EE 5356 Digital Image Processing

LAB Assignment 1

Zooming with Interpolation

1. Replication :
   1. Description – The image matrix is interlaced with zeros between each row and each column. Convolve the image matrix with zero order hold matrix. The resultant image matrix is double the size of the original image.
   2. Matlab program -

% file : Assig1\_1.m

% description : This program will magnify the original image

% 256 X 256 by two and give a 512 X 512 image using the

% replication method

% read the 256 X 256 image into a matrix

iImgSmall = imread('goldhill256.bmp');

iSizeSmall = length(iImgSmall);

iSizeLarge = 2 \* iSizeSmall

% create a zero matrix twice the size of the image matrix

dImgLarge = zeros(iSizeLarge);

% populate the new matrix with the values from the image matrix

% this is done to get the image matrix interlaced with zeros

for i = 1:1:iSizeSmall

for j = 1:1:iSizeSmall

dImgLarge(2\*i-1,2\*j-1) = iImgSmall(i,j);

end

end

H = ones(2);

% convolve the new matrix with one matrix of the 2nd order

% to get the magnified image matrix

iImgLarge = uint8(conv2(dImgLarge,H));

iImgLarge = iImgLarge(1:iSizeLarge,1:iSizeLarge);

figure(1);

imshow(iImgSmall);

figure(2);

imshow(iImgLarge);

* 1. Input – 
  2. Output –



1. Linear Interpolator :
   1. Description – The image matrix is interlaced with zeros between each row and each column. Convolve the image matrix with first order hold matrix. The resultant image matrix is double the size of the original image. It has better picture quality as compared to replication zooming.
   2. Matlab program –

% file : Assig1\_2.m

% description : This program will magnify the original image

% 256 X 256 by two and give a 512 X 512 image using the

% linear interpolation method

% read the 256 X 256 image into a matrix

iImageSmall = imread('goldhill256.bmp');

iSizeSmall = length(iImageSmall);

iSizeLarge = 2\*iSizeSmall;

% create a zero matrix twice the size of the image matrix to be

% used as image matrix for linear interpolation

dImageLinear = zeros(iSizeLarge);

% populate the FOH matrix with the values from the image matrix

% this is done to get the image matrix interlaced with zeros

for i = 1:1:iSizeSmall

for j = 1:1:iSizeSmall

dImageLinear(2\*i-1,2\*j-1) = iImageSmall(i,j);

end

end

H2 = [0.25 0.5 0.25;0.5 1 0.5;0.25 0.5 0.25];

% convolve the FOH matrix with one matrix of the 2nd order

% convolved with itself (H2 = H conv H ) normalized to one

% to get the magnified image matrix

iImageLinear = uint8(conv2(dImageLinear,H2));

iImageLinear = iImageLinear(1:iSizeLarge,1:iSizeLarge);

imshow(iImageSmall);

figure(2)

imshow(iImageLinear);

* 1. Input – 
  2. Output –



1. Cubic spline Interpolator:
   1. Description – The image matrix is interlaced with zeros between each row and each column. Convolve the image matrix with third order hold matrix. The resultant image matrix is double the size of the original image. It has better picture quality as compared to replication zooming and Linear Interpolation, it is known as cubic spline interpolation.
   2. Matlab program –

% file : Assig1\_3.m

% description : This program will magnify the original image

% 256 X 256 by two and give a 512 X 512 image using the

% cubic spline method

% read the 256 X 256 image into a matrix

iImageSmall = imread('goldhill256.bmp');

iSizeSmall = length(iImageSmall);

iSizeLarge = 2\*iSizeSmall;

% create a zero matrix twice the size of the image matrix

dImageLarge = zeros(iSizeLarge);

% populate the new matrix with the values from the image matrix

% this is done to get the image matrix interlaced with zeros

for i = 1:1:iSizeSmall

for j = 1:1:iSizeSmall

dImageLarge(2\*i-1 , 2\*j-1) = iImageSmall(i , j);

end

end

H = [1 1; 1 1];

H4 = conv2(H,conv2(H,conv2(H,H)));

H4 = H4 ./ H4(3,3);

% convolve the new matrix with one matrix of the 2nd order

% convolved with itself four times and normalized to 1 to get

% the magnified image matrix

iImageLarge = uint8(conv2(dImageLarge,H4));

iImageLarge = iImageLarge(1:iSizeLarge,1:iSizeLarge);

figure(1);

imshow(iImageSmall);

figure(2);

imshow(iImageLarge);

* 1. Input - 
  2. Output -

