

10-12-2012

Field Programmable Gate Array Based Target Detection and Gesture Recognition

Priyanka Mekala

Florida International University, pmeka001@fiu.edu

DOI: 10.25148/etd.FI12110703

Follow this and additional works at: <https://digitalcommons.fiu.edu/etd>

Recommended Citation

Mekala, Priyanka, "Field Programmable Gate Array Based Target Detection and Gesture Recognition" (2012). *FIU Electronic Theses and Dissertations*. 723.

<https://digitalcommons.fiu.edu/etd/723>

This work is brought to you for free and open access by the University Graduate School at FIU Digital Commons. It has been accepted for inclusion in FIU Electronic Theses and Dissertations by an authorized administrator of FIU Digital Commons. For more information, please contact dcc@fiu.edu.

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

FIELD PROGRAMMABLE GATE ARRAY BASED TARGET DETECTION AND
GESTURE RECOGNITION

A dissertation submitted in partial fulfillment of

the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

ELECTRICAL ENGINEERING

by

Priyanka Mekala

2012

To: Dean Amir Mirmiran
College of Engineering and Computing

This dissertation, written by Priyanka Mekala, and entitled Field Programmable Gate Array Based Target Detection and Gesture Recognition, having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this dissertation and recommend that it be approved.

Armando Barreto

Hai Deng

Deng Pan

Jeffrey Fan, Major Professor

Date of Defense: October 12, 2012

The dissertation of Priyanka Mekala is approved.

Dean Amir Mirmiran
College of Engineering and Computing

Dean Lakshmi N. Reddi
University Graduate School

Florida International University, 2012

© Copyright 2012 by Priyanka Mekala

All rights reserved.

DEDICATION

I dedicate this dissertation to my family, particularly my mother, Chandrakala Mekala, father, Dharma Reddy Mekala, and brother, Praveen Reddy Mekala. Without their love, support, and understanding in all endeavors the completion of this work would not have been possible. I want to let them know that I consider myself lucky to have them as my family and thank them from the bottom of my heart.

ACKNOWLEDGMENTS

I wish to first thank my parents; they made me feel loved, and showed me that under no circumstance I should feel alone or left behind. Their encouragement, support and guidance throughout my doctoral study and beyond have meant the world to me. I would also like to thank God for everything, especially for family, friends, the opportunities given, and the opportunities that lie ahead. Thank you for giving me the strength, courage, and determination to finish this Ph.D., I am truly grateful.

I would also like to thank especially my advisor Dr. Jeffrey Fan for inspiration, commitment, and faith he has shown me throughout my graduate career. His unconditional help means a lot to me, and most of my accomplishments have become a reality because of his guidance and support. Thank you for being a great mentor and also a great philosopher. I wish to express my deepest gratitude to him for his excellent guidance, caring, patience, and providing me with an excellent atmosphere for doing research.

I wish to thank all my committee members, Dr. Armando Barreto, Dr. Hai Deng and Dr. Deng Pan for their support throughout the dissertation process. I highly appreciate their advice. I would also like to thank members of the VLSI Laboratory, especially Dr. Wei Zhao and Mr. Xinwei Niu for all their help.

Finally, I am grateful to Maria Benincasa, Pat Brammer, Ana Saenz and Oscar Silveria who were always willing to collaborate with everything they could.

ABSTRACT OF THE DISSERTATION
FIELD PROGRAMMABLE GATE ARRAY BASED TARGET DETECTION AND
GESTURE RECOGNITION

by

Priyanka Mekala

Florida International University, 2012

Miami, Florida

Professor Jeffrey Fan, Major Professor

The move from Standard Definition (SD) to High Definition (HD) represents a six times increase in data, which needs to be processed. With expanding resolutions and evolving compression, there is a need for high performance with flexible architectures to allow for quick upgrade ability. The technology advances in image display resolutions, advanced compression techniques, and video intelligence. Software implementation of these systems can attain accuracy with tradeoffs among processing performance (to achieve specified frame rates, working on large image data sets), power and cost constraints. There is a need for new architectures to be in pace with the fast innovations in video and imaging. It contains dedicated hardware implementation of the pixel and frame rate processes on Field Programmable Gate Array (FPGA) to achieve the real-time performance.

The following outlines the contributions of the dissertation. (1) We develop a target detection system by applying a novel running average mean threshold (RAMT) approach to globalize the threshold required for background subtraction. This approach adapts the threshold automatically to different environments (indoor and outdoor) and

different targets (humans and vehicles). For low power consumption and better performance, we design the complete system on FPGA. (2) We introduce a safe distance factor and develop an algorithm for occlusion occurrence detection during target tracking. A novel mean-threshold is calculated by motion-position analysis. (3) A new strategy for gesture recognition is developed using Combinational Neural Networks (CNN) based on a tree structure. Analysis of the method is done on American Sign Language (ASL) gestures. We introduce novel point of interests approach to reduce the feature vector size and gradient threshold approach for accurate classification. (4) We design a gesture recognition system using a hardware/ software co-simulation neural network for high speed and low memory storage requirements provided by the FPGA. We develop an innovative maximum distant algorithm which uses only 0.39% of the image as the feature vector to train and test the system design. Database set gestures involved in different applications may vary. Therefore, it is highly essential to keep the feature vector as low as possible while maintaining the same accuracy and performance.

TABLE OF CONTENTS

CHAPTER	PAGE
I INTRODUCTION.....	1
1.1 Motivation.....	1
1.1.1 Trends in Video and Image Processing.....	2
1.1.2 System Architectures.....	3
1.2 Research Purpose and Difficulties.....	5
1.2.1 Target Detection and Tracking.....	5
1.2.2 Pattern Recognition.....	6
1.3 Significance of this Research.....	6
1.4 Structure of Dissertation.....	7
1.5 Chapter Summary.....	8
II BACKGROUND.....	9
2.1 Surveillance.....	9
2.1.1 General Information on surveillance.....	9
2.1.2 Motion and Tracking.....	10
2.2 General Pattern Recognition System.....	11
2.2.1 Neural Networks for Recognition.....	12
2.2.2 Network Topology.....	13
2.2.3 Back Propagation (BP) Algorithm.....	15
2.3 American Sign Language.....	18
2.3.1 Previous Research.....	20
2.3.2 Signing.....	20
2.3.3 Concerns and Issues affecting Sign language.....	21
2.4 Hardware Description Language and Field Programmable Gate Array.....	22
2.4.1 HDL.....	23
2.4.2 FPGA.....	25
2.5 Chapter Summary.....	26
III TRACKING IN SURVEILLANCE SYSTEMS.....	28
3.1 Tracking in Surveillance systems.....	28
3.1.1 Preprocessor filtering.....	29
3.1.2 Background subtraction analysis.....	30
3.1.3 State attributes estimation.....	32
3.2 Occlusion Detection.....	38
3.2.1 Motion of Objects.....	40
3.2.2 Two-Dimensional linear motion.....	42

3.2.3	Two-Dimensional non-linear motion.....	42
3.3	Calculation of Threshold.....	44
3.4	Algorithm flow control.....	47
3.5	Experimental Results.....	48
3.5.1	Linear Motion Analysis	48
3.5.2	Non-Linear motion analysis.....	51
3.6	Chapter Summary.....	55
IV	A GLOBAL APPROACH FOR TARGET DETECTION.....	56
4.1	DSP vs. FPGA.....	58
4.2	Target Detection Algorithm	61
4.2.1	Image Acquisition and Frame Generation	61
4.2.2	Grayscale conversion.....	62
4.2.3	Median Filtering for noise reduction	62
4.2.4	Disparity.....	65
4.2.5	Adaptive Threshold: A Global Approach.....	65
4.2.6	Target detection	67
4.3	Designed System Architecture using MicroBlaze processor	68
4.3.1	Development Tools.....	69
4.4	Output Display using SSH	72
4.5	Experimental Results.....	73
4.6	Chapter Summary.....	78
V	AUTOMATIC PATTERN RECOGNITION.....	79
5.1	Pattern Recognition System Architecture	80
Preprocessing module design		81
5.2	Designed Combinational Neural Network Architecture	82
5.2.1	Dataflow.....	83
5.2.2	Network Topology.....	84
5.3	Feature Extraction Logistics.....	85
5.3.1	Sign/ Gesture – SM Rule	85
5.4	Experimental Results.....	88
5.5	Chapter Summary.....	91
VI	GESTURE RECOGNITION ON HW/SW CO-SIMULATION PLATFORM	93
6.1	Hardware/Software Co-simulation.....	93
6.2	Introduction	93
6.3	Benefits of FPGA.....	96
6.4	Gesture Recognition Algorithm	99
6.4.1	Database Generation	99

6.4.2	Text file Generation	100
6.4.3	Memory Controller on Altera ModelSim	101
6.5	Co-simulation Neural Network	102
6.5.1	Training and Testing	104
6.5.2	Gesture Recognition.....	106
6.6	Device Utilization Factor	106
6.7	Simulation Output	108
6.8	Experimental Results.....	112
6.9	Chapter Summary.....	116
VII	CONCLUSION AND FUTURE WORK.....	118
7.1	Concluding Remarks	118
7.1.1	Motion and Occlusion Detection in Surveillance	118
7.1.2	A Global approach toward Target Detection	119
7.1.3	Pattern Recognition using Combinational Neural Networks.....	119
7.1.4	HW/SW co-simulation of Gesture Recognition	119
7.2	Future Work	120
7.3	Chapter Summary.....	121
REFERENCES	122
VITA	131

LIST OF TABLES

TABLE	PAGE
Table II-1 HDL vs. Procedural languages [13].....	23
Table IV-1 DSP vs. FPGA.....	59
Table IV-2 Median filter code	63
Table IV-3 Success of Target detection algorithm applied to different environments.....	77
Table IV-4 Device Utilization Summary	77
Table IV-5 Timing Summary.....	77
Table V-1 Motion Vector Direction	87
Table V-2 Noise Immunity of Signs.....	90
Table VI-1 Comparison of SW vs. HW/SW simulation platforms	115

LIST OF FIGURES

FIGURE	PAGE
Figure I-1 Conventional Video Processing block diagram.....	1
Figure I-2 System Architecture.....	3
Figure I-3 Proposed changes in Video processing units.....	7
Figure II-1 Video surveillance systems – space monitoring, shopping centers, government organizations, airports and traffic monitoring [10].....	10
Figure II-2 Neural Network Model – Input to neurons and synaptic weights modeled as a linear function and the Activation threshold modeled as a non-linear function [13]	14
Figure II-3 Back Propagation (BP) network topology [22].....	15
Figure II-4 American sign language alphabets [24]	19
Figure II-5 Implementation of a logic design with the FPGA	26
Figure III-1 Tracking in surveillance system.....	28
Figure III-2 Scene inconsistency involved- illumination and brightness	29
Figure III-3 Background subtraction analysis.....	30
Figure III-4 Position and Size attributes	33
Figure III-5 Kalman filter tracking stages	34
Figure III-6 Kalman filter for tracking – Prediction and correction stages [52].....	35
Figure III-7 Object tracking of a video sequence of 20 seconds	36
Figure III-8 Kalman filter estimate of Non-linear motion.....	37
Figure III-9 Kalman filter estimate of multiple objects - Non-linear motion.....	37
Figure III-10 Velocity estimate in object tracking	38
Figure III-11 Aircraft trajectories in space [10].....	39

Figure III-12 Multiple camera scene view	40
Figure III-13 Tracking of four objects using camera 1 [10]	41
Figure III-14 Tracking of four objects using camera 2 [10]	41
Figure III-15 Non-linear motion assumed to be sampled as linear motion – state space continuity relation is assumed.....	43
Figure III-16 An object in a bounding sphere.....	44
Figure III-17 Threshold calculation [10]	45
Figure III-18 Occlusion occurrence detection algorithm.....	47
Figure III-19 Three objects of motion with linearity	49
Figure III-20 Three objects of motion with linearity- Plot with the distance between two objects	50
Figure III-21 Three objects of motion with linearity - Plot with distances and different values of threshold	51
Figure III-22 Three objects of motion with non- linearity.....	52
Figure III-23 Three objects of motion with non- linearity- Plot with the distance between two objects	53
Figure III-244 Three objects of motion with non- linearity - plot with distances and different value of thresholds found based on the similar sizes and velocity of the objects	53
Figure III-255 Three objects of motion with non- linearity - plot with distances and different value of thresholds found different sizes and velocities of the objects.....	54
Figure IV-1 Xilinx FPGA EDK system design [65].	56
Figure IV-2 Block diagram of FPGA based target detection using adaptive threshold segmentation.	57
Figure IV-3 DSP processor vs. FPGA.....	60
Figure IV-4 Frame generation and Header file creation.	61

Figure IV-5 Xilinx ISE and SDK environment [63].	68
Figure IV-6 A view of MicroBlaze system.	70
Figure IV-7 Block Diagram of Target Detection.	71
Figure IV-8 Visual basic SSH remote shell ActiveX control for communication.	72
Figure IV-9 Case of Human in an Outdoor environment scenario.	74
Figure IV-10 Case of Human in an Outdoor environment scenario.	74
Figure IV-11 Case of the current frame with car as target for Indoor environment with various illumination levels.	75
Figure IV-12 Case of Human in Traffic signal in Outdoor Environment.	75
Figure IV-13 Case of Car in Traffic signal in Outdoor Environment.	76
Figure IV-14 Case of Human in an Indoor environment scenario.	76
Figure V-1 Flow diagram of Automatic Pattern Recognition [11].	79
Figure V-2 Image after background subtraction and hand tracking.	82
Figure V-3 Parallelism between Feature extraction and CNN tracking [11].	83
Figure V-4 Feature vector of n elements [11].	83
Figure V-5 CNN network layers	84
Figure V-6 Point of Interest I (Each fingertip of the hand) and II (midpoint of the palm) – POIs [12].	85
Figure V-7 Angle chart – Motion Vector related to direction [12].	86
Figure V-8 Gradient threshold algorithm of an input image- 40 features of feature vector	88
Figure V-9 MATLAB results of the input image and letter recognized	89
Figure V-10 Mean square error	90

Figure VII-1 System architecture of Gesture recognition on hardware/ software co-simulation platform.....	96
Figure VI-2 HDLDAEMON as the interface between MATLAB and ALTERA MODELSIM.	100
Figure VI-3 Simulation output waveform of Memory design controller.	101
Figure VI-4 Neural Network Model [18, 20].....	103
Figure VI-5 Log-sigmoid transfer function [22].....	104
Figure VI-6 Feature Extraction Algorithm for distance values.	105
Figure VI-7 Device Utilization Summary Report for memory storage.....	107
Figure VI-8 Device Utilization Summary Report for gesture recognition system test design module.....	107
Figure VI-9 RTL schematic of Design process flow.....	108
Figure VI-10 Simulation result in Xilinx ISE indicating the memory stored.....	109
Figure VI-11 Simulation output for Database set-1.....	110
Figure VI-12 Simulation output for Database set-2.....	110
Figure VI-13 Simulation output for Database set-3.....	111
Figure VI-14 Simulation output for Database set-4.....	111
Figure VI-15 Performance curve for Gesture recognition using SW (MATLAB software) analysis with reduced database sets for training.....	112
Figure VI-16 Performance curve for Gesture recognition using HW/SW co-simulation analysis.....	113
Figure VI-17 Performance curve for Gesture recognition using SW (MATLAB) simulation analysis.....	114
Figure VI-18 System design test output simulation for image gesture alphabet “P”.	115
Figure VI-19 Output sign recognition displayed as a grid matrix.	116

SYMBOLS AND ABBREVIATIONS

SYMBOLS	DEFINITION
α	Learning Rate
μ	Mean
σ	Standard Deviation
2D	Two Dimensional
3D	Three Dimensional
ASL	American Sign Language
ANN	Artificial Neural Network
APR	Automatic Pattern Recognition
BP	Back Propagation
CLB	Configurable Logic Block
CoG	Centre of Gravity
CPU	Central Processing Unit
EDK	Embedded Development Kit
FFT	Fast Fourier Transform
FPGA	Field Programmable Gate Array
FSM	Finite State Machine
FV	Feature Vector
GHz	Gigahertz
HDL	Hardware Descriptive Language
HMM	Hidden Markov Model
ISE	Integrated Software Environment

LED	Light Emitting Diode
LCD	Liquid Crystal Display
LoG	Laplacian of Gaussian
LUT	Look up Table
MAC	Multiply and Accumulate
MATLAB	Matrix Laboratory
MB	MicroBlaze
MV	Motion Vector
MB	Macro Block
MDS	Motion Detection System
MoG	Mixture of Gaussian
MSE	Mean Square Error
NN	Neural Network
PACE	Pin out and Area Constraints Editor
PC	Personal Computer
PDF	Probability Density Function
POI	Point of Interest
RAM	Random Access Memory
RTL	Register Transfer Logic
SLR	Sign Language Recognition
SoC	System-on-a Chip
SRAM	Static Random Access Memory
SSH	Secure Shell

TS	Tracking System
TSS	Tracking in Surveillance System
UART	Universal Asynchronous Receiver/Transmitter
UCF	User Constraint File
USB	Universal Serial Bus
VHDL	VHSIC Hardware Descriptive Language
VHSIC	Very High Speed Integrated Circuit
VGA	Video Graphics Array
XPS	Xilinx Platform Studio

I INTRODUCTION

1.1 Motivation

With the arrival of today's highly-integrated multimedia devices and fast emerging applications, video and image processing have become more important than ever. Many new and exciting innovations, such as HDTV, controller free gaming, digital cinema (3D and 4D), revolve around video and image processing [1]. These devices require complex video/image processing tasks leading to a very challenging design process; as it demands more efficient and high processing systems.

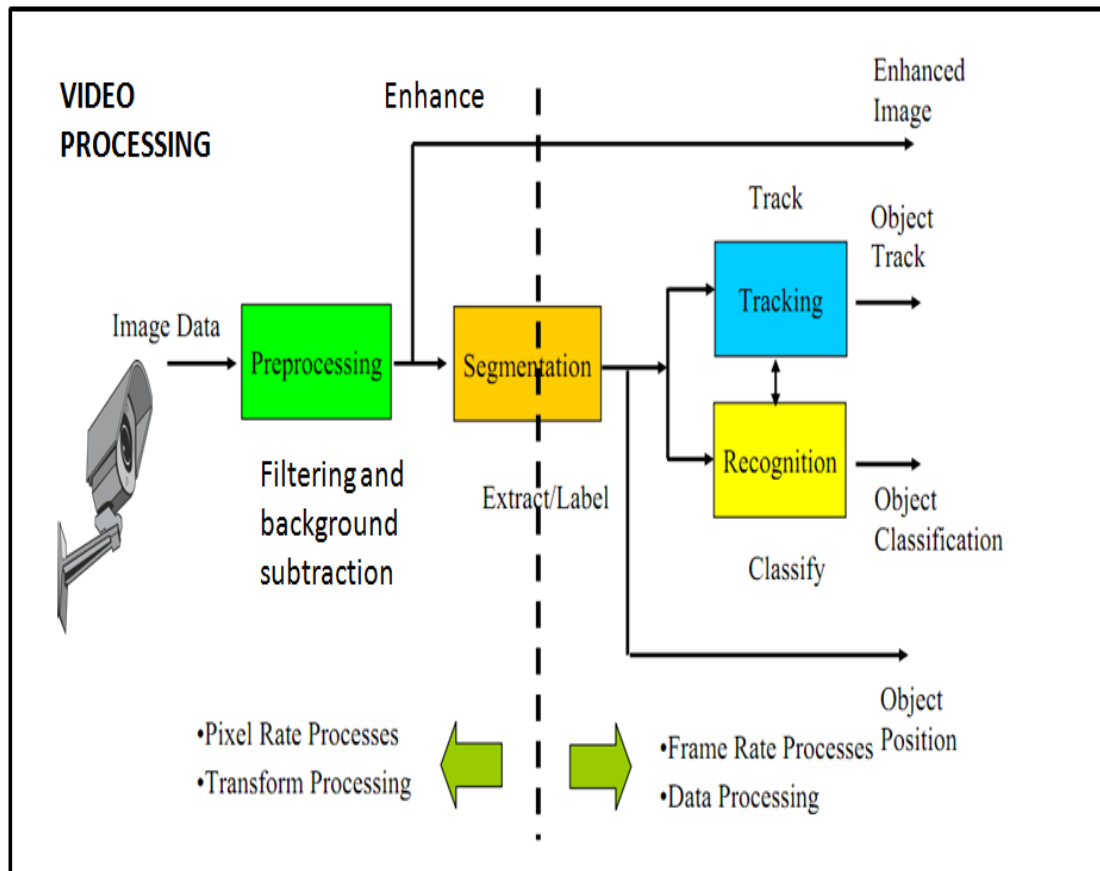


Figure I-1 Conventional Video Processing block diagram

A conventional video processing block diagram is represented in Figure I-1. It is generally divided into two parts. The first contains pre-processing and segmentation, which are pixel rate processes. The second contains tracking and recognition, which are frame rate processes [2].

1.1.1 Trends in Video and Image Processing

The move from standard definition (SD) to high definition (HD) represents a six times increase in data, which needs to be processed. Video surveillance is also moving from Common Intermediate Format (CIF) (352 x 288) to D1 format (704 x 576) as a standard requirement, with some industrial cameras even moving to HD at 1280 x 720. Military surveillance, medical imaging, and machine vision applications are also moving to very high resolution images [1]. With expanding resolutions and evolving compression, there is a need for high performance with flexible architectures to allow for quick upgrade ability.

Another rapidly evolving area is video intelligence. Cameras are being installed everywhere from a hand held device to super computers. The hard disk storage is efficiently used by involving motion detection algorithms to archive video frames where a motion threshold is passed. The video object recognition would allow for automated surveillance monitoring, which is much more effective than manual surveillance monitoring. With the technology advances in image capture and display resolutions, advanced compression techniques, and video intelligence ; software implementation of these systems can attain accuracy with tradeoffs among processing performance (to

achieve specified frame rates, working on large image data sets), power and cost constraints. New architectures as in Figure I-2 are to be developed to be in pace with the fast innovations in video and imaging. It contains dedicated hardware implementation of the pixel and frame rate processes to achieve the real-time performance.

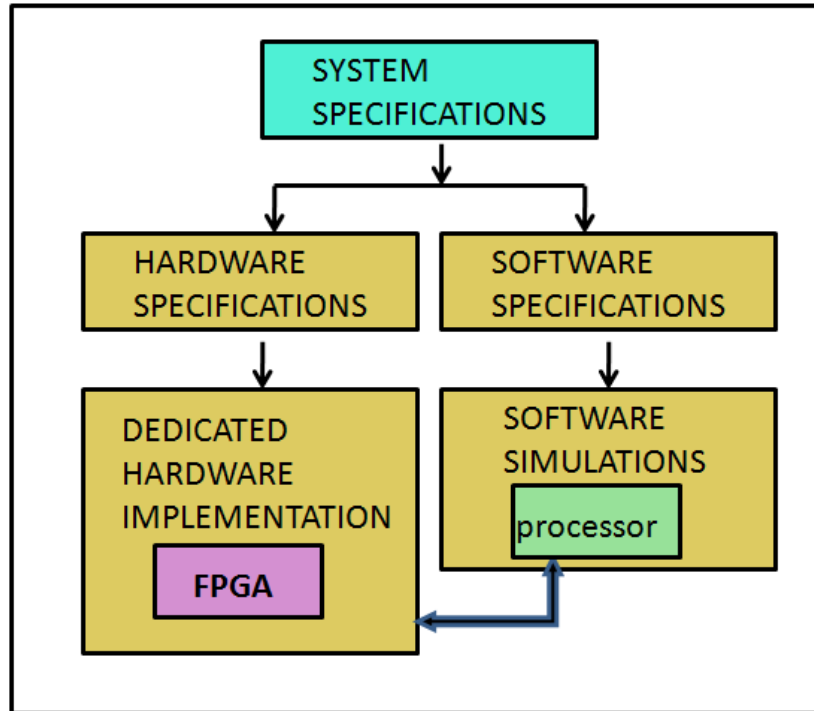


Figure I-2 System Architecture

1.1.2 System Architectures

The choice of system architecture includes standard cell Application Specific Integrated Circuits (ASICs), programmable solutions such as Digital Signal Processing (DSP) or media processors and Field Programmable gate Arrays (FPGAs). The approach is chosen depending on application requirements and solution availability. The ideal architecture should possess the following characteristics:

- High Performance

Performance not only applies to compression, but also pre- and post processing functions such as scaling, filtering, de-interlacing, and color space conversion. Processor-only architectures cannot meet the performance requirements with a single device. A state-of-the-art DSP running at 1 GHz cannot perform H.264 HD decoding or H.264 HD encoding, which is about ten times more complex than decoding. FPGAs (with or without external DSP processor) are the only re-programmable best solutions.

- Low Development Cost

In general the costs for masks and wafer, software, design verification, and layout, development of a typical 90-nm standard-cell ASIC is very high (approx. US\$30 million). Only the FPGAs can justify such high development costs.

- Flexibility and Easy Upgradeability

Architectures must be flexible and easy to upgrade with rapidly evolving technology.

- Migration Path to Lower Unit Costs

It is important to have a solution with a low-cost migration path as standards stabilize and volumes increase. Most silicon companies focus on video and imaging target applications such as video camcorders, set-top boxes, digital still cameras, cell phones and other portable products, or LCD TVs and monitors. Therefore, when designing a lower-volume type of application, it is best to consider FPGA [1, 2].

1.2 Research Purpose and Difficulties

The academic and industrial researchers have recently been focusing on analyzing images. While researchers are making progress, the problem is difficult and many existing algorithms are complex, slow and unreliable. The algorithms that run near real-time work on computers those are very expensive relative to the existing hand-held interface devices. The low power FPGA platforms are increasingly being used in most emerging technology applications. It provides the unique coupling of high- performance and flexibility to tackle many of today's most challenging tasks through exploiting parallelism (complementary co-processing functions) and hardware reconfiguration.

1.2.1 Target Detection and Tracking

In surveillance, the two most specific tasks of interest are motion detection and occlusion detection for tracking. These tasks require an efficient image processing system. Tracking is important in human-computer interaction, security and surveillance, video communication and compression, traffic control and medical imaging [3-6]. Presently there are various systems that can serve the purpose of tracking but one major constraint imposed is occlusion. The tracking is lost when occlusion occurs and results in loss of target. An intelligent video surveillance system is one that can detect the occurrence of occlusion and prevent the loss of track of the object. Solutions proposed are based mostly on multiple camera inputs [7-9]. When multiple cameras are involved, there is an increase in the bandwidth and memory requirement to store the data, extra power and time consumption to process that data. Thus focus on occlusion detection is

required for more accurate tracking with reduced resource constraints. Hence new system architecture called motion-position analysis is designed which meet the system constraints and also provide some prior information regarding the occlusion in tracking.

1.2.2 Pattern Recognition

Recently, there has been a surge in interest in recognizing patterns. Pattern recognition has various applications like animation, computer games, machinery control, and surveillance. One of the most structured sets of patterns belongs to sign language. In sign language, each gesture has an assigned meaning. Computer recognition of these patterns may provide a more natural human-computer interface, allowing people to point, or rotate a model by rotating their hands. Interactive applications pose particular challenges. The response time should be very fast as these applications are usually supposed to be implemented in real-time. They require very high processing speed in nanoseconds which are not possible by the software platform simulations. Hence we develop a new strategy for recognition system and enhance the speed of the design by implementing on a hardware/software co-simulation platform. Though the processing is achievable on serial processor, it can be beneficial to take advantage of the parallelism, low cost and low power consumption offered by FPGAs.

1.3 Significance of this Research

Hardware based systems save the video/image processing time and contributes to the reliability of the system by reducing extra work, like interfacing and scheduling of the software. Therefore, the hardware based system is attractive for video/image processing

systems which require huge data processing and high resolution. The deployment on to the FPGA improves the processing time and saves bandwidth. This dissertation explores the use of FPGAs in the area of video processing. FPGAs provide best solution for video and image processing applications, such as broadcast infrastructure, medical imaging, HD videoconferencing, video surveillance and military imaging.

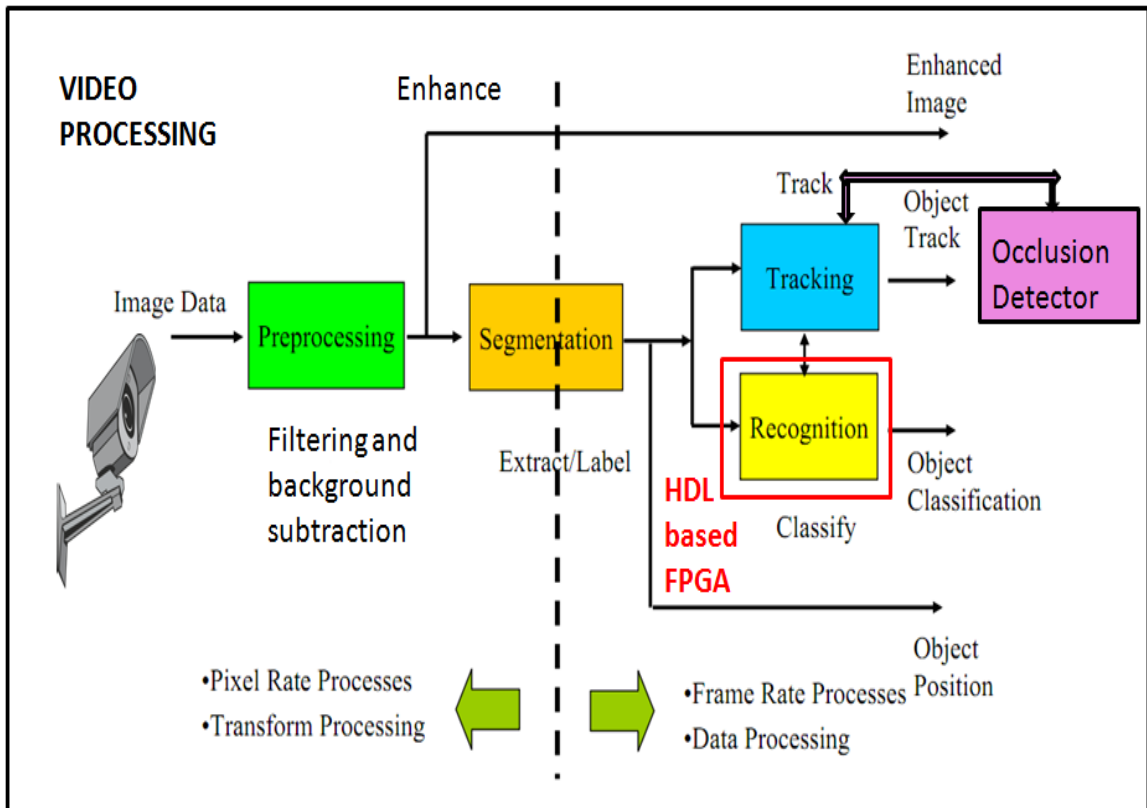


Figure I-3 Developed system design of Video processing units

1.4 Structure of Dissertation

The dissertation is divided into six stages. First, background knowledge and basic concepts are studied and established. This includes various imaging techniques involved in surveillance and pattern recognition systems. The following five chapters are based on

research descriptions of the four key aspects of the design enhancement of imaging techniques and are the focus of this dissertation: (1) Occlusion detection using motion-position analysis; (2) Global approach for target detection using adaptive threshold on FPGA (3) Gesture recognition using CNN; and (4) High speed Hardware/Software co-simulation platform for gesture recognition. Finally some conclusions are made based on the methods, algorithms and architectures designed. Also future research is being suggested to increase the scope of the dissertation.

1.5 Chapter Summary

Chapter 1 has given the motivation, research purpose, and difficulties associated with video and imaging applications. The current and future trends are indicated with a focus on the need for flexible system architectures. These efforts are specifically applied to surveillance (motion and occlusion detection) and gesture recognition systems. The characteristics of FPGA to provide the best solutions are listed with a light on need in video surveillance and gesture recognition. The structure of the dissertation is also discussed. This entire dissertation is based on research published in [10], [11], [12], and [13].

II BACKGROUND

Technology advances every year at a very high rate. Computer vision is one field which has seen increasing number of applications recently in various domains like surveillance systems, interactive systems like gesture recognition, animation and gaming, biomedical imaging, etc. To provide a real-time solution to the above domains one best consideration is to shift the applications on to a hardware platform. To grasp the advantage of the work developed, some background on the applications involving imaging techniques is described.

2.1 Surveillance

2.1.1 General Information on surveillance

Video surveillance systems are incorporated everywhere in airports, government organizations, traffic monitoring, industrial plants, shopping centers for observing an area or a property as in Figure II-1. Though digitized, these systems consume large memory if the complete recorded data has to be stored. The memory consumption can be decreased if only the data with useful information is stored.

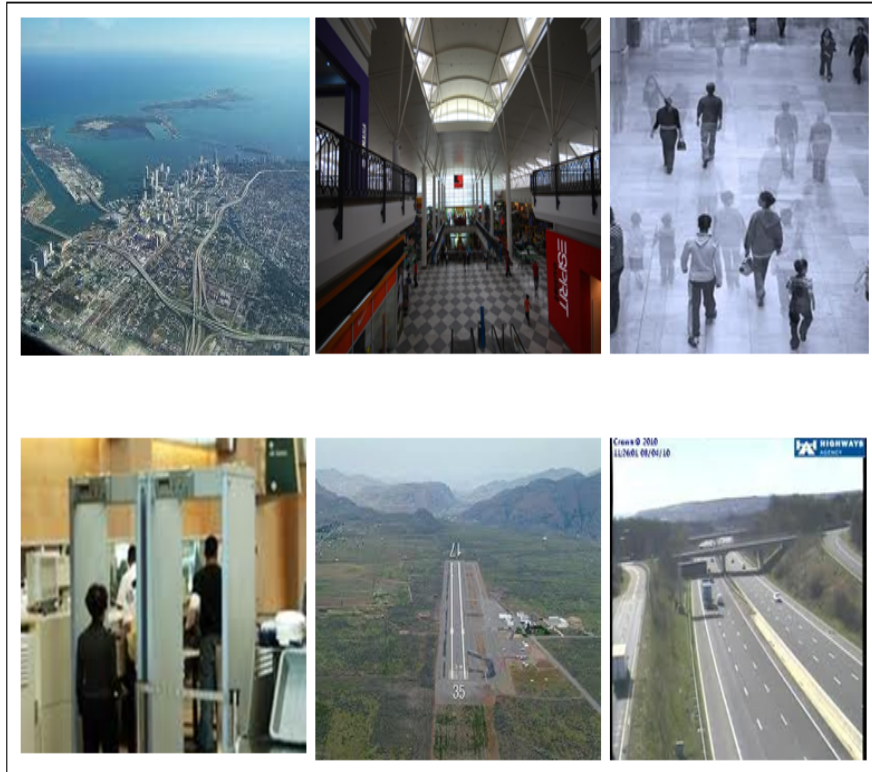


Figure II-1 Video surveillance systems – space monitoring, shopping centers, government organizations, airports and traffic monitoring [10]

2.1.2 Motion and Tracking

Video is a set of images or frames comprised of background and foreground. The useful information is referred as the foreground since it contains the object of interest's motion. Motion detection is the process of locating pixels that change their values of Motion Vectors (MVs) from the image frames. It could be single object or multiple moving objects with respect to time [14]. The moving targets within the video frame can be identified by a motion detection algorithm [15].

Tracking systems fall under the category of surveillance where not only the targets have to be identified but also tracked. Occlusion is a complex problem, which can cause

the loss of the target in the tracking process. Many efficient methods have been found to solve the occlusion problem [16-19]. Optimizing multiple object tracking and best-view video synthesis is essential for many multimedia applications, such as surveillance, smart rooms, sports analysis and video presentation enhancement. One solution for the occlusion problem is to detect the occlusion before it occurs or partially occurs and to track the moving objects using an additional camera separated by a known distance and known camera parameters, which can give a better view. Thus, the loss of a target can be prevented. Algorithms to perform target tracking under occlusion using motion-position analysis are used by calculating the shortest distances among the targets using a safe distance factor.

2.2 General Pattern Recognition System

Pattern recognition is a powerful technique for harnessing the information in the data and generalizing about it. Recognition systems though have huge range of classifications, possess one basic structure. The module level architecture includes:

- Image/Video Acquisition Unit – The input to the system is the high data content (raw) images/video obtained using a video camera which produces stream of RGB pixels.
- Preprocessing Unit – Raw video has to be processed to satisfy the memory requirements and the environmental scene conditions. These consist of illumination, background variations, camera parameters, and other scene complexities.

- Feature Extraction Unit – Specific attributes of the target considered important in describing and recognizing (called features) are required to increase the performance of the recognition engine. They vary from simple and basic geometrical description to precise functional information.
- Recognition Network Unit– Many architectures exist to build a recognition unit. Choosing a network that is fault tolerant, real-time operative and adaptive in nature forms the underlying aspects. Based on the features obtained an accurate and high speed processed output is to be delivered.
- Output display Unit – The output of the recognition engine.

2.2.1 Neural Networks for Recognition

Neural networks are based on the parallel architecture of brain neurons. It can be defined as a multiprocessor system network with high degree of interconnection and adaptive interaction between elements. A neural network image processor can free imaging applications from various constraints in terms of video acquisition, lighting and hardware settings. This degree of freedom is possible because a neural network allows you to build a network by learning examples. The more examples are learned, the more expert the network and sometimes it is quite easy to automate the learning. The tradeoff is shifted from the cost of equipment to a number of images necessary to train and build a robust engine. The choice of neural networks to recognize the object automatically is due to the following aspects as in [20]:

- Adaptive learning: Networks are trained using a set of data predefined in the database.

- Self-organization: Any addition or deletion of objects from the task space (training) is possible with minimum effort. Such modifications do not result in an exhaustive training of the system again.
- Real time operation: Computations may be carried out in parallel, such as using System-on-a-Chip (SoC) design.
- Fault tolerance: Partial destruction of a network does not lead to complete degradation of performance

The main disadvantage of using neural networks for pattern recognition is high time to train a model from a very complex data set. Neural techniques are computer intensive and are generally slow on low end PCs or machines. Though the system takes longer to train the overall time to results can still be faster than other data analysis approaches. Also neural networks do not require the time programming and debugging or testing assumptions that other analytical approaches do [21]. Hence the choice of neural networks is made for the pattern recognition system.

2.2.2 Network Topology

To model an artificial neuron from a biological neuron, three basic components are used - input to the neurons, synaptic weights and activation threshold function. The Figure II-2 shows the neural network engine [22]. Mathematically they can be considered as functions- two linear and one non-linear. The synapses of the biological neuron (i.e. the one which interconnects the neural network and gives the strength of the connection) are modeled as synaptic weights. All inputs are modified by the weights and summed

altogether. This activity is referred as a linear combination. The result is then passed through a non-linear activation threshold to determine the output. The activation function considered could be – step function (simplest non-linear function), ramp function or a sigmoid function.

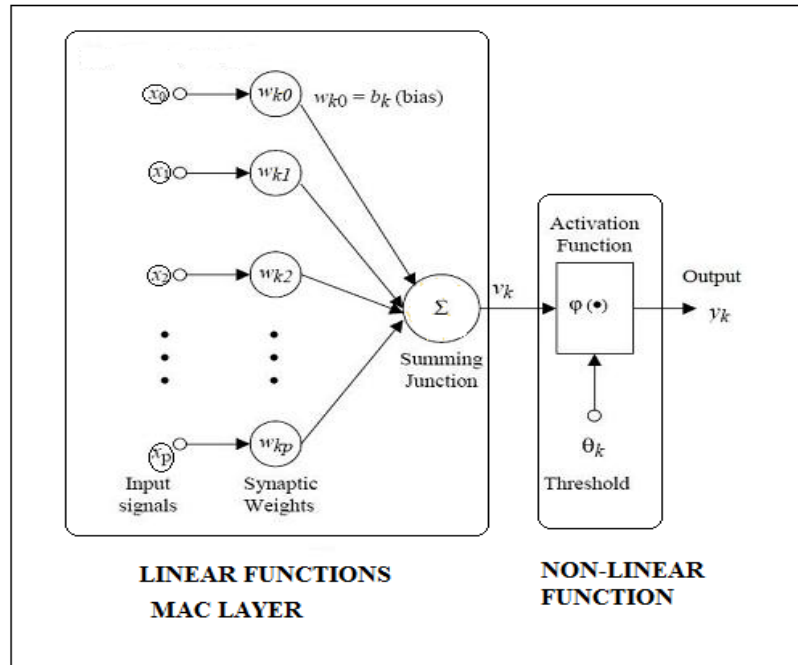


Figure II-2 Neural Network Model – Input to neurons and synaptic weights modeled as a linear function and the Activation threshold modeled as a non-linear function [13]

Each neuron receives several inputs i.e. x_i and generates pre-output v_k (k representing the neuron generating output) through the linear function given in Equation II-1.

$$v_k = \sum_{i=0}^p x_i w_{ki} \quad \text{Equation II-2}$$

The synaptic weight of the connection is given by w_{ki} ; where ‘ p ’ is the number of incoming inputs to the neuron. The output of the model is given by y_k given by the pre-output passed through the activation function $\varphi(\cdot)$ in Equation II-2.

$$y_k = \varphi(v_k) \quad \text{Equation II-2}$$

When building recognition systems with complex input output relations, sigmoid function is the best option for activation function since the nonlinearity makes the learning powerful, differential is possible and easy with simple equations and negative and positive value makes learning fast.

2.2.3 Back Propagation (BP) Algorithm

A feed-forward network has a layered structure shown in Figure II-3. Each layer consists of units which receive their input from units from a layer directly below and send their output to units in a layer directly above the unit. There are no connections within a layer.

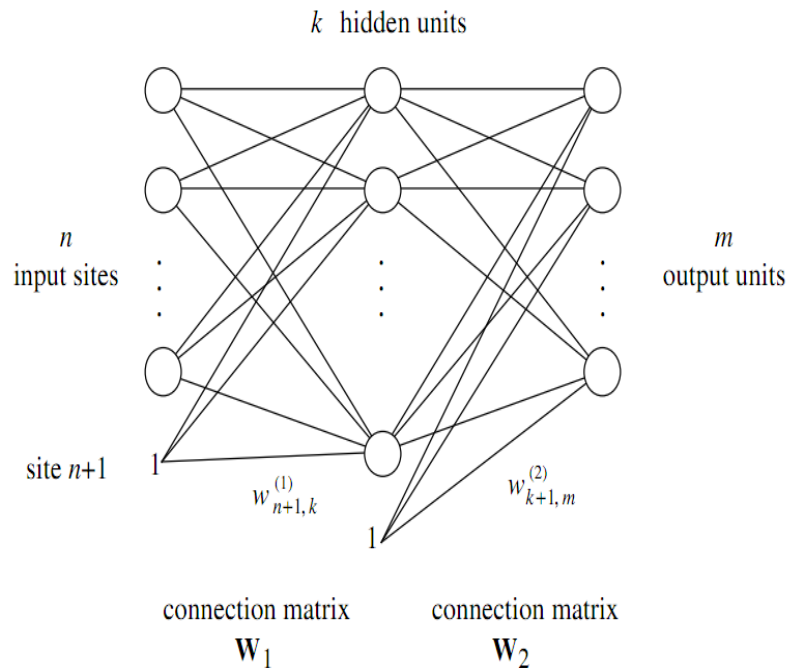


Figure II-3 Back Propagation (BP) network topology [22]

Algorithm [22]:

Step 1: Initialize weights to small random values in the range [-0.5 0.5].

Step 2: While stopping condition is false, do steps 3-10

Step 3: For each training pair do steps 4-9.

Feed Forward:

Step 4: Each input unit ($x_i; i=1, 2, \dots, n$) receives input signal and broadcasts this signal to all units in the layer above the hidden units.

Step 5: Each hidden unit ($z_j; j=1, 2, \dots, p$) sums its weighted input signals following Equation II-3.

$$z_in_j = v_{0j} + \sum_{i=1}^n x_i v_{ij} \quad \text{Equation II-3}$$

Then applies activation function $f(\cdot)$ to compute its output signal as in Equation II-4, and sends this signal to all units in the layer above (output units).

$$z_i = f(z_in_j) \quad \text{Equation II-4}$$

Step 6: Each output unit ($y_k; k=1, 2, \dots, m$) sums its weighted input signals, and then applies its activation function to compute the output signal following Equation II-5 and Equation II-6.

$$y_in_k = w_{0k} + \sum_{j=1}^p z_j w_{jk} \quad \text{Equation II-5}$$

$$y_k = f(y_in_k) \quad \text{Equation II-6}$$

Back Propagation of Error:

Step 7: Each output unit (y_k , $k=1,2,\dots,m$) receives a target pattern corresponding to the input training pattern, computes its error information term using Equation II-7. Calculates its weight correction term (used to update w_{jk} later) as given in Equation II-8.

$$\delta_k = (t_k - y_k) f'(y_{in_k}) \quad \text{Equation II-7}$$

$$\Delta w_{jk} = \alpha \delta_k z_j \quad \text{Equation II-8}$$

Calculates its bias correction term (used to update w_{0k} later), $\Delta w_{0k} = \alpha \delta_k$ and sends δ_k to units in the layer below.

Step 8: Each hidden unit (z_j , $j=1,2,\dots,p$) sums its delta inputs (from units in the layer above) using Equation II-9.

$$\delta_{in_j} = \sum_{k=1}^m \delta_k w_{jk} \quad \text{Equation II-9}$$

Error information is calculated by multiplying with the derivative of its activation function given by Equation II-10, weight correction term (used to update v_{ij} later) calculate using Equation II-11, bias correction term calculated (used to update v_{0j} later) using Equation II-12.

$$\delta_j = \delta_{in_j} f'(z_{in_j}) \quad \text{Equation II-10}$$

$$\Delta v_{ij} = \alpha \delta_j x_i \quad \text{Equation II-11}$$

$$\Delta v_{0j} = \alpha \delta_j \quad \text{Equation II-12}$$

Update weights and biases:

Step 9: Each output unit (y_k , $k=1,2,\dots,m$) updates its bias and weights ($j=1,2,\dots,p$) using Equation II-13 and each hidden unit (z_j , $j=1,2,\dots,p$) updates its bias and weights ($i=1,2,\dots,n$) using Equation II-14;

$$w_{jk}(new) = w_{jk}(old) + \Delta w_{jk} \quad \text{Equation II-13}$$

$$v_{ij}(new) = v_{ij}(old) + \Delta v_{ij} \quad \text{Equation II-14}$$

Step 10: Test the stopping condition.

2.3 American Sign Language

Since ages communication has served as a medium to build relationships, know people, understand technology and allow rapid growth and development on a global basis. Normal people can communicate their thoughts and ideas to others through speech. One important means of communication method for the hearing impaired community is the use of sign language, as in [23]. 500,000 and 2,000,000 people use Sign Language as their major daily communication tool. These numbers may deviate from other different sources but it is surprisingly popular as mentioned in Trudy Suggs book: American Sign Language is the 3rd most-used language in the United States. It seems that 3.68% of the total population is found to be hard of hearing and 0.3% of the total population is functionally Deaf, out of a total population of about 268,000,000 (2005) in the US.

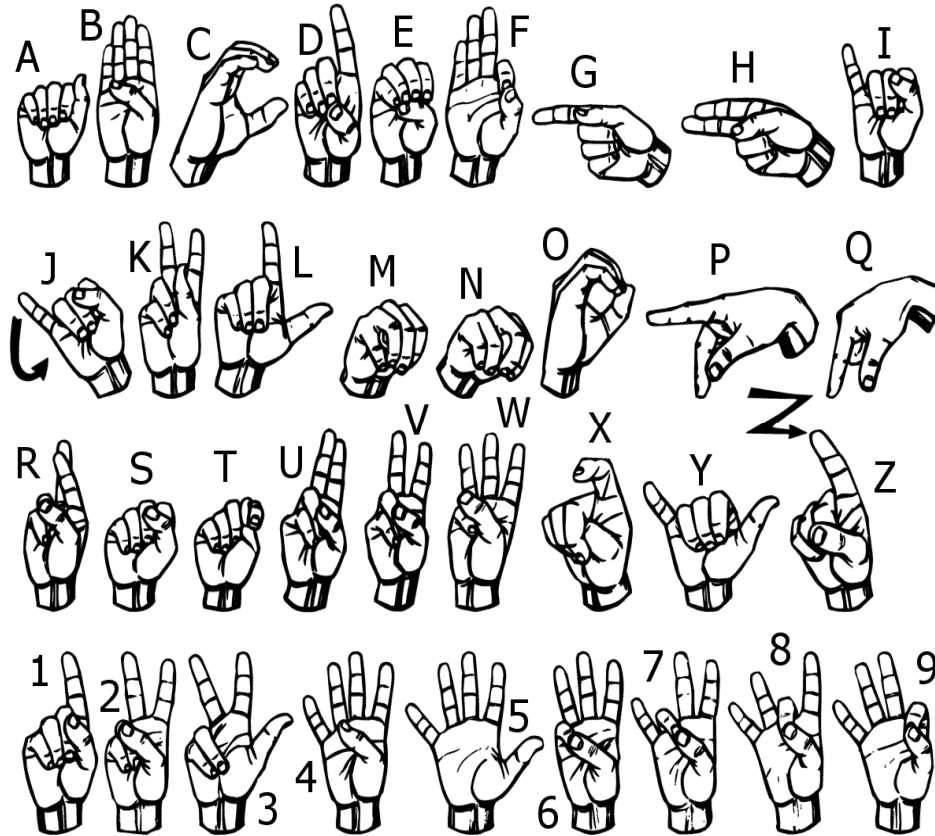


Figure II-4 American sign language alphabets [24]

In Canada and the U.S.A., American Sign Language (ASL) is generally preferred as the vehicle of communication for the hard of hearing and the deaf alike. However, ASL is one of the many sign languages of the world. ASL also has its own grammar that is different from English. ASL consists of approximately 6000 gestures of common words with finger spelling used to communicate obscure words or proper nouns [21]. Finger spelling uses one hand and 26 gestures to communicate the 26 letters of the alphabet. Some of the signs can be seen in Figure II-4 below.

2.3.1 Previous Research

Several methods have been proposed in the past to translate the signs using the gestures and features of the signer. Primarily, Ko and Yang developed a finger mouse that enables a user to specify commands with the fingers as in [26]. Other novel approaches include the colored glove based method, skin color segmentation, video sequence appearance modeling and Hidden Markov Model (HMM) systems as in [27]-[29]. Research on hand gestures can be classified into three categories. The first category, glove based analysis, employs sensors attached to a glove that converts finger flexions into electrical signals for determining the hand posture. The second category, vision based analysis (by creating the three-dimensional model of the human hand). The model is matched to images of the hand by one or more cameras, and parameters corresponding to palm orientation and joint angles are estimated. It is probably the most difficult to implement in a satisfactory way. The third category, analysis of drawing gestures, usually involves the use of a stylus as an input device. Analysis of drawing gestures can also lead to recognition of written text [30] – [35]. It only requires video-based data collection and hence leads to a better natural interface for the user.

2.3.2 Signing

Signing takes place in a 3D space, called signing space close to the trunk and the head, as in [36]. Signs are either one-handed or two-handed. For one-handed signs the so called dominant hand performs the sign, whereas for two handed signs the second hand, the non-dominant hand, is also needed. Sign language when compared to the spoken

language has different grammar. In spoken language the speech is group of sentences where words in the sentence are linear, one word followed by another, whereas in sign language, a simultaneous structure exists with a parallel temporal and spatial configuration.

2.3.3 Concerns and Issues affecting Sign language

A video image acquisition process is subjected to many environmental concerns such as the position of the video camera, environmental conditions like lighting sensitivity, background condition and number of cameras used.

- Occlusion plays a crucial role factor in real time as while signing; some fingers or even a whole hand can be occluded, as in [10].
- Sign boundaries have to be detected automatically. The start and end of a sign are required to be detected automatically from the video sequence sets captured.
- A sign is affected by the preceding and the subsequent sign (co articulation).
- The position of the signer in front of the camera may vary. This results to unwanted temporal and spatial change of the co-ordinate axis under consideration. Movements of the signer, like shifting in one direction or rotating around the body axis, must be considered.
- Each sign varies in time and space. The signing speed differs significantly. Even if one person performs the same sign twice, small changes of speed and position of the hands will occur.

- The projection of the 3D scene on a 2D plane results in loss of depth information. The reconstruction of the 3D-trajectory of the hand in space is not always possible, as in [37].
- The processing of a large amount of image data is time consuming, so real-time recognition is difficult.
- Higher resolution causes considerable delay in the execution of the acquisition process and longer processing time.
- Real-time processing: The translator is sufficiently fast to capture images of signer, process the images and display the sign translation on the computer screen.

SLR applications are mostly in real-time. With the use of software solutions to recognize the signs there is always a delay involved due to the time-consuming computations. Hence the need of a hardware model arises to satisfy the real-time scenario.

2.4 Hardware Description Language and Field Programmable Gate Array

The current trends involve conventional processor being replaced by the Field programmable gate array (FPGA) systems due to their high performance when processing large amount of data. The goal is to suggest an accelerated design using the hardware description language (HDL) on FPGA. The challenge is to design efficient, effective, and reliable system model with the highest possible reliability.

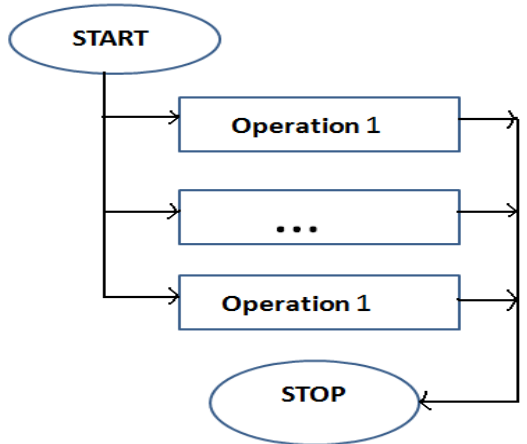
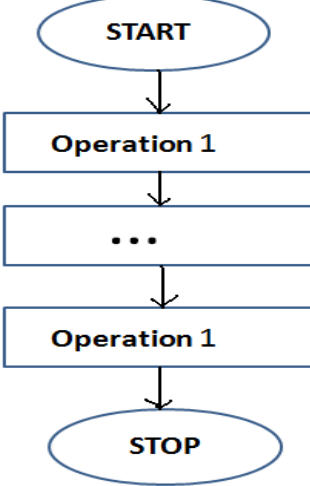
2.4.1 HDL

HDL is an acronym which stands for Hardware Description Language. VHDL and VERILOG are the most commonly used HDL languages. HDL has been at the heart of electronic design productivity since initial ratification by the IEEE in 1987. For almost 15 years the electronic design automation industry has expanded the use of HDL from initial concept of design documentation to design implementation and functional verification. It can be said that HDL fueled modern synthesis technology and enabled the development of ASIC semiconductor companies. The HDL is used to describe hardware from the abstract to the concrete level. HDL usage risen rapidly since its inception and is used around the globe to create sophisticated electronic products.

It provides a formal description of the hardware using specific description style for different abstraction levels of the design. On the most basic level, any block to be designed contains the ENTITY and the ARCHITECTURE. The ENTITY declaration is much like a declaration of a function in C++. The ARCHITECTURE statement is like the actual function in C++, it describes the logic behind the entity. Table II-1 shows the comparison of HDL with the procedural languages and outlines the advantages of characterizing digital hardware using hardware description language based on entity connections, concurrent operations, propagation delay and timing information [38]-[41].

Table II-1 HDL vs. Procedural languages [13]

HDL (Hardware descriptive)	Procedural languages (C, C++ , MATLAB)
HDL contains components that are concurrent i.e. run in parallel/ simultaneously.	Traditional software languages like C, C++, and MATLAB are sequential.

 <p style="text-align: center;">CONCURRENT</p>	 <p style="text-align: center;">SEQUENTIAL</p>
<p>HDL provides ways to describe propagation of time and signal dependencies. Hardware oriented – Digital logic design (The operations and structure are described in gate level and RT level – hierarchal design).</p>	<p>No way to describe time and signal dependency. Software oriented – Binary executable (Data flow language and non-hierarchal design)</p>
<p>HDL supports constructs that are useful in writing high-level models, test benches and other non-hardware artifacts needed in hardware logic design.</p>	<p>Explicit constructs and assignments are not supported by the procedural languages.</p>
<p>HDL has static type checking - many errors found before synthesis and/or simulation.</p>	<p>Errors can be analyzed only after debugging. Synthesis errors are hard to debug.</p>
<p>HDL has a rich collection of data types and well-defined standard with a full-featured language and module system (libraries and packages).</p>	<p>Object oriented programs are written with pure logical or algorithmic thinking. Inherently procedural (single-threaded), with limited syntactical and semantic support to handle concurrency</p>

2.4.2 FPGA

FPGA provide a potential alternative to speed up the hardware realization of computationally intensive algorithms. It involves low cost, higher density and shorter design cycle [42]-[45]. The ease of programmability and simplicity allows the user to easily realize the design in hardware. It provides a better solution for real-time processing applications due to dedicated hardware rather than software solutions. Implementation of a logic design with FPGA involves the following steps shown in Figure II-5.

- Description of the logic circuit using a HDL (such as VHDL/ VERILOG) or design using a schematic editor.
- Logic synthesizer to transform the HDL or schematics into a net list. The net list contains the description of the various logic gates of the design with the interconnections.
- Implementation tools to map the logic gates and interconnections into the FPGA. The FPGA consists of many configurable logic blocks, which can be further decomposed into look-up tables that perform logic operations. The CLBs and LUTs are interwoven with various routing resources. The mapping tool collects the net list gates into groups that fit the LUTs. The place & route tool assigns the groups to specific CLBs while opening or closing the switches in the routing matrices to form the connections.
- Generate programming file extracts the state of the switches in the routing matrices and generates a bit stream where the ones and zeroes correspond to open or closed switches.

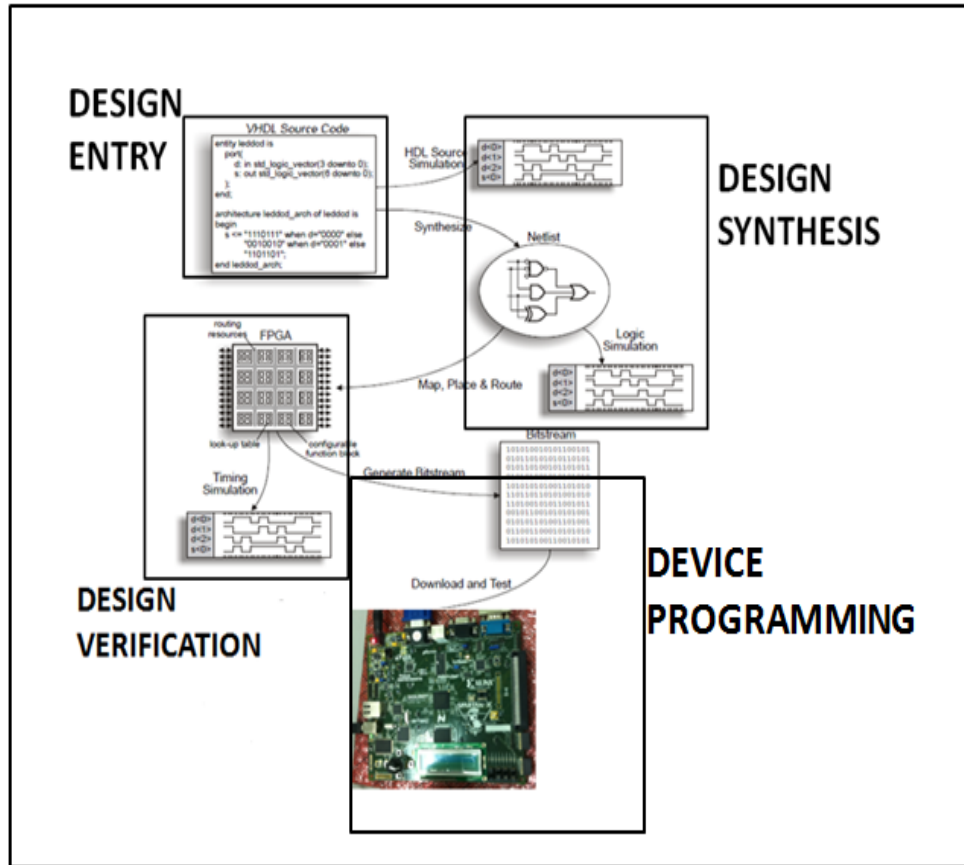


Figure II-5 Implementation of a logic design with the FPGA

IMPACT downloads the bit stream generated into a physical FPGA chip (usually embedded in some larger system). The electronic switches in the FPGA open or close in response to the binary bits generated. Once downloaded, the FPGA will perform the operations specified by the design.

2.5 Chapter Summary

Chapter 2 focuses on background review of various imaging and video applications. The major concerns of these applications are discussed to grasp the need of hardware models. The working and structure of neural network with back propagation

topology is seen in detail. A comparison of concurrent languages to sequential languages is analyzed to enhance the design of the imaging applications for real-time. The working and implementation of logic design on FPGAs is also studied.

III TRACKING IN SURVEILLANCE SYSTEMS

3.1 Tracking in Surveillance systems

Video analysis can be expressed as three major steps: detection of interesting moving objects, tracking of these objects from frame to frame, and analysis of object tracked to recognize their behavior. Novel system architecture is designed for the tracking in surveillance system (TSS) containing two major sub-systems - motion detection system (MDS) and a tracking system (TS). The block diagram of the TSS is shown in Figure III-1.

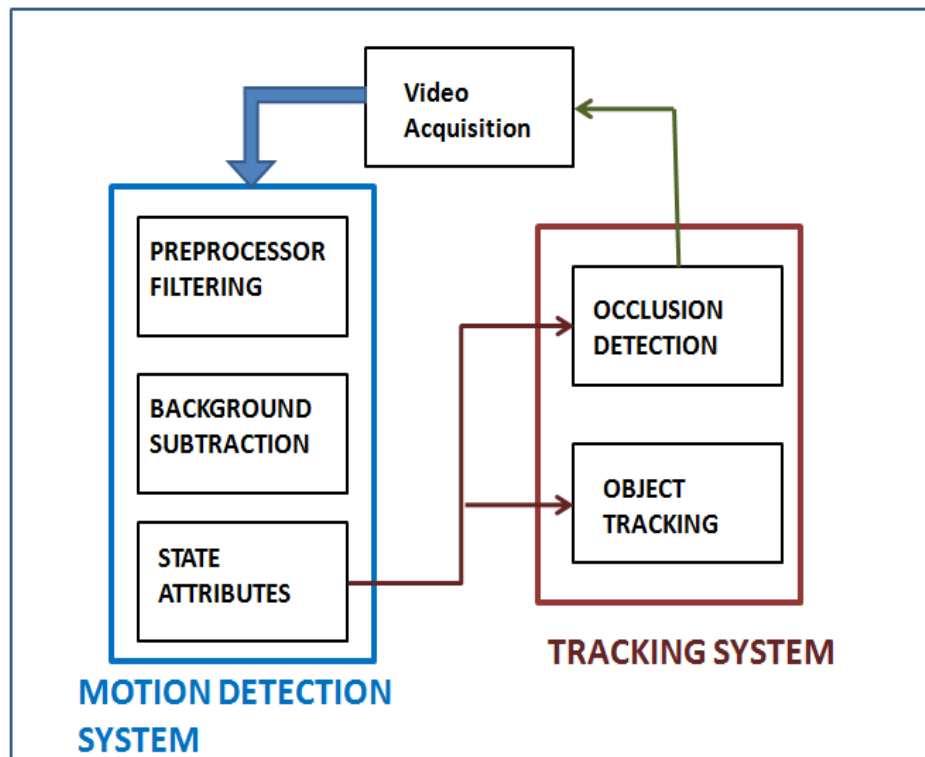


Figure III-1 Tracking in surveillance system

The motion detection system is implemented using the below described algorithm. It has three levels of operation being performed.

- Preprocessor filtering
- Segmentation of non-background objects from still background
- State attributes estimation - position, size, and velocity of the target

3.1.1 Preprocessor filtering

The video obtained from the video acquisition unit is the raw video. To satisfy the memory requirements and the environmental scene conditions, preprocessing of the raw video content is highly important [46]. Various factors like illumination, background, camera parameters, and viewpoint or camera location are used to address the scene complexity. These scene conditions affect images of the same object dramatically.

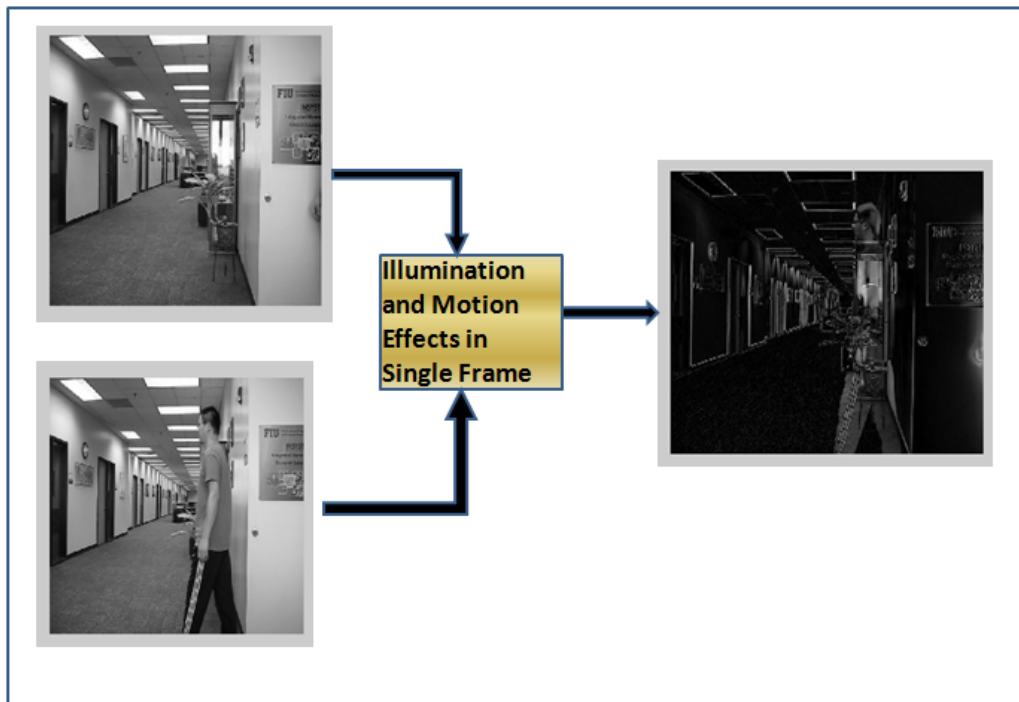


Figure III-2 Scene inconsistency involved- illumination and brightness

Figure III-2 shows the illumination effects and motion variations with time over the same area under surveillance. Generally when stationary camera is involved, it is

expected that the scene conditions remain the same in an indoor environment with time. But actually even under such conditions there is a change in the illumination. This could be because of shadows, variation of perceived illumination in indoors due to the changes in outside world through glass windows. It can be observed that the segmentation of the moving object cannot be classified due to the unwanted changes and interrupted errors present.

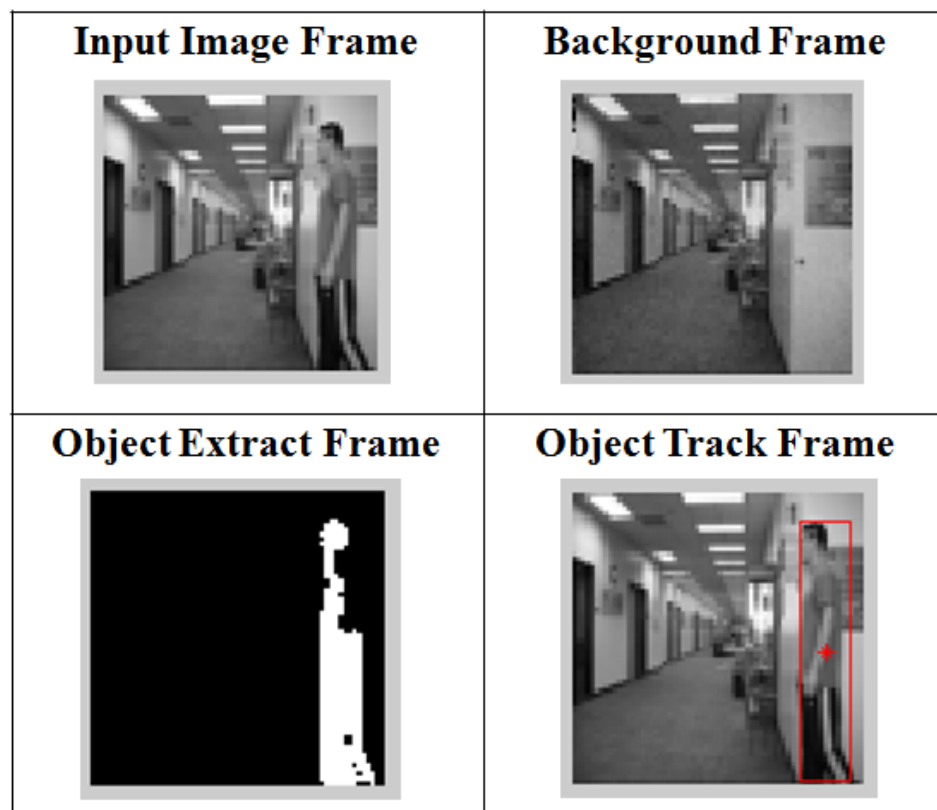


Figure III-3 Background subtraction analysis

3.1.2 Background subtraction analysis

A common approach for detecting a moving object is background subtraction. The main idea is to subtract the current image from a reference image i.e. background image that is

constructed from the static image pixels during a period of time. Background modeling/subtraction is the first step in detecting and identifying objects or people in videos. A model of the scene background is built and each pixel in the image is analyzed. A pixel's deviation in color and/or intensity values is used to determine whether the pixel belongs to the background or the foreground [46].

There are various problems in background modeling that must be addressed or handled.

A few are listed below [47]:

- Illumination changes - lighting conditions;
- Presence of a moving object during the initialization of the background scene;
- Effects of moving elements of the scene background (e.g. swaying tree branches);
- Very low speed or motionless foreground object, appears to be the same as a background object that moves and then becomes motionless;
- background objects that are inserted or removed from the scene;

Running Gaussian average method [48] is used to obtain the background subtraction as it is very fast and consumes low memory when compared to other methods. The pdf of the background is given by Equation III-1.

$$B_{i+1} = \alpha F_i + (1 - \alpha)B_i \quad \text{Equation III-1}$$

Where α is the learning rate (typically 0.05), i refers to the current frame index, B refers to the background frame, F refers to the actual image frame. This takes into consideration of the illumination changes like lightning, camera motion changes, high frequency background objects, such as the tree leaves and branches. The objects in each new frame

are obtained by calculating the difference between current frame and the scene's static background using Equation III-2.

$$|F_i - B_i| > T \quad \text{Equation III-2}$$

Where T represents the value of threshold chosen per requirement of the application. A closer approximation is made by choosing the value of threshold by fitting a Gaussian distribution (μ, σ) curve [70] to the histogram curve using Equations III-3 and III-4. T is chosen to be $(k\sigma)$.

$$\mu_{t+1} = \alpha F_t + (1 - \alpha)\mu \quad \text{Equation III-3}$$

$$\sigma_{t+1}^2 = \alpha(F_t - \mu_t)^2 + (1 - \alpha)\sigma_t^2 \quad \text{Equation III-4}$$

The Figure III-3 shows a single frame from a video where the background is subtracted using the Running Gaussian method. The input image frame, background frame, object extracted frame and object tracked frame is shown.

3.1.3 State attributes estimation

In state attributes include the Position, Size and Velocity of the object under consideration. After the background is subtracted, a binary image is formed with black representing the background and white representing the object/ target. The detected white pixels are bounded in a rectangle and the pixel location value is taken as the position of the object under consideration. Since the position needs to be represented by a point source, the centre of the rectangle is used. From the image the size and position are estimated as shown.

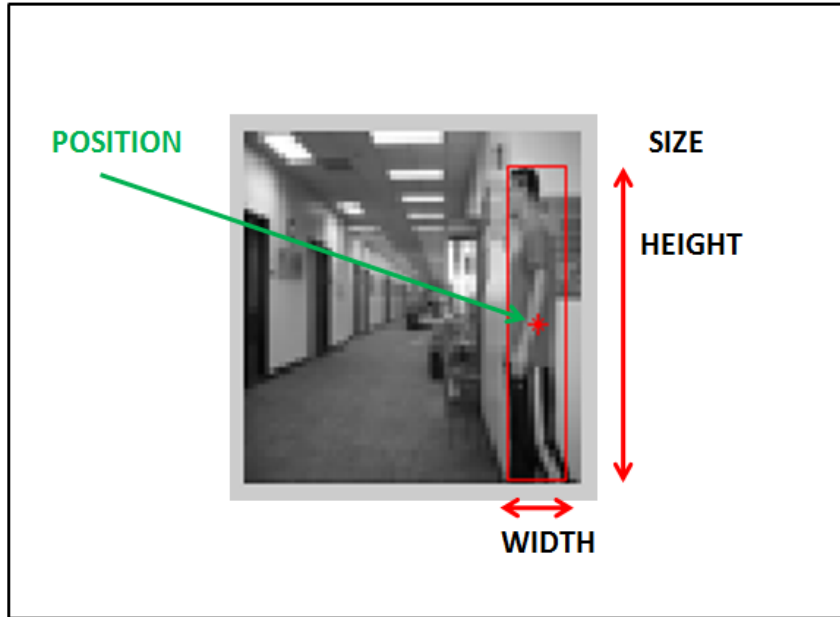


Figure III-4 Position and Size attributes

Take the midpoint of the horizontal sides (h_{end} and h_{start} indicate the horizontal end and start of the X co-ordinates – Width shown in Figure III-4) of the rectangle boundary detected:

$$X_{centroid} = (X_{h_{end}} - X_{h_{start}}) / 2 \quad \text{Equation III-5}$$

Take the midpoint of the vertical sides (v_{end} and v_{start} indicate the vertical end and start of the Y co-ordinates- Height shown in Figure III-4) of the rectangle boundary detected:

$$Y_{centroid} = (Y_{v_{end}} - Y_{v_{start}}) / 2 \quad \text{Equation III-6}$$

To calculate the size of the object, the object is bounded by a rectangle of a particular width and height obtained while tracking. Hence the area forms the size of the object under surveillance. Figure III-4 shows the object track frame obtained with the position and size being tracked using MATLAB code.

Velocity Estimation using Kalman filter tracking

In general adjacent frames of a video are similar and changes that occur are due to object or camera motion (hence possess temporal correlation). Kalman filter is an optimal recursive Bayesian filter for linear functions and Gaussian noise. The Kalman filter tracking consists of predicting and updating stages [49]. First uses the model to predict the future state, this prediction is then corrected by incorporating the observation. Finally the output from the correction is again used as input for the next prediction shown in Figure III-5.

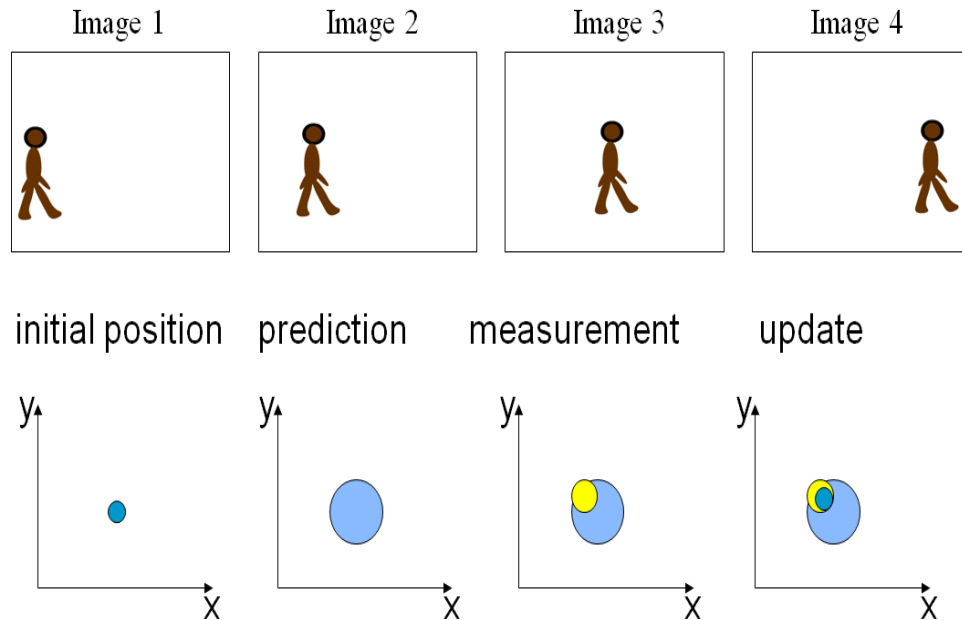


Figure III-5 Kalman filter tracking stages

Estimates the state x of a discrete-time controlled process that is governed by the linear stochastic difference Equation III-7 [50] – [52]

$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t \quad \text{Equation III-7}$$

With a measurement $z_t = C_t x_t + \delta_t$; $p(x) \approx N(\mu, \Sigma)$

A_t : Matrix ($n \times n$) that describes how the state evolves from t to $t+1$ without controls or noise.

B_t : Matrix ($n \times i$) that describes how the control u_t changes the state from t to $t+1$.

C_t : Matrix ($k \times n$) that describes how to map the state x_t to an observation z_t .

ε_t, δ_t : Random variables representing the process and measurement noise that are assumed to be independent and normally distributed with covariance R_t and Q_t respectively.

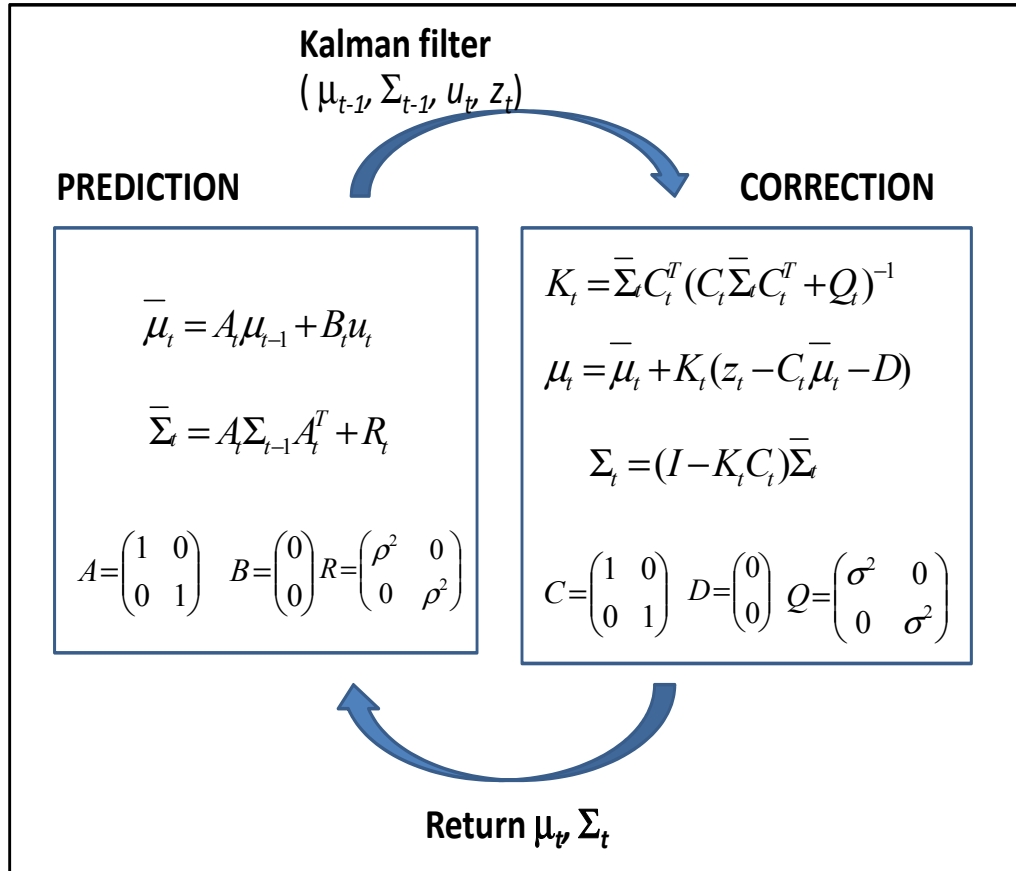


Figure III-6 Kalman filter for tracking – Prediction and correction stages [52].

The state of the system is represented as a vector or real numbers. At each discrete time instant, a linear operator is applied to the state to generate the new state with some noise mixed in shown in Figure III-6. Thus velocity of the object under consideration is being calculated during tracking.

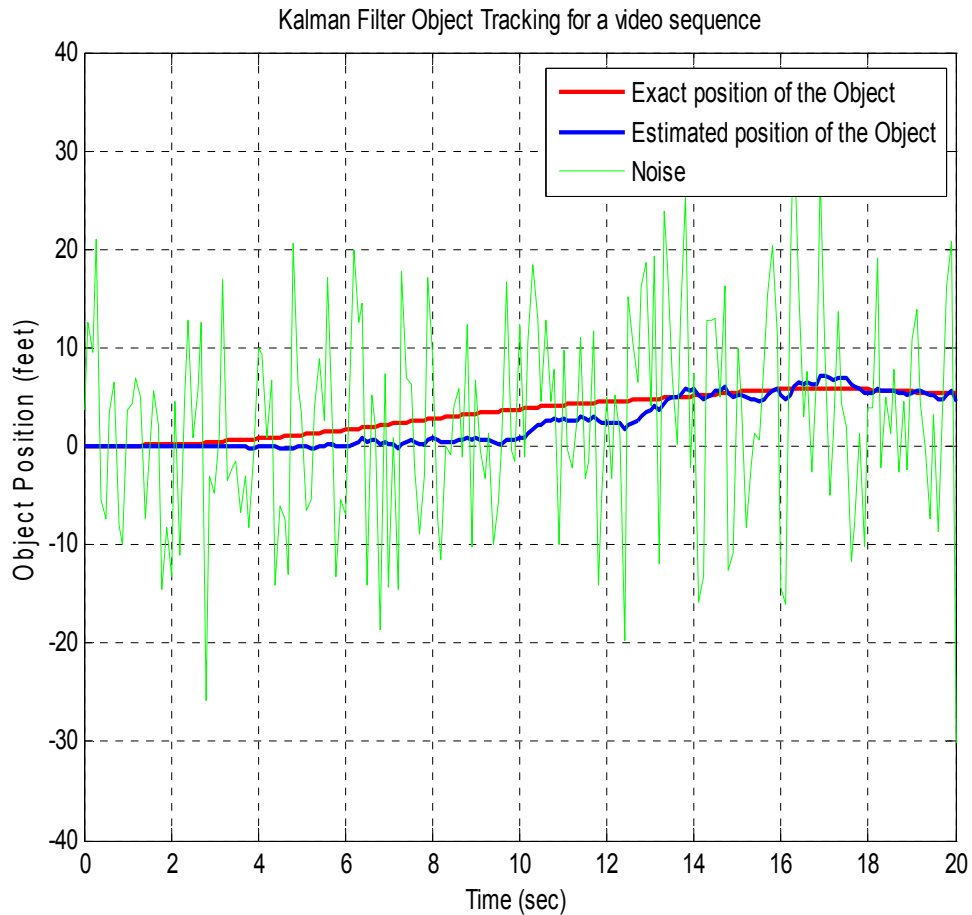


Figure III-7 Object tracking of a video sequence of 20 seconds

The Figure III-7 shows the Kalman filter object tracking of a video sequence of 20 seconds. The exact position of the object (red), the estimated position of the object (blue) and the measured noise (green) are being plotted. It can be observed that the tracking is almost exact.

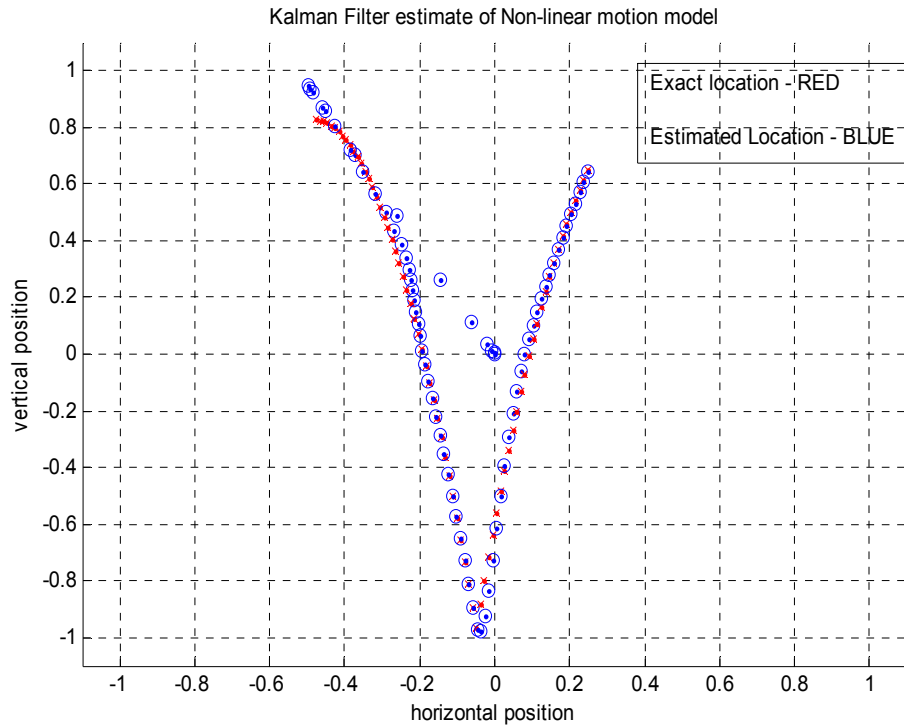


Figure III-8 Kalman filter estimate of Non-linear motion.

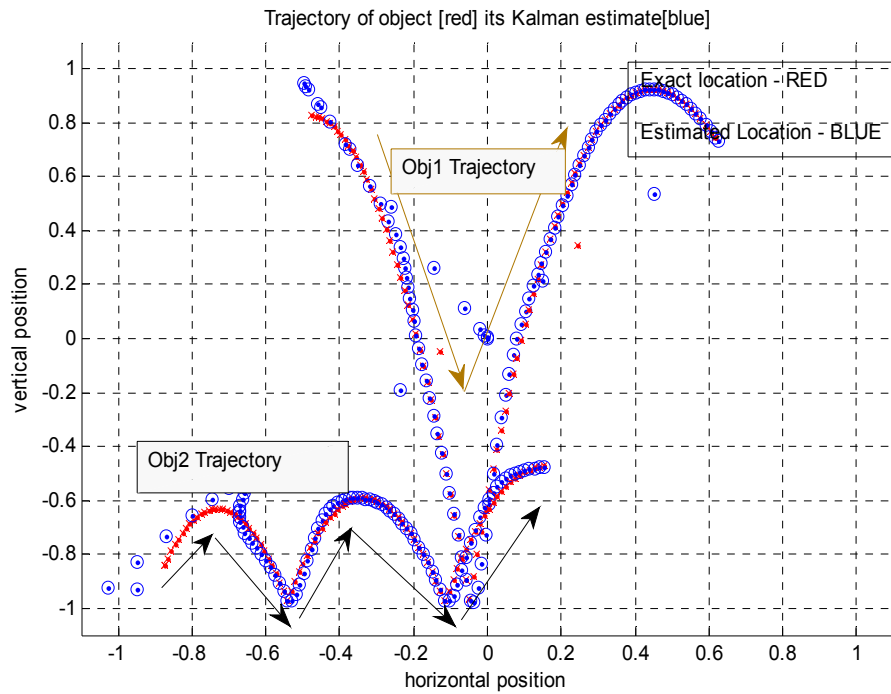


Figure III-9 Kalman filter estimate of multiple objects - Non-linear motion

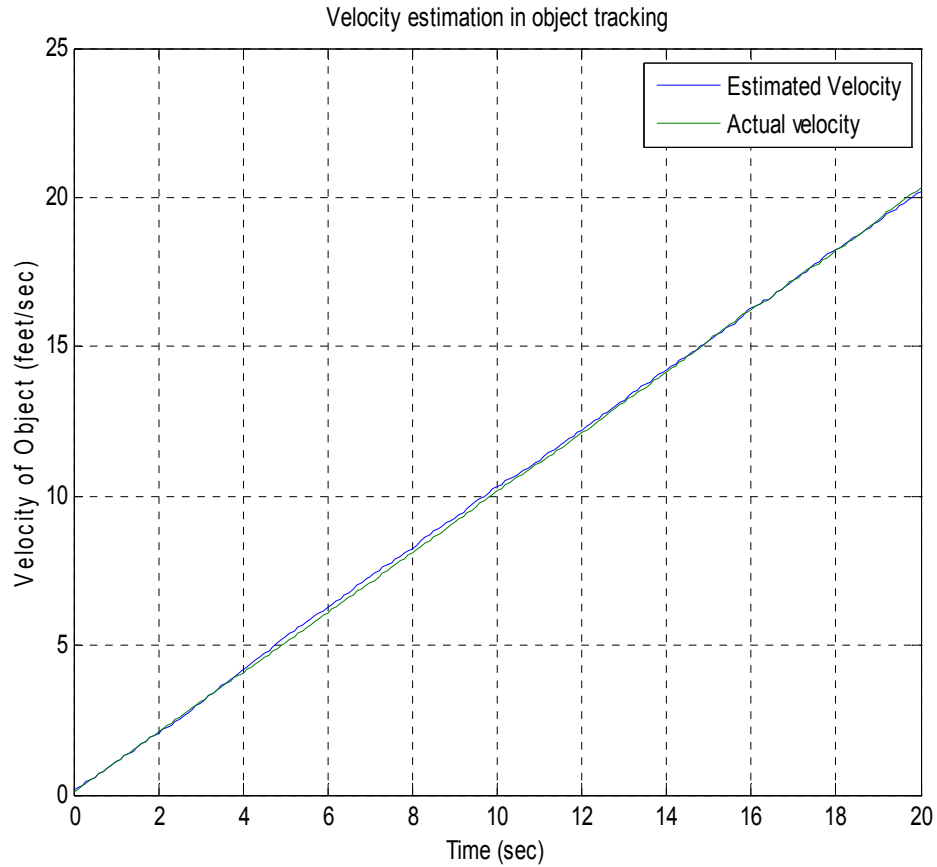


Figure III-10 Velocity estimate in object tracking

Figure III-8 and Figure III-9 show the tracking of non-linear motion models by Kalman filter estimates. Both single and multiple objects are considered for tracking. The model is highly accurate in estimating the position of the objects. Figure III-10 shows the velocity estimation of the object being tracked. The velocity depends on the size and position attributes of the object under consideration.

3.2 Occlusion Detection

Tracking is a challenging task when there are complex interactions between targets as seen in [53]-[55]. It is important to be able to track multiple objects simultaneously to

obtain favorable results. Though interactions between the objects exist, objects tend to keep relative positions or spatial lay out during a short period of time. The relative spatial layout is maintained throughout for linear motions and for a few frames in case of non-linear motion. Figure III-11 shows four aircraft in motion taken from one camera at different focal lengths – displaying without and with occlusion [10].



Figure III-11 Aircraft trajectories in space [10]

Tracking with multiple cameras not only increases the monitored area, but also helps to disambiguate in matching when subjects are occluded from a certain viewing angle [54]. The attributes POSITION, VELOCITY and SIZE describe the STATE of the object. In Figure III-12, Object 1 and Object 2 are partially occluded and the view from Camera 1 cannot capture the information of Object 1 after some time as it may be totally

occluded by Object 2. In such case, an additional camera is needed to change the view perspective into a different angle. Since the objects are all in 3D mapped on a 2D plane, it may not be effective to change the angle of a single camera to view all the multiple objects at the same time without occlusion [10].

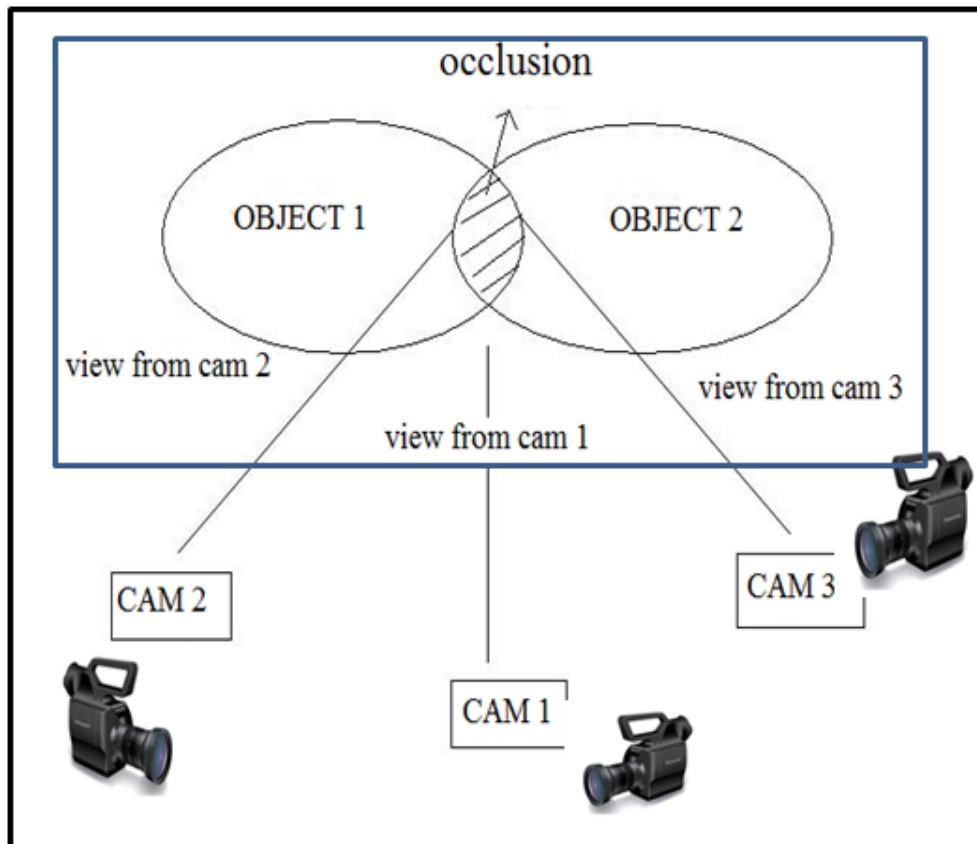


Figure III-12 Multiple camera scene view

3.2.1 Motion of Objects

To represent the objects we consider the center of gravity (CoG) as the object's position displayed using point sources. The co-ordinates of the CoG are obtained from the position attributes from the state attributes estimator. The objects are moving independently of each other. In Figure III-13, there are 3 objects (shown as point sources)

that are moving independently. The motion vectors of the objects found using the above state attributes estimator. Hence, the motion vectors would allow a prediction to be made that the state is continued until a few frames have passed. Thus, the speed and direction of each object is considered known [10].

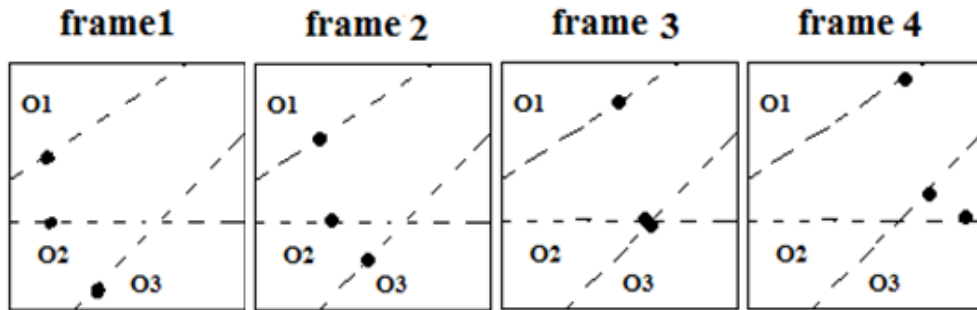


Figure III-13 Tracking of four objects using camera 1 [10]

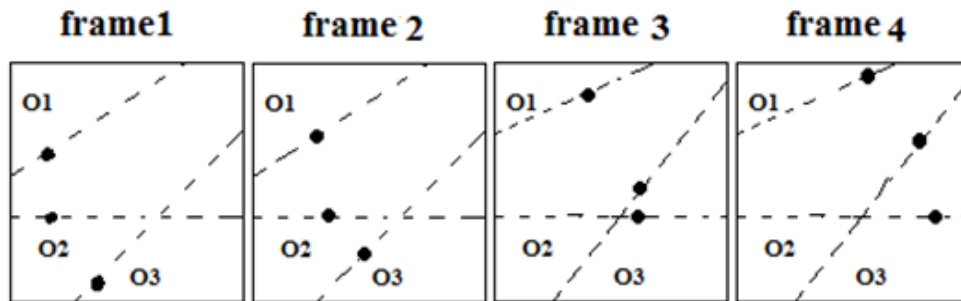


Figure III-14 Tracking of four objects using camera 2 [10]

In Figure III-14, the frames 1-4 show three moving objects O1, O2 and O3 at four different time intervals. It can be observed that in frame 3, occlusion has occurred and it may seem that only two objects are present at that particular time interval. Using our algorithm occlusion occurrence can be detected ahead. If two cameras are considered to be monitoring all the time there is a huge loss of memory, bandwidth and power consumption. Instead the other camera is triggered by the occlusion detector for the additional information to keep track of the objects under occlusion [10].

3.2.2 Two-Dimensional linear motion

Let us suppose that there exists a linear relationship for velocity defined by the Equation III-8 for a time interval $\Delta t = (t - t_i)$.

$$V = U + a\Delta t \quad \text{Equation III-8}$$

Where, V is velocity, U is an initial velocity, a is acceleration, t_i is initial time, and t is a current state time. Let $P_{ik}(r_{ik}, V_{ik}, R_{ik})$ represent the object parameters, where r_{ik} the position of the object in 2-D space is, V_{ik} is object's velocity, and R_{ik} is the object's attributes, for all i objects in all k time intervals [10].

Let $r_{ik} = (x_{ik}, y_{ik})$ be the position vector for a moving object in the two dimensional case.

Therefore, the distance between two objects is calculated using Equation III-9.

$$d_{ik} = \sqrt{(x_i - x_k)^2 + (y_i - y_k)^2} \quad \text{Equation III-9}$$

3.2.3 Two-Dimensional non-linear motion

Two cases can be considered for the kinematics model of the system depending on the dimension of the environment. In the first case it is considered a number of vehicles or systems are moving in a 2D environment. Systems are modeled as a point mass in 2D, or planar motion. In the second case, the system is modeled as a point mass in a 3D environment. The discrete matrix form for the equation governing the motion of both systems is given by Equation III-10 in a 3-D environment [10].

$$\begin{bmatrix} X \\ V \end{bmatrix}_{k+1} = A \begin{bmatrix} X \\ V \end{bmatrix}_k + Ba_k \quad \text{Equation III-10}$$

Where $X = [x \ y \ z]^T$, $V = [v_x \ v_y \ v_z]^T$, $a = [a_x \ a_y \ a_z]^T$,

$$A = \begin{bmatrix} I_3 & \Delta t, I_3 \\ O_3 & I_3 \end{bmatrix}, \text{ and } B = \begin{bmatrix} O_3 \\ \Delta t, I_3 \end{bmatrix}$$

The subscript k represents the discrete time step, I_3 represents an identity matrix of size 3×3 , T represents the transpose of a matrix, and O_3 is a zero matrix of size 3×3 . Vectors X , V and a , respectively, represent the position, velocity, and acceleration input in the initial frame [10].

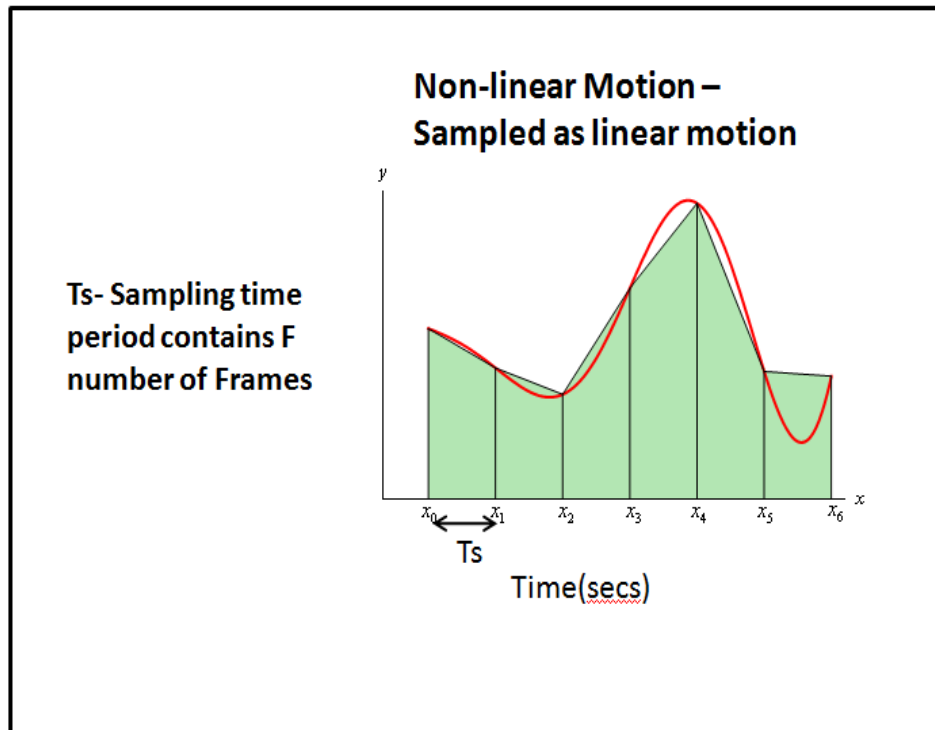


Figure III-15 Non-linear motion assumed to be sampled as linear motion – state space continuity relation is assumed

Formulae used to govern motion for a 2D environment are identical to formulae given above without the terms associated with the z-axis. The identity and zero matrices are reduced to size 2×2 , I_2 and O_2 respectively [56]. We consider only the 2D motion equations for the occlusion detection in this research and then in future extend to the 3D

3.3 Calculation of Threshold

The object can be of any rough shape, and one must therefore take into consideration about the size in limiting the threshold value. Any object could be completely bounded inside a sphere (for 3D) or a circle (for 2D) shown in Figure III-16.

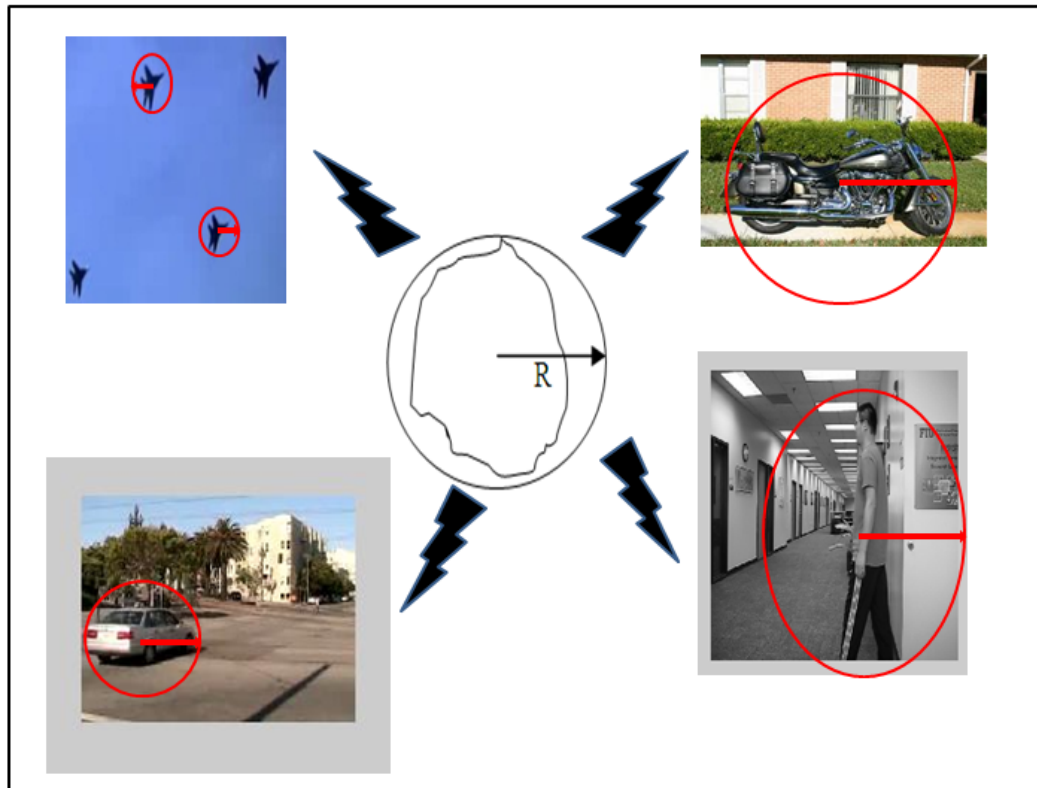


Figure III-16 An object in a bounding sphere

Let the objects be bounded in a sphere of radius ‘ R ’. Objects of different sizes are assumed as $R_1, R_2, R_3, \dots, R_n$ correspond to the radius of their bounding sphere. The threshold considerations are given in Equation III-11, 12, 13.

$$Threshold = 2xMin\{R_1, R_2, R_3, \dots, R_n\} + \eta \quad \text{Equation III-11}$$

$$2xMax\{R_1, R_2, R_3, \dots, R_n\} + \eta \quad \text{Equation III-12}$$

$$2xAvg\{R_1, R_2, R_3, \dots, R_n\} + \eta \quad \text{Equation III-13}$$

$$D_{ik} \geq R_i + R_k \quad \text{Equation III-14}$$

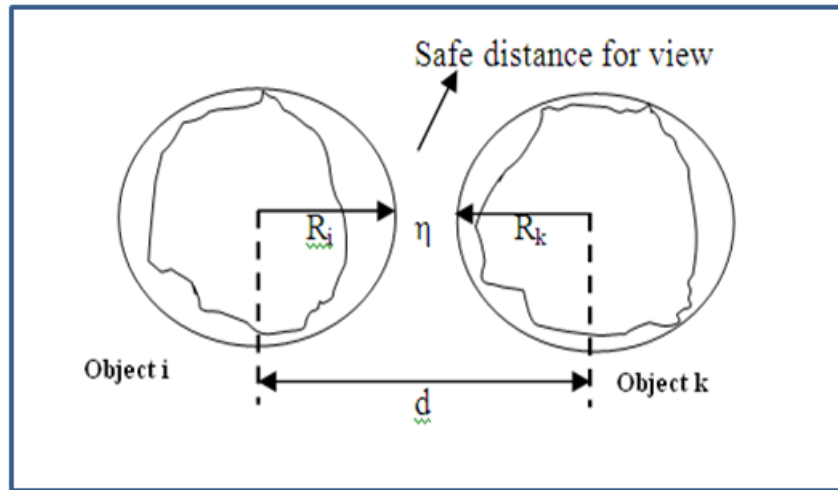


Figure III-17 Threshold calculation [10]

The distance between two objects D_{ik} is calculated using Equation III-14 shown in Figure III-17. The safe distance factor is denoted by ‘ η ’ which is a function of velocities of the moving objects and time correspondence factor listed in Equation III-15 [10].

The time correspondence factor is the time needed for the second camera to correlate and identify the objects in the frame with the first camera. Also, it is required to consider the focal lengths of the camera to maintain the spatial coordinate ratio with the frame ratio.

$$\eta = f(V_{ij}, \tau) * \rho \quad \text{Equation III-15}$$

Where V_{ij} is the velocity of i^{th} object at j^{th} time interval, τ is the time correspondence factor needed for the second camera to correspond with first camera, ρ is the ratio of spatial coordinates ratio to frame ratio [10].

3.4 Algorithm flow control

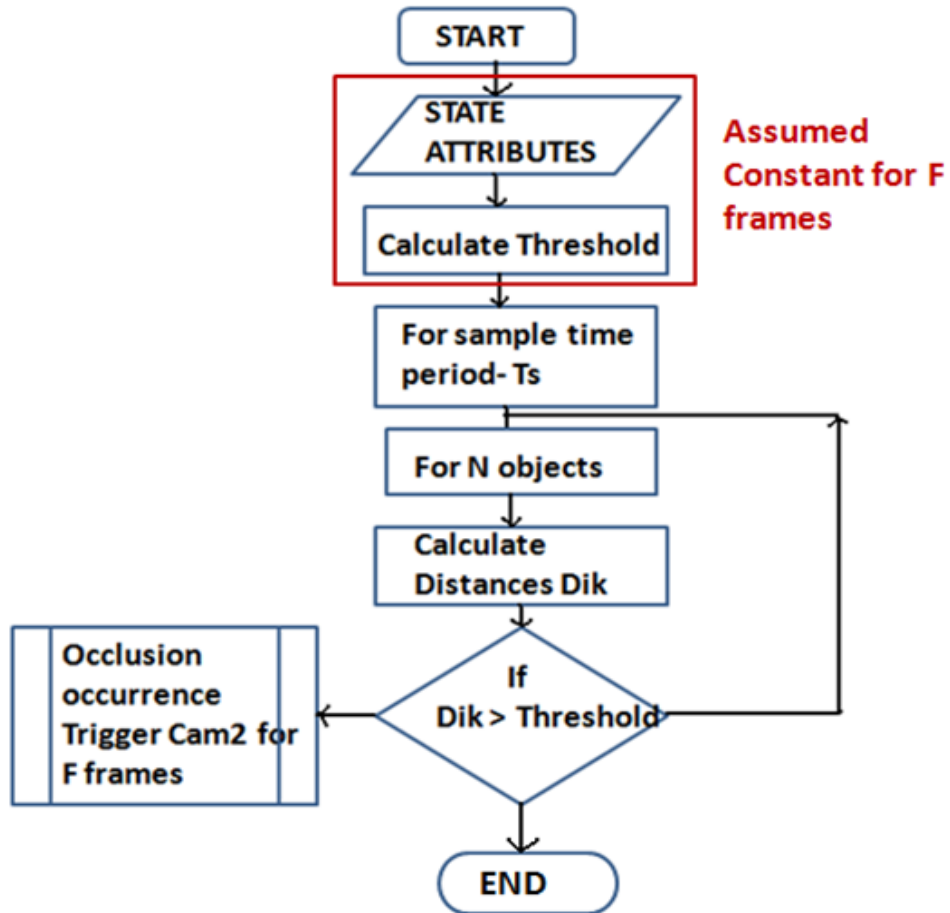


Figure III-18 Occlusion occurrence detection algorithm

The algorithm for the decision of occlusion occurrence is shown in Figure III-18. Once the occlusion is detected, the camera 2 is 'ON'. Now information is forthcoming from both cameras. In order for camera 2 to track the object, correlation algorithm should be executed on the images using Equation III-16 from camera 1 and camera 2 [57].

$$Cr(x, y) = \frac{\sum_{y'=0}^{h-1} \sum_{x'=0}^{w-1} C'(x', y) C(x+x', y+y')}{\sqrt{\sum_{y'=0}^{h-1} \sum_{x'=0}^{w-1} C'(x', y)^2 \sum_{y'=0}^{h-1} \sum_{x'=0}^{w-1} C(x+x', y+y')^2}}$$

Equation
III-16

$C'(x', y)$ is the pixel value from Camera 1 to be correlated to Camera 2. $C(x, y)$ is the pixel value from Camera 2 of the searching area at (x, y) . The correlation algorithm describes the similarity between the frames and identifies the objects corresponding to Camera 1 [10].

3.5 Experimental Results

Both the motion models have been implemented according to the algorithmic strategy developed using MATLAB simulation tool.

3.5.1 Linear Motion Analysis

As shown in the Figure III-19, three moving objects with linear motion are considered with the space continuity relation. It is assumed that the objects are moving with similar state attributes and continued velocity vectors for a certain time period (F set of frames) following the state of continuity. Figure III-20 is the plot of the calculated shortest distances between every combination of two objects present in the frame considered. Figure III-21 shows the various threshold lines calculated based on the different considerations to obtain threshold values. Using the minimum value of size attributes in Equation III-11 is unsatisfactory due to its lack of detection of the occurrence

of occlusion in either advance or the later. The cut-off line is the last line in the plot, and there are no values lying below the threshold indicating an absence of the occurrence of occlusion [10].

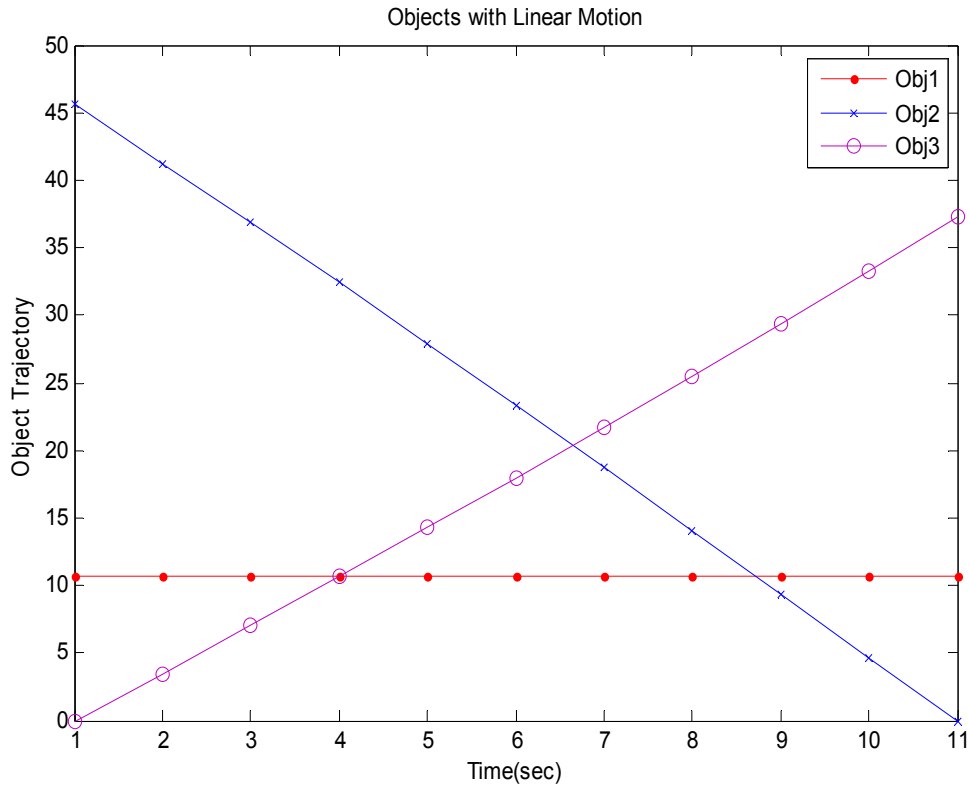


Figure III-19 Three objects of motion with linearity

Using the maximum of size attributes in Equation III-12 is not a sufficient option but, definitely a better option than Equation III-11 since it detects the occurrence of occlusion for two points as shown in Figure III-21. The first of the cut-off lines is the line considered, and there are two points below this threshold, implying the occurrence of two points of occlusion [10].

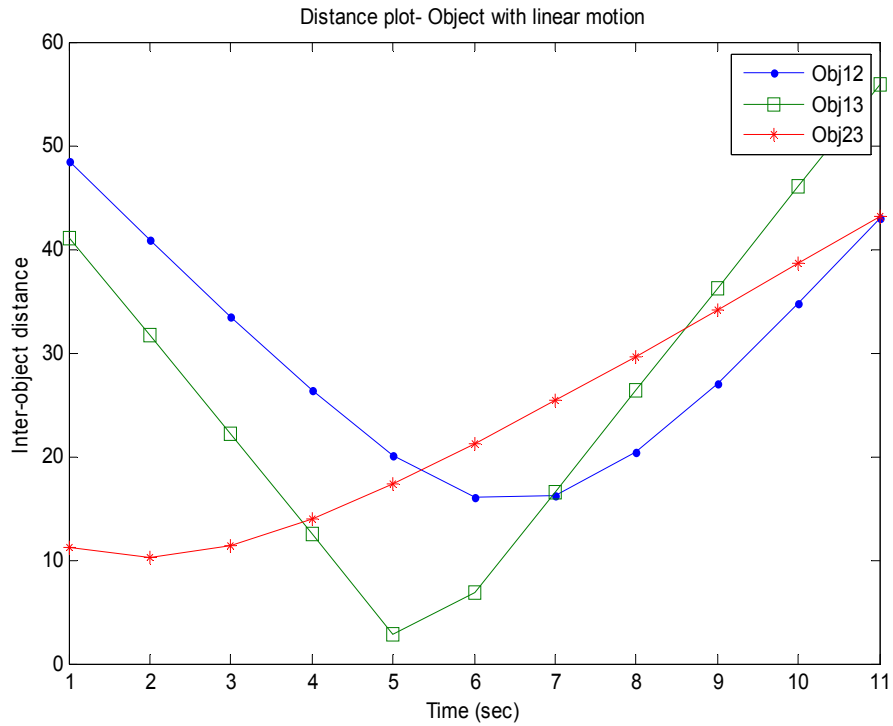


Figure III-20 Three objects of motion with linearity- Plot with the distance between two objects

The threshold using Equation III-14 is the best option to detect the occlusion but involves more evaluation time and more mathematical complications. Thus using the mean of the size attributes of objects added with the safe distance factor in Equation III-13 would yield the best results to detect and prevent occlusion [10].

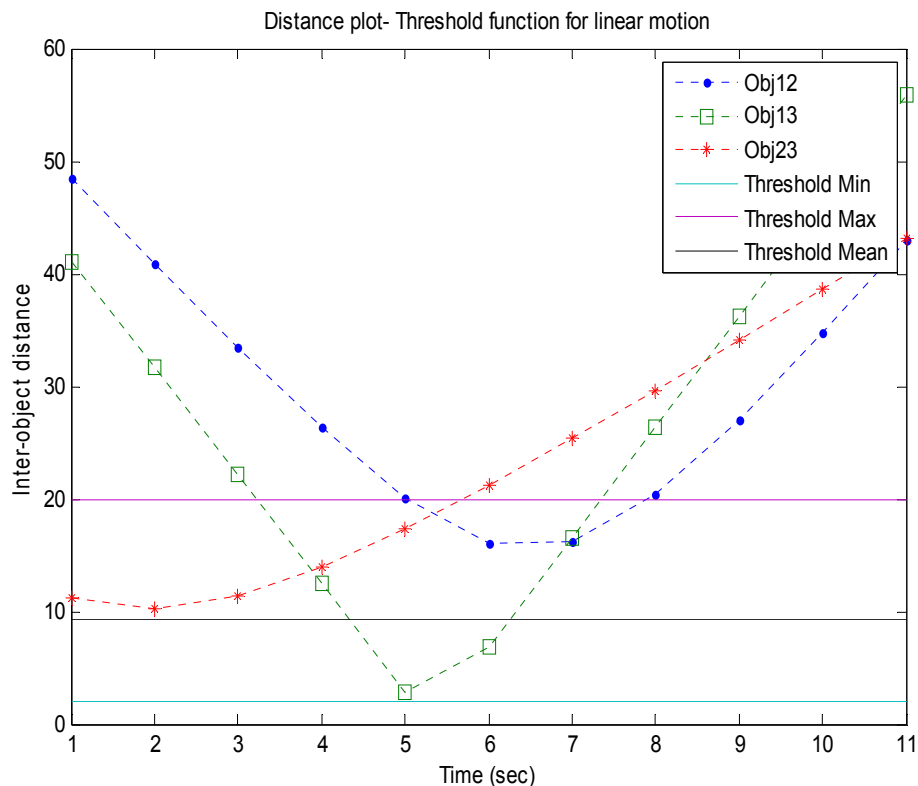


Figure III-21 Three objects of motion with linearity - Plot with distances and different values of threshold

It can be observed that from the plot there is the possibility of one occlusion event to take place during the whole consideration of the set of frames. Therefore, at this point another camera is activated, and using correlation, the correspondence between the objects is identified and used to track them from a different view and angle. Again, the algorithm is run to detect another occurrence of occlusion [10].

3.5.2 Non-Linear motion analysis

Figure III-22 shows the nonlinear motion of three different objects moving in space. It is assumed that the objects are moving with similar state attributes and

continued velocity vectors for a certain time period (F set of frames) following the state of continuity.

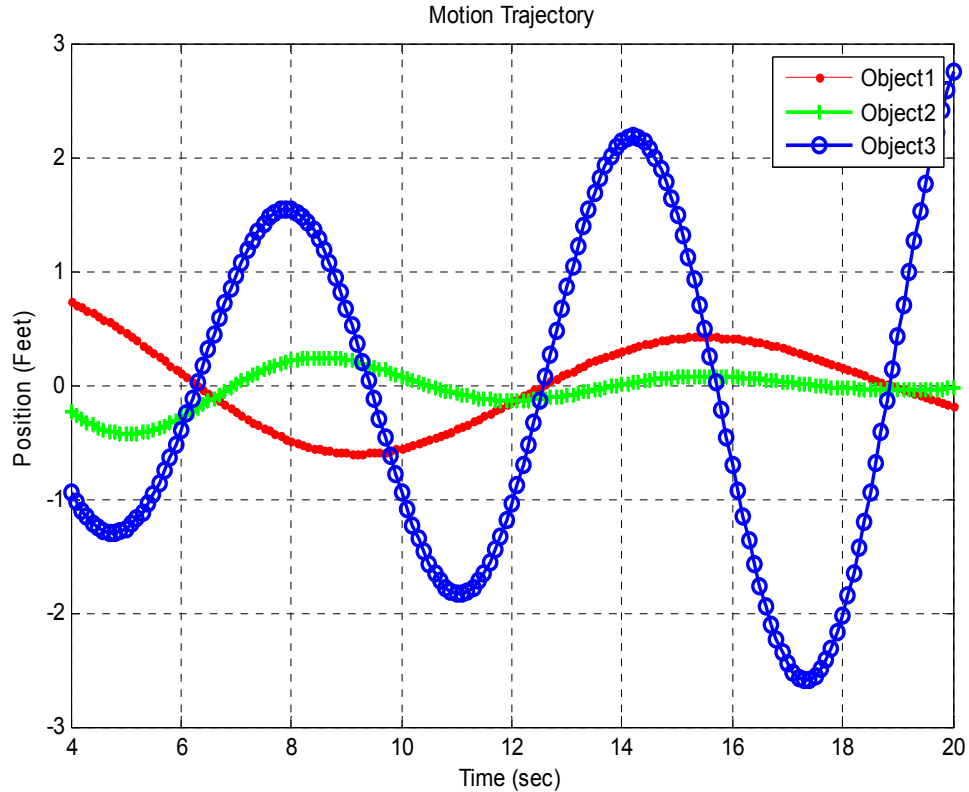


Figure III-22 Three objects of motion with non- linearity

In Figure III-23 the shortest distances between all combinations of two objects are considered and plotted. Again, different threshold values are considered and plotted in Figure III-24 and Figure III-25. It can be observed that in non linear motion there is more probability for occurrence of occlusion within a short interval of time periods compared to linear motion. The threshold depends on the velocities and size attributes of the objects considered, and vary accordingly. Figure III-24 indicates that all the threshold calculations lead to cut-off lines which are almost near or coinciding [10].

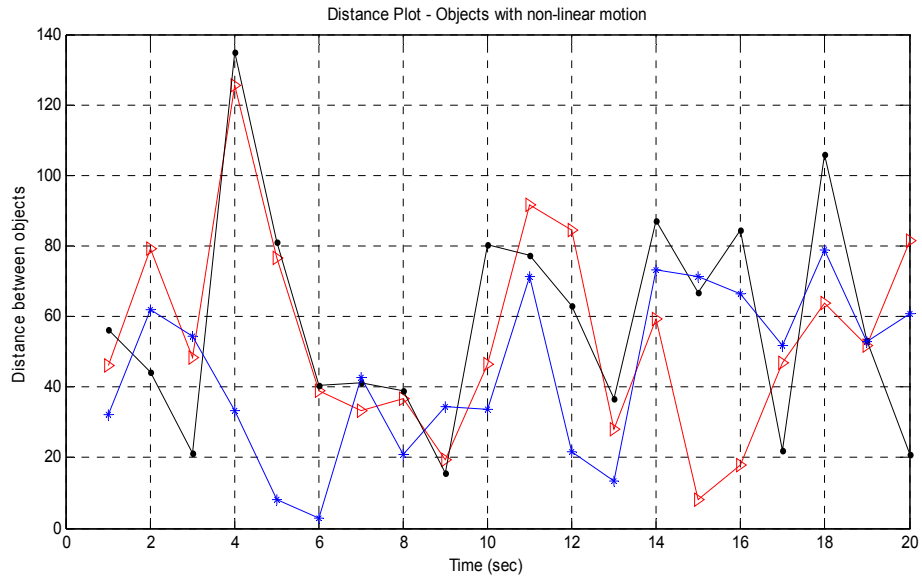


Figure III-23 Three objects of motion with non- linearity- Plot with the distance between two objects

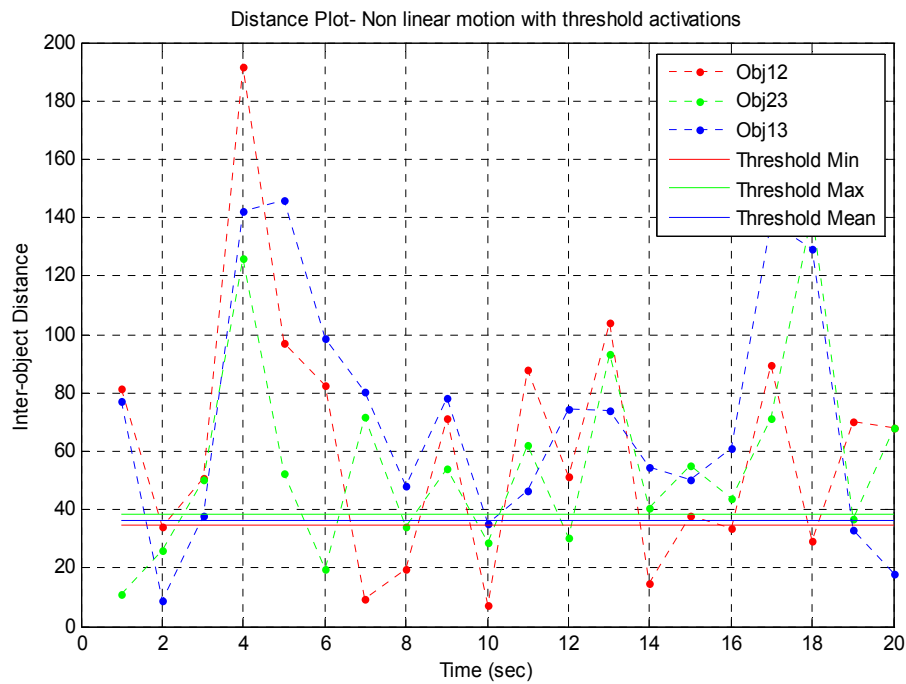


Figure III-244 Three objects of motion with non- linearity - plot with distances and different value of thresholds found based on the similar sizes and velocity of the objects

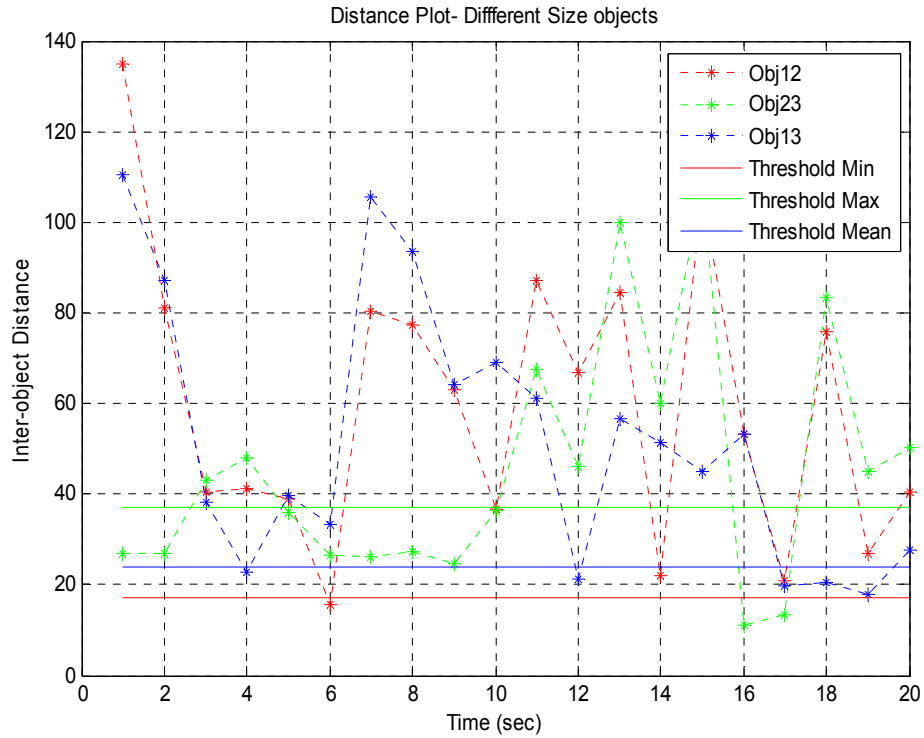


Figure III-255 Three objects of motion with non- linearity - plot with distances and different value of thresholds found different sizes and velocities of the objects

In the case where the attributes of all objects are very similar, the probability of occurrence of occlusion is very high and using two cameras to track the objects is a better solution. In Figure III-25, the thresholds corresponding to objects with different size and velocity attributes are plotted and can be seen that the number of occurrences of occlusion are less that that compared to Figure III-24. Hence, between the frames considered in time of occlusion, the second camera has to be activated to a get the better track of the objects [10].

3.6 Chapter Summary

Chapter 3 focuses on linear and non-linear motion models of moving objects under occlusion effects. These models are assumed to follow the space continuity relation. The algorithm for detection of occurrence of objects in the 2D model in the surveillance system called TSS was developed. Results supporting the algorithm designed indicate the efficiency of the algorithm in case of the multiple objects model using two cameras. Future work can include more complex considerations for the occlusion problem in 3D space and estimation of real-time HDL based occlusion free tracking.

IV A GLOBAL APPROACH FOR TARGET DETECTION

The target detection is the process of identifying region of interest (ROI) in images that one hope corresponds to target. Automated detection is probably the most difficult problem in computer vision. There are three major reasons why automated detection is so hard.

- Much information is lost when 3-D scenes are projected to 2-D.
- Target detection attempts to detect basic object regions (whose constitution is ambiguous to predict).
- Humans use brains extensively in their perceptual processes which are easy for them to recognize whether certain parts belong or do not belong together. This is not because of similar properties of the regions, but because that they form parts of the same known and recognizable object.

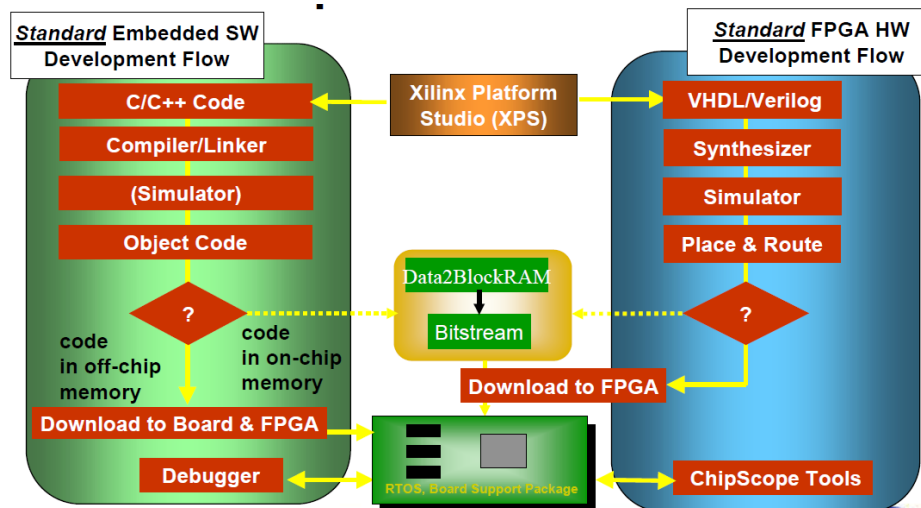


Figure IV-1 Xilinx FPGA EDK system design [65].

Target detection is one of the most important steps leading to the analysis of processed image data. Its main goal is to divide an image into parts that have a strong correlation with objects or areas of the real world contained in the image. A simple global approach can be used to segment an image frame into foreground and background. This kind of processing is called context independent processing which infers no object-related model and no priori- information related to the scene is used to obtain good results. When complex scenes are considered accurate target detection usually cannot be achieved in the low-level processing phase [58, 69]. Instead a reasonable approach is to use different adaptive threshold which yields different levels of segmentation which is used as an input to higher level processing.

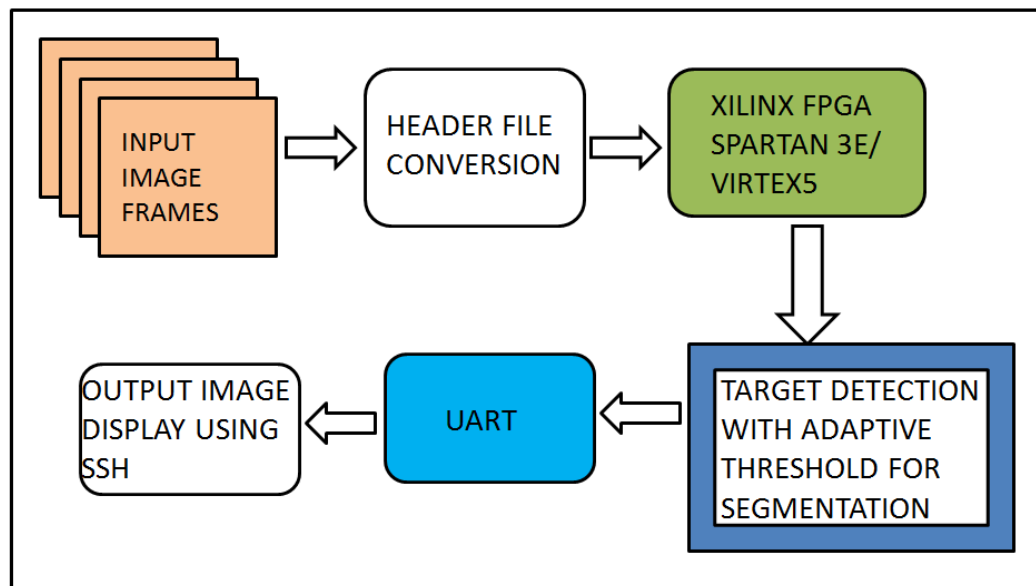


Figure IV-2 Block diagram of FPGA based target detection using adaptive threshold segmentation.

Adoption of Xilinx FPGA provides the advantage of implementing the logic required by an application using separate hardware for each function. FPGAs are inherently parallel. The hardware design provides higher speeds while retaining the reprogrammable flexibility of software at a relatively low cost. Hence FPGAs are well suited to video/image processing applications, particularly at the low and intermediate levels where they are able to exploit the parallelism inherent in images [59, 68]. Also, the FPGA recently manufactured possess embedded DSP processors. The FPGA's possess an optional feature of ARM processor to be included represents a significant advance in embedded board-level products. A view of the Xilinx FPGA EDK system design is being shown in Figure IV-1 which provides the tool chain of the hardware and software co-simulation platform.

A new architecture is designed as shown in Figure IV-2 below. The block diagram depicts the flow of the image frames as the input into the system which is then converted into header files for processing on the FPGA. The algorithm is global and can satisfy for any scene condition (indoor/ outdoor, lightning variations) and fits on any FPGA with minimum DDR SRAM of 1MB. Once the processing is finished, the output is being communicated via UART cable and displayed on SSH for convenience.

4.1 DSP vs. FPGA

The Digital Signal Processor (DSP) is a specialized microprocessor (generally programmed in C or assembly language) mostly used for extremely complex math-intensive tasks, with conditional processing. It is limited in performance by the clock rate,

and the number of useful operations done per clock. On the other hand, FPGA is an uncommitted "sea of gates". The device is programmed by connecting the gates together to form multipliers, registers, adders and so forth. The Table IV-1 below shows the comparison among both the devices.

Table IV-1 DSP vs. FPGA

DSP	FPGA
As the sample rate reaches a few MHz, it is difficult for the DSP to transfer the data without any loss. (Since the processor uses shared resources like memory busses, or even the processor core).	Dedicates logic for receiving the data, so can maintain high rates of I/O.
Needs optimized code for use of external memory to work on a large data set.	FPGAs have a limited amount of internal storage but huge external memory can be used via extension port.
As an example, a TMS320C6201 has two multipliers and a 200MHz clock - so can achieve 400M multiplies per second.	As an example, a 1M-gate VIRTEX-II™ device has 40 multipliers that can operate at more than 100MHz. In comparison with the DSP this gives 4000M multiplies per second.

A comparison of choosing the right processor is being shown with MAC based approach in Figure IV-3. Most MAC-based engines are real multipliers. A 24x24-bit multiplier is needed to process 24-bit input samples and coefficients.

To construct a complete MAC unit, an accumulator must be provided on the output of the multiplier. As products from the multipliers are accumulated, the number of bits in the result might exceed 48 bits. In fact, the maximum number of bits required in the accumulator depends on the number of products that will be accumulated (i.e. an application specific number). When implementing a MAC with FPGA, the architect is free to choose the precision of the accumulator. Using Xilinx VIRTEX FPGA technology this multiplier can be implemented using 348 logic slices. This is 11% of a low density device, such as the XCV300 [59].

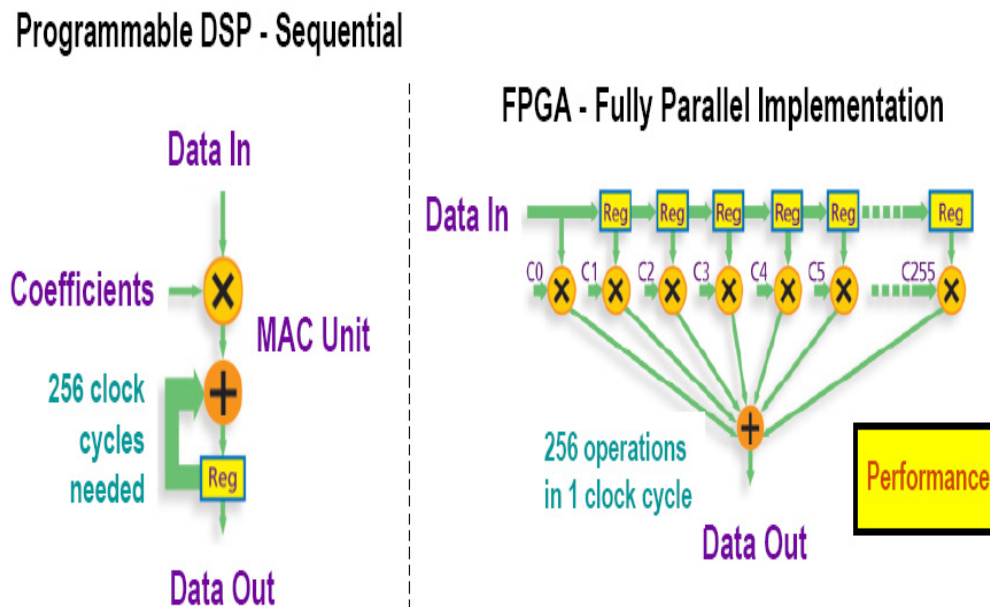


Figure IV-3 DSP processor vs. FPGA (comparison with respect to target detection).

Hence FPGAs are being preferred in a wide variety of video and image processing applications such as target detection because of the aspects:

- High speed performance
- low cost
- flexibility
- low power consumption

4.2 Target Detection Algorithm

4.2.1 Image Acquisition and Frame Generation

The images are capture using a Cannon camera which produces stream of RGB pixels. A brief 10-12 seconds video is recorded in *.avi* format. It is important to consider the temporal resolution requirements of the application. Lower resolution is preferred as it will have a significantly higher acquisition rate for the observation of faster events [60].

The default size of the video frame is 640x480 pixels in the Cannon camera.

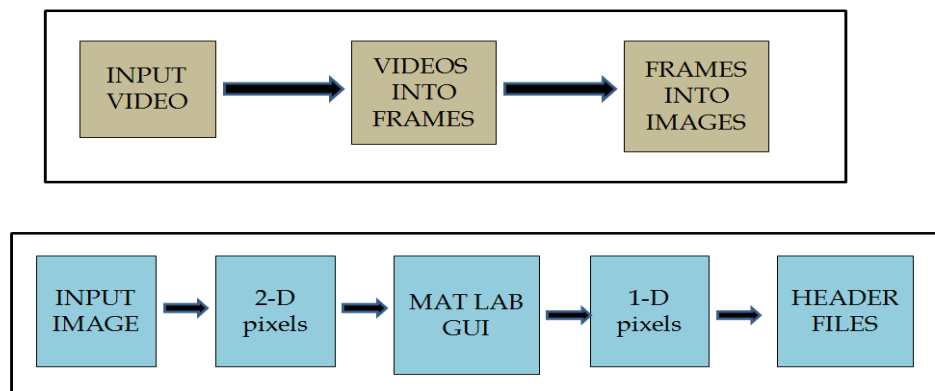


Figure IV-4 Frame generation and Header file creation.

The video obtained is read in to the computer using MATLAB. The software processes the entire video and converts it into Image frames at the rate of 10 frames per second. Depending on the accuracy required and computational capability of the system, the frames can be interlaced. The video is fed in the MATLAB program “Video2Pic.m”. The program reads the *.avi* file and converts it to frames as shown in Figure IV-4 above. The frames are produced at the rate of 10 frames per second. This implies for a 10 second video a total of 100 frames are produced in RGB format. These frames are then stored in JPEG format. Figure IV-3 shows the block diagram of the header file creation module. These files are arranged in the order of their occurrence in the video to configure the change in motion from one frame to another. Generally, the first frame is selected as the initial background frame.

4.2.2 Grayscale conversion

From the JPEG files before they are converted in to header files, the files are present in RGB format at a resolution of 640x480 pixels. These frames are then converted to grayscale within a range of 0-255. This reduces the coherent effect of the environment and allows us to easily separate the object from the background. It is then resized to 128 x 128 pixels to reduce the redundancy in the frames.

4.2.3 Median Filtering for noise reduction

The frames obtained from the image acquisition unit contain raw data. To satisfy the memory requirements and the environmental scene conditions, preprocessing of these

frames is highly important. Various factors like illumination, environmental effects or camera parameters affect images of the same object dramatically.

The low pass smoothing filters reduce noise. However, the underlying assumption is that the neighboring pixels represent additional samples of the same value as the reference pixel, i.e. they represent the same feature. At edges, this is clearly not true, and blurring of features results. There are nonlinear neighborhood operations that can be performed for the purpose of noise reduction that can do a better job of preserving edges than simple smoothing filters. The median filter is a non-linear digital filter normally used to reduce noise in an image.

Table IV-2 Median filter code

```
%MEDIAN FILTER
if (n%2==0)
{
median=(arr[n/2]+arr[n/2+1])/2.0;
}
else
{
median=arr[n/2+1];
}
return (median);
```

Table IV-2 displays the code of the median filter used in the target detection system developed. The low pass smoothing filters reduce noise. However, the underlying assumption is that the neighboring pixels represent additional samples of the same value as the reference pixel, i.e. they represent the same feature. At edges, this is clearly not true, and blurring of features results. There are nonlinear neighborhood operations that can be performed for the purpose of noise reduction that can do a better job of preserving

edges than simple smoothing filters. The median filter is a non-linear digital filter normally used to reduce noise in an image.

In median filtering, the neighboring pixels are ranked according to brightness (intensity) and the median value becomes the new value for the central pixel. Median filters can do an excellent job of rejecting certain types of noise, in particular, “shot” or impulse noise in which some individual pixels have extreme values. In the median filtering operation, the pixel values in the neighborhood window are ranked according to intensity, and the middle value (the median) becomes the output value for the pixel under evaluation [61].

The main advantages of the median filter:

- The median is a more robust average than the mean and so a single very unrepresentative pixel in a neighborhood will not affect the median value significantly. Since the median is less sensitive than the mean to extreme values (outliers), those extreme values are more effectively removed.
- Median filtering does not shift boundaries, as can happen with conventional smoothing filters (a contrast dependent problem).
- Since the median value must actually be the value of one of the pixels in the neighborhood, the median filter does not create new unrealistic pixel values. Thus the median filter is much better at preserving sharp edges than the mean filter. No reduction in contrast across steps, since output values available consist only of those present in the neighborhood (no averages).

In general, the median filter allows a great deal of high spatial frequency detail to pass while remaining very effective at removing noise on images where less than half of the pixels in a smoothing neighborhood have been effected.

4.2.4 Disparity

Once the gray scale images are converted into hexadecimal header files and noise filtering has been completed, the background subtraction is performed using the respective images [62]. The result represents the disparity between the two frames. This method of image subtraction eliminates the background and brings the target (region of interest) into focus, giving us information about its attributes such as shape, size and position. The accuracy in disparity obtained is high irrespective of the numbers of pixels being highly reduced during the input. Thus the number of computations performed is reduced highly with stability in accuracy obtained.

4.2.5 Adaptive Threshold: A Global Approach

To further enhance the resolution of the frame deviation, threshold for segmentation is implemented. The individual pixels in the grayscale image are marked as target pixels if their value is greater than nominal threshold value (initially set as 80) and as background pixels otherwise. In this case the pixel representing the target is given a value of "1" while a pixel representing the background is given a value of "0." In most applications it is required to set the value of threshold as a global parameter. But as scene complexities increase, it is difficult to obtain a unique value of threshold. Though an initial threshold is assumed, it may not be used globally for any kind of application.

Hence there is a necessity of making the threshold adaptive when the region of interest varies for different applications depending on the scene environment. In such cases, the adaptive threshold value is set by considering the mean or median value (if the object pixels are brighter than the background pixels). An iterative method has been implemented to obtain the adaptive threshold value depicted in Figure IV-5.

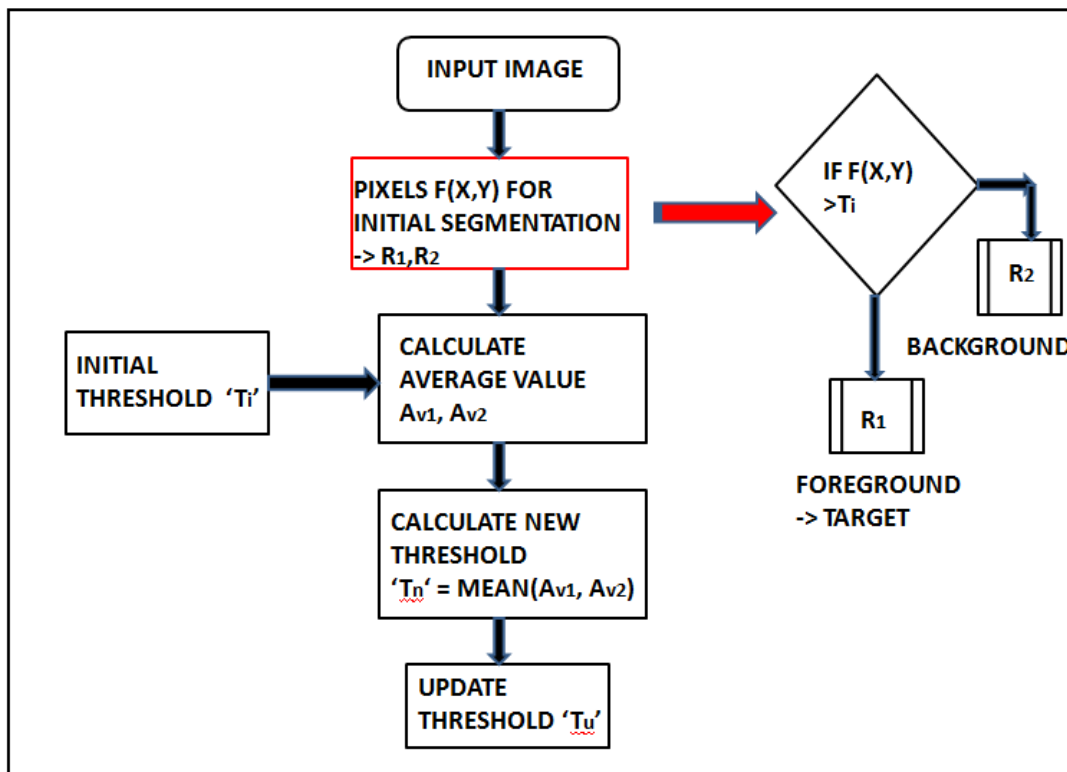


Figure IV-5 Frame generation and Header file creation

The algorithm is as described below.

1. Choose an initial random threshold value = ' T_1 '.
2. Initial segmentation is performed based on this value of threshold. Two sets of regions are generated represented by R_1 and R_2 :

$$R_1 = f(x, y) > T_i \quad \text{Equation IV-1}$$

Corresponding to the TARGET / FOREGROUND

$$R_2 = f(x, y) < T_i \quad \text{Equation IV-2}$$

Corresponding to the BACKGROUND

3. Each region average value is calculated represented by A_{v1} and A_{v2} .

$$A_{v1} = \text{AverageValueof}R_1 \quad \text{Equation IV-3}$$

$$A_{v2} = \text{AverageValueof}R_2 \quad \text{Equation IV-4}$$

4. The mean of the regional average value is the updated value of threshold represented by 'T_u'.

$$T_u = (A_{v1} + A_{v2}) / 2 \quad \text{Equation IV-5}$$

4.2.6 Target detection

The algorithm described above helps identify the object and gives us information about its attributes such as shape and size. In the case to track it, we can select the frames acquired from the video.

In state attributes include the Position, Size and Velocity of the object under consideration. After the background is subtracted, a binary image is formed with black representing the background and white representing the object/ target. The detected white pixels are bounded in a rectangle and the pixel location value is taken as the position of

the object under consideration. Since the position needs to be represented by a point source, the centre of the rectangle is used. From the image the size and position are estimated as shown. This is a clear indication of the 2D position of the object within that time frame. The iterative process is repeated over all the frames in the video file.

4.3 Designed System Architecture using MicroBlaze processor

Our FPGA-based development work has been based on hardware platforms using two different embedded CPUs. The Xilinx MicroBlaze is an established soft-core IP. It is based on a 32-bit RISC architecture and is a natural choice in smaller projects where CPU power is not essential. It is also the only choice for Xilinx FPGAs without an embedded hardcore CPU. The MicroBlaze can be run at speeds up to 66MHz and can be extended with additional coprocessors, instruction sets, and so forth. It shows the architecture used for MicroBlaze with our controller.

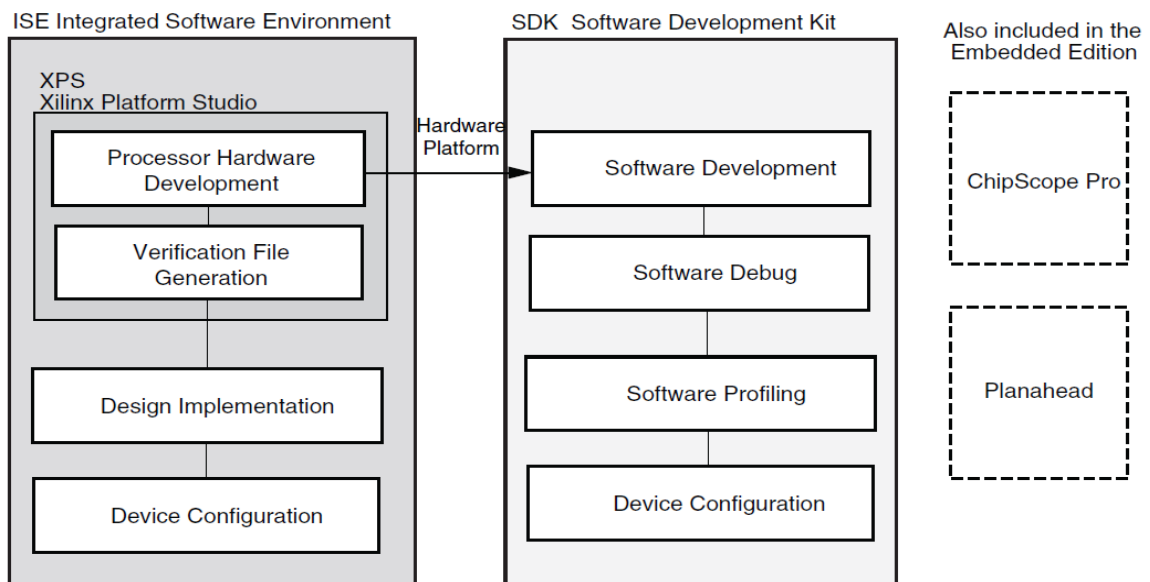


Figure IV-5 Xilinx ISE and SDK environment [63].

The MicroBlaze is a full 32-bit RISC CPU that is embedded in the Xilinx FPGA SPARTAN 3E and Virtex-5 FPGA families. It can be run at speeds up to 100MHz and is the best choice for CPU-intensive tasks in Xilinx FPGA based systems [63]. When developing embedded software we are always facing the host-target problem, meaning, we are developing the system on one machine, the host, and running it on another, the target. To solve this problem, we decided to also support the *x86* architecture from Intel. By doing so, it is possible to develop a large part of the system on an ordinary PC running Windows or any variant of UNIX. By simulating IP cores (firmware modules) as software objects, a system can be developed to an advanced state before it needs to be tested on the actual target [64]. Another benefit of this approach is that it allows concurrent development of multiple projects on a single target.

4.3.1 Development Tools

The FPGA / FPGA chip is supported with a complete set of software and hardware development tools, including Xilinx Embedded Development Kit (EDK) and Xilinx Platform Studio (XPS) tools development software shown in Figure IV-6 [65]. This tool is used to create a simple processor system and the process of adding a custom OPB peripheral (an 32-bit adder circuit) to that processor system by using the Import Peripheral Wizard. The microprocessors available for use in Xilinx Field Programmable Gate Arrays (FPGAs) with Xilinx EDK software tools can be broken down into two broad categories. There are soft-core microprocessors (MicroBlaze) and the hard-core embedded microprocessor (PowerPC).

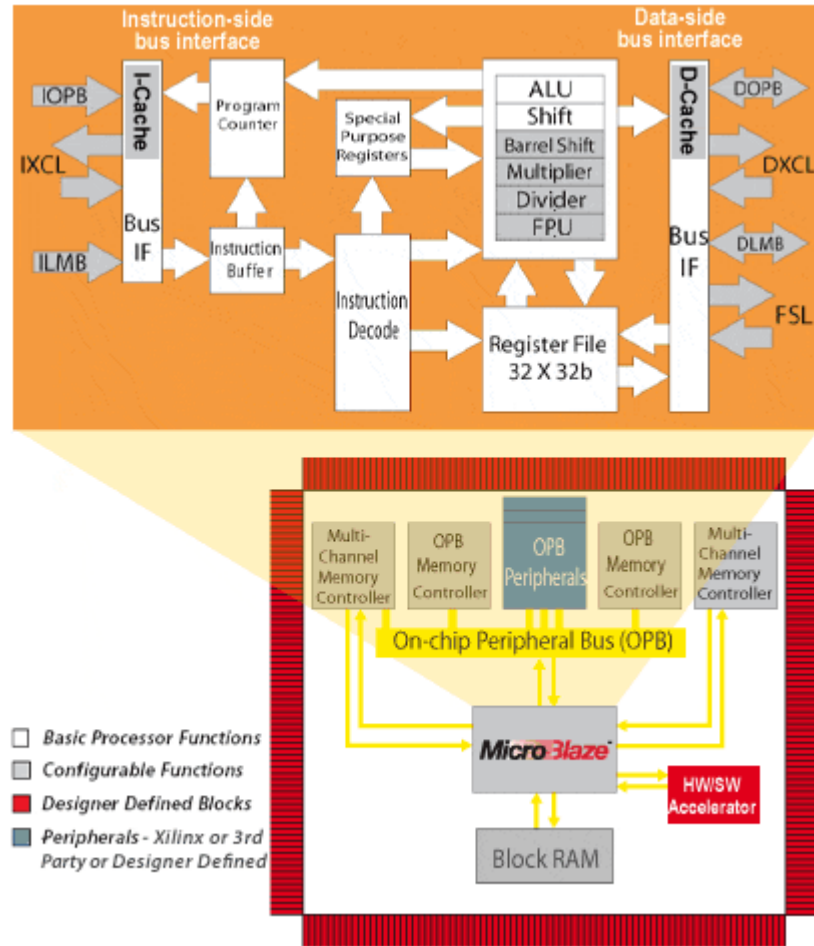


Figure IV-6 A view of MicroBlaze system.

The MicroBlaze is a virtual microprocessor that is built by combining blocks of code called cores inside a Xilinx Field Programmable Gate Array (FPGA). The MicroBlaze processor is a 32-bit Harvard Reduced Instruction Set Computer (RISC) architecture optimized for implementation in Xilinx FPGAs with separate 32-bit instruction and data buses running at full speed to execute programs and access data from both on-chip and external memory at the same time [66]. The items in white are the backbone of the MicroBlaze architecture while the items shaded gray are optional features available depending on the exact needs of the target embedded application as shown in Figure IV-7.

Because MicroBlaze is a soft-core microprocessor, any optional features not used will not be implemented and will not take up any of the FPGAs resources [67].

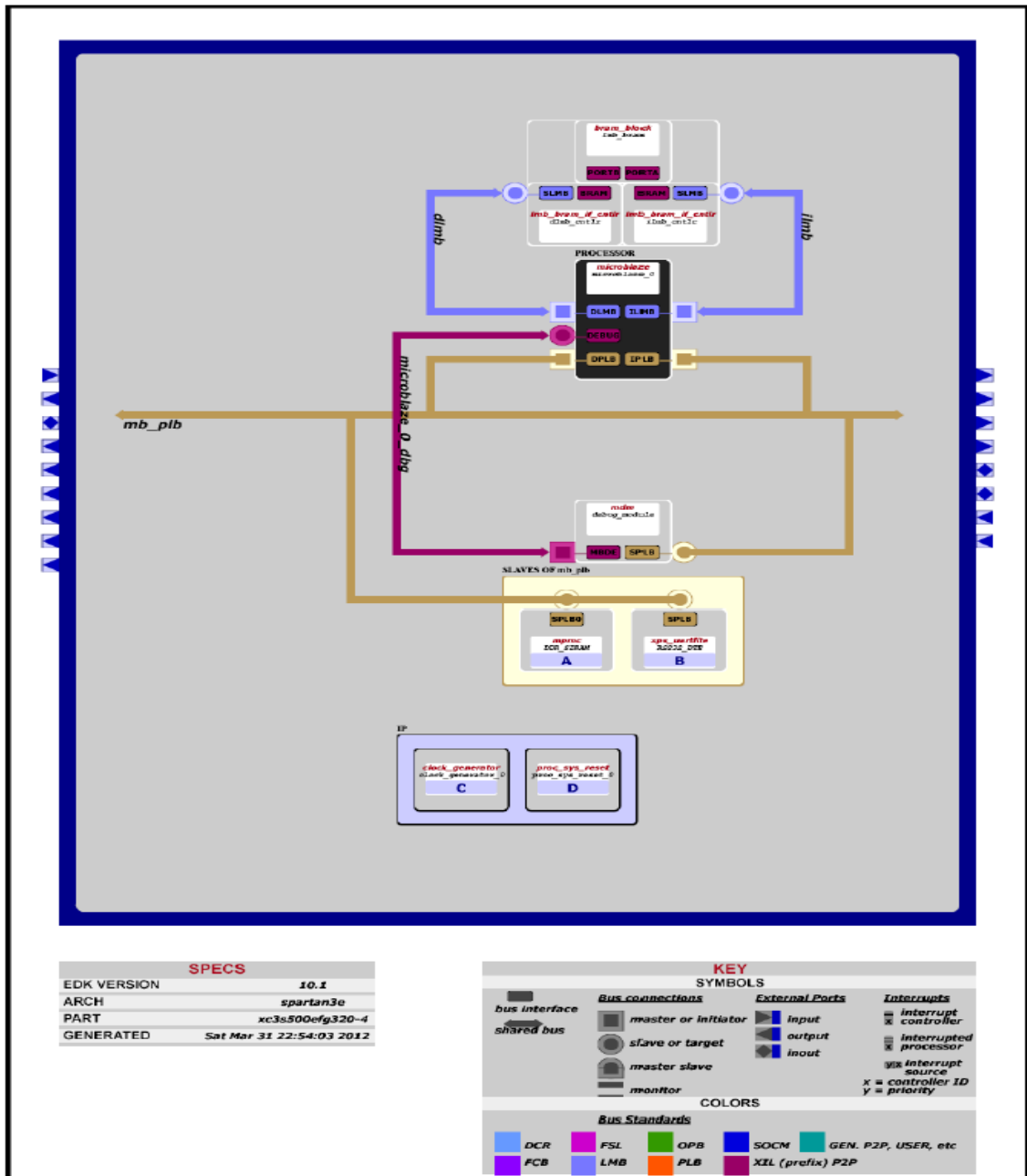


Figure IV-7 Block Diagram of Target Detection.

The Figure IV-8 below shows the block diagram of the system architecture developed for the target detection algorithm.

4.4 Output Display using SSH

The output of the frames is being displayed on to the PC using a visual basic 6.0 SSH active remote shell. The ActiveX provides asynchronous file transfers which means during transmission of a large file, the GUI does not block. A choice exists to wait until transfer is complete or to set an event which will be fired after the end of transmission. A sliding widow of 3x3 length buffer is used to display the results of the target detection algorithm. These sliding windows perform the edge detection after the background subtraction is being implemented. It detects the zeros and ones in the frame and provides an enhanced image to show the exact target detected.

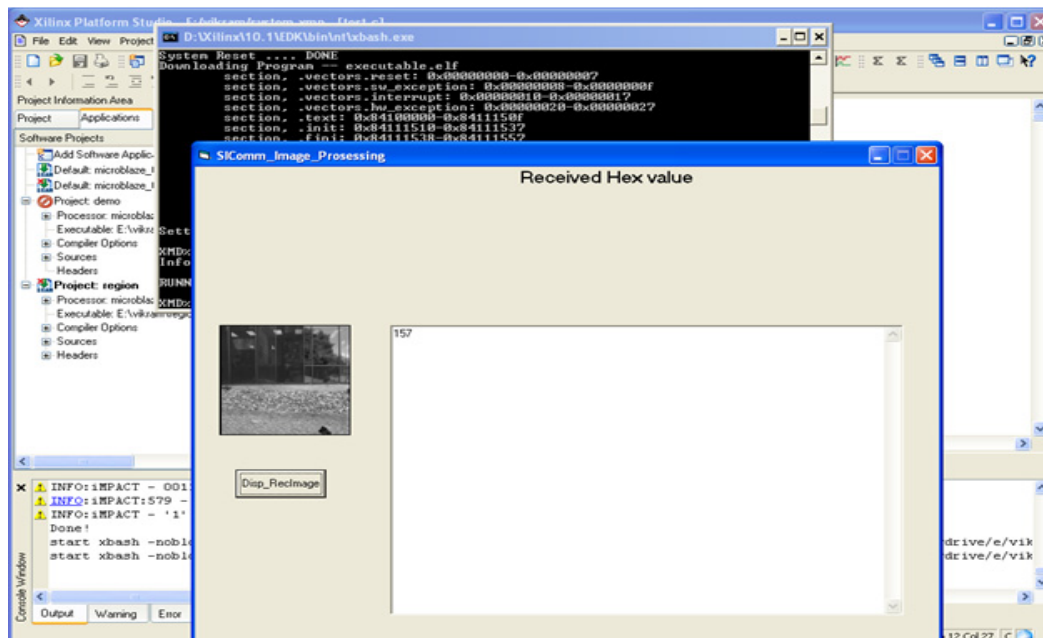


Figure IV-8 Visual basic SSH remote shell ActiveX control for communication.

4.5 Experimental Results

The system is designed to detect the target using the background subtraction. The computer used was a Pentium 4 with 1GB of RAM using Windows XP as operating system. To program and test the FPGA, we used Xilinx EDK 10.1 with MicroBlaze processor. It is highly important to take into account the limitations of the FPGA to be used in an application to avoid lack of memory, logical units or interfaces.

The design is simple and has the flexibility to choose the segmentation threshold value. This system is applicable to both indoor and outdoor environments. This target detection approach requires FPGA with minimum of DDR SRAM of 1MB. Since background subtraction forms the first step of any kind of image/ video processing algorithms, this implementation can be integrated to various applications like video surveillance, robot navigation, collision avoidance and path planning, gesture recognition and other high-level computer vision tasks. The method runs quickly, accurately and fits for the low power dissipation application. Figure 11 to Figure 15 show the output results of the FPGA based target detection architecture for various scenarios of environment, lightning and targets. Figure 12 displays the result of multiple targets in an indoor environment with different levels of illumination considered. The RAMT approach works efficiently for all the levels considered. Also the image frames used have a compression ratio of 75 and 18.75 respectively for 64x64 and 128x128 pixels resolution.

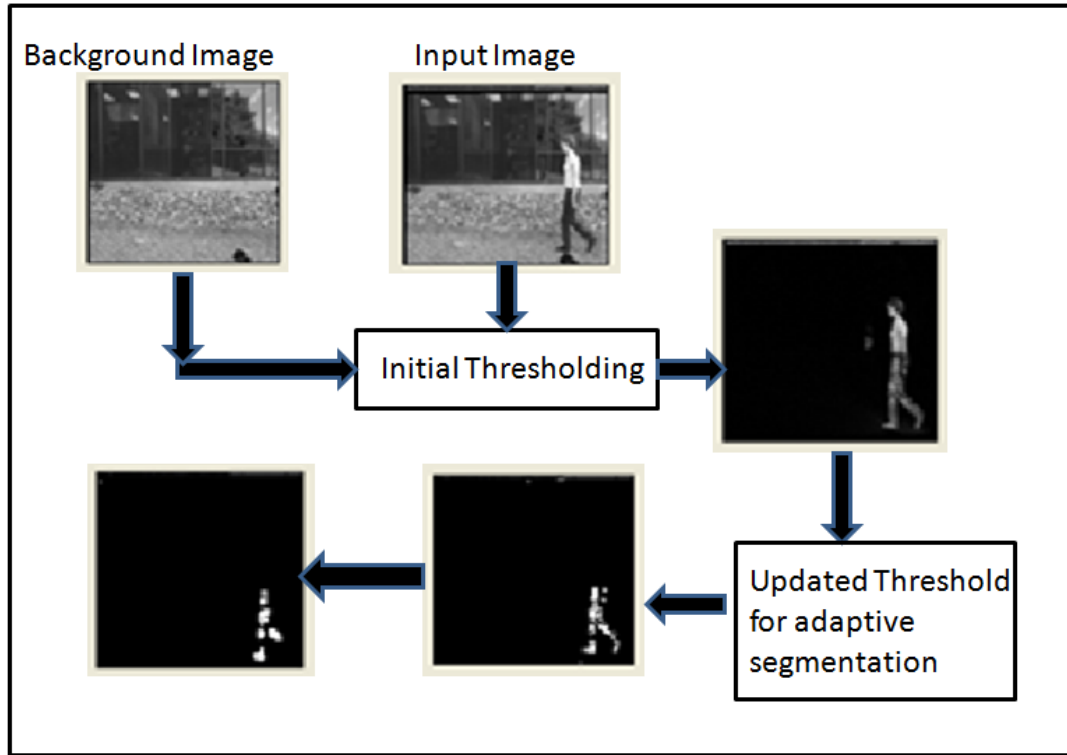


Figure IV-9 Case of Human in an Outdoor environment scenario.

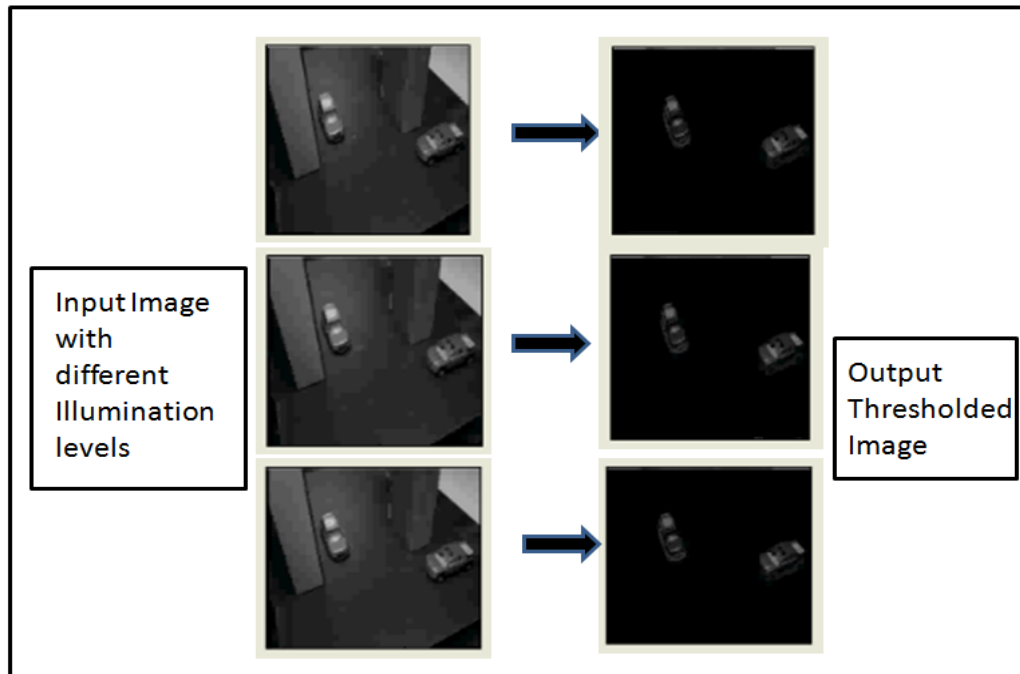


Figure IV-10 Case of Human in an Outdoor environment scenario.

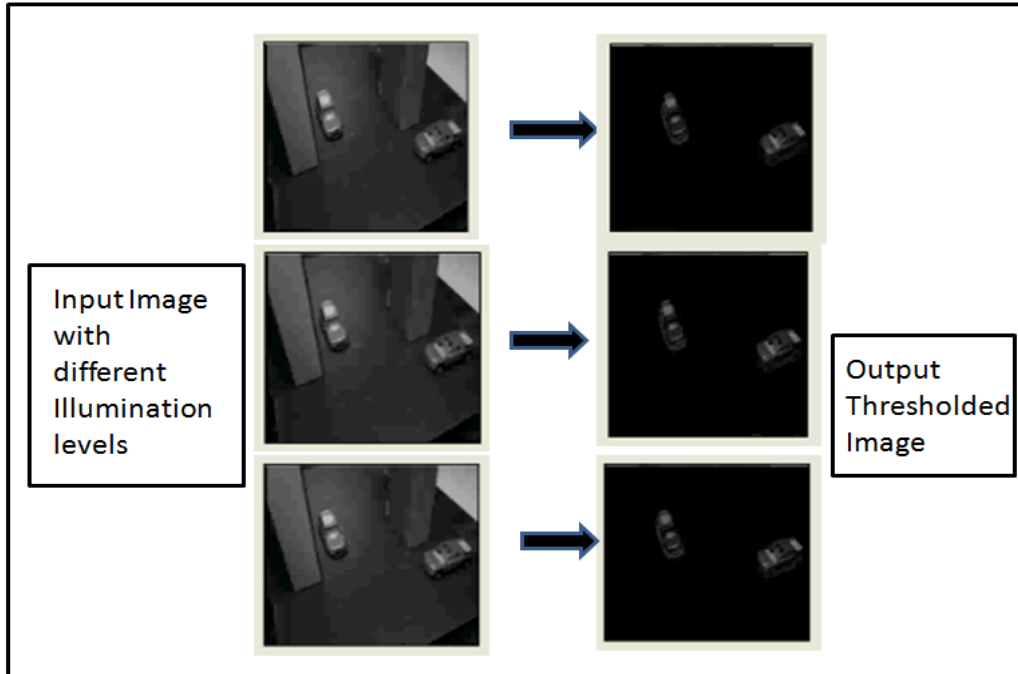


Figure IV-11 Case of the current frame with car as target for Indoor environment with various illumination levels.

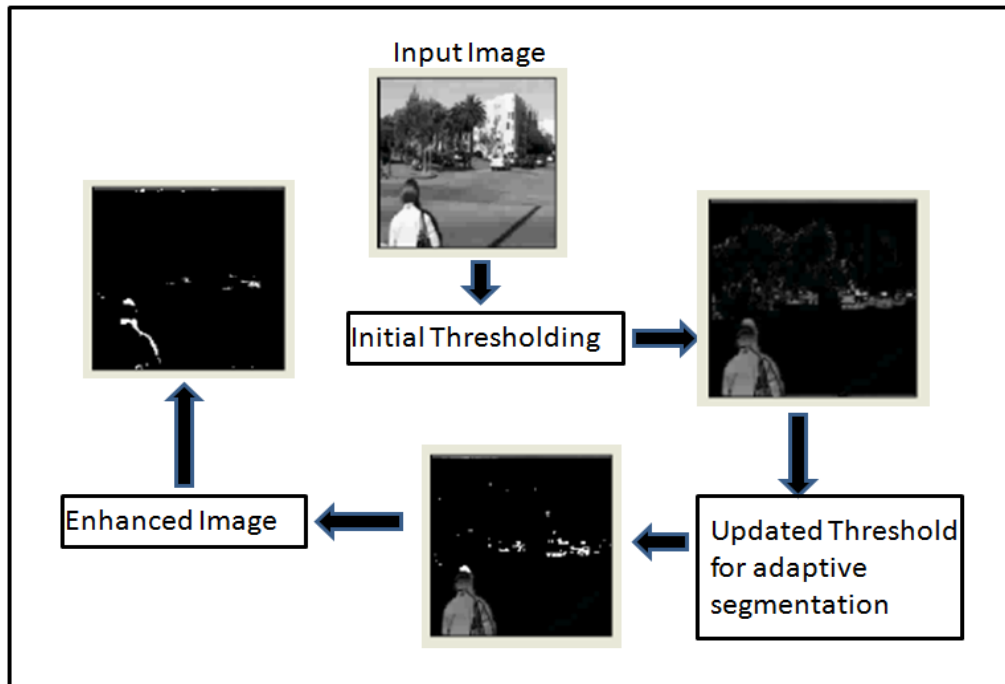


Figure IV-12 Case of Human in Traffic signal in Outdoor Environment.

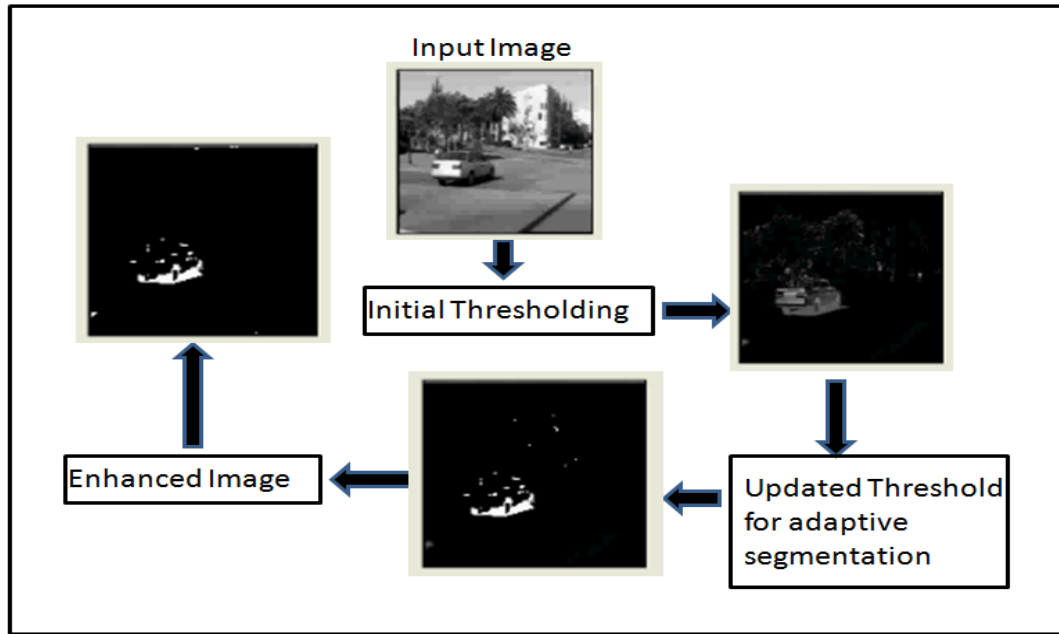


Figure IV-13 Case of Car in Traffic signal in Outdoor Environment.

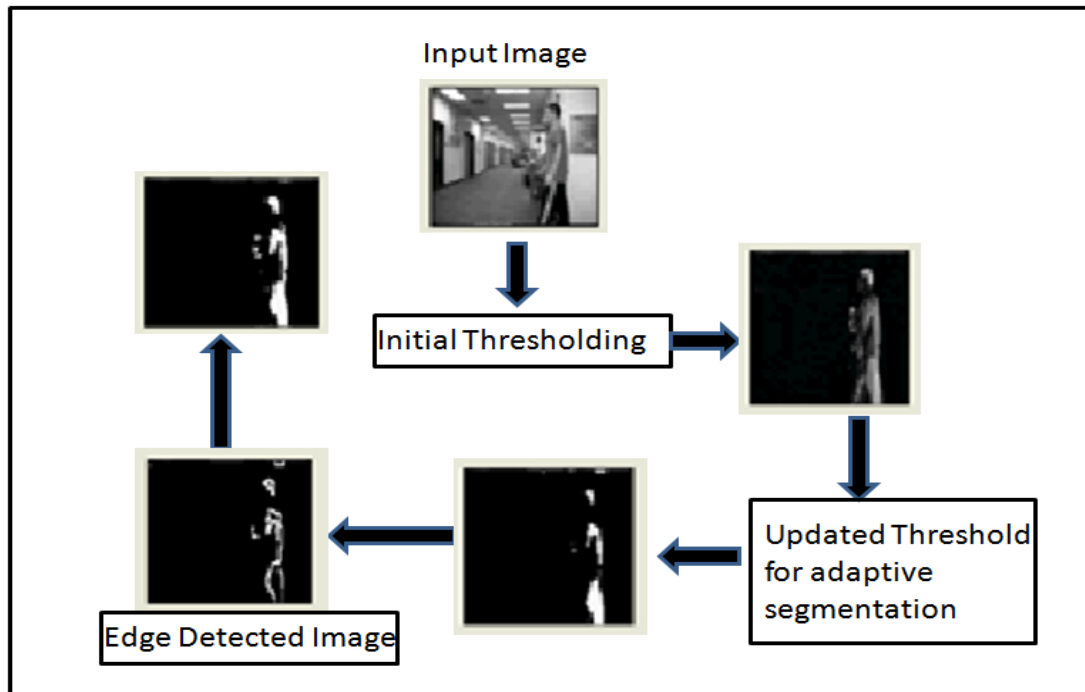


Figure IV-14 Case of Human in an Indoor environment scenario.

The algorithm is able to detect the targets successfully for the considered frames containing two environments (indoor/ outdoor) tabulated in the Table IV-3 shown below.

Table IV-3 Success of Target detection algorithm applied to different environments

Scene Environment	Human	Object	Target Detection
Indoor	Tested on 10 frames	Tested on 10 frames	Success
Outdoor	Tested on 10 frames	Tested on 10 frames	Success

Table IV-4 Device Utilization Summary

Device utilization summary:				

Selected Device : 3s500efg320-4				
Number of Slices:	2097	out of	4656	45%
Number of Slice Flip Flops:	2798	out of	9312	30%
Number of 4 input LUTs:	3024	out of	9312	32%
Number used as logic:	2493			
Number used as Shift registers:	211			
Number used as RAMs:	320			
Number of IOs:	49			
Number of bonded IOBs:	6	out of	232	2%
IOB Flip Flops:	39			
Number of BRAMs:	7	out of	20	35%
Number of MULT18X18SIOs:	3	out of	20	15%
Number of GCLKs:	7	out of	24	29%
Number of DCMs:	2	out of	4	50%

Table IV-5 Timing Summary

Timing Summary:

Speed Grade: -4
Minimum period: 12.384ns (Maximum Frequency: 80.749MHz)


```
Minimum input arrival time before clock: 41.553ns  
Maximum output required time after clock: 14.134ns  
Maximum combinational path delay: 3.344ns  
Total memory usage is 215260 kilobytes
```

A hardware prototype for the designed architecture is developed on a Xilinx Spartan-3E XC3S500E-4FG320C FPGA platform. Table IV-4, Table IV-5 indicate the module-level device utilization and timing results. The prototype is utilizing only around 32% of look-up tables (LUTs) and 30% of slice flip-flops from the whole FPGA resources and other than the embedded processor. The custom logic parts consume only 215260 KB (which is equal to 210.214844MB/ 0.205287933GB) of memory.

4.6 Chapter Summary

Chapter 4 focuses on the target recognition for automated surveillance monitoring, which is much more effective than manual surveillance monitoring. A new Running Average Mean Threshold (RAMT) algorithm is being derived to make the background subtraction (a major module of any image processing application) into a global approach rather than using a manually set threshold value. The design adapts the threshold automatically to different environments and different illumination levels to recognize the target in the scene. MicroBlaze processor is used to make it computationally efficient as compare to the other existing approaches that utilize only around 32% of look-up tables (LUTs), 30% of slice flip-flops and the output required time after clock is 14.134ns. SSH is used as a state-of-art to transfer the data and view directly on the host computer saving the cost of another external monitor for display.

V AUTOMATIC PATTERN RECOGNITION

The level of recognition in humans is very high with less effort even for a multitude of objects in images, despite the fact that the image of the objects may vary with respect to view angle, size, translation and rotation. A new scheme called Combination Neural Networks (CNN) recognition is being designed which uses parallelism between two units in the recognition systems. The layered structure introduction is a novel idea based on the concept of virtual address search of a CPU. Objects can even be recognized when they are partially obstructed from views of observers, i.e. partial occlusion due to the network topology of back propagation [15].

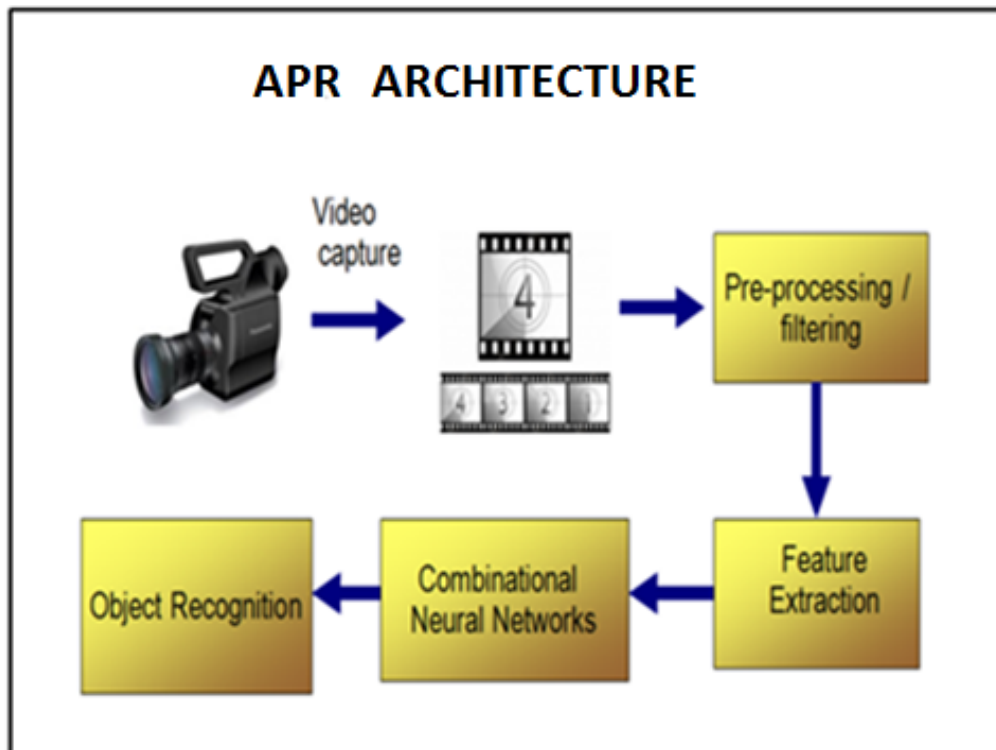


Figure V-1 Flow diagram of Automatic Pattern Recognition [11]

An “Automatic Pattern Recognition” (APR) system detects patterns (objects/ gestures/ face/ iris/ fingerprint) in the real world from an image or image sequence of the world, using different object models which are known a-priori. Humans can perform this task of recognition effortlessly and instantaneously. However, such a task of implementation based on machines is performed using algorithms and has been very difficult. In some applications, the patterns to be recognized are so fuzzy that they cannot be modeled with conventional tools. To solve these fuzzy logics, neural network engine can be used as the best tool. It is because that the processor can build a recognition system from simple image annotations made by the programmer. It then extracts characteristics or feature vectors from the annotated patterns and sends them to the neural network. Feature vectors characterizing visual objects can be as simple as raw pixel values, a histogram or distribution of intensities, intensity profiles along relevant axis or their gradients. More advanced features can include components from wavelet and Fast Fourier Transform (FFT). Neural networks are capable of generalization and can consequently classify situations never seen possible before by associating them to similar learned situations [15].

5.1 Pattern Recognition System Architecture

The architecture of APR consists of three main steps. Fig. 1 shows the APR system block diagram. Once the video is obtained, it is processed as a sequence of images. Processing raw images is a high memory constraint. While performing recognition using neural networks, scene constancy is also a major factor. Thus preprocessing and filtering is the first main block of the recognition system [71-72]. To the image is applied the

feature detector which identifies the locations of features that help in forming object hypotheses. Once the features are extracted, they are processed as inputs to the CNN to identify the object or groups of objects [20].

Preprocessing module design

The video obtained from the camera sensor is the raw video. To satisfy the memory requirements and the environmental scene conditions, preprocessing of the raw video content is highly important [73]. Various factors like illumination, background, camera parameters, and viewpoint or camera location are used to address the scene complexity. These scene conditions affect images of the same object dramatically. Under different scene conditions, the performance of different feature detectors will be significantly different. The nature of the background, existence of other objects (occlusion), and illumination must be considered to determine what kind of features can be efficiently and reliably detected [74-75].

Image is a scene captured by a camera sensor. The image may contain one or more objects and hence the objects are segregated from the background using the adaptive background subtraction. The preprocessing steps of an APR system consist of filtering and background subtraction using Mixture of Gaussian (MoG) [47] method. A rough example of the process described is being presented below as shown in Figure V- 2.

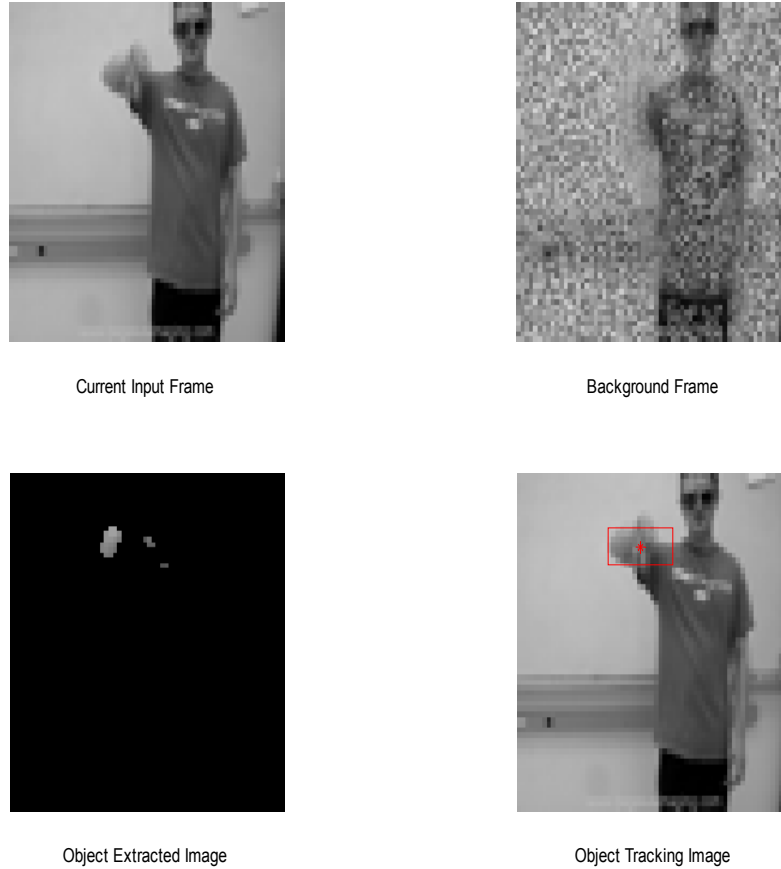


Figure V-2 Image after background subtraction and hand tracking

5.2 Designed Combinational Neural Network Architecture

Pattern recognition using neural networks is based on the learning of the network using a database set of patterns. The patterns differ depending on the application. In the case where the database is small, a sequential exhaustive search can solve the problem [76]. But as the applications grow and the necessity of universal database arrives, such sequential search algorithms fail to meet the timing and memory constraints. To reduce the size of search space a new search algorithm called combinational neural networks is

being developed in this paper. This architecture of CNN is based on the virtual address search concept of a CPU. The CNN is being described as follows. The structure resembles the tree structure architecture [11].

5.2.1 Dataflow

A parallelism exists between the feature extraction layer and the neural network layer. A dual bus is provided to allow the flow of data in both directions. The feature vector computation involves time and memory. In general, as the output classes are increased and the non-linearity of differentiation in the classes increases, more feature vectors are necessary for the object recognition technique. As a general practice, always the feature vector of the objects are derived all at a time and then sent as input to the first layer of network [11].

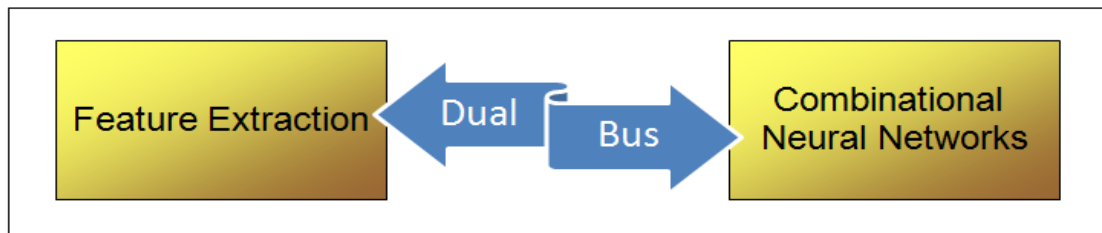


Figure V-3 Parallelism between Feature extraction and CNN tracking [11]



Figure V-4 Feature vector of n elements [11]

In our developed CNN, all the feature vectors are not being calculated at a time. Instead there exists a dual bus for communication between the feature extraction layer and the object recognition layer as shown in Figure V-3 below.

5.2.2 Network Topology

Back propagation, the three layer network, as described in [22] is used as the network topology. The network layer consists of 3 stages: stage 1, stage 2, and stage 3 as shown in Figure V-5. Each stage acts as a back propagation neural network layer that takes the elements of feature vector as input and outputs the class of object. Each stage receives their input elements from the feature extraction layer.

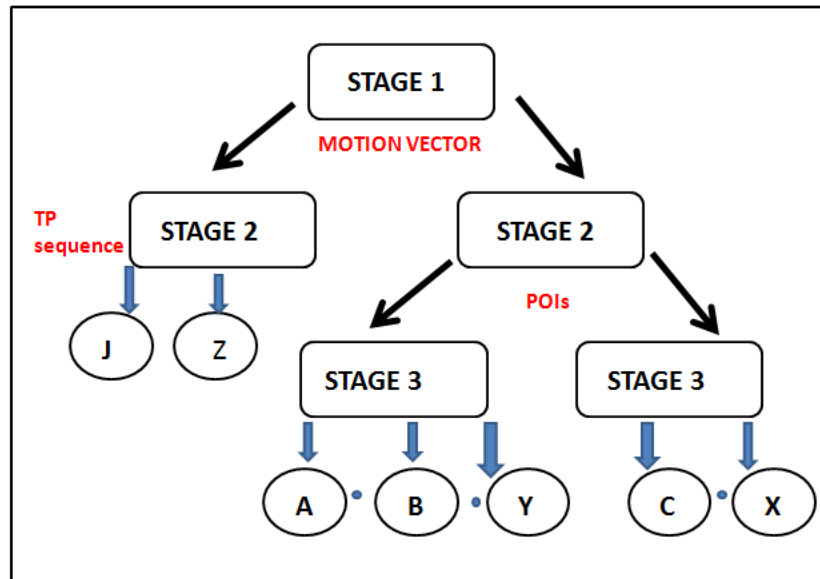


Figure V-5 CNN network layers

Stage 1 uses the motion vector sequence to output whether the sign has a motion or not. The output triggers the next stage of the CNN that is stage 2. The stage 2 uses the point of interests as the inputs to classify how the fingertips are oriented. The output of stage 2

triggers the stage 3 which then classifies the signs based on gradient threshold approach features [11].

5.3 Feature Extraction Logistics

A feature is an attribute of the object that is considered important in describing and recognizing the object in relation to other objects. Generally speaking, the pattern in the 2D image refers to a real 3D pattern in real world, which affects the recognition approach. The commonly used basic features are size, color, and shape. It is being chosen from a qualitative or functional description to precise geometric surface information.

5.3.1 Sign/ Gesture – SM Rule

Features are divided as two sub-categories: hand shape and hand movement. The state of the hand gestures are given by the attributes called Point of Interest (POI) of the hands. We consider two POIs to represent the ‘shape’ and ‘direction of movement’.

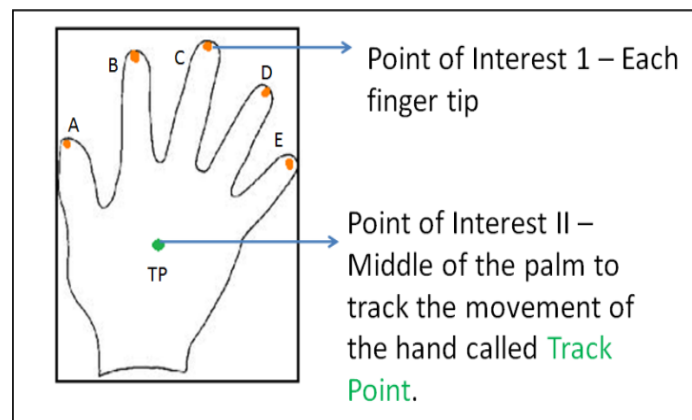


Figure V-6 Point of Interest I (Each fingertip of the hand) and II (midpoint of the palm) – POIs [12]

The finger tips are referred to as A, B, C, D, E points and the midpoint of the hand as Track-point (TP) as shown in Figure V-6. The motion vector of the TP indicates the direction of motion as an angle lying in one of the cycles of the angle chart as shown in Figure V-7.

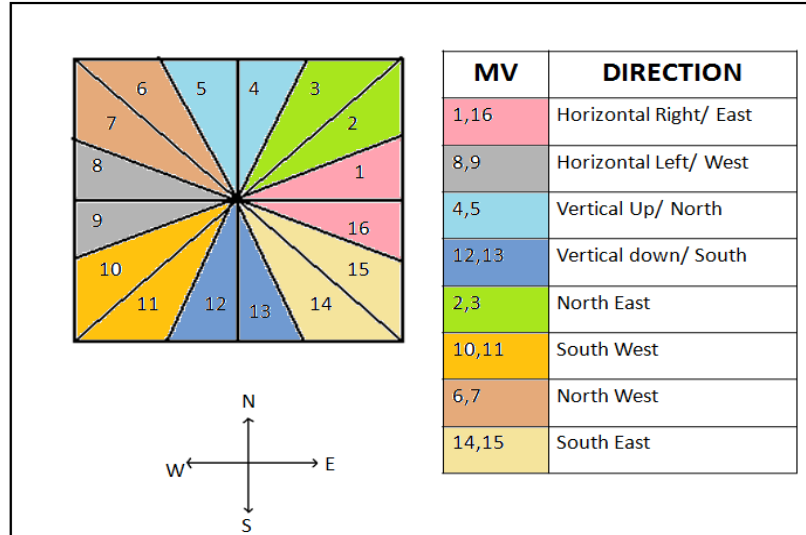


Figure V-7 Angle chart – Motion Vector related to direction [12]

The Table V-1 describes the relation between the Motion Vector (MV) direction and the angle chart. Since the angle chart is divided into 16 slots, the motion vector is categorized as one of them. Four bits are used to represent the MV since 16 intervals.

The feature vector consists of 55 features. These include 5- finger tip elements (A, B, C, D, E), 4- motion vector elements; 6 elements are the MV sequence; and the remaining 40 are from the gradient threshold approach of the image calculated [21, 25, 35].

Table V-1 Motion Vector Direction

Vector Direction from Angle Chart	Angle	Quadrant
1,2,3,4	0° – 90°	I
5,6,7,8	90° – 180°	II
9,10,11,12	180° – 270°	III
13,14,15,16	270° – 360°	IV

The gradient threshold algorithm depends on the feature vector used in pattern recognition systems by R.K. McConnell. This feature vector forms a histogram based on the orientation of edges in an image. The vector represents the local features of the image.

$$dx = I(x, y) - I(x + 1, y) \quad \text{Equation V-1}$$

$$dy = I(x, y) - I(x, Y + 1) \quad \text{Equation V-2}$$

$$Gdirection = \arctan(dx, dy) \quad \text{Equation V-3}$$

$$Gmagnitude = \sqrt{(dx * dx + dy * dy)} \quad \text{Equation V-4}$$

The flowchart describing the gradient threshold approach is given in Figure V-8 below. The gradients are calculated using Equation V-1 and V-2. The direction and magnitude of the gradients are calculated using Equations V-3 and V-4 [21, 25, and 35]. In the gradient, 360 degree is grouped in 40 groups, each of which is 9 degree.

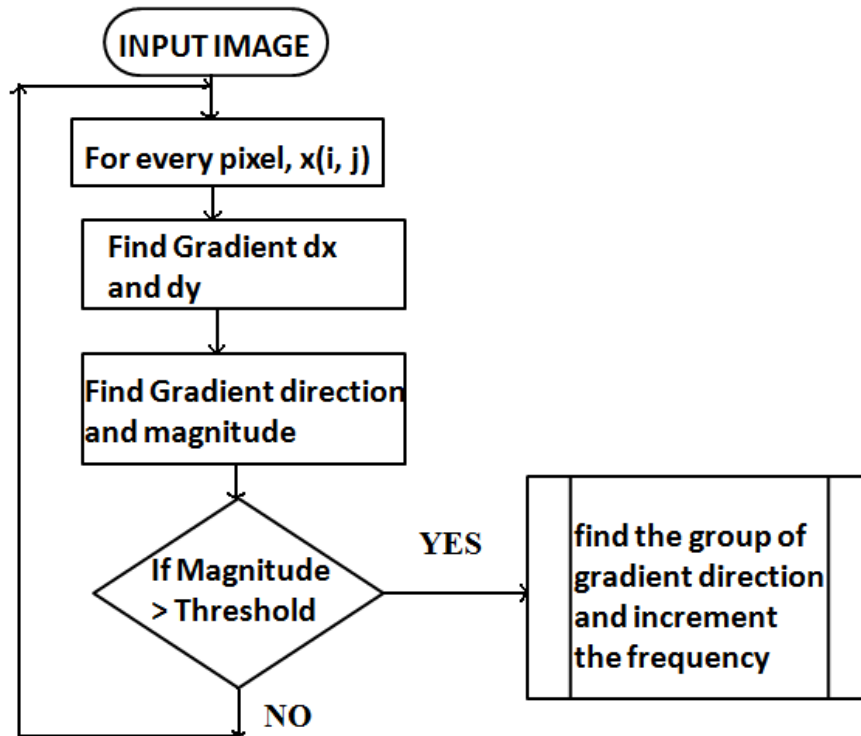


Figure V-8 Gradient threshold algorithm of an input image- 40 features of feature vector

The MV sequence of each gesture that is trained is unique. Every gesture has a corresponding stored model in the neural network model. Recognition of gesture by gesture is based on a stage where it reaches minimum energy at the end of a gesture. The number of key frames in one gesture varies from man to man due to difference in their speed of action. But the motion vector acts the same way for a unique gesture.

5.4 Experimental Results

The system is designed to recognize simple gestures or signs. The design is very simple and does not require any kind of gloves to be worn. This system is applicable to

different backgrounds. This sign language recognition approach requires a computer with at least 1GHz processor and at least 256 MB of free RAM. The training set consists of all alphabets A to Z (26 patterns). Figure 6 shows the sign language representation of English alphabets. The letters J and Z involve motion and hence the motion vector is needed to recognize them. All the other letters have 0 motion vectors sequence before they encounter the lowest energy frame of the gesture.

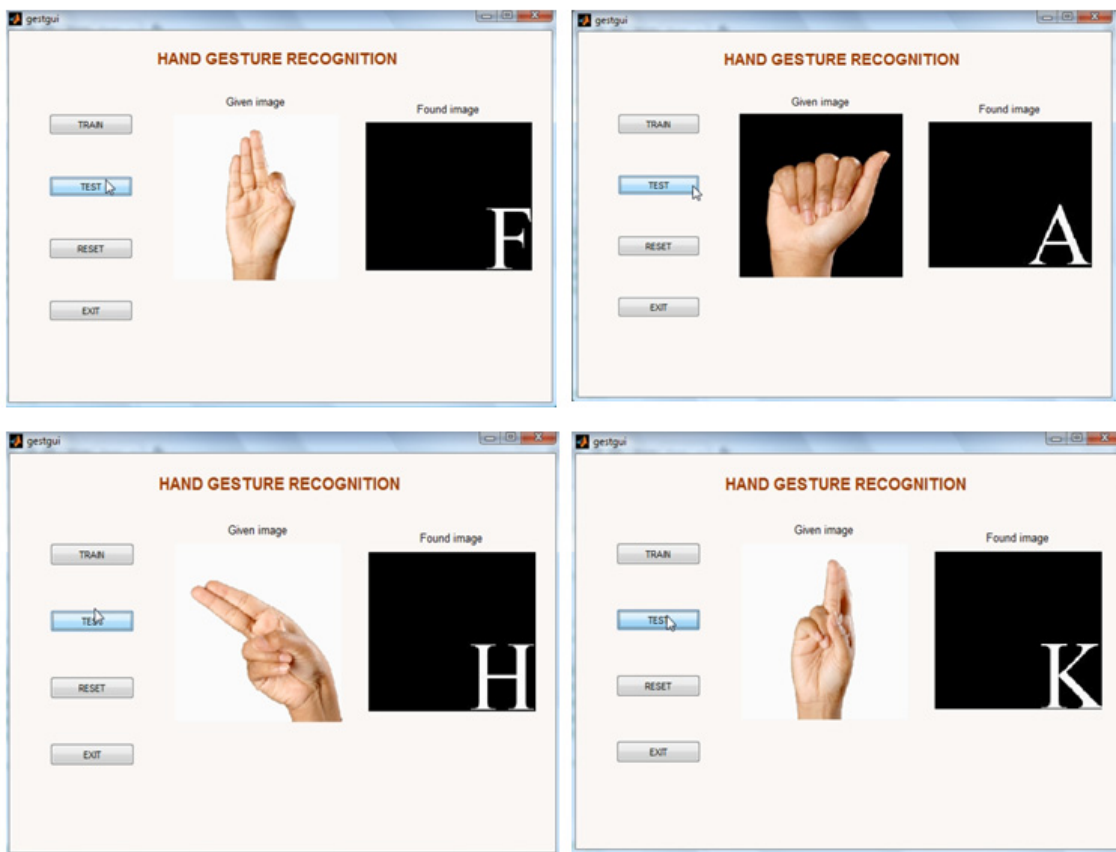


Figure V-9 MATLAB results of the input image and letter recognized

The Figure V-9 represents the output table of the algorithm performed using the MATLAB. The algorithm is able to detect all the alphabets from A to Z with 100% recognition rate for any of the training set pattern.

Table V-2 Noise Immunity of Signs

Alphabet	Noise Immunity	Average Noise Immunity	Standard Deviation
A	48%	-	-
G	30%	-	-
P	37%	-	-
Y	52%	-	-
Results		48%	10.0789

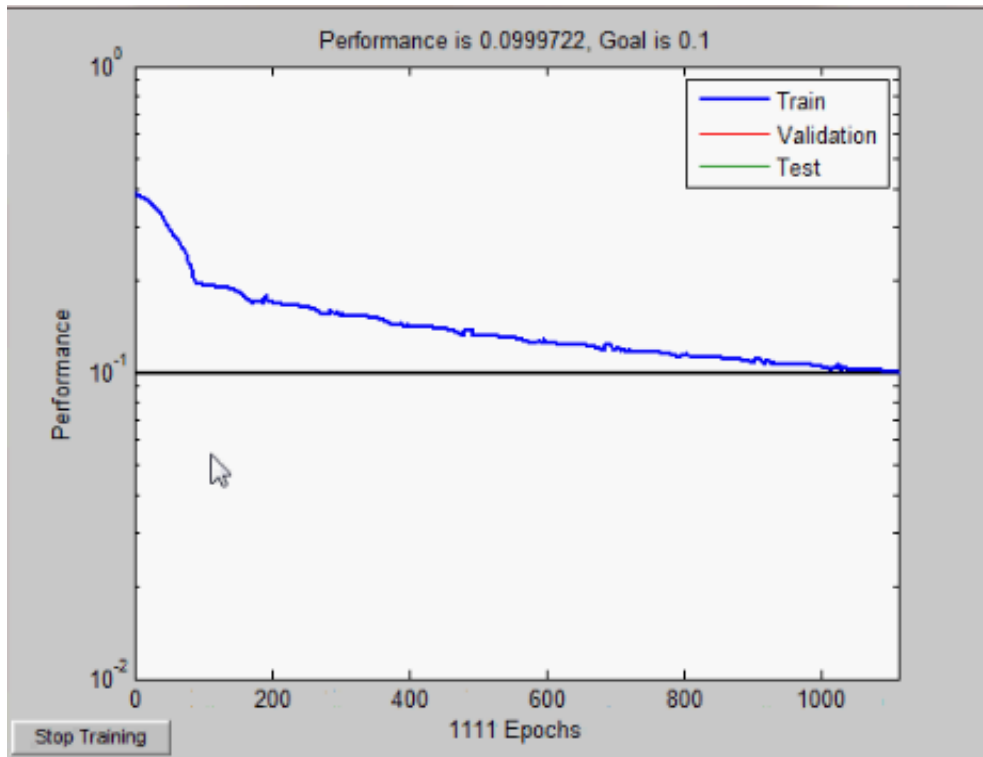


Figure V-10 Mean square error

In case of noise immunity, the experimental results that are shown in Table V-2 reveal that the algorithm is error free up to an average of 48%. It implies that in the case of noise corruption which results in loss of the feature vector elements of the input image, the gesture is detected error free [12]. Figure V-10 also shows the mean square error of the network during training.

Noise corruption may include noises due to environment or loss of information due to fading, blur or damaged. The next phase involves applying the algorithm to simple words of sign language gestures. The output of the algorithm can be connected to a 5x3 LED pixel array to indicate the letters. Here in the figure it can be observed that we have used a 5x3 array to represent the output.

5.5 Chapter Summary

Chapter 5 focuses on a novel architecture of CNN applied to pattern recognition systems. Many multiple features are extracted and used for the identification in two stages. By increasing the stages, the computation time for both training and recognition is decreased compared to the existing topologies. The system performance is good even under the conditions of partial occlusion. The signs for all the alphabets from A to Z are being recognized using the combinational neural networks architecture. Since it calculates the direction of edges, it is robust to illumination changes. Even light condition changes, the edges are about the same. The place of the object in the image is not important in calculation of feature vector. Therefore, translation of object in the image does not change the pattern to be recognized. The advantage of using the algorithm is

high processing speed which can produce results in real-time. It is also advantageous as even noise corrupted almost up to 50%; the signs can still be retrieved. For future research processing of words and sentence gestures can be included.

VI GESTURE RECOGNITION ON HW/SW CO-SIMULATION PLATFORM

6.1 Hardware/Software Co-simulation

Hardware/Software (HW/SW) co-simulation integrates software simulation and hardware simulation simultaneously. Usually, HW/SW co-simulation platform is used to ease debugging and verification for VLSI design. To accelerate the computation of the gesture recognition technique, a HW/SW implementation using FPGA technology is presented in this paper. The memory controller is designed in the VERILOG HDL. The testing part of the neural network algorithm is being hardwired to improve the speed and performance. The American Sign Language gesture recognition is chosen to verify the performance of the approach. Several experiments were carried out on four databases of the gestures (alphabet sign A to Z). The major benefit of this design is that it takes only few milliseconds to recognize the hand gesture which makes it computationally efficient as compare to the other existing approaches.

6.2 Introduction

In today's world, the FPGA technology has advanced enough to model complex chips replacing custom application-specific integrated circuits (ASICs) and processors for signal processing and control applications. FPGAs are preferred as higher-level tools evolve to deliver the benefits of reprogrammable silicon to engineers and scientists at all levels of expertise. Taking advantage from the current FPGA technology, this paper

proposes a hardware/software co-simulation methodology using HDL simulations on FPGA as an effort to accelerate the simulation time and performance [81, 94].

The conventional software simulation method has more flexibility in terms of parameters variation. The desired simulation parameters can be changed to study the system behavior under various conditions. The major drawback with the conventional approach is the intolerable simulation time. On the other hand, the complete hardware based approach can provide significant speedup in examining the system behavior but the flexibility will be nonetheless compromised. In this paper, we attempt to leverage the merits of the software simulation and hardware emulation to retain both the flexibility and performance by adopting a hardware/software based platform approach [81-82].

The use of sign language plays an important role in the means of communication method for the hearing impaired community [23]. American Sign Language is the 3rd most-used language and the choice for most deaf people in the United States. 500,000 and 2,000,000 people use Sign Language as their major daily communication tool. It seems that 3.68% of the total population is found to be hard of hearing and 0.3% of the total population is functionally Deaf, out of a total population of about 268,000,000 (2005) in the US [12]. Several methods have been developed previously to recognize the gestures. Gesture recognition was generally classified based on three different approaches. Primarily, glove based analysis was used in which either mechanical or optical sensors were attached to a glove for determining the hand posture. Currently the vision based analysis is used mostly which deals with the way human beings perceive

information about their surroundings. In this paper we have used the vision based approach for our gesture recognition application.

Many novel approaches have been developed previously which include the colored glove based method, skin color segmentation, video sequence appearance modeling and Hidden Markov Model (HMM) systems for gesture recognition. Ko and yang developed a finger mouse that enables an user to specify commands with the fingers as in [26]. Kjeldssen and Kender [83] and Hongo et al. [84] use skin color segmentation techniques and recognize gestures from the segmented hand images using back propagation and linear discriminant analysis techniques. Huang and Huang [85] proposed a system consisting of model based hand tracking, feature extraction using the scale and rotation invariant Fourier descriptors and recognition using a 3D Hopfield neural network. Huang and Jeng [86] suggest a model based recognition system using Principal Component Analysis (PCA), the Hidden Markov Model (HMM) and Viterbi algorithm. A computer vision system is proposed by Triesch and Von der Malsburg [87] that is based on Elastic Graph Matching, which is extended to allow combinations of different feature types at the graph nodes. Chen et al. [88] introduce a hand gesture recognition system to recognize continuous gesture before stationary background using HMM. Wachs et al. [89] identify static hand gesture poses by using Haar-like features to represent the shape of the hand. A new technique for shape-based hand recognition is proposed by Yoruk et al. [90] using color segmentation technique.

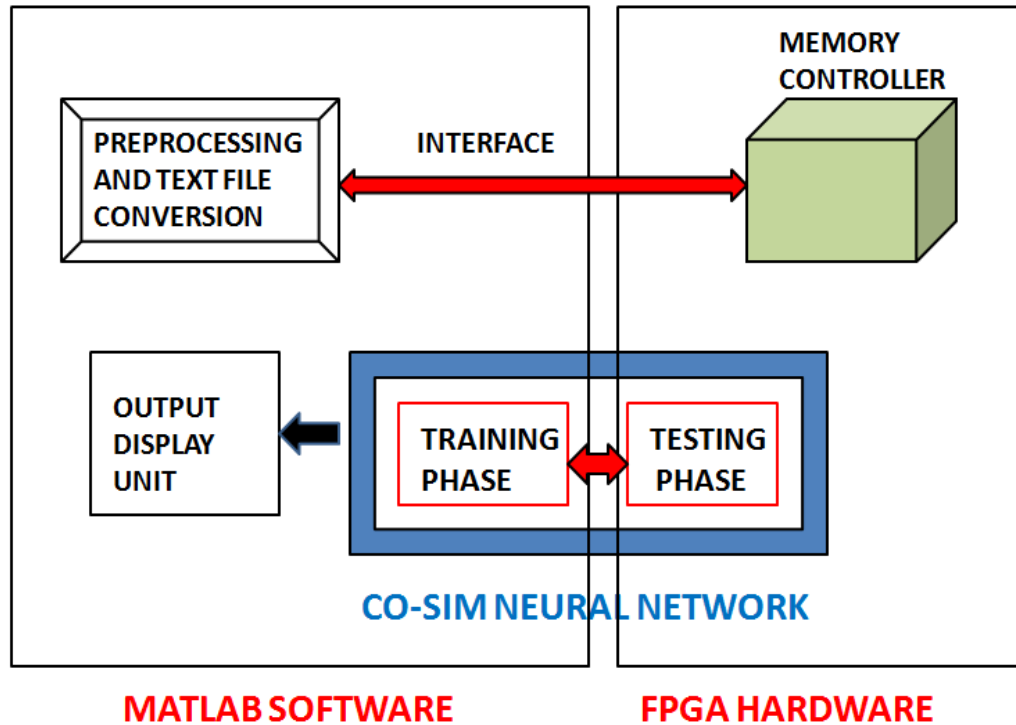


Figure VII-1 System architecture of Gesture recognition on hardware/ software co-simulation platform.

Figure VI-1 shows the system architecture being developed in this paper. MATLAB is used as the software platform and MODELSIM - ALTERA 6.3g_p1 is used as the hardware platform. QUARTUS II (8.1 web edition) design flow is used to simulate and verify the functionality of HDL code. Xilinx ISE 10.1 is used to understand the device and logic utilization, memory design and test control of the architecture developed.

6.3 Benefits of FPGA

Performance based on speed and power – FPGAs have the advantage of hardware parallelism performing more like concurrent execution of functions. Thus by breaking the

paradigm of sequential execution and accomplishing more per clock cycle the FPGAs exceed the computing power of digital signal processors (DSPs). Mostly inputs and outputs (I/O) of the applications are controlled at the hardware level to provide faster response times and dedicated parts of hardware reduce the power consumption for few complex operations.

Low Cost design - The nonrecurring engineering (NRE) expense of custom ASIC design far exceeds that of FPGA-based hardware solutions. The programmable silicon indicates no fabrication costs or long lead times for assembly. Due to changes in system requirements with time, the cost of making incremental changes to FPGA designs is negligible as compared to the large expense of an ASIC.

Reliability and maintenance – In general the software tools provide the programming environment where as the FPGA based design is completely hardware implementation of the application. Processor-based systems often involve several layers of abstraction to help schedule tasks and share resources between multiple processes. The driver layer controls hardware resources and the OS manages memory and processor bandwidth. For any given processor core, only a single instruction is executed at a time, and processor-based systems are continuously at risk of time-critical tasks preempting each other. FPGAs minimize the reliability concerns with true concurrent execution and dedicated hardware since they do not use the operating system. The growing availability of high-level software tools decrease the learning curve with layers of abstraction and often offers valuable IP cores (prebuilt functions) for advanced control and signal processing. [91]



Figure VI-2 American Sign Language Database.

6.4 Gesture Recognition Algorithm

6.4.1 Database Generation

The images for database are captured using a Cannon camera which produces image frames of RGB pixels. The American Sign Language (ASL) alphabet gestures are used to obtain the hand images. The database contains two different subjects with two different backgrounds. Figure VI-2 shows few of the database sets used in this particular application. The Cannon camera is actually not stationary since it is not in a fixed position but the background is maintained as either black or white. The images are stored in .jpg format. The temporal resolution requirements of the application have to be considered. The size of the image frame is 640x480 pixels. To maintain the resolution and decrease redundancy, the frames are resized to 64x64 pixels. Once the input images are obtained they are converted to grayscale values within a range of 0-255.

The database obtained is read in the computer using MATLAB. The software converts the entire database image frames into text files. These files are stored in the memory of the ALTERA MODELSIM using command "\$readmemh". The Altera Modelsim is being called from the MATLAB using HDLDAEMON as shown in Figure VI-3 below. HDLDAEMON controls the server that supports interactions with HDL simulators. To communicate with HDL simulators, the server uses one of two inter process communication methods: shared memory (which is the default), or TCP/IP sockets. Communication by shared memory is usually faster than sockets. When using shared memory, the server can communicate with only one HDL simulator at a time, and the HDL simulator must be running on the same host. When using sockets, the server

can communicate with multiple HDL simulators simultaneously, running on the same host or other hosts [92].

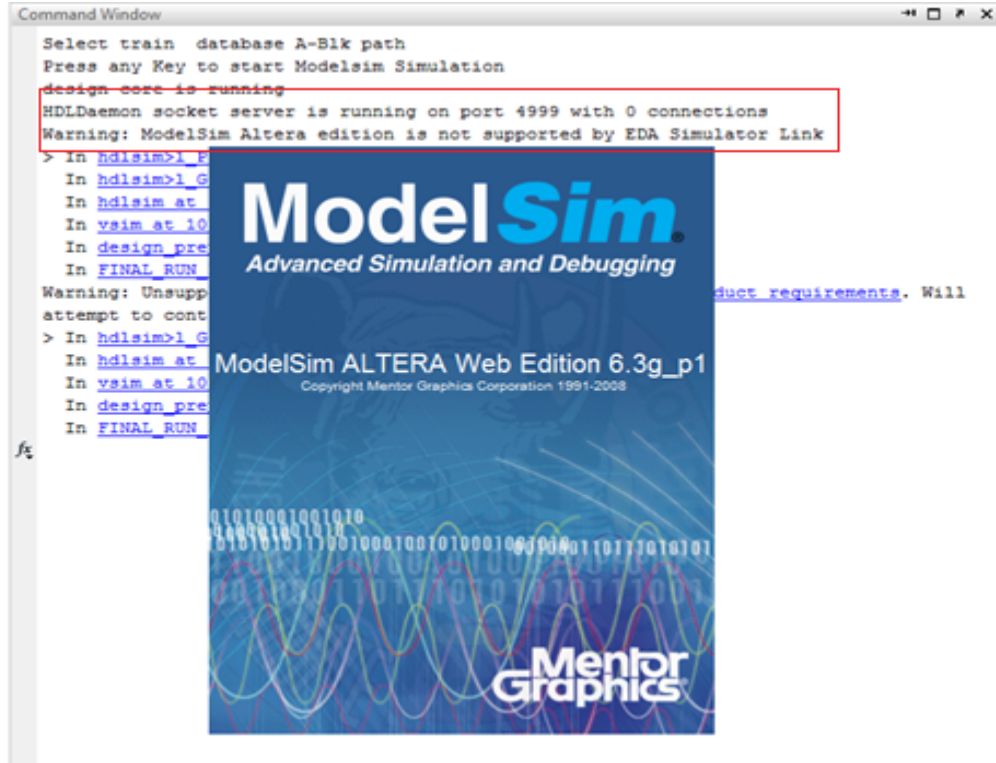


Figure VI-2 HDLDAEMON as the interface between MATLAB and ALTERA MODELSIM.

6.4.2 Text file Generation

The grayscale images are converted into text files which contain the hexadecimal value of the pixels. Each image is configured and stored with the appropriate configuration ID. The text file contains 4096 values to be read and stored on to the memory of the Altera Modelsim.

6.4.3 Memory Controller on Altera ModelSim

The text files obtained from the image files are stored onto another file in the memory locations of the Altera Modelsim. The files contain data represented by hexadecimal values and hence contain 16 digits of length. For every negative edge of the clock cycle, the data is read into the memory location. The Figure VI-4 displays the simulation waveform of the memory controller for the database sets. The value *test_design/k* shows the database set images ranging from 1 to N (the last image depending on number of database sets considered).

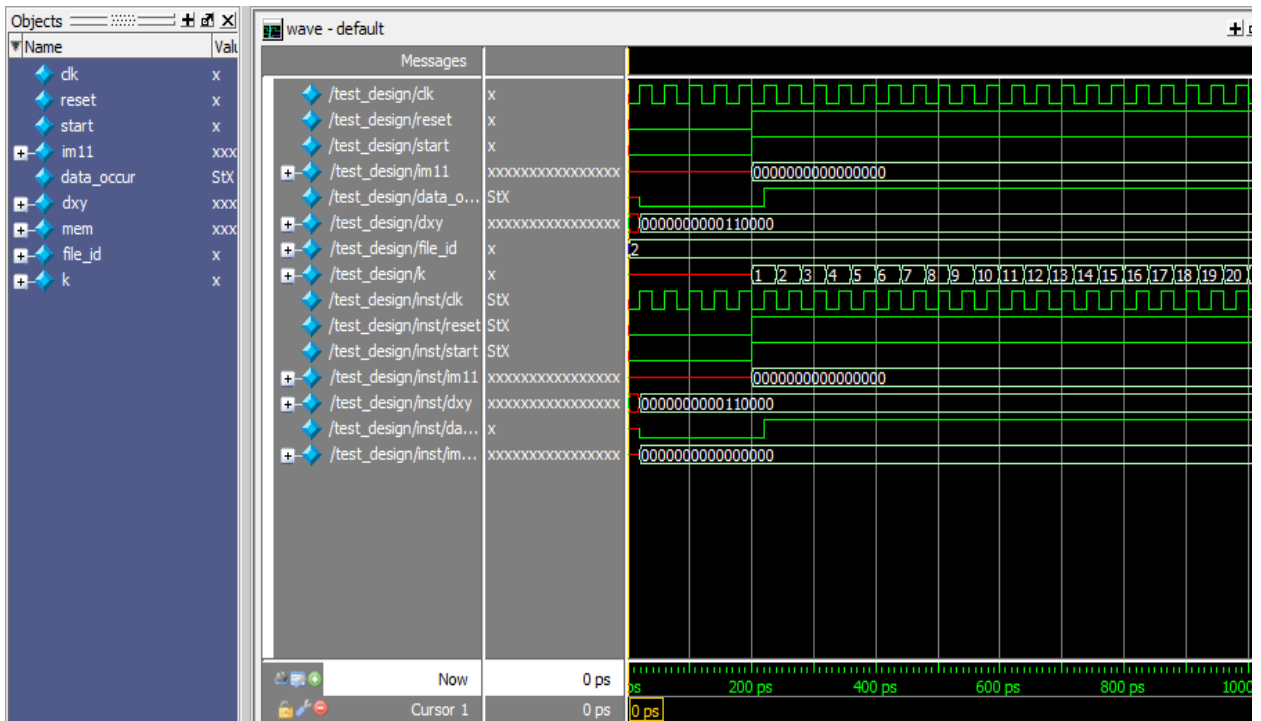


Figure VI-3 Simulation output waveform of Memory design controller.

6.5 Co-simulation Neural Network

Neural networks are based on the parallel architecture of neurons present in human brain. It can be defined as a multiprocessor system with very high degree of interconnections and adaptive interaction between the elements. The choice of neural networks to recognize the gestures automatically is due to the following aspects like adaptive learning (using a set of predefined database sets), self-organization from the training module, real-time operation with parallel computations and high fault tolerance capability [72, 38].

This paper focuses on recognizing static hand gesture images. Since static hand postures not only can express some concepts, but also can act as special transition states in temporal gestures recognition, thus estimating static hand postures play an important role in gesture recognition applications. A gesture recognition system takes an image as an input, processes it using a transform that converts the image into a feature vector, which will then be compared with the feature vectors of a training set of gestures. A new technique called co-simulation neural network is being adopted. In this method a part of the neural network is designed on the hardware with dedicated ports. An interface is being introduced among different levels of the neural network to communicate with one another on two different platforms. A simple neural network model is shown in Figure VI-5 which consists of input layer, hidden layers and the output layer with different number of neurons (R , S) in each [22]. Our network is built of 16 input neurons in the input layer, 50 neurons in the first hidden layer, 50 neurons in second hidden layer and 35

neurons in the output layer. The output is displayed as a visual representation of the gesture image and hence is a 7x5 (35 neurons) grid display of rows and columns.

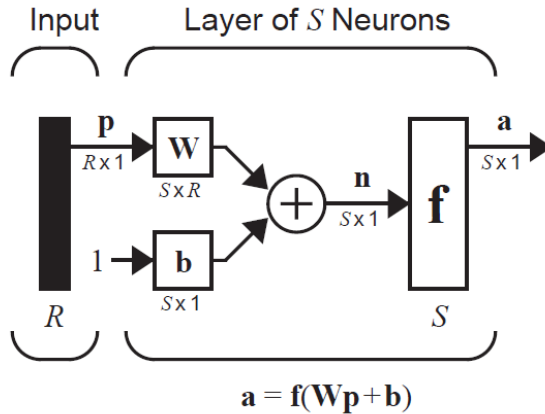


Figure VI-4 Neural Network Model [18, 20]

The transfer function (f) may be a linear or a nonlinear function. A particular transfer function is chosen based on the specific requirement of the application. A log-sigmoid function, also known as a logistic function is used as the transfer function in the network designed. The relationship is as shown in Equation VI-1 below.

$$\sigma(t) = \frac{1}{1 + e^{-\beta t}} \quad \text{Equation VI-1}$$

Where β is a slope parameter. This is called the log-sigmoid transfer function depicted in Figure VI-6. The log-sigmoid transfer function is commonly used in multilayer neural networks that are trained using the back propagation algorithm, since the function is differentiable. The log sigmoid has the similar property to that of the step function, with the addition of a region of uncertainty. A log-sigmoid function in this

respect is very similar to the input-output relationships of biological neurons, although not exactly the same. Below is the graph of a log-sigmoid function.

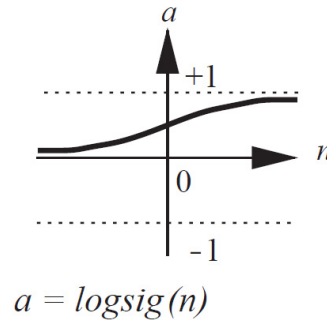


Figure VI-5 Log-sigmoid transfer function [22]

Sigmoid functions are also preferred because it is easy to calculate the derivatives, which is helpful for calculating the weight updates in certain training algorithms.

6.5.1 Training and Testing

We have used four different sets of data for training and testing of the co-simulation neural network designed. The memory module of the design and the testing of the network is being shifted on to the hardware level to speed up the performance. Since the neural network contains 16 input neurons, each database image is processed and stored as a feature vector of 16 values. The image is being compressed from 4096 pixel values into 16 feature vector values. The feature extraction algorithm is shown in the flowchart Figure VI-7. Once the input pre-processed image is obtained, the edge image is being calculated. The edge image is split into four quadrants and the maximum location

of each quadrant is being calculated. The distance values to be stored as the feature value is the distance between center of the image and each quadrant maximum images.

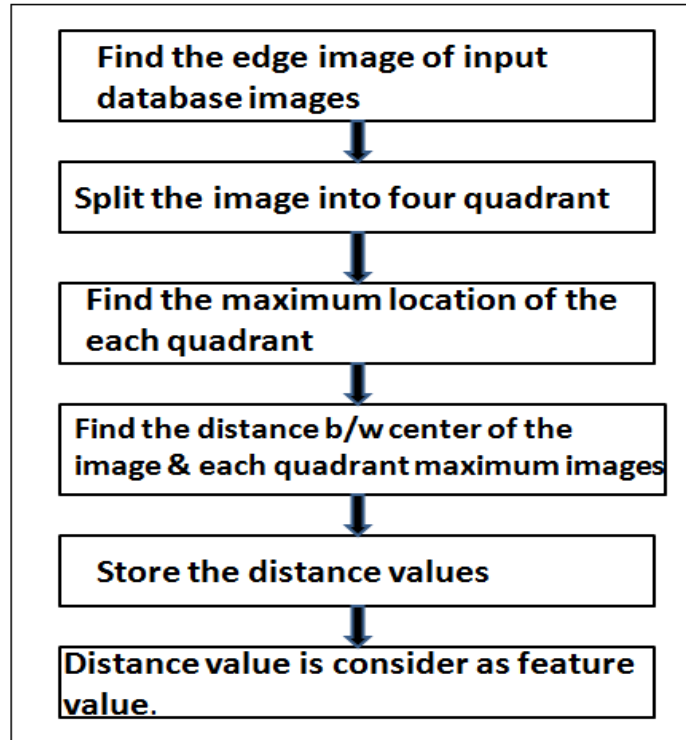


Figure VI-6 Feature Extraction Algorithm for distance values.

The compression ratio is 256 times that of the input values. Hence only 0.39% of the image is being used as the feature vector to train and test the neural network designed. As the database set gestures involved in the application may vary very rapidly, it is highly essential to keep the feature vector as low as possible with no tradeoff with respect to accuracy and performance. In this particular application the feature vector is maintained as 16 bit vector.

6.5.2 Gesture Recognition

Gesture Recognition is widely used in applications like sign language, human-computer interfaces, gaming and animation technology etc. Once a test image is selected, the neural network weights matrix is used to display the recognized gesture on a 7x5 display grid.

6.6 Device Utilization Factor

To understand the resource constraints, Xilinx ISE simulations are performed. The synthesis report provides the information about the device utilization in detail. Figure VI-8 and Figure VI-9 show the report with the representation as defined below [45].

Device Utilization - Indicates the FPGA elements, such as flip-flops, LUTs, block RAM, and DSP48s.

Estimated - Indicates the number of FPGA elements the algorithm might use based on the current directive configurations.

Total - Indicates the total number of FPGA elements available on the FPGA target.

Percent - Indicates the percentage of the FPGA elements that the algorithm might use.

From the summary reports developed, it is observed that only 33% of the IOBs are being used on the hardware platform. The report is being developed on a Xilinx FPGA SPARTAN 3E with target device XC3S250e-5tq144.

cosim Project Status (07/16/2012 - 11:53:06)			
Project File:	cosim.isc	Current State:	Programming File Generated
Module Name:	design_process	Errors:	No Errors
Target Device:	xc3s250e-stq144	Warnings:	No Warnings
Product Version:	ISE 10.1 - Foundation Simulator	Routing Results:	All Signals Completely Routed
Design Goal:	Balanced	Timing Constraints:	All Constraints Met
Design Strategy:	Xilinx Default (unlocked)	Final Timing Score:	0 (Timing Report)

cosim Partition Summary	
No partition information was found.	

Device Utilization Summary				
Logic Utilization	Used	Available	Utilization	Note(s)
Number of 4 input LUTs	10	4,896	1%	
Logic Distribution				
Number of occupied Slices	8	2,448	1%	
Number of Slices containing only related logic	8	8	100%	
Number of Slices containing unrelated logic	0	8	0%	
Total Number of 4 input LUTs	12	4,896	1%	
Number used as logic	10			
Number used as a route-thru	2			
Number of bonded IOBs	36	108	33%	
IOB Flip Flops	17			

Figure VI-7 Device Utilization Summary Report for memory storage.

cosim Project Status (07/16/2012 - 11:07:38)			
Project File:	cosim.isc	Current State:	Synthesized
Module Name:	design_process	Errors:	No Errors
Target Device:	xc3s250e-stq144	Warnings:	No Warnings
Product Version:	ISE 10.1 - Foundation Simulator	Routing Results:	
Design Goal:	Balanced	Timing Constraints:	
Design Strategy:	Xilinx Default (unlocked)	Final Timing Score:	

cosim Partition Summary	
No partition information was found.	

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	15	2448	0%
Number of 4 input LUTs	12	4896	0%
Number of bonded IOBs	36	108	33%
Number of GCLKs	1	24	4%

Detailed Reports					
Report Name	Status	Generated	Errors	Warnings	Infos
Synthesis Report	Current	Mon Jul 16 11:07:20 2012	0	0	0
Translation Report					
Map Report					

Figure VI-8 Device Utilization Summary Report for gesture recognition system test design module.

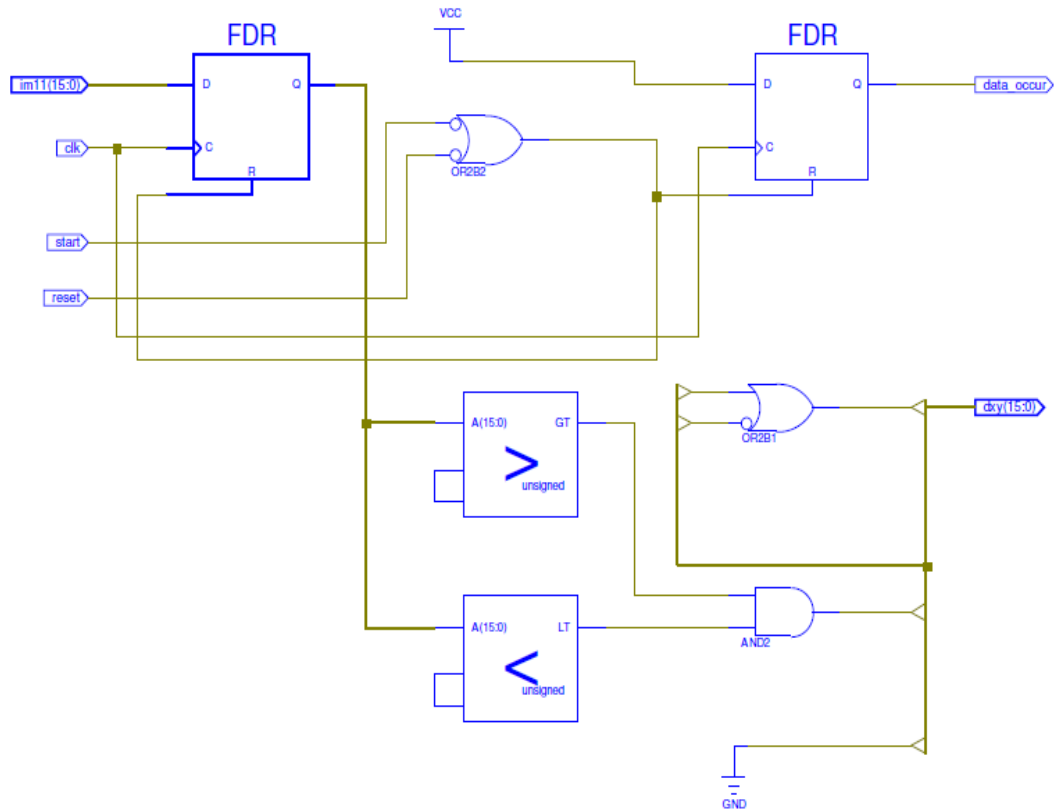


Figure VI-9 RTL schematic of Design process flow.

Each image is being read and stored on to the memory of the Altera MODELSIM. The RTL schematic design is being shown in Figure VI-10 above.

6.7 Simulation Output

The simulation output of the test design is being shown in Figure VI-11 below run on the Xilinx ISE. The memory *mem[0:4095]* loads 4096 values of each image and then processes it to extract features of set of 16 values and stores in *dxy[15:0]* shown. Figure VI-12, Figure VI-13, Figure VI-14 and Figure VI-15 show the simulation output for different database set used for the gesture recognition application. The simulation time

took was 164.04ps to load the memory into the Modelsim and test the system design. The clock is represented by *clk*, memory represented by *mem*, feature vector represented by *dxy* and the database image file is represented by *im11*. The test_design/k stores 106496 (= 4096 x 26) values which shows 4096 values of each of the 26 alphabet (A to Z) images considered.

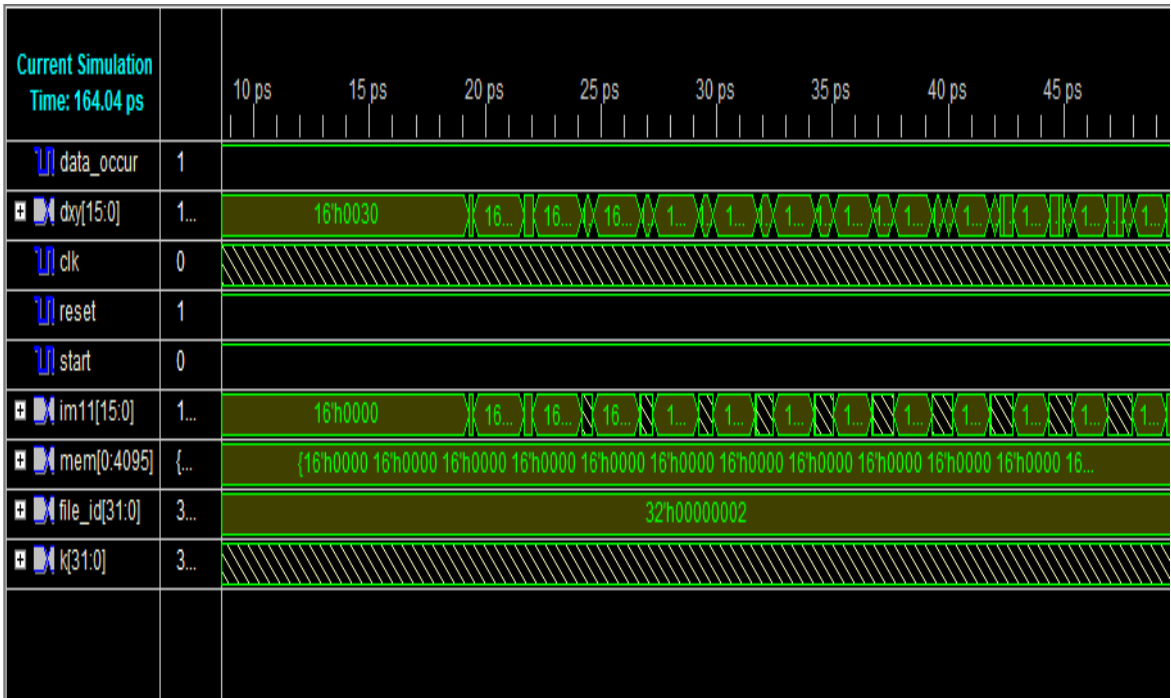


Figure VI-10 Simulation result in Xilinx ISE indicating the memory stored.

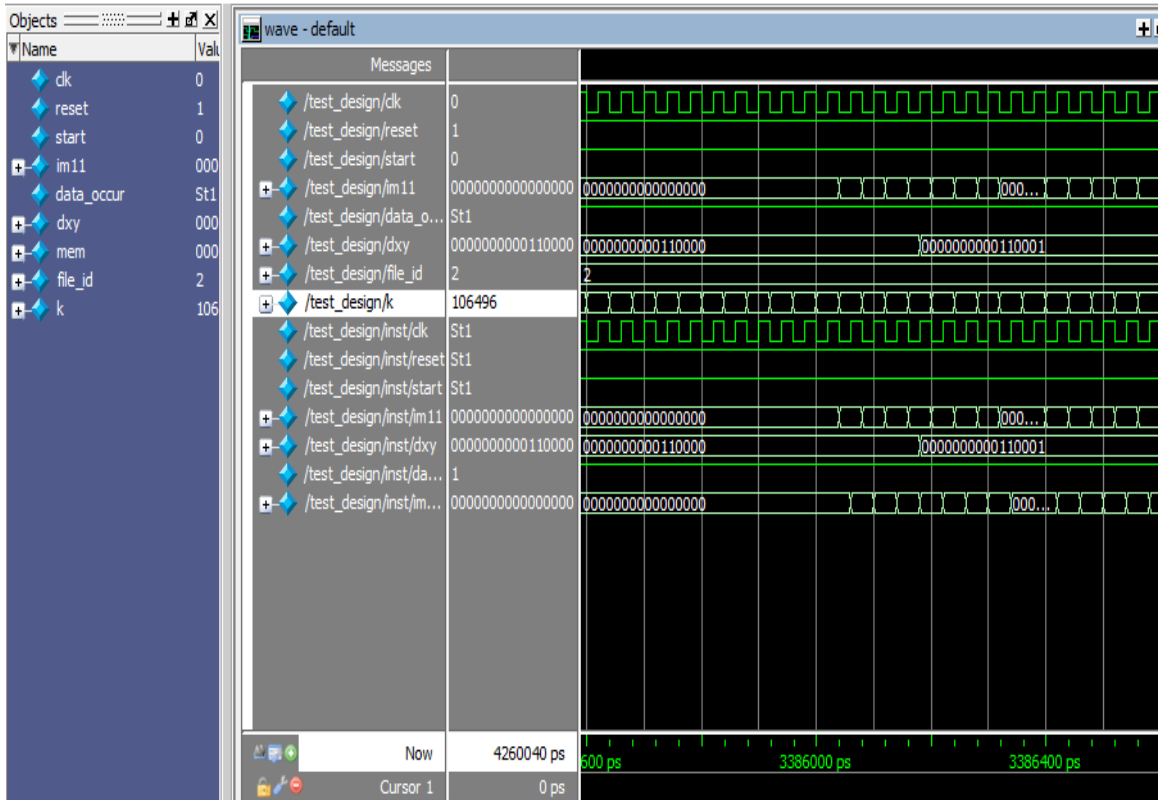


Figure VI-11 Simulation output for Database set-1

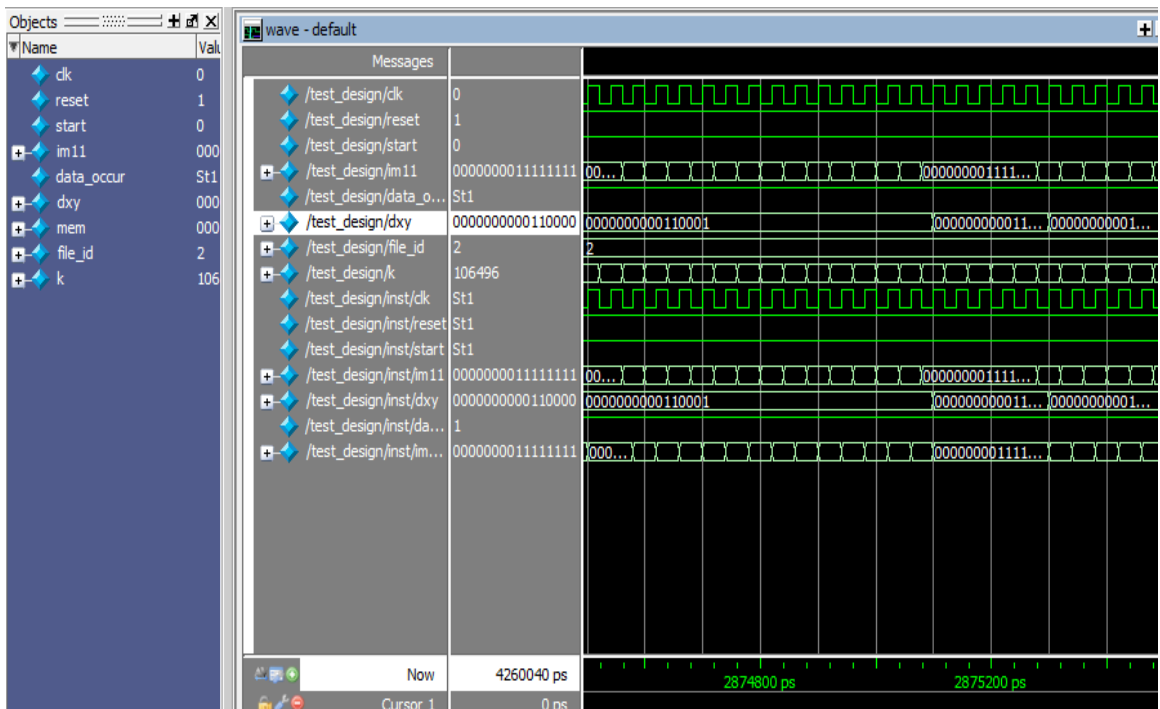


Figure VI-12 Simulation output for Database set-2

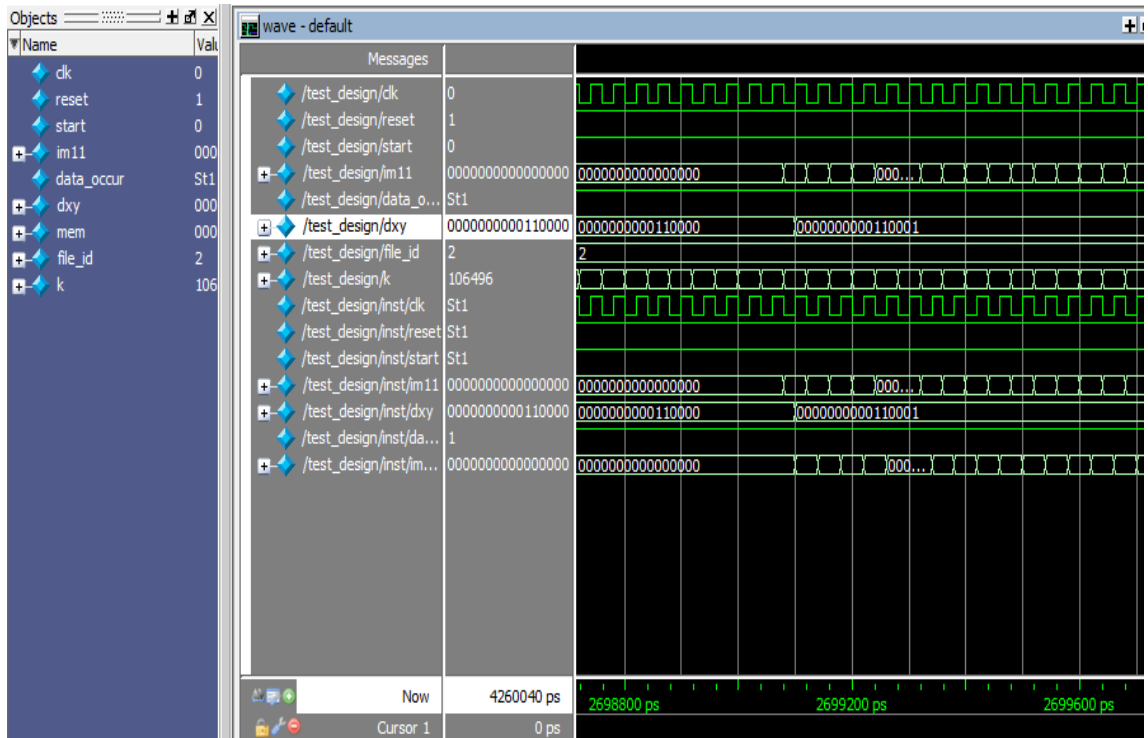


Figure VI-13 Simulation output for Database set-3

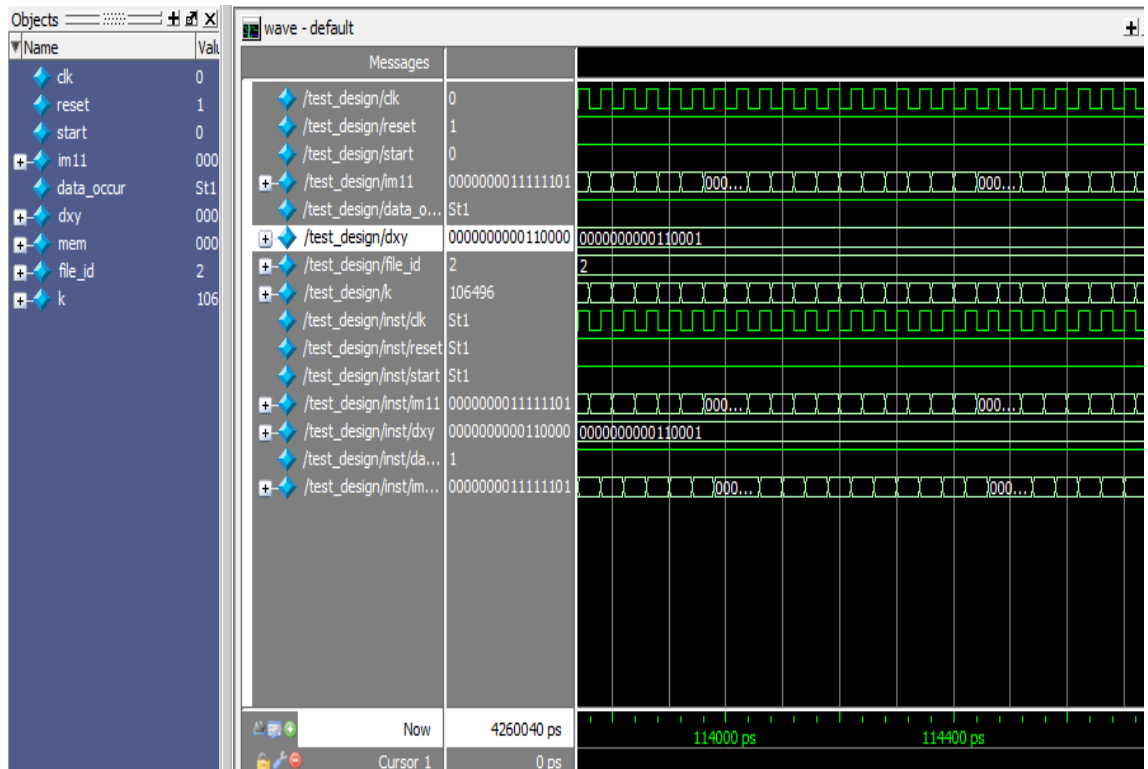


Figure VI-14 Simulation output for Database set-4

6.8 Experimental Results

The co-simulation platform is designed to improve the speed of the application. Current academic and industrial researchers have recently been focusing on analyzing images of people. While researchers are making progress, the problem is hard and many present day algorithms are complex, slow or unreliable. The algorithms that run near real-time require computers that are very expensive relative to the existing hand-held interface devices. The developed method runs quickly, accurately and fits for the low power dissipation application.

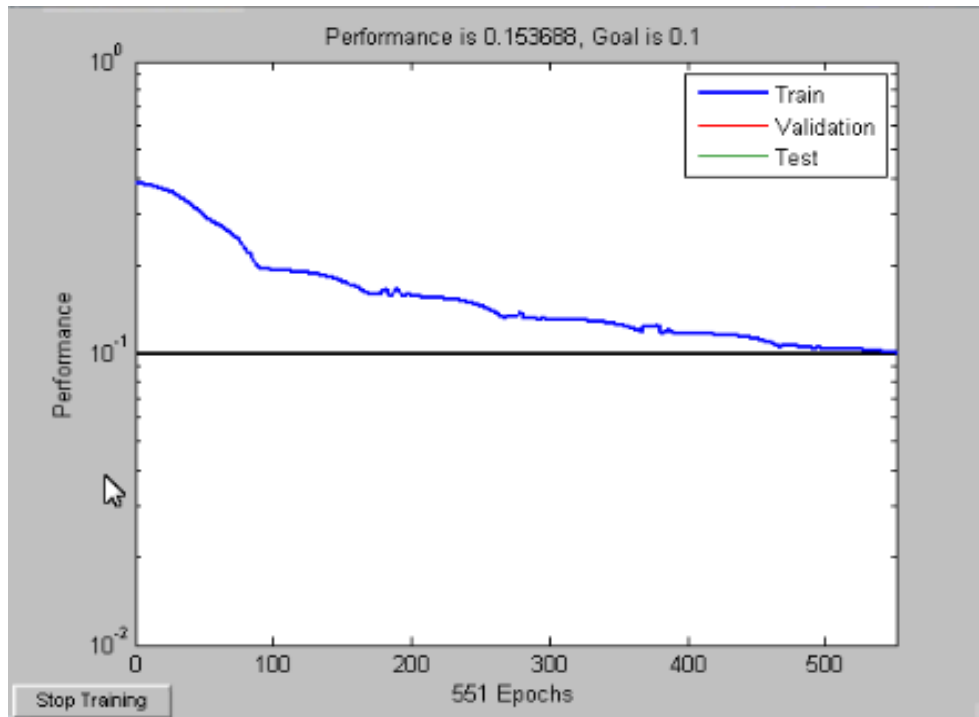


Figure VI-15 Performance curve for Gesture recognition using SW (MATLAB software) analysis with reduced database sets for training.

Figure VI-16 shows the performance curve for Gesture recognition using MATLAB software analysis with reduced database sets for training. Figure VI-17 shows

the performance curve for Gesture recognition using HW/SW co-simulation analysis (our approach). Figure VI-18 shows the performance curve for Gesture recognition using MATLAB software simulation analysis for complete database sets.

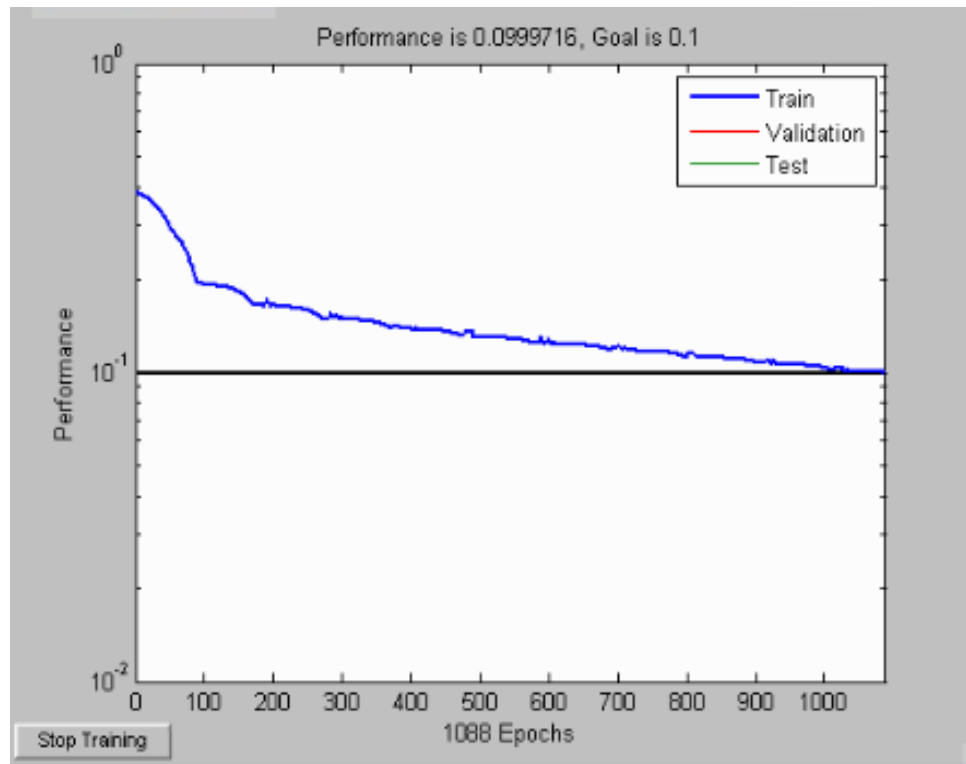


Figure VI-16 Performance curve for Gesture recognition using HW/SW co-simulation analysis.

The algorithm is able to detect the gesture input test image with 100% success rate for all the sign language alphabets (A to Z). Figure VI-19 and Figure VI-20 depict the output result for the test letter “P”. The influence on major factors like performance, epochs, MSE (mean square error), gradient and time are being compared to both approaches with various database memory and are tabulated in the Table VI-1 shown below.

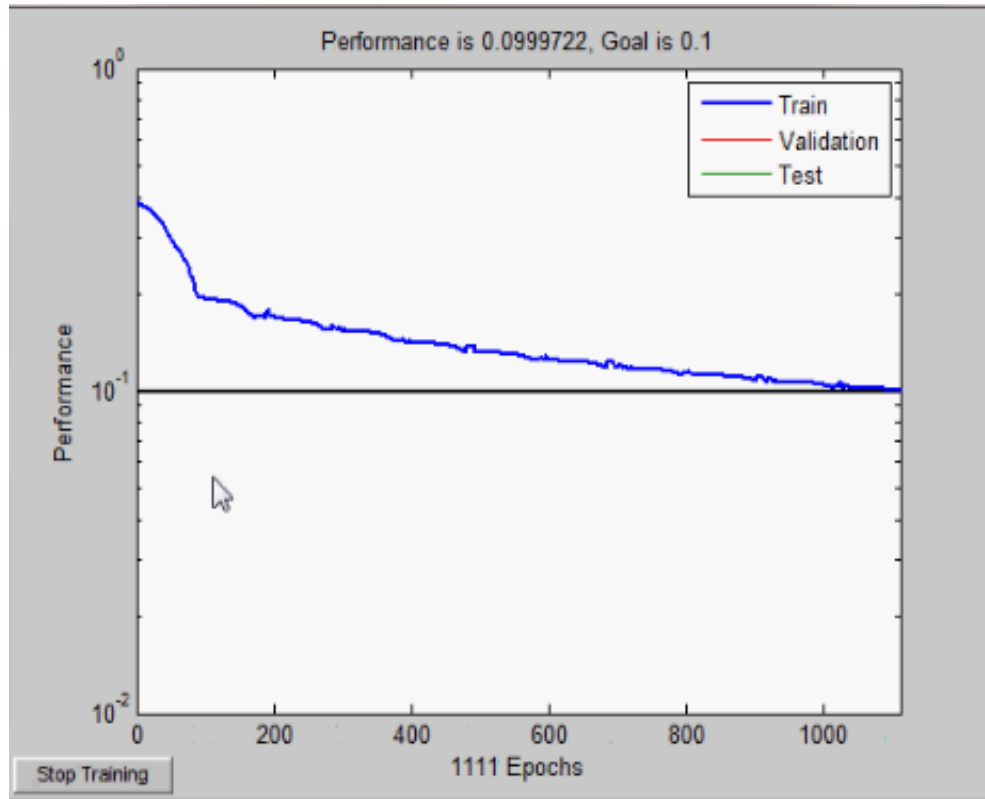


Figure VI-17 Performance curve for Gesture recognition using SW (MATLAB) simulation analysis.

It is observed from the results tabulated that there is a 10 times decrease in the MSE of the HW/SW co-simulation platform compared to completely SW based platform. The decrease in MSE is directly related to increase in accuracy of the approach adopted. The time indicated in the table represents the time taken to recognize the test image (time involved in providing the output gesture identified). From the table clearly the recognition time is reduced by 10.059 times by adopting the HW/SW co-simulation platform instead of completely SW based approach.

Table VI-1 Comparison of SW vs. HW/SW simulation platforms

	Software simulation (MATLAB)	HW/SW co-simulation (MATLAB and Altera MODELSIM Quartus II)
Performance	0.999722	0.999716
Epochs	1111/2000	1088/2000
MSE	0.998182/0.1	0.0999716/0.1
Gradient	0.00269182/1e-006	0.00204379/1e-006
Time	0.0058secs	5.7656e-004secs

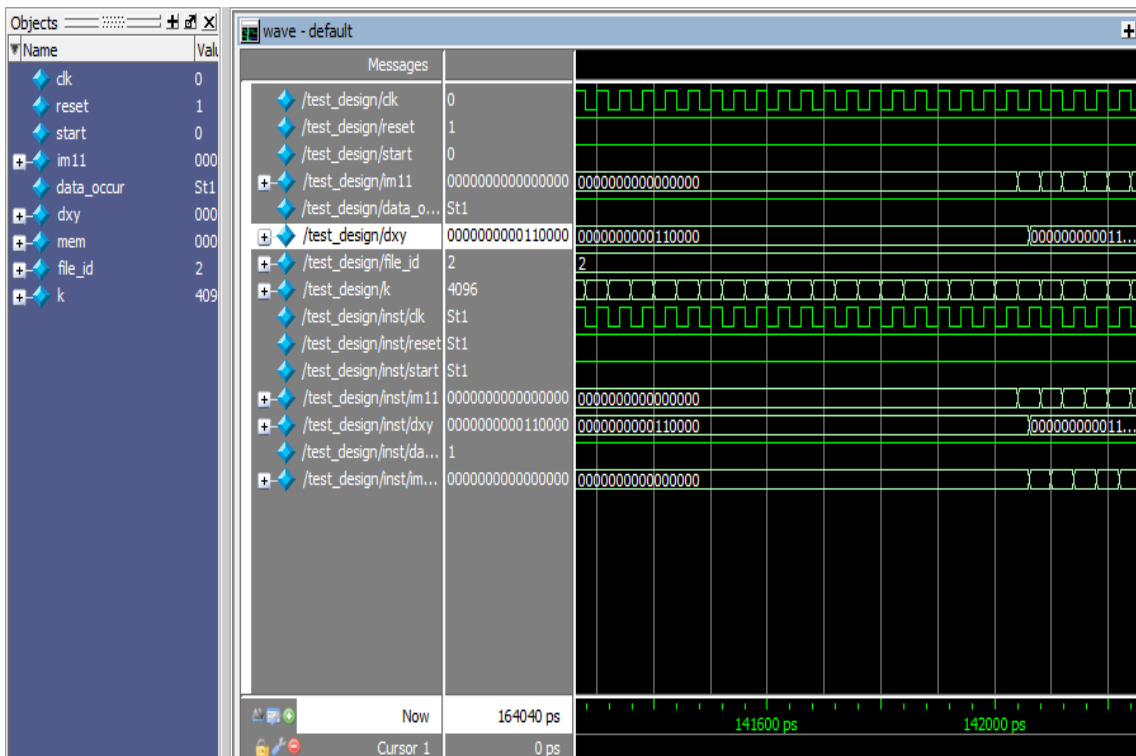


Figure VI-18 System design test output simulation for image gesture alphabet “P”.

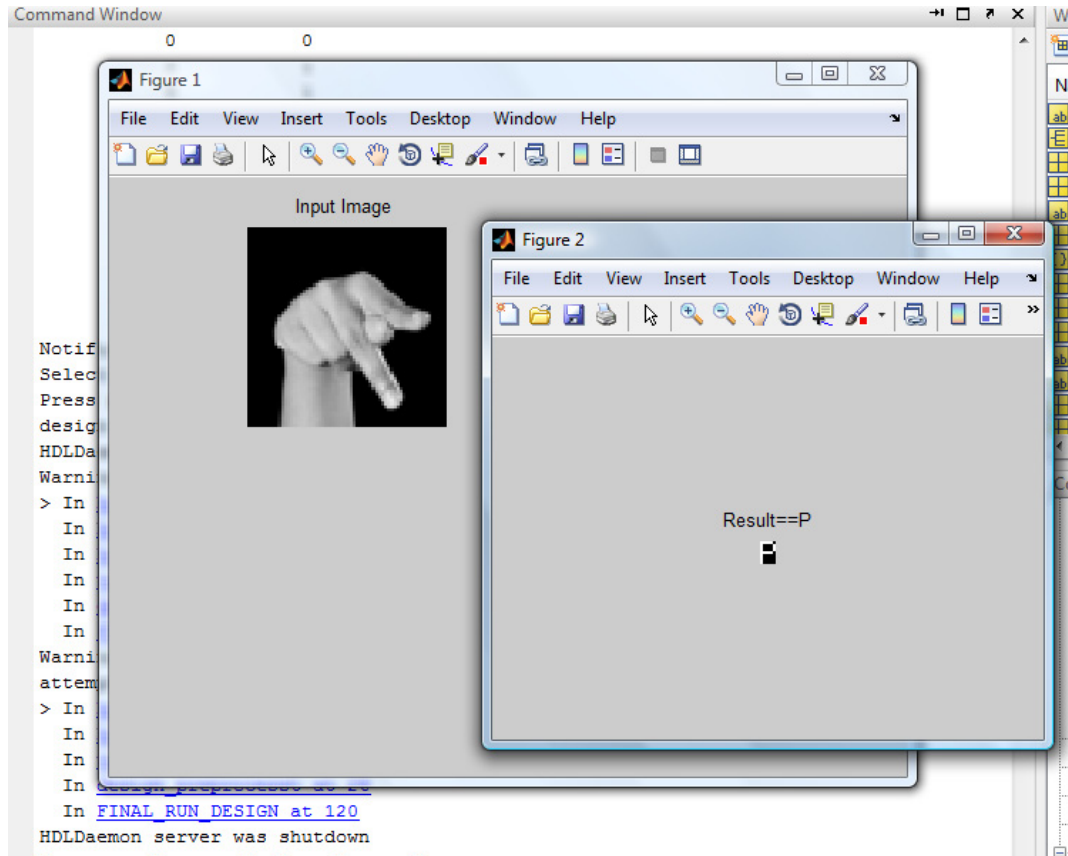


Figure VI-19 Output sign recognition displayed as a grid matrix.

6.9 Chapter Summary

Chapter 6 focuses on taking advantage from the current FPGA technology. A novel hardware/software co-simulation methodology using HDL simulations on FPGA is developed as an effort to accelerate the simulation time and performance. We attempt to leverage the merits of the software simulation and hardware emulation to retain both the flexibility and performance by adopting a hardware/software based platform approach. A novel maximum quadrant distance method is used to derive the feature vectors required to test and train the co-simulation neural network designed. The compression ratio is 256 times that of the input values. Hence only 0.39% of the image is being used as the feature

vector to train and test the neural network designed. Hence the method runs quickly, accurately and fits for the low power dissipation application. Clearly, the recognition time is reduced by 10.059 times by adopting the HW/SW co-simulation platform instead of completely SW based approach.

VII CONCLUSION AND FUTURE WORK

7.1 Concluding Remarks

A study for the importance of video and imaging techniques involved, shown in Figure I-1, is described in detail which concentrates on the key topics: (1) target detection and (2) gesture recognition. The work accomplished in this dissertation is meant to advance upon surveillance monitoring in space for defense and military purpose to improve the decision support in the case of occlusion. A generic recognition system with improved performance and training is developed on the HW/SW co-simulation platform. Specific example, focused upon, is sign language for pattern recognition.

7.1.1 Motion and Occlusion Detection in Surveillance

A novel TSS system is developed with motion estimation and occlusion occurrence detector units. This topic deals with an innovative technique for occlusion detection to avoid problems in tracking. Currently this methodology uses motion analysis techniques to predict the occurrence of occlusion that might occur in a certain time frame in future. To save memory, bandwidth and power involved for an extra surveillance camera, focus has been to trigger the camera only for certain time interval well predicted before. This algorithm works for both linear and non-linear motion of the targets assuming spatial layout is maintained for a certain time period.

7.1.2 A Global approach toward Target Detection

The design is simple and has the flexibility to choose the segmentation threshold value. This system is applicable to both indoor and outdoor environments. A new Running Average Mean Threshold (RAMT) algorithm is being derived to make the background subtraction (a major module of any image processing application) into a global approach rather than using a manually set threshold value. The design adapts the threshold automatically to different environments and different illumination levels to recognize the target in the scene. MicroBlaze processor is used to make it computationally efficient.

7.1.3 Pattern Recognition using Combinational Neural Networks

A new strategy for pattern recognition is developed using CNN. Analysis of the method is done on American Sign Language recognition. It is shown through simulations that the time for processing is reduced due to the tree structure adopted. Though results verify good performance, the time required for real-time processing was not acquired. Hence a HDL platform was used to study the implementation of the developed CNN.

7.1.4 HW/SW co-simulation of Gesture Recognition

A novel hardware/software co-simulation methodology using HDL simulations on FPGA is developed as an effort to accelerate the simulation time and performance. We attempt to leverage the merits of the software simulation and hardware emulation to retain both the flexibility and performance by adopting a hardware/software based

platform approach. A novel maximum quadrant distance method is used to derive the feature vectors required to test and train the co-simulation neural network designed. Thus the FPGAs constitute a very powerful option for implementing a part of the neural network since we can really exploit their parallel processing capabilities to improve the performance. The memory controller is designed in the VERILOG HDL. The testing part of the neural network algorithm is being hardwired to improve the speed and performance.

7.2 Future Work

In occlusion detection algorithm developed, safe distance factor was assumed to be known and hence more analysis can be made in future to find the statistics involved for determining its value. The TSS architecture can be developed with future research on integration of the different modules studied on FPGA. In the ASL recognition on HDL platform only the static images were processed. An inclusion of additional SRAM vector back to store the motion vectors of adjacent frames could be considered for future research to eliminate the limitations of dynamic images. Creating an ASL translator is not the desired result at this point. This would combine advanced grammar and syntax structure understanding of the system, which is outside the scope of this project. To advance ASL recognition further a real-time dictionary could be created with the use of video/ images.

7.3 Chapter Summary

Chapter 7 focuses on conclusions and future work for different areas of video and imaging techniques, which includes: (1) motion and occlusion detection in surveillance; (2) a global approach towards target detection; (3) pattern recognition using CNN; and (4) HW/SW co-simulation of gesture recognition. A specific example presented herein, are few outdoor and indoor video surveillance for target detection and ASL for pattern recognition.

REFERENCES

- [1] ALTERA corporation, "Video and Image Processing Design using FPGAs," March 2007.
- [2] E. Vorobiev and V. Belyaev, "Technology trends regarding video signal processing," Proceedings- SPIE the International society for optical engineering, vol. 6637, pp. 66370T, 2007; doi:10.1117/12.742964.
- [3] A. Amer, E. Dubois, and A. Mitiche, "Real-time system for high-level video representation: application to video surveillance," Proceedings SPIE Int. Symposium on Electronic Imaging, Conf. on Visual Communication and Image Processing (VCIP), Santa Clara, USA, vol. 5022, pp. 530-541, January 2003.
- [4] I. Haritaoglu, D. Harwood, and L. S. Davis, "Real-time surveillance of people and their activities," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22 , Issue 8 , pp. 809-830, August 2000.
- [5] Collins et al., A System for Video Surveillance and Monitoring," VSAM Final Report, Technical report CMU-RI-TR-00-12, Carnegie Mellon University, May, 2000.
- [6] F. Lv, J. Kang, R. Nevatia, I. Cohen, and G. Medioni, "Automatic Tracking and Labeling of Human Activities in a Video Sequence," Proceedings of the 6th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS04), Prague, Czech Republic, May 2004.
- [7] Dockstader et al, "Multiple camera tracking of interacting and occluded human motion," Proceedings of the IEEE , vol. 89 , Issue 10 , pp.1441-1455, October 2001.
- [8] T. H. Chang, S. Gong, "Tracking multiple people with a multi-camera system," Multi-Object Tracking, Proceedings. IEEE Workshop, pp.19-26, 2001; doi: 10.1109/MOT.2001.937977
- [9] S.L. Dockstader and A.M. Tekalp, "Multiple camera fusion for multi-object tracking," Proceedings. IEEE Workshop on Multi-Object Tracking, pp. 95-102, 2001.
- [10] P. Mekala, R. Salmeron, J. Fan, and A. Davari, J. Tan, "Occlusion Detection Using Motion-Position Analysis," IEEE 42nd Southeastern Symposium on System Theory (SSST'10), Tyler, TX, pp. 197-201, March 2010.

- [11] P. Mekala, S. Erdogan, and J. Fan, "Automatic object recognition using combinational neural networks in surveillance networks," IEEE 3rd International Conference on Computer and Electrical Engineering (ICCEE'10), Chengdu, China, vol. 8, pp. 387-391, November 2010.
- [12] P. Mekala, Y. Gao, Jeffrey Fan, and A. Davari, "Real-time sign language recognition based on neural network architecture," Joint IEEE International Conference on Industrial Technology & 43rd Southeastern Symposium on System Theory (SSST'11), Auburn, AL, pp. 197-201, March 2011.
- [13] P. Mekala, and J. Fan, "Design Enhancement of Combinational Neural Networks using HDL based FPGA Framework for Pattern Recognition," i-manager's Journal on Electronics Engineering (JELE), vol. 2, Issue 1, pp. 6-16, September - November, 2011.
- [14] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," ACM Computing Survey 38(4), Article 13, 45 pages. December 2006.
- [15] R. Collins, A. Lipton, H. Fujiyoshi, and T. Kanade, "Algorithms for Cooperative Multisensor Surveillance," Proceedings of IEEE, vol. 89, Issue 10, pp. 1456-1477, October 2001.
- [16] V. Salary and I. K. Sethi, "Feature point correspondence in the presence of occlusion," IEEE Transactions Pattern Analysis Machine Intelligence, vol. 12, Issue 1, pp. 87-91, January 1990.
- [17] A. Yilmaz, X. Li, and M. Shah, "Contour based object tracking with occlusion handling in video acquired using mobile cameras," IEEE Trans. Pattern Analysis Machine Intelligence, vol. 26, Issue 11, pp.1531–1536, 2004.
- [18] O. Lanz and S. Messelodi, "A sampling algorithm for occlusion robust multi target detection," Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), pp. 346-351, 2009.
- [19] Q. Cai and J. K. Aggarwal, "Tracking human motion using multiple cameras," Proceedings of the 13th International Conference on Pattern Recognition, vol. 3, pp. 68–72, August 1996.
- [20] G.A Carpenter and S. Grossberg, "The ART of Adaptive pattern recognition by a self-organizing Neural Network," COMPUTER, vol. 21, Issue 3, pp. 77-88, March 1988.

- [21] W. T. Freeman, M. Roth, "Orientation Histograms for Hand Gesture Recognition," Mitsubishi Electric Research Laboratory Technical report TR-94-03a, December 1994.
- [22] Laurene Fausett, 1994: Fundamentals of Neural Networks – architecture, algorithms and applications, Prentice Hall publishers.
- [23] Paulraj M P, S. Yaacob, M. S. B. Z. Azalan, and R. Palaniappan, "A Phoneme based sign language recognition system using skin color segmentation," International Colloquium on Signal Processing and Its Applications (CSPA), pp: 1 – 5, May 2010.
- [24] L. Rabiner and B. Juang, "An Introduction to Hidden Markov Models," IEEE ASSP Magazine, vol. 3, Issue 1, pp. 4–16, January 1986.
- [25] H. Lee, "Hand Gesture Recognition using Orientation histogram," Proceedings of the IEEE Region 10 Conference on TENCON, vol. 2, pp: 1355-1358, December 1999.
- [26] K. K. Byong and H. S. Yang, "Finger Mouse and Gesture Recognition System as a New Human Computer Interface," Computers & Graphics, vol. 21, Issue 5, pp. 555 - 561, 1997.
- [27] M. Maraqa and R. Abu-Zaiter, "Recognition of Arabic sign language (ArSL) using recurrent neural networks," First International Conference on Applications of Digital Information and Web Technologies (ICADIWT), pp: 478 – 481, August 2008.
- [28] Y. Quan, "Chinese Sign Language Recognition Based On Video Sequence Appearance Modeling," IEEE Conference on Industrial Electronics and Applications (ICIEA), pp: 1537 – 1542, June 2010.
- [29] K. Kawahigasi, Y. Shirai, J. Miura, and N. Shimada "Automatic Synthesis of training Data for Sign Language Recognition Using HMM," Proceedings of the 10th international conference on Computers Helping People with Special Needs (ICCHP'06), pp.623-626, 2006.
- [30] B. Bauer and H. Hienz, "Relevant features for video-based continuous sign language recognition," Fourth IEEE International Conference on Automatic Face and Gesture Recognition Proceedings, pp: 440 – 445, 2000.
- [31] L. T. Starner, J. Weaver, and A. Pentland, "Real-time American sign language recognition using desk and wearable computer based video," IEEE Transactions

- on Pattern Analysis and Machine Intelligence, vol. 20, Issue 12, pp. 1371–1375, December 1998.
- [32] K. Grobel, MIT Hidden–Markov–Modellen. Ph. D. Thesis, Aachen University of Technology, VDI-Verlag, D'usseldorf, 1999.
- [33] M. Maebatake, I. Suzuki, M. Nishida, Y. Horiuchi, and S. Kuroiwa, "Sign Language Recognition Based on Position and Movement Using Multi-Stream HMM," Second International Symposium Universal Communication (ISUC '08), pp: 478 – 481, 2008.
- [34] E. J. Holden, G. Lee, and R. Owens, "Australian Sign Language Recognition," Machine Vision and Applications, vol.16, pp. 312-320, 2005.
- [35] W. T. Freeman, D. B. Anderson, P. A. Beardsley, C. N. Dodge, M. Roth, C. D. Weissman, W. S. Yerazunis, H. Kage, K. Kyuma, Y. Miyake, and K. Tanaka, "Computer Vision for Interactive Computer Graphics," IEEE Computer Graphics and Applications, vol. 18, Issue 3, pp. 42-53, May-June 1998.
- [36] R. Liang and M. Ouhyoung, "Real-time continuous gesture recognition system for sign language", Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition, pp. 558-567, April 1998.
- [37] H. Brashear, K. H. Park, S. Lee, V. Henderson, H. Hamilton, T. Starner, "American Sign Language Recognition in Game Development for Deaf Children" Assets '06 Proceedings of the 8th international ACM SIGACCESS conference on Computers and accessibility, pp.79-86, 2006.
- [38] A. R. Omondi, J. C. Rajapakse, "FPGA Implementations of Neural Networks," 2006 Springer.
- [39] D. L. Berry, "VHDL programming by examples", McGraw-Hill, fourth edition, 2002.
- [40] J. Schemmel, K. Meier, and F. Schurmann, "A VLSI Implementation of an Analog Neural Network suitable for Genetic Algorithms," ICES '01 Proceedings of the 4th International Conference on Evolvable Systems: From Biology to Hardware, pp. 50-61, October 2001.
- [41] K. L. Short, 2009: VHDL for Engineer, NJ, Pearson Prentice Hall publishers.
- [42] P. J. Ashenden, 1995: The designer's guide to VHDL, San Francisco, Morgan Kaufmann publishers.

- [43] P. Coussy, D. Gajski, M. Meredith and A. Takach, "An Introduction to high -level synthesis," IEEE Design and Test of Computers, vol. 26, Issue 6, pp. 8-11, July-August 2009.
- [44] F. Choong, M. B. I. Reaz and F. Mohd-Yasin, "Power quality disturbance detection using artificial intelligence: A hardware approach," Proceedings of 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05), pp. 146a, April 2005.
- [45] Xilinx, XST User Guide, Xilinx Inc. 2009.
- [46] B. L. B. L. Pulito, T. R. Damarla, S. Nariani, "A Two-Dimensional Shift Invariant Image Classification Neural Network which overcomes the Stability / Plasticity Dilemma," Neural Networks, IJCNN International Joint Conference, pp. 825 – 833, vol.2, June 1990.
- [47] C. Stauffer and W. E. L. Grimson , "Adaptive background mixture models for real-time tracking," IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, 1999.
- [48] J. B. Kim, H. S. Park, M. H. Park, M. Piccardi, "Background subtraction techniques: a review," International Conference on Systems, Man and Cybernetics, vol.4, pp. 3099-3104, October 2004.
- [49] G. Welch and G. Bishop, "An introduction to the Kalman filter," Tech. Rep. TR 95-041, University of North Carolina at Chapel Hill, Chapel Hill, NC, USA, 2004.
- [50] M. Kohler, "Using the Kalman Filter to track Human Interactive Motion-Modelling and Initialization of the Kalman Filter for Translational Motion," Tech. Rep. #629, Informatik VII, University of Dortmund, 1997.
- [51] M. Grewal, and A. Andrews. Kalman Filtering Theory and Practice. Englewood Cliffs, NJ: Prentice-Hall, 1993.
- [52] H. Sorenson, Kalman Filtering: Theory and Application. Los Alamitos, CA: IEEE Press, 1985.
- [53] Z. Khan, T. Balch, and F. Dellaert, "An MCMC-based particle filter for tracking multiple interacting targets," Proceedings of 8th European Conference on Computer Vision (ECCV '04), vol. 4, pp. 279-290, Prague, Czech Republic, May 2004.

- [54] W. C. Lin and Y. Liu. "Tracking dynamic near-regular textures under occlusions and rapid movements," ECCV, 2006.
- [55] J. Sullivan and S. Carlsson, "Tracking and labeling of interacting multiple targets," ECCV, pp 619–632, 2006.
- [56] T. H. Chang, S. Gong, and E. J. Ong, "Tracking multiple people under occlusion using multiple cameras," Proceedings in British Machine Vision Conference, 2000, pp. 566–576.
- [57] T. A. Ademoye, A. Davari, C. Castello, S. Fan, and J. Fan, "Path planning via CPLEX optimization," 40th Southeastern Symposium on Southeastern Symposium on System Theory, pp. 92-96, 2008.
- [58] D. Singh and B. S. Khehra, "Digit Recognition System Using Back Propagation Neural Network," International Journal of Computer Science and Communication, vol. 2, No. 1, pp. 197-205, January-June 2011.
- [59] Donald G. Bailey, "Design for Embedded Image processing on FPGAs," Wiley-IEEE press, August 2011.
- [60] Chi-Jeng Chang, Pei-Yung Hsiao, Zen-Yi Huang (2006). 'Integrated Operation of Image Capturing and Processing in FPGA', IJCSNS International Journal of Computer Science and Network Security, vol.6 Issue.1A, pp. 173-179.
- [61] S. A. Fahmy, "Novel FPGA-based implementation of median and weighted median filters for image processing," International conference on Field Programmable Logic and Applications, pp. 142-147, August 2005.
- [62] Y. Benezeth, P. M. Jodoin, B. Emile, H. Laurent, and C. Rosenberger, "Review and evaluation of commonly-implemented background subtraction algorithms," 19th International Conference on Pattern Recognition (ICPR), pp. 1-4, December 2008.
- [63] Jay Gould, "High Performance Embedded systems," Xilinx Inc.
- [64] K. Natarajan, M. John and J. Selvaraj. Silicon Technologies for Speaker Independent Speech Processing and Recognition Systems in Noisy Environments. In: France Miheli (ed.) Speech Recognition: Technologies and Applications I-Tech Education and Publishing. pp. 495-526, 2008.

- [65] Xilinx user guide
http://www.xilinx.com/support/documentation/sw_manuals/xilinx12_4/edk_ctt.pdf.
- [66] R. S. Soni, and D. Asati, "Development of FPGA Based Embedded Web Server Using a Soft-core Processor," *International Journal of Advanced Research in Computer Engineering & Technology*, vol. 1, Issue 4, pp. 509 - 512, June 2012.
- [67] R. Jesman, F. M. Vallina, and J. Saine, R. Jesman, F. M. Vallina, and J. Saine, Embedded Computing and signal processing Laboratory (ECASP), Illinois Institute of Technology, <http://ecasp.ece.iit.edu>.
- [68] D. Crookes , K. Benkrid , A. Bouridane , K. Alotaibi , and A. Benkrid, 'Design and implementation of a high level programming environment for FPGA-based image processing', *Vision, Image and Signal Processing, IEE Proceedings*, vol. 147, Issue: 4 , pp. 377 -384, August 2000.
- [69] Kilmartin, M. O. Conghaile , "Real Time Image Processing Object Detection and Tracking Algorithms," *Proceedings of the Irish Signals and Systems Conference, NUI, Galway*, pp. 207-214, June 1999.
- [70] L. Tan, *Digital Signal Processing: Fundamentals and Applications*, Elsevier academic press, 2008.
- [71] H. J. Kim, "Unsupervised Moving Object Segmentation and Recognition Using Clustering and a Neural Network," *Proceedings of the International Joint Conference on Neural Networks (IJCNN '02)*, pp. 1240-1245, 2002.
- [72] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison - Wesley, Massachusetts, USA, 1992.
- [73] A. Wahi and E. Thirumurugan, "Recognition of Objects by supervised neural network using wavelet features," *First International Conference on Emerging Trends in Engineering and Technology (ICETET)*, pp. 55-61, 2008.
- [74] H. Pan and L. Xia, "Efficient Object Recognition Using Boundary Representation and Wavelet Neural Network," *IEEE Transactions on Neural Networks*, vol. 19, Issue 12, pp. 2132 - 2149.
- [75] H. Roberto, D. Zhang, and G. Lu, "Review of shape representation and description techniques," *Pattern Recognition*, vol. 37, no. 1, pp. 1-19, 2004.

- [76] T. Y. Young and K. Fu, "Handbook of Pattern Recognition and Image Processing," Academic Press Inc, 1986.
- [77] C. M. Bishop. Neural Network for Pattern Recognition. Clarendon Press. Oxford 1998.
- [78] H. K. Kwan, "Simple sigmoid like activation function suitable for digital hardware implementation," Electronic Letters, vol. 28, pp. 1379-1380, July 1992.
- [79] P. Mekala, Jeffrey Fan, "A global approach for target detection using adaptive threshold on Microblaze processor," Journal of Engineering, Hindawi Publishing Corporation (in review).
- [80] P. Mekala, Jeffrey Fan, "Gesture recognition using neural networks based on HW/SW co-simulation platform," Journal of Advances in Software Engineering, Hindawi Publishing Corporation (in review).
- [81] Taeweon Suh, Hsien-Hsin S. Lee, Shih-Lien Lu, and John Shen, "Initial Observations of Hardware/Software Co-simulation using FPGA," Architecture Research Workshop on Architecture Research using FPGA Platforms in conjunction with International Symposium on High-Performance Computer Architecture, Austin, Texas, February 2006.
- [82] C. R. Clark, R. Nathuji, and H. H. S. Lee, "Using FPGA as a Prototyping Platform for Multi-core Processor Applications," 1st Workshop on Architecture Research using FPGA Platforms, February 2005.
- [83] R. Kjeldsen and J. Kender, "Finding skin in color images," IEEE Second International Conference on Automated Face and Gesture Recognition, Killington, VT, USA, pp. 312- 317, October 1996.
- [84] H. Hongo, M. Ohya, M. Yasumoto, and K. Yamamoto, "Face and hand gesture recognition for human-computer interaction," Fifteenth International Conference on Pattern Recognition, Barcelona, Spain, vol. 2, pp. 921-924, 2000.
- [85] C. H. Huang and W. Y. Huang, "Sign language recognition using a 3D Hopfield Neural Network," International conference on Image Processing, vol. 2, pp. 611-614, October 1995.
- [86] C. L. Huang, S. H. Jeng, "A model-based hand gesture recognition system," Journal on Machine Vision and Applications, vol. 12, Issue 5, pp. 243-258, November 2001.

- [87] J. Triesch and C. Von der Malsburg, "A system for person-independent hand posture recognition against complex backgrounds," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, Issue 12, pp. 1449–1453, 2001.
- [88] F. S. Chen, C. M. Fu, and C. L. Huang, "Hand gesture recognition using a real-time tracking method and hidden Markov models," *Image Vision Computer*, vol. 21, Issue 8, pp. 745–758, 2003.
- [89] J. P. Wachs, H. I. Stern, Y. Edan, M. I. Gillam, J. Handler, C. Feied, and M. Smith, "A Real-Time Hand Gesture System Based on Evolutionary Search Vision", vol. 22, Issue 3, pp. 1-11, 2006.
- [90] E. Yoruk, E. Konukoglu, B. Sankur, J. Darbon, "Shape-based hand recognition," *IEEE Transactions on Image Processing*, vol. 15, Issue 7, pp. 1803–1815, 2003.
- [91] National Instruments Inc., Introduction to FPGA Technology, Internet document, April 2012. <http://www.ni.com/white-paper/6984/en>.
- [92] <http://www.mathworks.com>
- [93] Xiang Ling, Zhongqi Li, Jianhao Hu and Shihong Wu, "HW/SW Co-Simulation Platforms for VLSI Design", *Circuits and Systems, APCCAS*, pp. 578 – 581, 2008.

VITA
PRIYANKA MEKALA

Education

Ph.D. Candidate in Electrical Engineering Florida International University Advisor: Jeffrey Fan	Aug. 2009 – Present Miami, FL
Master of Science in Electrical Engineering Arizona State University	Aug. 2007 – May. 2009 Tempe, AZ
Bachelor of Science in Electronics and Communications Osmania University	Aug. 2004 – Apr. 2007 Hyderabad, India

Honors and Awards

- Awarded IEEE-IES scholarship to present paper entitled "Real-time sign language recognition based on neural network architecture" at ICIT conference, Alabama, February 2011.
- Speaker at the FIU Annual scholarly forum held in 2011.
- Speaker at the Southeastern symposium on system theory (SSST) held in 2010 and 2011.
- Reviewer for SSST conference, 2012.
- Organizer and participant of the ISA cultural events at ASU, 2008.
- Award winner in extempore, assembliz at SAMAVARTHAN, India 2006.
- Merit certification for mathematical Olympiad organized by APAMT, 2001.
- Member of IEEE (Institute of Electrical and Electronics Engineers) and IETE (The Institution of Electronics and Telecommunication Engineers).

Publications

JOURNALS

J-3: **P. Mekala**, Jeffrey Fan, "A global approach for target detection using adaptive threshold on Microblaze processor," Journal of Engineering, Hindawi Publishing Corporation, 2012 (in review).

J-2: **P. Mekala**, Jeffrey Fan, "Gesture recognition using neural networks based on HW/SW co-simulation platform," Journal of Advances in Software Engineering, Hindawi Publishing Corporation, 2012 (in review).

J-1: **P. Mekala**, J. Fan, "Design Progression of Combinational Neural Networks using HDL based FPGA Framework for Pattern Recognition," i-manager's Journal on Electronics Engineering (JELE), September - November Issue, 2011.

PEER REVIEWED CONFERENCE PAPERS

C-3: **P. Mekala**, J. Fan, A. Davari, "Real-time sign language recognition based on neural network architecture," Joint IEEE International Conference on Industrial Technology & 43rd Southeastern Symposium on System Theory (SSST'11), Auburn, AL, March 14-17, 2011.

C-2: **P. Mekala**, S. Erdogan, J. Fan, "Automatic object recognition using combinational neural networks in surveillance networks," IEEE 3rd International Conference on Computer and Electrical Engineering (ICCEE'10), Chengdu, China, Vol. 8, pp. 387-391, November 16-18, 2010.

C-1: **P. Mekala**, R. Salmeron, J. Fan, A. Davari, J. Tan, "Occlusion Detection Using Motion-Position Analysis," IEEE 42nd Southeastern Symposium on System Theory (SSST'10), Tyler, TX, pp. 197-201, March 7-9, 2010

REFERENCES AVAILABLE UPON REQUEST