Part I : Sampling & quantization

1. Discretization of continuous signals
2. Signal representation in the frequency domain
3. Effects of sampling and quantization

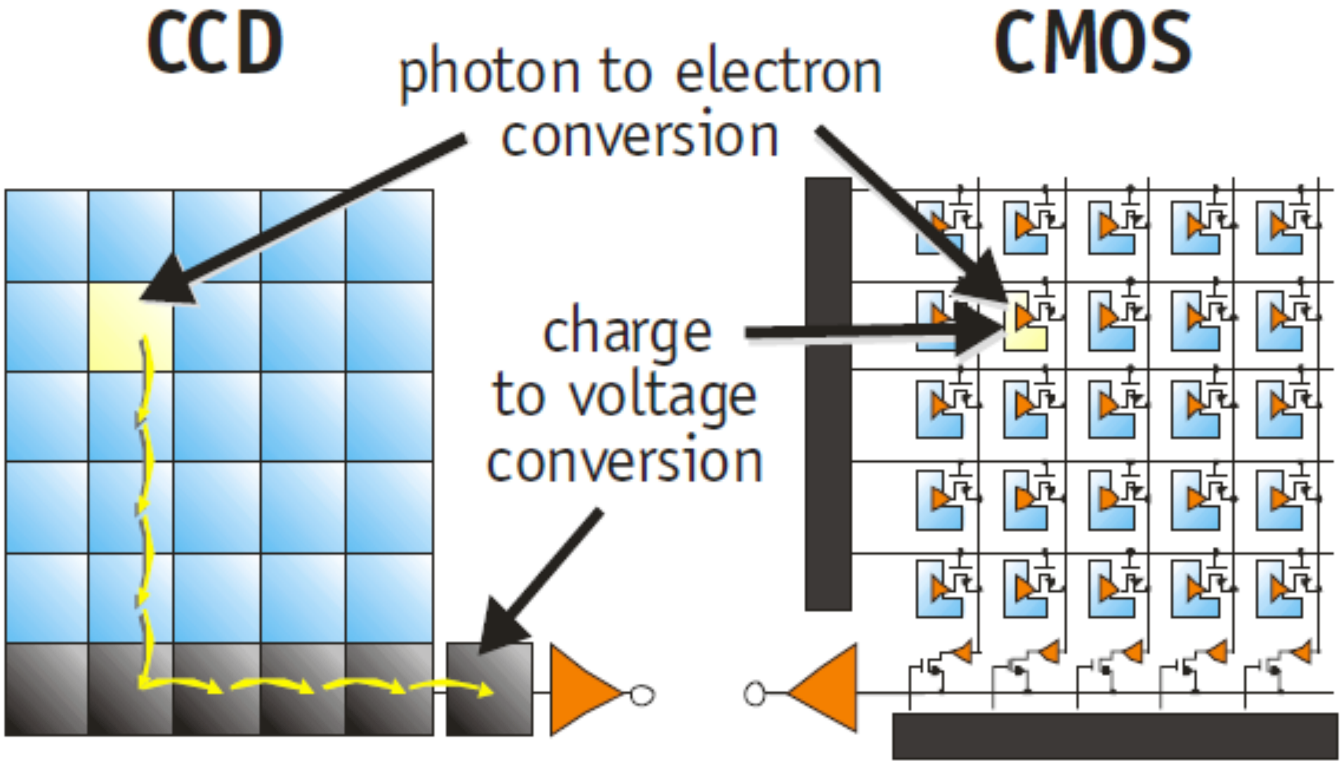
Part II : Image enhancement

1. Noise suppression
2. De-blurring
3. Contrast enhancement

Part I
Sampling
and
Quantization

Computer Vision

Recall cameras



CCD = Charge-coupled device

CMOS = Complementary Metal Oxide Semiconductor

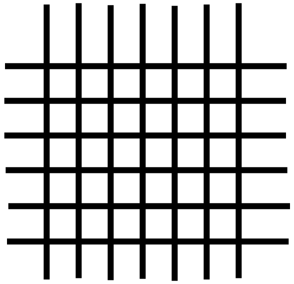
Computer Vision

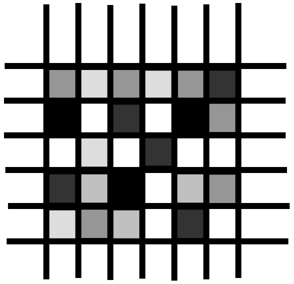
Discretization

Computer to process an image :

1. sampling ▸ "pixels"

2. quantisation ▸ "grey levels"





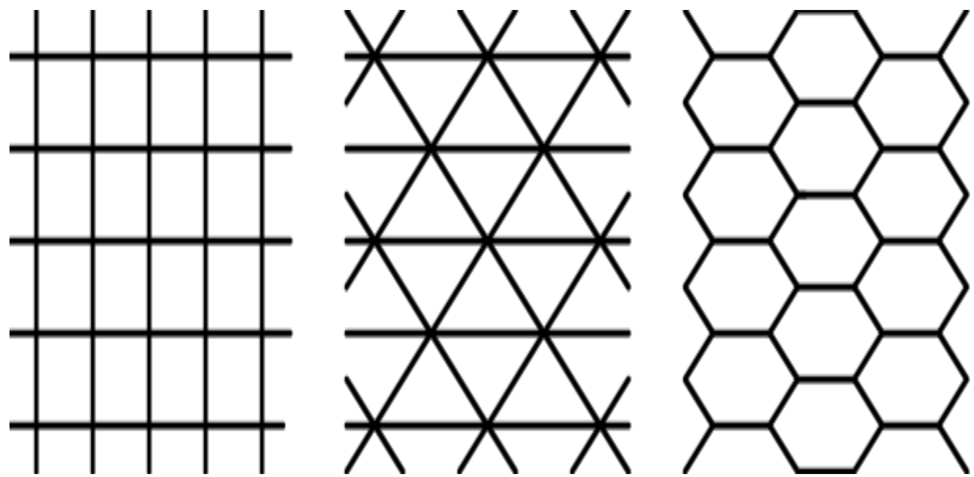
Sampling & quantization

84	133	226	212	218	218	222	212	218	222	226	218
75	156	177	218	212	218	218	218	218	222	218	218
96	84	133	203	218	218	218	222	212	218	222	218
123	75	111	156	212	218	212	212	218	218	218	226
93	75	71	133	185	231	226	226	222	212	218	218
51	75	75	75	156	206	218	218	218	222	212	222
44	110	75	65	143	194	231	218	218	218	218	218
52	123	69	84	60	156	199	231	231	222	226	226
52	75	84	81	65	69	150	231	231	226	231	231
36	36	84	93	84	71	156	160	240	240	231	231
36	40	113	75	69	75	71	133	194	240	240	240
52	52	105	85	69	75	75	123	111	222	231	231
69	44	69	93	81	75	75	69	150	177	247	240
73	44	40	96	101	75	75	75	84	133	231	240

Sampling schemes

regular, image covering tessellation

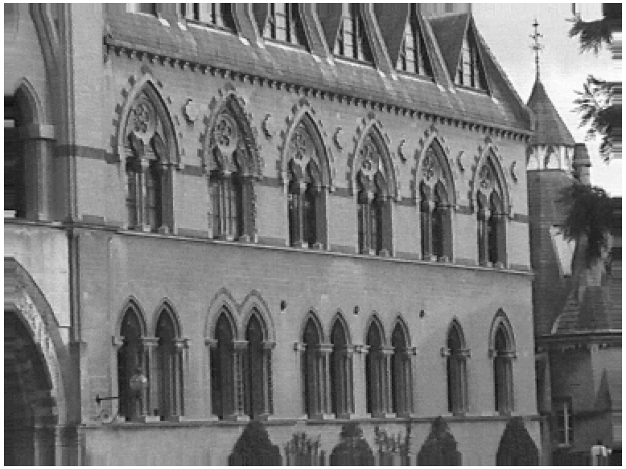
11 with regular polygons ‣ 3 if equal



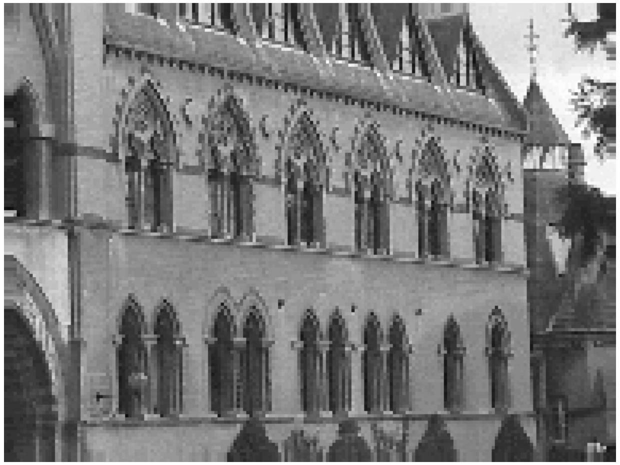
rectangular (square) most popular

hexagonal has advantages (more isotropic, no connectivity ambiguities, …) + similar structure in retina

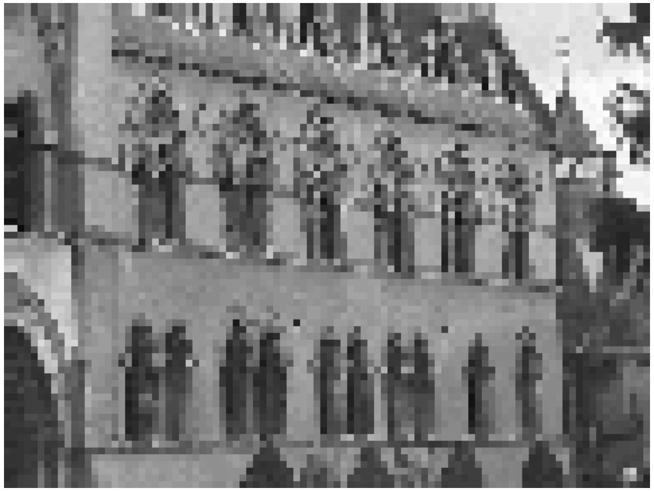
Example of sampling :



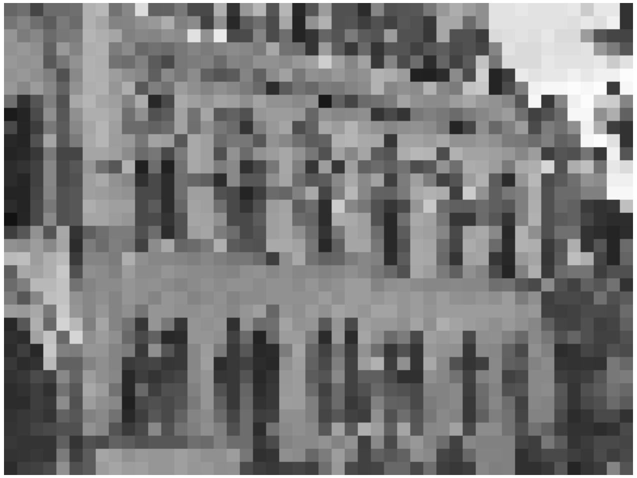
384 x 288 pixels



192 x 144 pixels



92 x 72 pixels

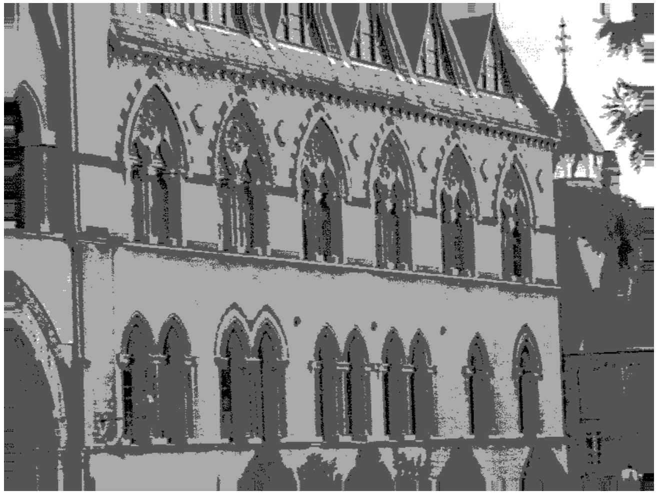


48 x 36 pixels

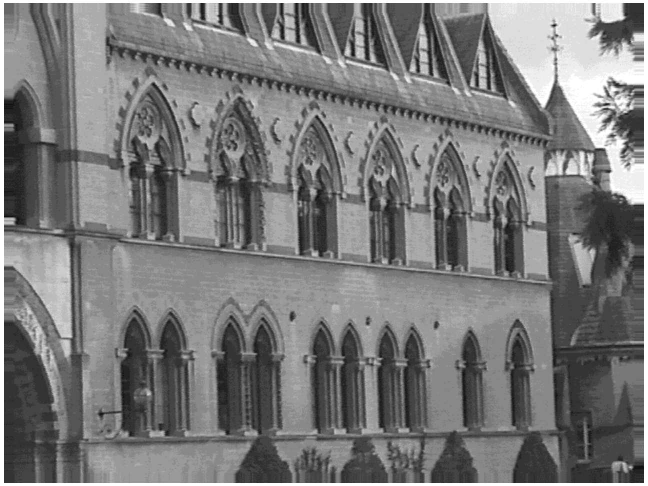
Computer Vision

Example of quantisation :

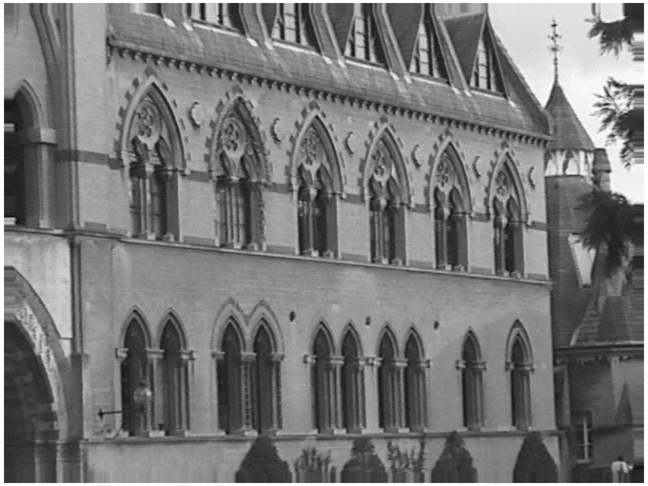
2 levels - binary



4 levels



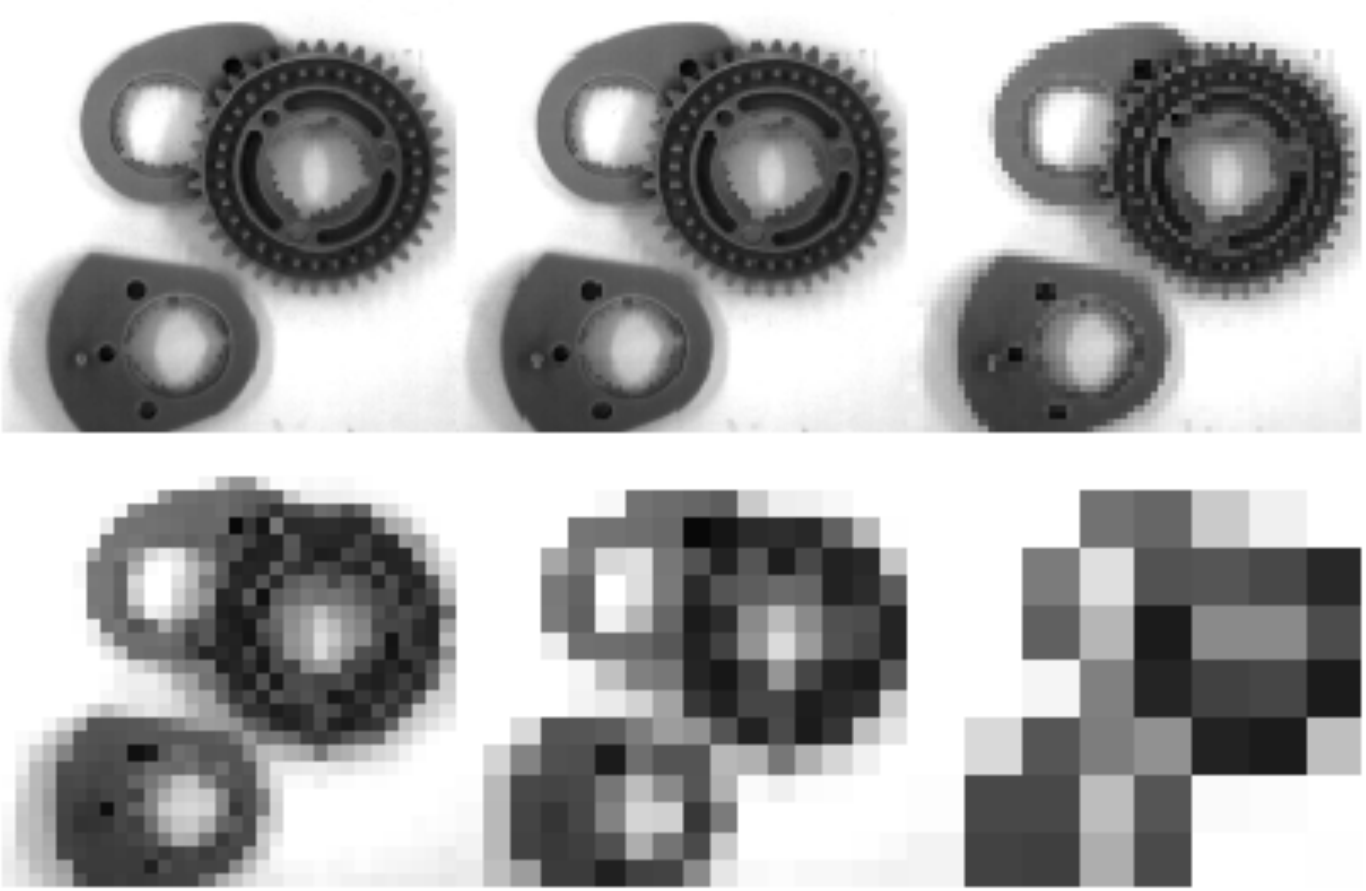
8 levels



256 levels – 1 byte

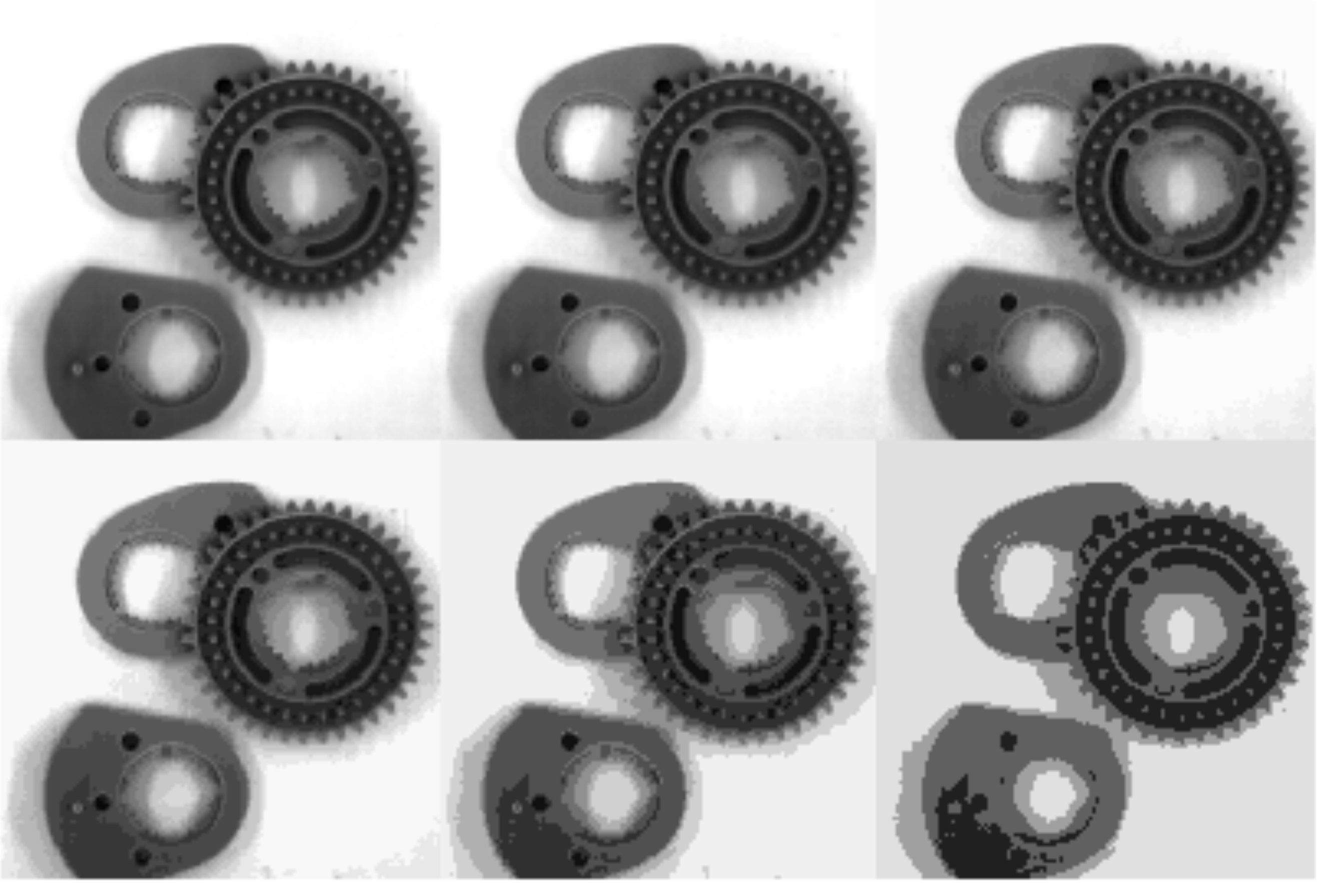
Computer Vision

Image distortion through sampling



Computer Vision

Image distortion through quantisation



Remarks

1. Binary images – 1-bit quantization – useful in industrial applications

2. Non-uniform sampling and/or quantization
   a. fine sampling for details
   b. fine quantization for homogeneous regions

A model for sampling

1. Integrate brightness over cell window

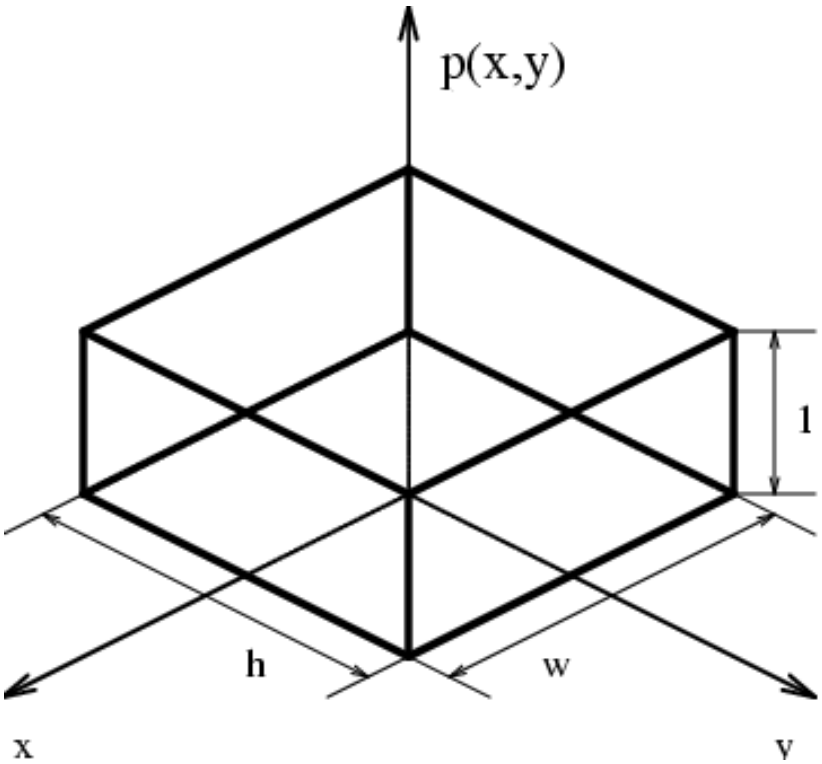
Image degradations

2. Read out values only at the pixel centers

Aliasing
Leakage

Computer Vision

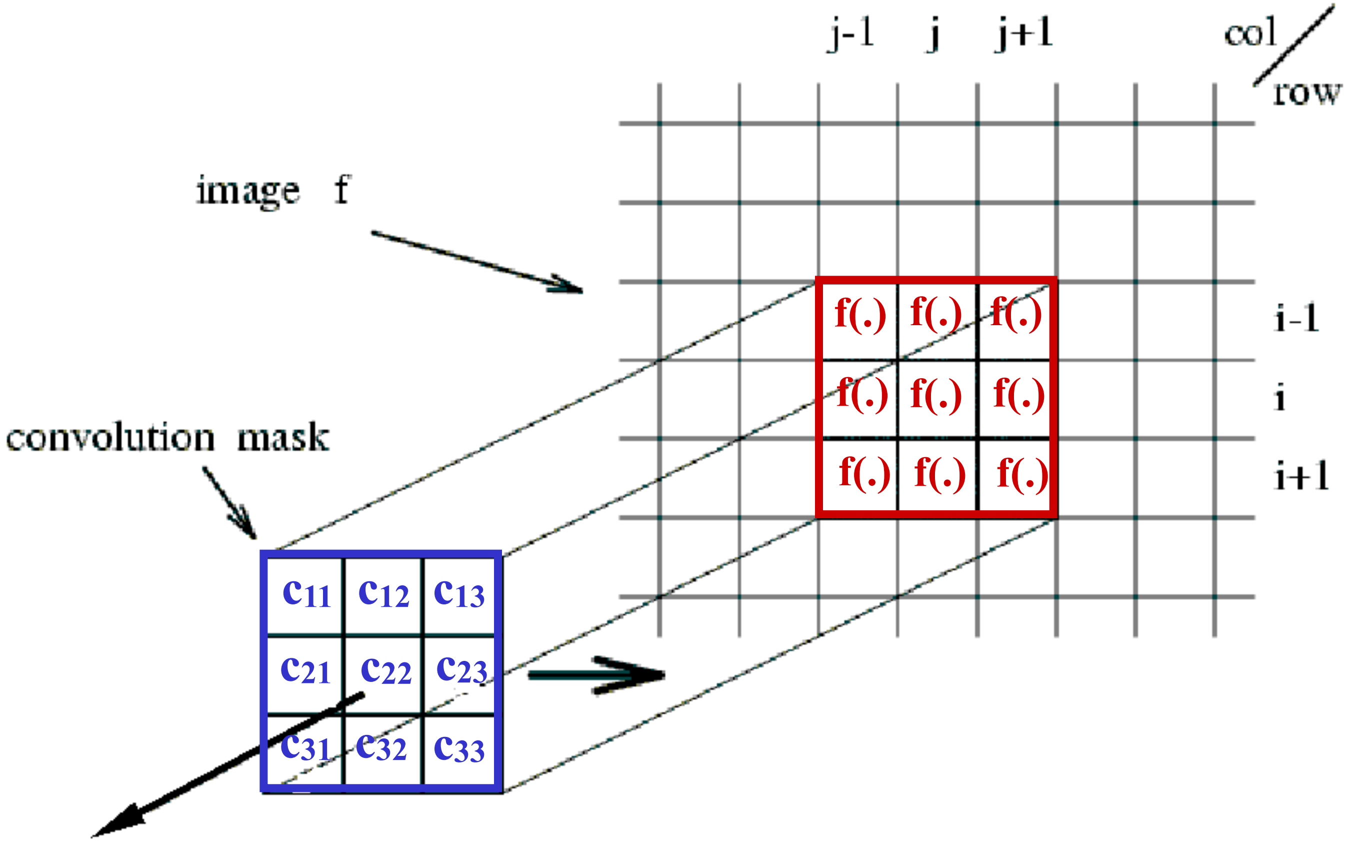
STEP 1 : integrating over a pixel cell



$$o(x',y') = \iint i(x,y)\,p(x-x',y-y')\,dx\,dy$$

This is a *convolution:* $i(x,y) * p(-x,-y)$

Convolution



$$o(i,j) = c_{11}\, f(i-1,j-1) + c_{12}\, f(i-1,j) + c_{13}\, f(i-1,j+1) + c_{21}\, f(i,j-1) + c_{22}\, f(i,j) + c_{23}\, f(i,j+1) + c_{31}\, f(i+1,j-1) + c_{32}\, f(i+1,j) + c_{33}\, f(i+1,j+1)$$

Properties of convolution

$$f * g = g * f$$

$$\begin{aligned} k &= h * f \\ &= (h_1 * h_2) * f \\ &= h_1 * (h_2 * f) \end{aligned}$$

Fourier transform

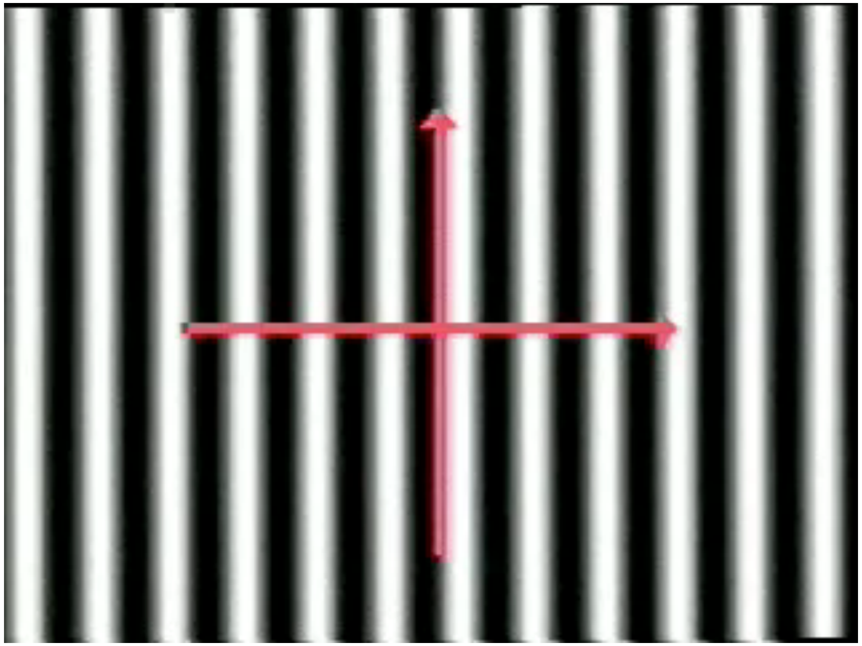
To understand the effect of the convolution in STEP 1 on the image

Computer Vision

Characterization of functions in the frequency domain

orthonormal basis functions $e^{i2\pi(ux+vy)}$

$$= \cos 2\pi(ux+vy) + i \sin 2\pi(ux+vy)$$



$$\lambda = \frac{1}{\sqrt{u^2+v^2}}$$

The Fourier transform

Linear decomposition of functions in the new basis
Scaling factor for basis function (*u,v*)

$$\mathcal{F}[f(x,y)] = F(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)e^{-i2\pi(ux+vy)}dxdy$$

→ The Fourier transform

Reconstruction of the original function in the spatial domain: weighted sum of the basis functions

$$\mathcal{F}^{-1}[F(u,v)] = f(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u,v)e^{i2\pi(ux+vy)}dxdy$$

→ The inverse Fourier transform

$$f(x,y) = \int_{-\infty}^{\infty} f(\alpha,\beta)\delta(x-\alpha,y-\beta)d\alpha d\beta$$

Computer Vision

Fourier coefficients

$F(u,v)$ is complex : $F_R(u,v) + iF_I(u,v)$

The magnitude

$$\left|F(u,v)\right| = \sqrt{F_R(u,v)^2 + F_I(u,v)^2}$$

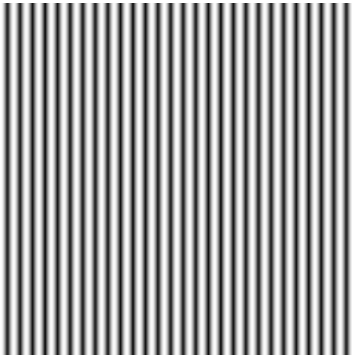
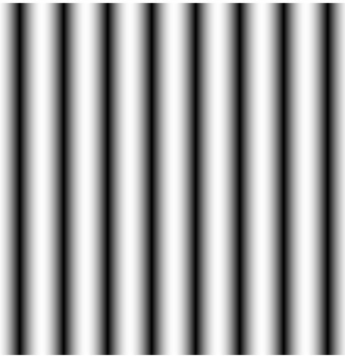
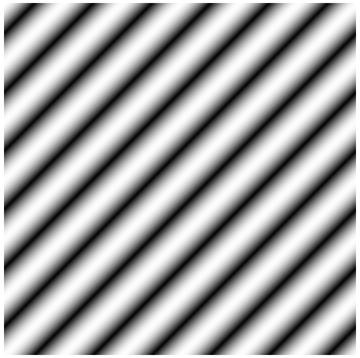
The phase angle

arctan $(F_I(u,v) / F_R(u,v))$

Computer Vision

Fourier decomposition of images

$f(x,y) =$ 

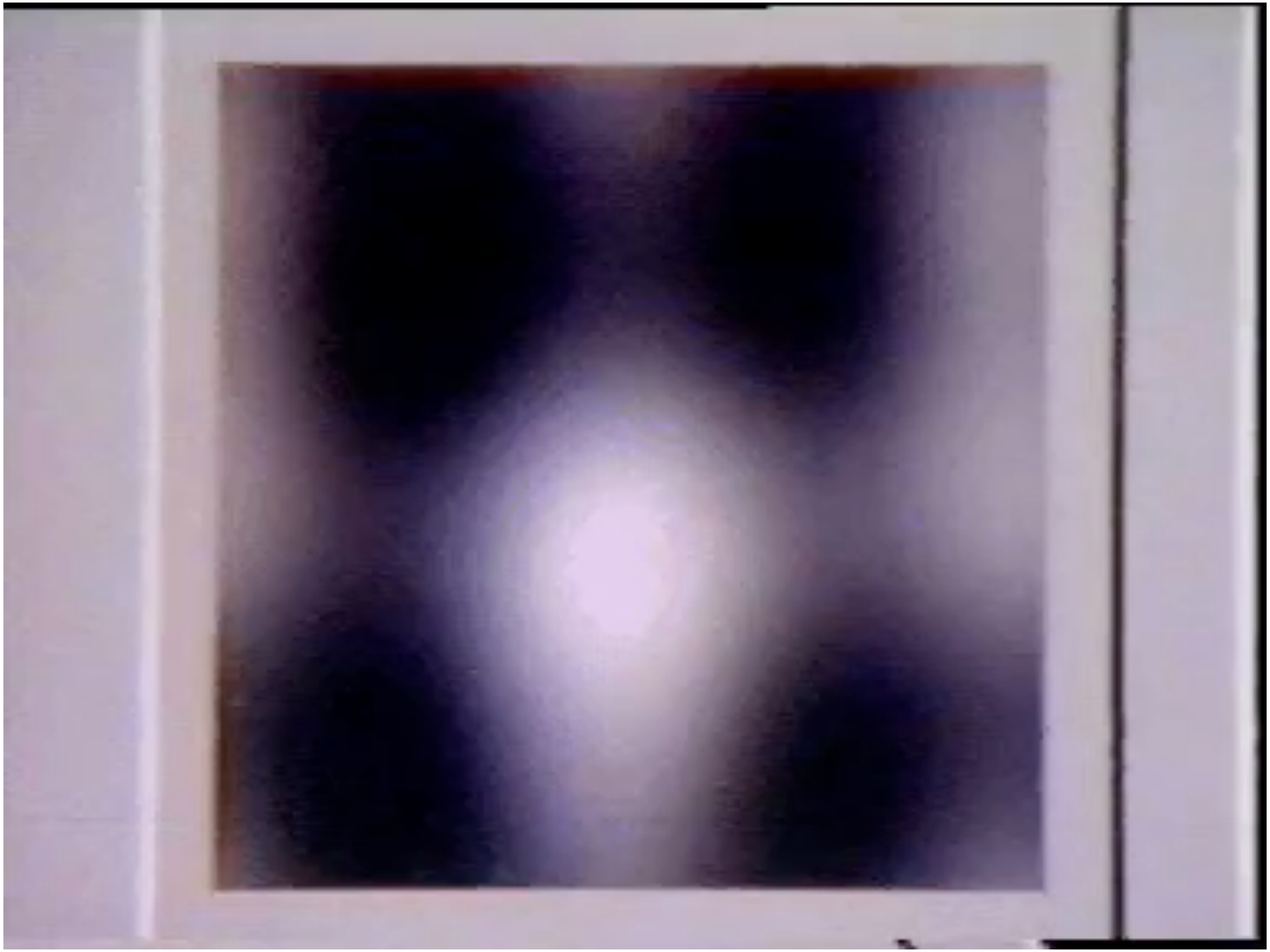
$= F(u,v) \times$  $+ F(u',v') \times$  $+ F(u'',v'') \times$  $+ \ldots$

Computer Vision

Fourier decomposition of images

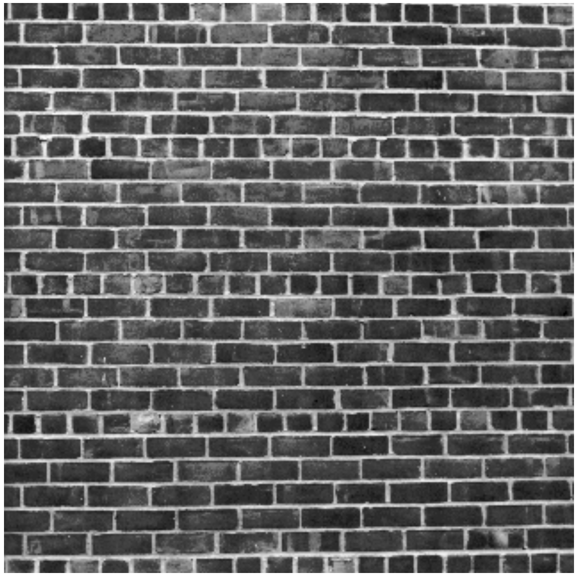
Computer Vision

Fourier decomposition of images

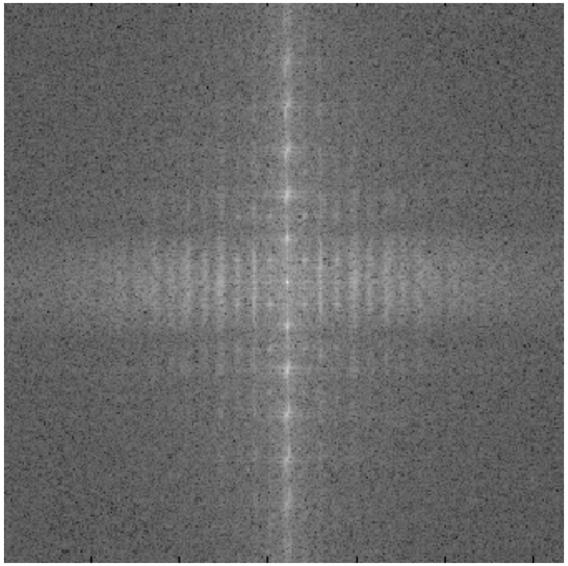


Example importance of magnitude

- Image with periodic structure



$f(x,y)$

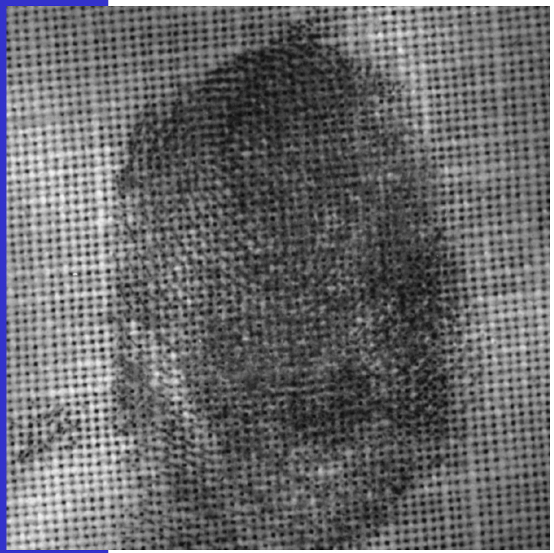


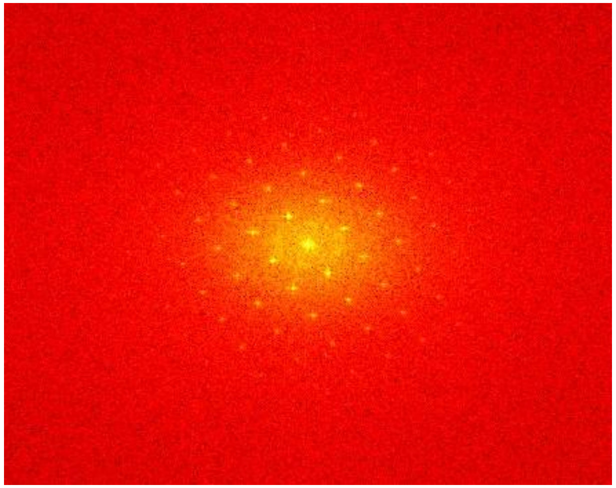
$|F(u,v)|$

FT has peaks at spatial frequencies of repeated texture

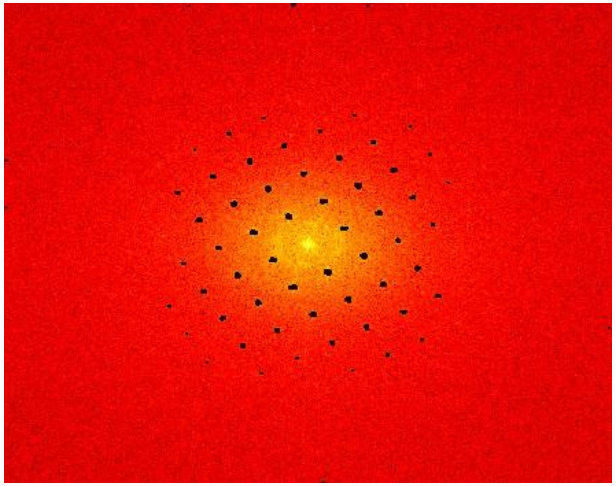
Computer Vision

Example importance of magnitude

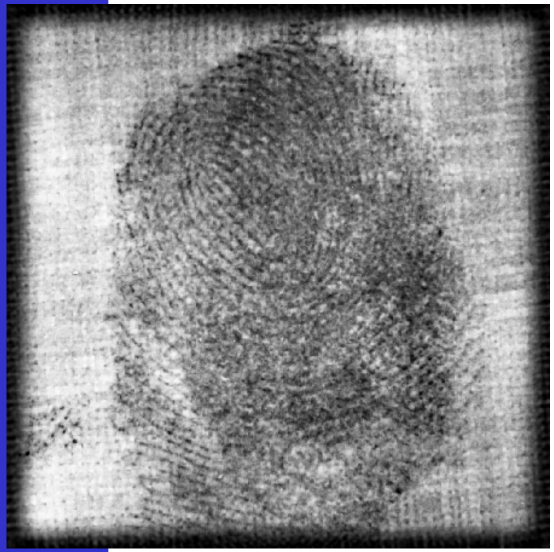




$|F(u,v)|$

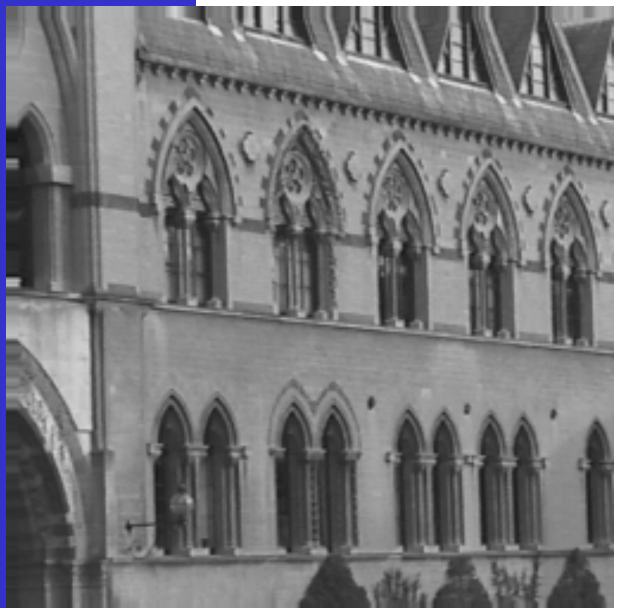


remove peaks



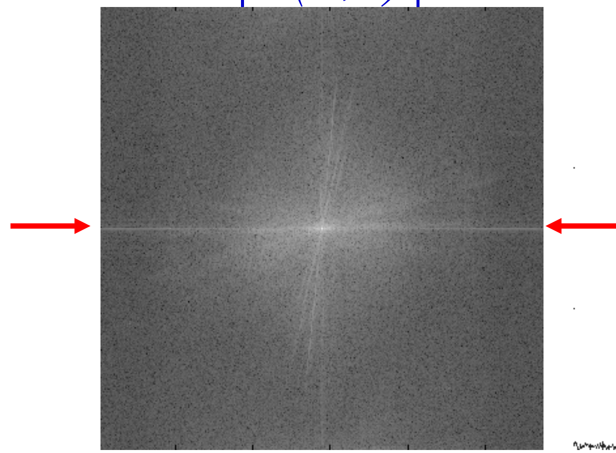
Periodic background removed

Example importance of magnitude

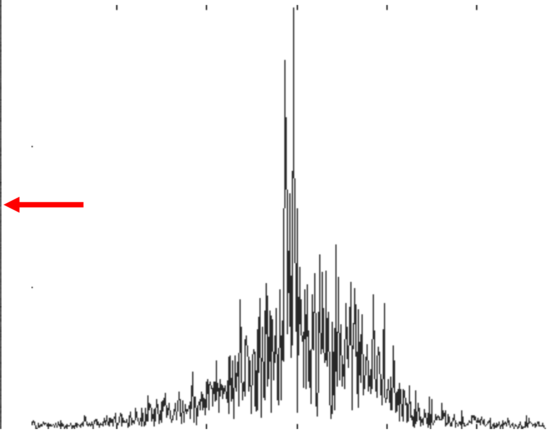


f(x,y)

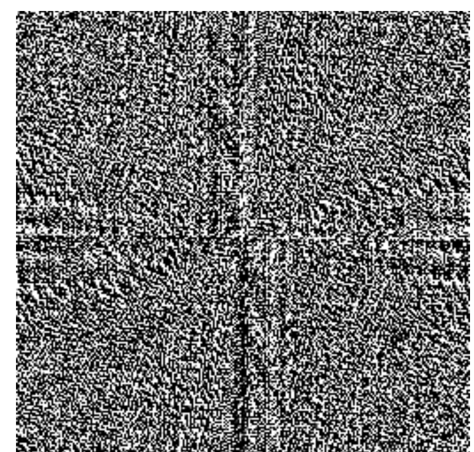
|F(u,v)|



cross-section



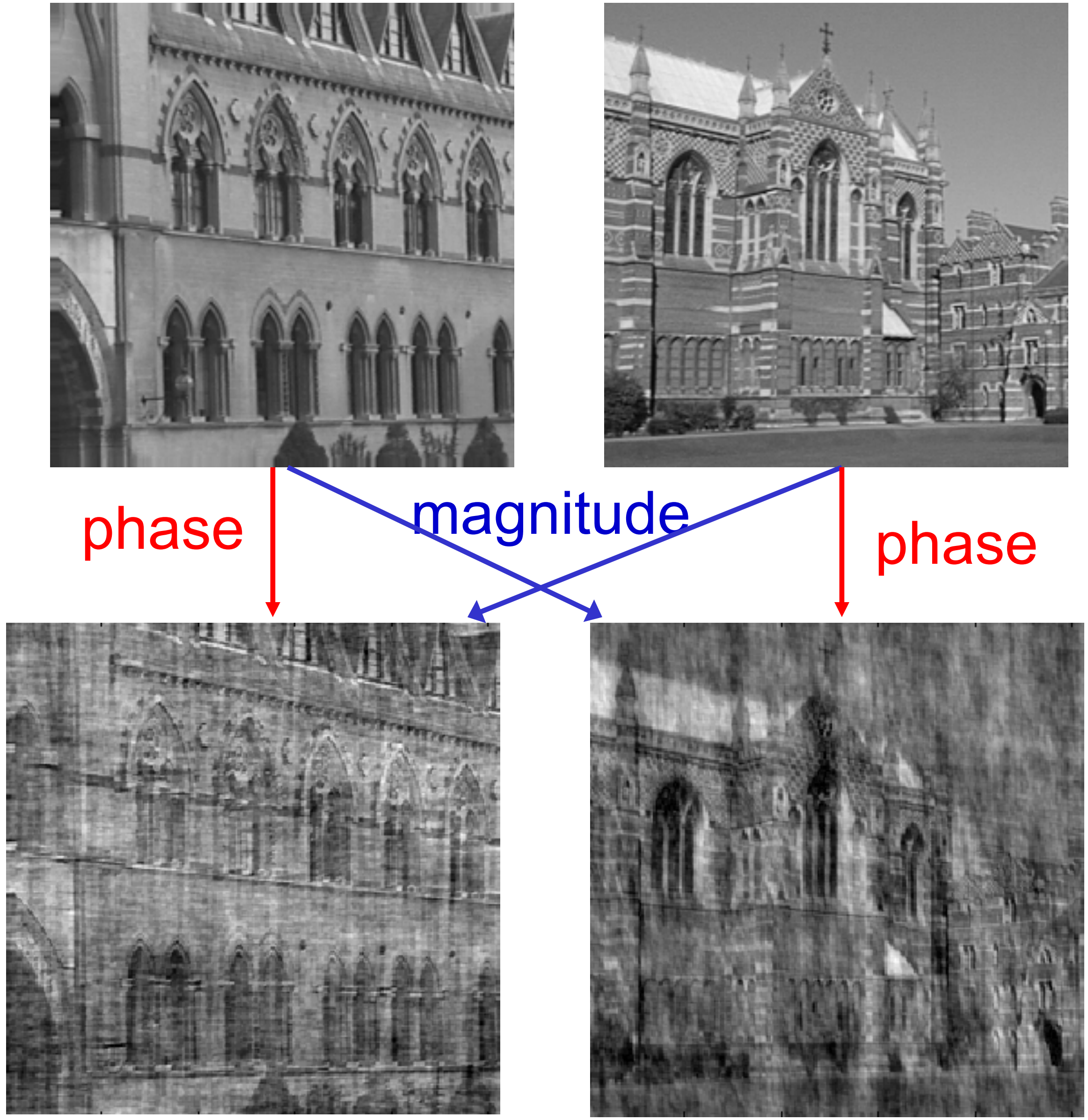
phase F(u,v)



- |F(u,v)| generally decreases with higher spatial frequencies
- phase appears less informative

Computer Vision

The importance of the phase



The convolution theorem

$$c(x,y) = a(x,y) * b(x,y)$$

$$\Downarrow \text{ Fourier}$$

$$C(u,v) =$$

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left[a(x,y) * b(x,y)\right]e^{-i2\pi(ux+vy)}dxdy$$

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} a(x-\alpha, y-\beta)b(\alpha,\beta)d\alpha d\beta\right]e^{-i2\pi(ux+vy)}dxdy$$

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} a(x-\alpha, y-\beta)e^{-i2\pi(ux+vy)}dxdy\right]b(\alpha,\beta)d\alpha d\beta$$

The convolution theorem

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} a(x-\alpha, y-\beta)e^{-i2\pi(ux+vy)}dxdy\right]b(\alpha,\beta)d\alpha d\beta$$

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} a(x^*, y^*)e^{-i2\pi(u(x^*+a)+v(y^*+b))}dx^*dy^*\right] b(\alpha,\beta)d\alpha d\beta$$

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} a(x^*, y^*)e^{-i2\pi(ux^*+vy^*)}dx^*dy^*\right]e^{-i2\pi(ua+vb)} b(\alpha,\beta)d\alpha d\beta$$

That is,

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} A(u,v)e^{-i2\pi(u\alpha+v\beta)}b(\alpha,\beta)d\alpha d\beta$$

$$= A(u,v)B(u,v)$$

Space convolution = frequency multiplication

Point spread function and Modulation transfer function

$$
\begin{aligned}
O(u,v) &= \mathcal{F}\{o(x,y)\} \\
&= \mathcal{F}\{i(x,y) * r(x,y)\} \\
&= I(u,v)R(u,v)
\end{aligned}
$$

$$
\begin{aligned}
R(u,v) &= \mathcal{F}\{r(x,y)\} \\
&= \mathcal{F}\{\text{point spread function}\}
\end{aligned}
$$

$=$ **modulation transfer function**

The convolution theorem: reciprocity

$$
\begin{aligned}
C(u,v) &= A(u,v)B(u,v) \\
c(x,y) &= a(x,y) * b(x,y)
\end{aligned}
$$

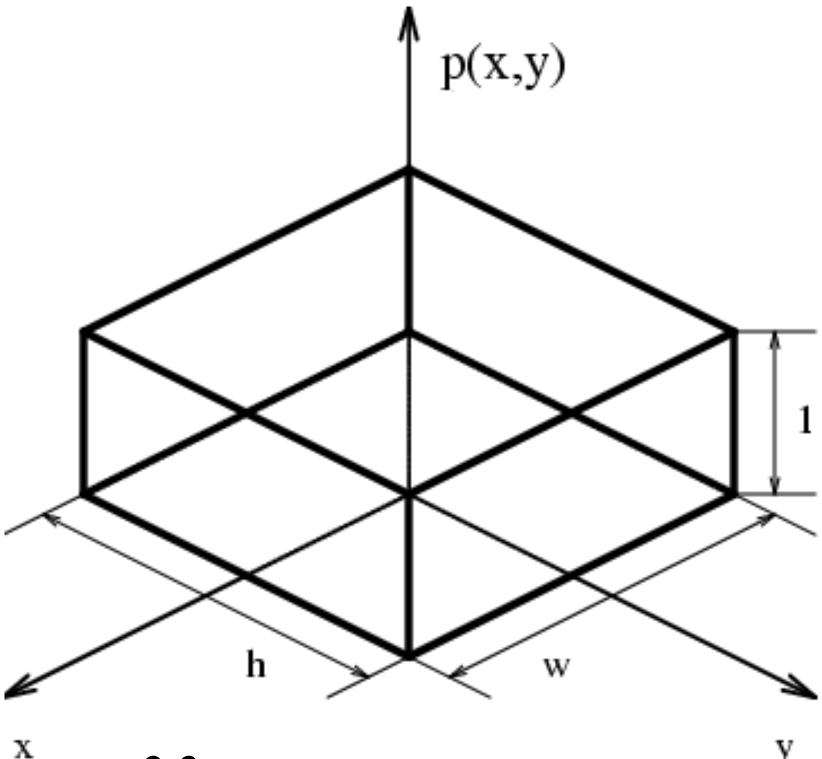
$$
\begin{aligned}
C(u,v) &= A(u,v) * B(u,v) \\
c(x,y) &= a(x,y)b(x,y)
\end{aligned}
$$

Space multiplication = frequency convolution

Back to STEP 1

Computer Vision

STEP 1 : integrating over a pixel cell



$$o(x',y') = \iint i(x,y)\,p(x-x',y-y')\,dx\,dy$$

This is *convolution:* $i(x,y) * p(-x,-y)$

$$O(u,v) = I(u,v)P(u,v)$$

Modulation Transfer Function of the window function

Fourier transform of window :

$$P(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{-i2\pi(ux+vy)} p(x,y)\,dx\,dy$$

$$= \int_{-w/2}^{w/2} e^{-i2\pi ux} dx \int_{-h/2}^{h/2} e^{-i2\pi vy} dy$$

$$= \left[\frac{e^{-i2\pi ux}}{-i2\pi u}\right]_{-w/2}^{w/2} \left[\frac{e^{-i2\pi vy}}{-i2\pi v}\right]_{-h/2}^{h/2}$$

$$= -\frac{1}{4\pi^2 uv}(-2i\sin(2\pi u\frac{w}{2}))(-2i\sin(2\pi v\frac{h}{2}))$$

$$= wh\,(\frac{\sin \pi wu}{\pi wu})(\frac{\sin \pi hv}{\pi hv})$$

Fourier transform of the window function

2D sinc :

real ⇒ no phase shifts

however, polarity reversals !

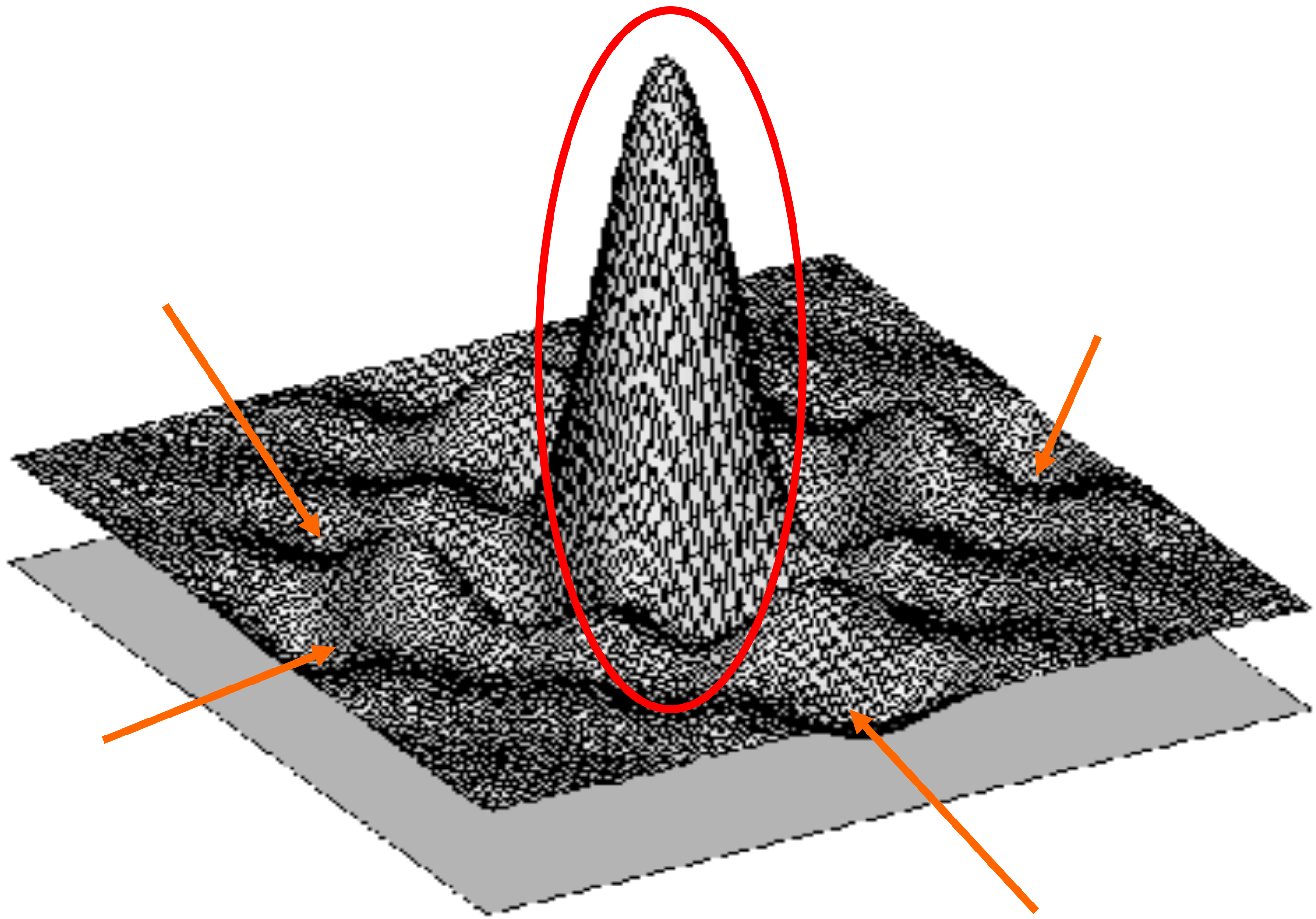
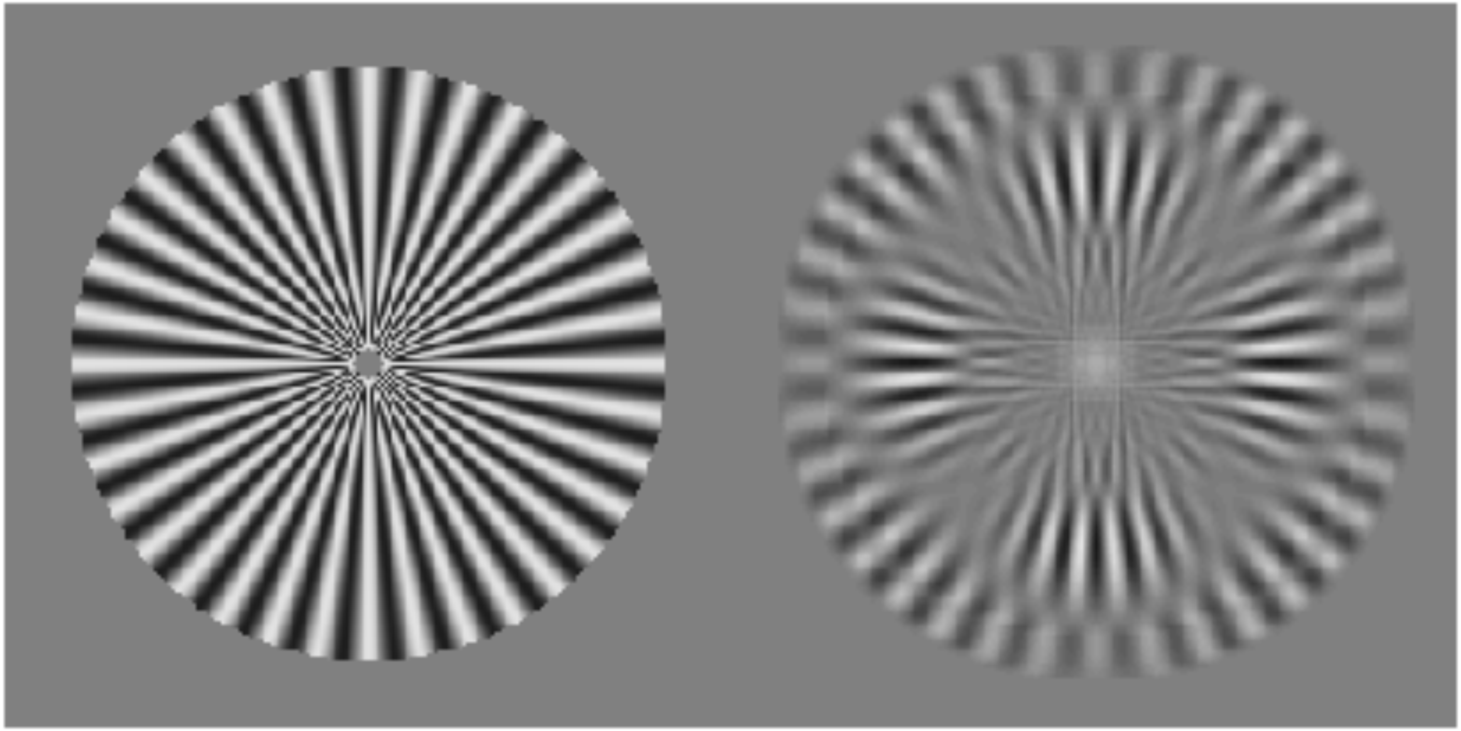


Illustration of the sinc



$$P(u,v) = wh\left(\frac{\sin \pi w u}{\pi w u}\right)\left(\frac{\sin \pi h v}{\pi h v}\right)$$

A model for sampling

1. Integrate brightness over cell window

   Image degradations

2. Read out values only at the pixel centers

   Aliasing
   Leakage

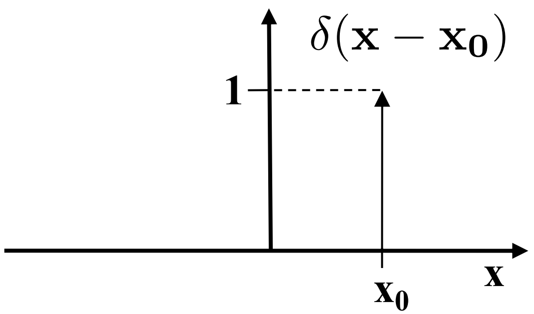
Computer Vision

STEP 2: local probing of functions

Distributions as extension of functions: the Dirac pulse

$$\delta(\mathbf{x}-\mathbf{x}_0) = 0 \quad \mathbf{x} \neq \mathbf{x}_0$$

$$\int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \delta(\mathbf{x}-\mathbf{x}_0) d\mathbf{x} = 1$$



Function probing (in 1D)

$$\int_{-\infty}^{\infty} \delta(x) f(x) dx = f(0)$$

$$\int_{-\infty}^{\infty} \delta(x-x_0) f(x) dx = f(x_0)$$

Discretization in the spatial domain is multiplication with a Dirac train

multiplication with 2D pulse train

$$\sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} \delta(x-kw, y-lh)$$

Fourier transform :

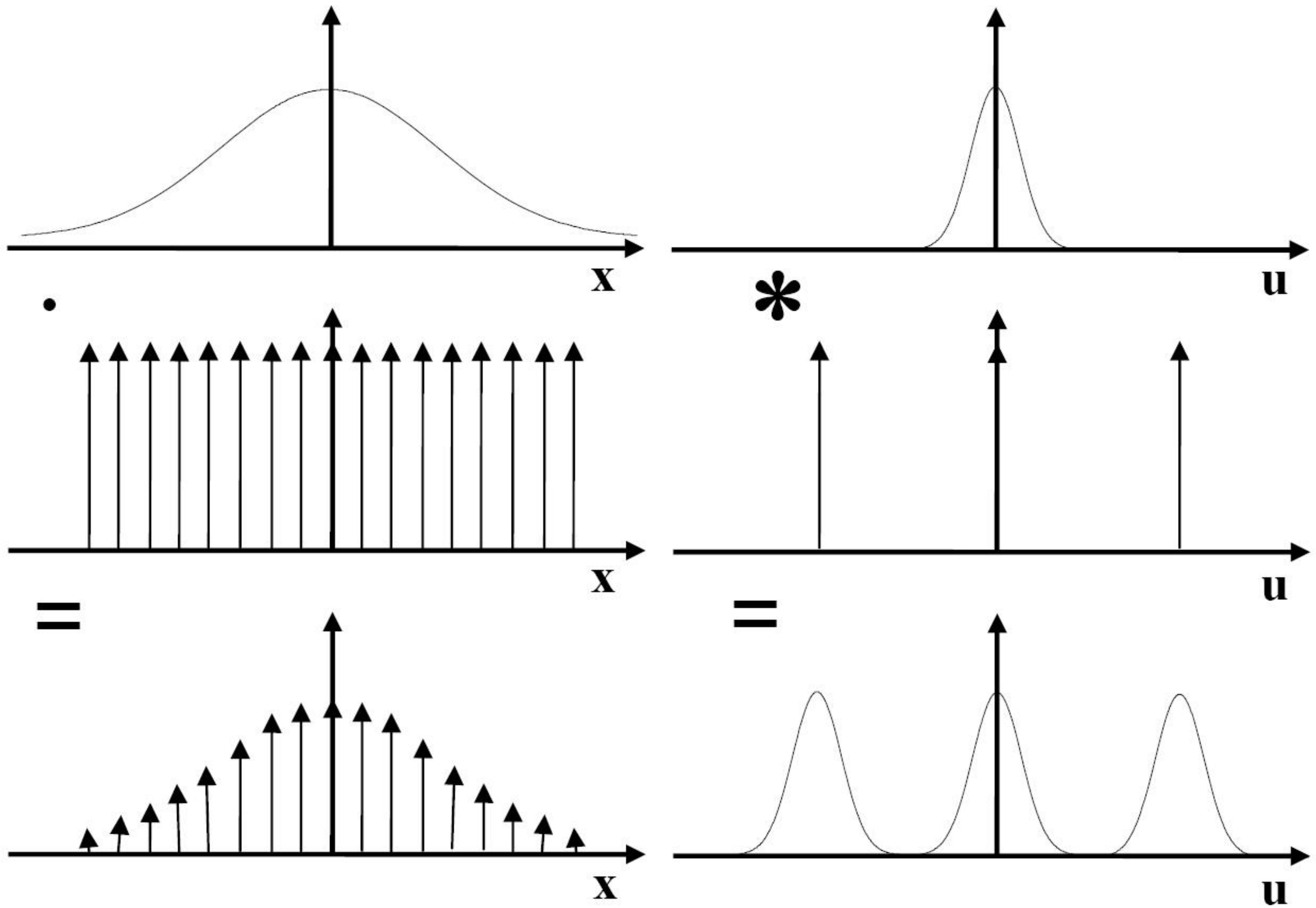
$$\frac{1}{wh} \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} \delta(x-k\frac{1}{w}, y-l\frac{1}{h})$$

Convolution with a Dirac train: periodic repetition

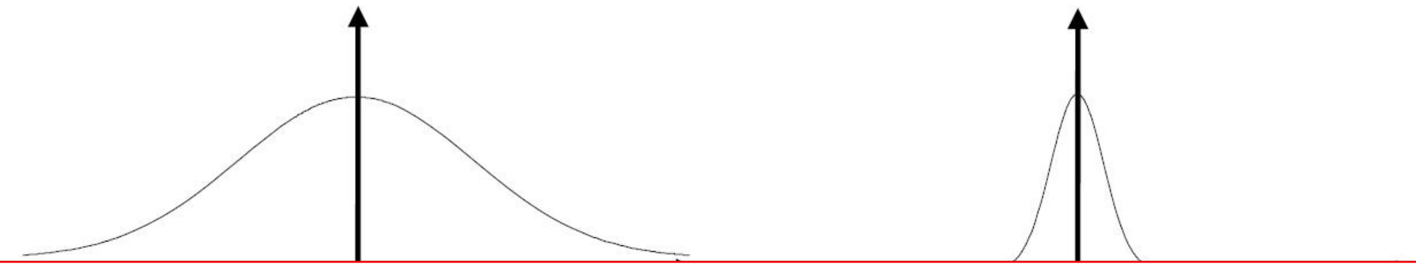
Yet another duality: discrete vs. periodic

Computer Vision

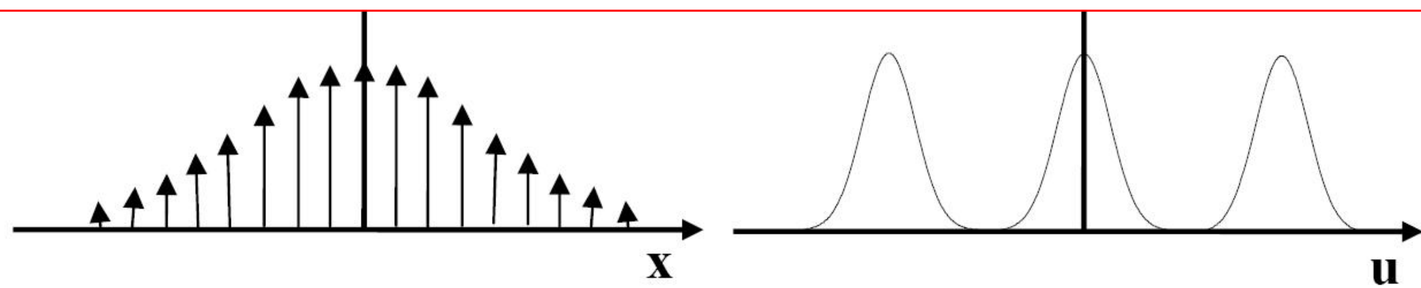
Effect on the frequency domain



Effect on the frequency domain



1. After sampling you may not get back the original signal
2. It depends on the frequency domain representation, only band limited signals can be sampled and retrieved back
3. Even then you need to sample at a certain rate



The sampling theorem

If the Fourier transform of a function $f(x,y)$ is zero for all frequencies beyond u_b and v_b, i.e. if the Fourier transform is *band-limited*, then the continuous periodic function $f(x,y)$ can be completely reconstructed from its samples as long as the sampling distances w and h along the x and y directions are such that $w \leq \frac{1}{2u_b}$ and $h \leq \frac{1}{2v_b}$

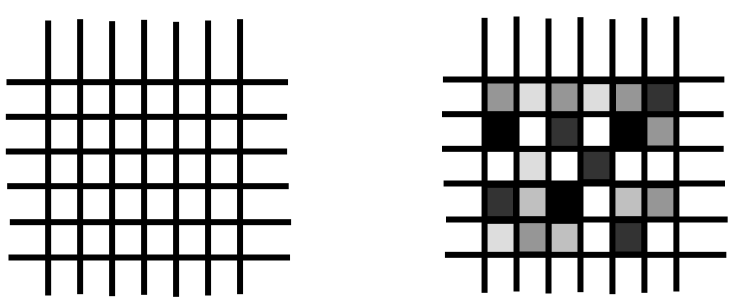
Computer Vision

Discretization

Computer to process an image :

1. sampling ‣ “pixels”

2. quantisation ‣ “grey levels”



Quantisation

Create K intervals in the range of possible intensities
measured in bits: *log2*(K)

Design choices

- Decision levels

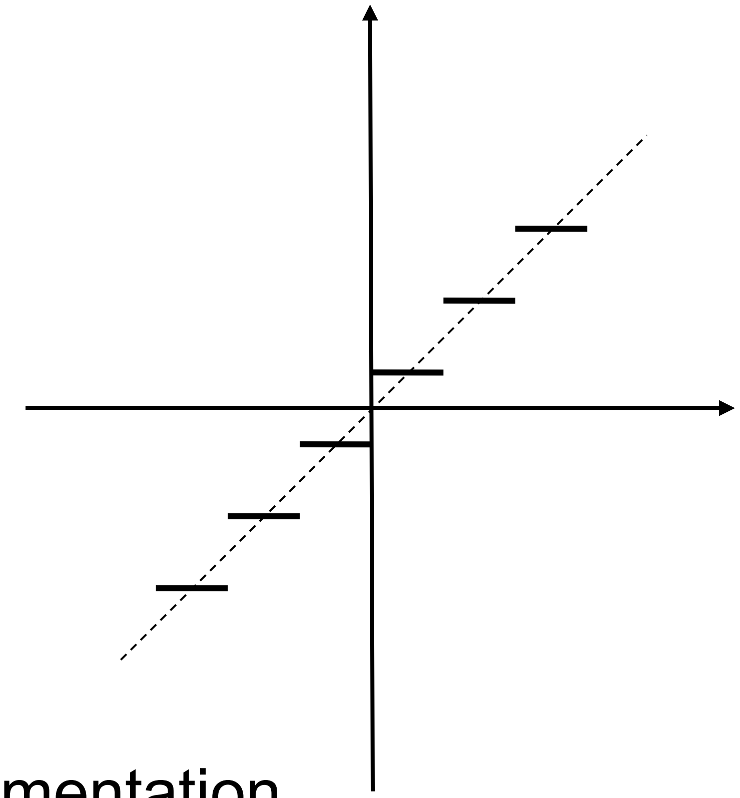
$$z_1, z_2, \ldots, z_{K+1}$$

- Representative value

interval $[z_k, z_{k+1}] \rightarrow q_k$

- Simplest selection
  - equal intervals
  - value is the mean
  - Δ uniform quantizer

The uniform quantizer



- simple implementation
- fine quantization needed perceptually (7-8 bits)
- can be reduced by optimal design, e.g.

minimize $\delta = \sum_{k=1}^{K} \int_{z_k}^{z_{k+1}} (z - q_k)^2 p(z) dz := \sum_{k=1}^{K} \delta_k$

(p(z)=prob. density function, for constant Δ uniform)

Underquantization example

256 gray level (8 bit)

11 gray level

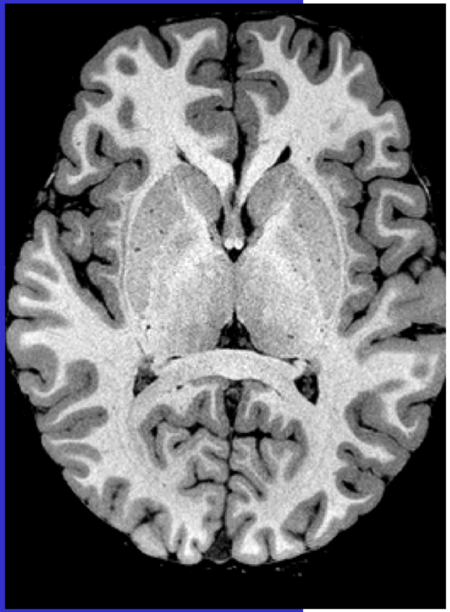
Remarks

- Quantization:
  - Often 8 bits per pixel (monochrome), 24 bits per pixel (RGB)
  - Medical images 12 bits (4096 levels) or 16 bits (65536 levels)

Part II
Image Enhancement

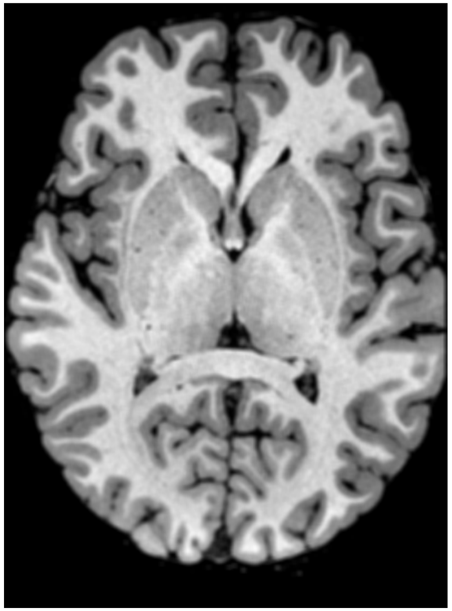
Three types of image enhancement

1. Noise suppression
2. Image de-blurring
3. Contrast enhancement

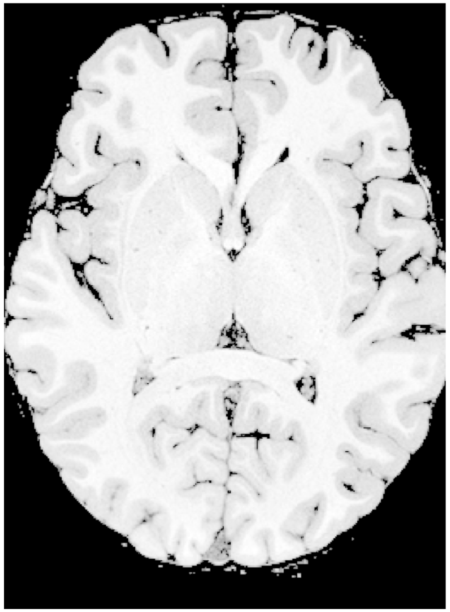


Original Image

Noise



Blur



Bad Contrast

Fourier transform

Signal and noise

Computer Vision

Reminders from previous lecture: Fourier Transform

Linear decomposition of functions in the new basis
Scaling factor for basis function (*u,v*)

$$\mathcal{F}[f(x,y)] = F(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)e^{-i2\pi(ux+vy)}dxdy$$

→ The Fourier transform

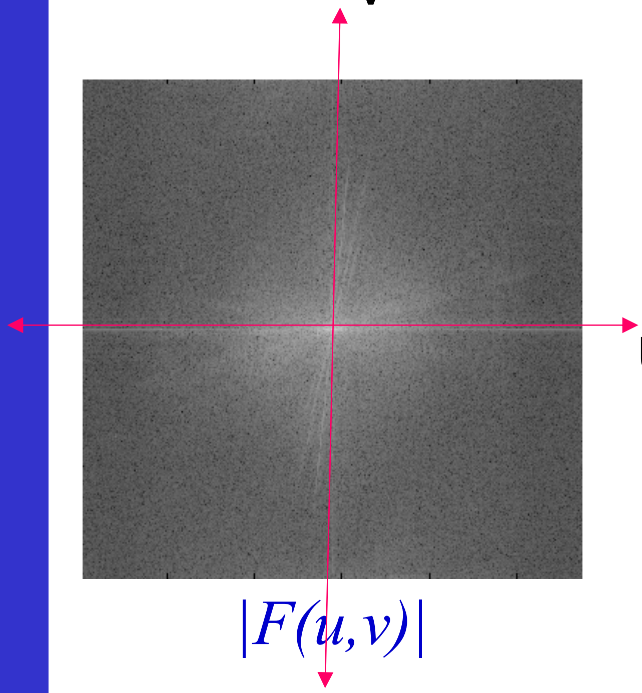
Reconstruction of the original function in the spatial domain: weighted sum of the basis functions

$$\mathcal{F}^{-1}[F(u,v)] = f(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u,v)e^{i2\pi(ux+vy)}dxdy$$

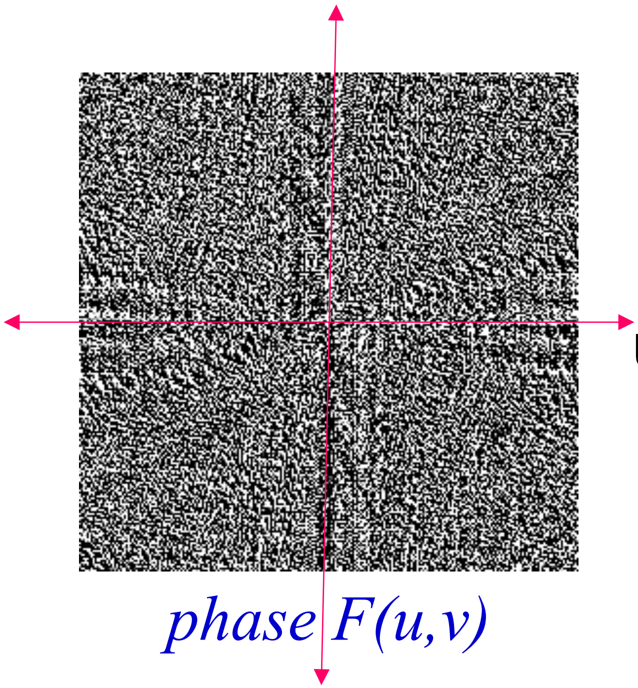
→ The inverse Fourier transform

Computer Vision

$f(x,y)$



$|F(u,v)|$



phase $F(u,v)$

Reminders from previous lecture: Convolution Theorem

$$C(u,v) = A(u,v)B(u,v)$$
$$c(x,y) = a(x,y) * b(x,y)$$

Space convolution = frequency multiplication

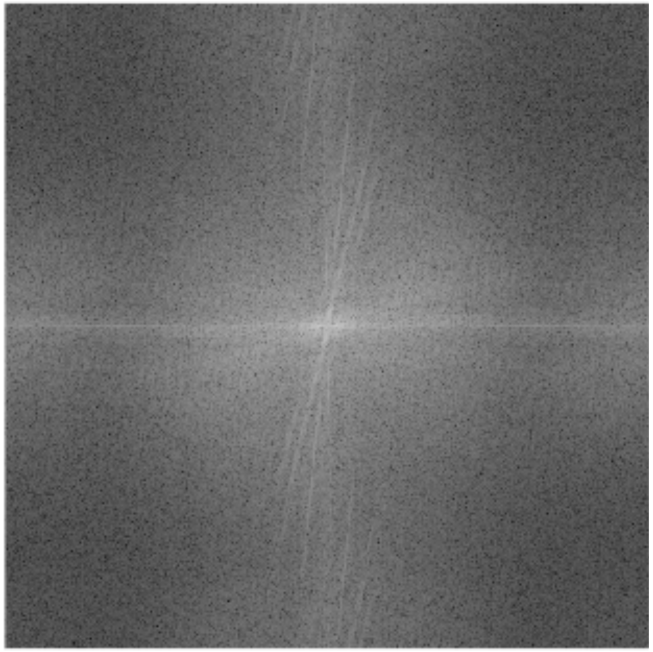
$$C(u,v) = A(u,v) * B(u,v)$$
$$c(x,y) = a(x,y)b(x,y)$$

Space multiplication = frequency convolution

Computer Vision

Fourier power spectra of images

$i(x,y)$



$\phi_{ii} = |I(u,v)|^2$

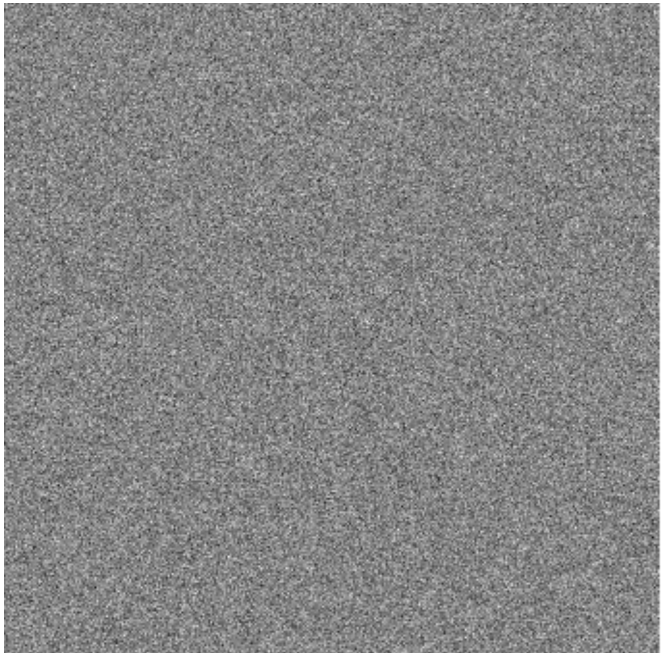
Amount of signal at each frequency pair

Images are mostly composed of homogeneous areas

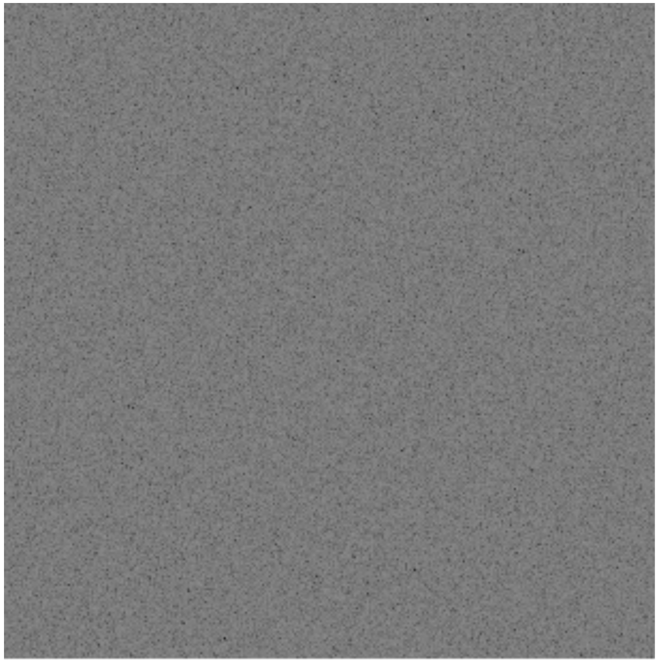
- Most nearby object pixels have similar intensity
- Most of the signal lies in low frequencies!
- High frequency contains the edge information!

Computer Vision

Fourier power spectra of noise



$n(x,y)$

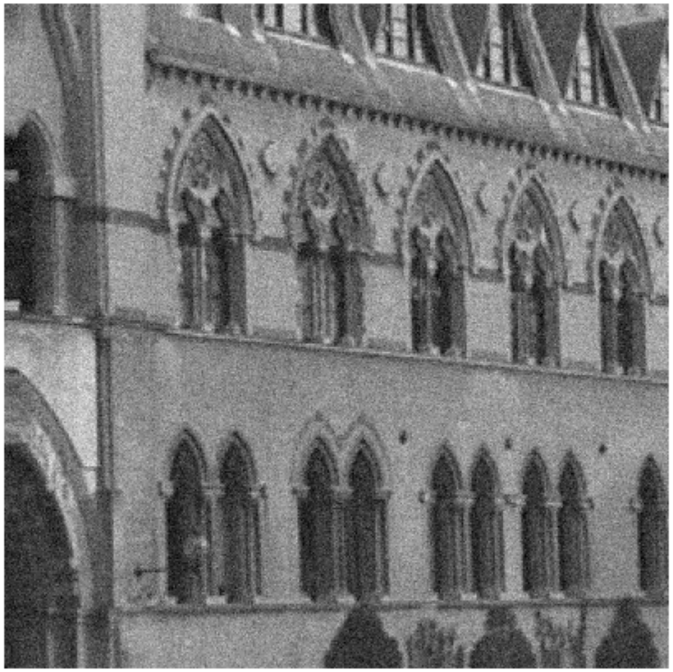


$\phi_{nn} = |N(u,v)|^2$

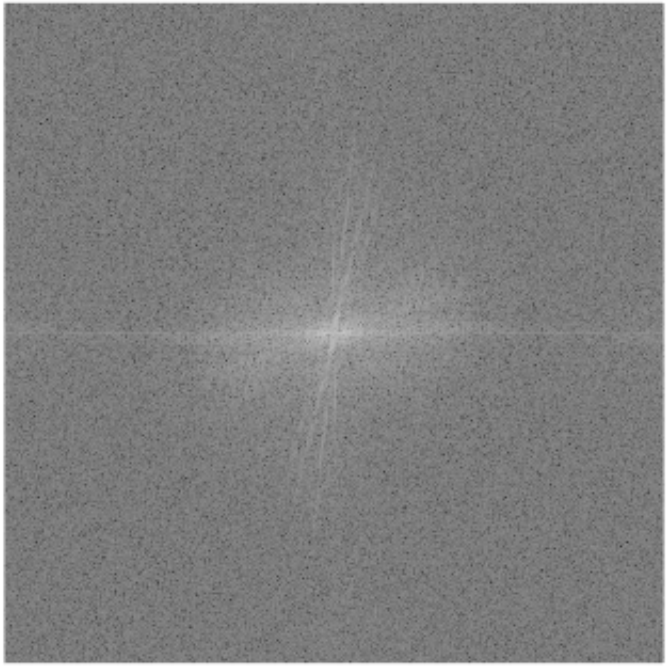
-Pure noise has a uniform power spectra
-Similar components in high and low frequencies.

Computer Vision

Fourier power spectra of noisy image



$f(x,y)$

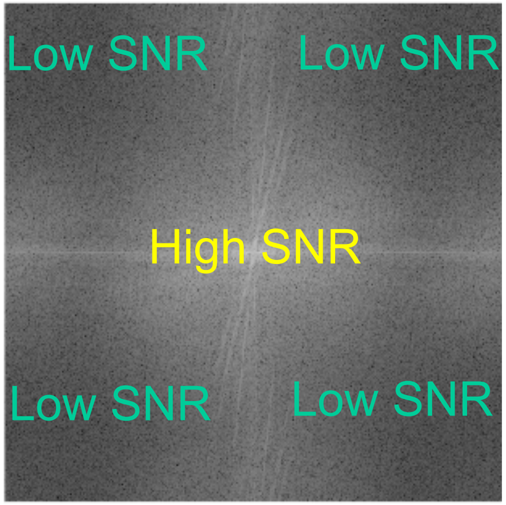


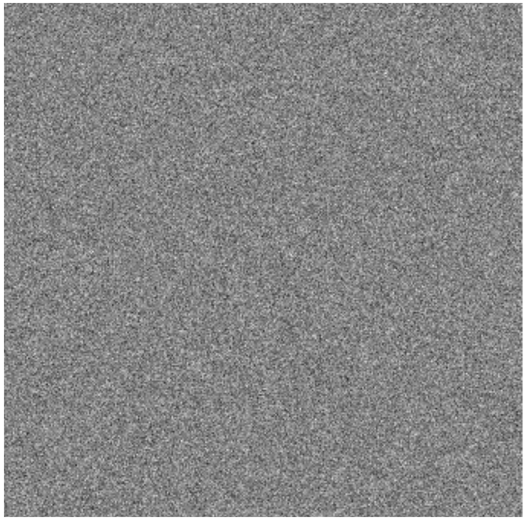
$\phi_{ff} = |F(u,v)|^2$

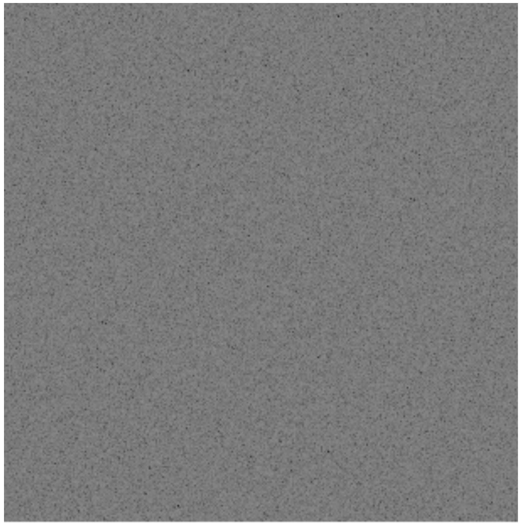
Power spectra is a combination of image and noise

Computer Vision

Signal to Noise Ratio





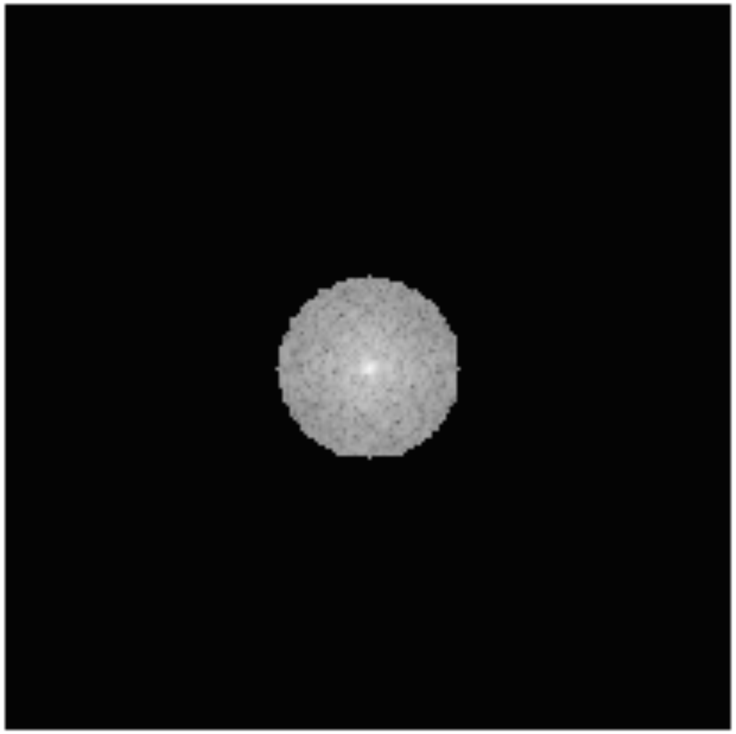


$$\phi_{ii}(u,v) \,/\, \phi_{nn}(u,v)$$

Computer Vision

Only retaining the low frequencies

Low signal/noise ratio at high frequencies ⇒ eliminate these



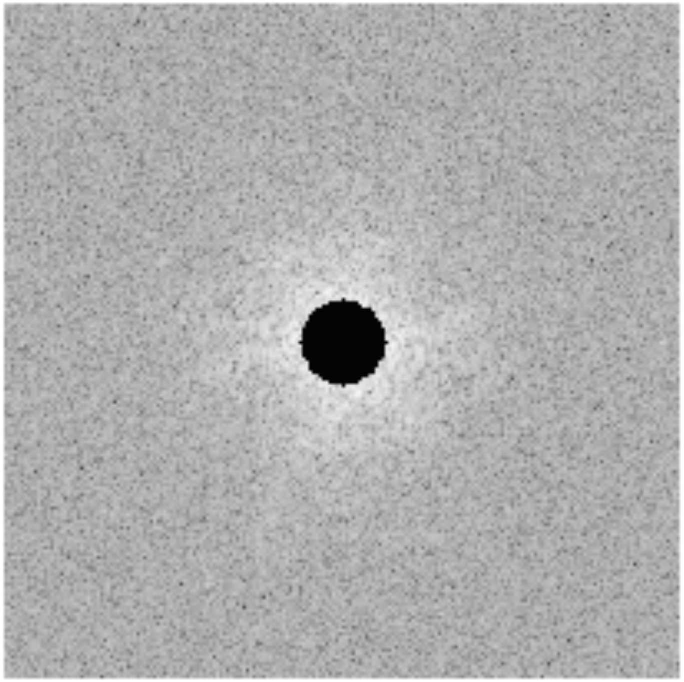
Smoother image but we lost details!

Computer Vision

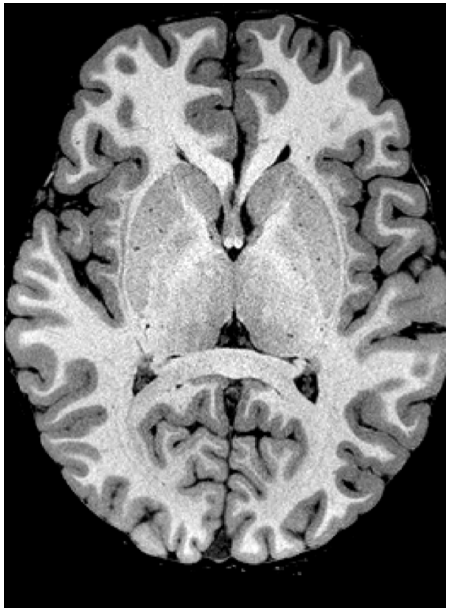
High frequencies contain noise but also Edges!

We cannot simply discard the higher frequencies

They are also introduces by edges ; example :



Computer Vision



Original Image

Noise Suppression

Noisy Observation

Noise suppression

specific methods for specific types of noise

we only consider 2 general options :

1. Convolutional linear filters
   - low-pass convolution filters

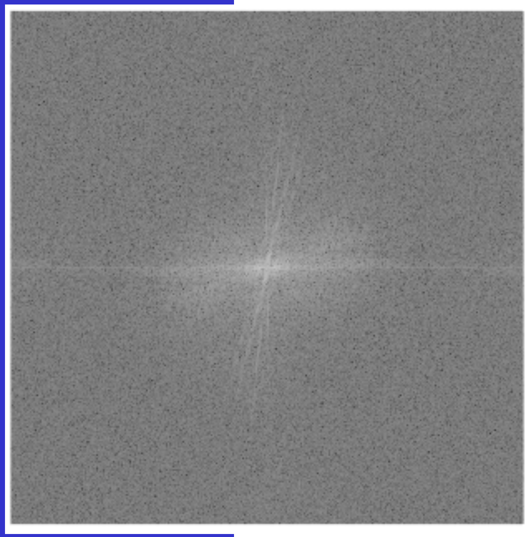
2. Non-linear filters
   - edge-preserving filters
   a. Median
   b. Anisotropic diffusion

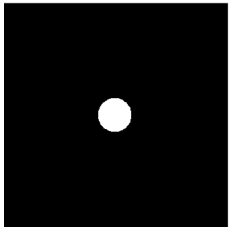
Low-pass filters: principle

Goal: remove low-signal/noise part of the spectrum

Approach 1: Multiply the Fourier domain by a mask

Such spectrum filters yield “rippling”
due to ripples of the spatial filter and convolution





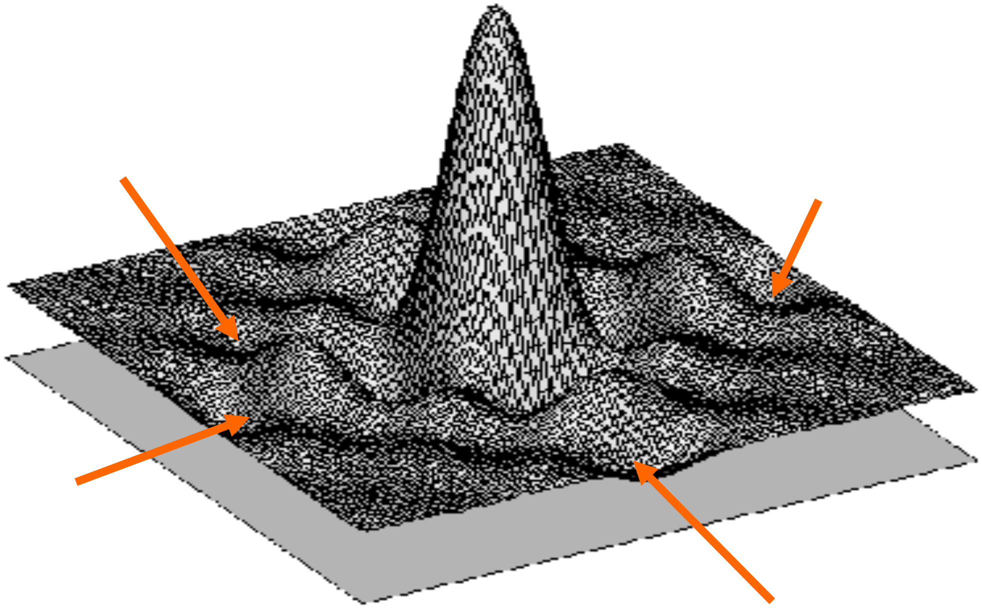
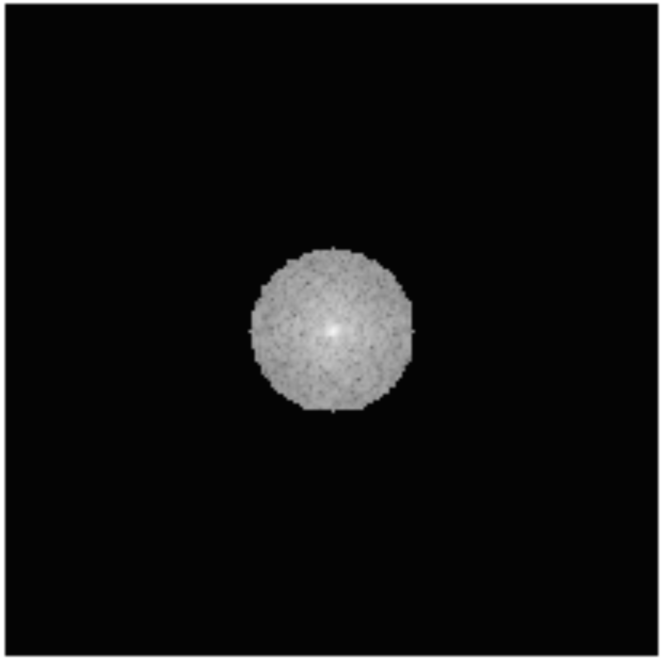


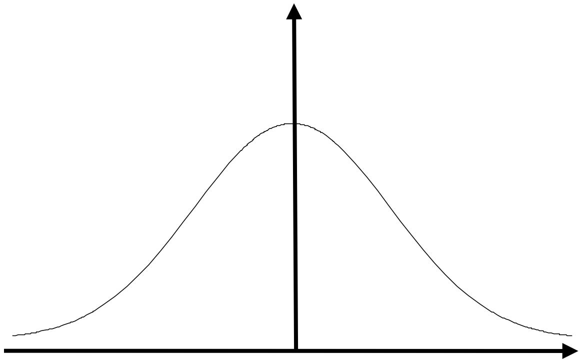
Illustration of rippling



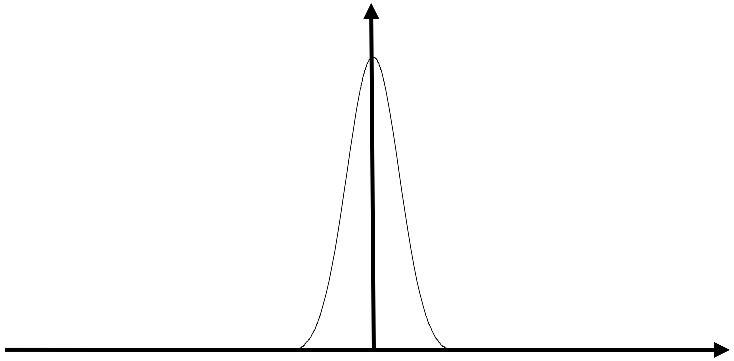
Approach 2: Low-pass convolution filters

generate low-pass filters that do not cause rippling

Idea: Model convolutional filters in the spatial domain to approximate low-pass filtering in the frequency domain



Convolutional filter



Frequency mask

Averaging

One of the most straight forward convolution filters: averaging filters

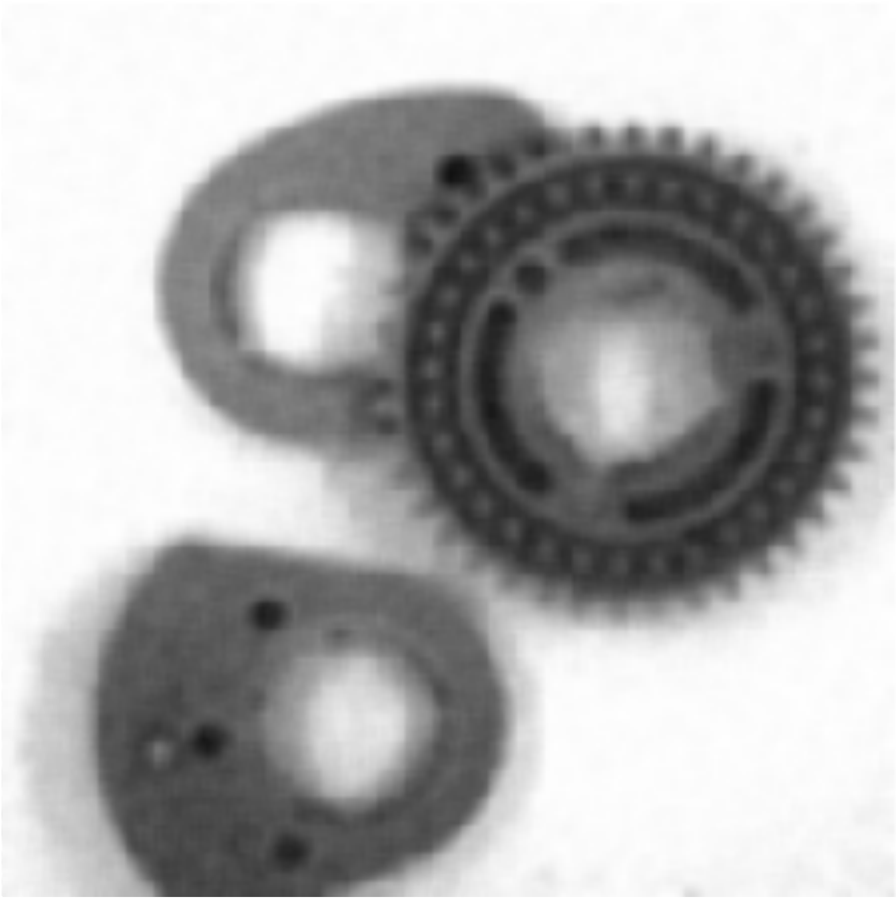
$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix} \qquad \frac{1}{25}\begin{bmatrix}1 & 1 & 1 & 1 & 1\\ 1 & 1 & 1 & 1 & 1\\ 1 & 1 & 1 & 1 & 1\\ 1 & 1 & 1 & 1 & 1\\ 1 & 1 & 1 & 1 & 1\end{bmatrix}$$

Separable:

$$1/9\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix} = 1/3\begin{bmatrix}1\\ 1\\ 1\end{bmatrix} * 1/3\begin{bmatrix}1 & 1 & 1\end{bmatrix}$$

$$o(x,y) = f(x,y) * i(x,y) = f_1(x,y) * (f_2(x,y) * i(x,y))$$

Example for box averaging

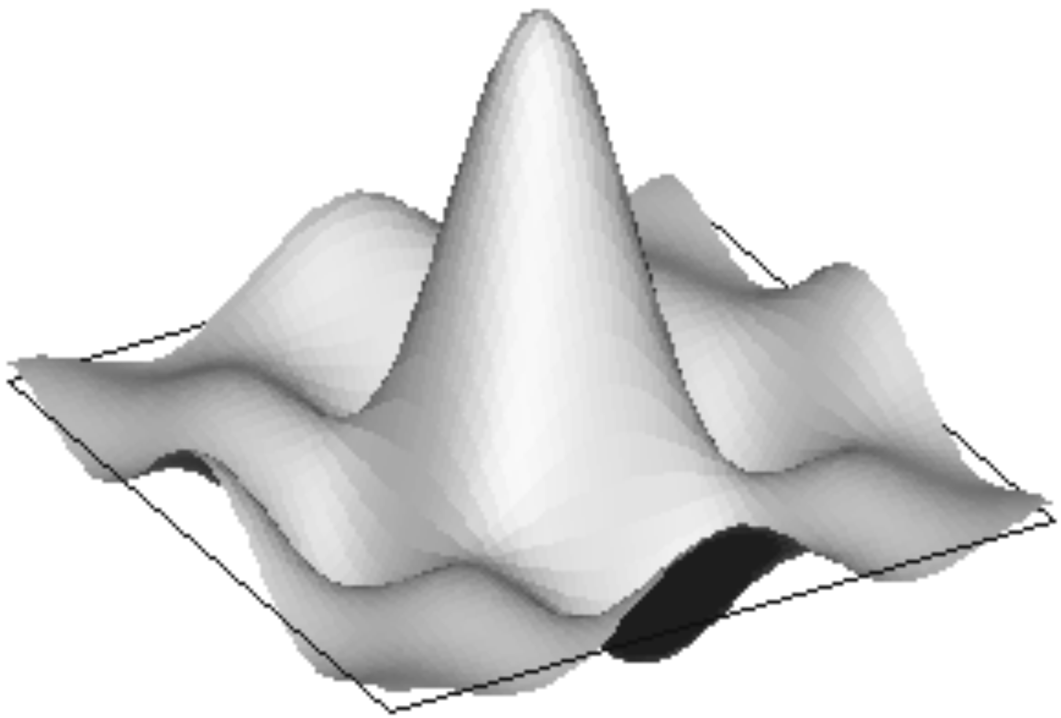


Noise is gone.
Result is blurred!

MTFs for averaging

5 x 5 (separable)

$$(1+2\cos(2\pi u)+2\cos(4\pi u))(1+2\cos(2\pi v)+2\cos(4\pi v))$$



not even low-pass!

So far

1. Masking frequency domain with window type low-pass filter yields sinc-type of spatial filter and ripples -> disturbing effect

2. box filters are not exactly low-pass, ripples in the frequency domain at higher freq. remember phase reversals?

no ripples in either domain required!

Solution: Binomial filters

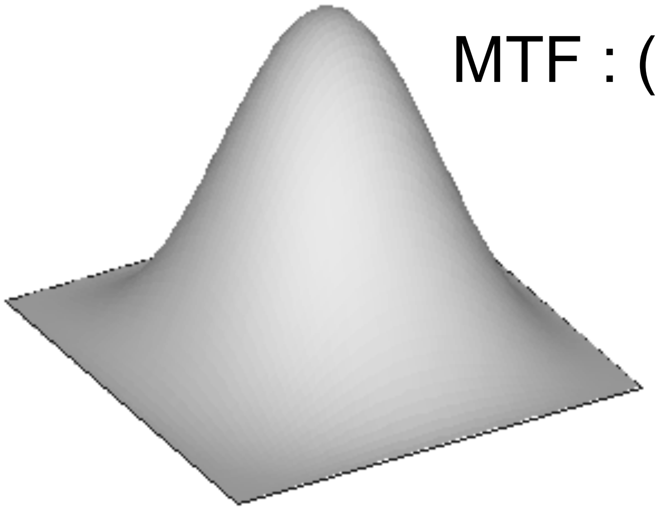
iterative convolutions of (1,1)

only odd filters : (1,2,1), (1,4,6,4,1)

2D :

1	2	1
2	4	2
1	2	1

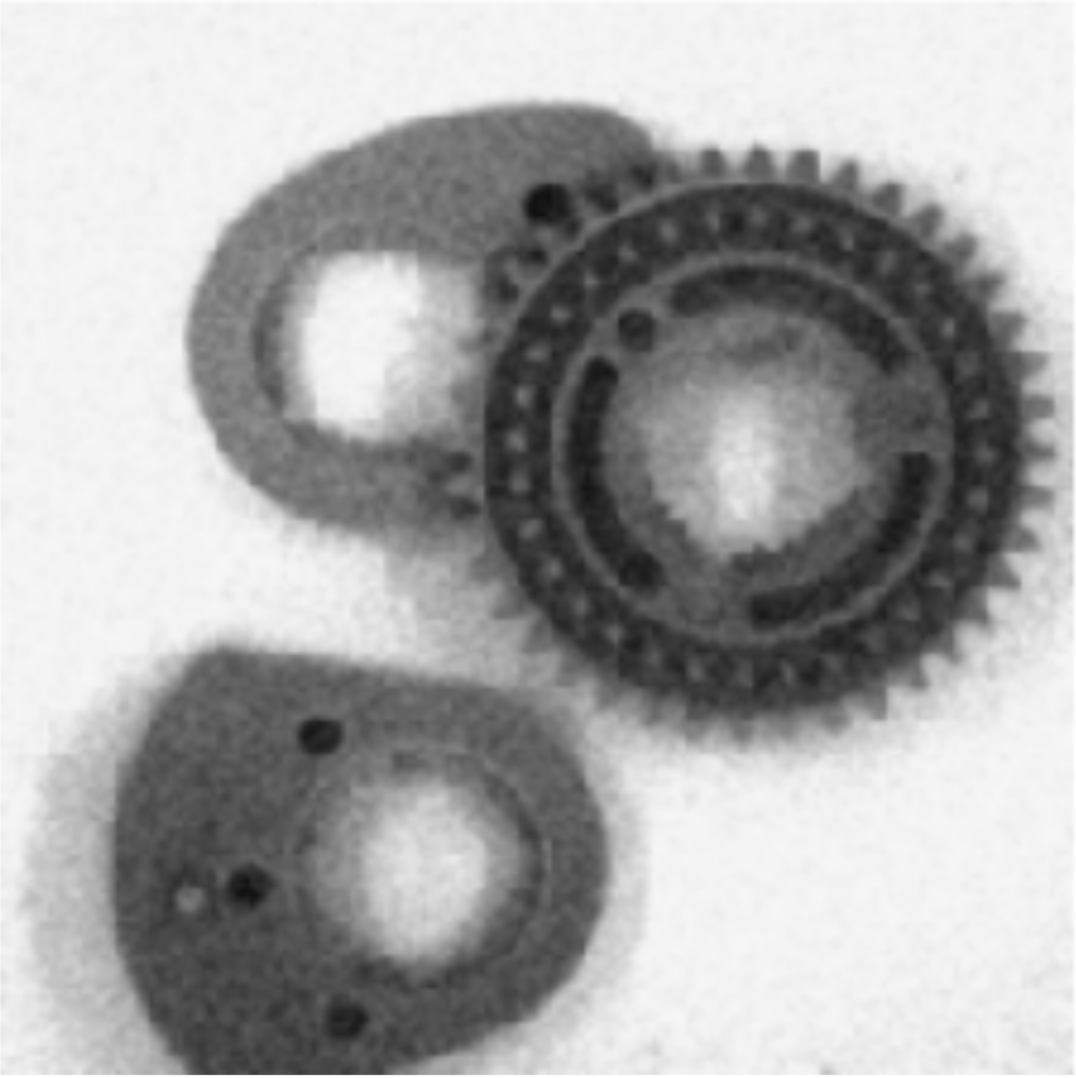
Also separable



MTF : $(2+2\cos(2\pi u))(2+2\cos(2\pi v))$

Computer Vision

Result of binomial filter



➔

Limit of iterative binomial filtering

f :

1	2	1
2	4	2
1	2	1

$$f(x,y) * f(x,y) * \cdots * f(x,y) = f^n(x,y)$$

$$f^n(x,y) \rightarrow a \exp\left(\frac{\|(x,y)\|^2}{b}\right), \ \text{as } n \rightarrow \infty$$

Gaussian

Gaussian smoothing

Gaussian is limit case of binomial filters

noise gone, no ripples, but still blurred…

Actually linear filters cannot solve this problem

Some implementation issues

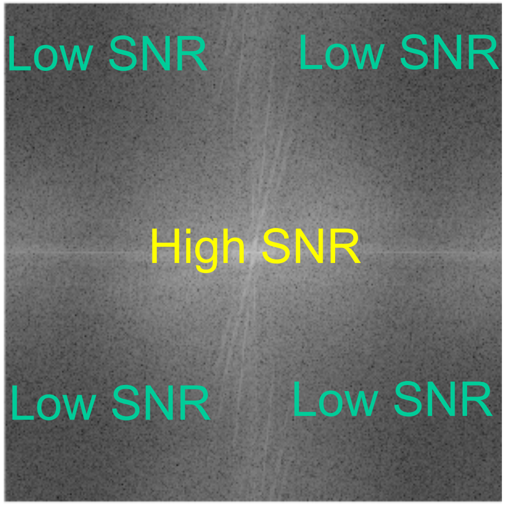
separable filters can be implemented efficiently

large filters through multiplication in the frequency domain

integer mask coefficients increase efficiency
powers of 2 can be generated using shift operations

Computer Vision

Question



Can a linear-shift-invariant systems do a perfect job?

Can they separate edge information from noise in the higher frequency components?

Why?

Noise suppression

specific methods for specific types of noise

we only consider 2 general options :

1. Convolutional linear filters
   - low-pass convolution filters

2. Non-linear filters
   - edge-preserving filters
   a. Median
   b. Anisotropic diffusion

Median filters : principle

non-linear filter

method :

- 1. rank-order neighbourhood intensities
- 2. take middle value

no new grey levels emerge...

Median filters : odd-man-out

advantage of this type of filter is its "odd-man-out" effect

e.g.

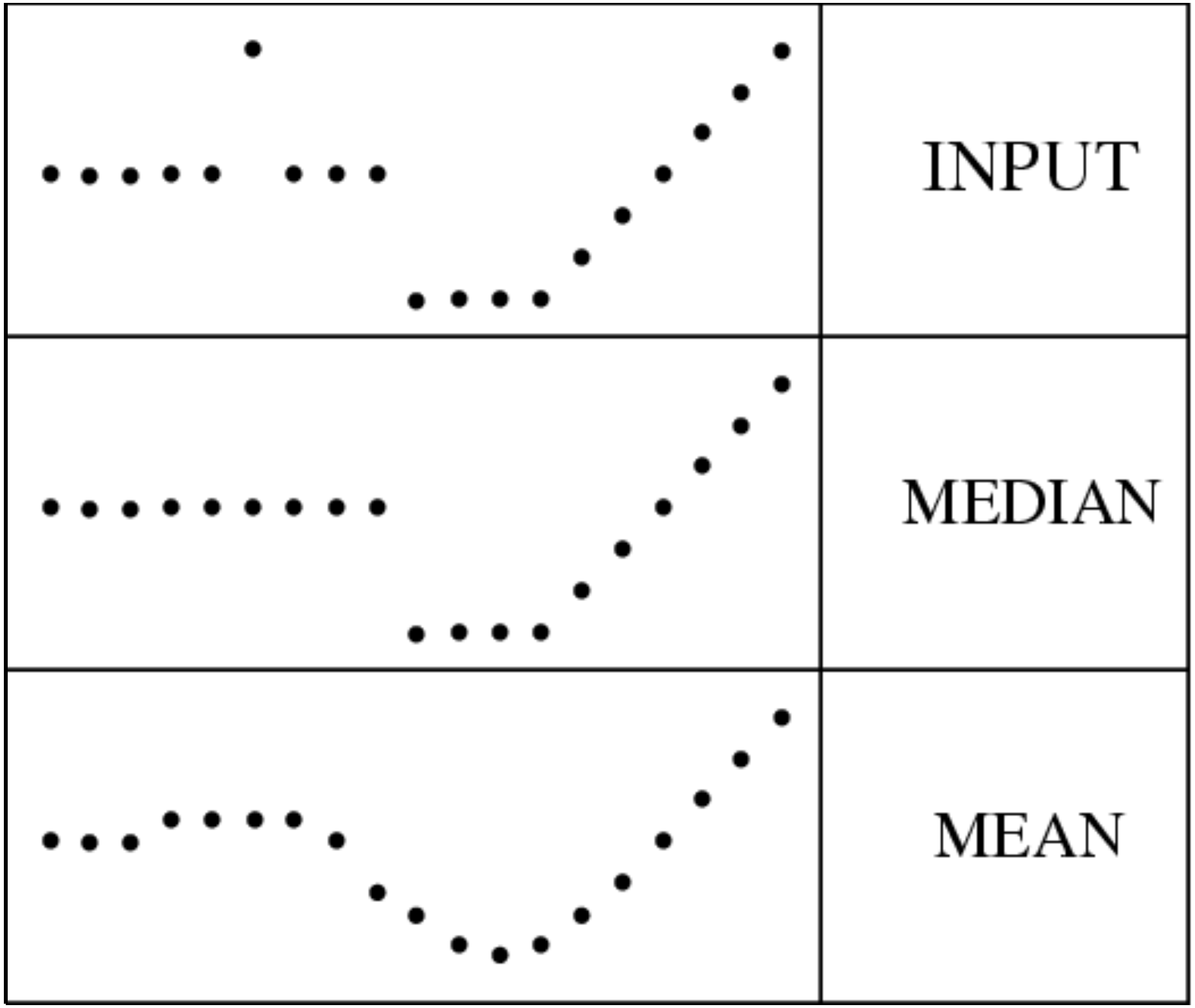
1,1,1,7,1,1,1,1

↓

?,1,1,1.1,1,1,?

Median filters : example

filters have width 5 :



Median filters : analysis

median completely discards the spike,
linear filter always responds to all aspects

median filter preserves discontinuities,
linear filter produces rounding-off effects

DON'T become all too optimistic

Median filter : results

3 x 3 median filter :

sharpens edges, destroys edge cusps and protrusions

Median filters : results

Comparison with Gaussian :

e.g. upper lip smoother, eye better preserved

Computer Vision

Example of median

10 times 3 X 3 median

patchy effect

important details lost (e.g. ear-ring)

Question

For what types of noise would you clearly prefer median filtering over Gaussian filtering?

a) Gaussian noise, i.e. noise distributed by independent normal distribution
b) Salt and pepper noise
c) Uniform noise, i.e. distributed by uniform distribution
d) Exponential noise model
e) Rayleigh noise

Anisotropic diffusion : principle

non-linear filter

method :

- 1. Gaussian smoothing across homogeneous intensity areas
- 2. No smoothing across edges

The Gaussian filter revisited

The diffusion equation

$$\frac{\partial f(\vec{x},t)}{\partial t} = \nabla \cdot (c(\vec{x},t)\nabla f(\vec{x},t))$$

Initial/Boundary conditions

$$f(\vec{x},0) = i(x,y), \ \text{for } \vec{x} \in \Omega$$

$$f(\vec{x},t) = 0, \ \text{for } \vec{x} \in \delta(\Omega)$$

If $c(\vec{x},t) = c$

$$\frac{\partial f(\vec{x},t)}{\partial t} = c\Delta f(\vec{x},t)$$ in1D: $$\frac{\partial f(x,t)}{\partial t} = c\frac{\partial^2 f(x,t)}{\partial x^2}$$

Solution is a convolution!

$$f(\vec{x},t) = f(\vec{x},0) * g(\vec{x},t) = i(\vec{x}) * g(\vec{x},t)$$

Diffusion as Gaussian lowpass filter

$$f(\vec{x},t) = i(\vec{x}) * \frac{1}{(2\pi)^{d/2}\sqrt{ct}} \exp\left\{-\frac{\vec{x}\cdot\vec{x}}{4ct}\right\}$$

Gaussian filter with time dependent standard deviation: $\sigma = \sqrt{2ct}$

Nonlinear version can change the width of the filter locally

$$c(\vec{x},t) = c(f(\vec{x},t))$$

Specifically dependening on the edge information through gradients

$$c(\vec{x},t) = c(|\nabla f(\vec{x},t)|)$$

Selection of diffusion coefficient

$$c(|\nabla f(\vec{x},t)|) = \exp\left\{-\frac{|\nabla f|^2}{2\kappa^2}\right\}$$

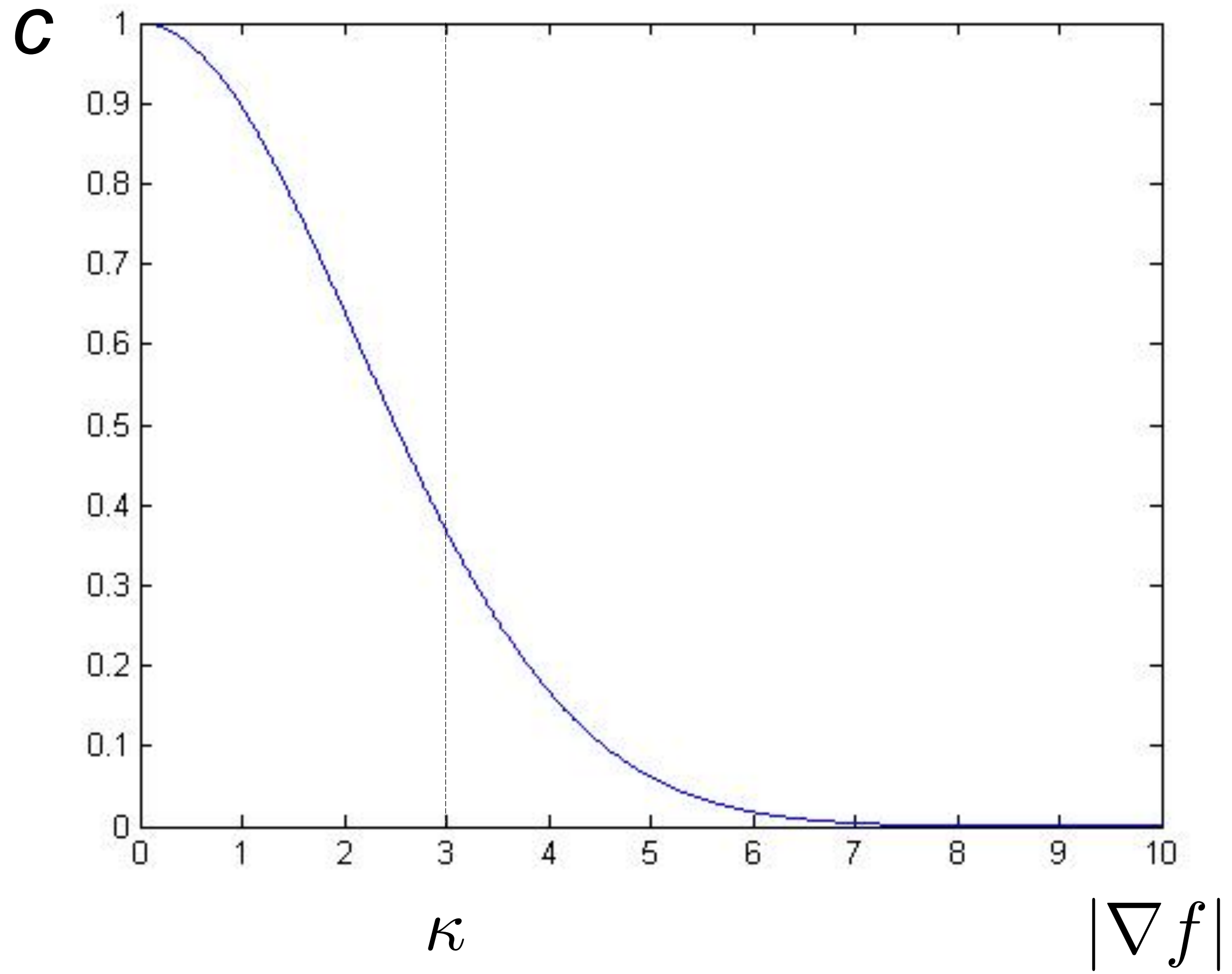
or

$$c(|\nabla f(\vec{x},t)|) = \frac{1}{1+\left(\frac{|\nabla f|}{\kappa}\right)^2}$$

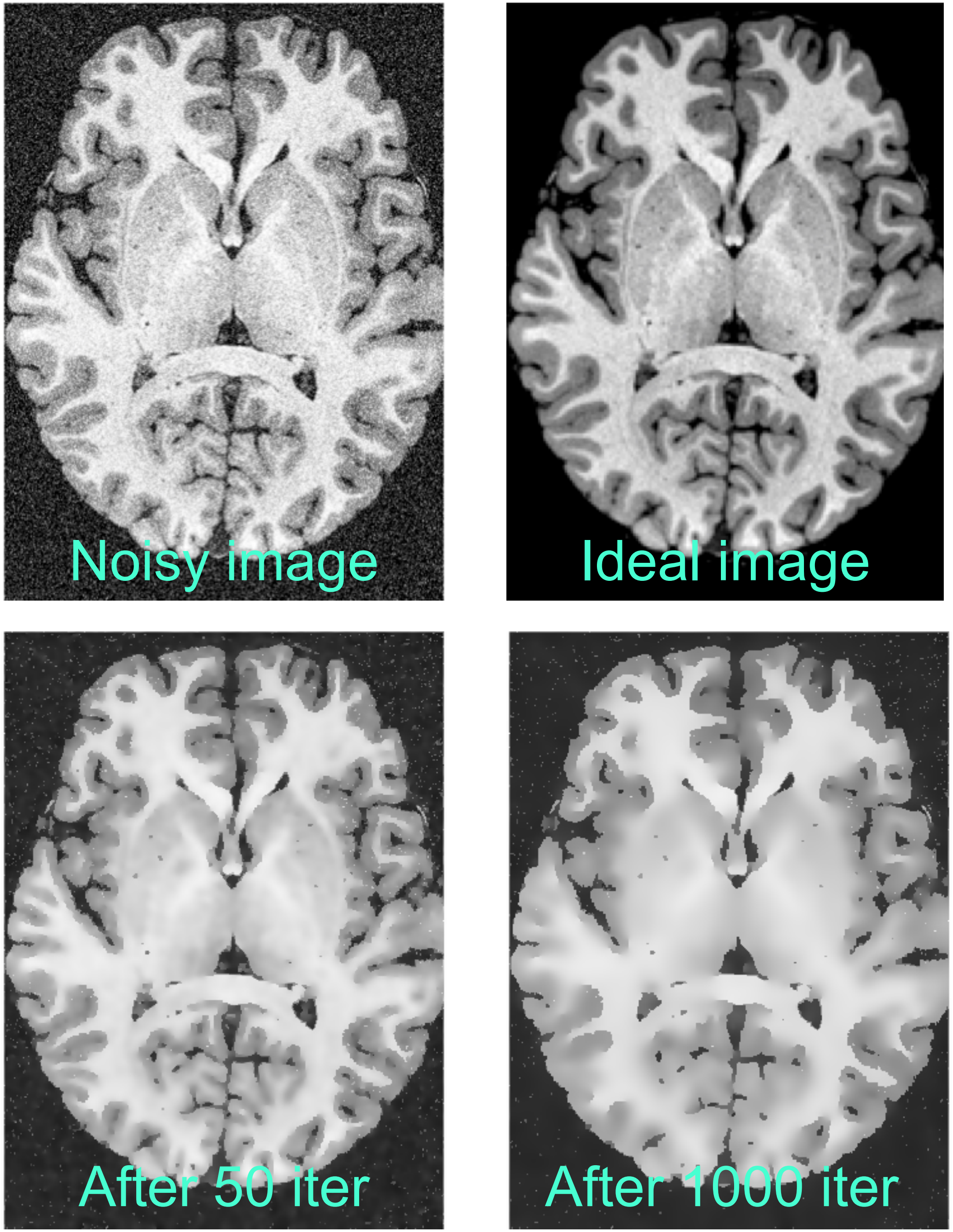
$\boldsymbol{\kappa}$ controls the contrast to be preserved by smooting actually edge sharpening happens

Computer Vision

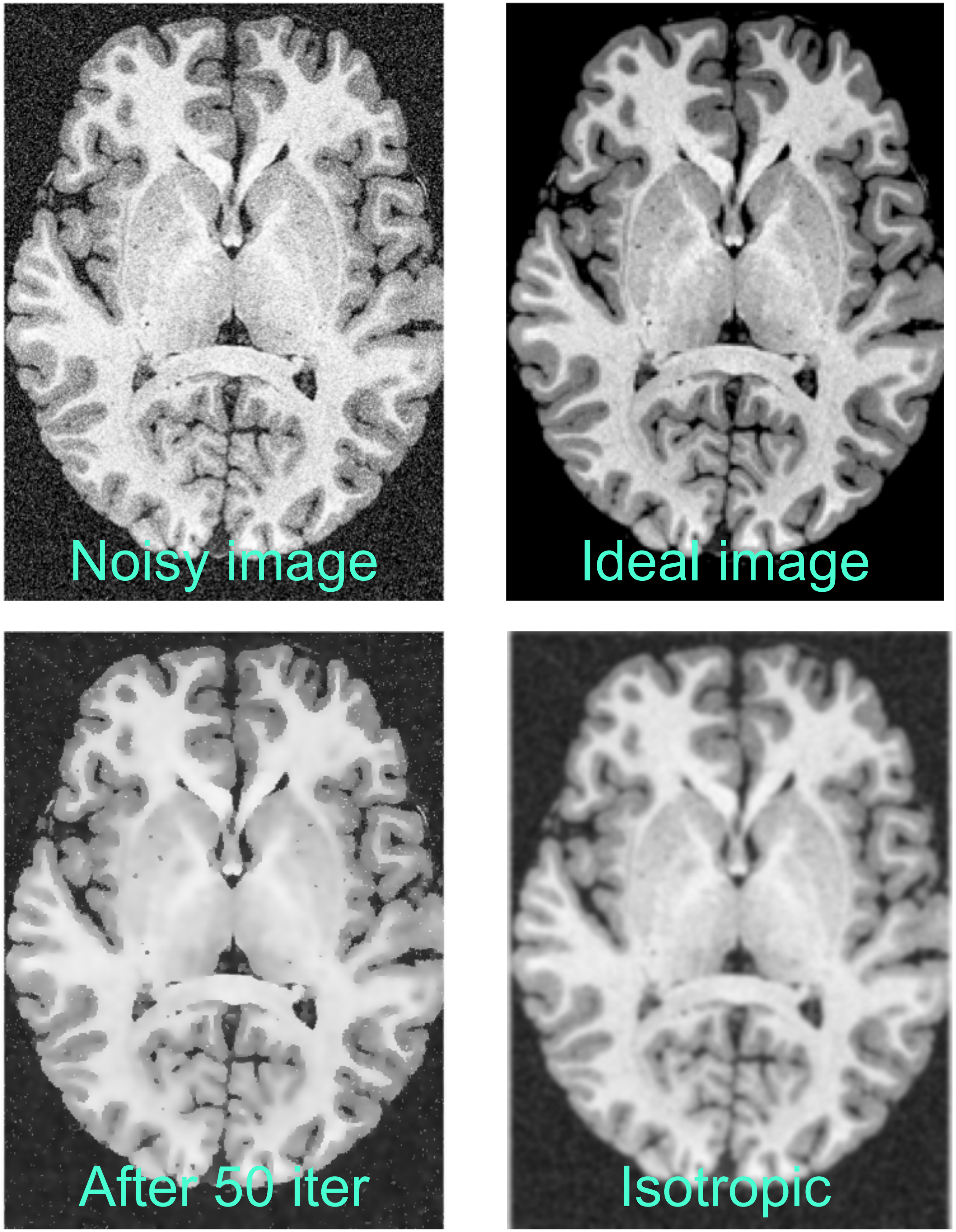
Dependence on contrast



Computer Vision



Computer Vision



Anisotropic diffusion: Output

End state is homogeneous

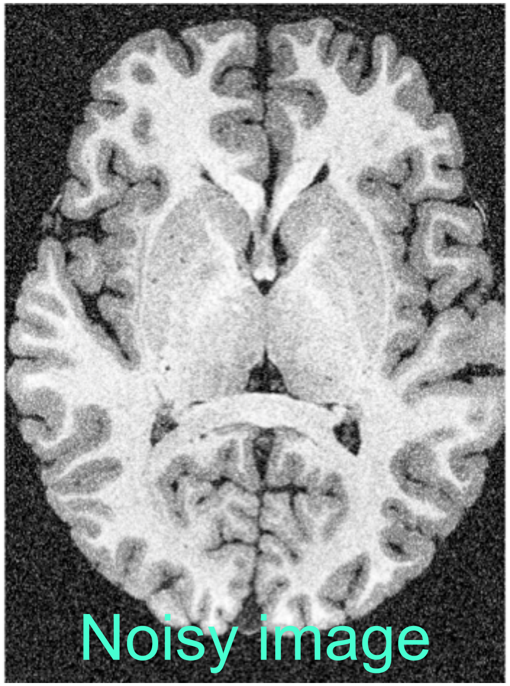
Computer Vision

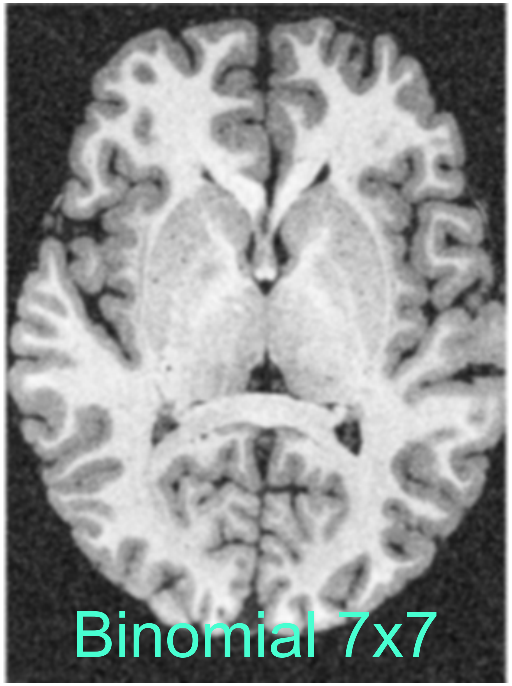
Restraining the diffusion

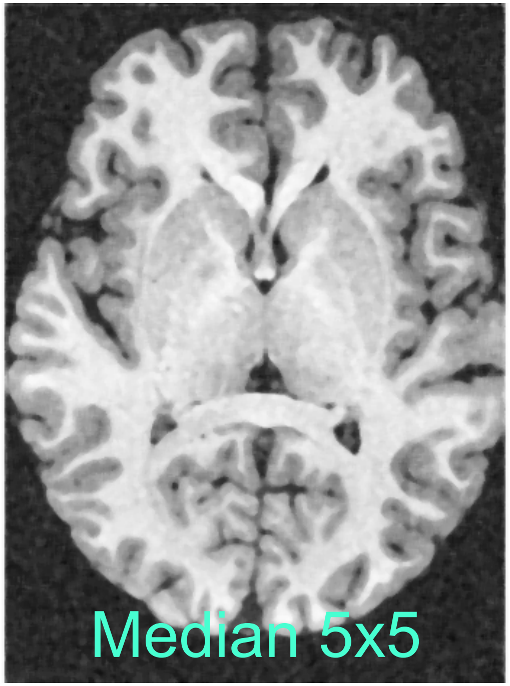
adding restraining force :

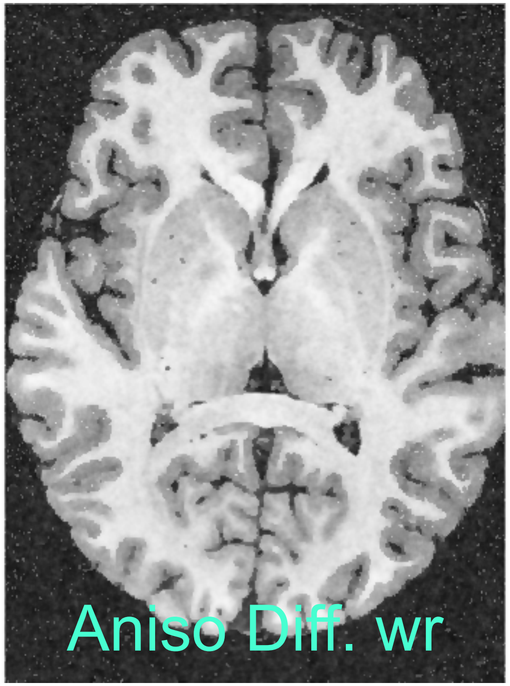
$$\frac{\partial f}{\partial t} = \Delta \cdot (c(|\nabla f|)\nabla f) - \frac{1}{\sigma^2}(f - i)$$

Computer Vision









Anisotropic diffusion: Numerical solutions

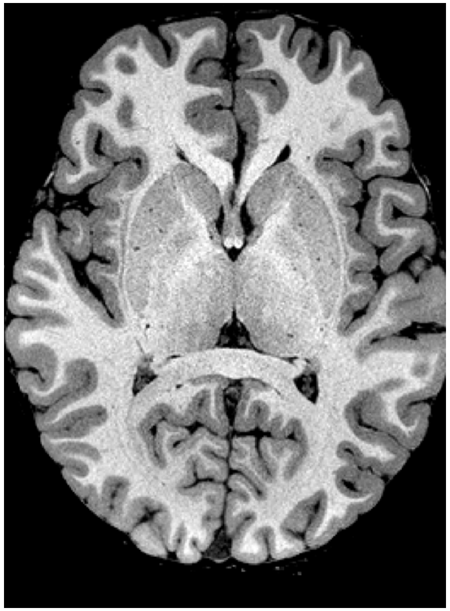
When c is not a constant solution is found through solving the equation

$$\frac{\partial f(\vec{x},t)}{\partial t} = \nabla \cdot (c(\vec{x},t)\nabla f(\vec{x},t))$$

Partial differential equation

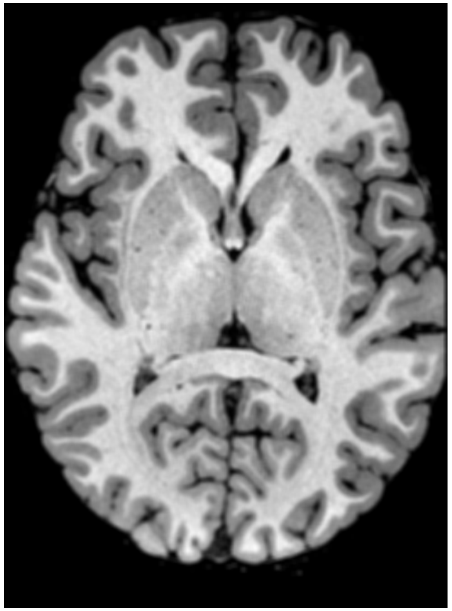
Numerical solutions through discretizing the differential operators and integrating

Finite differences in space and integration in time



Original Image
What we want

Deblurring



Blurred image
What we observe

Unsharp masking

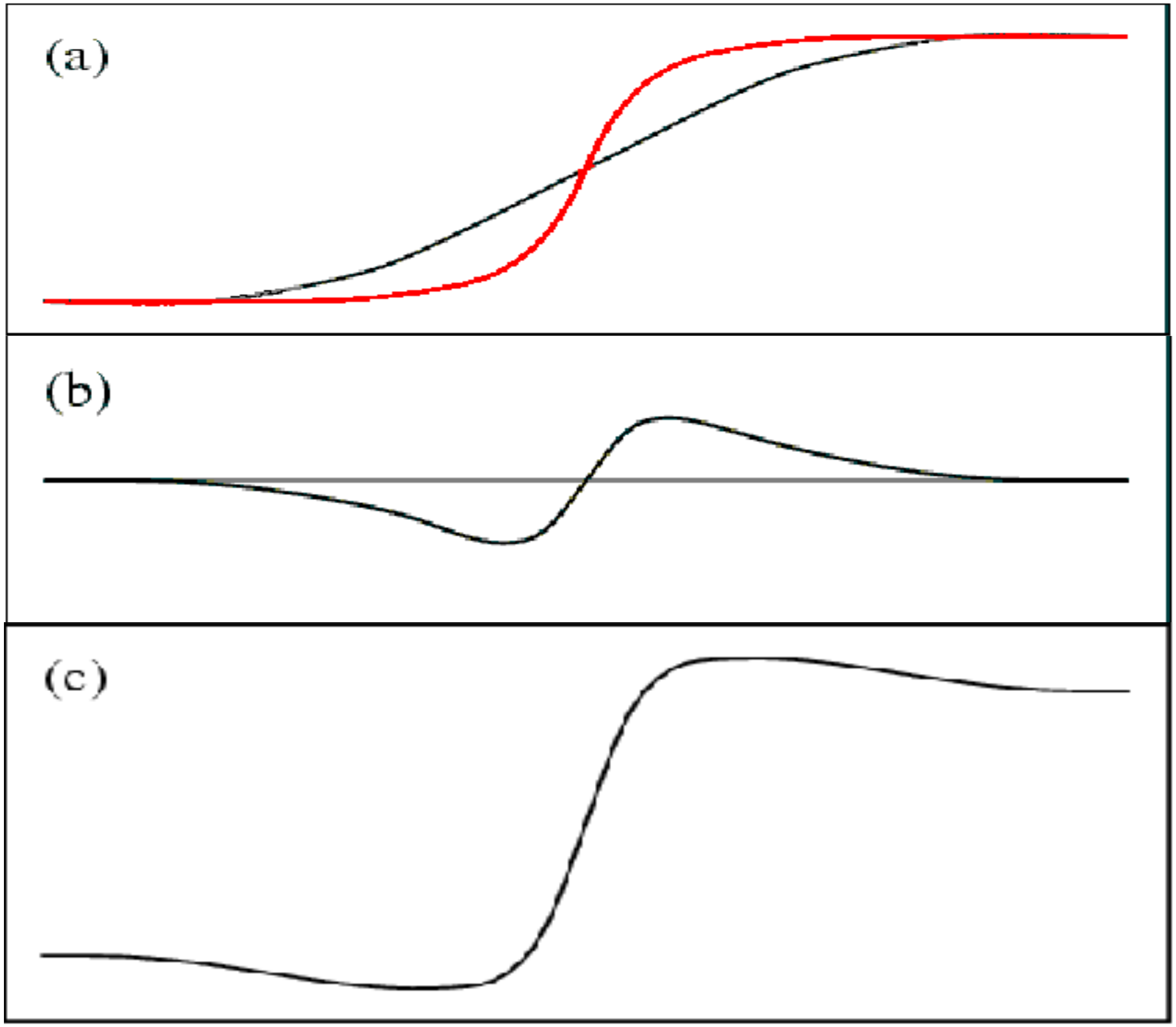
simple but effective method

image independent

linear

used e.g. in photocopiers and scanners

Unsharp masking : sketch



red = original
black = smoothed

orginal – smooth

original + difference

Unsharp masking : principle

Interpret blurred image as snapshot of diffusion process

$$\frac{\partial f}{\partial t} = c(\nabla^2 f)$$

In a first order approximation, we can write

$$f(x,y,t) \approx f(x,y,0) + \frac{\partial f}{\partial t} t$$

Hence,

$$f(x,y,0) \approx f(x,y,t) - \frac{\partial f}{\partial t} t = f(x,y,t) - ct\nabla^2 f$$

Unsharp masking produces *o* from *i*

$$o = i - k\nabla^2 i$$

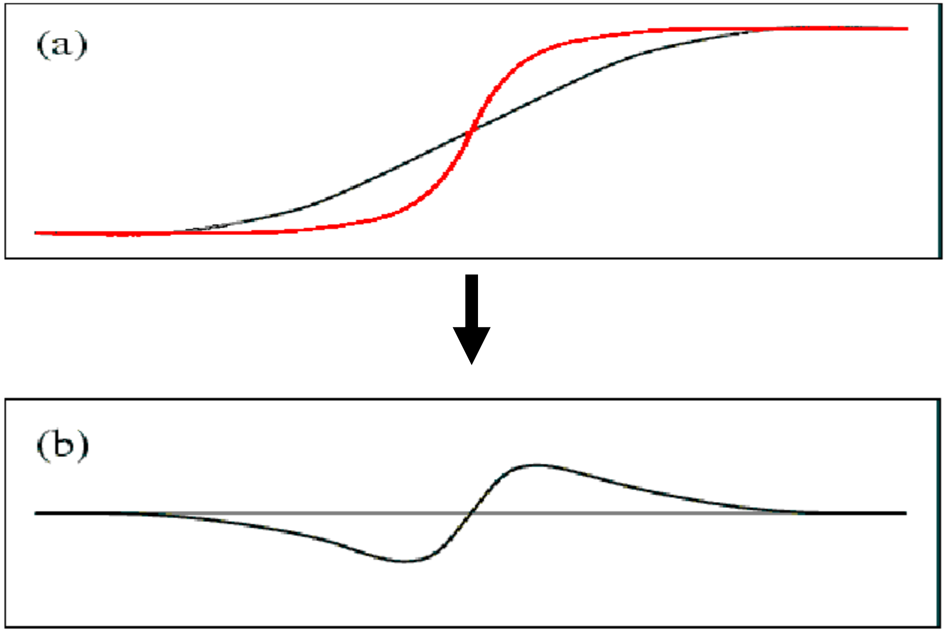
with *k* a well-chosen constant

Computer Vision

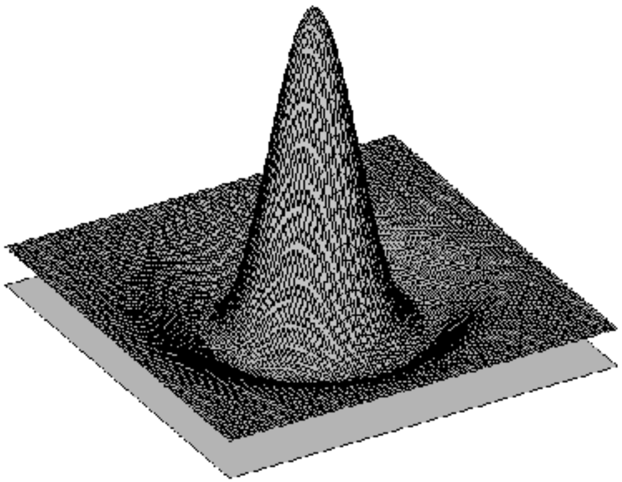
Need to estimate $\nabla^2 i(x,y)$

DOG (Difference-of-Gaussians) approximation for Laplacian :

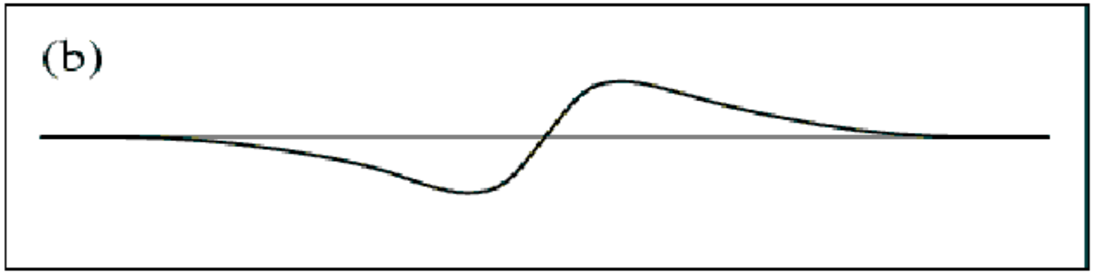
Our 1D example:



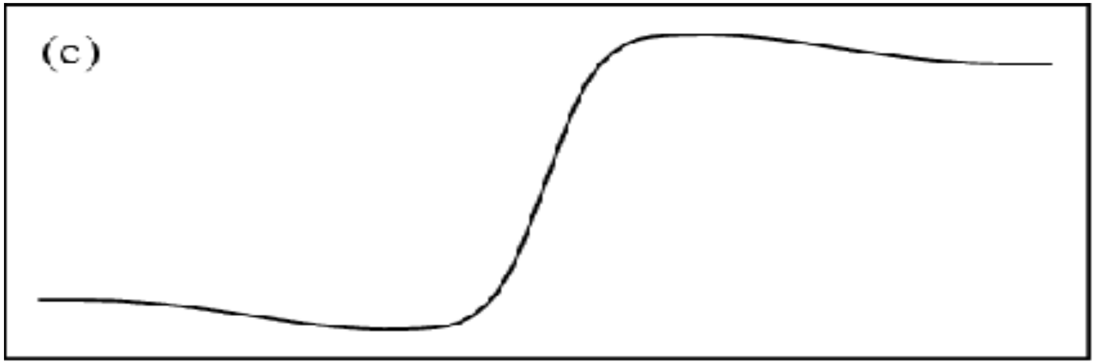
Convolution mask in 2D:



Unsharp masking: Analysis



$$o = i - k\nabla^2 i$$



The edge profile becomes steeper, giving a sharper impression

Under-and overshoots flanking the edge further increase the impression of image sharpness

Computer Vision

Unsharp masking : images

Inverse filtering

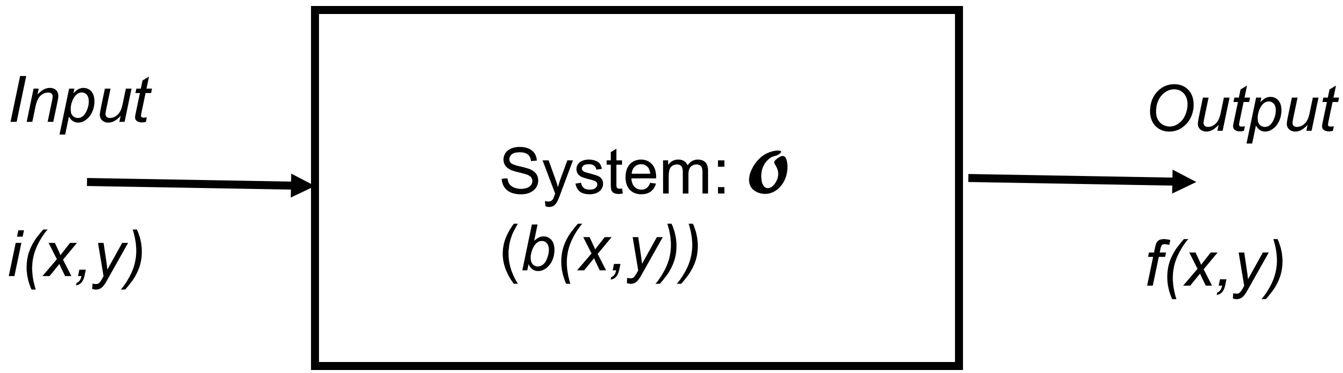
Relies on system view of image processing

Frequency domain technique

Defined through Modulation Transfer Function

Links to theoretically optimal approaches

A system view on image restoration



i,b known *f*=?: simulation, smoothing
i,f known *b*=?: system identification
b,f known *i*=?: image restoration

for de-blurring: b is the blurring filter

Inverse filtering : principle

Frequency domain technique

suppose you know the MTF $B(u,v)$ of the blurring filter

$$f(x,y) = b(x,y) * i(x,y)$$

$$F(u,v) = B(u,v)I(u,v)$$

to undo its effect new filter with MTF $B'(u,v)$ such that

$$B'(u,v)B(u,v) = 1$$

$$I(u,v) = B'(u,v)F(u,v)$$

Inverse filtering : formal derivation

$$B'(u,v) = 1/B(u,v)$$

For additive noise after filtering

$$F(u,v) = B(u,v)I(u,v) + N(u,v)$$

Result of inverse filter

$$F(u,v)B'(u,v) = I(u,v) + N(u,v)/B(u,v)$$

Problems of inverse filtering

$$F(u,v) = B(u,v)I(u,v) + N(u,v)$$

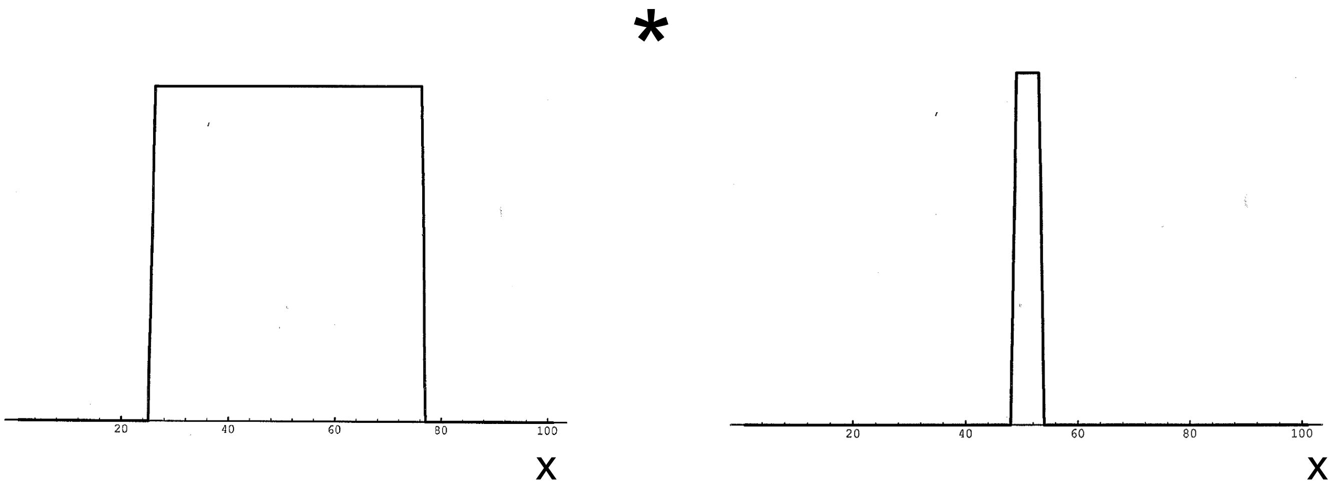
- Frequencies with $B(u,v) = 0$
  Information fully lost during filtering
  Cannot be recovered
  Inverse filter is ill-defined

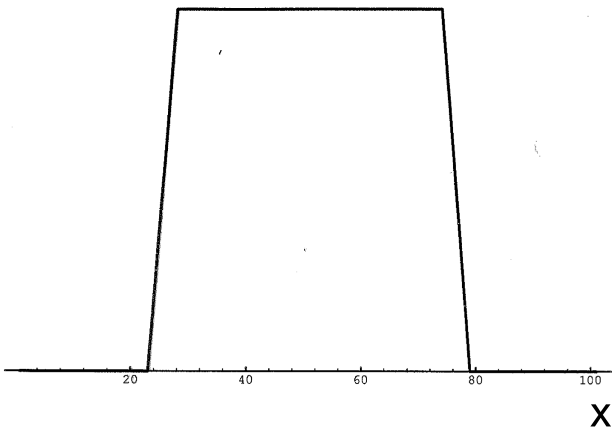
$$F(u,v)B'(u,v) = I(u,v) + N(u,v)/B(u,v)$$

- Also problem with noise added after filtering
  $B(u,v)$ is low -> $1/B(u,v)$ is high,
  VERY strong noise amplification

Computer Vision

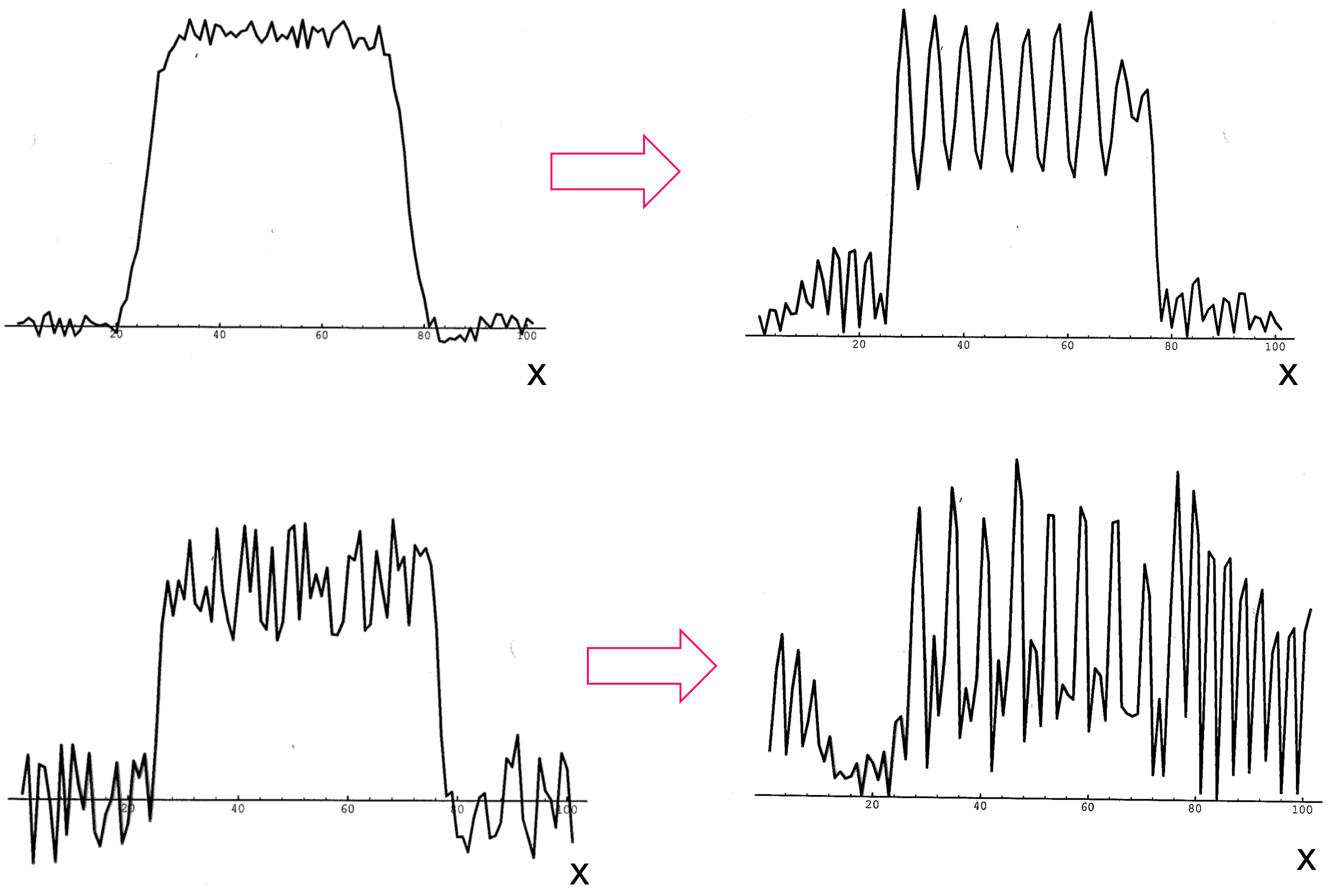
1D Example





Computer Vision

Restoration of noisy signals



Inverse filtering : 2D example

we will apply the method to a Gaussian smoothed example (σ = 16 pixels)

Inverse filtering : 2D example

noise leads to spurious high frequencies

The Wiener Filter

Looking for the optimal filter to do the deblurring

Take into account the noise to avoid amplification

Optimization formulation

Filter is given analytically in the Fourier Domain

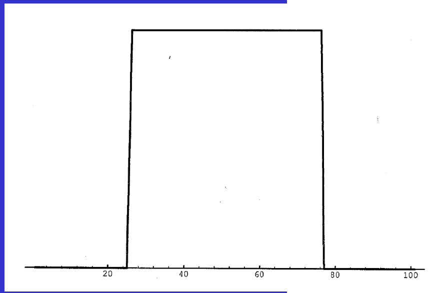
Wiener filter and its behavior

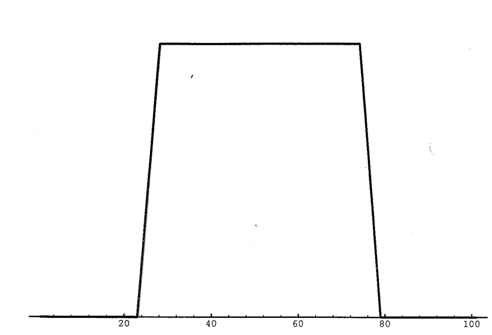
$$Wf(H) = H'(u,v) = \frac{H(u,v)}{H^*(u,v)H(u,v) + 1/\mathrm{SNR}}$$

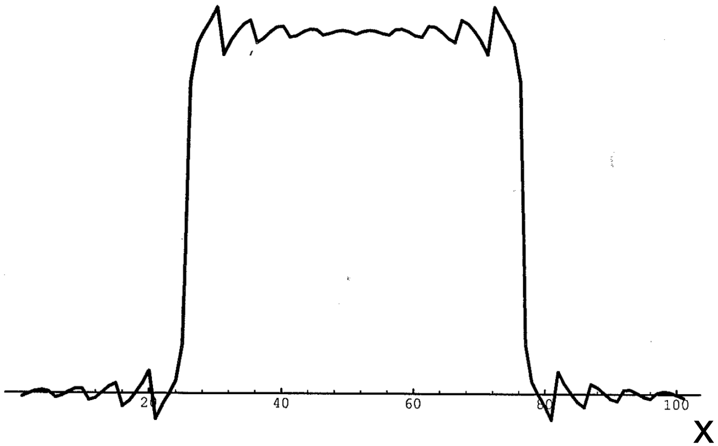
$$\mathrm{SNR} = \frac{\Phi_{ii}}{\Phi_{nn}}$$

- $H(u,v) = 0 \implies Wf(H) = 0$ ✓
- $\mathrm{SNR} \to \infty \implies 1/\mathrm{SNR} \to 0$, $Wf(H) \to \frac{1}{H}$ ✓
- $\mathrm{SNR} \to 0 \implies 1/\mathrm{SNR} \to \infty$, $Wf(H) \to 0$ ✓

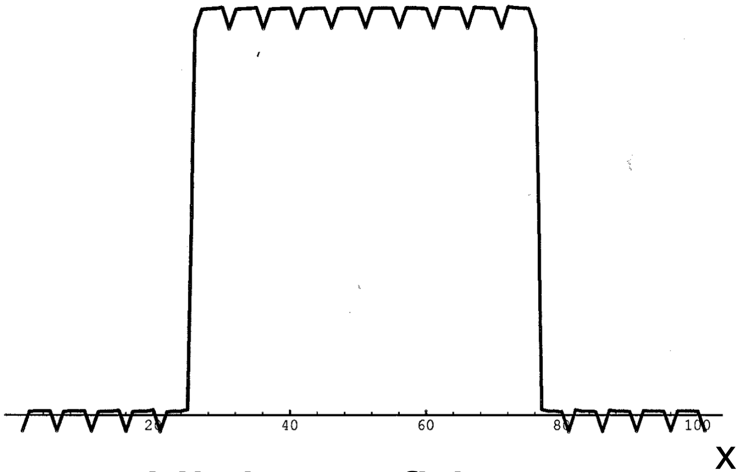
Wiener filter: Noiseless reconstruction







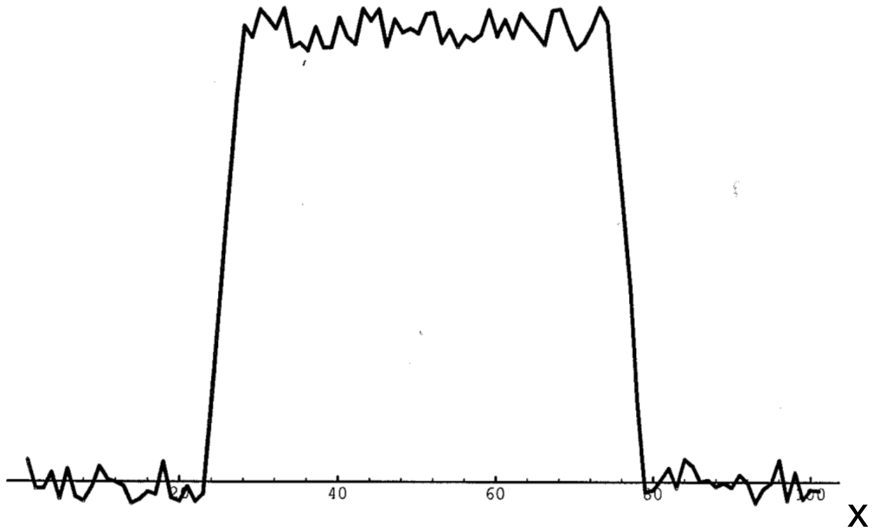
Medium confidence

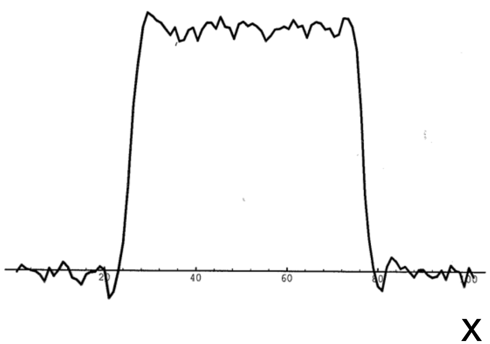


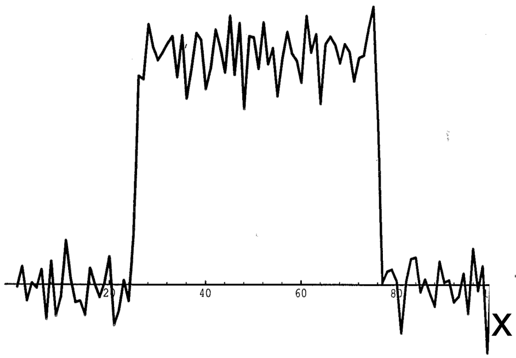
High confidence

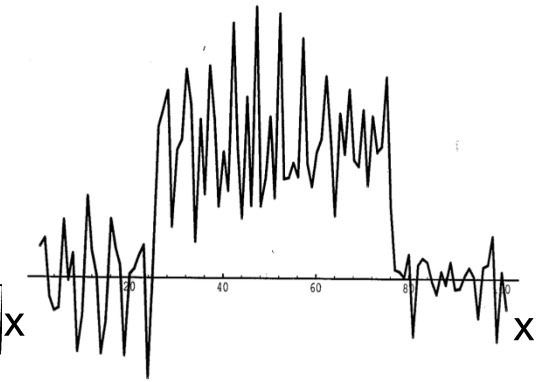
Computer Vision

Wiener filter: Noisy reconstruction









Low confidence

Medium confidence
Correct SNR

High confidence

Computer Vision

Wiener filtering : example

spurious high freq. eliminated, conservative

Wiener filter: problems of application

$$O(u,v) = Wf(H)(H(u,v)I(u,v))$$
$$= (Wf(H)H(u,v))I(u,v)$$

$Ef = Wf(H)H$ is the effective filter (should be 1)

- Conservative

  if SNR is low tends to become low-pass blurring instead of sharpening

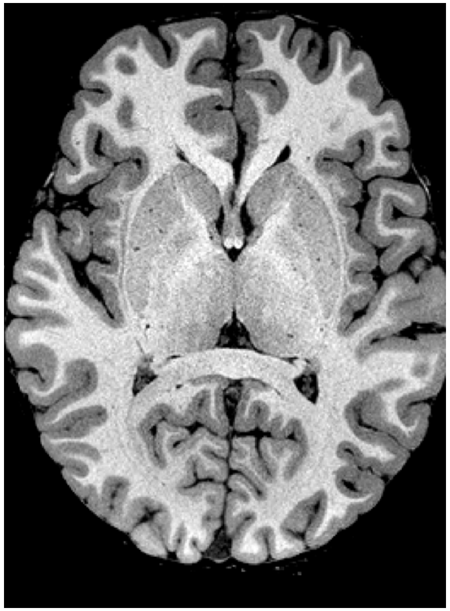
- $SNR = \Phi_{ii}(u,v)/\Phi_{nn}(u,v)$ depends on $I(u,v)$

  strictly speaking is unknown

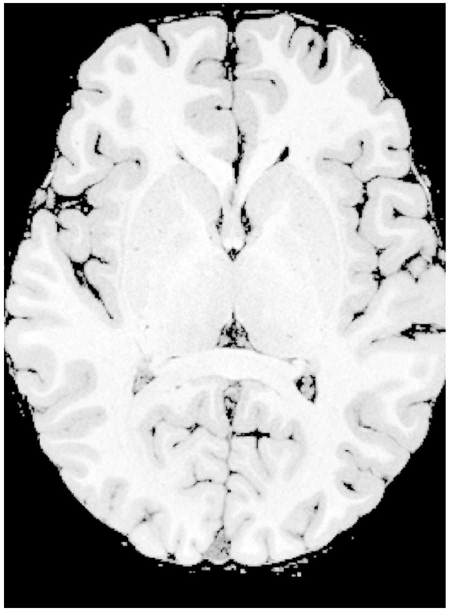
  power spectrum is not very characteristic

- $H(u,v)$ must be known very precisely

Computer Vision



Contrast
Enhancement



Original Image

Observation with Bad Contrast

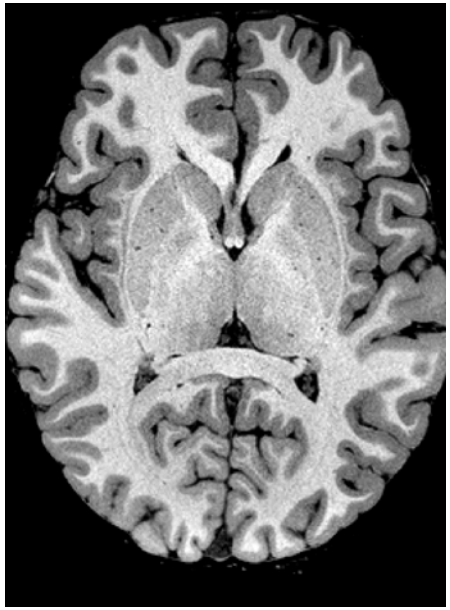
Contrast enhancement

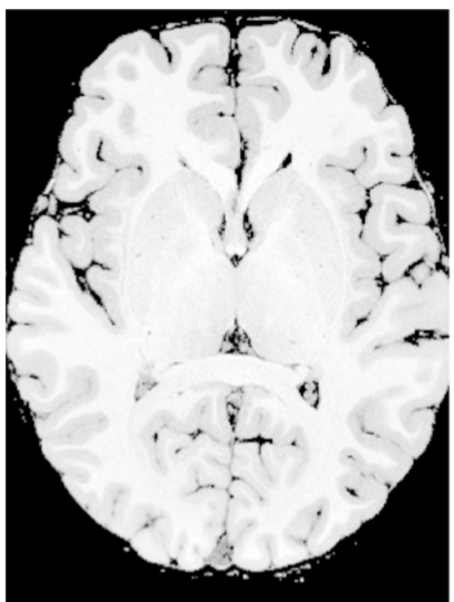
Use 1 : compensating under-, overexposure

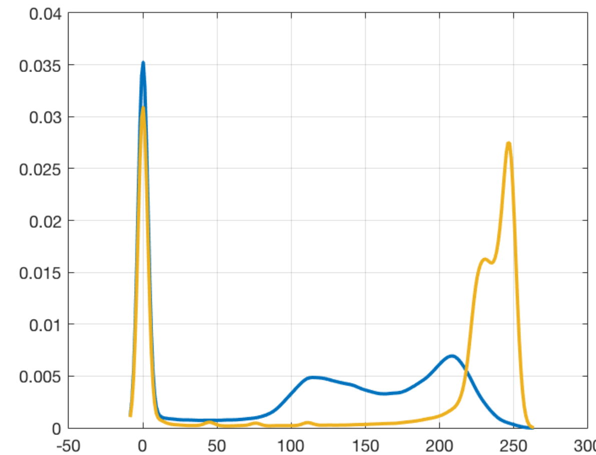
Use 2 : spending intensity range on interesting part of the image

We'll study *histogram equalisation*

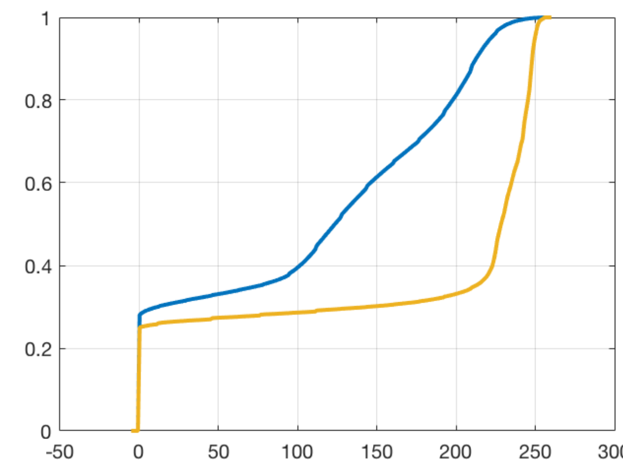
Intensity distribution







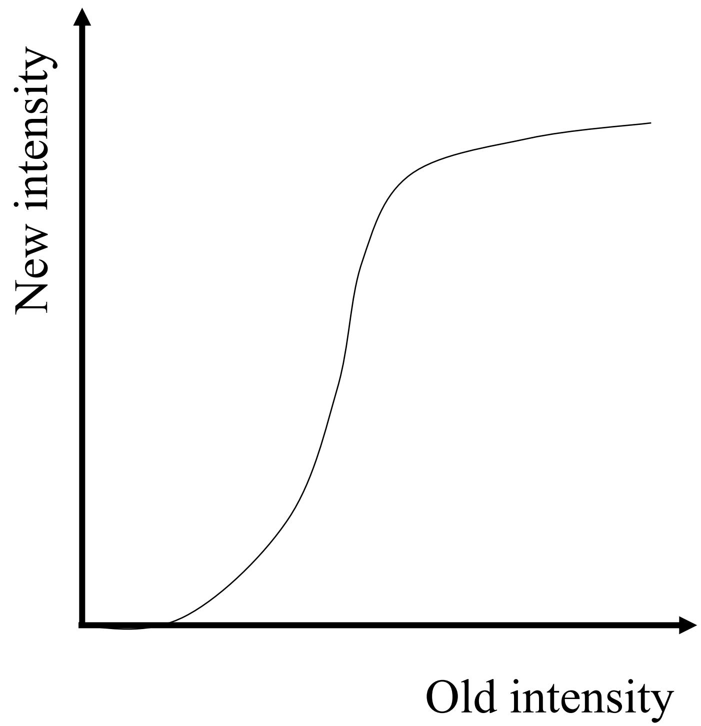
Histogram



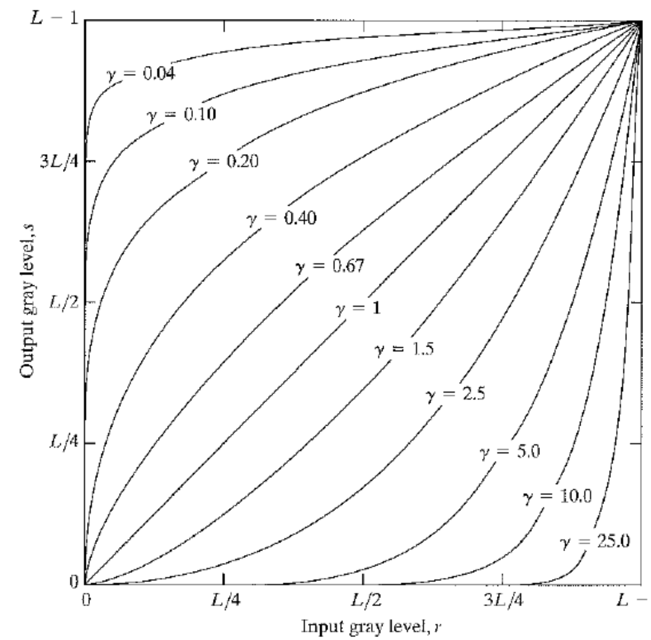
Cumulative histogram

Intensity mappings

Usually monotonic mappings required



Generic transformation function

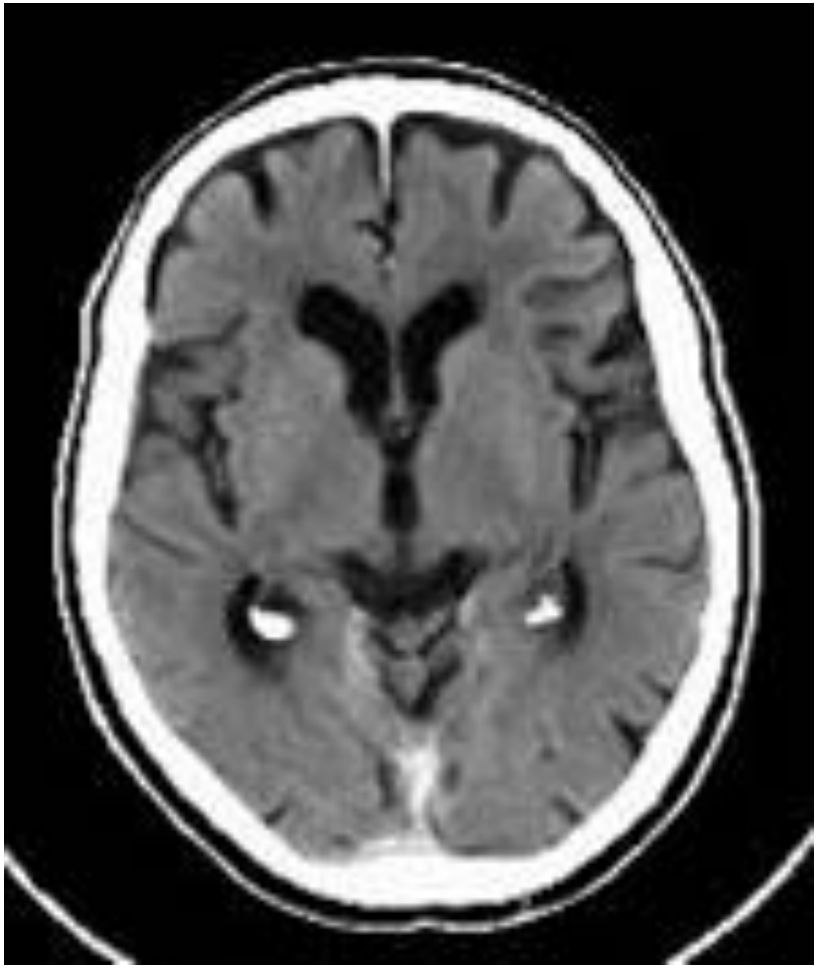


Power law transformation

$$I_{new} = I_{old}^{\gamma}$$

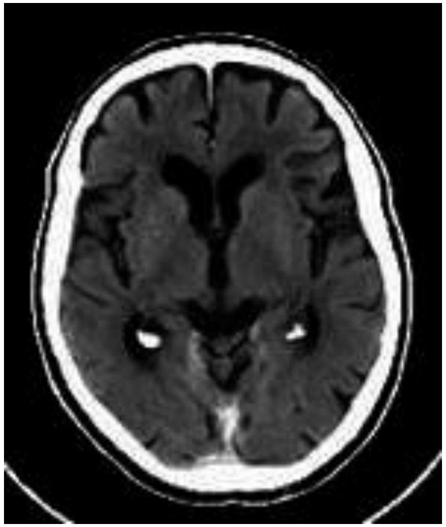
Computer Vision

Gamma correction



Original

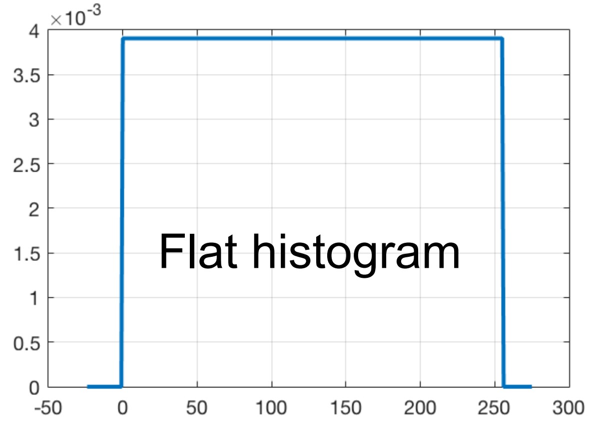
$\gamma = 2$

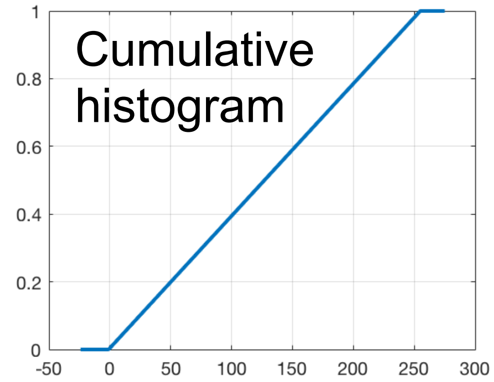


$\gamma = 0.5$

HISTOGRAM EQUALISATION

WHAT : **create a flat histogram**



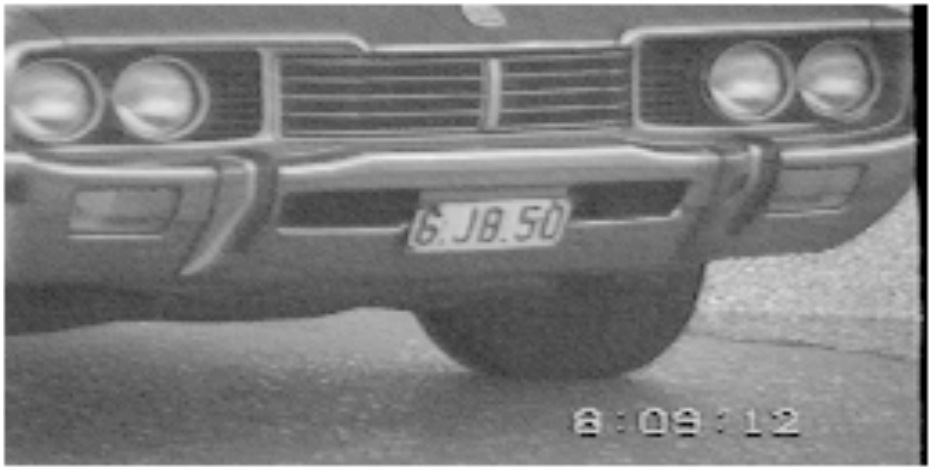


HOW : apply an appropriate intensity map depending on the image content

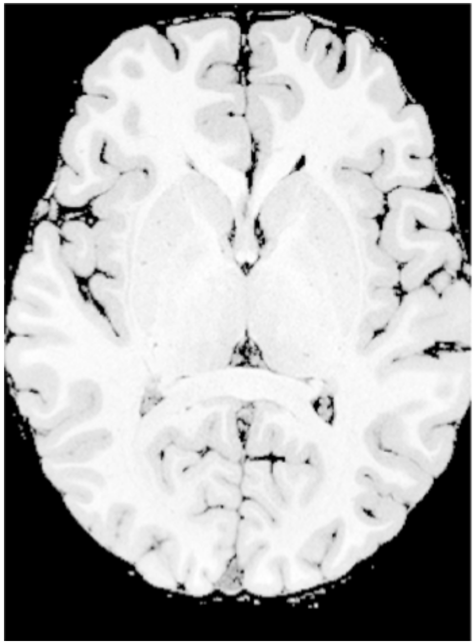
method will be generally applicable

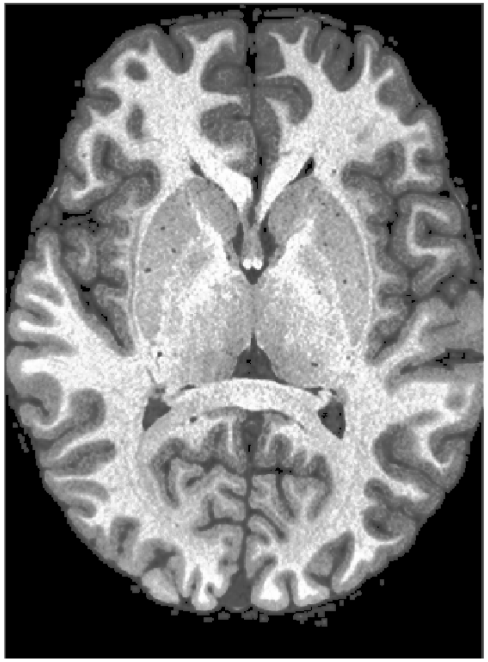
Computer Vision

Histogram equalisation : example



Histogram equalisation : example

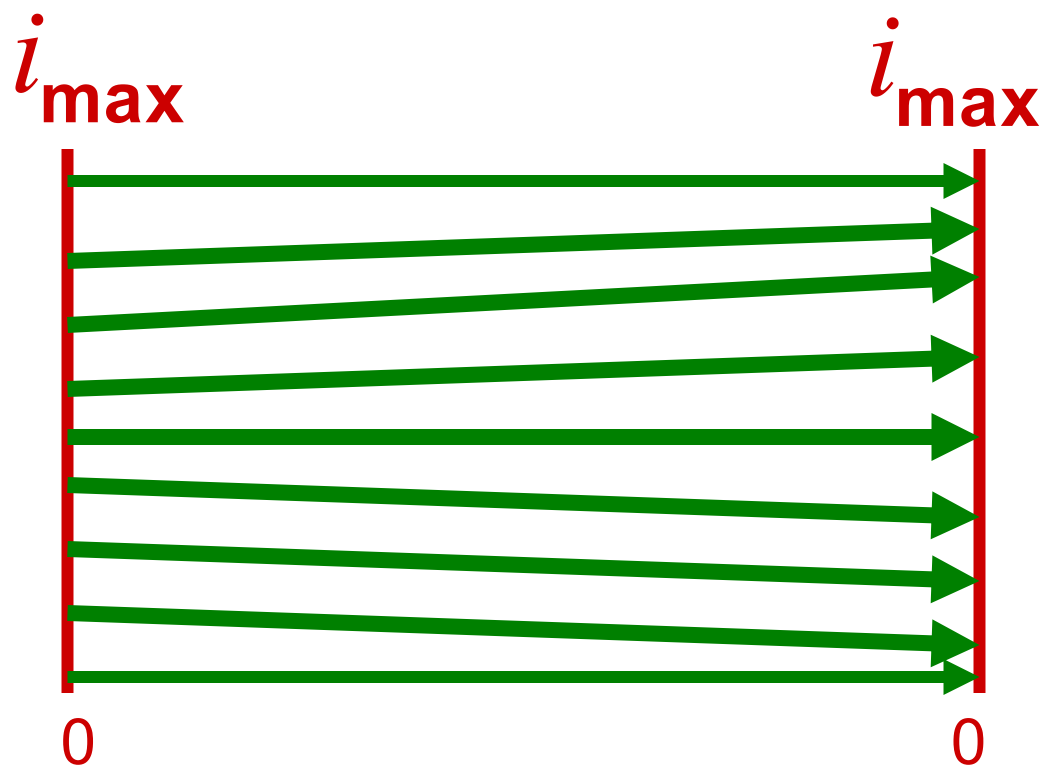




Histogram equalisation : principle

Redistribute the intensities, 1-to-several (1-to-1 in the continuous case) and keeping their relative order, as to use them more evenly

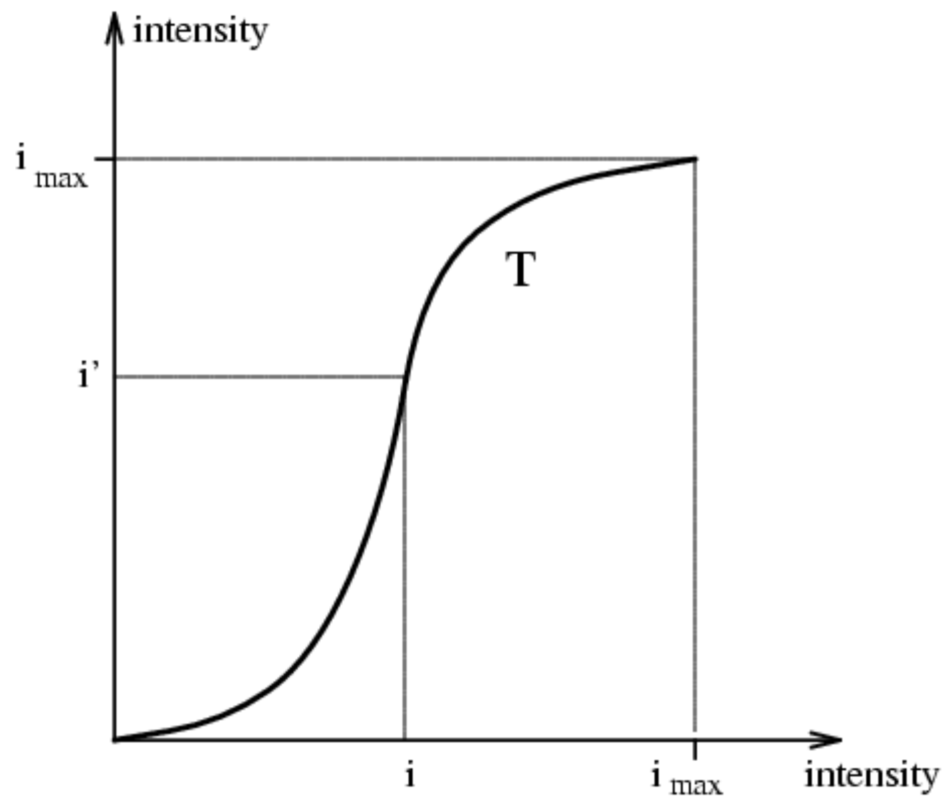
Ideally, obtain a constant, flat histogram



Histogram equalisation : algorithm

This mapping is easy to find:

It corresponds to the cumulative intensity probability,
i.e. by integrating the histogram from the left

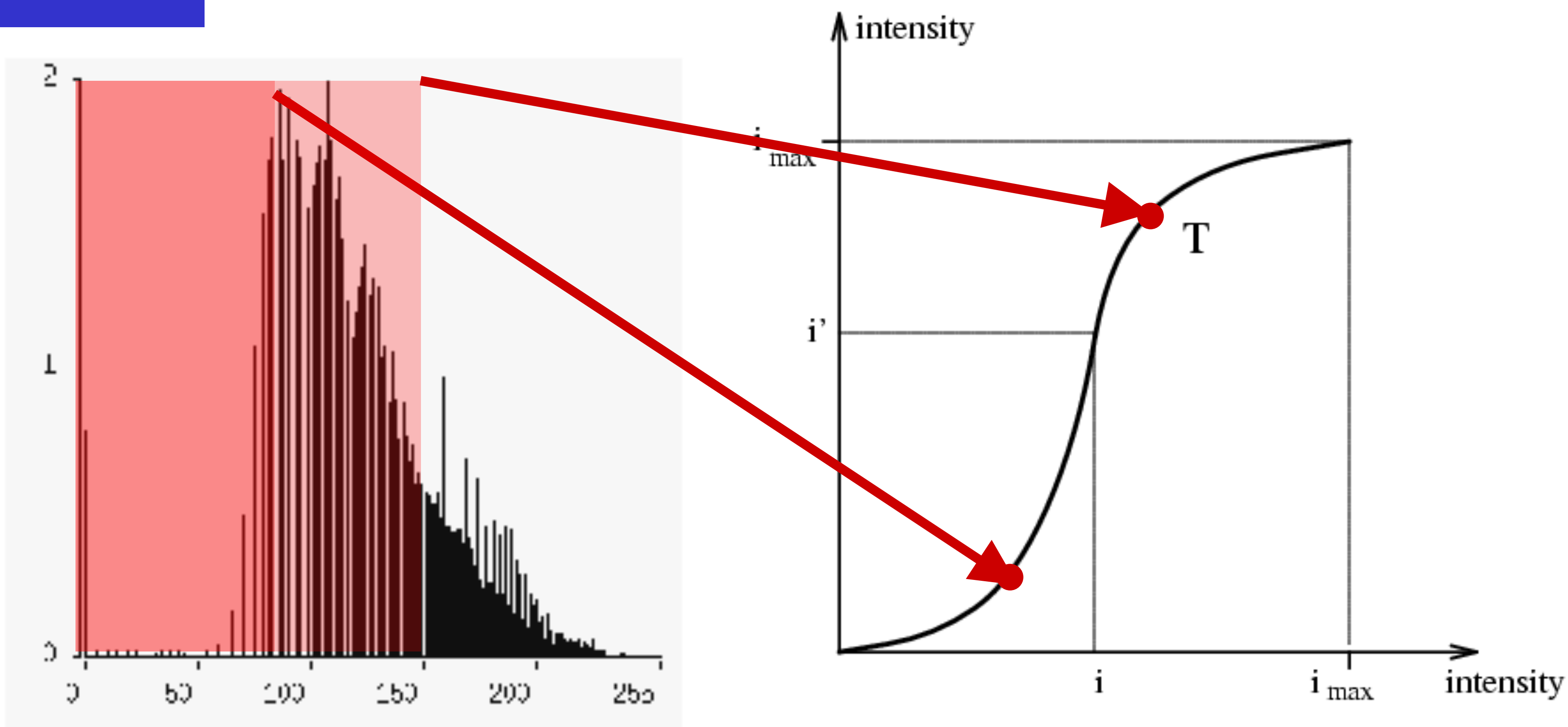


Histogram equalisation : algorithm

This mapping is easy to find:

It corresponds to the cumulative intensity probability,

i.e. by integrating the histogram from the left



Histogram equalisation : algorithm

suppose continuous probability density $p(i)$

cumulative probability distribution :

$$P(i) = \int_0^i p(i^*)di^*$$

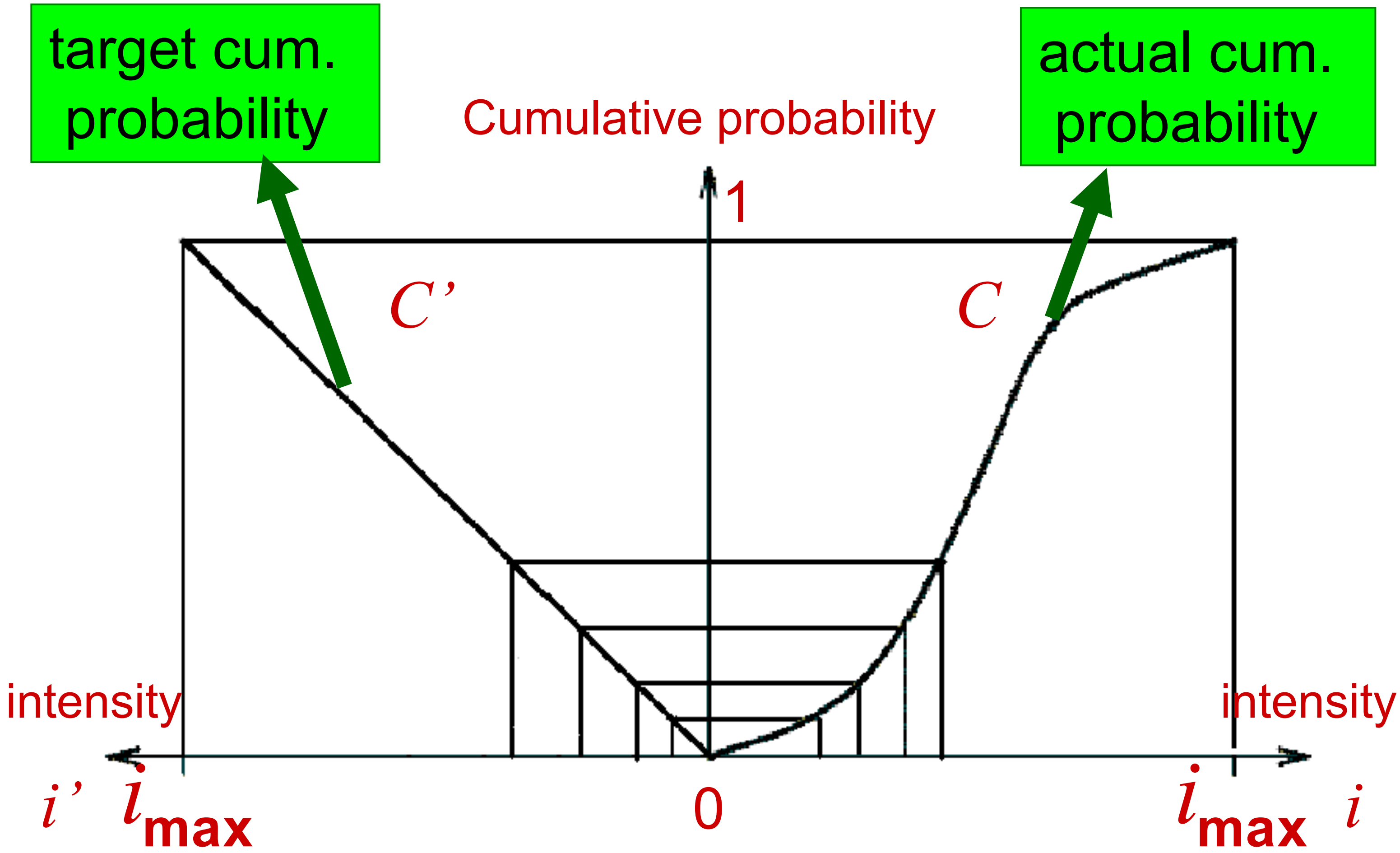
distribution as our map $T(i)$:

$$i' = T(i) = i_{\max} \int_0^i p(i^*)di^*$$

$$p' = p\frac{di}{di'} = p(\frac{1}{p})(\frac{1}{i_{\max}}) = \frac{1}{i_{\max}}!!!$$

Computer Vision

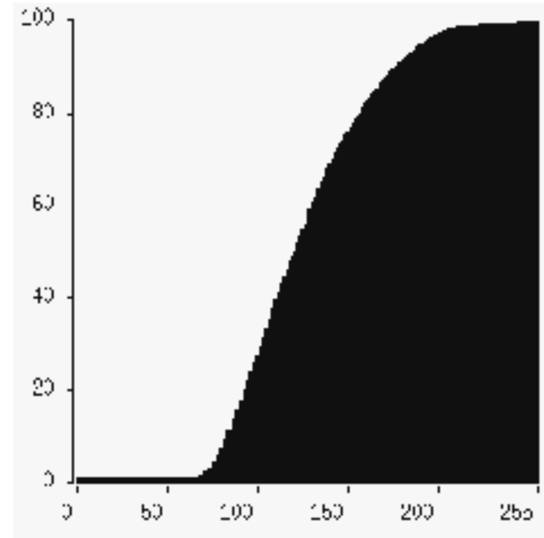
Histogram equalisation : sketch



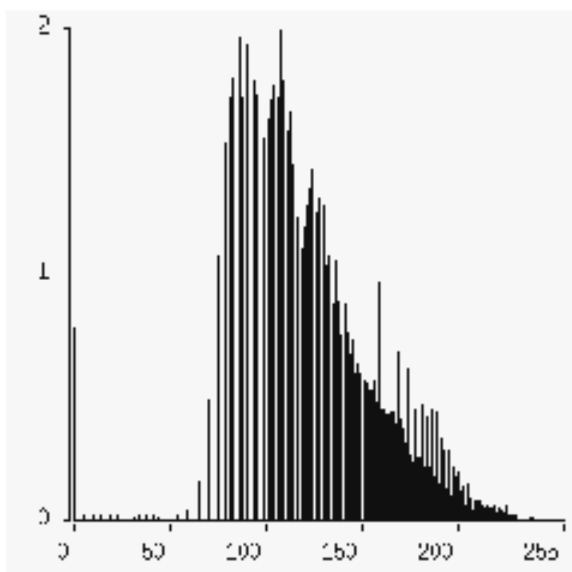
$$i' = T(i) = i_{max} C(i) = i_{max} \int_0^i p(i^*) di^*$$

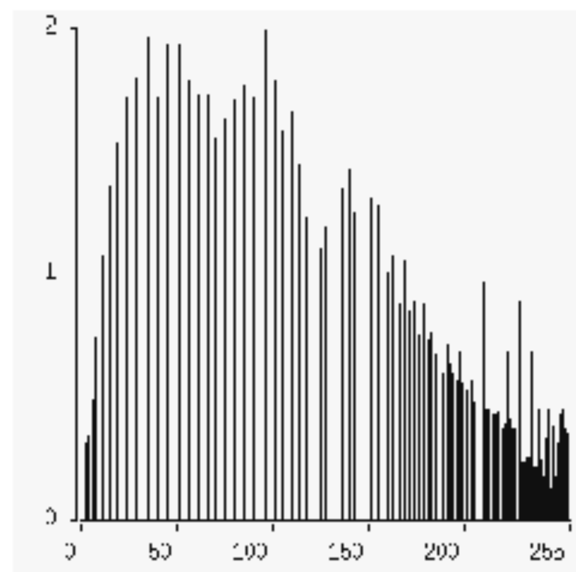
Histogram equalisation : result

intensity map :



original and flattened histograms :

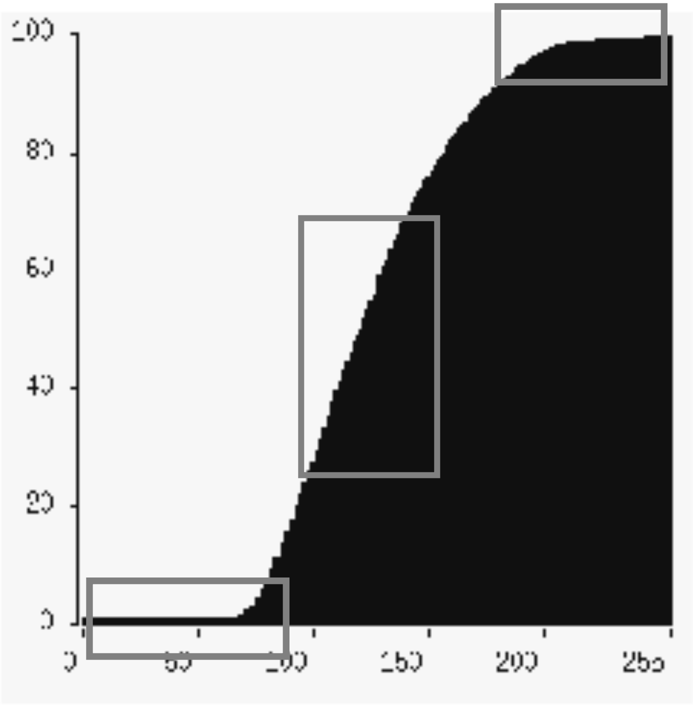




Computer Vision

Histogram equalisation : analysis

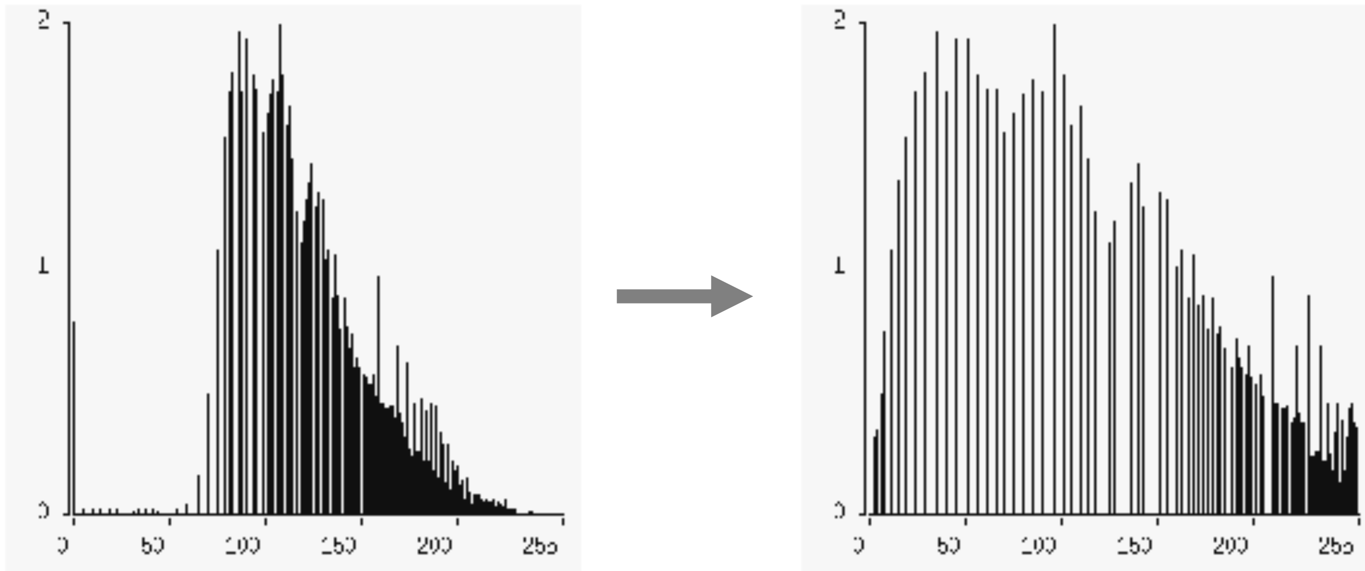
Intervals where many pixels are packed together are expanded



Intervals with only few corresponding pixels are compressed

Histogram equalisation : analysis

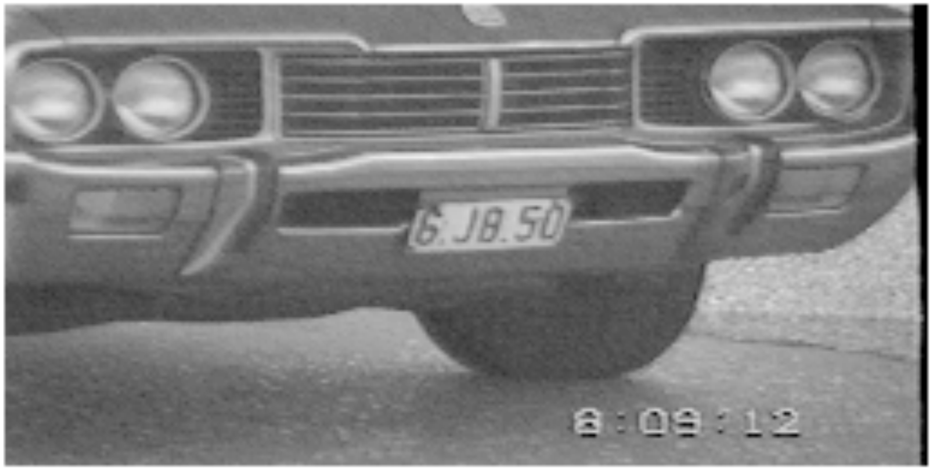
… BUT we don’t obtain a flat histogram



This is due to the discrete nature of the input histogram and the equalisation procedure

Jumps in the discretised cumulative probability distribution lead to gaps in the histogram

Histogram equalisation : example revisited



Histogram equalisation : generalisation

Find a map $i' = T(i)$ that yields probability density p'

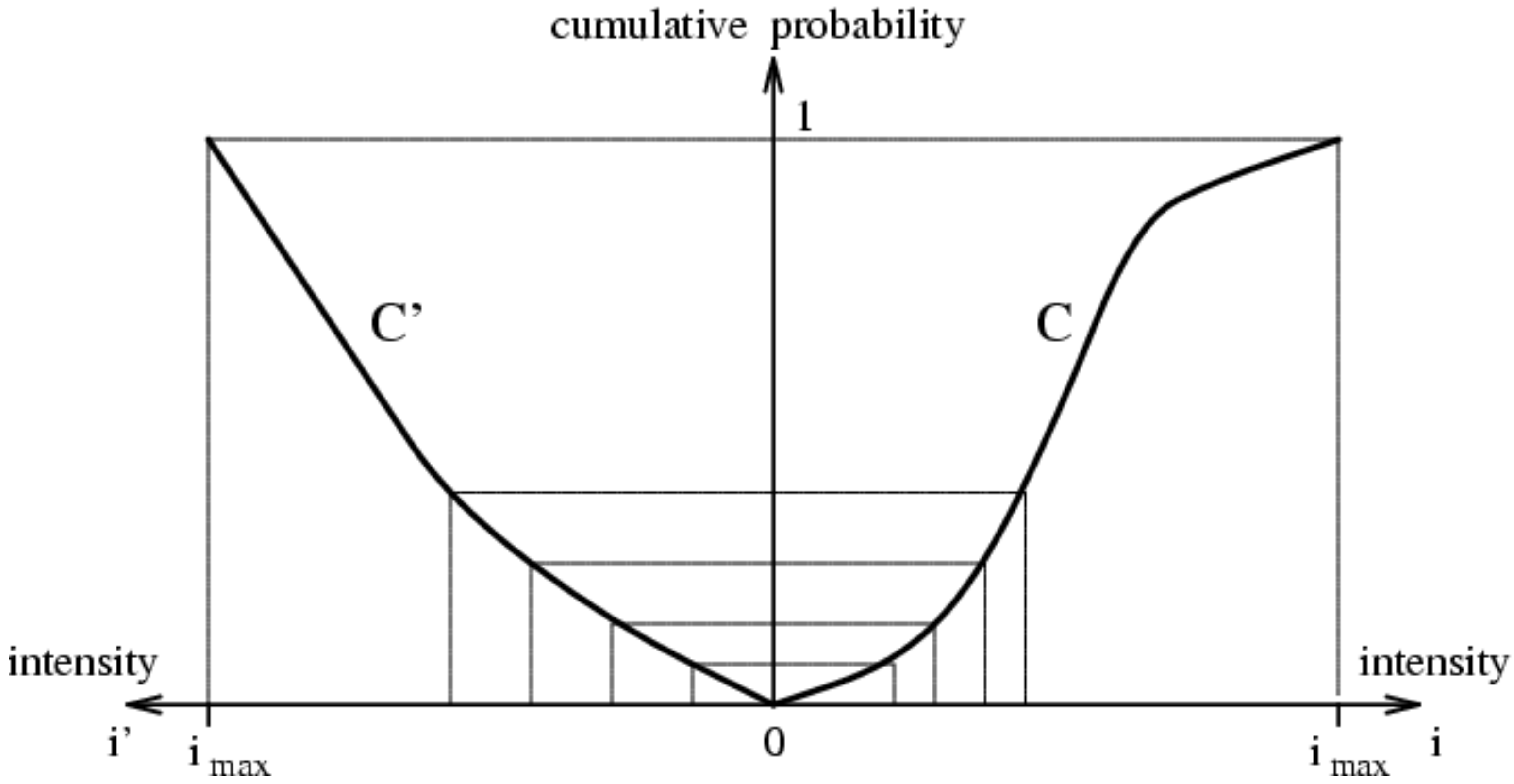
$$C'(i') = \int_0^{i'} p'(w)dw = \int_0^{i} p(v)dv = C(i).$$

with $C'(i')$ and $C(i)$ the prescribed and original cumulative probability distributions

Thus

$$i' = C'^{-1}(C(i))$$

Histogram equalisation : sketch



$$i' = C'^{-1}(C(i))$$