CHAPTER-1
INTRODUCTION

1.1 GENERAL

In these days, security is the major issue for all over the world. Security is very important in order to protect vulnerable and valuables assets such as a person, dwelling, community and nation from any harm. International security issue is also very important especially border and coast security to any country. The people of national security agencies, maritime security organization, military forces and other forces sacrifice their lives to protect their country people. The lives of forces are also very precious like another lives. So, by using advance technologies, the forces can protect their nation superiorly with minimum life losses. In the modern era, computer-based security equipment is very popular among forces because they are more advance and safer for themselves.

In our India, war was happened many times like Indo-Pakistan war in 1947, 1965, 1971 and 1999, Sino Indian war in 1962 and some terrorists' incidents happened in India are listed below.

Table 1.1 History of terrorist attack incidents

Incidents	Date	Incident & Description	Location	Persons Dead	Injured	Status of case
1	June 8, 1980	Mandai massacre	Tripura	400		N/A
2	August 2, 1984	Meenambakkam bomb blast	Tamil Nadu	30	25	Verdict given
3	July 7, 1987	1987 Haryana killings	Haryana	36	60	N/A

Incidents	Date	Incident & Description	Location	Persons Dead	Injured	Status of case
4	June 15, 1991	1991 Punjab killings	Punjab	126	200	N/A
5	October 17, 1991	1991 Rudrapur bombings	Uttarakhand	41	140	
6	March 12, 1993	1993 Bombay bombings	Mumbai	257	700+	verdict given
7	December 30, 1996	Brahmaputra Mail train bombing	Assam	33	150	N/A
8	February 14, 1998	1998 Coimbatore bombings	Tamil Nadu	58	200+	Verdict given
9	October 1, 2001	2001 Jammu and Kashmir legislative assembly car bombing	Jammu and Kashmir	38		
10	September 10, 2002	Rafiganj train wreck	Bihar	200	150+	
11	September 24, 2002	Attack on Akshardham temple	Gujarat	31	80	
12	August 25, 2003	25 August 2003 Mumbai bombings	Mumbai	52		
13	October 29, 2005	2005 Delhi bombings: Three powerful serial blasts in New	Delhi	70	250	

Incidents	Date	Incident & Description	Location	Persons Dead	Injured	Status of case
		Delhi at different places				
14	July 11, 2006	2006 Mumbai train bombings: Series of 7 train bombing during the evening rush hour in Mumbai	Mumbai	209	500	
15	September 8, 2006	2006 Malegaon bombings: Series of bomb blasts in the vicinity of a mosque in Malegaon, Maharashtra	Maharashtra	37	125	
16	February 18, 2007	2007 Samjhauta Express bombings	Haryana	68	50	
17	August 25, 2007	August 2007 Hyderabad bombings - Two blasts in Hyderabad's Lumbini park and Gokul Chat.	Hyderabad	42	54	
18	May 13, 2008	Jaipur bombings: 9 bomb blasts along 6 areas in Jaipur	Jaipur	63	200	
19	July 26, 2008	2008 Ahmedabad bombings: 17 serial bomb blasts in Ahmedabad	Gujarat	29	110	arrests made

Incidents	Date	Incident & Description	Location	Persons Dead	Injured	Status of case
20	October 30, 2008	2008 Assam bombings	Assam	81	470	
21	November 26, 2008	2008 Mumbai attacks	Mumbai	171	239	Verdict given
22	April 6, 2010	April 2010 Maoist attack in Dantewada	Chhattisgarh	42 (including 8 terrorists)	8	
23	28 May 2010	Jnaneswari Express train derailment	West Bengal	74	100+	
24	4-9 June 2015	2015 Manipur ambush	Manipur	176 (including 158 terrorists)	15	Surgical strike by Indian Armed Forces near India Myanmar border killing 158 terrorists.
25	February 14, 2019	2019 Pulwama attack	Awantipora, Jammu & Kashmir	46	250-300	Air strikes by iaf on pakistani militant group jaish-I-muhammad, which took the responsibility of the attack

The dead and injured peoples are more in the above listed Table 1.1 in order to overcome it there is a need of hyper security, which must overcome all the errors in the previous security systems.

For example, the drone technology which is controlled automatically by the computer. In this technology, target is selected and hit by using computer-based algorithms including the image processing techniques. The availability of high quality and inexpensive video cameras and the increasing need of automated video analysis has generated a great deal of interest in the area of motion detection, object tracking and object targeting.

But here this project is going to develop for protecting the nation from the foreign terrorist entry through border by automatically aims and fires the targets which are detected by using image processing algorithms called face detection and tracking. At first the face is detected, and then the gun starts the tracking process with the help of servos according to the movement of human face. Once it aims the target then the gun triggered automatically to head shoot.

1.2 OBJECTIVE

To avoid the soldiers suffering from over hotness and coldness during the winter and summer season. While guarding there is a need to develop autonomous guarding system. Thus, it reduces the monitoring work of the soldier from morning to night. It shoots an enemy if he crosses the border when soldier is not there. It reduces the death of border guard soldiers. It guards the border 24×7. It is used to manage the sudden terrorist ingress. It can avoid the periodic alteration of soldiers from borders. This prototype is used to trigger the gun automatically when it detects the enemy.

1.3 PROBLEM IDENTIFICATION

The periodic death of soldiers is one of the important reasons and the soldiers are guarding in unsafe environment and they are monitoring the border 24×7 without sleep. During any natural calamities like snow slides or landslides they may have chance to stuck in it. The health of the soldiers sometime being abnormal due to continuous work they may fell in sick. If any sudden terrorist attack happen some soldiers may injure heavily.

1.4 TYPES OF GUNS USED BY INDIAN ARMY

Table 1.2 Guns used by our Indian army

S. No	Name	Type	Caliber
1	Pistol Auto 9mm 1A	Semi-automatic pistol	9×19mm Parabellum
2	SIG Sauer P226	Semi-automatic pistol	9×19mm Parabellum
3	Glock 17	Semi-automatic pistol	9×19mm Parabellum
4	FN Five-seven	Semi-automatic pistol	FN 5.7×28mm
5	Franchi SPAS-15	Combat Shotgun	18.5 x 76 mm NATO
6	SAF Carbine 2A1	Sub-machine gun	9×19mm Parabellum
7	Micro-Uzi	Sub-machine gun	9×19mm Parabellum
8	Heckler & Koch MP5	Submachine gun	9×19mm Parabellum

9	Brügger & Thomet MP9	Machine Pistol	9×19mm Parabellum
10	FN P90	Personal defense weapon	5.7x28mm
11	Steyr AUG	Sub-machine gun	9×19mm Parabellum
12	FN SCAR	Assault rifle	5.56x45mm NATO and 7.62x51mm NATO
13	IMI Tavor TAR-21	Assault rifle	5.56×45mm NATO
14	FN F2000	Assault rifle	5.56x45mm NATO
15	SIG SG 550	Assault rifle	5.56x45mm NATO
16	AKM	Assault rifle	7.62×39mm
17	Vz. 58	Assault rifle	7.62x39mm
18	AK-103	Assault rifle	7.62×39mm
20	AK-203	Assault Rifle	7.62x39

The above Table 1.2 shows that some of the pistols and rifles used by our Indian army which can be fixed in the gun holder of the prototype. The gun holder is designed to hold all types of guns. Each gun is differing by their firing rate, ammos and cartridge size.

CHAPTER – 2
LITERATURE REVIEW

Yang and Huang (1994) have identified that the human face is a complex pattern. Finding human faces automatically in a scene is a difficult yet significant problem. It is the first important step in a fully automatic human face recognition system. In this paper a new method to locate human faces in a complex background is proposed. This system utilizes a hierarchical knowledge-based method and consists of three levels. The higher two levels are based on mosaic images at different resolutions. In the lower level, an improved edge detection method is proposed. In this research the problem of scale is dealt with, so that the system can locate unknown human faces spanning a wide range of sizes in a complex black-and-white picture.

Hjelmas and Low (2001) have presented a comprehensive and critical survey of face detection algorithms. Face detection is a necessary first-step in face recognition systems, with the purpose of localizing and extracting the face region from the background. It also has several applications in areas such as content-based image retrieval, video coding, video conferencing, crowd surveillance, and intelligent human–computer interfaces. However, it was not until recently that the face detection problem received considerable attention among researchers. The human face is a dynamic object and has a high degree of variability in its appearance, which makes face detection a difficult problem in computer vision. A wide variety of techniques have been proposed, ranging from simple edge-based algorithms to composite high-level approaches utilizing advanced pattern recognition methods. The algorithms presented in this paper are classified as either feature-based or image-based and are discussed in terms of their technical approach and performance.

Due to the lack of standardized tests, they do not provide a comprehensive comparative evaluation, but in cases where results are reported on common datasets, comparisons are presented. They also give a presentation of some proposed applications and possible application areas.

Lin *et al* (2005) have proposed that due to enhancement of computer performance and popular usage of webcam devices, it has become possible to acquire users' gestures for the human-computer-interface with PC via webcam. However, the effects of illumination variation would dramatically decrease the stability and accuracy of skin-based face tracking system; especially for a notebook or portable platform. In this study we present an effective illumination recognition technique, combining K-Nearest Neighbor classifier and adaptive skin model, to realize the real-time tracking system. We have demonstrated that the accuracy of face detection based on the KNN classifier is higher than 92% in various illumination environments. In real-time implementation, the system successfully tracks user face and eyes features at 15 fps under standard notebook platforms. Although KNN classifier only initiates five environments at preliminary stage, the system permits users to define and add their favorite environments to KNN for computer access. Eventually, based on this efficient tracking algorithm, we have developed a "Webcam Mouse" system to control the PC cursor using face and eye tracking. Preliminary studies in "point and click" style PC web games also shows promising applications in consumer electronic markets in the future.

Mian (2008) have described that surveillance cameras generally have a wide field of view and do not capture the human face with a resolution that is sufficient for machine recognition. To overcome this problem, this paper presents a real time face detection and tracking algorithm using a single PTZ (Pan, Tilt, Zoom) camera. Unlike existing algorithms which track the human face in the field of view of a fixed

camera, the proposed algorithm adjusts the pan, tilt and zoom of a PTZ camera in real-time so that human faces can be acquired at the camera's maximum resolution. The proposed algorithm addresses challenges like network delays, camera movement lags, oscillations and lost faces. Experiments were performed in realistic surveillance scenarios and accurate real-time tracking with pan, tilt and zoom was achieved.

Li *et al* (2011) have presented a novel boosting cascade-based face detection framework using SURF features. The framework is derived from the well-known Viola-Jones (VJ) framework but distinguished by two key contributions. First, the proposed framework deals with only several hundreds of multidimensional local SURF patches instead of hundreds of thousands of single dimensional haar features in the VJ framework. Second, it takes AUC as a single criterion for the convergence test of each cascade stage rather than the two conflicting criteria (false-positive-rate and detection rate) in the VJ framework. These modifications yield much faster training convergence and much fewer stages in the final cascade. We made experiments on training face detector from large scale database. Results shows that the proposed method is able to train face detectors within one hour through scanning billions of negative samples on current personal computers. Furthermore, the built detector is comparable to the state-of-the-art algorithm not only on the accuracy but also on the processing speed.

Sethi and Aggarwal (2011) have presented a face detection and tracking algorithm in real time camera input environment. The entire face tracking algorithm is divided into two modules. The first module is face detection and second is face tracking. To detect the face in the image, Haar based algorithm is used. On the face image, Shi and Thomasi algorithm is used to extract feature points and Pyramidal Lucas-Kanade algorithm is used to track those detected features. Results on the real

time indicate that the proposed algorithm can accurately extract facial features points. The algorithm is applied on the real time camera input and under real time environmental conditions.

Faux and Luthon (2012) have predicted a method of face detection and tracking by computer vision for multimedia applications. Contrary to current techniques that are based on huge learning databases and complex algorithms to get generic face models (*e.g.* active appearance models), the proposed method handles simple contextual knowledge representative of the application background thanks to a quick supervised initialization. The algorithm contains two main steps: detection and tracking. In the detection phase, an evidential face model is estimated by merging basic beliefs elaborated from Viola and Jones face detector and from a skin color detector, for the assignment of mass functions. These functions are computed as the merging of sources in a specific nonlinear color space. In order to deal with color information dependence in the fusion process, the Denoeux cautious rule is used. They are the entries of a bootstrap particle filter which yields face tracking at video rate. We show that the proper tuning of the evidential model parameters improves the tracking performance in real-time.

Belaroussi and Milgram (2012) have described that face localization is the first stage in many vision-based applications and in human–computer interaction. The problem is to define the face location of a person in a color image. The four boosted classifiers embedded in OpenCV, based on Haar-like features, are compared in terms of speed and efficiency. Skin color distribution is estimated using a non-parametric approach. To avoid drifting in color estimate, this model is not updated during the sequence but renewed whenever the face is detected again, that gives the ability to our system to cope with different lighting conditions in a more robust way. Skin color model is then used to localize the face represented by an ellipse:

connected component segmentation and a statistical approach, namely the coupled Camshift of Bradsky, are compared in terms of efficiency and speed. The pursuit algorithms are tested on various video sequences, corresponding to various scenarios in terms of illumination, face pose, face size and background complexity (distractor effects).

Yan *et al* (2013) had found that the state-of-the-art face detectors still have problems in dealing with images in the wild due to large appearance variations. Instead of leaving appearance variations directly to statistical learning algorithms, we propose a hierarchical part based structural model to explicitly capture them. The model enables part subtype option to handle local appearance variations such as closed and open month, and part deformation to capture the global appearance variations such as pose and expression. In detection, candidate window is fitted to the structural model to infer the part location and part subtype, and detection score is then computed based on the fitted configuration. In this way, the influence of appearance variation is reduced. Besides the face model, we exploit the co-occurrence between face and body, which helps to handle large variations, such as heavy occlusions, to further boost the face detection performance. We present a phrase-based representation for body detection, and propose a structural context model to jointly encode the outputs of face detector and body detector. Benefit from the rich structural face and body information, as well as the discriminative structural learning algorithm, our method achieves state-of-the-art performance on FDDB, AFW and a self-annotated dataset, under wide comparisons with commercial and academic methods.

Tripathy and Daschoudhury (2013) have implemented a real time Face detection and tracking the head poses position from high definition video using Haar Classifier through Raspberry Pi BCM2835 CPU processor which is a combination

of SoC with GPU based Architecture. OV5647 CMOS Image sensor with 5-megapixel used for obtaining high definition video H.264 video data via GPU's hardware video decoder to improve the playback of H.264 Video data supporting from 1080p at 30fps with complete user control over formatting and output data transfer also supporting with 720p/60HD video in full field of View (FOV). SimpleCV and OpenCV libraries are used for face detection and tracking the head poses position. The experimental result computed by using computer vision SimpleCV and OpenCV framework libraries along with above mentioned hardware results were obtained through of 30 fps under 1080p resolutions for higher accuracy and speediness for face detection and tracking the head poses position.

Mamata (2014) have represented a methodology for face detection robustly in real time environment. Here we use Harr like classifier and adaboost algorithm to track faces on OpenCV platform which is open source and developed by Intel. Face detection which is the task of localizing faces in an input image is a fundamental part of any face processing system. The aim of this paper is to present a review on various methods and algorithms used for face detection etc. Three different algorithms i.e. Haar cascade, adaboost, template matching was described Finally it includes some of applications of face detection.

Bhattacharjee *et al* (2015) have detected that the robotics domain has a couple of specific general design requirements which requires the close integration of planning, sensing, control and modelling and for sure the robot must take into account the interactions between itself, its task and its environment surrounding it. While earlier works have focused primarily on issues such as manipulation and navigation only, this proposal presents a conceptual and intuitive approach towards man-machine interaction in order to provide a secured live biometric logical authorization to the user access, while making an intelligent interaction with the

control station to navigate advanced gesture controlled wireless Robotic prototypes or mobile surveillance systems along desired directions through required displacements. The intuitions are based on tracking real-time 3-Dimensional Face Motions using skin tone segmentation and maximum area considerations of segmented face-like blobs, or directing the system with voice commands using real-time speech recognition. The system implementation requires designing a user interface to communicate between the Control station and prototypes wirelessly, either by accessing the internet over an encrypted Wi-Fi Protected Access (WPA) via a HTML web page for communicating with face motions or with the help of natural voice commands like “Trace 5 squares”, “Trace 10 triangles”, “Move 10 meters”, etc. evaluated on an iRobot Create over Bluetooth connectivity using a Bluetooth Access Module (BAM).

Kumbhar *et al* (2017) have proposed a real-time Face detection and tracking the head poses position from high definition video using Haar Classifier through Raspberry Pi BCM2835 CPU processor which is a combination of SoC with GPU based Architecture. SimpleCV and OpenCV libraries are used for face detection and tracking the head poses position. The experimental result computed by using computer vision SimpleCV and OpenCV framework libraries along with above mentioned hardware results were obtained through of 30 fps under 1080p resolutions for higher accuracy and speediness for face detection and tracking the head poses position.

Renuka *et al* (2017) have said that dynamic identification of biometric system in real world is difficult and it is an important task. With the advent of computers, there are growing interest on identifying context generated from human known as human biometric identification. The main aim of this project is to design and build a manually controlled biometric system. The main purpose of this system is that a

system should be able to roam around in a given environment. The work is to identify the realistic human face and eyes through camera. Our method consists of three main components they are HC-SR501(PIR) sensor which senses and detects the object and by transferring the data serially from Arduino to 12C Interface which is connected to camera for classifying the activities and recognizing the complex action classes. The captured video can be enhanced and made intelligible using further image processing on the remote PC thereby eliminating the need for extra hardware on the system.

Zeng *et al* (2018) have did research on numerous recent face detectors based on convolutional neural networks (CNNs) have significantly improved the detection performance. However, CNNs usually have a huge number of parameters which lead to very low detection speeds. To address this issue, this paper proposes a fast CNNs cascade face detector with multi-task learning and network acceleration techniques. In particular, the first stage of the detector is an elaborately designed fully convolutional network with a novel pyramid architecture, which can generate multi-scale face proposals efficiently with no more than twice image resizing operations. Several network compression and acceleration techniques including multi-layer merging and knowledge distilling are adopted to further improve inference speed. In addition, online and offline hard sample mining are jointly utilized to further strengthen the power of networks. The experimental results on challenging FDDB show that the proposed face detector is comparable in performance with the-state-of-arts while the speed reaches astonishing 165 fps on Titan GPU.

Sarikan and Ozbayoglu (2018) have found that anomaly detection is an important part of an Intelligent Transportation System. In this study, image processing and machine learning techniques are used to detect anomalies in vehicle movements. These anomalies include standing and traveling in reverse direction.

Images are captured using CCTV cameras from front and rear side of the vehicle. This capability makes the results robust to the variations in operational and environmental conditions. Multiple consecutive frames are acquired for motion detection. Features such as edges and license plate corner locations are extracted for tracking purposes. Direction of the traffic flow is obtained from the trained classifier. K-nearest neighbor is chosen as the classifier model. The proposed method is evaluated on a public highway and promising detection results are achieved.

Sanjaya *et al* (2018) have investigated the facial tracking and expression recognition to developed Social Robot called SyPEHUL (System of Physics, Electronics, Humanoid robot and machine Learning). This social robot contains by 12 Degree of Freedom (DoF) for actuating robotic head based on Arduino microcontroller to display four type different facial expressions and tracking of the human face. In this research, expression recognition processed by an algorithm based on Python 2.7 (with OpenCV library) using Cascade Classification and LBPH Face Recognizer. The result shows that the implementation of facial tracking and expression recognition to Social Robot show a good accuracy of recognition rate and works well for Human-Robot Interaction.

Kurka and Salazarhave (2018) have found that modern applications of industrial automation and robotics are increasingly relying on image processing techniques. This paper shows ways in which image processing can be applied to solve actual problems in robotics and instrumentation. The paper starts by presenting the fundamentals of camera models, digital acquisition of images and selected processing techniques, followed by examples of applications of such knowledge. Four examples of image processing applications are shown: rim detection in automotive wheel images, dimensional verification of crankshafts, measurement of wheel alignment angles of a car and a stereo visual odometry algorithm for mobile

robotics. The examples not only illustrate the uses of different image processing techniques, but may also inspire the development of new robotic and industrial automation products.

Madhuram *et al* (2018) have described that face detection and recognition from an image or a video is a popular topic in biometrics research. Face recognition technology has widely attracted attention due to its enormous application value and market potential, such as real-time video surveillance system. It is widely acknowledged that the face recognition has played an important role in surveillance system as it doesn't need the object's co-operation, we design a real-time face recognition system based on IP camera and image set algorithm by way of OpenCV and Python programming development. The system includes three parts: Detection module, training module and recognition module.

CHAPTER-3

METHODOLOGY

3.1 OVERVIEW

It is a closed loop system in which the inputs and the outputs are depends on each other. The camera reads the input and the input video is processed through image processing technique by OpenCV and the output is executed at the Arduino microcontroller. The gun rotate and tilt based on the human face movement and it is triggered when the microcontroller sends the feedback signal.

3.2 SCHEMATIC DIAGRAM

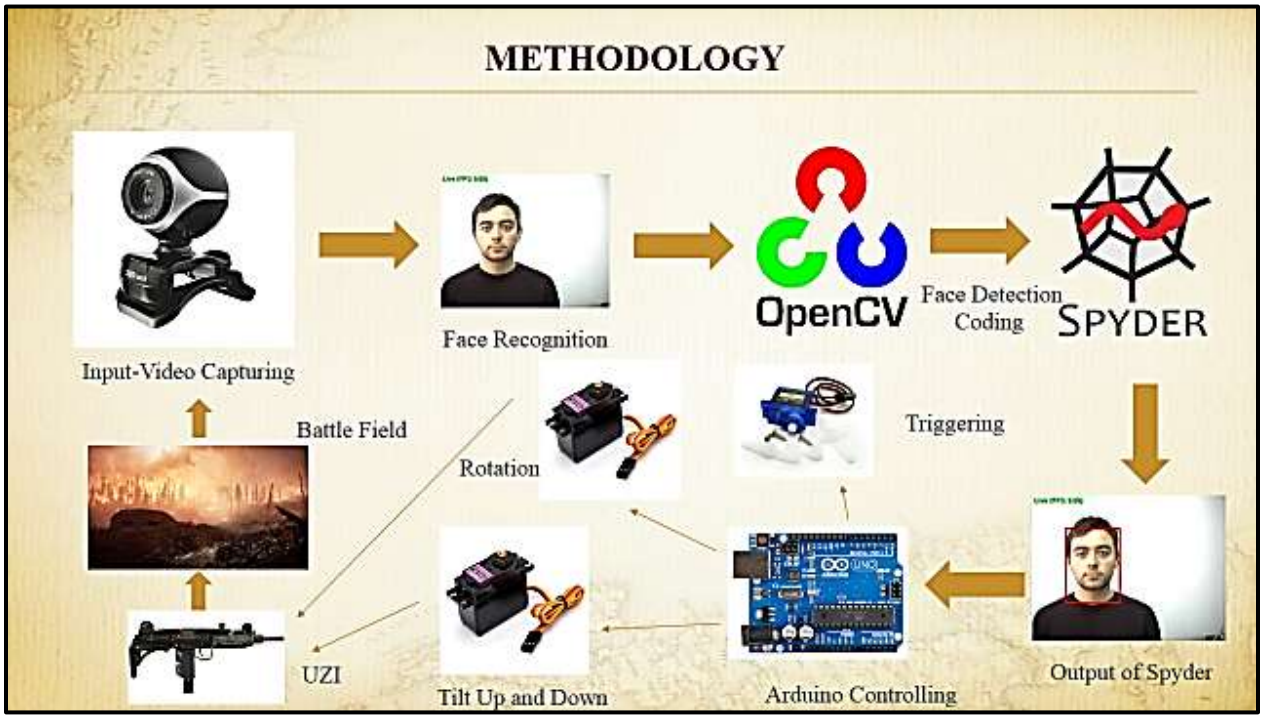

Figure 3.1 Schematic diagram

The above Figure 3.1 clearly explains the methodology of the project. In this chapter, the content mentioned in the schematic diagram are well explained

3.2.1 Input Video Capturing

The input for this prototype is captured using a webcam from the battle field. The web cam is a 10 mega pixel and it captures the video in actual colour. The capture video is passed into the OpenCV software for further operations.

3.2.2 Face Recognition

The video for face recognition is received from the webcam. From the captured video the human face is recognized by using OpenCV library files.

3.2.3 Face Detection and Coding

A program called haarcascade_frontalface_default.xml is call by a face detection program which is used to detect the human face from the captured video. Haarcascade_frontalface_default is a thousand-line program which describes the face structure and size of every humans. So, the software can easily identify the human face from the video. The received video is converted into gray color because processing the multicolor pixel is harder. The gray color video is process and the .xml file is used to identify the human face. After detecting the human face, a rectangle box is form over the face.

3.2.4 Output of Spyder – Detected Face

After compiling the program, the spyder software displays the output in the video format with a rectangle which shows the human face. The rectangle moves according the motion of the human face. The data of the rectangle is passed to the Arduino controller to drive the servo motors

3.2.5 Arduino Controlling

Arduino is loaded with the program to rotated the servomotors. But the servo motor remains stationery because it is actuating by the data passed from the spyder. The controller is waiting until the data to be received. After receiving the data from spyder it is used to drive the hardware like servomotors. The rotation of servo motor is directly proportional to the movement of the human face**.** The servo motor rotates according the movement of the human face.

3.2.6 Servo Motor Control

Servos are basically working on the angle passes from the micro controller. The micro controller generates the exact angle based on the data passed by the spyder software. This prototype is designed and fabricated with 3 servo motors, two 10kg/cm torque producing servo motor and one 1.4kg/cm torque producing servo motor. The 10kg/cm torque producing servo motors are used to rotate the base table and tilt the gun up and down. The 1.4kg torque producing servo motor is used to trigger the gun.

3.2.7 UZI Gun Movement

The gun can move in x and y axis within a specified range. The gun is fixed in the gun clamp. The gun clamp is fixed on the nylon rod, when the nylon rod tilted by the servo motor in y-axis the gun fixed in the gun clamp also tilts hence the up and down movement is obtained. The gun rotated in x-axis by a base plate using a servo motor.

3.2.8 Battle Field

The camera takes the input from the battle field, it captures the video from the enemy border area. The enemy detected range is depending on the camera quality we have been using. The entire system is a closed loop system because it collects the input from the battle field and it exit the output in the battle field. The input and output are depending on the each other.

3.3 UTILIZATION

This methodology is to be used in the border to protect our nation without any manpower. This method just needs a 12V electric supply which will be supply to it through a solar panel or a 12V battery.

CHAPTER – 4

CODING FOR IMAGE PROCESSING

4.1 IMAGE PROCESSING

Image processing is a technique in which the input is an image or a series of images or videos, such as photographs or frames of video. The output of image processing can be either an image or a set of characteristics or parameters related to the image. It also means "Analyzing and manipulating images with a computer". Image processing is performed in three steps

1. First, import images with an optical device like a scanner or a camera or directly through digital processing.
2. Second, manipulate or analyze the images in some way. This step can include image improvement and data summary, or the images are analyzed to find rules that aren't seen by the human eyes. For example, meteorologists use this processing to analyze satellite photographs.
3. Last, output the result of image processing. The result might be the image changed by some way or it might be a report based on analysis or result of the images.

4.1.1 Purpose of Image processing

The purpose of image processing is divided into 5 groups. They are:

- Visualization - Observe the objects that are not visible.
- Image sharpening and restoration - To create a better image.
- Image retrieval - Seek for the image of interest.

- Measurement of pattern – Measures various objects in an image.
- Image Recognition – Distinguish the objects in an image.

4.1.2 Applications of Digital Image Processing

Some of the major fields in which digital image processing is widely used are mentioned below.

- Image sharpening and restoration
- Medical field
- Remote sensing
- Transmission and encoding
- Machine/Robot vision
- Color processing
- Pattern recognition
- Video processing
- Microscopic Imaging

4.2 SOFTWARE DESCRIPTION

To detect and track the human face totally three software are adopted. They are

1. OpenCV
2. Anaconda
3. Spyder
4. Arduino

4.2.1 About OpenCV

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc. OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 14 million. The library is used extensively in companies, research groups and by governmental bodies.

Along with well-established companies like Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota that employ the library, there are many startups such as Applied Minds, VideoSurf, and Zeitera, that make extensive use of OpenCV. OpenCV's deployed uses span the range from stitching streetview images together, detecting intrusions in surveillance video in Israel, monitoring mine equipment in China, helping robots navigate and pick up objects at Willow Garage, detection of swimming pool drowning accidents in Europe, running interactive art in Spain and

New York, checking runways for debris in Turkey, inspecting labels on products in factories around the world on to rapid face detection in Japan.

It has C++, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. OpenCV leans mostly towards real-time vision applications and takes advantage of MMX and SSE instructions when available. A full-featured CUDA and OpenCL interfaces are being actively developed right now. There are over 500 algorithms and about 10 times as many functions that compose or support those algorithms. OpenCV is written natively in C++ and has a templated interface that works seamlessly with STL containers.

4.2.1.1 Python

Python is a high-level, dynamically typed multiparadigm programming language. Python code is often said to be almost like pseudocode, since it allows you to express very powerful ideas in very few lines of code while being very readable. There are currently two different supported versions of Python, 2.7 and 3.5. Somewhat confusingly, Python 3.0 introduced many backwards-incompatible changes to the language, so code written for 2.7 may not work under 3.5 and vice versa. Like most languages, Python has a number of basic types including integers, floats, Booleans, and strings. Python also has built-in types for complex numbers. Python includes several built-in container types: lists, dictionaries, sets, and tuples.

4.2.1.2 Python Packages

For scientific computing and computational modelling, we need additional libraries (so called *packages*) that are not part of the Python standard library. These allow us, for example, to create plots, operate on matrices, and use specialized numerical methods the packages we generally need are

- NumPy

- cv2
- pyserial

4.2.1.3 Numpy

Numpy is a highly optimized library for numerical operations. It gives a MATLAB-style syntax. All the OpenCV array structures are converted to-and-from numpy arrays. So whatever operations you can do in Numpy, you can combine it with OpenCV, which increases number of weapons in your arsenal.

So OpenCV-python is an appropriate tool for fast prototyping of computer vision problems.

4.2.1.4 pyserial

pySerial encapsulates the access for the serial port. It provides backends for Python running on Windows, OSX, Linux, BSD (possibly any POSIX compliant system) and IronPython. The module named "serial" automatically selects the appropriate backend.

4.2.2 Anaconda

Anaconda is a FREE enterprise-ready Python distribution for data analytics, processing, and scientific computing. Anaconda comes with Python 2.7 or Python 3.4 and 100+ cross-platform tested and optimized Python packages. All of the usual Python ecosystem tools work with Anaconda. Additionally, Anaconda can create custom environments that mix and match different Python versions (2.6, 2.7, 3.3 or 3.4) and other packages into isolated environments and easily switch between them using conda, our innovative multi-platform package manager for Python and other languages.

4.2.2.1 Using Python in Anaconda

Many people write Python code using a text editor like Emacs or Vim. Others prefer to use an IDE like Spyder, Wing IDE, PyCharm or Python Tools for Visual Studio. Spyder is a great free IDE that is included with Anaconda. To start Spyder, type the name spyder in a terminal or at the Command Prompt. The Python 2.7 version of Anaconda also includes a graphical Launcher application that enables you to start IPython Notebook, IPython QTConsole, and Spyder with a single click. On Mac, double click the Launcher.app, found in your ~/anaconda directory (or wherever you installed Anaconda). On Windows, you will find Launcher in your Start Menu. The Start Menu also has an Anaconda Command Prompt that, regardless of system and install settings, will launch the Python interpreter installed via Anaconda. This is particularly useful for troubleshooting. conda, an open source package and environment manager developed by Continuum Analytics. conda is designed to work very well with the complex binary packages found in the scientific and numerical Python ecosystem, which are oftentimes very difficult to build and install using standard Python packaging tools. When a conda package is available for a particular library, it is super easy to install it into Anaconda. it is also easy to update packages, install a particular version of a package, and mix and match packages into environments with your choice of Python version. To view the conda help menu, type the command conda -h from the terminal or command line.

4.2.2.2 Miniconda

Continuum Analytics also produces Miniconda installers. "Miniconda" only contains Python and conda and is much smaller than a full Anaconda installer. There are two variants of the installer: Miniconda is based on Python 2, while Miniconda3 is based on Python 3. Once Miniconda is installed, you can use the conda command to install any other packages and create environments (still containing any version

of Python you want). If you have a slow internet connection or limited disk space, Miniconda is the way to go.

4.2.2.3 Anaconda Add-Ons

You can also use conda to easily install the commercial packages from Continuum Analytics and other vendors into Anaconda. IOPro is a fast, memory-efficient Python interface for databases, data files, Amazon S3 and MongoDB. Accelerate includes NumbaPro, a compiler that targets multi-core CPUs and GPUs directly from simple Python syntax, and MKL Optimizations, which accelerates NumPy, SciPy, scikit-learn and NumExpr using Intel's Math Kernel Library.

4.2.3 Spyder

Spyder is the Scientific PYthon Development EnviRonment: A powerful interactive development environment for the Python language with advanced editing, interactive testing, debugging and introspection features and a numerical computing environment thanks to the support of IPython (enhanced interactive Python interpreter) and popular Python libraries such as NumPy (linear algebra), SciPy (signal and image processing) or matplotlib (interactive 2D/3D plotting).

Spyder may also be used as a library providing powerful console-related widgets for your PyQt-based applications – for example, it may be used to integrate a debugging console directly in the layout of your graphical user interface.

4.2.4 Arduino

Arduino is an open source prototyping platform based on easy to use hardware and software. It is a small micro controller board with USB plug to connect with computer and a number of connections with external electronics such as motors, LED, relays, and light sensors. It can be programmed to anything by sending set of

instructions to the Micro controller on the board. Arduino programming is done using Arduino Software.

Power supply to Arduino is given by

- USB Power Supply
- 7-12 V dc
- Regulated 5V dc

4.2.4.1 Types of Arduino

- Arduino UNO
- Arduino 101
- Arduino PRO
- Arduino Micro
- Arduino PRO Mini
- Arduino Nano

Depending on the requirement, processing power, input pins and size we can select the board.

4.2.4.2 Arduino UNO

Arduino Uno is basic and Inexpensive. It uses 8 bit AT mega328 Micro controller. It is available in through hole and SMD and it is shown in the Figure 4.1. It consists of 14 digital pins and Operated at 5v and 40mA. The Pins 0 and 1 are used to receive and transmit serial data (serial communication) which can be used in several ways like programming the board and communicate the user through serial monitor.

Figure 4.1. Arduino UNO

The Pins 2 and 3 can be configured to trigger the interrupt on a low value and the Pins 3,5,6,9,10 and 11 are 8 BIT PWM. The Pins 10 to 13 are used to send data between Micro controller and small peripherals such as SD CARDS, SENSORS and REGISTERS and the Pin 13is an in-built LED. It consists of 6 analog pins and it has 10-bit resolution.

The Power pins are

- Vin
- 5V and 3.3V used for power sensors
- Ground

4.3 PROGRAMMING

Programming is the main operating tool in image processing, so for image processing the following programs are listed from basics.

4.3.1 Original Image Program

This following program is used to read and display the original image, the Figure 4.2 (a) and Figure 4.2 (b) is the input and output of the program respectively.

Program

```
import cv2 as cv
imgFile = cv.imread(baby.jpg')
```

```
cv.imshow('babyoriginal', imgFile)
cv.waitKey(0)
cv.destroyAllWindows()
```

Figure 4.2 (a) baby.jpg (input)

Figure 4.2 (b) babyoriginal (output)

4.3.2 Gray Image Program

The following program is used to convert the image in to gray color. The Figure 4.3 (a) and Figure 4.3 (b) are the input and output of the following program respectively.

Program

```
import numpy as np
import cv2
img = cv2.imread('baby.jpg',0)
cv2.imshow('image',img)
k = cv2.waitKey(0)
k = cv2.waitKey(0) & 0xFF
if k == 27:         # wait for ESC key to exit
    cv2.destroyAllWindows()
elif k == ord('s'): # wait for 's' key to save and exit
    cv2.imwrite('babygray.png',img)
    cv2.destroyAllWindows()
```

Figure 4.3 (a) baby.jpg (input)

Figure 4.3 (b) babygray.png(output)

4.3.3 Actual Color Video Display Program

This program is used to capture the live video and display the video in actual color.

Program

```
import cv2
cap = cv2.VideoCapture(0)
while True:
    if cap.grab():
        flag, frame = cap.retrieve()
        if not flag:
            continue
        else:
            cv2.imshow('video', frame)
    if cv2.waitKey(10) == 27:
        break
```

Output:

After compiling it displays the live video in original color.

4.3.4 Gray Color Video Display Program

The following program is used to display the captured live video to display in gray color by removing the RGB.

Program

```
import numpy as np
import cv2
cap = cv2.VideoCapture(0)
while(True):
    # Capture frame-by-frame
    ret, frame = cap.read()
    # Our operations on the frame come here
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    # Display the resulting frame
    cv2.imshow('frame',gray)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
# When everything done, release the capture
cap.release()
cv2.destroyAllWindows()
```

Output:

After compilation it displays the captured video in gray color.

4.4 TYPES OF ALGORITHM ADOPTED

- Background subtraction algorithm
- Face and eye detection algorithm
- Face detection algorithm

4.4.1 Background Subtraction Algorithm

Background subtraction is a major preprocessing step in many vision-based applications. For example, consider the case of a visitor counter where a static camera takes the number of visitors entering or leaving the room, or a traffic camera extracting information about the vehicles etc. In all these cases, first you need to extract the person or vehicles alone. Technically, you need to extract the moving foreground from static background.

If you have an image of background alone, like an image of the room without visitors, image of the road without vehicles etc, it is an easy job. Just subtract the new image from the background. You get the foreground objects alone. But in most of the cases, you may not have such an image, so we need to extract the background from whatever images we have. It becomes more complicated when there are shadows of the vehicles. Since shadows also move, simple subtraction will mark that also as foreground. It complicates things.

Several algorithms were introduced for this purpose. OpenCV has implemented three such algorithms which are very easy to use. We will see them one-by-one.

4.4.1.1 Types of Background Subtraction

1. Background Subtractor MOG
2. Background Subtractor MOG2
3. Background Subtractor GMG

4.4.1.2 Background Subtractor MOG

It is a Gaussian Mixture-based Background/Foreground Segmentation Algorithm. It was introduced in the paper "An improved adaptive background mixture model for real-time tracking with shadow detection" by P. KadewTraKuPong and R. Bowden in 2001. It uses a method to model each background pixel by a mixture of K Gaussian distributions (K = 3 to 5). The weights of the mixture represent the time proportions that those colors stay in the scene. The probable background colors are the ones which stay longer and more static.

While coding, we need to create a background object using the function, cv.createBackgroundSubtractorMOG(). It has some optional parameters like length of history, number of gaussian mixtures, threshold etc. It is all set to some default

values. Then inside the video loop, use backgroundsubtractor.apply() method to get the foreground mask. An example of MOG program is given below.

```
import numpy as np
import cv2 as cv
cap = cv.VideoCapture('vtest.avi')
fgbg = cv.bgsegm.createBackgroundSubtractorMOG()
while(1):
    ret, frame = cap.read()
    fgmask = fgbg.apply(frame)
    cv.imshow('frame',fgmask)
    k = cv.waitKey(30) & 0xff
    if k == 27:
        break
cap.release()
cv.destroyAllWindows()
```

4.4.1.3 Background Subtractor MOG2

It is also a Gaussian Mixture-based Background/Foreground Segmentation Algorithm. It is based on two papers by Z.Zivkovic, "Improved adaptive Gaussian mixture model for background subtraction" in 2004 and "Efficient Adaptive Density Estimation per Image Pixel for the Task of Background Subtraction" in 2006. One important feature of this algorithm is that it selects the appropriate number of gaussian distribution for each pixel. (Remember, in last case, we took a K gaussian distributions throughout the algorithm). It provides better adaptability to varying scenes due illumination changes etc.

As in previous case, we have to create a background subtractor object. Here, you have an option of detecting shadows or not. If detectShadows = True (which is so by default), it detects and marks shadows, but decreases the speed. Shadows will be marked in gray color. An example for MOG2 program is given below.

```
import numpy as np
import cv2 as cv
cap = cv.VideoCapture('vtest.avi')
fgbg = cv.createBackgroundSubtractorMOG2()
while(1):
    ret, frame = cap.read()
    fgmask = fgbg.apply(frame)
    cv.imshow('frame',fgmask)
    k = cv.waitKey(30) & 0xff
    if k == 27:
        break
cap.release()
cv.destroyAllWindows()
```

4.4.1.4 Background Subtractor GMG

This algorithm combines statistical background image estimation and per-pixel Bayesian segmentation. It was introduced by Andrew B. Godbehere, Akihiro Matsukawa, and Ken Goldberg in their paper "Visual Tracking of Human Visitors under Variable-Lighting Conditions for a Responsive Audio Art Installation" in 2012. As per the paper, the system ran a successful interactive audio art installation called “Are We There Yet?” from March 31 - July 31 2011 at the Contemporary Jewish Museum in San Francisco, California.

It uses first few (120 by default) frames for background modelling. It employs probabilistic foreground segmentation algorithm that identifies possible foreground objects using Bayesian inference. The estimates are adaptive; newer observations are more heavily weighted than old observations to accommodate variable illumination. Several morphological filtering operations like closing and opening are done to remove unwanted noise. You will get a black window during first few frames.

It would be better to apply morphological opening to the result to remove the noises. An example for GMG program is given below.

```
import cv2 as cv
cap = cv.VideoCapture('vtest.avi')
kernel = cv.getStructuringElement(cv.MORPH_ELLIPSE,(3,3))
fgbg = cv.bgsegm.createBackgroundSubtractorGMG()
while(1):
    ret, frame = cap.read()
    fgmask = fgbg.apply(frame)
    fgmask = cv.morphologyEx(fgmask, cv.MORPH_OPEN, kernel)
    cv.imshow('frame',fgmask)
    k = cv.waitKey(30) & 0xff
    if k == 27:
        break
cap.release()
cv.destroyAllWindows()
```

4.4.1.5 Background Subtraction Program

The following program is used to extract the human from the live video by using a background image here the background image is “rough.jpg”. The captured video is converted into frames and chance it color to gray for increasing the processing speed. If the video color is unique the system can process quickly. This is one of the methods we have initially chosen for our project.

Program

```
import numpy as np
import cv2
video = "xx.avi"
cap = cv2.VideoCapture(0)
bg = cv2.imread('rough.jpg',0)
#bg = cv2.imread("background.jpg")
while True:
    ret, frame = cap.read()
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
```

```
    original_frame = frame.copy()
    if ret:
        # get foremask?
        fgmask = gray – bg
 # filter kernel for denoising:
kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (3, 3))
opening = cv2.morphologyEx(fgmask, cv2.MORPH_OPEN, kernel)
closing = cv2.morphologyEx(opening, cv2.MORPH_CLOSE, kernel)
# Dilate to merge adjacent blobs
dilation = cv2.dilate(closing, kernel, iterations = 2)
# show fg:dilation
cv2.imshow('fg mask', dilation)
cv2.imshow('original', original_frame)
k = cv2.waitKey(30) & 0xff
if k == 27:
      cap.release()
       cv2.destroyAllWindows()
           break
 else:
       break
```

Output:

It shows the human subtracted from the background image but in gray color.

4.4.2 Face and Eye Detection Algorithm

OpenCV comes with a trainer as well as detector. If you want to train your own classifier for any object like car, planes etc. you can use OpenCV to create one. Its full details are given here: Cascade Classifier Training.

Here we will deal with detection. OpenCV already contains many pre-trained classifiers for face, eyes, smile etc. Those XML files are stored in opencv/data/haarcascades/ folder. Let’s create face and eye detector with OpenCV.

First, we need to load the required XML classifiers. Then load our input image (or video) in grayscale mode.

Program

```
import numpy as np
import cv2
face_cascade                                                              =
cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
eye_cascade = cv2.CascadeClassifier('haarcascade_eye.xml')
img = cv2.imread('sachin.jpg')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
faces = face_cascade.detectMultiScale(gray, 1.3, 5)
for (x,y,w,h) in faces:
    img = cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)
    roi_gray = gray[y:y+h, x:x+w]
    roi_color = img[y:y+h, x:x+w]
    eyes = eye_cascade.detectMultiScale(roi_gray)
    for (ex,ey,ew,eh) in eyes:
        cv2.rectangle(roi_color,(ex,ey),(ex+ew,ey+eh),(0,255,0),2)

cv2.imshow('img',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Now we find the faces in the image. If faces are found, it returns the positions of detected faces as Rect(x,y,w,h). Once we get these locations, we can create a ROI for the face and apply eye detection on this ROI (since eyes are always on the face !!!). The output is shown as shown in Figure 4.4.

Output:

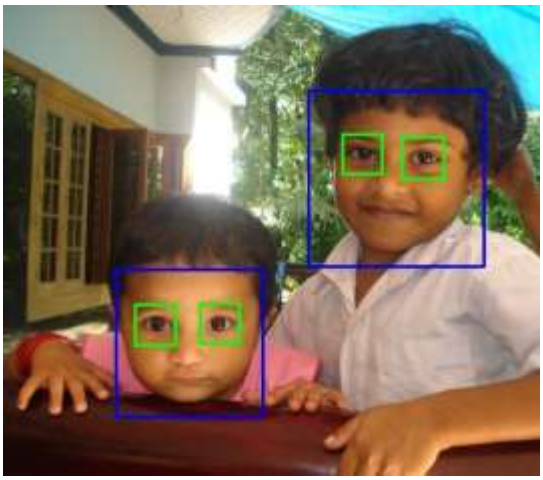

Figure 4.4 Face and eye detection

`4.4.2.1 Face and Eye Detection Program

The following program is used to detect and track the human face and eye movement. The rectangle box which covers the face and eye move according to the human face movement

Program

```
import cv2
cap = cv2.VideoCapture(0)
cap.set(3, 640) #WIDTH
cap.set(4, 480) #HEIGHT
face_cascade                                                    =
cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
eye_cascade = cv2.CascadeClassifier('haarcascade_eye.xml')
while(True):
    # Capture frame-by-frame
    ret, frame = cap.read()
    # Our operations on the frame come here
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray, 1.3, 5)
    print(len(faces))
    # Display the resulting frame
    for (x,y,w,h) in faces:
```

```
        cv2.rectangle(frame,(x,y),(x+w,y+h),(255,0,0),2)
        roi_gray = gray[y:y+h, x:x+w]
        roi_color = frame[y:y+h, x:x+w]
        eyes = eye_cascade.detectMultiScale(roi_gray)
        for (ex,ey,ew,eh) in eyes:
            cv2.rectangle(roi_color,(ex,ey),(ex+ew,ey+eh),(0,255,0),2)
    cv2.imshow('frame',frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
# When everything done, release the capture
cap.release()
cv2.destroyAllWindows()
```

Output:

After compiling the above program, it displays a video with rectangle which covers over the eye and face.

4.4.3 Face Detection Algorithm

This program uses a haarcascade_frontal_face.xml program to recognize the face and the movement of the face. After go through the all algorithm this is the final algorithm we adopt for our project. It works exactly as per our need.

4.4.3.1 Face Detection Program

This program is used to detect the data of the rectangle which covers over the face. This data is directly feed in to the Arduino microcontroller to rotate the servo motor. To transfer the output of this program to Arduino a command called pyserial is used.

Program

```
import cv2
import numpy as np
import serial
import struct
import time
a=0
```

```
b=0
x=0
y=0
ser = serial.Serial('com4',9600)
time.sleep(2)
font=cv2.FONT_HERSHEY_SIMPLEX
FaceCascade=cv2.CascadeClassifier('haarcascade_frontalface_default.xml'
)
cap=cv2.VideoCapture(0)
def BoxDraw():
cv2.line(flipit,(213,0),(213,480),(255,0,0),2)
cv2.line(flipit,(426,0),(426,480),(255,0,0),2)
cv2.line(flipit,(0,160),(640,160),(255,0,0),
cv2.line(flipit,(0,320),(640,320),(255,0,0),2)
pass
while True:
    ret,frame=cap.read()
    flipit=cv2.flip(frame,1)
    gray=cv2.cvtColor(flipit,cv2.COLOR_BGR2GRAY)
    face=FaceCascade.detectMultiScale(gray,1.2,4)
    try:
        for (x1,y1,w1,h1) in face:
            a=int((2*x1+w1)/2)
            b=int((2*y1+h1)/2)
            x=int(a/3.66)
            y=int(b/2.55)
            ser.write(struct.pack('>BB', x,y))
            cv2.rectangle(flipit,(x1,y1),(x1+w1,y1+h1),(0,255,0),2)
     except:
      pass
    cv2.imshow('flipit',flipit)
    k=cv2.waitKey(20) & 0xff
    if k==27:
        break
cap.release()
cv2.destroyAllWindows()
```

Output:

After compilation of this program a rectangle box covers the face and identifies the store of the rectangle.

4.5 ARDUINO PROGRAM FOR FACE DETECTION AND TRACKING

This program receives the input from the spyder output by pyserial command. The output from the spyder are received in the form of data of the rectangle. The data is directly passed in to the program to rotate the servo motors at exact angle.

Program:

```
#include <Servo.h>
int data_x = 0;
int data_y = 0;
int data[1];
Servo myservo_x;
Servo myservo_y;// create servo object to control a servo
// twelve servo objects can be created on most boards

//int pos = 0;    // variable to store the servo position

void setup() {
 Serial.begin(9600);
 myservo_x.attach(9);  // attaches the servo on pin 9 to the servo object
 myservo_y.attach(10);
 myservo_x.write(90);
 myservo_y.write(90);
}

void loop() {
 while (Serial.available() >= 2) {
  //    data_x=Serial.read();
  //    data_y=Serial.read();
  for (int i = 0; i < 2; i++) {
    data[i] = Serial.read();
```

```
  }

  myservo_x.write(data[0]);
  myservo_y.write(data[1]);

  Serial.println(data[0]);
  Serial.println(data[1]);
 }

}
```

Output:

After compiling the program the servo is rotated according to the data from the spyder.

CHAPTER-5

CAD MODELLING

5.1 SOLID WORKS SOFTWARE

The solid works software offers different approaches to model generation like parts design, surface design etc., allowing the user to employ the combination of methods to create a model. The modelling application also provides “feature based” solid bodies by directly editing capabilities, which allow to change and update solid bodes by directing editing the dimension of a solid feature or by using other geometric construction techniques.

5.2 DESIGN PROCESS

The various steps to create the model by using the solid works software package are,

- Getting a 2-D (Two Dimensional) sketch of the model.
- Operations carried out in solid works software are
    - ✓ Sketching
    - ✓ Dimensioning
    - ✓ Extruding
    - ✓ Chamfering
    - ✓ Assembling
    - ✓ Save drawing as solid works part file.

5.3 PART DIAGRAM

Our project involves number of parts, these parts are to be in perfect dimension, the parts are drawn by using solid works software. The part diagrams were drawn using simple commands like circle, rectangle, line, extrude, extrude cut,

hole wizard, circular pattern. The various parts involved in the assembly are shown in the following.

5.3.1 Base Table

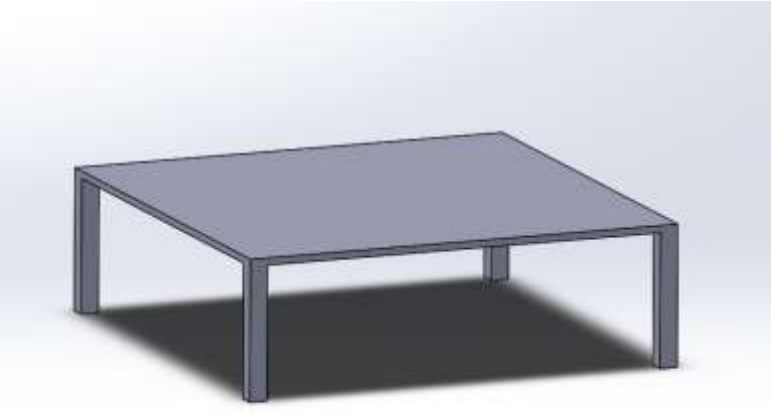

Figure 5.1 Base table

Base table is the place where the total assembly is fitted. The base table is shown in the Figure 5.1. It is square in size, and the base table is to be fix on its centre point.

5.3.2 Lower Base Plate

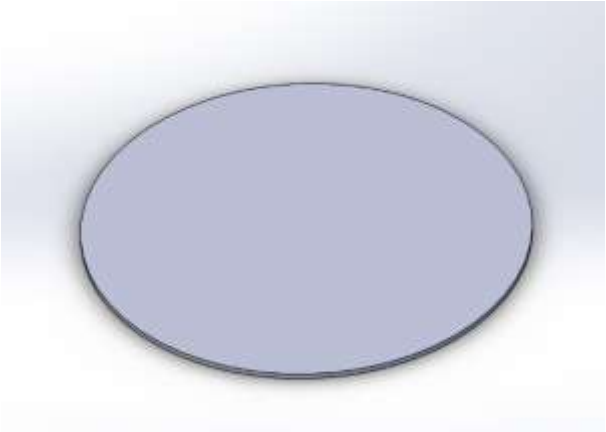

Figure 5.2 Lower base plate

Lower base plate is fixed on the base table by fasteners. It withstands the load of whole assembly. The lower base plate is shown in the Figure 5.2. this plate is fixed on the table in order to arrest shaft of the servo motor.

5.3.3 Upper Base Plate

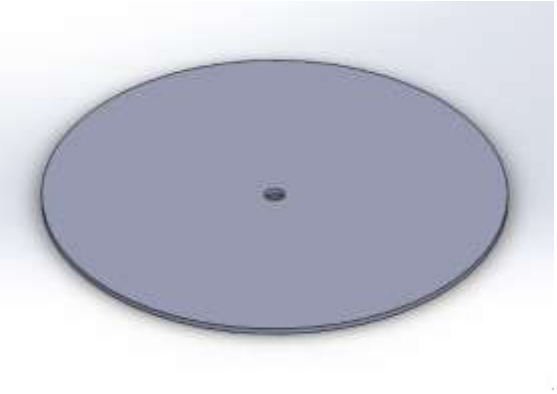

Figure 5.3 Upper base plate

Upper base plate rotates in clockwise and anti-clockwise direction with the help of stepper motor according to the face detected on the camera. This plate has seated on the flange and servo motor is connected with the flange through the hole in the plate and shaft of the servo motor is arrested in the flange. The upper plate is shown in the Figure 5.3.

5.3.4 Flange

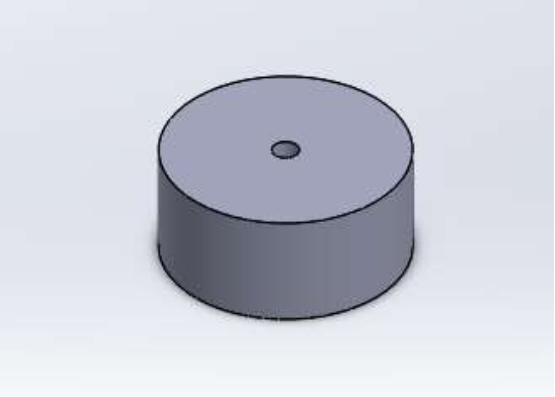

Figure 5.4 Flange

Flange is a device which is used to connect the shaft of the servo motor and upper base plate. The flange is fixed at the centre above the base plate to mount the stepper motor to rotate the upper base plate. The flange is shown in the Figure 5.4.

5.3.5 Left and Right Leg

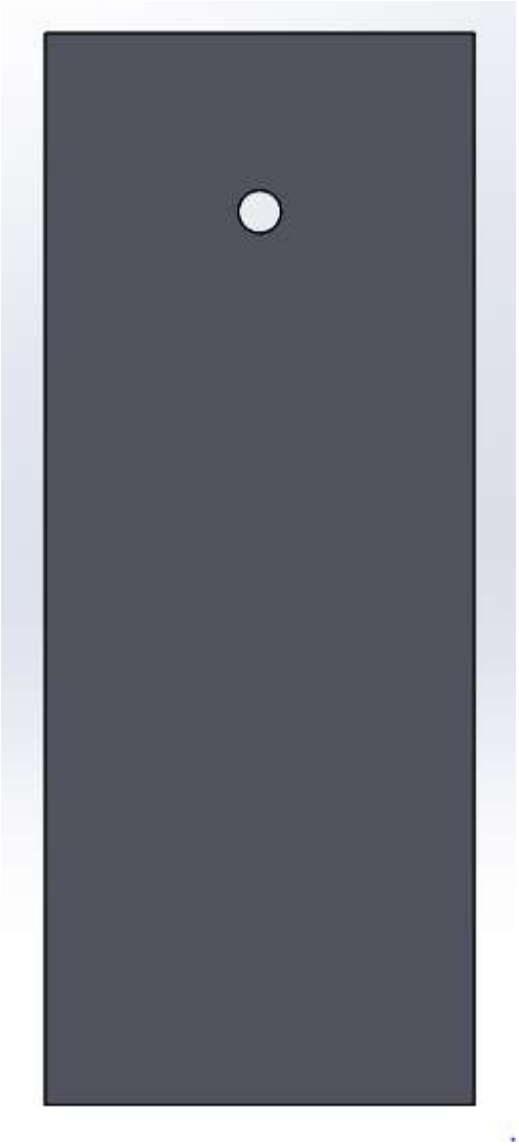

Figure 5.5 Left leg

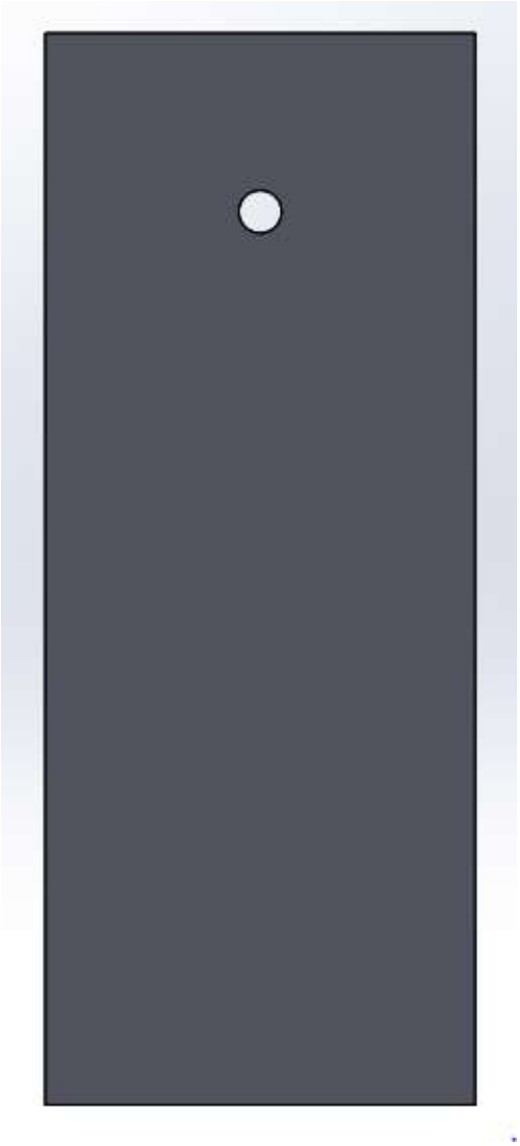

Figure 5.6 Right leg

Right and left leg is placed above the upper base plate with the help of L-clamp to hold the stepper motor, nylon rod, gun holder, gun clamp and gun. The Figure 5.5 and Figure 5.6 shows the left and right leg respectively.

5.3.6 Servo Motor

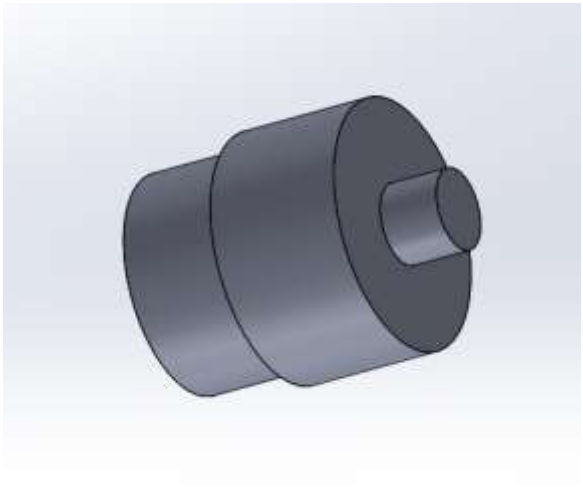

Figure 5.7 Servo motor

Servo motor placed in the upper base plate is to rotate the gun in clockwise and anticlockwise direction along x-axis and another servo motor is placed in the left leg, it is used to tilt the gun in upward and downward direction. The servo motor is shown in the Figure 5.7.

5.3.6.1 Calculation for Selection of Servo Motor

Mass of the base plate = 0.250kg

Mass of the gun =.380kg

Mass of the right leg =.050kg

Mass of the left leg =.050kg

Mass of the nylon rod =0.10kg

Acceleration due to gravity (a) = 9.81m/s^2

Radius of the circular disk (r) = 10.5cm

Total mass = Mass of the base plate + Mass of the gun + Mass of the right leg+ Mass of the left leg + Mass of the nylon rod

Total mass = .250 + .380 + .050 + .050 + 0.10

= 0.83kg

Force =Total mass × Acceleration

= 0.83 × 9.81

= 8.1423 N

Torque = Force × Radius

= 8.1423 × 10.5

= 85.4941 N-cm

In terms of torque in kg-cm

Torque = mass × radius

=.83 × 10.5

=8.715 kg-cm

In order to rotate a weight of .83kg kg, a servo motor of 10 kg-cm is selected.

5.3.7 Nylon Rod

Figure 5.8 Nylon rod

Nylon rod is fixed between the two legs and connected to the servo motor to tilt the gun holder which is fixed above the nylon rod. The Figure 5.8 shows the schematic diagram of nylon rod.

5.3.8 Gun Holder

Figure 5.9 Gun holder

Gun holder is used to hold the gun and also to reduce the recoiling force of the gun while firing. The gun holder is designed as an adjustable one to hold all type of guns. The above Figure 5.9 shows the gun holder with the UZI gun.

5.3.9 Final Assembly

Figure 5.10 Final Assembly

To make a product we have to assemble all the parts we have designed. To assemble each part a new assembly file is created to insert each and every component which is designed. The final assembly gives the aesthetic look of our all designed parts. The final assembly can rotate in x-axis and tilt in y-axis by the microcontroller commands to the servo motor. The Figure 5.10 shows the final assembly.

CHAPTER – 6

SIMULATION AND OUTPUT

To simulate the prototype the following program is to be compile and load in the Arduino microcontroller.

6.1 FACE DETECTION OPENCV PROGRAM

Face detection program is adopted for simulate the prototype. This program is used to detect the data of the rectangle which covers over the face. This data is directly feed in to the Arduino microcontroller to rotate the servo motor. To transfer the output of this program to Arduino a command called pyserial is used.

6.1.1 Program Embedded:

```
import cv2
import numpy as np
import serial
import struct
import time
A=0
b=0
x=0
y=0
ser = serial.Serial('com4',9600)
time.sleep(2)
font=cv2.FONT_HERSHEY_SIMPLEX
FaceCascade=cv2.CascadeClassifier('haarcascade_frontalface_default.xml'
)
cap=cv2.VideoCapture(0)
def BoxDraw():
cv2.line(flipit,(213,0),(213,480),(255,0,0),2)
cv2.line(flipit,(426,0),(426,480),(255,0,0),2)
cv2.line(flipit,(0,160),(640,160),(255,0,0),
cv2.line(flipit,(0,320),(640,320),(255,0,0),2)
```

```
pass
while True:
    ret,frame=cap.read()
    flipit=cv2.flip(frame,1)
    gray=cv2.cvtColor(flipit,cv2.COLOR_BGR2GRAY)
    face=FaceCascade.detectMultiScale(gray,1.2,4)
    try:
        for (x1,y1,w1,h1) in face:
            a=int((2*x1+w1)/2)
            b=int((2*y1+h1)/2)
            x=int(a/3.66)
            y=int(b/2.55)
            ser.write(struct.pack('>BB', x,y))
            cv2.rectangle(flipit,(x1,y1),(x1+w1,y1+h1),(0,255,0),2)
##            cv2.circle(flipit,(a,b),3,(0,0,255),-1)
     except:
      pass
    cv2.imshow('flipit',flipit)
    k=cv2.waitKey(20) & 0xff
    if k==27:
        break
cap.release()
cv2.destroyAllWindows()
```

Output:

After compilation of this program a rectangle box covers the face and identifies the store of the rectangle.

6.2 ARDUINO PROGRAM TO RUN HARDWARE

The following program drives the hardware according to the data passes by the spyder.

Program:

```
#include <Servo.h>
int data_x = 0;
int data_y = 0;
```

```
int data[1];
Servo myservo_x;
Servo myservo_y;// create servo object to control a servo
// twelve servo objects can be created on most boards

//int pos = 0;    // variable to store the servo position

void setup() {
 Serial.begin(9600);
 myservo_x.attach(9);  // attaches the servo on pin 9 to the servo object
 myservo_y.attach(10);
 myservo_x.write(90);
 myservo_y.write(90);
}

void loop() {
 while (Serial.available() >= 2) {
  //    data_x=Serial.read();
  //    data_y=Serial.read();
  for (int i = 0; i < 2; i++) {
    data[i] = Serial.read();
  }

  myservo_x.write(data[0]);
  myservo_y.write(data[1]);

  Serial.println(data[0]);
  Serial.println(data[1]);
 }

}
```

Output:

After compiling the program, the servo is rotated according to the data from the spyder.

6.3 OUTPUT

After compiling the above program the following output is executed. Inorder to get output the following steps are to be taken.

6.3.1 Intial Setup

Figure 6.1 Initial setup

The above Figure 6.1 shows the initial stage of the operation which shows the original image. It shows only the face without rectangle because the program is not compiled.

6.3.2 Detected Face

Figure 6.2 Detected face

The above Figure 6.2 shows the output of the program. The objective of program is to form a rectangular box on the detected human face and it moves when the human moves.

6.3.3 Data Passage

Figure 6.3 Data passage

After recognize of human face by opencv and spyder, the compiled data of the rectangle is passed by pyserial command to arduino microcontroller. The servo motor respond based on data passed by arduino as shown in Figure 6.3.

6.3.4 Rotation

Figure 6.4 Rotation

The above image Figure 6.4 shows that if the person face is detected left side. The servo motor turns the gun towards the left and vice versa.

6.3.5 Tilting

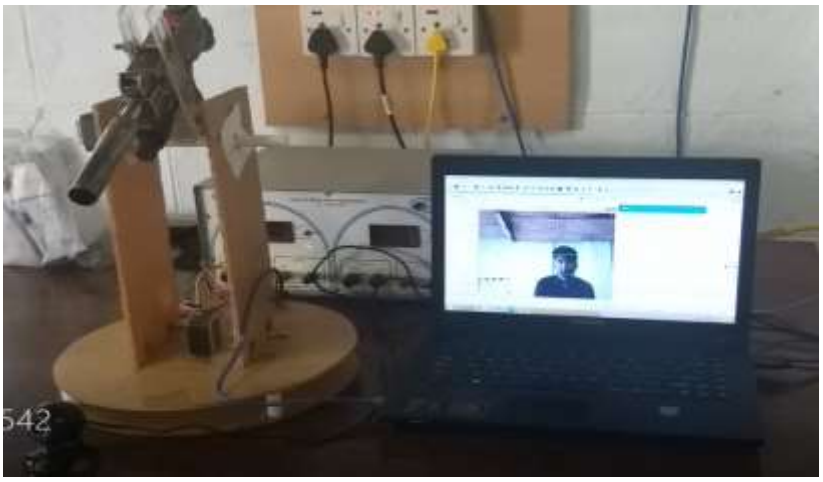

Figure 6.5 Tilting

The above picture Figure 6.5 shows if the person face detected at top. The servo motor tilts the gun up and vice versa.

CHAPTER - 7

FABRICATION OF PROTOTYPE

7.1 LIST OF COMPONENTS

In order to fabricate the prototype, we need the following components as mentioned in Table 7.1.

Table 7.1 List of components used for fabrication

S.No	Components	Quantity
1	Acrylic sheet	4mm plate 1
2	Nylon rod	12mm rod 1
3	10kg-cm Servo motor	2
4	1.4kg-cm Servo motor	1
5	Arduino UNO	1
6	Jumper wires	40
7	Webcam	1
8	Data cable	1
9	UZI gun	1

7.1.1 Acrylic Sheet

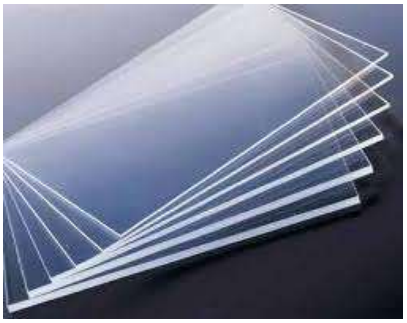

Figure 7.1 Acrylic sheet

Poly (methyl methacrylate) (PMMA), also known as acrylic, acrylic glass, or Plexiglas is a transparent thermoplastic often used in sheet form as a lightweight or shatter-resistant alternative to glass. The same material can be used as a casting resin, in inks and coatings, and has many other uses. The acrylic sheet is shown in the Figure 7.1.

Although not a type of familiar silica-based glass, the substance, like many thermoplastics, is often technically classified as a type of glass (in that it is a non-crystalline vitreous substance) hence it's occasional historical designation as acrylic glass. Chemically, it is the synthetic polymer of methyl methacrylate.

7.1.2 Nylon Rod

Figure 7.2 Nylon rod

Nylon rod is a generic designation for a family of synthetic polymers, based on aliphatic or semi-aromatic polyamides. Nylon is a thermoplastic silky material that can be melt-processed into fibers, films, or shapes .It is made of repeating units linked by amide links. Nylon polymers can be mixed with a wide variety of additives to achieve many different property variations. Nylon polymers have found significant commercial applications in fabric and fibers (apparel, flooring and rubber reinforcement), in shapes (molded parts for cars, electrical equipment, etc.). Nylon rod is shown in the Figure 7.2.

7.1.3 Servo Motor

Figure 7.3 Servo Motor

A servomotor is a rotary actuator or linear actuator that allows for precise control of angular or linear position, velocity and acceleration.It consists of a suitable motor coupled to a sensor for position feedback. It also requires a relatively sophisticated controller, often a dedicated module designed specifically for use with servomotors.

Servomotors are not a specific class of motor although the term servomotor is often used to refer to a motor suitable for use in a closed-loop control system. Servomotors are used in applications such as robotics, CNC machinery or automated manufacturing.The input to its control is a signal (either analogue or digital) representing the position commanded for the output shaft.

The motor is paired with some type of encoder to provide position and speed feedback. In the simplest case, only the position is measured. The measured position of the output is compared to the command position, the external input to the controller. If the output position differs from that required, an error signal is generated which then causes the motor to rotate in either direction, as needed to bring the output shaft to the appropriate position. As the positions approach, the error signal reduces to zero and the motor stops. Servo motor is shown in the Figure 7.3.

7.1.4 Arduino UNO

Figure 7.4 Arduino UNO

The Arduino UNO is shown in the Figure 7.4, it is an open-source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino.cc. The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits. The board has 14 Digital pins, 6 Analog pins, and programmable with the Arduino IDE (Integrated Development Environment) via a type B USB cable. It can be powered by a USB cable or by an external 9 volt battery, though it accepts voltages between 7 and 20 volts. It is also similar to the Arduino Nano and Leonardo. The hardware reference design is distributed under a Creative Commons Attribution Share-Alike 2.5 license and is available on the Arduino website. Layout and production files for some versions of the hardware are also available. "Uno" means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Uno board and version 1.0 of Arduino Software (IDE) were the reference versions of Arduino, now evolved to newer releases. The Uno board is the first in a series of USB Arduino boards, and the reference model for the Arduino platform. The ATmega328 on the Arduino Uno comes pre-programmed with a boot loader that allows uploading new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol. The Uno also

differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it uses the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

7.1.5 Jumper Wires

Figure 7.5 Jumper fires

A jump wire is shown in the Figure 7.5, it (also known as jumper wire, or jumper) is an electrical wire, or group of them in a cable, with a connector or pin at each end (or sometimes without them – simply "tinned"), which is normally used to interconnect the components of a breadboard or other prototype or test circuit, internally or with other equipment or components, without soldering.

Individual jump wires are fitted by inserting their "end connectors" into the slots provided in a breadboard, the header connector of a circuit board, or a piece of test equipment.

There are different types of jumper wires. Some have the same type of electrical connector at both ends, while others have different connectors. Some common connectors are:

- Solid tips – are used to connect on/with a breadboard or female header connector. The arrangement of the elements and ease of insertion on a breadboard allows increasing the mounting density of both components and jump wires without fear of short-circuits. The jump wires vary in size and color to distinguish the different working signals.
- Crocodile clips – are used, among other applications, to temporarily bridge sensors, buttons and other elements of prototypes with components or equipment that have arbitrary connectors, wires, screw terminals, etc.
- Banana connectors – are commonly used on test equipment for DC and low-frequency AC signals.
- Registered jack – are commonly used in telephone (RJ11) and computer networking (RJ45).
- RCA connectors – are often used for audio, low-resolution composite video signals, or other low-frequency applications requiring a shielded cable.
- RF connectors – are used to carry radio frequency signals between circuits, test equipment, and antenna

7.1.6 Webcam

Figure 7.6 Webcam

A webcam is a video camera that feeds or streams its image in real time to or through a computer to a computer network it is shown in the Figure 7.6. When "captured" by the computer, the video stream may be saved, viewed or sent on to other networks travelling through systems such as the internet, and e-mailed as an attachment. When sent to a remote location, the video stream may be saved, viewed or on sent there. Unlike an IP camera (which connects using Ethernet or Wi-Fi), a webcam is generally connected by a USB cable, or similar cable, or built into computer hardware, such as laptops.

The term "webcam" (a clipped compound) may also be used in its original sense of a video camera connected to the Web continuously for an indefinite time, rather than for a particular session, generally supplying a view for anyone who visits its web page over the Internet. Some of them, for example, those used as online traffic cameras, are expensive, rugged professional video cameras.

7.1.7 Data Cable

Figure 7.7 Data cable

A data cable is any media that allows baseband transmissions (binary 1,0s) from a transmitter to a receiver. It is shown in the Figure 7.7.

Examples Are:

- Networking Media
- Ethernet Cables (Cat5, Cat5e, Cat6, Cat6a)
- Optical fiber cable; see fiber-optic communication
- Serial cable
- Telecommunications Cable (Cat2 or telephone cord)
- USB cable

USB (abbreviation of Universal Serial Bus) is an industry standard that establishes specifications for cables, connectors and protocols for connection, communication and power supply between personal computers and their peripheral devices.

USB was designed to standardize the connection of peripherals like keyboards, pointing devices, digital still and video cameras, printers, portable media players, disk drives and network adapters to personal computers, both to communicate and to supply electric power. It has largely replaced interfaces such as serial ports and parallel ports, and has become commonplace on a wide range of devices.

7.1.8 UZI gun

Figure 7.8 UZI gun

The Uzi is a family of Israeli open-bolt, blowback-operated submachine guns. Smaller variants are often considered to be machine pistols. The Uzi was one of the first weapons to use a telescoping bolt design which allows the magazine to be housed in the pistol grip for a shorter weapon. It is shown in Figure 7.8.

The first Uzi submachine gun was designed by Major Uziel Gal in the late 1940s. The prototype was finished in 1950. First introduced to IDF Special Forces in 1954, the weapon was placed into general issue two years later. The Uzi has found use as a personal defence weapon by rear-echelon troops, officers, artillery troops and tankers, as well as a frontline weapon by elite light infantry assault forces.

The Uzi has been exported to over 90 countries. Over its service lifetime, it has been manufactured by Israel Military Industries, FN Herstal, and other manufacturers. From the 1960s through the 1980s, more Uzi submachine guns were sold to more military, law enforcement and security markets than any other submachine gun ever made.

7.2 PROCESS FLOW

There are various operations involved in fabricating the sentinel gun are displayed in the following Figure 7.9

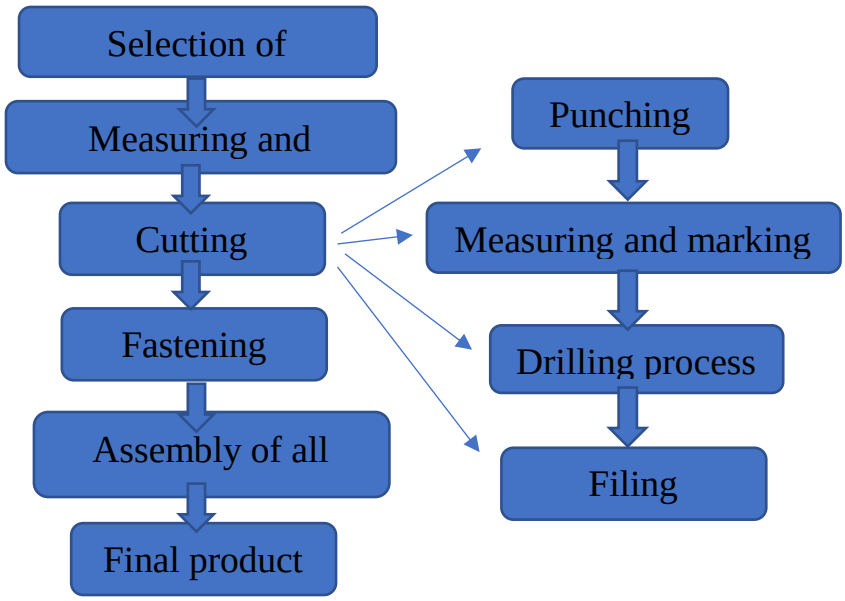

Figure 7.9 Flow of Process

7.2.1 Selection of Materials

Figure 7.10 Acrylic sheet

Large size acrylic sheet is cut to required size using jig saw. It is shown in the Figure 7.10. The required sizes are cut based on the prototype design drafted in the solidworks software.

Figure 7.11 Nylon rod

Nylon rod of diameter 12mm is selected from the store. It is shown in the Figure 7.11.

7.2.2 Measuring and Marking

Figure 7.12 Measuring

Before cutting the raw material into pieces the required cross section is to be measured using the steal ruler. The required dimensions are measured from the prototype design is drafted in solidworks. Measuring is shown in the Figure 7.12.

Figure 7.13 Marking

The measured work piece is now marked using pen or pencil. It is shown in the Figure 7.13. Marking in workpiece enables the accurate cutting.

7.2.3 Cutting Process

Figure 7.14 Cutting process

After marking the required dimension on the raw material cutting has been done using the jigsaw tool. Before on the jigsaw the acrylic sheet is fixed properly to avoid the devieation. The raw material cutting is shown in the Figure 7.14

7.2.4 Punching

The marked place is to be punched for drill a hole, punching is a process of making an indentation over the surface of the work material. Which helps to avoid the slippage of drill bit from the work piece.

7.2.5 Drilling Process

Figure 7.15 Drilling process

The linkages are joint by bolts and nuts, the bolts and nuts are connected in to the linkages by means of drilling a hole. Place the work in the vise and select the required drill bit to drill a hole. The Figure 7.15 represent the drilling process.

7.2.6 Filing

Figure 7.16 Filing

Filing is the operation which is done to remove the unwanted burr and to provide the good surface finish. It is done by using file tool. The Figure 7.16 shows the file operation.

7.2.7 Fastening

Figure 7.17 Fastening

The cut raw materials are joint using bolts and nuts, the left and right leg are assembled over the rotating base plate and the nylon rod is fixed between the two legs. It is shown in the Figure 7.17.

7.2.8 Assembly of All Parts

Figure 7.18 Final product

For fine moment of the base plate supports are necessary, here the supports are given in the fixed base plate at 0, 90, 180 and 270 respectively. A circlip is provided at the end of the nylon shaft to hold the leg at 90-degree Figure 7.17 shows the final assembly of 3D design the manufacturing product is shown in the Figure 7.18, the fabricated model is as per the design.

CHAPTER-8

ADVANTAGES & APPLICATIONS

8.1 ADVANTAGES

- It protects our soldiers from enemy attack.
- It reduces the count of death in borders.
- It avoids the periodic alteration of soldiers in the borders.
- It enhances easy tracking of enemies in borders.
- It guards the border for whole day.
- The man power is highly reduced.
- No need of soldiers for operating the equipment.
- It avoids enemy attack in public places.
- It can detect and track the multiple faces at the same time.
- It headshots the enemy exactly.
- It can be operating at night mode by using infrared camera.
- It continuously shot the enemy in a single trigger.

8.2 APPLICATION

- It is suitable in any places in the border.
- It is placed near to the army camp for surveillance.
- It replaces the soldiers in the border.
- Single sentinel gun can cover the battle field at a range of 120 degree.
- It can kill flying enemy.
- It can hold any gun by its versatile gun clamp.

CHAPTER -9

CONCLUSION

The theme of the work is to design a versatile gun holder and automatic aiming sentinel gun. To achieve this, image processing technology is applied and integrated in the newly designed gun holder. Web Camera is used along with the gun holder to capture the images. It acquires the live videos which should be then further analyzed for face detection and tracking. Precise coding is made for face detection and tracking algorithms. The captured video from the web camera is fed to spyder software where the coding is compiled in OpenCV platform. It gives the output of face detection and tracking of the captured image sequences and it is then fed to Arduino microcontroller. In Arduino, the face tracking is coupled with the servo motors which directly controls the Gun for tilting, rotation and triggering. Thus, it could fire the enemies automatically when they comes into the range of the camera.

This work will be useful for our country in Defence Sectors. It could save the Indian soldiers from natural calamities, sudden terrorist attacks and reduces guarding difficulties. In future, response rate of the face detection and firing accuracy will be improved by implementing high end DSLR camera and nanosecond controller. The sentinel gun will guard the borders of our nation with more reliability and with full safety.

CHAPTER – 10

FUTURE WORK

10.1 FUTURE WORK

- Microcontroller board will be replaced by Nano controller board.
- Raspberry-pi will be used in future to increase the accuracy.
- Prototype will be fabricated to exact design.
- Web camera will replace by DSLR camera to increase the operating range.
- The photographs and dress code of Indian army is feed in to the operating system to aim and fire only the terrorist or foreign military.

CHAPTER 11

OUTCOMES OF THE PROJECT

Table 11.1 Project Outcomes

Project Outcome No	Project Course Outcomes expected	Knowledge Level	Relevance of work carried
1.	Identifying a potential problem based on literature survey/ impending industrial/real Time needs.	K4	Based on the literature surveys, the death of our Indian army has been reported. In order to solve this and to serve our nation, this sentinel gun has been designed for autonomous guarding.
2.	Categorizing various solution methodologies to solve problem taken for study	K4	To develop such prototype using the image processing technique and integrated it with the Arduino board.
3.	Development of solution relevant to the problem.	K5	The python programming for Face Detection and Tracking has been developed and integrated it with Arduino for controlling the servo motors.

4.	Analyze design/experimental results.	K5	The coding is compiled in OpenCV, Spyder and the output is verified. The results show that the Face is detected with complete accuracy and tracked with a rectangular box.
5	Draw conclusion based on analysis and recommend solution to potential engineering problems	K5	It can be implemented in any environmental condition along the Indian border for autonomous guarding and it will be useful for our Indian soldiers to avoid unnecessary death.

PROGRAM OUTCOME RELEVANCE TO THE PROJECT

PO1: **Engineering knowledge**

Gathered knowledge about image processing which will be applied in defense application. Several journals, books have been referred to grab the basic idea for the project objective.

PO2: Problem analysis

Problem identification of this project is to restrict the periodic death of soldiers in borders. To guard the Indian border for 24 hours in an unsafe environment.

PO3: Development of solutions

To overcome this problem, we design a prototype to automatically detect and track human face of the enemies and start the firing when the enemies are coming under the camera range.

PO4: Conduct investigations of complex problems

Based on the investigation of problem identified the face detection and tracking algorithms has been developed when enemies comes under the range.

PO5: Modern tool usage

Solid works is used for the designing purpose. Spyder and OpenCV software is meant compilation of program.

PO6: The Engineer and Society

In an unsafe environment also, the sentinel gun is automatically starts guarding and fire the when the enemies come into the range. It will be useful to our society by unwanted loses of the Indian Soldiers to the nation.

PO7: Environmental and sustainability

At the time of natural calamities lies snow storm, high temperature etc the sentinel gun automatically guards the border and it does not give any pollution or wastes to the environment.

PO8: Ethics:

A standard method is adopted for implementing image processing. Firing is also done as per the Indian Army Ethics when someone unwantedly crosses the border.

PO9: Individual and team work

The project is completed by team work. Each members have their own individual contribution on fabrication, programming and designing.

PO10: **Communications**

Attended various reviews conducted internally and also presented in an international conference

PO11: Project management and finance

The project is finished on scheduled time period. The overall budget of the project is done below the estimated amount.

PO12: Lifelong learning

The researches can be done to increase the accuracy of the face detection and tracking in fast response manner. Future development is also applicable by implementing nano-second controller board.

PROGRAM SPECIFIC OUTCOMES

RELEVANT TO THE PROJECT

PSO1: Graduates will be able to create and analyses the Research and Development activities related to manufacturing

The image processing technique is implemented to manufacture an automatic sentinel gun. The researches can be processed in the area of programming path or accuracy in quick response rate.

PSO2: Graduates will be able to developed need based products in Mechanical Engineering and allied Industries

Though it is a defense project, the sentinel gun guarding idea will be given as a proposal to the DRDO with the support of the Indian Government.

REFERENCES

[1] Belaroussi.R, Milgram.M (2012), 'A comparitive study on face detection and tracking algorithm', Expert systems With Application, Vol 39, pp7158-7164.

[2] Bhattacharjee.A, Das.P (2015), 'A Real - Time Face Motion Based Approach towards Modeling Socially Assistive Wireless Robot Control with Voice Recognition', Internatinal Journal of Advanced Computer Science and Application, Vol 6, Issue 10, pp 205-219.

[3] Faux.F, Luther.F (2012), 'Theory of evidence for face detection and tracking', International Journals of Approximate Reasoning, Vol 53, Issue 5, pp 728-746.

[4] Hjelmas.E, Low.B.K (2001), ' Face Detection: A Survey ', Computer Vision and Image Understanding, Vol 83, pp 236-274.

[5] Kalas.M (2014), 'REAL TIME FACE DETECTION AND TRACKING USING OPENCV', International Journal of Soft Computing and Artificial Intelligence Vol 2, Issue1, pp 41-44.

[6] Kurka.P.R.G, Salazar.A.A.D (2018), 'Applications of image processing in robotics and instrumentation', Mechanical Systems and Signal Processing, Vol 124, pp 142-169.

[7] Li.J, Wang.T (2011), 'Face Detection using SURF Cascade', IEEE International Conference on Computer Vision Workshops, pp 2183-2190.

[8] Madhuram.M, Prithvi kumar.B (2018), 'Face detection and Recognition Using OpenCV', International Research Journal of Engineering and Technology(IRJET), Vol 5, Issue 10, pp 474-476.

[9] Renuka.M,Vimani.B (2017), 'Biometric Identification Using OpenCV Based On Arduino', International Research Journal of Engineering and Technology (IRJET), Vol 4, Issue 3, pp 1717-1722.

[10] Sarikan.S, Ozbayoglu.M (2018), 'Anomaly Detection in Vehicle Traffic with Image processing and Machine Learning', Complex Adaptive System Conference with Theme: Cyber Physical Systems and Deep Learning, Vol 140, Issue, pp 64-69.

[11] Sethi.N, Aggarwal.A (2011), 'Robust Face Detection and Tracking Using Pyramidal Lucas Kanade Tracker Algorithm', Nandita Sethi et al, Int. J.Comp.Tech. Appl., Vol 2, Issue 5, pp 1432-1438.

[12] Tripathy.R, Daschoudhury.RN (2013), 'Real-time Face Detection and Tracking using Haar Classifier on SoC', International Journal of Electronics and Computer Science Engineering, Vol 3, Issue 2, pp 175-184.

[13] Yan.J,Zhang.X (2013), 'Face detection by structural models', Image and Vision Computing, pp 10.

[14] Yang.G, Huang.T.S (1994), 'HUMAN FACE DETECTION IN A COMPLEX BACKGROUND', Pattern recognition, Vol 27, Issue 1, pp 53-63.

PHOTOGRAPH