CSE3018 : Content Based Image and Video Retrieval

Lab - 5

Naga Harshith Bezawada
19BCE1547

Each Sheet in the Excel Contains output of each Code.
Challenging Task in Sheet 6

Sl.No.	Method	Feature Vector Length	Time Taken
1	Gray Scale with 256 gray levels	2*2*256 = 1024	3.031303 seconds
2	Gray Scale with 8 gray levels	2*2*8 = 32	2.319719 seconds
3	Gray Scale with 16 gray levels	2*2*16 = 64	2.165261 seconds
4	Gray Scale with 32 gray levels	2*2*32 = 128	2.301229 seconds
5	Gray Scale with 64 gray levels	2*2*64 = 256	2.498560 seconds
6	Color Image with R,G,B, each 256 levels	3*2*2*256 = 3076	5.638420 seconds

Q 1. Implement a CBIR system that uses features derived from Color Auto Correlogram Descriptors.0

CODE : Input - blue10.jpg

```
tic;

D = './images';

S = dir(fullfile(D,'*.jpg'));

% Loading query image and converting to gray scale
```

```
query_image = imread('images/blue10.jpg');
query_image = rgb2gray(query_image);
Q_Row={'Query'};
%Extracting Horizontal and Vertical count of Query Image
[hc vc] = acg(query_image,[1 3]);
for i = 1:256
Q_Row{end+1} = hc(i,1);
End
for i = 1:256
Q_Row{end+1} = vc(i,1);
end
for i = 1:256
Q_Row{end+1} = hc(i,2);
end
for i = 1:256
Q_Row{end+1} = vc(i,2);
end
%Creating array with Column Names for Excel Sheet
CNames = {'file_name'};
for i = 1:256
CNames{end+1} = sprintf('H-1-%d,%d', i-1, i-1);
end
for i = 1:256
CNames{end+1} = sprintf('V-1-%d,%d', i-1, i-1);
end
for i = 1:256
CNames{end+1} = sprintf('H-2-%d,%d', i-1, i-1);
```

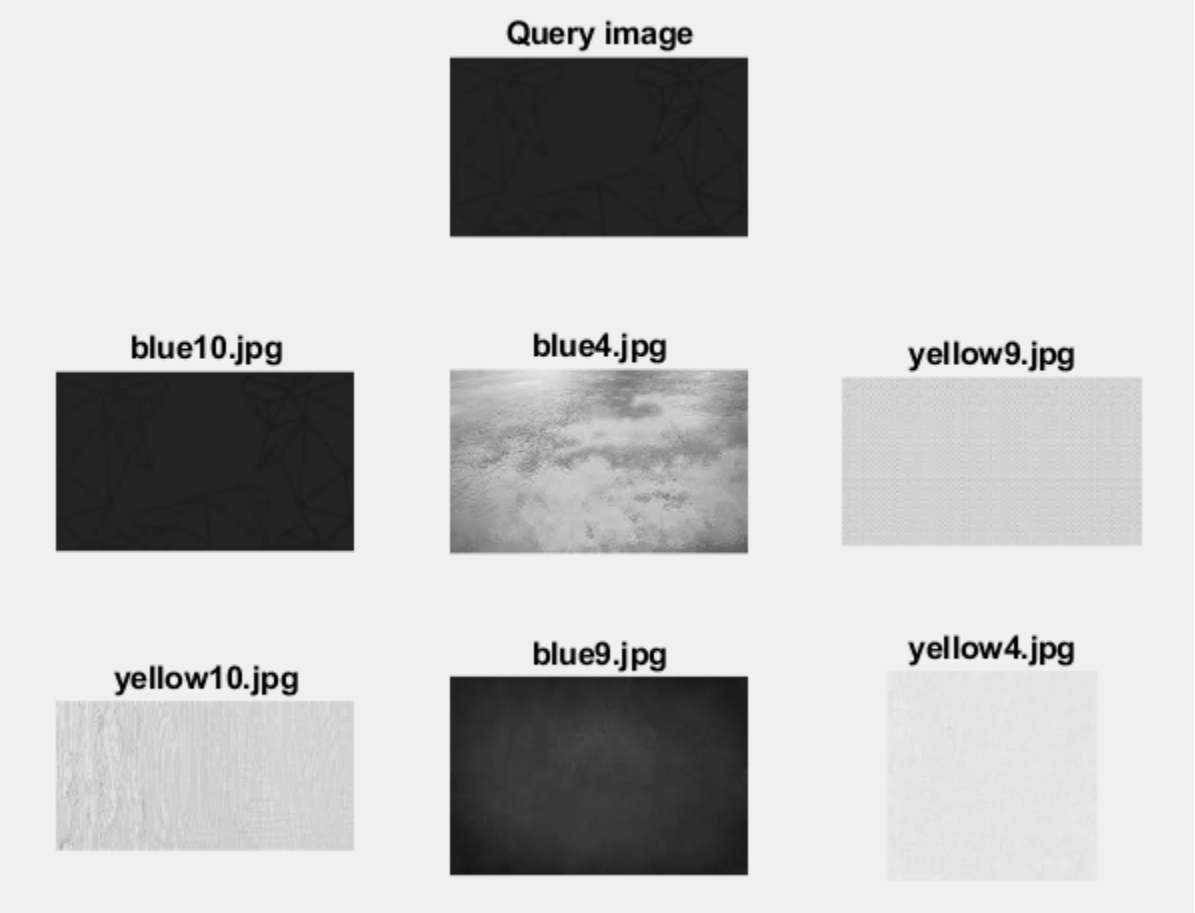
```
end
for i = 1:256
CNames{end+1} = sprintf('V-2-%d,%d', i-1, i-1);
end
CNames{end+1} = 'Chi-Square Distance';
info_table = cell2table(cell(0, 1026), 'VariableNames',CNames);
% Looping through all the images in the directory
for k = 1:numel(S)
F = fullfile(D,S(k).name);
I = imread(F);
I = rgb2gray(I);
S(k).data = I; % optional, save data.
I_Row={S(k).name};
[ihc ivc] = acg(I,[1 3]);
for i = 1:256
     I_Row{end+1} = ihc(i,1);
end
for i = 1:256
     I_Row{end+1} = ivc(i,1);
end
for i = 1:256
     I_Row{end+1} = ihc(i,2);
end
for i = 1:256
     I_Row{end+1} = ivc(i,2);
end
     tot_sum =0;
```

```
      for i = 2:1025
      num = (Q_Row{i} - I_Row{i})^2;
      denum = Q_Row{i} + I_Row{i};
      if(denum==0)
            csd = 0;
      else
            csd = num/denum;
      end
      tot_sum = tot_sum + csd;
      %disp(I_Row(1)+"-"+Q_Row{i}+"diff. sq."+I_Row{i}+"->"+tot_sum);
      end
      tot_sum = tot_sum * 0.5;
      I_Row{end+1} = tot_sum;
      info_table = [info_table;I_Row];
end
info_table = sortrows(fillmissing(info_table, 'previous'),
'Chi-Square Distance');
writetable(info_table, 'lab5_1.xlsx','Sheet',1);
% Displaying the first 5 nearest image
subplot(3, 3, 2);
imshow(query_image);
title('Query image');
% Extracting the filenames of the images
file_names = info_table(:, 'file_name').file_name;
for i = 1:6
F = fullfile(D,char(file_names(i)));
I = imread(F);
I = rgb2gray(I);
```

```
subplot(3, 3, i+3);

imshow(I);

title(char(file_names(i)));

end

toc;

function [horizontal_count, vertical_count] = acg(img, distances,
levels)

% Check if levels provided or not

if nargin == 2

levels = 256;

end

[Y, X] = size(img);

% Image quantization

img = gray2ind(img, levels);

% Set variable sizes

[~, num_of_distances] = size(distances);

horizontal_count = zeros(levels, num_of_distances);

vertical_count = zeros(levels, num_of_distances);

% For each row

for r = 1:Y

% For each column

     for c = 1:X

% For each distance

          for d = 1:num_of_distances

               D = distances(d);

               value = img(r,c); % Get the value

               % Increment the resp. counter, if pixels equivalent

               if(r + D <= Y && img(r + D, c) == value)
```

```
                horizontal_count(value+1, d) =
horizontal_count(value+1, d) + 1;

                end

                if(c + D <= X && img(r, c + D) == value)

                vertical_count(value+1, d) = vertical_count(value+1,
d) + 1;

                end

            end

      end

end

end
```

OUTPUT :



Q2. A. Quantize the Image using *imquantize( )* to 8 gray levels

CODE :

```
tic;
D = './images';
S = dir(fullfile(D,'*.jpg')); % pattern to match filenames.

% Loading query image and converting to gray scale
query_image = imread('images/blue4.jpg');
querry_image = rgb2gray(query_image);
query_image = imquantize(querry_image,8);
Q_Row={'Query'};
%Extracting Horizontal and Vertical count of Query Image
[hc vc] = acg(query_image,[1 3]);
for i = 1:8
Q_Row{end+1} = hc(i,1);
end
for i = 1:8
Q_Row{end+1} = vc(i,1);
end
for i = 1:8
Q_Row{end+1} = hc(i,2);
end
for i = 1:8
Q_Row{end+1} = vc(i,2);
end
%Q_Row{end+1} = 0;
%Creating array with Column Names for Excel Sheet
```

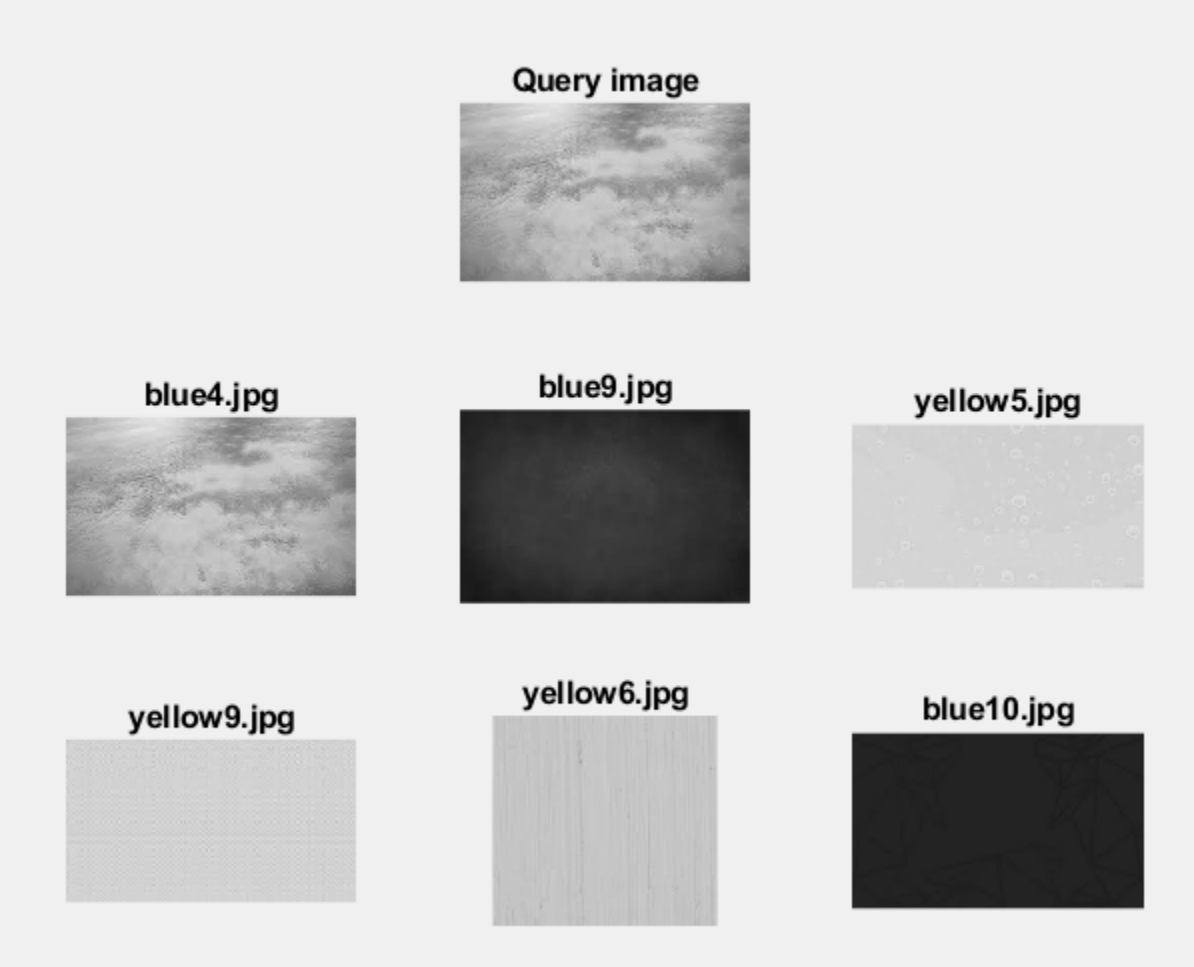
```
CNames = {'file_name'};
for i = 1:8
CNames{end+1} = sprintf('H-1-%d,%d', i-1, i-1);
end
for i = 1:8
CNames{end+1} = sprintf('V-1-%d,%d', i-1, i-1);
end
for i = 1:8
CNames{end+1} = sprintf('H-2-%d,%d', i-1, i-1);
end
for i = 1:8
CNames{end+1} = sprintf('V-2-%d,%d', i-1, i-1);
end
CNames{end+1} = 'Chi-Square Distance';
info_table = cell2table(cell(0, 34), 'VariableNames',CNames);
%info_table=[info_table;Q_Row];
% Looping through all the images in the directory
for k = 1:numel(S)
F = fullfile(D,S(k).name);
I = imread(F);
I = rgb2gray(I);
I = imquantize(I,8);
S(k).data = I; % optional, save data.
I_Row={S(k).name};
[ihc ivc] = acg(I,[1 3]);
for i = 1:8
     I_Row{end+1} = ihc(i,1);
```

```
end
for i = 1:8
     I_Row{end+1} = ivc(i,1);
end
for i = 1:8
     I_Row{end+1} = ihc(i,2);
end
for i = 1:8
     I_Row{end+1} = ivc(i,2);
end
     tot_sum =0;
     for i = 2:33
     num = (Q_Row{i} - I_Row{i})^2;
     denum = Q_Row{i} + I_Row{i};
     if(denum==0)
          csd = 0;
     else
          csd = num/denum;
     end
     tot_sum = tot_sum + csd;
     %disp(I_Row(1)+"-"+Q_Row{i}+"diff. sq."+I_Row{i}+"->"+tot_sum);
     end
     tot_sum = tot_sum * 0.5;
     I_Row{end+1} = tot_sum;
     info_table = [info_table;I_Row];
end
```

```
% Replacing the NaN with values in the previous cell and replacing
the

% rows in the table in the ascending order of city block distance

info_table = sortrows(fillmissing(info_table, 'previous'),
'Chi-Square Distance');

writetable(info_table, 'lab5_1.xlsx','Sheet',2);


% Displaying the first 5 nearest image

subplot(3, 3, 2);

imshow(querry_image);

title('Query image');

% Extracting the filenames of the images

file_names = info_table(:, 'file_name').file_name;

for i = 1:6

F = fullfile(D,char(file_names(i)));

I = imread(F);

I = rgb2gray(I);

subplot(3, 3, i+3);

imshow(I);

title(char(file_names(i)));

end


toc;

function [horizontal_count, vertical_count] = acg(img, distances,
levels)

% Check if levels provided or not

if nargin == 2

levels = 8;

end
```

```
[Y, X] = size(img);
% Image quantization
% img = gray2ind(img, levels);
% Set variable sizes
[~, num_of_distances] = size(distances);
horizontal_count = zeros(levels, num_of_distances);
vertical_count = zeros(levels, num_of_distances);
% For each row
for r = 1:Y
% For each column
     for c = 1:X
% For each distance
           for d = 1:num_of_distances
                D = distances(d);
                value = img(r,c); % Get the value
                % Increment the resp. counter, if pixels equivalent
                if(r + D <= Y && img(r + D, c) == value)
                horizontal_count(value+1, d) =
horizontal_count(value+1, d) + 1;
                end
                if(c + D <= X && img(r, c + D) == value)
                vertical_count(value+1, d) = vertical_count(value+1,
d) + 1;
                end
           end
     end
End end
```

OUTPUT :



Q2. B. Quantize the Image using *imquantize( )* to 16 gray levels

CODE : Input - Yellow2.jpg

```
tic;

D = './images';

S = dir(fullfile(D,'*.jpg')); % pattern to match filenames.


% Loading query image and converting to gray scale

query_image = imread('images/yellow2.jpg');

querry_image = rgb2gray(query_image);

query_image = imquantize(querry_image,16);
```

```
Q_Row={'Query'};
%Extracting Horizontal and Vertical count of Query Image
[hc vc] = acg(query_image,[1 3]);
for i = 1:16
Q_Row{end+1} = hc(i,1);
end
for i = 1:16
Q_Row{end+1} = vc(i,1);
end
for i = 1:16
Q_Row{end+1} = hc(i,2);
end
for i = 1:16
Q_Row{end+1} = vc(i,2);
end
%Q_Row{end+1} = 0;
%Creating array with Column Names for Excel Sheet
CNames = {'file_name'};
for i = 1:16
CNames{end+1} = sprintf('H-1-%d,%d', i-1, i-1);
end
for i = 1:16
CNames{end+1} = sprintf('V-1-%d,%d', i-1, i-1);
end
for i = 1:16
CNames{end+1} = sprintf('H-2-%d,%d', i-1, i-1);
end
```

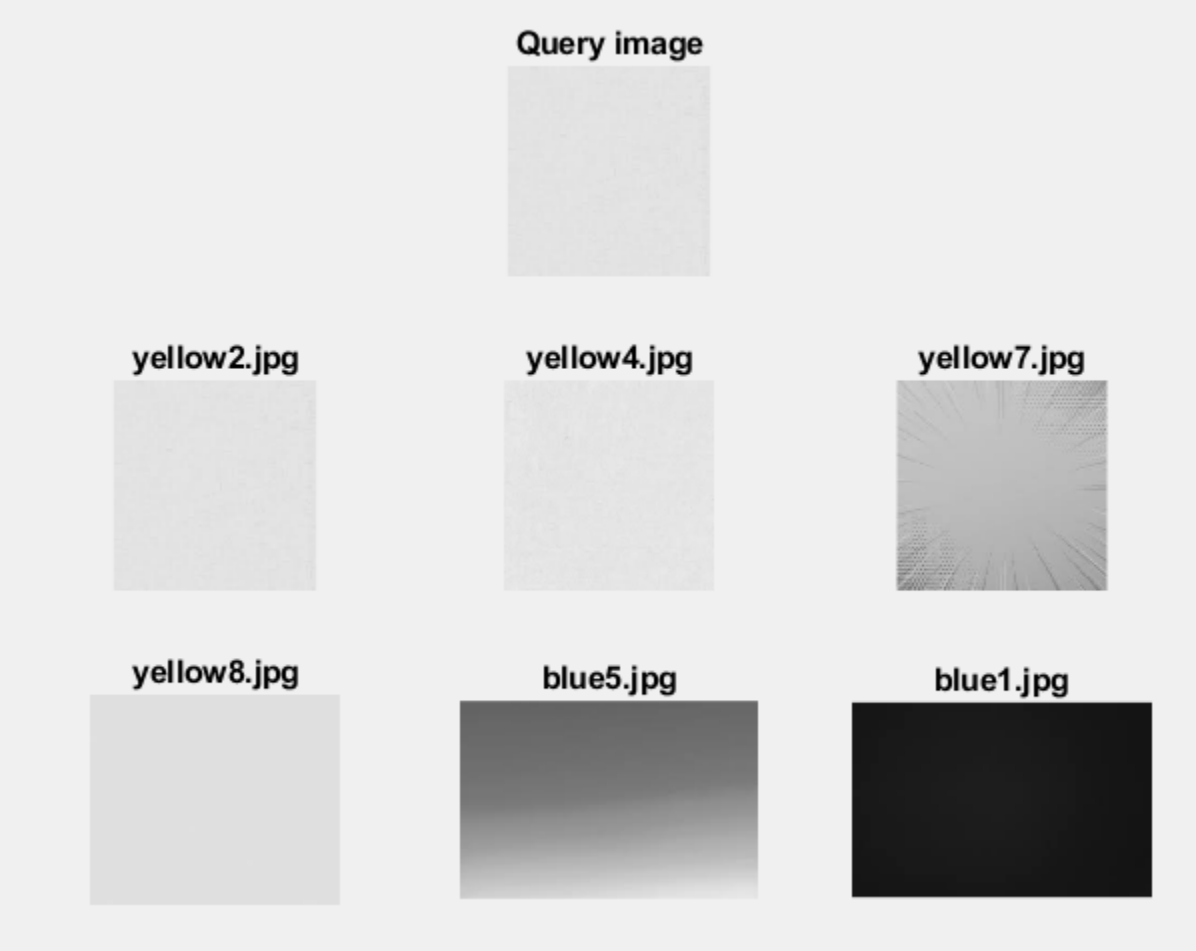
```
for i = 1:16
CNames{end+1} = sprintf('V-2-%d,%d', i-1, i-1);
end
CNames{end+1} = 'Chi-Square Distance';
info_table = cell2table(cell(0, 66), 'VariableNames',CNames);
%info_table=[info_table;Q_Row];
% Looping through all the images in the directory
for k = 1:numel(S)
F = fullfile(D,S(k).name);
I = imread(F);
I = rgb2gray(I);
I = imquantize(I,16);
S(k).data = I; % optional, save data.
I_Row={S(k).name};
[ihc ivc] = acg(I,[1 3]);
for i = 1:16
     I_Row{end+1} = ihc(i,1);
end
for i = 1:16
     I_Row{end+1} = ivc(i,1);
end
for i = 1:16
     I_Row{end+1} = ihc(i,2);
end
for i = 1:16
     I_Row{end+1} = ivc(i,2);
end
```

```
      tot_sum =0;
      for i = 2:65
      num = (Q_Row{i} - I_Row{i})^2;
      denum = Q_Row{i} + I_Row{i};
      if(denum==0)
            csd = 0;
      else
            csd = num/denum;
      end
      tot_sum = tot_sum + csd;
      %disp(I_Row(1)+"-"+Q_Row{i}+"diff. sq."+I_Row{i}+"->"+tot_sum);
      end
      tot_sum = tot_sum * 0.5;
      I_Row{end+1} = tot_sum;
      info_table = [info_table;I_Row];
end


% Replacing the NaN with values in the previous cell and replacing
the
% rows in the table in the ascending order of city block distance
info_table = sortrows(fillmissing(info_table, 'previous'),
'Chi-Square Distance');
writetable(info_table, 'lab5_1.xlsx','Sheet',3);


% Displaying the first 5 nearest image
subplot(3, 3, 2);
imshow(querry_image);
title('Query image');
```

```
% Extracting the filenames of the images

file_names = info_table(:, 'file_name').file_name;

for i = 1:6

F = fullfile(D,char(file_names(i)));

I = imread(F);

I = rgb2gray(I);

subplot(3, 3, i+3);

imshow(I);

title(char(file_names(i)));

end


toc;

function [horizontal_count, vertical_count] = acg(img, distances,
levels)

% Check if levels provided or not

if nargin == 2

levels = 16;

end


[Y, X] = size(img);

% Image quantization

% img = gray2ind(img, levels);

% Set variable sizes

[~, num_of_distances] = size(distances);

horizontal_count = zeros(levels, num_of_distances);

vertical_count = zeros(levels, num_of_distances);

% For each row

for r = 1:Y
```

```
% For each column
     for c = 1:X
% For each distance
          for d = 1:num_of_distances
               D = distances(d);
               value = img(r,c); % Get the value
               % Increment the resp. counter, if pixels equivalent
               if(r + D <= Y && img(r + D, c) == value)
               horizontal_count(value+1, d) =
horizontal_count(value+1, d) + 1;
               end
               if(c + D <= X && img(r, c + D) == value)
               vertical_count(value+1, d) = vertical_count(value+1,
d) + 1;
               end
          end
     end
end
end
```

OUTPUT :



Q2. C. Quantize the Image using *imquantize( )* to 32 gray levels

CODE : Input - blue4.jpg

```
tic;

D = './images';

S = dir(fullfile(D,'*.jpg')); % pattern to match filenames.


% Loading query image and converting to gray scale

query_image = imread('images/blue4.jpg');

querry_image = rgb2gray(query_image);
```

```
query_image = imquantize(querry_image,16);
Q_Row={'Query'};
%Extracting Horizontal and Vertical count of Query Image
[hc vc] = acg(query_image,[1 3]);
for i = 1:32
Q_Row{end+1} = hc(i,1);
end
for i = 1:32
Q_Row{end+1} = vc(i,1);
end
for i = 1:32
Q_Row{end+1} = hc(i,2);
end
for i = 1:32
Q_Row{end+1} = vc(i,2);
end
%Q_Row{end+1} = 0;
%Creating array with Column Names for Excel Sheet
CNames = {'file_name'};
for i = 1:32
CNames{end+1} = sprintf('H-1-%d,%d', i-1, i-1);
end
for i = 1:32
CNames{end+1} = sprintf('V-1-%d,%d', i-1, i-1);
end
for i = 1:32
CNames{end+1} = sprintf('H-2-%d,%d', i-1, i-1);
```

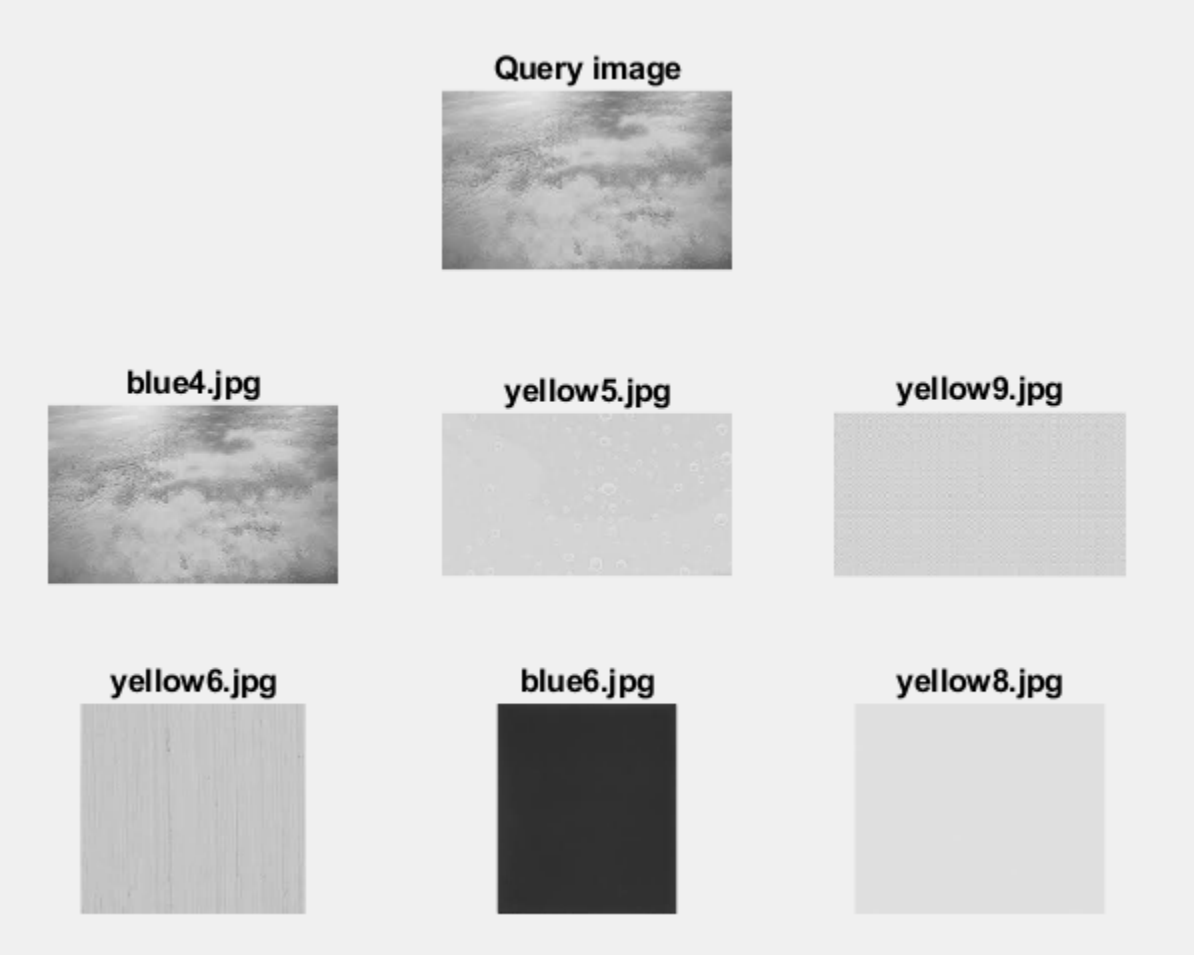
```
end
for i = 1:32
CNames{end+1} = sprintf('V-2-%d,%d', i-1, i-1);
end
CNames{end+1} = 'Chi-Square Distance';
info_table = cell2table(cell(0, 130), 'VariableNames',CNames);
%info_table=[info_table;Q_Row];
% Looping through all the images in the directory
for k = 1:numel(S)
F = fullfile(D,S(k).name);
I = imread(F);
I = rgb2gray(I);
I = imquantize(I,32);
S(k).data = I; % optional, save data.
I_Row={S(k).name};
[ihc ivc] = acg(I,[1 3]);
for i = 1:32
     I_Row{end+1} = ihc(i,1);
end
for i = 1:32
     I_Row{end+1} = ivc(i,1);
end
for i = 1:32
     I_Row{end+1} = ihc(i,2);
end
for i = 1:32
     I_Row{end+1} = ivc(i,2);
```

```
end
      tot_sum =0;
      for i = 2:129
      num = (Q_Row{i} - I_Row{i})^2;
      denum = Q_Row{i} + I_Row{i};
      if(denum==0)
            csd = 0;
      else
            csd = num/denum;
      end
      tot_sum = tot_sum + csd;
      %disp(I_Row(1)+"-"+Q_Row{i}+"diff. sq."+I_Row{i}+"->"+tot_sum);
      end
      tot_sum = tot_sum * 0.5;
      I_Row{end+1} = tot_sum;
      info_table = [info_table;I_Row];
end


% Replacing the NaN with values in the previous cell and replacing
the
% rows in the table in the ascending order of city block distance
info_table = sortrows(fillmissing(info_table, 'previous'),
'Chi-Square Distance');
writetable(info_table, 'lab5_1.xlsx','Sheet',4);


% Displaying the first 5 nearest image
subplot(3, 3, 2);
imshow(querry_image);
```

```
title('Query image');

% Extracting the filenames of the images

file_names = info_table(:, 'file_name').file_name;

for i = 1:6

F = fullfile(D,char(file_names(i)));

I = imread(F);

I = rgb2gray(I);

subplot(3, 3, i+3);

imshow(I);

title(char(file_names(i)));

end


toc;

function [horizontal_count, vertical_count] = acg(img, distances,
levels)

% Check if levels provided or not

if nargin == 2

levels = 32;

end


[Y, X] = size(img);

% Image quantization

% img = gray2ind(img, levels);

% Set variable sizes

[~, num_of_distances] = size(distances);

horizontal_count = zeros(levels, num_of_distances);

vertical_count = zeros(levels, num_of_distances);

% For each row
```

```
for r = 1:Y
% For each column
     for c = 1:X
% For each distance
          for d = 1:num_of_distances
               D = distances(d);
               value = img(r,c); % Get the value
               % Increment the resp. counter, if pixels equivalent
               if(r + D <= Y && img(r + D, c) == value)
               horizontal_count(value+1, d) =
horizontal_count(value+1, d) + 1;
               end
               if(c + D <= X && img(r, c + D) == value)
               vertical_count(value+1, d) = vertical_count(value+1,
d) + 1;
               end
          end
     end
end
End
```

OUTPUT :



Q2. D. Quantize the Image using *imquantize( )* **to 64 gray levels**

CODE : Input - blue4.jpg

```
tic;

D = './images';

S = dir(fullfile(D,'*.jpg')); % pattern to match filenames.


% Loading query image and converting to gray scale

query_image = imread('images/blue4.jpg');
```

```
querry_image = rgb2gray(query_image);
query_image = imquantize(querry_image,16);
Q_Row={'Query'};
%Extracting Horizontal and Vertical count of Query Image
[hc vc] = acg(query_image,[1 3]);
for i = 1:64
Q_Row{end+1} = hc(i,1);
end
for i = 1:64
Q_Row{end+1} = vc(i,1);
end
for i = 1:64
Q_Row{end+1} = hc(i,2);
end
for i = 1:64
Q_Row{end+1} = vc(i,2);
end
%Q_Row{end+1} = 0;
%Creating array with Column Names for Excel Sheet
CNames = {'file_name'};
for i = 1:64
CNames{end+1} = sprintf('H-1-%d,%d', i-1, i-1);
end
for i = 1:64
CNames{end+1} = sprintf('V-1-%d,%d', i-1, i-1);
end
for i = 1:64
```

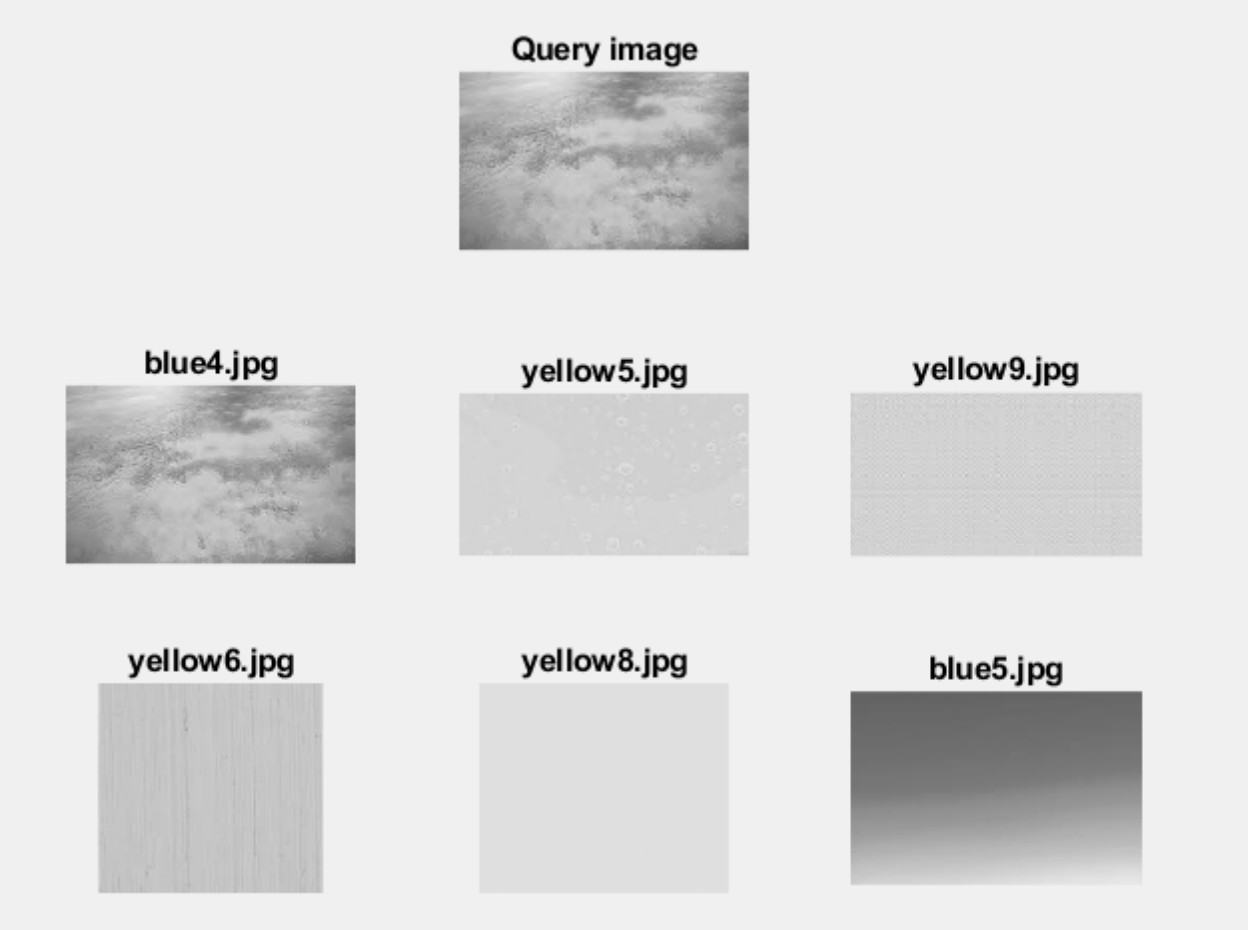
```
CNames{end+1} = sprintf('H-2-%d,%d', i-1, i-1);
end
for i = 1:64
CNames{end+1} = sprintf('V-2-%d,%d', i-1, i-1);
end
CNames{end+1} = 'Chi-Square Distance';
info_table = cell2table(cell(0, 258), 'VariableNames',CNames);
%info_table=[info_table;Q_Row];
% Looping through all the images in the directory
for k = 1:numel(S)
F = fullfile(D,S(k).name);
I = imread(F);
I = rgb2gray(I);
I = imquantize(I,64);
S(k).data = I; % optional, save data.
I_Row={S(k).name};
[ihc ivc] = acg(I,[1 3]);
for i = 1:64
     I_Row{end+1} = ihc(i,1);
end
for i = 1:64
     I_Row{end+1} = ivc(i,1);
end
for i = 1:64
     I_Row{end+1} = ihc(i,2);
end
for i = 1:64
```

```
      I_Row{end+1} = ivc(i,2);
end
      tot_sum =0;
      for i = 2:257
      num = (Q_Row{i} - I_Row{i})^2;
      denum = Q_Row{i} + I_Row{i};
      if(denum==0)
            csd = 0;
      else
            csd = num/denum;
      end
      tot_sum = tot_sum + csd;
      %disp(I_Row(1)+"-"+Q_Row{i}+"diff. sq."+I_Row{i}+"->"+tot_sum);
      end
      tot_sum = tot_sum * 0.5;
      I_Row{end+1} = tot_sum;
      info_table = [info_table;I_Row];
end


% Replacing the NaN with values in the previous cell and replacing
the
% rows in the table in the ascending order of city block distance
info_table = sortrows(fillmissing(info_table, 'previous'),
'Chi-Square Distance');
writetable(info_table, 'lab5_1.xlsx','Sheet',5);


% Displaying the first 5 nearest image
subplot(3, 3, 2);
```

```
imshow(querry_image);

title('Query image');

% Extracting the filenames of the images

file_names = info_table(:, 'file_name').file_name;

for i = 1:6

F = fullfile(D,char(file_names(i)));

I = imread(F);

I = rgb2gray(I);

subplot(3, 3, i+3);

imshow(I);

title(char(file_names(i)));

end


toc;

function [horizontal_count, vertical_count] = acg(img, distances,
levels)

% Check if levels provided or not

if nargin == 2

levels = 64;

end


[Y, X] = size(img);

% Image quantization

% img = gray2ind(img, levels);

% Set variable sizes

[~, num_of_distances] = size(distances);

horizontal_count = zeros(levels, num_of_distances);

vertical_count = zeros(levels, num_of_distances);
```

```
% For each row
for r = 1:Y
% For each column
     for c = 1:X
% For each distance
          for d = 1:num_of_distances
               D = distances(d);
               value = img(r,c); % Get the value
               % Increment the resp. counter, if pixels equivalent
               if(r + D <= Y && img(r + D, c) == value)
               horizontal_count(value+1, d) =
horizontal_count(value+1, d) + 1;
               end
               if(c + D <= X && img(r, c + D) == value)
               vertical_count(value+1, d) = vertical_count(value+1,
d) + 1;
               end
          end
     end
end
End
```

OUTPUT :


Query image
blue4.jpg
yellow5.jpg
yellow9.jpg
yellow6.jpg
yellow8.jpg
blue5.jpg

Challenging Task

Do the same exercise on the original color image in RGB plane

CODE :

```
tic;
D = './images';
S = dir(fullfile(D,'*.jpg')); % pattern to match filenames.

% Loading query image and converting to gray scale
query_image = imread('images/blue4.jpg');

% Extracting colour planes of query image
q_red = single(query_image(:,:,1));
q_green = single(query_image(:,:,2));
q_blue = single(query_image(:,:,3));

Q_Row={'Query'};
%Extracting Horizontal and Vertical count of Red Plane
[hc vc] = acg(q_red,[1 3]);
for i = 1:256
Q_Row{end+1} = hc(i,1);
end
for i = 1:256
Q_Row{end+1} = vc(i,1);
end
for i = 1:256
```

```
Q_Row{end+1} = hc(i,2);
end
for i = 1:256
Q_Row{end+1} = vc(i,2);
end
%Extracting Horizontal and Vertical count of Green Plane
[hc vc] = acg(q_green,[1 3]);
for i = 1:256
Q_Row{end+1} = hc(i,1);
end
for i = 1:256
Q_Row{end+1} = vc(i,1);
end
for i = 1:256
Q_Row{end+1} = hc(i,2);
end
for i = 1:256
Q_Row{end+1} = vc(i,2);
end
%Extracting Horizontal and Vertical count of Blue Plane
[hc vc] = acg(q_green,[1 3]);
for i = 1:256
Q_Row{end+1} = hc(i,1);
end
for i = 1:256
Q_Row{end+1} = vc(i,1);
end
```

```
for i = 1:256
Q_Row{end+1} = hc(i,2);
end
for i = 1:256
Q_Row{end+1} = vc(i,2);
end
%Creating array with Column Names for Excel Sheet
CNames = {'file_name'};
for i = 1:256
CNames{end+1} = sprintf('Red-H-1-%d,%d', i-1, i-1);
end
for i = 1:256
CNames{end+1} = sprintf('Red-V-1-%d,%d', i-1, i-1);
end
for i = 1:256
CNames{end+1} = sprintf('Red-H-2-%d,%d', i-1, i-1);
end
for i = 1:256
CNames{end+1} = sprintf('Red-V-2-%d,%d', i-1, i-1);
end
for i = 1:256
CNames{end+1} = sprintf('Green-H-1-%d,%d', i-1, i-1);
end
for i = 1:256
CNames{end+1} = sprintf('Green-V-1-%d,%d', i-1, i-1);
end
for i = 1:256
```

```
CNames{end+1} = sprintf('Green-H-2-%d,%d', i-1, i-1);
end
for i = 1:256
CNames{end+1} = sprintf('Green-V-2-%d,%d', i-1, i-1);
end
for i = 1:256
CNames{end+1} = sprintf('Blue-H-1-%d,%d', i-1, i-1);
end
for i = 1:256
CNames{end+1} = sprintf('Blue-V-1-%d,%d', i-1, i-1);
end
for i = 1:256
CNames{end+1} = sprintf('Blue-H-2-%d,%d', i-1, i-1);
end
for i = 1:256
CNames{end+1} = sprintf('Blue-V-2-%d,%d', i-1, i-1);
end
CNames{end+1} = 'Chi-Square Distance';
info_table = cell2table(cell(0, 3074), 'VariableNames',CNames);
%info_table=[info_table;Q_Row];
% Looping through all the images in the directory
for k = 1:numel(S)
F = fullfile(D,S(k).name);
I = imread(F);

% Extracting the colour plane of the current image
red = single(I(:, : , 1));
```

```
green = single(I(:, :, 2));
blue = single(I(:, :, 3));

S(k).data = I; % optional, save data.
I_Row={S(k).name};

[ihc ivc] = acg(red,[1 3]);
for i = 1:256
     I_Row{end+1} = ihc(i,1);
end
for i = 1:256
     I_Row{end+1} = ivc(i,1);
end
for i = 1:256
     I_Row{end+1} = ihc(i,2);
end
for i = 1:256
     I_Row{end+1} = ivc(i,2);
end

[ihc ivc] = acg(green,[1 3]);
for i = 1:256
     I_Row{end+1} = ihc(i,1);
end
for i = 1:256
     I_Row{end+1} = ivc(i,1);
end
```

```
for i = 1:256
     I_Row{end+1} = ihc(i,2);
end
for i = 1:256
     I_Row{end+1} = ivc(i,2);
end

[ihc ivc] = acg(blue,[1 3]);
for i = 1:256
     I_Row{end+1} = ihc(i,1);
end
for i = 1:256
     I_Row{end+1} = ivc(i,1);
end
for i = 1:256
     I_Row{end+1} = ihc(i,2);
end
for i = 1:256
     I_Row{end+1} = ivc(i,2);
end

     tot_sum =0;
     for i = 2:3073
     num = (Q_Row{i} - I_Row{i})^2;
     denum = Q_Row{i} + I_Row{i};
     if(denum==0)
          csd = 0;
```

```
      else
            csd = num/denum;
      end
      tot_sum = tot_sum + csd;
      %disp(I_Row(1)+"-"+Q_Row{i}+"diff. sq."+I_Row{i}+"->"+tot_sum);
      end
      tot_sum = tot_sum * 0.5;
      I_Row{end+1} = tot_sum;
      info_table = [info_table;I_Row];
end


% Replacing the NaN with values in the previous cell and replacing
the
% rows in the table in the ascending order of city block distance
info_table = sortrows(fillmissing(info_table, 'previous'),
'Chi-Square Distance');
writetable(info_table, 'lab5_1.xlsx','Sheet',6);


% Displaying the first 5 nearest image
subplot(3, 3, 2);
imshow(query_image);
title('Query image');
% Extracting the filenames of the images
file_names = info_table(:, 'file_name').file_name;
for i = 1:6
F = fullfile(D,char(file_names(i)));
I = imread(F);
subplot(3, 3, i+3);
```

```
imshow(I);

title(char(file_names(i)));

end


toc;

function [horizontal_count, vertical_count] = acg(img, distances,
levels)

% Check if levels provided or not

if nargin == 2

levels = 256;

end


[Y, X] = size(img);

% Image quantization

img = imquantize(img, levels);

% Set variable sizes

[~, num_of_distances] = size(distances);

horizontal_count = zeros(levels, num_of_distances);

vertical_count = zeros(levels, num_of_distances);

% For each row

for r = 1:Y

% For each column

     for c = 1:X

% For each distance

          for d = 1:num_of_distances

               D = distances(d);

               value = img(r,c); % Get the value

               % Increment the resp. counter, if pixels equivalent
```

```
                if(r + D <= Y && img(r + D, c) == value)
                horizontal_count(value+1, d) =
horizontal_count(value+1, d) + 1;
                end
                if(c + D <= X && img(r, c + D) == value)
                vertical_count(value+1, d) = vertical_count(value+1,
d) + 1;
                end
            end
      end
end
End
```

OUTPUT :