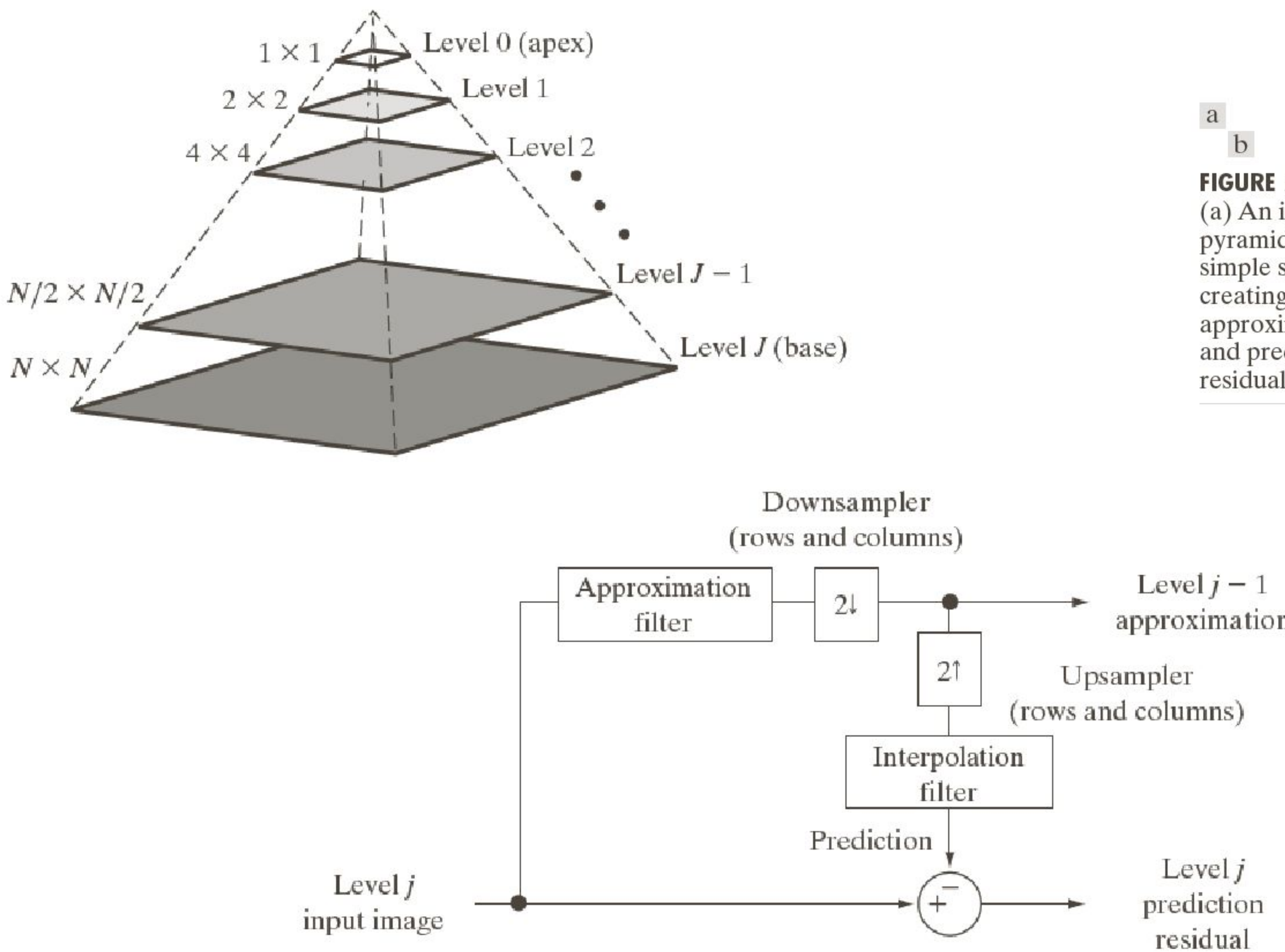
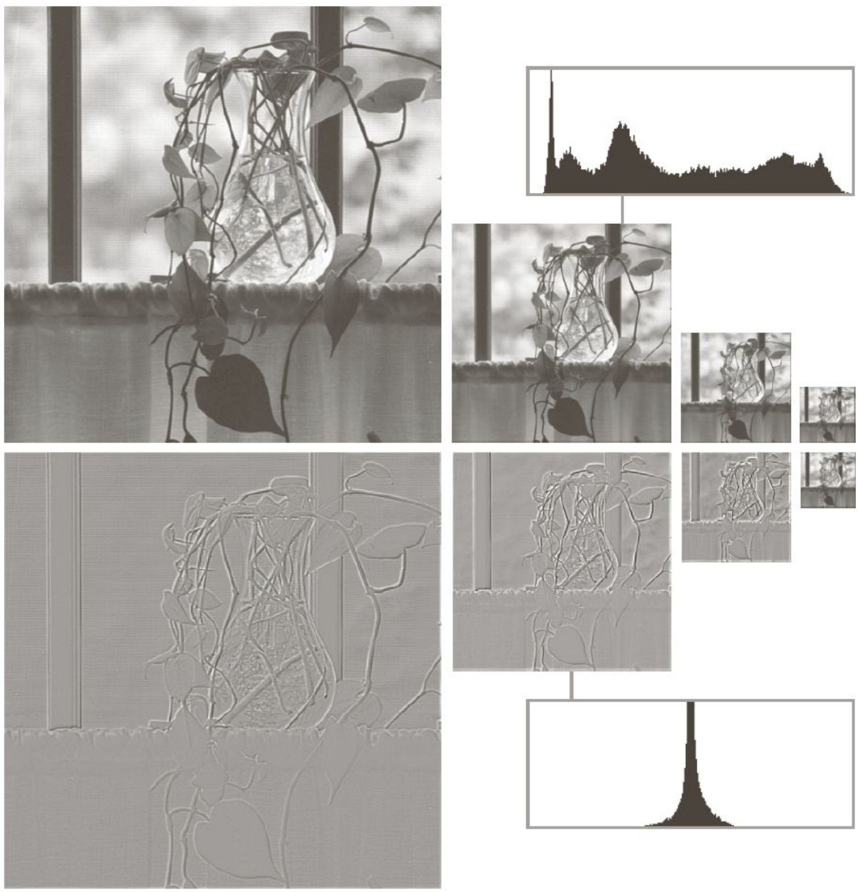
Image Pyramids

SUBRAHMANYAM MURALA
CVPR Lab
School of Computer Science and Statistics
Trinity College Dublin, Ireland



a b

FIGURE 7.2 (a) An image pyramid. (b) A simple system for creating approximation and prediction residual pyramids.



a
b

FIGURE 7.3 Two image pyramids and their histograms: (a) an approximation pyramid; (b) a prediction residual pyramid.

template (filter)

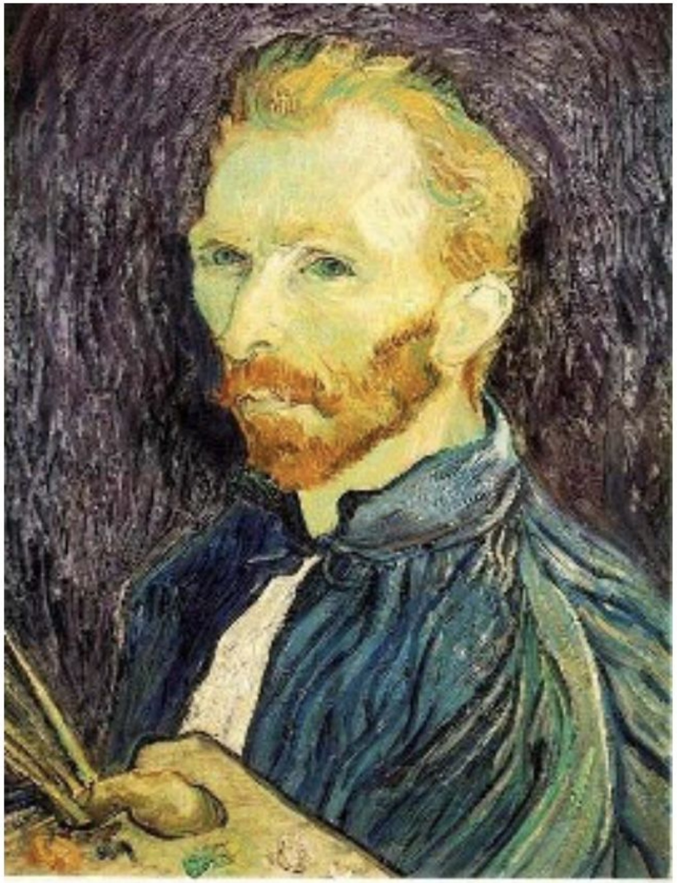
template (filter)

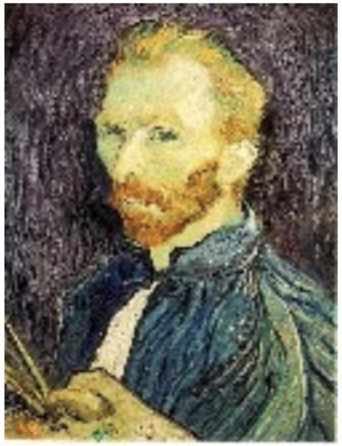
.

- Re-scale the image multiple times! Do correlation on every size!

template (filter)

- **Idea**: Throw away every other row and column to create a 1/2 size image

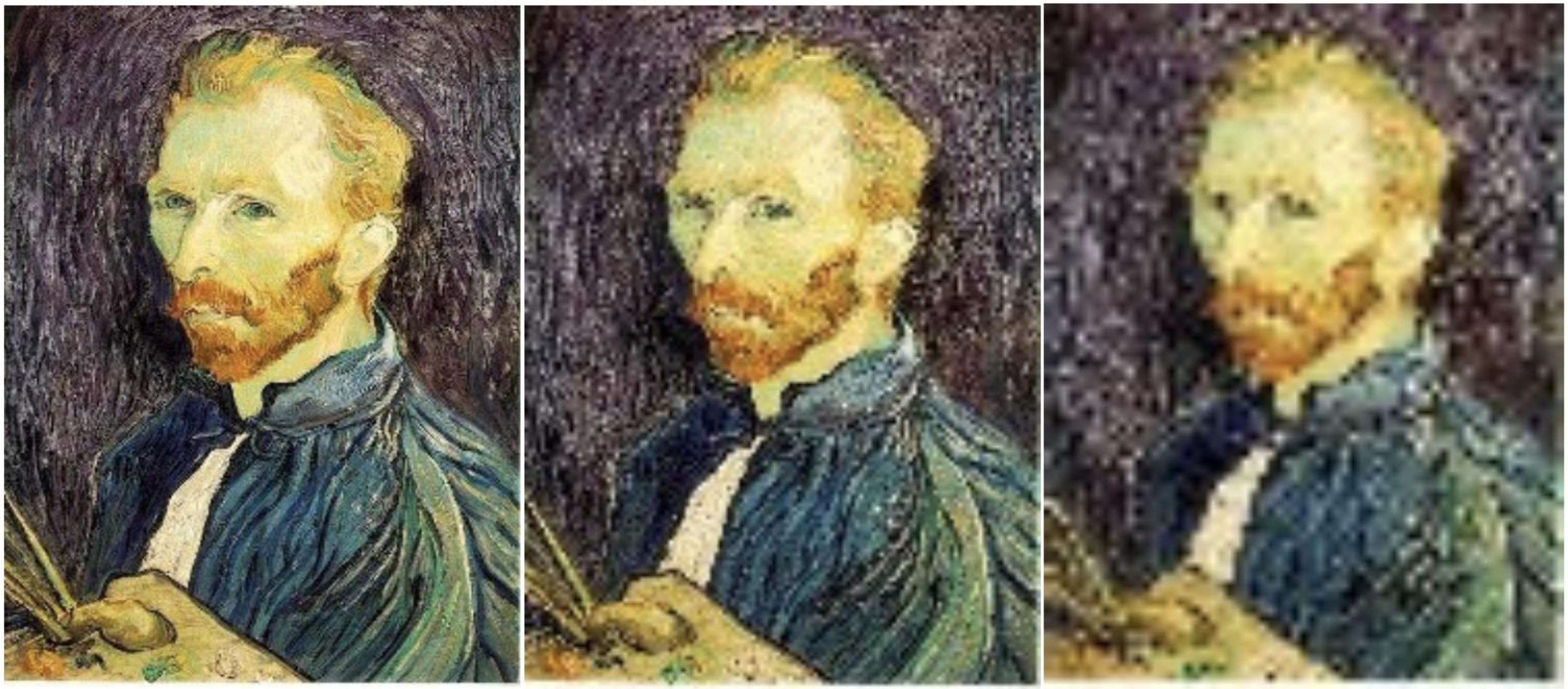




1/4

1/8

- Why does this look so crufty?



1/2   1/4 (2x zoom)   1/8 (4x zoom)

Interpolation

Interpolation

- What is the intensity value of *f (3.4, 7.9)?*

- The most simple form of interpolation is called zeroth-order interpolation. It rounds off to the value of the nearest possible pixel, i.e., f (3.4, 7.9)→f (3, 8).

- A better, but also more computational demanding, approach is to apply first-order interpolation (a.k.a. bilinear interpolation), which weights the intensity values of the four nearest pixels according to how close they are.

Nearest Neighbor Interpolation

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 Interpolation

Simply replicate the value from neighboring pixels

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

Nearest Neighbor Interpolation

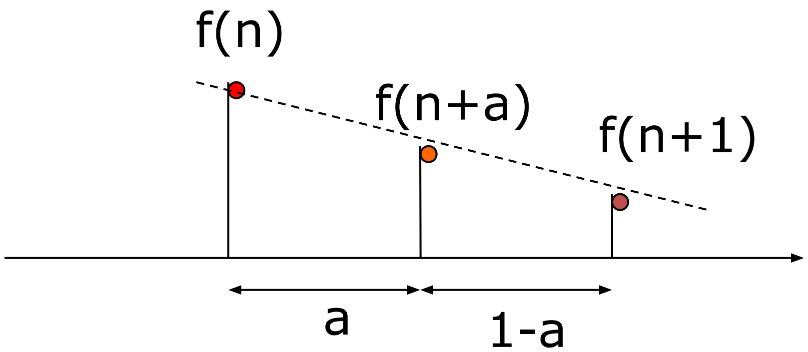
Simply replicate the value from neighboring pixels

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

Linear Interpolation Formula

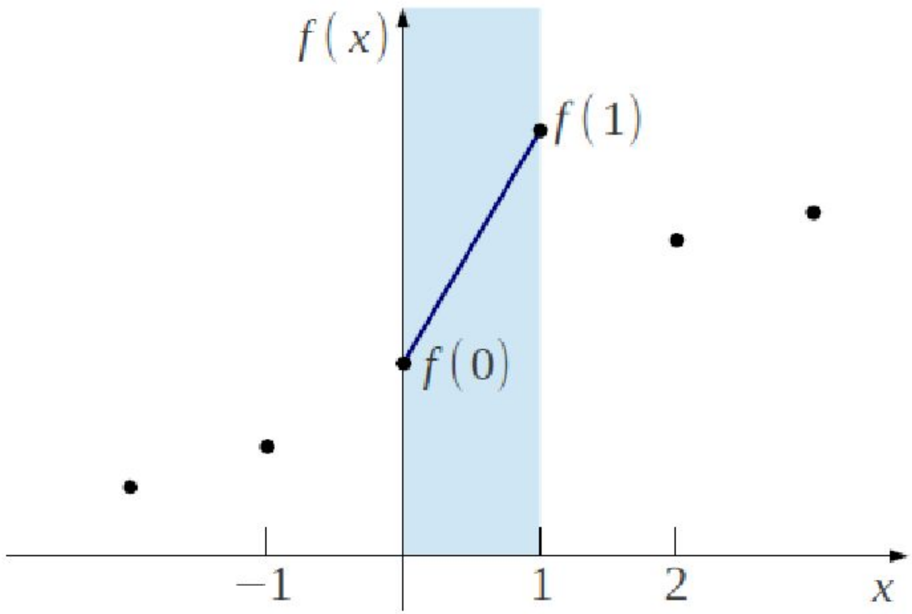
Heuristic: the closer to a pixel, the higher weight is assigned
Principle: line fitting to polynomial fitting (analytical formula)



$$\boxed{f(n+a)=(1-a)\times f(n)+a\times f(n+1),}$$
$0<a<1$

Note: when a=0.5, we simply have the average of two

Linear Interpolation Formula



- Normalization
- Model: $f(x) = a_1 x^1 + a_0 x^0$
- Solve: a_0, a_1

$$\begin{cases} f(0) = a_1 \cdot 0 + a_0 \cdot 1 \\ f(1) = a_1 \cdot 1 + a_0 \cdot 1 \end{cases}$$

Linear Interpolation Formula

$$\begin{cases} f(0) = a_1 \cdot 0 + a_0 \cdot 1 \\ f(1) = a_1 \cdot 1 + a_0 \cdot 1 \end{cases}$$

- Let $\mathbf{y} = \begin{bmatrix} f(0) & f(1) \end{bmatrix}^{\mathrm{T}}$, $\mathbf{B} = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$ and $\mathbf{a} = \begin{bmatrix} a_1 & a_0 \end{bmatrix}^{\mathrm{T}}$
- Then the equations can be written as $\mathbf{y} = \mathbf{Ba}$
- Thus $f(x) = \mathbf{ba} = \mathbf{bB}^{-1}\mathbf{y}$, where $\mathbf{b} = \begin{bmatrix} x^1 & x^0 \end{bmatrix}$
- Example:

$$\begin{aligned} f(0.5) &= \begin{bmatrix} 0.5^1 & 0.5^0 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}^{-1} \mathbf{y} \\ &= \begin{bmatrix} 0.5 & 1 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix} \mathbf{y} \\ &= \begin{bmatrix} 0.5 & 0.5 \end{bmatrix} \mathbf{y} \\ &= \tfrac{1}{2} f(0) + \tfrac{1}{2} f(1) \end{aligned}$$

Numerical Examples

f(n)=[0,120,180,120,0]

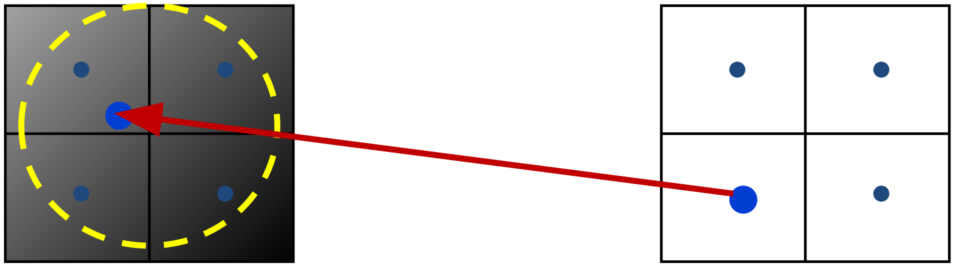
↓ Interpolate at 1/2-pixel

f(x)=[0,60,120,150,180,150,120,60,0], x=n/2

↓ Interpolate at 1/3-pixel

Bilinear Interpolation

The assigned value is an intermediate value between the four nearest pixels:



Bilinear interpolation

What about in 2D?

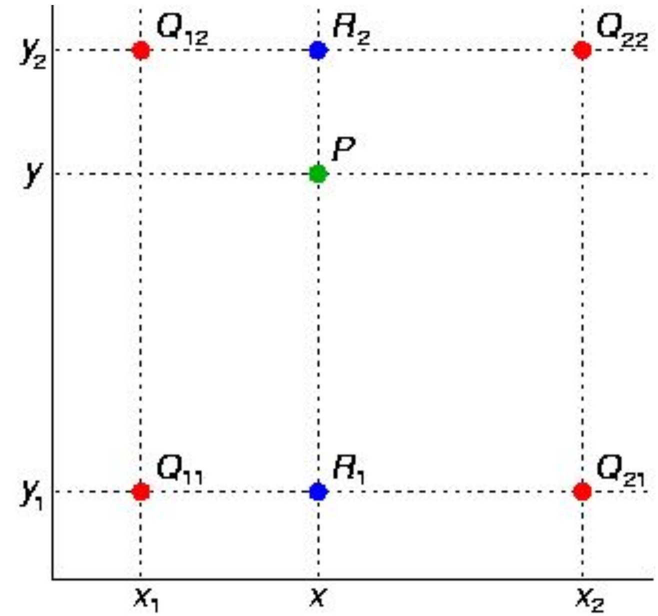
Interpolate in x, then in y

Example

We know the red values

Linear interpolation in x between red values gives us the blue values

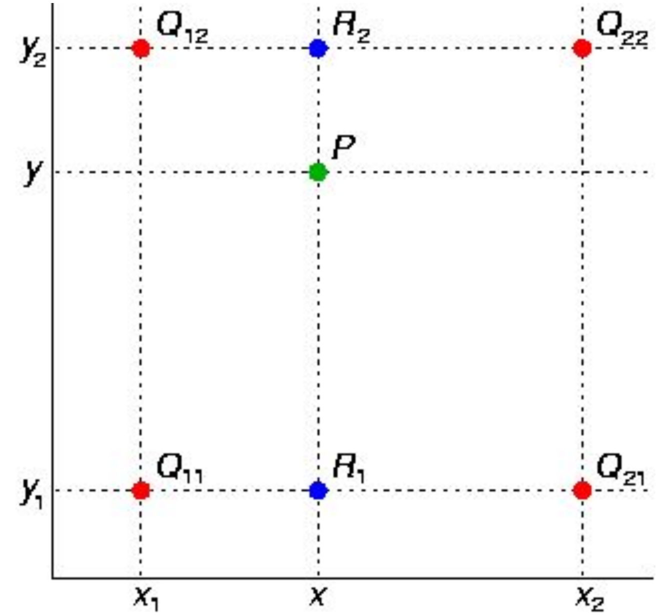
Linear interpolation in y between the blue values gives us the answer



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

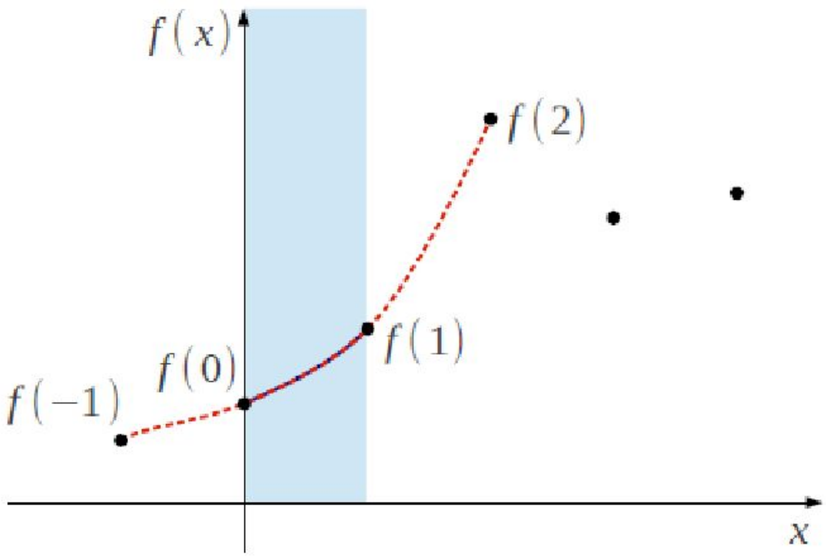
Bilinear interpolation

$$f(x,y) \approx \frac{f(Q_{11})}{(x_2-x_1)(y_2-y_1)}(x_2-x)(y_2-y) + \frac{f(Q_{21})}{(x_2-x_1)(y_2-y_1)}(x-x_1)(y_2-y) + \frac{f(Q_{12})}{(x_2-x_1)(y_2-y_1)}(x_2-x)(y-y_1) + \frac{f(Q_{22})}{(x_2-x_1)(y_2-y_1)}(x-x_1)(y-y_1).$$



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

Cubic Interpolation



- Model: $f(x) = \sum_{i=0}^{3} a_i x^i = a_3 x^3 + a_2 x^2 + a_1 x^1 + a_0$
- $$\begin{cases} f(-1) = a_3 \cdot (-1)^3 + a_2 \cdot (-1)^2 + a_1 \cdot (-1)^1 + a_0 \cdot (-1)^0 \\ f(0) = a_3 \cdot 0^3 + a_2 \cdot 0^2 + a_1 \cdot 0^1 + a_0 \cdot 0^0 \\ f(1) = a_3 \cdot 1^3 + a_2 \cdot 1^2 + a_1 \cdot 1^1 + a_0 \cdot 1^0 \\ f(2) = a_3 \cdot 2^3 + a_2 \cdot 2^2 + a_1 \cdot 2^1 + a_0 \cdot 2^0 \end{cases}$$

Cubic Interpolation

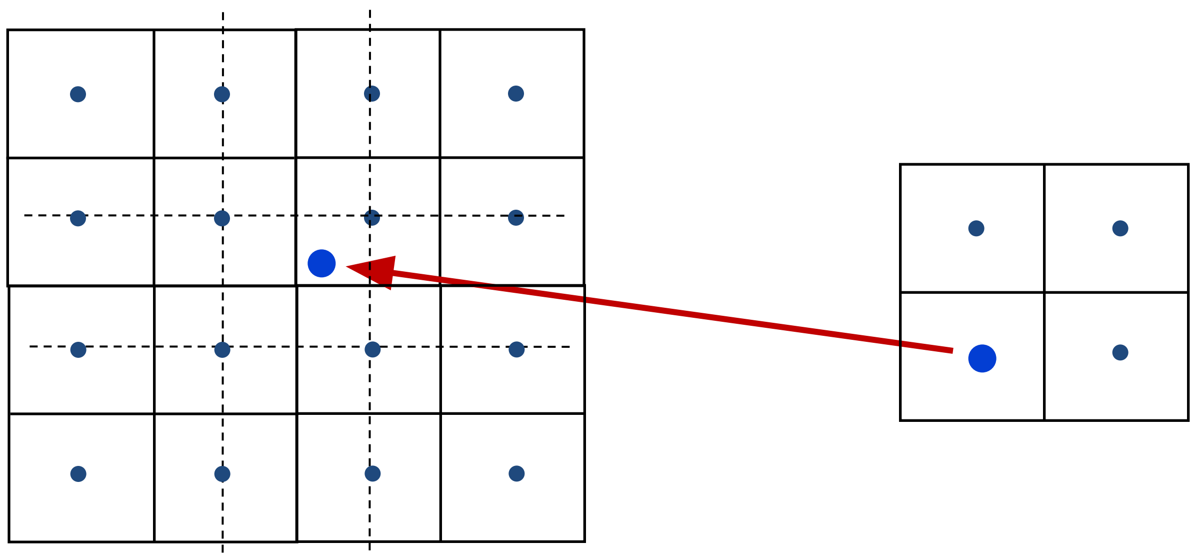
- Let
  - $\mathbf{y} = \begin{bmatrix} f(-1) & f(0) & f(1) & f(2) \end{bmatrix}^{\mathrm{T}}$
  - $\mathbf{B} = \begin{bmatrix} (-1)^3 & (-1)^2 & (-1)^1 & (-1)^0 \\ 0^3 & 0^2 & 0^1 & 0^0 \\ 1^3 & 1^2 & 1^1 & 1^0 \\ 2^3 & 2^2 & 2^1 & 2^0 \end{bmatrix} = \begin{bmatrix} -1 & 1 & -1 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 8 & 4 & 2 & 1 \end{bmatrix}$
  - $\mathbf{a} = \begin{bmatrix} a_3 & a_2 & a_1 & a_0 \end{bmatrix}^{\mathrm{T}}$
- Then the equations can be written as $\mathbf{y} = \mathbf{Ba}$
- Thus $f(x) = \mathbf{ba} = \mathbf{bB}^{-1}\mathbf{y}$, where $\mathbf{b} = \begin{bmatrix} x^3 & x^2 & x^1 & x^0 \end{bmatrix}$
- Example:

$$f(0.5) = \begin{bmatrix} 0.5^3 & 0.5^2 & 0.5^1 & 0.5^0 \end{bmatrix} \begin{bmatrix} -0.167 & 0.5 & -0.5 & 0.167 \\ 0.5 & -1 & 0.5 & 0 \\ -0.333 & -0.5 & 1 & -0.167 \\ 0 & 1 & 0 & 0 \end{bmatrix} \mathbf{y}$$
$$= \begin{bmatrix} -0.0625 & 0.5625 & 0.5625 & -0.0625 \end{bmatrix} \mathbf{y}$$
$$= \frac{1}{16} \begin{bmatrix} -1 & 9 & 9 & -1 \end{bmatrix} \mathbf{y}$$

Bicubic Interpolation

The assign value is a weighted sum of the 4x4 nearest pixels:

$$v(s,t) = \sum_{i,j=0}^{3} a_{ij} s^i t^j$$



Comparison of Interpolation Approaches

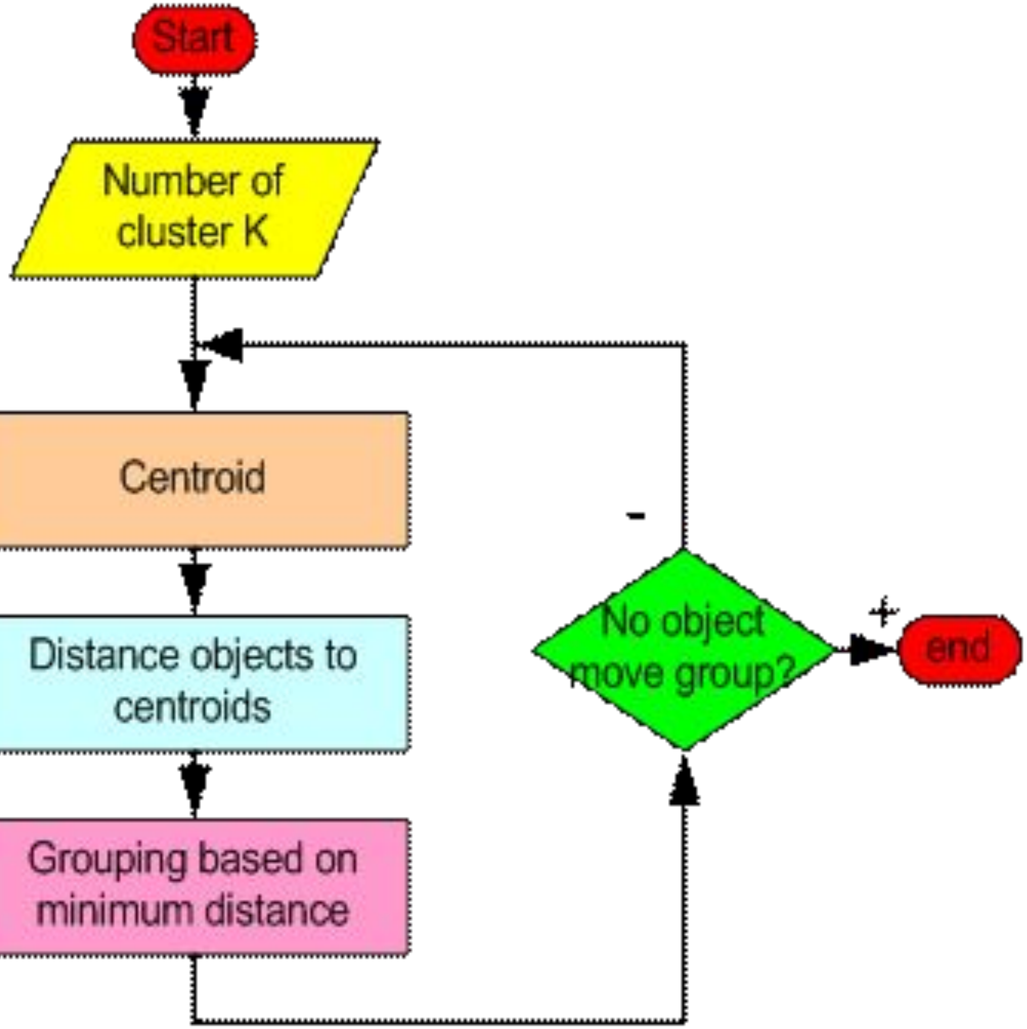
Nearest Neighbor

Bi-Linear

Bi-Cubic

Image Segmentation

K-Means Clustering



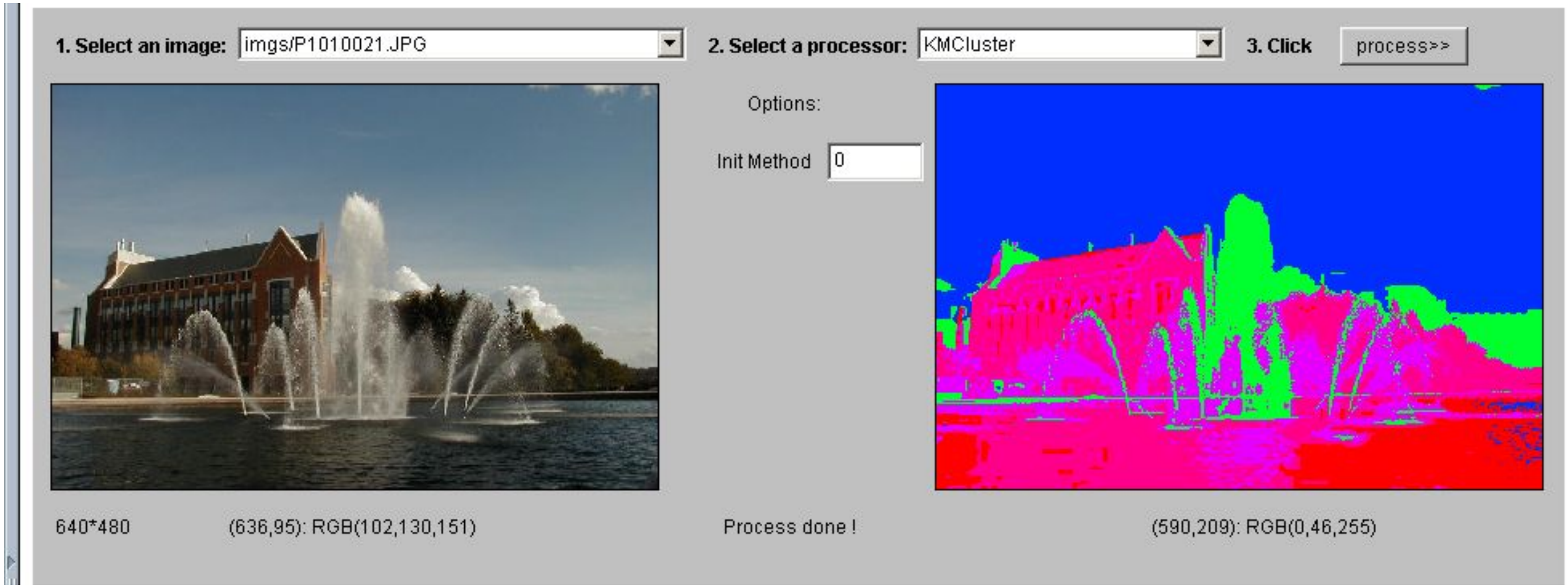
K-Means Algorithm

- assume K clusters $C_1, C_2, \ldots, C_K$ with means $m_1, m_2, \ldots, m_K$.
- *least squares error measure* measures how close the data are to their assigned clusters

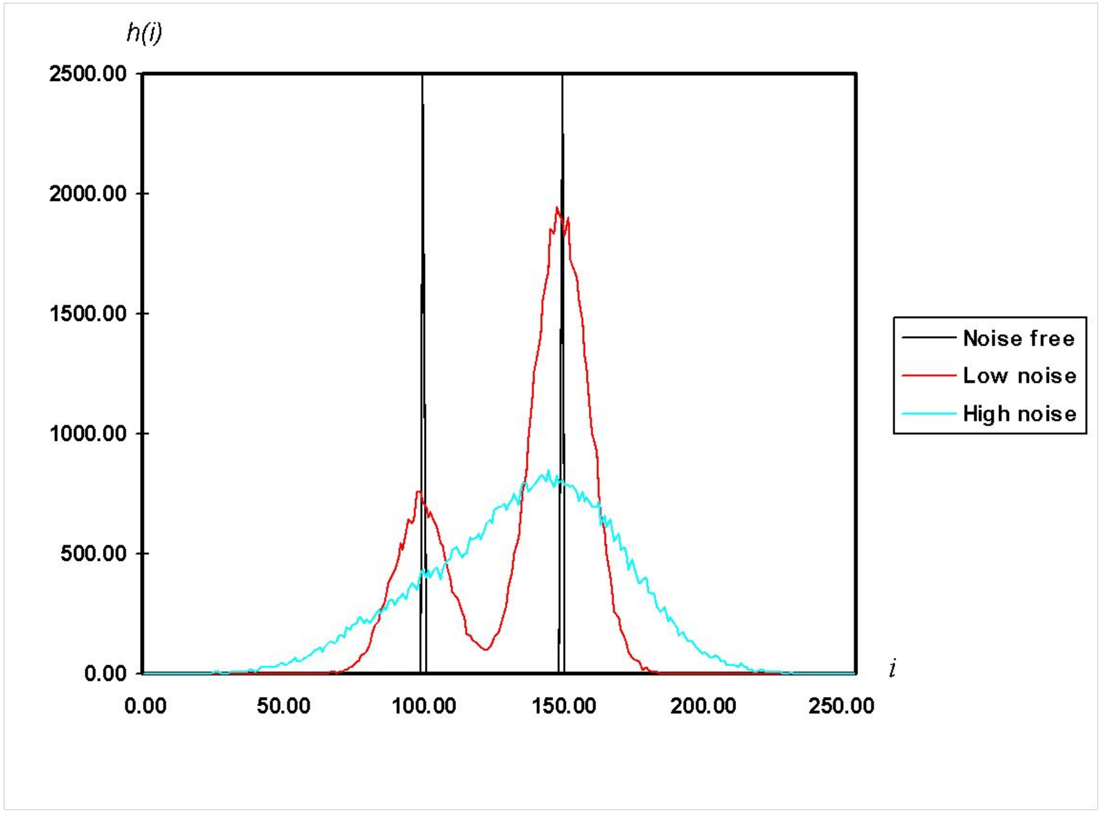
$$D = \sum_{k=1}^{K} \sum_{x_i \in C_k} \|x_i - m_k\|^2.$$

- could consider *all* possible partitions into K clusters and select the one that minimizes D
- is K known in advance?

K-Means Examples



Greylevel histogram-based segmentation



Semantic Segmentation

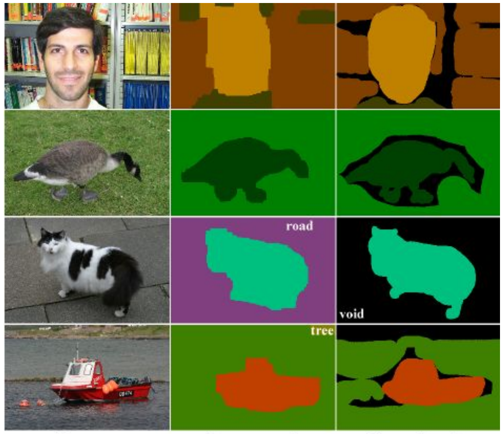


Image Our approach Ground truth