Face Detection and Tracking using OpenCV

S.V. Viraktamath*, Mukund Katti**, Aditya Khatawkar*** & Pavan Kulkarni****

*Faculty, Department of Electronics and Communication Engineering, SDM College of Engineering & Technology, Dharwad, Karnataka, INDIA. E-Mail: svvmath@gmail.com
**Student, Department of Electronics and Communication Engineering, SDM College of Engineering & Technology, Dharwad, Karnataka, INDIA. E-Mail: kgmukundkatti@gmail.com
***Student, Department of Electronics and Communication Engineering, SDM College of Engineering & Technology, Dharwad, Karnataka, INDIA.
****Student, Department of Electronics and Communication Engineering, SDM College of Engineering & Technology, Dharwad, Karnataka, INDIA.

***Abstract*—**An application for automatic face detection and tracking on video streams from surveillance cameras in public or commercial places is discussed in this paper. In many situations it is useful to detect where the people are looking for, e.g. in exhibits, commercial malls, and public places in buildings. Prototype is designed to work with web cameras for the face detection and tracking system based on open source platforms Arduino and OpenCV. The system is based on AdaBoost algorithm and abstracts faces Haar-Like features. This system can be used for security purpose to record the visitor face as well as to detect and track the face. A program is developed using OpenCV that can detect people's face and also track from the web camera.

***Keywords*—**Arduino; Haar Filter; OpenCV; Processing; Servos.

***Abbreviations*—**AdaptiveBoost (AdaBoost); Open Computer Vision (OpenCV).

I. INTRODUCTION

THE research purpose of computer vision aims to simulate the manner of human eyes directly by using computer. Computer vision is such kind of research field which tries to percept and represent the 3D information for world objects. Its essence is to reconstruct the visual aspects of 3D object by analyzing the 2D information extracted accordingly. 3D objects surface reconstruction and representation not only provide theoretical benefits, but also are required by numerous applications.

Face detection is a process, which is to analysis the input image and to determine the number, location, size, position and the orientation of face. Face detection is the base for face tracking and face recognition, whose results directly affect the process and accuracy of face recognition. The common face detection methods are: knowledge-based approach, Statistics-based approach and integration approach with different features or methods. The knowledge-based approach [Feng, 2004; Faizi, 2008] can achieve face detection for complex background images to some extent and also obtain high detection speed, but it needs more integration features to further enhance the adaptability. Statistics-based approach [Liang et al., 2002; Wang et al., 2008] detects face by judging all possible areas of images by classifier, which is to look the face region as a class of models, and use a large number of "Face" and "non-face" training samples to construct the classifier.

The method has strong adaptability and robustness, however, the detection speed needs to be improved, because it requires test all possible windows by exhaustive search and has high computational complexity. The AdaBoost algorithm [Zhang, 2008; Guo & Wang, 2009] arose in recent years; it trains the key category features to the weak classifiers, and cascades them into a strong classifier for face detection. The method has real-time detection speed and high detection accuracy, but needs long training time. The digital image of the face generated is a representation of a two-dimensional image as a finite set of digital values, called picture elements or pixels [Lu et al., 1999]. Pixel values typically represent gray levels, colours, heights, opacities etc. It is to be noted that digitization implies that a digital image is an approximation of a real scene. Recently there has been a tremendous growth in the field of computer vision. The conversion of this huge amount of low level information into usable high level information is the subject of computer vision. It deals with the development of the theoretical and algorithmic [Jiang, 2007] basis by which useful information about the 3D world can be automatically extracted and

analyzed from a single or multiple 2D images of the world as shown in figure 1.

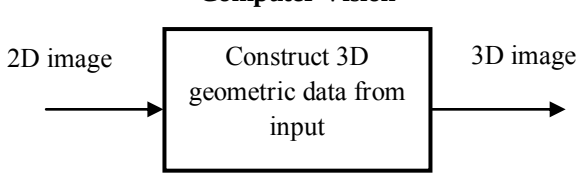


Figure1: Typical Data Processing in Computer Vision

This paper describes a system that can detect and track human face in real time using haar-like features where the detection algorithm is based on wavelet transform. In computer vision, low level processing involves image processing tasks in which the quality of the image is improved for the benefit of human observers and higher level routines to perform better [Viola & Jones, 2001]. Intermediate level processing involves the processes of feature extraction and pattern detection tasks.

Figure 2: Process Levels in Computer Vision

High level vision involves autonomous interpretation of scenes for pattern classification, recognition and identification of objects in the scenes as well as any other information required for human understanding as shown in figure 2. Statistics-based approach to this paper detects face by judging all possible areas of images by classifier, which is to look the face region as a class of models, and use a large number of "Face" and "Non-face" training samples to construct the classifier. The method has strong adaptability and robustness. The program can rectangle the face area with the data got from web camera video stream.

II. RELATED WORK

Face detection is used in biometrics, often as a part of (or together with) a facial recognition system. It is also used in video surveillance, human computer interface and image database management. Some recent digital cameras use face detection for autofocus [DCRP Review: Canon PowerShot S5 IS]. Face detection is also useful for selecting regions of interest in photo slideshows that use a pan-and-scale Ken Burns effect. Face detection is gaining the interest of marketers. A webcam can be integrated into a television and detect any face that walks by. The system then calculates the race, gender, and age range of the face. Once the information is collected, a series of advertisements can be played that is specific toward the detected race/gender/age. This paper shows prototype or partial implementation of this type of work. Face detection is also being researched in the area of energy conservation [Energy Conservation].

Methodology for face recognition based on information theory approach of coding and decoding the face image is discussed in [Sarala A. Dabhade & Mrunal S. Bewoor, 2012]. Proposed methodology is connection of two stages – Face detection using Haar Based Cascade classifier and recognition using Principle Component analysis. Various face detection and recognition methods have been evaluated [Faizan Ahmad et al., 2013] and also solution for image detection and recognition is proposed as an initial step for video surveillance. Implementation of face recognition using principal component analysis using 4 distance classifiers is proposed in [Hussein Rady, 2011]. A system that uses different distance measures for each image will perform better than a system that only uses one. The experiment show that PCA gave better results with Euclidian distance classifier and the squared Euclidian distance classifier than the City Block distance classifier, which gives better results than the squared Chebyshev distance classifier. A structural face construction and detection system is presented in [Sankarakumar et al., 2013]. The proposed system consists the different lightning, rotated facial image, skin color etc.

III. DESCRIPTION OF TOOLS

In this section the tools and methodology to implement and evaluate face detection and tracking using OpenCV are detailed.

3.1. OPENCV

OpenCV (*Open* Source Computer Vision Library) is a library of programming functions mainly aimed at real time computer vision, developed by Intel and now supported by Willow Garage [Lu et al., 1999]. It is free for use under the open source BSD license. The library is cross-platform. It focuses mainly on real-time image processing. If the library finds Intel's Integrated Performance Primitives on the system [Open Source Computer Vision Library Reference Manual-intel; Gary Bradski & Adrian Kaehler O"Reilly, 2008], it will use these proprietary optimized routines to accelerate it.

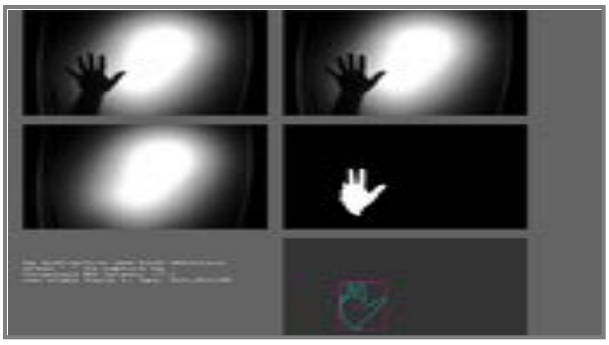


Figure 3: Object Detection Pattern using OpenCV

The library was originally written in C and this C interface makes OpenCV portable to some specific platforms such as digital signal processors. Wrappers for languages such as C#, Python, Ruby and Java (using JavaCV) have been developed to encourage adoption by a wider audience [Zhang, 2008]. However, since version 2.0, OpenCV includes both its traditional C interface as well as a new C++ interface. This new interface seeks to reduce the number of lines of

code necessary to code up vision functionality as well as reduce common programming errors such as memory leaks (through automatic data allocation and de-allocation) that can arise when using OpenCV in C as shown in figure 4.

Most of the new developments and algorithms in OpenCV are now developed in the C++ interface [Bradski & Kaebler, 2009]. Unfortunately, it is much more difficult to provide wrappers in other languages to C++ code as opposed to C code; therefore the other language wrappers are generally lacking some of the newer OpenCV 2.0 features. A CUDA-based GPU interface has been in progress since September 2010.

3.2. Processing Software

The Processing language is a text programming language specifically designed to generate and modify images. Processing strives to achieve a balance between clarity and advanced features. The system facilitates teaching many computer graphics and interaction techniques including vector/raster drawing, image processing, color models, mouse and keyboard events, network communication, and object-oriented programming. Libraries easily extend Processing‟s ability to generate sound, send/receive data in diverse formats, and to import/export 2D and 3D file formats [Ben Fry & Casey Reas, 2007].

Processing is for writing software to make images, animations, and interactions. Processing is a dialect of a programming language called Java; the language syntax is almost identical, but Processing adds custom features related to graphics and interaction as shown in figure 3. The graphic elements of Processing are related to PostScript (a foundation of PDF) and OpenGL (a 3D graphics specification). Because of these shared features, learning Processing is an entry-level step to programming in other languages and using different software tools.

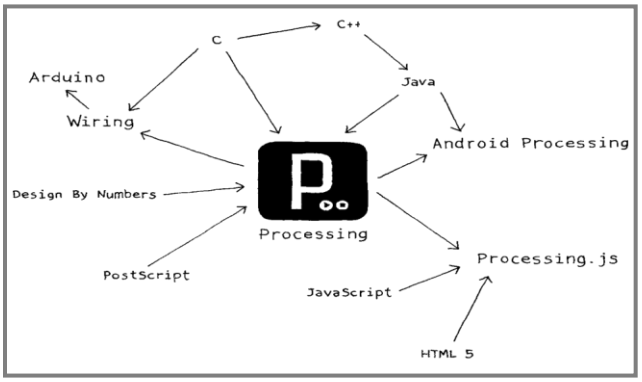


Figure 4: Structure Design of Processing

3.3. Arduino

Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. The hardware consists of a simple open hardware design for the Arduino board with an Atmel AVR processor and on-board input/output support. The software consists of a standard programming language compiler and the boot loader that runs on the board.

Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). Arduino projects can be stand-alone or they can communicate with software running on a computer (e.g. Flash, Processing, and MaxMSP)

The board as shown in figure 5 can be built by hand or purchased preassembled the software can be downloaded for free. The hardware reference designs (CAD files) are available under an open-source license.

Figure 5: Arduino UNO Board

IV. FACE DETECTION

In this section the base algorithm used to detect the face is discussed [Feng, 2004]. AdaBoost algorithm is discussed first then feature selection is discussed.

4.1. ADABOOST

In 1995, Freund and Schapire first introduced the AdaBoost algorithm [Faizi, 2008]. It was then widely used in pattern recognition.

The AdaBoost Algorithm

1. **Input**: Give sample set S = **(**x_1, y_1),.....(x_n,y_n) $x_i \epsilon X$, $y_i \epsilon Y = \{-1,+1\}$, number of iterations T
2. **Initialize**: $w_{i,j} = \frac{1}{N}$ i = 1,.....,N
3. **For** t = 1,2,...,T,
   i) Train weak classifier using distribution W_t.
   ii) Calculate the weight (w_i) training error for each hypothesis
   $$h_n\ \varepsilon_t = \sum_{i=1}^{N} W_{t,i}|h_i - y_i|$$
   iii) Set:
   $$a_t = \frac{1}{2} log \frac{1-\varepsilon_t}{\varepsilon_t}$$
   iv) Update the weights:
   $$W_{t+1,i} = 1 + \frac{W_{t,i}}{Z_t} \times \begin{matrix} e^{-a_t} \\ e^{a_t} \end{matrix}$$
   $$= \frac{w_{t,i}\ \exp(-a_t y_i h_t(x_i)}{Z_t}$$
   Z_t is normalization constant
4. **Output**: the final hypothesis, also the stronger classifier.
   $$H(x) = sign(\ \sum_{t=1}^{T} a_t h_t(x))$$

4.2. Feature Selection using Haar like Features

In the implementation of face detection, Xi contains a huge number of face features, and some of the features with low ϵ_i to train our strong classifier are selected. By AdaBoost algorithm this can be achieved automatically [Lu et al., 1999]. For each iteration ϵ_i with each feature in X_i can be calculated and then the lowest one is what we need. For doing this, the face detection rapid could be very fast. In next part, you will find there are many haar-like features, so it is hard to make use of all them. Face features are abstracted from the input image and are used to train the classifier, modify weights [Introduction to OpenCV].

Face features are abstracted from the input images and are used to train the classifiers, modify weights as mentioned in [Viola & Jones, 2001]. In 2001, Viola et al. first introduced the haar-like features. The haar-like features are rectangle features and value is that the sum of pixels in black district subtracts the sum of pixels in white district [Guo & Wang, 2009] as shown in figure 6 and figure 8. Rainer Lienhart had done an extended set of haar-like features which significantly enrich the basic set of simple haar-like features, and can get a better hit rate.

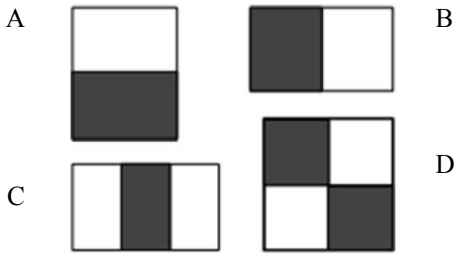


Figure 6: Haar-like Features Introduced in Viola's Paper

Two-rectangle features are „A" and „B". „C" is three-rectangle feature and „D" is four-rectangle feature. At a size of 24x 24, there are more than 180,000 rectangle features.

V. EXPERIMENTAL SETUP

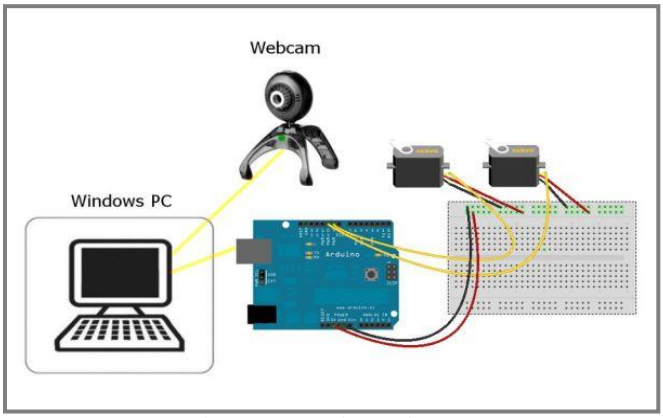


Figure 7: Experimental Setup

To implement face detection and tracking tools required are:

5.1. Software Required

OpenCV 2.3.1 super pack for windows, Arduino IDE 1.0 for windows, Processing IDE for windows.

5.2. Hardware Required

PC preferably running windows 7 sp1, Arduino uno or compatible plus power source (5v-dc), standard servos *2, webcam w/usb interface, breadboard, jump wires, hobby wire to tie pan/tilt servos and webcam together.

Figure 7 shows experimental setup used. Breadboard is used to make connections. The various connections required are as given below

SERVOS:

1. The yellow/signal wire for the pan (x axis) servo goes to digital pin 9.
2. The yellow/signal wire for the tilt (y axis) servo goes to digital pin 10.
3. The red/VCC wires of both servos go to the arduino's 5v pin.
4. The black/GND wires of both servos go to arduino's gnd pin.

WEBCAM:

The webcam's USB goes to the pc. The code will identify it via a number representing the USB port its connected.

ARDUINO:

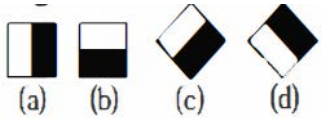
The arduino uno is connected to the pc via usb. Take note of the com port the USB is connected to. COM port can be found from the arduino tools/serial ports menu. Check mark next to the active USB port shows the COM port which is used to communicate with arduino.

VI. IMPLEMENTATION

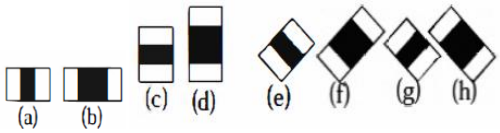
After a classifier is trained, it can be applied to a region of interest (of the same size as used during the training) in an input image. The classifier output is "1" if the region is likely to show the face and "0" otherwise. To search for the object in the whole image one can move the search window across the image and check every location using the classifier. Here we use two different codes for face detection and tracking respectively. The algorithm used for both the codes (Processing & Arduino) is detailed in this section.

Extended Haar-like Features

1. Edge Features



2. Line Features



3. Centre-Surround Features

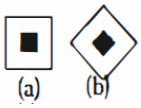


Figure 8: The Extended Rectangle Features

6.1. Implementation of Software

Processing takes the video input from the webcam and uses the OpenCV library to analyze the video. If a face is detected in the video, the OpenCV library will give the Processing sketch the coordinates of the face. The processing sketch will determine where the face is located in the frame, relative to the centre of the frame, and send this data through a serial connection to an Arduino. The Arduino will use the data from the Processing sketch to move the servos connected the Servo setup as shown in figure 9.

a) Basically haar-cascade classifier is used for detecting the faces.
b) The input video frame is read from camera and temporary memory storage is created to store this frame.
c) A window is created to capture the display frame and frame is continuously monitored for its existence.
d) A function is called to detect the face where the frame is passed as parameter.
e) Steps b-d is kept in a continuous loop until the user defined key is pressed.
f) The classifier, frame, memory storage & the window are destroyed.
g) The (X, Y) coordinate of the image is plotted according to movement of face.
h) The difference between face position and centre is calculated and sent to Arduino serially

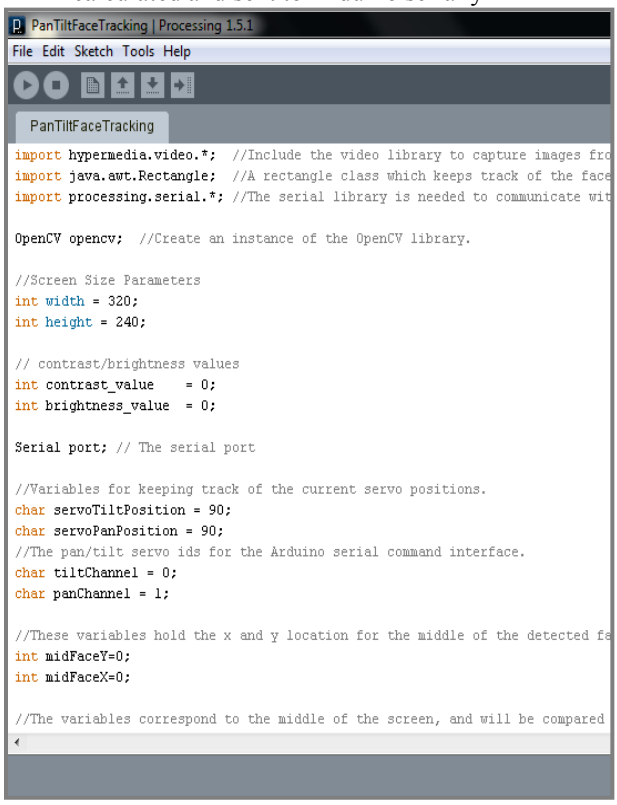


Figure 9: Processing Window with the Code

6.2. Implementation of Hardware

Basically Arduino will analyze a serial input for commands and set the servo positions accordingly. A command consists of two bytes: a servo ID and a servo position. If the Arduino receives a servo ID, then it waits for another serial byte and then assigns the received position value to the servo identified by the servo ID. The Arduino Servo library is used to easily control the pan and tilt servos. There's a character variable that will be used to keep track of the characters that come in on the Serial port.

a) Library named servo.h is used in arduino to control the servo motors, based on the data obtained by the openCV through COM port.
b) Depending on the difference found in step8 the 2 servo motors are sent with appropriate controls for the pan-tilt movement of camera.
c) Step b is kept in a continuous loop.

VII. RESULT AND ANALYSIS

The image of the face captured by web-cam with the help of Processing, OpenCV undergoes different steps as mentioned below.

Generate rectangle class which keeps track of the face coordinates. Create an instance of the OpenCV library. This serial library is needed to communicate with the Arduino. Adjust Screen Size Parameters on contrast/brightness values. Convert the image coming from webcam to greyscale format. Find out if any faces were detected. If a face is found, find the midpoint of the first face in the frame. Manipulate these values to find the midpoint of the rectangle.

Find out if the Y component of the face is below the middle of the screen, if it is below the middle of the screen. Update the tilt position variable to lower the tilt servo. Find out if the Y component of the face is above the middle of the screen. Find out if the X component of the face is to the left of the middle of the screen. Update the pan position variable to move the servo to the left. Find out if the X component of the face is to the right of the middle of the screen. Update the pan position variable to move the servo to the right. Update the servo positions by sending the serial command to the Arduino. The pan & tilt position of the servo motor linked with web camera is directly proportional to the serial command of the coordinates to the Arduino of the X & Y components of the face from midpoint of the rectangle. Figure 10 shows the result of the face detection and figure 11 shows the face detection as well as tracking.

By using this approach it was found that time taken to detect the face was less than 1 second which means that this setup can be used in real time. The detection efficiency was greatly improved by using OpenCV. The average frame rate was found to be 15 fps.

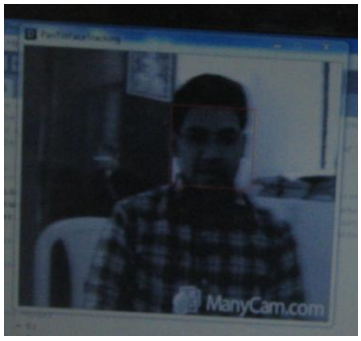


Figure 9: Output of Algorithm Showing the Face Detection

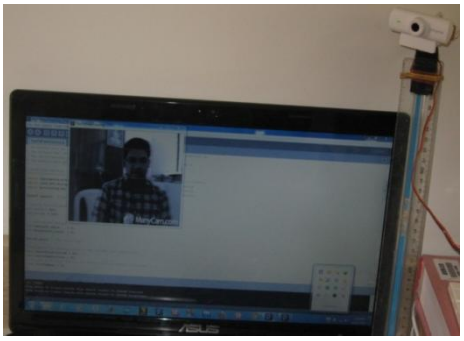


Figure 10: Face Detection with Camera

VIII. CONCLUSION

Prototype system for automatic face detection and tracking is successfully implemented and tested. The test results show that the detection method used in the paper can accurately detect and trace human face in real time. This paper shows the intersection of Image processing and embedded systems, by using openCV and arduino real time implementation is possible. Future Work: Along with face detection, face recognition may also be implemented.

REFERENCES

[1] A. Faizi (2008), "Robust Face Detection using Template Matching Algorithm,", *University of Toronto*, Canada.

[2] P. Feng (2004), "Face Recognition based on Elastic Template,", *Beijing University of Technology*, China.

[3] L.H. Liang, H.ZH. Ai & G.Y. Xu (2002), "A Survey of Human Face Detection," *J.Computers. China*, Vol. 25, Pp. 1–10.

[4] K.J. Wang, SH.L. Duan & W.X. Feng (2008), "A Survey of Face Recognition using Single Training Sample", *Pattern Recognition and Artificial Intelligence*, China, Vol. 21, Pp. 635–642.

[5] Z. Zhang (2008), "Implementation and Research of Embedded Face Detection using Adaboost", *Shanghai JiaoTong University*, China.

[6] L. Guo & Q.G. Wang (2009), "Research of Face Detection based on Adaboost Algorithm and OpenCV Implementation", *J. Harbin University of Sci. and Tech.*, China, Vol. 14, Pp. 123–126.

[7] CH. Y. Lu, CH.SH. Zhang & F. Wen (1999), "Regional Feature based Fast Human Face Detection", *J. Tsinghua Univ. (Sci. and Tech.)*, China, Vol. 39, Pp. 101–105.

[8] H. J. Jiang (2007), "Research on Household Anti-Theft System based on Face Recognition Technology", *Nanjing University of Aeronautics and Astronautics*, China.

[9] P. Viola & M. Jones (2001), "Rapid Object Detection using a Boosted Cascade of Simple Feature", *Conference on Computer Vision and Pattern Recognition. IEEE Press*, Pp. 511–518.

[10] G. Bradski & A. Kaebler (2009), "Learning OpenCV", *China: Southeast Univ. Press*.

[11] Ben Fry & Casey Reas (2007), "Processing: A Programming Handbook for Visual Designers and Artists", *MIT*.

[12] Open Source Computer Vision Library Reference Manual-intel

[13] Gary Bradski & Adrian Kaehler O'Reilly (2008), "Learning OpenCV", *O'REILLY Media*.

[14] "Introduction to OpenCV", [Online] Available: www.cse.iitm.ac.in/~sdas/courses/CV_DIP/PDF/INTRO_C

[15] "DCRP Review: Canon PowerShot S5 IS", Available: http://www.dcresource.com/reviews/canon/powershot_s5-review/

[16] "Energy Conservation", Available: http://portal.acm.org.offcampus.lib.washington.edu/citation.cfm?

[17] Sarala A. Dabhade & Mrunal S. Bewoor (2012), "Real Time Face Detection and Recognition using Haar - based Cascade Classifier and Principal Component Analysis", *International Journal of Computer Science and Management Research*, Vol. 1, No. 1.

[18] Faizan Ahmad, Aaima Najam & Zeeshan Ahmed (2013), "Image-based Face Detection and Recognition: State of the Art", *IJCSI International Journal of Computer Science Issues*, Vol. 9, Issue. 6, No. 1.

[19] Hussein Rady (2011), "Face Recognition using Principle Component Analysis with Different Distance Classifiers", *IJCSNS International Journal of Computer Science and Network Security*, Vol. 11, No. 10, Pp. 134–144.

[20] S. Sankarakumar, Dr.A. Kumaravel & Dr.S.R. Suresh (2013), "Face Detection through Fuzzy Grammar", *International Journal of Advanced Research in Computer Science and Software Engineering*, Vol. 3, No. 2.

S.V. Viraktamath is with SDMCET, Dharwad, Karnataka, India. He is serving as Assistant Professor in the Department of E&CE. He has received a gold medal from VTU Belgaum for securing first rank in M.Tech (DC&N). His research interests include error control coding, wireless communication and networking. Presently he is pursuing Ph.D from VTU Belgaum. He has more than 25 publications to his credit.

Mukund Katti is recently graduated from department of E&CE, SDMCET, Dharwad. His field of research and interest includes Robotics, Image processing and Error control coding.

Pavan Kulkarni is recently graduated from department of E&CE, SDMCET, Dharwad. Their field of research and interest includes Image processing and Error control coding.

Aditya Khatawkar is recently graduated from department of E&CE, SDMCET, Dharwad. Their field of research and interest includes Image processing and Error control coding.