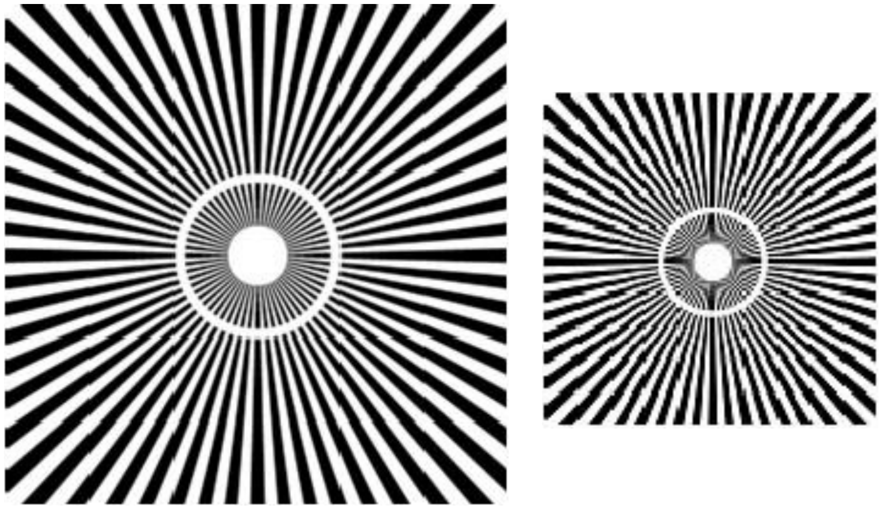
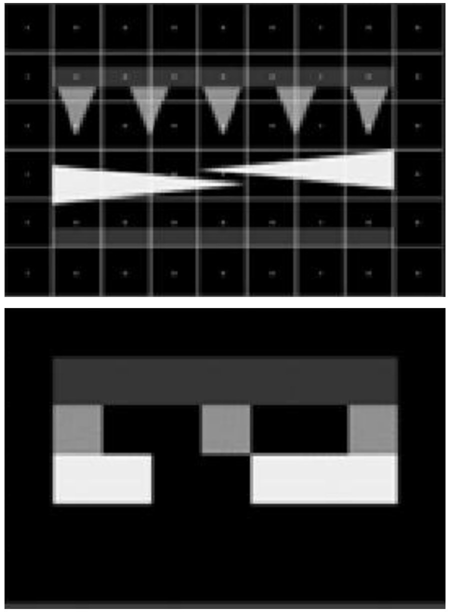
Image features I

Mårten Björkman

Image rescaling

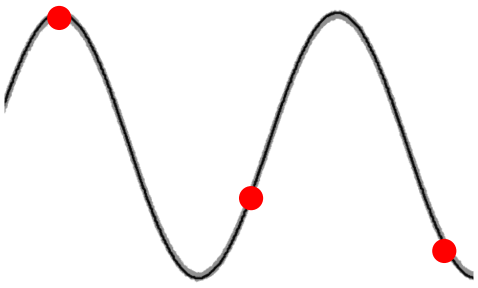
- If you rescale an image by just resampling the pixels, e.g. picking every second pixel, you easily get aliasing effects due to under-sampling.
- Fine structures can no longer be represented. Instead you see patterns that should not really be there.

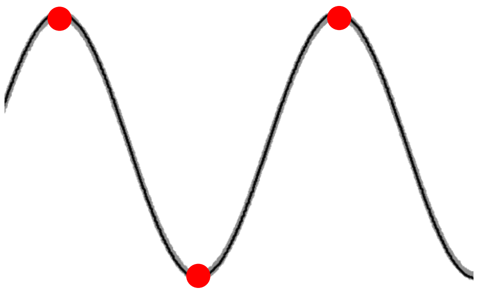


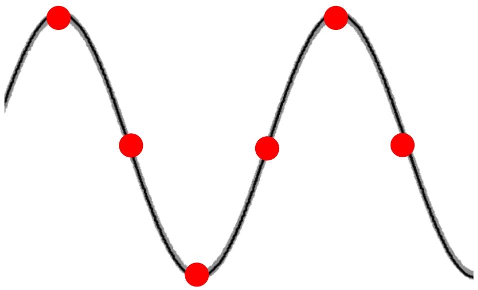


The Sampling Theorem

- The highest frequency that exists in a signal (or image) is called the bandwidth.
- In order to see all the variations in the image, the sampling rate needs to be at least twice the bandwidth, the Nyquist Rate.
- In practise this means, that you need at least two samples for each period of the corresponding sine wave, one dark sample and one white sample.

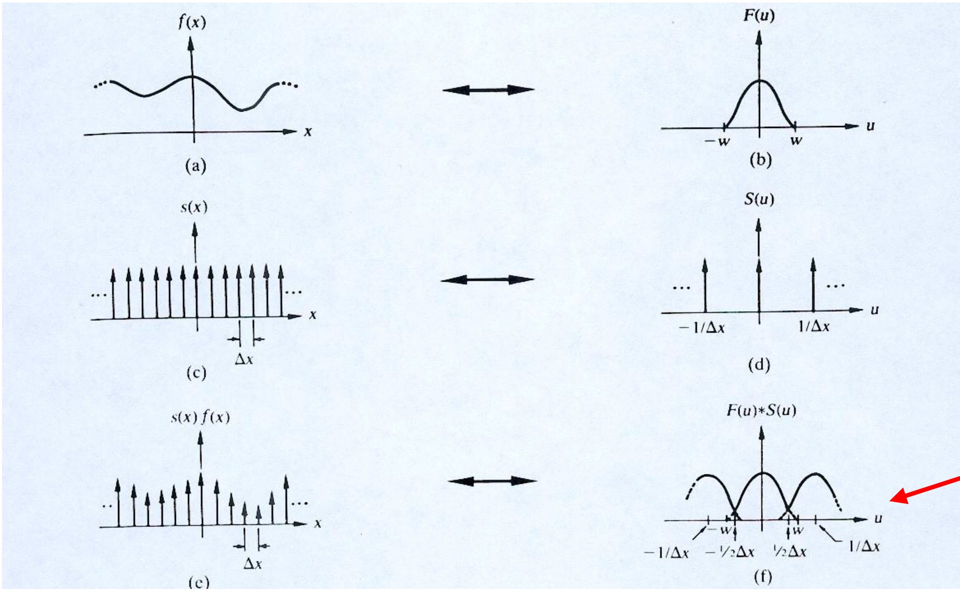






What happens during sampling?

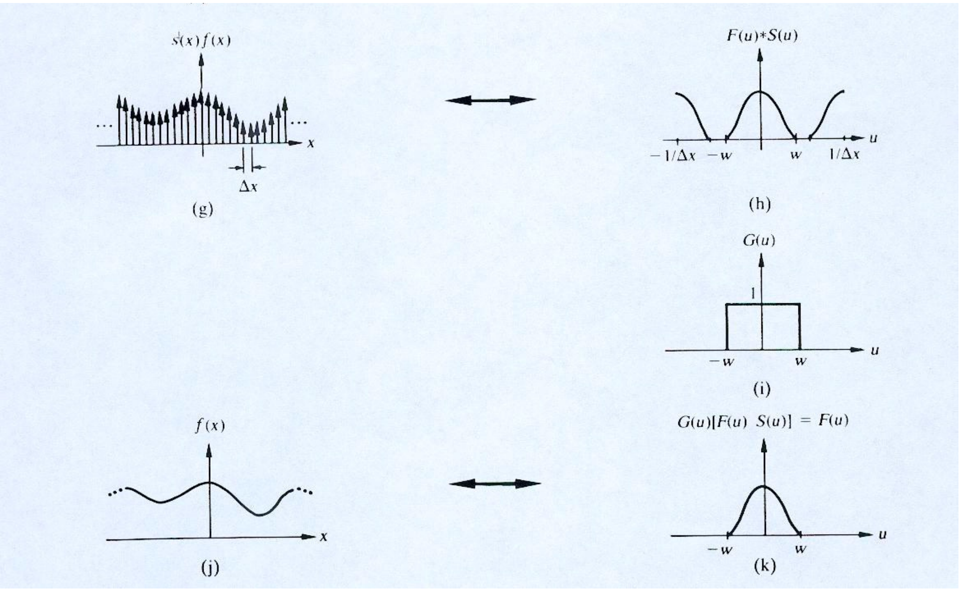
- Sampling is NOT a convolution, but rather a point-wise <u>multiplication</u> of the signal and a spike train (Dirac-functions).
- In the Fourier domain it becomes a <u>convolution</u>. The result is the frequency content replicated with copies spread by the sampling rate.



But here we get an overlap.

What happens during sampling?

- But if the sampling rate is high enough, the copies in the Fourier domain will be spread out and no longer overlap.
- To get the original signal back, we need to multiply with a ideal lowpass filter, which is a convolution with a sinc function in the spatial domain.

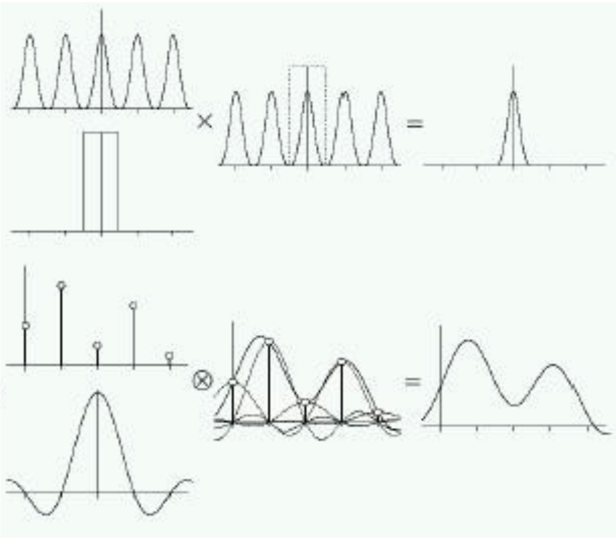


Reconstruction

- For reconstruction, we need to convolve with a sinc function.
  - It is the Fourier transform of the box function.
  - It has infinite support, which cannot be implemented.
- May be approximated by a Gaussian, cubic or even a “tent” function.

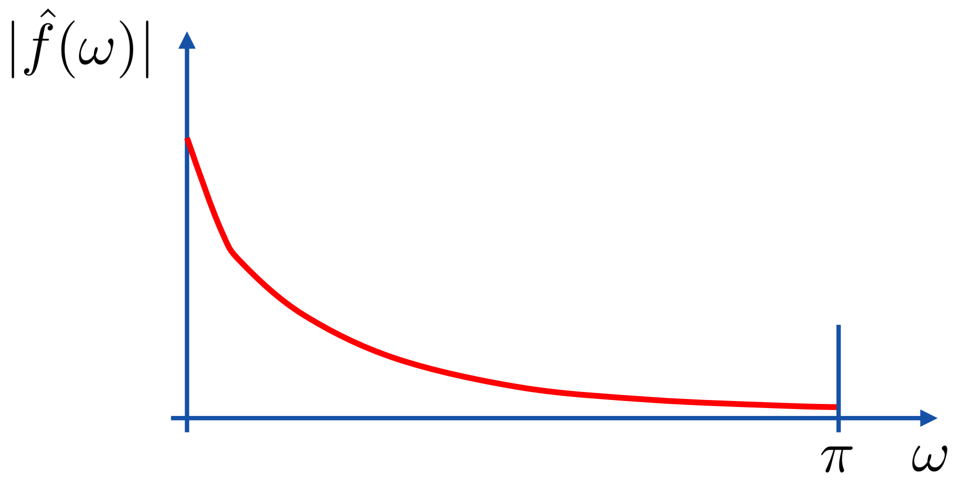
Frequency domain

Spatial domain



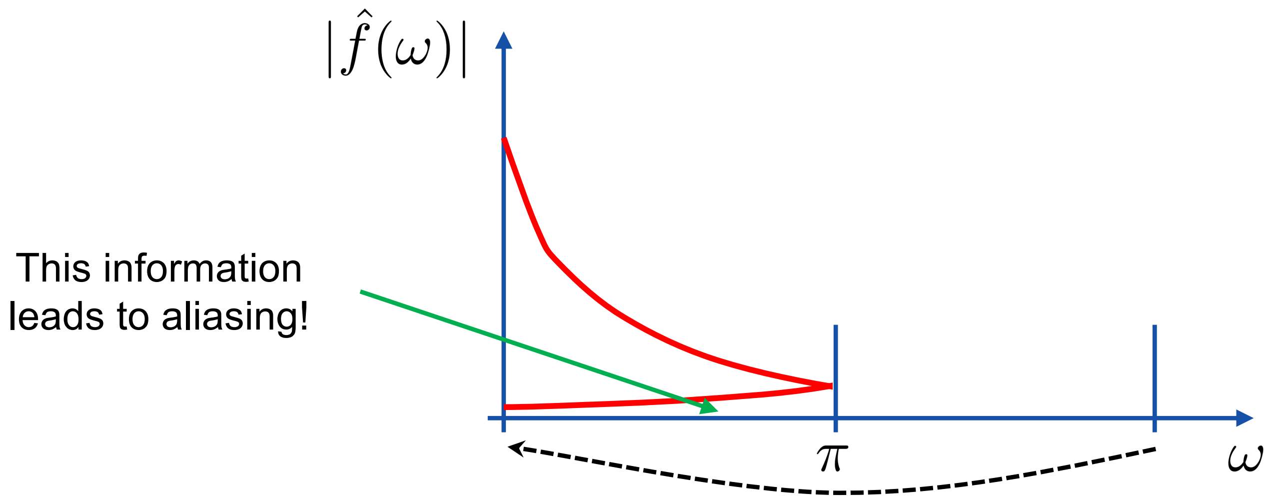
Another way of understanding aliasing

- Assume we have the Fourier transform of an image. The highest frequency is π, i.e. when you have two samples per period.



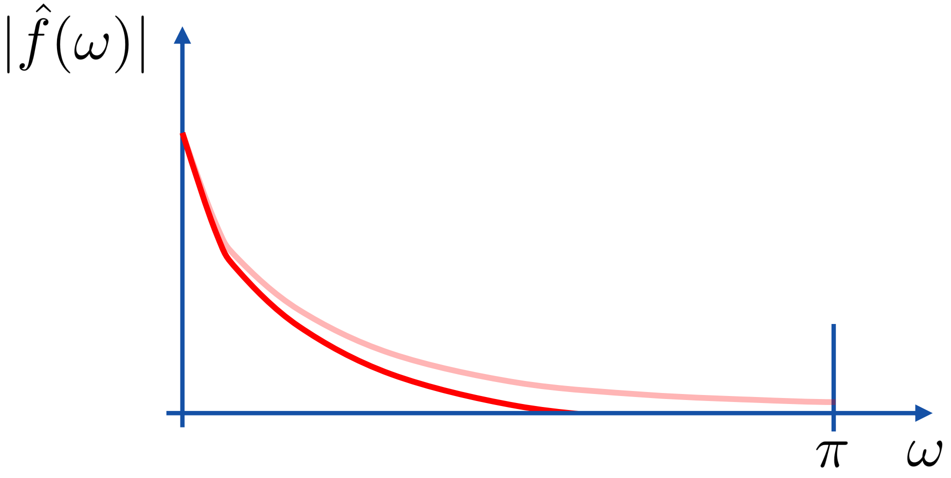
Another way of understanding aliasing

- If you reduce the sampling frequency by half, what is above π (with respect to the new sampling frequency) can no longer be represented.
- These frequencies will be folded at the new π, with the original highest frequency becoming frequency 0, i.e. a constant signal.

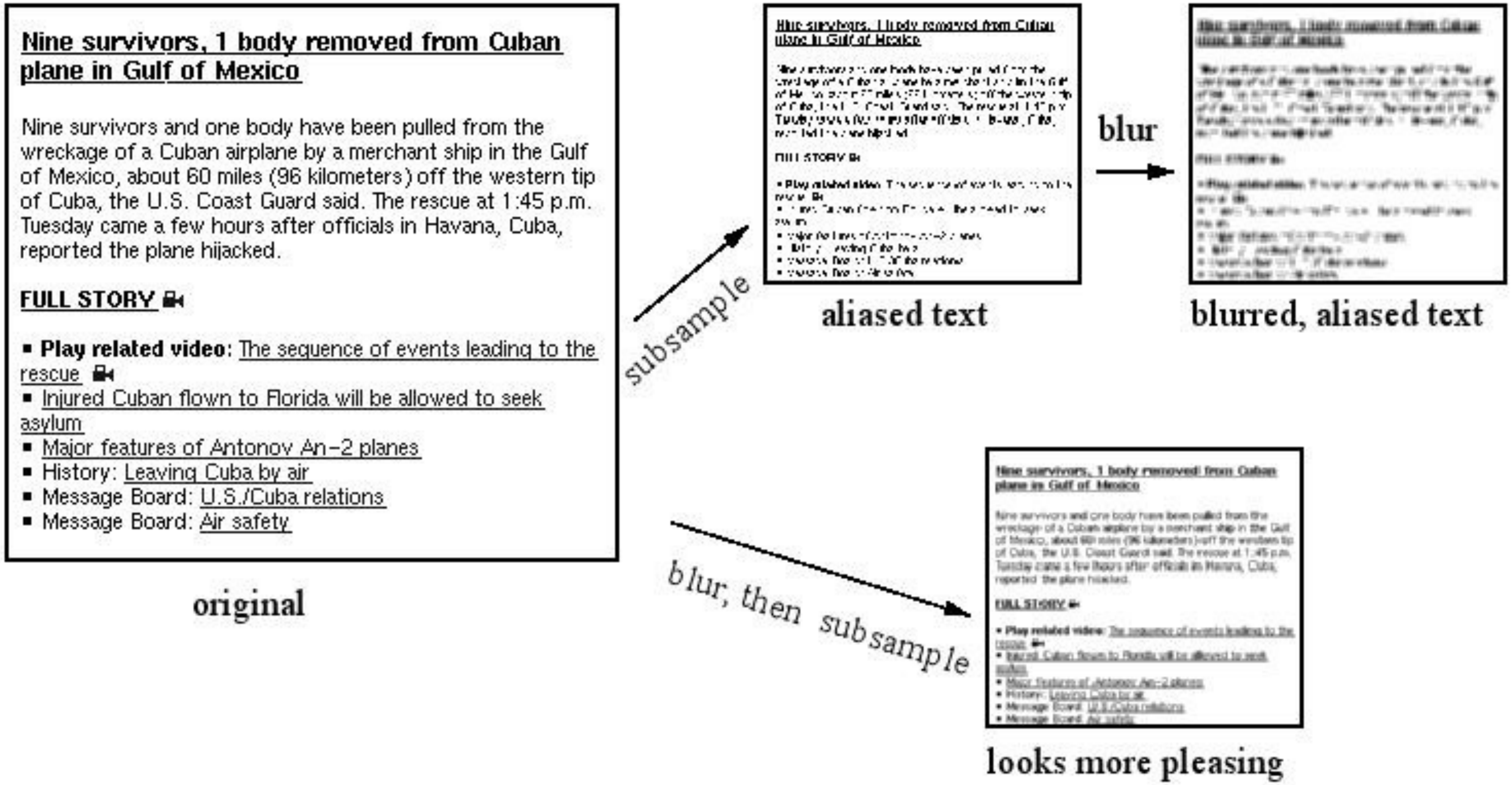


Another way of understanding aliasing

- The best way to avoid this problem is to first blur the image, then subsample.
- However, we need to find a good balance.
  - Too much blur and real content disappears.
  - Too little blur and we get aliasing.
- A suitable middle way is to blur with the filter $\begin{bmatrix} \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \end{bmatrix}$.



Anti-aliasing example



Matching image data

- What if you want to match two different image, what do you do?
- Problem: Trying to match two mega-pixel images on a per pixel level would lead to a combinatorial explosion. We need something less complex.

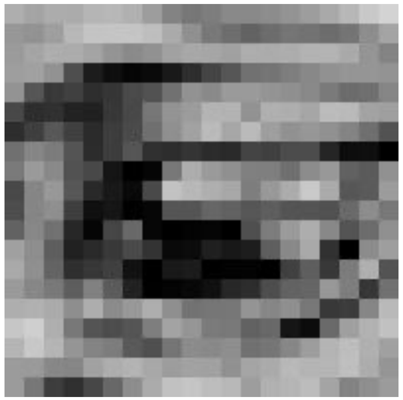
Feature extraction and matching

A common solution:

- Extract image features (corners, blobs, line, etc) that locally stand out.
- Represent these features with some easily matched descriptor.
- Match the features pair-wise between the images.
  - If enough features match, there is some similarity between the images.
  - From the feature matches we can compute differences between the images.

Feature detection

- How can we define features from image data, if the data looks like this?
- As human beings, we clearly see the eye of the horse.
- But to the computer it is just an array of numbers.



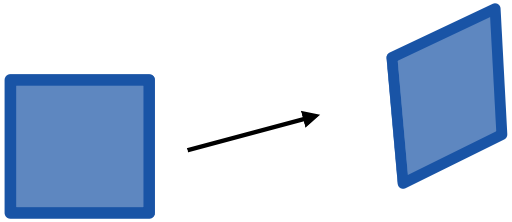
174	164	172	171	181	183	177	157	131	119	125	137	137	141	147	157	169	184	200	211
151	147	159	183	196	197	197	190	171	144	121	110	99	109	117	114	109	117	142	164
155	165	174	172	164	148	138	145	161	168	166	161	173	172	173	179	180	170	149	132
158	161	146	78	28	15	6	10	25	46	69	86	113	117	126	136	145	149	149	147
147	86	57	64	46	54	74	82	113	146	138	154	154	148	144	137	122	108	112	134
72	46	49	35	33	56	73	119	148	156	180	179	146	176	174	166	187	186	163	159
46	63	88	64	43	61	75	112	144	162	187	167	186	158	146	151	145	132	112	83
112	134	124	84	44	61	85	48	45	53	44	42	68	64	76	76	37	13	15	0
122	154	138	82	72	21	0	5	120	173	154	144	128	127	112	140	152	97	91	145
73	134	153	87	41	25	10	74	194	184	175	168	126	153	172	201	171	87	91	156
78	134	143	63	16	22	1	6	85	85	103	97	87	94	84	84	82	106	141	120
98	135	114	39	5	70	105	6	0	3	5	0	79	125	154	173	136	84	108	144
160	142	94	33	42	57	74	51	9	12	32	24	29	95	137	169	49	0	140	158
169	144	100	47	55	64	18	0	25	28	0	3	4	0	65	95	56	124	176	84
154	136	122	94	129	92	34	1	4	8	17	39	45	78	99	102	152	126	55	113
174	177	171	142	173	139	153	174	143	115	118	119	148	134	143	120	56	60	134	185
198	207	175	120	84	77	85	126	163	150	123	121	98	105	21	12	106	159	178	193
180	200	181	149	132	125	90	80	117	140	137	142	113	118	135	157	180	190	176	161
155	154	136	134	161	174	180	168	157	162	161	145	165	171	185	180	181	191	173	149
155	119	61	48	71	104	141	171	193	197	189	184	157	177	181	188	164	131	143	155

Inherent variabilities in image data

- Visual data vary substantially due to geometric transformations and lighting variations in the environment.
- Nevertheless the brain maintains a stable perception of the environment.

Image transformations

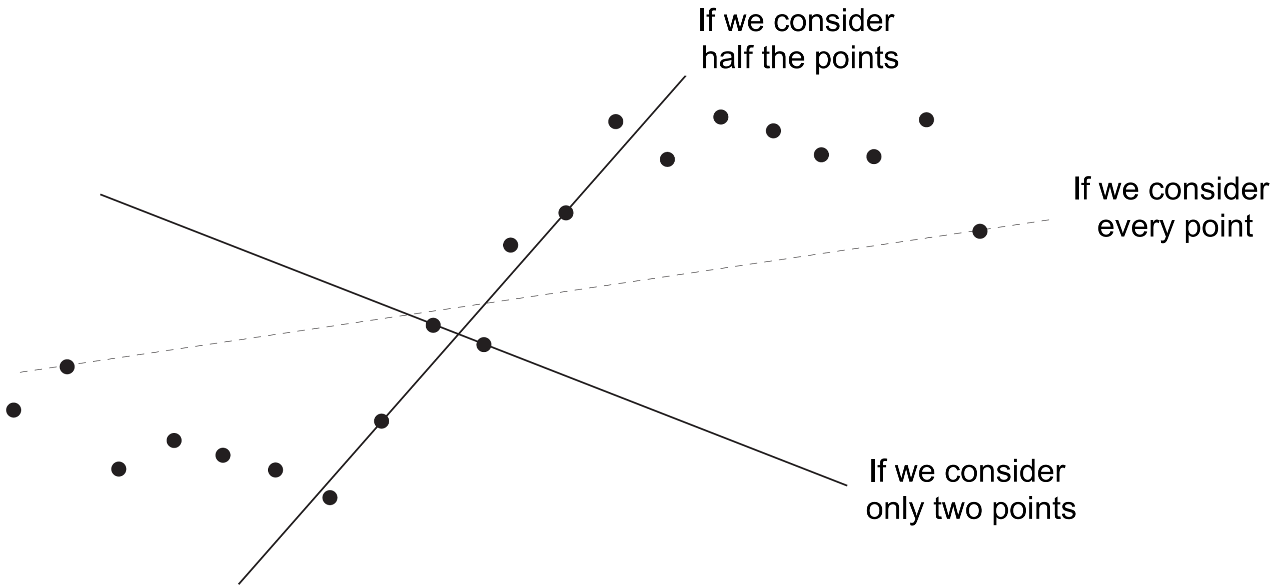
- Images may undergo large non-linear changes that can be locally linearized around each feature.
  - <u>scaling transformations</u> caused by objects of *different sizes* and at *different distances* to the observer
  - <u>affine transformations</u> modelling *image deformations* caused by variations in the viewing direction
- These changes to the features need to be taken into account, when trying to match features between the images.



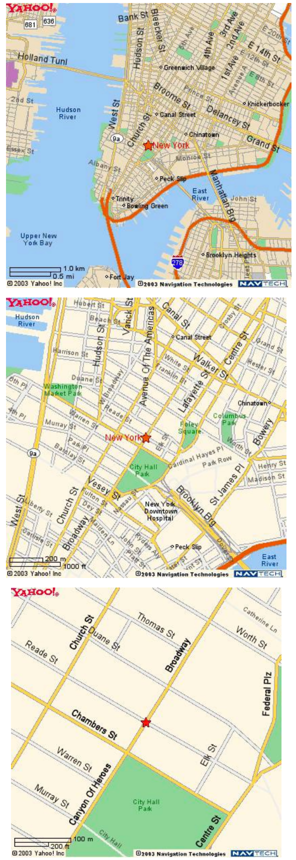
$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} \simeq \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Dependency on scale of observation

- Even an as “simple” problem as detecting the edges of an object may be strongly dependent on the scale of the image operators.
- Depending on the scale you are looking at, results may vary considerably.



Multi-scale descriptions



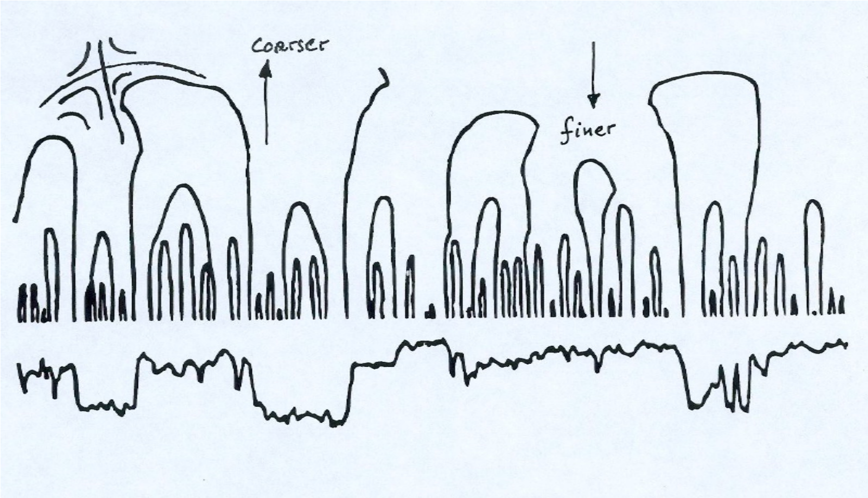
- Also in the real world, we see different things at different scales.
- Physics:
  - Different types of descriptions depending on the scale of analysis

    quantum mechanics ⟹ particle physics ⟹ thermodynamics ⟹ solid mechanics ⟹ astronomy ⟹ relativity theory
- Cartography
  - Maps with different degrees of abstraction depending on scale

    building ⟹ city ⟹ county ⟹ country ⟹ world
- Computer vision
  - Dynamically varying scale levels must be handled automatically

Scale-space representation

- Scale-space theory: a well-founded theory for multi-scale image structures
- Continuum of scale levels: $f(x) \mapsto L(x; s)$ with $L(x; 0) = f(x)$
- The transformation from a fine scale to a coarser scale must not introduce new structures that were not present in the original data.



Points where slope is locally maximum or minimum.

Figure adapted from Witkin (1983) “Scale-space filtering”, Proc. Int. Joint Conf. on Art. Intell., 1019–1022.

Scale-space symmetry properties

Find an operator $\mathcal{T}_s$ that can be used to create $L(x; s)$ from $f(x)$,

$$\mathcal{T}_s f(x) = L(x; s)$$

What properties do we want this operator to have?

- *Linearity*:

$$\mathcal{T}_s(af_1(x) + bf_2(x)) = a\mathcal{T}_s f_1(x) + b\mathcal{T}_s f_2(x)$$

- *Shift-invariance:*

$$\mathcal{T}_s\left(S_{\Delta x} f(x)\right) = S_{\Delta x}(\mathcal{T}_s f(x))$$

where $S_{\Delta x}$ is a shift operator, that is $S_{\Delta x} f(x) = f(x - \Delta\text{x})$.

Note: if these two properties are satisfied, $\mathcal{T}_s$ can be represented as a filter $g(x; s)$ that can be applied to the image through a convolution, $L(x; s) = g(x; s) * f(x)$.

Scale-space symmetry properties

- *Semi-group* structure over scale s

$$\mathcal{T}_{s_1}\mathcal{T}_{s_2}f(x) = \mathcal{T}_{s_1+s_2}f(x)$$

If we apply the operator twice with two different scales, the result is the same as if applied the operator once with the sum of the two scales.

- *Scale covariance* under scaling transformations $x' = Sx$

$$\mathcal{T}_s f(x) = \mathcal{T}_{s'} f'(x')$$

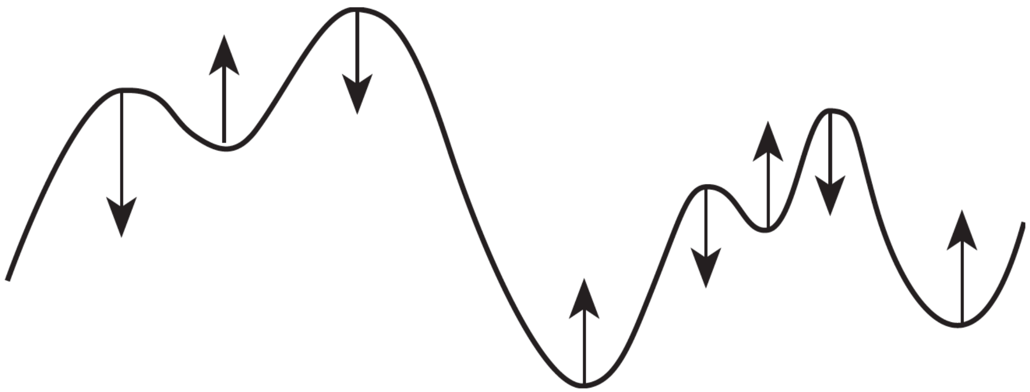
If we rescale $f(x)$ and get the image $f'(x')$, then for each scale s in $L(x; s)$ there is a matching scale s' in the new representation $L'(x'; s')$, so that $L'(x'; s') = L(x; s)$.

Scale-space symmetry properties

- *Non-enhancement of local extrema*: If at some scale s, a point x is a local maximum (minimum) for $L(x; s)$, then

$$\delta_s L(x; s) \leq 0 \text{ for any spatial maximum}$$
$$\delta_s L(x; s) \geq 0 \text{ for any spatial minimum}$$

This means that the extremum will always decay as we increase the scale s.



Necessity result

- If we require all these symmetry properties to be fulfilled, then the scale-space representation over a 2D spatial domain must satisfy

$$\delta_s L(x;s) = \frac{1}{2} \nabla_x^T \left(\Sigma_0 \nabla_x L(x;s) \right) - \delta_0^T \nabla_x L(x;s)$$

for some 2×2 covariance matrix Σ_0 and some 2D vector δ_0 with $\nabla_x = \left(\delta_{x_1}, \delta_{x_2}\right)^T$.

- If we further expect *rotational symmetry*, then

$$L(x;s) = g(x;s) * f(x),$$

where

$$g(x;s) = \frac{1}{2\pi s} e^{-x^T x/2s}$$

This corresponds to Gaussian blurring where scale s corresponds to the variance.

Proof in Lindeberg (2011) "Generalized Gaussian scale-space axiomatics comprising linear scale-space, affine scale-space and spatio-temporal scale-space”, Journal of Mathematical Imaging and Vision, 40(1): 36–81.

Gaussian scale-space representation

Original

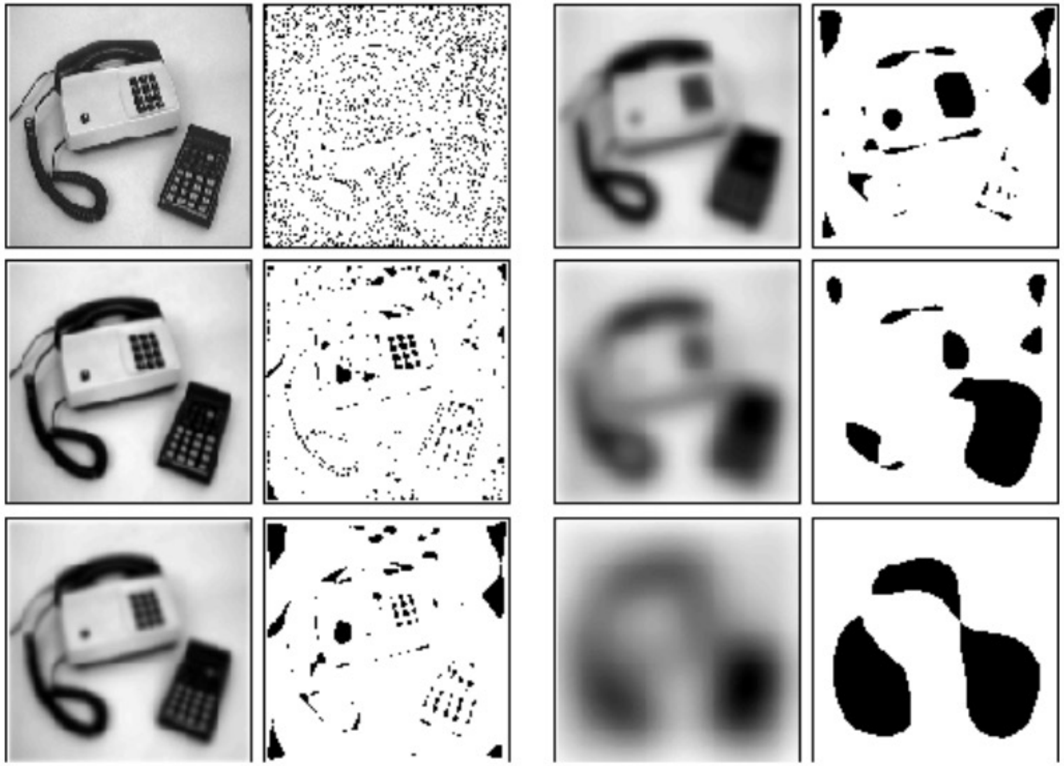
$s = 1$

$s = 8$

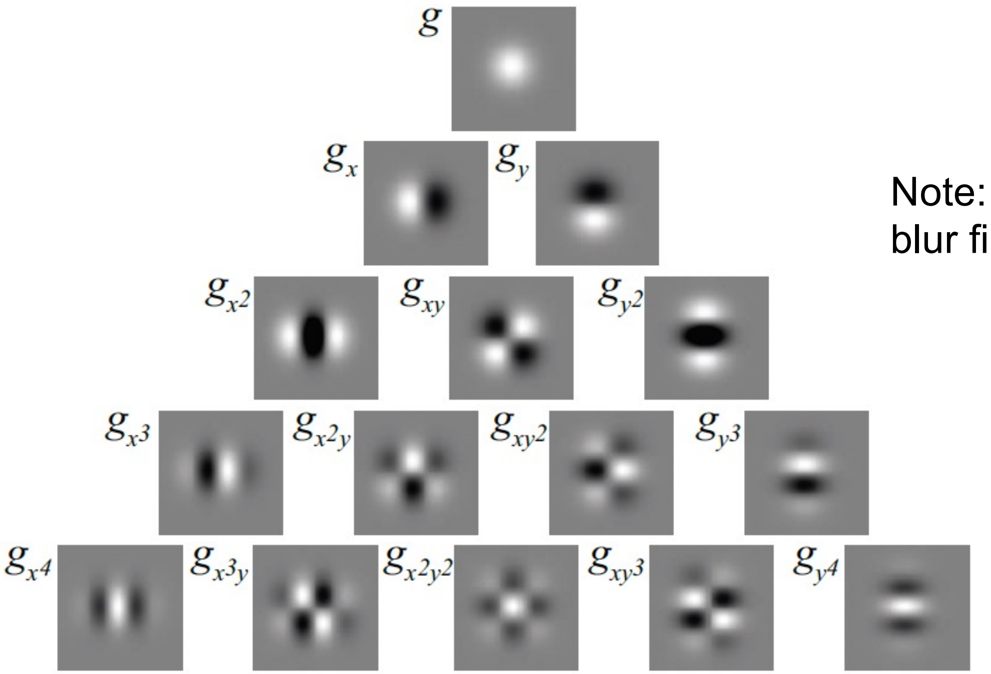
$s = 64$

Different image structures at different scales

Dark grey-level blobs (local minima with spatial extent) at multiple scales.



Gaussian derivative kernels

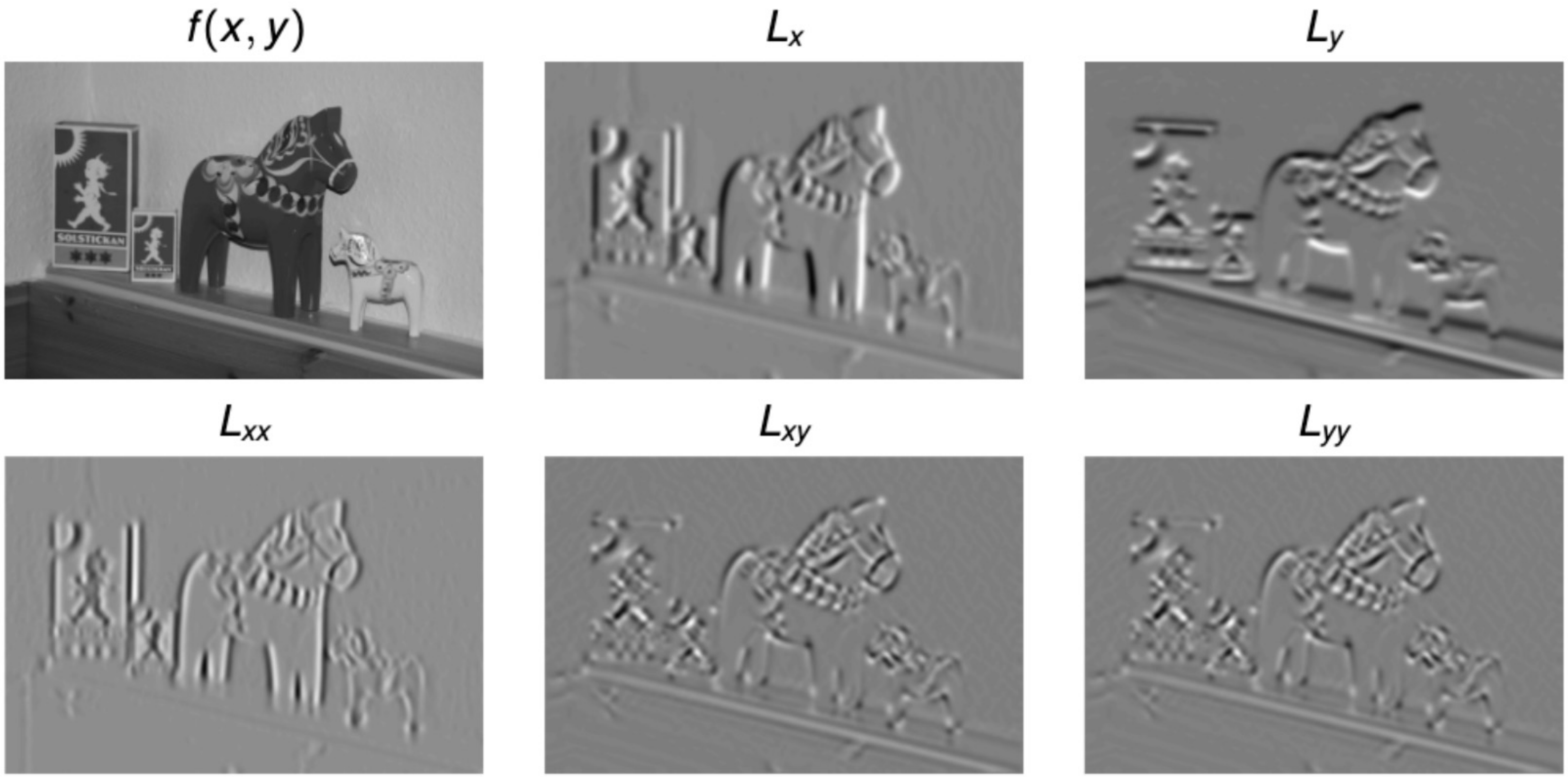


Note: since both operations are linear, we may either blur first and then compute derivates, or vice versa.

$$L_x(x;s) = g_x(x;s) * f(x) = g(x) * f_x(x;s)$$

These can be used as a general basis for expressing image operations such as feature detection, feature classification, surface shape, image matching and recognition.

Results of Gaussian derivative operators

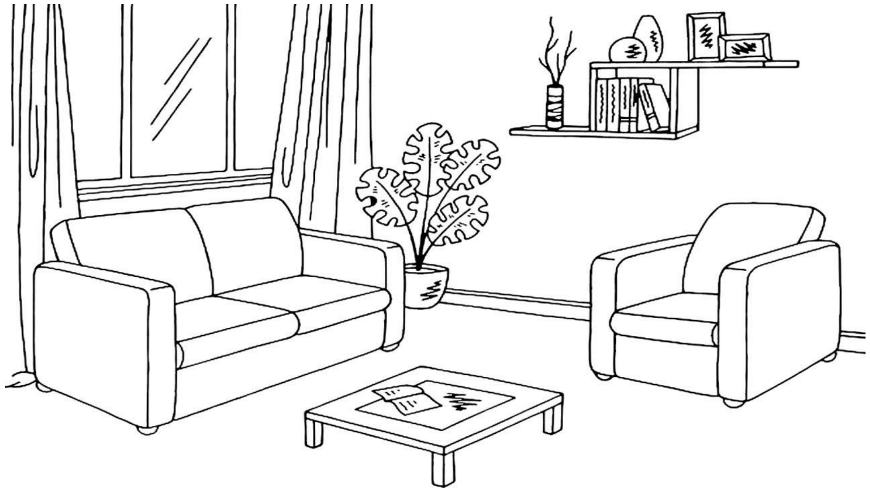


The importance of discontinuities

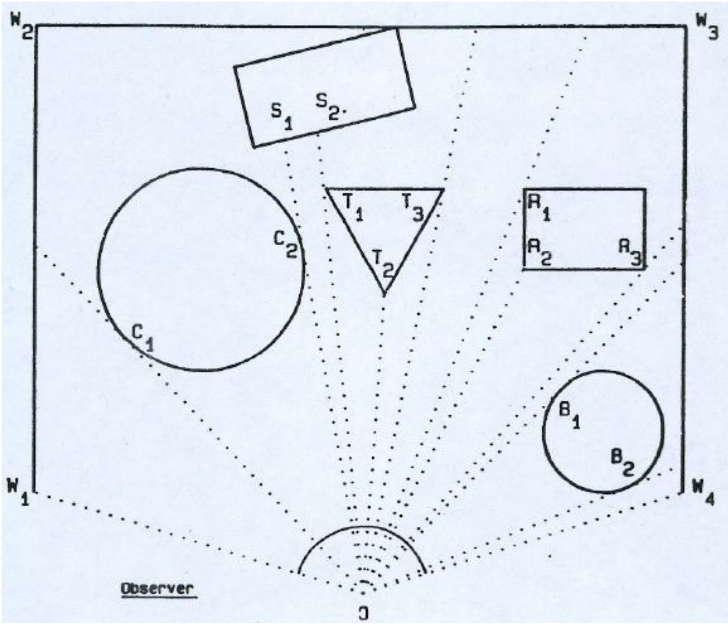
Under rather general assumptions about the image formation process:

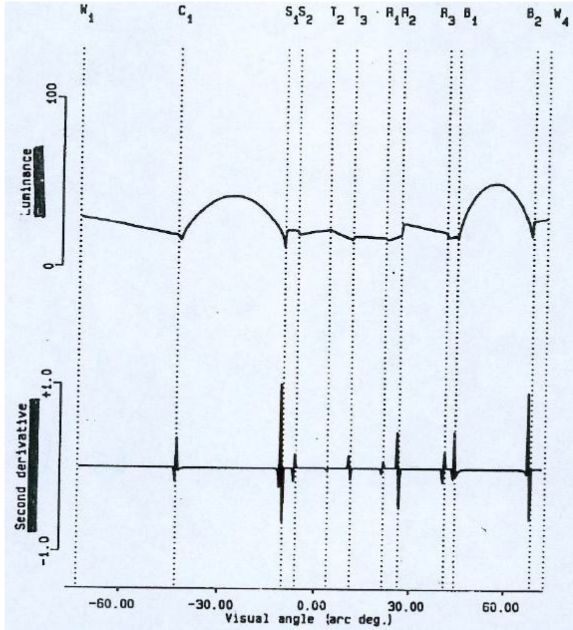
- the world consists of smooth surfaces with different reflectance properties
- where a discontinuity in image brightness corresponds to a discontinuity in:
  - depth
  - surface orientation
  - reflectance, or
  - Illumination

To understand the world it is valuable to detect discontinuities in image brightness (edges) and try to explain them.



Edge detection in idealized noise free situation

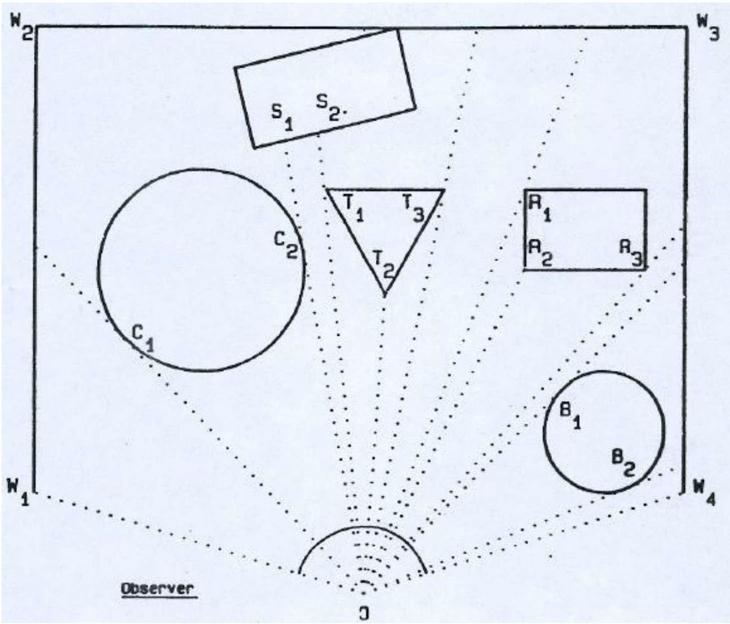


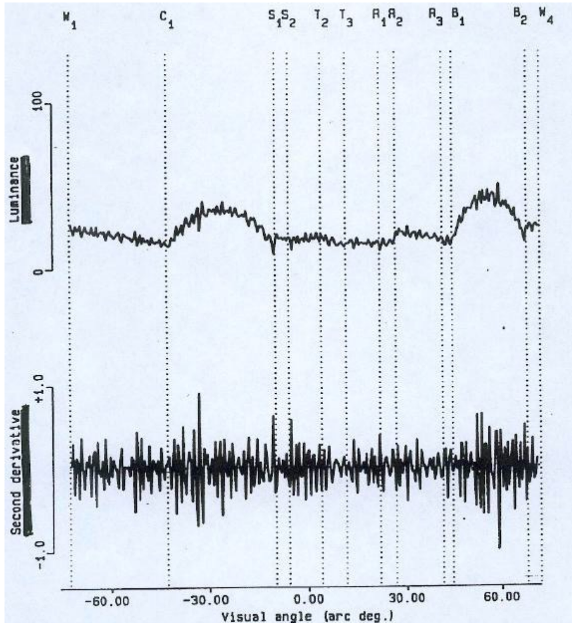


Noisy luminance and second order derivative

Figures from Watt (1988) Visual Processing: Computational, Psychophysical and Cognitive Research, Lawrence Erlbaum.

Edge detection in the presence of noise



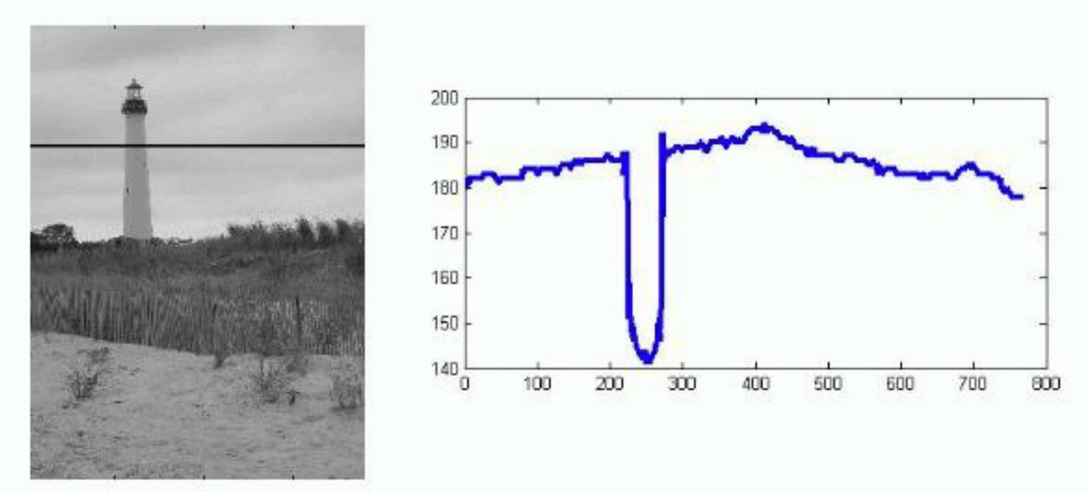


Noisy luminance and second order derivative

Figures from Watt (1988) Visual Processing: Computational, Psychophysical and Cognitive Research, Lawrence Erlbaum.

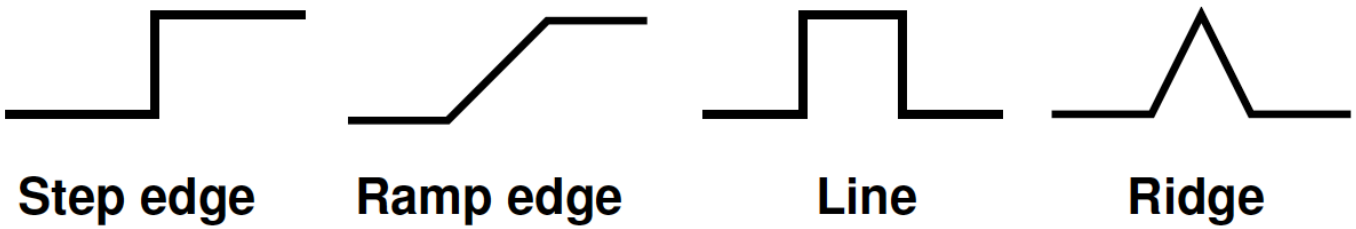
Why edges?

- Edge features constitute important features to humans
- Independent of illumination
- Easy to detect computationally
- Used to form higher level features (lines, curves, corners, etc)
- Natural primitives in CAD-like models of man-made objects



How do edges look in practice?

Idealized models:



In practice, edges are blurry and noisy:

Problem: Notion of discontinuity does not exist for discrete data!

Fundamental problem

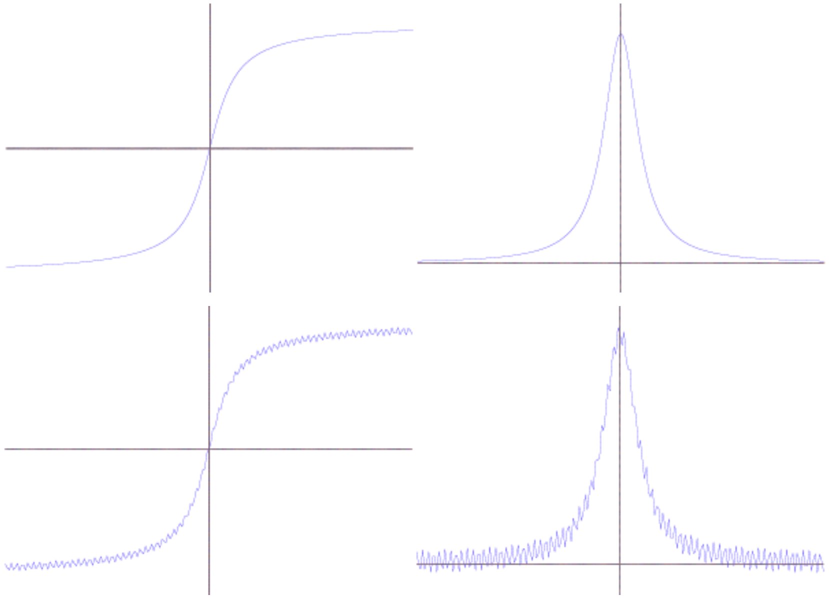
- Differentiation is ill posed - an arbitrary small perturbation in the input can lead to arbitrarily large perturbation in the output:

Ex: $f(x) = \arctan(x)$

$$f'(x) = \frac{1}{1+x^2}$$

$$f(x) = \arctan(x) + \varepsilon \sin(\omega x)$$

$$f'(x) = \frac{1}{1+x^2} + \varepsilon\omega \cos(\omega x)$$



- The difference $\varepsilon\omega \cos(\omega x)$ can be arbitrarily large if $\omega \gg \frac{1}{\varepsilon}$.

Noise reduction: Smoothing

Basic idea: Precede differentiation by smoothing.

$$L_x(x,s) = \delta_x(g(x;s) * f(x))$$

Trade-off problem:

- Increasing amount of smoothing (typically $s > 16$):
  - stronger suppression of noise,
  - higher distortions of "true" structures
- Decreasing amount of smoothing (typically $s < 1$):
  - more accurate feature detection,
  - higher number of "false positives".

Laplacian “edge detection”

Motivations:

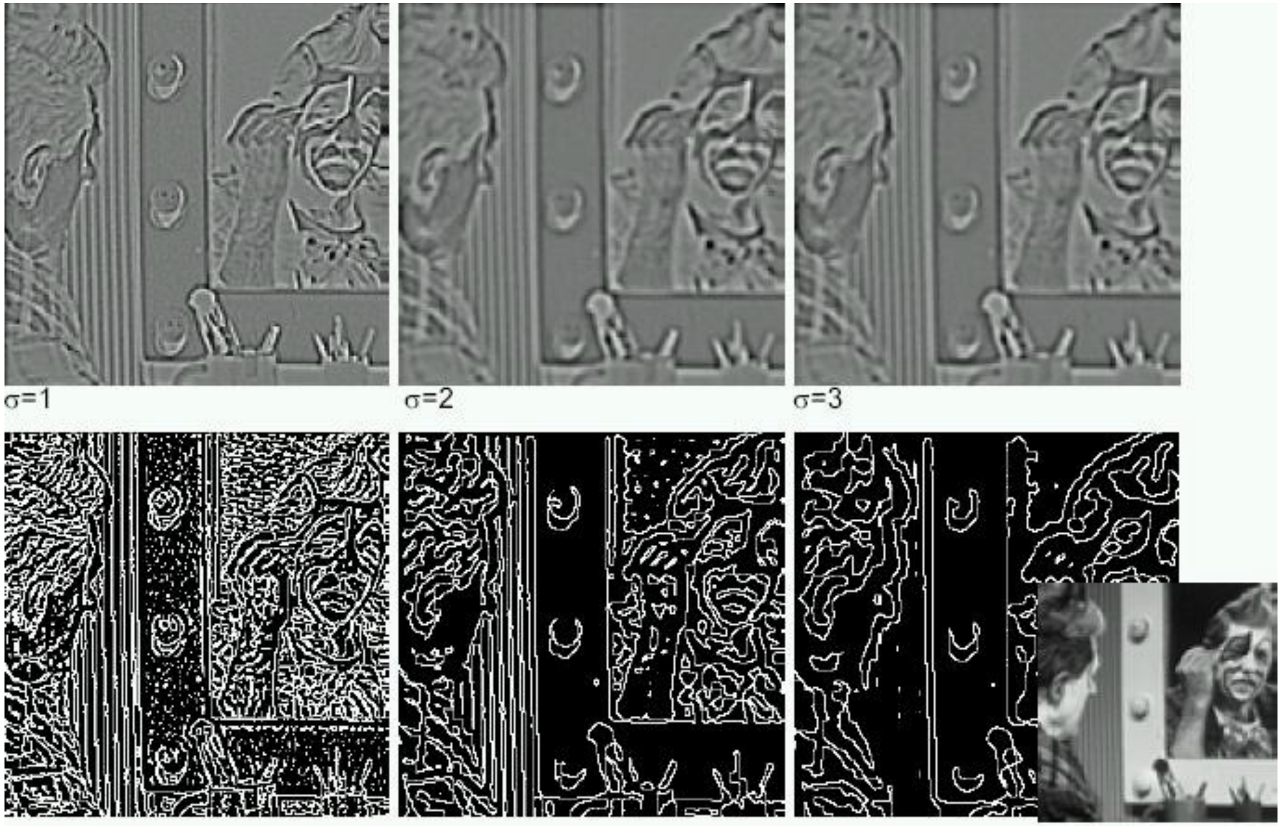
- For 1-D signals, edges correspond to peaks in the first order derivative and to zero-crossings in the second order derivative.
- For 2-D signals, the Laplacian operator $\nabla^2 L = L_{xx} + L_{yy}$ is a rotationally symmetric operator that coincides with the second-order derivative along one-dimensional straight lines

$\Longrightarrow$ Early attempt to detect edges by zero-crossings of the Laplacian, $\nabla^2 L = 0$, proposed by Marr and Hildreth in 1980.

Major problems:

- Zero-crossings of the Laplacian also respond to “false edges”.
- Poor localization for curved edges.

Zero-crossings of the Laplacian



Gradient based edge detection

- Convolve image by appropriate $g(x; s)$ prior to derivative computations.
- Compute gradient vector $\nabla L = \left(L_x, L_y\right)^T$
- Measure edge strength by gradient magnitude $|\nabla L| = \sqrt{L_x^2 + L_y^2}$

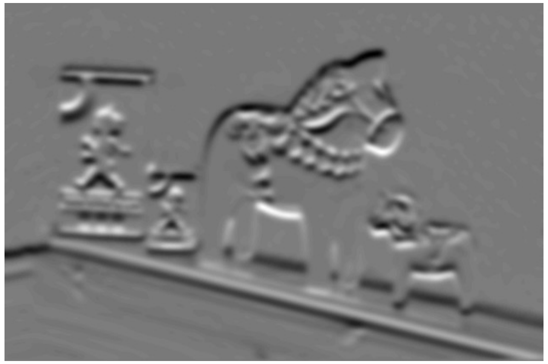
Gradient estimation

- Partial derivatives estimated by difference operators:

$$L_x(x,y) \approx \frac{L(x+h,y)-L(x-h,y)}{2h}, \qquad \text{Filter mask: } \begin{pmatrix} 0 & 0 & 0 \\ -1/2 & 0 & 1/2 \\ 0 & 0 & 0 \end{pmatrix} \text{ for } h = 1$$

$$L_y(x,y) \approx \frac{L(x,y+h)-L(x,y-h)}{2h}, \qquad \text{Filter mask: } \begin{pmatrix} 0 & 1/2 & 0 \\ 0 & 0 & 0 \\ 0 & -1/2 & 0 \end{pmatrix} \text{ for } h = 1$$

- Gradient direction: $\theta = arctan\left(\frac{L_y}{L_x}\right) + n\pi = atan2(L_x, L_y)$



Accuracy of derivative approximations

- Taylor expansions can be used to estimate the accuracy:

$$L(x+h) = L(x) + hL'(x) + \frac{1}{2}h^2 L''(x) + \cdots$$

$$L(x-h) = L(x) - hL'(x) + \frac{1}{2}h^2 L''(x) + \cdots$$

  By subtracting the second equation from the first we obtain

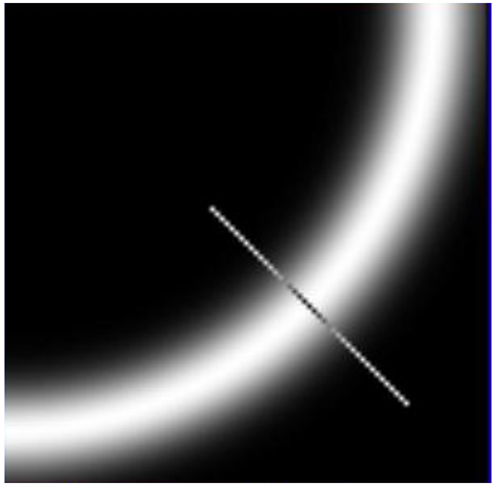
$$\frac{L(x+h) - L(x-h)}{2h} = L'(x) + \mathcal{O}(h^3)$$

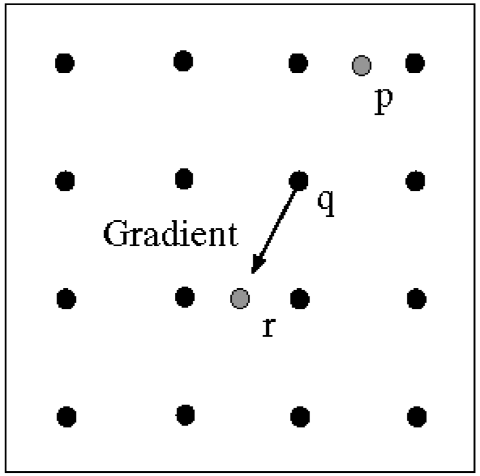
- Higher order derivative approximations can constructed and analysed in a corresponding manner.
- If we use larger kernels, the h^3-term can also be eliminated.

Canny edge detection (1986)

Typical problem: If you try to perform edge detection by thresholding on the edge strength, then the resulting edges may be several pixels wide. Solution:

1. Convolve image by smoothing kernel $g(x; s)$.
2. Estimate gradient magnitude $|\nabla L|$ and gradient direction.
3. Threshold on gradient magnitude and keep only points that are local extrema in the gradient direction (non-maximum suppression implemented by local search).



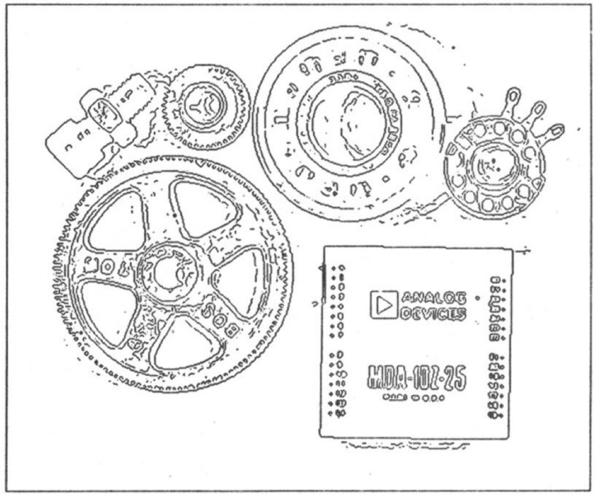


Hysteresis thresholding

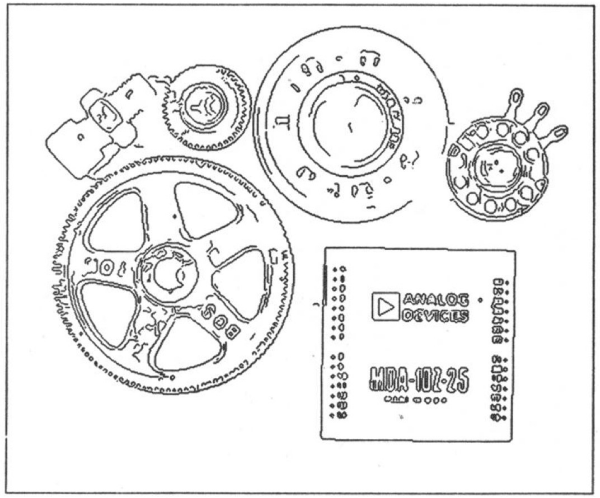
A scheme to remove short insignificant edges:

1. Remove all edge points with magnitude lower than a low threshold T_{low}.
2. Create chains of connected edge points.
3. If at least one point in a chain has magnitude higher than a high threshold T_{high}, keep that chain. Otherwise remove it.

Before:



After:



Differential edge detection (Lindeberg 1993)

- Non-maximum suppression: An edge point is a point where the gradient magnitude assumes a maximum in the gradient direction.
- This can be expressed in terms of:
  - Gradient $\nabla \mathrm{L} = \left(L_x, L_y\right)^T$ and gradient magnitude $|\nabla L| = \sqrt{L_x^2 + L_y^2}$
  - The normalized gradient direction: $e_v = \frac{\nabla \mathrm{L}}{|\nabla \mathrm{L}|} = \frac{\left(L_x, L_y\right)^T}{\sqrt{L_x^2 + L_y^2}}$
  - Directional derivative in any direction α:
    $$\delta_\alpha = \cos\alpha\ \delta_x + \sin\alpha\ \delta_y$$
  - Directional derivative in gradient direction:
    $$\delta_v = \frac{L_x}{\sqrt{L_x^2 + L_y^2}} \delta_x + \frac{L_y}{\sqrt{L_x^2 + L_y^2}} \delta_y \text{ with } L_v = \delta_v L = \sqrt{L_x^2 + L_y^2} = |\nabla L|$$

Differential geometric edge definition

Requirements for gradient magnitude to be maximal in gradient direction:

$$\begin{cases} \delta_v(L_v) = 0 \\ \delta_{vv}(L_v) < 0 \end{cases} \quad \text{or} \quad \begin{cases} L_{vv} = 0 \\ L_{vvv} < 0 \end{cases}$$

In terms of coordinates:

$$L_{vv} = (\cos\alpha\ \delta_x + \sin\alpha\ \delta_y)^2 L = \cos^2\alpha L_{xx} + 2\cos\alpha\sin\alpha L_{xy} + \sin^2\alpha L_{yy} =$$

$$= \frac{L_x^2}{L_x^2+L_y^2} L_{xx} + \frac{2L_xL_y}{L_x^2+L_y^2} L_{xy} + \frac{L_y^2}{L_x^2+L_y^2} L_{yy} = \frac{L_x^2L_{xx}+2L_xL_yL_{xy}+L_y^2L_{yy}}{L_x^2+L_y^2} = 0$$

Since the denominator is irrelevant, the edges are given by:

$$\widetilde{L_{vv}} = L_x^2L_{xx} + 2L_xL_yL_{xy} + L_y^2L_{yy} = 0$$

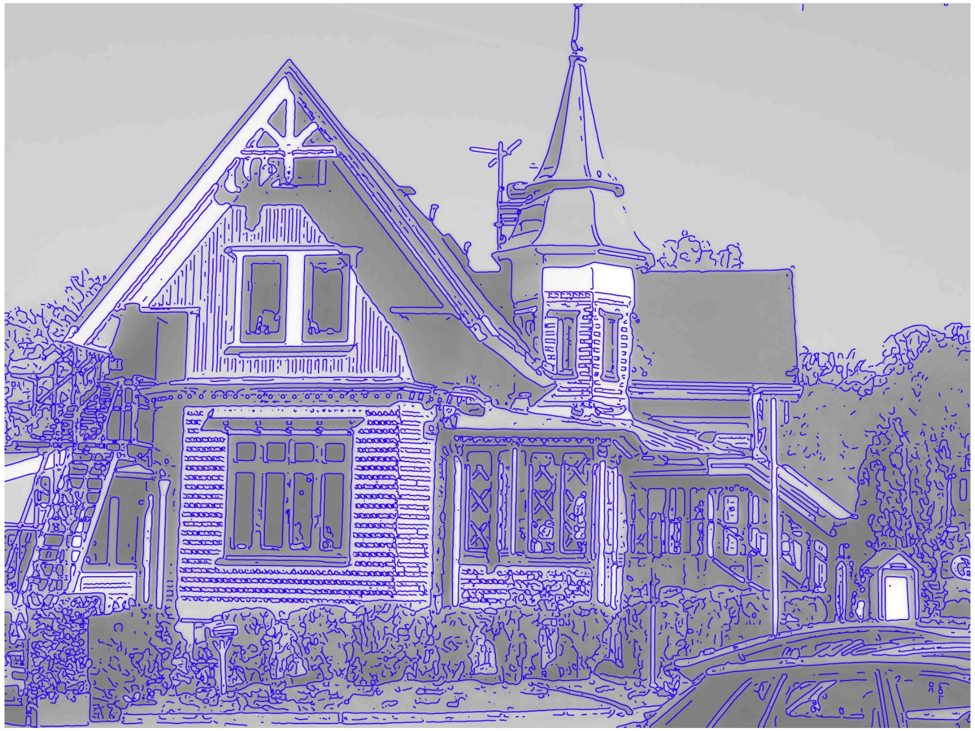
$$\widetilde{L_{vvv}} = L_x^3L_{xxx} + 3L_x^2L_yL_{xxy} + 3L_xL_y^2L_{xyy} + L_y^3L_{yyy} < 0$$

Differential edge detection

Differential edge detection in practice:

1. Convolve image by a Gaussian kernel.
2. Compute partial derivatives up to order three and combine these into the differential invariants $\widetilde{L_{vv}}$ and $\widetilde{L_{vvv}}$ at every image point.
3. Search for the zero-crossings of $\widetilde{L_{vv}}$ that satisfy $\widetilde{L_{vvv}} < 0$.
   - Gives subpixel accuracy and connected edge segments automatically.
   - Avoids issues of orientation estimation and handling as well as edge tracking in discrete non-maximum suppression.
4. Can be combined with either a single threshold on the gradient magnitude, $L_v > t$, or hysteresis thresholding using two thresholds.

Differential edge detection



Summary of good questions

- Why do we get image aliasing when subsampling and what to do about it?
- Why is the notion of scale important in image analysis and computer vision?
- What is a scale-space representation? On what basis is it constructed?
- What structural requirements are natural to impose on early visual operations?
- What is meant by a Gaussian derivative? Why are these important?
- Why is edge detection important for image understanding?
- What families of methods exist for edge detection?
- What information do image gradients provide?
- How does the Canny edge detector work?
- What is differential edge detection?
- What should the image derivatives be equal to on edge points?

Recommended reading

- Gonzalez & Woods: Chapters 4.3, 10.2
- Szeliski: Chapters 3.5.1-3.5.3, 7.2