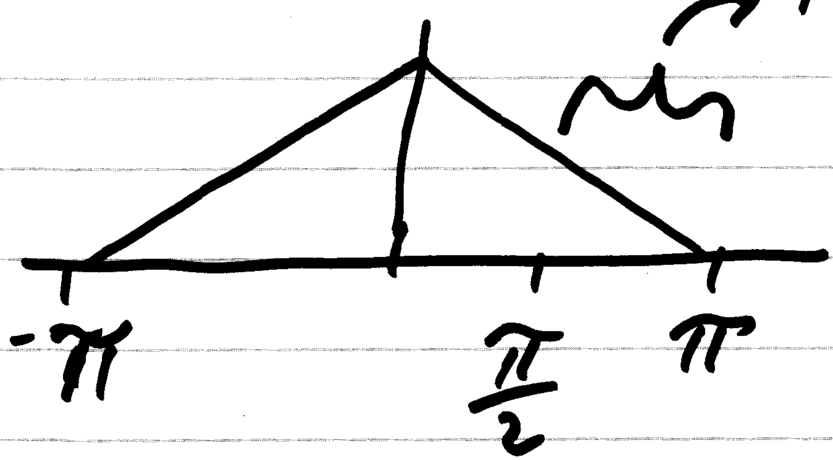
Decimation

- Need to convert an $N \times N$ image to $\left(\frac{N}{2}\right) \times \left(\frac{N}{2}\right)$

- Consider 1-D case

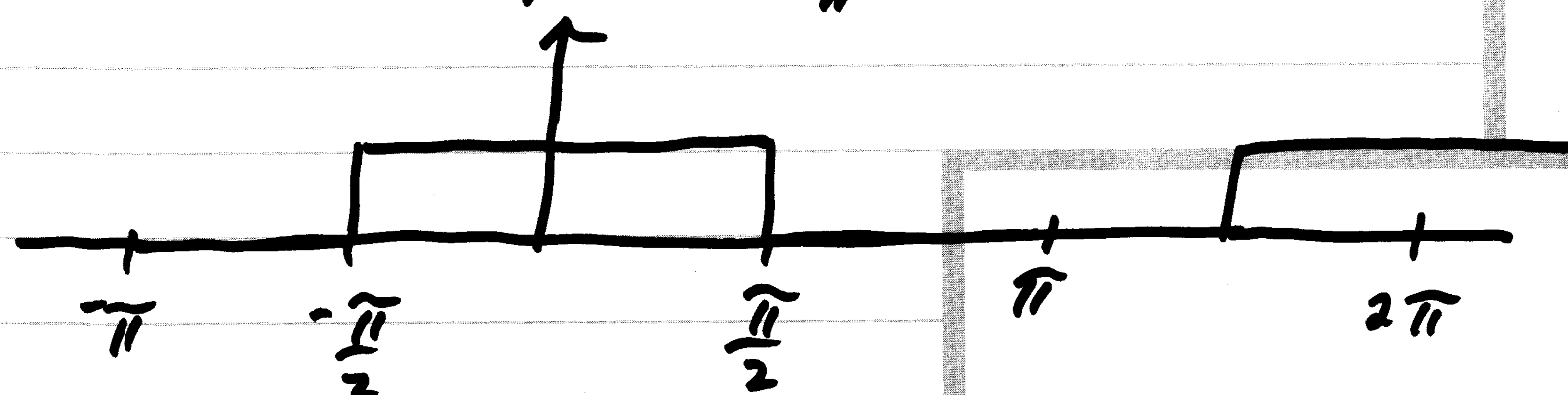
These freq.



Decimation System

$$x(n) \longrightarrow \boxed{H(e^{j\omega})} \longrightarrow \boxed{\downarrow 2} \longrightarrow y(n)$$

$$H(e^{j\omega}) = \mathrm{prect}_{\pi}(\omega)$$



$$h(n) = \frac{1}{2}\,\mathrm{sinc}(n/2)$$

Decimation by L

$$H(e^{j\omega}) = p\,\mathrm{rect}_{2\pi/L}(\omega)$$

$$h(n) = \frac{1}{L}\,\mathrm{sinc}(n/L)$$

$$x(n) \rightarrow \boxed{H(e^{j\omega})} \rightarrow \boxed{\downarrow L} \rightarrow y(n)$$

1-D Interpolation

- Need to convert $\frac{N}{2}$ point signal to an N point signal.

- Up sampling:

$$y(n) = \begin{cases} X(n/L) & \text{if } n = K \cdot L \text{ for some } K \\ 0 & \text{otherwise} \end{cases}$$

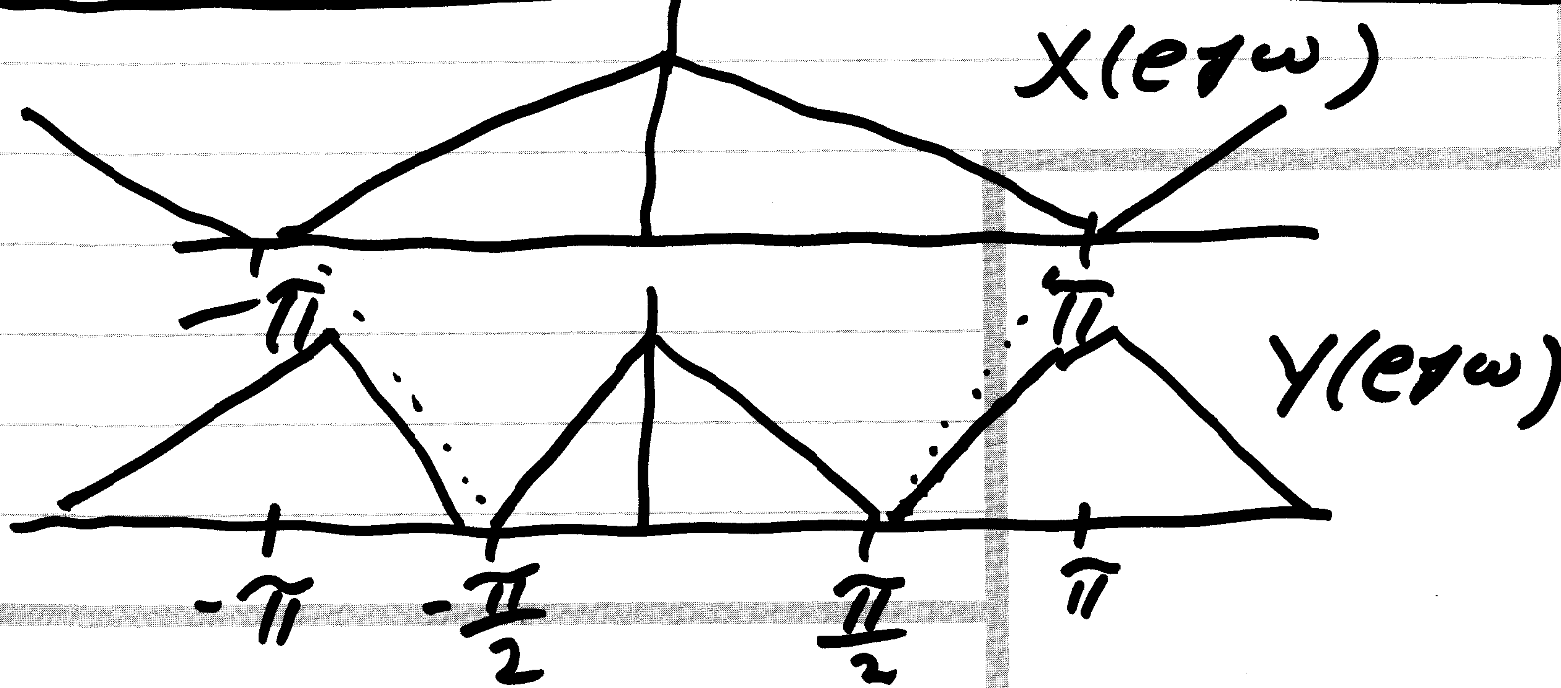
alternative:

$$y(n) = \sum_{m=-\infty}^{\infty} X(m)\,\delta(n - mL)$$

$$x(n) \rightarrow \boxed{\uparrow L} \rightarrow y(n)$$

- It can be shown

$$Y(e^{j\omega}) = X(e^{j\omega L})$$



Interpolation System

$$x(n) \rightarrow \boxed{\uparrow L} \rightarrow \boxed{H(e^{j\omega})} \rightarrow y(n)$$

$$H(e^{j\omega}) = L \cdot \mathrm{prect}_{2\pi/L}(\omega)$$

$$h(n) = \mathrm{sinc}\left(\frac{2\pi n}{L}\right)$$

Comments

- $h(n) = \frac{1}{L} \operatorname{sinc}(n/L)$

  has infinite support.

2-D Interpolation

$$y(m,n) = \sum_k \sum_\ell x(k,\ell)\,\delta(m-kL,\; n-\ell L)$$

$$H(e^{j\mu}, e^{j\nu}) = L^2\, \mathrm{prect}_{2\pi/L}(\mu) \cdot \mathrm{prect}_{2\pi/L}(\nu)$$

$$x(m,n) \rightarrow \boxed{\uparrow(L,L)} \xrightarrow{y(m,n)} \boxed{H(e^{j\mu}, e^{j\nu})} \rightarrow z(m,n)$$

Comments

1) "Ideal" Interpolation uses a 2-D sinc function interpolation filter

$$h(m,n) = \text{sinc}(m/L,\ n/L)$$

2) Infinite support.
$\Rightarrow$ high computation.

3) Causes ringing.

Common ~~data~~ Solution

1) Choose (Pixel Replication)

$$h(m,n) = \begin{cases} 1 & 0 \le m \le L-1 \text{ and } 0 \le n \le L-1 \\ 0 & \text{otherwise} \end{cases}$$

$L=3$

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & \boxed{1} & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$$

{ pixel replication

Low Res $\qquad\qquad\qquad\qquad$ **$L=2$**

$$\begin{matrix} a & b \\ c & d \end{matrix} \quad \xRightarrow{\boxed{\uparrow(2,2)}} \quad \begin{matrix} a & 0 & b & 0 \\ 0 & 0 & 0 & 0 \\ c & 0 & d & 0 \\ 0 & 0 & 0 & 0 \end{matrix}$$

$$\Big\Downarrow * h(m,n)$$

$$\begin{matrix} a & a & b & b \\ a & a & b & b \\ c & c & d & d \\ c & c & d & d \end{matrix}$$

2) Bilinear Interpolation

$$h(m,n) = \Lambda(m/2)\,\Lambda(n/2)$$

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1/4 & 1/2 & 1/4 & 0 \\ 0 & 1/2 & \boxed{1} & 1/2 & 0 \\ 0 & 1/4 & 1/2 & 1/4 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$$

After up-sampling

0	0	0	0	0
0	a	■	b	0
0	0	$\boxed{0}$	0	0
0	c	0	d	0
0	0	0	0	0

computing ■

$\frac{1}{2}a + \frac{1}{2}b$
$= \text{output}$

computing □

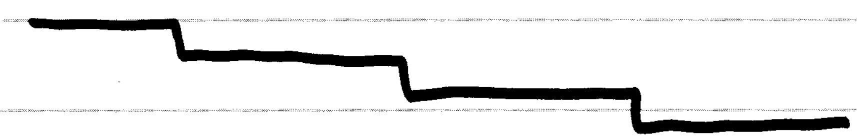
$\frac{1}{4}a + \frac{1}{4}b + \frac{1}{4}c$
$+ \frac{1}{4}d$

$= \text{output}$

Comments

1) Both Pixel Replication and bilinear interpolation ~~ea~~ will produce

a) Aliasing



b) Softness or blurring

2) Block Replication ⟹ Aliasing

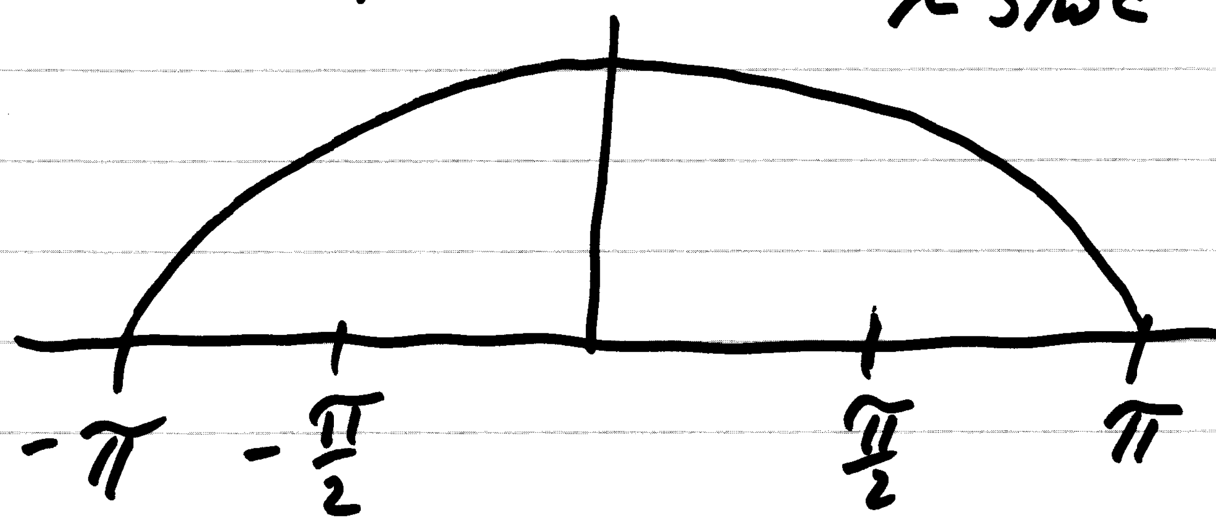
3) Bilinear ⟹ Blurry

4) Use sharpening as post-processing step

L=2

① Pixel Rep

$\approx$ sinc



Bilinear