Vishesh Chanana 665319085

COMPUTER VISION

Homerwork 4

Answer 1)

Original Image

Image after thresholding

Using the connected components algorithm we find that there are 7 connected components in the figure. We can verify this even by looking at this figure. The first connected component is the connected background, the black letters are the other three connected components and the white holes between the letters are the other three connected components.

Final image after substituting the binary image with the component that they belong to and then mapping it on a scale of 255.

The algorithm first labelled all the components by checking the northern and the western neighbor. After that it found out all the components that are connected to each other. In the next scan of the image, all the connected components were given the same label. Then, I mapped this range to a new range of 0 to 255. Thus, we can see the different shades in the final image.

Answer 2)

Display image

The original image the level 1 of both the pyramids

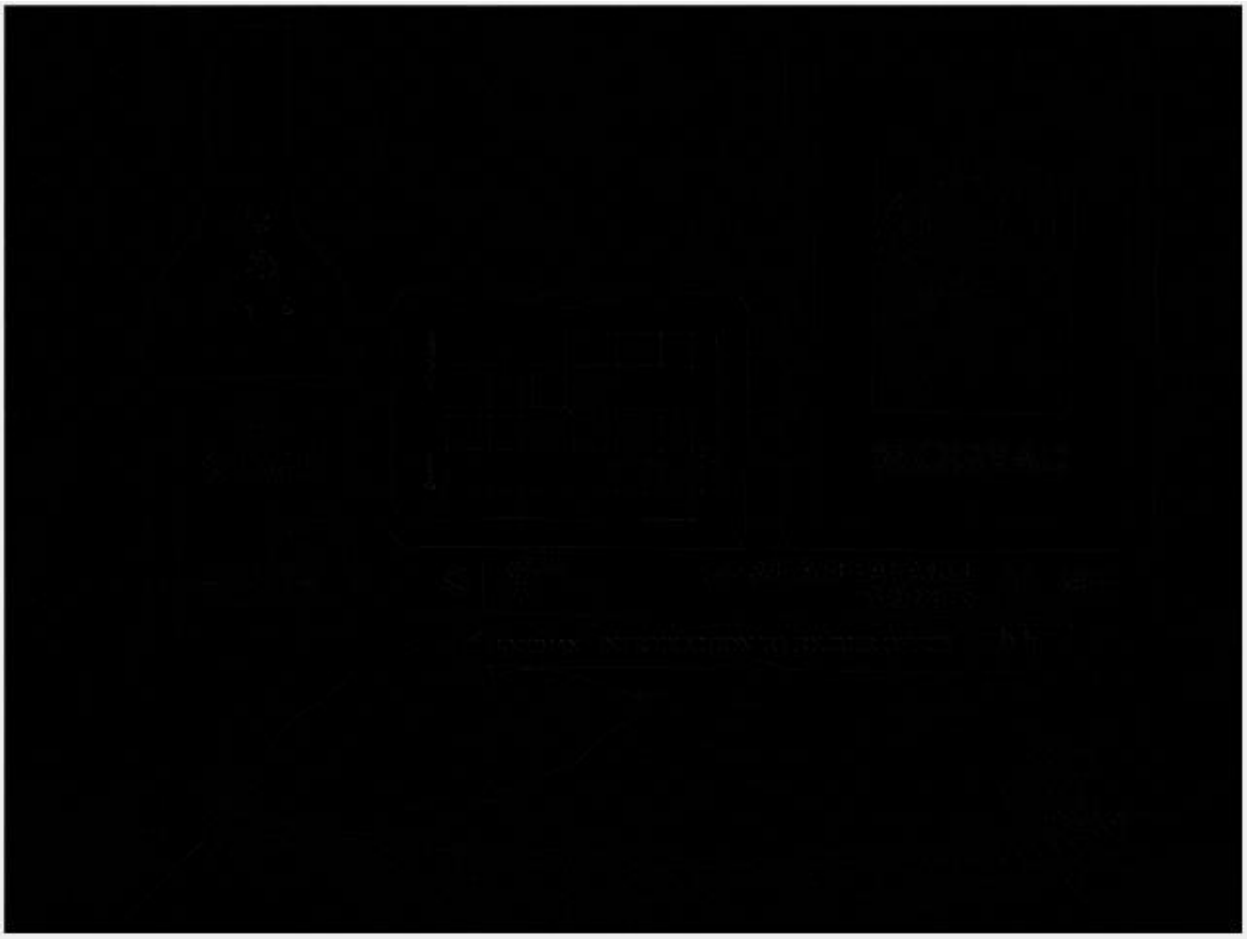
Gaussian pyramid level 2

Gaussian pyramid level 3

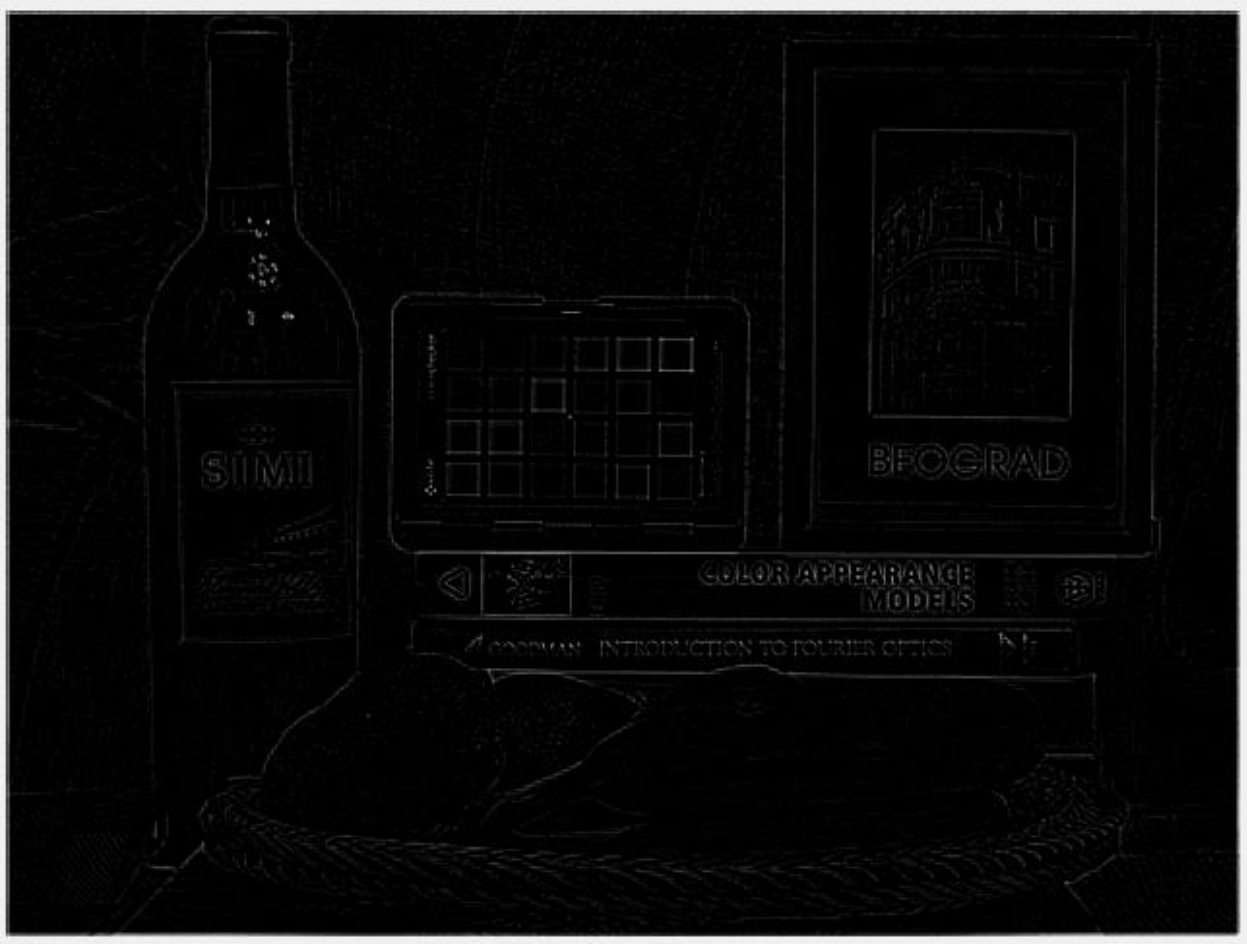
Gaussian pyramid level 4

Gaussian pyramid level 5

Laplacian pyramid level 2



Laplacian pyramid level 3



Laplacian pyramid level 4

Laplacian pyramid level 5

On applying the Gaussian pyramid to our image, we get the final image that is smoothened. This is basically a blurred image or an image that has the lower frequency content. We get the lower frequency content by using a binomial kernel and scaling the image to its half every time.

When we apply the Laplacian Kernel, we get a sharpened image or the edges of the image. This is done by selecting only the high frequency content. This is done by applying a Binomial Kernel and scaling down the image to half its original value. This scaled down image is then subtracted from the original to get the final output.

CODE

Answer 1)

```
clc;
clear all;
original_img = imread('Connected.bmp');
size_of_image = size(original_img);
no_of_rows = size_of_image(1);
no_of_cols = size_of_image(2);
imshow(original_img);
label = 1;
num = 1;

for i = 1:no_of_rows
    for j = 1:no_of_cols
        if(original_img(i,j) > 0)
            threshold_img(i,j) = 1;
        else
            threshold_img(i,j) = original_img(i,j);
        end
    end
end
imshow(threshold_img);

for i = 1:no_of_rows
    for j = 1:no_of_cols
        if (i ==1)
            if(j==1)
```

```
                inter_img(i,j) = label;
            else
                if((threshold_img(i,j) == threshold_img(i,j-1)))
                    inter_img(i,j) = inter_img(i,j-1);
                else
                    label = label+1;
                    inter_img(i,j) = label;
                end
            end
        else
            if(j==1)
                if(threshold_img(i,j) == threshold_img(i-1,j))
                    inter_img(i,j) = inter_img(i-1,j);
                else
                    label = label+1;
                    inter_img(i,j) = label;
                end
            else
                if(threshold_img(i,j) ~= threshold_img(i,j-1) &&
threshold_img(i,j) ~= threshold_img(i-1,j))
                    label = label+1;
                    inter_img(i,j) = label;
                else
                    if(threshold_img(i,j) == threshold_img(i,j-1) &&
threshold_img(i,j) == threshold_img(i-1,j))
                        inter_img(i,j) = min(inter_img(i,j-1),inter_img(i-
1,j));
                        if(inter_img(i,j-1) ~= inter_img(i-1,j))
                            match1(num) = inter_img(i-1,j);
                            match2(num) = inter_img(i,j-1);
                            num = num+1;
                        end
                    else
                        if(threshold_img(i,j) == threshold_img(i-1,j))
                            inter_img(i,j) = inter_img(i-1,j);
                        else
                            inter_img(i,j) = inter_img(i,j-1);
                        end
                    end
                end
            end
        end
    end
end

labels = zeros(max(max(inter_img)),1);
pixel = 1;
for i = 1:length(match1)
   if(labels(match1(i)) == 0 && labels(match2(i)) == 0)
       labels(match1(i)) = min((match1(i)),(match2(i)));
       labels(match2(i)) = min((match1(i)),(match2(i)));
   elseif(labels(match1(i)) == 0 || labels(match2(i)) == 0)
       if(labels(match1(i)) ~= 0)
           labels(match2(i)) = labels(match1(i));
       else
           labels(match1(i)) = labels(match2(i));
       end
```

```
   elseif(labels(match1(i)) ~= 0 && labels(match2(i)) ~= 0)
       previous = max(labels(match1(i)),labels(match2(i)));
       labels(match1(i))= min(labels(match1(i)),labels(match2(i)));
       labels(match2(i)) = min(labels(match1(i)),labels(match2(i)));
       for k=1:length(labels)
           if(labels(k) == previous)
               labels(k) = min(labels(match1(i)),labels(match2(i)));
           end
       end
   end
end

for i=1:no_of_rows
    for j=1:no_of_cols
        if(labels(inter_img(i,j)) == 0)
            final_img(i,j) = inter_img(i,j);
        else
            final_img(i,j) = labels(inter_img(i,j));
        end
    end
end

sum = 0;
for i=1:length(labels)
    if(labels(i) == 0)
        sum = sum+1;
    end
end

if (sum==0)
    no_of_connected_components = length(unique(labels));
else
    no_of_connected_components = length(unique(labels)) - 1 + sum;
end

max_value=max(max(final_img));
final = round(final_img*(256/max_value+1));
imshow(uint8(final));
```

Answer 2)

For Gaussian pyramid

```
clc;
close all;
original_img = imread('image.bmp');
imshow(original_img);
gaussian_kernel=[1/16 1/4 6/16 1/4 1/16];
padded_img = padarray(original_img,[2 2], 0);

for i=3:+2:956
    for j=3:+2:1276
```

```
        final_img((i),(j))= gaussian_kernel(1,1)*padded_img(i,j-2) +
gaussian_kernel(1,2)*padded_img(i,j-1) + gaussian_kernel(1,3)*padded_img(i,j)
+ gaussian_kernel(1,4)*padded_img(i,j+1) +
gaussian_kernel(1,5)*padded_img(i,j+2);
    end
end
temp1=final_img(3:2:end,:);
temp2=transpose(temp1);
final_img=temp2(3:2:end,:);
final_img=transpose(final_img);
imshow(final_img);

padded_img = padarray(final_img,[2 2],0);
for i=3:+2:479
    for j=3:+2:639
        final_img((i),(j))= gaussian_kernel(1,1)*padded_img(i,j-2) +
gaussian_kernel(1,2)*padded_img(i,j-1) + gaussian_kernel(1,3)*padded_img(i,j)
+ gaussian_kernel(1,4)*padded_img(i,j+1) +
gaussian_kernel(1,5)*padded_img(i,j+2);
    end
end
temp1=final_img(3:2:end,:);
temp2=transpose(temp1);
final_img=temp2(3:2:end,:);
final_img=transpose(final_img);
imshow(final_img);

padded_img = padarray(final_img,[2 2],0);
for i=3:+2:241
    for j=3:+2:321
        final_img((i),(j))= gaussian_kernel(1,1)*padded_img(i,j-2) +
gaussian_kernel(1,2)*padded_img(i,j-1) + gaussian_kernel(1,3)*padded_img(i,j)
+ gaussian_kernel(1,4)*padded_img(i,j+1) +
gaussian_kernel(1,5)*padded_img(i,j+2);
    end
end
temp1=final_img(3:2:end,:);
temp2=transpose(temp1);
final_img=temp2(3:2:end,:);
final_img=transpose(final_img);
imshow(final_img);

padded_img = padarray(final_img,[2 2],0);
for i=3:+2:122
    for j=3:+2:162
        final_img((i),(j))= gaussian_kernel(1,1)*padded_img(i,j-2) +
gaussian_kernel(1,2)*padded_img(i,j-1) + gaussian_kernel(1,3)*padded_img(i,j)
+ gaussian_kernel(1,4)*padded_img(i,j+1) +
gaussian_kernel(1,5)*padded_img(i,j+2);
    end
end
temp1=final_img(3:2:end,:);
temp2=transpose(temp1);
final_img=temp2(3:2:end,:);
final_img=transpose(final_img);
imshow(final_img);
```

Laplacian pyramid

```
clc;
close all;
original_img= imread('image.bmp');
imshow(original_img);
title('Original Image');
Laplacian_Kernel=[1/16 1/4 6/16 1/4 1/16];
padded_img = padarray(original_img,[2 2],0);
for i=3:956
    for j=3:1276
        final_img((i-2),(j-2))= Laplacian_Kernel(1,1)*C(i,j-2) +
Laplacian_Kernel(1,2)*C(i,j-1) + Laplacian_Kernel(1,3)*C(i,j) +
Laplacian_Kernel(1,4)*C(i,j+1) + Laplacian_Kernel(1,5)*C(i,j+2);
    end
end
final_img=original_img-final_img;
figure;
imshow(final_img);
temp1=final_img(1:2:end,:);
temp2=transpose(temp1);
final_img=temp2(1:2:end,:);
final_img=transpose(final_img);

padded_img = padarray(final_img,[2 2],0);
inter_img=final_img;
for i=3:479
    for j=3:639
        final_img((i-2),(j-2))= Laplacian_Kernel(1,1)*C(i,j-2) +
Laplacian_Kernel(1,2)*C(i,j-1) + Laplacian_Kernel(1,3)*C(i,j) +
Laplacian_Kernel(1,4)*C(i,j+1) + Laplacian_Kernel(1,5)*C(i,j+2);
    end
end
final_img=inter_img-final_img;
figure;
imshow(final_img);
temp1=final_img(1:2:end,:);
temp2=transpose(temp1);
final_img=temp2(1:2:end,:);
final_img=transpose(final_img);

padded_img = padarray(final_img,[2 2],0);
inter_img=final_img;
for i=3:241
    for j=3:321
        final_img((i-2),(j-2))= Laplacian_Kernel(1,1)*C(i,j-2) +
Laplacian_Kernel(1,2)*C(i,j-1) + Laplacian_Kernel(1,3)*C(i,j) +
Laplacian_Kernel(1,4)*C(i,j+1) + Laplacian_Kernel(1,5)*C(i,j+2);
    end
end
final_img=inter_img-final_img;
figure;
imshow(final_img);
```

```
temp1=final_img(1:2:end,:);
temp2=transpose(temp1);
final_img=temp2(1:2:end,:);
final_img=transpose(final_img);
padded_img = padarray(final_img,[2 2],0);
inter_img=final_img;
for i=3:122
    for j=3:162
        final_img((i-2),(j-2))= Laplacian_Kernel(1,1)*C(i,j-2) +
Laplacian_Kernel(1,2)*C(i,j-1) + Laplacian_Kernel(1,3)*C(i,j) +
Laplacian_Kernel(1,4)*C(i,j+1) + Laplacian_Kernel(1,5)*C(i,j+2);
    end
end
final_img=inter_img-final_img;
imshow(final_img);
```