Review

Continuous real image

Digitization (e.g. scanning) — sampling

square array of numbers (abstract pixels)

each physical pixel covers an area — reconstruction

display (physical pixels)

The eye blurs pixels into continuous image

perceived image

Review, Fourier transform

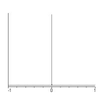
Decomposes into frequency components

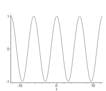
$$X(\omega) = \mathcal{F}[x](\omega) = \int_{-\infty}^{\infty} x(t)e^{-i\omega t}dt$$

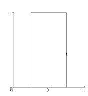
Inverse transform reconstructs the function

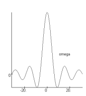
$$x(t) = \mathcal{F}^{-1}[X] = \frac{1}{2\pi}\int_{-\infty}^{\infty} X(\omega)e^{i\omega t}d\omega$$

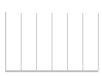
Review, FT pairs

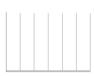

delta function

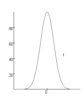

wave

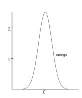

box


sinc


comb


comb


Gaussian


Gaussian

Review, convolution

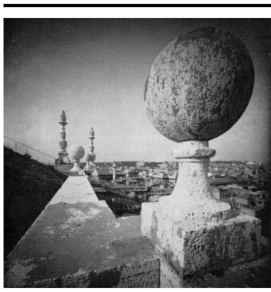
Convolution with a box = local averaging

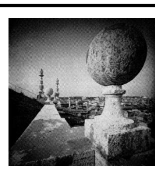
multiplication in the frequency domain is convolution in the spatial domain:

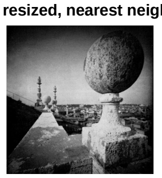
$$U(\omega) = X(\omega)Y(\omega)$$

$$u(t) = \int_{s=-\infty}^{+\infty} x(s)y(t-s)ds$$

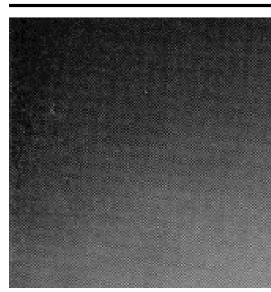
Shrinking


original


resized, nearest neighbor


resized, 11-point filter

Shrinking


original


resized, nearest neighbor


resized, 11-point filter

Image transformation

continuous original $x(t)$

sampling

often no control
(done by scanner, camera, etc.)

discrete $x[n]$

Can choose

discrete transformed $y[n]$

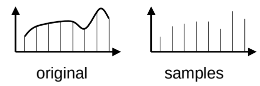
reconstruction

no control (display + eye)

Continuous reconstructed $y(t)$

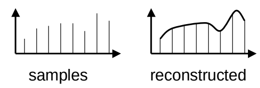
Sampling and reconstruction

Sampling = multiplication by comb



$\hat{x}(t) = comb(t)x(t)$

Reconstruction = convolution with sinc



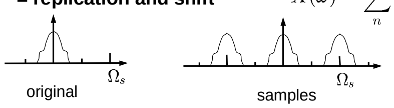
$x(t) = sinc(t) * \hat{x}(t)$

Samping and reconstruction

Frequency domain

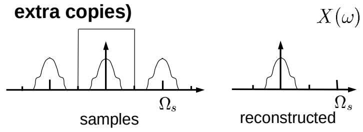
Sampling = convolution with comb
= replication and shift

$$\hat{X}(\omega) = \sum_n X(\omega - n\Omega_s)$$



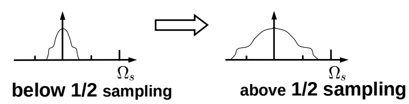
Reconstruction = multiplication by box (get rid of extra copies)

$$X(\omega) = box(\omega)\hat{X}(\omega)$$

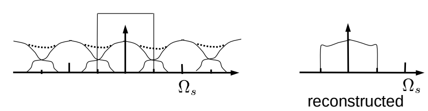


Shrinking: problem

Shrinking by a factor *a*
in freq. domain becomes stretching by *a*

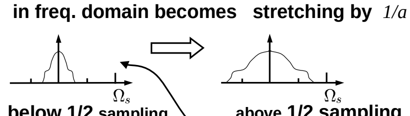


Won't be able to reconstruct correctly
= won't see the expected image



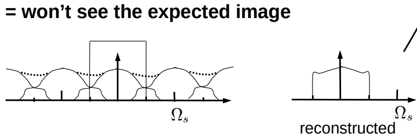
Shrinking:problem

Shrinking by a factor *a < 1*
in freq. domain becomes stretching by *1/a*



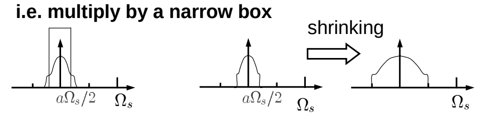
Won't be able to reconstruct correctly
= won't see the expected image

compare
to the original

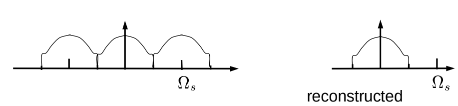


Shrinking: solution

Theoretical solution:
BEFORE shrinking, remove high frequencies,
i.e. multiply by a narrow box



Now, there is no overlap, can reconstruct
(= see the right thing)



Shrinking: discrete form

To make it practical, have to reduce to discrete operations

original samples

reconstruct continuous
↓
eliminate high frequency
↓
shrink
↓
sample

resulting samples

Combine into a single discrete operation

Shrinking: discrete form

original samples

reconstruct continuous
↓
eliminate high frequency
↓
shrink
↓
sample

resulting samples

Freq. domain:

$$\left(a\mathrm{box}(\omega)\hat{X}(a\omega)\right) * \Omega_s \mathrm{comb}(\omega)$$

spatial domain (after some calculations):

$$y[n] = a \sum_{m=-\infty}^{\infty} x[m] h(T(n - am))$$

$$h = \frac{1}{2\pi} sinc(\frac{\Omega_s t}{2\pi})$$

Filters

Problem: infinite summation!

$$y[n] = a \sum_{m=-\infty}^{\infty} x[m] h(T(n - am))$$

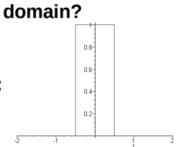
Need to approximate h with finite filters

Some choices:
- **box (fast and simple, low quality)**
- **hat (better)**
- **Lanczos, Mitchell, cubic interpolation...**

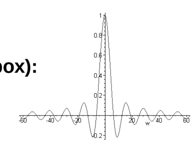
Filters

Theoretical quality criterion: how close to box in freq. domain?

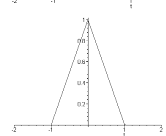
box:



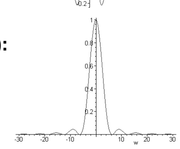
FT(box):



hat:



FT(hat):



Filters: properties

Other important properties:
- **symmetric**
- **integrates to 1 (preserve brightness)**
- **short (efficiency)**

Generalization to 2D:
- **just take h(x)h(y);**
- **do all operations twice, once for vertical, once for horizontal direction**

Filters: boundary

Real signals and images are finite.

What do we do at the boundary?

- **everything outside is zero (may get darker edges)**
- **pixels outside is the same as on the boundary**
- **reflect the image across boundaries**