Real Time Vehicle Security System Through Face Recognition

By

Capt Syed Ahsan Izhar
Capt Waqas Iftikhar
Capt Syed Hasnain Kabir

Submitted to the Faculty of Department of Electrical Engineering,

Military College of Signals, National University of Science and Technology,

Rawalpindi in fulfillment for the requirements of a B.E Degree in

Electrical (Telecom) Engineering

JUNE 2016

ABSTRACT

The issue of vehicle theft have been an unending challenge even with recent advancement in car security systems such as the use of immobilizers, tracking systems and modified alarm systems which have in recent time occasionally constituted nuisance in the environment being triggered even in the event of no safety issue, however irrespective of these advancement the issue of car crime has not ended. This project strives in offering an advance car security system that uses the results of a facial recognition in gaining an ignition response. The project focuses on the utilization of the face image processing in restricting access.

CERTIFICATE

It is hereby certified that the contents and form of the project report entitled "**Real Time Vehicle Security System Through Face Recognition"** submitted by 1) Captain Syed Ahsan Izhar 2) Captain Waqas Iftikhar 3) Captain Syed Hasnain Kabir have been found satisfactory as per the requirement of the B.E. Degree in Electrical (Telecom) Engineering.

Supervisor:

Dr Lt Col Adnan Rashidi

MCS, NUST

____

DECLARATION

We hereby declare that no content of work presented in this report has been submitted in support of another award of qualification or degree either in this institution or anywhere else.

DEDICATION

Our Parents and honorable faculty without whom

we would have not achieved this.

TABLE OF CONTENTS

LIST OF FIGURES

APPENDICES

1. Introduction

a. Overview

Increase in car theft has always been an issue not just in undeveloped countries, but also in developed world irrespective of several types of security systems put in place to control it. Criminal masterminds adopt numerous strategies to beat most common car security application and systems. According to the Stolen Motor Vehicle (SMV) database the total number of cars stolen as at the end of December 2011 was about 7.1 million worldwide; which is more than the population of some countries. Several security measures have been developed over the years in response to this issue, most popular of such security system is the alarm system which turns out to be inept.

This project uses an image processing technique in the design of a car security system, using face recognition as a tool to gain access and control in a car system. The Car is programmed to recognize its owner and grant access

b. Problem Statement

Access to a vehicle is usually via a key system of which in most security conscious design is attached to an alarm system; this has not ended the issue of car theft as car thieves have developed strategies to override the ignition system. The use of image processing based software approach in identifying who should have access or not while starting a car is not popularly available. This is why a thief can get away in split seconds if he or she can turn on the car via the ignition. Also relying on the alarm system has not been adequate as making it too sensitive constitute nuisance even when there is no attempt to steal the car and if tuned with a lower sensitivity the alarm might not even come on in the event of an earthquake. The image processing based security system is designed not only to minimize car theft but also to retrieve the facial identity of Vehicle thieves; this would help law enforcement agencies to remove the car thieves off the streets as they are responsible for all this security measures in the first place.

c. Approach

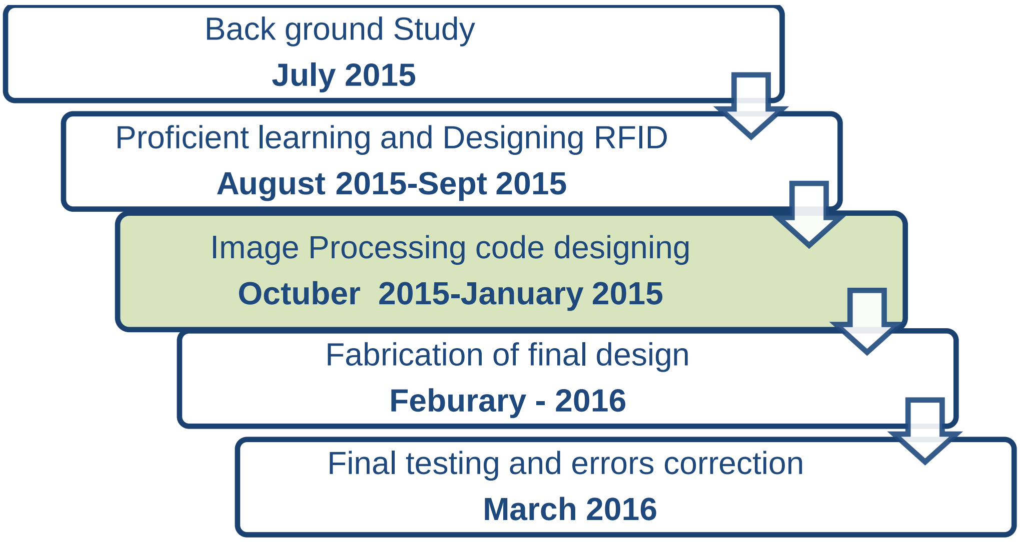
Our proposed device is a system application that, with the help of a digital photo or video, has the ability of identifying or authenticating a person. An automated thinking device which would only allow an authorized person to start the car. Camera is incorporated in a secure place to take and process the image. Through face recognition system will be able to differentiate between the authorized or unauthorized person and identify the right person for use of vehicle. Ignition of car would be entirely dependent on the results of the application system. Another feature, Radio Frequency identification, is also introduced in the car that will increase the easiness of the user.

d. Objectives

The objectives are

- To choose an already existing facial recognition technique by considering and analyzing its real life practical implementation and feasibility.
- To embed the facial recognition software program into a hardware for a real-time practical demonstration.
- Design a car security system that grants access via facial recognition.

e. Organization



2. Background Study

Image processing is a form of signal processing which utilizes an image input, in this context refers to the processing of digital images. The processing of digital images in this context refers to what we do with the input image such as filtering, resizing, and colour extraction. The aim of a security system is to safeguard a person or object by access restriction, satisfying the requirements for developing quality access restriction provides an efficient security system.

a. Face Recognition

Face recognition is an important ability possessed by humans; even an infant responds to face shapes after birth and can discriminate his or her mother's face from a stranger at a tender age of 45 hours (Voth 2003). This is not the case in machine recognition, getting a machine to recognise human faces or objects brings to light machine learning techniques which is a broad field of artificial intelligence, artificial intelligence aims at mimicking intelligent abilities of humans by machines(Rätsch 2004). The machine learning algorithms to differentiate within exemplars are developed to enable machines differentiate objects, objects are seen as patterns in machine learning.

Face Detection

Feature Extraction

Face Recognition

Identification/Verification

Input Image/Video

Face recognition system can be divided into three main steps in achieving its purpose, face detection, feature extraction, and face recognition

b. Face detection

Face detection is the discovery of faces contained in an image by a machine, it involves algorithms used in identifying the sub-region of an image containing faces and aids in the alignment of the face image. There are various face detection algorithms some of which include the Intel computer vision library which contains extended realization of viola-jones object detection algorithm, Face detection library(FDLib) developed by Keinzle et al. (Degtyarev and Seredin 2010) and Face detection algorithm developed in University of Surrey known as the Algorithm UniS. However this project focuses on OpenCV extended Viola jones algorithm as it is the most popular, free algorithm and is of relatively very good performance.

(1) Viola-Jones

The machine learning algorithm utilized in the face detection is the

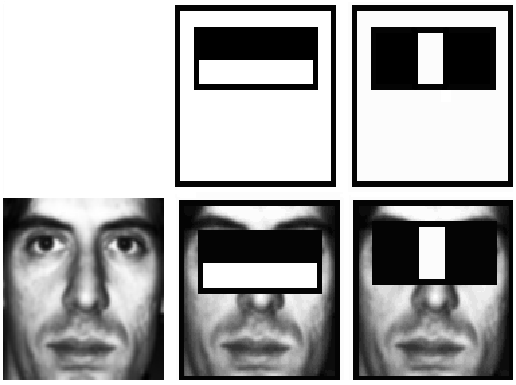


Figure 1 - Viola Jones

viola-jones method implemented in MATLAB for the purpose of design and also implemented using the free computer vision C\C++ library Open CV for the purpose of embedding in hardware.

The **Viola–Jones object detection framework** is the first object detection framework to provide competitive object detection rates in real-time proposed in 2001 by Paul Viola and Michael Jones. The viola jones algorithm uses rectangular features known as the Haar-like features in detecting a faces. Each haar-like feature has a white and dark side in the rectangle. The characteristics of Viola–Jones algorithm which make it a good detection algorithm are:

- Robust – very high detection rate (true-positive rate) & very low false-positive rate always.
- Real time – For practical applications at least 2 frames per second must be processed.
- Face detection only (not recognition) - The goal is to distinguish faces from non-faces (detection is the first step in the recognition process).

(2) Principle component analysis

The principal component analysis

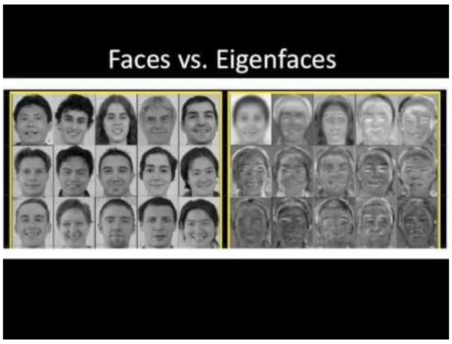


Figure 2 - PCA

(PCA) is used in the face recognition which presents a quick and easy approach in dimensionality reduction and pattern matching.

The goal of **PCA** is dimensionality reduction; it is a data-reduction method that re-expresses a raw data with an alternative set of parameters such that the noise and redundancy of the data is kept minimal. It uses an orthogonal transformation to convert a set of input images (Faces) into a smaller number of Eigenvectors of these faces known as Eigen faces (Turk and Pentland 1991).

The data noise and redundancy is expressed by a covariance matrix such that an image with m pixels can be re-expressed as having m rows, one column matrix (Zhang et al 2012).

i. Steps in PCA

a. Convert image of training sets into image vectors
b. Normalize the face vectors
c. Calculate the eigen vectors
d. Reduce dimensionality
e. Represent each face image a linear combination of all eigen vectors

c. Radio Frequency Identification

Radio-frequency identification (**RFID**) is the wireless use of electromagnetic fields to transfer data, for the purposes of automatically identifying and tracking tags

Figure 3 - RFID

attached to objects. The tags contain electronically stored information. Some tags are powered by electromagnetic induction from magnetic fields produced near the reader. Some types collect energy from the interrogating radio waves and act as a passive transponder. RFID, or Radio Frequency Identification, is a system for transferring data over short distances (typically less than 6 inches). Often only one of the two devices needs to be powered, while the other is a passive device. This allows for easy use in such things as credit cards, key fobs, and pet collars as there is no need to worry about battery life. The downside is that the reader and the information holder (ie credit card) must be very close, and can only hold small amounts of data.

3. Schematic Diagram

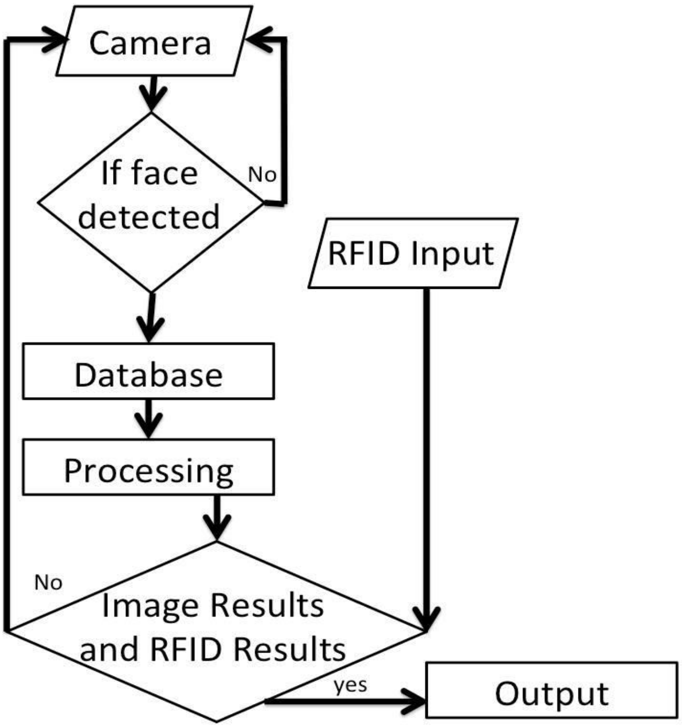


Figure 4 - Flow Chart

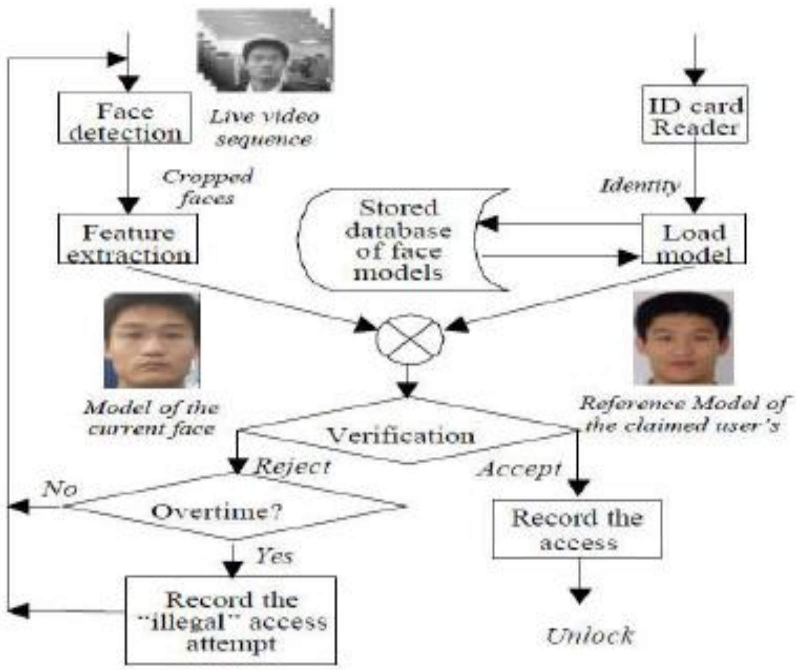


Figure 5 - Schematic Model

4. Design Specificaions

a. Arduino UNO R3

- ATmega328 microcontroller
- Input voltage - 7-12V
- 14 Digital I/O Pins (6 PWM outputs)
- 6 Analog Inputs
- 32k Flash Memory
- 16Mhz Clock Speed

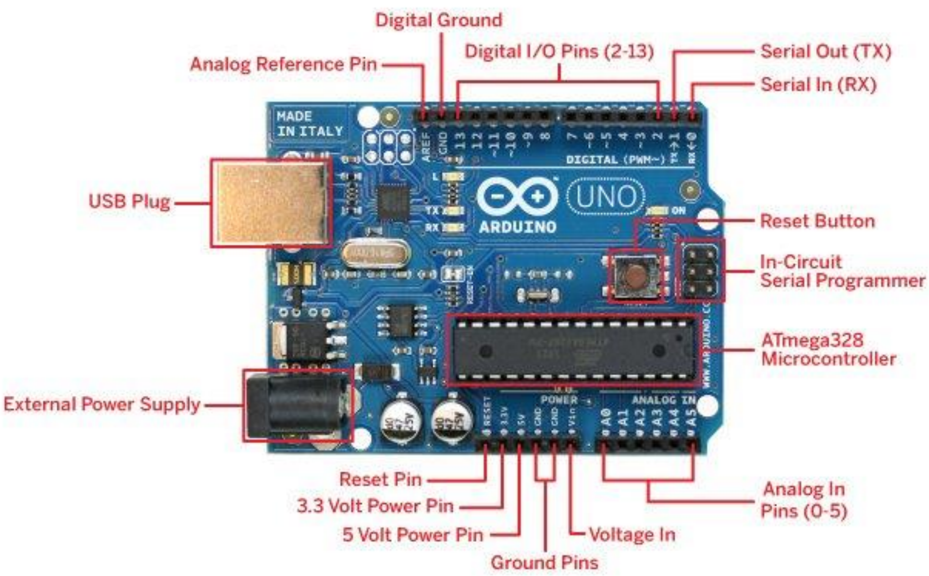


Figure 6 - Arduino Uno Pinout

b. RFID- RC522

- Working current：13—26mA/ DC 3.3V
- Standby current：10-13mA/DC 3.3V
- Sleeping current：<80uA

Figure 7 - RC522

- Peak current：<30mA
- Working frequency：13.56MHz
- Card reading distance ：0～60mm（mifare1 card）
- Protocol：SPI
- Data communication speed： Maximum 10Mbit/s
- Dimension：40mm×60mm
- Environment
- Working temperature：-20—80 degree
- Storage temperature：-40—85 degree
- Humidity： relevant humidity 5%—95%
- Max SPI speed: 10Mbit/s

c. Misc Items

Further required items are as following:

- 2x16 LCD
- Battery
- Laptop
- Relay

5. Design Implementation

a. How this system works?

This system strives in offering an advance car security system that uses the results of a facial recognition in gaining an ignition response. The project focuses on the utilization of the face image processing in restricting access.

(1). System Architecture

- Face detection and Acquisition
- Convert RGB to Gray
- Matching with the data base
- Result passed to the hardware
- Access granted or not

(2). Face detection and Acquisition

Image is acquired through the camera. Once the date base is created, it is continuously comparing the live image with the date base.

(3). Convert RGB to Gray

Image acquired by a common cameras are in RGB color model by default. RGB stands for Red Green Blue. In this model each of the primary color i.e. Red Green Blue valued form 0 to 255 this means combination of intensities of these three colors identifies different unique colors e.g. if intensities of these three colors in RGB model is 255, 255 and 255 respectively, the visual effect will be white. Also if the intensities are 255, 0, 0 the result will be Red etc.

We can process our image using RGB model but this would result in high processing time and heavy load on processor as it not only processing a single matrix but three different matrices. So to simplify and improve the efficiency of our processing we convert the acquired image from RGB to Gray scale. Gray scale image commonly called black and white image. In this we have only one matrix to cater. Each value of the pixel of the image varies from 0 to 255. These values are the intensity value of the pixel. Zero is the darkest value and 255 is the brightest value.

(4). Matching with the data base

The captured image is compared with the saved data base images. Image is compared with the lowest eigen distance among all the saved images of a same person.

(5). Result passed to the hardware

Result is passed to the Hardware which will analyze the result on the basis of comparison.

b. Configuring Arduino UNO with RFID

On the Arduino many of the pins are not swappable. Because this device uses the SPI bus, who's pins cannot be moved around, pins 11, 12, 13 must remain as shown. RST and IRQ are user specified.

This device is NOT a 5 volt powered device. You MUST power it with 3.3 volts. If you do not, you risk overheating the RFID. Most Arduino boards include a 3.3V supply pin which can be used to power the RFID module. If 3.3 volts is not accessible, there are LD33V regulators available at Addicore.com that supply 3.3 volts.

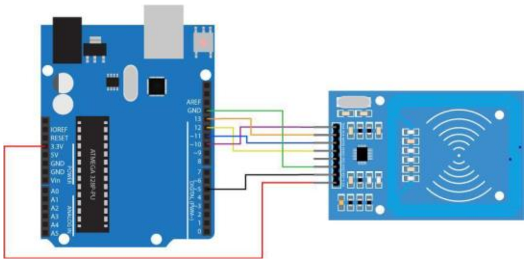


Figure 8 - UNO with RFID

RFID-RC522 Module	Arduino Uno
1 - SDA	Digital 10
2 - SCK	Digital 13
3 - MOSI	Digital 11
4 - MISO	Digital 12
5 - IRQ	--unconnected--
6 - GND	Gnd
7 - RST	Digital 5
8 - 3.3V	3.3v

c. Program the Arduino

After installing the AddicoreRFID library in the steps above the library will be available to use in sketches but any example sketches included with the library will not be shown in *File* > *Examples* until after restarting the Arduino IDE.

1. Restart the Arduino IDE by closing all currently open Arduino IDE windows and then opening the Arduino IDE.

2. When the Arduino IDE opens navigate to *File* > *Examples* > *AddicoreRFID*. Select the “Addicore_RFID_Example” sketch. This will open a sketch which we will use with the RFID module that we wired to our Arduino earlier.

A second option instead of opening the example sketch is to copy the code below into a new sketch.

```
// Example sketch to read the ID from an Addicore 13.56MHz RFID tag

// as found in the RFID AddiKit found at:

// http://www.addicore.com/RFID-AddiKit-with-RC522-MIFARE-Module-RFID-Cards-p/126.htm

#include <AddicoreRFID.h>

#include <SPI.h>

#define uchar unsigned char
```

#define uint unsigned int

//4 bytes tag serial number, the first 5 bytes for the checksum byte

uchar serNumA[5];

 uchar fifobytes;

uchar fifoValue;

AddicoreRFID myRFID; // create AddicoreRFID object to control the RFID module

//

//set the pins

//

const int chipSelectPin = 10;

const int NRSTPD = 5;

//Maximum length of the array

#define MAX_LEN 16

void setup() {

Serial.begin(9600); // RFID reader SOUT pin connected to Serial RX pin at 9600bps

```
// start the SPI library:

SPI.begin();

pinMode(chipSelectPin,OUTPUT); // Set digital pin 10 as OUTPUT to connect it to the RFID /ENABLE pin

digitalWrite(chipSelectPin, LOW); // Activate the RFID reader

pinMode(NRSTPD,OUTPUT); // Set digital pin 10 , Not Reset and Power-down

digitalWrite(NRSTPD, HIGH);

myRFID.AddicoreRFID_Init();

}

void loop()

{

uchar i, tmp, checksum1;

uchar status;

uchar str[MAX_LEN];

uchar RC_size;

uchar blockAddr; //Selection operation block address 0 to 63

String mynum = "";

str[1] = 0x4400;
```

```
//Find tags, return tag type

status = myRFID.AddicoreRFID_Request(PICC_REQIDL, str);

if (status == MI_OK)

{

Serial.println("RFID tag detected");

Serial.print(str[0],BIN);

Serial.print(" , ");

Serial.print(str[1],BIN);

Serial.println(" ");

}
```



```
//Anti-collision, return tag serial number 4 bytes

status = myRFID.AddicoreRFID_Anticoll(str);

if (status == MI_OK)

{

checksum1 = str[0] ^ str[1] ^ str[2] ^ str[3];

Serial.println("The tag's number is : ");

//Serial.print(2);
```

```
Serial.print(str[0]);

Serial.print(" , ");

Serial.print(str[1],BIN);

Serial.print(" , ");

Serial.print(str[2],BIN);

Serial.print(" , ");

Serial.print(str[3],BIN);

Serial.print(" , ");

Serial.print(str[4],BIN);

Serial.print(" , ");

Serial.println(checksum1,BIN);

// Should really check all pairs, but for now we'll just use the first

if(str[0] == 156) //You can change this to the first byte of your tag
by finding the card's ID through the Serial Monitor

{

Serial.print("Hello Craig!\n");

} else if(str[0] == 244) { //You can change this to the first byte of
your tag by finding the card's ID through the Serial Monitor

Serial.print("Hello Erin!\n");
```

```
}
Serial.println();
delay(1000);
}
myRFID.AddicoreRFID_Halt(); //Command tag into hibernation
```

3. Now connect your Arduino to your computer and upload the code.

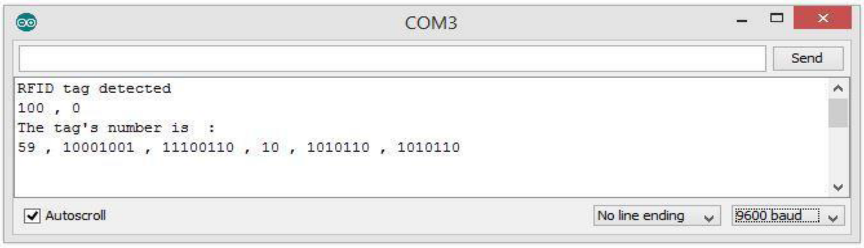
d. Reading RFID Tags

With your Arduino programmed with the “Addicore_RFID_Example” sketch and with it still plugged into your computer open the serial monitor by clicking the Serial Monitor icon or pressing *CTRL+SHIFT+M*.

If it isn’t already selected set the serial baud rate to 9600 baud by clicking on the drop down selection in the lower right corner of the Serial Monitor window as shown below and selecting 9600 baud.

Now take one of the RFID cards or fobs that came in your RFID AddiKit and hold it near the white graphic printed on your RFID-RC522 module as shown below:

Once the module has read the RFID tag your Serial Monitor should show something similar to the following:

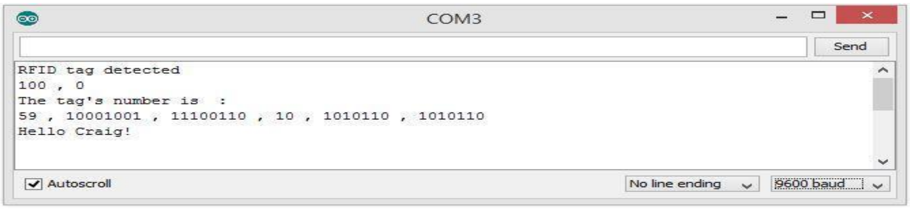


When the RFID reader senses a card, it starts by printing “RFID tag detected” on the Serial Monitor. The next line verifies the card type. In normal use, this would be unnecessary. The next two lines print out the data stored in the card, in this case an ID identifying the specific tag scanned.

Note the number 59 from the screenshot above. This is the first byte of the scanned tag’s ID. The tag you scan will have a different ID than the one used in this example. In the case of this example I will write down 59 but you will need to write down the number preceding the first comma of the tag’s ID that your Serial Monitor shows. Close the Serial Monitor and find in the Addicore_RFID_Example code the following lines of code:

```
// Should really check all pairs, but for now we'll just use the first
if(str[0] == 156)                    //You can change this to the first byte of your tag
{
    Serial.print("Hello Craig!\n");
} else if(str[0] == 244) {           //You can change this to the first byte of your tag
    Serial.print("Hello Erin!\n");
}
```

Upload your changed code to your Arduino, reopen the Serial Monitor, and then scan the same RFID tag as before. Now when you scan the tag the Arduino will recognize the ID and will display the desired text. Below is an example of what mine looks like:



e. Image Processing

The microcomputer which contains the image processing unit embedded within it performs the face detection and authorizes the person. The processing of image involves two parts, face detection and face recognition.

(1) Face Detection

The acquired image is processed to detect the face using the Viola Jones algorithm (Viola and Jones, 2002) which effectively uses the cascade object detection. Nandakumar et al cascade detector detects the face of the acquired image and the face region is extracted. The authentication based security system has the database which stores the face images of the authorized persons under different environments. The face images are enhanced by normalizing them to remove the unwanted information due to illumination constraints while acquiring the image and are stored in the database. Now the task of face recognition must be performed with the detected faces.

(2) Face Recognition

Face recognition can be performed with various algorithms which are feature based or model based.

Mostly feature based algorithms are used in the security systems involved in real time. Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) algorithms are efficient in terms of extracting the features to perform recognition. Both algorithms are compared (S K Hese and M R Banwaskar, 2013) and they found to have similar features but Linear Discriminant Analysis (LDA) outperforms Principal Component Analysis (PCA) algorithm when large training sets are involved in recognition. Also, LDA discriminates most of the information present in the image efficiently by computing the intra class and inter class scatter matrices. Using the database which contains normalized face images, the recognition is performed in the vehicle security system through the LDA algorithm. LDA performs the feature extraction of the stored images in the database which are called the training images and the camera acquired face image which is called the test image. The test image is to be compared with the database images and the classifier used in the algorithm decides the image as known or unknown using the Euclidean distance and the threshold value. The Euclidean distance is calculated between the corresponding weights of features and the image which produces minimum distance is best matched with the test image. The person is classified as known or authorized when the Euclidean distance is smaller than the threshold value and the person is classified as unknown or unauthorized when the distance value exceeds the threshold.

f. Principle Component Analysis

(1) Procedure explanation

Principal component analysis (PCA) involves a mathematical procedure that transforms a number of (possibly) correlated variables into a (smaller) number of uncorrelated variables called *principal components*. The first principal component accounts for as much of the variability in the data as possible, and each succeeding component accounts for as much of the remaining variability as possible.

(2) Objectives of principal component analysis

- To discover or to reduce the dimensionality of the data set.
- To identify new meaningful underlying variables.

(3) How to start

We assume that the multi-dimensional data have been collected in a TableOfReal data matrix, in which the rows are associated with the cases and the columns with the variables. The Table Of Real is therefore interpreted as *number Of Rows* data vectors, each data vector has *number of Columns* elements.

Traditionally, principal component analysis is performed on the Covariance matrix or on the Correlation matrix. These matrices can be calculated from the data matrix. The covariance matrix contains scaled sums of squares and cross products. A correlation matrix is like a covariance matrix but first the variables, i.e. the columns, have been standardized. We will have to standardize the data first if the variances of variables differ much, or if the units of measurement of the variables differ. You can standardize the data in the Table Of Real by choosing Standardize columns.

To perform the analysis, we select the Tabel Of Real data matrix in the list of objects and choose To PCA. This will result in a new PCA object in the list of objects.

(4) Mathematical background on principal component analysis

The mathematical technique used in PCA is called eigen analysis: we solve for the eigen values and eigenvectors of a square symmetric matrix with sums of squares and cross products. The eigenvector associated with the largest eigen value has the same direction as the first principal component. The eigenvector associated with the second largest eigen value determines the direction of the second principal component. The sum of the eigen values equals the trace of the square matrix and the maximum number of eigenvectors equals the number of rows (or columns) of this matrix.

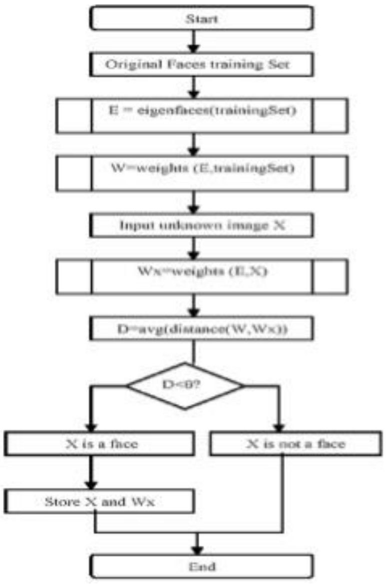


Figure 9 - Eigen Face base Recognition Algortihm

(5) Eigenfaces for Face Detection/Recognition

Computation of the eigenfaces

<u>Step 1:</u>

Obtain face images I1 , I2 , ..., IM (training faces) (very important: the face images must be centered and of the same size)

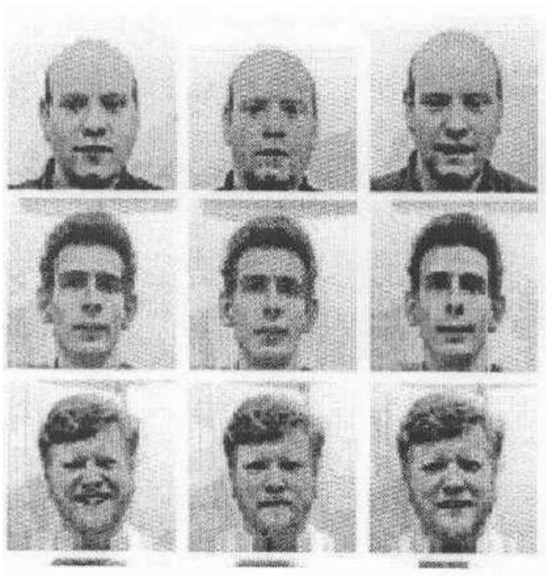


Figure 10 - Training Faces

<u>Step 2:</u>

Represent every image Ii as a vector Γi

<u>Step 3:</u>

Compute the average face vector Ψ: Ψ = 1 M M i=1 Σ Γi

<u>Step 4:</u>

Subtract the mean face: Φi = Γi − Ψ

<u>Step 5:</u>

Compute the covariance matrix C: C = 1 M M n=1 Σ ΦnΦ T n = AAT (N 2 xN 2 matrix) where A = [Φ1 Φ2 ... ΦM] (N 2 xM matrix)

<u>Step 6:</u>

Compute the eigenvectors *ui* of *AAT*

The matrix *AAT* is very large --> not practical !!

<u>Step 6.1:</u> consider the matrix *AT A* (*M*x*M* matrix)

<u>Step 6.2:</u> compute the eigenvectors *vi* of *AT A*

AT Avi = m

i vi

What is the relationship between *usi* and *vi*?

AT Avi = m

i vi => *AAT Avi* = m

i Avi =>

CAvi = m

i Avi or *Cui* = m

i ui where *ui* = *Avi*

Thus, *AAT* and *AT A* have the same eigenvalues and their eigenvectors

are related as follows: *ui* = *Avi* !!

Note 1: *AAT* can have up to *N*2 eigenvalues and eigenvectors.

Note 2: *AT A* can have up to *M* eigenvalues and eigenvectors.

Note 3: The *M* eigenvalues of *AT A* (along with their corresponding

eigenvectors) correspond to the *M largest* eigenvalues of *AAT* (along

with their corresponding eigenvectors).

Step 6.3: Compute the *M* best eigenvectors of *AAT* : *ui* = *Avi*

(**important:** normalize *ui* such that ||*ui*|| = 1)

Step 7:

Keep only *K* eigenvectors (corresponding to the *K* largest eigenvalues)

(6) Representing faces onto this basis

Each face (minus the mean) F_i in the training set can be represented as a linear

combination of the best K eigenvectors:

$$\hat{F}_i - mean = \sum_{j=1}^{K} w_j u_j, \quad (w_j = u_j^T F_i)$$

(we call the u_j's *eigenfaces*)

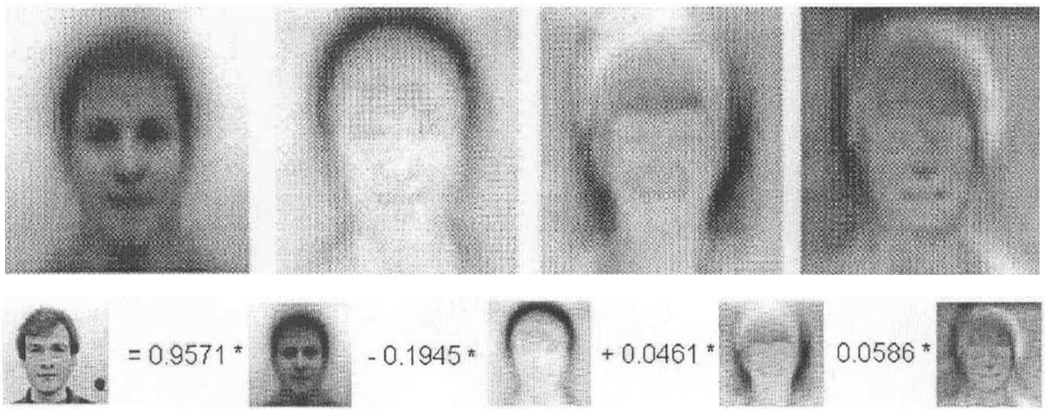


Figure 11 - u j's Eigenfaces

(6) Reconstruction of faces and non-faces

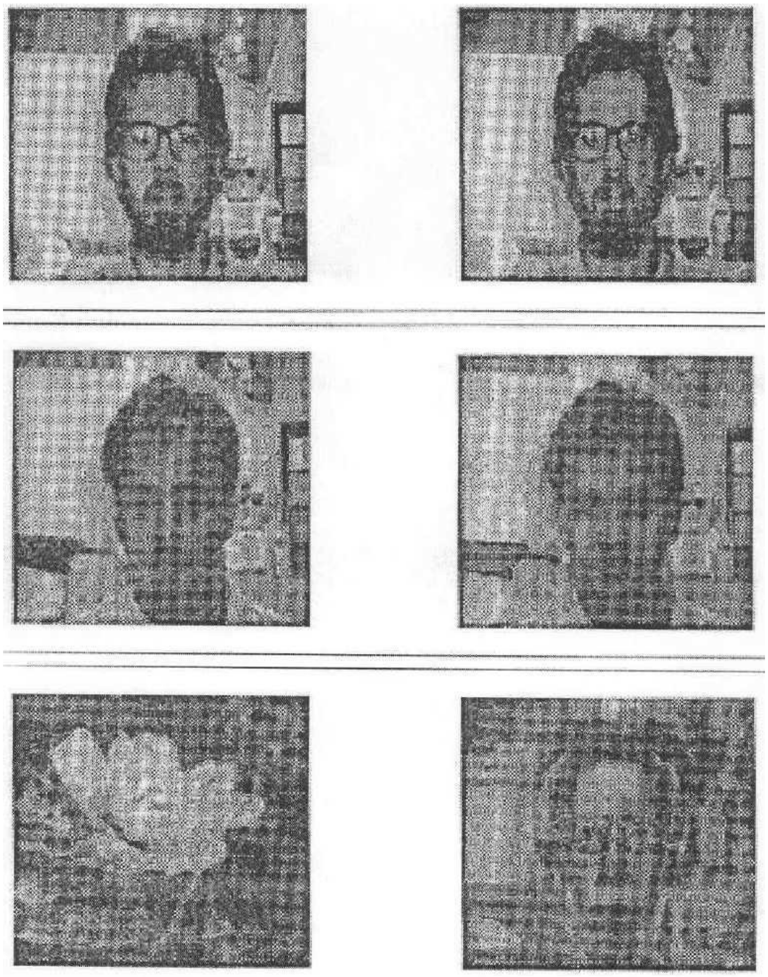


Figure 12 - Recognizing Faces and Non faces

g. Microsoft Visual Studio

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to developcomputer programs for Microsoft Windows, as well as web sites, web applications and web services. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. It can produce both native code and managed code.

Visual Studio includes a code editor supporting IntelliSense (the code completion component) as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger. Other built-in tools include a forms designer for building GUI applications, web designer, class designer, and database schema designer. It accepts plug-ins that enhance the functionality at almost every level—including adding support for source-control systems (like Subversion) and adding new toolsets like editors and visual designers for domain-specific languages or toolsets for other aspects of the software development lifecycle (like the Team Foundation Server client: Team Explorer).

Visual Studio supports different programming languages and allows the code editor and debugger to support (to varying degrees) nearly any programming language, provided a language-specific service exists. Built-in languages include C C++ and C++/CLI (via Visual

C++), VB.NET (via Visual Basic .NET), C# (via Visual C#), and F# (as of Visual Studio 2010). Support for other languages such as Python, Ruby, Node.js, and M among others is available via language services installed separately. It also supports XML/XSLT, HTML/XHTML,JavaScript and CSS. Java (and J#) were supported in the past.

h. Why Open Cv over Matlab

OpenCV has the edge over Matlab for image and video processing development . Although Matlab has an easy learning

	Matlab	OpenCV
Ease of Use	9	3
Speed	2	9
Resources Needed	4	9
Cost	4	10
Development Environment	8	6
Memory Management	9	4
Portability	3	8
Development of usefull programming skills	3	8
Help and Sample Code	8	9
Debugging	9	5
Total:	59	71

curve, built in memory management, a great help section, it is very slow to execute code, and is expensive to get started in. While OpenCV can be difficult to debug and requires much “housework code” needed for memory management, header files, etc., it wins out due to its free cost, the magnitude of sample code available on the internet, the short development path from prototype code to embedding code, the useful programming skills learnt from its use, and its super-fast speed. Matlab is a more “generic” programming language in that it was designed for many uses, demonstrated by its numerous toolboxes ranging from financial to specialized DNA analyzing tools. On the other hand, OpenCV was made for image processing. Each function and data structure was designed with the image processing coder in mind.

6. Results

Face detection and creating a data base

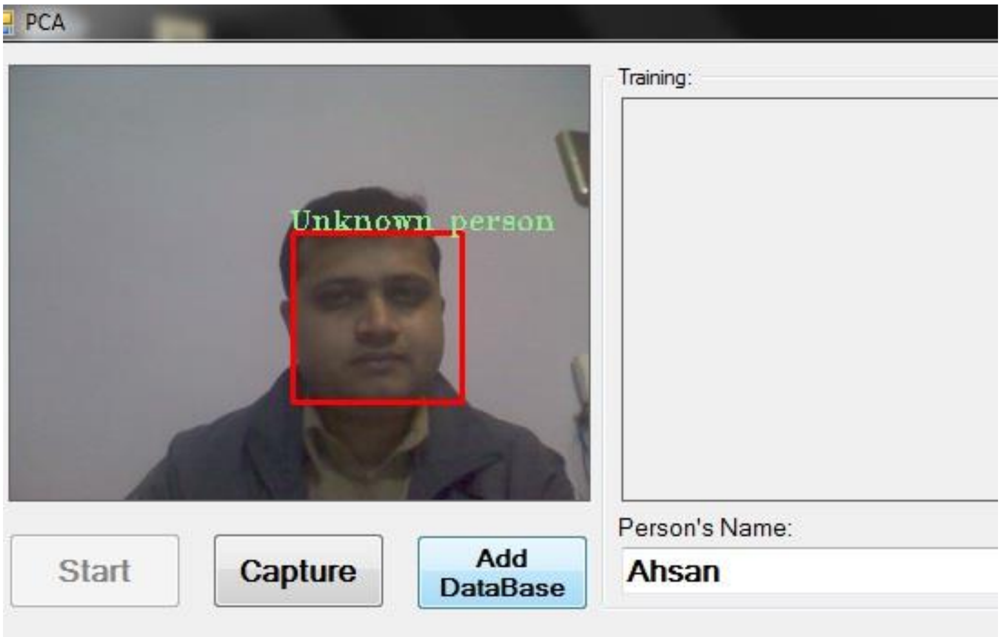


Figure 13 - Demo Picture

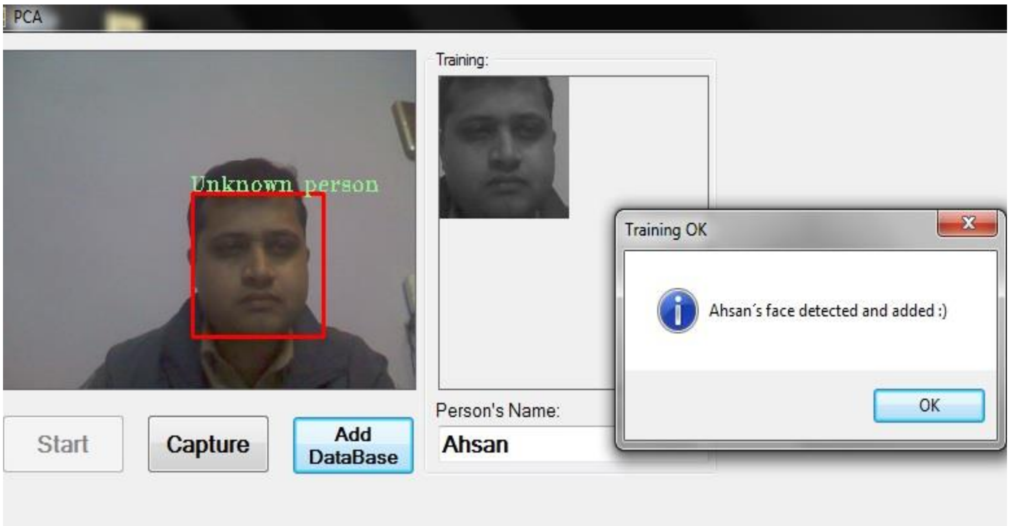


Figure 14 - Converting in a gray scale image

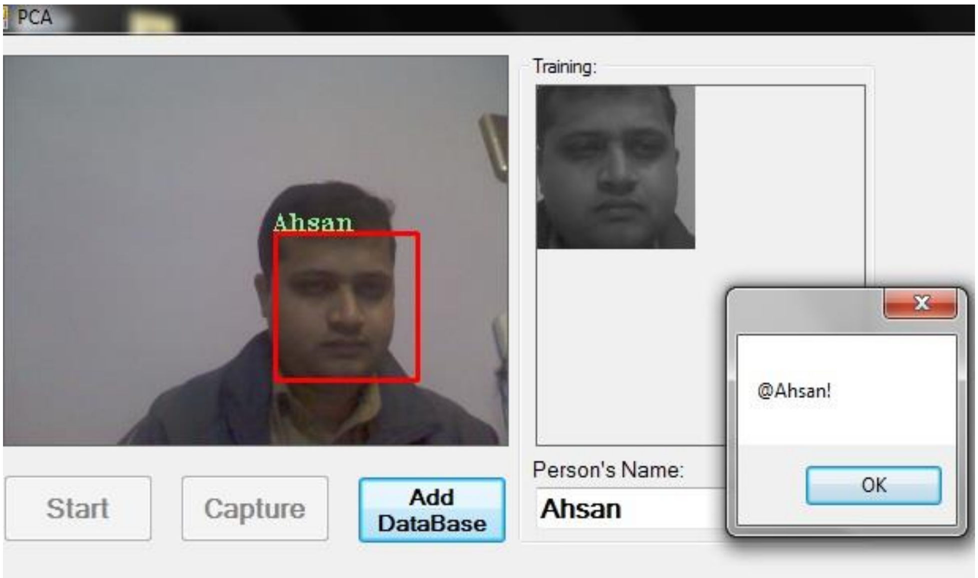


Figure 15 - Comparing with the date base and passing an image result to hardware

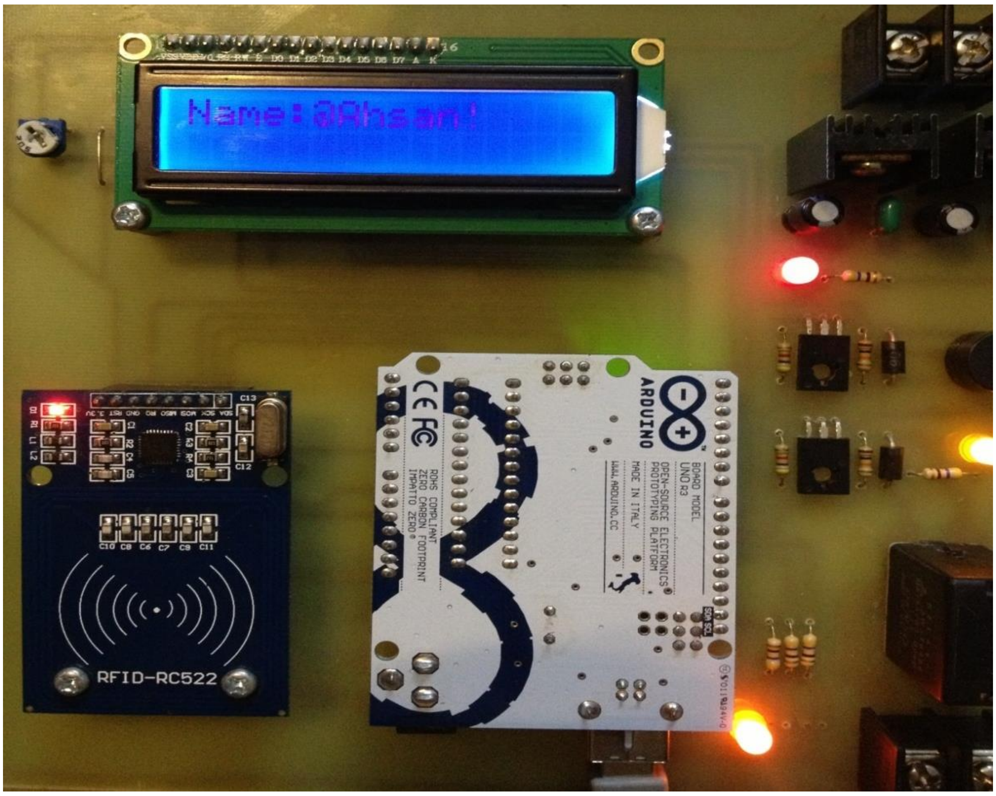


Figure 16 - Hardware results

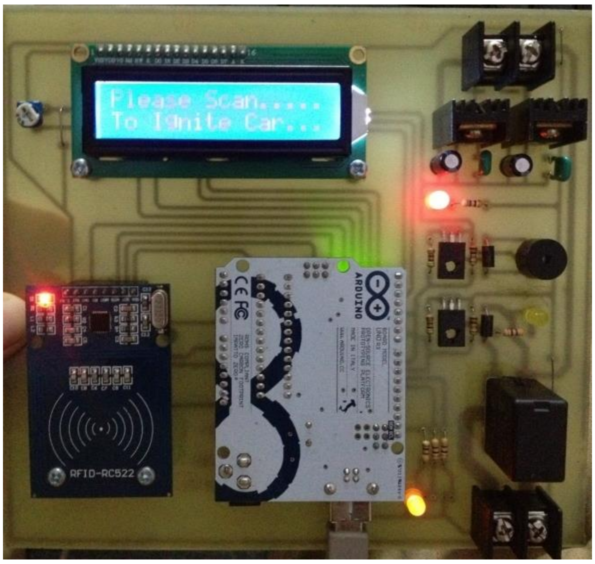
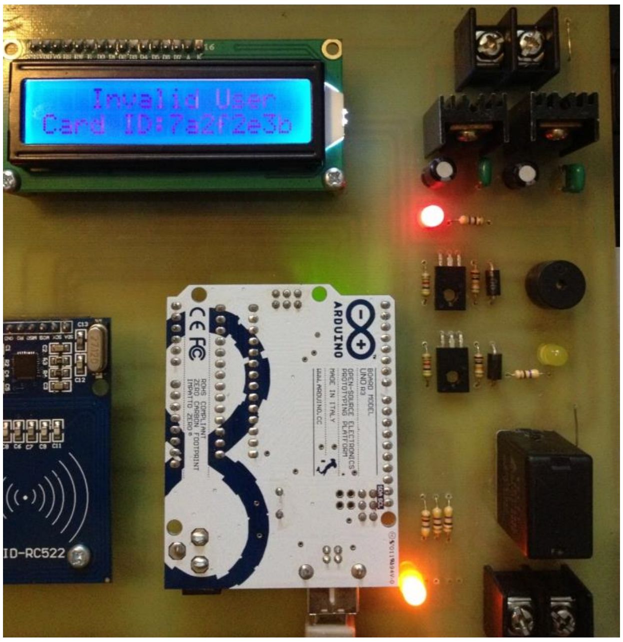


Figure 17 - RFID Results



7. Recommendations for Future work

a. System should be interfaced with the GPS for continuous monitoring as well

b. System should be interfaced with the GSM module to receive a sms alert in case of theft approach

c. RFID tag system used be installed with the door systems of important installations

8. Conclusion

From this we have implemented image recognition techniques that can provide the important functions by advanced intelligent automobile security, to avoid vehicle theft and protect the use of unauthenticated users. Secured and safety environment system for automobile users and also key points for the investigators can easily find out the hijacked image. We can predict the theft by using this in our daily life.

9. Bibliography

- http://www.uniassignment.com/essay-samples/information-technology/image-processing-based-car-security-system-information-technology-essay.php
- http://www.ripublication.com/iraer-spl/iraerv4n4spl_12.pdf
- http://thesai.org/Downloads/Volume5No5/Paper_25-Using_Digital_Image_Processing_to_Make_an_Intelligent_Gate.pdf
- https://en.wikipedia.org/wiki/Viola%E2%80%93Jones_object_detection_framework#cite_note-1
- https://www.addicore.com/v/vspfiles/downloadables/Product%20Downloadables/RFID_RC522/RFIDQuickStartGuide.pdf

Appendix A

Project Synopsis

REAL TIME VEHICLE SECURITY SYSTEM THROUGH FACE RECOGNITION

Extended Title: Car security using image processing through Arduino and key detection through RFID.

Brief Description of The Project: As vehicles theft is increasing in the present Era, there is an urge to make them more secure. So we are proposing to secure the vehicle using image processing as normal security system can be easily compromised. Only authorized person will be able to drive the vehicle. For the guest driver the concept of RFID will be used.

Scope of Work: The project is mainly divided into two parts. In first part the concept of image processing will be used. Second part will be based on the RFID recognition.Camera is incorporated in a secure place to take and process the image. Through face recognition, system will be able to differentiate between the authorized or unauthorized person and identify the right person and unauthorized use of vehicle.

Academic Objectives:

- Image processing
- Matlab,
- Programming of arduino,

Application / End Goal Objectives :

- Security of vehicles,
- Person identity,
- Remote sensing,
- Low cost surveillence.

Previous Work Done on The Subject : So far nothing is done in this field in Pakistan.

Material Resources Required:

• Arduino, • Radio Frequency Identifier, • Cameras, • Laptops and other circuitry elements.
No of Students Required : 3 **Group Members:**1. Capt Syed Ahsan Izhar 2. Capt Waqas Iftikhar 3. Capt Hasnain Kabir
Special Skills Required: Matlab coding, C and C++ coding and Arduino coding