Automated Video Surveillance System Using Python and a Mobile App

A Project Report

Submitted By

Rishi Verma (11BCE0377)
Satvik Dhandhania (11BCE0431)
Sahil Govel (11BCE0472)

in partial fulfillment for the award of the degree of

Bachelor of Technology

In

Computer Science and Engineering

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

VIT®
UNIVERSITY
(Estd. u/s 3 of UGC Act 1956)
VELLORE ▪ CHENNAI
www.vit.ac.in

May 2015

VIT UNIVERSITY
(Estd. u/s 3 of UGC Act 1956)
VELLORE ■ CHENNAI
www.vit.ac.in

School of Computing Science and Engineering

DECLARATION

We hereby declare that the project entitled **“Automated Video Surveillance System Using Python and a Mobile App”** submitted by us to the School of Computing Science and Engineering, VIT University, Vellore in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering** is a record of bonafide work carried out by us under the supervision of **Dr. Senthil Jayavel, Associate Professor and Asst. Director Academics (Systems).** I further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma of this institute or of any other institute or university.

Rishi Verma (11BCE0377)

Sahil Govel (11BCE0472)

Satvik Dhandhania(11BCE0431)

School of Computing Science and Engineering

CERTIFICATE

The project report entitled "**Automated Video** Surveillance **System Using Python and a Mobile App**" is prepared and submitted by **Rishi Verma (Register No: 11BCE0377), Sahil Govel (Register No: 11BCE0472)** and **Satvik Dhandhania (Register No: 11BCE0431)**.It has been found satisfactory in terms of scope, quality and presentation as partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering** in **VIT University, India.**

Internal Guide
Dr. Senthil Jayavel
Associate Professor

Internal Examiner **External Examiner**

ACKNOWLEDGEMENT

We are using this opportunity to express my gratitude to everyone who supported us throughout the course of this Engineering project. We are thankful for their aspiring guidance, invaluably constructive criticism and friendly advice during the project work. We are sincerely grateful to them for sharing their truthful and illuminating views on a number of issues related to the project. A special gratitude we give to our Project Guide, Dr. Senthil Jayavel, for his continuous support and guidance.

We would also like to express our gratitude to Prof. Hari Seetha (Program Chair, B.Tech CSE) and Prof. Venkateswaran S (Dean) who provided us with the facilities being required and conductive conditions for our Engineering project.
Finally, we would like to express our sincere thanks to Vellore Institute of Technology, which has supported us with this project.

Thank you,

Rishi Verma (11BCE0377)

Sahil Govel (11BCE0472)

Satvik Dhandhania (11BCE0431)

Table of Contents

List of Tables and Figures

List of Abbreviations

Abbreviation	Expansion
SaaS	Software-as-a-Service
PaaS	Platform-as-a-Service
GCM	Google cloud messaging service
HTTP(S)	Hyper Text Transfer Protocol (Secure)
JSON	JavaScript Object Notation
SMS	Short Message Service
GSM	Global System for Mobile Communications
ID	Identity Document
API	Application Programming Interface
REST	Representational State Transfer

Executive Summary

In the world today security and surveillance is a multibillion dollar industry. Video surveillance is increasing significance approach as organizations seek to safe guard physical and capital assets. At the same time, the necessity to observe more people, places, and things coupled with a desire to pull out more useful information from video data is motivating new demands for scalability, capabilities, and capacity. These demands are exceeding the facilities of traditional analog video surveillance approaches. Providentially, digital video surveillance solutions derived from different data mining techniques are providing new ways of collecting, analyzing, and recording colossal amounts of video data.

The scenes obtained from a surveillance video are usually with low resolution. Most of the scenes captured by a static camera are with minimal change of background. Objects in the outdoor surveillance are often detected in far field. Most existing digital video surveillance systems rely on human observers for detecting specific activities in a real-time video scene. However, there are limitations in the human capability to monitor simultaneous events in surveillance displays.

The project will provide a portable mobile app which can detect intrusion in your home or office by alerting activity of anyone in the video feed. This will give real-time notifications and will raise any red flags when there are anomalies in front of camera.

1. Introduction

1.1. Objective

The objective is to monitor the behavior, activities, or other changing information, usually of people for the purpose of influencing, managing, directing, or protecting them. In this study, we focus on detecting humans and do not consider recognition of their complex activities. Human detection is a difficult task from a machine vision perspective as it is influenced by a wide variety of factors and methods to store images in the computer. There are two main purposes to a video surveillance system. The first and most important one is to deter crime. The second is to help catch criminals when a crime has been committed.

1.2. Motivation

The primary purpose of this project is to free the owner of the extensive task of manually reviewing the video to identify the person who is present in front of the Video Surveillance System. Thus, the users of this system will be notified through the mobile app which will receive push notifications from the Video Surveillance system and display it to the owner. For each person it detects, it will give the name of the person if and only if he/she exists in the predefined database, otherwise it will pass the message that an Unknown person has been detected. Thus the user will be informed remotely about the people present in front of the surveillance system.

1.3. Background

For an intelligent video surveillance system, the detection of a human being is important for abnormal event detection, person identification and tracking etc. A system, which needs to be programmed according to the location it is to be deployed in, would require lots of initial overheads while installing it. This overhead includes enumeration of the kind of activities which would happen in this area and then coming up with a model which accurately captures routine activities and flags non routine ones. Clearly, this overhead is large and makes a programmed approach unsuitable for large-scale deployment. Hence there is a need for unsupervised video surveillance system, which is able to learn routine activities on its own data. A system with self-learning ability would be easy to deploy and would make it possible to have large scale monitoring.

- **Person tracking and identification:** A person in a visual surveillance system can be identified using face recognition and gait recognition techniques. The detection and tracking of multiple people in cluttered scenes at public places is difficult due to a partial or full occlusion problem for either a short or long period of time.

- **Abnormal event detection:** The most obvious application of detecting humans in a surveillance video is to early detect an event that is not normal. Abnormal events are classified as single-person loitering, multiple-person interactions (e.g. fighting and personal attacks) and person-facility/location interactions (e.g. object left behind and trespassing). Detecting sudden changes and motion variations in the points of interest and recognizing human action could be done by constructing a motion similarity matrix or adopting a probabilistic method. Methods based on probability statistics use the minimum change of time and space measure to model the method of probability.
- **Notifying the system administrator/owner:** Once an unknown person is detected, the intelligent system should send a message to the administrator that an unknown person has been detected or the name of the person from the list of safe people.

2. Project Description and Goals

There are two main purposes to a video surveillance system. The first and most important one is to deter crime. The second is to help catch criminals when a crime has been committed. Some of the ways a video surveillance system accomplishes both these goals is by doing the following:

1. Monitoring stores and stock.
2. Providing a visible presence or warning that video cameras are used in the store
3. Recording any intrusion.
4. Allowing people to monitor the cameras and see what is happening at any time of the day.
5. Providing an identification method, by which people can be screened before entering.
6. Allowing security personnel to check who is in a building at any time.

There are two major computing aspects to the system:

1. **The Viola/jones face detector**- This is a machine learning approach for visual object detection, which is capable of processing images extremely rapidly and achieving high detection rates.
2. **Android app**- This application is used to receive the push notifications from the Surveillance System remotely via the Google Cloud Messaging service.

3. Design Approach and Details

3.1.Design Approach

As developers of the project the first task is to identify the various components of the product. The product is primarily an open CV code which generates results of face detection and object tracking. We then send this as a message to the mobile app through push notifications.

<u>The Viola/Jones face detector</u>

This method is a machine learning approach for visual object detection, which is capable of processing images extremely rapidly and achieving high detection rates. This work is distinguished by three key contributions:

i. The first is the introduction of a new image representation called the “Integral Image” which allows the features used by our detector to be computed very quickly.
ii. The second is a learning algorithm, based on AdaBoost, which selects a small number of critical visual features from a larger set and yields extremely efficient classifiers.
iii. The third contribution is a method for combining increasingly more complex classifiers in a “cascade” which allows background regions of the image to be quickly discarded while spending more computation on promising object-like regions.

AdaBoost

In our system a variant of AdaBoost is used both to select a small set of features and train the classifier. In its original form, the AdaBoost learning algorithm is used to boost the classification performance of a simple (sometimes called weak) learning algorithm.

There are a number of formal guarantees provided by the AdaBoost learning procedure. Freund and Schapiro proved that the training error of the strong classifier approaches zero exponentially in the number of rounds. More importantly a number of results were later proved about generalization performance.

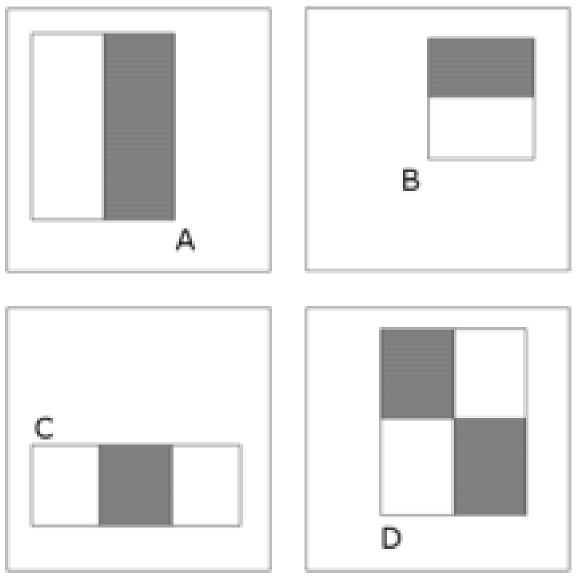
The key insight is that generalization performance is related to the margin of the examples and AdaBoost achieves large margins rapidly. Recall that there are over 180,000 rectangle features associated with each image sub-window, a number far larger than the number of pixels. Even though each feature can be computed very efficiently, computing the complete set is prohibitively expensive. Our method, which is borne out by experiment, is that a very small number of these features can be combined to form an effective classifier.

The main challenge is to find these features. In support of this goal, the weak learning algorithm is designed to select the single rectangle feature which best separates the positive and negative examples (this is similar to the approach in the domain of image database Retrieval). For each feature, the weak learner determines the optimal threshold classification function, such that the minimum number of examples is misclassified.

Boosting is a classification scheme that works by combining weak learners into a more accurate ensemble classifier

Weak learner: classifier with accuracy that need be only better than chance

We can define weak learners based on rectangle features:

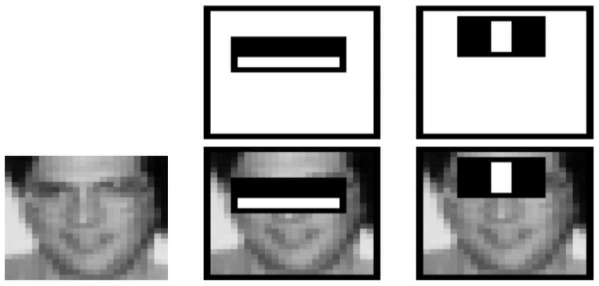


(It is similar to HAAR wavelets)

The process of boosting:
Initially, give equal weight to each training example

- Iterative training procedure
- Find best weak learner for current weighted training set
- Raise the weights of training examples misclassified by current weak learner

- Compute final classifier as linear combination of all weak learners (weight of each learner is related to its accuracy). Fig below shows how the process works, similar to HAAR classifiers.



Google Cloud Messaging

Google Cloud Messaging (GCM) is a free service that enables developers to send downstream messages (from servers to GCM-enabled client apps), and upstream messages (from the GCM-enabled client apps to servers). This could be a lightweight message telling the client app that there is new data to be fetched from the server (for instance, a "new email" notification informing the app that it is out of sync with the back end), or it could be a message containing up to 4kb of payload data (so apps like instant messaging can consume the message directly). The GCM service handles all aspects of queueing of messages and delivery to and from the target client app.

Architectural Overview

A GCM implementation includes a Google-provided connection server, a 3rd-party app server that interacts with the connection server, and a GCM-enabled client app. For example, this diagram shows GCM communicating with a client app on an Android device:

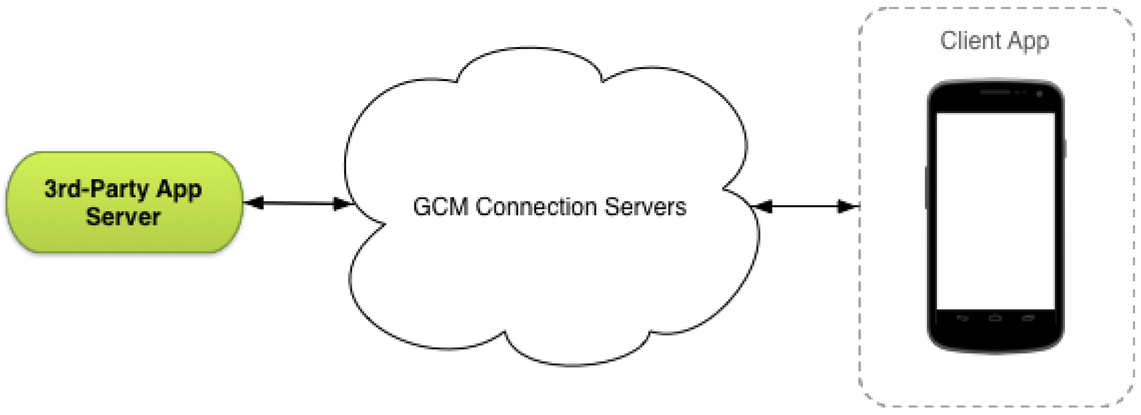


Figure 1: GCM Architecture

- Google-provided **GCM Connection Servers** take messages from a 3rd-party app Server and send these messages to a GCM-enabled client app (the "client app"). Currently Google provides connection servers for HTTP and XMPP.
- The **3rd-Party App Server** is a component that we implement to work with our chosen GCM connection servers. App servers send messages to a GCM connection server, the connection server enqueues and stores the message, and then sends it to the client app.
- The **Client App** is a GCM-enabled client app. To receive GCM messages, this app must register with GCM and get a registration ID. If we are using the XMPP (CCS) Connection server, the client app can send "upstream" messages back to the 3rd-party app server also.

Lifecycle Flow

Client Side (Mobile Application)

- **Register to enable GCM**. A client app registers to receive messages.
- **Send and receive downstream messages.**
    1. **Receive a message**. A client app receives a message from a GCM server.
    2. **Send and receive upstream messages.** This feature is only available if you're using the XMPP Cloud Connection Server (CCS).
        - **Send a message**. A client app sends messages to the 3rd-party app server:

1. On the device, the client app sends messages to XMPP (CCS).See your platform-specific documentation for details on how a client app can send a message to XMPP (CCS).
2. XMPP (CCS) enqueues and stores the message if the server is disconnected.
3. When the 3rd-party app server is re-connected, XMPP (CCS) sends the message to the 3rd-party app server.

3rd Party Server

Sending a message to the client application

1. The 3rd-party app server sends a message to GCM connection servers.
2. The GCM connection server enqueues and stores the message if the Device is offline.
3. When the device is online, the GCM connection server sends the Message to the device.
4. On the device, the client app receives the message according to the platform.

Receive a message. A 3rd-party app server receives a message from XMPP (CCS) and then does the following:

1. Parses the message header to verify client app sender information.
2. Sends "ack" to GCM XMPP connection server to acknowledge receiving the message.
3. Optionally parses the message payload, as defined by the client app.

4. Technical Specifications

While the overall goals, strategies and objectives have been stated, the specifications of the components will be determined as they are identified for their applicability in the project. The overall system will meet the specifications stated in Table 4.1.

Components	
GCM Connection Servers	The Google-provided servers involved in sending messages between the 3rd-party app server and the client app.
Client App	A GCM-enabled client app that communicates with a 3rd-party app server.

3rd-party App Server	An app server that you write as part of implementing GCM. The 3rd-party app server sends data to a client app via the GCM connection server.
Credentials	
Sender ID	A project number you acquire from the API console. The sender ID is used in the registration process to identify a 3rd-party app server that is permitted to send messages to the client app.
Sender Auth Token	An API key that is saved on the 3rd-party app server that gives the app server authorized access to Google services. The API key is included in the header of POST requests.
Application ID	The client app that is registering to receive messages. How this is implemented is platform-dependent. For example, an Android app is identified by the package name from the manifest. This ensures that the messages are targeted to the correct Android app.
Registration ID	An ID issued by the GCM servers to the client app that allows it to receive messages. Note that registration IDs must be kept secret.

Table 1 GCM Specifications

4.1.Downstream HTTP or XMPP messages (JSON)

The following table lists the targets, options, and payload for HTTP or XMPP JSON messages.

Parameter	Protocol	Usage	Description
Targets			
To	XMPP	Required, string	This parameter specifies the recipient of a message. The value must be a registration ID /notification key. This parameter is used in XMPP in place of registration_ids or notification_key in HTTP.

registration_ids	HTTP	Required if notification_ key not present, string array	This parameter specifies the list of devices (registration IDs) receiving the message. It must contain at least 1 and at most 1000 registration IDs. Multicast messages (sending to more than 1 registration IDs) are allowed using HTTP JSON format only. This parameter or notification_key is used in HTTP in place of to in XMPP.
notification_key	HTTP	Required if registration_ids not present, string	This parameter specifies the mapping of a single user to multiple registration IDs associated with that user. This allows a 3rd-party app server to send a single message to multiple app instances (typically on multiple devices) owned by a single user. A 3rd-party app server can use notification_key as the target for a message instead of an individual registration ID (or array of registration IDs). The maximum number of members allowed for a notification_key is 20.This parameter or registration_ids is used in HTTP in place of to in XMPP.
Options			
message_id	XMPP	Required, string	This parameter uniquely identifies a message in an XMPP connection.

collapse_key	HTTP, XMPP	Optional, string	This parameters identifies a group of messages (e.g., with collapse_key: "Updates Available") that can be collapsed, so that only the last message gets sent when delivery can be resumed. This is intended to avoid sending too many of the same messages when the device comes back online or becomes active. Note that there is no guarantee of the order in which messages get sent. Messages with collapse key are also called send-to-sync messages messages. Note: A maximum of 4 different collapse keys is allowed at any given time. This means a GCM connection server can simultaneously store 4 different send-to-sync messages per client app. If you exceed this number, there is no guarantee which 4 Collapse keys the GCM connection server will keep.
delay_while_idle	HTTP, XMPP	Optional, JSON Boolean	When this parameter is set to true, it indicates that the message should not be sent until the device becomes active. The default value is false.
time_to_live	HTTP, XMPP	Optional, JSON number	This parameter specifies how long (in seconds) the message should be kept in GCM storage if the device is offline. The maximum time to live supported is 4 weeks. The default value is 4 weeks.

delivery_receipt_ requested	XMPP	Optional, JSON boolean	This parameter lets 3rd-party app server request confirmation of message delivery. When this parameter is set to true, CCS sends a delivery receipt when the device confirms that it received the message. The default value is false.
restricted_package_ name	HTTP	Optional, string	This parameter specifies the package name of the application where the registration IDs must match in order to receive the message.
dry_run	HTTP	Optional, JSON boolean	This parameter, when set to true, allows developers to test a request without actually sending a message. The default value is false.
Payload			
data	HTTP, XMPP	Optional, JSON object	This parameter specifies the key- value pairs of the message's payload. There is no limit on the number of key-value pairs, but there is a total message size limit of 4kb. For instance, in Android, data:{"score":"3x1"} would result in an intent extra named score with the string value 3x1. The key should not be a reserved word (from or any word starting with google). It is also not recommended to use words defined in this table (such as collapse_key) because that could yield unpredictable outcomes. Values in string types are recommended. You have to convert values in objects or other non-string data types (e.g., integers or booleans) to string.

Table 2 Attributes of HTTP or XMPP messages(JSON)

4.2.Codes and Standards

1. UMTS, WCDMA - HSPA/HSPA+: 3G data standards
2. USB: Needed to interface the the smartphone to the computer
3. IEEE 802.11: WiFi standard
4. RFC 2616: HTTP/1.1 standard
5. UHF radio waves in the ISM band from 2.4 to 2.485 GHz
6. NIEM 3.0 for XML encoding of the standard. Currently NIEM 2.1 is used for ANSI/NIST-ITL schema.

4.3. Constraints

The following have been identified as the constraints of the product:

1. **Economic Constraint:** The system will require the deployment of the video surveillance system at every entrance.
2. **Social Constraint**: The usage of such a system can be deemed as violation of privacy in public areas.
3. **Sustainability Constraint:** The usage of this system has to be sustained for large campuses and needs to be scaled accordingly; the sustainability of such an expansive system needs to be tested.
4. **Logistical Constraint:** The system cannot take large amount of data's and it requires active internet connection all the time.
5. **Environment Constraint:** The system may not function properly in certain environments like in dim lights or weak internet connection zones.
6. **Delivery Constraint:** The application completely depends on the Google Cloud Messaging platform which may add delay to send the notifications to users since it is a free and open service.

4.4. Tradeoffs

Integrity vs Ease of Use:
The administering of both OpenCV and android push notification makes the entire process slower, and harder to use. However, using both preserves the integrity of the system and ensures perfection. This helps the system to be more efficient at the cost of easiness.
Smartphone vs GSM:
The use of smartphone app makes the entire system very cost effective and the fact that everyone can now afford a smartphone. The other alternatives were GSM Smart Messaging System but that would give a user messages that can sometimes be a false alarm. Also, the cost of the system would increase as each SMS would be charged by the carrier.

5. Schedule, Tasks and Milestones

There are several major milestones as well as sever smaller tasks that must be achieved in order to reach the milestones. Tasks will be split up among group members according to each member's level of expertise or comfort. Each task will have a leader who is responsible for completion of that task. However, the other group members are expected to provide assistance if needed. This way, an engineer can concentrate on a task but still get help if needed. Also, each tasks respective leader is held accountable if the task fails. A Gantt chart outlining important tasks and goals can be seen in the Appendix.

The milestones are:

- Researching for algorithms and theories relating to computer vision
- Installing and setting up all the OpenCV libraries and functions.
- Designing the mobile App on Android Studio.
- Development of program that detects the face in a feed.
- Development of the 3rd Party Application Server on the Google App Engine
- Interlinking of the 3rd Party Application Server to the Google Cloud Messaging Server
- Configuring the Device to be able to receive messages and display them in correct format.

6. Market and Cost Analysis

6.1. Market Analysis

The existing method used in most places is limited to a person who can see footage of camera to do the surveillance. This method is limited and susceptible to human error. This method also varies from place to place and is not very efficient and requires crowd sourcing.

Our product is not susceptible to manual error since the face detection is completely autonomous this helps the system to be efficient and fast. It is easy to use and portable and the application created specifically for the system can be used by any person with an android phone.

6.2. Cost Analysis

The cost analysis for the entire system boils down to the following part costs. The part costs are as described in the following table. The system will work with any smartphone with an active internet connection. However that will not be factored into the cost analysis because it is a common entity of use.

The online infrastructure cost for any institution should be nil as all open source resources have been used. When used in large contexts, in order to scale to serve more client endpoints then the online infrastructure must be expanded in a pay-per-usage manner which has also been included below:

Entity	**Cost**
Basic smart phones	50$
Google Developers Console- One time Registration fee	25$
Online infrastructure cost Google App Engine (Only for usage past quota)	0-50(15$ - DBaaS + 35$ - PaaS per month)

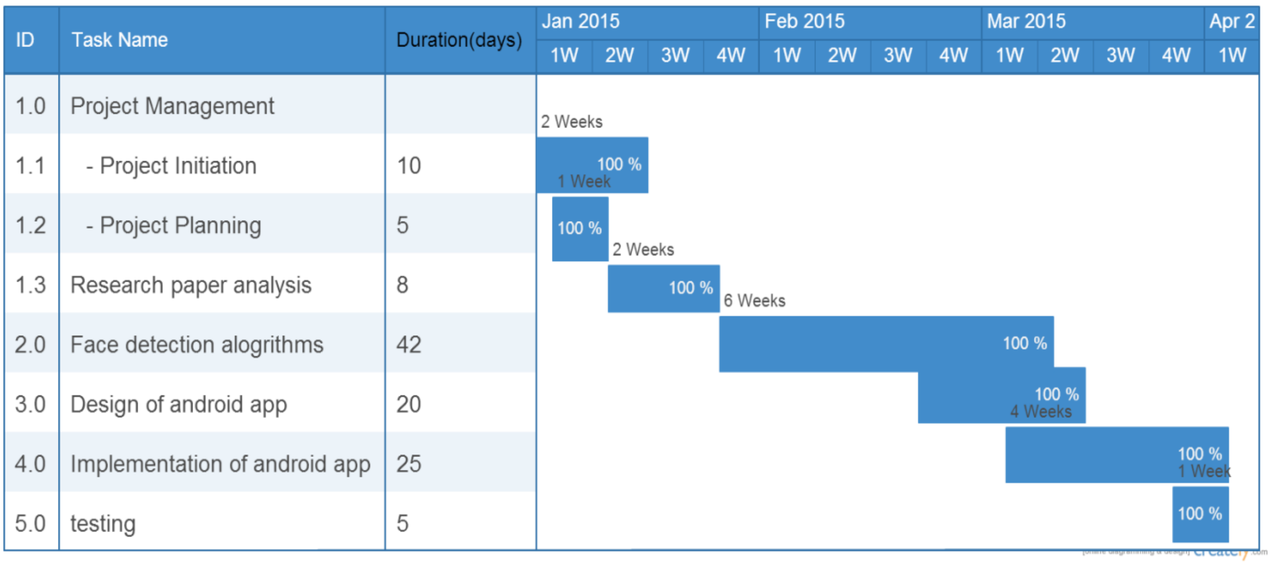
Table 3 Cost Analysis

7. Summary

8. References

1. M.Hunke and A. Waibel, “Face Locating And Tracking for Human Computer Interaction”,in Proc. Conf. Signals Systems and Computers, Nov 1994, vol2 pp 1277- 1281.
2. Abbas El Gamal, “Introduction To Statistical Signal Processing”, Lecture Notes.
3. D. Cahi and K.N. Ngan, “Face segmentation using skin color map”, IEEE Trans. On Circuit and Systems for Video Technology, vol 9, no 4, pp. 551-564, Jun. 1999.
4. io.js JavaScript Runtime and Documentation: https://iojs.org/en/index.html.
5. Sirovich, L. and Kirby, M.: 1987, Low dimensional procedure for the characterization of human faces, J. Opt. Soc. Am. A 4, 519–524.

Appendix A- Gantt Chart



ID	Task Name	Duration(days)
1.0	Project Management	
1.1	- Project Initiation	10
1.2	- Project Planning	5
1.3	Research paper analysis	8
2.0	Face detection alogrithms	42
3.0	Design of android app	20
4.0	Implementation of android app	25
5.0	testing	5