A PROJECT REPORT

ON

Heart Beat Rate Monitoring

Submitted to

KIIT Deemed to be University

In Partial Fulfilment of the Requirement for the Award
BACHELOR'S DEGREE IN ELECTRONICS & COMPUTER SCIENCE ENGINEERING

By

SUDHANSU KUMAR MAHARANA - 1730196

SATYAKI CHAKRABORTI - 1730182

MADHAB JYOTI MOHANTY – 1730163

SONU KUMAR - 1730191

UNDER THE GUIDANCE OF
PROF: PRAVIN KUMAR SAMANTA

SCHOOL OF ELECTRONICS ENGINEERING
KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY
BHUBANESWAR, ODISHA – 751024
APRIL 2021

KIIT Deemed to be University
School of Electronics Engineering
Bhubaneswar, ODISHA 751024

CERTIFICATE

This is to certify that the project entitled

"Heart Beat Rate Monitoring"

Submitted by:

SUDHANSU KUMAR MAHARANA – 1730196
SATYAKI CHAKRABORTI - 1730182
MADHAB JYOTI MOHANTY – 1730163
SONU KUMAR -1730191

is a record of Bonafide work carried out by them, in the partial fulfilment of the requirement for the award of Degree of Bachelor of Engineering (Electronics & Computer Science Engineering) at KIIT Deemed to be university, Bhubaneswar. This work is done during year 2020-2021, under our guidance.
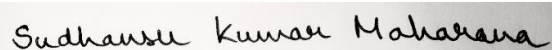
Date: 13 / 04 / 2021

PROF. PRAVIN KUMAR SAMANTA
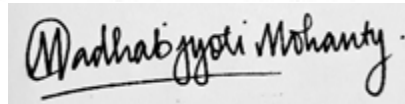**Project Guide**

ACKNOWLEDGEMENTS
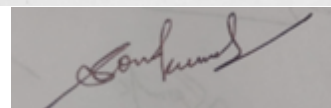
SUDHANSU KUMAR MAHARANA

SATYAKI CHAKRABORTI

MADHAB JYOTI MOHANTY

SONU KUMAR

ABSTRACT

During this age of innovation and technology, people's life is getting easier as new gadgets and devices are coming out with the capability complementing their work lives. There are gadgets that can monitor their health parameters constantly irrespective of time or location. Though the biggest setback would be the price. Majority of deaths in the world happens due to cardiovascular diseases (CVD). As soon one can know their health parameters the better. With the increase in population and workload, people ignore their health most of the time because of constraints on time and place. One of the examples of CVD would be Atrial Fibrillation (a form of Arrhythmia) where the tempo or the time between each heartbeat will vary. The demand for easy and affordable health care is increasing rapidly and especially in remote locations. In this era, the cost of basic health monitoring devices have decreased and the communication gap between the doctors or medical personnel and patients is also decreasing. So, in this report, the capability of fast and easy access through web page and Python are implemented together as optimization techniques to reduce error and make things platform-independent to monitor the patient in real-time.

Keywords*: Heart, Heartbeat, Cardiovascular diseases (CVD), Python.*

CONTENTS

List of Figures

List of Tables

List of Symbols / Abbreviations

Symbol / Abbreviation	Description
CVD	Cardio Vascular Disease
HRM	Heart Rate per Minute
BP	Blood Pressure
IoT	Internet of Things
GSM	Global System for Mobile Communications
VLSI	Very Large Scale Integration
BRISK	Binary Robust Invariant Scalable Key-points
OpenCV	Open Source Computer Vision
LBPH	Local Binary Pattern Histogram
CME	Coronal Mass Ejections
GUI	Graphical User Interface
HTML	Hyper Text Mark-up Language
CSS	Cascading Style Sheets
RGB / RBG	Red Green Blue Colour code format
UI	User Interface
GPU	Graphical Processing Unit
CPU	Central Processing Unit
FPS	Frames per Second
vidP	Video Path
vidT	Video Time
BPM	Beats Per Minute
μ_{ref}	Median
μ_i	Median of ith batch
δ	Deviation

List of Algorithms

List of Equations

I. INTRODUCTION

From smallest beings like ants to the largest beings like whales, heart is the most important organ, it has the responsibility for pumping blood through the blood vessels of the circulatory system. Oxygen and nutrients that are used for metabolic waste removal which is a very vital process for our survival which is carried out using blood. The average heartbeat of a healthy person is close to 72 heart beats per minute at resting state. The beats per minute can go down to 30 – 40 beats per minute for well-trained athletes. So the heart rate increases while exercising and if a person stays persistent in this their heart beat decreases at resting state which is a very healthy sign. Now as for how the heartbeat rhythm while pumping of the blood came to be, it is because of the pace making cells present in the sinoatrial node.

The most well-known heart diseases include coronary artery disease, congenital heart defects (diseases which are obtained during birth), arrhythmias (heart rhythm issue) and many more. These are just some the examples but there are range of conditions which affects a person's heart are described by heart diseases.

Life style of people has changed with time, as compared to today the lifestyle used to be straight forward and had abundant amount of physical labour, stress was less and the diet used to be fixed, pure and not as toxic as it is at present. Which resulted in not many people from that era not suffering from major heart diseases.

But if we look into the present condition, cardiovascular diseases are the biggest threat and is responsible of 31% of all global deaths as reported in 2016 which is estimated to be 17.2 million deaths due to CVDs. Good health care requires decent financial sums which is a big factor and in India a big set-back at preventing deaths.

At present if we look into people's work culture, it is very complex, fast paced and very taxing to both the body and the mind and as for the environment it is very unhealthy and polluted. The food consumption at present is pretty toxic and with all these factors combined make things very stressful for most of the people. So with these we encounter an increase in heart diseases complimented with other diseases such as diabetes, hypertension, blood pressure, etc. which are also known to be lifestyle diseases.

As the saying goes, "Better late than never", the symptoms of heart disease or any disease should not be ignored however minor it would be. The symptoms could be very subtle and are often neglected. So if theses abnormalities are detected in the early stages then the dire situation would never come to be.

So as we move forward in life, newer technologies are introduced and are becoming a part of everyday life. Faster, cheaper and devices with better accuracy are being produced. There are machines to monitor every aspect of a human body. These are very easy to get our hands on as they could be found at most of the pharmaceutical stores. But in case of rural areas these devices are hard to find and could be expensive for most of the people. Financially stable people can monitor their health regularly and don't need to visit the hospital that often but in case people who are not very economically stable have the issue of not being able to visit the hospitals nor monitor their health because of their busy lives. This is also a problem for older people who are unable to visit medical facilities very often which could cause major problems in case of emergencies.

II.   PREVIOUS WORKS

The literature survey section is mostly dependent on the unique content of significant papers which remarkably proved helpful in creating a new device capable of measuring heartbeat using camera sensors and image processing. An alert signal is returned with the help of Arduino whenever the heart rate is not within the predefined safety range as described. Infrared sensors are used to track the heart beat rate [1], [2]. This paper discussed about how we can transfer data of heart related parameters to a smartphone wirelessly using Raspberry Pi [3]. In this paper a prototype has been developed to measure the body temperature, blood pressure (BP), pulse rate, electrocardiogram (ECG) and glucose level of the person / patient, this was done using multiple sensors, this is also capable of tracking the location of the patient and all the data are stored and processed in the micro-processor [4], [5]. Here the concept of Internet of Things (IoT) is used for PPG based heartbeat monitoring. In this paper a wireless heartbeat monitoring system in made and integrated with GSM technology. This has the potential to be an integral part of a suite of personal healthcare equipment as described in [6], [7], and [8]. Parameters like temperature of the body, sweat rate, heart beat rate are measured and transferred to the computer so that the health condition of the person could be monitored remotely and transmit the data to the doctor by using GSM technology in [9]. To reduce the death rate of humans because of heart disease, the solution presented here is to periodically monitor the patient's heart parameter using biosensors wirelessly and also integrating GSM technology in [10], [11]. This paper describes a system that monitors the heartbeat and body temperature using multiple sensors, of the patient by using their fingertip, this system is also capable of tracking the location of the patient through GSM technology [12], [13]. For this paper there are three factors that affect the performance are environmental variants, sensor variants and most importantly individual variants, the details have been taken through biometric identification and verification, which had been done using IoT (Internet of Things) as shown in [14]. Here, the heartbeat, along with the chances of heart attack with respect to the age groups have different ranges for maximum and minimum values of the heartrate by using hardware system that consists of Node MCU and pulse sensor integrated with IoT based system in [15]. Raspberry Pi 3 model is used here to create an efficient and cost-effective continuous heartbeat monitoring system integrated along with IoT (Internet of Things) in order to transfer data over the internet without the aid of any person or computer interaction in [16]. A system that can be used remotely, is used for sensing the parameter of the human body which consists of pulse and temperature which is tracked constantly for the health status of a patient through web-based technology in [17]. This system uses PPG (Photo Plethysmography), ATMEGA328PU Microcontroller with PPG techniques and GSM modem that is used for sensing the heart beat and all the variations in the heartbeat and all the data are stored using IoT in cloud as described in [18]. Here a portable remote chest strap and bracelet are used as wrist wearables which provide various heartbeat sensors and currently new sensor devices were implemented using the pulse rate monitor, Raspberry Pi 3, Arduino UNO and think speak cloud provides quick service to the patients if the doctor was a long distance away or in case of being monitored by the guardian in [19] [20]. In this case various sensors were used from getting the real-time heart condition of the patients and with the usage of IoT, Raspberry Pi kit technologies were used. So by using these technologies health officer could easily track the patient [21], [22]. This paper uses GSM and Very Large-Scale Integration (VLSI) technologies in for monitoring patient's physiological characteristics, like heartbeat rate, temperature, etc. with the help of sensors which give the input as an Analog data, which is processed and stored in the AT89S52 microcontroller and output returned in digital format, this process is done on regular intervals as described in [23], [24]. IR base heartbeat sensor, Arduino Uno were used to the screen physical parameters like heartbeat, temperature, BP,

ECG, etc. Through web application to avoid a financial hit the data is sent directly to the professionals from the patient's side [25], [26].

This paper is all about how the feature detectors and feature matchers available in OpenCV are implemented. The result was that Speeded-Up Robust Features (SURF) detector helps in detecting the maximum number of features whereas for an image pair, Brute Force (BF) matcher detected the maximum number of features after comparing both images. Further ahead SURF was successful in detecting 1907.2 features on an average in 1538.61ms and BF was successful in matching features in 160.24ms on average. The best pair of tools of OpenCV came out to be BF matcher and Binary Robust Invariant Scalable Key-points (BRISK) detector with an average detection rate of 80.2 and 1132 features respectively in 265.67ms. They thereby concluded that SURF is better if number of features are crucial and BF matcher is preferable if number of features matching is of utmost importance [27].

This paper is about a robotic mechanism that can sense different objects based on colour characteristics and perform different operations on them like object sorting and object grabbing using an integrated three-axis truss manipulator structure. They used 'Machine vision' and OpenCV to perform this intelligent recognition and intelligent handling-based research [28].

Here they proposed about real-time image processing methods using computer vision libraries. Here they discuss about Gaussian filter and how they are really effective in reducing noise of target edges with real time edge detection and maximizes the accuracy of detection by enhancing Sobel edge detection algorithm. Image processing mobile port and machine vision technology is also implemented with Linux OS into smartphones. They proposed an algorithm that improved detection accuracy of edges and fulfilled fast detection. They started by setting a camera monitor and loading weight parameters and configuring files onto the system. Then they headed up for model initiation which eventually made the model work fine. It processes the frame it takes by sending this image to a network model. There it does Gauss filtering, Sobel edge detection and apply Canny edge detection which draws the target frame and extracts the final target [29].

This paper focused on detecting coffee rust infections in coffee plantations. They researched by training 2 neural networks which could detect the presence of coffee rust using NVIDIA Digits and OpenCV open-source libraries to compare between the two detections and thereby calculating the precision percentage [30].

Here the authors have proposed a way of detecting helmet on motorcyclist. The proposed system is based on Tensorflow and Keras along with Computer Vision. This system helps in recognizing if the rider is wearing helmet or not in real time. If the system detects that the motorcyclist is not wearing helmet, it will declare rule violations [31].

This paper proposed to design an application for facilitating engineering students in the field of computer vision, by making the learning process efficient and easier and a lot more dynamic. They have developed this application for Android OS by which it will be a lot easier for the students to learn concepts that could have been difficult otherwise to learn. Their objective is to observe real time effects generated on images using image processing algorithms included in open-source libraries of OpenCV [32].

Here the authors researched over vehicle detection to find a solution of traffic monitoring and controlling traffic to mitigate the problem of overcrowding especially in larger cities. This project aimed to detect the movement of vehicles and observer count of vehicles through different categories. The system proposed here is applicable for mainly daylight conditions. The detection speed is

enhanced by distributing the processing load parallelly in different threads for image processing, motion detection, control and display. The opposite traffic direction and sidewalks are filtered to avoid processing in those areas [33].

This paper is based on student counting in a classroom and deals with the problem of students bunking or not attending lectures. This system will reduce the tedious task carried by management manually. OpenCV is used to count number students present in a class and also helps in giving best occupancy ratio. It will help in identifying the person in the image, and count the number of students. The count will be stored into firebase which will be also used for prediction at a later stage. The prediction of occupancy ratio will help reducing the wastage of space and efficiently will help in planning schedule. Few factors are taken under dynamic allocation like classroom size, course capacity, classrooms equipped with projectors. The whole data will be encapsulated under Firebase by Google for maintaining the record of personal data, attendance sheets etc. [34].

The authors through this paper, proposed the working of a humanoid robot using a python-based chessboard recognition method known as 'Go robots'. Few advanced functions like projection transformation is used to correct the image perceived and binarization to enhance the image for better and swift processing. Other methods like colour space transformation, high pass filtering, threshold transformation, Huff transform are used to detect as well as locate segment chess pieces in a chess game. It came out to be very accurate and has a high recognition rate [35]

This paper proposed an automated attendance system. This system is based on the concept of face detection and recognition through the field of open computer vision (OpenCV). This is also concentrated on the issue of traditional attendance system and to avoid the misuse of assets and time, the attendance system is switched from a standard attendance system to a digital system. It is mostly proposed to improve the adaptability and performance besides the tiring and long-term time load and work. It also has the ability perform and manipulate attendance notes of individuals and automatic calculation of number of present and absentees. Here they are using face recognition with haarcascade and Local Binary Pattern Histogram (LBPH) model to finally generate spreadsheet of number of students present [36].

The authors proposed a way to detect any suspicious behaviour in public areas that could be dangerous and may cause heavy casualties. The proposed system concentrates on recognizing suspicious activities carried out in public and target the activities automatically which can be achieved using computer vision. This system also works at par with motion influence map which represent motion analysis and frequent changes in position by using pixel level presentation [37].

Through this paper authors proposed the way of tracking sunspots using OpenCV. This research was made as coronal mass ejections (CME) could negatively affect communication and navigation as it disturbs the whole magnetic field of earth and interfere with wave signals around earth. This uses continuum images of Sun for identification of sunspots that could lead to Coronal Mass Ejection and track them. This system tracks those sunspots over video. It also uses Canny Edge Detection that is capable of identifying edges of sunspots. This system uses multi object tracker function that is capable of tracking multiple sunspots [38].

Articulating all researchers past work about health monitoring through various Technologies, they had not used optimizing technologies and used hardware which could be expensive. to perform proper monitoring of the patient.

III. System Design

Data Flow Diagram
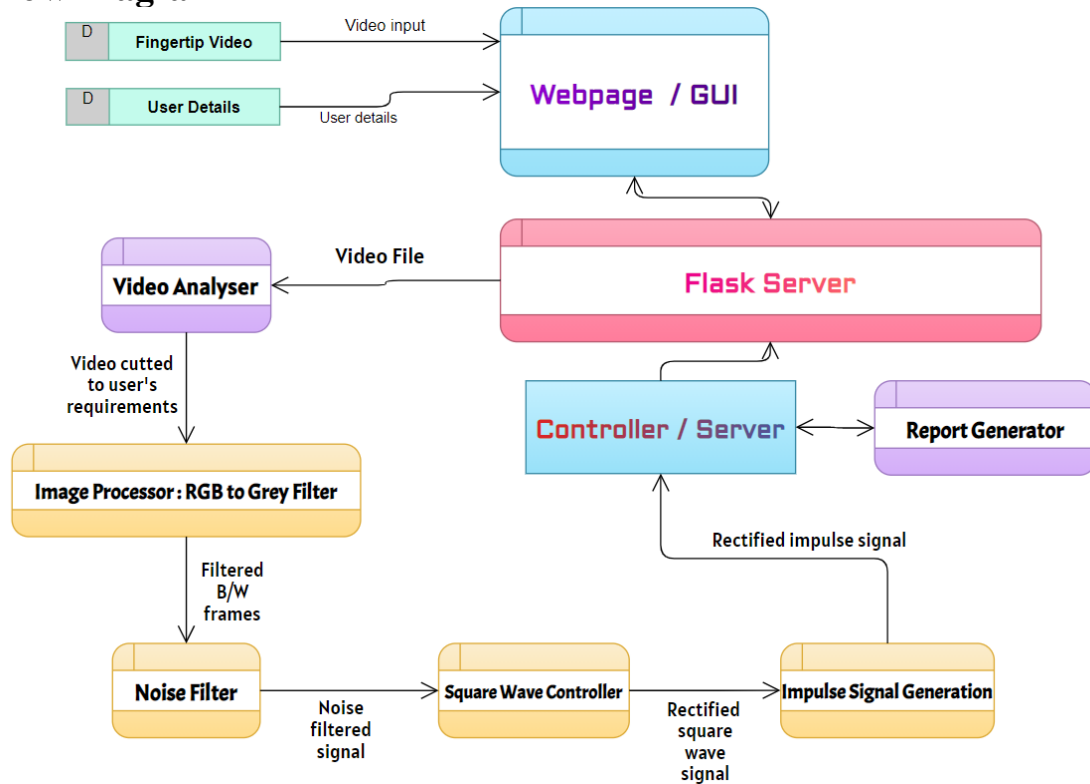
Figure 1 Data Flow Diagram

As seen in the above diagram Flask server is the main component that connects all the parts together. It connects and renders the HTML pages. The data returned from the HTML pages are stored in Flask application file and sent to the video analyzer in the HRM method. This processes, filterers, rectify and generate the output signal. This output is returned to Flask application, which then passes it to the final HTML page.

The complete flow of the system is shown in the following image.
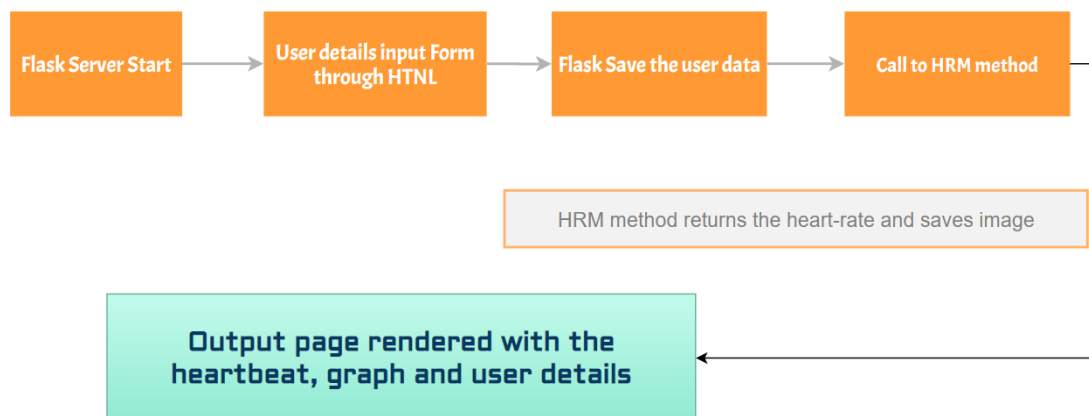
Figure 2 Flow Diagram

Use Case Diagram
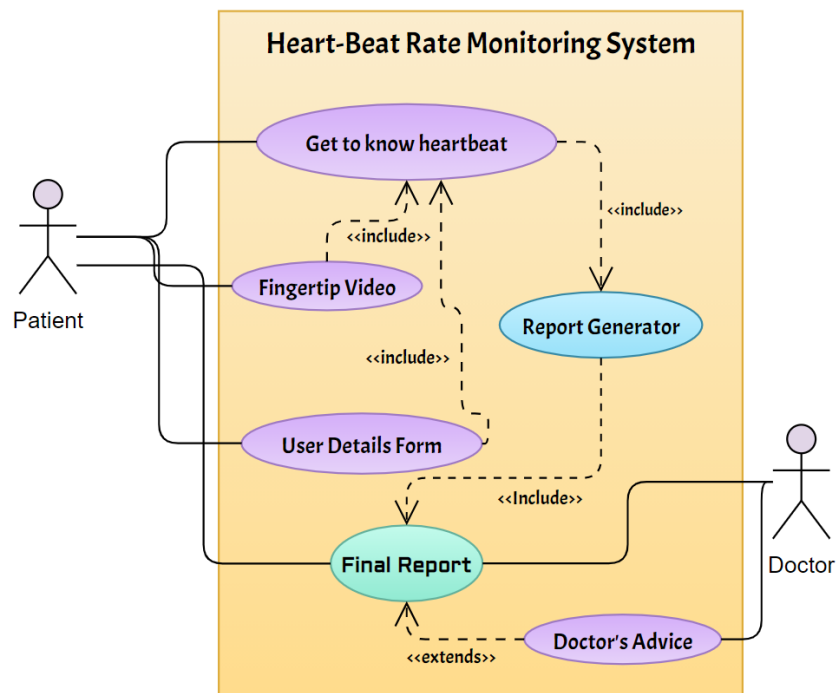
Figure 3 Use Case Diagram

IV. Webpage / GUI

Figure 4 Index / Homepage Layout

Figure 5 User Detail / Output Page Layout

V. PROCEDURE

The goal of our model is to minimize the use of hardware, reduce the cost and most importantly omitting the communication gap between the patient and the medical personnel. This model's frontend has 2 web pages, one is for data collection and other is for presenting the data along with heart beat rate report.

Step – 1: First thing the user will interact with is the data registration page which is made using HTML and CSS. In this web page the user will enter the required infor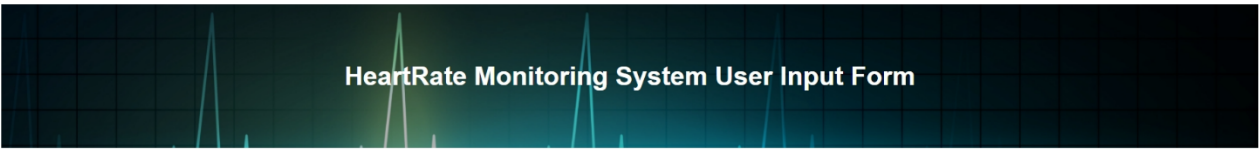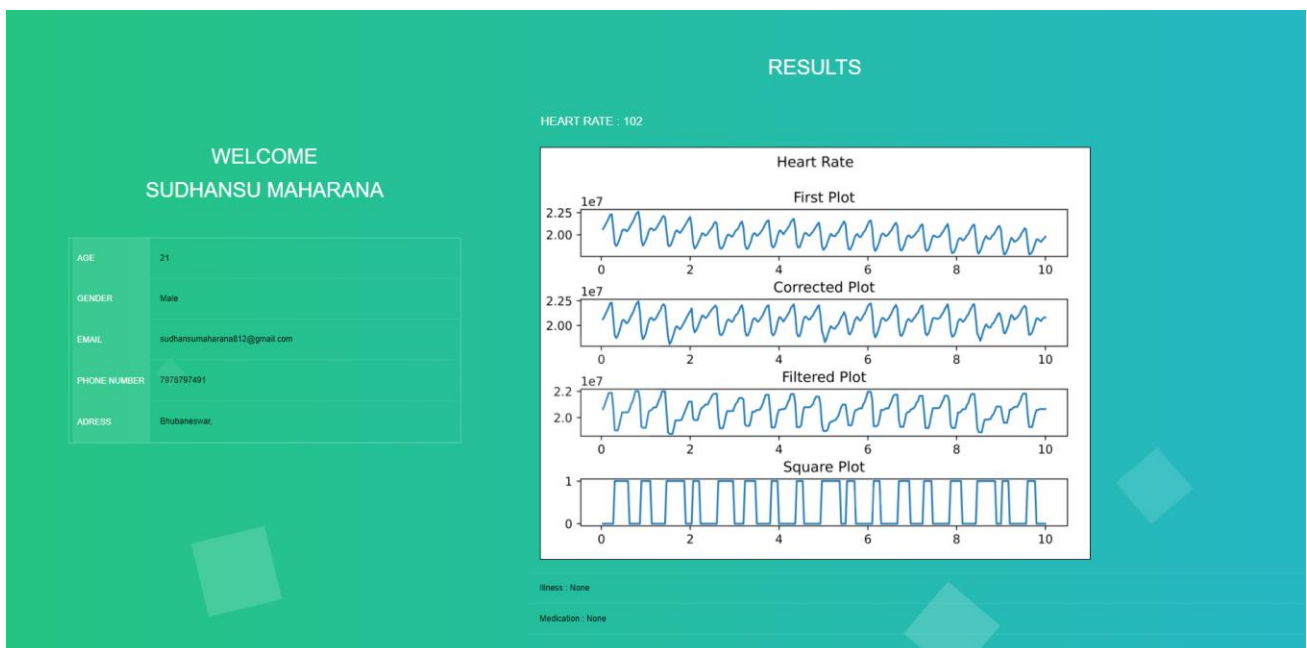mation namely – First & Last Name, email, phone number, gender, address, Illness (optional), Additional details (optional) and most importantly the video path and time of the video that is to be processed, these details are sent to the python script, this is done with the help of flask which is used to connect/route HTML pages along with python scripts.

Step – 2: After getting the video address and time, these parameters are sent to the python script which will calculate and return the heart beat rate and the plot.

Step – 3: With the video path, the video is accessed and stored in a variable using OpenCV (Open Source Computer Vision Library).

Step – 4: Then the video is checked if it is same or greater than the time provided by the user for processing. If true then the video is trimmed accordingly and the trimmed video is used further for processing. Else if it turns out to be false the script ends after throwing error message to the user.

Step – 5: Now that the foundation is made, the parameters of the trimmed video are captured such as the number of frames, frames per second and total length.

Step – 6: In this step the video is converted from RBG colourized to Greyscaled. This is done because the data required is the intensity of light in each frame, and the change in intensity depicts the heart beat rate. In Python a greyscaled frame is stored in form of a 2-D matrix of dimensions same as resolution of the video. In this matrix each cell value is the white value of that pixel ranging from 0 to 255.

Step – 7: Here the greyscaled data is iterated and sum of all the cells of one frame are stored in an array which will be of size same as the number of frames the trimmed videos has.

Step – 8: With the data we had right now a graph is sub-plotted, in this plot the dipping values represent the heartbeats. As shown in Step – 6, whenever the white light values dips, shows that at that moment the heart pumped blood. Whenever blood passes through the arteries the light passing through the finger will flicker/dim down.

Step – 9: Due to environmental noise or camera adjusting light, the graph could appear to be tilting along the y – axis. In order to stop this the graph is straightened out with respect to the y – axis.

Step – 10: Another sub plot is plotted to represent the outcome of Step – 9's algorithm.

Step – 11: Now to remove further noise from the graph, it is filtered using median filter algorithm with a factor of 9.

Step – 12: Another sub plot is plotted to represent the outcome of step – 11.

Step – 13: Now that we have the necessary data to calculate the heart beat rate, in order to calculate the plot is converted to a square wave.

Step – 14: Another sub plot is plotted to represent the outcome of step – 13.

Step – 15: From this square wave the number of changes are calculate, i.e. whenever value changes from 0 to 1 or 1 to 0 counter is incremented.

Step – 16: The counter in step – 15, is now halved and ceiling value is considered. With the video time a multiplying factor is calculated.

Step – 17: The counter is multiplied with the factor which gives the heart beat per minute.

Step – 18: These values are returned along with the complete graph.

Step – 19: Finally the user detail HTML page is rendered through Flask where all the details with the plot and heart beat rate per minute is displayed.

VI. Our Proposed Methodology

This system is made with the combination of HTML, CSS as the UI, Python OpenCV for image processing, and Python Flask for connecting the algorithm with the main Python script.

Folder structure:

Table 1 Folder Structure

/static/style.css
/static/user_style.css
/template/index.html
/template/user.html
/main.py
/script.py

Part – 1: Flask

From the above folder structure main.py is the python script that has Flask functionalities. So, Flask is a micro web framework that is written using Python. This is a template engine which helps us to compile modules, packages, libraries that helps us in writing the web application.

```python
from flask import Flask
from flask import request
from flask import escape
from flask import render_template
from script import *

app = Flask(__name__, static_folder="C:\Users\KIIT\\Desktop\\Final\\static")

valDict = {'age' : '','first_name' : '', 'last_name' : '', 'email' : '', 'phone_number' : '',
    'gender' : '' ,'address' : '','city' : '','region' : '','postal' : '' ,
     'vidPath' : '', 'country' : '','vidTime'  : '','illness' : '','misc_msg' : ''}

@app.route("/")
def index():
    valDict['first_name'] = str(escape(request.args.get('first_name', '')))
    valDict['last_name'] = str(escape(request.args.get('last_name', '')))
    valDict['email'] = str(escape(request.args.get('email', '')))
    valDict['phone_number'] = str(escape(request.args.get('phone_number', '')))
    valDict['address'] = str(escape(request.args.get('address', '')))
    valDict['city'] = str(escape(request.args.get('city', '')))
    valDict['region'] = str(escape(request.args.get('region', '')))
    valDict['postal'] = str(escape(request.args.get('postal', '')))
    valDict['country'] = str(escape(request.args.get('country', '')))
    valDict['illness'] = str(escape(request.args.get('illness', '')))
    valDict['misc_msg'] = str(escape(request.args.get('misc_msg', '')))
    valDict['age'] = str(escape(request.args.get('age', '')))
    valDict['gender'] = str(escape(request.args.get('gender', '')))
    valDict['vidPath'] = str(escape(request.args.get('vidPath', '')))
    valDict['vidTime'] = escape(request.args.get('vidTime',type=int))
    return render_template("index.html")
```

```
@app.route("/user")
def user():
    vidTime = int(valDict['vidTime'])
    hrm = hrtRate(valDict['vidPath'],vidTime)
    name = valDict['first_name'] + ' ' + valDict['last_name']
    return render_template("user.html",name=name, age=valDict['age'],
    gender = valDict['gender'], email = valDict['email'], phone_number =
valDict['phone_number'],
    address = valDict['address'], city = valDict['city'], region = valDict['region'],
    postal = valDict['postal'], country = valDict['country'], illness = valDict['illness'],
    misc_msg = valDict['misc_msg'], vidTime = vidTime, hrm = hrm)



if __name__ == "__main__":
    app.run(debug=True)
```

Algorithm 1 Flask Code

In the above code, first thing here is to get the Flask constructor with __name__, and the static folder specifier. This specifier is used for telling the interpreter where the CSS and other static folders are. After that a dictionary is made which stores all the required data. Dictionary is used instead of an array or list because, as the amount of data increased, it will be easier to identify what is what from a developer's perspective.

After this route to the homepage (i.e. index.html) and output page (i.e. user.html) is made. Whenever the server is hosted locally, the first page to open with the IP will be the index page. The homepage consists of a form which will take in the user data along with the video path and video time that are to be processed. After all the data are saved after the using the save button, they are stored into the dictionary as shown in the above code. When submitted the page will redirect the user to the output page.

The output page (i.e. user.html) calls the main python script which is described in Part – 2. The python script returns the heartbeat per minute and stores the output graph locally. The heart beat rate per minute is rendered along with all the user data in the output page.

This links two HTML pages (i.e. the index.html / homepage and user.html / output page) with the Python script in the back end.

Part – 2: OpenCV

In this part OpenCV takes a major part as in order to calculate the heartbeat rate, the system needs to process and filter the video accordingly. So, what is OpenCV?

Open CV is a collection of programming functions primarily aimed at real-time computer vision. In the beginning, Open CV was originally developed by Intel, Later on, it was developed by willow garage followed by itself. This library is a cross-platform library that enables the user to use it for free under the guidelines of the open-source Apache 2 license. Open CV also features GPU acceleration for real-time operation. The OpenCV project was first launched by Intel, it was an Intel research initiative to advance CPU-intensive applications, which also includes projects such as real-time ray tracing and 3d display walls.

There are various application that OpenCV includes such as:

1.   2D and 3d feature tool kit

2.   Egomotion estimation

3.   Facial recognition system

4.   Gesture recognition

5.   Mobile robotics

6.   Object detection

7.   Segmentation and recognition etc.

In this model's instance OpenCV is used for image processing and filtering.

Part – 3: Main Heartrate Monitoring Algorithm

The following libraries / packages are used in this algorithm.

```
from cv2 import cv2
import numpy as np
import matplotlib.pyplot as plt
from scipy.signal import medfilt
import math
import sys
import os
from moviepy.video.io.ffmpeg_tools import ffmpeg_extract_subclip
```
Algorithm 2 Packages Used

Here, cv2 is the OpenCV package, numpy is used for better support of arrays, matrices and better mathematical support. Matplotlib is used for plotting the graphs, scipy is for filtering the data, math as the name suggests is for mathematical operations, sys & os are user for accessing the local directories. Moviepy is for editing the video in our case it used for trimming the video to the length we need.

Let's take a look at the input and how it is used.

Figure 6 Finger Data unfiltered

This is one frame of the input, in this image the fingertip is on top of a small light source, so what happens here is that the light flickers / dims down whenever blood passes through our veins. So whenever blood passes it represents a heartbeat. Here, we have to find the number of times the light intensity dimmed in the time span provided by the user and calculate the heart beat rate.

```
vidPath = vidP
vidObj = cv2.VideoCapture(vidPath)
nFrames = int(countFrames(vidPath))
fps = int(vidObj.get(cv2.CAP_PROP_FPS))
if fps == 31 or fps == 61:
    fps=fps-1
vidTime  = int(vidT)
print(vidTime)
totalLen = int(vidTime  * (fps + 1))
```

Algorithm 3 Video Input and data variable initialization

The above section shows all the necessary variables used, the video path (vidP) and video time (vidT) are passed as parameters to this method. Using the video path video object is instantiated with the help of OpenCV methods. Again using OpenCV frames per second (fps) and total length of the video are calculated. Also the fps is rectified using if – else, as for some videos the fps showed 1 higher than it should be, so if it ever happened the fps is decremented by 1.

After this the number of frames are calculated which is done using the following method.

```
def countFrames(videoPath):
    vid = cv2.VideoCapture(videoPath)
    nof = 0
    while True:
        ret, frame = vid.read()
        if not ret:
            break
        nof=nof+1
    vid.release()
    return nof
```

Algorithm 4 Method to Count the Number of Frames

This method returns the total number of frames the video object has. In this method the video path is passed as parameter again and a new object is created. After that the object is iterated till the video reaches its end and a counter is incremented each time it iterated. Finally the object is freed and the number of frames is returned.

Now that we have the basic data needed, the video is to be trimmed according to the time provided by the user. This is done in order to reduce the processing as processing the whole video instead of using just the time rage will cost more time and slow down the whole system.

```
newVidPath = 'trim.mp4'
fileExists = os.path.exists('trim.mp4')
if(fileExists):
    os.remove('trim.mp4')
ffmpeg_extract_subclip(vidPath, 0, 0 + vidTime,targetname = 'trim.mp4')
newVidObj = cv2.VideoCapture('trim.mp4')
for i in range(nFrames):
    tFrame[i] = (i+1) / fps
```

Algorithm 5 Video trimming algorithm

The above code first checks if there is any file named trim.mp4, if not then if creates one in the

specified path. The video is trimmed with respect to the parameters provided. In this case it is 0 to 0 + vidTime. Let's assume vidTime is 10 and the total length is 30 then the trimmed part will be of 0 to 10 seconds. After this the video parameters are taken of trimmed video and this will be used throughout the program. Now time period is calculated, this is done by using the formula $^1/_{frequency}$, so in this case fps is the frequency and time period for each frame is calculated.
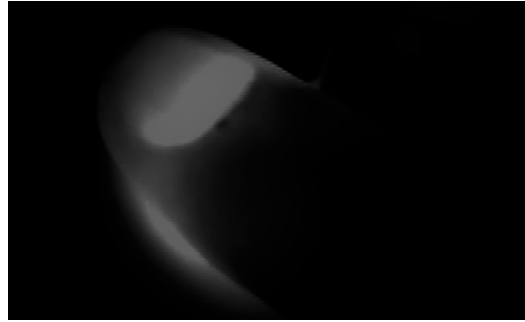

Figure 7 Finger Data filtered

In order to get this greyscaled result the following algorithm is used. Along with calculating the main data array for plotting and calculation the HRM.

```
i=0
while(True):
    ret, frame = newVidObj.read()
    if not ret:
        break
    gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
    gData[i] = np.sum(gray)
    i += 1
```
Algorithm 6 OpenCV RBG2GREY Filtering algorithm

Now the video is to be converted into greyscaled, as in colorized form the data it has is of no use. A video in Python is stored in matrix form, each frame has a matrix. So, the dimension of the matrix is same as the dimension of the video file, for example if the video is of 720p, then the dimension of the matrix will be 1366×720 cells. Each cell will store the data of each pixel. Now in colorized form the pixel data will be in hexadecimal format and will not tell us about the light intensity. So the video is converted to greyscaled. The difference in greyscaled video is that the data here ranges from 0 to 255. 0 means pitch black and 255 means completely white. So each cell of each frame in this case holds enough data for us to calculate the heart beat rate per minute. As shown in the above algorithm, the process is done using loops, each frame of the video is converted to greyscaled. Now we have a 0 initialized array which is of the size same as the number of frames the videos has. So with each iteration the sum of all the values in that are present in the frame matrix are summed up and stored into an index of the array.

Equation 1 Sum of all elements in a matrix

$$arr[i]\,(i\colon 0 \to number\ of\ frames) = \sum_{x=0}^{m}\sum_{y=0}^{n} greyData[x][y]$$

For example, let's assume one frame with a matrix of 3×3 as greyData, so sum will be.

Equation 2 Sum of all elements of a matrix example

$$
\begin{matrix} 2 & 2 & 3 \\ 2 & 4 & 4 \\ 3 & 4 & 5 \end{matrix} \implies arr[0] = (2 + 2 + 3) + (2 + 2 + 4) + (3 + 4 + 5) \implies arr[0] = 27
$$

But in case of real matrices the size is not this small each frame could have at least 300,000 cells. So in the algorithm and using the formula shown above, data of each and every frame are stored. Now, this much data will produce a basic graph. This graph is completely raw that is it could be tilted, contain noise, etc.
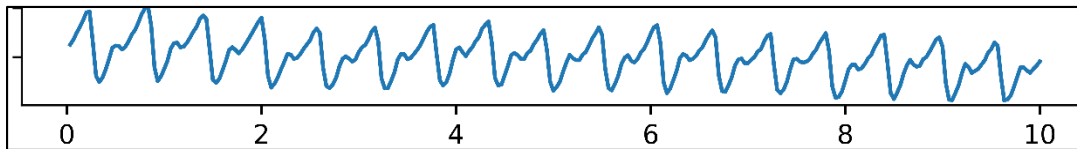

Figure 8 Unfiltered and Unrectified Graph

As it can be seen in the above graph, the plot is tilting downwards as it goes forwards towards x – axis.

In order to rectify the graph, it has to be straightened out for accurate results. If the data is sent further for processing without rectification, the heart beat rate will be erroneous. This is because in the next steps a threshold will be used to calculate the HRM, if the values are below the threshold it will be counted as a beat. But if the graph is tilted some values might not even appear below the threshold missing the beats and some could be counted as one beat. So in order to avoid the situation this algorithm is used.

```
framesPerBatch = int(0.5 * fps)
min_var = sys.maxsize * 2 + 1
max_var = -min_var - 1
k=0
while k < framesPerBatch:
   if rectifiedGraph[k] >= max_var:
      max_var = rectifiedGraph[k]
   elif rectifiedGraph[k] <= min_var:
      min_var = rectifiedGraph[k]
   k = k + 1
median = int((max_var + min_var)/2)
deviation = 0
i=0;   j=0;   k=0
rGraph = np.zeros(nFrames)
while i < nFrames:
   min_var = sys.maxsize * 2 + 1
   max_var = -min_var - 1
   k=i
   while k < i + framesPerBatch:
      if k >= nFrames:
         break
```

```
        if rectifiedGraph[k] >= max_var:
            max_var = rectifiedGraph[k]
        elif rectifiedGraph[k] <= min_var:
            min_var = rectifiedGraph[k]
        k = k + 1
    batch_median = int((max_var + min_var)/2)
    deviation = int(median - batch_median)
    while j < i + framesPerBatch:
        if j >= nFrames:
            break
        rGraph[j] = rectifiedGraph[j] + deviation
        j = j + 1
    i = i + framesPerBatch
```

Algorithm 7 Graph Rectification Algorithm

The data is rectified by taking batches. The first batch is taken for reference and the rest of the batches will be rectified with respect to the first reference batch. So, in the first batch a median is calculated, which is done by taking the peak and trough of the batch graph.

Equation 3 Median of the Batch

$$median\ (\mu_{ref}) = \frac{(peak + trough)}{2}$$

This median is used further for calculating the deviation throughout the rest of the batches. Now that we have the median, the next part is to go the next batch and calculate the median and deviation of this median with the median of the reference batch. Again the peak and trough of the next batch is calculated and median is calculated. Now the reference batch's median is subtracted with the current batch to get the deviation.

Equation 4 Deviation of each batch from the reference batch

$$deviation\ (\delta) = \mu_{ref} - \mu_i$$
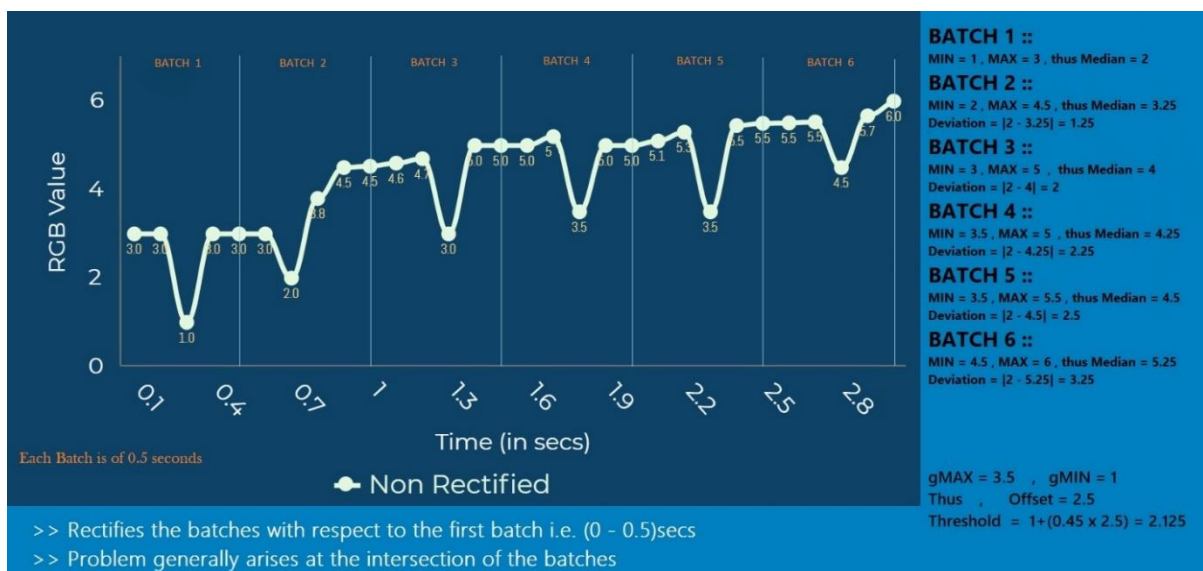
Figure 9 Example of rectification

The above graph calculates all the medians and deviations. Each bath is taken of 0.5 seconds, so in this case there are 6 batches including the reference batch. Now to rectify the deviation equation is used, whatever the deviation came out at the respective batch, it is added to each and every value of the batch. The results looks like the following graph. This graph has all the values along a straight axis.
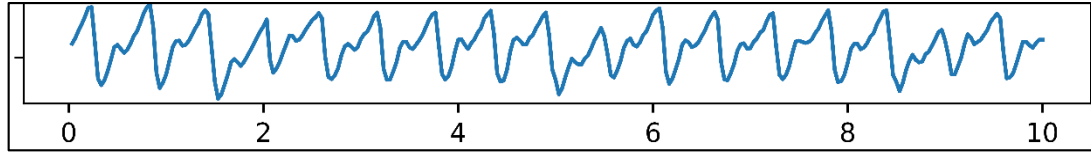
Figure 10 Rectified Data output
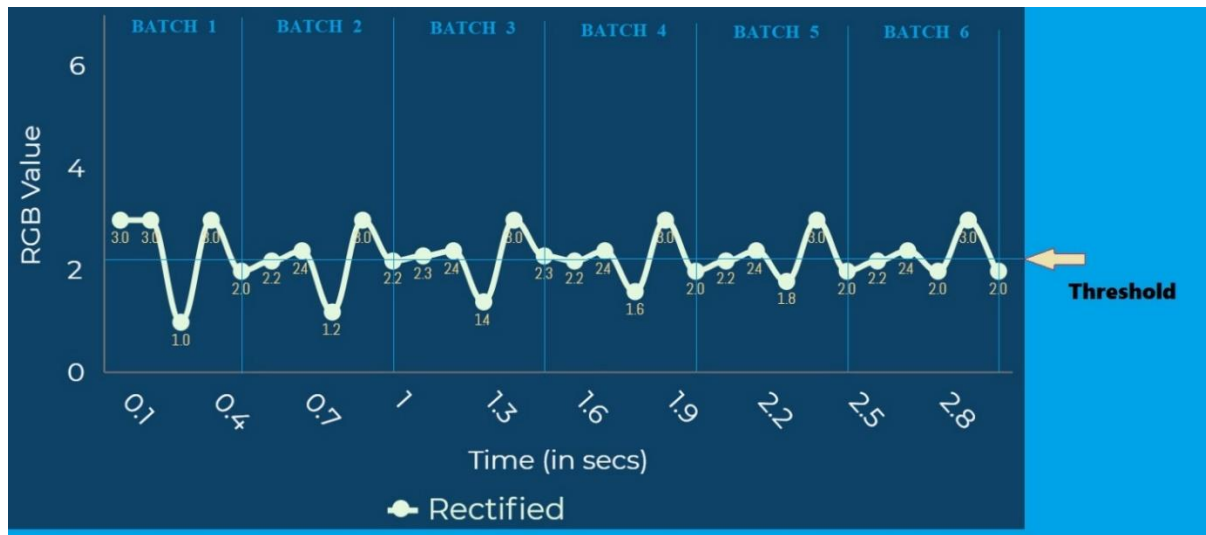
Figure 11 Example output of rectified data

Now that the graph is rectified and corrected, the other noises are to be corrected. For this median filter is used with a factor of 5.

Equation 5 Median Filter formula

$$fData\ (x,y) = \sum_{j=-1}^{1} \sum_{i=-1}^{1} 1 \times arr\ (x+i, y+j)$$

$$fData_{normalized}(x,y) = \frac{1}{\sum_{j=-1}^{1}\sum_{i=-1}^{1}} \sum_{j=-1}^{1} \sum_{i=-1}^{1} 1 \times arr(x+i, y+j)$$

Median filtering is a non – linear function used to remove noise from images.
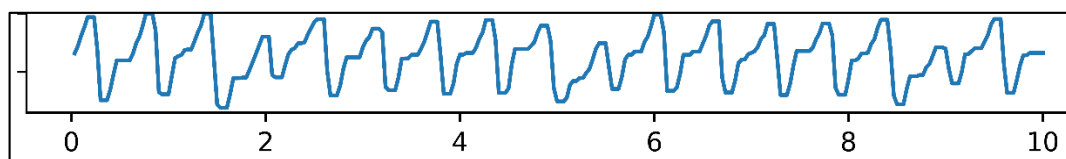
Figure 12 Filtered Data Output

Now that the graph is filtered, we can proceed to find the heartrate per minute. In order to do that the graph is now converted to square data. This is done by taking a threshold. This threshold is calculated by taking the peak and trough of the whole graph. An offset is calculated by subtracting peak and trough. After getting the offset threshold is calculated with the following equation.

To remove noise from images, a nonlinear method is used called median filtering. It removes noise from images effectively while still preserving the edges. Mainly effective for removing 'salt and pepper' type noise. Image pixel are traversed by the median filter one by one, changing each value of pixel with the median value of surrounding pixels. 'Window' is a type of pattern of surrounding pixels while slides over each pixel and finally over the entire image. The pixel values are sorted out in a numerical fashion for ease of calculation of median, which then helps in replacing the values of pixels with the median pixel value.

Equation 6 Offset and threshold formula

Offset = peak – trough

Threshold = trough + (0.55 * offset)

```
g_max = np.amax(fData)
g_min = np.amin(fData)
offset = int(g_max - g_min)
threshold = (g_min) + (0.55*offset)
sqData = medfilt(rGraph,5)
i=0
for i in range(nFrames):
    if sqData[i] <= threshold:
        sqData[i] = 1
    else:
        sqData[i] = 0
c=0
i=1
prev = sqData[0]
for i in range(nFrames):
    if sqData[i] != prev:
        c = c+1
        prev = sqData[i]
    else:
        prev = sqData[i]
```

Algorithm 8 Converting data to square wave
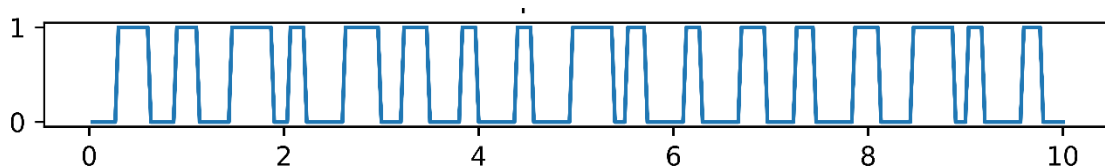
Figure 13 Square wave plot

In order to get the heart beat rate, so here everytime in this graph whenever 0 changes to 1 or 1 to 0 the counter increments.

```
def make_plot(axs, tFrame, data, j, s):
    plt.figure(num=None, figsize=(12, 8), dpi=1200, facecolor='w', edgecolor='k')
    axs[j].plot(tFrame,data)
    axs[j].set_title(s)


fig, axs = plt.subplots(4)
fig.tight_layout(h_pad = 2)
fig.suptitle('Heart Rate')
plt.subplots_adjust(top=0.85)
make_plot(axs, tFrame, gData,0, "First Plot")
make_plot(axs, tFrame, rGraph, 1, "Corrected Plot")
make_plot(axs, tFrame, fData, 2, "Filtered Plot")
make_plot(axs, tFrame, sqData, 3, "Square Plot")
```
Algorithm 9 Plotting Code

Now the counter is used for further calculation. A multiplying factor is calculated using the following formula.

Equation 7 Multiplying Factor Formula

$$multiplying\ factor = \frac{60}{total\ video\ length\ entered\ by\ the\ user}$$

Using this multiplying factor the heart rate per minute is calculated by;

Equation 8 Heartrate per minute formula

$$HRM = ceiling\_value\_of\left(\frac{c}{2}\right) \times multiplying\_factor$$

This HRM (heart rate per minute) is finally returned to the flask application.

VII. Result And Analysis

After giving the desired input the system correctly calculated the heartbeat rate and displayed graphs along with the time (in terms of BPM).

The program then swiftly processes the video input, and analyzes heartbeat for faster processing. It then processes the saturation changes of light and plots a graph. To decrease noise from the preceding generated signal it filters (filtering parts of the signal that are exceeding threshold) and rectifies the graph to generate an impulse signal. The impulse signal thereby generated helps the system to measure heart rate and finally after processing the impulse graph, the system generates a report with patient details, date and time of report generation. For further details a report is tabulated based on observations made for 5 consecutive days taken in morning as well as evening hours to find out the variations. This tabulation indicates the patient's heart rate after any vigorous /normal activity done by the patient and resting heart rate. From the above tabulation, we get a perspective of the varying heart rate for both morning and evening hours with the state of the patient during the measurement of heart rate.

Figure 14 Final Output Plot

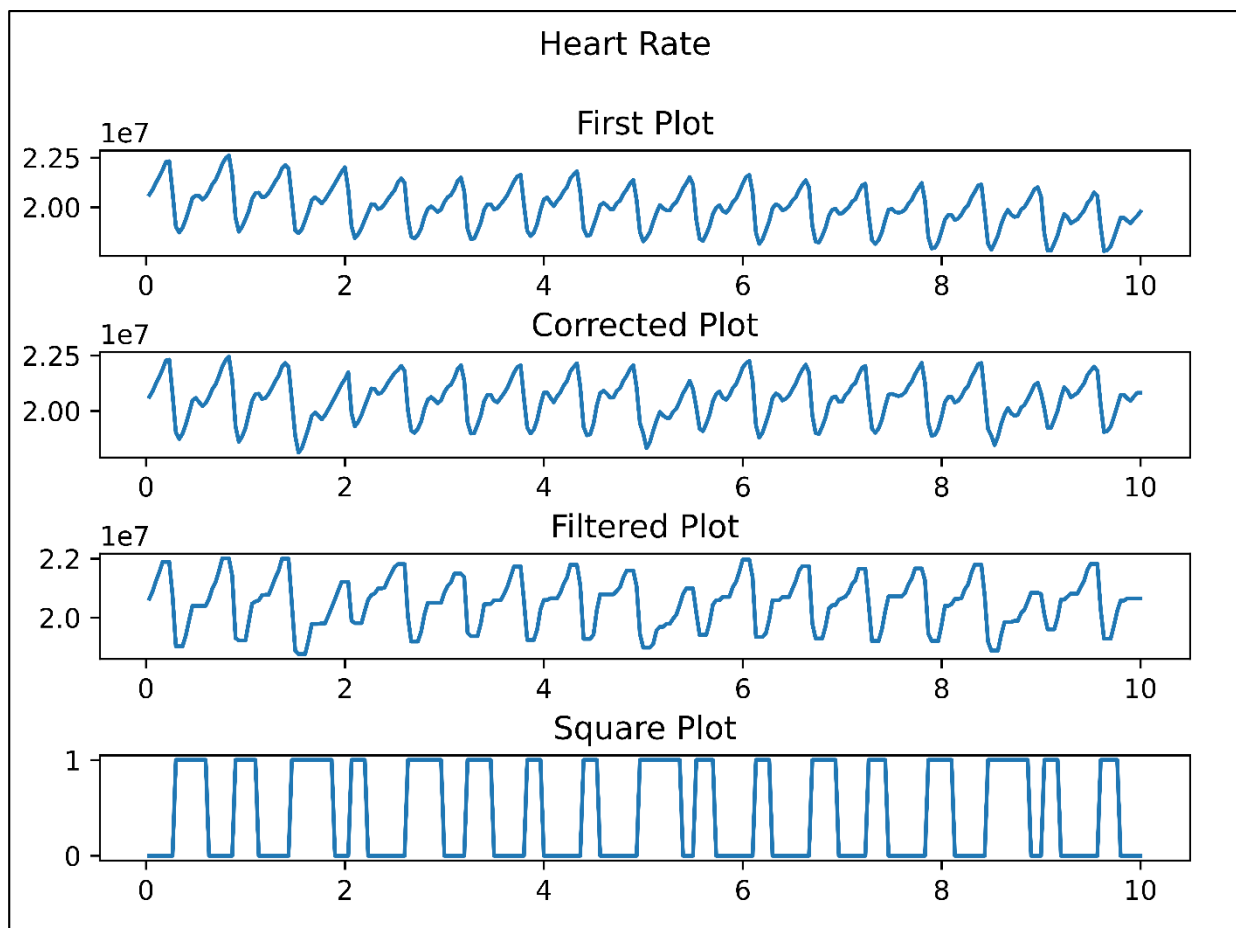This data collected in table – 2 is of a person in his early 20's. Different conditions state different details of the person's state when he took the fingertip video. Resting conditions states that the person was on resting condition and have not moved from his place for at least past 10 mins. Moderate activity states that the person has been doing some mild activity which includes walking or standing.

Strenuous activity states that the person was doing some rigorous work which may include jogging, running etc. The average maximum heartrate of a person in his early 20's is 200 and resting heart rate ranges from 60 to 100 bpm for a normal healthy person. So the above data shows that the person is normal and will lead a healthy life.

Table 2 Observations for concurrent days 10 seconds

Days	Morning				Evening				Status	
	Heartbeat Rate	**Machine HRM**	**Condition**	**Error**	**Heartbeat Rate**	**Machine HRM**	**Condition**	**Error**	**F**	**R**
Day – 1	78	74	Resting	-4	84	89	Moderate Activity	+5	U N F I L T E R E D	U N R E C T I F I E D
Day – 2	78	73	Resting	-5	90	88	Moderate Activity	-2		
Day – 3	84	88	Moderate Activity	+4	102	99	Heavy Activity	-3		
Day – 4	72	74	Resting	+2	84	85	Moderate Activity	+1		
Day – 5	78	81	Resting	+3	96	93	Heavy Activity	-3		
Day – 6	84	83	Moderate Activity	-1	84	83	Moderate Activity	+1	U N F I L T E R E D	R E C T I F I E D
Day - 7	90	90	Heavy Activity	0	84	84	Moderate Activity	0		
Day – 8	84	85	Moderate Activity	+1	78	79	Resting	+1		
Day – 9	78	79	Resting	+1	90	90	Heavy Activity	0		
Day - 10	78	78	Resting	0	96	93	Heavy Activity	-3		
Day – 11	72	71	Resting	-1	84	84	Moderate Activity	0	F I L T E R E D	R E C T I F I E D
Day - 12	78	78	Resting	0	90	90	Heavy Activity	0		
Day – 13	78	78	Resting	0	90	90	Heavy Activity	0		
Day – 14	72	72	Resting	0	78	78	Resting	0		
Day - 15	72	72	Resting	0	84	83	Moderate Activity	-1		

As seen from the above graph the error lies between -6 to +6. This is because of the length of the video, so as the length is 10 seconds for the above table the multiplying factor is 6. This changes if the graph is then rectified, which decreases the error values as can be seen in the Unfiltered, rectified section of the observation table. Now let's use data for 20 seconds of video.

Table 3 Observations for concurrent days 20 seconds

Days	Morning				Evening				Status	
	Heartbeat Rate	Machine HRM	Condition	Error	Heartbeat Rate	Machine HRM	Condition	Error	F	R
Day – 1	102	100	Heavy Activity	-2	72	71	Resting	-1	U N F I L T E R E D	U N R E C T I F I E D
Day – 2	78	78	Resting	0	78	77	Resting	-1		
Day – 3	75	76	Resting	+1	72	72	Resting	0		
Day – 4	90	90	Heavy Activity	0	72	71	Resting	-1		
Day – 5	96	96	Heavy Activity	0	78	79	Resting	+1		
Day – 6	105	105	Heavy Activity	0	72	73	Resting	+1	U N F I L T E R E D	R E C T I F I E D
Day - 7	72	72	Resting	0	72	71	Resting	-1		
Day – 8	105	105	Heavy Activity	0	78	78	Resting	0		
Day – 9	84	83	Moderate Activity	-1	81	81	Moderate Activity	0		
Day - 10	102	102	Heavy Activity	0	81	82	Moderate Activity	+1		
Day – 11	93	93	Heavy Activity	0	84	85	Moderate Activity	+1	F I L T E R E D	R E C T I F I E D
Day - 12	96	96	Heavy Activity	0	78	78	Resting	0		
Day – 13	102	101	Heavy Activity	-1	78	78	Resting	0		
Day – 14	93	92	Heavy Activity	-1	81	81	Moderate Activity	0		
Day - 15	87	87	Moderate Activity	0	84	84	Moderate Activity	0		

As it can be observed in the above table for 20 seconds the errors decreases significantly, and the error that occur is in the range of -3 to +3.

For reference, a pulse oximeter is used and the data is compared with the readings of the pulse oximeter. The readings for the pulse oximeter was taken for at least for 1 minute for accurate readings. The following figures shows the output of both our system and the oximeter.
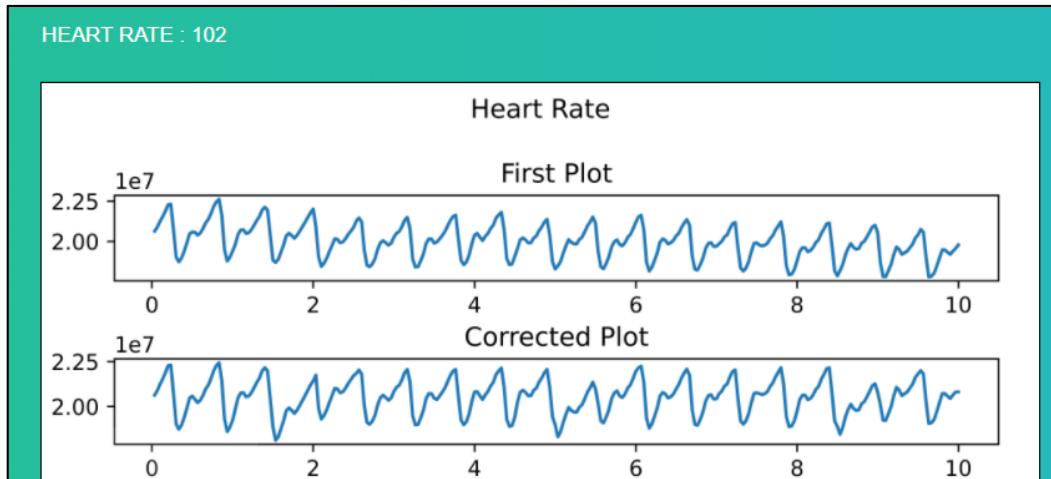

Figure 15 HRM System Output

Figure 16 Oximeter Readings

The oximeter readings are taken in the span of 1 minute and the HRM for our system is for 10 seconds. This gives a difference of 2 beats, but the pulse rate could differ at the time of reading, like with prolonged time of resting the heart rate will start dropping.
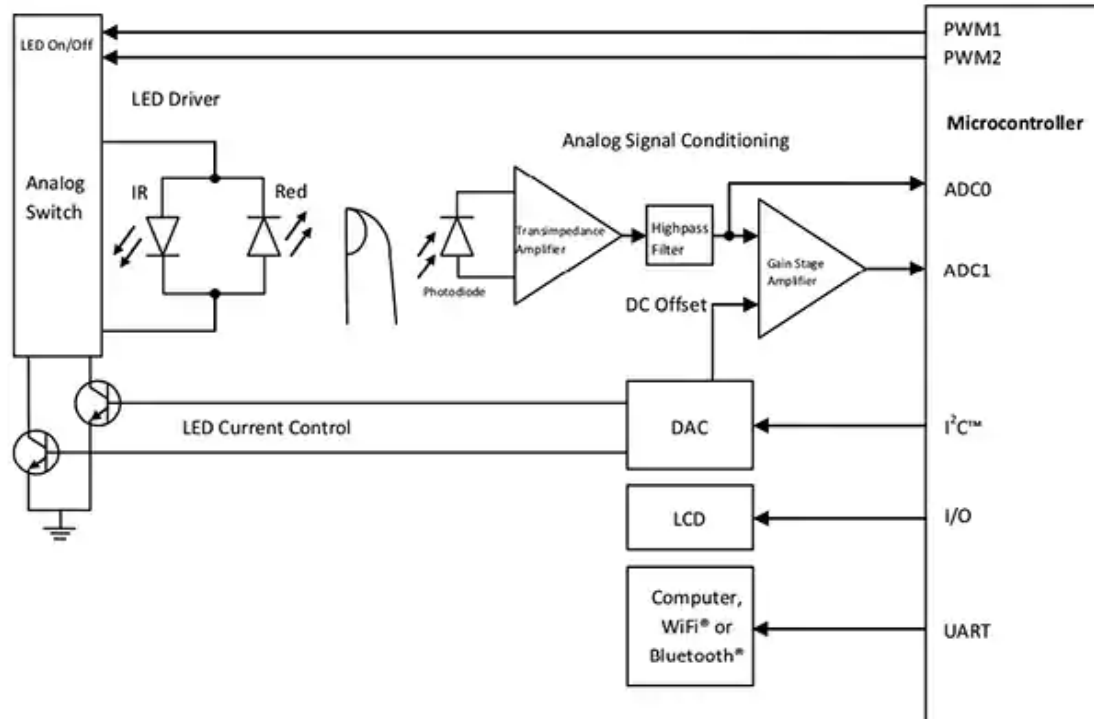
Figure 17 Typical Oximeter design

As seen in the above circuit design, the pulse oximeter photodiode to detect the intensity of light passing through the finger. Our system does the same thing but using OpenCV technology and no use of expensive hardware like diodes, microcontroller, transistor, etc. Our system uses things that are now present in almost every single household for the results without dropping the quality of readings. Another advantage it has over the oximeter is that it can handle very high heart rates, i.e. in critical conditions like 150 and above pulse rate, most oximeter fail to handle and flat-line the result, since our system is using OpenCV and each frame of each seconds is being processed, it is able to handle the extreme test cases.

VIII. Conclusion And Future Scope

Nowadays taking care of health must be of utmost importance especially as we neglect ourselves of following healthy life majorly due to the hectic life we all are going and surviving through to make both ends meet. But maintaining this simple yet important aspect is an arduous task. Having to go to the doctors now and then could be tedious but also in the contrary could reduce the risk of critical cases. Another difficulty could be for the people living in the rural areas as the medical establishment could be far away. But as technology is developing, almost everything can be done from home. This system thus is capable of generating medical report without any intervention of doctors and also patients get an added advantage of having the option to have a personal medical feedback from an experienced doctor. If any abnormalities are found beforehand critical situations could be avoided. Our purposed model intends to reduce the cost of monitoring patients regardless of location.

In future, more parameters of a patient's body could be taken into consideration to further improve the result of our model.

PLANNING AND PROJECT MANAGEMENT

Table 4 Showing details about project planning and management

| Activity	Starting week	Number of weeks
Literature review	November	4
Idea Generation	3rd and 4th week of November	2
Software and environment setup	1st and 2nd week of December	2
Concept Training	December	4
Design of basic HRM	3rd and 4th week of December	2
Time Flexibility functionality added	1st and 2nd week of January	2
Noise Rectification	1st and 2nd week of January	2
Web Application	3rd and 4th week of January, 1st and 2nd week of February	4
Flask Server	3rd and 4th week of February, March	4

The Gantt chart is shown below:
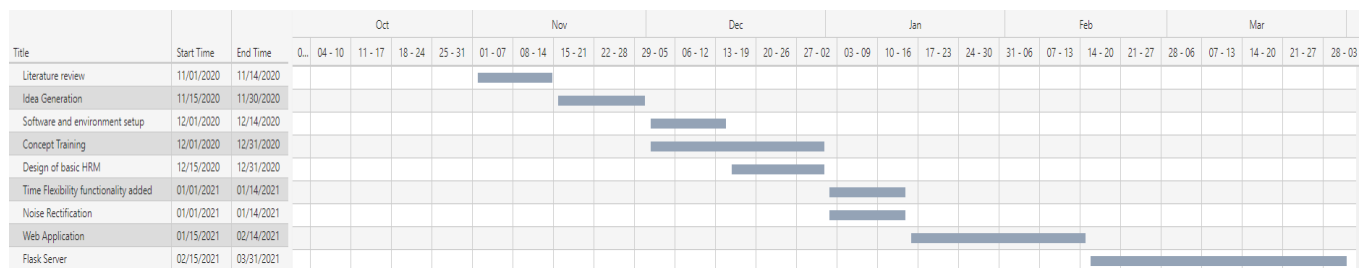
Figure 18 Gantt Chart

REFERENCES

1. B. Mallick, and A. K. Patro, "Heart rate monitoring system using fingertip through Arduino and processing software", International Journal of Science, Engineering and Technology Research (IJSETR), vol. 5, no.1, 2016.

2. J. M. B. Lakshmi, R. Hariharan, P. U. S. C. N. Devi, and N. Sowmiya, "Heart beat detector using infrared pulse sensor", International Journal for Scientific Research & Development (IJSRD), vol.3, no.9, 2015.

3. S. Das "The Development of a Microcontroller Based Low Cost Heart Rate Counter for Health Care Systems" International Journal of Engineering Trends and Technology, vol. 4, no.2, 2013.

4. S.U. Ufoaroh, C. O. Oranugo, and M. E. Uchechukwu, "Heartbeat monitoring &alert system using GSM technology" International Journal of Engineering Research and General Science Vol. 3, Issue 4, 2015.

5. C. S. K. Subudhi, "Intelligent Wireless Patient Monitoring and Tracking System (Using Sensor Network and Wireless Communication)", 2014.

6. P. Gothwal, "Designing of PPG based Heart-beat Monitor", International Journal for Research in Applied Science & Engineering Technology (IJRASET), 2016.

7. P. V. Rao, V. Akhila, Y. Vasavi, and K. Nissie, "An IoT based patient health monitoring system using Arduino Uno", International Journal of Research in Information Technology (IJRIT), vol. 1, no.1, 2017.

8. R. Chandra, A. Rathee, P. Verma, and A. Chougule, "GSM Based Health Monitoring System" Proceedings of IRF International Conference,Pune, ISBN: 978-93-84209-04-9, 2014.

9. M. B Prasad," GSM based Health Care Monitoring System," International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, vol.8, no.2S, 2018.

10. S. Vinodhini, J. Haritha, and E. S. Gayathri, "Heart Rate Monitoring Using GSM Technology" Iconic Research and Engineering Journals, ISSN: 2456-8880, vol.2, no.5, 2018.

11. R. Nanaware, and T. B. Sonawane, "Portable Health Monitoring System with GPS, GSM and emergency Switch" IOSR Journal of Engineering (IOSRJEN), www.iosrjen.org ISSN (e): 2250-3021, ISSN (p): 2278-8719, vol. 09, No.5, PP. 14-19, 2019.

12. M. Yeole, and R. Daryapurkar, "Design and Implementation of Wireless Heartbeat Measuring Device for Remote Health Monitoring", VJER-Vishwakarma Journal of Engineering Research, www.vjer.in,vol.1,no.2, ISSN: 2456-8465, 2017.

13. S.U. Atiya, and K.S. Madhuri, "GSM Based Patient Monitoring System using biomedical sensors" International Journal of Computer Engineering in Research Trends. pp. 620-624, ISSN (O): 2349-7084, vol.3, no.9, 2016.

14. M. Susarla, C. Akhil, A. Reddy, and D. D. Hema, "Heartbeat Detection and Monitoring Using IOT" Journal of Network Communications and Emerging Technologies (JNCET), www.jncet.org, ISSN: 2395-5317, vol.8, no.5, 2018.

15. T. V. Sethuraman, K. S. Rathore, G. Amritha, and G. Kanimozhi," IoT based system for Heart Rate Monitoring and Heart Attack Detection" International Journal of Engineering and Advanced Technology (IJEAT), ISSN: 2249-8958,vol.8,no.5, pp.1459-1464, 2019.

16. N. Kaleeswari, R. A. Chakravarthy, and M. Arun,"Adaptive Heart Monitoring System Using Iot, "International Journal of Recent Technology and Engineering (IJRTE), ISSN: 2277-3878, vol.7, no.6S5, 2019.

17. S. Gowrishankar, M. Y. Prachita and A. Prakash, "IoT based Heart Attack Detection, Heart Rate and Temperature Monitor" International Journal of Computer Applications, ISSN: (0975 − 8887), vol.170, no.5, 2017.

18. B. Priyadharshini, and K. Priya, "Heart Rate Monitoring System based on IOT" IOSR Journal of Engineering (IOSR JEN), www.iosrjen.org ISSN (e): 2250-3021, ISSN (p): 2278-8719, PP 41-48, 2019.

19. S. K. Sufiya, G. Bajantri, and T. Thite, "Remote Heart Rate Monitoring System Using IoT" International Research Journal of Engineering and Technology (IRJET), e-ISSN: 2395-0056, p-ISSN: 2395-0072, vol.5, no.4, 2018.

20. R. Devi, S. G. Gowri, and K. Sethuraman, " A Study on Heart Rate Monitoring Systems Using IoT" International Conference on Emerging Trends in Science, Technology and Mathematics (ICETSTM-2018), An ISO 3297: 2007 Certified Organization, ISSN (Online) : 2319 - 8753, ISSN (Print): 2347 - 6710, vol.7, no.8, 2018.

21. R. K. Dubey, S. Mishra, S. Agarwal, R. Sharma, N. Pradhan, and V. Saran, "Patient's Health Monitoring System using Internet of Things (IoT)" International Journal of Engineering Trends and Technology(IJETT), ISSN:2231-5381, http://www.ijettjournal.org, vol.59, no.3, pp.155 ,2018.

22. A. Senapati, A. Maitra, N. Saha, A. K. Kashyap, B. K. Mondal, and S. Chatterjee, "An IOT based Portable Health Monitoring Kit" International Journal for Research in Applied Science & Engineering Technology (IJRASET), ISSN:2321-9653; vol.6, no.8, 2018.

23. J. S. Adivarekar, A. D. Chordia, H. H. Baviskar, P. V. Aher, and S. Gupta, " Patient Monitoring System Using GSM Technology" International Journal Of Mathematics And Computer Research" ISSN :2320-7167, vol.1 , no.2, pp.73-78, March 2013.

24. Sk. M. Subhani , G. N. V. Sateesh, Ch. Chaitanya and G. B. Prakash, "Implementation of GSM Based Heart Rate and Temperature Monitoring System" Research Journal of Engineering Sciences, ISSN: 2278 − 9472, vol. 2, no.4, pp.43-45, 2013.

25. R. S. P. Talluri, Y. JaiSurya, and S. L. Manchala, "Heart Rate Monitoring System using Heart Rate Sensor and Arduino Uno with Web Application", International Journal of Engineering and Advanced Technology (IJEAT), ISSN: 2249 − 8958, vol.8, no.4, pp.350-352, 2019.

26. N. Surekha, N. Yamuna, A. J. A. Kumar, and K. G. N. Kumar, "Patient Monitoring System Using IOT" International Journal of Innovative Research in Advanced Engineering (IJIRAE), ISSN:2349-2163, vol.5, no.176, 2018

27. Frazer K. Noble, "Comparison of OpenCV's Feature Detectors and Feature Matchers", IEEE Xplore: 2016 23rd International Conference on Mechatronics and Machine Vision in Practice (M2VIP), DOI: 10.1109/M2VIP.2016.7827292, ISBN: 978-1-5090-2765-1

28. Wenbin Zhang, Chengliang Zhang, Chengbin Li, He Zhang, "Object color recognition and sorting robot based on OpenCV and machine vision", IEEE Xplore: 2020 IEEE 11th International Conference on Mechanical and Intelligent Manufacturing Technologies (ICMIMT), DOI: 10.1109/ICMIMT49010.2020.9041220, ISBN:978-1-7281-5333-9

29. Jinshu Bai, Yicen Li, Lan Lin, Lixin Chen, "Mobile Terminal Implementation of Image Filtering and Edge Detection Based on OpenCV", IEEE Xplore: 2020 IEEE International Conference on Advances in Electrical Engineering and Computer Applications (AEECA), DOI: 10.1109/AEECA49918.2020.9213537, ISBN:978-1-7281-6522-6

30. Edwin Mauricio Torres Caballero, Alicia María Reyes Duke, "Implementation of Artificial Neural Networks Using NVIDIA Digits and OpenCV for Coffee Rust Detection", IEEE Xplore: 2020 5th International Conference on Control and Robotics Engineering (ICCRE), DOI: 10.1109/ICCRE49379.2020.9096435, ISBN:978-1-7281-6792-3

31. Akanksha Soni, Arun Pratap Singh, "Automatic Motorcyclist Helmet Rule Violation Detection using Tensorflow & Keras in OpenCV", IEEE Xplore: 2020 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS), DOI: 10.1109/SCEECS48394.2020.55, ISBN:978-1-7281-4863-2, Electronic ISBN:978-1-7281-4862-5, Electronic ISSN: 2688-0288, Print on Demand(PoD) ISSN: 2688-027X, INSPEC Accession Number: 19593019\

32. Jose Sigut, Miguel Castro, Rafael Arnay, Marta Sigut, "OpenCV Basics: A Mobile Application to Support the Teaching of Computer Vision Concepts", IEEE Xplore: IEEE Transactions on Education (Volume: 63, Issue: 4, Nov.2020), DOI:10.1109/TE.2020.2993013, Electronic ISSN: 1557-9638, INSPEC Accession Number: 20095077

33. Angel Ciprian Cormoş, Răzvan Andrei Gheorghiu, Valentin Alexandra STAN, Ion Spirea Dănăilă, "Use of TensorFlow and OpenCV to detect vehicles", IEEE Xplore: 2020 12th International Conference on Electronics, Computers and Artificial Intelligence (ECAI), DOI: 10.1109/ECAI50035.2020.9223173, ISBN:978-1-7281-6844-9, INSPEC Accession Number: 20062174

34. Jahanvi Mehariya, Chaitra Gupta, Niranjan Pai, Sagar Koul, Prashant Gadakh, "Counting Students using OpenCV and Integration with Firebase for Classroom Allocation", IEEE Xplore: 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC), DOI: 10.1109/ICESC48915.2020.9155825, ISBN:978-1-7281-4109-1,

INSPEC Accession Number: 19876950

35. Yongjie Gui, Yanyan Wu, Yajie Wang, Chunpeng Yao, "Visual Image Processing of Humanoid Go Game Robot Based on OPENCV", IEEE Xplore: 2020 Chinese Control and Decision Conference (CCDC), DOI: 10.1109/CCDC49329.2020.9164541, ISBN:978-1-7281-5856-3, Electronic ISSN: 1948-9447, INSPEC Accession Number: 19871764

36. Naman Gupta, Purushottam Sharma, Vikas Deep, Vinod Kumar Shukla, "Automated Attendance System Using OpenCV", IEEE Xplore: 2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), DOI: 10.1109/ICRITO48877.2020.9197936, ISBN:978-1-7281-7017-6, INSPEC Accession Number: 19988217

37. Arun Kumar Jhapate, Sunil Malviya, Monika Jhapate, "Unusual Crowd Activity Detection using OpenCV and Motion Influence Map", IEEE Xplore: 2nd International Conference on Data, Engineering and Applications (IDEA), DOI: 10.1109/IDEA49133.2020.9170704, ISBN:978-1-7281-5719-1, INSPEC Accession Number: 19896448

38. Ruben du Toit , Günther Drevin, Nicolaas Maree, Du Toit Strauss , "Sunspot Identification and Tracking with OpenCV", IEEE Xplore: 2020 International SAUPEC/RobMech/PRASA Conference, DOI: 10.1109/SAUPEC/RobMech/PRASA48453.2020.9040971, ISBN:978-1-7281-4163-3, INSPEC Accession Number: 19472644