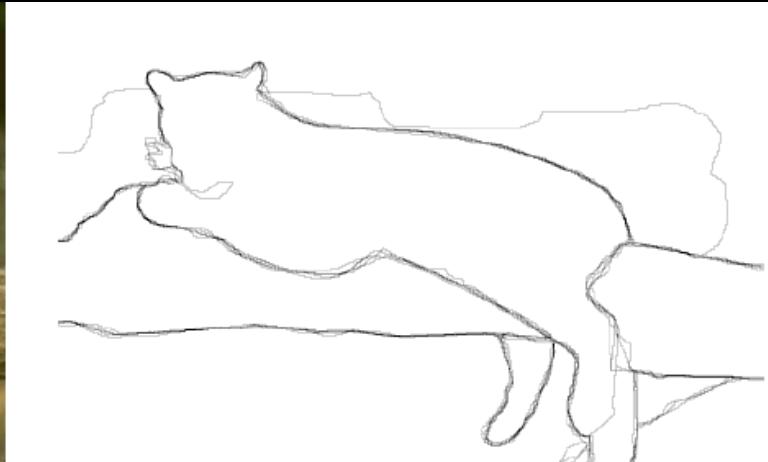


Machine Learning for Computer Vision

30 September 2013

MVA – ENS Cachan



Lecture 1: Introduction to Classification

Iasonas Kokkinos

iasonas.kokkinos@ecp.fr

Center for Computational Vision / Galen Group
Ecole Centrale Paris / INRIA-Saclay



Lecture outline

Introduction to the class

Introduction to the problem of classification

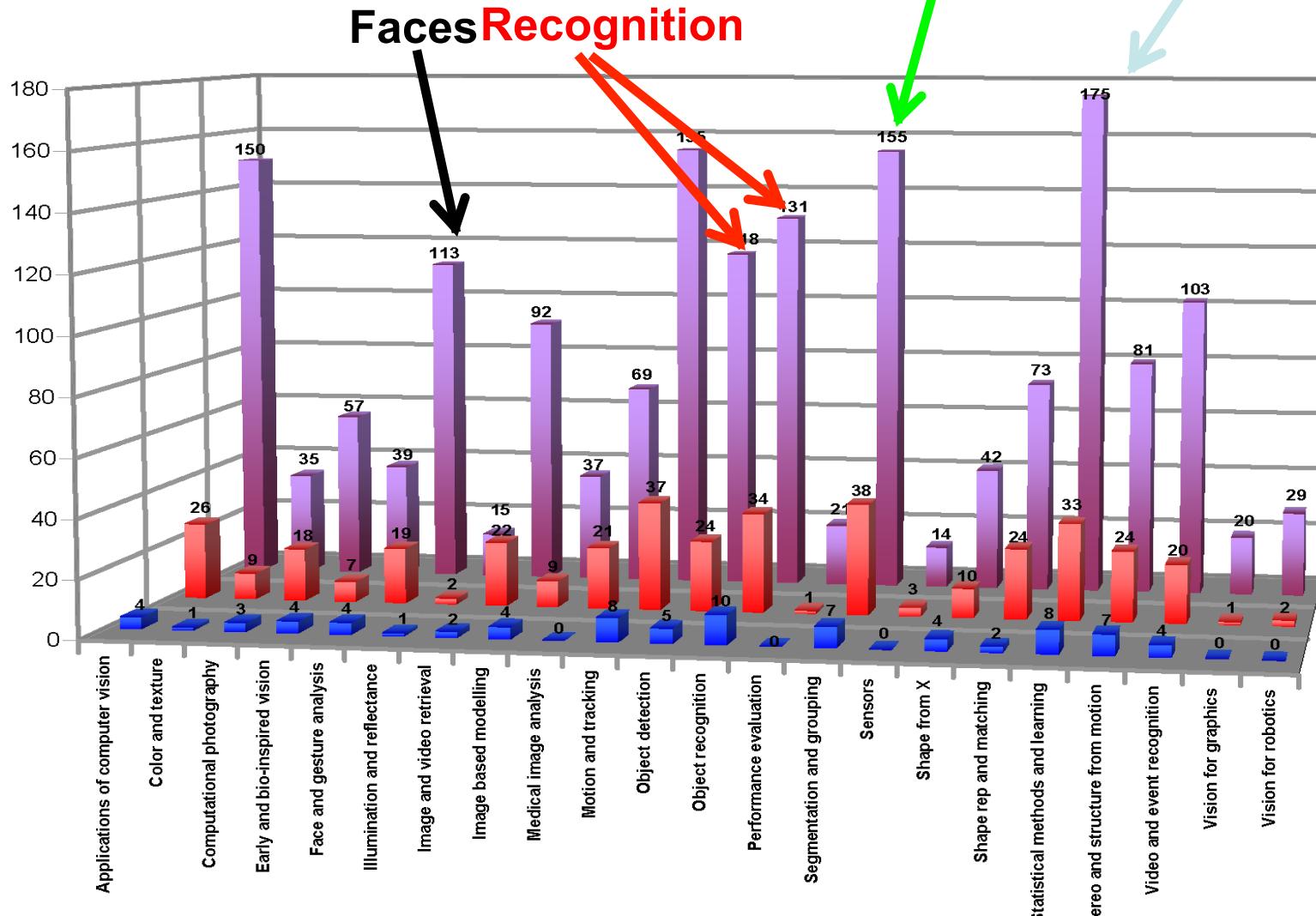
Linear classifiers

Image-based features

Class objectives

- Treatment of a broad range of learning techniques.
- Hands-on experience through computer vision applications.
- By the end: you should be able to understand and implement a paper lying at the interface of vision and learning.

Who will need this class?



Submission/Acceptance Statistics from CVPR 2010

Boundary detection problem

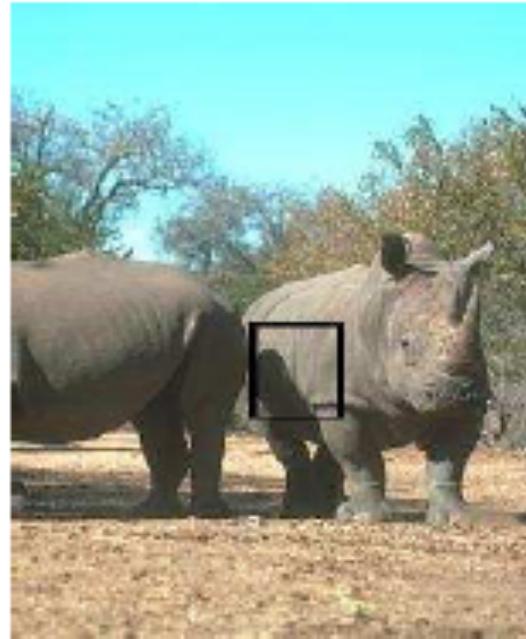
Object/Surface Boundaries



Signal-level challenges



Poor contrast

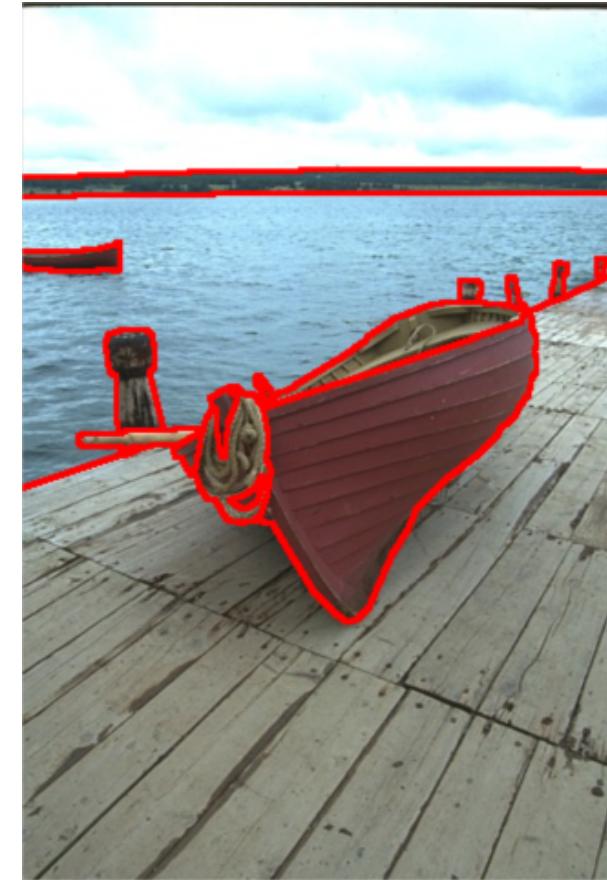
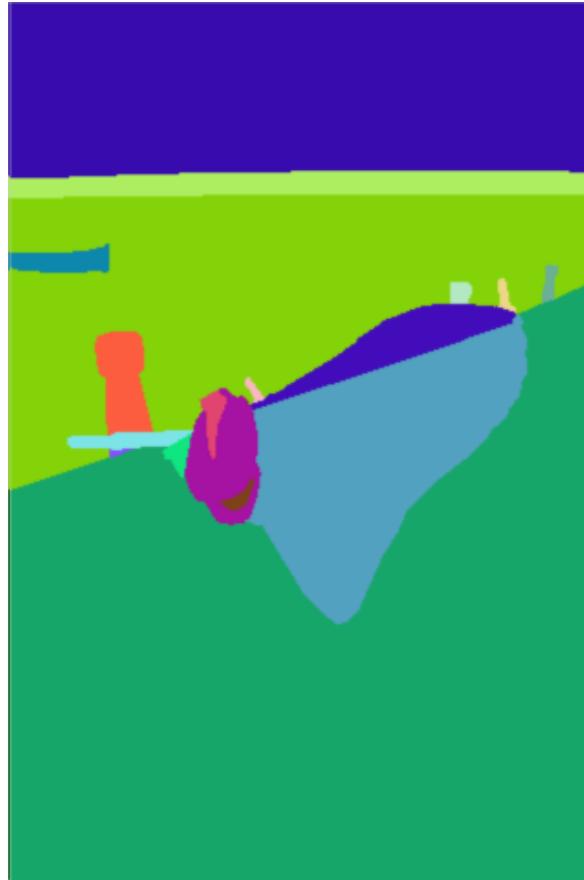


Shadows

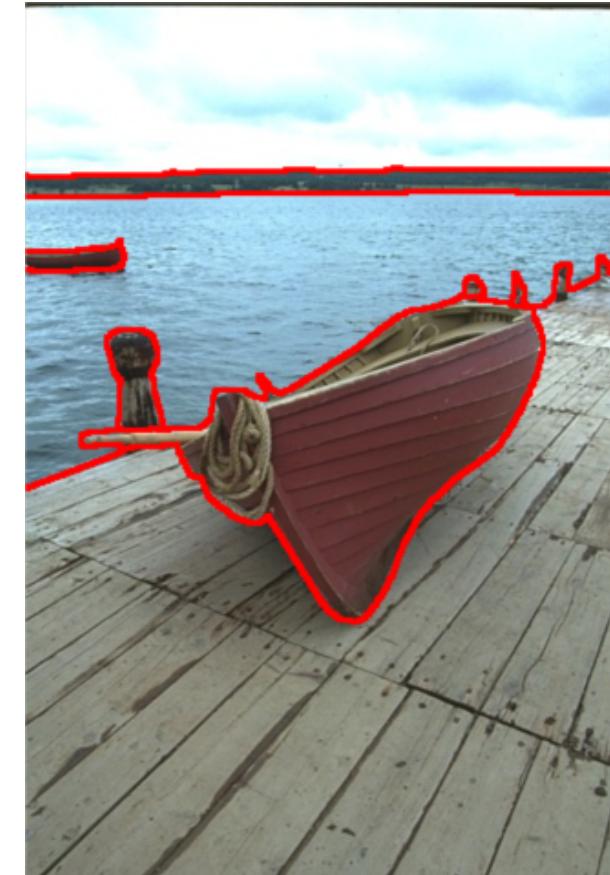
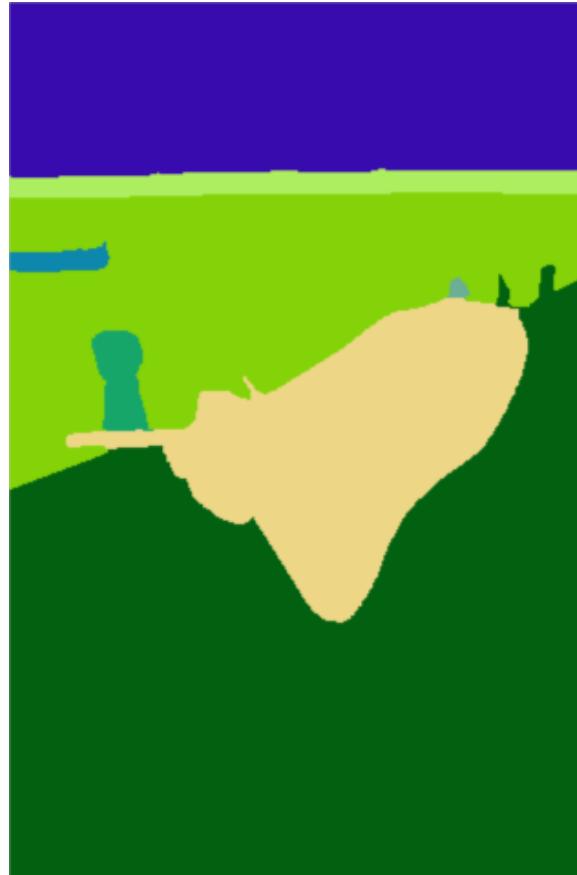


Texture

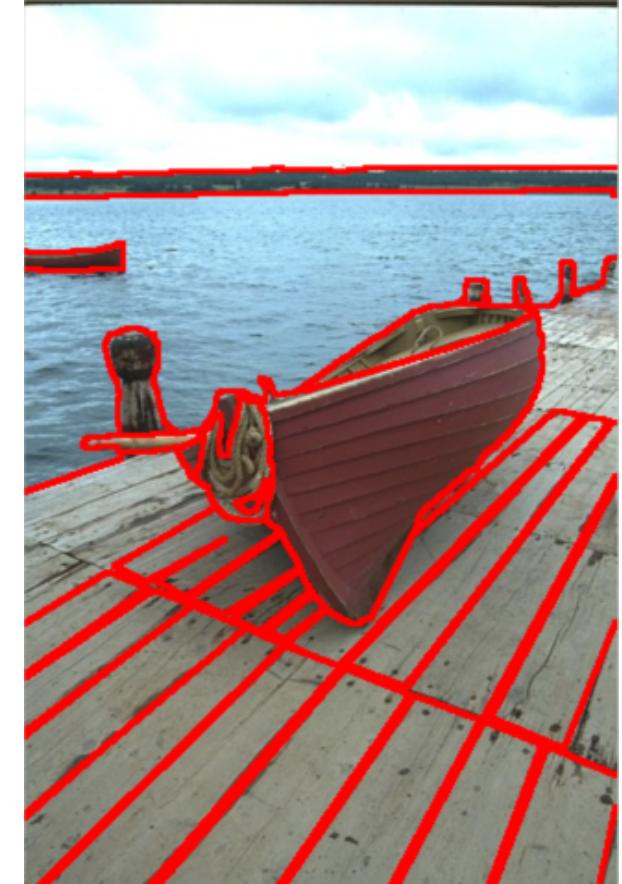
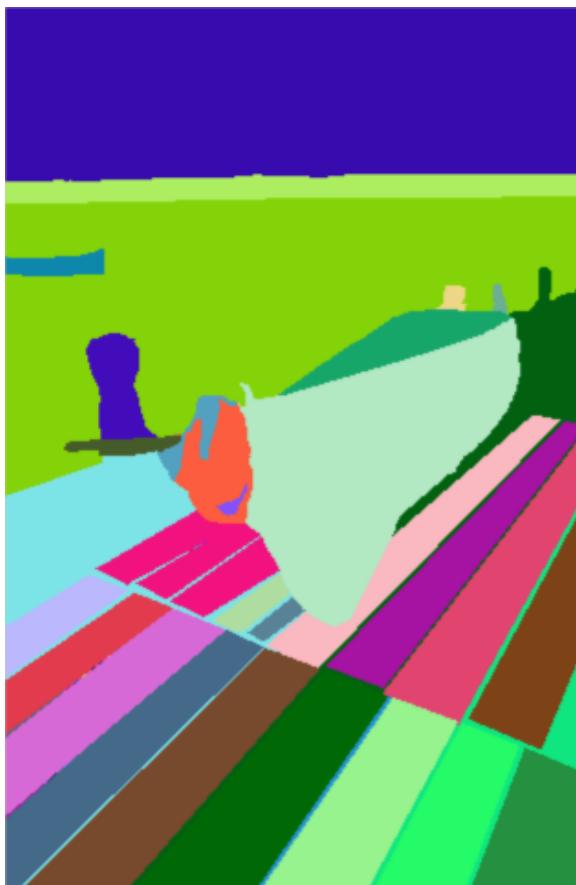
Fundamental challenges: can humans do it?



Fundamental challenges: can humans do it?



Fundamental challenges: can humans do it?



How can we detect boundaries?

Filtering approaches

Canny (1984), Morrone and Owens (1987), Perona and Malik (1991),..

Scale-Space approaches

Tony Lindeberg 'Edge Detection and Ridge Detection with Automatic Scale Selection.', IJCV, 30(2), 117-156, (1998)

Variational approaches

V. Caselles, R. Kimmel, G. Sapiro: Geodesic Active Contours. IJCV22(1): 61-79 (1997)

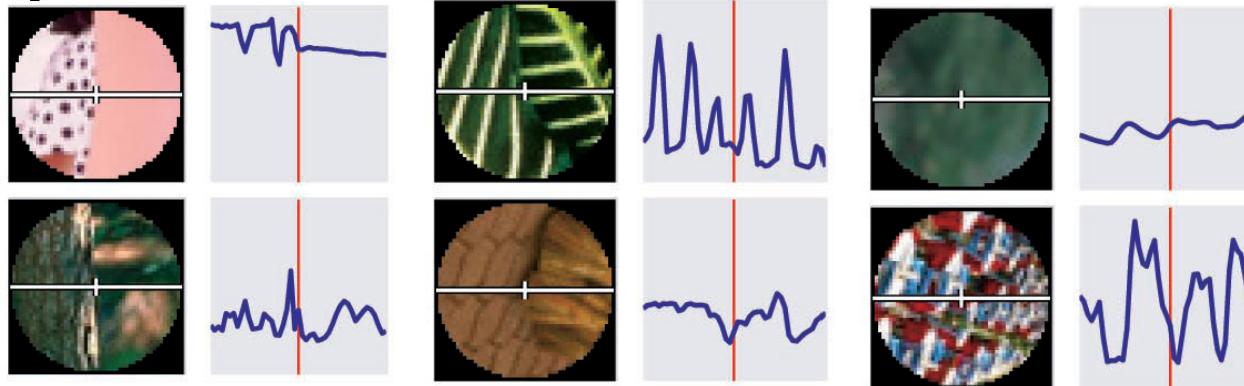
K. Siddiqi, Y. Lauzière, A. Tannenbaum, S. Zucker: Area and length minimizing flows for shape segmentation. IEEE TIP 7(3): 433-443 (1998)

Statistical approaches

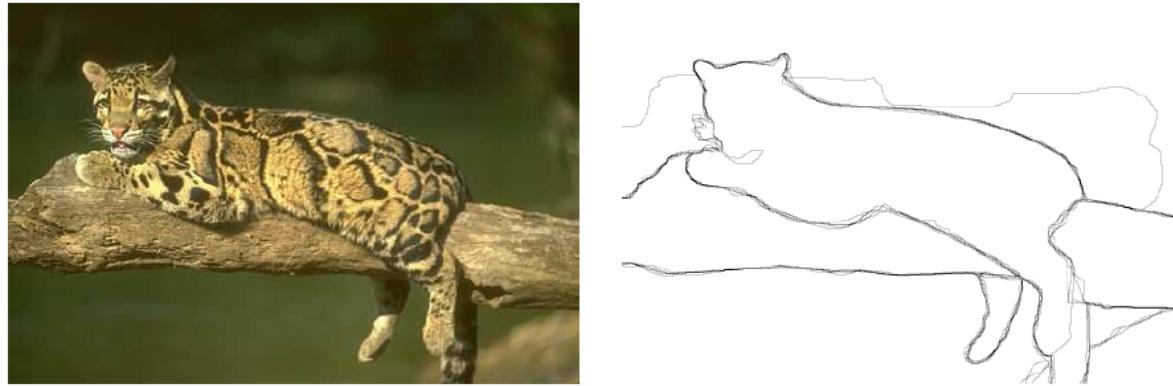
Agnès Desolneux, Lionel Moisan, Jean-Michel Morel: 'Meaningful Alignments'. International Journal of Computer Vision 40(1): 7-23 (2000)

Learning-based approaches

Boundary or non-boundary?



Use human-annotated segmentations



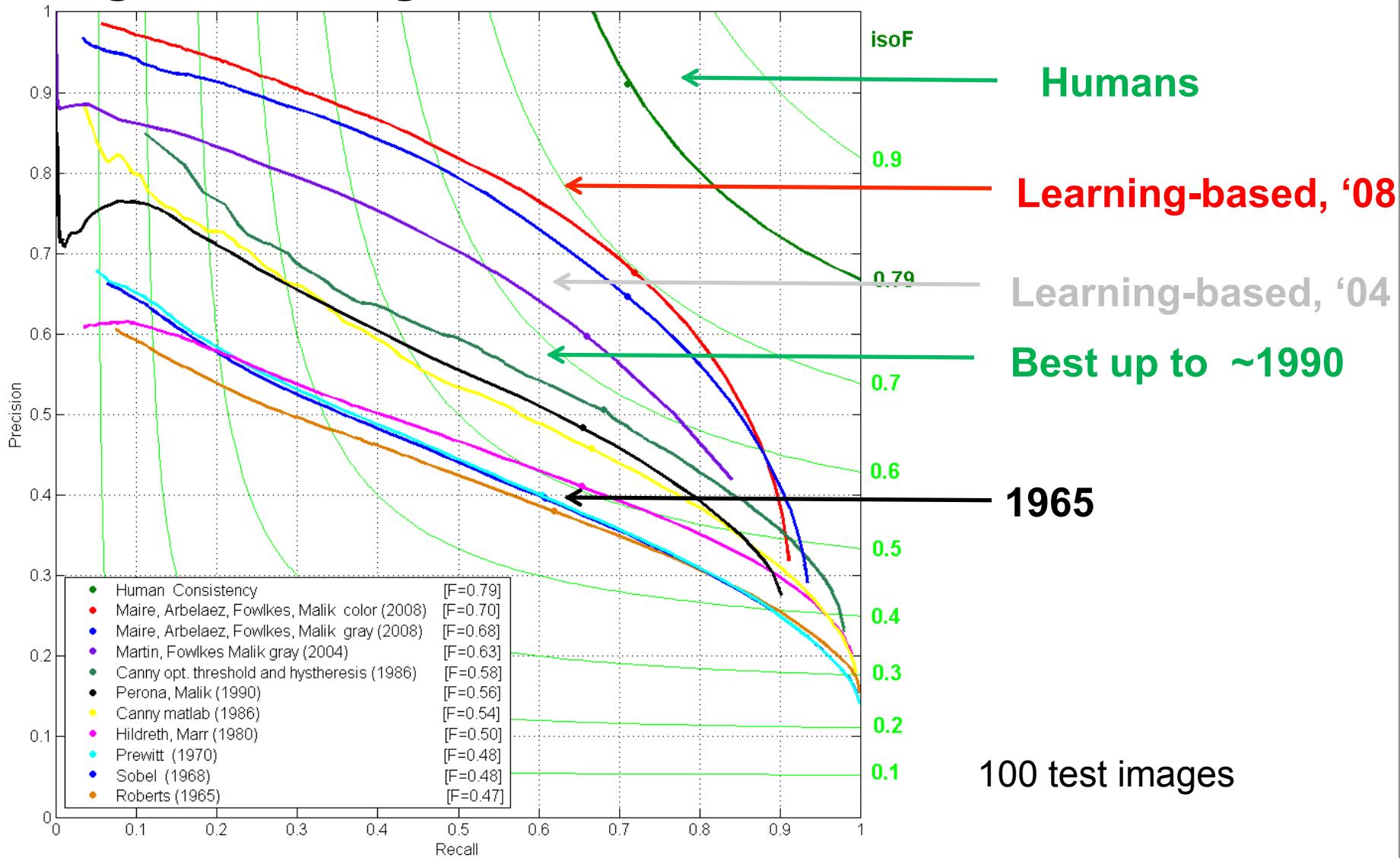
Use any visual cue as input to the decision function.

Use decision trees/logistic regression/boosting/... and *learn* to combine the individual inputs.

S. Konishi, A. Yuille, J. Coughlan, S.C. Zhu, "Statistical Edge Detection: Learning and Evaluating Edge Cues", IEEE PAMI, 2003

D. Martin, C. Fowlkes, J. Malik. "Learning to Detect Natural Image Boundaries Using Local Brightness, Color and Texture Cues", IEEE PAMI, 2004

Progress during the last 4 decades



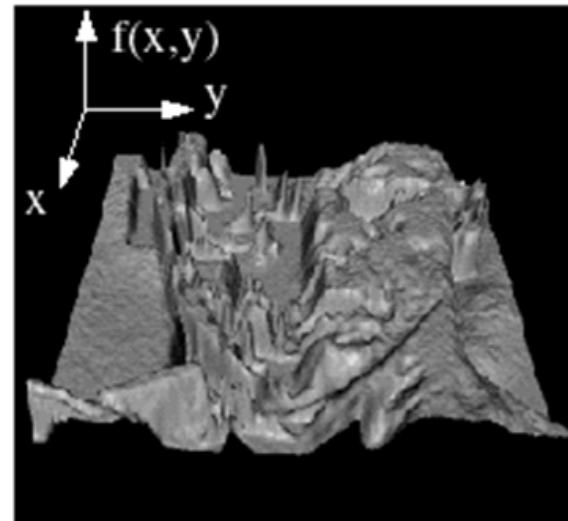
Reference: Maire, Arbelaez, et. al., IEEE PAMI 2011

Learning and Vision problem II: Face Detection

- How do digital cameras detect faces?

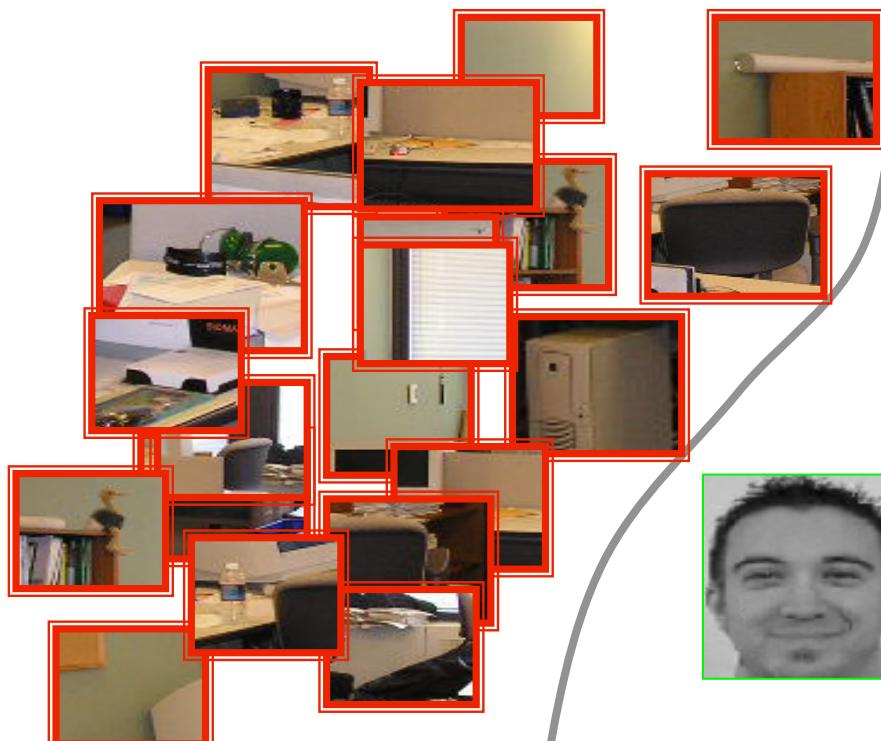


- Input to a digital camera: intensity at pixel locations



‘Faceness function’: classifier

Background



Decision boundary

Face



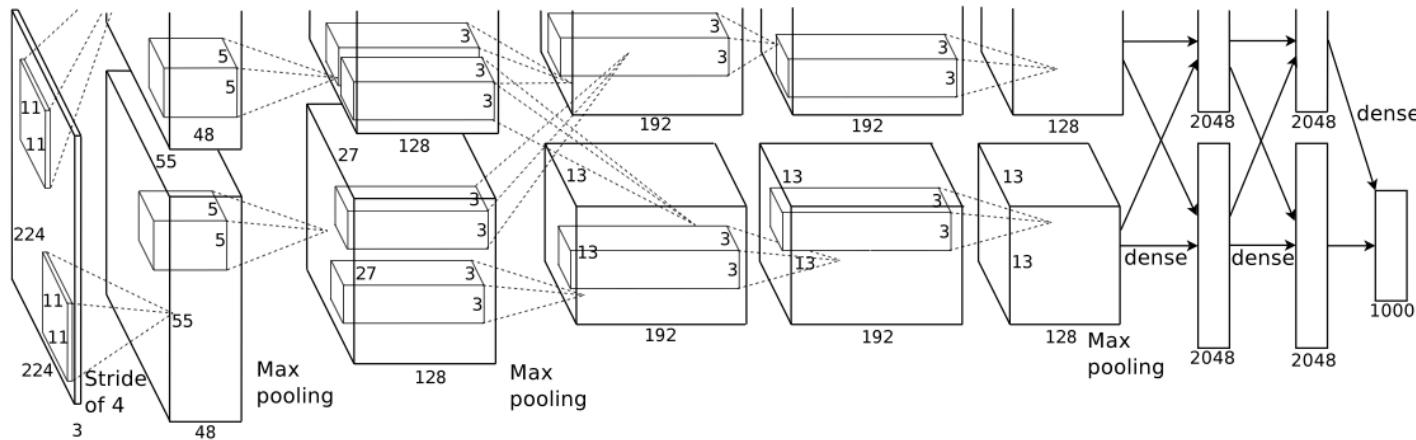
Sliding window approaches

- Scan window over image
 - Multiple scales
 - Multiple orientations
- Classify window as either:
 - Face
 - Non-face



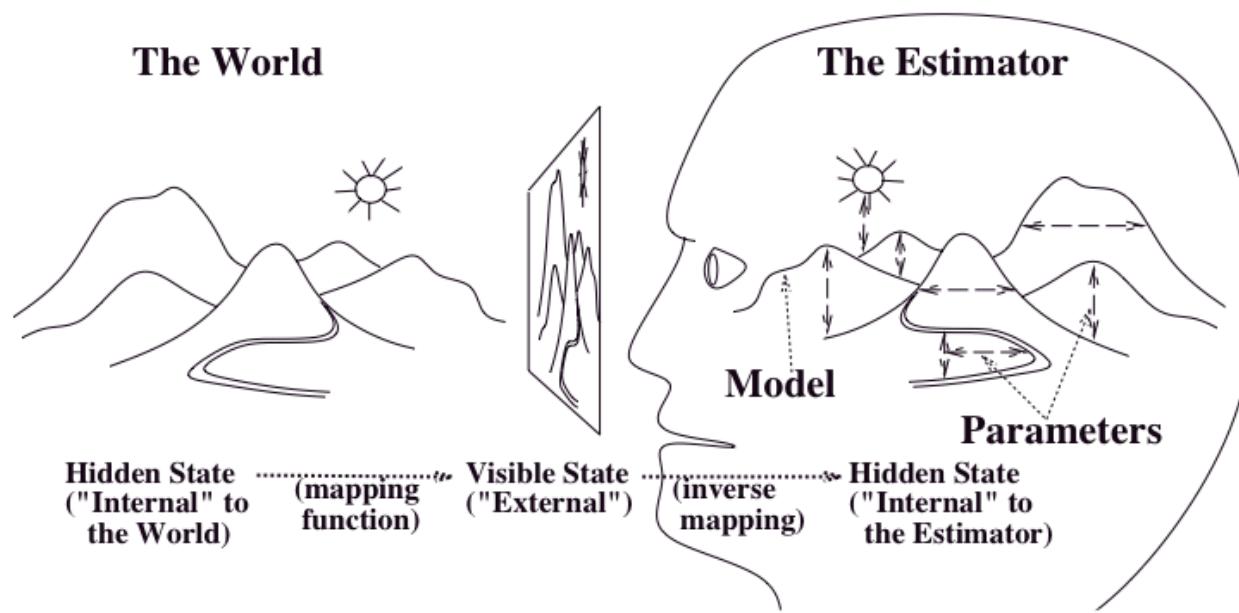
Two Main Approaches

- Discriminative



Two Main Approaches

- Generative



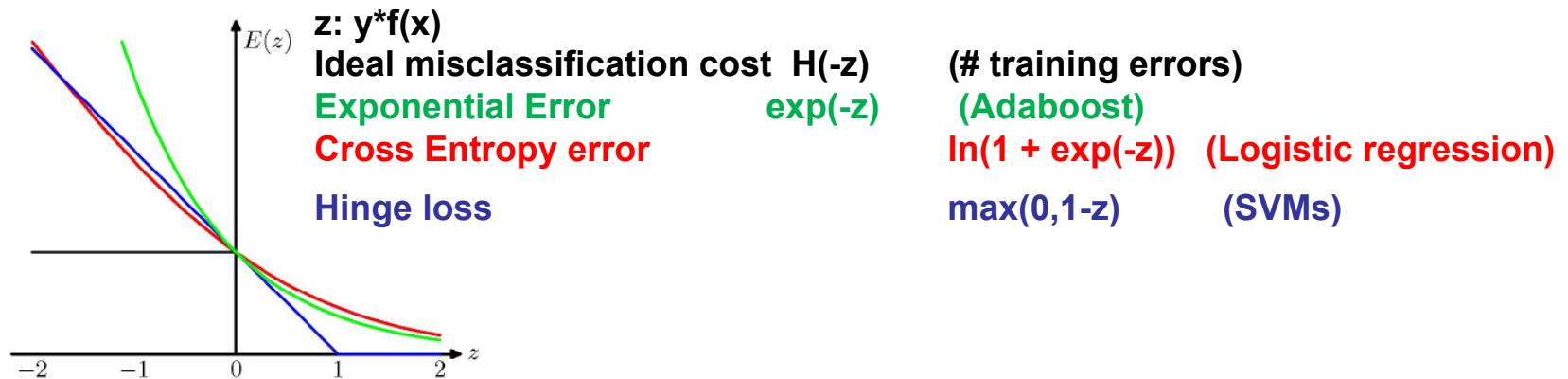
Posterior

Likelihood Prior

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

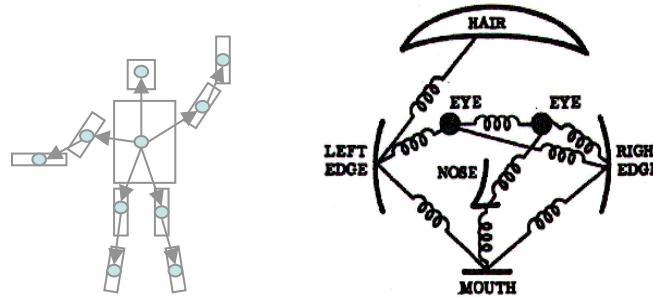
Discriminative techniques

- Lectures 1-4:
 - Linear and Logistic Regression
 - Adaboost, Decision Trees, Random Forests
 - Support Vector Machines
- Unified treatment as loss-based learning

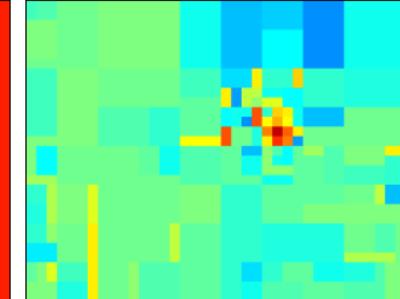
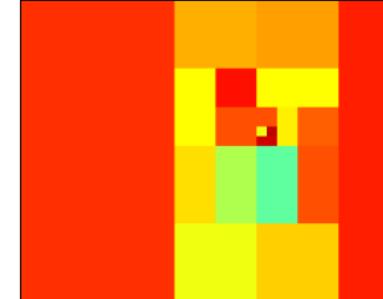
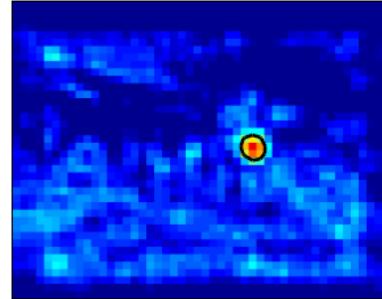


Generative Techniques, Structured models

- Lectures 5-7
 - Hidden Variables, EM, Component Analysis
 - Structured Models (HMMs, Deformable Part Models)

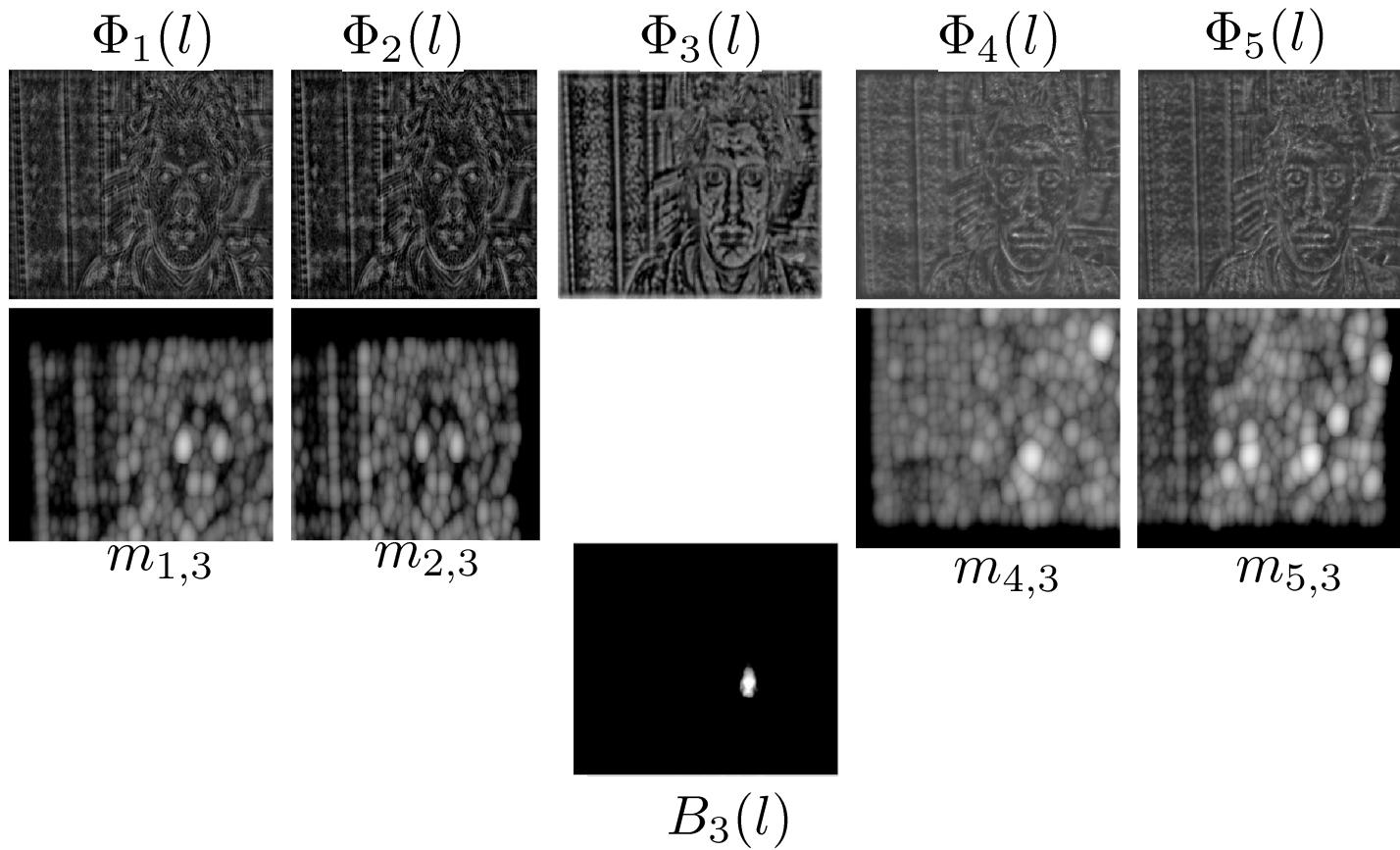
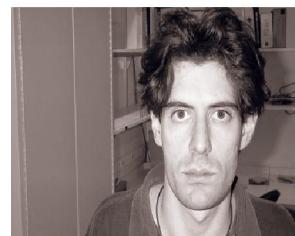


- Latent SVM/Multiple Instance Learning
- Efficient object detection algorithms (Branch & Bound)



(a) Input & Detection result (b) Detector score $S(x)$ (c) BB for $\arg \max_x S(x)$ (d) BB for $S(x) \geq -1$.

From part 1 to part 2



Administrative details

- 2 lab exercises (10 points)
 - Start with small preparatory exercises (synthetic data)
 - Evaluation: real image data
- 1 Project (10 points)
 - finish the object detection system of Labs 1 & 2
 - or implement a recent ICCV/CVPR/ECCV/NIPS/ICML paper
 - or work on a small-scale research project (20/20)
- Tutorials, slides & complementary handouts
<http://www.mas.ecp.fr/vision/Personnel/iasonas/teaching.html>
- Today: I need a list with everyone's email!



Lecture outline

Introduction to the course

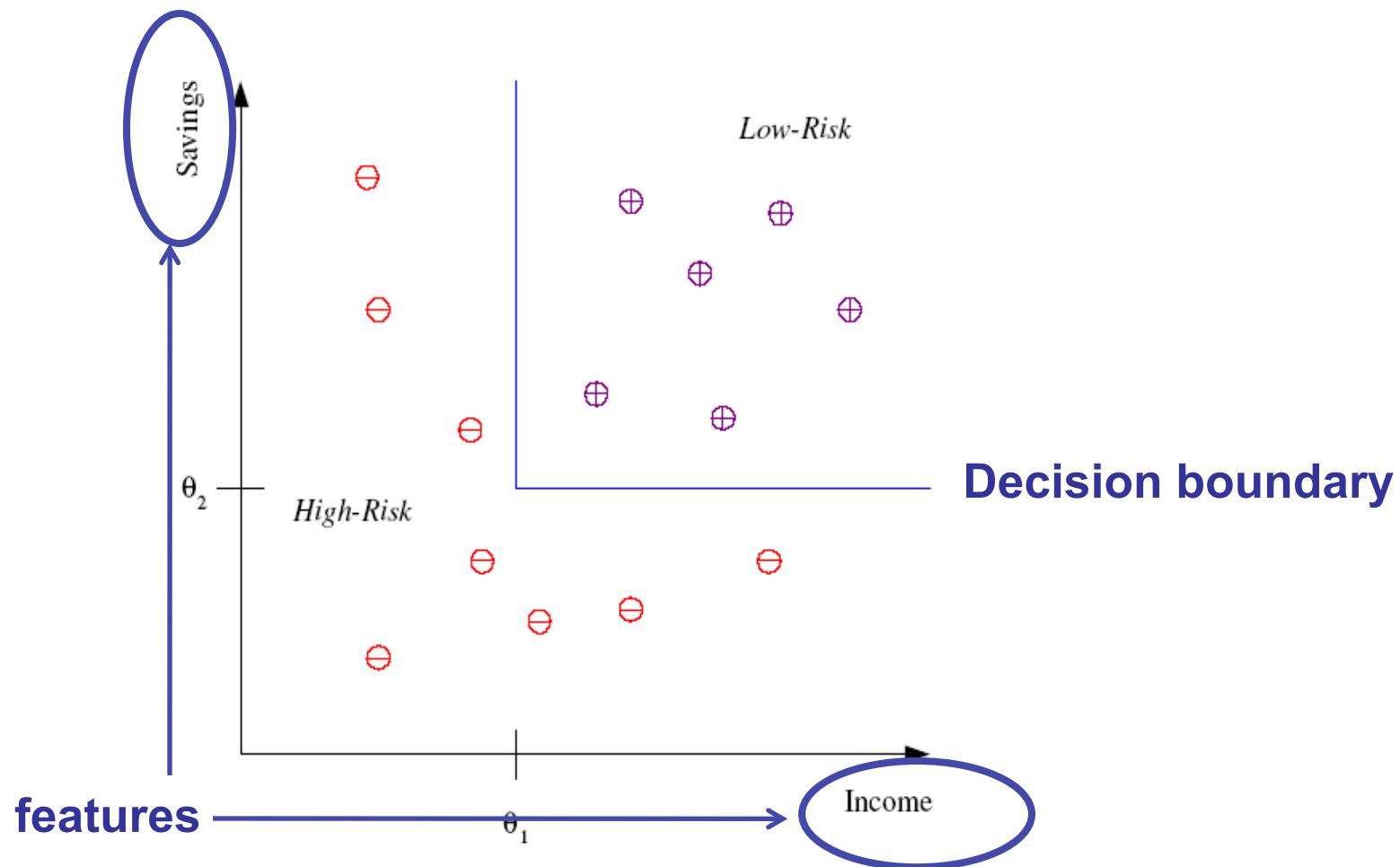
Introduction to the classification problem

Linear Classifiers

Image-based features

Classification Problem

- Based on our experience, should we give a loan to this customer?
 - Binary decision: yes/no



Learning problem formulation

Given: Training set of feature-label pairs $S = \{(x^i, y^i)\} \ i = 1, \dots, N$

$$y^i \in \{0, 1\} \quad x^i \in \mathcal{X}$$

Wanted: ‘simple’ $f : \mathcal{X} \rightarrow \{0, 1\}$ that ‘works well’ for S

Why ‘simple’? good **generalization** outside training set

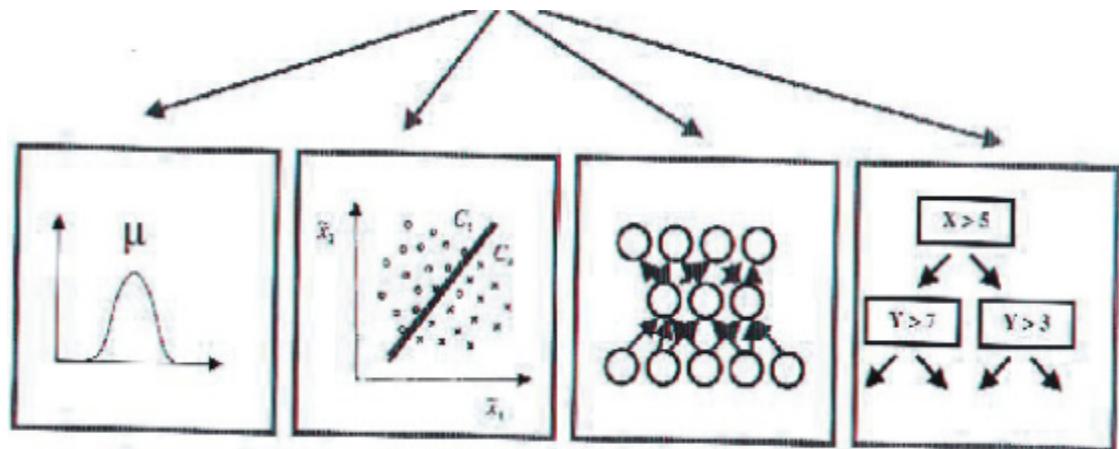
‘works well’: quantified by loss criterion $L(S, f)$

Classifier function

- Input-output mapping

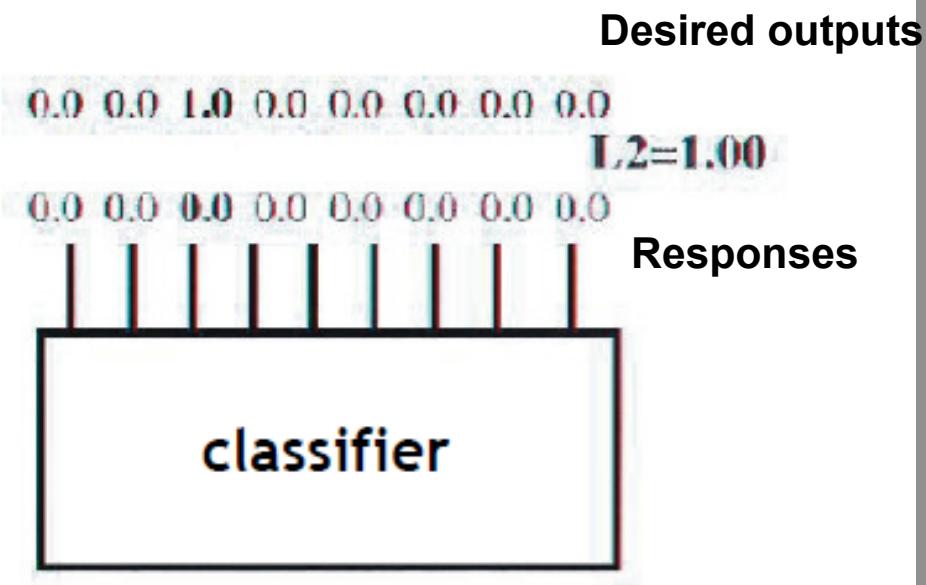
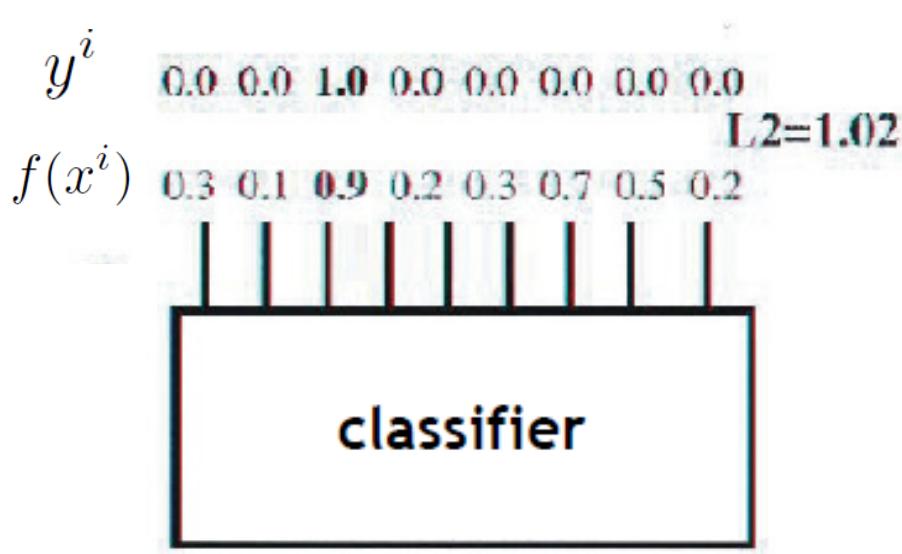
- Output: y
- Input: x
- Method: f
- Parameters: w

$$y = f_w(x) \quad (= f(x, w))$$



- Aspects of the learning problem
 - Identify methods that fit the problem setting
 - Determine parameters that properly classify the training set
 - Measure and control the ‘complexity’ of these functions

Loss criterion



- Observations
 - Euclidean distance is not so good for classification
 - Maybe we should weigh positives more?
- Loss should quantify the probability of error, while keeping the learning problem tractable (e.g. leading to convex objectives)



Lecture outline

Introduction to the class

Introduction to the problem of classification

Linear classifiers

Linear regression and least squares

Regularization: ridge regression

Bias-Variance decomposition

Logistic regression

Linear regression

Classifier: mapping from features $x^i \in \mathcal{X}$ to labels $y^i \in \{0, 1\}$

$$f : \mathcal{X} \rightarrow \{0, 1\}$$

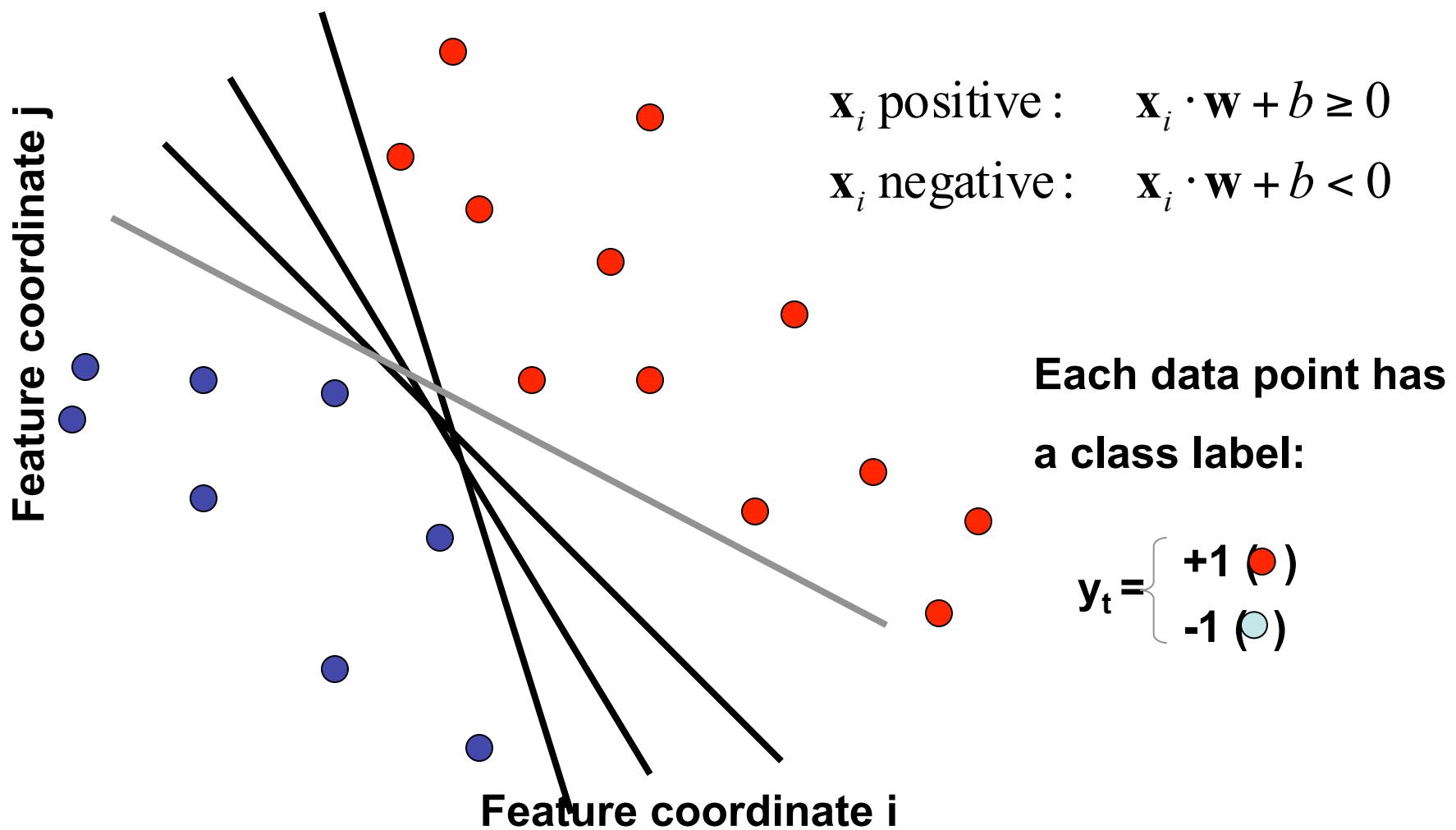
Linear regression: linear $f : R^K \rightarrow R$

$$y = f_w(x) = \langle x, w \rangle = \sum_{k=1}^K x_k w_k$$

binary decision can be obtained by thresholding f

Linear Classifiers

- Find linear expression (*hyperplane*) to separate positive and negative examples



Loss function for linear regression

Training: given $S = \{(x^i, y^i)\} \ i = 1, \dots, N$, estimate optimal w

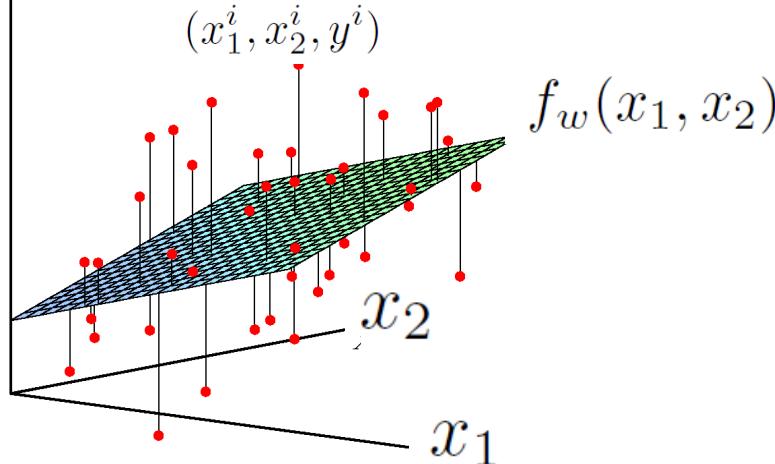
Loss function: quantify appropriateness of w

$$L(S, f_w) = \sum_{i=1}^N l(y^i, f_w(x^i)) = \sum_{i=1}^N (y^i - \langle x^i, w \rangle)^2$$

y

sum of individual errors ('additive')

quadratic



Why this loss function?

Easy to optimize!

Least squares solution for linear regression

Loss function: $L(w) = \sum_{i=1}^N (y^i - \langle x^i, w \rangle)^2$

Introduce vectors and matrixes to rewrite as quadratic expression:

$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \quad \mathbf{X} = \begin{bmatrix} x_1^1 & \dots & x_k^1 & \dots & x_K^1 \\ & & \vdots & & \\ x_1^N & \dots & x_k^N & \dots & x_K^N \end{bmatrix} \quad \mathbf{w} = \begin{bmatrix} w_1 \\ \vdots \\ w_k \\ \vdots \\ w_K \end{bmatrix}$$

Residual : $\mathbf{e} = \mathbf{y} - \mathbf{X}\mathbf{w}$

$$L(\mathbf{w}) = \mathbf{e}^T \mathbf{e} = \mathbf{y}^T \mathbf{y} - 2\mathbf{w}^T \mathbf{X}^T \mathbf{y} + \mathbf{w}^T \mathbf{X}^T \mathbf{X} \mathbf{w}$$

$$J(\mathbf{w}) = -2\mathbf{X}^T \mathbf{y} + 2\mathbf{X}^T \mathbf{X} \mathbf{w}$$

$$\mathbf{w}^* = \arg \min_{\mathbf{w}} L(\mathbf{w}) = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$$

Questions

Is the loss function appropriate?

Quadratic loss: convex cost, closed-form solution

But does the optimized quantity indicate classifier's performance?

Is the classifier appropriate?

Linear classifier: fast computation

But could e.g. a non-linear classifier have better performance?

Are the estimated parameters good?

Parameters recover input-output mapping on training data

How can we know they do not simply memorize training data?

Questions

Is the loss function appropriate?

Quadratic loss: convex cost, closed-form solution

But does the optimized quantity indicate classifier's performance?

Is the classifier appropriate?

Linear classifier: fast computation

But could e.g. a non-linear classifier have better performance?

Are the estimated parameters good?

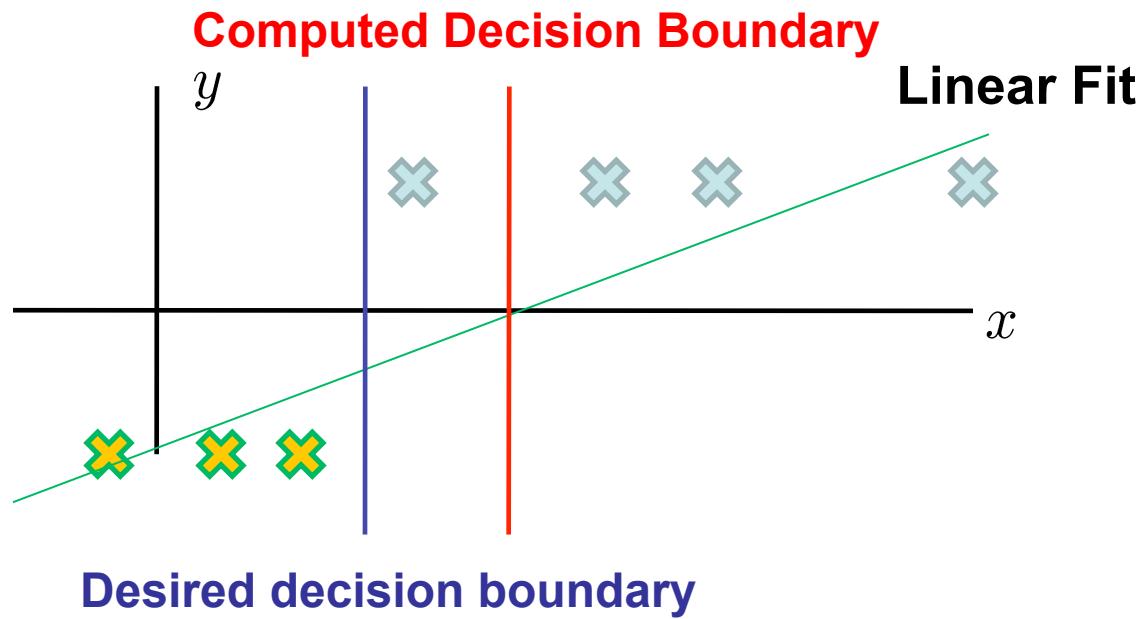
Parameters recover input-output mapping on training data

How can we know they do not simply memorize training data?

Inappropriateness of quadratic penalty

We chose the quadratic cost function for convenience
Single, global minimum & closed form expression

But does it indicate classification performance?



Quadratic norm penalizes outputs that are 'too good'

Logistic regression, SVMs, Adaboost: more appropriate loss

Questions

Is the loss function appropriate?

Quadratic loss: convex cost, closed-form solution

But does the optimized quantity indicate classifier's performance?

Is the classifier appropriate?

Linear classifier: fast computation

But could e.g. a non-linear classifier have better performance?

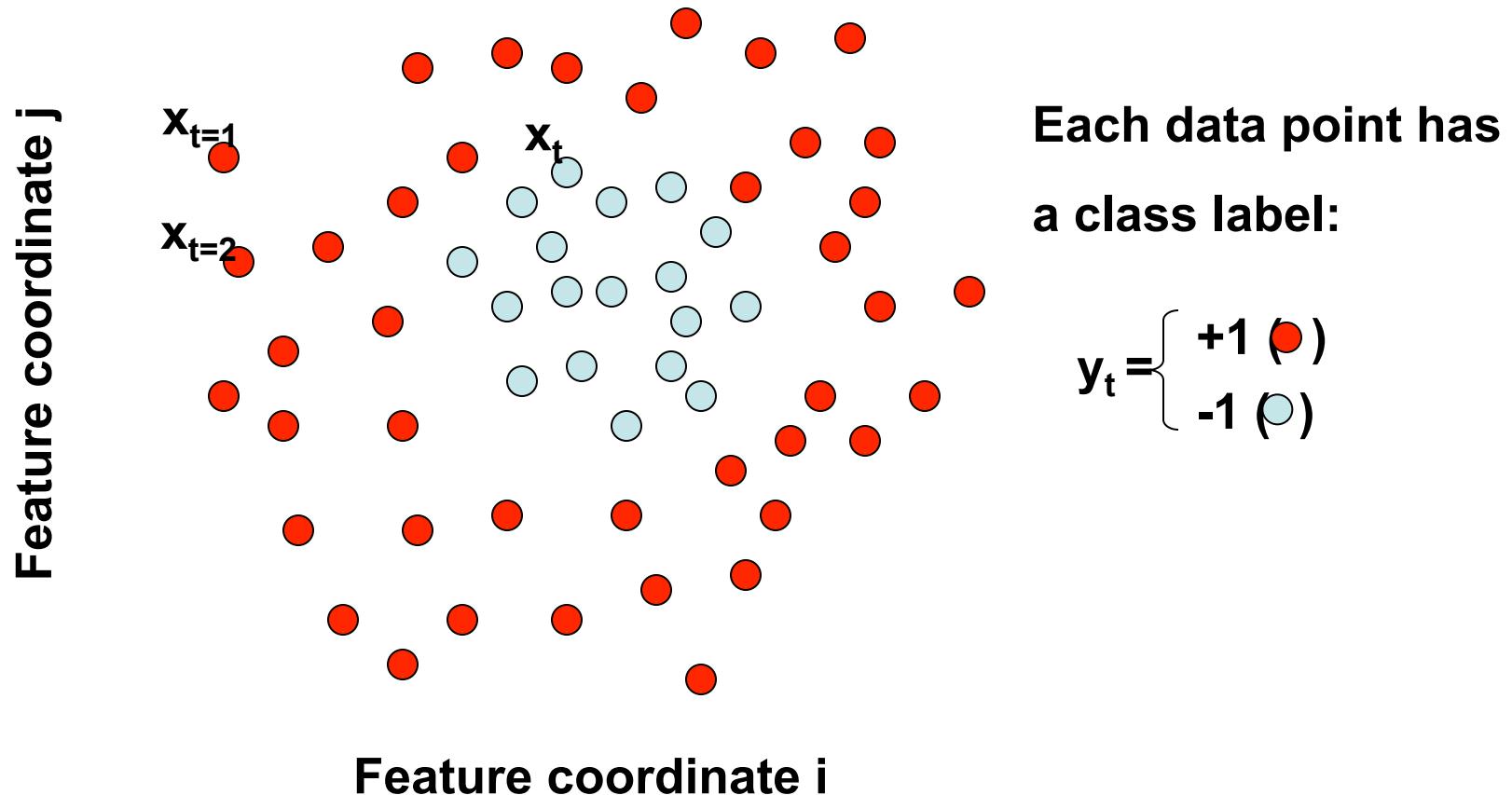
Are the estimated parameters good?

Parameters recover input-output mapping on training data

How can we know they do not simply memorize training data?

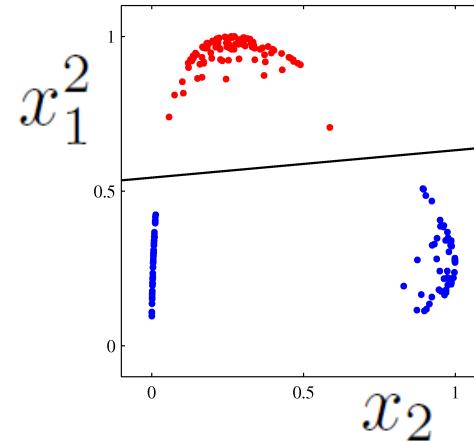
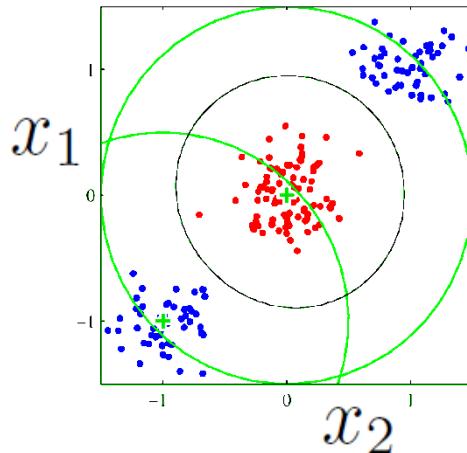
Classes may not be linearly separable

Linear classifier cannot properly separate these data



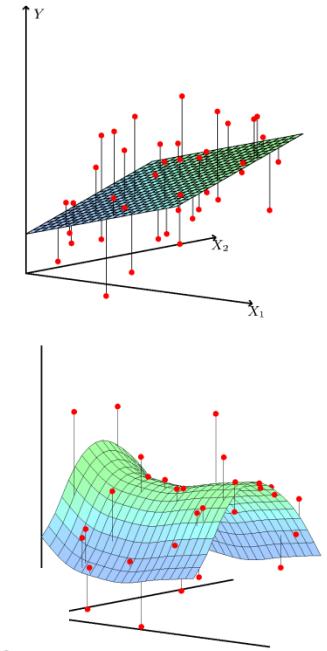
Beyond linear boundaries

Non-linear features: non-linear classifiers & decision boundaries



$$x \rightarrow \phi(x) = [\phi_1(x), \dots, \phi_M(x)]$$

$$h(x, w) = f(\phi(x), w) = \sum_{m=1}^M \phi_m(x) w_m$$



How do we pick the right features?

This class: domain knowledge

Next classes: kernel trick (svms) greedy selection (boosting)

Questions

Is the loss function appropriate?

Quadratic loss: convex cost, closed-form solution

But does the optimized quantity indicate classifier's performance?

Is the classifier appropriate?

Linear classifier: fast computation

But could e.g. a non-linear classifier have better performance?

Are the estimated parameters good?

Parameters recover input-output mapping on training data

How can we know they do not simply memorize training data?



Lecture outline

Introduction to the class

Introduction to the problem of classification

Linear regression

Linear regression and least squares

Regularization: ridge regression

Bias-Variance decomposition

Image-based features

Overfitting problem

Learning problem: 100 faces, 1000 background images

Image resolution: 100×100 pixels (10000 intensity values)

Linear regression: $y^i \simeq f_w(x^i) = \langle w, x^i \rangle \quad w \in R^{10000}$
 $i \in \{1, \dots, 100\}$

More unknowns than equations: ill posed problem

perfect performance on training set

unpredictable performance on new data

$$\mathbf{w}^* = \left(\underbrace{\mathbf{X}^T \mathbf{X}}_{10^4 \times 10^4} \right)^{-1} \mathbf{X}^T \mathbf{y}$$

Rank-deficient matrix

‘Curse of dimensionality’: in high-dimensional spaces data become sparse

L2 Regularization: Ridge regression

Penalize classifier's L2 norm: $\|w\|_2^2 = \sum_{k=1}^K w_k^2 = \mathbf{w}^T \mathbf{w}$

Loss function: $L(\mathbf{w}) = \mathbf{e}^T \mathbf{e} + \lambda \mathbf{w}^T \mathbf{w}$ $\mathbf{e} = \mathbf{y} - \mathbf{X}\mathbf{w}$

 data term complexity term residual

$$= \mathbf{y}^T \mathbf{y} - 2\mathbf{w}^T \mathbf{X}^T \mathbf{y} + \mathbf{w}^T (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I}) \mathbf{w}$$

$$\mathbf{w}^* = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y}$$

Full-rank matrix

So how do we set λ ?

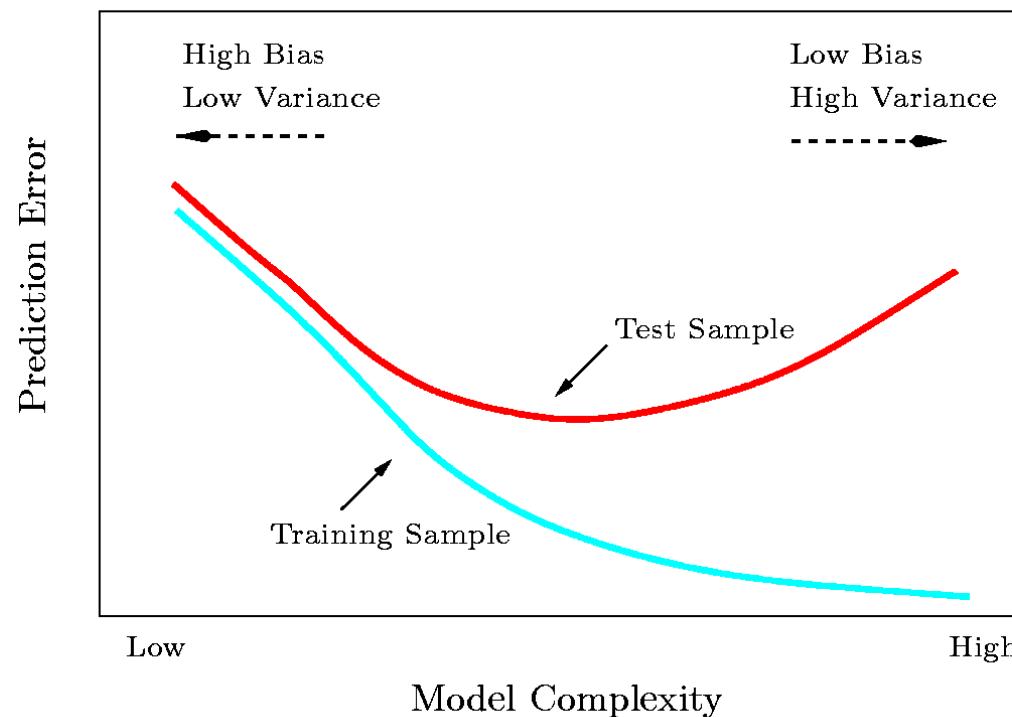
Tuning the model's complexity

A flexible model approximates the target function well in the training set

but can overtrain and have poor performance on the test set

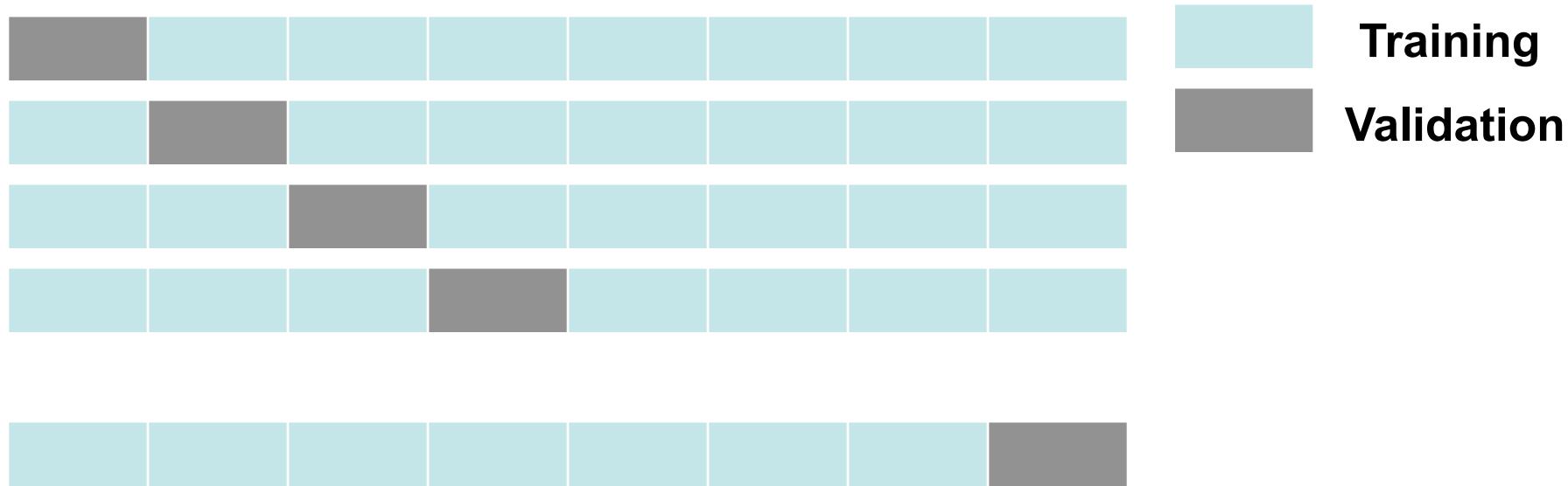
A rigid model's performance is more predictable in the test set

but the model may not be good even on the training set



Selecting λ with cross-validation

- Cross validation technique
 - Exclude part of the training data from parameter estimation
 - Use them only to predict the test error
- 10-fold cross validation:



- Use cross-validation for different values of λ
 - pick value that minimizes cross-validation error



Lecture outline

Introduction to the class

Introduction to the problem of classification

Linear classifiers

Image-based features

Domain knowledge

We may know that data undergo transformations irrelevant to their class

E-mail address: capital letters (`Iasonas@gmail.com` = `iasonas@gmail.com`)

Speech recognition: voice amplitude is irrelevant to uttered words

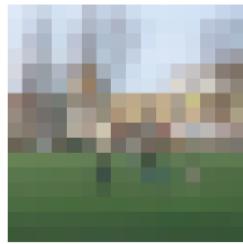
Computer vision: illumination variations



Invariant features: not affected by irrelevant signal transformations

Photometry-invariant patch features

Photometric transformation: $I \rightarrow aI + b$



Original Patch and Intensity Values



Brightness Decreased



Contrast increased,

- Make each patch have zero mean:

$$\mu = \frac{1}{N} \sum_{x,y} I(x, y)$$

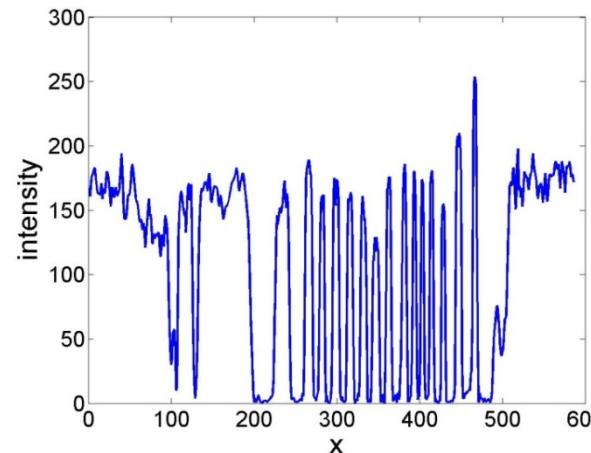
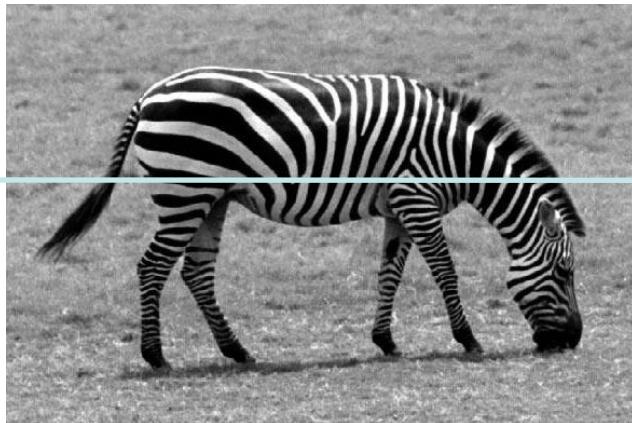
$$Z(x, y) = I(x, y) - \mu$$

- Then make it have unit variance:

$$\sigma^2 = \frac{1}{N} \sum_{x,y} Z(x, y)^2$$

$$ZN(x, y) = \frac{Z(x, y)}{\sigma}$$

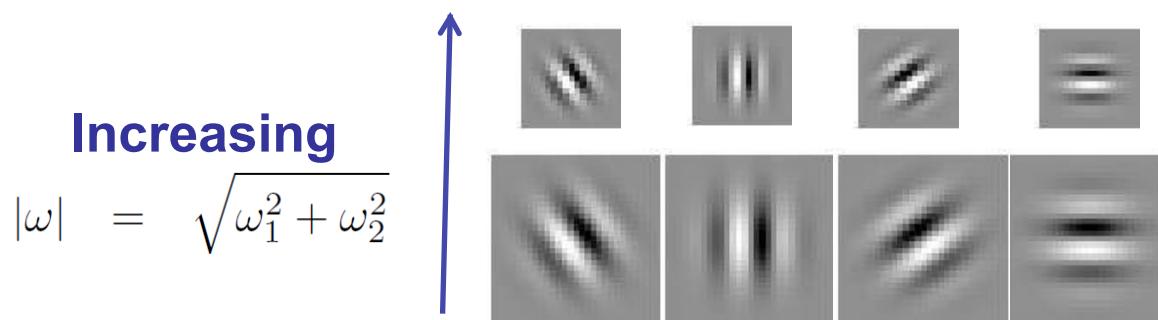
Dealing with texture



What kind of features can appropriately describe texture patterns?

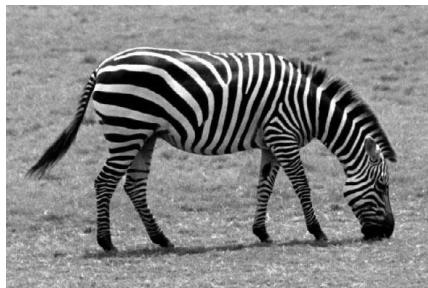
'appropriately': in terms of well-behaved functions

Gabor wavelets: $G_{\omega_1, \omega_2, \sigma}(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \exp(j\omega_1 x + \omega_2 y)$

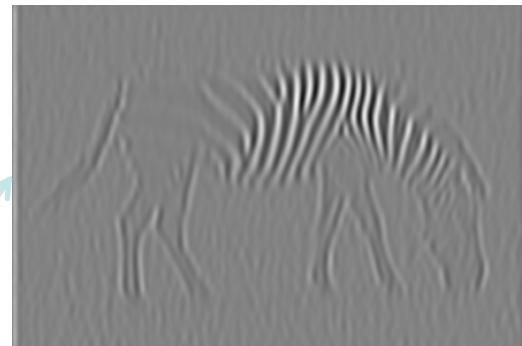


Envelope estimation (demodulation)

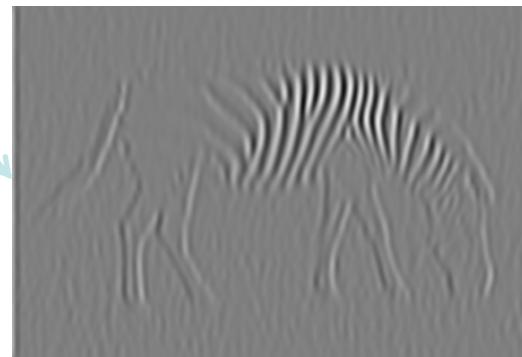
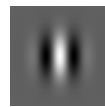
$$G_o(x, y) = G_{\sigma, \vec{\omega}}^{\text{odd}}(x, y) * f(x, y) = \sqrt{G_e^2(x, y) + G_o^2(x, y)}$$



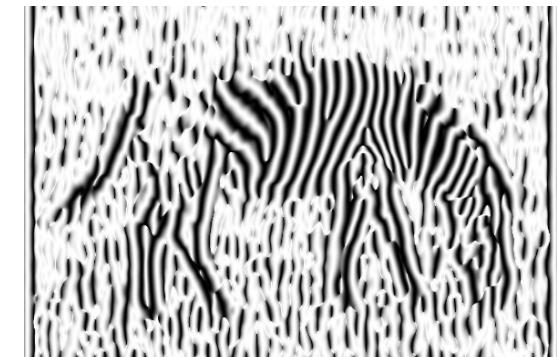
Convolve



$$G_e(x, y) = G_{\sigma, \vec{\omega}}^{\text{even}}(x, y) * f(x, y)$$

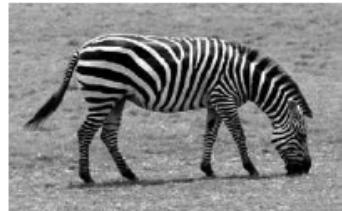


$$\phi(x, y) = \tan^{-1} \frac{G_o(x, y)}{G_e(x, y)}$$

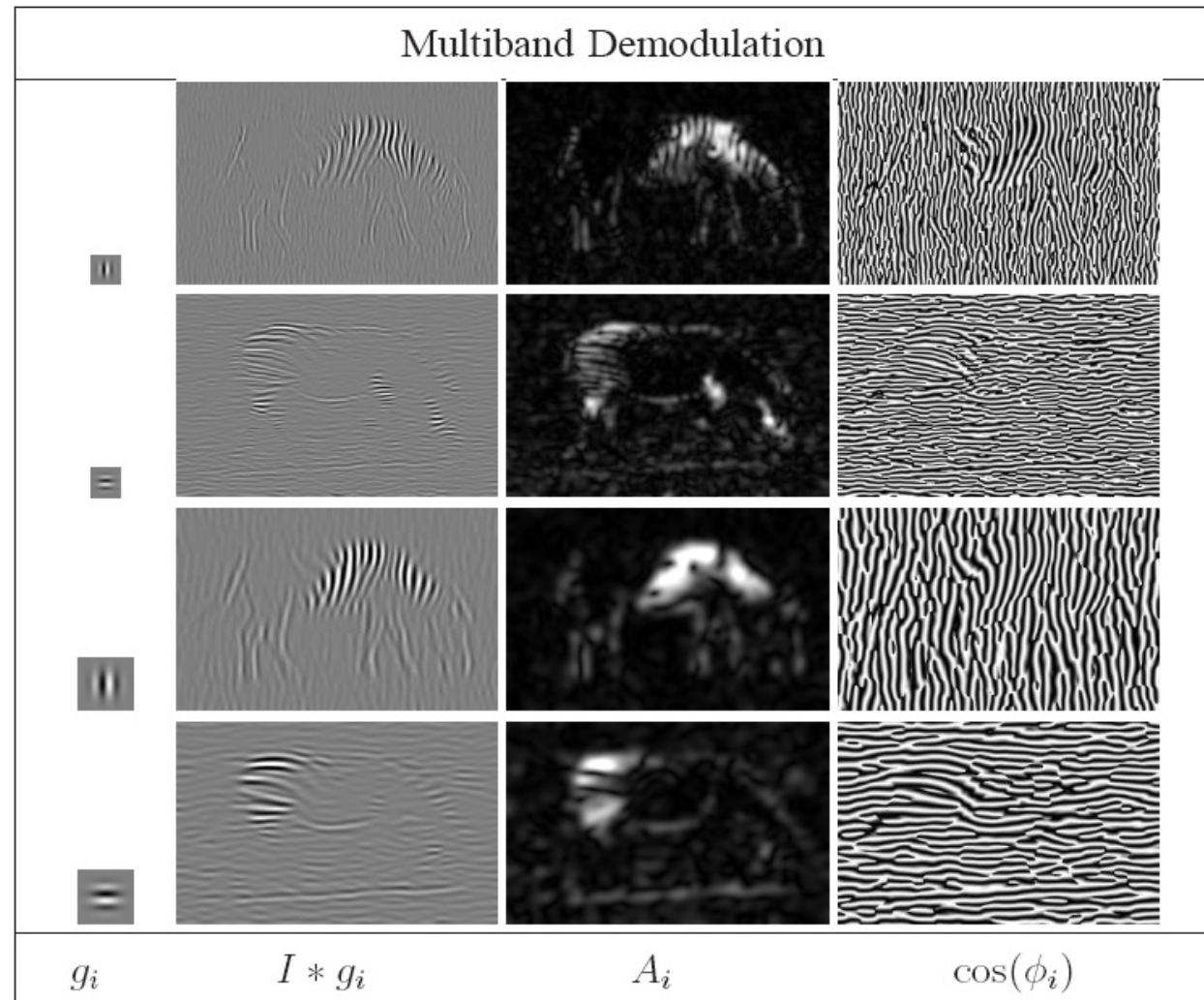


$$I(x, y) = a(x, y) \cos(\phi(x, y)) + c, \quad \vec{\omega}(x, y) = \nabla \phi(x, y)$$

Multiband demodulation with a Gabor filterbank



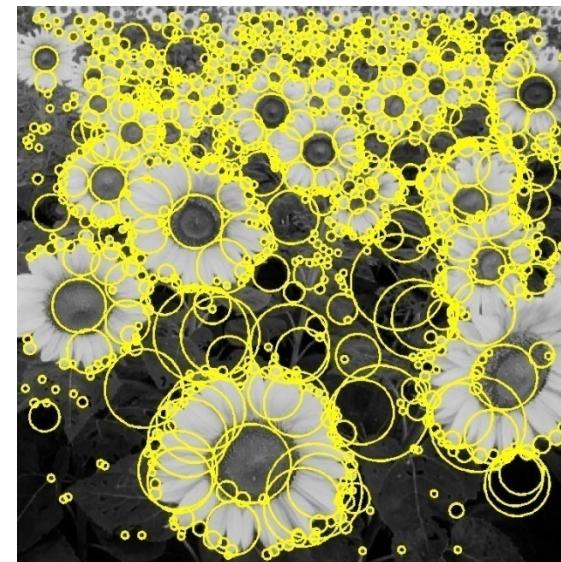
Analysis
→



Dealing with changes in scale and orientation



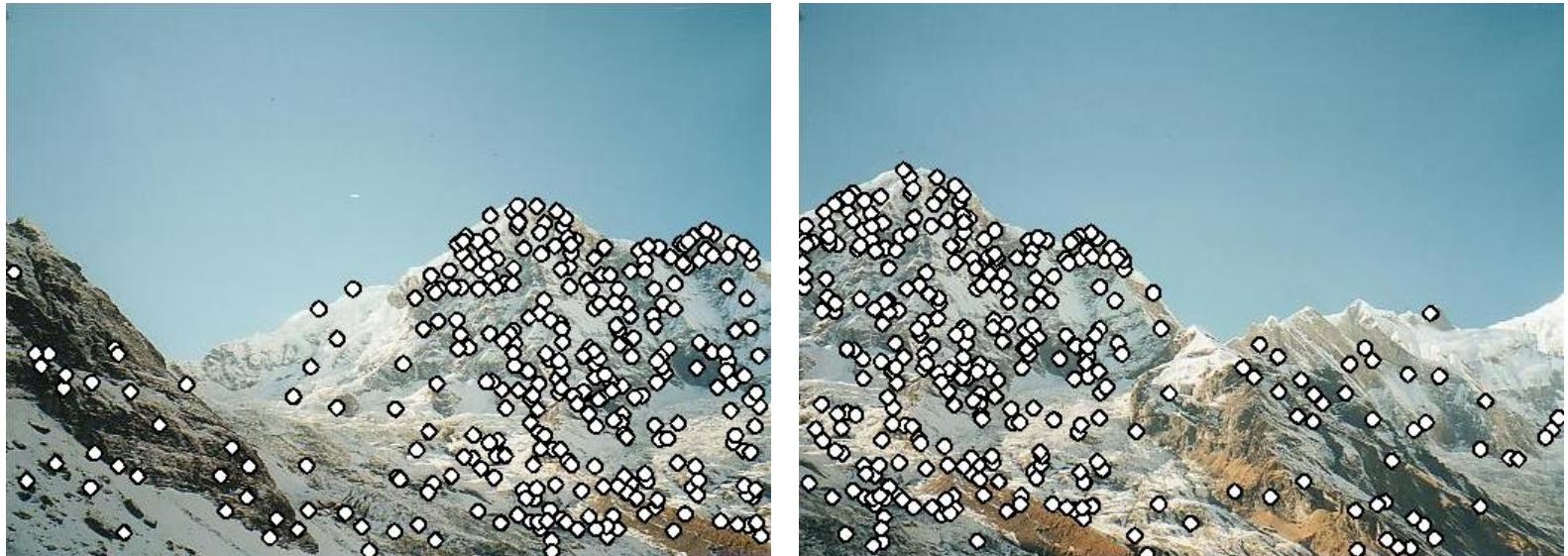
Scale-invariant blob detector



Application: Image Stitching

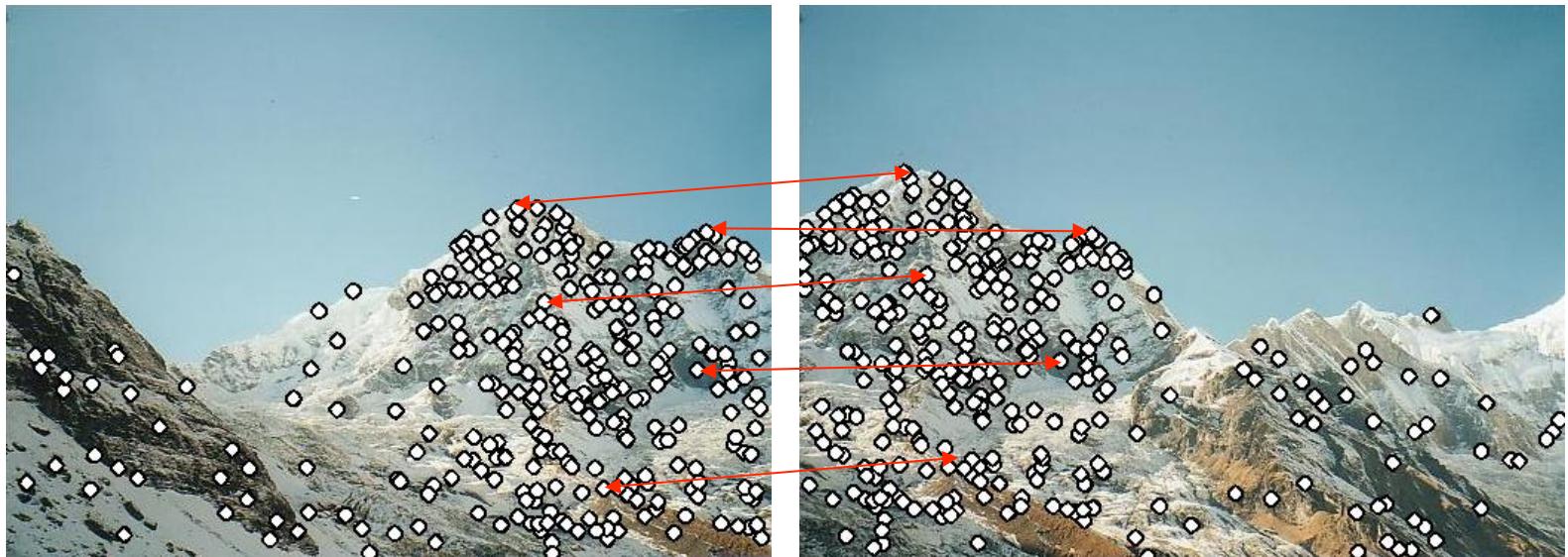


Application: Image Stitching



- Procedure:
 - Detect feature points in both images

Application: Image Stitching



- Procedure:
 - Detect feature points in both images
 - Find corresponding pairs

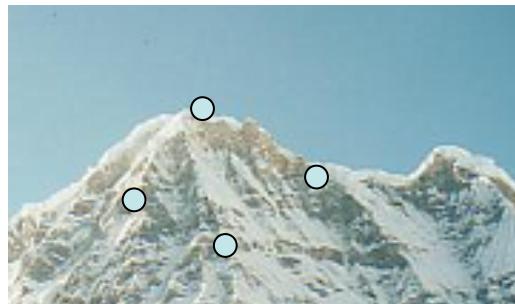
Application: Image Stitching



- Procedure:
 - Detect feature points in both images
 - Find corresponding pairs
 - Use these pairs to align the images

Common Requirements

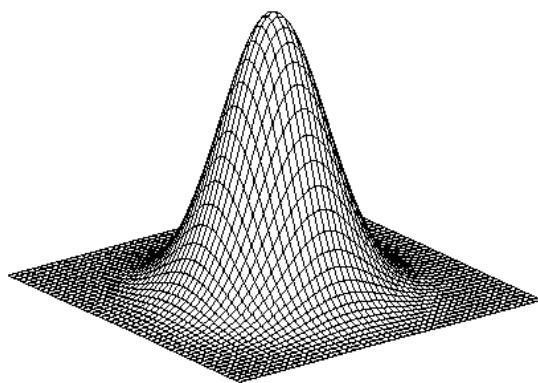
- Problem 1:
 - Detect the same point *independently* in both images



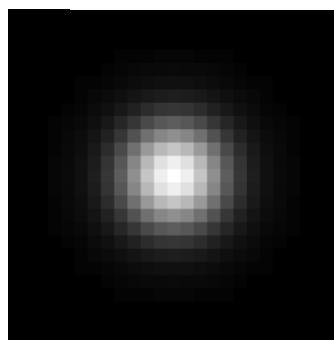
No chance to match!

Laplacian-of-Gaussian

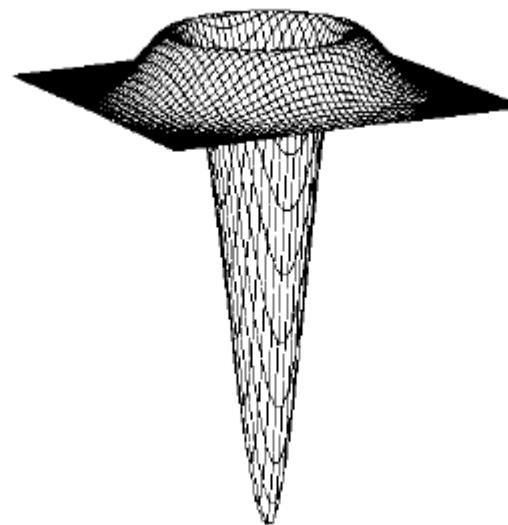
Gaussian



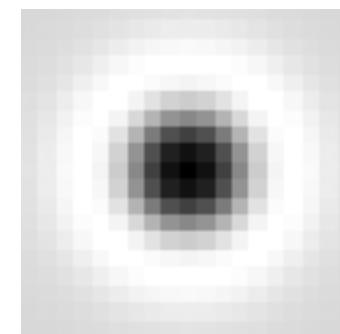
$$g_\sigma(x, y)$$



Laplacian of Gaussian

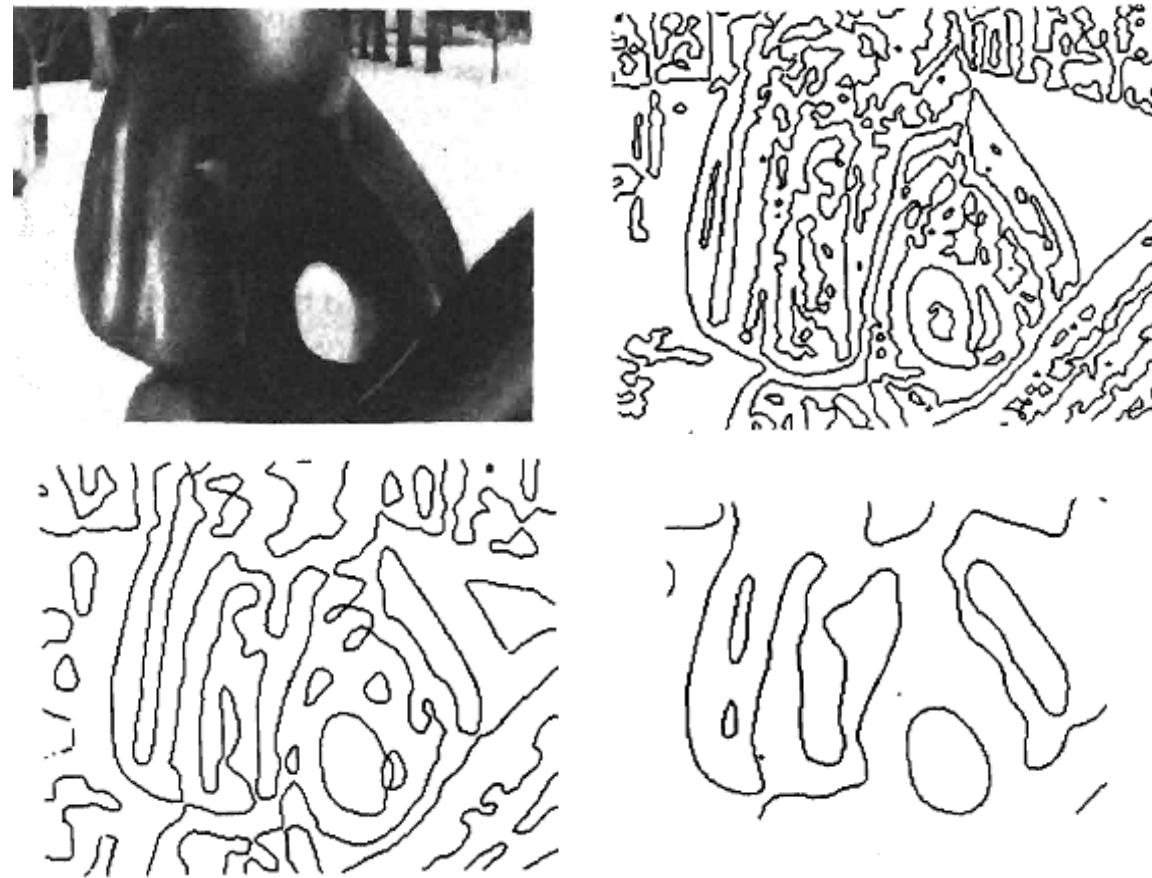


$$\nabla^2 g_\sigma(x, y) = \frac{\partial^2 g_\sigma(x, y)}{\partial x^2} + \frac{\partial^2 g_\sigma(x, y)}{\partial y^2}$$



Early edge detection research

- Zero-crossings of LoG operator at increasing scales

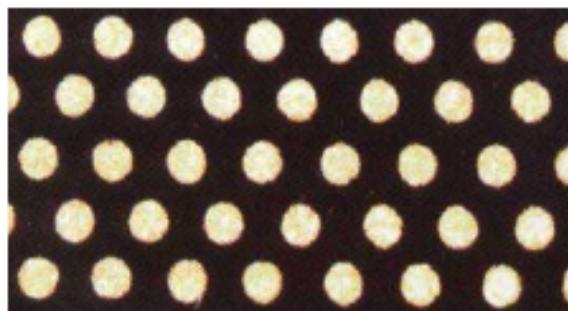


- Different take: go for the maxima/minima

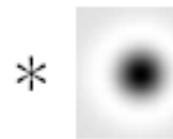
Finding blobs

Filtering= inner product between image patch and filter: template matching

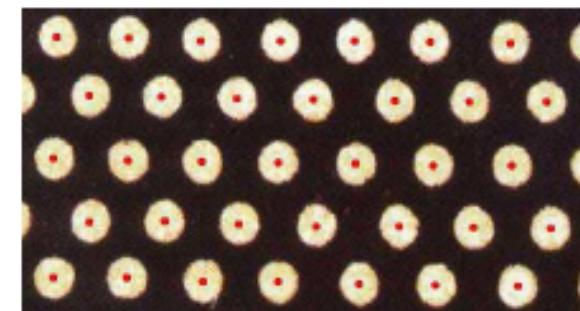
$$\begin{aligned}|I - f|^2 &= \langle I - f, I - f \rangle \\&= \langle I, I \rangle + \langle f, f \rangle - 2\langle f, I \rangle \\&= C - 2\langle I, f \rangle\end{aligned}$$



Polka Dots



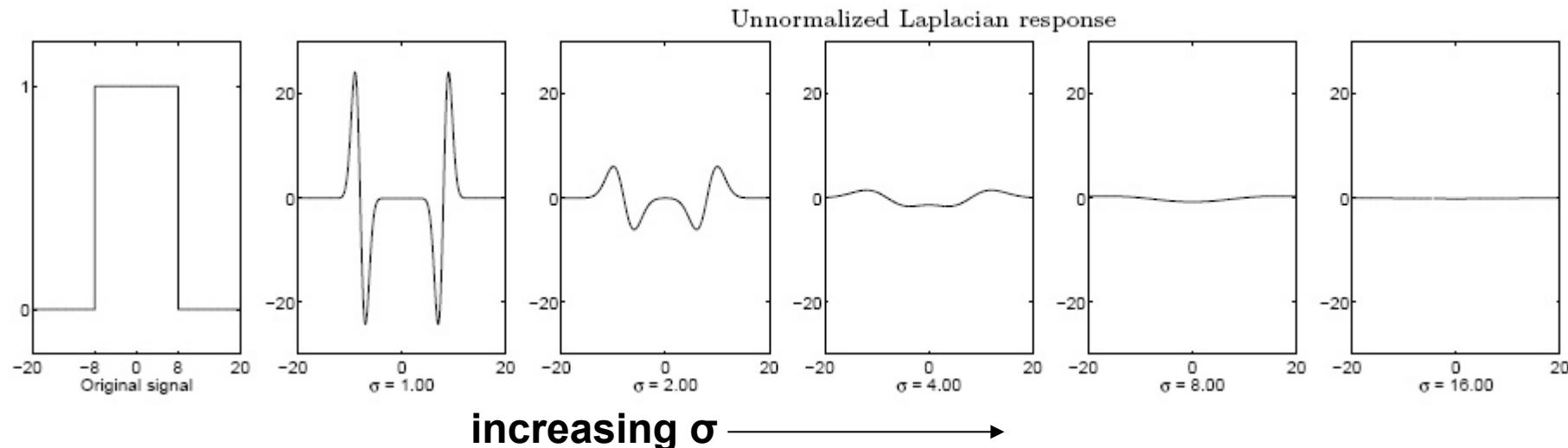
*



Detected Blobs

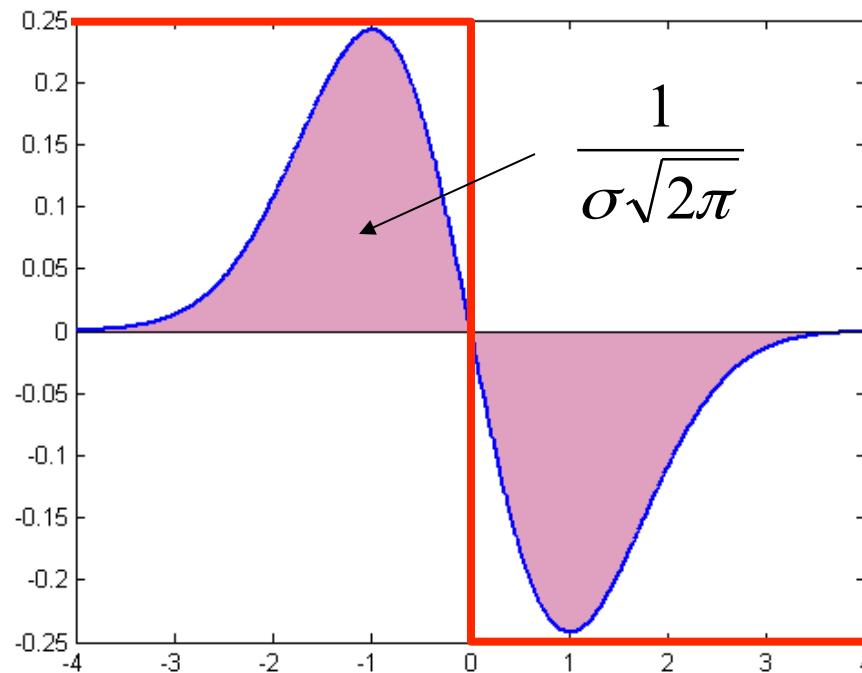
Scale selection

- First idea: convolve with Laplacians at several scales and find maximum in scale
- Observation: Laplacian decays as scale increases:



Scale normalization

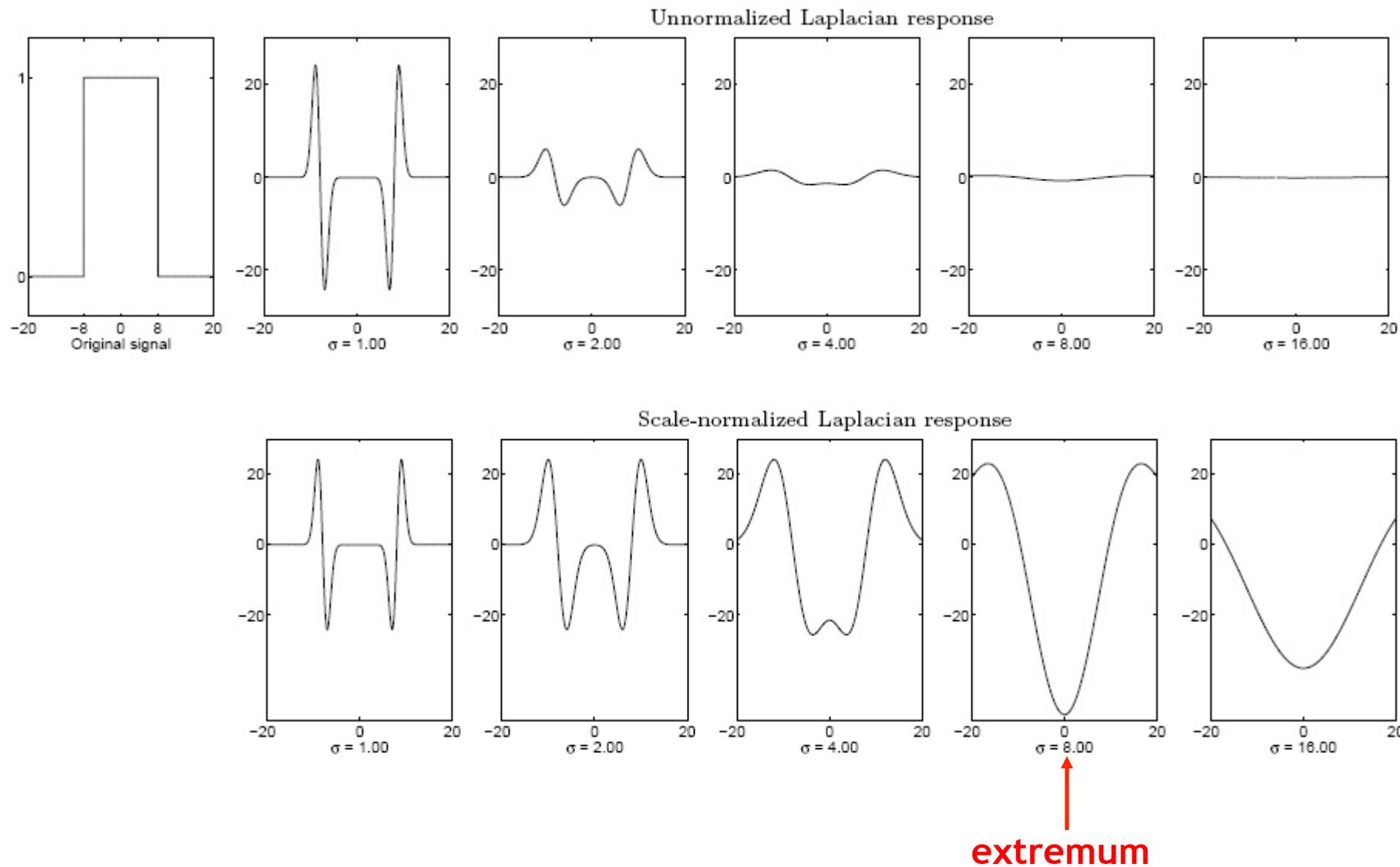
- The response of a derivative of Gaussian filter to a perfect step edge decreases as σ increases



Scale normalization

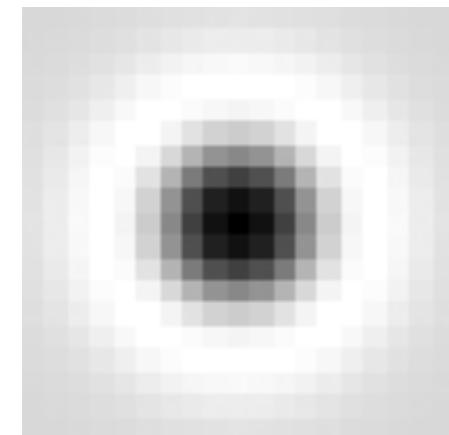
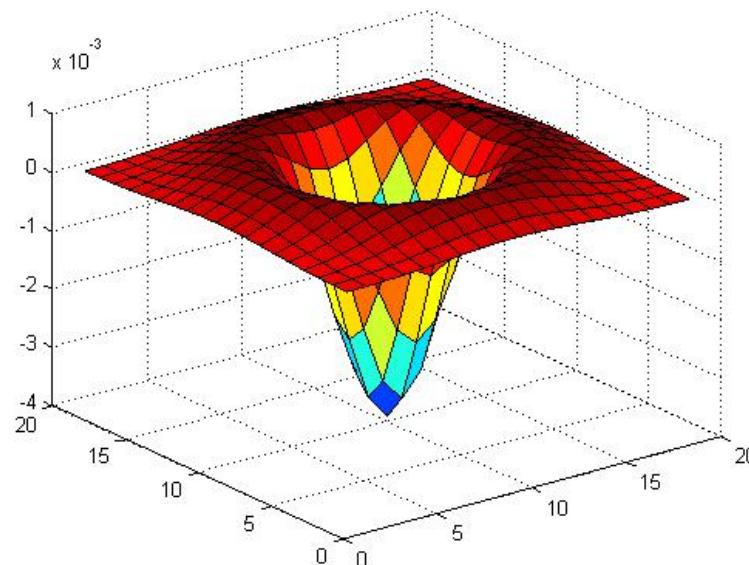
- The response of a derivative of Gaussian filter to a perfect step edge decreases as σ increases
- To keep response the same (scale-invariant), must multiply Gaussian derivative by σ
- Laplacian is the second Gaussian derivative, so it must be multiplied by σ^2

Effect of scale normalization



Blob detection in 2D

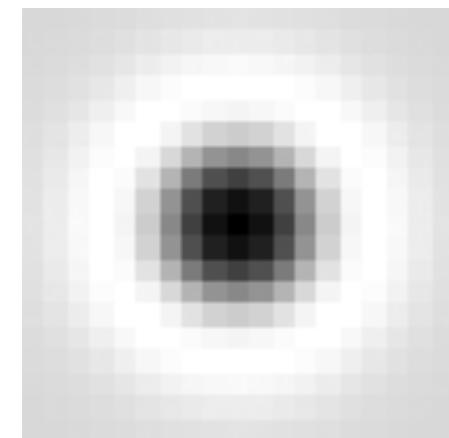
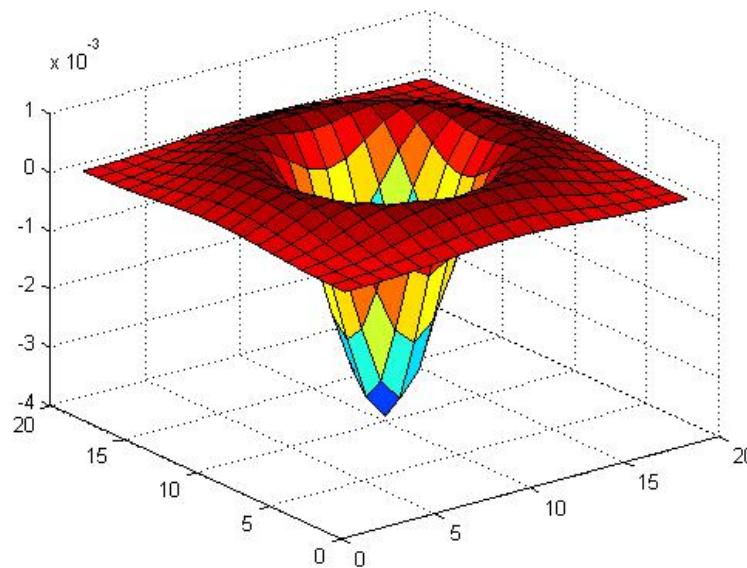
Laplacian of Gaussian: Circularly symmetric operator for blob detection in 2D



$$\nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2}$$

Blob detection in 2D

Laplacian of Gaussian: Circularly symmetric operator for blob detection in 2D

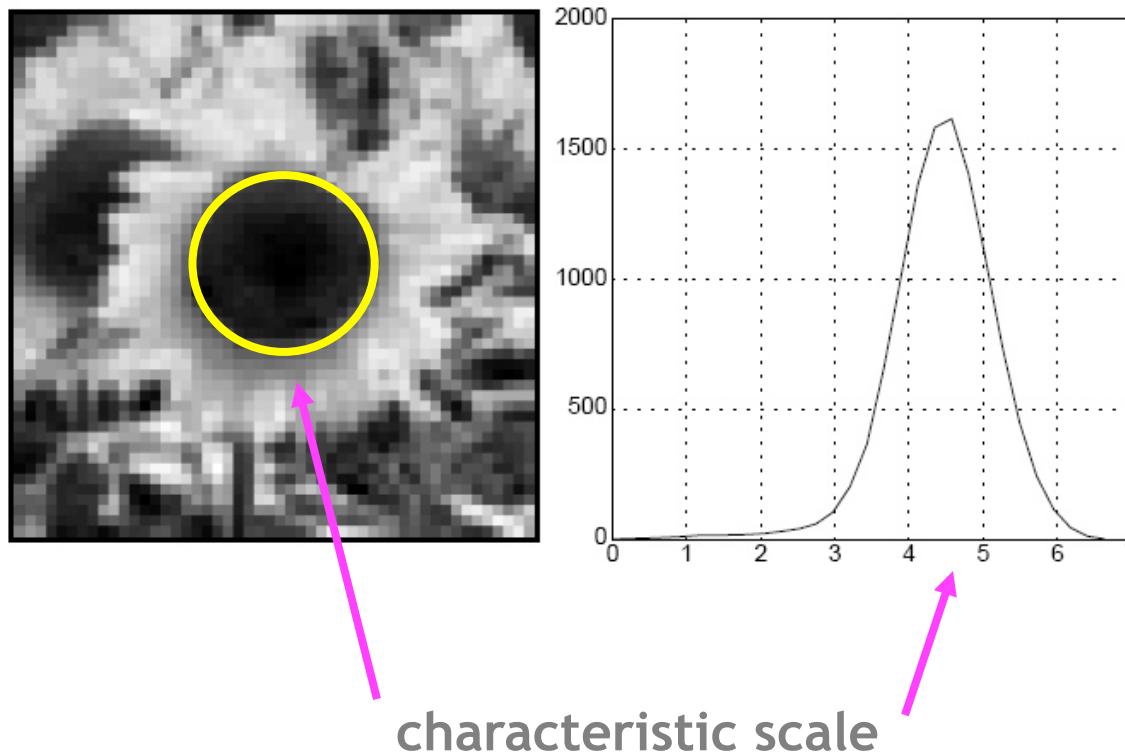


Scale-normalized:

$$\nabla_{\text{norm}}^2 g = \sigma^2 \left(\frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2} \right)$$

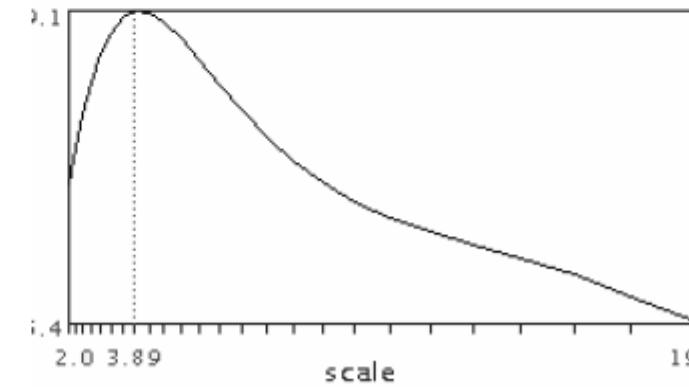
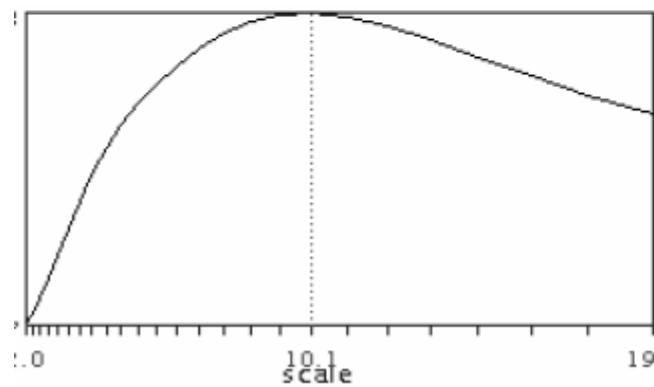
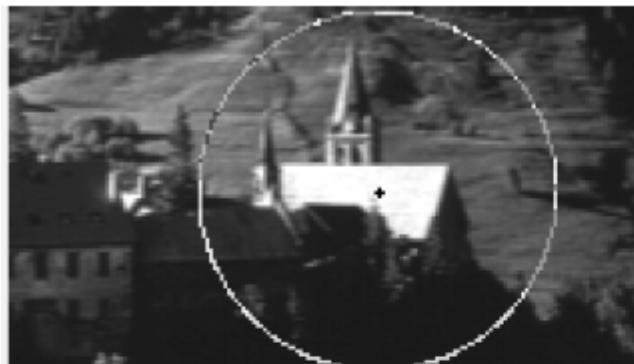
Scale selection

- Characteristic scale: peak of normalized Laplacian response



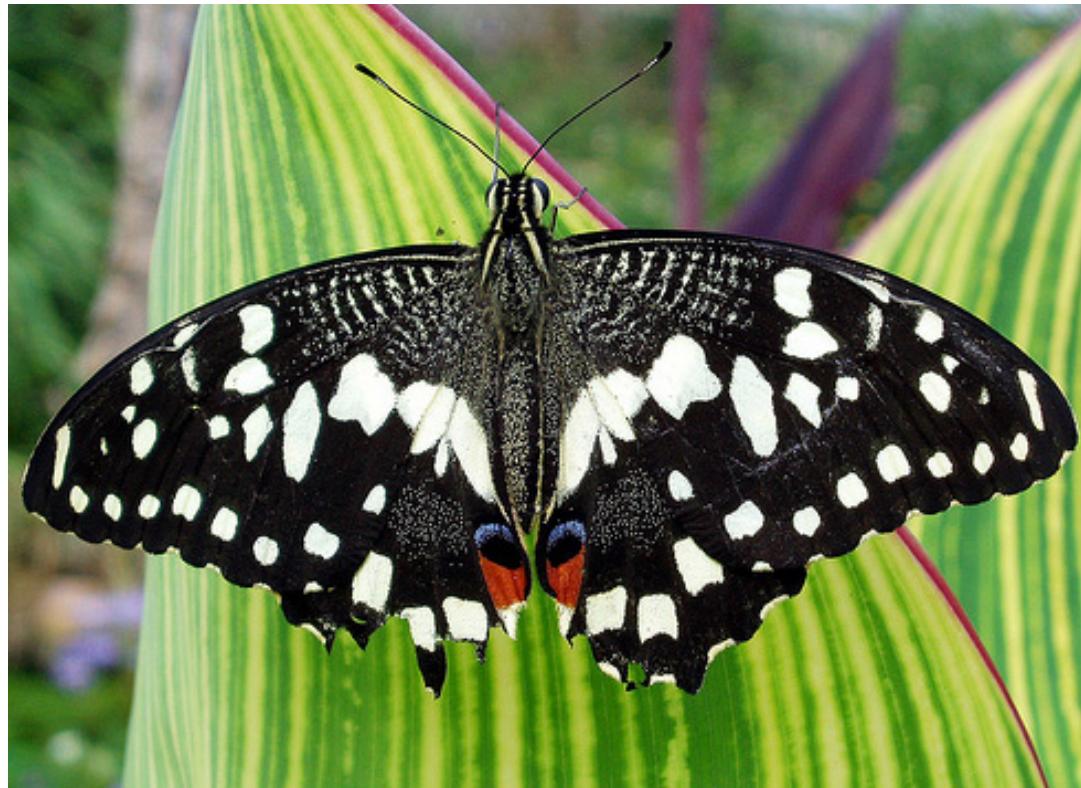
Tony Lindeberg: Feature Detection with Automatic Scale Selection. International Journal of Computer Vision 30(2): 79-116 (1998)
Tony Lindeberg: Edge Detection and Ridge Detection with Automatic Scale Selection. International Journal of Computer Vision 30(2): 117-156 (1998)

Scale invariance using scale selection

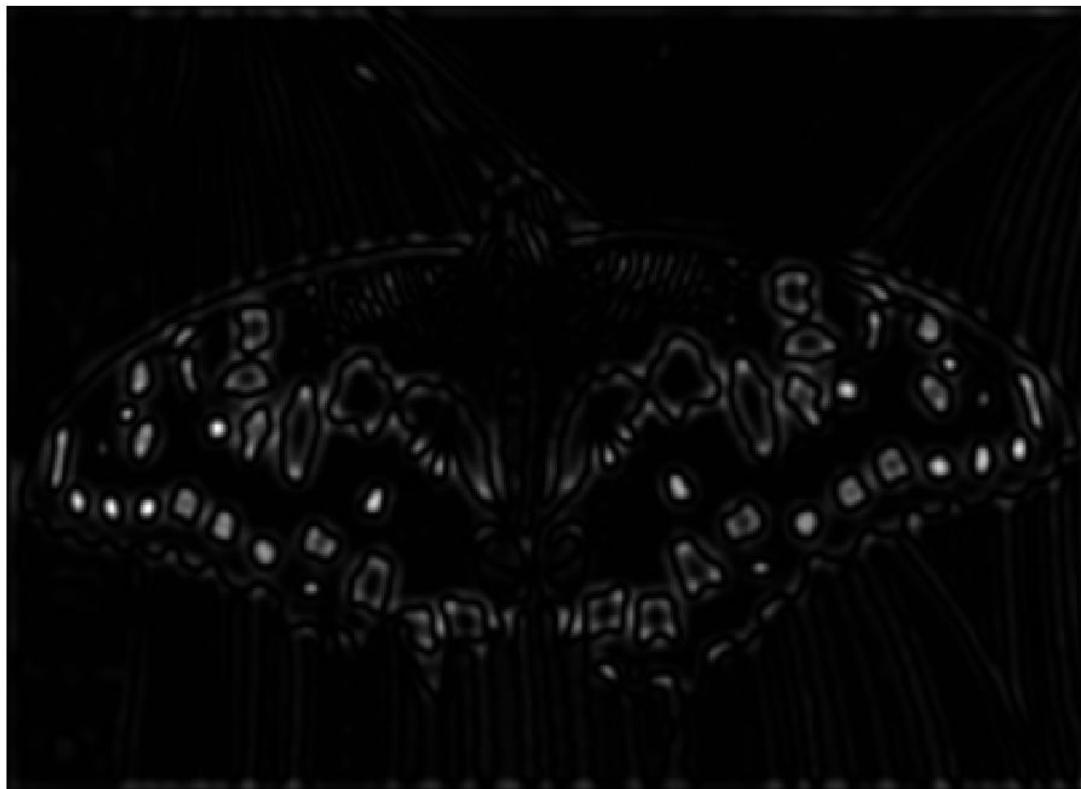


Laplacian

Scale-space blob detector: Example



Scale-space blob detector: Example



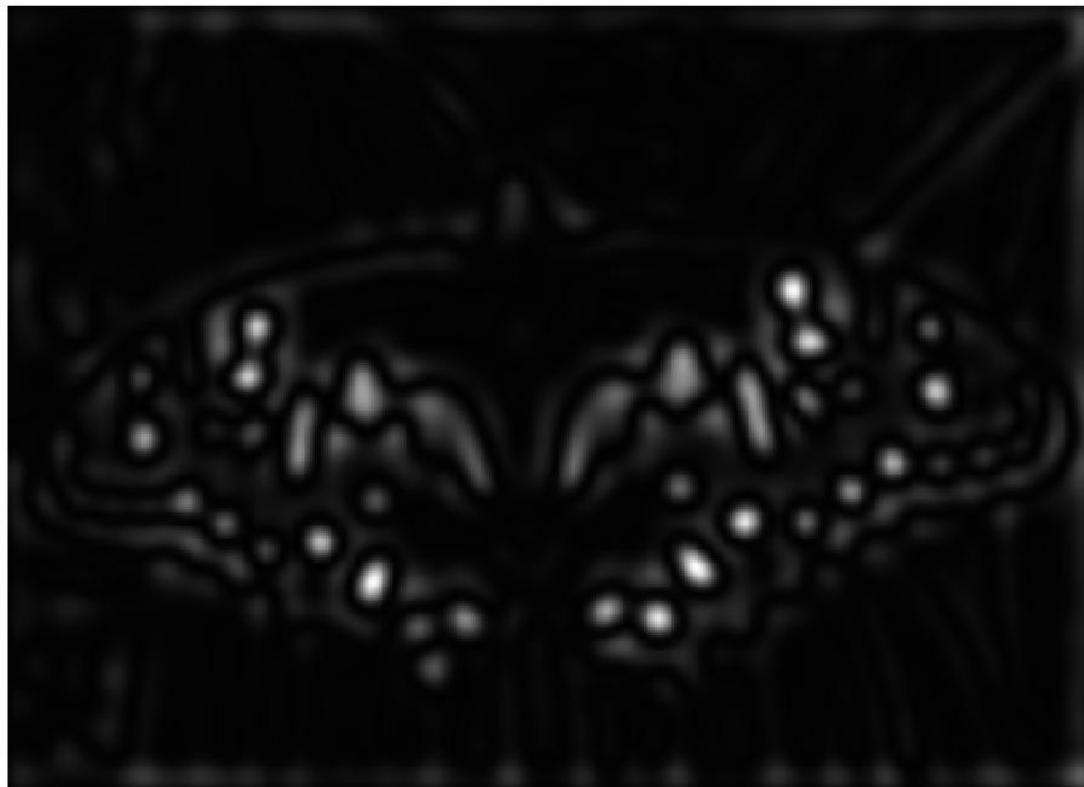
$\sigma = 3.1296$

Scale-space blob detector: Example



$\sigma = 4.8972$

Scale-space blob detector: Example



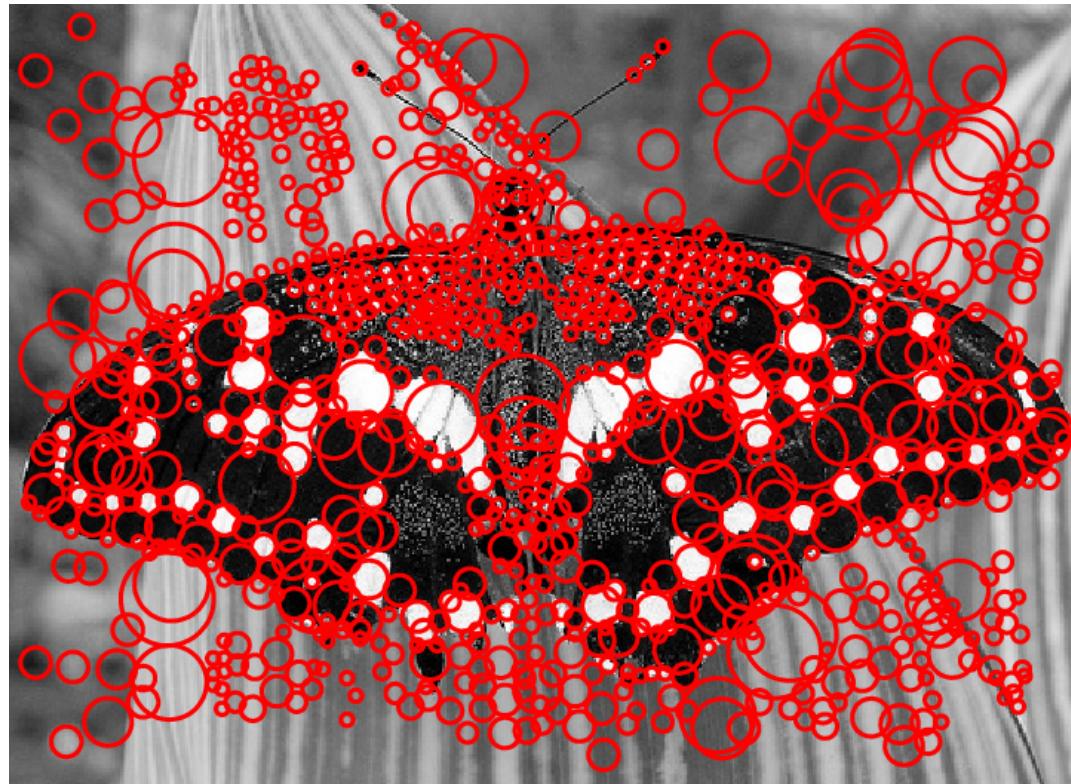
$\sigma = 7.6631$

Scale-space blob detector: Example



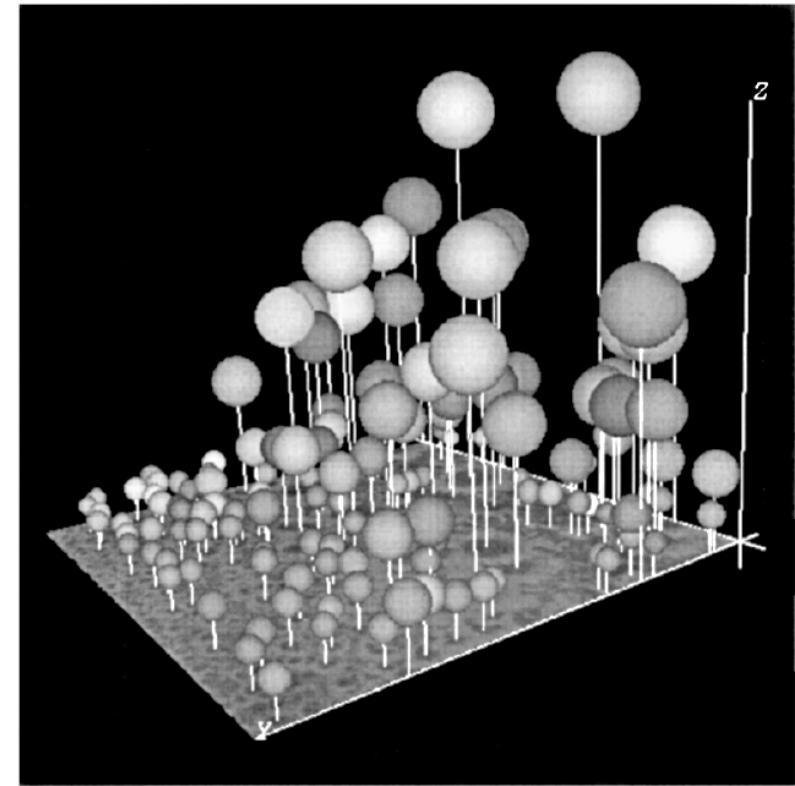
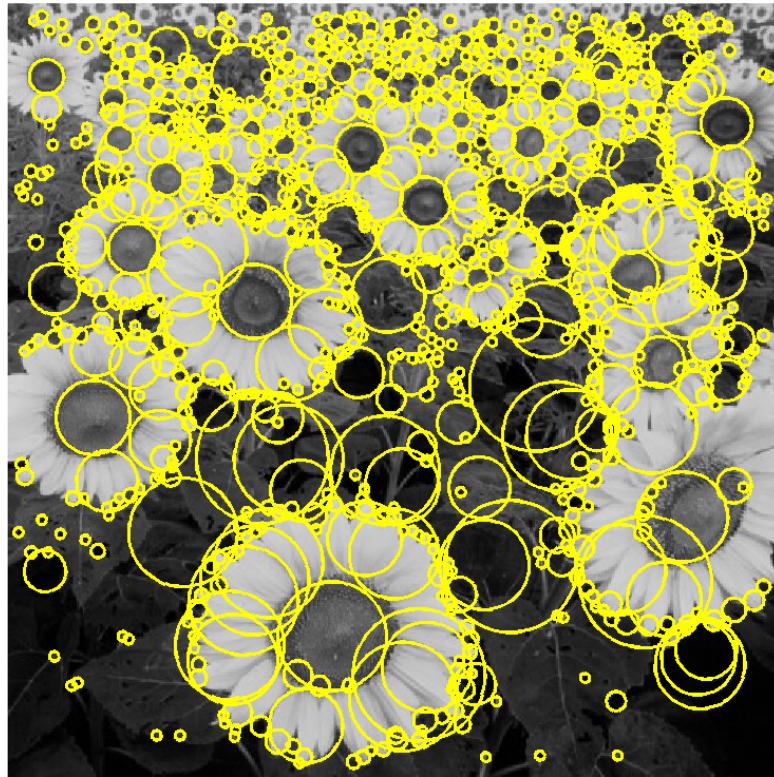
$\sigma = 11.9912$

Scale-space blob detector: Example



**Tony Lindeberg: Feature Detection with Automatic Scale Selection.
International Journal of Computer Vision 30(2): 79-116 (1998)**

Blob coordinates: (x,y,scale)



**Tony Lindeberg: Feature Detection with Automatic Scale Selection.
International Journal of Computer Vision 30(2): 79-116 (1998)**

Laplacian of Gaussian ~= Difference of Gaussian

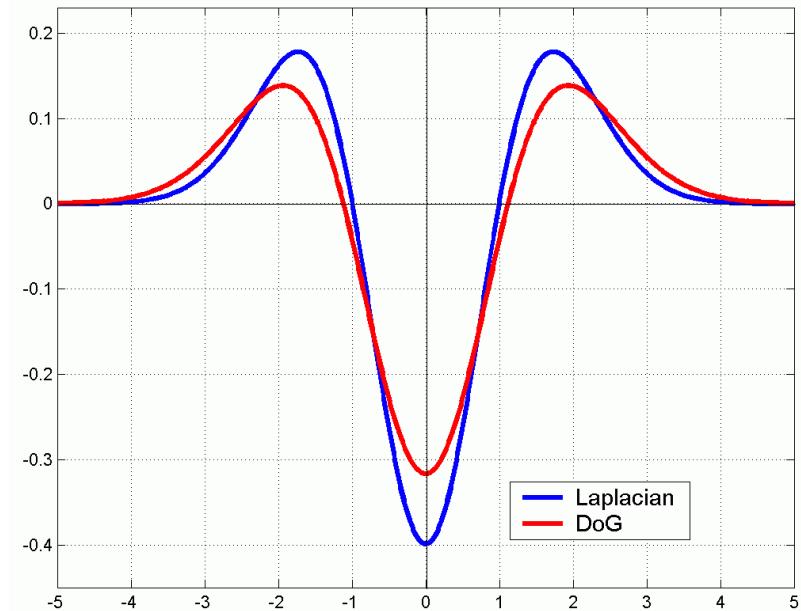
- We can efficiently approximate the Laplacian with a difference of Gaussians:

$$L = \sigma^2 \left(G_{xx}(x, y, \sigma) + G_{yy}(x, y, \sigma) \right)$$

(Laplacian)

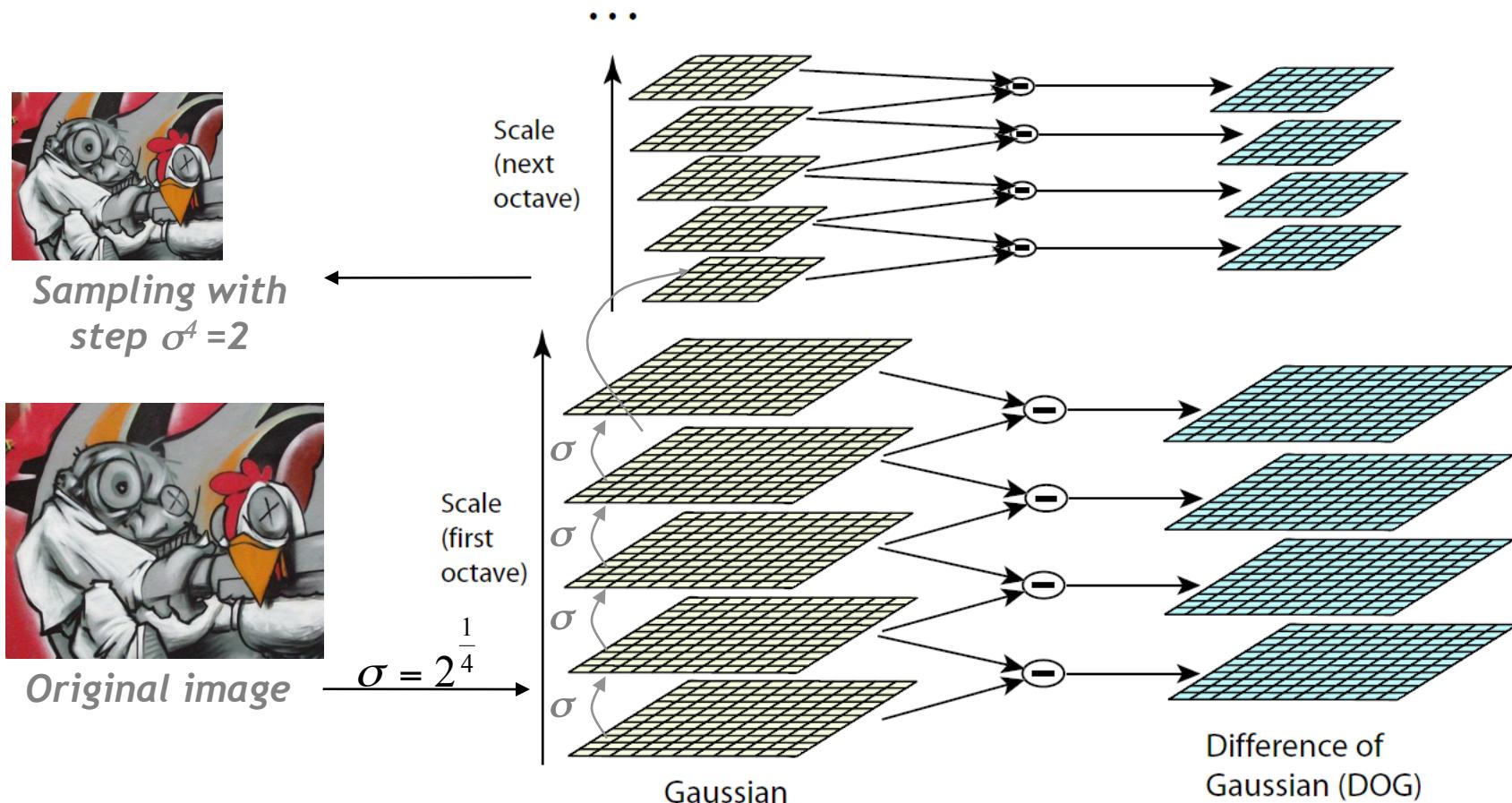
$$DoG = G(x, y, k\sigma) - G(x, y, \sigma)$$

(Difference of Gaussians)



Efficient Computation (SIFT)

- Computation in Gaussian scale pyramid



David G. Lowe: Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision 60(2): 91-110

Keypoint Detection (SIFT)



(a)



(b)



(c)

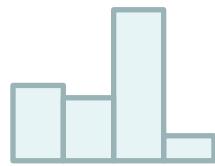
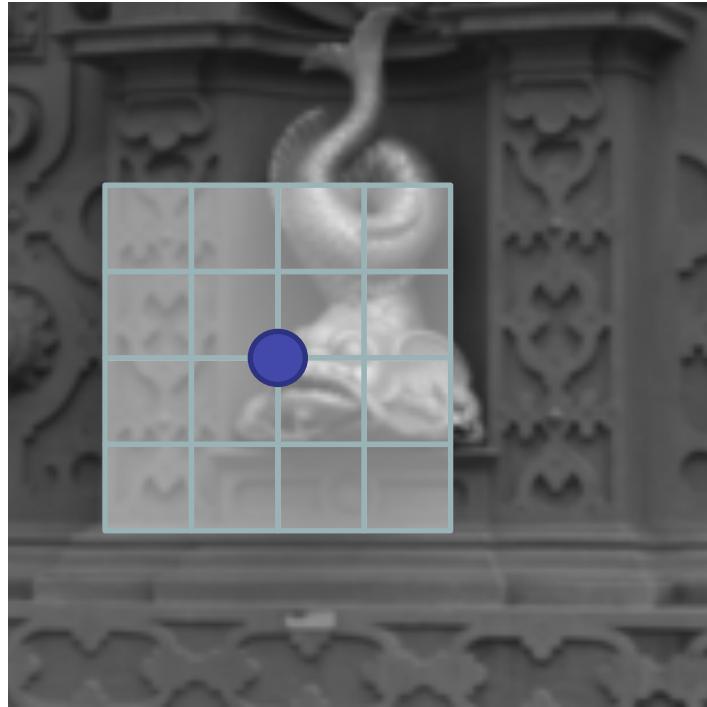


(d)

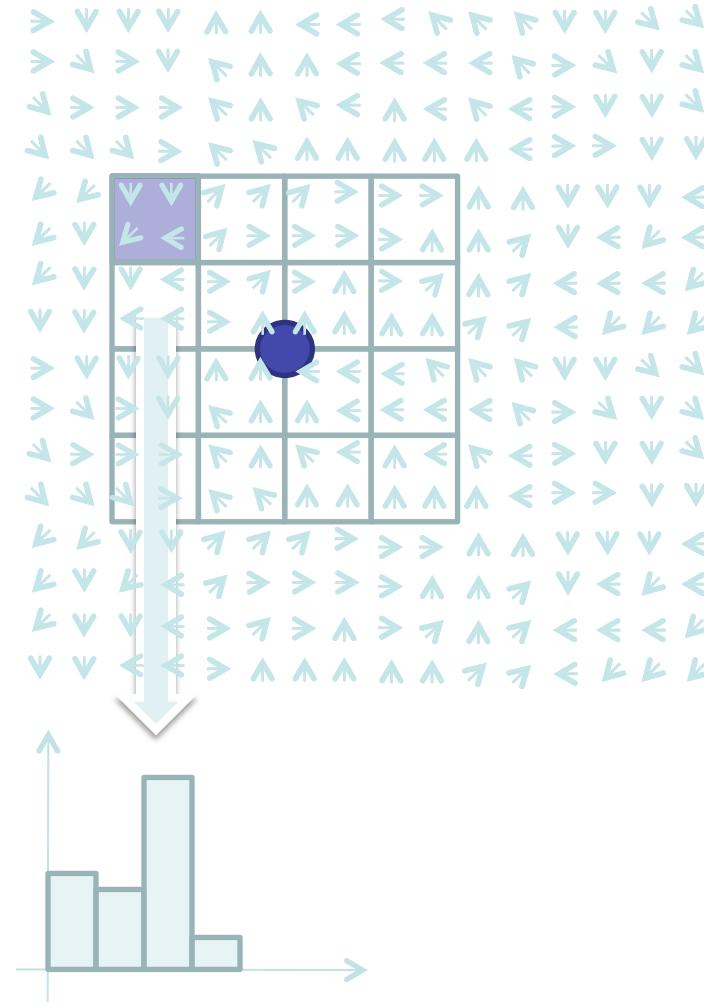
- (a) 233x189 image
- (b) 832 DoG extrema
- (c) 729 left after peak value threshold
- (d) 536 left after testing ratio of principle curvatures (removing edge responses)

David G. Lowe: Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision 60(2): 91-110

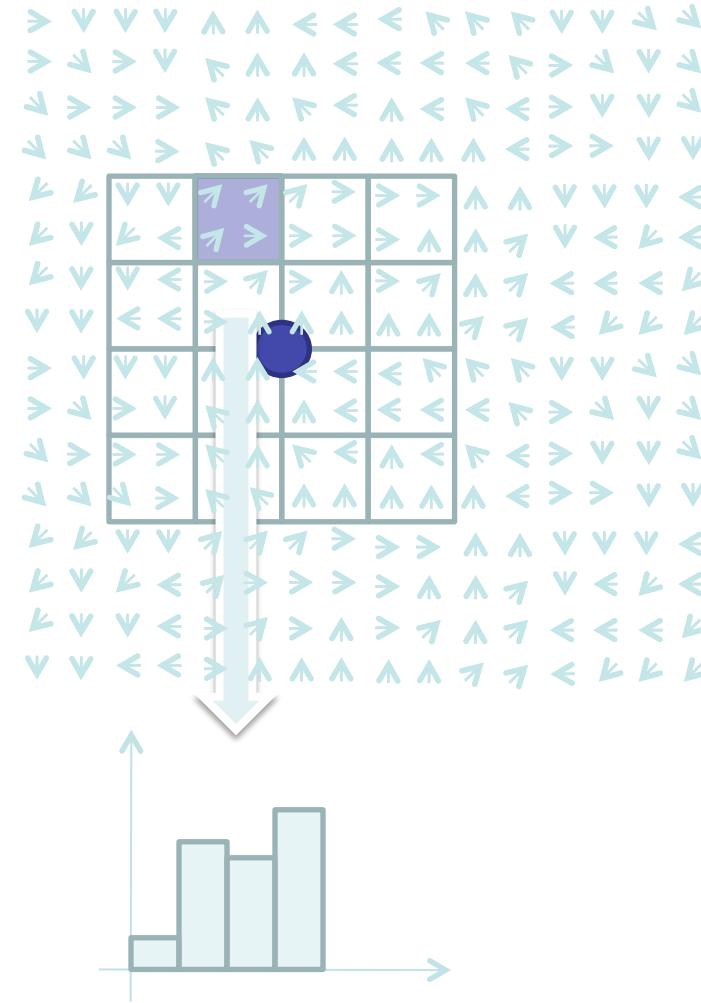
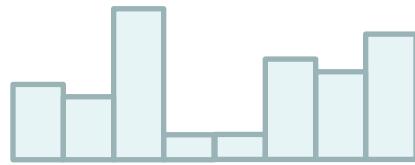
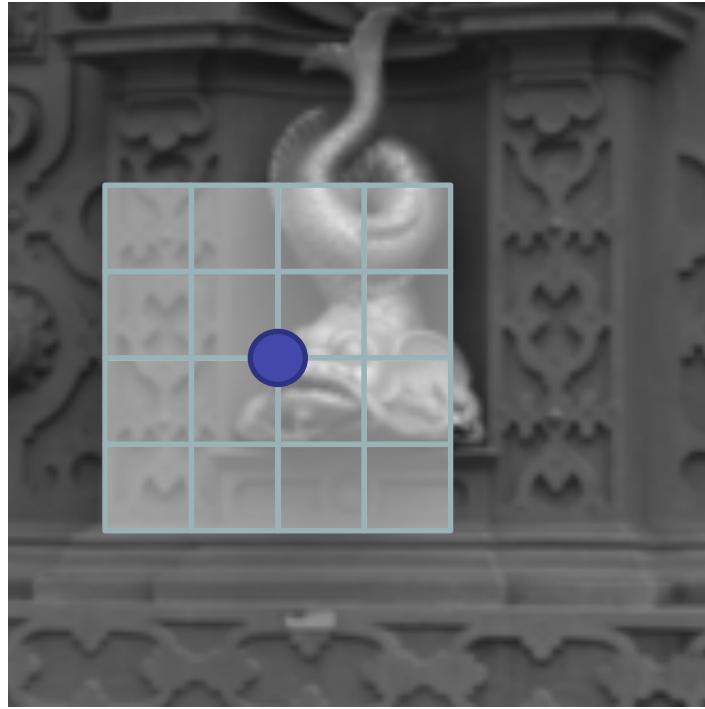
SIFT computation



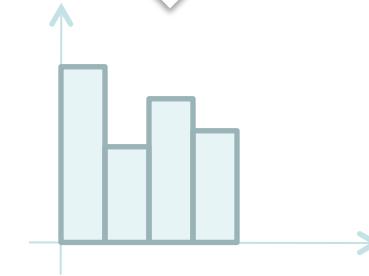
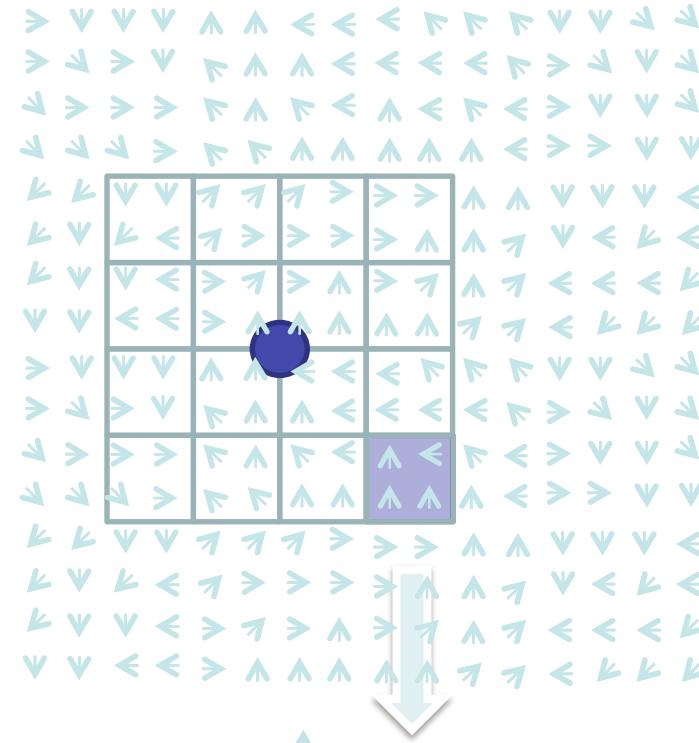
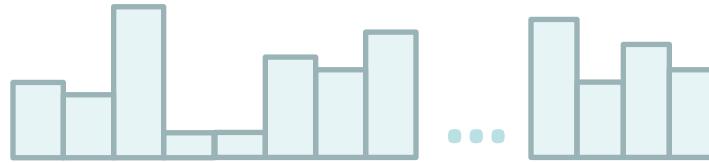
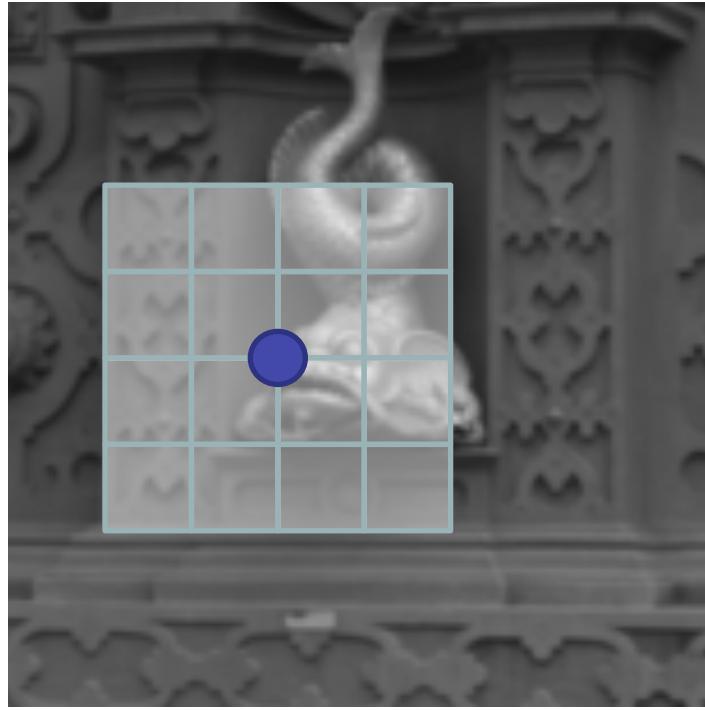
...



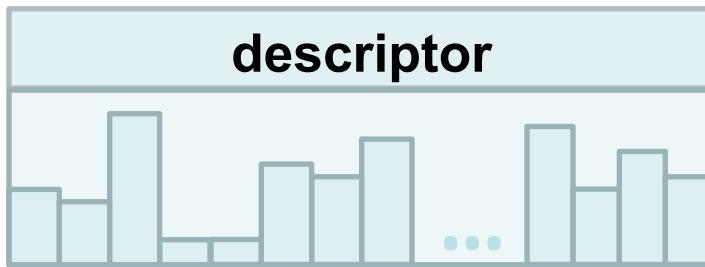
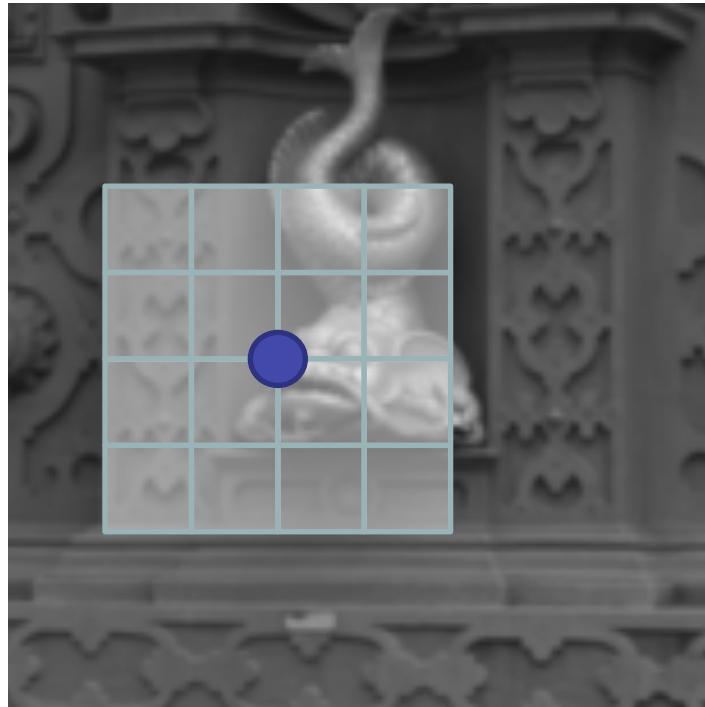
SIFT computation



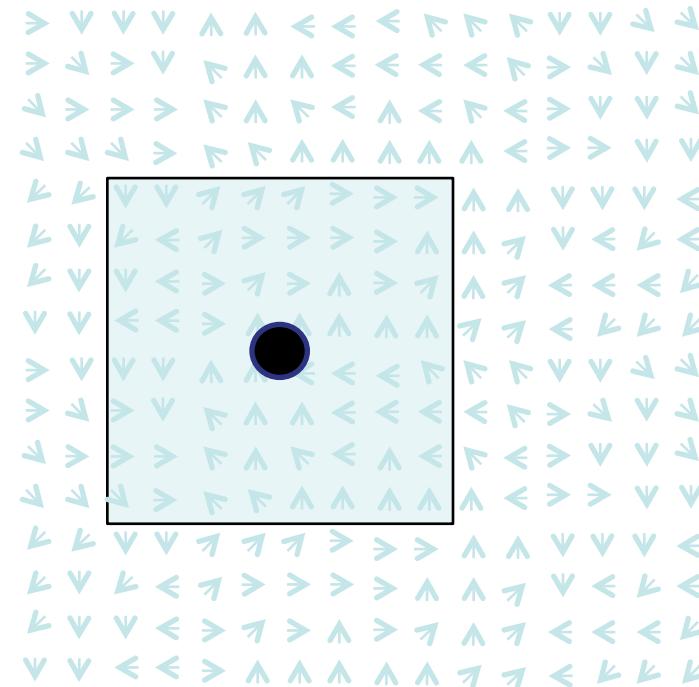
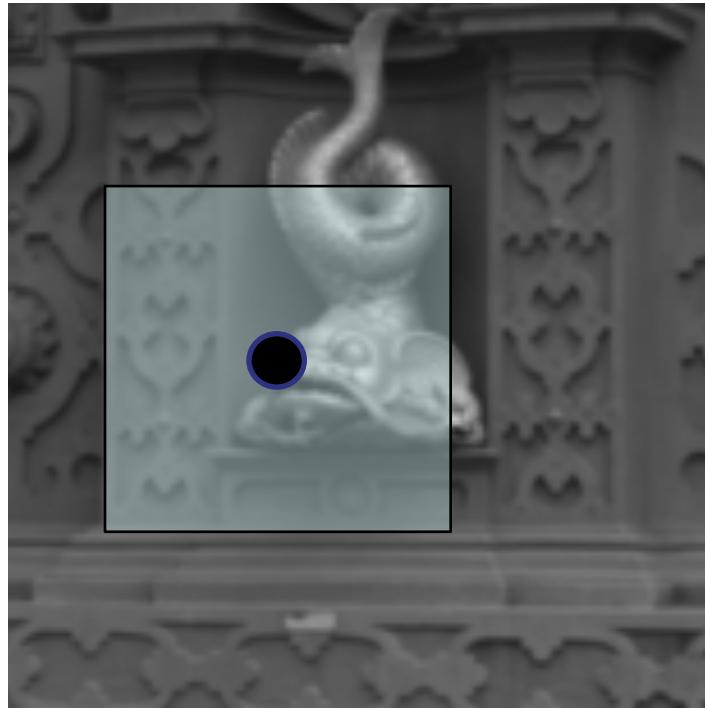
SIFT computation



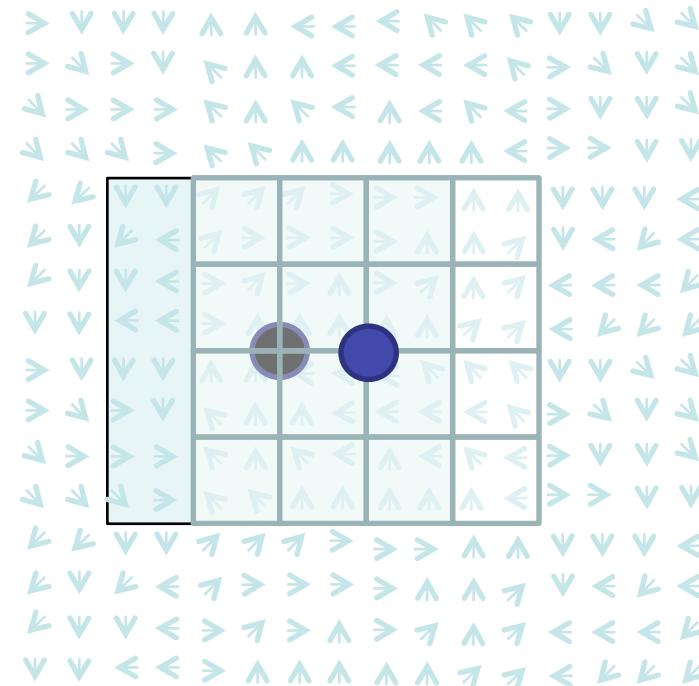
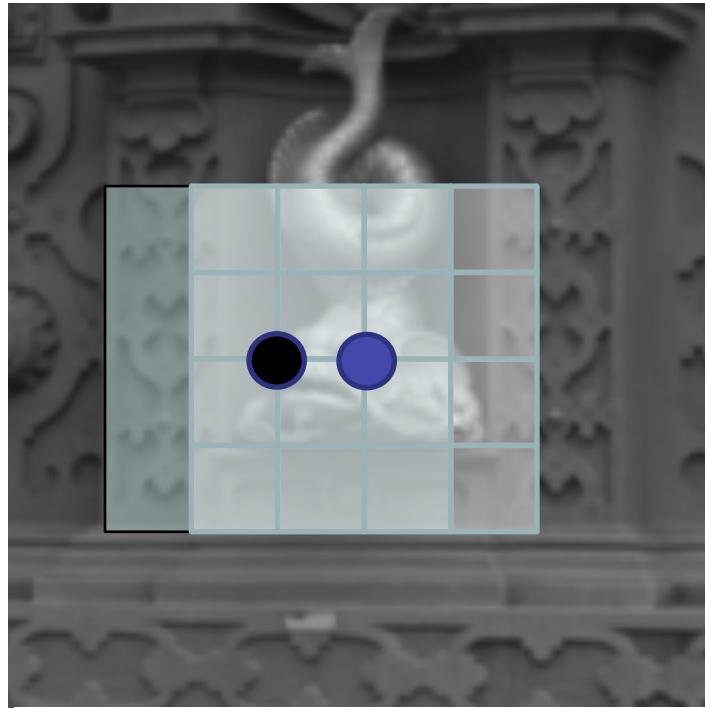
SIFT computation



SIFT computation

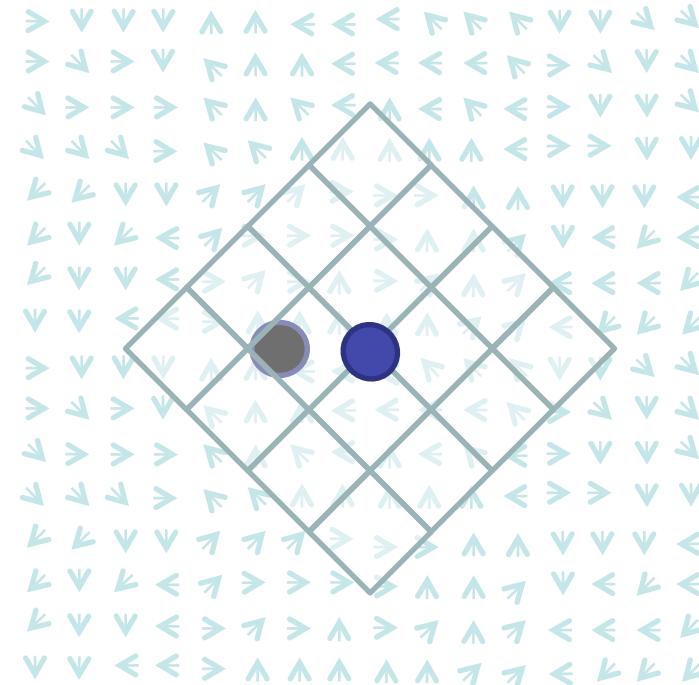
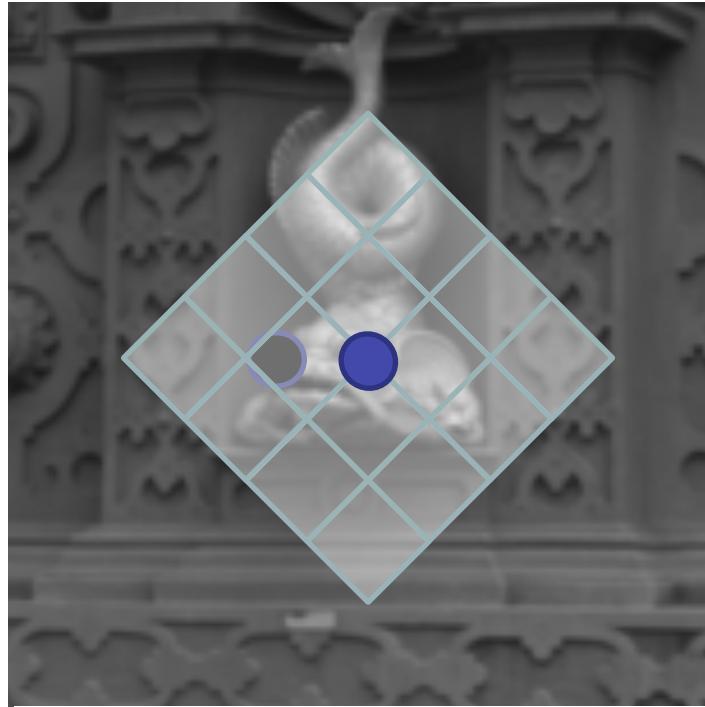


SIFT computation



Slide credit: E. Tola

SIFT computation

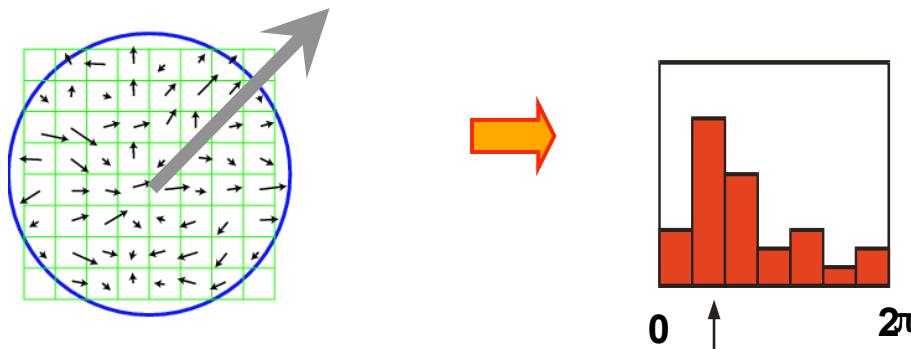


Slide credit: E. Tola

Scale-Invariant Feature Transform (SIFT) descriptor

Use location and characteristic scale given by blob detector

Estimate orientation from orientation histogram

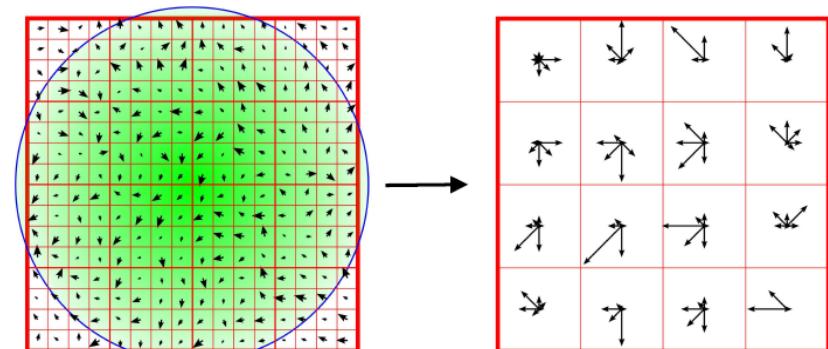


Break patch in 4×4 location blocks

8-bin orientation histogram per block

$$8 \times 4 \times 4 = 128\text{-D descriptor}$$

Normalize to unit norm

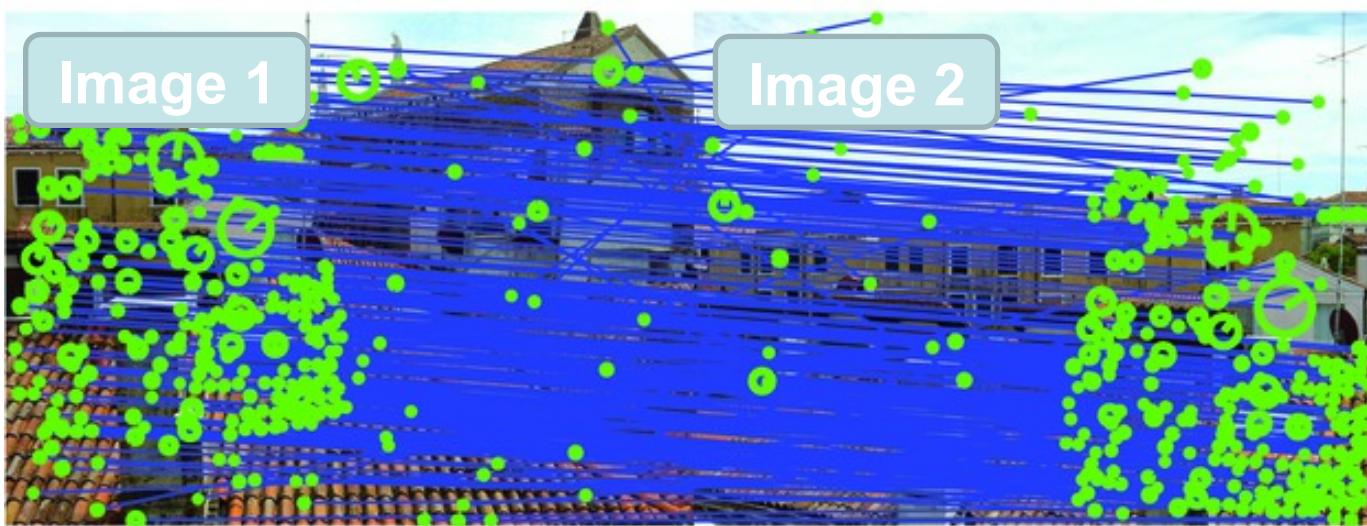


Invariance to: scale, orientation, multiplicative & additive changes

Application: Image Matching

Assumption: images undergo global deformations with a few degrees-of-freedom (e.g. scaling, rotation)

Correspondences of a few points suffice
(found e.g. with SIFT)



Open source implementation: www.vlfeat.org

The screenshot shows a Google Chrome browser window with the title bar "VLFeat - Tutorials". The address bar contains the URL "www.vlfeat.org/overview/tut.html". The page content is a tutorial section for VLFeat, featuring a sidebar with links to Home, Download, Documentation, Tutorials (which is selected), Applications, and Other. The main content area describes the tutorials and lists links for various features, clustering, and other topics.

This section features a number of tutorials illustrating some of the main algorithms implemented in VLFeat. The tutorials can be categorized into two classes. The first class of algorithms detect and describe image regions ([features](#)). The second class of algorithms ([cluster](#)) performs clustering on image data.

Features

- [Covariant detectors](#). An introduction to computing co-variance features like Harris-Affine.
- [Histogram of Oriented Gradients \(HOG\)](#). Getting started with this ubiquitous representation for object recognition.
- [Scale Invariant Feature Transform \(SIFT\)](#). Getting started with this popular feature detector / descriptor.
- [Dense SIFT \(DSIFT\) and PHOW](#). A state-of-the-art descriptor for image categorization.
- [Maximally Stable Extremal Regions \(MSER\)](#). Extracting MSERs from an image.
- [Image distance transform](#). Compute the image distance transform for fast part models and edge matching.

Clustering

- [Integer optimized k-means \(IKM\)](#). A quick overview of VLFeat fast k -means implementation.
- [Hierarchical k-means \(HIKM\)](#). Create a fast k -means tree for integer data.
- [Agglomerative Information Bottleneck \(AIB\)](#). Cluster discrete data based on the mutual information between the data and cluster centers.
- [Quick shift](#). An introduction which shows how to create superpixels using this quick mode seeking method.
- [SLIC](#). An introduction to SLIC superpixels.

Other

- [Pegasos SVM](#). Learn a binary classifier and check its convergence plotting the energy value.
- [Forests of kd-trees](#). Approximate nearest neighbor queries in high dimensions using an optimized forest of kd-trees.
- [Plotting functions for rank evaluation](#). Learn how to plot ROC, DET, and precision-recall curves.
- [MATLAB Utilities](#). A list of useful MATLAB functions bundled with VLFeat.

© 2007–13 The authors of VLFeat

Further reading (literature ‘seeds’)

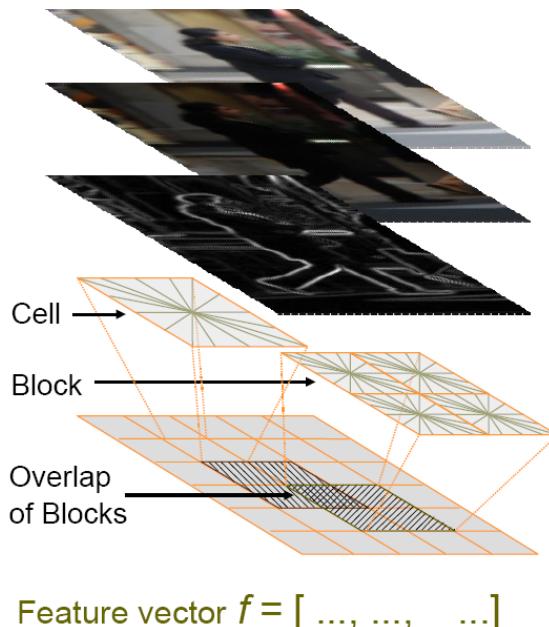
- **Compact Codes & Large-scale Retrieval**
 - J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. ICCV, 2003.
 - Nister, D., Stewenius, H.: Scalable recognition with a vocabulary tree. CVPR. (2006)
 - M. Perdoch, O. Chum, and J. Matas. Efficient representation of local geometry for large scale object retrieval. In Proc. CVPR, 2009
 - H. Jegou, M. Douze, C. Schmid, and P. Perez. Aggregating local descriptors into a compact image representation. CVPR, 10
 - A. Babenko and V. Lempitsky, The Inverted Multi-Index, CVPR 12
 - R. Arandjelović, A. Zisserman, All about VLAD, CVPR 2013
- **Fast/Compact Descriptors**
 - M. Calonder, V. Lepetit, C. Strecha, and P. Fua, BRIEF: Binary Robust Independent Elementary Features, (ECCV), 2010.
 - T. Trzcinski, M. Christoudias, P. Fua, and V. Lepetit, Boosting Binary Keypoint Descriptors. (CVPR), 2013.
 - SURF, FAST, ORB, FREAK,...

Further reading (literature ‘seeds’)

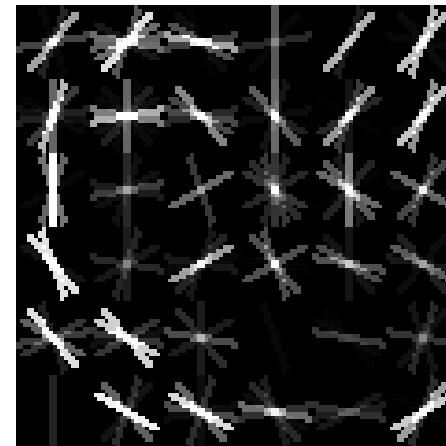
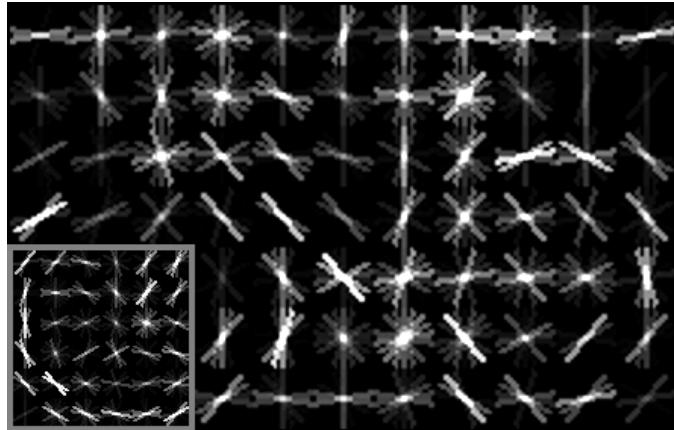
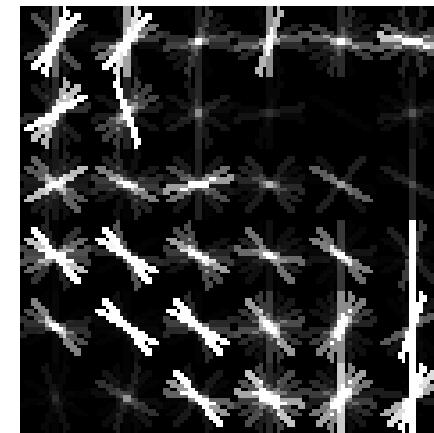
- **Feature encoding**
 - Improving the fisher kernel for large-scale image classification, F. Perronnin, J. Sánchez, and T. Mensink. In Proc. ECCV, 2010.
 - The devil is in the details: an evaluation of recent feature encoding methods, K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman, BMVC, 2011
 - Sparse Kernel Approximations for Efficient Classification and Detection, A. Vedaldi and A. Zisserman, in Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2012
- **Descriptor Learning**
 - Simon A. J. Winder, Matthew Brown: Learning Local Image Descriptors. CVPR 2007
 - S. Winder, G. Hua, and M. Brown. Picking the best daisy. In Proc. CVPR, 2009.
 - Descriptor Learning for Efficient Retrieval, J. Philbin, M. Isard, J. Sivic, A. Zisserman, ECCV 10
 - K. Simonyan, A. Vedaldi, and A. Zisserman. Descriptor learning using convex optimisation. In Proc. ECCV, 2012

Histogram of Orientated Gradients (HOG) descriptor

- Dalal and Triggs, ICCV 2005
 - Like SIFT descriptor, but for arbitrary box aspect ratio, and computed over all image locations and scales
 - Highly accurate detection using linear classifier



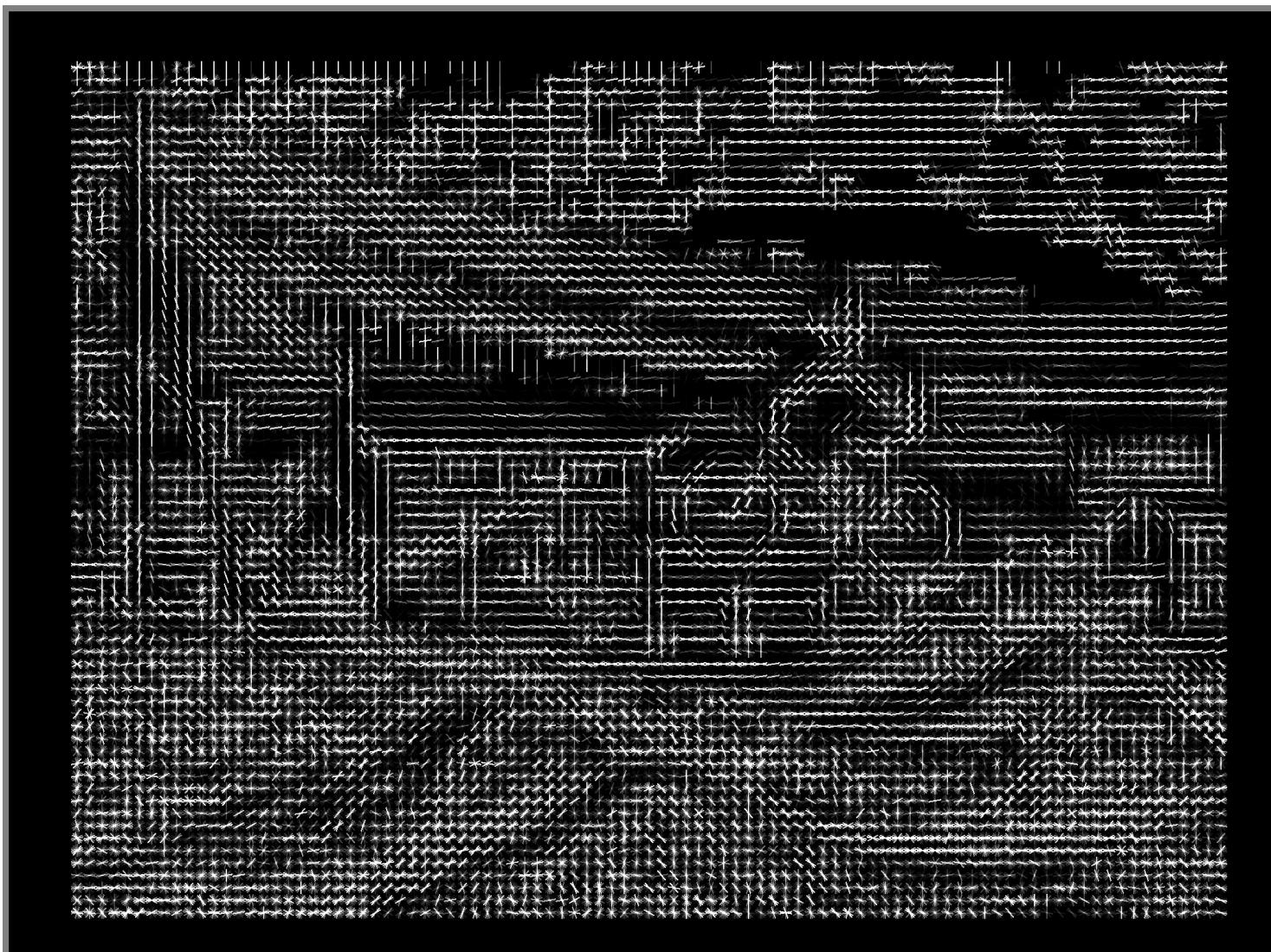
Part score computation

 $\mathbf{w}[y]$  $\mathbf{h}[x + y]$

$$s[x] = \sum_y \langle \mathbf{h}[x + y], \mathbf{w}[y] \rangle$$

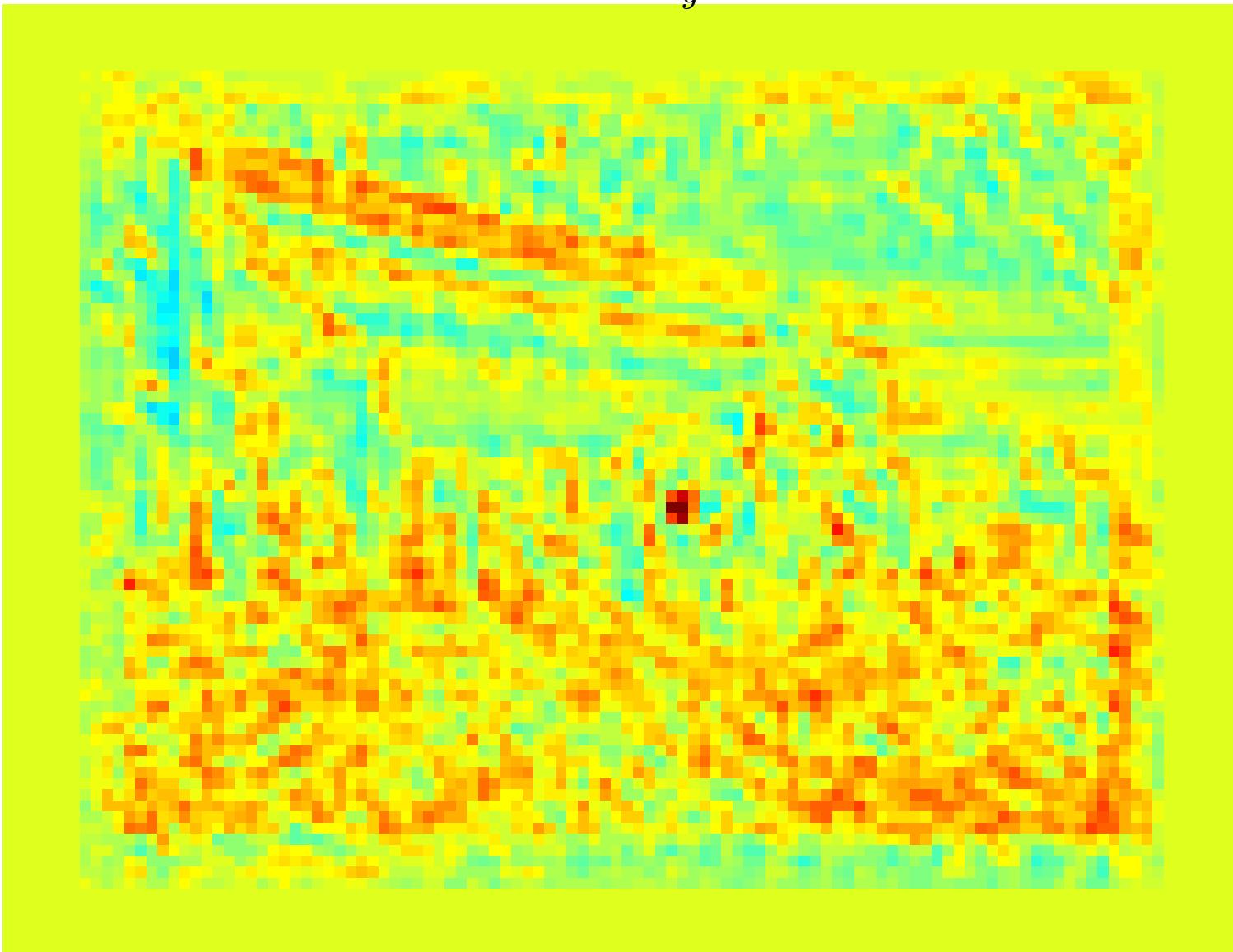
Part score

$$\mathbf{h}[x] \quad s[x] = \sum_y \langle \mathbf{h}[x+y], \mathbf{w}[y] \rangle$$

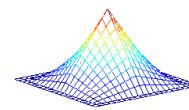


Part score

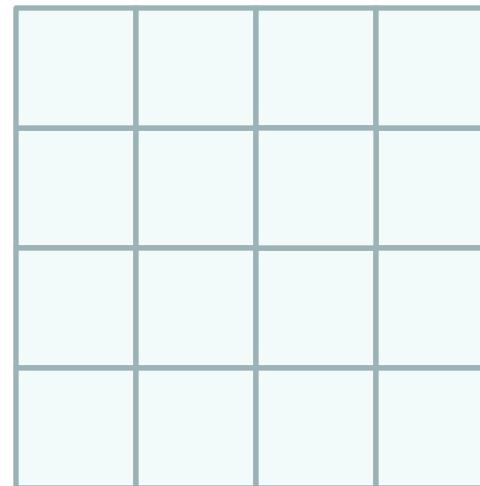
$$\mathbf{h}[x] \quad s[x] = \sum_y \langle \mathbf{h}[x + y], \mathbf{w}[y] \rangle$$



SIFT-> DAISY

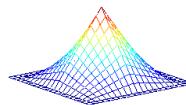
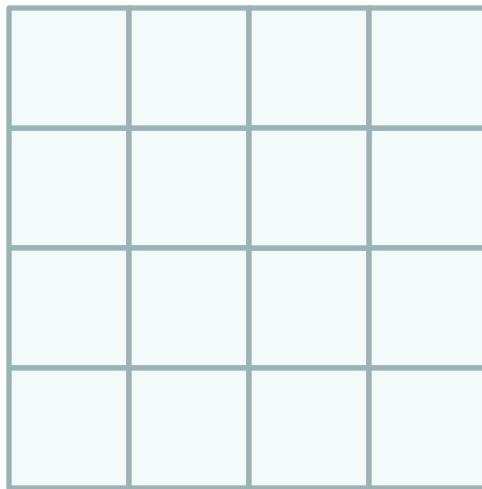
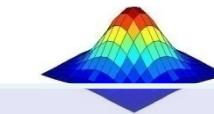
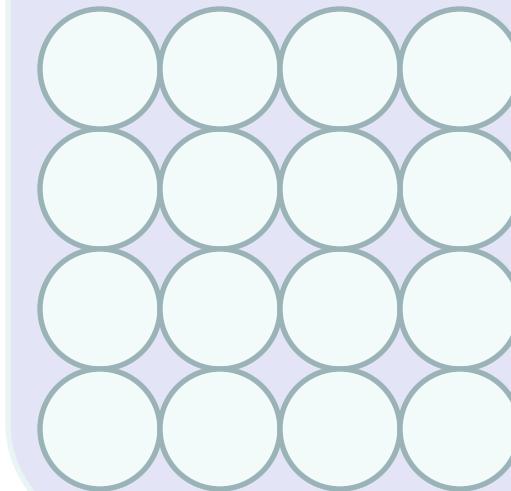
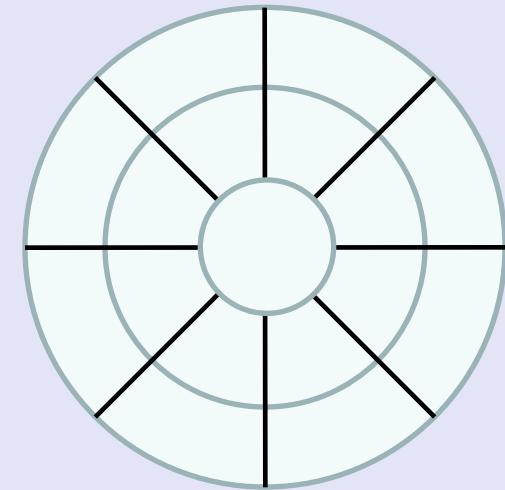


SIFT

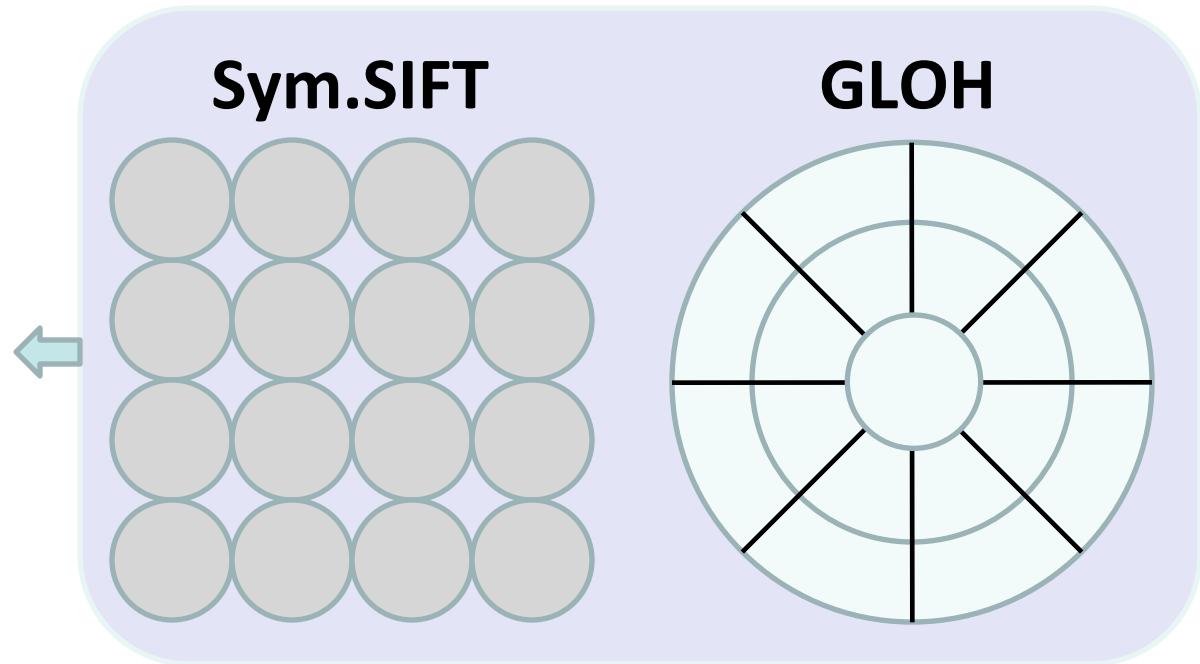
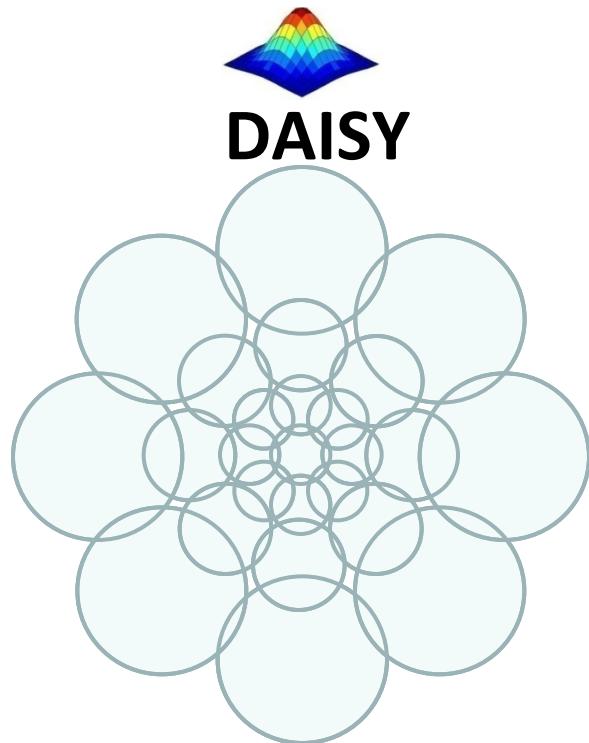


- + Good
- Not suitable for dense computation

SIFT-> DAISY

**SIFT****+ Good****- Not suitable for dense computation****Sym.SIFT****+ Gaussian Kernels : Suitable for Dense Computation****GLOH*****+ Good Performance****- Not suitable for dense computation**

SIFT-> DAISY

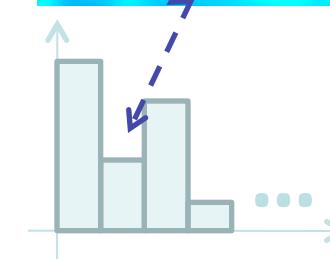
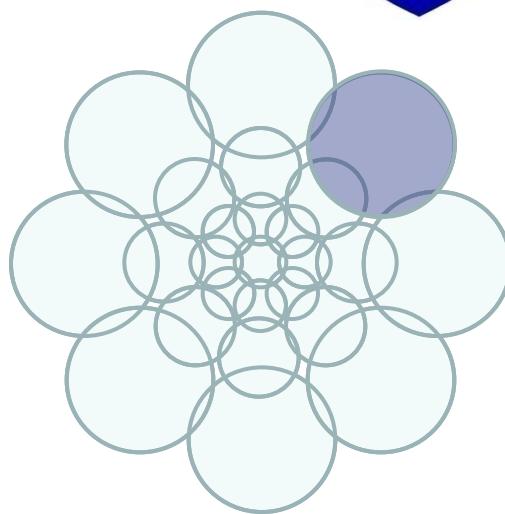
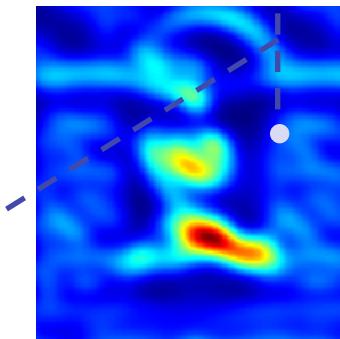
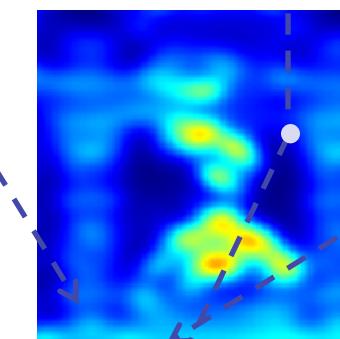
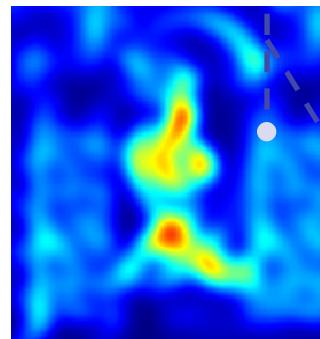
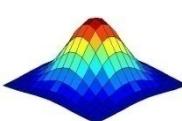
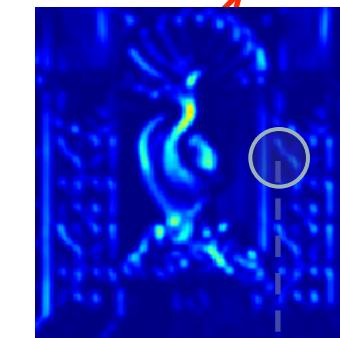
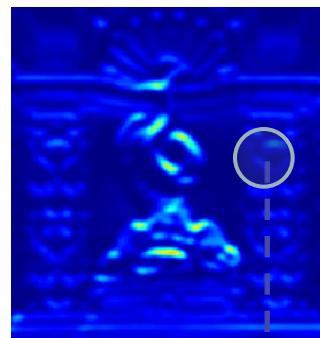
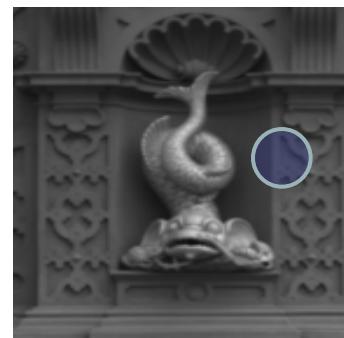


- Suitable for dense computation**
- + Improved performance:**
 - + Precise localization**
 - + Rotational Robustness**

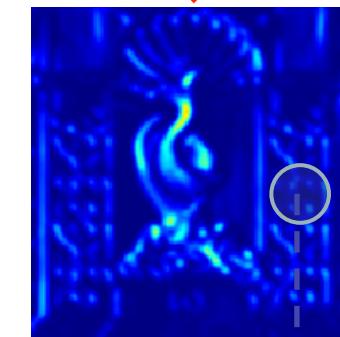
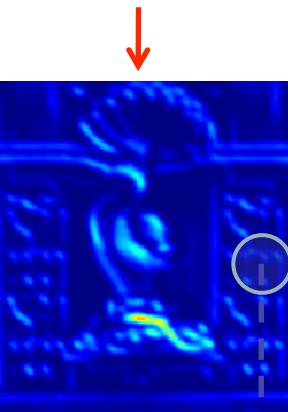
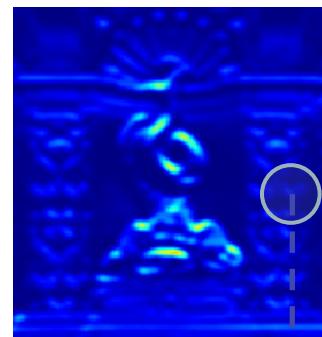
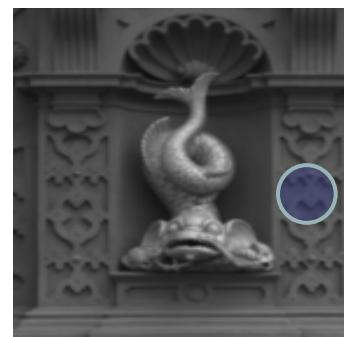
+ Suitable for Dense Computation

+ Good Performance
- Not suitable for dense computation

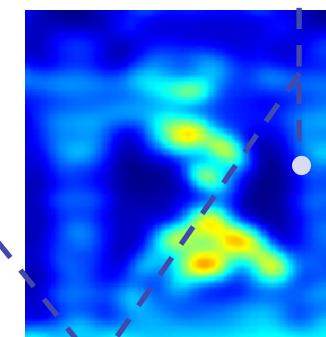
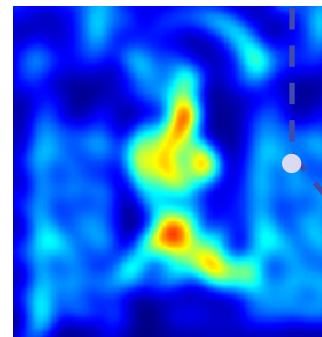
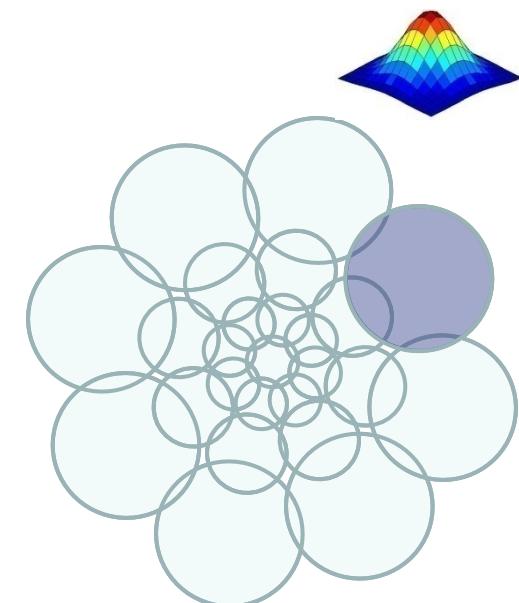
Daisy computation →



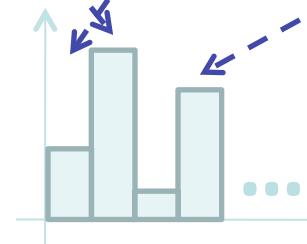
Daisy computation →



Three light blue dots indicating the continuation of the feature maps.

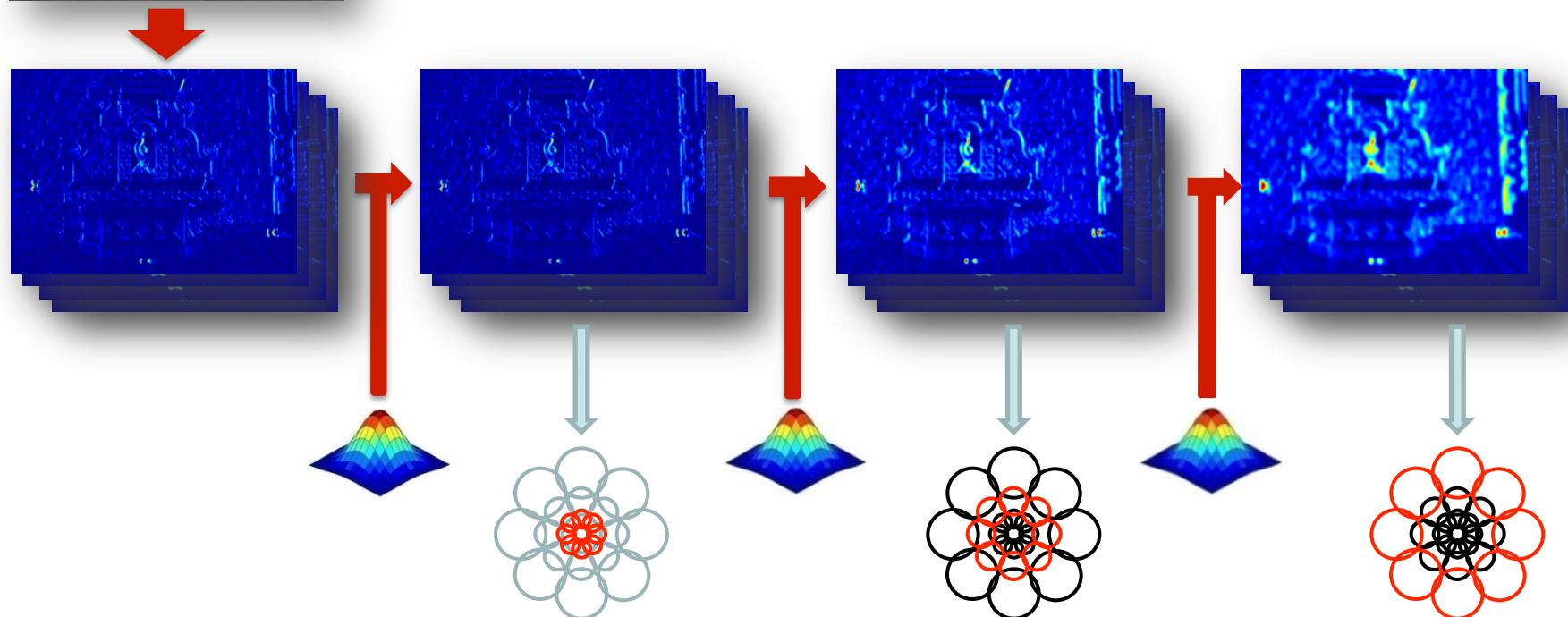


Three light blue dots indicating the continuation of the feature maps.



Daisy computation

DAISY : 5s
SIFT : 250s



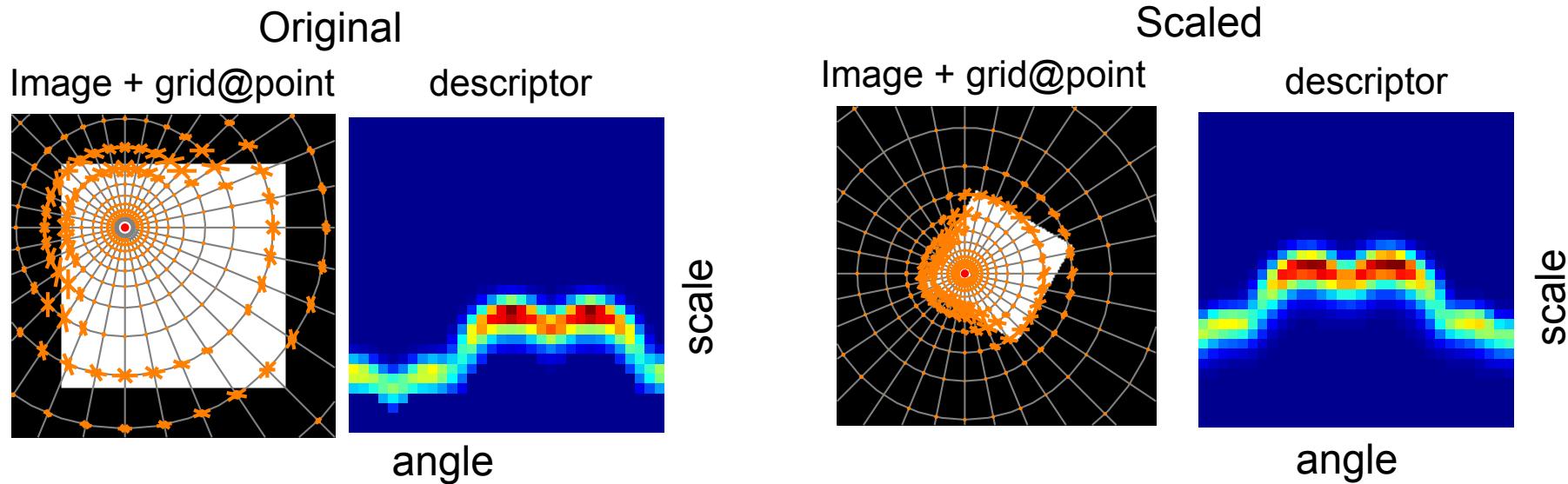
- Rotating the descriptor only involves reordering the histograms.
- The computation mostly involves 1D convolutions, which is fast.

Scale- and rotation- invariance & Fourier

Fact 1: *Signal translation does not affect the signal's Fourier Transform Magnitude:*

$$f[i - n_i, j - n_j] \xrightarrow{\mathcal{F}} F \exp\left(-j\left(n_i \frac{2\pi}{N} + n_j \frac{2\pi}{K}\right)\right) \xrightarrow{| \cdot |} |F|$$

Fact 2: log-polar sampling *turns image scaling and rotation to translation*:

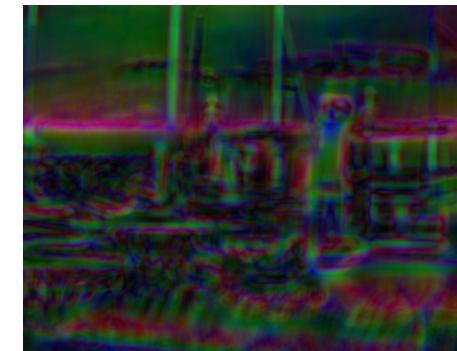
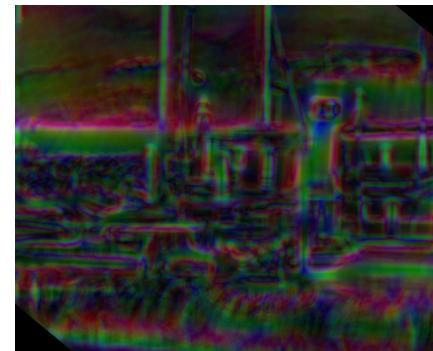
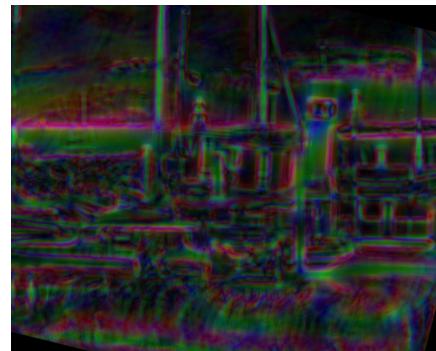
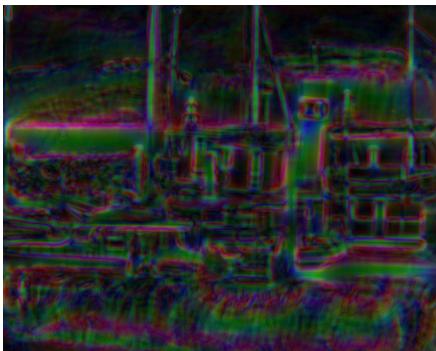
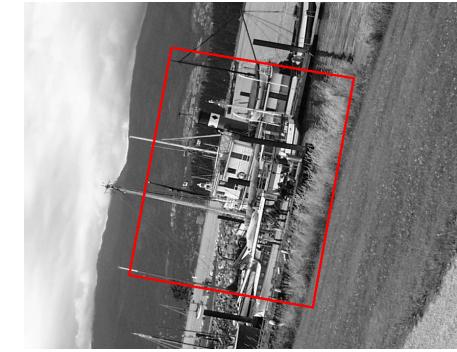


Fact 1+2: the *Fourier Transform Modulus of log-polar descriptors is invariant*

I. Kokkinos and A. Yuille, **Scale Invariance without Scale Selection**, CVPR, 2008.

D. Casasent and D. Psaltis, **Rotation and scale-invariant optical correlation**, Applied Optics, 1976

Dense Scale-Invariant Descriptors



I. Kokkinos and A. Yuille, Scale Invariance without Scale Selection, CVPR, 2008.
D. Casasent and D. Psaltis, Rotation and scale-invariant optical correlation, Applied Optics, 1976



Lecture summary

Introduction to the class

Introduction to the problem of classification

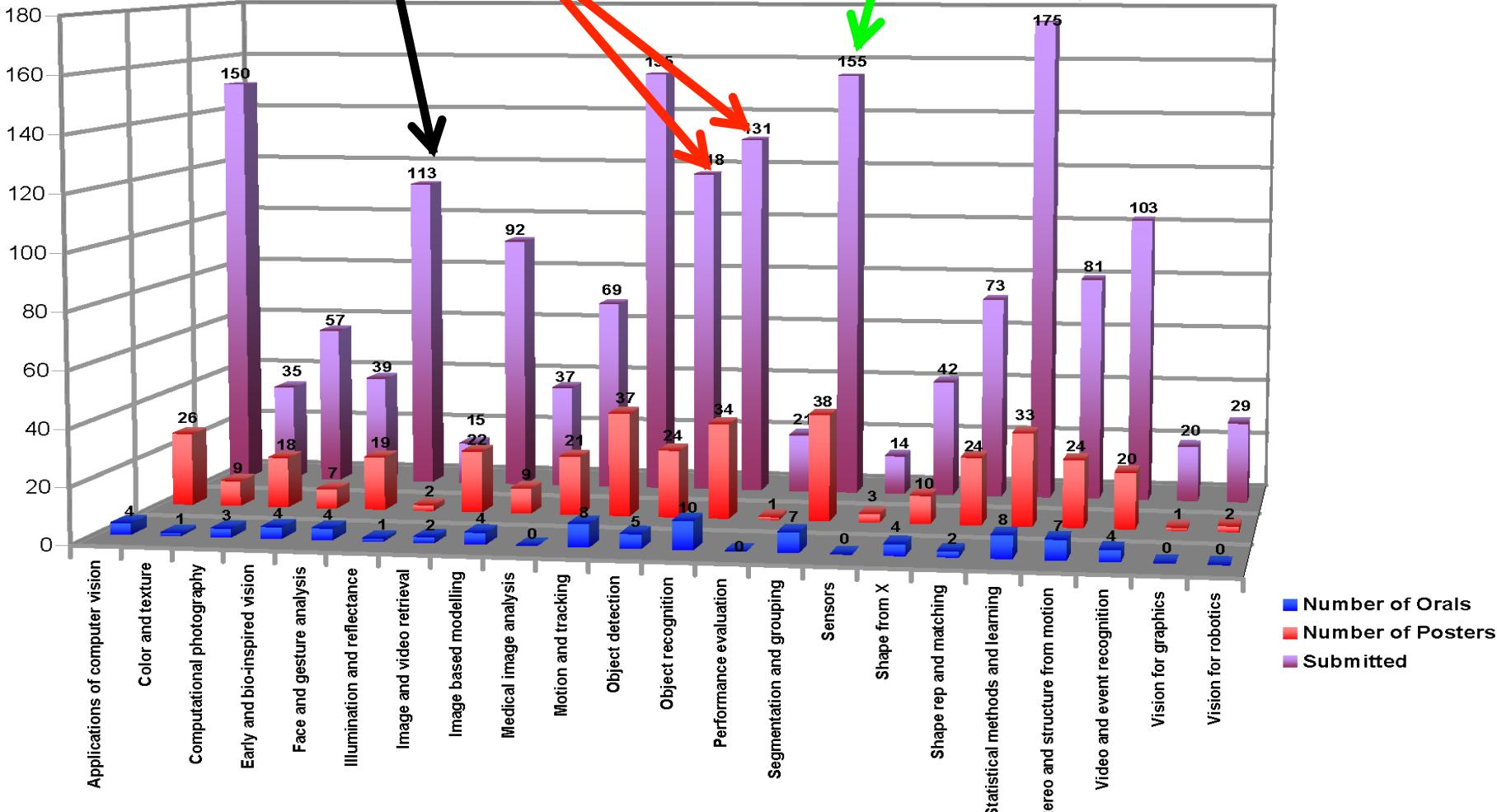
Linear classifiers

Image-based features

Who will need this class?

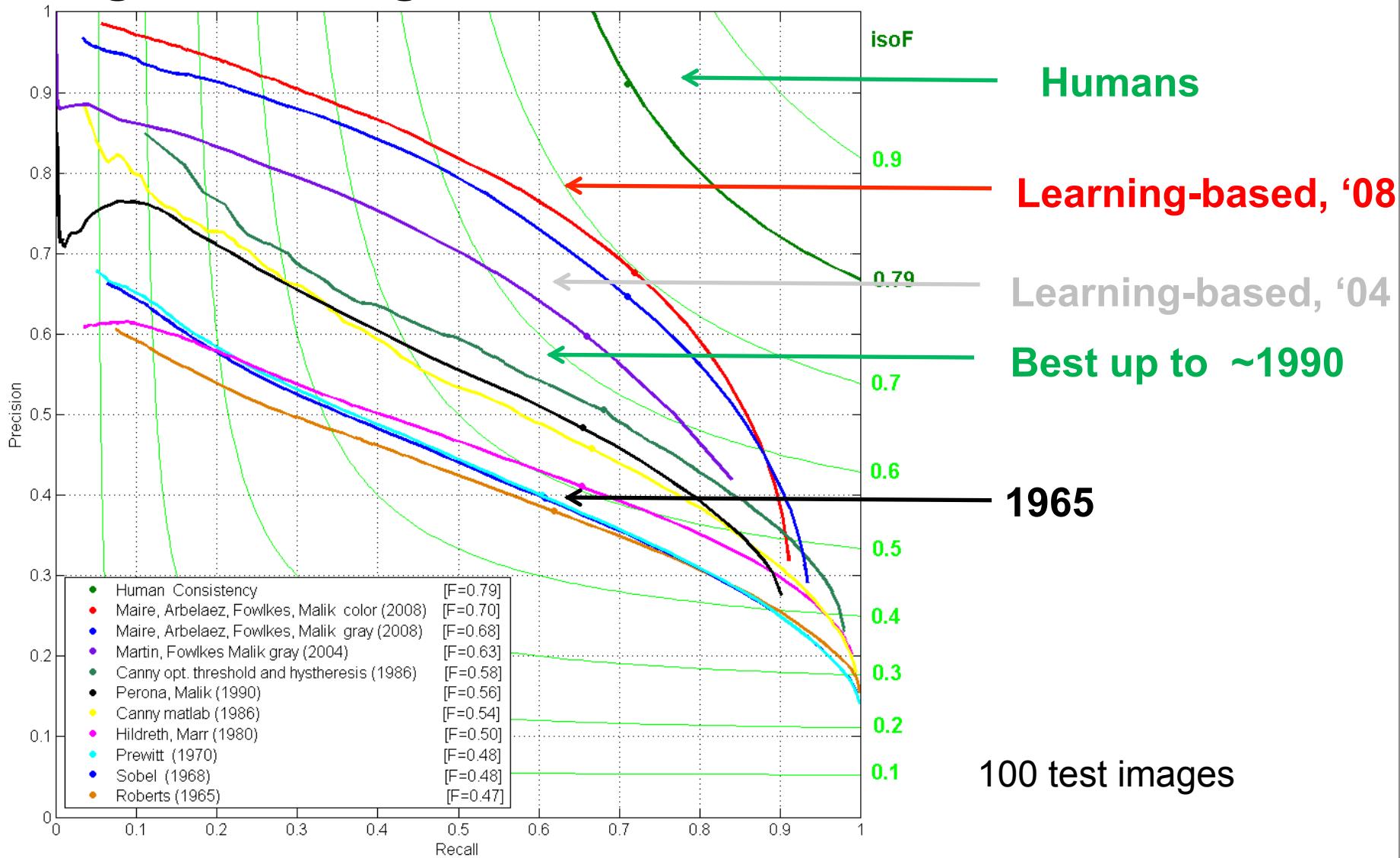
Faces Recognition

Segmentation Learning



Submission/Acceptance Statistics from CVPR 2010

Progress during the last 4 decades





Lecture summary

Introduction to the class

Introduction to the problem of classification

Linear classifiers

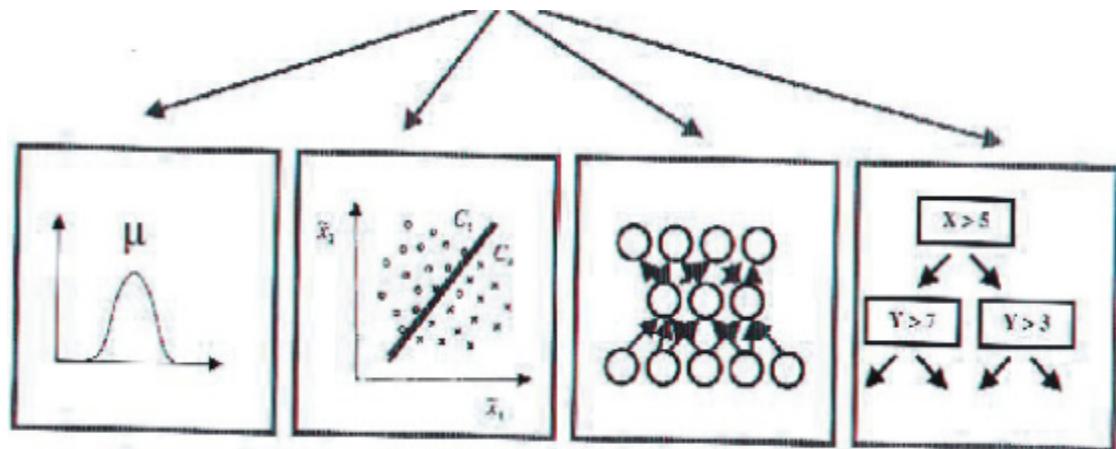
Image-based features

Classifier function

- Input-output mapping

- Output: y
- Input: x
- Method: f
- Parameters: w

$$y = f_w(x) \quad (= f(x, w))$$



- Aspects of the learning problem
 - Identify methods that fit the problem setting
 - Determine parameters that properly classify the training set
 - Measure and control the ‘complexity’ of these functions



Lecture summary

Introduction to the class

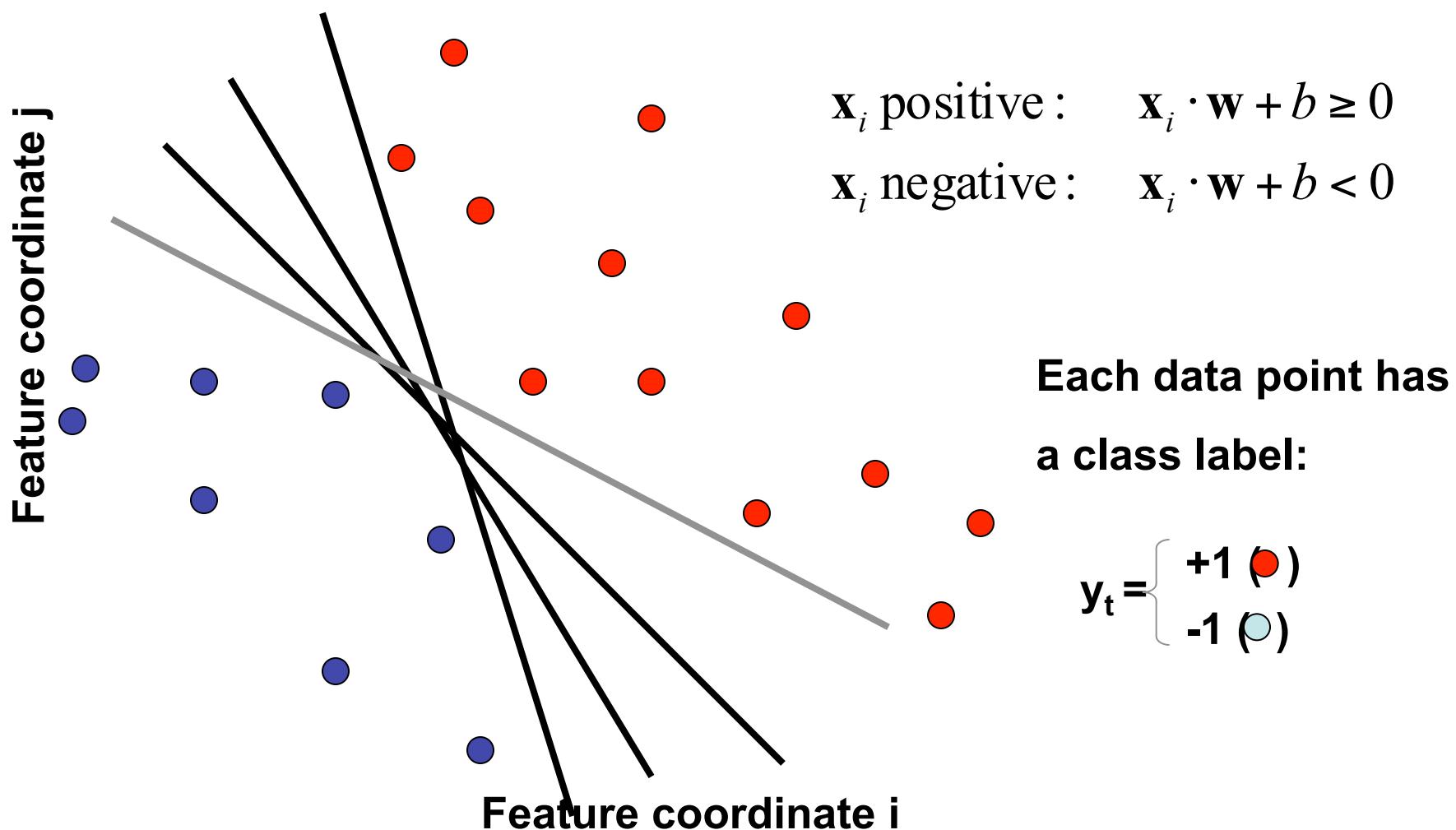
Introduction to the problem of classification

Linear classifiers

Image-based features

Linear Classifiers

- Linear expression (*hyperplane*) to separate positive and negative examples



Linear regression

Least-squares: $L(\mathbf{w}) = \mathbf{e}^T \mathbf{e}$ $\mathbf{e} = \mathbf{y} - \mathbf{X}\mathbf{w}$

$$\mathbf{w}^* = \arg \min_{\mathbf{w}} L(\mathbf{w}) = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$$

Ridge regression: $L(\mathbf{w}) = \mathbf{e}^T \mathbf{e} + \lambda \mathbf{w}^T \mathbf{w}$

$$\mathbf{w}^* = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y}$$

Tuning λ : cross-validation

L2 Regularization: Ridge regression

Penalize classifier's L2 norm: $\|w\|_2^2 = \sum_{k=1}^K w_k^2 = \mathbf{w}^T \mathbf{w}$

Loss function: $L(\mathbf{w}) = \mathbf{e}^T \mathbf{e} + \lambda \mathbf{w}^T \mathbf{w}$

$$\begin{aligned}\mathbf{e} &= \mathbf{y} - \mathbf{X}\mathbf{w} && \text{data term} \quad \text{complexity term} \\ &= \mathbf{y}^T \mathbf{y} - 2\mathbf{w}^T \mathbf{X}^T \mathbf{y} + \mathbf{w}^T (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I}) \mathbf{w}\end{aligned}$$

$$\mathbf{w}^* = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y}$$

Full-rank matrix

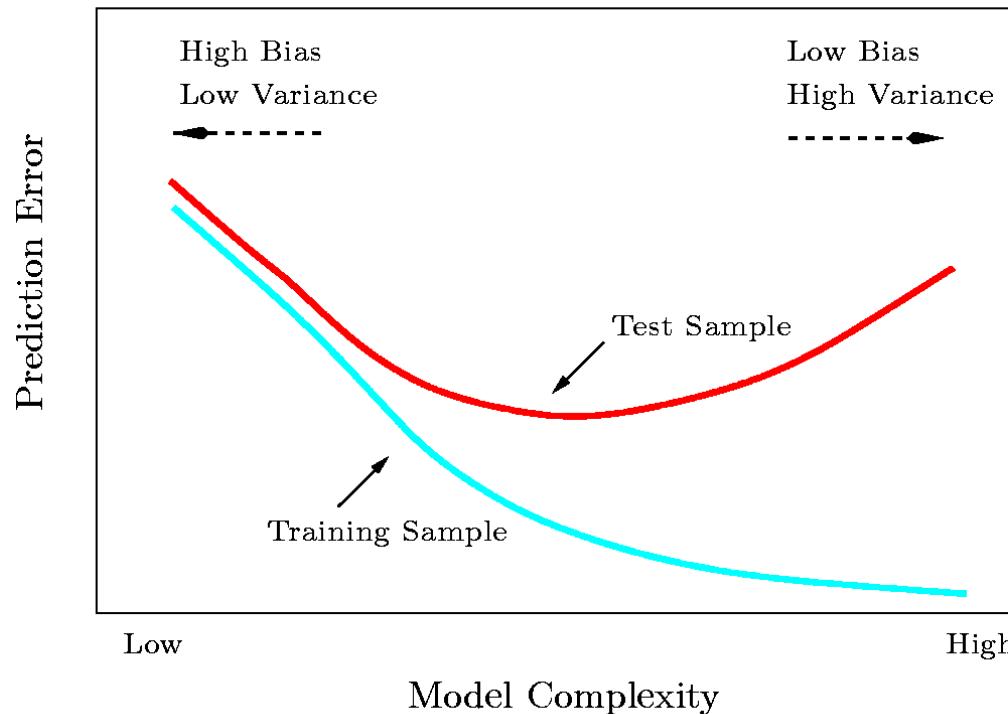
So how do we set λ ?

What is a good tradeoff between accuracy and complexity?

Tuning the model's complexity

A flexible model approximates the target function well in the training set
but can be fooled by noise and overtrain

A rigid model is more robust
but will not always provide a good fit





Lecture summary

Introduction to the class

Introduction to the problem of classification

Linear classifiers

Image-based features

Gabor, SIFT, HOG, Haar...

Encapsulate domain knowledge about desired invariances

computational efficiency

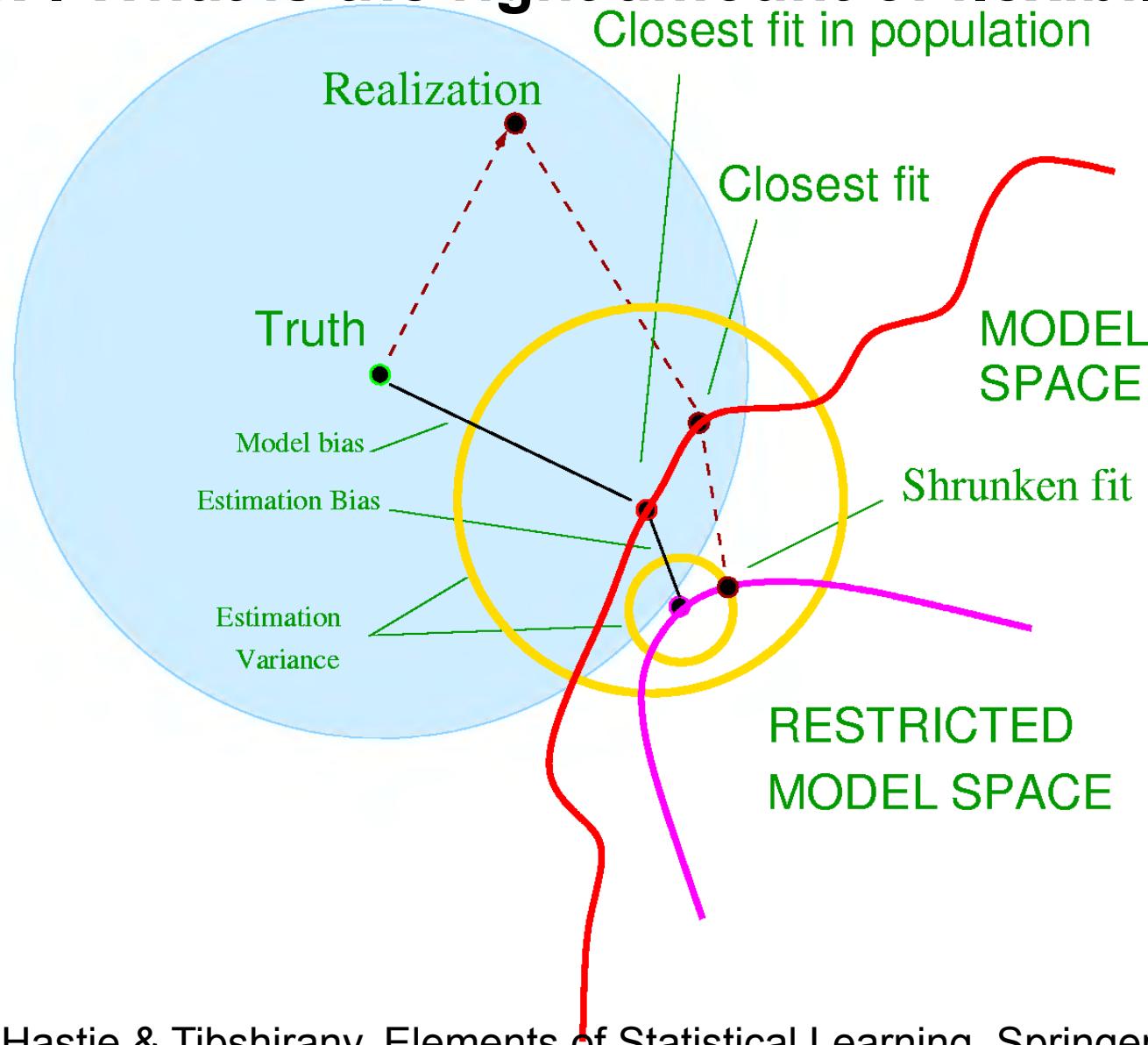
degree of invariance

task-specific performance

analytical tractability

...

Appendix-I What is the right amount of flexibility?



Bias-Variance-I

Assume underlying function: $y = g(x) + \epsilon$

Our model approximates it by: $\hat{y} = \hat{g}(x) = f_{\hat{w}}(x)$ $\hat{w} = h(\lambda, S)$

Approximation quality: affected by model's flexibility, and the training set.

Different training set realizations: different models

Model's value at x_0 : random variable

Express the expected generalization error of the model at x_0 :

$$\begin{aligned}\text{Err}(x_0) &= E[(y - \hat{g}(x_0))^2] = E[(y - g(x_0) + g(x_0) - \hat{g}(x_0))^2] \\ &= E[((y - g(x_0)) + (g(x_0) - E[\hat{g}(x_0)]) + (E[\hat{g}(x_0)] - \hat{g}(x_0)))^2] \\ &= \underbrace{\sigma^2 + (g(x_0) - E[\hat{g}(x_0)])^2}_{\text{Bias}} + \underbrace{E[(E[\hat{g}(x_0)] - \hat{g}(x_0))^2]}_{\text{Variance}}\end{aligned}$$

Appendix-II: Ridge regression = parameter shrinkage

Reference: Hastie & Tibshirani, Elements of Statistical Learning, Springer 2001

Least squares parameter estimation: minimization of

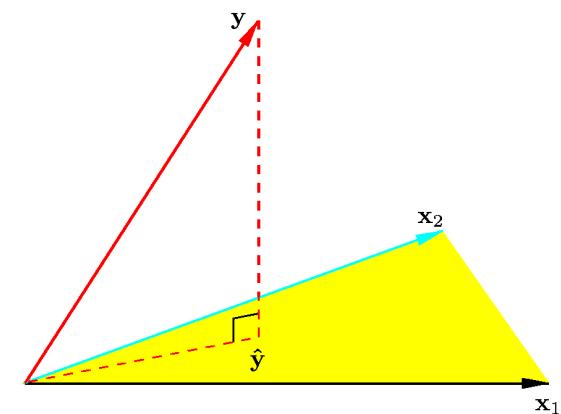
$$RSS(w) = (\mathbf{y} - \mathbf{X}w)^T(\mathbf{y} - \mathbf{X}w)$$

$$\frac{\partial RSS(w)}{\partial w} = 2\mathbf{X}^T(\mathbf{y} - \mathbf{X}w)$$

$$\mathbf{X}^T\mathbf{y} = \mathbf{X}^T\mathbf{X}\hat{w}$$

$$\hat{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

$$\hat{\mathbf{y}} = \mathbf{X}\hat{w} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$



SVD-based interpretation of least squares

Singular Value Decomposition (SVD) of \mathbf{X}

$$\underbrace{\mathbf{X}}_{M \times N} = \underbrace{\mathbf{U}}_{M \times N} \underbrace{\mathbf{D}}_{N \times N} \underbrace{\mathbf{V}^T}_{N \times N}$$

$$\mathbf{D} = \text{diag}(d_1, \dots, d_{\min(M,N)}, \underbrace{0, \dots, 0}_{\max(M-N,0)}), \quad d_i \geq d_{i+1}, d_i \geq 0$$

Reconstruction of \mathbf{y} on the subspace spanned by \mathbf{X} 's columns

$$\hat{\mathbf{y}} = \mathbf{X} (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$$

$$\begin{aligned}\hat{\mathbf{y}} &= \mathbf{UDV}^T (\mathbf{VD}^T \mathbf{U}^T \mathbf{UDV}^T)^{-1} \mathbf{VD}^T \mathbf{U}^T \mathbf{y} \\ &= \mathbf{U} (\mathbf{U}^T \mathbf{y})\end{aligned}$$

- \mathbf{U} orthonormal basis for d -dimensional subspace of R^M .
- $\mathbf{U}^T \mathbf{y}$ expansion coefficients for projection of \mathbf{Y} onto this basis.
- $\hat{\mathbf{y}} = \mathbf{U} \mathbf{U}^T \mathbf{y}$ reconstruction (projection) of \mathbf{Y} on basis U

SVD-based interpretation of Ridge Regression

- Minimization of

$$E_{ridge}(w, \lambda) = RSS(w) + \lambda C_{ridge}(w)$$

$$= \sum_{i=1}^M (y^i - \mathbf{X}w)^T (\mathbf{y} - \mathbf{X}w) + \lambda w^T w$$

- Regularization: penalty on large values of $w^T w$

- Solution

$$\hat{w} = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y}$$

- SVD interpretation

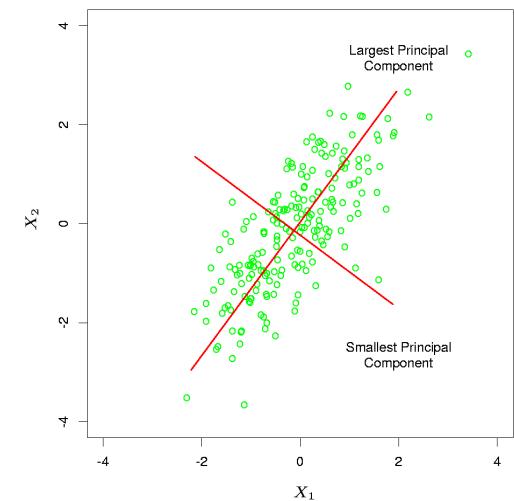
$$\begin{aligned}\hat{\mathbf{y}} &= \mathbf{X} (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y} \\ &= \mathbf{U} \mathbf{D} (\mathbf{D}^2 + \lambda \mathbf{I})^{-1} \mathbf{D} \mathbf{U}^T \mathbf{y}\end{aligned}$$

- ‘Shrinkage’

$$= \sum_{m=1}^M \mathbf{u}_m \frac{d_m^2}{d_m^2 + \lambda} \mathbf{u}_m^T \mathbf{y}$$

Feature Space Interpretation of ridge regression

- Covariance matrix (centered data): $\mathbf{X}^T \mathbf{X} = \mathbf{V} \mathbf{D}^T \mathbf{U}^T \mathbf{U} \mathbf{D} \mathbf{V}^T = \mathbf{V} \mathbf{D}^2 \mathbf{V}^T$
- \mathbf{D}^2 eigenvectors of covariance matrix
- \mathbf{V} : eigenvalues
- Shrinkage: downplay coefficients corresponding to smaller axes
- Effect for $x_1 = x_2$
 - Projections:
$$c = E(x_1^2) = \frac{x_1 + x_2}{2} \quad \frac{x_1 - x_2}{2}$$
- Eigenvalues $\frac{c^2}{c^2 + \lambda}$
- Shrinkage factors 0



Lasso

- Minimization of $E_{Lasso}(w, \lambda) = RSS(w) + \lambda C_{Lasso}(w)$

$$= \sum_{i=1}^M (y^i - \mathbf{X}w)^T (\mathbf{y} - \mathbf{X}w) + \lambda \sum_i |w_i|$$

- Regularization: penalty on sum of absolute values of w
- Comparison with Ridge Regression

$$\frac{\partial(E_{Lasso}(w))}{\partial w_i} = \lambda \text{sign}(w_i) + \sum_{i=1}^M -2(y^i - \mathbf{X}w)^T w_i$$

$$\frac{\partial(E_{Ridge}(w))}{\partial w_i} = \lambda(w_i) + \sum_{i=1}^M -2(y^i - \mathbf{X}w)^T w_i$$

- Gradient does not depend on value of w
- Sparsity & subset selection