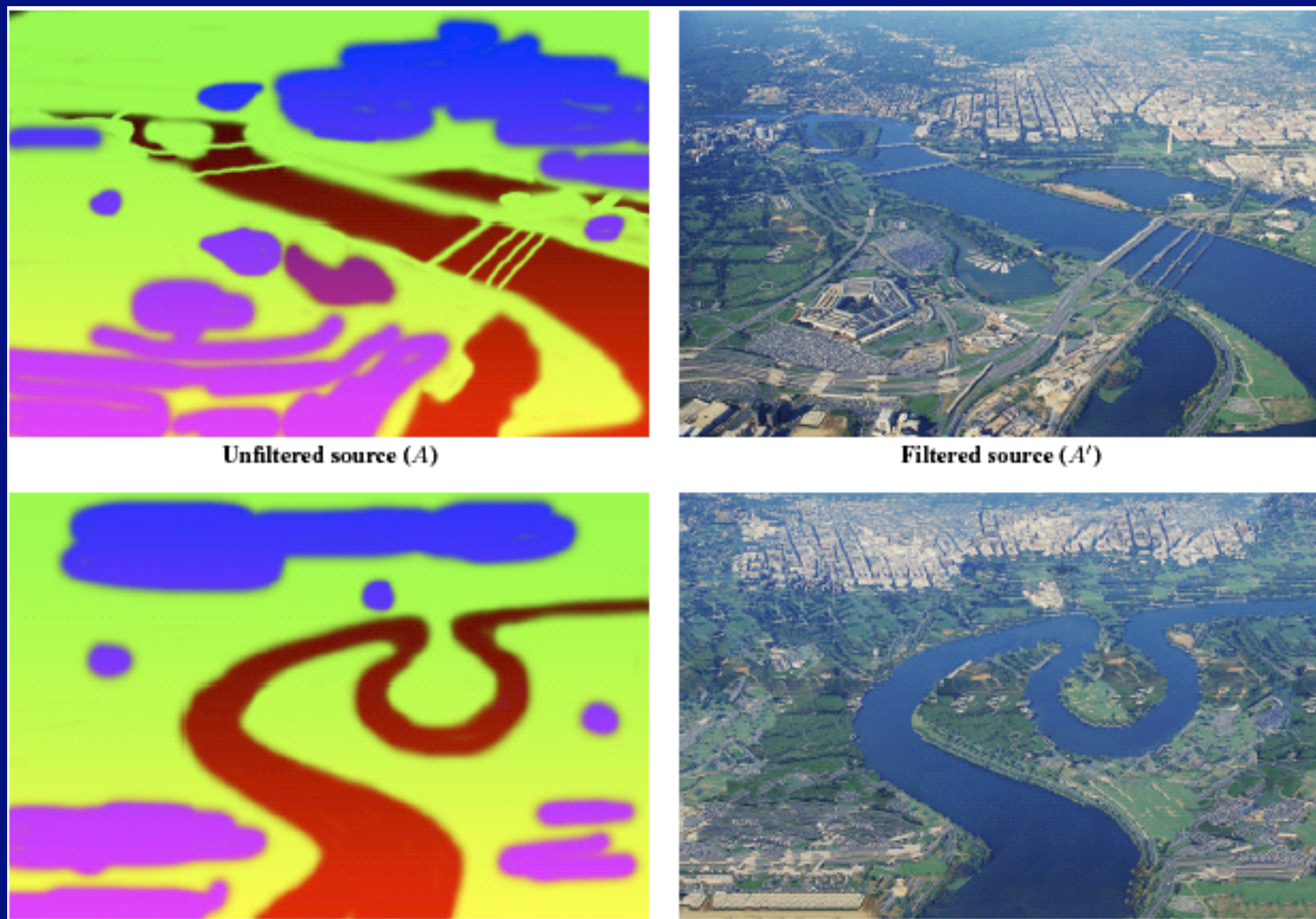


From “Image analogies”, Herzmann et al, SIGGRAPH 2001

Entropy

- Given a discrete probability distribution $p(x) = \mathrm{prob}(X = x)$

- Its ENTROPY is

$$H(p) = H(X) = -\sum_{x \in X} p(x) log_2 p(x)$$

- Right way to think of entropy:
  - the number of bits required, on average, to communicate the identity of X
  - eg die

Joint entropy

- Consider a pair of random variables, X, Y with p(x, y)
- Joint entropy is:

$$H(X,Y) = -\sum_{x \in X, y \in Y} p(x,y) \log_2 p(x,y)$$

- Number of bits required, on average, to give identities of X and of Y

Conditional Entropy

- How many bits, on average, you need to supply to specify Y given X is known

$$
\begin{aligned}
H(Y|X) &= \sum_{x \in X} p(x) H(Y|X=x) \\
&= \sum_{x \in X} p(x) \left[-\sum_{y \in Y} p(y|x) \log_2 p(y|x) \right. \\
&= -\sum_{x \in X, y \in Y} p(x,y) \log_2 p(y|x)
\end{aligned}
$$

KL divergence

- We would like to compare two probability distributions
  - perhaps model and reality?
  - model1 and model2
  - etc
- use Kullback-Leibler divergence

$$
\begin{aligned}
D(p \parallel q) &= \sum_{x \in X} p(x) \log_2 \frac{p(x)}{q(x)} \\
&= E_p(\log_2 \frac{p(x)}{q(x)})
\end{aligned}
$$

- always non-negative, 0 iff p=q, not a metric

KL divergence

- average number of bits that are wasted by encoding events from a distribution p using a code based on q

Evaluating string models

- Assume we have a N iid samples x_i from a process with pdf p, which is unknown
- We have models with pdf q_j
- We would like to compare models
- Idea: compute D(p||q)

- But we don't know p?

$$\frac{1}{N}\sum_i(-\log_2(q(x_i))) \to \sum p(x)\log_2(\frac{1}{q(x)}) = E_p(\log_2(\frac{p(x)}{q(x)})) - E_p(\log_2(p(x)))$$
$$= D(p \parallel q) + H(X)$$

Evaluating string models

- i.e. ranking models in order of average negative log-likelihood ranks them in order of D(p||q)
  - we don’t know H(x)
  - but if we use a really really good model, then negative log-likelihood could be quite close to H(x)

String models of English

- Recall we're working with letters

- uniform pdf on letters 4.76
- first order 4.03
- second order 2.8
- people guessing 1.3 (1.34)

Collocation

- Characterized by:
  - limited compositionality
    - meaning is not a straightforward composition
      - kicked the bucket -> kicked the cat
      - hear it through the grapevine -> hear it through the air (speakers?)
  - non-substitutability
    - cannot substitute even if words are appropriate
      - white wine -> yellow wine
  - non-modifiability
    - generally, can't apply grammatical transformations, additional material
      - bacon and eggs-> bacon and fried eggs
      - kick the bucket -> kick the red plastic bucket

Collocation: range

- Examples
  - she knocked on his door
  - they knocked at his door
  - 100 tourists knocked on Donaldson's door
  - a man knocked on the metal front door
- Notice "knock" ... "door"
  - (rather than "hit", "beat", "rap", etc.)
- We want methods to find pairings like this
  - non-accidental
  - over some range
  - note possible vision applications

Strategy

- Find pairs
  - with possible inserts
  - that occur with high frequency
  - where there is little support for the hypothesis that the pair is accidental
- Technology:
  - Hypothesis testing

Approach 1

- Count mean and variance of separation between words in a k-word window
- Low variance suggests collocation/pattern

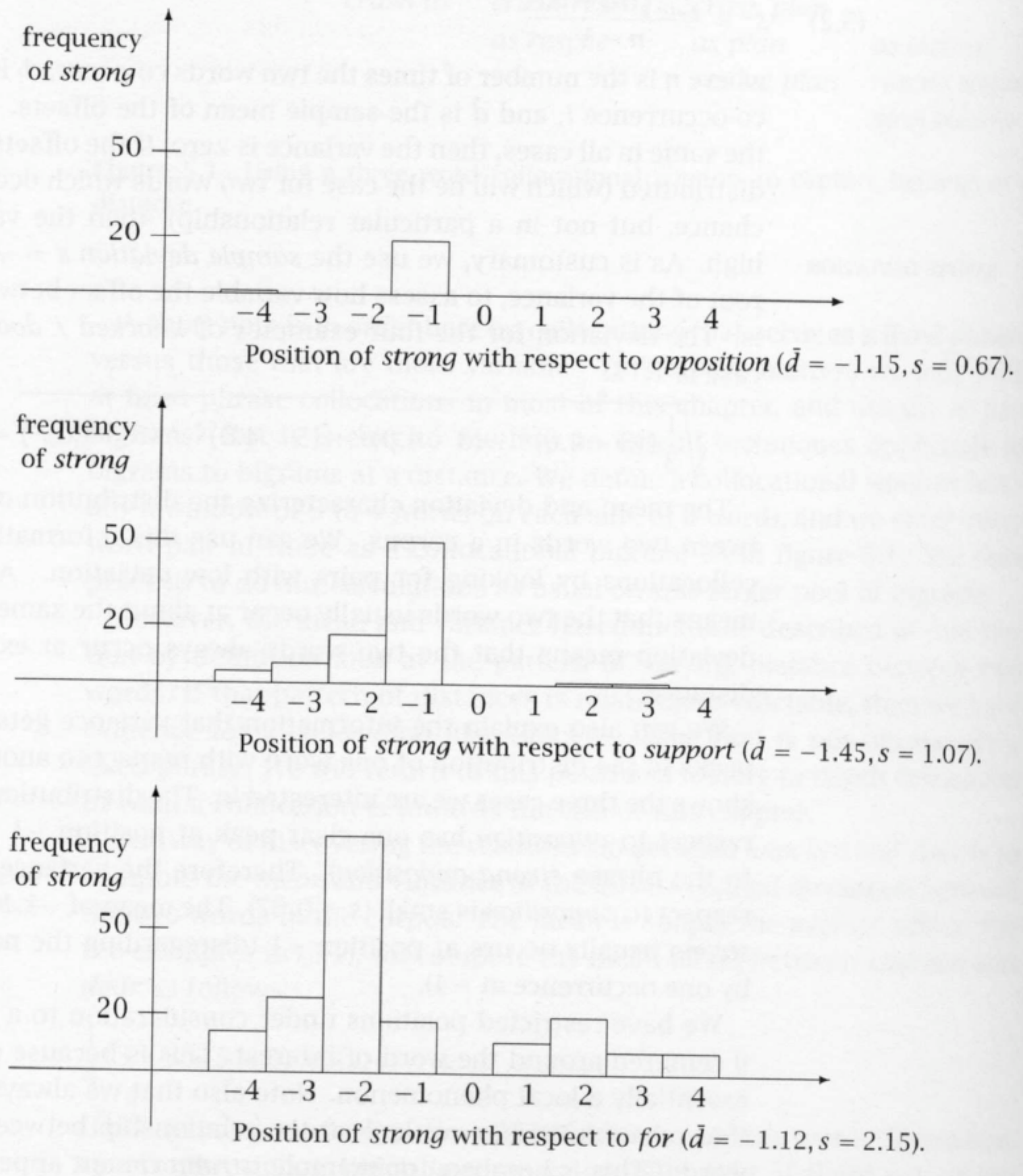


Figure 5.2 Histograms of the position of *strong* relative to three words.

Figure from Manning and Schutze

s	$\bar{d}$	Count	Word 1	Word 2
0.43	0.97	11657	New	York
0.48	1.83	24	previous	games
0.15	2.98	46	minus	points
0.49	3.87	131	hundreds	dollars
4.03	0.44	36	editorial	Atlanta
4.03	0.00	78	ring	New
3.96	0.19	119	point	hundredth
3.96	0.29	106	subscribers	by
1.07	1.45	80	strong	support
1.13	2.57	7	powerful	organizations
1.01	2.00	112	Richard	Nixon
1.05	0.00	10	Garrison	said

Table 5.5 Finding collocations based on mean and variance. Sample deviation s and sample mean $\bar{d}$ of the distances between 12 word pairs.

Figure from Manning and Schutze

The t-test

- We have a data set x_i
- We wish to test the hypothesis that this data set comes from a univariate normal distribution of mean μ
- we compute

$$T = \frac{\overline{x} - \mu}{\sqrt{\frac{s^2}{N}}}$$

- this has known distribution. We can look up in tables
  - P(T=obs|mu)
  - if this is too small, we reject

T-test and collocations

- Work with bigram counts
- Null hypothesis: P(w1 w2)= P(w1)P(w2)
- Compute numbers from frequencies
- Pretend P(w1 w2) is normal
- Compute T statistic, test for significance
- Wrinkle
  - almost nothing is significant
  - instead, rank by T

T-test and collocation: example

- Numbers:
  - #(tokens)=14, 307, 668
  - #(new)=15, 828
  - #(companies)=4, 675
  - #(bigrams)=14, 307, 668
  - #(new companies)=8
- Probabilities:
  - P(new companies)= P(new) P(companies)
    - (15828/14307668)*(4675/14307768)=3.615e-7
    - a bernoulli trial with p=3.615e-7
      - mean is 3.616e-7
      - variance is p(1-p)

T-test and collocation: example

- Probabilities:
  - P(new companies)=8/14307668=5.591e-7
  - again, bernoulli trial, p, so variance is approx 5.591e-7
- Statistics:
  - t=(5.591e-7-3.615e-7)/sqrt(5.591e-7/14307668)=0.999932
  - critical value for significance of 0.05 is t=2.576
  - can't reject null hypothesis

t	$C(w^1)$	$C(w^2)$	$C(w^1\ w^2)$	w^1	w^2
4.4721	42	20	20	Ayatollah	Ruhollah
4.4721	41	27	20	Bette	Midler
4.4720	30	117	20	Agatha	Christie
4.4720	77	59	20	videocassette	recorder
4.4720	24	320	20	unsalted	butter
2.3714	14907	9017	20	first	made
2.2446	13484	10570	20	over	many
1.3685	14734	13478	20	into	them
1.2176	14093	14776	20	like	people
0.8036	15019	15629	20	time	last

Table 5.6 Finding collocations: The t test applied to 10 bigrams that occur with frequency 20.

Figure from Manning and Schutze

Another cute use of the t-test

- Which words best distinguish between two other words?
  - e.g. which words best distinguish between strong and powerful?
  - which words occur most significantly more often with strong than with powerful?
- Test:
  - are two sets of data from different normal distributions?
  - form:

$$T = \frac{\overline{x}_1 - \overline{x}_2}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}$$

  - which has a t- distribution

Comparing collocates

- Example:
  - We want words such that P(strong| word) is very different from P(powerful|word)
  - bernoulli distribution, variance from this, compute t, rank

t	$C(w)$	$C(strong\ w)$	$C(powerful\ w)$	word
3.1622	933	0	10	computers
2.8284	2337	0	8	computer
2.4494	289	0	6	symbol
2.4494	588	0	6	machines
2.2360	2266	0	5	Germany
2.2360	3745	0	5	nation
2.2360	395	0	5	chip
2.1828	3418	4	13	force
2.0000	1403	0	4	friends
2.0000	267	0	4	neighbor
7.0710	3685	50	0	support
6.3257	3616	58	7	enough
4.6904	986	22	0	safety
4.5825	3741	21	0	sales
4.0249	1093	19	1	opposition
3.9000	802	18	1	showing
3.9000	1641	18	1	sense
3.7416	2501	14	0	defense
3.6055	851	13	0	gains
3.6055	832	13	0	criticism

Table 5.7 Words that occur significantly more often with *powerful* (the first ten words) and *strong* (the last ten words).

Figure from Manning and Schutze

Chi-square testing

- T-test assumes normal distributions
  - but data isn't normal
- Chi-square tests difference between observed values and values expected under null hypothesis
- Statistic:

$$X^2 = \sum_{i,j} \frac{(O_{ij} - E_{ij})^2}{E_{ij}}$$

  - has known distribution
  - can look up probability that this statistic has this value under null hypothesis

Chi-square and collocations

- Assume that the words are independent; then we can get probabilities from counts, table should look like:

	w1=new	w1 $\neq$ new
w2=companies	$P(c)P(n)N$	$P(c)(1-P(n))N$
w2$\neq$companies	$(1-P(c))P(n)N$	$(1-P(c))(1-P(n))N$

Example:

	$w_1 = new$	$w_1 \neq new$
$w_2 = companies$	8 (*new companies*)	4667 (e.g., *old companies*)
$w_2 \neq companies$	15820 (e.g., *new machines*)	14287181 (e.g., *old machines*)

Table 5.8 A 2-by-2 table showing the dependence of occurrences of *new* and *companies*. There are 8 occurrences of *new companies* in the corpus, 4,667 bigrams where the second word is *companies*, but the first word is not *new*, 15,820 bigrams with the first word *new* and a second word different from *companies*, and 14,287,181 bigrams that contain neither word in the appropriate position.

- Here chi-squared is 1.55, and critical value is 3.841 for 0.05 significance

Figure from Manning and Schutze

Chi-squared and translation

- Take aligned sentence pairs
- for one french, one english word, form table
- e.g. are vache and cow independent?

	cow	¬ *cow*
vache	59	6
¬ *vache*	8	570934

Table 5.9 Correspondence of *vache* and *cow* in an aligned corpus. By applying the χ^2 test to this table one can determine whether *vache* and *cow* are translations of each other.

- No, chi-squared=456400

Figure from Manning and Schutze

Chi-square and corpus similarity

	corpus 1	corpus 2
word 1	60	9
word 2	500	76
word 3	124	20
	…	

Table 5.10 Testing for the independence of words in different corpora using χ^2. This test can be used as a metric for corpus similarity.

- Are two corpora drawn from same underlying source?
  - do they have the same word frequencies?

Figure from Manning and Schutze

Likelihood ratio tests

- Two hypotheses
  - H1: P(w2|w1)=p=P(w2|~w1)
  - H2: P(w2|w1)=q which is not r=P(w2|~w1)
- Estimate p, q, r by counts
- now compute
- P(counts|H1)/P(counts|H2)

$-2 \log \lambda$	$C(w^1)$	$C(w^2)$	$C(w^1 w^2)$	w^1	w^2
1291.42	12593	932	150	most	powerful
99.31	379	932	10	politically	powerful
82.96	932	934	10	powerful	computers
80.39	932	3424	13	powerful	force
57.27	932	291	6	powerful	symbol
51.66	932	40	4	powerful	lobbies
51.52	171	932	5	economically	powerful
51.05	932	43	4	powerful	magnet
50.83	4458	932	10	less	powerful
50.75	6252	932	11	very	powerful
49.36	932	2064	8	powerful	position
48.78	932	591	6	powerful	machines
47.42	932	2339	8	powerful	computer
43.23	932	16	3	powerful	magnets
43.10	932	396	5	powerful	chip
40.45	932	3694	8	powerful	men
36.36	932	47	3	powerful	486
36.15	932	268	4	powerful	neighbor
35.24	932	5245	8	powerful	political
34.15	932	3	2	powerful	cudgels

Table 5.12 Bigrams of *powerful* with the highest scores according to Dunning's likelihood ratio test.

Figure from Manning and Schutze

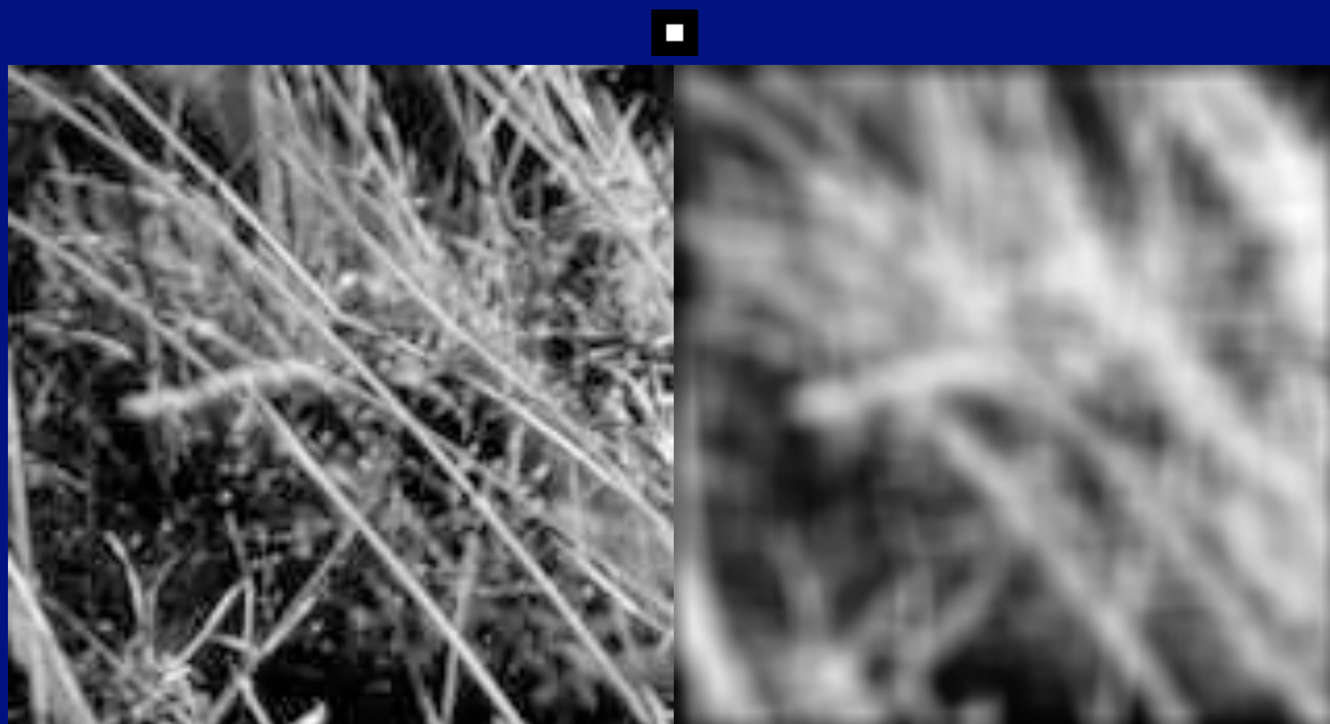
Computer Vision: Example problems

- Obstacle avoidance
  - A cricketer avoids being hit in the head (->) (<-)
  - the gannet pulls its wings in in time, by measuring time to contact
- Reconstructing representations of the 3D world
  - from multiple views
  - from shading
  - from structural models, etc
- Recognition
  - draw distinctions between what is seen
    - is it soggy?
    - will it eat me?
    - can I eat it?
    - is it a cat?
    - is it my cat?

Linear Filters

- Example: smoothing by averaging
  - form the average of pixels in a neighbourhood
- Example: smoothing with a Gaussian
  - form a weighted average of pixels in a neighbourhood
- Example: finding a derivative
  - form a weighted average of pixels in a neighbourhood

Smoothing by Averaging

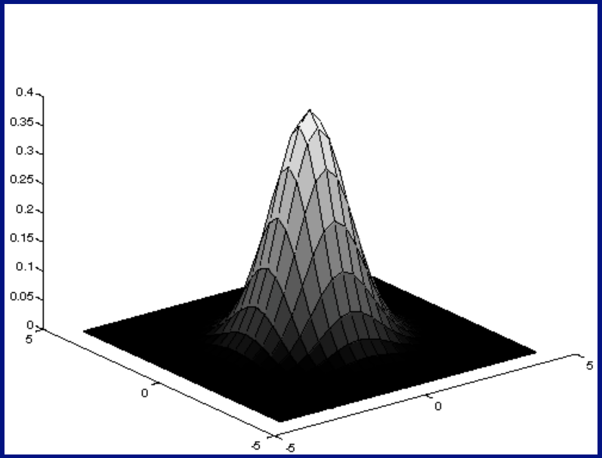


$$N_{ij} = \frac{1}{N} \Sigma_{uv} O_{i+u,j+v}$$

where u, v, is a window of N pixels in total centered at 0, 0

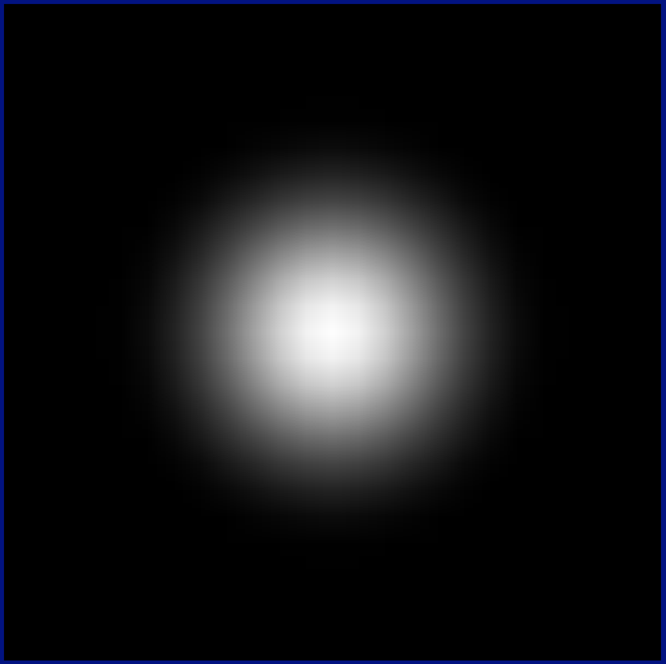
Smoothing with a Gaussian

- Notice “ringing”
  - apparently, a grid is superimposed
- Smoothing with an average actually doesn’t compare at all well with a defocussed lens
  - what does a point of light produce?



- A Gaussian gives a good model of a fuzzy blob

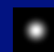
Gaussian filter kernel

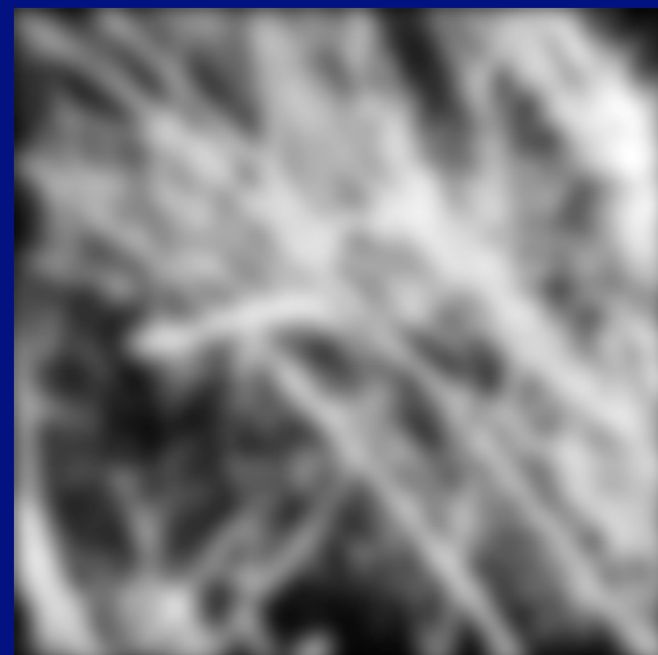


$$K_{uv} = \left(\frac{1}{2\pi\sigma^2}\right)\exp\left(\frac{-\left[u^2+v^2\right]}{2\sigma^2}\right)$$

We're assuming the index can take negative values

Smoothing with a Gaussian

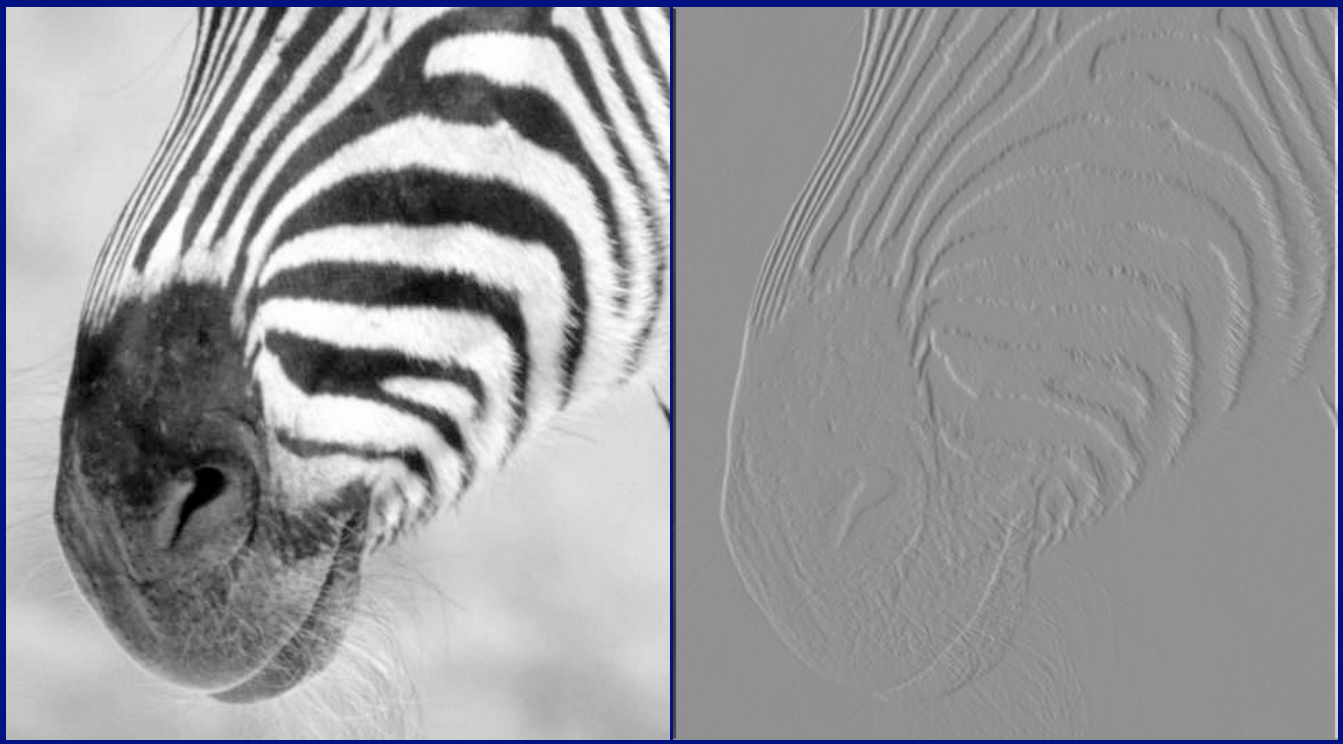




$$N_{ij} = \sum_{uv} O_{i-u,j-v} K_{uv}$$

Notice the curious looking form

Finding derivatives



$$N_{ij} = \frac{1}{\Delta x}(I_{i+1,j} - I_{ij})$$

Convolution

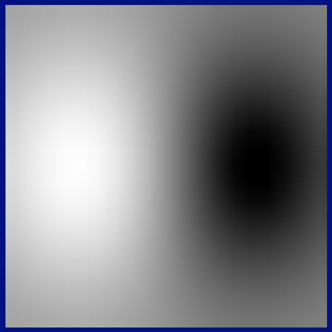
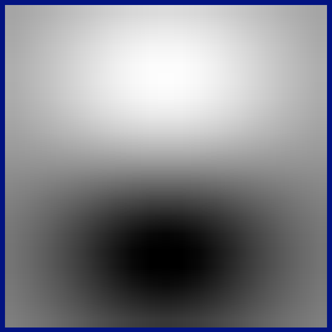
- Each of these involves a weighted sum of image pixels
- The set of weights is the same
  - we represent these weights as an image, H
  - H is usually called the kernel
- Operation is called convolution
  - it's associative
- Any linear shift-invariant operation can be represented by convolution
  - linear: G(k f)=k G(f)
  - shift invariant: G(Shift(f))=Shift(G(f))
  - Examples:
    - smoothing, differentiation, camera with a reasonable, defocussed lens system

$$N_{ij} = \sum_{uv} H_{uv} O_{i-u,j-v}$$

Filters are templates

$$N_{ij} = \sum_{uv} H_{uv} O_{i-u,j-v}$$

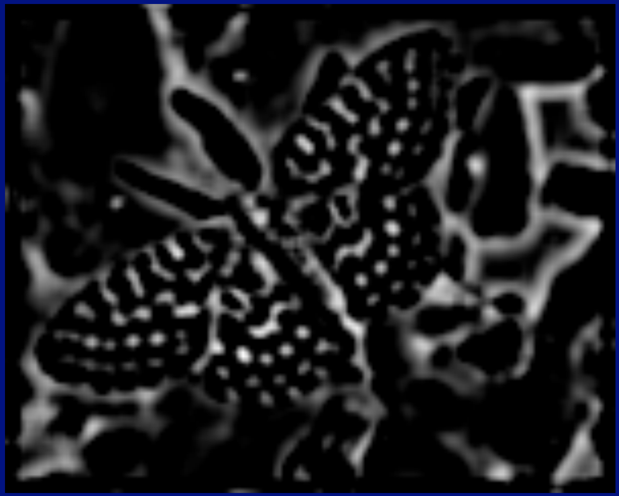
- At one point
  - output of convolution is a (strange) dot-product
- Filtering the image involves a dot product at each point
- Insight
  - filters look like the effects they are intended to find
  - filters find effects they look like

Normalised correlation

- Think of filters of a dot product
  - now measure the angle
  - i.e normalised correlation output is filter output, divided by root sum of squares of values over which filter lies
  - Tricks:
    - ensure that filter has a zero response to a constant region
      - helps reduce response to irrelevant background
    - subtract image average when computing the normalising constant
    - absolute value deals with contrast reversal

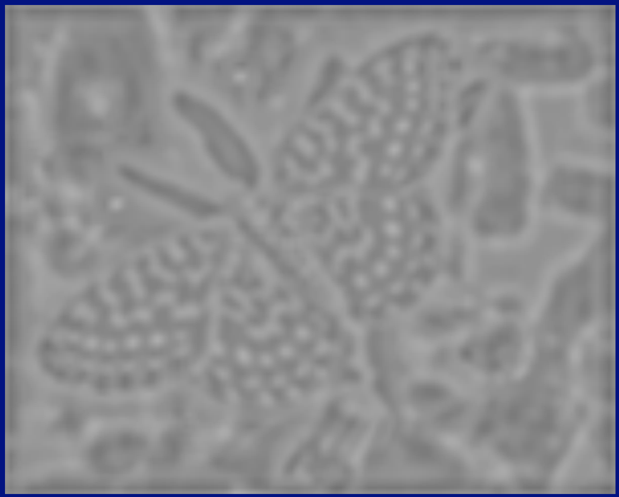
normalised correlation
with non-zero mean filter

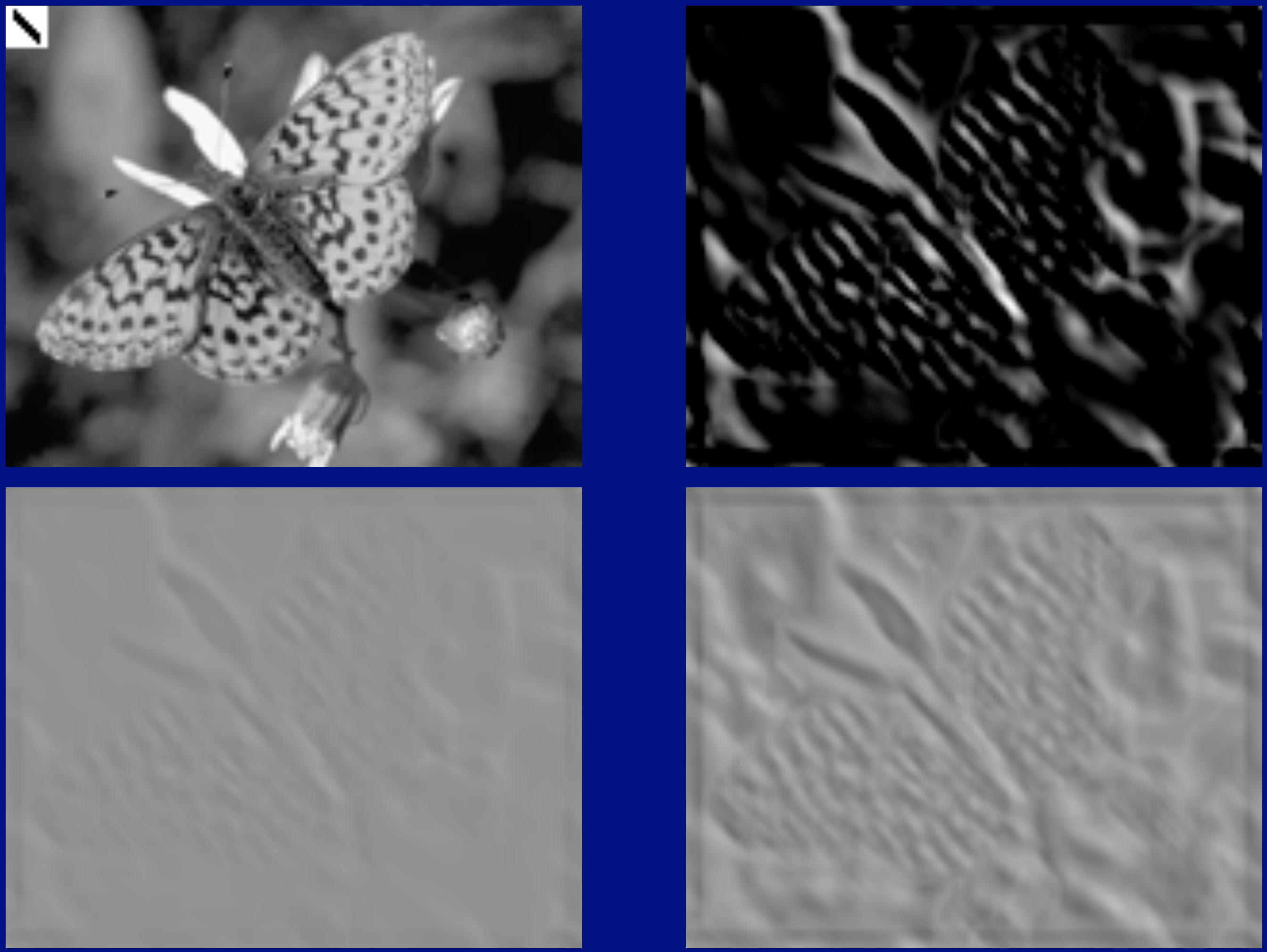


Positive responses

Zero mean image, -1:1 scale

Zero mean image, -max:max scale





Finding hands

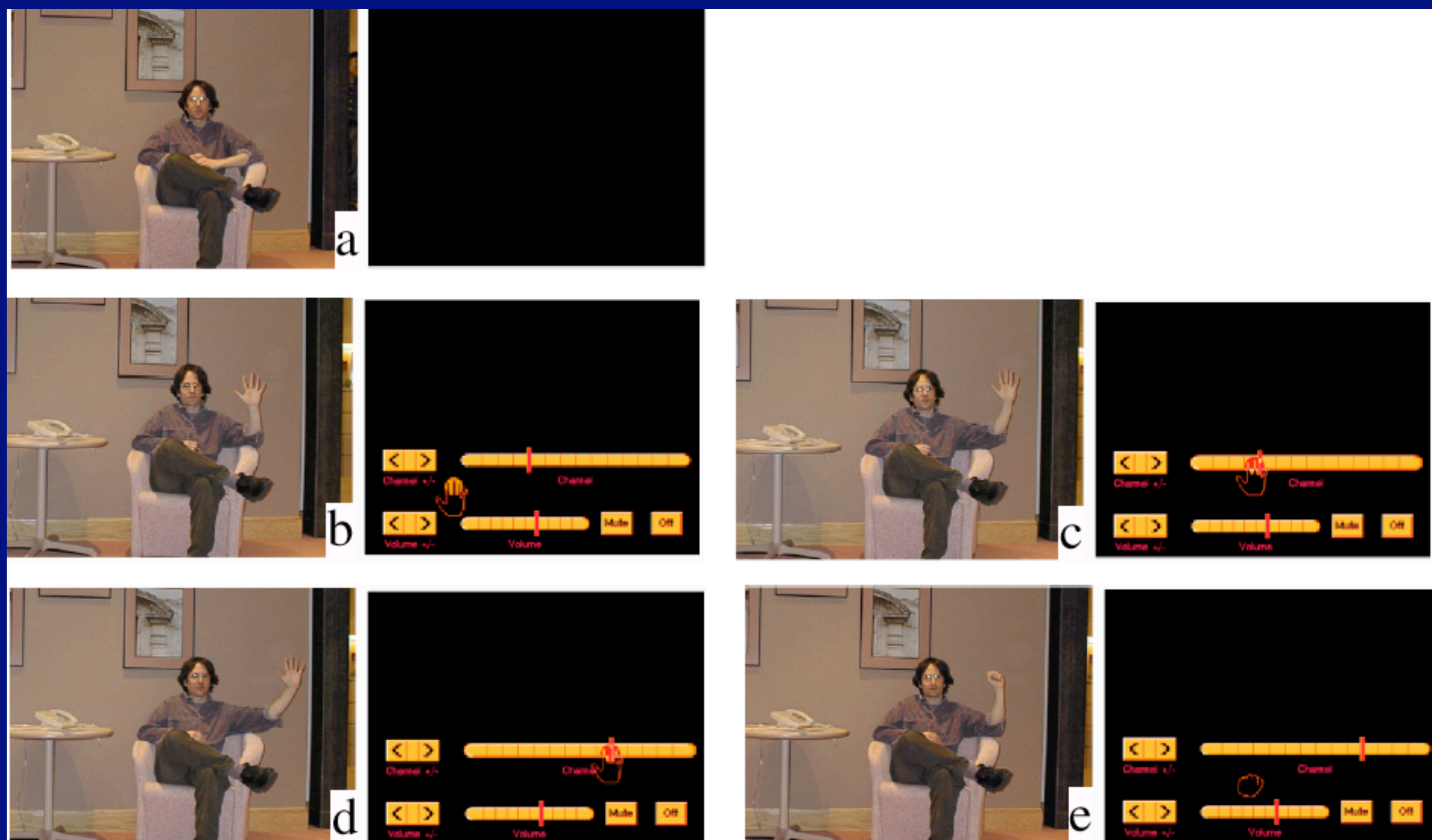


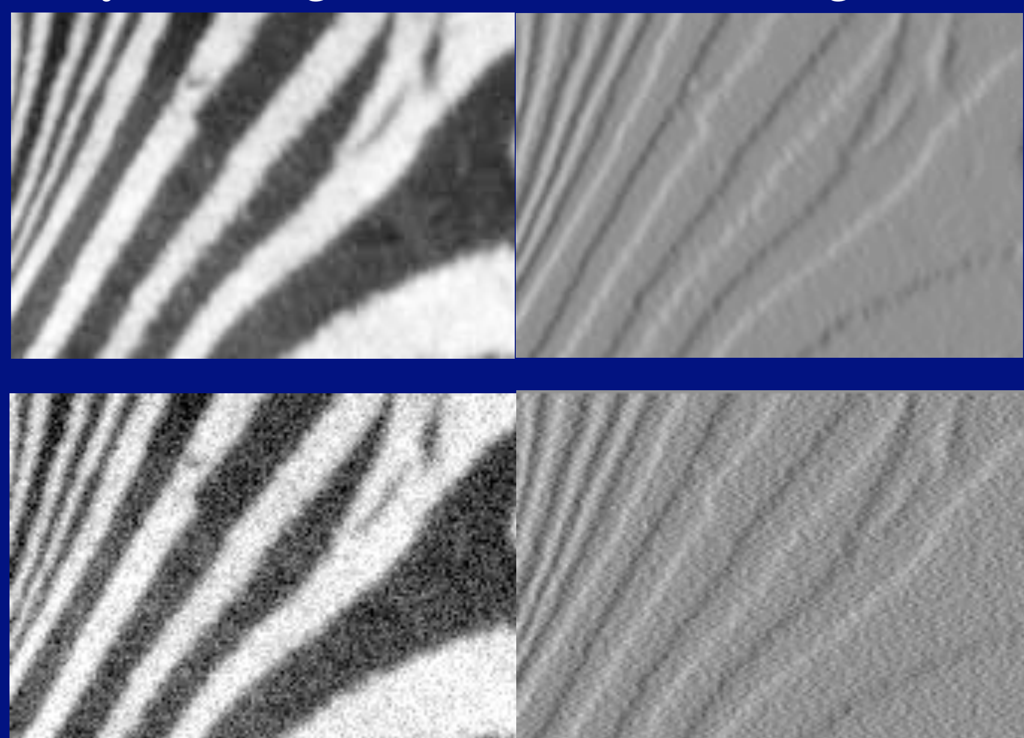
Figure from "Computer Vision for Interactive Computer Graphics," W.Freeman et al, IEEE Computer Graphics and Applications, 1998

Gradients and edges

- Points of sharp change in an image are interesting:
  - change in reflectance
  - change in object
  - change in illumination
  - noise
- Sometimes called edge points
- General strategy
  - determine image gradient
  - now mark points where gradient magnitude is particularly large wrt neighbours

Differentiation and noise

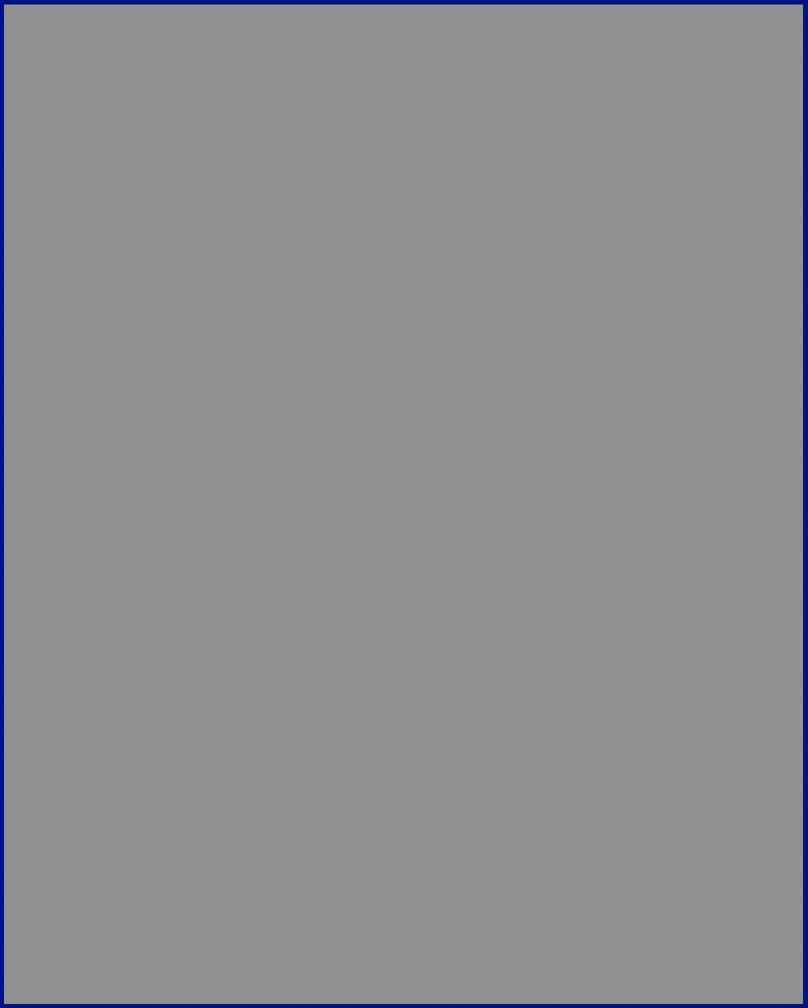
- Simple derivative filters respond strongly to noise
  - obvious reason: noise is associated with strong changes, as above
- Generally, the larger the noise the stronger the response



Noise

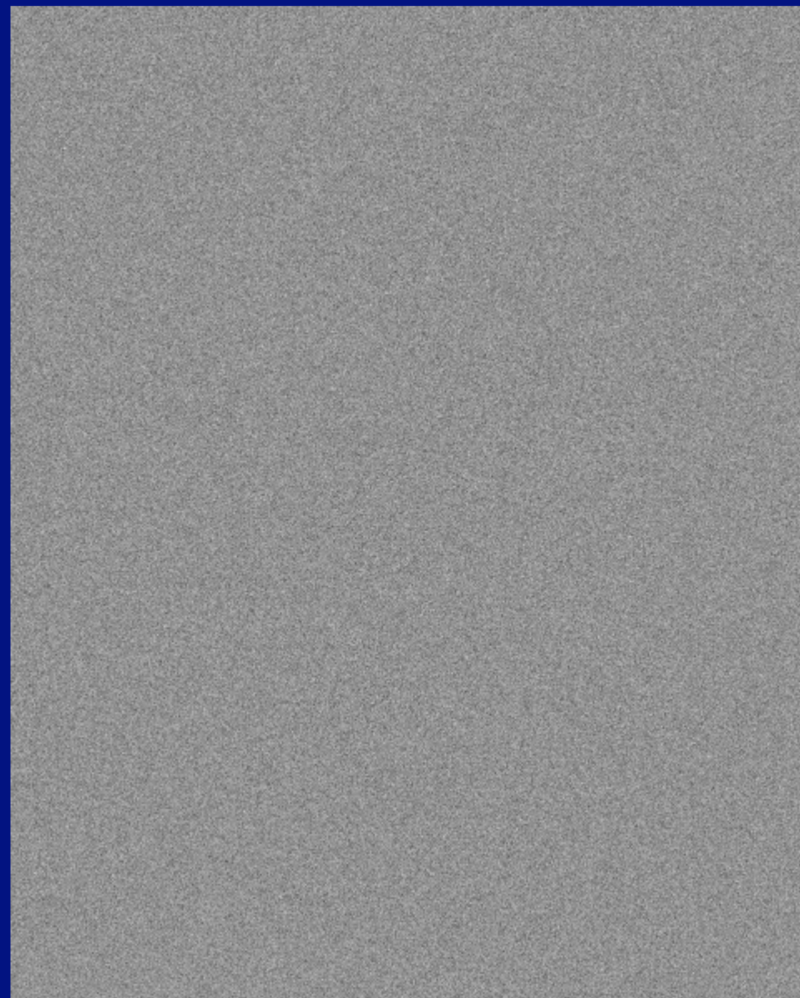
- Simplest noise model
  - independent stationary additive Gaussian noise
  - the noise value at each pixel is given by an independent draw from the same normal probability distribution
- Issues
  - allows values greater than maximum camera output or less than zero
    - for small standard deviations, this isn’t too much of a problem
  - independence may not be justified (e.g. damage to lens)
  - may not be stationary (e.g. thermal gradients in the ccd)

sigma=1



sigma=4

sigma=
16



sigma=1

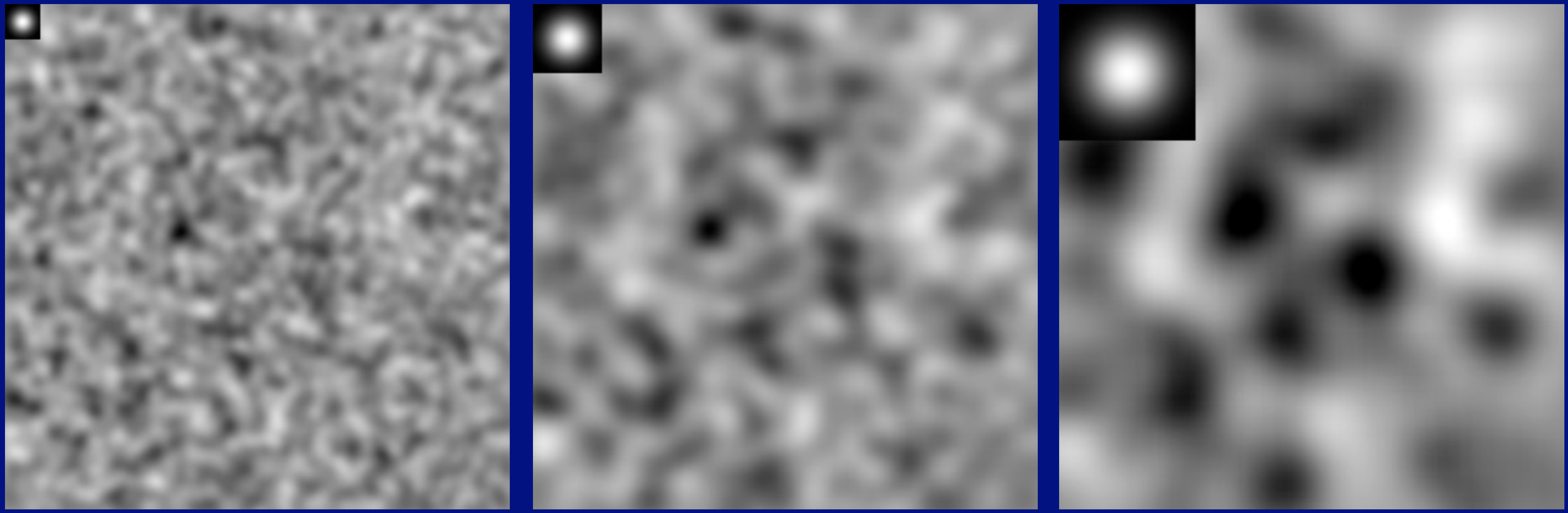
sigma=16

The response of a linear filter to noise

- Do only stationary independent additive Gaussian noise
  - get mean and variance of response by pattern matching
- Note that outputs are quite strongly correlated
  - useful trick for constructing simple textures

Filter responses are correlated

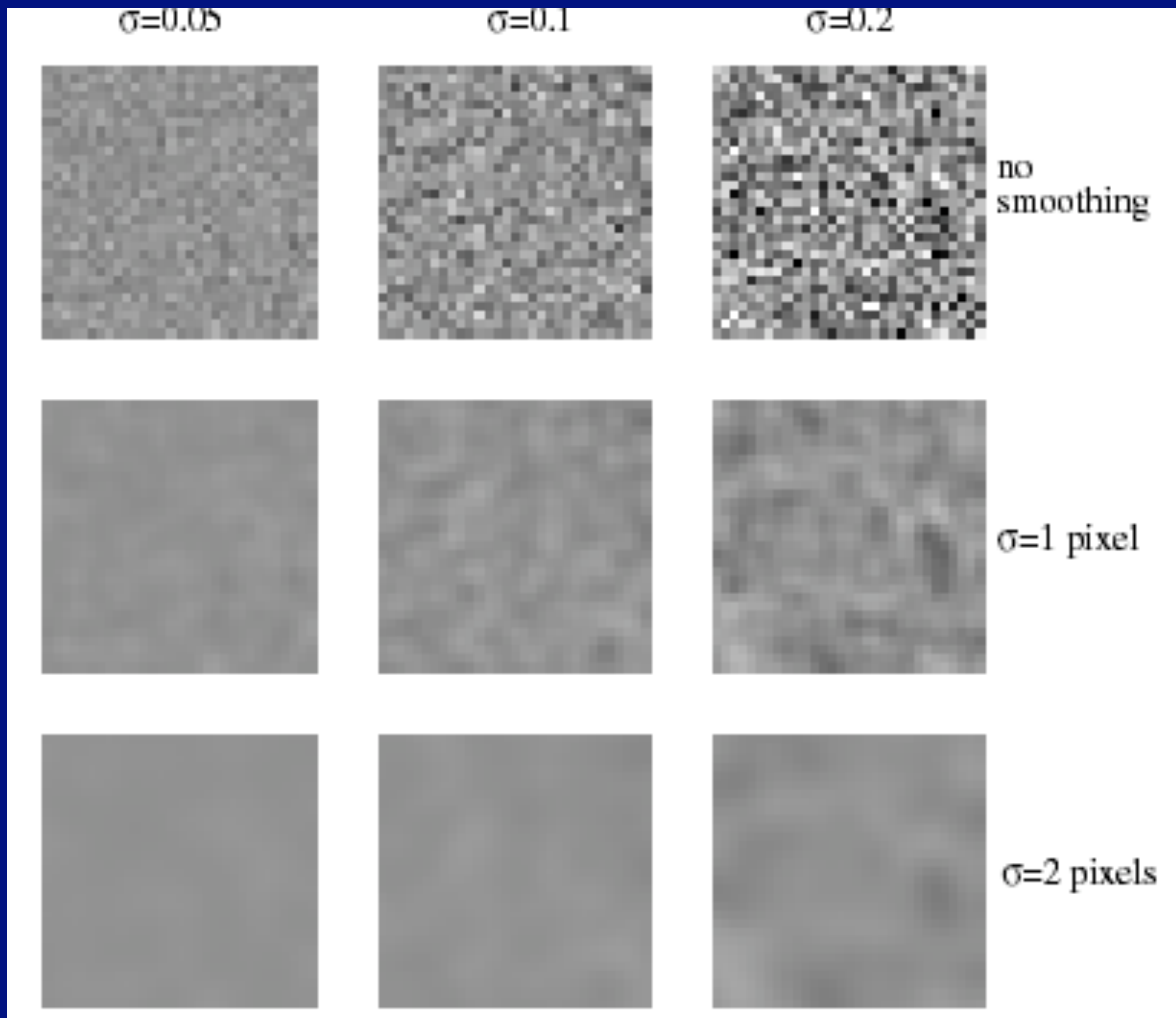
- (Fairly obviously) over scales similar to the scale of the filter



Smoothing reduces noise

- Generally expect pixels to "be like" their neighbours
  - surfaces turn slowly
  - relatively few reflectance changes
- Expect noise to be independent from pixel to pixel
  - Implies that smoothing suppresses noise, for appropriate noise models
- Scale
  - the parameter in the symmetric Gaussian
  - as this parameter goes up, more pixels are involved in the average
  - and the image gets more blurred
  - and noise is more effectively suppressed

$$K_{uv} = \left(\frac{1}{2\pi\sigma^2}\right)\exp\left(\frac{-\left[u^2+v^2\right]}{2\sigma^2}\right)$$


σ=0.05
σ=0.1
σ=0.2
no smoothing
σ=1 pixel
σ=2 pixels

More complex template matching

- Encode an object as a set of patches
  - centered on interest points
  - match by
    - voting
    - spatially censored voting
    - inference on a spatial model
- Patches are small
  - even if they’re on a curved surface, we can think of them as being plane

Correspondence

- Local representation of image properties make things easier
  - identify points which are easily localised
    - corners
    - which lie on edges
  - compare with points in next image
    - points which “look similar” may well match
  - search radius is constrained by geometry
    - in ways we will not discuss

Local Representations

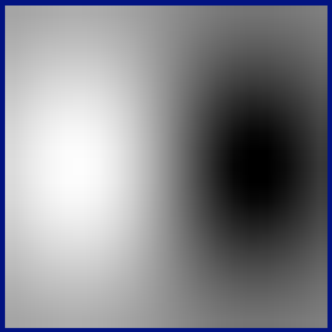
- What do edge responses look like nearby?
  - SIFT features
- What is the “general pattern” of grey levels?
  - statistics of filters

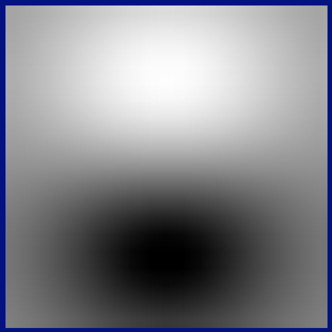
Edge detection

- Find points where image value changes sharply
- Strategy:
  - Estimate gradient magnitude using appropriate smoothing
  - Mark points where gradient magnitude is
    - Locally biggest and
    - big

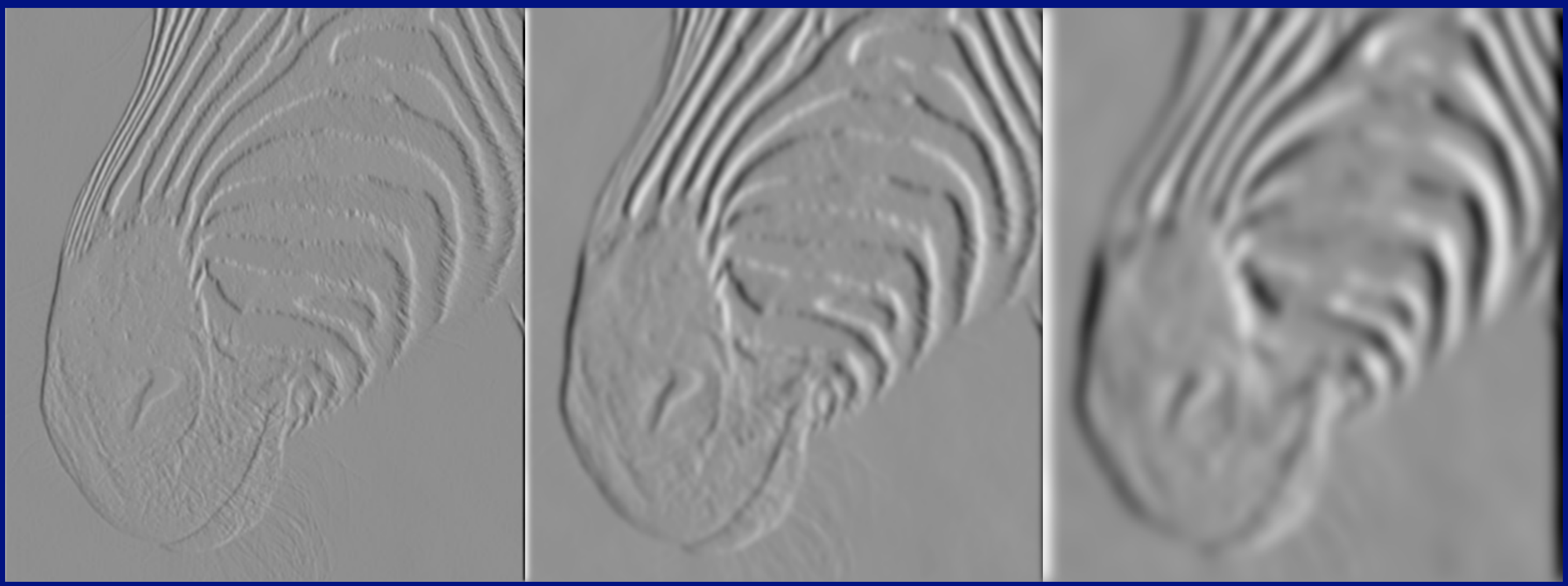
Smoothing and Differentiation

- Issue: noise
  - smooth before differentiation
  - two convolutions to smooth, then differentiate?
  - actually, no - we can use a derivative of Gaussian filter



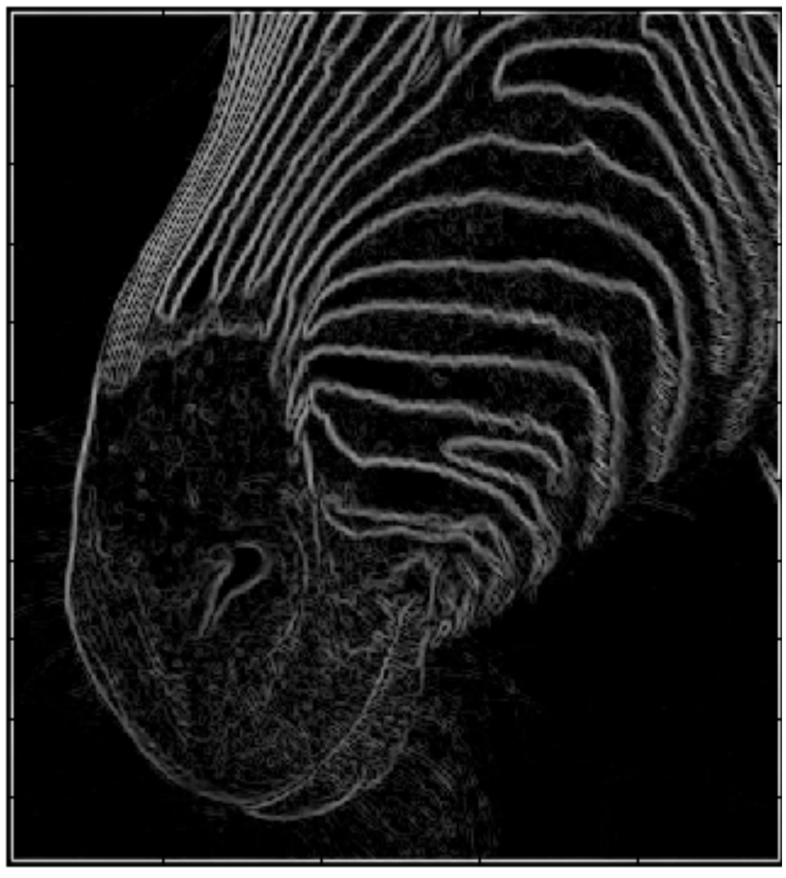


Scale affects derivatives

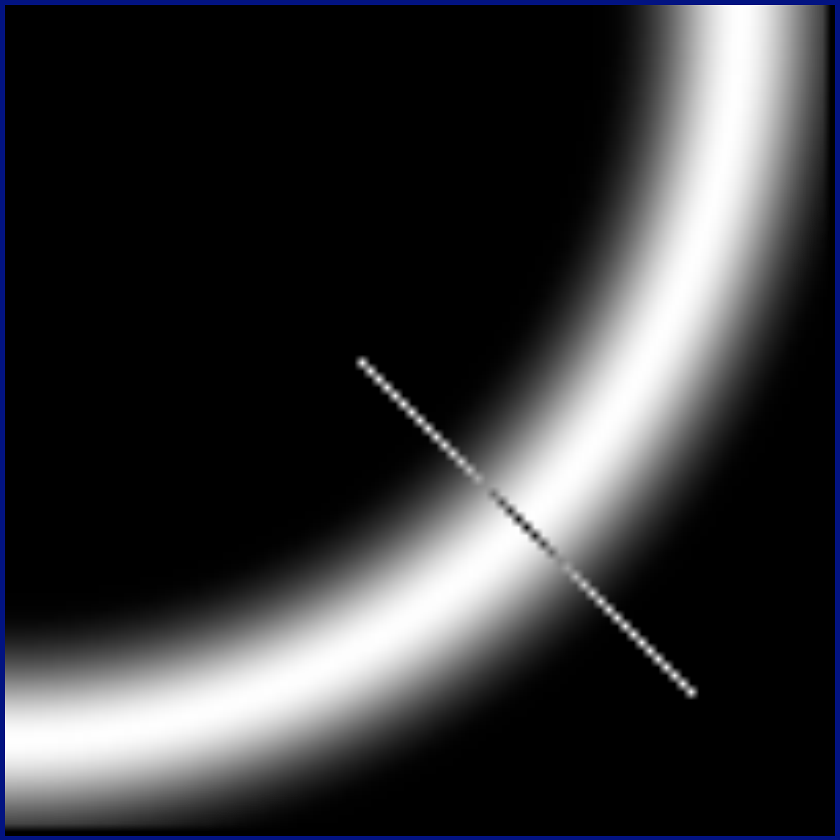


1 pixel 3 pixels 7 pixels

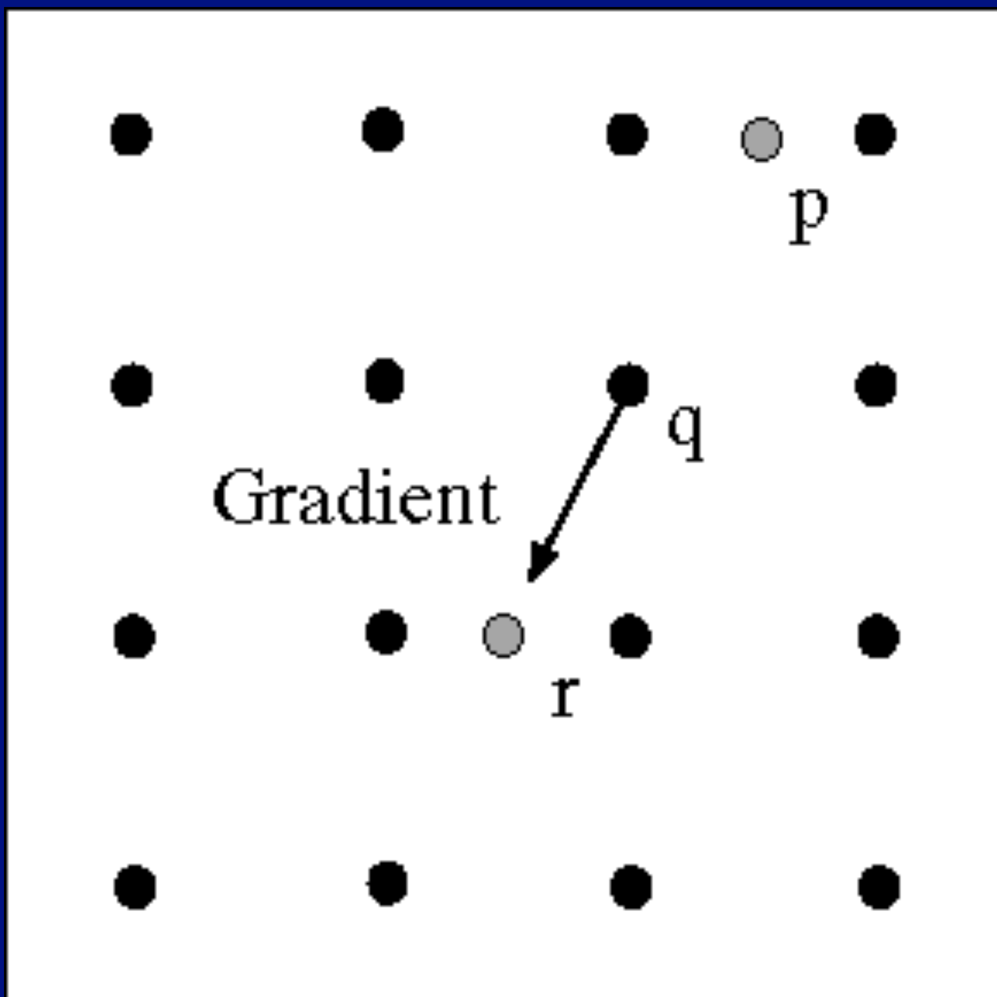
Scale affects gradient magnitude

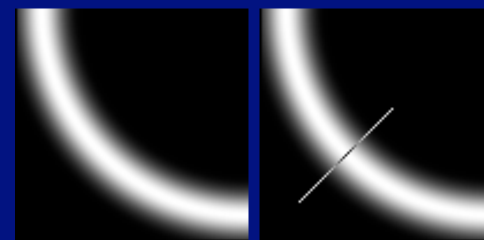


Marking the points

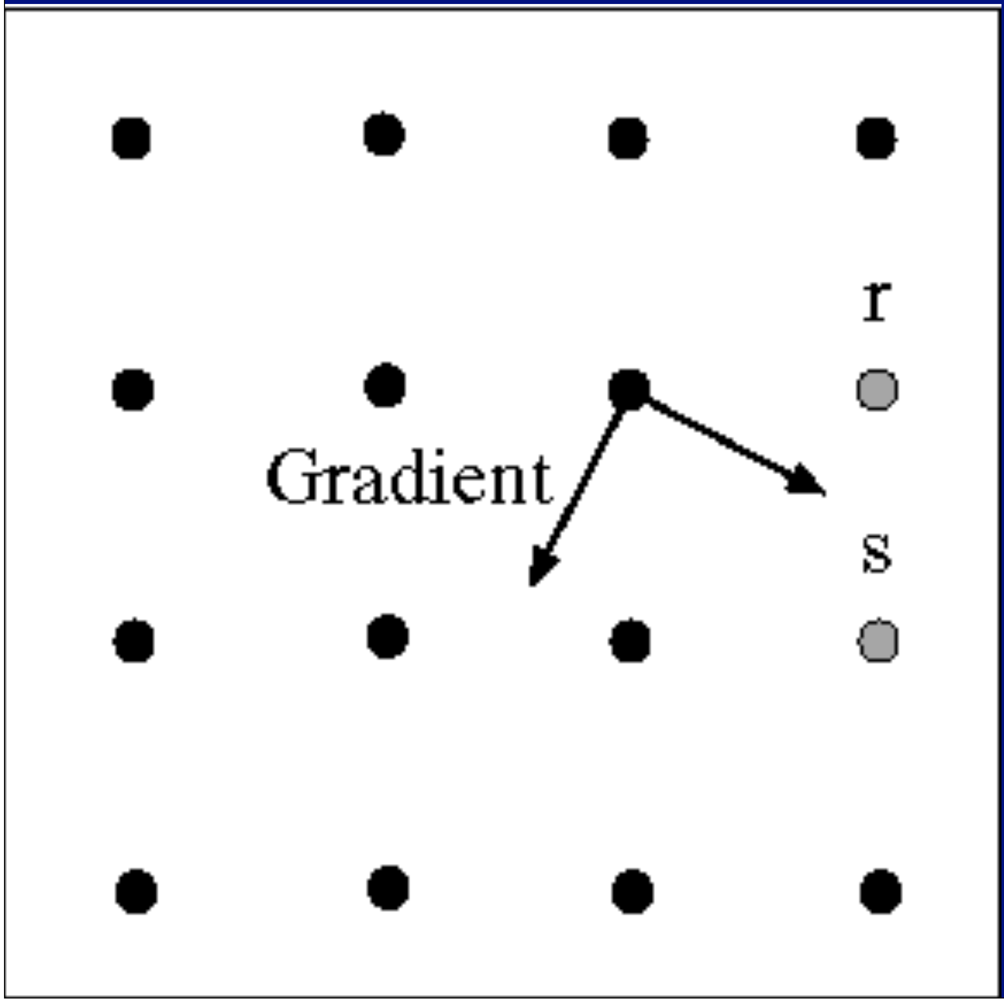


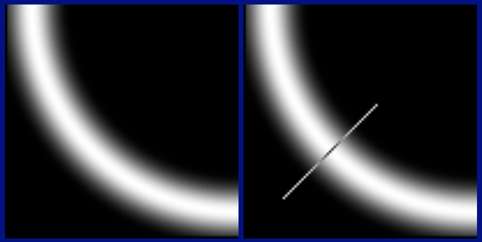
Non-maximum suppression





Predicting the next edge point



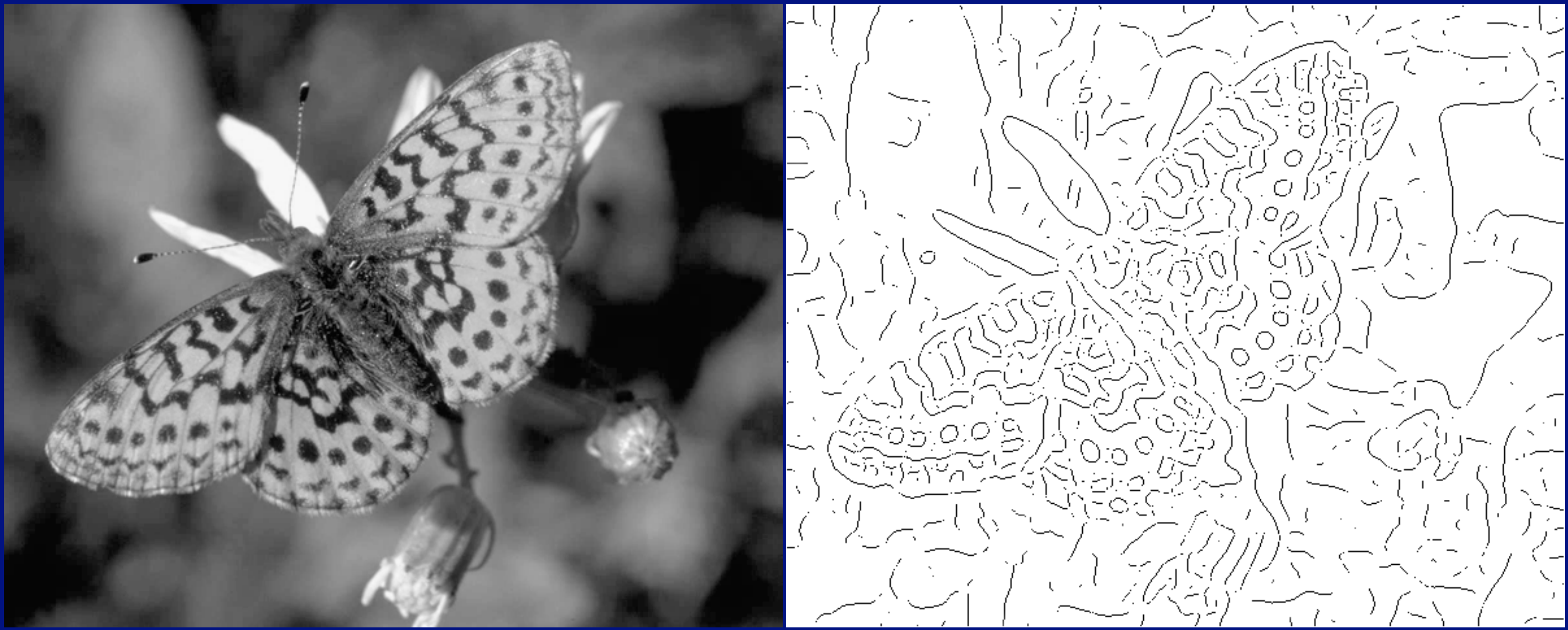


Remaining issues

- Check maximum value of gradient value is sufficiently large
  - drop-outs?
    - use hysteresis

Notice

- Something nasty is happening at corners
- Scale affects contrast
- Edges aren't bounding contours



The Laplacian of Gaussian

- Another way to detect an extremal first derivative is to look for a zero second derivative
- Appropriate 2D analogy is rotation invariant
- Zero crossings of Laplacian
  - Bad idea to apply a Laplacian without smoothing
  - smooth with Gaussian, apply Laplacian
  - this is the same as filtering with a Laplacian of Gaussian filter
  - Now mark the zero points where
    - there is a sufficiently large derivative,
    - and enough contrast

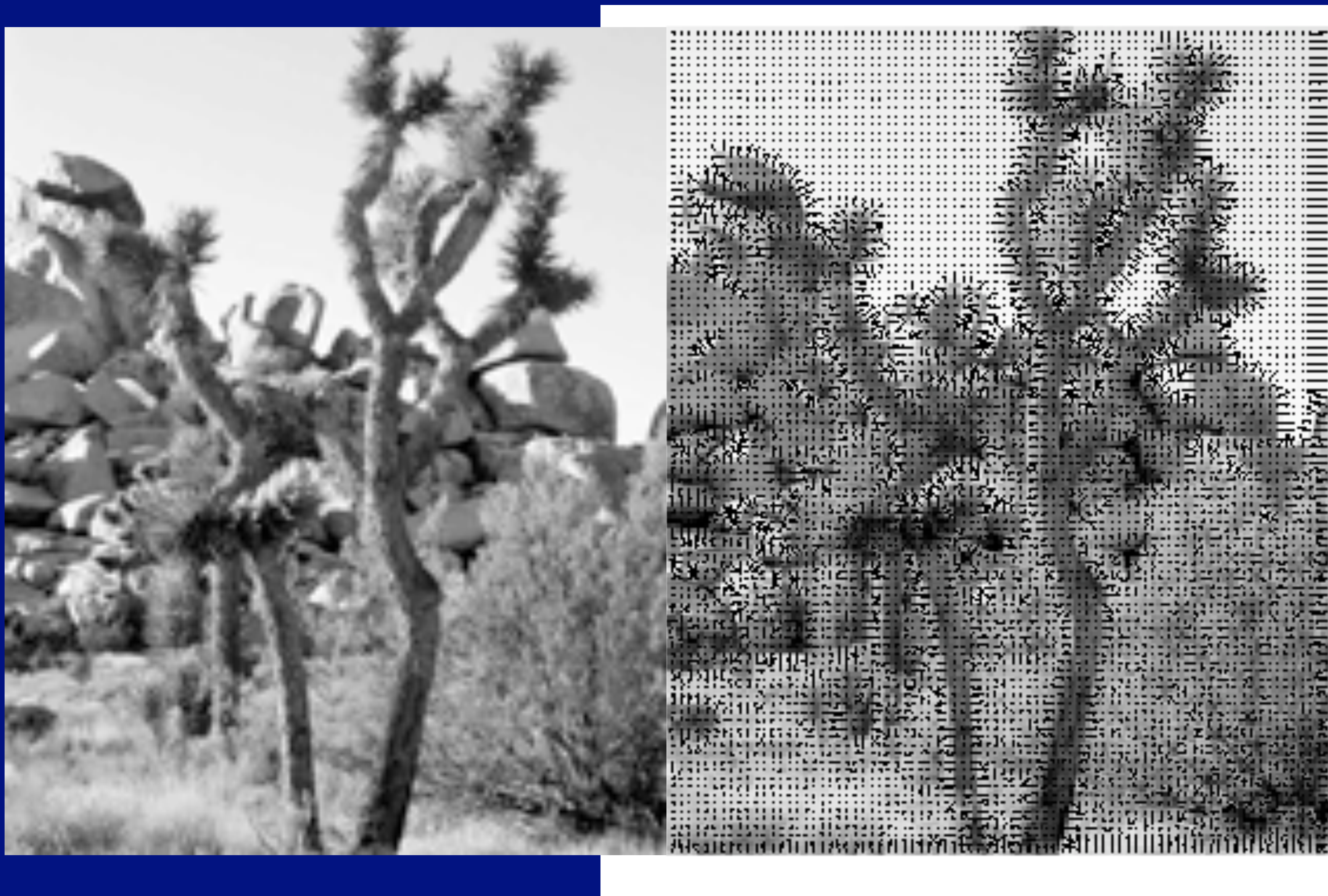
Orientation representations

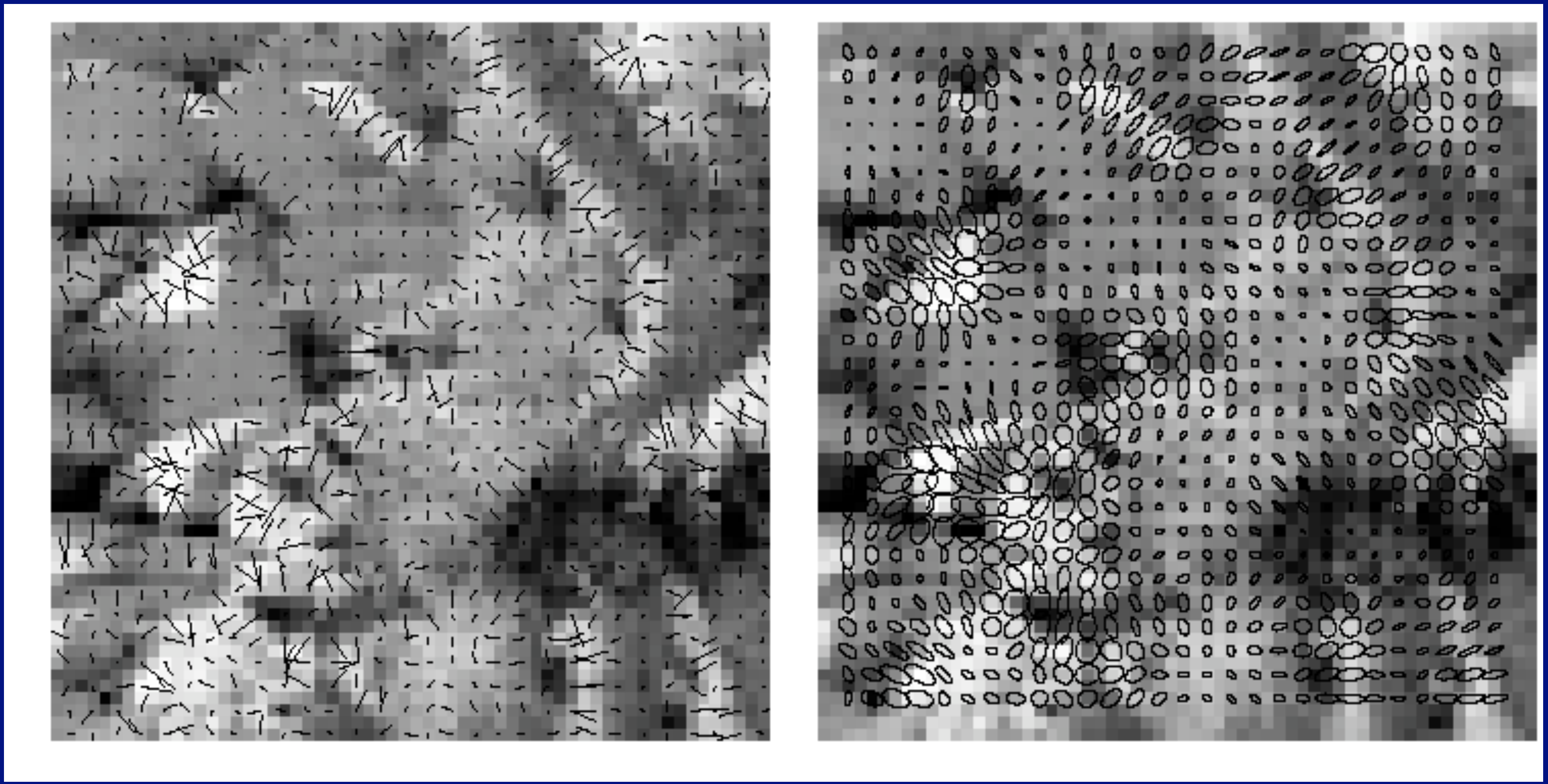
- Gradient magnitude is affected by illumination changes
  - but it's direction isn't
- Describe image patches by gradient direction
- Important types:
  - constant window
    - small gradient mags
  - edge window
    - few large gradient mags in one direction
  - flow window
    - many large gradient mags in one direction
  - corner window
    - large gradient mags that swing

Representing Windows

- Types
  - constant
    - small eigenvalues
  - Edge
    - one medium, one small
  - Flow
    - one large, one small
  - corner
    - two large eigenvalues

$$H = \sum_{\text{window}} (\nabla I)(\nabla I)^T$$



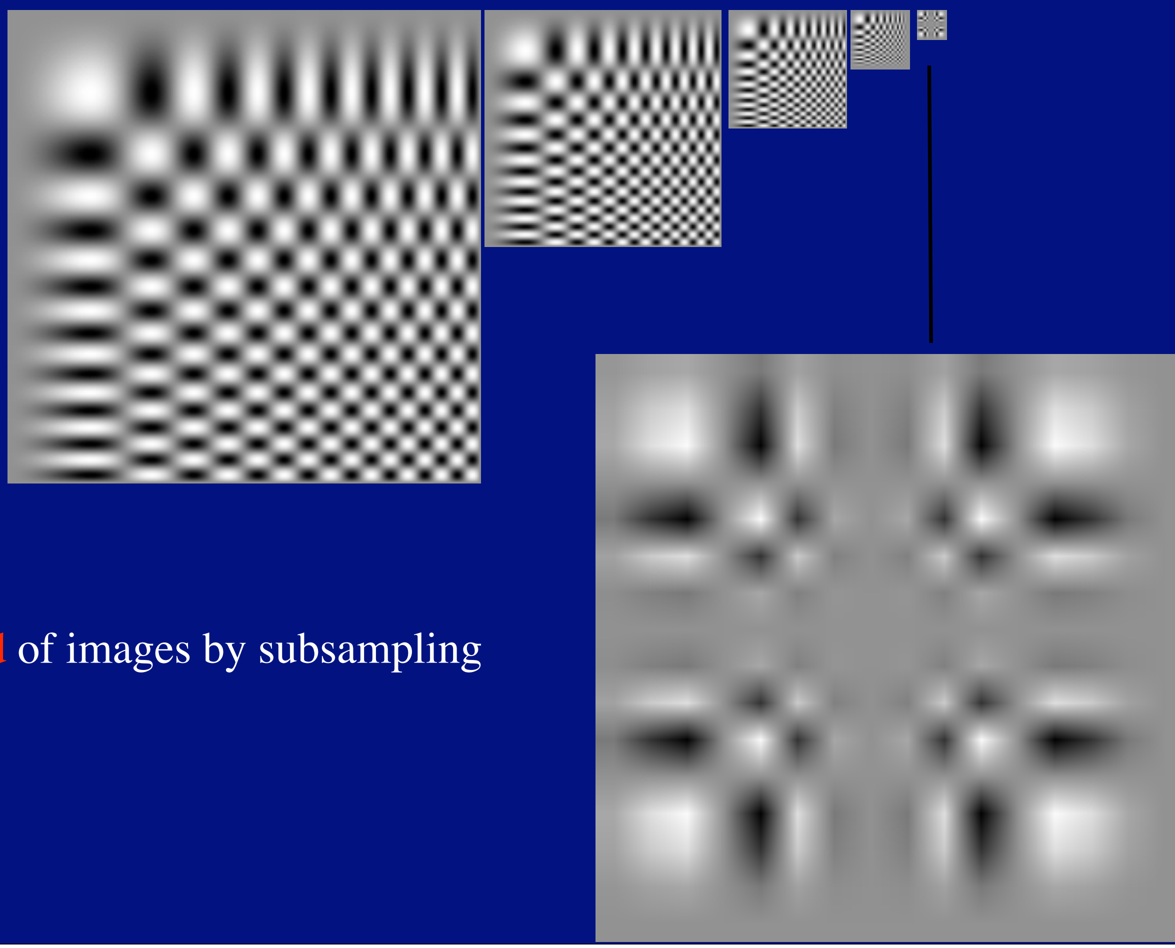


Plots of $\mathbf{x}H^{-1}\mathbf{x} = 0$

Scaled representations

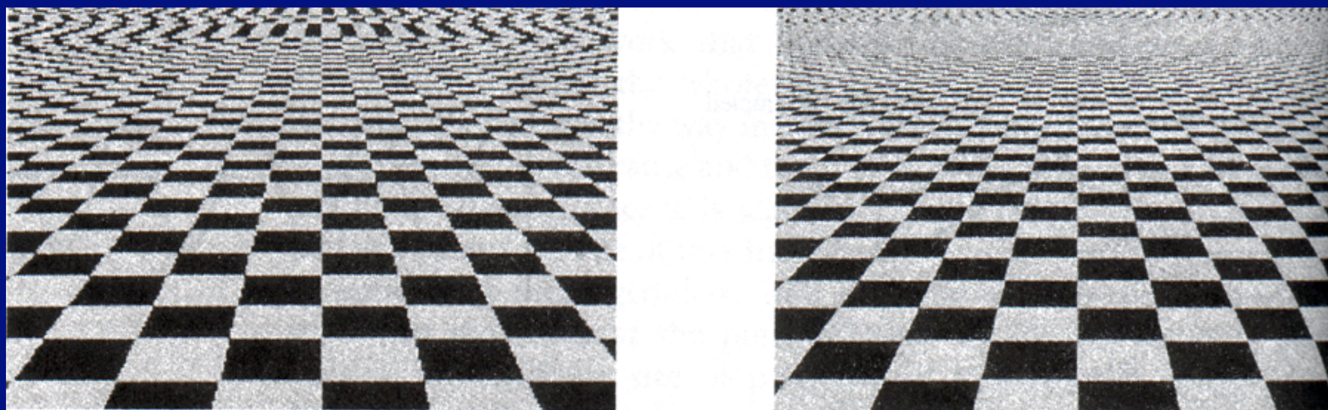
- Represent one image with many different resolutions
- Why?
  - Search for correspondence
    - look at coarse scales, then refine with finer scales
  - Edge tracking
    - a "good" edge at a fine scale has parents at a coarser scale
  - Control of detail and computational cost in matching
    - e.g. finding stripes
    - terribly important in texture representation

Carelessness causes aliasing



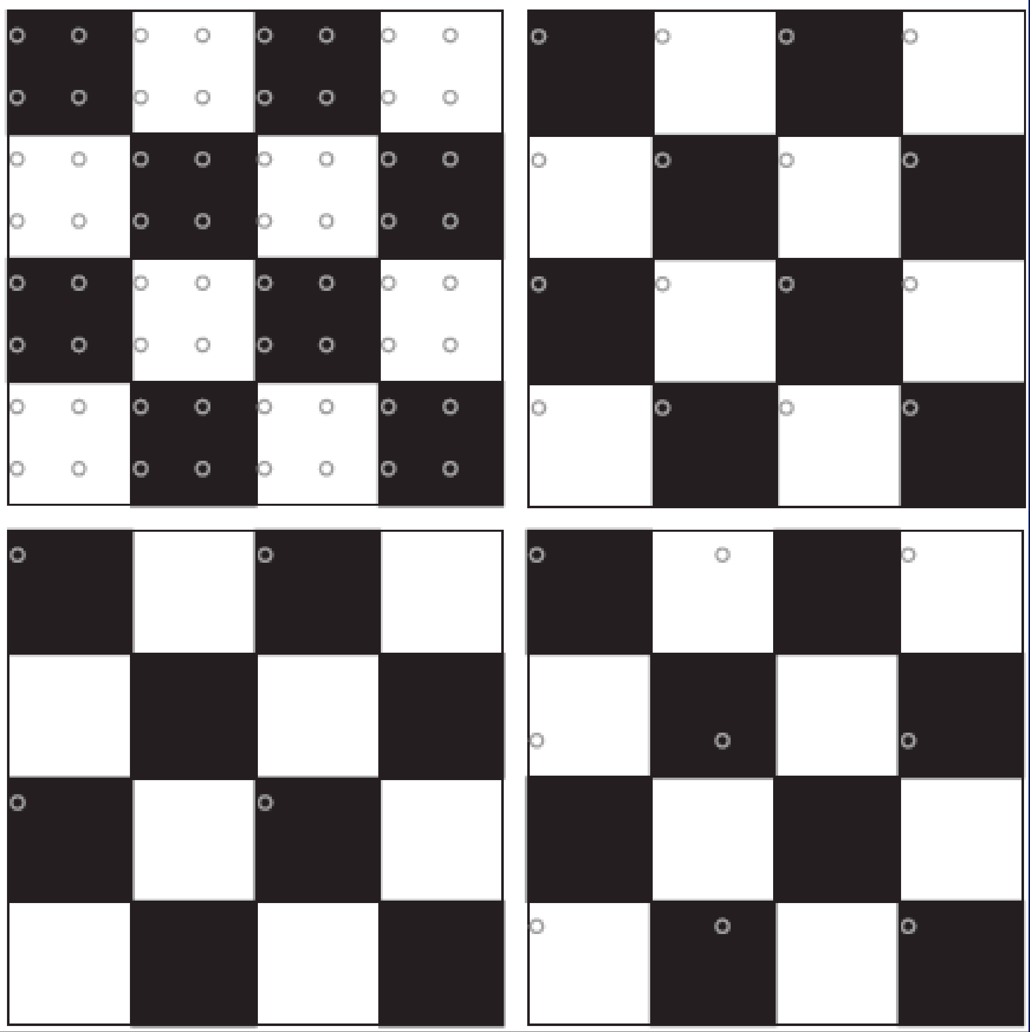
Obtained pyramid of images by subsampling

Aliasing



from Watt and Policarpo, The Computer Image

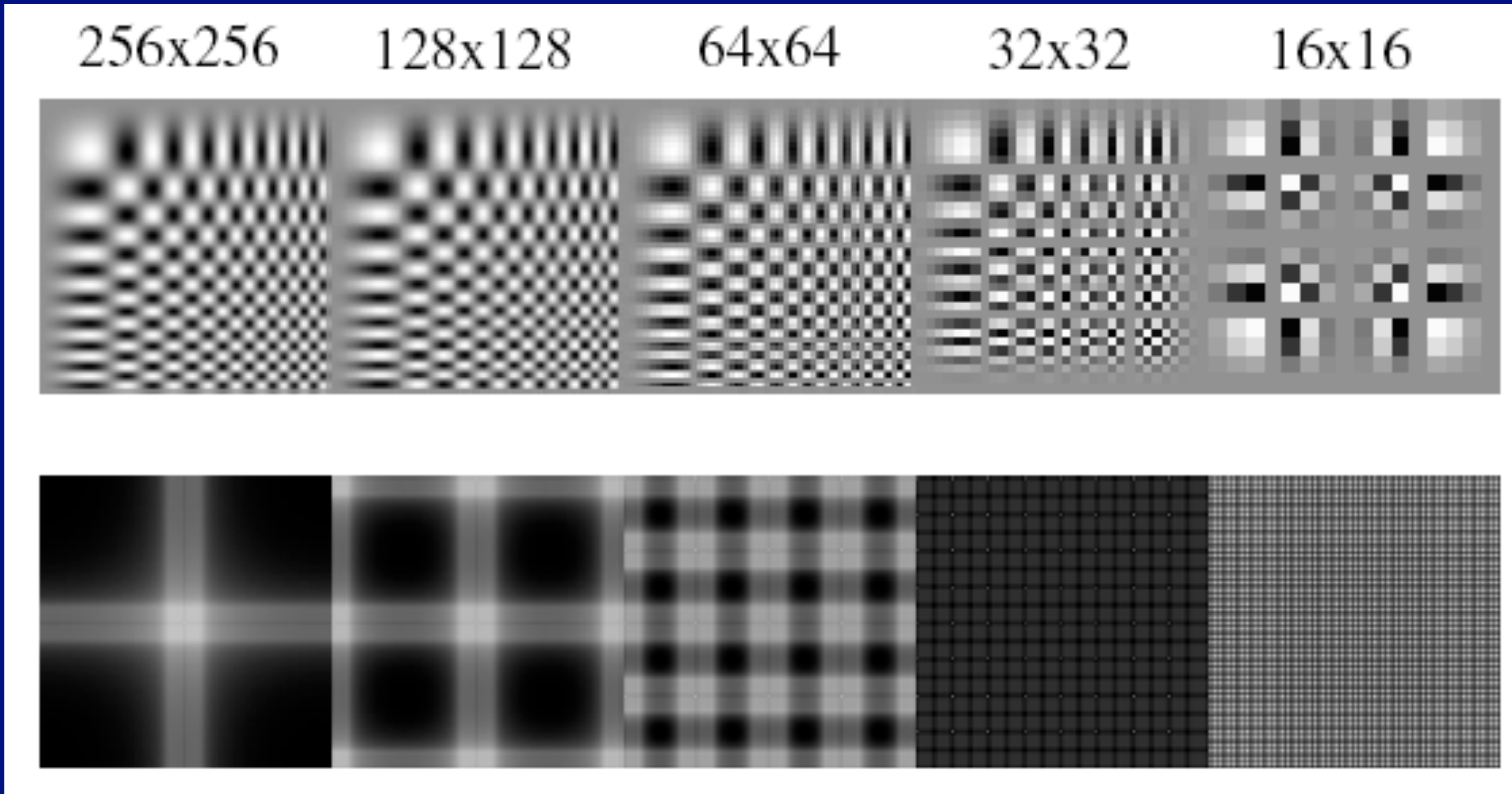
Aliasing - smoothing helps

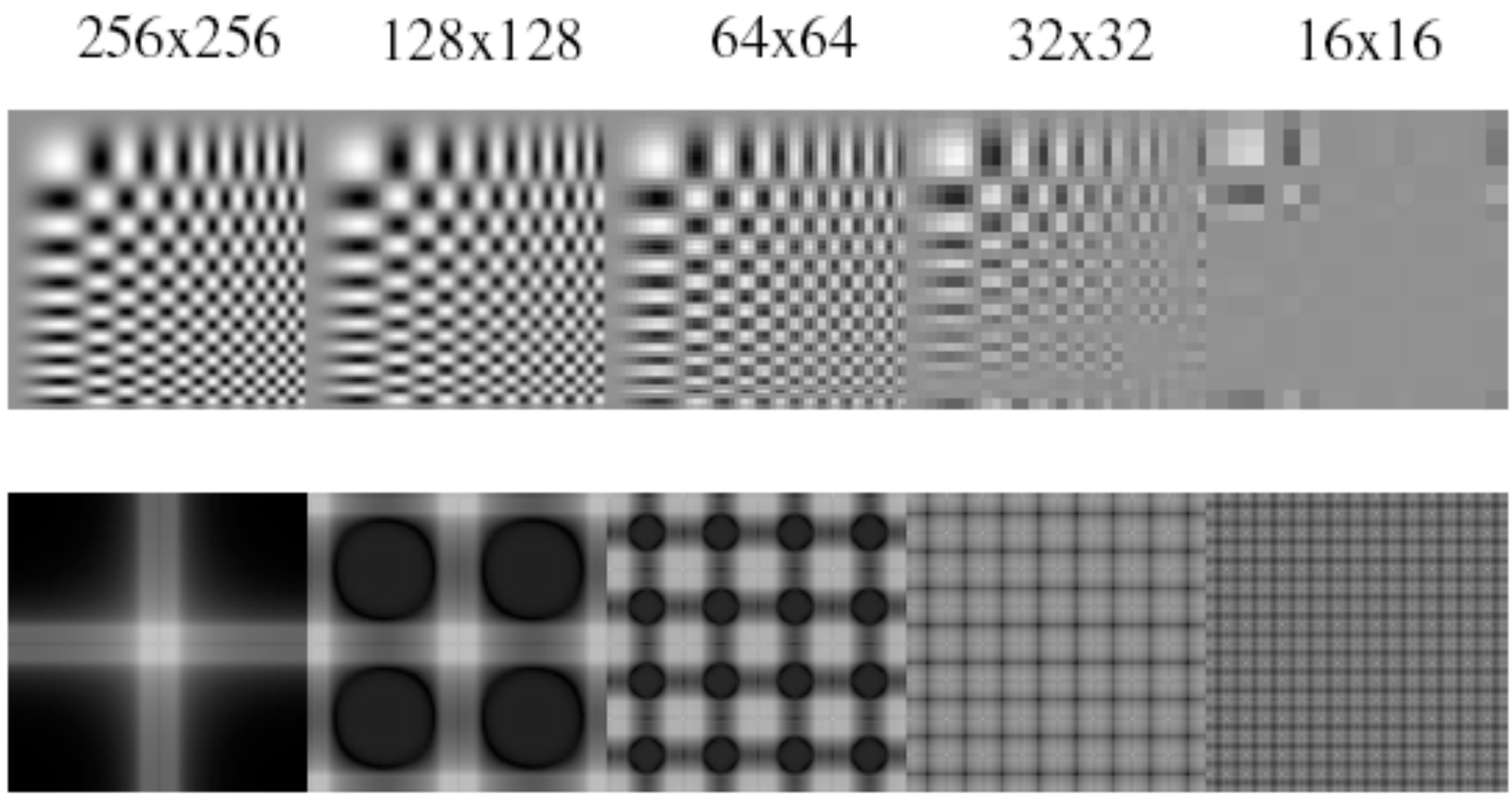


The Gaussian pyramid

- Smooth with gaussians, because
  - a gaussian*gaussian=another gaussian
- Synthesis
  - (making a pyramid from an image)
  - smooth and sample
- Analysis
  - (making an image from a pyramid)
  - take the top image
- Gaussians are low pass filters, so repn is redundant

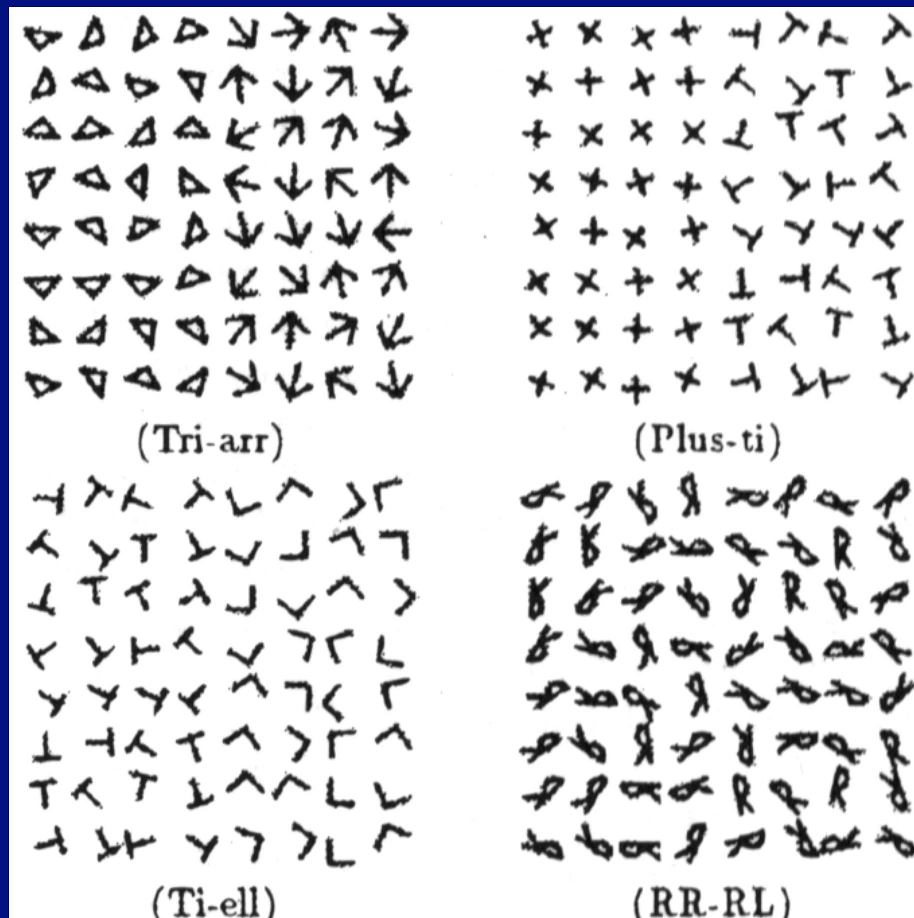

512
256
128
64
32
16
8

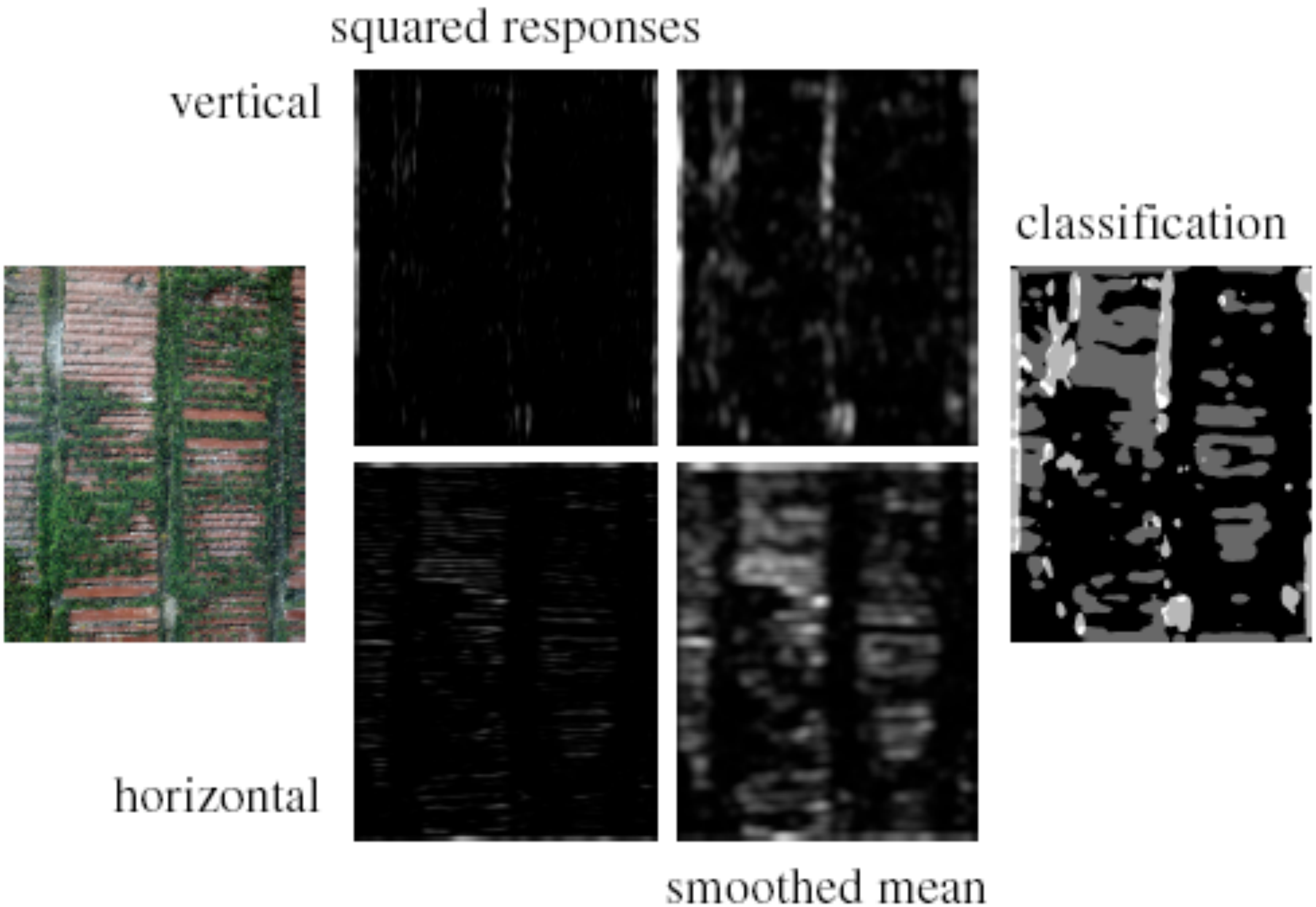

256x256
128x128
64x64
32x32
16x16

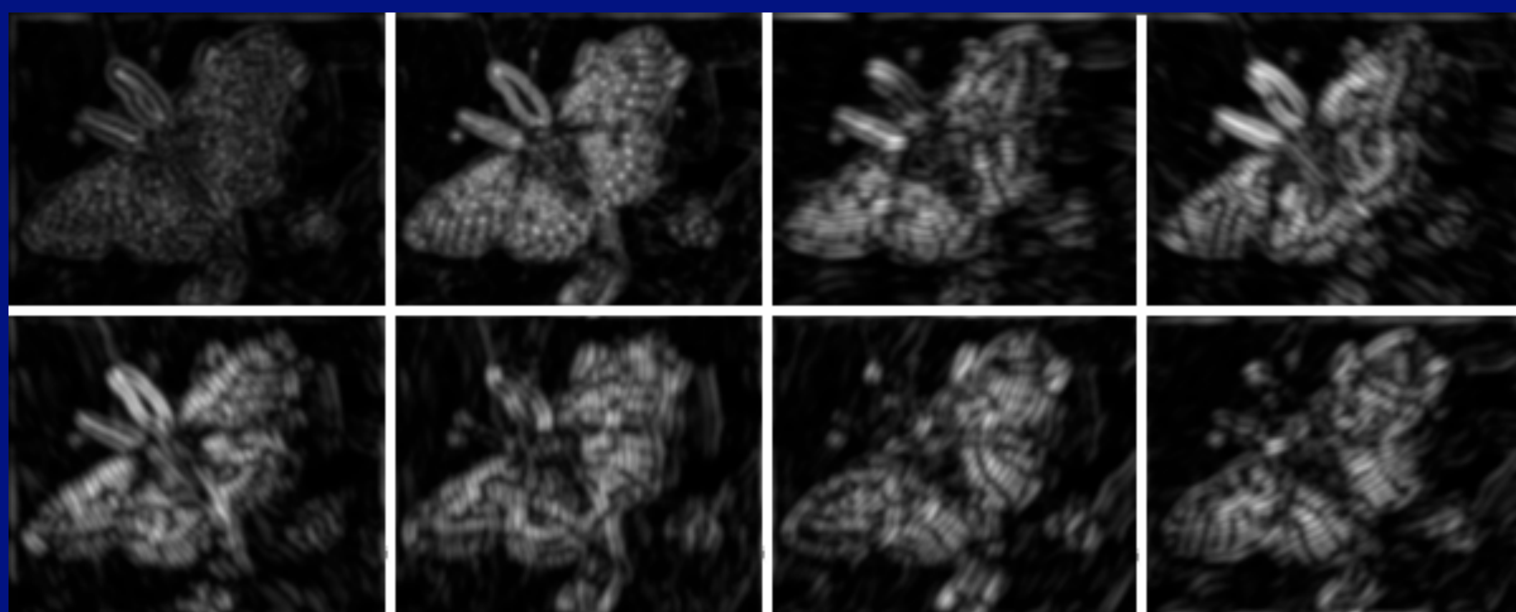

256x256
128x128
64x64
32x32
16x16

Texture

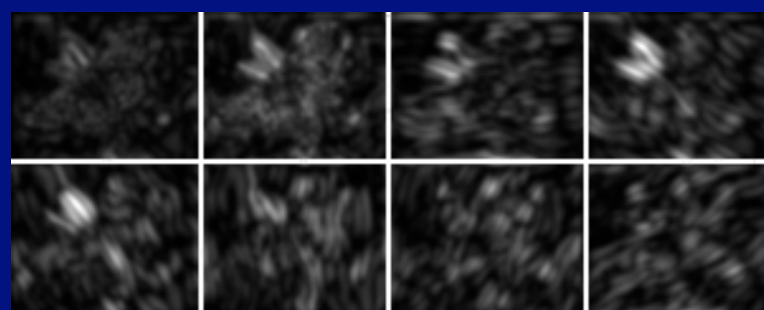
- Key issue: representing texture
  - Texture based matching
    - little is known, key issue seems to be representing texture
  - Texture segmentation
    - key issue: representing texture
  - Texture synthesis
    - useful; also gives some insight into quality of representation
  - Shape from texture
    - cover superficially

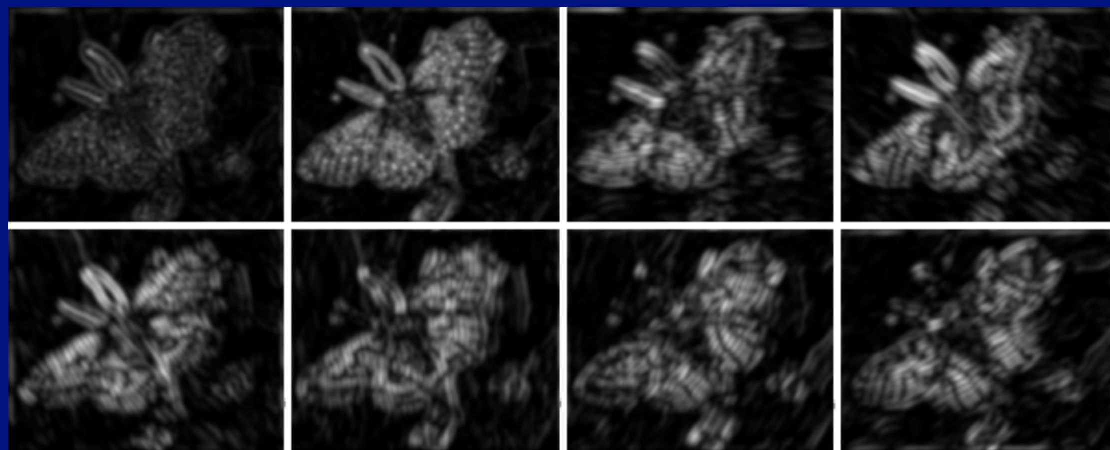
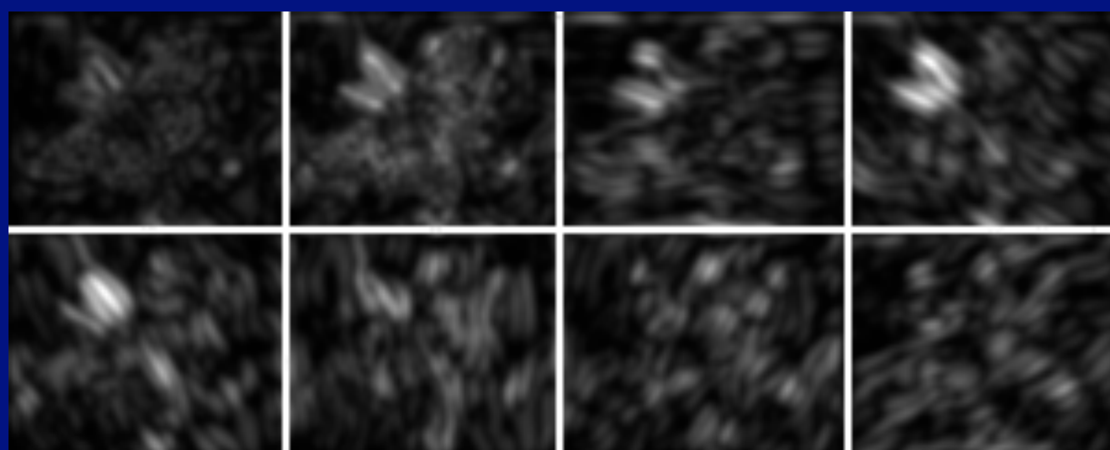

(Tri-arr)
(Plus-ti)
(Ti-ell)
(RR-RL)


squared responses
vertical
classification
horizontal
smoothed mean



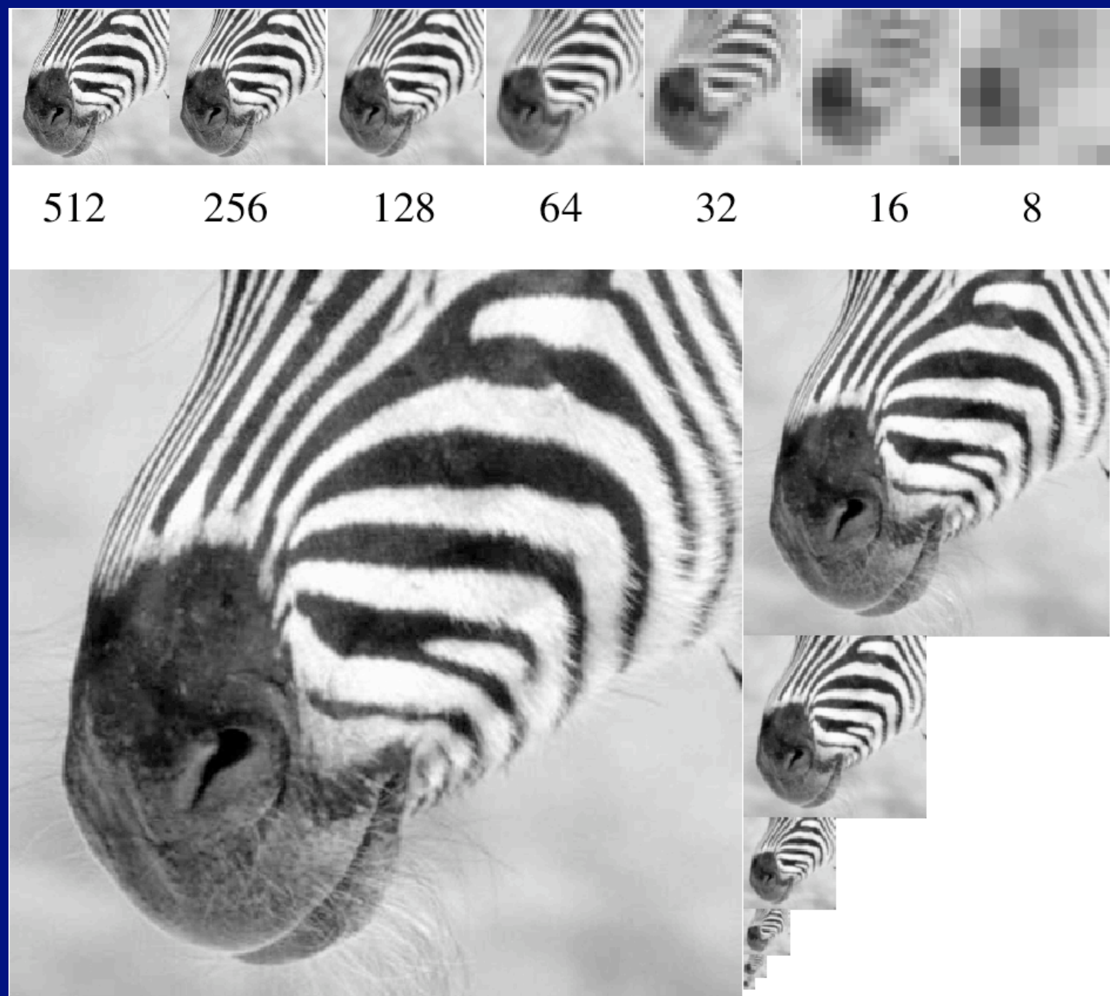


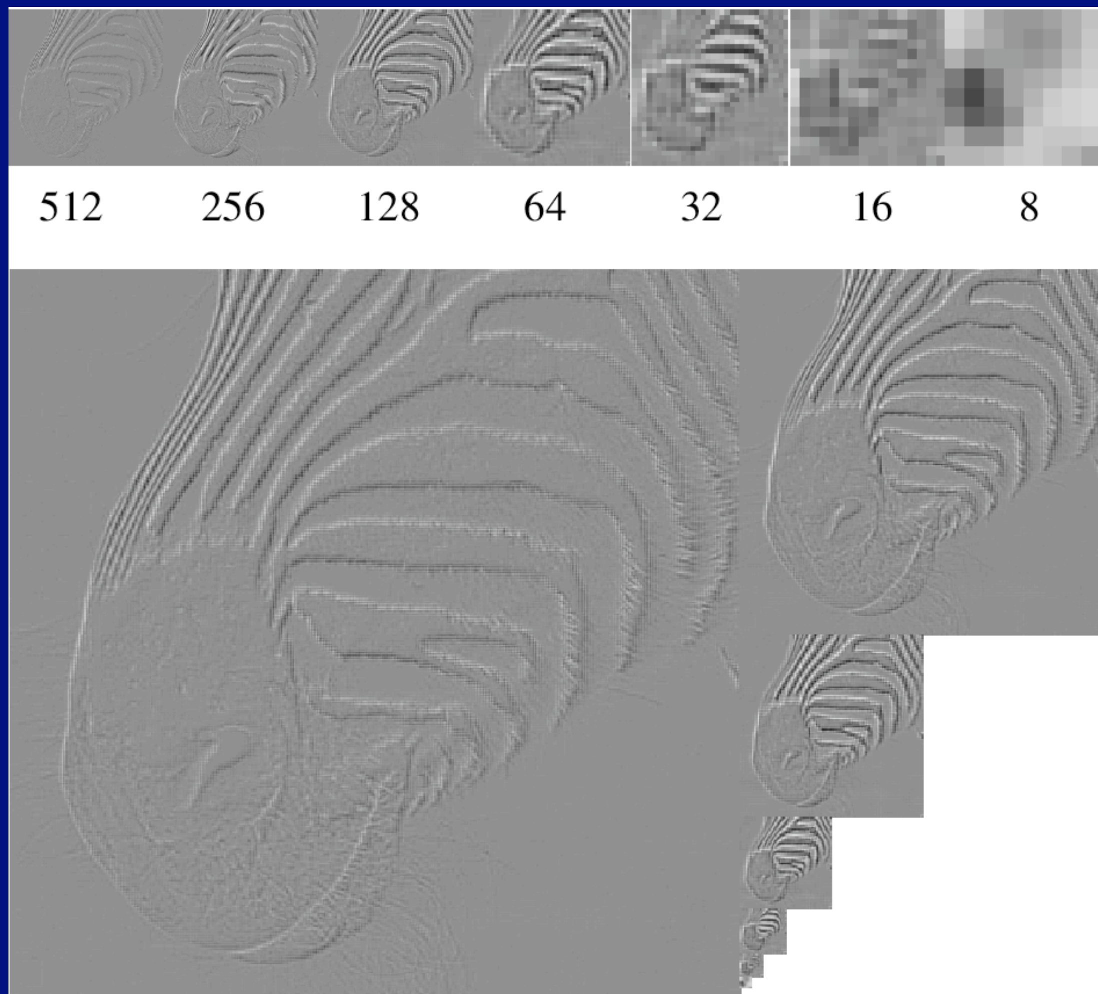


The Laplacian Pyramid

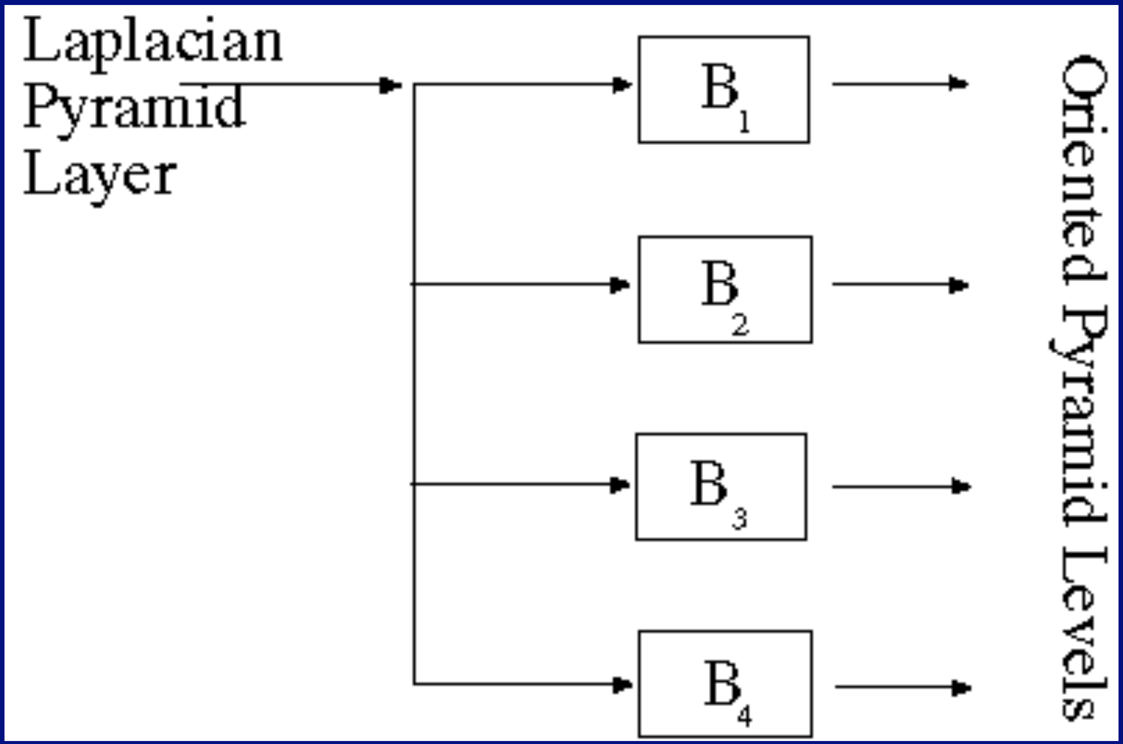
- Synthesis
  - preserve difference between upsampled Gaussian pyramid level and Gaussian pyramid level
  - band pass filter - each level represents spatial frequencies (largely) unrepresented at other levels
- Analysis
  - reconstruct Gaussian pyramid, take top layer


512
256
128
64
32
16
8


512
256
128
64
32
16
8

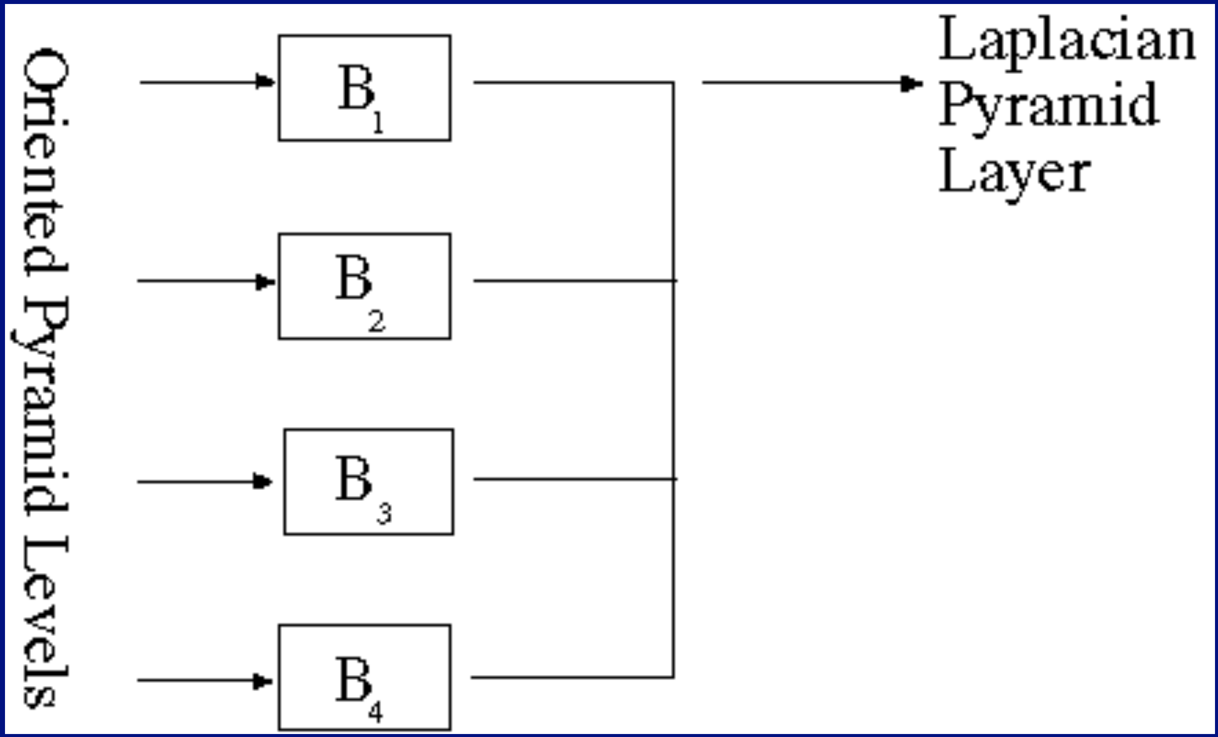
Oriented pyramids

- Laplacian pyramid is orientation independent
- Apply an oriented filter to determine orientations at each layer
  - by clever filter design, we can simplify synthesis
  - this represents image information at a particular scale and orientation

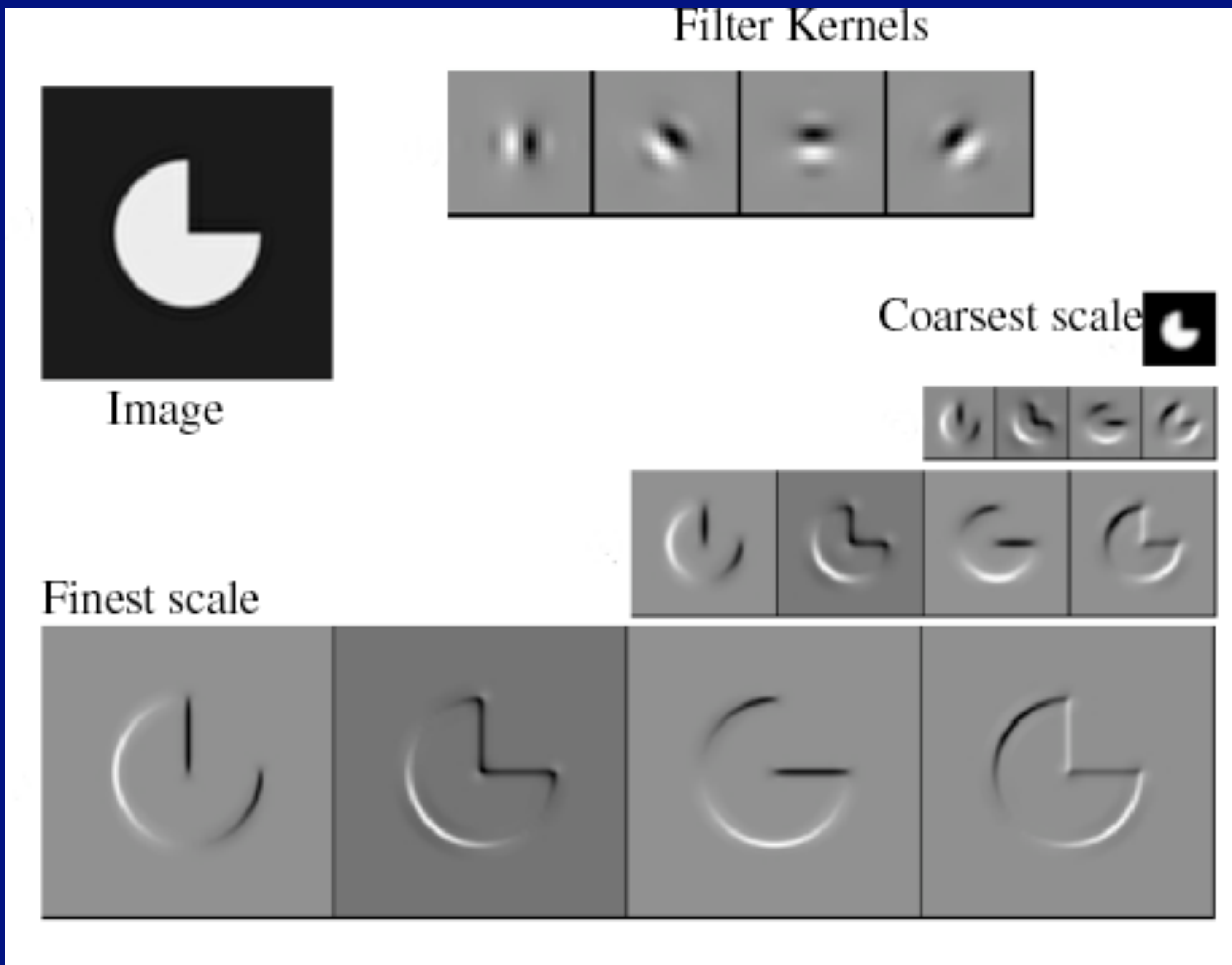

Laplacian
Pyramid
Layer
B_1
B_2
B_3
B_4
Oriented Pyramid Levels

Analysis

synthesis

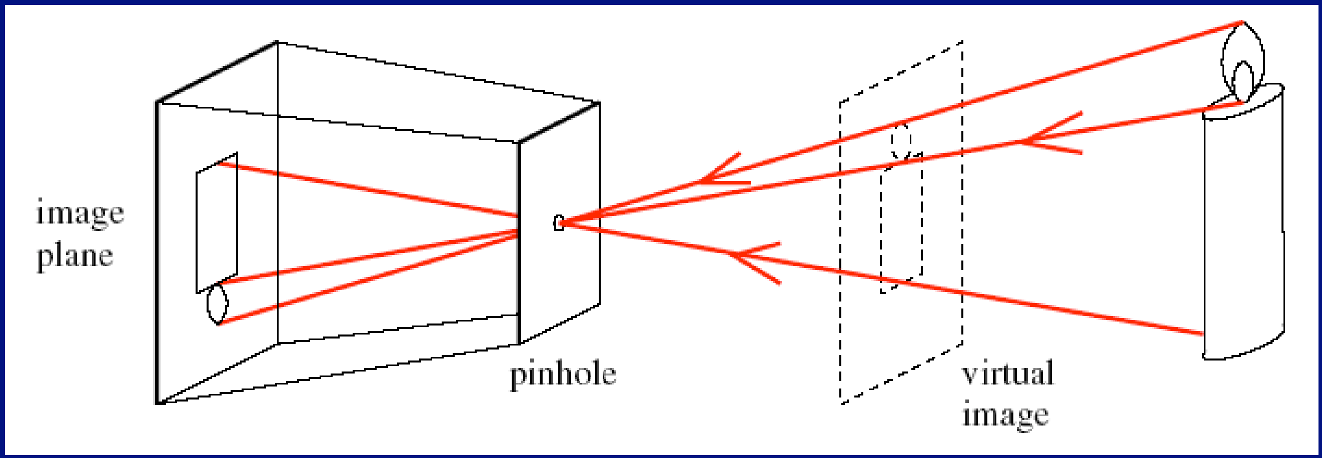


Oriented Pyramid Levels
B1
B2
B3
B4
Laplacian Pyramid Layer

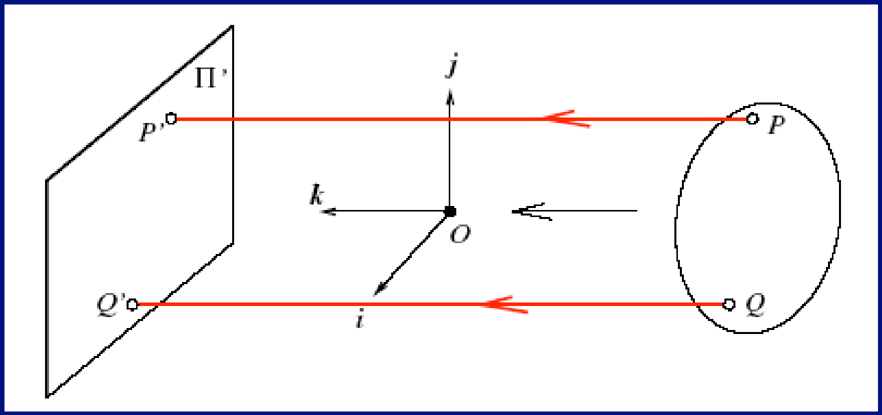

Filter Kernels
Image
Coarsest scale
Finest scale

View variation for a plane patch

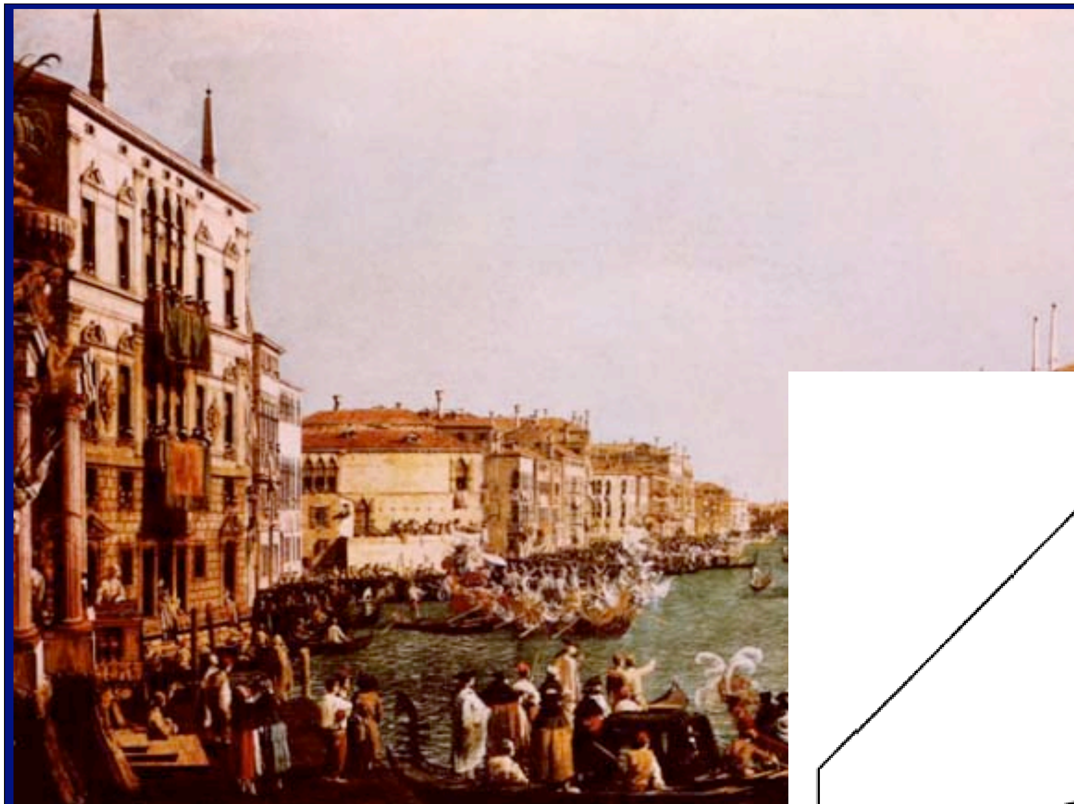
- Plane patches look different in different views



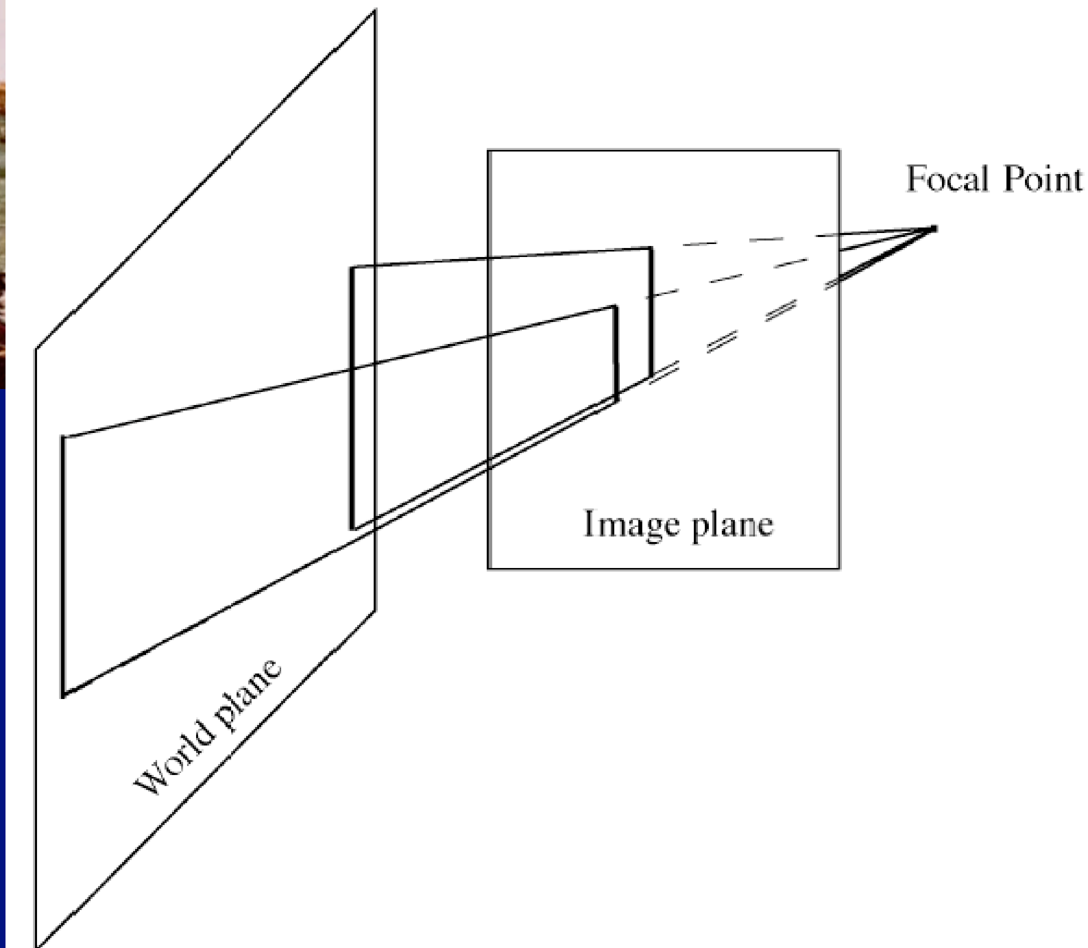
Pinhole camera (F+P, p31)

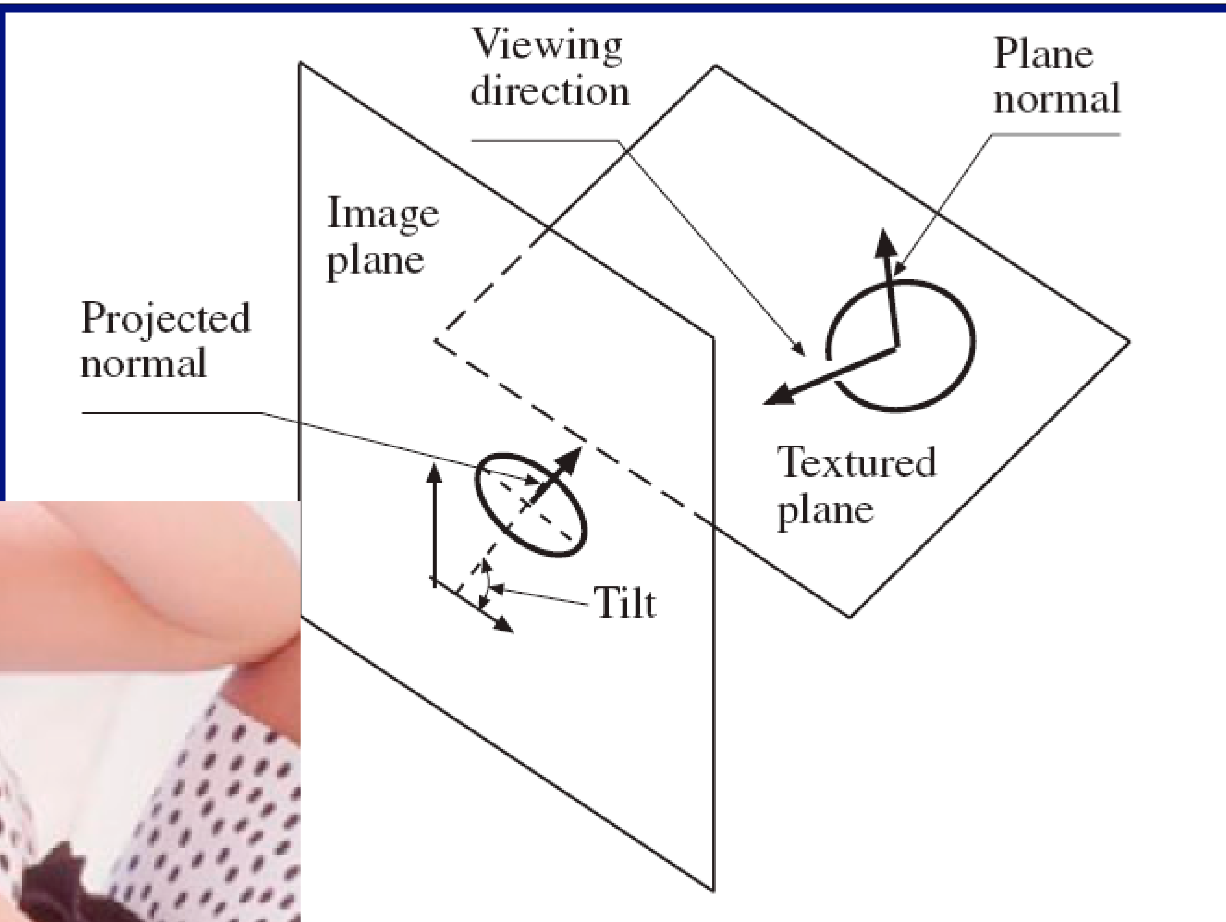


Orthographic camera (F+P, p33)



Views of plane patches in perspective cameras induce homographies





Views of plane patches in orthographic cameras induce a special subset of homographies

Interest points and local descriptions

- Find localizable points in the image
  - e.g. corners - established technology,
    - eg find image windows where there tend to be strong edges going in several different directions
- Build at each point
  - a local, canonical coordinate frame
    - Euclidean+scale
    - Affine
  - Do this by searching for a coordinate frame within which some predicate applies
    - E.g. Rotation frame from orientation of gradients
    - E.g. Rotation + scale orientation of gradients, maximum filter response
  - a representation of the image within that coordinate frame
    - this representation is invariant because frame is covariant

Example: Lowe, 99

- Find localizable points in the image
  - find maxima, minima of response to difference of gaussians
    - over space
    - over scale
    - using pyramid
- Build at each point
  - a Euclidean + scale coordinate frame
    - scale from scale of strongest response
    - rotation from peak of orientation histogram within window
  - Representation
    - SIFT features

Lowe’s SIFT features

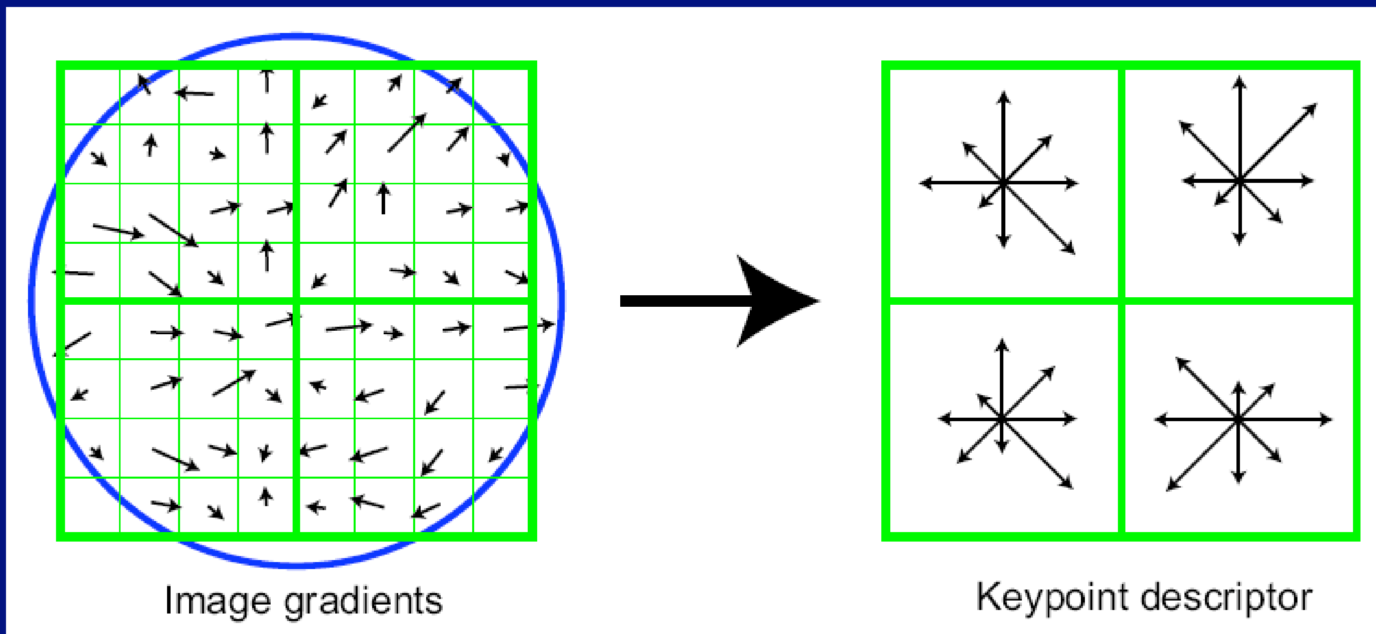
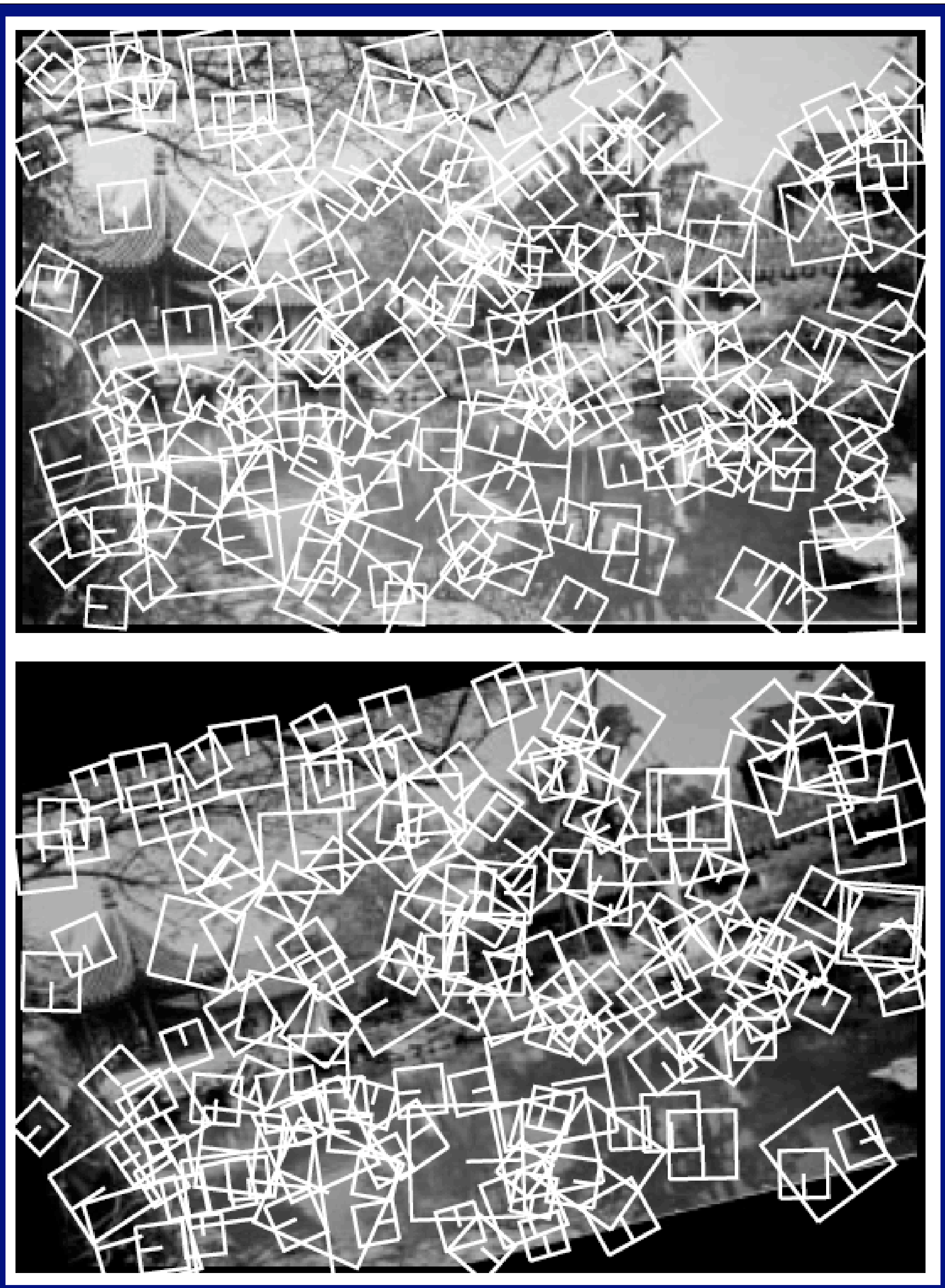
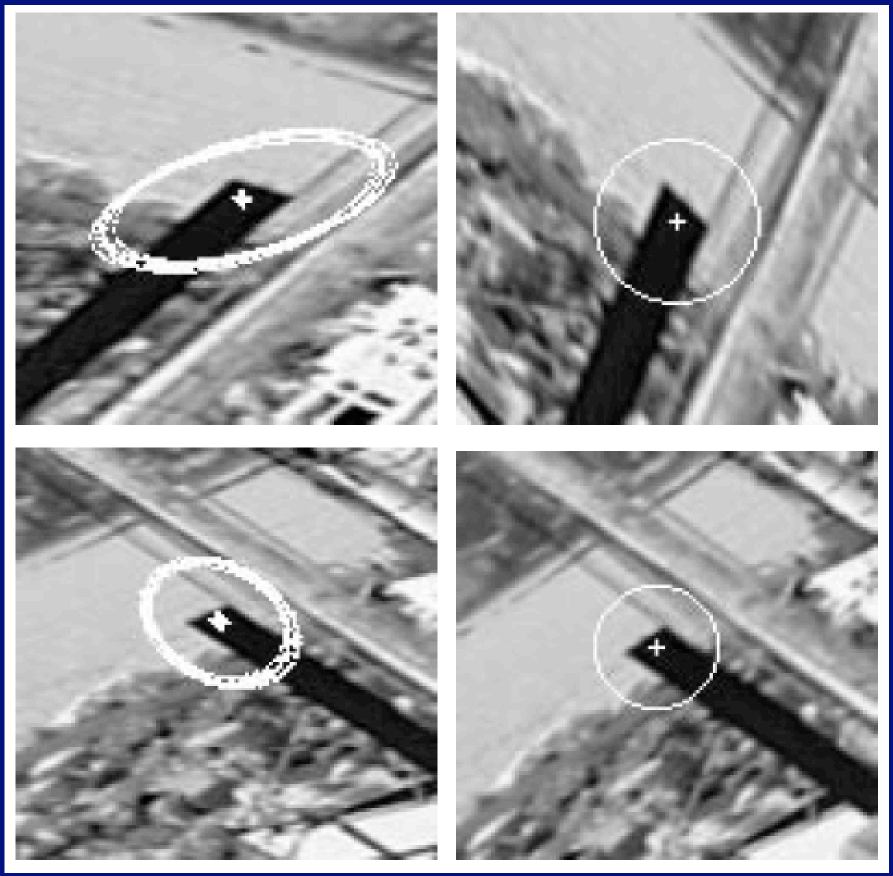


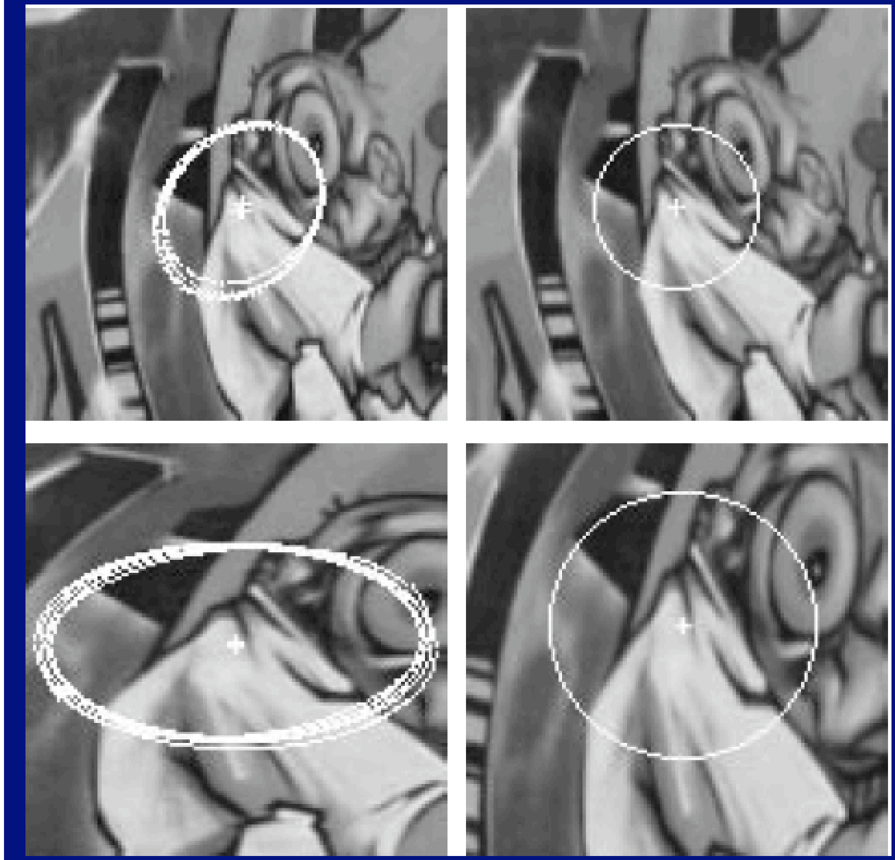
Fig 7 from:
Distinctive image features from scale-invariant keypoints
David G. Lowe, *International Journal of Computer Vision,* 60, 2 (2004), pp. 91-110.



From Lowe, 99, **Object Recognition from Local Scale-Invariant Features**

Mikolaczyk/Schmid coordinate frames





Matching objects with point features

- Voting
  - each point feature votes for every object that contains it
  - object with most votes wins
  - Startlingly effective (see figures)

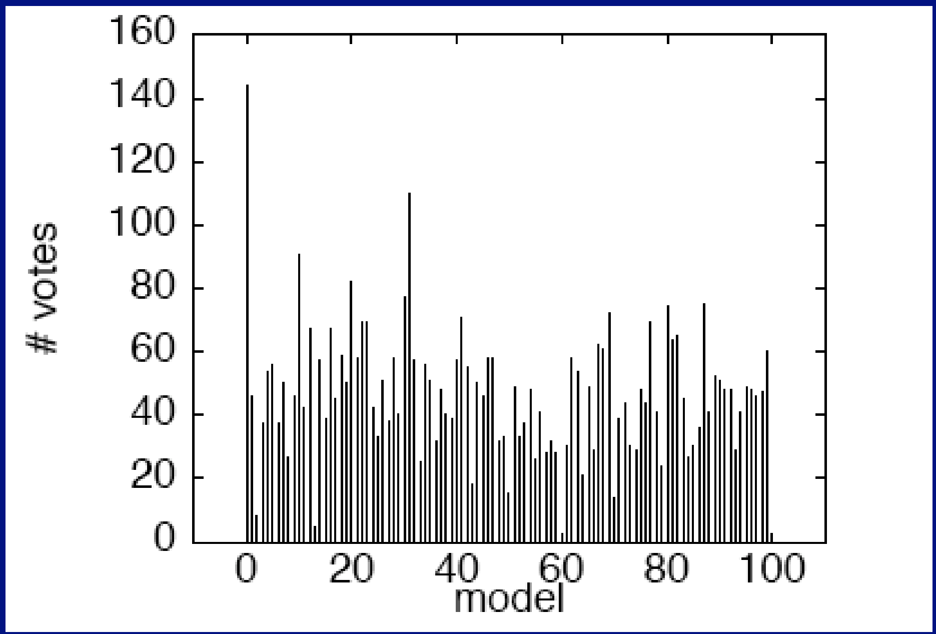


Figure from "Local grayvalue invariants for image retrieval," by C. Schmid and R. Mohr, IEEE Trans. Pattern Analysis and Machine Intelligence, 1997 c 1997, IEEE as used in Forsyth + Ponce, p 609

Probabilistic interpretation

- Write

$$P\{\text{patch of type } i \text{ appears in image}|j\text{'th pattern is present}\} = p_{ij}$$

$$P\{\text{patch of type } i|\text{no pattern is present}\} = p_{ix}$$

- Assume

$p_{ij} = \mu$ if the pattern can produce this patch and 0 otherwise

$p_{ix} = \lambda < \mu$ for all i.

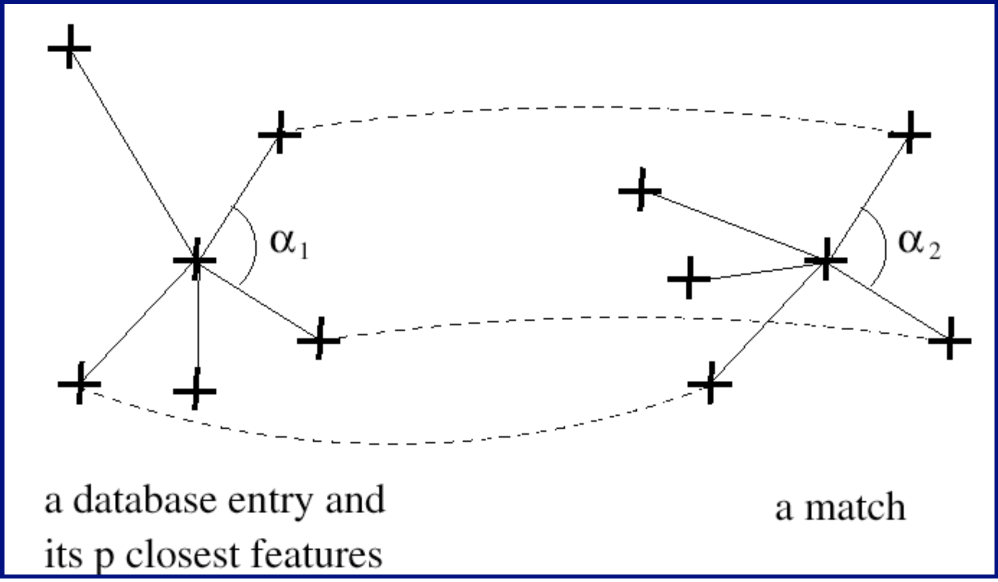
- Likelihood of image given pattern

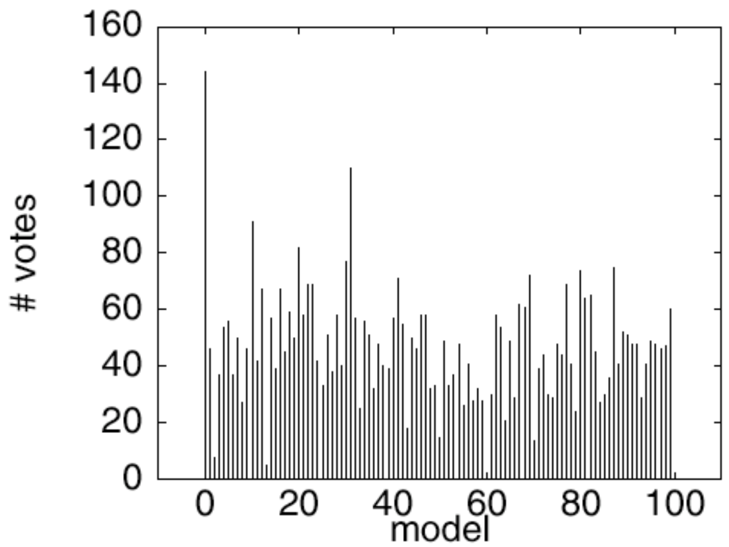
that n_p patches came from that pattern and $n_i - n_p$ patches come from noise, is

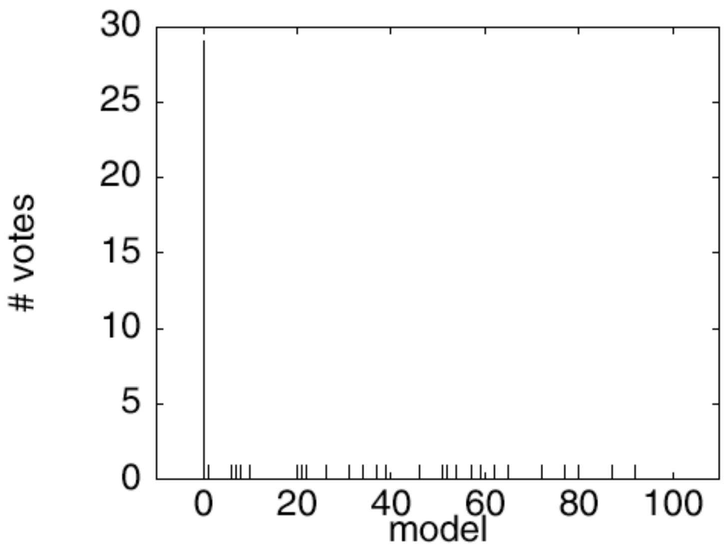
$$P(\text{interpretation}|\text{pattern}) = \lambda^{n_p}\mu^{(n_i - n_p)}$$

Employ spatial relations

Figure from "Local grayvalue invariants for image retrieval," by C. Schmid and R. Mohr, IEEE Trans. Pattern Analysis and Machine Intelligence, 1997 c 1997, IEEE as used in Forsyth + Ponce, p 609







Possible alternative strategies

- Notice:
  - different patterns may yield different templates with different probabilities
  - different templates may be found in noise with different probabilities

Pose consistency

- A match between an image structure and an object structure implies a pose
  - we can vote on poses, objects

From Lowe, 99, **Object Recognition from Local Scale-Invariant Features**

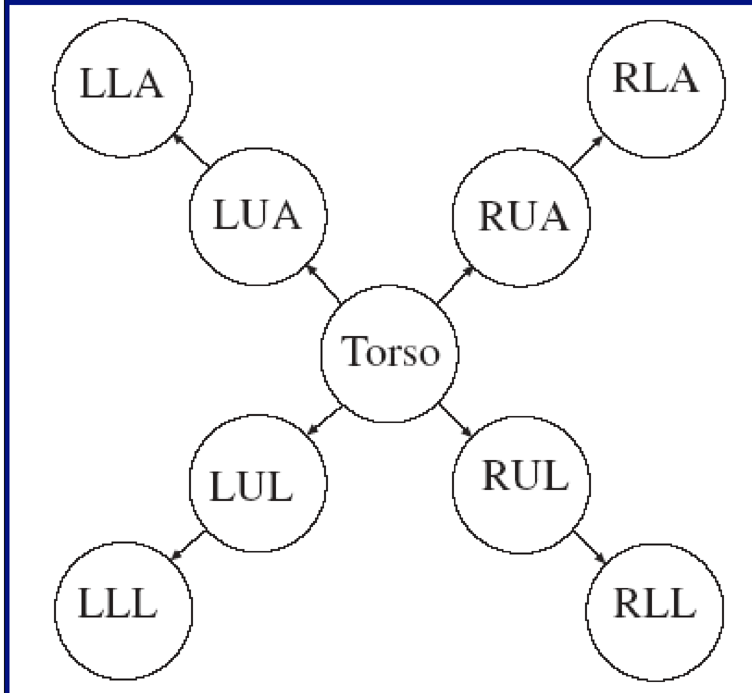
From Lowe, 99, **Object Recognition from Local Scale-Invariant Features**

Kinematic grouping

- Assemble a set of features to present to a classifier
  - which tests
    - appearance
    - configuration
    - whatever
- Classifier could be
  - handwritten rules (e.g. Fleck-Forsyth-Bregler 96)
  - learned classifier (e.g. Ioffe-Forsyth 99)
  - likelihood (e.g. Felzenszwalb-Huttenlocher 00)
  - likelihood ratio test (e.g. Leung-Burl-Perona 95; Fergus-Perona-Zisserman 03)

Pictorial structures

- For models with the right form, one can test “everything”
  - model is a set of cylindrical segments linked into a tree structure
    - model should be thought of as a 2D template
      - segments are cylinders, so no aspect issue there
      - 3D segment kinematics implicitly encoded in 2D relations
      - easy to build in occlusion
  - putative image segments are quantized
  - => dynamic programming to search all matches
  - What to add next? (DP deals with this)
  - Pruning? (Irrelevant)
  - Can one stop?
    - (Use a mixture of tree models, with missing segments marginalized out)
  - Known segment colour - Felzenszwalb-Huttenlocher 00
  - Learned models of colour, layout, texture - Ramanan Forsyth 03, 04



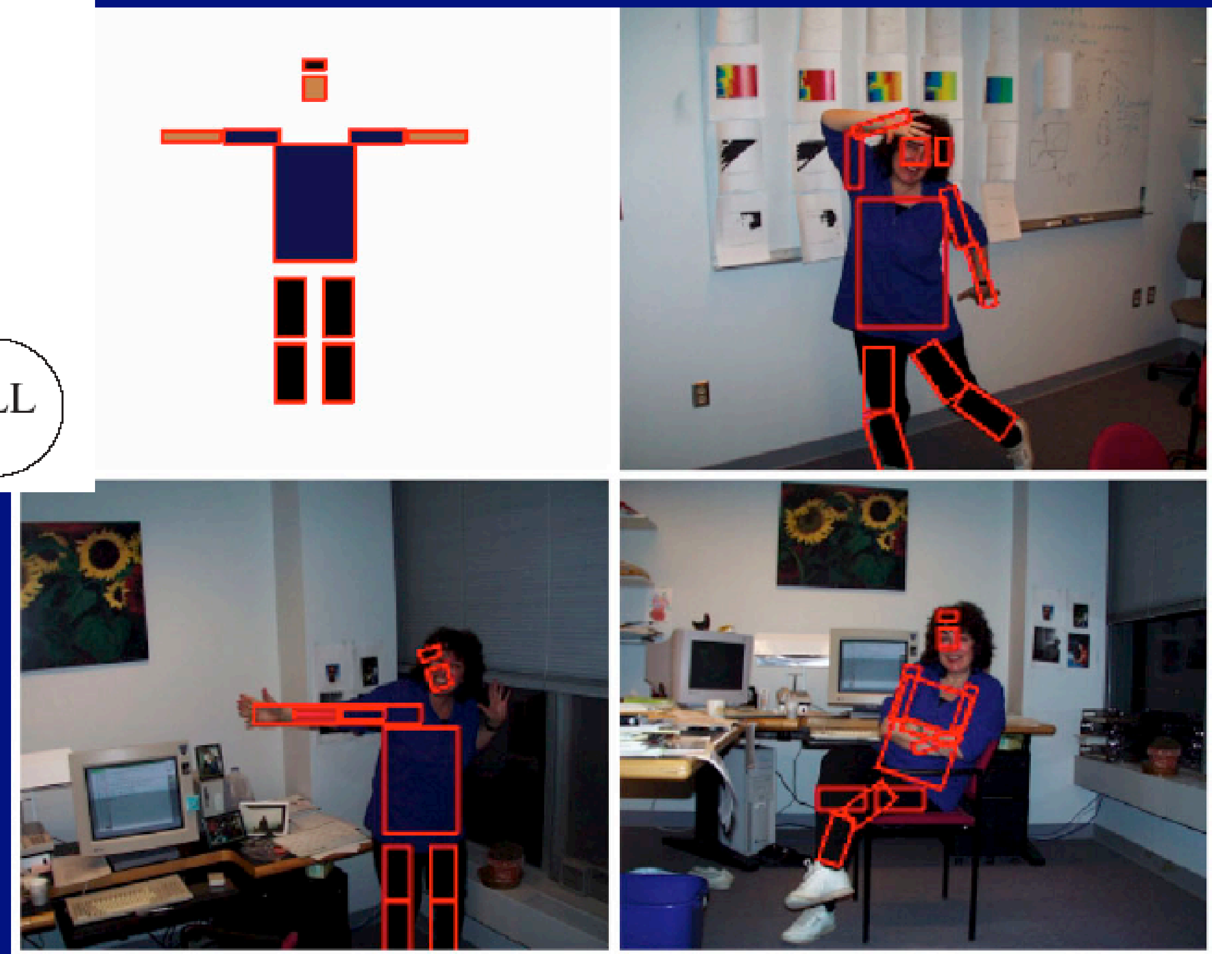


Figure from “Efficient Matching of Pictorial Structures,”
P. Felzenszwalb and D.P. Huttenlocher, Proc. Computer Vision and Pattern Recognition
2000, c 2000, IEEE as used in Forsyth+Ponce, pp 636, 640

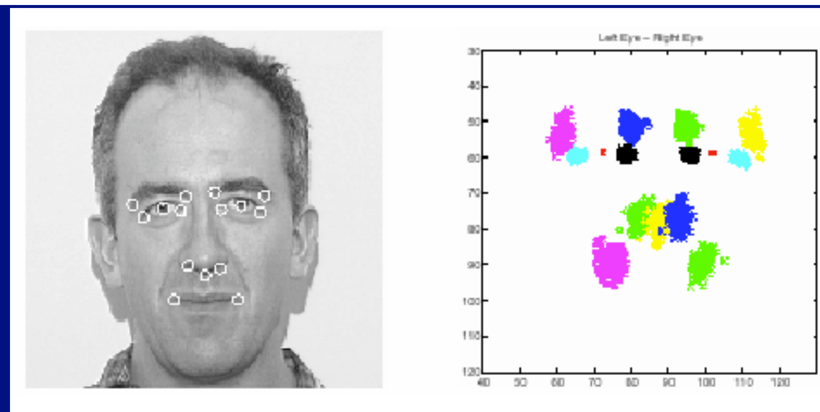
Finding faces using relations

- Strategy: compare

$$P(\text{one face at } \boldsymbol{F} | \boldsymbol{X}_{\text{le}} = \boldsymbol{x}_1, \boldsymbol{X}_{\text{re}} = \boldsymbol{x}_2, \boldsymbol{X}_{\text{m}} = \boldsymbol{x}_3, \boldsymbol{X}_{\text{n}} = \boldsymbol{x}_4, \text{all other responses})$$

with

$$P(\text{no face} | \boldsymbol{X}_{\text{le}} = \boldsymbol{x}_1, \boldsymbol{X}_{\text{re}} = \boldsymbol{x}_2, \boldsymbol{X}_{\text{m}} = \boldsymbol{x}_3, \boldsymbol{X}_{\text{n}} = \boldsymbol{x}_4, \text{all other responses})$$



Detection

$$P(\text{one face at } \boldsymbol{F} | \boldsymbol{X}_{\text{le}} = \boldsymbol{x}_1, \boldsymbol{X}_{\text{re}} = \boldsymbol{x}_2, \boldsymbol{X}_{\text{m}} = \boldsymbol{x}_3, \boldsymbol{X}_{\text{n}} = \boldsymbol{x}_4, \text{all other responses}) =$$
$$P(\text{one face at } \boldsymbol{F} | \boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4) P(\text{all other responses}) \propto$$
$$P(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4 | \text{one face at } \boldsymbol{F}) P(\text{all other responses}) P(\text{one face at } \boldsymbol{F})$$

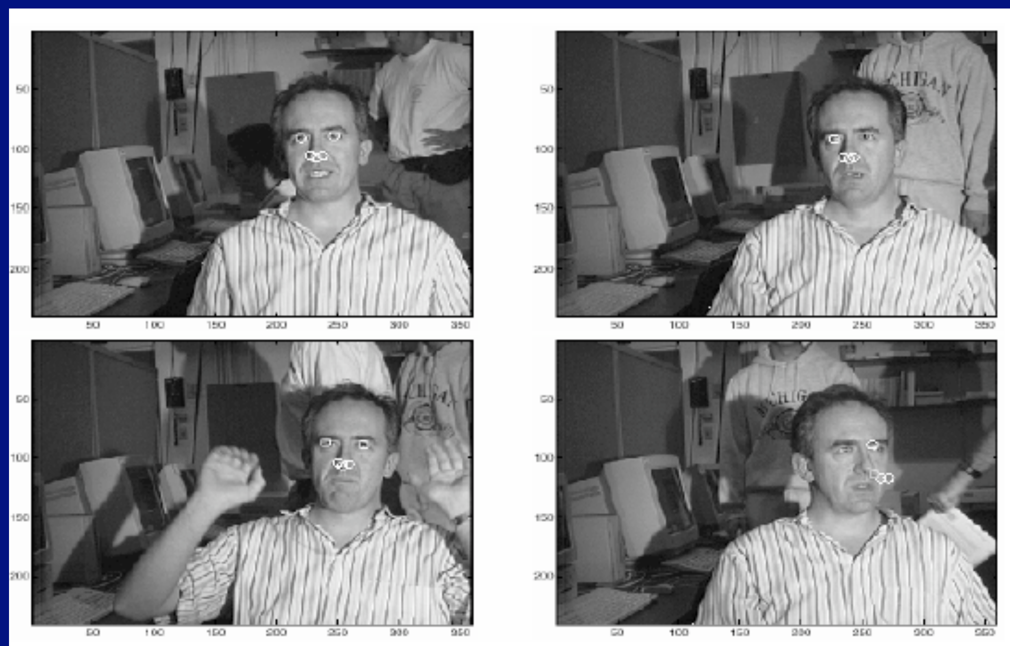
This means we compare

$$P(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4 | \text{one face at } \boldsymbol{F})$$

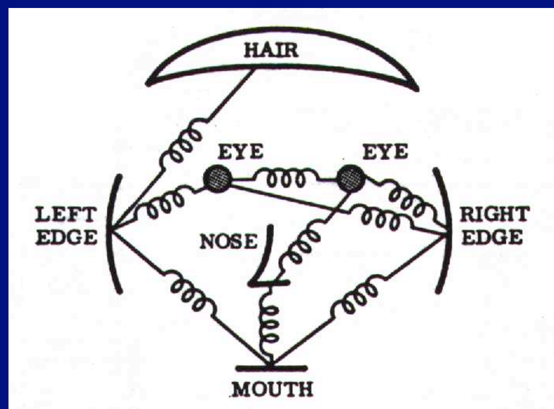
with

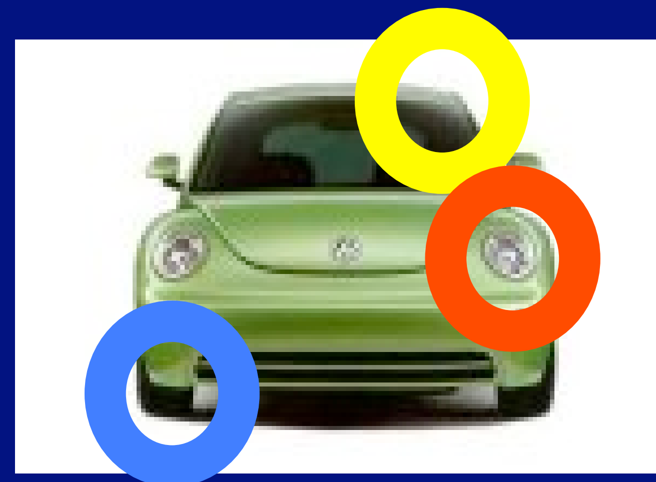
$$(P(\text{noise responses}) P(\text{no face}) / P(\text{one face at } \boldsymbol{F})) \, (\text{term in relative loss})$$

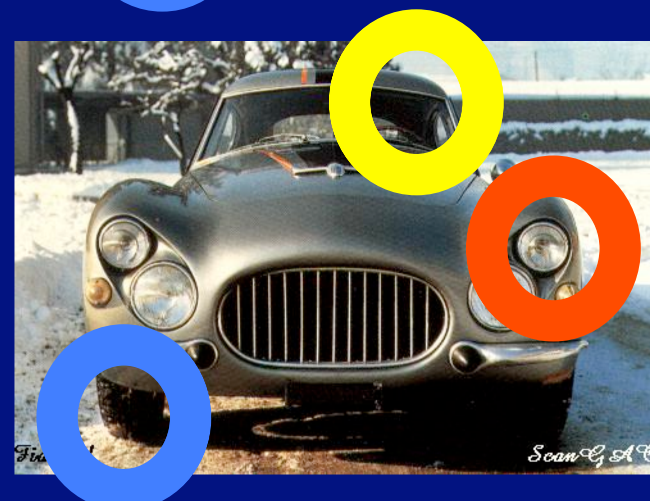

50
100
150
200
50
100
150
200
250
300
350



Constellations of parts







Fischler & Elschlager 1973

Yuille '91
Brunelli & Poggio '93
Lades, v.d. Malsburg et al. '93
Cootes, Lanitis, Taylor et al. '95
Amit & Geman '95, '99
Perona et al. '95, '96, '98, '00
Agarwal & Roth '02

Generative model for plane templates (Constellation model)

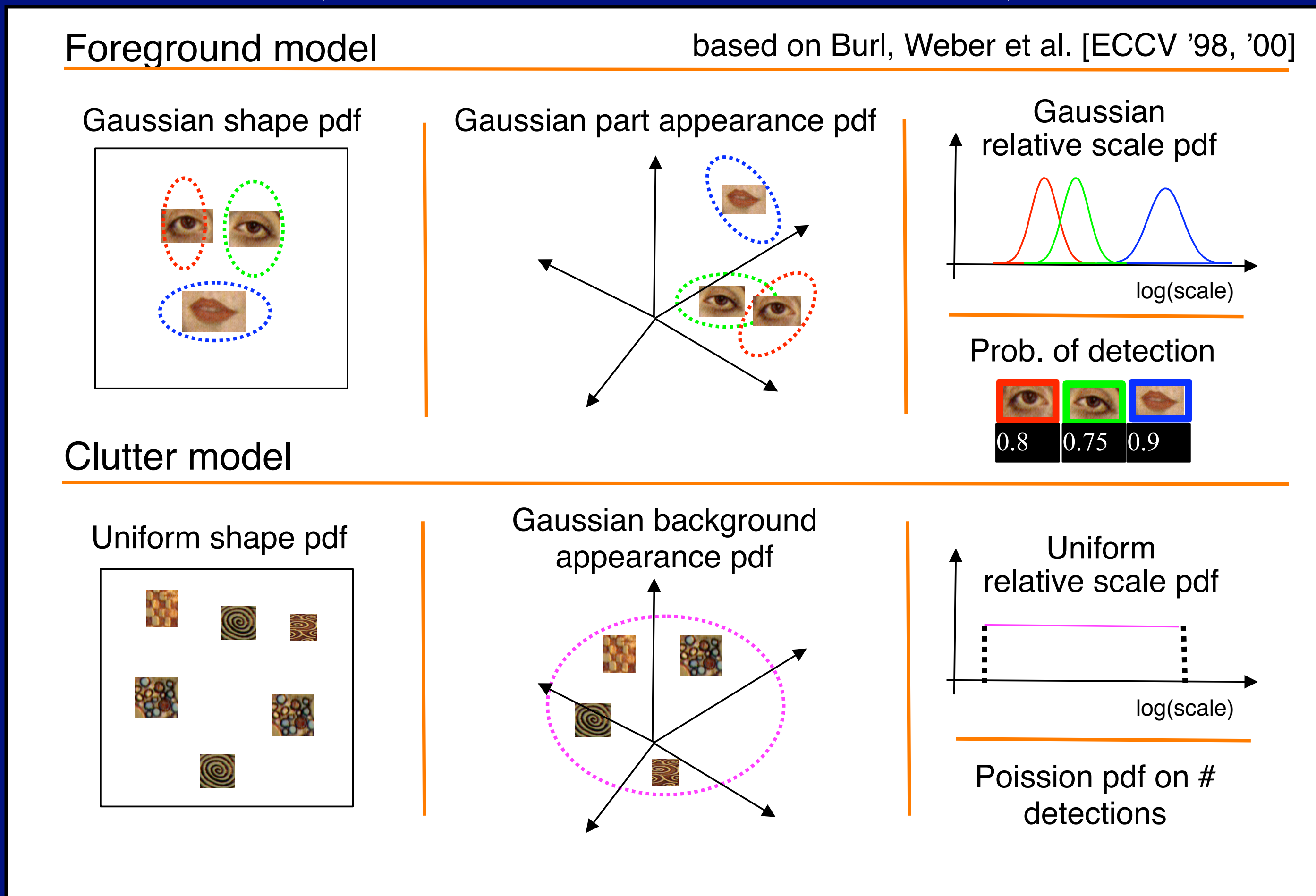
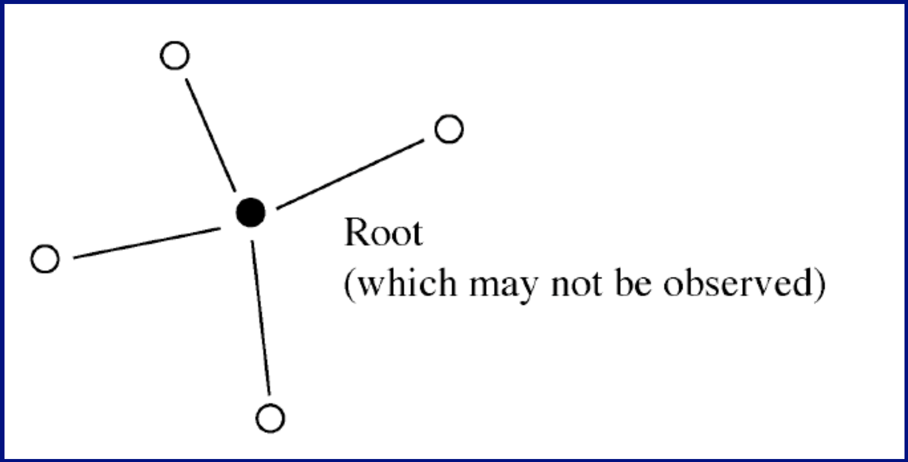


Figure after Fergus et al, 03; see also Fergus et al, 04

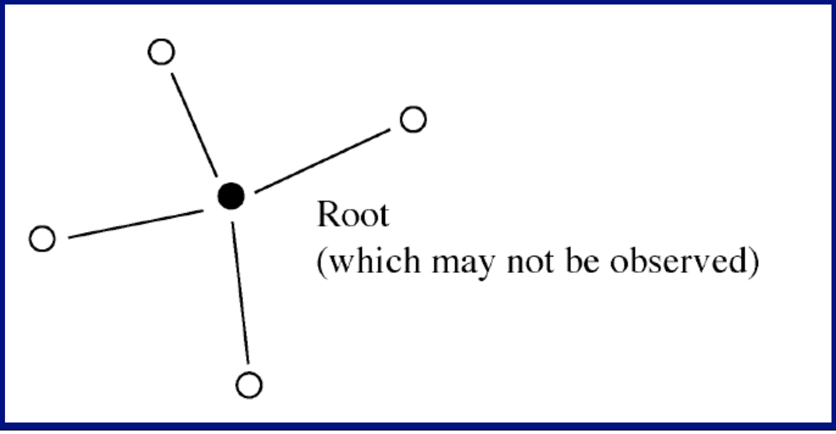
Star-shaped models

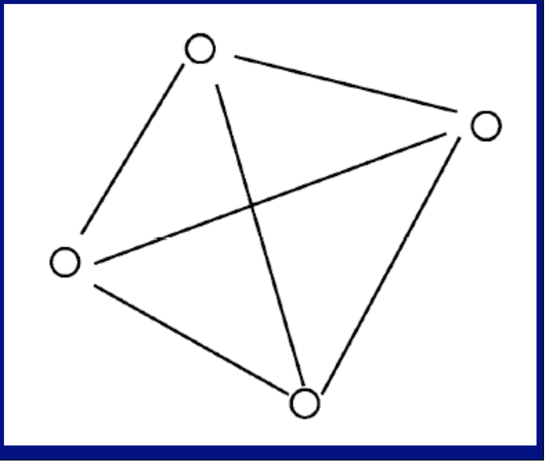
- Features generated at parts
  - at image points that are conditionally independent given part location
  - with appearance that is conditionally independent given part type
- Part locations are conditioned on root
- Easy to deal with
  - very like a pictorial structure
  - inference is dynamic programming
  - localization easy
  -



Other types of model

- We've already seen a tree-structured model!
  - (pictorial structure)
- Complete models are much more difficult to work with
  - because there is no conditional independence
  - means fewer features





Constellation models

- Learning model
  - on data set consisting of instances, not manually segmented
  - choose number of features in model
  - run point feature detector
  - each response is from either one “slot” in the model, or bg
    - this known, easy to estimate parameters
    - parameters known, this is easy to estimate
  - missing variable problem -> EM
- Detecting instance
  - search for allocation of feature instances to slots that maximizes likelihood ratio
  - detect with likelihood ratio test

Typical models

Motorbikes

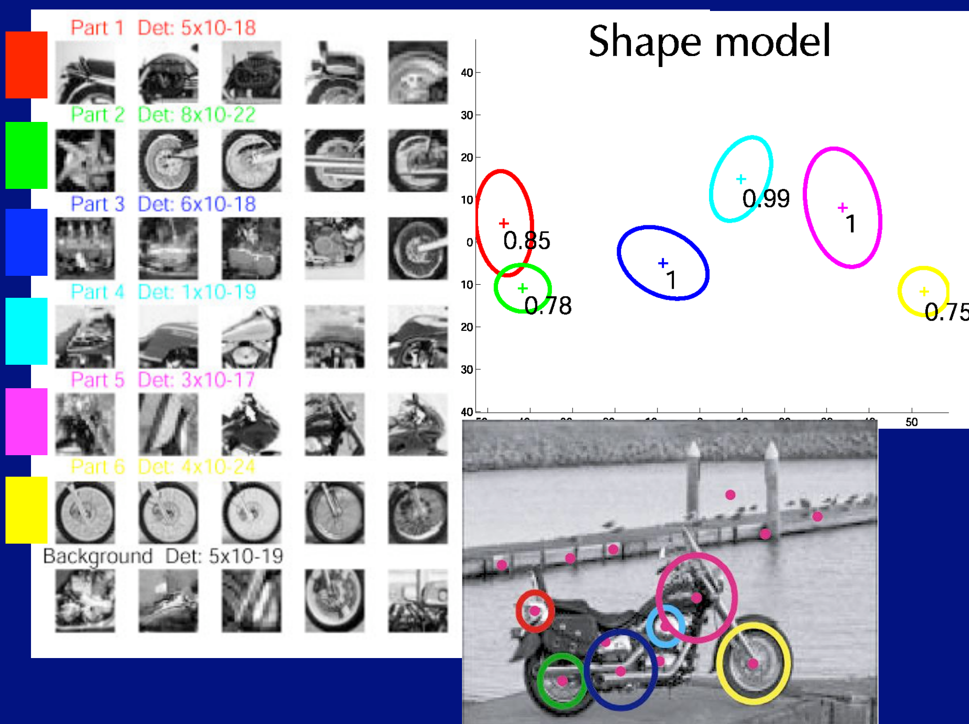
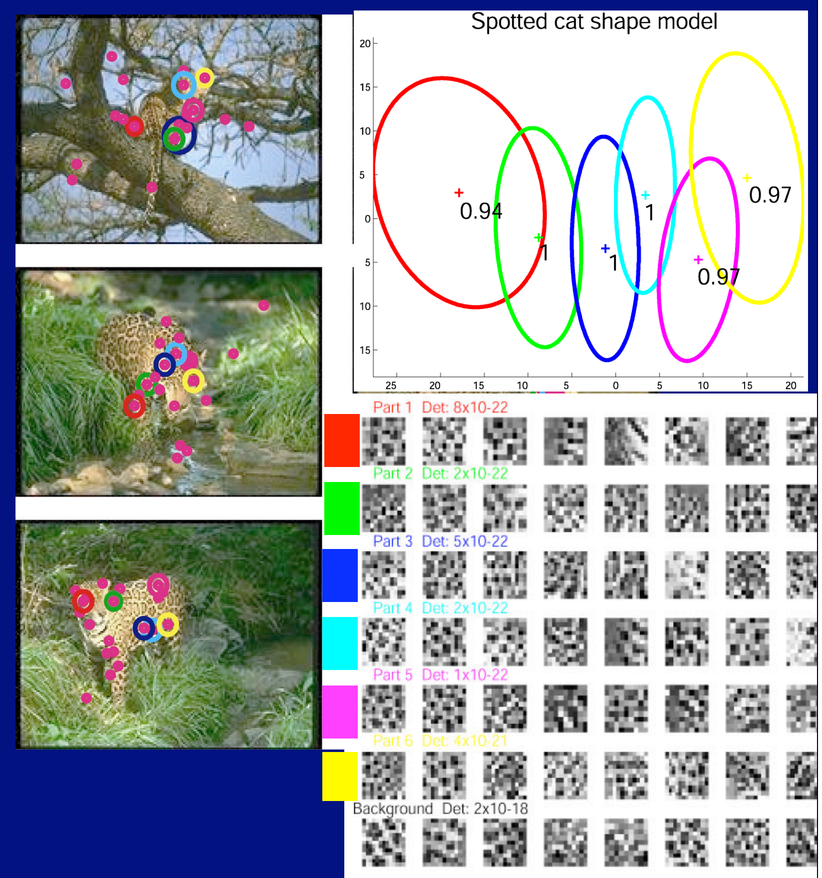


Figure after Fergus et al, 03; see also Fergus et al, 04

Spotted cats



Summary of results

Dataset	Fixed scale experiment	Scale invariant experiment
Motorbikes	7.5	6.7
Faces	4.6	4.6
Airplanes	9.8	7.0
Cars (Rear)	15.2	9.7
Spotted cats	10.0	10.0

% equal error rate

Note: Within each series, same settings used for all datasets

Figure after Fergus et al, 03; see also Fergus et al, 04

Caution: dataset is known to have some quirky features