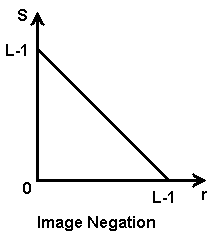
Aim: - To study image enhancement in spatial domain

Tools Used: - MatLab

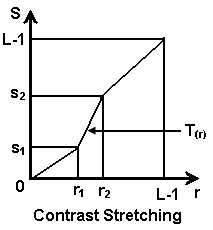
Theory: -

Image Negative: - The negative of an image with gray levels in the range [0, L-1] is obtained by using the negative transformation given by the expression, S= L – 1 – r (1).

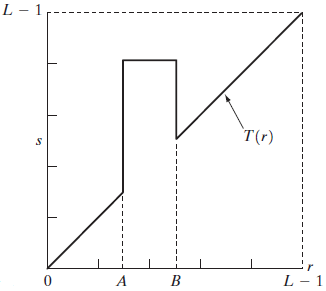
This is according to the transformation S = T (r). In above transformation (1), the intensity of the output image decreases as the intensity of the input increases. The type of processing is particularly suited for enhancing white or gray detail embedded in dark regions of an image especially when black areas are dominants in site.



Contrast Stretching: - Low contrast images can result from poor illumination, lack of dynamic range in the imaging sensor etc. The idea behind contrast stretching is to increase the dynamic range of the gray levels in the image being processed. The transformation function for contrast stretching is given by,



Grey-Level Slicing: - Highlighting a specific range of gray levels in an image often is desired. Applications include enhancing features such as masses of water in satellite imagery and enhancing flaws in X-ray images. There are several ways of doing level slicing, but most of them are variations of two basic themes. One approach is to display a high value for all gray levels in the range of interest and a low value for all other gray levels.



Log Transformation: -

The log transformation curve shown in fig. is given by the expression,

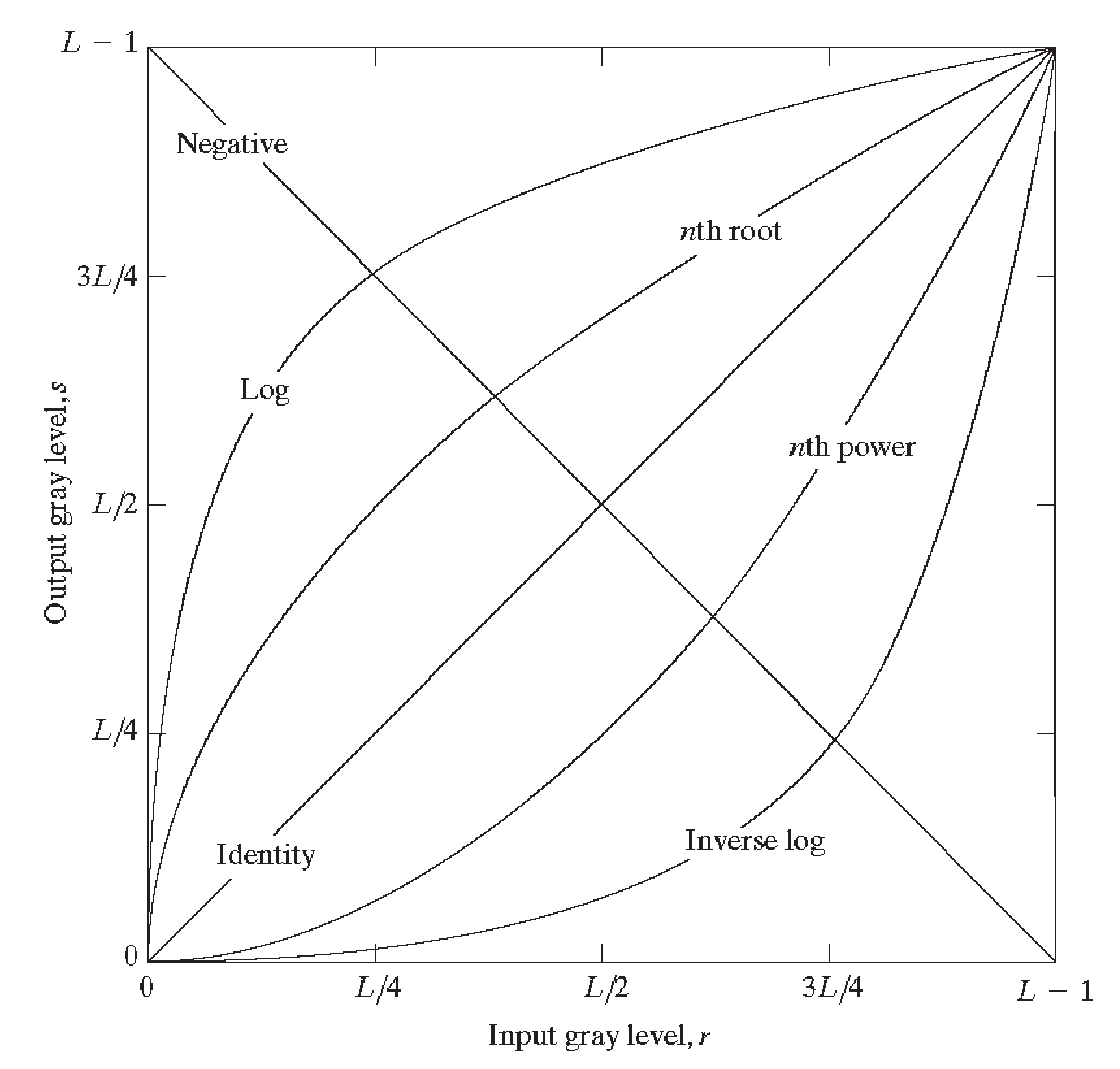
where c is a constant and it is assumed that r ≥ 0. The shape of the log curve in fig. tells that this transformation maps a narrow range of low-level grey scale intensities into a wider range of output values. And similarly maps the wide range of high-level grey scale intensities into a narrow range of high level output values. The opposite of this applies for inverse-log transform.

This transform is used to expand values of dark pixels and compress values of bright pixels.

Power-Law Transformation: -

The nth power and nth root curves shown in fig. can be given by the expression,

This transformation function is also called as gamma correction. For various values of  different levels of enhancements can be obtained. This technique is quite commonly called as Gamma Correction. If you notice, different display monitors display images at different intensities and clarity. That means, every monitor has built-in gamma correction in it with certain gamma ranges and so a good monitor automatically corrects all the images displayed on it for the best contrast to give user the best experience.



Algorithm: -

Image Negative: -

1. Read i/p image
2. Read maximum gray level pixel of i/p image
3. Replace input image by o/p = (maximum – i/p )
4. Display o/p image

Image Contrast Stretching: -

1. Read input image
2. Get values r1,r2,s1,s2
3. Calculate and  slopes.
4. If input pixel value is ≤ r1 then o/p =  \* input
5. If input pixel is > r1and ≤ r2 then o/p =  \* (r-r1)+s1
6. otherwise o/p =  \* (r-r2)+s2
7. Display i/p image
8. Display o/p image

Image Grey-Level Slicing: -

1. Read input image
2. Get values A and B
3. If input pixel is ≥ A and ≤ B then o/p = 
4. otherwise o/p = i/p
5. Display i/p image
6. Display o/p image

Log Transformations: -

1. Read input image
2. Convert all pixel values to variable type double
3. Get value for constant c
4. For each pixel calculate
5. Store calculated values to o/p matrix
6. Convert all pixel values of o/p matrix to variable type 8 bit unsigned integer
7. Display i/p image
8. Display o/p image

Power-Law Transformations: -

1. Read input image
2. Convert all pixel values to variable type double
3. Get value for 
4. For each pixel calculate
5. Store calculated values to o/p matrix
6. Convert all pixel values of o/p matrix to variable type 8 bit unsigned integer
7. Display i/p image
8. Display o/p image

Program:-

Experiment 2

%Aim: To study image enhancement in spatial domain using Matlab.

clc;

clear all;

close all;

A=imread('cameraman.tif');

[m,n]=size(A);

p=max(max(A))/2;

% Digital negative

D=p-A;

%Contrast Stretch Image

r1=input(' enter value of r1');

r2=input(' enter value of r2');

s1=input(' enter value of s1');

s2=input(' enter value of s2');

alpha=s1/r1;

beta=(s2-s1)/(r2-r1);

gamma=(p-s2)/(p-r2);

a=double(A);

for i=1:m

for j=1:n

if (a(i,j)<r1)

c(i,j)=alpha\*a(i,j);

elseif (a(i,j)>=r1&&a(i,j)<r2)

c(i,j)=beta\*(a(i,j)-r1)+s1;

else

c(i,j)=gamma\*(a(i,j)-r2)+s2;

end

end

end

%Clipping

beta=(s2-s1)/(r2-r1);

for i=1:m

for j=1:n

if (a(i,j)<r1)

clipp(i,j)=0;

elseif (a(i,j)>=r1&&a(i,j)<r2)

clipp(i,j)=beta\*(a(i,j)-r1)+s1;

else

clipp(i,j)=s2;

end

end

end

%Thresholding

for i=1:m

for j=1:n

if (a(i,j)<r1)

threshold(i,j)=0;

else

threshold(i,j)=255;

end

end

end

% log transformed image

org=double(A)/255;

const=input('c = ');

h=const.\*log(1+org);

logt=uint8(h\*255);

% power law transformed image

pgamma=input('gamma = ');

f1=const.\*(org.^pgamma);

powerlog=uint8(f1\*255);

% grey level slicing - without background

r1=input(' enter value of r1');

r2=input(' enter value of r2');

A1=double(A);

for i=1:m

for j=1:n

if (A1(i,j)>=r1&&A1(i,j)<=r2)

nobg(i,j)=255;

else

nobg(i,j)=0;

end

end

end

% grey level slicing - with background

for i=1:m

for j=1:n

if (A1(i,j)>=r1&&A1(i,j)<=r2)

withbg(i,j)=255;

else

withbg(i,j)=A1(i,j);

end

end

end

% 8 bit slicing

for i=1:m

for j=1:n

z=A1(i,j);

zb=dec2bin(z,8);

b8(i,j)=uint8(bin2dec(zb(:,1)));

b7(i,j)=uint8(bin2dec(zb(:,2)));

b6(i,j)=uint8(bin2dec(zb(:,3)));

b5(i,j)=uint8(bin2dec(zb(:,4)));

b4(i,j)=uint8(bin2dec(zb(:,5)));

b3(i,j)=uint8(bin2dec(zb(:,6)));

b2(i,j)=uint8(bin2dec(zb(:,7)));

b1(i,j)=uint8(bin2dec(zb(:,8)));

end

end

figure(1)

subplot(3,3,1); imshow(A); title('original image');

subplot(3,3,2); imshow(D); title('image negative');

subplot(3,3,3); imshow(uint8(c)); title('constrast stretched image');

subplot(3,3,4); imshow(uint8(clipp)); title('clipped image');

subplot(3,3,5); imshow(uint8(threshold)); title('threshold image');

subplot(3,3,6); imshow(logt); title('log transformed image');

subplot(3,3,7); imshow(powerlog); title('power law transformed image');

subplot(3,3,8); imshow(uint8(withbg)); title('Grey level with BG');

subplot(3,3,9); imshow(uint8(nobg)); title('Grey level without BG');

figure(2)

subplot(3,3,1); imshow(A); title('original image');

subplot(3,3,2); imshow(b8\*255); title('Bit plane 8(MSB)');

subplot(3,3,3); imshow(b7\*255); title('Bit plane 7');

subplot(3,3,4); imshow(b6\*255); title('Bit plane 6');

subplot(3,3,5); imshow(b5\*255); title('Bit plane 5');

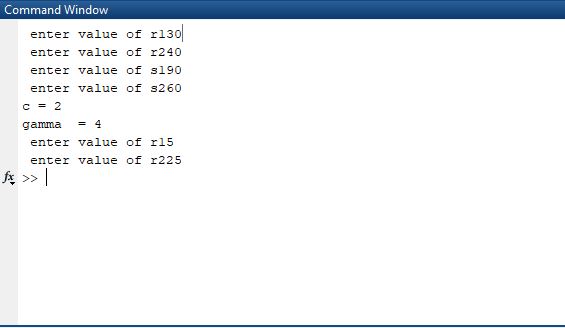
subplot(3,3,6); imshow(b4\*255); title('Bit plane 4');

subplot(3,3,7); imshow(b3\*255); title('Bit plane 3');

subplot(3,3,8); imshow(b2\*255); title('Bit plane 2');

subplot(3,3,9); imshow(b1\*255); title('Bit plane 1 (LSB)');

Command Window:



Output:



