INTERPOLATION

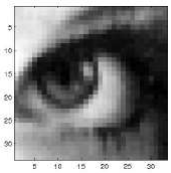
LABORATORIO AVANZADO DE PROCESAMIENTO DE IMÁGENES
FACULTAD DE INGENIERÍA

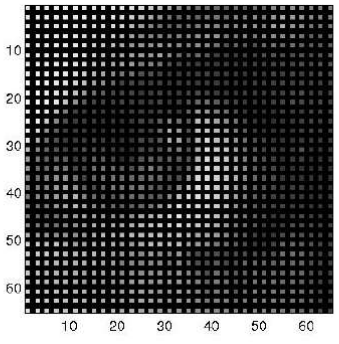
DRA. JIMENA OLVERES MONTIEL
DR. BORIS ESCALANTE RAMÍREZ

1

Image interpolation occurs in all digital images at some stage

- Resizing (resampling)
- Remapping (geometrical tansformations- rotation, change of perspective,...)
- Inpainting (restauration of *holes*)
- Morphing, nonlinear transformations

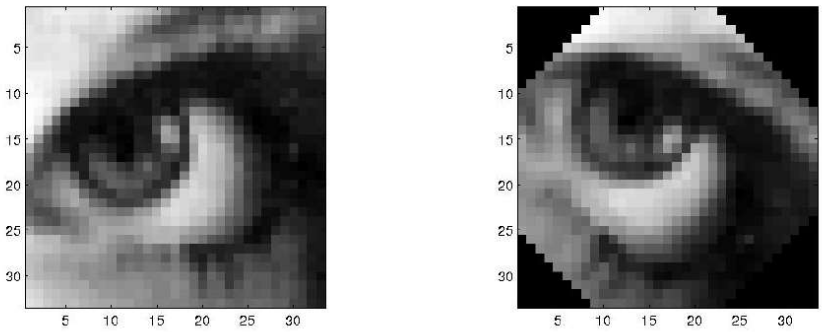




2

Image interpolation occurs in all digital images at some stage

- Resizing (resampling)
- Remapping (geometrical tansformations- rotation, change of perspective,...)
- Inpainting (restauration of *holes*)
- Morphing, nonlinear transformations



3

Image interpolation occurs in all digital images at some stage

- Resizing (resampling)
- Remapping (geometrical tansformations- rotation, change of perspective,...)
- Inpainting (restauration of *holes*)
- Morphing, nonlinear transformations

4

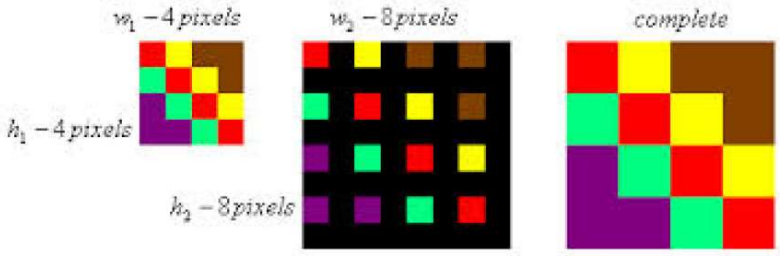
Image interpolation occurs in all digital images at some stage

- Resizing (resampling)
- Remapping (geometrical tansformations- rotation, change of perspective,...)
- Inpainting (restauration of *holes*)
- Morphing, nonlinear transformations

5

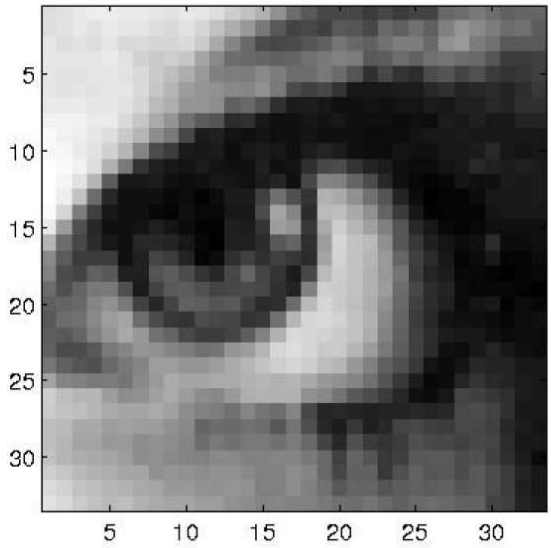
Nearest Neighbor

- Most basic method
- Requires the least processing time
- Only considers one pixel: the closest one to the interpolated point
- Has the effect of simply making each pixel bigger



6

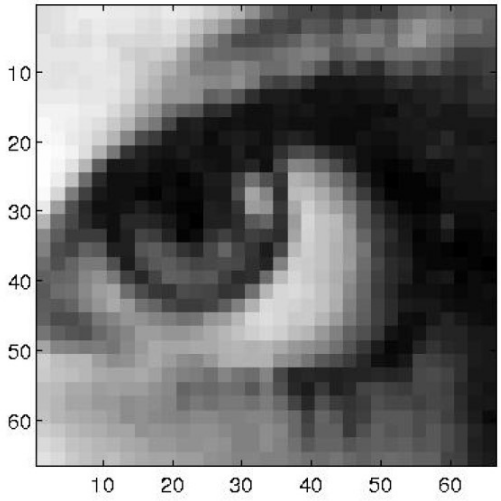
Nearest Neighbor



7

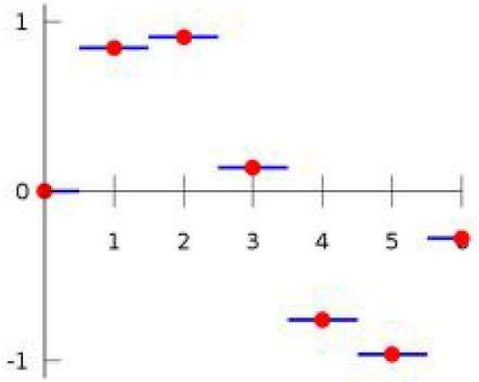
Nearest Neighbor

double pixels



8

Relationship with 1D interpolation



9

Bilinear

- Considers the closest 2x2 neighborhood of known pixel values surrounding the unknown pixels
- Takes a wheighted average of these 4 pixels to arrive at the final interpolated values
- Results in smoother looking images than nearest neighborhood
- Needs of more processing time

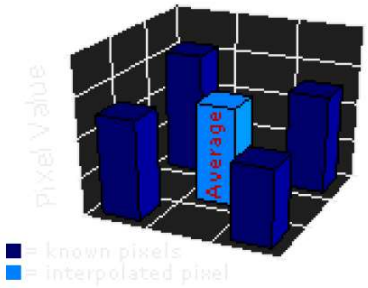
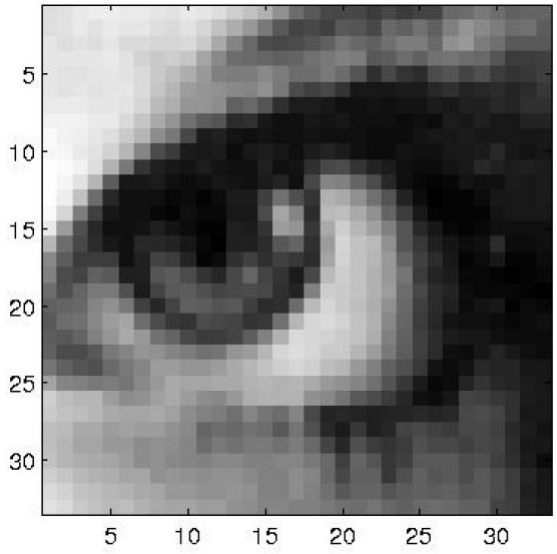


Figure: Case when all known pixel distances are equal. Interpolated value is simply their sum divided by four.

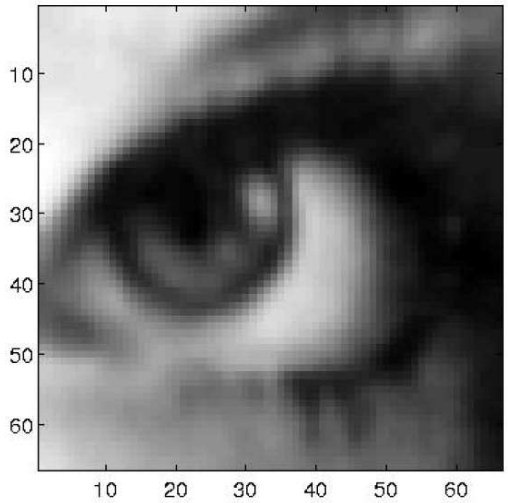
10

Bilinear



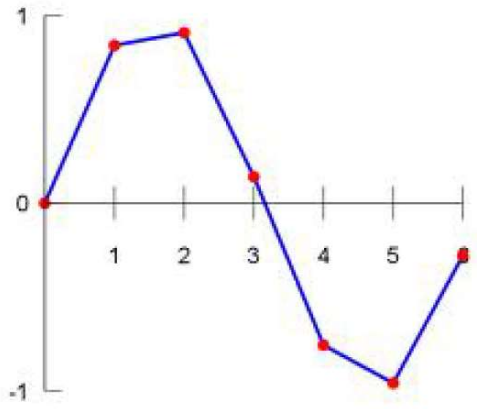
11

Bilinear



12

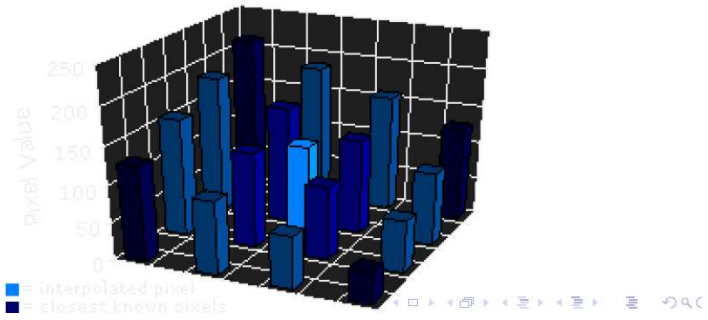
Relationship with 1D interpolation



13

Bicubic

- One step beyond bilinear by considering the closest 4x4 neighborhood of known pixels, for a total of 16 pixels
- Since these are at various distances from the unknown pixel, closer pixels are given a higher weighting in the calculation
- Produces sharper images than the previous two methods.
- Good compromise between processing time and output quality
- Standard in many image editing programs, printer drivers and in-camera interpolation



14

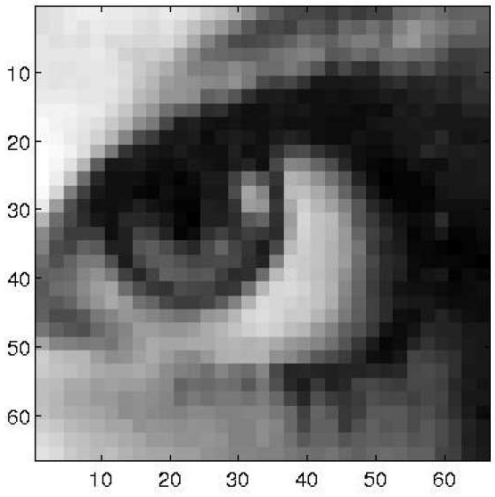


Figure: Nearest

15

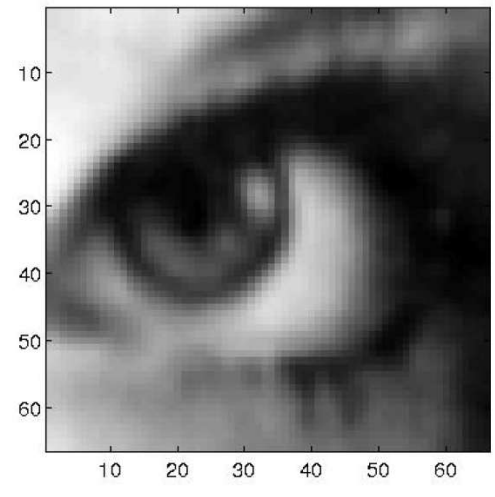


Figure: Bilinear

16

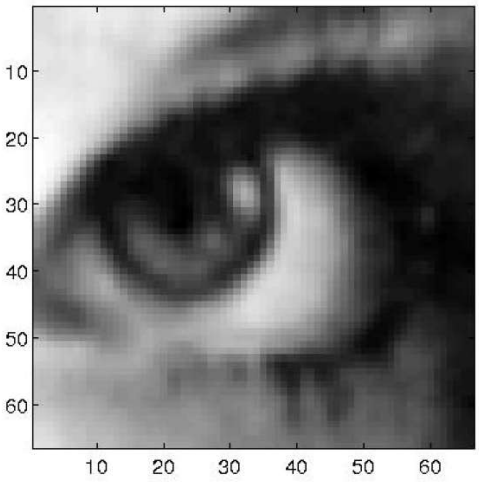
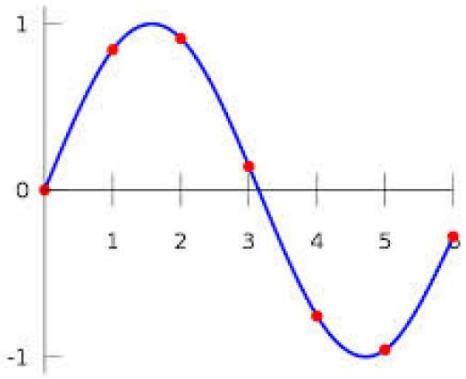


Figure: Bicubic

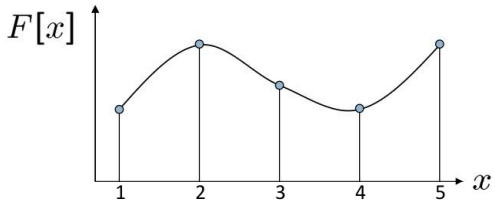
17

Relationship with 1D interpolation



18

Image interpolation



d = 1 in this example

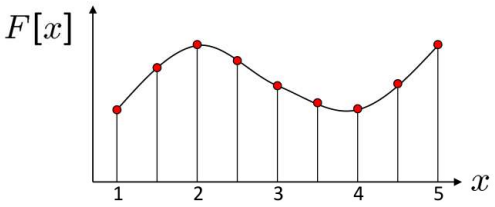
Recall how a digital image is formed

$$F[x, y] = \text{quantize}\{f(xd, yd)\}$$

- It is a discrete point-sampling of a continuous function
- If we could somehow reconstruct the original function, any new image could be generated, at any resolution and scale

19

Image interpolation



d = 1 in this example

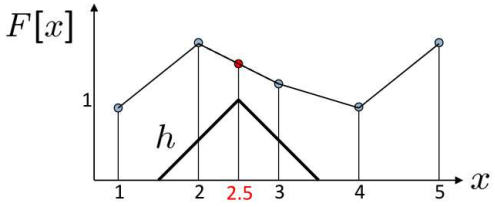
Recall how a digital image is formed

$$F[x, y] = \text{quantize}\{f(xd, yd)\}$$

- It is a discrete point-sampling of a continuous function
- If we could somehow reconstruct the original function, any new image could be generated, at any resolution and scale

20

Image interpolation



d = 1 in this example

- What if we don't know f ?
  - Guess an approximation: $\tilde{f}$
  - Can be done in a principled way: filtering
  - Convert F to a continuous function:

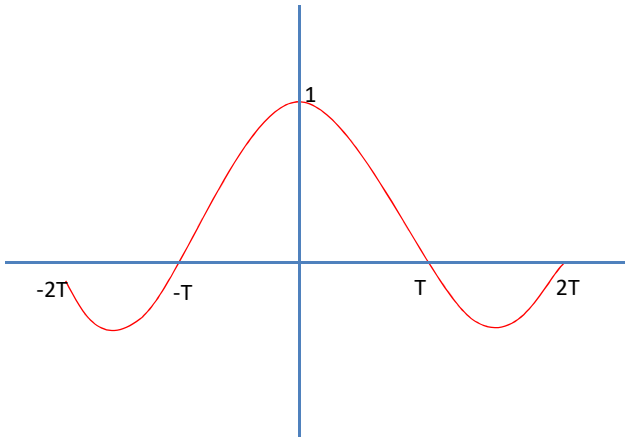
$$f_F(x) = F(\tfrac{x}{d}) \text{ when } \tfrac{x}{d} \text{ is an integer, 0 otherwise}$$

  - Reconstruct by convolution with a *reconstruction filter, h*

$$\tilde{f} = h * f_F$$

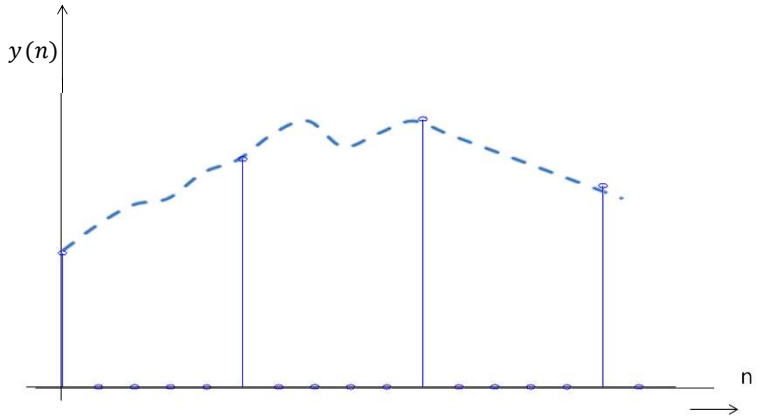
21

Bicubic Interpolation

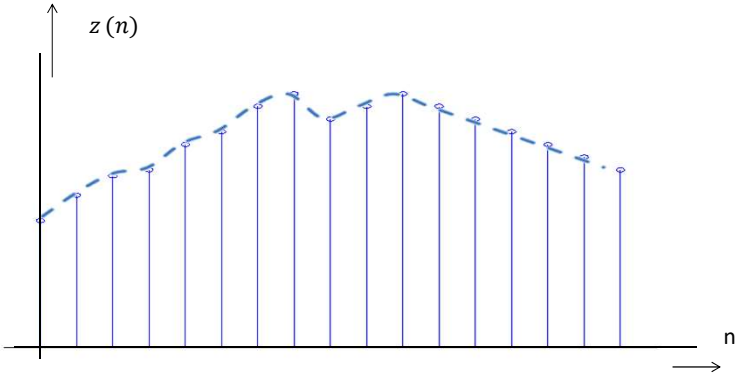


22

Bicubic Interpolation

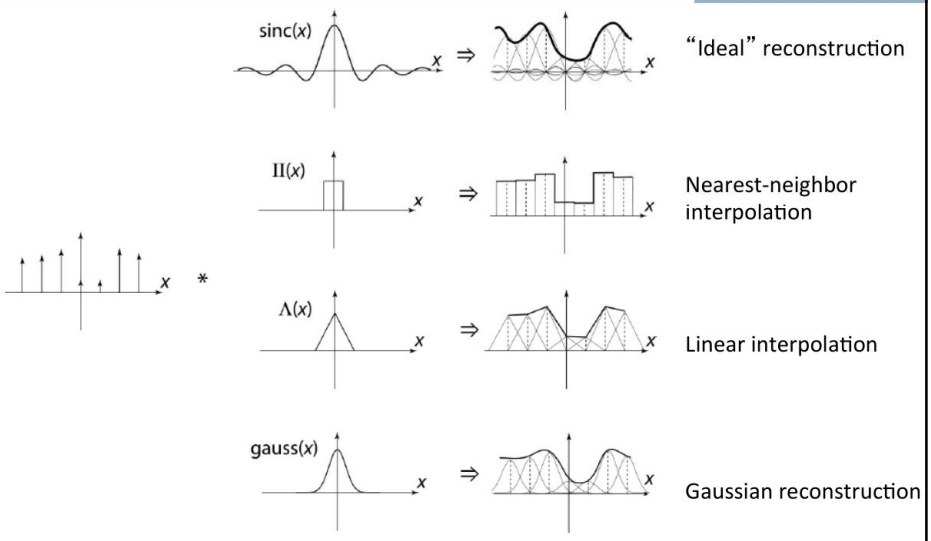


23



24

Image interpolation



25

Another example

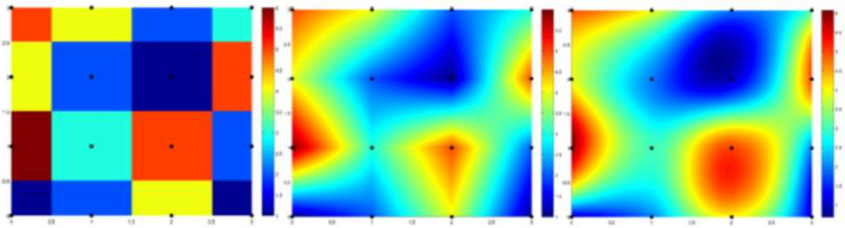
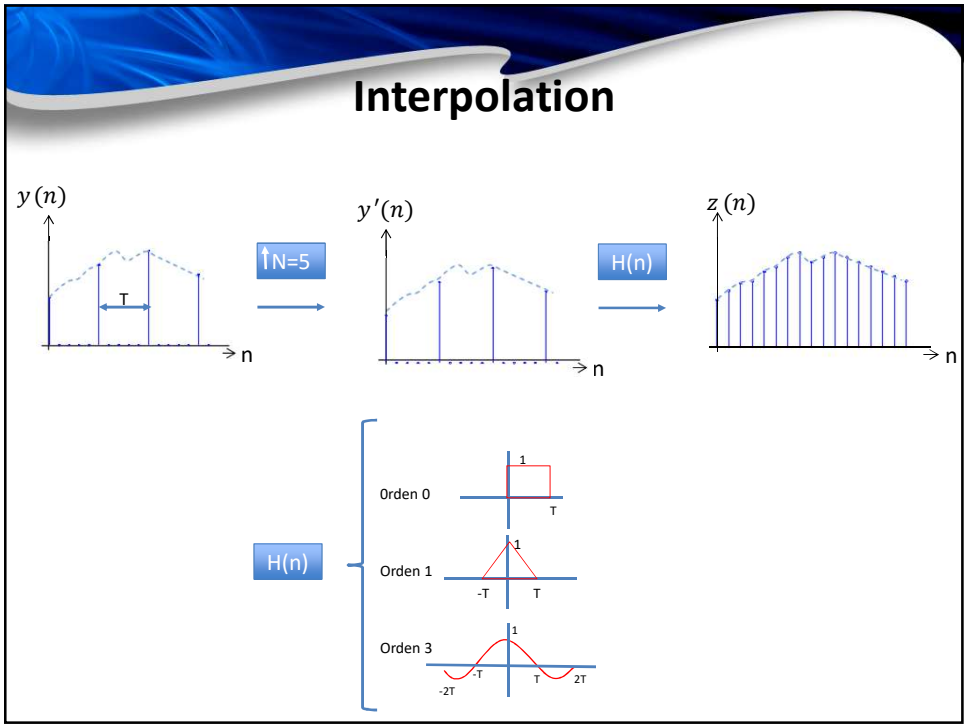
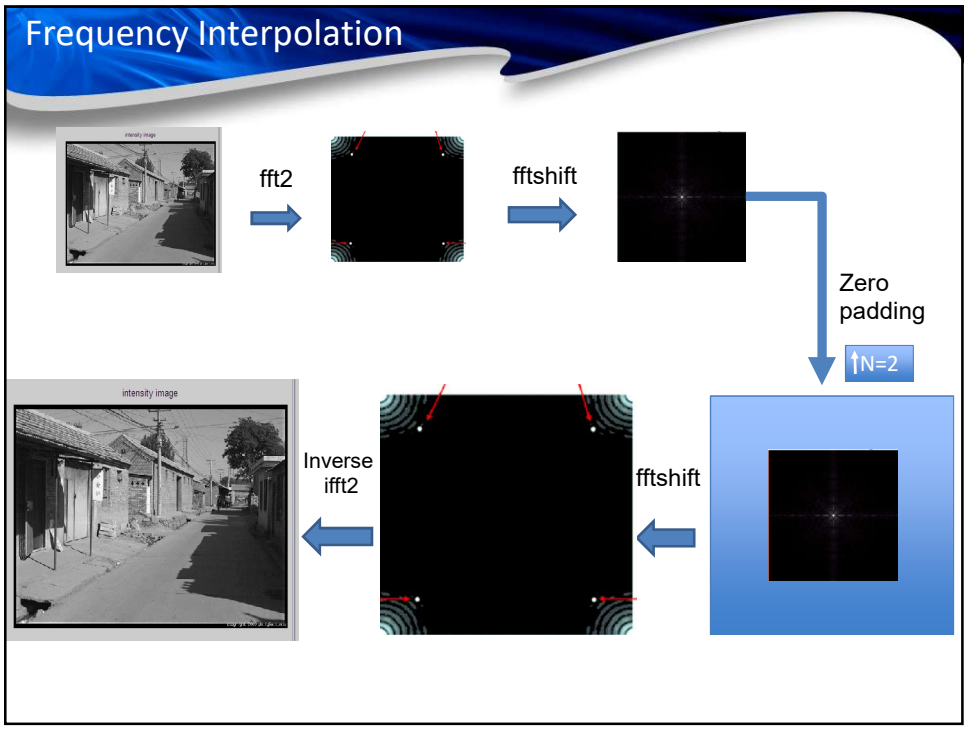


Figure: Nearest, bilinear and bicubic intepolations

26


Interpolation
y (n)
T
n
N=5
y′(n)
n
H(n)
z (n)
n
H(n)
Orden 0
1
T
Orden 1
1
-T
T
Orden 3
1
-T
T
2T
-2T

27

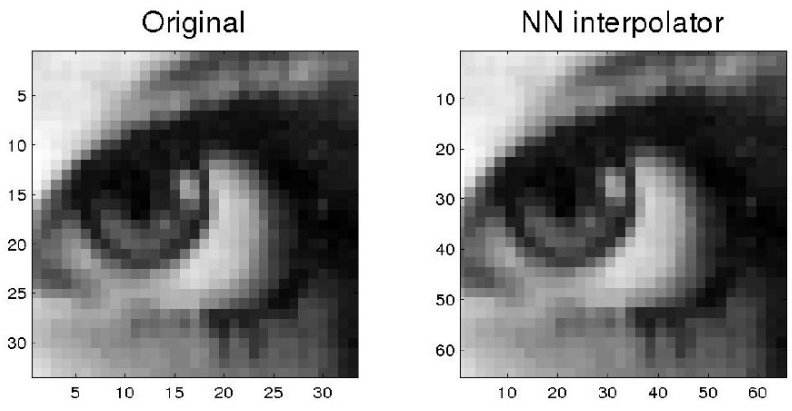

Frequency Interpolation
fft2
fftshift
Zero
padding
N=2
fftshift
Inverse
ifft2
intensity image

28

Matlab-generic interpolation

```
clear all
I=imread('lena_eye.png');
I=double(I);

[m n]= size(I);
[x,y] = meshgrid(1:n, 1:m);              % grid of input image

r=0.5;                                   % scale factor
[p,q]=meshgrid(1:r:n, 1:r:m);            % grid for output image
I2=interp2(x,y,I,p,q,'nearest');         % interpolation
                                         % 'nearest', ...
                                             'bilinear','bicubic'

figure
subplot(1,2,1),imagesc(I),axis image
title('Original','FontSize',18)
subplot(1,2,2),imagesc(I2),axis image
title('NN interpolator','FontSize',18)
colormap(gray)

print -djpeg eye_ori_NN.jpg
```

29



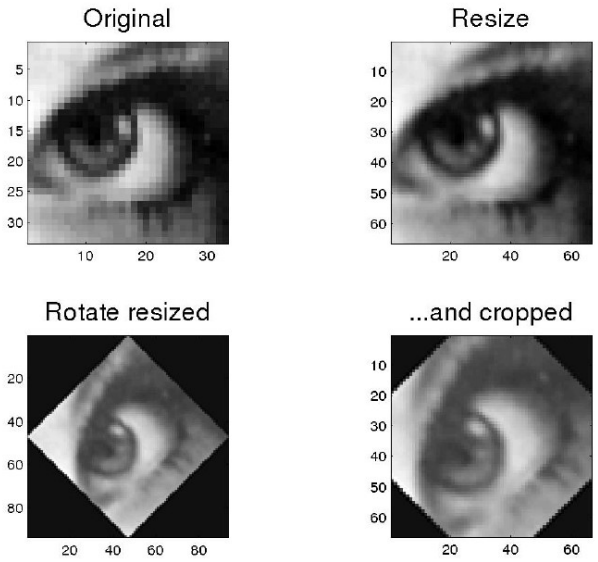
30

Image interpolation-Direct commands

```
clear all
I=imread('lena_eye.png');
I=double(I);

r=2;theta=45;
I2=imresize(I,r,'bicubic');              % resize by factor r
I3=imrotate(I2,theta,'bicubic');         % rotate theta degrees
I4=imrotate(I2,theta,'bicubic','crop'); % 'crop'-> original size

figure
subplot(2,2,1),imagesc(I),axis image
title('Original','FontSize',18)
subplot(2,2,2),imagesc(I2),axis image
title('Resize','FontSize',18)
subplot(2,2,3),imagesc(I3),axis image
title('Rotate resized','FontSize',18)
subplot(2,2,4),imagesc(I4),axis image
title('...and cropped','FontSize',18)
colormap(gray)

print -djpeg eye_several.jpg
```

31

Transformations



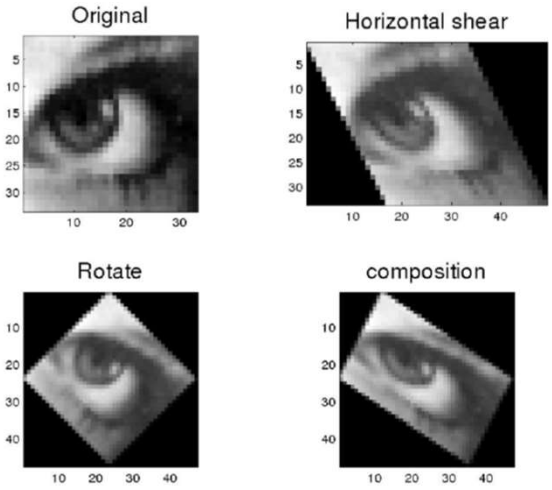
32

Affine transformations

```
clear all
I=imread('lena_eye.png');
I=double(I);

tform2 = maketform('affine',[1 0 0; .5 1 0; 0 0 1]);          % ...
    shear
I2 = imtransform(I,tform2);

theta=pi/4;                                                    % ...
    rotation
A=[cos(theta) sin(theta) 0; -sin(theta) cos(theta) 0; 0 0 1];
tform3 = maketform('affine',A);
I3 = imtransform(I,tform3);

tform4 = maketform('composite',[tform2,tform3]);               % ...
    composition
I4 = imtransform(I,tform4);

figure
subplot(2,2,1),imagesc(I),axis image
title('Original','FontSize',18)
subplot(2,2,2),imagesc(I2),axis image
```

33

Affine Transformations



34