Image Processing

Eyes Extraction From An Image

Face detection and Eyes extraction using Sobel Edge Detection and Morphological Operations

By Snehil Suman (N047), Tarun Tanmay (N048), Anushka Khare (N063

Roll No:	**Name:**
N047 N049 N063	Snehil Suman Tarun Tanmay Anushka Khare
Class: MBA Tech CE (3rd Year)	**Batch:** B3
Date of Experiment: 24/09/2020	**Date of Submission:** 18/10/2020
Grade:	

A. Paper Title:

Face detection and Eyes extraction using Sobel Edge Detection and Morphological Operations

B. Authors' Name:

Anamika Singh, Manminder Singh, Birmohan Singh

C. Algorithm/ Steps:

1. The first stage of the eye detection using the Sobel and Morphological functions was to read the image and convert it to grey scale.

2. Following that, the image was resized to x = 384, y = 288.

3. Part 1: Face Detection
    i. The first part was to apply **Sobel Edge Detection**
        1. For **Sobel Edge Detection**, we created two masks, running in the x and the y directions:
            a. G(x) =

-1	0	1
-2	0	2
-1	0	1

            b. G(y) =

1	2	1
0	0	0
-1	-2	-1

2. A nested ‘for’ loop was used to apply the given mask to the image

ii. Next, **Morphological** and **Conditional Dilation** was applied to the image.

1. For **Morphological Dilation**, function *imdilate()* was implemented. After rigorous experimenting, the disc size was decided to be 1. This improved the white edges of the image and made them thicker.
2. **Conditional Dilation** was used filling the holes, which filled the area of interest with a white background. The function *imfill()* utilized here.

4. Part 2: Eyes Extraction
   i. To commence the extraction of the eyes from the image, the uncropped, **Morphological Closing** was performed.
      1. For this, *imclose()* function was used.
      2. **Morphological closing** enhances the white ‘blobs’ to make them more prominent.
      3. For the structuring element, diamond shape was chosen.
   ii. **Morphological Erosion** was applied next.
      1. For this, *imeorde()* function was used.
      2. Morphological erosion helped to filter out all the white areas that were not necessary, without losing the white area that represented eyes.
      3. The disc size for this, after rigorous experimentation, was set to be ‘6’.

5. The main image (greyscale) was **Cropped** out to obtain the focus of the image; the face.
   i. The first image was halved and the first and last non-zero pixel were found, which were the pixels for the head of the person.
   ii. Post this, the topmost non-zero pixels were found.
   iii. The average of the two were calculated.
   iv. The last non-zero pixel was found after traversing the same column downwards, which was the last pixel for the cropped image.

6. After cropping the greyscale image, the **morphologically eroded image** was also **Cropped** with reference to the greyscale image.

7. After the cropping, **Centroids** of all white regions were found.
   i. For finding the centroids, *regionprops()* function was used.

8. The final step was computing the distance between each centroid and finding the right distance for the eyes.
    i. Then, the differences between the x coordinates of the centroids are found and the centroids with the minimum distance is found.
    ii. The height and width of the white regions was found by finding the last non-zero pixel in the row and column of the centroid.
    iii. Then, the location for the lowest point was found using the location of the centroid, the height and the width
    iv. Using these, a rectangular box was made around the eyes and thus, the eyes were detected

D. Code: (With all proper comments)

```
%-------------------------------------------------------------
%                  Image Processing Project
%                           On
%               Face detection and Eyes extraction
%                 using Sobel Edge Detection
%                         and
%                Morphological Operations
%-------------------------------------------------------------

clc; %Clearing the command window
close all; %clearing all the figures
clear; %erasing all the existing variables
workspace;

% Tarun Tanmay:
%-------------------------------------------------------------
%                           Input
%-------------------------------------------------------------

input_img=imread('C:\Users\Bladarc\Desktop\01_01.JPG');
%reading the input image from its directory path

%-------------------------------------------------------
%                  PREPROCESSING
%-------------------------------------------------------

%Converting to grayscale and resizing
gray_img = rgb2gray(input_img);
```

```
img=imresize(gray_img,[288,384]);
%resizing of the gray scale image into 288*384
%resized image will be used for Sobel edge detection
technique
figure('name','Figures');
subplot(3,3,1);
imshow(img);
title('Input image');
%displaying the input image figure
[row,col]=size(img);
img=double(img);
%storing the size of image in rows and columns
%converting that image in double
```

% Anushka Khare:

```
%-------------------------------------------------------------
%                          FACE DETECTION
%-------------------------------------------------------------

%Apply Sobel
sobel_x = [-1 0 1; -2 0 2; -1 0 1];
sobel_y = [1 2 1; 0 0 0; -1 -2 -1];
%finding sobel x and sobel y
sobelximg = [];
sobelyimg = [];
sobel_img = [];
for i = 1:row - 2
    for j = 1:col - 2
        sobelximg = sum(sum(sobel_x.*img(i:i+2, j:j+2)));
        sobelyimg = sum(sum(sobel_y.*img(i:i+2, j:j+2)));
        sobel_img(i+1, j+1) = sqrt(sobelximg.^2 +
sobelyimg.^2);
    end
end

%-------------------------------------------------------------
%                        Sobel Edge Detection
%                                on
%                        Gray Scale Image
%-------------------------------------------------------------

%sobel x and y used for calculating the gradient
```

```
thresholdValue = 125;
%threshold value set as 125, so that background is
separated
sobel_img = max(sobel_img, thresholdValue);
sobel_img(sobel_img == round(thresholdValue)) = 0;
%the face region is separated from background and edges are
detected
subplot(3,3,2);
imshow(sobel_img);
title('After Sobel');
%displaying the sobel image as figure
```

% Snehil Suman:

```
%-------------------------------------------------------
%               Morphological Operation: Dilation
%                           on
%                     Binary  Image
%-------------------------------------------------------

%Morphological Dilation
se = strel('disk',1);
%using the disk structuring element for performing dilation
on binary image
dil = imdilate(sobel_img,se);
subplot(3,3,3);
imshow(uint8(dil));
title('After Dilation');
%displaying the dilate image as figure

%-------------------------------------------------------
%         Morphological Operation: Conditional dilation
%                           on
%                     Dilated Image
%-------------------------------------------------------

%Conditional Dilation
xi=imfill(dil);
%fillingw white space for conditional dilation
subplot(3,3,4);
imshow(xi);
title('After CD');
%displaying the conditionally dilated image as figure
```

```
% Anushka Khare:

%-------------------------------------------------------------
%                       Image Cropping
%                            of
%                 Conditional Dilated Image
%-------------------------------------------------------------

%Face Detection using Image Cropping
%Cropping rows
[rows, columns] = find(xi)
%using the conditional dilated image for cropping
%using the find function to store rows and columns
firstcol = min(columns);
firstrow = min(rows);
lastcol=max(columns);
lastrow=max(rows);

%first and last columns and rows are calculated for non
zero pixel
firstrowfirstpixel = find(xi(firstrow, :), 1, 'first');
firstrowlastpixel = find(xi(firstrow,:), 1, 'last');

%since shoulder will have first and last non pixel which we
don't want
%the shoulder region is removed from the conditional
dilated image
avg=(firstrowfirstpixel+firstrowlastpixel)/2;
%average of top and bottom non zero pixel rows is taken
%Image rows are divided into half

last=firstrow;
x=xi(last,avg);
while(x~=0)
    last=last+1;
    x=xi(last,avg);
end

op=[];
k=1;
for i=firstrow:last
    l=1;
```

```
    for j=firstcol:lastcol
        op(k,l)=img(i,j);
        l=l+1;
    end
    k=k+1;
end
%using loop in order to divide image into half

%Cropping columns
[r,c]=size(xi);
%again storing conditional dilated image's rows and columns
r=floor(r/2);
%r is the middle row of whole image where face is divided
b=xi(1:r,:);
%storing first to mid row, and all columns in b
[r1,c1]=find(b);
min=min(c1);
max=max(c1);
%finding min and max non zero pixel column range
%since shoulder region was removed,
%therefore non-zero pixel first and last columns are near
ears
a=op(:,min:max);
subplot(3,3,5);
imshow(uint8(a));
%image is cropped from first to last column
title('Cropped Image');
%displaying cropped image as figure
```

% Tarun Tanmay:

```
%-------------------------------------------------------------
%                          EYES EXTRACTION
%                  Morphological Operation: Closing
%                                on
%                           Dilated image
%-------------------------------------------------------------

%Morphological Closing
se1=strel("diamond",3);
%using the diamond structuring element for eyes extraction
closed_img=imclose(dil,se1);
%closing operation is performed on dilated image
```

```
subplot(3,3,6);
imshow(closed_img);
title('After Closing');
%closing imahge is displayed as figure
```

% Snehil Suman:

```
%-------------------------------------------------------
%                          EYES EXTRACTION
%                  Morphological Operation: Erosion
%                                on
%                          Closing image
%-------------------------------------------------------

%Morphological Erosion
se3=strel('disk',6);
%disk is used to perform erosion operation
eroded_img=imerode(closed_img,se3);
%erosion is performed on the closing image
subplot(3,3,7);
imshow(eroded_img);
title('After Erosion');
%displaying the eroded image as figure
```

% Anushka Khare:

```
%-------------------------------------------------------
%                      Cropping Eroded Image
%-------------------------------------------------------

cropped_eroded =[];
%storing the copped image in a matrix
k=1;
%initiating k variable as 1 for using in loop
for i=firstrow:last
    l=1;
    for j= min:max
        cropped_eroded(k,l)= eroded_img(i,j);
        l=l+1;
    end
    k=k+1;
end
```

```
%displaying cropped_eroded image

% Snehil Suman:


%-------------------------------------------------------------
%                     Centroid
%-------------------------------------------------------------

label = bwlabel(cropped_eroded);
stat = regionprops(label,'centroid');
%finding centroid of the cropped_eroded image
subplot(3,3,8);
imshow(cropped_eroded); hold on;
title('Eroded: Cropped')
for x = 1: numel(stat)
    plot(stat(x).Centroid(1),stat(x).Centroid(2),'ro');
end
for i=1:length(stat)
    x_centroid(i) = stat(i).Centroid(1);  %y coordinates
    y_centroid(i) = stat(i).Centroid(2);  %x coordinates
end

% Anushka Khare and Snehil Suman:

%-------------------------------------------------------------
%              Finding distances between centroids
%-------------------------------------------------------------

diff=[];
for i=1:length(y_centroid)
    for j=1:length(y_centroid)
        if i>j
            diff(i,j)=abs(y_centroid(i)-y_centroid(j));
        else
            diff(i,j)=0;
        end
    end
end
% we have calculated the absolute distance between the
centroids
```

```
%-------------------------------------------------------------
%              Finding centroids with minimum x distance
%-------------------------------------------------------------

dmin=[double(diff(2,1)) 2 1];
for i=1:length(y_centroid)
    for j=1:length(y_centroid)
        if diff(i,j)<dmin(1) && i>j
            dmin=[double(diff(i,j)) i j];
        end
    end
end
```

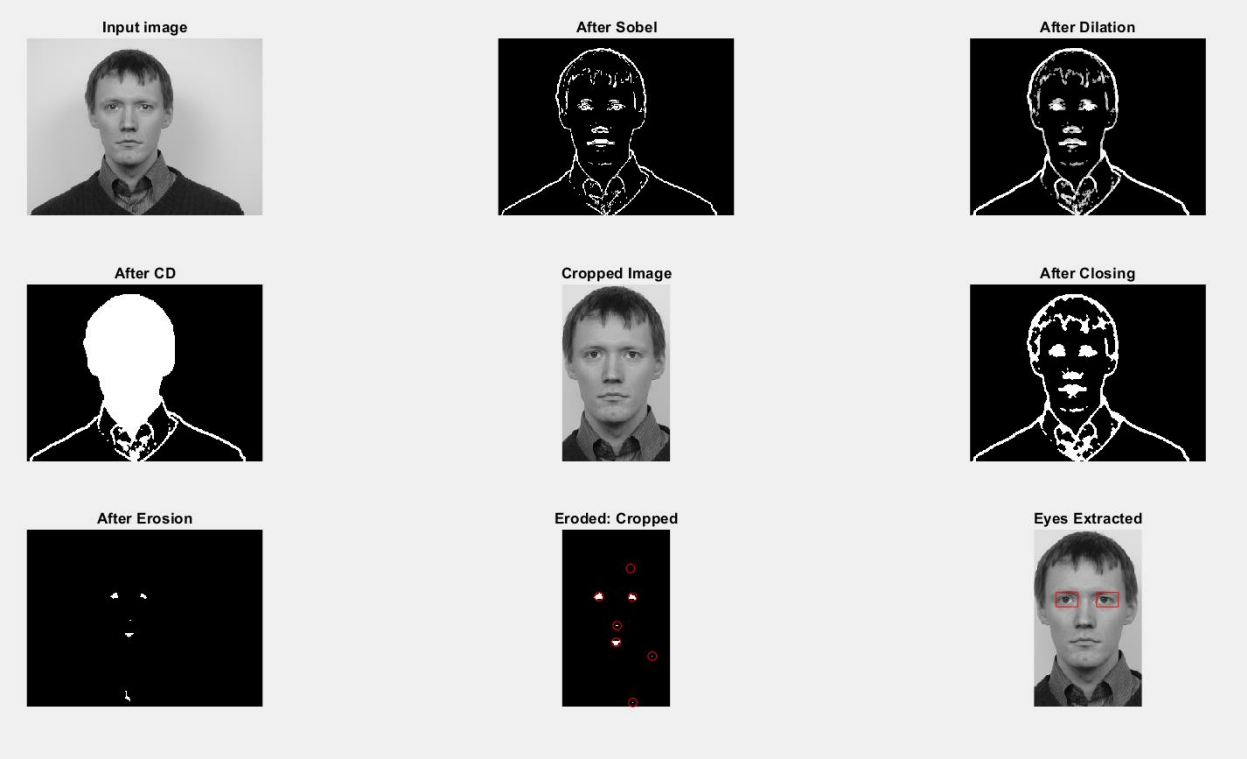
% Anushka Khare:

```
%-------------------------------------------------------------
%                       Eye Detection
%                      Displaying Box
%-------------------------------------------------------------

%Making box around one eye
x=floor(x_centroid(dmin(2)));
y=floor(y_centroid(dmin(2)));
r=cropped_eroded(y,x);
while r~=0
    x=x+1;
    r=cropped_eroded(floor(y_centroid(dmin(2))),x);
end
h=ceil(x-x_centroid(dmin(2)));
s=cropped_eroded(y,floor(x_centroid(dmin(2))));
while s~=0
    y=y+1;
    s=cropped_eroded(y,floor(x_centroid(dmin(2))));
end
w=ceil(y-y_centroid(dmin(2)));

% ------------------------------
% Displaying image and one box
%-------------------------------
subplot(3,3,9);
imshow(uint8(a));
title('Eyes Extracted');
```

```
rectangle('position',[floor(y_centroid(dmin(2)))-2*w
floor(x_centroid(dmin(2)))-2*h w*10 h*4],'EdgeColor','r');
%displaying a rectangle shaped box around the eye region

% -------------------------
% Making box around one eye
% -------------------------

x=floor(x_centroid(dmin(3)));
y=floor(y_centroid(dmin(3)));
r=cropped_eroded(y,x);
while r~=0
    x=x+1;
    r=cropped_eroded(floor(y_centroid(dmin(3))),x);
end
h=ceil(x-x_centroid(dmin(3)));
s=cropped_eroded(y,floor(x_centroid(dmin(3))));
while s~=0
    y=y+1;
    s=cropped_eroded(y,floor(x_centroid(dmin(3))));
end
w=ceil(y-y_centroid(dmin(3)));
%the cropped_eroded image's centroids are calculated
%boxes are made around the eye region for detection

%Displaying other box
rectangle('position',[floor(x_centroid(dmin(3)))-4*h
floor(y_centroid(dmin(3)))-w h*6 w*4], 'EdgeColor','r');
%displaying rectangle box for detection of another eye
```

Input Image:

Output:


Input image
After Sobel
After Dilation
After CD
Cropped Image
After Closing
After Erosion
Eroded: Cropped
Eyes Extracted