



**IJIRCCCE**

e-ISSN: 2320-9801 | p-ISSN: 2320-9798



# INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

**Volume 9, Issue 6, June 2021**

**ISSN** INTERNATIONAL  
STANDARD  
SERIAL  
NUMBER  
INDIA

**Impact Factor: 7.542**



9940 572 462



6381 907 438



ijircce@gmail.com



www.ijircce.com

# A Face Recognition Based Smart Attendance System Using Opencv

Erla Sunitha Rani<sup>1</sup>, Battula Anusha<sup>2</sup>, Dantala Praveen<sup>3</sup>, Dokku Naveen<sup>4</sup>, A.Vishnu Vardhan<sup>5</sup>

UG Students, Dept. of Computer Science and Engineering, Vasireddy Venkatadri Institute of Technology, Nambur, Guntur, Andhra Pradesh, India<sup>1,2,3,4</sup>

Assistant Professor, Dept. of Computer Science and Engineering, Vasireddy Venkatadri Institute of Technology, Nambur, Guntur, Andhra Pradesh, India<sup>5</sup>

**ABSTRACT:** To maintain the attendance record with day-to-day activities is a challenging task. The conventional method of calling name of each student is time consuming and there is always a chance of proxy attendance. The following system is based on face recognition to maintain the attendance record of students. The daily attendance of students is recorded. The system automatically starts taking snaps and then apply face detection and recognition technique to the given image and the recognized students are marked as present and their attendance update with corresponding time. We have used OpenCv to develop this system, histogram of oriented gradient method is used to detect faces in images and OpenCv methods is used to compute and compare feature facial of students to recognize them. Our system is capable to identify multiple faces in real time.

**KEYWORDS:** HOG, OpenCV, camera, attendance, biometric, face recognition, spreadsheet

## I. INTRODUCTION

Maintenance of attendance is incredibly necessary altogether at the institutes for checking the performance of employees/students. Some institutes take attending manually using paper or file-based approaches and a few have adopted ways of automatic attendance using some biometric techniques. The recent methodology for taking attendance is by calling out the name or roll number of the scholar to record attendance. It's a long and less efficient method of marking attending as a result of as we all know the info written within the paper typically may be lost or is less accurate as a result of students often mark every other's attending proxy. Therefore, to unravel these issues and avoid errors, we recommend computerizing this method by providing a system that records and manages student's attending mechanically with no need for lecturers' interference.

Every biometric system consists of the enrolment process during which distinctive features of someone are stored within the database and then there are processes of identification and verification. These 2 processes compare the biometric feature of someone with previously stored biometric details captured at the time of enrollment. Biometric templates are of many varieties like Fingerprints, Eye Iris, Face, Signature, and Voice. Our system uses the face recognition approach for the automated Attendance Management System.

By considering this truth our system is going to be quicker and correct in marking the attendance of individual students. Face recognition consists of two steps, in the first step faces are detected in the image and then these detected faces are compared with the database for verification. Face detection is employed to find the position of face region and face recognition is employed for marking the attendance. The info can store the faces of scholars. Once the face of the scholar matches with one in each of the faces kept within the database then the attendance is recorded. Various sturdy algorithms are developed that offer accurate performance to tackle face detection and recognition problems. These algorithms or methods are the most successfully and widely used for face detection and recognition applications. The algorithms are as follow:

### 1. Principal Component Analysis (PCA)

It is a dimensionality reduction technique that employs Eigenvalues and EigenVectors to reducedimensionality and project a training sample/data on small feature space.

### 2. Linear Discriminant Analysis (LDA)

This methodology uses concepts of class. Its goal is to perform dimensionality reduction while saving as much of the class discriminatory information as possible. It minimizes variation within each class and maximizes class separation.

### 3. Skin Color Based Approach

One of the simplest algorithms for detecting skin pixels is to use a skin color algorithm. Each pixel is classified as skin color and non-skin color. This classification is based on its color component, which is modeled by Gaussian probability density.

### 4. Artificial neural networks based algorithm

An artificial neural network (ANN) is mostly used as a method for recognition. ANN will be implemented once a face has been detected to identify and recognize who the person is by calculating the weight of the facial information .

## II. LITERATURE SURVEY

In Kawaguchi introduced a lecture attendance system with a new method called continuous monitoring, and the student's attendance marked automatically by the camera which captures the photo of a student in the class. The architecture of the system is simple since two cameras equipped with the wall of the class. The first one is a capturing camera used to capture the image student in the class and the second camera is sensor camera is used to getting the seat of a student inside the class and the camera capturing will snap the image of the student. The system compares the picture taking from a camera capturing images and faces in the database done much time to perfect the attendance.

In 2012 N. Karthik introduced an automated attendance management system using face recognition technique which used the Principal Component Analysis To implementation the system, use two libraries such OpenCV is a computer vision library and FLTK (Light Tool Kit. Both libraries helped the development such as OpenCV support algorithm and FLTK used to design the interface. In the system, there are Request Matching and Adding New fact to Database. In Request Matching, the first step is open the camera and snap the photo after the extraction the frontal face. The next step is recognizing the face with the training data and project the extracted face onto the Principal Component Analysis. The decisive step displays the nearest face with the acquired images. Apart from that, adding a new face into the database is snap the photo after that extract the frontal face images and then perform the Haar cascade Method to find the object in the image in different window size. The next step is to store the image into the database and to learn the face then perform the Principal Component Analysis Algorithm. The last step is storing the information inside the face XML file. The system is focused on the algorithm to improve the face detection from acquired images or videos.

In [3] the author also proposed a system which implements automatic attendance using face recognition. The system which can extract the object in the face such nose, mouth by using MATLAB with Principal Component Analysis (PCA). The system designed to resolve the issues of attendance marking system such as time-consuming. As the result of the experiment show that this paper, the system can recognize in case the dark background or difference view of the face in the classroom.

## III. METHODOLOGY / APPROACH

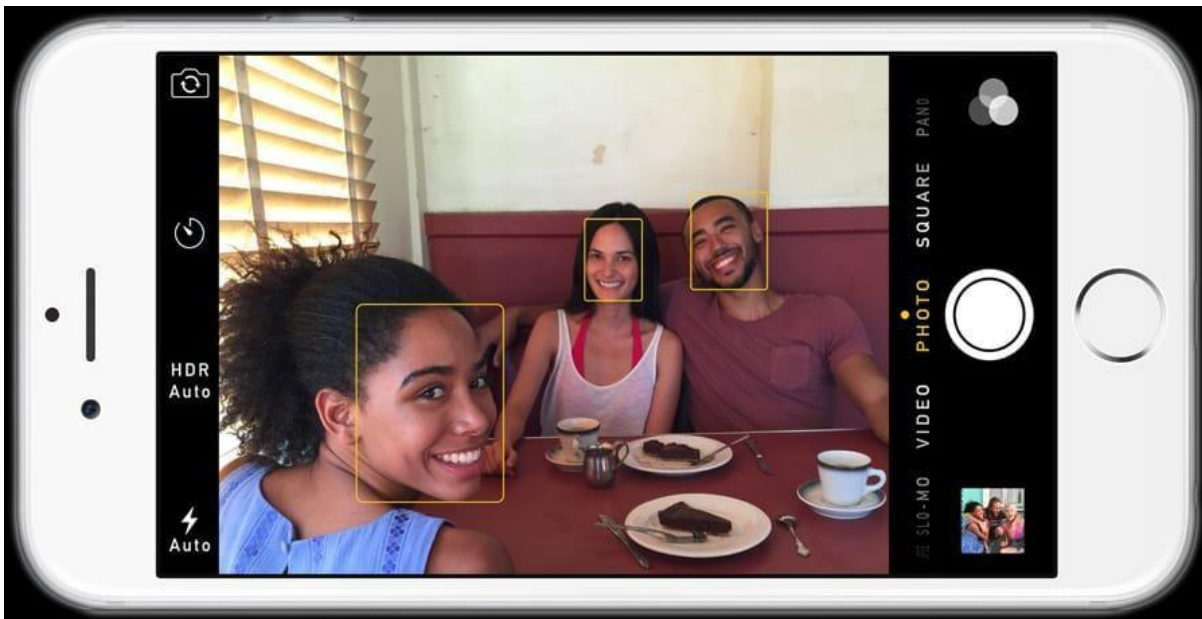
Let us tackle this problem one step at a time. For each step, we will learn about a different machine learning algorithm. I am not going to explain every single algorithm completely to keep this from turning into a book, but you will learn the main ideas behind each one and you will learn how you can build your own facial recognition system in Python using OpenFace and dlib.

### Step 1: Finding all the Faces

The first step in our pipeline is *face detection*. Obviously, we need to locate the faces in a photograph before we can try to tell them apart!

If you have used any camera in the last 10 years, you have seen face detection in action:



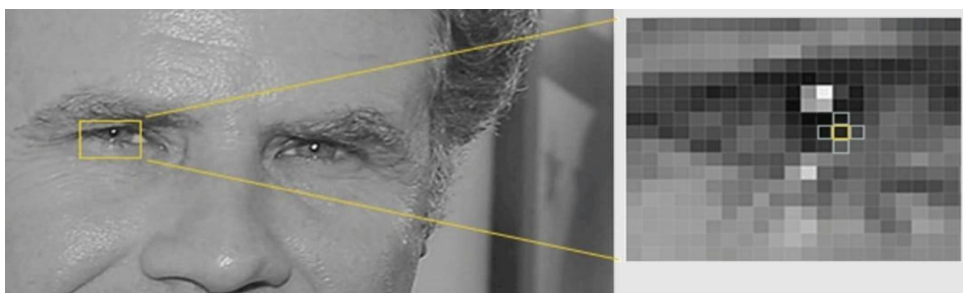


*Fig: 1 Finding Faces*

Face detection is a splendid feature for cameras. When the camera can automatically pick out faces, it can make sure that all the faces are in focus before it takes the picture. But we will use it for a different purpose — finding the areas of the image we want to pass on to the next step in our pipeline.

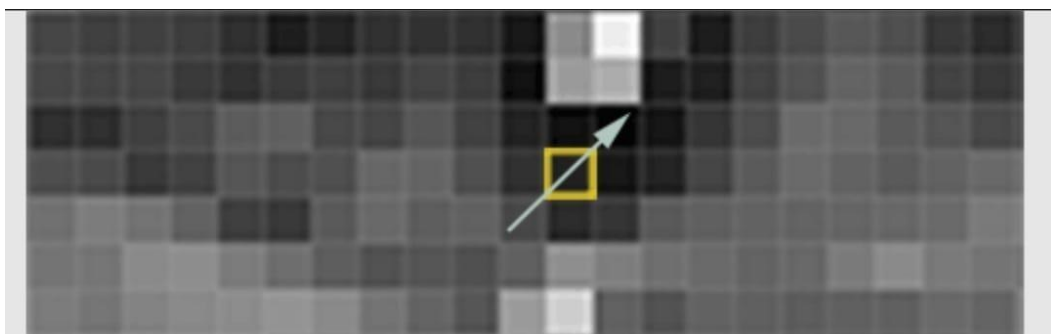
Face detection went mainstream in the early 2000's when Paul Viola and Michael Jones invented a way to detect faces that was fast enough to run on cheap cameras. However, much more reliable solutions exist now. We are going to use a method invented in 2005 called Histogram of Oriented Gradients — or just **HOG** for short.

To find faces in an image, we will start by making our image black and white because we do not need color data to find faces:



*Fig 2 Detecting Pixels*

Our goal is to figure out how dark the current pixel is compared to the pixels directly surrounding it. Then we want to draw an arrow showing in which direction the image is getting darker:



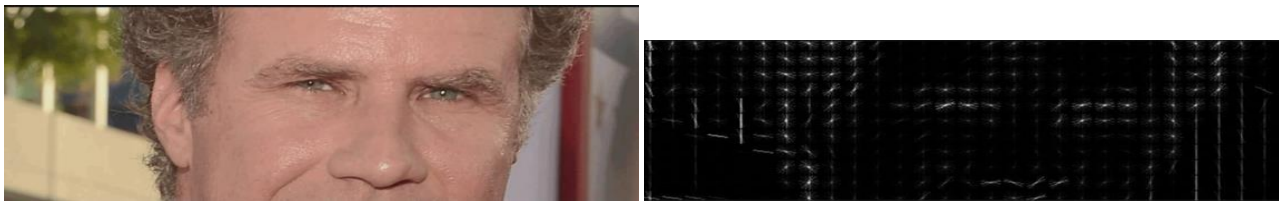
Looking at just this one pixel and the pixels touching it, the image is getting darker towards the upper right

This might seem like a random thing to do, but there is a good reason for replacing the pixels with gradients. If we analyze pixels directly, dark images and light images of the same person will have different pixel values. But by only considering the *direction* that brightness changes, both dark images and bright images will end up with the same exact representation. That makes the problem a lot easier to solve!

But saving the gradient for every single pixel gives us too much detail. We end up missing the forest for the trees. It would be better if we could just see the basic flow of lightness/darkness at a higher level so we could see the basic pattern of the image.

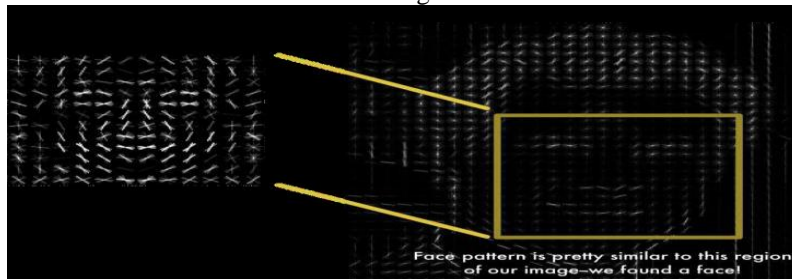
To do this, we will break up the image into small squares of 16x16 pixels each. In each square, we will count how many gradients point in each major direction (how many points up, point up-right, point right, etc....). Then we will replace that square in the image with the arrow directions that were the strongest.

The result is we turn the original image into a quite simple representation that captures the basic structure of a face in a straightforward way:



The original image is turned into a HOG representation that captures the major features of the image regardless of image brightness.

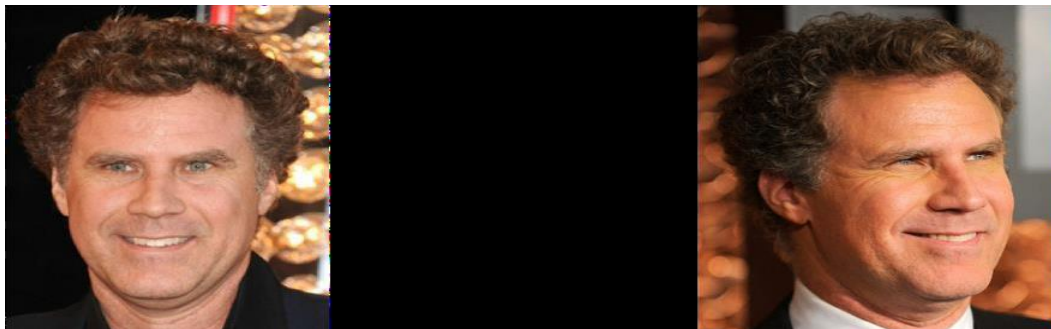
To find faces in this HOG image, all we must have to do is find the part of our image that looks the most like a known HOG pattern that was extracted from a bunch of other training faces:



HOG face pattern generated from lots of face images & HOG version of our image

## Step 2: Posing and Projecting Faces

When, we isolated the faces in our image. But now we must deal with the problem that faces turned different directions look different to a computer:



Humans can easily recognize that both images are of Will Ferrell, but computers would see these pictures as two completely different people.

To account for this, we will try to warp each picture so that the eyes and lips are always in the same place in the image. This will make it a lot easier for us to compare faces in the next steps.

To do this, we are going to use an algorithm called **face landmark estimation**. There are lots of ways to do this, but we are going to use the approach invented in 2014 by Vahid Kazemi and Josephine Sullivan.

The basic idea is we will produce 68 specific points (called *landmarks*) that exist on every face — the top of the chin, the outside edge of each eye, the inner edge of each eyebrow, etc. Then we will train a machine learning algorithm to be able to find these 68 specific points on any face:

Now that we know where the eyes and mouth are, we will simply rotate, scale, and shear the image so that the eyes and mouth are centered as best as possible. We will not do any fancy 3d warps because that would introduce distortions into the image. We are only going to use basic image transformations like rotation and scale that preserve parallel lines (called affine transformations):

Now no matter how the face is turned, we can center the eyes and mouth are in the same position in the image. This will make our next step a lot more accurate.

If you want to try this step out yourself using Python and dlib, here is the code for finding face landmarks and here is the code for transforming the image using those landmarks.

### Step 3: Encoding Faces

Now we are to the meat of the problem — telling faces apart. This is where things get interesting!

The simplest approach to face recognition is to directly compare the unknown face we found in Step 2 with all the pictures we have of people that have already been tagged. When we find a previously tagged face that looks like our unknown face, it must be the same person. Seems like an innovative idea, right?

There is a huge problem with that approach. A site like Facebook with billions of users and a trillion photos cannot loop through every previous-tagged face to compare it to every newly uploaded picture. That would take way too long. They need to be able to recognize faces in milliseconds, not hours.

#### *The most reliable way to measure a face:*

Ok, so which measurements should we collect from each face to build our known face database? Ear size? Nose length? Eye color? Something else?

It turns out that the measurements that seem obvious to us humans (like eye color) do not really make sense to a computer looking at individual pixels in an image. Researchers have discovered that the most accurate approach is to let the computer figure out the measurements to collect itself. Deep learning does a better job than humans at figuring out which parts of a face are important to measure.

The solution is to train a Deep Convolutional Neural Network (just like we did in Part 3). But instead of training the network to recognize pictures objects like we did last time, we are going to train it to generate 128 measurements for each face.

The training process works by looking at 3 face images at a time:

1. Load a training face image of a known person
2. Load another picture of the same known person
3. Load a picture of a different person

After repeating this step millions of times for millions of images of thousands of different people, the neural network learns to reliably generate 128 measurements for each person. Any ten different pictures of the same person should give the same measurements.

Machine learning people call the 128 measurements of each face an **embedding**. The idea of reducing complicated raw data like a picture into a list of computer-generated numbers comes up a lot in machine learning (especially in language translation). The exact approach for faces we are using was invented in 2015 by researchers at Google but many similar approaches exist.

### *Encoding our face image*

This process of training a convolutional neural network to output face embeddings requires a lot of data and computer power. Even with an expensive NVidia Telsa video card, it takes about 24 hours of continuous training to get good accuracy.

But once the network has been trained, it can generate measurements for any face, even ones it has never seen before! So, this step only needs to be done once. Lucky for us, the fine folks at OpenFace already did this and they published several trained networks which we can directly use. Thanks Brandon Amos and team!

### **Step 4: Finding the person's name from the encoding**

This last step is the easiest step in the entire process. All we must have to do is find the person in our database of known people who has the closest measurements to our test image.

You can do that by using any basic machine learning classification algorithm. No fancy deep learning tricks are needed. We will use a simple linear SVM classifier, but lots of classification algorithms could work.

All we need to do is train a classifier that can take in the measurements from a new test image and tells which known person is the closest match. Running this classifier takes milliseconds. The result of the classifier is the name of the person!

### *Running this yourself*

Let us review the steps we followed:

1. Encode a picture using the HOG algorithm to create a simplified version of the image. Using this simplified image, find the part of the image that most looks like a generic HOG encoding of a face.
2. Figure out the pose of the face by finding the main landmarks in the face. Once we find those landmarks, use them to warp the image so that the eyes and mouth are centered.
3. Pass the centered face image through a neural network that knows how to measure features of the face. Save those 128 measurements.
4. Looking at all the faces we have measured in the past, see which person has the closest measurements to our face's measurements. That is our match!

Now that you know how this all works, here's instructions from start-to-finish of how run this entire face recognition pipeline on your own computer.

## **IV.RESULTS & DISCUSSION**

Smart Attendance Management System is simple and works efficiency. The system works automatically once the registration of individual student created by the administration.

It consists of the following modules,

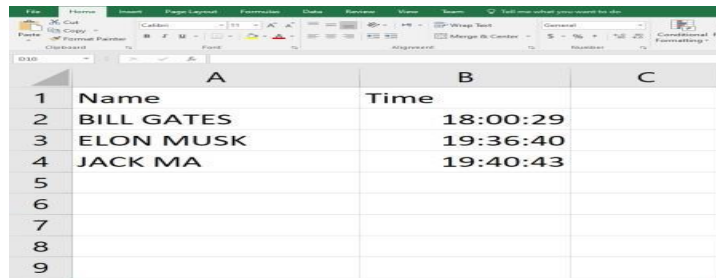
- Face Recognition
- Addition of name with their corresponding time.
- Attendance sheet generation and import to Excel (xlsx) format.





**Fig 3 Detecting Faces**

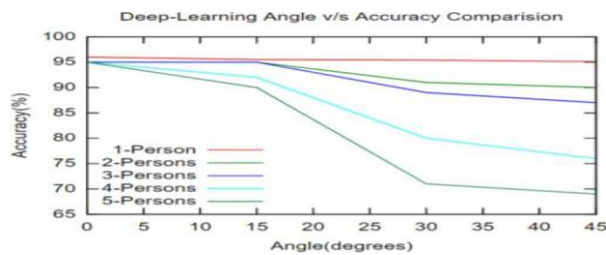
Attendance system proved to recognize images in different angle and light conditions. The faces which are not in our training dataset are marked as unknown. The attendance of recognize images of students is marked in real time. And import to excel sheet and saved by the system automatically.



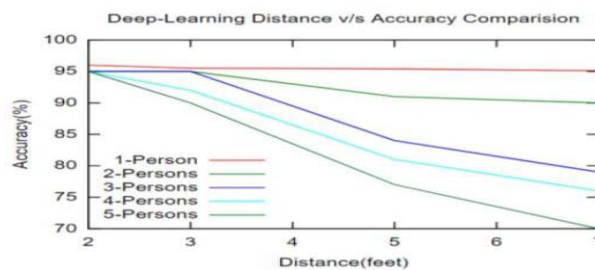
	A	B	C
1	Name	Time	
2	BILL GATES	18:00:29	
3	ELON MUSK	19:36:40	
4	JACK MA	19:40:43	
5			
6			
7			
8			
9			

**Fig 4 Attendance Sheet**

Excel sheet generated by the system automatically with the corresponding subject id, date, and time.



**Fig 5 Accuracy graph in terms of angles**



**Fig 6 Accuracy graph in terms of distance**



## V.CONCLUSION

Smart attendance management system is designed to solve the issues of existing manual systems. We have used face recognition concept to mark the attendance of student and make the system better. The system performs satisfactory in different poses and variations.

In future this system needs, be improved because this system sometimes fails to recognize students from some distance, also we have some processing limitation, working with a system of high processing may result even better performance of this system.

## REFERENCES

- 1.R. Manjunatha, Dr. R. Nagaraja (2017), "Home Security System and Door Access Control Based on Face Recognition", International Research Journal of Engineering and Technology (IRJET), Volume 4, Issue 3.
2. Kennedy O. Okokpujie, Etinosa Noma-Osaghae, Olatunji J. Okesola, Samuel N. John (2017), "Design and Implementation of a Student Attendance System Using Iris Biometric Recognition", IEEE (International Conference on Computational Science and Computational Intelligence).
3. Miss. Nilofer Tamboli, Dr. M. M. Sardeshmukh (2017), "Facial Based Attendance Monitoring System Using Discriminative Robust Local Binary Pattern", Third International Conference on Computing, Communication, Control And Automation (ICCUBEA).
4. Prof.A.D.Sawant, Arti Dongare, Ichha Gilbile, Amitsingh Thakur, PoojaTekawade (2017), "A Survey on Smart Attendance System Based on Various Technologies", International Journal of Innovative Research in Computer and Communication Engineering (IJIRCE), Volume 5 Issue 10.
5. Jun Iio, Higashinakano, Hachioji-shi (2016), "Attendance Management System using a Mobile Device and a Web Application", International Conference on Network-Based Information Systems (NBIS).
6. Samuel Lukas, Aditya Rama Mitra, Ririn Ikana Desanti, Dion Krisnadi (2016), "Student Attendance System in Classroom Using Face Recognition Technique", International Conference on Information and Communication Technology Convergence (ICTC) .
7. Aman Jobanputra, Shubham Jain, Kruttika Choithani (2016), "Smart Attendance Management System", International Journal of Computer Science Trends and Technology (IJCT), Volume 4 Issue 5.



**INNO**  **SPACE**  
SJIF Scientific Journal Impact Factor  
**Impact Factor: 7.542**



**ISSN** INTERNATIONAL  
STANDARD  
SERIAL  
NUMBER  
**INDIA**



# INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

 **9940 572 462**  **6381 907 438**  **ijircce@gmail.com**



[www.ijircce.com](http://www.ijircce.com)

Scan to save the contact details