REAL TIME SURVEILLANCE WITH SMS ALERT (RTSS)

By:

Maj Syed Furqan Farooq
Capt Zarrar Tariq
Capt Babar Sultan

Submitted to the Faculty of Department of Electrical Engineering,
Military College of Signals, National University of Sciences and Technology,
Islamabad
In partial fulfillment for the requirements of a B.E Degree in
Telecom Engineering
June 2015

ABSTRACT

(RTSS)

Due to the increasing security risks in the modern world, it is a critical requirement to understand global vulnerabilities and protect everyone’s property from destruction, theft and other malicious intentions. Video surveillance and monitoring are among the latest and fascinating technologies highlighting these concerns. The project Real Time Surveillance with SMS alert (RTSS) mainly focuses on the real time detection of motion (human) in the target area. Its main purpose is surveillance that does not require continuous human monitoring as it triggers an alarm and sends a SMS via GSM module to alert a distant user regarding human intrusion in the area of interest. Wavelet based technology has been employed to reduce computational complexity of the algorithm in real time application. Traditionally the surveillance is being done by simply implying CCTV camera system which requires constant human observation for any suspicious or unwanted act in the area of coverage. Inculcating the human detection technique in the existing CCTV network will reduce the human effort required which is prone to many flaws and short falls.

CERTIFICATE

It is hereby certified that the contents and form of the project report entitled “Real Time Surveillance with SMS Alert (RTSS)”, submitted by the syndicate of :

1. Maj Syed Furqan Farooq
2. Capt Zarrar Tariq
3. Capt Babar Sultan

has been found satisfactory as per the requirement of the B.E. Degree in Electrical (Telecom) Engineering.

Supervisor:

Col Imran Tauqir

Department of EE

MCS, NUST

DECLARATION

No content of work presented in this thesis has been submitted in support of another award of qualification or degree either in this institution or anywhere else.

DEDICATED TO

Almighty Allah,

Faculty for their help

And our family and friends for their support

ACKNOWLEDGEMENTS

Praise be to Allah, the Magnificent and the Merciful, for showering us with His countless blessings and for giving us the strength and patience to bear all hardships. Whatever we have achieved, we owe to Him, in totality. We are grateful to our parents without their prayers, support and hard work this would not have been possible. We are thankful to our teachers and especially our supervisor, Col Imran Tauqir, for being our torch bearer and guiding us when else failed.

We would like to thank our family for bearing with us patiently and our friends who have been there to support us morally and technically.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

3D	Three Dimensional
COD	Cascade Object Detector
CHT	Classical Hough Transform
FPS	Frames per Second
GHT	Generalized Hough Transform
GSM	Global System for Mobile communication
GUI	Graphical User Interface
HOG	Histogram of Oriented Gradients
ICSP	In-Circuit Serial Programming
RTSS	Real Time Surveillance with SMS alert
ROI	Region of Interest
SMS	Short Message Service

Chapter 1

1. Introduction

1.1 Background/Motivation

The motivation to undergo this project was to enhance the security of a home, office or any premises where unwanted human presence must be detected and reported in real time to ensure the territorial integrity. Computational complexity has been reduced using wavelet base technology hence making RTSS efficient. This project also reduces the human effort required for surveillance and actions to detect and report it. It provides evidence of motion triggered after a breach has occurred in the area of interest hence using the memory intelligently rather than having complete video backup of 24/7 which requires huge hard disks.

In modern society, security systems such as houses equipped with alarm systems are very popular means of crime deterrence. The rapidly flourishing market for security systems brings fascinating technologies and associated features. However, many security systems are unable to effectively operate and at the same time are not cost effective. The security systems are not real time and there is a delay in detection and generation of alarm which makes the system nonproductive as immediate remedial measures can’t be taken to stop the intrusion there and then. It is very common in the security systems that the alarm gets triggered because of an animal or bird or movement of objects due to wind in the target area because of the use of traditional motion detection techniques which jeopardizes the usefulness and reliability of the complete system. Mostly present security systems require huge data storage capacity to provide video backup which is not a practical approachas 24/7 recording is of no use because one only requires video backup of the duration when intrusion or territorial integrity is being challenged.

1.2 Project Description

For motion detection, first we capture live video frames of the target area under surveillance. This is achieved by using a camera which continuously provides a sequence of video frames at a particular speed of FPS.

Image processing apply in this system to initially detect any object movement and after detecting a motion efficiently, identifying human rather than creating a false alarm created by the presence of an animal or wind in the target area . After the human detection phase the tracking part comes into play. The human in the camera field of view will be tracked. To apply these techniques some analysis must be done. In this project, the analysis is done by using MATLAB's Computer Vision. In real time systems processing delay is undesirable and can jeopardize the complete system, therefore we have used the wavelet based technology to reduce the computational time manifold hence making RTSS efficient.

If human is detected, immediately an alarm is generated with the help of speakers and a SMS alert is sent to a stored number using the GSM module (SIM 900) which is interfaced with the system using Arduino UNO board. It is required to store such motion to make it available to the user in the near future. Thus, there is no need to store the video 24/7 which reduces the storage requirements and makes the system more efficient using less storage resources. This also helps the user to provide an evidence of any inappropriate activity.

1.2.1 Schematic Diagram

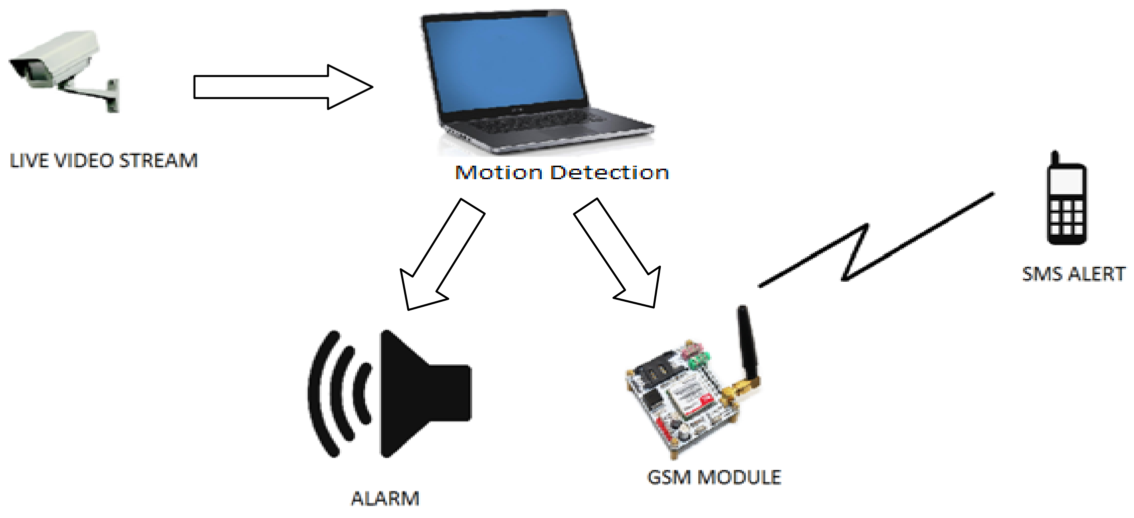


Figure 1: Schematic of RTSS

1.2.2 Methodology of RTSS

The conduct methodology can be divided in two main parts

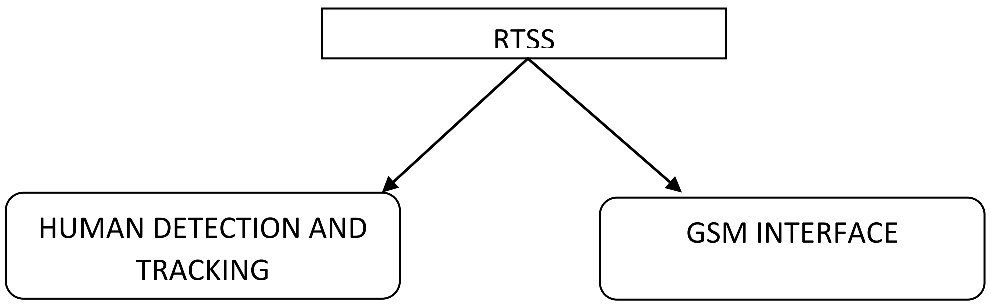


Figure 2: Main parts of RTSS

1.2.3 Application Areas

a. Home Security

The significance of a security system in home is incomparable. Rather than worrying about leaving home unattended, RTSS can be applied on the outer parameter of the house or in the house and whenever there is any unauthorized entry it will trigger the alarm and will send a SMS to the user or police station or any other person in the neighbours which have been previously asked to help in the time of need. RTSS can also be applied on the important places in the house where people keep their valuables when they have to leave the house.

b. Vehicle protection

In today's challenging economy it is very difficult to have a vehicle and if a person has it he can't afford to lose it. RTSS can be off great help as it provides real time protection. RTSS can be applied in the garages so it will trigger an alarm if an unauthorized person enters the garage and even if he manages to steal the vehicle one will have the evidence of the person to apprehend him in the court.

c. Border Control

Border Control is an established state-coordinated effort to achieve operational control of the country's border with the priority mission of supporting the homeland's security against threats including smuggling, illegal cross border traffic, terrorism, and criminal activities. The rate at which violent crime decreases and society's security increases is significantly improved by efficient border control.Real time surveillance system alarms on unauthorized crossings and provides database of border crossings.

d. Bank Security

Banks are the places where people keep their hard earned money and valuables thus bank becomes a lucrative target for the robbers and now a days the cases of bank robberies have gone sky high. RTSS can be applied in the bank most vulnerable areas like cashier booths, vaults after the service timings of the banks to detect any human motion and sound an alarm and informing the law enforcement agencies to react immediately without any delay due to the SMS feature.

1.3 Scope, Specifications and Deliverables

1.3.1 Scope

Area of interest will be monitored using a camera and captured frames will be processed continuously to detect any human presence and motion. An alarm will be generated upon human detection and human motion will be continuously tracked in the target area. Efficient storage of only those frame in which human is present to reduce the huge storage requirement. With the help of a GSM module a SMS will be sent to a pre stored number.These scopes are determined in order to complete this project:-

- Application of image processing technique by using MATLAB Computer Vision.
- GSM module interfacing.
- Thorough understanding of MATLAB its toolboxes and algorithms.
- Implementing wavelet based technology to reduce computational complexity and making RTSS efficient.

1.3.2 Design Specifications

a. Camera

Logitech QuickCam Orbit AF

Device Type	:	Webcam
Pixels	:	2 MP
Max Digital Video Resolution	:	1600 x 1200
Image Sensor	:	CCD
Interface	:	USB 2.0

Frame Rate	:	30 FPS

Figure 3: Logitech QuickCam Orbit AF

b. Laptop

HP Probook 450 G1

RAM	:	4 GB
Processor	:	Intel Core i5-4200M CPU 2.50 GHz
System	:	64 bit Windows 8.1

c. Matlab

MATLAB 2014a

d. Speaker

Xpod BT-11

e. Arduino

The Arduino Uno is a microcontroller board based on the ATmega328. It consists of 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a power jack, a USB connection, an ICSP header, and a reset button. It is fully equipped to support the microcontroller; we just need a USB cable to connect it to a computer or power it with a battery or AC-to-DC adapter to get started. Following are the technical specifications:-

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V

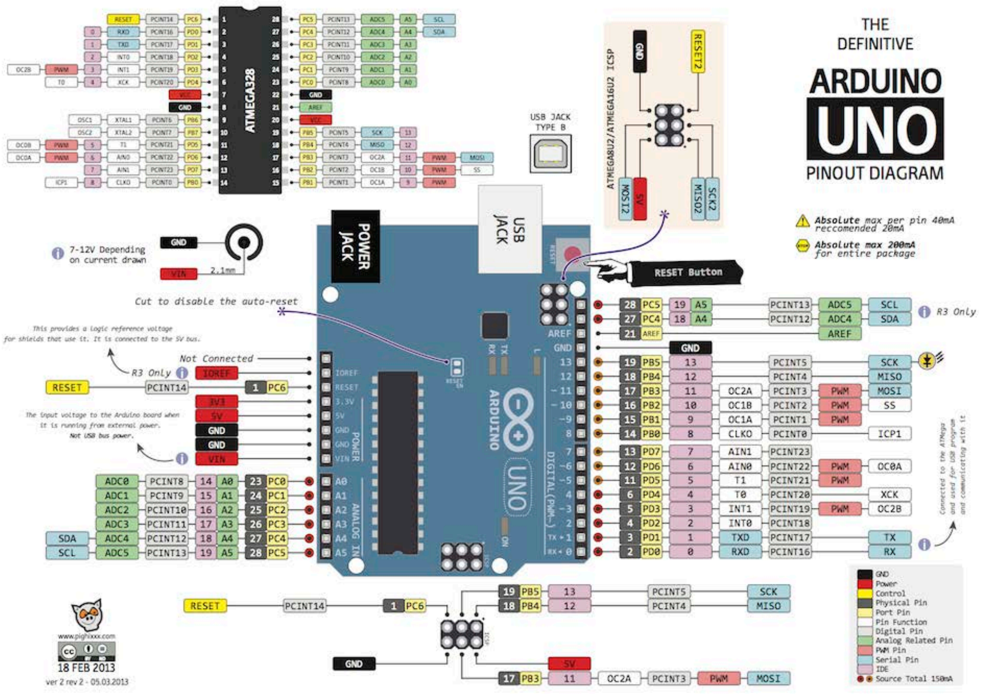
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by boot loader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz
Length	68.6 mm
Width	53.4 mm
Weight	25 g

Figure 4: ARDUINO UNO pinout Diagram

Figure 5: ARDUINO UNO

f. GSM Module

SIM 900GSM/GPRS Modem is built with Dual Band GSM/GPRS engine. SIM900,



operates on frequencies 900/ 1800 MHz.The Modem is available with RS232 interface,

which enables connection with PC and microcontroller with RS232 Chip (MAX232). The baud rate is configurable from 9600-115200 through AT command. The onboard Regulated Power supply allows you to connect wide range unregulated power supply. This modem enables audio calls, SMS, read SMS, attend the incoming calls through simple AT commands.

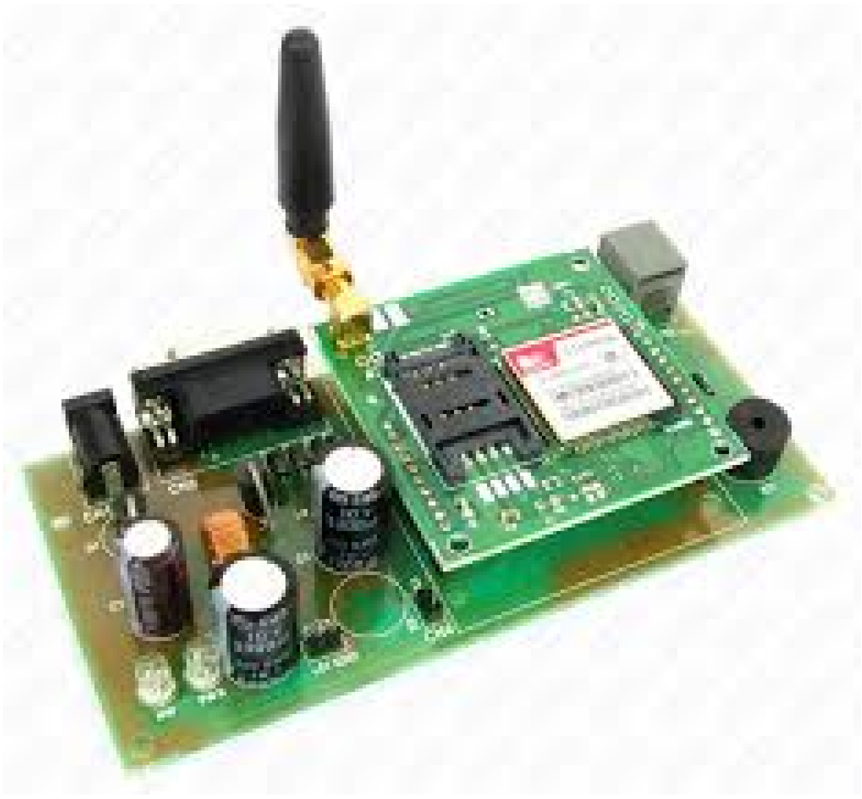


Figure 6: GSM Module

1.3.3 Deliverables

Following are some deliverables of the RTSS:-

- Human Detection in live video stream
- Alarm Generation
- SMS Alert
- Efficient Video Storage

Chapter 2

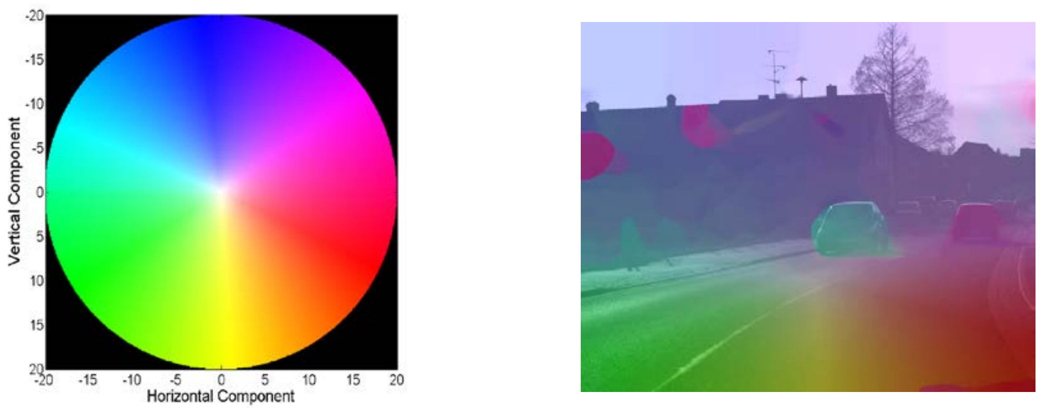
2. Literature Review / Background Study

Considerable work done has been done in the field of motion detection. Many research papers have been published on the said subject using different techniques. References are given at the end. Many algorithms have been devised for motion detection and are mentioned below.

a. Optical Flow

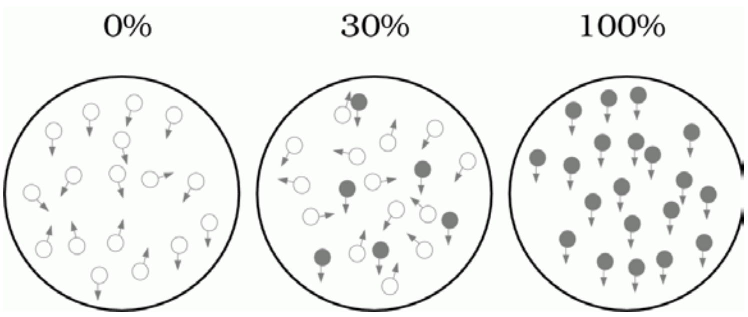
Optical flow is the pattern of apparent motion of objects, surfaces, and edges in a visual scene that result from the relative motion between an observer (an eye or a camera) and the scene. At the pixel level, the optical flow was first proposed by Horn and Schunck almost 20 years ago. The rotation of sphere generates the optical flow in the center. There have been attempts at computing the performance of optical flow techniques including generation of a simple sequence by first distorting the base image. However it is not easy to know whether the base image is matched to its 3D image. The connected difficulties with this algorithm are motion discontinuities, camera noise and shadows.

Figure 7: Optical Flow



b. Cross Correlation

Cross correlation stands as another effective tool to match images. It can be



normalized to allow pattern matching and is quite vigorous to noise. It is a measure of similarity between two images. Motion can be detected by computing the cross correlation coefficient of successive frames. A level is set and if the numerical value is below that acceptable level, motion is depicted in the field of vision. A fundamental problem with most of the cross correlation schemes is the assumption that the image (or a large portion of it) moves as a whole between the two frames. The critical problems for these techniques are the images containing independently moving objects and image distortions induced by the unrestricted motion of objects in space.

Figure 8: Cross Correlation

c. Joint Difference

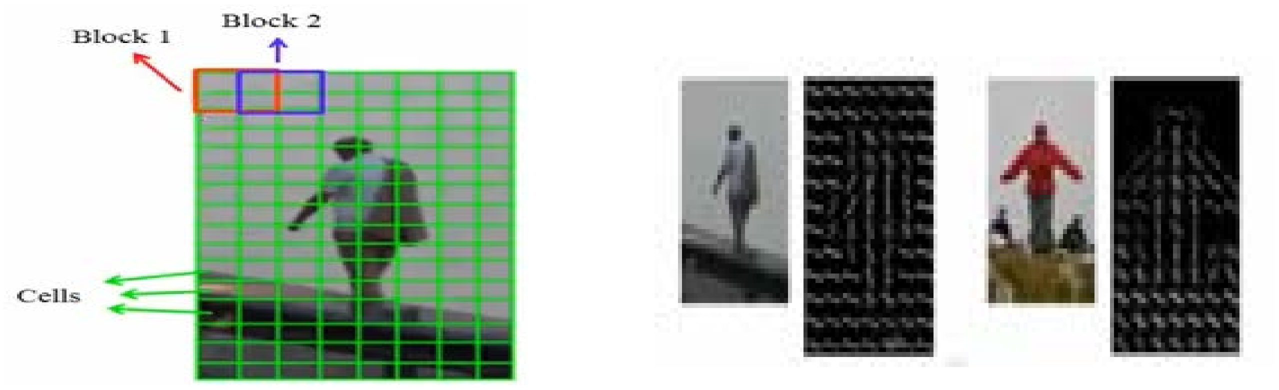
This algorithm uses frames differencing information. It then corrects the pixel classification by the background subtraction algorithm and updates the model in accordance to the classification. The algorithm is not susceptible to global changes and is not efficient for human detection.

d. Histogram of Oriented Gradients

Another method which we are studying to implement human intruder detector is Histogram of Oriented Gradients. HOG is a feature descriptor used in computer vision and image processing aiming to detect objects. The technique counts occurrences of gradient orientation in localized portions of an image. This technique is similar to edge orientation histograms and shape contexts, scale-invariant feature transform descriptors, but differs in the method of computation which is carried out on a dense grid of uniformly spaced cells and uses overlapping local contrast normalization to enhance accuracy.

HOG descriptors were first described by NavneetDalaland Bill Triggs in June 2005, the researchers at the French National Institute for Research in Computer Science and

Control (INRIA). Initially they focused on the problem of pedestrian detection in static



images, later they extended their tests to include human detection in video, film, and a variety of vehicles and common animals in static imagery. The key concept behind HOG descriptors is that shape and appearance of a local object within an image can be illustrated by edge directions or distribution of intensity gradients.These descriptors can be implemented by dividing the image into small connected regions, called cells, and for each cell compiling HOG directions or edge orientations for the pixels within the cell. The descriptor is then represented by the combination of these histograms. The local histograms can be contrast-normalized by calculating a measure of the intensity across a larger region of the image, called a block, and then using this value to normalize all cells within the block.The outcome of this normalization is improved accuracy and it also makes it invariant to changes in illumination or shadowing.

Figure 9: Histogram of Oriented Gradients

e. Hough Transform

The applications of Hough transform as a feature extraction technique are found in computer vision, digital image processing and image analysis. The aim of this technique is to find imperfect instances of objects within a certain class of shapes by a voting procedure. There are two types of Hough transforms:

1) Classical Hough Transform

2) General Hough Transform

Classical Hough Transform (CHT) is mostly employed for the detection of regular geometric shapes, for example circles, lines and rectangles. When a simple analytic description of a feature is quite difficult or impossible, a Generalized Hough Transform (GHT) can be employed. So far we are modeling the human body. In 1981, Dana H. Ballard first introduced GHT. It is based on the principle of template matching and is considered as the modified version of the Hough Transform. This modification enables the Hough Transform to be employed for the detection of an arbitrary object described by its model in addition to the detection of an object described with an analytic equation (e.g. circle, line etc.). One of the solutions to solve the problem of detection of an arbitrary object in an image is to locate the position of the model in the image. GHT transforms the problem of locating the position of the model to a problem of finding the transformation's parameter that maps the model into the image. As long as we know the value of the transformation's parameter, we can determine the position of the model in the image.

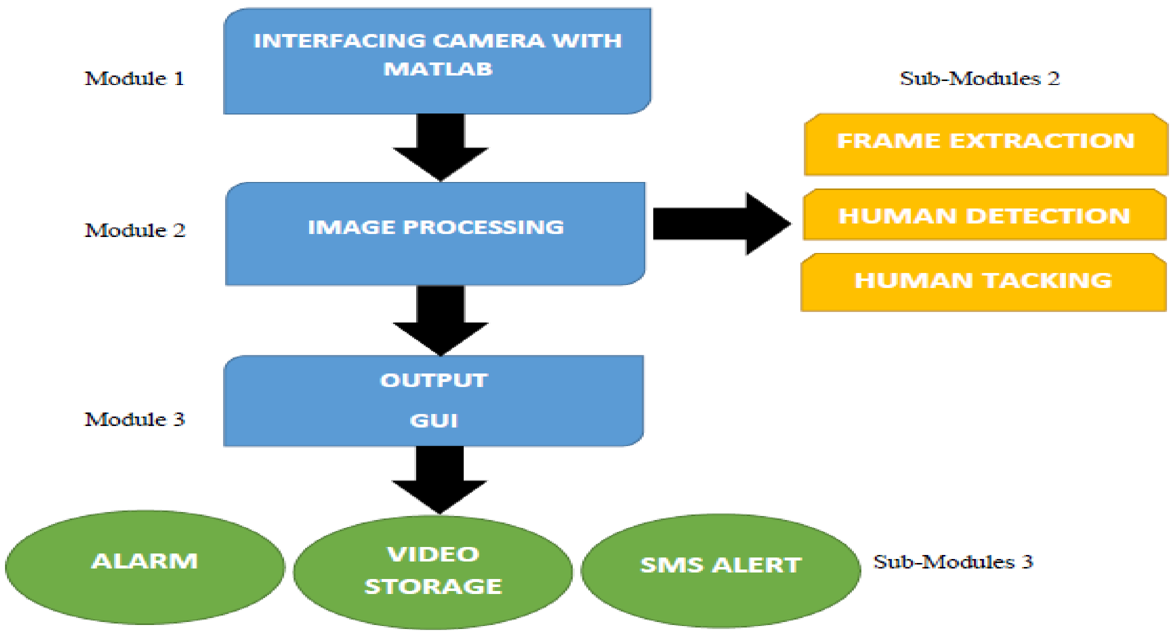
Mapping from orientation of an edge point to a reference point of the shape is carried out by edge information in the original implementation of the GHT. For a binary image, where pixels are either white or black, every black pixel can be a black pixel of the desired pattern thus creating a locus of reference points in the Hough Space. In the image, every pixel votes for its corresponding reference points. The maximum points of the Hough Space indicate possible reference points of the pattern in the image. Maxima can be obtained by scanning the Hough Space or by solving a relaxed set of equations, each of them corresponding to a black pixel. The steps involved in Hough transform are given below:

- Threshold image to get a binary shape image.
- Edge pixels extraction of the shape image.
- R-Table creation of edge gradients that maps gradients to parameter space.
- Accumulator formation with values from the shape.

Chapter 3

3. Detailed Design

3.1 System Model



We divided the RTSS into modules and sub-modules as under:-

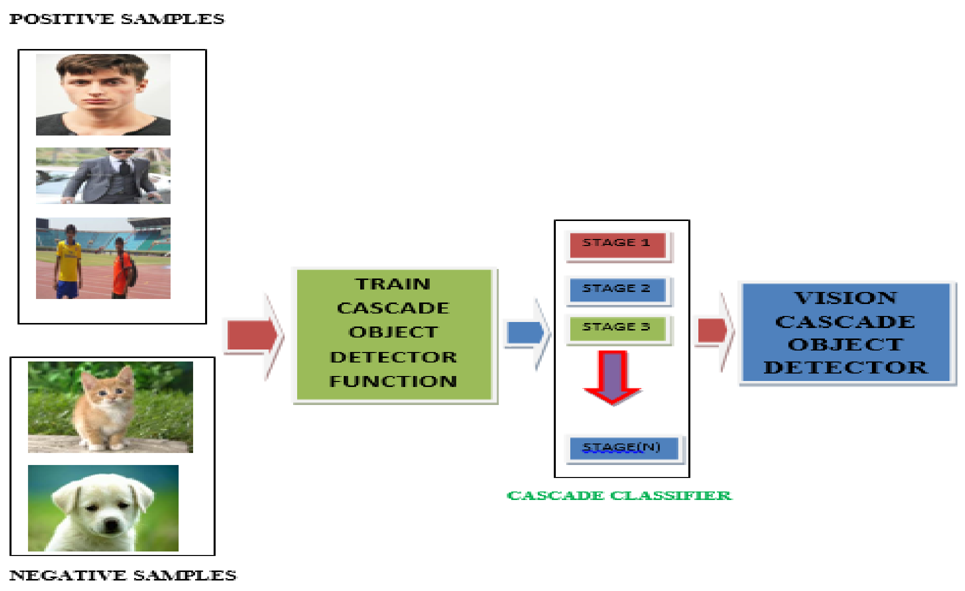
Figure 10: System Model

3.2 Tasks Completed

All modules were completed and implementedsuccessfully. Explanation of tasks completed is as under:-

3.2.1 Human Detection on the Basis of Face

The Computer Vision System Toolbox (CVST) can recognize the object whose image projection attributes does not show significant variation. Objects with approximately fixed aspect ratio include stop signs, faces or cars viewed from one side. The Cascade Object Detector system object uses sliding window method to detect objects in images. Then a cascade classifier is employed to decide whether the window contains the desired object .

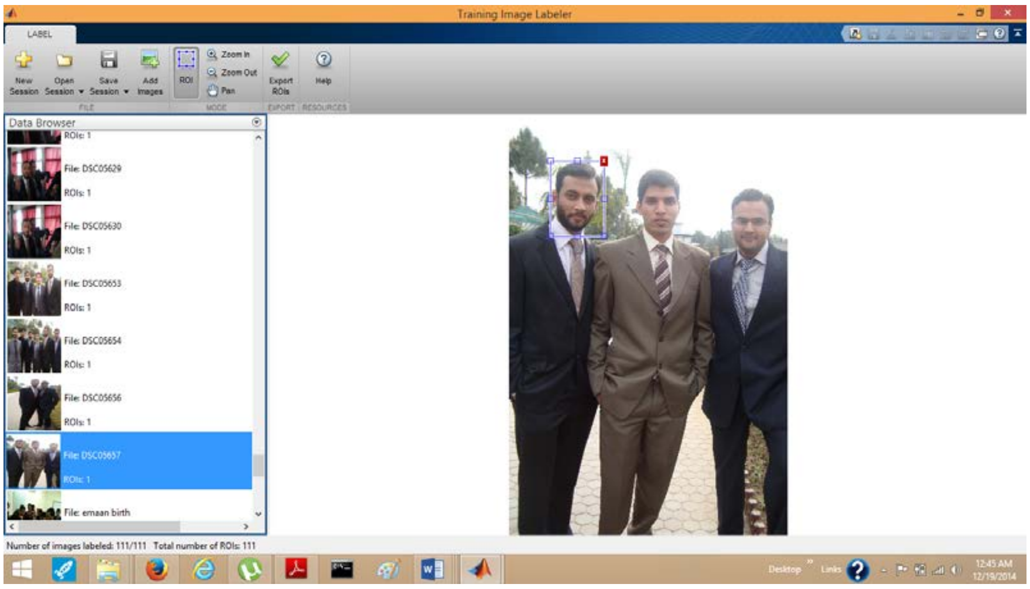


For detection of objects at different scales, the size of the window varies, however its aspect ratio remains unchanged. COD System object comes with several pre-trained classifiers to detect profile faces, frontal faces, eyes and the upper. However, these classifiers may be insufficient for a particular application. CVST provides the train COD function to train a custom classifier. We have done an extensive exercise to make our own detector to identify the object of our own choice by mean of "train COD" function to create a XML file consisting the object of our interest on the basis of which we want to do detection.

Figure 11: Training a Cascade Object Detector

Supply Positive Samples

Positive samples can be specified in two ways. First method is to specify rectangular regions in a larger image. The regions contain the desired objects. The secondmethodology is



to crop out the desired object from the image and save it as a separate image, later which can be specified as the region (entire image). More positive samples can also generated from theexisting ones by varying contrast or brightness, or contrast.

Figure 12: Region of Interest

Supply Negative Samples

Negative samples are not specified explicitly. Instead, the train COD function generates negative samples automatically from user-supplied negative images that do not contain desired objects. Prior to training each new stage, the function runs the detector consisting of the stages already trained on the negative images. If any objects are detected from these, they must be false positives which are used as negative samples. In this way, the mistakes made by previous stages are corrected by training each new stage of the cascade.

The detector's overall false positive rate declines by adding more stages, making it more difficult to generate negative samples. Due to this reason, it is suggested to supply as many negative images as possible. Training accuracy can be enhanced by supplying the negative images that contain backgrounds typically associated with the objects of interest. Furthermore, it is recommended to include negative images that contain no objects similar in appearance to the desired objects. For example, in order to train a stop-sign detector, the negative images should contain other road signs and shapes.

Choose the Number of Stages

There is a trade-off between more stages with a higher false positive rate per stage or fewer stages with a lower false positive rate per stage. Stages with a higher false positive rate contain fewer weak learners than stages with a lower false positive rate, hence the later are more complex.Generally, it is recommended to design large number of simple stages because the false positive rate falls exponentially at each stage. For example, if the false positive rate at each stage is 50%, then for a cascade classifier, the overall false positive rate will be 25% and 12.5% for two stages and three stages respectively, and so on. However, the amount of training data required by the classifier increases with an increase in the number of stages.Furthermore, number of stages has direct relationship with the false negative rate. This increases the probability of wrongly rejecting a positive sample. First, the number of stages (NumCascadeStages) and the false positive rate (FalseAlarmRate) should be set to yield a satisfactory overall false positive rate. Then, experiments can be carried out to tune these two parameters.

In some cases, training might terminate early. For example, even though you set the number of stages to 20, training may terminate after eight stages. This usually happens when the function fails to generate enough negative samples. It is a strong observation that the function does not produce the same result when itis executed again even by setting the number of stages to eight. This happens because of the recalculation of the number of negative and positive samples for each stage with the new number of stages.

The above mentioned steps can be summarized as

- Add images with positive samples to Training Image Labeler. We have used 135 images.
- Specify region of interest (ROI) in every image.

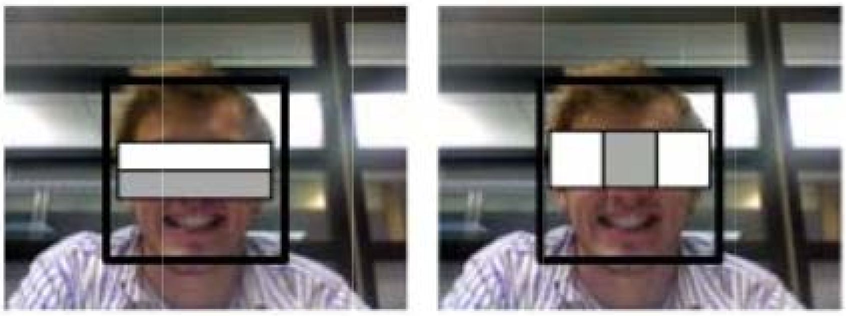
- Create a .mat file.
- Load the data containing positive samples from .mat file. The bounding boxes and the file names are contained in an array of structures named 'data'.
- Add the positive image directory to the MATLAB path.
- Specify the folder for negative images. We have used 171 images.
- Train a cascade object detector

3.2.2 Human Detection and Tracking on the Basis of Upper body

We have utilized the vision COD System object for human detection in a video frame. COD employs the Viola-Jones detection algorithm and a trained classification model for detection proposed by Paul Viola and Michael Jones. They used Haar-like features, digital imagefeatures used in object recognition and they combined them into an efficient scalable classifier. They owe their name to their intuitive similarity with Haar wavelets and were employed in the first real-time face detector. Historically, the task of feature calculation was computationally expensive due to working with image intensities only (i.e., the RGBpixel values at each and every pixel of image). Papageorgiou et al discussed an alternative to working with usual image intensities and proposed a feature set based on Haar wavelets. Viola and Jones developed the so-called Haar-like featuresby adapting the idea of using Haar wavelets. Haar-like feature considers adjacent rectangular regions at a specific location in a detection window, adds up the pixel intensities in each region and calculates the difference between these sums. This difference is then used to categorize subsections of an image. For example, let us consider an image database with human faces. It is a common observation that among all faces the region of the eyes is darker than the region of the cheeks. Therefore, a common Haar feature for face detection is a set of two adjacent rectangles that lie above the eye and the cheek region. The position of these rectangles is defined relative to a detection window that acts like a bounding box to the target object (the face in this case).

In the detection phase of the Viola–Jones object detection framework, a window of the target size is moved over the input image, and the Haar-like feature is calculatedfor each subsection of the image. This difference is then compared to a predefined threshold that separates objects from non-objects. Such Haar-like feature is only a classifieror weak learner, (with slightly better detection quality than random guessing) thus, for sufficient accuracy, a large number of Haar-like features are required. In the Viola–Jones object detection framework, the Haar-like features are therefore organized in something called a classifier cascade to form a strong learner or classifier.

Calculation speedis the key advantage of a Haar-like feature over most other features. It is possible to calculate Haar-like feature in a constant time for any size (approximately 60



microprocessor instructions for a 2-rectangle feature) due to the use of integral images.

Figure 13: Haar-like features (3, 2 Rectangles)

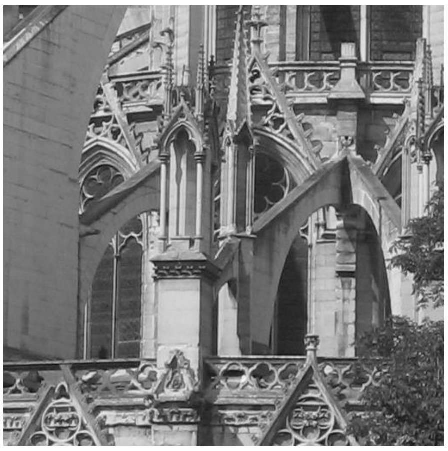
Haar waveletcompressionisanefficientwaytoperform both losslessandlossyimagecompression. Itrelies onaveraging and differencingvaluesinanimagematrixtoproduce amatrix whichissparse ornearly sparse. Asparsematrixisamatrixinwhichalargeportionofitsentriesare0.Asparsematrix canbestoredinan efficientmanner,leadingtosmallerfilesizes.

Inthesenoteswewillconcentrateongrayscaleimages;however,rgbimagescanbehandled bycompressing eachofthecolorlayersseparately.Thebasicmethod istostartwithanimageA,whichcanberegarded asan m×nmatrix with values0to255. InMatlab,this wouldbeamatrix with unsigned8-bit integer values. We then subdividethisimageinto8×8blocks,paddingasnecessary. Itisthese8×8blocksthatweworkwith.

Below is a 512 × 512 pixel grayscale image of the flying buttresses of the Notre Dame Cathedral in Paris:

Figure14: Flying Buttresses of Notre Dame de Paris

And the following is the upper left 8 × 8 section of our image.



$$A = \begin{pmatrix} 88 & 88 & 89 & 90 & 92 & 94 & 96 & 97 \\ 90 & 90 & 91 & 92 & 93 & 95 & 97 & 97 \\ 92 & 92 & 93 & 94 & 95 & 96 & 97 & 97 \\ 93 & 93 & 94 & 95 & 96 & 96 & 96 & 96 \\ 92 & 93 & 95 & 96 & 96 & 96 & 96 & 95 \\ 92 & 94 & 96 & 98 & 99 & 99 & 98 & 97 \\ 94 & 96 & 99 & 101 & 103 & 103 & 102 & 101 \\ 95 & 97 & 101 & 104 & 106 & 106 & 105 & 105 \end{pmatrix}$$

We will concentrate on the first row:

$$r1 = \begin{pmatrix} 88 & 88 & 89 & 90 & 92 & 94 & 96 & 97 \end{pmatrix}$$

Our transformation process will occur in three steps. The first step is to group all of the columns in pairs:

$$[88 \quad 88],[89 \quad 90],[92 \quad 94],[96 \quad 97]$$

We replace the first 4 columns of r1 with th average of these pairs and replace the last 4 columns of r1 with $\frac{1}{2}$ of the difference of these pairs. We will denote this new row as r1h1:

$$r1h1 = (88 \quad 89.5 \quad 93 \quad 96.5 \quad 0 \quad -0.5 \quad -1 \quad -0.5)$$

The first 4 entries are called the approximation coefficients and the last 4 are called detail coefficients. Next, we group the first 4 columns of this new row:

$$[88 \quad 89.5],[93 \quad 96.5]$$

And replace the first two columns of r1h1 with the average of the pairs and the next 2 columns of r1h1 with $\frac{1}{2}$ of the difference of theses pairs. We leave the first row of r1h1 unchanged. We will denote this second row as r1h1h2.

$$r1h1h2 = (88.75 \quad 94.75 \quad -0.75 \quad -1.75 \quad 0 \quad -0.5 \quad -1 \quad -0.5)$$

Finally, our last step is to group the first 2 entries of r1h1h2
together:

$$[88.75 \quad 94.75]$$

And replace the first column of r1h1h2 with the average of the pairs and the second column of r1h1h2 with $\frac{1}{2}$ of the difference of these pairs. We leave the last 6 rows of r1h1h2 unchanged. We denote this last new row as r1h1h2h3:

$$r1h1h2h3 = (91.75 \quad -3 \quad -0.75 \quad -1.75 \quad 0 \quad -0.5 \quad -1 \quad -0.5)$$

We then repeat this process for remaining rows of A. After this, we repeat the same process to columns of A, grouping rows in the same manner as columns. The resulting matrix is:

$$\begin{pmatrix} 96 & -2.0312 & -1.5312 & -0.2188 & -0.4375 & -0.75 & -0.3125 & 0.125 \\ -2.4375 & -0.0312 & 0.7812 & -0.7812 & 0.4375 & 0.25 & -0.3125 & -0.25 \\ -1.125 & -0.625 & 0 & -0.625 & 0 & 0 & -0.375 & -0.125 \\ -2.6875 & 0.75 & 0.5625 & -0.0625 & 0.125 & 0.25 & 0 & 0.125 \\ -0.6875 & -0.3125 & 0 & -0.125 & 0 & 0 & 0 & -0.25 \\ -0.1875 & -0.3125 & 0 & -0.375 & 0 & 0 & -0.25 & 0 \\ -0.875 & 0.375 & 0.25 & -0.25 & 0.25 & 0.25 & 0 & 0 \\ -1.25 & 0.375 & 0.375 & 0.125 & 0 & 0.25 & 0 & 0.25 \end{pmatrix}$$

Notice that this resulting matrix has several 0 entries and most of the remaining entries are close to 0. This is a result of the differencing and the fact that adjacent pixels in an image generally do not differ by much. We will now discuss how to implement this process using matrix multiplication which is called as Haar wavelet transform matrix.

If we let,

$$H_1 = \begin{pmatrix} \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & -\frac{1}{2} & 0 \\ 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} \\ 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & -\frac{1}{2} \end{pmatrix}$$

Then AH_1 is equivalent to the first step above applied to all of the rows of A. In particular,

$$AH_1 = \begin{pmatrix} 88 & 89.5 & 93 & 96.5 & 0 & -0.5 & -1 & -0.5 \\ 90 & 91.5 & 94 & 97 & 0 & -05 & -1 & 0 \\ 92 & 93.5 & 95.5 & 97 & 0 & -035 & -0.5 & 0 \\ 93 & 94.5 & 96 & 96 & 0 & -0.5 & 0 & 0 \\ 92.5 & 95.5 & 96 & 95.5 & -0.5 & -0.5 & 0 & 0.5 \\ 93 & 97 & 99 & 97.5 & -1 & -1 & 0 & 0.5 \\ 95 & 100 & 103 & 101.5 & -1 & -1 & 0 & 0.5 \\ 96 & 102.5 & 106 & 105 & -1 & -1.5 & 0 & 0 \end{pmatrix}$$

Similarly, by defining H_2 by:

$$H_2 = \begin{pmatrix} \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Then AH_1H_2 is equivalent to the two steps above applied to all of the rows of A. In particular,

$$AH_1H_2 = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Then $AH_1H_2H_3$ is equivalent to the all three steps above applied to all of the rows of A. In particular,

$$AH_1H_2H_3 = \begin{pmatrix} 91.75 & -3 & -0.75 & -1.75 & 0 & -0.5 & -10 & -0.5 \\ 93.125 & -2.375 & -0.75 & -1.5 & 0 & -1.5 & -10 & 0 \\ 94.5 & -1.75 & -0.75 & -0.75 & 0 & -0.5 & -0.5 & 0 \\ 94.875 & -1.125 & -0.75 & 0 & 0 & -0.5 & 0 & 0 \\ 94.875 & -0.875 & -1.5 & 0.25 & -0.5 & -0.5 & 0 & 0.5 \\ 96.625 & -1.625 & -20 & 0.75 & -10 & -10 & 0 & 0.5 \\ 99.875 & -2.375 & -2.5 & 0.75 & -10 & -10 & 0 & 0.5 \\ 102.375 & -3.125 & -3.25 & 0.5 & -10 & -1.5 & 0 & 0 \end{pmatrix}$$

If we let H be the product of these three matrices, then

$$H = H_1H_2H_3 = \begin{pmatrix} \frac{1}{8} & \frac{1}{8} & \frac{1}{4} & 0 & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{8} & \frac{1}{8} & \frac{1}{4} & 0 & -\frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{8} & \frac{1}{8} & -\frac{1}{4} & 0 & 0 & \frac{1}{2} & 0 & 0 \\ \frac{1}{8} & \frac{1}{8} & -\frac{1}{4} & 0 & 0 & -\frac{1}{2} & 0 & 0 \\ \frac{1}{8} & -\frac{1}{8} & 0 & \frac{1}{4} & 0 & 0 & \frac{1}{2} & 0 \\ \frac{1}{8} & -\frac{1}{8} & 0 & \frac{1}{4} & 0 & 0 & -\frac{1}{2} & 0 \\ \frac{1}{8} & -\frac{1}{8} & 0 & -\frac{1}{4} & 0 & 0 & 0 & \frac{1}{2} \\ \frac{1}{8} & -\frac{1}{8} & 0 & -\frac{1}{4} & 0 & 0 & 0 & -\frac{1}{2} \end{pmatrix}$$

To apply the procedure to the columns, we just multiply A on the left by H^T. So the resulting matrix is

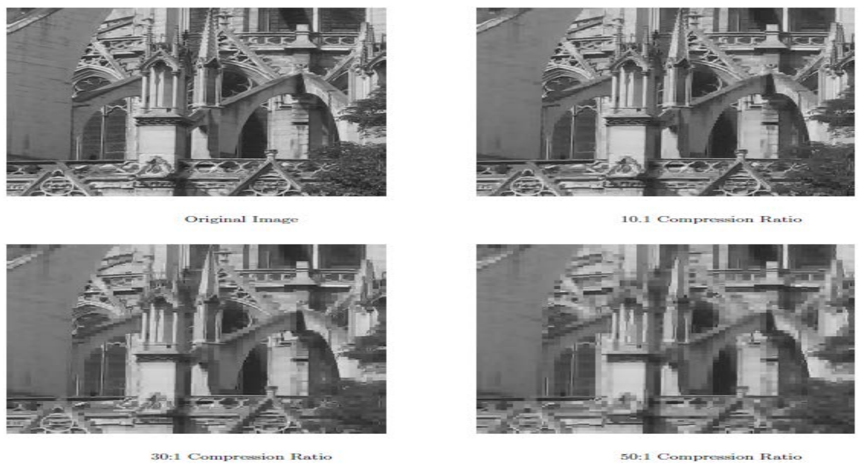
$$H^TAH = \begin{pmatrix} 96 & -2.0312 & -1.5312 & -0.2188 & -0.4375 & -0.75 & -0.3125 & 0.125 \\ -2.4375 & -0.0312 & 0.7812 & -0.7812 & 0.4375 & 0.25 & -0.3125 & -0.25 \\ -1.125 & -0.625 & 0 & -0.625 & 0 & 0 & -0.375 & -0.125 \\ -2.6875 & 0.75 & 0.5625 & -0.0625 & 0.125 & 0.25 & 0 & 0.125 \\ -0.6875 & -0.3125 & 0 & -0.125 & 0 & 0 & 0 & -0.25 \\ -0.1875 & -0.3125 & 0 & -0.375 & 0 & 0 & -0.25 & 0 \\ -0.875 & 0.375 & 0.25 & -0.25 & 0.25 & 0.25 & 0 & 0 \\ -1.25 & 0.375 & 0.375 & 0.125 & 0 & 0.25 & 0 & 0.25 \end{pmatrix}$$

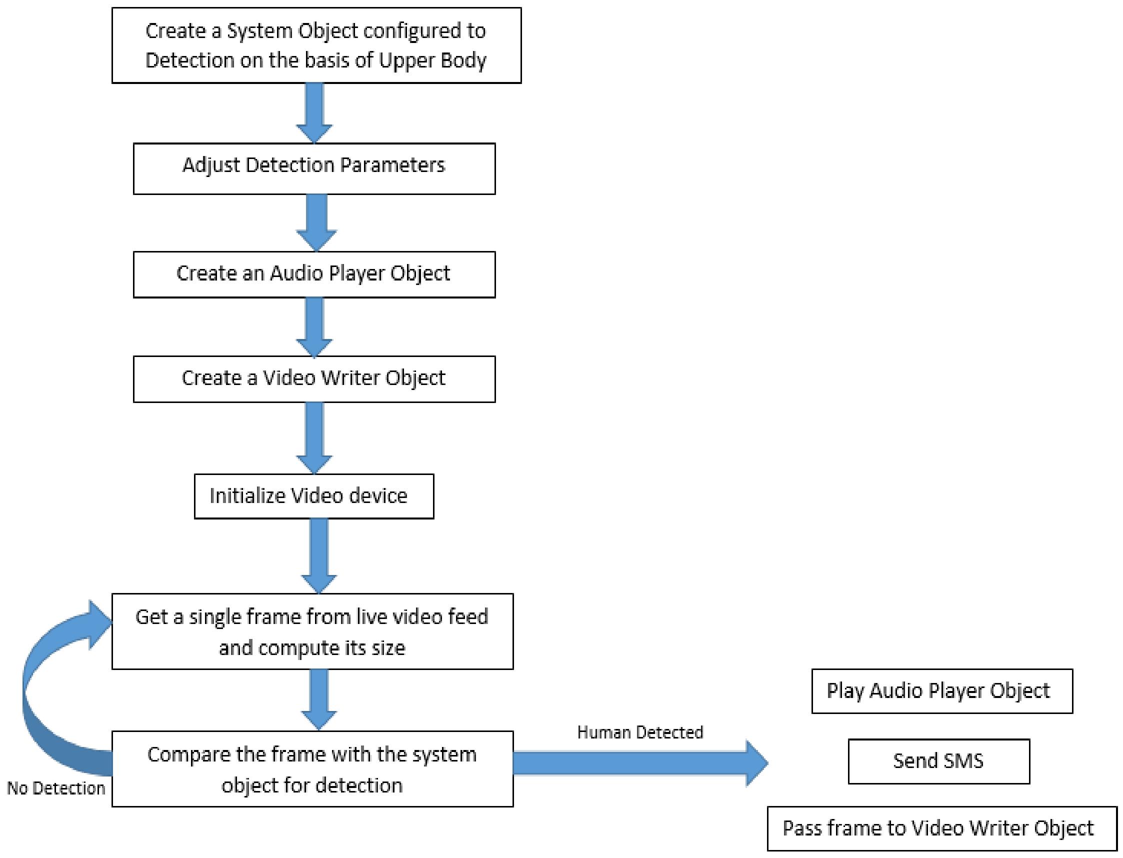
The important part of this process is that it is reversible. In particular, H is an invertible matrix. If

$B = H^T AH$ Then $A = \left(H^T\right)^{-1} MH^{-1}$

B represents the compressed image of A (in the sense that it is sparse). Moreover, since H is invertible, this compression is lossless. Some examples of our beginning image, compressed by various ratios are given below.

Figure 15: Different Compression Ratios





Create a System Object configured to Detection on the basis of Upper Body
Adjust Detection Parameters
Create an Audio Player Object
Create a Video Writer Object
Initialize Video device
Get a single frame from live video feed and compute its size
Compare the frame with the system object for detection
No Detection
Human Detected
Play Audio Player Object
Send SMS
Pass frame to Video Writer Object

The algorithm for the Human Detection on the basis of Upper Body is as under:-

Figure 16: Upper Body Algorithm

3.2.3 GUI Development

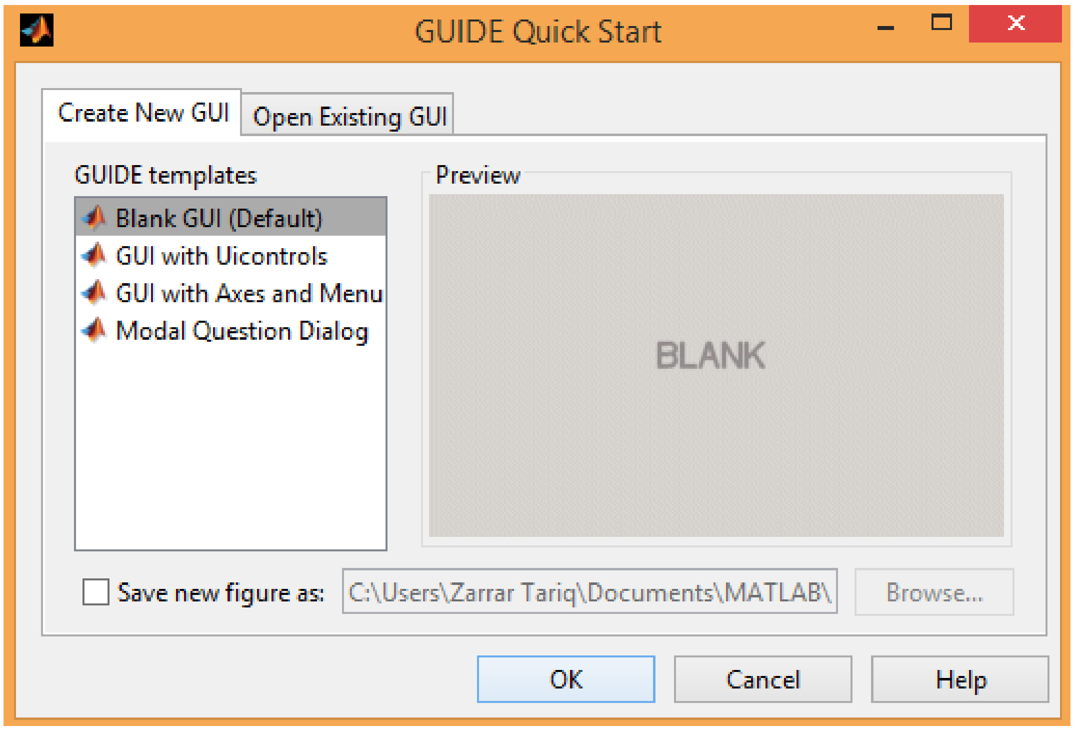
A graphical user interface (GUI) is a type of interface that enables a user to perform interactive tasks through visual indicators and graphical components. Unlike coding programs to accomplish tasks, the user of a GUI need not to create a script or type commands at the command line to perform the tasks and need not to understand the details of how the tasks are performed. Toolbars, menus, radio and push buttons, sliders and list boxes are some of the GUI components. GUIs created employing MATLAB tools can also read and write data files, perform any type of computation, communicate with other GUIs, and display data as plots or tables. Typically, GUIs wait for an end user to manipulate a control, and then respond to each user action. Each control, and the GUI itself, has one or more callbacks, named for the fact that they "call back" to MATLAB to ask it to do things. A particular user action, passing the cursor over an icon or pressing a screen button triggers the execution of each callback. The GUI then responds to these events. Event handling by the components is defined by callbacks written by GUI creator.

Such programming is often referred to as event-driven programming in whichthere is asynchronous callback execution and the events external to the software trigger callback execution. In MATLAB, GUI can respond to different kind of events although most of the events are user interactions with the GUI, for example, connecting a device to the computer orthe creation of a file. Callbacks can be coded in two distinct ways:

- As MATLAB language functions stored in files
- As strings containing MATLAB expressions or commands

It is preferred to use functions stored in code files as callbacks rather than using the strings, because functions are more flexible and powerful and they have access to arguments. MATLAB scripts (sequences of statements stored in code files) cannot be used as callbacks. Although callback can be provided with certain data and we can make it do anything we want, however we have no control over when callback executes. That is, while using GUI, we have no control over the sequence of events that trigger particular callbacks or which other callbacks might still be running at the same time. This is the distinguishing feature of event-driven programming from other types of control flow, for example, processing sequential data files. We have created a GUI using callbacks which are expressions and strings. The detailed procedure is explained step by step below:

1) Goto New > Graphical User Interface and select Create New GUI > Blank



GUI.

Figure 17: Blank GUI

2) A .fig will open .Here we have to place axes and pushbuttons and static textboxes. We are using axes to display our video output and static images for the layout of our GUI and push buttons to interact with the GUI.

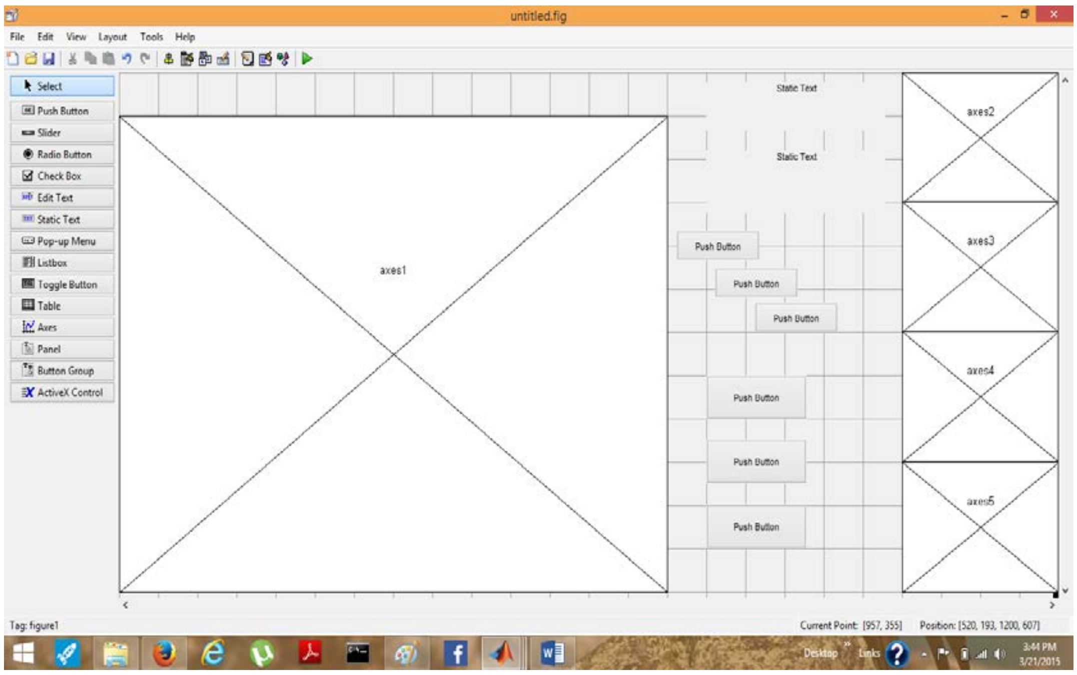
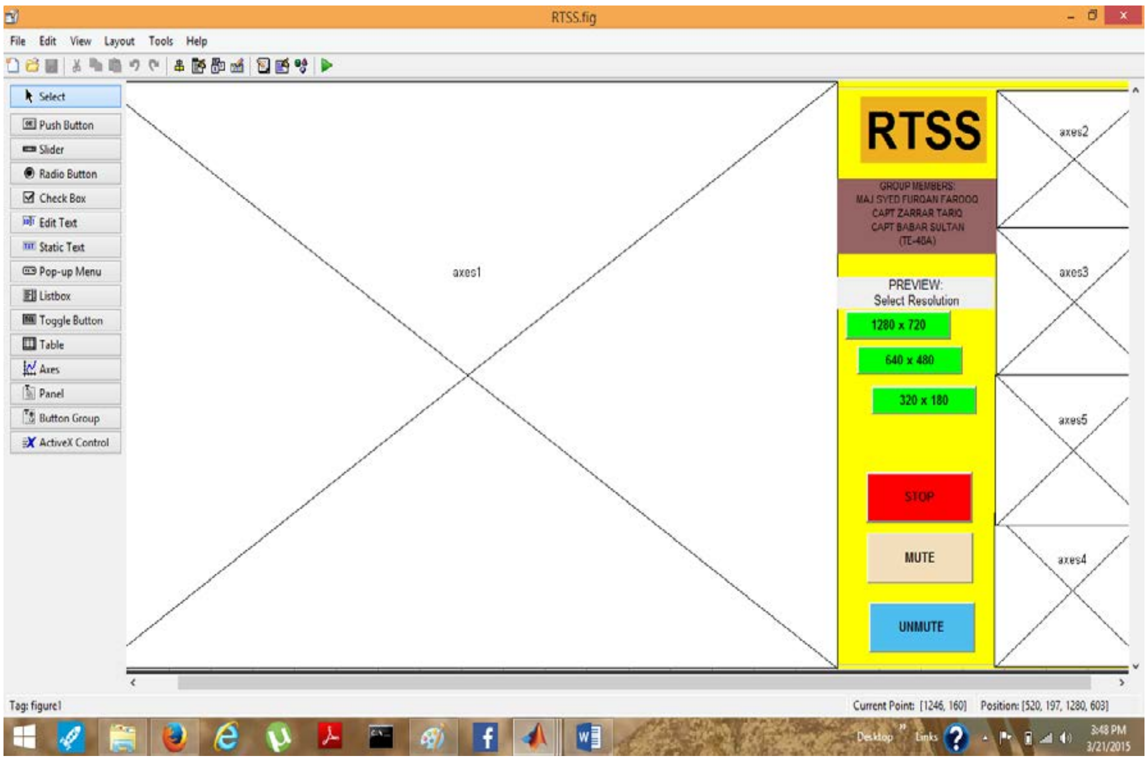


Figure 18:GUI with AXES and PUSHBUTTONS

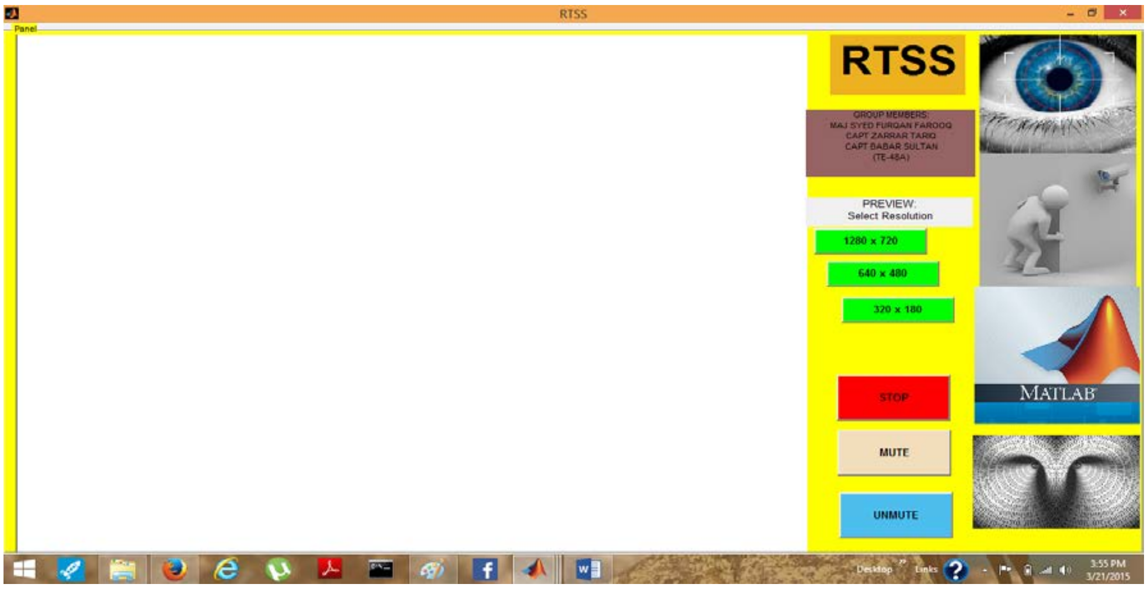
3) After making the axes and pushbuttons we have to do the editing and after the



editing the GUI looks like:

Figure 19: GUI with AXES and PUSHBUTTONS Edited

4) In the command window now we have to edit the .m file which has been created automatically to assign each axes a figure and each pushbutton its respective callback function which the GUI executes when we interact with it. The final outlook of the GUI when



it run becomes:

Figure 20: GUI Output

3.2.4 Alarm Generation

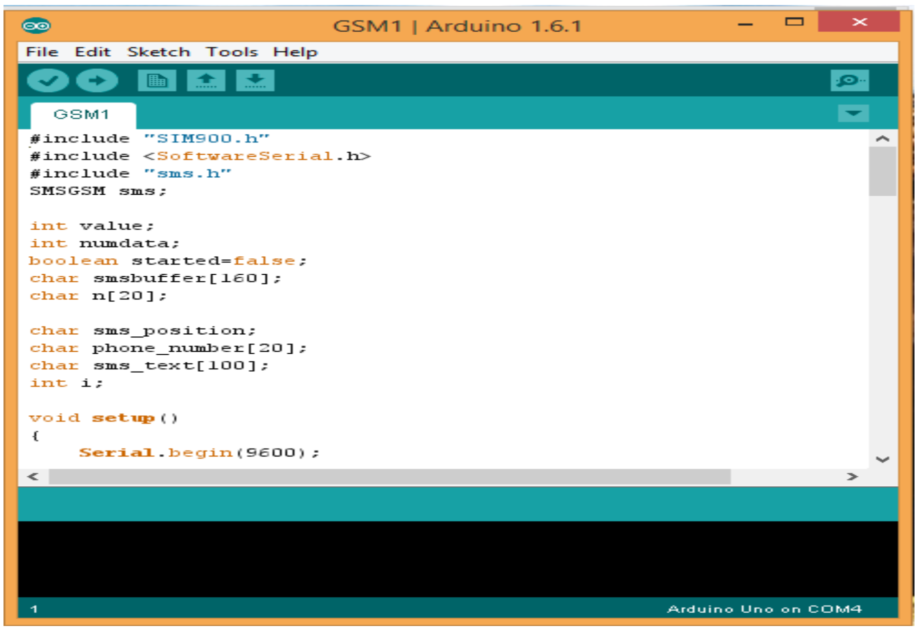
In RTSS an alarm is sounded whenever a human is detected in the target area. In MATLAB to play a sound we have to first get its data and sampling rate which is done with the help of audio read() function which returns the data and sampling rate of the audio file. Single-channel (mono) audio data is specified by an m-by-1 column vector and stereo playback by an m-by-2 matrix, where m denotes the number of audio samples. If data is in the form of m-by-2 matrix, then the first and the second columns correspond to the left and the right channel respectively. Stereo playback is available only if the system supports it. The range of samples of data is from –1.0 to 1.0. Then we have to create an audio player object which takes the data samples and the sampling rate info ,which can then be played with the help of play() command.

3.2.5 Video Storage

Efficient Video storage is also a feature of RTSS which ensure minimum usage of data storage space which is a very efficient approach by only recording those frames in which human motion is present and is of concern. We have to create a Video writer object and then subsequently place each frame in which human is present and specify the FPS of the playback.

3.2.6 GSM Interfacing

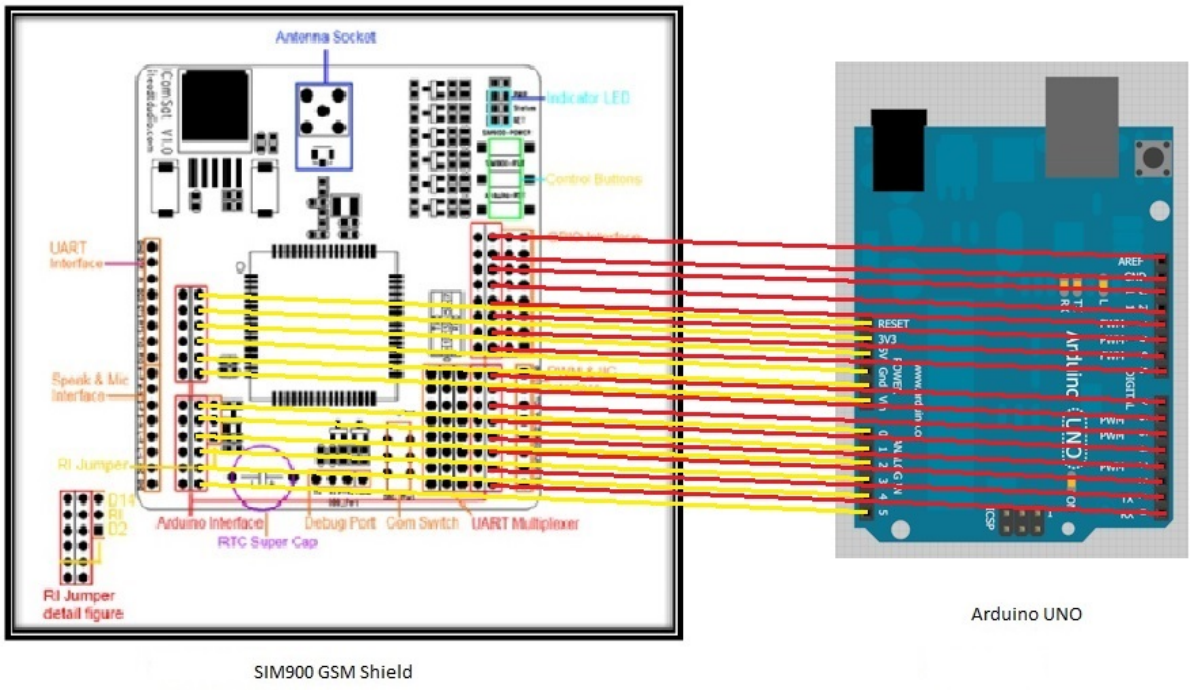
One of the cardinal feature of RTSS is its SMS alert which makes it a really practical and effective system in the vague of current security environment. After the successful detection of human in the target area a SMS is sent to a distant user hence alerting him of undesired human intrusion upon which he can take the necessary counter measure. Primarily, this feature is more concerned with the home security. We have used the Arduino Uno as a platform to connect the SIM 900 GSM module and sending a SMS alert. Arduino Uno is programmed and configured with the help of its interface.



```
#include "SIM900.h"
#include <SoftwareSerial.h>
#include "sms.h"
SMSGSM sms;

int value;
int numdata;
boolean started=false;
char smsbuffer[160];
char n[20];

char sms_position;
char phone_number[20];
char sms_text[100];
int i;

void setup()
{
    Serial.begin(9600);
```

Figure 21: Arduino Interface

After uploading the code to the Arduino Uno board the connections are made with the SIM 900.We have used the IComsat which is the GSM/GPRS shield for Arduino based on the SIM900 Quad band(850/900/1800/1900MHz) GSM/GPRS module. It is fully compatible with the Arduino Uno and controlled by the AT commands.IComsat is connected with the



Arduino Uno.

Figure 22: IComsat and Arduino Uno Connection

Once the connections have been made and the code has been uploaded on the Arduino board than the IComsat can be controlled by the Matlab by simply creating an object and setting the parameters like specifying the serial port, baud rate, data bits, parity etc. Then the serial object is opened (COM port). Once the human is detected for the first time it will send the SMS and to avoid congestion it will again check in the 60th frame and its multiples and send SMS accordingly.

Chapter 4

4. Project Analysis and Evaluation

During the development of RTSS we took 2 approaches which are "Human Detection on the basis of Face" and "Human Detection on the basis of Upper Body". Both techniques were thoroughly tested and their results were analyzed and the observations along with results are as under:-

a. In "Human Detection on the basis of Face" approach we created our own classifier in the cascade object detector by providing positive and negative samples and trained the Matlab to perform the detection on the basis of our own classifier. Results were satisfactory but a lot of parameters were to be kept in consideration which are:-

- Number of positive Samples
- Number of negative Samples
- Region of Interest specification
- Number of stages of cascade classifier
- False Alarm rate
- True Positive rate

Due to all the above mentioned parameters and keeping in view the requirements of RTSS "Human Detection on the basis of Upper Body" was selected to be our image processing technique because it was more feasible and appropriate as we are working in a real time environment and dealing with live camera feed. It is also pertinent to mention that Human detection on the basis of Face was not feasible for RTSS because of the distance between the camera and the intruder hence Human detection on the basis of upper body was more suitable and practical because the Upper body features are visible and identifiable even from a longer range.

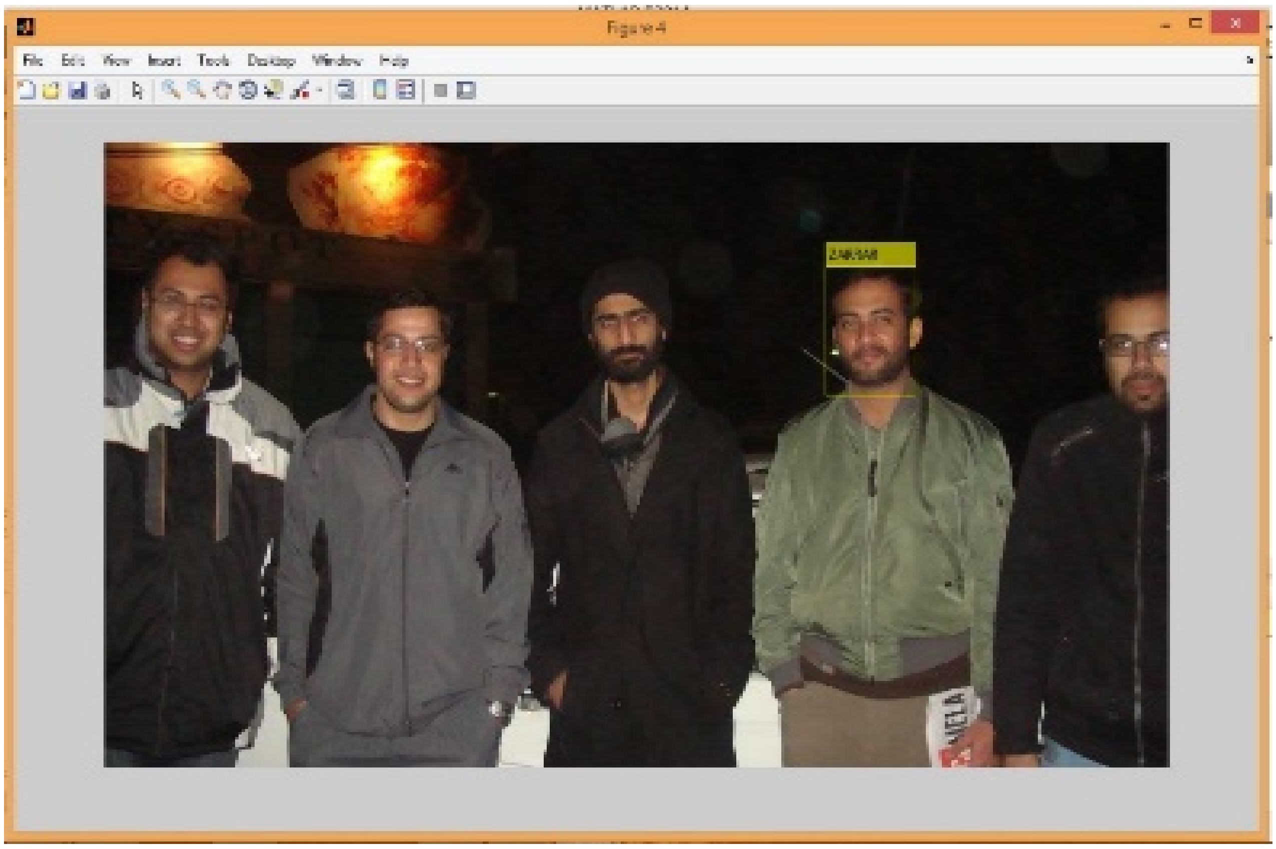


Figure 23: Human Detection on the basis of Face

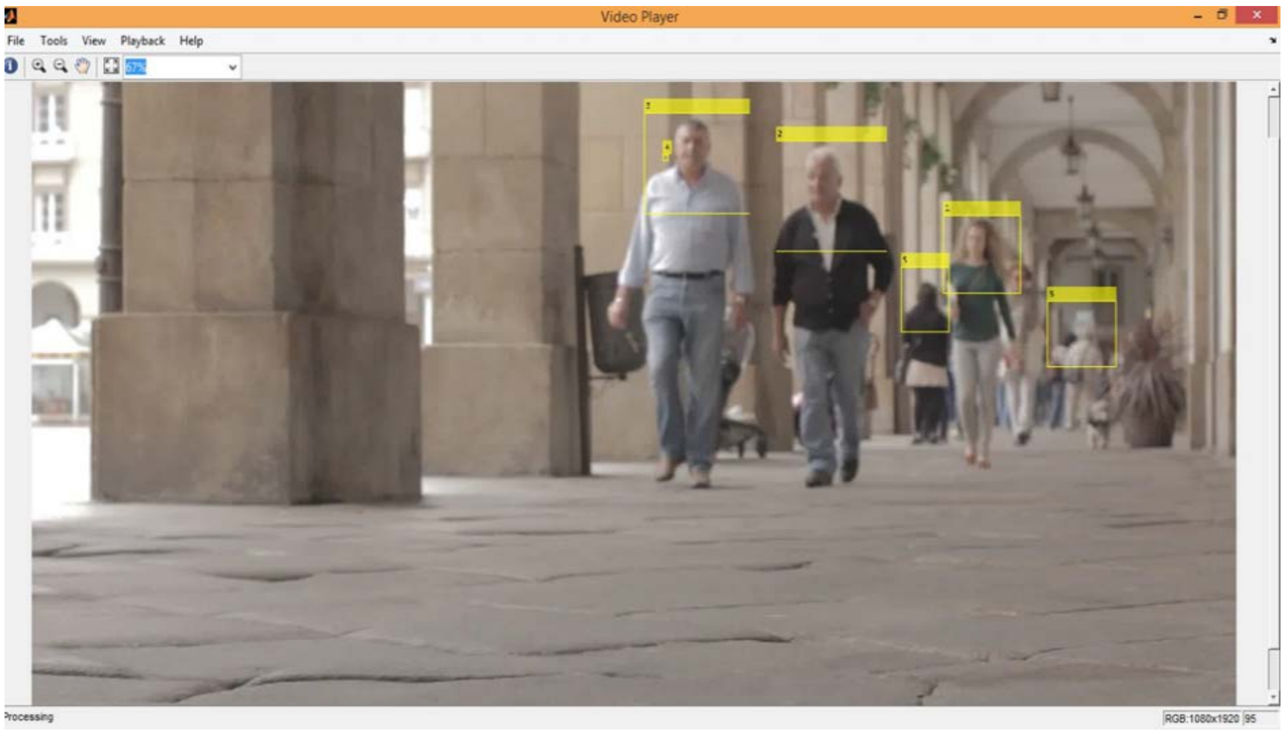
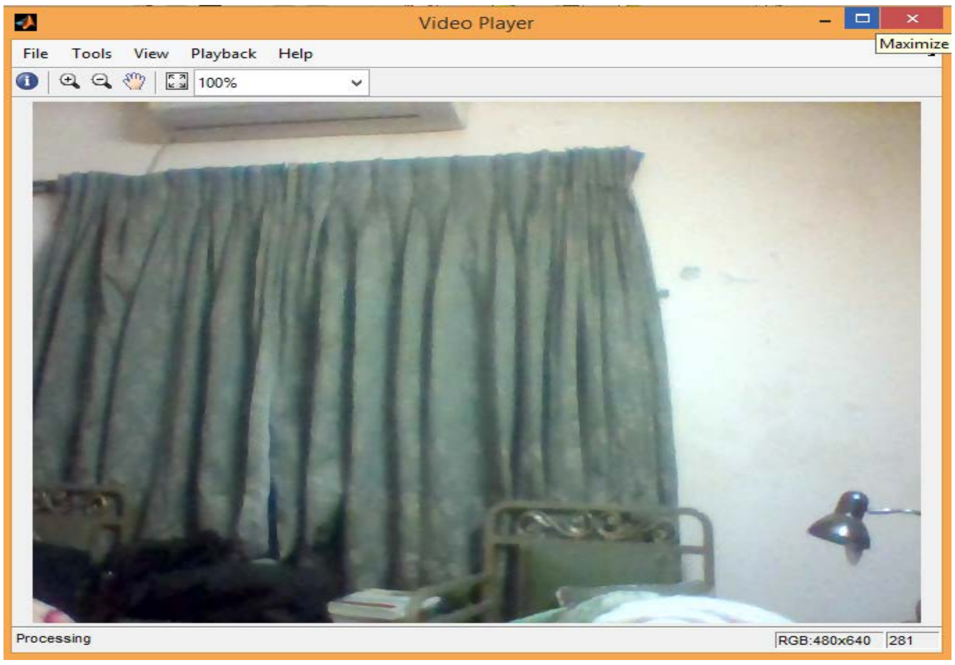


Figure 24: Human Detection and Tracking on the basis of Upper Body

b. We used the video player of Computer Vision which is a feature of Matlab to view the real time images/video after the image processing has been done on the frames from the live video feed but when we had to dovetail the same with the GUI while developing it we had to change the complete approach and code because Vision .video player creates its own independent GUI and puts frame in it by default and the frames can’t be accessed by another



GUI. Hence we had to manually show the frames on the axes of our developed GUI.

Figure 25: Computer Vision Video Player

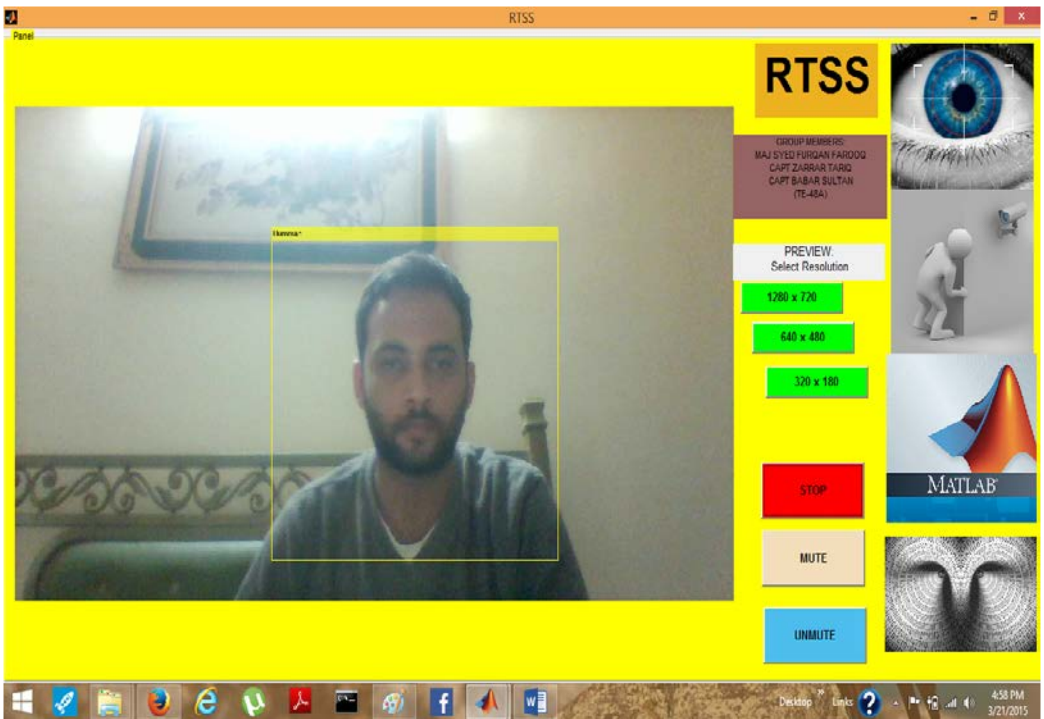


Figure 26: GUI

c. To track the human over time we also tried using Kanade-Lucas-Tomasi (KLT) algorithm. The source code of KLT is in the public domain, available for both commercial and non-commercial use. The aim of KLT algorithm is to track a set of feature points across the video frames. Once the human is detected, the next step is the identification of feature points that can be reliably tracked. After that, Vision Point Tracker System object is used to track these points. The point tracker attempts to find a point in the current frame corresponding to every point in the previous frame. Then the rotation, translation and scale between the new points and the old points is estimated. This transmutation is applied to the bounding box around the face. KLT uses detect Min Eigen Features function to get the initial points for tracking .The inculcation of KLT algorithm introduced delay in the overall working of RTSS and false alarm rate also increased hence we developed our own efficient tracking mechanism. As each frame is being checked for human detection and bounding box is applied on detected upper body hence we are inserting object annotations and displaying the same frame with the bounding box applied on each which serves the purpose of tracking without calling any other function hence reducing the computational time and achieving higher accuracy.

d. The camera placement is also an important and key factor for the successful human detections as viola jones algorithm is sensitive to plane changes hence greatly effecting the results of RTSS. Camera must be placed such that frontal portion of human upper body must be completely and clearly captured in the frames so that analysis can be done leading to successful detection. This is a constraint as present which can be overcome by intelligent siting of the camera generally on the entrances or gates.

e. Some statistics are as under:-

METHOD	DETETCTION %	FALSE POSITIVE %	PROCESSING TIME ms
Viola Jones	71.2	0.012	30

Chapter 5

5. Recommendations for Future Work

In the development of RTSS we have been able to successfully detect humans only and the results are quite good but there is always a room for improvement and the technology must evolve rapidly in order to satisfy the mankind’s need. Following are certain recommendations for future development:-

a. Viola Jones algorithm is very sensitive to plane changes which means that the camera must be placed in such a way that it covers the frontal view of the human. Future work must be focused towards incorporating such techniques and changes by which detection becomes more efficient and resistant to plane changes.

b. At present the RTSS is implemented using a laptop because we have done all the image processing on MATLAB .Efforts must be made to make the RTSS a standalone system doesn’t requiring a laptopyet keeping MATLAB as the backend image processing platform.

c. SMS alert is a key feature of the RTSS but currently there is a delay in the reception of the SMS due to the GSM network limitations which must be reduced in future.

d. 3G and 4G have been launched in Pakistan but right now its coverage area is only limited to some parts of major cities and also the speed is not feasible for real time video streaming. In future when the coverage area and data rate increases then the feature of live video feed can be incorporated in RTSS.

e. We have been able to successfully run the RTSS on 1280 x 720 resolution which might be further increased without making it slow and laggy.

6. Conclusion

The purpose of developing RTSS was to make an efficient surveillance system which only triggers alarm on the basis of human detection in the target area. An effort has been made to make the detection efficient in terms of computational complexity by inculcating wavelet based technology. GSM module has been incorporated in the RTSS to alert a distant user regarding human intrusion in the target area. Following are some points:-

a. Achievements

- High video resolution 1280 x 720
- Efficient image processing hence minimizing computational time
- Alarm generation
- Efficient video storage
- SMS alert
- GUI development

b. Limitations

- Camera placement
- GSM network delay in receiving SMS alert
- Requirement of laptop and MATLAB

c. Applications

- Home security
- Vehicle protection
- Border control
- Bank security

7. References

1) Picardi, M. (2004). Background subtraction techniques: a review. IEEE international conference, vol.31, no.4 ,92-100
2) D.keller, J.W (1994).Towards Robust Automatic Traffic Scene Analysis in Real-time. Proc.ICPR'94, 126-131
3) "A Novel comprehensive method for real time Video Motion Detection Surveillance" by Sumita Mishra, Prabhat Mishra, Naresh K Chaudhary, Pallavi Asthana.
4) International Journal of Computational Engineering Research||Vol, 03||Issue, 4|||April||2013||Page 65 Design and development of Optical flow based Moving Object Detection and Tracking (OMODT) System Ms.ShamshadShirgeri, MsPallaviUmeshNaik,Dr.G.R.Udupi, Prof.G.A.Bidkar

5) Active motion detection and object detection by Denzler,J and Paulus,D.W.R in ImageProcessing,1994.Proceedings.ICIP94.,IEEEInternationalConference (Volume:3)1994.(http://ieeexplore.ieee.org/xpl/abstractAuthors.jsp?arnumber=413812)
6) Motion Tracking Project, Fabian Wauthier, University of California, Berkeley Email: w@berkeley.edu.(http://www.stats.ox.ac.uk/~wauthier/tracker/tracker-synopsis.pdf)
7) Object tracking and detection. (http://www.mathworks.com/help/vision/gs/object-detection-and-tracking.html)
8) Motion & Object Detection from Contiguous Video Frames .9th All Pakistan Inter Colleges/ Universities Computer Projects Exhibition & Competition COMPPEC-2010 by Rizwan ahmed ,Javed Akbar ,Jafar Shahbaz ,Asim Naeem
9) L”utkebohle, I. (n.d.). Capabilities and Limitations of Visual Surveillance
10) www.researchgate.net/profile/Aju_John/publication/249643445_Object_Detection_using_Wavelet_Transform_Method_for_Three_way_Decomposed_Images/links/0046351e57851e0b9c000000.
11) www.researchgate.net/publication/260364338_A_joint_of_optical_flow_and_principle_component_analysis_approach_for_motion_detection_from_outdoor_videos
12) Lucas, Bruce D. and Takeo Kanade. "An Iterative Image Registration Technique with an Application to Stereo Vision,"*Proceedings of the 7th International Joint Conference on Artificial Intelligence*, April, 1981, pp. 674–679
13) Tomasi, Carlo and Takeo Kanade. *Detection and Tracking of Point Features*, Computer Science Department, Carnegie Mellon University, April, 1991
14) Shi, Jianbo and Carlo Tomasi. "Good Features to Track," *IEEE Conference on Computer Vision and Pattern Recognition*, 1994, pp. 593–600
15) Krerngkamjornkit, R and Simic, M 2013, 'Enhancement of human body detection and tracking algorithm based on Viola and Jones framework', in B. D. Milovanovic (ed.) Proceedings of 2013 11th International Conference on Telecommunications in Modern Satellite, Cable and Broadcasting Services, Serbia (TELSIKS), Serbia, 16 - 19 October2013, pp. 115-118
16) Optimizing Viola-Jones Face Detection for use in webcams by Theo Ephriam and Tristan Himmelman
17) Nao detection with a cascade of boosted weak classifiers based on Haar-like features by Duncan S.tenVelthuis ,Bachelor Thesis 2014, University of Amsterdam

APPENDIX A

PORJECT PROPOSAL

Extended Title: Real Time Surveillance With SMS Alert (GSM MODULE)

Brief Description of The Project / Thesis with Salient Specs: The project mainly focuses on the real time detection of motion (mainly human) in the required area. Its main purpose is surveillance that does not require continuous human monitoring as it triggers an alarm and sends a sms via GSM module to info a distant user regarding motion in the area of interest. Wavelet based technology will be employed to reduce computational complexity of the algorithm in real time application.

Scope of Work: Field will be monitored using a camera and captured frames will be compared continuously to detect any motion. There are many motion detection algorithms available such as Cross Correlation, Joint difference, Background Subtraction etc. Moving object will be continuously tracked and an alarm will be activated .With the help of a GSM module a sms will be sent to a pre stored number.

Academic Objectives: The project encompasses a great deal of knowledge both at hardware and software end. Image processing will be used extensively. Dovetailing the GSM module will be a major challenge.

Application / End Goal Objectives : The hardware/software can be easily modified to implement intelligent solutions for many different problems :-

- Home/workplace surveillance
- Vehicle protection
- Strategic facility examination
- Trespassing scrutiny
- Border security
- Pet/child watcher
- Traffic monitoring

- Violent behavior detection on streets

Previous Work Done on The Subject:

- Active motion detection and object detection by Denzler,J and Paulus,D.W.R in Image Processing, 1994. Proceedings. ICIP-94., IEEE International Conference (Volume:)1994.http://ieeexplore.ieee.org/xpl/abstractAuthors.jsp?arnumber=413812)
- Motion Tracking Project, Fabian Wauthier, University of California, Berkeley Email: w@berkeley.edu. (http://www.stats.ox.ac.uk/~wauthier/tracker/tracker-synopsis.pdf)
- Object tracking and detection. (http://www.mathworks.com/help/vision/gs/object-detection-and-tracking.html)
- Motion & Object Detection from Contiguous Video Frames .9th All Pakistan Inter Colleges/ Universities Computer Projects Exhibition & Competition COMPPEC-2010 by Rizwan ahmed ,Javed Akbar ,Jafar Shahbaz ,Asim Naeem (http://www.comppec.com)

Material Resources Required : Following resources will be required:-

- Camera
- Laptop
- Software's
- GSM module
- Alarm

No of Students Required : 3 x Student officers

Special Skills Required : Motion detection, image processing, GSM module knowledge Arduino Board Integration

APPENDIX B

CODES

1. MATLAB CODE

```
functionvarargout = RTSS(varargin)
%RTSS M-file for RTSS.fig
%      RTSS, by itself, creates a new RTSS or raises the existing
%      singleto n*.
%
%      H = RTSS returns the handle to a new RTSS or the handle to
%      the existing singleton*.
%
%      RTSS('Property','Value',...) creates a new RTSS using the
%      given property value pairs. Unrecognized properties are passed via
%      varargin to RTSS_OpeningFcn.  This calling syntax produces a
%      warning when there is an existing singleton*.
%
%      RTSS('CALLBACK') and RTSS('CALLBACK',hObject,...) call the
%      local function named CALLBACK in RTSS.M with the given input
%      arguments.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help RTSS

% Last Modified by GUIDE v2.5 27-Apr-2015 20:13:16

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @RTSS_OpeningFcn, ...
                   'gui_OutputFcn',  @RTSS_OutputFcn, ...
                   'gui_LayoutFcn',  [], ...
                   'gui_Callback',   []);
ifnargin&&ischar(varargin{1})
gui_State.gui_Callback = str2func(varargin{1});
end
```

```
ifnargout
   [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
gui_mainfcn(gui_State, varargin{:});
end

% End initialization code - DO NOT EDIT


% --- Executes just before RTSS is made visible.
functionRTSS_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   unrecognized PropertyName/PropertyValue pairs from the
%            command line (see VARARGIN)
% Choose default command line output for RTSS
handles.output = hObject;
guidata(hObject, handles);
a = imread('1.jpg');
axes(handles.axes2)
imshow(a);
b = imread('2.jpg');
axes(handles.axes3)
imshow(b);
c = imread('3.jpg');
axes(handles.axes4)
imshow(c);
d = imread('4.png');
axes(handles.axes5)
imshow(d);



% --- Outputs from this function are returned to the command line.
functionvarargout = RTSS_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
HumanDetector = vision.CascadeObjectDetector('UpperBody');
HumanDetector.MinSize = [150 150];
HumanDetector.ScaleFactor = 1.3;
HumanDetector.MergeThreshold = 5;
[data,fs]=audioread('alarm1.wav');
player=audioplayer(data,fs);
writerObj = VideoWriter('playback.avi');
writerObj.FrameRate=5;
open(writerObj);
handles.getVideo = imaq.VideoDevice('winvideo',1,'RGB24_1280X720');
I2 = step(handles.getVideo);
global true;
global mute;
mute=0;
true=1;
globalSerPIC;
SerPIC = serial('COM4');
set(SerPIC,'BaudRate',9600);
set(SerPIC,'DataBits',8);
set(SerPIC,'Parity','none');
set(SerPIC,'StopBits',1);
set(SerPIC,'FlowControl','none');
fopen(SerPIC);
flag = 0;
flag1=0;
while true==1;
    I2 = step(handles.getVideo);
bboxBody = step(HumanDetector, I2);
IBody = insertObjectAnnotation(I2, 'rectangle',bboxBody,'Human');
axes(handles.axes1);
imshow(IBody);
    flag1=flag1+1;
if (~isempty(bboxBody))&&(mute==0)
play(player);
if (mod(flag,200)==0);
fprintf(SerPIC,'1');
flag=flag+1;
disp(flag);
else
flag=flag+1;
end
if  flag==590;
flag=0;
end
end
if ~isempty(bboxBody)
writeVideo(writerObj,IBody);
end
    flag1=flag1+1;
```

```
disp(flag1);
if flag1==590;
flag=0;
        flag1=0;
end
end
fclose(SerPIC);
delete(SerPIC)
clearSerPIC
close(writerObj);
clear all

function pushbutton1_CreateFcn(hObject, ~, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% --- Executes on button press in pushbutton8.
function pushbutton8_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
HumanDetector = vision.CascadeObjectDetector('UpperBody');
HumanDetector.MinSize = [50 50];
HumanDetector.ScaleFactor = 1.2;
HumanDetector.MergeThreshold = 10;
[data,fs]=audioread('alarm1.wav');
player=audioplayer(data,fs);
writerObj = VideoWriter('playback.avi');
writerObj.FrameRate=5;
open(writerObj);
handles.getVideo = imaq.VideoDevice('winvideo', 1, 'RGB24_640x480');
I2 = step(handles.getVideo);
global true;
global mute;
mute=0;
true=1;
globalSerPIC;
SerPIC = serial('COM4');
set(SerPIC,'BaudRate',9600);
set(SerPIC,'DataBits',8);
set(SerPIC,'Parity','none');
set(SerPIC,'StopBits',1);
set(SerPIC,'FlowControl','none');
fopen(SerPIC);
flag = 0;
flag1=0;
while true==1;
    I2 = step(handles.getVideo);
bboxBody = step(HumanDetector, I2);
```

```
IBody = insertObjectAnnotation(I2, 'rectangle',bboxBody,'Human');
handles.IBody=IBody;
axes(handles.axes1);
imshow(IBody);
if (~isempty(bboxBody))&&(mute==0)
play(player);
if (mod(flag,200)==0);
fprintf(SerPIC,'1');
flag=flag+1
else
flag=flag+1;
end
if  flag==590
flag=0
end
end
if ~isempty(bboxBody)
writeVideo(writerObj,IBody);
end
   flag1=flag1+1;
disp(flag1);
if flag1==590;
flag=0;
       flag1=0;
end
end
fclose(SerPIC);
delete(SerPIC)
clearSerPIC
close(writerObj);
clear all

% --- Executes on button press in pushbutton9.
function pushbutton9_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
HumanDetector = vision.CascadeObjectDetector('UpperBody');
HumanDetector.MinSize = [100 100];
HumanDetector.MergeThreshold = 10;
[data,fs]=audioread('alarm1.wav');
player=audioplayer(data,fs);
writerObj = VideoWriter('playback.avi');
writerObj.FrameRate=5;
open(writerObj);
handles.getVideo = imaq.VideoDevice('winvideo',1,'RGB24_320X180');
I2 = step(handles.getVideo);
global true;
global mute;
mute=0;
```

```
true=1;
SerPIC = serial('COM4');
set(SerPIC,'BaudRate',9600);
set(SerPIC,'DataBits',8);
set(SerPIC,'Parity','none');
set(SerPIC,'StopBits',1);
set(SerPIC,'FlowControl','none');
fopen(SerPIC);
flag = 0;
flag1=0;
while true==1;
    I2 = step(handles.getVideo);
bboxBody = step(HumanDetector, I2);
IBody = insertObjectAnnotation(I2, 'rectangle',bboxBody,'Human');
handles.IBody=IBody;
axes(handles.axes1);
imshow(IBody);
if (~isempty(bboxBody))&&(mute==0)
play(player);
if (mod(flag,200)==0);
fprintf(SerPIC,'1');
flag=flag+1;
else
flag=flag+1;
end
if  flag==590;
flag=0;
end
end
if ~isempty(bboxBody)
writeVideo(writerObj,IBody);
end
    flag1=flag1+1;
disp(flag1);
if flag1==590;
flag=0;
        flag1=0;
end
end
fclose(SerPIC);
delete(SerPIC)
clearSerPIC
close(writerObj);
clear all

% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
global true;
true=0;



% --- Executes on button press in pushbutton6.
function pushbutton6_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global mute;
mute=1;


% --- Executes on button press in pushbutton7.
function pushbutton7_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global mute;
mute=0;
```

2. ARDUINO CODE

```
#include "SIM900.h"
#include <SoftwareSerial.h>
#include "sms.h"
SMSGSM sms;
int value;
intnumdata;
boolean started=false;
charsmsbuffer[160];
char n[20];
charsms_position;
charphone_number[20];
charsms_text[100];
inti;
void setup()
{    Serial.begin(9600);
if (gsm.begin(9600))
   {       Serial.println("\nstatus=READY");
```

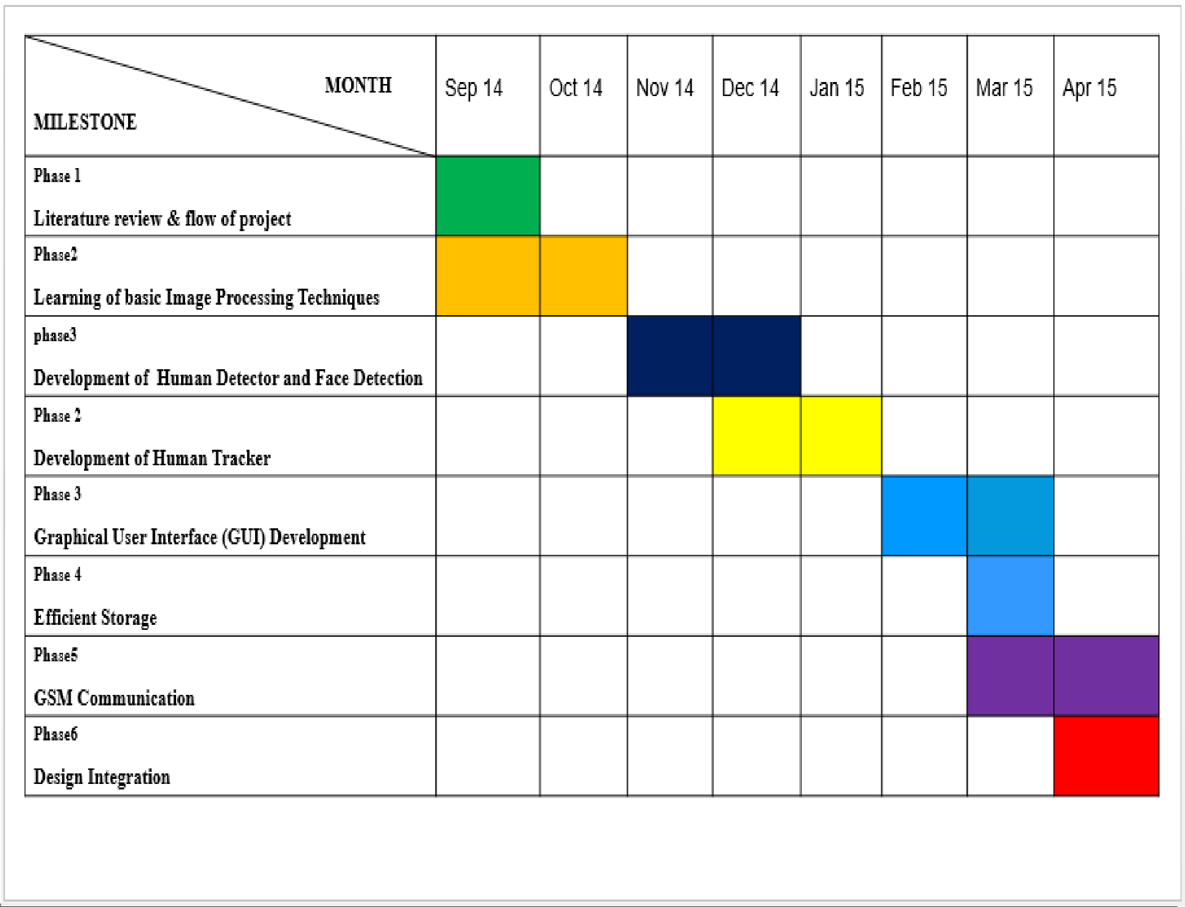
```
                    started=true;
   }
else
Serial.println("\nstatus=IDLE");

if(started)
   {
if (sms.SendSMS("+923335522017", "***RTSS STARTUP SMS***"))
      {
Serial.println("\nSMS sent OK.");
      }
else
      {
Serial.println("\nError sending SMS.");
      }
   }
};

void loop()
{
if (Serial.available())       // Read while there is data in the buffer
  {
value = Serial.read();
Serial.println(value);          // Debug - double check it make sense?
if (value==49)
{ if(started)
   {
if (sms.SendSMS("+923335522017", "***HUMAN INTRUSION***"))
      {
Serial.println("\nSMS sent OK.");
      }
else
```

```
        {
Serial.println("\nError sending SMS.");
        }
    }
 }
if (value==50)
{ if(started)
    {
if (sms.SendSMS("+923335522017", "NOT DONE"))
        {
Serial.println("\nSMS sent OK.");
        }
else
        {
Serial.println("\nError sending SMS.");
        }
    }
 }

if (value==51)
{ if(started)
    {
if (sms.SendSMS("+923335522017", "WARNING"))
        {
Serial.println("\nSMS sent OK.");
        }
else
        {
Serial.println("\nError sending SMS.");
        }
    }
 }
```

```
if (value==52)
{ if(started)
    {
if (sms.SendSMS("+923335522017", "ERROR"))
        {
Serial.println("\nSMS sent OK.");
        }
else
        {
Serial.println("\nError sending SMS.");
        }
    }
 }


  }
if(started)
    {
sms_position=sms.IsSMSPresent(SMS_UNREAD);
if (sms_position)
        {
Serial.print("SMS postion:");
Serial.println(sms_position,DEC);
sms.GetSMS(sms_position, phone_number, sms_text, 100);
Serial.println(phone_number);
Serial.println(sms_text);
        }
delay(2000);
    }
}
```

APPENDIX C

TIMELINE



MONTH / MILESTONE	Sep 14	Oct 14	Nov 14	Dec 14	Jan 15	Feb 15	Mar 15	Apr 15
Phase 1 Literature review & flow of project	🟩							
Phase2 Learning of basic Image Processing Techniques	🟧	🟧						
phase3 Development of Human Detector and Face Detection			🟦	🟦				
Phase 2 Development of Human Tracker				🟨	🟨			
Phase 3 Graphical User Interface (GUI) Development						🟦	🟦	
Phase 4 Efficient Storage							🟦	
Phase5 GSM Communication							🟪	🟪
Phase6 Design Integration								🟥

APPENDIX D

COST BREAK DOWN

	ITEM	**COST** (Rs)
1.	ARDUINO UNO	1000
2.	GSM MODULE	6000
3.	SPEAKERS	2000
4.	MISCELLANEOUS	1000
	TOTAL	10000