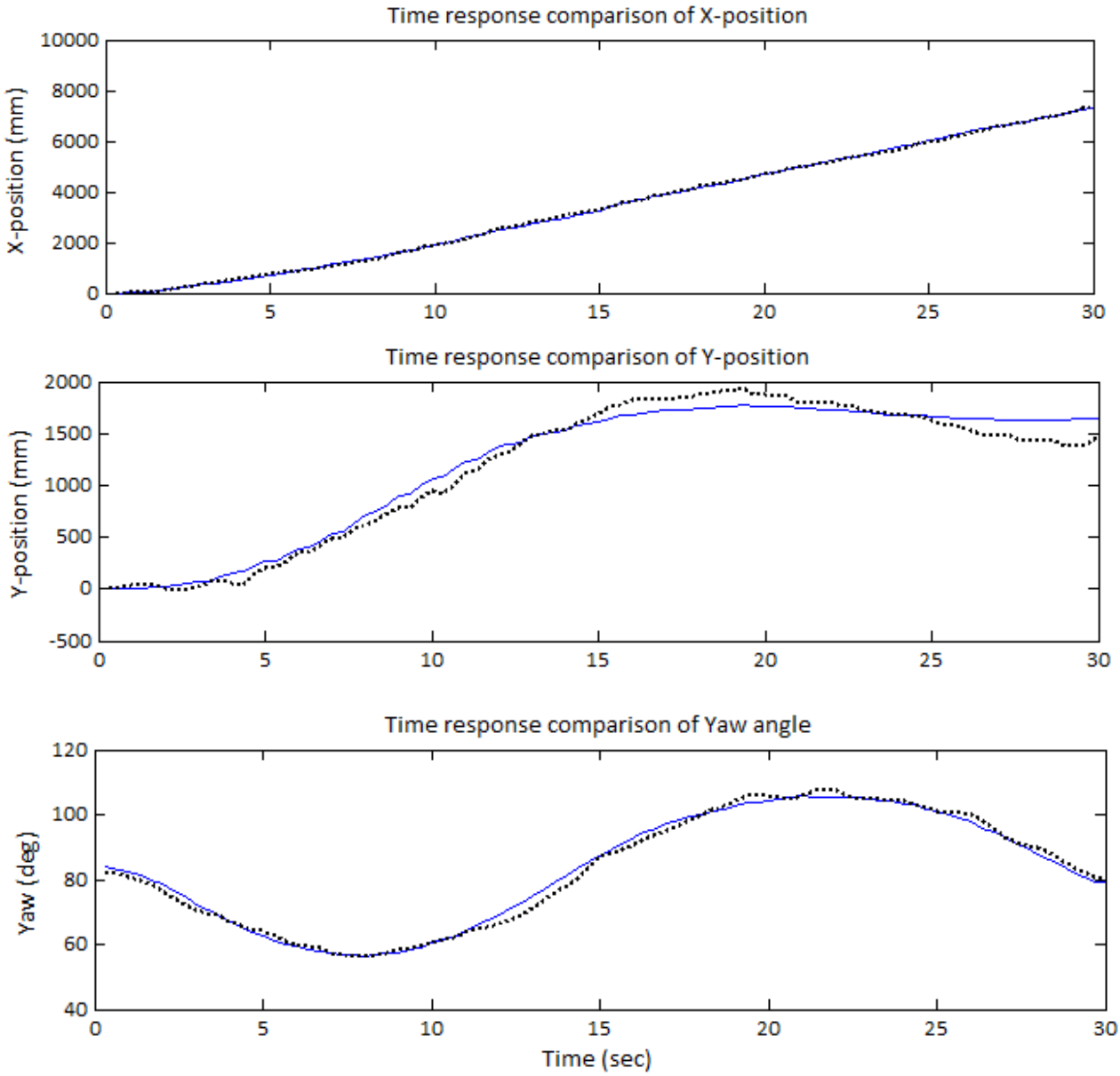

Figure 6.13 True measured (solid line) and model predicted (dotted line) outputs of ASV system.

position coordinates because the position data is critical for path-following control strategy. In this regard, yaw angle ψ and yaw rate $\dot{\psi}$ are not the required output states to be controlled, so we are not paying much attention to them. As a matter of fact, the yaw dynamics, i.e. yaw angle ψ

Table 6.1 Percentage goodness of fit of X, Y, & yaw.

	Percentage goodness of fitness (NRMSE)
Position in X-coordinate	97.99 %
Position in Y-coordinate	83.72 %
Yaw angle	60.79 %

and yaw rate $\dot{\psi}$ are coupled with the surge and sway motion [4] so they are necessary in the system identification process of ASV. Hence, it is concluded that the identified model of order 4 validate the position data in Cartesian coordinates very well.

6.6 Discussion Regarding OKID-identified Model

In the previous sections of the chapter, the time-domain observer Kalman identification (OKID) state-space algorithm was utilized to find the system matrices of the ODU-ASV so that they could be used in the optimal control design for its motion control. This is basically a discrete linear time-invariant mapping technique for characterizing the inherent nonlinear relationship between experimental input output data without imposing user-defined a-priori assumptions regarding model order or model structure [38]. In the OKID identification procedure, the observer Markov parameters were obtained first from the real-time experimental input and output data collected from the ASV open-loop water trials and from them the system Markov parameters, and the observer gain Markov parameters were calculated. Then, the state space model of the ASV was realized from the system Markov parameters using ERA/DC algorithm, which starts with the formation of generalized block data matrix known as Hankel matrix and later on to fac-

torize it by singular value decomposition. The true system order of ASV is identified to be four by selecting the first significant non-zero singular values from HSV and MSV plots. The estimate of state-space system matrices of ASV such as $\hat{A}$, $\hat{B}$, and $\hat{C}$ are obtained using Eq. 3.32.

Numerical simulations are presented, and it can be concluded from the simulation of output data reconstruction that the prediction as well as estimation regarding position of ASV in Cartesian X, Y coordinates, and yaw angle is in perfect agreement with their respective actual observed outputs of the ASV system. As far as the yaw rate is concerned, the reconstructed data from the identified forward observer matches the actual output data quite better than that from the identified system model. Therefore, overall performance of the identified ASV's model is good enough in predicting the ODU-ASV response to the known inputs.

The accuracy of the identified model for ASV dynamical has also been quantified with the benchmark residual analysis; and the results of these tests shows that the ACF of each output residual fall approximately within 99% confidence interval around zero except for the one at zero lag. Moreover, the correlation between the residual of each output of the identified model with each left and right motor time-lagged PWM inputs lie within the 95% confidence interval and do perform well. From cross validation test of the said model shows that simulated output response in case of Cartesian X, Y position data of ASV obtained through the OKID identified model closely matches their respective experimental observed data; in this case, the percentage goodness of fitness (NRMSE) values for predictions of X & Y position coordinates and yaw angle of the ASV are 97.99 %, 83.72 %, and 60.79 %, respectively. So, the identified ASV model can adequately be used for the design of closed-loop state feedback optimal linear quadratic tracking controller for its path-following control strategy.

6.7 Optimal Linear Quadratic Tracking Control

Oftentimes in controlling the position of ASV, regulating the state to or near zero is not the point of intrigue but rather tracking of nonzero reference waypoints or the command signal is the sole purpose of control design. The position of ASV in X and Y coordinates while coasting is susceptible to environmental disturbances such as wind and currents and could sufficiently be off the target waypoints during the path-following control strategy under the action of simply PID control. This behavior consequently can show a delayed response by the ASV in coasting from its current position to the target waypoint. This problem could be solved by using optimal discrete feedback linear quadratic tracking (LQT) control which is the extension of LQR control design approach. Hence, in this kind of optimal tracking or servo-design problem, an additional feedforward term is added to the control input along with the fundamental LQR feedback loop which makes the controller robust to uncertainties and disturbances as well as closed-loop stability of the system is ensured. As we have state-space realization for ASV obtained in section 6.4, the optimal discrete feedback LQT control was designed and implemented in this section. For this purpose, path-following motion control strategy was adopted for the specified paths using X and Y Cartesian coordinates as reference waypoints in the present work, wherein all the states from necessary sensors and the inputs from actuators were available to be used. The design of optimal discrete LQT control for ASV motion is outlined as below.

6.7.1 The Design Outline of LQT Control

From the OKID methodology, the finite-dimensional discrete linear time-invariant state space model, as given in Eqs. 4.12 - 4.13, was identified in section 6.4 above, that characterizes the dynamics of ASV wherein the identified system matrices $A, B, C, \& D$ are given in Eq. 6.2. The n-dimensional, where $n = 4$, state vector that is considered as: $x = [x \ y \ \psi \ \dot{\psi}]^T$ represent-

ing position of ASV in X & Y Cartesian coordinates, yaw angle, and yaw rate respectively. These states are observed by the pose sensors at sampled time. The r-dimensional, where $r = 2$, control input vector comprise of the applied PWM voltage values, i.e. $u(k) = [u_1(k)\, u_2(k)]^T$, that causes to produces thrusts by port and starboard astern motors, thereby change the heading or yaw angle ψ; and consequently the position of ASV varies too. Before designing the proposed controller, it's necessary to determine the open-loop poles of the ASV, system which are the eigenvalues of matrix A as shown:

$$eig(A) = \begin{bmatrix} 0.9791 + 0.0121\, i \\ 0.9791 - 0.0121\, i \\ 1.0104 \\ -0.0303 \end{bmatrix} \tag{6.5}$$

It is evident that three of the open-loop poles of ASV system are on the right-half plane of the pole-zero map, so it indicates that the system is unstable in open-loop. Therefore, the next step is to design a stable optimal discrete LQT control for path-following purpose. It must also be ascertained that the realization of Eqs. 6.2 is minimum by ensuring that (A, B) is controllable and (A, C) is observable, because multiple state-space realizations could be possible for the ASV dynamical system. This test was confirmed that each of the pair has been determined to have rank four, which is also the order of the identified ASV system. Having identified the system matrices as discussed, the design outline of optimal discrete LQT control is given below; whereas the detail account regarding the derivation of the proposed control design is given in Chapter 5.

1. First, the user-defined waypoints $r_k \; \forall \; k = 0 \sim N$ for the desired path were specified and store in the file of computer memory.
2. In the optimal control, minimization of some predefined performance index (PI) or cost function is performed utilizing the symmetric weighting matrices $Q \in \mathbb{R}^{4\times4} \geq 0$ and $R \in \mathbb{R}^{2\times2} > 0$ of system states and control inputs respectively, such as:

$$J_i = \frac{1}{2}(y_N - r_N)^T P\,(y_N - r_N) + \frac{1}{2}\sum_{k=i}^{N-1}[(y_k - r_k)^T Q\,(y_k - r_k) + u_k^T R u_k\,] \qquad (6.6)$$

where r_k, $y_k = Cx_k$, and u_k are the reference track, certain linear combination of states, and control at instant $k = [0\ N]$. Also, $P \geq 0$ is the solution to algebraic Riccati equation (ARE) [34], [35] given in Eq. (5.7).

3. The initial value v_0 of the auxiliary sequence v_k is computed offline by solving v_k, as given in Eq. (5.9), backward in time utilizing already stored waypoints r_k and the value is stored in the computer memory for later use.
4. As ARE does not depend on the state trajectory [34]; therefore, the sequence S_k, as given in Eq. (5.29), as well as the gain sequences K_k and K_k^v, as given in Eqs. (5.36) and (5.37) respectively, are computed offline prior to the control application to the ASV dynamics.
5. The LQT gains K_k and K_k^v are stored in the onboard computer memory for later use in the control run for controlling the motion of ASV.

The weighting matrices are determined by the genetic algorithm (GA) based multi-objective optimization technique discussed as follows:

6.7.2 Multi-objective Genetic Algorithm for Weighting Matrices Selection

The performance of discrete-time LQR and its extension such as LQT controllers depend on the state and control weighting matrices Q and R of the performance index, which are used as panelizing the states and control inputs respectively. Therefore in the present work, the best optimal choice for these weighting matrices is determined by the GA based multi-objective optimization technique using MATLAB® optimization toolbox™ and is called multi-objective genetic algorithm (MOGA); so that the selection of diagonal elements of PI weighting matrices Q and R ascertains the basic optimality criterion of the said control law.

A multi-objective optimization problem refers to finding a set of feasible non-dominated solutions, called Pareto optimal set, instead of a single objective optimization problem and, therefore, each solution satisfies all the objective functions to an acceptable level without being dominated by any other solution in the solution space with respect to multiple objectives. The values of the objective function corresponding to the Pareto set form the Pareto front [57], [58]. The GA, being larger class of the evolutionary algorithms, is regarded to be promising method for multi-objective optimization problems [59]. GA is basically a stochastic global search process which resembles the natural biological evolution phenomena. The GA algorithm starts randomly on the population of potential solutions without knowing the correct solution for the design variables beforehand. The principle of survival of the fittest is incorporated in GA algorithms using evolution operators such as reproduction, cross over, and mutation in order to have better and better approximations to a solution [57], [59]. Hence, the main function of the proposed method is to assure the convergence characteristics of GA based multi-objective optimization for searching of the optimal choice of weighting matrices so that the Pareto curve tends to approach the origin as much as feasible. The concise explanation of GA based multi-objective optimization technique is given in Appendix-A.

The most vital step in GA based multi-objective optimization technique is the characterization of objective functions. In this work, the two objective functions are developed and used as given in Eqs (6.7) and (6.8) below [50], [60]. For this purpose, simulation of the fundamental LQR design criteria as well as closed-loop response of the unit step inputs were utilized. If the tracking error is defined as $e = y - r$ [39]; then, Eq. (6.6) can be written as the first objective function:

$$F_1(Q,R) = \frac{1}{2} {e_N}^T P\, e_N + \frac{1}{2}\sum_{k=0}^{N-1}[e^T Q\, e + u^T R\, u\,] \tag{6.7}$$

and the second objective function is:

$$F_2(Q,R) = {}^{1}/_{c} + [\max(\text{real}(\text{eig}(\text{A} - \text{B K})))] \tag{6.8}$$

whereas c is a small positive number and adjusting coefficient to avoid computational error. In this technique, the diagonal elements of weighting matrices are tuned by the MOGA until the optimal controller gains K are obtained for the desired performance of the closed control system. The diagonal elements of both weighting matrices Q and R together constitute six decision variables x_d which are the key elements of objective functions. The weighting matrices can be written

$$Q = diag\,[\,q_1 \;\; q_2 \;\; q_3 \;\; q_4\,], \qquad R = diag\,[\,r_1 \;\; r_2\,], \tag{6.9}$$

and the decision variables vector is:

$$x_d = [\,q_1 \;\; q_2 \;\; q_3 \;\; q_4 \;\; r_1 \;\; r_2\,] \tag{6.10}$$

Therefore, the MOGA formulation for the LQR, and optimal LQT control, weighting matrices is written in the following form:

Minimize $\quad F\,(x_d) = [\,F_1(x_d),\, F_2(x_d)\,]$ (6.11)

subject to $\quad x_d(i) \geq 0 \;\; for\; 1 < i \leq n,$ and

$$x_d(i) > 0 \;\; for\; n < i \leq n + r$$

Initial values and the lower and upper bounds for the decision variables vector were specified as follows:

$$x_d = [50 \;\; 50 \;\; 30 \;\; 20 \;\; 5 \;\; 5];$$

$$x_{d,lb} = \left[0 \;\; 0 \;\; 0 \;\; 0 \;\; \frac{1}{10000} \;\; \frac{1}{10000}\right];$$

and $\quad x_{d,ub} = [8000 \;\; 8000 \;\; 8000 \;\; 8000 \;\; 500 \;\; 500]$

By default, the initial population of $N = 200$ individuals which is a set of points in the design space is generated randomly. Then, non-dominated measures such as: rank based on the relative fitness and crowding distance of individuals of the current generation are the criteria for the computation of next generation. These individuals are a set of possible solution to the problem at hand. Therefore, six individuals represent the diagonal elements of both weighting matrices. The multi-objective GA function in optimization toolbox of MATLAB® utilizes a variant of nondominated sorting genetic algorithm II (NSGA-II), which is known as control elitist GA,

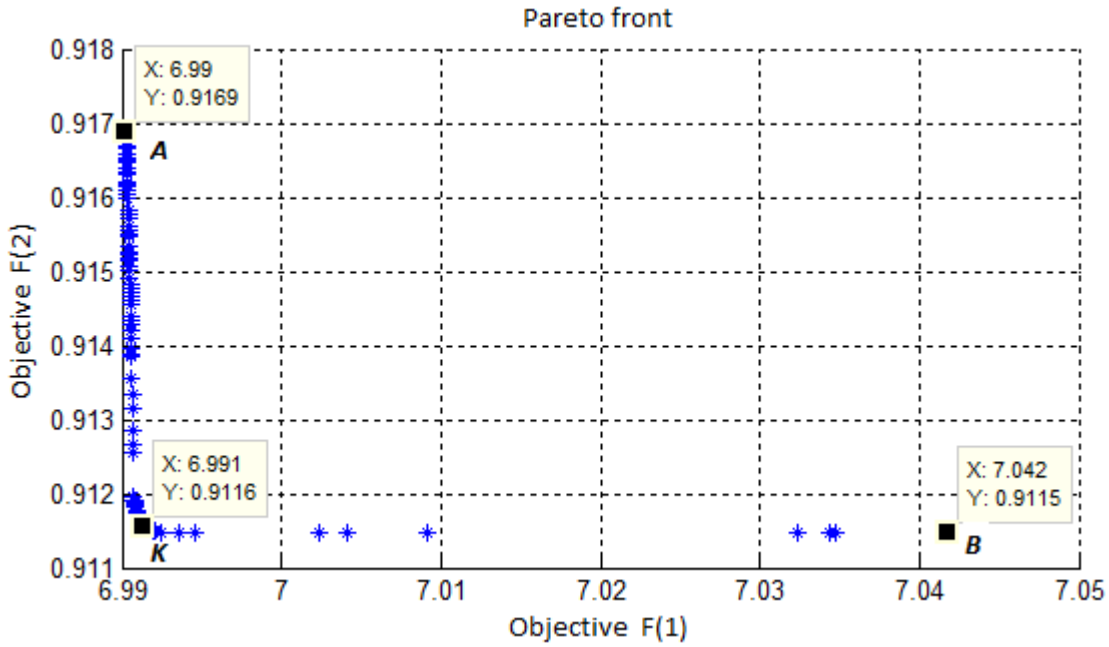

Figure 6.14 Pareto-optimal solutions with knee point "K_n".

in which individual with better rank or relative fitness value is selected but the increase in the diversity of population for convergence to an optimal Pareto front is ensured [58]. Different sim-

ulations were performed for achieving good configuration parameters for the GA. In this work, the configuration parameters selected are given in the Table 6.2, while other parameters are left as default. After each run of MOGA in MATLAB®, a set of solutions is provided instead of just one solution, and each one is called *individual* in the population, which is evaluated in bi-objective space formulated as above. During running of the MOGA, when the average change in the spread of Pareto front over the stall-generation limit, which is 150, is less than tolerance limit of 10^{-3}, then the generation stops and the Pareto-optimal or *trade-off* curve is obtained as shown in Figure 6.14. From the Pareto curve, an obvious *knee* i.e. point K_n of nondominated trade-off solution set for the elements of Q and R is apparently visible. The objective function F_1 increases by 0.014 % from point A to K_n and 0.73 % from point K_n to B, while the objective function F_2 decreases by 0.58 % from point A to K_n and 0.011 % from point K_n to B. Therefore, point K_n is an attractive compromised solution without the preference information. This points stands for a point of decreasing marginal objectives.

Table 6.2 Configuration parameters for MOGA

Size of initial population	200
Distance measure function	Crowding Distance
Domain of distance measure function	Genotype
Pareto front population fraction	0.5
Tolerance limit	1×10^{-3}
Stall generation limit	150

From Eq. (6.10), the trade-off solution for the decision variable vector x_d given by the point K_n is as follows:

$$x_d = [2808.6 \;\; 507.8 \;\; 3637.6 \;\; 279 \;\; 372.6 \;\; 34.72] \tag{6.13}$$

where the elements of x_d, thus determined from MOGA; populate the diagonals of optimal weighting matrices Q and R alternately as:

$$Q = \begin{bmatrix} 2808.6 & 0 & 0 & 0 \\ 0 & 507.8 & 0 & 0 \\ 0 & 0 & 3637.6 & 0 \\ 0 & 0 & 0 & 279.0 \end{bmatrix}, \quad R = \begin{bmatrix} 372.6 & 0 \\ 0 & 34.72 \end{bmatrix} \tag{6.14}$$

Once we determined the Q and R matrices, the feedback gain matrix K is calculated from Eq. (5.7) for the case of step inputs wherein matrix P is calculated by solving discrete algebraic Riccati equation (ARE) as given in Eq. (5.6). So the optimal feedback gain matrix that led to a good step response, as shown in Fig. 6.15, to both left and right motor step inputs is given as:

$$K = \begin{bmatrix} -36135.5 & 2.2067 & 13520.1 & 33828.7 \\ 10676.2 & -3.2803 & -3748.6 & -9735.5 \end{bmatrix} \tag{6.15}$$

The discrete linear time-invariant identified model of ASV as given in Eqs. (4.12) and (4.13).

6.8 Numerical Simulations for Controller Performance

The objective of numerical simulation study of under-actuated ASV's control design is mainly to verify the controller's desirable parameters for its effective performance and to assure its digital implementation for the waypoints path-following motion. In the path-following control strategy, the waypoints navigation system was utilized in which user-defined waypoints of various desired paths are provided sequentially as reference navigation points for optimal discrete LQT controller. A linear time-invariant system of ASV was developed in section 6.4 and later on the identified model along with the proposed controller designed in section 6.7 was then used for precise path-following simulations incorporating waypoints guidance. Therefore in this work, different types of computer simulations such as: the controller response to track the very basic

step inputs and also tracking the sinusoidal, zigzag, and S-shape maneuvering paths were carried out in order to test important characteristics of the controller as well as robustness to the randomness in the complex path curvature. These numerical simulations were carried out using MATLAB® control system toolbox™ and discussed as follows.

6.8.1 Simulation Results of Step Inputs

Simulations of the closed-loop linear quadratic tracker utilizing the previously identified linear time invariant system of ASV were performed by applying the unit step inputs for both motors. The step responses of the closed-loop system of ASV are analyzed in order to check robustness and hence adequacy of the proposed control parameters, i.e. the state and control weighting matrices Q and R, respectively, which are obtained from GA based multi-objective optimization. The optimal selection of these parameters is very important because they are used in the optimal LQT controller for tracking the waypoints of selected trajectories accurately. The weighting matrices thus obtained improve the step responses of both left and right motor inputs. From the step response of closed-loop ASV system for its both astern motors and under the optimal LQT control, as depicted in the Fig. 6.15, the four closed-loop performance criteria such as: overshoot M_p, rise time T_r, settling time T_s, and steady-state error E_{ss} were determined. The values of these performance parameters indicates that the ASV system has zero or limited overshoot; settling time and the steady state errors are also within admissible bounds. The values of these closed-loop performance parameters for unit step inputs for left and right motors are given in Tables 6.3 and 6.4, respectively. Also, from step response of the proposed controller, it is noted that the steady-state error for each step response is not perfectly zero but rather a very small number. Hence, it is concluded that the close-loop gain adjustment is also achieved with the help of two objective functions as described earlier so that a trade-off between the desired transient

response and the desired steady-state accuracy is accomplished; and permissible step input performance specifications are achieved. Usually, the steady-state error decreases with an increase in feedback gain values and vice versa [61].

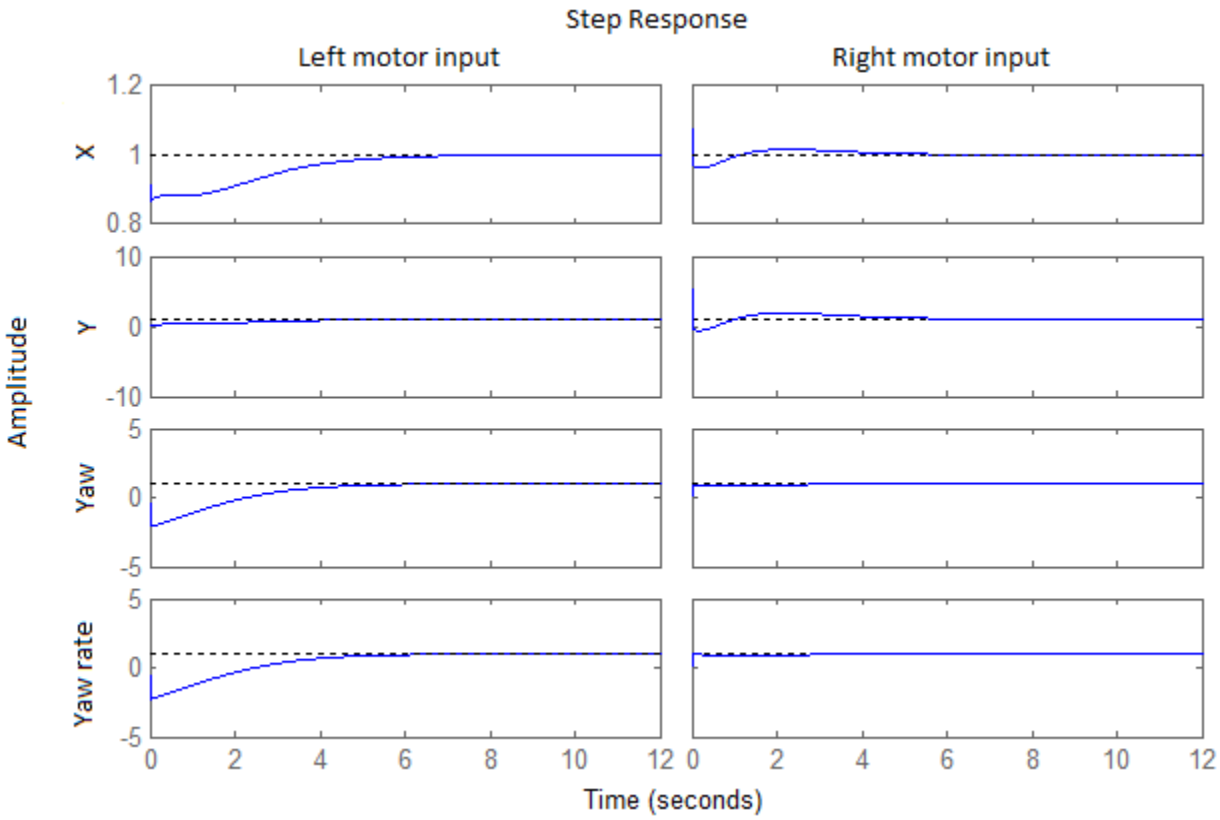

Figure 6.15 Closed-loop unit step response of the proposed controller.

Table 6.3: Step response parameters for left motor input

	$T_r(s)$	$T_s(s)$	$M_p(\%)$	E_{ss}
X	3.38	7.01	0	1.0×10^{-5}
Y	3.38	6.9	0	2.7×10^{-5}
Yaw	3.34	6.7	0	2.8×10^{-5}
Yaw rate	3.35	6.7	0	2.8×10^{-5}

Table 6.4: Step response parameters for right motor input

	$T_r(s)$	$T_s(s)$	$M_p(\%)$	E_{ss}
X	1.57	3.6	7.49	8.1×10^{-7}
Y	1.54	3.5	5.39	8.2×10^{-6}
Yaw	0.005	3.4	0	2.2×10^{-5}
Yaw rate	0.004	3.4	0	2.1×10^{-5}

6.8.2 Simulation Results of Tracking Various Trajectories

It can been seen from the simulation of proposed controller following sinusoidal, zigzag, and S-shape paths that simulated outputs given by the controller and the reference outputs are in good agreement. There are some small discrepancies found in simulated x-direction position of ASV which can be seen at the ends of zigzag and S-shape courses as given in Figs. 6.17 and 6.18 respectively. The designed discrete optimal LQT control could follow and converge to the desired reference paths pretty fast in all simulations. This is also apparent from the step response of the closed-loop ASV system under the proposed controller as discussed earlier. The exemplary trajectory paths chosen here, such as: sinusoidal, zigzag, and S-shape, are taken from several simulations that were performed and these are shown in Figures 6.16, 6.17, and 6.18. Their RMSE values for X- and Y-position tracking in simulation are given in Table 6.5 below.

Table 6.5: RMSE of X- & Y-position tracking in simulation

	X-direction	Y-direction
Sinusoidal Path	9.9411×10^{-3}	3.4332×10^{-3}
Zigzag Path	8.0744×10^{-2}	6.2907×10^{-3}
S-shape Path	1.8702×10^{-1}	1.0538×10^{-2}

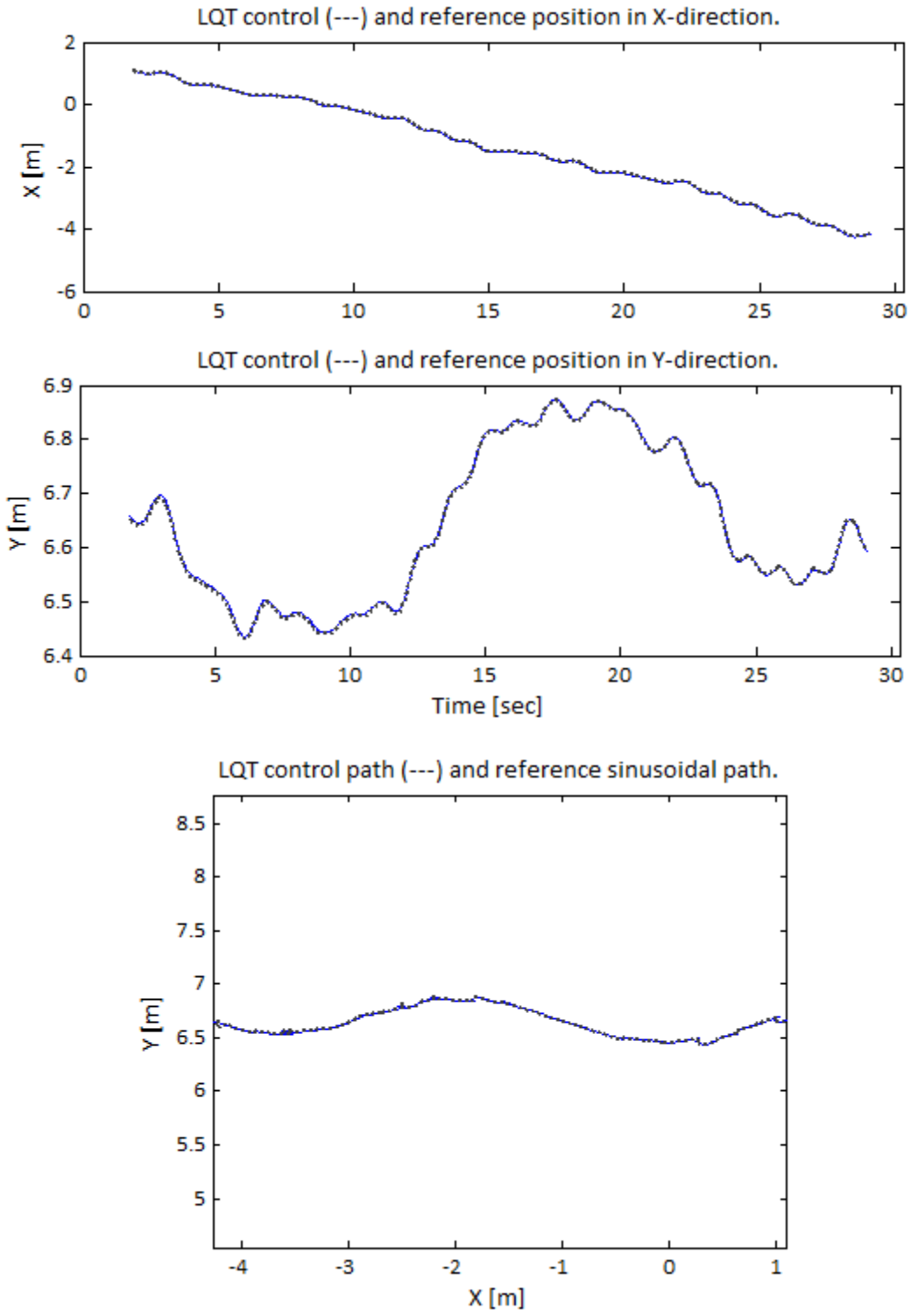

Figure 6.16 LQT control performance simulation of tracking sinusoidal path: tracking in X-direction (top), tracking in Y-direction (middle), and tracking sinusoidal path (bottom).

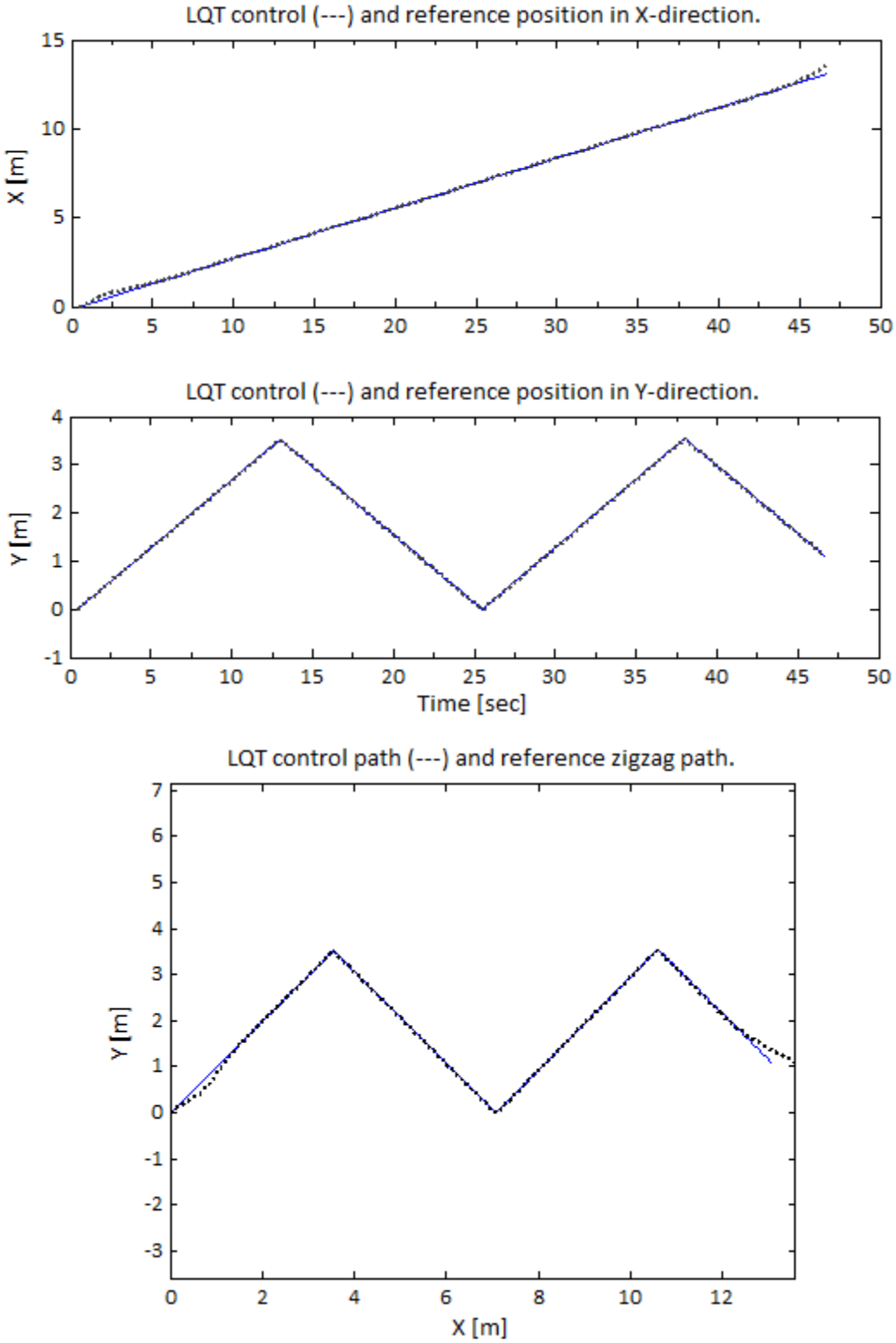

Figure 6.17 LQT control performance simulation of tracking zigzag path: tracking in X-direction (top), tracking in Y-direction (middle), and tracking zigzag path (bottom).

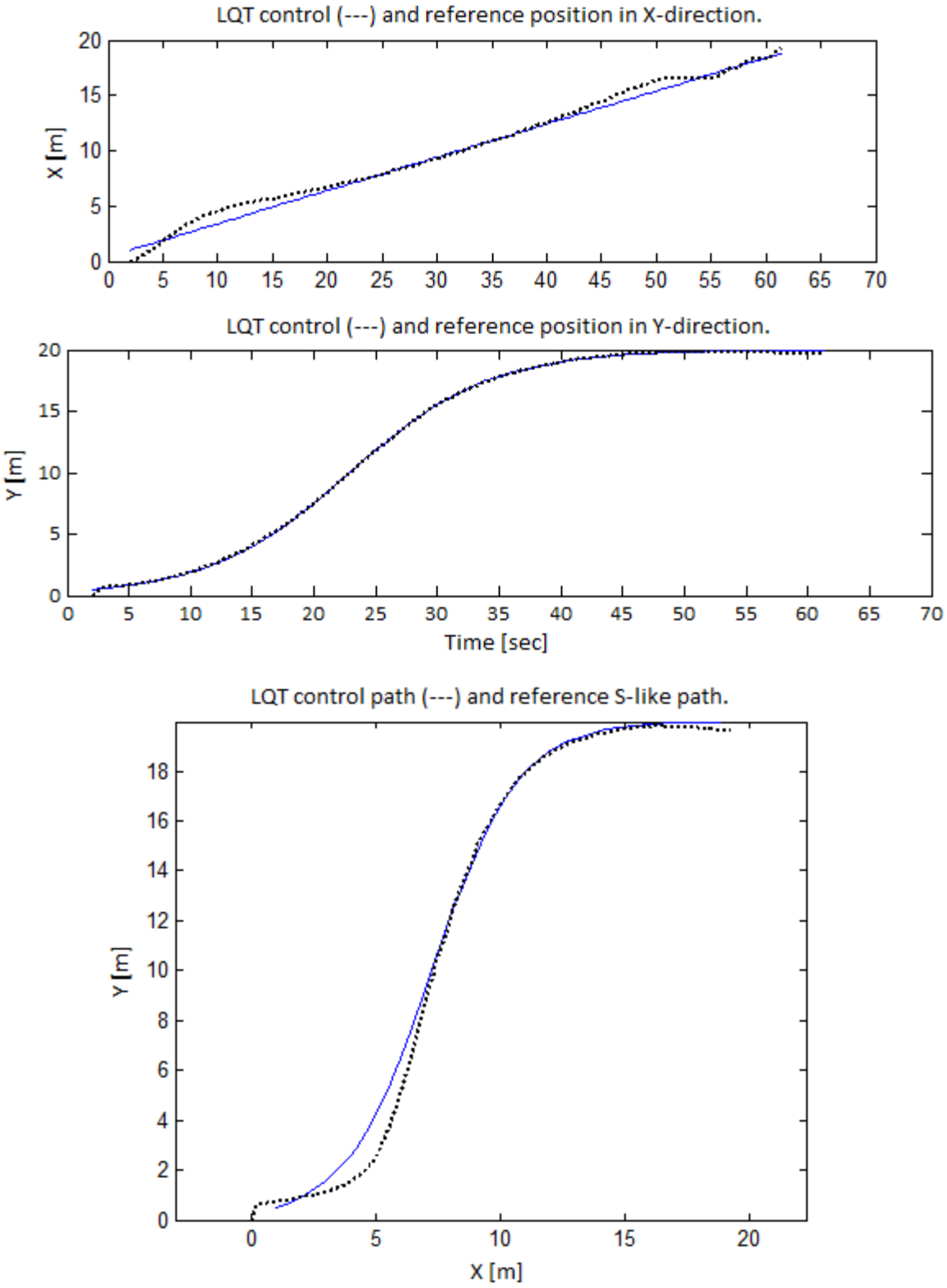

Figure 6.18 LQT control performance simulation of tracking S-shape path: tracking in X-direction (top), tracking in Y-direction (middle), and tracking S-shape path (bottom).

6.9 Discussion Regarding LQT Control Design

The problem of an effective design of optimal linear quadratic tracking control for path-following motion approach of ODU-ASV is addressed. The model of vehicle horizontal dynamics is identified by the OKID method for its subsequent control task under the assumptions that vertical motion dynamics were negligible and hence not considered in the design. Also, the speed of the ASV is coupled with the position of ASV in Cartesian coordinates, so the X and Y position coordinates, yaw, and yaw arte of the ASV are considered in its system identification task. In the controller design, Q and R weighting matrices were determined optimally with the help of a GA based multi-objective optimization method called multi-objective optimization GA (MOGA) so that performance index of the said controller is minimized with the balanced adjustment of both matrices. After designing the proposed controller optimally, numerical simulations were performed in order to test its performance incorporating the identified model of ODU-ASV dynamical system. The simulation results regarding step response show that the proposed optimal LQT controller offered much better efficiency in tracking the step inputs with admissible rise time, settling time, and overshoot. Due to optimal design of feedback gains values with the help of MOGA technique, the steady state error has also been greatly reduced which is also evident from the step response simulation. Moreover, the performance of controller during simulation of various paths, such as sinusoidal, arc-like and S-shape courses, was confirmed. The validity of the proposed controller was also proved in real-time water-trials by tracking the sinusoidal and arc-like paths reasonably well in the presence of environmental disturbance of wind and currents as well as error produced by noisy readings of low cost GPS.

6.10 Implementation and Experimental Validation of Control

In the previous sections, the discrete-time closed-loop optimal linear quadratic trajectory tracking controller or simply linear quadratic tracker (LQT) is designed over the entire range of the waypoints $[0\ N]$ for the ASV motion, where N represents the total number of waypoints for the desired trajectory path to be followed. In order to check effectiveness of the proposed control, its implementation strategy at both software and hard level is described here in detail; then, outdoor real-time water-trials were carried out and their results are presented in this section.

6.10.1 Implementation of Optimal LQT controller

The implementation of optimal discrete LQT controller involves both hardware and software environments. The proposed controller was implemented in hardware using the twin-motors of ODU-ASV system having onboard computer, microcontroller boards, and pose sensors such as: Memsense Nano IMU and low-cost 3DR uBlox GPS receiver for full state feedback that were considered. An overview of the experimental setup for twin-motors ASV was discussed in section 6.2. The only difference in the experimental setup for the real-time control testing is the use of GPS receiver for position measurement of ASV instead of Lidar. The software (programming code) implementation of the designed control as well as real-time data logging from Memsense IMU are coded in MATLAB®, while the position data logging from GPS receiver as well as the tasks of autonomous or manual mode via remote control of ASV are accomplished in open source Arduino-C programming language. Therefore, in this regard, the guidance code generates the desired position of ASV in the Cartesian coordinates by utilizing the user-given waypoints of desire path and provides the position data to the control module. Navigation code manages the sensors data, i.e. GPS receiver and IMU to generate ASV's position in X and Y Cartesian coordinates, yaw, and yaw rate as the state variables for the control module.

The control module generates the command signals for the port and starboard motors in a feedback loop by using guidance output commands and navigation sensors' outputs such as GPS and IMU.

The output commands of the control module are the control signal or manipulated variables which are then mapped by the trained neural networks, one for each motor, with the executable control commands representing PWM voltage values. The mapping is discussed in the forthcoming section 6.10. The PWM values are then sent to the two actuators that control the position and heading of the ASV by producing appropriate thrust T by each propeller. The working layout of the proposed control system is illustrated in Fig. 6.19 below. The main steps of the optimal LQT control implementation in real-time are summarized as follows:

1. The waypoints r_k of the desired track are known in advance and stored in the onboard computer memory, where k is the index 0 to N, the total number of waypoints.
2. The initial value v_0 of the auxiliary sequence v_k is computed offline by solving v_k, as given in Eq. (5.28), backward in time utilizing already stored waypoints r_k and the values are stored in the computer memory in order to be used for real-time control validation testing.
3. As ARE does not depend on the state trajectory [34]; therefore, the sequence S_k, as given in Eq. (5.29), as well as the LQT gains sequences K_k and K_k^v, as given in Eqs. (5.36) and (5.37) respectively, are computed offline prior to the control application to the ASV dynamics. These values are stored in the onboard computer memory for later use in the real-time control validation run for controlling the motion of ASV.

During the actual control run, the only work is to solve Eq. (5.34) for v_{k+1} starting with the initial condition v_0 as well as to compute the optimal control using Eq. (5.29) at each time

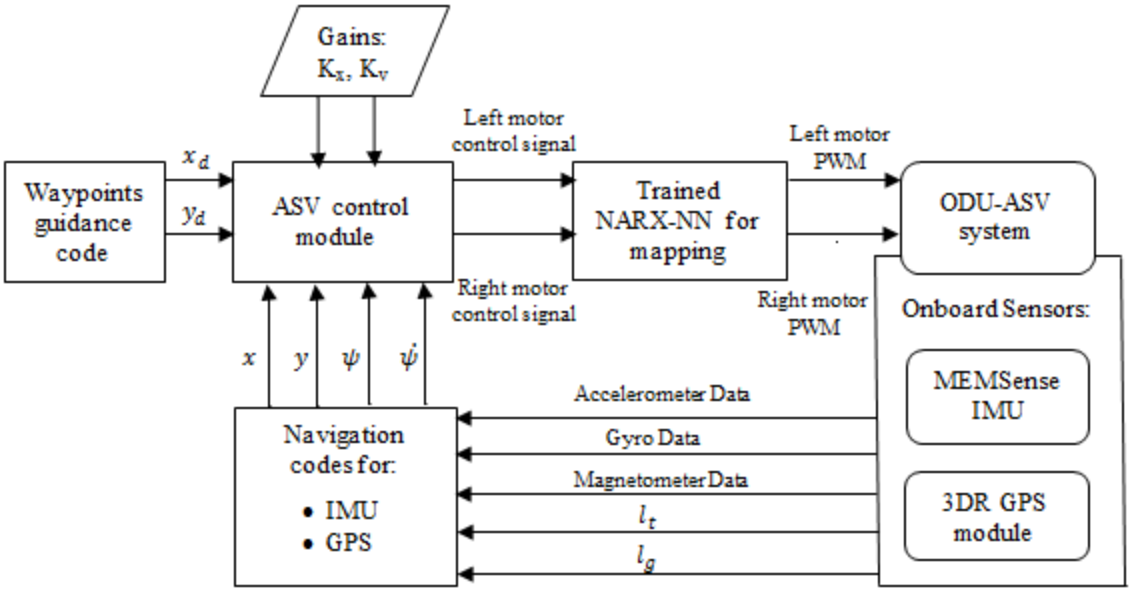

Figure 6.19 Waypoints guidance, navigation, and control system for ODU-ASV.

step, and then the control values are sent to drive-Arduino through Wi-Fi bullets for controlling the two astern motors of ASV.

6.10.2 Neural Network Mapping of Control Signal

Neural networks (NNs), the artificial intelligence techniques, have the ability to characterize the complex nonlinear relationship between two or more data types that is inputs and target outputs of the neural network. As in many dynamic time-series data applications, significant correlation exists between the modeled time-series and independent external data set. Therefore, it has been observed in the present work that there is nonlinear relation exists in the mapping of control signals obtained from the optimal discrete linear quadratic tracking controller and their corresponding actual PWM voltage values required for controlling the two onboard motors of ASV. Each of the respective PWM signals, in turn, rotate the port and starboard motors which

induce the thrust for forward motion of ASV. Therefore, there is scope to apply NNs in this domain of problem. In order to achieve the nonlinear mapping task through NNs, MATLAB® neural network toolbox™ was used, which is discussed in the forthcoming sections.

6.10.2.1 NARX NN Architecture for Nonlinear Mapping

There are many different kinds of NN architectures available, but the proposed NN-based nonlinear mapping between two different time-series data, such as: control signal or manipulated variables and their respective actual PWM voltage values for each motor, is characterized by nonlinear autoregressive with eXogenous inputs (NARX) feedback neural network. This network, based on the linear ARX model, is recurrent dynamic network having feedback connections enveloping many network layers as given in following Eq. 7.1.

$$y_{k+1} = f\left(y_k,\ y_{k-1}, \cdots,\ y_{k-n_y+1};\ u_k,\ u_{k-1}, \cdots,\ u_{k-n_u+1}\right) + \epsilon(k) \tag{7.1}$$

It is clear from Eq. 7.1 of NARX network that future value of the dependent output signal i.e. y_{k+1} is regressed on the n_y past values of the output signal y and n_u past values of the independent eXogenous input signal u. Where $k = 0,1,2, \ldots$ and $n_y \geq 1$ and $n_u \geq 1$ subject to $n_y \geq n_u$ represents the memory orders for tapped delay lines of the input and output, respectively. The nonlinear function f in Eq. (7.1) corresponds to nonlinear mapping, which is unknown in advance and can be approximated by the training of a feed-forward multilayer perceptron with the help of the network weights and biases as shown in Fig. 6.20. The term $\epsilon(k)$ is the approximation error of the signal y at time step k which needs to be minimized during training of the network [62]. A rather detail account of the NARX network is given in the Appendix-B.

Various network topologies for mapping of both left and right motor control values according to complexity level can be chosen by specifying the number of hidden layers and the number of neuron per layer as well as various training algorithms; this sort of selection was

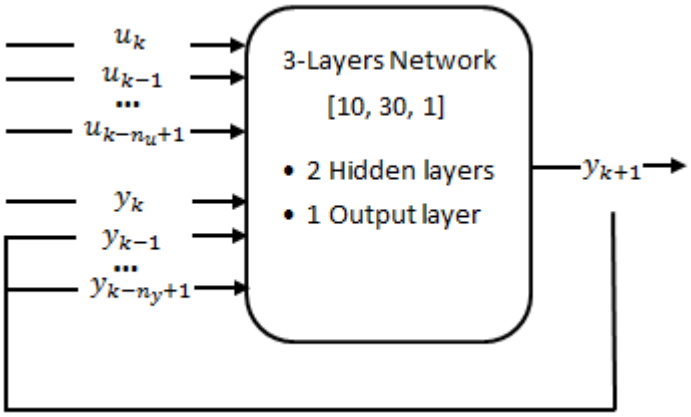

Figure 6.20 Nonlinear autoregressive with exogenous inputs (NARX) neural network.

optimized through trial-and-error process to have network topology that could render the best performance. Therefore in the present work, separate NARX network was designed for the mapping of control values of each left and right motors, which avoids over-fitting of the input data to the targets while the generalization and computing power of each network increase instead. For the mapping problem at hand, the structure of each NARX network comprises of an input layer, two hidden layers, and an output layer of neurons which accomplish separate tasks. The input layer is actually the input fanning which is used just for summing the inputs and biases (if any) to the network and consists of 2 neurons for the two series of eXogenous inputs which made up of left and right motor control values obtained from the designed controller and 1 neuron for the output signal as provided to the network. The two hidden layers having [10, 30] neurons respectively take information from the input layer and compare it with the desired targets using supervised learning rule. For this purpose, the nonlinear activation functions such as *tan-sigmoid* and *log-sigmoid* are used for the neurons of first and second hidden layer respectively; while the lin-

ear transfer function is used for one neuron of the output layer. The selected parameters of each NARX network design are tabulated in the Table 6.6 below, while other required network parameters are remained as default.

Table 6.6 NARX network structure used for mapping

Number of input neurons	03
Number of hidden layers	02
Number of neurons for each hidden layer	[10, 30]
Number of input and output/feedback delays	7, 7
Number of output neurons	1
Training function	trainlm
Network performance function	mse
Transfer function of neurons in 1st hidden layer	tan-sigmoid
Transfer function of neurons in 2nd hidden layer	log-sigmoid
Output neurons' transfer function	Purelin

6.10.2.2 Data Collection and Pre-allocation for NARX Network

The actual PWM data for both left and right motors has been collected at the sampling rate of 140 Hz from 5 out of 15 total outdoor water-trials; and are considered as the first discrete time-series data set as y_k to each network. The discrete exogenous input data set u_k for each network is the control values of both left and right motors which are obtained from optimal discrete LQT controller. These two time-series of exogenous input data are obtained from the perfect simulation of the proposed controller for trajectory paths of those five outdoor trials; the perfect simulation means that when the actual trajectory paths and the simulated response of the proposed controller match perfectly then their corresponding control inputs, i.e. the actual PWM values and controller values after simulation, were collected and used for the training of each left

and right motors' network. The total selected data set from the five outdoor water-trials was 6000 samples consisting of actual PWM values and their corresponding controller values from simulations to form two time series for each motor of ASV. This entire training data set has been divided randomly into three different data subsets, namely: 70%, 15%, and 15% of the total data set were used for the purpose of network training, validation, and testing respectively. The unused inputs and target data set of one of the 15 water-trials are kept aside for ensuring the validation of each neural network design.

6.10.2.3 Network Training

The NARX network for mapping the controller values of each left and right motor has been trained using Levenberg-Marquardt backpropagation (LMBP) learning rule. As LMBP is designed to approximate the second order derivative of the network errors without calculating the Hessian matrix *H*; so it is, most of the time, the fastest backpropagation algorithm [62], [63]. The Jacobian matrix *J* is utilized for calculation assuming the performance function to be mean square error *mse*. During LMBP training algorithm, weights & biases are automatically updated iteratively according to its error after the complete input-output training data set is provided to the network. The network weights and biases were initialized by the default function of MATLAB® neural network toolbox™. The LMBP training algorithm is also elaborated in the Appendix-B. A *series-parallel architecture* is used for training the NARX network because the actual outputs are available during the training instead of feeding back the estimated outputs as shown in the Fig. 6.21. Each tapped delay line of input and output were provided with initial 7 data points of given training data set. During training of the network, validation data is utilized for measuring network generalization; so when the network generalization is not improving for

some specified epochs, then the network training is halted and trained network is evaluated afterwards. The training parameters for both the networks are given in Table 6.7 below.

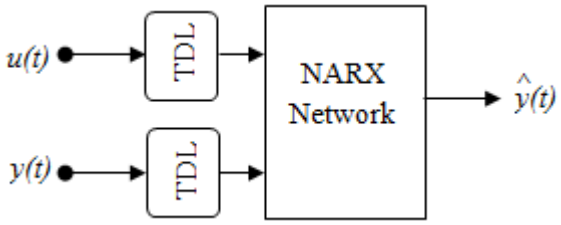

Figure 6.21 Series-parallel architecture for NARX network's training for mapping purpose.

Table 6.7 Training parameters for each NARX network

Max. Number of epoch	800
Performance goal set *(mse)*	1×10^{-6}
Performance goal achieved *(mse)*	0.094916 (left) & 0.47686 (right)
Regression R values	0.99969 (left) & 0.99968 (right)
Training data divide function	*dividerand*
Training parameter max. fail	8

It is clear from the networks performance during training as shown in Fig. 6.22, that there is no major problem with the training stage. The validation and test plots are rather similar; it implies that the test curve did not increase before rising of validation curve indicating no significant over-fitting has occurred. This means that network did not just over fit the network inputs to the targets during training and hence the network generalized well for the new data set.

The network validation is also confirmed by the regression plot which shows the relation-

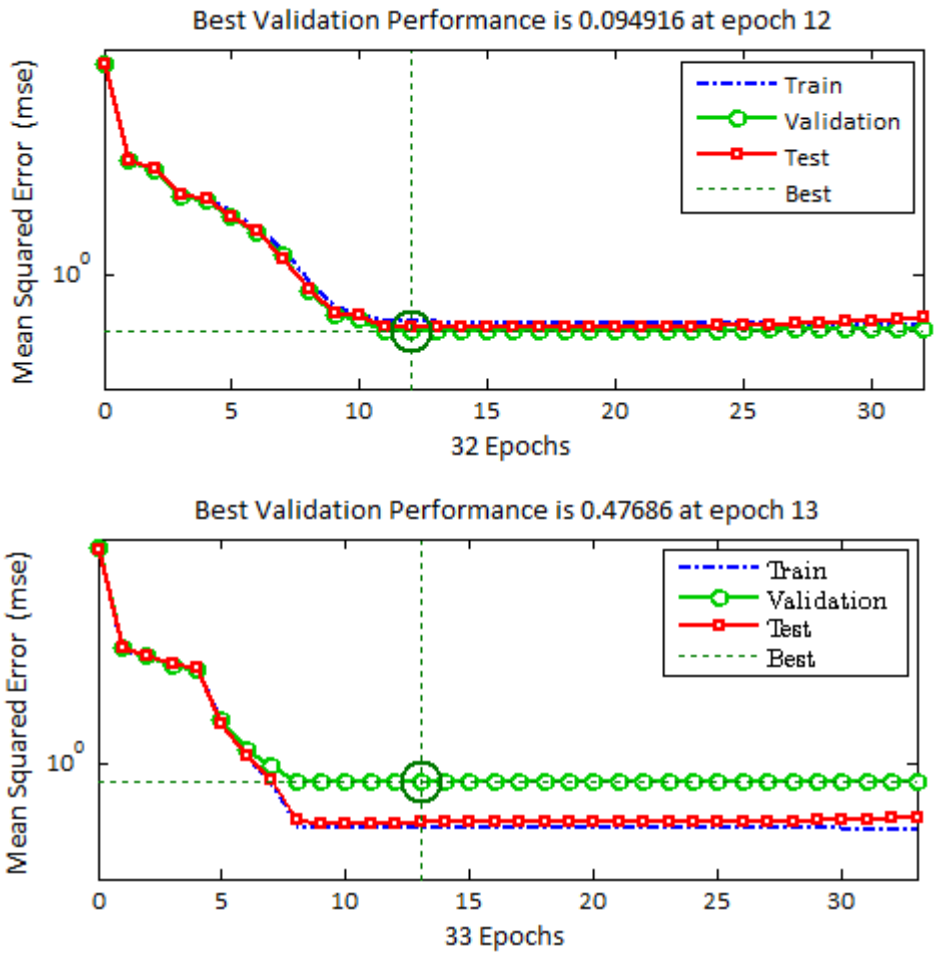

Figure 6.22 Network performances during training: mapping of left motor values (top) and right motor values (bottom).

ship between the network outputs and its respective targets denoted by the R value. If the network output is exactly equal to the corresponding targets then R is equal to one, which indicates that the network training is perfect; but this rarely happens in real practice. The regression plots for left and right motor inputs are given in Fig. 6.23. It is apparent from the R values in the regression plots that the networks' training for both left and right motors' PWM values is adequate.

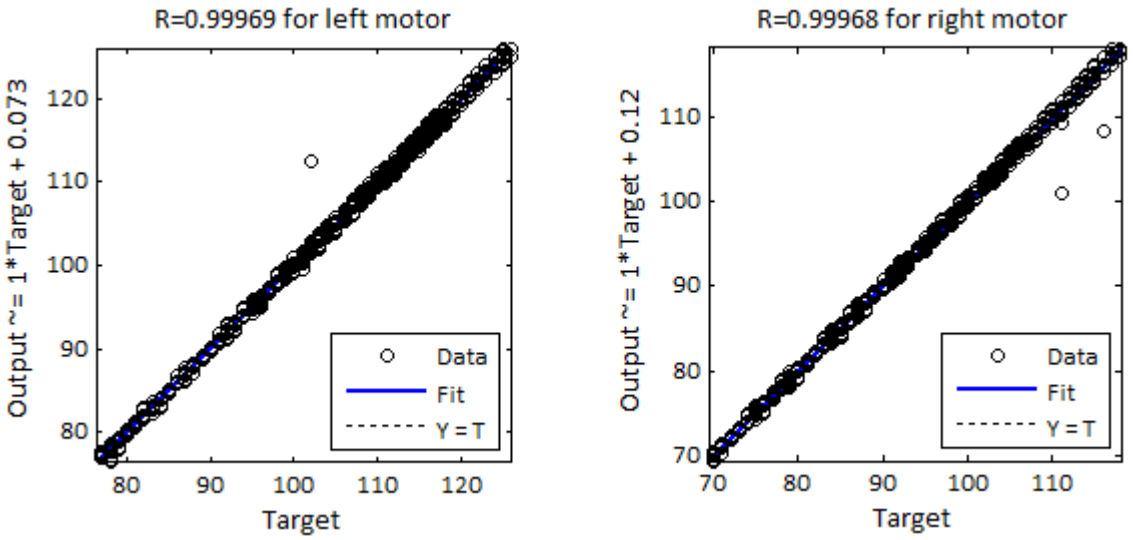

Figure 6.23 Regression plots for checking validation of both networks.

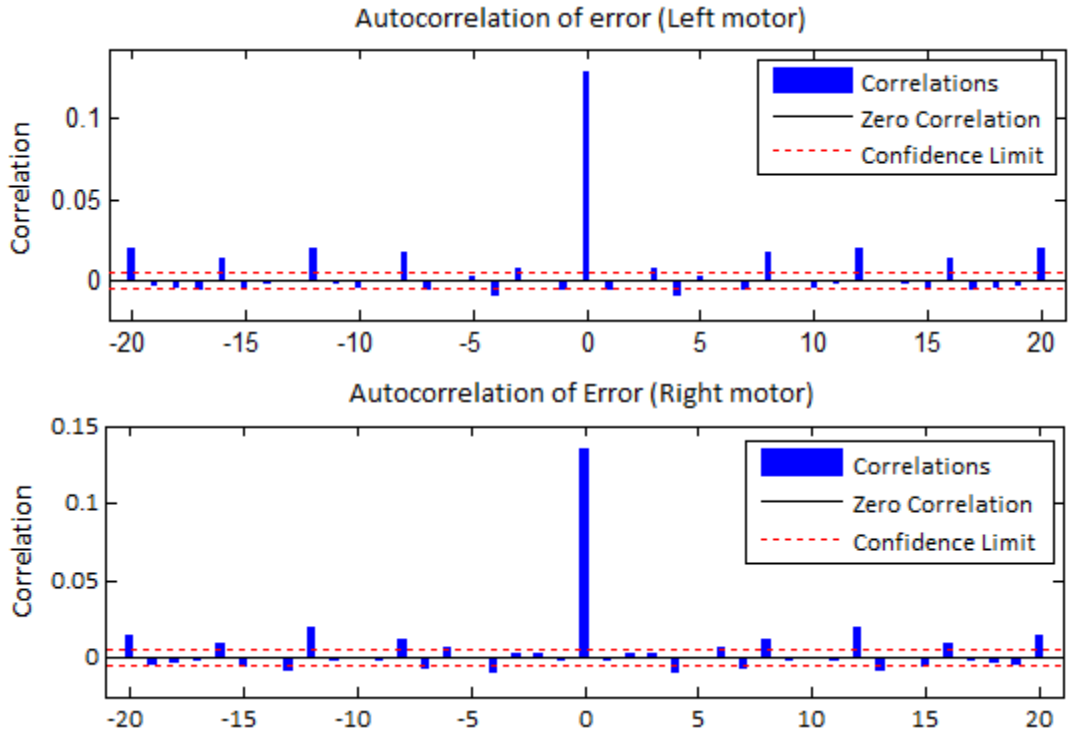

Figure 6.24 Error autocorrelation function for checking validation of both networks' performance

The network performance in terms of *mse* metrics for both left and right motors mapping is validated by the error-autocorrelation function as shown in Fig. 6.24. The autocorrelation function (ACF) of errors produced by each network, in the case of left and right motors mapping, shows how the residuals of each network are related in time. From Fig. 6.24, it can be seen that there is only one nonzero value *mse* of the ACF at zero lag for each mapping network of left and right motors, which indicates that residuals are completely uncorrelated with each other. The correlations of residuals, except for the one at zero lag, are approximately within 95% confidence limits around zero. Additional confirmation regarding the tracking performance of each network is obtained by performing input-error cross-correlation function as given in Fig. 6.25, which shows how the residuals are correlated with the input sequence. In this case, all the cross-correlations are within 95% confidence limits around zero. It is concluded that each network is trained adequately for mapping of left and right control signals to their respective PWM values.

6.10.2.4 Evaluation and Implementation of the Trained Network

In order to check the generalization behavior of each of the network and see its response to new inputs, it is tested for the fresh unused input data to the network that it has never seen before. For this purpose, the data set consisting of inputs and target outputs for both left and right motor case were already kept aside is used. Initial values of 7 data points for each tapped delay line of input and output were provided as extrapolated data points from the start of the training data. The weights and biases were initialized randomly by the MATLAB® function. The trained networks were tested against new data that was not utilized in the training task. The results of mapping rendered by the trained networks for PWM values of both left and right motors are shown in the Fig. 6.26, which shows perfect match of actual PWM values and the NARX given PWM values. The network prediction capacity was additionally confirmed for multistep ahead

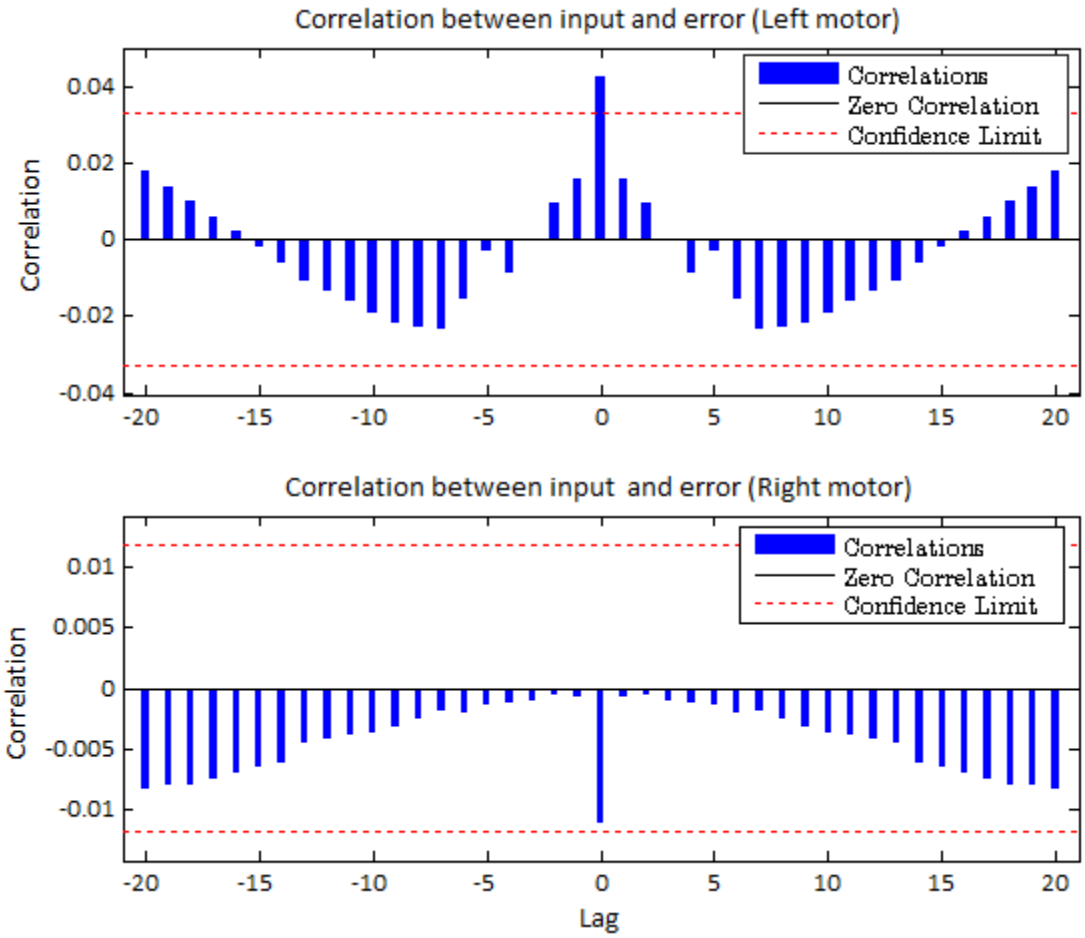

Figure 6.25 Input-error cross-correlations for checking validation of both networks' performance.

prediction in closed-loop format after following the known sequence as given in Fig. 6.27. After validation performance, the well trained network is saved and utilized for real-time control validation testing; the results therein shows that the trained networks work properly online for mapping purpose.

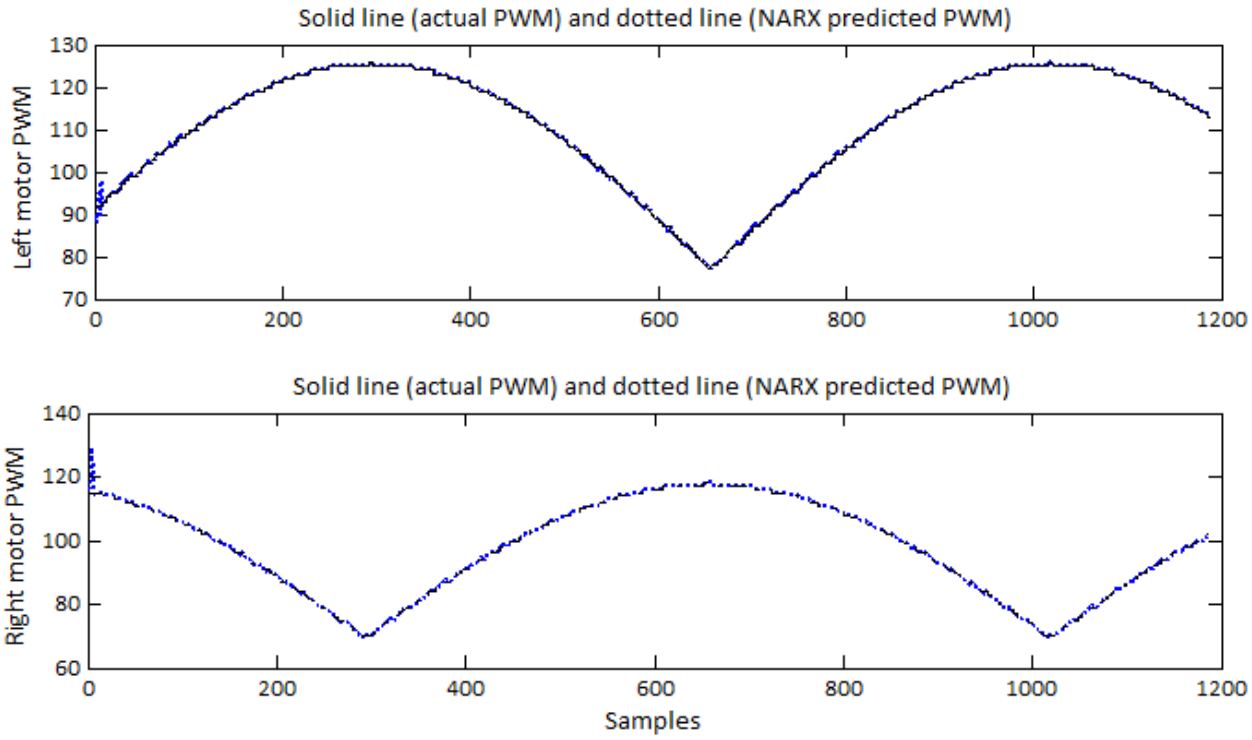

Figure 6.26 Mapping prediction of left and right motors' control input values.

6.10.3 Real-time Experimental Validation of Control

The real-time experimental validation of control is a vital and final step in the process of testing the accuracy and adequacy of the designed controller at hand. In this step, the right decision is normally taken regarding validity of the controller after proving its adequacy. The real-time validation water-trials of the ASV having the proposed control incorporated were performed on the Lafayette River. As already discussed, that a series of user-defined waypoints in terms of Cartesian coordinates for the desired paths to be followed are specified before starting the real-time water-trials. The control values for both motors are computed in MATLAB® with the help of designed optimal feedback LQT control so that the ASV is aimed at the reference waypoints

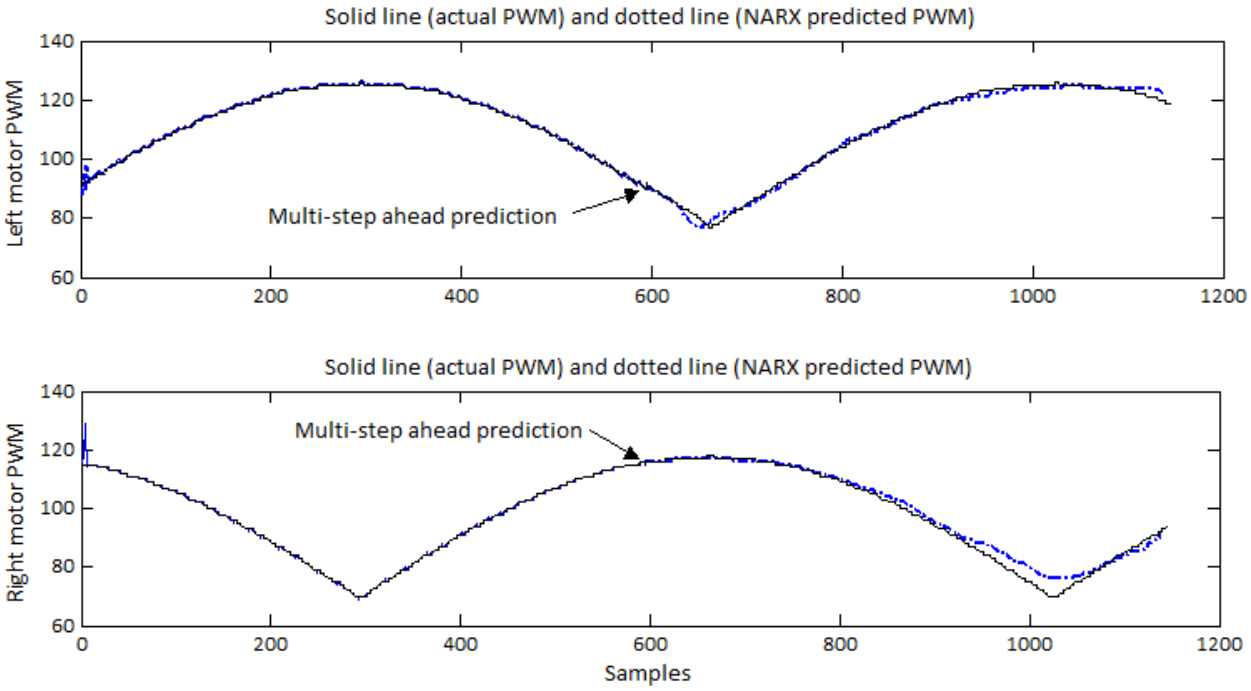

Figure 6.27 Mapping prediction of left and right motors' control input values with multistep ahead closed-loop prediction.

sequentially in the presence of disturbances due to wind and currents. In order to achieve the required position while following the specified trajectory, the ASV is propelled by the differential thrust of two motors installed on the stern of ASV. Prior to start the real-time path following control run, the ASV was given motion in manual mode through R/C transmitter for a couple of seconds and then the autonomous feedback controller mode is initiated. The computed control values in real-time for both left and right motors are mapped by the trained networks to get the PWM values which were send to the drive-Arduino to run and control the two outboard motors through speed controllers. Hence the required waypoints were followed.

In these tests, the sinusoidal and arc-like trajectory paths were selected due to their

smooth curvature and their waypoints as the reference steering commands are stored in the onboard computer memory for real-time testing the designed controller response to those points. The selected maneuvering paths are more common in routine ASV dynamical motion. Even without DGPS technique or real-time kinematic (RTK) GPS corrections, the system model and designed controller proved to be working properly. The performance of the control system of ASV was validated in real-time for sinusoidal and arc-like paths as shown in the Fig. 6.28. The NRMSE values for both paths are given in Table 6.8 and it shows that the proposed controller is efficient in error minimization under persistent disturbances from the environment.

Table 6.8: NRMSE of X- & Y-position tracking in real-time validation

	X-coordinate	Y-coordinate
Sinusoidal Path	0.7231	0.7074
Arc-like Path	0.7964	0.8067

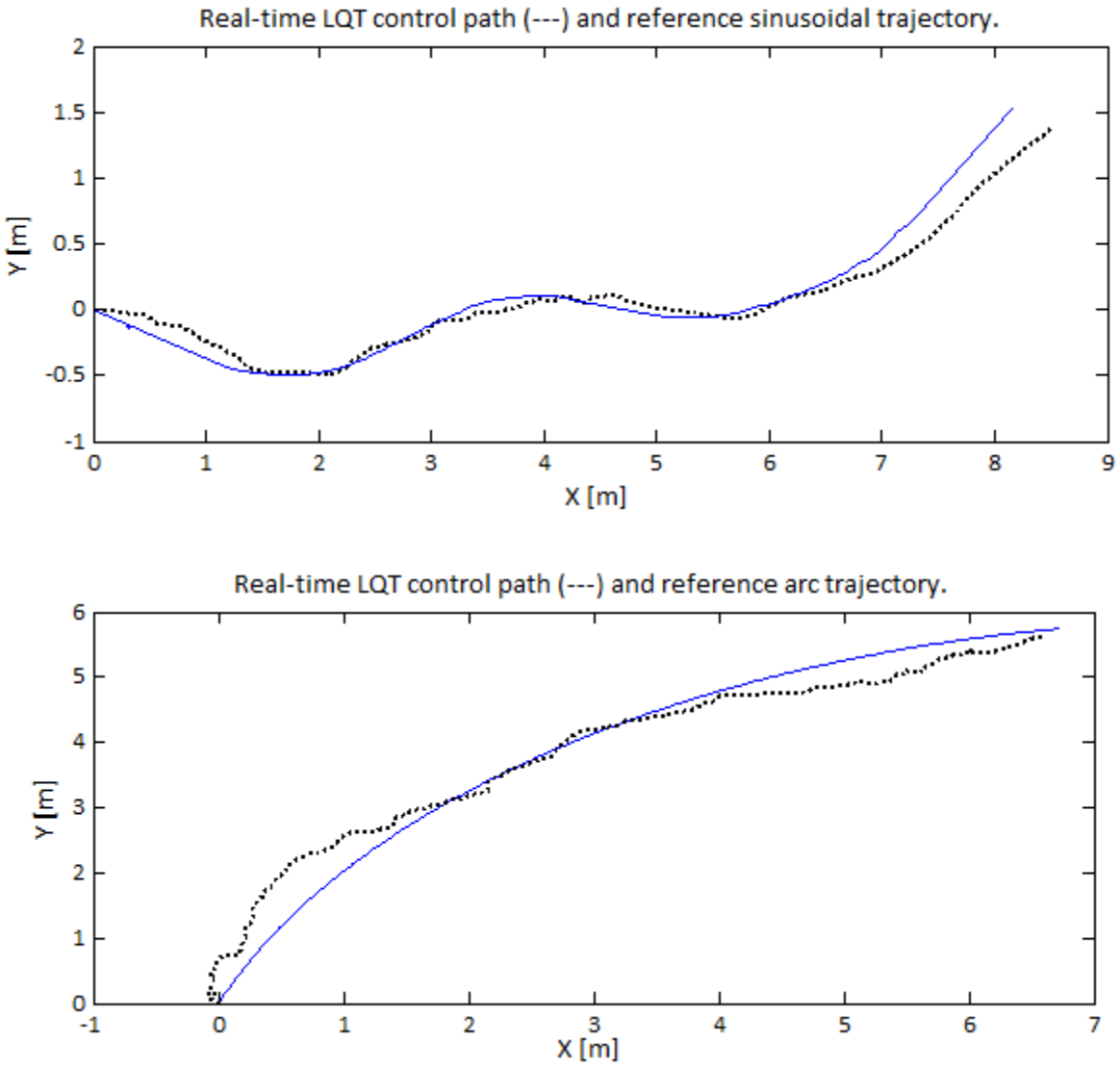

Figure 6.28 LQT control real-time path-following performance: tracking of sinusoidal path (top) and tracking of arced path (bottom).

6.11 Concluding Remarks

In order to identify the system dynamics of ODU-ASV *Big Blue*, some systematic tasks of the research work are accomplished in order such as: the acquisition of raw data from the outdoor real-time water-trials, preprocessing of acquired raw data, system identification algorithm i.e. OKID passes and analysis, assessment tests of the identified model, design, simulation & implementation of the controller for ASV, and finally the real-time validation testing for the identified model of ASV and its subsequent controller.

The linear time-invariant state-space model of order fourth of the ASV is identified from experimental data utilizing the OKID method as well as ERA/DC algorithm. Most of the time sinusoidal, zigzag, and arc-like maneuvers were performed by the ASV during the outdoor water-trials in order to fully assess that the validity of its dynamics and optimal feedback control design is executable. The length of the data record was quite reasonable from the 15 water-trials of different trajectory paths. The accuracy of the identified model for ASV dynamical has also been quantified with the benchmark residual analysis, and the results of these tests show that the OKID-identified could adequately be used for the design of closed-loop state feedback optimal linear quadratic tracking controller for its path-following control strategy.

Irrespective of all the limitations aforementioned in this work, the proposed method presented in this research brought about to a good linear state-space model of ASV and based on the identified model, the optimal discrete linear quadratic tracking control for its motion control was designed and validated in real-time in the presence of environmental disturbance of wind and currents.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

7.1 Conclusions

In this dissertation, the open-loop system identification of ODU-ASV and its optimal control design based on the identified model has been proposed. The observer Kalman filter system identification is utilized. The OKID technique, originally developed at NASA, together with the combination of its complementary algorithm of ERA comprised the OKID/ERA or OKID/ERA-DC has been proved to be a successful algorithm that has potential to identify the discrete linear time-invariant state-space system in the time-domain as the system matrices $A, B, C,$ and D along with the steady state Kalman filter gain matrix simultaneously from the experimental input and output data [38], [49], [64], [65]. Hence, the OKID methodology proved to be powerful alternative to the traditional modeling technique called *modeling by first principles*. The identified state-space model obtained from the experimental data represents the behavior of the dynamical system of ODU-ASV. The length of the experimental input and output data collected from water-trials was selected to be sufficiently large, so that the dynamics of the ODU-ASV is fully assessed. From the experimental input output data, the observer Markov parameters (OMPs) are obtained from the set of Markov parameters of the state-space observer equation. The OMPs are the combination of system Markov parameters and observer gain Markov parameters. The system matrices $A, B, C,$ and D are computed from the former using ERA algorithm while the observer gain matrix G is determined by the latter parameters. A fourth-order model was selected for the discrete linear time-invariant state-space system after performing sin-

gular value decomposition. The benchmark assessment tests have been performed including: data reconstruction from the identified model, residual analysis, and cross-validation tests, in order to show the validity and accuracy of the identified model of order four for the ODU-ASV dynamics. After having the linear state-space model, the optimal LQT controller is designed for achieving path-following task while minimizing the predefined performance index; during this process, the state variable feedback gains are obtained utilizing weighting matrices that are found by multiobjective optimization genetic algorithm (MOGA) technique. Finally, the neural network is trained for mapping the controller command and the actual PWM values required for the two trolling motors and then stored the trained network in order to use in real-time control run for mapping purpose. The results of real-time water-trials confirm the validity of the OKID identified model of ODU-ASV and its proposed controller.

7.2 Future Extension of the Research

For horizontal planar motion control of the ODU-ASV, the optimal discrete linear quadratic path-following controller is designed and implemented in this research considering sway motion negligible. Also, the temporal state of ASV in the forward X direction i.e. velocity was not included in the system identification process due to the fact that ASV, being slow moving vehicle with the velocity of approx. 1 m/s, is coasting with nearly constant velocity during path-following control scheme after the initial acceleration for a couple of seconds; and it keeps moving at that speed until the time it stops moving at the end of its path-following task.. The future work is proposed to be focused on the OKID system identification with the addition of velocity states. The ASV dynamical motion in backward direction was not tested and studied for its system identification task due to time constraint. The dynamic characteristics of the reverse motion of ASV might be slightly different than the forward one due to the expected error in finding the

location of CG point, the difference in hydrodynamic shape of the aft and fore of the ASV, as well as the variation of thrusting power between the port and starboard trolling motors; therefore, the identification experiments need to be performed for finding the reverse dynamical motion model for ASV so that its entire motion characteristics would be identified. Additionally, the specific optimal discrete linear quadratic tracking control is also designed and implemented for the reverse motion control of ASV.

A way-point guidance approach was used in this research in which different sets of way-points of the predefined paths, which are already stored in the computer memory, were provided for the guidance of ASV. This was done to ensure the validity of OKID identified model of ASV and its subsequent designed controller for its motion. A more robust and fully autonomous guidance strategy can be used for the motion control of ASV ranging from simple to more complex ones such as: variable radius line of sight (LOS) guidance system [14], [66], proportional navigation (PN), and guidance of ASV based on fuzzy logic and trained neural networks (NN) algorithms.

NN are nonlinear learning algorithms acting as universal approximators; therefore, they can be utilized to cross calibrate the GPS signals with the more accurate LIDAR measurements for a large set of data for various maneuvering paths. For this purpose, various NN algorithms such as the feedforward multi-layer perceptron networks trained with the Levenberg-Marquardt algorithm or the input-output nonlinear model of neural network can be used. In this work, a low-cost GPS receiver is used for position estimate of the ODU-ASV during the real-time control run so that the position data is utilized as position states feedback in the proposed control law for computing the control inputs. As the GPS position data of ASV is more erroneous than that of the onshore LIDAR position measurements. Therefore, it has been attempted to cross-calibrate

the position data of ASV given by GPS with the simultaneously recorded position data from the LIDAR with the help of neural network method; this was the intention to have more accurate GPS position data after passing its raw position data through trained neural network in real-time. However, due to time constraints, this intended work is recommended to be done in the future.

On-line Observer Controller Identification (OCID) augmented by on-line neural network (NN) routines will also be the next future work concern. This new research idea will greatly help in avoiding the discontinuity in controlled motion of ASV just in case the ASV's position coordinates given by GPS are missing due to no connection with the communication satellites. Consequently in this scenario, the predicted controller commands would be provided by ANN to the ASV dynamical system in order to have unobstructed motion of the ASV. Moreover, if we use the fastest computer processor and subsequently the processing time of the algorithm would be enhanced enough, then the OCID method will be used to find the controller gain values directly of the ASV forward dynamics in online fashion and use them in the feedback control.

The outdoor water-trials were performed in a semi-circular area having diameter of 16 meters because this is the accurate measurement range of the LIDAR. Therefore, it is suggested to perform full-scale sea-trials on a comparatively large-size ASV having long distance accurate LIDAR or DGPS/RTK GPS along with the existing expensive MEMSense IMU for the accurate readings of position, velocity, and compass information of the ASV. The OKID methodology could be performed in order to identify its full-scale dynamical model. Moreover, the marine environment induced disturbance signals such as waves, tidal currents, and wind condition is suggested to be included as error dynamics system besides the main identified system; these environmental disturbance signals can be measured by their respective sensors, so in this way a robust controller can be designed for the ASV move in open sea with more harsh environment.

REFERENCES

[1] Manley, J. E., (2008). *Unmanned surface vehicles, 15 years of developmen*t. In: Proceeding Oceans 2008 MTS/IEEE Quebec Conference and Exhibition, Quebec City, Canada. pp. 1-4.

[2] Kitts, C., Adamek, T., Rasal, K., Howard, V., Li, S., Badaoui, A., Kirkwood, W., Wheat, G., Hulme, S., (2012). *Field operation of a robotic small waterplane area twin hull boat for shallow-water bathymetric characterization*. Journal of Field Robotics, (29)6, pp. 924-938.

[3] Murphy, R., R., Steimle, E., Hall, M., Lindemuth, M., Trejo, D., Hurlebaus, S., Medina-Cetina, Z., Slocum, D., (2011). *Robot-assisted bridge inspection*. Journal of Intelligent & Robotic Systems, (64)1, pp. 77-95.

[4] Fossen T.I., (2011). *Handbook of marine craft hydrodynamics and motion control.* John Wiley & Sons Ltd, The Atrium Southern Gate Chichester West Sussex, PO19 8SQ, United Kingdom

[5] Fossen, I. Thor and Johansen, T.A., (2006). *A survey of control allocation methods for ships and underwater vehicles*. In Control and Automation, 14th Mediterranean Conference on IEEE.

[6] R.Skjetne, T.I. Fossen, and P. Kokotovic, (2004). *Robust output maneuvering for a class of nonlinear systems.* Automatica 40 (3), pp. 373-383.

[7] Elkaim, G. H., (2006). *Station keeping and segmented trajectory control of a wind-propelled autonomous catamaran*. 45 IEEE Conference on Decision and Control, 13-15 Dec. 2006.

[8] Sonnenburg, C. R., Gadre, A., Horner, D., Kragelund, S., Stilwell, D. J., and Woolsey, C. A., (2010). *Control-oriented planar motion Modeling of unmanned surface vehicle*. In. Oceans 2010 MTS/IEEE Seattle, pp. .

[9] Sarda, Edoardo I., Qu, Huajin, Bertaska, Ivan R., von Ellenrieder, Karl D., (2016). *Station-keeping control of an unmanned surface vehicle exposed to current and wind disturbances.* Ocean Engineering 127, pp. 305-324.

[10] Ashrafiuon, H., Muske, K. R., McNinch, L. C., Soltan, R. A. (2008). *Sliding-mode tracking control of surface vessels*. IEEE Transactions on Industrial Electronics. 55(11), pp. 4004-4012.

[11] Gomes, P., Silvestre, C., Pascoal, A., and Cunha, R., (2006). *A path following controller for the DELFIMx autonomous surface craft.* Lisboa, Portugal: Institute for Systems and

Robotics.

[12] Sonnenburg, R., Christian, Woolsey, C. A., (2013). *Modeling, identification, and control of an unmanned surface vehicle.* Journal of Field Robotics, 30(3), pp. 371-398.

[13] Jerzy, B., (2012). *Design of robust, nonlinear control system of the ship course angle, in a model following control (MFC) structure based on an input-output linearization.* J. Siencitif Maritime University of Szczecin, 30 (102), pp. 25-29.

[14] Naeem, W., Xu, T., Sutton, R., and Tiano, A., (2008). *The design of a navigation, guidance, and control system for an unmanned surface vehicle for environmental monitoring.* Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment, 222(2), pp. 67-79.

[15] Naeem, W., Xu, T., Sutton, R., and Chudley, J., (2006). *Design of an unmanned surface vehicle for environmental monitoring*. In. World Maritime Technology Conference, London, United Kingdom. pp. 1-6.

[16] Elkaim, G. H., (2009). *System identification based control of an unmanned autonomous wind-propelled catamaran*. Control Engineering Practice, 17 (1), pp. 158-169.

[17] Caccia, M., (2006). *Autonomous Surface Craft: Prototypes and Basic Research Issues*. In: 14th Mediterranean Conference on Control and Automation, 2006. MED '06, Ancona, Italy. 1-6.

[18] Caccia, M., Bibuli, M., Bono, R., and Bruzzone, G., (2008). *Basic Navigation, Guidance, and Control of an Unmanned Surface Vehicle.* Autonomous Robots, pp. 349-365.

[19] Cheng, J., Yi,, J., and Zhao, D., (2007). *Design of a sliding mode controller for trajectory tracking problem of marine vessels*. IET Control Theory and Applications. 1, pp. 233-237.

[20] Wei, M., Chen, G., Yang, L. (2012). *Nonlinear sliding mode formation control for underactuated surface vessels*. In: 10th World Congress on Intelligent Control and Automation (WCICA), 2012 Beijing, China. 1655–1660.

[21] Shr, S. H., Jeng, Y. J., (2011). *Robust nonlinear ship course-keeping control under the influence of high wind and large wave disturbaces*. In. Control Conference (ASCC), 2011 8th Asian, pp. 393-398.

[22] Guerreiro, Bruno J., Silvestre, C., and Cunha, R., (2014). *Nonlinear model predictive control (NMPC) techniques for the problem of trajectory tracking control for autonomous surface craft*. IEEE Transactions on Control Systems Technology, 22(6), pp.

[23] Annamalai, A., Motwani, A., Sutton, R., Yang, C., Sharma, S., Culverhouse, P., (2013). *Integrated navigation and ccontrol system for an uninhabited surface vehicle based on*

interval Kalman filtering and model predictive control. In: IET Conference on Control and Automation 2013. pp,1–6.

[24] Sharma, S. K., Sutton, R., Motwani, A., Annamalai, A. (2013). *Non-linear control algorithms for an unmanned surface vehicle*. Proceedings of the Institution of Mechanical Engineers, Part M. Journal of Engineering for the Maritime Environment 2008. 227(4), pp, 1–10.

[25] Sharma, S., Naeem, W., & Sutton, R. (2012). *An autopilot based on a local control network design for an unmanned surface vehicle*. Journal of Navigation, 65(2), pp. 281-301.

[26] Dong, Z., Wan, L, Li, Y., Liu, T., Zhang, G., (2015). *Trajectory tracking control of underactuated USV based on modified backstepping approach*. Int. J. Nav. Archit. Ocean Eng . 7, pp. 817-832.

[27] Aguiar, A. P., and Hespanha, J. P., (2007). *Trajectory-tracking and path-following of underactuated autonomous vehicles with parametric modeling uncertainty*. IEEE Transactions on Automatic Control, 52, pp. 1362-1379.

[28] Aguiar, A. P., Pascoal, A.M., (2007). *Dynamic positioning and way-point tracking of under-actuated AUVs in the presence of ocean currents.* Int. J. Control, 80 (7), pp. 1092-1108.

[29] Peng, Z.,Wang, D., Wang, W., Liu, L., (2016). *Neural adaptive steering of an unmanned surface vehicle with measurement noise*. Neurocomputing, 186, pp. 228-234.

[30] Ren, J., and Zhang, X., (2013). *Ship course-keeping adaptive fuzzy controller design using command filtering with minimal parameterization*. Control and Decision Conference (CCDC), 2013 25th Chinese. pp. 243-247.

[31] Pan, C. Z., Lai, X. Z., Yang, S. X., and Wu, M., (2007). *An efficient neural network approach to tracking control of an autonomous surface vehicle with unknown dynamics*. J. Expert Systems with Applications. 40, pp. 1629-1635.

[32] Larrazabala, J. Menoyo, Peñas, M., Santos, (2016). *Intelligent rudder control of an unmanned surface vessel.* Expert Systems with Applications, 55, pp. 106-117.

[33] Aguiar, A. P., Dačić, D. B., Hespanha, J. P., and Kokotovic, P., (2004). *Path-following or reference-tracking? An answer relaxing the limits to performance.* IFAC Proceedings 37 (8), pp. 167-172.

[34] Lewis Frank L., (2012). *Optimal control*. John Wiley & Sons, Inc., Hoboken, New Jersey.

[35] Levine, William S., (2011). *Control system advanced methods*. CRC Press, Taylor and Francis Group LLC, 6000 Broken Sound Parkway NW, Suite 300 Boca Raton, FL

33487-2742.

[36] Arun K. Tangirala (2015). *Principles of system identification: Theory and practice.* 6000 Broken Sound Parkway NW, Suite 300 Boca Raton, FL 33487: CRC Press.

[37] Ljung, L. (1994). *Modeling of dynamic systems.* Englewood Cliffs, New Jersey 07632: PTR Prentice Hall.

[38] Jer-Nan Juang (1994). *Applied system identification.* Prentice-Hall PTR, Upper Saddle River, New Jersey 07458.

[39] Eskandarian, A., (2012). *Handbook of intelligent vehicles*. Springer-Verlag London Ltd.

[40] available at: https://www.memsense.com/assets/docs/uploads/nano/DOC00290-NANO-IMU-PSUG-Rev-J.pdf.

[41] Konvalin, C.J., (2008), *Calculating bank, elevation, and heading*, Retrieved from: http://memsense.com/vishal/MTD-0801_1_0_Calculating_Heading_Elevation_Bank_Angle.pdf.

[42] Tilt-sensing with Kionix MEMS acceleromters. Kionix Corp., NY. Retrieved August 14, 2017 from http://kionixfs.kionix.com/en/document/AN005-Tilt-Sensing-with-Kionix-MEMS-Accelerometers.pdf.

[43] Luczak, S., Grepl, R., Bodnicki, M., (2017*). Selection of MEMS accelerometers for tilt measurements*. Hindawi Journal of Sensors, Vol. 2017, Article ID 9796146, 13 pages.

[44] Kort Nozzles. Retrieved from http://www.submarineboat.com/kort_nozzles.htm

[45] Ljung, L. (1999). *System identification: Theory for the user 2nd edition.* PTR, Upper Saddle River, NJ: prentice-Hall.

[46] Karel J. Keesman (2011). *System Identification: An Introduction.* Springer-Verlag London Limited.

[47] Juang, Jer-Nan, Phan, Minh Q., (2004). *Identification and control of mechanical systems*. Cambridge University Press, The Edinburgh Building, Cambridge CB2 2RU, UK.

[48] Jer-Nan Juang, Lucas G. Horta, and Minh Phan (1992*). System Observer Controller Identification Toolbox.* NASA Technical Memorandum 107566, National Aeronautics and Space Administration, Langley Research Center Hampton, Virginia 23665.

[49] Chen, C.W., Huang, J.-K., Phan, M., and Juang, J.-N., (1992). *Integrated system identification and modal state estimation for control of large flexible space structures.* Journal of Guidance, Control, and Dynamics, 15(1), pp. 88-95.

[50] Stevens, Brian L., Lewis, Frank L., Johnson, Eric N., (2016). *Aircraft control and simulations.Ed. 3rd.* John Wiley & Sons, Inc., Hoboken, New Jersey, USA.

[51] Xue, D., Chen, Y., and Atherton, Derek, P., (2009). *Linear feedback control: Analysis and design with MATLAB.* Society for Industrial and Applied Mathematics, 3600 Market St., 6th Floor, Philadelphia, PA 19104-2688.

[52] Corke, P., (2011). *Robotics, vision, and control.* Springer-Verlag Berlin Heidelberg, Germany.

[53] Kenzo Nonami (2013). *Autonomous control system and vehicles: Intelligent unmanned systems.* Springer, Tokyo Japan Limited.

[54] Chen, W., Xiao, H., Zhao, L., and Zhu, M., (2016). *Integrated vehicle dynamics and control.* John Wiley & Sons Singapore Pte. Ltd.

[55] Rolf Isermann (2011). *Identification of dynamic systems: An introduction with applications.* Heidelberger Platz 3 14197 Berlin Germany: Springer-Verlag GmbH.

[56] Ljung, L. (2013). *System identification Toolbox: User's Guid,R2013b.* Matlab & Simulink, The MathWorks, Inc. 3 Apple Hill Drive Natick, MA 01760-2098.

[57] Deb, K., (2001). *Multi-objective optimization using evolutionary algorithm.* John Wiley & Sons, Ltd., Chichester, England.

[58] Bhuvaneswari,M.,C., (2015). *Application of evolutionary algorithms for multi-objective optimization in VLSI and embedded systems.* Springer, India.

[59] Goldberg, David E., (1989). *Genetic algorithms: in search, optimization and machine learning; 1 edition.* Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA.

[60] Li, Y., Liu, J., Wang, Y., (2008). *Design approach of weighting matrices for LQR based on multi-objective evolution algorithm.*Proceedings of the 2008 IEEE - International Conference on Information and Automation, pp. 1188-1192.

[61] Nise, Norman S., (2015). *Control systems engineering; 7 edition.* John Wiley & Sons, Inc. 111 River Street, Hoboken, NJ 07030-5774.

[62] Beale, Mark, H., Hagan, Martin, T., and Demuth, Howard, B., (2017). *Neural network toolbox™: User's guide.* The MathWorks, Inc.3 Apple Hill Drive Natick, MA 01760-2098.

[63] Khan, N., Ullah, I., Al-Grafi, M., (2015). *Dimensional synthesis of mechanical linkages using artificial neural networks and Fourier descriptors.* Mechanical Sciences. 6, pp.29-34.

[64] Chien-Hsun Kuo, and Huang, J.-K. (1998). *System identifcation and control of cavity noise reduction,* (PhD Disseration). Retrieved from Old Dominion University Libraries Norfolk, USA.

[65] Vicario, F., (2014). *OKID as a general approach to linear and bilinear system identification.* (Doctoral dissertation). Retrieved from Columbia University Libraries at URL: https://doi.org/10.7916/D8WS8RVC

[66] Moreira, L., Fossen, T. I., Guedes Soares, C. (2007). *Path following control system for a tanker ship model*. J. Ocean Engineering, 34(14 -15): 2074 -2085.

[67] Belegundu, Ashok, D., Chandrupatla, Tirupathi, R., (2011). *Optimization concepts and application in engineering;* 2 edition. Cambridge University Press,32 Avenue of the Americas, New York, NY 10013-2473, USA.

[68] Karsten Berns (2009). *Autonomous Land Vehicles: Steps Towards Service Robots.* Vieweg+Teubner GWV Fachverlage GmbH Wiesbaden, Germany.

[69] Kenzo Nonami (2010). *Autonomous Flying Robots: Unmanned Aerial Vehicles and Micro Aerial Vehicles.* Springer, Tokyo Japan Limited..

[70] Bibuli, M., Bruzzone, G., and Caccia, M., (2009). *Path-following algorithms and experiments for an unmanned surface vehicle.* Journal of Field Robotics, pp. 669-688.

[71] Encarnacao, P., and Pascoal, A. (2001). *Combined trajectory tracking and path following: An application to the coordinated control of autonomous marine craft.* Proceedings of the IEEE - Conference on Decision and Control, pp.964-969.

[72] Larson, J., Bruch, M., and Ebken, J., (2006). *Autonomous navigation and obstacle avoidance for unmanned surface vehicles.* San Diego, CA: Space and Naval Warfare Systems Center.

[73] Schworer, I., (2005). *Navigation and control of an autonomous vehicle.* Blacksburg, VA: Virginia Polytechnic Institute and State University.

[74] Vanzwieten, T., (2003). *Dynamic simulation and control of an autonomous vehicle.* Boca Raton, FL: Florida Atlantic University.

[75] V., Zoran, B., Bruno, (2001). *Guidance and control systems for marine vehicles. In The Ocean Engineering Handbook, El-Hawary, F., Ed. 1st.* Boca Raton, CRC Press LLC. 25-27.

APPENDIX A

MULTI-OBJECTIVE GENETIC ALGORITHM

A.1 Multi-Objective Optimization and Pareto Optimality Criteria

A multi-objective optimization problem (MOP) refers to a set of optimal trade-off solutions instead of single objective optimization problem; therefore, each solution satisfies all the objective functions to an acceptable level without being dominated by any other solution in the solution space with respect to multiple objectives. Mathematically the multi-objective problem is defined as:

$$\text{Minimize/Maximize} \quad F(x) = [F_1(x), F_2(x), \dots F_k(x)\,] \tag{A1}$$

where $F_1(x), F_2(x), \dots F_k(x)$ are the objective functions. These objective functions are of the minimization and maximization kind. The trade-off solutions are comprised of all the feasible non-dominated solutions which is called the Pareto optimal set. The values of the objective function correspond to the Pareto set form the Pareto front [57], [58]. These of non-dominated points make the Pareto curve which represents a trade-off curve; the curve for two objective functions is shown in Fig. A1 below. From observation of this curve, an appealing point called "knee" denoted by K_n, is to be chosen such that moving away from this point on the curve in any direction would not significantly improve the two objective or cost functions simultaneously. Hence, the knee stands for a point of decreasing marginal utility. If all the objectives are required to be minimized, a feasible solution $\mathbf{x}$ is said to dominate another feasible solution $\mathbf{y}$ if and only $\mathbf{x}$ is as good as $\mathbf{y}$ and better in at least one objective [58], [67] as given by Eqs.A2 and A3 :

$$F_i(\mathbf{x}) \leq F_i(\mathbf{y}), \ \forall\, i \in [1, 2, \dots., K] \tag{A2}$$

and $F_j(\mathbf{x}) < F_j(\mathbf{y}),\ \forall\, j \in [1, 2, \ldots., K]$ (A3)

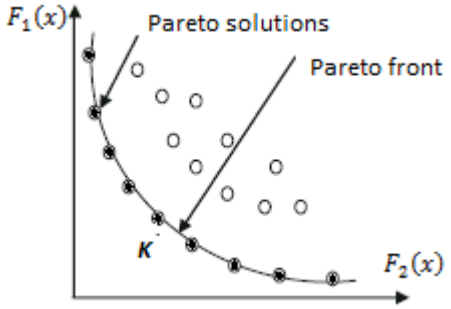

Figure A1. Pareto curve for two objective functions.

A.2 Multi-Objective Genetic Algorithm

Genetic algorithm (GA), being larger class of the evolutionary algorithms (EAs), is the population based promising and robust technique for solving complex and large MOP in every application domain [59]. The main motive of utilization of GA is its efficacy to deal with a large set of possible solutions simultaneously; which results in determining the considerably large members of Pareto optimal set in the single run. GA is basically the stochastic global search method based on natural selection and survival of the fittest in order to have better and better approximations to a solution, which resembles the process of natural evolution. Therefore, multi-objective genetic algorithm (MOGA) is found to be the most suitable and robust methods to solve MOPs. The most vital step in GA based multiobjective optimization technique is the characterization of objective functions. The fitness of GA depends on the objective function for unconstrained optimization problems, so the objective functions, as given in Eq. A1, are used to

evaluate the fitness of an individual. Therefore, the firmness function differentiates between relative good and bad individuals or potential solutions [57].

In order to achieve the best solution for the design variables, GA is always depending on environmental responses and evolution operators such as: *selection*, *crossover*, and *mutation*. In the selection process, the fittest individual from a population of the current generation is selected by comparing its fitness in relation to other individuals to send to the next generation. On the other hand, crossover produces new individuals with new genetic material by exchanging genetic material between individuals of a population which are selected by selection operator. A crossover between two individuals is performed by choosing two crossover points on their chromosomes randomly with the crossover probability p_c and crossed the chromosomes or genetic material between these points. The resulting offspring replace parents in the new population. In the mutation process causes a random modification to the chromosomes of an individual with the mutation probability p_m in order to produce a new individual which may not resemble to the current individual. The mutation probability is fixed before the optimization process. For each individual, a random number between 0 and 1 is calculated and compared with the mutation probability. If the random number is smaller than the probability of mutation, then a gene is mutated so that to avoid a premature convergence and local optima. The process of GA is explained briefly as:

First, the required fitness function ***F(x),*** population size P, crossover probability p_c, and mutation probability p_m are specified. Then, GA initiate with random initialization of the population i.e. P_0 of potential solutions known as individuals c_i which are evaluated upon their fitness without knowing the correct solution for the design variables beforehand. The evolution or genetic operators namely *selection*, *crossover*, and *mutation* make the transition of one population

to the next one in a generation. The fitness of each individual of the population is calculated in the first generation i.e. $F(c_i)$ with $i = 1, \ldots . P$ and assigns a value equivalent to the performance. In each iteration, a new population is generated by applying three genetic operators: i.e. selection, crossover with probability p_c, and mutation with probability p_m to the individuals of population. If the maximum generation $g = g_{max}$ is attained, optimization process is terminated; otherwise, the fitness is calculated and genetic operators are applied. The fittest individual thus obtained represents the solution of optimization problem. The GA process is depicted in Fig. A2.

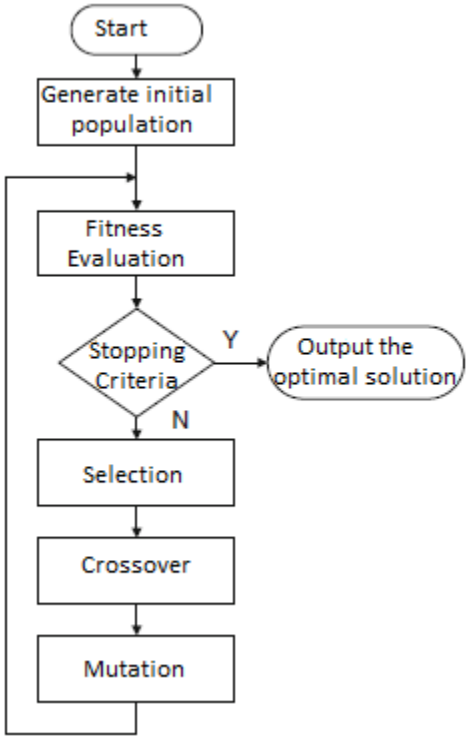

Figure A2. Flowchart of GA.

A.3 Controlled Elitist Genetic Algorithm

In order to construct the Pareto curve and solve the MOP subsequently, an extension of MOGA algorithm available in MATLAB® global optimization toolbox is used; which is called a variant of non-dominated sorting genetic algorithm-II (NSGA-II) called control elitist GA which works on population using genetic operators such as MOGA. In this algorithm, elitism is preserved by keeping of good solutions that is achieved using suitable selection methods thereby increasing convergence. Hence, an individual with better rank or relative fitness value is selected, but the increase in the diversity of population for convergence to an optimal Pareto front is ensured [58]. Multiple Pareto-optimal solutions are determined with the help of this algorithm in one single simulation run. A controlled elitist GA prefers individuals that can assist enhancing the diversity of the population even if they have low rank. The diversity of individuals on the Pareto front is preserved in both aspects i.e. along and lateral to the Pareto-optimal front by controlling the elite members of the population during the progress of algorithm in order to ensure better convergence [57], [58]. This is accomplished by limiting the number of elite individuals on the Pareto front and utilizing the distance metric assist in preserving diversity on the front by preferring individuals that that have larger crowding distance, that is, solution residing in the less-crowded area.

APPENDIX B

NARX NEURAL NETWORK

B.1 Neural Network

The artificial intelligence techniques including artificial neural networks, fuzzy logic, genetic algorithms, and support vector machines are nowadays being widely used for their potential capabilities of solving complex nonlinear modeling, prediction, and mapping problems accurately. These methods, similar to the statistical methods, utilize the past time-series data having the behavior of the dynamic system or process to be modeled. Among the aforementioned methods, neural network (NN) is computer algorithmic architecture which also has the ability to characterize the complex nonlinear relationship between two or more data types that is inputs and target outputs to the neural network. The NN is comprised of many artificial neurons which are simply mathematical representation of the biological neural networks. The number of neurons depends on the task at hand. In general, NNs are an extension of regression, but with one difference that the neural network makes use of one or more hidden layers of neurons, in which the input variables are transformed by a special function, known as transfer function or activation function, into a desired quantity [62]. In order to use any architecture of neural network, the MATLAB® neural network toolbox™ can be used for this purpose.

The NN is simply a network of regression units called neuron piled in a specific configuration. The neuron takes input from the previous layer, combines that input according to learning rule and then applies an activation or transfer function on the result. A feedforward NN configuration is shown in Fig. $B1$ as following.

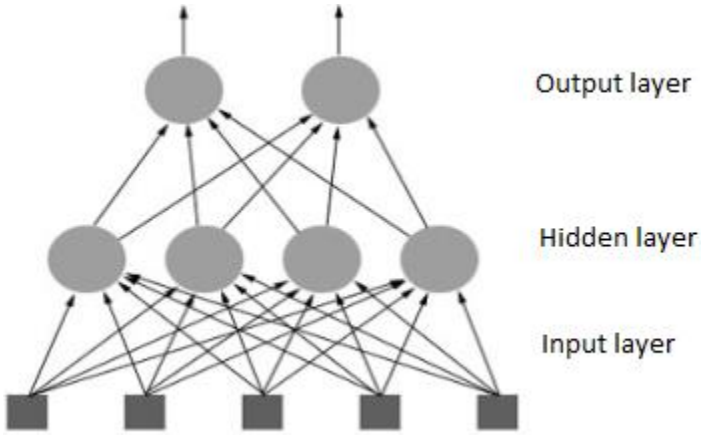

Figure B1. A typical feedforward neural network architecture.

Transfer function or activation function having continuous first derivative bounds the neuron's output within a particular range. It may be may be a linear or non-linear, e.g. a sigmoid function. The selection of a particular transfer function for solving a complex problem by the neuron is accomplished according to the desired behaviour of the output. There are other methods which are used as neurons' activation functions such as step function, threshold logic, binary step function, or hyperbolic tangent function. Instead of using a simple step (threshold) activation function, the one which softens the output of each neuron to produce a symmetrical curve can be used; for example, the *sigmoid* function that makes S-shaped curve. Three of the most commonly used transfer functions are: *tan-sigmoid, linear, and log-sigmoid*.

The data to the network is divided randomly into three different data subsets, namely: 70% training, 15% validation, and 15% testing data sets. The validation data is utilized for measuring network generalization. When the network generalization is not improving for some speci-

fied epochs, then the network training is halted and trained network is evaluated afterwards. In order to check the generalization behavior of the network and see its response to new inputs, it is tested for the new input data that it has never seen before.

B.2 NARX Architecture

The nonlinear autoregressive with eXogenous inputs (NARX) feedback neural network NN architectures is based on the linear ARX model and is given in following Eq. B1.

$$y_{k+1} = f\left(y_k,\ y_{k-1}, \cdots,\ y_{k-n_y+1};\ u_k,\ u_{k-1}, \cdots,\ u_{k-n_u+1}\right) + \epsilon(k) \tag{B1}$$

It is clear from Eq. B1 of NARX network that future value of the dependent output signal i.e. y_{k+1} is regressed on the n_y past values of the output signal y and n_u past values of the independent eXogenous input signal u. Where $k = 0,1,2,\ldots$ and $n_y \geq 1$ and $n_u \geq 1$ subject to $n_y \geq n_u$ represents the memory orders for tapped delay lines of the input and output, respectively. The nonlinear function f in Eq. B1 corresponds to nonlinear mapping which is unknown in advance, and can be approximated by the training of a feed-forward multilayer perceptron with the help of the network weights and biases. The term $\epsilon(k)$ is the approximation error of the signal y at time step k which needs to be minimized during training of the network [62]. So, the NARX is a recurrent dynamic network having feedback connections enveloping many network layers. They have tapped delay lines for both inputs and outputs data set as shown in Fig. B2.

Various network topologies according to complexity level can be chosen by specifying the number of hidden layers and the number of neuron per layer as well as various training algorithms; this sort of selection is optimized through trail-and-error process to have the network topology that can render best performance and avoids over-fitting of the input data to the targets while its generalization and computing power increase instead. The structure of NARX network

comprises of an input layer, hidden layers, and an output layer of neurons which accomplish

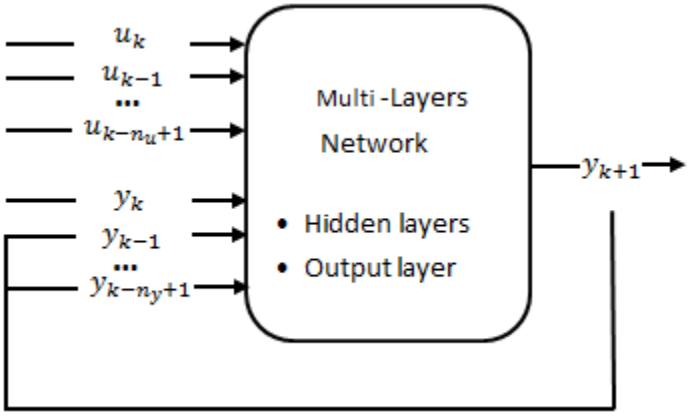

Figure B2. Nonlinear autoregressive with exogenous inputs (NARX) neural network.

separate tasks. The input layer is actually the input fanning which is used just for summing the inputs and biases (if any) to the network and consists of neurons for the eXogenous inputs and a neuron for the output signal as provided to the network. The hidden layers having required neurons take information from the input layer and compare it with the desired targets using supervised learning rule. For this purpose, the activation functions such as *tan-sigmoid* or *log-sigmoid* are used for the neurons of hidden layers; while the linear transfer function is used in the output layer.

B.3 LMBP Algorithm

There are many learning rules for training a network, so the NARX network can be trained using Levenberg-Marquardt backpropagation (LMBP) learning rule. As LMBP is designed to approximate the second order derivative of the network errors without calculating the

Hessian matrix H; it is, most of the time, the fastest backpropagation algorithm [62], [63]. The Jacobian matrix J is utilized for calculation and if the performance function has the form of a sum of squares, then the Hessian matrix can be approximated as:

$$H = J^T J \tag{B.2}$$

and the gradient can be computed as follows

$$g = J^T e \tag{B.3}$$

In Eq. (B.2) and (B.3), the Jacobian matrix J has the first derivatives of the network errors e with respect to the weights and biases in all training samples. To estimate the Jacobian matrix, the standard backpropagation algorithm is used to approximate the Hessian matrix which is simpler method than computing the Hessian matrix. The LMBP algorithm utilizes the Newton-like update as given below:

$$x_{k+1} = x_k - [J^T J + \mu I]^{-1} J^T e \tag{B.4}$$

If the performance function is mean square error *mse* which is the mean squared error between actual output and the desired output of the network or error sum of squares *sse*, then back propagating the network error is utilized to compute the gradient of the performance function with respect to weight and bias variables.

The back-propagation network is basically a gradient descent algorithm, a generalized Widow-Hoff learning rule applied to multilayer feed-forward network with nonlinear differentiable activation function, in which the network weights are adjusted along the negative of gradient of the performance function. It is has been observed that multilayer feed-forward network with back-propagation learning algorithm can approximate any complex sort of function to a required level of accuracy [62].

VITA

Nadeem Khan was born on April 20, 1975 in Peshawar City, Pakistan. He was raised in Peshawar City and received his Bachelor of Science in Engineering in September, 2001 from the University of Engineering and Technology Peshawar, Pakistan. He joined Pakistan Atomic Energy Commission (PAEC), Islamabad as Junior Engineer on April 03, 2002 where he worked until May, 2011 and left the commission as Senior Engineer. In PAEC, he was the officer incharge of precision computer numerical control (CNC) machine shop which includes CNC milling, CNC lathe, and CNC vertical turning center. He was also involved in the managerial tasks of the organization. In the meantime, he continued his studies and received the Master of Science degree in Mechanical Engineering from the University of Engineering and Technology Peshawar, Pakistan in 2011. Afterwards, he got the government funded scholarship for higher studies and traveled to the United States where he joined the department of Mechanical and Aerospace Engineering at Old Dominion University, Norfolk Virginia in Fall-2011. In March, 2018 he successfully defended his dissertation to receive the Doctor of Philosophy degree in Mechanical Engineering.