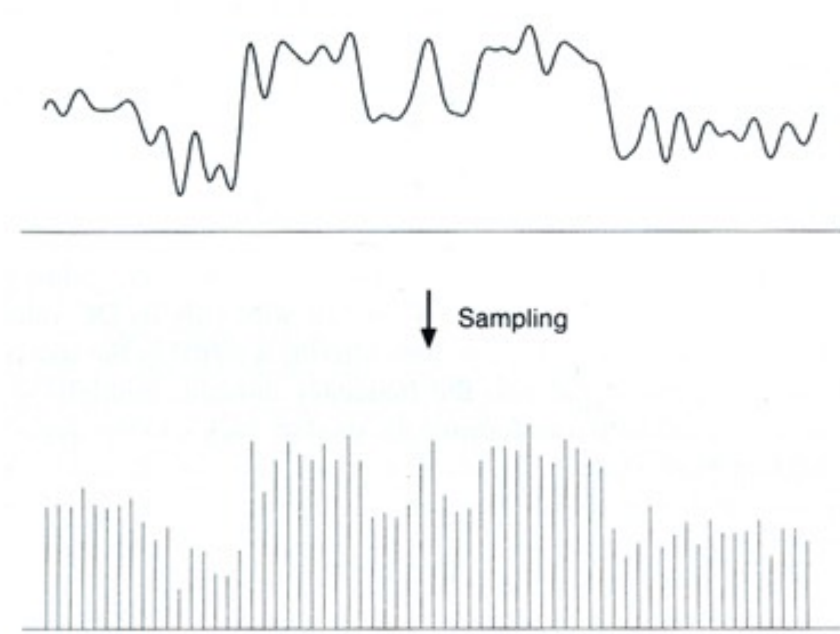
Interpolation in images

Steve Marschner
CS 4620
Cornell University

Sampled representations

- How to store and compute with continuous functions?
- Common scheme for representation: samples

  write down the function's values at many points



[FvDFH fig.14.14b / Wolberg]

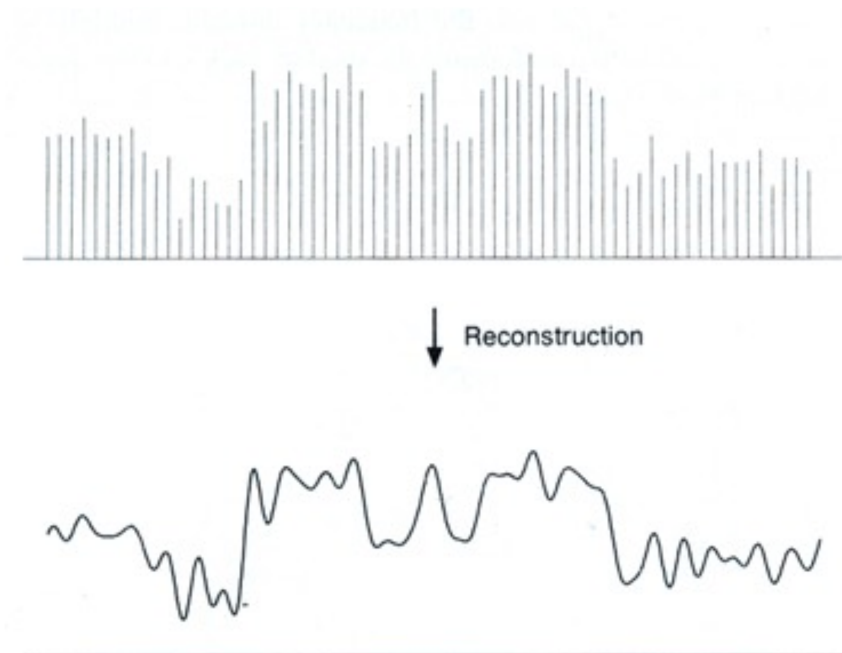
Interpolation aka. reconstruction*

- Making samples back into a continuous function

    for output (need realizable method)

    for analysis or processing (need mathematical method)

    amounts to "guessing" what the function did in between



* We'll see that technically interpolation refers to a particular kind of reconstruction but the terms are often used interchangeably in practice.

[FvDFH fig.14.14b / Wolberg]

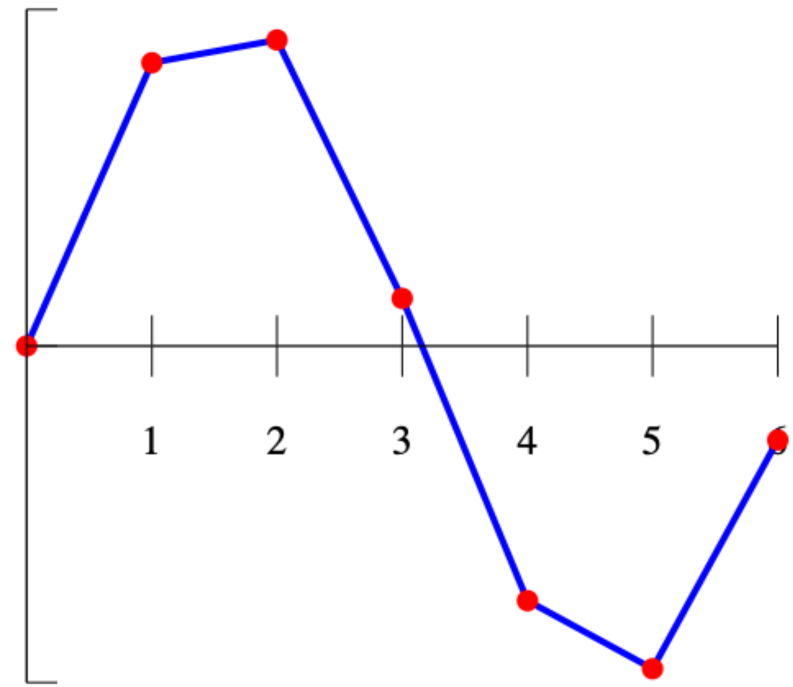
Nearest-neighbor interpolation

- Simplest, cheapest, fastest method: just round to the nearest pixel

Mojang studios | *Minecraft*

Linear interpolation

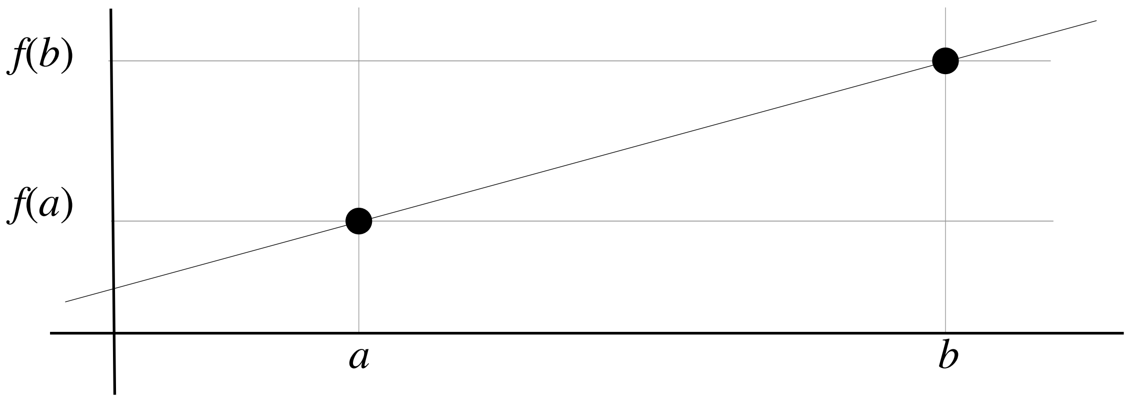
- Sometimes you want more continuity
- Linear interpolation: connect the dots with straight lines
- In math-speak: the interpolated function is a piecewise linear function; in each interval between samples it is equal to the unique linear function that interpolates the two samples at the ends
  - i.e. connect the samples with line segments!



Berland via Wikimedia Commons

Linear interpolation, 1D domain

- **Given values of a function f(x) for two values of x, you can define in-between values by drawing a line**



See textbook Sec. 2.6

  - there is a unique line through the two points
  - can write down using slopes, intercepts
  - …or as a value added to f(a)
  - …or as a convex combination of f(a) and f(b)

$$
\begin{aligned}
f(x) &= f(a) + \frac{x-a}{b-a}(f(b) - f(a)) \\
&= (1-\beta)f(a) + \beta f(b) \\
&= \alpha f(a) + \beta f(b)
\end{aligned}
$$

Linear interpolation in 1D

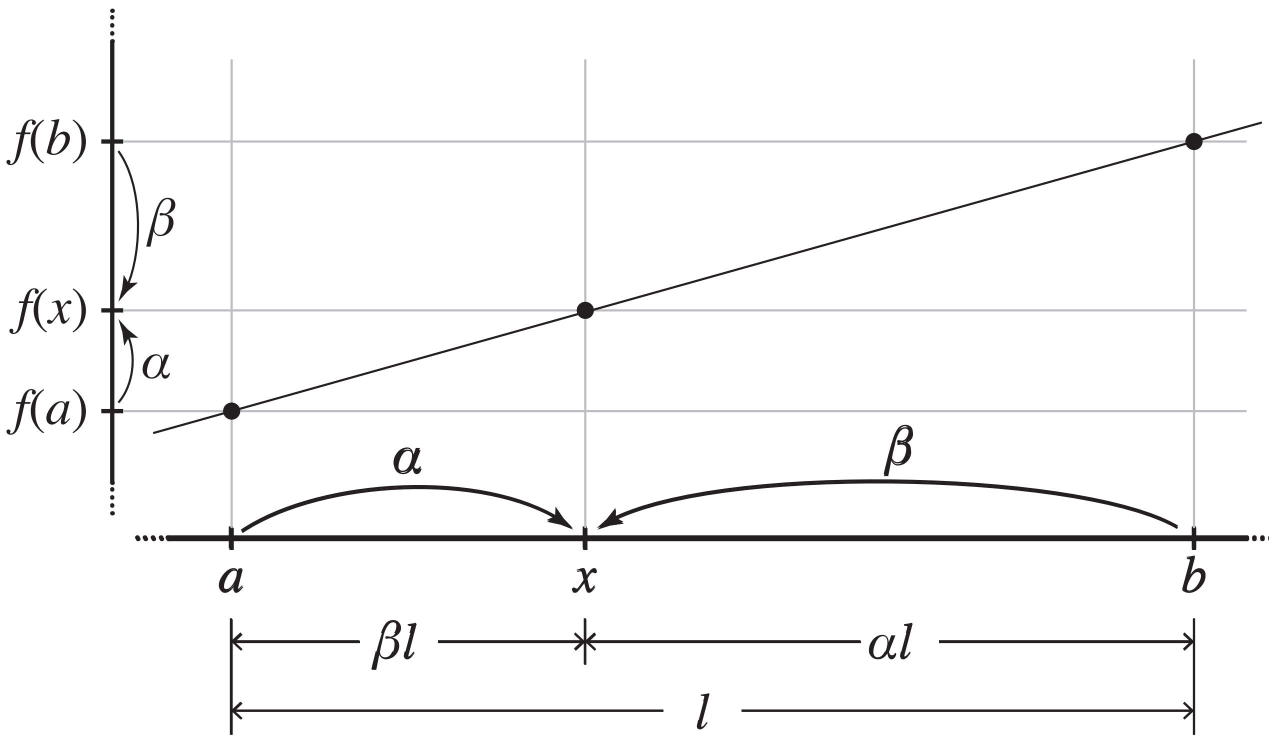
- **Alternate story**
  1. write *x* as convex combination of a and b

     $$x = \alpha a + \beta b \quad \text{where } \alpha + \beta = 1$$

  2. use the same weights to compute ***f***(***x***) as a convex combination of ***f***(***a***) and ***f***(***b***)

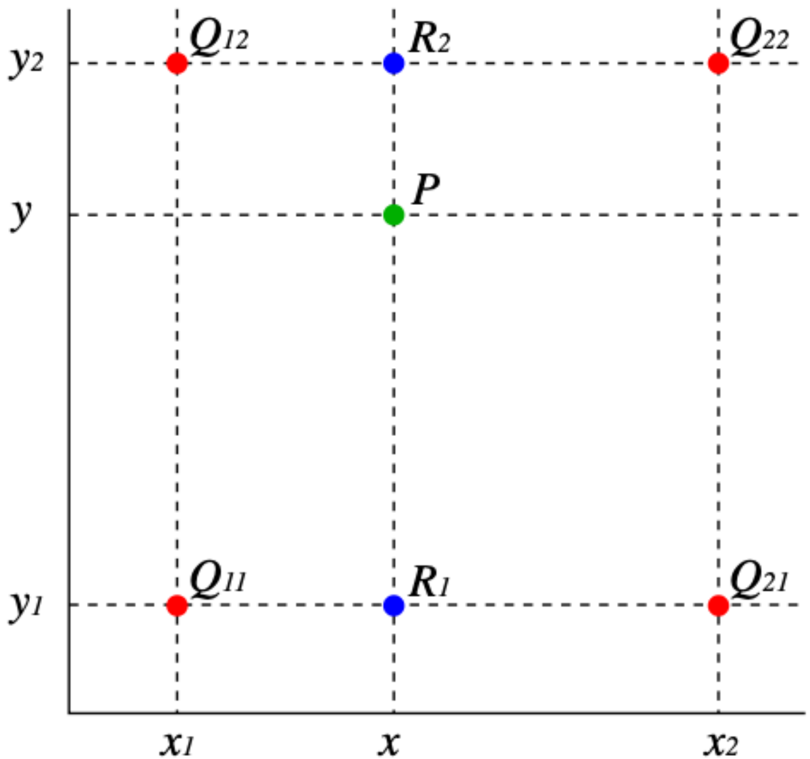
     $$f(x) = \alpha f(a) + \beta f(b)$$

Linear interpolation in 1D



Bilinear interpolation

- Linear interpolation is for 1D sequences—what about images?
- Bilinear interp. is *one way* to generalize linear interpolation to 2D



One more convolution

- Continuous–discrete convolution

$$(a \star f)(x) = \sum_i a[i] f(x - i)$$

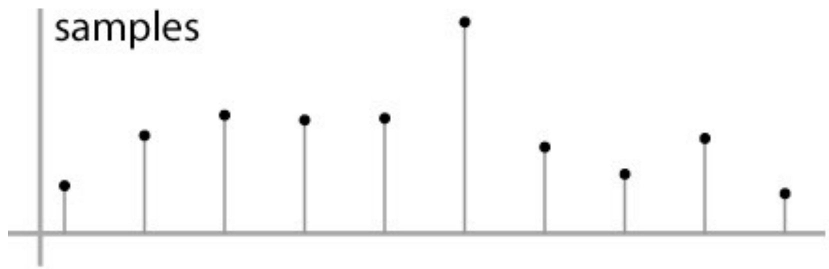
$$(a \star f)(x, y) = \sum_{i,j} a[i, j] f(x - i, y - j)$$

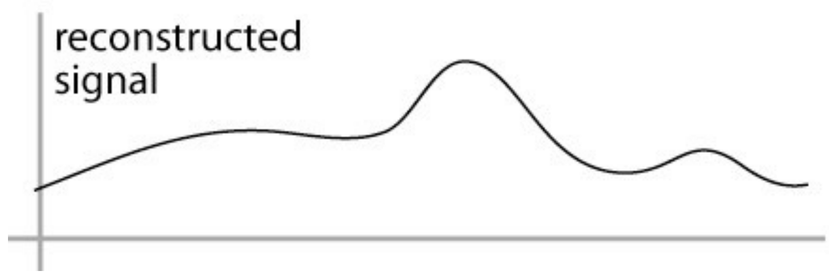
input: a sequence and a continuous function

output: a continuous function

used for reconstruction and resampling

Continuous-discrete convolution





And in pseudocode…

$$
\begin{array}{l}
\textbf{function } \text{reconstruct(sequence } a\text{, filter } f\text{, real } x\text{)} \\
\text{s} = 0 \\
\text{r} = \text{f.radius} \\
\textbf{for } i = \lceil x - r \rceil \textbf{ to } \lfloor x + r \rfloor \textbf{ do} \\
\quad s = s + a[i] f(x - i) \\
\textbf{return } \text{s}
\end{array}
$$

Cont.–disc. convolution in 2D

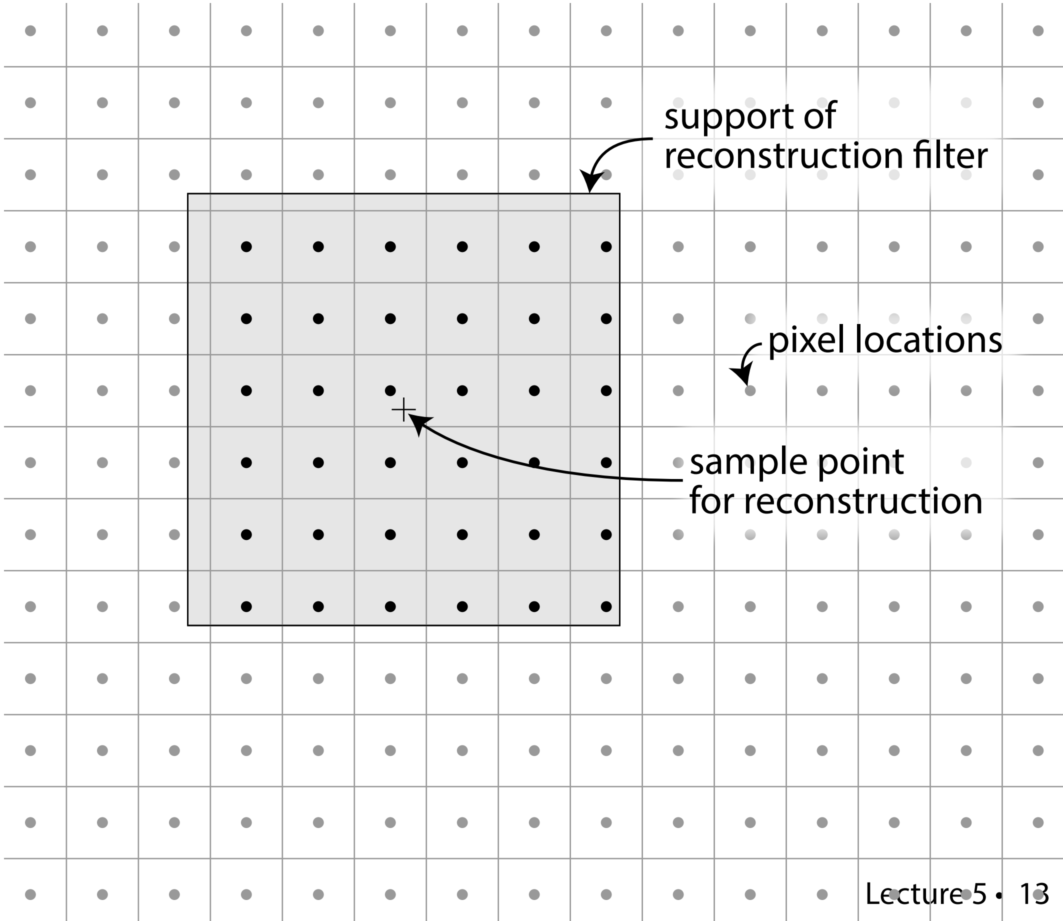
- same convolution—just two variables now

$$(a \star f)(x,y) = \sum_{i,j} a[i,j] f(x-i, y-j)$$

loop over nearby pixels,
  average using filter weight

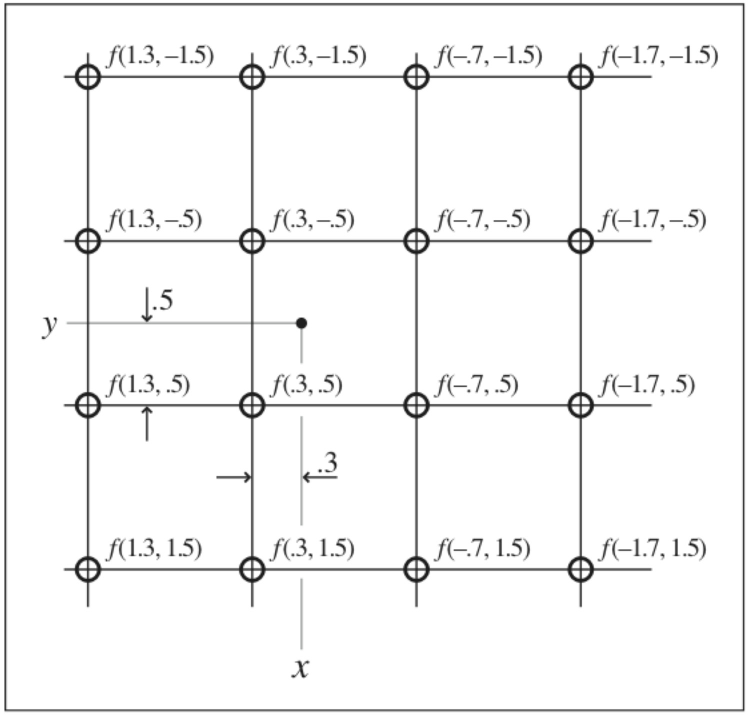
looks like discrete filter,
  but offsets are not integers
  and filter is continuous

remember placement of filter
  relative to grid is variable



Cont.–disc. convolution in 2D

$$(a \star f)(x,y) = \sum_{i,j} a[i,j] f(x-i, y-j)$$



Example showing filter weights used to compute a reconstructed value at a single point.

And in pseudocode…

$$
\begin{array}{l}
\textbf{function } \text{reconstruct}(\text{sequence } a, \text{ filter } f, \text{ real } x, \text{ real } y) \\
s = 0 \\
\textbf{for } j = \lceil y - f.r_y \rceil \textbf{ to } \lfloor y + f.r_y \rfloor \textbf{ do} \\
\quad \textbf{for } i = \lceil x - f.r_x \rceil \textbf{ to } \lfloor x + f.r_x \rfloor \textbf{ do} \\
\qquad s = s + a[i, j] f(x - i, y - j) \\
\textbf{return } s
\end{array}
$$

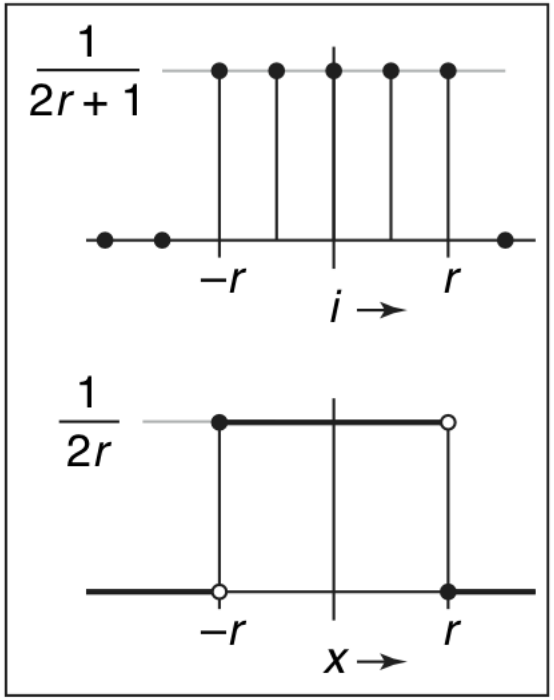
Reconstruction filter gallery

- Box filter
  - Simple and cheap
- Tent filter
  - Linear interpolation
- B-spline cubic
  - Very smooth
- Catmull-rom cubic
  - Interpolating
- Mitchell-Netravali cubic
  - Good for image upsampling
- Lanczos
  - Good for upsampling

Box filter

$$a_{\text{box},r}[i] = \begin{cases} 1/(2r+1) & |i| \le r, \\ 0 & \text{otherwise.} \end{cases}$$

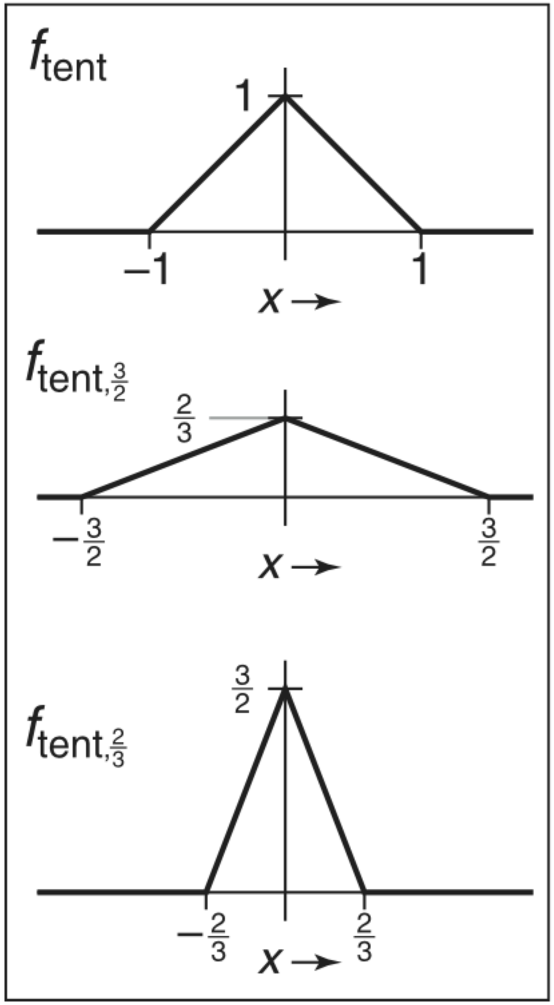
$$f_{\text{box},r}(x) = \begin{cases} 1/(2r) & -r \le x < r, \\ 0 & \text{otherwise.} \end{cases}$$



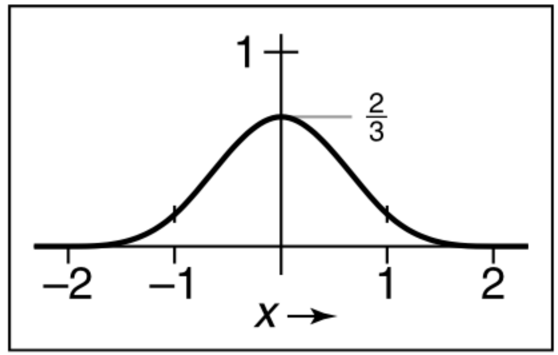
Tent filter

$$f_{\text{tent}}(x) = \begin{cases} 1 - |x| & |x| < 1, \\ 0 & \text{otherwise}; \end{cases}$$

$$f_{\text{tent},r}(x) = \frac{f_{\text{tent}}(x/r)}{r}.$$

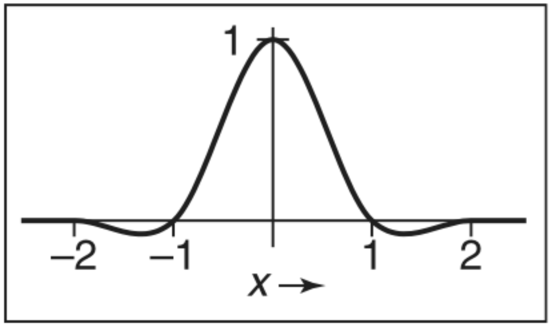


B-Spline cubic



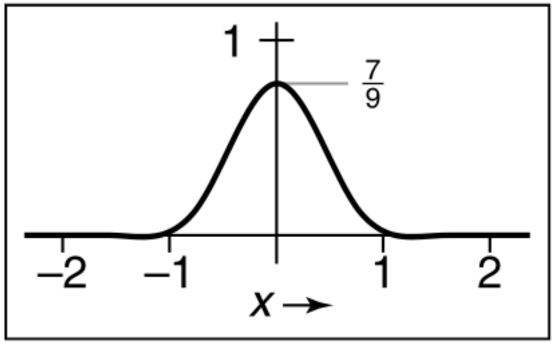
$$f_B(x) = \frac{1}{6} \begin{cases} -3(1-|x|)^3 + 3(1-|x|)^2 + 3(1-|x|) + 1 & -1 \leq x \leq 1, \\ (2-|x|)^3 & 1 \leq |x| \leq 2, \\ 0 & \text{otherwise.} \end{cases}$$

Catmull-Rom cubic



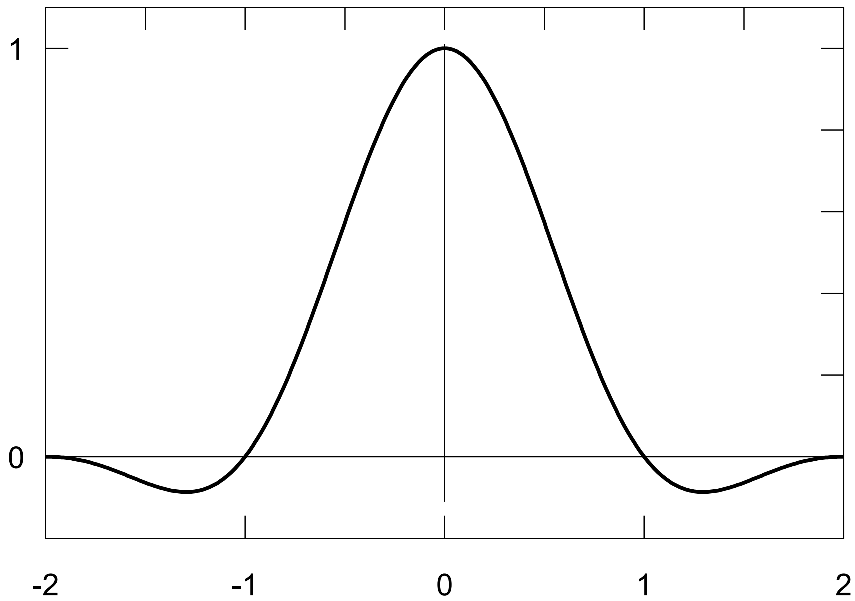
$$f_C(x) = \frac{1}{2}\begin{cases} -3(1-|x|)^3 + 4(1-|x|)^2 + (1-|x|) & -1 \le x \le 1, \\ (2-|x|)^3 - (2-|x|)^2 & 1 \le |x| \le 2, \\ 0 & \text{otherwise.} \end{cases}$$

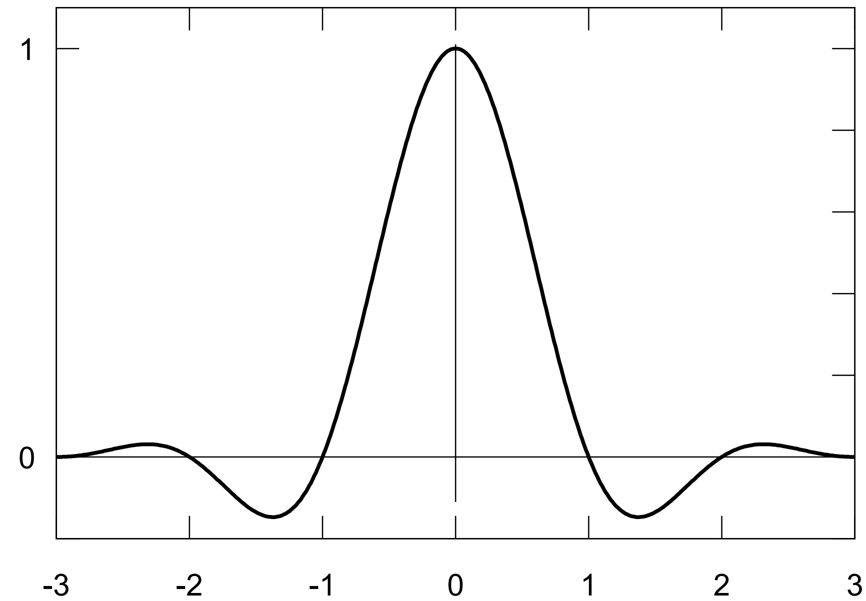
Mitchell-Netravali cubic



$$f_M(x) = \frac{1}{3} f_B(x) + \frac{2}{3} f_C(x)$$
$$= \frac{1}{18} \begin{cases} -21(1-|x|)^3 + 27(1-|x|)^2 + 9(1-|x|) + 1 & -1 \leq x \leq 1, \\ 7(2-|x|)^3 - 6(2-|x|)^2 & 1 \leq |x| \leq 2, \\ 0 & \text{otherwise.} \end{cases}$$

Lanczos





$$f_{L2}(x) = \begin{cases} \mathrm{sinc}(x)\,\mathrm{sinc}(x/2) & |x| < 2 \\ 0 & \text{otherwise} \end{cases}$$

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

$$f_{L3}(x) = \begin{cases} \mathrm{sinc}(x)\,\mathrm{sinc}(x/3) & |x| < 3 \\ 0 & \text{otherwise} \end{cases}$$

Effects of reconstruction filters

- For some filters, the reconstruction process winds up implementing a simple algorithm
- Box filter (radius 0.5): nearest neighbor sampling

  box always catches exactly one input point

  it is the input point nearest the output point

  so output[i, j] = input[round(x(i)), round(y(j))]
    x(i) computes the position of the output coordinate i on the input grid
- Tent filter (radius 1): linear interpolation

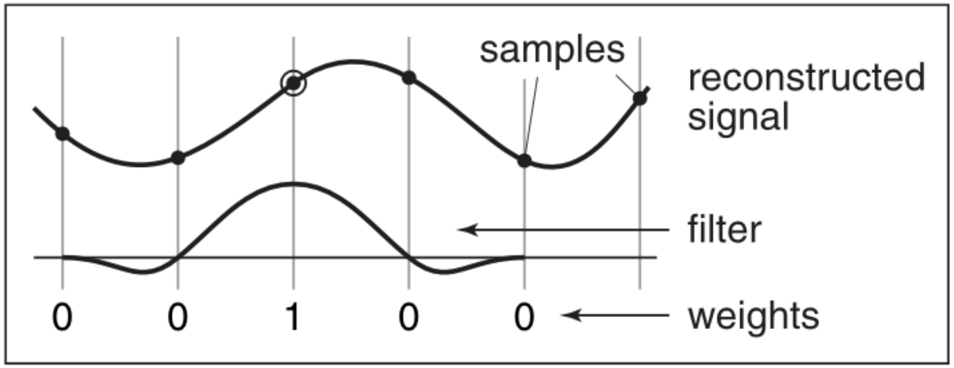
  tent catches exactly 2 input points

  weights are a and (1 – a)

  result is straight-line interpolation from one point to the next

Properties of filters

- Degree of continuity
- Impulse response
- Interpolating or no
- Ringing, or overshoot

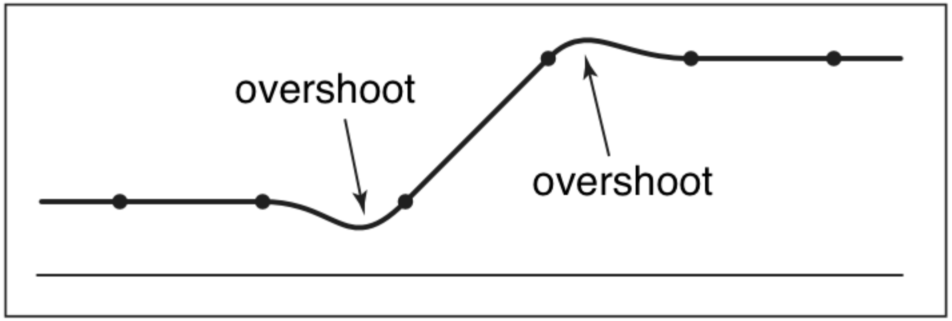


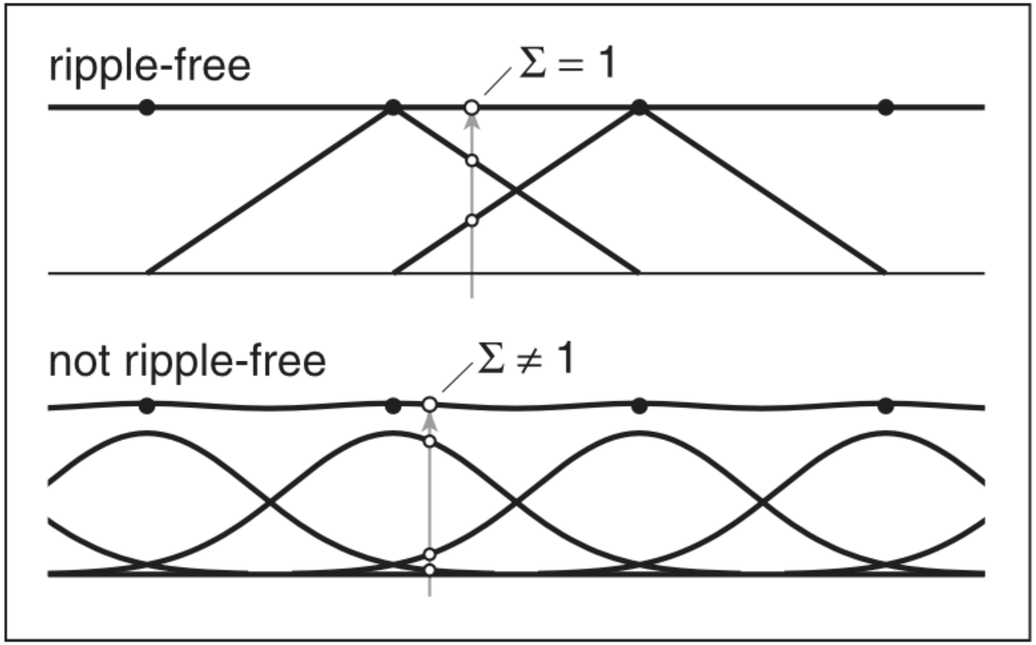
interpolating filter used for reconstruction

Ringing, overshoot, ripples

- Overshoot
  - caused by
    - negative filter values
- Ripples
  - constant in,
    - non-const. out
  - ripple free when:

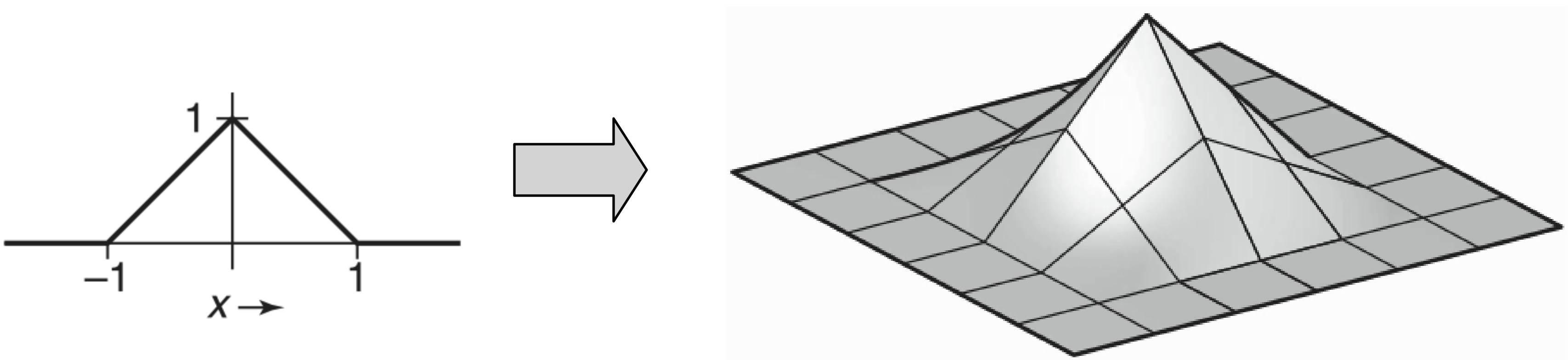
$$\sum_i f(x+i) = 1 \quad \text{for all } x.$$

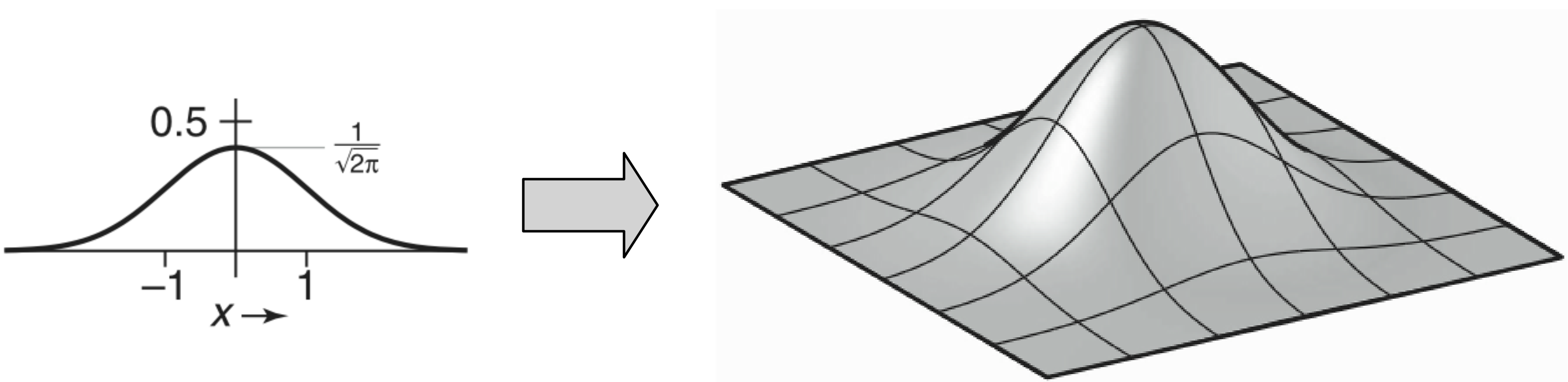




Constructing 2D filters

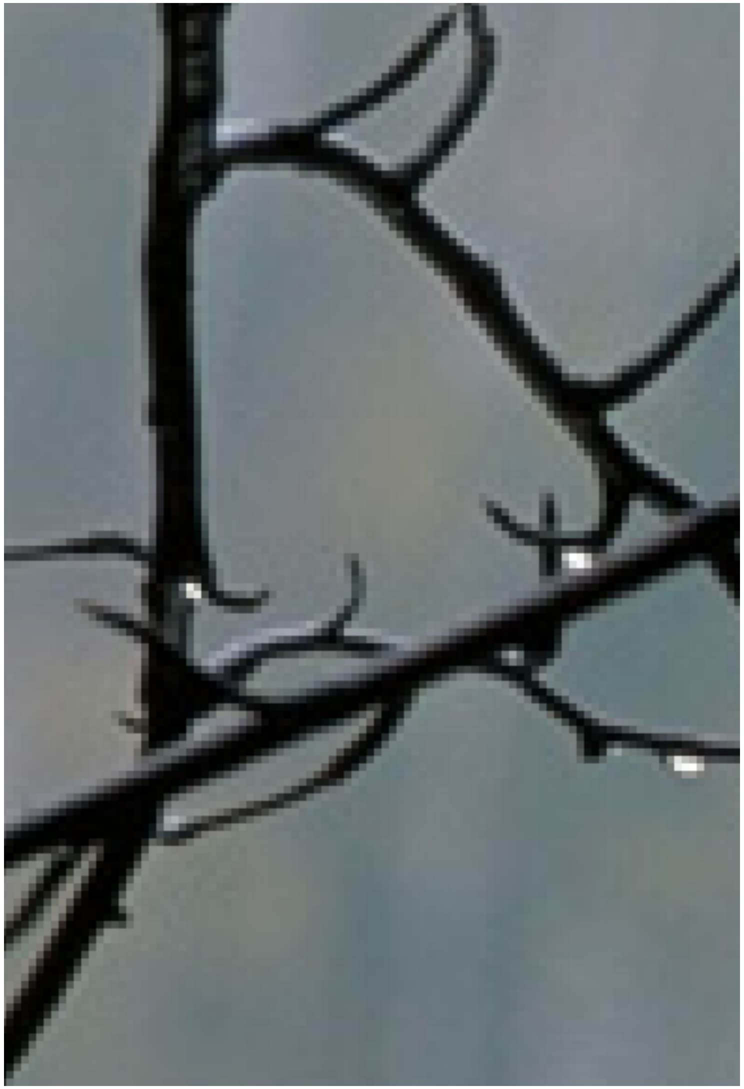
- Separable filters (most common approach)

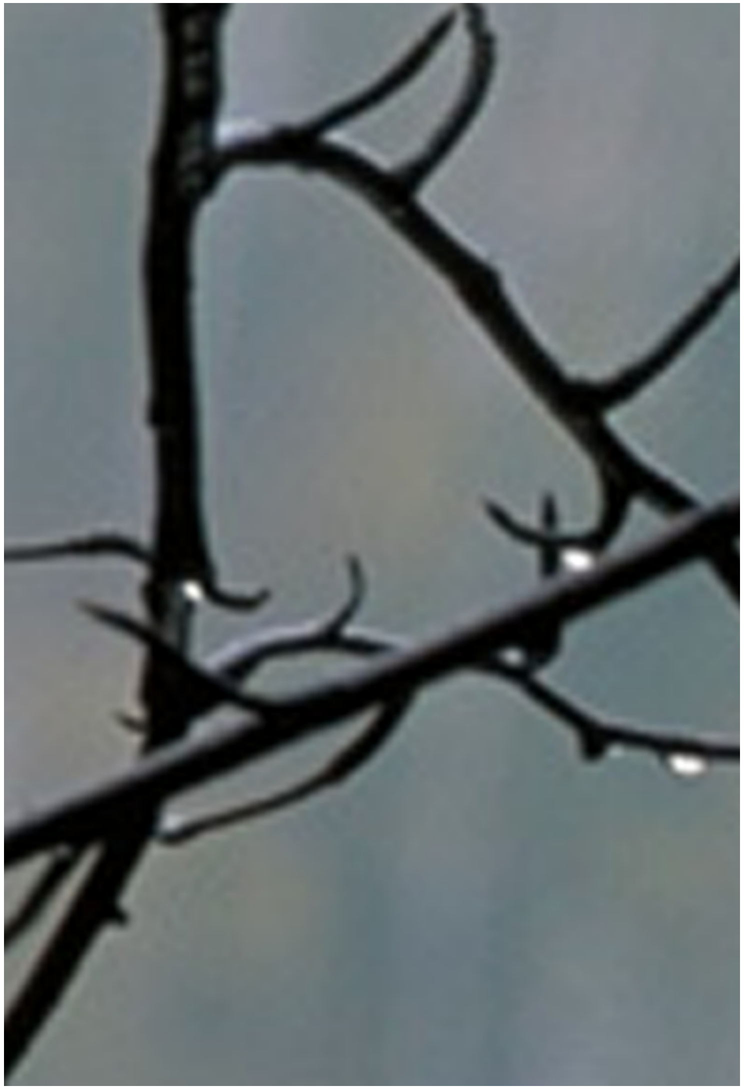




1000 pixel width

[Philip Greenspun]


box reconstruction filter


bicubic reconstruction filter

[Philip Greenspun]

4000 pixel width