**Image and Video Processing**

**Programming Assignment – Report**

**Week 4: Re-sampling and Warping**

Problem1

%ReSample.m

%Script to downsample and then upsample the image by factors of 2

f = double(imread('barbara.bmp'));

dim = size(f);

%Downsampling

for m = 1:dim(1)/2-1

for n = 1:dim(2)/2-1

% Downsamping without pre-filtering

fd1(m,n)=f(2\*m,2\*n);

% Downsampling with an averaging filter

fd2(m,n)=(f(2\*m,2\*n) + f(2\*m,2\*n +1) + f(2\*m+1,2\*n) + f(2\*m+1,2\*n+1))/4;

end

end

subplot(1,2,1), imshow(uint8(fd1)); subplot(1,2,2), imshow(uint8(fd2));

imwrite(uint8(fd1),'Down1.jpg');

imwrite(uint8(fd2),'Down2.jpg');

%Upsampling

outImg1 = image2Upsample(fd1,'bilinear');

outImg2 = image2Upsample(fd1,'bicubic');

outImg3 = image2Upsample(fd2,'bilinear');

outImg4 = image2Upsample(fd2,'bicubic');

imwrite(uint8(outImg1),'hw1.2.1.jpg');

imwrite(uint8(outImg2),'hw1.2.2.jpg');

imwrite(uint8(outImg3),'hw1.2.3.jpg');

imwrite(uint8(outImg4),'hw1.2.4.jpg');

function [ outImg ] = image2Upsample(img, order)

%imageUpsample function upsamples the input image by 2 using bilinear and cubic

%interpolation

dim = size(img);

outImg = zeros(2\*dim(1),2\*dim(2));

%Bilinear Interpolation

if strcmp(order,'bilinear')

for m = 1:dim(1)-1

for n = 1:dim(2)-1

outImg(2\*m,2\*n) = img(m,n);

outImg(2\*m,2\*n+1)= (img(m,n)+img(m,n+1))/2;

outImg(2\*m+1,2\*n)= (img(m,n)+img(m+1,n))/2;

outImg(2\*m+1,2\*n+1)= (img(m,n)+img(m,n+1)+img(m+1,n)+img(m+1,n+1))/2;

end

end

%Bicubic Interpolation

elseif strcmp(order,'bicubic')

for m = 2:dim(1)-2

for n = 2:dim(2)-2

outImg(2\*m,2\*n) = img(m,n);

outImg(2\*m,2\*n+1)= -(1/8)\*img(m,n-1)+(5/8)\*img(m,n)+(5/8)\*img(m,n+1)-(1/8)\*img(m,n+2);

end

end

for m = 2:dim(1)-2

for n = 2:dim(2)-2

outImg(2\*m,2\*n) = img(m,n);

outImg(2\*m+1,2\*n)= -(1/8)\*img(m-1,n)+(5/8)\*img(m,n)+(5/8)\*img(m+1,n)-(1/8)\*img(m+2,n);

end

end

else

disp('Function accepts only bilinear or bicubic for order');

return;

end

end

Images:



Downsampled:



Figure : Simple Downsampling



Figure : With Avg filter

Upsampled:

Case1 – Upsampled Figure1 Bilinear

Case2 – Upsampled Figure2 Bicubic

Case3 – Upsampled Figure3 Bilinear

Case4 – Upsampled Figure4 Bicubic



Figure Case1



Figure Case2



Figure Case3



Figure Case4

Problem 2

% imRegister.m

% Script to import two images, select common features from both and use

% least square solution

grayImg = imread('lena\_gray.bmp');

len = 100;

img1 = grayImg(:,1:end-len);

img2 = imread('tilt\_lena.jpg');

cpselect(img2, img1);

% Organizing the points into matrix to obtain Least Square Solution for the

% affine matrix

A = [[1;1;1;1] fixedPoints];

x = movingPoints(:,1);

y = movingPoints(:,2);

p = transpose(A);

a = inv(p\*A)\*p\*x;

b = inv(p\*A)\*p\*y;

% invA = inverse(A);

% affine transform between the two images;

tForm = [a(2) b(2) 0; a(3) b(3) 0; a(1) b(2) 1];

tA = tForm(1:2,1:2);

tB = [tForm(3,1); tForm(3,2)];

% apply affine transform to obtain mapped co-ordinates.

% (buggy)

index = 1;

for x = 1:dim(1)

for y = 1:dim(2)

X = [x;y];

U = inv(tA)\*(X-tB);

UI(index) = U(1);

VI(index) = U(2);

index = index+1;

end

end

%Interpolation to realize the image

[X,Y] = meshgrid(1:dim(1),1:dim(2));

outImg = interp2(X,Y,double(img1),UI,VI,'linear');

Images:

 