Table of Contents

[2 Introduction: 2](#_Toc100155963)

[3 Face detection methods: 2](#_Toc100155964)

[3.1 Knowledge based vs. Feature-based approach: 2](#_Toc100155965)

[3.1.1 Low-level analysis 3](#_Toc100155966)

[3.1.2 Feature analysis: 3](#_Toc100155967)

[3.1.3 Face vs no face detection: 3](#_Toc100155968)

[3.1.4 Face width and height: 3](#_Toc100155969)

[4 Implementation and testing: 3](#_Toc100155970)

[4.1 Implementation 3](#_Toc100155971)

[4.1.1 Segmentation of Object: 3](#_Toc100155972)

[4.1.2 Detection of Faces: 4](#_Toc100155973)

[4.1.3 Plotting the rectangle: 5](#_Toc100155974)

[4.2 Results: 5](#_Toc100155975)

[4.3 Bad-Lightning Condition: 5](#_Toc100155976)

[4.4 Perfect Conditions: 6](#_Toc100155977)

[4.5 False-Positives: 7](#_Toc100155978)

[4.6 CONCLUSIONS 8](#_Toc100155979)

[5 Appendix: 9](#_Toc100155980)

[5.1 Face Detection 9](#_Toc100155981)

[5.2 Testing other Images: 11](#_Toc100155982)

Face Detection Using Skin Tone Segmentation and Feature Space

# Introduction:

In recent years, face recognition has become a widely used technology that has been successfully integrated into apps. Identification of criminals in public security systems, security verification systems, credit card verification, medicine, file management, video conference, HMI systems, verification of actual holders of driver's licenses or passports, monitoring systems, automatic entry systems, and so on are just some of the applications.

Color correction, face detection, and face normalization are the three subproblems. Each of these sub-problems can be further subdivided, for example, eye detection, mouth detection, feature extraction, and so on.

# Face detection methods:

The Face detection indicates if there is a face in the input image or not. The impact of complex background, particularly the impact of skin-like background and brightness, has a significant impact on face detection using color information.

There exist many different approaches to detect faces. There are two categories for face detection:

* Knowledge-based approach
* Feature-based approaches
* Model-based approaches

In our project, we solely discuss knowledge based approach; we don't use the training model; instead, we use knowledge-based approaches.

## Knowledge based vs. Feature-based approach:

Many researchers combine these two methods to obtain a real human face because color- and feature-based detections can discover humans quickly and reliably. However, in the case of varied lighting conditions, the classic color-based method has difficulty detecting skin colour, and the normal feature-based method has a large calculation complexity. In this project, we suggest a new illumination adjustment system to solve the color-based method's difficulty and make feature-based detection easier.

This method is based on extracting features from an image and matching them against a database of face features. The three steps of the feature-based approach are as follows:

### Low-level analysis

The low-level analysis is the segmentation of visual components based on edge detection, grayscale analysis, motion, color information, and other factors. Face detection based on edge detection with noise reduction using and contrast adjustment using histogram equalization. The Sobel operator is then used to create an edge image. Because facial characteristics such as pupils, brows, and lips are darker than the surrounding area, face detection based on grayscale analysis seems promising. Contrast stretching and gray-scale morphological algorithms are used to the input image to improve the quality of dark areas. Moving dark regions such as the face and other elements of the body are retrieved by thresholding cumulative frame differences in face detection based on motion information.

### Feature analysis:

These methods, such as the Viola-Jones method, Gabor feature method, and Constellation method, are based on structural traits that exist in different poses, light changes, and multiple views that are utilized to locate faces.

### Face vs no face detection:

Following the face localization process, we can see many regions that could represent human faces. Then, for each candidate block, the features of height to width ratio, mouth, and eyes are detected sequentially. Because each of these three detections has the potential to reject candidate blocks, the low computation module is given top priority. The detection of the height to width ratio is quick and easy. If the ratio is between 1.75 and 0.75, there's a chance it's a face.

### Face width and height:

Using the width and height from the already developed databases, we determined an approach the length of pixels for each dimension which fits the face and easily crops the face out of picture. But this algorithm is not reliable and sometimes we don’t get our results accurate.

# Implementation and testing:

Note: To run this project, you need to execute the main.m file.

## Implementation

The project is solely based on three main functions which are: segmentation, detection of faces and plotting of rectangles. We will discuss this implementation one by one.

### Segmentation of Object:

First of all, we need to cut the object from the background using the thresholding concepts. But we converted the rgb image to YCbCr to adjust the better lightning intensities. This function returns the foreground pixels.

function s=segmentation(C,lower\_thresh,higher\_thresh)  
% input: Input RGB Image,  
% lower\_thresh,higher\_thresh: setting of chromatic intesity  
% paramter for objects  
% Output: Segmented image  
H=size(C,1); % height  
W=size(C,2); % width  
YCbCr=rgb2ycbcr(C); % RGB to YCbCr  
Cr=YCbCr(:,:,3);  
s=zeros(H,W);  
[SkinIndexRow,SkinIndexCol] =find(lower\_thresh<Cr & Cr<higher\_thresh); % finding the co-ordinates of foreground  
for i=1:length(SkinIndexRow)  
 s(SkinIndexRow(i),SkinIndexCol(i))=1;  
end  
figure,imshow(s);  
title('Segmented Image');  
end

### Detection of Faces:

In this function, we used the Matlab function bwlabel and regionprops for labeling and segmenting the image further according to the desired shape. The most interesting thing is height to width ratio which thresholds the non-face vs. face objects. The limits are tuned after literature review and number of tries to get the suitable results. Moreover, for elliptical or square faces, we determined an approach for face detection by setting the limits on height and width of face.

function [CurBB,foundFaces,num\_faces,bboxes]=instance\_detec(bwi)  
% Input: Foreground Pixels  
% Output: Rectangle pixels for each face, number of faces found, bounding boxes for each foreground object  
labeld = bwlabel(bwi,4); % labeling the binary image using 4-connectivity  
BB = regionprops(labeld, 'BoundingBox');  
bboxes= cat(1, BB.BoundingBox); % bounding box  
widths=bboxes(:,3);  
heights=bboxes(:,4);  
hByW=heights./widths; % New Feature for face detection  
lenRegions=size(bboxes,1); % Total number of Faces  
foundFaces=zeros(1,lenRegions);  
num\_faces=0;  
probable\_faces=label2rgb(labeld);  
figure,imshow(probable\_faces);  
title('face candidates');  
for i=1:lenRegions  
 % height to width ratio, computed above.  
 if (hByW(i)>1.75 || hByW(i)<0.75)  
 % this cannot be a mouth region. discard  
 continue;  
 end  
 % Impose a min face dimension constraint  
 if (heights(i)<30 && widths(i)<25)  
 continue;  
 end  
 % get current region's bounding box  
 CurBB=bboxes(i,:);  
 foundFaces(i)=1;  
 num\_faces=num\_faces+1;  
  
end  
end

### Plotting the rectangle:

This function plots the rectangle around the given co-ordinates.

function plot\_rectangle(bboxes,CurBB,foundFaces,num\_faces,image)  
% This function plots the rectangle around the faces.  
disp('Number of faces found :');  
disp(num\_faces);  
if (num\_faces>0)  
 indexs=find(foundFaces==1);  
 CurBB=bboxes(indexs,:);  
 disp('The Length,width,x and y features of rectangles are :')  
 disp(CurBB)  
else  
 close all;  
end  
imshow(uint8(image));  
title(['Face Detection (',num2str(num\_faces),') Faces '])  
hold on  
for i=1:size(CurBB,1)  
 BB = CurBB(i,:);  
 rectangle('Position', [BB(1),BB(2),BB(3),BB(4)],'EdgeColor','g','LineWidth',2) ;  
end  
end

## Results:

As explained in the code that some of the parameters can affect the results. For example setting the threshold for segmentation can help us in improving the accuracy for certain conditions. We will explain the results in detail as given below.

## Bad-Lightning Condition:

As seen in the image, we correctly detected the mostly faces but the problem lies for the last row faces. This is due to illuminous conditions which are not homogeneous through the picture. So the algorithm is unable to segmentate some true objects. This is one of the main drawbacks of our algorithm.

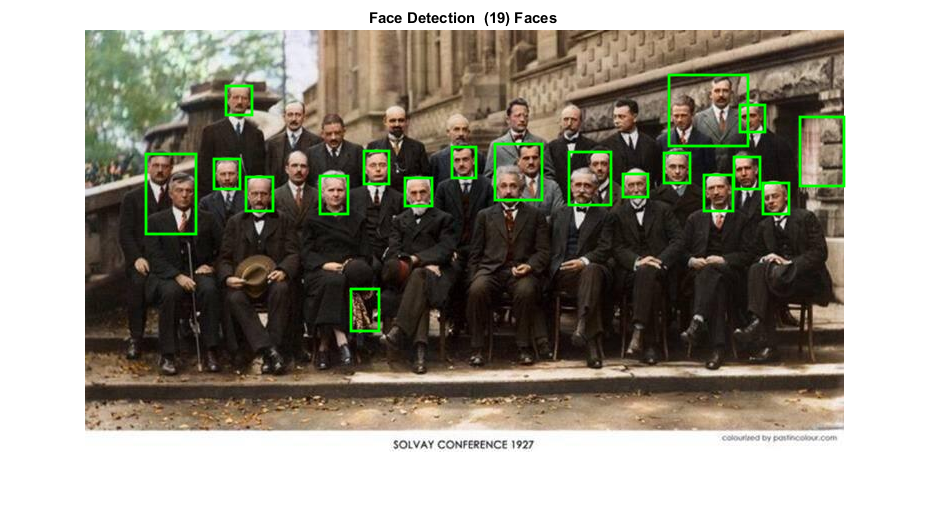


Figure : Detected Face 1

## Perfect Conditions:

In this picture the algorithm works perfectly and we get best possible results.

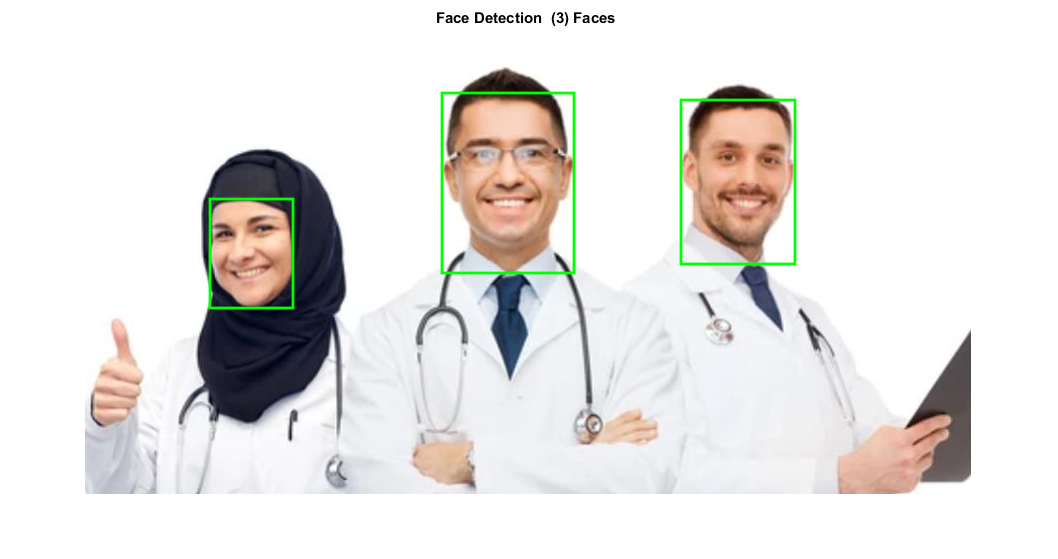


Figure : Detected Face 2

## False-Positives:

In this image, we are unable to get all the true objects and that is due to the facial feature extraction. This can also be seen as the drawback of the algorithm

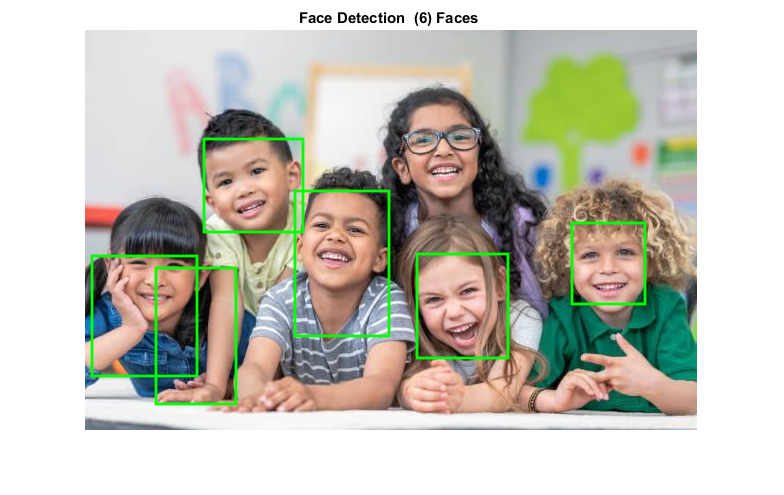


Figure : Detected Face 3

The reason of the following results is that some children are covering the face with the hand and some of them have very titled face. That’s why the feature space could not match with the given features.

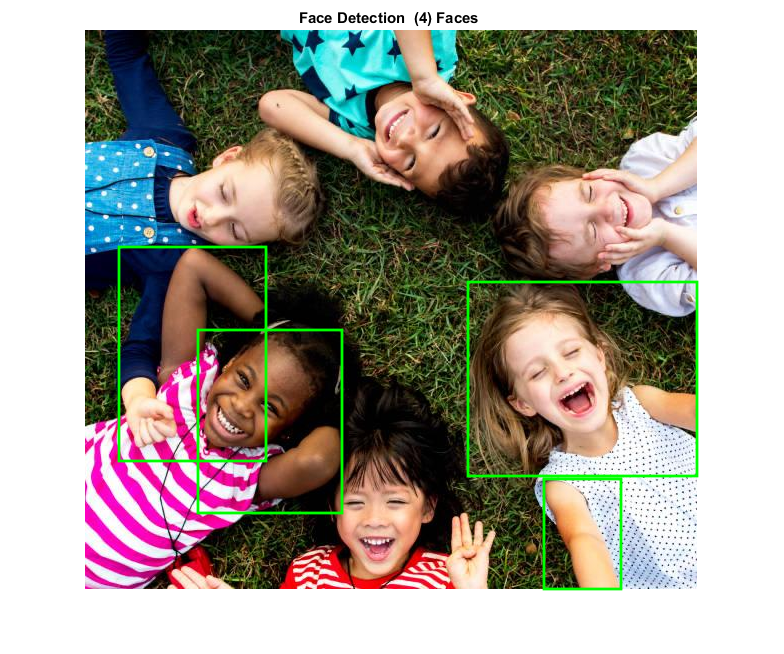


Figure : Detected Face 4

## CONCLUSIONS

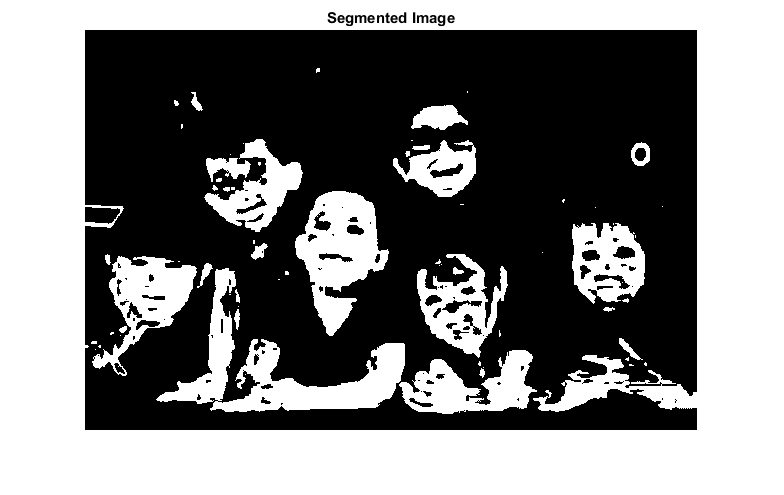
We proposed an accurate and high speed color face detection system for efficient computation. The color and feature-based detections were adopted to find skin-color fast and selected candidate blocks carefully. We used lighting compensation to improve the performance of color-based scheme, and reduce the computation of feature-based scheme.

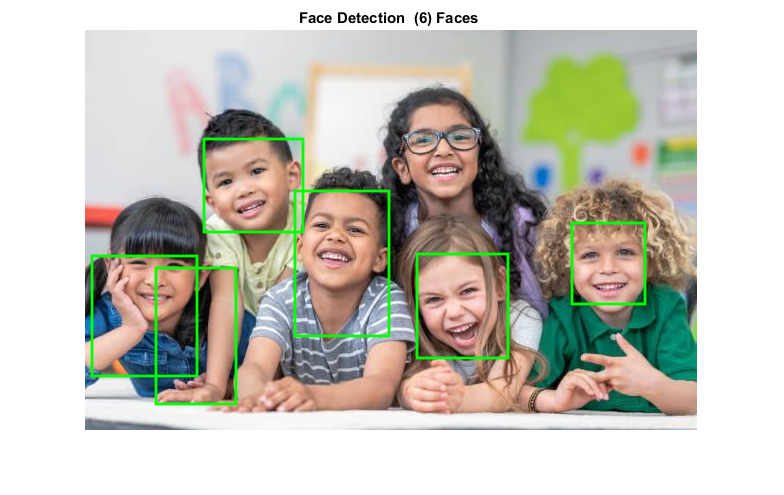
# Appendix:

## Face Detection

clc  
clear all  
close all  
I=imread('3.jfif'); % Loading the image  
% These parameters set the segmentation of image by choosing the in  
% intesity level , Ideally thes paramters should work globally for every  
% unseen image but due to limitation of this algorithm, we had to tune the  
% parameter only for the first image to get appropiraite results.  
lower\_thresh=20;  
higher\_thresh=45;  
mask=segmentation(double(I),lower\_thresh,higher\_thresh); % This function segmentates the object from background.  
[CurBB,foundFaces,num\_faces,bboxes]=instance\_detec(mask); % This function evaluates the criteria of face.  
plot\_rectangle(bboxes,CurBB,foundFaces,num\_faces,I); % This function plots the rectangle around faces

Number of faces found :  
 6  
  
The Length,width,x and y features of rectangles are :  
 7.5000 226.5000 105.0000 120.0000  
 71.5000 238.5000 80.0000 136.0000  
 119.5000 109.5000 99.0000 93.0000  
 210.5000 161.5000 94.0000 145.0000  
 332.5000 224.5000 91.0000 104.0000  
 487.5000 193.5000 73.0000 81.0000

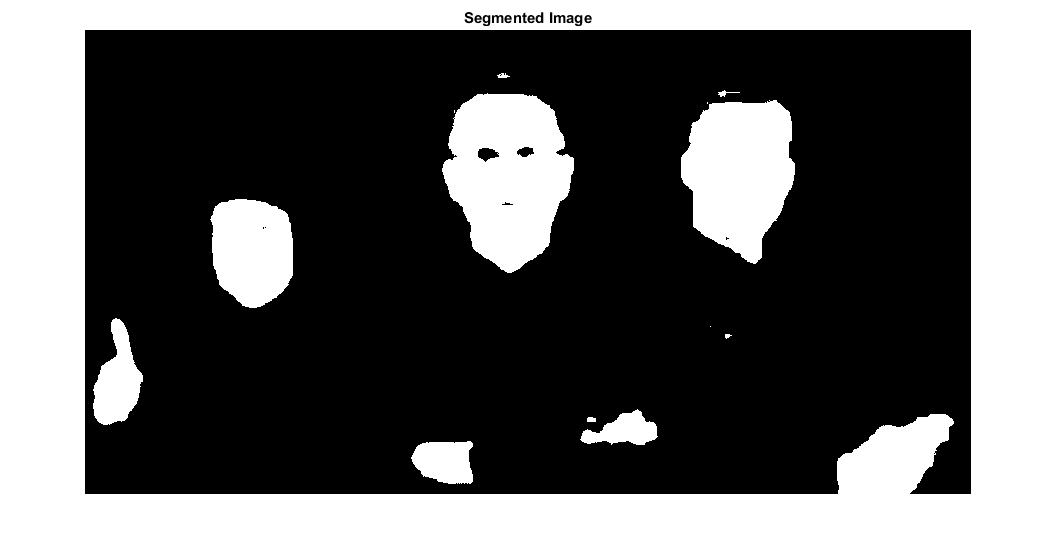


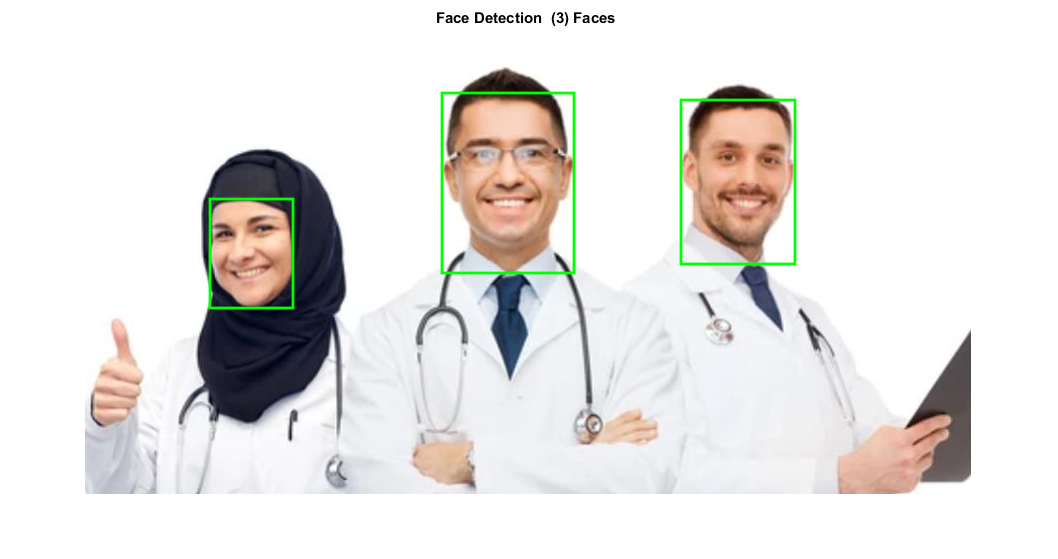


## Testing other Images:

I=imread('4.jfif');  
lower\_thresh=10;  
higher\_thresh=45;  
mask=segmentation(double(I),lower\_thresh,higher\_thresh);  
[CurBB,foundFaces,num\_faces,bboxes]=instance\_detec(mask);  
plot\_rectangle(bboxes,CurBB,foundFaces,num\_faces,I);

Number of faces found :  
 3  
  
The Length,width,x and y features of rectangles are :  
 125.5000 169.5000 83.0000 109.0000  
 357.5000 63.5000 132.0000 180.0000  
 596.5000 70.5000 114.0000 164.0000

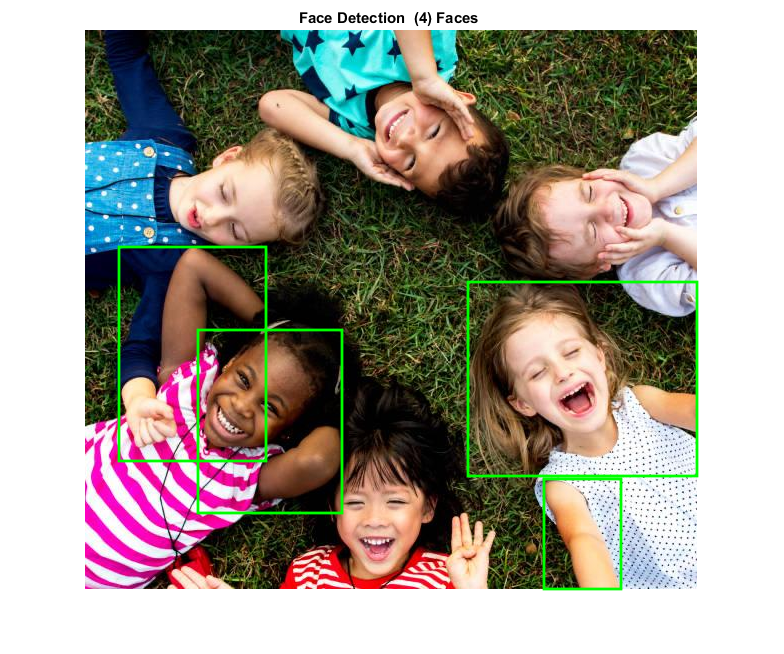




I=imread('5.jfif');  
lower\_thresh=10;  
higher\_thresh=45;  
mask=segmentation(double(I),lower\_thresh,higher\_thresh);  
[CurBB,foundFaces,num\_faces,bboxes]=instance\_detec(mask);  
plot\_rectangle(bboxes,CurBB,foundFaces,num\_faces,I);

Number of faces found :  
 4  
  
The Length,width,x and y features of rectangles are :  
 34.5000 217.5000 147.0000 214.0000  
 113.5000 300.5000 144.0000 183.0000  
 383.5000 252.5000 229.0000 194.0000  
 459.5000 449.5000 77.0000 110.0000





I=imread('6.jfif');  
lower\_thresh=10;  
higher\_thresh=45;  
mask=segmentation(double(I),lower\_thresh,higher\_thresh);  
[CurBB,foundFaces,num\_faces,bboxes]=instance\_detec(mask);  
plot\_rectangle(bboxes,CurBB,foundFaces,num\_faces,I);

Number of faces found :  
 19  
  
The Length,width,x and y features of rectangles are :  
 61.5000 124.5000 50.0000 80.0000  
 129.5000 129.5000 26.0000 30.0000  
 141.5000 56.5000 26.0000 29.0000  
 161.5000 147.5000 27.0000 34.0000  
 235.5000 146.5000 28.0000 38.0000  
 266.5000 259.5000 28.0000 42.0000  
 279.5000 121.5000 25.0000 33.0000  
 320.5000 148.5000 27.0000 28.0000  
 367.5000 117.5000 24.0000 31.0000  
 410.5000 114.5000 47.0000 56.0000  
 484.5000 122.5000 42.0000 53.0000  
 538.5000 144.5000 25.0000 23.0000  
 579.5000 123.5000 26.0000 30.0000  
 584.5000 45.5000 79.0000 71.0000  
 619.5000 145.5000 29.0000 36.0000  
 649.5000 127.5000 26.0000 32.0000  
 655.5000 75.5000 25.0000 27.0000  
 678.5000 153.5000 26.0000 31.0000  
 715.5000 87.5000 44.0000 69.0000

