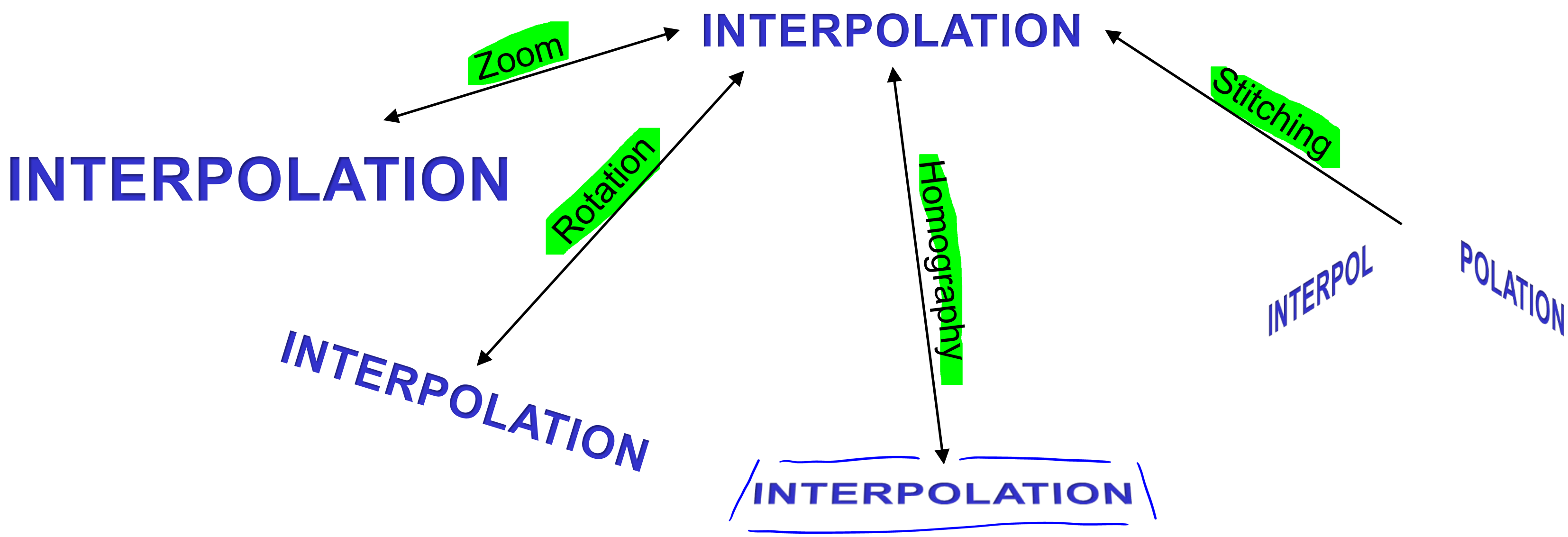
5 Image Interpolation

5.1 Interpolation of Image Signals

Goal: use available data to estimate sample values at new locations

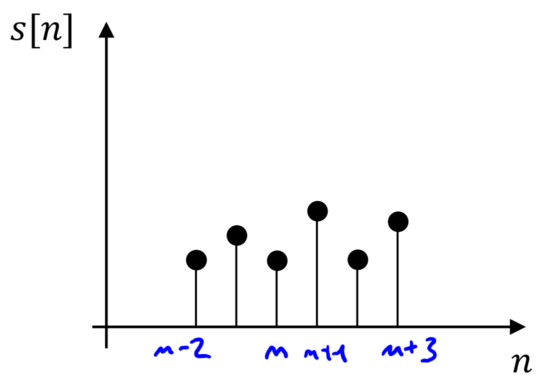
- Resize (zoom in/out)
- Distortions (affine transforms, homographies)
- Up-sampling (e.g. for sub-pixel accuracy)

Not video interpolation => frame rate upconversion (FRUC) in this lecture



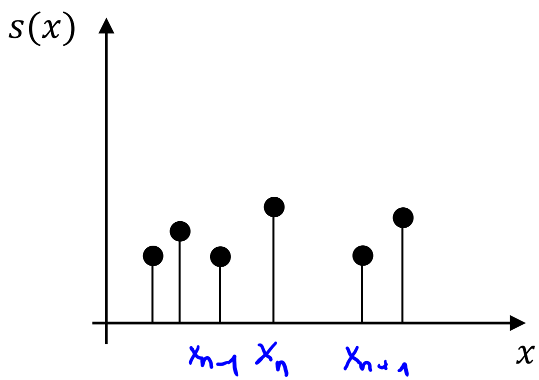
In general: equally (1) vs non-equally (2) sampled points

1



Advantage: allows separability
- First interpolate rows then columns
- Nearest neighbor, Bi-linear (Bi-cubic), Lanczos, Splines

2

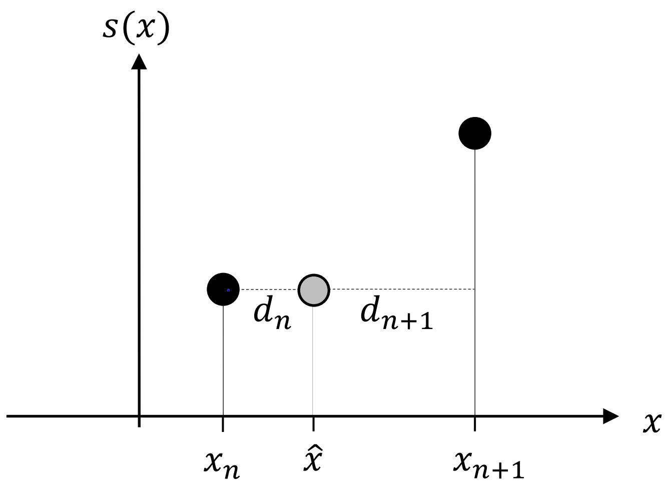


No assumption about sample locations
- Requires generic interpolation methods
- Typically based on triangulation
- No easy generalization from 1D → 2D

Zero-Order Interpolation

Interpolated value

- Copy the value of the nearest available neighbor sample



Rule:

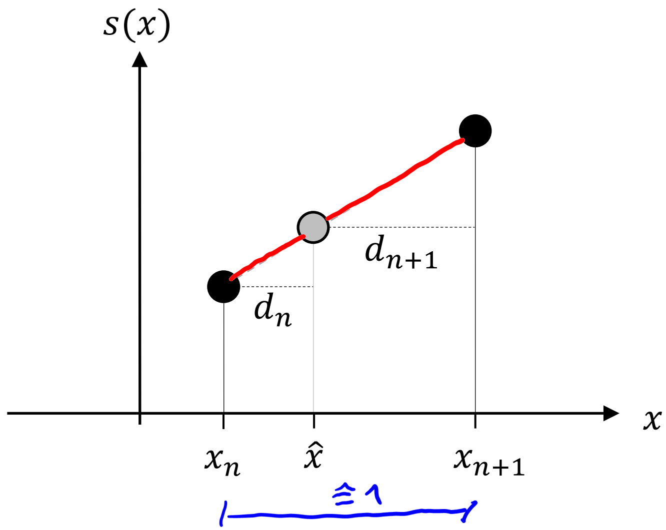
$$s(\hat{x}) = \begin{cases} s(x_n) & d_n \leq d_{n+1} \\ s(x_{n+1}) & d_{n+1} > d_n \end{cases}$$

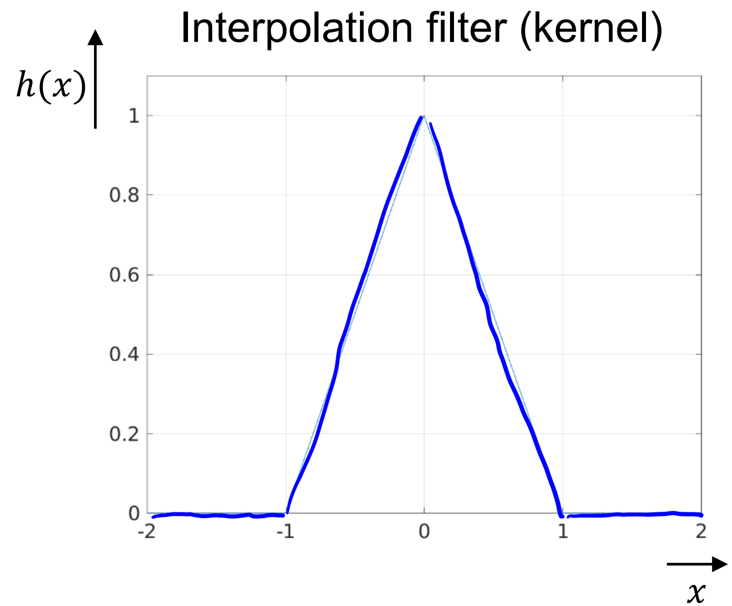
Produces considerable artefacts , but computationally easy

- E.g. distortions of straight edges

First-Order Interpolation

Fits straight line between two consecutive samples ⇨ **linear** interpolation





$$s(\hat{x}) = \frac{x_{n+1} - \hat{x}}{x_{n+1} - x_n} s(x_n) + \frac{\hat{x} - x_n}{x_{n+1} - x_n} s(x_{n+1})$$

$$= \frac{d_{n+1}}{d_n + d_{n+1}} s(x_n) + \frac{d_n}{d_n + d_{n+1}} s(x_{n+1})$$

For unitary pixel spacing: $d_n + d_{n+1} = 1$

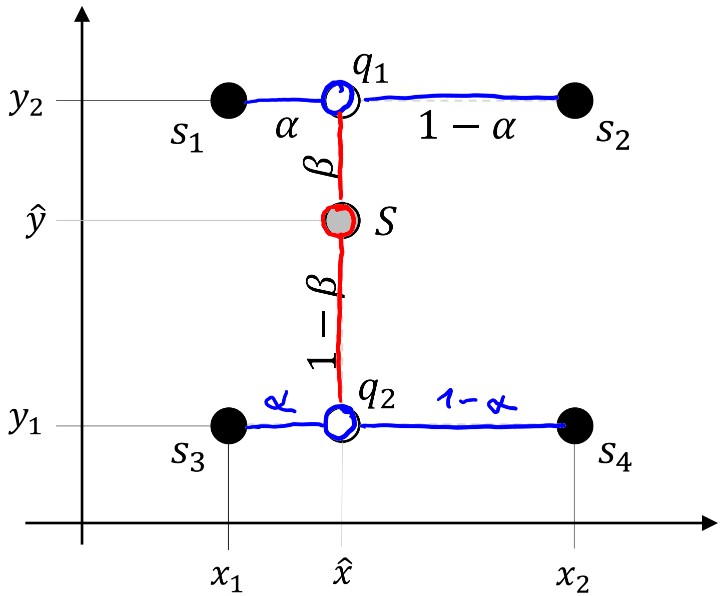
$$\boxed{s(\hat{x}) = d_{n+1} s(x_n) + d_n s(x_{n+1})}$$

or $s(\hat{x}) = \sum_{\forall n} s(x_n) \cdot h(\hat{x} - x_n)$ $\hat{=}$ conv.

Bilinear Interpolation => 2D

For image signals

- first applied to rows then to columns (separability) or vice versa



Normalized distances

$$\alpha = \frac{\hat{x} - x_1}{x_2 - x_1} \qquad \beta = \frac{\hat{y} - y_1}{y_2 - y_1}$$

1) Linear interpolations

$$q_1 = (1-\alpha)s_1 + \alpha s_2$$

(*)

$$q_2 = (1-\alpha)s_3 + \alpha s_4$$

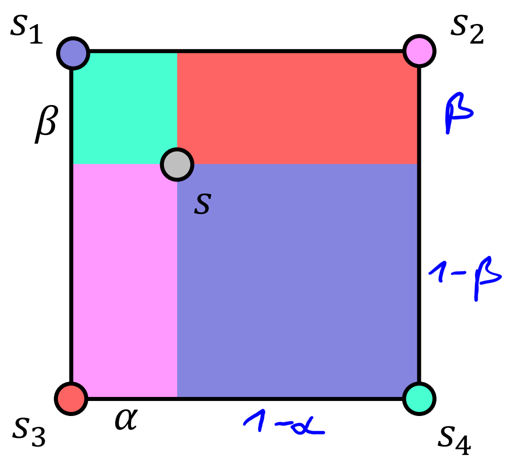
Bilinear interpolation

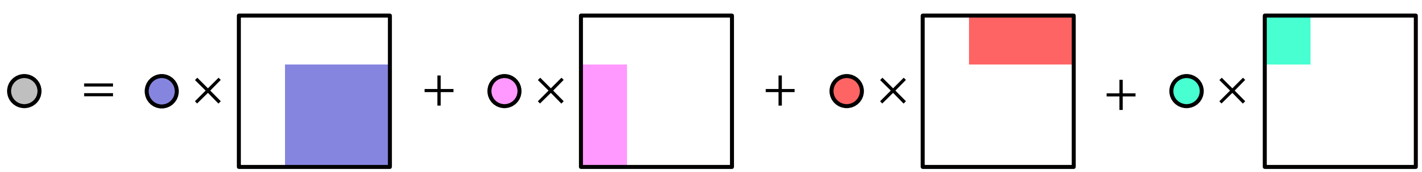
$$\boxed{s = (1-\beta)q_1 + \beta q_2 \overset{(*)}{=} (1-\beta)(1-\alpha)s_1 + (1-\beta)\alpha s_2 + \beta(1-\alpha)s_3 + \beta\alpha s_4}$$

Bilinear Interpolation

Interpretation: area-based weighted linear combination

$$s = \boxed{(1-\beta)(1-\alpha)}s_1 + \boxed{(1-\beta)\alpha}s_2 + \boxed{\beta(1-\alpha)}s_3 + \boxed{\beta\alpha}s_4$$





Ideal Interpolation

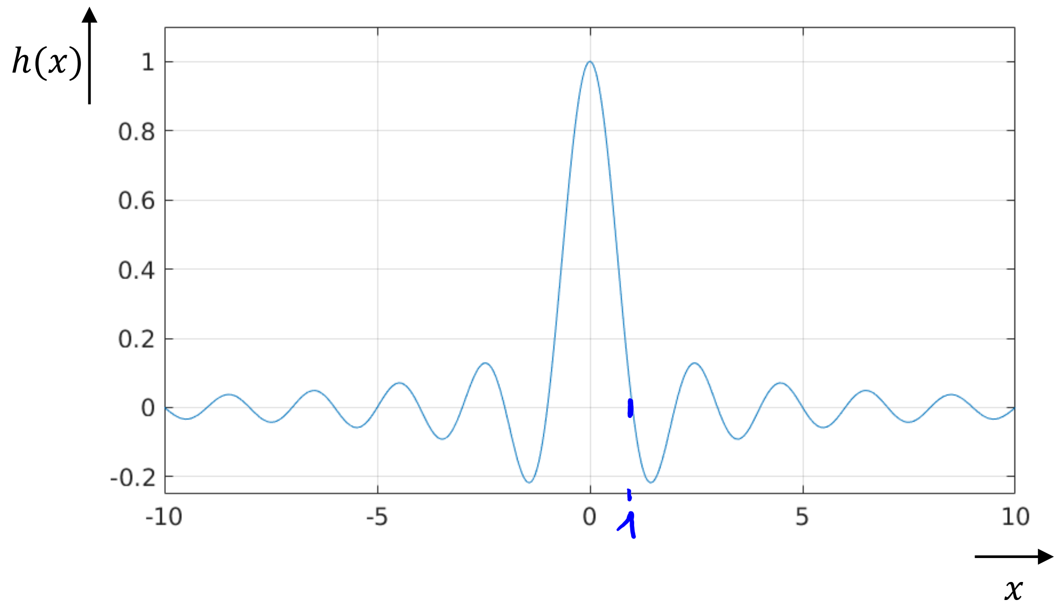
Sinc filter

- Maintains frequency content (doesn't affect LF, doesn't add HF)
- Ideal low-pass filter (perfect cut-off HF)

$$h(x) = \frac{\sin(\pi x)}{\pi x} = \mathrm{sinc}(x)$$

$g(\hat{x}) = \sum_{\forall n} g(x_n)\, \mathrm{sinc}(\hat{x} - x_n)$

(see 5-5)



Problems

- Infinite summation required
- Blurring due to finite signal length and instationarity of image signals

Lanczos Interpolator

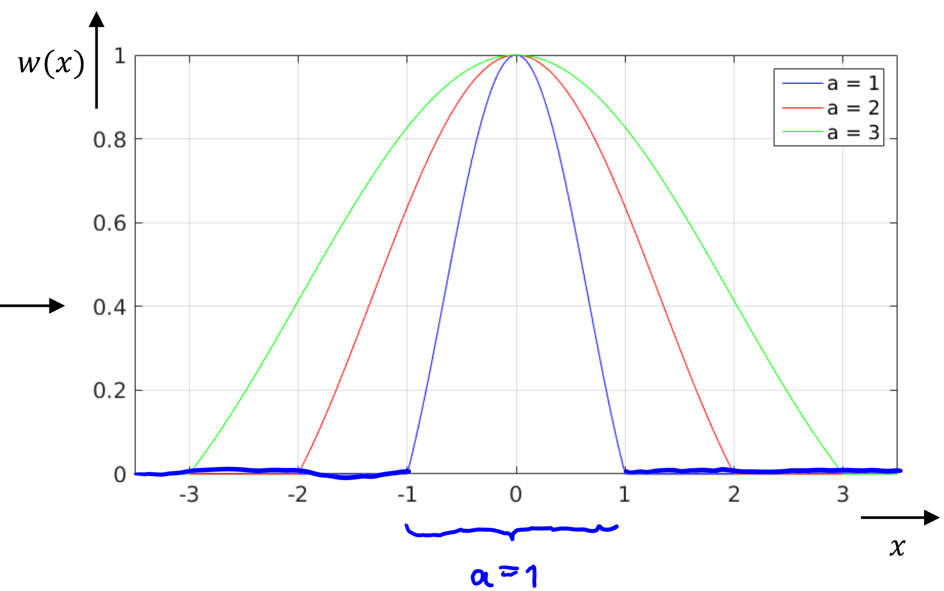
Practical approximation of sinc filter

- Finite support

Kornél Lánczos
(1893-1974)

Multiply sinc filter by **Lanczos window**

$$w(x) = \begin{cases} \operatorname{sinc}\left(\frac{x}{a}\right) & -a < x < a \\ 0 & \text{otherwise} \end{cases}$$

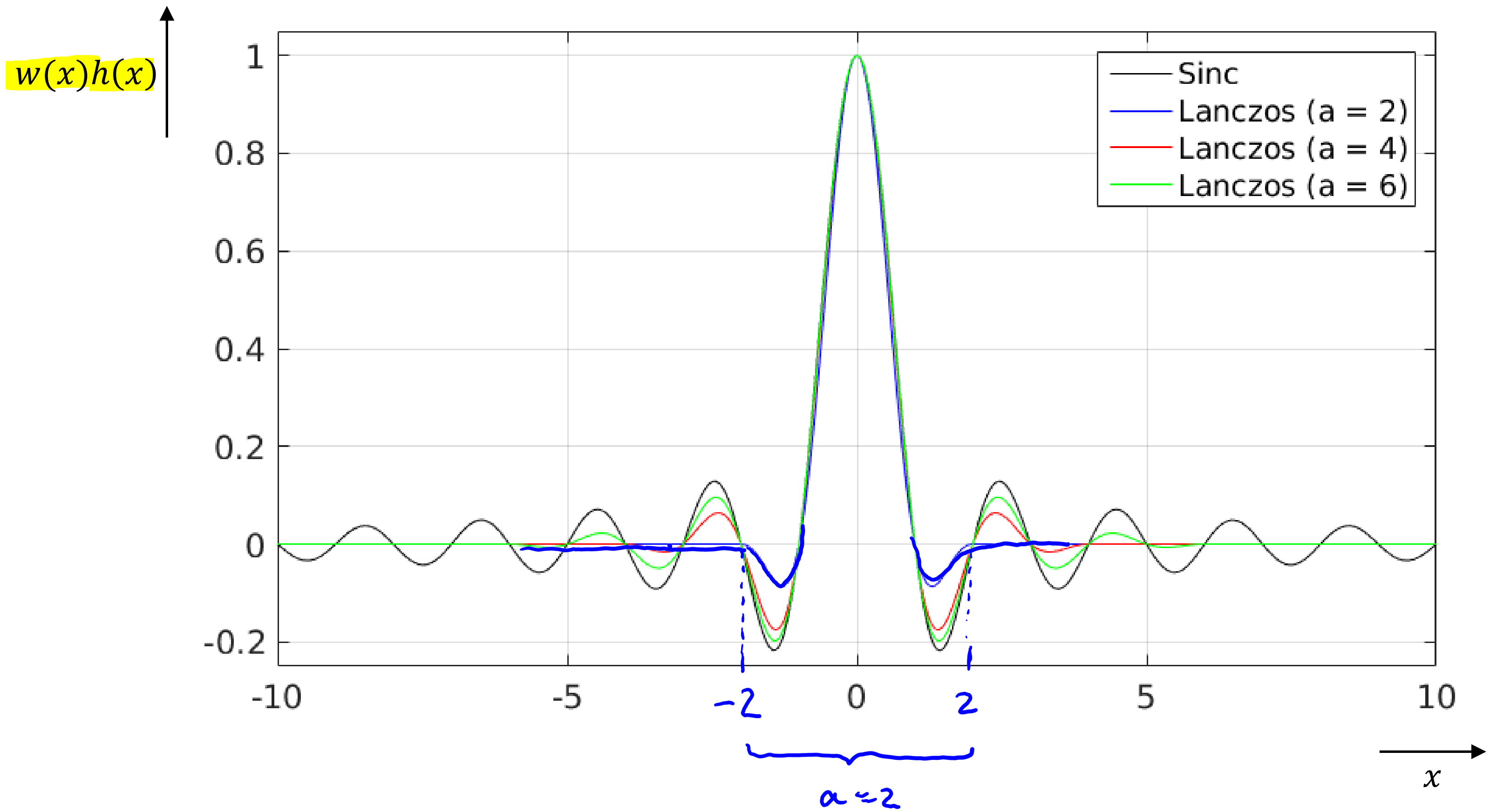


Parameter a

- Determines filter width

Lanczos Interpolator

Comparison



Examples (2× magnification)

„zero order"

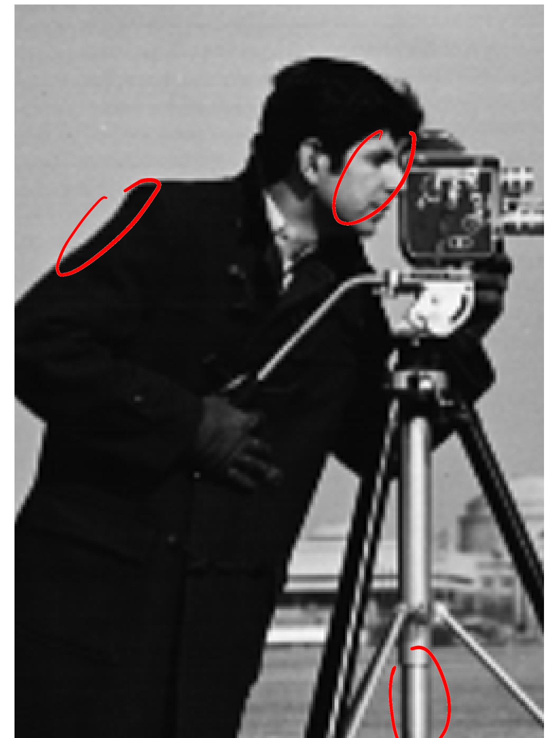
Nearest neighbor

„first order"

Linear interpolation

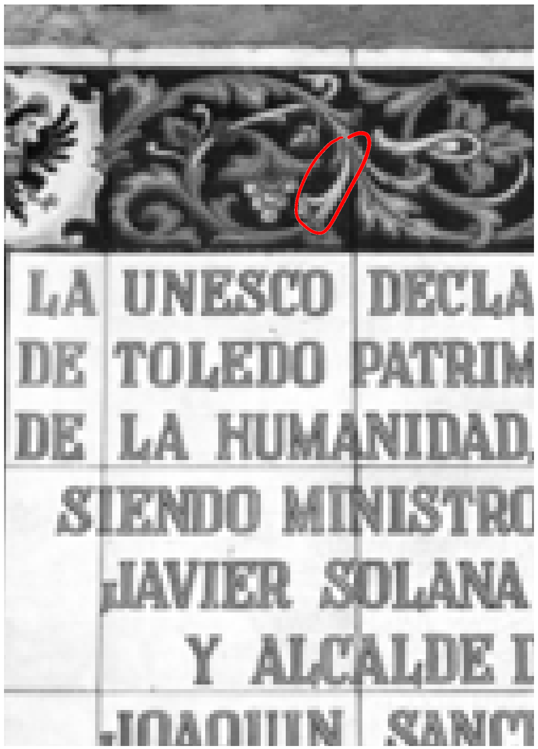
„windowed ideal"

Lanczos interpolation



Examples (2× magnification)

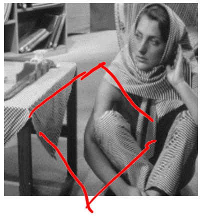
Nearest neighbor



Linear interpolation

Lanczos interpolation

Examples (45º rotation)



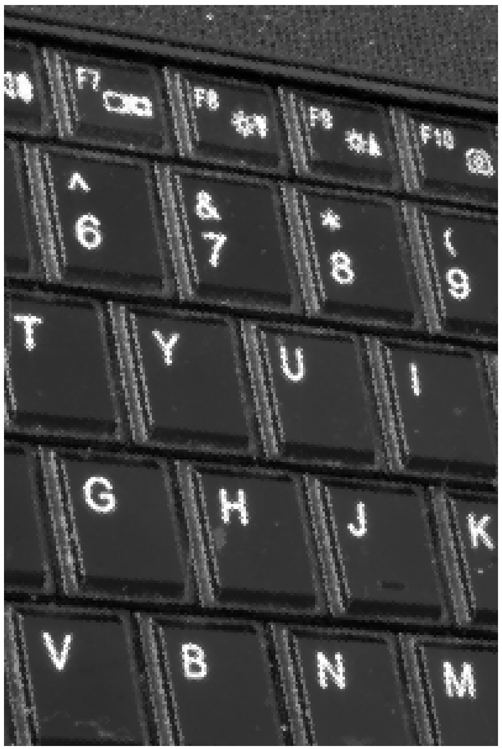
Nearest neighbor

Linear interpolation

Lanczos interpolation

Examples (-45º rotation)

Nearest neighbor

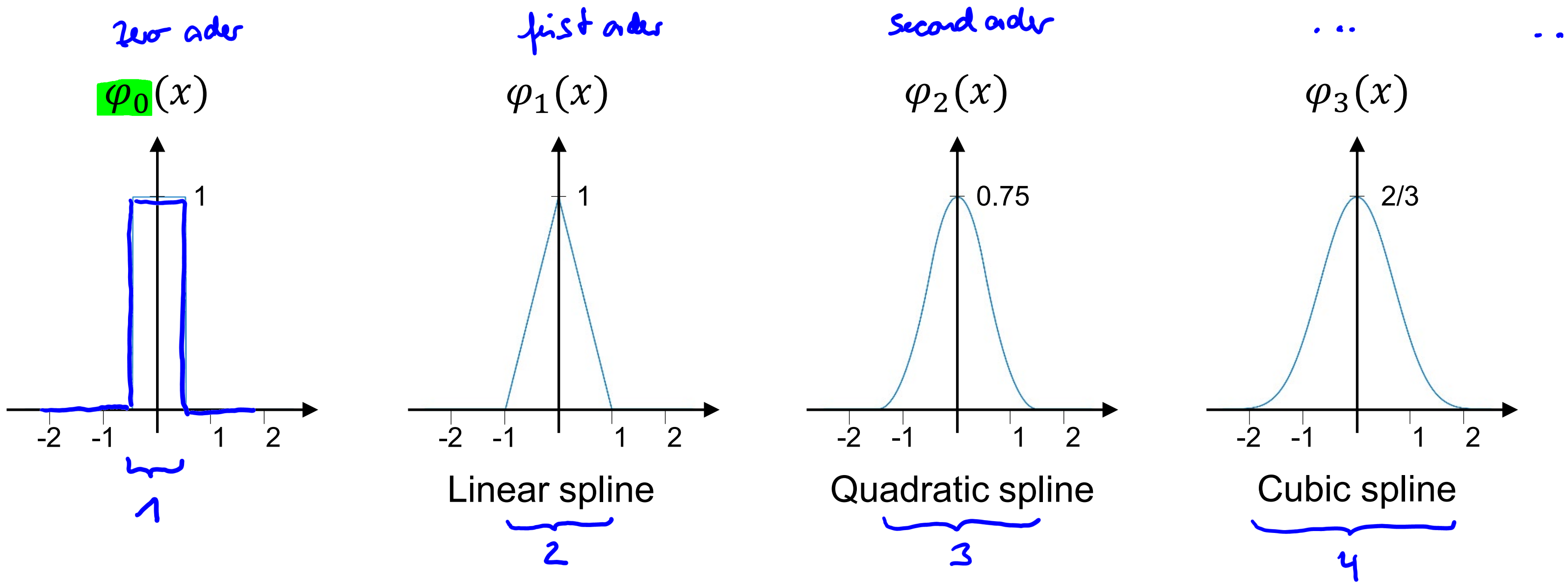


Linear interpolation

Lanczos interpolation

5.2 B-Splines Basis Functions

Construction rule for equidistant spacing of signal samples



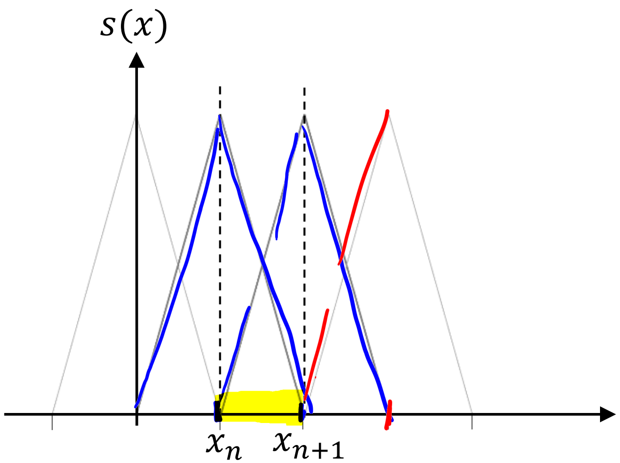
Zero-order B-spline:

$$\varphi_0(x) = \begin{cases} 1 & -1/2 < x < 1/2 \\ 1/2 & |x| = 1/2 \\ 0 & \text{otherwise} \end{cases}$$

Recursion:

$$\boxed{\varphi_n(x) = \varphi_0(x) * \varphi_{n-1}}$$

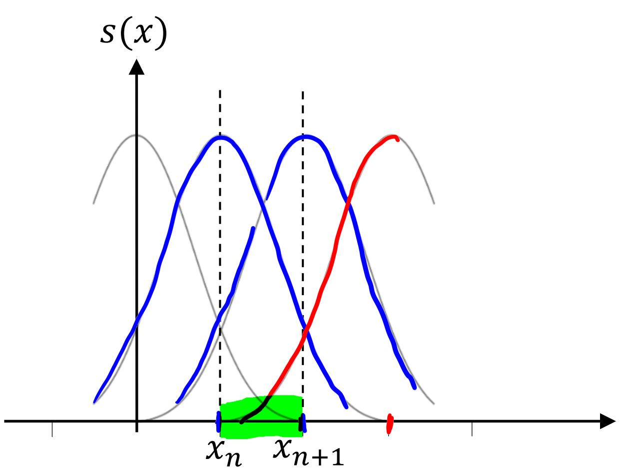
Superposition of B-Splines



Linear splines

- Two overlapping splines between two (consecutive) sampling points

$$s(x) = c_n \varphi_1(x - x_n) + c_{n+1} \varphi_1(x - x_{n+1})$$



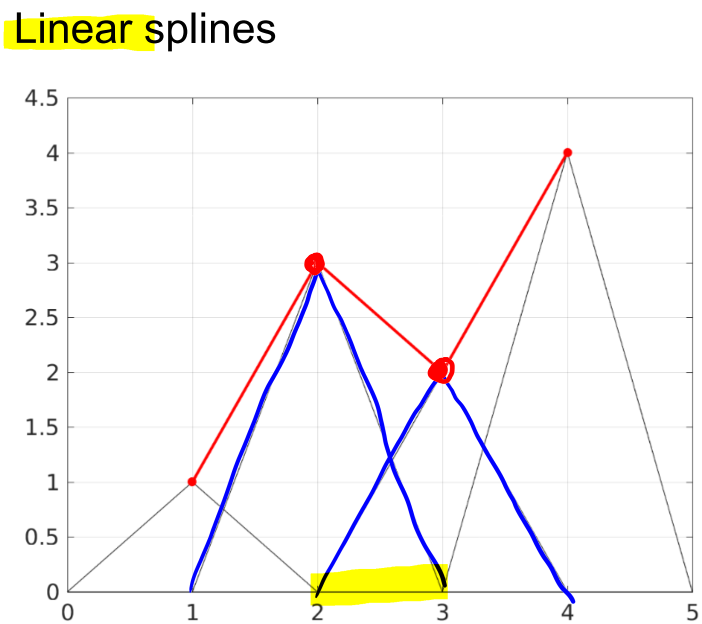
Cubic splines

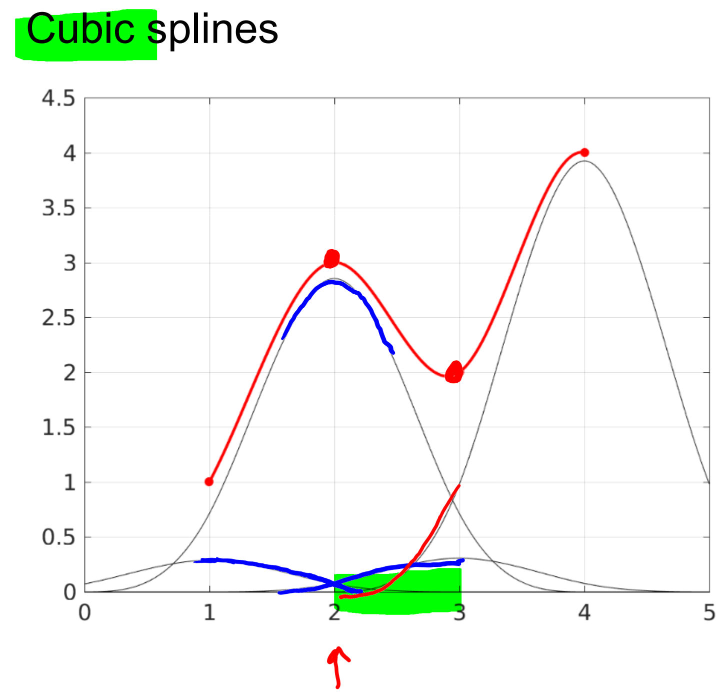
- Four overlapping splines between four (consecutive) sampling points

$$s(x) = c_{n-1} \varphi_3(x - x_{n-1}) + c_n \varphi_3(x - x_n) +$$
$$+ c_{n+1} \varphi_3(x - x_{n+1}) + c_{n+2} \varphi_3(x - x_{n+2})$$

Superposition of Weighted B-Splines

Example: Interpolate four sample values $s(x) = [1,3,2,4]$ at positions $x_n = [1,2,3,4]$

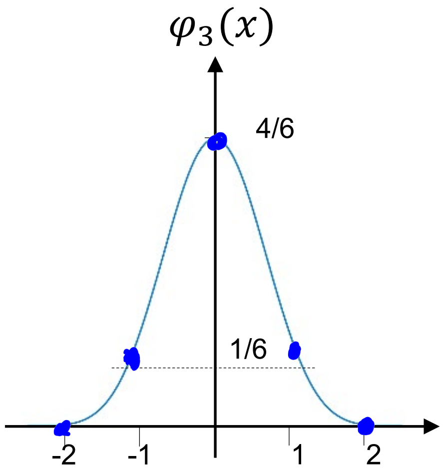




Obtaining Spline Coefficients

Consider cubic spline interpolation

$$\varphi_3 = \begin{cases} \frac{2}{3} - x^2 + \frac{|x|^3}{2} & \text{for } 0 \le |x| < 1 \\ \frac{(2-|x|)^3}{6} & \text{for } 1 \le |x| < 2 \\ 0 & \text{otherwise} \end{cases}$$



$$\varphi_3(-1) = \frac{1}{6} \qquad \varphi_3(0) = \frac{4}{6} \qquad \varphi_3(1) = \frac{1}{6}$$

Constraint

- Spline interpolation reconstructs **exactly** available samples $g(x_n)$

$$s(x_n) = \frac{1}{6}(c_{n-1} + 4c_n + c_{n+1})$$

given point

Obtaining Spline Coefficients

Matricial notation for signal $s(x_n)$ for $n = 0, 1, \dots, N-1$

$$\begin{bmatrix} s(x_0) \\ s(x_1) \\ s(x_2) \\ \vdots \\ s(x_{N-1}) \end{bmatrix} = \frac{1}{6} \underbrace{\begin{bmatrix} 4 & 1 & 0 & 0 & \cdots & 0 \\ 1 & 4 & 1 & 0 & \cdots & 0 \\ 0 & 1 & 4 & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 4 & 1 \\ 0 & \cdots & 0 & 0 & 1 & 4 \end{bmatrix}}_{\text{all weights of cubic spline at sampling points}} \underbrace{\begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{N-1} \end{bmatrix}}_{\text{unknown}}$$

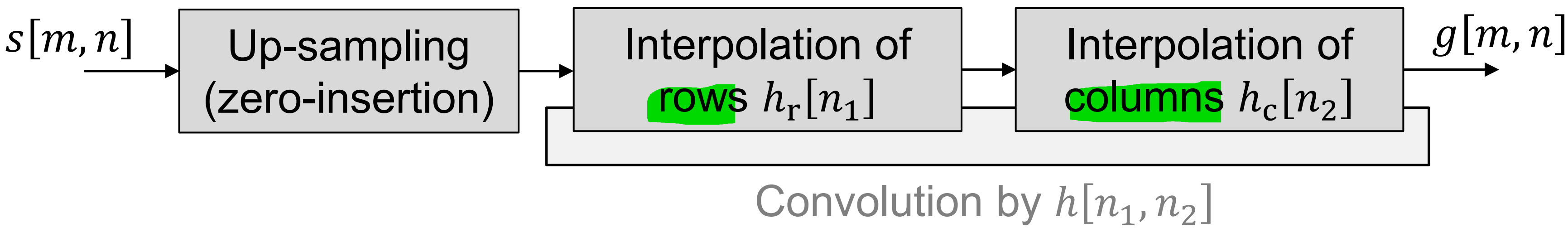
given sampling points

5.3 Image Up-sampling

Special case of image interpolation

- Both available and unknown samples are equally spaced

Up-sampling by factor L



Row vector

Interpolation filter: $\boxed{h[m,n] = h_\mathrm{r}[m] * h_\mathrm{c}[n]}$

Column vector

- Filter gain of L^2 to compensate energy loss due to insertion of zeros

Zero-Order Interpolation

Zero-order hold filter (nearest neighbor)

- Repeat (hold) pixels along scan line, afterwards repeat each scan line

$$h_\mathrm{r}[m] = h_\mathrm{c}^T[n] = \mathbf{1}_{L\times 1} \longrightarrow \boxed{h[m,n] = \mathbf{1}_{L\times L}}$$

Example (L=2): $h[m,n] = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$ 2D impulse response function

$$\begin{bmatrix} 1 & 3 & 2 \\ 4 & 5 & 6 \end{bmatrix} \longrightarrow \begin{bmatrix} 1 & 0 & 3 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 5 & 0 & 6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \longrightarrow \begin{bmatrix} 1 & 1 & 3 & 3 & 2 & 2 \\ 1 & 1 & 3 & 3 & 2 & 2 \\ 4 & 4 & 5 & 5 & 6 & 6 \\ 4 & 4 & 5 & 5 & 6 & 6 \end{bmatrix}$$

original image

First-Order Interpolation

Linear interpolation (first-order)

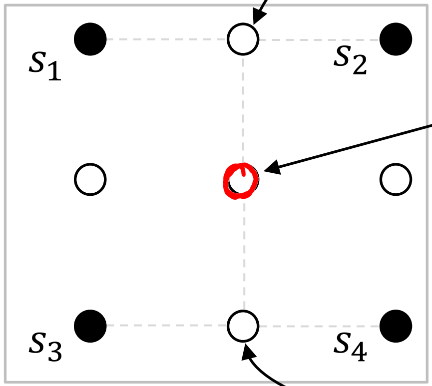
- Fits straight line between pixels along row / column

$$h_{\mathrm{r}}[m] = h_{\mathrm{c}}^{\mathrm{T}}[n] = \frac{1}{L}\begin{bmatrix} 1 & 2 & \cdots & L-1 & L & L-1 & \cdots & 2 & 1 \end{bmatrix}$$

Example (L=2)

$$h_{\mathrm{r}}[m] = h_{\mathrm{c}}^{\mathrm{T}}[n] = \begin{bmatrix} \frac{1}{2} & 1 & \frac{1}{2} \end{bmatrix}$$

$$h[m,n] = \begin{bmatrix} 1/4 & 1/2 & 1/4 \\ 1/2 & 1 & 1/2 \\ 1/4 & 1/2 & 1/4 \end{bmatrix}$$



$$a = \frac{s_1 + s_2}{2}$$

$$c = \frac{a+b}{2} = \frac{s_1 + s_2 + s_3 + s_4}{4}$$

$$b = \frac{s_3 + s_4}{2}$$

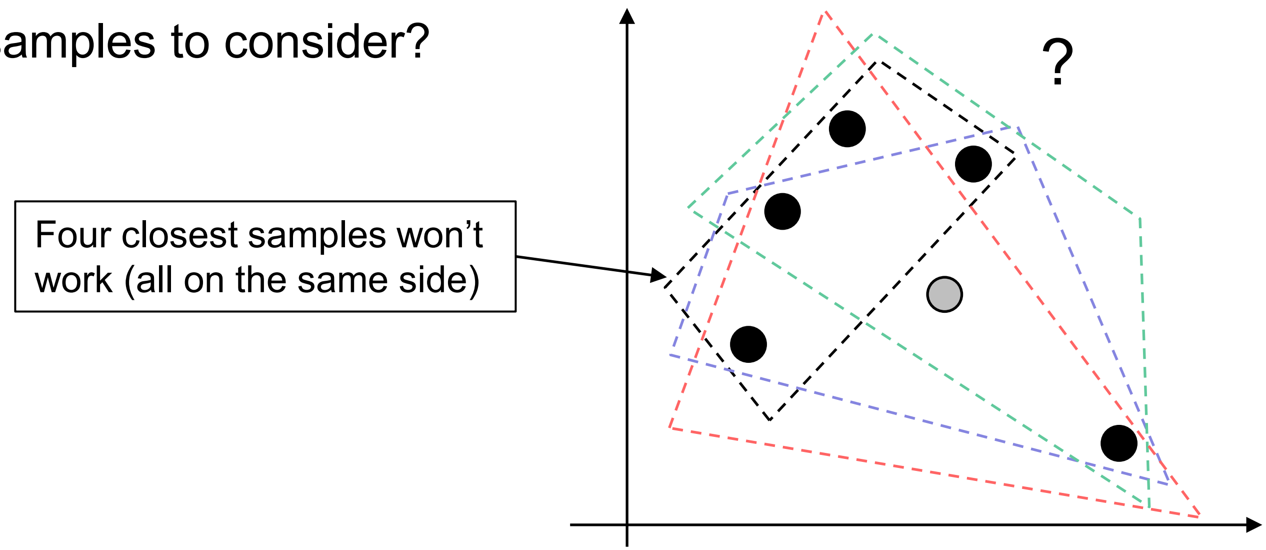
5.4 Not-equally Spaced Samples

Sample location are arbitrary

- No regularity can be assumed
- No separability

Example: bilinear interpolation

- Which 4 samples to consider?

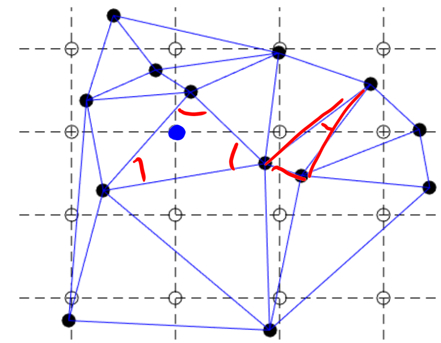


Typical solution: triangulation

Triangulation

Triangulate the available mesh of samples

- Each unknown pixel is enclosed by a triangle
- Many possible triangulations



Delaunay triangulation

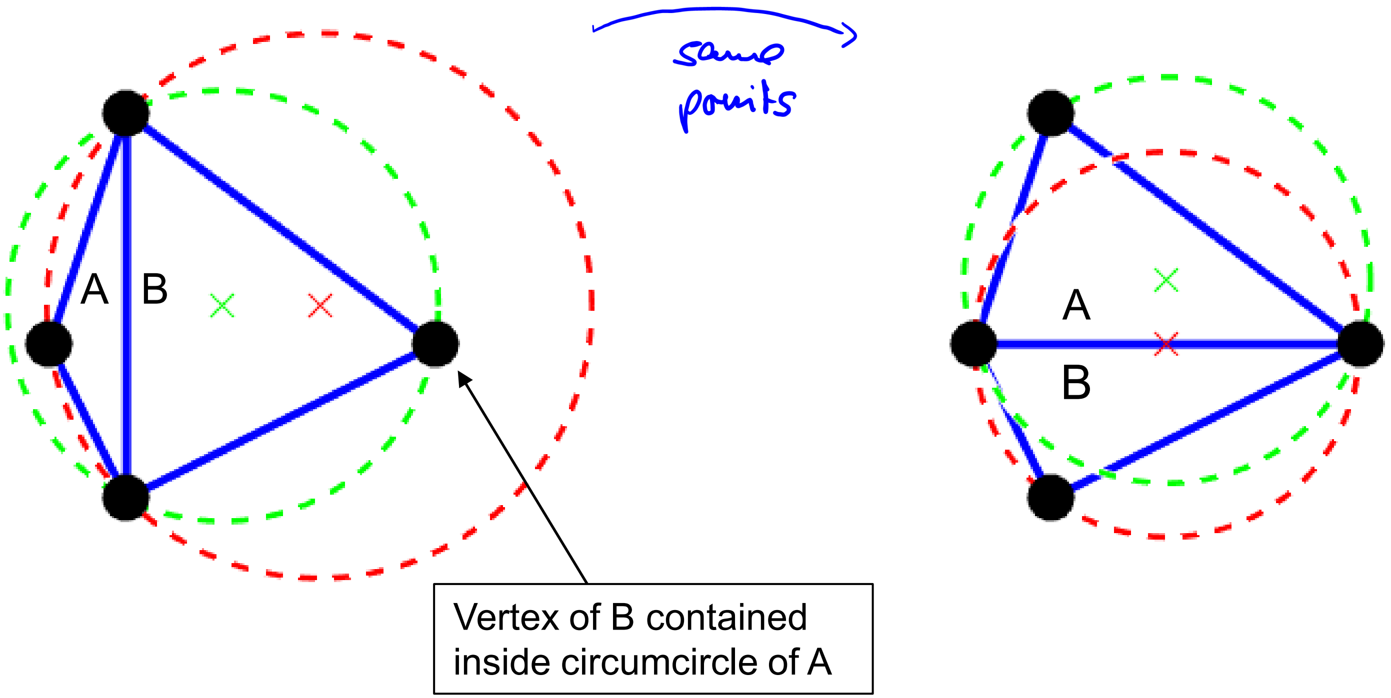
- Most commonly used
- Maximizes the minimum angle in a triangle
  - Avoids extremely thin (degenerated) triangles
- Circumcircle of a triangle contains **only** its vertices (and no other samples)

Boris Nikolaevich Delaunay (1890-1980)

Delaunay Triangulation

Possible triangulation

Delaunay triangulation

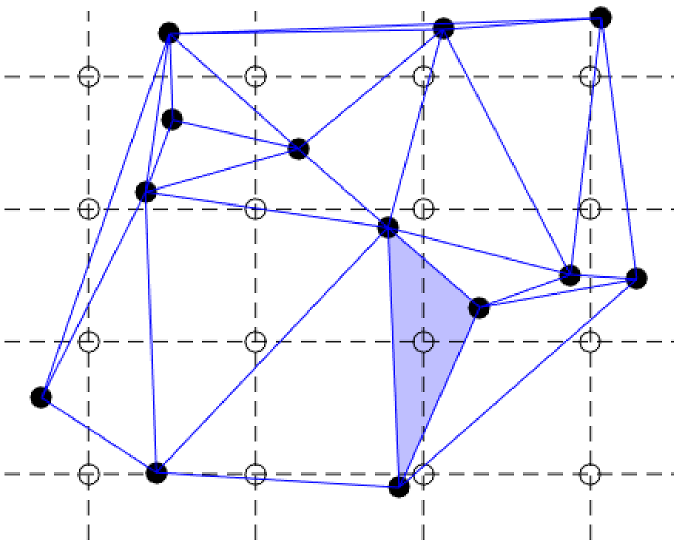


Angles in A: [135º, **27º**, **18º**]
Angles in B: [63º, 63º, 54º]

Angles in A: [63º, 90º, 27º]
Angles in B: [71º, 71º, 38º]

Linear Interpolation

Straightforward extension of 1D linear interpolation to 2D



Enclosing Delaunay triangle

- Three points define a plane (linear surface)

$$\boxed{ax + by + cz + d = 0}$$

(x, y)	sample coordinates
z	pixel value (color)

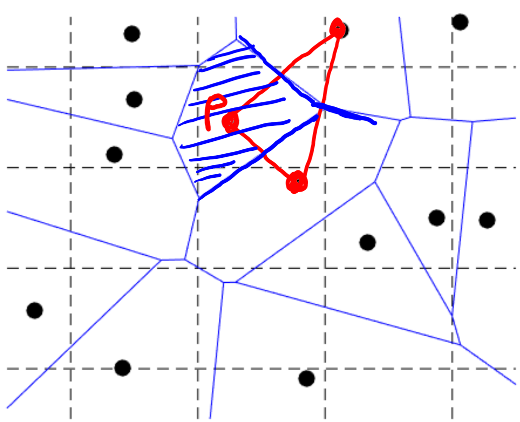
Interpolated value (z-coordinate) is obtained by evaluating plane equation at query position (x, y)

Natural Neighbor Interpolation

Based on **Voronoi tessellation**

- Dual graph of Delaunay triangulation
- Obtained by connecting centers of circumcircles (of each triangle)

≙ nearest neighbor classification (cf. Chap. 9)



Voronoi cell corresponding to sample point $\boldsymbol{p}$

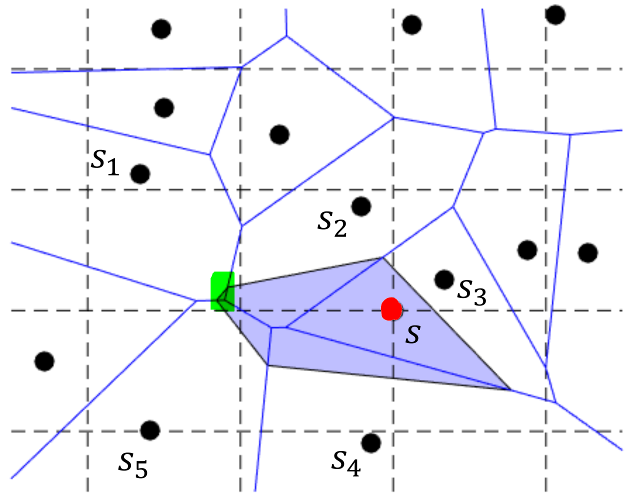
- Region where all locations are closer to $\boldsymbol{p}$ than to any other sample point

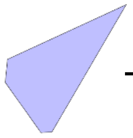
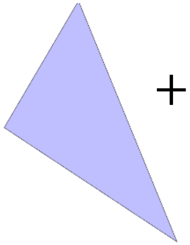
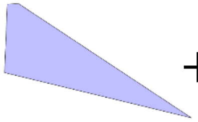
Georgy Feodosevich Voronoi (1868-1908)

Natural Neighbor Interpolation

Place Voronoi cell at query point $S \Rightarrow$ add another vertex, change tessellation accordingly

- Interpolated value s is based on (normalized) area overlap with amplitudes s_i of neighboring cells



$s = s_1 \times$  $+ s_2 \times$  $+ s_3 \times$  $+ s_4 \times$  $+ s_5 \times$