Cape Peninsula
University of Technology

DESIGN AND IMPLEMENTATION OF LINEAR ROBUST NETWORKED CONTROL SYSTEMS

by

NCEDO SANDISO MKONDWENI

Thesis submitted in fulfilment of the requirements for the degree

Doctor of Technology: Electrical Engineering

in the Faculty of Engineering

at the Cape Peninsula University of Technology

Supervisor: Prof. R. Tzoneva

Bellville
March 2013

DECLARATION

I, Ncedo Sandiso Mkondweni, declare that the contents of this thesis represent my own unaided work, and that the thesis has not previously been submitted for academic examination towards any qualification. Furthermore, it represents my own opinions and not necessarily those of the Cape Peninsula University of Technology.

Signed **Date**

ABSTRACT

Networked Control Systems is a control system where the plant and the controller exchange information via a shared communication network and the network is considered as part of the closed loop control system. Unfortunately the network introduces network induced random varying time delays and data packet loss amongst the communication network imperfections. The network delays are considered to be between the controller and the actuator and between the sensor and the controller. These network imperfections degrade the performance of the closed loop control system and result in closed loop system instability.

The complexity of measuring the communication network imperfection in networked control systems makes it difficult for the control engineers to develop methods for design of controllers that can incorporate and compensate these imperfections in order to improve the performance of the networked control systems.

In this thesis a co-simulation toolset called LabNS2 is developed to address the first problem of measuring the communication network imperfections by providing an ideal environment that can be used to investigate the influence of network time delays or packet loss. The software environment of the toolset is based on LabVIEW™ and Network Simulator Version 2 (NS2).

A new robust predictive optimal controller design method is developed to address the problem of the destabilising effect of the network induced time delay between the controller and the actuator. The design approach is based on time shifting of the optimisation horizon and a state predictor. The design of the controller is based on a model of the plant with delay in the control vector equal to the delay between the controller and the actuator or to the sum of the delays between the controller and the actuator and between the sensor and the controller. The time shifting approach allows the design of the controller to be performed for a model without time delay. Then the control action is based on the future values of the state space vector estimates. The state predictor is developed to predict these future values of the state using the present and past values of the state estimates and control actions. This technique is made possible by the use of the plant model Transition Matrix.

A Discrete Kalman Filter is modified to address the problem of the destabilising effect of the network induced time delay between the sensor and the controller. An additional state estimation vector is added to the filter estimate at every current moment of time.

The developed methods are implemented for networked control of a dish antenna driven by two stepper motors.

The outcomes of the thesis can be used for the education and fundamental research purposes, but the developed control strategies have significant sense towards the Square Kilometer Array projects and satellite systems industry.

ACKNOWLEDGEMENTS

I wish to express my humble and sincerely thanks and acknowledgements to:

- My family, especially my wife for her encouragements and support.
- Prof. Raynitchka Tzoneva, my supervisor for her guidance, persistence and confidence in me throughout the journey of my research.
- Square Kilometre Array South Africa (SKASA) project for creating an opportunity for me to consider continuing my studies.
- Mr C.S. Gumede for his assistance and encouragement.
- Anele for providing accommodation and transportation during my block studies in Cape Town.

The financial assistance of the National Research Foundation towards this research is acknowledged. Opinions expressed in this thesis and the conclusions arrived at, are those of the author, and are not necessarily to be attributed to the National Research Foundation.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

Terms/Acronyms/Abbreviations	Definition/Explanation
ATNF	Australia Telescope National Facility
Az	Azimuth
CAN	Controller Area Network
CSN	Cyclic Service Network
CSMA/BA	Carrier Sense Multiple Access with Bitwise Arbitration
COTS	Commercially Off The Shelve
DEC	Declination
DKF	Discrete Kalman Filter
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
IP	Internet Protocol provides network routing using IP addressing eg 192.168.1.1
NCS	Networked Control System
NITD	Network Induced Time Delay
NS-2	Network Simulator Version 2
OS	Operating System
PID	Proportional, Integral and Differential
PDF	Probability Distribution Function
PP	Pole Placement
RA	Right Ascension
RAN	Random Access Network
SNR	Signal to Noise Ratio
TCL	Tool Command Language
TCP	Transmission Control Protocol - IP with ports to distinguish among processes running on

	same host. Connection oriented, stream transfer, full duplex, reliable with data verification.
TX	Transmit
LTI	Linear Time Invariant
LQG	Linear Quadratic Gaussian
RX	Receive
UDP	User Datagram Protocol - IP with ports to distinguish among processes running on same host. No data verification.
UT	Universal Time
E.M.F	Electromagnetic force
LabVIEW™	Laboratory Virtual Instrumentation Engineering Workbench
RTT	Round-Trip Time
OSI	Open Standard Interface
RTDNCS	Real Time Distributed Networked Control System
ADC	Analogue to Digital Conversion
DW	Delay Window
VI	Virtual Instrument
MATLAB	Matrix Laboratory
Single baseline	A single baseline is a distance between two antennas.
Interferometer	Instrument used to interfere waves.
Zenith	The great circle that cuts through the north celestial pole
Nadir	The great circle the cuts through the south celestial pole
LGQ	Linear Quadratic Gaussian
MIMO	Multiple Input Multiple Output

DRE	**Difference Riccati Equation**
i.i.d	**Independent and Identically distributed**

MATHEMATICAL NOTATION

Symbols/Letters	Definition/Explanation
W_{tcp}	Average TCP window size (packets)
$R(t)$	Round Trip Time (RTT)
q	Average queue length (packets)
C	Channel bandwidth (packets/s)
N_{tcp}	Loading factor (that is number of TCP sessions)
$K(k)$	Kalman filter gain
$P(k)$	Error covariance
$\hat{x}(k)$	Estimated state space vector
θ_k^{sc}	Packet loss between the sensor and the controller
Θ_k^{ca}	Packet loss between the controller and the actuator
BW	Primary beam width
d	Dish diameter
λ_a	Wavelength of the observed frequency
N	Number of antennas
D	Baseline
N	Number of antennas
H	Object's hour angle
Λ	Object's right ascension
δ	Object's declination
φ	Object's latitude
Π	Object's longitude
Az	Object's azimuth
alt	Object's altitude
V	Input Voltage

K_t	**Motor toque constant**
R	**Electrical resistance**
L	**Electric inductance**
i	**Armature current**
e_a	**Back EMF of armature**
K_b	**Electromotive force constant**
ω_m	**Angular velocity of the motor**
θ_m	**Angular position of the motor**
T_L	**Reflected load toque**
T_w	**Toque of the wind**
T_m	**Motor toque**
J_m	**Motor moment of inertia**
J_L	**Load moment of inertia**
J_T	**Total moment of inertia of the motor and load.**
B_m	**Motor damping coefficient**
B_L	**Load damping coefficient**
B_T	**Total damping coefficient of the motor and load.**
N_m	**Number of teeth of the motor gear**
N_L	**Number of teeth of the load gear**
$N_g = \frac{N_m}{N_L}$	**Gear ratio**
c_t	**Dimensionless torque coefficient**
α_p	**Static air density**
k_T	**Quadratic law coefficient**
υ_w	**Steady state wind speed**
s	**Number of steps the rotor performs per revolution**
p	**Number of pole pairs**

m	Number of the stator phases
$\Delta\theta_m$	Stepping angle
τ^{ca}	Network delay between the controller and the actuator
τ^{sc}	Network delay between the sensor and the controller
k_p	Proportional constant
k_i	Integral constant
k_d	Derivative constant
M	Ultimate gain
Pu	Ultimate period
a	Amplitude
Ts	Settling Time
ς	Damping Ratio
ω_n	Undamped natural frequency
ω_d	Desired undamped natural frequency
a	Third pole
τ_c	Critical delay
$A \in R^{nxm}$	Constant state matrix
$B \in R^{nxm}$	Constant control input matrix
$C \in R^{lxn}$	Constant measurement matrix
w	Process noise
v	Measurement noise
Σ	Process noise covariance
Q	Measurement noise covariance
$\hat{x}^-$	Priory state estimate
e^-	Posteriory estimate error
P^-	Priory estimate error covariance

P	**Posteriory estimate error covariance**
K	**Kalman gain**
L	**State space controller gain**
L_g	**Matrix for disturbance compensation**
L_w	**Matrix for wind compensation**
g_w	**Vector solution of the Riccatti**
$x \in R^n$	**State space vector**
$u \in R^m$	**Control vector**
$y \in R^l$	**Output vector**
$y_d \in R^l$	**Desired output vector**
$\lambda \in R^n$	**Co-state variable**
W	**Wind constant matrice**
$\varphi_u(k)$	**Initial function for the time delay period**
$\psi(k)$	**State transition matrix**

CHAPTER ONE
STATEMENT OF THE PROBLEM: PROBLEM OBJECTIVES HYPOTHESIS AND ASSUMPTIONS

1.1 Introduction

Networked Control Systems also known as Integrated Communication and Control Systems (Halevi and Ray, 1988) are control systems where the plant and the controller exchange information via a shared communication network and the network is considered as part of the closed loop control system. Unfortunately the network introduces network induced random varying time delays and data packet loss amongst the network imperfections. These delays and packet loss affect the dynamic performance of the closed loop systems and may be the cause of system instability. This chapter presents the awareness of the problem, problem statement, objectives of the research, hypothesis, delimitations of the research, assumptions, deliverables and thesis chapter breakdown.

1.2 Awareness of the problem

"A problem well defined is the problem half solved"; it is this statement that makes it necessary to define clearly the areas that are directly affected by the problem considered in this thesis.

Networked control system overlaps between two areas of study; Information Systems and Feedback Control System as shown in Figure (1.1). Information Systems Networks have been investigated for decades from the computer science point of view and their fundamental function is to transfer information from one point to another with no concerns about the:

- confirmation that the information has arrived
- the time at which information has arrived, and
- feedback from the recipient confirming receipt of that information.

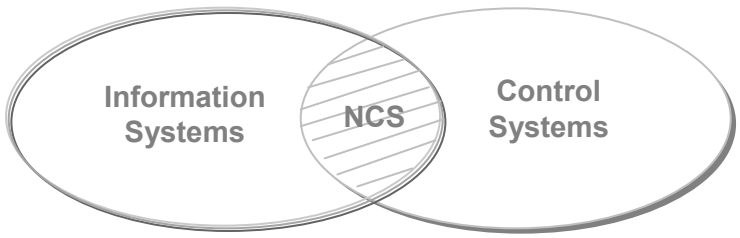


Figure 1.1: NCS areas of study

In a feedback control systems confirmation that the information has arrived, the time at which the information has arrived, and the feedback from the recipient confirming the

receipt of the information are critical, because in a feedback control system the plant sends back a feedback signal and the feedback signal is compared with the desired set point resulting in an error that is sent to the controller and the controller sends back the control action to the plant. This loop runs until the desired set point is reached, and in order to be implemented the sent data have to arrive in time.

Control systems have existed for the past decades; applications are for the chemical process plants, refiners, power plants, airplanes, etc. Feedback Control Systems is one of the popular areas of Control Systems. These systems are partially distributed and may be operated in an asynchronous way, but have their operations organized to achieve a desired overall goal. The selection of appropriate technology to gather information from these partially distributed systems is imperative as it has cost impact on the overall control system design. Real-time distributed networked control system (RTDNCS) is a control system that comprises of the process to be controlled, actuators, sensors and controllers that are all connected in a real-time distributed network. (Baillieul and Antsaklis, 2007).

Feedback Control System is mostly considered for cases where the controller and the plant are close to one another and there were no network imperfections, see Figure (1.2).

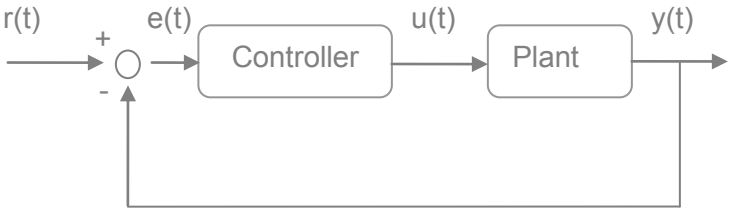


Figure 1.2: Typical closed loop control system without network imperfections
r(t) : Reference, e(t): error, u(t): plant input, and y(t): plant output

In a network control system the controller and plant are distributed via the network and the network is considered explicitly as part of the closed loop feedback control system as shown in Figure (1.3).

Figure 1.3: Typical Network Control System architecture
In this architecture the network is considered explicitly as part of the control system, where S = Sensor, A = Actuator, Enc = Encoder, Dec = Decoder

It must be clear that the pursuit to use low cost processing at remote location via embedded technologies has long been an ideal for networked control system as the information can be transmitted reliably via shared digital networks or wireless connections. This however, had strong implications on the control system as it affects the control system explicitly (Baillieul and Antsaklis, 2007). The networked control system is very important in industry as the control system may improve the production of the process being controlled by operating the plants effectively and efficiently. These networked control systems are challenging theoretically and technically as their design is based on different research areas including but not limited to information theory, control system, communication networks, embedded technologies and computer science.

The latter clearly shows that the networked control system encompasses all the research areas highlighted above and to merge these areas whilst considering all the constraints and limitations associated with them posses a big challenge. These different research areas encompassed in this research topic pose a major risk to the overall project as they can easily derail the research proposed in one of the researchable areas. As a result it is imperative that the focus and emphasis of this research topic is defined clearly. The emphasis is hence summarised:

The research investigations in this thesis are looking at the solution of the following research questions:

- What are the performance limitation and effects of the communication network on the linear or non-linear NCS. These limitations include:
    a) Network induced time delays and jitter.
    b) Accuracy of the transmitted and received information over the network.
    c) Temporally disorder on the communication packets sent and received.
    d) Bandwidth limitations.
    e) Protocol used for communication, e.g. Ethernet
    f) Timing and synchronisation of networked devices, these include computers, embedded hardware that are nodes in the network, sensors.
    g) Limited quantization of sensors.
    h) Limited computational power of the controller.
    i) Network reliability, consistence and availability.

- How to overcome the influence the communication networks have on the behaviour of the linear or non-linear NCS. This implies investigating what has to be changed in the methods for optimal control design in order to overcome the limitation introduced by the communication network.

.

- What is the convenient control technology for simulation, and implementation of a real time distributed networked control system.

The thesis gives answers to some of the above research questions by development of methods for design of a time delay predictive controllers and Kalman filters and development of special software and hardware environment where simulation and implementation of network control systems can be done.

1.3 Statement of the problem

The main problem is to investigate the process of merging the mentioned above different researchable fields encompassed in the NCS and to investigate their impact over the development of new theoretical methods for design of control and their software and hardware implementation. The basic structure of a networked control system is shown in Figure (1.3).

Two types of sub-problems have to be solved in each of above mentioned areas;

- Design based sub-problems, and
- real time implementation sub-problems

1.3.1 Design based sub-problems

Sub-problem 1: Analysis and design of a NCS of a complex industrial system without paying attention to the network drawback influence.

Design sub-problem 2: Simulation and determination of the characteristics of the NCS of a complex industrial system using the controller designed in point 1.

Design sub-problem 3: Determination of the maximum values of the network drawbacks causing instability to the closed loop system.

Design sub-problem 4: Development of a joint network and process model incorporating dynamics of the process and of the communication network.

Design sub-problem 5: Investigate the limitations introduced by the communication network using simulations of the joint process model.

Design sub-problem 6: Development of new methods for linear NCS optimisation and for design of a linear robust towards the network influence controller.

Design sub-problem 7: Investigation of the dynamic characteristics of the closed loop robust NCS using LabVIEW^{TM} and Matlab software environment.

1.3.2 Real time implementation sub-problems

Real-time sub-problem 1: Implementation of the developed NCS model in the framework of distributed system for control of different plants based on PC for emulation and control, wired communication network, Matlab/Simulink and LabVIEW software environment.

Design sub-problem 2: Develop simulator for the NCS using Matlab and Simulink.

1.4 The objectives of the research project

The aim of the project is to merge the different researchable fields encompassed in the NCS and to develop new theoretical methods and software to implement the NCS. The implementation is done for an example of a dish antenna control using PC with Matlab/Simulink and LabVIEW software. Commercial Off The Shelve hardware and designed and built hardware are investigated and used for the implementation.

The following are the main objectives of the research:

a) Comparative analysis of the methods for analysis and design of robust NCS.
b) Characterize NCS of a complex industrial system from control point of view.
c) Development of a dynamic model of the NCS incorporating the process and the network models.
d) Investigate the limitations introduced by the communication network over the control system performance.
e) Develop new methods for NCS optimal and robust controller design based on a Kalman filter.
f) Develop simulator for investigation of the designed NCS using Matlab, Simulink and LabVIEW.
g) Implement the developed NCS in the framework of distributed control systems on the basis of a PC for emulation and control implementation.

1.5 Hypothesis

The hypothesis is based on the provided critical review of the theory and application of the NCS.

1. The effects of the limitations of the communication network over the performance of the NCS can be reduced through new approach to the modeling and design of the control of these systems. The limitations can be represented by a dynamic model additional to the process model.
2. The design of the optimal control of the NCS based on the augmented or joint model is capable to overcome the influence of the network limitations over the whole system performance.
3. The open design of the hardware and software of the lab-scale NCS allows further future extension according to the types of the monitored and control processes, types of the networks (introduction of wireless one) and use of more types of controllers.

1.6 Delimitation of the research

The research project emphasizes on the merging of the different researchable fields that constitute the design and implementation of the NCS and avoids being derailed to only one of the researchable fields.

The first approach in developing the networked control system is to start with linear time invariant system (LTI) and then proceed to the multivariable linear or non-linear systems.

The real-time emulation technology hardware implementation is chosen based on the outcomes of the simulation of the designed NCS.

The network limitation will be based on the outcomes as observed from the control point of view. The latter is based on each research field having different outcomes when it comes to network performance and requirements (Lian et al, 2007).

The project is realized by using standard PC. The standard PC performs the following functions:

1. State estimation algorithms
2. Optimization algorithms.
3. Controller design.
4. Logic function thus allocating the results to the designated hardware that will later execute the control action.
5. Reconfiguration of the model and controller type.
6. Changing set point and controller parameters in real time on the designated hardware that will later execute the control action in real-time.

The controllers proposed in this thesis are:

1. PID,
2. Robust predictive Linear state space

The state estimation problem is solved by the developed Kalman Filter:

1.7 Assumptions

The research is conducted based on these communication network assumptions:

1. Network transmissions are error free.
2. Every frame or packet always has the same constant length.
3. The difference between the sampling times of the controller and of the sensor is constant.
4. The computational delay is constant and is much smaller than the sampling period.
5. The network traffic cannot be overloaded.
6. Every dimension of the output measurement or the control signal can be packed into one single frame or packet.
7. The communication network is wired and can be extended to use wireless communication network.
8. The communication network is reliable, visible and can easily be diagnosed.
9. The communication network is based on Ethernet. This includes network nodes at a sensor level.

10. The data rates are based on the minimal data rates for the selected network.
11. The drawbacks of the communication channel are presented by two variable delays - between the controller and the actuator and between the sensor and the controller.

The research is conducted based on these control methods assumptions:

1. Sensor and controller signals are transmitted over the network.
2. Stochastic control system can be a good description of the NCS.
3. Multivariable NCS are considered.
4. The NCS is distributed in functionality and time. The communication time delays can be used in a process for design of robust predictive controllers.

1.8 The deliverables of the project are:

- Mathematical model derivations for the plant.
  - Development of the plant model based on the servo motor.
  - Development of the plant model based on the Permanent Magnet (PM) stepper motor.
- Investigation of the influence of the network induced time delays for the case of standard controllers using simulations.
- Derivation of the Discrete Kalman Filter.
- Development of a method for design of a robust predictive controller.
- Investigation of the influence of the network induced delays for the case of the Kalman filter and the predictive controller.
- Development of simulation and emulation environment.
- Real time implementation using the trainer antenna based on DMX-ETH-17 integrated stepper motor.

1.9 Chapter Breakdown

This thesis has ten chapters. The chapters include problem statement, problem analysis, model development, controller design, system simulation, system implementation, results of the project and conclusion.

Chapter 1 presents the problem statement and gives the background of the project and analyses the different researchable areas in which the research investigations are done.

The project statements include: the research aim and objectives, statement of the problem, hypothesis, research delimitation, research motivations, and assumptions.

Chapter 2 presents the review of papers that are related to networked control systems. It pays more attention to papers considering Network types and Communication Protocols, Models of time delay systems, Discrete Kalman Filter for systems with time delays, Models of packet loss, Methods for design of robust controllers, Simulators and Emulators.

Chapter 3 presents the plant considered in the thesis. The background of Radio Telescope, comparison between Single Dish Radio Telescope and Interferometer, antenna control structure, antenna pointing accuracy, coordinate systems and simulation of horizontal coordinate transformation system are presented in this chapter.

Chapter 4 describes the derivation of detailed mathematical models of the plant including motors for a dish antenna where two motors are considered to represent one antenna. The models of the servo motor and the stepper motor that could be used to drive the antenna are described in detail first. Based on the understanding of the servo motor and stepper motor systems physical structure, the nonlinear and linear mathematical models are derived. The models are simulated in Matlab/Simulink.

Chapter 5 describes the controller design in continuous time domain. The closed loop control system design is considered without paying attention to network imperfections. The aim of the design and simulation is to determine the maximum network induced time delay that causes the closed loop control system to become unstable. This maximum value is used as threshold and based on this threshold a robust controller is designed in Chapter 7 to overcome the network induced delay limitations.

Chapter 6 describes the derivation of detailed Discrete Kalman Filter (DKF). Based on the understanding of the DKF a Predictive Discrete Kalman Filter is designed incorporating the time delay between the sensor and the controller in the measurement output and defining the estimated state variable.

Chapter 7 describes the development of stochastic Linear Quadratic controller for a Networked Control System. Three cases are considered:

- Case 1: Design of an optimal controller for the plant without deterministic disturbance.
- Case 2: Design of an optimal controller for the plant with deterministic wind disturbance
- Case 3: Design of an optimal robust predictive controller for the augmented model of the plant by incorporation of the network delays in the plant model control vector.

Chapter 8 presents investigation of different cases of the Network Control System by simulation in the Matlab software environment.
The following cases of design of the controller and the Discrete Kalman Filter (DKF) are considered:

- Controller and the DKF are designed for a NCS without delay. The delays between controller and actuator and between the sensor and controller are introduced and their impact over the closed loop system behaviour is investigated using simulations.
- Control augmentation approach. In this approach the time delay between the controller and the actuator is incorporated in the control vector of the plant. The plant is considered as a system with time delay during the design of the controller. A predictive controller is designed in State Space form for the augmented plant. The DKF is designed for a NCS without delay between the sensor and actuator. The influence of the delays between the sensor and the controller and between the controller and actuator is investigated by simulations.
- Plant output augmentation approach. The delay between the sensor and the controller is incorporated also in the plant control vector. The plant is considered as a system with time delay during design of the predictive controller. The influence of the Network induced delays is investigated by simulations.

Chapter 9 is split into two parts. Part I describes the architecture of the toolset developed in the thesis and used to emulate the developed Networked Control System. The toolset is called LabNS2. The name is derived from LabVIEW^{TM} and Network Simulator version 2 (NS-2) which are the main software environments used in developing the toolset. The toolset uses Windows environment as an operating system. NS-2 is developed for UNIX environment. In order to emulate UNIX environment within the Windows environment, Cygwin is used. Cygwin is the UNIX emulator for Windows environment.

Part II presents the implementation of the developed methods in real time using DMX-ETH-17. The DMX-ETH-17 is an integrated Stepper Motor, with Encoder, Driver, Controller and Ethernet communication capabilities developed by Arcus Techonology. Two of the DMX-ETH-17 stepper motors are used to represent Azimuth and Altitude drive which makes one dish antenna. A trainer antenna model is designed using these stepper motors and to illustrate the functionality of the NCS.

Chapter 10 presents the conclusion, and the deliverables of the thesis. The application of the thesis results, the future work, and the publications.

1.10 Conclusion

This chapter presents the problem statement and give the background of the project and analyses the different researchable areas in which the research investigations are done. The project statement includes: the research aim and objectives, statement of the problem, hypothesis, research delimitation, research motivations, and assumptions
In order to understand the depth of the problem it is critical to review literature specifically looking at the papers that investigate the same or similar problem in the different research areas and papers that are looking at NCS in general.

Chapter 2 outlines the existing literature on Networked Control System considering most of the sub-problems that constitute the NCS.

CHAPTER TWO
REVIEW OF PAPERS ON NETWORK CONTROL SYSTEMS

2.1. Introduction

This chapter reviews existing papers on networked control systems. Traditional control system considers only actuators, sensors and controller as part of the closed loop (Baillieul and Antsaklis, 2007), see Figure (2.1) where A is the actuator, S is the sensor, Enc is the encoder, and Dec is the decoder. The controller and the plant are close to each other and because of this fact, timing and other constraints are negligible and hence not considered as part of the closed loop control system. Networked Control System is defined as closed loop control system where the network is considered explicitly as part of the closed loop control system design. Figure (2.2) shows a Networked Control System (NCS).

Figure 2.1: Traditional Control System

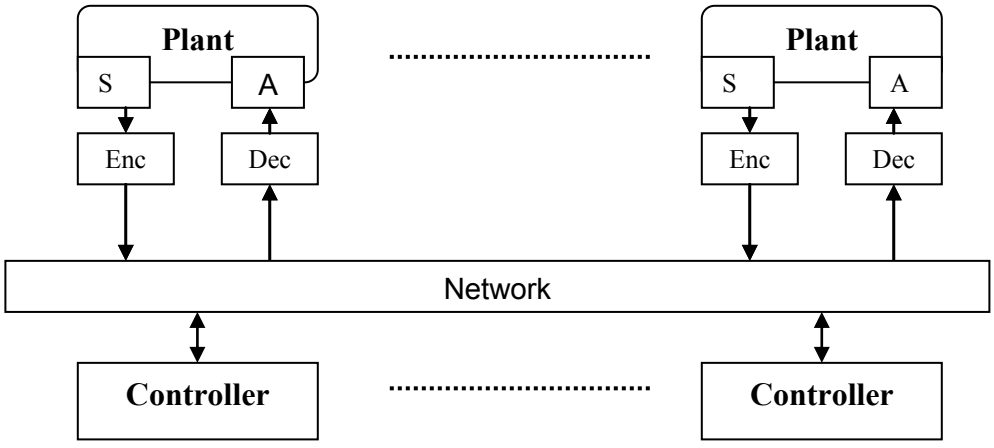


Figure 2.2: General Network Control System.

The fundamental challenge with NCS is that the network component comes with its own imperfections, as a result these imperfections affects the performance and stability of the

closed loop control system (Branicky et al, 2000), (Ye-Guo and Shi-Yin, 2011). These imperfections are:

- Network induces time delays and jitter.
- Temporally disorder and loss of the packets sent and received.
- Accuracy of transmitted and received information over the network reduction.
- Channel bandwidth limitations.
- Network protocol, e.g. TCP, UDP, CAN or Wi-Fi influence
- Timing and synchronisation of networked devices, these include computers, embedded hardware that are nodes in the network, sensors.
- Limited quantization of sensors.
- Limited computational power.
- Network reliability, consistence and availability.

These imperfections within the Ethernet network were never an issue a couple of years ago because Ethernet was developed for information systems and not for control systems because feedback information was never an issue (www.tcpipguide.com). In feedback control theory feedback information from the control nodes is very crucial for the proper behaviour of the process under control.

Section 2.2 of this chapter considers the effects of network type and communication protocols on the closed loop control system's performance. In this section the different types of networks and the effect they have on the NCS are described. Different types of communication protocols are also described. Ethernet is chosen as the preferred protocol for development of the investigations in the thesis. Section 2.4 reviews the Discrete Kalman Filter for systems with time delays as used in NCS. Section 2.3 to section 2.6 review the effects of network induced time delays, data rates, bandwidth, quantisation and packet loss on the NCS. Section 2.7 reviews control methods for design of controllers in the NCS. Section 2.8 reviews the papers on Simulators and Emulators of NCS.

2.2. Network types and communication protocols

There are two major types of communication networks and they both affect the performance and stability of the closed loop control system. They are Cyclic Service Network (CSN), (Tipsuwan and Chow, 2003), that is deterministic, and Random Access Network (RAN), that is non-deterministic.

Typical examples of CSN industrial protocols are Profibus, Foundation Fieldbus, Controller Area Network (CAN), Device-Net, PROFINET, and EtherCAT. The latter two protocols became popular in industrial applications that required reliability of communication networks. These industrial protocols are robust and reliable for real-time control system; however they are proprietary and are not commercially available off the-shelve. There are also doubts in terms of their performance. Prytz (2008) conducted a performance analysis between PROFINET IRT which is part of PROFINET protocol suite developed by Siemens and EtherCAT which was developed by Beckhoff in order to identify which of the two protocols give a better performance. EtherCAT came out to be a better protocol than PROFINET. German companies (Tipsuwan and Chow, 2003) developed these industrial protocols for use in car industry in the 1980's.
Typical example of RAN network is the Ethernet which evolved from Slooted ALOHA and ARPA-NET. These protocols were developed about 30-40 years ago (Tipsuwan and Chow, 2003). Ethernet is the considered network type in the thesis.

According to the Open Standard Interface (OSI) model the communication protocols are defined by three of the OSI layers (www.tcpipguide.com) being; Data Link, Network, and Transport layers as shown in Table 2.1.

Table 2.1: OSI Model

Layer Number	OSI Model	Protocol
7	Application Layer	
6	Presentation Layer	
5	Session Layer	
4	**Transport Layer**	**UDP, TCP**
3	**Network Layer**	**IP**
2	**Data Link Layer**	**Ethernet**
1	Physical Layer	

Ethernet is a layer 2 protocol that runs on top of a layer 3 protocols called Internet Protocol (IP). IP is a best effort protocol because it is connectionless and there is no guarantee of packet delivery (www.tcpipguide.com). The IP packages the datagram and sends it without any verification if connection exists and the recipient is available. One of the fundamental challenges of IP is the fact that there is no acknowledgement from the recipient on whether the packet has been received or not. In control system the feedback

is crucial for feedback control system. The latter causes a fundamental problem as the non-deterministic nature of the IP leads to network induced time delays which ultimately result in packet loss (Chow and Tipsuwan, 2003). To try and compensate for this IP limitation Transmission Control Protocol (TCP) was developed to provide network layer connectivity, connection establishment, flow control and reliability (http://ipv6.com/articles/general/User-Datagram-Protocol.htm). See Figure (2.3), for the TCP protocol format.

<table>
<tr><td colspan="3">Source Port (16bits)</td><td>Destination Port (16bits)</td></tr>
<tr><td colspan="4">Sequence Number (32bits)</td></tr>
<tr><td colspan="4">Acknowledgment Number (32bits)</td></tr>
<tr><td>Header Length (4bits)</td><td>Reserved (6bits)</td><td>Flag (6bits)</td><td>Window (16bits)</td></tr>
<tr><td colspan="3">Checksum (16bits)</td><td>Urgent Pointer (16bits)</td></tr>
<tr><td colspan="4">Options (0 or 32bits)</td></tr>
<tr><td colspan="4">Data (Variable)</td></tr>
</table>

Figure 2.3: TCP Segment Format

Later it was realised that the reliability function brought overheads and compromised the performance (Antoniou S., 2007). UDP was developed as a solution with less overhead as shown in Figure (2.4).

<table>
<tr><td>Source Port (16)</td><td>Destination Port (16)</td></tr>
<tr><td>Length (16)</td><td>Checksum (16)</td></tr>
<tr><td colspan="2">Data (Variable)</td></tr>
</table>

Figure 2.4: UDP Segment Format

Table 2.2 shows the comparison between TCP and UDP protocols characteristics.

Table 2.2: **TCP and UDP Characteristics (Antoniou S., 2007).**

TCP	UDP
Reliable	Unreliable
Connection-oriented	Connectionless
Segment retransmission and flow control through windowing	No windowing or retransmission
Segment sequencing	No sequencing
Acknowledge segments	No acknowledgment

The challenge when considering the TCP and UDP protocols as part of the control system is to model the TCP and UDP latency and throughput based on the segmentation formats and the characteristics of the two protocols. The first step in understanding the constraints of TCP and UDP protocols was to develop mathematical models that describe the protocols and later incorporate the models into the closed loop control system. Padhye et al (1998) proposed a steady-state throughput model as a function of loss rate and round trip time for a bulk transfer of TCP flow that is flow with unlimited amount of data to send. Anderson et al (2000) extended the model of (Padhye J. et al, 1998) by capturing start up effects. They described connection establishment and data transfer latency as a function of packet loss conditions and no loss conditions.

The communication protocol has been one of the main challenges when it comes to industrial application of Network Control System. Control engineers used an approach of designing the controller without regards for the delays induced by the network, then design industrial communication protocol that minimises or represents the delay as a constant (Branicky et al, 2000). Rostan (2008) in a review on industrial Ethernet technologies has classified the industrial protocols into three classes being Class A, B and C. The classes are defined as;

Class A:

- Completely TCP/UDP/IP based and
- Ordinary Ethernet controllers, switches and routers are used.

Class B:

- Process Data: Parallel channel to TCP/UDP/IP,
- TCP/UDP/IP timing controllers by process data driver, and
- Ordinary Ethernet controllers, switches and routers.

Class C:

- Process Data: Parallel channel to TCP/UDP/IP
- TCP/UDP/IP timing controllers by process data driver, and

- Special real-time Ethernet controllers or switches and routers.

The limitations of Class B and C industrial protocols are; they are proprietary to their vendors and some need customised hardware for them to function. Typical example of these protocols is the CAN which is based on Carrier Sense Multiple Access with Bitwise Arbitration (CSMA/BA) protocol used to decrease the delay (Rostan, 2008).

Class A is the considered class in the thesis because of the advantages it has that are: low costs, high speed, friendly usage, established infrastructure and commercial off the shelve software and hardware. Ethernet attracted a lot of control engineers and this type of network became the biggest competition to the industrial protocols (Otanez et al, 2007) and (Tipsuwan and Chow, 2003). The latter resulted in investigation of constraints associated with the usage of Ethernet for control application and considering Ethernet protocol explicitly within the control system (Lian et al, 2007). In the thesis Ethernet network is considered as part of the control loop and a controller design method is proposed that reduces the drawbacks introduced by network induced time delays.

2.3. Models of time delay in the networked control systems

Time delays and the impact they have on performance and stability of closed loop control systems have been investigated for decades (Kurzweil, 1964), (Richard, 2003), (Sipahi et al, 2010). The reasons that make the subject of interest to researches is because time delays systems are applicable in many field e.g. manufacturing systems, biology, physics, operational research, economics and engineering to mention the few. Time delay systems are also called systems with after-effects or dead-time, hereditary systems, equations with deviating argument or differential-difference equations (Sipahi et al, 2010), (Richard, 2003). In some systems the time delays may cause the system to become unstable whilst in other systems the delays are the prerequisite for the stability of the system (Gao et al, 2008), (Peng et al, 2013), (Donkers et al, 2012).

Networked control system is closely related to time-delay systems because of network induced time delays. This phenomenon happens when the plant, controller, and sensors are connected via a shared communication network in what is referred to as Networked Control Systems as shown in Figure (2.2). Almost fifty years ago Kurzweil (1964) refers to the similar effect caused by network induced time delays considered as pure transport delays which at that time were as a result of either long transmission line effects or distributed effects in the process of communication.

The general trend of representing the time delays within a dynamic model is to consider the delay in series with the dynamic model as shown in Figure (2.5) (Lam et al, 2007), (Gao et al, 2008), (Zhang and Yu, 2008) (Kurzweil, 1964).

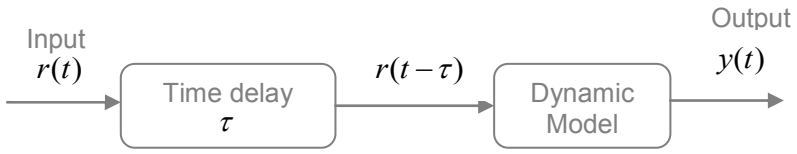


Figure 2.5: Dynamic model with time delay

Existing literature has shown that the stability and performance of networked control systems is affected by time varying network induced delays (Lam et al, 2007), (Gao et al, 2008), (Sadjadi, 2003), (Branicky et al, 2000), (Jia et al, 2010), (Schenato et al, 2007) (Tipsuwan and Chow, 2003).

In the networked control system the delays are generally represented as a time delay between the controller and actuator (τ^{sc}) and a time delay between the sensor and controller (τ^{sc}). The two delays are mostly assumed to be either constant (Lian, 2002), (Nilsson, 1998) appearing in a singular or a simple form or time varying (Tipsuwan and Chow, 2003), (Lam et al, 2006). In the real environment the case of constant delays does not exists as the delays vary depending on the network dynamics.

In some cases probabilistic Markov chain and Independent Identical Distribution (iid) Bernoulli process are used to represent the two time varying delays existing between the controller and actuator and between the sensor and controller (Imer et al, 2005), (Nilsson, 1998), (Moayedi et al, 2010), (Li and Sun, 2012).

Instead of representing the delays in a singular form, Nillson (1998) modelled the delays as independent random delays and used an empirical approach to determine the model for the network induced delays using CAN bus as a protocol. Based on the experiment a Probability Distribution Function (PDF) governed by the underlying Markov chain is defined and τ^{ca} and τ^{sc} are derived and given by Equations (2.1) and (2.2), Figure (2.6).

$$f_{\tau}\left(\tau_k^{sc}\right)=\begin{cases}\delta\left(\tau_k^{sc}-a\right)\left(1-p_{sc}\right), & \tau_k^{sc}=a,\\ p_{sc}/\left(b-a\right), & \tau_k^{sc}\in(a,b), a<b,\\ 0, & \tau_k^{sc}\notin[a,b],\end{cases} \tag{2.1}$$

$$f_{\tau}\left(\tau_k^{ca}\right)=\delta\left(\tau_k^{ca}-a\right).\left(1-p_{ca}\right)+\delta\left(\tau_k^{ca}-b\right).p_{ca} \tag{2.2}$$

where $f_\tau(\tau_k^{ca})$ and $f_\tau(\tau_k^{sc})$ are the distribution function for the independent time varying delays, $\delta(.)$ is the Direc Delta function, ***a*** and ***b*** are constants and $p_{sc}, p_{ca} \in [0,1]$ are parameters of the network.
The equations are implemented in Matlab (Appendix (2.1))for simulation Figure (2.6) and the results are shown in Figure (2.7) below.

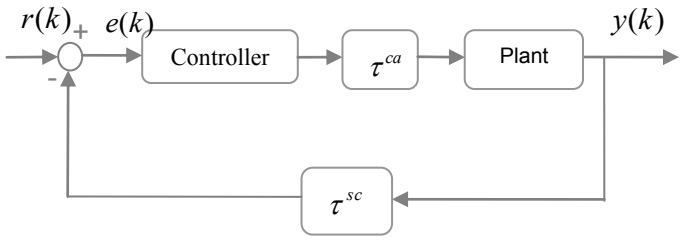


Figure 2.6: **Network Control Systems independent delays τ^{ca} and τ^{sc}**

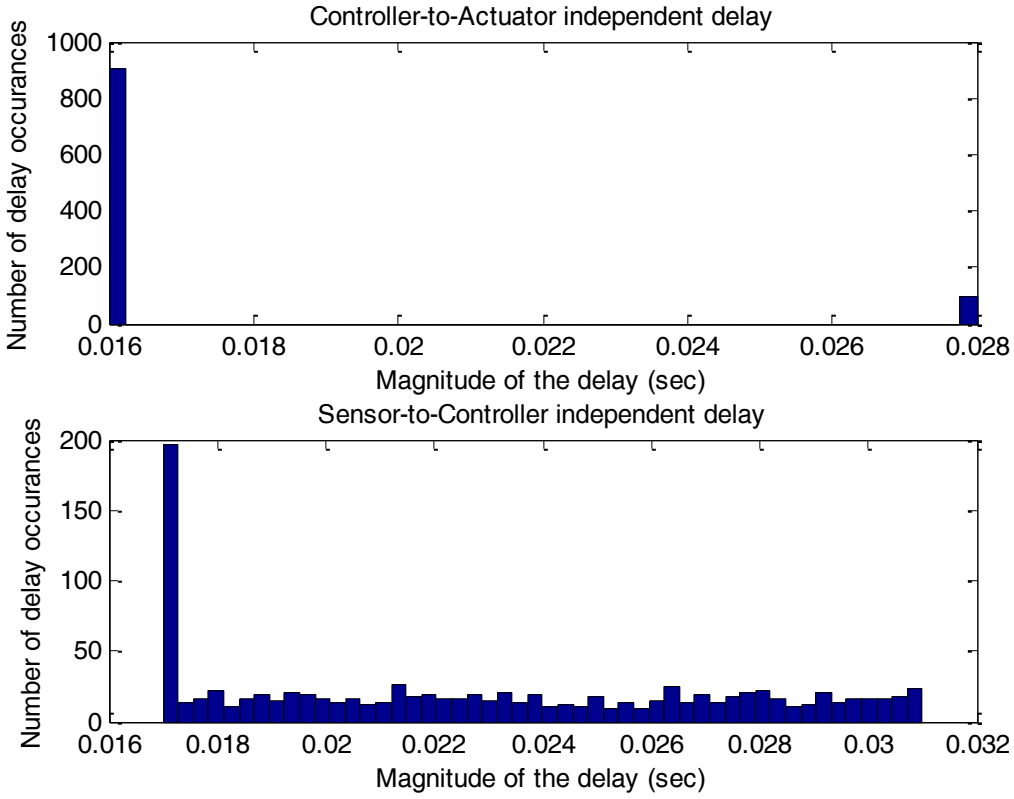


Figure 2.7: **Probability Distribution Function describing τ^{ca} and τ^{sc}**

The simulations show clearly that the two delays are completely independent and have different PDF structures, implying different influence on the network control system. The values of the delays are selected based on the maximum allowable time varying delay represented by parameters $p_{sc}, p_{ca} \in [0,1]$ of the distribution.

In other cases the delays are assumed to be less than a sampling period (Heemels, 2009), (Zhang and Yu, 2008) and in some cases considered to be less or greater than the sampling period (Branicky et al, 2000).

Cloosterman et al (2010) argued that when the delays are less that a sampling period the control design problem can be solved by using lifted state vector and robust control methods for parameter uncertainties or by applying the Lyapunov-Krasovskii Functional (LKF) approach (Lam et al, 2007). Lifted state vector is the mapping between sampling at the sensor and sampling at the actuator. This is because of dual sampling rates between the sensor and actuator (Ding and Chen, 2005).

In the case where delays are greater than the sampling period, Linear Matrix Inequalities and Predictive control are suggested (Ma et al, 2006), (Wang and Liu, 2008). Predictive control is used in the thesis.

Another approach to determine the model of the network induced time delays is by using the model of Transmission Control Protocol (TCP) described by differential equations (Lin C, 2005), based on the model congestion-avoidance mode. Key network parameters such as TCP sessions, channel bandwidth and Round-Trip Time (RTT) protocol are considered.

The TCP Model developed by (Lin C, 2005) is created using fluid-flow and stochastic differential equations and is described by the following Equations (2.3) and (2.4):

$$\dot{W}_{tcp}(t) = \frac{1}{R(t)} - \frac{W_{tcp}(t)}{2}\frac{W_{tcp}(t-R(t))}{R(t-R(t))}p(t-R(t)), \qquad \textbf{(2.3)}$$
$$W_{tcp}(0) = W_{tcp\,0}$$

$$\dot{q} = \begin{cases} \dfrac{N(t)}{R(t)}W_{tcp}(t) - C, q(0) = q_0, q > 0, \\ \max\left(0, \dfrac{N(t)}{R(t)}W_{tcp}(t) - C\right), q = 0, \end{cases} \qquad \textbf{(2.4)}$$

where W_{tcp} is the average TCP window size (packets); q is the average queue length (packets); $R(t)$ is the Round Trip Time (RTT); C is the channel bandwidth (packets/s); N_{tcp} is the loading factor (that is number of TCP sessions); $p \in [0,1]$ is the probability of packet mark.

Based on the model described above, the RTT is described by Equation (2.5)

$$R_i(t) = \frac{q(t)}{C} + T_p, i = 1,...., N_{tcp}, \quad \textbf{(2.5)}$$

where T_p, is the fixed propagation delay, and $\frac{q(t)}{C}$ is the queuing delay

2.4. Discrete Kalman Filters for systems with time delay

The problem of state estimation and control with delayed measurement output and control is not new and has been studied before the advent of Network Control Systems (Mishra and Rajamani, 1975), (Shi et al, 2009). One method that became popular in addressing the issue of time delays within a system is the use of a Discrete Kalman Filter. Since the development of the filter by Rudolph Kalman in the sixties (Kalman, 1960), the filter has been used in many applications including control, signal processing and communication (Shi et al, 2009), (Gopalakrishnan et al, 2010).

One of the features that makes the filter popular is its ability to interpolate, filter and predict the sample value of the measured signal in what is generally referred to as estimation. To achieve the latter the Kalman Filter have general equations that are grouped into time update equations and measurement update equations (Welch and Bishop, 2006).

Time update equations:

$$\hat{x}(k)^- = A\hat{x}(k-1) + Bu(k-1) \quad \textbf{(2.6)}$$

$$P^-(k) = AP(k-1)A^T + Q \quad \textbf{(2.7)}$$

Measurement equations:

Kalman Gain:

$$K(k) = P^-(k)C^T[CP^-(k)C^T + R]^{-1} \quad \textbf{(2.8)}$$

State Estimate:

$$\hat{x}(k) = \hat{x}(k)^- + K(k)[y(k) - C\hat{x}(k)^-] \quad \textbf{(2.9)}$$

Error covariance:

$$P(k) = [I - K(k)C]P^-(k) \quad \textbf{(2.10)}$$

Where $(k)^-$ and k denotes the instants just before and just after the measurement is made and incorporated. These equations are for systems without time delay.

The methods proposed in the literature to handle delayed measurements are in (Alexander, 1991, Larsen et al, 1998, Zhang et al, 2005, Julier and Vhlmann, 2005, Van der Merwe, 2004, and Tasdis et al, 2008). They rely on fusing the information from the delayed measurements directly into the filter once they arrive.

Another approach is augmentation of the state space model (Gudi et al, 1994, Gudi et al, 1995, Tatiraju et al, 1999, Mutha et al, 1997, and Amirthalingam et al, 2000). The state augmentation approach is used in the chemical and biochemical processes because the state-space formulation is preserved which makes simple extension of delay handling methods to other approaches, as for example the moving horizon estimation (Rao et al, 2003 and Vachhani et al, 2005), nonlinear dynamics data reconciliation (NDDR) (Vachhani et al, 2005), Unscented Kalman Filter (UKF), particle filters etc. The fixed-lag smoothing algorithm is widely used in process control (Gelb, 1974), other algorithm is to recalculate the filter for the entire delay period when the delayed measurement arrives (Mutha et al, 1997), when the value of the time delay is of the order of a few sampling periods and is assumed to be unknown ahead. Extension to uncertain delays, has not received enough development. When the delay is big or unknown the methods of fix-delay smoothing or a filter recalculation are used but they result in expensive computations. This leads to a need for new methods that can give solution for the cases of irregular and time-varying delay efficiency.

Gopalakrishnan et al, (2010) makes a critical analysis and does simulation studies of the existing methods capable to cope with multi-rate estimation using Extended Kalman Filter for process systems. All methods are presented in a unified framework. The methods are also divided in two groups:

- fusing delayed measurements on arrival and
- relying on state augmentation

The first type methods are mostly used for tracking and navigation systems, then in process systems.

2.4.1 Output equations for measurements with delay and without delays

Equation output for the measurements without delay at time k. is given by Equation (2.11).

$$y^1(k) = g^1(x(k)) + v^1(k) \qquad \textbf{(2.11)}$$

Output equation for measurements arriving with time delay at time k but are measured at time s is;

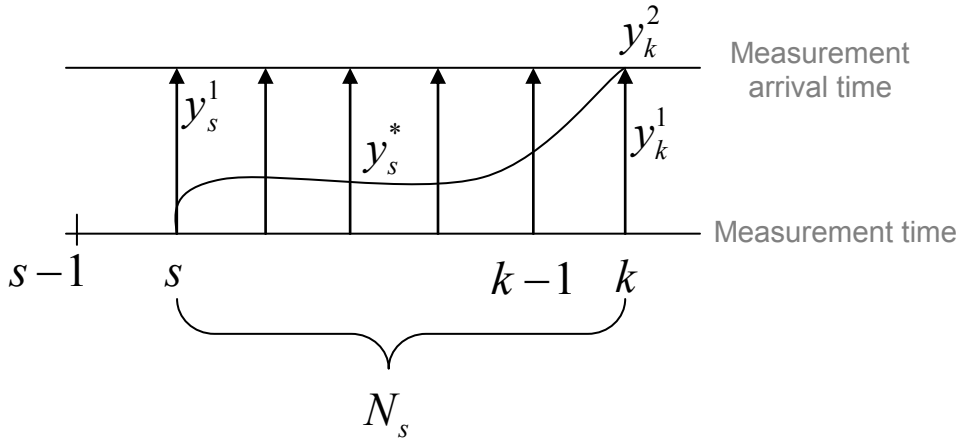


Figure 2.8: **Measurement time - Timing diagram**

$$y^2(k) = g^2(x(s)) + v^2(s), \quad if \quad k = s + Ns \tag{2.12}$$

or

$$y^2(k) = g^2(x(k - Ns)) + v^2(k - Ns) \tag{2.13}$$

where Ns is the value of the delays in sampling periods.

2.4.2 State estimation with delays - Filter Recalculation

The Kalman Filter is extended by going back to the time step when the sample of the delayed measurement was taken and to incorporate the measurement and re-compute the entire trajectory of estimates till the current step (Prasad et al, 2002). The prediction method is used for controller design in the thesis.

When the delayed variable is sampled at time s, the state estimate $\hat{x}(s \mid s-1)$ and covariance of prediction error $P(s \mid s-1)$ are stored until the arrival of this measurement at moment $k = s + Ns$, all not delayed measurements $y^1(s).....y^1(k-1)$ are also stored. During this period only the non delayed measurements are stored at every moment k. At the moment $k + Ns$ both delayed and non delayed $y^2(k)$ and $y^1(k)$ arrive. At this moment the state estimates from time s to time $s + Ns$ are recomputed. The measurements $y^1(s)$ and $y^2(s + Ns)$ are used to update the estimates and their error covariance matrix at time s and further at the subsequent time instances. The available non delayed measurement $y^1(s), k = \overline{s, k-1}$ are used in the update step.

Main difficulties of this method are the large memory for storage of data and a lot of computations, especially if Ns period is longer. The method is not practical for on-line implementation.

2.4.3 Alexander's method (Alexander, 1991)

This method is based on the idea of sequential processing of measurements. This can be done if the measurement's errors are statistically independent (Mendel, 1971). The non delayed measurements are first fused through the Kalman Filter using equations of the KF. This results in the Kalman gain $K^1(k)$ and $P^1(k|k)$ which are then used to compute state estimate $\hat{x}^1(k|k)$. At all sampling moments k it is used $\hat{x}\ (k|k) = \hat{x}^1(k|k)$ and $P(k|k) = P^1(k|k)$ because only $y^1(k)$ are available.

If $y^2(k)$ are available without delay then the calculation of the Kalman gain and the covariance matrix continue as

$$K^2(k) = P^1(k|k)C(k)^{2T}[C^2(k)P^1(k|k)C^{2T}(k) + \theta_2]^{-1} \quad \textbf{(2.14)}$$

$$P^2(k|k) = [I - K^2(k)C^2(k)]P^1(k|k) \quad \textbf{(2.15)}$$

The error covariance $P^2(k|k)$ includes the effects of both sets of measurements. A correction is added to the state estimates at the moment $k + Ns$ to sequentially fuse the measurements $y^2(k)$:

$$\delta\hat{x}(k) = K^2(k)\left[y^2(k) - C^2(k)x(k|k-1)\right] \quad \textbf{(2.16)}$$

If the measurements $y^2(k)$ are delayed they cannot be fused using (Rao et al, 2003, Vacnhani et al, 2005, Prasa et al, 2002). Alexander's method (Alexander, 1991) is used on the fact that for linear systems the error covariance and the Kalman gain are dependant only on the variance of the error in the measurements and not on the measurement value. In this way if the delayed measurement is not available at time s, the Equation (2.14) and (2.15) could be used to update the Kalman gain and the error covariance if the non delayed measurements are available. As $y^2(k)$ is not available, the updates of the state estimates are calculated using only the non delayed measurements. At the moment $s + Ns$ the delayed measurement is available then the correction term added to the filtered state estimate:

$$\delta\hat{x}(s + Ns) = WsK^2(s)\left[y^2(k) - C^2(k)\hat{x}(s|s-1)\right] \quad \textbf{(2.17)}$$

$K^2(s)$ is the Kalman gain matrix at the moment s. It is updated under the assumption that the delayed measurement is updated immediately as shown in Equation (2.14). If Equation (2.16) and (2.17) are compared it could be seen that the matrix Ws is:

$$Ws = \prod_{i=1}^{i=Ns} (I - K^1{}_{(s+i)} C^1{}_{s+i}) A_{s+i-1} \quad \textbf{(2.18)}$$

This matrix takes account for the measurement delay.

$K^1(k)$ is calculated for $k \in [s+1, s+Ns-1]$

$K^1(k)$ is computed for with $P(s \mid s) = P^2(s \mid s),\ K^1(k)$ is required with $P(s \mid s) = P^1(s \mid s)$.

This method determines

- sub-optimal estimates in the interval $[s+1 \div s+Ns-1]$
- optimal state estimate at the moments $s+Ns$ based on the correction (Mendel, 1971) and (Rubinstein, 1978).

This method was extended in Larsen et al, (1998) in the case of unknown measurement error variance for the moment when the delayed measurements are done.

The method requires only the correction term to be stored and the computation burden is lower. The method also can be used for varying time delay and also for unknown time delays.

2.4.4 Method of parallel filters

It is suggested by Larsen et al (1998) that optimal estimates can be obtained also at the interval $[s+1 \div s+Ns-1]$ if second parallel filter can be used with the first one. This filter can start at the moment s using the updated Kalman gain and error covariance matrix as in the Alexander's method. This filter calculates only $K^1(k)$ and Ws to be used at the moment $s+Ns$. For $k \in [s, s+Ns-1]$ only $K^1(k)$ is computed in the Alexander's method, but both $K^1(k)$ and $K^1(s)$ and the corresponding covariance are computed in this method as $K^1(k)$ is computed in the first filter for the considered period.

2.4.5 State augmentation methods

This method is based on augmenting the current state with the past information needed for taking account of the effect of the delayed measurement. The augmented state vector z is:

$$z(k+1) = \varphi_k z(k) + \Gamma_k u(k) + \Psi_k \varepsilon_k \quad \textbf{(2.19)}$$

The measurement equations are:

$$y^1(k) = \Sigma_k^1 z(k) + v^1(k) \quad \textbf{(2.20)}$$

$$y^2(k) = \Sigma_k^2 z(s) + v^2(s), \text{ if } k = s + Ns \quad \textbf{(2.21)}$$

The positive quality of this method is that it retains the state space description of the system.

2.4.6 Fixed-lag smoothing

This algorithm is considered in (Anderson and Moore, 1979 and Gelb, 1974). It is based on the fact that the delayed measurement affects all state for the period $[s, s+Ns]$, all information is stored by augmenting the current state with the past Ns states, as follows:

$$z(k) = [x^T(k), x^T(k-1) \ldots \ x^T(k-Ns)]^T \quad \textbf{(2.22)}$$

The matrices of the augmented model are:

$$\varphi_k = \begin{bmatrix} A_k & 0 & .. & 0 & 0 \\ I & 0 & ... & 0 & 0 \\ 0 & I & ... & 0 & 0 \\ : & : & : & : & : \\ 0 & 0 & ... & I & 0 \end{bmatrix} \quad \Gamma_k = \begin{bmatrix} B_k \\ 0 \\ 0 \\ : \\ 0 \end{bmatrix} \quad \Psi_k = \begin{bmatrix} 1 \\ 0 \\ 0 \\ : \\ 0 \end{bmatrix} \quad \textbf{(2.23)}$$

$$\begin{bmatrix} \Sigma_k^1 \\ \Sigma_k^2 \end{bmatrix} = \begin{bmatrix} C_k^1 & 0 & ... & 0 & 0 \\ 0 & 0 & ... & 0 & C_k^2 \end{bmatrix}$$

These models are used to calculate the Kalman gain and covariance matrices. The method leads to smoothing of the past Ns states based on the non-delayed measurements at the moments k, k=1,2... when the delayed measurement arrives, both delayed and non-delayed measurements are used to obtain smoothed estimates from s to s+Ns. For the case of unknown and time varying delay the system can be augmented with N_{max}, where N_{max} is the maximum delay.

The method is also characterised with excessive computational burden. As the augmented state estimates are obtained at every moment the number of calculations is bigger than in the previous methods.

2.4.7 Measurement augmentation

This algorithm uses only the values of the delayed measurements to be added to the vector of state space. The dimension of the matrices is lower than in the previous step, but still the calculation burden is big.

2.4.8 Sample state augmentation

This algorithm is based on the fact that the delayed measurement is a function of the state x(s). The only information for this state is needed to be kept until the time s+Ns.

Using this fact the system state is augmented with the state at time s, when the delayed measurement is done (Van der Merwe, 2004). The augmented vector is:

$$z(k) = [x^T(k), x^T(s)]^T \quad \textbf{(2.24)}$$

and the matrices are

$$\varphi_k = \begin{bmatrix} A_k & 0 \\ 0 & I \end{bmatrix}, \Gamma_k = \begin{bmatrix} B_k \\ 0 \end{bmatrix}, \psi_k = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \Sigma_k = \begin{bmatrix} C_k^1 & 0 \\ 0 & C_k^2 \end{bmatrix} \quad \textbf{(2.25)}$$

Then when the delayed variable is measured, the augmented state estimate is:

$$\hat{z}(s) = \begin{bmatrix} \hat{x}(s \mid s) \\ \hat{x}(s \mid s) \end{bmatrix} \quad \textbf{(2.26)}$$

The covariance of the augmented estimate error is

$$P^{aug}(s \mid s) = \begin{bmatrix} P(s \mid s) & P(s \mid s) \\ P(s \mid s) & P(s \mid s) \end{bmatrix} \quad \textbf{(2.27)}$$

The Kalman filter will update the current state and the delayed state x(s) during the time of the delay. This can be considered as a fixed-point smoothing of the lagged state x(s) using the future measurements $y^1(s), y^1(s+1), ...y^1(s+Ns-1)$. When the delayed measurement arrives at s+Ns it is fused with the smoothing estimate of x(s) using $\sum_k$ from Equation (2.25). At this moment the delayed state x(s) is not needed anymore and the state is un-augmented $z(k) = x(k)$.

The method is applicable for uncertain and time varying delays. The information for the maximum delay is not necessary in this case. This case is considered in the thesis.

Other methods include modification of the DKF, state augmentation combined with the standard DKF, (Gopalakrishnan et al, 2010), fractional DKF (Sierociuk et al, 2010) and Robust Extended Kalman Filter (Xiong et al, 2011), probabilistic approach of Kalman filter over packet-delaying network given the delay distribution and determine the minimum required buffer length (Shi et al, 2009). According to the review by Xiong (2011) there are two popular Robust Kalman Filter algorithms and they are Riccati Equation approach (Zhu et al, 2002) and Linear Matrix Inequality (LMI) (Lu et al, 2003). Zhu et al (2002), in designing a robust Kalman filter considered the Riccati equation approach.
Robust Kalman filter usually yields a suboptimal solution.

2.5. Models of packet loss in networked control systems

Packet loss is as a result of network induced time delays, random queuing and network inconsistency (Sadjadi, 2003). It is important to note that the communication between the controller and actuator and sensor and the controller is happening at the network layer, implying that packets sent are either lost or received, unlike for the case of network induced time delays, where packets eventually get to the destination. When packet loss occurs the packets are gone as the name suggests. Researches have derived different methods for modelling packet loss within a network and the most popular model is to represent packet loss activity as an ON and OFF switch, that is 1 when u_k is not dropped and 0 when u_k is dropped as shown in Equation (2.23). (Heemels M., 2009), where u_k is the control signal, r_k is the reference signal, h_k is the sampling rate.

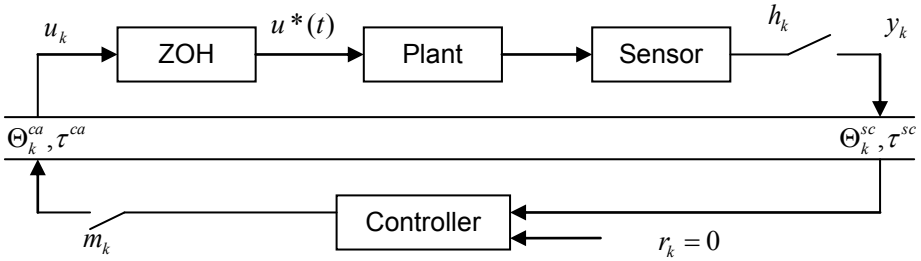


Figure 2.9: NCS with packet loss (Adapted from (Heemels M., 2009))

$$m_k = \begin{cases} 1, & u_k \text{ is not dropped} \\ 0, & u_k \text{ is dropped} \end{cases} \quad \textbf{(2.28)}$$

The approach above is simple but some questions appear on what does the actuator do when it losses the packets: Does it use the last value (Zero Order Hold approach) or does it apply zero control? The questions have led to investigations considering a stochastic process model that models the unreliable nature of the link based on Identical Independent Bernoulli process (Imer et al, 2005), (Wang and Zhang, 2009), (Jia et al, 2010), (Schenato et al, 2007). If the link fails the independent Bernoulli process parameter is zero, otherwise it is one. Note that there are two Bernoulli process parameters, one for the link between the controller and the actuator Θ_k^{ca} and the other between the sensor and the controller Θ_k^{sc}. Then the model of the process is described by the following stochastic difference equations, (Imer et al, 2005):

$$\begin{aligned} x_{k+1} &= A_k x_k + \Theta_k^{ca} B_k u_k \\ y_k &= \Theta_k^{sc} C_k x_k, k = 0,1,... \end{aligned} \quad \textbf{(2.29)}$$

Ishii (2009), uses the similar approach as Heemels (2009), but has a loss probability function Θ_k^{ca}.

$$\theta_k = \begin{cases} 1, & \text{If the packet is received} \\ 0, & \text{If the packet is lost} \end{cases} \quad \textbf{(2.30)}$$

$$\Theta_k^{ca} = \{\theta_k = 0\}, \forall k \quad \textbf{(2.31)}$$

2.6. Quantization

Quantization effects affect the stability and performance of NCS (Ishido et al, 2011), because typical control system signals take real values and these values have to be converted from continuous to discrete values, see Figure (2.10).

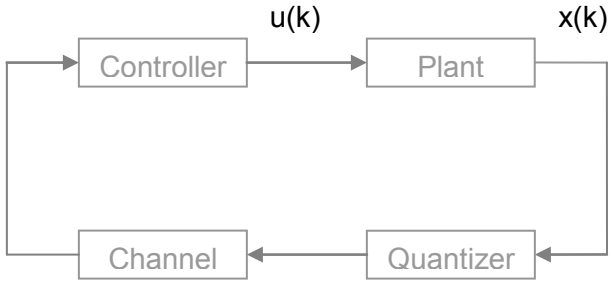


Figure 2.10: Quantized Control

If the quantization resolution is high the system performs reasonable and the system is unstable if the resolution is coarse. Luck (1990) suggested that there exists a quantization state and control scheme to stabilize the system and he proposed making the system time invariant. The main questions asked by control engineers are;

- What is the minimum number of bits required for stability of NCS?
- Are there quantizer structures suitable for NCS?

Typically quantization error is modelled as additive white noise; however, this model becomes inaccurate when quantization resolution is coarse (Ishii, 2009). Ishii (2009) proposed an estimation approach for design of the controller in NCS as shown in Figure (2.11).

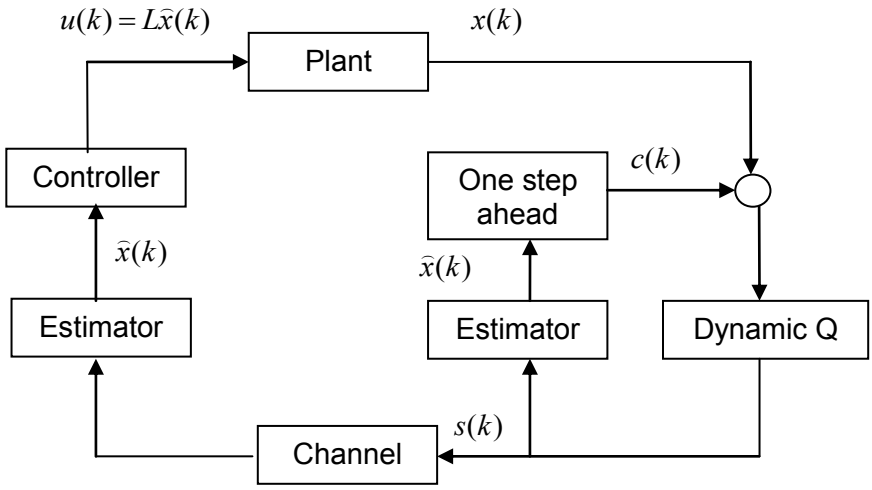


Figure 2.11: **Estimation based structure of the NCS (Adapted from (Ishii H., 2009))**

where $\hat{x}(k)$ is the coarse estimate of state from quantized signal, $c(k)$ is the estimate of one-step ahead and $c(k+1) = A\hat{x}(k) + Bu(k), \quad c(0) = 0$.

The effects of the ZOH over the performance of the networked control systems is done using investigations of both continuous and discrete systems.

Generally dynamic plant models are represented by difference equations (Nilsson, 1998), (Cloosterman et al, 2010), (Zhang and Hristu-Varsakelis 2005) or deferential equations (Gao et al, 2008), when considering the problem definition for analysis of the networked control systems in discrete time domain as shown:

$$x(k+1) = A(k)x(k) + B(k)u(k) + w(k) \quad \textbf{(2.32)}$$

$$y(k) = C(k)x(k) + v(k) \quad \textbf{(2.33)}$$

where $x(k) \in R^n$ is the state of the system, $y(k) \in R^l$ is the observation, $u(k) \in R^d$ is the control, $v(k) \in R^m$ and $w(k) \in R^m$ are independent discrete time white Gaussian processes, and $A(k), B(k)$ and $C(k)$ are known matrices with proper dimensions.

Or in continuous time domain:

$$\dot{x}(t) = A(t)x(t) + B(t)u(t) + w(t) \quad \textbf{(2.34)}$$

$$y(t) = C(t)x(t) + v(t) \quad \textbf{(2.35)}$$

where $x(t) \in R^n$ is the state of the system, $y(t) \in R^l$ is the observation, $u(t) \in R^m$ is the control, $v(t) \in R^l$ and $w(t) \in R^n$ are independent discrete time white Gaussian processes, and A, B and C are known matrices with proper dimensions. Both representations have the advantages and disadvantages as shown by Zhang and Yu

(2008). The main challenge when considering continuous system is the ZOH that introduces another delay factor.

Different approach has been used to solve the problem firstly by considering the effects when the plant is either discrete (Nilsson, 1998), (Cloosterman et al, 2010), (Zhang and Hristu-Varsakelis 2005) or continuous (Gao et al, 2008).

When considering a continuous system the ZOH is in series with the dynamic plant model and the time delay. ZOH is often not analysed when considering the influence of network induced time delay even thought it has a great impact on the stability and performance of the closed loop control system, because ZOH introduces a time-varying delay within each control update interval which linearly increases the overall network induced time delay (Cloosterman et al, 2010), (Hu et al, 2006). One approach of looking at the ZOH challenge is to ignore the data and regard the data as lost instead of holding the data (Zhang and Hristu-Varsakelis, 2005). Another approach is to apply the ZOH and keep the old data (Imer et al, 2005).

2.7. Methods for design of robust controllers for NCS

The fundamental challenge when designing a robust controller for Networked Control System is the amount of uncertainties introduced by the network that affects the stability and performance of the system (Wang and Yang, 2012).

The list of the uncertainties is endless, in the thesis the major uncertainties that are considered because of the impact they have on system stability and performance are the network induced time delays. The latter agrees with the definition of Robust Controller, which is “the control of unknown plants with unknown dynamics subject to unknown disturbancies ” (Rollins, 1999). The early methods of Bode and others were fairly robust; the state-space methods invented in the 1960's and 1970's were sometimes found to lack robustness (Rollins, 1999).

There exists a fundamental trade-off between robustness and performance, therefore a design shall be able to optimize the two of them. The challenge is to find admissible control $u^*(k)$ which causes the system to follow an admissible trajectory $x^*(k)$ that minimizes the performance measure J (Kirk, 2004).

The second challenge is the precision of derivation of the plant model. If the model is very close to the real system it makes it easier to design a more robust system as most of the system dynamics may have been incorporated in the model and the controller can

be designed to compensate for them. In the thesis during model derivation wind disturbance on both the dish structure and dish drives is incorporated in the model and the controller is designed to compensate for them.

There are two main control strategies for the design of NCS being:

- Design the controller without regard to the network induced delays, then design communication protocols to minimise or make the delay to be constant.
- Design the controller to compensate for the network induced delays (Kurzweil (1964).

In the thesis a Pole Placement, Linear Quadratic (LQ) predictive optimal controllers are designed. They incorporate the network induced time delays represented by the derived LTI mathematical model.

2.7.1 Linear Quadratic Gaussian (LQG) Controller

Different methods for designing of the controller have been investigated, considering different combination of the imperfections highlighted in the sections above. Linear Quadratic Gaussian is one of the popular controller design methods when it comes to networked control systems (Lian, 2002), (Sadjadi, 2003), (Schenato et al, 2006). The problem definition when designing LQG optimal controller is mostly considered in discrete form as for the process described by the model:

$$\begin{aligned} x(k+1) &= A(k)x(k) + B(k)u(k) + w(k) \\ y(k) &= C(k)x(k) + v(k), \end{aligned} \qquad \textbf{(2.36)}$$

where $x(k) \in R^n$ is the state of the system, $y(k) \in R^l$ is the observation, $u(k) \in R^m$ is the control, $v(k) \in R^l$ and $w(k) \in R^n$ are independent discrete time white Gaussian processes, and $A(k), B(k)$ and $C(k)$ are known matrices with proper dimensions. Find a feedback controller such that the quadratic cost function

$$J = \frac{1}{N} E\left\{ x_N^T Q_N x_N + \sum_{k=0}^{N-1} (x_N^T Q_k x_k + u_k^T R_k u_k) \right\}, \rightarrow \min \qquad \textbf{(2.37)}$$

is minimised, where R_k is a positive definite matrix, Q_k is a positive semi-definite matrix, $u_k = \mu_k(I_k)$ is an admissible control, $I_k = \{y_0, y_1, ..., y_k, u_0, u_1, ...u_{k-1}\}$ is the information vector and $I_0 = \{y_0\}$ is the initial information vector. Minimized Information vector is considered when TCP, UDP or both are used, and it is the combination of observation $\{y_k\}$, control command $\{u_k\}$, and control command packet delivery indicator

$\{\Theta_k\}$ in the case of packet loss (Sadjadi, (2003), (Imer, 2005), (Zhang and Hristu-Varsakelis, 2005), (Garone, 2010).

The optimal control problem in Equation (2.36) and (2.37) has a state space controllers as a solution. This controller needs the values of the states of the process for its real-time implementation. This means that design of a state estimator is needed. The closed loop system can be described by Equations (2.38) and (2.39)

$$\mu_k^*(I_k) = L_k E\{x_k \mid I_k\} \quad \textbf{(2.38)}$$

$$L_k = -(R_k + B_k^T P_{k+1} B_k)^{-1} B_k^T P_{k+1} A_k, \quad \textbf{(2.39)}$$

where L_k is the controller gain matrix and P_{k+1} is the solution of Riccati equation.

The closed loop system is then given by Equation (2.40), where the case of using state estimator is considered.

$$x_{k+1} = A_k x_k + B_k L_k \hat{x}_k + w_k, \quad \textbf{(2.40)}$$

where $\hat{x}_k$ is the estimated state space vector.

Methods for solution of LQG problem are mainly developed considering packet loss (Schenato et al, 2006), (Gupta et al, 2006), (Zhang and Hristu-Varsakelis, 2005) and in some cases considering both packet loss and time delay (Wang and Zhang, 2009), (Imer et al, 2005), (Mitter and Borkar, 1995). Lian (2002) did considered the LQG problem using distributed constant delays.

The general method of representing packet loss and time delay when designing LQG controller is to introduce random diagonal matrix $\Theta_k^{ca} = diag(\theta_k^1,...,\theta_k^m)$, $\theta_k^i = 0$ indicates that the control command u_k^i . The i^{th} component of u_k, is lost and $\theta_k^i = 1$ indicates that this component of the control command is delivered.

$$\begin{aligned} x(k+1) &= A(k)x(k) + B(k)\Theta_k u(k) + w(k) \\ y(k) &= C(k)x(k) + v(k), \end{aligned} \quad \textbf{(2.41)}$$

The same applies to the packet loss between the sensor and controller. The sensor data that arrives at the estimator can be modelled as:

$$y_k = \theta_k^{sc} y_k, \quad \textbf{(2.42)}$$

where $\theta_k^{sc} = diag(\theta_k^1,...,\theta_k^l)$ is a diagonal random matrix. Both θ_k^{sc} and Θ_k^{ca} are represented as independent and identically distributed (i.i.d) Bernoulli processes (Wang and Zhang, 2009), (Imer et al, 2005).

The optimization problem is mainly solved in finite-horizon and infinite-horizon for both UDP and TCP information vectors. Based on Equation (2.41) a quadratic cost function as show below is used for the purpose of design of the controller (Imer et al, 2005):

$$J = E\left\{ x_N^T Q_N x_N + \sum_{k=0}^{N-1} (x_N^T Q_k x_k + \Theta_k u_k^T R_k u_k) \right\}, \quad \textbf{(2.43)}$$

As mentioned above the solution of the problem is through design of a state estimator and of a state controller. The information vectors used for the UDP and TCP networks are: (Imer et al, 2005).

$$\begin{aligned} I_k^{UDP} &= \left\{ y_0, y_1, ..., y_k, u_0, u_1, ... u_{k-1}, \theta_0^{sc}, ... \theta_k^{sc} \right\}, k = 1,2,... \\ I_0^{UDP} &= \left(y_0, \theta_0 \right) \end{aligned} \quad \textbf{(2.44)}$$

and

$$\begin{aligned} I_k^{TCP} &= \left\{ I_k^{UDP}; \Theta_0^{ca}, ..., \Theta_{k-1}^{ca} \right\}, k = 1,2,... \\ I_0^{TCP} &= I_0^{UDP} \end{aligned} \quad \textbf{(2.45)}$$

Equation (2.46) shows the state estimator for TCP information vector, considering the unreliable communication link even thought TCP has packet acknowledgement.

$$\hat{x}_k = \begin{cases} A\hat{x}_{k-1} + \Theta_{k-1} B u_{k-1}, \theta_k^{sc} = 0 \\ x_k, \theta_k^{sc} = 1 \end{cases} \quad \textbf{(2.46)}$$

and the optimal control is given by:

$$u_k^* = L_k E\left\{ x_k \mid I_k^{TCP} \right\} \quad \textbf{(2.47)}$$

where L_k is the matrix of the state space controller.

Schenato et al, (2006) used a different approach by investigating the theoretical implication of using unreliable network for control and estimation. TCP and UDP protocols were considered using optimal LQG control and the findings were that the separation principle holds for TCP protocol because of packet acknowledgement that is inherent in the TCP structure. For UDP protocol the separation principle does not hold because there is no packet acknowledgement on arrival of packets. It was found that for TCP the general LQG control is linear and for UDP it is non-linear. The packet drops between the controller and the actuator and the sensor and the controller are governed by two independent Bernoulli processes.

Zhang and Hristu-Varsakelis (2005) proposed another approach where LQG problem is solved without relying on dynamic programming for solving LQ optimal problem like in (Sadjadi, 2003) and (Imer, 2005) and the considered MIMO system. To illustrate the above a state estimation using Kalman filtering approach and LQ optimal controller design is proposed.

2.7.2 H∞ controller

Another approach that is used for design of controllers is H_∞ where the controller is designed based on state feedback and on output feedback (Jia et al, 2010), (Wang and Yang, 2011). Gao et al, 2008 considered a new delay system approach to networked-based control by considering multiple successive delay components in the state. The approach proposed is Lyapunov-Krasovskii Functional for stability analysis and H_∞ for performance analysis. Jia et al, 2010 considered the case where multiple packet loss over the network occurs and proposed the H_∞ controller design.

2.7.3 Fuzzy PID and Smith Predictor Controllers

There are different controller design methods to ensure the overall system robustness. Chen and Lin (2005) used a Fuzzy PID controller to ensure that tracking performance of the overall system is achieved and an adaptive Smith predictor to compensate for uncertainties introduced by the network induced time delays is developed. The traditional Smith predictor could not cope with the difficulty of network induced delays and a neural network featuring approximating capabilities was used as an ideal choice, substituting the Smith predictor. The latter improved the time-delay compensation.

The disadvantage of this approach is that the Smith predictor requires a precise model of the plant, in this case a model of the network dynamics. The latter makes it even more challenging as there is a general lack of network dynamic models to describe the delay behaviour. Another method used for delay estimation online is the Fuzzy PID controller. This method is based on the Delay Window (DW) where DW is set and the controller keeps historical and current delays values at the specific sampling time (Zhang and Hua-Jing, 2005). Ma et al (2006) took the research one step further by determining the Maximum Allowable Delay Bound (MADB) proposing Linear Matrix Inequalities (LMI) as a method for design of a controller (Jian and Shen-Quan, 2012). MADB considers delays greater than one sampling period. Predictive control model is suggested by Wang and Liu (2008) to overcome the influence of stochastic time delay.

The control and estimation problems where the sensor signals are transmitted to various sub-systems over a network are considered. In contrast to traditional control and estimation problems where the sensor signals are not transmitted via a shared network, here the observation and control packets may be lost or delayed due to the network limitations. The unreliability of the underlying communication network is modelled stochastically by assigning probabilities to the successful transition of packets. This requires a novel theory which penalises classical control/estimation paradigm.

The central contribution is to characterise the output of the network reliability on the performance of the feedback loop. There exists a critical threshold of the network reliability below which the optimal controller fails to stabilise the system (Shi et all, 2009). Further for these protocols (where the successful transmission is acknowledged at the receiver) the separation principle holds and the optimal LQ controller is linear function of the estimated state.

2.8. Simulators and Emulators

Simulation is the imitation or representation of the operation or features of one process or system through the use of computer simulation. Emulator is the hardware device or a program that pretends to be another particular device or program that other programs expect to interact with (www.whatis.com). Simulation and emulation are mainly used to design, analyse and verify the design as it saves a lot of time and costs to be certain of the design before system is implemented.

Networked control system overlaps between two areas of study that are information systems and feedback control system as shown in Figure (2.12).

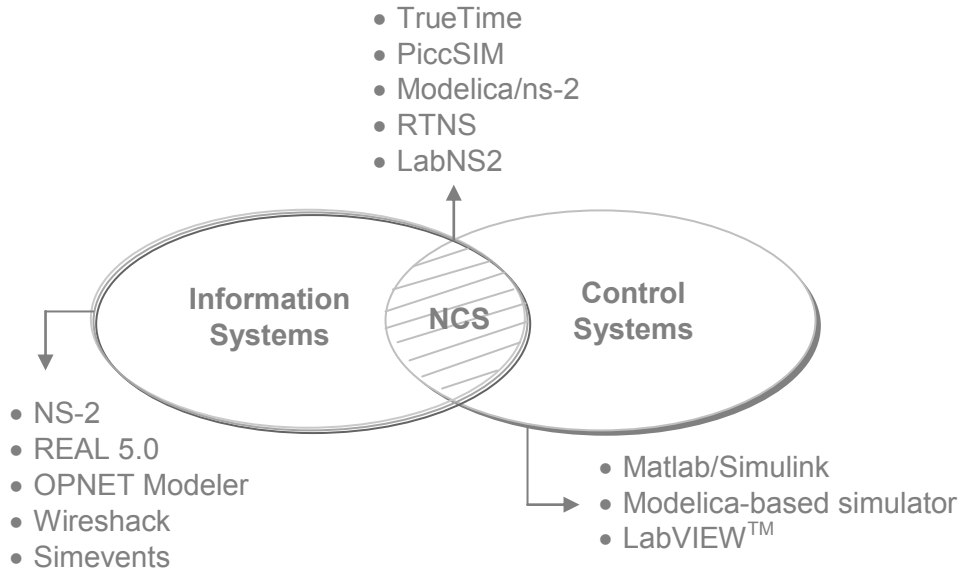


Figure 2.12: Simulators and Emulators for NCS

The two areas have existed in isolation for decades and as such both have their own tools used to simulate and emulate the environment for the design of respective applications. The simulation and emulation tools are shown in Figure (2.12) above for the information systems, control systems and network control systems. For information systems, the NS2 is the most widely used simulator and emulator, (Fall and Varadhan, 2008). The approach adopted by control engineers in designing NCS simulators and emulators is to leverage from existing information systems simulators and control systems simulators and emulators; a typical example is PiccSIM (Pohjola and Nethi, 2009) and Modellica/ns2 (Al-Hammouri et al, 2007). In the thesis LabNS2 is presented and described in Part I of Chapter 9 as an integration between LabVIEW and NS2. The integration is based on the advantages associated with LabVIEW in control systems design, simulation and emulation.

Another approach is to develop the simulators and emulators for NCS from the first principles. This resulted in simulators and emulators for Control Systems, Networks and Networked Control Systems. Network simulators are typically discrete event simulators, with focus on wired and wireless networks, devices and protocol (Cervin, 2009).

2.8.1 Information systems simulators and emulators

Network Simulator version 2 (NS2) is a discrete event simulator developed in C++. NS2 simulates mainly the bottom four layers of the OSI model being;

- the physical layer for both wired and wireless systems,
- the medium Access layer,
- the link layer and
- the network layer.

The version of NS2 that is used in the thesis is ns-2.31 and the latest version is ns-2.34 with ns-3 under development (http://nsnam.isi.edu/nsnam/). The software is available as freeware

REAL is a network simulator intended to study the flow and congestion control scheme of packet-switched data networks (http://www.cs.cornell.edu/skeshav/real/overview.html). The software is written in C and intended to run on UNIX platform. The software is available as freeware.

OPNET Modeler is a network simulator intended to analyse and design communication networks, devices, protocols, and applications (http://www.opnet.com/). OPNET Modeler is a commercial software package written by OPNET Modeler.

Wireshack is a network protocol analyser and its function is to capture network traffic and show it in a way that humans can read (http://www.wireshark.org/). The software is available as freeware.

2.8.2 Hybrid system simulators and emulators

These simulators and emulators have a high level of abstraction and are mainly used in multiple fields. Each field has a developed toolbox which is used to simulate and emulate the environment.

LabVIEW™ is a graphical programming language where icons are dragged and dropped to define functionality. LabVIEW functions are called Virtual Instruments (VI). LabVIEW is a commercial software package written by National Instruments (NI) (www.ni.com). LabVIEW allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including Matlab, C, C++.

"Matlab/Simulink is a numerical computing environment and fourth-generation programming language developed by MathWorks. MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, and Fortran. Although MATLAB is intended primarily for numerical computing, an optional toolbox uses the MuPAD symbolic engine, allowing access to symbolic computing capabilities. An additional toolbox. **Simulink**, adds graphical multi-domain simulation and Model-Based Design for dynamic and embedded systems. Matlab/Simulink is a commercial software package written by Mathworks (http://www.mathworks.com).

Simevent is a discrete event simulator software which form an extension for the Simulink (http://www.mathworks.com).

"Scilab is an open source, cross-platform numerical computational package and a high-level, numerically oriented programming language. It can be used for signal processing, statistical analysis, image enhancement, fluid dynamics simulations, numerical optimization, and modeling and simulation of explicit and implicit dynamical systems.

MATLAB code, which is similar in syntax, can be converted to Scilab. Scilab is one of several open source alternatives to MATLAB (http://en.wikipedia.org/wiki/Scilab)".

Modelica-based simulator is a non-proprietary, object-oriented, equation-based language to conveniently model complex physical systems containing, e.g., mechanical, electrical, electronic, hydraulic, thermal, control, electric power or process-oriented sub-components. (http://www.modelica.org)

2.8.3 Networked Control System simulators and emulators

TrueTime is a Matlab/Simulink based simulator which has its own real-time kernel that facilitates co-simulation of control task execution, network communication, and plant dynamics developed by Lund University (Cervin, 2009). TruTime is available as freeware and the latest version can be downloaded on this link http://www.control.lth.se/truetime.

PiccSIM is a co-simulator for network and networked control system developed by Helsinki University of Technology (Pohjola and Nethi, 2009). The tool is integration between Simulink and NS-2. In addition to NCS the tool chain has a graphical user interface for modelling the network and control systems. Eyisi (2012) used the similar approach in developing the Networked Control Systems for Wind Tunnel (NCSWT) (Eyisi et al, 2012).

Modelica/ns-2 is a co-simulator for network and networked control system developed at Case Western Reserve University, OH. The tool is integration between Modelica and NS-2 (Al-Hammouri et al, 2007).

Real Time Network Simulator (RTNS) is a co-simulator intended to model operating system mechanisms for distributed networked applications developed by Retis Lab. The tool is integration between NS-2 and RTSim (Real Time Operating System SIMulator). (http://rtns.sssup.it/RTNSWebSite/RTNS.html). RTNS is available as freeware.

SystemC/Matlab is a co-simulator developed by (Quaglia et al, 2012). The tool is integration between SystemC and Matlab.

LabNS2 is a co-simulator for network and networked control system developed by Cape Peninsula University of Technology (Mkondweni and Tzoneva, 2012). The tool is

integration between LabVIEW and NS-2. This tool is presented in Chapter 9 of this thesis.

2.9. Conclusion

This chapter outlines the existing literature on Networked Control System considering most of the sub-problems that constitute the NCS. The considered papers are grouped according to network type and communication protocols, models of time delay, Discrete Kalman Filters for systems with time delays, models of packet loss, methods for design of robust controllers for NCS, LQ controllers for NCS, H-infinity controllers, Fuzzy PID and Smith Predictor controllers. Review on simulators and emulators as used in NCS is described.

The general approach of the existing literature is; packet loss are represented as independent and identically distributed (i.i.d) Bernoulli process between the controller and actuator and the sensor and controller (Zhang and Zheng, 2012), which in reality is not the case as demonstrated by Nilsson, 1998. However this approach can be considered as a start phase to understand the theoretical impact of designing controllers and estimators in the presence of network deficiency as shown by Schenato et al, 2006. The information vector using dynamic programming has been used to represent data transmitted by TCP and UDP protocols. The separation principle has been investigated to demonstrate that it holds at some instances and in others it does not. It was clear that both TCP and UDP protocols have their advantages and disadvantages because of packet delivery possibilities on TCP and non-packet delivery on UDP.

There are different approaches that have been considered in designing the LQG control and the most popular one is the use of dynamic programming and obtaining optimal control in the kind of $u_k^* = L_k E\{x_k \mid I_k^{TCP}\}$. In this case the Information vector is for TCP protocol. A suboptimal approach is used where the controller and estimator are designed separately and the Kalman filter is used as the state estimator. General Riccati equations are used to calculate the controller and estimator gains. This approach is used in the thesis.

It has been shown that incorporation of DKF and modification of the Filter is one of the methods that can be considered to improve the influence of time delays, however the issue of computational burden has to be looked at.

Based on the review from this chapter the following will be considered during the design of the methods for networked control system in the thesis.

- Ethernet network will be considered during the design.
- The derivations will be based on the UDP protocol.
- Simulation and co-simulation will be considered
- The problem will be limited to network induced time delays.
- The delays will be considered between controller and actuator and sensor and controller.
- Simulation will be considered using bounded constant and independent delays.
- Pole Placement and LQ controller shall be used as predictive controller design methods.

The developments in this thesis will be done for the case of a dish antenna considered as part of a networked control system used for space investigations. The description of the dish antenna is done in Chapter 3.

CHAPTER THREE
RADIO ASTRONOMY-OBJECT COORDINATES

3.1 Introduction

Radio Telescope as used in Radio Astronomy is described as a plant under consideration where azimuth and altitude mount drives are controlled to drive the radio telescope dish. The azimuth and altitude mount implies that there are two motors that have to be driven sharing a single load that is a dish, also referred to as an antenna. The background of Radio Astronomy is described in Section 3.2. A comparison between Single Dish Radio Telescope and Interferometer Radio Telescope is described in Section 3.3. Antenna control structure is described in Section 3.4. Antenna pointing accuracy is one of the performance measures (Gary, 2005) of an antenna and it is described in Section 3.5. Coordinate transformation systems are described in Section 3.6.

3.2 Radio Astronomy considered as an application field

Generally people knew of astronomy as a study of visible objects in the sky. The latter would be achieved by the use of naked eyes, binoculars and optical telescopes. This lasted till 1931 when a radio engineer Karl G. Jansky at Bell Telephone Laboratories was assigned a task to study radio frequency interference from thunderstorms in order to help Bell design an antenna that would minimize the problem that occurs when beaming radio-telephone signals across the ocean (Miler, 1998). The unintended outcome of the study was the birth of Radio Astronomy as Jansky identified radiation signals that came outside of earth; it was clear from the signal that the radiation did not come from any nearby source.

Following Jansky's work, the first Radio Telescope prototype was built in 1937. This was built by another radio engineer Groter Reber, and the telescope was named after him Reber's Prototype Radio Telescope (Miler, 1998).

So what is Radio Astronomy: Miler describes it as instruments used to gather up the radiation and reflects it to a central focus, where the radiation is then concentrated (Miler, 1998). The weak current at the focus can then be amplified by a radio receiver so that it is strong enough to be measured and recorded. This development gave birth to the single dish radio telescope, considered in the thesis.

.

Figure 3.1: Reber's Prototype Radio Telescope
(Adapted from http://en.wikipedia.org/wiki/Reber_Radio_Telescope)

3.3 Single Dish Radio Telescope vs. Interferometer

Over the year's radio astronomers have been trying to get higher sensitivity from the single dish radio telescope. The challenge has been the surface area to collect the radiated waves from the objects in the sky. Physics dictates that a radio telescope can have a limited size diameter in terms of dish surface area and the larger the dish the more complex it is to control and build. As a result scientists resulted in building larger non-steerable dishes and rely on the rotation of the earth. The largest single dish radio telescope in the world with dish diameter of 306m is found in Arecibo, see Figure (3.2).

Figure 3.2: Largest single dish radio telescope
The 306m in diameter found in Arecibo (Gary, 2005).

To address the challenges associated with a single dish radio telescope, in 1946 a radio astronomer came up with a novel idea of using smaller dishes to create a big logic dish by recollecting the data from the smaller dishes, in what is called radio interferometer. See Figure 3.3 for a typical radio interferometer.

Figure 3.3: Interferometer Array
ATNF, Narrabri, Australia 2006, 22m dish diameter in a 6 dish array.

3.4 Antenna Control Structure

In this section the antenna control structure is presented. The main focus is on the antenna control system for a typical radio telescope. Figure 3.4 shows the structure of the antenna.

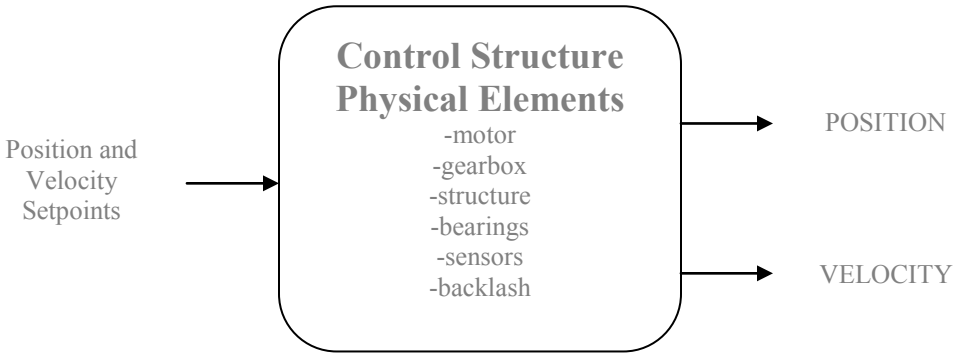


Figure 3.4: Antenna control structure

The elements of the control structure are shown in Figure 3.5. The closed loop control system receives as input the set points for the position or velocity of the antenna.

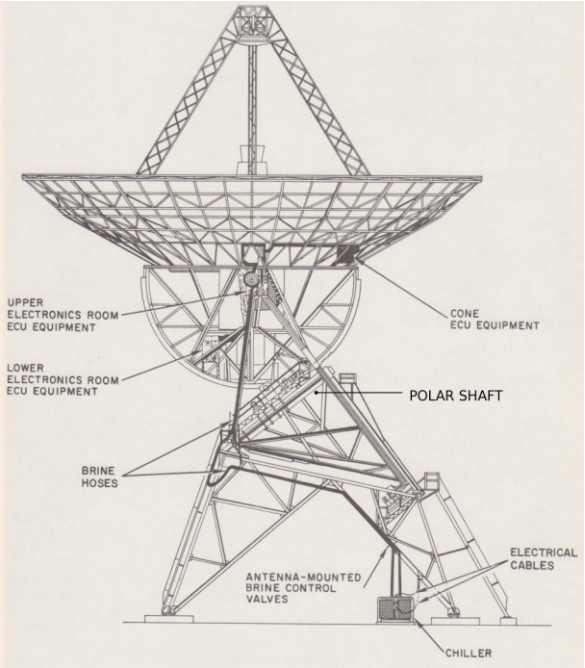


Figure 3.5: **Antenna control elements**

The dish antenna structure and dish drives are the two sub-systems that make out the model of the dish antenna. The dish drives model is influenced by the dynamics of the following dish elements:

- Antenna structure
- Drive motor
- Gearbox
- Armature
- Sensors
- Amplifiers and
- Backlash

Typically antenna structure model is derived using finite element model (Gawronski, 2008). This finite element model is incorporated as part of the model for the plant. In the thesis this model of the structure is not considered but the load of the antenna structure is incorporated using the gearbox model. The focus is on the dish drive motor incorporating the gearbox as the two have more influence on the dynamics of the antenna performance and operations. Wind disturbance is considered as part of the controller design due to its influence on the performance of the control system.

3.5 Antenna pointing accuracy

Typical radio telescopes have their pointing accuracy determined as a percentage of the primary Beam Width (BW) of the radio telescope. For point sources, tracking accuracy should be less than 10% of primary BW (Gary, 2005). For extended objects (e.g. the Sun), which covers a broader part of the primary beam, the pointing accuracy should be even better than 20% of primary BW (Gary, 2005).

The primary BW has a direct relation with the observed frequency and the diameter of the dish. For a single dish the relation is given by Equation (3.1).

$$BW = 1.22 * \frac{\lambda_a}{d} \quad \textbf{(3.1)}$$

Where: BW = Primary Beam Width

λ_a = Wavelength of the observed frequency

d = Dish diameter

For an interferometer, the relation is given by Equation (3.2), where Baseline of the interferometer is considered instead of a single dish diameter

$$BW = 1.22 * \frac{\lambda_a}{D} \quad \textbf{(3.2)}$$

Where: D = Baseline given by Equation (3.3):

$$D = \frac{N(N-1)}{2} \quad \textbf{(3.3)}$$

Where: N = Number of antennas

At high frequencies, where the primary beam is narrow, some pointing calibration is done. For very high frequency observations, antennas are pointed to a known source in the sky (close to where the antenna has to track on the sky) and any pointing offset is corrected, this is essentially pointing calibration. At low frequencies, where the beam is large and the pointing errors due to antenna servo system are small compared to the beam size, the errors due to small antenna pointing errors are also small. Gary (2005) defines antenna pointing error as “The antenna pointing error is the difference between the actual pointing position and the desired one”. The pointing error sources have a complicated directional dependence due to:

- Misalignment of antenna rotation axes
- Gravitational deformation also known as Sagging
- Atmospheric refractions
- Differential heating of the antenna structure
- The antenna encoders that are not co-located with the RF beam.

- The antenna control torques that are applied at the motor locations, while wind disturbances are distributed over the antenna surface (Gawronski, 2008).
- Model uncertainty. Manufacturing imperfections and the variable antenna configuration.
- Non-linearity's: Backlash and friction at the antenna drives, rate and acceleration limits are the main sources of nonlinearity.
- Wind loading

Most of these dependencies can be corrected for; however the main source of antenna disturbance is the wind loading (Gawronski, 2008). It requires proper control to be designed in such a way that the influence of the wind is eliminated.

The radio interferometer has one major advantage over the single dish telescope when it comes to pointing accuracy that is, it is less affected by pointing errors because position of the phase tracking centre is determined by the observatory clock, and is independent of the pointing of the individual antenna elements (Wong, 2003).
The set points for the dish control system are determined using the azimuth and altitude coordinates determined in some system of coordinate transformation.

3.6 Coordinate system

Generally people know of **coordinates of the earth** that is longitudes and latitude. Longitude of a point on earth's surface is measured east or west along the equator. It is measured as angular distance and can also be represented in time unit because earth is rotating 360^0 in 24 hours around its own axis.
Latitude of a point on earth's surface is measured north or south of the equator. It is measured as angular distance in degrees.

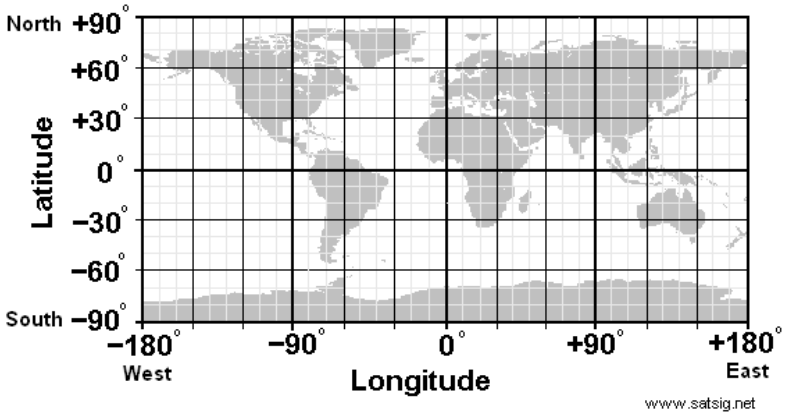


Figure 3.6: Earth longitude and latitude (http://www.satsig.net/lat_long.htm)

Earth coordinates are different to the coordinates of the star being observed by the telescope. The coordinates of the star are called **celestial equatorial coordinates**. Similar to earth coordinate, the analogue for longitude is Right Ascension in equatorial coordinates and the analogue for latitude is Declination.

Right ascension of a star is angular distance between the meridians through the first point in Ares and the star. Ares is a zero point chosen on the celestial sphere at which the Sun crosses the equator, on about March 21. This is the northern Vernal (spring) Equinox, and the southern hemisphere Autumnal equinox, see Figure (3.7).

Declination of a star is angular distance measured in degrees between the celestial equator (0^0) and the star. The north celestial pole is at positive ninety degrees and the south celestial pole is at negative ninety degrees.

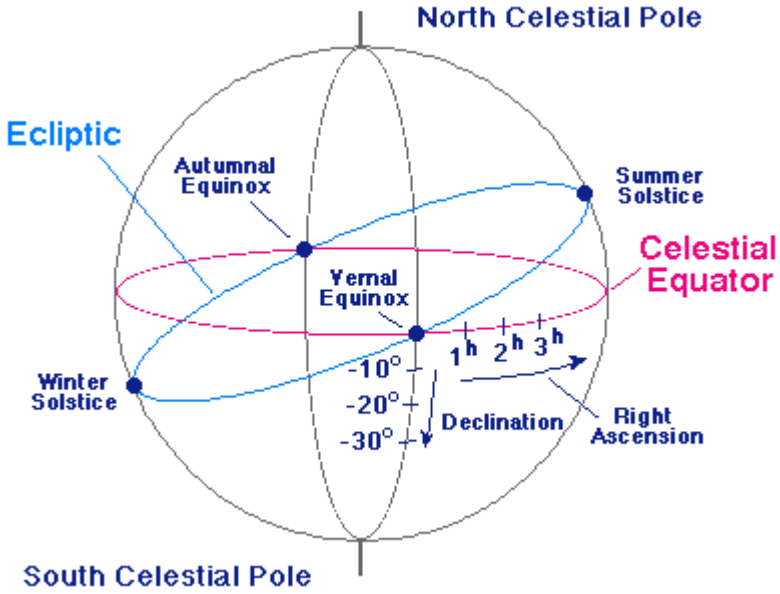


Figure 3.7: Celestial equatorial coordinates
http://csep10.phys.utk.edu/astr161/lect/time/coordinates.html

Horizontal coordinate also called alt-az system is used to convert from earth coordinate system (latitude longitude) to celestial equatorial coordinate system (right ascension declination) using observer's local horizon as the fundamental plane as shown in Figure (3.8).

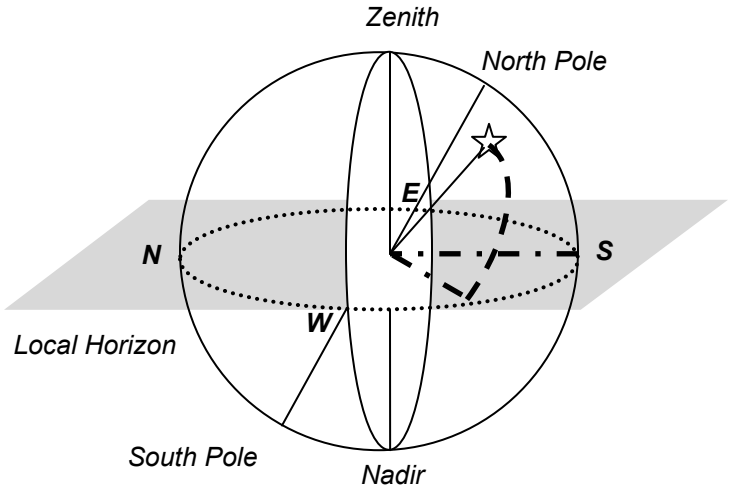


Figure 3.8: Horizontal coordinates
Azimuth from the North point also from the South point toward the West (round dots), Altitude, (dash dot)
(adapted from) http://en.wikipedia.org/wiki/Horizontal_coordinate_system)

Zenith is the great circle that cuts through the north celestial pole. Nadir is the great circle that cuts through the south celestial pole (Gaylard, 2009).

The following steps are adapted from Ephemeris Almanac (http://www.ephemeris.com/space-time.html) and are used in the thesis to convert from Right Ascension and Declination (celestial coordinates) to Azimuth and Altitude (earth coordinates) using horizontal coordinate transformation:

- Determine the Greenwich Sidereal Time (GST)
- Convert GST to Local Sidereal Time (LST)
- Convert right ascension and declination to azimuth and altitude at a given latitude and sidereal time using Equations (3.4) and (3.5).

$$az = \arctan\left(\frac{-(\sin(H)\cos(\delta)}{\cos(\varphi)\sin(\delta)-\sin(\varphi)\cos(\delta)\cos(H)}\right) \quad \textbf{(3.4)}$$

$$alt = \arcsin(\sin(\varphi)\sin(\delta)+\cos(\varphi)\cos(\delta)\cos(H)) \quad \textbf{(3.5)}$$

Where: H = Object's hour angle

Λ = Object's right ascension

δ = Object's declination

φ = Object's latitude

Π = Object's longitude

Az = Object's azimuth

alt = Object's altitude

Hour angle is an angular distance measured east or west of the observer's zenith meridian. Hour angle is given by Equation (3.6)

$$H = LST - \Lambda \qquad (3.6)$$

3.7 Simulation of the horizontal coordinate system

The horizontal coordinates are used to point the dish. As the position of the star (celestial object) is given using the celestial coordinates, a procedure is necessary to be developed for calculation of the respective horizontal coordinates. The requirements towards this procedure are that it has to be capable to quickly change the initial data and to perform the calculations in real-time. In addition it has to give the results from the calculation in time domain, separately to the azimuth and altitude and to calculate the trajectory of the star (celestial object) in time domain. These trajectories are used as desired ones for the corresponding motors. LabVIEW™ software real-time environment is selected to implement the above requirements.

Equation (3.4) and Equation (3.5) are implemented using LabVIEW™. The results are verified by using Radio Eyes (http://www.radiosky.com/radioeyesishere.html) a commercial of the shelf software used by astronomers to look for objects in the sky. The LabVIEW block diagram is for implementation of the horizontal coordinate system is shown in Figure (3.9). The results are show below, Figure (3.10).

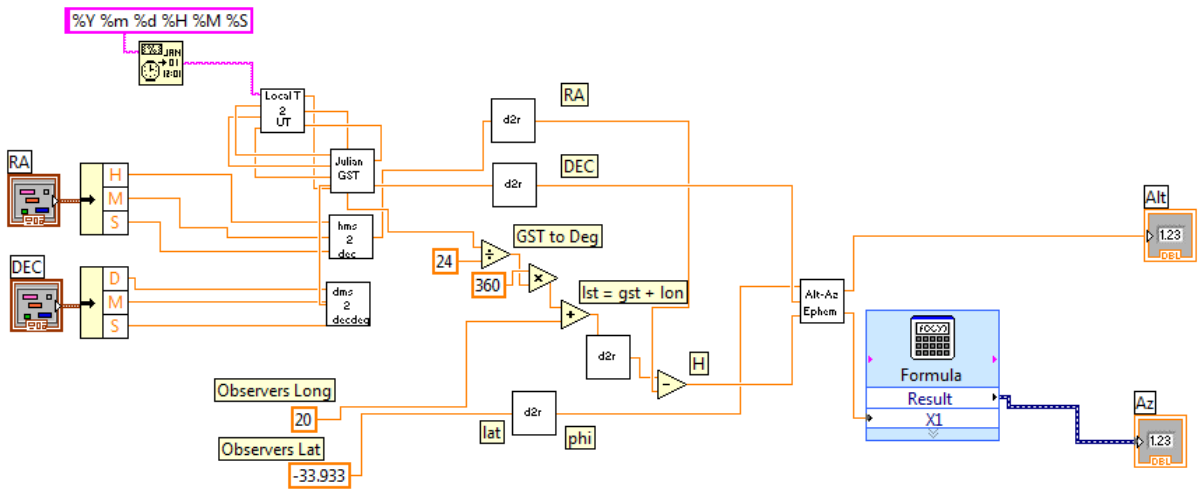


Figure 3.9: Block Diagram for Horizontal coordinate system using LabVIEW™

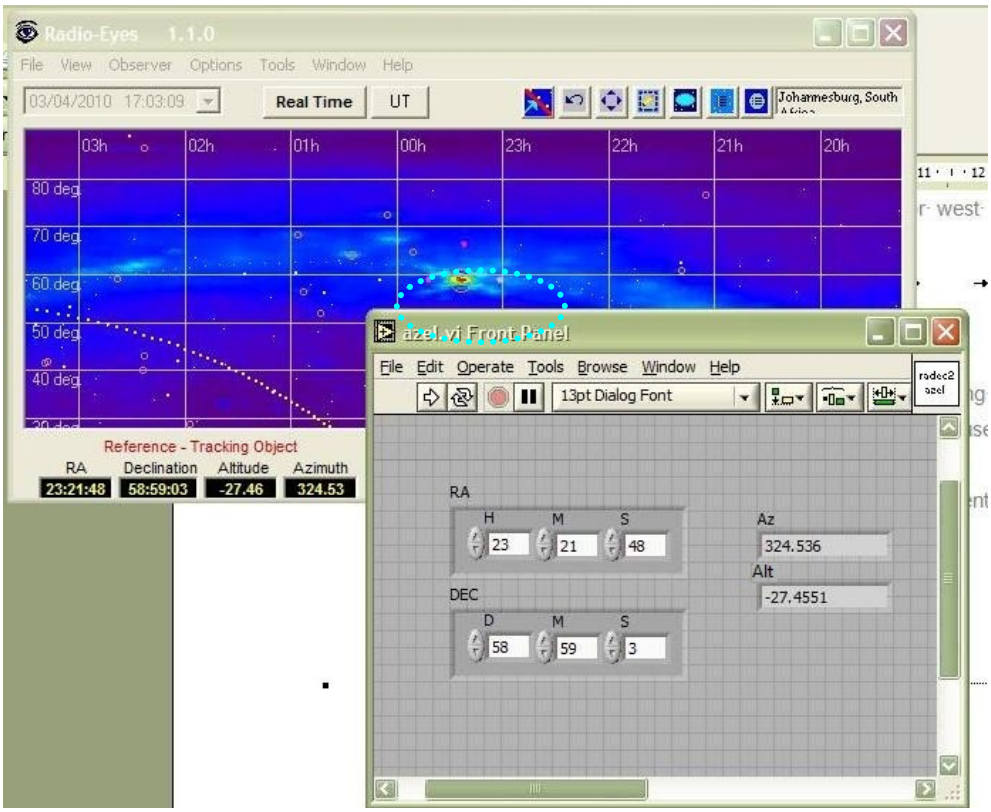


Figure 3.10: **Horizontal coordinate system using LabVIEW™ and Radio Eyes**

3.8 Conclusion

The background of Radio Telescope as used in Radio Astronomy is described and considering the different types of telescope and the pointing accuracy as a performance measure for the closed control of a Radio Telescope has been presented.

Horizontal coordinate system has been identified as the coordinate transformer to be used when converting from celestial equatorial coordinate (Right Ascension and Declination) to earth coordinate system (Longitude and Latitude). Horizontal coordinate transformer software module is developed in LabVIEW™ using equations adapted from Ephemeris (Almanac) and the results are verified against Radio Eyes, a commercial of the shelf software used by radio astronomers when looking at objects in the sky.

The antenna receives as inputs the set point for the position of the antenna (Azimuth and Altitude) from the output of the coordinate transformer in time domain. The antenna structure is not modelled using finite element model, instead in the thesis the load of the antenna structure is incorporated using gear box model. Wind has been identified as the source of antenna disturbance as a result in the thesis during controller design wind has been incorporated in the closed loop control system in order for the controller to eliminate the influence introduced by the wind.

In order to design the controller for the plant a mathematical model is needed. The model is derived in Chapter four in continuous form and later converted to discrete form for design of the controller, design of the filter, and real-time implementation.

CHAPTER FOUR
MATHEMATICAL DESCRIPTION OF THE RADIO TELESCOPE SYSTEM

4.1 Introduction

A radio telescope has different mount configurations when observing objects in the sky. There is Equatorial mount, where the radio telescope has one motor driving the azimuth shaft and altitude is fixed to the equator level. The second mount configuration is mostly referred to as altitude and azimuth (alt-az) mount. This mount has two motors one driving the azimuth position and the other driving the altitude position. Alt-Az mount is considered in the thesis.

Traditionally telescope controllers are designed such that each axis (azimuth and altitude) is actuated independently by its servo driver and the servo drives are controlled by the facility controller. Figure (4.1) shows the structure of a typical telescope closed loop control system.

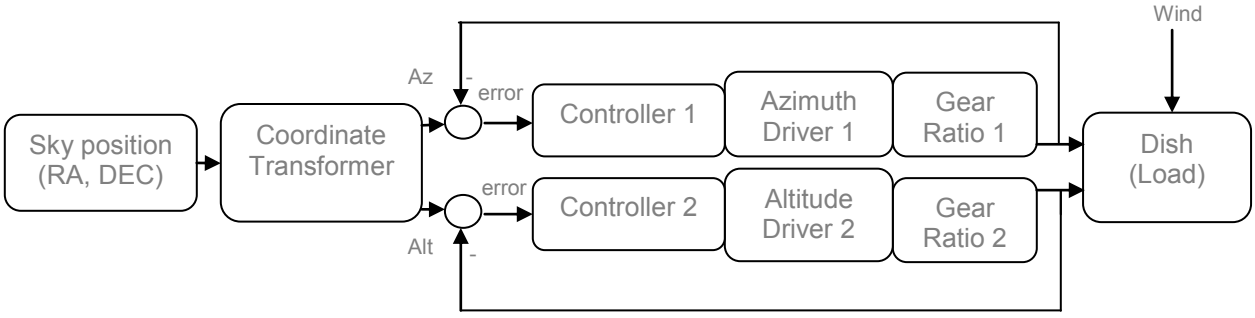


Figure 4.1: Typical telescope control structure

The position in the sky where the dish must be driven to is given by the desired Right Ascension (RA) and Declination (DEC) coordinates. The right ascension and declination are transformed by the coordinate transformer to azimuth and altitude. The azimuth and altitude are used as set points for the controllers.

By definition process models are defined as different order parameter models (Ljung, 1987), and are determined by using laws of physics. The aim of this chapter is to derive the mathematical models of the servo and stepper motors driving the antenna structure paying attention to the load introduced by the antenna structure.

The developments in the chapter are as follows:

- Derivation of the servo motor mathematical model is described in section 4.2.
- State space description of the developed mathematical model is described in section 4.3.
- Derivation of the dish model is described in section 4.5.

- Derivations of the full model is described is section 4.6.
- Derivation of the stepper motor mathematical model is described in section 4.3.

4.2 Development of the mathematical model of the radio telescope system based on a servo motor.

4.2.1 Derivation of the servo motor model

The process model in the thesis is defined as a second order parameter model, determined by using Newton and Kirchhoff's laws of physics. The model is derived for a single motor, but the derivation is the same for the second motor, what is specific are the parameters for each motor as shown in Table (4.1) below. Each motor has its own values for all the parameters.

The general structure of the DC motor with the gear, the load, and the wind disturbance torque is shown in Figure (4.2) (Gawronski, 2008). The definitions of the notations in the figure are given in Table (4.1).

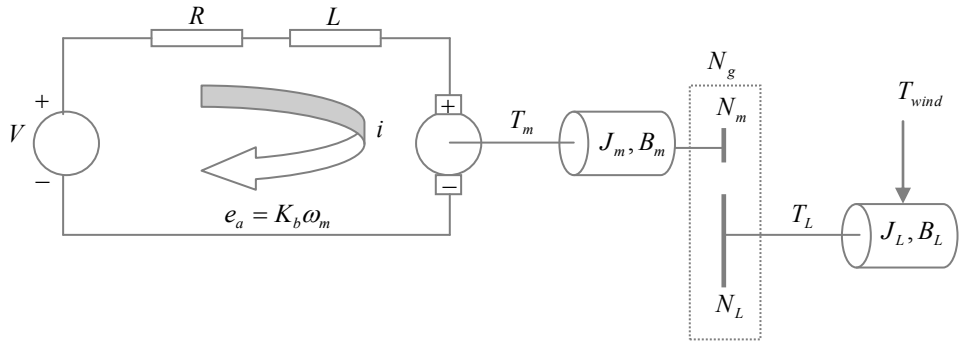


Figure 4.2: Physical architecture of the DC motor

The motors are modeled using Kirchhoff and Newton's law.

Table 4.3: General motor parameters

Symbols /Letters	Definition/Explanation	Altitude Values	Azimuth Values	Units
V	Input Voltage	100	100	Volts
K_t	Motor toque constant	2	2	none
R	Electrical resistance	0.4	0.4	Ohms
L	Electric inductance	0	0	H
K_b	Electromotive force constant	2	2	none
J_m	Motor moment of inertia	8	5	kg/m^2
J_L	Load moment of inertia	1200	700	kg/m^2
B_m	Motor damping coefficient	3.5	2	Ns/m

B_L	Load damping coefficient	1300	800	Ns/m
N_m	Number of teeth of the motor gear	100	100	none
N_L	Number of teeth of the load gear	1000	1000	none
D	Dish diameter	12	12	m
c_t	Dimensionless torque coefficient	-0.05 to 0.25	-0.05 to 0.25	none
α_p	Static air density	0.6126	0.6126	Ns^2/m^4

Using Kirchhoff laws and considering Figure (4.2) above the following equation connecting the voltage and current in the armature can be written:

$$V - e_a = Ri + L\frac{di}{dt} \quad \textbf{(4.1)}$$

The relation between back emf and angular velocity of the motor is through the electromotive force constant (K_b)

$$e_a = K_b \omega_m \quad \textbf{(4.2)}$$

The angular velocity can be described as first derivative of position of the shaft that is $\omega_m = \dot{\theta}_m$.

Representing Equation (4.1) and (4.2) in Laplace domain it is obtained that

$$V(s) - E_a(s) = RI(s) + LsI(s) \quad \textbf{(4.3)}$$

$$E_a(s) = K_b \omega_m(s) = K_b s\theta_m(s) \quad \textbf{(4.4)}$$

Substituting Equation (4.4) in Equation (4.3)

$$V(s) - K_b s\theta_m(s) = I(s)(R + Ls) \quad \textbf{(4.5)}$$

Solving for $I(s)$ in Equation (4.5)

$$I(s) = \frac{V(s) - K_b s\theta_m(s)}{R + Ls} \quad \textbf{(4.6)}$$

Equation (4.6) represents the first equation of the model derivation using Kirchhoff's Law. Newton's Law for the motor with rotating load through the gear ratio as illustrated in Figure (4.2).

Using Newton's laws the toque of the motor is represented as:

$$T_m = J_T \dot{\omega}_m + B_T \omega_m \quad \textbf{(4.7)}$$

The motor toque (T_m) has a relation with armature current (i) through a motor torque constant (K_t)

$$T_m = K_t i \quad \textbf{(4.8)}$$

Representing Equation (4.7) and (4.8) in Laplace domain it is obtained that

$$T_m(s) = J_m s\omega_m(s) + B_m\omega_m(s) \quad \textbf{(4.9)}$$

$$T_m(s) = K_t I(s) \quad \textbf{(4.10)}$$

Substituting Equation (4.10) in Equation (4.9)

$$K_t I(s) = J_m s\omega_m(s) + B_m\omega_m(s) \quad \textbf{(4.11)}$$

Expressing ω_m as $\dot{\theta}_m$ in Equation (4.11)

$$K_t I(s) = J_T s^2\theta_m(s) + B_T s\theta_m(s) \quad \textbf{(4.12)}$$

Substituting Equation (4.6) in Equation (4.12)

$$K_t\left(\frac{V(s) - K_b s\theta_m(s)}{R + Ls}\right) = J_m s^2\theta_m(s) + B_m s\theta_m(s) \quad \textbf{(4.13)}$$

The transfer function of the motor $\frac{\theta_m(s)}{V(s)}$ is derived from Equation (4.13)

$$K_t\left[V(s) - K_b s\theta_m(s)\right] = \left[R + Ls\right]\left[J_m s^2\theta_m(s) + B_m s\theta_m(s)\right] \quad \textbf{(4.14)}$$

$$K_t V(s) - K_t K_b s\theta_m(s) = RJ_m s^2\theta_m(s) + RB_m s\theta_m(s) + LJ_m s^3\theta_m(s) + LB_m s^2\theta_m(s) \quad \textbf{(4.15)}$$

$$K_t V(s) = \left[LJ_m s^3 + \left(RJ_m + LB_m\right)s^2 + \left(RB_m + K_t K_b\right)s\right]\theta_m(s) \quad \textbf{(4.16)}$$

$$G_m(s) = \frac{\theta_m(s)}{V(s)} = \frac{K_t}{LJ_m s^3 + \left(RJ_m + LB_m\right)s^2 + \left(RB_m + K_t K_b\right)s} \quad \textbf{(4.17)}$$

$$G_m(s) = \frac{\theta_m(s)}{V(s)} = \frac{K_t}{s\left(LJ_m s^2 + \left(RJ_m + LB_m\right)s + \left(RB_m + K_t K_b\right)\right)} \quad \textbf{(4.18)}$$

Assuming that $L \approx 0$, then the model can be reduced to:

$$G_m(s) = \frac{\theta_m(s)}{V(s)} = \frac{K_t}{s\left(RJ_m s + RB_m + K_t K_b\right)} \quad \textbf{(4.19)}$$

$$G_m(s) = \frac{\theta_m(s)}{V(s)} = \frac{K_t \big/ RJ_m}{s\left(s + \dfrac{RB_m + K_t K_b}{RJ_m}\right)} \quad \textbf{(4.20)}$$

Due to the complexity of the numerator and denominator terms of the derived transfer functions, new variables are used to represent the two complex terms. K is used for the upper term and α is used to represent the second term of the denominator resulting in a

simplified transfer function of the motor as shown in Equation (4.21). It is assumed that K_t and K_b are equal and k^2 is used to represent the two constants.

$$\frac{\theta_m(s)}{V(s)} = \frac{K}{s(s+\alpha)} \tag{4.21}$$

where: $K = \frac{K_t}{RJ_m}$ **(4.22)**

And: $\alpha = \frac{RB_T + k^2}{RJ_m}$ **(4.23)**

This model represents open loop transfer function of the Altitude motor without load and disturbance. Block diagram of the motor without load and wind disturbance is shown in Figure (4.3).

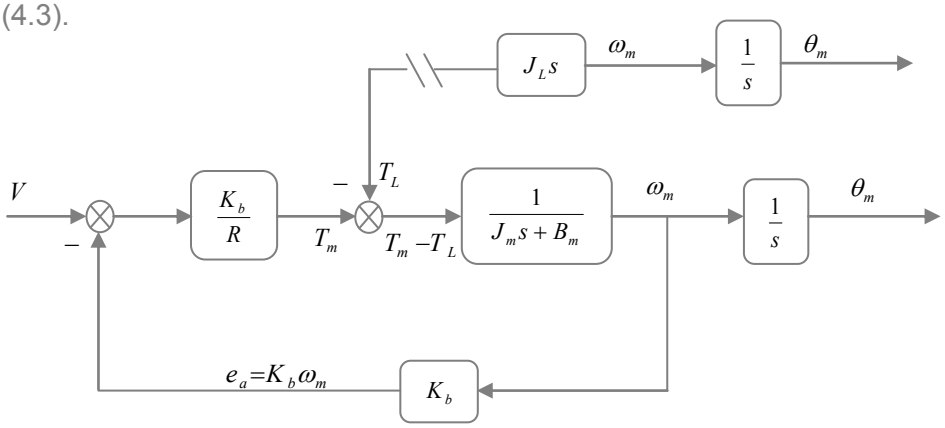


Figure 4.3: Block diagram representation of the DC motor without load and disturbance

4.2.2 Servo motor model description in state space form

For ease of calculations and simulation it is necessary to convert the transfer function Equation (4.21) of the plant into state space form.

Rewriting Equation (4.21) with respect to angular position.

$$\theta_m(s) = \frac{K}{s(s+\alpha)} * V(s) \tag{4.24}$$

$$s^2\theta_m(s) + s\theta_m(s)\alpha = KV(s) \tag{4.25}$$

The representation of Equation (4.25) in time domain under zero initial conditions is:

$$\ddot{\theta}_m = -\dot{\theta}_m\alpha + KV \tag{4.26}$$

Equation (4.26) is used to derive the state space model.

The output of the model → $y = \theta_m$

Control input → $u = V$

State space vector $X = [X_1, X_2] = [\theta_m, \dot{\theta}_m]^T$

Introduction of the state space vector components

$y = \theta_m = X_1 \rightarrow$ First component of the state space vector

$\dot{y} = \dot{\theta}_m = \dot{X}_1 = X_2 \rightarrow$ Second component of the state space vector

$\ddot{y} = \ddot{\theta}_m = \ddot{X}_1 = \dot{X}_2 =$ Equation (4.26)

Using the above transformations the state space model of the motor is:

$$\dot{X}_1 = X_2 \quad \textbf{(4.27)}$$

$$\dot{X}_2 = \ddot{y} = -\alpha X_2 + Ku \quad \textbf{(4.28)}$$

$$y = X_1 \quad \textbf{(4.29)}$$

In matrix format:

$$\begin{bmatrix} \dot{X}_1 \\ \dot{X}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -\alpha \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} + \begin{bmatrix} 0 \\ K \end{bmatrix} u \quad \textbf{(4.30)}$$

$$y = \theta_m = X_1 = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \quad \textbf{(4.31)}$$

In common notation the model can be expressed in the following form:

$$\dot{x}(t) = Ax(t) + Bu(t) \quad \textbf{(4.32)}$$

$$y(t) = Cx(t) \quad \textbf{(4.33)}$$

where:

$$A = \begin{bmatrix} 0 & 1 \\ 0 & -\alpha \end{bmatrix}, B = \begin{bmatrix} 0 \\ K \end{bmatrix}, C = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

4.2.3 Simulation of open loop step response of the motor model in a simple form

A step response is obtained using Matlab and Simulink software environment, see Figure (4.4). The results show that the dynamic behavior of the position of the motor is of a second order integrating system.

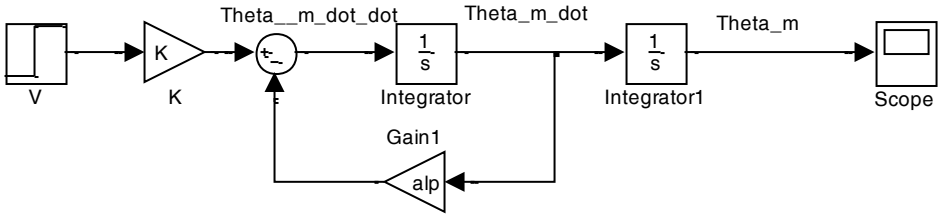


Figure 4.4: Motor model in State Space form

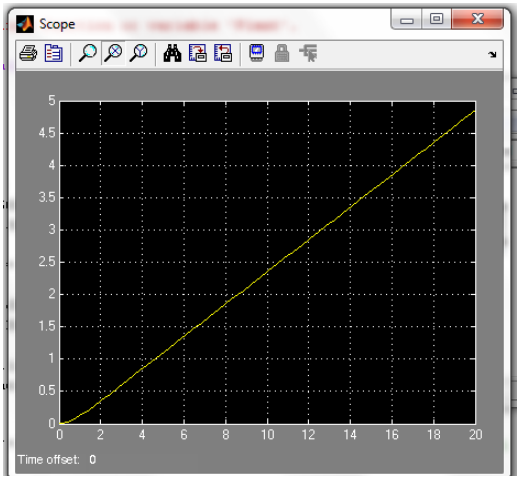


Figure 4.5: Step response for the state space model of the servo motor

4.2.4 Derivation of the dish model incorporating the model of the motor

4.2.4.1 Antenna model

The motor is considered for the case of a load of the antenna structure and of the wind disturbance. The block diagram representing the system - motor, antenna, and wind disturbance is shown in Figure (4.6) where T_w is the wind torque disturbing the motor and antenna.

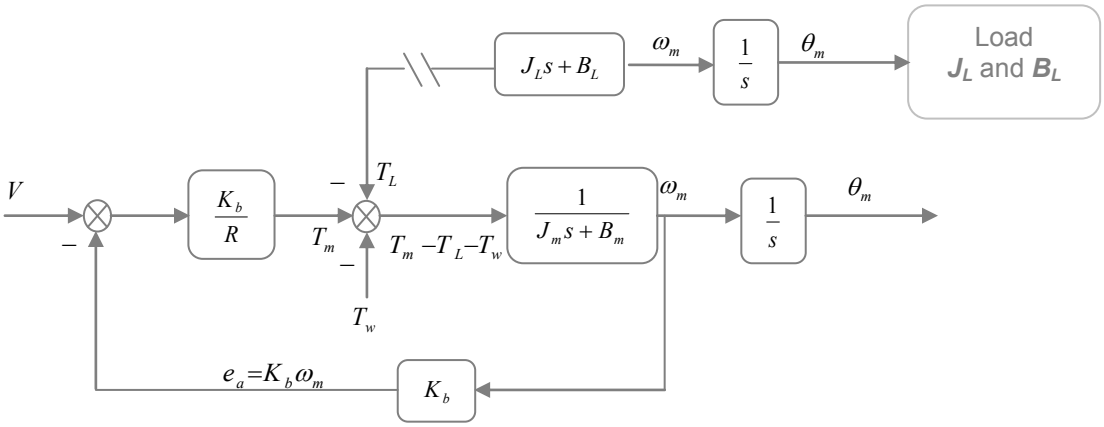


Figure 4.6: Block diagram representation of the DC motor with disturbance and load

A rigid antenna in the open loop configuration has torque input and velocity output. The relationship between the torque, damping coefficient moment of inertia and angular velocity follows from the Newton inertia law (Gawronski, 2008).

$$T_L = J_L \ddot{\theta}_m + B_L \dot{\theta}_m \qquad \textbf{(4.34)}$$

The motor overcomes the torque T_L in order to drive the antenna. The transfer function of the load may be represented in Laplace domain as shown in Figure (4.7).

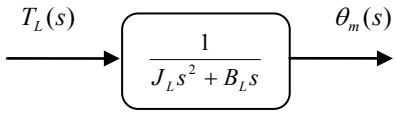


Figure 4.7: Model of the dish (load)

$$T_L(s) = \left(J_L s^2 + B_L s\right)\theta_m(s) \tag{4.35}$$

$$\frac{\theta_m(s)}{T_L(s)} = \frac{1}{J_L s^2 + B_L s} \tag{4.36}$$

The next step is to derive the model of the dish with a single motor by incorporating the model of the motor and wind disturbance. Input voltage of the motor V is considered as input to this mathematical model and angular position of the dish is considered to be equal to the angular position of the motor, representing the output of the model θ_m. This model is shown in Figure (4.8).

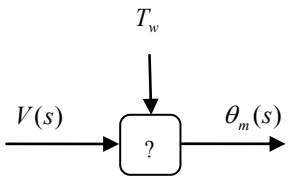


Figure 4.8: Model of the dish (load), motor and wind disturbance

4.2.4.2 Derivation of the wind disturbance model

The model of the steady state wind torque T_w acting on the drives is derived in the section below (Gawronski, 2008)

$$T_{wind} = k_T \upsilon_m{}^2 \tag{4.37}$$

where: k_T = Quadratic law coefficient

υ_m = Steady state wind speed in km/h

The wind torque depends on the wind pressure and pressure-torque relationship and is given by Equation (4.38). It is determined based on wind tunnel experiments (Gawronski, 2008).

$$T_{wind} = c_t A d p_n \tag{4.38}$$

where: c_t = Dimensionless torque coefficient

A = Antenna dish surface area

d = Antenna dish diameter

p_n = Dynamic pressure of wind

The dynamic wind pressure depends on the wind velocity in the same way as the wind torque.

$$p_n = \alpha_p \upsilon_m^{\,2} \tag{4.39}$$

where: α_p = Static air density

Substituting Equation (4.38) into Equation (4.39) yields Equation (4.40)

$$k_T = c_t \alpha_p A d = c_t \alpha_p \frac{\pi d^3}{4} \tag{4.40}$$

Substituting Equation (4.40) into Equation (4.37) yields Equation (4.41)

$$T_{wind} = c_t \alpha_p \frac{\pi d^3}{4} \upsilon_m^{\,2} \tag{4.41}$$

4.2.4.3 Derivation of the mathematical model of the dish and the motor under the influence of the wind disturbance

Representing moment of inertia and damping coefficients using the load torque results in increasing of the moment of inertia and damping coefficient of the system consisting of the motor and load. The total inertia J_T and damping coefficient B_T are as follows:

$$J_T = J_m + J_L \left(\frac{N_m}{N_L}\right)^2 = J_m + J_L N_g^{\,2} \tag{4.42}$$

$$B_T = B_m + B_L \left(\frac{N_m}{N_L}\right)^2 = B_m + B_L N_g^{\,2} \tag{4.43}$$

where $\frac{N_m}{N_L}$ is the gear ratio represented by N_g. Following the diagram in Figure (4.6) and adding the wind torque disturbing the motor torque, the results can be written:

$$T_m - T_w = \left(J_m + J_L N_g^{\,2}\right)\ddot{\theta}_m + \left(B_m + B_L N_g^{\,2}\right)\dot{\theta}_m \tag{4.44}$$

$$T_m - T_w - J_L N_g^{\,2}\ddot{\theta}_m - B_L N_g^{\,2}\dot{\theta}_m = J_m\ddot{\theta}_m + B_m\dot{\theta}_m \tag{4.45}$$

but $J_L N_g^{\,2}\ddot{\theta}_m + B_L N_g^{\,2}\dot{\theta}_m = T_L$, then (4.46)

$$T_m - T_w - T_L = J_m\ddot{\theta}_m + B_m\dot{\theta}_m \tag{4.47}$$

Expressing Equation (4.47) in terms of T_L gives:

$$T_L = T_m - T_w - J_m\ddot{\theta}_m - B_m\dot{\theta}_m \tag{4.48}$$

Substituting Equation (4.8) in Equation (4.48)

$$T_L = K_t i - T_w - J_m \ddot{\theta}_m - B_m \dot{\theta}_m \quad \textbf{(4.49)}$$

In Laplace domain Equation (4.46) is:

$$T_L = K_t I(s) - T_w(s) - J_m s^2 \theta_m - B_m s \theta_m \quad \textbf{(4.50)}$$

Substituting Equation (4.6) in Equation (4.50)

$$T_L = K_t \left(\frac{V(s) - K_b s \theta_m(s)}{R + Ls} \right) - T_w(s) - J_m s^2 \theta_m - B_m s \theta_m \quad \textbf{(4.51)}$$

$$(R + Ls) T_L = K_t \left(V(s) - K_b s \theta_m(s) \right) - T_w(s)(R + Ls) - J_m s^2 \theta_m (R + Ls) - B_m s \theta_m (R + Ls) \quad \textbf{(4.52)}$$

Substituting in Equation (4.52) the Equation (4.46) to obtain the dependence between $\theta_m(t)$ and $V(t)$

$$(R + Ls)\left(J_L N_g{}^2 s^2 \theta_m + B_L N_g{}^2 s \theta_m \right) = K_t \left(V(s) - K_b s \theta_m(s) \right) - T_w(s)(R + Ls) - J_m s^2 \theta_m (R + Ls) - B_m s \theta_m (R + Ls) \quad \textbf{(4.53)}$$

Transformation of Equation (4.53) to time domain

$$RJ_L N_g{}^2 \ddot{\theta}_m + RB_L N_g{}^2 \dot{\theta}_m + LJ_L N_g{}^2 \dddot{\theta}_m + LB_L N_g{}^2 \ddot{\theta}_m = K_t V - K_t K_b \dot{\theta}_m - RT_w - RJ_m \ddot{\theta}_m - LJ_m \dddot{\theta}_m - RB_m \dot{\theta}_m - LB_m \ddot{\theta}_m \quad \textbf{(4.54)}$$

Expression of $\dddot{\theta}_m$

$$\left(LJ_L N_g{}^2 + LJ_m \right) \dddot{\theta}_m = -\left(RJ_L N_g{}^2 + LB_L N_g{}^2 + RJ_m + LB_m \right) \ddot{\theta}_m - \left(RB_L N_g{}^2 + K_t K_b + RB_m \right) \dot{\theta}_m + K_t V - RT_w \quad \textbf{(4.55)}$$

$$\dddot{\theta}_m = -\left(\frac{RJ_L N_g{}^2 + LB_L N_g{}^2 + RJ_m + LB_m}{LJ_L N_g{}^2 + LJ_m} \right) \ddot{\theta}_m - \left(\frac{RB_L N_g{}^2 + K_t K_b + RB_m}{LJ_L N_g{}^2 + LJ_m} \right) \dot{\theta}_m + \frac{K_t V}{LJ_L N_g{}^2 + LJ_m} - \frac{RT_w}{LJ_L N_g{}^2 + LJ} \quad \textbf{(4.56)}$$

Equation (4.56) is simplified by representing coefficients of angular position, input voltage and wind disturbance with coefficients P_1, P_2, P_3, P_4.

where $P_1 = \dfrac{RJ_L N_g{}^2 + LB_L N_g{}^2 + RJ_m + LB_m}{LJ_L N_g{}^2 + LJ_m}$ **(4.57)**

$$P_2 = \frac{RB_L N_g{}^2 + K_t K_b + RB_m}{LJ_L N_g{}^2 + LJ_m} \quad \textbf{(4.58)}$$

$$P_3 = \frac{K_t}{LJ_L N_g{}^2 + LJ_m} \quad \textbf{(4.59)}$$

$$P_4 = \frac{R}{LJ_L N_g{}^2 + LJ} \quad \textbf{(4.60)}$$

$$\ddot{\theta}_m = -P_1\ddot{\theta}_m - P_2\dot{\theta}_m + P_3 V - P_4 T_w \quad \textbf{(4.61)}$$

Equation (4.61) is used to derive the state space model of the structure.

The output of the model → $y = \theta_m$

Control input → $u = V$

State space vector $X = [X_1, X_2, X_3] = [\theta_m, \dot{\theta}_m, \ddot{\theta}_m]^T$

Introduction of the state space vector components

$y = \theta_m = X_1$ → First component

$\dot{y} = \dot{\theta}_m = \dot{X}_1 = X_2$

$\ddot{y} = \ddot{\theta}_m = \ddot{X}_1 = \dot{X}_2 = X_3$

$\dddot{y} = \dddot{\theta}_m = \dddot{X}_1 = \ddot{X}_2 = \dot{X}_3$ = Equation (4.61)

The state space equations are:

$$\dot{X}_1 = X_2 \quad \textbf{(4.62)}$$

$$\dot{X}_2 = X_3 \quad \textbf{(4.63)}$$

$$\dot{X}_3 = -P_1\ddot{\theta}_m - P_2\dot{\theta}_m + P_3 V - P_4 T_w \quad \textbf{(4.64)}$$

In matrix format:

$$\begin{bmatrix} \dot{X}_1 \\ \dot{X}_2 \\ \dot{X}_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & -P_2 & -P_1 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ P_3 \end{bmatrix} u + \begin{bmatrix} 0 \\ 0 \\ -P_4 \end{bmatrix} T_w, u = V \quad \textbf{(4.65)}$$

$$y = \theta_m = X_1 \quad \textbf{(4.66)}$$

$$\dot{x}(t) = Ax(t) + Bu(t) + WT_w \quad \textbf{(4.67)}$$

$$y(t) = Cx(t) \quad \textbf{(4.68)}$$

where:

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & -P_2 & -P_1 \end{bmatrix}, B = \begin{bmatrix} 0 \\ 0 \\ P_3 \end{bmatrix}, W = \begin{bmatrix} 0 \\ 0 \\ -P_4 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$

Assuming that $L \approx 0$, then Equation (4.54) can be reduced to:

$$RJ_L N_g{}^2\ddot{\theta}_m + RB_L N_g{}^2\dot{\theta}_m = K_t V - K_t K_b\dot{\theta}_m - RT_w - RJ_m\ddot{\theta}_m - RB_m\dot{\theta}_m \quad \textbf{(4.69)}$$

Expression of $\ddot{\theta}_m$

$$\left(RJ_L N_g{}^2 + RJ_m\right)\ddot{\theta}_m = -\left(RB_L N_g{}^2 + K_t K_b + RB_m\right)\dot{\theta}_m + K_t V - RT_w \quad \textbf{(4.70}$$

$$\ddot{\theta}_m = -\left(\frac{RB_L N_g{}^2 + K_t K_b + RB_m}{RJ_L N_g{}^2 + RJ_m}\right)\dot{\theta}_m + \left(\frac{K_t}{RJ_L N_g{}^2 + RJ_m}\right)V - \left(\frac{R}{RJ_L N_g{}^2 + RJ_m}\right)T_w \quad \textbf{(4.71)}$$

Equation (4.71) is simplified by representing coefficients of the angular position, input voltage and wind disturbance with coefficients Q_1, Q_2, Q_3.

where $Q_1 = \dfrac{RB_L N_g{}^2 + K_t K_b + RB_m}{RJ_L N_g{}^2 + RJ_m}$ **(4.72)**

$$Q_2 = \frac{K_t}{RJ_L N_g{}^2 + RJ_m} \quad \textbf{(4.73)}$$

$$Q_3 = \frac{R}{RJ_L N_g{}^2 + RJ_m} \quad \textbf{(4.74)}$$

$$\ddot{\theta}_m = -Q_1\dot{\theta}_m + Q_2 V - Q_3 T_w \quad \textbf{(4.75)}$$

Equation (4.75) is used to derive the state space model of the structure.

The output of the model → $y = \theta_m$

Control input → $u = V$

State space vector $X = [X_1, X_2]^T = [\theta_m, \dot{\theta}_m]^T$

Introduction of the state space vector components

$y = \theta_m = X_1$ → First component

$\dot{y} = \dot{\theta}_m = \dot{X}_1 = X_2$

$\ddot{y} = \ddot{\theta}_m = \ddot{X}_1 = \dot{X}_2$ = Equation (4.75)

$$\dot{X}_1 = X_2 \quad \textbf{(4.76)}$$

$$\dot{X}_2 = \ddot{y} = -Q_1 X_2 + Q_2 u - Q_3 T_w \quad \textbf{(4.77)}$$

In matrix format:

$$\begin{bmatrix}\dot{X}_1 \\ \dot{X}_2\end{bmatrix} = \begin{bmatrix}0 & 1 \\ 0 & -Q_1\end{bmatrix}\begin{bmatrix}X_1 \\ X_2\end{bmatrix} + \begin{bmatrix}0 \\ Q_2\end{bmatrix}u + \begin{bmatrix}0 \\ -Q_3\end{bmatrix}T_w \quad \textbf{(4.78)}$$

$$y = \theta_m = X_1 = \begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}X_1 \\ X_2\end{bmatrix} \quad \textbf{(4.79)}$$

$$\dot{x}(t) = Ax(t) + Bu(t) + WT_w \quad \textbf{(4.80)}$$

$$y(t) = Cx(t) \tag{4.81}$$

where:

$$A = \begin{bmatrix} 0 & 1 \\ 0 & -Q_1 \end{bmatrix}, B = \begin{bmatrix} 0 \\ Q_2 \end{bmatrix}, W = \begin{bmatrix} 0 \\ -Q_3 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

4.2.4.4 Simulation of the model of the dish

A step response is obtained using Matlab and Simulink software environment, see Figure (4.10). The results show that the dynamic behavior of the position of the motor is of a second order integrating system.

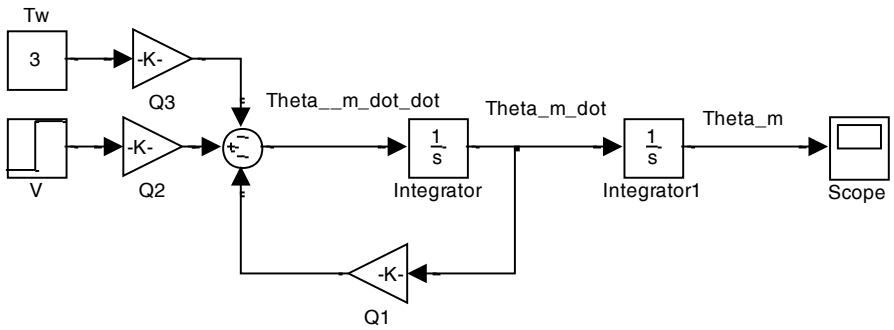


Figure 4.9: Motor model in State Space form with wind disturbance and load

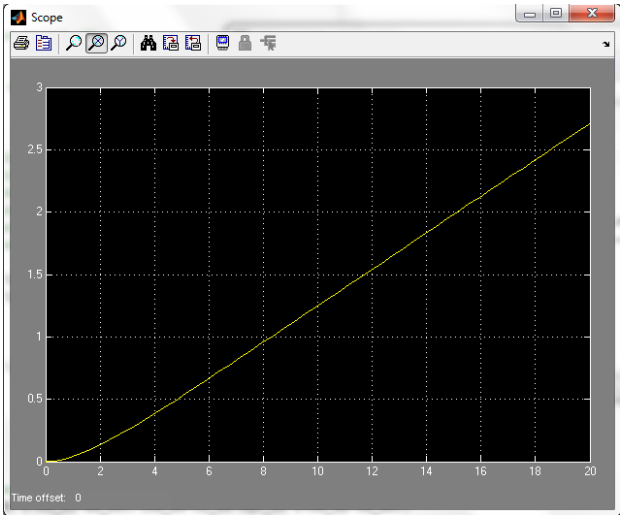


Figure 4.10: Step response for the servo motor model with wind disturbance and load

4.2.5 Derivation of the full model of the dish

The reduced model derived in the equations above is for a single motor. This reduced state space model is used to model the azimuth and altitude motors. The coefficients of the state vector Q_1, control input Q_2 and wind disturbance Q_3, are represented in terms of azimuth and altitude parameters. The new coefficients are:

$$Q_{1az} = \frac{R_{az}B_{Laz}N_{gaz}{}^2 + K_{taz}K_{baz} + R_{az}B_{maz}}{R_{az}J_{Laz}N_{gaz}{}^2 + R_{az}J_{maz}} \quad \textbf{(4.82)}$$

$$Q_{2az} = \frac{K_{taz}}{R_{az}J_{Laz}N_{gaz}{}^2 + R_{az}J_{maz}} \quad \textbf{(4.83)}$$

$$Q_{3az} = \frac{R_{az}}{R_{az}J_{Laz}N_{gaz}{}^2 + R_{az}J_{maz}} \quad \textbf{(4.84)}$$

$$\ddot{\theta}_{maz} = -Q_{az1}\dot{\theta}_{maz} + Q_{2az}V_{az} - Q_{3az}T_{waz} \quad \textbf{(4.85)}$$

Altitudes drive dish parameters:

where $Q_{1alt} = \dfrac{R_{alt}B_{Lalt}N_{galt}{}^2 + K_{talt}K_{balt} + R_{alt}B_{malt}}{R_{alt}J_{Lalt}N_{galt}{}^2 + R_{alt}J_{malt}}$ **(4.86)**

$$Q_{2alt} = \frac{K_{talt}}{R_{alt}J_{Lalt}N_{galt}{}^2 + R_{alt}J_{malt}} \quad \textbf{(4.87)}$$

$$Q_{3alt} = \frac{R_{alt}}{R_{alt}J_{Lalt}N_{galt}{}^2 + R_{alt}J_{malt}} \quad \textbf{(4.88)}$$

$$\ddot{\theta}_{malt} = -Q_{1alt}\dot{\theta}_{malt} + Q_{2alt}V_{alt} - Q_{3alt}T_{walt} \quad \textbf{(4.89)}$$

The full model of the dish is given:

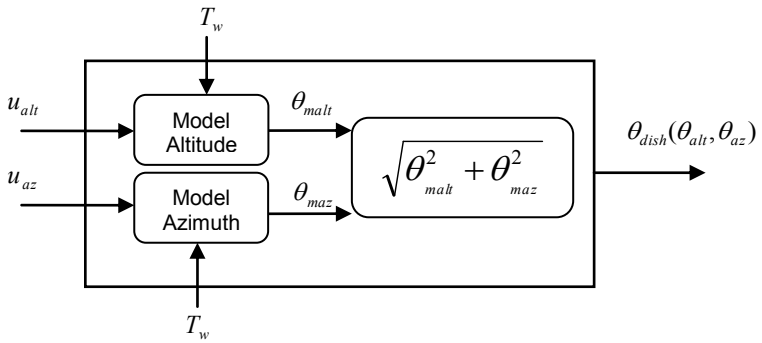


Figure 4.11: Model of the dish

4.2.6 Simulation of the full dish model

A step response is obtained using Matlab and Simulink software environment, see Figure (4.13). The results show that the dynamic behavior of the position of the motor is of a second order integrating system.

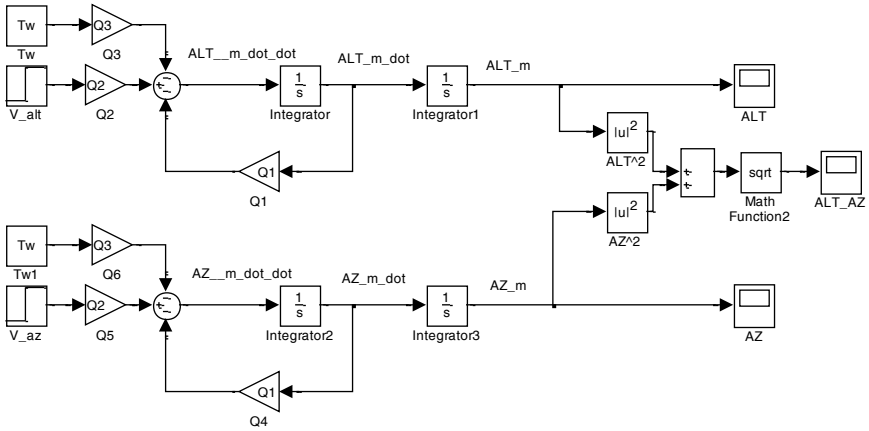


Figure 4.12: Dish and servo motor model with wind disturbance and load

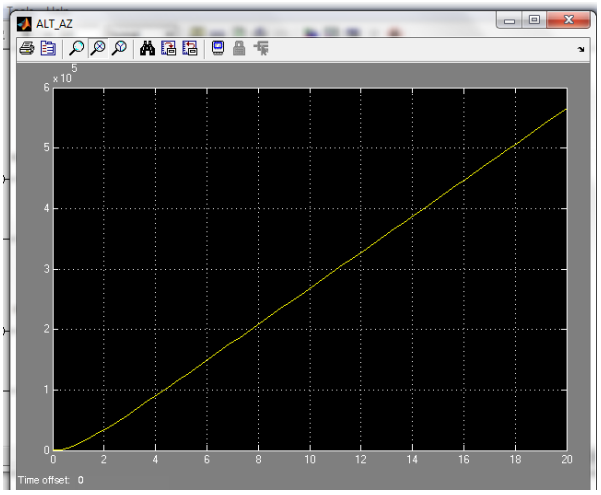


Figure 4.13: Step response for the dish and servo motor model with wind disturbance and load

4.3 Development of the mathematical model of the radio telescope system based on a stepper motor.

Stepper motors are very often used in as actuators for pointing systems as antennas, telescopes, and so on. (Morar, 2003). There are three types of stepper motors:

- Variable Reluctance (VR)
- Permanent Magnet (PM)
- Hybrid.

The thesis investigations are based on the permanent magnet stepper motor with two phase and 90^0 step size of these phases.

The dynamics of the stepper motor has to be considered together with the dynamics of the system that the motor will be driving. This requires a mathematical model of the stepper motor to be described.

The Permanent Magnet Stepper Motor model consists of electrical and mechanical sections. The electrical section is represented by an equivalent circuit, given in Figure (4.14) and the mechanical section is given in Figure (4.15). The equivalent circuits have been built with the assumption that the magnetic circuit is linear (no saturation) and the mutual inductance between phases is negligible. The definitions of the notations in the figure are given in Table (4.2).

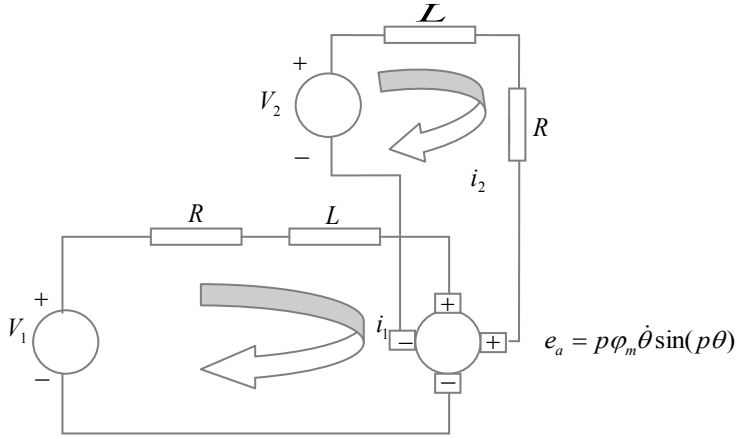


Figure 4.14: **Block diagram representation of phase windings of a PM Stepper Motor**

Table 4.2: **DMX-ETH-17 motor parameters**

Symbols /Letters	Definition/Explanation	Altitude Values	Azimuth Values	Units
V	Input Voltage	12	12	Volts
K_t	Motor toque constant	0.26	0.26	none
R	Electrical resistance	1.5	1.5	Ohms
L	Electric inductance	3.0	3.0	mH
K_b	Electromotive force constant	0.26	0.26	none
J_m	Motor moment of inertia	12.303	12.303	kg/m^2
J_L	Load moment of inertia	0	0	kg/m^2
B_m	Motor damping coefficient	0	0	Ns/m
B_L	Load damping coefficient	0	0	Ns/m
N_m	Number of teeth of the motor gear	0	0	none
N_L	Number of teeth of the load gear	0	0	none
D	Dish diameter	0	0	m
c_t	Dimensionless torque coefficient	-0.05 to 0.25	-0.05 to 0.25	none
α_p	Static air density	0.6126	0.6126	Ns^2/m^4
p	Number of poles pairs	50	50	none
m	Number of phases	2	2	none
θ_m	Stepping angle	1.8	1.8	degrees

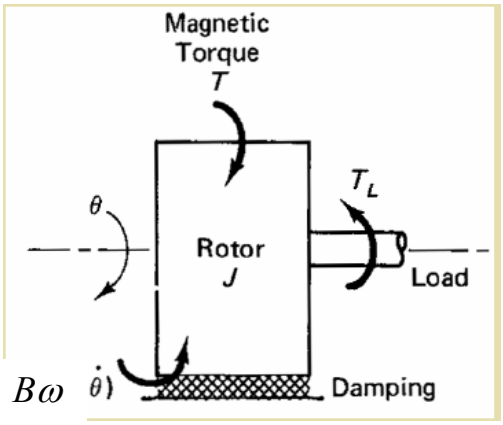


Figure 4.15: PM Stepper Motor torques adapted from (Craig K., 2004)

The diagram of the two phase PM stepper motor is given in Figure (4.16). The rotor is a permanent magnet with one north and south poles.

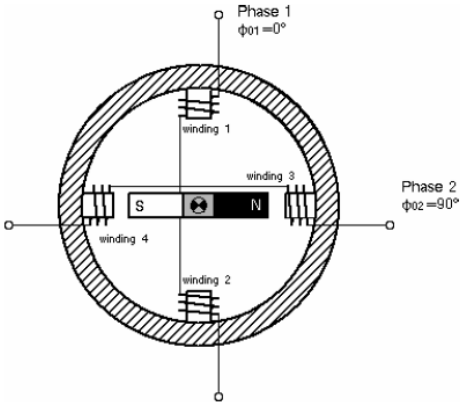


Figure 4.16: PM Stepper motor principle (adapted from Morar, 2003)

When power is supplied to one of the phases, generation of a magnetic dipole occurs on the stator. The movement of the rotor is due to the energizing of the phases one after another. The number of steps the rotor performs per one revolution is given as

$$s = 2pm \tag{4.90}$$

where (p) is the number of pole pairs and m is the number of the stator phases. The stepping angle is

$$\Delta\theta_m = \frac{360}{s} \tag{4.91}$$

Usually it is assumed that the magnetic field in the air gap of the motor has sinusoidal characteristics. Then the contribution of each phase j on the toque of the motor T_{mj} is (Morar, 2003):

$$T_{mj} = k_t \sin(p\theta_m(t) + \theta_{m0_j}) i_j(t) \tag{4.92}$$

where k_t is the motor constant $\theta_m(t)$ is the actual rotor position measured from the detent position of the presently excited phase that gives the relative position of the rotor during each step. θ_{m0j} is the position of the phase j coil in the stator. $i_j(t)$ is the current in the j-th coil.

For the considered case $\theta_{01} = 0^0$ and $\theta_{02} = 90^0$. The current in the coil j is determined by the supplied voltage V_j and the coil resistance and inductance , Figure (4.14). This dependence is given by the equation

$$V_j = e_j(t) + Ri_j(t) + L\frac{di_j(t)}{dt}, j = \overline{1,m} \quad \textbf{(4.93)}$$

where e_j is the electromotive force (EMF) induced in the phase j of the stator. R is the resistance of the core, and L is the inductance of the coil. It is assumed that all phases have the same values of R and L. The back emf can be expressed by a sinusoidal function of the rotor position

$$e_j = K_b(t)\sin(p\theta_m(t) + \theta_{m0j})\omega \quad \textbf{(4.93)}$$

where ω is the rotational velocity of the rotor and K_b is the electromotive force constant. Then:

$$V_j = K_b \sin[p\theta_m + \theta_{m0j}]\omega + Ri_j + L\frac{di_j}{dt}, j = \overline{1.m} \quad \textbf{(4.94)}$$

The total torque produced by all phases of the stepper motor T_m is:

$$T_m = \sum_{j=1}^{m} K_t \sin[p\theta_m + \theta_{m0j}]i_j \quad \textbf{(4.95)}$$

This torque drives the motor rotor and the load attached to it.

$$\sum_{j=1}^{m} T_{mj} = J_T\frac{d\omega}{dt} + B_T\omega + T_w \quad \textbf{(4.96)}$$

where J_T is the total inertia of the rotor and the load, B_T is the viscous damping constant for both the rotor and the load and T_w is the wind disturbance torque. Then the torque that is available to drive the load and to compensate the wind disturbance is

$$J_T\frac{d\omega}{dt} = \sum_{j=1}^{m} T_{mj} - B_T\omega - T_w = \sum_{j=1}^{m} K_t \sin[p\theta_m + \theta_{m0j}]i_j - B_T\omega - T_w \quad \textbf{(4.97)}$$

The load acceleration torque is the difference between the sum of the torques produced by each phase and the resistance torque created by the friction and viscous damping. The torque of the load is expressed by the Equation (4.34).

This equation represents the model of the dish considered as a load.

4.3.1 Derivation of the mathematical model of the dish and motor under the wind disturbance

Equations (4.94) and (4.97) represent the model of the motor, the load and the disturbance. The block diagram representing the system is shown in Figure (4.17)

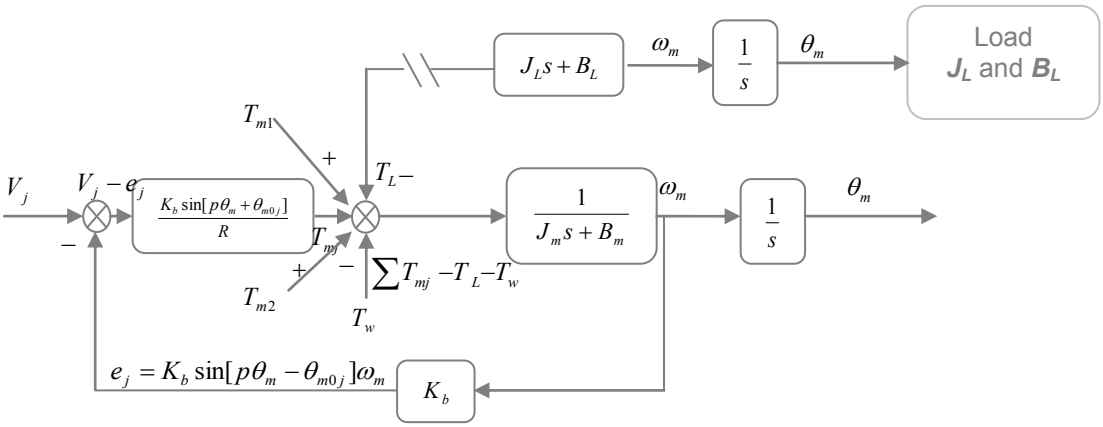


Figure 4.17: **Block diagram model of the stepper motor with disturbance and load**

In the motor equations the total moment of inertia and the total damping coefficients can be expressed by the corresponding coefficients of the motor and the gear connecting the motor and the dish antenna. Equation (4.42) and 4.43).

Adding the wind torque disturbing the motor torque and the expression of the gear ratio $N_g = \frac{N_m}{N_L}$, where N_m is the number of winding in the motor side and the N_L is the number of winding in the load side, then Equation (4.96) can be written in the following way:

$$\sum_{j=1}^{m} T_{mj} - T_w = (J_m + J_L N_g^2)\ddot{\theta}_m + (B_m + B_L N_g^2)\dot{\theta}_m$$
$$= (J_m \ddot{\theta}_m + B_m \dot{\theta}_m) + J_L N_g^2 \ddot{\theta}_m + B_L N_g^2 \dot{\theta}_m \quad \textbf{(4.98)}$$
$$= J_m \ddot{\theta}_m + B_m \dot{\theta}_m + T_L$$

The expression for $\sum_{j=1}^{m} T_{mj}$ and Equation (4.95) is substituted in Equation (4.98).

$$\sum_{j=1}^{m} K_t \sin[p\theta_m + \theta_{m0j}] i_j - T_w = (J_m + J_L N_g^2)\ddot{\theta}_m + (B_m + B_L N_g^2)\dot{\theta}_m \quad \textbf{(4.99)}$$

Two types of state space nonlinear models of the stepper motor are developed on the basis of Equations (4.94) and (4.99):

- state space model with state space variables given by the currents, position angle, and the rotor velocity
- state space model with state variables given by the position and velocity.

Derivation of these models is given below.

4.3.2 Model with state variables given by the position and velocity

If it is assumed that R, L and V are the same for all phases then $i_j = i$ and can be considered outside of the sum in Equation (4.99). Then the current can be expressed as

$$i = \frac{(J_m + J_L N_g^2)\ddot{\theta}_m + (B_m + B_L N_g^2)\dot{\theta}_m + T_w}{\sum_{j=1}^{m} K_t \sin[p\theta_m + \theta_{m0j}]} \quad \textbf{(4.100)}$$

The model of the system consisting of the motor and the antenna with wind disturbance has for an input the voltage applied to the motor phases and as an outputs the angle, velocity and its derivative. In order to obtain this model the value for the current in Equation (4.100) has to be substituted in Equation (4.94) for the case when i, R, L and V are the same for all phases.

The Equation (4.94) is

$$V = K_b \sin[p\theta_m + \theta_{m0j}]\dot{\theta} + Ri + L\frac{di}{dt}, j = \overline{1.m} \quad \textbf{(4.101)}$$

It is necessary first to derive $\frac{di}{dt}$ on the basis of Equation (4.100) and then to substitute i and $\frac{di}{dt}$ in the Equation (4.101).

$$\frac{di}{dt} = \frac{\left\{(J_m + J_L N_g^2)\dddot{\theta}_m + (B_m + B_L N_g^2)\ddot{\theta}_m\right\}\sum_{j=1}^{m} K_t p \sin[p\theta_m + \theta_{m0j}] -}{} \quad \textbf{(4.102)}$$

$$- \frac{\left\{(J_m + J_L N_g^2)\ddot{\theta}_m + (B_m + B_L N_g^2)\dot{\theta}_m\right\}\sum_{j=1}^{m} K_t p \cos[p\theta_m + \theta_{m0j}]\dot{\theta}_m}{\left\{\sum_{j=1}^{m} K_t \sin[p\theta_m + \theta_{m0j}]\right\}^2}$$

Equation (4.100) and (4.102) are substituted into Equation (4.101).

$$V = K_b \sin[p\theta_m + \theta_{m0j}]\dot{\theta} + \frac{R\left\{(J_m + J_L N_g^2)\ddot{\theta}_m + (B_m + B_L N_g^2)\dot{\theta}_m + T_w\right\}}{\sum_{j=1}^{m} K_t \sin[p\theta_m + \theta_{m0j}]}$$

$$+ L\frac{\left\{(J_m + J_L N_g^2)\dddot{\theta}_m + (B_m + B_L N_g^2)\ddot{\theta}_m\right\}\sum_{j=1}^{m} K_t p \sin[p\theta_m + \theta_{m0j}] -}{} \quad \textbf{(4.103)}$$

$$- \frac{\left\{(J_m + J_L N_g^2)\ddot{\theta}_m + (B_m + B_L N_g^2)\dot{\theta}_m\right\}\sum_{j=1}^{m} K_t p \cos[p\theta_m + \theta_{m0j}]\dot{\theta}_m}{\left\{\sum_{j=1}^{m} K_t \sin[p\theta_m + \theta_{m0j}]\right\}^2}$$

After some algebraic transformations it is obtained for the stepper motor and antenna model:

$$\left\{\sum_{j=1}^{m} K_t \sin[p\theta_m + \theta_{m0j}]\right\}^2 V - K_b \sin[p\theta_m + \theta_{m0j}]\dot{\theta}\left\{\sum_{j=1}^{m} K_t \sin[p\theta_m + \theta_{m0j}]\right\}^2 -$$

$$- R\sum_{j=1}^{m} K_t \sin[p\theta_m + \theta_{m0j}]\left\{(J_m + J_L N_g^2)\ddot{\theta}_m + (B_m + B_L N_g^2)\dot{\theta}_m + T_w\right\} -$$

$$- L\left\{(J_m + J_L N_g^2)\dddot{\theta}_m + (B_m + B_L N_g^2)\ddot{\theta}_m\right\}\sum_{j=1}^{m} K_t p \sin[p\theta_m + \theta_{m0j}] - \quad \textbf{(4.104)}$$

$$- L\left\{(J_m + J_L N_g^2)\ddot{\theta}_m + (B_m + B_L N_g^2)\dot{\theta}_m\right\}\sum_{j=1}^{m} K_t p \cos[p\theta_m + \theta_{m0j}]\dot{\theta}_m = 0$$

Equation (4.104) is a nonlinear dynamic model of the system of a motor and dish antenna. It is very complex because of the nonlinear dependence between the first and second order derivative of the motor angle. If it is assumed that the inductance L is very small $L \approx 0$, Equation (4.104) can be simplified as follows.

$$V\sum_{j=1}^{m} K_t \sin[p\theta_m + \theta_{m0j}] - \left[K_b \sin[p\theta_m + \theta_{m0j}]\sum_{j=1}^{m} K_t \sin[p\theta_m + \theta_{m0j}]\right]\dot{\theta}_m - \quad \textbf{(4.105)}$$

$$- R\left\{(J_m + J_L N_g^2)\ddot{\theta}_m + (B_m + B_L N_g^2)\dot{\theta}_m + T_w\right\} = 0$$

This equation can be solved according to the highest order derivative $\ddot{\theta}_m$

$$\ddot{\theta}_m = \frac{R(B_m + B_L N_g^2) + K_b \sin[p\theta_m + \theta_{m0j}]\sum_{j=1}^{m} K_t \sin[p\theta_m + \theta_{m0j}]}{R(J_m + J_L N_g^2)}\dot{\theta}_m + \quad \textbf{(4.106)}$$

$$+ \frac{V\sum_{j=1}^{m} K_t \sin[p\theta_m + \theta_{m0j}]}{R(J_m + J_L N_g^2)} - \frac{R}{R(J_m + J_L N_g^2)}T_w \ , j = \overline{1,m}$$

Equation (4.106) can be represented in a state space form. First notations are introduced for the nonlinear functions in front of the derivatives, the control input and the disturbance, as follows:

$$S_1(\theta_m, \theta_{m0j}) = \frac{R(B_m + B_L N_g^2) + K_b \sin[p\theta_m + \theta_{m0j}] \sum_{j=1}^{m} K_t \sin[p\theta_m + \theta_{m0j}]}{R(J_m + J_L N_g^2)},\ j = \overline{1,m}$$

$$S_2(\theta_m, \theta_{m0j}) = \frac{\sum_{j=1}^{m} K_t \sin[p\theta_m + \theta_{m0j}]}{R(J_m + J_L N_g^2)},\ j = \overline{1,m} \qquad \textbf{(4.107)}$$

$$S_3 = \frac{R}{R(J_m + J_L N_g^2)},\ j = \overline{1,m}$$

After substituting of Equation (4.107) in Equation (4.106)

$$\ddot{\theta}_m = S_1(\theta_m, \theta_{m0j})\dot{\theta}_m + S_2(\theta_m, \theta_{m0j})V - S_3 T_w\ ,\ j = \overline{1,m} \qquad \textbf{(4.108)}$$

The state space model is derived following Equations (4.75) to Equation (4.80) for the state space vector $X = \begin{bmatrix} X_1 & X_2 \end{bmatrix}^T = \begin{bmatrix} \theta_m & \dot{\theta}_m \end{bmatrix}^T$, and u = V.

$$\begin{bmatrix} \dot{X}_{1j} \\ \dot{X}_{2j} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -S_1(\theta_m, \theta_{m0j}) \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} + \begin{bmatrix} 0 \\ S_2(\theta_m, \theta_{m0j}) \end{bmatrix} u + \begin{bmatrix} 0 \\ S_3 \end{bmatrix} T_w$$
$$y_j = \theta_m = X_1 = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}, j = \overline{1,m} \qquad \textbf{(4.109)}$$

where the state space matrices are

$$A_j = \begin{bmatrix} 0 & 1 \\ 0 & S_1 \end{bmatrix},\quad B_j = \begin{bmatrix} 0 \\ S_2 \end{bmatrix},\ w_j = \begin{bmatrix} 0 \\ -S_3 \end{bmatrix},\ e_j = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

The state space model of the stepper motor has the same expression as this of the DC servo motor but the state space matrices are nonlinear functions of the motor angle. The difference is that it is only for one phase of the motor - j.

The model in Equation (4.109) is derived only for one phase. The model for all phases is obtained through augmentation of the vector of state variables of this phase with the corresponding vector for all other phases:

$$\begin{bmatrix} \dot{X}_{11} \\ \dot{X}_{21} \\ \dot{X}_{12} \\ \dot{X}_{22} \\ : \\ \dot{X}_{1m} \\ \dot{X}_{21} \end{bmatrix} = \begin{bmatrix} \dot{\theta}_{m1} \\ \dot{\theta}_{m1} \\ \dot{\theta}_{m2} \\ \dot{\theta}_{m2} \\ : \\ \dot{\theta}_{mm} \\ \ddot{\theta}_{mm} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & .. & 0 & 0 \\ 0 & -S_1(\theta_m, \theta_{m01}) & 0 & 0 & .. & 0 & 0 \\ 0 & 0 & 0 & 1 & .. & 0 & 0 \\ 0 & 0 & 0 & S_2(\theta_{m1}, \theta_{m02}) & .. & 0 & 0 \\ .. & .. & .. & .. & .. & 0 & 0 \\ 0 & 0 & 0 & 0 & .. & 0 & 1 \\ 0 & 0 & 0 & 0 & .. & 0 & S_1(\theta_{m1}, \theta_{m0m}) \end{bmatrix} \begin{bmatrix} \dot{\theta}_{m1} \\ \dot{\theta}_{m1} \\ \dot{\theta}_{m2} \\ \dot{\theta}_{m2} \\ : \\ \dot{\theta}_{mm} \\ \ddot{\theta}_{mm} \end{bmatrix} +$$

$$+ \begin{bmatrix} 0 \\ -S_1(\theta_m, \theta_{m01}) \\ 0 \\ S_2(\theta_{m1}, \theta_{m02}) \\ : \\ 0 \\ S_2(\theta_{m1}, \theta_{m02}m) \end{bmatrix} u + \begin{bmatrix} 0 \\ S_3 \\ 0 \\ S_3 \\ : \\ 0 \\ S_3 \end{bmatrix} T_w$$

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 & .. & 0 & 0 \\ 0 & 0 & 1 & 0 & .. & 0 & 0 \\ 0 & 0 & 0 & 0 & .. & 0 & 0 \\ 0 & 0 & 0 & 0 & .. & 0 & 0 \\ .. & .. & .. & .. & .. & .. & .. \\ 0 & 0 & 0 & 0 & .. & 0 & 0 \\ 0 & 0 & 0 & 0 & .. & 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_{m1} \\ \dot{\theta}_{m1} \\ \dot{\theta}_{m2} \\ \dot{\theta}_{m2} \\ : \\ \dot{\theta}_{mm} \\ \ddot{\theta}_{mm} \end{bmatrix} \quad \textbf{(4.110)}$$

This model can be expressed in terms of matrices $A_j, B_j, w_j,$ and $e_j,$ as follows:

$$\begin{bmatrix} \dot{\theta}_{m1} \\ \ddot{\theta}_{m1} \\ : \\ \dot{\theta}_{mm} \\ \ddot{\theta}_{mm} \end{bmatrix} = \begin{bmatrix} A_1 & 0 & 0 & 0 & 0 \\ 0 & A2 & 0 & 0 & 0 \\ .. & .. & .. & .. & .. \\ 0 & 0 & 0 & A_j & 0 \\ 0 & 0 & 0 & 0 & A_m \end{bmatrix} \begin{bmatrix} \dot{\theta}_{m1} \\ \dot{\theta}_{m1} \\ : \\ \dot{\theta}_{mm} \\ \ddot{\theta}_{mm} \end{bmatrix} + \begin{bmatrix} B_1 \\ B_1 \\ : \\ B_j \\ B_m \end{bmatrix} u + \begin{bmatrix} w_1 \\ w_1 \\ : \\ w_j \\ w_m \end{bmatrix} T_w$$

$$y = \begin{bmatrix} y_1 \\ y_2 \\ : \\ : \\ y_m \end{bmatrix} = \begin{bmatrix} \theta_{m1} \\ \theta_{m2} \\ : \\ : \\ \theta_{mm} \end{bmatrix} = \begin{bmatrix} C_1 & 0 & 0 & 0 & 0 \\ 0 & C_2 & 0 & 0 & 0 \\ 0 & 0 & C_j & 0 & 0 \\ .. & .. & .. & .. & .. \\ 0 & 0 & 0 & 0 & C_m \end{bmatrix} \begin{bmatrix} \theta_{m1} \\ \dot{\theta}_{m1} \\ : \\ \theta_{mm} \\ \dot{\theta}_{mm} \end{bmatrix} \quad \textbf{(4.111)}$$

where the dimension of the matrix $A \in R^{2mx2m}, B \in R^{2mx1}, w \in R^{2mx1}$, and $C \in R^{mx2m}$

Model given by Equation (4.111) is based on the assumption that the voltage applied to all phases are the same which introduces limitations on the possibilities to control the motor. The second model do not introduce this assumption.

4.3.3 Model with state variables given by the currents, position and velocity

The Equation (4.94) and (4.99) are used to build the state space model of the motor and the dish. In these equations $i_j, j=\overline{1,m}$ is the variable current in each phase, V_j is the variable voltage applied to every phase, which is considered as a control input, θ_m is the variable rotor angle and ω is variable speed of the motor. The variables i_j, θ_m and ω are selected as state space variables. Then Equation (4.94) and (4.99) are rewritten according to the derivatives of these variables, as follows;

- for the currents $i_j, j=\overline{1,m}$

The derivative $\frac{di_j}{dt}$ is expressed from Equation (4.94)

$$\frac{di_j}{dt}=\frac{V_j}{L}-\frac{K_b\sin[p\theta_m+\theta_{m0j}]\omega}{L}-\frac{Ri_j}{L}, j=\overline{1.m} \quad \textbf{(4.112)}$$

for the motor angle θ_m

$$\frac{d\theta_m}{dt}=\omega \quad \textbf{(4.113)}$$

for the motor velocity the Equation (4.99) is used to express the derivative $\ddot{\theta}_m=\dot{\omega}$ and $\dot{\theta}_m=\omega$, as follows:

$$\sum_{j=1}^{m}K_t\sin[p\theta_m+\theta_{m0j}]i_j-T_w=(J_m+J_LN_g^2)\frac{d\omega}{dt}+(B_m+B_LN_g^2)\omega \quad \textbf{(4.114)}$$

from here

$$\frac{d\omega}{dt}=\frac{\sum_{j=1}^{m}K_t\sin[p\theta_m+\theta_{m0j}]i_j}{(J_m+J_LN_g^2)}-\frac{T_w}{(J_m+J_LN_g^2)}-\frac{(B_m+B_LN_g^2)\omega}{(J_m+J_LN_g^2)} \quad \textbf{(4.115)}$$

Equation (4.112), (4.113) and (4.115) describe the model of the motor, gear and the dish. The vector of the state variables is:

$$X=\left[i_1\ i_2\ ...i_j\ ...i_m\ \theta_m\ \omega\right]^T\ \in R^{mx2}$$

The initial conditions of the state space variables are $i_1(0)=i_{10}(0)=0...i_j(0)=i_{0j}=0, \theta_m(0)=0, \omega(0)=0$.

The state space model is built on the basis of Equations (4.112), (4.113) and (4.115).

$$\dot{X}=\begin{bmatrix}\dot{i}_1\\ \dot{i}_2\\ \vdots\\ \dot{i}_j\\ \vdots\\ \dot{i}_m\\ \dot{\theta}_m\\ \dot{\omega}\end{bmatrix}=\begin{bmatrix}\frac{-R}{L} & 0 & .. & 0 & .. & 0 & 0 & -\frac{K_b\sin[p\theta_m+\theta_{m01}]}{L}\\ 0 & \frac{-R}{L} & .. & 0 & .. & 0 & 0 & -\frac{K_b\sin[p\theta_m+\theta_{m02}]}{L}\\ \vdots & \vdots & .. & & .. & \vdots & \vdots & \vdots\\ 0 & 0 & .. & \frac{-R}{L} & .. & 0 & 0 & -\frac{K_b\sin[p\theta_m+\theta_{m0j}]}{L}\\ \vdots & \vdots & .. & \vdots & .. & \vdots & \vdots & \vdots\\ 0 & 0 & .. & 0 & .. & 0 & 0 & -\frac{K_b\sin[p\theta_m+\theta_{m0m}]}{L}\\ 0 & 0 & .. & 0 & .. & \frac{-R}{L} & 0 & -\frac{K_b\sin[p\theta_m+\theta_{m01}]}{L}\\ \frac{K_b\sin[p\theta_m+\theta_{m01}]}{(J_m+J_LN_g^2)} & \frac{K_b\sin[p\theta_m+\theta_{m02}]}{(J_m+J_LN_g^2)} & .. & \frac{K_b\sin[p\theta_m+\theta_{m0m}]}{(J_m+J_LN_g^2)} & .. & 0 & & -\frac{(B_m+B_LN_g^2)}{(J_m+J_LN_g^2)}\end{bmatrix}\begin{bmatrix}i_1\\ i_2\\ \vdots\\ i_j\\ \vdots\\ i_m\\ \theta_m\\ \omega\end{bmatrix}+$$

$$+\begin{bmatrix}1/L & 0 & .. & 0 & .. & 0\\ 0 & 1/L & .. & 0 & .. & 0\\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\ 0 & 0 & .. & 1/L & 0 & 0\\ 0 & 0 & .. & 0 & 0 & 0\\ 0 & 0 & .. & 0 & 0 & 0\end{bmatrix}\begin{bmatrix}V_1\\ V_2\\ \vdots\\ V_j\\ \vdots\\ V_m\end{bmatrix}+\begin{bmatrix}0\\ 0\\ \vdots\\ 0\\ 0\\ -\frac{1}{(J_m+J_LN_g^2)}\end{bmatrix}T_w \qquad \textbf{(4.116)}$$

or $\dot{X}(t)=A(\theta_m)X(t)+Bu(t)+WT_w$, where $u(t)=V(t)\in R^m$ is the control input. The model in Equation (4.116) is a nonlinear one because the matrix A depends on sin functions of the state variable θ_m. The dimensions of the matrices are $A\in R^{(mx2)x(mx2)}, B\in R^{(mx2)xm}, w\in R^{2mx1}$.

This derived model in Equation (4.116) for the case of motor with two phases takes the form:

$$\dot{X}=\begin{bmatrix}\dot{i}_1\\ \dot{i}_2\\ \dot{\theta}_m\\ \dot{\omega}\end{bmatrix}=\begin{bmatrix}\frac{-R}{L} & 0 & 0 & -K_b\sin[p\theta_m+\theta_{m01}]\\ 0 & \frac{-R}{L} & 0 & -K_b\sin[p\theta_m+\theta_{m02}]\\ 0 & 0 & 0 & 1\\ \frac{K_b\sin[p\theta_m+\theta_{m01}]}{(J_m+J_LN_g^2)} & \frac{K_b\sin[p\theta_m+\theta_{m02}]}{(J_m+J_LN_g^2)} & 0 & -\frac{(B_m+B_LN_g^2)}{(J_m+J_LN_g^2)}\end{bmatrix}\begin{bmatrix}i_1\\ i_2\\ \theta_m\\ \omega\end{bmatrix}+$$

$$+\begin{bmatrix}\frac{1}{L} & 0\\ 0 & \frac{1}{L}\\ 0 & 0\\ 0 & 0\end{bmatrix}\begin{bmatrix}V_1\\ V_2\end{bmatrix}+\begin{bmatrix}0\\ 0\\ 0\\ -\frac{1}{(J_m+J_LN_g^2)}\end{bmatrix}T_w \qquad \textbf{(4.117)}$$

The initial conditions for the motor with two phases and one pole of the rotor are:

$i_1(0)=i_2(0)=0,\ \theta_m(0)=0[rad]=0$

The positions of the phase coils are:

$$\theta_{m01}=0,\ \theta_{m02}=\frac{360}{2*2*1}=\frac{\pi}{2}rad=90^0$$

For this case the model in Equation (4.117) can be simplified on the basis that $\sin(p\theta_m + \pi/2) = -\cos(p\theta_m)$.

$$\dot{X} = \begin{bmatrix} \dot{i}_1 \\ \dot{i}_2 \\ \dot{\theta}_m \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} \frac{-R}{L} & 0 & 0 & -K_b \sin(p\theta_m) \\ 0 & \frac{-R}{L} & 0 & K_b \cos(p\theta_m) \\ 0 & 0 & 0 & 1 \\ \frac{K_b \sin(p\theta_m)}{(J_m + J_L N_g^2)} & \frac{-K_b \cos(p\theta_m)}{(J_m + J_L N_g^2)} & 0 & -\frac{(B_m + B_L N_g^2)}{(J_m + J_L N_g^2)} \end{bmatrix} \begin{bmatrix} i_1 \\ i_2 \\ \theta_m \\ \omega \end{bmatrix} +$$

$$+ \begin{bmatrix} \frac{1}{L} & 0 \\ 0 & \frac{1}{L} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ -\frac{1}{(J_m + J_L N_g^2)} \end{bmatrix} T_w \qquad \textbf{(4.118)}$$

$$y = \begin{bmatrix} \theta_m \\ \omega \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} i_1 \\ i_2 \\ \theta_m \\ \omega \end{bmatrix}$$

where $A \in R^{4x4}, B \in R^{4x2}, C \in R^{2x4}, w \in R^{4x1}, X \in R^4, u \in R^2, T_w \in R^1$

The model given by Equation (4.118) is used for the simulation of the stepper motor in the thesis.

4.3.4 Simulation of the model of the motor and the load in Matlab/Simulink

A step response is obtained using Matlab and Simulink software environment, see Figure (4.18). The results show that the dynamic behavior of the position of the motor is of a sinusoidal type as shown in Figure (7.49).

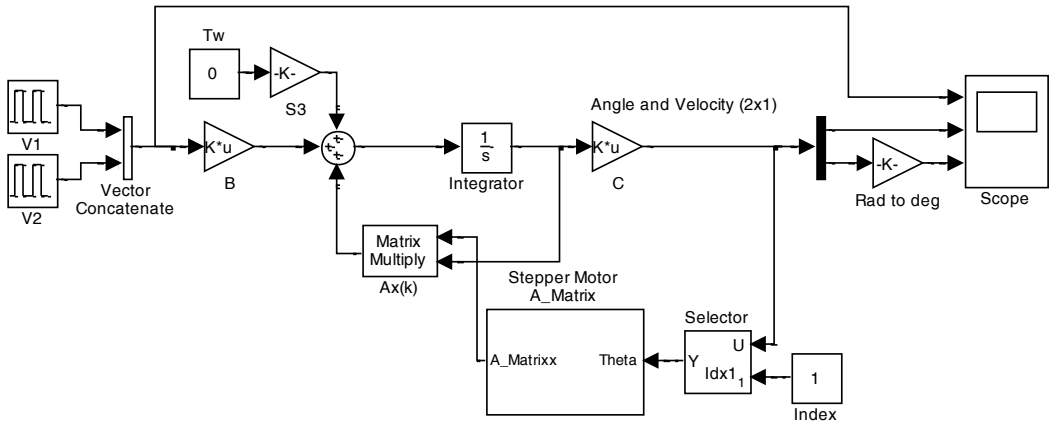


Figure 4.18: Dish and stepper motor model with wind disturbance and load

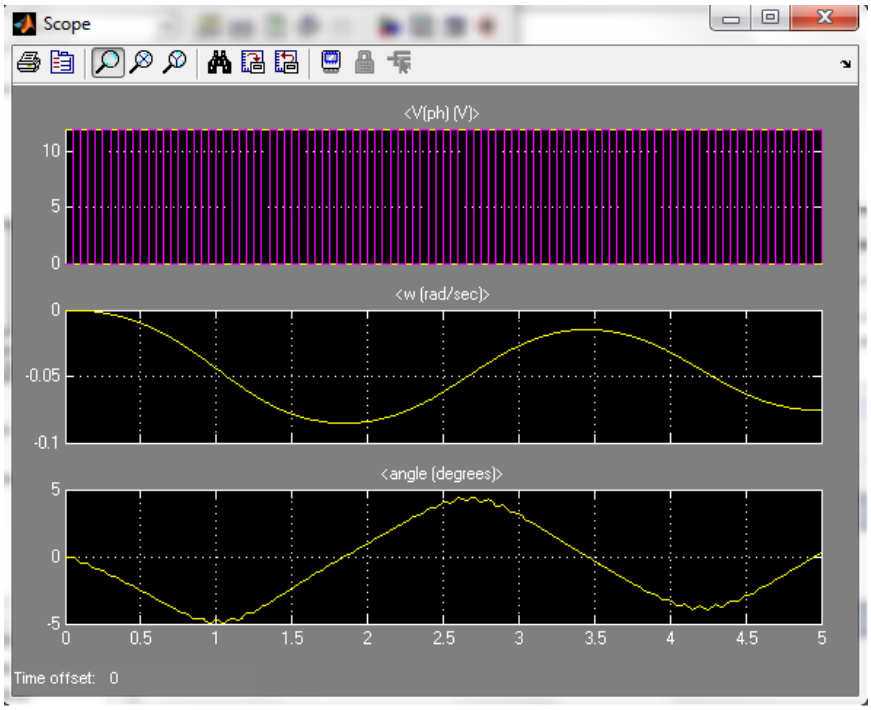


Figure 4.19: Open loop step response for the dish and stepper motor model with wind disturbance and load

4.4 Conclusion

Servo motor and Permanent Magnet stepper motor with wind disturbance mathematical model derivations are presented. The models are developed using laws of physics and represented in both transfer function and state space forms. For the servo motor the model is found to be linear second order type system. For the stepper motor the model is found to be nonlinear.

Matlab/Simulink is used to simulate the open loop step response on the servo and stepper motor..

The derived models for the system of the motor and the dish has the same structure for both the azimuth and altitude. The difference is in the values of the parameters of the motor circuits and the gear ratios. For the considered case in the thesis the two motors used have the same characteristics, the only difference is in the gear ratio that connects the stepper motor to the dish load.

The model with the load and wind disturbance will be used in the thesis and the controller shall be designed to eliminate the disturbance introduced by the wind.

The next chapter investigates the influence of network induced time delays over the system transition behavior for the case of standard type of controllers. The simulations in the thesis are performed for the model of the plant based on a DC servo motor using the dish parameters of the scaled model of the SKA Radio Telescope project, but the implementation in real-time is done on the basis of a permanent magnet two phase stepper motor.

CHAPTER FIVE
INVESTIGATION OF THE INFLUENCE OF THE NETWORK INDUCED DELAYS FOR THE CASE OF STANDARD CONTROLLERS USING SIMULATIONS

5.1 Introduction

This chapter is used to find the values of the critical delays to be used further for the design of the controller. The closed loop control system design is considered without paying attention to network imperfections. The structure of the closed loop control system without network imperfections is shown in Figure (5.1). The aim of the design and simulation is to determine the maximum network induced time delay that causes the closed loop control system to become unstable. This maximum value is used as threshold and based on this threshold a robust controller can be further designed to overcome the delay influence. The requirements for the closed loop system are described in section 5.2. The derivation of the closed loop controller is described in section 5.4. The controller design methods considered are PID controller using second method of Ziegler-Nichols to determine the PID parameters and State Space pole placement controller design method. The controllers are designed using transfer function and state space representation of the plant. The simulations of the closed loop control system considering the plant and the different types of controllers are shown in section 5.5.

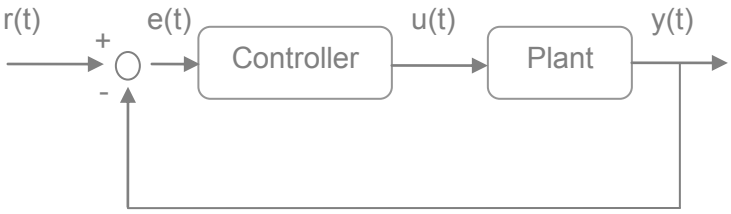


Figure 5.1: Typical closed loop control system without network imperfections
r(t) : Reference, e(t): error, u(t): plant input and y(t): plant output

5.2 System requirements and performance

The requirements for the closed loop control system considered in the thesis are summarized in Table (5.1), (5.2), and (5.3). Note that the pointing accuracy requirement is for the overall system performance. The closed loop control system shall be designed to make sure that these requirements are met. The requirements are considered for a single dish radio telescope.

Table 5.1: Summary of the Azimuth and Altitude requirements

Requirement	Azimuth (Value)	Altitude (Value)
Overall pointing accuracy	< 0.02 deg	< 0.004 deg
Stop distance	< 2 deg	< 2 deg
Maximum speed	> 2 deg/sec	> 1 deg/sec
Tracking speed	0 – 0.5 deg/sec	0 – 0.5 deg/sec
Usage range	-185 deg to 275 deg (460 deg)	0 - +95 deg

Table 5.2: Summary of disturbance rejection requirements

Requirement	Value
Wind	36km/h
Stow wind	45km/h
Oscillation amplitude	< 0.0014 deg
Structural natural frequency	> 3Hz

Table 5.3: Performance Specification

Requirement	Value
Settling Time	Ts <10sec
Percentage Overshoot	1%<PO<25%

Based on the requirements in Table (5.1), (5.2) and (5.3) the following performance measures are used to monitor the overall performance of the:

- Settling time,
- Rising time,
- Percentage Overshoot, and
- Steady State error.

5.3 Controller design methods

The controller design method depends on the type of the system process model. State Space and Transfer function plant models as derived in Chapter 4 are considered. The mathematical model of the plant considered in these simulations is for the servo motor. Two types of controllers are designed being:

- PID Controller in Laplace domain and
- State-space controller in state space continuous time domain.

Two design methods are applied:

- The second method of Ziegler-Nichols in Laplace domain and the
- Pole placement method in Laplace and continuous time domain.

These methods for the design of the controllers are used to compare the behaviour of the system when the controller is designed without paying attention to network induced time

delays during the design process with the behaviour of the system when the controllers are designed paying attention to these disturbances. The following cases are considered:

- PID Controller is designed for a NCS without delay using the 2^{nd} method of Ziegler Nichols. The delays between the controller and the actuator and between the sensor and controller are introduced and their impact over the closed loop system behaviour is investigated.
- PID Controller is designed for a NCS without delay using the pole placement method in Laplace domain. The delays between the controller and the actuator and between the sensor and controller are introduced and their impact over the closed loop system behaviour is investigated.
- State space Proportional Controller is designed for a NCS without delay using the pole placement method. The delays between the controller and the actuator and between the sensor and controller are introduced and their impact over the closed loop system behaviour is investigated.
- State space Proportional-Integral (PI) Controller is designed for a NCS without delay using the pole placement method. The delays between the controller and the actuator and between the sensor and controller are introduced and their impact over the closed loop system behaviour is investigated.

5.3.1 Simulation Environment

The simulation of the closed loop control system is realised using Matlab and Simulink for two motors representing Azimuth and Altitude. The delays are implemented between the controller and the actuator τ^{ca} and between the sensor and the controller τ^{sc}.

The overview of the simulation files and the relation to their respective controller design methods is shown in Figure (5.2). General simulation parameters for the plant model are shown in Table (4.1). Parameters specific to the controller are described in the section that describes that controller.

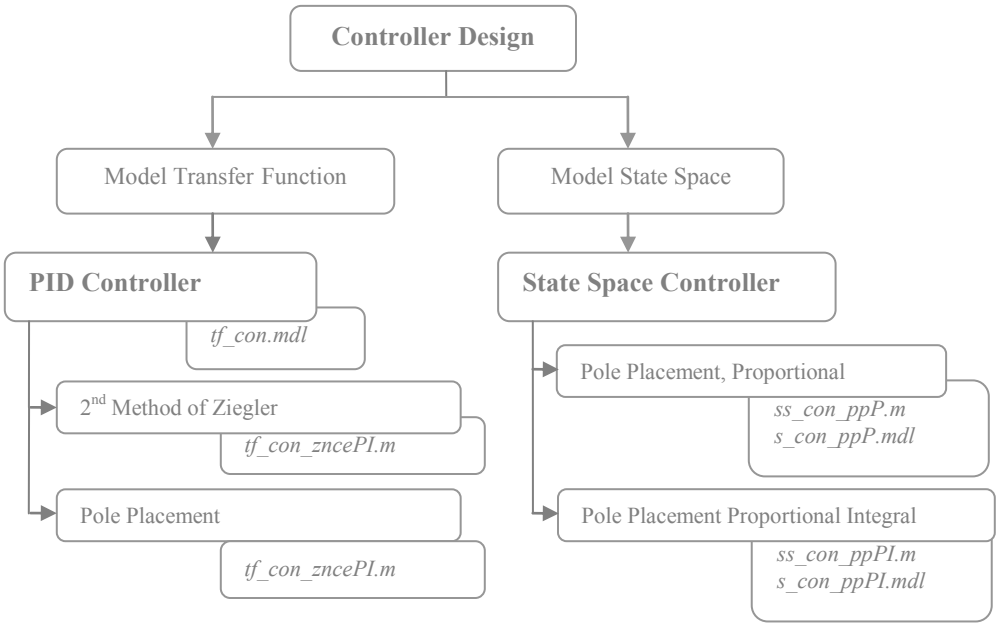


Figure 5.2: Simulation Hierarchy
.mdl file refers to Simulink model files and .m file refers to Matlab script file.

For simulation procedures where network induced time delays are considered, the values of the total delay in the NCS is selected to be:

- delay equal zero, $\tau_T = 0$, where $\tau_T = \tau^{ca} + \tau^{sc}$, where the system type transition behaviour is undamped and stable.
- delay is greater than zero but less than critical delay, $0 < \tau_T < \tau_c$, where the system transition behaviour type is undamped, but the system has overshoot but still stable.
- delay is equal to critical delay (τ_c), $\tau_T = \tau_c$, where the system transition behaviour type is oscillatory and the overall system is unstable.
- delay is greater than the critical delay, $\tau_T > \tau_c$, where the system transition behaviour type is unstable.

These considerations are repeated for **cases** where:

- delays are considered to be between controller and actuator ONLY; τ^{ca}
- delays are considered to be between sensor and controller ONLY; τ^{sc}
- delays are considered to be between the controller and the actuator τ^{ca} and between the sensor and the controller τ^{sc}.

The network time delays are considered for different controller designs.

The resultant from the step response simulation graphs can be interpreted as follows :

- Each simulation has an output graph that has four sub-plots which represent the response of the closed loop system for the different delay considerations as described in each section of the controller design method.
- X axis data is the continues time is seconds.
- The network induced time delay input is specified as number of seconds.
- Y axis data displays the magnitude of the reference input, the measurement output, and the Control input (after the delay between the controller and the actuator).

5.4 PID controller design using 2nd Method of Ziegler-Nichols

PID controller has been used for the past five decades and originated from the process control industry (Moog, 2003). The PID controller output is achieved by multiplying the Proportional constant (k_p) with the error over a certain time interval plus Integral constant (k_i) multiplied by the integral of the error plus the Derivative constant (k_d) multiplied by the derivative of the error over a certain time interval. The output of the controller is given by Equation (5.1).

$$g(t) = k_p e(t) + k_i \int_0^t e(t)dt + k_d \frac{de(t)}{dt} \quad \textbf{(5.1)}$$

Taking Laplace Transform and representing the controller in the (s) domain the transfer function of the PID controller is:

$$G_c(s) = \frac{k_d s^2 + k_p s + k_I}{s} \quad \textbf{(5.2)}$$

Tuning the PID parameters such that the desired closed loop system output is achieved is the main challenge with PID controllers. Some of the methods used to tune the PID parameters are; trial and error method, first method of Ziegler-Nichols, second method of Ziegler-Nichols, Pole placement method and many others (Astrom and Hagglund, 1995).

Ziegler-Nichols method of finding PID parameter has been used for the past decades in industry (Vance, 1996). Ziegler-Nichols have different types of controller design methods depending on the type of the process. For this process a second method of Ziegler-Nichols also known as the oscillation method is used. The method is based on the system gain. The system gain is increased until the system reaches constant stable oscillations. The system gain used to obtain stable oscillation is called the ultimate gain (M) and the corresponding period of oscillations is called the ultimate period (Pu). The

ultimate gain, ultimate period and the amplitude (***a***) of the oscillations are used to calculate the PID parameters based on the following equations (Xue et al, 2007):

$$Ku = \frac{4M}{\pi * a} \quad \textbf{(5.3)}$$

$$k_p = 0.6Ku \quad \textbf{(5.4)}$$

$$k_I = 0.5Pu \quad \textbf{(5.5)}$$

$$k_d = 0.125Pu \quad \textbf{(5.6)}$$

5.4.1 Design of the PID controller using 2nd method of Ziegler-Nichols in Matlab/Simulink environment and simulation of the closed loop system

Using Matlab and LabVIEW the sustained oscillation could not be obtained and as such a third party unlicensed version of n^3 control software is used to get the oscillations, see Figure (5.3a) below. The sustained oscillations are shown in Figure (5.3b). The attained ultimate values and the calculated PID parameters are shown in Table (5.4).

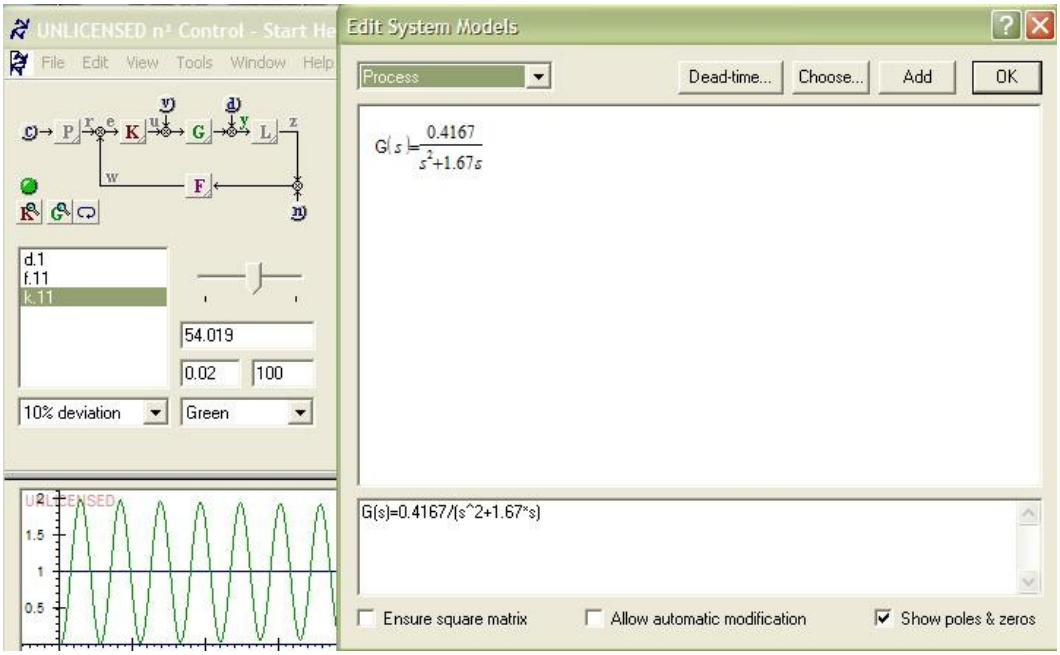


Figure 5.3a: **Sustained oscillations using N^3 Software**

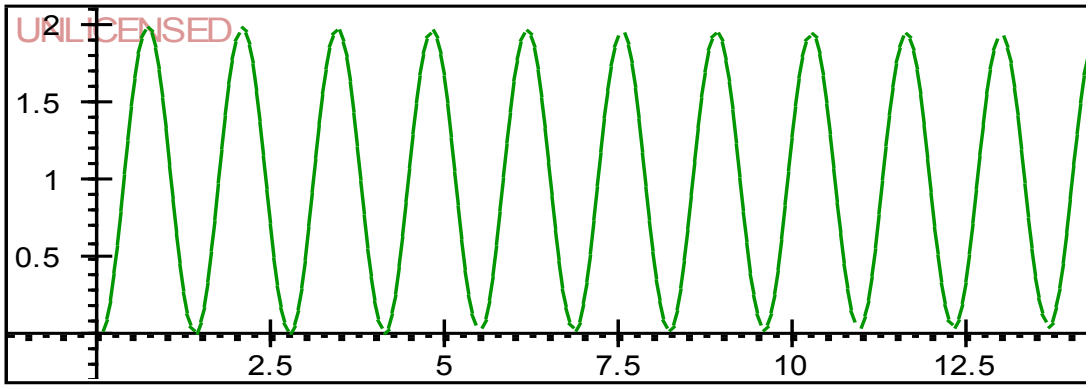


Figure 5.3b: Sustained oscillations using N[3] Software

Table 5.4: Parameters for the PID controller design using 2nd Method of Ziegler Nichols

Description	Symbol	Units	Altitude	Azimuth
Ultimate gain	M	none	54.019	54.019
Ultimate period	a	s	1.96	1.96
Ultimate amplitude	P_u	none	1.4	1.4
Ziegler parameter	$Ku = \frac{4M}{\pi * a}$	None	35.0914	35.0914
Proportional constant	$k_p = 0.6Ku$	none	21.05484	21.05484
Integral constant	$k_I = 0.5Pu$	none	0.7	0.7
Deferential constant	$k_d = 0.125Pu$	none	0.175	0.175
Set Point	$r(t)$	Degrees	60	250

Parameters of the PID controller from Table (5.4) are used to build the closed loop system and investigate the influence of the delays in the communication network.

5.4.2 Simulation of the closed loop system using the PID controller designed by a Ziegler-Nichols Method

This simulation is performed considering ***Plant*** and ***Controller*** with ***constant network induced time delays*** in continuous form. In this simulation the delays are considered between the controller and the actuator and between the sensor and the controller. The purpose of this simulation is to illustrate the influence of the network induced time delay over the system transition behaviour in the case when the controller is not designed to cope with the delays. Table (5.5) shows the values of the time delays used in simulation of the closed loop control system using PID controller parameters designed using Ziegler-Nichols method. The Simulink block diagram is shown in Figure (5.4) for the closed loop control system where the controller is implemented using PID and the plant is represented by the model described in Chapter 4, Equation 4.21. No wind disturbance

is considered. The Matlab file (tf_con_zncePI.m) with data for the simulation can be seen in Appendix B.

The results are shown in Figure (5.5), Figure (5.6) and Table (5.6).

Table 5.5: Delays values used to investigate the influence of the delays over the behavior of the closed loop system using PID controller design using 2nd Method of Ziegler Nichols

Delay Values [sec]	τ_{alt}^{ca}	τ_{alt}^{sc}	τ_{az}^{ca}	τ_{az}^{sc}
$\tau_T = 0$	0	0	0	0
$0 < \tau_T < \tau_c$	0.7	0.8	0.5	0.7
$\tau_T = \tau_c$	2	1.6	1	1.5.
$\tau_T > \tau_c$	3	2.6	1.3	1.7

The set point of the antenna is $r(t) = \sqrt{r_{az}^2 + r_{alt}^2} = 257$.

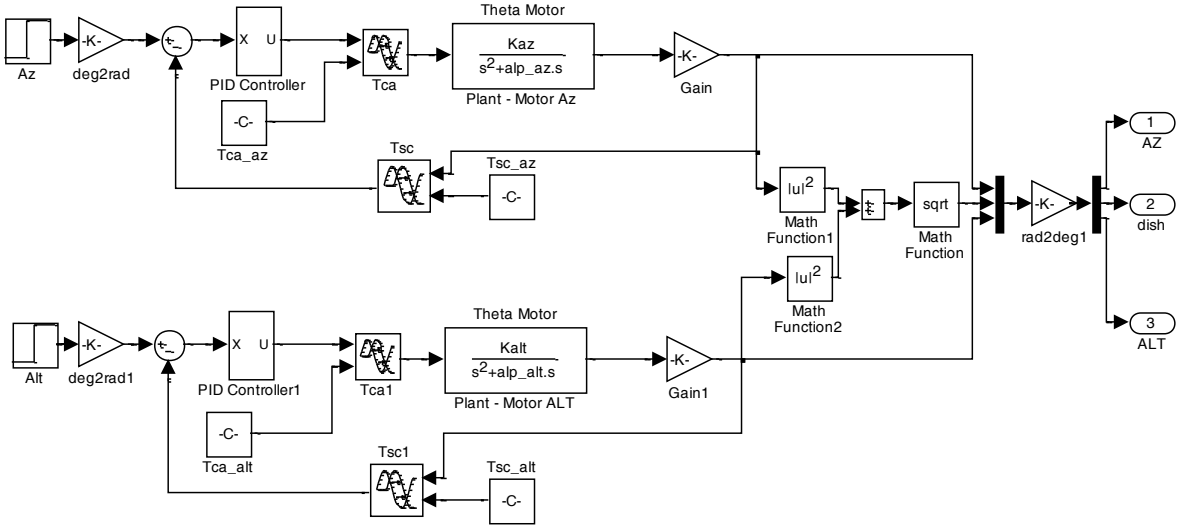


Figure 5.4: Simulink block diagram of a closed loop system - PID controller *tf_con.mdl*

The results from the simulations are shown in Figure (5.5). The Matlab program (tf_con_zncePI.m) with the values of the model and controller parameters is given in Appendix (B5.1).

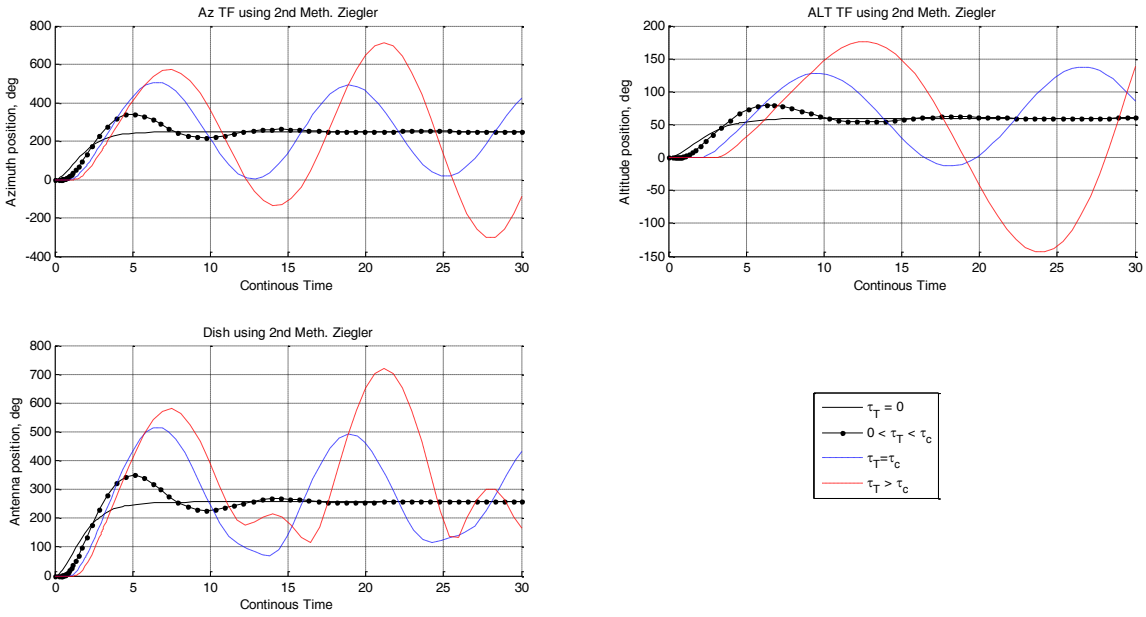


Figure 5.5: PID (2nd Method Ziegler Nichols) controller for different delay values

Figure (5.6) shows a timing diagram zooming at the first seconds of the step response results graphs. This level of zooming allows for second by second analysis which gives a better understanding of the system performance in terms of:

- the introduced time delays
- the delay influence on the measurement output,
- control input after the introduction of the delay between the controller and actuator.

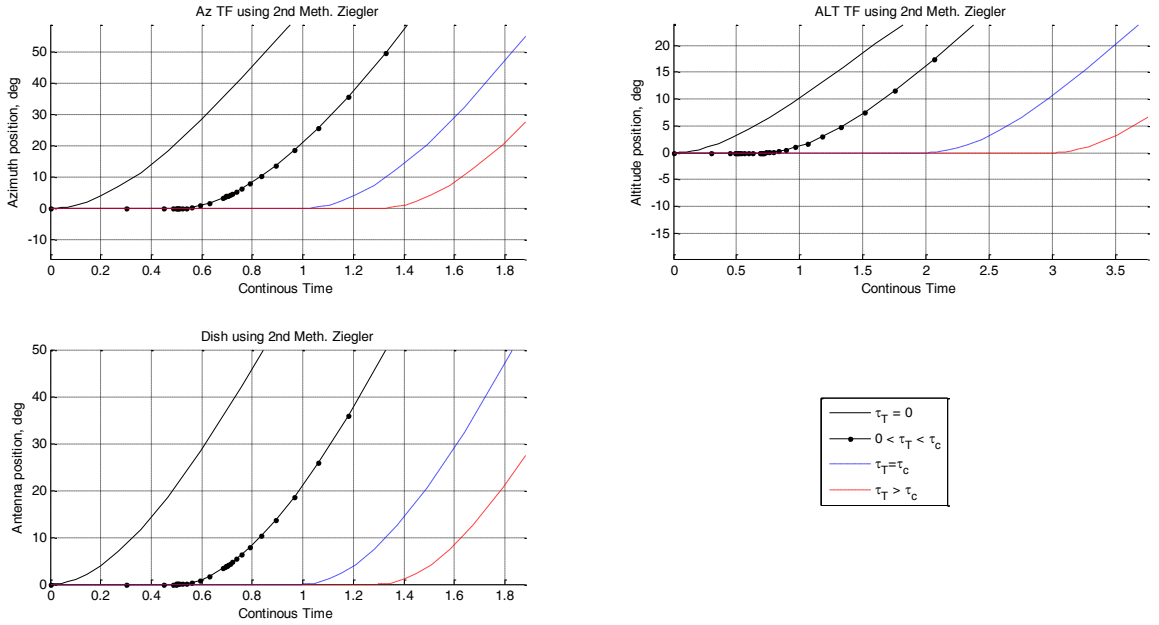


Figure 5.6: Timing diagram considering the four delay influence

Table 5.6: **Performance measures of the closed loop system behavior for the cases of PID controller design using 2nd Method of Ziegler Nichols method**

Performance Measure	2nd Method Ziegler-Nichols		
$\tau_T = 0$			
	Altitude	**Azimuth**	**Antenna**
Rise Time	4.49 sec	3.1 sec	3.1 sec
Settling Time	7.7 sec	5.7 sec	5.7 sec
Percentage Overshoot	0%	0%	0%
Steady State Error	0%	0%	0%
Measured delay value τ_y [sec]	0	0	0
$0 < \tau_T < \tau_c$			
Rise Time	2.4 sec	1.8 sec	1.8 sec
Settling Time	15 sec	18.74 sec	18.74 sec
Percentage Overshoot	32.5%	36%	74%
Steady State Error	0.1%	0.1%	0.1%
Measured delay value τ_y [sec]	0.7 sec	0.5 sec	0.5 sec

Discussion

For the cases of $0 < \tau_T < \tau_c$, $\tau_T = \tau_c$ and $\tau_T > \tau_c$. The plant output is delayed with the delay value equal to τ^{ca}. The time delay between the sensor and the controller τ^{sc} does not influence the value of the delay in the system output.

5.5 PID Controller parameter tuning using pole placement method

Eshbach (1990), considered the pole placement method because of the advantage it has in achieving the desired closed loop pole locations without using pole-zero cancellation and without increasing the order of the system. The desired pole location can be chosen to ensure a desired degree of stability or damping and speed response of the closed loop system. There is no convenient way to ensure in advance that the closed loop system satisfies other performance specifications such as an acceptable disturbance avoidance and desired level of insensitivity to plant parameter variations and compatibility of control effort with actuator limitations.

This means that for robustness of the designed controller is low. The plant considered in the thesis has a lot of disturbances that may cause the system to be unstable and possible not work within the acceptable performance boundaries, that is why this design method is used only for simulation and investigating the influence of the delays.

The algorithm of the pole placement method is:

1. The closed loop transfer function in which the PID controller parameters are unknown is formed. Its characteristics equation is represented as a polynomial according to the powers of the Laplace variables.
2. The desired characteristics equation of the same order as this of the closed loop system characteristics equation is formed on the basis of the required specifications for the process behaviour.
3. The two characteristics equations are compared and the values of the PID controller parameters are calculated.

The pole placements technique is applied by determining the closed loop characteristic equation which is solve by cascading the PID controller and the Plant as shown in Figure (5.1), where the controller is given by equation 5.2 and the plant is given by Equation (4.21).

The transfer function of the closed loop system is given by:

$$G_{cl}(s) = \frac{G_c(s)G_p(s)}{1 + G_c(s)G_p(s)}; \quad \textbf{(5.7)}$$

Where:

$$G_c(s) = \frac{k_d s^2 + k_p s + k_I}{s} \text{ and} \quad \textbf{(5.8)}$$

$$G_p(s) = \frac{K}{s(s+\alpha)} \quad \textbf{(5.9)}$$

The solution for closed loop system is given by Equation (5.11) as:

$$G_{cl}(s) = \frac{\frac{k_d s^2 + k_p s + k_I}{s} * \frac{K}{s(s+\alpha)}}{1 + \frac{k_d s^2 + k_p s + k_I}{s} * \frac{K}{s(s+\alpha)}} \quad \textbf{(5.10)}$$

$$G_{cl}(s) = \frac{Kk_d s^2 + Kk_p s + Kk_I}{s^3 + s^2(\alpha + Kk_d) + Kk_p s + Kk_I} \quad \textbf{(5.11)}$$

From Equation (5.11) is evident that the closed loop system is a third order system, implying that the desired closed loop characteristic equation has three roots. It is selected , that one is real and the other two are complex conjugate. These two are expressed by the desired damping ratio and the closed loop natural frequency.

$$\Delta_{des}(s) = (s-a)(s^2 + 2\varsigma\omega_d s + \omega_d{}^2) \quad \textbf{(5.12)}$$

$$\Delta_{des}(s) = s^3 + s^2(2\varsigma\omega_d + a) + s(\omega_d{}^2 + 2\varsigma\omega_d a) + a\omega_d{}^2 \quad \textbf{(5.13)}$$

The damping ratio (ς) is calculated according to the performance specification, Table (5.3). The desired closed loop damping ratio (ς) is calculated from the percentage overshoot formula (Bishop, 1993) as:

$$\varsigma = \frac{\left|\ln\left(\frac{PO}{100}\right)\right|}{\sqrt{\pi^2 + \left[\ln\left(\frac{PO}{100}\right)\right]^2}} \quad \textbf{(5.14)}$$

Where PO is the Percentage Overshoot.

Based on the damping ratio, the undamped natural frequency is calculated using the Settling Time (Ts) equation (Bishop, 1993):

$$\omega_n = \frac{4}{\varsigma Ts} \quad \textbf{(5.15)}$$

The desired undamped natural frequency is given by:

$$\omega_d = \omega_n\sqrt{1-\varsigma^2} \quad \textbf{(5.16)}$$

The third pole a is selected empirical following the requirement that it has to be bigger than the real parts of the complex conjugate poles in order to assure good stability of the closed loop system.

The PID parameter k_p, k_I and k_d can then be determined by comparing the closed loop polynomial Equation (5.11) with the equation of the desired closed loop characteristic, Equation (5.13).

$$s^3 + s^2(\alpha + Kk_d) + Kk_p s + Kk_I = s^3 + s^2(2\varsigma\omega_d + a) + s({\omega_d}^2 + 2\varsigma\omega_d a) + a{\omega_d}^2 \quad \textbf{(5.17)}$$

PID parameters using pole-placement synthesis based on the plant transfer function are:

$$k_p = \frac{{w_d}^2 + 2\varsigma\omega_d a}{K} \quad \textbf{(5.18)}$$

$$k_I = \frac{a{\omega_d}^2}{K} \quad \textbf{(5.19)}$$

$$k_d = \frac{2\varsigma\omega_d + a - \alpha}{K} \quad \textbf{(5.20)}$$

5.5.1 Simulation of the closed loop system using the PID controller designed by a pole placement method

Simulink model shown in Figure (5.4) is used for this simulation. The difference is the method used to calculate the parameters for the PID controller. Table (5.7) gives the

controller parameters calculated for the azimuth and altitude closed loop system. The results are shown in Figure (5.7) and Figure (5.8). Table (5.7) shows the time delays used in simulation of the closed loop control system using PID controller parameters designed by pole placement method. The characteristics of the transition behaviour of the closed loop system for different values of the time delays are shown in Table (5.8). Comparison of the influence of the delays for the different case of the design methods is shown in Table (5.9).

Table 5.7: Parameters for PID controller design using pole placement method

Description	Symbol	Units	Altitude	Azimuth
Damping Ratio	ς	none	0.7071	0.7071
Third pole	a	S	6.4550	6.4550
Desired natural frequency	ω_d	*rad/s*	1.4	1.4
Proportional constant	k_p	none	29.1532	29.1532
Integral constant	k_I	none	0	0
Differential constant	k_d	none	15.4919	15.4919
Set Point	$r(t)$	Degrees	60	250

Table 5.8: Delays values used to investigate the influence of the delays over the behavior of the closed loop system

Delay Values [sec]	τ_{alt}^{ca}	τ_{alt}^{sc}	τ_{az}^{ca}	τ_{az}^{sc}
$\tau_T = 0$	0	0	0	0
$0 < \tau_T < \tau_c$	0.7	0.8	0.5	0.7
$\tau_T = \tau_c$	1.3	1.5	1.1	1.05
$\tau_T > \tau_c$	1.7	1.4	1.5	1.05

The set point of the antenna is $r(t) = \sqrt{r_{az}^2 + r_{alt}^2} = 257$.

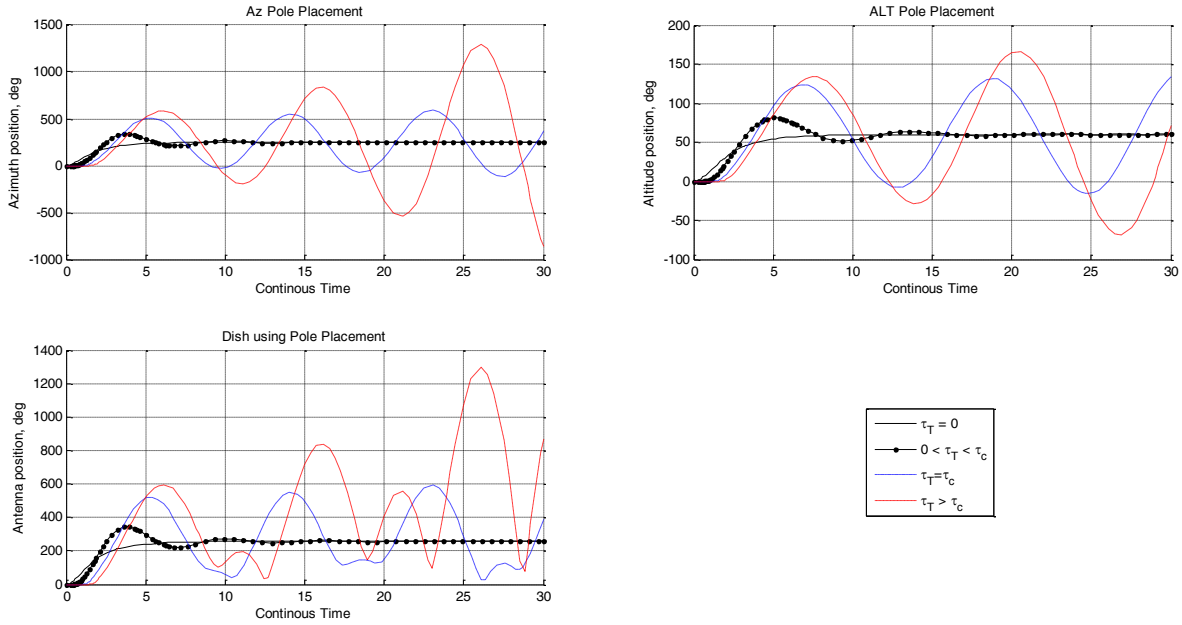


Figure 5.7: Simulation results of the closed loop system with introduced network delay

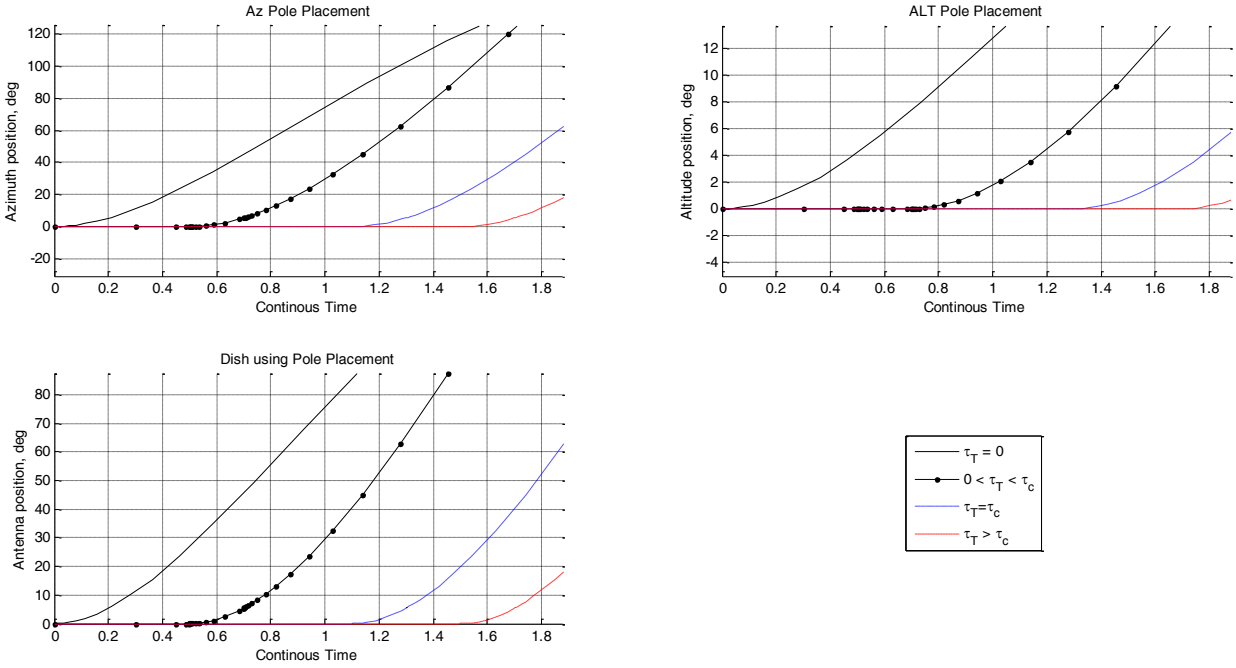


Figure 5.8: Timing diagram considering the four delay influence

Table 5.9: Performance comparison of the closed loop system behavior for the cases of PID controller design using 2nd Method of Ziegler Nichols and the Pole Placement method

Performance Measure	2nd Method Ziegler-Nichols			Pole Placement Method		
$\tau_T = 0$						
	Altitude	**Azimuth**	**Antenna**	**Altitude**	**Azimuth**	**Antenna**
Rise Time	4.49 sec	3.1 sec	3.1 sec	4.4 sec	3.8 sec	3.8 sec
Settling Time	7.7 sec	5.7 sec	5.7 sec	8.5 sec	7.58 sec	7.58sec
Percentage	0%	0%	0%	0%	0%	0%

Overshoot						
Steady State Error	0%	0%	0%	0.1%	0%	0%
τ_y [sec]	0	0	0	0	0	0
$0 < \tau_T < \tau_c$						
Rise Time	2.4 sec	1.8 sec	1.8 sec	2 sec	1.36 sec	1.39sec
Settling Time	15 sec	18.74 sec	18.74 sec	11 sec	14.14 sec	14.14sec
Percentage Overshoot	32.5%	36%	74%	35.5%	34.5%	75%
Steady State Error	0.1%	0.1%	0.1%	0.1%	0.1%	0.1
τ_y [sec]	0.7 sec	0.5 sec	0.5 sec	0.7 sec	0.5 sec	0.5sec

Discussion

For the case of $\tau_T = 0$, the plant output is not delayed and the system is critically damped and performs according to the specification.

For the cases of $0 < \tau_T < \tau_c$, $\tau_T = \tau_c$ and $\tau_T > \tau_c$, the plant output is delayed with the delay value equal to τ^{ca}.

The PID controller methods designed using Pole Placement is giving better transition behaviour than the PID controller method designed using 2nd Method of Ziegler-Nichols.

5.6 State space proportional controller design using pole placement method

This control method is based on the state feedback gain matrix *L*. State feedback gain matrix $[L]^{2x1}$ is calculated to get the system to the desired closed loop poles. The matrix is calculated by first getting the state space representation of the open loop transfer function as shown in Equation (5.21). The open loop state space model is then rewritten in a closed loop state space model incorporating the *L* matrix. The characteristic equation of the desired poles is determined and compared to the characteristic equation of the closed loop control system. The *L* matrix is then calculated by comparison of the two characteristics equations.

The state space model of the plant is:

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -\alpha \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) \quad \textbf{(5.21)}$$

$$y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} \quad \textbf{(5.22)}$$

where: $x_1(t) = \theta_m, x_2(t) = \dot{\theta}_m$

The state space controller has the form:

$u(t) = Fr(t) - Lx(t)$, Where F s a coefficient to be determined by tuning.

Determinant for the closed loop control system is given by:

$$\Delta_{cl}(s) = \left| sI - (A - BL) \right| = 0 \quad \textbf{(5.23)}$$

where: $A = \begin{bmatrix} 0 & 1 \\ 0 & -\alpha \end{bmatrix}$, $B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, $L = \begin{bmatrix} l_1 & l_2 \end{bmatrix}$

Substituting ***A***, ***B***, and ***L*** and solving Equation (5.23) results in:

$$\Delta_{cl}(s) = \left| \begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix} - \left[\begin{bmatrix} 0 & 1 \\ 0 & -\alpha \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ l_1 & l_2 \end{bmatrix} \right] \right| = 0 \quad \textbf{(5.24)}$$

$$\Delta_{cl}(s) = \left| \begin{bmatrix} s & -1 \\ l_1 & s + \alpha + l_2 \end{bmatrix} \right| = 0 \quad \textbf{(5.25)}$$

$$\Delta_{cl}(s) = s^2 + s(\alpha + l_2) + l_1 \quad \textbf{(5.26)}$$

The characteristic equation for the desired closed loop control system is formed on the basis of two complex conjugate desired poles. It has the following expression:

$$\Delta_d(s) = s^2 + 2\varsigma\omega_d s + \omega_d{}^2 \quad \textbf{(5.27)}$$

The characteristics equation of the closed loop system Equation (5.26) and the characteristics equation of the desired closed loop system given by Equation (5.27) are compared to obtain l_1 and l_2:

$$s^2 + s(\alpha + l_2) + l_1 = s^2 + 2\varsigma\omega_d s + \omega_d{}^2 \quad \textbf{(5.28)}$$

$$l_1 = \omega_d{}^2 \quad \textbf{(5.29)}$$

$$l_2 = 2\varsigma\omega_d - \alpha \quad \textbf{(5.30)}$$

where: ω_d and ς are calculated by the Equations (5.15) and (5.14) respectively.

5.6.1 Simulation of the closed loop state space system with the designed proportional state space controller

Figure (5.6) shows the Simulink block diagram of the closed loop state space system where the controller is represented with the gain matrix. Table (5.10) shows the parameters used in simulation of the closed loop control system. Investigation of the influence of the delays is for the values shown in Table (5.11). The results from the simulations are shown in Figure (5.9) and (5.10). The timing diagram in Figure (5.10) allow the influence of the delay at the beginning of the transition behaviour to be measured. The Matlab program (ss_con_ppP.m) with the values of the model and controller parameters is given in Appendix (B5.2).

Table 5.10: **Parameters for design of the proportional state space controller design using Pole Placement method**

Description	Symbol	Units	Azimuth	Altitude
Desired Damping Ratio	ς	none	0.7071	0.7071
Desired natural frequency	ω_d	rad/s	0.9989	0.9989
Proportional parameter h_1	l_1	none	0.9977	0.9977
Proportional parameter h_2	l_2	none	-0.2880	0.0537
Set Point	$r(t)$	Degrees	60	250

Table 5.11: **Delays values for investigation of the influence of the delay over the behavior of the closed loop system**

Delay Values [sec]	τ_{alt}^{ca}	τ_{alt}^{sc}	τ_{az}^{ca}	τ_{az}^{sc}
$\tau_T = 0$	0	0	0	0
$\tau_T > 0$	0.5	0.3	0.4	0.6
$\tau_T = \tau_c$	0.82	0.9	0.9	1
$\tau_T > \tau_c$	1.1	1.03	1.1	1.05

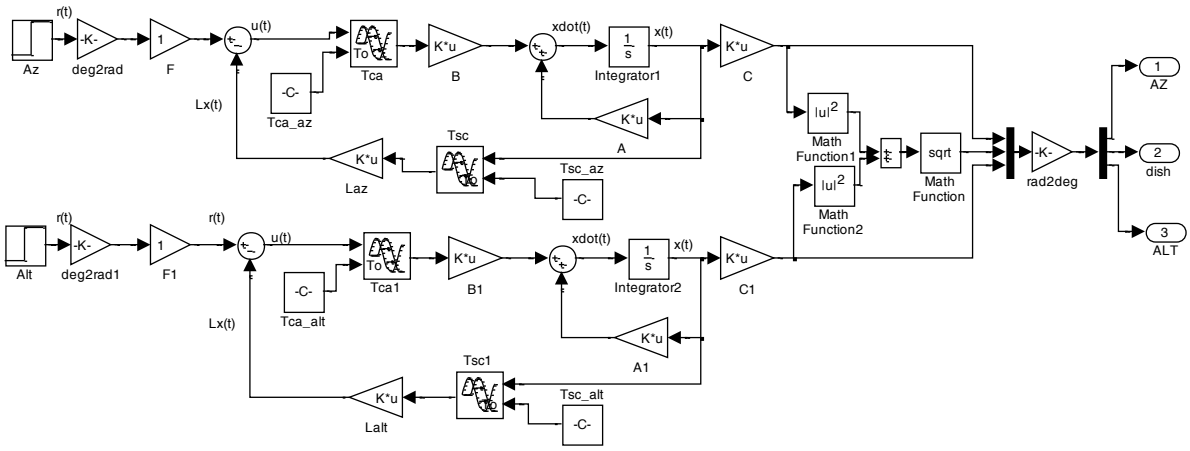


Figure 5.9: **Simulink block diagram of a closed loop system - Proportional SS controller *ss_con_ppP.mdl***

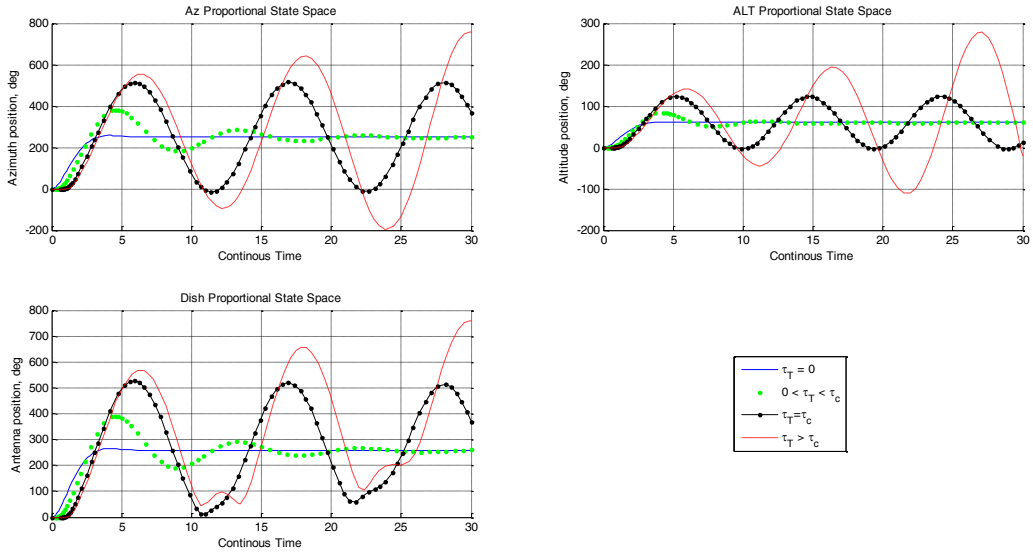


Figure 5.10: **Simulation results for the behaviour of the closed loop system with introduced network delays**

Transition behaviour of the closed system with Proportional state space controller for different values of the network delays.

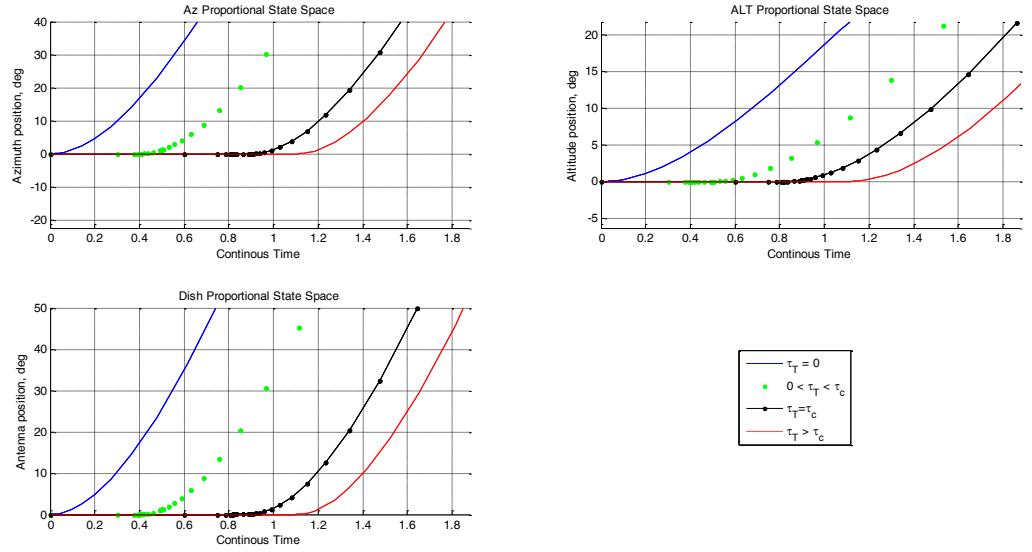


Figure 5.11: **Timing diagram considering the four delays influence**

Table 5.12: **Transition behavior characteristics. Measured delays in the plant output.**

Performance Measure	**State Space - Proportional Controller**		
$\tau_T = 0$			
	Altitude	**Azimuth**	**Antenna**
Rise Time	2.1sec	2.1sec	2.1sec
Settling Time	6.1sec	4.2sec	4.9sec
Percentage Overshoot	3.3%	3.05%	3.3%
Steady State Error	0.4%	0.4%	0.4%
Measured delay value τ_y [sec]	0sec	0sec	0sec

$0 < \tau_T < \tau_c$			
Rise Time	3.9sec	3.7sec	3.9sec
Settling Time	15sec	25.3sec	26.1sec
Percentage Overshoot	40.2%	52%	52%
Steady State Error	2%	1.5%	1.2%
Measured delay value τ_y [sec]	0.5sec	0.4sec	0.4sec

Discussion

The simulations show a steady state error that can not be reduced by the gain matrix alone. For the case of $\tau_T = 0$. The plant output is not delayed.

For the cases of $0 < \tau_T < \tau_c$, $\tau_T = \tau_c$ and $\tau_T > \tau_c$, the plant output is delayed with the delay value between the controller and the actuator τ^{ca}.

5.7 State space PI controller design using pole placement method

The Proportional state space controller could not reduce the steady state error. PI state space controller that can reduce steady state error is presented in this section using the pole placement method. Equations (5.31) and (5.32) present an extended closed loop state space control system incorporating an integral component (Nise, 2004).

$$\begin{bmatrix} \dot{x}(t) \\ \dot{e}(t) \end{bmatrix} = \begin{bmatrix} A & 0 \\ -C & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ e(t) \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} r(t) \quad \textbf{(5.31)}$$

$$y(t) = \begin{bmatrix} C & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ e(t) \end{bmatrix} \quad \textbf{(5.32)}$$

Where: $A = \begin{bmatrix} 0 & 1 \\ 0 & -\alpha \end{bmatrix}, B \begin{bmatrix} 0 \\ K \end{bmatrix}, C = \begin{bmatrix} 1 & 0 \end{bmatrix}, e(t) = r(t) - y(t) = r(t) - Cx(t)$

Substituting *A, B, C,* in Equation (5.31) results in a closed loop state space model.

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{e}(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -\alpha & 0 \\ -1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ e(t) \end{bmatrix} + \begin{bmatrix} 0 \\ K \\ 0 \end{bmatrix} u(t) + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} r(t) \quad \textbf{(5.32)}$$

$$y(t) = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ e(t) \end{bmatrix} \quad \textbf{(5.33)}$$

$$u(t) = -[l_1 \quad l_2 \quad l_e] \begin{bmatrix} x_1(t) \\ x_2(t) \\ e(t) \end{bmatrix} \quad \textbf{(5.34)}$$

$$u(t) = u_1(t) + u_2(t) \tag{5.35}$$

$$u_1(t) = -[l_1 \quad l_2]\begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}; u_2(t) = +l_2 e(t) \tag{5.36}$$

Where: $x_1(t) = \theta_m, x_2(t) = \dot{\theta}_m, u(t) = Fr(t) - Lx(t) + l_e e(t)$

Determinant for the closed loop control system is given by:

$$\Delta_{cl}(s) = |sI - (A - BL)| = 0 \tag{5.37}$$

Where: $A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -\alpha & 0 \\ -1 & 0 & 0 \end{bmatrix}$, $B = \begin{bmatrix} 0 \\ K \\ 0 \end{bmatrix}$, $L = \begin{bmatrix} l_1 & l_2 & l_e \end{bmatrix}$

Substituting A, B, and L and solving Equation (5.37) results in:

$$\Delta_{cl}(s) = \left| \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & s \end{bmatrix} - \left[\begin{bmatrix} 0 & 1 & 0 \\ 0 & -\alpha & 0 \\ -1 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 0 & 0 & 0 \\ Kl_1 & Kl_2 & Kl_e \\ 0 & 0 & 0 \end{bmatrix} \right] \right| = 0 \tag{5.38}$$

$$\Delta_{cl}(s) = \left| \begin{bmatrix} s & -1 & 0 \\ Kl_1 & s + \alpha + Kl_2 & Kl_e \\ 1 & 0 & s \end{bmatrix} \right| = 0 \tag{5.39}$$

$$\Delta_{cl}(s) = s^3 + s^2(\alpha + Kl_2) - sKl_1 + Kl_e \tag{5.40}$$

The characteristic equation for the desired closed loop control system is formed on the basis of two complex conjugate desired poles. It has the following expression:

$$\Delta_d(s) = s^3 + s^2(2\varsigma\omega_d + a) + s({\omega_d}^2 + 2\varsigma\omega_d a) + a{\omega_d}^2 \tag{5.41}$$

The characteristics equation of the closed loop system Equation (5.40) and the characteristics equation of the desired closed loop system given by Equation (5.41) are compared to obtain l_1 and l_2 and l_e as shown in Equation (5.42):

$$s^3 + s^2(\alpha + l_2) + sl_1 + Kl_e = s^3 + s^2(2\varsigma\omega_d + a) + s({\omega_d}^2 + 2\varsigma\omega_d a) + a{\omega_d}^2 \tag{5.42}$$

The solution for the parameters for controller gain matrix are:

$$l_1 = -\frac{{\omega_d}^2 + 2\varsigma\omega_d a}{K} \tag{5.43}$$

$$l_2 = \frac{2\varsigma\omega_d + a - \alpha}{K} \tag{5.44}$$

$$l_e = \frac{a{\omega_d}^2}{K} \tag{5.45}$$

Where: ω_d and ς are calculated according to Equations (5.15) and (5.14) respectively.

5.7.1 Simulation of the closed loop state space system using the PI state space controller

Figure (5.12) shows the Simulink block diagram of the closed loop system with a PI state space controller. Table (5.13) shows the parameters used in the simulation of the closed loop control system. The results are shown in Figure (5.13) and (5.14) and performance index is shown in Table (5.14). The Matlab program (ss_con_ppPI.m) with the values of the model and controller parameters is given in Appendix (B5.4).

The values of the delays to be used for the delay influence investigation are given in Table (5.13)

Table 5.13: Parameters for design of the closed loop state space system using PI state space controller

Description	Symbol	Units	Altitude	Azimuth
Zeta	ξ	none	0.7071	0.7071
Desired natural frequency	ω_d	rad/s	0.9989	0.9989
Third pole	a	s	0.75	0.9682
Proportional parameter h_1	l_1	none	2.0317	2.3326
Proportional parameter h_2	l_2	none	0.8037	0.6802
Proportional parameter h_e	l_e	none	2.9932	2.3185

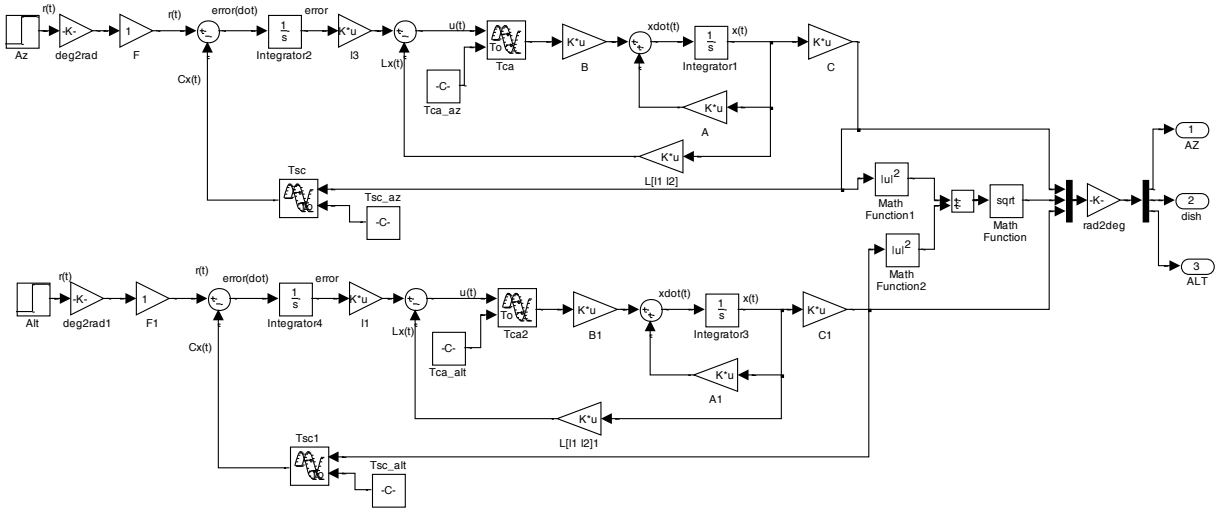


Figure 5.12: Simulink block diagram of a closed loop system using Proportional and Integral state space controller
s_con_ppPI.mdl

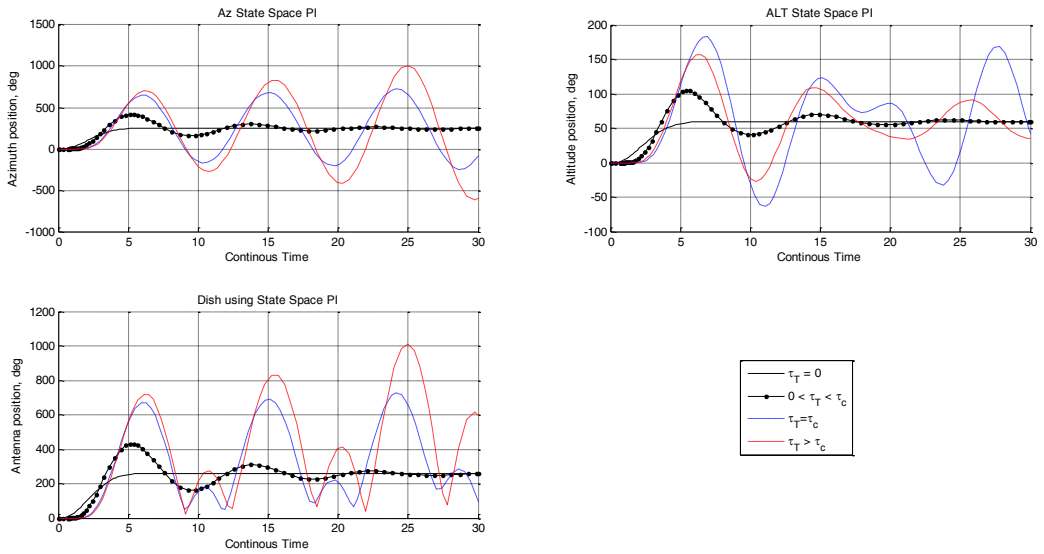


Figure 5.13: Transition behaviour of the closed loop state space system for different values of the time delays

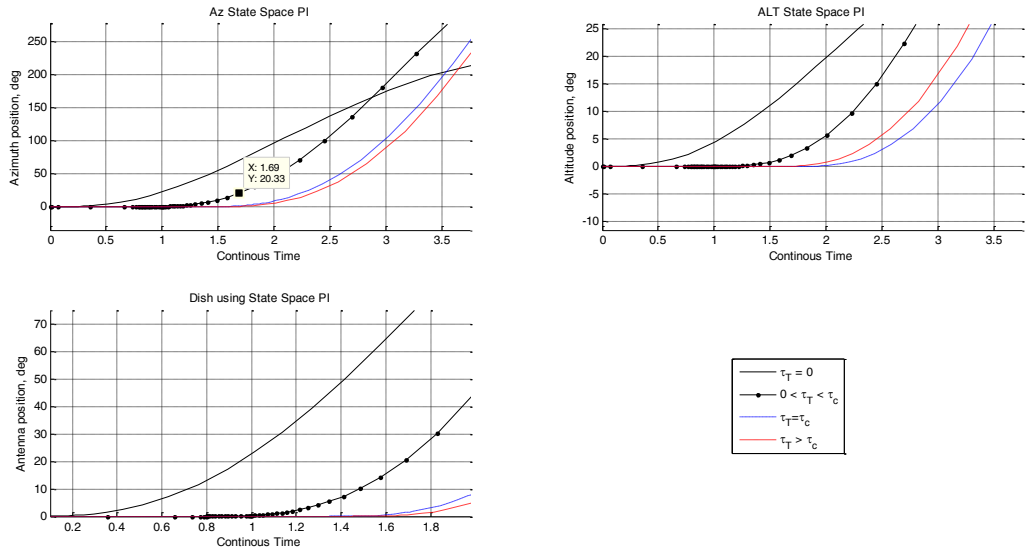


Figure 5.14: Timing diagram using different delay values

Table 5.14: Performance indices for of the closed loop state space with PI controller system

Performance Measure	Proportional Integral (PI) Controller		
$\tau_T = 0$			
	Altitude	**Azimuth**	**Antenna**
Rise Time	3.2sec	2.98sec	2.8sec
Settling Time	5.8sec	5.5sec	5.1sec
Percentage Overshoot	0%	0%	0%
Steady State Error	0%	0%	0%
Measured delay value τ_y [sec]	0sec	0sec	0sec
$0 < \tau_T < \tau_c$			
Rise Time	1.582sec	1.253sec	3.254sec
Settling Time	25.5sec	21.3sec	27.5sec

Percentage Overshoot	67.4%	74.6%	67.6%
Steady State Error	0.05%	0.04%	0.01%
Measured delay value τ_y [sec]	1sec	0.8sec	0.9sec

Discussion:

The steady state error is removed by the Proportional and Integral controller. For the case of $\tau_T = 0$. The plant output is not delayed.

For the cases of $0 < \tau_T < \tau_c$, $\tau_T = \tau_c$ and $\tau_T > \tau_c$, the plant output is delayed with the delay value between the controller and the actuator τ^{ca}.

5.8 Conclusion

The following conclusion is drawn from the investigation of the influence of network induced time delays using the controllers that are **not** designed to compensate for the network induced time delay:

- Different methods of controller design responds differently from network induced time delay with each type of controller design there exists a different critical delays that is where the system starts to oscillates.
- The state space closed loop system using proportional controller behaves better; however the main problem is the steady state error that is constantly visible with or without the network induced time delay.
- Ziegler Nichols controller method behaves better on smaller time delays but it could not cope with larger time delays.
- It has been observed that at the summing point just before the controller actuator delay τ^{ca} the control is not delayed.
- The dish measurement output is delayed by the delay value between the controller and the actuator. On the basis of the above a conclusion can be done that the delay between the controller and the actuator can be used for the design of the predictive controller and the delay between the sensor and the controller can be used for the design of the predictive Kalman filter. Investigations will be done also for the sum of the two delays to be used for the design of the predictive controller and the Kalman filter to be designed for non delayed measurements.

The next chapter described the derivations of the Discrete Kalman Filter as used in the development of the Predictive Discrete Kalman Filter (PDKF). The PDKF is developed considering the network induced time delays only. The considered delay during the development of the PDKF is between the sensor and the controller.

CHAPTER SIX
DERIVATION OF DISCRETE KALMAN FILTER

6.1 Introduction

Kalman published (Kalman, 1960) the recursive algorithm of the discrete filter for linear systems. The filter is represented by a set of equations which determine a recursive algorithm for estimation of the state of the process such that the mean of the square error between the real and estimated states is minimised. The filter supports estimation of past, present and future states even when the system under consideration is not known fully. Timing and measurement assumptions can be considered:

- The measurement times are synchronised.
- The sensor measurements are part of the sensor data packet that includes the measurements, the sensor ID, and the measurement time stamp.
- The time period required for a sensor data packet to be transmitted from a sensor to another device is fixed. This time period is the transmission time. Communication time can be different.

The development of this chapter are as follows:

- Derivation of the Kalman filter is described in section 6.2.
- Derivation of the Kalman filter for the case of delay on measurement data is described in section 6.3.
- The conclusion is given in section 6.4.

6.2 Derivation of the Kalman filter

Practical introduction of the discrete Kalman filter is done in (Kalman, 1960) where the following considerations are used for the derivation of the filter:

6.2.1 The process state model

The process state to be estimated is described by a linear difference stochastic Equation (6.1) and (6.2). For the purpose of the design of the Kalman filter the process equation is considered one step backwards:

$$x(k) = Ax(k-1) + Bu(k-1) + w(k-1), x \in R^n, x(0) = x_0 \quad \textbf{(6.1)}$$

with measurement $y \in R^l$

$$y(k) = Cx(k) + v(k) \quad \textbf{(6.2)}$$

The variables $w(k)$ and $v(k)$ represent the process and the measurement noise respectively. These variables are assumed independent on each other, white and having normal probability distributions of:

$$P(w) \approx N(0,\Sigma),\ or\ \mathrm{E}\{w(k), w^T(g)\} = \begin{Bmatrix} 0, k \neq g \\ \Sigma(k), k = g \end{Bmatrix}$$
$$P(v) \approx N(0,Q),\ or\ \mathrm{E}\{v(k), v^T(g)\} = \begin{Bmatrix} 0, k \neq g \\ Q(k), k = g \end{Bmatrix} \quad \textbf{(6.3)}$$

where Σ is the process noise covariance and Q is the measurement noise covariance matrices, which are assumed to be constant. These matrices can change with every step of measurement, $P(w)$ and $P(v)$ are the probability distributions. The requirements for the noise characteristics preserve the diagonal structure of the covariance matrices. Further the noises $w(k)$, $v(k)$ and the initial state are not correlated and $\mathrm{E}\{v(k)w^T(k)\} = 0, \vee k$. The values of the covariance matrices Σ and Q represent the uncertainly in the state and output measurement vectors. The initial state covariance matrix $\mathrm{E}[x(0), x^T(0)] = P_0 \geq 0$.

$A \in R^{nxm}$ is a constant state matrix, but can be changed with every step **k**. $B \in R^{nxm}$ relates the control input at moment **k-1** to the state at the moment **k**, $C \in R^{lxn}$, is assumed to be a constant matrix.

6.2.2 Formulation of the problem for design of the Kalman filter

The problem is to find the values of the matrix of the Kalman gain at every moment of time in such a way that the variances of state estimates from their true values are minimised according to the criterion

$$J = \frac{1}{2}\mathrm{E}\{[\hat{x}(k) - x(k)]^2\} \longrightarrow \min \quad \textbf{(6.4)}$$

where $\hat{x}(k)$ is the estimate of the state vector at the moment **k** obtained by the Kalman filter.

6.2.3 Computation basis of the filter

1 It is assumed that the knowledge for the process prior to step **k** is given.

2 Then it is defined that $\hat{x}^-(k) \in R^n$ is estimated on the basis of this given knowledge and is called a priory state estimate.

3 $\hat{x}(k)$ is determined at step k using the given measurement y(k) and is called a posteriory state estimate.

4 A priory estimate error is defined as:

$$e^-(k) = \hat{x}^-(k) - x(k) \text{ and } e^-(k-1) = \hat{x}^-(k-1) - x(k-1) \quad \textbf{(6.5)}$$

5 A posteriory estimate error is defined as

$$e(k) = \hat{x}(k) - x(k) \text{ and } e(k-1) = \hat{x}(k-1) - x(k-1) \quad \textbf{(6.6)}$$

6 A priory estimate error covariance is calculated as

$$P^{-}(k) = \mathrm{E}\left\{e^{-}(k)e^{-}(k)^{T}\right\} \text{ and } P^{-}(k-1) = \mathrm{E}\left\{e^{-}(k-1)e^{-}(k-1)^{T}\right\} \quad \textbf{(6.7)}$$

7 A posteriory estimate error covariance is calculated as

$$P(k) = \mathrm{E}\left\{e(k)e(k)^{T}\right\} \text{ and } P(k-1) = \mathrm{E}\left\{e(k-1)e(k-1)^{T}\right\} \quad \textbf{(6.8)}$$

8 A priory estimate $\hat{x}^{-}(k)$ is obtained from the model Equation (6.9).

It is supposed that $\hat{x}(k-1)$ is known and its update $\hat{x}^{-}(k)$ is

$$\hat{x}^{-}(k) = A\hat{x}(k-1) + Bu(k-1) \quad \textbf{(6.9)}$$

9 A posteriory and a priory estimates are connected by the equation:

$$\hat{x}(k) = \hat{x}^{-}(k) + K(k)[y(k) - C\hat{x}^{-}(k)] \quad \textbf{(6.10)}$$

This means that the posteriory estimate is calculated on the basis of the value of the priory estimate and added to it the value of the weighted difference between the actual measurement y(k) and the prediction of the measurement $C\hat{x}^{-}(k)$. The difference $y(k) - C\hat{x}^{-}(k)$ is called the measurement innovation or the residual. It shows the error between the real measurement and the estimated measurement. The matrix $K(k) \in R^{nxl}$ is called the gain, or the blending factor that minimises the aposteriori error covariance $P(k) = \mathrm{E}\left\{e(k)e^{T}(k)\right\}$. $P(k)$ is a measure of the value of the estimation error. A zero covariance matrix $P(k)$ means that the state estimate $\hat{x}(k)$ is the perfect one. If the diagonal elements of $P(k)$ have big values, the estimate is very poor.

6.2.4 Derivation of the Kalman gain

On the basis of the above, it follows that it is necessary to derive optimal expression for $P^{-}(k)$ and $P(k)$, because in this way the considered errors between the state values and their estimates will be minimised. The expression for P^{-} is based on the process Equation (6.9) and it does not depend on the Kalman gain. Because of this, only the minimisation of the error covariance matrix $P(k)$ has to be considered. The objectives for design of the Kalman filter are:

- To show that the estimated state covariance matrix is bounded.
- To show that the bias errors of the estimated state mean vector are bounded.

- Determine the effect of various timing measurement errors on the bias error of the estimated state mean vector.

The covariance matrix of the estimated state can be considered as a statistical description of the state mean vector estimation error and can be used to check if the estimation error have bounded variance. If the bias errors of the estimated state mean vector are bounded, then the estimated state mean vector converges to the true value of the state mean vector. The bounded state covariance matrix is a necessary condition for a bounded bias error of the estimated state mean vector. The standard Kalman filter assumes that the model of the plant and the statistics of the noises are known fully. The Kalman gain is derived based on requirement that the remaining errors $e(k)$ should be uncorrelated.

The are many derivations of the equations of the Kalman filter. The thesis proposes the following one :

1 Derivation of $P^{-}(k)$

The Equation (6.9) is substituted into Equation (6.5)

$$\begin{aligned}\bar{e}(k) &= \hat{x}^{-}(k) - x(k) = \left[A\hat{x}(k-1) + Bu(k-1)\right] - \left[Ax(k-1) + Bu(k-1) + w(k-1)\right] \\ &= A\left[\hat{x}(k-1) - x(k-1)\right] - w(k-1) = Ae(k-1) - w(k-1)\end{aligned} \quad \textbf{(6.11)}$$

The Equation (6.11) is substituted into Equation (6.7) in order to calculate $P^{-}(k)$.

$$\begin{aligned}P^{-}(k) &= \mathrm{E}\left\{e^{-}(k)e^{-}(k)^{T}\right\} = \mathrm{E}\left\{\left[Ae(k-1) - w(k-1)\right]\left[Ae(k-1) - w(k-1)\right]^{T}\right\} \\ &= \mathrm{E}\left\{Ae(k-1)e(k-1)^{T}A^{T}\right\} - \mathrm{E}\left\{Ae(k-1)w(k-1)^{T}\right\} - \mathrm{E}\left\{w(k-1)Ae(k-1)^{T}A^{T}\right\} + \\ &+ \mathrm{E}\left\{w(k-1)w(k-1)^{T}\right\}\end{aligned} \quad \textbf{(6.12)}$$

It could be seen that $w(k-1)$ and $e(k-1)$ are not correlated as $x(k)$, not $e(k)$ directly depends on $w(k-1)$. This means that

$$\mathrm{E}\left\{e(k-1)w(k-1)^{T}\right\} = 0 \text{ and } \mathrm{E}\left\{w(k-1)e(k-1)^{T}\right\} = 0 \quad \textbf{(6.13)}$$

In addition according to Equations (6.3) and (6.8)

$$\mathrm{E}\left\{w(k-1)w(k-1)^{T}\right\} = \sum(k-1) \text{ and } \mathrm{E}\left\{e(k-1)e(k-1)^{T}\right\} = P(k-1) \quad \textbf{(6.14)}$$

Then the Equation (6.12) becomes

$$P^{-}(k) = A^{T}P(k-1)A^{T} + \sum(k-1) \quad \textbf{(6.15)}$$

2 Derivation of the optimal expression for $P(k)$

Equation (6.2) is substituted into Equation (6.10)

$$\begin{aligned}\hat{x}(k) &= \hat{x}^{-}(k) + K(k)[Cx(k) + v(k) - C\hat{x}^{-}(k)] \\ &= \left[I - K(k)C\right]\hat{x}^{-}(k) + K(k)Cx(k) + K(k)v(k)\end{aligned} \quad \textbf{(6.16)}$$

Equation (6.16) is substituted into Equation (6.6) to calculate the error $e(k)$.

$$\begin{aligned} e(k) &= \hat{x}^-(k) - x(k) = [I - K(k)C]\hat{x}^-(k) + K(k)Cx(k) + K(k)v(k) - x(k) \\ &= [I - K(k)C]\hat{x}^-(k) + K(k)Cx(k) + K(k)v(k) - Ix(k) - K(k)Cx(k) + K(k)Cx(k) \\ &= [I - K(k)C][\hat{x}^-(k) - x(k)] + K(k)v(k) = [I - K(k)C]e^-(k) + K(k)v(k) \end{aligned} \quad \textbf{(6.17)}$$

$P(k)$ can be calculated on the basis of Equation (6.8)

$$\begin{aligned} P(k) &= \mathrm{E}\{[[I - K(k)C]e^-(k) + K(k)v(k)][[I - K(k)C]e^-(k) + K(k)v(k)]^T\} \\ &= \mathrm{E}\{[I - K(k)C]e^-(k)e^-(k)^T[I - K(k)C]^T\} + \mathrm{E}\{[I - K(k)C]e^-(k)v^T(k)K(k)^T\} \\ &+ \mathrm{E}\{K(k)v(k)e^-(k)^T[I - K(k)C]^T\} + \mathrm{E}\{K(k)v(k)v^T(k)K(k)\} \end{aligned} \quad \textbf{(6.18)}$$

It can be seen that $v(k)$ and $e^-(k)$ are not correlated as $e(k)$, not $e^-(k)$ directly depends on $v(k)$.Then

$$\mathrm{E}\{e^-(k)v^T(k)\} = 0 \text{ and } \mathrm{E}\{v(k)e^-(k)^T\} = 0 \quad \textbf{(6.19)}$$

In addition $\mathrm{E}\{v(k)v^T(k)\} = Q(k)$ and $\mathrm{E}\{e^-(k)e^{-T}(k)\} = P^-(k)$ **(6.20)**

according to Equation (6.3) and (6.7). Then Equation (6.18) becomes:

$$P(k) = [I - K(k)C]P^-(k)[I - K(k)C]^T + K(k)Q(k)K^T(k) \quad \textbf{(6.21)}$$

3 Derivation of the optimal value of $K(k)$

Equation (6.21) is another expression for the criterion Equation (6.4). It is necessary to minimise the error covariance according to the Kalman gain K(k) which is unknown. The criterion Equation (6.4) can be expressed in the following way

$$J[K(k)] = trace(P(k)) = Trace\{[I - K(k)C]P^-(k)[I - K(k)C]^T + K(k)Q(k)K^T(k)\}, \quad k = 1, 2, \ldots \quad \textbf{(6.22)}$$

The optimal value of K(k) can be obtained on the basis of the necessary conditions for optimality of the criterion Equation (6.22), as follows

$$\frac{\partial J[K(k)]}{\partial K(k)} = 0 \quad \textbf{(6.23)}$$

The derivative in Equation (6.23) can be expressed following the trace dependencies

$$\frac{\partial Tr(DBD^T)}{\partial D} = D(B + B^T) \quad \textbf{(6.24)}$$

Following Equation (6.23) it is obtained

$$\frac{\partial J[K(k)]}{\partial K(k)} = -[I - K(k)C][P^-(k) + P^{-T}(k)]C^T + K(k)[Q(k) + Q(k)^T] = 0 \quad \textbf{(6.25)}$$

In Equation (6.25) $P^{-}(k) = P^{-T}(k)$ and $Q(k) = Q^{T}(k)$ as they are symmetrical matrices. The solution of Equation (6.25) is as follows:

$$-P^{-}(k)C^{T} + K(k)CP^{-}(k)C^{T} + K(k)Q(k) = 0 \quad \textbf{(6.26)}$$

$$K(k)\left[CP^{-}(k)C^{T} + Q(k)\right] = P^{-}(k)C^{T} \quad \textbf{(6.27)}$$

$$K(k) = P^{-}(k)C^{T}\left[CP^{-}(k)C^{T} + Q(k)\right]^{-1} \quad \textbf{(6.28)}$$

4 Calculation of P(k)

The optimal value of *K(k)* can be used to calculate the optimal value of *P(k)* by substitution of the value of *K(k)* into Equation (6.21). Then

$$\begin{aligned} P(k) &= \left[I - K(k)C\right]P^{-}(k)\left[I - K(k)C\right]^{T} + K(k)Q(k)K^{T}(k) = \\ &= P^{-}(k) - K(k)CP^{-}(k) - P^{-}(k)C^{T}K^{T}(k) + K(k)\left[CP^{-}(k)C^{T} + Q(k)\right]K^{T}(k) \\ &= P^{-}(k) - K(k)CP^{-}(k) \\ &= \left[I - K(k)C\right]P^{-}(k) \end{aligned} \quad \textbf{(6.29)}$$

It can be seen from Equation(6.29) that the update stage of the Kalman filter decreases the covariance of the estimation error, while the propagation stage increases the propagation covariance, Equation (6.15)

5 Recursive form of the Kalman filter

The propagation and the measurement update equations can be combined together to form a recursive form of the Kalman filter. To achieve this the equation for calculation of $\hat{x}(k)$, Equation (6.10) is substituted in Equation (6.9) for calculation of $\hat{x}^{-}(k)$, and Equation (6.28) is substituted into Equation (6.15). The obtained estimation algorithm is

$$\hat{x}^{-}(k) = A\hat{x}^{-}(k-1) + Bu(k-1) + AK(k-1)\left[y(k-1) - C\hat{x}^{-}(k-1)\right] \quad \textbf{(6.30)}$$

$$K(k) = P^{-}(k)C^{T}\left[CP^{-}(k)C^{T} + Q(k)\right]^{-1} \quad \textbf{(6.31)}$$

$$P^{-}(k) = A^{T}P(k-1)A^{T} + AK(k)CP(k-1)A^{T}) + \sum(k-1) \quad \textbf{(6.32)}$$

where Equation (6.32) is the discrete Riccati equation of the Kalman filter.

6 Algorithm of the Kalman filter

The filter estimates the state using a form of a feedback control

- the process state is estimated at some time
- the filter estimate is obtained using the noisy measurements

This means that the equations of the Kalman filter belong to two groups: **time update equations** and **measurement update equations**.

The time update equations forward in time the projections of the current state and error covariance estimates to obtain the a priory estimate for the next step. These equations are also referred to as predictor equations.

Time Update Equations:

$$\hat{x}^{-}(k) = A\hat{x}(k-1) + Bu(k-1) \quad \textbf{(6.33)}$$

$$P^{-}(k) = A^{T}P(k-1)A^{T} + \sum(k-1) \quad \textbf{(6.34)}$$

given initial estimates for $\hat{x}(k-1)$ and $P(k-1)$

The measurement update equations are responsible for the feedback and for incorporation of new measurement into a priory estimate to obtain an improved a posteriory estimate. These equations are also referred to as corrector equations.

Measurement Equations:

$$K(k) = P^{-}(k)C^{T}\left[CP^{-}(k)C^{T} + Q(k)\right]^{-1} \quad \textbf{(6.35)}$$

$$\hat{x}(k) = \hat{x}^{-}(k) + K(k)\left[y(k) - C\hat{x}^{-}(k)\right] \quad \textbf{(6.36)}$$

$$P(k) = \left[I - K(k)C\right]P^{-}(k) \quad \textbf{(6.37)}$$

6.2.5 Algorithm of the Kalman filter

1 Determine the model Equations (6.1), (6.2) and noise characteristics, Equation (6.3).

2 Initialisation

- determine the initial state estimate $\hat{x}(t_0) = \hat{x}_0$
- determine the initial covariance matrix $P_0 = \mathrm{E}\left\{e(t_0)e(t_0)^{T}\right\}$

3 Calculate the Kalman filter gain from Equation (6.28) with $\hat{x}_0^{-} = \hat{x}_0$ and $P_0^{-} = P_0$

4 Update the state estimate according to Equation (6.9) and update the covariance matrix for the estimation error according to the Equation (6.29).

5 Propagate the calculated at the step 4 state estimate to the next step using Equation (6.2) and propagate the calculated at step 4 covariance matrix to the next step using the Equation (6.15)

Parameters and tuning of the filter:

1 The measurement noise covariance $Q(k)$ is usually calculated in front in order to operate by off-line sample measurements.

2 Determination of the process noise covariance $\sum(k)$ is more difficult because an uncertainty is injected into the process by selecting $\sum(k)$.

3 Tuning of $\sum$ and Q to receive better performance of the filter offline.

4 If $\sum$ and Q are constant, both P(k) and K(k) will stabilise quickly and then remain constant. The values of $\sum$ and Q can also be calculated offline.

6.2.6 Steady state Kalman filter

For time invariant systems the error covariance matrix P reaches the steady state values very quickly. In this case a steady state constant value of the gain K can be pre-computed using the steady state covariance matrix and the number of computations is reduced significantly. The steady state version of the recursive Kalman filter is:

$$\hat{x}(k) = Ax(k-1) + Bu(k-1) + AK\left[y(k-1) - C\hat{x}(k-1)\right] \quad \textbf{(6.38)}$$

$$K = PC^T\left[CPC^T + Q\right]^{-1} \quad \textbf{(6.39)}$$

$$P = APA^T - APC^T\left[CPC^T + Q\right]^{-1}CPA^T + \sum \quad \textbf{(6.40)}$$

The requirements to the Kalman filter are: to be stable, to be sensitive to the changes in the measurements, to have fast convergence towards the real state space variables and to have less amount of computations for implementation of the algorithm.

6.3 Kalman filter estimation for the case of a delay in the measurement data

The standard Kalman filter uses fixed set of the plant output measurements that are taken in every sampling interval in order to calculate the state estimations. The algorithm of this Kalman filter is designed to handle only one measurement vector generated at each sampling time ***k*** and is not designed to work with missing data or significant delay between the moment of measurement of the data and the moment of its arrival to the filter (Larsen, 1998).These types of delays in the NCS require bigger flexibility and tolerance towards the data arrival changes, which means new filter architecture and algorithms are required. Some of the required characteristics of the Kalman filter capable to work in the NCS are given in (Larsen, 1998):

- to accept measurements delivered asynchronously from different sensors and to use them in the state estimate.
- to function not depending on the characteristics of a given sensor.
- to incorporate the delayed measurement that brings information for the previous plant state into the current state estimate.
- to be capable of processing algorithm function quickly.

Delays are caused by unavailability of the sensor data at the moment of measurement. These sensor data normally arrive to the Kalman filter with some delays, which can be different and unknown. Most of the papers considering delaying measurement in the Kalman filter take into account the cases where measurements are not fused in the period of the delay. Later papers considered fusing of the faster measurements in the delay period. If not many measurements to be fused in the delay period the optimal state estimate including the delayed measurement can be obtained by recalculating of the filter through the period of the delay. This procedure can be very complex and time consuming. A method for calculation of a compensation term and adding it to the state estimate at the moment of arrival of the delayed measurement is developed in (Alexander, 1991). The method accommodate the arrival at time ***k*** of the delayed data that carry information for the plant state at a time $l < k$.

This method is applied in the thesis.

Equations (6.33)-(6.37) of the normal Kalman filter are considered also for the case of delayed measurements. These equations were derived for the case of the model of the system and the measured output given by Equation (6.1) and (6.2).

In the case that some of the outputs of the system delay with τ^{sc} samples a new output equation has to be introduced.

$$y^*(k+\tau^{sc}) = C^*(k)x(k) + v^*(k+\tau^{sc}), \quad v^*(k+\tau^{sc}) \approx N(0, \sum^*) \qquad \textbf{(6.41)}$$

Graphical representation of the delay is shown in Figure (6.1) (Larsen et al, 1998).

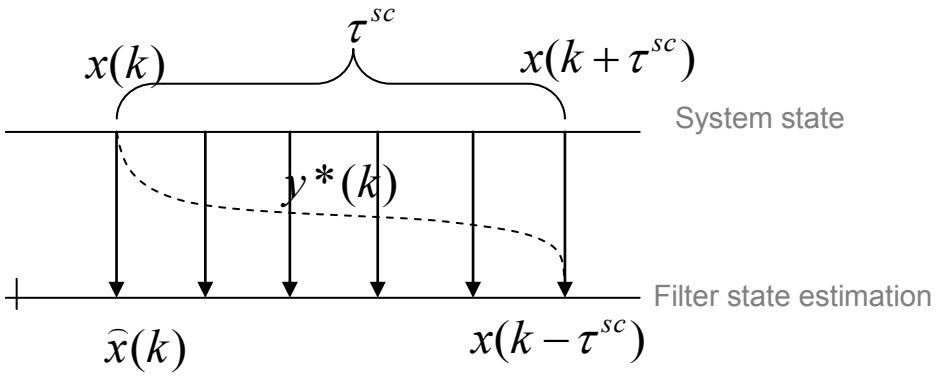


Figure 6.1: System with τ^{sc} samples delay
(Adapted from Larsen et al, 1998)

The delayed measurement can not be fused using the normal Kalman filter equations. There is a need for modification of the structure of the filter (Larsen et al, 1998). The method of (Alexander, 1991) is applied for the considered process in the thesis.

The derivations are based on the normal equations for the Kalman filter - Equations (6.33) - (6.37). The method of Alexander, (1991) is based on computation of an update of

the estimated state at the moment of fusion of the delayed measurement to the input of the Kalman filter. This update represents the difference between the estimate of the moment of fusion of the delayed measurement for the case of not delay and the state estimate for the case of existing delay. Derivations are based on Equation (6.33) and (6.36) considered for every sampling period.

Let a delay of τ^{sc} samples be considered, which may appear at the moment k and which can be fused at the moment $k+\tau^{sc}$.

Then two cases are considered:

- no delay of the measurement done at the moment k.
- delay of the measurement done at the moment k.

6.3.1 First Case - No delay of the measurement done at the moment k

If no delay, then all equations of the Kalman filter are applicable, the state estimates and the corresponding covariance matrices can be calculated. For the apriory estimate of the state at the moment $k+\tau^{sc}$ can be written on the basis of all calculations from the moment k till the moment $k+\tau^{sc}$. For the ease of the derivations it is accepted that $\tau^{sc}=3$ and $x(k+\tau^{sc}-3)=\hat{x}(k)$

$$\hat{x}^{-}(k+\tau^{sc}) = A\hat{x}(k+\tau^{sc}-1)+Bu(k+\tau^{sc}-1)$$

$$= A\left[\hat{x}^{-}(k+\tau^{sc}-1)+K'(k+\tau^{sc}-1)y(k+\tau^{sc}-1)-C\hat{x}^{-}(k+\tau^{sc}-1)\right]+Bu(k-\tau^{sc}-1)$$

$$= A\left\{\left[I-K'(k+\tau^{sc}-1)C\right]\hat{x}^{-}(k+\tau^{sc}-1)+K'(k+\tau^{sc}-1)y(k+\tau^{sc}-1)\right\}+Bu(k+\tau^{sc}-1)$$

$$= A\left\{\left[I-K'(k+\tau^{sc}-1)C\right]\left[A\hat{x}(k+\tau^{sc}-2)+Bu(k+\tau^{sc}-2)\right]+K'(k+\tau^{sc}-1)y(k+\tau^{sc}-1)\right\}+Bu(k+\tau^{sc}-1)$$

$$= A\left\{\begin{bmatrix}\left[I-K'(k+\tau^{sc}-1)C\right]\begin{Bmatrix}A\left[I-K'(k+\tau^{sc}-2)C\right]\hat{x}^{-}(k+\tau^{sc}-2)\\+K'(k+\tau^{sc}-2)y(k+\tau^{sc}-2)\end{Bmatrix}\\+K'(k+\tau^{sc}-1)y(k+\tau^{sc}-1)\end{bmatrix}\right\}+Bu(k+\tau^{sc}-1)$$

$$= A\left\{\begin{bmatrix}[I-K'(k+\tau^{sc}-1)C]\begin{Bmatrix}A\left[\left[I-K'(k+\tau^{sc}-2)C\right]\left[A\hat{x}^{-}(k)+Bu(k)\right]+K'(k+\tau^{sc}-2)\right]\\+B(k+\tau^{sc}-2)\end{Bmatrix}\\+K'(k+\tau^{sc}-1)y(k+\tau^{sc}-1)\end{bmatrix}\right\}+Bu(k+\tau^{sc}-1)$$

$$= A\left\{\begin{bmatrix}[I-K'(k+\tau^{sc}-1)C]\begin{Bmatrix}A\left[\left[I-K'(k+\tau^{sc}-2)C\right]\left[A\left[\hat{x}^{-}(k)+K_2(k)\left[y(k)-C\hat{x}^{-}(k)\right]\right]+Bu(k)\right]+K'(k+\tau^{sc}-2)\right]\\+B(k+\tau^{sc}-2)\end{Bmatrix}\\+K'(k+\tau^{sc}-1)y(k+\tau^{sc}-1)\end{bmatrix}\right\}$$

$$\begin{aligned}&+Bu(k+\tau^{sc}-1)\\&=A\left[I-K'(k+\tau^{sc}-1)C\right]A\left[I-K'(k+\tau^{sc}-2)C\right]A\left[\hat{x}^{-}(k)+K_2(k)\left[y(k)-C\hat{x}^{-}(k)\right]\right]+\\&+A\left[I-K'(k+\tau^{sc}-1)C\right]A\left[I-K'(k+\tau^{sc}-2)C\right]Bu(k)+\\&+A\left[I-K'(k+\tau^{sc}-1)C\right]AK'(k+\tau^{sc}-2)y(k+\tau^{sc}-2)+\\&+A\left[I-K'(k+\tau^{sc}-1)C\right]B(k+\tau^{sc}-2)+AK'(k+\tau^{sc}-1)y(k+\tau^{sc}-1)+Bu(k+\tau^{sc}-1)\end{aligned}\qquad \textbf{(6.42)}$$

The prime on the Kalman gains for $k+\tau^{sc}-1$ and $k+\tau^{sc}-2$ shows that the gains are influenced by the update of the covariance matrix at the moment ***k***.

6.3.2 Second Case - delayed measurement from the moment k

The measurement in this case cannot be incorporated at the moment ***k***. Then the resulting state estimate at the moment $k+\tau^{sc}$ is:

$$\hat{x}^{-}(k+\tau^{sc})=A\left[\begin{bmatrix}[I-K(k+\tau^{sc}-1)C]\left[A\begin{bmatrix}\left[I-K(k+\tau^{sc}-2)C\right]\left[A\hat{x}^{-}(k)+Bu(k)\right]+\\K(k+\tau^{sc}-2)y(k+\tau^{sc}-2)\end{bmatrix}+Bu(k)\right]+\\K(k+\tau^{sc}-1)\end{bmatrix}\right]+$$

$$\begin{aligned}&+Bu(k+\tau^{sc}-1)\\&=A\left[I-K(k+\tau^{sc}-1)C\right]A\left[I-K(k+\tau^{sc}-2)C\right]A\hat{x}^{-}(k)+\\&+A\left[I-K(k+\tau^{sc}-1)C\right]A\left[I-K(k+\tau^{sc}-2)C\right]Bu(k)+\\&+A\left[I-K(k+\tau^{sc}-1)C\right]AK(k+\tau^{sc}-2)y(k+\tau^{sc}-2)+\\&+A\left[I-K(k+\tau^{sc}-1)C\right]B(k+\tau^{sc}-2)+AK(k+\tau^{sc}-1)y(k+\tau^{sc}-1)+Bu(k+\tau^{sc}-1)\end{aligned}\qquad \textbf{(6.43)}$$

In this case the Kalman gains for the moments $k+\tau^{sc}-1$ and $k+\tau^{sc}-2$ are different from these for the case without delay based on the fact that they are not based on any update of the covariance matrix P at the time k. The difference between the two cases shows the influence of the measurement update at the time k on the value of the state estimation for the case of delayed measurement.

This difference is:

$$
\begin{aligned}
&\delta\hat{x}(k+\tau^{sc}) = \hat{x}'^{-}(k+\tau^{sc}) - \hat{x}^{-}(k+\tau^{sc}) = \\
&= A\left[I - K'(k+\tau^{sc}-1)C\right]A\left[I - K'(k+\tau^{sc}-2)C\right]K(k)\left[y(k) - C\hat{x}^{-}(k)\right] + \\
&+ A\left[\left[I - K'(k+\tau^{sc}-1)C\right]A\left[I - K'(k+\tau^{sc}-2)C\right]\right] - \\
&- \left[I - K'(k+\tau^{sc}-1)C\right]A\left[I - K'(k+\tau^{sc}-2)C\right]A\hat{x}^{-}(k) + Bu(k) + \\
&+ A\begin{bmatrix}\left[I - K'(k+\tau^{sc}-1)C\right]AK'(k+\tau^{sc}-2) - \\ -\left[I - K(k+\tau^{sc}-1)C\right]AK'(k+\tau^{sc}-2)\end{bmatrix}y(k+\tau^{sc}-2) + \\
&+ A\left[\left[I - K'(k+\tau^{sc}-1)C\right] - \left[I - K(k+\tau^{sc}-1)\right]Bu(k+\tau^{sc}-2)\right] + \\
&+ A\left[K'(k+\tau^{sc}-1) - K(k+\tau^{sc}-1)\right]y(+\tau^{sc}-1)
\end{aligned}
\qquad \textbf{(6.44)}
$$

The first term of the expression gives the direct effect of the measurement update at the moment k. It is affected by the following measurements till the time $k+\tau^{sc}$. The other terms have indirect influence based on the update of the covariance matrix P at the time k and also based on the differences between the Kalman gains K' and K.

The calculations of the update can be done in the following way:

- calculation of $\delta\hat{x}(k+\tau^{sc})$ at time $k+\tau^{sc}$
- update $\hat{x}^{-}(k+\tau^{sc})$ using the update as follows:

$$
\hat{x}(k+\tau^{sc}) = \hat{x}^{-}(k+\tau^{sc}) + \delta\hat{x}(k+\tau^{sc}) = \delta\hat{x}^{-}(k+\tau^{sc}) \qquad \textbf{(6.45)}
$$

If this is done the obtained estimate $\hat{x}(k+\tau^{sc})$ exactly will incorporate the measurement y(k), in the same way if it had been incorporated at the moment k. The error covariance matrix P has to be updated also

Calculation of the update is very difficult and time consuming. Simplification is proposed in Alexander, (1991) based on the fact that the output matrix C and the measurement noise covariance matrix $\sum$ can be calculated in front, based on the error covariance matrix that is updated at time k, nevertheless that the update of $\hat{x}$ is delayed till the moment $k+\tau^{sc}$.

The update in this case is:

$$
\begin{aligned}
\hat{x}'^{-}(k+\tau^{sc}) &= A\left[I - K'(k+\tau^{sc}-1)C\right]A\left[I - K'(k+\tau^{sc}-2)C\right]K(k)\left[y(k) - C\hat{x}^{-}(k)\right] \\
&= A^{3=\tau^{sc}}\left[\left[I - K'(k+\tau^{sc}-1)C\right]\left[I - K'(k+\tau^{sc}-2)C\right]\right]K(k)\left[y(k) - C\hat{x}^{-}(k)\right] \\
&= A^{3}\prod_{i=1}^{\tau^{sc}-1}\left\{\left[I - K'(k+\tau^{sc}-i)C\right]\right\}K(k)\left[y(k) - C\hat{x}^{-}(k)\right] \qquad \textbf{(6.46)} \\
&= A^{\tau^{sc}}\prod_{i=1}^{\tau^{sc}-1}\left\{\left[I - K'(k+\tau^{sc}-i)C\right]\right\}K(k)\left[y(k) - C\hat{x}^{-}(k)\right] \\
&= M(k,\tau^{sc})K(k)\left[y(k) - C\hat{x}^{-}(k)\right]
\end{aligned}
$$

where $M(k,\tau^{sc}) = A^{\tau^{sc}}\prod_{i=1}^{\tau^{sc}-1}\left\{\left[I - K'(k+\tau^{sc}-i)C\right]\right\}$

The calculation procedure:

1. Initial conditions for calculation of the Kalman filter are determined and there is a delay of τ^{sc} at every measurement.
2. Calculation of $\hat{x}^{-}(k) = A\hat{x}(k-1) + Bu(k-1)$, for k=1, $\hat{x}(0)$ is given
3. Calculation of $P^{-}(k) = A^{T}P(k-1)A^{T} + \sum(k-1)$ for k=1, where P(0) is given.
4. Calculation of $K(k) = P^{-}(k)C^{T}\left[CP^{-}(k)C^{T} + Q(k)\right]^{-1}$ for k=1.
5. Check for available measurement at k=1.
6. If no measurement $\hat{x}(k) = \hat{x}^{-}(k)$; $P(k) = \left[I - K(k)C\right]P^{-}(k)$
7. If there is measurement for moment k=1

   $\hat{x}(k) = \hat{x}^{-}(k) + K(k)\left[y(k) - C\hat{x}^{-}(k)\right]$

   $P(k) = \left[I - K(k)C\right]P^{-}(k)$
8. Calculation of $M(k, \tau^{sc})$ for k=1.
9. Calculation for k=2, and so on.

6.4 Conclusion

The derivation of the normal discrete Kalman filter is described where different considerations are discussed and the derivations are presented. The derivation of the Kalman filter estimation for the case of a delay in the measurement data is presented considering two cases. The first case is considered where there is no delay measurement at moment k. In this case the normal Kalman filter equation are applicable. The second case is considered where there is delay measurement at moment k. In this case the normal Kalman filter equation as not applicable as a result a predictive Kalman filter is derived based on (Alexander, 1991) and (Larsen et al, 1998) methods.

The next chapter described the development of the Robust State Predictive Controller (RSPC). The RSPC is developed considering the network induced time delays only. The considered delay during the development of the RSPC is between the controller and actuator and the sum between the controller and the actuator and between the sensor and the controller

CHAPTER SEVEN
DESIGN OF A ROBUST STATE PREDICTIVE CONTROLLER BASED ON A DISCRETE KALMAN FILTER

7.1 Introduction

This chapter describes the design of a robust discrete state predictive controller for a Networked Control System. The design problem has three major sub-problems that have to be considered before the design of the overall controller for the NCS is considered. The sub-problems are:

- Natural dynamics of the physical process as described in Chapter 4.
- Determination of the uncertainties as described in Chapter 2 focusing on network induced time delays (with the exclusion of delays caused by sampling) and packet loss due to network congestion.
- Design of the Discrete Kalman Filter based on plant output measurements and the known control input.

In order to design a controller considering the sub-problems highlighted above a design philosophy adapted from Athans (1971) has been adopted. The design of the robust controller is done for different cases:

- Case 1: Design of an optimal controller for the plant without deterministic disturbance. This case is described is section 7.2.
- Case 2: Design of an optimal controller for the plant with deterministic wind disturbance. This case is described is section 7.3.
- Case 3: Design of an optimal robust predictive controller for the augmented model of the plant by incorporation of the network delays in the plant model control vector. This case is described is section 7.4.

For all these 3 cases it is supposed that stochastic noises in the model of the plat and the model of measurement exist. These noises are considered during the discrete Kalman filter design.

The general approach is to consider a plant with or without deterministic disturbances and with stochastic noises and to control it by using a deterministic controller based on state estimates obtained by a discrete Kalman filter designed for the known characteristics of the state and output noises. Then the problems for design of the controller and the problem for design of the Kalman filter can be considered separately.

The network induced delays are used in order to design robust state predictive controller. The prediction of the state can be for one of the delays between the controller and the actuator, between the sensor and the controller, or for the sum of these two delays.

In this case the design of the Kalman filter is for a system without delays. The equations derived in Chapter 6 are used.

7.2 Case 1: Design of an optimal controller for the model of the plant without the deterministic wind disturbance

7.2.1 Model without deterministic disturbance

The assumptions for the development of the model are that; there are no plant deterministic disturbances; no actuator errors; all state variables and output variables can be measured with some noises; all parameter values are known. Based on these assumptions and the model, a deterministic controller is designed by solving the Linear Quadric problem. The stochastic noises are taken under account during the design of the discrete Kalman filter.

The equations of the process and the filter for the case of missing deterministic constant wind disturbance and with state and output noises are:.

$$x(k+1) = Ax(k) + Bu(k) + w(k) \quad \textbf{(7.1)}$$

$$y(k) = Cx(k) + v(k) \quad \textbf{(7.2)}$$

$$\hat{x}(k) = A\hat{x}(k-1) + Bu(k) + K(k)\left[y(k) - C\hat{x}(k-1)\right] \quad \textbf{(7.3)}$$

where $x(k) \in R^n$ is the state vector of the system, $\hat{x}(k) \in R^n$ is the estimated state $y(k) \in R^l$ is the measurement, $u(k) \in R^m$ is the control vector, $K(k)$ is the Kalman Gain $k = k_0, k_1, ...k_{f-1}$ A, B and C are known matrices with proper dimensions

The controller is designed according to the desired output and has the expression (Naidu, 2003):

$$u(k) = -L(k)\hat{x}(k) + L_g(k)g(k+1) \quad \textbf{(7.4)}$$

The derivation for Equations (7.3) and Equation (7.4) are described in sections 7.2.2 and 7.2.3 respectively. The diagram of the closed loop system is shown in Figure (7.1).

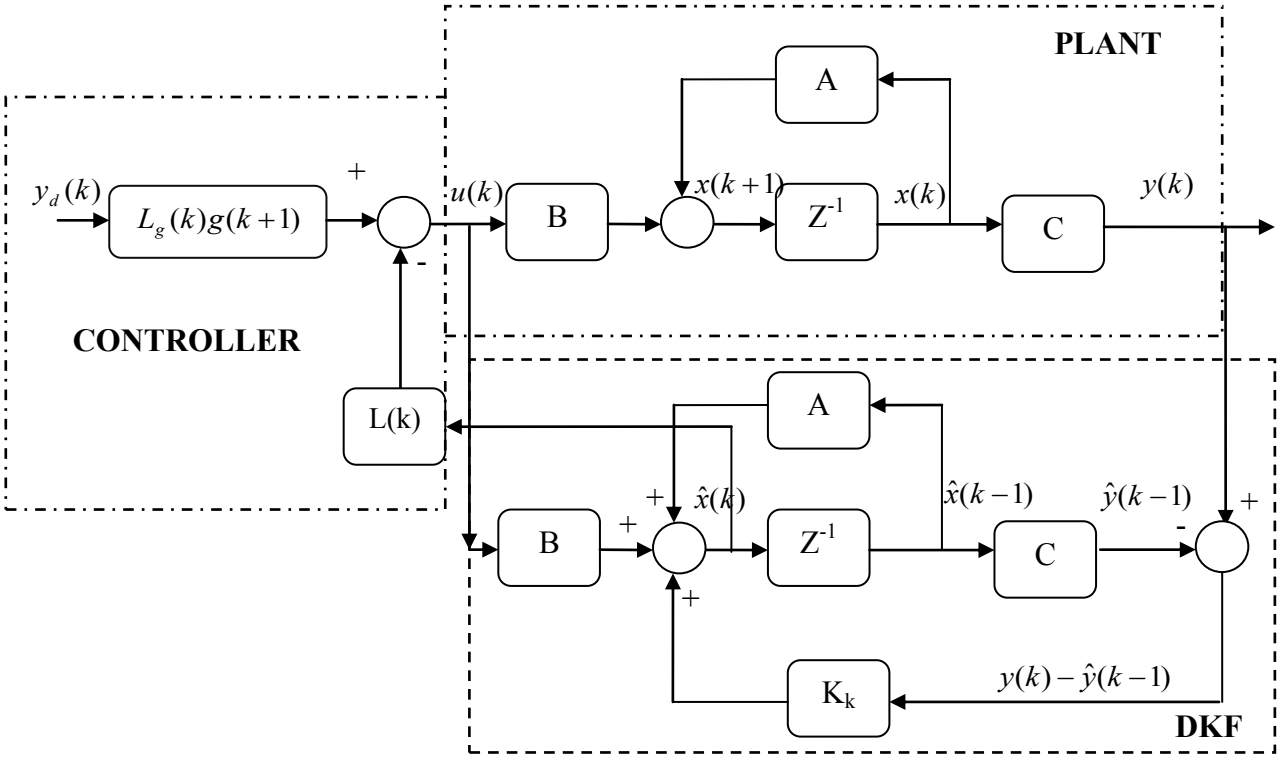


Figure 7.1: Closed loop structure without constant wind disturbance

7.2.2 Solution of the deterministic Linear Quadratic problem

The problem for design of an optimal tracking reference control is to find the optimal control which causes the discrete linear time-invariant (LTI) system:

$$x(k+1) = Ax(k) + Bu(k),\ \ x(0) = x_0 \tag{7.5}$$

$$y(k) = Cx(k) \tag{7.6}$$

to give the trajectory that minimizes a performance index:

$$\begin{aligned} S = {} & \frac{1}{2}\left[Cx(k_f) - y_d(k_f)\right]^T F\left[Cx(k_f) - y_d(k_f)\right] \\ & + \frac{1}{2}\sum_{k=k_0}^{k_f-1}\left\{\left[Cx(k_f) - y_d(k_f)\right]^T Q\left[Cx(k_f) - y_d(k_f)\right] + u^T(k)Ru(k)\right\} \end{aligned} \tag{7.7}$$

where, $x(k) \in R^n, y_d(k) \in R^l, u(k) \in R^m,$ and $y(k) \in R^l$ are state, desired output, control and output vectors, respectively. Where $R \in R^{nxm}$ is a symmetric positive definite matrix, $Q \in R^{lxl}$, are symmetric positive semi-definite matrices, and $F \in R^{lxl}$. The initial conditions are given as: $x(k = k_0) = x(k_0);\ x(k_f)$ is free, and k_f is fixed. The aim is to make the error $e(k) = y(k) - y_d(k)$ as small as possible, with minimum control effort. The approach used to find the solution that minimises the function of Equation (7.7) is adapted from Naidu (2003) and it has the following steps:

- Hamiltonian formulation
- State and Co-state system expression
- Open loop optimal control derivation
- Riccati and vector equations solution
- Closed loop optimal control derivation

7.2.2.1 Hamiltonian formulation

In this section the Hamiltonian for the problem in Equation (7.5)-(7.7) is formulated as follows:

$$H(x(k),u(k),\lambda(k+1)=\frac{1}{2}\sum_{k=k_0}^{k_f-1}\left\{\left[Cx(k)-y_d(k)\right]^T Q\left[Cx(k)-y_d(k)\right]+u^T(k)Ru(k)\right\} + \lambda^T(k+1)\left[Ax(k)+Bu(k)\right] \quad \textbf{(7.8)}$$

where $\lambda \in R^n$ are co-state variables.

The optimal solution of the problem (7.5) - (7.7) is based on the necessary conditions for optimality of the Hamiltonian as follows:

$$\lambda^*(k)=\frac{\partial H(x^*(k),u^*(k),\lambda^*(k+1))}{\partial x^*(k)} \quad \textbf{(7.9)}$$

$$0=\frac{\partial H(x^*(k),u^*(k),\lambda^*(k+1))}{\partial u^*(k)} \quad \textbf{(7.10)}$$

$$x^*(k)=\frac{\partial H(x^*(k-1),u^*(k-1),\lambda^*(k))}{\partial \lambda^*(k)} \quad \textbf{(7.11)}$$

where (*) means the optimal value of the variables.

7.2.2.2 State and Co-state system

Using the Hamiltonian equations for state, co-state and control respectively yields the state equation:

$$\frac{\partial H}{\partial \lambda^*(k+1)}=x^*(k+1)\longrightarrow x^*(k+1)=Ax^*(k)+Bu^*(k) \quad \textbf{(7.12)}$$

the co-state equation:

$$\frac{\partial H}{\partial x^*(k+1)}=\lambda^*(k)\longrightarrow \lambda^*(k)=A^T\lambda^*(k+1)+Vx^*(k)-Gy_d(k), \quad \textbf{(7.13)}$$

and the control equation:

$$\frac{\partial H}{\partial u^*(k)}=0\longrightarrow 0=B^T\lambda^*(k+1)+Ru^*(k), \quad \textbf{(7.14)}$$

The final condition becomes:

$$\lambda(k_f) = C^T F C x(k_f) - C^T F y_d(k_f) \quad \textbf{(7.15)}$$

7.2.2.3 Open-Loop Optimal Control

The relation in Equation (7.14) yields the open loop optimal control as:

$$u^*(k) = -R^{-1}B^T\lambda^*(k+1) \quad \textbf{(7.16)}$$

and using state (Equation 7.12) and co-state (Equation 7.13) where $u^*(k)$ is substituted by Equation (7.16) the Hamiltonian system can be written as:

$$\begin{bmatrix} x^*(k+1) \\ \lambda^*(k) \end{bmatrix} = \begin{bmatrix} A & -E \\ V & A^T \end{bmatrix} \begin{bmatrix} x^*(k) \\ \lambda^*(k+1) \end{bmatrix} + \begin{bmatrix} 0 \\ -G \end{bmatrix} y_d(k) \quad \textbf{(7.17)}$$

where $E = BR^{-1}B^T$, $V = C^TQC$, $G = C^TQ$

7.2.2.4 Riccati equations

The closed loop optimal control is solved by expressing co-state function in the optimal control Equation (7.16) in terms of the state variables. Using the final condition in Equation (7.15), the co-state for every moment of the optimisation period can be presented as:

$$\lambda^*(k) = P(k)x^*(k) - g(k) \quad \textbf{(7.18)}$$

Where the symmetrical matrix $P \in R^{nxn}$ and the vector $g \in R^n$ are derived in the following section. Vector $g(k)$ depends on the value of the desired trajectory $y_d(k)$. In order to solve for $P(k)$, the co-state $\lambda(k)$ from canonical system from Equation (7.17) is eliminated using transformation from Equation (7.18).

$$x^*(k+1) = Ax^*(k) - EP(k+1)x^*(k+1) + Eg(k+1) \quad \textbf{(7.19)}$$

which is solved for $x^*(k+1)$ to yield:

$$x^*(k+1) = \left[I + EP(k+1)\right]^{-1}\left[Ax^*(k) + Eg(k+1)\right] \quad \textbf{(7.20)}$$

Now using Equation (7.20) and Equation (7.18) in the Hamiltonian system in Equation (7.17), yields:

$$\begin{aligned} &\Big[-P(k) + A^TP(k+1)\left[I + EP(k+1)\right]^{-1}A + V\Big]x^*(k) + \\ &\quad \Big[g(k) + A^TP(k+1)\left[I + EP(k+1)\right]^{-1}Eg(k+1) - A^Tg(k+1) - Gy_d(k)\Big] = 0 \end{aligned} \quad \textbf{(7.21)}$$

Equation (7.21) has to hold for all values of the state $x^*(k)$ which in turn leads to the fact that coefficients in front of $x^*(k)$ and the rest on the term in Equation (7.21) must individually disappear, yielding:

$$P(k) = A^TP(k+1)\left[I + EP(k+1)\right]^{-1}A + V \text{ or} \quad \textbf{(7.22)}$$

$$P(k) = A^T \left[P^{-1}(k+1) + E\right]^{-1} A + V \quad \textbf{(7.23)}$$

and

$$g(k) = A^T \left\{I - \left[P^{-1}(k+1) + E\right]^{-1} E\right\} g(k+1) + G y_d(k) \text{ or} \quad \textbf{(7.24)}$$

$$g(k) = \left\{A^T - A^T P(k+1)\left[I + EP(k+1)\right]^{-1} E\right\} g(k+1) + G y_d(k) \quad \textbf{(7.25)}$$

Equations (7.22) and (7.24), or Equation (7.23) and (7.29) can be solved backward from the final point of the optimization period to the first one.

To obtain the boundary conditions for Equation (7.22) and Equation (7.24), a comparison is made between Equation (7.15) and Equation (7.18):

$$P(k_f) = C^T F C \quad \textbf{(7.26)}$$

$$g(k_f) = C^T F y_d(k_f) \quad \textbf{(7.27)}$$

Equation (7.22) is called Difference Riccati Equation (DRE) to be solved backwards using final condition in Equation (7.26) and the liner vector difference Equation (7.25) is solved backwards using final condition in Equation (7.27).

7.2.2.5 Closed loop optimal controller

In order to solve for the optimal controller $u^*(k)$ the transformation in Equation (7.18) and the control relation in Equation (7.16) are used to get the closed loop optimal control as:

$$u^*(k) = -R^{-1} B^T \left[P(k+1)x(k+1) - g(k+1)\right] \quad \textbf{(7.28)}$$

and substituting for the state from Equation (7.12) in Equation (7.28),

$$u^*(k) = -R^{-1} B^T P(k+1)\left[Ax^*(k) + Bu^*(k)\right] + R^{-1} B^T g(k+1) \quad \textbf{(7.29)}$$

Simplifying Equation (7.29) by multiplying by R and solving for optimal control yields,

$$u^*(k) = -L(k)x^*(k) + L_g(k)g(k+1) \quad \textbf{(7.30)}$$

The first component of the closed loop control stabilizes the system in an optimal way. The second component compensates for the error between the desired output and its real value.

Where the feedback gain $L(k)$ and feed forward gain $L_g(k)$ are given by:

$$L(k) = \left[R + B^T P(k+1)B\right]^{-1} B^T P(k+1)A \quad \textbf{(7.31)}$$

$$L_g(k) = \left[R + B^T P(k+1)B\right]^{-1} B^T \quad \textbf{(7.32)}$$

The optimal state trajectory is then given from Equation (7.12) and Equation (7.30) as:

$$x^*(k+1) = \left[A - BL(k)\right]x(k) + BL_g(k)g(k+1) \quad \textbf{(7.33)}$$

The procedure of designing a linear quadratic feedback controller can be summarized as to:

- Solve the reduced-matrix Riccati equation for the matrix P. If a positive-definite symmetrical matrix P exists, the system is stable.
- Substitute this matrix P into the Equations (7.31) and (7.32), the matrices L and L_g are the optimal matrices of the controller.

The implementation of the discrete-time optimal tracker based on Kalman filter is shown in Figure (7.1).

The main disadvantage of the designed deterministic controller is the exact measurements of state variables, which are impossible to measure in practical situations. That is why the discrete Kalman filter is used to take care of the state and output noises and to calculate an optimal state estimation. The filter Equations (6.35)-(6.40) are considered.

The filter generates on the basis of the past sensor measurements a set of time functions which are as close as possible to the true value of the physical plant state and output variables at any instant of time. The Kalman filter is considered as the state estimator.

7.3 Case 2: Design of an optimal controller for the model of the plant with presence of the deterministic wind disturbance

7.3.1 Model with disturbances.

In this case the model of the plant is considered with presence of a deterministic disturbance and state and output noises as shown in Equation (7.34) and (7.35). The two noises are state variable errors and sensor errors.

$$x(k+1) = Ax(k) + Bu(k) + WT_w(k) + w(k) \quad \textbf{(7.34)}$$

$$y(k) = Cx(k) + v(k) \quad \textbf{(7.35)}$$

where $x(k) \in R^n$ is the state vector of the system, $y(k) \in R^l$ is the observation, $u(k) \in R^m$ is the control vector, $T_w(k)$ is the Torque of the wind disturbance, $k = k_0, k_1, ...k_{f-1}$ A, B, C and W are known matrices with proper dimensions and assumed to be non-random for all values of k. $w(k)$ and $v(k)$ are discrete time white noise process. The wind disturbance $T_w(k)$ is deterministic, constant and measurable.

The addition of the white noise emphasises the unpredictability of the real world because the state variable $x(k)$ can change in an unpredictable way as a result of the added white noise.

The discrete Kalman filter equation is:

$$\hat{x}(k) = A\hat{x}(k-1) + Bu(k) + WT_w(k) + K_k\left[y(k) - C\hat{x}(k-1)\right] \quad \textbf{(7.36)}$$

where $\hat{x}(k)$ is the state estimate at the moment ***k***.

The closed loop structure for the case of deterministic constant wind disturbance is shown in Figure (7.2).

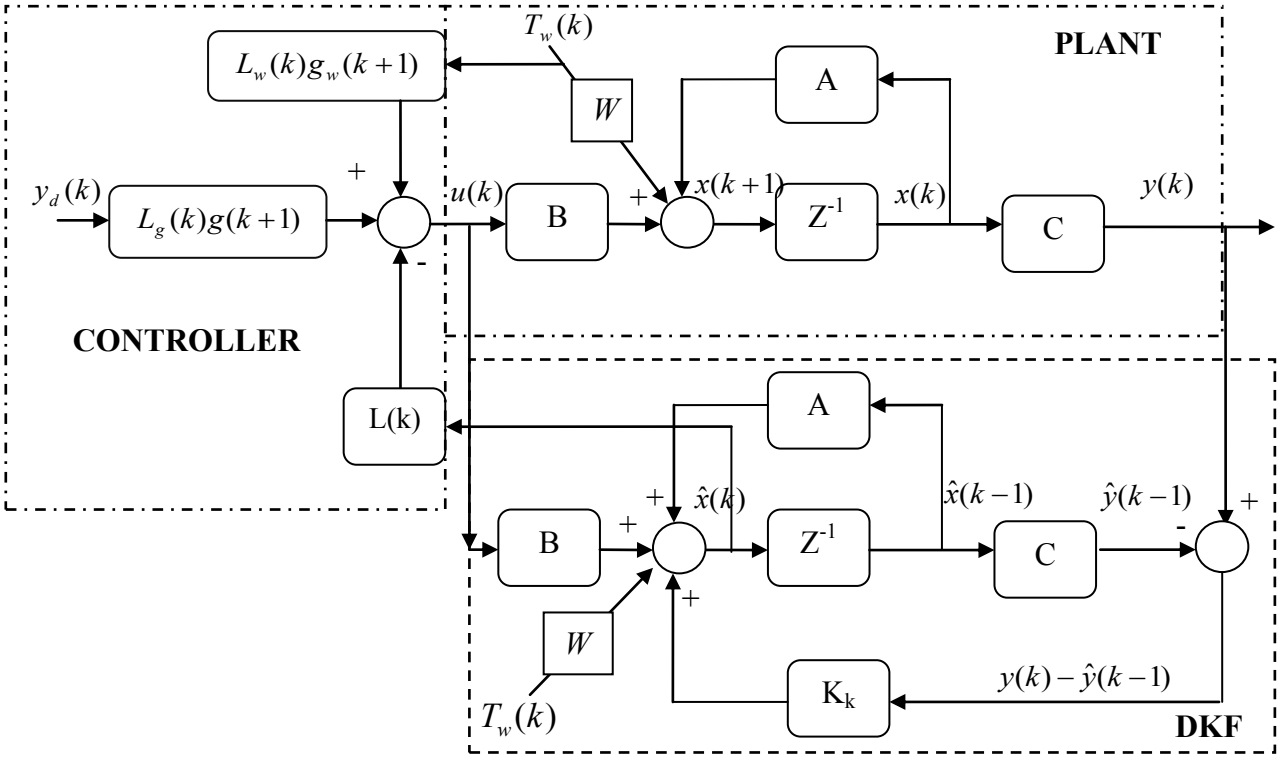


Figure 7.2: **Closed loop control structure with constant wind disturbance**

7.3.2 Design of an optimal controller with deterministic disturbance compensator

The controller is designed according to the desired output of the plant in order to remove the action of the constant disturbance and to lead the plant output to its desired value. Normally in this case an integral controller is used in order to reduce the closed loop steady state error to zero. The controller is designed following the procedure from point 7.2. This controller has an additional component for compensation of the disturbance and is given by:

$$u(k) = -L(k)\hat{x}(k) + L_g(k)g(k+1) + L_w(k)g_w(k+1) \quad \textbf{(7.37)}$$

where L_w is a matrix for disturbance compensation and g_w is a vector solution of the Riccatti type difference equation for the disturbance.

7.4 Case 3: Design of a robust controller for the networked control systems with delays in the communication channel

7.4.1 Incorporation of the communication delays in the plant model

A network control system characterised with time delays between the controller and the actuator (τ^{ca}) and between the sensor and the controller (τ^{sc}) is considered. As discussed in Chapter 2 one of the ways to represent mathematically the model of the plant incorporating the communication delays is to augment the plant with the model of the delays.

The augmentation can be considered in the following ways:

- the delay between the controller and the actuator is considered as a delay in the control vector of the plant model and the delay between the sensor and the controller is considered as delay in the output of the plant model.

  Then the plant model will be represented as follows:

  $$x(k+1) = Ax(k) + Bu(k-\tau) + WT_w(k) + w(k),\ x(0) = x_0 \quad \textbf{(7.38)}$$

  $$y(k) = Cx(k) + v(k) \quad \textbf{(7.39)}$$

  $$u(k) = \varphi_u(k),\ -\tau \le k \le 0 \quad \textbf{(7.40)}$$

  where $\tau = \tau^{ca}$

  $\varphi_u(k)$ is the initial function for the time delay periods. The delay between the sensor and the controller is considered incorporated in the Kalman filter equations.

- both time delays are considered as a delay equal to the sum of them in the control vector of the plant model. In this case the plant model is represented in the following way:

  $$x(k+1) = Ax(k) + Bu(k-\tau) + WT_w(k) + w(k), x(0) = x_0 \quad \textbf{(7.41)}$$

  $$y(k) = Cx(k) + v(k),\ u(k) = \varphi_u(k)\ -\tau \le k \le 0 \quad \textbf{(7.42)}$$

  $$u(k) = \varphi_u(k),\ -\tau \le k \le 0 \quad \textbf{(7.43)}$$

  where $\tau = \tau^{ca} + \tau^{sc}$

The second model of the plant with incorporation of the communication channel delays is simpler from point of view of design of a controller and a Kalman filter. In this case the Kalman filter is considered to be independent on the measurement delay and its normal equations can be used. In the first case the structure of the equations of the Kalman filter is influenced by the delay in the measurement.

In both cases the design of the controller is for the same type of plant model - state space model with delay in the control vector.

The Kalman filter equations are derived in Chapter 6. Derivations of the controller equations is done in this chapter.

7.4.2 Design of an optimal robust predictive controller

The communication delays are normally with statistical nature, they have different values and can appear at different moments. For the purpose of a controller design it is assumed that the values of the delays are known at every moment k in the following ways:

- Every control loop in the network control sytsem is characterised by a given delay, for example in a Square Kilometer antenna array the dishes are situated at different distances from the centralised controller. The delays in this case can be measured in front of the controller design. These values can be used for the design of the controller.
- the values of the delays can be measured every moment using special network tools and their values can be used in the equations of the controller in real-time.
- critical values of the communication delays leading the system to instability can be determined through simulations and experiments. Then the controller is designed for the critical delays and can successfully compensate for delays less than the critical ones.

The purpose is to design an optimal controller to be robust towards the appearance of the communication delays and to be with a predictive control action in order to compensate the effect of the delays over the transition behaviour of the plant. The use of such a controller will guarantee some level of system performance irrespective of the changes of the plant and communication network dynamics.

The problem for design of the optimal, robust, predictive controller can be formulated in the following way:

Determine the closed loop control $u(k), k = \overline{0, k_f - 1}$ such that the plant output follows the desired output trajectory minimising the performance index:

$$S = \frac{1}{2}\left[Cx(k_f) - y_d(k_f)\right]^T F\left[Cx(k_f) - y_d(k_f)\right] + \\ + \frac{1}{2}\sum_{k=0}^{k_f-1}\left\{\left[Cx(k) - y_d(k)\right]^T Q\left[Cx(k) - y_d(k)\right] + u^T(k)Ru(k)\right\} \quad \textbf{(7.44)}$$

for the plant model

$$x(k+1) = Ax(k) + Bu(k-\tau) + WT_w(k),\ x(0) = x_0 \quad \textbf{(7.45)}$$

$$y(k) = Cx(k) \quad \textbf{(7.46)}$$

$$u(k) = \varphi_u(k),\ \ \tau \le k \le 0 \quad \textbf{(7.47)}$$

The state and output measurement noises are not considered here as they are used in the discrete Kalman filter equations.

7.4.2.1 Shifting the optimization period

Time delay in the control vector creates difficulties in the algorithm for design of the controller. This problem can be solved by introduction of a new control vector, determined at every moment k.

$$q(k) = u(k-\tau) \quad \textbf{(7.48)}$$

Then the model of the plant can be written in the following way

$$x(k+1) = Ax(k) + Bq(k) + WT_w(k),\ x(0) = x_0, k = \overline{0, k_f - 1} \quad \textbf{(7.49)}$$

In can be seen that for $0 \le k \le \tau - 1,\ q(k) = \varphi_u(k)$ and for $k = \tau, q(k) = u(0)$. If $\varphi_u(k) = 0$, then the plant will experience control action only at the moment $k = \tau$ and further. In the considered case of communication delays the initial control functions is not known and is accepted to be zero. In order to solve the problem for design of the control action (controller) $q(k)$ it is necessary to express the criterion in terms of $q(k)$. For this purpose the control $u(k)$ has to be expressed by $q(k)$.Using Equation (7.48) it can be written, that

$$u(k) = q(k+\tau) \quad \textbf{(7.50)}$$

Substituting Equation (7.50) into the criterion Equation (7.44) it can be written:

$$\begin{aligned} S &= \frac{1}{2}\left[Cx(k_f) - y_d(k_f)\right]^T F\left[Cx(k_f) - y_d(k_f)\right] + \\ &+ \frac{1}{2}\sum_{k=0}^{k_f-1}\left\{\left[Cx(k) - y_d(k)\right]^T Q\left[Cx(k) - y_d(k)\right] + q^T(k+\tau)Rq(k+\tau)\right\} = \\ &= \frac{1}{2}\left[Cx(k_f) - y_d(k_f)\right]^T F\left[Cx(k_f) - y_d(k_f)\right] + \\ &+ \frac{1}{2}\sum_{k=0}^{\tau-1}\left[Cx(k) - y_d(k)\right]^T Q\left[Cx(k) - y_d(k)\right] + \frac{1}{2}\sum_{k=\tau}^{k_f-1}\left\{\left[Cx(k) - y_d(k)\right]^T Q\left[Cx(k) - y_d(k)\right] + q^T(k)Rq(k)\right\} \end{aligned} \quad \textbf{(7.51)}$$

The criterion for the time period of $\left[0 \div (\tau - 1)\right]$does not depend on the control action and it can be omitted during the process of design. Then the problem state and control can be considered in a new coordinating system with initial point $k = \tau$. This means the

design problem can be considered for the period $\lfloor \tau \div (k_f - 1) \rfloor$ in which the plant model is free from the delay. When the solution of the problem is obtained it can be shifted back to the time $k = 0$.

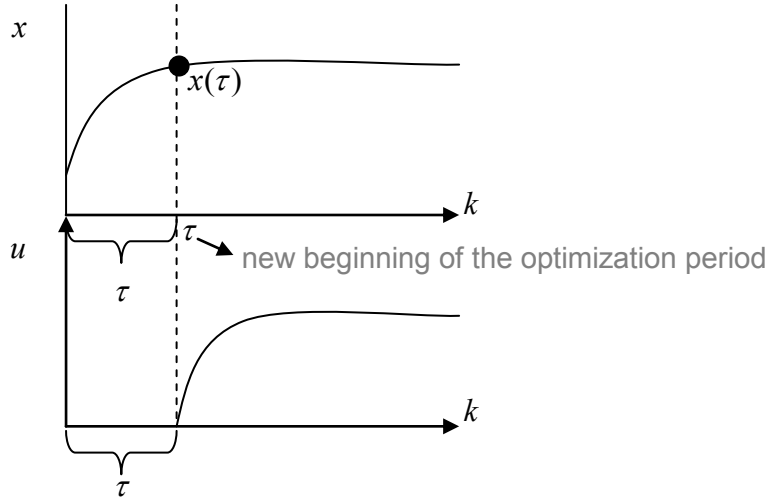


Figure 7.3: Shifting of the optimization period

The initial state in the new optimisation period is $x(\tau)$. It can be calculated using the plant mathematical model on the basis of x(0) and $\varphi_u(k)$, $0 \le k \le \tau - 1$.

7.4.2.2 Solution of the design problem

The mathematical formulation of the problem for design of the controller is:

Find the closed loop optimal trajectory $q(k), \overline{\tau \div k_f - 1}$ such that the criterion

$$S = \frac{1}{2}\left[Cx(k_f) - y_d(k_f)\right]^T F\left[Cx(k_f) - y_d(k_f)\right] + \\ + \frac{1}{2}\sum_{k=\tau}^{k_f-1}\left\{\left[Cx(k) - y_d(k)\right]^T Q\left[Cx(k) - y_d(k)\right] + q^T(k)Rq(k)\right\} \quad \textbf{(7.52)}$$

is minimized and satisfies the plant model equations

$$x(k+1) = Ax(k) + Bq(k) + WT_w(k),\ x(k=\tau) = x(\tau) \quad \textbf{(7.53)}$$

$$y(k) = Cx(k) \quad \textbf{(7.54)}$$

The problem (7.53)-(7.54) is equivalent of the problem considered in case 2 - point 7.3. Its solution is given by the Equation (7.37) and is

$$q(k) = -L(k)\hat{x}(k) + L_g(k)g(k+1) + L_w(k)g_w(k+1),\ k = \overline{\tau, k_f - 1} \quad \textbf{(7.55)}$$

where $\hat{x}(k)$ is the discrete Kalman filter estimate.

In the case of constant model matrices and long optimisation period it can be accepted that L, L_g and L_w are constants.

In order to obtain the solution for the original optimisation period the time is shifted from the moment $k=\tau$ to the moment $k=0$. Then Equation (7.55) becomes

$$u(k)=q(k+\tau)= \\ =-L(k)\hat{x}(k+\tau)+L_g g(k+1+\tau)+L_w g_w(k+1+\tau),\ k=\overline{0,k_f-1} \quad \textbf{(7.56)}$$

where $L=R^{-1}B^T A^{-T}[P-Q],$ and the expression for L_g and L_w is can be seen in (Naido, 20003).

During solution of the problem for design of the controller the trajectories of the vectors g(k) and $g_w(k)$ are obtained and are known. This means the values $g(k+1+\tau)$ and $g_w(k+1+\tau)$ are known and can be used. The value of the estimate $\hat{x}(k+\tau)$ is not known because the estimates are calculated for the current moment k. At the same time the values of the previous estimates are known. Then the expression of the future values of the state estimate can be obtained by the past values of the control and state estimates and the control action can be done on the basis of the transition matrix of the system $\psi(k)$. For the period of the moment k to the moment $k+\tau$, the state estimate can be written:

$$\begin{aligned}\hat{x}(k+\tau)&=\psi(\tau)\hat{x}(k)+\sum_{\varsigma=k}^{k+\tau-1}\left[\psi(k+\tau-\varsigma)\left[Bq(\varsigma)+WT_w(\varsigma)\right]\right] \\ &=\psi(\tau)\hat{x}(k)+\sum_{\varsigma=k-\tau}^{k-1}\left[\psi(k-\varsigma)\left[Bq(\varsigma+\tau)+WT_w(\varsigma+\tau)\right]\right] \quad \textbf{(7.57)} \\ &=\psi(\tau)\hat{x}(k)+\sum_{\varsigma=k-\tau}^{k-1}\left[\psi(k-\varsigma)\left[Bu(\varsigma)+WT_w(\varsigma+\tau)\right]\right]\end{aligned}$$

where $q(\varsigma+\tau)=u(\varsigma)$ according to Equation (7.50) and

The State Transition Matrix $\psi(k)$ is an $n\,x\,n$ matrix which is multiplied with the initial state $x(0)$ to make the state $x(k)$ at any time $k\geq 0$ as:

$$x(k)=\psi(k)x(0) \quad \textbf{(7.58)}$$

It satisfies the homogenous state equation with zero input:

$$x(k+1)=Ax(k) \quad \textbf{(7.59)}$$

$$\psi(k+1)x(0)=A\psi(k)x(0) \quad \textbf{(7.60)}$$

$$\psi(k+1)=A\psi(k) \quad \textbf{(7.61)}$$

Where the initial condition is $\psi(0)=I$

To find the solution for $\psi(k)$ the z-transformation is used:

$$z\psi(z)-z\psi(0)=A\psi(z) \quad \textbf{(7.62)}$$

and solve for $\psi(z)$

$$\psi(z) = \left[zI - A\right]^{-1} z\psi(0) = \left[zI - A\right]^{-1} z \tag{7.63}$$

$$\psi(z) = \left[I - z^{-1}A\right]^{-1} \tag{7.64}$$

$$\psi(z) = I + Az^{-1} + A^2 z^{-2} + \ldots \tag{7.65}$$

Taking the inverse transform of $\psi(z)$ it is obtained

$$\psi(k) = Z^{-1}\{\psi(z)\} = Z^{-1}\left\{\left[zI - A\right]^{-1} z\right\} \tag{7.66}$$

$$\psi(k) = A^k \tag{7.67}$$

Equation (7.67) is the discrete time LTI state transition matrix.

If the transition matrix $\psi(\tau)$ is considered for the model Equations (7.53) and (7.54) it can be written

$$\hat{x}(k+\tau) = A^{\tau}\hat{x}(k) + \sum_{\varsigma=k-\tau}^{k-1}\left[A^{k-\varsigma}\left[Bu(\varsigma) + WT_w(\varsigma+\tau)\right]\right] \tag{7.68}$$

Equation (7.68) represents the future value of the state estimates through the current and past vales of the state estimates and control action and through the future values of the deterministic disturbance. On the basis of the Equation (7.68) the closed loop control is expressed in the following way:

$$u(k) = -L\left\{A^{\tau}\hat{x}(k) + \sum_{\varsigma=k-\tau}^{k-1}\left[A^{k-\varsigma}\left[Bu(\varsigma) + WT_w(\varsigma+\tau)\right]\right]\right\} + L_g g(k+\tau+1) + L_w g_w(k+1+\tau) \tag{7.69}$$

$$k = \overline{0, k_f - 1}$$

For simplification the value of the deterministic disturbance is accepted as a constant. The diagram of the closed loop system is shown in Figure (7.4).

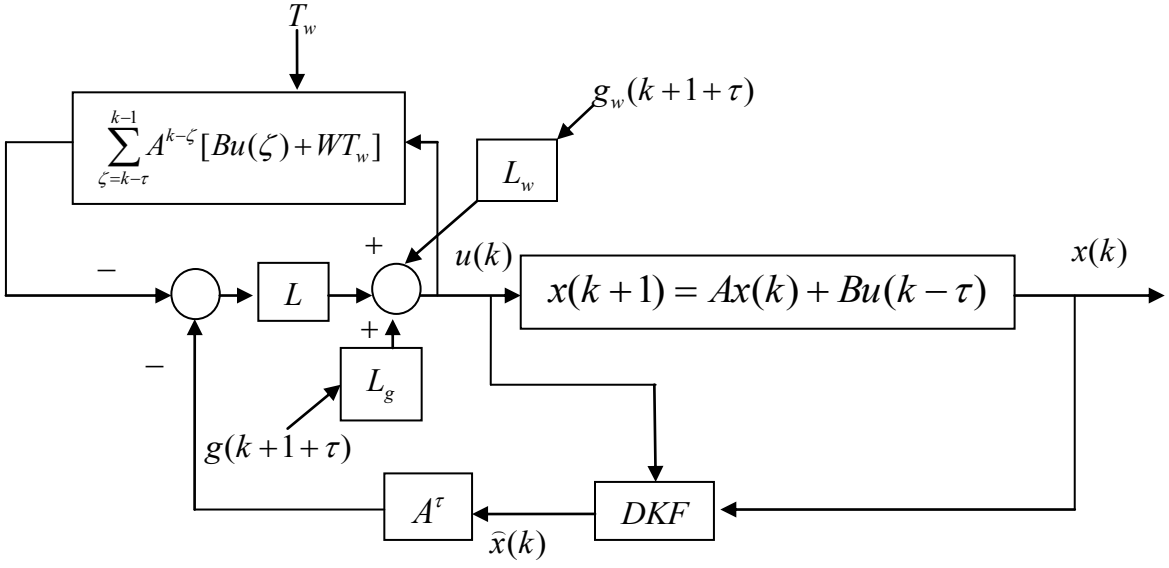


Figure 7.4: **Closed loop control system with time delay represented with State Matrix**

Buffers are used in the real-time implementation of the control in order to keep the past values of the state estimates and control action for a period equal to the time delay. The new values are updated at every time moment **k**.

The controller given by Equation (7.69) is optimal because it minimises the criterion for optimality. It is predictive because its control action is calculated on the basis of the future values of the state estimates, expressed by the past values of the state estimates and control actions for a period equal to the delay. The controller is also robust according to the influence of the delays in the communication channel because it guarantees stability and good performance of the system, never the less the value of the delays.

The structure in Figure (7.4) is used for further investigations of the influence of the delays in the communication channels over the performance of the closed loop system and of the capability of the designed robust predictive controller to overcome the influence of the delays between the controller and the actuator and between the sensor and the controller.

The design of the controller matrices can be done also using other methods as poles placement.

7.5 Conclusion

The problem for the destabilising effect of the network induced time delay between the controller and the actuator is solved by development of state space predictive controller using time shifting approach and state predictor. The design of the controller is based on a model of the plant with delay in the control vector equal to the delay between the controller and the actuator and to the sum of the delays between the controller and the actuator and between the sensor ad the controller. The time shifting approach allows the design of the controller to be performed for a model without time delay. Then the control action is based on the future values of the state space vector estimates.

The state predictor is developed to predict the future value of the state using the present and past values of the state estimates and control actions. This technique is made possible by the use of the plant model Transition Matrix. The general transition matrix is adapted and used for the prediction of the future state. The solution is simple and works with time delays that are greater than the sampling period, also it has minimum storage requirements and it is not computationally heavy.

The verification of the capabilities of the designed optimal predictive robust controller is done in Chapter 8 where simulations for various scenarios of the delays and controllers are performed.

CHAPTER EIGHT
SIMULATION OF THE DEVELOPED METHODS

8.1 Introduction

This chapter presents simulation of the method developed for networked control system. The derivations of the methods is given in Chapter 7.

The following steps are considered when designing the controller that will compensate for the effects caused by network induced time delay:

- Controller and DKF are designed for a NCS without delay using Pole Placement and Optimal Control methods. The delays between controller and actuator and between the sensor and controller are introduced and their impact over the closed loop system behaviour is investigated.
- Control augmentation approach. In this approach the time delay between the controller and actuator is incorporated in the control vector of the plant model. The plant is considered as a system with time delay during the design of the robust predictive controller.
- Control augmentation approach. The sum of the time delays between the controller and the actuator and between the sensor and the controller is incorporated in the control vector of the plant model. Robust predictive controller is designed.
- In all considered cases the normal Kalman filter equations are used.

8.2 Simulation Environment

The general structure for the simulation is given in Figure (8.1). Matlab and Simulink are used as simulation tools for the experiments.

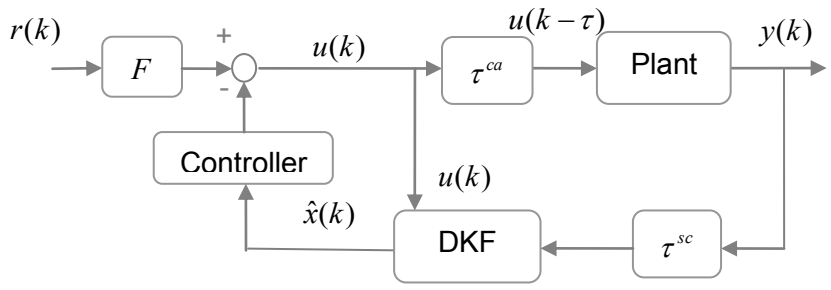


Figure 8.1: Simulation Structure with NITD

The continuous plant model is derived in Chapter 4. Matlab function "c2d.m" (refer to "c2d" in Matlab help) converts the continuous LTI model to a discrete time model. The discrete time LTI model of the plant as used in these simulations is therefore.

$$x(k+1) = Ax(k) + Bu(k) \quad \textbf{(8.1)}$$

$$y(k) = Cx(k) \quad \textbf{(8.2)}$$

Where A, B and C are discrete time matrices with proper dimensions. The discrete state space model matrices are obtained for the sampling period T = 0.02 seconds.

The following DKF time and measurement update equations are used to simulate the DKF and they are derived in Chapter 6 Equation (8.3) to (8.7) for systems without measurement delays.

Time update equations:

$$\hat{x}(k)^{-} = A\hat{x}(k-1) + Bu(k) \quad \textbf{(8.3)}$$

$$P^{-}(k) = AP(k-1)A^{T} + Q \quad \textbf{(8.4)}$$

Measurement update equations:

$$K(k) = P^{-}(k)C^{T}[CP^{-}(k)C^{T} + R]^{-1} \quad \textbf{(8.5)}$$

$$\hat{x}(k) = \hat{x}(k)^{-} + K(k)[y(k) - C\hat{x}(k)^{-}] \quad \textbf{(8.6)}$$

$$P(k) = [I - K(k)C]P^{-}(k) \quad \textbf{(8.7)}$$

Where mean and covariance of the noises in the state and measurement equations are calculated infront:

The pole placement controller synthesis is derived in Chapter 5 and is given by the following equation for the systems without delays:

$$u(k) = Fr(k) - L\hat{x}(k), \quad \textbf{(8.8)}$$

and by the equation:

$$u(k) = Fr(k) - L\left[A^{\tau^{ca}}\hat{x}(k) + \sum_{\zeta=k-\tau^{ca}}^{k-1} A^{k+\tau^{ca}-\zeta}Bu(\zeta)\right] \quad \textbf{(8.9)}$$

for systems with delay, where: $L = \begin{bmatrix} 0.5521 & 0.5600 \end{bmatrix}$ and $F = 0.55$.

The Optimal Controller is derived in Chapter 7 and is given by the equation:

$$u(k) = Fr(k) - L(k)\hat{x}(k) + L_{g}(k)g(k+1) + L_{w}(k)g_{w}(k+1) \quad \textbf{(8.10)}$$

for a final period of control, and is given by the equation:

$$u(k) = -L\hat{x}(k) + L_{g}g + L_{w}g_{w} \quad \textbf{(8.11)}$$

for infinite period of control. Equation (8.11) is used in the simulations for systems without delays where:

$L_{g} = \begin{bmatrix} 0.5521 & 0.5600 \end{bmatrix}$ and $L_{w} = \begin{bmatrix} 0.5521 & 0.5600 \end{bmatrix}$.

For systems with delays the controller is:

$$u(k) = Fr(k) - L\left[A^{\tau^{ca}} \hat{x}(k) + \sum_{\zeta=k-\tau^{ca}}^{k-1} A^{k+\tau^{ca}-\zeta} Bu(\zeta) \right] + L_g g + L_w g_w \quad \textbf{(8.12)}$$

8.3 Simulation Procedures

Using the structure in Figure (8.1) and Equations (8.1) to (8.12) the simulations procedures are performed. The simulations are considered for both Azimuth and Altitude drives. The results shown in the graphs are plotted for Altitude drive only as the response is from the same type on both drives. The reference input is given in degrees.

For simulation procedures where network induced time delays are considered, the values of the delays in the NCS are selected to be:

- delay is less than the sampling period, $\tau_T < T$, where $T = 0.02$ seconds (sampling period).
- delay is greater than the sampling period, $T < \tau_T < \tau_c$
- delay is equal to the critical delay (τ_c), $\tau_T = \tau_c$.
- delay is greater than the critical delay, $\tau_T > \tau_c$

Where $\tau_T = \tau^{ca} + \tau^{sc}$ **(8.13)**

These considerations are repeated for **cases** where:

- delays are considered to be between the controller and the actuator ONLY; τ^{ca}
- delays are considered to be between the sensor and the controller ONLY; τ^{sc}
- delays are considered to be between the controller and the actuator τ^{ca} and between the sensor and the controller τ^{sc}.
- delays are independent and calculated using the distribution from Nilsson, 1998.
  - $$f_\tau\left(\tau_k^{sc}\right) = \begin{cases} \delta\left(\tau_k^{sc} - a\right).\left(1 - p_{sc}\right), & \tau_k^{sc} = a, \\ p_{sc} / \left(b - a^+\right), & \tau_k^{sc} \in (a,b), a < b, \\ 0, & \tau_k^{sc} \notin [a,b], \end{cases} \quad \textbf{(8.14)}$$
  - $$f_\tau\left(\tau_k^{ca}\right) = \delta\left(\tau_k^{ca} - a\right).\left(1 - p_{ca}\right) + \delta\left(\tau_k^{ca} - b\right).p_{ca} \quad \textbf{(8.15)}$$

Based on the performed simulations the delay boundaries are identified and used to measure the performance of the developed control methods.

The following sample delay values are used on all the simulations that have network induced time delays.

Table 8.1: **Delays values for different considerations**

Delay τ_T Consideration	Delay [number of samples(n)]			Delay τ_T [seconds]
	τ^{ca}	τ^{sc}	τ_T	
$\tau_T = T$	0	0	0	0
$\tau_T > T$	5	5	10	0.2
$\tau_T = \tau_c$	15	20	35	0.7
$\tau_T > \tau_c$	25	25	50	1

Instead of using a step response input an input signal produced by a pulse generator is used for the analysis of the response of the system. It has the following specifications:

- Amplitude = 1
- Period (number of samples) = 500 (10 sec)
- Pulse width (number of samples) = 250 (5 sec)
- Initial time *k=0*.

The pulse generator is chosen because the signal changes set point from 1 to 0 during a simulation cycle allowing to observe the behaviour of the time delay on the falling and rising edges of the pulse, thereby verifying consistency of the life cycle of each sample and the delays associated with the sample during the cycle.

The resultant from the pulse generated response simulation graphs can be interpreted as follows :

- Each simulation has an output graph that has four sub-plots which represent the response of the closed loop system for the different delay considerations as described in Table (8.1).
- X axis data is the discrete time which is computed as $k*T$, where k is the number of samples and T is the sampling period.
- The network induced discrete-time delay input is specified as number of samples.
- Y axis data displays the magnitude of the reference input, the measurement output, the DKF output and the Control input (after the delay between the controller and the actuator).
- The delays are measured from the rising edge of the first pulse period that is starting at time 5 seconds (250 samples).

Three types of simulations are implemented in order to describe the influence of the network delays over the closed loop system performance:

1. Open loop control of the system consisting of the plant and the DKF - no delay are considered.
2. Closed loop control of the system consisting of the plant and the DKF - no delays are considered.
3. Closed loop control of the system consisting of the plant and the DKF - delays are considered
   - the two delays - between the controller and the actuator and between the sensor and the controller.
   - one delay - between the controller and the actuator **OR** between the sensor and the controller only.

The type two simulations are implemented for two cases of state space controllers - designed using Pole Placement and LQG methods.

The type 3 simulations are implemented for the following cases:

I. Investigation of NCS induced time delays impact over the closed loop behaviour in the case of state controller designed using **Pole Placement method** for a system ***without delays***. Use of a DKF designed for a system ***without delays***.

II. Investigation of the NCS induced time delays impact over the closed loop behavior in the case of state controller designed using **Optimal Control method** for the system ***without delays***. Use of a DKF designed for a system ***without delays***.

III. Investigation of the NCS induced time delays impact over the closed loop behavior in the case of state robust predictive controller designed using **Pole Placement method** for a system ***with delay*** in the plant control and with a constant disturbance. Use of a DKF designed for a system ***without time delays***.

IV. Investigation of the NCS induced time delays impact over the closed loop behavior in the case of state robust predictive controller designed using **Optimal Control method** for a system ***with delay*** in the plant control and with a constant disturbance. Use of a DKF designed for a system ***without time delays***.

8.4 Open and Closed loop system without time delays behavior.

Simulation procedure 1:

This simulation procedure is performed considering the ***Plant*** and the ***DKF*** without network induced time delays. The Plant model is derived in Chapter 4, Equations (8.1) and (8.2) and the DKF equations are derived in Chapter 7, Equations (8.3) to (8.7). The

purpose of this procedure is to verify the functionality of the filter and confirm that it tracks the response of the open loop discrete plant. The Simulink block diagram is shown in Figure (8.2). The Matlab file (Exp_one.m) with data for the simulation can be seen in Appendix (B8.1).

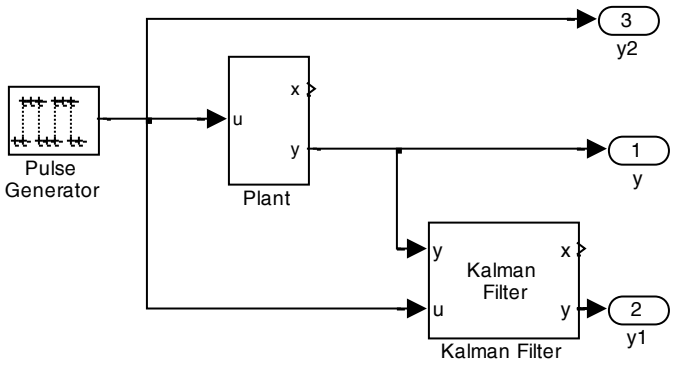


Figure 8.2: **Simulink Block diagram of Generic Plant and Generic DKF**

The results from the simulation are shown in Figure (8.3).

Response results:

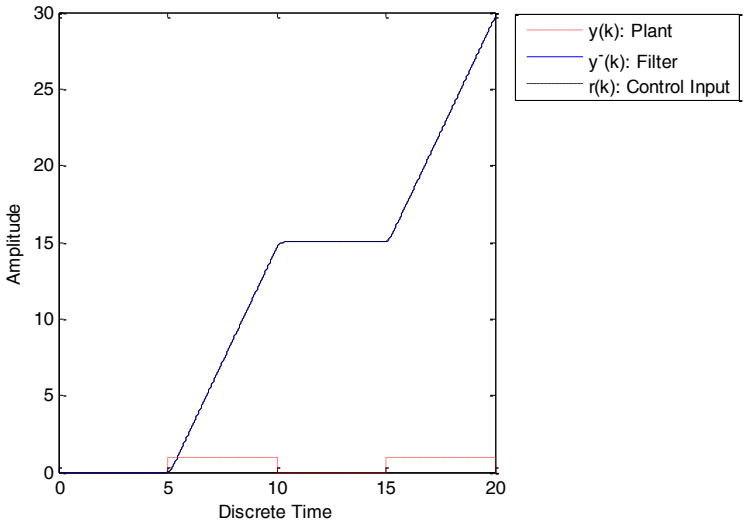


Figure 8.3: **Plant and DKF step response**

Discussion:

The result demonstrates that the DKF tracks the response of the system and confirms the functionality of the DKF.

Simulation procedure 2:

This simulation procedure is performed with the ***Plant, Controller*** and ***DKF*** without consideration of network induced time delays. The purpose of this procedure is to design a controller without paying attention to the network induced time delays and to evaluate the behavior of the closed loop system under its control. The Pole Placement controller

design method is considered in this experiment and subsequent experiments. The design of the controller gain matrix is derived in Chapter 5 in continuous form and in Chapter 7 in discrete form. In order to convert the desired continuous poles p to desired discrete pole z, the following conversion equation is used.

$$z = e^{p*T} \qquad \textbf{(8.16)}$$

where T is the sampling period.

The controller gain matrix is $L = \begin{bmatrix} 0.5531 & 0.0104 \end{bmatrix}$ and $F = 0.55$

The Simulink block diagram for the simulation is shown in Figure (8.4). The Matlab file (Exp_two.m) with data for the simulation can be seen in Appendix (B8.2).

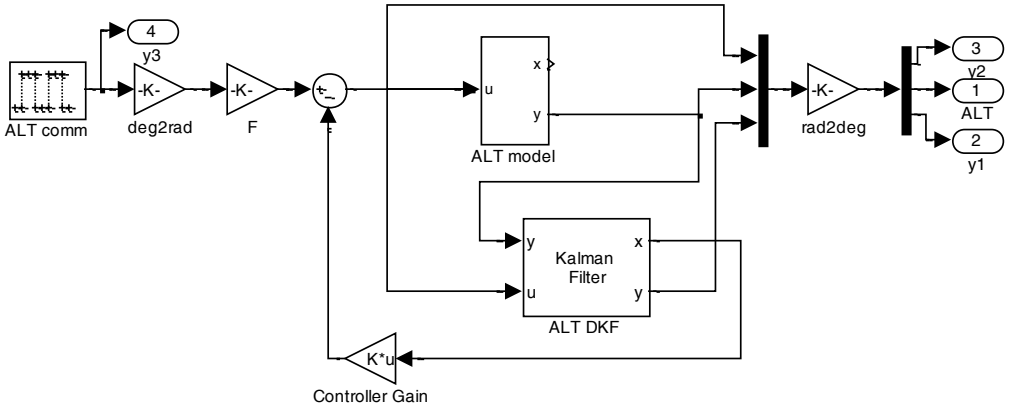


Figure 8.4: **Simulink Block diagram of the closed loop system with Pole Placement controller - no Network Induced Time Delay (NITD)**

Response results using Pole Placement and LQ controllers:

Data from the output of the plant, DKF, Controller and the reference signal are plotted in Figure (8.5) and (8.6).

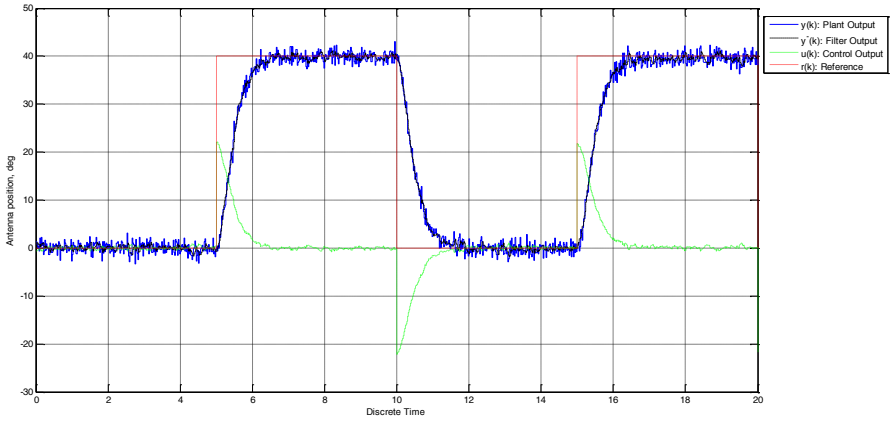


Figure 8.5: **Outputs of the Plant, Controller (PP), DKF and Reference (Pulse Generator)**

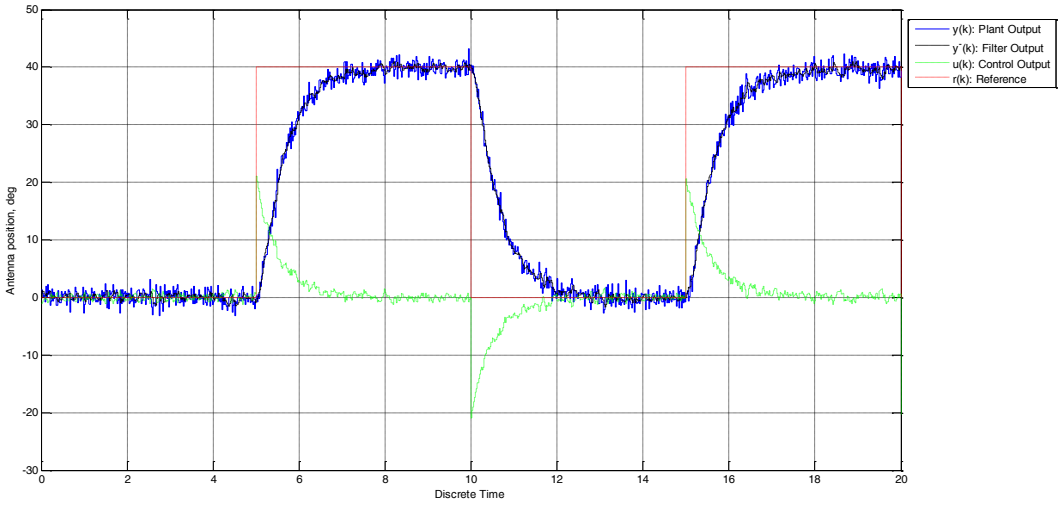


Figure 8.6: **Outputs of the Plant, Controller (LQC), DKF and Reference (Pulse Generator)**

Performance results and discussion:

The performance results for Pole Placement and LQ controllers are shown in Table (8.2):

Table 8.2: Performance comparison between Pole Placement and LQ controller

Performance Measure	Pole Placement	LQ
Rise Time	1sec	1.48sec
Settling Time	1.42sec	1.5sec
Percentage Overshoot	0%	0%
Steady State Error	0.56%	0.42%

The results meet the performance specification as defined in Chapter 5. The DKF successfully follows the plant output. In this case the Pole Placement has a better transition behavior compared to the LQ controller.

8.5 Closed loop system with constant time delay behavior with the controller and the DKF designed for a plant without delays

Simulation procedure 3:

This simulation procedure is performed with the same ***Plant, Controller*** and ***DKF*** but ***constant network induced time delays*** are introduced. In this simulation the delays are considered between the controller and the actuator and between the sensor and the controller. The purpose of this procedure is to illustrate the influence of the network induced time delay in the case when the controller and DKF are not designed to cope

with the delays. The controller designed using Pole Placement method is considered in this experiment using the same parameters as in Simulation procedure 2. The Simulink block diagram for the simulation is shown in Figure (8.7). The Matlab file (Exp_three.m) with data for the simulation can be seen in Appendix (B8.3).

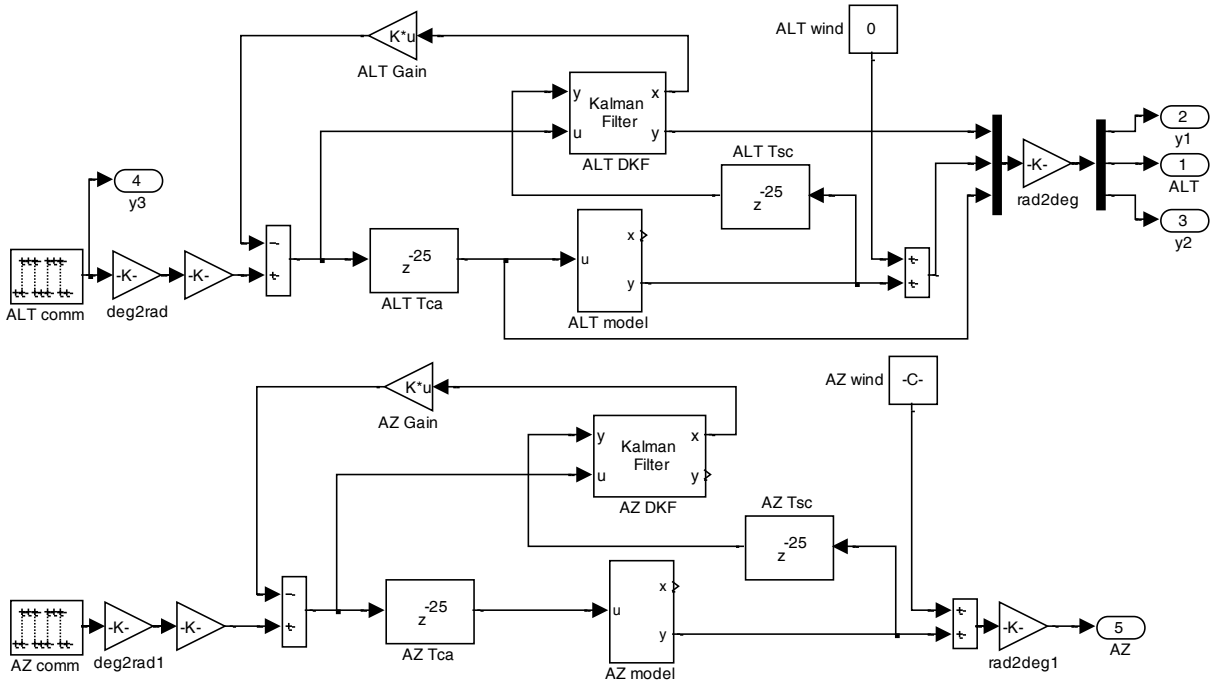


Figure 8.7: **Simulink block diagram of the closed loop system with PP Controller - constant NITD** (τ^{sc}, τ^{ca})

Response results:

The system response results are for the case when both time delays are present. The outputs of the plant, controller, DKF and the pulse generator are shown in Figure (8.8) for the four considered cases of delays, Table (8.1).

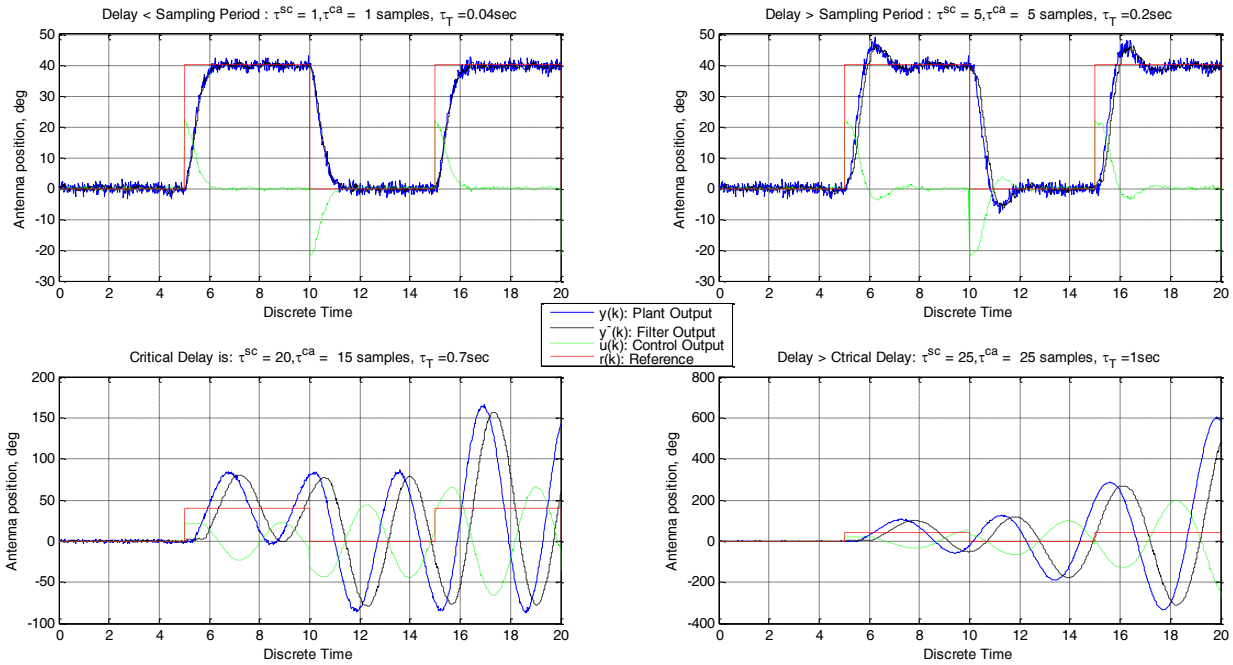


Figure 8.8: **Outputs of the Plant, Controller, DKF and Pulse generator - constant NITD (τ^{sc}, τ^{ca})**

Figure (8.9) shows a timing diagram zooming at the first fifty samples of the step response results graphs. This level of zooming allows for sample by sample analysis which gives a better understanding of the system performance in terms of:

- the introduced time delays
- the delay influence on the measurement output,
- the delay influence on DKF measurement output and
- control input after the introduction of the delay between the controller and actuator.

Note that the noise must be switched off in order to get a clear view of the delays. The delays are extracted and populated in the same manner for all the subsequent experiments.

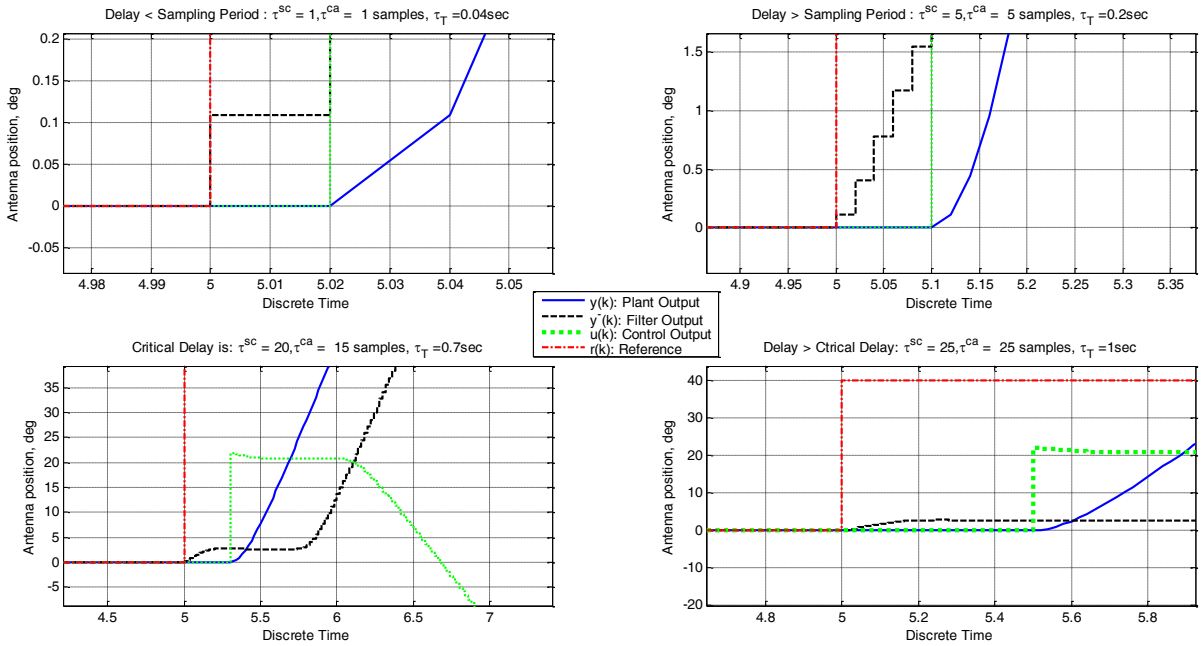


Figure 8.9: Timing diagram both delays (τ^{sc}, τ^{ca})

Performance results:

The characteristics of the transition behaviour are shown in Table (8.3).

Table 8.3: Transition behavior characteristics. Measured delays in the plant output, controller input and DKF output - both delays.

Introduced		Performance Measure				Measured Delays	
Delay Cases	Injected Delay Values	Rise Time	Settling Time	Percentage Overshoot	Steady State Error	τ_y	τ_u
PLANT OUTPUT VALUES							
$\tau_T < T$	0.02	0.96	1.42	0	0	0.02	0
$T < \tau_T < \tau_c$	0.2	0.75	2.98	14%	0.5%	0.1	0.1
$\tau_T = \tau_c$	0.7	The system oscillates $\tau_T = 0.7$ sec				**0.3**	**0.4**
$\tau_T > \tau_c$	1	The system is unstable $\tau_T = 1$ sec				**0.5**	**0.5**
DKF OUTPUT VALUES						τ_y	$\tau_{\bar{y}}$
$\tau_T < T$	0.02	0.96	1.42	0	0	0.02	-0.04
$T < \tau_T < \tau_c$	0.2	0.88	2.8	14%	0.5%	0.1	0.08
$\tau_T = \tau_c$	0.7	The system oscillates $\tau_T = 0.7$ sec				**0.7**	**0.35**
$\tau_T > \tau_c$	1	The system is unstable $\tau_T = 1$ sec				**0.1**	**0.5**

Discussion:

For the case of $T < \tau_T < \tau_c$. The plant output is delayed with the delay value of the controller actuator - $y(k-\tau^{ca})$. The control input is delayed with the delay value of the controller actuator - $u(k-\tau^{ca})$. The DKF output is delayed with the delay values $\tau_T = \tau^{ca} + \tau^{sc}$. At the environment of the initial conditions and during the period of τ_T the filter is working without delay but at the end of τ_T period and during the rise time the sensor controller delay makes it impossible for the filter to follow the measurement output. The delay between the plant output and the filter output is equal to τ^{sc}.

For $\tau_T = \tau_c$. The control output delays with delay equal to τ^{ca}. The plant output delays with delay equal to τ^{ca}. The filter output delays with delay equal to $\tau_T = \tau^{ca} + \tau^{sc}$. In the environment of the initial conditions and for the period of τ^{sc} the filter output is faster than the plant output, but after this period it delays from the plant output with delay equal to τ^{sc}.

For $\tau_T > \tau_c$. The plant output and its control input delays with τ^{ca}. The filter output delays with delay equal to τ_T. In this case the filter output has no period where its values are ahead the values of the plant output. This means that the plant always works with delayed values of the filter estimates and this causes instability of the system.
In the previous case where there were some periods where the filter contributed to the stability of the system. These periods depend on the values of the delays.

Simulation procedure 4:

This simulation procedure is performed with the same ***Plant, Controller*** and ***DKF*** but ***constant network induced time delays*** are introduced. This simulation procedure is similar to simulation procedure 3, the difference is that in this simulation the delays are considered between **the controller and the actuator** ONLY. The Simulink block diagram for simulation is shown in Figure (8.10). The Matlab file (Exp_four) with data for the simulation can be seen in Appendix (B8.4).

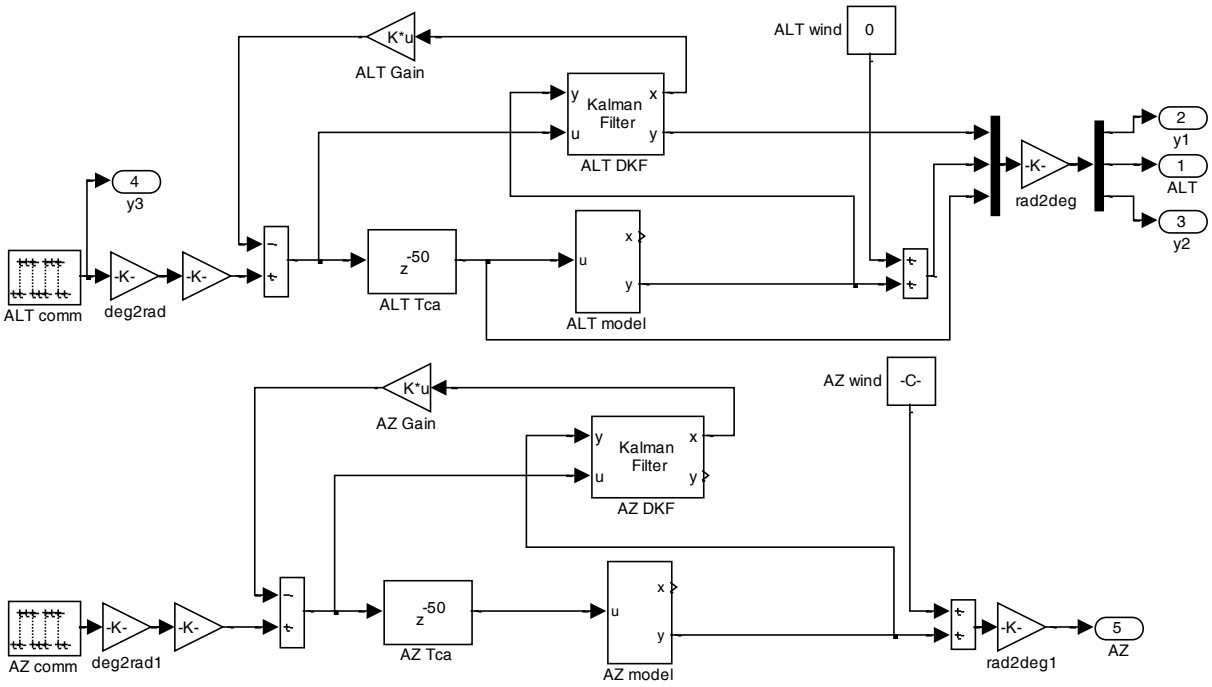


Figure 8.10: **Simulation block diagram of the closed loop system State Space Controller (PP) and Constant NITD (τ^{ca})**

Response results:

The system response results are for the case when the controller actuator delay is present. The outputs of the plant, controller, DKF and the pulse generator are shown in Figure (8.11) and Figure (8.12).

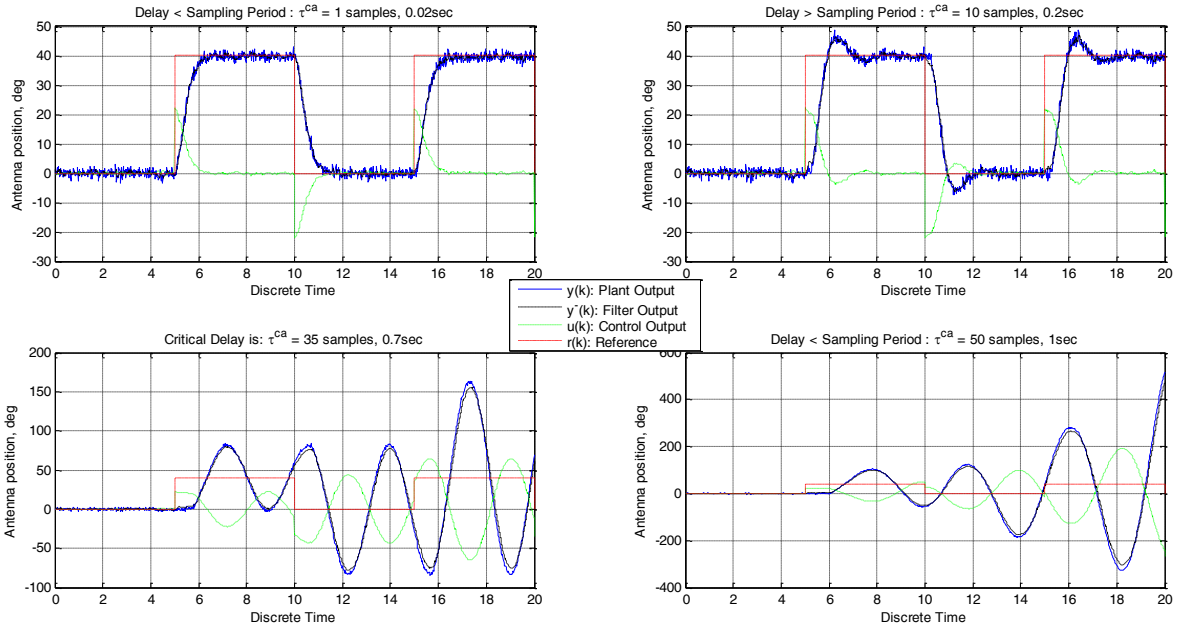


Figure 8.11: **Output of the Plant, Controller (PP) , DKF and Pulse generator -constant NITD (τ^{ca})**

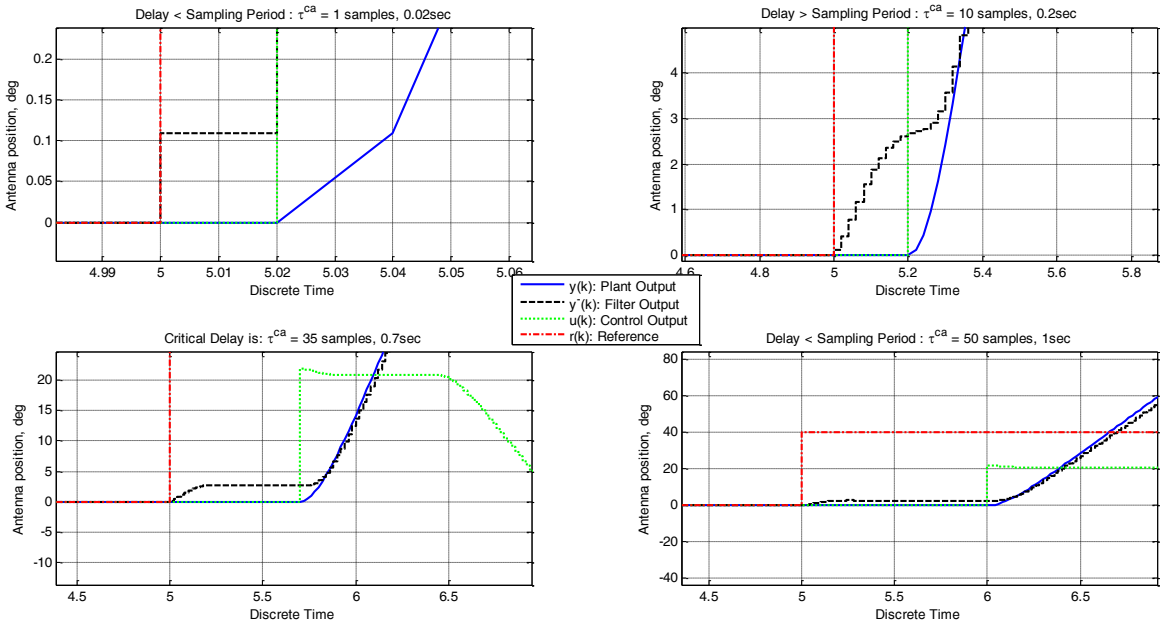


Figure 8.12: Timing diagram for constant NITD (τ^{ca})

Performance results:

The characteristics of the transition behaviour are shown in Table (8.4).

Table 8.4: Transition behavior characteristics. Measured delays in the plant output, controller input and DKF output - controller actuator delay only.

Delay Considerations		Performance Measure				Measured Delays	
Criteria	Injected delay values	Rise Time [sec]	Settling Time [sec]	Percentage Overshoot	Steady State Error	τ_y [sec]	τ_u [sec]
	PLANT OUTPUT VALUES						
$\tau^{ca} < T$	0.01	0.95	1.42	0	0	0.02	0.02
$T < \tau^{ca} < \tau_c$	0.2	0.85	2.14	14%	0.5%	0.2	0.2
$\tau^{ca} = \tau_c$	**0.7**	**The system oscillates $\tau^{ca} = 0.7$ sec**				**0.7**	**0.7**
$\tau^{ca} > \tau_c$	**1**	**The system is unstable $\tau^{ca} = 1$ sec**				**1**	**1**
	DKF OUTPUT VALUES					τ_y	$\tau_{\hat{y}}$
$\tau^{ca} < T$	0.01	0.95	1.42	0	0	0.02	0.04
$T < \tau^{ca} < \tau_c$	0.2	0.87	1.42	14%	0.5%	0.2	0
$\tau^{ca} = \tau_c$	**0.7**	**The system oscillates $\tau^{ca} = 0.7$ sec**				**0.7**	**0.4**
$\tau^{ca} > \tau_c$	**1**	**The system is unstable $\tau^{ca} = 1$ sec**				**1**	**1**

Discussion:

For the case of $\tau^{ca} < T$. The control input and plant output delays with τ^{ca}. The filter works with delay. At the environment of the initial conditions and during the period of τ^{ca} the filter is working without delay but after this period the filter exactly follows the plant output and $\tau_F = 0$.

For the case of $T < \tau^{ca} < \tau_c$. The plant output is delayed with the delay value of the controller actuator $-y(k-\tau^{ca})$. The control input is delayed with the delay value of the controller actuator $-u(k-\tau^{ca})$. The DKF output is delayed by the delay value of the controller actuator $\hat{y}(k-\tau^{ca})$. The filter follows the measurement output starting at 0.35sec. Before this time the values of the filter output are bigger than the values of the plant output.

For the case of $\tau^{ca} = \tau_c$. The output of the plant, filter and control input have the same delay equal to τ^{ca}. The values of the filter output during this period are bigger than the values of the output at the end of τ^{ca} the filter follows the plant output without delay.

For the case of $\tau^{ca} > \tau_c$. The behavior of the system is similar to this of the previous case. The delay between the controller and actuator appears in all considered trajectories. Its values influence the quality of transition behavior and stability of the system and makes the filter output not to follow exactly the plant output.

Simulation procedure 5:

This simulation procedure is performed with the same ***Plant, Controller*** and ***DKF*** but ***constant network induced time delays*** are introduced. This simulation procedure is similar to simulation procedure 3, the difference is that here the delays are considered between **sensor and controller** ONLY. The Simulink block diagram for simulation is shown in Figure (8.13). The Matlab file (exp_five.m) with data for the simulation can be seen in Appendix (B8.5).

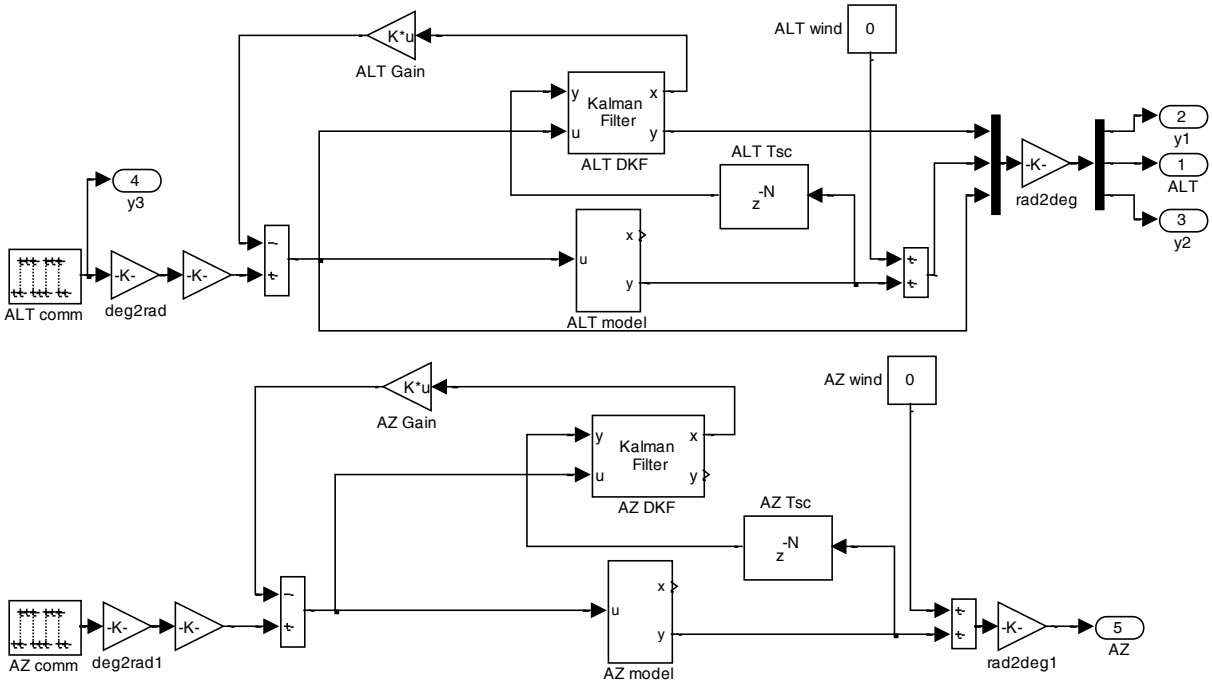


Figure 8.13: Simulation block diagram of the closed loop system State Controller (PP) and constant NITD (τ^{sc})

Response results:

The results from the simulations are shown in Figure (8.14) and (8.15).

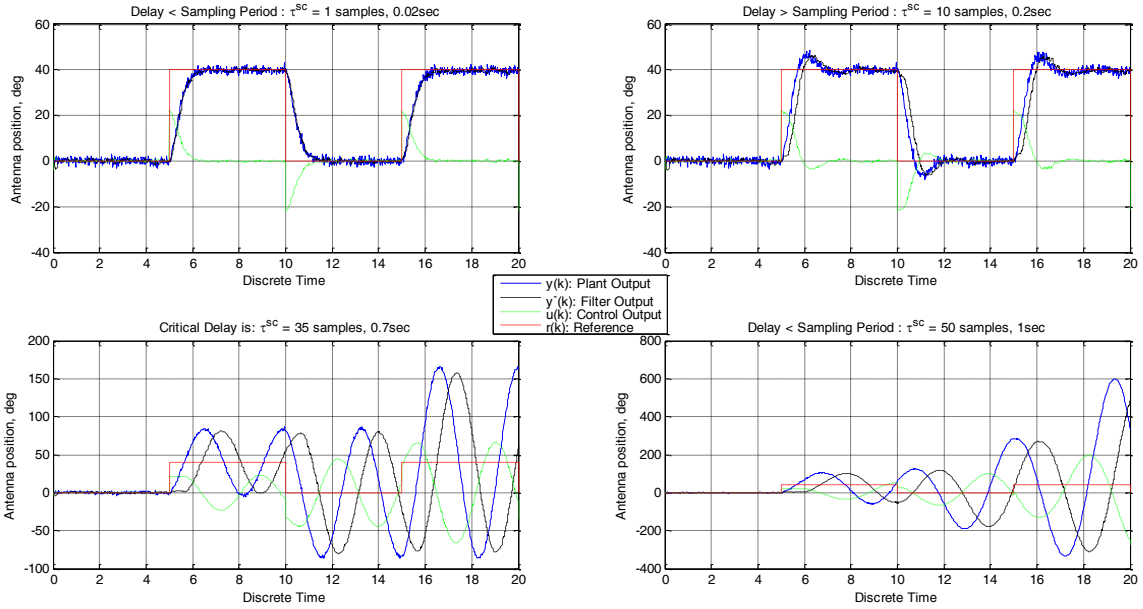


Figure 8.14: Output of the Plant, Controller, DKF and Pulse generator - constant NITD (τ^{sc})

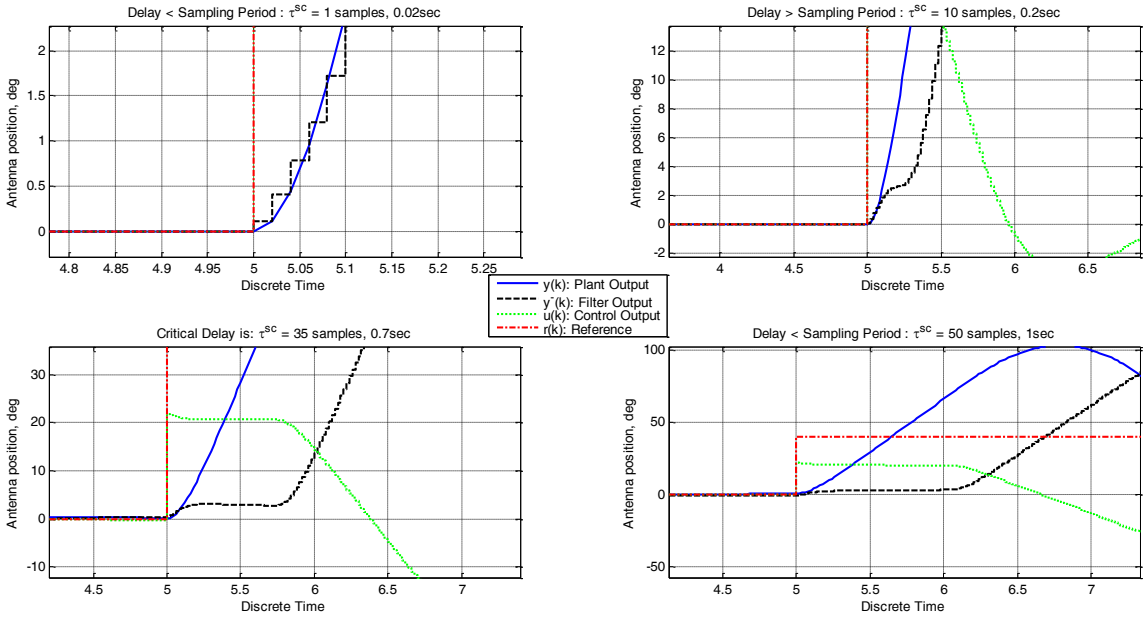


Figure 8.15: Timing diagram for constant NITD (τ^{sc})

Performance results:

The characteristics of the transition behavior of the plant output and DKF output are shown in Table (8.5).

Table 8.5: Transition behavior characteristics. Measured delays in the plant output, controller input and DKF output - sensor controller delay only.

Delay Consideration		Performance Measure				Measured Delays	
Criteria	Injected Delay Values	Rise Time	Settling Time	Percentage Overshoot	Steady State Error	τ_y	τ_u
PLANT OUTPUT VALUES							
$\tau^{sc} < T$	0.01	0.96	1.42	0%	0.5%	0	0
$T < \tau^{sc} < \tau_c$	0.2	0.66	2.58	0.18	0.5%	0	0
$\tau^{sc} = \tau_c$	**0.7**	**The system oscillates $\tau^{sc} = 0.7$ sec**				**0**	**0.7**
$\tau^{sc} > \tau_c$	**1**	**The system is unstable $\tau^{sc} = 1$ sec**				**0**	**1**
DKF OUTPUT VALUES						τ_y	$\tau_{\bar{y}}$
$\tau^{sc} < T$	0.01	0.96	1.42	0%	0.5%	0.02	0.02
$T < \tau^{sc} < \tau_c$	0.2	0.66	2.58	14.5%	0.5%	0.2	0.2
$\tau^{sc} = \tau_c$	**0.7**	**The system oscillates $\tau^{sc} = 0.7$ sec**				**0.7**	**0.7**
$\tau^{sc} > \tau_c$	**1**	**The system is unstable $\tau^{sc} = 1$ sec**				**1**	**1**

Discussion:

For the case of $\tau^{sc} < T$. No delays in any of the measured trajectories. The filter follows exactly the plant output.

For the case of $T < \tau^{sc} < \tau_c$ .The measurement output and control input are not delayed by the value of τ^{sc}. The DKF output is delayed by the delay value of τ^{sc}. The filter follows the measurement output signal with delay of τ^{sc}.

For the case of $\tau^{sc} = \tau_c$. The filter follows the plant output with delay equal to τ^{sc}. The controller input is delayed with delay value of τ^{sc}. No delays in the plant output.

For the case of $\tau^{sc} > \tau_c$. The behavior according to time delay is similar to the above one.

It appears that the delay between the sensor and the controller appears only in the output of the DKF, but its value influences the quality of behavior and stability of the system.

8.6 Discussion on results for constant network induced time delay case

The values of the time delays measured in the plant, controller and DKF outputs are grouped in Table (8.6).

Table 8.6: Transition behavior characteristics. Measured delays in the plant output, controller input and DKF output - Analysis.

Considerations	Measured delays	τ^{ca} [sec]			τ^{sc} [sec]			Both delays [sec]		
Measurements/Scenarios	Injected delay values	$y(k)$	$\hat{y}(k)$	$u(k)$	$y(k)$	$\hat{y}(k)$	$u(k)$	$y(k)$	$\hat{y}(k)$	$u(k)$
$\tau_T < T$	0.02	0.02	0.01	0.02	0	0.02	0	0.01	0.02	0.01
$T < \tau_T < \tau_c$	0.2	0.2	0.02	0.2	0	0.2	0	0.1	0.2	0.1
$\tau_T = \tau_c$	0.7	0.7	0.7	0.7	0	0.7	0	0.3	0.4	0.3
$\tau_T > \tau_c$	1	1	1	1	0	1	0	0.5	1	0.5

The results demonstrate that when the time delay between the **controller and the actuator** is considered alone the results are:

- the measurement output is delayed by τ^{ca} time delay value.

- DKF is not delayed by τ^{ca} time delay value but it follows the measurement output with some error that is bigger when the value of the delay is bigger. This means that the total delay of the DKF output is τ^{ca}.
- The output of the control measured after the controller actuator delay, is delayed by τ^{ca}.

When the delay between the sensor and the controller is considered alone the results are:

- the measurement output is NOT delayed by τ^{sc} time delay value, instead the performance of the controller is affected by the sensor controller delay.
- DKF is delayed by τ^{sc} time delay value and the DKF tracks the measurement output with some error that depends on the value of the delay.
- The output of the controller is NOT delayed.

When the delays between the controller and the actuator and the sensor and the controller are considered the results are:

- The measurement output is delayed by τ^{ca} time delay value and the performance of the controller is affected by τ^{sc} delay value.
- DKF output is delayed by $\tau^{ca} + \tau^{sc}$ time delay value and tracks the measurement output with a delay introduced by τ^{sc}.
- The output of the control measured after the controller actuator delay, is delayed by τ^{ca}, and the control is not affected by τ^{sc} delay time value.

Based on these experiments a conclusion is made that the time delay between the sensor and the controller does not appear on the measurement output. The critical time delay for the system is found to be 35 samples at 0.02 sampling rate giving 0.7seconds. If the time delay is greater than the critical delay the system becomes unstable.

The control action is not produced according to state estimate at the moment k but it is produced according to the delayed state of the system, meaning the quality of control deteriorates in the presence of delays.

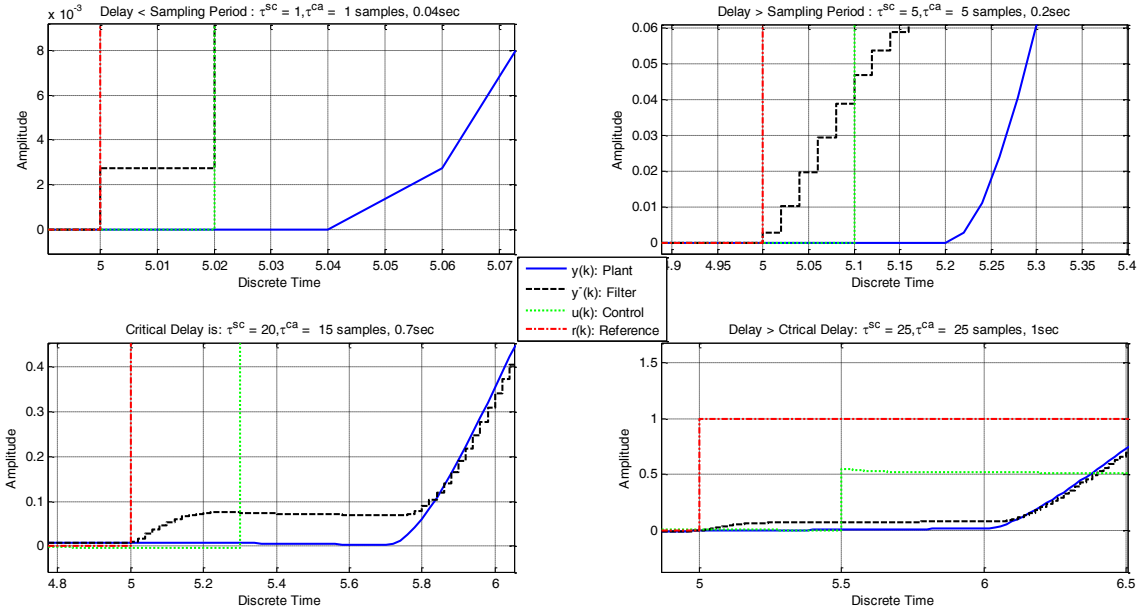


Figure 8.16: Timing diagram for constant NITD (τ^{sc})

For the case of two delays τ^{ca} and τ^{sc}. It has been observed that at the summing point just before the controller actuator delay the control is not delayed. This is the reason why the sensor controller delay does not add to the controller actuator delay. The latter implies that the two delays do not appear as an additive delay in the plant control vector as it might have been implied in a lot of papers, (Nilsson, 1998), (Heemels, 2009), (Zhang and Yu, 2008), Cloosterman et al (2010). The delay between the sensor and the controller affects the stability of the system but does not affect the measurement output delay. The model of the plant has to be augmented considering the delay between the sensor and the controller and the delay between the controller and the actuator.

The DKF output depends on the τ^{sc} delay. This delay appears in the filter output and it influences it tracking capabilities. The results obtained till now show that the time delay between the controller and the actuator and between the sensor and the controller have to be considered separately when a robust controller or filter have to be designed. The results supports the developments done in the thesis. This means that the design of the robust controller has to be based on the delay between the controller and the actuator and the design of the robust DKF has to be based on the delay between the sensor and the controller.

8.7 Constant Network Induced Time Delay Simulation using LQ method of design

This sections simulates the effects of network induced time delays considering discrete plant, discrete Kalman Filter and Linear Quadratic controller. Simulation 3 is repeated in

this section and option for LQ controller is selected from the workspace. Both delays are considered using LQ controller method to calculate controller gain matrix.

The controller gain matrix is: $L = \begin{bmatrix} 0.5521 & 0.5600 \end{bmatrix}$ and $F = 0.55$.

Simulation procedure 3a:

Response results

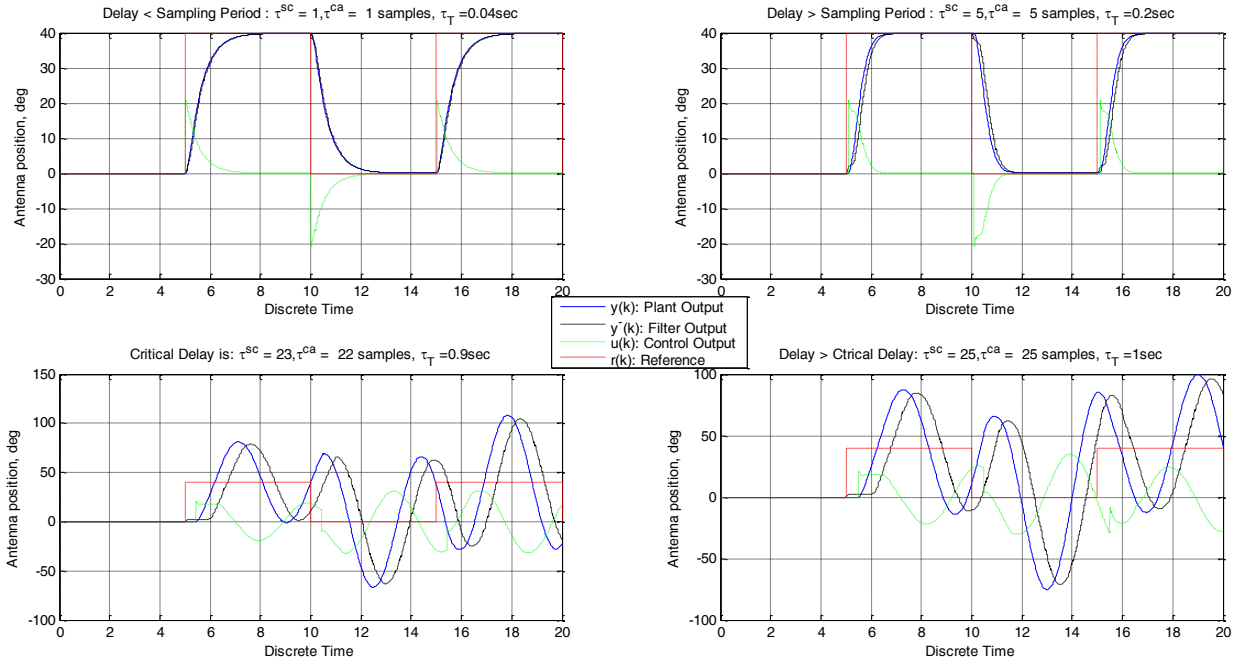


Figure 8.17: **Output of the Plant, Controller, DKF and Pulse generator LQ controller- constant NITD (τ^{ca}, τ^{sc})**

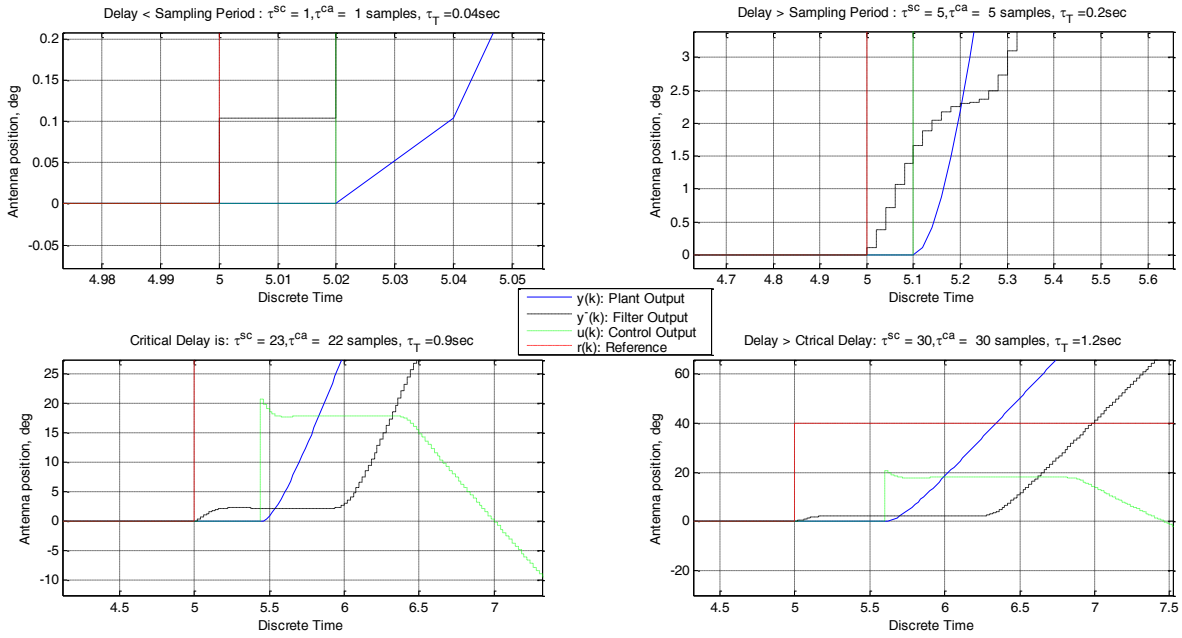


Figure 8.18: Timing diagram (Both delays)

Performance results:

The characteristics of the transition behavior of the plant output and DKF output are shown in Table (8.7).

Table 8.7: Transition behavior characteristics. Measured delays in the plant output, controller input and DKF output - both delay

Delay Consideration		Performance Measure				Measured Delays	
Criteria	Injected Delay Values	Rise Time	Settling Time	Percentage Overshoot	Steady State Error	τ_y	τ_u
PLANT OUTPUT VALUES							
$\tau^{sc} < T$	0.01	0.96	1.42	0%	0.5%	0	0
$T < \tau^{sc} < \tau_c$	0.2	0.66	2.58	0.18	0.5%	0	0
$\tau^{sc} = \tau_c$	**0.7**	**The system oscillates $\tau^{sc} = 0.7$ sec**				**0**	**0.7**
$\tau^{sc} > \tau_c$	**1**	**The system is unstable $\tau^{sc} = 1$ sec**				**0**	**1**
DKF OUTPUT VALUES						τ_y	$\tau_{\bar{y}}$
$\tau^{sc} < T$	0.01	0.96	1.42	0%	0.5%	0.02	0.02
$T < \tau^{sc} < \tau_c$	0.2	0.66	2.58	14.5%	0.5%	0.2	0.2
$\tau^{sc} = \tau_c$	**0.7**	**The system oscillates $\tau^{sc} = 0.7$ sec**				**0.7**	**0.7**
$\tau^{sc} > \tau_c$	**1**	**The system is unstable $\tau^{sc} = 1$ sec**				**1**	**1**

Discussion:

The behavior of the closed loop system is similar to the one with controller designed using Pole Placement method for all the simulations procedures. The difference is the performance, when considering critical time delay value and percentage overshoot.

The critical time delay τ_c has improved from 0.7sec to 0.9 sec when using the LQ controller.

For the case where τ_T is 0.2 sec, there is no overshoot, compared with 14.5% overshoot when using Pole Placement controller method.

For the case where $\tau_T > \tau_c$ the system is unstable.

The LQ controller proves to be more robust compared with the Pole Placement controller.

8.8 Simulation results using developed methods

Simulation procedure 6

In this simulation procedure the proposed control method for networked control system is presented. The model of the plant is augmented with the time delay between the controller and actuator τ^{ca}. The closed loop controller is designed and given by the Equation (8.17). The derivations are detailed in Chapter 7.

$$u(k) = Fr(k) - Lx(k + \tau^{ca}) \quad \textbf{(8.17)}$$

The controller gain matrix (*L*) is calculated in the same way as with experiments 3 and experiments 4 by using Pole Placement synthesis in state space form.

The solution for the controller gain matrix is: $L = \begin{bmatrix} 0.5531 & 0.0104 \end{bmatrix}$.

The outcomes of the developed method using the shifting approach is that the controller depends on the future value of the state given by ($x(k+\tau)$) as shown by Equation (8.18). where $\tau = \tau^{ca} + \tau^{sc}$.

$$x(k+\tau) = A^{\tau}x(k) + \sum_{\zeta=k-\tau}^{k-1} A^{k+\tau-\zeta} Bu(\zeta) \quad \textbf{(8.18)}$$

A state predictor is developed to predict the future value of the state using the present and past values of the state and control. This technique is made possible by the use of Transition Matrix. The general transition matrix is adapted and used for the prediction of the future state. The Simulink block diagram that implements the state predictor is shown in Figure (8.19). The results are shown in Figures (20) to (23).

This simulation procedure is performed with the same **Plant**, **Predictive Robust Controller** and **DKF and constant network induced time delays are introduced**. This simulation procedure is similar to simulation procedure 4. The Simulink block diagram for the simulation is shown in Figure (8.19). The Matlab file (exp_six.m) with data for the simulation can be seen in Appendix (B8.6). The level 2 M-file S-function code is attached as part of Appendix (B8.6)

The simulations are performed using the delay values as used in the previous experiments. The results shows that the developed control method is robust and works with network induced delays that are greater than the sampling period.

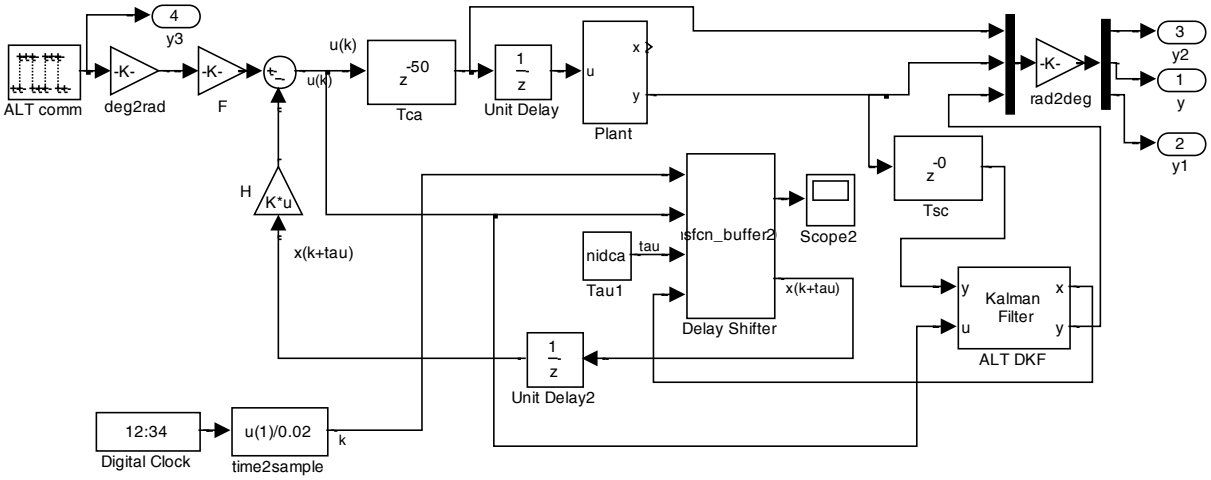


Figure 8.19: **Simulink block diagram for the closed looped developed method - NITD (τ^{ca})**

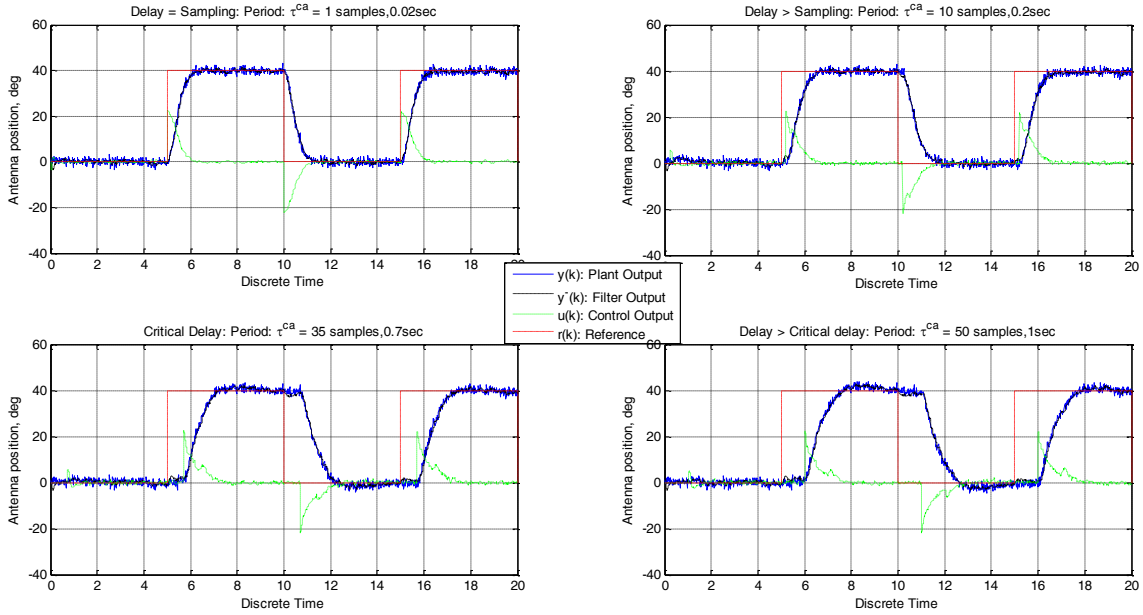


Figure 8.20: **Output of the Plant, Controller and DKF - NITD (τ^{ca})**

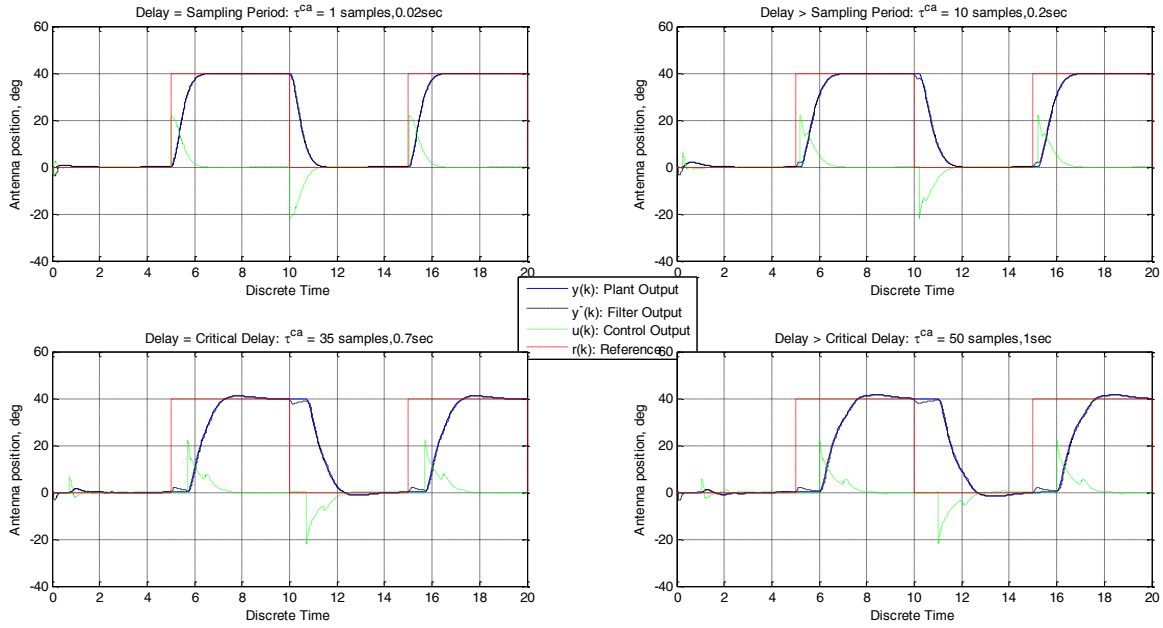


Figure 8.21: Output of the Plant, Controller without noise- constant NITD (τ^{ca})

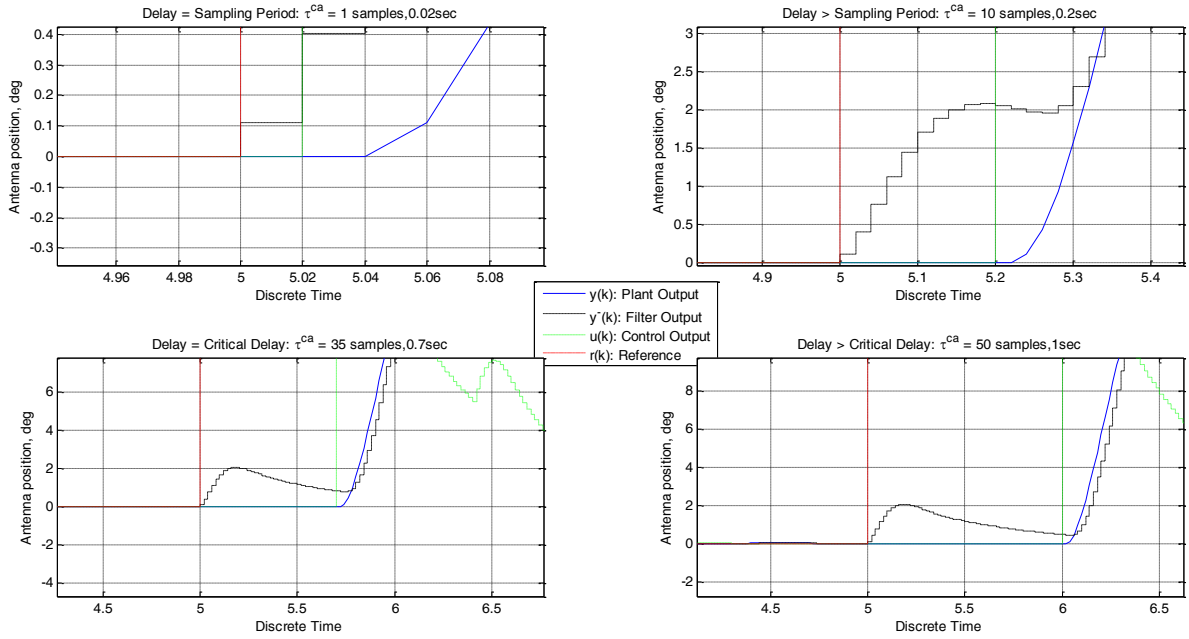


Figure 8.22: Timing diagram for constant NITD (τ^{ca})

Performance results:

The characteristics of the transition behavior of the plant output and DKF output are shown in Table (8.8).

Table 8.8: **Transition behavior characteristics. Measured delays in the plant output, controller input and DKF output - Controller Actuator delay τ^{ca}**

Delay Consideration		Performance Measure				Measured Delays	
Criteria	Injected Delay Values	Rise Time	Settling Time	Percentage Overshoot	Steady State Error	τ_y	τ_u
PLANT OUTPUT VALUES							
$\tau^{ca} < T$	0.02	0.74	1.28	0%	1%	0.02	0.02
$T < \tau^{ca} < \tau_c$	0.2	0.9	1.7	0%	1%	0.2	0.2
$\tau^{ca} = \tau_c$	0.7	1.04	3.76	3%	1%	0.7	0.7
$\tau^{ca} > \tau_c$	1	1.16	4.6	3.8%	1%	1	1
DKF OUTPUT VALUES						τ_y	$\tau_{\bar{y}}$
$\tau^{ca} < T$	0.02	0.74	1.28	0%	1%	0.02	0.02
$T < \tau^{ca} < \tau_c$	0.2	0.9	1.7	0%	1%	0.2	0.2
$\tau^{ca} = \tau_c$	0.7	1.04	3.76	3%	1%	0.7	0.7
$\tau^{ca} > \tau_c$	1	1.16	4.6	3.8%	1%	1	1

Discussion:

For the case of $\tau^{ca} < T$. The plant output is delayed with the delay value of the controller actuator $-y(k-\tau^{ca})$. The control input is delayed with the delay value of the controller actuator $-u(k-\tau^{ca})$. The DKF output is delayed with the delay values τ^{ca}. The systems is stable, under damped, and there is no overshoot. The system is stable, under damped, and there is no overshoot.

For the case of $T < \tau^{ca} < \tau_c$. The plant output is delayed with the delay value of the controller actuator $-y(k-\tau^{ca})$. The control input is delayed with the delay value of the controller actuator $-u(k-\tau^{ca})$. The DKF output is delayed with the delay values τ^{ca}. The systems is stable, under damped, and there is no overshoot. The system is stable, under damped, and there is no overshoot.

.

For the case of $\tau^{ca} = \tau_c$. The plant output is delayed with the delay value of the controller actuator $-y(k-\tau^{ca})$. The control input is delayed with the delay value of the controller

actuator $-u(k-\tau^{ca})$. The DKF output is delayed with the delay values τ^{ca}. The systems is stable, under damped, and there is no overshoot.

For the case of $\tau^{ca} > \tau_c$. The behavior of the system is similar to the previous cases. The delay between the controller and the actuator appears in all considered trajectories, however the developed control method proves to be robust and stable and can work with network induced time delays that are greater than the sampling period. The results are summarised in Figure (8.24).

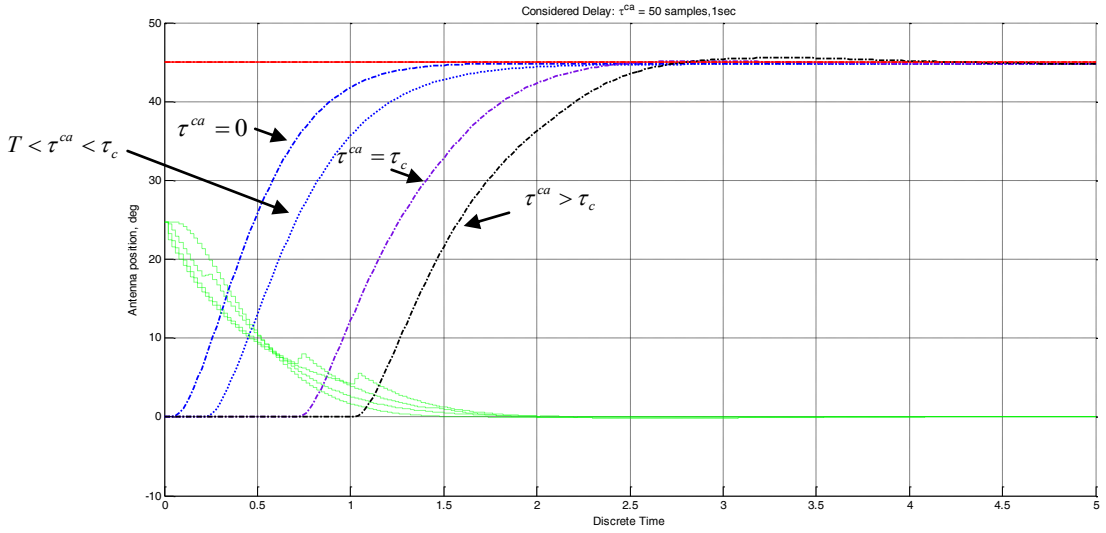


Figure 8.23: **Timing diagram analysis using developed method and considering constant NITD (τ^{ca})**

In Figure (8.24) the four delay consideration are shown, where the values of the delays in the NCS are selected to be:

- delay is less than the sampling period, $\tau^{ca} = 0$,
- delay is greater than the sampling period, $0 < \tau^{ca} < \tau_c$
- delay is equal to the critical delay (τ_c), $\tau^{ca} = \tau_c$.
- delay is greater than the critical delay, $\tau^{ca} > \tau_c$

In all the four cases the controller proves to be robust and there is no overshoot. The plant output is delayed by the same delay value at the beginning but it does not become unstable and there are no oscillations observed. The control input is NOT delayed. The observations are considering the time delay between the controller and the actuator only.

Simulation procedure 7:

This simulation procedure is performed with the same ***Plant, Predictive Controller*** and ***DKF*** and ***constant network induced time delays*** are introduced. In this simulation the delays are considered between **the controller and the actuator** and between **the sensor and the controller.** The purpose of this procedure is to illustrate the influence of the network induced time delay in the case when the controller is designed to cope with the delays. The controller designed is the same as the one in Simulation 6. The Simulink block diagram for the simulation is shown in Figure (8.20). The Matlab file (Exp_seven.m) with data for the simulation can be seen in Appendix (8.6).

Response results:

The system response results are for the case when both time delays are present. The outputs of the plant, controller, DKF and the pulse generator are shown in Figure (8.25) for the four considered cases of delays, Table (8.9).

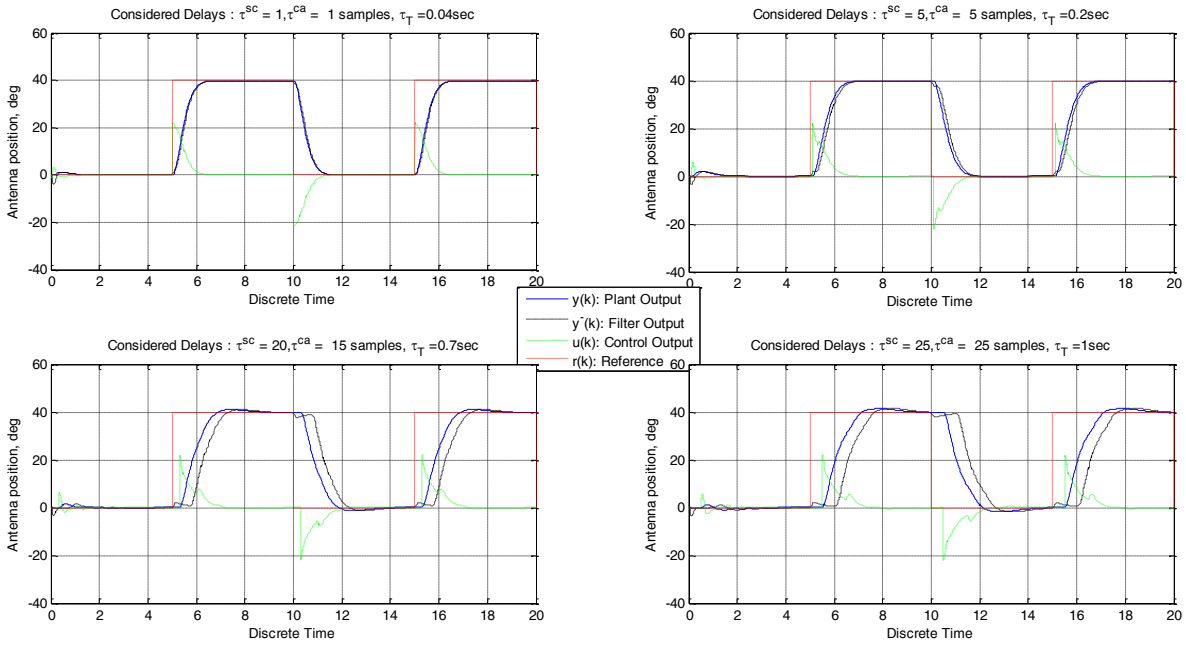


Figure 8.24: **Outputs of the Plant, Controller, DKF and Pulse generator - constant NITD (τ^{sc}, τ^{ca})**

Figure (8.9) shows a timing diagram zooming at the first fifty samples of the step response results graphs. This level of zooming allows for sample by sample analysis which gives a better understanding of the system performance in terms of:

- the introduced time delays
- the delay influence on the measurement output,
- the delay influence on DKF measurement output and

- control input after the introduction of the delay between the controller and actuator.

Note that the noise must be switched off in order to get a clear view of the delays. The delays are extracted and populated in the same manner for all the subsequent experiments.

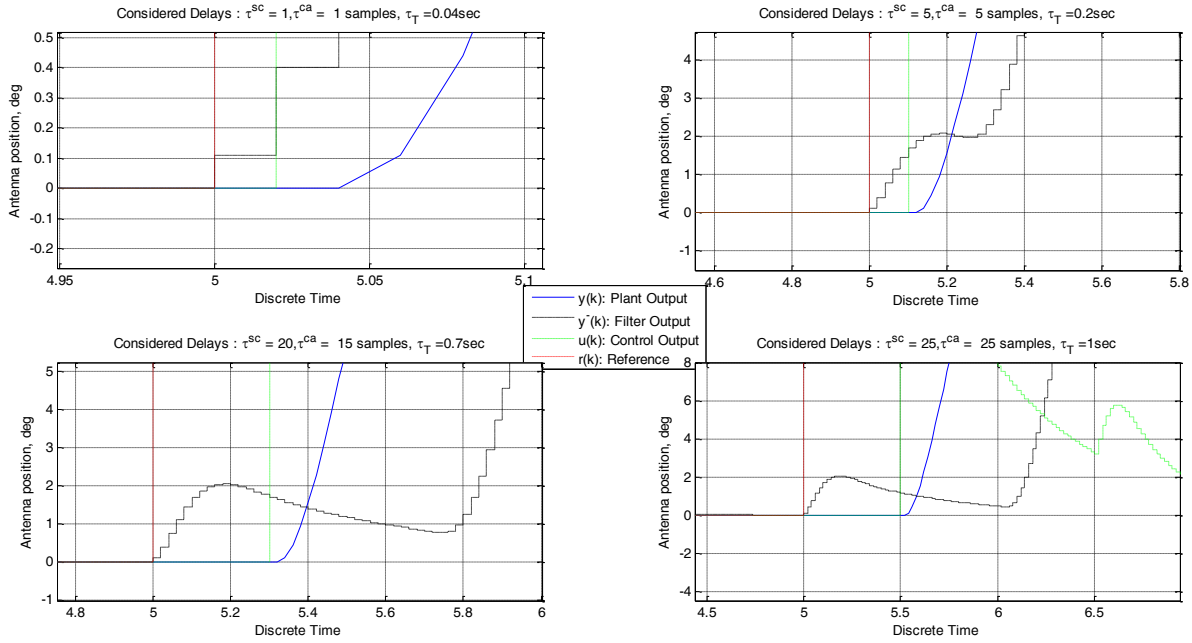


Figure 8.25: Timing diagram both delays using developed method (τ^{sc}, τ^{ca})

Performance results:

The characteristics of the transition behaviour are shown in Table (8.9).

Table 8.9: Transition behavior characteristics. Measured delays in the plant output, controller input and DKF output - both delays.

Delay Consideration		Performance Measure				Measured Delays	
Criteria	Injected Delay Values	Rise Time	Settling Time	Percentage Overshoot	Steady State Error	τ_y	τ_u
PLANT OUTPUT VALUES							
$\tau = 0.04$	0.04	0.74	1.28	0%	1%	0.04	0.04
$T < \tau < \tau_c$	0.2	0.9	1.7	0%	1%	0.2	0.2
$\tau = \tau_c$	0.7	1.04	3.76	3%	1%	0.7	0.7
$\tau > \tau_c$	1	1.16	4.6	3.8%	1%	1	1
DKF OUTPUT VALUES						τ_y	$\tau_{\bar{y}}$
$\tau = 0.04$	0.04	0.74	1.28	0%	1%	0.04	0.04

$T < \tau < \tau_c$	0.2	0.9	1.7	0%	1%	0.2	0.2
$\tau = \tau_c$	0.7	1.04	3.76	3%	1%	0.7	0.7
$\tau^{ca} > \tau_c$	1	1.16	4.6	3.8%	1%	1	1

Discussion:

For the case of $T < \tau_T < \tau_c$. The plant output is delayed with the delay value of the controller actuator - $y(k - \tau^{ca})$. The control input is delayed with the delay value of the controller actuator - $u(k - \tau^{ca})$. The DKF output is delayed with the delay values $\tau_T = \tau^{ca} + \tau^{sc}$. At the environment of the initial conditions and during the period of τ_T the filter is working without delay but at the end of τ_T period and during the rise time the sensor controller delay makes it impossible for the filter to follow the measurement output. The delay between the plant output and the filter output is equal to τ^{sc}. The systems is stable, under damped, and there is no overshoot.

For $\tau_T = \tau_c$. The control output delays with delay equal to τ^{ca}. The plant output delays with delay equal to τ^{ca}. The filter output delays with delay equal to $\tau_T = \tau^{ca} + \tau^{sc}$. In the environment of the initial conditions and for the period of τ^{sc} the filter output is faster than the plant output, but after this period it delays from the plant output with delay equal to τ^{sc}. The systems is stable, under damped, and there is no overshoot.

For $\tau_T > \tau_c$. The plant output and its control input delays with τ^{ca}. The filter output delays with delay equal to τ_T. In this case the filter output has no period where its values are ahead the values of the plant output. This means that the plant always works with delayed values of the filter estimates and this causes instability of the system. The developed controller method compensate for this by using the shifting method and as such the systems is stable, undamped, and there is no overshoot.

8.9 Conclusion

The following conclusion is drawn from the investigation of the influence of network induced time delays using the controller that is **not** designed to compensate for the network induced time delay:

- It has been observed that at the summing point just before the controller actuator delay the control is not delayed.
- The plant output is delayed with the delay value of the controller actuator - $y(k-\tau^{ca})$. The control input is delayed with the delay value of the controller actuator - $u(k-\tau^{ca})$. The DKF output is delayed with the delay values $\tau_T = \tau^{ca} + \tau^{sc}$. At the environment of the initial conditions and during the period of τ_T the filter is working without delay but at the end of τ_T period and during the rise time the sensor controller delay makes it impossible for the filter to follow the measurement output. The delay between the plant output and the filter output is equal to τ^{sc}.
- The plant always works with delayed values of the filter estimates and this causes instability of the system even thought the control and the plant output are not delayed by the delay between the sensor and the controller.
- The latter implies that the two delays have different influence on the measurement output. The delay between the controller and the actuator causes the delay and instability to the measurement output and the delay between the sensor and the controller causes the instability and does not affect the delay of the measurement output as it might have been implied in a lot of papers, (Nilsson, 1998), (Heemels, 2009), (Zhang and Yu, 2008), Cloosterman et al (2010).
- The DKF output depends on the τ^{ca} delay. This delay appears in the filter output and it influences it tracking capabilities. The total delay of the DKF is equal to $\tau^{ca} + \tau^{sc}$.
- The results obtained show that the time delay between the controller and the actuator and between sensor and the controller have to be both considered when a robust controller or filter have to be designed.

The following conclusion is drawn from the investigation of the influence of the network induced time delays using the robust controller designed to compensate for the network induced time delay:

- The developed control method proves to be robust and stable and can work with network induced time delays that are greater than the sampling period, and equal or greater than the critical time delay.

The implementation of the thesis results is described in Chapter 9 where description of the software tool LabNS2 is done. Its use for simulation, emulation and real-time control is discussed. Real-time implementation is done for the prototype of a dish antenna driven by two stepper motors.

CHAPTER NINE
EMULATION AND IMPLEMENTATION OF THE DEVELOPED METHODS

9.1 Introduction

The complexity of Networked Control System NCS makes it a necessity to develop simulation and emulation toolset that can be used during the design of NCS (Anderson et al, 2005). The toolset is used to validate and verify the designed controller methods as used in NCS. Different types of toolsets have been developed by control engineers as discussed in Chapter 2. During the NCS implementation, the network induced delays and packet loss are considered between the controller and the actuator and between the sensor and the controller.

Two general configurations of NCS are the direct structure shown in Figure (9.1) and the hierarchical structure shown in Figure (9.2). In the direct structure the controller is distributed and the control signal and sensor measurements are sent and received via the network. In the hierarchical structure there are two controllers meaning two control loops, of the main controller and of the remote controller. In this structure only the reference signal and sensor measurements are sent and received via the network. The developed LabNS2 supports simulation and emulation for both hierarchical structure and direct structure of the networked control systems.

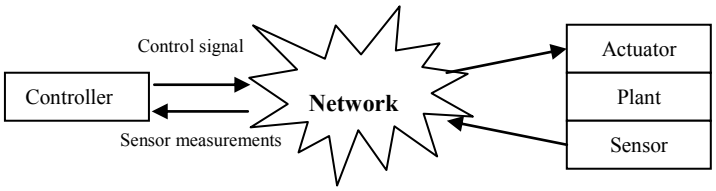


Figure 9.1: Network control system in a direct structure
Adapted from (Tipsuwan and Chow, 2003) Network part is implemented through NS-2

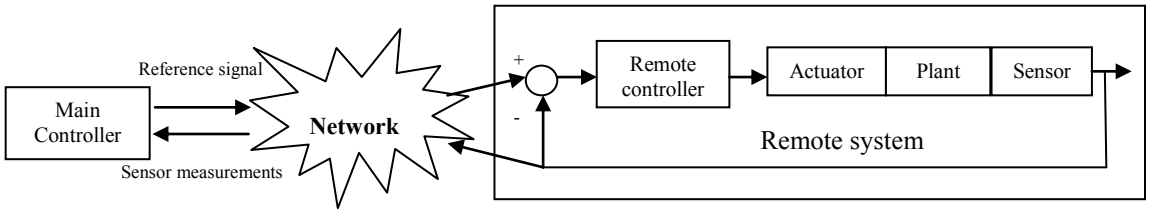


Figure 9.2: Network control system in a hierarchical structure
Adapted from (Tipsuwan and Chow, 2003) Network part is implemented through NS-2. Main controller is the same as reference computer referred to in the architecture

Part I of this chapter describes the architecture and functionality of the developed LabNS2 toolset as used to investigate the influence of networked induced time delays and packet loss in a typical NCS. The software tools used to develop the toolset are described and discussed in section 9.2.1. The design on LabNS2 and the developed LabVIEW subvi's for LabNS2 are described in section 9.2.4. A typical implementation case and results from LabNS2 is shown in section 9.2.5

Part II of this chapter describes the real-time implementation of the developed method using the software modules developed in LabNS2 and a DMX-ETH-17 integrated stepper motor. The hardware environment used for the implementation of the developed methods is described in Section 9.3.2. The DMX-ETH-17 integrated stepper motor, Ethernet Hub, and Controller Computer are described in section 9.3.2. as the main hardware used for the implementation. The interface between the Controller computer and DMX-ETH-17 hardware is described in section 9.3.3. A typical implementation environment and results of the NCS are shown in section 9.3.4.

9.2 Part I: LabNS2 (Integration of LabVIEW^{TM} and NS2)

LabNS2 is derived from LabVIEW^{TM} and Network Simulator version 2 (NS-2) which are the main tools used in developing the toolset. The toolset uses Windows environment as an operating system. NS-2 is developed for UNIX environment. In order to emulate UNIX environment within the Windows environment, Cygwin is used. (http://www.cygwin.com/).

9.2.1 Existing software tools used to develop LabNS2

LabVIEW, NS2 and Cygwin are the main software tools used during the development of LabNS2. The following sections described the software tools used to develop LabNS2.

9.2.1.1 LabVIEWTM

LabVIEW^{TM} is a graphical programming language where icons are dragged and dropped to define functionality (http://www.ni.com). LabVIEW functions are called Virtual Instruments (vi). LabVIEW^{TM} is used in thesis to develop software modules for the implementation of the Plant, Controller and the Reference trajectory generating computer (Supervisory computer). The software modules communicate with each other by using Sockets. LabNS2 is implemented and tested on LabVIEW^{TM} Version 7.1 with Matlab runtime engine Verison 6.0.0.88 Release 12.

9.2.1.2 NS2

NS2 is a discrete event network simulator that is developed in C++ and object oriented Tool Command Language (OTcl) to manage control of data path implementations (Shin and Kim, 2009). The simulator supports a class compiled hierarchy in C++ and a corresponding interpreted hierarchy within the OTcl. C++ is used for protocol implementation and for all the implementation where every packet journey has to be processed. OTcl is used for setup and configuration. Tcl scripts are used to access the Otcl using Tcl syntax, e.g. assigning of variables (*set a 3*), procedures (*proc sum {a b}{expr $a +$b}* etc.

NS2 supports the following traffic sources File Transfer Protocol (FTP), Constant Bit Rate (CBR) and Real-Time Ethernet data (Shin and Kim, 2009).

NS-2 has a built in real-time scheduler which makes it possible to inject live network data to and from the network to allow for a real-time investigation of network dynamics.

NS2 supports a trace queuing method which is an offline method of measuring packet loss and time delay (Fall and Varadhan, 2008). The output trace file format is shown in Table 9.1.

Table 9.1: NS2 Trace file format

Event	Abbreviation	Type	Value
Normal Event	r: Receive d: Drop e: Error +: Enqueue -: Dequeue	double	Time
		int	(Link-layer) Source Node
		int	(Link-layer) Destination Node
		string	Packet Name
		int	Packet Size
		string	Flags
		int	Flow ID
		int	(Network-layer) Source Address
		int	Source Port
		int	(Network-layer) Destination Address
		int	Destination Port
		int	Sequence Number
		int	Unique Packet ID

The animation of NS2 results from the output simulation traces (trace file) and real-time packet traces is viewed using Nam. Nam is an animation tool based on Tcl/Tk. NS2 environment is summarized is Figure (9.3).

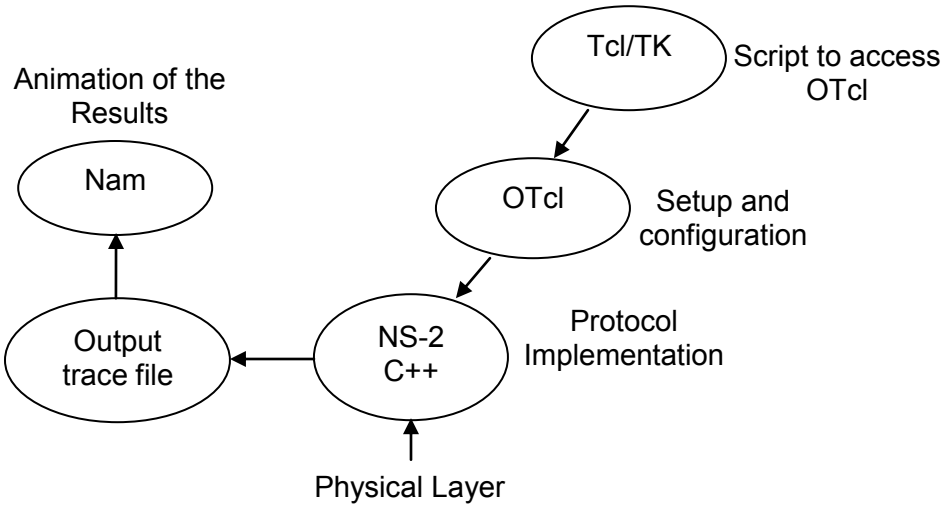


Figure 9.3: NS2 software environment

LabNS2 is implemented and tested on ns-release-2.31 software version of NS2. This does not mean the other upper versions of NS2 are not compatible with LabNS2. Table 9.2 gives a list of required software packages for ns-release-2.31. The packages are included in ns-allinone 2.31 software package.

Table 9.2: NS-2.31 required software packages (Adapted from ns-allinone-2.31 README file)

Package	Version
Tcl	Tcl release 8.4.
Tk	Tk release 8.4.14
Otcl	otcl release 1.13
TclCL	tclcl release 1.19
Ns	**ns release 2.31**
Nam	Nam release 1.13

9.2.1.3 Cygwin

Cygiwn is the UNIX emulation environment for Windows. The Cgwin dynamic link library (dll) used for LabNS2 is a version 1.7.17.-1. This dll acts as an UNIX API layer and provides UNIX API functionality.

9.2.2 Design of LabNS2

LabVIEW™ communicates with NS2 using sockets as shown in Figure (9.4). Socket is a software endpoint that establishes bidirectional communication between a server program and one or more client programs. Sockets use standard protocol, like IP, UDP and TCP. A socket binds an IP (Internet Protocol) address and an UDP port.

The architecture of the toolset is based on a node approach where the plant, the controller, and the supervisory computer are considered as nodes of the closed loop control system as shown in Figure (9.4). LabVIEW™ is used to develop the software for each node. Each node has a built in Ethernet software module which allows for sending and receiving of Ethernet datagram using sockets.

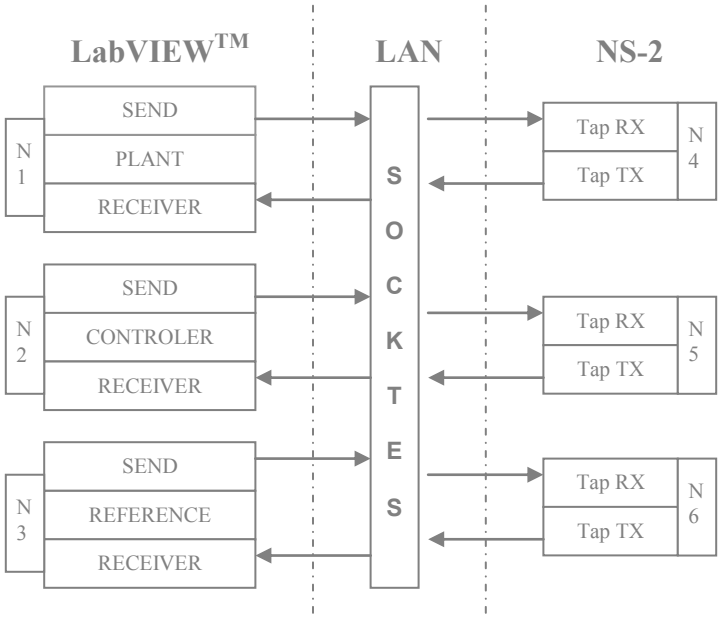


Figure 9.4: Emulation and Simulation Architecture of LabNS2
Toolset Architecture. Note that "N1" means node 1. Nodes may be implemented on a single or multiple computers. It is recommended that each node be implemented on a separate computer N1, N2 and N3 run Windows as an Operating System (OS) and LabVIEW™, and N4, N5 and N6 run Windows OS, Cygnus UNIX Emulator and NS-2. N4, N5 and N6 may be executed on Linux or Unix environment

To measure time delay and packet loss between the nodes, Tcl scripts are developed to capture live data from the physical layer using NS2 real-time scheduler which exposes the simulator to the live network traffic. The Tcl scripts are shown in Appendix D.

The Tcl scripts generate an output trace file which is further streamed by Gawk scripts. Gawk scripts are shown in Appendix E.

Note that the measuring scripts run at the same time as nodes, however, the Gawk scripts are executed based on the output files of the Tcl scripts. The connection between NS2 and LabVIEW™ is shown in Figure (9.5).

The ability for the toolset to measure live network traffic is due to the NS-2 emulation facility which exposes the simulator to the live network traffic. The emulator has built in network object which allows for simulator to receive live network traffic and to inject live network traffic into the network (Perbellini, 2005). The network object is based on a tap interface. A tap interface is a virtual Ethernet interface that looks like network hardware to

the Operating System (OS) (Yonan, 2003). Instead of writing bits to the Ethernet interface it writes the bits to a user space. Network object has an associated Tap Agent class. The tap agent sends and receives packets between network objects, see Figure (9.5). The tap agent handles the setting of the common header packet size field and field type. The simulator environment can now be exposed to the network and accessed as a node within the network. The network object allows access to the protocol layer e.g. link layer, raw IP, UDP and with particular access mode (read-only, write-only, or read-write). LabNS2 toolset uses this feature as the core for interfacing LabVIEW™ and NS-2 see Figure (9.5).

The modularity of LabNS2 makes it easy to implement direct structure and hierarchical structures of the networked control systems.

The installation procedure of Cygwin, NS-2, and LabNS2 are described in Appendix A.

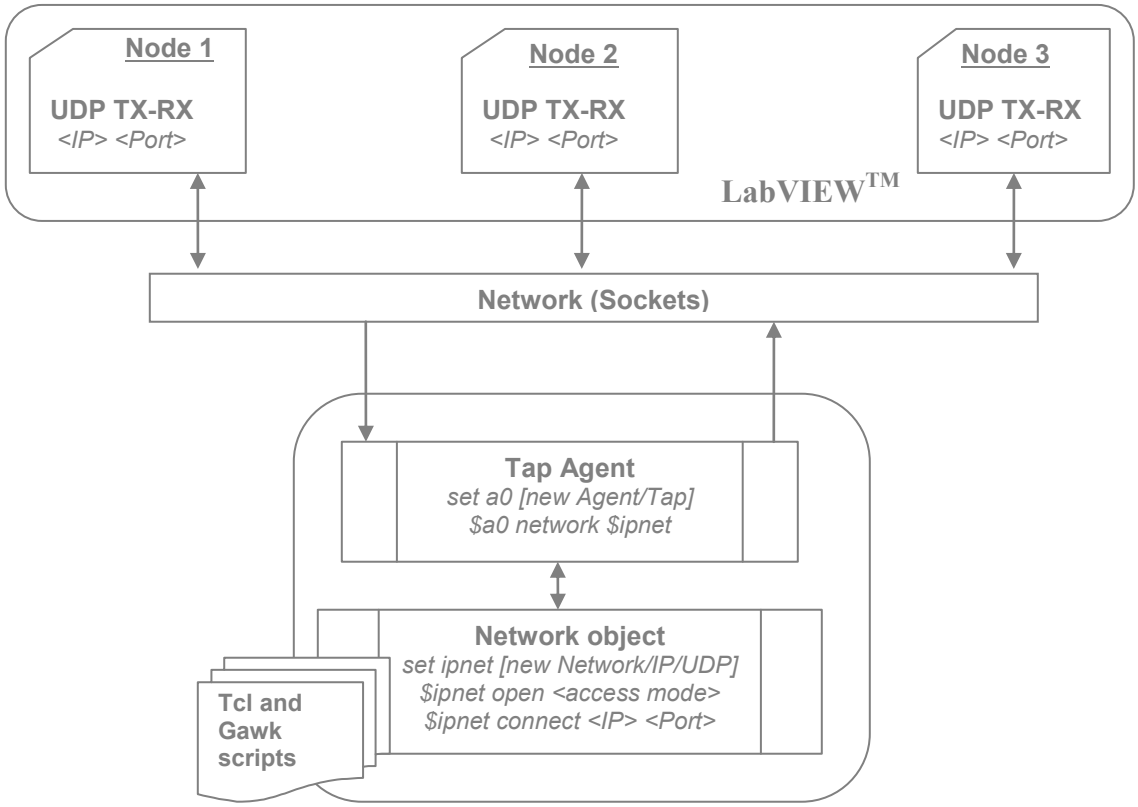


Figure 9.5: Communication structure and interface between LabVIEW™ and NS-2 to make LabNS2 NS-2 emulator block implements Node 4, 5 and 6.

9.2.3 Developed block library for LabNS2

LabVIEW™ subvi's for the plant, the controller, the reference generating computer and the socket parser are shown in Table 9.3.

The list of Tcl scripts for measuring time delay and packet loss are shown in Table 9.4. Gawk scripts used to calculate packet loss and time delay are listed in Table 9.4. The Gawk scripts work from off line data using the output trace file from the measuring Tcl scripts. In the thesis Tcl and Gawk scripts are adapted from Ke (2005). The scripts have been modified to accommodate live UDP data and are based on trace-queue to perform packet based analysis.

Table 9.3: LabVIEW™ block library subvis for LabNS2

Software modules for LabNS2			
Horizontal coordinate system			
Name	**Input parameters**	**Out parameters**	**Function**
lt2ut.vi	Local Time (String format)	Universal Time (Y:M:D:H:M:S)	Converts local time to Universal Time (UT)
Juliandate.vi	Universal Time (Y:M:D:H:M:S)	Julian date and Greenwich Sidereal Time (GST)	Converts UT to GST
hms2dd.vi	Hours, Minutes, Seconds	Decimal degrees (Double)	Converts from Hour Minutes Seconds to Decimal degrees
dms2deg.vi	Degrees, Minutes, Seconds	Decimal degrees (Double)	Converts from Degrees Minutes Seconds (DMS) to Decimal degrees
Azel.vi	Right Ascension, Declination, GST, Observer Longitude, Observer Latitudes	Azimuth and Altitude*	Converts from RA,DEC to Alt Az using Ephemeris (Almanac) equations
Network sub vi's			
UDP_TX_aN.vi	Packet Data (Az,El) String	Packet Data	Receives packet data and send it via network. Where **N** in the name is the antenna number.
UDP_parser.vi	UDP reads output (Az, El) String	Control Value (Az, El)(Double)	Parses data from UDP read.vi and sends it to the controller.
Controller sub vi's			
Simple_PID.vi	K_p, K_i, K_d	Control action (Double)	Implements the PID controller.
State_space_con.vi	Matrices A, B, C and H	Control Action (Double)	Implements state space controller for both Proportional and Proportional Integral controllers.
Plant sub vi's			
Dish_N_tf.vi	Numerator and Denominator	Desired set point (Double)	Implements the plant model in transfer function form. Where N is the dish number. Note each dish has its own vi.
Dish_N_ss.vi	Matrices A, B, and C	Desired set point (Double)	Implements the plant model in state space form. Where N is the dish number. Note each dish has its own vi.
Reference Generating Trajectory Computer			

Reference_DM.vi	RA, DEC	Azimuth and Altitude	Generates trajectory for all the antenna nodes.

* The angles for position are in degrees.

Figure (9.6) shows the hierarchy of LabVIEW™ subvis that are developed for LabNS2.

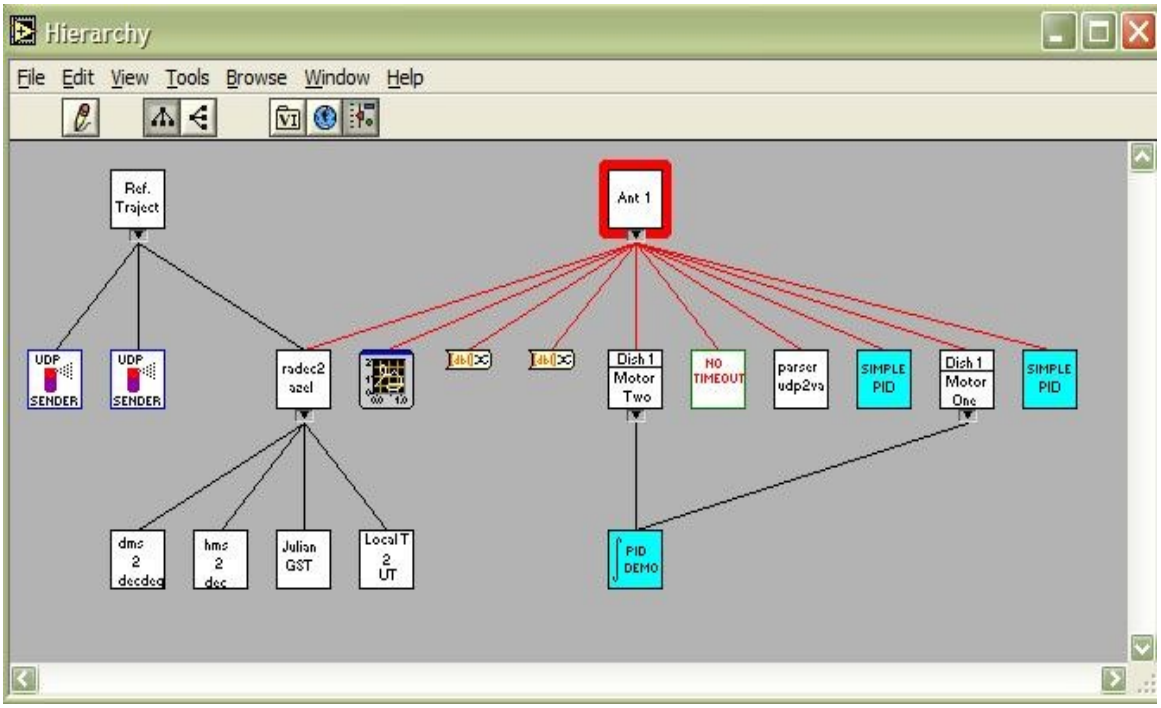


Figure 9.6: LabVIEW™ software hierarchy
LabVIEW™ block library subvis for LabNS2 as described in Table 9.3

Table 9.4: NS2 Tcl and Gawk developed scripts for LabNS2

Software modules for LabNS2			
Scripts	**Input**	**Out**	**Function**
measure-tdpl-rc-aN.tcl <cbw> <ex_time>	Channel band width (cbw) and execution time (ex-time)	Nam and trace files	Measure time delay and packet loss between the reference computer and the controller
measure-tdpl-ca-aN.tcl <cbw> <ex_time>	Channel band width (cbw) and execution time (ex-time)	Nam and trace files	Measure time delay and packet loss between the controller and actuator.
measure-tdpl-sc-aN.tcl <cbw> <ex_time>	Channel band width (cbw) and execution time (ex-time)	Nam and trace files	Measure time delay and packet loss between the sensor and the controller.
calculate_packet_loss.awk	NS-2 Trace file	Packet sent, lost and received	Calculates packet loss from trace file.
Calculate-time_delay.awk	NS-2 Trace file	Excel file - Time delays	Calculates time delays from trace file.

9.2.4 IP address allocation summary

The IP addresses and UDP port allocation for a hierarchical network control system are given in Table 9.5.

Table 9.5: IP Addresses and UDP port allocation

Description	LabVIEW/NS2	Internal Tap IP address	Port IN (RX)	Remote PC IP address	Port OUT (TX)
RC - UDP TX **Antenna 1**	LabVIEW™	192.168.1.2	4490	192.168.1.4 *(127.0.01)*	4420
RC - UDP TX **Antenna 2**	LabVIEW™	N/A	4490	192.168.1.3 *(127.0.0.1)*	4400
ns-measure-rc-a1	NS-2	224.1.127.2	4420	192.168.1.4 *(127.0.0.1)*	4430
Antenna 1 (Controller + Plant)	LabVIEW™	192.168.1.4 *(127.0.0.1)*	4430	N/A	N/A
ns-measure-rc-a2	NS-2	224.1.127.2	4400	192.168.1.5 *(127.0.0.1)*	4410
Antenna 2 (Controller + Plant)	LabVIEW™	192.168.1.5 *(127.0.0.1)*	4410	n/a	n/a

This IP configuration is for the hierarchal network control structure shown in Figure (9.2).

The glue between LabNS2 code and NS-2 is through Ethernet interface. NS-2 real-time scheduler is used to interface with real live network and allows for sending and receiving of raw UDP data back and forth thereby creating an environment for measurement of network induced time delay and packet loss as discussed in the previous sections. The LabVIEW™ subvi that implements the sockets is shown in Figure (9.7).

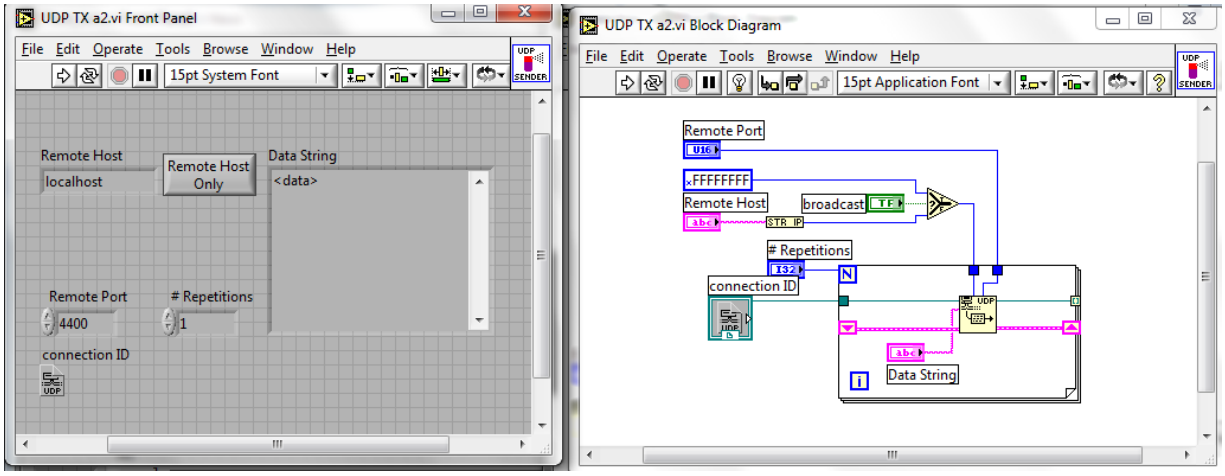


Figure 9.7: Front Panel and Block diagram subvi of sockets implementation

9.2.5 Implementation of the LabNS2

To illustrate how the emulator operates the control structure of the typical Radio Telescope dish as shown in Figure (4.1) is considered. This control structure has four

sub-elements that is; Coordinate Transformer, Model of the DC Motor (Plant) and the Controller. The plant model is derived in Chapter 4. Typically the reference trajectory is considered to be the object in the sky identified with the Right Ascension (RA) and Declination (DEC) coordinates. A coordinate transformer is required to convert from RA, DEC to Altitude and Azimuth (Alt, Az) coordinates as described in Chapter 3. The Alt, Az becomes the reference that is sent to the respective controllers. The controller is typically PID or Pole Placement state space one as described in Chapter 5 and 8. The implementation of the NCS for the case of a Radio Telescope or Satellite system is shown in Figure (9.8). Computer 1 of Figure (9.8) implements the reference trajectory software and the controller software developed in LabVIEW™. The controller and the plant are connected via the network using NS2 to measure the time delay and packet loss between the controller and actuator and between the sensor and the controller as shown in Figure (9.8). Computer 3 in Figure (9.8) executes the Tcl scripts to measure time delay and packet loss. Computer 2 of Figure (9.8) executes the plant model, sends and receives delayed control action values, and sends sensor measurement values to the NS2 computer that further sends the data to the controller's computer.

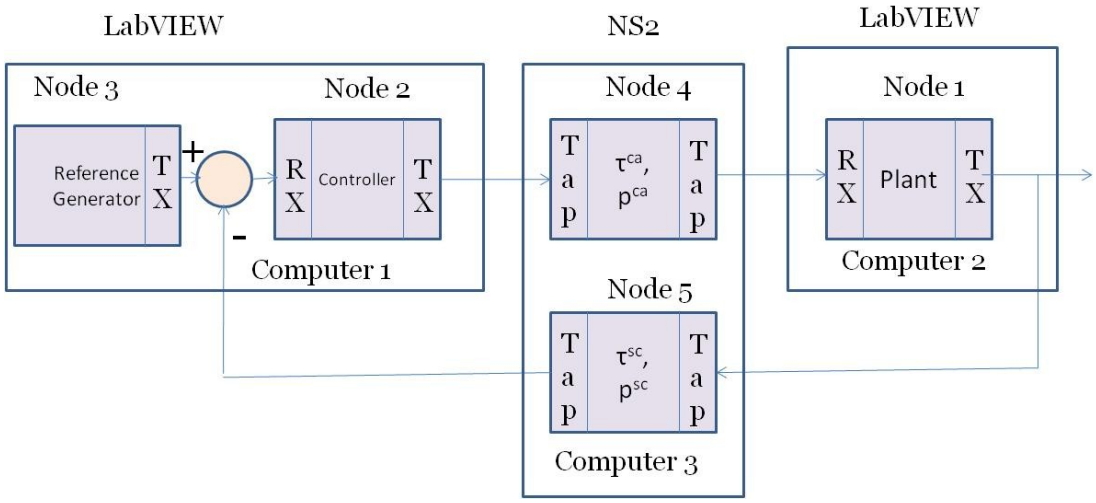


Figure 9.8: Emulation block diagram

Computer 1 of Figure (9.8) generates a control signal and sends the signal to Computer 2 via Computer 3. Computer 3 runs NS2 Tcl scripts that measure in real-time the network induced time delay and packet loss. The measured data is stored in the NS2 trace file. The trace file is processed using the Gawk scripts to calculate network induced time delays and packet loss.

9.2.6 How to run LabNS2

To run LabNS2 start the following vi in LabVIEW™:

- ***Dish_N_tf.vi (Antenna1.vi) – Plant and controller***
- ***Trajectory_rc_DM.vi***

To run LabNS2 start the following scripts from the home directory in Cygwin:

- cd ***tcl-scripts-cput/***
- nse ***measure-tdpl-rc-a1.tcl <cbw> <ex_time>*** **%**Script to measure time delay and packet loss

LabNS2 has four software modules that have to be executed to emulate the NCS environment. The modules are:

- Reference generating trajectory subvi is implemented in LabVIEW as shown in Figure (9.9). This module uses the software libraries described in Table 9.3 and it generates the desired positions for altitude and azimuth mounts.

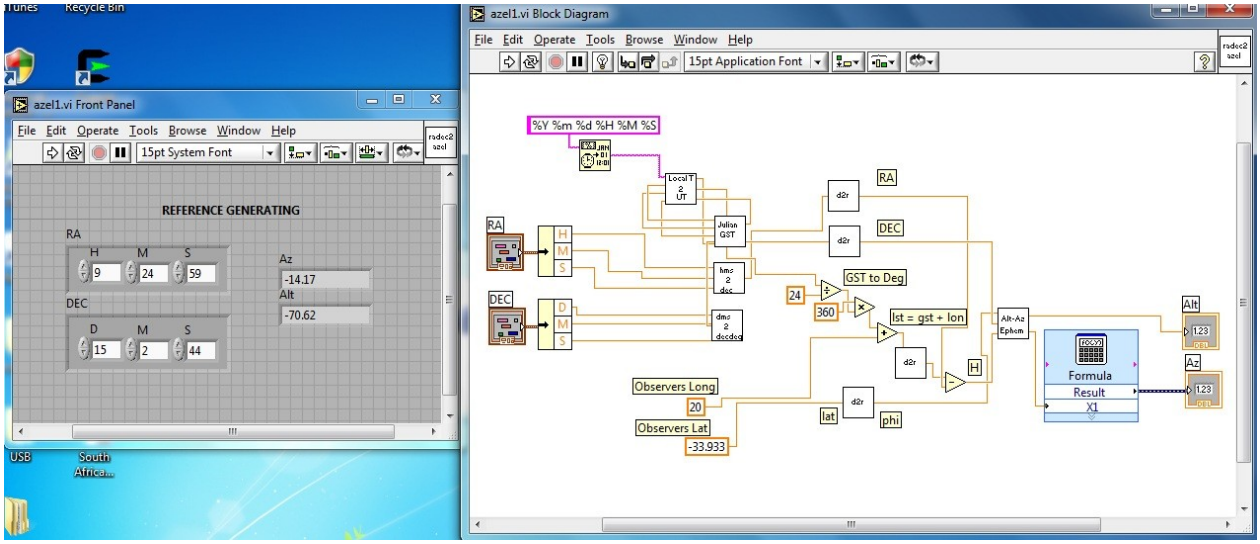


Figure 9.9: Front Panel and Block diagram of the Reference trajectory generators subvi

- Controller with reference generating subvi and Plant subvi are implemented in LabVIEW as shown in Figure (9.10) and (9.11). These modules use the software libraries described in Table 9.3. The Controller subvi sends the control signal via the network through NS2 as shown in Figure (9.8). It realises the controller and displays actual vs desired positions for both altitude and azimuth as shown in Figure (9.12).

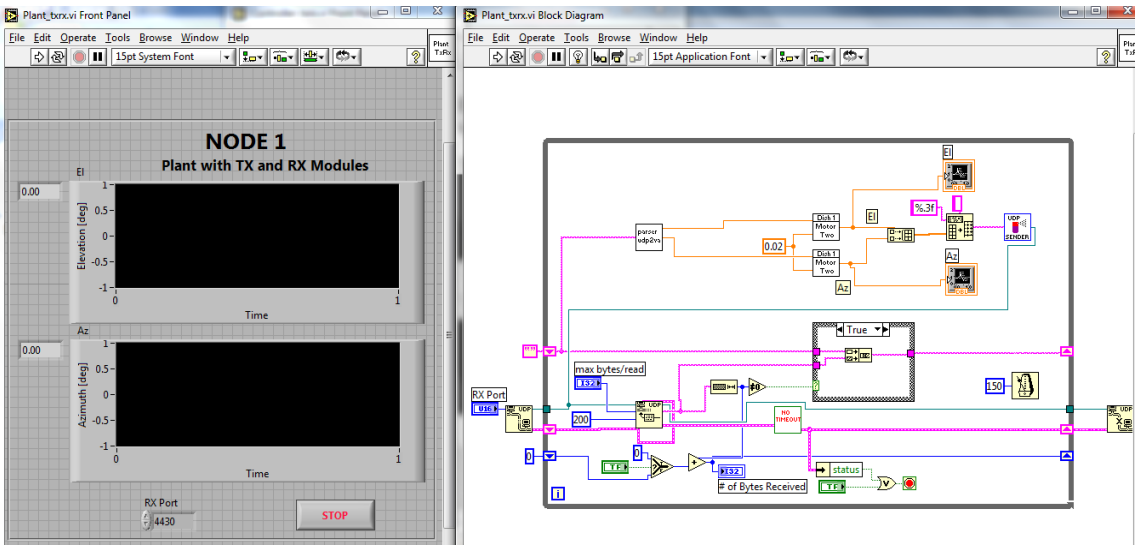


Figure 9.10: Plant vi with Transmit and Receive modules

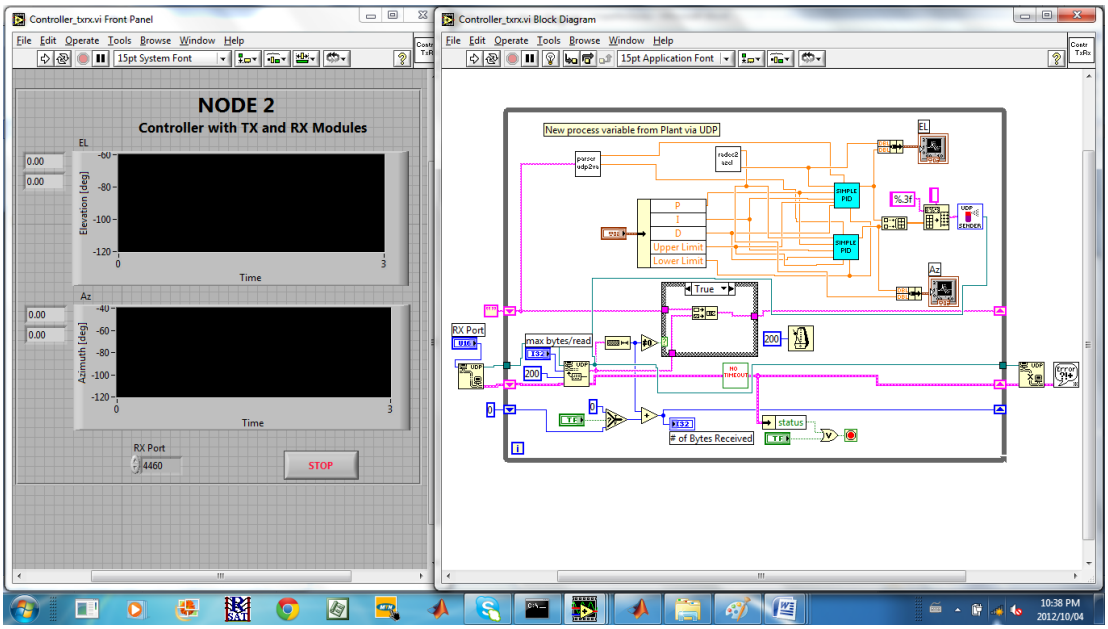


Figure 9.11: Controller vi with Transmit and Receive modules

- Packet loss and time delays script implemented using Tcl scripts. The measuring script runs for pre defined time. The execution time, bandwidth and Ethernet drop tail are configured in the measuring scripts. When the execution time expires a trace file is written in the home directory where the measuring script is executed.

The snapshot of the results of the Plant output connected through NS2 and via the communication network is shown in Figure (9.12). In the figure the two lines indicate the reference trajectory and the measurement output. If one of the nodes is disconnected the error grows and the measurement output cannot track the reference trajectory as shown

in Figure (9.12). Note that as soon as the network is restored the measurement output tracks the reference trajectory

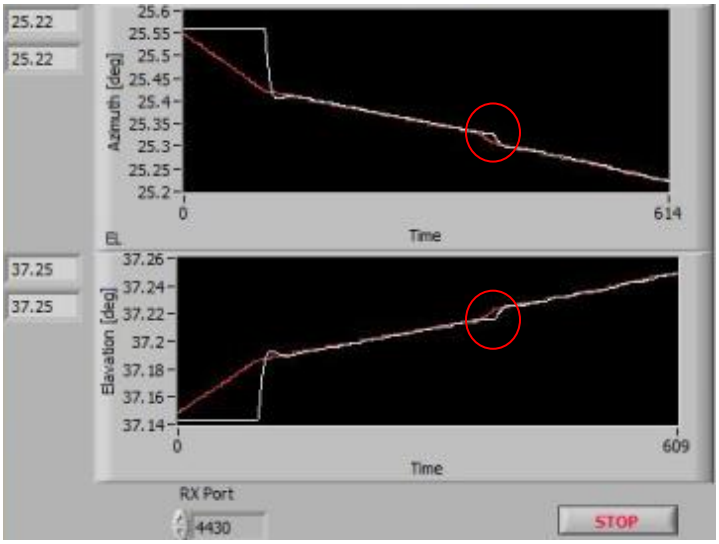


Figure 9.12: **Snapshoot of the results**

To process the trace file, execute the Gawk script by using these commands:

- gawk –f <***calculate_packet_loss.awk***> <***tracefile.tr***>
- gawk –f <***calculate_time_delay.awk***> <***tracefile.tr >*** > ***<excelfile.xls>***

Typical results that are received after executing the Gawk scripts are shown in Figure (9.13). The results of the packet loss counter are shown on the command prompt see Figure (9.13) where 250 packets were sent and 0 packets were lost and 249 packets were received. The second command above writes the output file into a Microsoft Excel file which may further be analysed.

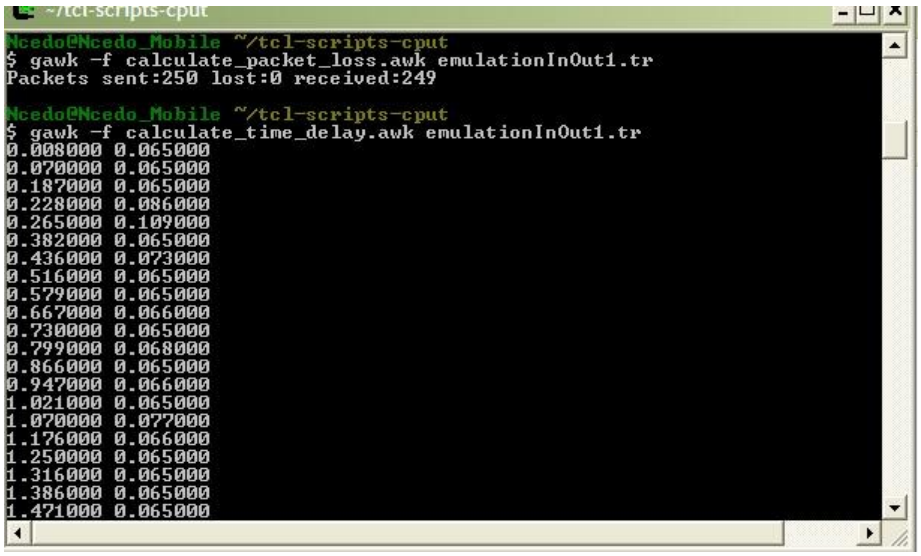


Figure 9.13: **LabNS2 Emulation results. Gawk scripts output**

Figure (9.14) shows how LabNS2 is used to investigate the relationship between channel bandwidth and network induced time delay. The results show that time delay drops as the channel bandwidth increases. In this emulation LabNS2 is used to measure time delay between reference computer and controller (τ^{rc}).

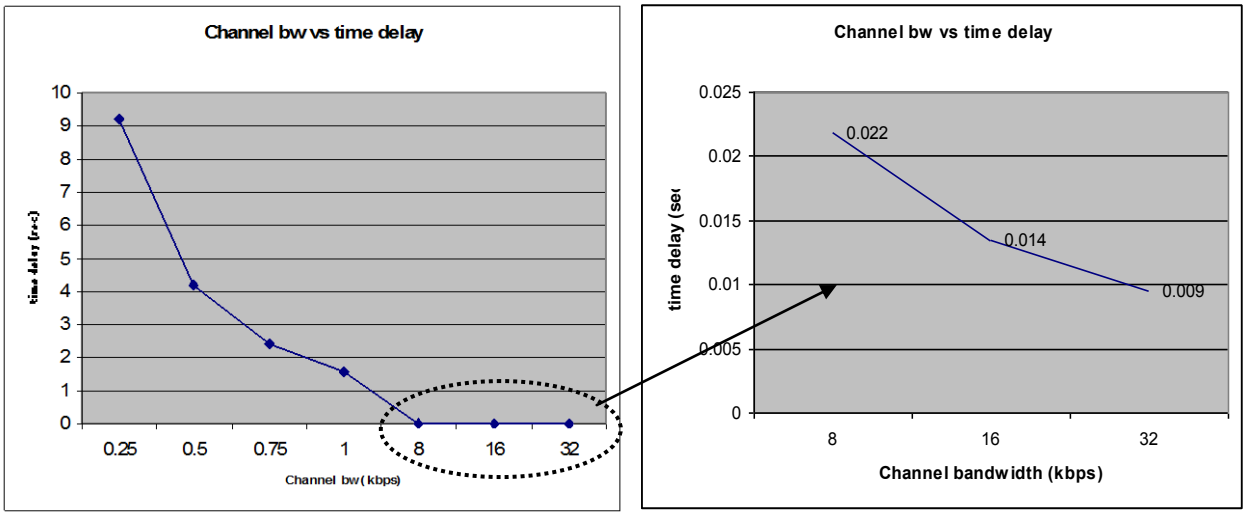


Figure 9.14: **LabNS2 Emulation results. Processing results of Gawk scripts**
The graph on the right is the zoomed portion of channel bandwidth 8, 16 and 32kbps from the graph on the right

9.3 Part II: Implementation of the developed method

Part II of this chapter presents hardware and software environments used for the implementation of a Real Time Network Control System using the Controller Computer, dish antenna implemented using DMX-ETH-17 integrated stepper motors and LabVIEW™ as driver software. Hardware and software parts of the networked control systems are described and discussed. LabVIEW™ code is presented.

The developed software programmes have two modes. The first mode is an open loop controller mode where the motor position is commanded using the desired number of steps of the motor. Based on the steps to the desired position, the motor goes to the target position without any feedback from the sensor.

The second mode is the closed loop controller mode where the motor position is commanded using the desired target position. Based on the encoder feedback from the motor the error is calculated and the new value of the control signal is sent to the motor until the error is reduced and the desired position is achieved. The next sections describe the hardware and software environments used to implement the developed software.

9.3.1 Hardware environment

The hardware environment used for the implementation of the NCS consists of the Controller Computer, Ethernet Hub and DMX-ETH-17 stepper motor. The hardware integration of the network control system is shown in Figure (9.15).

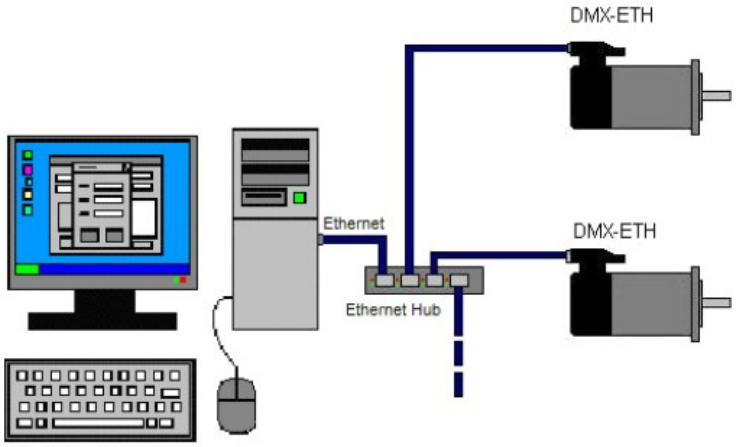


Figure 9.15: **Network based configuration**
Adapted from Arcus Technology, 2008

9.3.1.1 Stepper motor

Stepper Motors have been in existence for the past decades and became popular in the early 1960's (www.ti.com/ww/en/motor_drive_and_control_solutions/). According to www.tim.com they were designed as low costs alternative to servo motors for use in applications that require position control.

As can be inferred from the name, a stepper motor is an electromechanical device that is driven by digital pulses rather than voltage to achieve mechanical movement through conversion of the electrical pulses. The main advantage of the stepper motor is that there is no need for feedback to maintain control of position which makes it an open-loop system. The drawbacks of the stepper motor are:

- lack of smooth motion, particularly at slow speeds due to predefined stepping angle;
- significant loss of torque at high speeds;
- positioning errors in the presence of static friction and external torque.
- step skipping (when the rotor does not move itself to the desired position in time before the next step starts)

Some of the drawbacks can be eliminated by using a *microstepper* motor system. In the thesis the considered motor is using *microstepper* system where the positioning step of 1.8^0 is divided into **500** parts, which increases the positioning accuracy (in the absence

of external torque) to **0.0036^0** positioning step and also increases motion smoothness. The difference between conventional full-stepping and microstepping is shown by the phase diagram in Figure (9.16).

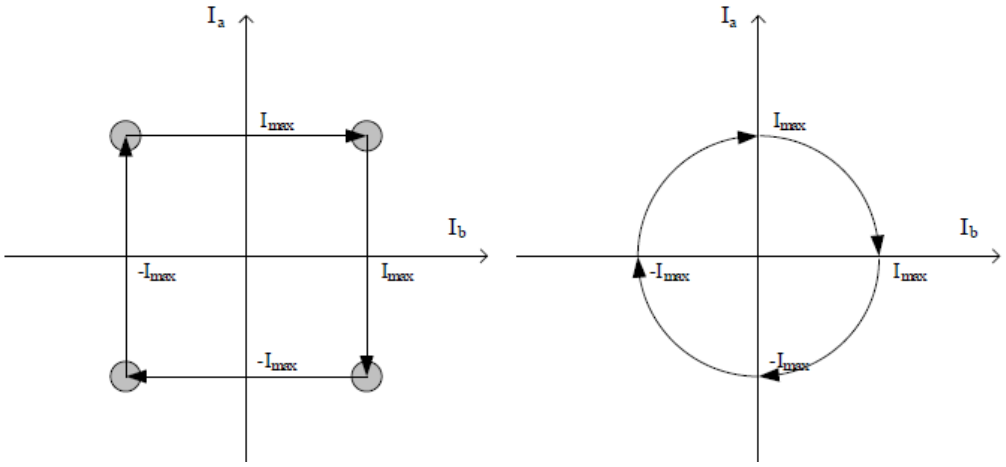


Figure 9.16: **Conventional stepping motor (left) and microstepping motor (right) phase diagrams. adapted from www.1-core.com**

It is assumed that there are two stator poles. In the case of a conventional stepper motor a current in one of the windings is instantly switched to the opposite to make a step. In the case of the microstepping the winding currents are changed smoothly.

.

9.3.1.2 DMX-ETH-17 Integrated stepper motor

The dish antenna prototype is built using 2 X DMX-ETH-17 integrated stepper motors for the implementation of Altitude and Azimuth drives. DMX-ETH-17 is an integrated Stepper Motor, with Encoder, Driver, Controller and Ethernet communication capabilities developed by Arcus Techonology (Arcus Technology, 2008), as shown in Figure (9.17).

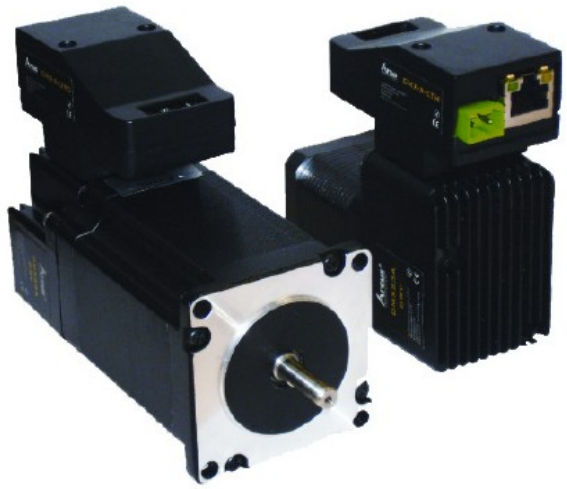


Figure 9.17: DMX-ETH-17

The summary of the features for DMX-ETH-17 are bulleted below, and the details are further found in the User manual see Arcus Technology, (2008):

- **10Mbps Ethernet communication**
  - ASCII
  - TCP/IP
- **Opto-isolated I/O**
  - 2 x inputs
  - 2 x outputs
  - +Limit/-Limit/Home inputs
- **Homing routines:**
  - Home input only (high speed)
  - Home input only (high speed + low speed)
  - Limit only
  - Z-index encoder channel only
  - Home input + Z index encoder channel
- **S-curve or trapezoidal acceleration profile control**
- **On-the-fly speed change**
- **1000 line incremental encoder (4000 counts/rev with 4x quadrature decoding)**
  - StepNLoop closed loop control (position verification)
- **Stepper driver**
  - 12-48 VDC
  - 3.0 Amp max current setting (peak current)
  - 2 to 500 micro-step setting
  - 1 MHz max pulse support

- **Stepper motor**
  - NEMA 17/23 motor sizes available in different stack sizes
  - 1.8° step angle

The antenna configuration of the two stepper motors is shown in Figure (9.18).

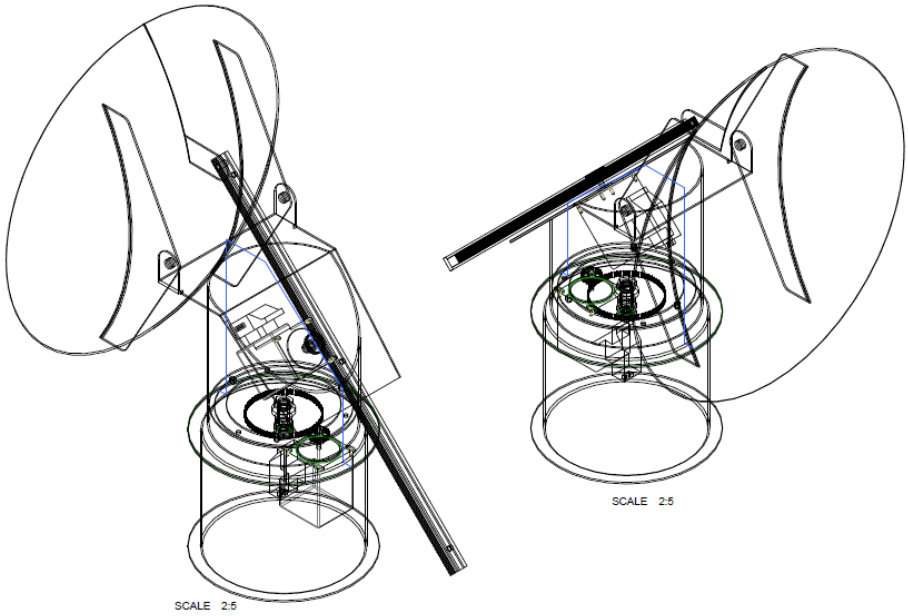


Figure 9.18: Antenna prototype

The antenna has three pieces; Dish, York and the Pedestal. The Dish is connected to the York through the gears and the pedestal is connected to the York through the Azimuth stepper motor as shown in Figure (9.18). The Altitude motor is connected through the gears to the Altitude arm

9.3.1.3 Ethernet Hub

A Gigabit Ethernet hub is used to connect the controller PC and the two motors. The switch has the following capabilities, it supports:

- Interface options of 10BaseT
- Performance of 10/100 Mbps
- 4 Ethernet ports
- Wireless connectivity.

9.3.1.4 Controller Computer

The computation time, Human Machine Interface (HMI) graphics and the complexity of the developed algorithms require the computer with the following minimum specifications:

- Intel Inside Core^{TM}i5 CPU @ 2.00Ghz
- 8GRAM

- 300GHD
- 32-bit or 64 bit Operating System
- Windows XP or 7.
- Ethernet port 10/100Mbps

The software is developed using Windows environment and using LabVIEW™ and the motor software.

9.3.2 Software environment

The same software environment used for the simulation and emulation as shown in Table 9.1 and 9.2 is used during the implementation of the developed method.

9.3.2.1 Description of the interface between Controller Computer and DMX-ETH-17

The communication between the Controller Computer and Antenna is based on the Ethernet communication protocol using standard sockets. The DMX-ETH-17 has by defaults its Socket Port set to 5001. The socket port settings can be changed using DMX-ETH-17 commands, see Arcus Technology, (2008). ASCII Protocol is used for sending and receiving commands between the Controller Computer and the DMX-ETH-17. The examples are:

For querying the x-axis polarity

Send: POX[NUL]

Reply: 7[NUL]

For jogging the x-motor in positive direction

Send: JX+[NUL]

Reply: OK[NUL]

The rest of the commands are listed in the Arcus Technology (2008) manual.

Using these commands the software is developed in LabVIEW to implement the flowchart shown in Figure (9.19).

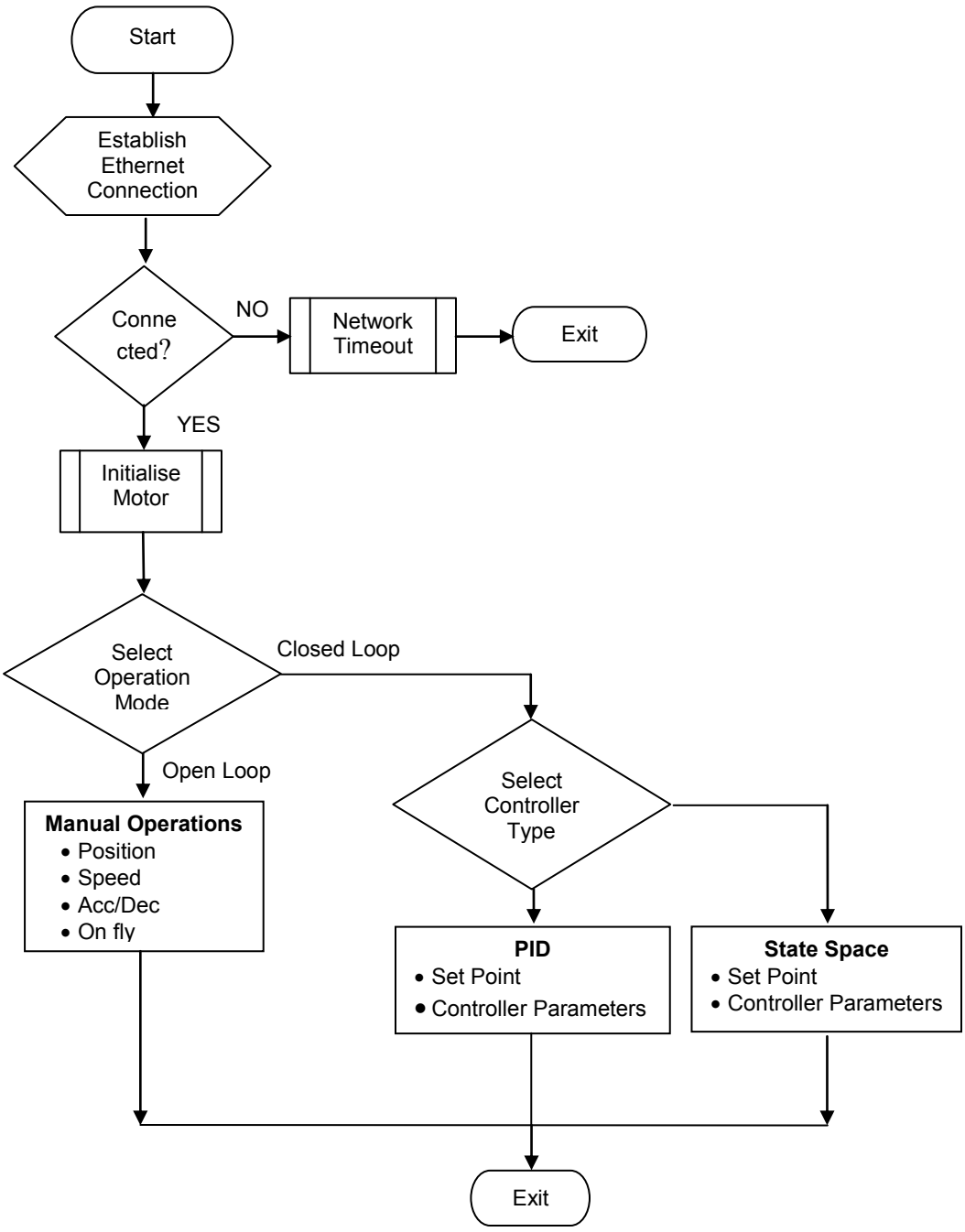


Figure 9.19: **Implementation Software Flowchart**

The method for the development of the software for implementation of the flowchart on Figure (9.19) is based on the event structures as shown in Figure (9.20).

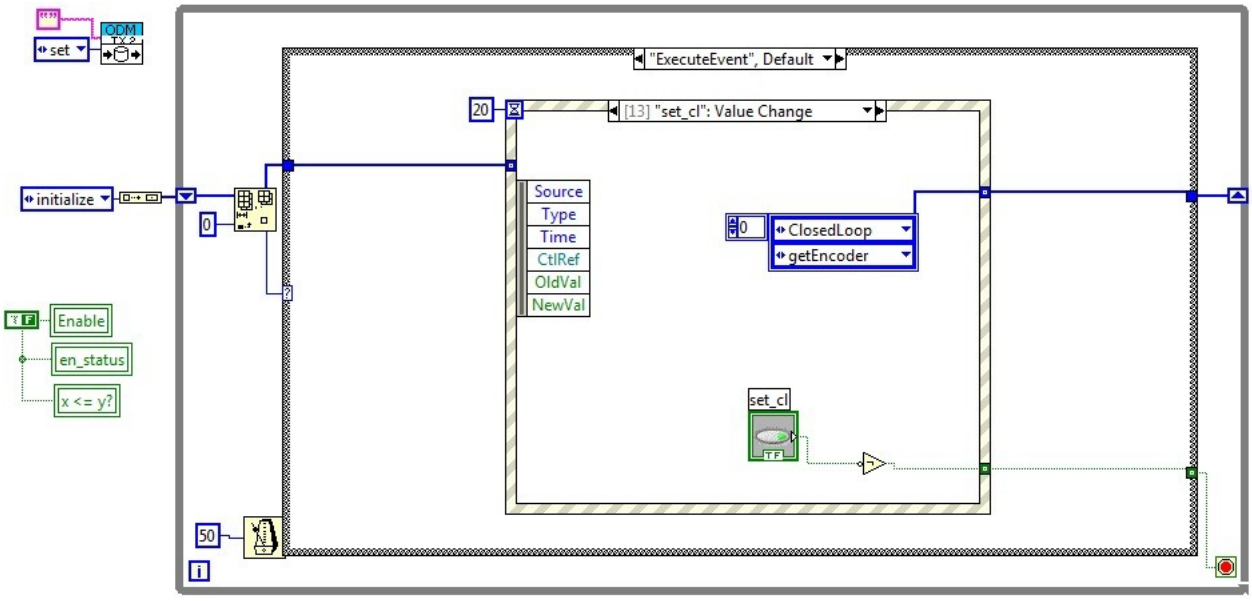


Figure 9.20: Event Structure

The hierarchy of the developed software subvi's is shown in Figure (9.21).

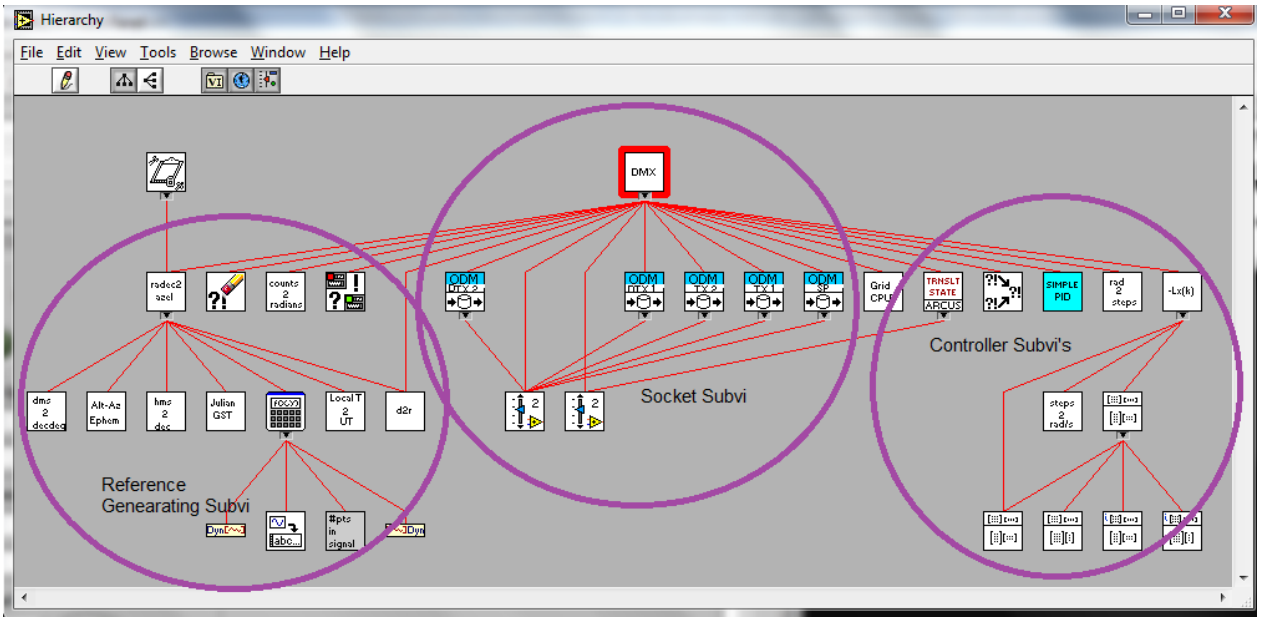


Figure 9.21: Developed Subvi Hierarchy

The main subvis are:

1. Ethernet connection Subvi.
2. PID subvi
3. State Space Controller - Proportional and Proportional and Integral and
4. Predictive Controller - Pole Placement

9.3.2.2 Ethernet connection Subvi

This Subvi makes it possible to send and receive data messages interactively between TCP connections. Based on the specified Machine Name and Port settings it establishes

connection. It then waits for the buffer to have data or the timeout period before it sends data to the port and passed to the TCP communicator.

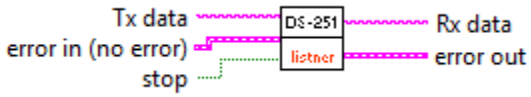


Figure 9.22: Ethernet connection Subvi
Adapted from LabVIEW TCP Passive.vi

9.3.2.3 PID Subvi

This Subvi implements the PID controller, based on the input parameters (on the left of the subvi) as shown in Figure (9.23). The PID parameters are determined using Second Method of Ziegler Nichols and the derivations are described in Chapter 5.

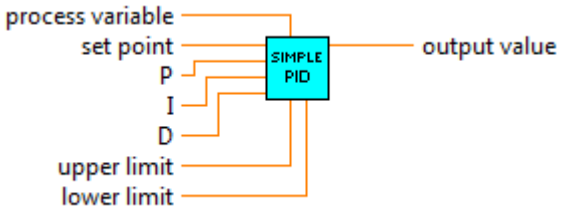


Figure 9.23: PID Subvi

9.3.2.4 State Space Subvi

This subvi implements the State Space Controller using Pole Placement given by Equation (9.1):

$$u = -Lx(k) \tag{9.1}$$

Where $L = \begin{bmatrix} 0.5521 & 0.5600 \end{bmatrix}$ is the Controller gain obtained using Pole Placement Technique as shown in Chapter 8. The input to the subvi is the feedback encoder position from the stepper motor and the velocity of the stepper motor. The output is the control signal that is summed with the reference signal as shown in Figure (9.24).

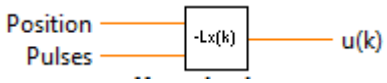


Figure 9.24: State space controller subbvi

9.3.2.5 Implementation of the developed method

The implementation environment is the same as the one described in Section 8. The difference is that the simulated plant model implemented in LabVIEW as shown in Figure (9.8) is replaced by the dish antenna implemented using the DMX-ETH-17 integrated

stepper motors that have been described in Section 9.2.1 under hard environment. The new implementation structure is shown in the Figure (9.25) and (9.26). All the nodes in Figure (9.25) perform the same function as the nodes in Figure (9.8). In Figure 9.26 the implementation is considered with NS-2 for time delays and packet loss measurements tools. Two control loops are considered for Azimuth and Altitude in Figure (9.25) and are explicitly shown in Figure (9.26)..

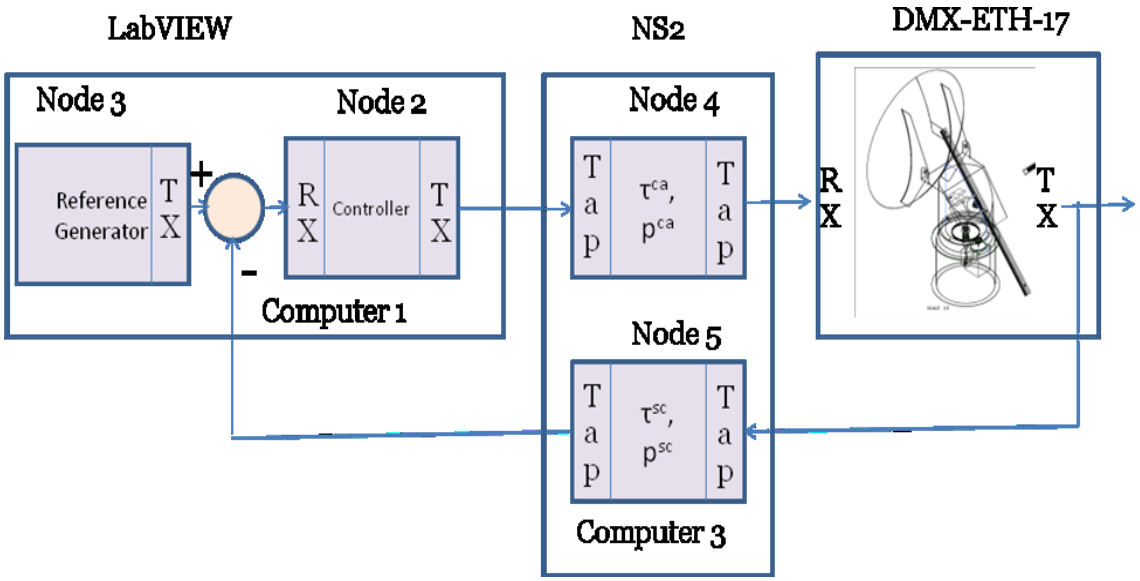


Figure 9.25: **Implementation structure**

The following DMX-ETH-17 stepper motor configurations are considered during the implementation:

1. Position closed loop control system using the stepper motor in Incremental mode.
2. The micro-step is set to **500** micro-steps in order to get higher step resolutions of **0.0036** degrees and achieve the specification pointing accuracy resolution of **< 0.004** degrees.
3. For a step size of **0.0036** degrees, in order to complete **360** degrees, **100 000** steps are required.
4. To convert from radians (angle) to motor steps, the radians value is multiplied by the conversion factor of $\frac{50000}{\pi}$.
5. The encoder counts 1000 counts in 360 degrees, with encoder resolution of 0.36 encoder counts per step. To convert from encoder counts to radians the encoder value is multiplied by the conversion factor of $\frac{\pi}{500}$.

The conversion factors are implemented as part of the developed LabVIEW software. Figure (9.26) shows how the closed loop control is implemented using the conversion factors. Where Calt is the Controller for the Altitude, Caz is the controller for the Azimuth, TXalt, RXalt, Socket transmit and receive, RA is the Right Ascension, DEC is the Declination angle.

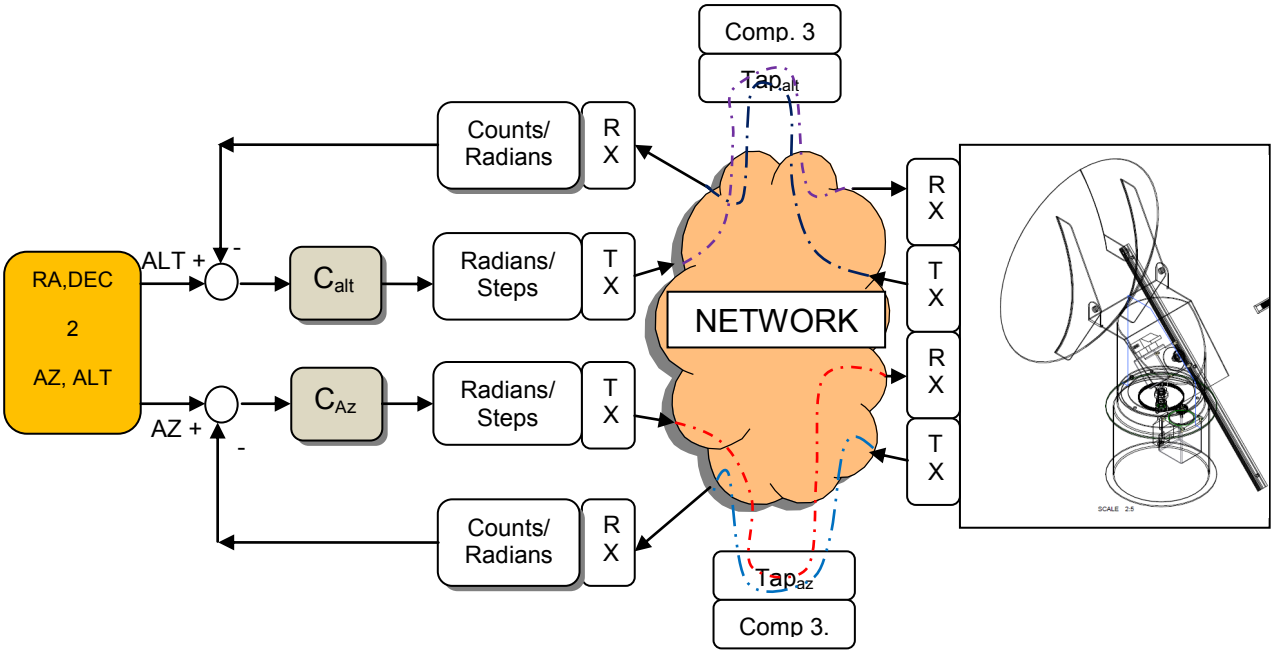


Figure 9.26: **Implemented Closed loop control using DMX-ETH-17 integrated stepper motor**

9.3.2.6 Results

The developed software has two modes of operation, manual mode and closed loop control mode as shown in Figure (9.27a). The block diagram of the developed software is shown in Figure (9.27b).

In the manual mode the motor position is commanded using the number of steps to the desired position. Based on these steps the motor goes to the target position without any feedback from the sensor. This mode is implemented under the manual tab of Figure (9.27c).

In the closed loop controller mode the motor position is commanded using the given target position as a reference. Based on the encoder feedback from the motor the error is calculated and the new control value is sent to the motor until the error is reduced and the desired position is achieved. This mode is implemented under control loop tab of Figure (9.27d).

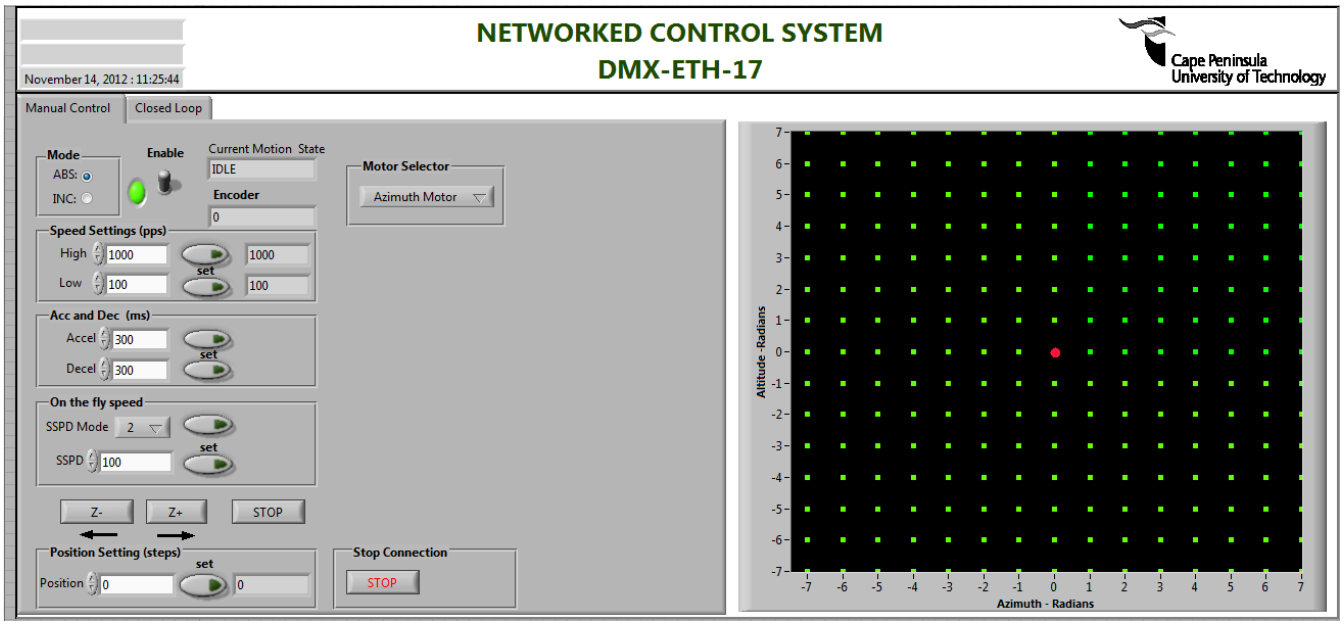


Figure 9.27a: Front panel of the developed implementation software

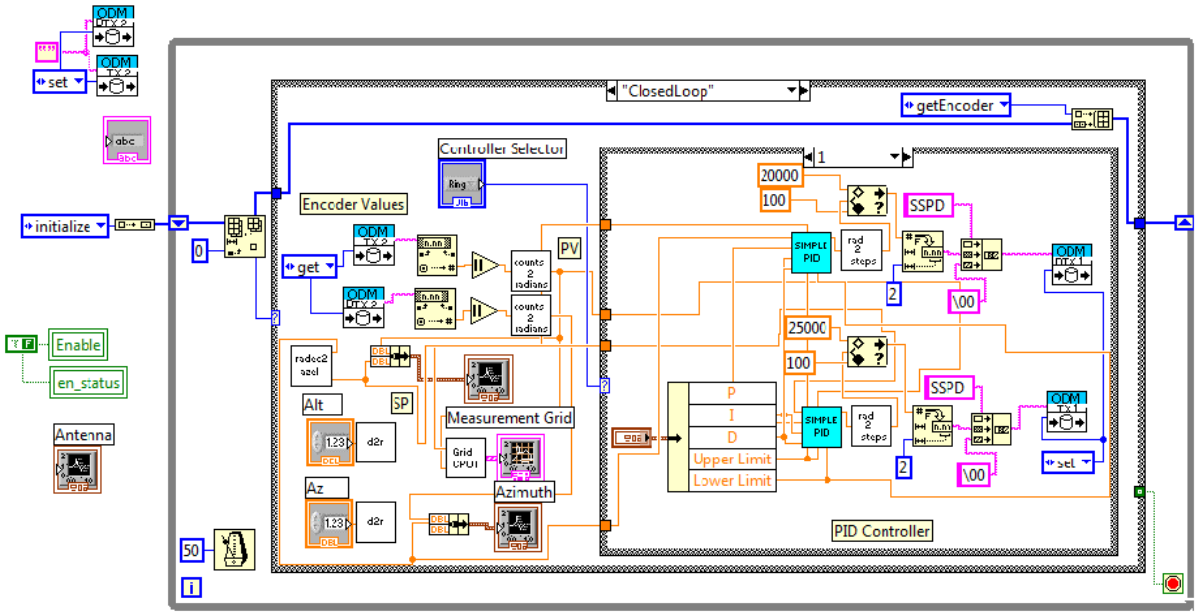


Figure 9.27b: Block diagram of the developed implementation software

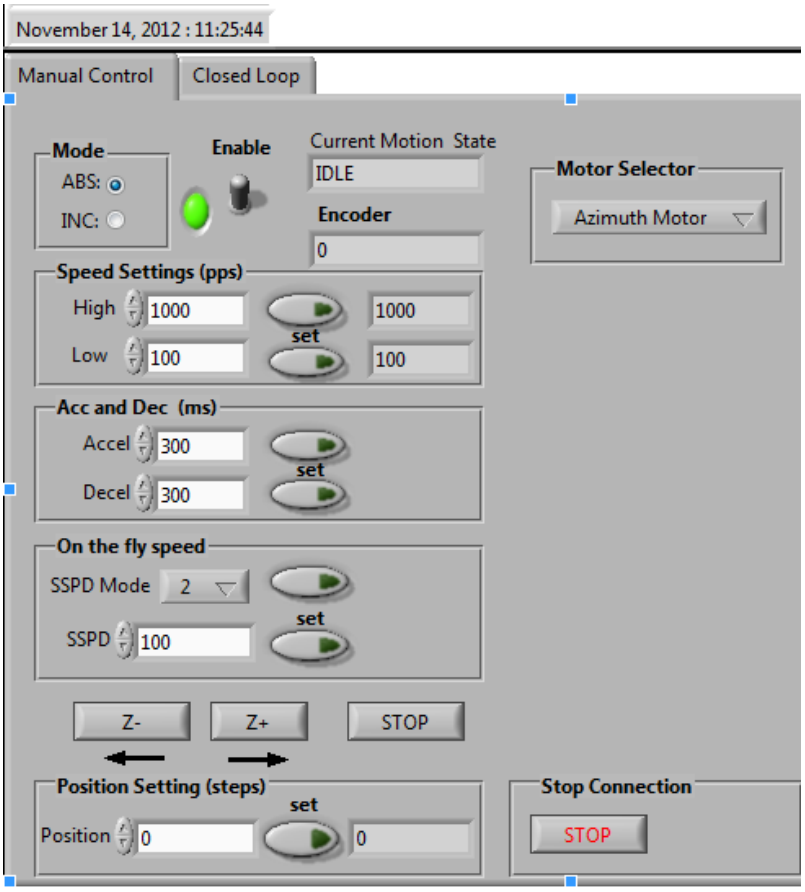


Figure 9.27c: Front panel of the manual tab mode

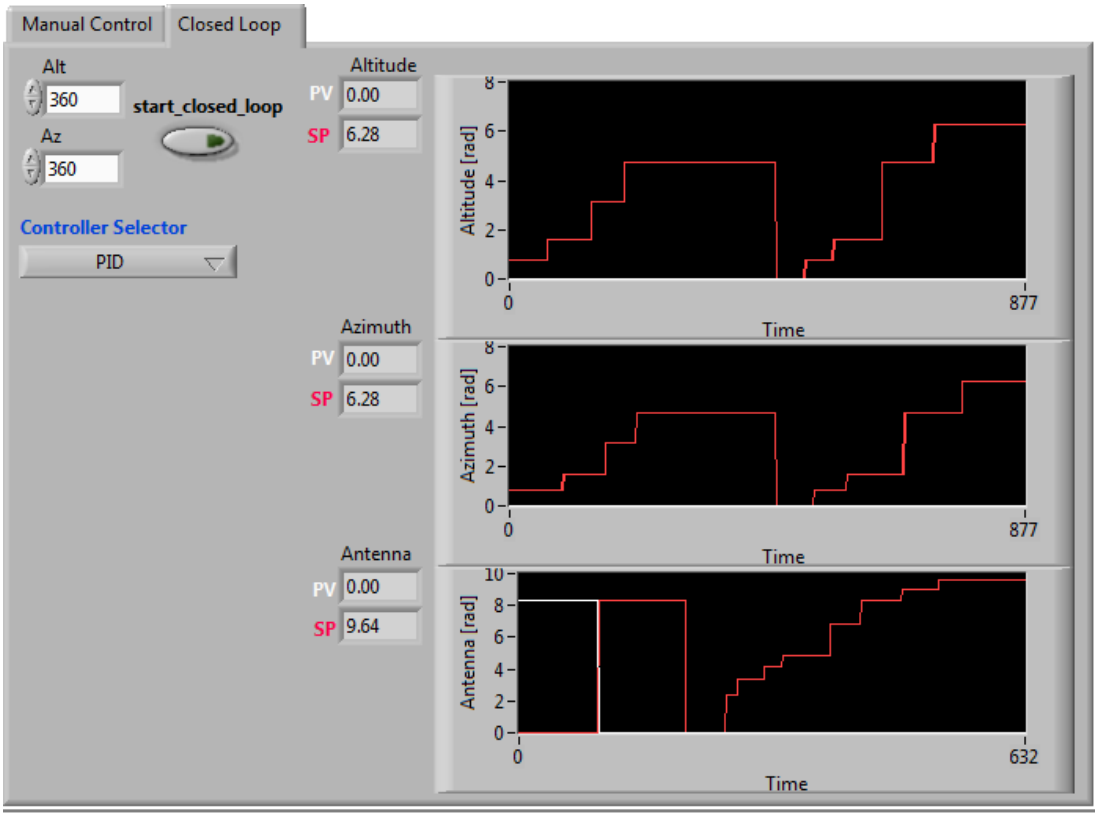


Figure 9.27d: Front panel of the closed loop control mode

For the case of the manual mode the following controls are available as shown in Figure (9.27c). The details of each control are described in details in Arcus Technology (2008):

- Absolute mode and Incremental mode.
- Motor speed setting
- Acceleration and deceleration control
- On the fly speed control
- Motor selector drop down, note that only one motor can be controller at a time.
- Position control in motor steps.

For the case of the manual mode the following current motion status indicator is available: The current motion status indicator supports one of the following status:

- IDLE: motor is not moving
- ACCEL: motion is in acceleration
- DECEL: motion is in deceleration
- CONST: motion is in constant speed
- -LIM ERR: minus limit error
- +LIM ERR: plus limit error

For the case of the closed loop control mode the following controls are available as shown in Figure (9.27d).

- Controller selector, there are two types of controllers that are supported, PID and State space controller.
- Manual entry button for Altitude and Azimuth.
- The Altitude, Azimuth and Antenna graphs show the comparison between the desired reference position vs the actual position. The displayed position values are based on the readings from the encoder.

The snapshot of the results of the dish antenna connected through NS2 and via the communication network is shown in Figure (9.27d).

9.4 Conclusion

This chapter presents a simple, scalable, and effective tool for simulation, emulation and real time implementation of NCS using LabVIEW™, and NS-2 (LabNS2). The architecture for LabNS2 is described and discussed. Tcl and Gawk scripts are developed, presented and discussed. LabVIEW™ codes are developed, presented and discussed. Radio Telescope is

presented as typical closed loop NCS. The implementation and operation procedure of the LabNS2 is described.

LabNS2 has four software modules that have to be executed to emulate the NCS environment. The modules are:

- Reference generating trajectory module implemented in LabVIEW. This module uses the software libraries described in Table 9.3 and it generates the desired positions for altitude and azimuth mounts.
- Plant and Controller software implemented in LabVIEW. This module uses the software libraries described in Table 9.3. It receives desired reference positions via the network through NS2 as shown in Figure (9.8). It realises the controller and displays actual vs desired positions for both altitude and azimuth as shown in Figure (9.12).
- Network induced time delays and packet loss are measured using Tcl scripts. The scripts are shown in Table 9.4.
- Network induced time delays and packet loss are processed offline and can also be online from the recorded output trace file.

The complexity of measuring the communication network imperfection in a networked control systems, including network induced time delays and packet loss makes it difficult for the control engineers to develop methods that can compensate or incorporate these imperfections in order to design new control design methods for networked control systems. These network imperfections degrade the performance of the closed loop control system and result in a closed loop system instability. LabNS2 addresses this problem by providing an ideal environment that can be used to investigate the influence of network imperfections especially the network induced time delay and packet loss. It is done in a real-time environment by using tools that have been used for decades in their respective knowledge areas that are the information systems in the case of NS2 and the feedback control systems in the case of LabVIEW.

LabNS2 toolset can be used successfully for simulation, emulation of hardware in the feedback control systems, and for real time systems implementation. LabNS2 can also be used for educational purposes and research work in control systems, signal processing, communication systems, network systems, verification of mathematical models for communication networks (e.g. TCP models, etc), and in the implementation of Lab experiments.

Part II of this chapter presents a real-time environment based on the DMX-ETH-17 integrated stepper motor. The environment allows for the implementation of the NCS using the software modules developed in LabNS2 by simple replacing the plant model with the real-time motor in this case the DMX-ETH-17 integrated stepper motor.

CHAPTER TEN
CONCLUSION

10.1 Introduction

Networked Control System (NCS), due to an inherent flexibility of the applied design approach provides a valuable control for real time distributed systems, not only as an interesting problem for control engineers, but also to commercial saving on multi controllers where controller can be centralized and the control action be sent from one point. However, many prerequisites and variables should be taken into consideration in NCS implementation. Moreover, the control of all the processes is mandatory, which otherwise would render the mode of operation complexity useless. In addition to the complexity of the Networked Control System, the control mode and network imperfections to be considered should be measured, analysed and incorporated as part of the closed loop networked control systems.

Traditionally control algorithms have been designed without any consideration of their implementation details. Digital control effects are maybe taken into account, but microprocessor clock speeds have increased so rapidly that in many cases, a continuous model is appropriate. However, when communication networks are used to carry feedback signals for a control system, the limited bandwidth and type of communication protocol induces unavoidable communication delays. These delays lead to subsequent packet losses resulting in an unstable system. As more data is transmitted, the delays magnitude increases.

The allocation of control tasks to processing nodes on a network affects the performance of the system. The delay between the start of the message and the delivery time is influenced by the speed of the network, message size, network traffic and communication protocol.

The communication protocol affects the reliability and fault tolerance by its priority assignment and error correction scheme.

In order to successfully analyse the behaviour of a network control system the type and location of the induced delays must be characterised and the effects on the performance of the control system be understood. These delays are mostly considered to be between the controller and actuator and between the sensor and the controller.

The control implementation of a networked control system can implicate many difficulties: as for example network induced time delays and packet loss. These aspects point to the need of a Networked Control System control strategy, which considers network imperfections as part of the closed loop control system and design robust controllers that overcome the effect of these network imperfections.

Modern control system techniques allow control engineers to optimise the control system based on costs and performance (Rollins, 1999). The optimisation techniques do not look at the system environment as a result the system could become unstable whilst trying to operate at its optimal criteria. The latter gave rise to robust control system, which considers the system environment as part of control system design. The thesis uses this technique and considers network imperfections as system environment.

10.2 Problems solved in the thesis

Two types of sub-problems are solved in the thesis;

- Design based sub-problems, and
- Real time implementation sub-problems:

10.2.1 Design based sub-problems

- Sub-problem 1: Development of mathematical model of the plant to be controlled.
- Sub-problem 2: Analysis and design of a NCS of a complex industrial system without paying attention to the network drawback influence.
- Design sub-problem 3: Simulation and determination of the characteristics of the NCS of a complex industrial system using the controller designed in point 1.
- Design sub-problem 4: Determination of the maximum values of the network drawbacks causing instability to the closed loop system.
- Design sub-problem 5: Development of a joint network and process model incorporating dynamics of the process and of the communication network.
- Design sub-problem 6: Investigate the limitations introduced by the communication network using simulations of the joint process model.
- Design sub-problem 7: Development of new methods for linear NCS optimisation and for design of a linear robust towards the network influence controller.
- Design sub-problem 8: Investigation of the dynamic characteristics of the closed loop robust control NCS using LabVIEW^{TM} and Matlab software environment.

10.2.2 Real-Time implementation based sub-problems

- Real-Time implementation sub-problem 1: Develop simulator for the NCS using Matlab and Simulink.
- Real-Time implementation sub-problem 2: Development of s simulation, emulation and real-time software environment for NCS for measurement and modelling of the communication channel imperfections with capabilities to be used in the control loop.
- Real-Time implementation sub-problem 3: Implementation of the developed NCS model in the framework of distributed system for control of different plants based on PC for emulation and control, wired communication network, Matlab/Simulink and LabVIEW software environment..

10.3 Thesis deliverables

The following deliverables have been achieved in the thesis.

10.3.1 Literature review

The literature on NCS is reviewed considering the different sub-problems that makes the NCS. The paper are grouped according to network type and communication protocols, models of time delay, Discrete Kalman Filters for systems with time delays, models of packet loss, methods for design of robust controllers for NCS, LQG controllers for NCS, H-infinity controllers, Fuzzy PID and Smith Predictor controllers. Review on simulators and emulators as used in NCS is presented.

10.3.2 Mathematical model derivations for the plant

In the thesis the mathematical model of the servo motor and Permanent Magnet (PM) stepper motor are derived using the Kirchhoff and Newton's laws of physics. The state space transformation is used to represent the mathematical models. The utilization of Matlab/Simulink packages made the development much easier when it comes to verification of the model. The results received on both packages were the same, proving the operation of the algorithm and also the consistence was tested. The results from the simulation are used later for the development of the plant controller.

10.3.2.1 Development of the plant model based on the servo motor

- The development of the mathematical model of the radio telescope system based on a servo motor is done.
- The derivation of the servo motor model is done using Kirchhoff and Newton's laws of physics.

- The description of the servo motor model is done in state space form .
- The simulation results of the open loop unit step response of the motor model in simple form is demonstrated using Matlab/Simulink. The results show that the model describes linear, second order integrating system.
- The derivation of the dish model incorporating the model of the motor is described.
- The simulation results of the open loop unit step response of the model of the full dish is demonstrated using Matlab/Simulink. The results show that the model describes a linear, second order integrating system.

10.3.2.2 Development of the plant model based on the PM stepper motor

- The model with state variable position and velocity of the motor is derived.
- Model of the stepper motor is derived and discussed for comparison.
- The derivation of the mathematical model of the dish and the motor under the wind disturbance is described.
- The simulation results of the open loop unit step response of the model of the motor and the load is demonstrated using Matlab/Simulink. The results show that the plant has nonlinear behavior.

10.3.3 Investigation of the influence of the network induced time delays for the case of standard controllers using simulations

In order to understand the influence of network induced time delays, an investigation is performed using standard controllers and the results are simulated in Matlab/Simulink. The controllers are designed for the servo-motor model without considering the network delays given a closed loop system requirements and performance measures. The controller design methods and the simulation environment are described and discussed. Simulations are performed for various values of the delays. The considered cases are:

- PID controller is designed using 2nd Method of Ziegler-Nichols and is used in a closed loop system with delays between the controller and the actuator and between the sensor and the controller. The simulation results showing the performance of the closed loop system are presented using Matlab/Simulink.
- PID Controller designed using Pole Placement method is used in the closed loop system described above. The simulation results of the closed loop system behaviour for various values of the delays are presented using Matlab/Simulink.
- The state space proportional controller is designed using Pole Placement and is used in the considered closed loop system. The simulation results of the closed loop

system performance for various values of the delay are presented using Matlab/Simulink.

- The state space PI controller is designed using the Pole Placement method and the simulation results of the closed loop system behavior for various values of the delay are presented using Matlab/Simulink.

In all the cases the results showed that the time delay between the controller and the actuator and the time delay between the sensor and the controller affect the performance and stability of the system. The performance of the system is worse for larger values of the delays. In all cases there is a critical value of the delays at which the system becomes unstable. On the basis of the analysis of the obtained results it is concluded that the delay between the controller and the actuator influences the output of the system, and the delay between the sensor and the controller influences the amount of data reaching the controller. The two delays are considered during the design of the predictive controller incorporating the delay between the controller and the actuator and the sensor and the controller.

The critical values of the delay are used further in Chapter 6 and 7 to design robust predictive Kalman filter and state space controllers which can overcome the destabilising effect of the delays.

10.3.4 Derivation of the Discrete Kalman Filter

The steps followed during the derivation of the Discrete Kalman Filter are:

- Design of the DKF for the case without delay in the data coming from the sensor.
- Design of the DKF for the case of delay in the data coming from the sensor.

The equations of these filters are derived. The application of the DKF for the case without delay is done in the thesis as the delay is between the sensor and the controller is considered during the design of the robust controller.

10.3.5 Development of a method for design of a robust predictive controller

The problem for the destabilising effect of the network induced time delay between the controller and the actuator or as a sum of the delays between the controller and the actuator and between the sensor and the controller is solved by development of state space predictive controller using time shifting approach and state predictor. The design of the controller is based on a model of the plant with delay in the control vector. The time shifting approach allows the design of the controller to be performed for a model without

time delay. Then the control action is based on the future values of the state space vector.

The state predictor is developed to predict the future value of the state using the present and past values of the state. This technique is made possible by the use of the plant model Transition Matrix. The general transition matrix is adapted and used for the prediction of the future state. The solution is simple and works with time delays that are greater than the sampling period, also it has minimum storage requirements and it is not computationally heavy.

The use of Matlab/Simulink and LabVIEW simulation and implementation software packages made it easy to implement the algorithm in real-time. The comparison is made on the results received from the two software packages. The comparison proves that the implementation of the algorithm is operational by receiving the same results on both simulation tools.

10.3.6 Investigation of the influence of the network induced delays for the case of the Kalman filter and the predictive controller.

The problem for the destabilising effect of the network induced time delay between the sensor and the controller and between the sensor and the controller is investigated. Three types of simulations are implemented in order to investigate the influence of the network delays over the closed loop system performance:

1. Open loop control of the system consisting of the plant and the DKF - no delay are considered.
2. Closed loop control of the system consisting of the plant and the DKF - no delays are considered.
3. Closed loop control of the system consisting of the plant and the DKF - delays are considered:
    - the two delays - between the controller and the actuator and between the sensor and the controller.
    - one delay - between the controller and the actuator OR between the sensor and the controller only.

The type two simulations are implemented for two cases of state space controllers - designed using Pole Placement and LQG methods.

The type 3 simulations are implemented for the following cases:

I. The NCS induced time delays impact over the closed loop behaviour in the case of state controller designed using Pole Placement method for a system without

delays is investigated. Use of a DKF designed for a system without delays is also investigated. The simulation results showing the performance of the closed loop system are presented using Matlab/Simulink.

II. The NCS induced time delays impact over the closed loop behavior in the case of state robust predictive optimal controller designed using Optimal Control method for the system without delays is investigated. Use of a DKF designed for a system without delays is also investigated. The simulation results showing the performance of the closed loop system are presented using Matlab/Simulink

III. The NCS induced time delays impact over the closed loop behavior in the case of state robust predictive optimal controller designed using Pole Placement method for a system with delay in the plant control and with a constant disturbance is investigated. Use of a DKF designed for a system without time delays is also investigated. The simulation results showing the performance of the closed loop system are presented using Matlab/Simulink

IV. The NCS induced time delays impact over the closed loop behavior in the case of state controller designed using Optimal Control method for a system with delay in the plant control and with a constant disturbance is investigated. Use of a DKF designed for a system without time delays is also investigated. The simulation results showing the performance of the closed loop system are presented using Matlab/Simulink.

10.3.7 Development of simulation and emulation environment

A toolset for the simulation and emulation of NCS environment in order to investigate the influence of network imperfection in a real environment is developed and discussed. The architecture of the developed toolset is presented. The toolset is called LabNS2. The name is derived from LabVIEW™ and Network Simulator version 2 (NS-2) which are the main software environments used in developing the toolset. The toolset uses Windows environment as an operating system. NS-2 is developed for UNIX environment. In order to emulate UNIX environment within the Windows environment, Cygwin is used. Cygwin emulates UNIX environment in the Windows environment. The developed software is listed in Table 10.1.

Table 10.4: Developed simulation and emulation software

Item	Simulation Software	Description	Appendix
1	Nilson_pdf_tca_tsc.m	Implementation of Nilson, 1998 PDF function for Tsc and Tca.	B2.1
2	tf_con_zncePI.m	Transfer Function Pole Placement Proportional Integral Controller	B5.1
3	ss_con_ppP.m	State Space Pole Placement, Proportional Controller.	B5.2
4	ss_con_ppPI.m	State Space Pole Placement, Proportional Integral Controller.	B5.3
5	Exp_one	This m.file simulates a open loop structure where plant and the DKF are considered without consideration for network induced time delays. The purpose of this procedure is to verify the functionality of the filter and to confirm that the filter tracks the step response of the open loop discrete plant.	B8.1
6	Exp_two	This simulation procedure is performed considering Plant, Controller and DKF considering constant network induced time delays. The purpose of this procedure is to illustrate the influence of network induced time delay considering a plant, controller and DKF that are not designed to cope with the delay. An option to use either Pole Placement controller design method OR LQG controller design method is available.	B8.2
7	Exp_three	This simulation procedure is performed considering Plant, Controller and DKF considering constant network induced time delays, between the controller and the actuator and the sensor and the controller.	B8.3
8	Exp_four	This simulation procedure is similar to simulation procedure 3, the difference is that in this simulation the delays are considered between controller and actuator ONLY.	B8.4
9	Exp_five	This simulation procedure is similar to simulation procedure 3, the difference is that in this simulation the delays are considered between sensor and controller ONLY.	B8.5
10	Exp_six	This simulation procedure is performed considering Plant, and Robust Predictive Controller considering constant network induced time delays. The delays are considered between	B8.6

		controller and actuator ONLY.	
11	measure-tdpl-aN.tcl	Measure time delay and packet loss.	D9.1
12	calculate_packet_loss.awk	Calculates the packet loss.	E9.1
13	Calculate-time_delay.awk	Calculates time delay	E9.2
14	lt2ut.vi	Converts local time to Universal Time (UT)	F9.1
15	Juliandate.vi	Converts UT to GST	F9.2
16	hms2dd.vi	Converts from Hour Minutes Seconds to Decimal degrees	F9.3
17	dms2deg.vi	Converts from Degrees Minutes Seconds (DMS) to Decimal degrees	F9.4
18	Azel.vi	Converts from RA,DEC to Alt Az using Ephemeris (Almanac) equations	F9.5
19	udp_tx_aN.vi	Receives packet data and send it via network. Where **N** in the name is the antenna number.	F9.6
20	udp_parser.vi	Parses data from UDP read.vi and sends it to the controller.	F9.7
21	PID.vi	Implements the PID controller.	F9.8
22	ss_con.vi	Implements state space controller for both Proportional and Proportional Integral controllers.	F9.9
23	dish_N_tf.vi	Implements the plant model in transfer function form.	F9.10

10.3.8 Real-Time implementation

The environment for real-time implementation is based on the DMX-ETH-17 integrated stepper motor. The stepper motor is developed by Arcus Technologies and has the following structure: Encoder, Driver and Ethernet communication. Based on this motor a prototype dish model is developed where the two stepper motors are mounted and used to drive the dish according to the Azimuth and Altitude coordinates set points. The developed software modules and methods are used to drive the dish prototype (trainer-antenna).

10.4 Benefits of the developed methods

Benefits of the developed methods for design of NCS controller and Kalman filter come from the improved plant and business performance. In general terms, the revenues come from the use of centralised control compared to distributed control where each plant has its own controller. Reduced energy consumption, and less radio interference for the case of Radio Telescope applications could be obtained. The developed methods may also influence reduction in a number of other operating costs including maintenance, equipment wear and staff utilization.

The engineering benefits come from improved process troubleshooting and assistance in making quicker and more accurate decisions. This ultimately relates to improved process operations and avoiding the danger of closed loop system instability.

10.5 Application of developed strategy, methods, algorithm and programs

The thesis deliverables can be successfully applied in the following field in industry:

- Radio Telescope.
- Satellite systems.
- Motor industry.
- Mining industry for extraction of metal ions.
- Research and education at Universities.
- Robotic systems.

10.6 Future developments of methods and their applicability

The future developments can be decomposed in the following way:

- Testing of the developed methods on the working plant.
- Testing of the state shifting program for different experiments.
- Development of the methods for mathematical modelling of the network imperfection and incorporation of this model into the developed methods.
- Application of the developed methods and programmes for different processes in industry.

10.7 Publications in connection with the dissertation

- Mkondweni NS and R. Tzoneva. 2012. LabVIEW^{TM} and NS2 (LabNS2) Co-Simulation, Co-Emulation, and Real-Time control toolset for the Investigation of Network Induced Time Delays and Packet Loss in Networked Control System. Submitted to International Journal of Innovative Computing Information and Control (IJICIC), 2012 (IJICIC-12-11004)

- Mkondweni NS and R. Tzoneva. 2012 Design of an optimal, robust, predictive controller for networked control systems. Submitted to Control Engineering and Applied Informatics Journal.

REFERENCE

Alekal Y., Brunovsky P., Chyung D.H. and Lee E.B. 1971. The Quadratic Problem for systems with time delays. IEEE Transactions on Automatic control, Vol. AC-16. 529:551. December 1971

Alexander H.L. 1991. State estimation for distributed systems with sensing delay. In. SPIE vol. 1470 Data Structures and Target Classification, 1991.

Al-Hammouri A, Liberatore V, Al-Omari H, Al-Qudah Z, Branicky M, and Agrawal D. 2007. A Co-Simulation Platform for Actuator Networks. ACM SenSys 2007

Amirzhalingam R., Sung S. and Lee J. 2000. Two-step procedure for date-based modeling for inferential control application, AICHE Journal, V.76, 1974:1988

Anderson B., and Moore J. 1979. Optimal filtering, Prentice-Hall, Englewood Cliffs, NY, 1979

Anderson M., Henriksson D., Cervin A. and Arzen K. 2005. Simulation of Wireless Networked Control Systems, Proceedings of the 44th IEEE Conference on Decision and control, and the European Control Conference, 2005. December, 2005.

Antoniou S. 2007. http://www.trainsignaltraining.com/networking-basics-tcp-udp-tcpip-osi-models/2007-10-29. October 2007

Arcus Technology. 2008. DMX ETH Manual, January 2008

Astrom K., and Wittenmark B. 1997. Computer-controlled systems, Prentice-Hall, IHE, Upper Saddle River, NY, USA

Astrom, K. and Hagglund, T. 1995. PID Controllers: Theory, Design, and Tuning 2nd Edition, ISBN 1-55617-516-7

Athans M. 1971. The role and use of the stochastic Linear-Quadratic-Gaussian problem in control system design. IEEE Transactions on Automatic control, Vol. AC-16. 529:551. December 1971

Baillieul, J. and Antsaklis, P.J. 2007. Control and communication challenges in networked real-time systems, *Proceedings of IEEE, Vol 95, No1, January 2007 pp9-28*.

Baillieul, J. and Antsaklis, P.J. 2007. Special issue on technology of networked control systems, *Proceedings of IEEE, Vol 95, No1, January 2007 pp5-8.*

Barakat C. 2003. Exercises on “ns-2”. Inria, Planete research group. November 2003

Behboodian, A. 2005. Model-based design and FPGA implementation with simulink, Mathworks, *Webmania*.

Bemporad A. 2009. Decentralized and hybrid model predictive control. 3rd WIDE Phd School on Networked Control Systems, Siena, Italy, July 7-9, 2009

Bishop R.H.1993. Modern control systems analysis and design using Matlab. Addison-Wesley.

Borkar V.S. and Mitter S.K 1995. LQG Control with communication constraints. LIDS-+ 2326. December 1995.

Boston Technical Books.1994. PC Instrumentation for the 90s Fourth Edition: 251:260

Branicky M. S., Phillips M. and Zhang. 2000. Stability of Networked Control Systems: Explicit Analysis of Delay. Proceedings of the American Control Conference. June 2000

Bullo R. 2009. Distributed control and coordination algorithms. 3rd WIDE Phd School on Networked Control Systems, Siena, Italy, July 7-9, 2009

Cardwell N. Savage S. and Anderson T. 2000. Modelling TCP Latency. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, Vol. 3 (2000). 1742-1751

Cervin A. 2009. Simulation of networked control systems. 3rd WIDE Phd School on Networked Control Systems, Siena, Italy, July 7-9, 2009

Chow M. and Tipsuwan Y. 2003. Gain adaptation of networked DC Motor controllers based on QOS variations. IEEE Transactions on Industrial Electronics, Vol. 50. October 2003

Chow, P. Seo, S.O. Rose, J. Chung, K. Paez-Monzon, G. and Rahardga, I. 1999. The design of an SRAM-Based field-programmable gate Array – Part I: Architecture, *IEEE Transactions on very large scale integration (VLSI) systems*, VOL. 7, NO 2, June 1999

Cloosterman M.B.G, Hetel L., Van de Wouw N., Heemels W.P.M.H., Daafouz J., and Nijmeijer H. 2010. Controller synthesis for networked control systems. Automatica 46 (2010) 1584-1594. June 2010

Craig K. 2004. Stepper Motor, Rensselaer Polytechnic Institute

De Silva C.W. and Wang Y. 2008. Control Output Devices-Actuators and Displays for Process Automation Part 12 in a series of tutorials in instrumentation and measurement *IEEE Instrumentation & Measurement Magazine*, February 2008: 38-45.

Ding F. and Chen T. 2005. Hierarchical identification of Liftef State-Space models for general dual-rate systems. IEEE Transactions on circuits and systems, Vol. 52. 1179-1187

Donkers M.C.F, Heemels W.P.M.H., Bernardini D., Bemporad A. and Sheer V. 2012. Stability analysis of stochastic networked control systems. Automatica 48 (2012) 917-925

Dressler R.M. and Tabak D.1971. Satellite tracking by combining optimal estimation. IEEE Transactions on Automatic control, Vol. AC-16. 529:551. December 1971

Eshbach OW and Poston TR, Tapley BD. 1990. Eshbach's handbook of Engineering fundaments. John Wiley & Sons Inc. 8-209. 1990

Eyisi E., Bai J., Riley D., Weng J., Yan W., Xue Y., Koutsoukos X. and Sztipanovits J. 2012. NCSWT: An integrated modelling and simulation tool for networked control systems. Simulation Modelling Practice and Theory 27 (2012) 90-111

Fall K and Varadhan K. 2008, The ns Manual, The VINT Project, January 2008

Flaus, J.M. Pons, M.N. Cheruy, A. Engasser, J.M.1989. Adaptive algorithm for estimation and control of fed-batch bioprocess.

Gao H., Chen T. and Lam J. 2008. A new delay system approach to network-based control. Automatica 44 (2008) 39-52. April 2008

Garone, E., Sinopoli, B. and Casavola, A. 2010 LQG control over lossy TCP-like networks with probabilistic packet acknowledgements, Int. J. Systems, Control and Communications, Vol.

Gary D. E. Radio Astronomy: Lecture #9, Physics 728 Spring 2005, NJIT, Calibrations

Gawronski W. 2004. Control and Pointing challenges of antennas and (Radio) Telescopes, IPN Progress Report 42-159. 15 November 2004

Gawronski W. 2008. Modeling and control of antennas and telescopes. Springer, New York

Gelb A. 1974., Applied Optimal Estimation, The Analysis Science Corporation, 1974.

Ghidella, J. 2005. Model-based design of embedded control systems with simulink, Mathworks, *Webmania*.

Gopalakrishnan A., Kaisare NS. and Narasimhan S. 2010. Incorporating delayed and infrequent measurements in Extended Kalman Filter based nonlinear state estimation. Journal of Process Control 21 (2011): 119-129

Gudi R., Shan S., and Gray M. 1994. Multirate state and parameter estimation in an antibiotic fermentation with delayed measurements. Biotech, Bioeng. V.44:1271-1278

Gudi R., Shan S., and Gray M. 1995. Adaptive multirate state and estimation strategies with application to a bioreactor. AICHE Journal, V.71, 2451-2764

Gupta A. 2010. A survey on Certainty Equivalence Property and Separation Property in Stochastic Stochastic Differential Games under various Information Structure.

Gupta V., Hassibi B. and Murray R.M. 2006. Optimal LQG control across packet-dropping links. Systems Control Lett. (2006), doi:10.1016/j.sysconle.2006.11.003

Hadjicostis C. N and Touri R. 2002. Feedback control utilizing packet dropping network links.

Halevi Y. and Ray A. 1988. Integrated Communication and Control Systems: Part I – Analysis. Journal of Dynamic Systems, Measurement, and Control. Vol. 110 December 1988

Halevi Y. and Ray A. 1988. Integrated Communication and Control Systems: Part II – Design Considerations. Journal of Dynamic Systems, Measurement, and Control. Vol. 110 December 1988

Heemels M. 2009. Stability and stabilization of networked control systems. 3rd WIDE Phd School on Networked Control Systems, Siena, Italy, July 7-9, 2009

Hespanha, J.P. Naghshtabrizi, P. and Xu Y, A. survey of recent results in networked control systems

Himwich, W. E. 1993. Pointing model derivation, NASA/Goddard Space Flight Center Space Geodesy Project, Version 8.2, September 1.

http://c-perl-programming.blogspot.com/2009/05/ns2-trace-queue-vs-monitor-queue.html May 2009, NS2 trace-queue v.s monitor queue, Software, System and Network

http://csep10.phys.utk.edu/astr161/lect/time/coordinates.html

http://en.wikipedia.org/wiki/Horizontal_coordinate_system

http://en.wikipedia.org/wiki/Linear-quadratic-Gaussian_control

http://en.wikipedia.org/wiki/MATLAB

http://en.wikipedia.org/wiki/Reber_Radio_Telescope

http://en.wikipedia.org/wiki/Scilab

http://ipv6.com/articles/general/User-Datagram-Protocol.htm

http://nsnam.isi.edu/nsnam/

http://rtns.sssup.it/RTNSWebSite/RTNS.html

http://www.control.lth.se/truetime

http://www.cs.cornell.edu/skeshav/real/overview.html

http://www.cygwin.com/

http://www.ece.rutgers.edu/~gajic/psfiles/lqr.pdf

http://www.ephemeris.com/space-time.html

http://www.mathworks.com/

http://www.modelica.org/

http://www.ni.com/

http://www.opnet.com/

http://www.radiosky.com/radioeyesishere.html

http://www.satsig.net/lat_long.htm

http://www.scilab.org/

http://www.tcpipguide.com /

http://www.ti.com/ww/en/motor_drive_and_control_solutions/motor_control_type_stepper.htm

http://www.wireshark.org/

Hu L., Bai T., Shi P. and Wu Z. 2006. Sampled-data control of networked linear control systems. Automatica. November 2006.

Huang D and Nguang S. 2009. Robust Control for Uncertain Networked Control Systems with Random Delays, Springer

Imer O., Yuksel S. and Basar T. 2005. Optimal control of LTI systems over unreliable communication links. Automatica 42 (2005). 1429-1439 September 2005. ScienceDirect

Ishido Y., Takaba K., and Quevedo D.E. 2011. Stability analysis of networked control systems subject to packet-dropouts and finite -level quantization. Systems and Control Letter 60(2011) 325-332

Ishii H. 2009. Feedback control over limited capacity channels. 3rd WIDE Phd School on Networked Control Systems, Siena, Italy, July 7-9, 2009

Jia T., Niu Y. and Wang X. 2010. H Control for networked systems with data packet dropout. International Journal of Control, Automation, and Systems (2010) 8(2):198-203.

Jian F. and Shen-Quan W. 2012. Reliable Fuzzy control for a class of nonlinear networked control systems with time delay. Acta Automatica Sinica, Vol.38, No.7 July, 2012

Johansson M. 2009. Distributed optimization. 3rd WIDE Phd School on Networked Control Systems, Siena, Italy, July 7-9, 2009

Julier S. and Uhlmann. 2005. Fusion of time delayed measurements with uncertain time delays. Proceedings of the American Control Conference, Portland, OR USA:4028-4033

Jung, S. and Kim, S. 2005. Hardware implementation of a neural network controller with an MCU and an FPGA for a nonlinear system, IFAC

Kalman R.E. 1960. A new approach to linear filtering and prediction problems. Transactions of the ASME - Journal of Basic Engineering, 82 (Series D): 35-45. 1960

Karris S, Introduction to simulink with engineering applications, Mathwork, Webmania

Ke C. 2005, How to measure packet loss rate, jitter, and end-to-end delay for UDP-based applications. http://140.116.72.80/~smallko Phd candidate, EE Department, NCKU, Taiwan

Kirk D.E. 2004. Optimal control theory. An introduction, 2004. Prentice-Hall, Inc. ISBN: 0486434842

Kumar P.R. 2009. Fundamental issues in networked control systems – 1: Information. 3rd WIDE Phd School on Networked Control Systems, Siena, Italy, July 7-9, 2009

Kurzweil F. 1963. The Control of Multivariable Processes in the Presence of Pure Transport Delays. IEEE Transactions on automatic control. 27-34 January 1963

Lall. S. 2009. Decentralized control. 3rd WIDE Phd School on Networked Control Systems, Siena, Italy, July 7-9, 2009

Lam J., Gao H and Wang C. 2007. Stability analysis for continuous systems with two additive

time-varying delay components. Systems & Control Letters 56 (2007), 16-24 July 2006. ScienceDirect

Larsen T.D., Anderson N.A., Ravn O. and Roulsen N. 1998. Incorporation of Time Delayed Measurements in a Discrete-time Kalman Filter., in Proceedings of the 37th IEEE Conference on Decision and Control, Tampa, FL, USA, 1998:3972:3977.

Lemmon M.D. 2009. Event-triggered feedback in control estimation, and optimization. 3rd WIDE Phd School on Networked Control Systems, Siena, Italy, July 7-9, 2009

Lennon, T. Eryilmaz, B. 2005. Using simulink parameter estimation and test data to calibrate a model of an electronic motor, *Mathworks Webmania*.

Lewis, D.M. Galloway, D. R. Van Ierssel, M. Rose, J. and Chow, P. 1998. The transmogrifier-2: A 1 Million gate rapid-prototyping system, *IEEE Transactions on very large scale integration (VLSI) systems*, VOL. 6, NO 2, June 1998

Li K and Baillieul J. 2004. Robust Quantization for Digital Finite Communication Bandwidth (DFCB) Control. IEEE Transactions on automatic control. 1573- 1584 September 2004

Li X. and Sun S. 2012. Robust H_∞ control for networked systems with random packet dropouts and time delays. 2012 International Workshop on Information and Electronic Engineering (IWIEE 2012). Procedia Engineering 29 (2012) 4192-4197)

Lian F. Moyne J.R. and Tilbury D.M. 2007. Performance evaluation of control networks: Ethernet, ControlNet and DeviceNet. Proceedings of the IEEE. Vol. 95, No.1, January 2007.

Lian F., Moyne J. and Tilbury D. 2002. Optimal controller design and evaluation for a class of networked control systems with distributed constant delays. Proceedings of the American Control Conference. May 2002.

Lian F., Moyne J., and Tilbury D.M. 2001. Network Control Systems Toolkit: A Simulation Package for Analysis and Design of Control Systems with Network Communication, Technical Report UM-ME-01-04

Lian, F. 2001. Analysis, design, modelling, and control of networked control systems, PhD Thesis, Mechanical Engineering, University of Michigan

Lin C. 2008. Lab assignment 1. Telcom 2321-CS 2520 Spring term 2008

Lin C. and Chen C. 2005. Robust control of interconnected network schemes subject to random time-varying communication delays. Int. J. Electrin. Commun. (AEU) 60 (2006). 647-658 September 2005

Liou L. and Ray A. 1991. A stochastic regulator for Integrated Communication and Control Systems: Part I – Formulation of Control Law. Journal of Dynamic Systems, Measurement, and Control. Vol. 113 December 1991

Ljung, L. 1987. System Identification: Theory for the User, Englenwood Cliffs, New Jersey, USA. Prentice Hall, Inc

Lu C. Tsai J.S.Su T. and Jong G. 2003. Delay-dependent robust Kalman filtering for interval systems with time delay. Proceedings of the 42nd IEEE Conference on Decision and Control

Maui, Hawaii USA, December 2003.

Ma C, and Fang H. 2005. Research on mean square exponential stability of networked control systems with multi-step delay. Applied Mathematical Modelling 30 (2006). 941-950 June 2005. ScienceDirect.

Ma C., Chen S. and Liu W. 2007. Maximum allowable delay bound of network control systems with multi-step delay. Simulation Modelling Practice and Theory xxx (2007). ScienceDirect

Maral G and Bousquet M. 2002, Satellite communication systems, systems, technologies and technology, Fourth Edition, ISBN:0471971669:263:265

Mendel J. 1971. Computational Requirements for Discrete Kalman Filter, IEEE Transactions on Automatic control, Vol. AC-16. 778-758. December 1971

Miler D. F. Basics of Radio Astronomy for the Goldstone-Apple Valley Radio Telescope, Jet Propulsion Laboratory (JPL), JPL D-13835. April 1998: 63-74

Mishra J. and Rajamani V.S. 1975. Least-Square State estimation in time delay systems with colored observation noise: An innovation approach. IEEE Transactions on automatic control , February 1975.

Moayedi M., Foo Y. and Soh Y. 2010. Optimal and suboptimal minimum-variance filtering in networked systems with mixed uncertainties of random sensor delays, packet dropouts and missing measurements. International Journal of Control, Automation, and Systems (2010) 8(6):1179-1188.

Moog 2003. Industrial newsletter, Issue 2, November 2003

Morar A. 2003. Stepper motor model for dynamic simulation, Petru Maior University of Targu-Mures, Romania, Vol. 44, Number 2, 2003.

Morris, K. 2006: Go, Stop, Yield Dude! Where's my chip?, FPGA and structured ASIC journal

Mutha R.W., Cluette., and Penlidis A. 1997. A new multirate measurement based estimator: emulsion copolymerization batch reactor case study. Ind. Eng. Chem Res, V.36, 1036-1047

Naidu D. S. 2003. Optimal Control Systems. CRC Press LLC. 2003.

Nair G. N. and Evans R. J. 2000. Stabilization with data-rate-limited feedback: tightest attainable bounds. Systems & Control Letters 41 (2000) 49-56

Nair G. N. and Evans R. J. 2004. Stabilizability of stochastic linear systems with finite feedback data rates. Soc. Indus. Appl. Math., USA, vol. 43, no2, (2004) 413-436

Nair, G.N. Fagnani, F. Zampieri, S. and Evans, R. J. Feedback Control under Data Rate Constraints and Overview *Accepted for publication in Proceedings of the IEEE (special issue on "The Emerging Technology of Networked Control Systems"), early 2007.*

Nilsson, J. 1998. Real-Time control systems with delays, Phd Thesis, *Lund Institute of Technology*, ISSN 0280-5316

Nise, N.S. 2004, Control System Engineering, Fourth Edition, ISBN 0-471-44577-0

Otanez P.G. Parrott J.T. Moyne J.R. and Tilbury D.M. 2007. The Implications of Ethernet as a Control Network. Proceedings of the IEEE. Vol. 95, No.1. January 2007.

Padhye J. Firoiu V. Towsley D. and Kurose J. 1998. Modeling TCP Throughput: A Simple Model and its Emperical Validation. SIG-COMM. September 1998

Pelt R. 2005, Implementing a FPGA-based broadband modem using model-based design, Alter Corporation San Jose, California, USA 95134

Peng C., Fei. and Tian E. 2013. Networked control for a class of T-S fuzzy systems with stochastic sensor faults. Fuzzy Sets and Systems 212 (2013) 62-77

Perbellini G. 2005. Network advanced modelling in NS-2, Universita degli Studi di Verona, Facolta di Scienze MM.FF.NN

Pohjola M and Nethi S. PiccSIM version 0.81 Manual Helsinki University of Technology, June 2009

Prasad V., Senley M., Russo L. and Wayne Bequette B. 2002. Product property and production rate control of styreme polymerization. J. Process Control V.12, 353-372

Prytz G. 2008. A performance analysis of EtherCAT and PROFINET IRT. ABB, Emerging Technologies and Factory Automation (ETFA), 408-415. 2008

Quaglia D., Muradore R., Bragantini R. and Fiorini P. 2012. A SystemC/Matlab co-simulation tool for networked control systems. Simulation Modelling Practice and Theory 23 (2012) 71-86

Rao C., Rawlings J., and Mayne D. 2003. Constrained state estimation for nonlinear discrete-time systems: stability and moving horizon approximations. IEEE Transaction Automation. Control, V98, 2003:276-258

Rhodes I. 1971. A Tutorial introduction to estimation and filtering. IEEE Transactions on Automatic control, Vol. AC-16. 529:551. December 1971

Richard J. 2003. Time-delay systems: an overview of some recent advances and open problems. Automatica 39 (2003) 1667-1694. April 2003

Robertson D., Lee J., and Rawlings J. 1996. A moving horizon-based approach for least square estimation, AICHE Journal, V.72, 2209-2227

Rollins L. 1999. Robust Control Theory, Carnegie Mellon University

Rostan, M. 2008. *Industrial Ethernet Technologies Presentation*, EtherCAT, November 2008

Rubinstein H. 1978. Smoothing properties of Discrete Time zero-lag Kalman Filter, Automation, V.14, 387-401

Sadjadi B.A. 2003. Stability of Networked Control Systems in the presence of packet losses. Proceedings of the 43rd IEEE Conference on Decision and Control, 1205-1210. 2002

Schenato L. 2009. Distributed estimation and consensus. 3rd WIDE Phd School on

Networked Control Systems, Siena, Italy, July 7-9, 2009

Schenato L., Sinopoli B., Franceschetti M., Poolla K., and Sastry S. 2006 Optimal LQG Control over lossy packet network.

Schenato, L. Sinopoli, B. Franceschetti, M. Poolla, K. and Sastry, S.S. 2007. Foundations of control and estimation over lossy networks, *Proceedings of IEEE, Vol 95, No1, January 2007, p166*

Shi L., Xie L. and Murray R.M. 2009. Kalman filtering over a packet-delaying network: A probabilistic approach. Automatica, V.45, 2009: 2134-2140.

Shin J. Jang J and Kim J. 2009. The simulation and emulation verification that was based on NS-2, *IJCSNS International Journal of Computer Science and Network Security, VOL.9 No.3*, March 2009: 210-214.

Sierociuk D., Tejado I. and Vinagre B.M. 2010. Improved fractional Kalman filter and its application to estimation over lossy network. Signal Processing 91 (2011): 542-552

Sipahi R., Niculescu S., Abdallah C.T., Michiels W. and Gu K., 2011. Stability and stabilization of systems with time delay, limitations and opportunity. IEEE Control systems magazine. 38-65. February 2011.

Solberg J.J. 2009. Modelling random processes for engineers and managers. Purdue University. John Wiley & Sons, Inc. 2009

Tasdis D., Adonis N. and Hand D. 2008. Simulation and analysis of delay handling mechanism in sensor networks. Proceedings of X-th International Conference on Computer Modelling and Simulation, UKSIM 2008:661-666

Tatiraju S., Soroush M., and Ogunnaike. 1999. Multirate nonlinear state estimation with application to a polymerization reactor, AICHE Journal. V.75, 769-780

Tian Y., Yu Z. and Fidge C. 2007. Multifractal nature of network induced time delay in networked control systems. Physics Letters A 361 (2007). 103-107 September 2006. ScienceDirect

Tipsuwan, Y. and Chow, M. 2003. Control methodologies in networked control systems, Control Engineering Practice II, p1099-1111.

Vachhani P., Rengaswamy R., Gangwal V., and Narasimnan S. 2005. Recursive estimation in constrained nonlinear dynamical system, AICHE Journal, V.51 946-959

Van der Merwe R. 2004. Sigma-Point Kalman filters for probabilistic inference in dynamic state-space models. Dissertation, OGI School of Science & Engineering at Oregon Health & Science University. April 2004.

Vance, J. Vandoren. 1996. Examining the fundamentals of PID controls, *Control Engineering*, February 1996

Wang F. and Liu D., 2008. Networked Control System, *Theory and Application*s: Springer

Wang H and Zhang Y. 2009. LQ Control for network systems with input delay and packet

loss. Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference, 6858 -6863. December 2009.

Wang J. and Yang H. 2011. H_∞ control of a class of networked control systems with time delays and packet dropout. Applied Mathematics and Computation 217 (2011): 7469-7477.

Wang J. and Yang H. 2012. Exponential stability of a class of networked control systems with time delays and packet dropouts. Applied Mathematics and Computation 218 (2012) 8887-8894

Welch G. and Bishop G. 2006. An introduction to the Kalman Filter. UNC-Chapel Hill, TR 95-041, July 2006.

Williams R.L and Lawrence D.A. 2007. Linear state-space control systems. John Wiley and Sons, 26 Jan 2007: 242-248

Wong T. Interferometry in Radio Astronomy. Synthesis Workshop. May 2003

Xiong J and Lam J. 2006. Stabilization of linear systems over networks with bounded packet loss. Automatica 43 (2007). 80-87 July 2006. ScienceDirect.

Xiong K., Wei C.L. and Liu L.D. 2011. Robust Kalman filtering for discrete-time nonlinear systems with parameter uncertainties. Aerospace Science and Technology (2011). March 2011.

Xue D. Chen Y and Atherton D.P. 2007. Linear Feedback Control. 185:235

Yan H., Huang X., Wang M. And Zhang. 2006. Delay-dependent stability criteria for a class of networked control systems with multi-input and multi-output. Chaos, Solitons and Fractals xxx (2006). ScienceDirect

Yang W.Y., Chang T.G., Song I.H., Cho Y.S., Heo J., Jeon., Lee J.W. and Kim J.K. 2009. Signals and systems with Matlab. Springer. 361:379

Ye-Guo S. and Shi-Yin Q. 2011. Stability and stabilization of networked control systems with bounded packet dropout. Acta Automatica Sinica. Vol.37, No.1 January, 2011

Yi J., Wang Q., Zhao D. And Wen J.T. 2006. BP neural network prediction-based variable-period sampling approach for networked control system. Applied Mathematics and Computation 185 (2007). 976-988

Yonan J. 2003. The user-space VPN and open VPN, Understanding the user-space VPN-History, Conceptual Foundation, and Practical Usage. 2003

Zhang B. and Zheng W. 2012. H_∞ filter design for nonlinear networked control system with uncertain packet-loss probability. Signal Processing 92 (2012) 1499-1507

Zhang K., Rongli X. and Zhu Y. 2005. Optimal update with out of sequence measurement. IEEE Tran. Sig. Proc. V.53(6), 1992-2007

Zhang L and Huag-Jing F. 2005. A novel controller design and evaluation for networked control systems with time-variant delays. Journal of the Franklin Institute 343 (2006). 161-167 September 2005.ScienceDirect

Zhang L. and Hua-Jing F. 2008. A novel controller design and evaluation for networked control systems with time-variant delays. Journal of the Franklin Institute 343 (2006):161-167.

Zhang L., and Hristu-Varsakelis D. 2005. LQG Control under limited communication. 44th IEEE Conference on Decision and Control, and the European Control Conference. December 2005.

Zhang W and Yu L. 2007. Output Feedback Stabilization of Networked Control Systems With packet Dropouts. IEEE Transactions on Automatic Control, Vol. 52, No 9 1705-1710, September 2007

Zhang W. and Yu L. 2008. Modelling and control of networked control systems with both network-induced delay and packet-dropout. Automatica 44 (2008):3206-3210

Zhang X., Zheng Y. and Lu G. 2008. Stochastic stability of networked control systems with network-induced delay and data dropout. Journal Control Theory and Applications 2008 6 (4):405-409.

Zhou C., Du M. and Chen Q. 2012. Co-design of dynamic scheduling and H-infinity control for networked systems. Applied Mathematics and Computation 218 (2012) 10767-10775.

Zhu X. Soh Y.C. and Xie L. 2002. Robust Kalman Filter Design for discrete time-delay systems. Circuits Systems Signal Processing. Vol. 21, No. 3, 2002. 319:335

APPENDICES

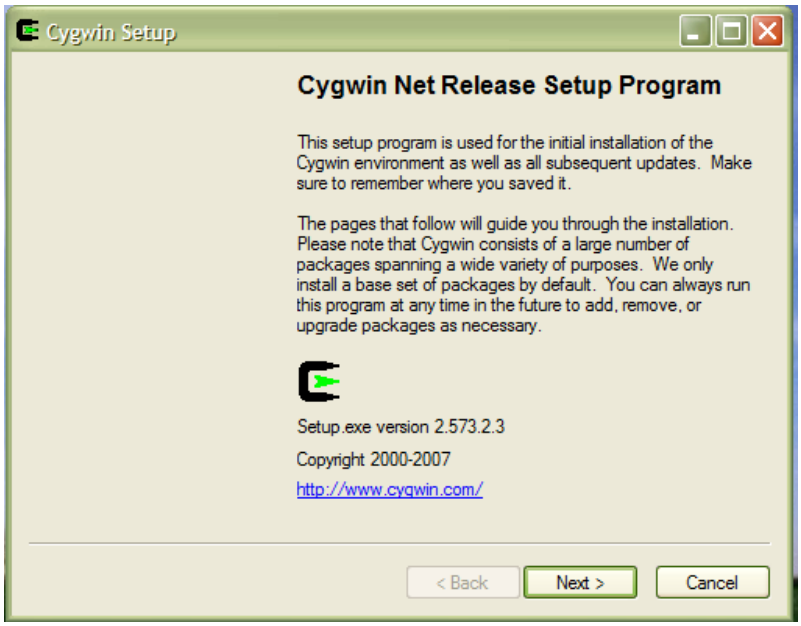
APPENDIX A: LabNS2 Installation procedure

Windows Installation Procedure

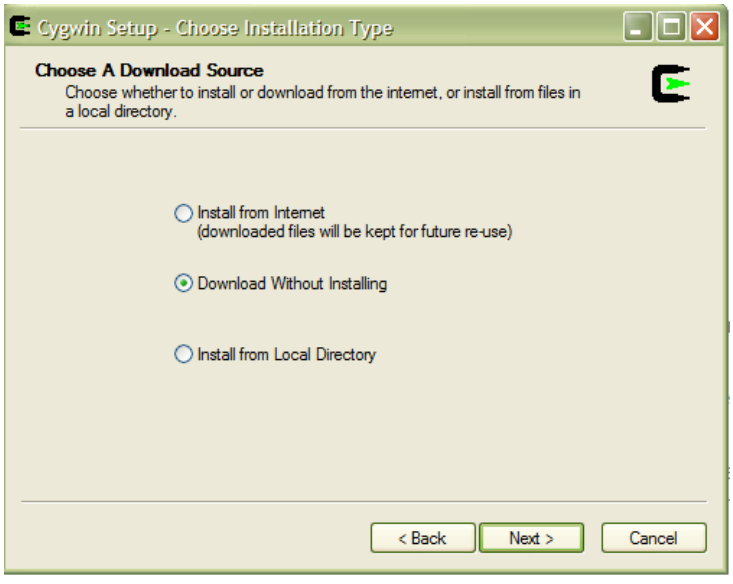
Use standard windows installation procedure.

Cygnus Installation Procedure

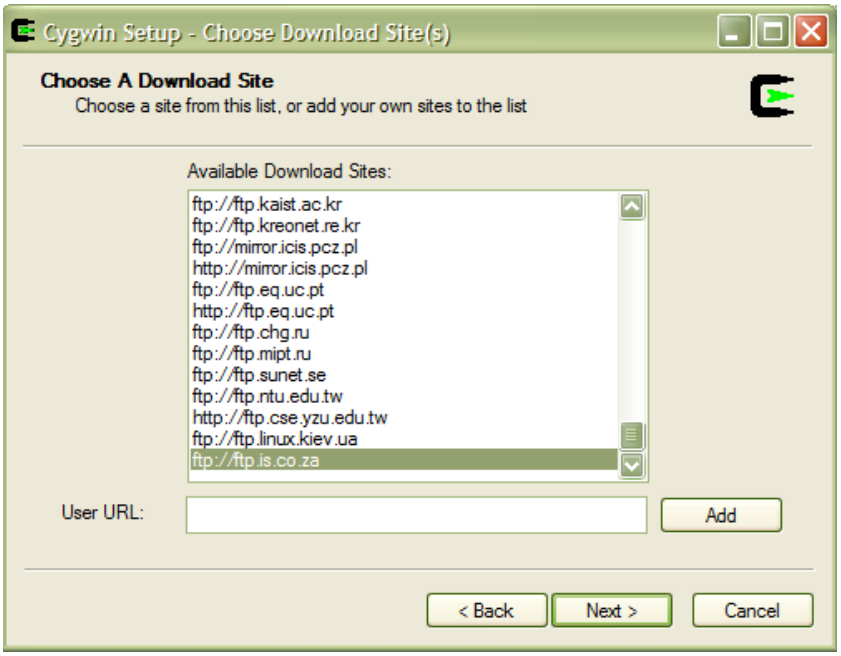
1. Download the setup file from http://www.cygwin.com/ and save it (C:\cygwin\setup)
2. Run Cygwin Setup from C:\cygwin\setup.



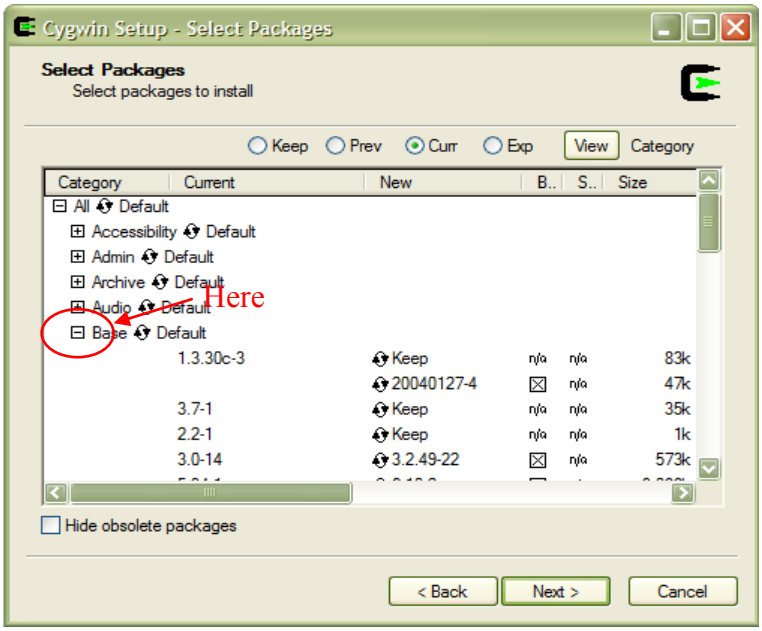
3. Click on "Next"



4. Select “Install from Internet “option if the internet connection is good.
5. OR select “Download Without Installing” option if there is more than one cygwin installation to be performed. At one stage "Choose Download Site(s)" a prompt for which source to use will pop out, choose a computer server in South Africa e.g. ftp://ftp.is.co.za. The downloaded file may be used for future installation to save on bandwidth downloads.
6. OR Select “Install from Local Directory” option if the mirror of cygwin is already downloaded.



7. . Click “Next” until you get to the “Select Packages” screen is reached.



8. Select the following packages and expand by clicking on “+” and collapse using “-“:
    a. Base , Devel, Editors, Util, Interpreters, X11
    b. Click on “Default” to “Install, Uninstall, or Reinstall a package.
9. “Cygwin” shortcut will be added on the desktop.
10. With this the install action of Cygwin is completed.

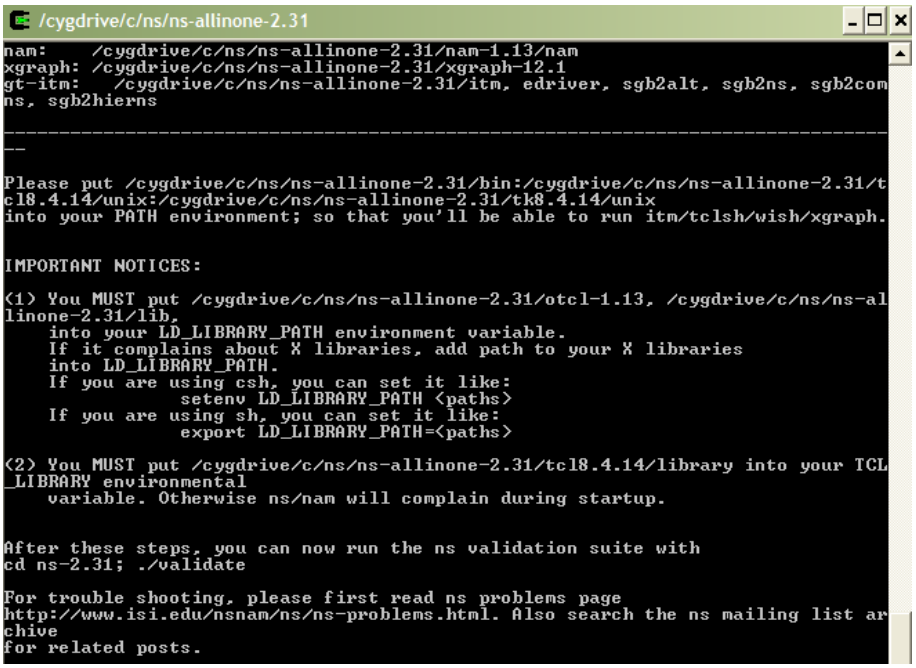
NS-2 Installation Procedure

1 Download ns-allinone-2.31.tar.gz from http://sourceforge.net/projects/nsnam/files/allinone/ns-allinone-2.31/ and save it (C:\ns\setup)

2 Run “cygwin” shortcut from the desktop.

3 From the command prompt execute the following commands:

- cd c:
- cd ns/setup
- tar -xzvf ns-allinone-2.31.tar.gz
- cd ns-allinone-2.31
- Read the ***README*** file
- ./install
- You might get an error message about missing gcc, xorg-X11 or xfree86-base, press Y and continue with the installation.
- Read through the important notice on how to set-up the library paths.



- Replace */cygdrive/c* with *c:*\ in windows environment
- Make sure that the LD_LIBRARY_PATH and TLC_LIBRARY environmental variables are set in System Variables entries. To do that in windows go to:
- **Control Panel, Systems, Advanced Tab, Environmental Variables**
- cd ns-2.31
- ./validate
- After validation when trying to execute an example script I found an error similar to this one:

```
210
0.0037499999999999999
running nam...
[root@DELL ex]# nam:
[code omitted because of length]
: no event type or button # or keysym
while executing
"bind Listbox <MouseWheel> {
%W yview scroll [expr {- (%D / 120) * 4}] units
}"
invoked from within
"if {[tk windowingsystem] eq "classic" || [tk windowingsystem] eq "aqua"} {
bind Listbox <MouseWheel> {
%W yview scroll [expr {- (%D)}] units
}
bind Li..."
```

- The solution to the error is found in:
- http://forums.fedoraforum.org/showthread.php?t=206795
- Download the patch file from the link above and copy it to:
- ns-allinone-2.31/tk8.4.14 directory
- Execute the patch using this command:
- patch –p1 < ./tk-8.4-lastevent.patch
- A prompt will pop up for file to patch (File to patch:)
- Type Generic/tk.h
- After the patch make sure that you change directory to ns-2.31 and execute ./install

4 To install the NS-2 Emulator, execute the following commands from command prompt:

- cd ~
- cd c:
- cd ns/ns-allinone-2.31/ns-2.31
- Edit the “Makefile” file and comment out “emulate/ether.o”.

- Execute the following command from command prompt:
- make nse
- To run the nse use the following command:
- ./nse ***<script name>***

APPENDIX B: MATLAB Script files -

APPENDIX B2.1:

```
% Probability Distribution Function (PDF) describing time delay between sensor and the controller and delay
% between controller and actuator.
% The function implements Nilson, 1998 PDF function.

% NS Mkondweni, 2012

a_ca = 0.016;       % Weight of the distribution
b_ca = 0.028;       % Weight of the distribution
a_sc = 0.017;       % Weight of the distribution
b_sc = 0.031;       % Weight of the distribution
pca = 0.9;      % Distribution
psc = 0.8;  % Distribution
Tscc = rand(1);

for i = 1:1000
            Tca = rand(1);
            if Tca <= pca
       Tca_c(1,i) = a_ca;
            else
       Tca_c(1,i) = b_ca;
   end
end

for i = 1:1000
            Tsc = rand(1);
            if Tsc <= psc
       Tscc = rand(1);
       Tsc_c(1,i) = Tscc*(a_sc-b_sc)+b_sc;
            else
       Tsc_c(1,i) = a_sc;
   end
end
T_sc = mean(Tsc_c);

subplot(2,1,1)
hist(Tca_c,50)
TITLE('Controller-to-Acumulator independent delay')
YLABEL('Number of delay occurances')
XLABEL('Magnitude of the delay')

T_ca = mean(Tca_c);
subplot(2,1,2)
hist(Tsc_c,50)

TITLE('Sensor-to-Controller independent delay')
YLABEL('Number of delay occurances')
XLABEL('Magnitude of the delay')
```

APPENDIX B5.1:

```
%                   tf_con_zncePI.m
% This simulation is performed considering Plant and Controller with
```

```
% constant network induced time delays in continuous form.
% The purpose of this simulation is to illustrate the influence of the network
% induced time delay in the case when the controller is not designed to cope with the delays

% PID Controller is used tuned using both 2nd Method of Ziegler-Nichols and Pole
% Placement.

% Parameters for Azimuth drive and Altitude drive
clear all
alt_az_param

% Simulations
pause on;
type=input('Select type of Controller 1=2nd Ziegler; 0=Pole Placement: ','s');
type=str2num(type);
if type == 1
% Calculation of PID parameters using pole placement method
M = 54.019;
aa = 1.96;
Pu = 1.4;
Ku = (4*M)/(pi*aa);      % gain

Kp = 0.6*Ku;
Ki = 0; %0.5*Pu;
Kd = 0.125*Pu;

nidca_az = 0;
nidsc_az = 0;
nidca_alt = 0;
nidsc_alt = 0;

for i = 1:4
   tspan = [0 30];
   [t,x,AZ,dish,ALT] = sim('tf_con',tspan,[]);
   subplot(2,2,1);
   title('Az TF using 2nd Meth. Ziegler')
   ylabel('Azimuth position, deg')
   xlabel('Continous Time')
   hold on
   grid on
   if i == 1
      plot(t,AZ,'k')
      subplot(2,2,2)
      title('ALT TF using 2nd Meth. Ziegler')
      ylabel('Altitude position, deg')
      xlabel('Continous Time')
      hold on
      grid on
      plot(t,ALT,'k')
      subplot(2,2,3)
      hold on
      grid on
      title('Dish using 2nd Meth. Ziegler')
      ylabel('Antenna position, deg')
      xlabel('Continous Time')
      plot(t,dish,'k')
      nidca_az = 0.5;
```

```
        nidsc_az = 0.5;
        nidca_alt = 0.5;
        nidsc_alt = 0.5;
    elseif i == 2
        plot(t,AZ,'k.-')
        subplot(2,2,2)
        hold on
        grid on
        plot(t,ALT,'k.-')
        subplot(2,2,3)
        hold on
        grid on
        plot(t,dish,'k.-')
        nidca_az = 1;
        nidsc_az = 1.5;
        nidca_alt = 2;
        nidsc_alt = 1.6;
    elseif i == 3
        plot(t,AZ,'b-.')
        subplot(2,2,2)
        hold on
        grid on
        plot(t,ALT,'b-.')
        subplot(2,2,3)
        hold on
        grid on
        plot(t,dish,'b-.')
        nidca_az = 1.3;
        nidsc_az = 1.7;
        nidca_alt = 3;
        nidsc_alt = 2.6;
    elseif i == 4
        plot(t,AZ,'r--')
        subplot(2,2,2)
        hold on
        grid on
        plot(t,ALT,'r--')
        subplot(2,2,3)
        hold on
        grid on
        plot(t,dish,'r--')
    end
end
legend('\tau_T = 0','0 < \tau_T < \tau_c', '\tau_T=\tau_c', '\tau_T > \tau_c',-1)
elseif type == 0
% The controller is designed by using Pole Placement Method via Transfer Function

PO = 10; % Desired Percentage Overshoot
Ts = 5; % Desired Settling Time
Z = abs(log(PO/100))/sqrt(((pi()^2)+(log(PO/100)^2))) % Damping Ratio calculated from the PO

Wd = 4/(Ts*Z); % Undamped natural frequency calculated from Settling Time (Ts)
a = sqrt(Kaz)*10;                    % Third pole

% Calculation of PID parameters using Pole Placement method
Kp = ((Wd^2) + (2*Z*Wd*a))/Kaz;
Ki = 0; %(a*(Wd^2))/Kaz;
```

```
Kd = ((2*Z*Wd)+a-alp_az)/Kaz;

nidca_az = 0;
nidsc_az = 0;
nidca_alt = 0;
nidsc_alt = 0;

for i = 1:4
   tspan = [0 30];
   [t,x,AZ,dish,ALT] = sim('tf_con',tspan,[]);
   subplot(2,2,1);
   title('Az Pole Placement')
   ylabel('Azimuth position, deg')
   xlabel('Continous Time')
   hold on
   grid on

   if i == 1
      plot(t,AZ,'k')
      subplot(2,2,2)
      title('ALT Pole Placement')
      ylabel('Altitude position, deg')
      xlabel('Continous Time')
      hold on
      grid on
      plot(t,ALT,'k')
      subplot(2,2,3)
      hold on
      grid on
      TITLE('Dish using Pole Placement')
      ylabel('Antenna position, deg')
      xlabel('Continous Time')
      plot(t,dish,'k')
      nidca_az = 0.5;
      nidsc_az = 0.7;
      nidca_alt = 0.7;
      nidsc_alt = 0.8;
   elseif i == 2
      plot(t,AZ,'k.-')
      subplot(2,2,2)
      hold on
      grid on
      plot(t,ALT,'k.-')
      subplot(2,2,3)
      hold on
      grid on
      plot(t,dish,'k.-')
      nidca_az = 1.1;
      nidsc_az = 1.05;
      nidca_alt = 1.3;
      nidsc_alt = 1.5;
   elseif i ==3
      plot(t,AZ,'b-.')
      subplot(2,2,2)
      hold on
      grid on
      plot(t,ALT,'b-.')
```

```
        subplot(2,2,3)
        hold on
        grid on
        plot(t,dish,'b-.')
        nidca_az = 1.5;
        nidsc_az = 1.05;
        nidca_alt = 1.7;
        nidsc_alt = 1.4;
    elseif i == 4
        plot(t,AZ,'r--')
        subplot(2,2,2)
        hold on
        grid on
        plot(t,ALT,'r--')
        subplot(2,2,3)
        hold on
        grid on
        plot(t,dish,'r--')
    end
end
legend('\tau_T = 0','0 < \tau_T < \tau_c', '\tau_T=\tau_c', '\tau_T > \tau_c',-1)
else
    break;
end
```

APPENDIX B5.2:

```
%                       ss_con_ppP.m
% This m-file simulates a closed loop control system using constant netowrk induced time delay,
% The Proportional State Space Controller  is designed by using Pole Placement Method .
% The NITD is considered as part of a control system forming NCS:

% Parameters for Azimuth drive and Altitude drive
clear all
alt_az_param;


PO = 5; % Desired Percentage Overshoot
Z = abs(log(PO/100))/sqrt(((pi()^2)+(log(PO/100)^2))); % Damping Ratio calculated from the PO
Wd =(4/2*Z)*sqrt(1-Z^2);

h1az = Wd^2;
h2az = ((2*Z*Wd)-alp_az);
h1alt = Wd^2;
h2alt = ((2*Z*Wd)-alp_alt);

nidca_az = 0;
nidsc_az = 0;
nidca_alt = 0;
nidsc_alt = 0;

for i = 1:4
    tspan = [0 30];
    [t,x,AZ,dish,ALT] = sim('s_con_ppP',tspan,[]);
    subplot(2,2,1);
    title('Az Proportional State Space')
    ylabel('Azimuth position, deg')
```

```
xlabel('Continous Time')
hold on
grid on
if i == 1
    plot(t, AZ, 'b')
    subplot(2,2,2)
    [rtppP_d1 stppP_d1 over_ppP_d1 sseppP_d1] = perform_index(t, AZ)
    title('ALT Proportional State Space')
    ylabel('Altitude position, deg')
    xlabel('Continous Time')
    hold on
    grid on
    plot(t,ALT, 'b')
    subplot(2,2,3)
    hold on
    grid on
    title('Dish Proportional State Space')
    ylabel('Antenna position, deg')
    xlabel('Continous Time')
    plot(t,dish, 'b')
    nidca_az = 0.4;
    nidsc_az = 0.6;
    nidca_alt = 0.5;
    nidsc_alt = 0.3;
elseif i == 2
    plot(t, AZ,'g.')
    subplot(2,2,2)
    hold on
    grid on
    plot(t,ALT,'g.')
    subplot(2,2,3)
    hold on
    grid on
    plot(t,dish,'g.')
    nidca_az = 0.9;
    nidsc_az = 1;
    nidca_alt = 0.82;
    nidsc_alt = 0.9;
elseif i == 3
    plot(t, AZ,'k.-')
    subplot(2,2,2)
    hold on
    grid on
    plot(t,ALT,'k.-')
    subplot(2,2,3)
    hold on
    grid on
    plot(t,dish,'k.-')
    nidca_az = 1.1;
    nidsc_az = 1.05;
    nidca_alt = 1.1;
    nidsc_alt = 1.03;
elseif i == 4
    plot(t, AZ,'r--')
    subplot(2,2,2)
    hold on
    grid on
```

```
        plot(t,ALT,'r--')
        subplot(2,2,3)
        hold on
        grid on
        plot(t,dish,'r--')
    end
end
legend('\tau_T = 0','0 < \tau_T < \tau_c', '\tau_T=\tau_c', '\tau_T > \tau_c',-1)
```

APPENDIX B5.3:

```
%                       tf_con_zncePI.m
% This simulation is performed considering Plant and Controller with
% constant network induced time delays in continuous form.
% The purpose of this simulation is to illustrate the influence of the network
% induced time delay in the case when the controller is not designed to cope with the delays

% PID Controller is used tuned using both 2nd Method of Ziegler-Nichols and Pole
% Placement.

% Author.  N Mkondweni.  March 2011.

% Parameters for Azimuth drive and Altitude drive
clear all
alt_az_param

% Simulations
pause on;
type=input('Select type of Controller 1=2nd Ziegler; 0=Pole Placement: ','s');
type=str2num(type);
if type == 1
% Calculation of PID parameters using pole placement method
M = 54.019;
aa = 1.96;
Pu = 1.4;
Ku = (4*M)/(pi*aa);      % gain

Kp = 0.6*Ku;
Ki = 0; %0.5*Pu;
Kd = 0.125*Pu;

nidca_az = 0;
nidsc_az = 0;
nidca_alt = 0;
nidsc_alt = 0;

for i = 1:4
    tspan = [0 30];
    [t,x,AZ,dish,ALT] = sim('tf_con',tspan,[]);
    subplot(2,2,1);
    title('Az TF using 2nd Meth. Ziegler')
    ylabel('Azimuth position, deg')
    xlabel('Continous Time')
    hold on
    grid on
    if i == 1
```

```
    plot(t,AZ,'k')
    subplot(2,2,2)
    [rtppP_d1 stppP_d1 over_ppP_d1 sseppP_d1] = perform_index(t,AZ)
    title('ALT TF using 2nd Meth. Ziegler')
    ylabel('Altitude position, deg')
    xlabel('Continous Time')
    hold on
    grid on
    plot(t,ALT,'k')
    subplot(2,2,3)
    hold on
    grid on
    title('Dish using 2nd Meth. Ziegler')
    ylabel('Antenna position, deg')
    xlabel('Continous Time')
    plot(t,dish,'k')
    nidca_az = 0.5;
    nidsc_az = 0.7;
    nidca_alt = 0.7;
    nidsc_alt = 0.8;
  elseif i == 2
    plot(t,AZ,'k.-')
    subplot(2,2,2)
    hold on
    grid on
    plot(t,ALT,'k.-')
    subplot(2,2,3)
    hold on
    grid on
    plot(t,dish,'k.-')
    nidca_az = 1;
    nidsc_az = 1.5;
    nidca_alt = 2;
    nidsc_alt = 1.6;
  elseif i == 3
    plot(t,AZ,'b-.')
    subplot(2,2,2)
    hold on
    grid on
    plot(t,ALT,'b-.')
    subplot(2,2,3)
    hold on
    grid on
    plot(t,dish,'b-.')
    nidca_az = 1.3;
    nidsc_az = 1.7;
    nidca_alt = 3;
    nidsc_alt = 2.6;
  elseif i == 4
    plot(t,AZ,'r--')
    subplot(2,2,2)
    hold on
    grid on
    plot(t,ALT,'r--')
    subplot(2,2,3)
    hold on
    grid on
```

```
        plot(t,dish,'r--')
    end
end
legend('\tau_T = 0','0 < \tau_T < \tau_c', '\tau_T=\tau_c', '\tau_T > \tau_c',-1)
elseif type == 0
% The controller is designed by using Pole Placement Method via Transfer Function

PO = 10; % Desired Percentage Overshoot
Ts = 5; % Desired Settling Time
Z = abs(log(PO/100))/sqrt(((pi()^2)+(log(PO/100)^2))) % Damping Ratio calculated from the PO

Wd = 4/(Ts*Z); % Undamped natural frequency calculated from Settling Time (Ts)
a = sqrt(Kaz)*10;                    % Third pole

% Calculation of PID parameters using Pole Placement method
Kp = ((Wd^2) + (2*Z*Wd*a))/Kaz;
Ki = 0; %(a*(Wd^2))/Kaz;
Kd = ((2*Z*Wd)+a-alp_az)/Kaz;

nidca_az = 0;
nidsc_az = 0;
nidca_alt = 0;
nidsc_alt = 0;

for i = 1:4
    tspan = [0 30];
    [t,x,AZ,dish,ALT] = sim('tf_con',tspan,[]);
    subplot(2,2,1);
    title('Az Pole Placement')
    ylabel('Azimuth position, deg')
    xlabel('Continous Time')
    hold on
    grid on

    if i == 1
        plot(t,AZ,'k')
        subplot(2,2,2)
        [rtppP_d1 stppP_d1 over_ppP_d1 sseppP_d1] = perform_index(t,AZ)
        title('ALT Pole Placement')
        ylabel('Altitude position, deg')
        xlabel('Continous Time')
        hold on
        grid on
        plot(t,ALT,'k')
        subplot(2,2,3)
        hold on
        grid on
        TITLE('Dish using Pole Placement')
        ylabel('Antenna position, deg')
        xlabel('Continous Time')
        plot(t,dish,'k')
        nidca_az = 0.5;
        nidsc_az = 0.7;
        nidca_alt = 0.7;
        nidsc_alt = 0.8;
    elseif i == 2
        plot(t,AZ,'k.-')
```

```
        subplot(2,2,2)
        hold on
        grid on
        plot(t,ALT,'k.-')
        subplot(2,2,3)
        hold on
        grid on
        plot(t,dish,'k.-')
        nidca_az = 1.1;
        nidsc_az = 1.05;
        nidca_alt = 1.3;
        nidsc_alt = 1.5;
    elseif i ==3
        plot(t,AZ,'b-.')
        subplot(2,2,2)
        hold on
        grid on
        plot(t,ALT,'b-.')
        subplot(2,2,3)
        hold on
        grid on
        plot(t,dish,'b-.')
        nidca_az = 1.5;
        nidsc_az = 1.05;
        nidca_alt = 1.7;
        nidsc_alt = 1.4;
    elseif i == 4
        plot(t,AZ,'r--')
        subplot(2,2,2)
        hold on
        grid on
        plot(t,ALT,'r--')
        subplot(2,2,3)
        hold on
        grid on
        plot(t,dish,'r--')
    end
end
legend('\tau_T = 0','0 < \tau_T < \tau_c', '\tau_T=\tau_c', '\tau_T > \tau_c',-1)
else
    break;
end
```

APPENDIX B5.4:

```
%                       ss_con_ppPI.m
% This m-file simulates a closed loop control system using constant NITD,
% Proportional and Integral State Space controller is used
% The NITD is consodered as part of a control system forming NCS:
% The NITD is consodered as part of a control system forming NCS:

%Author.  N Mkondweni.  August 2011.

clear all;
alt_az_param;

PO = 5; % Desired Percentage Overshoot
```

```
Z = abs(log(PO/100))/sqrt(((pi()^2)+(log(PO/100)^2))) % Damping Ratio calculated from the PO
Wd =(4/2*Z)*sqrt(1-Z^2);

a_az = sqrt(Kaz)*1.5;                    % Third pole
a_alt = sqrt(Kalt)*1.5;                  % Third pole
% The controller is designed by using Pole Placement Method via State Space

h1az = ((Wd^2)+2*Z*Wd*a_az);
h2az = ((2*Z*Wd)+a_az-alp_az);
h3az = a_az*(Wd^2)/Kaz;

h1alt = ((Wd^2)+2*Z*Wd*a_alt);
h2alt = ((2*Z*Wd)+a_alt-alp_alt);
h3alt = a_alt*(Wd^2)/Kalt;


nidca_az = 0;
nidsc_az = 0;
nidca_alt = 0;
nidsc_alt = 0;

for i = 1:4
   tspan = [0 30];
   [t,x,AZ,dish,ALT] = sim('s_con_ppPI_0',tspan,[]);
   subplot(2,2,1);
   title('Az Proportional State Space')
   ylabel('Azimuth position, deg')
   xlabel('Continous Time')
   hold on
   grid on
   if i == 1
      plot(t, AZ, 'b')
      subplot(2,2,2)
      [rtppP_d1 stppP_d1 over_ppP_d1 sseppP_d1] = perform_index(t, AZ)
      title('ALT Proportional State Space')
      ylabel('Altitude position, deg')
      xlabel('Continous Time')
      hold on
      grid on
      plot(t,ALT, 'b')
      subplot(2,2,3)
      hold on
      grid on
      title('Dish Proportional State Space')
      ylabel('Antenna position, deg')
      xlabel('Continous Time')
      plot(t,dish, 'b')
      nidca_az = 0.4;
      nidsc_az = 0.6;
      nidca_alt = 0.5;
      nidsc_alt = 0.3;
   elseif i == 2
      plot(t, AZ,'g.')
      subplot(2,2,2)
      hold on
      grid on
      plot(t,ALT,'g.')
```

```
        subplot(2,2,3)
        hold on
        grid on
        plot(t,dish,'g.')
        nidca_az = 0.9;
        nidsc_az = 1;
        nidca_alt = 0.82;
        nidsc_alt = 0.9;
    elseif i == 3
        plot(t, AZ,'k.-')
        subplot(2,2,2)
        hold on
        grid on
        plot(t,ALT,'k.-')
        subplot(2,2,3)
        hold on
        grid on
        plot(t,dish,'k.-')
        nidca_az = 1.1;
        nidsc_az = 1.05;
        nidca_alt = 1.1;
        nidsc_alt = 1.03;
    elseif i == 4
        plot(t, AZ,'r--')
        subplot(2,2,2)
        hold on
        grid on
        plot(t,ALT,'r--')
        subplot(2,2,3)
        hold on
        grid on
        plot(t,dish,'r--')
    end
end
legend('\tau_T = 0','0 < \tau_T < \tau_c', '\tau_T=\tau_c', '\tau_T > \tau_c',-1)
```

APPENDIX B8.1:

Exp_one.m

```
% This m.file simulates a open loop structure where plant and the DKF are
% considered without consideration for network induced time delays.
% The purpose of this procedure is to verify the functionality of the
% filter and confirm that it does track the step response
% of the open loop discrete plant.

dt = 0.02;
A = [1 0.0921;0  0.8462];
B = [0.005;0.092];
C = [1 0];
D = 0;

% State and Measuremnt noise coefficients
Q = 0.001;
R = 0.02;
x0 = [0;0];
```

```
% For Kalman filter, we use the same model
A1 = A;
B1 = B;
C1 = C;
D1 = D;
Q1 = diag([0;1]);
R1 = 1;
x1 = [0;0];     %but with different initial state estimate
P1 = eye(2);

% Plotting settings
tspan = [0 10];
r=1;
[t,x,y,y1] = sim('Exp_1',tspan,[],[0 r]);
plot(t,y,'-b',t,y1,'--k')
grid on
legend('y(k): Plant','y^-(k):Filter',-1)
ylabel('Amplitude')
xlabel('Discrete Time')
```

APPENDIX B8.2:

Exp_two.m

```
% This simulation procedure is performed using Plant, Controller
% and DKF considering constant network induced time delays. The purpose of
% this procedure is to illustrate the influence of network induced time delay
% considering a plant, controller and DKF that are not designed to cope with
% this delay. An option to use either Pole Placement controller design method
% OR LQ controller design method is considered using the same parameters used
% in simulation procedure 2.

% Author: NS Mkondweni, November 2011
clear all;
A = [1 0.1;0  0.85];
B = [0.005;0.1];
C = [1 0];
D = 0;

% State and Measuremnt noise coefficients

Q = 0; %0.001;
R = 0; %0.02;
x0 = [0;0];

% Select the type of controller to use i.e. Pole Placement OR LQG

pause on;
type=input('Select type of Controller 1=Pole Placement; 0=LGQ: ','s');
type=str2num(type);
if type == 1
   % The controller is designed by using Pole Placement Method via State Space
   PO = 10; % Desired Percentage Overshoot
   Ts = 5; % Desired Settling Time
   zeta = abs(log(PO/100))/sqrt(((pi()^2)+(log(PO/100)^2))); % Damping Ratio
```

```
    Wd =(4/Ts*zeta)*(sqrt(1-zeta^2)); % Undamped natural frequency
    dt = 0.02;
    po = [1 (2*zeta*Wd) (Wd^2)];
    rt = roots(po);
    j = imag(rt(1,1));

    p = [-4-j*i -4+j*i];
    z1 = exp(p(1,1)*dt);
    z2 = exp(p(1,2)*dt);
    pz = [z1 z2];
    K = place(A,B,pz);  % Gain matrix using Pole Placement
    type1='Pole Placement';
elseif type == 0
    % The controller is designed by using LQG via State Space
    QQ = diag([0.325;1]);
    RR = 1;
    H = dlqr(A,B,QQ,RR);
    type1='LQG';
else
    break;
end

% For Kalman filter, we use the same model
A1 = A;
B1 = B;
C1 = C;
D1 = D;
Q1 = diag([0;1]);
R1 = 1;
x1 = [0;0];     %but with different initial state estimate
P1 = eye(2);

tspan = [0 20];

[t,x,y,y1,y2,y3] = sim('Exp_2',tspan,[]);
plot(t,y,'-b',t,y1,'--k',t,y2,':g',t,y3,'-.r')
grid on
title(['Controller, Plant and DKF using: ',type1]);
legend('y(k): Plant','y^-(k): Filter', 'u(k): Control', 'r(k): Reference',-1)
ylabel('Amplitude')
xlabel('Discrete Time')
```

APPENDIX B8.3:

Exp_three.m

```
%                       BOTH DELAYS CONSIDERED
% This simulation procedure is performed using Plant, Controller
% and DKF considering constant network induced time delays, between controller
% and actuator and bewteen the sensor and the controller. The purpose of this procedure
% is to illustrate the influence of network induced time delay
% considering a plant, controller and DKF that are not designed to cope with
% this delay. The Pole Placement controller design method is considered in
% this experiment using the same parameters used in simulation procedure 2.

% Author: NS Mkondweni, November 2011
```

```
clear all;
A = [1 0.1;0  0.85];
B = [0.005;0.1];
C = [1 0];
D = 0;

r =0;
% State and Measuremnt noise coefficients
Q = 0; %0.001;
R = 0; %0.02;
x0 = [0;0];

% The controller is designed by using Pole Placement Method via State Space
PO = 10; % Desired Percentage Overshoot
Ts = 5; % Desired Settling Time
zeta = abs(log(PO/100))/sqrt(((pi()^2)+(log(PO/100)^2))); % Damping Ratio

Wd =(4/Ts*zeta)*(sqrt(1-zeta^2)); % Undamped natural frequency
dt = 0.02;
po = [1 (2*zeta*Wd) (Wd^2)];
rt = roots(po);
j = imag(rt(1,1));

p = [-4-j*i -4+j*i];
z1 = exp(p(1,1)*dt);
z2 = exp(p(1,2)*dt);
pz = [z1 z2];
H = place(A,B,pz);  % Gain matrix using Pole Placement

% For Kalman filter, we use the same model
A1 = A;
B1 = B;
C1 = C;
D1 = D;
Q1 = diag([0;1]);
R1 = 1;
x1 = [0;0];     %but with different initial state estimate
P1 = eye(2);

nid_sc = 1;
nid_ca = 1;
hold on
grid on
for i=1:4
   tspan = [0 10];
   r=1;
   [t,x,y,y1,y2] = sim('Exp_3',tspan,[],[0 r]);
   ylabel('Amplitude')
   xlabel('Discrete Time')
   if i == 1
      grid on
      grid on
      subplot(2,2,1)
      plot(t,y,'-b',t,y1,'--k',t,y2,':r')
      title(['Delay < Sampling Period : \tau^s^c = ',num2str(nid_sc),...
         ',\tau^c^a =  ',num2str(nid_ca),' samples, ',...
```

```
            num2str((nid_sc+nid_ca)*dt),'sec'])
        nid_sc = 5;
        nid_ca = 5;
    elseif i == 2
        grid on
        subplot(2,2,2)
        plot(t,y, '-b',t,y1,'--k',t,y2,':r')
        title(['Delay > Sampling Period : \tau^s^c = ',num2str(nid_sc),...
            ',\tau^c^a =  ',num2str(nid_ca),' samples, ',...
            num2str((nid_sc+nid_ca)*dt),'sec'])
        nid_sc = 20;
        nid_ca = 15;
    elseif i == 3
        grid on
        subplot(2,2,3)
        plot(t,y, '-b',t,y1,'--k',t,y2,':r')
        title(['Critical Delay is: \tau^s^c = ',num2str(nid_sc),...
            ',\tau^c^a =  ',num2str(nid_ca),' samples, ',...
            num2str((nid_sc+nid_ca)*dt),'sec'])
        nid_sc = 25;
        nid_ca = 25;
    elseif i == 4
        grid on
        subplot(2,2,4)
        plot(t,y, '-b',t,y1,'--k',t,y2,':r')
     end
end
grid on
legend('y(k): Plant','y^-(k): Filter', 'u(k): Control',-1)
title(['Delay > Ctrical Delay: \tau^s^c = ',num2str(nid_sc),...
    ',\tau^c^a =  ',num2str(nid_ca),' samples, ',...
    num2str((nid_sc+nid_ca)*dt),'sec'])
ylabel('Amplitude')
xlabel('Discrete Time')
```

APPENDIX B8.4:

Exp_four.m

```
%                CONTROLLER AND ACTUATOR DELAY CONSIDERED
% This simulation procedure is performed using Plant, Controller and
% DKF considering constant network induced time delays. This simulation
% procedure is similar to simulation procedure 3, the difference is that
% in this simulation the delays are considered between controller and
% actuator ONLY.

% Author: NS Mkondweni, November 2011

clear all;
A = [1 0.1;0  0.85];
B = [0.005;0.1];
C = [1 0];
D = 0;

r =0;
% State and Measuremnt noise coefficients
Q = 0.001;
```

```
R = 0.02;
x0 = [0;0];

% The controller is designed by using Pole Placement Method via State Space
PO = 10; % Desired Percentage Overshoot
Ts = 5; % Desired Settling Time
zeta = abs(log(PO/100))/sqrt(((pi()^2)+(log(PO/100)^2))); % Damping Ratio

Wd =(4/Ts*zeta)*(sqrt(1-zeta^2)); % Undamped natural frequency
dt = 0.02;
po = [1 (2*zeta*Wd) (Wd^2)];
rt = roots(po);
j = imag(rt(1,1));

p = [-4-j*i -4+j*i];
z1 = exp(p(1,1)*dt);
z2 = exp(p(1,2)*dt);
pz = [z1 z2];
H = place(A,B,pz);  % Gain matrix using Pole Placement

% For Kalman filter, we use the same model
A1 = A;
B1 = B;
C1 = C;
D1 = D;
Q1 = diag([0;1]);
R1 = 1;
x1 = [0;0];     %but with different initial state estimate
P1 = eye(2);

nid_ca = 1;
hold on
grid on
for i=1:4
   tspan = [0 10];
   r=1;
   [t,x,y,y1,y2] = sim('Exp_4',tspan,[],[0 r]);
   ylabel('Amplitude')
   xlabel('Discrete Time')
   if i == 1
      grid on
      grid on
      subplot(2,2,1)
      plot(t,y,'-b',t,y1,'--k',t,y2,':r')
      title(['Delay < Sampling Period : \tau^c^a = ',num2str(nid_ca),...
          ' samples, ', num2str(nid_ca*dt),'sec'])
      nid_ca = 10;
   elseif i == 2
      grid on
      subplot(2,2,2)
      plot(t,y, '-b',t,y1,'--k',t,y2,':r')
      title(['Delay > Sampling Period : \tau^c^a = ',num2str(nid_ca),...
          ' samples, ', num2str(nid_ca*dt),'sec'])
      nid_ca = 35;
   elseif i == 3
      grid on
      subplot(2,2,3)
```

```
        plot(t,y, '-b',t,y1,'--k',t,y2,':r')
        title(['Critical Delay is: \tau^c^a = ',num2str(nid_ca),...
            ' samples, ', num2str(nid_ca*dt),'sec'])
        nid_ca = 50;
    elseif i == 4
        grid on
        subplot(2,2,4)
        plot(t,y, '-b',t,y1,'--k',t,y2,':r')
      end
end
grid on
legend('y(k): Plant','y^-(k): Filter', 'u(k): Control',-1)
title(['Delay < Sampling Period : \tau^c^a = ',num2str(nid_ca),...
            ' samples, ', num2str(nid_ca*dt),'sec'])
ylabel('Amplitude')
xlabel('Discrete Time')
```

APPENDIX B8.5:

Exp_five.m

```
%                SENSOR CONTROLLER DELAY CONSIDERED
% This simulation procedure is performed usingPlant, Controller and
% DKF considering constant network induced time delays. This simulation
% procedure is similar to simulation procedure 3, the difference is that
% in this simulation the delays are considered between sensor and
% controller ONLY.

clear all;
A = [1 0.1;0  0.85];
B = [0.005;0.1];
C = [1 0];
D = 0;

r =0;
% State and Measuremnt noise coefficients
Q = 0; %0.001;
R = 0; % 0.02;
x0 = [0;0];

% The controller is designed by using Pole Placement Method via State Space
PO = 10; % Desired Percentage Overshoot
Ts = 5; % Desired Settling Time
zeta = abs(log(PO/100))/sqrt(((pi()^2)+(log(PO/100)^2))); % Damping Ratio

Wd =(4/Ts*zeta)*(sqrt(1-zeta^2)); % Undamped natural frequency
dt = 0.02;
po = [1 (2*zeta*Wd) (Wd^2)];
rt = roots(po);
j = imag(rt(1,1));

p = [-4-j*i -4+j*i];
z1 = exp(p(1,1)*dt);
z2 = exp(p(1,2)*dt);
pz = [z1 z2];
H = place(A,B,pz);  % Gain matrix using Pole Placement
```

```
% For Kalman filter, we use the same model
A1 = A;
B1 = B;
C1 = C;
D1 = D;
Q1 = diag([0;1]);
R1 = 1;
x1 = [0;0];     %but with different initial state estimate
P1 = eye(2);

nid_sc = 1;
hold on
grid on
for i=1:4
   tspan = [0 10];
   r=1;
   [t,x,y,y1,y2] = sim('Exp_5',tspan,[],[0 r]);
   ylabel('Amplitude')
   xlabel('Discrete Time')
   if i == 1
      grid on
      grid on
      subplot(2,2,1)
      plot(t,y,'-b',t,y1,'--k',t,y2,':r')
      title(['Delay < Sampling Period : \tau^s^c = ',num2str(nid_sc),...
          ' samples, ', num2str(nid_sc*dt),'sec'])
      nid_sc = 10;
   elseif i == 2
      grid on
      subplot(2,2,2)
      plot(t,y, '-b',t,y1,'--k',t,y2,':r')
      title(['Delay > Sampling Period : \tau^s^c = ',num2str(nid_sc),...
          ' samples, ', num2str(nid_sc*dt),'sec'])
      nid_sc = 35;
   elseif i == 3
      grid on
      subplot(2,2,3)
      plot(t,y, '-b',t,y1,'--k',t,y2,':r')
      title(['Critical Delay is: \tau^s^c = ',num2str(nid_sc),...
          ' samples, ', num2str(nid_sc*dt),'sec'])
      nid_sc = 50;
   elseif i == 4
      grid on
      subplot(2,2,4)
      plot(t,y, '-b',t,y1,'--k',t,y2,':r')
    end
end
grid on
legend('y(k): Plant','y^-(k): Filter', 'u(k): Control',-1)
title(['Delay < Sampling Period : \tau^s^c = ',num2str(nid_sc),...
          ' samples, ', num2str(nid_sc*dt),'sec'])
ylabel('Amplitude')
xlabel('Discrete Time')
```

APPENDIX B8.6:

Exp_six.m

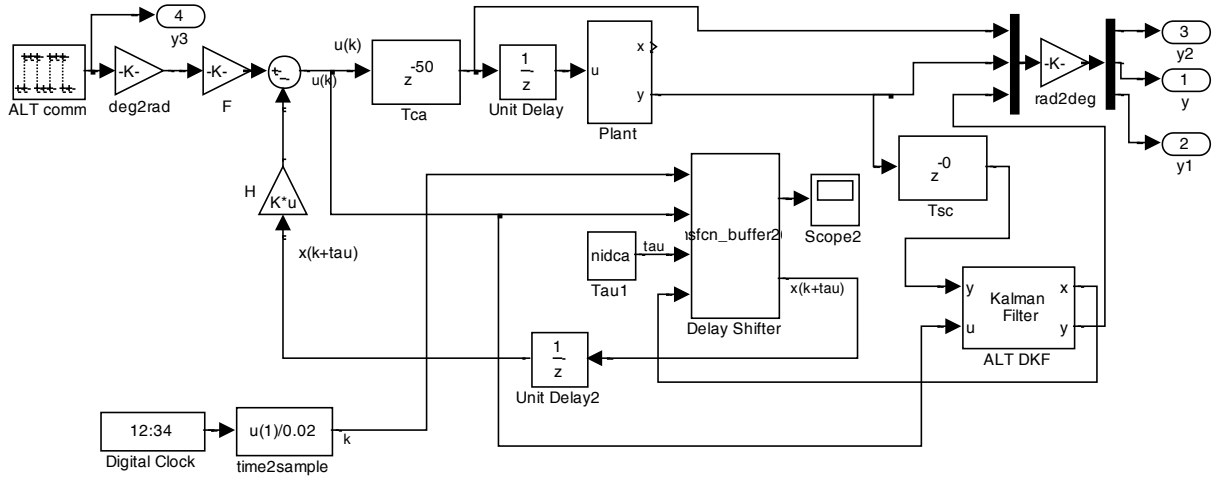
```
%               DEVELOPED CONTROL METHODS
% This simulation procedure is performed using Plant, Robust Predictive Controller, and
% Kalman fiter considering constant network induced time delays.
% The delays are considered between controller and actuator ONLY.

% Author: NS Mkondweni, September 2012

clear all;
A = [1 0.1;0  0.85];
B = [0.005;0.1];
C = [1 0];
D = 0;

% State and Measurement noise coefficients
Q = 0; %0.001;
R = 0; %0.02;
x0 = [0;0];

% The controller is designed by using Pole Placement Method via State Space
PO = 10; % Desired Percentage Overshoot
Ts = 5; % Desired Settling Time
zeta = abs(log(PO/100))/sqrt(((pi()^2)+(log(PO/100)^2))); % Damping Ratio

Wd =(4/Ts*zeta)*(sqrt(1-zeta^2)); % Undamped natural frequency
dt = 0.02;
po = [1 (2*zeta*Wd) (Wd^2)];
rt = roots(po);
j = imag(rt(1,1));

p = [-4-j*i -4+j*i];
z1 = exp(p(1,1)*dt);
z2 = exp(p(1,2)*dt);
pz = [z1 z2];
L = place(A,B,pz)  % Gain matrix using Pole Placement

nidca = 35;

grid on
hold on
tspan = [0 20];
[t,x,y,y1,y2] = sim('Exp_6',tspan,[]);
ylabel('Antenna position, deg')
xlabel('Discrete Time')
hold on
grid on
%subplot(2,2,4)
plot(t,y,'-b')
hold on
grid on
stairs(t,y1,'-.r')
hold on
grid on
stairs(t,y2,':g')
hold on
```

```
grid on
title(['Considered Delay: \tau^c^a = ',num2str(nidca),...
        ' samples,', num2str(nidca*dt),'sec'])
legend('y(k): Plant Output', 'r(k): Reference','u(k): Control Output',-1)
ylabel('Antenna position, deg')
xlabel('Discrete Time')

function msfcn_method(block)
% S-Function to implement the shifting
% Author: NS Mkondweni, May 2012

setup(block);

%endfunction

function setup(block)

  %% Register number of input and output ports
  block.NumInputPorts  = 4;
  block.NumOutputPorts = 2;

  %% Setup functional port properties to dynamically
  %% inherited.
  block.SetPreCompInpPortInfoToDynamic;
  block.SetPreCompOutPortInfoToDynamic;

  block.InputPort(1).Dimensions = 1;        % index k
  block.InputPort(2).Dimensions = 1;        % input (u)
  block.InputPort(3).Dimensions = 1;        % tau
  block.InputPort(4).Dimensions = [2 1];    % State x(t)

  block.OutputPort(1).SamplingMode = 'Sample';
  block.OutputPort(1).Dimensions   = [1 35]; % Buffer length

  block.OutputPort(2).SamplingMode = 'Sample';
  block.OutputPort(2).Dimensions   = [2 1];

  %% Set block sample time to inherited
  block.SampleTimes = [-1 0];


  %% Run accelerator on TLC
  block.SetAccelRunOnTLC(false);

  %% Register methods
  block.RegBlockMethod('Outputs',                 @Output);

%endfunction

function Output(block)
  k = block.InputPort(1).Data;
  u = block.InputPort(2).Data;
  tau = block.InputPort(3).Data;
  X = block.InputPort(4).Data;
  sys = block.OutputPort(1).Data;
  c = block.OutputPort(2).Data;

  A = [1 0.1;0  0.85];
  B = [0.005;0.1];

  N = length(sys);
  if k==0
     sys = ones(1,tau)*0.025;
  else
```

```
    sys = wrev(sys);
    sys(2:N) = sys(1:N-1);
    sys = wrev(sys);
    sys(N)=u;
  end
  c = [0;0];
  for l=1:N-1
    c = c + A^(k+tau-l)*B*sys(l);
  end
  block.OutputPort(1).Data = sys;
  block.OutputPort(2).Data = (A^(tau))*X  + c;

%endfunction
```

APPENDIX C: SIMULINK block diagrams -

Simulink block diagram for the closed looped developed method.



The simulink block diagrams are also found in Figures (8.1), (8.2), (8.4), (8.7), (8.10), (8.13), and Figure (8.19)..

APPENDIX D: TCL Script files –

APPENDIX D9.1:

```
#!/bin/sh
#Create a simulator object

set ns [new Simulator]
$ns use-scheduler RealTime

set namFile [open emulationInOut1.nam w]
$ns namtrace-all $namFile

set traceFile [open emulationInOut1.tr w]
$ns trace-all $traceFile

proc finish {} {
        global ns
```

```
        global namFile
        global traceFile
        $ns halt
        $ns dumpq
        close $traceFile
        exit 0
}
#The following lines define the three network nodes.
set node0 [$ns node]
set node1 [$ns node]

#The next lines allow to introduce live UDP traffic into the simulator from a UDP client through 4420 port:

set ipnetIN1 [new Network/IP/UDP]
$ipnetIN1 open readeonly
$ipnetIN1 bind 224.1.127.2 4420

#The Network ipnetIN object is connected to a Tap Agent called aIn.
set aIn1 [new Agent/Tap]
$aIn1 network $ipnetIN1

#The next step is to define the IP-UDP Network Object to inject UDP traffic from the simulator into the live network.
set ipnetOUT1 [new Network/IP/UDP]
$ipnetOUT1 open writeonly
#$ipnetOUT connect 192.168.1.4 4430
$ipnetOUT1 connect 127.0.0.1 4430

#The Network ipnetOUT object is connected to a Tap Agent called aOut.
set aOut1 [new Agent/Tap]
$aOut1 network $ipnetOUT1

#The next lines connect the nodes.
$ns duplex-link $node0 $node1 [lindex $argv 0]kb 5ms DropTail

#The next line attaches the aIn Tap Agent to node node0.
$ns attach-agent $node0 $aIn1
$ns attach-agent $node1 $aOut1
$ns connect $aIn1 $aOut1
$ns at [lindex $argv 1] "finish"
$ns run
```

APPENDIX E: GAWK Script files –

APPENDIX E9.1:

```
#This program is used to calculate the packet loss rate

BEGIN {

# Initialization. Set two variables. fsDrops: packets drop. numFs: packets sent

    fsDrops = 0;
    numFs = 0;
    fsReceived = 0;
}
{
  action = $1;
  time = $2;
  from = $3;
  to = $4;
  type = $5;
  pktsize = $6;
  flow_id = $8;
  src = $9;
```

```
  dst = $10;
  seq_no = $11;
  packet_id = $12;
      if (from==0 && to==1 && action == "+")
            numFs++;
      if (action == "d")
            fsDrops++;
           if (action == "r")
                     fsReceived++;
}
END {
      printf("Packets sent:%d lost:%d received:%d\n", numFs, fsDrops, fsReceived);
}
```

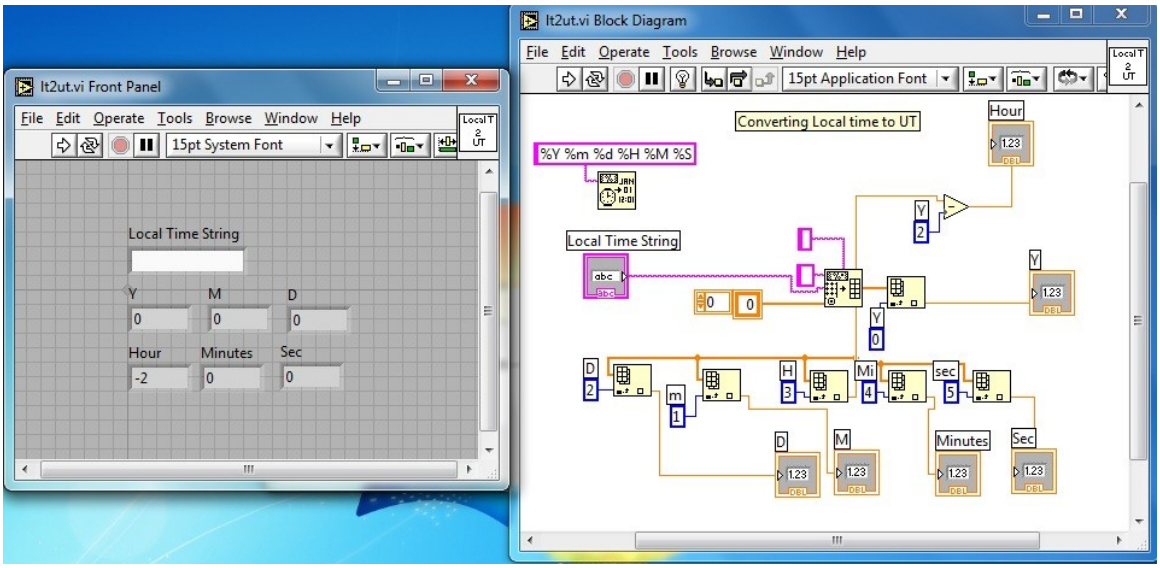
APPENDIX E9.2:

```
#This program is used to calculate the end-to-end time delay

BEGIN {

    highest_packet_id = 0;
}
{
  action = $1;
  time = $2;
  from = $3;
  to = $4;
  type = $5;
  pktsize = $6;
  flow_id = $8;
  src = $9;
  dst = $10;
  seq_no = $11;
  packet_id = $12;

  if ( packet_id > highest_packet_id )
      highest_packet_id = packet_id;

  if ( start_time[packet_id] == 0 )
      start_time[packet_id] = time;

  if ( flow_id == 0 && action != "d" ) {
    if ( action == "r" ) {
      end_time[packet_id] = time;
    }
  } else {
    end_time[packet_id] = -1;
  }
}
END {
   for (packet_id = 0; packet_id <= highest_packet_id; packet_id++ ) {
      start = start_time[packet_id];
      end = end_time[packet_id];
      packet_duration = end - start;

      if ( start < end ) printf("%f %f\n", start, packet_duration);
  }
}
```

APPENDIX F: LABVIEWTM vi and subvi files -

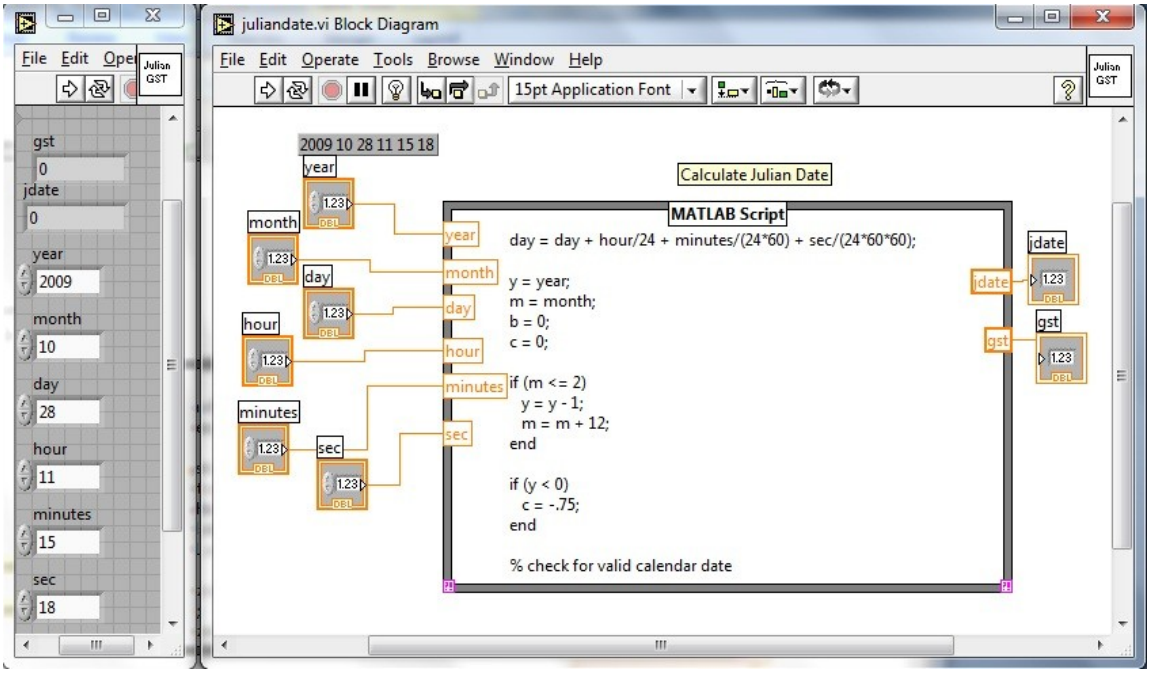
APPENDIX F9.1:

LabVIEW VI that Converts Local Time to Universal Time



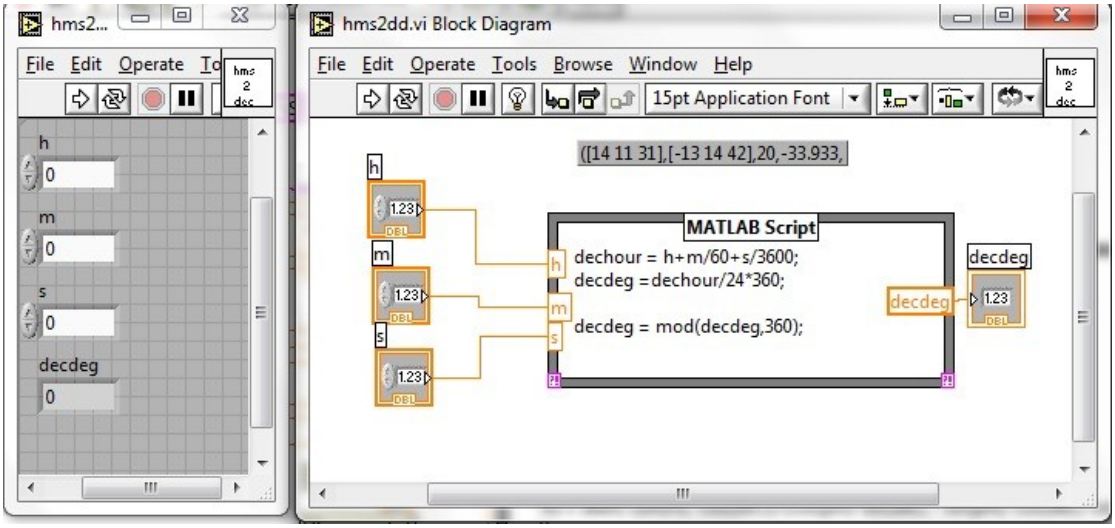
APPENDIX F9.2:

LabVIEW VI that Calculates Julian Date



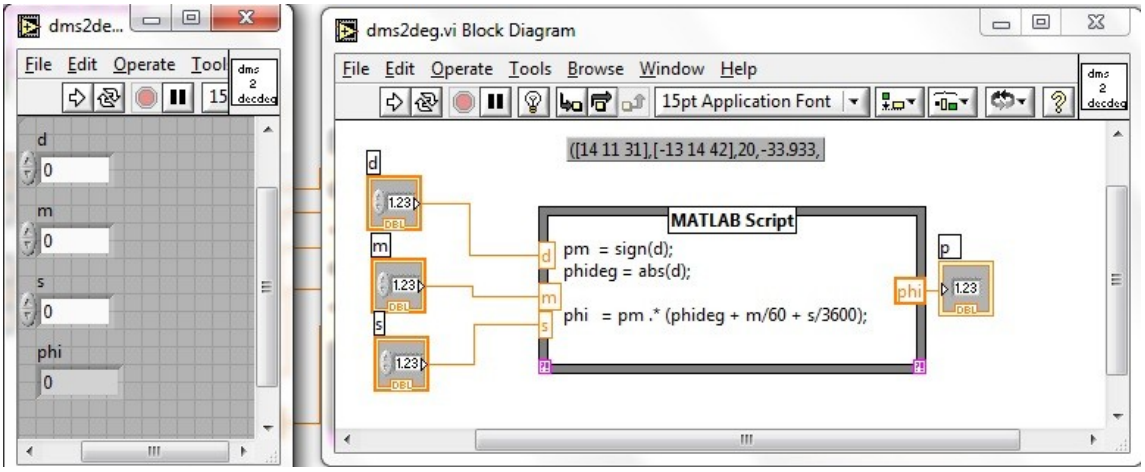
APPENDIX F9.3:

LabVIEW VI that Converts hours minutes seconds to decimal degrees.



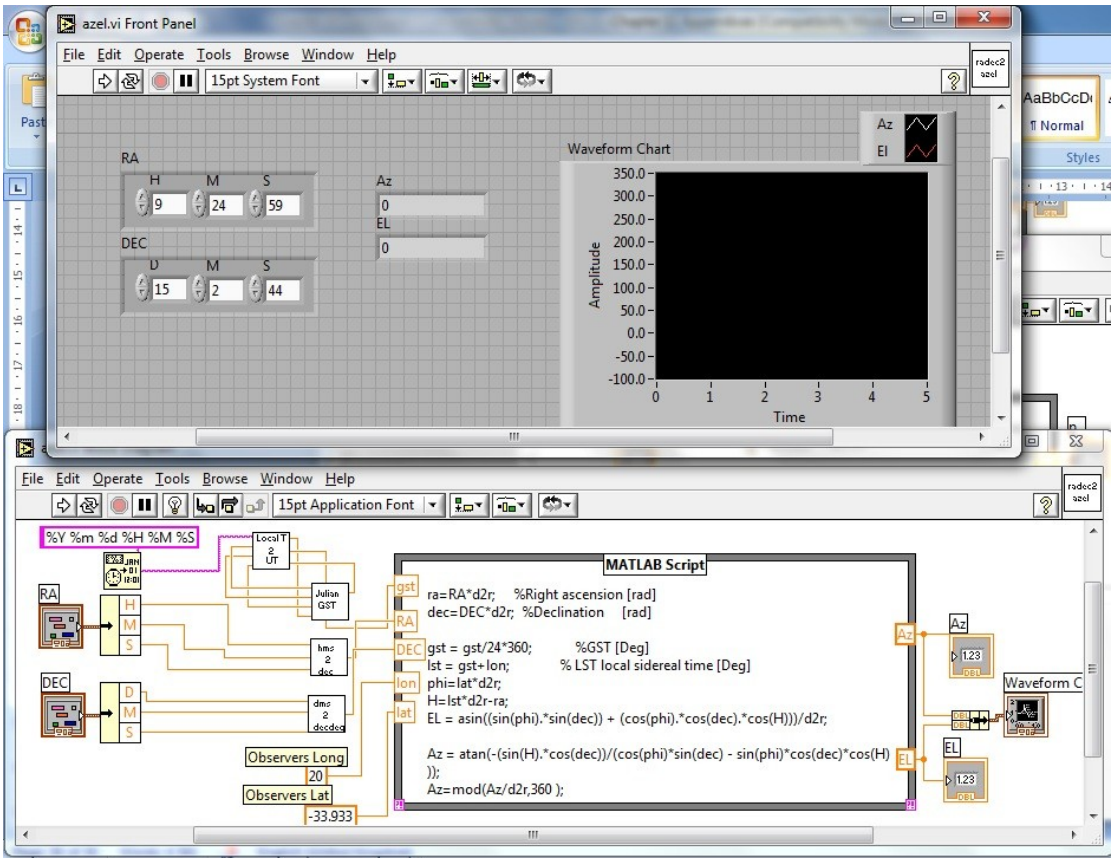
APPENDIX D9.4:

LabVIEW VI that Converts degrees minutes seconds to decimal degrees.



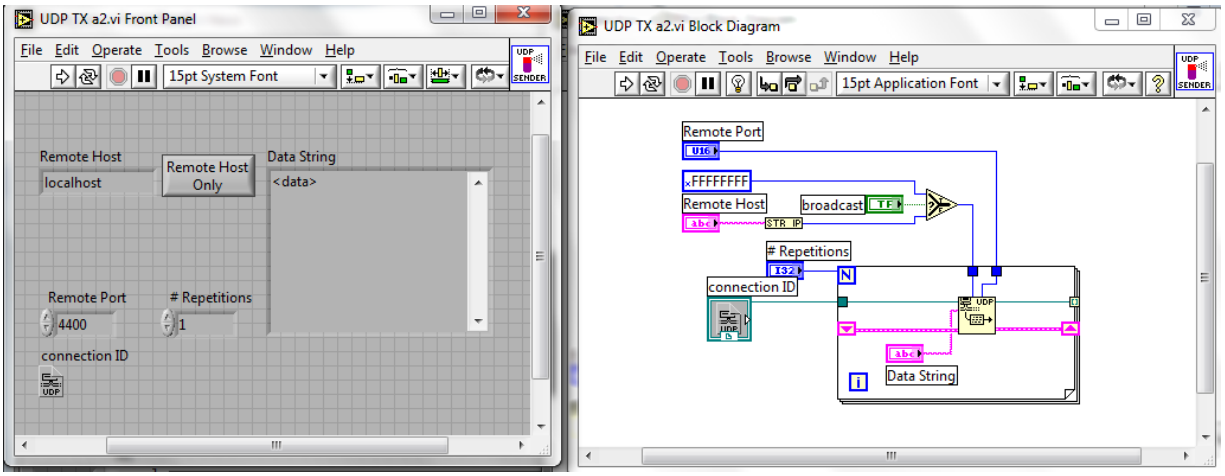
APPENDIX D9.5:

LabVIEW VI that converts RA, DEC to Altitude and Azimuth.



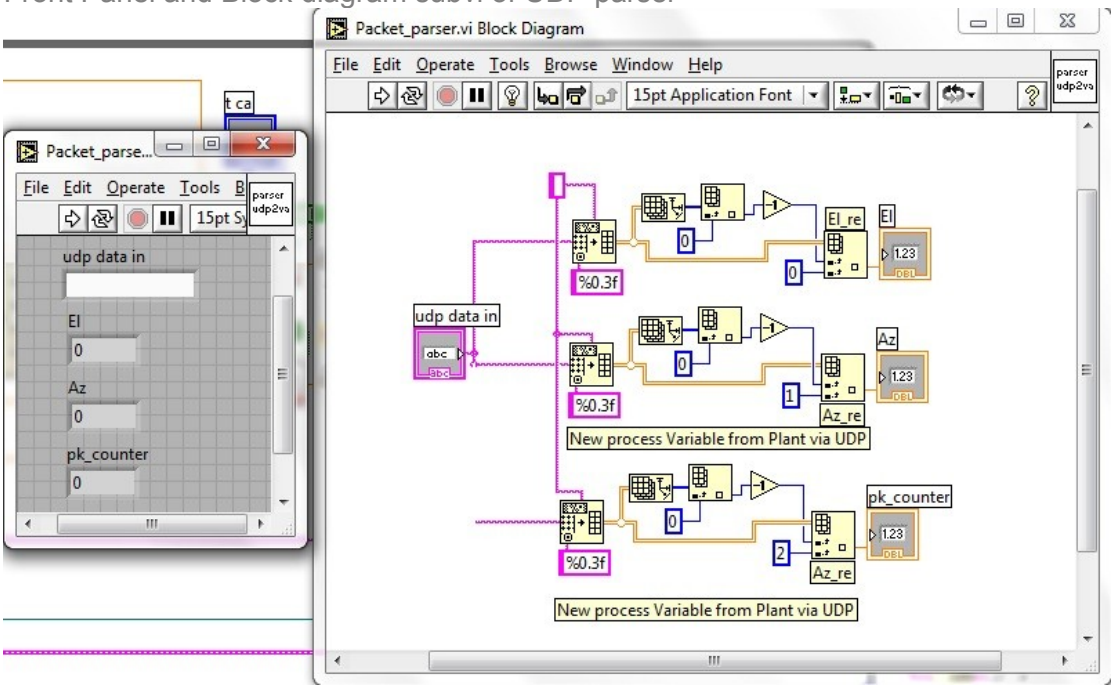
APPENDIX D9.6:

Front Panel and Block diagram subvi of sockets implementation



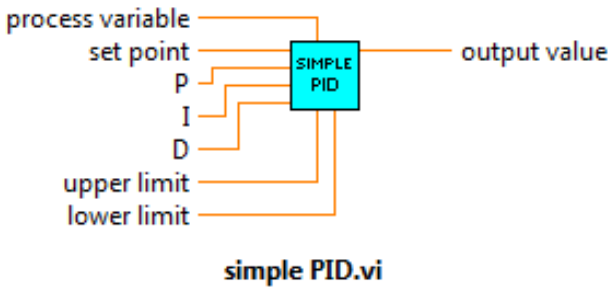
APPENDIX D9.7:

Front Panel and Block diagram subvi of UDP parser



APPENDIX D9.8:

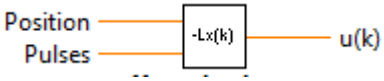
LabVIEW VI to implement PID Controller.



simple PID.vi

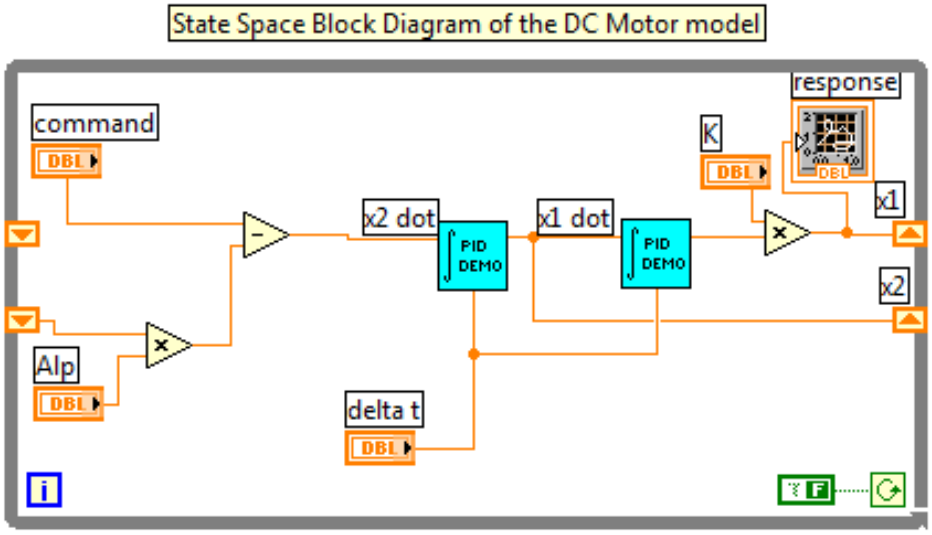
APPENDIX D9.9:

LabVIEW VI to implement State Space Controller.



APPENDIX D9.10:

LabVIEW VI to implement Plant model in State Space form.



LabVIEW VI are also found in Chapter 9, Figure (9.7), (9.10), (9.11), (9.22) and Figure (9.27a)-(9.27d)