Digital Image Processing Mini Project

CONTOUR BASED PEOPLE COUNTING USING OPENCV

Group Members :

SRN	NAME
01FB16EEC004	ABHIJITH B N
01FB16EEC003	AASISH TAMMANA

PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)
100-ft Ring Road, Bengaluru – 560 085, Karnataka, India

A Technical Mini Project Report

'CONTOUR BASED PEOPLE COUNTING'

Submitted in Partial fulfillment of the Requirements for V Semester

BTECH
In
Electronics and Communication

By

ABHIJITH B N (01FB16EEC004)
AASISH TAMMANA (01FB16EEC003)

AUG - DEC 2018

Under the guidance of

Mrs. H.R. VANAMALA

Dept of ECE

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

PES UNIVERSITY, BANGALORE-85

ACKNOWLEDGEMENT

I wish to express my profound gratitude and indebtedness to Mrs. H.R. VANAMALA (department of ECE (Electronics and Communication Engineering) PES UNIVERSITY, Bangalore) for her valuable guidance throughout the project work. Also I would like to extend my gratitude to my teammate for an extraordinary teamwork and effort, my parents for their continuous support and all my friends who have patiently extended all sorts of help for accomplishing this undertaking. We would also like to extend our sincere gratitude to **Dr. Anuradha M,** the Chairperson, ECE Department.

ABSTRACT

Image processing is a method of extracting some useful information by converting image into digital information by performing some operations on it. In recent years, processing the images has been a growing research interest because of establishment and development of automatic methods especially in security applications, compression, and perceptual user interface. OpenCV-Python is a library of Python bindings designed to solve computer vision problems. OpenCV-Python makes use of **Numpy**, which is a highly optimized library for numerical operations with a MATLAB-style syntax. It also supports a multitude of algorithms related to Machine Learning and image processing and is expanding day by day.

CONTENTS

INTRODUCTION

1.1 Overview-Background:

A **people counter** is an electronic device that is used to measure the number of people traversing a certain passage or entrance. Examples include simple manual clickers, infrared beams, thermal imaging systems, WiFi trackers and video counters using advanced machine learning algorithms. They are commonly used by retail establishments to judge the effectiveness of marketing campaigns, building design and layout, and the popularity of particular brands. For us, the use case includes retail stores, shopping malls etc where security is a major concern. The inputs can be taken from web camera or any other camera. These videos are subjected to modest OPENCV algorithms which pre-process the footage and detect the region of interest and count the number of people passing.

1.2 Brief Description:

There are basically 2 part which we have implemented in this project. First of all we capture the video using any tool such as a camera. We capture the footage to a frame for processing. After this we convert this colored image to grayscale image and then to black and white binary image. In this the other colors such as red, green and blue does not play any role. For the first algorithm. The input image which is obtained from the camera will be in RGB format. This RGB image is converted to Binary image (BW) using MATLAB function im2bw or thresholding. Once the binary image is obtained, the skin region will be represented by the white (1) pixels and the background is represented by black (0) pixels. In this approach, the algorithm will count the number of white pixels and check the range in which the count falls. The ranges must be predefined in the algorithm by taking some sample values. By finding the white pixel count we can find the number of fingers raised. Since the output depends only on the white pixel count, this algorithm is invariant to rotation of hand [7].

The 2nd and 3rd algorithms are a combined to give best results which include both

morphological operations and scanning method. This is the robust algorithm when compared to the above algorithm [6]. In this method, RGB image is converted to Binary image. The skin regions are represented by white pixels and the background is represented by black pixels. In order to enhance the skin region, the binary image is preprocessed by a median filter of suitable dimension. The image is divided into two halves. The left half contains the thumb and the right half contains the remaining fingers. Now, a vertical scan is performed in left half and a horizontal scan is performed in right half. The presence of thumb is detected in the right half whereas the remaining fingers are detected in the left half. The total count is equal to the sum of counts in two halves. Then erosion and dilation morphological operations are performed for better detection of objects which are fingers in our case. A comparison is done after scanning for transitions from (0-1) is applied in both horizontal and vertical directions to count number of fingers. In this algorithm the RGB image is converted to Binary image. The binary image is filtered using a median filter of dimension 3. The obtained binary image is eroded by using appropriate structural element of dimension 3 to 5. This image is further dilated by using another structural element of dimension 10 to 12. This pre-processing of the binary image is done to enhance the skin region [5].

The same algorithm is applied to the video as well taking each frame as an image and performing the same operations on it.

1.3 Problem Definition:

The main focus in this research project is to experiment deeply with, and find and calculate the number of fingers present in the image by applying different algorithms which involve intense mathematics required for digital image processing. Three main stages are identified in such applications. First, it is necessary to locate and extract the number of pixels which corresponds to the hand only in the image and then define appropriate ranges for the number of fingers to implement the first algorithm (pixel count algorithm). Second, understanding what exactly morphological operations do and how can they be used in the scanning algorithm. Third, use the morphological operations to implement the scanning algorithm. Do intense number of trial and

error to find appropriate threshold values to count the number of objects present to count the number of fingers.

SOFTWARE REQUIREMENTS

Here we describe the implementation of all the 3 algorithms using MATLAB. MATLAB is a very powerful software tool used to implement the tasks that require extensive computations. It provides easy and quicker implementation of algorithms compared to C and C++. The key feature in MATLAB is that it contains a rich library functions for image processing and data analysis. This makes MATLAB an ideal tool for faster implementation and verification of any algorithm before actually implementing it on a real hardware. Sometimes, debugging of errors on actual hardware turns out to be a very painful task. MATLAB provides an easy approach for debugging and correction of errors in any algorithm. Other than this, MATLAB contains many features including workspace, plot, imread, imshow, im2bw, rgb2gray, bwlabel, imdilate, imerode, med2filt for data analysis and image processing, which makes it a better choice over other software languages like C and C++.

Considering the above advantages, the writer of this project initially only described the algorithms for Gesture Recognition using MATLAB. The algorithm uses various inbuilt functions and implemented few user defined routines related to image processing which are very explained in MATLAB documentation Once the algorithm was developed, it was verified with multiple input images containing different fingers. A flow-chart showing the basic implementation of algorithm is shown on next page.

SOFTWARE DESIGN / FLOWCHARTS

1. Pixel Count Algorithm:

- RGB Image
- Gray Image
- Binary Image
- Count pixels for hand/fingers
- Apply a median filter to remove noise

- Define range for each count of finger
- Output image
- Count of fingers

2. and 3. Morphological operations and Scanning:

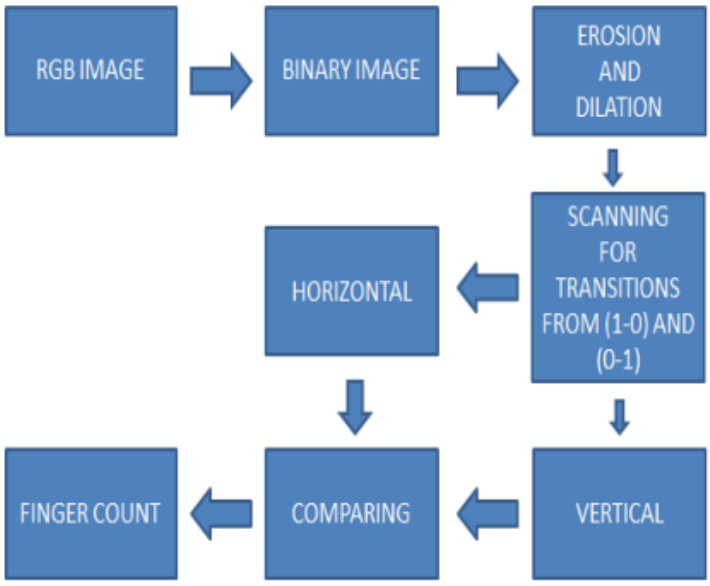


Figure 1. Flowchart of Scanning Method

1. Pixel Count Algorithm:

-import the image

-apply pre-processing techniques (gray scale, gray to binary, median filter)

-define the threshold range for different number of fingers

-find the pixel count of the image and compare them with the threshold

- find the number of finger.

2. Scanning method:

-import the image

-apply pre-processing techniques (gray scale, gray to binary, erode and dilate)

-divide the image into 2 halves, one with fingers and the other with the thumb

-apply vertical scanning for the fingers and horizontal scanning for the thumb

-the obtained image is then subjected to dilating

-Using bwlabel function the objects in the image are labelled

-If the number of pixels in the given object is more than the threshold the increment the finger counts. Apply the same technique for the thumb count.

Limitations:

-the technique used here does not detect the image.

-it does not work on real time video

It might not work properly if the hand distance from camera is changed.

IMPLEMENTATION

1. Pixel Count Algorithm:

Code:

```
clc;
```

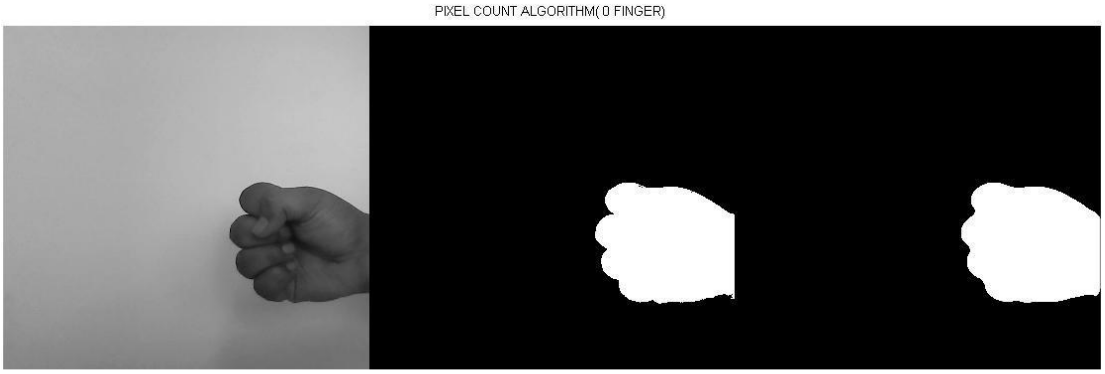
```
clear all;
close all;
img0 = imread('G:\College Work\VI sem\DIP\0.jpg');
img1 = imread('G:\College Work\VI sem\DIP\1.jpg');
img2 = imread('G:\College Work\VI sem\DIP\2.jpg');
img3 = imread('G:\College Work\VI sem\DIP\3.jpg');
img4 = imread('G:\College Work\VI sem\DIP\4.jpg');
img5 = imread('G:\College Work\VI sem\DIP\5.jpg');
I = img0;
Ig = rgb2gray(I);
cnt=0;
thres=105;
[r c] = size(Ig);
for i = 1:r              %or use im2bw function
   for j =1:c
      if Ig(i,j)<thres
         I1(i,j)=255;
         cnt=cnt+1;
      else
         I1(i,j)=0;
      end
   end
end

Imed = medfilt2(I1,[20 20]);
thres1 = 0;
thres2 = 58000;
thres3 = 61000;
```

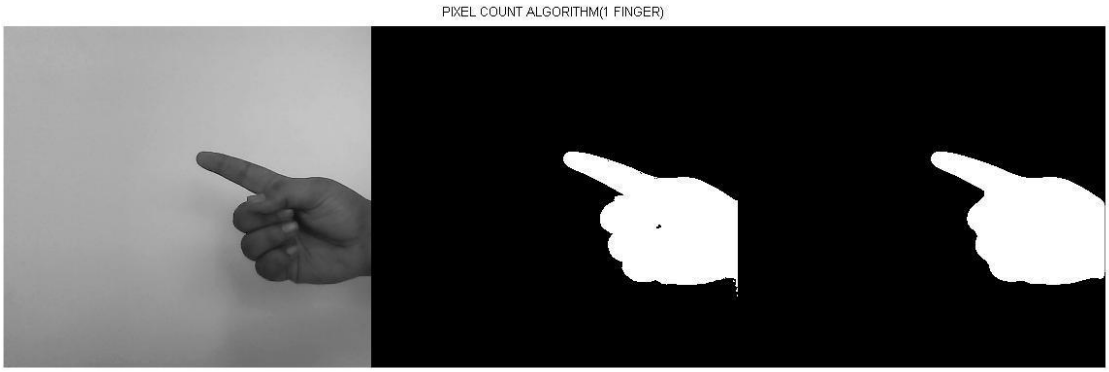


```
thres4 = 68000;
thres5 = 83000;
```

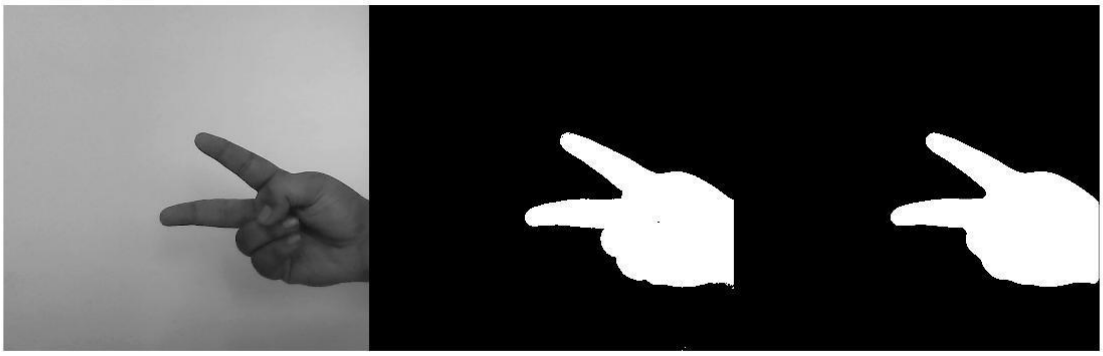
```
thres6 = 99000;
thres7 = 107000;
finger_count = -1;
if     (cnt > thres1 && cnt <= thres2)
   finger_count = 0;
elseif (cnt > thres2 && cnt <= thres3)
   finger_count = 1;
elseif (cnt > thres3 && cnt <= thres4)
   finger_count = 2;
elseif (cnt > thres4 && cnt <= thres5)
   finger_count = 3;
elseif (cnt > thres5 && cnt <= thres6)
   finger_count = 4;
elseif (cnt > thres6 && cnt <= thres7)
   finger_count = 5;
end
disp('No.of fingers:');
disp(finger_count);
figure();
imshow([Ig I1 Imed]);
```

Outputs:

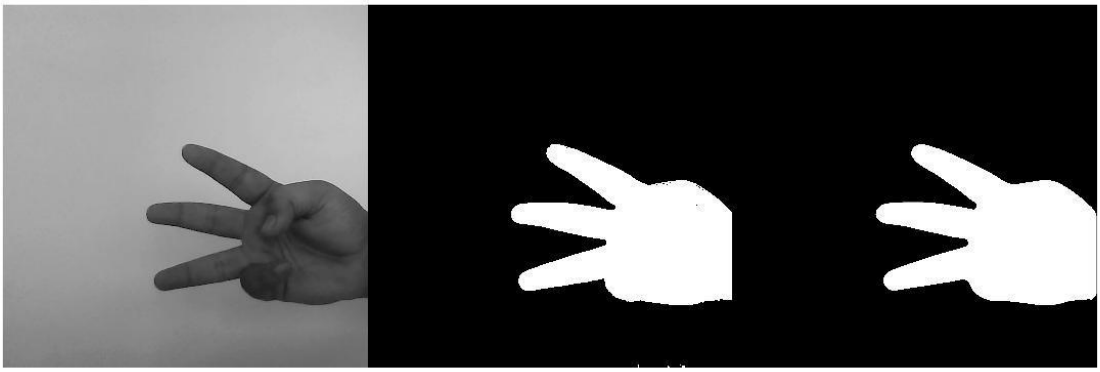

PIXEL COUNT ALGORITHM(0 FINGER)


PIXEL COUNT ALGORITHM(1 FINGER)

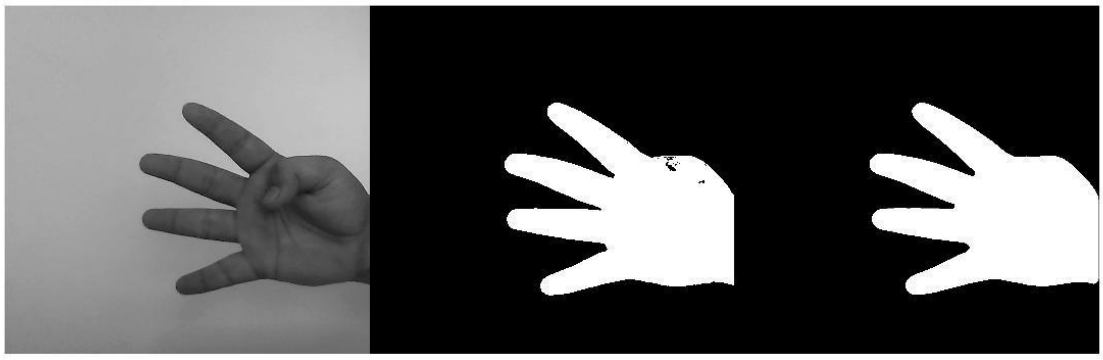
PIXEL COUNT ALGORITHM(2 FINGERS)



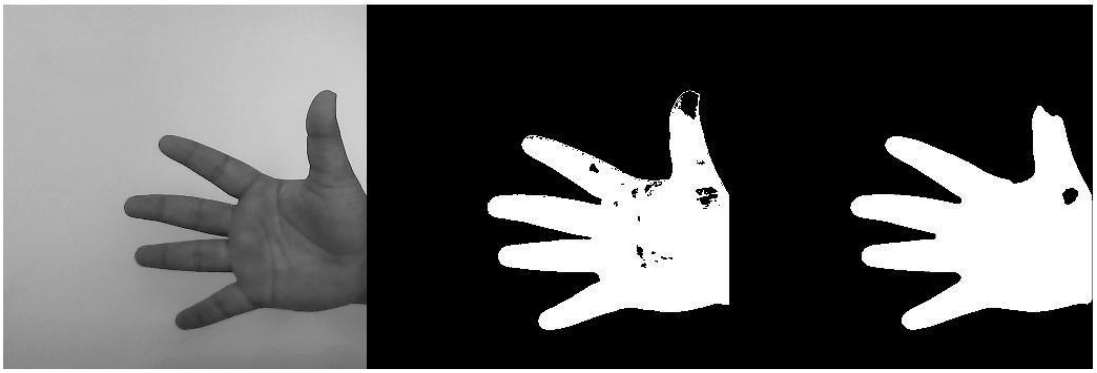
PIXEL COUNT ALGORITHM(3 FINGERS)



PIXEL COUNT ALGORITHM(4 FINGERS)



PIXEL COUNT ALGORITHM(5 FINGERS)



2. Scanning Algorithm:

Code:

```
clc;
clear all;
close all;
image0 = imread('G:\College Work\VI sem\DIP\img (1).png');
image1 = imread('G:\College Work\VI sem\DIP\img (2).png');
image2 = imread('G:\College Work\VI sem\DIP\img (3).png');
image3 = imread('G:\College Work\VI sem\DIP\img (4).png');
image4 = imread('G:\College Work\VI sem\DIP\img (5).png');
image5 = imread('G:\College Work\VI sem\DIP\img (6).png');
img0   = imread('G:\College Work\VI sem\DIP\0.jpg');
img1   = imread('G:\College Work\VI sem\DIP\1.jpg');
img2   = imread('G:\College Work\VI sem\DIP\2.jpg');
img3   = imread('G:\College Work\VI sem\DIP\3.jpg');
img4   = imread('G:\College Work\VI sem\DIP\4.jpg');
img5   = imread('G:\College Work\VI sem\DIP\5.jpg');
img    = img0;
img_gray = rgb2gray(img);
img_bin =  im2bw(img,0.39); % Binary Image
img_neg = 1-img_bin; % Convert to Negative image
erode = strel('diamond',3);
dilate = strel('diamond',11);
img_erode = imerode(img_neg,erode);
```

```
img_dilate = imdilate(img_erode,dilate);
[r,c] = size(img_dilate);
for i = 1 : r - 1
   for j = 1 : (0.65*c)
      img_fing(i,j) = 0;
      if(img_dilate(i,j) == 0 && img_dilate(i+1,j) == 1)
         img_fing(i,j) = 1;
      end
   end
end
for i = 1 : (0.5*r)
```



```
   for j = ceil((0.76*c)):c-1
      img_thumb(i,j) = 0;
      if(img_dilate(i,j) == 0 && img_dilate(i,j+1) == 1)
         img_thumb(i,j) = 1;
      end
   end
end
img_thumb = imdilate(img_thumb,dilate);
img_thumb = medfilt2(img_thumb,[10,10]);
img_label2=bwlabel(img_thumb);
img_fing = imdilate(img_fing,dilate);
img_fing = medfilt2(img_fing,[10,10]);
img_label1=bwlabel(img_fing);
thumb_count=0;
finger_count =0;
[r,c]=size(img_label1);
for k=1:max(max(img_label1))
   cnt=0;
   for i=1:r
```

```
        for j=1:c
            if(img_label1(i,j)==k)
                cnt=cnt+1;
            end
        end
    end
    count(k)=cnt;
    if(count(k)>2000)
        finger_count=finger_count+1;
    end
end
if(finger_count==4)
    [r,c]=size(img_label2);
    for k=1:max(max(img_label2))
        cnt=0;
        for i=1:r
            for j=1:c
```



```
                if(img_label2(i,j)==k)
                    cnt=cnt+1;
                end
            end
        end
        count(k)=cnt;
        if(count(k)>2000)
            thumb_count=thumb_count+1;
        end
    end
end
total_fingers = finger_count+thumb_count;
figure(1);
```

```
subplot(224)
if(total_fingers==0)
   imshow(image0);
elseif(total_fingers==1)
   imshow(image1);
elseif(total_fingers==2)
   imshow(image2);
elseif(total_fingers==3)
   imshow(image3);
elseif(total_fingers==4)
   imshow(image4);
elseif(total_fingers==5)
   imshow(image5);
end
subplot(223);
imshow(img_label2);
subplot(222);
imshow(img_label1);
subplot(221);
imshow(img_dilate);
figure(2);
subplot(224);
imshow(img_erode);
```

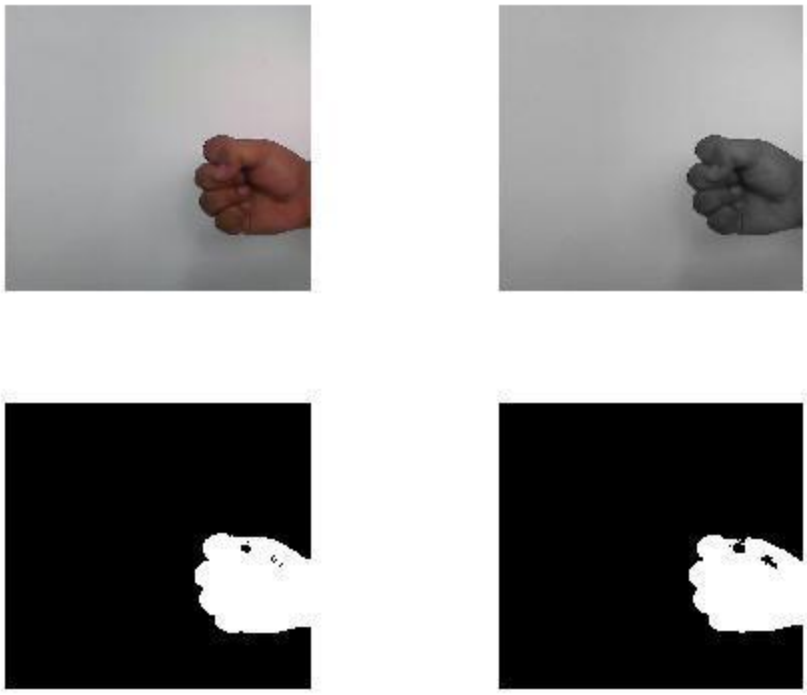


```
subplot(223);
imshow(img_neg);
subplot(222);
imshow(img_gray);
subplot(221)
imshow(img);
disp('No. of fingers:');
```

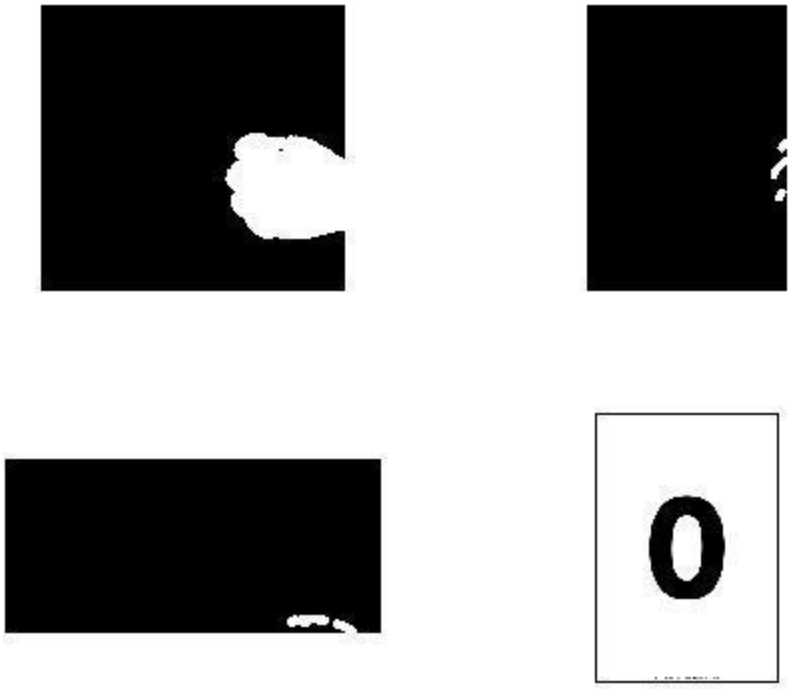
```
disp(total_fingers);
```


Outputs:

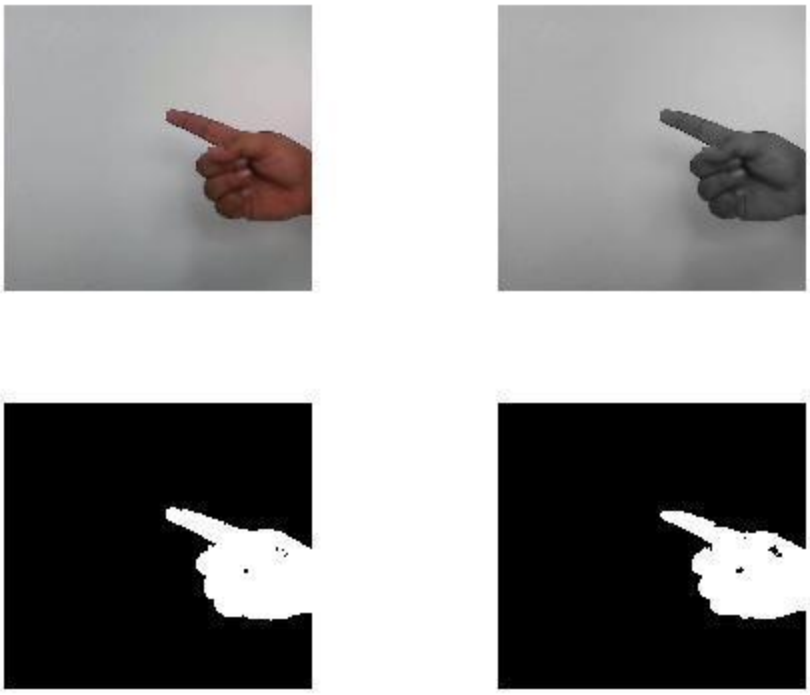
SCANNING ALGORITHM



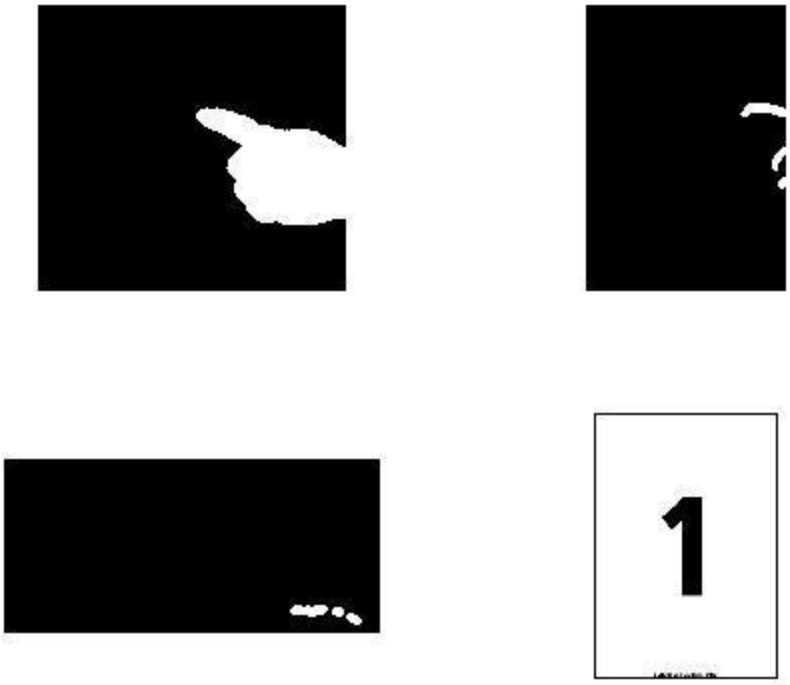
SCANNING ALGORITHM



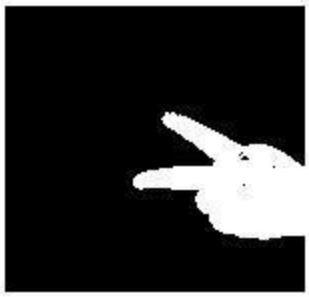
SCANNING ALGORITHM

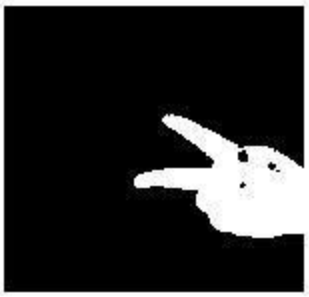


SCANNING ALGORITHM

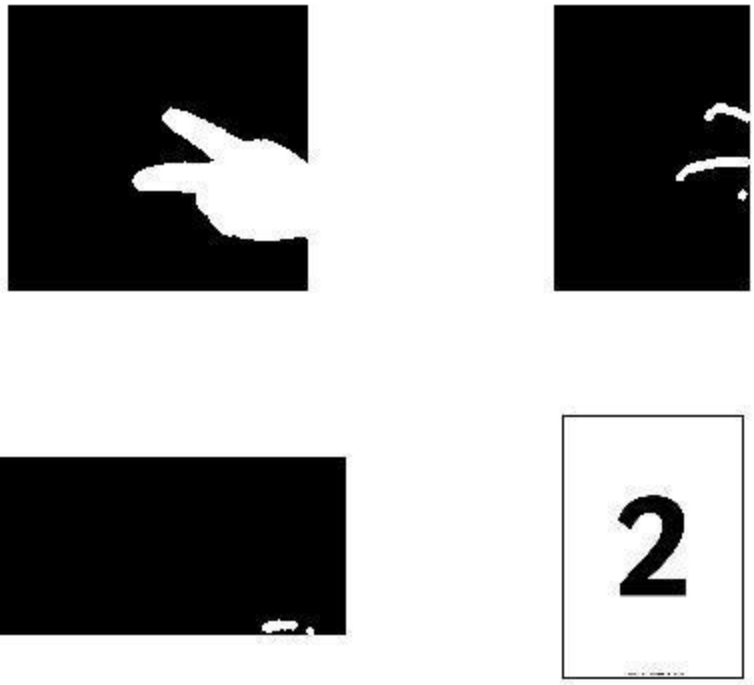


SCANNING ALGORITHM

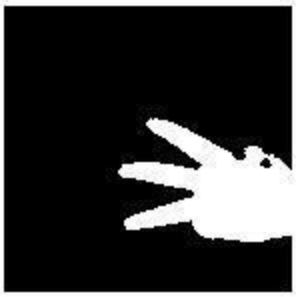




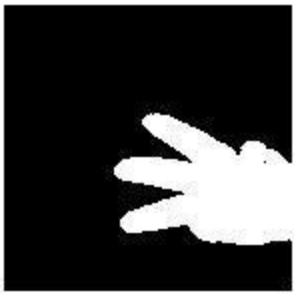
SCANNING ALGORITHM



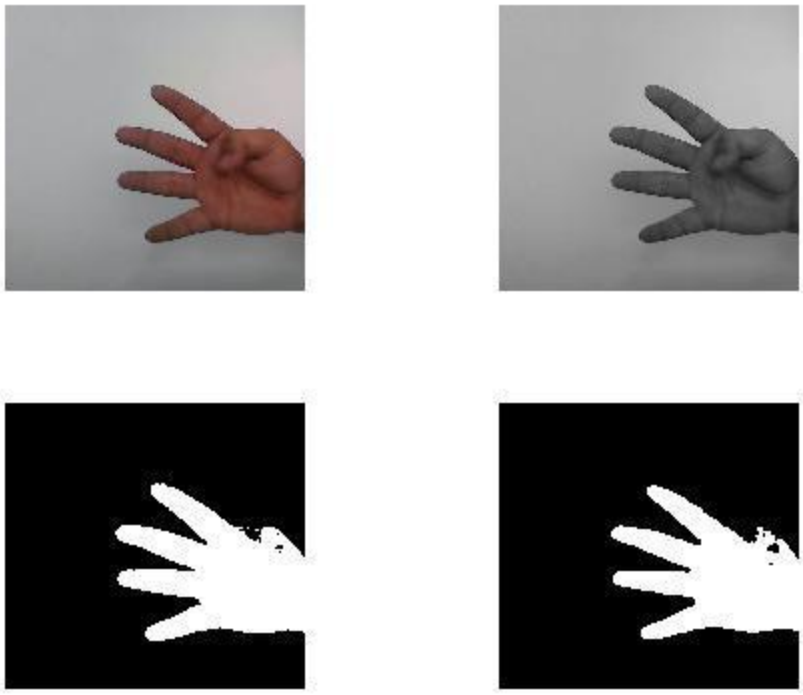
SCANNING ALGORITHM



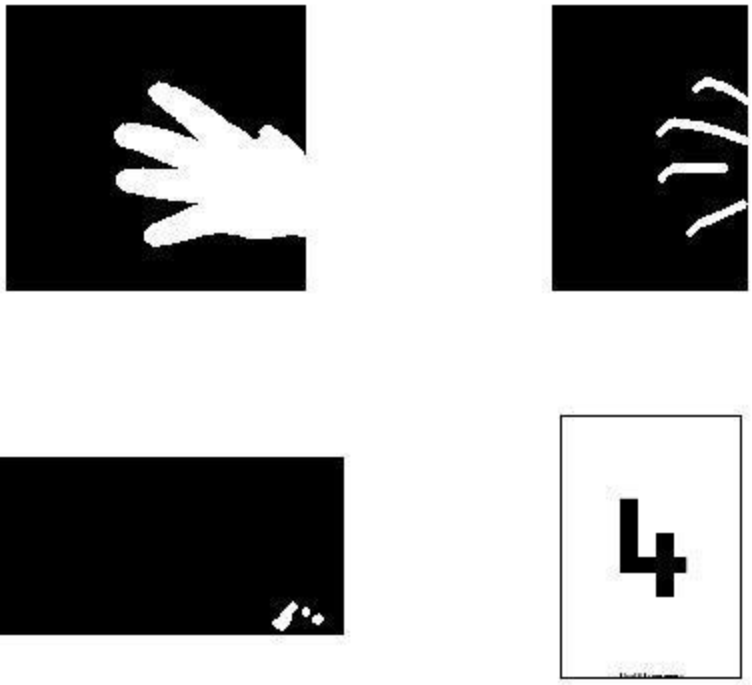
SCANNING ALGORITHM



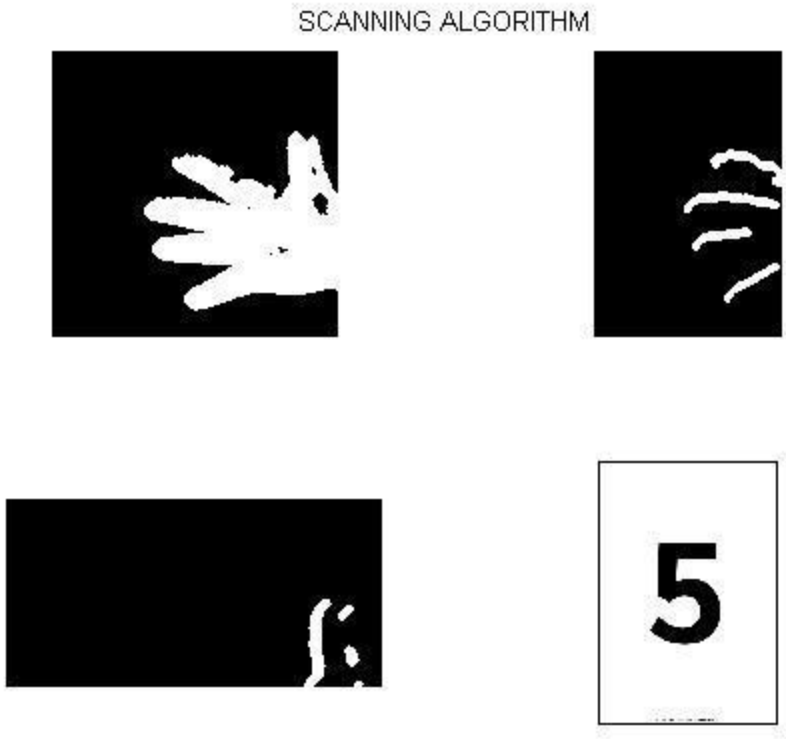
SCANNING ALGORITHM



SCANNING ALGORITHM



SCANNING ALGORITHM



3. Video Scanning:

Code:

```
clc;
clear all;
close all;
image0 = imread('G:\College Work\VI sem\DIP\img (1).png');
image1 = imread('G:\College Work\VI sem\DIP\img (2).png');
image2 = imread('G:\College Work\VI sem\DIP\img (3).png');
image3 = imread('G:\College Work\VI sem\DIP\img (4).png');
```

```
image4 = imread('G:\College Work\VI sem\DIP\img (5).png');
image5 = imread('G:\College Work\VI sem\DIP\img (6).png');
vid=VideoReader('G:\College Work\VI sem\DIP\0.mp4');
numFrames = vid.NumberOfFrames;
n=numFrames;
for c = 1:20:n-20
  img = read(vid,c);
  img_gray = rgb2gray(img);
img_bin =  im2bw(img,0.39); % Binary Image
 img_white = 1-img_bin; % Convert to White image
erode = strel('diamond',3);
dilate = strel('diamond',11);
img_erode = imerode(img_white,erode);
img_dilate = imdilate(img_erode,dilate);
img_fing=0;
img_thumb=0;
[r,c] = size(img_dilate);
for i = 1 : r - 1
    for j = 1 : (0.40*c)
        img_fing(i,j)=0;
        if(img_dilate(i,j) == 0 && img_dilate(i+1,j) == 1)
            img_fing(i,j) = 1;
        end
    end
end
for i = 1 : (0.3*r)
```



```
    for j = ceil((0.35*c)):0.55*c-1
        img_thumb(i,j) = 0;
        if(img_dilate(i,j) == 0 && img_dilate(i,j+1) == 1)
            img_thumb(i,j) = 1;
```

```
        end
    end
end
img_thumb = imdilate(img_thumb,dilate);
img_thumb = medfilt2(img_thumb,[10,10]);
img_label2=bwlabel(img_thumb);
img_fing = imdilate(img_fing,dilate);
img_fing = medfilt2(img_fing,[10,10]);
img_label1=bwlabel(img_fing);
thumb_count=0;
finger_count =0;
[r,c]=size(img_label1);
for k=1:max(max(img_label1))
    cnt=0;
    for i=1:r
        for j=1:c
            if(img_label1(i,j)==k)
                cnt=cnt+1;
            end
        end
    end
    count1(k)=cnt;
    if(count1(k)>1400)
        finger_count=finger_count+1;
    end
end
if(finger_count==1)
 count3=0;
 for k=1:max(max(img_label1))
 cnt=0;
    for i=1:r
```

```
        for j=ceil(0.2*c):0.7*c
```



```
            if(img_label1(i,j)==k)
                cnt=cnt+1;
            end
        end
    end
    count3(k)=cnt;
 end
    if(max(count3)>200)
        finger_count=1;
    else
        finger_count=0;
    end
end
if(finger_count==4)
    [r,c]=size(img_label2);
    for k=1:max(max(img_label2))
        cnt=0;
        for i=1:r
            for j=1:c
                if(img_label2(i,j)==k)
                    cnt=cnt+1;
                end
            end
        end
        count2(k)=cnt;
        if(count2(k)>700)
            thumb_count=thumb_count+1;
        end
    end
```

```
end
total_fingers = finger_count+thumb_count;
figure();
subplot(241);
imshow(img);
subplot(242);
imshow(img_gray);
```



```
subplot(243);
imshow(img_white);
subplot(244);
imshow(img_erode);
subplot(245);
imshow(img_dilate);
subplot(246);
imshow(img_label1);
subplot(247);
imshow(img_label2);
subplot(248);
if(total_fingers==0)
   imshow(image0);
elseif(total_fingers==1)
   imshow(image1);
elseif(total_fingers==2)
   imshow(image2);
elseif(total_fingers==3)
   imshow(image3);
elseif(total_fingers==4)
   imshow(image4);
elseif(total_fingers==5)
   imshow(image5);
```

```
end
disp('No. of fingers:');
disp(total_fingers);
end
```

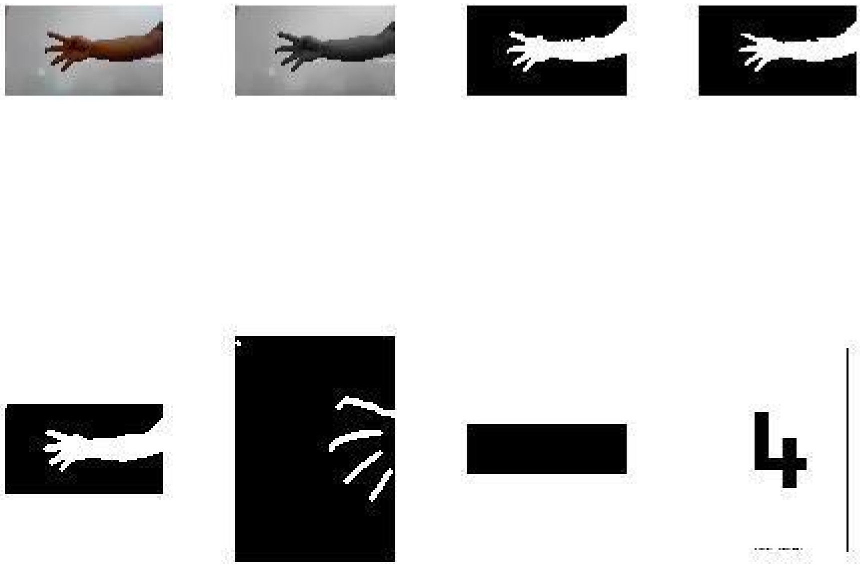

Output:

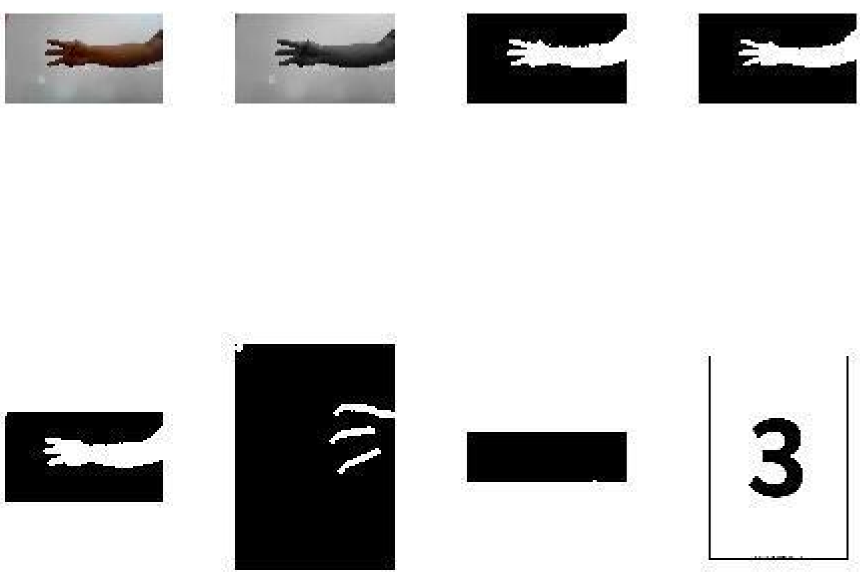

0


5


2


1


4


3

CONCLUSIONS AND FUTURE WORK

Thus, we have implemented different simple MATLAB algorithms for gesture recognition which are variant and invariant respectively to rotation. Out of all the algorithms scanning method is the robust method which delivers accurate results for 82.47% of images. In the above proposed algorithms the background is clear without any objects, this can be extended to gesture recognition with varying background. The proposed algorithms can be extended to video processing in real time. This can also be extended to interpret sign language. This MATLAB code can be converted to HDL or VHDL code and can be embedded in FPGA for hardware execution. This future work can be implemented by using Xilinx System Generator software which is linked with Xilinx FPGAs by implementing hardware co-simulation. The results obtained from the above algorithms can be used to control any electronic appliance. Some of them can be controlled the VLC media player or power point presentation without having any physical contact with the computer thus establishing a better human computer interaction. The user has to train the computer the activity or operation to be done when different gestures with different active finger count is shown. Basically we get five gestures for five finger count and assigning each gesture to each operation. This is summarized in table below:

Gesture	Operation
One	Play/Pause

Two	Volume increase
Three	Volume decrease
Four	Full screen
Five	Stop

REFERENCES

[1] S.M.Alex Raj,Sreelatha G and Supriya. M.H. “Gesture Recognition Using Field Programmable Gate Arrays”, International Conference on Devices, Circuits and Systems(ICDCS), 2012.

[2] Panwar, M., “Hand Gesture Recognition based on Shape Parameters”, 2012 International Conference on Computing, Communication and Applications (ICCCA), pp. 1-6, IEEE Conference Publication.

[3] Gonzalez, Rafael C. & R. E. Woods, Digital image processing 3rd ed c2008, Printice Hall, ISBN: 9780131687288 .

[4] Md.Rajibul Islam, Md.Shohel Sayeed, and Andrews Samraj, “Technology Review: Image Enhancement, Feature Extraction and Template Protection of a Fingerprint Authentication System,” Journal of Applied Sciences, vol. 10, no.14, pp.1397-1404, July 2010.

[5] V Elamaran, K Narasimhan, and P V M Vijayabhaskar, “Comparison of Wavelet Filters in Hybrid Domain Watermarking,” Research Journal of Information Technology, vol.5, no.3, pp.393-401, 2013.

[6] Geoff Dougherty, “Digital Image Processing for Medical Applications,” Cambridge University Press, 2009.

[7] Qidwai U, and C.H.Chen, “Digital Image Processing: An Algorithmic Approach with Matlab,” CRC Press, 2010.

[8] Gopi E.S., “Algotithm Collections for DSP Applications using MATLAB,” Springer publishers, 2007.

[9] P. R. V. Chowdary, M. N. Babu, T. V. Subbareddy, B. M. Reddy, V. Elamaran, "Image processing algorithms for gesture recognition using MATLAB", *Int. Conf. Adv. Commun. Control Comput. Technol*, Jan. 2015.