National University of Sciences and Technology (NUST)

Digital Image Processing

Khawar Khurshid

Sampling
Quantization
Interpolation

Sampling and Quantization

Sampling:

Digitization of the spatial coordinates (x,y)

Quantization:

Digitization in amplitude (also known as gray level quantization)

- 8 bit quantization: $2^8 = 256$ gray levels

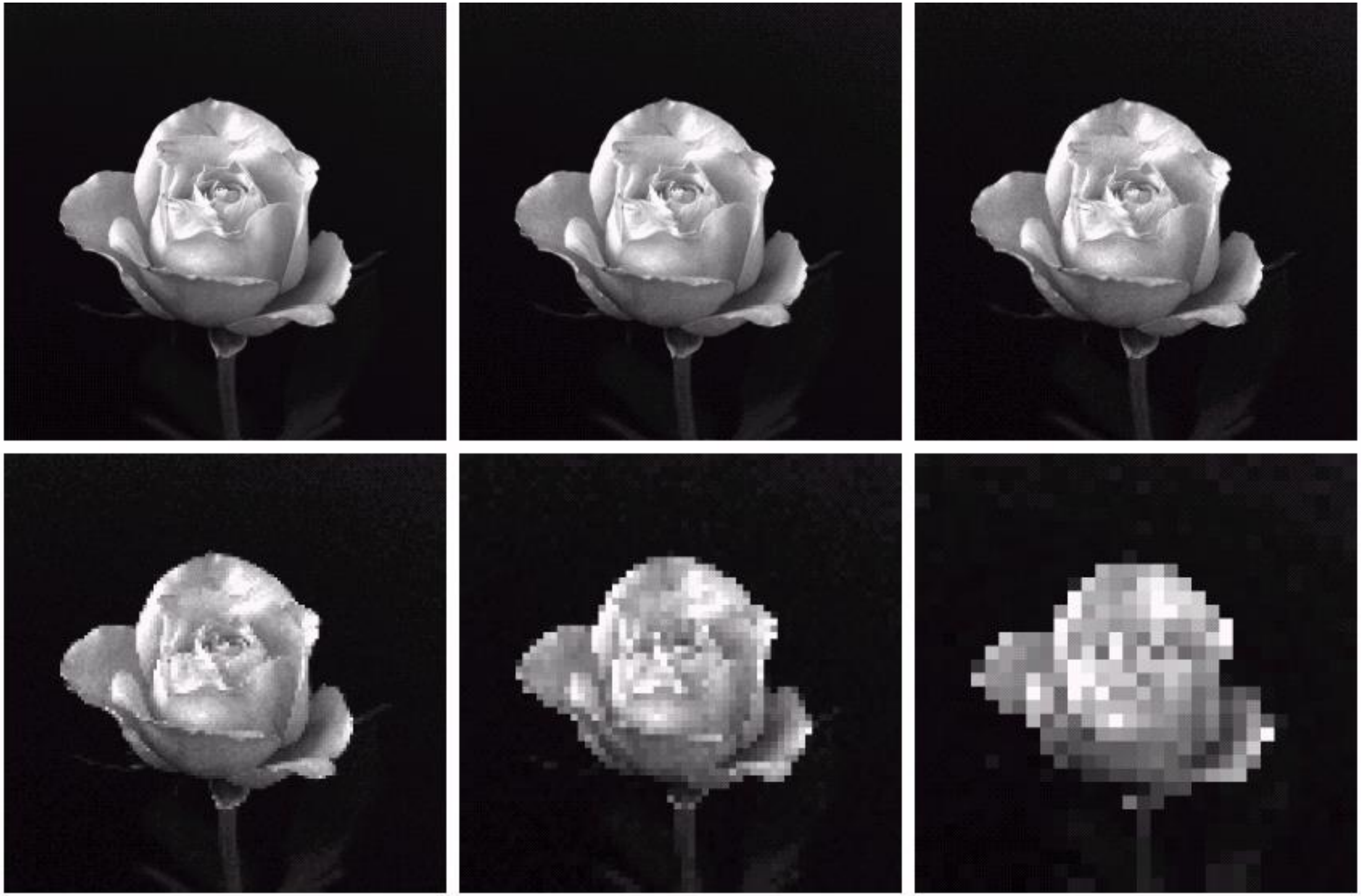
  (0: black, 255: white)

- 1 bit quantization: $2^1 = 2$ gray levels

  (0: black, 1: white)

Sampling

Sampling

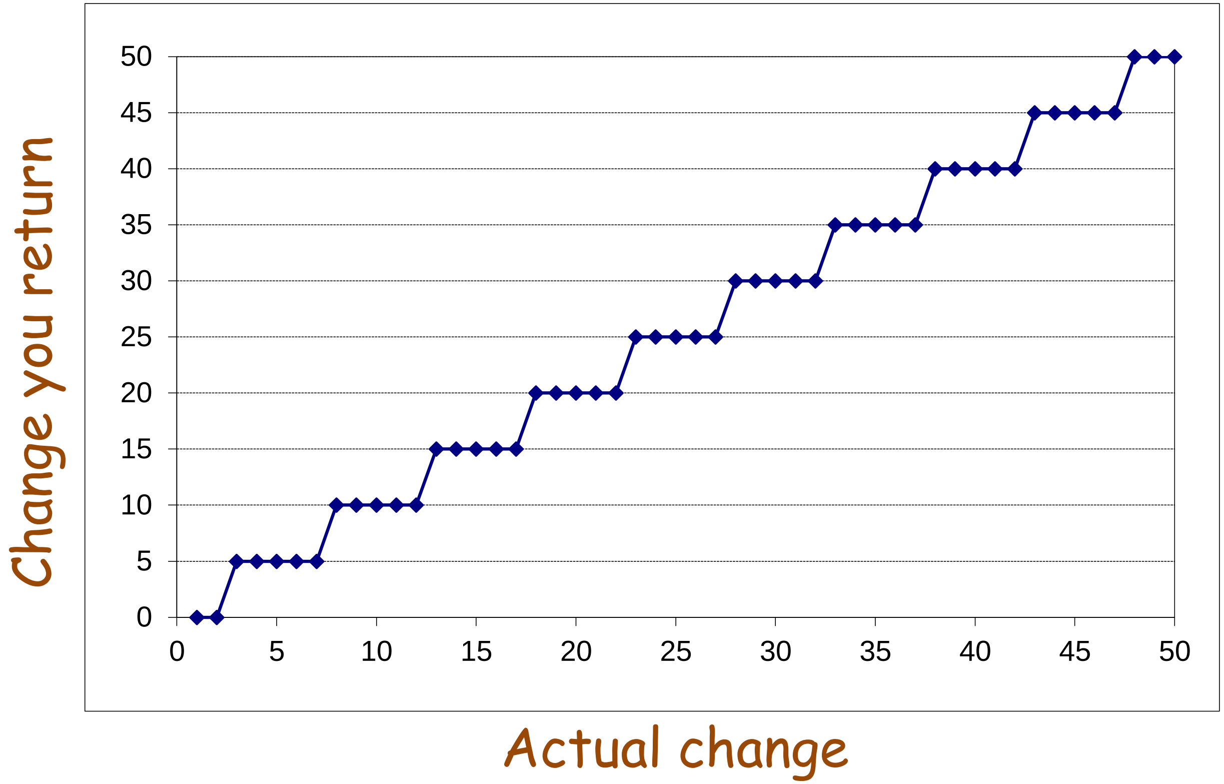


Quantization – (Example)

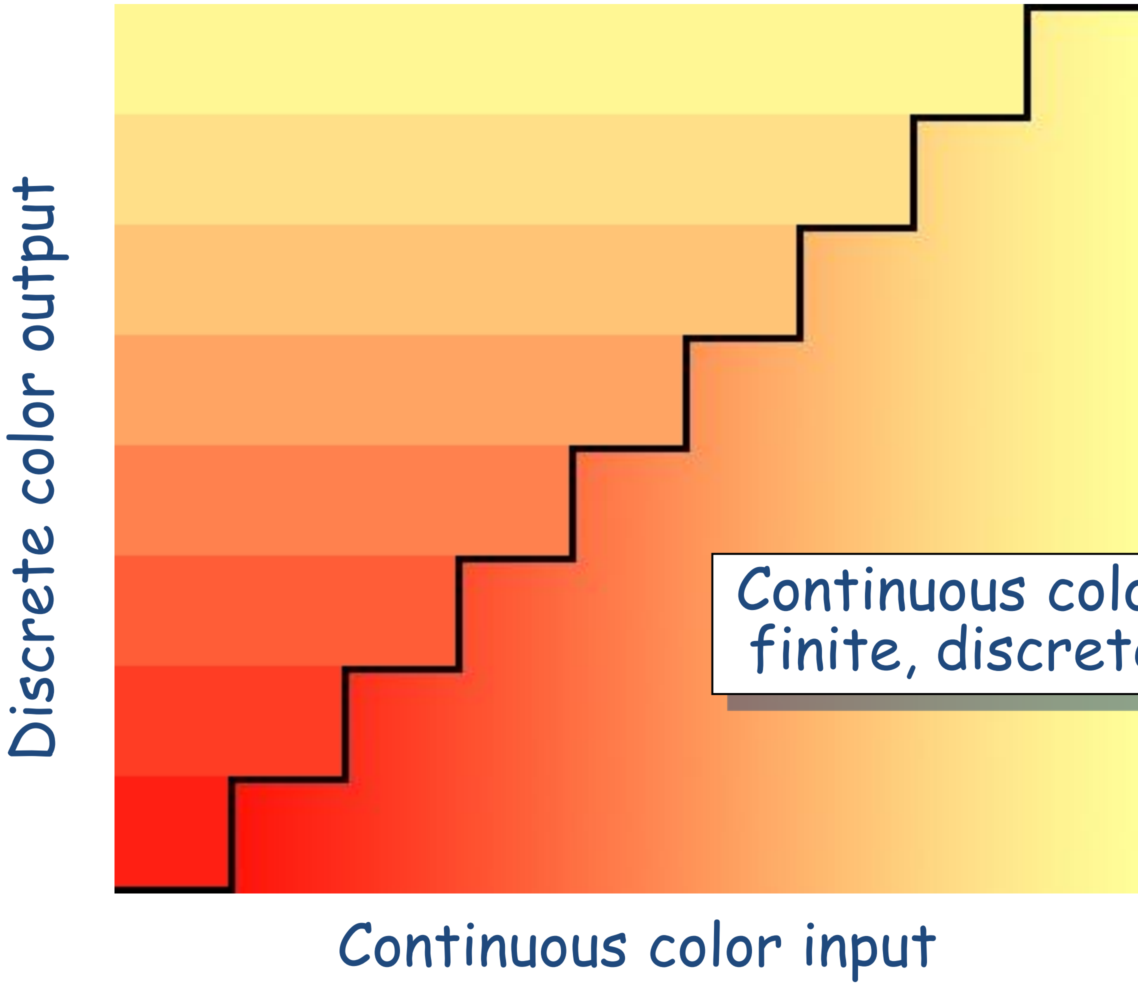
Return change using only these notes

Quantization – (Example)

For Rs. 3	Return 5
For Rs. 7	Return 5
For Rs. 9	Return 10
For Rs. 12	Return 10
For Rs. 23	Return 25
…	…



Quantization – (Example)



Continuous colors mapped to a finite, discrete set of colors.

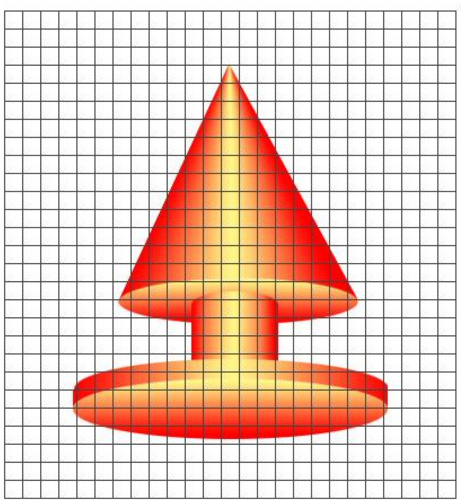
Quantization – (Example)

16 million colors

16 colors

Sampling and Quantization

Pixel Grid



Real Image

Sampled

Quantized

Sampled &
Quantized

Sampling and Quantization

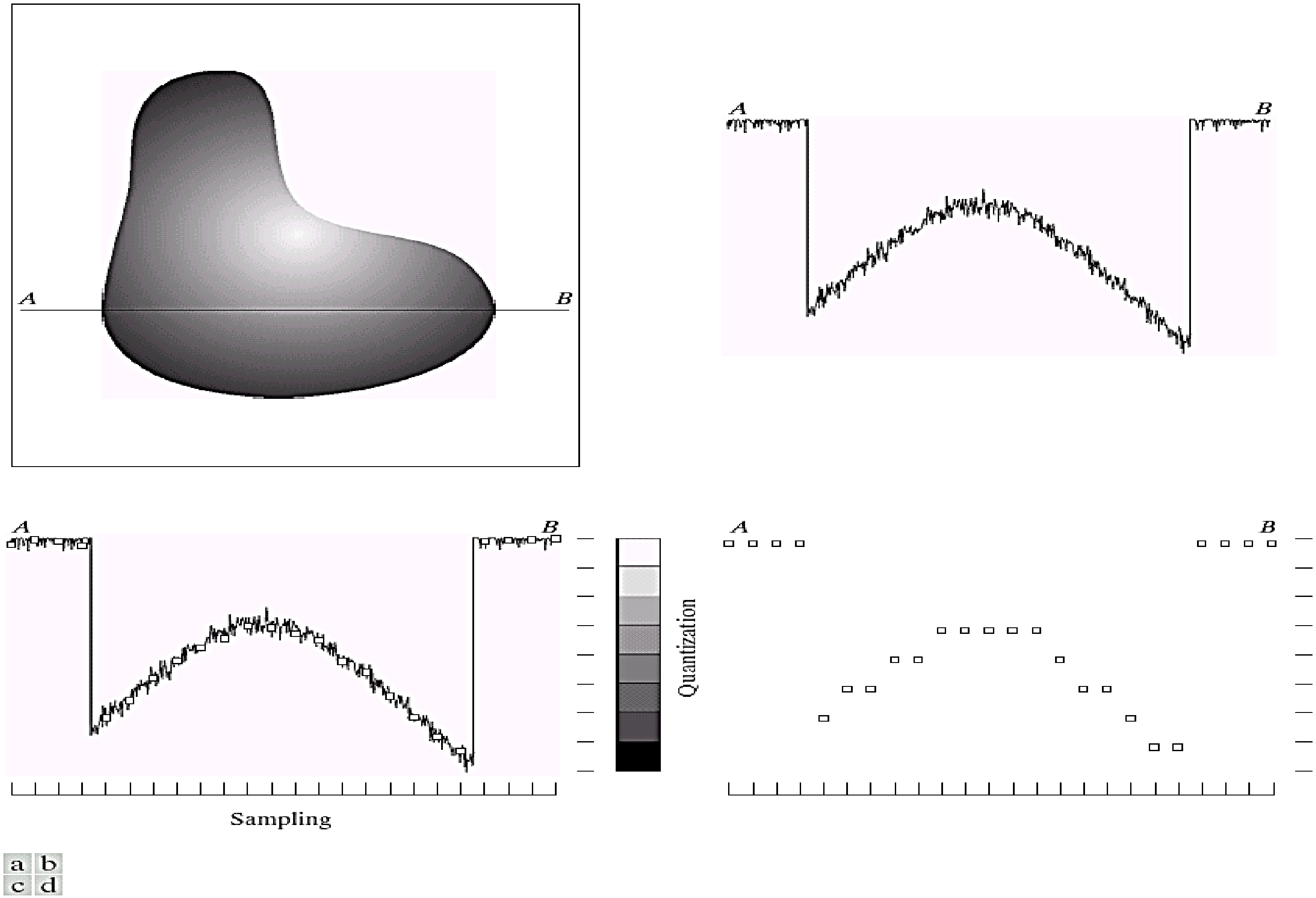
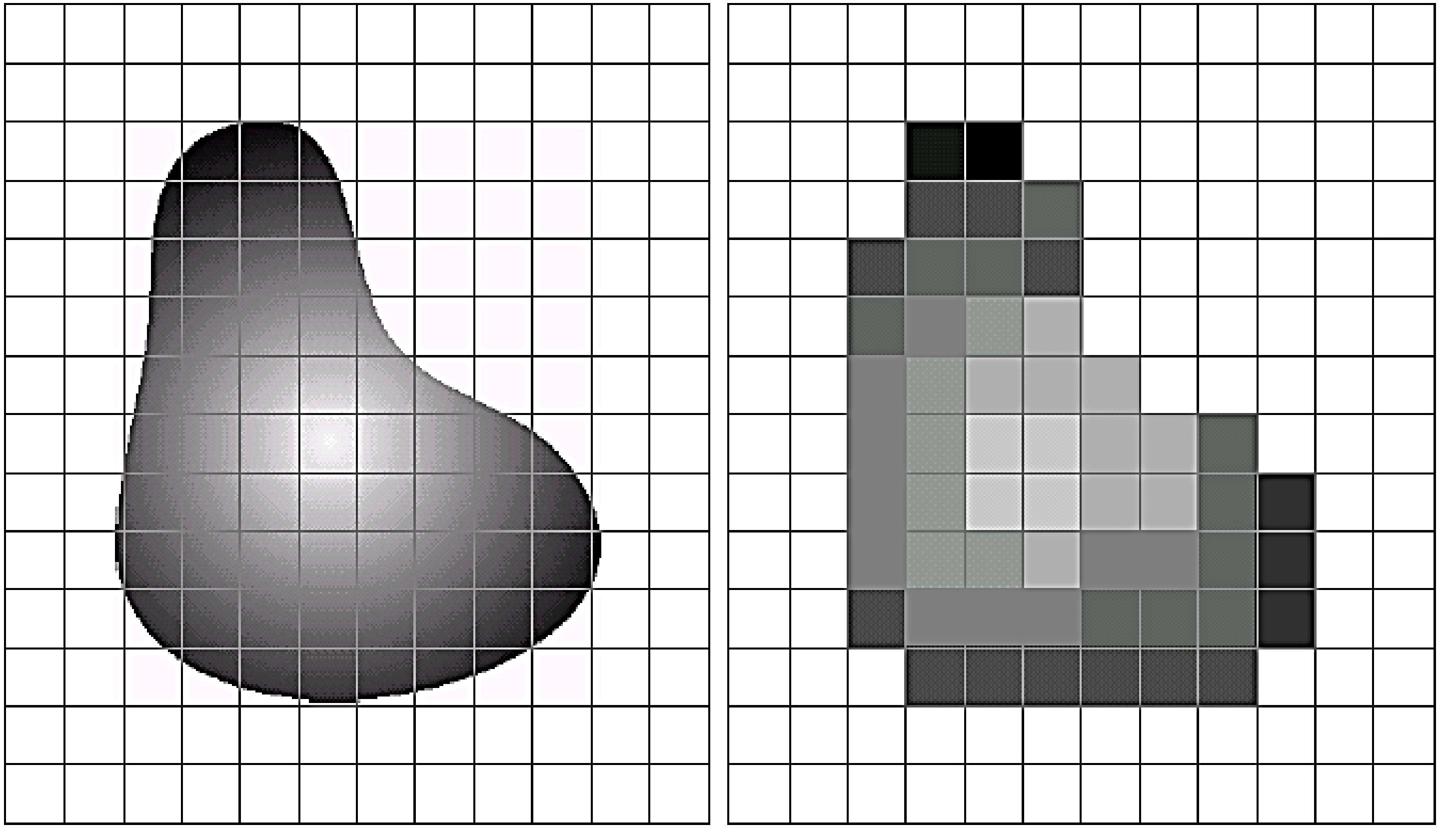


FIGURE 2.16 Generating a digital image. (a) Continuous image. (b) A scan line from A to B in the continuous image, used to illustrate the concepts of sampling and quantization. (c) Sampling and quantization. (d) Digital scan line.

Sampling and Quantization



a b

FIGURE 2.17 (a) Continuos image projected onto a sensor array. (b) Result of image sampling and quantization.

Sampling and Quantization

256x256
256 gray levels

128x128

64x64

32x32

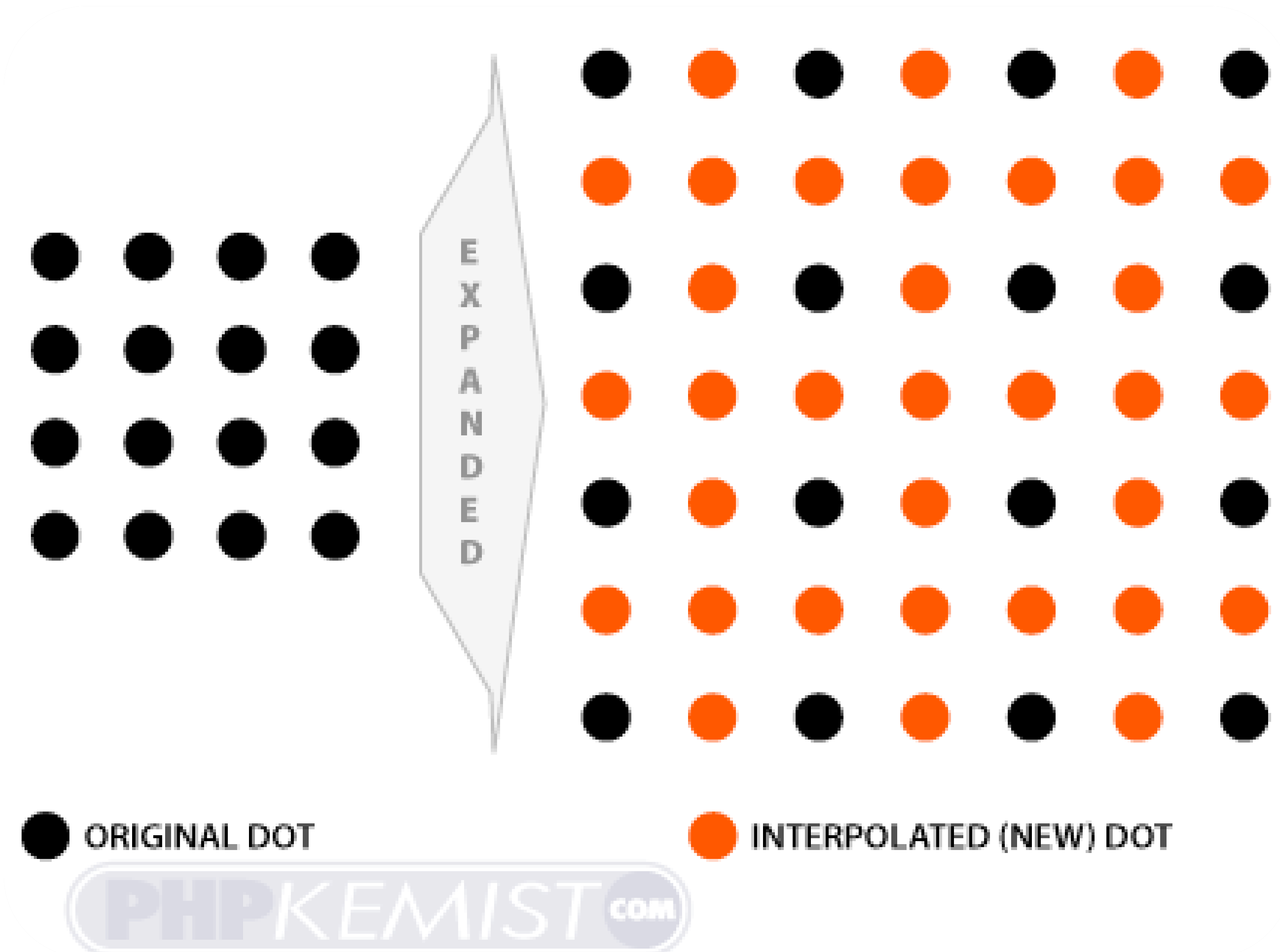
16 gray levels

8 gray levels

4 gray levels

Binary Image

Interpolation



Interpolation

Why?

- Changing Perspective
- Rotation
- Resizing
- Warping

Interpolation

Why?

- Changing Perspective
- Rotation
- Resizing
- Warping

Interpolation

Why?

- Changing Perspective
- Rotation
- Resizing
- Warping

Interpolation

Why?

- Changing Perspective
- Rotation
- Resizing
- Warping

Interpolation

Why?

- Changing Perspective
- Rotation
- Resizing
- Warping

Nearest Neighbor Replacement

Simply Replicate the value from neighboring pixels.

1	0	1
1	1	0
1	0	1

1			0			1
1			1			0
1			0			1

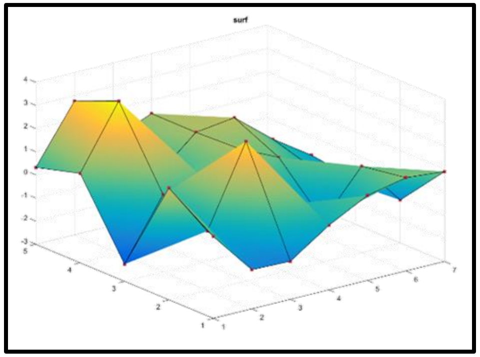
Nearest Neighbor Replacement

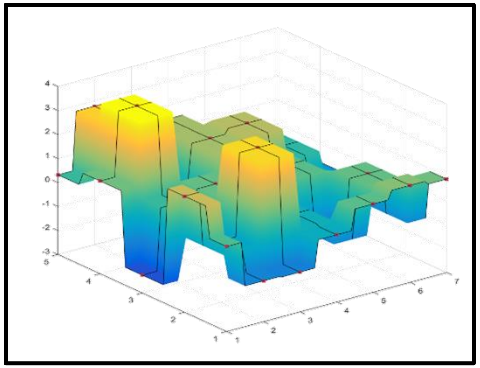
Adds blocky effect.

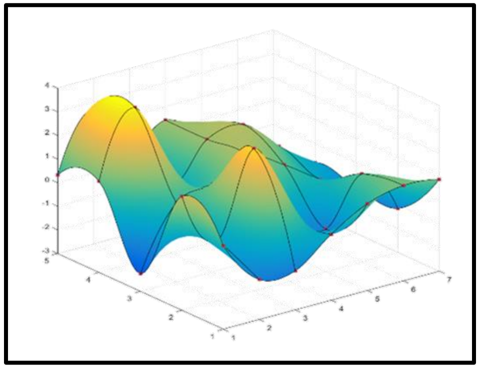
1	0	1
1	1	0
1	0	1

1	1	0	0	0	1	1
1	1	0	0	0	1	1
1	1	1	1	1	0	0
1	1	1	1	1	0	0
1	1	1	1	1	0	0
1	1	0	0	0	1	1
1	1	0	0	0	1	1

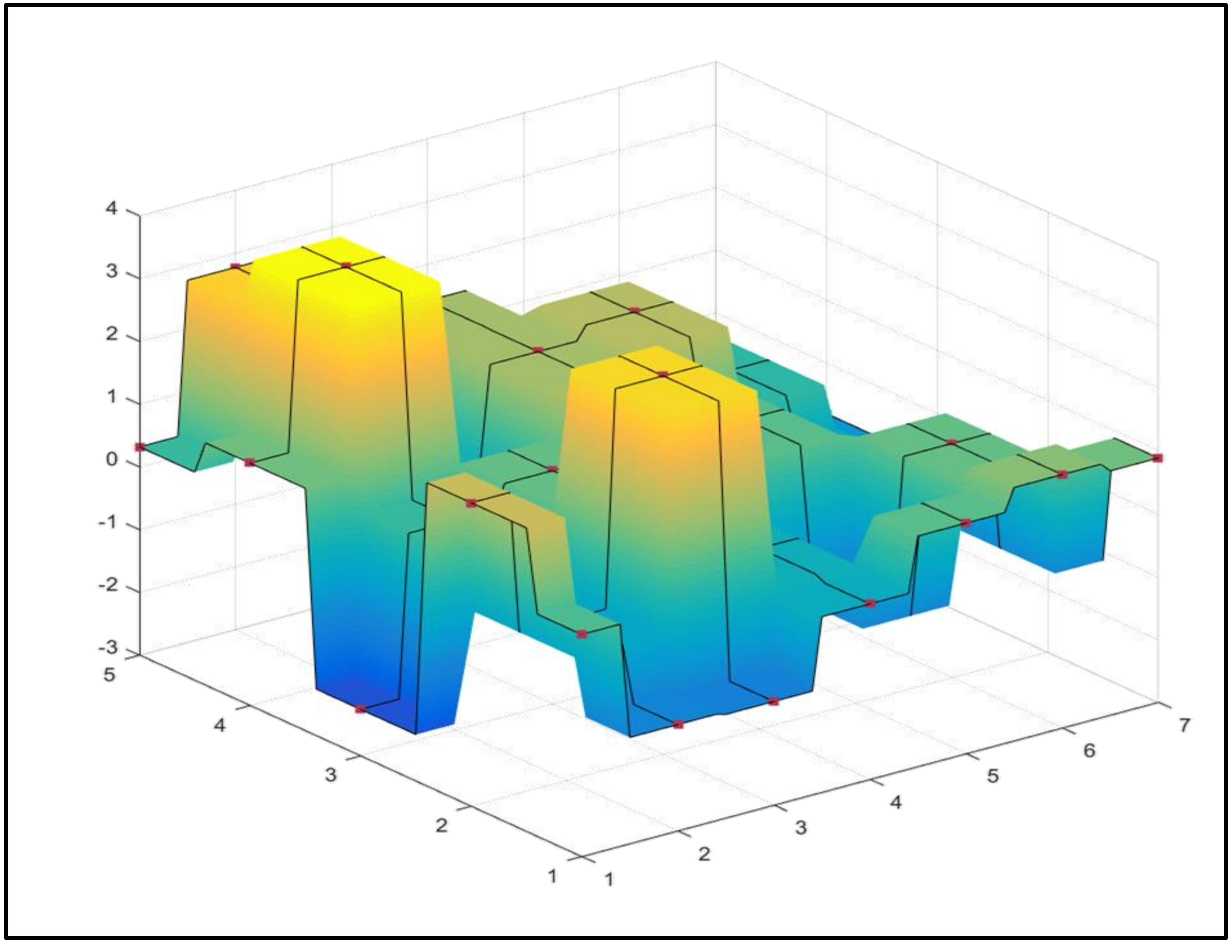
Interpolation







Nearest-neighbor Interpolation

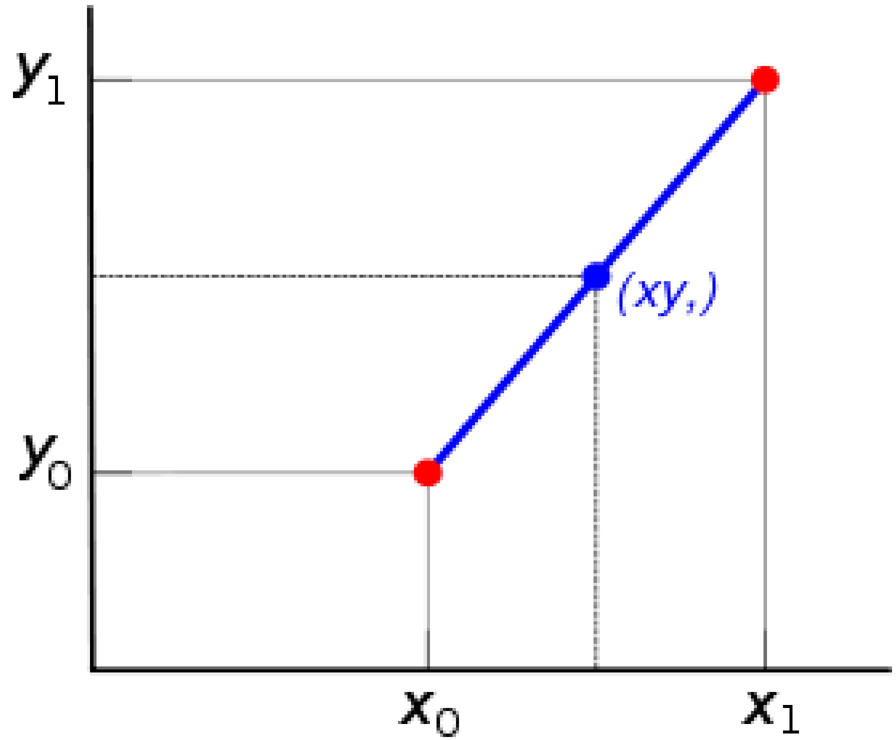


Linear Interpolation

For example: if you wanted to know the temperature at 12PM

but only measured it at 11AM and 1PM.

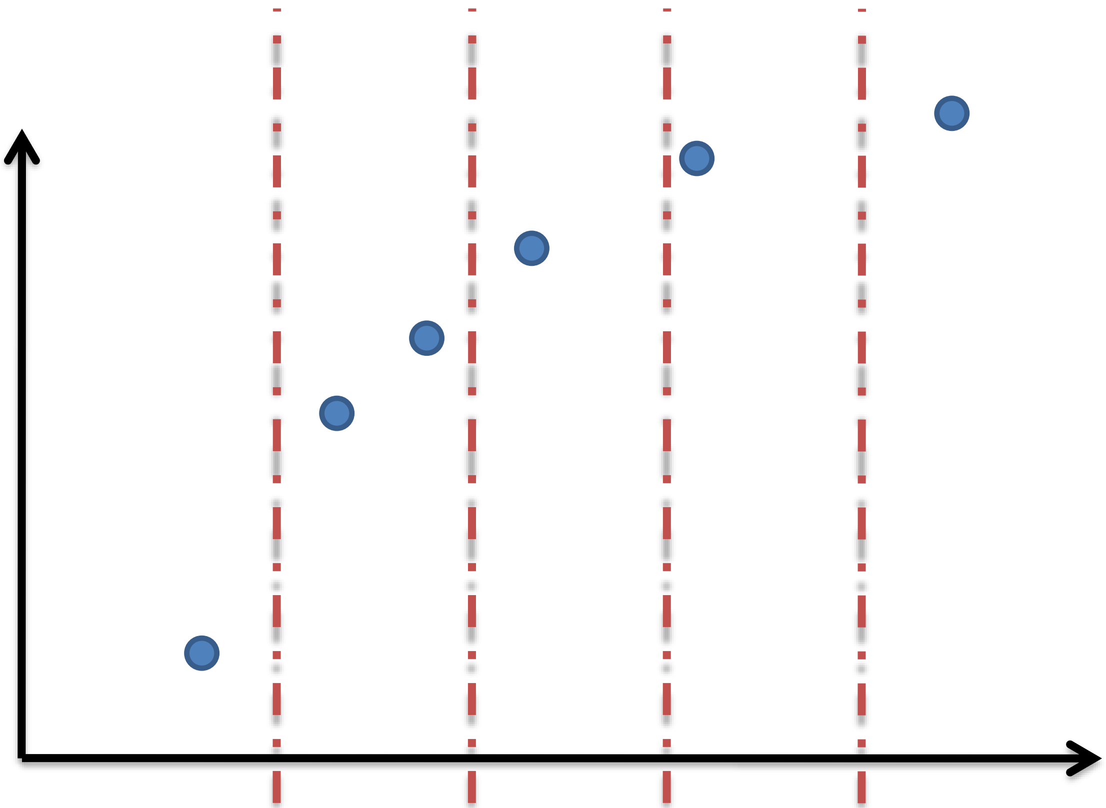
you could estimate its value by performing a linear interpolation:



1-D Interpolation

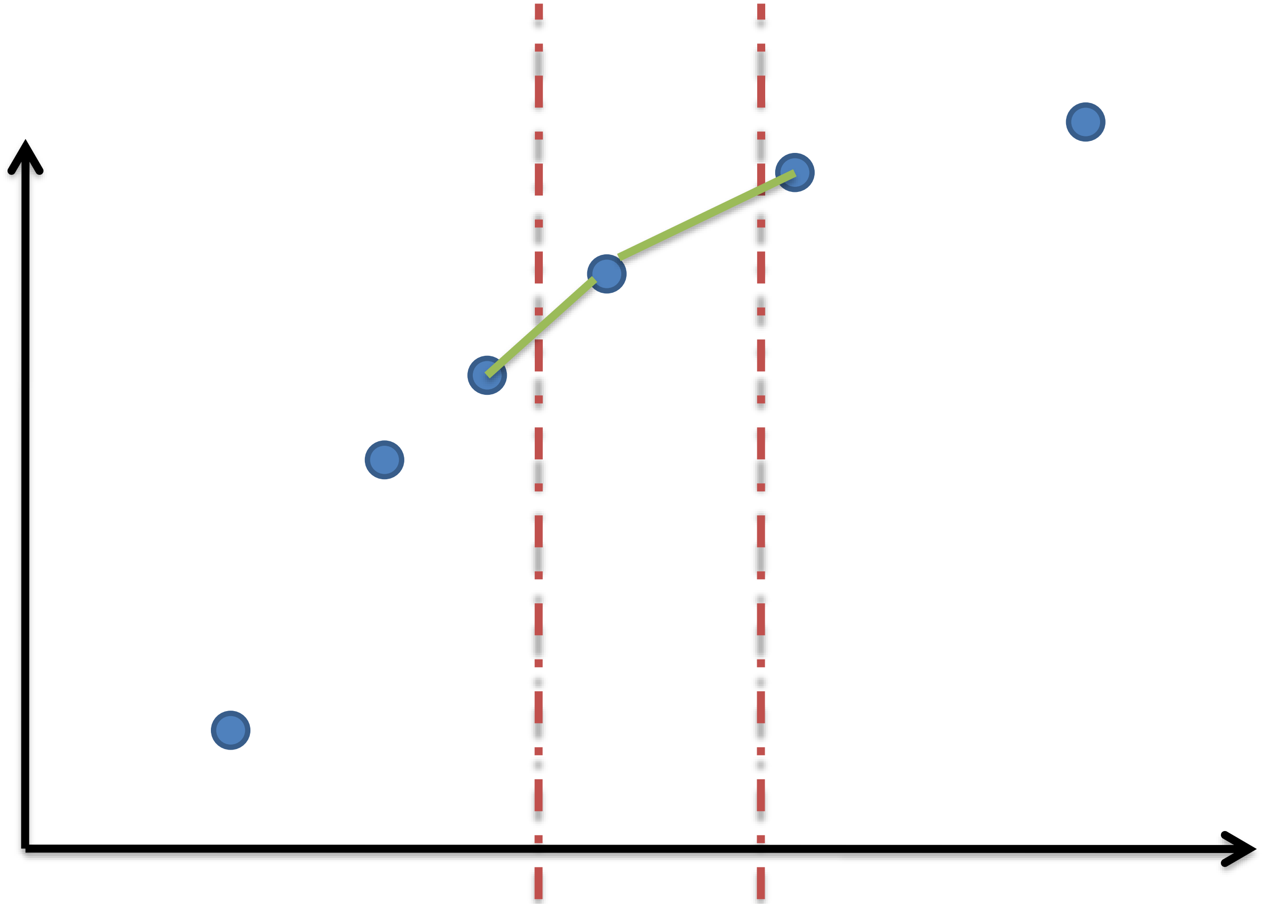
NUST

Interpolation works by using known data to estimate values at unknown points.



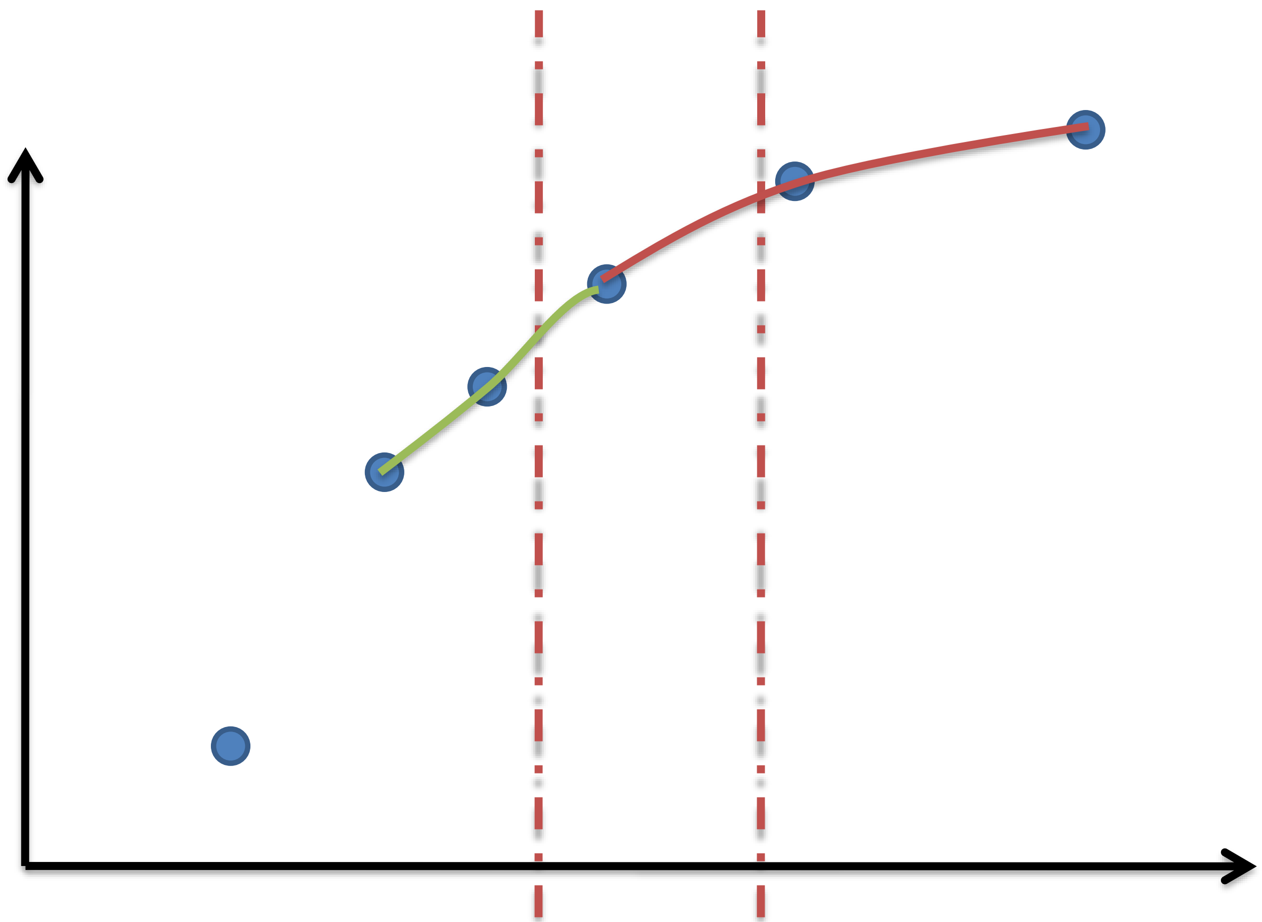
1-D Interpolation

Linear Interpolation: Fit a linear function piecewise between the points.



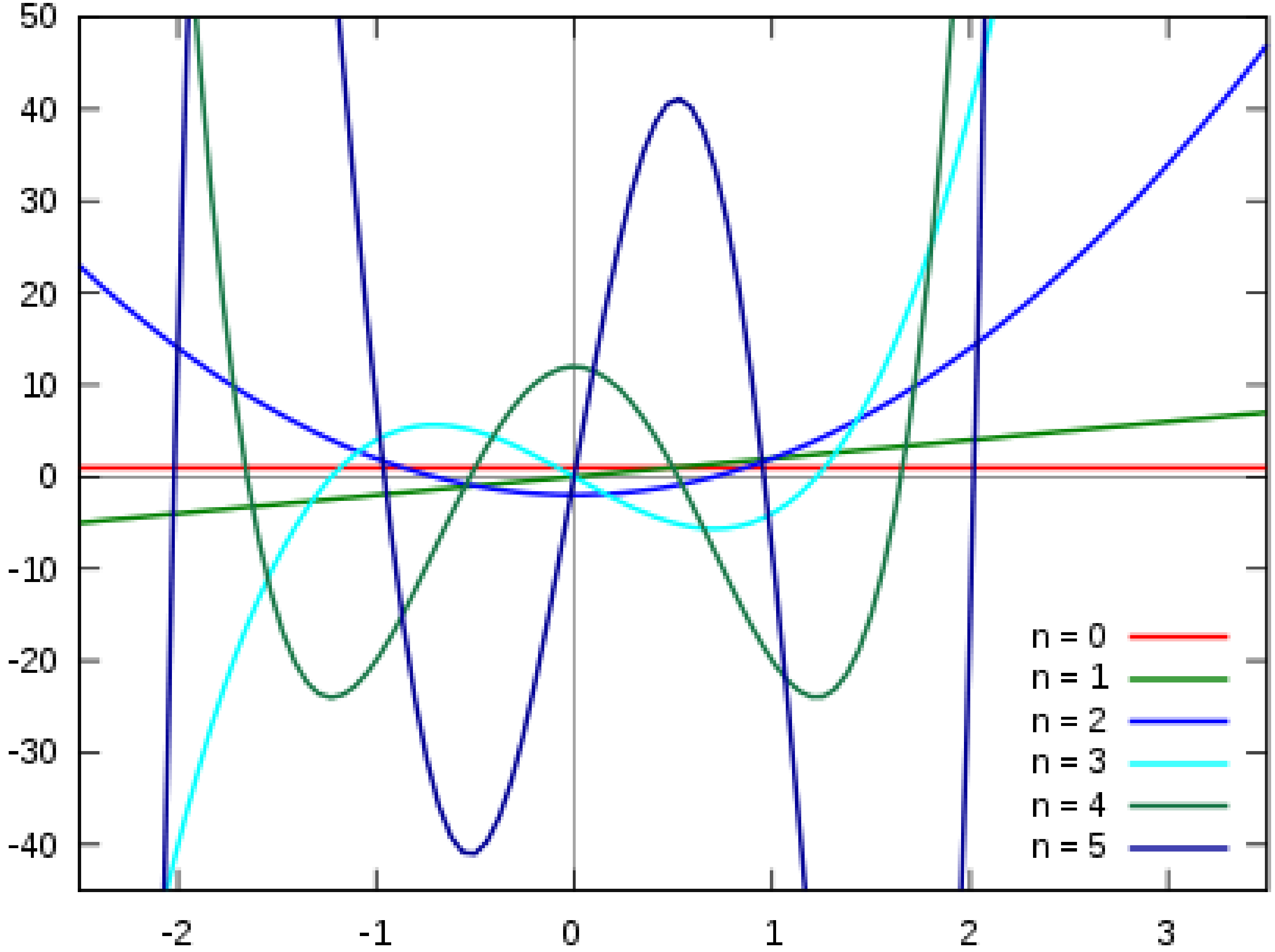
1-D Interpolation

Quadratic: Fit a Quadratic Polynomial between the three points.



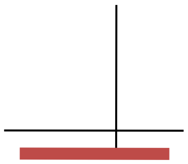
1-D Interpolation

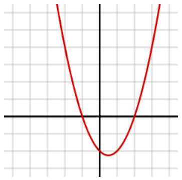
Fitting Polynomial to data points.



1-D Interpolation

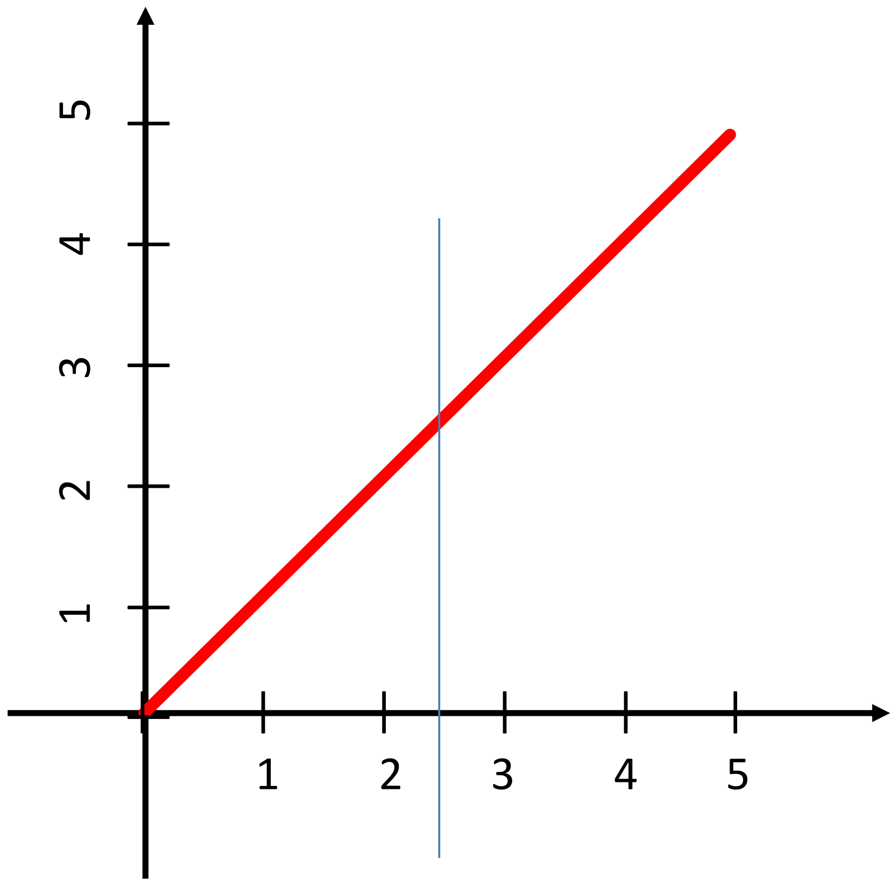
Degree	Polynomial	How it looks	Req. Pts.
Zero Polynomial	$f(x) = 0$		None
0	$f(x) = a_0$	horizontal line with *y*-intercept a_0	1
1 (Linear)	$f(x) = a_0 + a_1x$	an oblique line with *y*-intercept a_0 and slope a_1	2
2 (Quadratic)	$f(x) = a_0 + a_1x + a_2x^2$	Parabola	3
3 (Cubic)	$f(x) = a_0 + a_1x + a_2x^2 + a_3x^3$	Spline	4





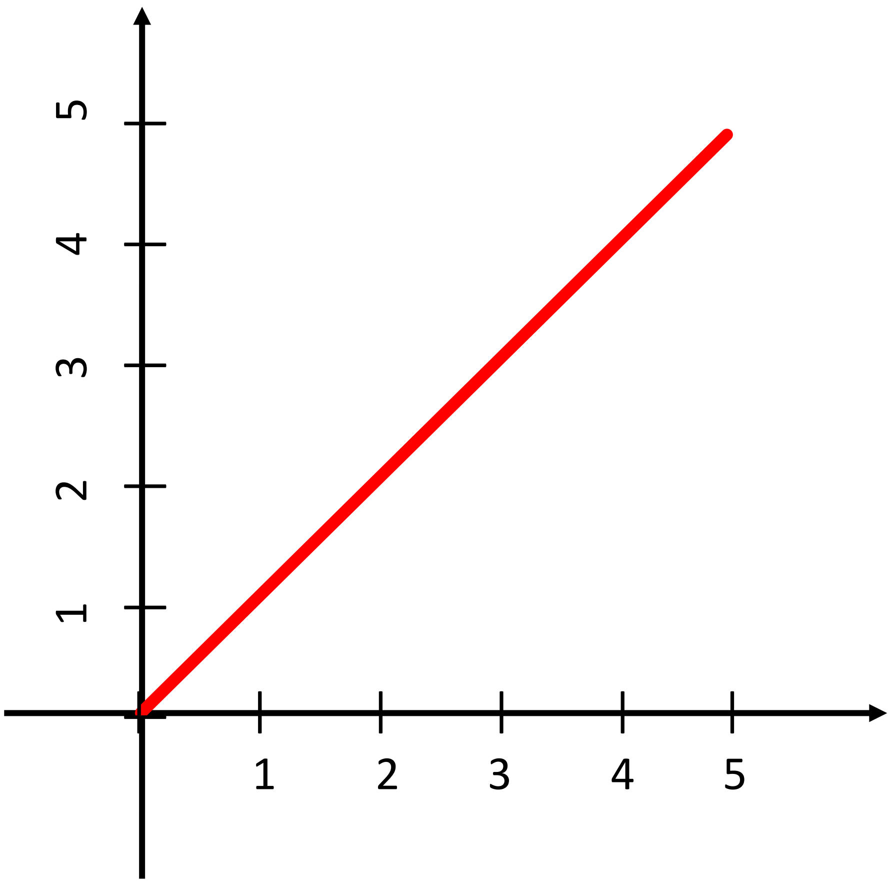
- No. of unknowns in a polynomial of degree n-1 = n
- Select n nearest known neighbours to generate n equations
- Solve the linear system of equations to estimate the constants
- Simply put the unknown x value and interpolate the y value

Linear Interpolation



What is the value of y at x= 2.5?

Linear Interpolation



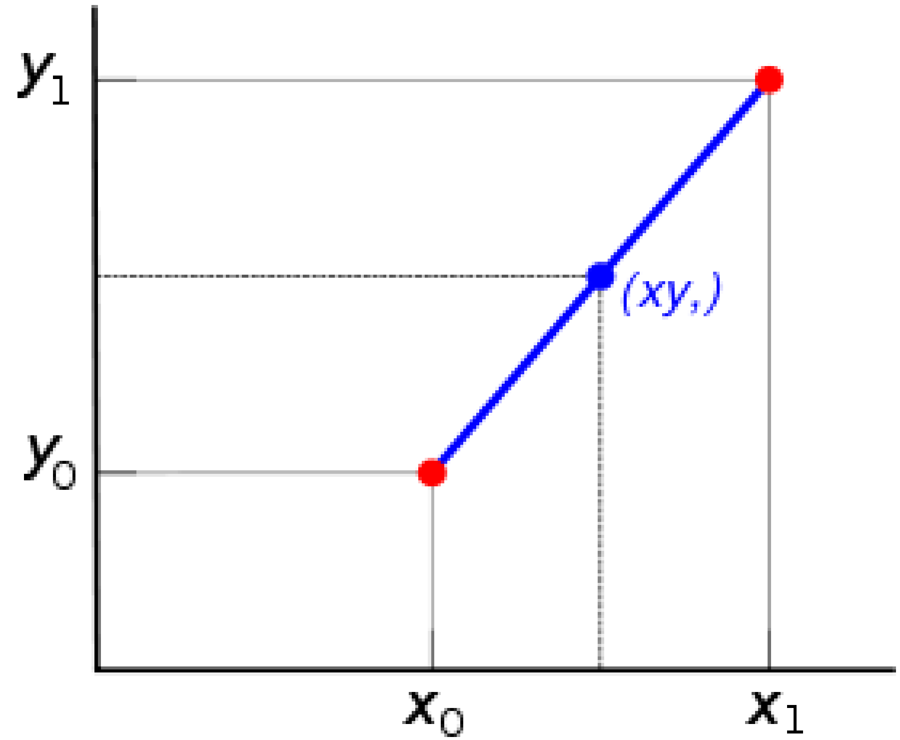
What is the value of y at
x= 2.5
Y= 2.5.

Linear Interpolation

- Equation of a line

$$\frac{y - y_0}{x - x_0} = \frac{y_1 - y_0}{x_1 - x_0}$$

$$y = \frac{y_1 - y_0}{x_1 - x_0}(x - x_0) + y_0$$

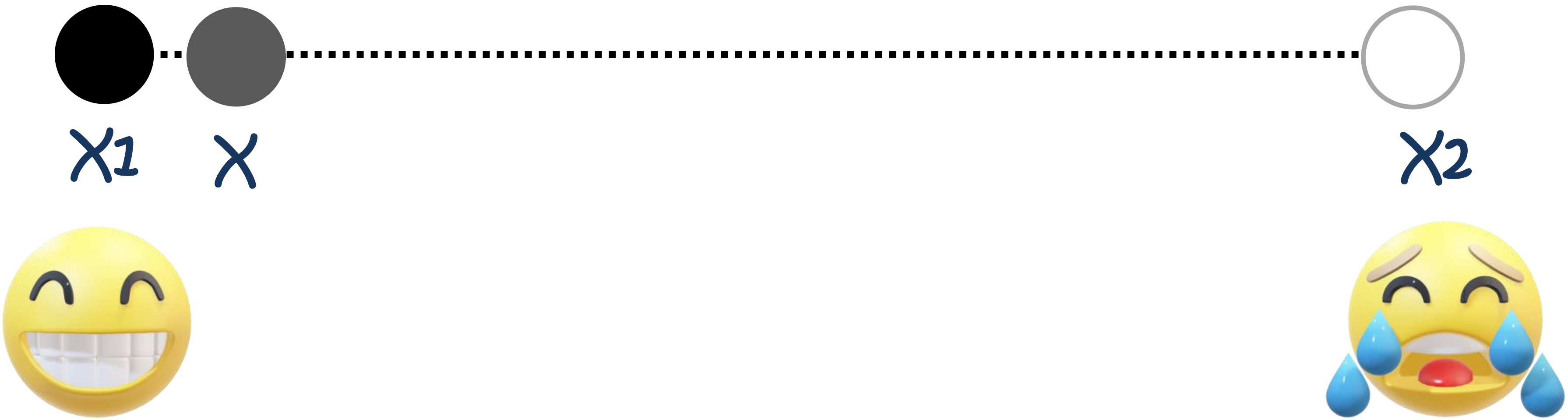


- Interpolation using Linear Polynomial
  - $y = a_0 + a_1 x$
  - Requires two unknowns

Activity

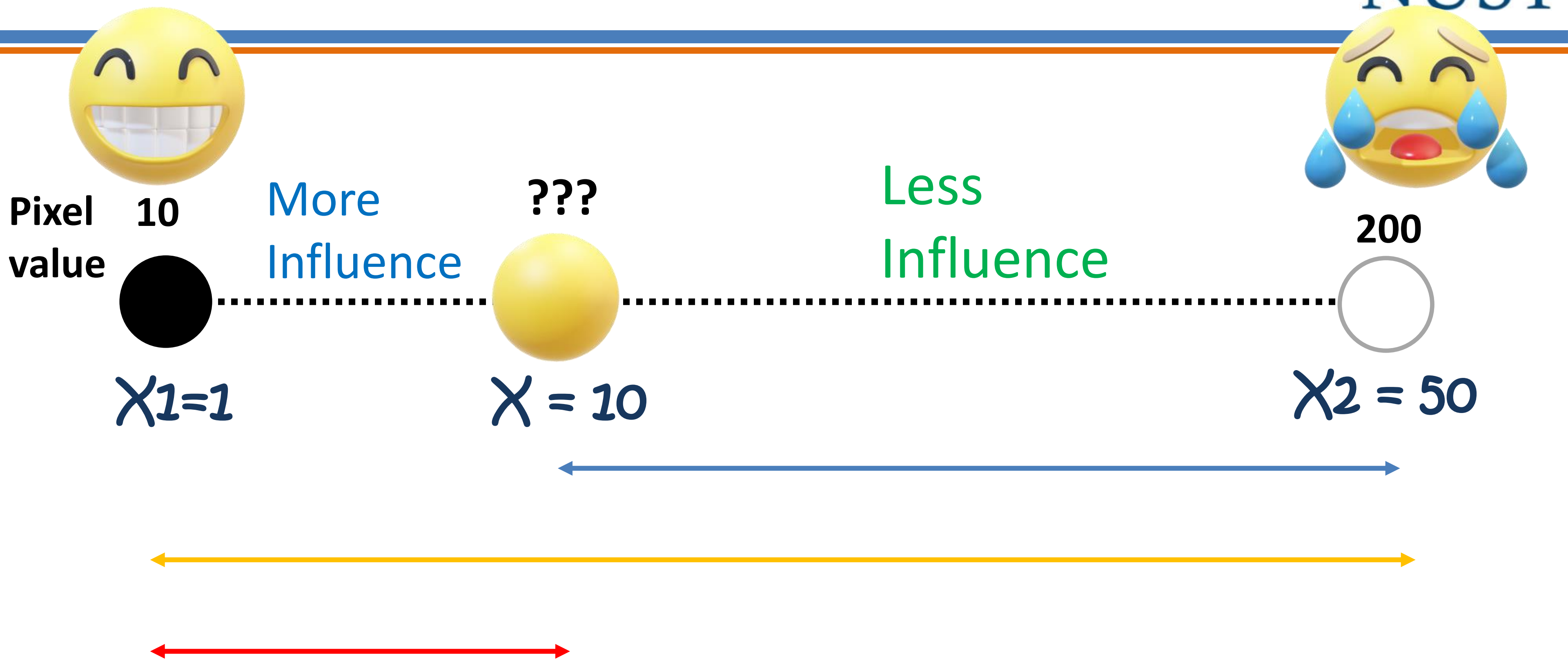
- 3-Volunteers

Bi-linear Interpolation



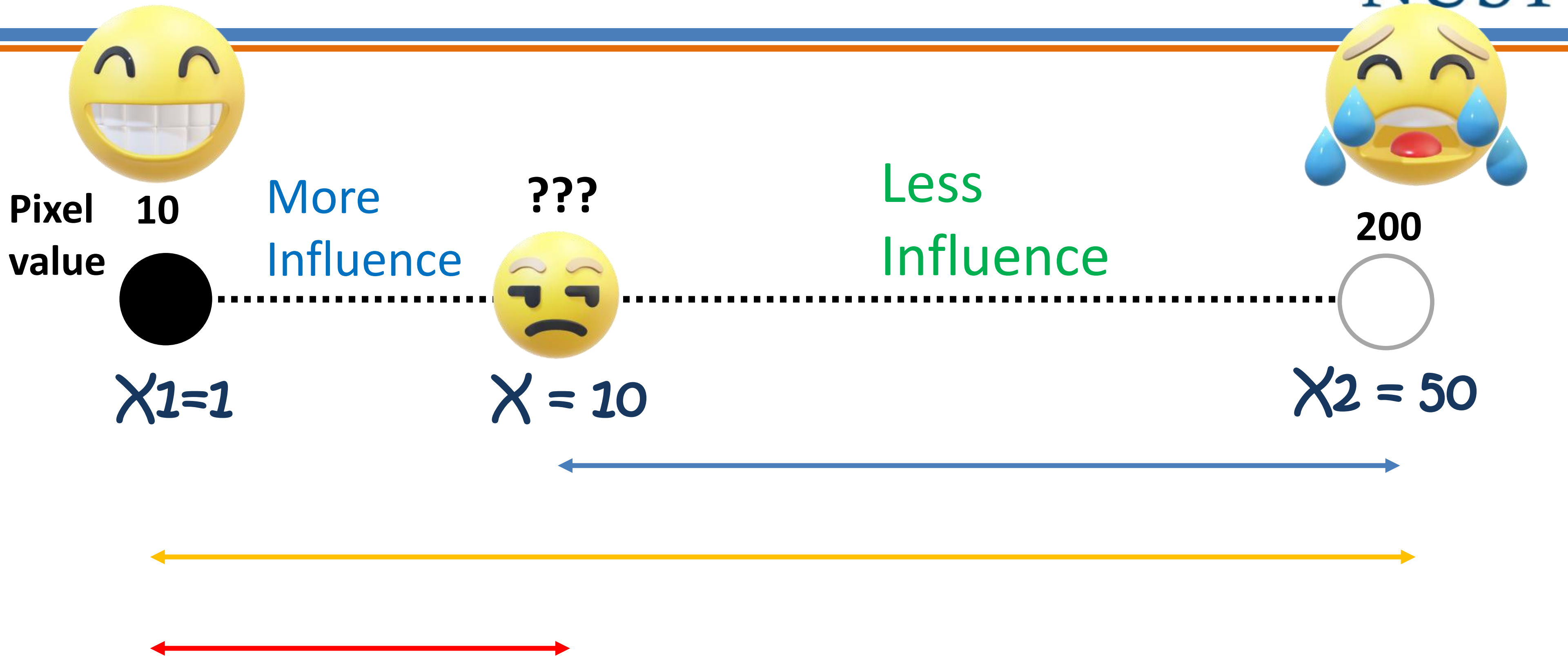
Which point has more influence on X ?

Bi-linear Interpolation



Calculate f(X) ?

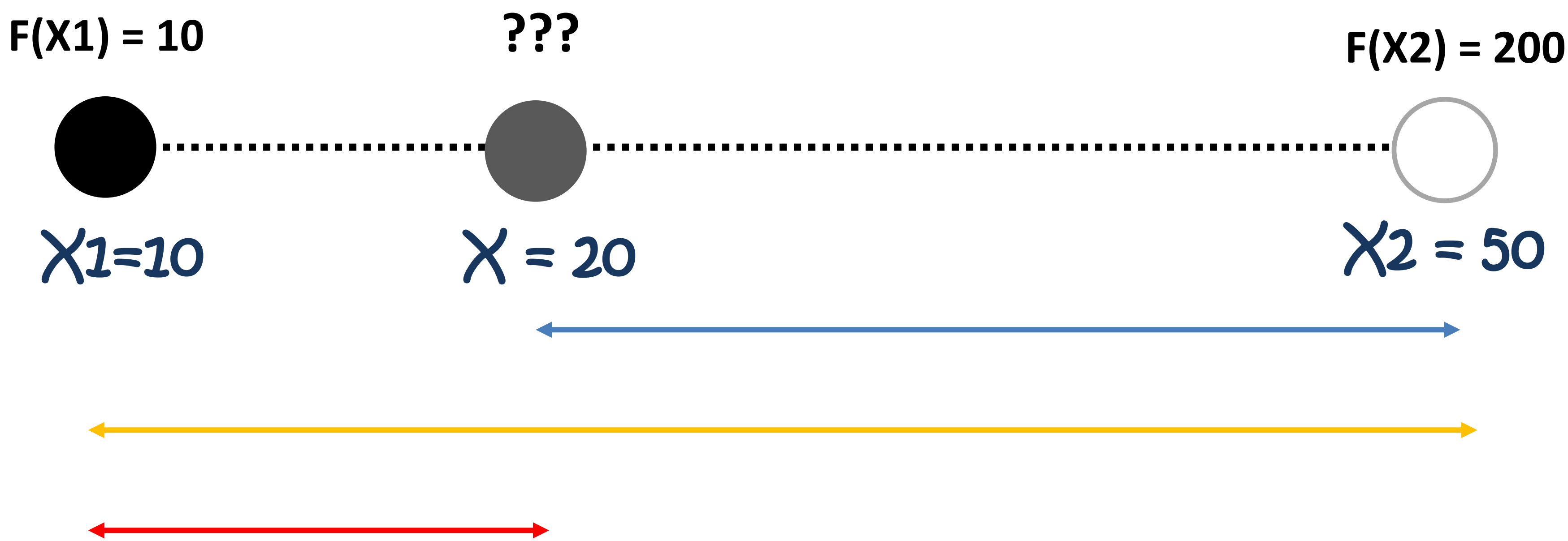
Bi-linear Interpolation



Calculate f(X) ?

Bi-linear Interpolation

Pixel value

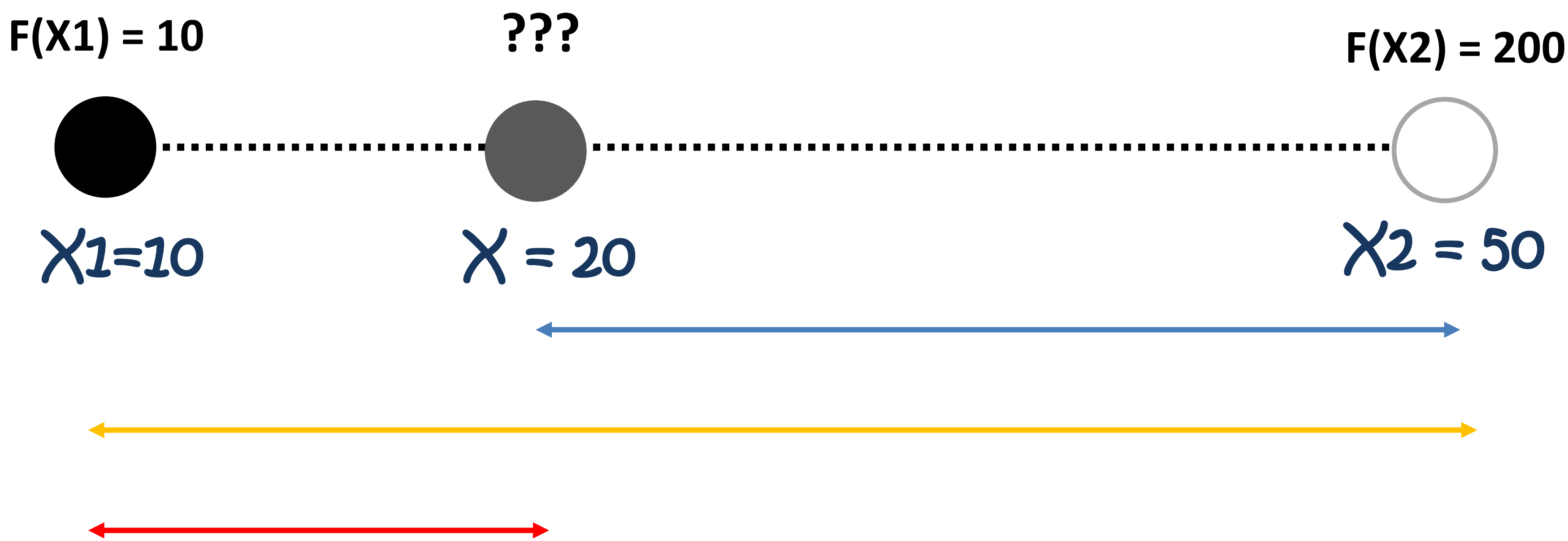


$$F(X) = \frac{X2-X}{X2-X1} * F(X1) + \frac{X-X1}{X2-X1} * F(X2)$$

$$F(X) = (30/40) * 10 + (10/40) * 200$$

Bi-linear Interpolation

Pixel value



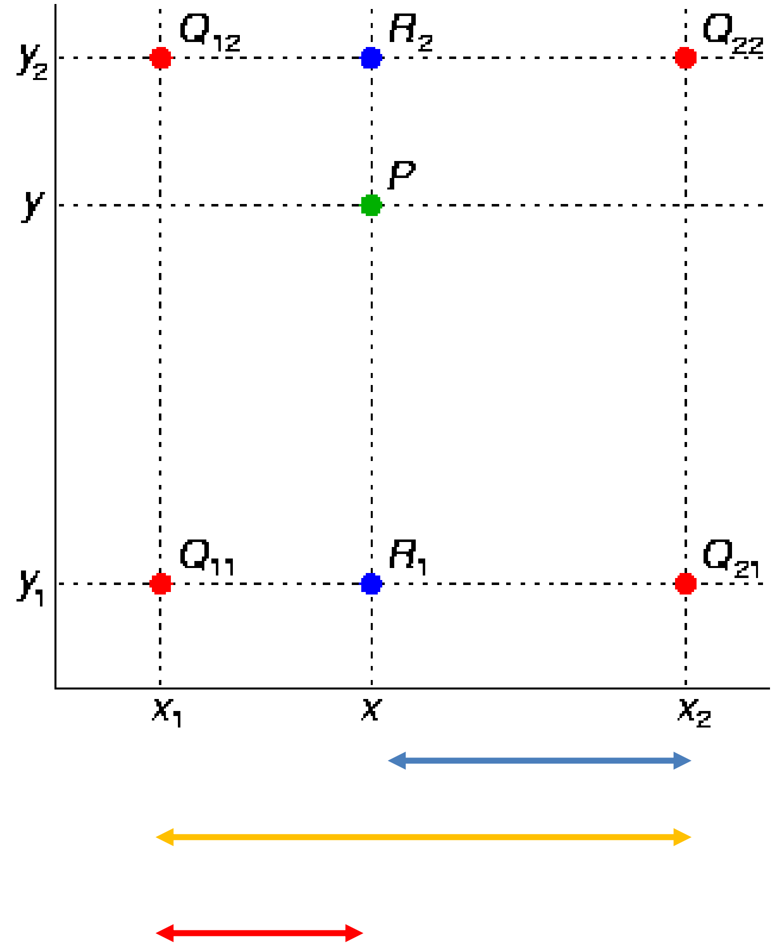
$$F(X) = (30/40) * 10 + (10/40) * 200$$

More Influence

Less Influence

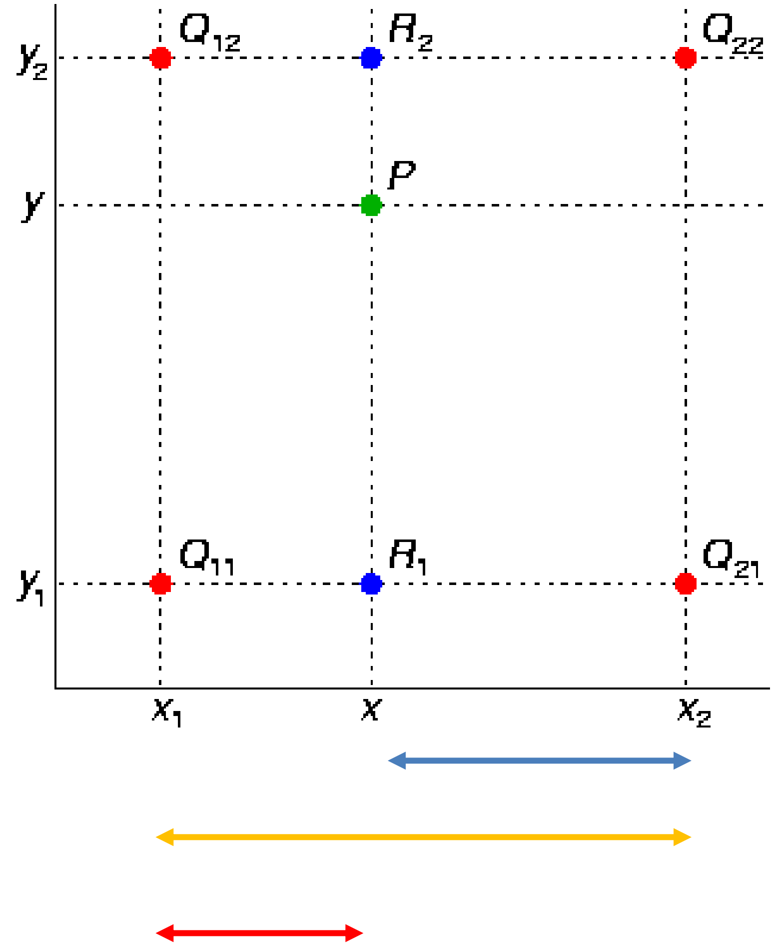
Bi-linear Interpolation

- Key Points:
  - $Q_{11} = (x_1, y_1)$, $Q_{12} = (x_1, y_2)$,
  - $Q_{21} = (x_2, y_1)$, $Q_{22} = (x_2, y_2)$.
- Linear interpolation in the x-direction followed by linear interpolation in y-direction



Bi-linear Interpolation

- Key Points:
  - $Q_{11} = (x_1, y_1)$, $Q_{12} = (x_1, y_2)$,
  - $Q_{21} = (x_2, y_1)$, $Q_{22} = (x_2, y_2)$.
- Linear interpolation in the x-direction followed by linear interpolation in y-direction



$$f(R_1) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21})$$

$$f(R_2) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22})$$

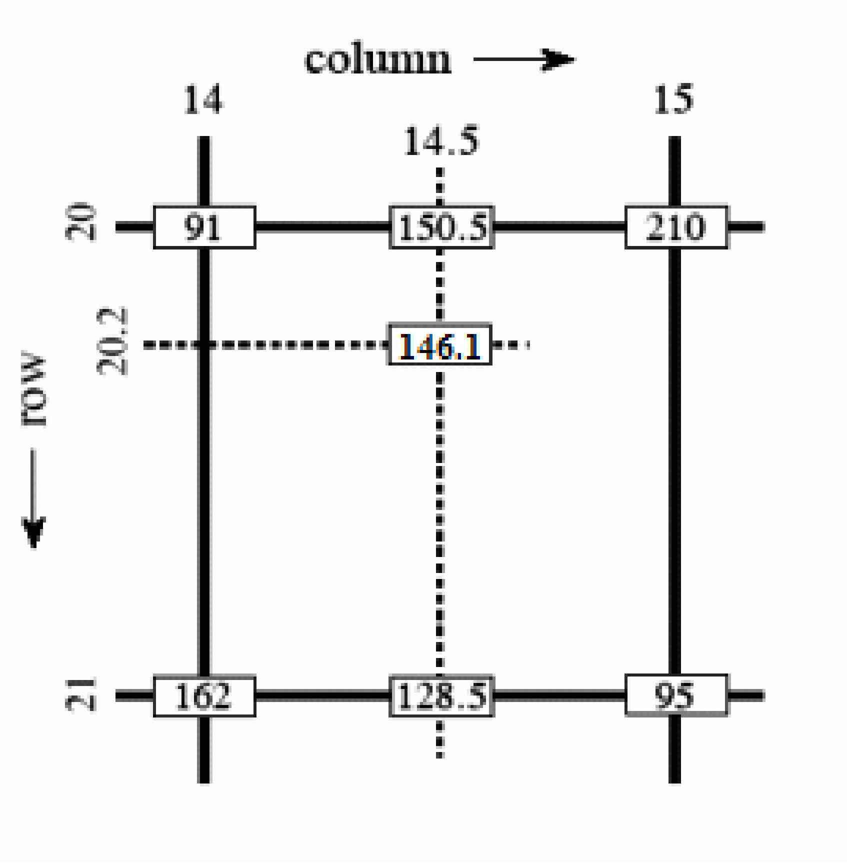
$$f(P) \approx \frac{y_2 - y}{y_2 - y_1} f(R_1) + \frac{y - y_1}{y_2 - y_1} f(R_2).$$

Bi-linear Interpolation: homework

$$f(R_1) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21})$$

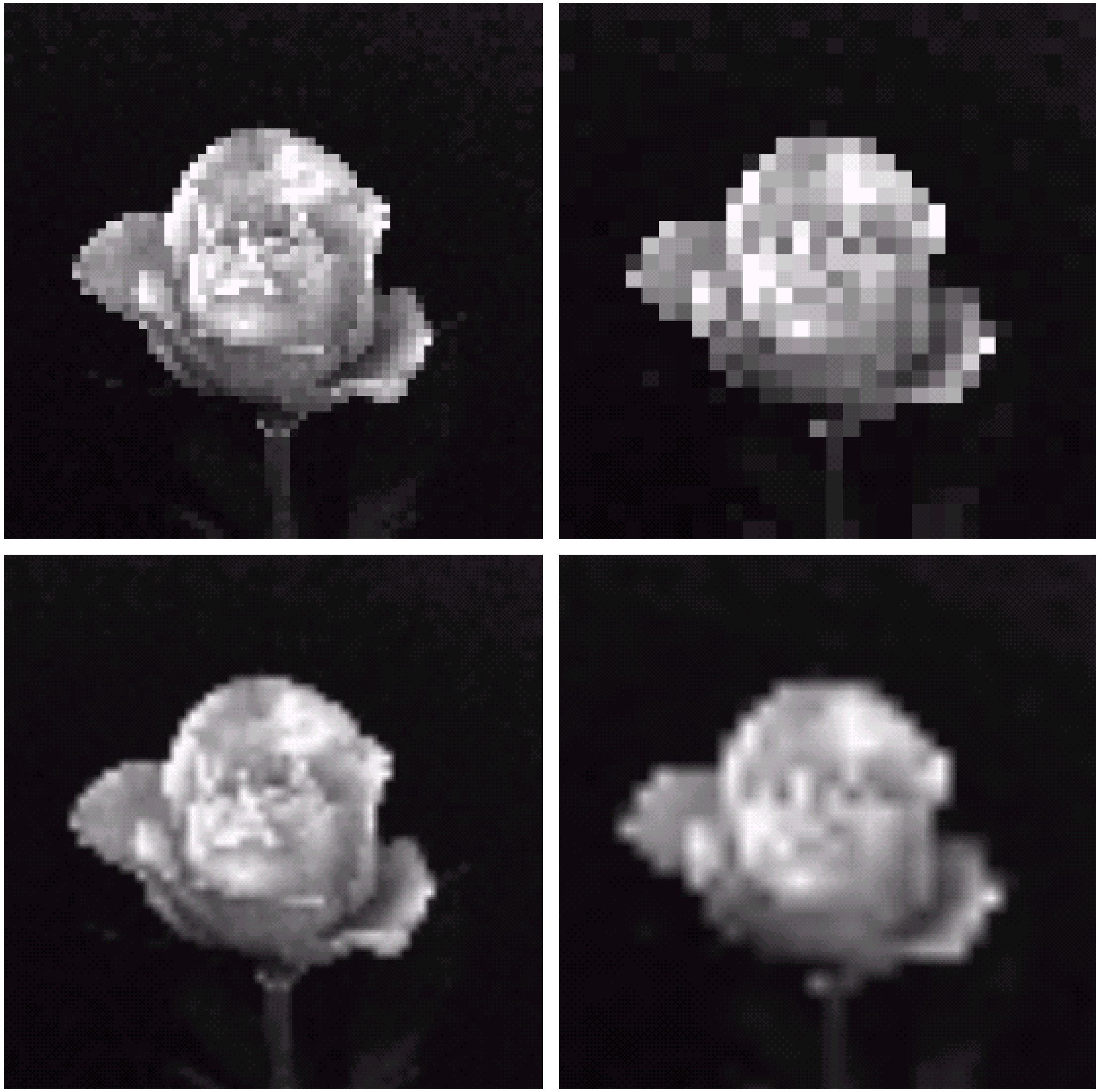
$$f(R_2) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22})$$

$$f(P) \approx \frac{y_2 - y}{y_2 - y_1} f(R_1) + \frac{y - y_1}{y_2 - y_1} f(R_2).$$



https://en.wikipedia.org/wiki/Bilinear_interpolation

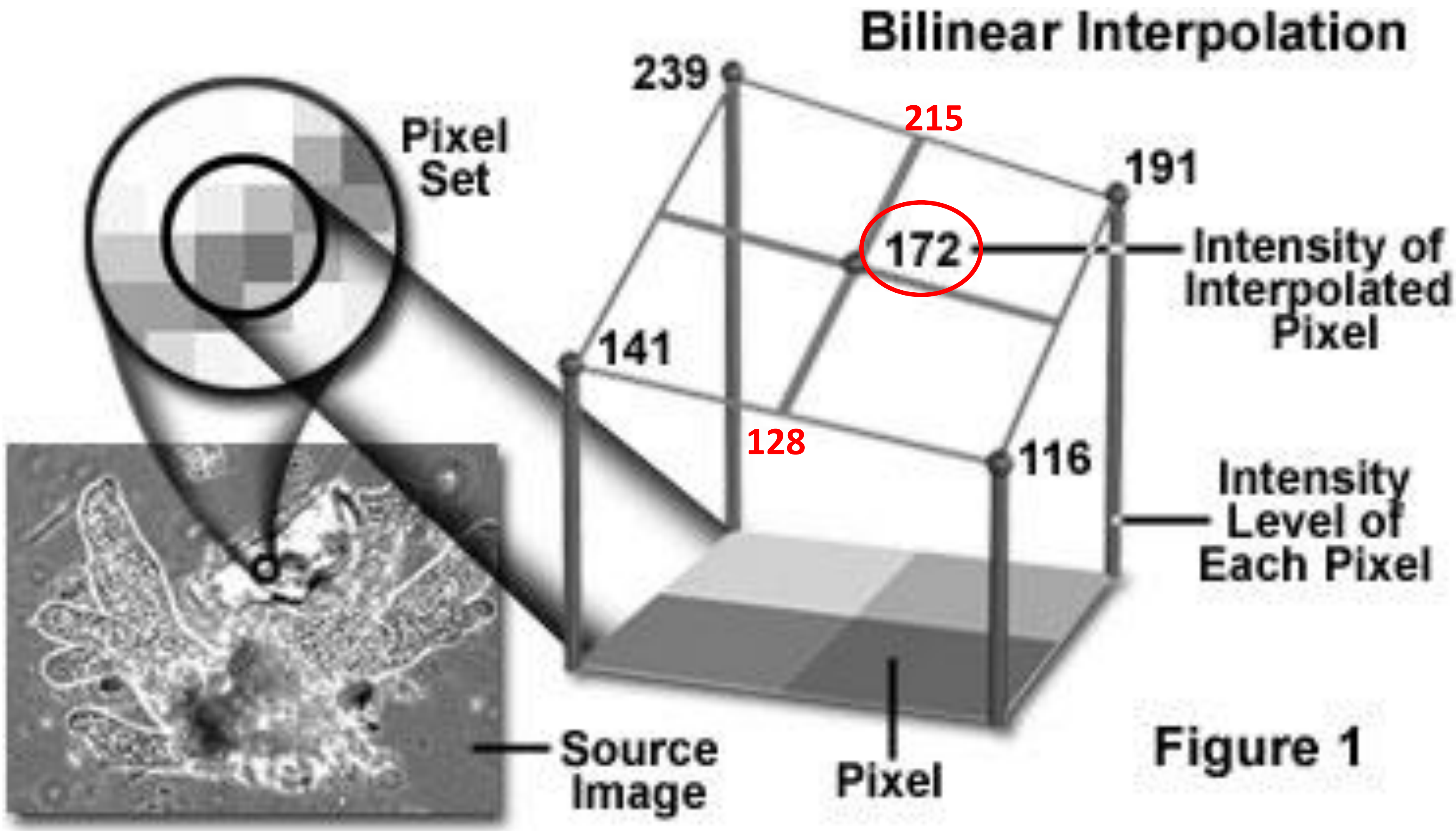
Interpolation



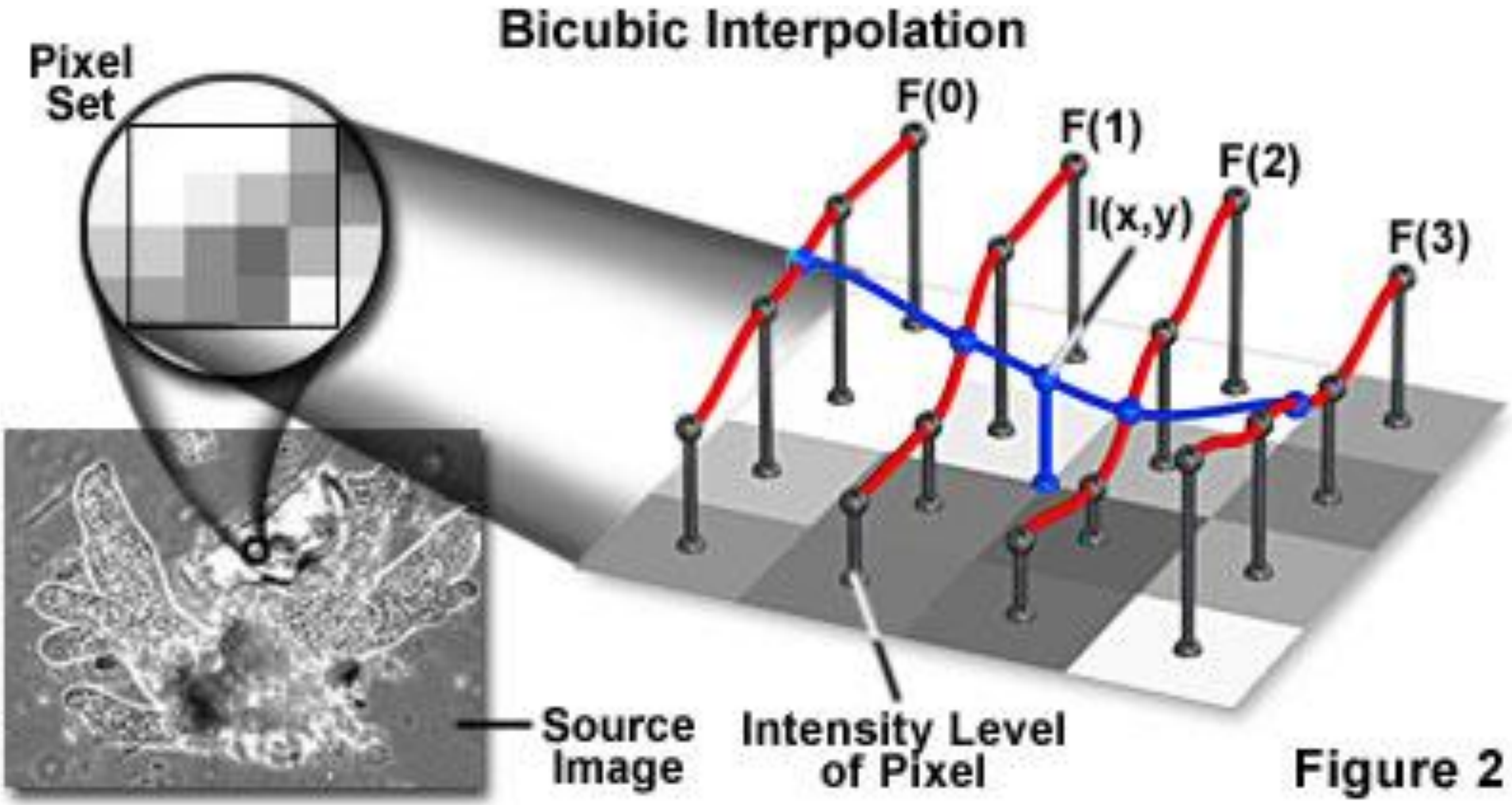
Nearest Neighbor

Bi-Linear

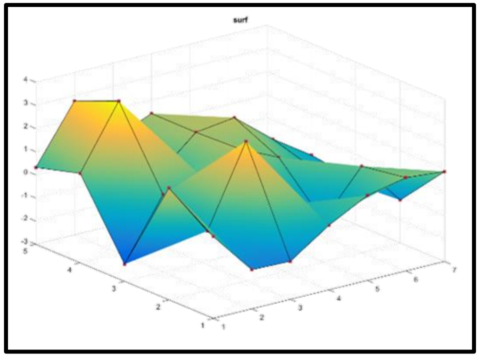
Interpolation

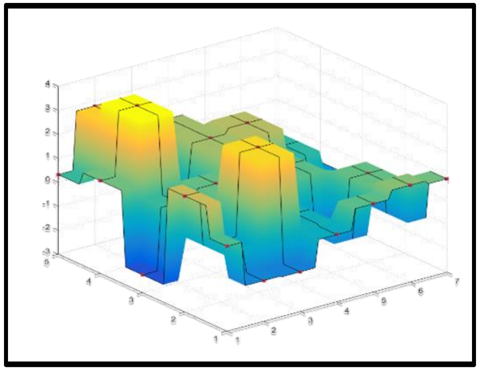


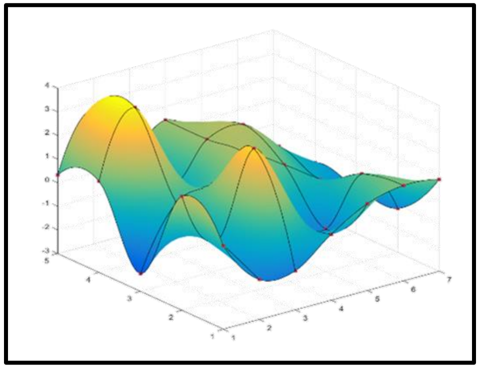
Interpolation



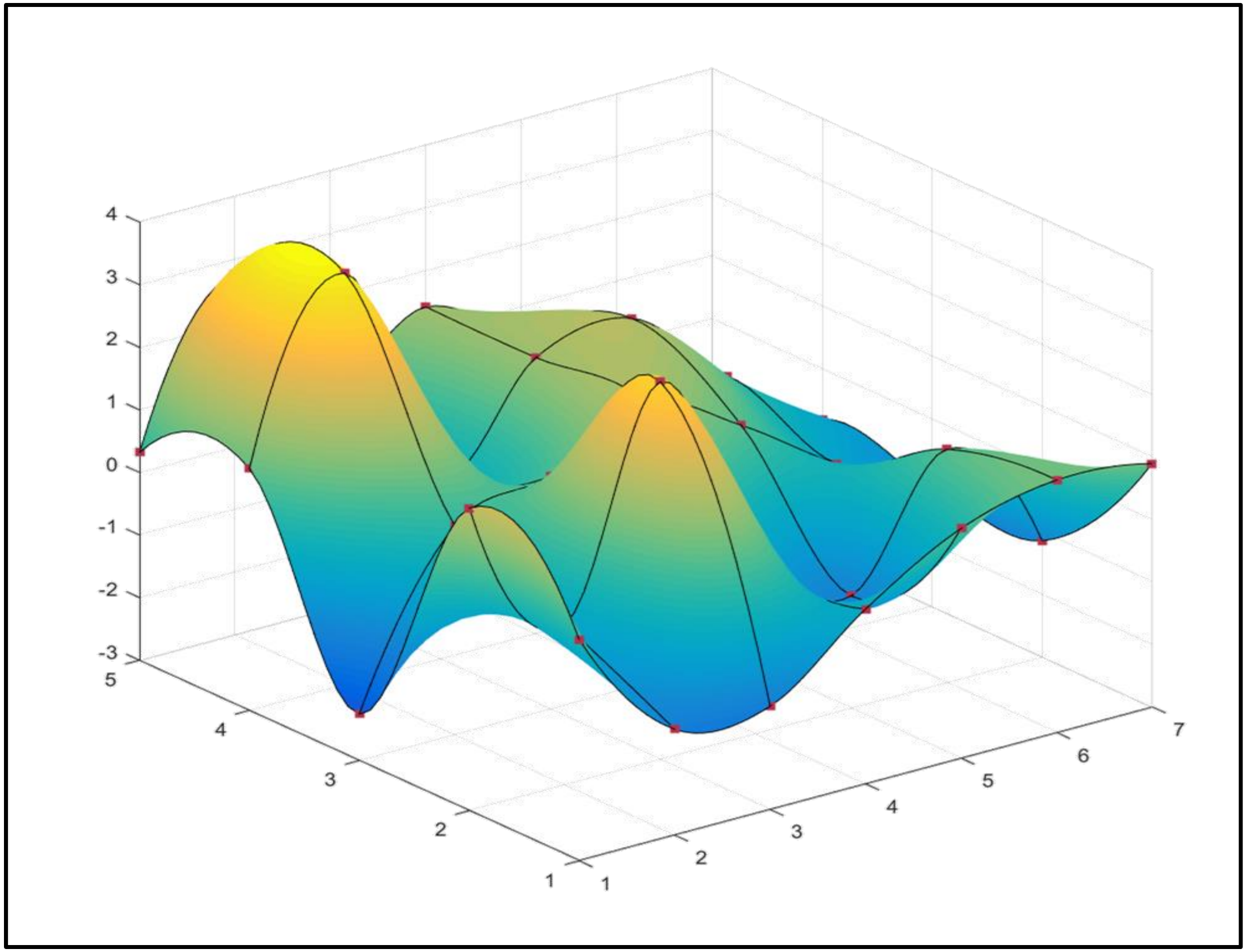
Interpolation





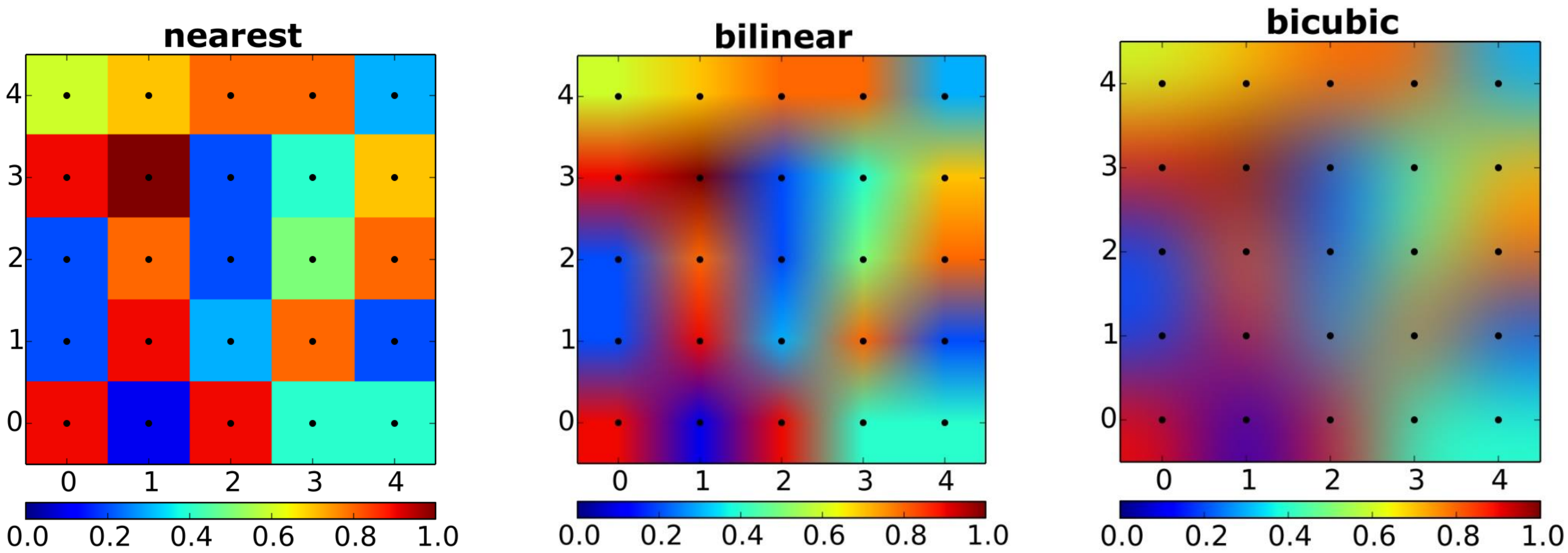


Cubic Interpolation

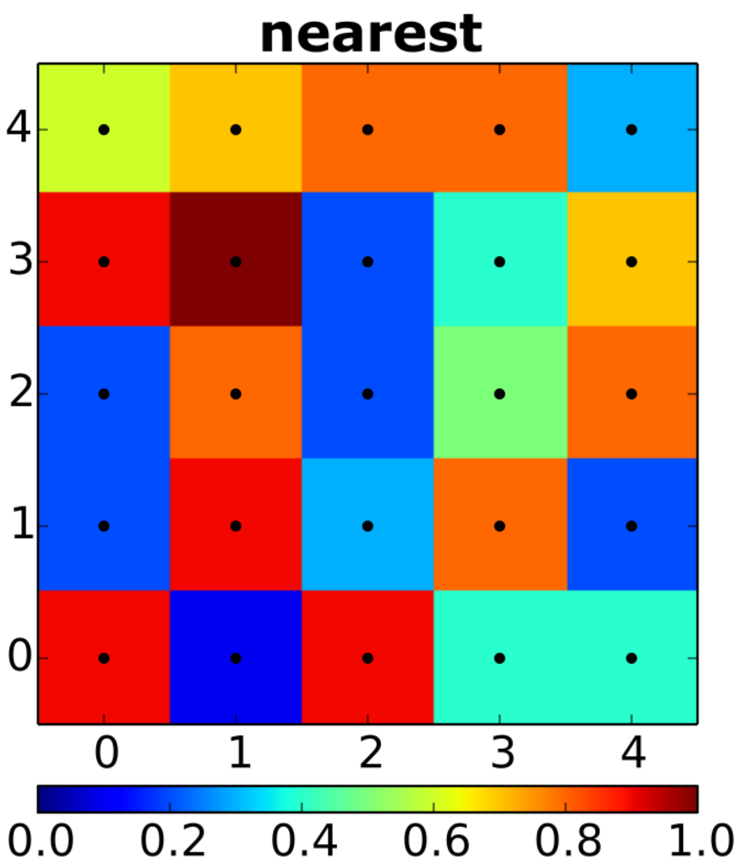


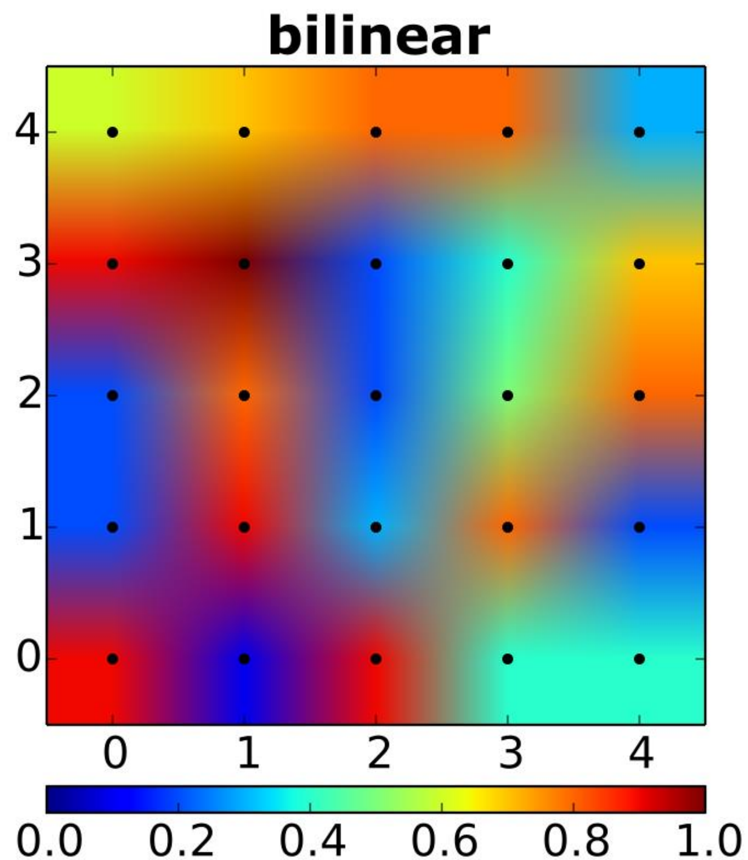
Khawar Khurshid

Interpolation

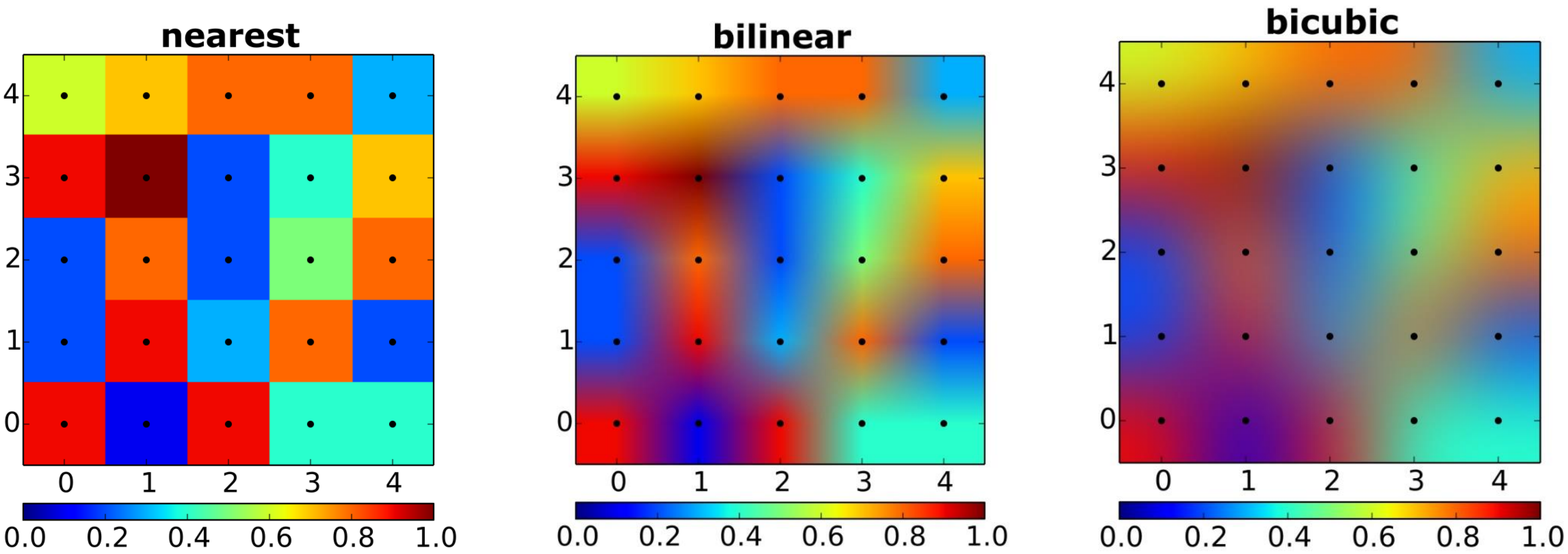


Interpolation

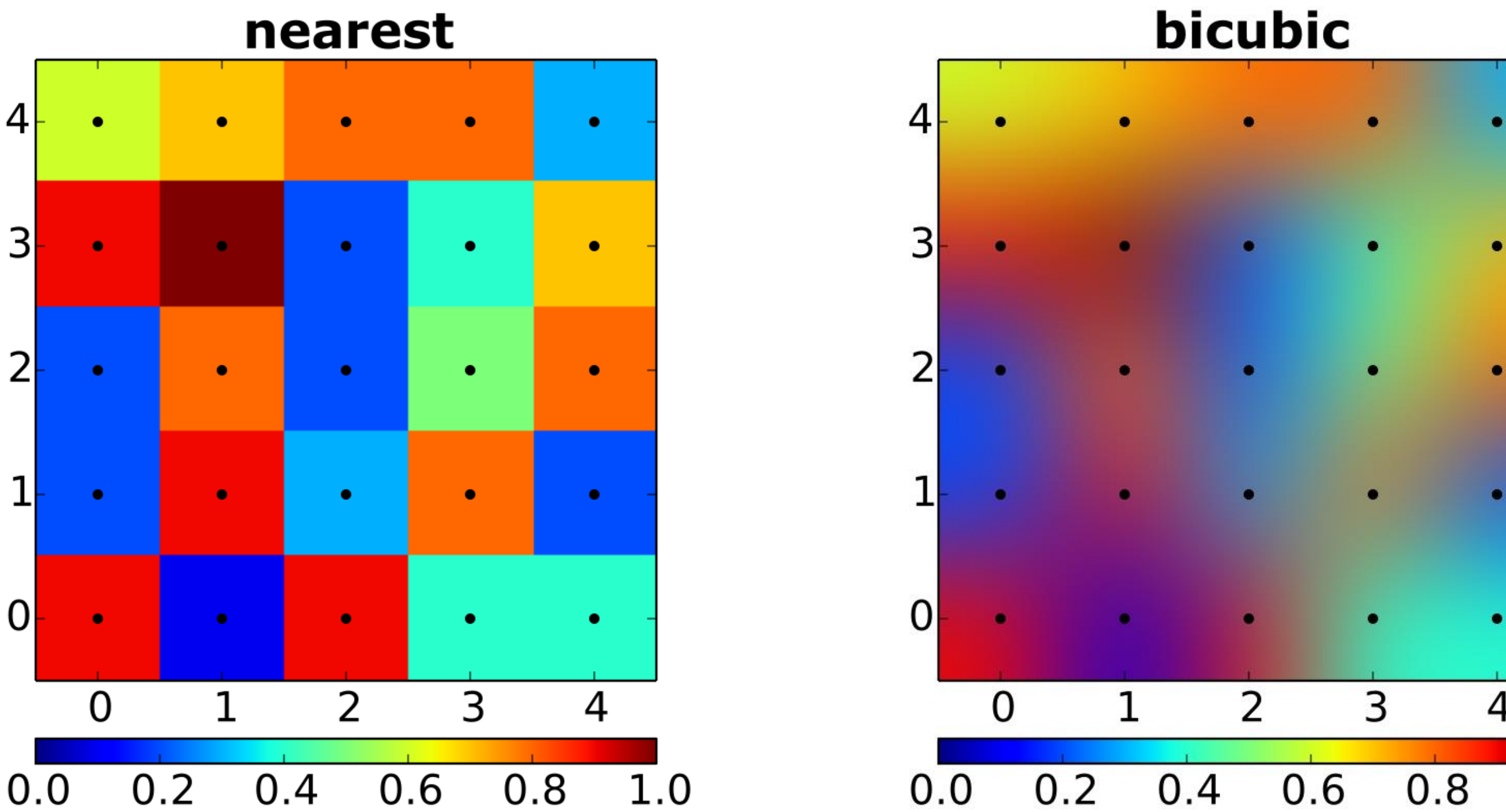




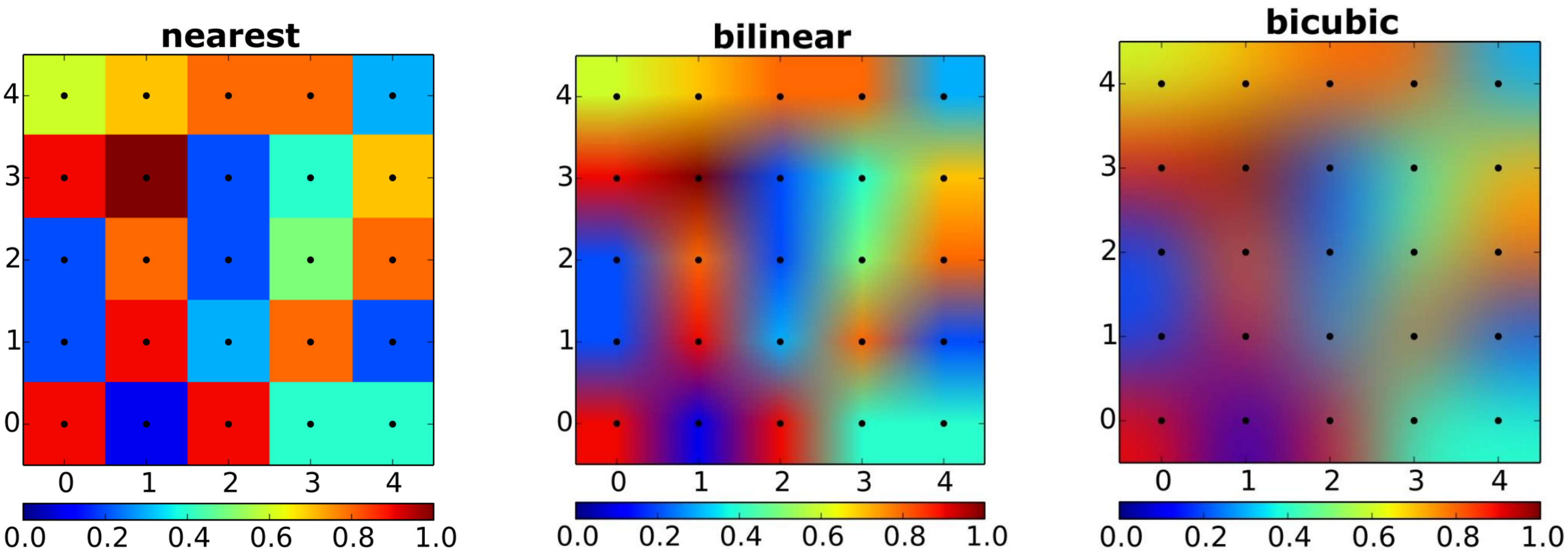
Interpolation



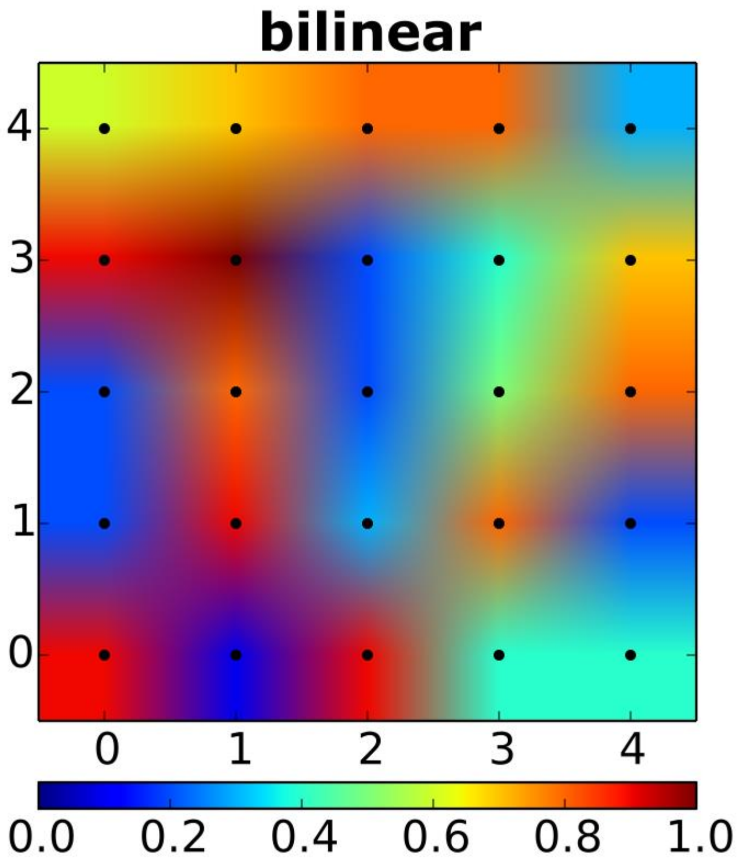
Interpolation

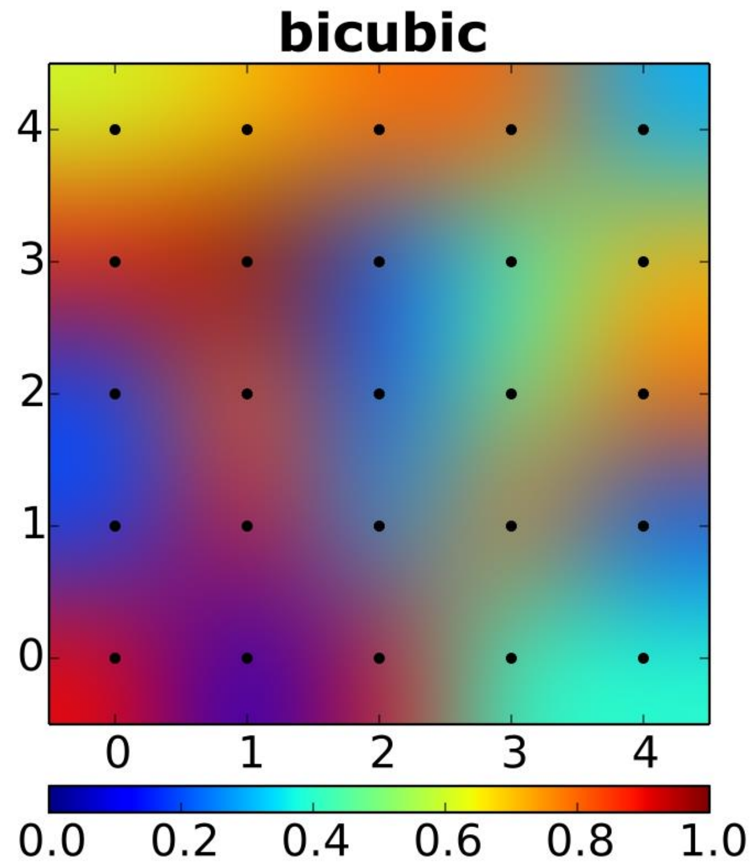


Interpolation



Interpolation





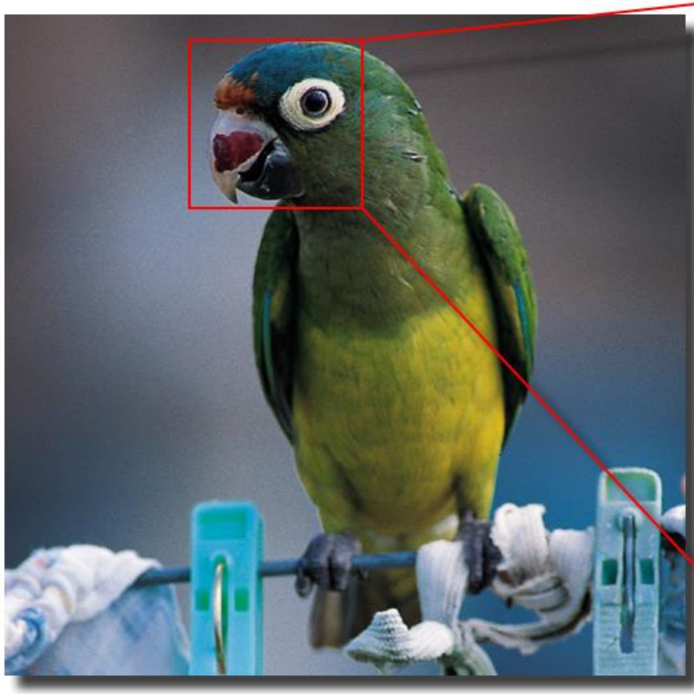
Interpolation – (Comparison)

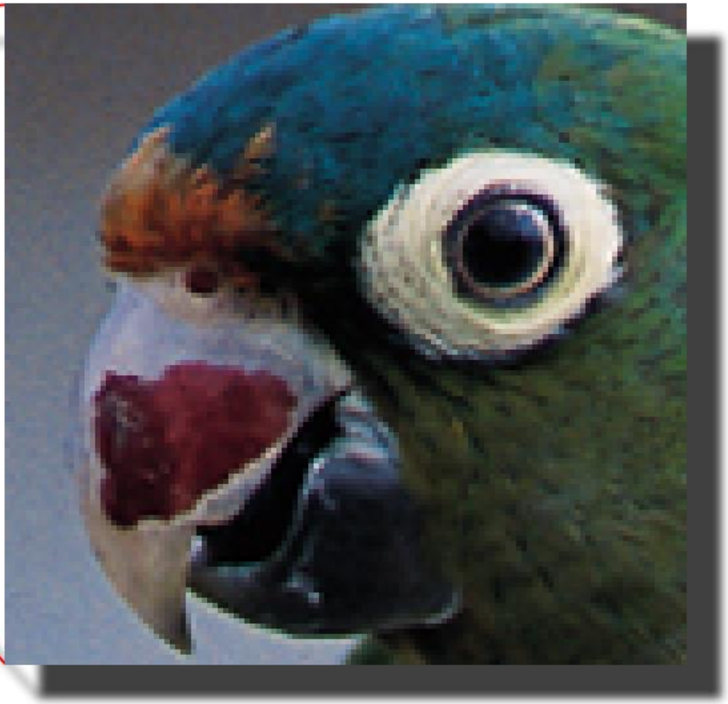
We'll enlarge this image by a factor of 4 …

… via bilinear interpolation and compare it to a nearest neighbor enlargement.

Interpolation – (Comparison)

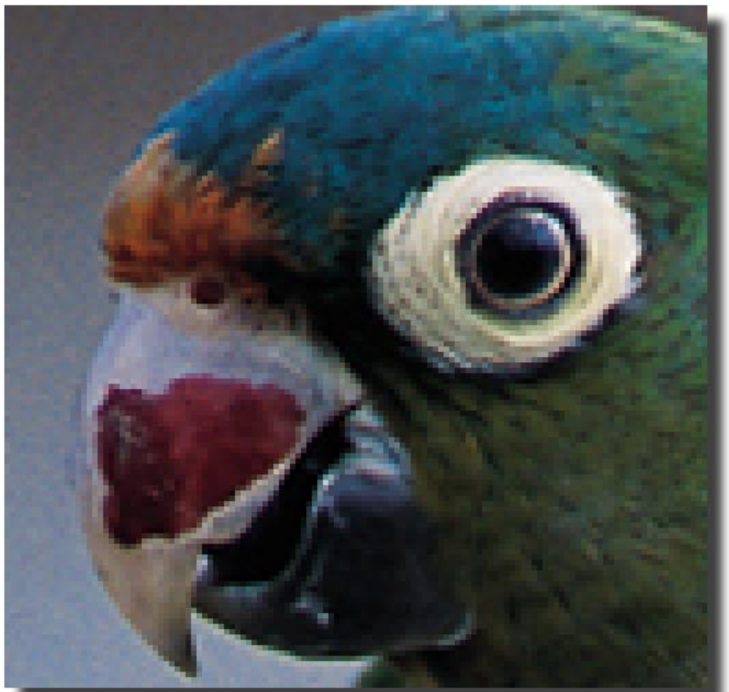
Original Image





To better see what happens, we'll look at the parrot's eye.

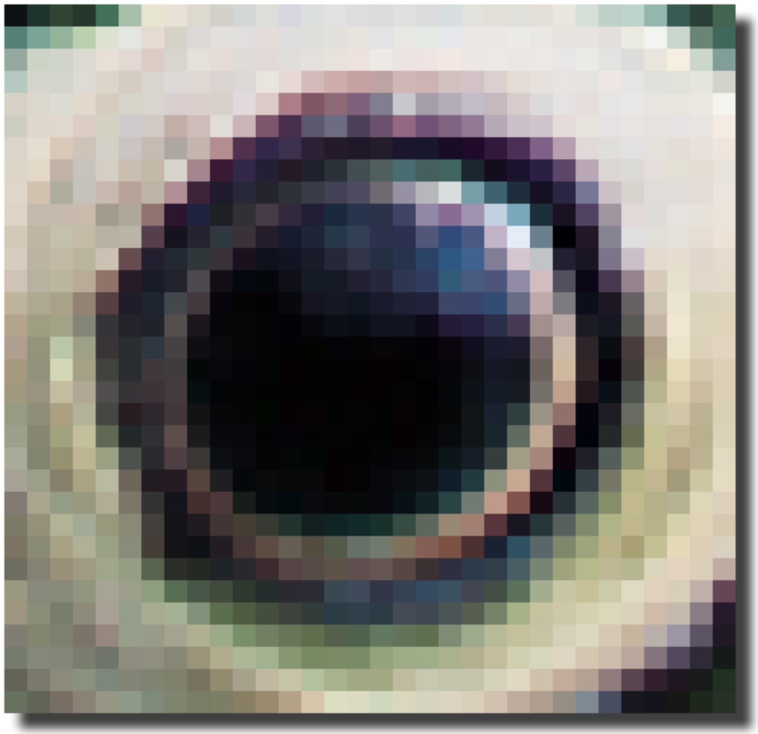
Interpolation – (Comparison)



Pixel replication

Bilinear interpolation

Interpolation – (Comparison)

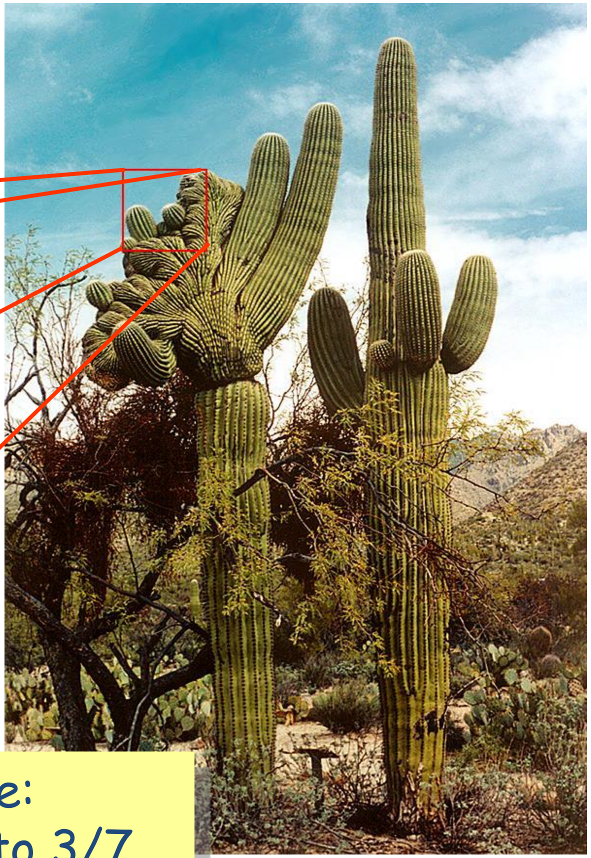


Pixel replication

Bilinear interpolation

Interpolation – (Non Integer)

Zoom in on a section for a closer look at the process

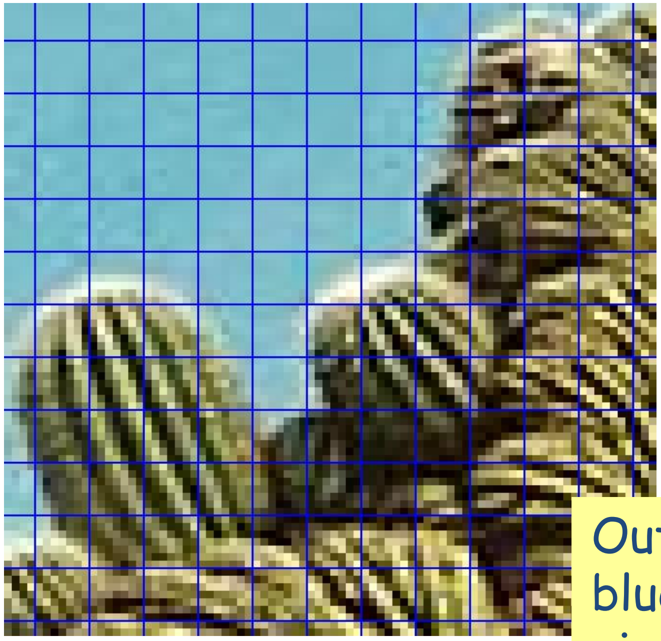


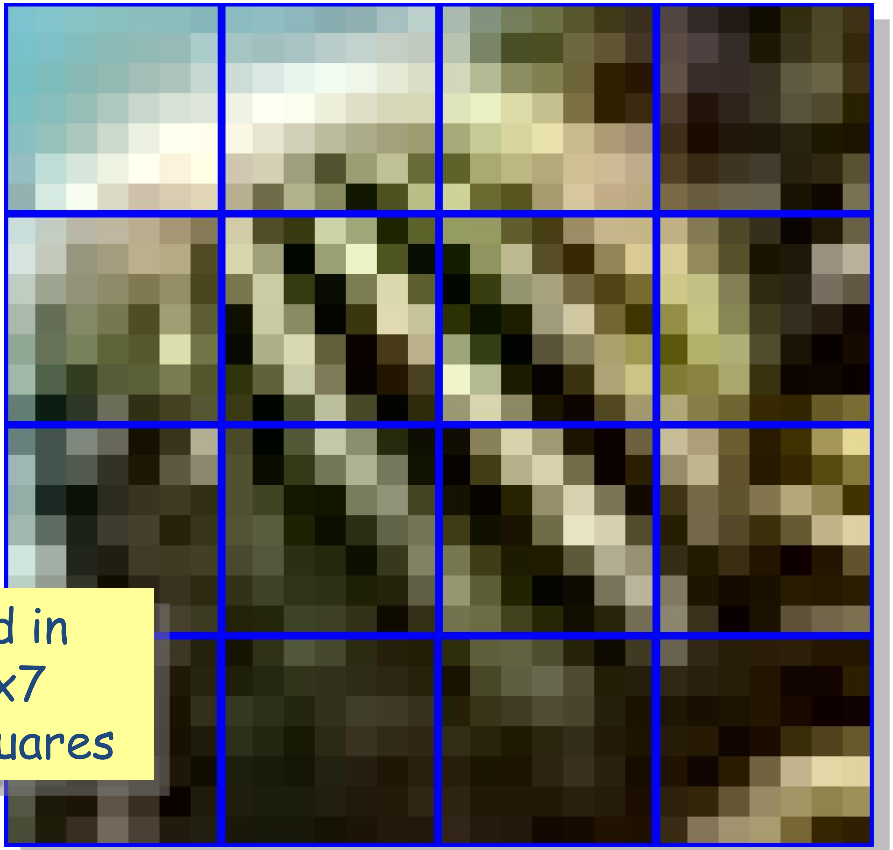
Example: resize to 3/7 of the original

Interpolation – (Non Integer)

Interpolation – (Non Integer)

3/7 resize

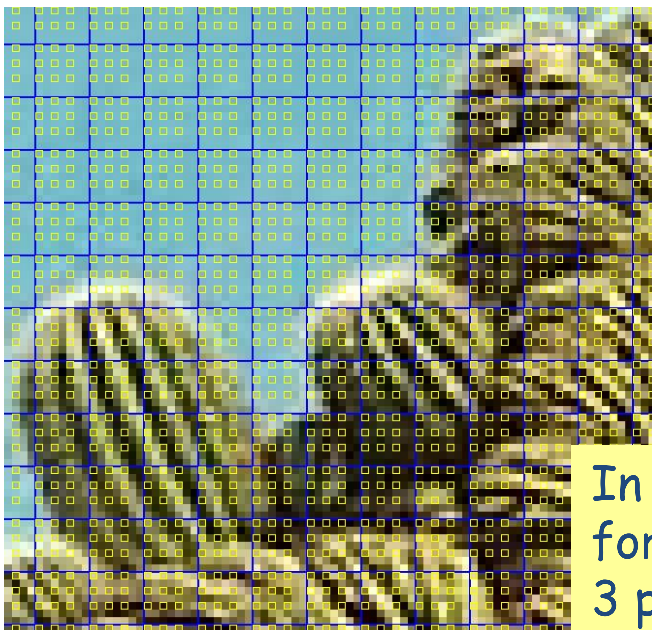


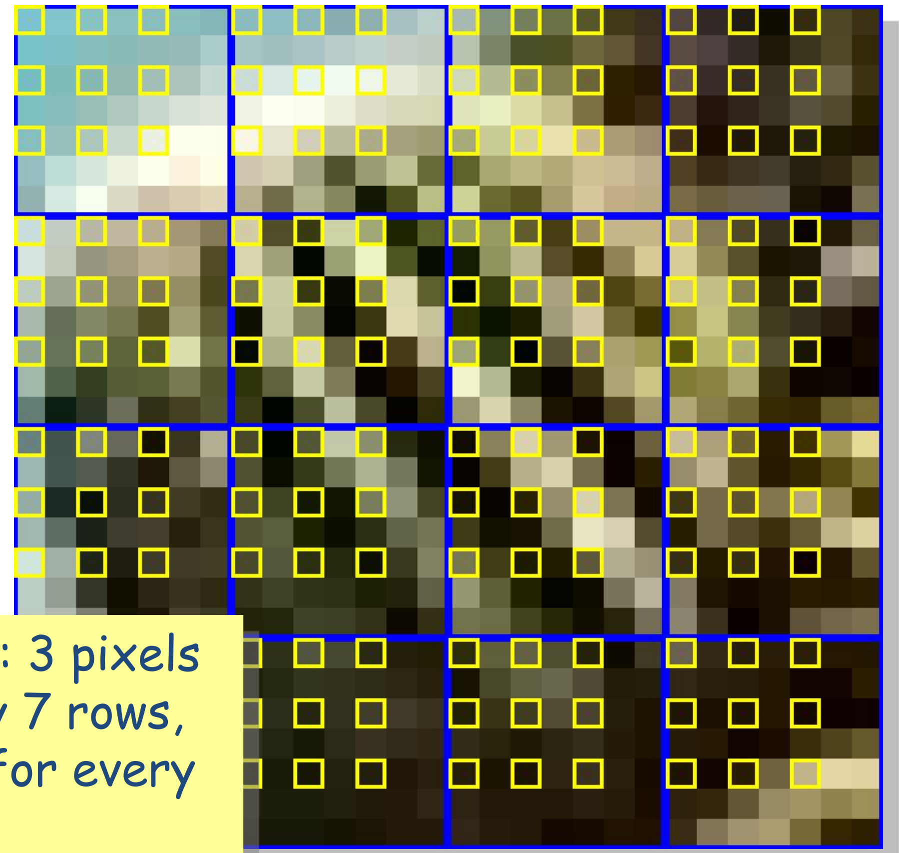


Outlined in blue: 7x7 pixel squares

Interpolation – (Non Integer)

3/7 resize

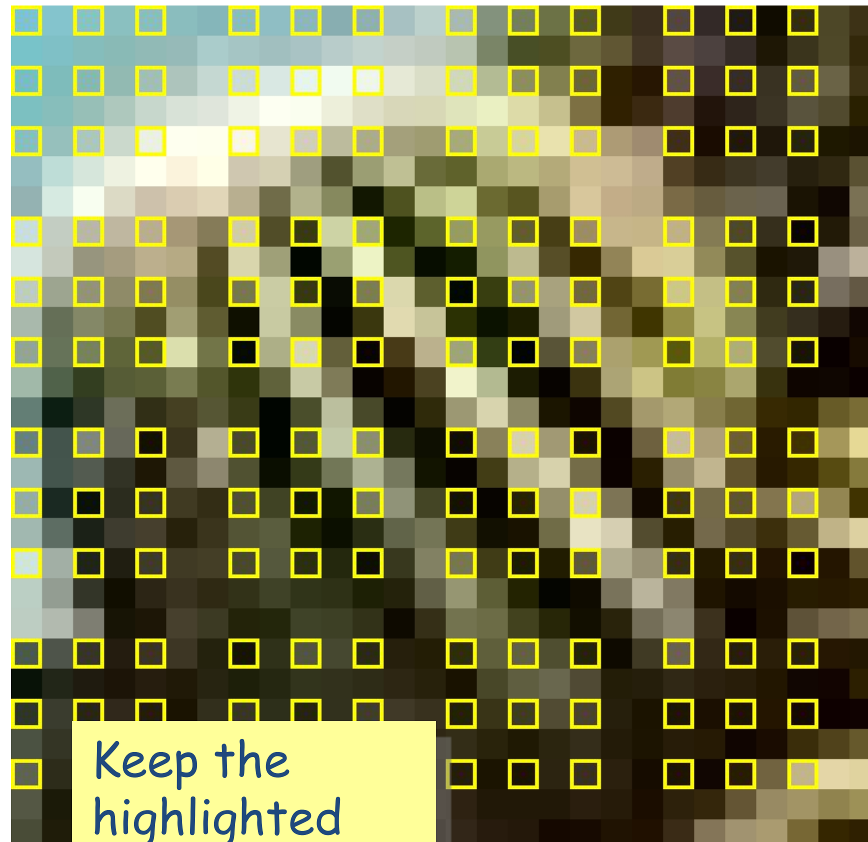




In yellow: 3 pixels for every 7 rows, 3 pixels for every 7 cols.

Interpolation – (Non Integer)

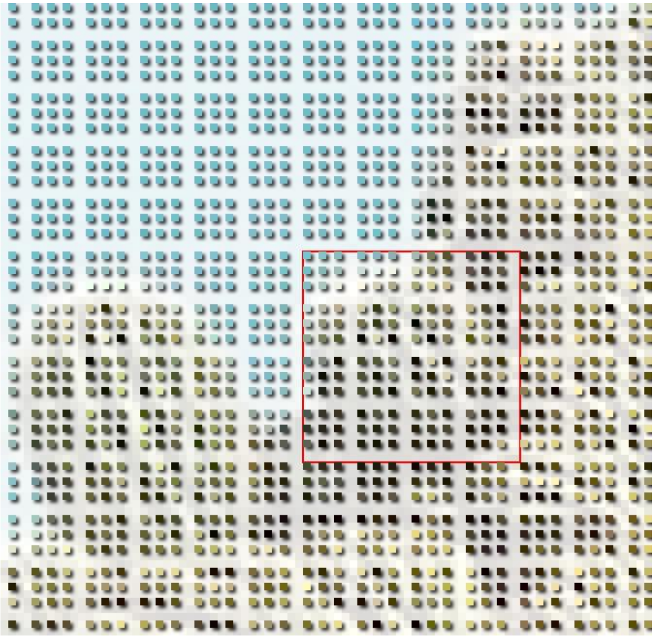
3/7 resize



Keep the highlighted pixels…

Interpolation – (Non Integer)

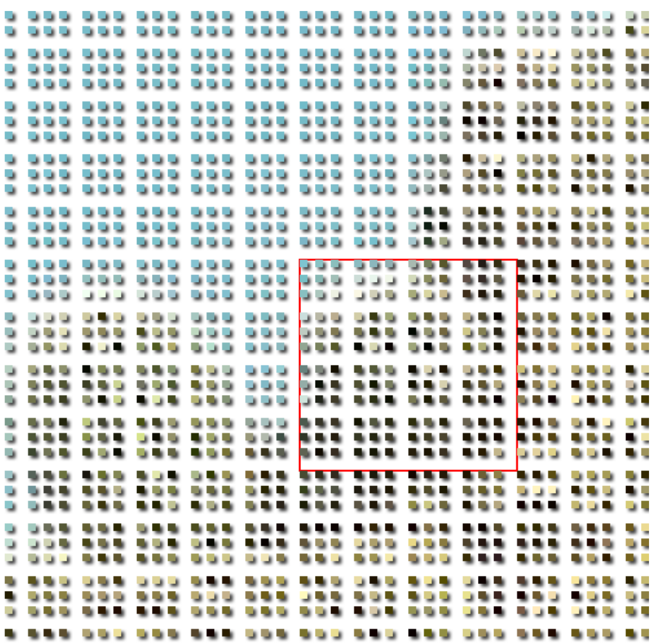
3/7 resize

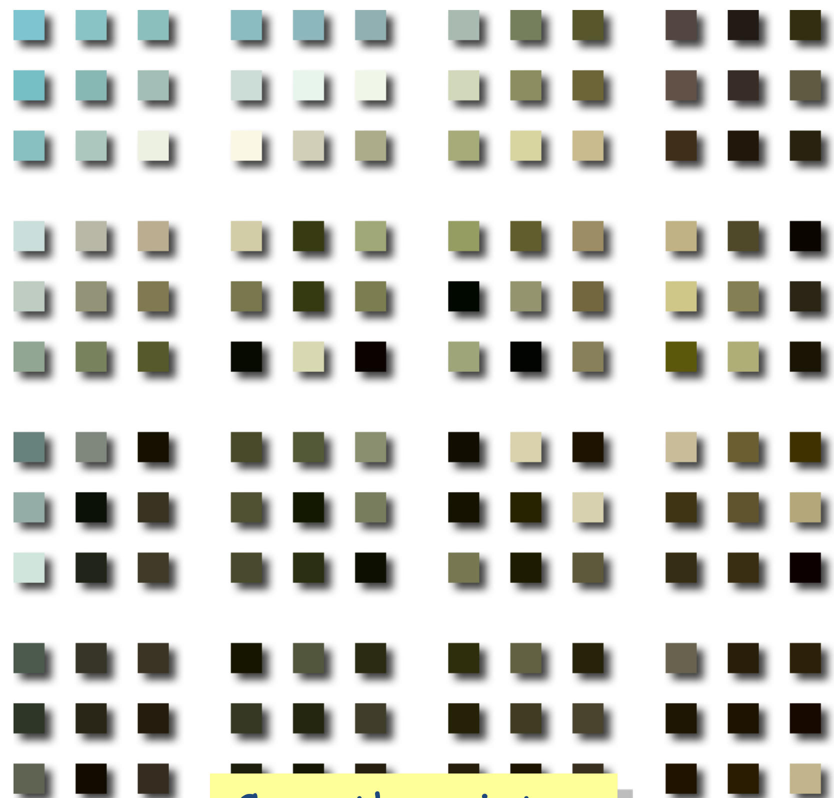


… don't keep the others.

Interpolation – (Non Integer)

3/7 resize

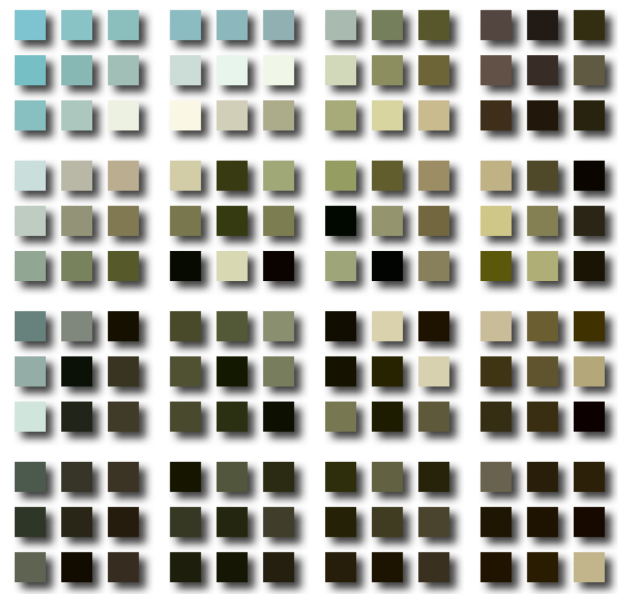




Copy them into a new image.

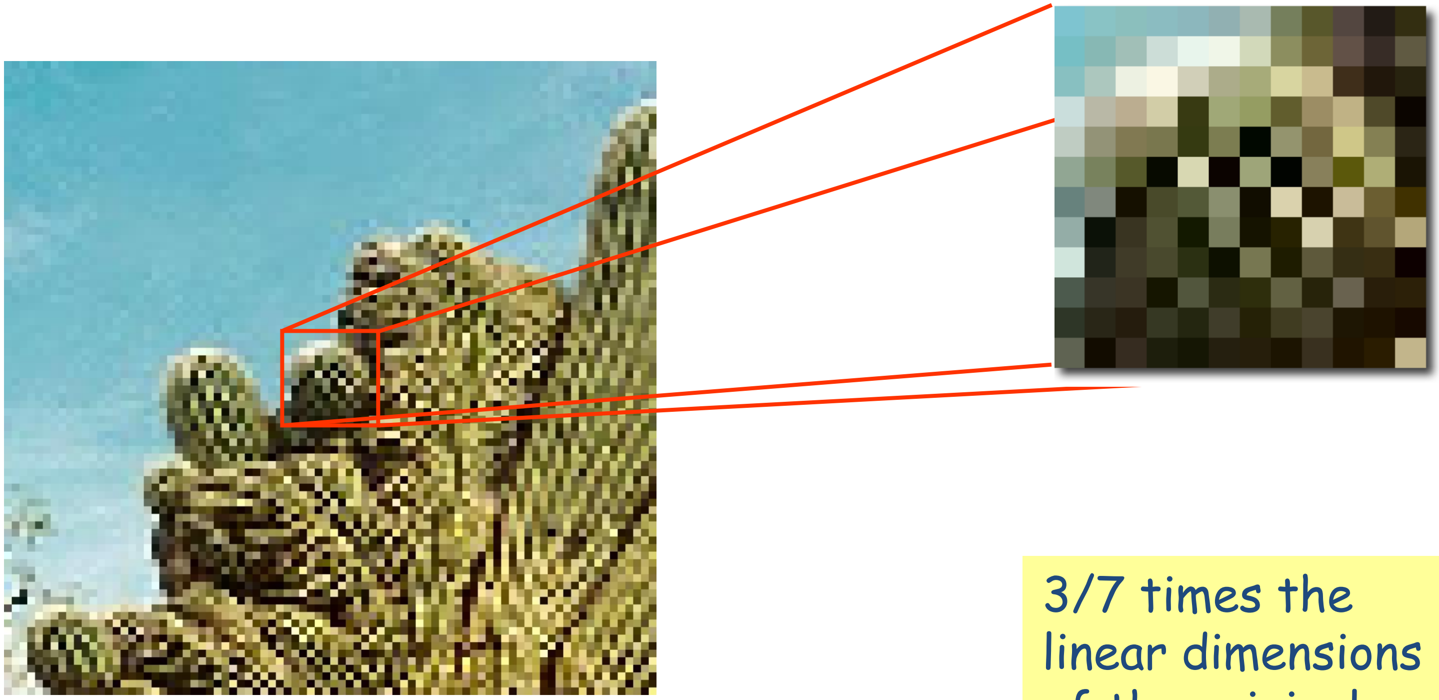
Interpolation – (Non Integer)

3/7 resize



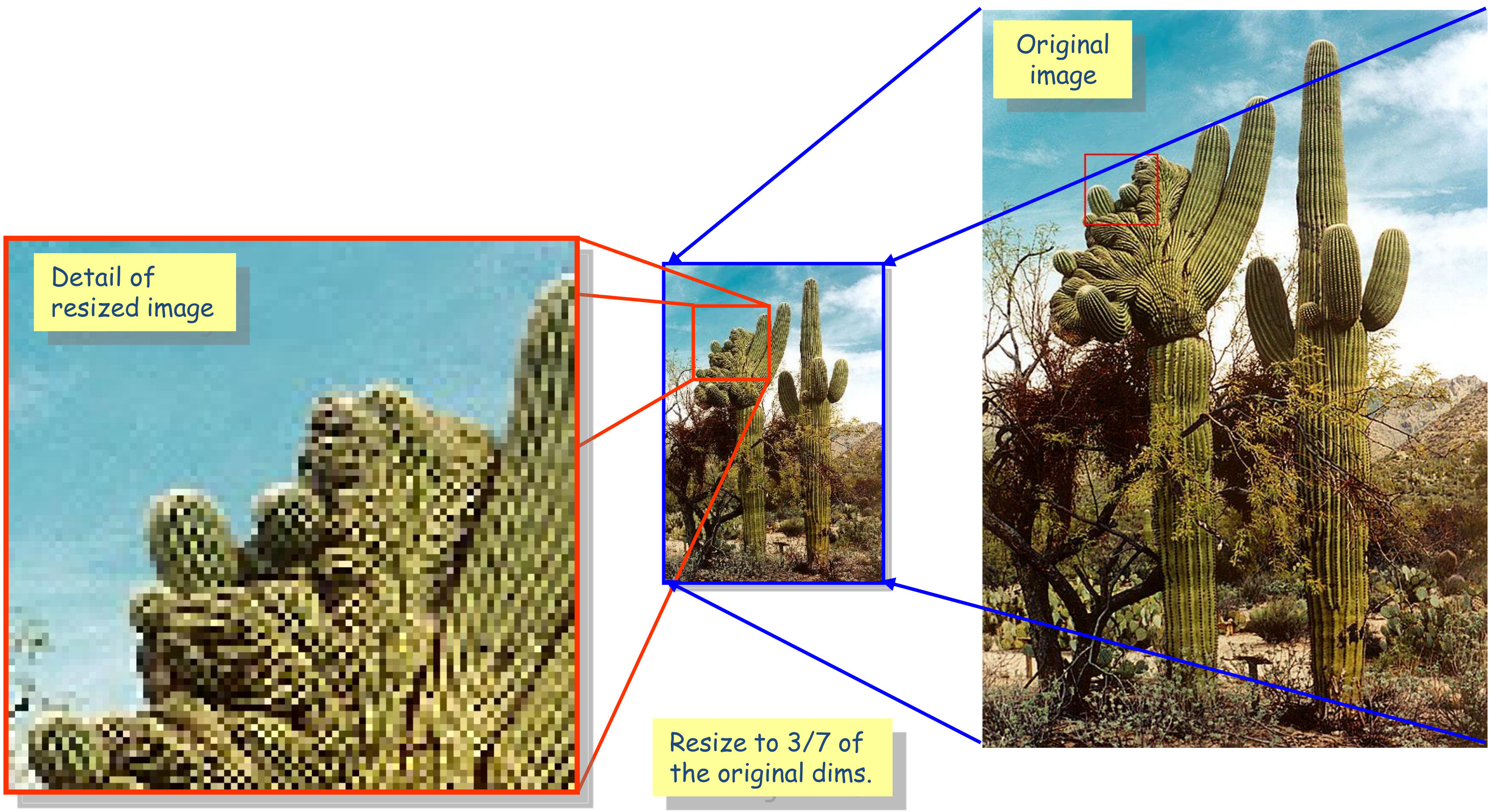
Copy them into a new image.

Interpolation – (Non Integer)



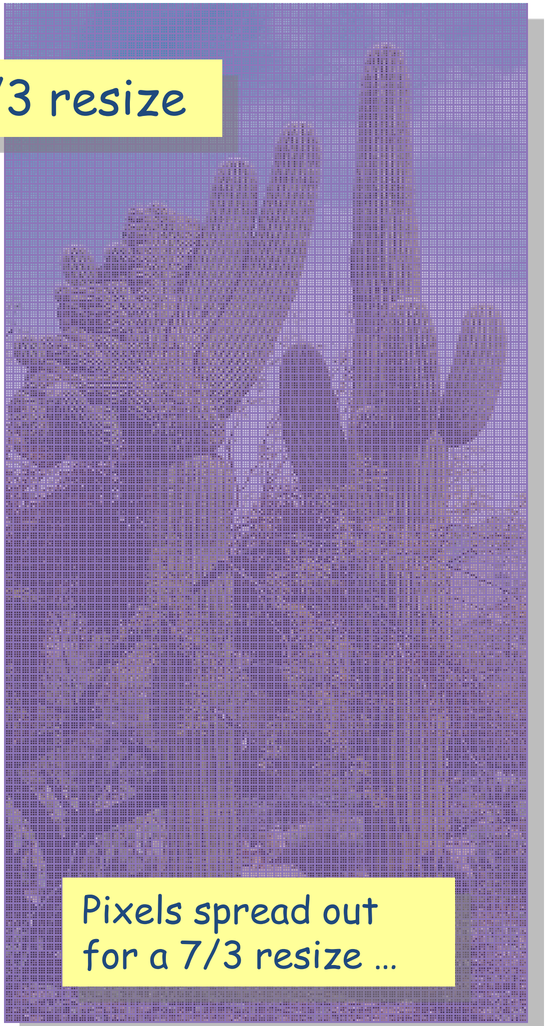
3/7 times the linear dimensions of the original

Interpolation – (Non Integer)



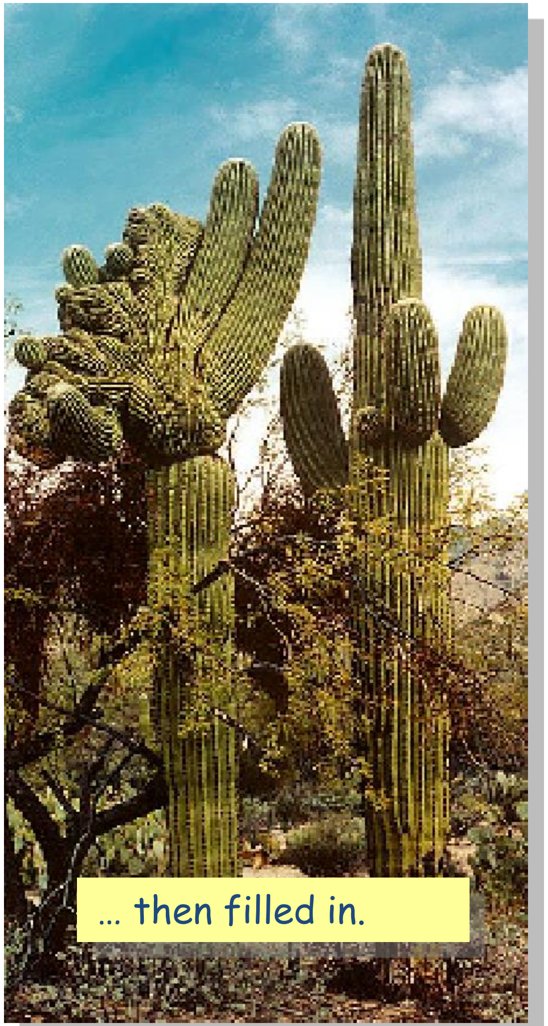
Interpolation – (Non Integer)

Original image



7/3 resize

Pixels spread out for a 7/3 resize …



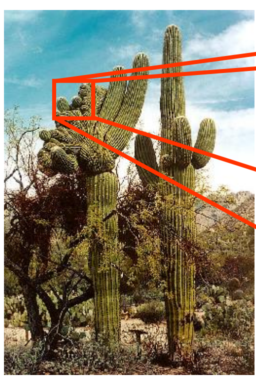
… then filled in.

Interpolation – (Non Integer)

7/3 resize

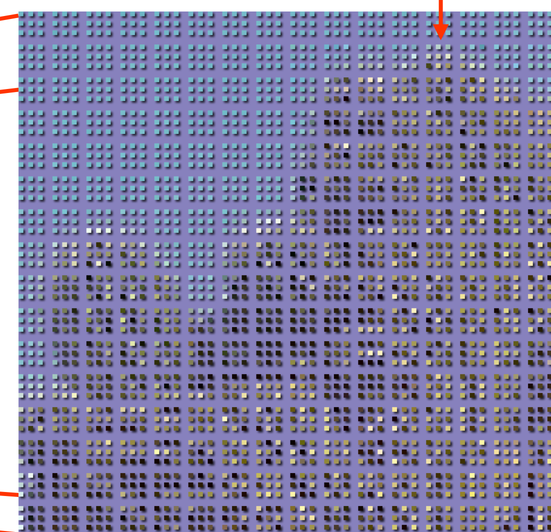
Each 3x3 block of pixels from here …

… is spread out over a 7x7 block here



Original image

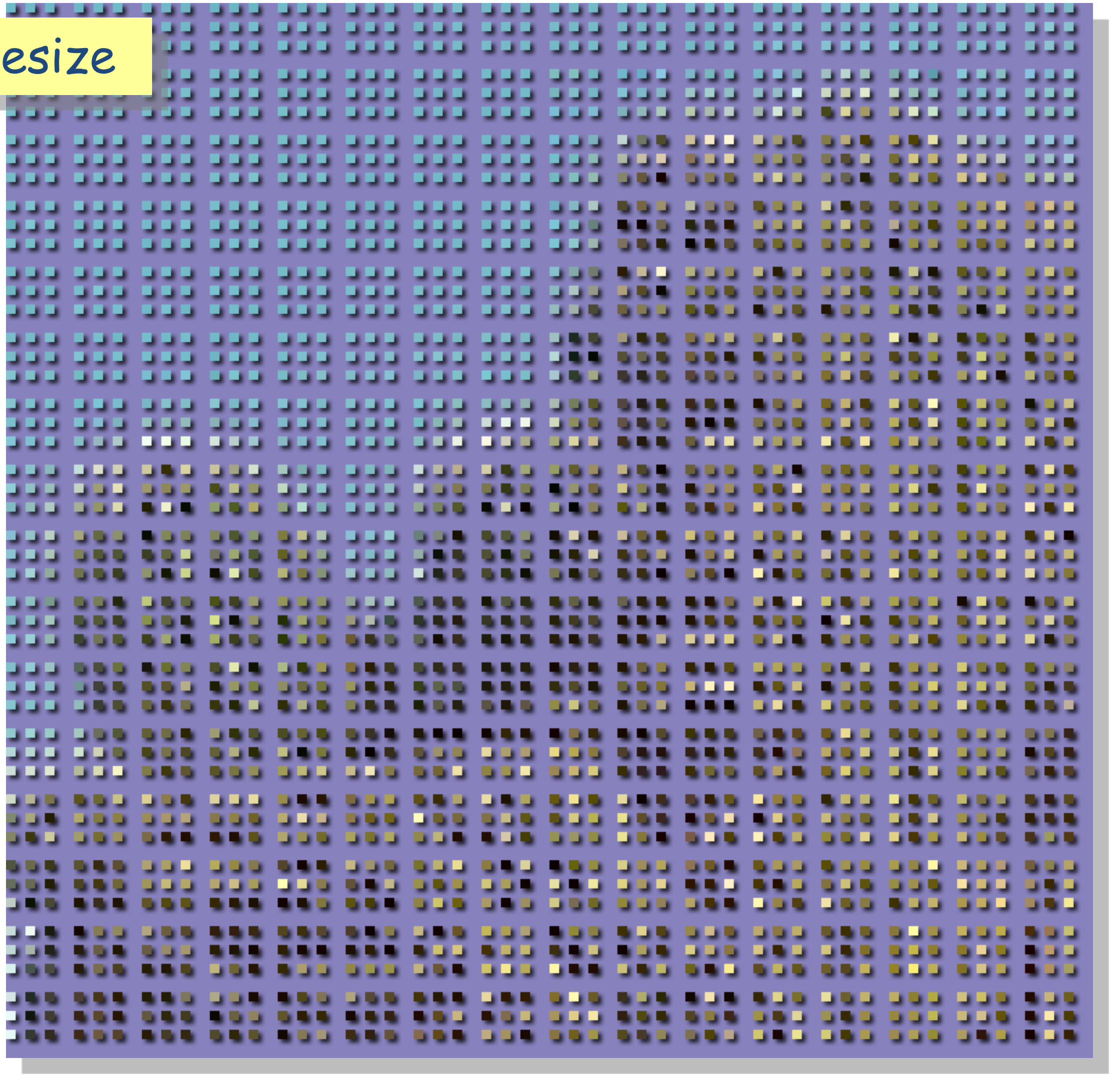
Detail



Interpolation – (Non Integer)

7/3 resize

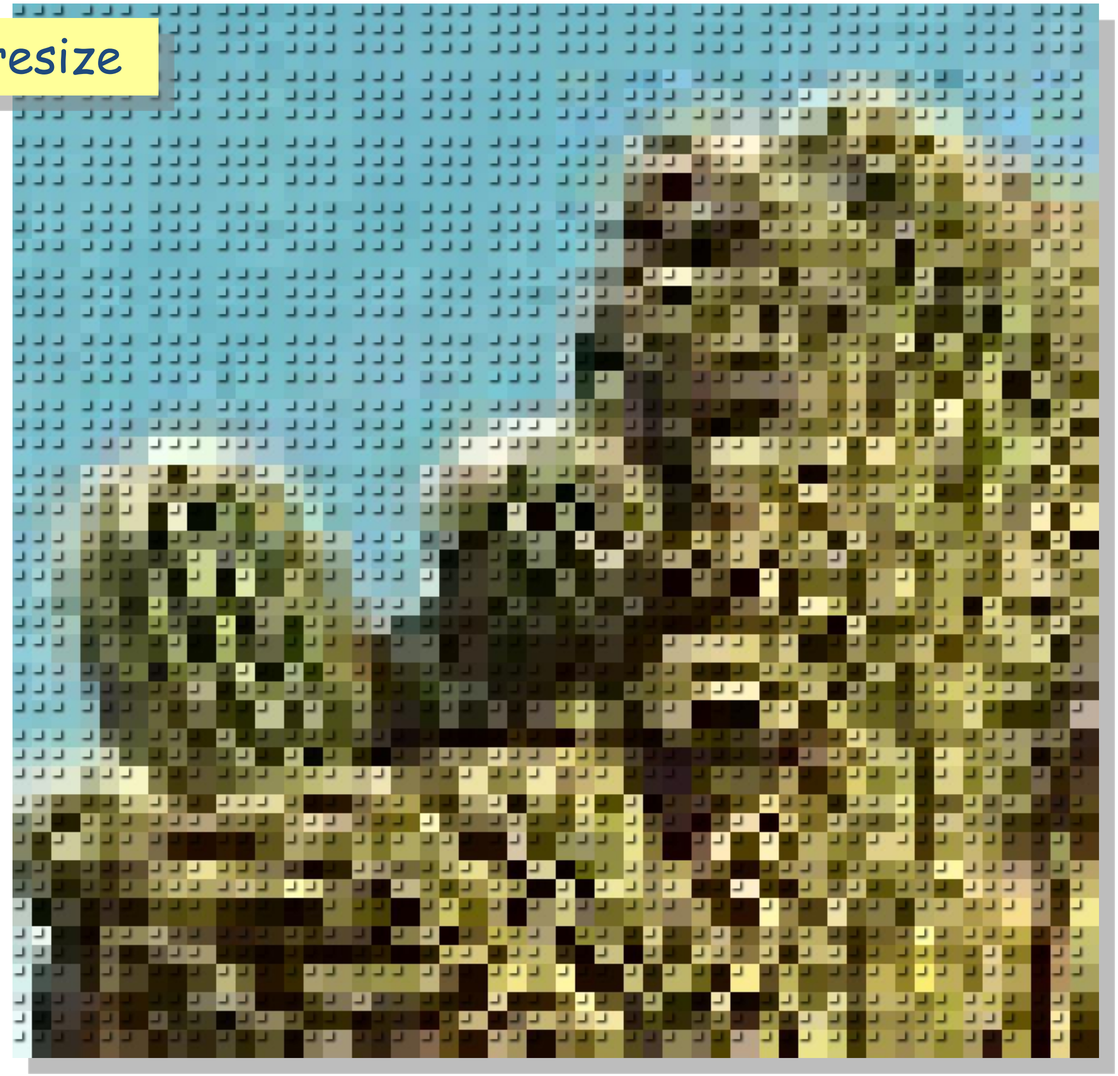
3x3 blocks distributed over 7x7 blocks



Interpolation – (Non Integer)

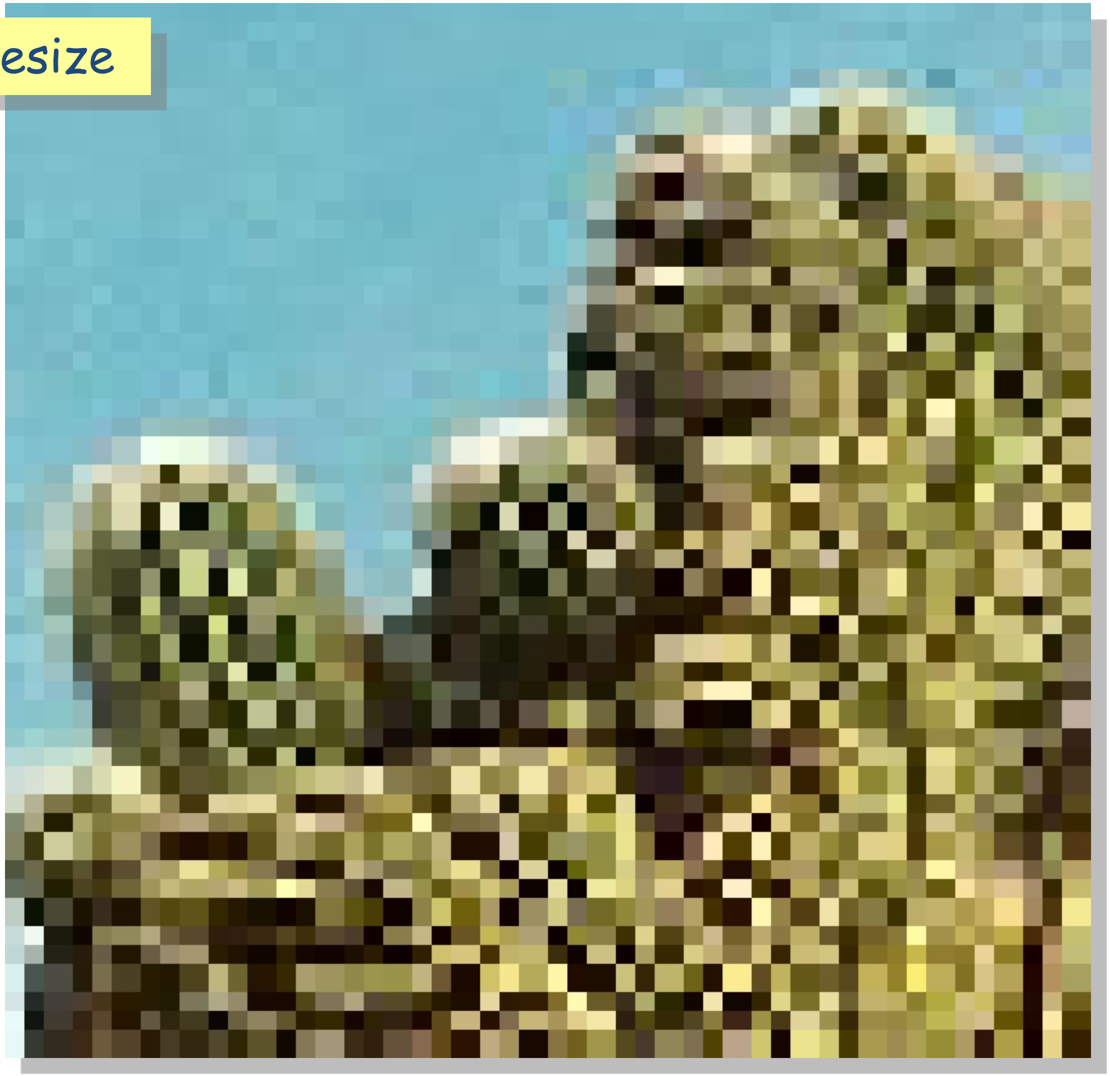
7/3 resize

Empty pixels filled with color from non-empty pixel



Interpolation – (Non Integer)

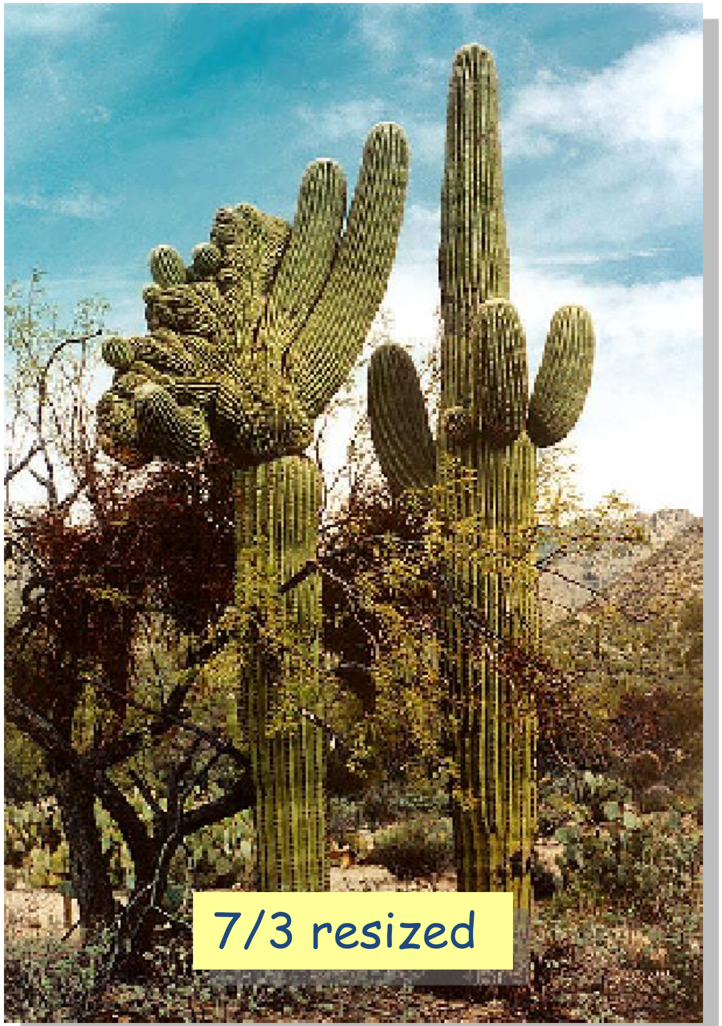
7/3 resize



Empty pixels filled with color from non-empty pixel

Interpolation – (Non Integer)

Original image



7/3 resized

Interpolation

- **Non-adaptive algorithms**
  - Examples
    - nearest neighbour, bilinear, bicubic, spline etc.
  - Depending on their complexity, these use anywhere from 0 to 256 (or more) adjacent pixels when interpolating.
  - The more adjacent pixels they include, the more accurate they can become, but this comes at the expense of much longer processing time.
  - These algorithms can be used to both distort and resize a photo

- **Adaptive algorithms**
  - Many of these apply a different version of their algorithm (on a pixel-by-pixel basis) when they detect the presence of an edge-- aiming to minimize unsightly interpolation artifacts in regions where they are most apparent.
  - These algorithms are primarily designed to maximize artifact-free detail in enlarged photos, so some cannot be used to distort or rotate an image.

End

Sampling, Quantization, Interpolation