LAB Manual

PART A

(PART A : TO BE REFFERED BY STUDENTS)

**Experiment No.02**

**A.1 Aim:**

**To write a menu driven program in Matlab to compare two images.**

**A.2 Prerequisite:**

1. Understanding of fundamental programming functions/commands and environment of Matlab (Refer the Matlab manual),

2. Understanding of Switch case statement in Matlab.

2. Availability of Soft copy of your Photograph for experiment.

3. Understanding of correlation among pixel intensities and concept of distance calculation.

**A.3 Outcome:**

**After successful completion of this experiment students will be able to**

1. Explore and understand the importance of Correlation among neighborhood pixel intensities.
2. Understand various distance-measuring equations and compare them w.r.t. time complexity.
3. Write program to differentiate between two images based on following intensity distance calculation methods:
4. Minkowaski Distance
5. Euclidean distance
6. City block distance
7. Chess board distance

**A.4 Theory:**

**A.4.1 Relationship among pixels:**

Each pixel in the digital image has correlation with its neighboring pixel with respect to the intensity values they carry. This relationship focuses on the connectivity of the pixel with another pixel in the image. The connectivity among pixels can be determined in 3 ways : 4 connectivity, 8 connectivity and m connectivity. Lesser the difference among the pixel intensities w.r.t. to the neigbouring pixel can provide better quality of the image as a whole and this can be said to have the strong correlation among them.

There are various distance measuring methods (given in section A.4.2 below) which can be utilized to check the correlation among pixels. The distances calculated by these methods can be used to decide the level of correlation among pixels. Lesser the distance calculated that mean there is better correlation exists among the pixels and vice versa.

**A.4.2. Various distance measures:**

1. **Minkowaski Distance:**

The general formulae of Minkowaski distance is given below:

…..Equation (1)

1. **Euclidian Distance:**

The Eucledian distance formula can be obtained if the value of p =2 taken in the Minkowaski equation given in the equation (1).

1. **City Block Distance:**

The city block distance of two pixels D4 [(i,j), (k,l) ] = |i-k| +|j-l|. This metric measures the path between the pixels based on 4 connected neighbourhood as given in Figure 1 below:

|  |  |  |
| --- | --- | --- |
|  | O |  |
| O | x,y | O |
|  | O |  |

**Figure 1: Four connected neighbors**

For a pixel ‘p’ with the coordinates (x,y), the set of pixels given by

**N4 (p) = { (x+1,y), (x-1,y), (x,y+1), (x, y-1)}** ….. Equation (2).

**|x1-x2| + |y1-y2|**

1. **Chess Borad Distance:**

The chess board distance of two pixels is given by

**D8 [ (i, j), (k, l )] = max [ |i, k|, | j - l|]**  ….. Equation (3).

**max(| x1-x2| , | y1-y2 |)**

|  |  |  |
| --- | --- | --- |
| O | O | O |
| O | i ,j | O |
| O | O | O |

**Comparison between 2 images:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | **1** | **2** | **3** | **2** | | **0** | **2** | **3** | **1** | | **2** | **0** | **2** | **1** | | |  |  |  |  | | --- | --- | --- | --- | | **1** | **1** | **2** | **2** | | **0** | **1** | **2** | **2** | | **1** | **0** | **3** | **2** | |

**Image 1**

**Each pixel of Image1 is compared with each pixel of image 2**

**A.5 Procedure/Algorithm:**

**TASK 1:**

1. Create a new matlab file.
2. Read two input images.
3. Select the user choice of distance measure to be used.
4. Ask for comparison whether to be done between same images or two different images
5. Compare both images pixel by pixel using distance measuring method chosen.
6. Comment on the time taken by the program to complete the task and comment whether both input images are same or different.
7. Complete PART B of lab manual.
8. Save and close the file and name it as **EXP2\_Task1\_your Roll no.m**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

PART B

(PART B : TO BE COMPLETED BY STUDENTS)

***(Students must submit the soft copy as per following segments within two hours of the practical. The soft copy must be uploaded on the Blackboard or emailed to the concerned lab in charge faculties at the end of the practical in case there is no Black board access available)***

|  |  |
| --- | --- |
| Roll No. | Name: |
| Class : | Batch : |
| Date of Experiment: | Date of Submission |
| Grade : |  |

**B.1 Software Code written by student:**

clc;

clear all;

img = imread("gg.jpg");

imgG = rgb2gray(img);

A = imresize(img1G, [256, 256]);

Img2 = imread("ff.jpg");

img2G = rgb2gray(img2);

B = imresize(img2G, [256, 256]);

fprintf("\tWithout loop function\n")

fprintf("\t\tEuclidean\n");

tic;

ed1 = nthroot(sum(abs((double(A) - double(B))).^2, "all"), 2);

toc;

display(ed1);

fprintf("\t\tCity Block\n");

cb1 = sum(abs(double(A) - double(B)), "all");

toc;

display(cb1);

fprintf("\t\tChess Board\n");

tic;

cbd1 = max(abs( double(A) - double(B) ), [], "all");

toc;

display(cbd1);

fprintf("\t\tMinkowski\n");

p=3;

tic;

mkd1 = nthroot(sum(abs((double(A) - double(B))).^p, "all"), p);

toc;

display(mkd1);

fprintf("\n\n\tWith loop function\n")

fprintf("\t\tEuclidean\n");

tic;

ed = eucl(A, B);

toc;

display(ed);

fprintf("\t\tCity Block\n");

tic;

cb = cblock(A, B);

toc;

display(cb);

fprintf("\t\tChess Board\n");

tic;

cbd = cboard(A, B);

toc;

display(cbd);

fprintf("\t\tMinkowski\n");

tic;

mkd = mink(A, B, 3);

toc;

display(mkd);

function euclideanDistance = eucl(A, B)

e = 0;

for i = 1:256

for j = 1:256

e = e + abs( double(A(i,j)) - double(B(i,j)) ).^2;

end

end

euclideanDistance = sqrt(e);

end

function cityBlockDistance = cblock(A, B)

cityBlockDistance = 0;

for i = 1:256

for j = 1:256

cityBlockDistance = cityBlockDistance + abs( double(A(i,j)) - double(B(i,j)) );

end

end

end

function chessBoardDistance = cboard(A, B)

chessBoardDistance = 0;

for i = 1:256

for j = 1:256

k = abs( double(A(i,j)) - double(B(i,j)) );

if k > chessBoardDistance

chessBoardDistance = k;

end

end

end

end

function minkowskiDistance = mink(A, B, p)

mk = 0;

for i = 1:256

for j = 1:256

mk = mk + abs(double(A(i,j)) - double(B(i,j)) ).^p;

end

end

minkowskiDistance = nthroot(mk, p);

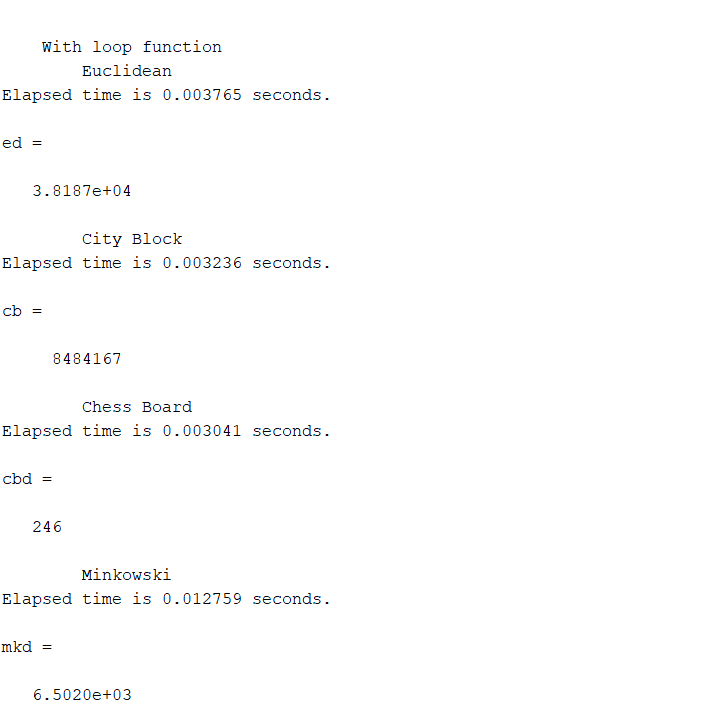
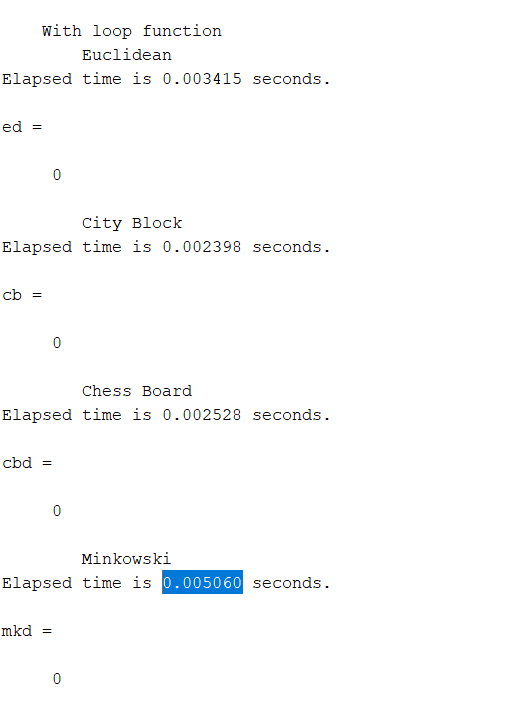
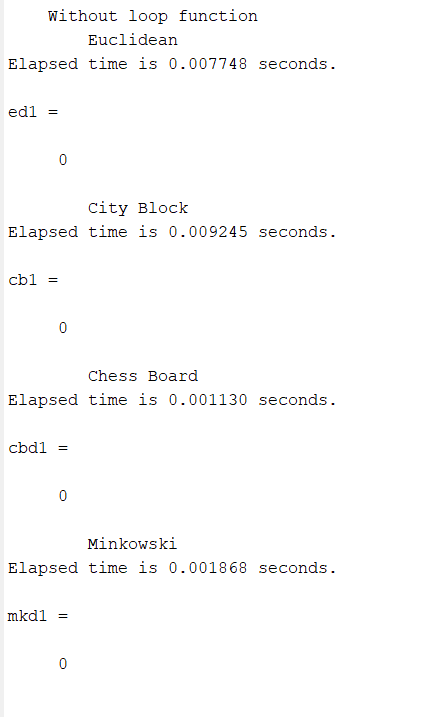
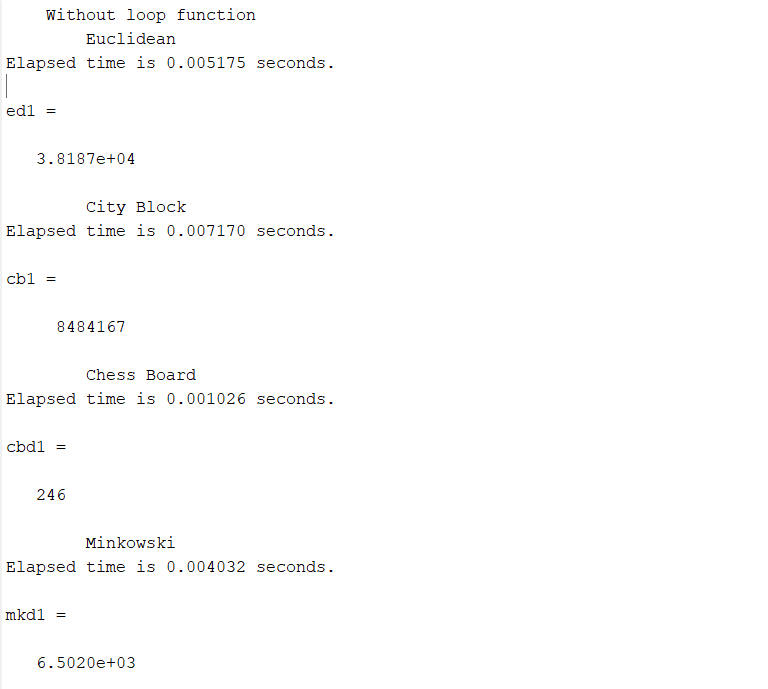
end

**B.2 Input and Output:**

**B.2.2 TASK 1**

**Input Images:**

**** 

**Output: ** 

1. Choice of distance measure used
2. Type (Similar & Dissimilar) of images used for comparison.
3. Output for each choices w.r.t. to comparison of same image and two different images
4. Time taken to compare images for all distance measures.

**B.3 Observations and learning:**

**Without Loop**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **For Same Image** | **MD** | | **ED** | | **City Block** | | **Chess Board** | |
|  | **Dist.** | **Time(s)** | **Dist.** | **Time(s)** | **Dist.** | **Time(s)** | **Dist.** | **Time(s)** |
|  | 0 | 0.001868 | 0 | 0.007748 | 0 | 0.009245 | 0 | 0.001130 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **For Different Images** | **MD** | | **ED** | | **City Block** | | **Chess Board** | | |
|  | **Dist.** | **Time** | **Dist.** | **Time** | **Dist.** | **Time(s)** | **Dist.** | **Time(s)** |
|  | 6.5020e+03 | 0.004032 | 3.8187e+04 | 0.003815 | 8484167 | 0.007170 | 246 | 0.001026 |

**With Loop:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **For Same Image** | **MD** | | **ED** | | **City Block** | | **Chess Board** | |
|  | **Dist.** | **Time(s)** | **Dist.** | **Time(s)** | **Dist.** | **Time(s)** | **Dist.** | **Time(s)** | |
|  | 0 | 0.005060 | 0 | 0.003415 | 0 | 0.002398 | 0 | 0.002528 | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **For Different Images** | **MD** | | **ED** | | **City Block** | | **Chess Board** | |
|  | **Dist.** | **Time** | **Dist.** | **Time** | **Dist.** | **Time(s)** | **Dist.** | **Time(s)** | |
|  | 6.5020e+03 | 0.012759 | 3.8187e+04 | 0.003765 | 8484167 | 0.003236 | 246 | 0.003041 | |

**We can clearly observe that the loop increases the time complexity of our program**

**This test case is not enough to test which distance algo is fastest as there is conflicting data**

**The purpose of these algos is generally to measure the similar-ness of 2 images, therefor they are more useful in their matrix format rather than condensed into a single number.**

**B.4 Conclusion:**

* Learned to implement various distance algorithms and use them to find pixel distance between two images of same matrix size.
* Understood the time complexity of the algorithms.
* Also, understood the importance of relationship between pixels

**B.5 Question of Curiosity**

***(To be answered by student based on the practical performed and learning/observations)***

Why do you feel the relationship among all pixels in any digital image is essential to study and ponder over? How it affects the overall quality of an image? Give an working example.

****

This image is a gif illustrating the importance of distance measuring in an image.

In general the relationship between pixels of an image is important in image quality as the brightness, saturation, colour profile of an image is based on the values of the pixels and for a specific camera is determined, then we can manipulate the images in post processing. Thus raw files are most workable in the image processing industry as opposed to compressed images.