Real Time Classification of Ballistic Missiles for Defence Applications

A thesis submitted during 2015 to the University of Hyderabad in partial fulfillment of the award of a Ph.D. degree in the School of Computer and Information Sciences

by

Upendra Kumar Singh

School of Computer and Information Sciences

University of Hyderabad

P.O. Central University, Gachibowli

Hyderabad – 500046

Andhra Pradesh, India

August-2015

CERTIFICATE

This is to certify that the thesis entitled **"Real Time Classification of Ballistic Missiles for Defence Applications"** submitted by **Upendra Kumar Singh** bearing **Reg. No. 08MCPC03** for fulfillment of the requirements for the award of **Doctor of Philosophy** in **Computer Science** is a bonafide work carried out by him under my supervision and guidance.

The thesis has not been submitted previously in part or in full to this or any other University or Institution for the award of any degree or diploma.

(Dr. Vineet Padmanabhan)
Advisor
School of Computer and Information Sciences
University of Hyderabad
Hyderabad – 500046, India

Dean
School of Computer & Information Sciences
University of Hyderabad
Hyderabad – 500046, India

DECLARATION

I, **Upendra Kumar Singh**, hereby declare that this thesis entitled **"Real Time Classification of Ballistic Missiles for Defence Applications"** submitted by me under the guidance and supervision of **Dr. Vineet Padmanabhan** is a bonafide research work which is also free from plagiarism. I also declare that it has not been submitted previously in part or in full to this University or any other University or Institution for the award of any degree or diploma. I hereby agree that my thesis can be deposited in Shodganga/INFLIBNET.

A note on plagiarism statistics from the Universiy Librarian is enclosed

Date : **Name: Upendra Kumar Singh**

Signature of the Student:

Reg. No.: 08MCPC03

// Countersigned//

Signature of the Supervisor(s):

Abstract

Classification of ballistic missiles is of paramount importance for the success of any air-defense application. Since ballistic missiles have the capability to deliver weapons of mass destruction, it is important to *classify* these targets and take suitable measures to neutralize them. This thesis addresses real-time classification of different ranges of ballistic missiles based on kinematic attributes like specific energy, acceleration, altitude and velocity which in turn are acquired by radars. The problem of real-time classification is formulated using Real-Time Neural Network (RTNN) and Hidden Markov Model (HMM). ART2 algorithm, an unsupervised network, is used for training on initial samples to enable RTNN and HMM for supervised training and testing. The idea behind RTNN algorithm is to calculate the output in one pass rather than training and computing over large number of iterations. Besides, to meet the conflicting requirements of classifying small as well as long-range trajectories, a formulation is proposed for partitioning the trajectory by using *moving window* concept. This concept allows to use parameters in localized frame which helps in reducing the problem of large as well as small trajectories to fit into the same network. A model is developed to interface the input trajectories with RTNN and HMM to classify the *type* of target in very short time. These networks are evaluated using the simulated data generated from 6 degrees-of-freedom (6DOF) mathematical model, which models missile trajectories. Experimental results show that both the networks are classifying above 95% with real-time neural network outperforming HMM in terms of time of computation on same data. To the best of my knowledge this is the first time an attempt is made to classify ballistic missiles using RTNN and HMM.

To my father Kanhaiya Singh, without whose support and encouragement, this would not have been possible.

Acknowledgements

First and foremost, I would thank my supervisor Dr Vineet Padmnabham for sharing his wealth of knowledge and wisdom with me, supporting me throughout the period of my journey. I was tremendously fortunate to have him as my supervisor.

I was very much privileged to work under a great advisory committee called Doctorate Review Committee (DRC). I thank Prof Arun Agarwal for his expertise, availability, commitment and encouragement. I am also grateful to Prof Hrushikesha Mohanty for his guidance, coaching, help and undiminished support. My deepest appreciation to Dr SK Chaudhury, Ex-Director RCI, my external guide.

The invaluable comments, advice and suggestions extended by my colleague Mr Vamsi Krishna, Scientist DRDL, is highly appreciated. I am indeed indebted for his indepth analysis, time spent with me and suggestions given by him during this research work. My special thanks to my colleague Mr Amit Kumar, Scientist DRDL who was always available for discussions and deliberations on the topic of my research.

I wish to thank Shri DLS Rao, Scientist, RCI who was always there to encourage and cheer me. Thanks are due to Mr KV Suresh, Scientist, RCI, and Anil Vishwakarma, Scientist, RCI for constant probing about the status of completion to make me struggle a bit more than I could have done myself. My thanks to my colleagues Mr Gokul Panda, Venu Gopal and Manish Raushan for their technical support. The technical help and deliberations with all of them have in no small measure contributed to the successful completion of this thesis. All my colleagues in Weapon system of Programme AD helped in ways too numerous to mention.

”Thank you” also goes to the reviewers of my publications, who gave useful comments and remarks which was useful to enhance the quality of this thesis. I wish to thank the Programme Director, PGAD and Director, RCI for allowing me to pursue my PhD and encouraging time-to-time to facilitate early completion.

Without stealing time from family members of Poonam Singh (my wife), Siddharth (son), Shweta (niece) and Adarsh (nephew), and their unconditional support, I would not have been able to finish this thesis. I really appreciate them.

Upendra Kumar Singh

Contents

List of Figures

List of Tables

LIST OF TABLES

Chapter 1

Introduction

Classification of tactical and strategic missiles is of paramount importance for the success of any air-defense application. One of the key elements (challenges) in air defense is ascertaining which element in the threat cloud is the lethal object [37, 54]. Air space during war typically consists of a combination of fighters, bombers, helicopters, transport planes, other air-crafts, ballistic and cruise missiles. Accurate and fast recognition of such targets using measured data is an important problem especially for military applications of target classification [58]. Since ballistic missiles (BM) have the capability to deliver weapons of mass destruction, it is important to identify/classify these targets and take suitable measures to neutralise them. In this thesis we focus primarily on the classification of *ballistic missile trajectories* which is a challenging problem due to its time-varying dynamics as well as short response time available for interception and which to the best of our knowledge has not been addressed before.

Ballistic missiles follow a sub-orbital flight-path with the objective of delivering one or more warhead to a predetermined target location. They have three phases of flight, namely *boost phase, mid-course phase* and *terminal phase*. Figure 1.1 shows one scenario where enemy missile is launched from point zero at x-axis and aims to destroy the targeted-location at the range of 1500 Km. Boost-phase and early mid-phase are difficult to destroy as detection requires sensors in enemy aerospace or satellite intelligence, which is not possible and not done. This leaves late mid-phase and terminal phase as possible phases of interception of the enemy missile. Though the general policy of Ballistic missile defense is to neutralise incoming threat at higher height and longer range so that debris falls away from intended impact zone [37], this is often not

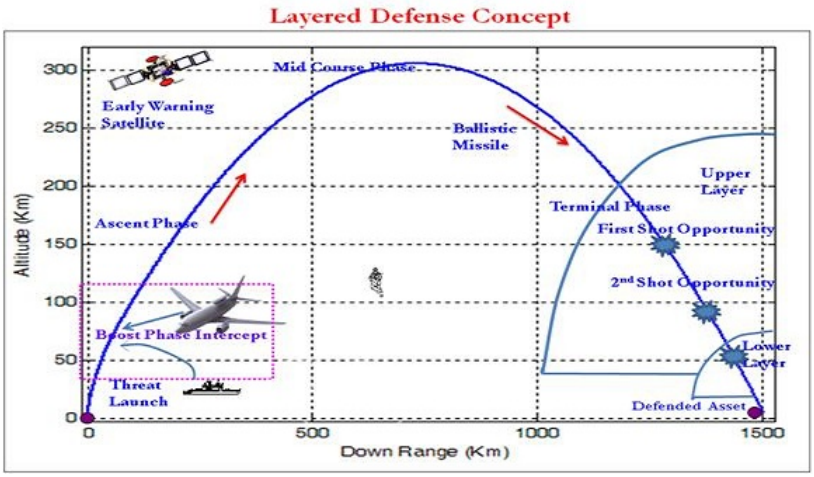

Figure 1.1: Missile Layered Defence with Different Phases of Target

realised due to late detection and short response time of launch. It is also the case that, since, ballistic missiles can be programmed to fly in *nominal, lofted* and *depressed* mode, the parameters used for classification varies for same as well as different classes of target missiles. Figure 1.2 shows a typical ballistic missile defense (BMD) scenario wherein three segments of equipments are shown: a) Sensors on the top segment which can be based on land, sea and in air such as long range land based radars, ship based radars and air-floats, UAV, satellites or advance early-warning air-borne surveillance systems. b) Middle segment shows ballistic missiles and interceptor missiles launched for annihilation of enemy intentions. c) Bottom segment shows command, control, battle management and communication systems, where classification and threat management is done.

In a typical BMD scenario, the radar [3, 26, 55] first detects the target missile and information is communicated to the launcher. The information provided by these radars can be used to classify the targets which are either direct measurement or derived parameters which includes *position, velocity, acceleration, jerk factor, specific energy, altitude, radar cross section (RCS)*, etc. The radar measures only the position of the target and this information is noisy. The noisy position data is processed using the Kalman filter to estimate position, velocity and acceleration. The filtered kinematic parameters of missiles are passed to a linear quantizer to convert the data into discrete forms. In this thesis, we are using 4 kinematic parameters, which are computed based

Figure 1.2: Generic BMD Scenario

on the kalman filter estimates. These 4 parameters are (a) `Specific energy` (b) `Acceleration` (c) `Altitude` and (d) `Velocity`. The number of classes of ballistic missiles we consider in this thesis are five, namely M400 (400 Km range), M600, M900, M1000 and M2000 class of ballistic missiles.

In ballistic defense, the `response time` needed to neutralize a ballistic missile is dependent on many factors. These include *range of radars* (needed for acquiring target of the given radar cross section (RCS)), *target velocity, preparation time* of interceptor, *velocity* of interceptor, *time of flight* of interceptor, *planned height* and *range of kill*. `Apogee` (the peak height obtained by the missile during its total flight) and range of travel of ballistic missile depends on velocity, height, and altitude of vechicle at the time of burn out of propellant. `Burnout time` is time at which propellant is spent out with respect to the `time of lift-off` of the vehicle. The launch bracket depends on time taken for preparation of missile and time required by interceptor to reach the `kill height`. Interceptor has variable flight duration depending on feasible height of kill. The time of `preferred-kill` is time at which interceptor can reach interception point based on its maximum capability and time at which it is launched to achieve preferred-kill is preferred-launch time. Inorder to arrive at response time we developed a 6 DOF (degree of freedom) mathematical model for simulating the behaviour of ballistic missiles and generate the trajectories. Table 1.1 shows various parameters of missile from 6-DOF simulation to arrive at response time. All time information given in the table is time with respect to launch of target missile

Table 1.1: Various parameters from 6-DOF simulation to arrive at response time

Missile Range (Km)	Flight Time (Sec)	Burnout velocity (m/sec)	Burnout time (Sec)	Apo gee (Km)	Detection time (Sec)	Preferred launch (Sec)	Preferred kill @ 140km	Response time (Sec)
400	353	1500	43	115	45	148	248*	+103
600	410	2200	50	150	60	160	280	+100
1000	540	2800	62	250	356	426	546	+ 70
1500	680	3500	80	375	466	492	612	+ 26
2000	742	4200	110	500	566	546	666	-20

Interceptor missile: $t_{flightmax} = 120$ sec, $t_{flightmin}$ = 80 sec.
Response time = Detection time - Preferred launch time
$*$ Kill altitude is 100km, since Ballistic missile (BM) apogee is 115 km

for different class of ballistic missiles. Missile range in the first column is down-range it can travel. First five columns in the table relate to parameters of ballistic missile and last four columns enlist events related to interception.

A radar of specific acquisition range will detect a target within its capability of range acquisition with specified radar cross section. With radar placed at 50 Km ahead of impact position of target and with a radar acquisition of fix range of 600 Km, 6th column of Table-1 shows the detection time of the target missile of various ranges. Keeping range of radar constant and height of preferred-kill as 140 Km, it is clear from the table that response time has decreased from 100 sec for 600 Km to negative for 2000 Km class of missile and for a given interceptor flight time, target cannot be engaged at an altitude of 140 Km or more. *Thus, response time is very short and a real-time classification can play a vital role in initiating counter measures of launching interceptor missile against incoming ballistic missile*.

1.1 Objectives

I am working in program AD (Air Defence), RCI (Research Centre Imarat), a premier institution of Defence Research and Development Organisation (DRDO) Hyderabad, which is involved with design and development of interceptor missiles. Working with ballistic missile defence program has enabled me to work in the area of *target classification* for classifying flying objects using radar data. The main objectives of this thesis

are:

(a) To study the present target classification techniques and propose new approaches for classification of aerial objects for achieving real time performance.

(b) To decide parameters which will be minimal and sufficient to distinguish objects without any confusion.

(c) To evolve methodology for real-time classification.

(d) Formulate the problem of classification using Real-time Neural Network (RTNN).

(e) Formulate the problem of classification using Hidden Markov Model (HMM).

(f) To compute time of clean classification.

(g) To provide initial data using unsupervised training of Adaptive Resonance Theory (ART2) for continuous data.

(h) Compare the results of Real-Time Neural Network (RTNN) with Hidden Markov Model (HMM).

(i) The proposed techniques are experimented and demonstrated based on radar and simulated ballistic missile trajectory.

1.2 Thesis Outline

Chapter 1 of the thesis is the Introduction, where we review the importance of real-time classification and its use in missile defence applications. We also narrate the motivation, point out the objectives and end up with an outline of the thesis and summary of results.

Chapter 2 of the thesis gives a brief overview of different classifiers used in the literature for general classification problems. All classifiers are used for offline analysis and training time is too high for real-time applications. Same model and methodologies suitable for catering classification of small as well as large trajectories are not attempted to the best of our knowledge. Input data analysis, Time Domain analysis

of Ballistic missile scenario, Data capturing by Radars, six degree-of-freedom (6DOF) generated trajectories, Noise mixing to the order of actual available radars and Kalman filtering these noisy data are reflected in this chapter. The parameters used for classification, air situation trajectories in battle scenario, analysis of parameters for classification are also presented in this chapter.

Based on Huang's [29] constructive network model and real-time learning algorithm, in **Chapter 3**, we propose a formulation for classification of ballistic as well as quasi-ballistic missile trajectories using Real-time neural network (RTNN). The real-time neural-network architecture, its topology, role of quantizers and formulae used for analytical calculations are shown in this chapter. Two scenarios of trajectories are tested: one with trajectory of M400, Helicopter and Aircraft; and other with trajectories of M400, M1000 and M2000 class of missiles. Besides, to meet the conflicting requirements of classifying small as well as long-range trajectories, we also propose a formulation for partitioning the trajectory by using moving window concept. This concept allows us to use parameters in localized frame which helps in handling wide range of trajectories to fit into the same network. The proposed algorithms are evaluated using simulated data generated from 6 degree-of-freedom (6DOF) mathematical model. External disturbance due to wind and misalignment is also considered in the model. Temporary loss of data as well as track offset is experimented to simulate quasi-ballistic maneuver. Our experimental results demonstrate that RTNN is able to classify both Ballistic as well as Quasi-Ballistic missile within 1 ms. The small classification time enables the use of real-time classification neural network in complex scenario of multi-radar, multi-target engagement by interceptor missiles. *To the best of our knowledge this is the first time an attempt is made to classify ballistic missiles using RTNN*. The results of this study are published in Proceedings of "International Joint Conference on Neural Network" (IJCNN-2013) and "World Academy of Science, Engineering and Technology" (WASET-2011).

Investigations on the suitability of hidden Markov models (HMM) for real-time trajectory classification forms the major content of **Chapter 4**. The trajectory classification problem is formulated as to identify the class $C_i (i = 1 \ldots L)$ to which the

trajectory state sequence belongs. Each trajectory class C_i is represented by a Model, λ_i , where $\lambda_i = \{\pi, A, B\}$, in which π is initial state probabilities, A is state transition probabilities of dimension (N $\times$ N) and B is observation symbol probability distribution of dimension (N $\times$ M). Since HMM considered is discrete, number of states for HMM for missile classification are considered as N=25; Number of Observation Symbols, M, for Specific Energy is 200, Acceleration is 40, Height is 60 and Velocity is 80. Initial state probability, π and state transition probabilities are set to be equal to 1/N. Observation symbol probability, B is based on Gaussian (with mean and variance) representation for the distributions of observations within each state. Experimental results of HMM demonstrate that pass percentage achieved is more than 95.5% over 900 samples of test data. The trajectory is validated on a model with different number of states in HMM. HMM with 20 numbers of states and above are giving results above 95% whereas performance of the model remains unsatisfactory with 5,10 or 15 states. Time taken by the model for testing new sample of data for 5, 10, 15, 20 and 25 states are 4.816 milli-seconds (ms), 8.02 ms, 18.17 ms, 29.06 ms and 49.24 ms respectively. The formulation is also tested with two important and practical conditions of losses of data in the *track* and *manoeuvring* targets, the later being a recent improvement in the ballistic missile development. The results of this study are published in Proceedings of "Sixth International Conference on Contemporary Computing" (IC3)-2013.

Chapter 5 deals with a self organising neural network called (ART2) which is based on Adaptive Resonance Theory (ART) and is used in training the initial data. ART2 rapidly self-organizes pattern recognition categories in response to arbitrary sequences of either analogue or binary input patterns which is essential for training both RTNN and HMM during initial period. ART-2 finds the initial clusters using unsupervised learning which is then fed to HMM for classification using recursive method. Basic architecture of ART, ART-2 theory, network components, selection of parameters for network training, learning laws modes of learning and experimental results are described in detail. The results of this study are accepted by the "Fifth International conference on Pattern Recognition and Machine Intelligence" (PReMI)-2013.

Chapter 6 presents comparison of results for Real-Time Neural networks (RTNN) and Hidden Markov Model (HMM), once initial data is trained with Adaptive resonance Theory (ART-2). Practical scenario of introduction of noise due to sensor measurement, loss of measurement data for upto 20 samples and maneuvering target scenario is also tested with RTNN and HMM. This is the first time an attempt is made to classify ballistic missiles using ART2, RTNN and HMM. The results has been accepted for publication in "Applied soft computing Journal" in March 2014.

We conclude the thesis by summarizing the contributions in **Chapter 7**. This Chapter also outlines some directions for future research.

1.3 Thesis Contributions

- Developed a 6 DOF mathematical model for simulating the behaviour of ballistic missiles and generate their trajectories.
- Developed a model for dynamic classification wherein a moving window concept is introduced that allows to fit large as well as small trajectories in the same network.
- For the first time it is demonstrated that real-time neural network (RTNN) can be used to classify ballistic missile trajectories.
- First ever attempt to develop a hidden Markov model (HMM) for real-time trajectory classification.
- The use of Adaptive Resonance Theory (ART) for training the data which is essential for training both RTNN and HMM during initial period.

Chapter 2

Foundational Concepts and Related Work

In this chapter, we introduce the basic scenario of ballistic missile defence wherein we explain the need for neutralization of weapons of mass-destruction. We also outline the requirements for real-time classification and data generation using radars. Target state estimator for noisy data filtering as well as response time analysis is also explained. The modules needed for developing a 6-DOF (degrees-of-freedom) mathematical model for simulating the behaviour of ballistic missiles is discussed at length.

2.1 Ballistic missile defence scenario

Ballistic missiles are either single stage or multi-stage vehicles which are launched vertically to clear the canister or cage from which it is generally launched. After a pre-programmed period of vertical rise, the missile pitches down to fly in the desired azimuth [46]. During this phase, the missile is controlled in altitude using some form of thrust vector control (TVC) and once adequate velocity has been obtained, the control functions are taken over by aerodynamic control. Once the flight vehicle is out of the atmosphere, the closed loop guidance starts with the objective of achieving the desired injection conditions in terms of height, velocity and flight angle. After obtaining the proper conditions, the thrust cut off is commanded. Later, the vehicle coasts till impact. The flight vehicle during its coast phase does not carry out any guidance function, however, the control functionalities are exercised.

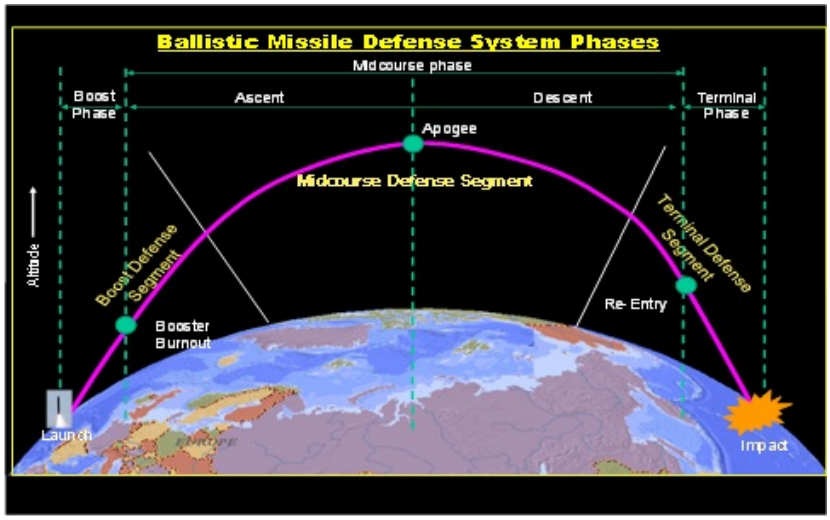

Figure 2.1: Three main phases of ballistic missile

A ballistic missile has three phases of flight, namely boost, midcourse and finally a terminal phase which are graphically shown in Figure 2.1. The boost phase has a distinct beginning and end. It begins with liftoff of the missile and ends with thrust cut-off. The midcourse phase also has distinct beginning with thrust cut-off but ends with re-entry in atmosphere where lift and drag becomes effective. The terminal phase is effective utilization of atmospheric forces to alter speed and trajectory. Ballistic missiles are categorised as Short Range Ballistic Missiles(SRBM), Medium Range

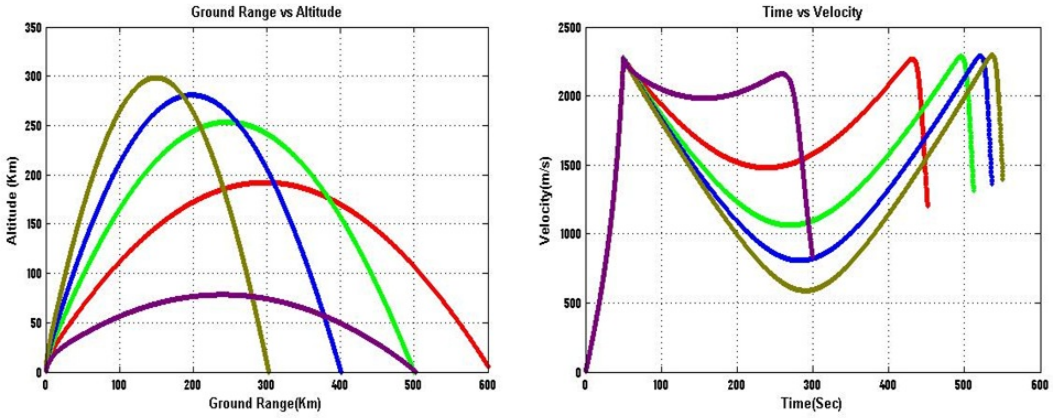

Figure 2.2: Trajectory and Velocity Profile of a SRBM missile (600 Km)

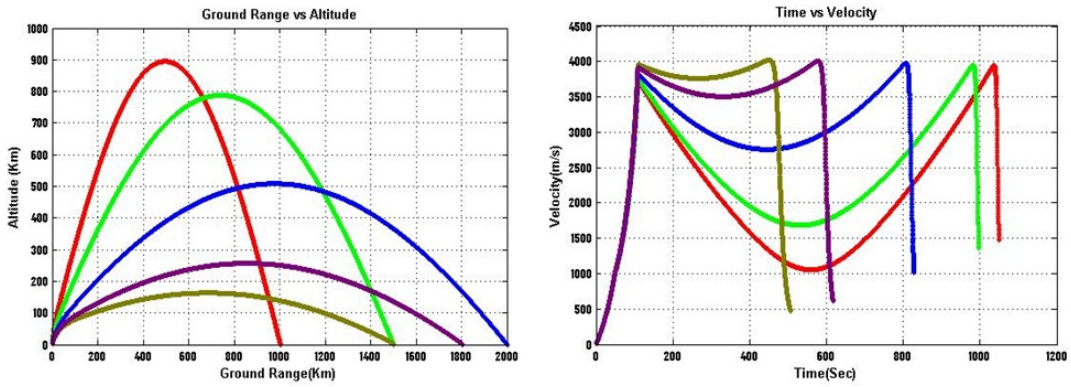

Figure 2.3: Trajectory and Velocity Profile of a MRBM missile (2000 Km)

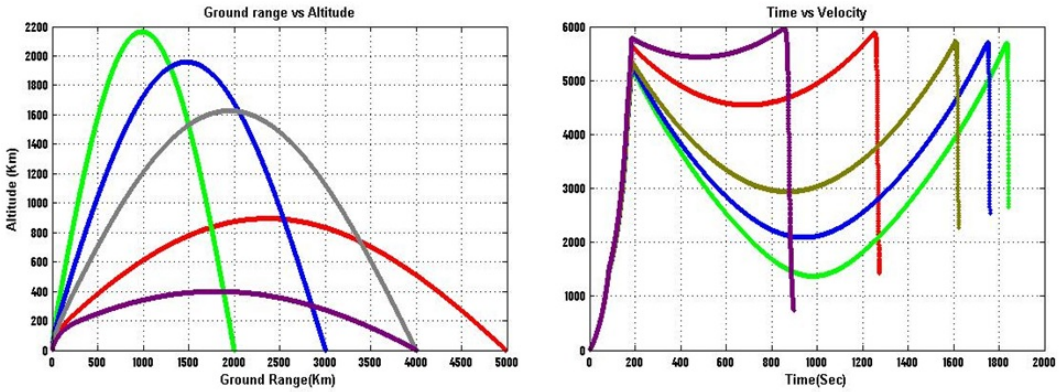

Figure 2.4: Trajectory and Velocity Profile of an ICBM missile (5000 Km)

Ballistic Missiles (MRBM) and Inter-continental Ballistic missile (ICBM). Ballistic missiles trajectories for 600 Km, 2000 Km and 5000 Km classes are shown in Figure 2.2, Figure 2.3 and Figure 2.4 for *ground range coverage vs altitude* and *time vs. velocity*. Each figure shows 5 different possible trajectories of the same class. SRBM of 600 Km range varies from 300 Km to 600 Km of range coverage as in Figure 2.2. MRBM of 2000 Km class of missiles ranges from 1000 Km to 2000 Km as in Figure 2.3 and ICBM of 5000 Km covers from 2000 Km to 5000 Km as in Figure 2.4.

2.1.1 Ballistic Missile Neutralization

An incoming ballistic missile may be neutralized in any one of its flight phases namely (a) boost phase (b) mid-course and lastly (c) terminal or re-entry phase [37]. Boost

phase defense is when the target missile is boosting whereas mid-course intercept is when it is in free fall. The terminal phase intercept is when there are few tens of seconds before impacting on its target. A capability to intercept a missile in the boost phase could destroy a missile regardless of its range or intended aim-point and provide a very large coverage capability. However, it is very challenging as additional intelligence is required that missile was really intended for perceived task. A mid-course intercept capability could provide wide coverage of a region, while a terminal defense would protect a localized area. The defense system becomes multi layered when more than one type of interceptor is used and significantly increases the system effectiveness.

Ascent phase and early mid-course phase are difficult to destroy as detection requires sensors in enemy aero-space or satellite intelligence, which is not possible and not done. This leaves late mid-course phase and terminal phase as possible phases for interception of the enemy missile. Upper layer indicated in the Figure 2.5 is called exo-atmospheric region whereas lower layer is known as endo-atmospheric zone as indicated by two arcs. Multiple shots can be fired in upper layer because of larger zone of operation. Though, the general policy of BM(ballistic missile) defence is to neutralize incoming threats at higher height and at longer range so that debris falls away from intended impact zone [54], this may not be realized due to late detection and short response time of the launch. A major reason, the defense operates with multiple

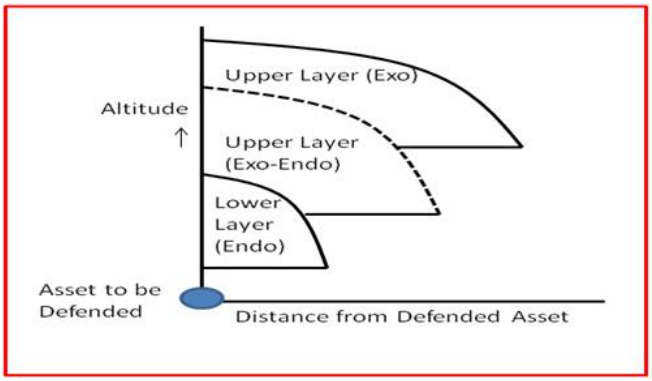

Figure 2.5: Layered defense system

layers is leakage reduction. If a single layer destroys 90% of the attacking Re-Entry vehicle (RV), two independent layers operating sequentially can destroy 99% of the RVs. By definition, each layer in a multiple-layer defense has its own complement of

interceptors and its own interception phases [53]. Since each interceptor has its own associated probability of failing to kill its target; a single interceptor has an associated leakage. The second layer can cover leakage from the first layer because of either discrimination failure or interception failure.

2.2 Sensor Network

Long range sensors like radars are the initiating points for ballistic missile defence for detection and classification of hostile targets. Radars use the electromagnetic spectrum for transmitting and receiving signals. The radar generates a signal, which is transmitted through an antenna into free space. The antenna is designed to concentrate the radar energy in a particular direction towards the target. The radar beam broadens as the wave propagates from the radar and therefore its strength reduces with distance. The energy incident upon the target also reduces with distance. For a metallic target, most of the incident energy is reflected and a small amount is absorbed. The energy reflected from the target tends to propagate in all directions and the amount directed back towards the radar is dependent upon the wavelength of the signal, the size, shape and the materials of the aircraft. Hence, only a small proportion of the energy actually illuminating the target-missile, or any other target, is reflected back in the direction of the radar as shown in Figure 2.6. Radars measure the range and the velocity of the target. Its position in azimuth and elevation could also be measured, so that its altitude could then be estimated. From the strength of the signal reflected from the target, an estimate could be made of its radar cross-section, which provides a rough measure of its size [7]. Many types of sensors are connected in the sensor network for full coverage of the protected region. Sensor network consists of a number of long-range radars

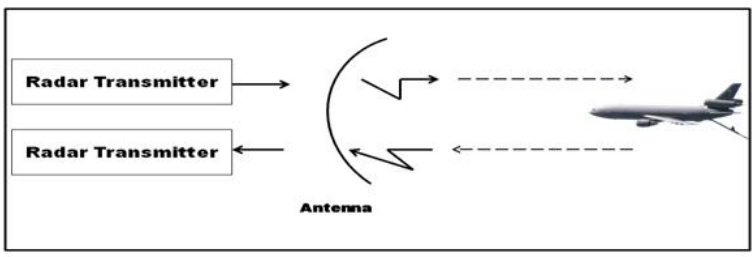

Figure 2.6: Transmitting and receiving a signal from a target

placed in the forward region for surveillance as well as short range fire control radar located in the region where the asset is to be protected. Airborne warning and control system (AWACS), satellite based sensors and air-floats also form part of the sensor network. These nodes have redundant connectivity of leased line through public network and VSAT (very small aperature terminal) besides last-mile connectivity through point-to-point and point to multi-point connectivity [27, 29, 56].

2.3 Ballistic Missile State Estimator

The problem of real time estimation or tracking of ballistic target especially re-entry vehicle (RV) from its radar observations is a complex problem in non linear filtering. Tracking the ballistic missiles (BM), currently is of great interest to the BMD(ballistic missile defense) community. The major challenge lies in the tracking of non-manoeuvering and manoeuvering BM targets, whose drag profiles deviate significantly from prior predictions [38, 51]. Kalman filter (KF) has been used to provide optimum estimates (in least squares or maximum likelihood sense) of the states of linear dynamical systems. This idea was soon extended by Kalman and Bucy to estimate the states of nonlinear dynamical systems using the extended Kalman filter (EKF). The EKF technique generally employs a Taylor series expansion [12, 15] of original nonlinear plant dynamics to obtain linearized equations around previous state estimates. In expansion process, generally second and higher order terms are neglected. These linearized equations are used to compute the error covariance matrix. Both state and covariance equations are propagated initially. Based on the available measurements at propagated time, the time varying filter gains are computed and both the states and corresponding covariances are updated.

Filter plays a vital role for estimation of trajectory of BMD. The estimator with state equations in cartesian frame and measurements in polar frame has been selected for applications. Filtering of BM state variables (position and velocity components) is very important for classification of the missile. The external sensor like radar measures target range, azimuth and elevation often getting contaminated by measurement noise. The aim of the EKF estimator is to estimate position, velocity and acceleration components of the target from the above available measurements within accuracy based on sensor noise characteristics [59, 65]. Cartesian-Polar filter has been developed for

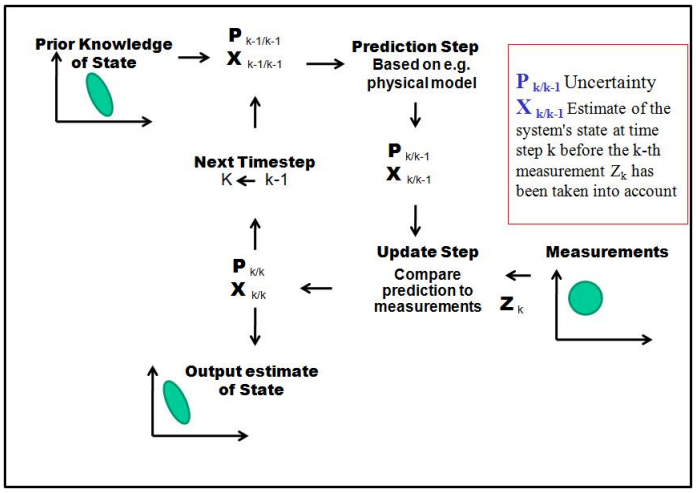

Figure 2.7: State Estimator

estimating the BM position, velocity and acceleration components for trajectory. The governing equations of motion of BM are prerequisite for filter design.

2.3.1 Reference Frames

Figure 2.8(a) shows the locations of radar and the target with respect to WGS-84 ellipsoid model of the earth i.e. Earth Centred Earth fixed(ECEF) frame. The radar is located on the surface of the earth with height (h_s), geodetic latitude (λ) and longitude (μ). Generally (x, y, z) is the commonly used Cartesian system since it corresponds to the physical space. In present axes system, z is directed along the local vertical and x and y lie in a local horizontal plane, with x pointing east and y pointing north. The second Cartesian system shown in Figure 2.8(b) is x^1, y^1, z^1. This frame moves with the target. This system can be obtained from x, y, z by a rotation in azimuth of angle (ψ) and elevation of angle (θ).

2.3.2 Ballistic Missile Dynamics and Target Models

This section describes the forces acting on the ballistic missile and its equations of motion in ECEF. The main forces acting on the ballistic missile are thrust, drag, gravity, centrifugal, coriolis and lift forces. The effects of centrifugal and coriolis forces are significant for long-range trajectories. During Boost phase, the target is subjected to

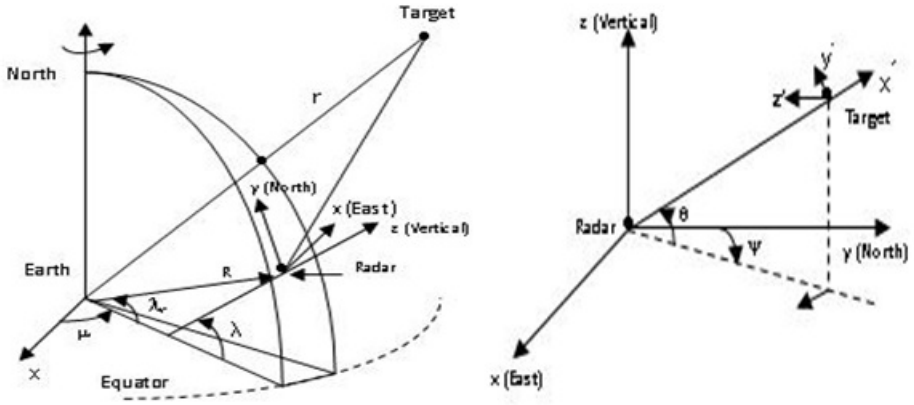

Figure 2.8: a) Radar and target locations in ECEF frame b) Radar and target location

large acceleration due to thrust, atmospheric drag and the earth's gravity. The forces considered during ballistic and re-entry phases are Gravity, drag, centrifugal and carioles forces. The ballistic model is a 6-state EKF with gravity and coriolis terms. The state elements are position and velocity. State propagation, however, includes gravity and coriolis forces, even though the state does not contain acceleration. The re-entry model is a 7-state EKF, identical to the ballistic state but has a 7^{th} element, the (inverse) ballistic coefficient.

2.3.3 Atmospheric Dynamics

The motion of the ballistic missile(BM) in the ballistic and re-entry phase is modelled in the ECEF frame. The model takes into account the following forces: a) the Gravity Force b) the Aerodynamic Force, and c) the coriolis and centrifugal Forces.

Gravity Force: It is assumed that the gravity field is directed from the BM's centre of gravity (CG) to the center of the earth. The acceleration due to gravity is given by

$$|g| = \frac{\alpha}{r^2}$$

where, $|r|$ is the distance between the BM CG and center of the earth. α is the universal gravitational constant (G.M.m) and involves Gravity (G), Mass of Earth (M) and body mass (m) terms. ($\alpha = 3.9742184 * 1014 m.s^{-2}$).

Aerodynamic Force: The acceleration due to the aerodynamic force is modelled as

follows:

$$A_{aero} = \frac{\rho(h)|\overrightarrow{V}|\overrightarrow{V}}{2\beta}$$

where, $\rho(h)$ is the air density which is function of the height. The height is the BM position over the surface of the earth. β is the ballistic coefficient and is given by $\beta = \frac{m}{S*C_{d0}}$; S is Surface Area of Body; $\overrightarrow{V}$ and $|\overrightarrow{V}|$ are the velocity vector and its magnitude of BM target.

The Coriolis and Centrifugal forces: These forces are fictitious forces which arise due to the filtering frame reference rotation. The acceleration due to the coriolis and the centrifugal forces is given by

$$A_{ficti} = -2\omega \times \overrightarrow{V} - \omega \times \omega \times \overrightarrow{r}$$

where, $\overrightarrow{r}$ is the BM position vector; $\omega = [0, 0, \Omega]$ where, Ω is the Earth Rotation Rate which is = 7.2717×10^{-5} rad per sec.

2.3.4 Boost Phase Model

The boost phase of a ballistic missile(BM) lasts from launch to stage burnout. During this phase, the target is subjected to large acceleration due to thrust, atmospheric drag and the earth's gravity. The thrust acceleration profile varies largely from one target to another depending on the type of the target. In BM, the rate of change of acceleration is considerable as the burn time increases due to the decrease in the propellant weight. The target model considered is constant jerk model originally developed by Mehrotra et al [38]. The state and measurement equations corresponding to jerk model in Cartesian frame are given below:

State Equations The boost model is a 12-state EKF in ECEF Cartesian coordinates for the purpose of composite tracking. The state elements are position, velocity, acceleration and jerk. The equations of motion for boost model is given by

$$X_{k+1} = \phi_k X_k + W_k$$
$$X = [x \;\; v_x \;\; a_x \;\; j_x \;\; y \;\; v_y \;\; a_y \;\; j_y \;\; z \;\; v_z \;\; a_z \;\; j_z]$$

X is target state vector consisting of (x,y,z) the position components, (v_x, v_y, v_z) the velocity components, (a_x, a_y, a_z) the acceleration components and (j_x, j_y, j_z) the jerk components of BM target in ECEF frame.

$$\phi = Diag \begin{bmatrix} 1 & T & \frac{T^2}{2} & \frac{T^3}{6} \\ 0 & 1 & T & \frac{T^2}{2} \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

W_k: is zero mean with Gaussian process noise.

2.3.5 Ballistic and Re-entry Phase Model

The forces acting on the ballistic missile are described in Section 2.3.2 The target motion in ECEF frame is described by the following continuous time-varying non-linear dynamical equations.

Target State Model The general system of equations, which are nonlinear, describing the motion of the ballistic target during atmospheric phase, are given below:

$$\dot{X}(t) = f(X(t)) + w(t)$$

where f (.) are partial nonlinearities that are encountered in modeling of the re-entry dynamics.

$$f(X(t)) = \begin{bmatrix} v_x \\ v_y \\ v_z \\ \left(\frac{-\alpha}{r^3} + \Omega^2\right) x + 2\Omega v_y - \frac{\rho(h)}{2\beta} v v_x \\ \left(\frac{-\alpha}{r^3} + \Omega^2\right) y + 2\Omega v_x - \frac{\rho(h)}{2\beta} v v_y \\ \frac{-\alpha}{r^3} z - \frac{\rho(h)}{2\beta} v v_z \end{bmatrix}$$

$w(t)$ is zero mean with Gaussian Process noise. $E[w(t)] = 0$, $E[w_k {w_k}^T] = Q_k$ is the Process noise Covariance. The EKF is a 7-state filter whose states are

$$X = [x \;\; y \;\; z \;\; v_x \;\; v_y \;\; v_z \;\; 1/\beta]^T$$

Here, X(t) is target state vector of dimension 7x1. The variables (x,y,z) are the ballistic target position components and (v_x, v_y, v_z) are target velocity components along ECEF frame and β is ballistic coefficient.

Measurement equations The nonlinear measurement model is described as

$$Z_k = h(X_k) + v_k$$
$$E[v_k] = 0 \;\; E[v_k {v_k}^T] = R_k$$

where $Z_k = [r_k \;\; \theta_k \;\; \psi_k]^T$ are radar measurements available at k^{th} instant of time. The function $h(X_k)$ is a non-linear observation matrix and v_k is zero mean with Gaussian noise. The available measurements (r, θ, ψ) are related to the filter states in ECEF as follows:

$$\begin{bmatrix} x_{env} \\ y_{env} \\ z_{env} \end{bmatrix} = M_{ECEF}^{ENV} \begin{bmatrix} x - x_r \\ y - y_r \\ y - y_r \end{bmatrix}$$

$$\begin{bmatrix} r \\ \theta \\ \psi \end{bmatrix} = h(X_k) = M_{ECEF}^{ENV} \begin{bmatrix} \sqrt{{x_{env}}^2 + {y_{env}}^2 + {z_{env}}^2} \\ \tan^{-1}\left[\frac{z_{env}}{\sqrt{{x_{env}}^2 + {y_{env}}^2}}\right] \\ \tan^{-1}\left[\frac{x_{env}}{y_{env}}\right] \end{bmatrix}$$

$$M_{ECEF}^{ENV} = \begin{bmatrix} -\sin\lambda & \cos\lambda & 0 \\ -\sin\psi_d \cos\lambda & -\sin\psi_d \sin\lambda & \cos\psi_d \\ \cos\psi_d \cos\lambda & \cos\psi_d \sin\lambda & \sin\psi_d \end{bmatrix}$$

$$\psi_d, \lambda : \;\; Geodetic\ Latitude\ and\ Longitude$$

$$\begin{bmatrix} x_r & y_r & z_r \end{bmatrix}^T \;\; : Radar\ position\ in\ ECEF\ frame$$

$$\begin{bmatrix} x_{env} & y_{env} & z_{env} \end{bmatrix}^T : Target\ position\ in\ ENV\ frame$$

$$M_{ECEF}^{ENV} \;\; : DCM\ for\ ECEF\ to\ ENV\ Frame\ conversion$$

2.3.6 Filter Equations

The Kalman filter keeps track of the estimated state of the system and the variance or uncertainty of the estimate. The estimate is updated using a state transition model and measurements. $X_{k/k-1}$ denotes the estimate of the system's state at time step k before the k^{th} measurement Z_k has been taken into account; $P_{k/k-1}$ is the corresponding uncertainty as shown in Figure 2.7 and the detail equations are given below:

1. State vector estimation at the time t_k is based on the measurements upto time t_k.

$$X_{k/k} = X_{k/k-1} + K_k(Z_k - h(X_{k/k-1}))$$

2. Predicted state vector at the time t_k computed is based on the estimate at time t_{k-1}.

$$X_{k/k-1} = X_{k-1/k-1} + \int_{t_{k-1}}^{t_k} f(X(t))dt$$

3. Kalman gain

$$K_k = P_{k/k-1}H_k^T(H_kP_{k/k-1}H_k^T + R_k)$$

where $H_k = \left[\frac{\partial H(X)}{\partial X}\right]_{X=X_{k/k-1}}$

H_k is the linearised observation matrix. R_k is the measurement noise co-variance matrix.

4. Predicted Error Covariance matrix compute at the time t_k is based on the time t_{k-1}.

$$P_{k/k-1} = \phi_{k-1}P_{k-1/k-1}\phi_{k-1}^T + Q_{k-1}$$

$$with\ Q_{k-1}\ being\ the\ process\ noise\ matrix.$$

$$\phi_{k-1}\ being\ the\ state\ transition\ matrix\ computed\ at\ time\ t_{k-1}$$

where

$$\phi = \left[\frac{\partial F(X)}{\partial X}\right]_{X=X_{k/k-1}}$$

is the linearised system dynamics matrix at the time t_k.

5. Estimated Error covariance matrix at the time t_k.

$$P_{k/k} = (I - K_kH_k)P_{k/k-1}$$

2.3.7 Interactive Multiple Model(IMM) Filter Model

The IMM filter comprises of the following two Kalman filter models: (i) Jerk model in Section 2.3.4 and (ii) Ballistic/Re-entry model in Section 2.3.5. The block diagram of IMM filter consisting of models as jerk filter along with ballistic/re-entry Filter is shown in Figure 2.9. At every time step 'k', a linear combination of the previous output (states and covariance) is input to each model. The current measurements are passed to each model concurrently and residues are computed along with the corresponding likelihood functions as shown in Figure 2.9. The different parameters in Figure 2.9 can

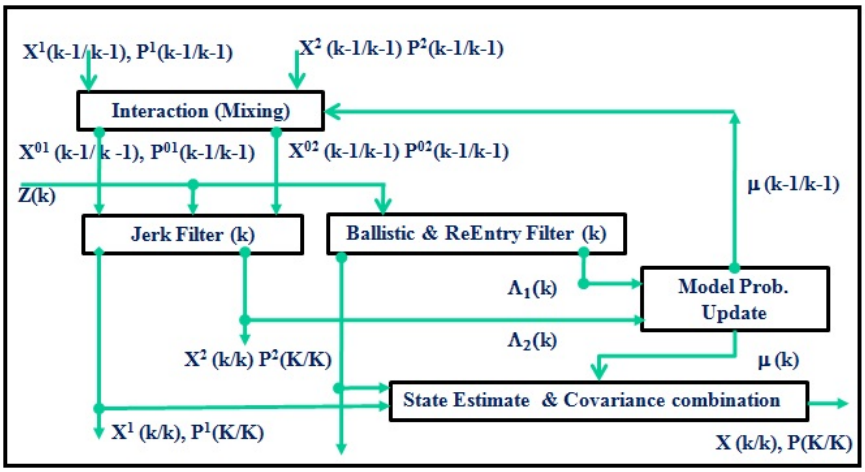

Figure 2.9: One cycle of Interactive Multiple Model(IMM)

be explained as follows:

$$\hat{x}^{j}_{k/k}, P^{j}_{k/k} : State\ estimate\ of\ j^{th}\ KF\ at\ time\ k\ and\ its\ covariance$$

$$\hat{x}^{0j}_{k/k}, P^{0j}_{k/k} : Mixed\ initial\ condition\ for\ the\ j^{th}\ KF\ at\ time\ k$$

$$\hat{x}_{k/k},\ P_{k/k} : Combined\ state\ estimate\ and\ its\ covariance\ (output)$$

$$\mu_k : Probability\ at\ time\ k$$

$$\mu_{k/k} : Mixing\ probability\ at\ time\ k$$

$$\Lambda_j(k)\ : Likelihood\ function\ of\ j^{th}\ KF$$

In the interactive multiple model(IMM) based filtering, the different filter models interact at input and output. The boost model described earlier contains 12 states; whereas the other model contains only 7 states. This type of asymmetric mixing can be done by inserting zeros, where the states are different in two models.

2.3.8 Results

The system model pertaining to jerk model(boost phase) and ballistic/re-entry Model (re-entry phase) along with overall IMM formulation have been discussed in Section 2.3.4 and Section 2.3.5. Based on this formulation, IMM with two EKF formulations for the above mentioned system model have been mechanized. For validation of this filter, the BM trajectory has been generated by using the 6 DOF simulations. Based on this data, radar measurements (r, θ, ψ) have been generated. In the present context, the measurement update rate is 10 Hz. The radar noise specification along range, azimuth and elevation as σ_r = 20m, σ_θ = 2 milliradian (mrad), σ_ψ = 2 mrad is assumed. The filter errors in terms of position, velocity and acceleration errors along (X,Y,Z) axes along with one sigma bound are shown in Figure 2.12, Figure 2.13 and Figure 2.14. All results are in ECEF coordinate frame of reference.

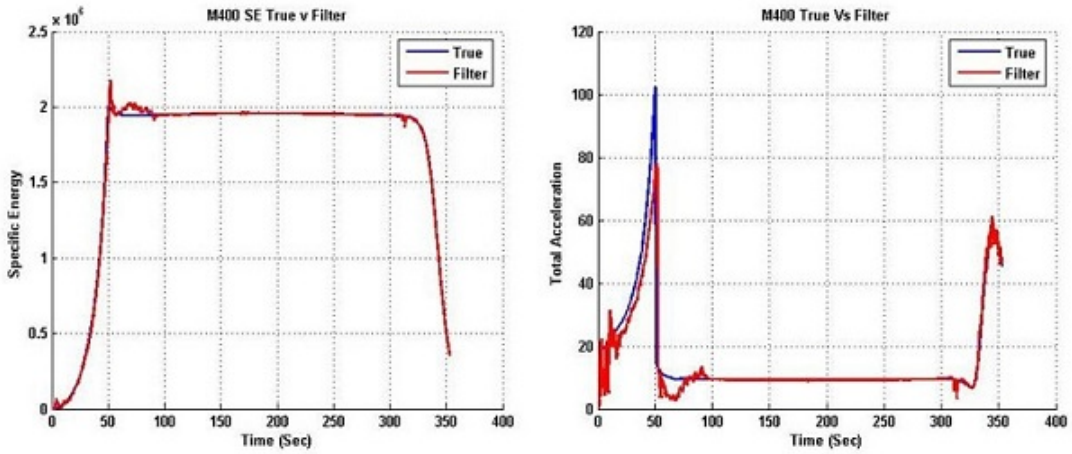

Figure 2.10: a) Time Vs Specific Energy b)Time Vs Acceleration

Figure 2.10(a), Figure 2.10(b), Figure 2.11(a) and Figure 2.11(b) show SE, Acceleration, altitude and velocity respectively for noisy as well as filtered data. Figure

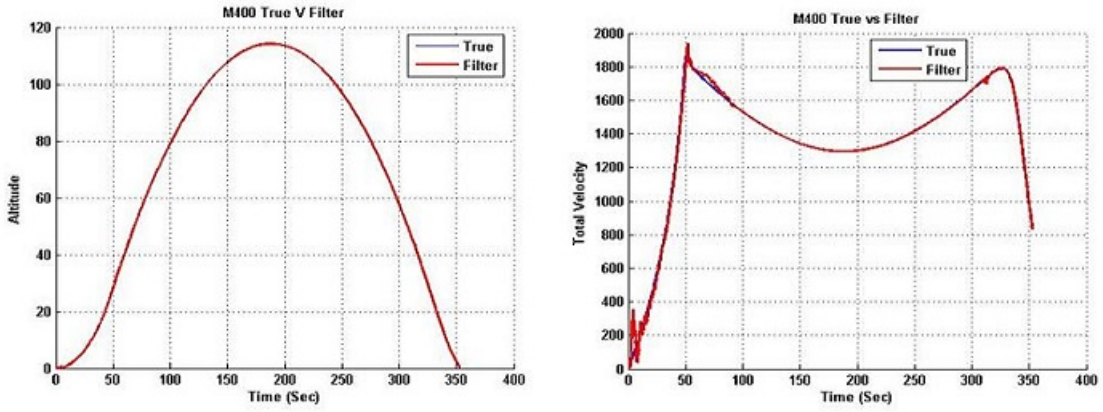

Figure 2.11: a) Time Vs Altitude b) Time Vs Velocity

2.12(a), Figure 2.12(b) and Figure 2.13(a) show the time history of position errors along (X, Y, Z) direction. The estimates from jerk model, ballistic/re-entry model and Interactive Multiple Model(IMM) as combination of both are superimposed together in the plot. In the same plot, time-history of measurement error along with one sigma standard deviation of the estimation error corresponding to IMM are shown.

Similarly time-history of velocity errors along (X, Y, Z) direction along with one sigma standard deviation of estimation error are shown in Figure 2.13(b), Figure 2.14(a) and Figure 2.14(b).

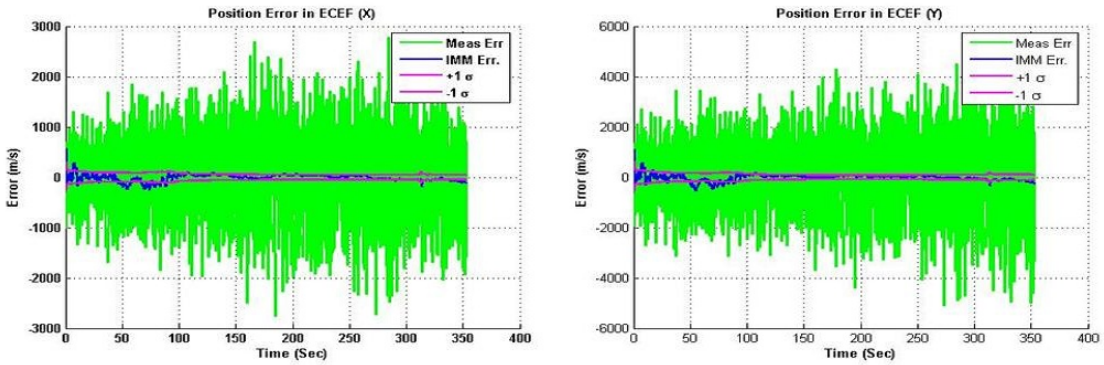

Figure 2.12: Position Error in a) ECEF(X) b) ECEF(Y)

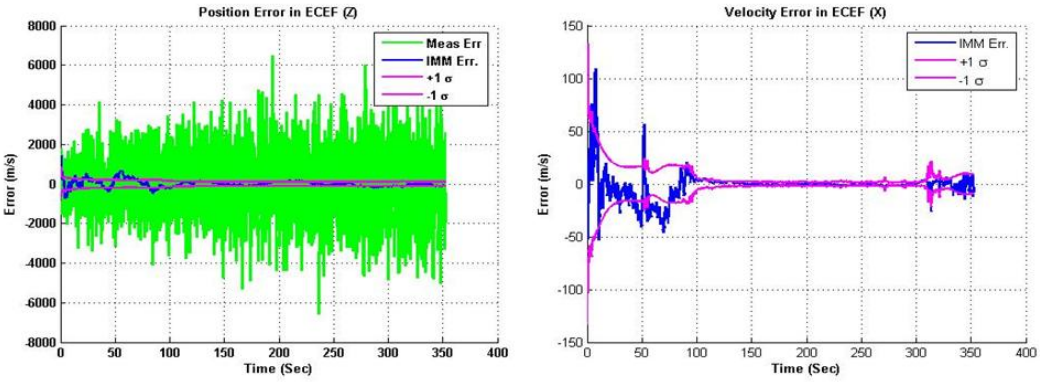

Figure 2.13: a) Position Error in ECEF(Z) b) Velocity Error in ECEF(X)

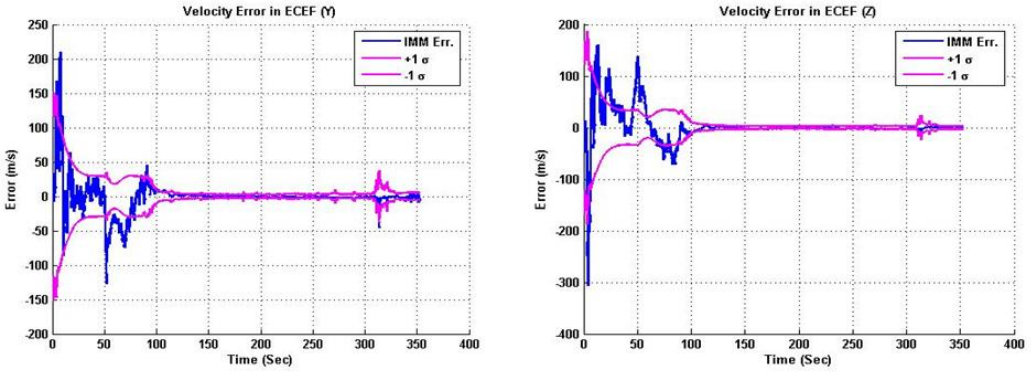

Figure 2.14: Velocity Error in a) ECEF(Y) b) ECEF(Z)

2.4 Input Data Generation and Processing

A moving object location is recorded by a set of sensors in the three dimensional space-time at each successive time instant. The result of this trajectory generation process is a two-dimensional N-tuple corresponding to the x and y-axis projections of the object centroid at each instant of time.

2.4.1 Input Data Analysis of Aerial targets

It is pertinent to know characteristics of targets so as to classify them. Main characteristics, their features and zone of operations are discussed in brief in this section.

Possible types of aerial targets and their Characteristics There are various types of targets in the air. They are fighter aircraft, commercial aircraft, helicopter, unmanned air vehicle (UAV), jammer aircraft, satellites, cruise missile, ballistic missile, air-to-surface Missile, anti-radiation missile and air-to-air missile. The characteristics of these vehicles can be summarized as follows :

1. Fighter Aircraft
    - Capable of doing a variety of manoeuvers
    - Generally attack as a pack (in a formation)
    - Cluster testing would be useful in identifying a fighter plane
2. Commercial Aircraft
    - Follow only chartered flight paths and fly alone
    - Have a maximum gravity turn of 1 g
3. Helicopter
    - Helical Motion to Move Up or Down
4. Un-Manned Air Vehicle (UAV)
    - Are very light, don't not move very fast and hovers around the target
    - General flying altitude is around 2000 m
    - Can have straight, surveillance manoeuvers
5. Jammer Aircraft
    - Carries out armed reconnaissance, electronic warfare and jamming operations
    - Uses powerful Continuous wave transmitters to direct the jamming signal at the threat
    - The aircraft cruises at 775 Km/h and maximum altitude is 12550 m.

6. Satellite
   A satellite is an artificial object which has been intentionally placed into orbit. Operational satellites are in placed in three orbits: namely low-earth orbit (altitude between 160 Km to 2000 Km), medium-earth orbit (at 20,000 km), and geostationary orbit (at 36,000 km). The orbital velocity needed to maintain a stable low earth orbit is about 7.8 km/s.

7. Cruise Missile

   - In most ways, similar to fighter planes and can do all the moves of a fighter plane
   - The differences are that it has constant velocity (i.e. 0 acceleration) and it flies alone
   - The flight has 3 stages
     - The initial launch stage where it goes to a very high altitude
     - The DIP stage where it directs itself in the direction of the target and dips down to a very low altitude
     - The final stage where it flies low with constant velocity till it strikes the target

8. Air-To-Surface Missile

   - These are launched from fighter planes
   - Once launched, these follow projectile motion

9. Anti-Radiation Missile

   - Launched form fighter planes
   - Seeks the source of radiation (RADAR) and strikes it

10. Air-To-Air Missile

    - Launched from air
    - High initial velocity

11. Ballistic Missile
    - High Initial Velocity During Propulsion
    - Controlled Vehicle fall Under Gravity after Boost

2.4.2 Different Parameters Of Targets

- Velocity (V) : Velocity is rate with which displacement takes place in a direction. Besides total velocity of the body, the three related components, velocity Component-X, velocity Component-Y, and velocity Component-Z in body-frame can also be used for distinction.
- Ground velocity: Ground speed is the horizontal speed of an aircraft relative to the ground. An aircraft heading vertically would have a ground speed of zero.
- Acceleration : Acceleration is rate with which velocity (V) change occurs in a direction. Besides total acceleration of the body, its three components Acceleration-X , Acceleration-Y, Acceleration-Z in body-frame can also be used for distinction.
  $Acceleration(A) = V^2/R$
- Jerk Factor : Jerk is the rate of change of acceleration; that is, the derivative of acceleration with respect to time. Jerk is a vector. The SI units of jerk are meters per second cubed, m/s^3.
  $jerkfactor = (Acceleration2 - Acceleration1)/(Time2 - Time1)$
- Latitude and Longitude Positions : Latitude and longitude are angles that uniquely define points on a sphere. Together, the angles comprise a coordinate scheme that can locate or identify geographic positions on the surfaces of planets such as the earth.
- Rate of Ascent /Descent : The rate of climb (RoC) is a flight's vertical speed i.e. the rate of change in altitude. It is commonly expressed in metre per second (m/s). The rate of decrease in altitude is referred to as the rate of descent or sink rate. A decrease in altitude corresponds with a negative rate of climb.
  Rate of Ascent/Descent = $(Altitude2 - Altitude1)/(Time2 - Time1)$

- Altitude : Altitude is a distance measurement, usually in the vertical or ”up” direction, between a reference datum and a point or object.
- Radius of turn : The turning radius of a vehicle is the radius of the smallest circular turn (i.e. U-turn) that the vehicle is capable of making.
- g-Turn : It is maximum acceleration with which a body can take a turn with respect to gravity. It is expressed in terms of meter per square second.
  g-Turn = $A/9.8ms^{-2}$
- Cluster-id : A cluster is defined as a concentration of enterprises producing same or similar products and is situated within a contiguous geographical area spanning over a few meters. If multiple objects emanate from same parent body, all these objects are provided with same cluster-id.

2.4.3 Parameter Based Distinction

There are number of parameters which can be utilised for distinction of types of vehicles. The parameters which distinguish one vehicle from another vehicle are range of operational velocity, altitude, manoeuvrability and rate of climb/descent.

Table 2.1: Parameters based categorization

Sl No	Descriptor of Vehicle	Velocity Range (Km/Hr)	Altitude Range (Km)	Manoeuver-ability	Rate of Climb/ Descent (mps)
1	Helicopter	<200 To 400	< 1 to 4	< 2g to 5 g	< 50
2	Transport	200 to 1000	< 1 to 14	< 2g to 3 g	< 50
3	Civil	400 to 1000	4 to 14	< 2g to 2 g	< 50
4	UAV	<200 to 400	< 1 to 4	< 2g to 5 g	< 50
5	Cruise missile	200 to 1000	<1 to 4	< 2g to > 5 g	< 50
6	Fighter	200 to >1000	< 1 to 14	< 2g to > 5 g	< 50 to 400
7	Jammer	400 to 1000	1 to 8	< 2g to >5 g	<50 to 400
8	Air-2-Ground	800 to > 1000	< 1 to 8		< 50 to 400

9	Anti-Radiation-Missile	800 to >1000	< 1 to 14		< 50 to 400
10	Ballistic Missile	800 to >1000	>50	< 2g to > 5 g	>400

Distinction Based on velocity Velocity is an important parameter for distinction of vehicles in air. Velocity of helicopter and UAV ranges from less than 200 KM/Hour to 400 Km/Hour. Civilian Aircraft range of velocity varies from 400 KM/Hour to 1000 Km/Hour whereas velocity of air transport vehicle varies from 200 KM/Hr to 400 Km/Hr. Cruise Missile velocity ranges from 200 KM/Hr to 1000 Km/Hr. Lower velocity values for fighter aircraft, jammer missile, air-to-ground missile and anti-radiation missile are 200 KM/Hr, 400 Km/Hr, 800Km/Hr and 800 Km/Hr, whereas higher velocity values are 1000 or greater than 1000 Km/Hr as shown in Table 2.1.

Distinction Based on altitude As shown in 4^{th} column of Table 2.1, altitude of operation of UAV, Helicopter and cruise missile are less than 4 KM. Jammer and Air-to ground missile altitudes are ranging 1 Km to 8 KM. Civilian aircraft, transport aircraft, fighter and anti-radiation missile flies less than 14 Km. Altitude of ballistic missiles are approximately one-fourth of down-range coverage [37]. Ballistic Missiles altitude varies from 75 Km to 1000 Km for 300 KM Class of Missiles to 5000 Km of Missiles.

Distinction Based on manoeuvrability As shown in 5^{th} column of Table 2.1, manoeuvrability of helicopter, UAV , Civilian, Transport aircraft are less than 3g. Fighter aircraft, jammer, Air-to-Ground and Anti-radiation missile are upto 5 g. Ballistic missile will have manoeuvrability upto 15g.

Distinction Based on climb/descent Rate of Climb and Descent of UAV, Helicopter, civilian Aircraft, Transport aircraft are upto 50 meter per second. For fighter aircraft, jammer, Air-to-Ground and Anti-radiation missile, rate of climb/descent are upto 400 mps. Ballistic missile will have rate of climb/descent upto 7000 mps.

Table 2.2: Class of Missiles and their Important Parameters

Class/ Range (Km)	Time of Flight (Sec)	Apogee (Km)	Velocity at burn out (Km/s)	Range at burn out (Km)	Altitude at burn out (Km)	Time at Burn Out (Sec)
M300	240	60	1.55	40	35	35
M600	350	120	2.20	50	45	40
M1000	440	200	2.84	80	55	45
M2000	630	400	4.02	120	100	60
M5000	990	1000	6.36	200	130	110

2.5 Time Domain Analysis

Class of missiles with their important parameters are shown in Table 2.2. First column shows class of missiles with down range of 300 Km, 600 Km, 1000 Km, 2000 Km and 5000 Km. Other columns indicate time of flight, apogee of trajectory, velocity, range, altitude and time at burn-out for the above mentioned class of missiles. These parameters pertain to generic missiles and can vary for specific missiles. There are two classes of interceptors, one which can intercept in endo-atmospheric region and other in exo-atmospheric region. Approximate time of first detection for targets is 50 to 60 seconds, with probable kill bracket of 100 to 300 Km. The classification time for target is from 30 to 50 second for intercepting in exo-atmospheric region. The response time for this class of interceptor is 20 to 100 seconds. Similarly, for interceptor of endo-atmospheric region, though classification time is large enough but there is very less time for salvo-launch because of low altitude of interception and small kill-bracket.

Response Time of Interceptor: The response time of Interceptor depends on:

- **Range of Sensors and Its Placement:** Time-line is very much dependent on range of detection, searching and tracking of radar. Longer the range of radar, early the detection and more the response time we get. Similarly, placement of radar with respect to point of origination of hostile missile plays an important

role in early action. Figure 2.15 shows placement of three radars, two of them as long-range radars and one as short range radars. It is obviously an advantage to

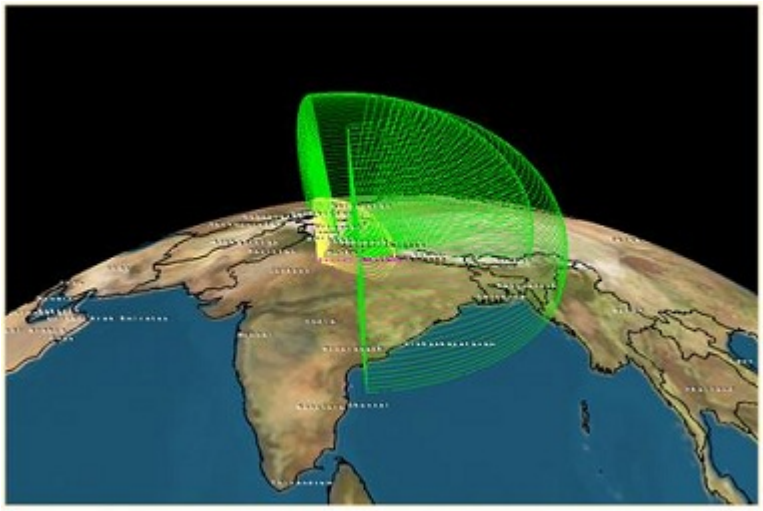

Figure 2.15: Coverage zone of Radars

place the radar in forward region to cover regions of origination of target. Figure 2.16 and Figure 2.17 shows detection by radar of 600Km of acquisition range placed at the point of impact of target for detection and tracking of 1000 Km and 1500 Km class of Target.

- **Auto-launch time (onboard systems' readiness):** Response time also depends on boot-up time of the sub-systems of interceptor missiles, INS readiness (DTG-dynamically tuned gyros are accurate but heavy and take long initialization time), ground-based computational algorithm convergence time, data loading time, etc.
- **Height of kill and footprint to be defended:** Footprint is defined as an area which is projected on the ground so that it becomes a tangible measure of the defence. 'Defence-in-Depth' and 'keep-out of range' determines 'safe boundary' around the asset to be defended [37]. Defending footprint diagram is shown in Figure 2.18. Higher the height of kill, larger will be the defended foot-print and less likelihood for releasing multiple warheads. This will also ensure that the debris falls in enemy area rather than in the defended area.

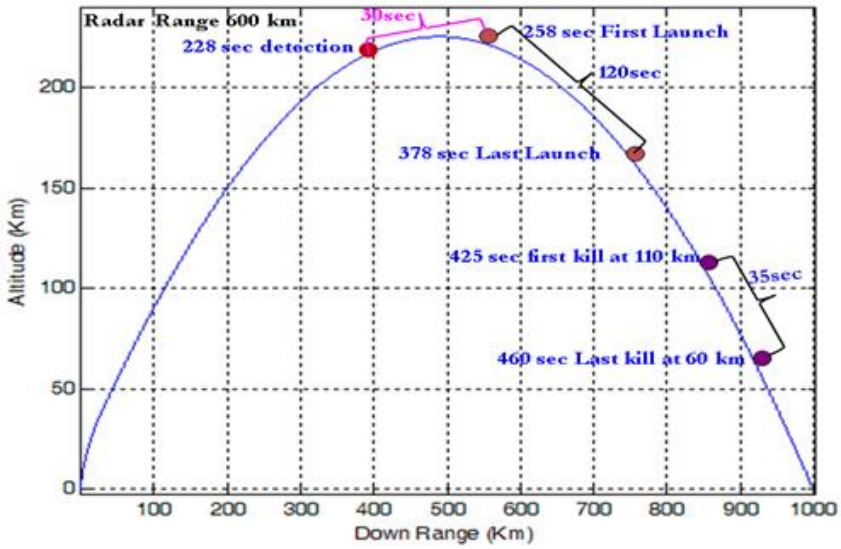

Figure 2.16: M1000 Class of Missile Engagement Scenario and Response Time

- **Velocity of the Interceptor:** Higher the velocity of interceptor, lesser is the time to reach to the target. Faster interceptor is preferable but it also requires that the response time for classification needs to be fast.

- **Weight of the Missile:** Weight of missile plays very critical role as more the weight, more will be energy required and less will be agility of the vehicle. More fuel requires larger propulsion system which itself adds to overall weight. Thus, a right trade-off is required between capability, propulsion system, airframe, warhead and other systems.

- **Policy of Launch:** A perfect interceptor is the one that hits and kills its target with a probability of one. However, practical interceptors have a nonzero probability of failure to kill. Such a failure can be caused by an inaccurate interception (i.e., the interceptor misses the target) or by insufficient destructive capability. To ameliorate this shortcoming, the defense can choose to fire several interceptors at once at each object called shoot-shoot-shoot philosophy, or shoot one interceptor, perform a kill assessment, and then shoot at the object again if necessary, called Shoot-Look-Shoot policy [62, 63].

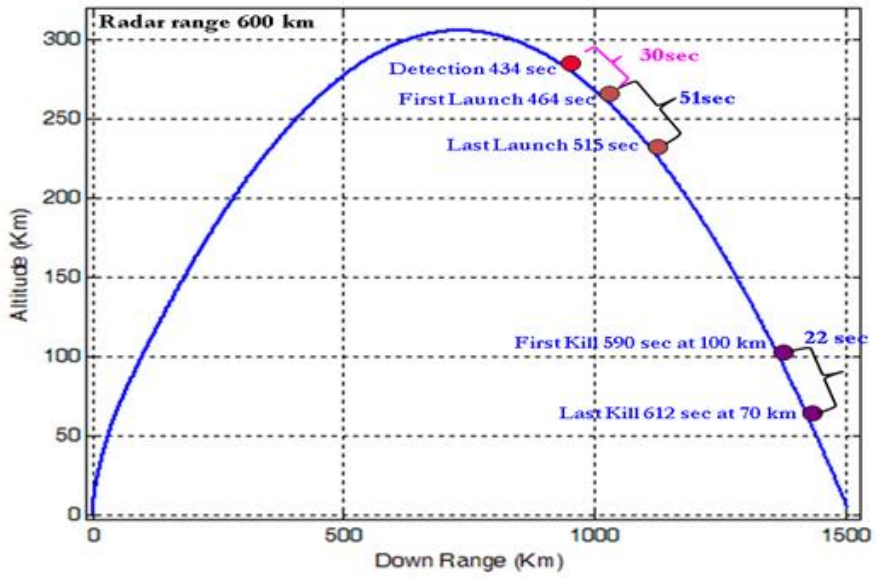

Figure 2.17: M1500 Class of Missile Engagement Scenario and Response time

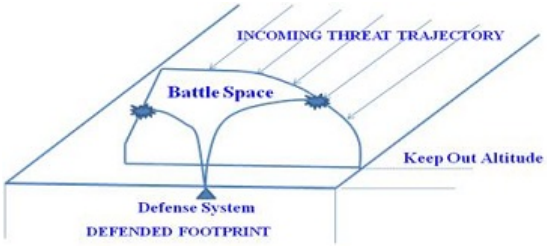

Figure 2.18: Defended Footprint and Threat Trajectory

- Other factors which affect response time are: 1) division of time slot for boost and coast time 2) whether configuration of vehicle is single stage or Multi-Stage 3) how much is lateral-acceleration capability (Latax) of the vehicle and 4) how much is networking capability between ground systems for close coordination and communication.

Response Time Analysis: A simulation study with 6-degrees-of-freedom for a given radar range for following vehicle characteristics of targets and interceptors are carried out to analyse response time requirement of interceptor:

- Radar range of 600 Km

  Target Characteristics (1000 Km)

  Target of 1000 Km down range is considered with single stage vehicle configuration. Its burnout velocity is of the order of 2.8 Km/sec at approximately 62 sec from launch. Velocity of target at interest zone of interception is of the order of 2.1 Km/sec.

  Interceptor Characteristics and Response time

  Interceptor is considered as two stage vehicle having booster with propulsion and kill-vehicle without propulsion. Booster burning time considered is 62 seconds with separation height of 35 to 38 Km, cut-off velocity of of the order of 1.8 Km/sec and interception altitude bracket of 60 to 110 Km. Kill-vehicle will have capability to coast for minimum period of 35 sec and maximum duration of 105 seconds. Thus, total time of flight for interceptor is 97 sec to 167 sec. Figure 2.16 shows the down range versus altitude trajectories of target with important events of interceptor marked on it. In this scenario, target is detected at 228 sec of the launch of the target. If we want to intercept at height of 110 Km, launch of interceptor has to commence at 258 sec considering maximum flight of interceptor as 167 sec as given above. Thus, response time of interceptor from detection to launch is only 30 sec. If we want to intercept the target at an altitude of 60 Km, the launch time changes to 378 sec gaining a response time of 150 sec.

  Target Characteristics (1500 Km)

  Target is considered again as single stage vehicle. Burnout velocity considered for 1500 Km class of vehicle is of the order of 3.2 Km/sec at approximately 100 sec from launch. Velocity of target at interest zone of interception is of the order of 2.8 Km/sec.

  Interceptor Characteristics and Response Time

  Keeping everything unchanged for interceptor parameters, if a class of 1500 Km range of target has to be intercepted with minimum response time of 30 sec for first launch, then interception at 110 Km is not possible. Figure 2.17 shows the first launch possibility at 464 sec from launch of the target. As target is flying with higher velocity, kill is not possible at 60 Km. Making kill altitude of 70 Km, the time difference between first and second launch reduces from 120 sec

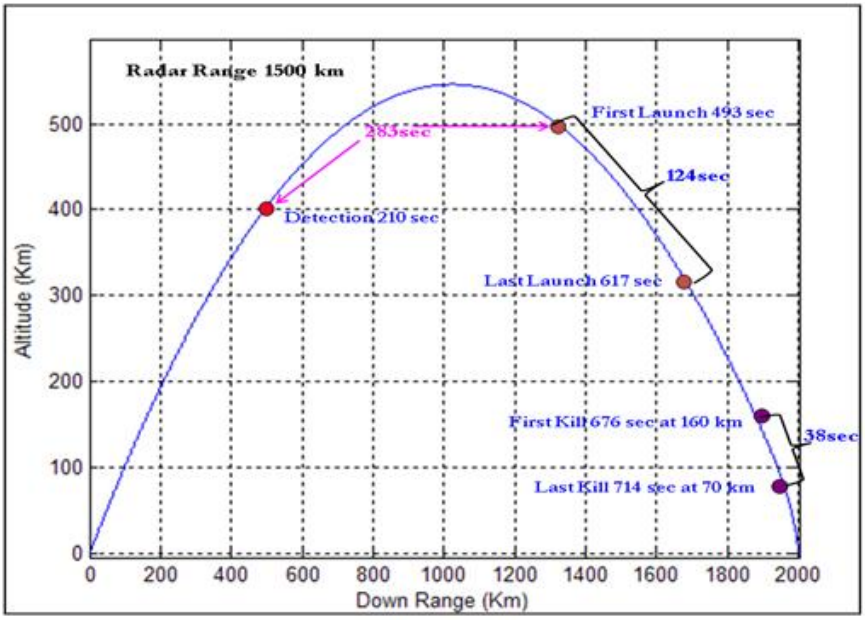

Figure 2.19: M2000 Class of Missile Engagement Scenario and Response time

to 51 sec. And time difference between first kill and last kill altitude reduces from 35 sec to 22 sec.

- Radar range of 1500 Km
  Another experiment is considered with radar acquisition range of 1500 Km.
  Target Characteristics (2000 Km)
  Target is considered as single stage vehicle. Burnout time considered for 2000 Km class of vehicle is of the order of 4.1 Km/sec at approximately 110 sec from launch. Velocity of target at interest zone of interception is of the order of 4.0 Km/sec.

  Interceptor Characteristics and Response Time
  Interceptor is considered as two stage vehicle with booster with propulsion and kill-vehicle without propulsion. Kill-vehicle will have capability to coast for minimum period of 35 sec and maximum duration of 121 seconds. Thus, total time of flight for interceptor is 97 sec to 183 sec. Booster burning time considered is 62 seconds with separation height of 35-38 Km, cut-off velocity of 1.8 Km/sec and interception altitude bracket of 70 Km to 160 Km. Figure 2.19

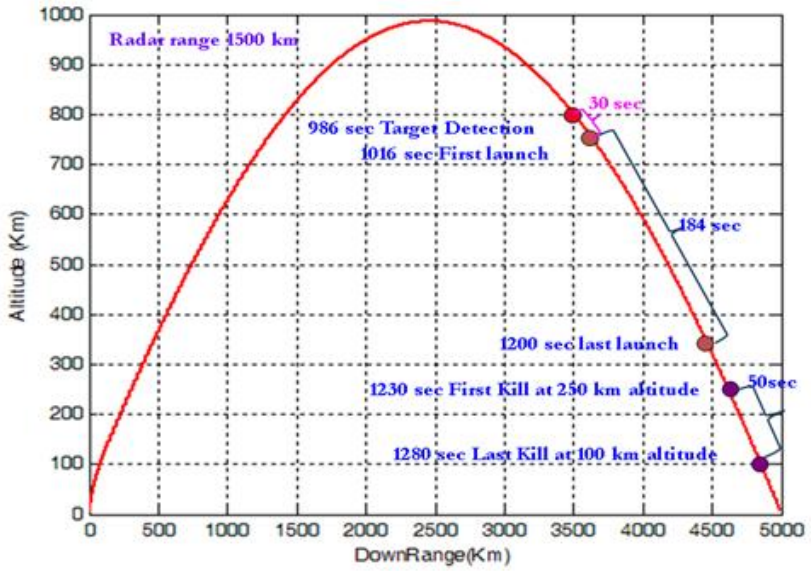

Figure 2.20: M5000 Class of Missile Engagement Scenario and Response time

shows the down range versus altitude trajectories of target and interceptor. In this scenario, target is detected at 210 sec of the launch of the target. If we want to intercept at height of 160 Km, launch of interceptor has to commence at 493 sec considering maximum flight of interceptor as 183 sec as given above. Thus response time of interceptor from detection to launch is 283 sec. If we want to intercept the target at an altitude of 70 Km, the launch time changes to 617 sec gaining a response time of 124 sec.

Target Characteristics (5000 Km)

Target is considered as multi-stage vehicle. Burnout velocity considered for 5000 Km class of vehicle is of teh order of 5.8 Km/sec at approximately 140 sec from launch. Velocity of target at interest zone of interception is 4.7 Km/sec. With 1500 Km range of Radar, there is huge difference in response time for M2000 and M5000 class of missile. Figure 2.19 and Figure 2.20 show that response time has decreased from 283 sec to 30 sec.

Interceptor Characteristics and Response Time

Interceptor is considered as three stage vehicle with booster and sustainer with propulsion and kill-vehicle without propulsion. Kill-vehicle will have capability

to coast for minimum period of 48 sec and maximum duration of 182 seconds. Thus, total time of flight for interceptor is minimum of 80 sec and maximum of 214 sec. Booster burn out height considered is 2 Km for 8 sec, sustainer separation height of 40 Km for 24 sec at cut-off velocity of 2.5 Km/sec and interception altitude bracket of 100 to 250 Km. Keeping radar range of acquisition as 1500 Km, if a class of 5000 Km range of target has to be intercepted with minimum response time of 30 sec for first launch, then interception at 250 Km of altitude can be done only if interceptor is launched at 1016 sec from launch of target. Figure 2.20 shows the last launch possibility at 1200 sec from launch of the target to kill at height of 100 km. As target is flying with larger velocity, kill is not possible below 100 Km.

Thus, it is clear from the above discussion that time of classification is very critical to the decimation of target missile. Though, there are other factors like range of radar and its placement, which can govern the response time of the interceptor, they are static and cannot be changed dynamically. Once target trajectory is available from radars, it is the classification method which primarly defines the response time of launch of the interceptor.

2.6 Six Degree of Freedom (6-DOF)

Input data for training and testing for missile trajectory is either taken from radar for opportunity targets or generated from 6-DOF model. A missile is a physical entity whose motion and orientation are controlled to intercept a target entity with a specified accuracy. The basic elements of a missile include an airframe, inertial sensors to stabilize the attitude (angle) and assist in guidance, a flight controller and models external to the missile such as the launch platform (cold launch or hot launch), and any environmental models (aerodynamic model, gravity model, etc) [4, 25]. Sub-Modules of 6-DoF are propulsion, gravity, aero-dynamics, atmospheric and missile configuration as depicted in Figure 2.21. External disturbance due to wind and misalignment is also considered. Runga-Kutta 4 is used for solving the 6 equations, 3 for translational and 3 for rotational motions of the body. Missile systems can be described by nonlinear differential equations, partial differential equations, and/or discrete-time equations.

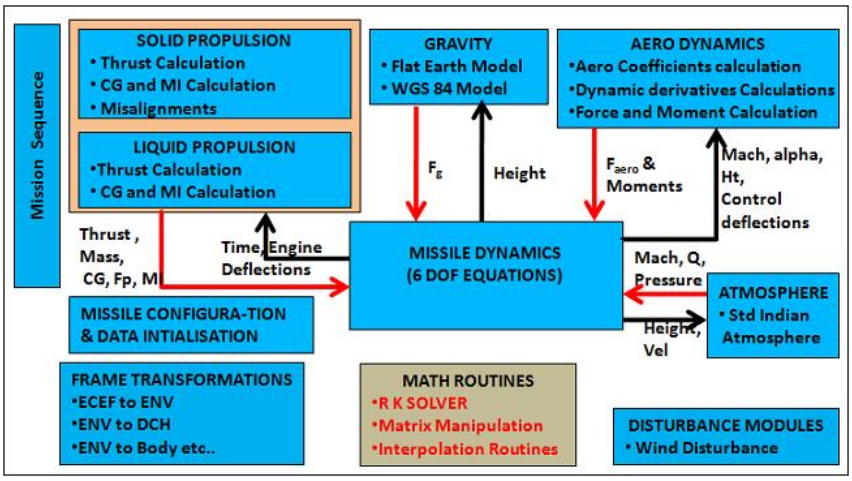

Figure 2.21: Six DOF Components

These models may encompass high-fidelity aerodynamics involving tables of wind tunnel measurements, time-varying propulsion characteristics, digital autopilots, inertial sensors, communication links, and one or more guidance laws.

2.6.1 Propulsion Module

Propulsion provides the missile thrust forces, moments, and mass variations. As can be seen from Figure 2.21, solid or liquid propulsion system can be used. Thrust-time profile is generated with actual firing of propulsion systems and the data generated is used in the model. A table of empirical data is used for accounting CG/MI variations due to the movement of the liquid propellant in the tank (sloshing effects).

2.6.2 Gravity Module

The gravity of earth, denoted by g', refers to the acceleration that the earth imparts to objects on or near its surface. It has an approximate value of 9.81 m/s^2 on the earth's surface. Its value for any height can be calculated from equation: $g' = r/(1+h/Rg)^2$, where g' is the gravity measure at height above the sea level, 'g' is the standard value of the gravity at the surface of the earth (9.8 m/s^2). Rg is the earth's mean radius and 'h' is the height above the surface.

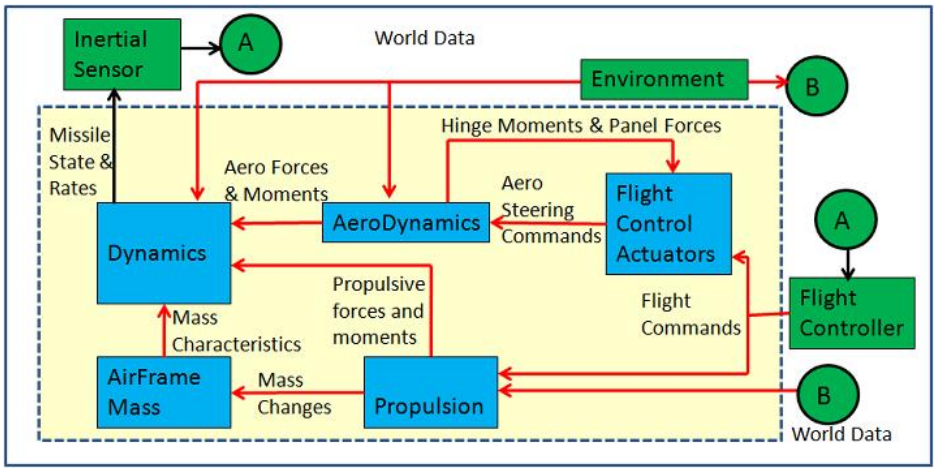

Figure 2.22: Air-frame Model with Inertial System and Flight Controllers

2.6.3 Aero-Dynamics Module

The aero module computes the various aero forces and moments generated by the vehicle movement. An airframe model computes missile states from forces and moments. Missile states are defined as those attributes that define missile translational and rotational motion: typically, they are the cartesian position, velocity, the quaternion (for orientation), and the angular velocity. The airframe receives steering commands from the flight controller which are then realized as forces and moments by the actuators. The airframe model has several components: missile dynamics, airframe mass, aerodynamics, propulsion, and flight control actuators (Figure 2.22). Dynamics are the continuous-time missile states (position, velocity, orientation, and angular rates) and state derivatives (velocity, acceleration, angular rates, and angular accelerations) for the given forces, moments, and mass characteristics.

Usually, the moments act about the center of mass of the rigid body so there is no gravity moment (in a uniform gravitational field). Airframe mass calculates the missile characteristics (e.g., center of mass and moment of inertia) as inputs to the dynamics. Aerodynamics computes the aerodynamic forces and moments as functions of the flight conditions and control surface deflections, i.e., the forces and moments are obtained via lookups and interpolation in multidimensional tables.

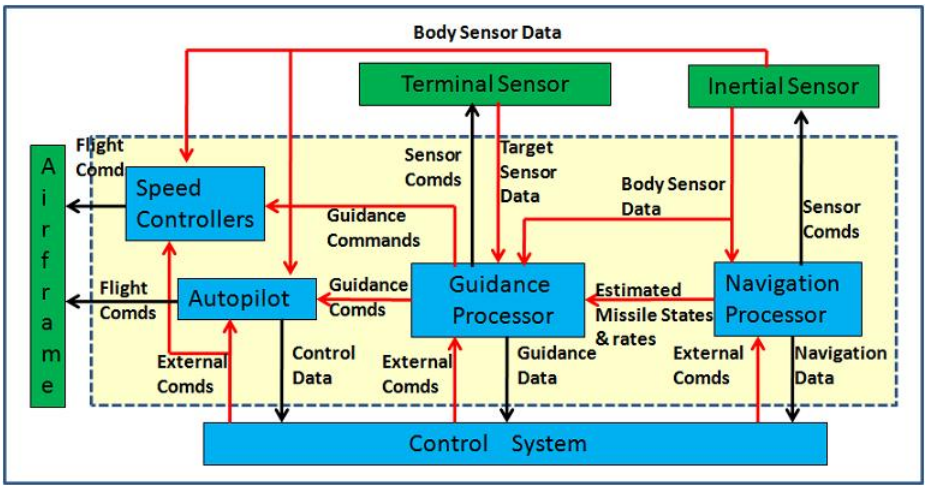

Figure 2.23: Flight Controller

2.6.4 Atmospheric Module

The properties of earth's atmosphere like pressure, temperature and density vary not only with height above the earth's surface but also with the location on earth, from day to day and even during the day. A hypothetical model is employed as an approximation to what is expected. This model is called *standard atmosphere*. The air in this model is assumed to be devoid of dust, moisture and water vapour and to be at rest with respect to earth.

2.6.5 Flight Controller

Flight control actuators are those mechanisms that physically realize the control commands, e.g., control surface deflections. For an endo-atmospheric system, the control surfaces typically are tails, ailerons, or canards, but other flight control systems may include thrust-vector control vanes or pulse-width modulated thrusters. The airframe provides truth information to the inertial sensors (typically, accelerometers and rate gyros) and interacts with the physical world through environmental objects such as gravity and atmosphere (Mach, speed of sound, temperature, wind, weather, etc.). The inertial sensors measure the airframe$'s$ motion and feed it back to the flight controller.

The flight controller consists of the navigation processor, the guidance processor, the autopilot, and the speed controller (Figure 2.23). Typically, the navigation pro-

cessor consists of an inertial navigation system (INS). The INS provides estimates of the missile states to the guidance processor based on measurements obtained from rate gyros and accelerometers (the inertial sensors) but may include processing of communication uplinks or possibly GPS measurements. The guidance processor consists of the guidance law(s) and the guidance filter(s). The type of autopilot modelled is the standard proportional autopilot with body acceleration feedback as the main feedback controlling the loop gain and body rate feedback controlling the damping [21]. The guidance filter processes the measurements from the terminal sensor(s) and inertial sensors to produce well-behaved estimates of the target states. The guidance law(s) combines these with the estimated missile states to produce steering commands. There may be a different guidance law and filter for each phase of flight. The autopilot transforms the steering commands to control surface commands for the airframe. In addition, the autopilot may have multiple control options based on the phase in flight. If the missile motor uses solid propellant, the speed controller is omitted; otherwise, it typically throttles the engine to achieve the required speed. All of these flight control functions are, of course, critical elements in a 6-DOF simulation, but they cannot act alone, so the other subsystems and external models are implemented to simulate the actual system's operation.

2.6.6 Missile Configuration

Missiles can be configured to have single stage or two stages with the second stage in powered mode or coasting. Vehicle can have more than two stages also. In the case of exo-atmospheric region, reaction control systems are incorporated for pitch, yaw and roll control as fins will not be effective. The mission sequence of the vehicle can be programmed as a user desires.

2.7 Related Work

Currently, the conventional methods using correlation based logic are able to deliver results up to 85% confidence level [50]. The success of these statistical or template matching methods depends highly on a-priori knowledge of range of values of param-

eters being used and the probability model used for the distribution of the parameters in problem space.

In the domain of ballistic missile defence, *classification* of ballistic missile trajectories have received much attention [1, 14, 20, 23, 47, 54, 61] and at the same time much advancement has also been made at the theoretical level [10, 30, 57, 63]. One of the major drawbacks involved in the target classification models as outlined in the above mentioned works is that most of the results are based on *off-line data* for *non-real time* classification. Recent research activities in neural classification [68] have shown promising alternatives. For instance, Tamura, et. al. [56] introduced the idea of quantizers to show that a four-layered feed-forward neural network with (N/2)+3 hidden units can give any input-output relations with an arbitrarily small error for N samples of input vectors. Huang [27, 29] extended the idea for utilizing the network for real-time computation and used numerous real-world regression data sets to verify the performance of their algorithm.

Silberman [54] describes a generic parametric approach to build classifier models, which is illustrated with an example of building a classifier for an infrared sensor. Approach in the paper is to characterize how each object class will appear to a given sensor as a function of engagement geometry, object dynamics, object properties, sensor noise, and observation period. An ensemble of time series signatures is generated for each object class by varying the parameters for those inputs that will be unknown during an engagement, while fixing those parameters for those inputs that will be known. Though the approach is generic, it is non-realtime classification and too many signatures have to be accounted to arrive at the decision. Performace for clasificaton is not mentioned in the paper.

Classification of *motion* trajectories on recordings from visual surveillance is presented in [8, 35, 42, 43, 48] and trajectory classification/modelling based on hidden Markov models (HMM) [7, 19, 49] as well as Gaussian mixture-based HMMs are described in [5, 40, 41]. All these works are about trajectory classification for either *pedestrian activities* or *moving ground vehicles* but none of them talk about trajectory classification for ballistic missiles.

Classification of moving targets based on motion and appearance by extracting static and dynamic characteristic features has been carried out in [8]. These features

are in turn used to assign the targets to one of several predefined categories. Background subtraction, connected component analysis, morphological filtering and temporal differencing of consecutive frames are the methods used. It classifies human, vechicle, animals, rigid, non-rigid objects and its hybrid with 87% to 99% confidence in non-realtime mode.

In [43] attempts are made to classifiy and track moving ground vehicles by combining kinematic association hypotheses with accumulated target classification information obtained from high range resolution (HRR) radars. Inverse synthetic aperture radar (ISAR) and synthetic aperture radar (SAR) signatures are also used to obtain improved classification and association.

Trajectory between the current location and launch point is predicted by backward integration in time in [2]. However, in the free-flight phase, *launch, target points* and *full trajectory* cannot be predicted properly. Moreover, the authors claim that their model can predict within 1% error of its max range which means for 600 Km trajectory, there will be position error of 6 Km, which is not suitable for missile defence. Another related work [34] estimates the trajectory and launch point of a tactical ballistic missile using *Line of sight* (LOS) measurements from one or more satellite-based sensors. The work assumes that the *magnitude* and *orientation* of the missile thrust vector is known. Sensors are so far away from the target that a significant change in target position is reflected only as a slight variation in the LOS measurements, which results in poor observability of the target motion. Consequently the maximum likelihood trajectory estimation problem is ill-conditioned for most realistic target-sensor geometries.

Classification of ballistic missiles during ascent phase using specific feature of perceived trajectory motion is addressed in [46]. This paper assumes that ballistic missile has unique launch sequence of vertical launch phase, kick-turn phase and gravity turn phase. Altitude measured by radar is fitted as a cubic polynomial function of time and vertical plane. Threshold of error between reference (for max range) and measurements is determined by Monte-Carlo Simulation which requires many iterative runs for each missile trajectory. Further, assumption that ballistic missile is confined only in the vertical plane makes it too restrictive to fit into real missile trajectory. Another difficulty arises due to conjecture of ascent phase detection which is not practical due to placement of radar away from from the location of launch. Ascent phase detection and destruction of ballistic target has not been realised by any country. In all realised

systems, phase of detection and destruction is done during mid-course and early terminal phase and the classification technique proposed in this thesis tries to achieve this end.

2.8 Summary

This chapter gave an overview of ballistic missile defence scenario and the importance of *response time* in the context of missile defence. All factors affecting response time is described to bring out the importance of real-time classification. The trajectories obtained from radar suffer due to internal noise and bias of radar. Interactive Multiple Model (IMM) is developed to filter noisy trajectory data which becomes smooth and even for classification. Since large number of live data cannot be obtained from radar, a mathematical model of six degree-of-freedom (6DOF) is developed to generate trajectories of various types of missiles. Effective noise present in radar environment is imposed on true trajectories generated by 6-DOF to match it with real-radar environment trajectory. This chapter describes all pre-processing required after target trajectory data is captured by radar or generated by 6-DOF. It also presents parameters required for real-time classification by RTNN (Real Time Neural Network), HMM (hidden Markov model) and ART-2 methods which are explained 3rd, 4th and 5th Chapters. The EKF generated parameters from measured polar coordinates data are in form of position, velocity and acceleration. Besides using total acceleration, total velocity and absolute altitude, specific energy (SE) is computed to feed four parameters to the networks. Missile classification approaches by other authors and their limitations are mentioned in the last section of this chapter.

Chapter 3

Real-Time Neural Network

In the previous two chapters we have seen that the classification of ballistic missile trajectory is a challenging problem due to its time-varying dynamics and short response time available for interception. In general, classification approaches can be grouped as supervised/unsupervised, parametric/nonparametric and hard/soft (fuzzy) classification [66]. A normally distributed dataset is assumed to exist for parametric classifiers. The parametric classifiers assume that a normally distributed dataset exists, and that the statistical parameters (e.g. mean vector and covariance matrix) generated from the training samples are representative. However, the assumption of normal spectral distribution is often violated, especially in complex landscapes. In addition, insufficient, non-representative, or multimode distributed training samples can further introduce uncertainty to the classification procedure. Another major drawback of the parametric classifiers lies in the difficulty of integrating spectral data with ancillary data. The maximum likelihood is the most commonly used parametric classifier in practice, because of its robustness, simplicity and requirement of less number of variables.

With non-parametric classifiers, the assumption of a normal distribution of the dataset is not required. No statistical parameters are needed to separate classes. Much previous research has indicated that non-parametric classifiers may provide better classification results than parametric classifiers in complex landscapes [36, 45, 66]. Among the most commonly used non-parametric classification approaches are neural networks, decision trees, support vector machines, and expert systems. In particular, the neural network approach has been widely adopted in recent years. Conventional neural networks achieve learning through progressive adjustment of weighted interconnections

of neurons, guided by a learning algorithm [27]. Learning algorithms of neural networks widely use iterative searching methodology [17], e.g., backpropagation (BP) algorithm [16] and its variants [16, 39, 60, 67]. In each iteration gradual updation of parameters of the network is carried out and therefore learning becomes time consuming for applications that have large-scale observations and expect high accuracy. The problem with such kind of learning algorithms is that they need long hours for training and may have issues related to stopping criteria as well as may end up with local minima. Hence, these algorithms, if used in time critical applications might face unexpected difficulties. Therefore classification of variable-length trajectory and time varying dynamic attributes which are important ingredients for ballistic missile classification cannot be done using conventional techniques like neural networks

In this chapter, we outline a novel method for classifying ballistic as well as quasi-ballistic missiles using real- time neural network. "A neural network system is called a real-time learning system if it can finish a learning procedure with good generalization performance for a new application within expected fast response time defined by external requirements" [29]. The Real-time neural network (RTNN) has to react, solve and respond to external events within a given time-constraint. The time of reaction can range from microseconds, milliseconds or even seconds depending on a particular application. This expected response time could be any reasonable time for a given application that users can afford to wait without any side-effects due to time. Real-time learning can be widely used in HCI (human-computer interface), UAV (unmanned aerial vehicles), robotics, tracking and navigation, GIS (geographical information systems), expert systems, intelligent networking etc. It is also the case that for offline applications wherein a need for speed is crucial, real-time learning can help as it reduces training time as well as human effort. It has been reported in the Literature about multilayer feedforward networks which are capable of learning observations in a single iteration. It has been shown in Huang and Babri [28] that a "single-hidden layer feedforward neural network (SLFNs) with at most N hidden neurons and with almost any nonlinear activation function can learn N distinct observations with zero error". But the drawback is that for large-scale applications the required number of hidden neurons will be very large. Ordinary computers will not be able to support such computational capabilities.

RTNN should not be confused with recurrent trainable neural networks [22] (RNN). RNN's incorporates some form of Multi-Layer Perceptron as a sub-system and consists of a series of cascaded sub-networks, each of which consists of multiple layers of nodes. Each of these sub-networks is entirely feed-forward except for the last layer, which can have feedback connections among itself. Each of these sub-nets is connected only by feed forward connections. RNN's have scaling issues and cannot be easily trained for neither large numbers of neuron units nor for large number of input units. Successful training for RNN's has been mostly in time series problem with few inputs

Huang [29] proved in a novel constructive method that a two hidden layer feedforward neural network (TLFN) can learn any distinct samples (x_i, t_i) with any arbitrarily small error with at most $(2\sqrt{(m+2)N))}$ hidden neurons. For large-scale problems this method enables to reduce the space complexity of a network and thereby makes it possible for an ordinary computer to implement such large-scale systems. Here, (x_i, t_i) are input samples and desired outputs, where 'i' is ranging from 1 to N, number of distinct observation vector, 'm' is number of output nodes. Based on Huang's [29] constructive network model and real-time learning algorithm, a model has been developed in this chapter to classify aerial objects as well as ballistic missile trajectories. The details of the network, equations used and model formulations are discussed in the subsequent sections of this chapter.

3.1 Network Architecture

The Network consists of 4 layers, namely, one Input Layer(IL), two Hidden Layers (HL) and one Output Layer (OL). Data samples are taken through "input layer" (IL). The first HL contains standard neurons as well as quantizers Q_A and Q_B in pairs. All neurons in first HL are connected to IL but only to few neurons in second HL as demonstrated in Figure 3.1 All weights of the connections from input layer to first hidden layer (FHL) are assigned randomly. The weights of the connections linking the first hidden layer to second hidden layer can be determined analytically at one time. This is quite contrary to conservative iterative methods adopted in most artificial neural network(ANN). The weights of the connections linking the second hidden layer and the output layer can be analytically determined and remain as a constant value C. The

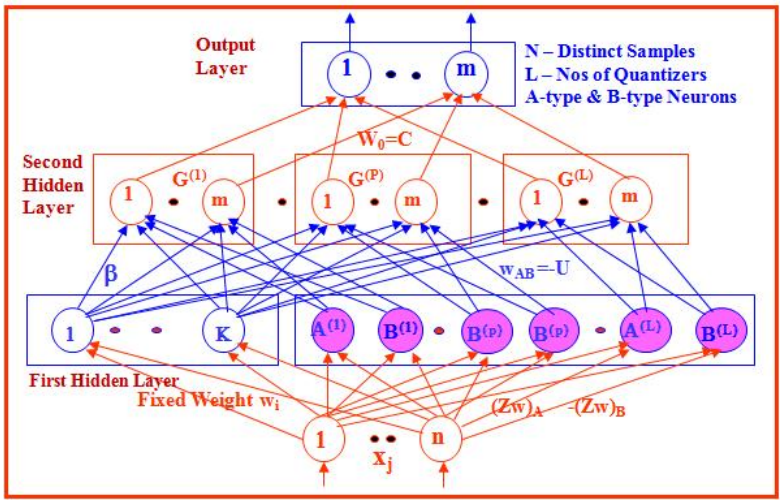

Figure 3.1: RT Neural Network

RTNN is trained by only adjusting weight size factor C and quantizer factors T and U (will be explained in next section) to optimize generalization performance.

3.1.1 Network Description

The network has 4 input nodes for taking four different inputs required for the classification. The parameters taken as inputs are described in section 3.4.1. The first hidden layer has two sets of nodes connected to the Input Layer- simple hidden nodes connected with random weights and quantizers nodes in pairs connected with Z.w and -Z.w weights. Number of output nodes considered is 'm'.

The Real-Time Neural Network (RTNN) is a supervised ANN which can train N distinct samples (x_i, t_i), where $x_i = [x_{i1}, x_{i2}, \ldots, x_{iN}]^T \varepsilon R^n$ are input samples and $t_i = [t_{i1}, t_{i2}, \ldots, t_{im}]^T \varepsilon R^m$ are desired output samples. With L as number of classes or groups to be classified, there are K =(N/L) neurons in the first HL and mL neurons in the second hidden layer. Numbers of quantizers are user defined and depends on number of groups. Each neuron quantizer consists of a A-type neuron and B-type neuron which are two sigmoidal neurons. A-type neuron and B-type neuron of the p^{th} quantizer are denoted as neuron $A^{(p)}$ and neuron $B^{(p)}$. The p^{th} neuron quantizer only links to the p^{th} neuron group, $G^{(p)}$ in the second hidden layer, $p = 1, \ldots, L$. Each group, $G^{(p)}$ consisting of 'm' neurons. Training of the RTNN consists of two

phases: a) determination of weights and bias of standard TLFN and b) determination of weights and biases of neural quantizers.

3.1.2 Determination of Weights and Bias of Standard TLFN

Most important feature of RTNN is choosing weights, w_i and bias, b_i of connection linking input layer to the non-quantizers neurons of first hidden layer randomly. Here, index of weight, w_i and bias, b_i runs from 1 to K, where $K = N/L$. Since all the input samples x_i are distinctive, a vector **w** can be almost randomly chosen such that $w.x_1, w.x_2, \ldots, w.x_N$ are all different. These numbers can be put in following order after re-indexing i:

$$w.x_1 < w.x_2 < \ldots < w.x_N$$

The finite input of N sample vectors, x_i are segregated into 'L' groups consisting of K=N/L input vectors each. These 'L' groups of input vectors are denoted as $V^{(1)}, \ldots, V^{(L)}$. Generalising, these vectors are represented as

$$V^{(p)} = \{x_i \ \ such \ \ that \ \ w.x_{(p-1)N/L+1} < w.x_i < w.x_{pN/L}\}$$

where 'p' ranges from 1 to L.

Basic idea behind the algorithm is to calculate the output in one pass rather than training and computing over large number of iterations. Desired output are user defined. Since the desired output and hidden layer function are known, the algorithm applies *logit* function to calculate the value at the input of the second-hidden-layer neurons. Logit function is the inverse of sigmoid function. The sigmoid function used for hidden layers is given in equation (3.1) which, caters for real values of **x** from -1 to +1 and produces output over the range of 0 to +1.

$$y = f(x) = \frac{1}{(1 + \exp^{(-x)})} \tag{3.1}$$

The logit of a variable 'y' between 0 and 1 is given by equation 3.2

$$logit(y) = \log \frac{y}{1-y} = log(y) - log(1-y) \tag{3.2}$$

Weight size factor 'C', which is connecting second hidden layer to output layer, is chosen any value to make the desired output value lie between -0.5 and 0.5.

$$-0.5 < \frac{t_{ij}}{C} < 0.5 \tag{3.3}$$

where, t_{ij} is desired output. Generalizing from Figure 3.1, p^{th} Neuron Quantizer catering for group 'p' only links to the p^{th} neuron group in the second hidden layer. Network is trained by only adjusting weight size factor 'C' and quantizer factors 'Z' and 'U'. Quantizers inhibit inputs within some intervals by adjusting weights and biases, while allowing inputs belonging to only one class.

3.1.3 Calculation of second hidden layer weight using logit function

(x_k, t_k) is k^{th} Input-Output pair. Calculation of weight of second hidden layer, $\beta^{(p)}$ is of prime importance and it is achieved by equating values, $H^{(p)}$ and $T^{(p)}$, calculated by forward method from the given input x_k and backward method from desired output t_k using logit function, respectively. The weights connecting nodes of first hidden layer to the neurons in the p^{th} group in the second hidden layer is given by equation (3.4).

$$\beta^{(p)} = (H^{(p)})^{-1} T^{(p)} \tag{3.4}$$

where, $\beta^{(p)} = \left[\beta_1^{(p)}, \ldots, \beta_k^{(p)}\right]^T$. $\beta_i^{(p)}$ denotes the weight vector connecting the i^{th} neuron in the FHL of the sub-TLFN to the p^{th} neuron group $G^{(p)}$ in the second layer, $i = 1, \ldots, K$, $K = N/L$ and where,

$$T^{(p)} = T(t_{\frac{(p-1)N}{L}+1}, \ldots, t_{pN/L}) \tag{3.5}$$

$t_{\frac{(p-1)N}{L}+1}$ is the first output element of the network output corresponding to p^{th} group and $t_{pN/L}$ is the last element of the p^{th} group. Since output node is linear combination of product of second hidden layer output and corresponding weight plus bias, values at output node can be expressed in terms of following equation:

$$t_{\frac{(p-1)N}{L}+1} = t'_{(p-1)N/L} C - 0.5C \tag{3.6}$$

where, t' is output of the second hidden layer and it is computed using equation (3.7) which is derived from the above equation.

$$t'_{(p-1)N/L} = 0.5C + t_{\frac{(p-1)N}{L}+1} \tag{3.7}$$

Since, inverse of sigmoid function is $s^{-1}(A) = logA/1 - A$ the input to the second hidden layer is given by $t_{\frac{(p-1)N}{L}+1}$ and represented in equation (3.8) using equation (3.7).

$$T(t_{\frac{(p-1)N}{L}+1}) = \ln\left(\frac{\left(0.5+\frac{t_{\frac{(p-1)N}{L}+1,1}}{c}\right)}{1-\left(0.5+\frac{t_{\frac{(p-1)N}{L}+1,1}}{c}\right)}\right)...\ln\left(\frac{\left(0.5+\frac{t_{\frac{(p-1)N}{L}+1,m}}{c}\right)}{1-\left(0.5+\frac{t_{\frac{(p-1)N}{L}+1,m}}{c}\right)}\right) \tag{3.8}$$

where, 'm' is number of output nodes in output layer. Simplifying equation (3.8) gives us equation (3.9).

$$T(t_{\frac{(p-1)N}{L}+1}) = \ln\left(\frac{\left(0.5+\frac{t_{\frac{(p-1)N}{L}+1,1}}{c}\right)}{\left(0.5-\frac{t_{\frac{(p-1)N}{L}+1,1}}{c}\right)}\right)...\ln\left(\frac{\left(0.5+\frac{t_{\frac{(p-1)N}{L}+1,m}}{c}\right)}{\left(0.5-\frac{t_{\frac{(p-1)N}{L}+1,m}}{c}\right)}\right) \tag{3.9}$$

$T^{(p)}$, input to the second hidden layer is equated as equivalent to product of the output vector of first hidden layer, $H^{(p)}$ and corresponding weight, Beta($\beta^{(p)}$).

$$T^{(p)} = T\left(t_{\frac{(p-1)N}{L}+1},\ldots,t_{\frac{(p)N}{L}}\right)$$

$$= \begin{bmatrix} \ln\left(\frac{\left(0.5+\frac{t_{\frac{(p-1)N}{L}+1,1}}{c}\right)}{\left(0.5-\frac{t_{\frac{(p-1)N}{L}+1,1}}{c}\right)}\right)\ldots\ln\left(\frac{\left(0.5+\frac{t_{\frac{(p-1)N}{L}+1,m}}{c}\right)}{\left(0.5-\frac{t_{\frac{(p-1)N}{L}+1,m}}{c}\right)}\right) \\ \\ \ln\left(\frac{\left(0.5+\frac{t_{\frac{pN}{L},1}}{c}\right)}{\left(0.5-\frac{t_{\frac{pN}{L},1}}{c}\right)}\right)\ldots\ln\left(\frac{\left(0.5+\frac{t_{\frac{pN}{L}+1,m}}{c}\right)}{\left(0.5-\frac{t_{\frac{pN}{L},m}}{c}\right)}\right) \end{bmatrix} \tag{3.10}$$

Now, taking input vector $x^{(p)}$ as input to the network and connecting weight $w^{(p)}$, the output of first hidden layer, $H^{(p)}$ is given by equation (3.11).

$$H^{(p)} = H\left(x_{\frac{(p-1)N}{L}+1},\ldots,x_{\frac{pN}{L}},\quad b_1,\ldots,b_k\right)$$
$$\begin{pmatrix} g\left(w_1x_{\frac{(p-1)N}{L}+1}+b_1\right)\ldots g\left(w_kx_{\frac{(p-1)N}{L}+1}+b_k\right) \\ \ldots \\ g\left(w_1x_{\frac{pN}{L}}+b_1\right)\ldots g\left(w_kx_{\frac{pN}{L}}+b_k\right) \end{pmatrix} \tag{3.11}$$

where, b_i is bias of the corresponding layer.

3.2 Setting of weight and biases of neural quantizers

Set the weights $w_A^{(p)}$ and $w_B^{(p)}$ of the connections linking the input layer to quantizer neurons $Q_A^{(p)}$ and $Q_B^{(p)}$, p=1, . . . , L of FHL as follows:

$$\begin{aligned} w_A^{(p)} &= Z.w \\ w_B^{(p)} &= -Z.w \end{aligned} \tag{3.12}$$

where, parameter Z is called a quantizer factor and should be large enough. In other words, all the weights of connections linking the inputs to all A-type neurons can be chosen as Z.w and all the weights of connections linking the inputs to all B-type neurons chosen as -Z.w. Here, it is pertinent to note that weights are same in magnitude but opposite in polarity. Set the biases of neurons $Q_A^{(p)}$ and $Q_B^{(p)}$ as

$$b_A^{(p)} = -Z.(0.5wx_{\frac{pN}{L}} + 0.5wx_{\frac{pN}{L}+1}) \qquad if \;\; p \neq L$$

$$b_A^{(p)} = -Z.(wx_{pN/L} + max_{j=1..N-1}(wx_{j+1} - wx_j)) \;\; if \;\; p = L \tag{3.13}$$

$$b_B^{(p)} = Z.(0.5wx_{\frac{(p-1)N}{L}} + 0.5wx_{\frac{pN}{L}}) \qquad if \;\; p \neq 1$$

$$b_B^{(p)} = Z.(wx_1 - max_{j=1..N-1}(wx_{j+1} - wx_j)) \qquad if \;\; p = 1 \tag{3.14}$$

Quantizer factor, Z can be chosen large enough such that for any input within input vector group , $V^{(p)}$, only the outputs of both neurons $A^{(p)}$ and $B^{(p)}$ are almost zero while one of the outputs of neurons $A^{(l)}$ and $B^{(l)}$ of each neural quantizer is almost one, where 'p' is not equal to l. Z is given as in equation (3.15).

$$Z = 2\ln\left(\frac{2U}{min_{1,\ldots,N}\ln\left(\frac{\left(c+\frac{\epsilon}{sqrt(m)}-2t_{ij}\right)(c+2_{ij})}{\left(c+\frac{\epsilon}{sqrt(m)}+2t_{ij}\right)(c-2_t ij)}\right)-1}\right) / (min_{j=1,\ldots,N-1}(wx_{j+1} - wx_j)) \tag{3.15}$$

Set the weights W_{AB} of the connections linking neurons $Q_A^{(p)}$ and $Q_B^{(p)}$ to the second hidden layer as $W_{AB} = -U$, where U parameter is called a quantizer factor and is

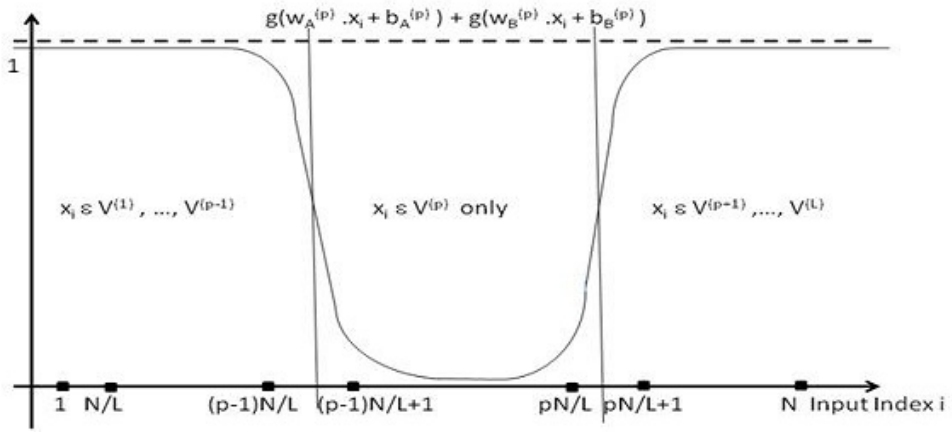

Figure 3.2: Output of pth quantizer of RTNN

given by equation (3.16).

$$U = \ln\left(2\sqrt{m}C\frac{L}{\epsilon} - 1\right) + max_{p=1,\ldots,L,\ q=1,\ldots,N/L,\ s=1,\ldots,L}|H_q^{(p)}\beta^{(s)}| + min_{1,\ldots,N}\ln\left(\frac{\left(c + \frac{\epsilon}{sqrt(m)} - 2t_{ij}\right)(c + 2t_{ij})}{\left(c + \frac{\epsilon}{sqrt(m)} + 2t_{ij}\right)(c - 2t_{ij})}\right) \quad (3.16)$$

U is set to a large enough value such that the input to the p^{th} neuron group $G^{(p)}$ in the second hidden layer from neurons $A^{(p)}$ and $B^{(p)}$ (p^{th} quantizer) has small values e_i for any input x_i within input vector $V^{(p)}$ group. Similarly they have large negative values E_i for any input within other vector groups $V^{(l)}, p = 1, \ldots, L$, and p is not equal to 1. E_i and e_i can be made arbitrarily large and small, respectively, by setting quantizer factors T and Z sufficiently large [29]. E_i goes to negative infinity and e_i goes to zero. The value of epsilon, ϵ is taken as 10^{-6} as the acceptable error for the network.

The weight vectors of the connections linking the input neurons to the Quanizer hidden neurons $A^{(p)}$ and $B^{(p)}$ are chosen as $w_A^{(p)} = Z.w$ and $w_B^{(p)} = -Z.w$, respectively, where Z is a large positive value, p=1, . . . , L. Since $w.x_1 < w.x_2 < \ldots < w.x_N$, regarding each quantizer hidden neuron $A^{(p)}$, $p = 1, \ldots, L$, its bias $b_A^{(p)}$ can be chosen as in equation (3.13) such that $w_A^{(p)}.x_i + b_A^{(p)} < 0$ for $i = 1, \ldots, pN/L$ and $w_A^{(p)}.x_i + b_A^{(p)} > 0$ for $i = pN/L + 1, \ldots, N$. In other words, for all $x_i \in V^{(l)}$, $w_A^{(p)}.x_i + b_A^{(p)} < 0$ if $l \leq p$ and $w_A^{(p)}.x_i + b_A^{(p)} < 0$ if $l > p$.

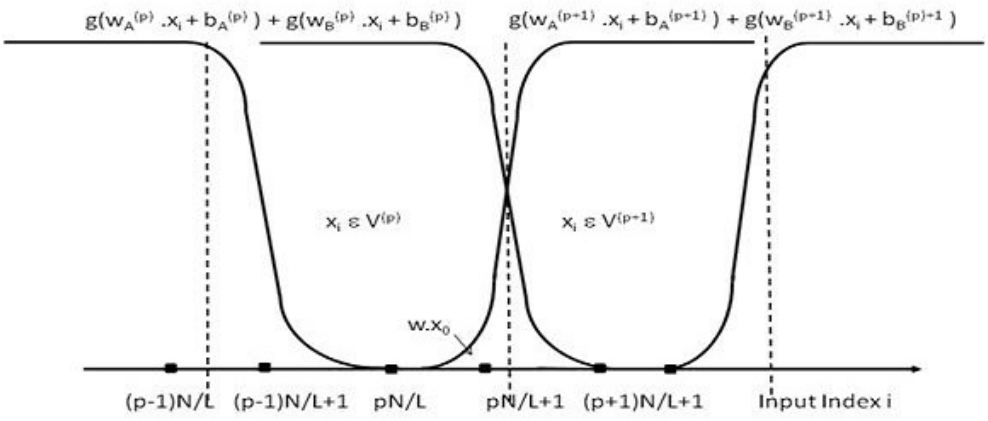

Figure 3.3: Output of Two Contiguous Neural Quantizers

For each quantizer hidden neuron $B^{(p)}, p = 1, ..., L$, its bias $b_B^{(p)}$ can be chosen as in equation (3.14) such that $w_B^{(p)}.x_i + b_B^{(p)} < 0$ for $i = (p-1)N/L + 1, ..., N$ and $w_B^{(p)}.x_i + b_B^{(p)} > 0$ for $i = 1, ..., (p-1)N/L$. Thus, Z can be set large enough so that, for all $x_i \epsilon V^{(p)}$, $p = 1, ..., L$, the outputs of quantizer neurons $A^{(1)}, \ldots, A^{(p)}$ are almost zero and the outputs of quantizer neurons $A^{(p+1)}, \ldots, A^{(L)}$ are almost one. Similarly, the outputs of quantizer neurons $B^{(1)}, \ldots, B^{(p-1)}$ are almost one and the outputs of quantizer neurons $B^{(p)}, \ldots, B^{(L)}$are almost zero as shown in Figure 3.2. Figure 3.3 shows two contiguous neurons and their separation. The size of gap not covered by both contiguous quantizers becomes near zero by adding second term of equation (3.13) and (3.14).

3.3 Model of Real Time Training and Testing

Conventionally, training a trajectory with small data points requires a network configuration with few number of nodes in the network, whereas a trajectory with large number of data points need a network configuration with many nodes in the network. To meet the conflicting requirements of classifying small as well as large trajectories, we are proposing a formulation for partitioning the trajectory by using moving window concept. Moving window concept works on the principle of keeping window size fixed and moving along the trajectory with arrival of each sample as shown in Figure 3.4. In other words, window is moving with time and the training and testing for the

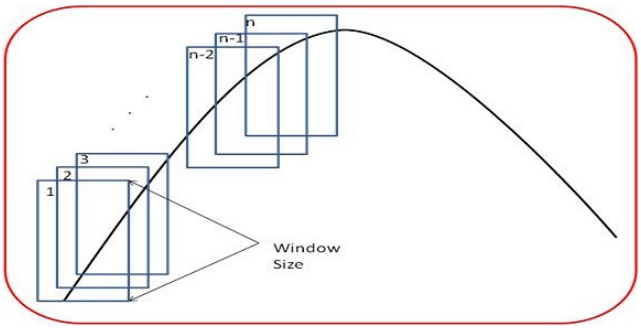

Figure 3.4: Moving Window Concept

class of trajectories are done only on segments of trajectory within the window size. This concept allows us to use parameters in localized frame which helps in reducing the problem to fit into the same network. This moving window concept of testing and training brings out the knowledge embedded in the system and deals with variable length inputs. The length of the moving window is selected such that it fully captures the dynamics of the target which is intended to be classified. Detailed procedure for classifying different classes of ballistic missiles(BM) is given below:

(a) Define a set of classes based on the scenario of threat.

(b) First set of data for a defined window size (we call it a segment) for each class is either taken from a known source or specified using unsupervised ANN like ART-2 (explained in Chapter 5).

(c) RTNN is trained using initial specified samples of each class.

(d) New sample measured by a sensor (like radar) is appended at the end of each segment of defined set of classes.

(e) Oldest sample is deleted from each segment of data temporarily to keep length of the segment matching the window size.

(f) These new segments are tested for closeness towards each class against a defined set of measures.

(g) This new sample is assigned to the matched class and the old sample is deleted whereas unmatched classes of sample are restored to its original values.

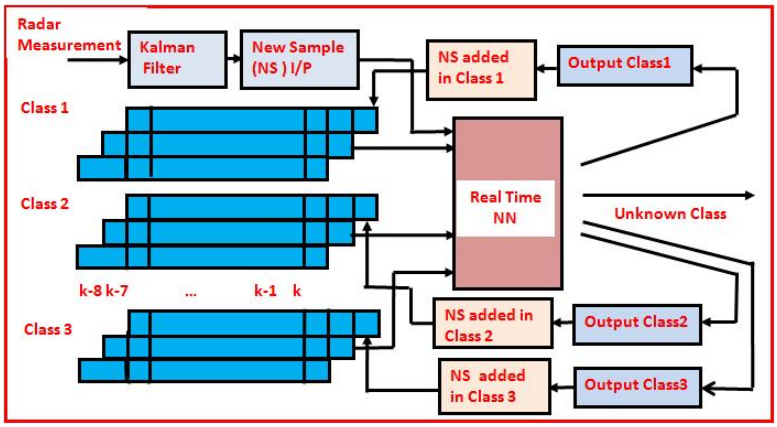

Figure 3.5: Model for Real-Time Processing

(h) Thus, matched segment moves to new location consisting of latest samples of defined window size.

(i) Process is repeated from step (d) to (h) for all test samples.

Figure 3.5 shows the scheme of operations formulated for unified training and testing. The central block is the processing block which is the Real-Time Neural Network (RTNN) shown in figure-3.1, which takes input parameters from any three streams with no specific order between input streams. Class1, Class2 and Class3 are three input streams which are nothing but output taken from kalman filter for four parameters belonging to missiles classes described in section 2.3. The possible corresponding output streams are OutputClass1, OutputClass2 and OutputClass3. There is one additional output which is called ”Unknown”, if output does not match any of three designated outputs within threshold value.

3.4 Implementation and Simulation Results

Two scenarios are considered for experimentation in order to analyse the performance of the RTNN as described in sections 3.4.3 and 3.4.4 to 3.4.7. In the first scenario, we have considered three different aerial objects trajectory, that is, helicopter, aircraft and M400 class of missile without noise. In the second scenario, RTNN is evaluated for classifying ballistic missiles with noise as comparable to real-environment.

Here, we have considered M400, M1000 and M2000 class of missiles corresponding to the range coverage of 400 Km, 1000 Km and 2000 Km respectively. Section 3.4.4 discusses input trajectories with noise and profile of parameters of the missiles used for experimentation. Section 3.4.5 presents results of RTNN for trajectory with noise whereas section 3.4.6 describes results of input trajectories with loss. Section 3.4.7 covers trajectories with deviation which mimics manoeuvring ballistic missiles. First, we shall deliberate on the parameters considered for input to the network.

3.4.1 Input parameters to the network

Radar measures the target positions in spherical coordinates, range (r), azimuth (ψ) and elevation (θ) angle, centered at its mounting location. Position (X, Y, Z), velocity (V_x, V_y, V_z) and acceleration (a_x, a_y, a_z) are estimated by using the radar measured data after passing through kalman filter as described in Section 2.3. In our study, we have selected following parameters by taking into account the accuracy, separation distance with other classes and their observability:

- **Specific Energy (SE):** Specific energy is energy per unit mass. Specific energy is computed using $0.5(v^2 + gh)$, where 'v' is velocity, 'h' is absolute height and 'g' is gravity. The SI unit for specific energy is the joule per kilogram (J/kg).

- **Acceleration:** Acceleration is the rate of change of velocity with time. Acceleration occurs anytime an object's speed increases, decreases, or changes direction.

  $$a = \triangle v / \triangle t = (v - v_0) \triangle t$$

  The SI unit for acceleration is m/s^2 (meters per second squared).

- **Height:** Height or altitude is taken as third parameter for input. It is vertical distance of the object from mean sea level. Unit for measurement is Kilometer (Km).

- **Velocity:** Linear velocity is fourth parameter to be fed to the network for classification. Velocity is a vector quantity that denotes the rate of change of position with respect to time.

  $$v = \triangle s / \triangle t = (s - s0) / \triangle t$$

The scalar (absolute value) magnitude of the velocity vector is the speed of the motion. The SI unit for velocity is m/s (meters per second).

3.4.2 Unified training and testing steps for RTNN

The steps involved in unified training and testing can be summarized as follows:

(a) 8 samples for each class of pre-specified tracks are passed through the network along with the desired output of $(1, -1)$, $(-1, 1)$ and $(-1, -1)$ corresponding to three input classes of M400, M1000 and M2000.

(b) Weight of connections between second layer and output layer, C is set as per equation (3.3), $-0.5 < \frac{t_{ij}}{C} < 0.5$, where t_{ij} is desired output vector.

(c) Input to second hidden layer, T is calculated using logit function, $s^{-1}(A) = ln(A/(1-A))$ as per equation (3.10).

(d) Output of first hidden layer, $H^{(p)}$ is computed using function $g(wx + b)$, where x is input, w is weight and b is bias of the corresponding node using equation (3.11).

(e) Second hidden layer weight, $\beta^{(p)}$ is found by using equation (3.4).

(f) Quantizer factors Z and U are calculated using equation (3.15) and (3.16) respectively to be used for calculating bias and weight of quantizer neurons.

(g) Test vector is passed through the network.

- If output matches to any specified class, then first data out of 8 samples is deleted from the set and new sample is added in the end. In this way, a window of length 8 is maintained for entire duration of classification process. This unique process of integrated unified training and testing is used for real-time classification of target trajectory.
- If the output does not match any specified class, then it is declared as unknown and no action is taken i.e. old sets of data is used for the next iteration.

(h) Steps (a) to (g) are repeated for all trajectory data sets.

3.4.3 M400 class of Missile, Helicopter, Aircraft (F-16) without noise

The attributes that define the trajectory are position, velocity and acceleration. These attributes along with GPS time stamp are measured by phased array radars which have capability to track targets even with RCS(Radar Cross Section) less than 0.1 square meters at a range beyond 500 KM. The magnitude of these target kinematic attributes

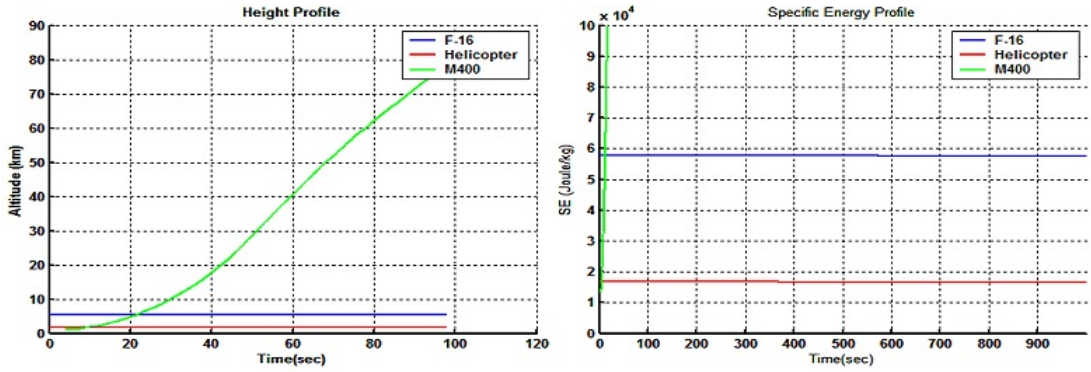

Figure 3.6: a) Height profile b) SE profile of Helicopter and Aircraft (F-16)

is highly dependent on target type like F-16 fighter aircraft, helicopter, missile, etc as shown in Figure 3.6(a). Figure 3.6(b) shows the specific energy profile for F16 and Helicopter. Similarly, Figure 3.7(a) and Figure 3.7(b) show the corresponding height and velocity profile for F16 and helicopter. The F-16 fighter aircraft can travel with velocity and altitude of 100 m/s and 5.4 Km respectively. Similarly, helicopter moves with velocity and altitude of 50 m/s and 1.6 Km respectively. The specific energy for F-16 and helicopter is in the range of 58000 Joules/Kg and 17000 Joules/Kg respectively as indicated in Figure 3.6(b).

The input to the network is coming from three targets as given by simulator at regular interval of 100 msec. For showing the results, we have taken 100 samples from each target belonging to the track of helicopter, aircraft and M400 class of missile. The fourth combination of desired values of (+1,+1) is not allocated to any objects as we have considered only for three targets for this experiment. The desired output for these vehicles is set as in Table 3.1.

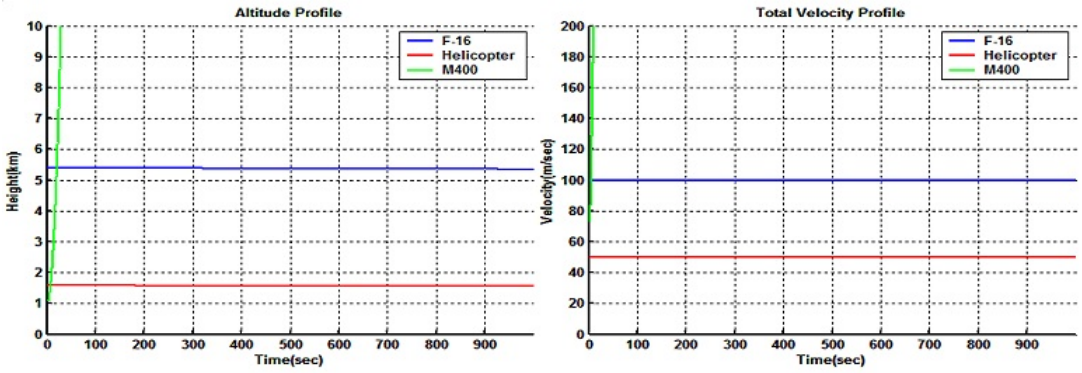

Figure 3.7: a) Altitude b) Velocity profile of Helicopter and Aircraft (F-16)

Table 3.1: Desired output of the Network

Sl No	Class of Target	Desired Output for 2nd Output Node	Desired Output for 1st Output Node
1	Helicopter	-1	-1
2	Aircraft	-1	+1
3	Missile	+1	-1

The outputs of first and second output nodes are segregated and plotted as consecutive samples and shown in Figure 3.8(a) and Figure 3.8(b). It is clearly visible that segregated first 100 samples have output have output values corresponding to node1 and node2 closure to -1 and -1 respectively corresponding to helicopter. The mean and variance of the output is -0.945 and 0.0148 as shown in first row of Table 3.2. Similarly, output for aircraft is closure to desired values of -1 and +1 as is visible from computed mean in second row Table 3.2. The computed output for aircraft is 0.9945 with variance of 0.0633. Similarly as noted in third row of Table 3.2, the computed mean of output for M400 class of missile is 1.0073 with variance of 0.0050.

The pass percentage of classification for test scenario with helicopter, aircraft and M400 missile is given in Table 3.3. Since Missile attributes are quite different than that of helicopter and aircraft, the pass percentage for missile is 100% whereas for aircraft and helicopter, the correct classification is 98% and 94% respectively.

Table 3.4 shows track rate update from radar environemnt, training time and testing time per sample for the scenario in which M400 class of missile, helicopter and aircraft

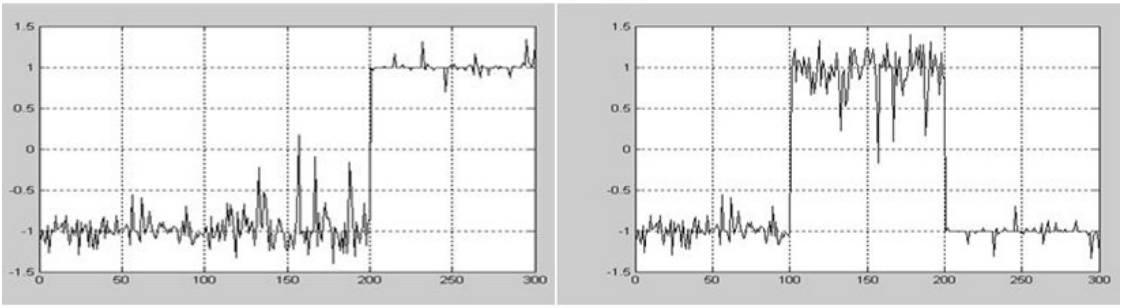

Figure 3.8: a) Second node b) First Node of Output layer

Table 3.2: Computed output of the Network

Sl No	Class of Target	Desired Node1	Desired Node2	Computed Output Node1	Computed Output Node2	Computed Variance
1	Helicopter	-1	-1	-0.9945	-0.9945	0.0148
2	Aircraft	-1	+1	-0.9945	+0.9945	0.0633
3	Missile	+1	-1	+1.0073	-1.0073	0.0050

trajectories are exercised for classification.

3.4.4 M400, M1000, M2000 Input Trajectories

Another scenario with three different classes of missile with similar characteristics during initial phase of flight has been experimented. Figure 3.9 and Figure 3.10 show acceleration, velocity, altitude and specific energy profile of three classes of missiles of range 400 Km, 1000 Km and 2000 Km during complete duration of flight. The trajectory parameters of SE, acceleration, altitude and velocity are extracted using Kalman filter as described in section 2.3 and are subjected to sequence of processing as shown

Table 3.3: Pass percentage of the Network

Sl No	Class of Target	Classified Percentage	Unclassified percentage
1	Helicopter	94.0	6.0
2	Aircraft	98.0	2.0
3	Missile	100	0.0

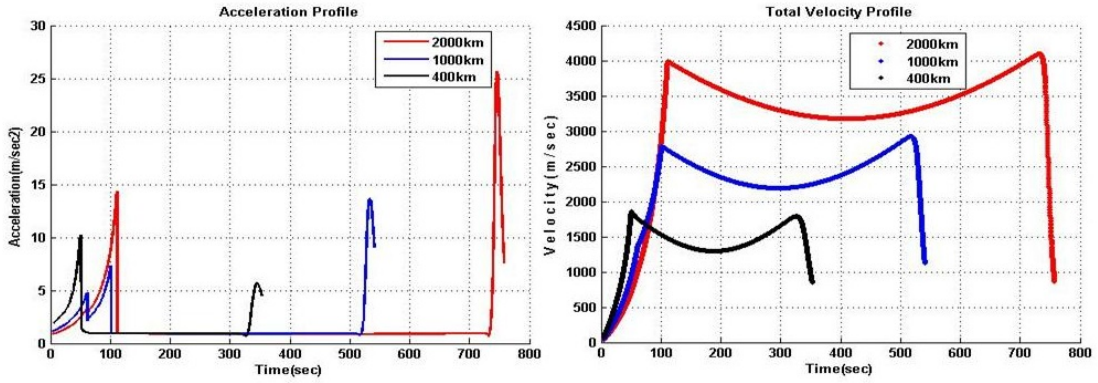

Figure 3.9: a) Acceleration b) Velocity profile of M400, M1000 and M2000 Class of Missile

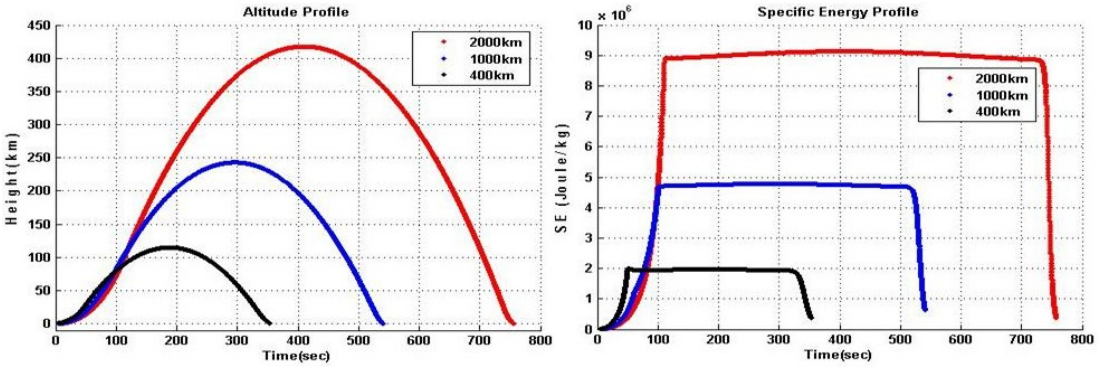

Figure 3.10: a) Altitude b) SE profile of M400, M1000 and M2000 Class of Missile

in Figure 3.13 for determining the class of samples received either from radar or 6-DOF simulation test bed as described in section 2.6.

The parameters taken for training and testing are from 15 sec from the launch of the missile. Duration of testing and training is for 30 seconds consisting of 300 samples of trajectories of three different types of missiles. The missile parameter in the zone of interest corresponding to SE, Acceleration, Altitude and Velocity for first 100 sec is shown from Figure 3.11 and Figure 3.12. Samples beyond these points will have much larger segregation which is easier to classify.

The proposed model has three input classes which extracts parameters and feeds input to real-time neural network(RTNN). The network produces output closure to the trained class. The complete procedure of operation for real-time classification of

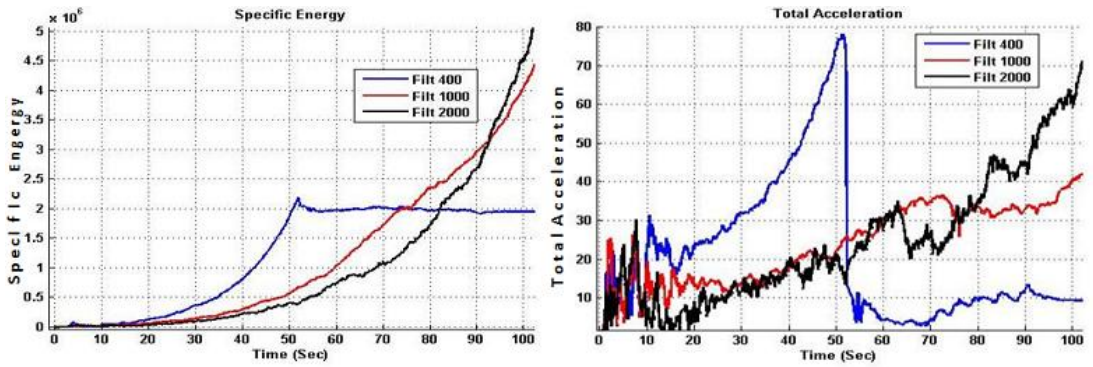

Figure 3.11: a) SE b) Acceleration from Launch to 100 sec

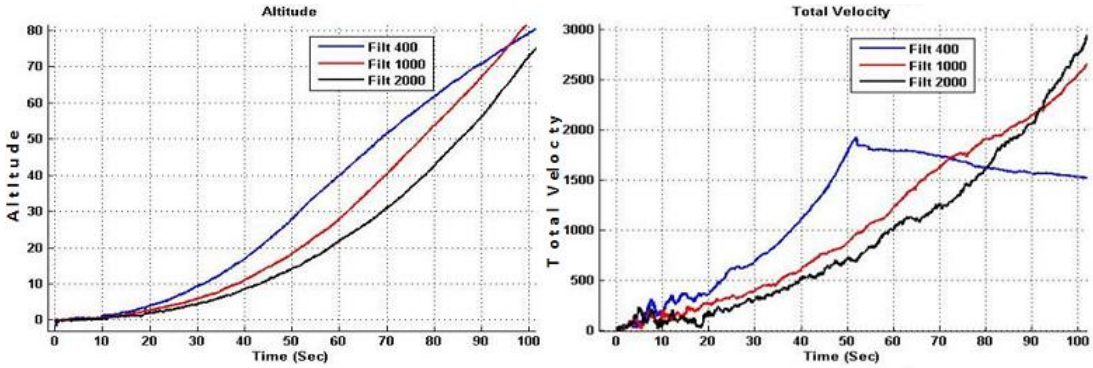

Figure 3.12: a) Altitude b) Velocity from launch to 100 sec

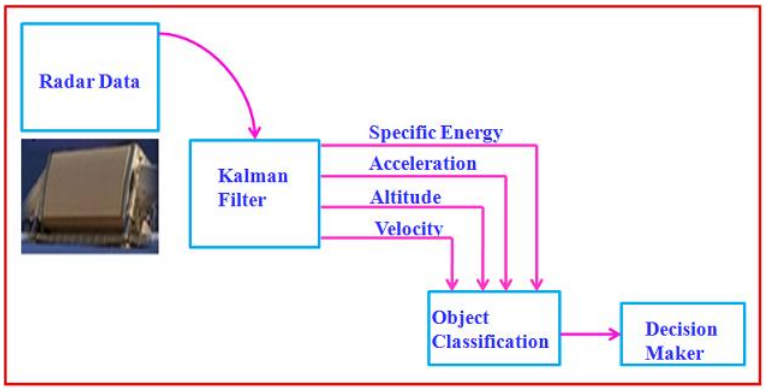

Figure 3.13: Processing Sequence for RTNN Trajectory Classification

Table 3.4: Computation Timing of RT algorithm

Sl No	Description	Time
1	Track Rate Update	100 msec
2	Training Time	75 msec
3	Testing Time for One New Sample	0.69 msec

Table 3.5: Platform Specification of Experimentation

Sl No	Description	Uses
1	Machine Type	Pentium-4
2	Clock Speed	3 GHz
3	Main Memory	512 MB
4	Operating System	Windows XP
5	Inverse Algorithm Used	SVD (Singular Value Decomposition

M400, M1000 and M2000 class of missiles is as given in section 3.4.2.

3.4.5 Missiles classification with noise

The trajectory data from 6 DOF (degree of freedom) is converted into spherical coordinates centered at radar and corrupted with noise of 20 m in range and 2 milli radians in both azimuth and elevation angle. This noisy data is processed using an extended Kalman filter before passing RTNN for classification. The input to the network is coming from three targets as given by target simulator at regular interval of 100 msec. For comparing the results, we have taken 300 samples from each target class. The algo-

Table 3.6: Mean and Variance of Results for Run1

Sl No	Class of Targets	Expected Values of Node1	Expected Values of Node2	Computed Mean Node1	Computed Mean of Node2	Computed Variance
1	M2000	-1.0	-1.0	-0.9729	-0.9729	0.05490
2	M1000	-1.0	1.0	-0.7870	0.7870	0.40060
3	M400	1.0	-1.0	0.9507	-0.9507	0.09264

Table 3.7: Mean and Variance of Results for Run2

Sl No	Class of Targets	Expected Values of Node1	Expected Values of Node2	Computed Mean Node1	Computed Mean of Node2	Computed Variance
1	M2000	-1.0	-1.0	0.9518	0.9518	0.07869
2	M1000	-1.0	1.0	0.8212	0.8212	0.3727
3	M400	1.0	-1.0	0.8860	0.8860	0.2101

Table 3.8: Mean and Variance of Results for Run3

Sl No	Class of Targets	Expected Values of Node1	Expected Values of Node2	Computed Mean Node1	Computed Mean of Node2	Computed Variance
1	M2000	-1.0	-1.0	-0.9959	-0.9959	0.0278
2	M1000	-1.0	1.0	-0.9794	0.9794	0.0295
3	M400	1.0	-1.0	0.9036	-0.9036	0.1871

rithm is realised in Microsoft Visual C++ with MS Windows XP running on Pentium-4 CPU with 3 GHz clock speed and 512 MB RAM as shown in Table 3.5.

Tables 3.6, 3.7 and 3.8 show desired values and results of three different runs-Run1, Run2 and Run3 for the trajectories mentioned in the above-paragraph after passing through RTNN. Third and fourth column of tables 3.6 to 3.8 show desired values of the network. Mean values of computed output corresponding to three different classes of missiles are placed in column-6 and column-7. Variance is presented in the last column of tables 3.6 to 3.8. The threshold value of pass criteria can be adjusted to constrain or expand the acceptable value. The threshold range for classified output is kept as $+/-0.5$ from desired value. If desired output is $(-1, 1)$, then acceptable ranges are from -0.5 to -1.5 for node1 and 0.5 to 1.5 for node2. As statistics of classification for three different runs given in table 3.6, table 3.7 and table 3.8 show, computed mean for M2000 class of missile varies from 0.9518 to 0.9959 for desired value of 1. The variance for three runs for M2000 varies from 0.0278 to 0.0549. Similarly, computed mean for M1000 class of missile ranges from 0.7870 to 0.9794 for corresponding desired value of 1. The variance for these runs varies from 0.0295 to 0.40060. For M400 class of missile, the mean variation is from 0.8860 to 0.9507 against desired value of 1

and variance varies from 0.09264 to 0.2101. It is evident from the graphs (Figure 3.14,

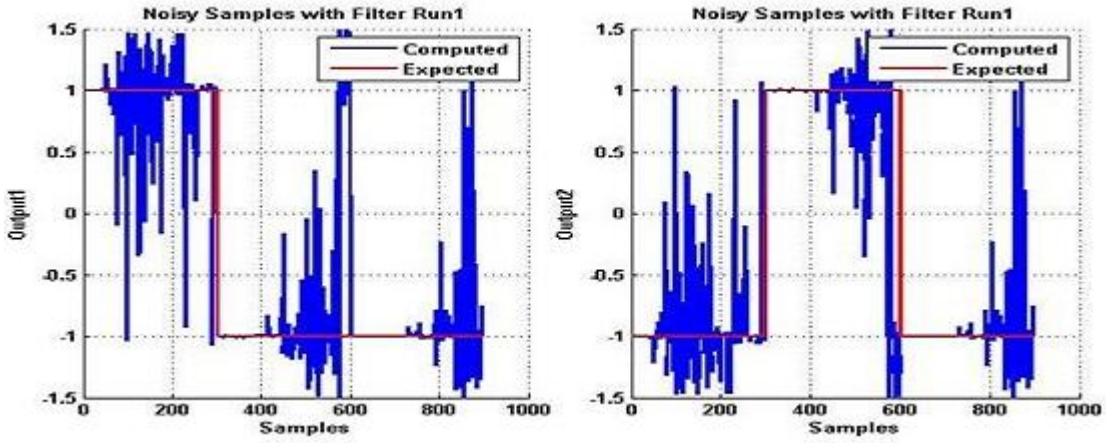

Figure 3.14: RTNN Results showing Desired and Computed Values for Run1

Figure 3.15 and Figure 3.16) that the network is able to retain its desired output during full testing duration.

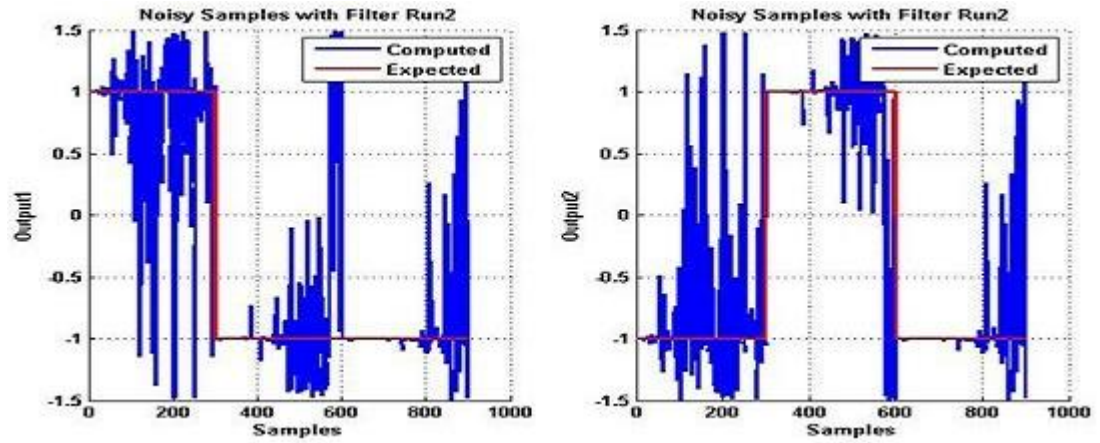

Figure 3.15: RTNN Results showing Desired and Computed Values for Run2

Confusion matrix presents its diagonal elements as pass counts and off-diagonal elements as fail counts. The confusion matrix presented here also includes in last two columns unclassified count of samples and its percentage. Confusion matrix of Table 3.9, Table 3.10 and Table 3.11 show result summary of three runs. Both number of passes and pass percentage are shown in the table. Mean pass percentage for these

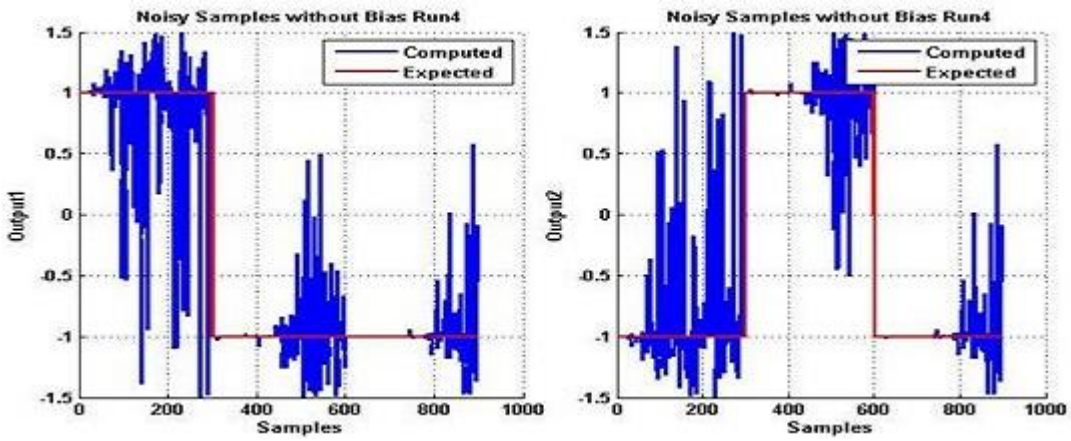

Figure 3.16: RTNN Results showing Desired and Computed Values for Run3

runs are more than 93%. Major advantage of RTNN is its ability to classify some of the noisy data as unknown instead of misclassifying. The unclassified percentage of data in three different runs ranges from 2% to 6%. Mean misclassification is recorded as 2.66% with best value as 0% and worst value as 8.66%.

3.4.6 Results with Sample Losses

Real-life scenario may lead to temporary track loss by sensor(radar). To evaluate this condition, the network is also subjected to trajectories with loss of samples during classification to evaluate the methodologies for its robustness. Figure 3.17 shows original trajectory of M400 and trajectory with loss of samples. Figure 3.18 to Figure 3.19 show input samples of SE, acceleration, altitude and velocity of M400, M1000 and M2000 class of missiles trajectories with M400 class of missile being subjected to sample-losses for 2 seconds of 20 samples.

The results of these trajectories are presented in Figure 3.20 and Figure 3.21. The confusion matrix in Table 3.12 and Table 3.13 show pass number, misclassified numbers and unclassified numbers. These tables also bring out pass percentage, misclassified percentage and unclassified percentage for each trajectory. Diagonal cells represent pass percentage whereas non-diagonal cells represent misclassified values and percentage. Last two columns define unclassified values and unclassified percentage. The confusion matrix clearly brings out performance of the RTNN network by enlist-

Table 3.9: Confusion Matrix for RTNN Run1

		Actual Class							
		Class 1		Class 2		Class 3		Unclassified	
		Pass/ Fail Count	Percentage	Pass/ Fail Count	Percentage	Pass/ Fail Count	Percentage	Number	Percentage
Predicted Count	Class M1000	265 / 300	88.3%	3	1.0%	0	0	6	2.0%
	Class M2000	0	0	292 / 300	95.0%	0	0	8	2.6%
	Class M400	26	8.6%	0	0	285 / 300	97.3%	15	5.0%

Table 3.10: Confusion Matrix for RTNN Run2

		Actual Class							
		Class 1		Class 2		Class 3		Unclassified	
		Pass/ Fail Count	Percentage	Pass/ Fail Count	Percentage	Pass/ Fail Count	Percentage	Number	Percentage
Predicted Count	Class M1000	269/ 300	89.66%	10	3.33%	0	0	8	2.66%
	Class M2000	0	0	287/ 300	95.66%	0	0	13	4.33%
	Class M400	23	7.66%	0	0	283/ 300	94.33%	17	5.66%

ing pass and failed count. In Table 3.12 for run-1, M1000 class shows 96.33 as pass percentage, M2000 as 97.66% and M400 as 93.21%. The misclassified percentage is 4.33% for M1000 and 0.66% for M400 class of missiles. M2000 has no misclassification. The unclassified percentage, in last two columns, varies from 2.14% to 3.33%. In Table 3.13, for another instance of simulation runs, M1000 class shows 92.33 as pass percentage , M2000 as 99% and M400 as 93.92%. The misclassified percentage is 2% for M1000 and 5% for M400 class of missiles. M2000 has no misclassification. The

Table 3.11: Confusion Matrix for RTNN Run3

		Actual Class							
		Class 1		Class 2		Class 3		Unclassified	
		Pass/ Fail Count	Percentage	Pass/ Fail Count	Percentage	Pass/ Fail Count	Percentage	Number	Percentage
Predicted Count	Class M1000	289/ 300	96.33%	10	3.33%	0	0	11	3.66%
	Class M2000	0	0%	294/ 300	98.00%	0	0	6	2.00%
	Class M400	0	0	0	0	275/ 300	91.66%	15	5.00%

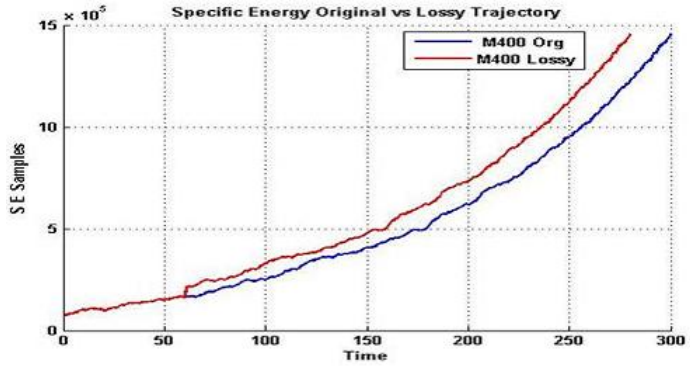

Figure 3.17: Specific Energy comparison of Lossy Trajectory with Original for M400

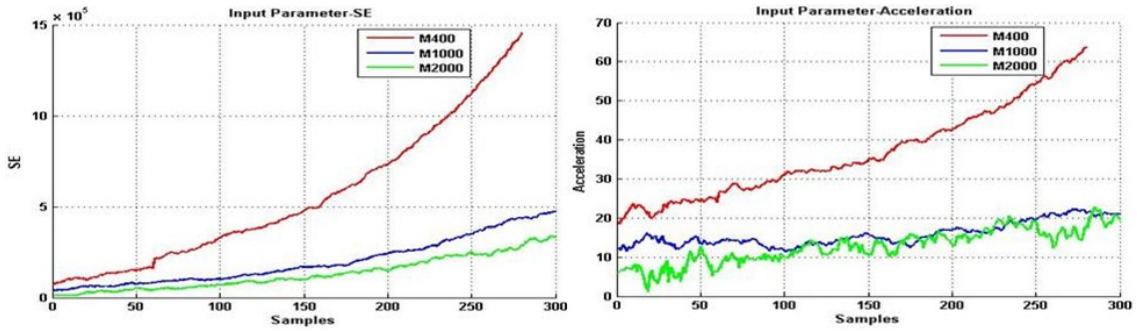

Figure 3.18: a) Specific Energy b) Acceleration Profile with Loss of 20 Samples

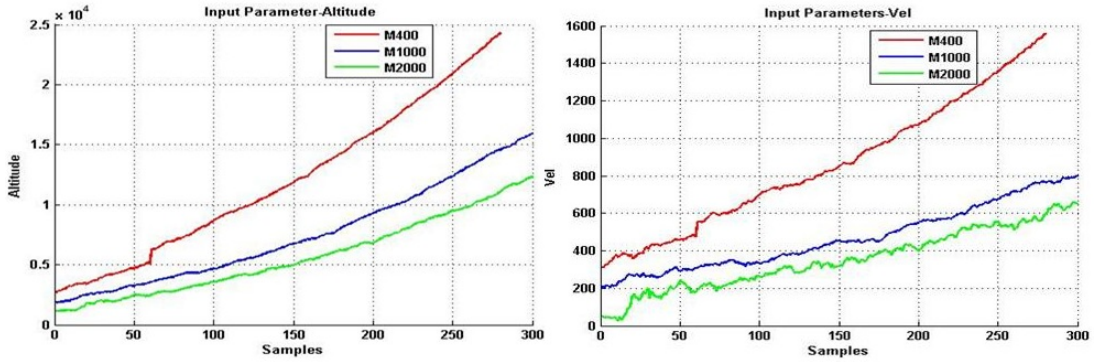

Figure 3.19: a) Altitude b) Velocity Profile with Losses for 20 Samples for M400 Class

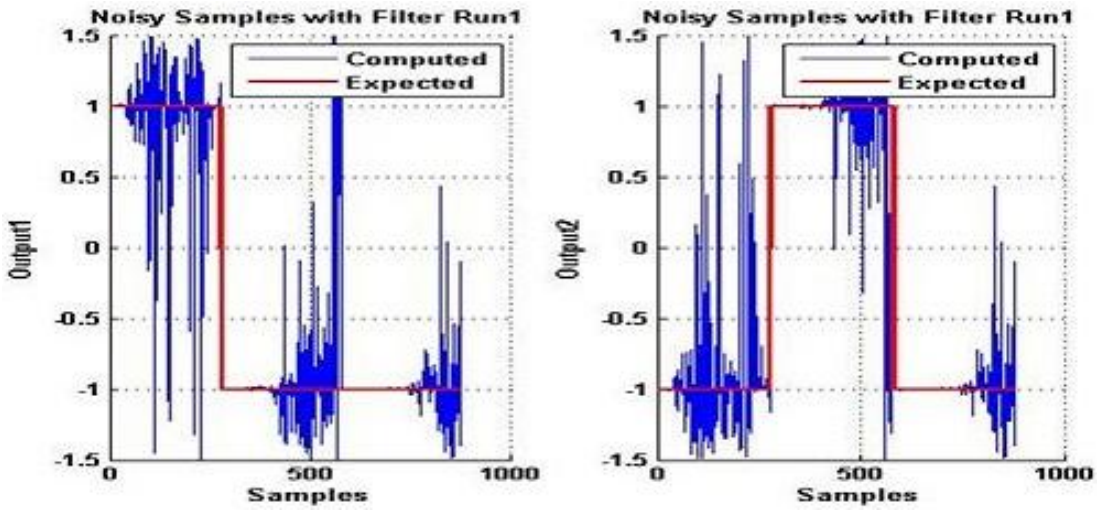

Figure 3.20: Output of RTNN with Losses for 20 Samples for M400 Class Run1

Unclassified percentage, in last two columns, varies from 1% to 3.92%.

3.4.7 Results with Deviations in Trajectory

The algorithm is also subjected to trajectories with deviation in path of flight which is equivalent to manoeuvering targets. Figure 3.22 and Figure 3.23 show input samples of SE, Acceleration, altitude and velocity of M400, M1000 and M2000 class of missiles trajectories with M400 class of missile incorporating deviation or manoeuvering from samples 100 to 170. The results of classification with these trajectories as input to

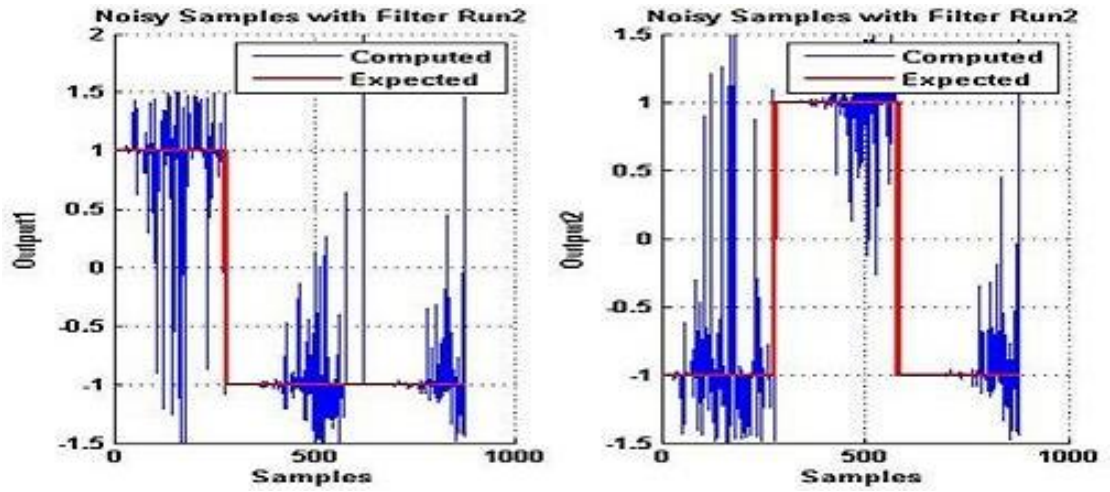

Figure 3.21: Output of RTNN with Losses for 20 Samples for M400 Class Run2

Table 3.12: Matrix with loss of 20 Samples of M400 Class of Missiles

		Actual Class							
		Class 1		Class 2		Class 3		Unclassified	
		Pass/ Fail Count	Percentage	Pass/ Fail Count	Percentage	Pass/ Fail Count	Percentage	Number	Percentage
Predicted Count	Class M1000	289/ 300	96.33%	0	0	13	4.33%	10	3.33%
	Class M2000	0	0%	293/ 300	97.66%	0	0	7	2.33%
	Class M400	1	0.66%	0	0	261/ 280	93.21%	6	2.14%

RTNN are presented in Figure 3.24, Figure 3.25 and Figure 3.26. The confusion matrix in Table 3.17, Table 3.18 and Table 3.19 show pass number, misclassified numbers and unclassified numbers. These tables also bring out pass percentage, misclassified percentage and unclassified percentage for each trajectory. In Figure 3.24 to Figure 3.26, red line defines expected results, whereas, blue line shows computed results from RTNN. Most of the computed values match with expected results. Table 3.14 to Table 3.16 show expected results of $(-1,-1)$, $(-1,1)$ and $(1,-1)$ and computed mean and

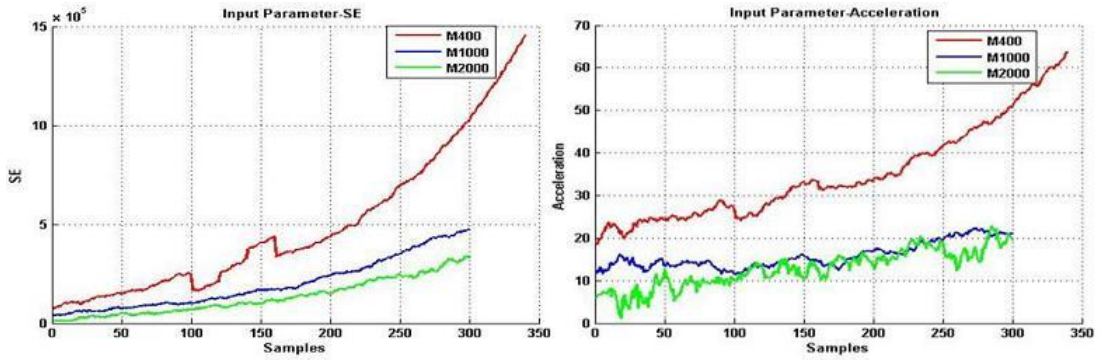

Figure 3.22: a) SE b) Acceleration Input Samples with Deviation in M400 Trajectory

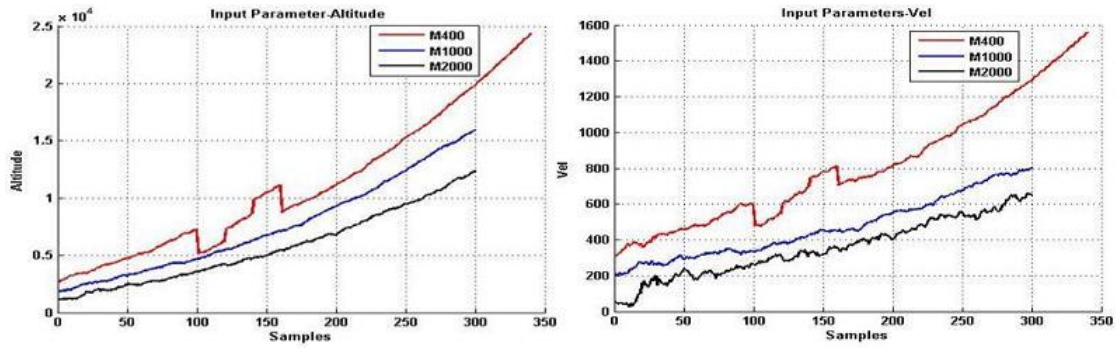

Figure 3.23: a) Altitude b) Velocity Input Samples with Deviation in M400 Trajectory

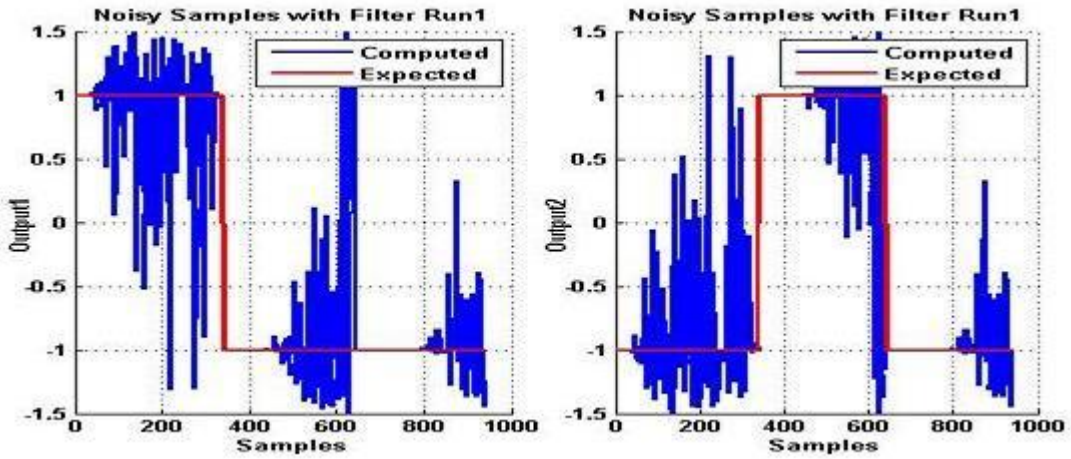

Figure 3.24: Results of Deviated Trajectory Run1

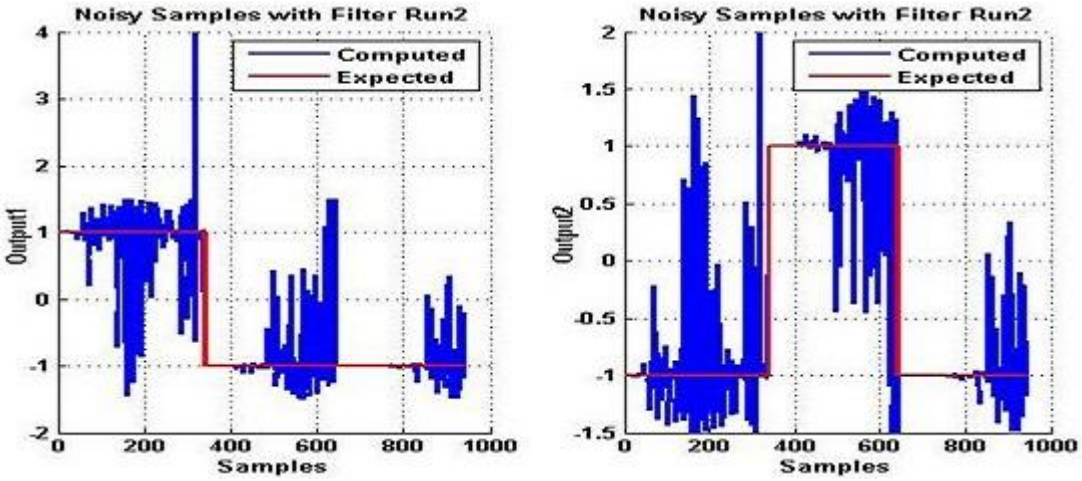

Figure 3.25: Results of Deviated Trajectory Run2

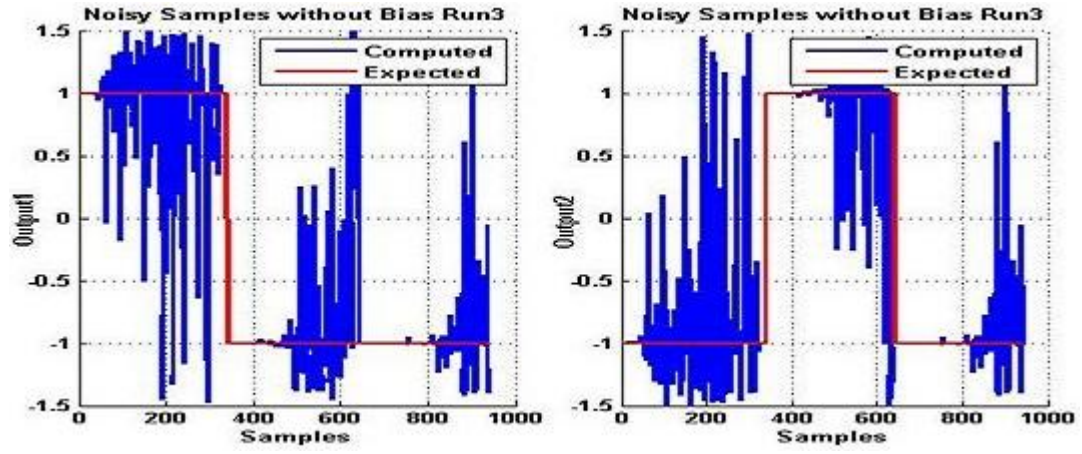

Figure 3.26: Results of Deviated trajectory Run3

Table 3.13: Matrix with loss of 20 Samples of M400 Class of Missiles

Run 2		Actual Class							
		Class 1		Class 2		Class 3		Unclassified	
		Pass/ Fail Count	Percentage	Pass/ Fail Count	Percentage	Pass/ Fail Count	Percentage	Number	Percentage
Predicted Count	Class M1000	277/ 300	92.33%	0	0	6	2.00%	09	3.00%
	Class M2000	0	0%	297/ 300	99.00%	0	0	3	1.00%
	Class M400	14	5.00%	0	0	263/ 280	93.92%	11	3.92%

Table 3.14: Mean and Variance of Results of Run1

SlNo	Class of Target	Expected Values of Node1	Expected Values of Node2	Computed Mean of Node1	Computed of Node2	Computed Variance
1	Bm2000	-1.0	-1.0	-0.9936	-0.9936	0.0185
2	Bm1000	-1.0	1.0	-0.8382	-0.8382	0.3231
3	Bm400	1.0	-1.0	0.9453	- 0.9453	0.1130

variance for M2000, M1000 and M400 Class of missiles for three runs. Column 3 and column 4 show mean of node 1 and node2, whereas column 5 shows variance from expected values. It can be very clearly seen that mean of computed values are near expected values. From Table 3.14 to Table 3.16, over three runs, mean for M2000 varies from -0.9626 to -0.9936 against expected value of -1.0; whereas mean of M1000 varies from -0.8382 to -0.8690 against expected value of -1.0; whereas mean of M400 varies from 0.9285 to 0.9453 against expected value of 1.0. The variance observed for M2000 over three runs varies from 0.0185 to 0.0551; for M1000 it varies from 0.2689 to 0.3231 and for M400, variance observed are 0.11300 to 0.1687.

In Table 3.18, M1000 class shows 91.00 as pass percentage, M2000 as 96.00% and M400 as 92.06%. The misclassified percentage is 2.66% for M1000 and 4.70% for M400 class of missiles. M2000 has no misclassification. The Unclassified percentage,

Table 3.15: Mean and Variance of Results of Run2

SlNo	Class of Target	Expected Values of Node1	Expected Values of Node2	Coputed Mean of Node1	Computed Mean of Node2	Computed Variance
1	Bm2000	-1.0	-1.0	-0.9711	-0.9711	0.0413
2	Bm1000	-1.0	1.0	-0.8690	0.8690	0.2689
3	Bm400	1.0	-1.0	0.9285	- 0.9285	0.1646

Table 3.16: Mean and Variance of Results of Run3

SlNo	Class of Target	Expected Values of Node1	Expected Values of Node2	Computed Mean of Node1	Computed of Node2	Computed Variance
1	Bm2000	-1.0	-1.0	-0.9626	-0.9626	0.0551
2	Bm1000	-1.0	1.0	-0.8390	0.8390	0.3074
3	Bm400	1.0	-1.0	0.9243	- 0.9243	0.1687

Table 3.17: Confusion Matrix for Results of Run1

		Actual Class							
		Class 1		Class 2		Class 3		Unclassified	
		Pass/ Fail Count	Percentage	Pass/ Fail Count	Percentage	Pass/ Fail Count	Percentage	Number	Percentage
Predicted Count	Class M1000	269/ 300	89.66%	0	3.33%	5	0	11	3.66%
	Class M2000	0	0	296/ 300	98.66%	0	0	4	1.33%
	Class M400	20	5.88%	0	0	317/ 340	93.23%	18	5.29%

in last two columns, varies from 3.66% to 5.58%. In Table 3.19, M1000 class shows 89.66 as pass percentage, M2000 as 96.00% and M400 as 91.47%. The misclassified percentage is 2.33% for M1000 and 5.58% for M400 class of missiles. M2000 has no

Table 3.18: Confusion Matrix for Results of Run2

		Actual Class							
		Class 1		Class 2		Class 3		Unclassified	
		Pass/ Fail Count	Percentage	Pass/ Fail Count	Percentage	Pass/ Fail Count	Percentage	Number	Percentage
Predicted class	Class M1000	273/ 300	91.00%	0	0	8	2.66%	11	3.66%
	Class M2000	0	0	288/ 300	96.00%	0	0	12	4.00%
	Class M400	16	4.70%	0	0	313/ 340	92.06%	19	5.58%

misclassification. The Unclassified percentage, in last two columns, varies from 3.00% to 6.17%. Thus, it is clearly demonstrated that RTNN performance is excellent even for manoeuvring trajectories where pass percentage is above 92 %.

3.5 Summary of RTNN

In this chapter, we have presented a framework for classification of ballistic missiles trajectory using RTNN in real-time. Model developed allows RTNN to integrate training and testing while taking care of short as well as long range trajectories to be addressed by the same model. Further, it is established that the RTNN is able to classify the new sample in less than 1 milli-seconds which makes it useful for real-time applications. The model overall performance is around 93% and time taken by RTNN is near 1 ms on Pentium-4 processor. newpage

Analysis of experiment conducted on trajectory with noise, trajectories with lossy samples and deviated trajectories shows that RTNN combined with unified model for training and testing is proving excellent for real-time applications. The results of these experiments are recorded in Table 3.20 for two runs. The pass percentage for two runs for M1000 is 88.33% and 89.63%, for M2000 is 97.3% and 94.3% and for M400 is 95.0% and 95.6% for noisy data. The misclassified percentage for these three classes for two runs is 8.6% and 7.6%, 0.00% and 0.00% , 1.0% and 3.3% respectively. In

Table 3.19: Confusion Matrix for Results of Run3

		Actual Class							
		Class 1		Class 2		Class 3		Unclassified	
		Pass/ Fail Count	Percentage	Pass/ Fail Count	Percentage	Pass/ Fail Count	Percentage	Number	Percentage
Predicted class	Class M1000	269/ 300	89.66%	0	0	8	2.33%	12	4.00%
	Class M2000	0	0	288/ 300	96.00%	0	0	9	3.00%
	Class M400	19	5.58%	0	0	311/ 340	91.47%	21	6.17%

other words, M2000 has no misclassification. The unclassified percentage, for run1 varies from 2.00% to 5.00% in 2^{nd} to 5^{th} column of Table 3.20 and 2.6% to 5.6% for run2. For trajectory with losses and deviated trajectories also, pass percentage varies from 91.66% as lowest to 98.6% as highest for these two runs. Fail percentage for same runs recorded 0% to 5.33% whereas unclassified percentage varied from 1% to maximum of 5.88%.

Table 3.20: Summary of RTNN Results

RTNN	**Noisy Data**			**Lossy Trajectory**			**Deviated Trajectory**		
Rl	M1000	M2000	M400	M1000	M2000	M400	M1000	M2000	M400
Pass %age	88.33	97.3	95.0	96.3	97.6	93.2	89.6	98.6	93.2
Failed %age	8.66	0.0	1.0	0.66	0.0	4.33	5.8	0.0	1.6
Unclassifed	2.00	2.66	5.0	3.33	2.3	2.14	3.66	1.33	5.29
R2									
Pass %age	89.63	94.3	95.6	92.33	99.00	93.9	96.33	91.66	97.33
Failed %age	7.6	0.0	3.3	2.00	0.00	0.0	5.33	0.00	2.66
Unclassifed	2.6	4.3	5.6	3.00	1.00	3.9	3.66	4.00	5.88

Experiments were also conducted to distinguish between aircraft, helicopter and M400 class of missile and the pass percentage acheived is good due to well separated energy levels.

Chapter 4

Hidden Markov model

In the previous chapter we outlined a framework for classification of ballistic missile trajectory using Real time neural network(RTNN). In this chapter we address the classification of ballistic missile trajectory using hidden Markov Models (HMM).

Real-world processes generally produce observable outputs which can be characterised as signals [49]. Models can be developed to characterise these signals. Study about these models can be divided into two classes: deterministic model and statistical models. Gaussian processes, Poisson processes, Markov Processes and Hidden Markov processes form examples of Statistical model. Statistical model has underlying assumption that it follows well-defined distribution and it can be estimated in well-defined manner. Hidden Markov Model (HMM) is characterised as parametric random process which has been utilised in this thesis for trajectory classification. The HMM is widely used in speech recognition, natural language modelling, on-line handwriting recognition, and for the analysis of biological sequences such as proteins and DNA [7]. Hidden Markov models (HMMs) form a wide class of discrete-time stochastic processes, intensively used in many areas as speech recognition [31, 49], biology for heterogeneous DNA sequences analysis, neurophysiology, econometrics, time series analysis, and image segmentation [9].

Hidden Markov models as opposed to symbolic approaches has the important advantage, that the model parameters required can be trained automatically from sample data. Also, merely the parameters of the models and not their configuration, i.e., the structure and the number of free parameters can be determined automatically by the training algorithms. For this purpose even in the framework of statistical methods, the

experience of experts and extensive experimental evaluations are required. Furthermore, almost all known estimation methods require the availability of an initial model, which is then optimized step by step. The choice of the starting point, therefore, can critically influence the performance of the final Markov model [18]. The application of HMMs for pattern recognition tasks is always backed by the fundamental assumption that the patterns considered are at least in principle outputs of a comparable stochastic model and that the properties of this model can be described reasonably well by HMMs.

One of the most powerful properties of hidden Markov models is their ability to exhibit some degree of invariance to local warping of the time axis. HMM has shown lot of promise in area of speech processing [12, 18, 19, 49]. The trajectory classification is similar to the speech recognition tasks. A trajectory is a continuous quantity which can be described as the position of the object in time. HMMs are finite state stochastic machines that robustly model temporal variations in time series data which satisfies the Markovian property [1]. The basic theoretical strength of the HMM is that it combines modelling of stationary stochastic processes and the temporal relationship among the processes together in a well-defined probability space [68]. HMM are Stochastic automata that move within a finite set of states, emitting an observation in each step [52]. Each state has a distinct distribution over the possible observations, but in general, it is not possible to uniquely identify the state that a given observation was generated. The Viterbi algorithm finds the sequence states that is most likely to have generated a given observation sequence. The Baum-Welch algorithm, an instantiation of EM, can be used to estimate the most likely HMM parameters given a collection of observation sequences.

4.1 Problem suitability for HMM

HMM has many successful application in the areas of on-line Handwriting recognition, Speech recognition, Gesture recognition, Language modelling, Motion video analysis and tracking, Protein sequence/gene sequence alignment, Stock price prediction [24], sonar signal classification [32], acoustic scattering data [44]. An HMM includes a sequence of observations $O = o_1, o_2, ..., o_T$ and a sequence of hidden underlying states $S = s_1, s_2, ..., s_N$, which follow a Markov process [44]. Trajectory classification is

similar to the speech recognition tasks [5, 40]. A trajectory is a continuous quantity, that can be described analytically as the position of the object in time. Its discrete representation is used together with its temporal derivations (velocity and acceleration). Thus, an object trajectory 'O' is a potentially infinite sequence of state vectors. The trajectory classification problem can be formulated as to identify the class c_i (i = 1,...,L) to which belongs the trajectory state sequence.

The advantage of using HMM for trajectory classification can be stated as follows:

- Natural model structure: doubly stochastic process
  - sequences with temporal constraints
  - spatial variability along the sequence
- Efficient and good modelling tool for
  - sequences with temporal constraints
  - spatial variability along the sequence
  - real world complex processes
- Efficient evaluation, decoding and training algorithms
  - Mathematically strong
  - Computationally efficient
- Proven technology
  - Successful stories in many applications

The observation sequence from time t=1 to T is represented as $O = O_1, O_2, ..., O_T$ and the corresponding state sequence is given as $Q = q_1, q_2, ..., q_T$. A system being modelled for HMM has following conditions:

- System being Modeled is assumed to be a Markov Process with unobserved state. Generally the states are interconnected in such a way that any state can be reached from any other state. The individual states are denoted as $S = s_1, s_2, ..., s_N$.

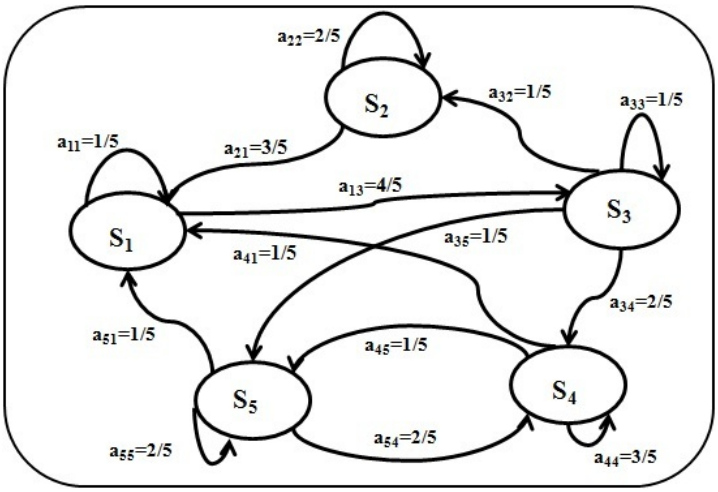

Figure 4.1: A 5-States System with state transition probabilities

- Has 'N' states and the state at time 't' is q_t.
- There are discrete time steps, t = 0, t = 1,...
- On the t^{th} time step the system is in exactly one of the available states, q_t
- Between each time step, the next state is chosen randomly as per the state transition matrix
- The current state determines the probability distribution for the next state
- The Probability Distribution of q_{t+1} depends only on q_t

Figure 4.1 shows one typical configuration of HMM with five states designated as $S_1, S_2, ..., S_5$ and one of possible transition probability values from one state to other. Transition $a_{11}, a_{22}, ..., a_{55}$ are self loops, whereas a_{ij} is transition probability from state i to state j. Sum of values of transition probabilities from one state to other states should be one.

4.2 HMM Formal Definition

Formally, HMM is defined by 5-tuple parameters

$$\lambda =< N, M, \pi, A, B >$$

where,

- N is the number of states in the Model. In HMM, though states are hidden, often some physical significance is attached.
- M is the number of possible observation symbols per state. The observation symbols correspond to the physical output of the system being modeled. Individual symbols will be denoted as $V = V_1, V_2, ..., V_k, ..., V_M$.
- $\pi_1, \pi_2, ..., \pi_N$ are the starting state probabilities.
  $P(q_0 = S_i) = \pi_i, \quad 1 \leq i \leq N$
  Defines the probability of initial state, q_0 at time t=0 to be in one of states S_i.
- The state transition probabilities $P(q_{t+1} = S_j | q_t = S_i) = a_{ij}$ is NxN matrix which indicates transition of state from S_i at time t to S_j at time t+1 and represented by the following matrix:

$$A = \begin{bmatrix} a_{11}\ a_{12}\ ...\ a_{1N} \\ a_{21}\ a_{22}\ ...\ a_{2N} \\ ...\ ...\ ... \\ a_{N1}\ a_{N2}\ ...\ a_{NN} \end{bmatrix}$$

- The observation probabilities $P(O_t = V_k | q_t = S_j) = b_j(k)$ is NxM matrix which indicates observation symbol probability distribution in state j at time t with j ranging from 1 to N and k ranging from 1 to M.

$$B = \begin{bmatrix} b_1(1)\ b_1(2)\ ...\ b_1(M) \\ b_2(1)\ b_2(2)\ ...\ b_2(M) \\ ...\ ...\ ... \\ b_N(1)\ b_N(2)\ ...\ b_N(M) \end{bmatrix}$$

4.3 Learning HMMs

The task of the learning algorithm is to find the best set of state transitions and observation (sometimes also called emission) probabilities. Given observation sequence $O_1, O_2, ..., O_T$, finding the maximum likelihood HMM that could have produced this string of observations at random is done using Baum-Welch (BW) Algorithm as described in section 4.5.3. In other words, adjusting the model parameters A, B, π in

order to maximize $P\{O|\lambda\}$, whereas a model (λ) and a sequence of observations $O = O_1, O_2, ..., O_T$ are given.

4.3.1 State Estimation Using Forward-Backward Algorithm

Given the observance sequence ($O_1, O_2, ..., O_T$) and a model ($\lambda(\pi, A, B)$), the probability of observance sequence $P(O|\lambda)$, is efficiently computed using recursive method. In other words, we have to choose the model that best matches the observations

$$P(q_T = S_i|O_1, O_2, ..., O_T)$$

4.3.2 Process of State Estimation Using Forward Algorithm

Defining Forward variable, $\alpha_t(i)$ as

$$\alpha_t(i) = P(O_1, O_2, ..., O_T, q_t = S_i|\lambda)$$

which is nothing but the probability of partial observation sequence, $O_1, O_2, ..., O_t$, (until time t) and state S_i at time t, given the model λ. $\alpha_t(i)$ is solved inductively as in step two:

- Initialization
  Initializes the forward probability as the joint probability of state S_i and initial observation O_1 $\alpha_1(i) = \pi_i b_i(O_1),\ \ 1 \leq i \leq N$
- Induction
  $\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i)a_{ij}\right] b_j(O_{t+1})\ \ 1 \leq t \leq T-1; 1 \leq j \leq N$
- Termination
  $P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$

  Figure 4.2 shows how state S_j can be reached at time t+1 from the N possible states, S_i , $1 \leq i \leq N$, at time t. The detail about forward variable is described as follows:

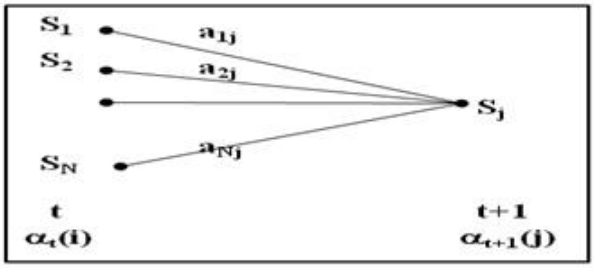

Figure 4.2: Sequence of operations for forward variable $\alpha_t(i)$

- Since $\alpha_t(i)$ is the probability of joint event that $O_1, O_2, ..., O_t$ are observed, and the state at time t is S_i, the product $\alpha_t(i).a_{ij}$ is then the probability of joint event $O_1, O_2, ..., O_t$ that are observed, and state S_j is reached at time t+1 via state S_i at time t.
- Summing this product over all the N possible states S_i ,$1 \leq i \leq N$ at time t results in the probability of S_j at time t+1 with all the accompanying previous partial observations.
- Once S_j is known, $\alpha_{t+1}(j)$ is obtained by accounting for observation O_{t+1} in state j, i.e. by multiplying the summed quantity by the probability $b_j(O_{t+1})$.
- The computation is performed for all states, j, running from 1 to N, for a given t.
- The computation is iterated for t=1,2,...,T-1.
- Finally, desired calculation of $P(O|\lambda)$ is obtained as the sum of the Terminal forward variable $\alpha_T(j)$.

4.3.3 Process of State Estimation Using Backward Algorithm

Backward Variable is defined as the probability of partial observation sequence from t+1 to end, given state S_i at time t and the model λ 4.3. Backward Variable

$$\beta_t(i) = P(O_{t+1}, O_{t+2}, ..., O_T, q_t = S_i|\lambda)$$

$\beta_t(i)$ is solved inductively as:

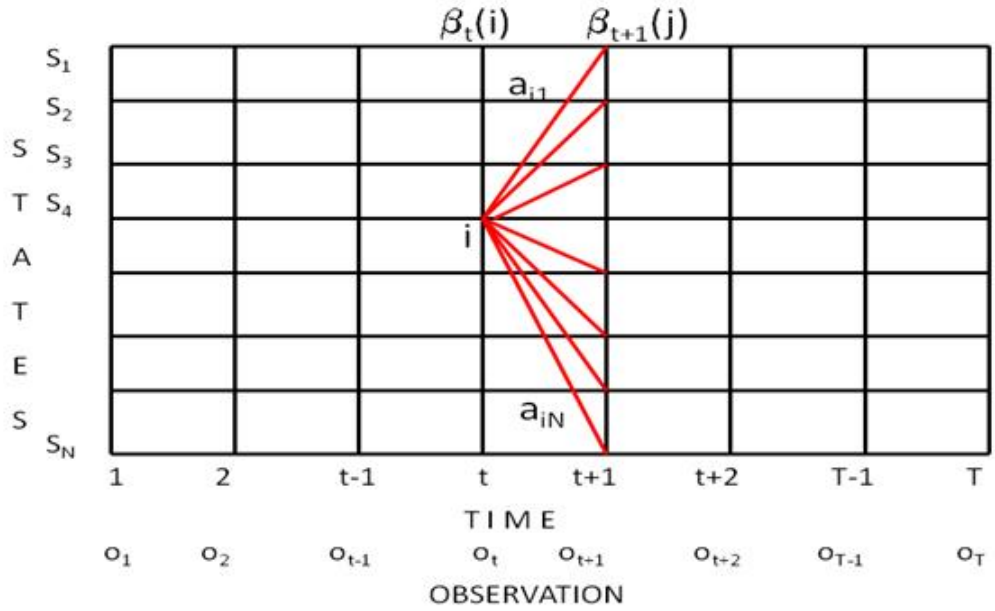

Figure 4.3: Sequence of operations required for the computation of $\beta_t(i)$

- Initialization : Initializes this step arbitrarily to be 1 for all i.
  $\beta_T(i) = 1\,; \quad 1 \leq i \leq N$

- Induction:
  $\beta_t(i) = \left[\sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)\right] \quad t = T-1, T-2, ..., 1; 1 \leq i \leq N$
  Induction step shows that in order to have in state S_i at time t, and to account for observation sequence from t+1 on, we have to consider all possible states S_i to S_j (a_{ij} term), as well as observation O_{t+1} in state $j(b_j(O_{t+1})$ term), and then account for the remaining partial observation sequence from state j ($\beta_{t+1}(j)$ term).

4.3.4 The optimal state sequence for the given observation sequence

There are several optimality criteria. One possible criterion is to choose the states q_t which is individually most likely. This criterion maximises the expected number of correct individual states. To implement this solution, we define $\gamma_t(i)$, the probability

of being in state S_i at time t, given the observation sequence and the model λ.

$$\gamma_t(i) = P(q_t = S_i|O, \lambda)$$

Expressing $\gamma_t(i)$ in terms of forward-backward variables, we get:

$$\gamma_t(i) = \alpha_t(i)\beta_t(i)/P(O|\lambda) = \alpha_t(i)\beta_t(i)/(\sum_{i=1}^{N} \alpha_t(i)\beta_t(i))$$

The normalization factor $P(O|\lambda) = \sum_{i=1}^{N} \alpha_t(i)\beta_t(i)$ makes $\gamma_t(i)$ a probability measure so that $\sum_{i=1}^{N} \gamma_t(i) = 1$. Solving individually most likely state q_t at time t is given by,

$$q_t = argmax[\gamma_t(i)], 1 \leq t \leq T, 1 \leq i \leq N = \alpha_t(i)\beta_t(i)/P(O|\lambda)$$

$$= \alpha_t(i)\beta_t(i)/(\sum_{i=1}^{N} \alpha_t(i)\beta_t(i))$$

4.4 Viterbi Algorithm

Viterbi Algorithm is a formal technique for finding single best state sequence based on dynamic programming methods. To find the single best state sequence, $Q = \{q_1, q_2, ..., q_T\}$, for the given observance $O = (O_1, O_2, ..., O_T)$ we need to define the quantity δ_t:

$$\delta_t(i) = \max_{q_1, q_2, ..., q_{t-1}} P[q_1, q_2, ..., q_t = i, O_1, O_2, ..., O_t|\lambda]$$

It is best score along a single path, at time t, which accounts for the first t observations and ends in state S_i. We Define another variable $\psi_t(j)$ to keep track of the argument which is maximised for each t and j. Viterbi algorithm is defined iterativerly as follows:

- Initialization :
  $\delta_1(i) = \pi_i(b_i(O_1)), \quad 1 \leq i \leq N$
  $\psi_1(i) = 0$

- Recursion :
  $\delta_t(j) = \max_{1 \leq i \leq N}[\delta_{t-1}(i)\ a_{ij}]b_j(O_t), \qquad 2 \leq t \leq T; 1 \leq j \leq N$
  $\psi_t(j) = argmax_{1 \leq i \leq N}[\delta_{t-1}(i)\ a_{ij}], \qquad 2 \leq t \leq T; 1 \leq j \leq N$

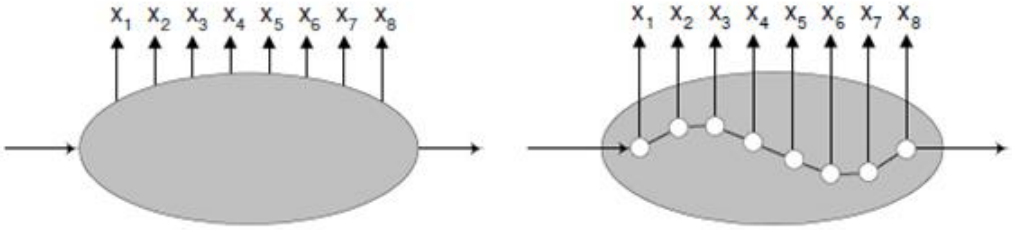

Figure 4.4: a) Observations recorded b) Obtaining most Probable States using VA

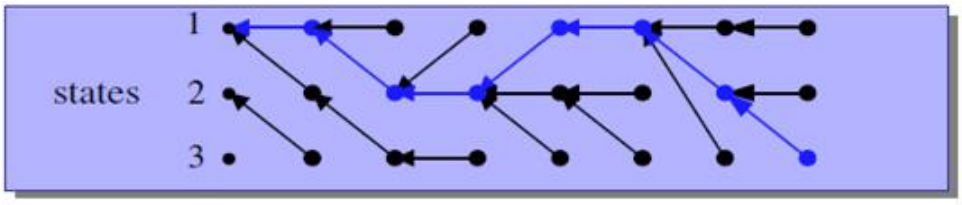

Figure 4.5: Backtracking from Terminal to Starting Node using VA

- Termination :
  $P^* = max_{1 \leq i \leq N}[\delta_T(i)]$
  $q_T^* = argmax_{1 \leq i \leq N}[\delta_T(i)]$

- Path (state sequence)backtracking

  $q_t^* = \psi_{t+1}(q_{t-1}^*), \quad t = T-1, T-2, ..., 1$

  Viterbi Algorithm (VA) is similar to forward calculation except for the backtracking step. The major difference is maximization in Viterbi recursion step in place of summing procedure. Figure 4.4 and Figure 4.5 demonstrate viterbi process.

4.5 Method to Adjust Model Parameters

The most widely used method for the optimization of HMMs is given by the so-called Baum-Welch algorithm [18]. The algorithm improves a given model λ depending on certain example data **O** in such a way that the optimized model generates the training

set with equal or greater probability:

$$P(O|\hat{\lambda}) \geq P(O, \lambda)$$

There is no analytical method to find a way to maximize the probability of observation sequence. We can choose $\lambda = (A, B, \pi)$ such that $P(O|\lambda)$ is locally maximized using iterative procedure such as Baum-Welch method or Expectation-Maximization method or using Gradient Technique.

4.5.1 Procedure for re-estimation of HMM parameters

Defining variable $\xi(i, j)$, the probability of being in state S_i at time t, and state S_j at time t+1, given the model and the observation sequence:

$$\xi_t(i, j) = P(q_t = S_i, q_{t+1} = S_j|O, \lambda)$$

4.5.1.1 Expected Number of S_i Visiting

$\gamma_t(i)$ the probability of being in state S_i at time t, given the observation sequence and the model λ.

$$\gamma_t(i) = P(q_t = S_i|O, \lambda)$$

$$= P(q_t = S_i, O/\lambda)/P(O/\lambda)$$

$$= \frac{\alpha_t(i)\beta_t(j)}{\sum_j \alpha_t(j)\beta_t(j)}$$

$$= \sum_{j=1}^{N} \xi_t(i, j)$$

$\sum_{t-1}^{T-1} \gamma_\tau(i)$ = expected number of transitions made from the state S_i.

4.5.1.2 Expected Number of Transitions

Expected number of transitions from state i to state j is represented by $\xi_t(i, j)$ and mathematically represented as:

$$\xi_t(i, j) = (\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j))/P(O|\lambda)$$

$$= \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{\sum_{i=1}\sum_{j=1}\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)} \quad i, j = 1, ..., N$$

Expected number of transitions made from the state S_i to state S_j over time period from start to time T is given as follows:

$$\sum_{t=1}^{T-1} \xi_t(ij) = \ expected\ number\ of\ transitions\ made\ from\ S_i\ to\ state\ S_j$$

4.5.2 A set of re-estimation formulas for π, A, B

A set of re-estimation formulas for π, A, B are

$$\pi_i = expected\ frequency\ in\ state\ Si\ at\ time(t=1) = \gamma_1(i)$$

$$a_{ij} = \frac{expected\ number\ of\ transitions\ made\ from\ the\ state\ S_i\ to\ state\ S_j}{expected\ number\ of\ transitions\ made\ from\ the\ state\ S_i}$$

$$b_j(k) = \frac{expected\ number\ of\ times\ in\ state\ j\ and\ observing\ symbol\ V_k}{expected\ number\ of\ times\ in\ state\ j}$$

finds HMM, $\lambda^* = maxP(O/\lambda)$ that maximizes the probability of the Observation O.

4.5.3 Steps of Expectation-Maximization (EM)

The EM algorithm is a general method of finding the maximum-likelihood estimate of the parameters of an underlying distribution from a given data set when the data is incomplete or has missing values [6].

1. Get Observations $O_1, O_2, ..., O_T$
2. First estimate $\lambda(0)$, k=0
3. k=k+1
4. Given $O_1, O_2..., O_T, \lambda(k)$, Compute

$$\gamma_t(i), \xi_t(i,j) 1 \leq t \leq T, 1 \leq i \leq N, 1 \leq j \leq N$$

5. Compute expected frequency of visiting state i, and expected frequency of State, i moving to state j
6. Compute new Estimates of $a_{ij}, b_j(k), \pi_i$ accordingly. Call them $\lambda(k+1)$
7. Goto Step 3, unless Converged

4.5.4 Stochastic constraints of the HMM parameters

Stochastic constraints of the HMM parameters that need to get satisfied at each iteration are:

$$\sum_{i=1}^{N} \pi_i = 1$$

$$\sum_{j=1}^{N} a_{ij} = 1 \;\; 1 \leq i \leq N$$

$$\sum_{k=1}^{M} b_j(k) = 1 \;\; 1 \leq j \leq N$$

- Convergence Criteria
  The convergance criteria for HMM is when change of maximum value of parameters becomes less than 0.005, that is, Value=Max(DeltaPI, DeltaA, DeltaB). If (Value $<$ 0.005), Stop Iteration.

4.6 Formulation of Model for Trajectory Classification

Flow of data from radar to decision making for HMM is shown in Figure 4.6. The radar measures only the position of the target and this information is noisy. The noisy position data is processed by using the Kalman filter as mentioned in section 2.3. The Kalman filter estimates position, velocity and acceleration by processing the noisy measured position data. The filtered kinematic parameters of missiles are passed to linear quantizer to convert the data into discrete forms before it passes through discrete HMM. For the present work, we are using 4 kinematic parameters, which are computed based on the kalman filter estimates. These 4 parameters are (a) Specific energy (SE), (b) Acceleration, (c) Altitude, and (d) Velocity. For sake of experimentation, the numbers of classes considered are three, namely M400 class of ballistic missiles(BM) corresponding to 400 Km range, M1000 class of BM corresponding to 1000 Km range and M2000 class of BM corresponding to 2000 Km range.

4.6.1 Digitisation

In order to represent a signal with sufficient accuracy in digital form, it is sampled, i.e. the analog values are measured at certain regular time intervals, and subsequently quantized. In this way, the analog quantities are mapped onto a finite discrete domain

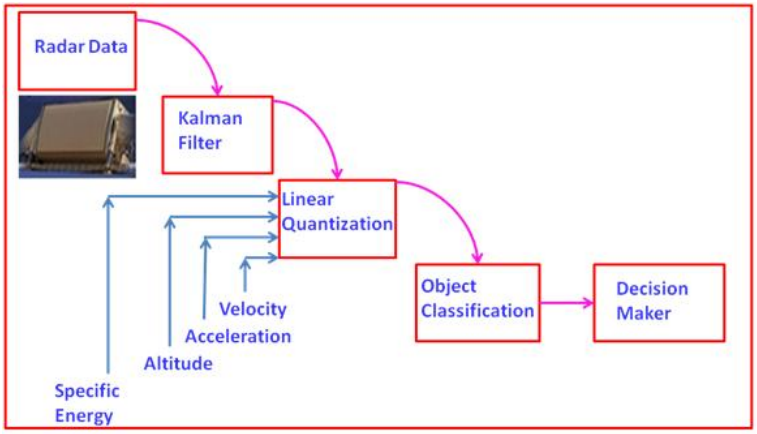

Figure 4.6: Processing Sequence for HMM Trajectory Classification

of values. The combination of sampling and quantization is referred to as digitization [18]. Missile trajectory is sampled with 10Hz and four inputs of specific energy, acceleration, altitude and velocity are mapped upto 200, 40, 60 and 80 discrete values. Figure 4.7(b) to Figure 4.10(b) show discrete values of specific energy, acceleration, altitude and velocity for the corresponding Figure 4.7(a) to Figure 4.10(a).

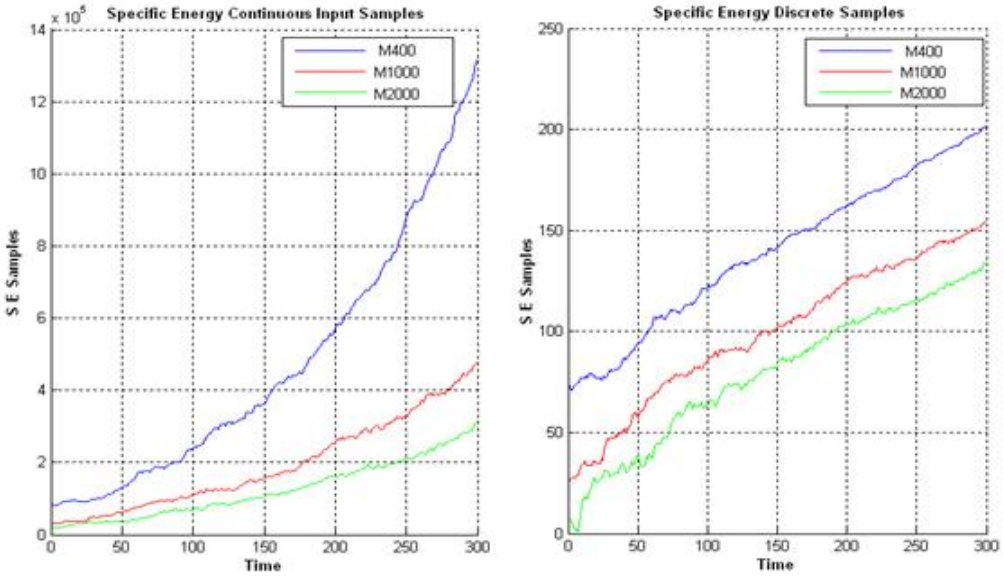

Figure 4.7: SE Input data a) Sampled by Radar b) Discrete Data with 200 Symbols

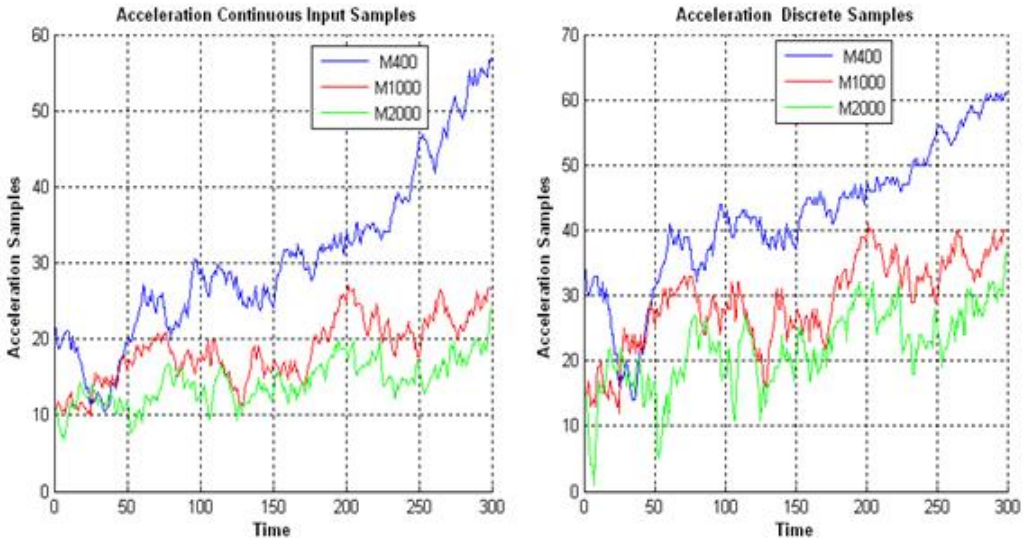

Figure 4.8: Acceleration Input data a) Sampled by Radar b) Discrete Data with 40 Symbols

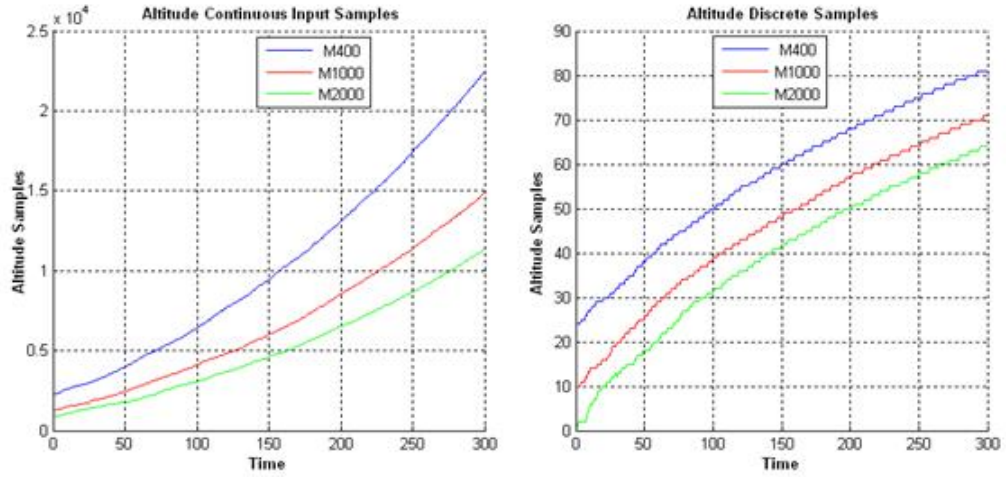

Figure 4.9: Altitude Input data a) Sampled by Radar b) Discrete Data 60 symbols

4.6.2 Trajectory Classification by Hidden Markov model (HMM)

As mentioned earlier, HMMs are finite state stochastic machines that robustly model temporal variations in time series data which satisfies the markovian property [20]. The basic theoretical strength of the HMM is that it combines modelling of station-

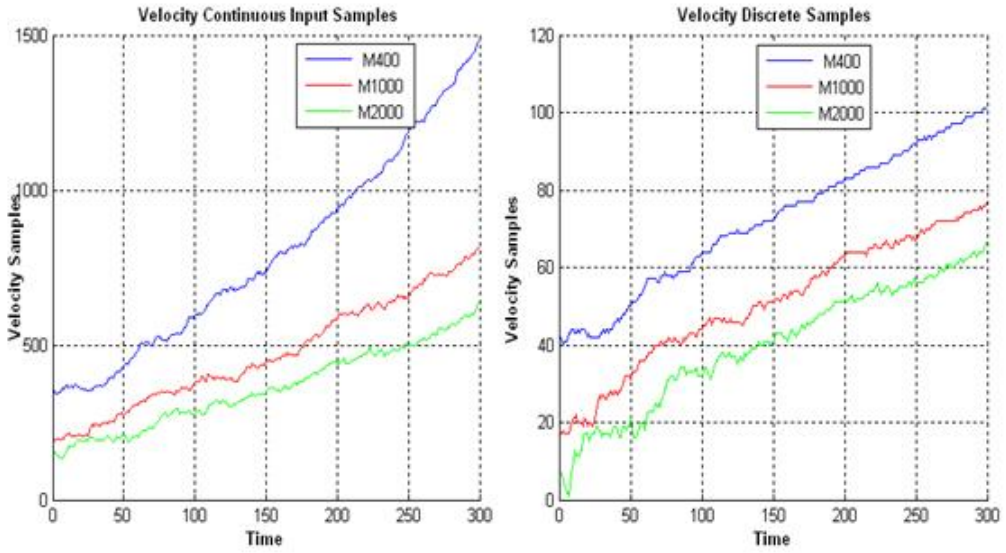

Figure 4.10: Velocity Input data a) Sampled by Radar b) Discrete Data 80 symbols

ary stochastic processes and the temporal relationship among the processes together in a well-defined probability space [49]. Considering the trajectory of missile of a particular class to be described at any time as being in one of a set of N distinct states $s_1, s_2, ..., s_N$. At regularly spaced discrete times, the system undergoes a change of state according to a set of probabilities associated with the state. Denoting the time instants associated with the state changes as $t = 1, 2, \ldots,$ the actual state at time t is denoted as q_t.

The trajectory classification problem is formulated as to identify the class $C_i (i = 1, \ldots, L)$ to which the trajectory state sequence belongs. Each trajectory class C_i is represented by a Model, λ_i , where $\lambda_i = \pi, A, B$ in which π is initial state probabilities, A is state transition probabilities of dimension (N x N) and B is observation symbol probability distribution of dimension (N x M). Number of states for HMM for missile classification are considered as N=25; Number of observation symbols M, for specific energy is 200, acceleration is 40, height is 60 and velocity is 80. Initial state probability, π and state transition probabilities are set to be equal to 1/N. Observation symbol probability B, is based on gaussian (with mean and variance) representation for the distributions of observations within each state.

HMM recognizer as given in Figure 4.11 shows three HMM models $(\lambda_1, \lambda_2, \lambda_3)$

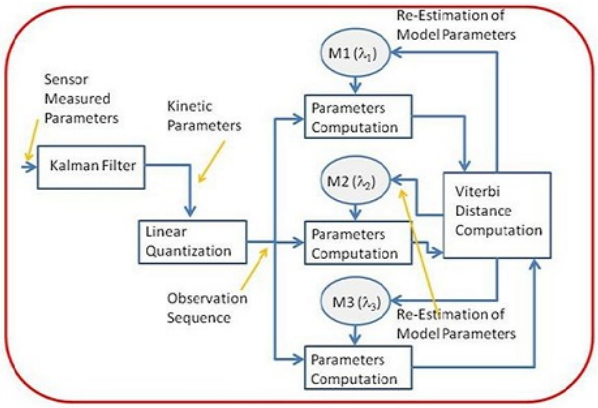

Figure 4.11: HMM Models for Trajectory Classification

corresponding to three different types of trajectories belonging to M400, M1000 and M2000 class of missiles. The HMM recognizer can be extended to any number of classes with enough memory for computation and number of states suitably. Radar measured data is passed through the Kalman filter as indicated by first block of Figure 4.11. Observation sequence is linearly quantized and passed through Baum-Welch [7, 49] computation to arrive at updated parameters of HMM model. Viterbi [7, 49] distance is used to classify how a new sample is related with trained HMM model. The steps involved in unified training and testing can be summarized as follows:

1. Define all normal trajectory classes in the scene.
2. Define initial models corresponding to each class.
3. Grouped Data from ART-2 output are taken and models are adapted according to them using Baum-Welch algorithm.
4. The output is a new classification model.
5. New set of data is evaluated against each HMM model using Viterbi distance.
6. Least Viterbi distance is taken as criteria to assign new data to the corresponding class.
7. New data is added at the end of running window of that class and first data is deleted.

8. Training is repeated to adapt the models.

9. Step (5) to (8) is repeated for all new set of data.

Termination criteria of BW iteration occurs if maximum changes for consecutive iteration for all three $\{\pi, A, B\}$ parameters are less than 10e-8. Re-estimation formulas for $\{\pi, A, B\}$ are given in [49] and stated in section 4.5.2.

4.7 Results and Analysis

Target trajectory data of different missiles required to test and validate the network is taken from 6-DOF (degree of freedom) simulation data and noise is added to make it near realistic scenario. The trajectory information is corrupted with the noise as per the radar measurement accuracies. The measurement accuracies considered for azimuth and elevation channels are 2 milli-radians and range with 20 m. The measurement data is passed to estimate the target state i.e. position, velocities and acceleration. Based on this estimated data the kinematic parameters required for the model are computed. The input data of specific energy, acceleration, altitude and velocity as mentioned in section 4.6.1 are fed with periodicity of 100 milli-seconds (ms) with uniform track rate. As mentioned in section 4.6.2, the HMM recognizer has three models corresponding to three classes of missiles M400, M1000 and M2000. Figure 4.11 shows in diagrammatic form how input data is passed through KF, its linear quantization of continuous data and three models of HMM corresponding to three classes.

4.7.1 Missile Classification with Noise

Figure 4.12 and Figure 4.13 shows viterbi distance of three classes of missiles for a typical run for 900 samples, each with 300 samples. Figure 4.12(a) clearly shows that M400 class of trajectory samples are segregated from other two classes for most of the duration. However, at some points, few samples of M400 trajectory are misclassified as M1000 class. Similarly, Figure 4.13(b) shows that M1000 class of trajectory samples are separated from M400 and M2000 class of missiles. Figure 4.13(a) shows that 300 samples of M2000 class of trajectory samples are distinct from other two classes of M400 and M1000 class.

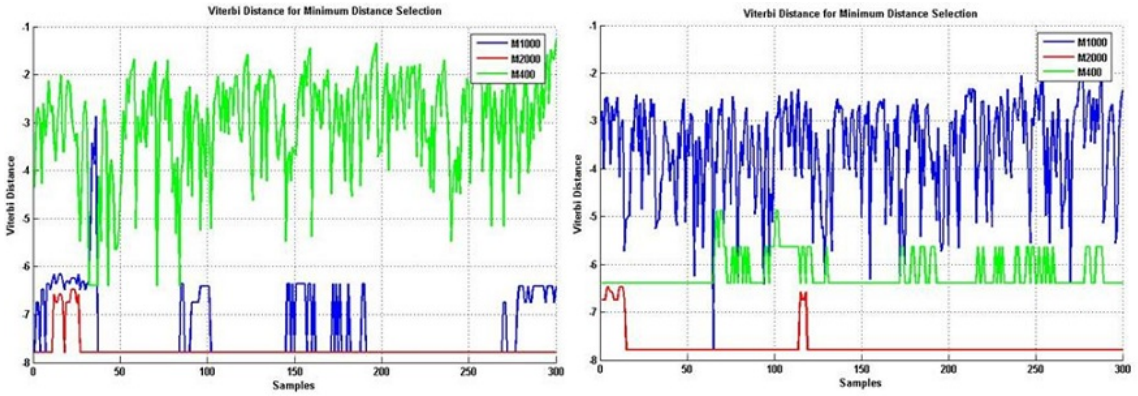

Figure 4.12: Results of HMM for a) M400 b) M1000 Class of Missile for 300 samples

Figure 4.13(b) shows how well desired and computed outputs are matching. Desired symbols for M400, M1000 and M2000 classes are (2, 1, 0) respectively where a symbol count is incremented during testing if computed values finds match with correct class. This figure shows that for most of the duration, desired and computed values are overlapping except for very small duration.

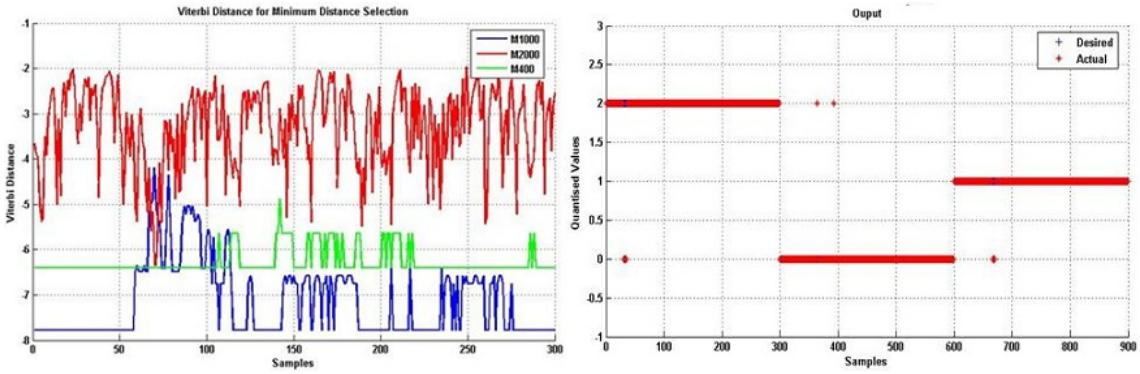

Figure 4.13: Results of HMM for a) M2000 b) Category Class of Missile for 300 samples

Results of HMM as recorded in Table 4.1 demonstrate that pass percentage achieved is more than 98% over 900 samples of test data. However, misclassification is less than 2%. In table 4.1, confusion matrix shows that all but two samples of trajectory of M1000 class of missile are classified correctly. However, M2000 class of missile could correctly classify 296 samples out of 300 samples whereas 4 samples are misclassified

Table 4.1: Performance of HMM Model for Noisy Data

		Actual Class					
		Class 1(Pass/Fail)		Class 2(Pass/Fail)		Class 3(Pass/Fail)	
		Count	Percentage	Count	Percentage	Count	Percentage
Predicted Class	Class M1000	2698/ 300	99.33%	4	1.33%	5	1.66
	Class M2000	0	0	296 / 300	98.66%	0	0.00%
	Class M400	2	0.66%	0	0	295 / 300	98.33%

as M1000. In case of trajectory of M400, 295 samples are correctly classified whereas 5 samples are misclassified as M2000 as is evident in Figure 4.12 and Figure 4.13.

As shown in Table 4.2, HMM takes 1.234 sec for training of 20 samples of initial values in addition to overheads for output storage. HMM takes 29.546 sec of time for unified testing and training of 600 samples which brings 49.353 ms for one sample. The trajectory is validated on model with different number of states in HMM. HMM with 20 states, 25 numbers of states and above are giving results above 95 % whereas performance of the model remains unsatisfactory with 5, 10 or 15 states. Time taken by the model for testing new sample of data for 5, 10, 15, 20 and 25 states are 4.816 milli-seconds (ms), 8.02 ms, 18.17 ms, 29.06 ms and 49.24 ms respectively.

Table 4.2: Timing for Training and testing

Samples	Training Time(Sec)	Number of States	Test Samples	Testing Time	Time/ Samples
20 Samples	1.234	25	600	29.546 ms	49.353 ms
	0.863	20		17.424 ms	29.040 ms
	0.597	15		10.902 ms	18.020 ms
	0.412	10		04.812 ms	08.020 ms
	0.304	5		02.889 ms	04.816 ms

Table 4.3: Results with 20 Samples Loss

		Actual Class					
		Class 1(Pass/Fail)		Class 2(Pass/Fail)		Class 3(Pass/Fail)	
		Count	Percentage	Count	Percentage	Count	Percentage
Predicted Class	Class M1000	296/ 300	98.66%	0	0	0	0
	Class M2000	4	1.33%	298 / 300	99.30%	0	0
	Class M400	0	f0	2	0.66%	280 / 280	100%

4.7.2 Results with Samples Losses

To validate the robustness of network, input data to the network is removed for 2 seconds equivalent to 20 samples in order to simulate the loss of radar track temporarily from one of the class(M400). Figure 4.14 shows specific energy (SE) plot of M400 class of missile without and with loss of 20 samples of data. Plot in blue is original and that in red is with loss of data samples. The model continues to classify the input data correctly as presented in Table 4.3 with the help of confusion matrix. All correct classification lies on diagonal elements of matrix where magnitude of correct classification is indicated by numerator and denominator denotes total samples passed for validation.

Confusion matrix in Table 4.3 clearly indicates correct classification of 296 out of 300 for class-1, 298 out of 300 for class-2 and 280 out of 280 for class-3. Thus, the correct classification of 98.66 % for Class1 (M1000), 99.33 % for Class2 (M2000) and 100 % for Class3 (M400) is obtained. Total correct classification percentage for 900 samples are 99.31% and incorrect classification of 0.69 % is obtained for all 3 classes.

Figure 4.15 and Figure 4.16 show results of outputs of the HMM model for three classes of trajectory of M400, M1000 and M2000 class of missile. Class category of M400, M1000 and M2000 class of missile are shown in Figure 4.16(b).

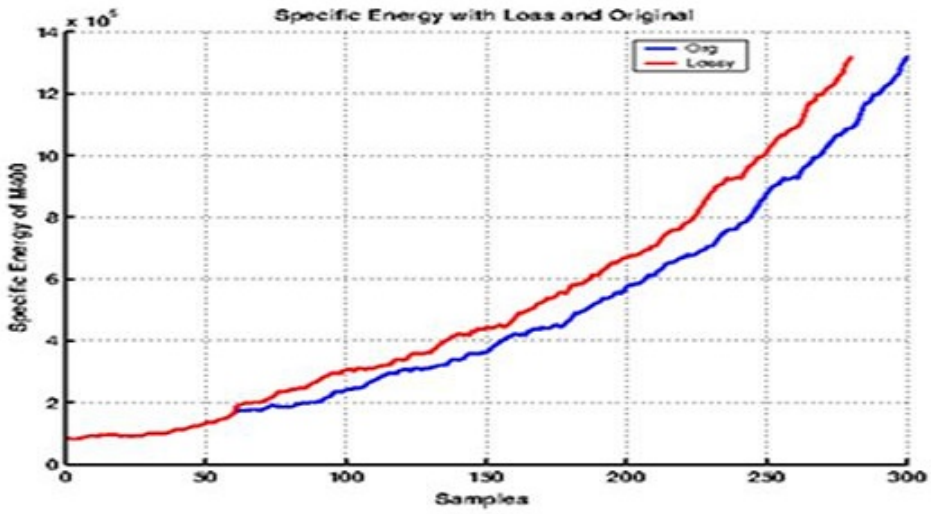

Figure 4.14: Specific Energy for Trajectory with 2 seconds loss for M400

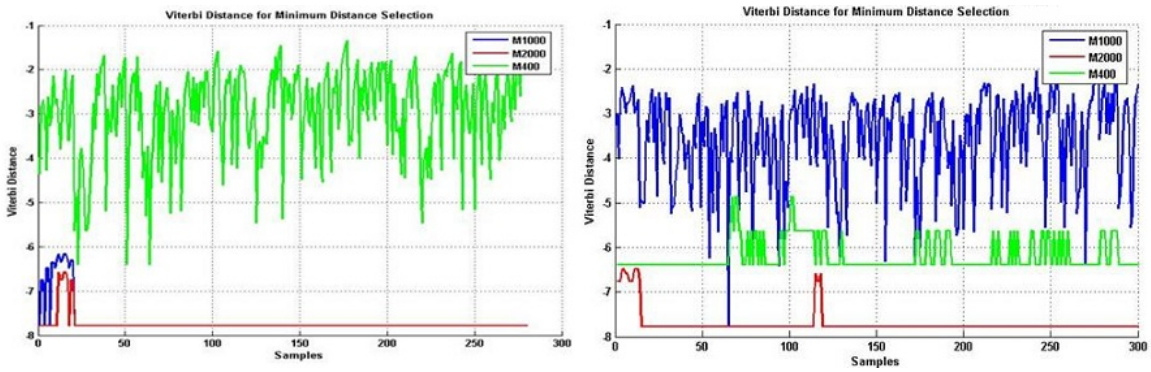

Figure 4.15: Results of HMM for a) M400 b) M1000 Class of Missile for 280 samples

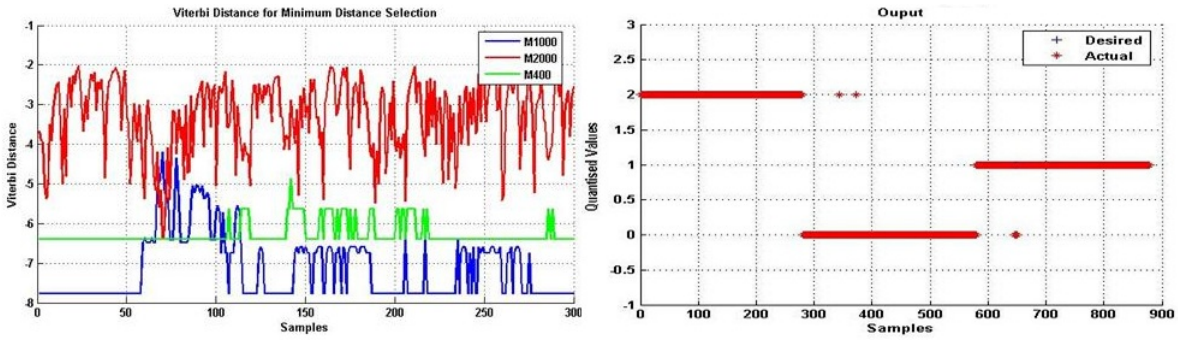

Figure 4.16: Results of HMM for a) M2000 b) Category Class of Missile for 280 samples

Table 4.4: Trajectory Deviation Results

		Actual Class					
		Class 1(Pass/Fail)		Class 2(Pass/Fail)		Class 3(Pass/Fail)	
		Count	Percentage	Count	Percentage	Count	Percentage
Predicted Class	Class M1000	300/ 300	100%	4	1.33%	0	0
	Class M2000	0	0	296 / 300	98.66%	2	0.66%
	Class M400	0	0	0	0	338 / 340	99.40%

4.7.3 Results with Deviations in Trajectory

To verify and confirm the model robustness further, the HMM model is checked against manoeuvring trajectory as shown in Figure 4.17 to Figure 4.18 corresponding to four input parameters. Quasi-ballistic missiles with manoeuvering capability are being developed in different countries these days to escape neutralization by ballistic missile defence. Therefore it is necessary to classify such trajectory for its annihilation.

Forty samples of data are added in the trajectory to mimic quasi-ballistic missile trajectory. Changes in input parameters are depicted as in Figure 4.17(a) for specific energy, Figure 4.17(b) for acceleration, Figure 4.18(a) for altitude and Figure 4.18(b)for velocity. Segregation of trajectory parameters is made in such a way that there is clear distinction of input parameters throughout the period of test regime. Results for input of values of Figure 4.17 and Figure 4.18 are demonstrated in Table 4.4. Confusion matrix in Table 4.4 indicates correct classification of 300 out of 300 for class-1, 296 out of 300 for class-2 and 338 out of 340 samples for class-3 trajectory. Percentage of success for each of the class is given in second column of each class. This indicates the correct cumulative classification of 99.36%, incorrect classification for 0.64%.

The Viterbi distance for trajectories of three classes of M400, M1000, M2000 are presented in Figure 4.19 and Figure 4.20. The desired class category for three classes of M400, M1000 and M2000 are (2, 1, 0). The desired class type is shown in blue

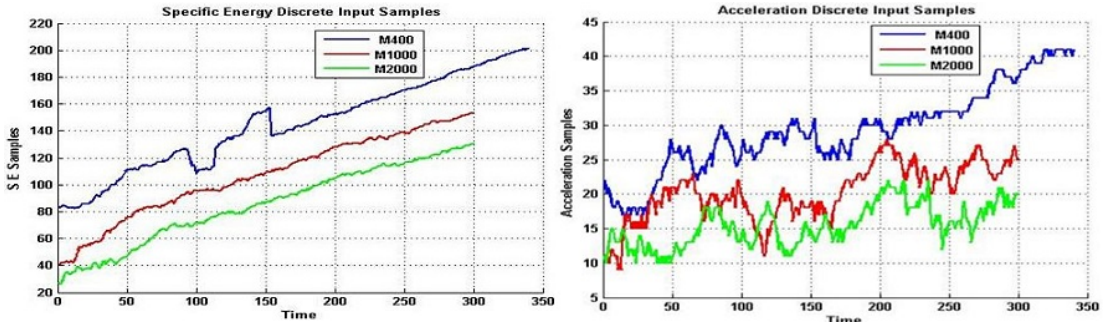

Figure 4.17: a) SE b) Acceleration Input data with Deviation in M400 Class of Missile

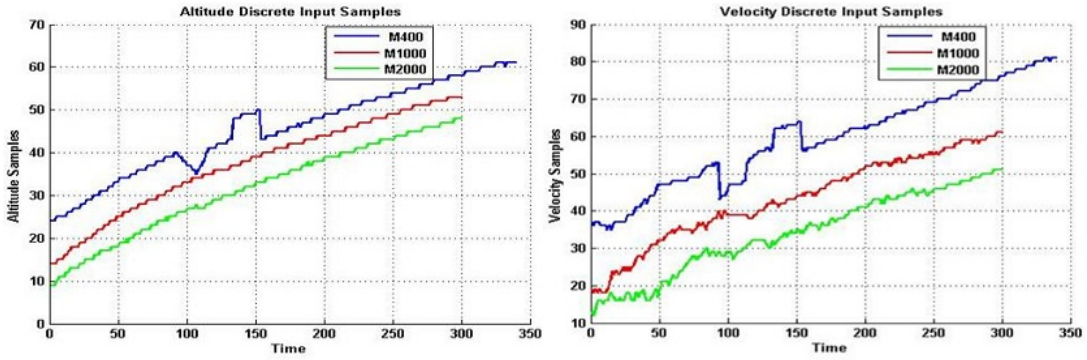

Figure 4.18: a) Altitude b) Velocity Input data with Deviation in M400 Class of Missile

colour, whereas actual class is shown in red. Few samples which are misclassified are displaced from its mother type as shown in Figure 4.20(b).

4.8 Summary of HMM

In this chapter, we have presented a framework for classification of ballistic missiles trajectory using HMM in real-time with Baulm-Welch algorithm. Mathematical model is developed to simulate and generate trajectories of different ranges of ballistic missiles, which is utilized for testing and evaluation of HMM. Moving window concept is also used for classifying trajectories using HMM, which enables us to integrate training and testing while taking care of short as well as long range trajectories. Further, we have established that the HMM is able to classify the new sample in less than 50 milliseconds which makes it useful for real-time applications. The model performance

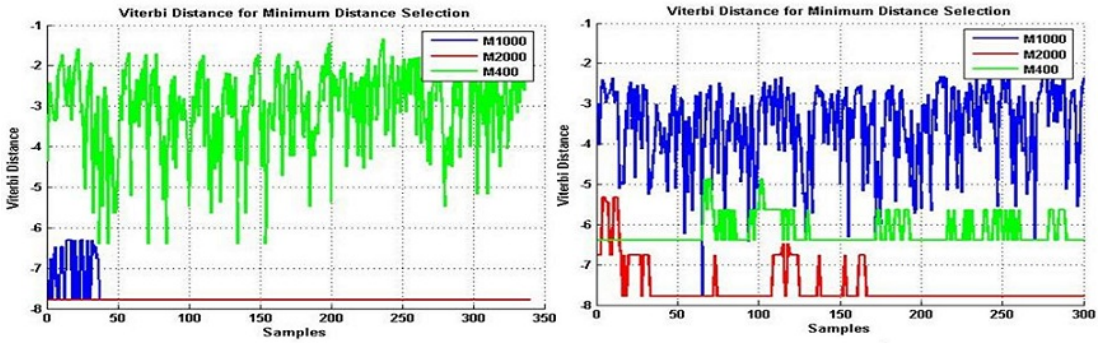

Figure 4.19: Results of HMM for a) M400 b) M1000 Class of Missile for 340 samples

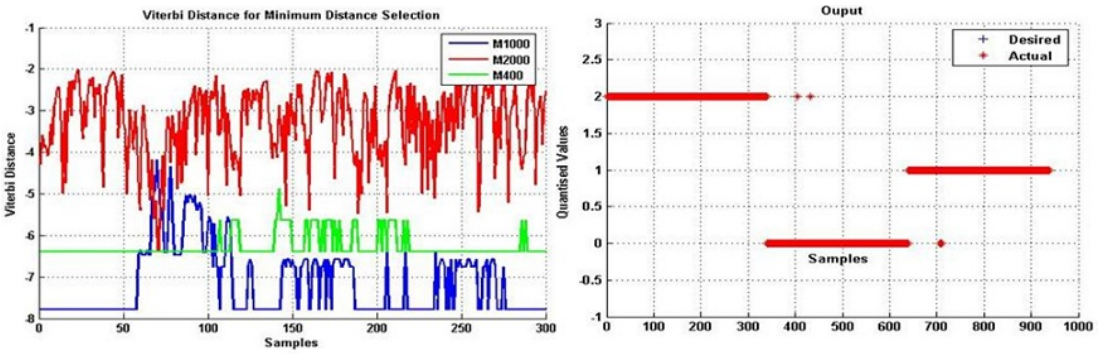

Figure 4.20: Results of HMM for a) M2000 b) Category Class of Missile for 300 samples

is around 95% although time taken by HMM is near 50 ms on Pentium-4 processor. Analysis of parameters used for the experiment are summarised as follows:

- Larger window size
  - Captures larger Characteristics of the moving body
  - Takes care of sample losses
  - Becomes wieldy, training and testing time increases
- Smaller window size
  - Deviations in sample inputs makes it useless
  - Faster training and testing time

Table 4.5: Summary results of HMM

HMM	**Noisy Data**			**Lossy Trajectory**			**Deviation Trajectory**		
	M1000	M2000	M400	M1000	M2000	M400	M1000	M2000	M400
Pass %age	99.3	98.66	98.3	98.66	99.3	100	100	98.6	99.4
Failed %age	0.66	1.33	1.66	1.33	0.66	00	0	1.33	0.66

  - Useful if network allows faster and reliable data inputs

- Classification correctness

  - Greater than 95%

Table 4.5 shows that for all cases from filtered noisy data to samples losses to deviated trajectory, pass percentage is above 98% , whereas fail percentage is less than 2%.

In this chapter, we have been able to demonstrate that HMM can classify missile trajectories with high accuracies with careful selection of number of states and model window size.

Chapter 5

Adaptive Resonance Theory (ART)

In networks with supervised learning algorithms, such as a perceptron or a backpropagation network, an input training set is presented sequentially until the network finishes learning the entire training set. When an additional pattern is presented to the network, the network has to be completely retrained with all the training patterns. If the network learns an additional pattern alone, in the process, it forgets the previous learning and classifies the previously learned patterns incorrectly. *Plasticity* denotes the ability of a network to learn a new pattern, and the ability for the new learning not to be affected by the previous learning is called *stability*. The quest for stable-plastic networks led to the development of ART networks [13, 33].

Adaptive resonance theory (ART) is a new type of adaptive neural network for competitive learning, designed to overcome the problem of learning stability. The learning instability occurs because of the networks adaptability (or plasticity), which causes prior learning to be eroded by more recent learning. The strength of the network is receptiveness to significant new patterns and yet remains stable in response to irrelevant patterns. Grossberg and Carpenter [11] developed the ART to address the stability plasticity dilemma. The key innovation of ART is the use of 'expectations' and can be summarised as follows:

- As each input is presented to the network, it is compared with the prototype vector that is most closely matches the expectation.
- If the match between the prototype and the input vector is NOT adequate, a new prototype is selected. In this way, previous learned memories (prototypes) are

not eroded by new learning.

5.1 Basic Architecture

The basic architecture of an adaptive neural network consists of three groups of neurons, namely F1 Layer, F2 layer and a Reset Unit.

F1 layer : Also called Input Processing units, has two parts : Input Units (F1-(a)) and Interface Units (F1-(b)). Interface Unit (IU) receives input from external world. Some processing is also done in this unit in ART-2. IU combine signals from the input units and F2 Layer to compare the input signal similarity to the weight vectors of the cluster units that has been selected as the candidate for learning.

F2 layer : Also called Cluster Units (CU), is a competitive layer. The unit with the largest network input becomes the candidate to learn the input pattern. The activation of all other F2 units is set to zero. Information from the input and cluster units are combined by the interface units. Learning of the input pattern by this cluster unit is based on the similarity between its top-down weight vector and the input vector. The reset unit makes this decision based on the signals received from the interface and input portions of the F1 layer. If the reset units decision is to not allow the cluster unit to learn, then a new cluster unit is selected as the candidate.

Reset Units : The degree of similarity of patterns placed in the same cluster is controlled by a reset mechanism via a vigilance parameter, ρ [64]. Each time a pattern is presented, an appropriate cluster unit is chosen, and the cluster's weights are adjusted to let the cluster unit learn the pattern.

Connection Weights: There are two sets of connections between layers F1(b) and F2. The bottom-up weights connecting F1(b) to F2 are denoted by b_{ij} and the top-down weights connecting F2 to F1(b) are designated t_{ji}.

5.1.1 ART networks

Figure 5.1 demonstrates basic architecture of ART. The main features of network are:

- Learning of ART networks occurs in a set of feedback connections from layer-2(L2) to layer-1(L1). These connections are outstars which perform pattern recall.

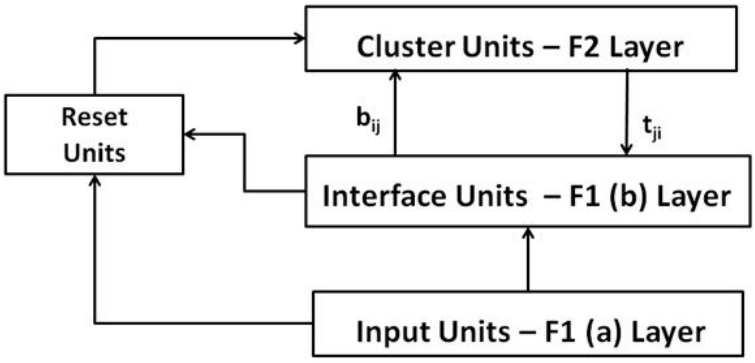

Figure 5.1: Basic ART Architecture

- When a node in Layer-2 is activated, this reproduces a prototype pattern (the expectation) at layer-1.
- Layer-1 then performs a comparison between the expectation and the input pattern.
- When the expectation and the input pattern are NOT closely matched, the orienting subsystem causes a reset in layer-2.
- The reset disables the current winning neuron, and the current expectation is removed.
- A new competition is then performed in layer-2, while the previous winning neuron is disabled.
- The new winning neuron in layer-2 projects a new expectation to Layer 1, through the L2-L1 connections.
- This process continues until the L2-L1 expectation provides a close enough match to the input pattern.

5.1.2 Learning Laws and Modes of Learning

These are two separate learning laws for ART-2: One for the L1-L2 connections, (instar) and another for L2-L1 connections (outstar). Both L1-L2 connections and L2-L1 connections are updated at the same time, whenever the input and the expectation have

an adequate match. The process of matching, and subsequent adaptation, is referred to as resonance. Learning can occur in two modes: *fast learning* and *slow learning*. Slow learning is less susceptible to noise, and is not influenced by the order of presentation of the patterns. Fast learning reduces learning time and is useful for real-time applications.

Fast Learning : On any particular trial weight updates occur rapidly during resonance relative to the length of time a pattern is presented. The weights reach equilibrium on each trial as shown in figures given in towards the end of this chapter (Figure5.6 to Figure 5.11). A component of the weight vector for a particular cluster unit that is set to zero during a learning trial will not become nonzero during a subsequent learning trial. The weights of the nonzero components will change on each learning trial to reflect the relative magnitudes of the nonzero components of the current input vector.

Slow Learning : Relative to the duration of a learning trial the weight changes occur slowly; on a particular trial the weights do not reach equilibrium. More number of presentations of the patterns are required in the case of slow learning than is required for fast learning. On each learning trial only one weight update with a small learning rate takes place.

5.2 Adaptive resonance Theory (ART-2)

ART-2 is suitable to operate for continuous-valued inputs. More complex F1 field takes care of input vectors arbitrarily close together. Y_j units compete in a winner-take-all mode for the right to learn each input pattern. ART-2 treats small components as noise and does not distinguish between patterns that are merely scaled versions of each other.

5.2.1 ART-2 Components

- **F1 Field**
  - Includes a combination of normalization and noise suppression.
  - Compare bottom-up and top-down signals needed for reset mechanism.
  - Consists of six types of units (W, X, U, V, P, Q).

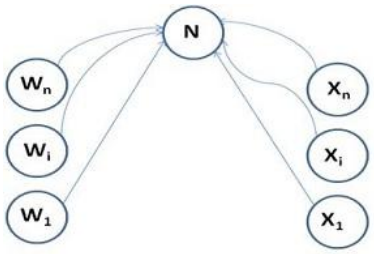

Figure 5.2: Connections from W units to X Units

- A supplemental unit between the W and X units receives signals from all of the W units, computes the norm of the vector W, and sends this (inhibitory) signal to each of the X units. Each of these also receives an excitatory signal from the corresponding W unit.
- A similar supplemental unit performs the same role between the
  - P and Q units, and
  - V and U units
- Each X unit is connected to the corresponding V unit, and each Q unit is connected to the corresponding V unit also.
- U units perform a role to the input phase of the F1 layer.
- P units play the role of the interface F1 units.
- Units X and Q apply an activation function to their network input.
- Suppresses any components of the vectors of activations at those levels that fall below the user-selected value q. The connection paths from U to W and Q to V have fixed weights a and b, respectively.

- **F2 Field**
  The action of the F2 layer (units Yj) units compete in a winner-take-all mode for the right to learn each input pattern. Figure 5.2 shows supplemental unit N to perform normalization between W and X Units.

5.2.2 Parameter Choices

Table 5.1 shows the parameters chosen for arriving at fast and correct classification. The significance of each parameter is stated as under:

Table 5.1: Parameter choice for HMM

Variable Name	**description**	**Value**
NoF1	Number of input units (F1 layer)	4
NoF2	Number of input units (F2 layer)	4
NoEpochs	Number of Epochs	1
NoITER	Number of Iterations	200
a,b	Fixed weights in the F1 layer	10,10
c	Fixed weight used in testing for reset	0.1
d	Activation of winning F2 unit	0.9
θ	Noise suppression parameter	1/sqrt(NoF1)
α	Learning rate	0.3
ρ	Vigilance parameter	0.999984
$t_{ji}(O)$	Initial Top-down weight	0
$b_{ij}(O)$	Initial Bottom-up Weights	$1/((1$ - d$)sqrt$(NoF1))

- "An epoch is one presentation of each pattern".
- "Setting either a = 0 or b = 0 produces instability in the network; other than that, the network is not particularly sensitive to the values chosen."
- "A small c gives a larger effective range of the vigilance parameter. Typical value is 0.1."
- "c and d must be chosen to satisfy the inequality $c * d/(1 - d) \leq 1$ in order to prevent a reset from occurring during a learning trial. The ratio should be chosen close to 1 to achieve a larger effective range for vigilance."
- θ "is the noise suppression parameter. Components of the normalized input vector that are less than this value are set to zero. For our experiment, value of θ is taken as zero."
- "Along with the initial bottom-up weights, vigilance (ρ) parameter determines formation of number of clusters. Value of ρ should lie between approximately

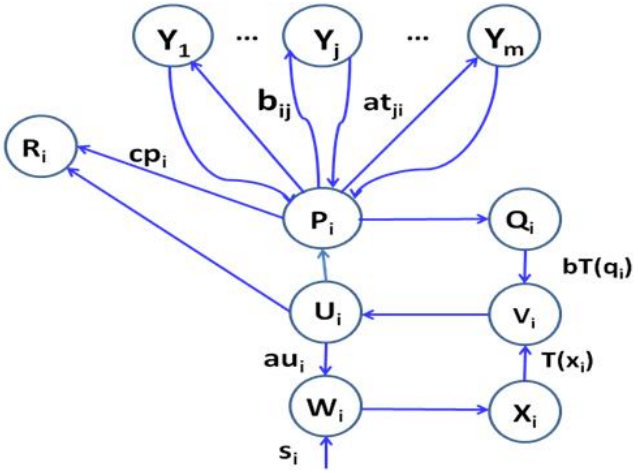

Figure 5.3: ART-2 Architecture

0.7 and 1 to perform any useful role in controlling the number of clusters."

- "Initial top-down weights must be small to ensure that no reset occurs for the first pattern placed on a cluster unit."

- "Initial bottom-up weights must be chosen to satisfy the inequality"

$$b_{ij}(0) \leq \frac{1}{(1-d)\sqrt{NoF1}}$$

to prevent the possibility of a new winner being chosen during "resonance" as weights change. Larger values of b_{ij} encourage the network to form more clusters.

5.2.3 ART-2 Algorithm

A typical ART-2 architecture [17] is re-produced in Figure 5.3. There are additional layers present in the F1 layer compared with the basic architecture presented in Figure 5.1. Let **x** = $(x_1, x_2, \ldots, x_n)$. $N(x) = Normalize(x) = \frac{x}{||x||}$ where, $||x||$ is norm of vector **x** and it is represented as: $||x|| = \sqrt{((x_1)^2 + (x_2)^2 + ... + (x_n)^2)}$. $T(x) = Threshold(x) = (T(x_1), T(x_2), \ldots, T(x_n))$ where $T(x_i) =$

$$Threshold(x_i) = \begin{cases} x_i & \text{if } x_i \geq \theta \\ 0 & \text{if } x_i < \theta \end{cases}$$

A learning trial consists of one presentation of one input pattern. The input signal $\mathbf{s} = (s_1, ..., s_i, ..., s_n)$ continues to be sent while all of the actions are performed. The computation cycle for a particular learning trial within the F1 layer can be described as follows:

- Vectors **u**, **q** and **p** are initialized to zero and **w** is set input vector **s** as mentioned in step-3 in the later portion of this section.
- Thus, **x** gets input only from **s** and **v** gets thresholded value of **s** during first time. Unit U receives normalized value of **v** to pass it to W_i unit after multiplication with constant 'a'.
- Next, a signal is sent from each unit P_i to its associated units W_i and P_i. The activations of units W_i and P_i are then computed. Unit W_i sums the signal it receives from P_i and the input signal s_i.
- P_i sums the signal it receives from V_i and the top-down signal it receives if there is an active F2 unit. The activations of X_i and Q_i are normalized versions of the signals at W_i and P_i, respectively.
- An activation function is applied at each of these units before the signal is sent to V_i. V_i then sums the signals it receives concurrently from X_i and Q_i. This completes one cycle of updating the F1 layer.
- After two updates of the F1 layer, activations of the V and P units reach equilibrium. Any signal that is less than θ is treated as noise by this function and is suppressed i.e., it is set to zero. The value of the parameter θ is specified by the user, which we have set to zero. Due to noise suppression stable cluster formation is acheived by the network. The network is stable when the first cluster unit chosen for each input pattern is accepted and no reset occurs.
- The P units send their signals to the F2 layer once the activations of the F1 units have reached equilibrium. Thereafter a candidate cluster unit is chosen to learn the input pattern through a winner-take-all competition.

- The corresponding reset unit Ri receives signals from the units V_i and P_i in the F1 layer. Each time a signal from P_i is received, the reset mechanism can check for a reset as the necessary computations are based on the value of the most recent signal the unit R has received from V_i. The point to note is that this needs to be done only when P_i first receives a top-down signal because parameter values are chosen such that no reset will occur if no F2 unit is active, or after learning has started.

- The candidate cluster unit will either be rejected as not similar enough to the input pattern or will be accepted once the conditions for reset have been checked. On rejection of the cluster unit, it will be prevented from further participation in the current learning trial. The cluster unit with the next largest network input is chosen as the candidate. Until an acceptable cluster unit is chosen (or the supply of available cluster units is exhausted) the process continues. Learning will occur when a candidate cluster unit that is chosen passes the reset conditions.

The algorithm consists of the following steps :

1. **Step-0**:
   Initialize the following parameters of the ART-2 network
   $a, b, q, c, d, e, \alpha, \rho$
   Do NoF1(number of input units in F1 layer) times for variable i
   Do NoF2(number of input units in F2 layer) times for variable j

   begin
   $t_{ji} = 0$
   $b_{ij} = \frac{1}{(1-d)\sqrt{NoF1}}$
   end

2. **Step-1**
   Do steps 2 to 12 for NoEpochs (number of epochs) times

3. **Step-2**
   Do steps 3 to 11 for each input vector **s**.

4. **Step-3**

   $\mathbf{u} = 0$ $\quad$ $\mathbf{w} = \mathbf{s}$ $\quad$ $\mathbf{p} = 0$

   $\mathbf{x} = N(\mathbf{s})$ $\quad$ $\mathbf{q} = 0$ $\quad$ $\mathbf{v} = T(\mathbf{x})$

   $\mathbf{u} = N(\mathbf{v})$ $\quad$ $\mathbf{w} = \mathbf{s} + a * \mathbf{u}$

   $\mathbf{p} = \mathbf{u}$ $\quad$ $\mathbf{x} = N(\mathbf{w})$

   $\mathbf{q} = N(\mathbf{p})$ $\quad$ $\mathbf{v} = T(\mathbf{x}) + b * T(\mathbf{q})$

5. **Step-4**

   Compute the network input of the F2 units

   Do NoF2 times (for variable j)

   begin

   $y_j = \sum_{i=0}^{NoF1} b_{ij} p_i$

   end

6. **Step-5**

   While reset is true, do steps 6-7

7. **Step-6**

   Find F2 unit $Y_J$with largest network input, $J = j$.

8. **Step-7**

   Check for reset $\mathbf{u} = N(\mathbf{v})$

   Do NoF1 times (for variable i)

   begin

   $p_i = u_i + d * t_{Ji}$

   $r_i = \frac{u_i + c * p_i}{||\mathbf{u}|| + ||\mathbf{p}||}$

   end

   if $||\mathbf{r}|| < \rho$ then

   begin

   $Y_J = -1$(inhibit J^{th} node from participation)

   reset=true

   repeat step 5

   end

   if $||\mathbf{r}|| \geq \rho$ then

begin

$\mathbf{w} = \mathbf{s} + a\mathbf{u}$ $\quad\quad$ $\mathbf{x} = N(\mathbf{w})$

$\mathbf{q} = N(\mathbf{p})$ $\quad\quad$ $\mathbf{v} = T(\mathbf{x}) + bT(\mathbf{q})$

reset=false

end

go to step 8

9. **Step-8**
   Do steps 9 to 11 for NoITER times (NoITER-Number of Iterations)

10. **Step-9**
    Update weights for unit J.
    $T_{Ji} = \alpha * d * u_i + \{1 + \alpha * d * (d-1)\} * t_{Ji}$
    $B_{iJ} = \alpha * d * u_i + \{1 + \alpha * d * (d-1)\} * b_{iJ}$

11. **Step-10**
    $\mathbf{u} = N(\mathbf{v}),$ $\quad\quad$ $\mathbf{w} = \mathbf{s} + a\mathbf{u}$
    $\mathbf{p} = \mathbf{u} + d * \mathbf{t}_J,$ $\quad\quad$ $\mathbf{s} = N(\mathbf{w})$
    $\mathbf{q} = N(\mathbf{p}),$ $\quad\quad$ $\mathbf{v} = T(\mathbf{s}) + b * T(\mathbf{q})$

12. **Step-11**
    Test stopping condition for weight updates

13. **Step-12**
    Test stopping condition for number of epochs

5.3 Experimental Set-Up and Results

To take advantage of the unsupervised learning of ART-2 on continuous data, initial samples as collected from radar are subjected to ART-2 learning and classification. The code is implemented in visual C++ on Windows platform. First 20 samples of SE, acceleration, altitude and velocity, each from M400, M1000 and M2000 class of trajectories as recorded in first four columns in Table 5.2 to Table 5.4 are passed through ART-2 network. The network clustered the 60 patterns (20 each given in Table 5.2, Table 5.3 and Table 5.4) into three different clusters.

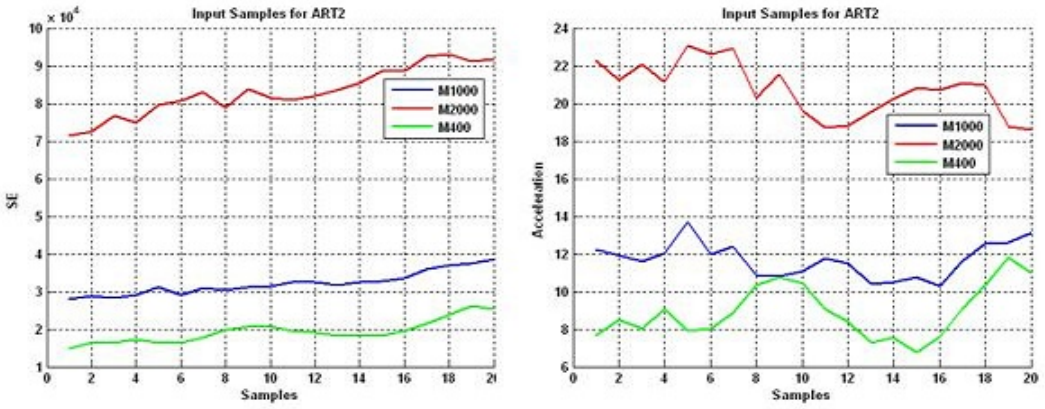

Figure 5.4: Input Samples of a) Specific Energy b) Acceleration

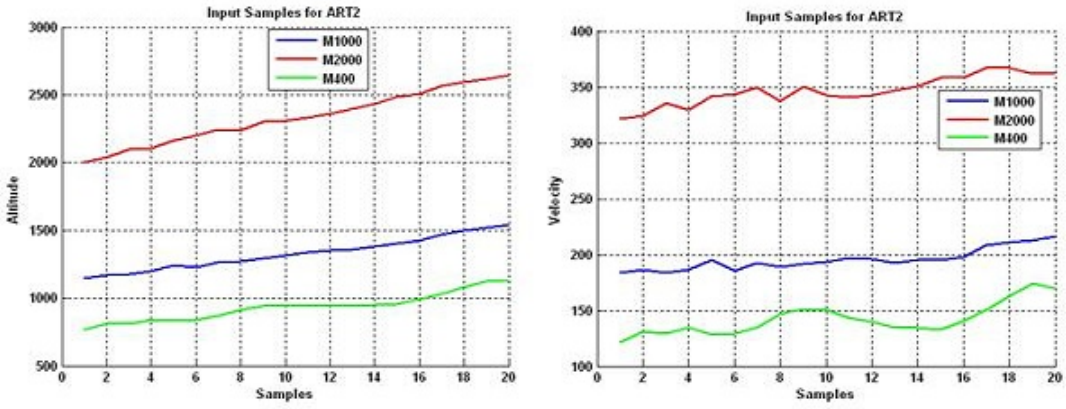

Figure 5.5: Input Samples of a) Altitude b) Velocity

After carefully choosing parameters of the network as in Table 5.1, results which has been arrived at, are tabulated in 6^{th} column of Table 5.2 to Table 5.4 against expected values in column 5^{th} for M400, M1000 and M2000 class of missile. 7^{th} column defines values for layer R which is compared with vigilance parameter for decision of creating a template for new class or placing new set of samples in same class. A value of 0.999984 is used for the vigilance parameter, and fast learning mode is used for updating weights (NoITER = 200). The noise suppression parameter (θ) is chosen to be 0. The network converged in one epoch only. An epoch is one presentation of each pattern to the network. Figure 5.4 and Figure 5.5 give graphical representation of input samples of four parameters of SE, acceleration, altitude and velocity provided to

Table 5.2: M400 Class of Missile

SE	Acc	Altitude	Velocity	Expected Output	Actual Output	Layer R
14047.9	12.2	1146.6	183.6	0	0	1
14403.0	11.9	1169.9	186.2	0	0	1
14188.6	11.6	1174.3	183.7	0	0	1
14576.9	12.0	1197.1	186.7	0	0	1
15612.1	13.7	1238.6	195.4	0	0	0.999999
14601.7	12.0	1225.4	185.4	0	0	0.999998
15431.9	12.4	1262	192.3	0	0	1
15134.7	10.9	1270.1	188.8	0	0	1
15572.1	10.8	1295.8	192.1	0	0	1
15782.3	11.1	1314.5	193.3	0	0	1
16219.4	11.8	1339.3	196.5	0	0	1
16205.2	11.5	1353.2	195.7	0	0	1
15899.0	10.4	1360.2	192.2	0	0	0.999999
16290.0	10.5	1383.9	195.0	0	0	1
16366.4	10.7	1399.3	195.0	0	0	1
16776.9	10.3	1424.2	198.0	0	0	1
18010.5	11.6	1469.8	207.9	0	0	0.999999
18459.6	12.5	1494.7	211.1	0	0	1
18726.5	12.6	1516.1	212.6	0	0	1
19218.9	13.1	1542.3	216.0	0	0	1

the network.

First four column of Table 5.2 to Table 5.4 gives input samples of four parameters of specific energy (SE), acceleration, altitude and velocity. Fifth and sixth columns show expected and actual output of the class. Last column is layer R output which along with vigilance parameter determines formation of number of different classes. Table 5.2 to Table 5.4 clearly shows that expected and actual output are matching for vigilance parameter $\rho = 0.999998$. The time taken by ART-2 parameter for 60 set of samples with one epoch and 200 number of iterations is 2ms.

Initial top-down weights are initialized to zero, whereas initial bottom-up weights

Table 5.3: M1000 Class of Missile

SE	Acc	Altitude	Velocity	Expected Output	Actual Output	Layer R
35696.7	22.3	2000.7	321.8	1	1	1
36201.1	21.2	2038.4	323.8	1	1	1
38390.5	22.1	2100.4	335.3	1	1	1
37387.8	21.1	2105.9	329.1	1	1	1
39794.2	23.1	2164.7	341.7	1	1	1
40317.7	22.6	2198.6	343.8	1	1	1
41484.9	22.9	2243.4	349.2	1	1	1
39399.8	20.3	2239.7	337.2	1	1	0.999999
41902.2	21.5	2301.2	350.0	1	1	1
40710.8	19.6	2309	342.9	1	1	1
40479.2	18.7	2331.8	340.9	1	1	1
40923.9	18.8	2360.5	342.7	1	1	1
41791.9	19.6	2399.5	346.6	1	1	1
42669.4	20.2	2433	350.7	1	1	1
44320.8	20.8	2485.6	358.6	1	1	1
44330.7	20.7	2509.3	358.0	1	1	1
46206	21.1	2566.1	366.8	1	1	1
46471.7	21.0	2595.4	367.4	1	1	1
45579.9	18.8	2613.4	362.1	1	1	1
45914.8	18.6	2646.5	363.0	1	1	1

are initialized to 1/(1-d)*$\sqrt{NoF1}$. NoF1 in our experiment is 4 and value of 'd' is set to 0.9. Thus, values of bottom weights are initialized to 5.0.

Many experiments were conducted by varying vigilance parameters and results are recorded in Table 5.5. The vigilance parameter is very sensitive to the set of data being presented to the network. The value of vigilance parameter suitable for data set of the given trajectories is 0.999995. Smaller values (0.9 to 0.999) in first two rows of Table 5.5) classified all data set into same class and larger values (0.999999 to 0.9999995) in the last two rows of Table 5.5) breaks the class into finer classes.

Table 5.4: M2000 Class of Missile

SE	Acc	Altitude	Velocity	Expected Output	Actual Output	Layer R
7452.1	7.7	769.3	121.4	2	2	1
8288.6	8.5	809.6	131.5	2	2	0.999997
8166.5	8.0	814.0	129.3	2	2	1
8627.7	9.1	837.1	134.5	2	2	0.999999
8231.4	7.9	832.5	128.9	2	2	0.999999
8288.0	8.0	841.9	129.0	2	2	1
8819.0	8.9	869.0	135.1	2	2	0.999999
9886.5	10.3	915.5	147.0	2	2	0.999996
10319.1	10.8	939.5	151.2	2	2	0.999999
10309.5	10.5	949.2	150.4	2	2	1
9781.9	9.1	942.9	143.7	2	2	0.999998
9546.4	8.4	944.8	140.2	2	2	0.999999
9157.6	7.3	941.9	134.8	2	2	0.999998
9209.9	7.5	951.6	134.9	2	2	1
9141.0	6.8	960.4	133.2	2	2	1
9808.3	7.6	993.2	140.6	2	2	0.999999
10713.1	9.1	1031.7	150.4	2	2	0.999997
11926.5	10.3	1080.5	162.9	2	2	0.999995
12612.6	11.8	1125.9	174.3	2	2	0.999999
12681.7	11.0	1125.4	169.3	2	2	1

5.4 Summary of ART-2

The results obtained in this chapter indicate the potential of using unsupervised ART-2 neural networks for trajectory classification of missiles from kinetic parameters of the vehicles captured by radar. The main advantages of the ART-2 neural network approach are high recognition speed, ability to classify very close trajectories, minimal memory storage, and ease of computation. Since, time taken by ART-2 network for training of 60 samples is only 2ms, the network is suitable for real-time applications.

Table 5.5: Sensitivity of Vigilance Parameter

Vigilance Sensitivity	**M400**	**M1000**	**M2000**	**Remarks**
0.9	60	00	00	All are classified in the same class
0.999	60	00	00	All are classified in the same class
0.999985	20	32	08	12 of M2000 are classified as M1000
0.999988	20	23	17	03 of M2000 are classified as M1000
0.999995	20	20	20	All classification Done Properly
0.999999	20	15	07	05 of M1000 as Separate Class 09 of M2000 as M1000 Class 04 of M2000 as Another Class
0.9999995	19	14	01	01 of M400 is classified as M1000 06 samples of M1000 are broken into three classes of size 04, 01, 01 19 samples of M2000 are classified as M1000

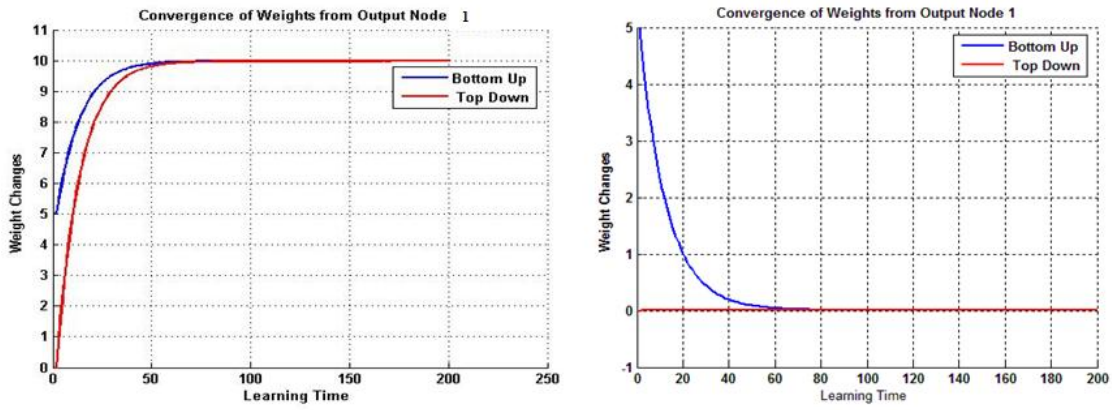

Figure 5.6: Changes of Weights a) Node1 (t_{11} and b_{11}) b) Node1 (t_{12} and b_{21})

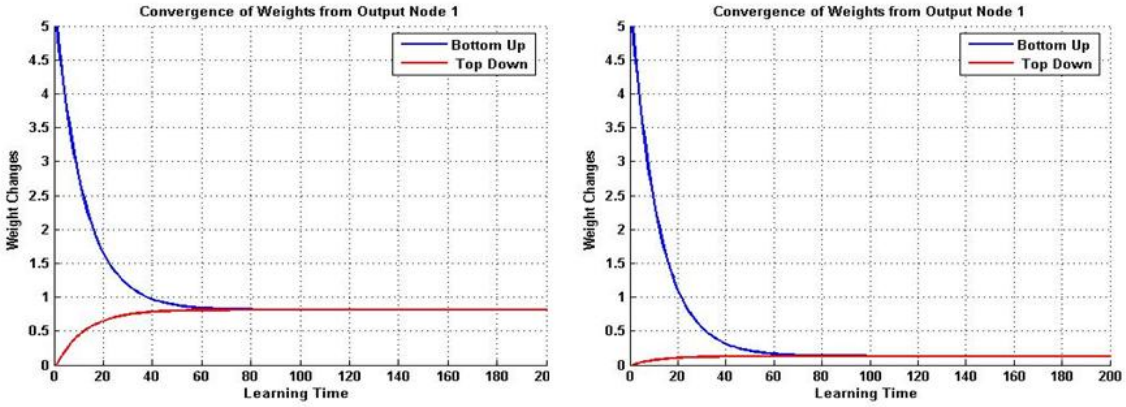

Figure 5.7: Changes of Weights a) Node1 (t_{13} and b_{31}) b)Node1 (t_{14} and b_{41})

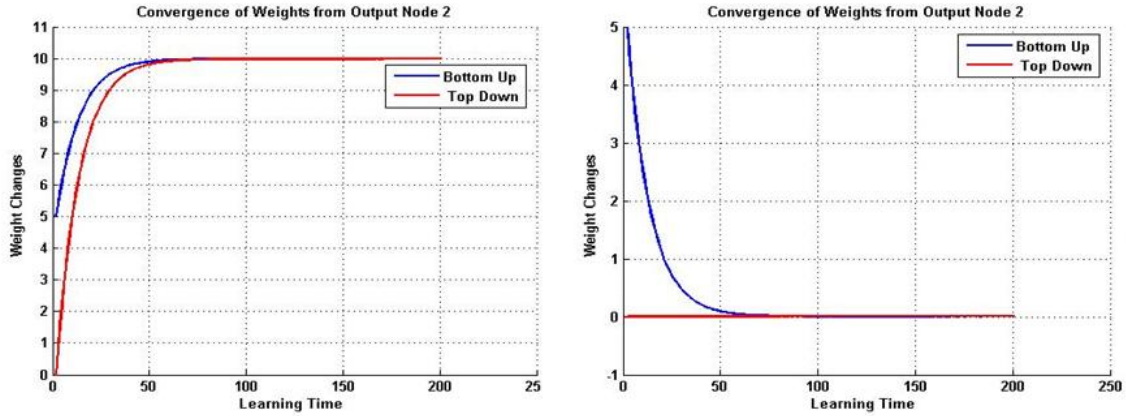

Figure 5.8: Changes of Weights a) Node2 (t_{21} and b_{12}) b) Node2 (t_{22} and b_{22})

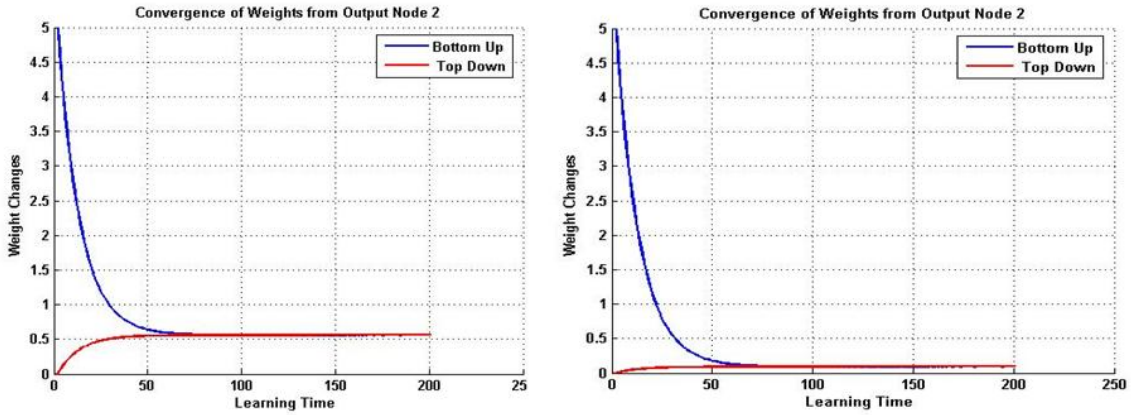

Figure 5.9: Changes of Weights a) Node2 (t_{23} and b_{32}) b) Node2 (t_{24} and b_{42})

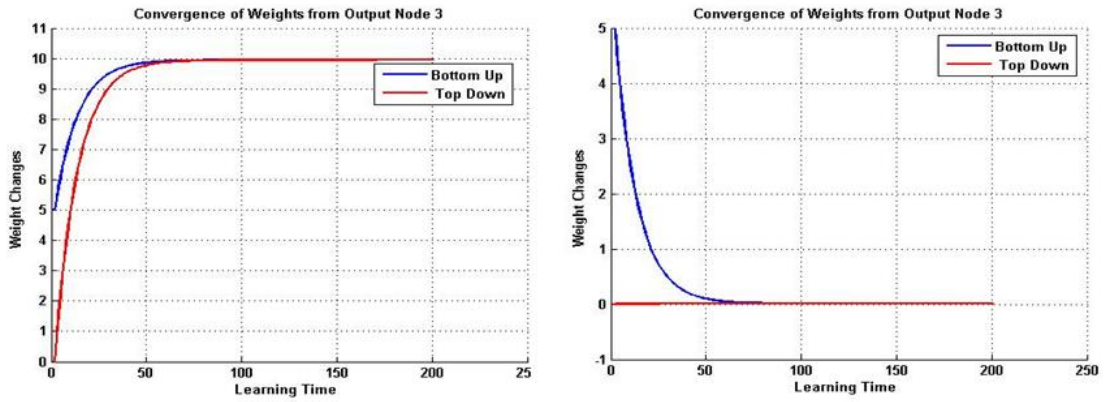

Figure 5.10: Changes of Weights a) Node3 (t_{31} and b_{13}) b) Node3 (t_{32} and b_{23})

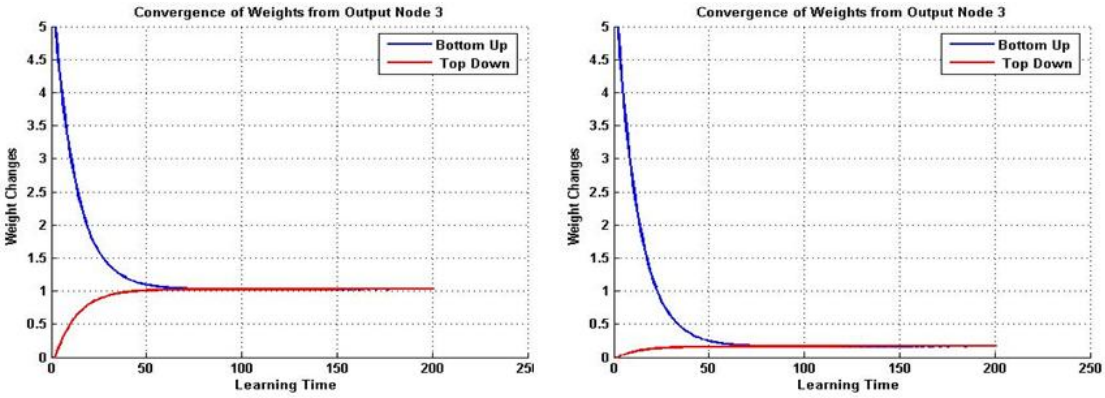

Figure 5.11: Changes of Weights a) Node3 (t_{33} and b_{33}) b) Node3 (t_{34} and b_{43})

Chapter 6

Generalisation & Performance Comparison of RTNN and HMM

To check generalisation of the network and the model presented in section 3.1 and section 4.6.2, we extend the network model for both RTNN and HMM to test larger classes. For RTNN, the number of nodes in output layer of Figure 3.1 is extended from two to three for classes more than four but less than 9 using 2^n formula, where n is number of output nodes. Similarly, the number of node-set in second hidden layer is made five for 5 groups. The number of HMM models for five classes are considered as five and two more models λ_4 and λ_5 in Figure 4.11 are added to cater for two additional classes.

6.1 Input Parameters

Let (x,y,z), (V_x, V_y, V_z), (a_x, a_y, a_z) are target position, velocity and acceleration given by the Kalman filter in geographic frame. The parameters required for classification are computed by using the following formulas:

$$The\ Total\ Velocity\ is\ V = \sqrt{{V_x}^2 + {V_y}^2 + {V_z}^2}$$

$$The\ Total\ Acceleration\ is\ A = \sqrt{{a_x}^2 + {a_y}^2 + {a_z}^2}$$

$$And\ the\ height\ is\ given\ by\ h = z\ and SE = gh + 0.5V^2.$$

As described in section 3.4.1 and 4.6, we are using 4 kinematic parameters, which are computed based on the Kalman filter estimates. These 4 parameters are: (a) Specific energy (SE), (b) Acceleration (A), (c) Altitude (h) and (d) Velocity (V). Two additional class of missiles (M600 and M900) are added to compare five classes of missiles for validating model performance. They are: a) M400 Class of Ballistic Missile b) M600 Class of Ballistic Missile c) M900 Class of Ballistic Missile d) M1000 Class of Ballistic Missile, and e) M2000 Class of Ballistic Missile.

Any missile can be programmed to fly in nominal, lofted and depressed mode as shown in Figure 6.1. The duration of flight may vary depending on modes of shaping the trajectory of missiles. All experiments have been done for nominal flight. Figure 6.2 shows the total velocity profile for M400, M600, M900, M1000 and M2000 class of missiles. Similarly, Figure 6.3 and Figure 6.4 show the corresponding height and specific energy for the above mentioned five classes of missile.

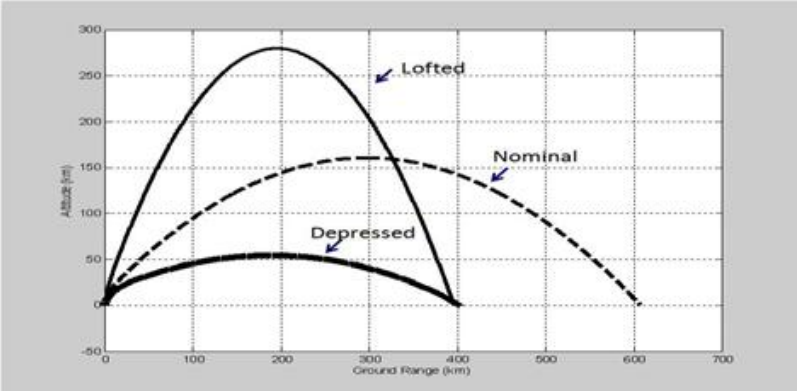

Figure 6.1: Different Trajectory behaviour followed by the same class of Vehicle

The M400 class of missile can travel for duration of 353 seconds with maximum velocity and maximum altitude of 1800 m/s and 115 Km respectively. Similarly, M1000 class of missile goes with maximum velocity and maximum altitude of 2815 m/s and 250 Km respectively for duration of 540 sec. The M2000 class of missile travels with maximum velocity and maximum altitude of 4200 m/s and 430 Km respectively for a period of 742 sec. M600 class of missile has flight duration of 412 seconds with maximum velocity of 2200 m/sec and an apogee of 160 Km. M900 class of missile has flight duration of 500 seconds with maximum velocity of 2750 m/sec and an apogee of 230 Km.

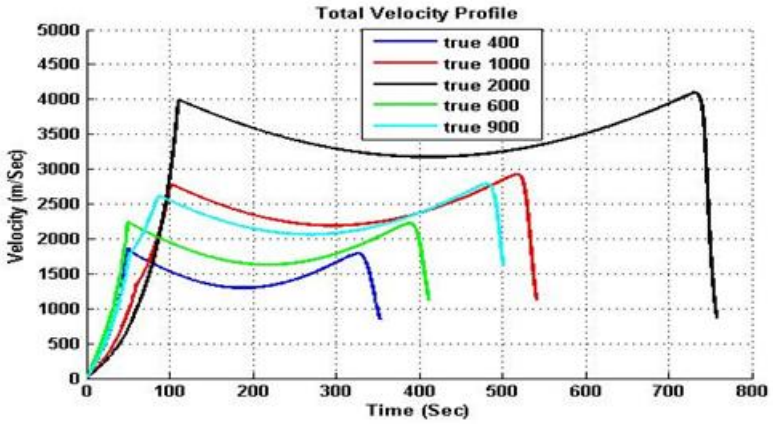

Figure 6.2: Velocity Profile of M400, M600, M900, M1000 & M2000 Class of Missile

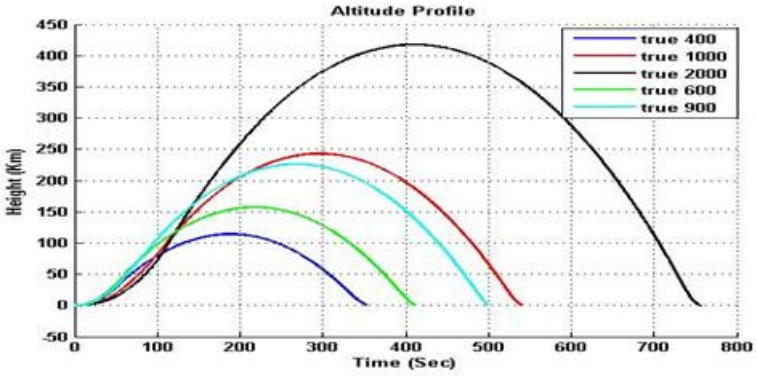

Figure 6.3: Height Profile of M400, M600, M900, M1000 & M2000 Class of Missile

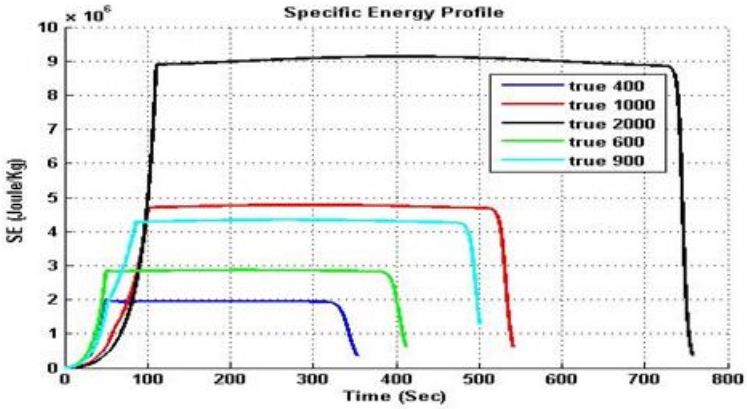

Figure 6.4: SE Profile of M400, M600, M900, M1000 & M2000 Class of Missile

The specific energy (SE) for M400, M1000 and M2000 class of missile increases exponentially during the boost phase to 2 million joules/Kg, 4.8 million joules/Kg and 9 million joules/Kg respectively. M600 class of missile has SE increasing to 2.9 million joules/Kg and M900 has SE increasing to 4.2 million joules/Kg. In descending phase, the specific energy decreases and becomes zero as shown in Figure 6.4. Thus specific energy, altitude and velocity has clear distinction in magnitude for all these cases.

6.2 Performance of HMM

For comparison sake, 5 trajectories of 200 samples each are experimented with. The trajectories that are taken into consideration are for missiles of range 600 Km and 900 Km besides earlier considered class of range 400 Km, 1000 Km and 2000 Km. Figure 6.5 to Figure 6.7 show viterbi distance of 5 types of missiles with results summary of correct and incorrect classification for a total of 1000 samples of trajectory using HMM.

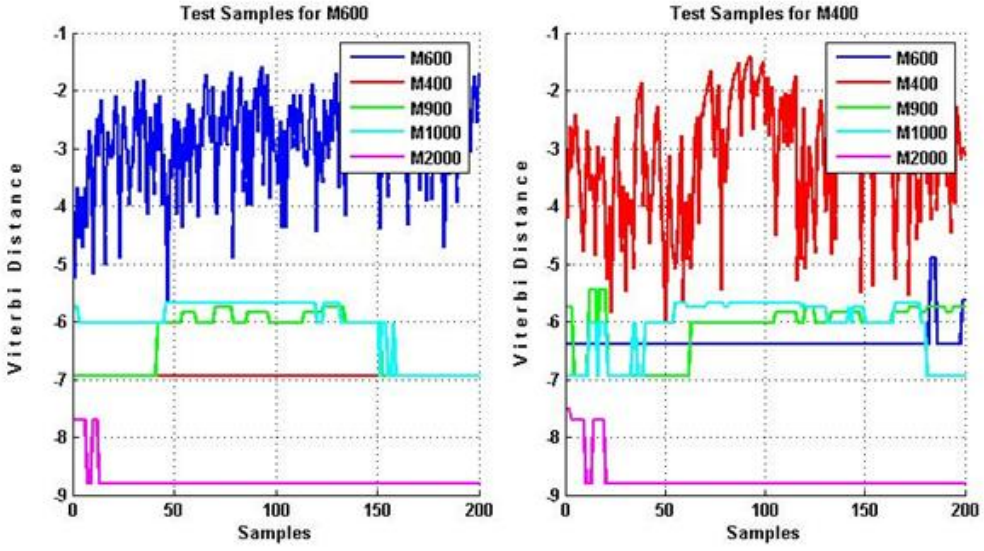

Figure 6.5: Results of HMM a) Test of M600 Class b) Test with M400 Class

Figure 6.5(a) and Figure 6.5(b) show test samples from M600 and M400 class of missile being segregated from samples corresponding to other trajectories respectively. Similarly, Figure 6.6(a), Figure 6.6(b) and Figure 6.7 show test samples from M900,

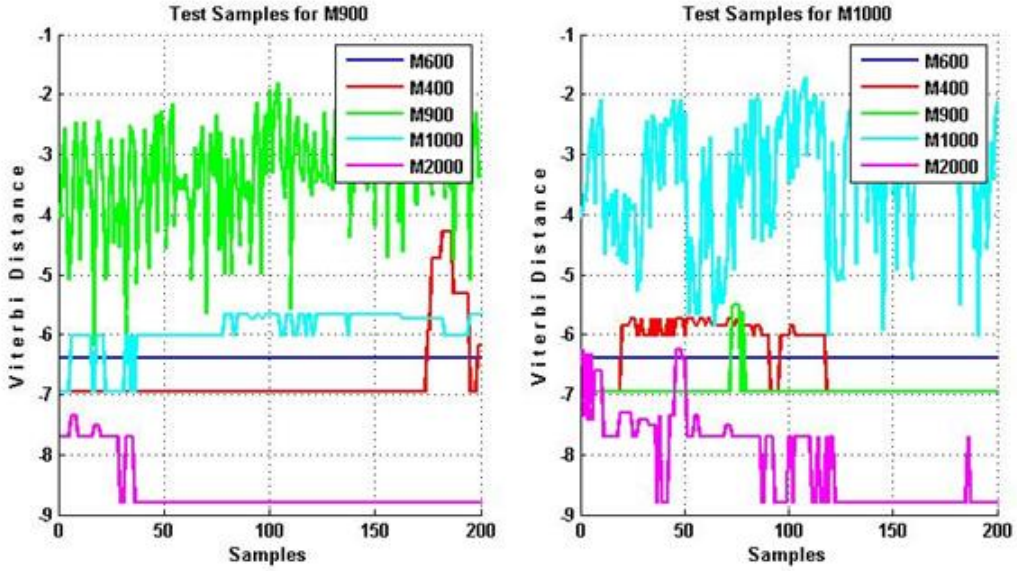

Figure 6.6: Results of HMM a) Test of M900 Class b) Test with M1000 Class

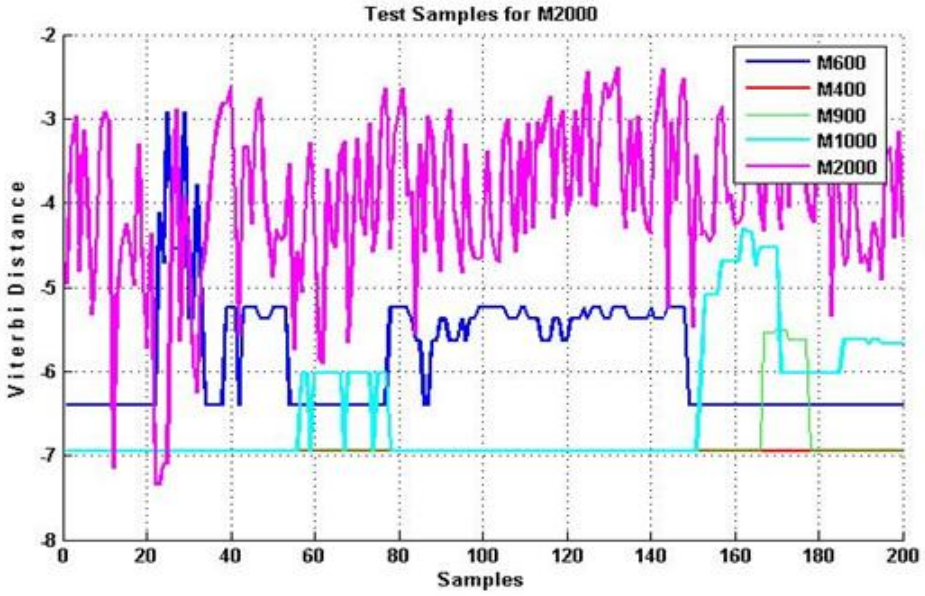

Figure 6.7: Results of HMM Test of M2000 Class

M1000 and M2000 class of missile being segregated from samples corresponding to other trajectories. However, during small duration of M900, M1000 and M2000 test

Table 6.1: Results of HMM Test on 5 types of trajectories

		Actual Class				
		M400	**M600**	**M900**	**M1000**	**M2000**
Predicted class	**Class M400**	**200/ 200**	**0**	**0**	**13**	**20**
	Class M600	**0**	**200/200**	**0**	**0**	**0**
	Class M900	**0**	**0**	**196 / 200**	**0**	**0**
	Class M1000	**0**	**0**	**4**	**187/ 200**	**0**
	Class M2000	**0**	**0**	**0**	**0**	**180 / 200**

cases few samples are misclassified. Correct classification and misclassification samples are shown in confusion matrix of Table 6.1. A total of 37 samples out of 1000 are mis-classified which results into 3.7 percent of mis-classification and 96.3 percent of correct classification. 20 samples of M2000 are misclassified as M400 whereas 13 samples of M1000 are misclassified as M400. Second row shows that 4 samples of M900 are misclassified as M1000.

6.3 Performance of RTNN

Same exercise was done with RTNN, whose results are recorded in confusion matrix of Table 6.2. Table shows that a total of 35 samples are misclassified out of 1000 samples, which forms 3.5% of misclassification, whereas correct classification is 96.5% for the corresponding run. Table 6.2 clearly shows that most of the samples of all categories are classified correctly as shown in diagonal cells. Off-diagonal values shows misclassification. As per the Table 6.2, 18 samples of M2000 class of missile are misclassified as M1000 and 17 samples of M900 are misclassified as M600. Other three classes have no mis-classification.

6.3.1 Weight Adjustment for RTNN

Basic algorithm adapted for real-time classification has to run many times to get best result. If weight corresponding to input layer to first hidden layer is fixed to one value, best result is achieved in very first iteration. For classification of these five classes of

Table 6.2: Results of RTNN Test on 5 types of trajectories

		Actual Class				
		M400	**M600**	**M900**	**M1000**	**M1000**
Predicted class	**Class M400**	**200/ 200**	**0**	**0**	**0**	**0**
	Class M600	**0**	**200 / 200**	**17**	**0**	**0**
	Class M900	**0**	**0**	**183 /200**	**0**	**0**
	Class M1000	**0**	**0**	**0**	**200 /200**	**18**
	Class M2000	**0**	**0**	**0**	**0**	**182/200**

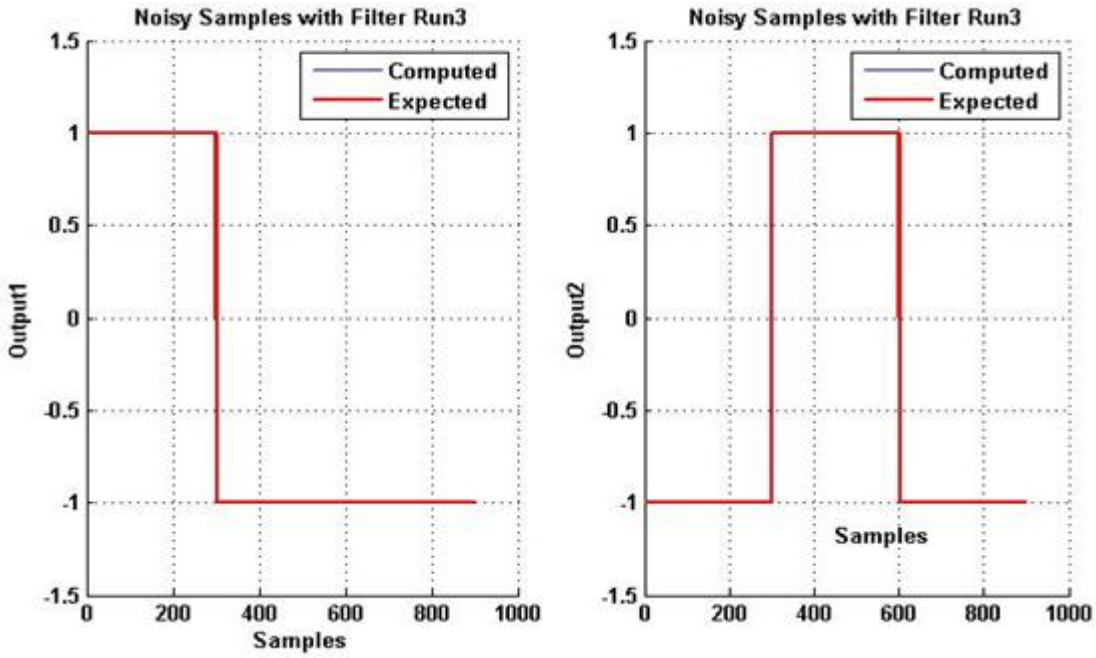

Figure 6.8: Results with adapted weight

missiles, input weight was fixed to 0.5 and classification achieved is 100 percent. The graph of computed output matched with desired output is shown in Figure 6.8. It is clearly visible from Figure 6.8 that with adapted weight, computed result is exactly following desired output with no misclassification. The mean output for the adapted weight is exactly matching the desired values and variance is zero as shown in Table 6.3.

Table 6.3: Mean and Variance of Results for Fixed Weight in Input Layer

Sl No	Class of Target	Expected Values of Node1	Expected Values of Node2	Mean of Node1	Mean of Node2	Variance of Node1 and Node2
1	M2000	-1.0	-1.0	-1	-1	0.00
2	M1000	-1.0	+1.0	-1	+1	0.00
3	M400	+1.0	-1.0	+1	-1	0.00

The reason for this improvement is weighted input values getting naturally grouped into corresponding category due to input vector distance from each other. Random weight assigned for input layer used for RTNN results presented in Section 3.4.5 is resulting in closed data from two different classes being swapped or misclassified during sorting of weighted input.

6.4 Timing

The real time classification of ballistic missiles using RTNN and HMM is developed and 6DOF data is utilized for testing and evaluation. We have established that the RTNN and HMM are able to classify the new sample in less than 50 ms. Both networks performance is around 95% although time taken by HMM is nearly 50 ms whereas RTNN takes less than 0.5 ms on Pentium-4 processor. Small processing time enables to process multi-target scenario with large number of targets being discriminated simultaneously by single computer. There are numerous real-life applications, both in civilian as well as military domains which demands quick response to the order of few ms. Automatic air-traffic classification, tracking and navigation, tracking multiple independently targetable re-entry vehicle (MIRV) and manoeuverable Re-Entry Vehicle (MaRV) and *adaptive packet routing* in communication forms some of the broad applications in general that requires quick response time for classification.

6.5 Summary

HMM network performance is higher than RTNN network with random weights in the input layer. If weights are adapted as fixed in the input layer as indicated in Section 6.3.1 RTNN performance becomes higher than HMM. However, time of testing for HMM is two-order higher than RTNN. HMM takes approximately 49.353 ms for testing of one sample as indicated in Table 4.2 in comparison to 0.69 ms by RTNN as indicated in Table 3.4.

Chapter 7

Conclusions and Future Work

In this work, we have presented a framework for trajectory classification of ballistic missiles using Real time neural network (RTNN) and Hidden Marokov model (HMM). The thesis starts with a brief introduction to the ballistic missile scenario wherein mode of flight by the same missile into *nominal*, *depressed* and *lofted* trajectory is discussed at length. Interception in exo-atmospheric and endo-atmospheric layer, multi-layer concept to increase probability of interception and importance of fast classification to meet the quick response time-line is outlined. First event of the activity in the process of neutralisation of target missile with weapon of mass destruction is the detection by the network of sensors. Thereafter communication to launcher system for selection of correct type of weapon and preparation of interceptor for launch operation is carried out. Measurements by radar system is embedded with inherent noise. The Interactive multiple model (IMM) filters and estimates parameters used for classification. Graphs are plotted to show the difference in trajectory parameters *with* and *without* noise (after application of IMM filter).

Since gathering multiple live trajectories is difficult, mathematical model catering to 6-degree-of freedom (6-DOF) of a moving vehicle is developed to generate large number of trajectories for various classes of missiles for training and testing by the RTNN and HMM method of classification. True trajectories are superimposed with realistic noise to imitate real behaviour of target trajectory and the same IMM is applied as in the case of real trajectory to smoothen the noisy data and generate position, velocity and acceleration of the vehicle in the cartesian coordinate system.

A typical war-scenario presents a complex gamut of flight objects including fighter aircraft, helicopter, UAV, satellite, cruise missile, air-to-air missile, etc. The characteristics of each flight type is presented with respect to kinematic parameters of velocity, acceleration, rate of ascent/descent, radius of turn, g-turn, cluster-id, etc. Response time of interceptor with respect to short-range ballistic missile(SRBM), medium-range ballistic missile(MRBM) and long-range ballistic missile (LRBM) are analysed and presented succinctly to demonstrate the criticality of fast-response classification.

A novel method for classifying ballistic as well as quasi-ballistic missiles using real-time neural network is proposed that classifies target class in real-time. Fast-response is achieved by the network due to analytical computation of network parameter rather than conventional gradient descent method. RTNN computes weight of hidden nodes by equating forward computation from first layer and backward computation using logit function from output layer. Quantizers used in the hidden layer of RTNN allow samples bearing similar characteristics to pass while inputs of different classes are inhibited.

HMM has been applied for trajectory classification based on its successful application in the area of on-line handwriting recognition, speech recognition, gesture recognition, language modelling, motion video analysis and tracking. The trajectory classification is similar to the speech recognition tasks, which is a continuous quantity, that can be described analytically as the position of the object in time. Its discrete representation is used together with its temporal derivations (velocity and acceleration). Thus, an object trajectory is a potentially infinite sequence of state vectors. The trajectory classification problem is formulated to identify the class c_i (i = 1...N) to which belongs the trajectory state sequence. Moving window concept allows both RTNN and HMM to integrate testing and training while taking care of smaller as well as larger trajectories to be addressed by the same network.

Further, we have established that the RTNN and HMM are able to classify the new sample in less than 50 milliseconds. Both networks performance is around 95% although time taken by HMM is near 50 ms whereas RTNN takes less than 0.5 ms on a Pentium-4 processor. Small processing time enables to process multi-target scenario with large number of targets being discriminated simultaneously by a single computer. There are numerous real-life applications, both in civilian as well as military domains

which demands quick response to the order of few ms. Automatic air-traffic classification, tracking and navigation, tracking multiple independently targetable re-entry vehicle (MiRV) and manoeuvrable Re-Entry Vehicle (MaRV), adaptive packet routing in communication form broad applications in general.

7.1 Recommendations and Future Work

While results of 95% are very impressive, distortion penalty associated with vector quantization (VQ) leads to degradations in performance since we are representing an entire region of the vector space by a single space in case of HMM. Continuous HMM can be tried and its performance evaluated against discrete HMM as continuous HMM does not require discretization and therefore quantization effect can be avoided by using it. In the future, study needs to be carried out with trajectories crossing each other in some domain which may occur under certain conditions and tests needs to be carried out to check the suitability of our proposed technique for higher number of trajectories. RTNN and HMM are supervised training methods and the class for initial data has to be known using some other method, which in our case is done through unsupervised training (ART2). Investigations need to be carried out to find other suitable methods for finding the class of initial data.

References

[1] R.S.A RAJA ABDULLAH, M.F. RASID, M.W. AZIS, AND MOHAMED KHALAFALLA. **Target prediction In Forward Scattering Radar**. In *Asia Pacific Conference on Applied Electromagnetics (APACE)*, pages 1–5, 2007. (42, 79)

[2] ALI AKGUL AND SARTUK KARASOY. **Development Of a Tactical Ballistic Missile Trajectory Prediction Tool**. *Electrical & Electronics Engineering*, **5**(2):1463–1467, 2005. (43)

[3] STUART J. ANDERSON. **Target Classification, Recognition and Identification with HF Radar**. In *RTO SET Symposium on "Target Identification and Recognition Using RF Systems"*, pages 25–1–25–19, 2004. (2)

[4] BRUCE BARNETT. *Trajecrtory Equations For A Six Degree Of Freedom Missile*. PN, 1967. (37)

[5] FAISAL I. BASHIR, ASHFAQ A. KHOKHAR, AND DAN SCHONFELD. **Object Trajectory-Based Activity Classification and Recognition using Hidden Markov Models**. *IEEE Transactions on Image Processing*, **16**(7):1912–1919, 2007. (42, 80)

[6] JEFF A. BILMES. **A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models**. *International Computer Science Institute Berkeley*, 1998. (89)

[7] CHRISTOPHER M. BISHOP. **Pattern Recognition and Machine Learning**. *Information Science and Statistics. Springer-Verlag New York, Inc., Secaucus, NJ, USA.*, 2006. (13, 42, 78, 94)

[8] Y. BOGOMOLOV, S. LAPCHEV, E. RIVLIN, M. RUDZSKY, AND G. DROR. **Classification of moving targets based on motion and appearance**. In *British Machine Vision Conference*, pages 429–438, 2003. (42)

[9] LAURENT BORDES AND PIERRE VANDEKERKHOVE. **Statistical inference for partially hidden Markov models**. *IEEE Transactions On Information Theory*, 1991. (78)

[10] GERALD BROWN, MATTHEW CARLYLE, DOUGLAS DIEHL, JEFFREY KLINE, AND KEVIN WOOD. **A Two-Sided Optimization for Theater Ballistic Missile Defense**. *Operations Research*, **53**(5):745–763, 2005. (42)

[11] GAIL A. CARPENTER AND STEPHEN GROSSBERG. **A massively parallel architecture for a self-organizing neural pattern recognition machine**. *Computer Vision, Graphics, and Image Processing*, **37**(1):54–115, 1987. (104)

[12] ASHTON B. CARTER AND DAVID N. SCHWARTZ. *Ballistic Missile Defence*. Brookings Institution Press, 1984. (14, 79)

[13] KISHORE L SUBRATA CHATTERJEE AND T.C. CHANG. **Feature recognition using ART2: a self-organizing neural network**. *Journal of Computer Science*, **8**(3):203–214, 1997. (104)

[14] JOSEPH DITURI. *Ballistic Missile Trajectory Estimation*. Master's thesis, Naval Postgraduate, 2006. (42)

[15] BARY LEONARD (EDITOR). *History of Strategic air and Ballistic Missile Defence*. Diane Publishing, 2010. (14)

[16] DENNIS ERDOGMUS, OSCAR FONTENLA-ROMERO, JOSE C. PRINCIPE, AMPARO ALONSO-BETANZOS, AND ENRIQUE CASTILLO. **Linear-least-squares initialization of multilayer perceptrons through backpropagation of the desired response**. *IEEE Trans. Neural Netw*, **16**(2):325 –337, 2005. (46)

[17] LAURENE V. FAUSETT. *Fundamentals of Neural Networks: Architectures, Algorithms and Applications*. Pearson, 1994. (46, 110)

[18] GERNOT A. FINK. *Markov Models for Pattern Recognition*. Springer, 2008. (79, 87, 91)

[19] ANDREW M FRASER. **Hidden Markov Models and Dynamical Systems**. *Society for Industrial and Applied Mathematics*, 2008. (42, 79)

[20] JESUS GARCA AND OSCAR PREZ CONCHA. **Trajectory classification based on machine-learning techniques over tracking data**. In *9th International conference on Information Fusion*, pages 1–8, 2006. (42, 92)

[21] R.M. GORECKI. **A Baseline 6 Degree of Freedom (DOF) Mathematical Model of a Generic Missile**. *DSTO Systems Sciences Laboratory*, 2003. (41)

[22] ALEX GRAVES, MARCUS LIWICKI, S. FERNANDEZ, ROMAN BERTOLAMI, HORST BUNKE, AND JÜRGEN SCHMIDHUBER. **A Novel Connectionist System for Unconmstrained Handwriting recognition**. *IEEE Transactions on Pattern and Machine Intelligence (PAMI)*, **31**(5):855–868, 2009. (47)

[23] W. J. HARLIN AND DAVID A. CICCI. **Ballistic missile trajectory prediction using a state transition matrix**. *Applied Mathematics and Computation 188 (2007)*, **53**(2):1832 –1847, 2007. (42)

[24] MD. RAFIUL HASSAN, BAIKUNTH NATH, AND MICHAEL KIRLEY. **A fusion model of HMM, ANN and GA for stock market forecasting**. *Expert Systems with Applications*, **33**:171–180, 2007. (79)

[25] PATRICIA A. HAWLEY AND ROSS A. BLAUWKAMP. **Six Degree of Freedom Digital Simulations for Missile Guidance Navigation and Control**. *Johns Hopkins Apl Technical Digest*, **29**(1):71–84, 2010. (37)

[26] CHENG HU, TAO ZENG, AND CHAO ZHOU. **Accurate three-dimensional tracking method in bistatic forward scatter radar**. *URASIP Journal on Advances in Signal Processing*, **66**:1–9, 2013. (2)

[27] GUANG-BIN HUANG. **Learning capability and storage capacity of two-hidden-layer feedforward networks**. *IEEE Transactions on Neural Networks*, **14**(2):274–281, 2003. (14, 42, 46)

[28] GUANG-BIN HUANG AND HAROON A. BABRI. **Upper Bounds on the Number of Hidden Neurons in Feedforward Networks with Arbitrary Bounded Nonlinear Activation Functions**. *IEEE Transactions On Neural Networks*, **9**(1):224–229, 1998. (46)

[29] GUANG-BIN HUANG, QIN-YU ZHU, AND CHEE KHEONG SIEW. **Real-time learning capability of neural networks**. *IEEE Transactions on Neural Networks*, **17**(4):863–878, 2006. (6, 14, 42, 46, 47, 53)

[30] JEFFREY A. ISSACSON AND DAVID R. VAUGHAN. *Estimation and Prediction of Ballistic Missile Trajectories*. RAND, 1996. (42)

[31] B. H. JUANG AND LAWRENCE R. RABINER. **Hidden Markov Models for Speech Recognition**. *Technometrics*, **33**(3):251–272, 1991. (78)

[32] AMLAN KUNDU AND GEORGE C. CHEN. **An Integrated Hybrid Neural Network And Hidden Markov Model Classifier For Sonar Signal Classification**. *0-7803-2431-5/95 IEEE*, 1995. (79)

[33] R. J. KUO, J. L. LIAO, AND C. TU. **Integration of ART2 neural network and genetic K-means algorithm for analyzing Web browsing paths in electronic commerce**. *Decision Support Systems*, **40**(2):355–374, 2005. (104)

[34] YICONG LI, THIAGALINGAM KIRUBARAJAN, YAAKOV BAR-SHALOM, AND MURALI YEDDANAPUDI. **Trajectory and Launch Point Estimation for Ballistic Missiles from Boost Phase LOS Measurements**. In *Proceedings of the 7th Mediterranean Conference on Control and Automation (MED99)*, pages 930–952, 1999. (43)

[35] ALAN J. LIPTON, HIRONOBU FUJIYOSHI, AND RAJU S. PATIL. **Moving target classification and tracking from real-time video**. In *WACV*, pages 8–14, 1998. (42)

[36] DENGSHENG LU AND QIHAO WENG. **A survey of image classification methods and techniques for improving classification performance**. *International journal of Remote Sensing*, **28**(5):823–870, 2007. (45)

[37] PETER J MANTLE. *The Missile Defense Equation-Factors for decision making*. American Institute of Aeronautics and Astronautics Inc., 2003. (1, 11, 29, 31)

[38] KISHORE MEHROTRA AND PRAVAS R. MAHAPATRA. **A Jerk Model for Tracking Highly Maneuvering Targets**. *IEEE Transactions on Aerospace and Electronics and Systems*, **33**(4):1094–1105, 1997. (14, 17)

[39] DONALD MICHIE, DAVID SPIEGELHALTER, AND CHARLES TAYLOR. *Machine Learning, Neural and Statistical Classification*. Overseas Press, 2009. (46)

[40] JOZEF MLCH AND PETR CHMELA. **Trajectory classification based on Hidden Markov Models**. *Proceeding of 18 th International Conference on Computer Graphics Vision Moscow*, pages 101–105, 2008. (42, 80)

[41] JACINTO C. NASCIMENTO, JORGE S. MARQUES, AND MRIO A.T. FIGUEIREDO. **Trajectory Classification Using Switched Dynamical Hidden Markov Models**. *IEEE Transactions On Image Processing*, **19**(5):1338–1348, 2010. (42)

[42] JACINTO C. NASCIMENTO, JORGE S.MARQUES, AND MARIO A.T. FIGUEIREDO. **Classification Of Complex Pedestrian Activities From Trajectories**. *Proceedings of 2010 IEEE 17th International Conference on Image Processing*, pages 3481–3484, 2010. (42)

[43] DUY H. NGUYEN, JOHN H. KAY, BRADLEY J. ORCHARD, AND ROBERT H. WHITING. **Classification and Tracking of Moving Ground Vehicles**. *LINCOLN LABORATORY Journal*, **13**(2):275–308, 2002. (42, 43)

[44] KAI NI, YUTING QI, AND LAWRENCE CARIN. **Multi-Aspect Target Detection via the Infinite Hidden Markov Model**. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2007, Hawaii, USA, April 15-20, 2007*, pages 433–436, 2007. (79)

[45] JUSTIN D. PAOLA AND ROBERT A. SCHOWENGERDT. **A Review and Analysis of Backpropagation Neural Networks for Classification of Remotely Sensed Multispectral Imagery**. *International Journal of Remote Sensing*, **16**(16):3033–3058, 1995. (45)

[46] SANGHYUK PARK, JAEHYEON JEONG, CHANG-KYUNG RYOO, AND KEEYOUNG CHOI. **Detection and Classification of a Ballistic Missile in Ascent Phase**. *11th International Conference on Control, Automation and Systems*, pages 26–29, 2011. (9, 43)

[47] ALAIN PERES, FREDERIC PERRIN, AND ERIC SAUTY. **Method for determining the trajectory of a ballistic missile**. `http://www.google.com/patents/US20110246069/`, 2011. US Patent App. 13/015,956. (42)

[48] D. PFEIFFER AND R. REULKE. **Trajectory-based scene description and classification by analytical functions**. In *Object Extraction for 3D City Models, Road Databases and Traffic Monitoring -Concepts, Algorithms and Evaluation-CMRT-09*, pages 41–46, 2009. (42)

[49] LAWRENCE R RABINER. **A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition**. *Proceedings of the IEEE*, **77**(2):257–286, 1989. (42, 78, 79, 93, 94, 95)

[50] INTERNAL REPORT. **Analysis of classification using Radar Data**. *10-11-154*, 2009. (41)

[51] AJOUDJ RDA AND BOUKELIF AOUED. **Artificial Neural Network-Based Face Recognition**. In *First International Symposium on Control, Communications and Signal Processing*, pages 439–442, 2004. (14)

[52] TOBIAS SCHEFFER AND STEFAN WROBEL. **Active Learning of Partially Hidden Markov Model**. *Theory to Applications, Springer*, 2001. (79)

[53] OGAWA SHINICHI. **Missile Defense and Deterrence**. Technical report, National Institute of Defence Studies, 2002. (13)

[54] GEOFFREY L. SILBERMAN. **Parametric classification techniques for theater ballistic missile defense**. *Johns Hopkins Apl Technical digest*, **19**(3):322–339, 1998. (1, 12, 42)

[55] PETER TAIT. *Introduction to Radar Target Recognition*. The Institution of Engineering and Technology, 2005. (2)

[56] SHIN'ICHI TAMURA AND MASHIKO TATEISHI. **Capabilities of a Four-layered Feed-forward Neural network: Four layers versus three**. *IEEE transactions on Neural Networks*, **8**(2):251–255, 1997. (14, 42)

[57] JACQUELINE K. TELFORD. **Sensitivity Analysis Using Design of Experiments in Ballistic Missile Defense**. In *In Proceedings of U.S. Army conference on applied statistics*, 2000. (42)

[58] G. TURHAN-SAYAN. **Applications Of Artificial Neural Networks And Genetic Algorithms To Electromagnetic Target Classification**. *RTO SCI Symposium on "The application of Information technologies to Mission Systems", held in Monterey, California, USA*, **23**, 1998. (1)

[59] V. VAIDEHI, N. CHITRA, M. CHOKKALINGAM, AND C.N. KRISHNAN. **Neural network aided Kalman filtering for multitarget tracking applications**. *Computers and Electrical Engineering*, **27**:217–228, 2001. (14)

[60] PHILIP D. WASSERMAN. *Neural Computing: Theory and Practice*. Van Nostrand Reinhold, 1989. (46)

[61] BRIAN L. WEAVER. *A Methodology For Ballistic Missile Defense Systems Analysis Using Nested Neural Networks*. PhD thesis, Georgia Institute of Technology, 2008. (42)

[62] STEPHEN D. WEINER AND SOL M. ROCKLIN. **Discrimination Performance Requirements for Ballistic Missile Defense**. *The Lincoln Laboratory*, **7**:63–87, 1994. (32)

[63] DEAN A. WILKENING. **A Simple Model for Calculating Ballistic Missile Defense Effectiveness**. *Science & Global Security*, **8**(2):183–215, 1999. (32, 42)

[64] KYOUNG-JAE WON, ADAM PRGEL-BENNETT, AND ANDERS KROGH. **Training HMM Structure with Genetic Algorithm for Biological Sequence Analysis**. *Bioinformatics*, **20**:3613–3619, 2004. (105)

[65] JIA XIN, WU ZUOLONG, AND GUAN HSIN. **The Target Vehicle Movement State Estimation Method with Radar Based on Kalman Filtering Algorithm**. In *Proceedings of the 2nd International Symposium on Computer, Communication, Control and Automation (ISCCCA-13)*, pages 342–345, 2013. (14)

[66] JUNGI YAMATO, JUN OHYA, AND KENICHIRO ISHII. **Recognizing human action in time-sequential images using hidden Markov model**. *Proc. of the Conf. on Computer Vision and Pattern Recognition, Champaign, IL*, pages 379–385, 1992. (45)

[67] ANTHONY ZAKNICH. **Neural Networks for Intelligent Processing**. *World Scientific Publishing*, **4**, 2003. (46)

[68] GUOQIANG PETER ZHANG. **Neural Networks for Classification: A Survey**. *IEEE Transactions of Systems, Man, And Cybernetics-PART C: Applications and Reviews*, **30**(4):451–462, 2000. (42, 79)

List of Publications

[69] UPENDRA KUMAR SINGH, VINEET PADMANABHAN, ARUN AGARWAL. **Dynamic classification of ballistic missiles using neural networks and hidden Markov models**. *Journal of applied soft computing*, volume 19, pages 280-289, 2014.

[70] UPENDRA KUMAR SINGH, VINEET PADMANABHAN, ARUN AGARWAL. **A novel method for training and classification of ballistic and quasi-ballistic missiles in real-time** . *International joint conference on neural networks*, IJCNN-2013, pages 1-8, Texas USA, August 4-9, 2013.

[71] UPENDRA KUMAR SINGH, VINEET PADMANABHAN. **Training by ART-2 and classification of ballistic missiles using hidden Markov models**. *5th international conference on pattern recognition and machine intelligence*, PREMI-2013, pages 108-114, ISI Kolkata, India, December 10-14, 2013.

[72] UPENDRA KUMAR SINGH, VINEET PADMANABHAN. **Training and classification of ballistic missiles using hidden Markov models**. *6th international conference on contemporary computing*, IC3-2013, pages 301-306, Noida, India, August 8-10, 2013.

[73] UPENDRA KUMAR SINGH, VINEET PADMANABHAN. **Classification of aerial targets using real-time neural network for defense applications**. *WASET-2011*, pages 1832-1840, Paris, France, June 24-26, 2011.