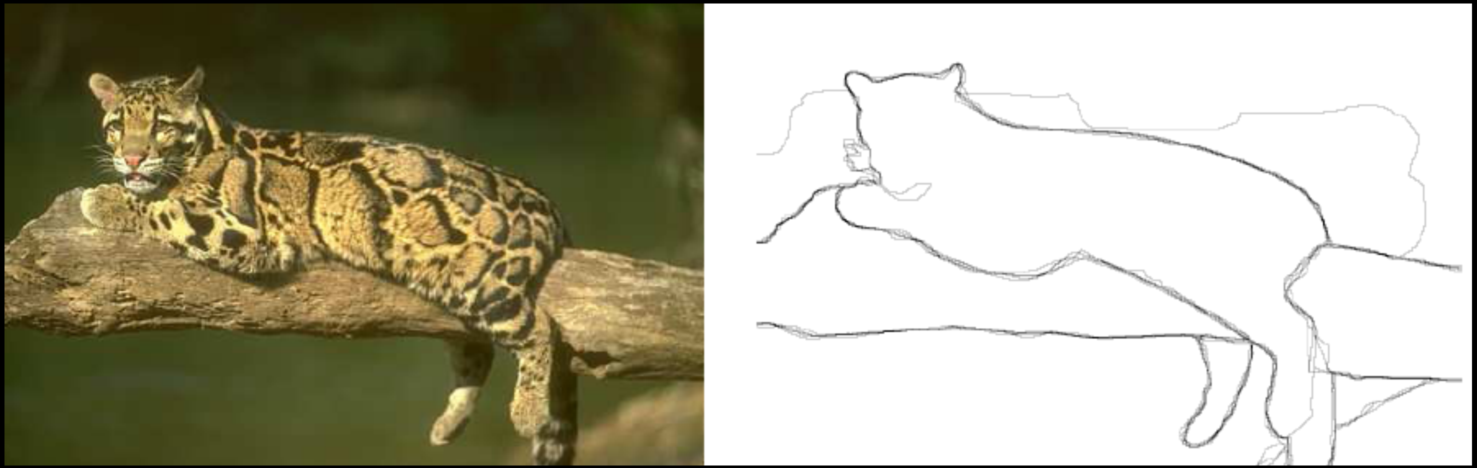
Machine Learning for Computer Vision

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

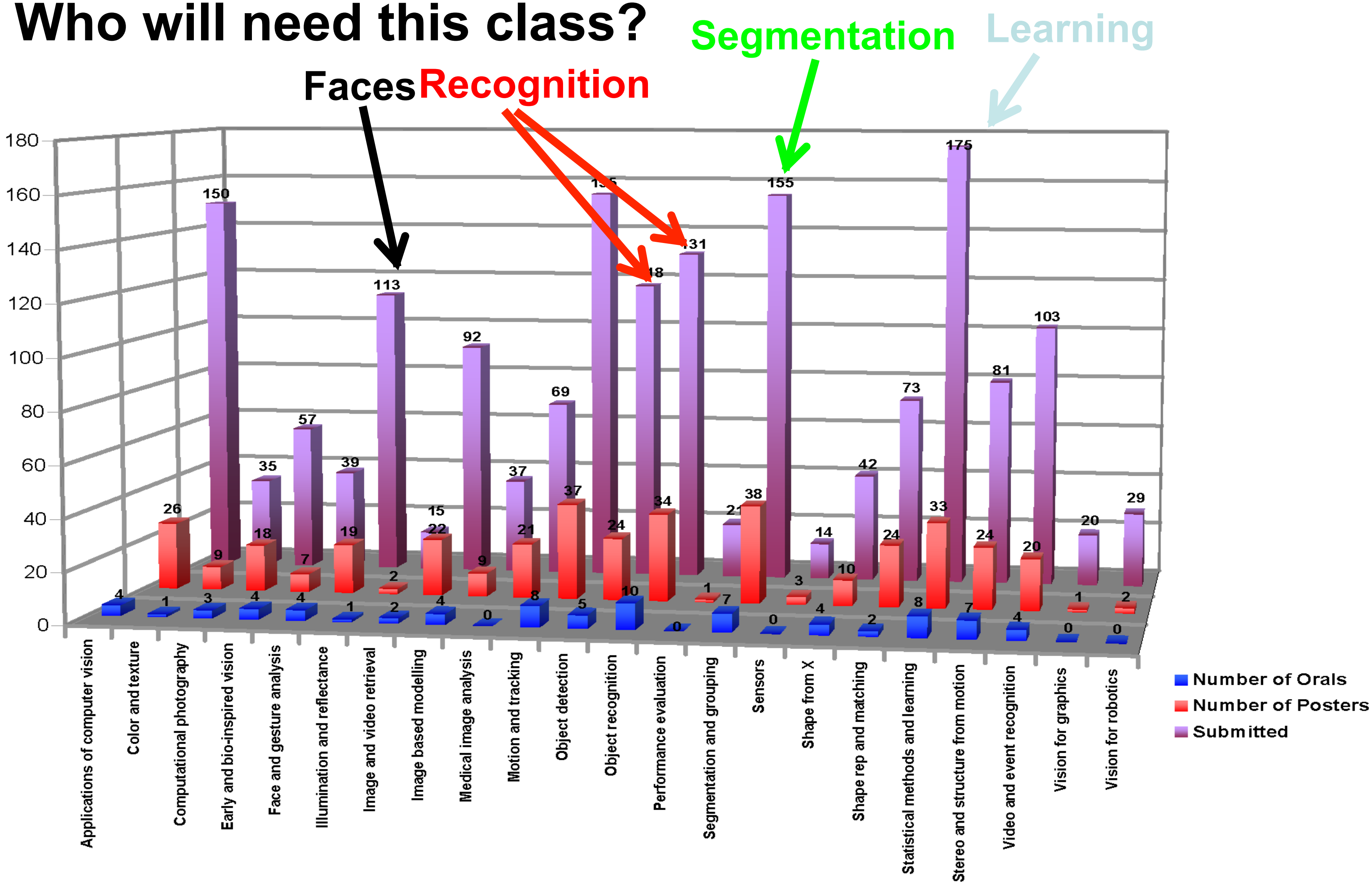
Image-based features

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?

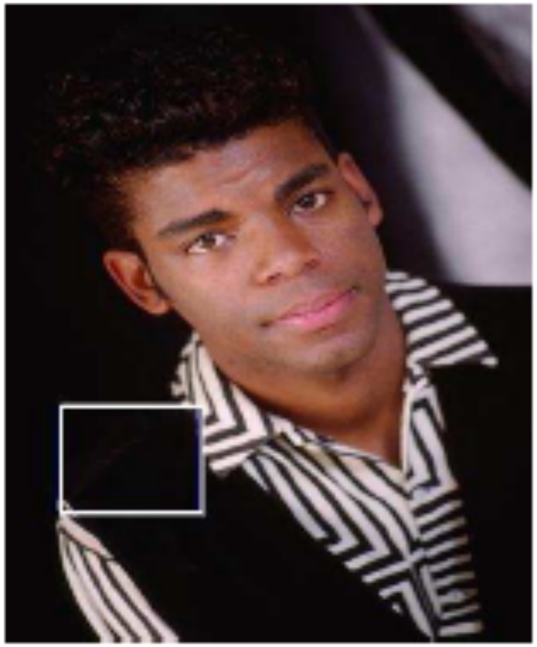


Submission/Acceptance Statistics from CVPR 2010

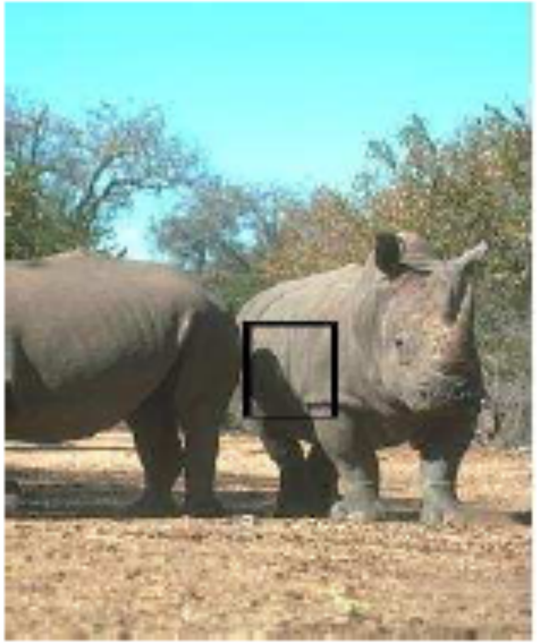
Boundary detection problem

Object/Surface Boundaries

Signal-level challenges



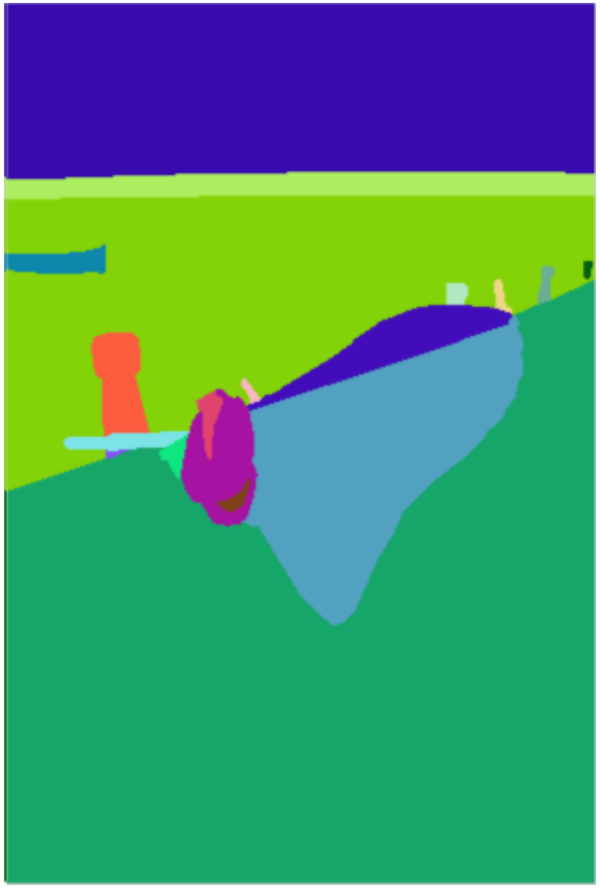
Poor contrast



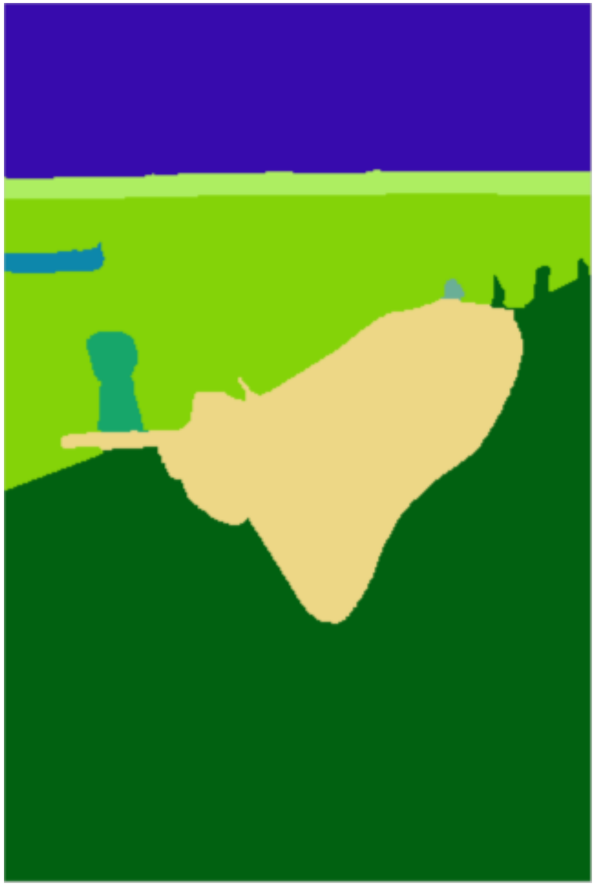
Shadows

Texture

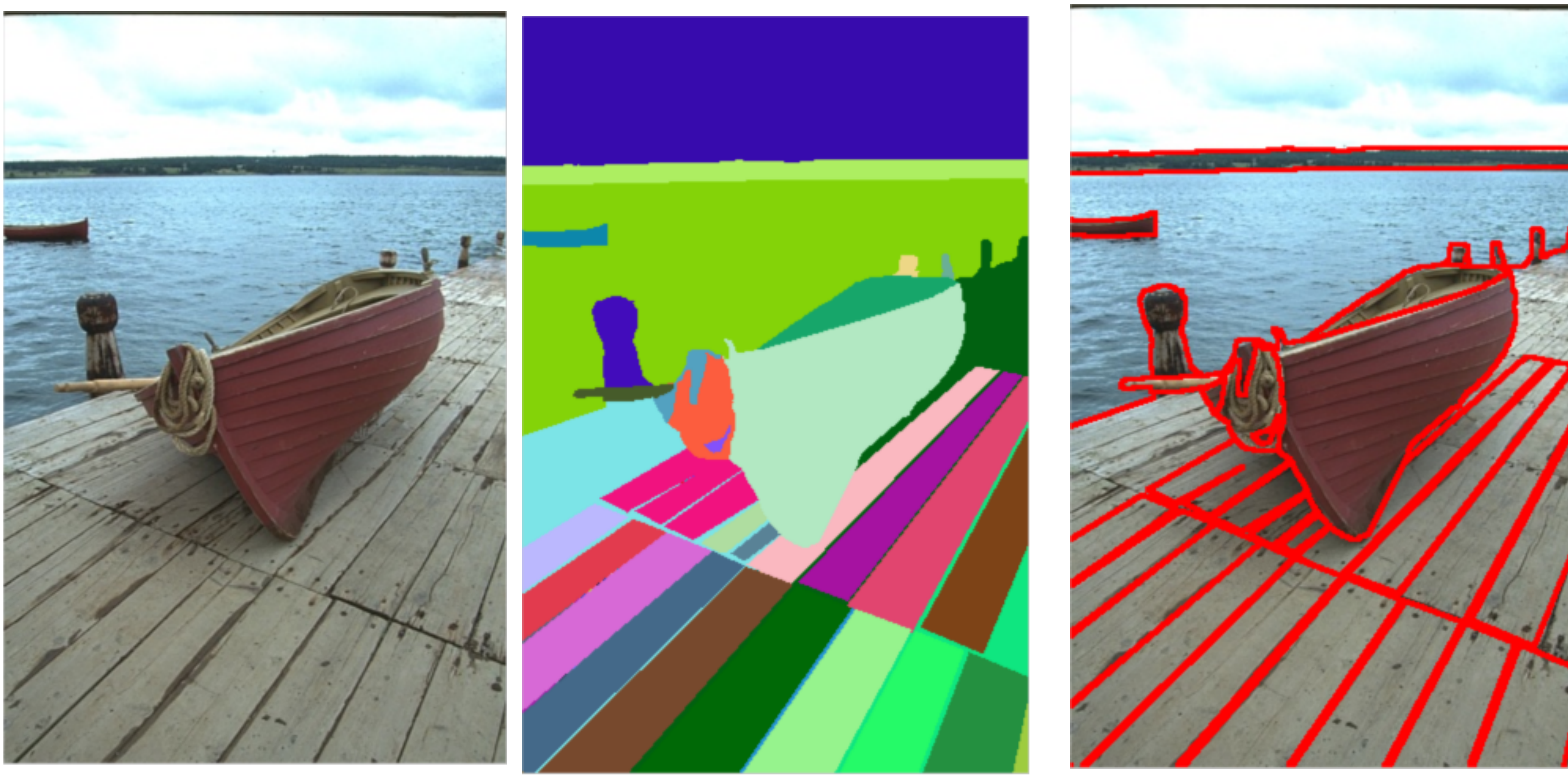
Fundamental challenges: can humans do it?



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)

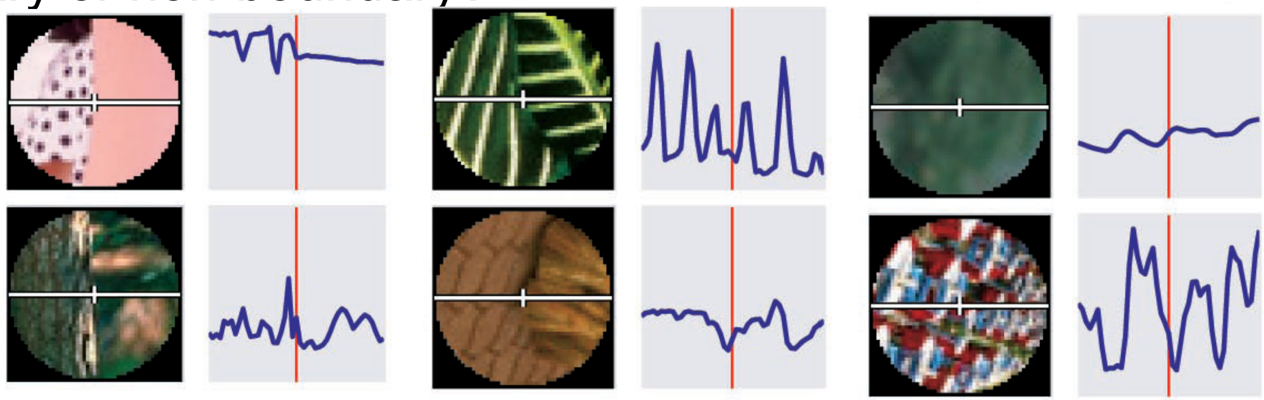
K. Siddiqi, Y. Lauzière, A. Tannenbaum, S. Zucker: Area and length minimizing flows for shape segmentation. IEEE TIP 7(3): 433-443 (1998)

Statistical approaches

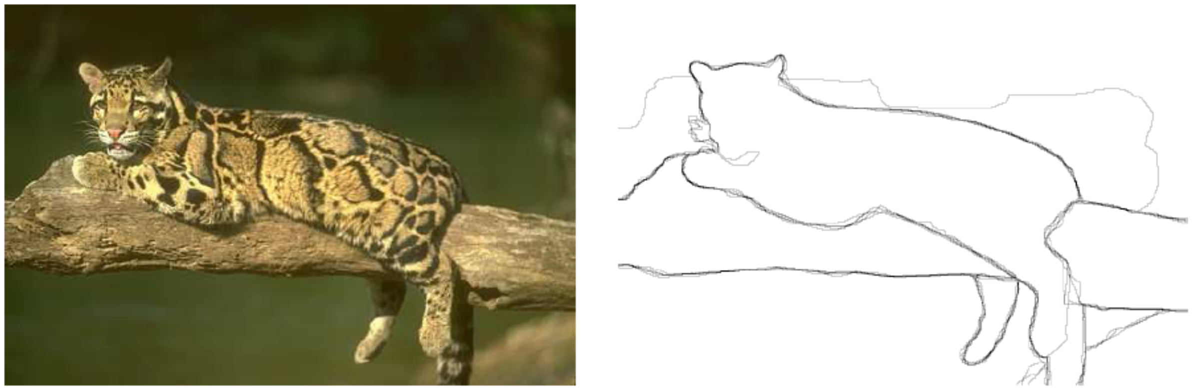
Agnès Desolneux, Lionel Moisan, Jean-Michel Morel: `Meaningful Alignments'. International Journal of Computer Vision 40(1): 7-23 (2000)

Learning-based approaches

Boundary or non-boundary?



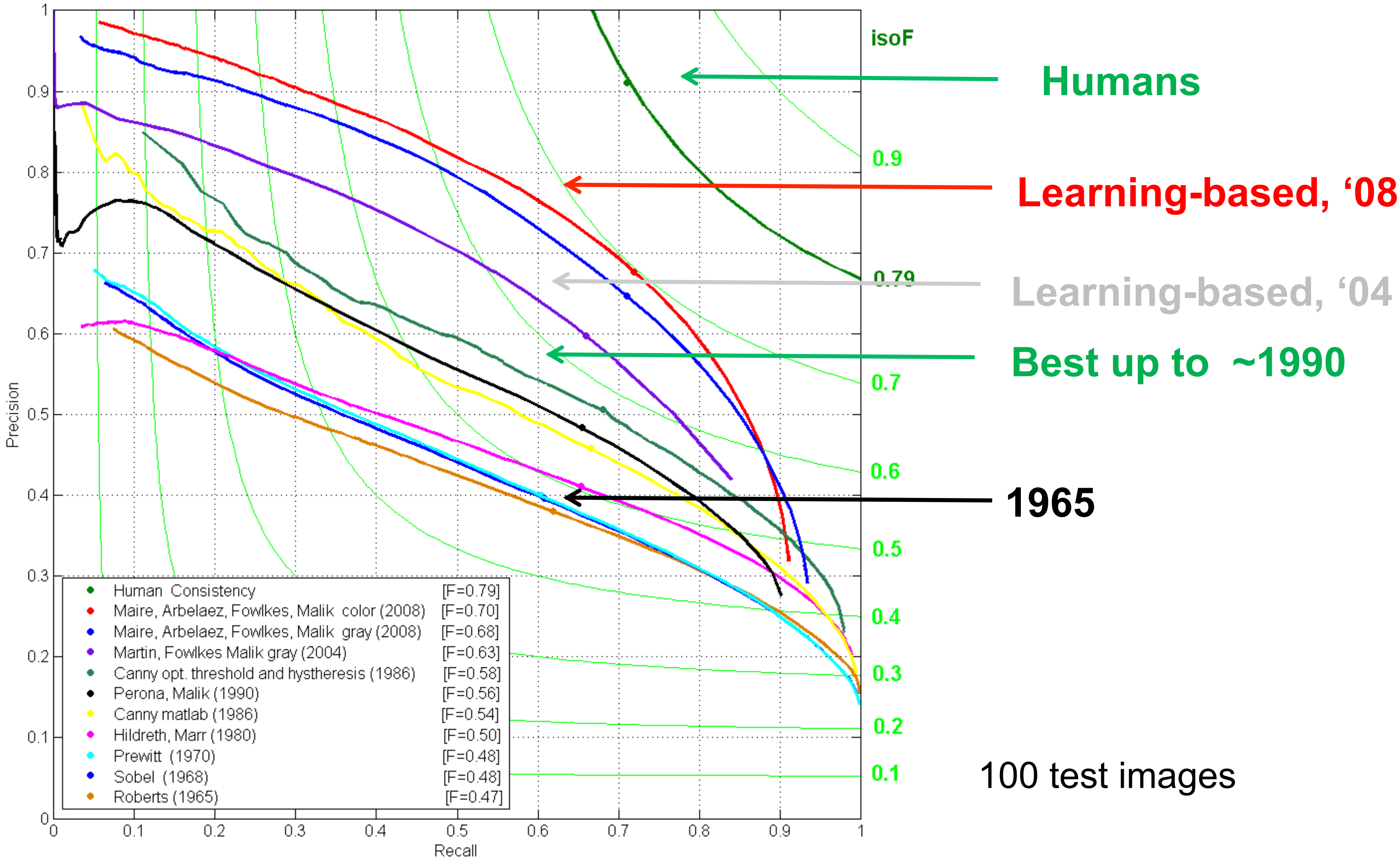
Use human-annotated segmentations



Use any visual cue as input to the decision function.
Use decision trees/logisitic regression/boosting/… and *learn* to combine the individual inputs.

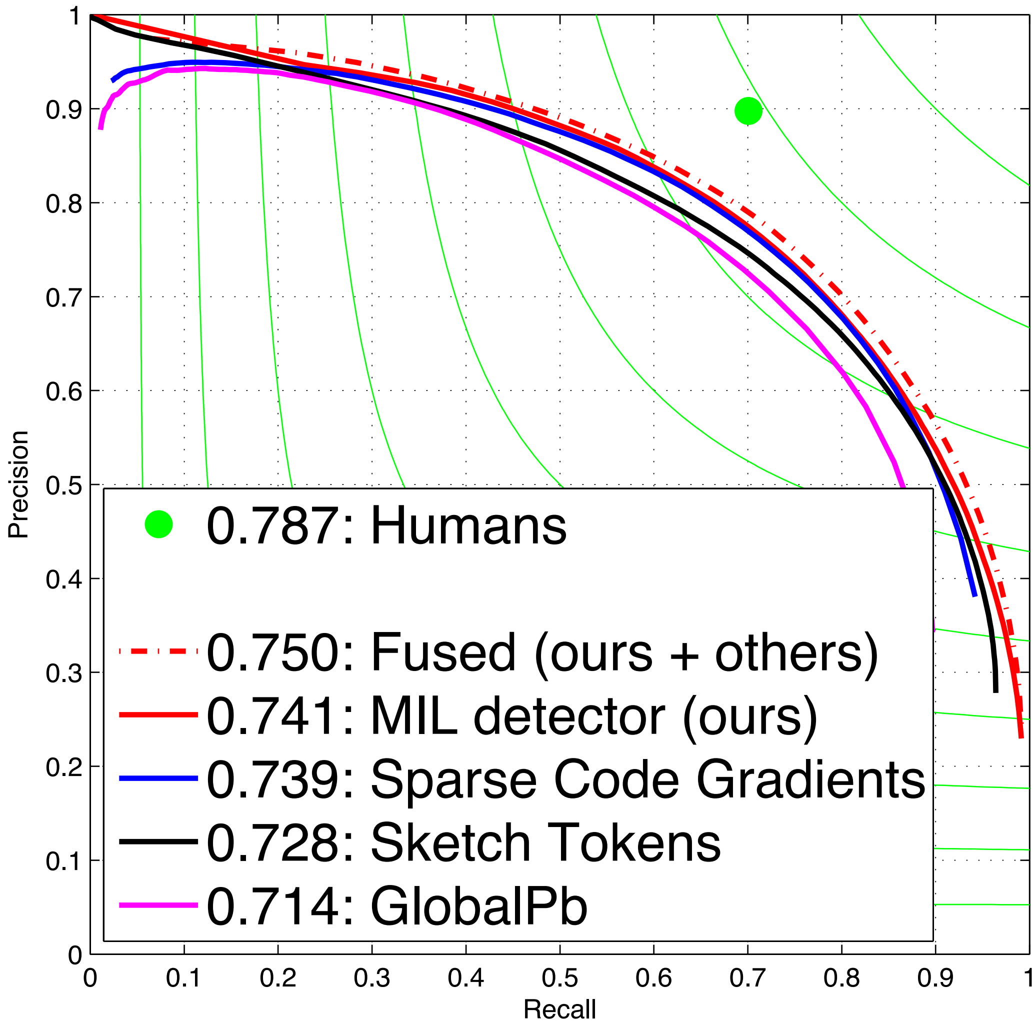
S. Konishi, A.Yuille, J. Coughlan, S.C. Zhu, “Statistical Edge Detection: Learning and Evaluating Edge Cues”, IEEE PAMI, 2003
D. Martin, C. Fowlkes, J. Malik. "Learning to Detect Natural Image Boundaries Using Local Brightness, Color and Texture Cues", IEEE PAMI, 2004

Progress during the last 4 decades



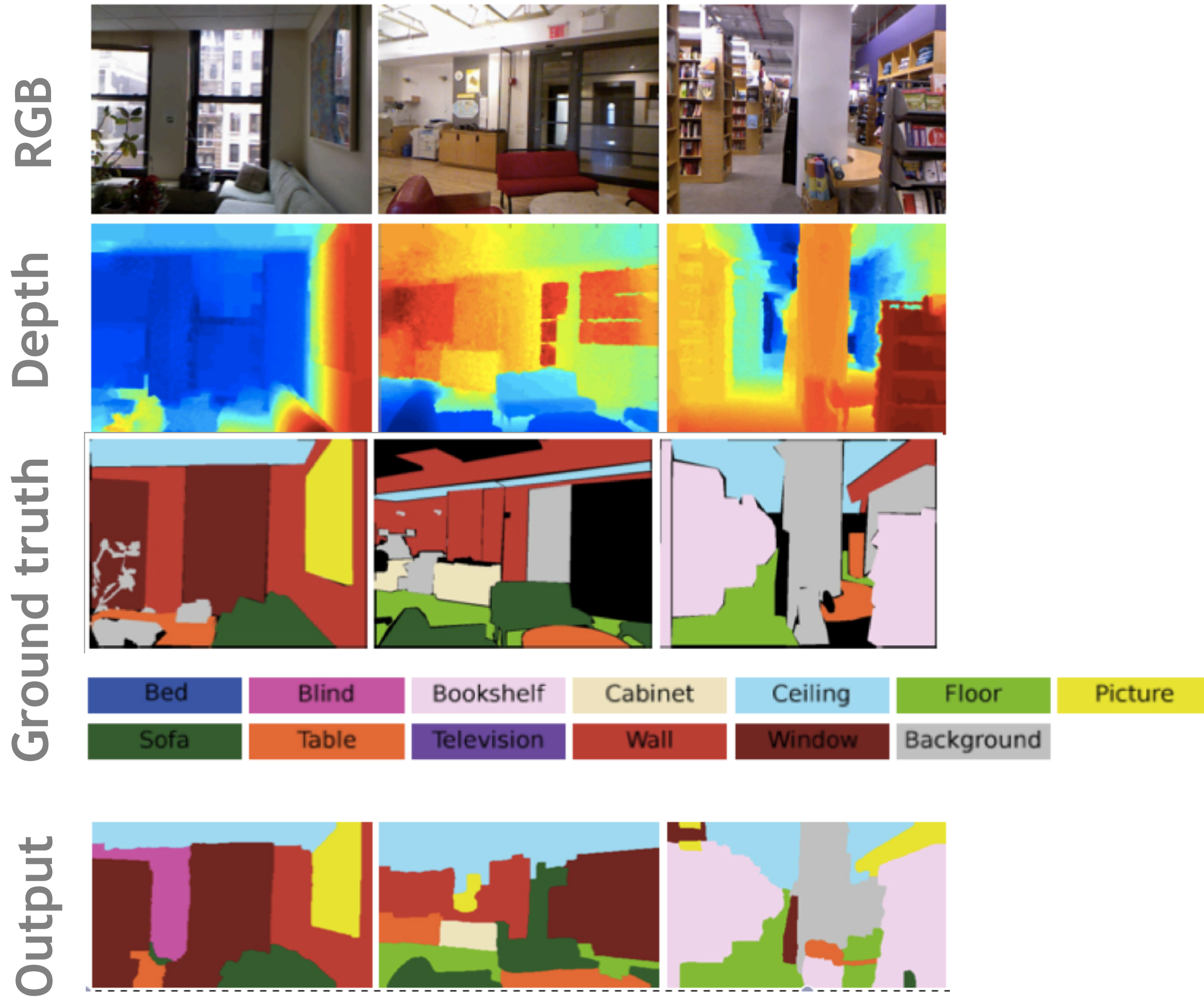
Reference: Maire, Arberaez, et. al., IEEE PAMI 2011

Progress during this decade



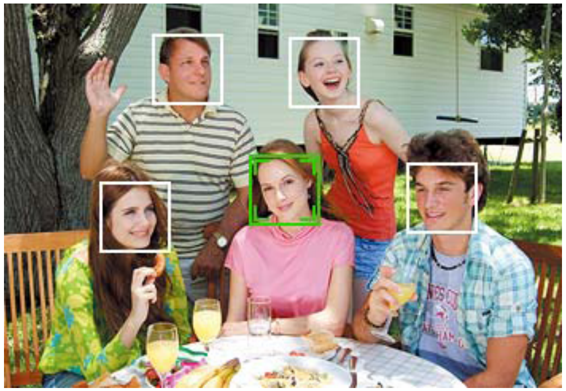
Precision-Recall Curves on the Berkeley Benchmark

Learning and Vision problem II: RGB-D scene labeling

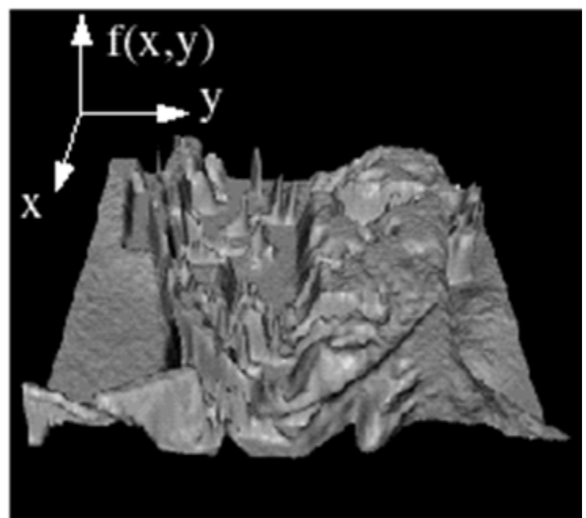


Learning and Vision problem III: Face Detection

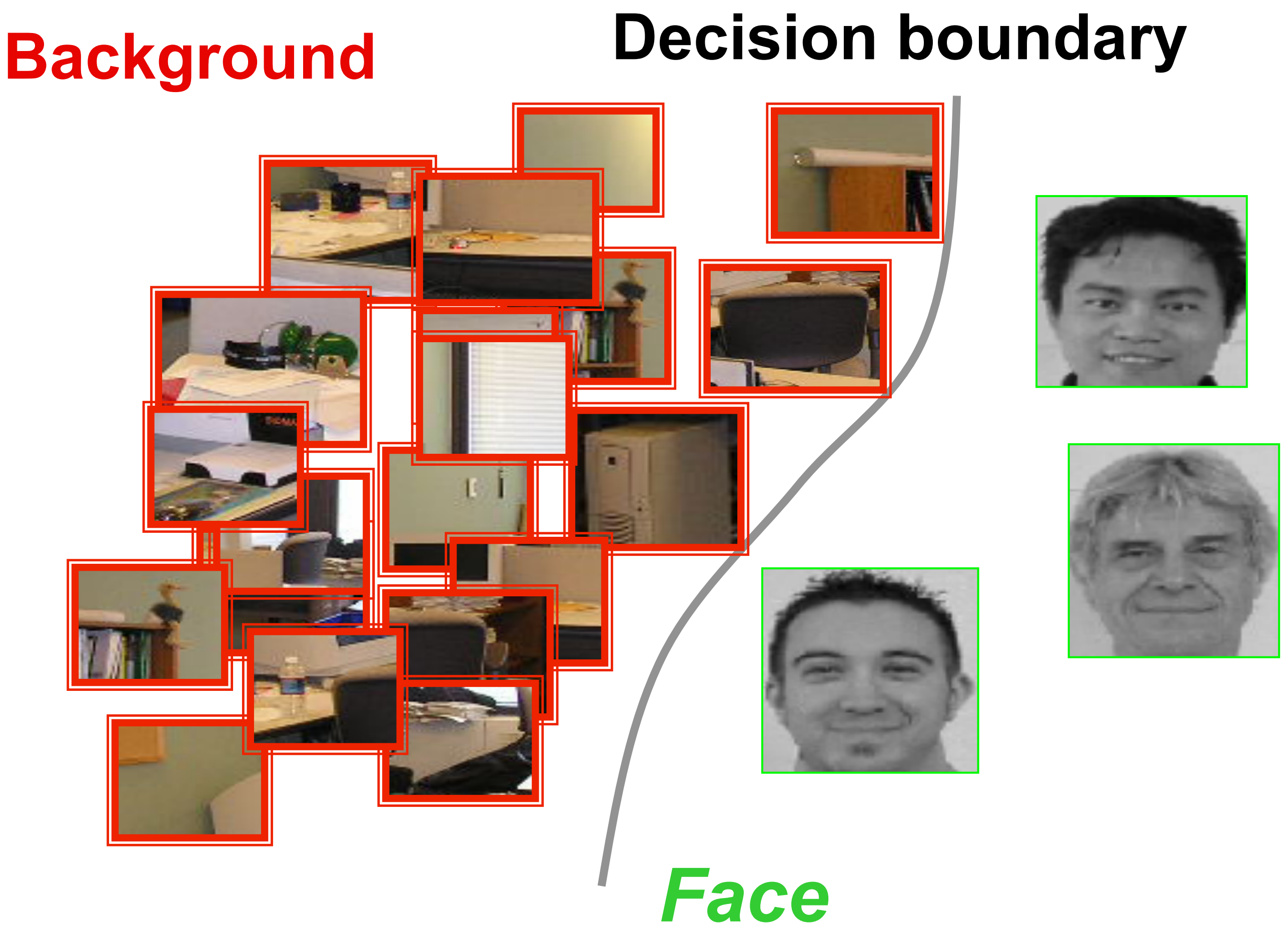
- How do digital cameras detect faces?



- Input to a digital camera: intensity at pixel locations



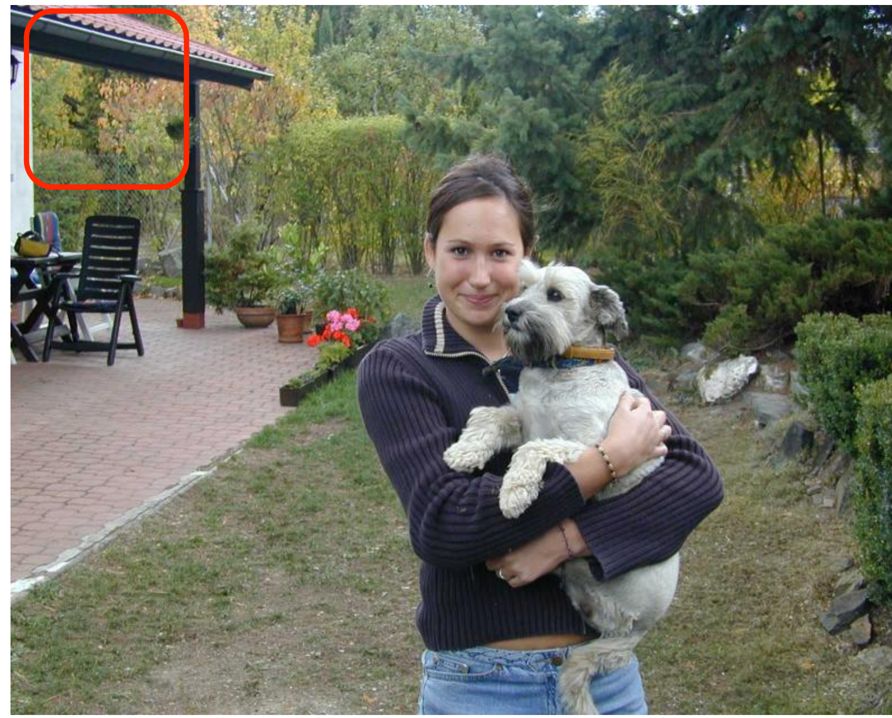
`Faceness function': classifier



Sliding window approaches

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

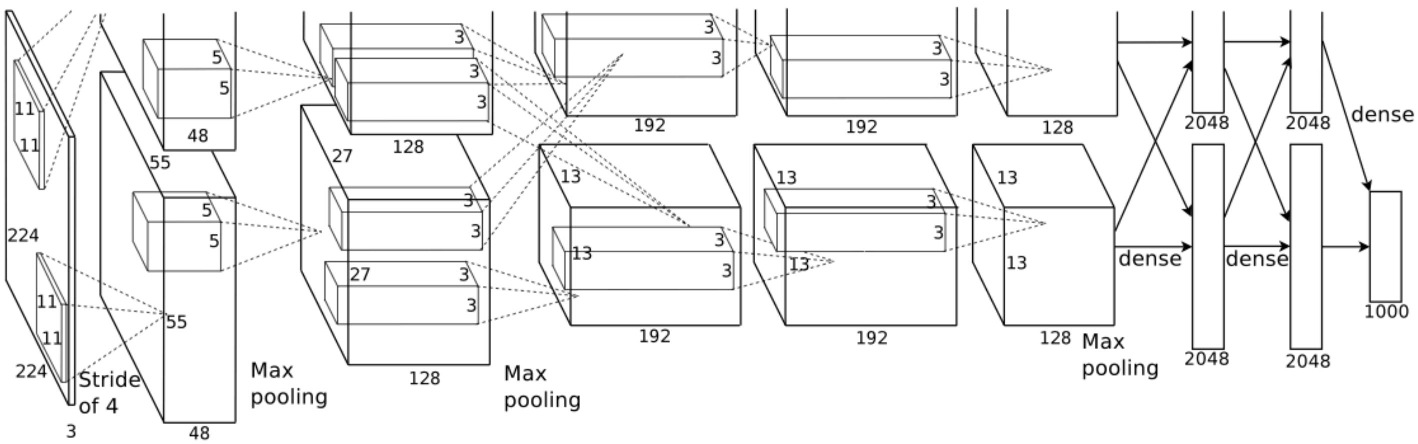
Window → **Classifier** → **Face** / **Non-face**



Slide credit: B. Leibe

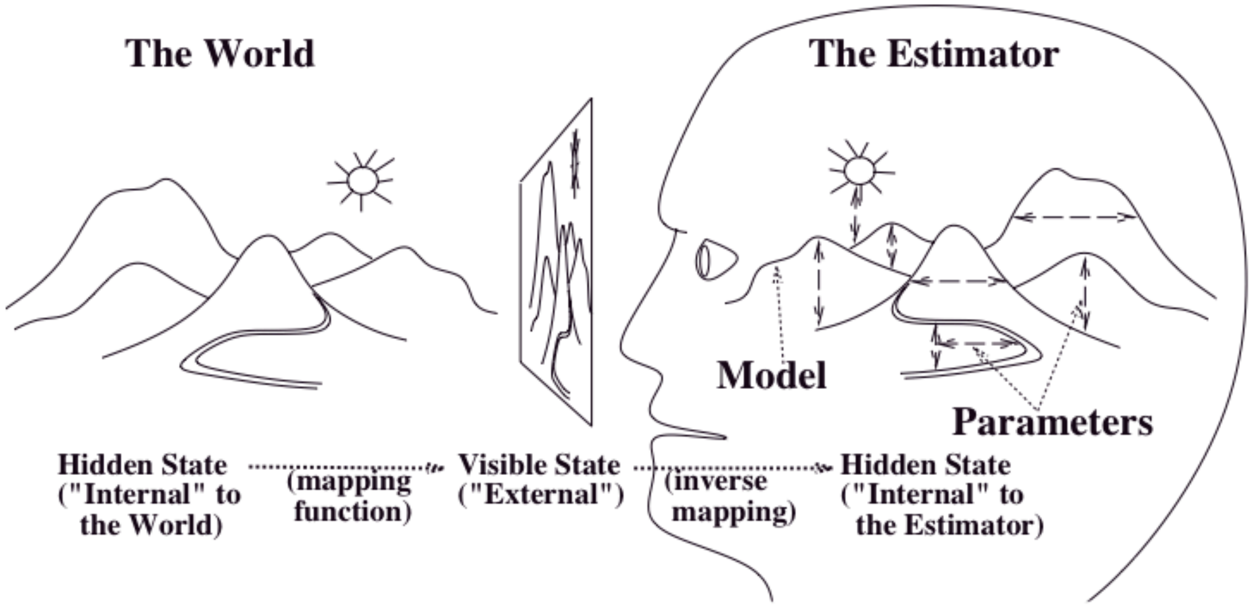
Two Main Approaches

- **Discriminative**



Two Main Approaches

- **Generative**



Likelihood **Prior**

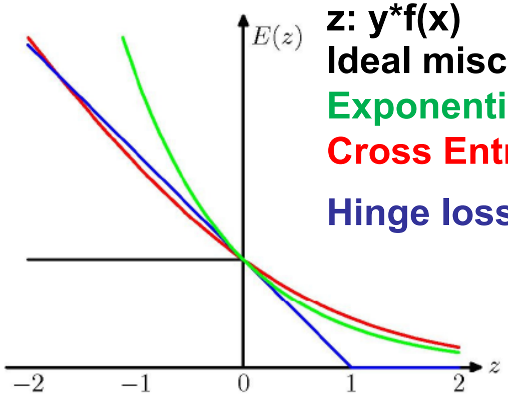
Posterior

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

Discriminative techniques

- Lectures 1-4:
  - Linear and Logistic Regression
  - Adaboost, Decision Trees, Random Forests
  - Support Vector Machines

- Unified treatment as loss-based learning



z: y*f(x)

Ideal misclassification cost H(-z) (# training errors)

Exponential Error exp(-z) (Adaboost)

Cross Entropy error ln(1 + exp(-z)) (Logistic regression)

Hinge loss max(0,1-z) (SVMs)

Generative Techniques

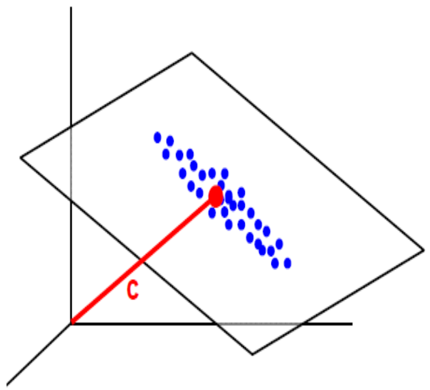
- **Lectures 5-7**
  - Hidden Variables, EM, Component Analysis
  - Structured Models (HMMs, Deformable Part Models)

- **Lecture 8**
  - ~~Discriminative Learning of Structured Models (2013)~~
  - Deep Learning and Object Detection

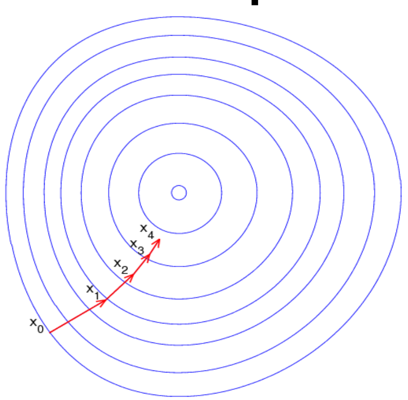
Coupling of theory with applications

Lecture 5: PCA + Newton-Raphson

PCA

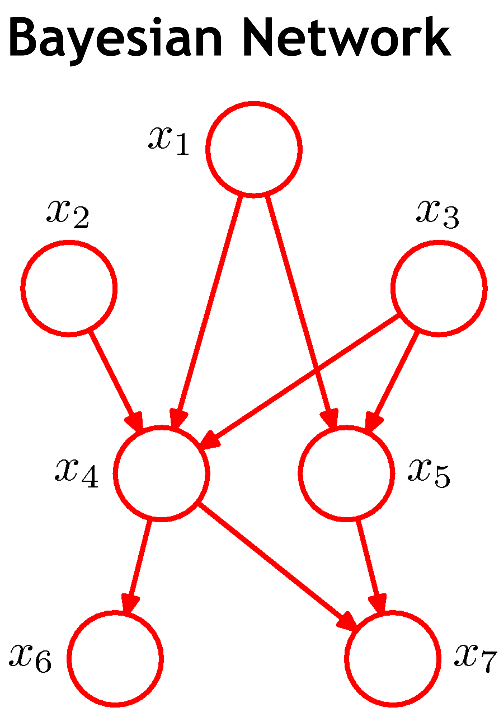


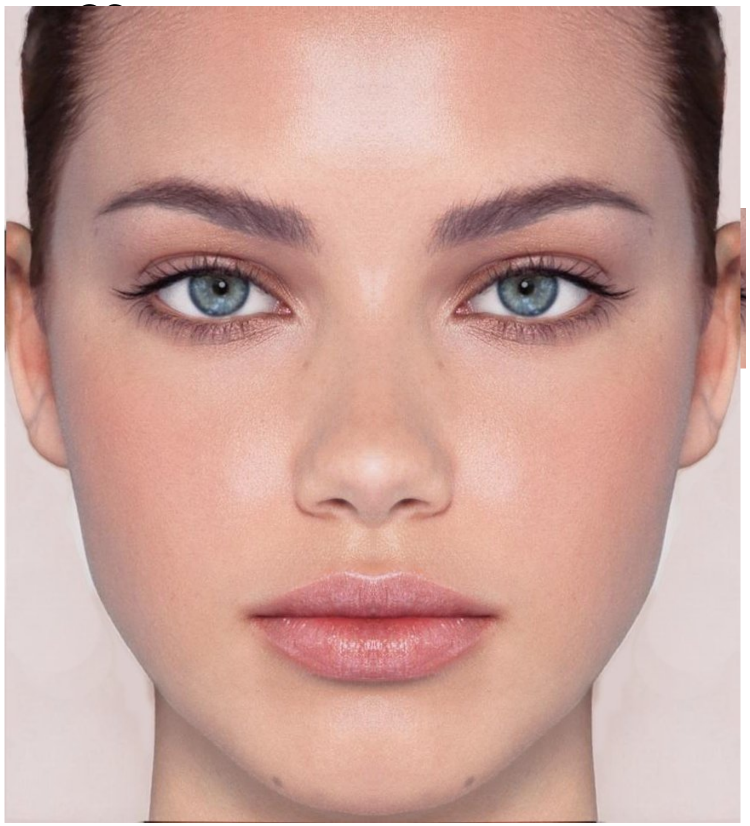
Newton-Raphson



Coupling of theory with applications

Lectures 5-6: Graphical Models + Detection

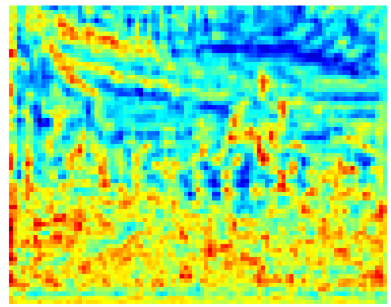




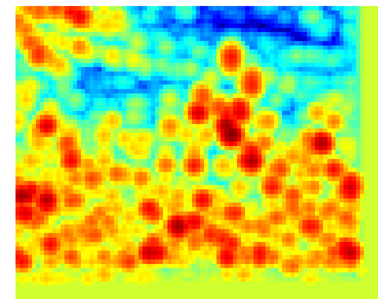
Coupling of theory with applications

Lecture 6: Dynamic Programming + Detection

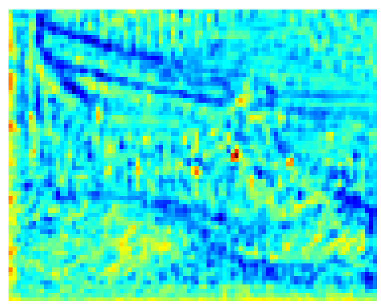
$$U_p(x') = \langle \mathbf{w}_p, \mathbf{H}(x') \rangle \qquad \max_{x'} [U_p(x') + B_p(x, x')]$$



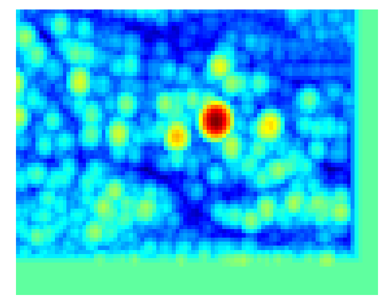
$p = 1$

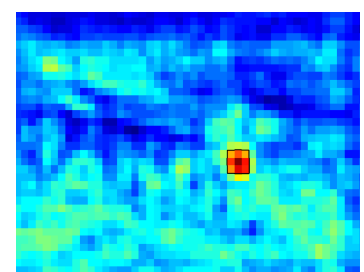


$\vdots$



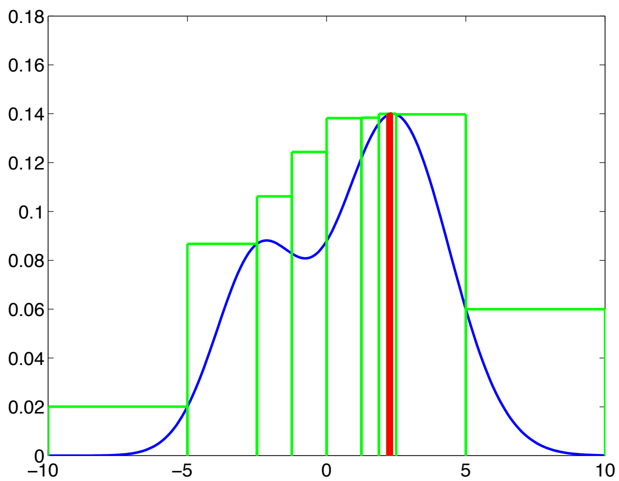
$p = P$





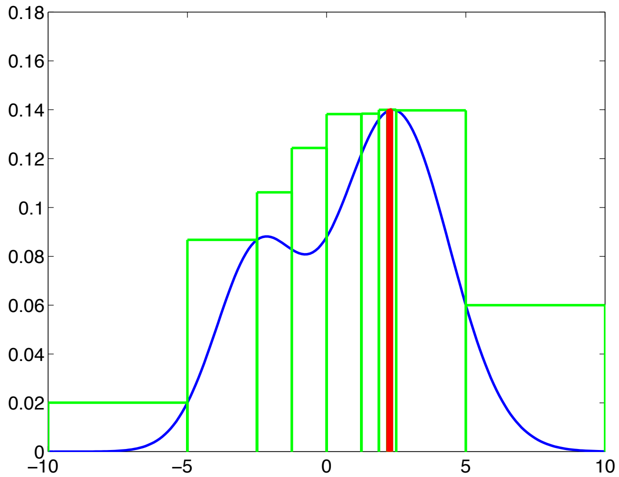
Coupling of theory with applications

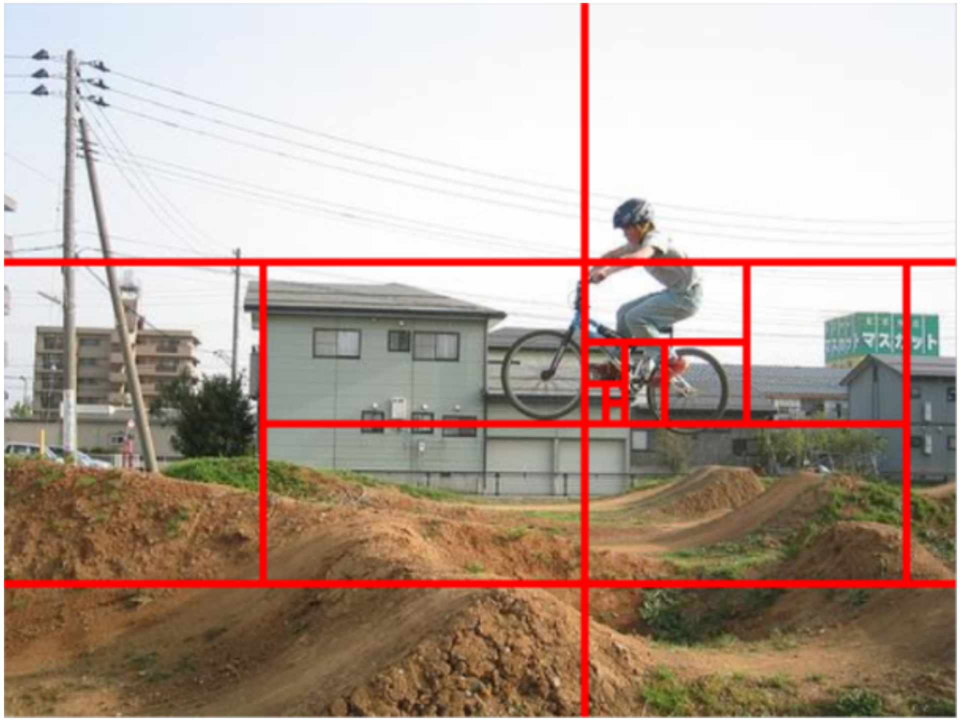
Lecture 7: Branch-and-Bound + Detection



Coupling of theory with applications

Lecture 7: Branch-and-Bound + Detection

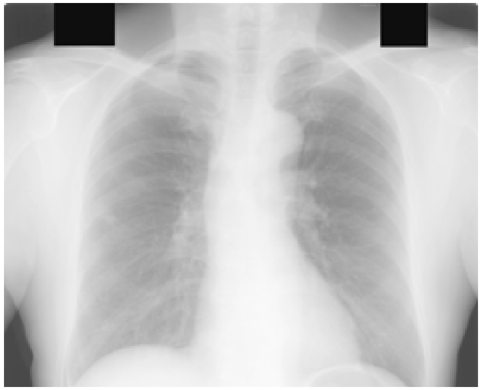




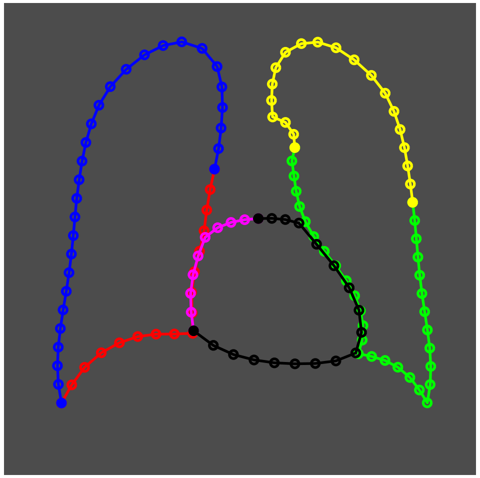
Coupling of theory with applications

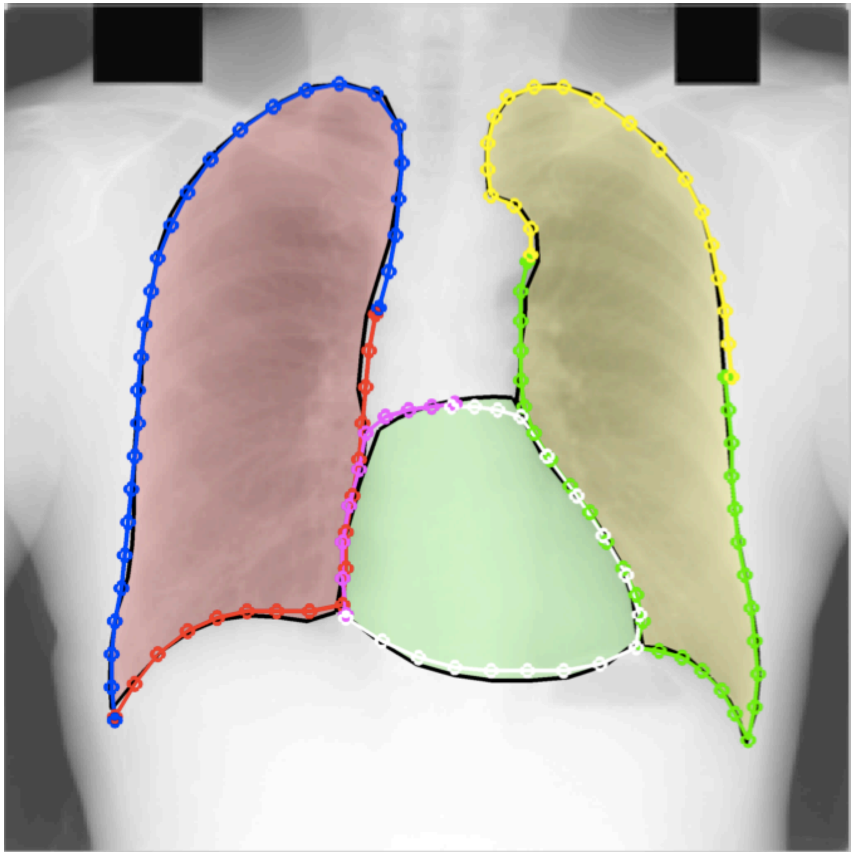
Lecture 7: ADMM + MRFs for Shape Segmentation

Input image



The model





Organization

- 3 labs in Matlab (10 points)
  - Start with small preparatory exercises (synthetic data)

- 1 Project (10 points)
  - Extension of 3 labs to real data

- Or: small-scale research project (20/20)
- Class webpage:

  http://cvn.ecp.fr/personnel/iasonas/teaching.html

- First class: ENS-Cachan, Oct. 2, Thursday 9h45

- Further questions: iasonas.kokkinos@ecp.fr

Lecture outline

Introduction to the course

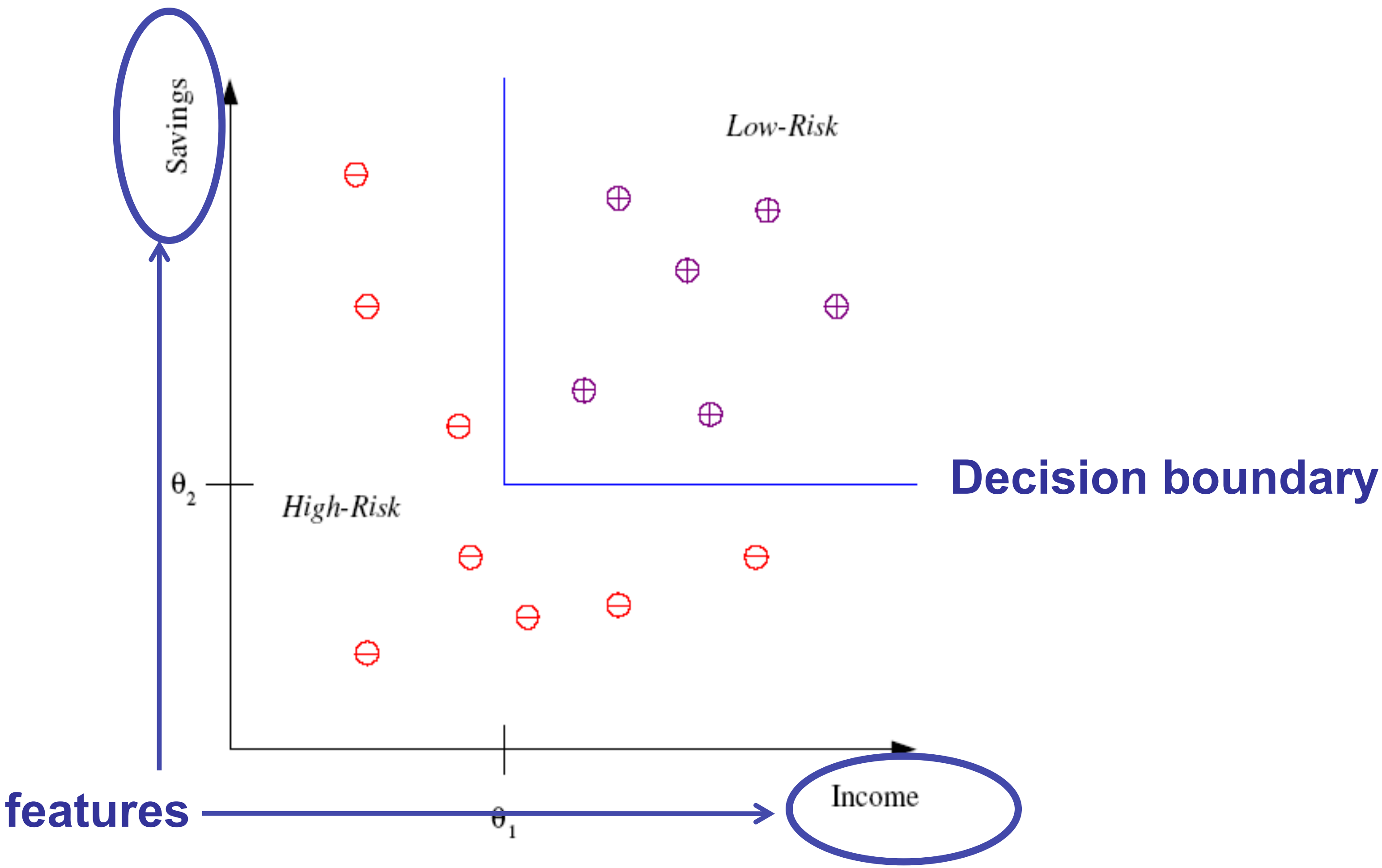
Introduction to the classification problem

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 $S = \{(x^i, y^i)\}\ i = 1, \ldots, N$

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

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

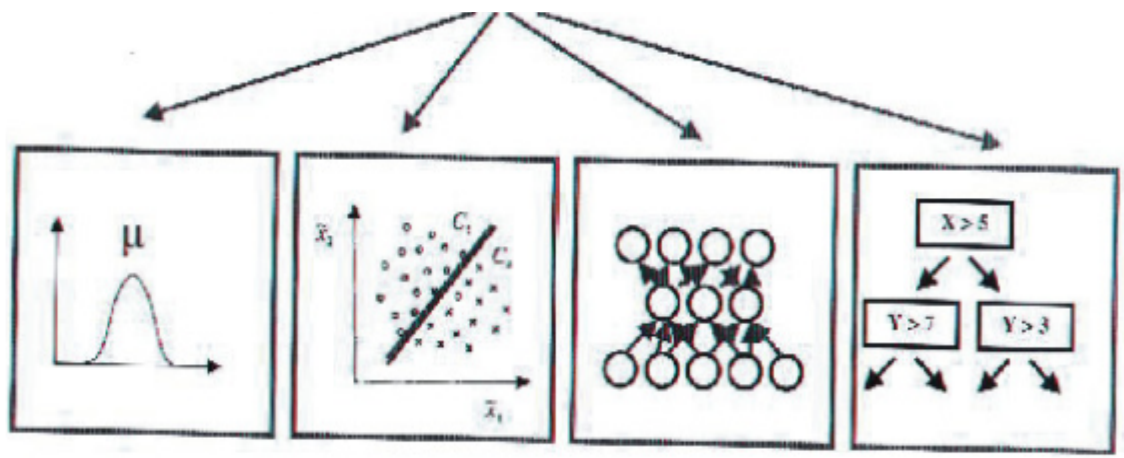
Why `simple'? good generalization outside training set

`works well': quantified by loss criterion $L(S, f)$

Classifier function

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

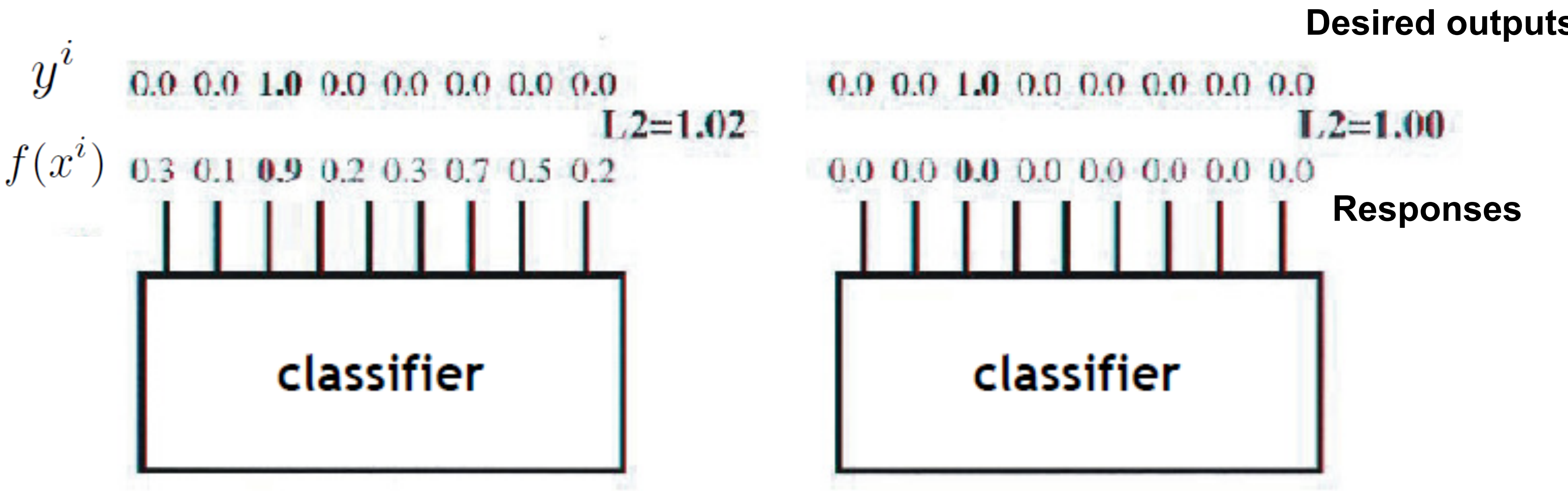
$$y = f_w(x) \quad (= f(x, w))$$



- 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

Slide credit: B. Leibe/B. Schiele

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)

Slide credit: B. Leibe/B. Schiele

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 $y^i \in \{0,1\}$

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

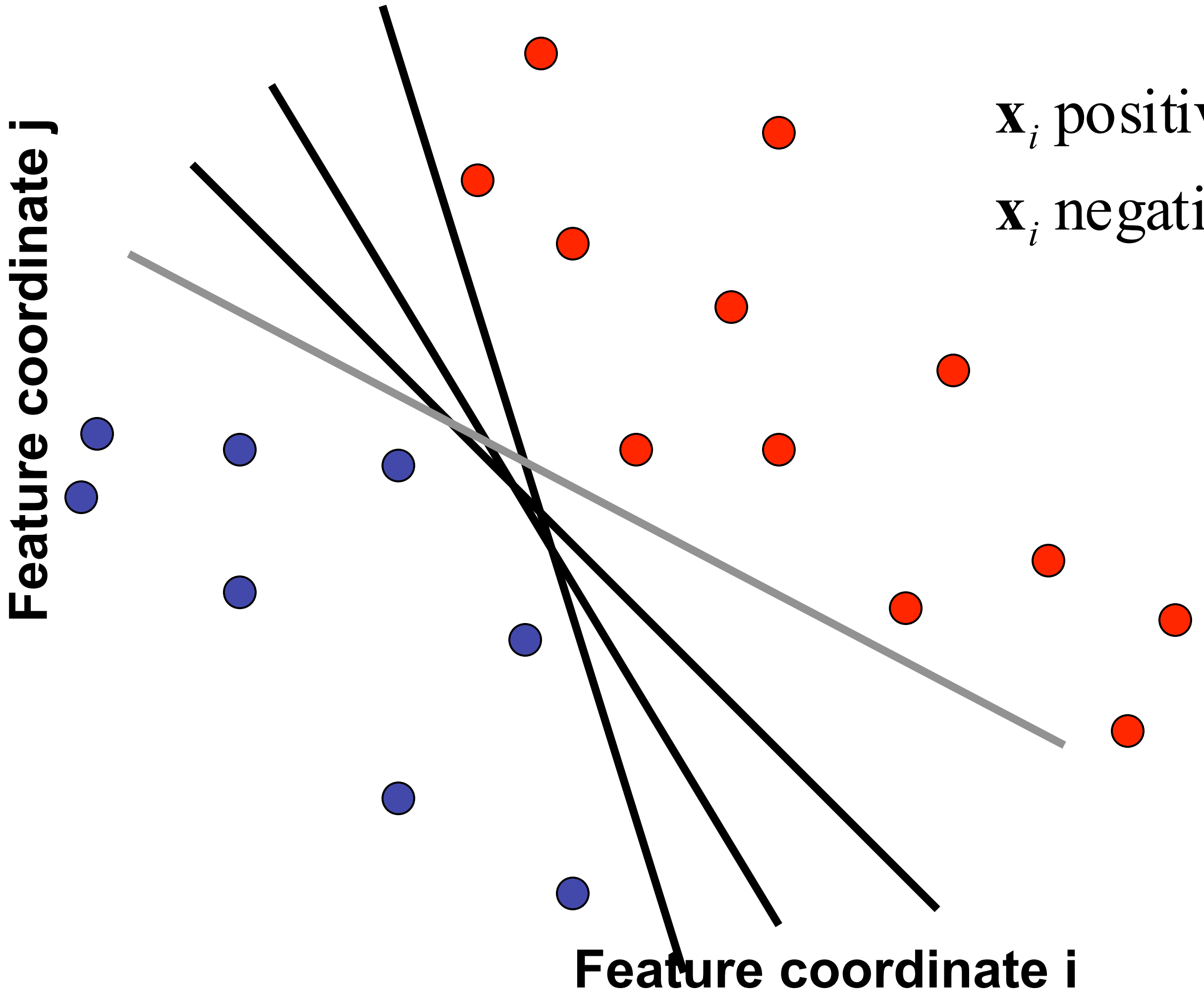
Linear regression: linear $f : R^K \rightarrow R$

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

binary decision can be obtained by thresholding f

Linear Classifiers

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



$$\mathbf{x}_i \text{ positive}: \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 0$$

$$\mathbf{x}_i \text{ negative}: \quad \mathbf{x}_i \cdot \mathbf{w} + b < 0$$

Each data point has a class label:

$$y_t = \begin{cases} +1 \ (\bullet) \\ -1 \ (\circ) \end{cases}$$

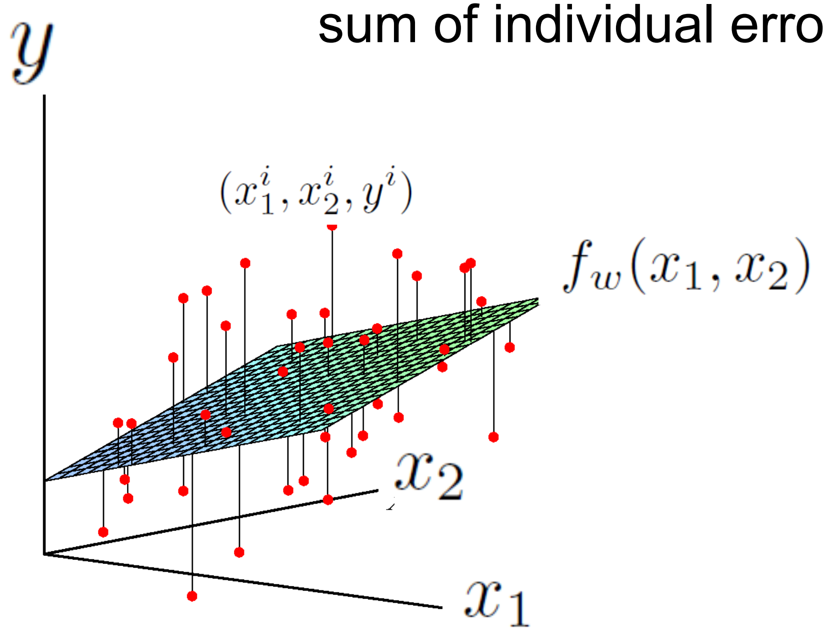
Loss function for linear regression

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

Loss function: quantify appropriateness of w

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

sum of individual errors (`additive') quadratic



Why this loss function?

Easy to optimize!

Least squares solution for linear regression

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

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

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

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

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

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

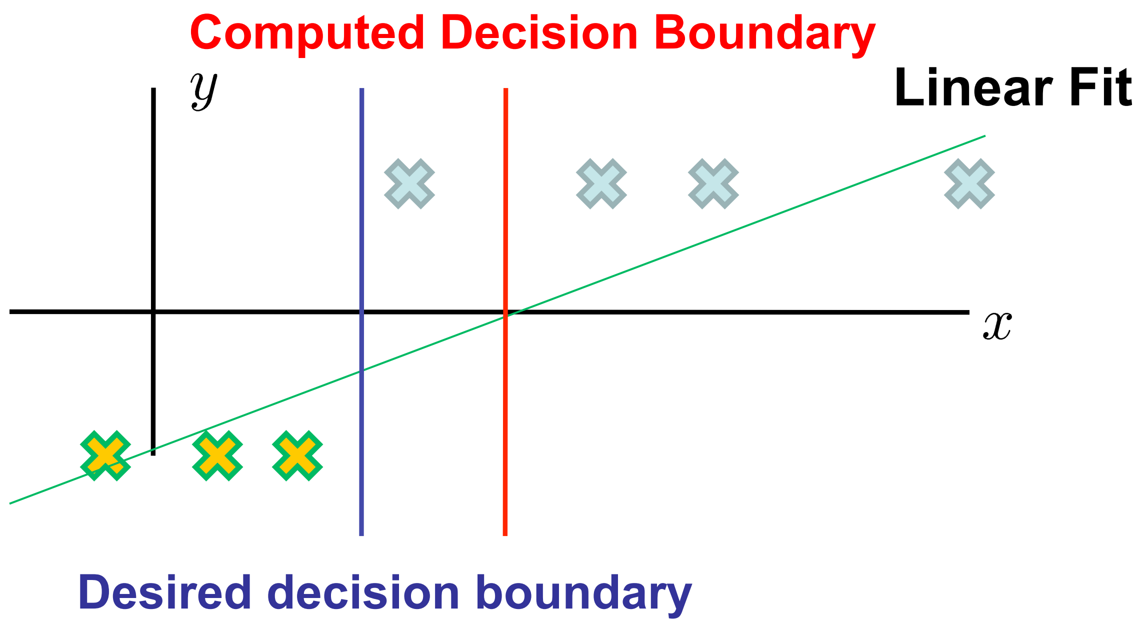
How can we know they do not simply memorize training data?

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?

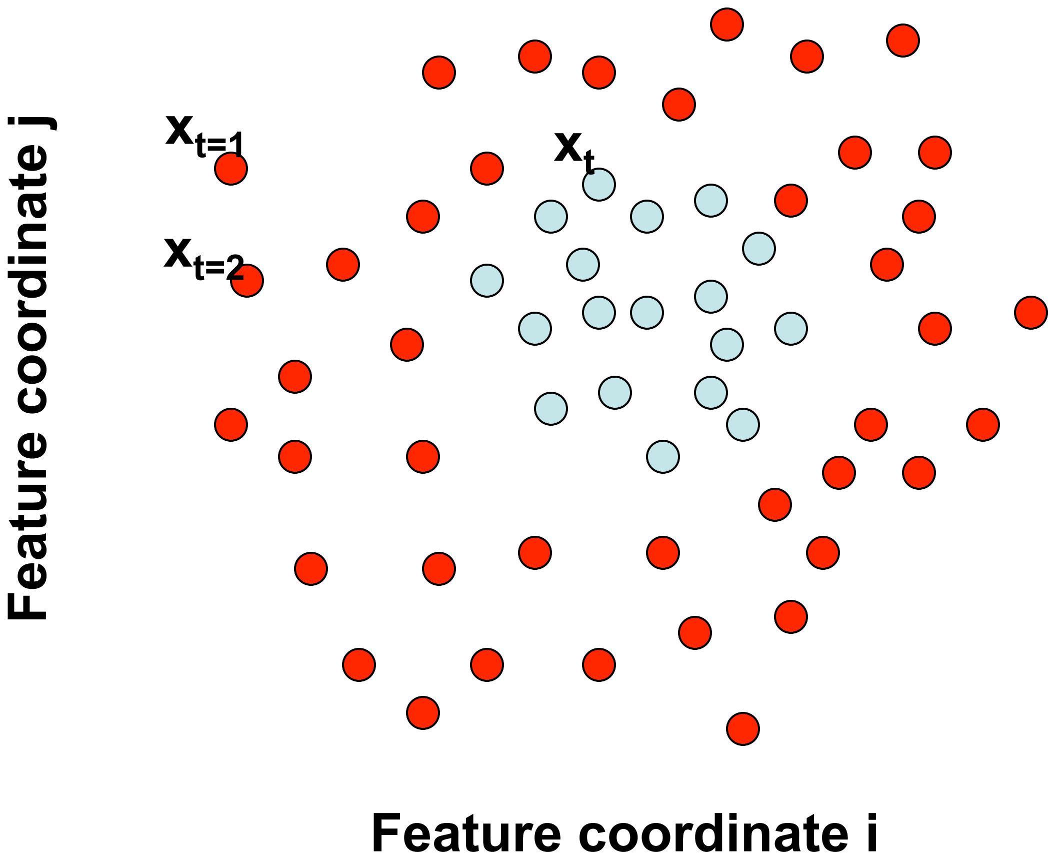
Are the estimated parameters good?

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

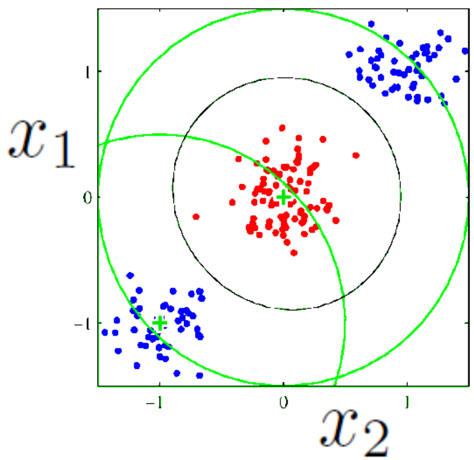


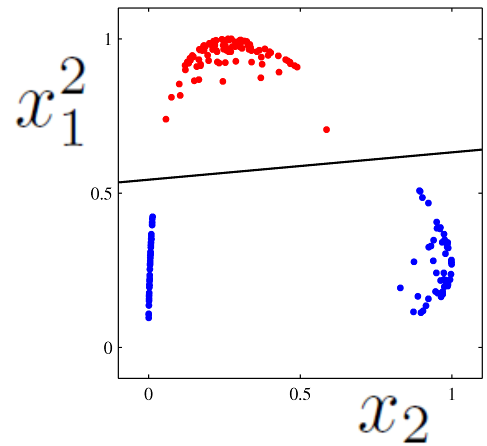
Each data point has a class label:

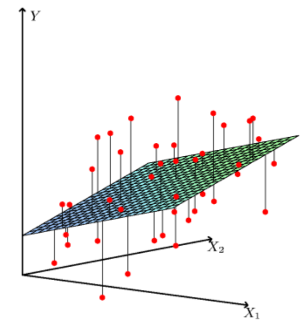
$$y_t = \begin{cases} +1 \; (\;) \\ -1 \; (\;) \end{cases}$$

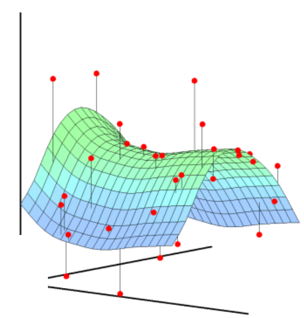
Beyond linear boundaries

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









$$x \to \phi(x) = [\phi_1(x), \ldots, \phi_M(x)]$$

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

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 x 100 pixels (10000 intensity values)

Linear regression: $y^i \simeq f_w(x^i) =< w, x^i >$ $\quad w \in R^{10000}$

$i \in \{1, \ldots, 100\}$

More unknowns than equations: ill posed probler

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

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

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

Full-rank matrix

So how do we set λ ?

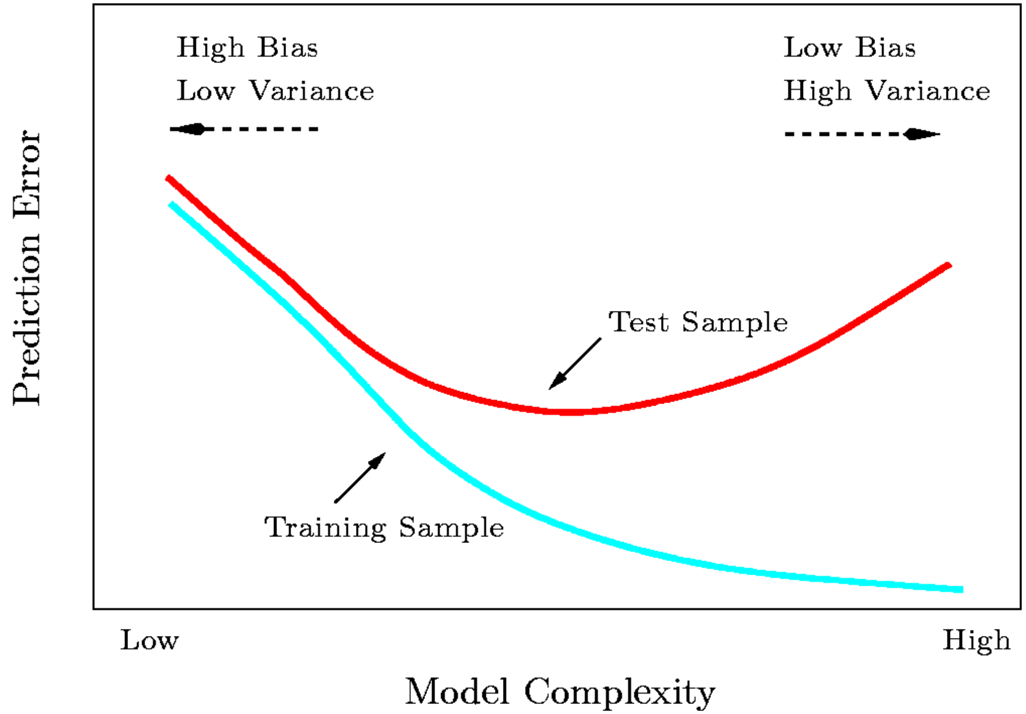
Tuning the model's complexity

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

but can overtrain and have poor performance on the test set

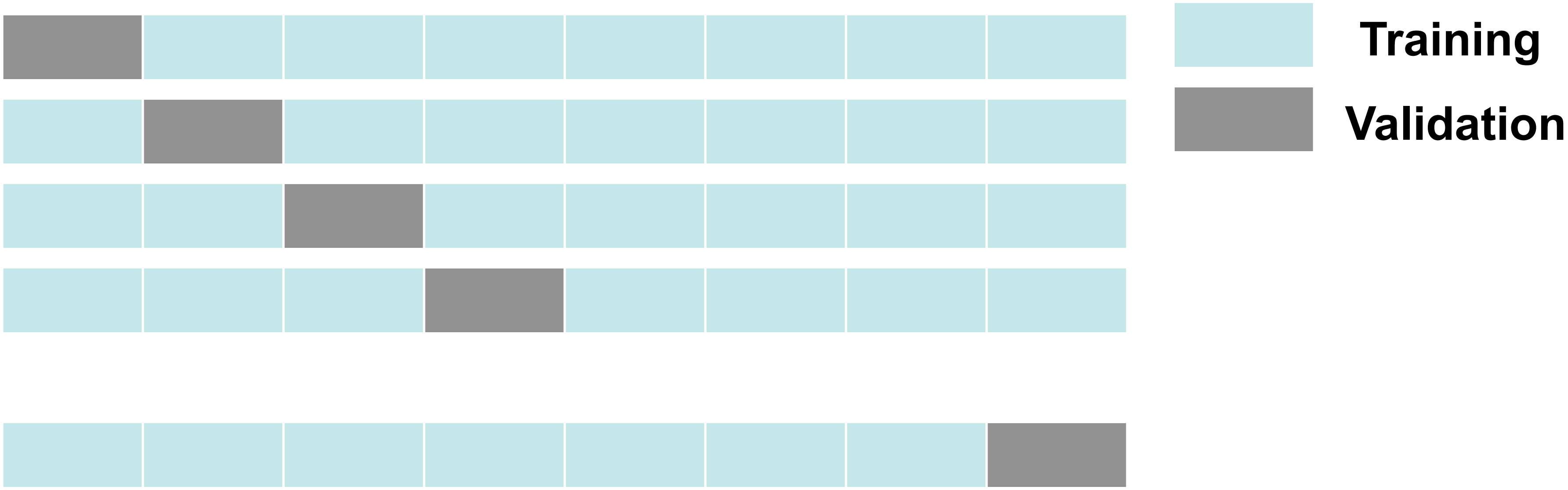
A rigid model's performance is more predictable in the test set

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)

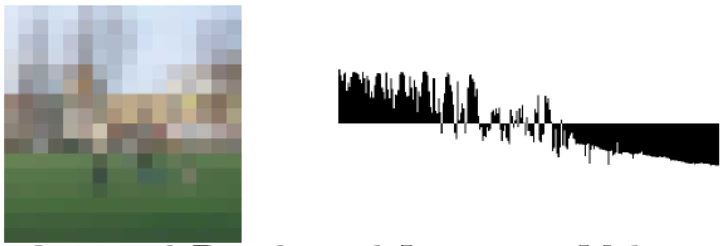
Speech recognition: voice amplitude is irrelevant to uttered words

Computer vision: illumination variations

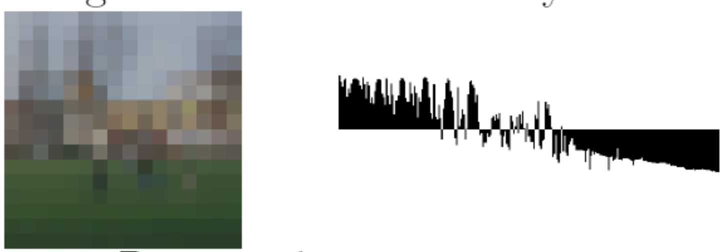
Invariant features: not affected by irrelevant signal transformations

Photometry-invariant patch features

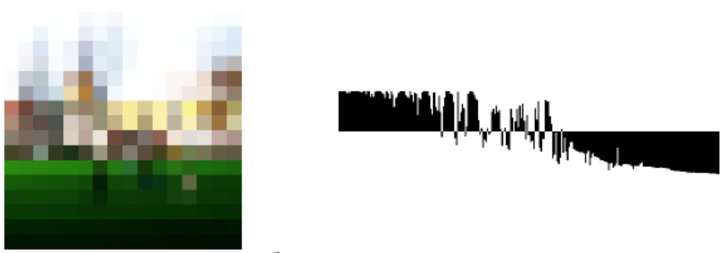
Photometric transformation: $I \rightarrow a\,I + b$



Original Patch and Intensity Values



Brightness Decreased



Contrast increased,

- Make each patch have zero mean:

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

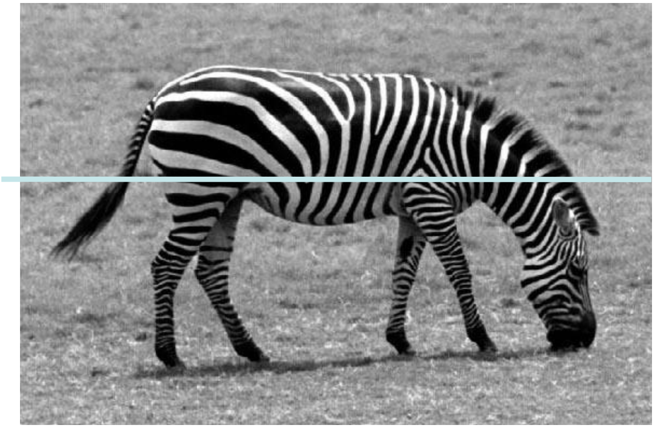
$$Z(x,y) = I(x,y) - \mu$$

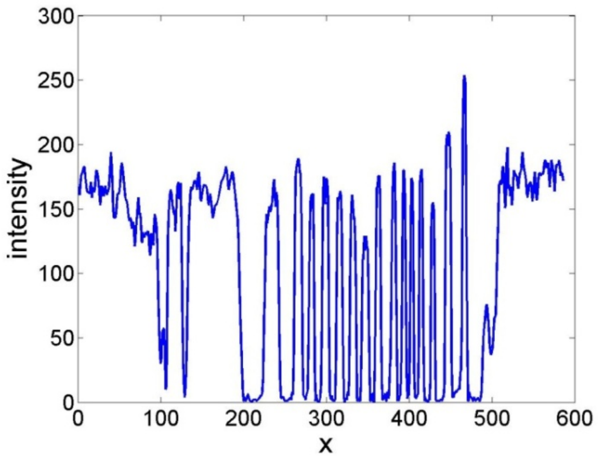
- Then make it have unit variance:

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

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

Dealing with texture





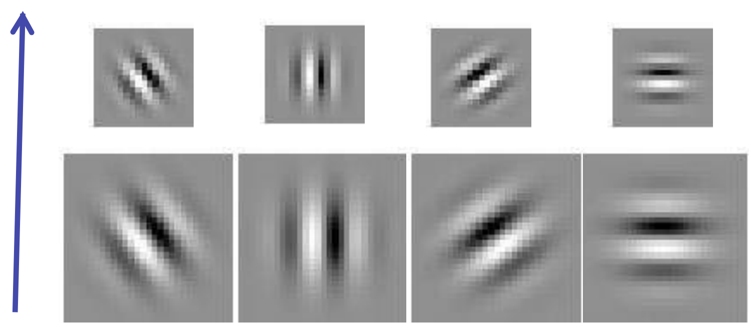
What kind of features can appropriately describe texture patterns?

`appropriately': in terms of well-behaved functions

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

Increasing

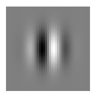
$|\omega| = \sqrt{\omega_1^2+\omega_2^2}$

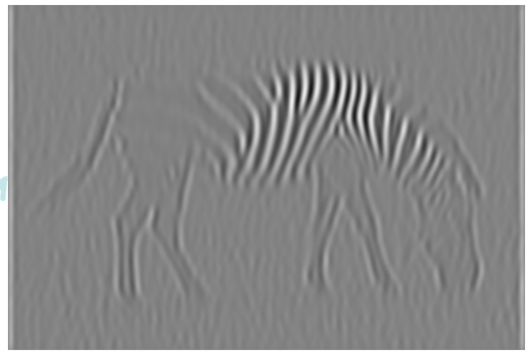


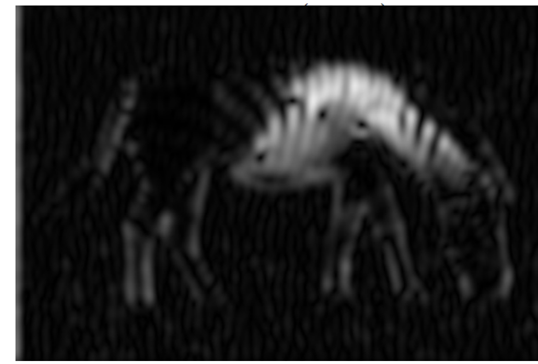
Envelope estimation (demodulation)

$$G_o(x,y) \quad = \quad G^{\mathrm{odd}}_{\sigma,\vec{\omega}}(x,y) * f(x,a(x,y) \quad = \quad \sqrt{G_e^2(x,y) + G_o^2(x,y)^2}$$

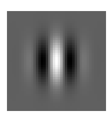
Convolve

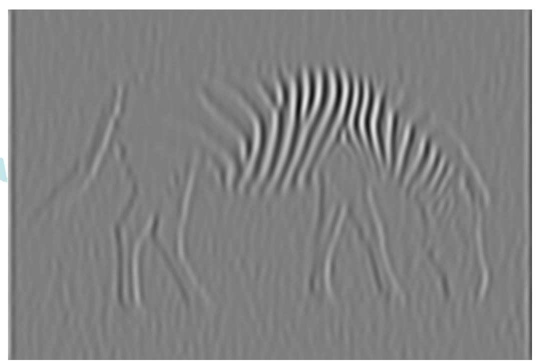


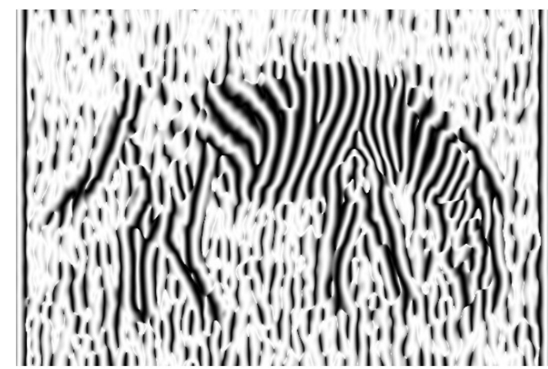




$$G_e(x,y) \quad = \quad G^{\mathrm{even}}_{\sigma,\vec{\omega}}(x,y) * f(x,y) \qquad \phi(x,y) \quad = \quad \tan^{-1}\frac{G_o(x,y)}{G_e(x,y)}$$

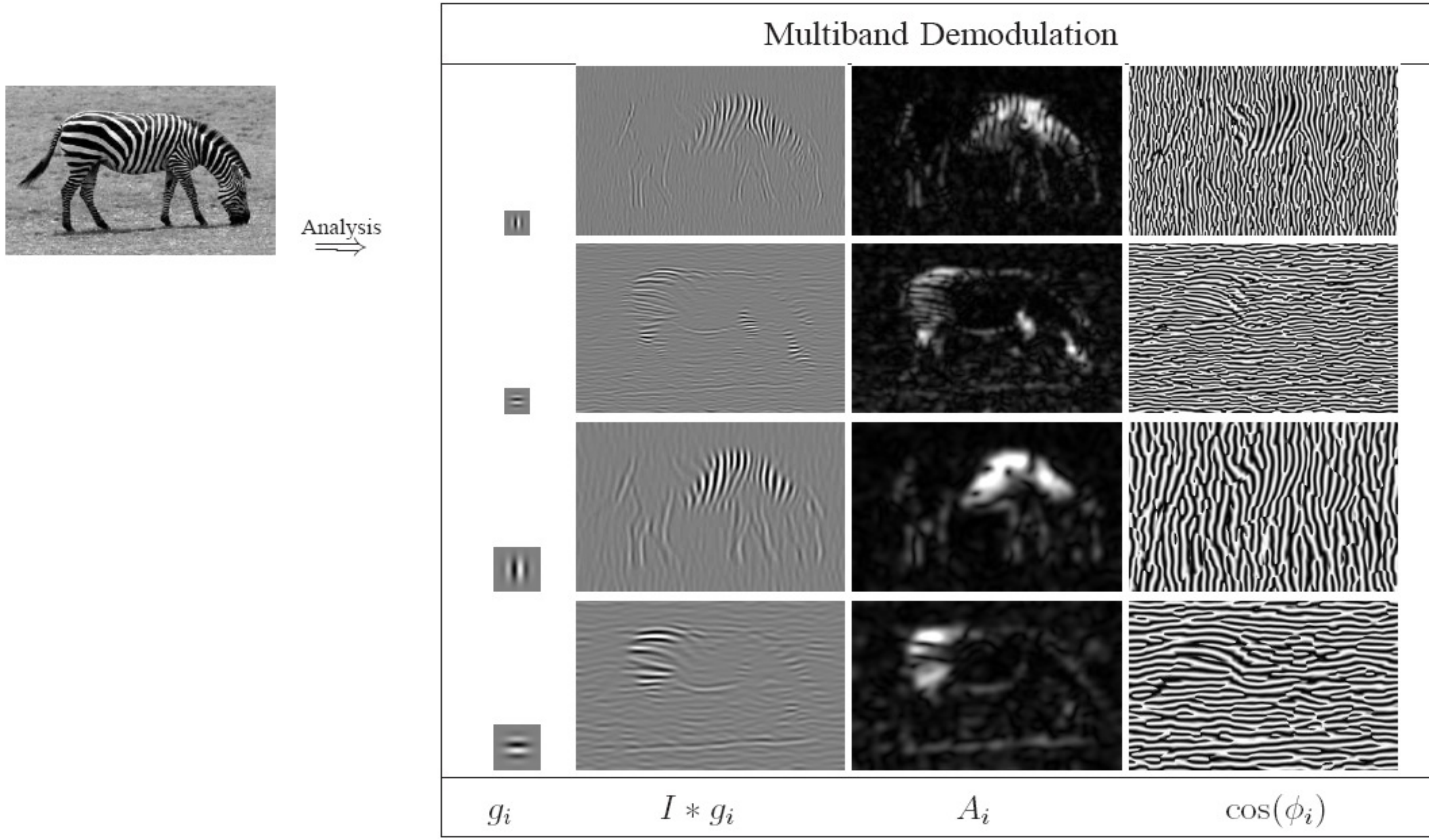






$$I(x,y) = a(x,y)\cos(\phi(x,y)) + c, \qquad \vec{\omega}(x,y) = \nabla\phi(x,y)$$

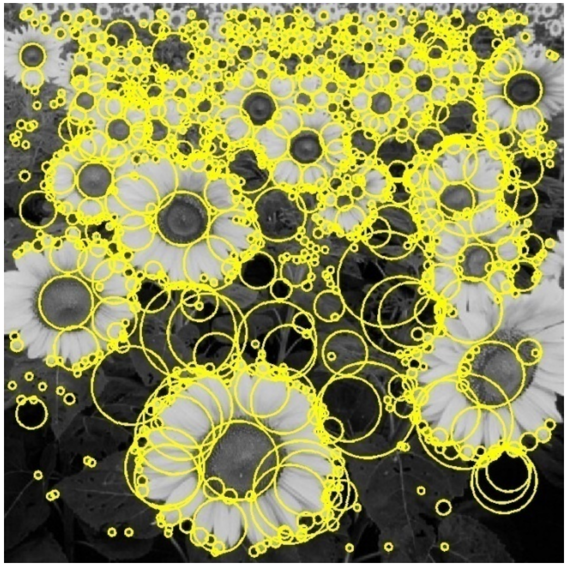
Multiband demodulation with a Gabor filterbank



Havlicek & Bovik, IEEE TIP ’00

Dealing with changes in scale and orientation

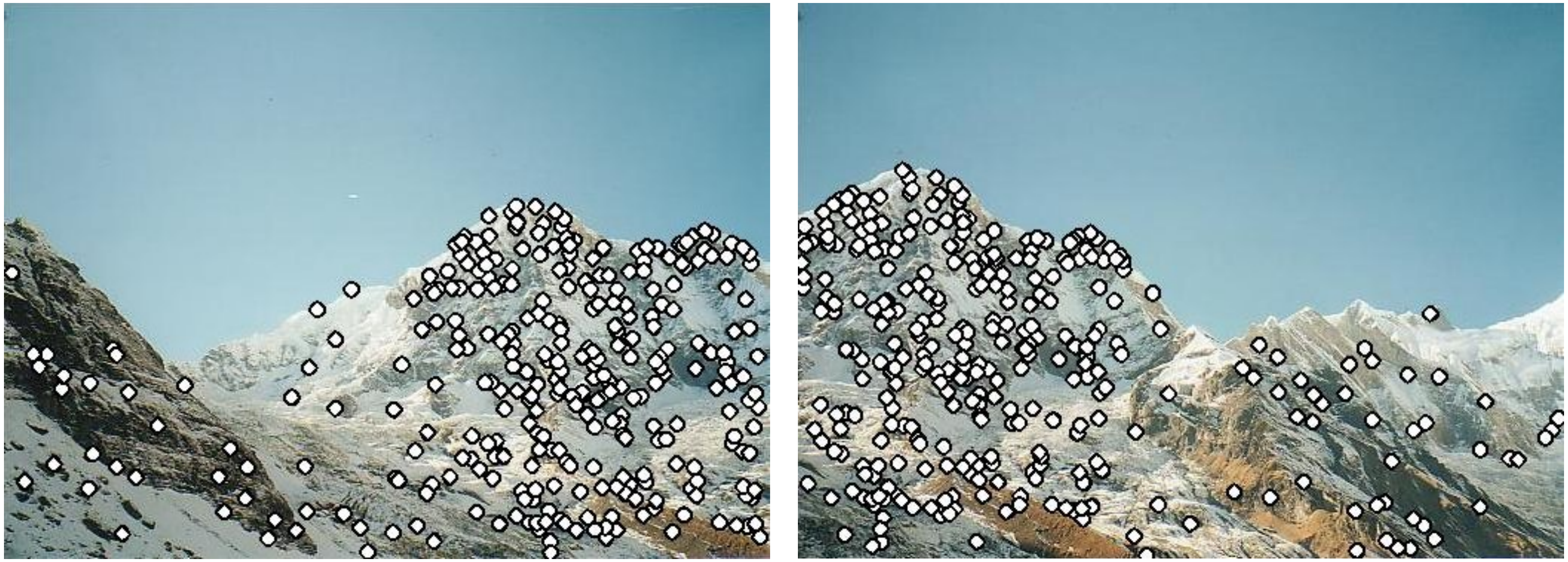
Scale-invariant blob detector



Application: Image Stitching

Slide credit: Darya Frolova, Denis Simakov

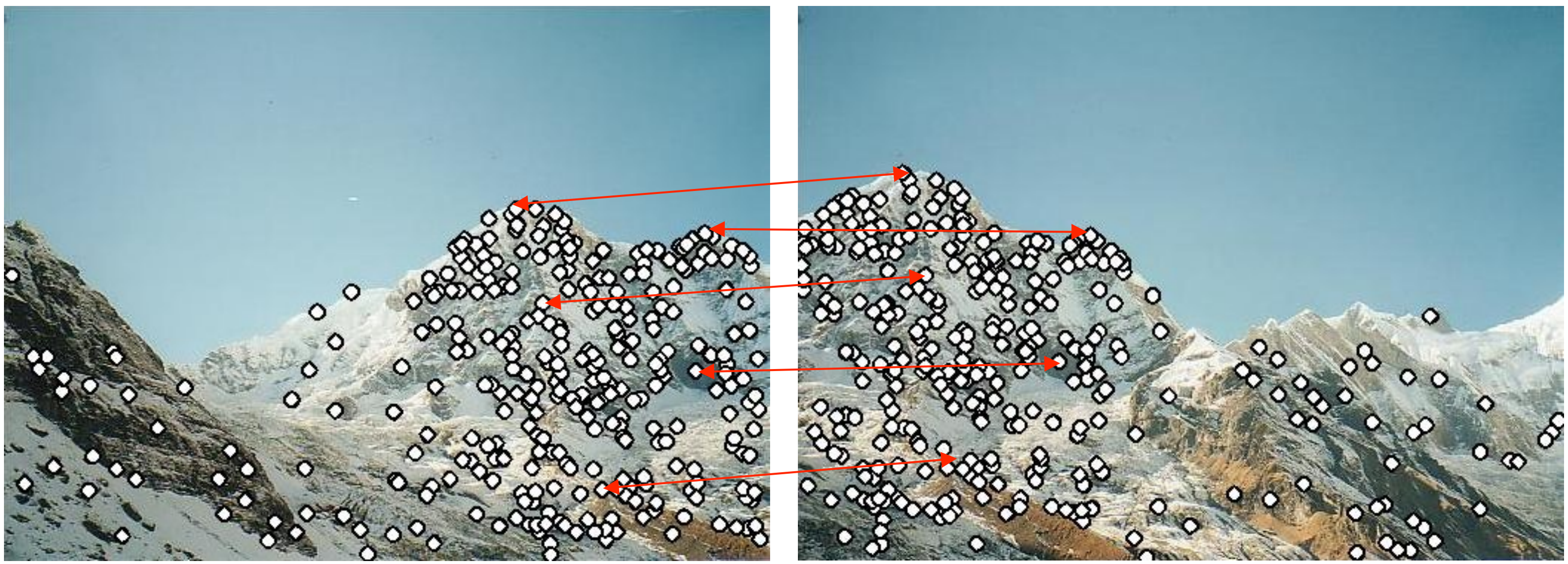
Application: Image Stitching



- Procedure:
  - Detect feature points in both images

Slide credit: Darya Frolova, Denis Simakov

Application: Image Stitching



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

Slide credit: Darya Frolova, Denis Simakov

Application: Image Stitching

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

Slide credit: Darya Frolova, Denis Simakov

Common Requirements

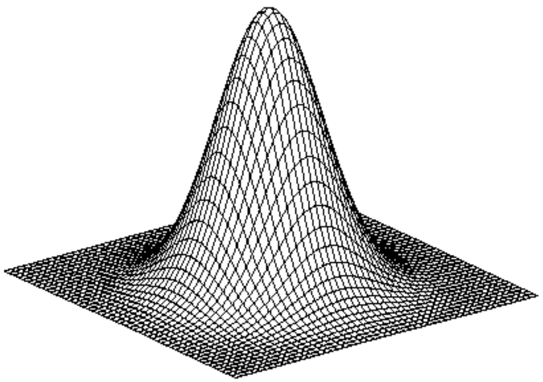
- Problem 1:
  - Detect the same point *independently* in both images

No chance to match!

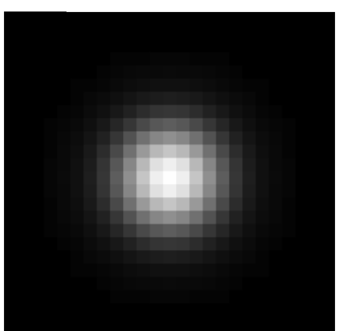
Slide credit: Darya Frolova, Denis Simakov

Laplacian-of-Gaussian

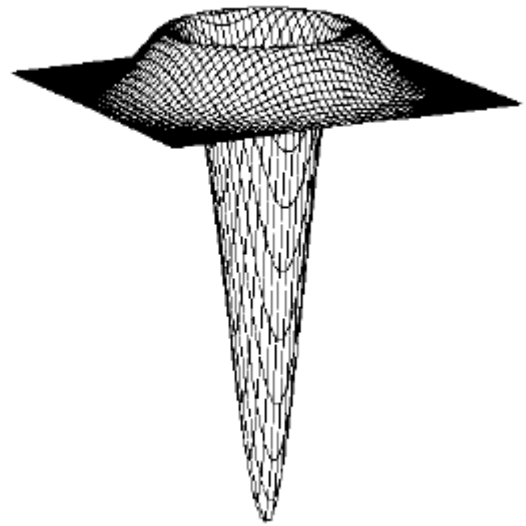
Gaussian



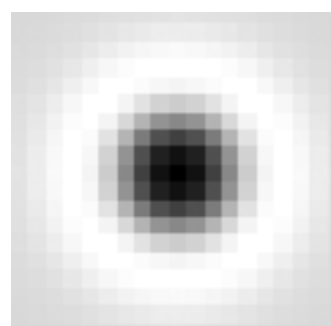
$$g_\sigma(x, y)$$



Laplacian of Gaussian

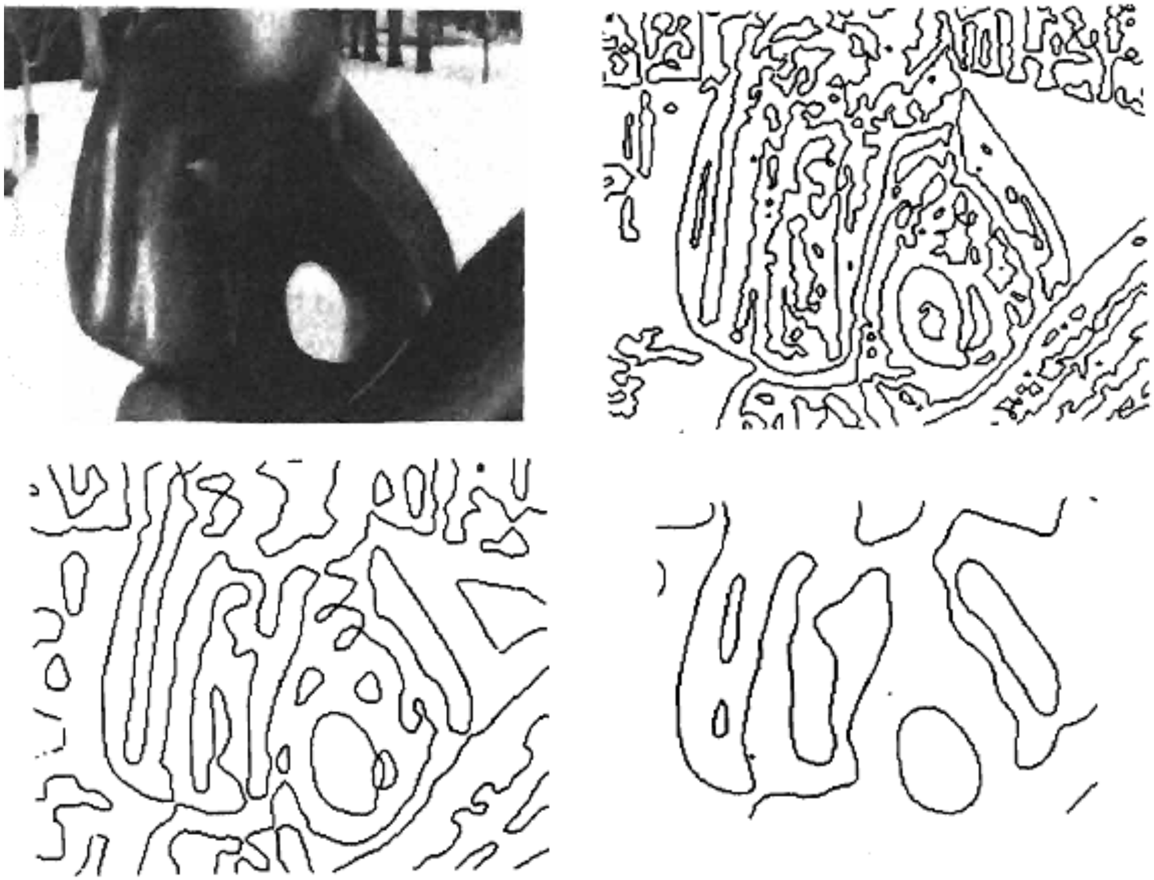


$$\nabla^2 g_\sigma(x, y) = \frac{\partial^2 g_\sigma(x, y)}{\partial x^2} + \frac{\partial^2 g_\sigma(x, y)}{\partial y^2}$$



Early edge detection research

- Zero-crossings of LoG operator at increasing scales

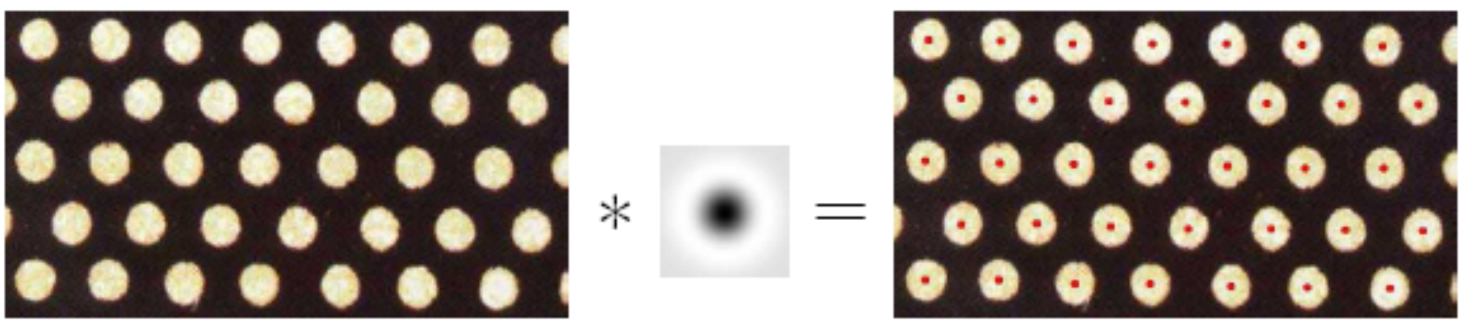


- Different take: go for the maxima/minima

Finding blobs

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

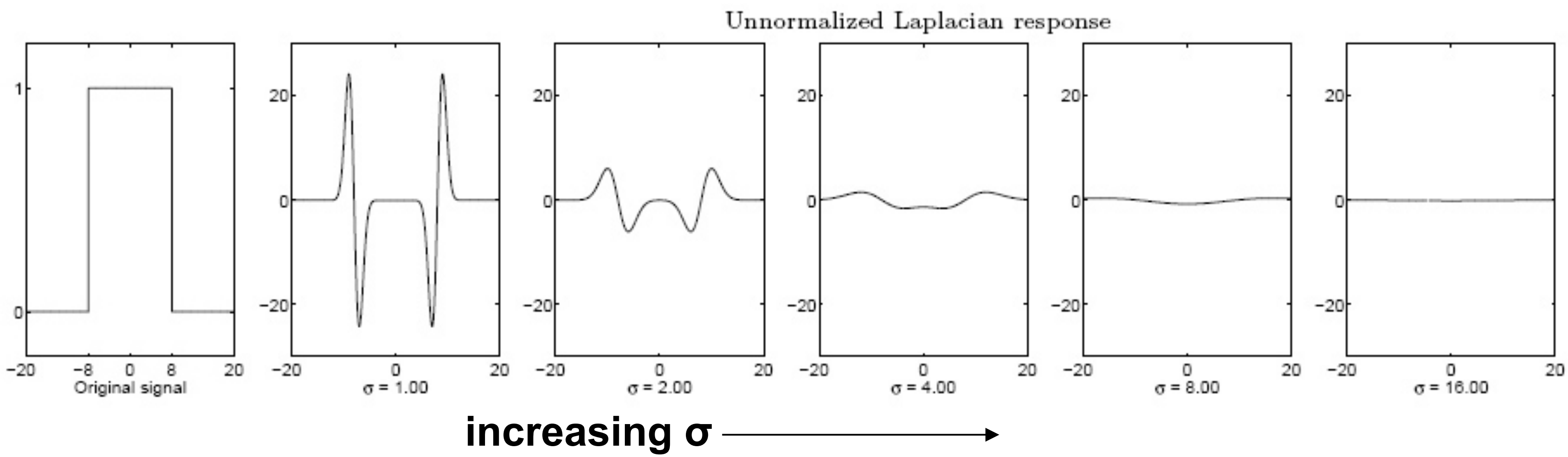
$$
\begin{aligned}
|I - f|^2 &= \langle I - f, I - f \rangle \\
&= \langle I, I \rangle + \langle f, f \rangle - 2\langle f, I \rangle \\
&= C - 2\langle I, f \rangle
\end{aligned}
$$



Polka Dots Detected Blobs

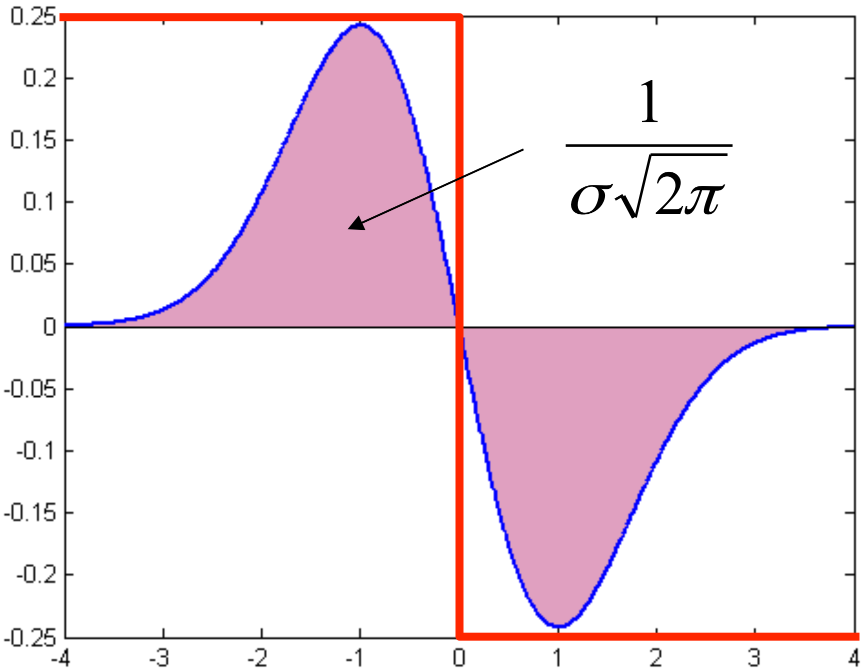
Scale selection

- 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

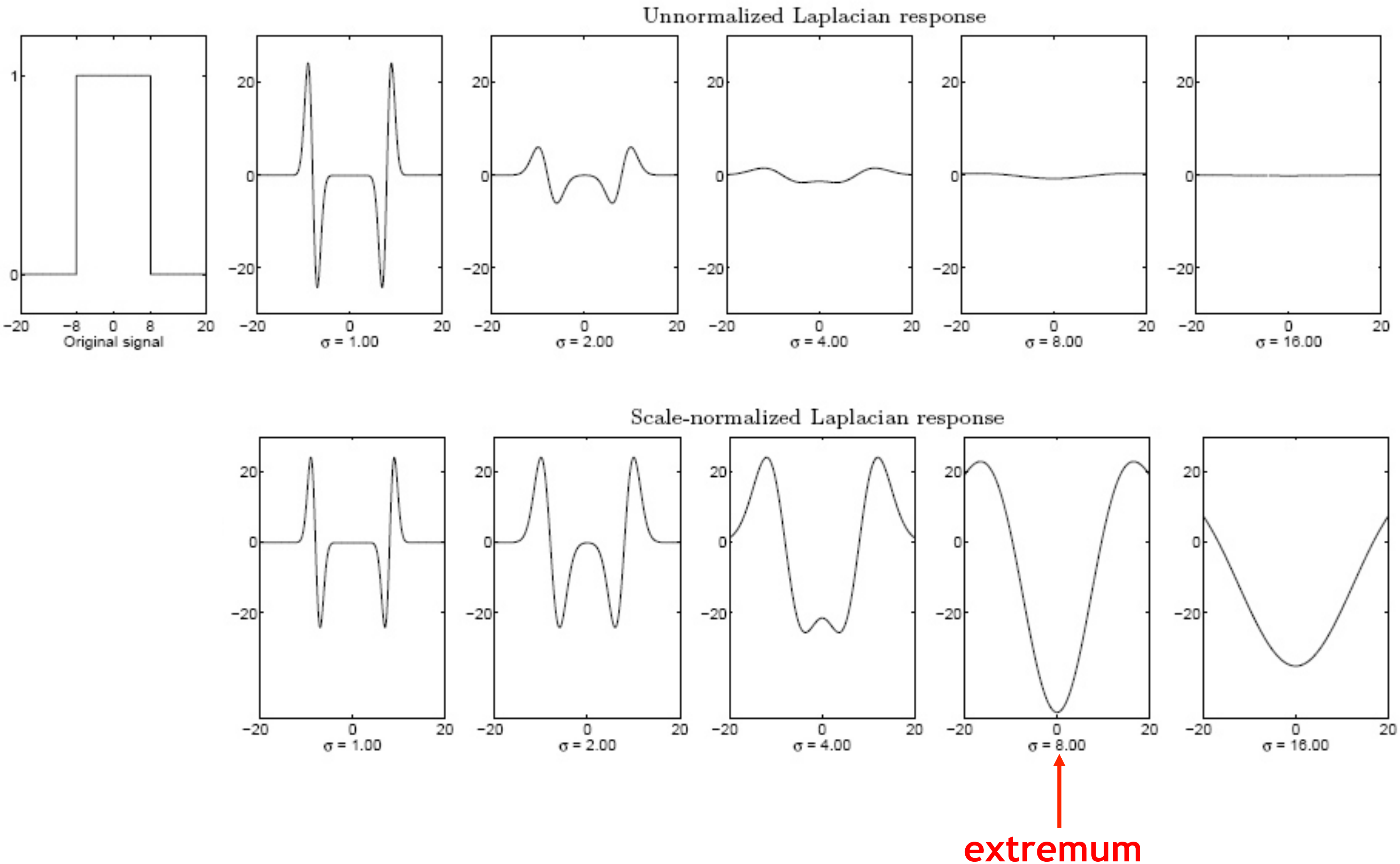


$$\frac{1}{\sigma\sqrt{2\pi}}$$

Scale normalization

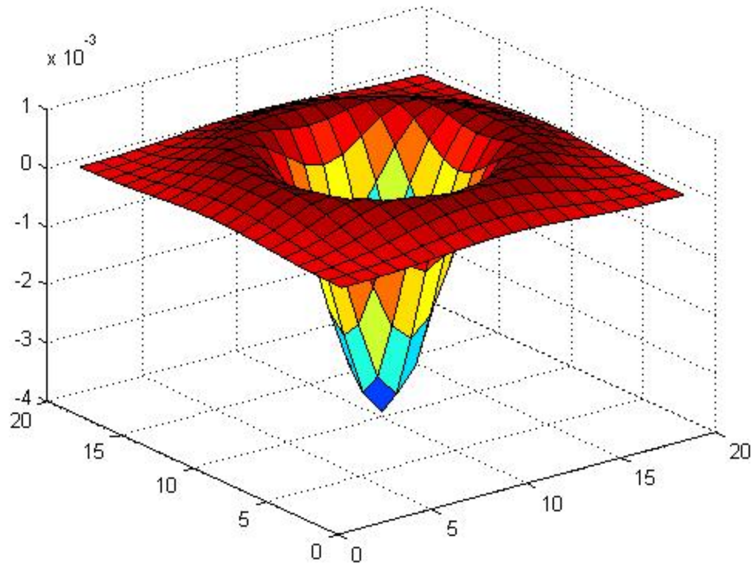
- The response of a derivative of Gaussian filter to a perfect step edge decreases as σ increases
- To keep response the same (scale-invariant), must multiply Gaussian derivative by σ
- Laplacian is the second Gaussian derivative, so it must be multiplied by σ^2

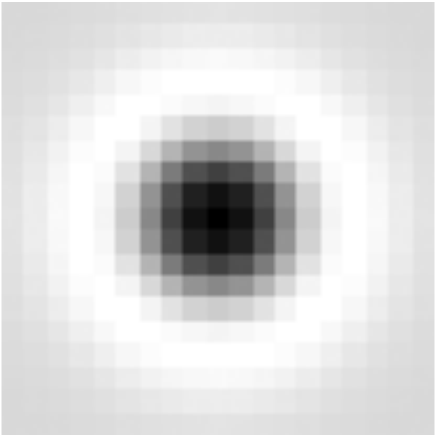
Effect of scale normalization



Blob detection in 2D

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

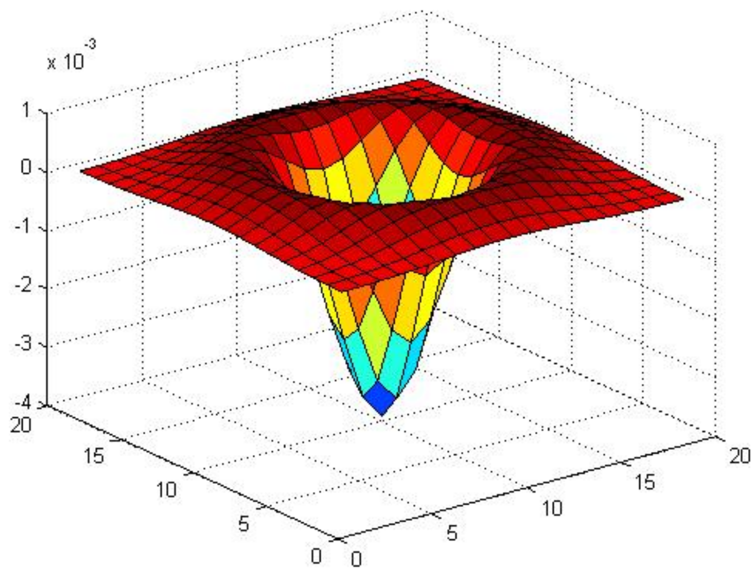


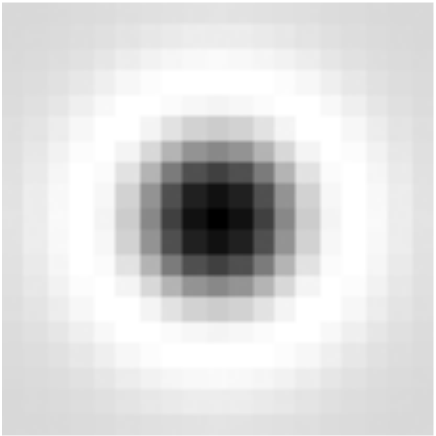


$$\nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2}$$

Blob detection in 2D

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



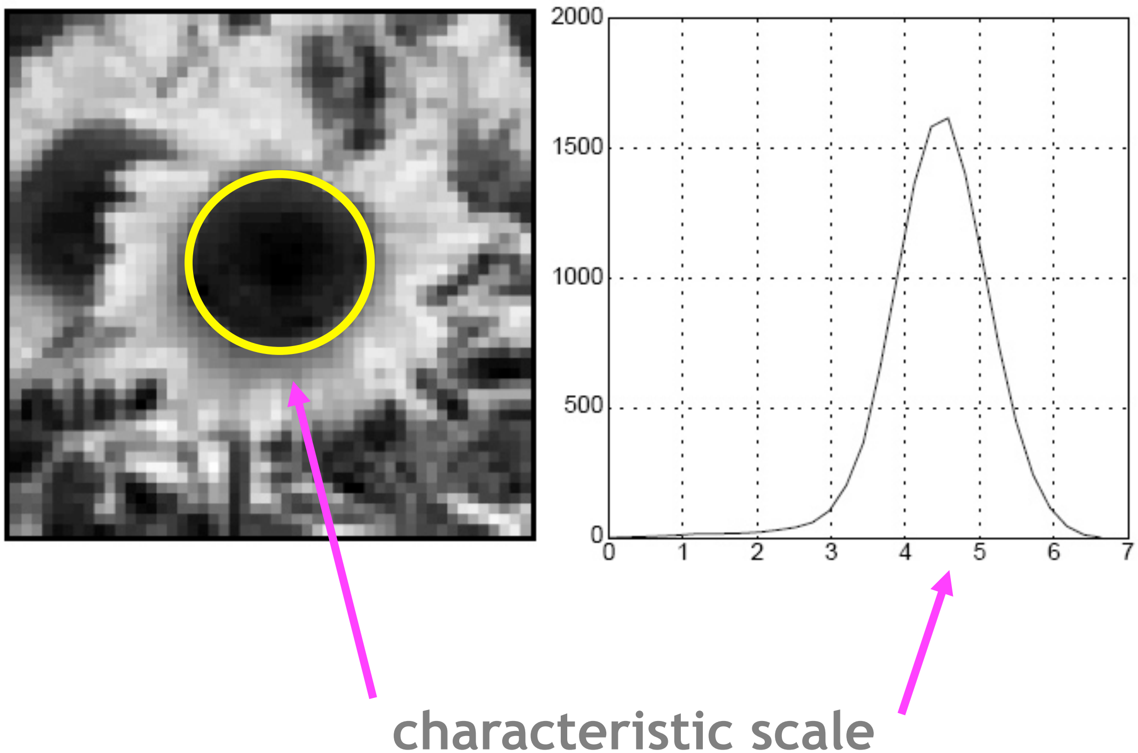


Scale-normalized:

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

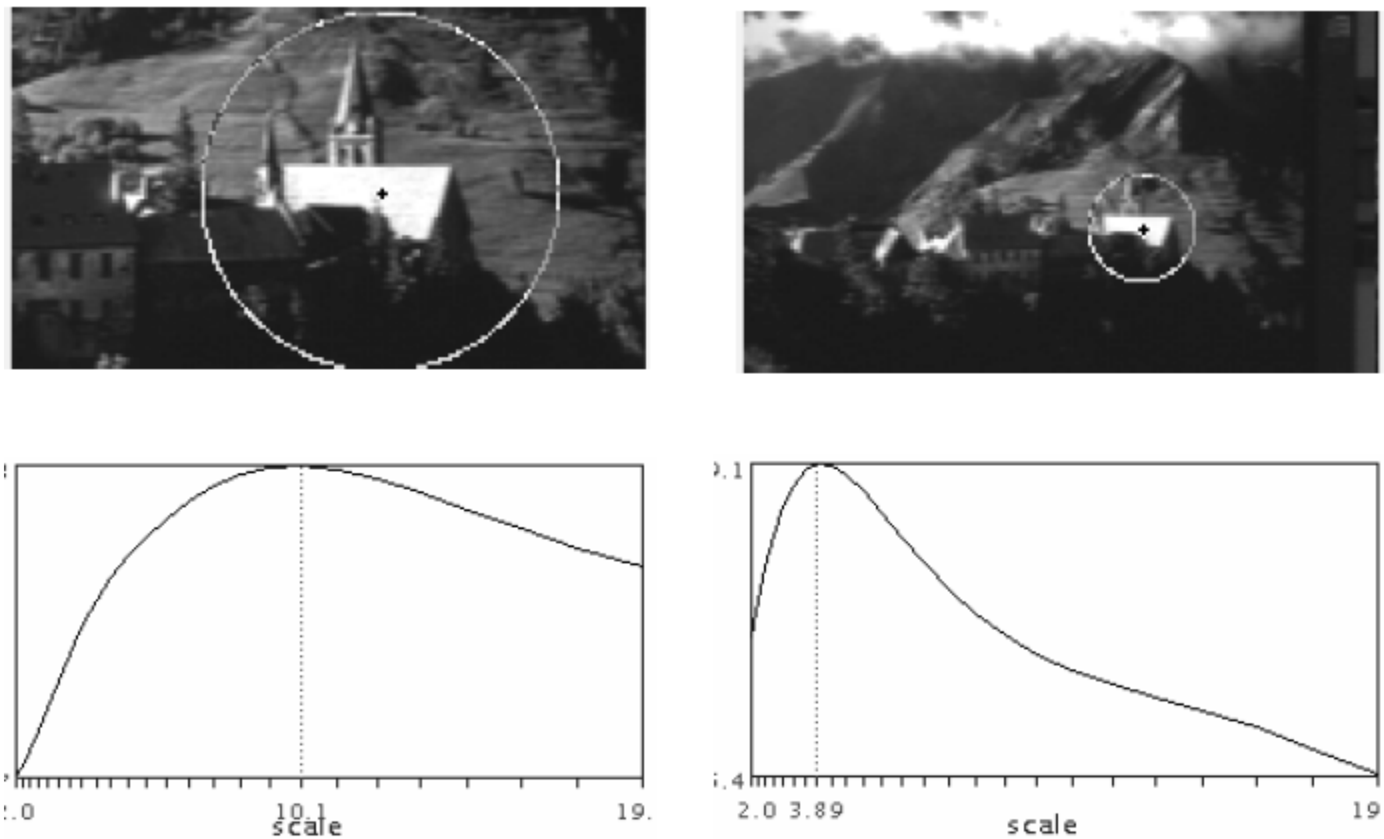
Scale selection

- 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

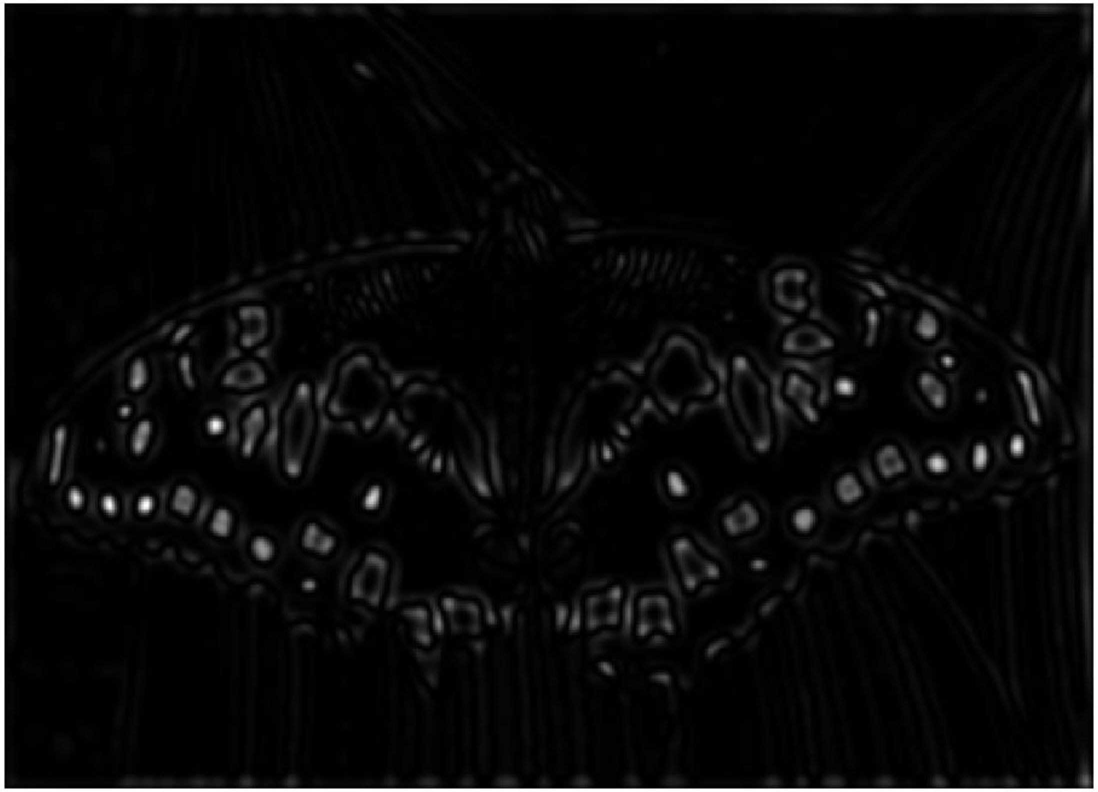


Laplacian

Scale-space blob detector: Example

Slide credit: S. Lazebnik

Scale-space blob detector: Example



sigma = 3.1296

Slide credit: S. Lazebnik

Scale-space blob detector: Example

sigma = 4.8972

Slide credit: S. Lazebnik

Scale-space blob detector: Example

sigma = 7.6631

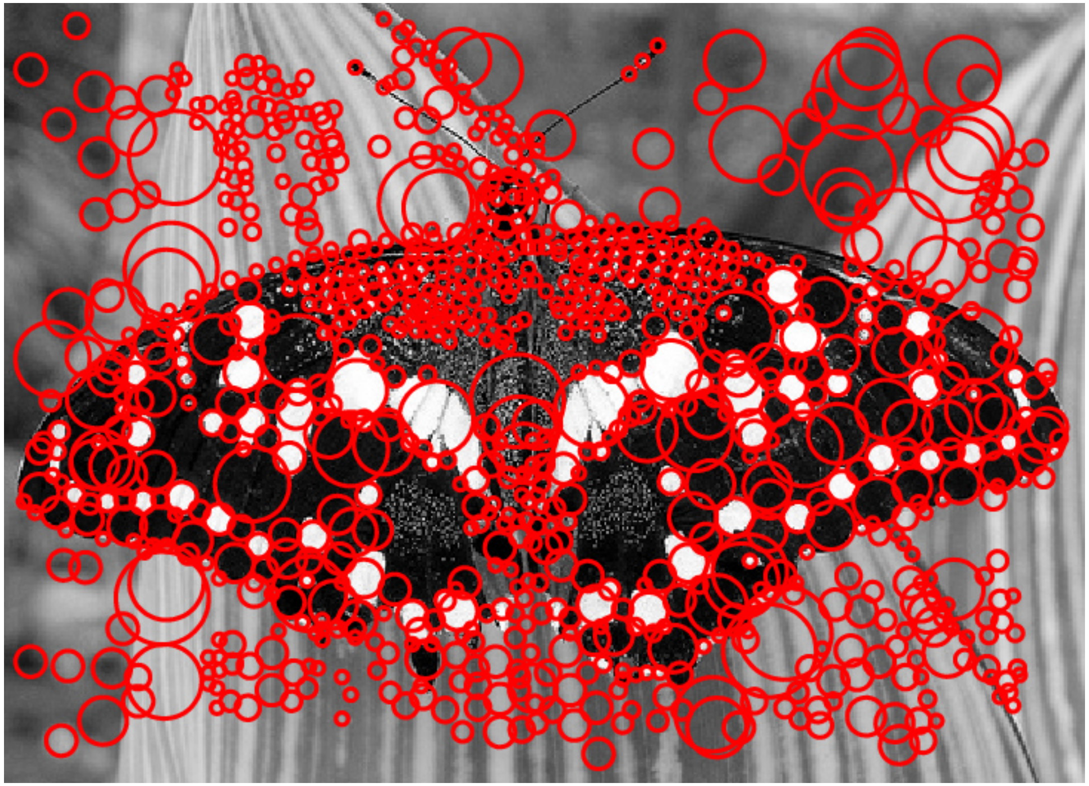
Slide credit: S. Lazebnik

Scale-space blob detector: Example

sigma = 11.9912

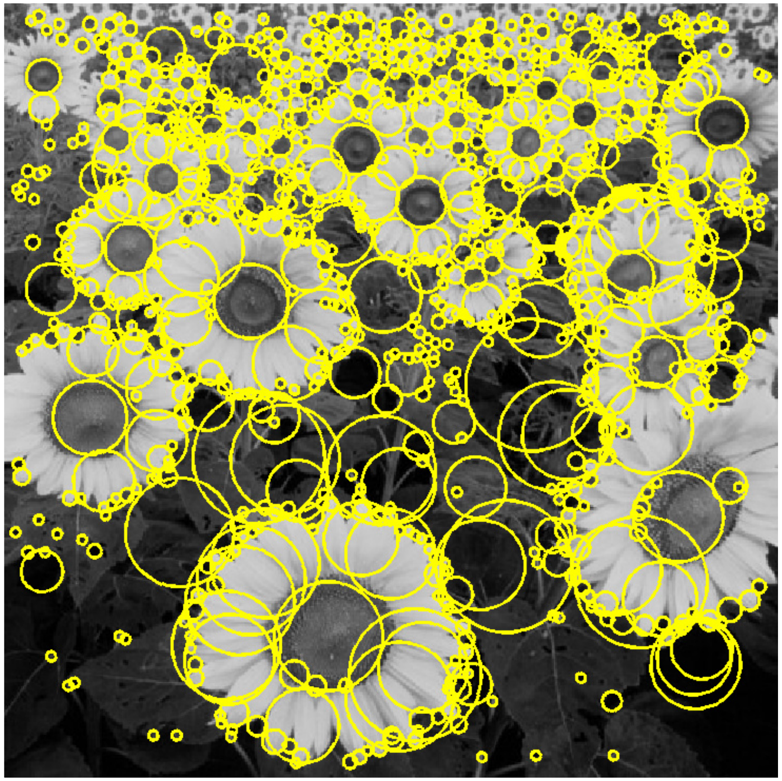
Slide credit: S. Lazebnik

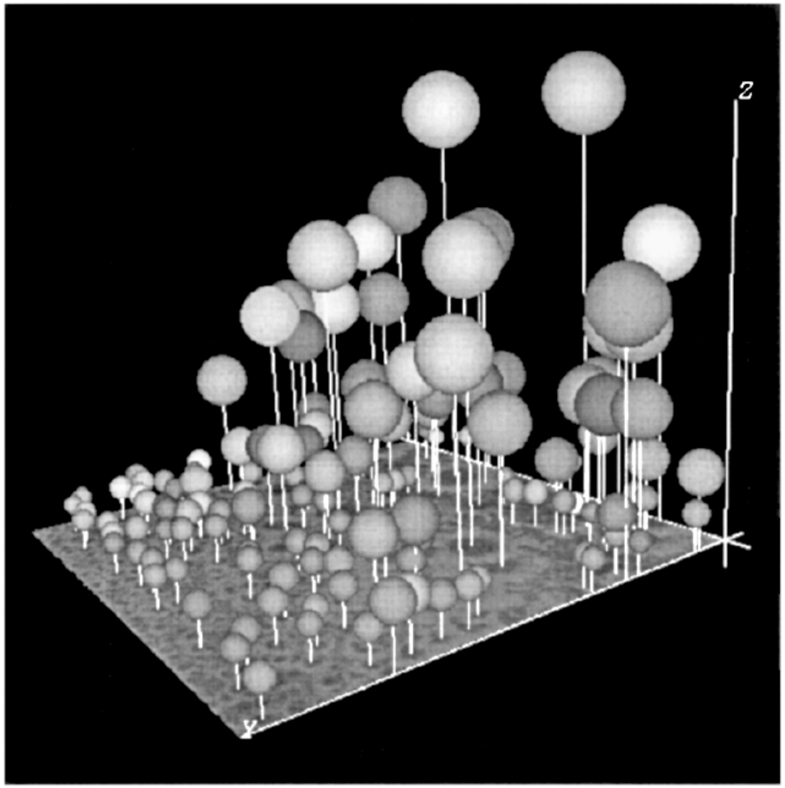
Scale-space blob detector: Example



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

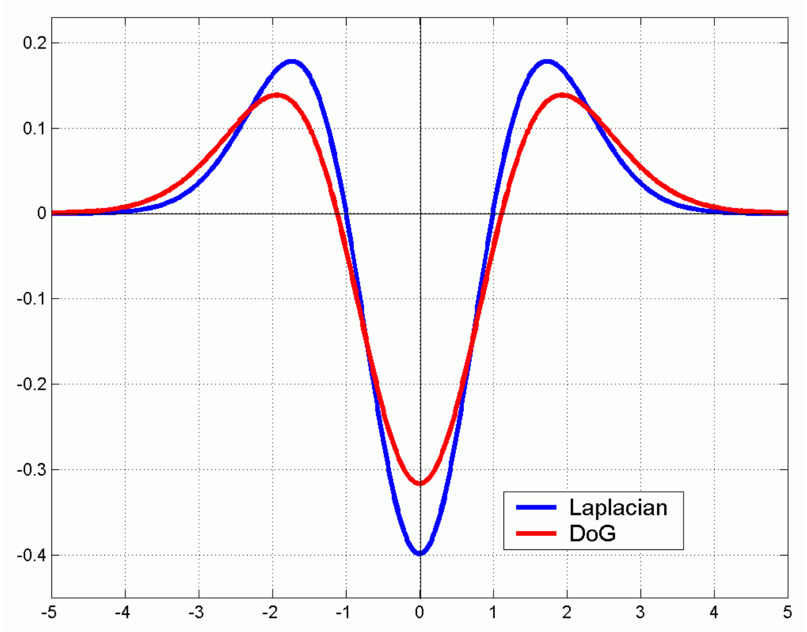
- We can efficiently approximate the Laplacian with a difference of Gaussians:

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

(Laplacian)

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

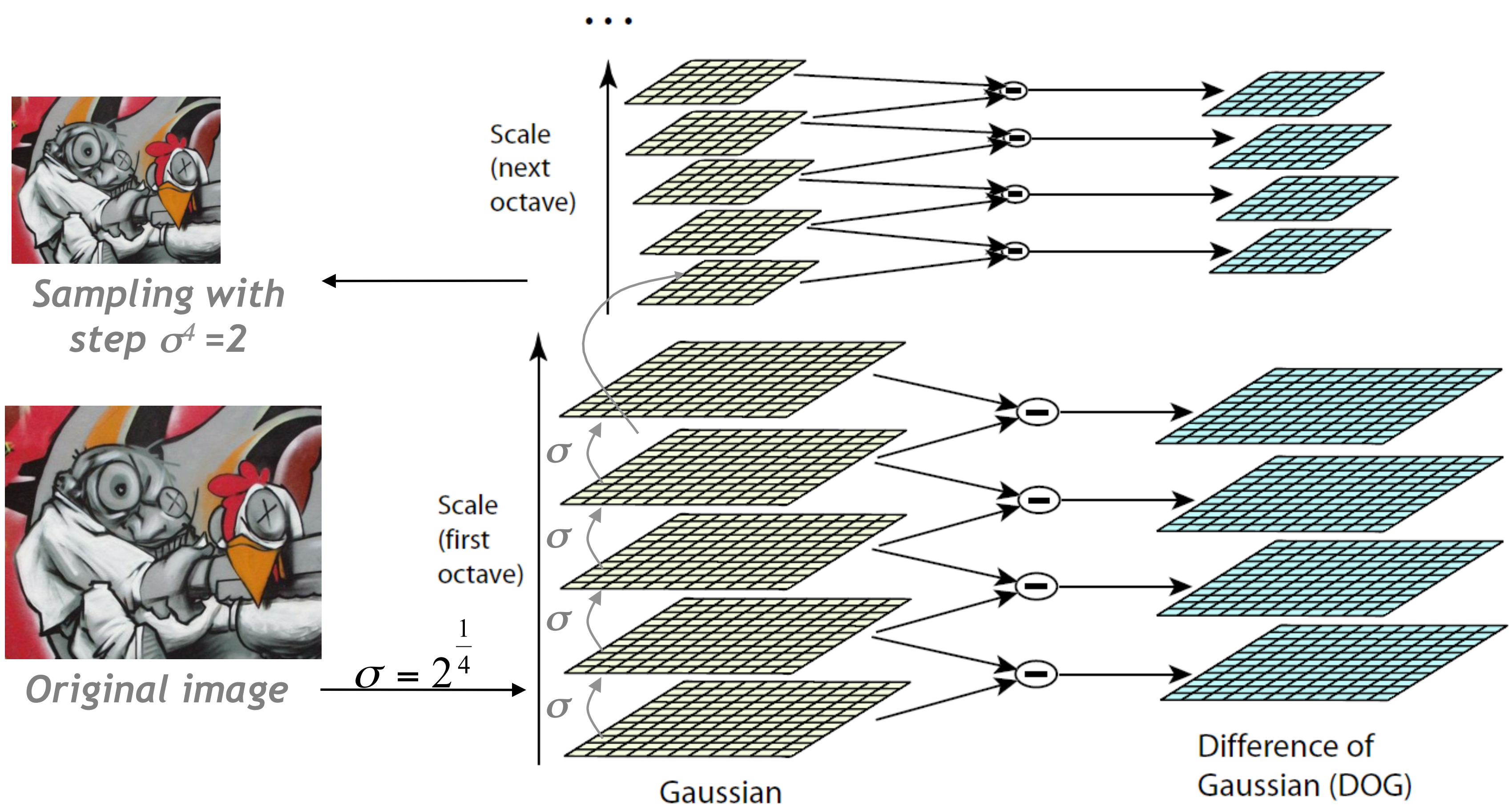
(Difference of Gaussians)



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

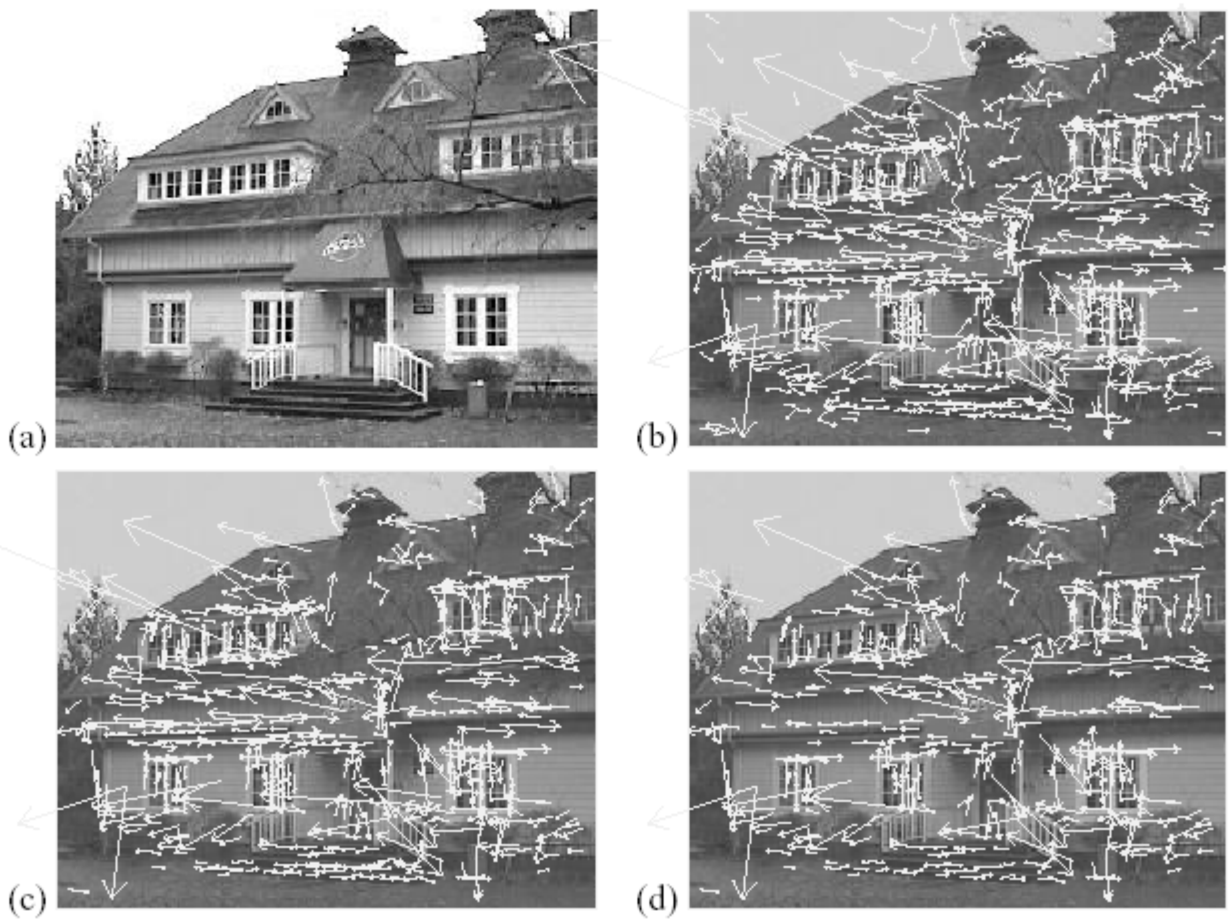
Efficient Computation (SIFT)

- Computation in Gaussian scale pyramid



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

Keypoind Detection (SIFT)



(a) 233x189 image

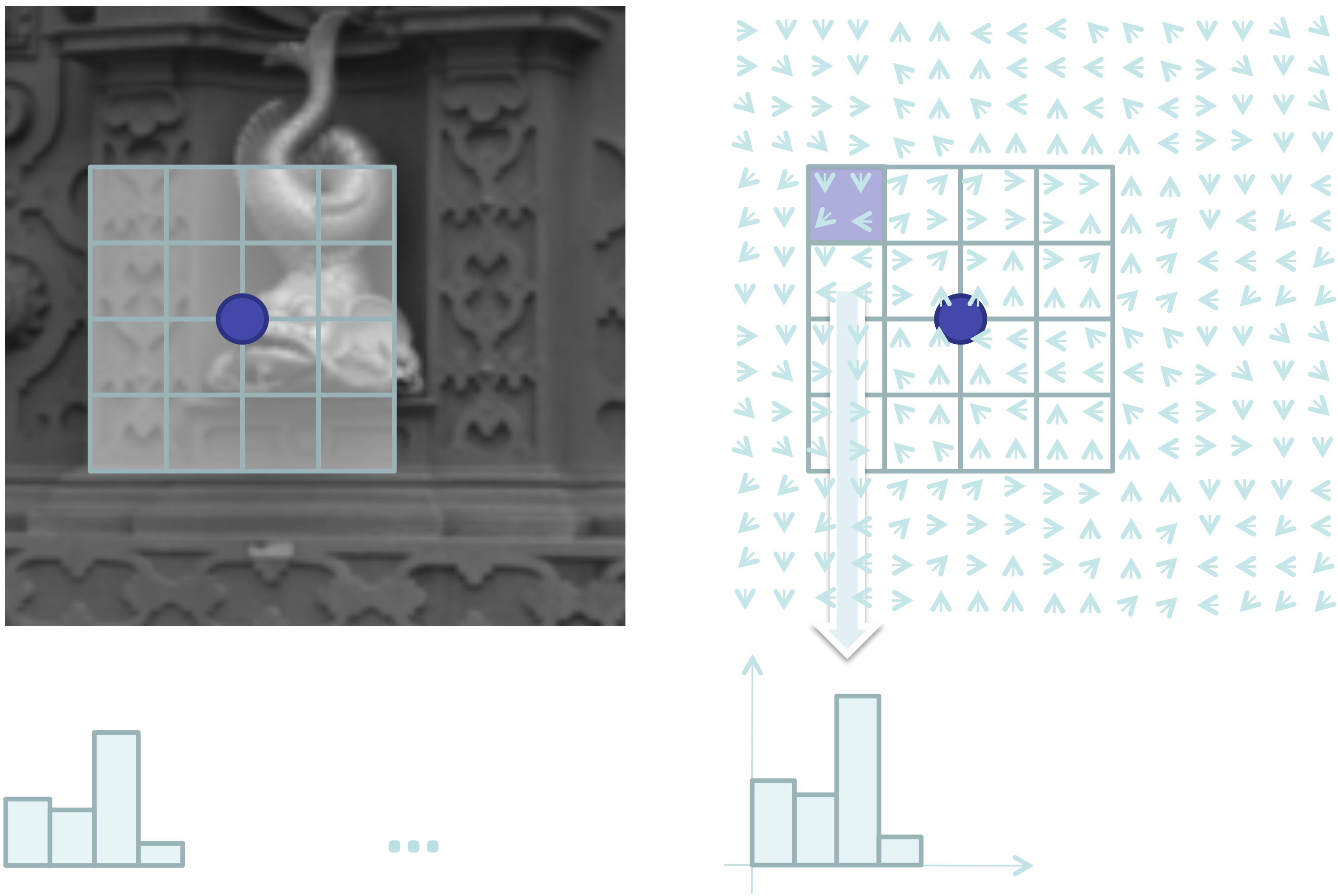
(b) 832 DoG extrema

(c) 729 left after peak value threshold

(d) 536 left after testing ratio of principle curvatures (removing edge responses)

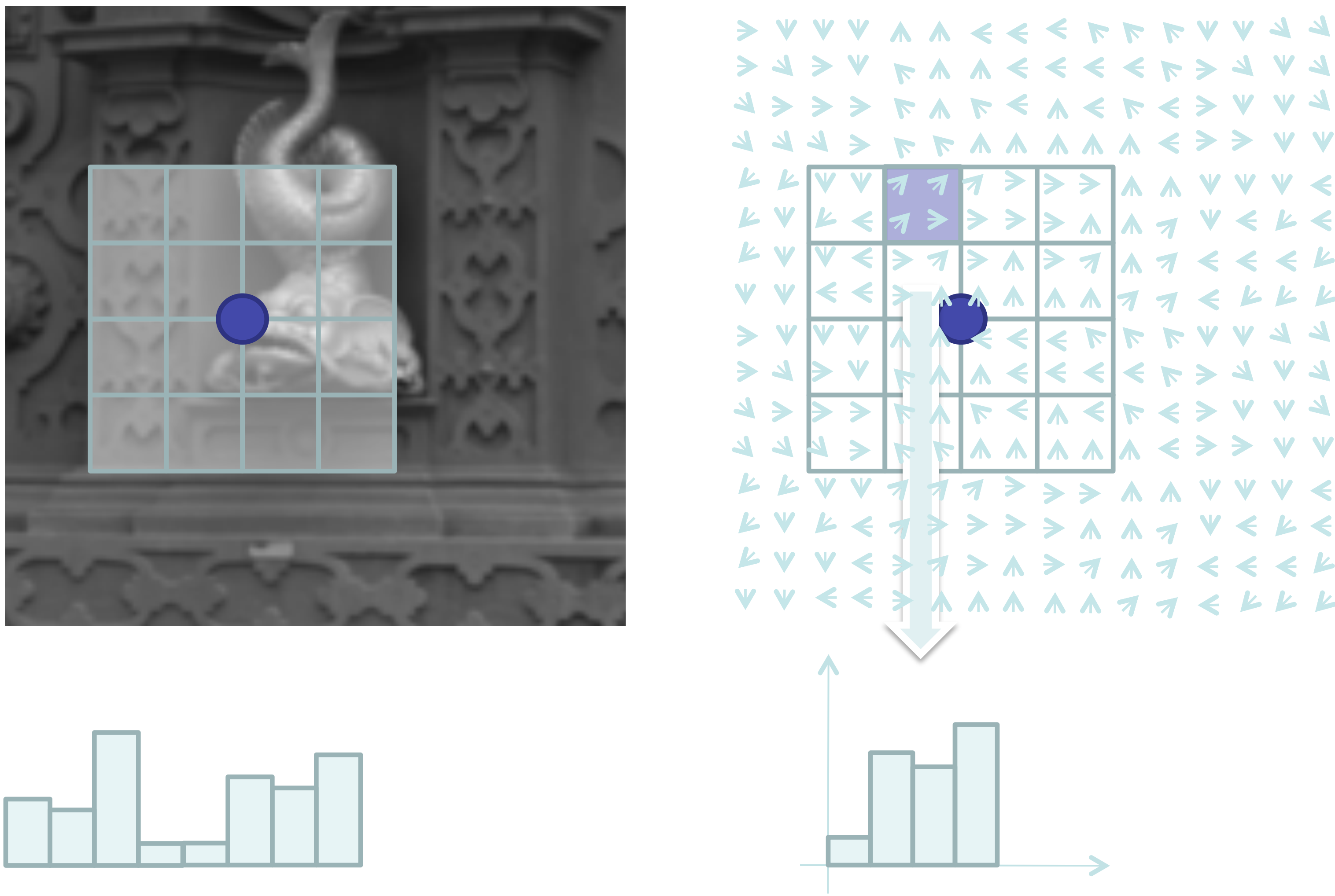
David G. Lowe: Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision 60(2): 91-110

SIFT computation



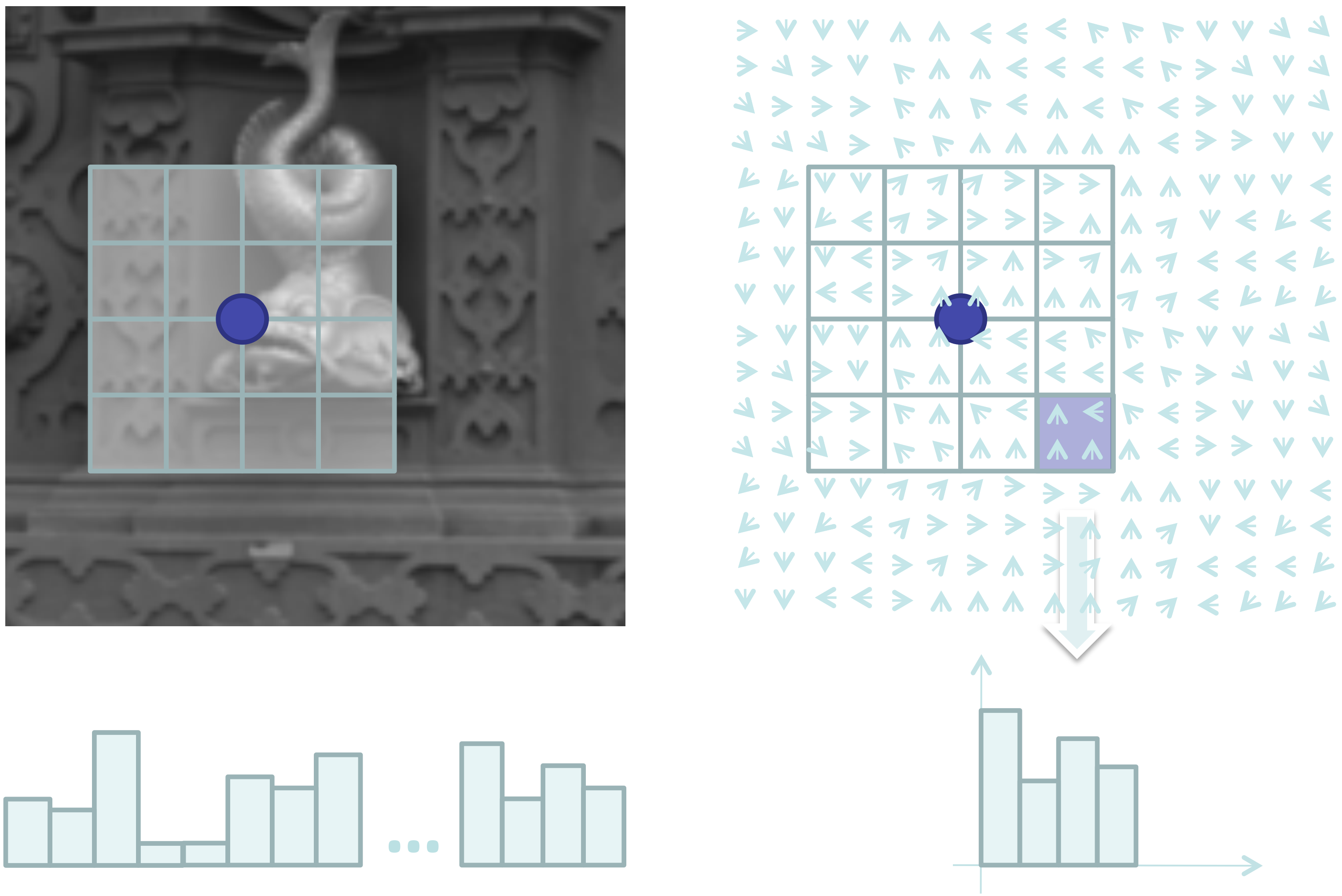
Slide credit: E. Tola

SIFT computation



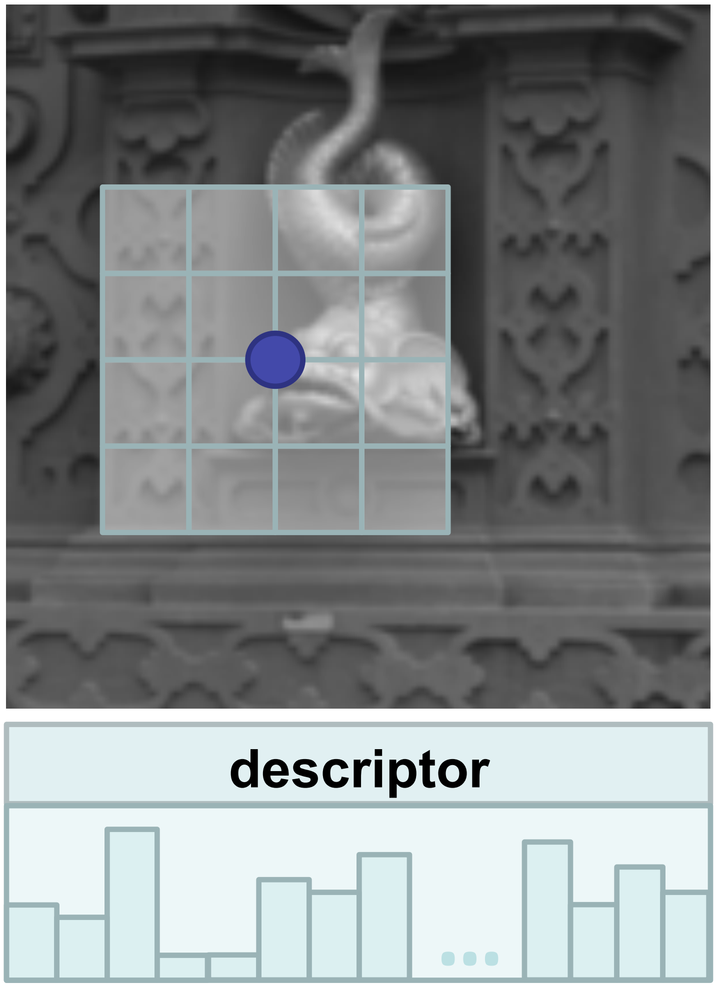
Slide credit: E. Tola

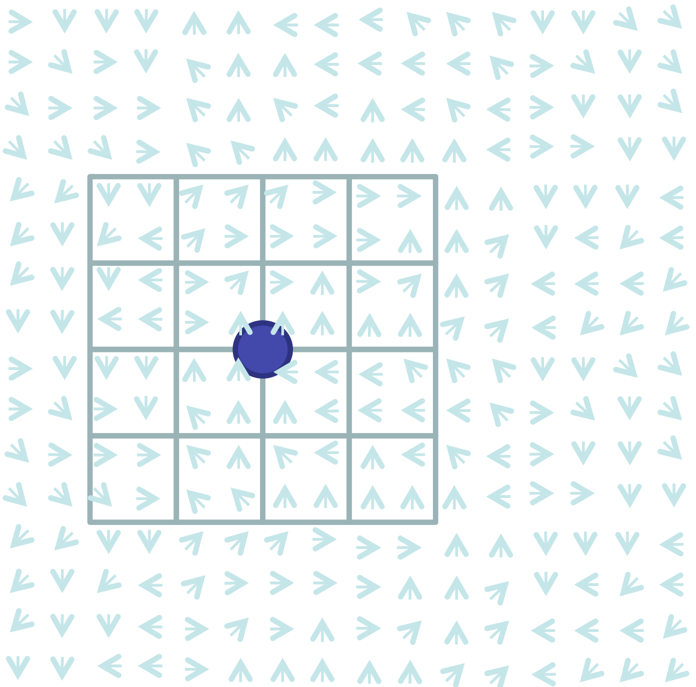
SIFT computation



Slide credit: E. Tola

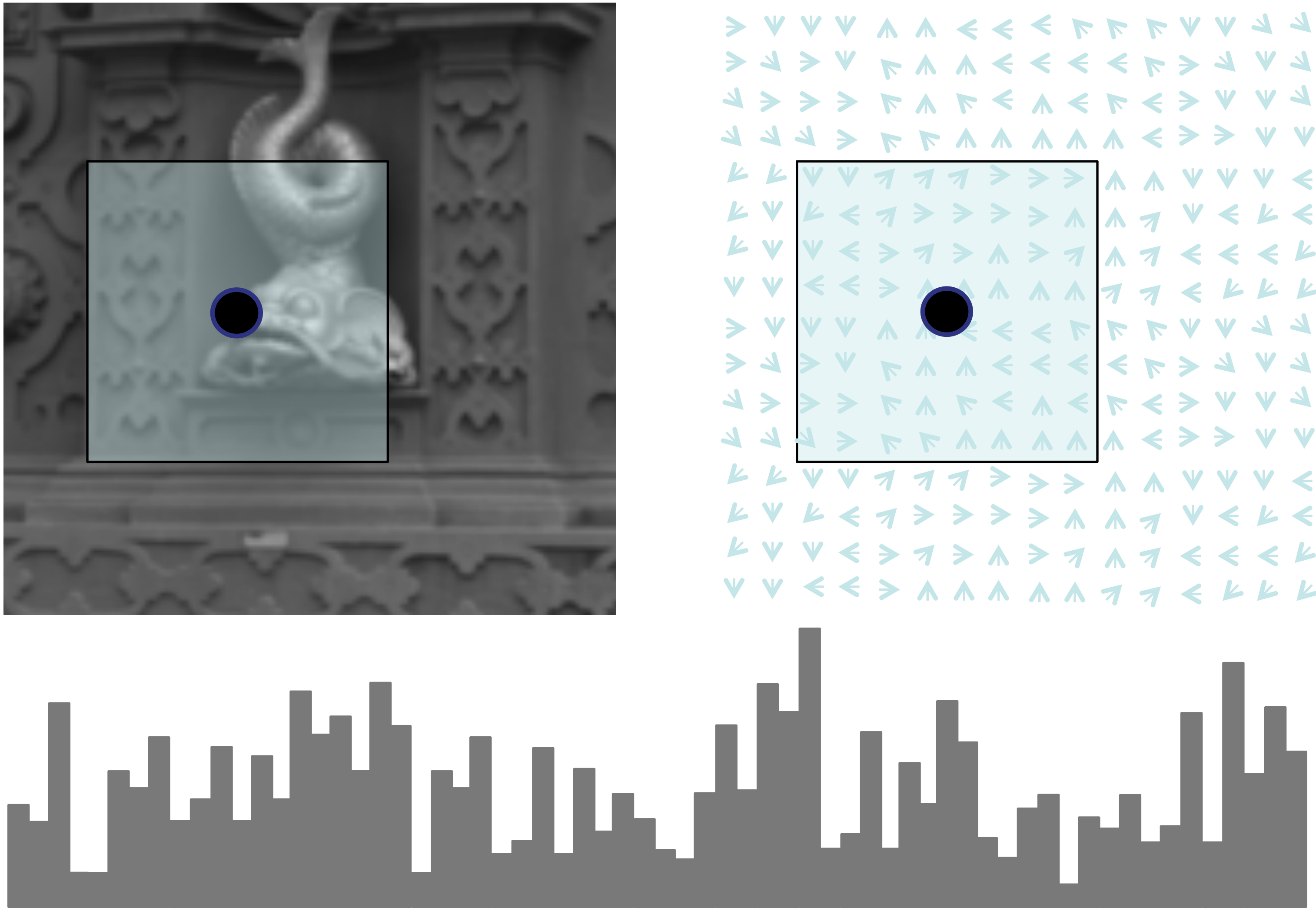
SIFT computation





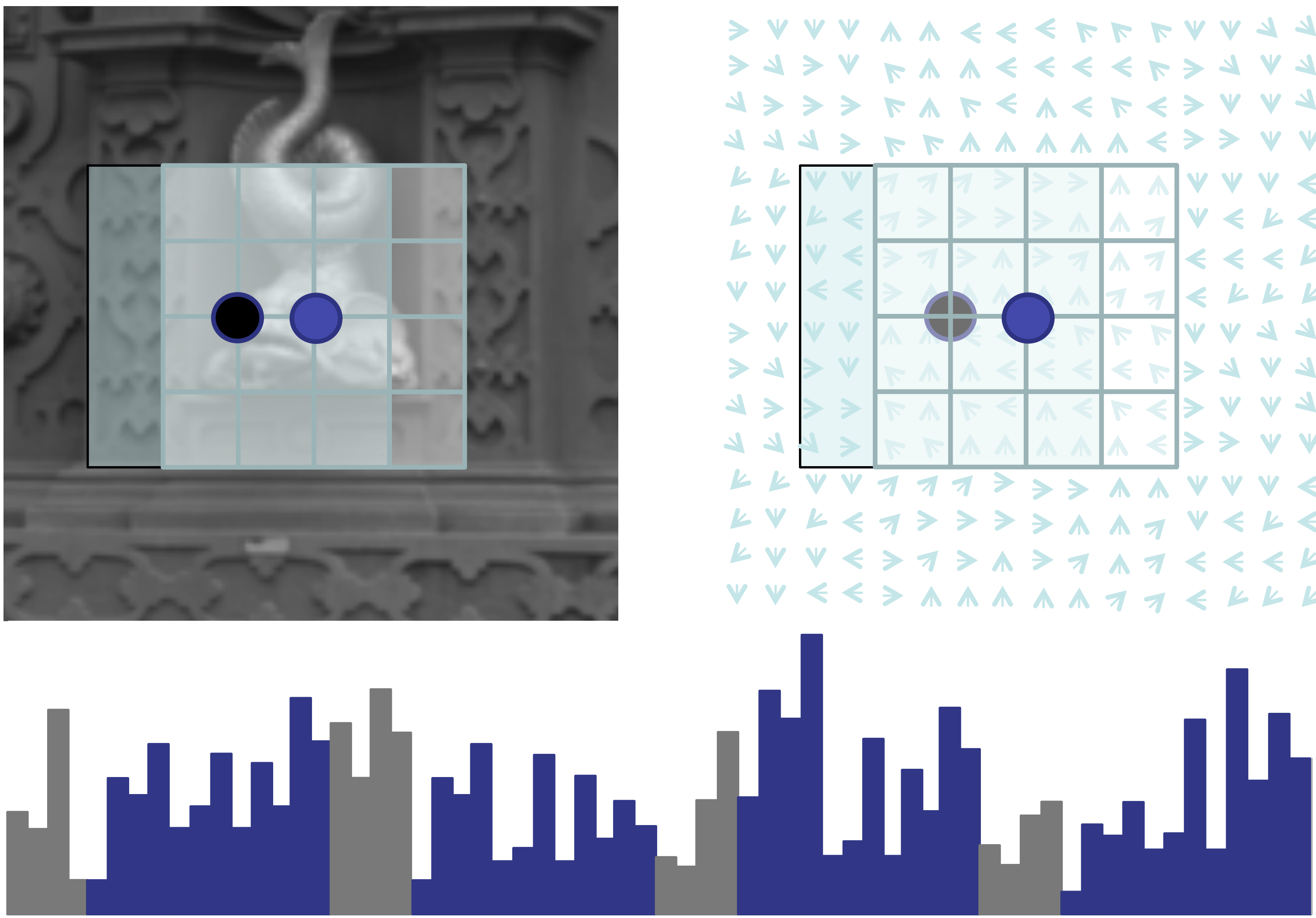
Slide credit: E. Tola

SIFT computation



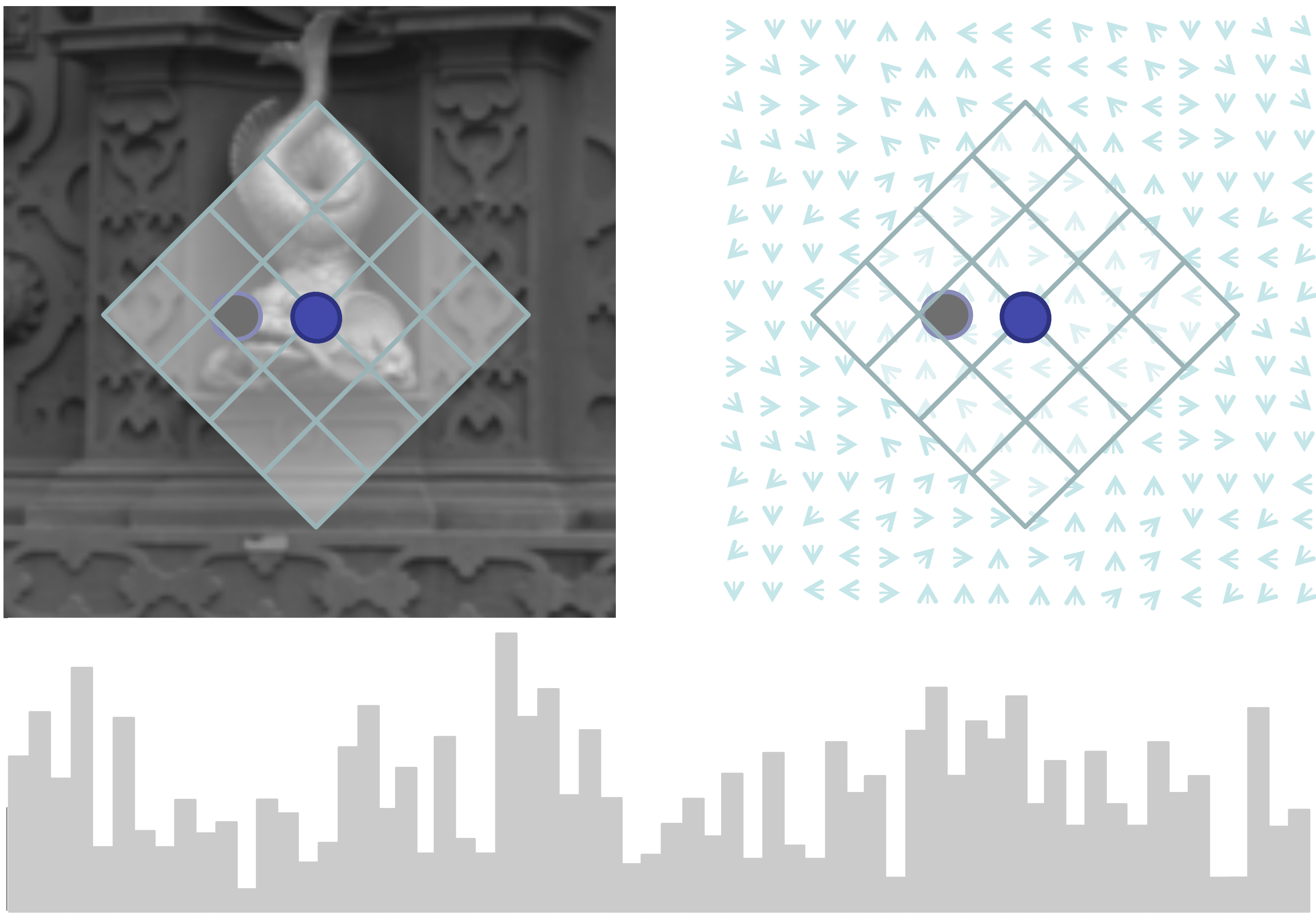
Slide credit: E. Tola

SIFT computation



Slide credit: E. Tola

SIFT computation

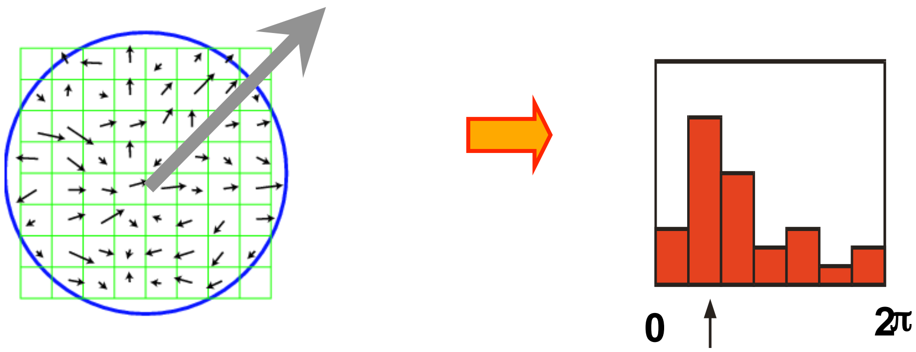


Slide credit: E. Tola

Scale-Invariant Feature Transform (SIFT) descriptor

Use location and characteristic scale given by blob detector

Estimate orientation from orientation histogram

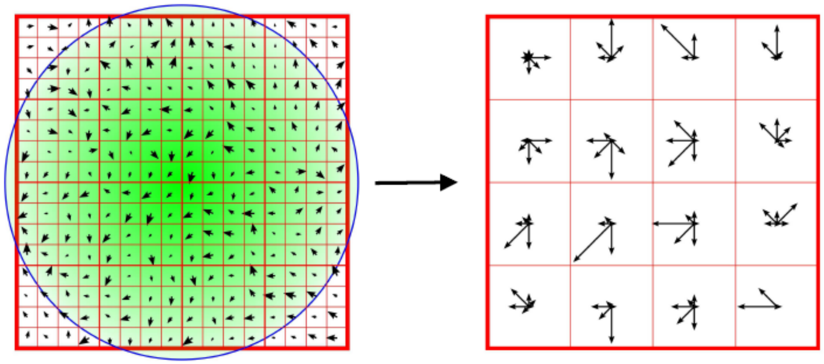


Break patch in 4x4 location blocks

8-bin orientation histogram per block

8x4x4 = 128-D descriptor

Normalize to unit norm

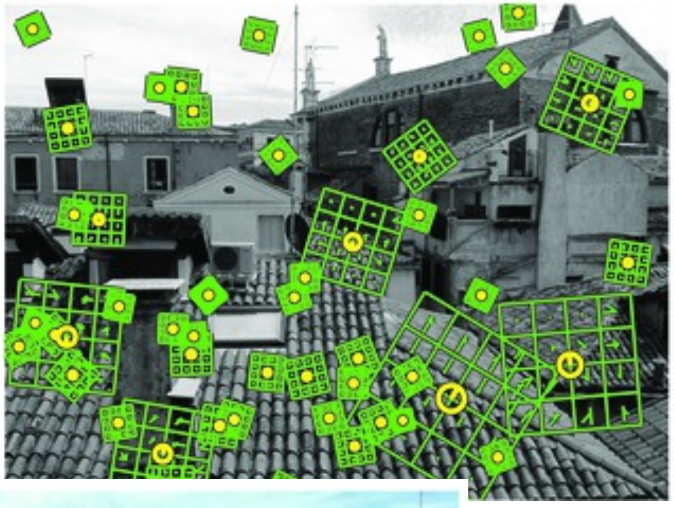


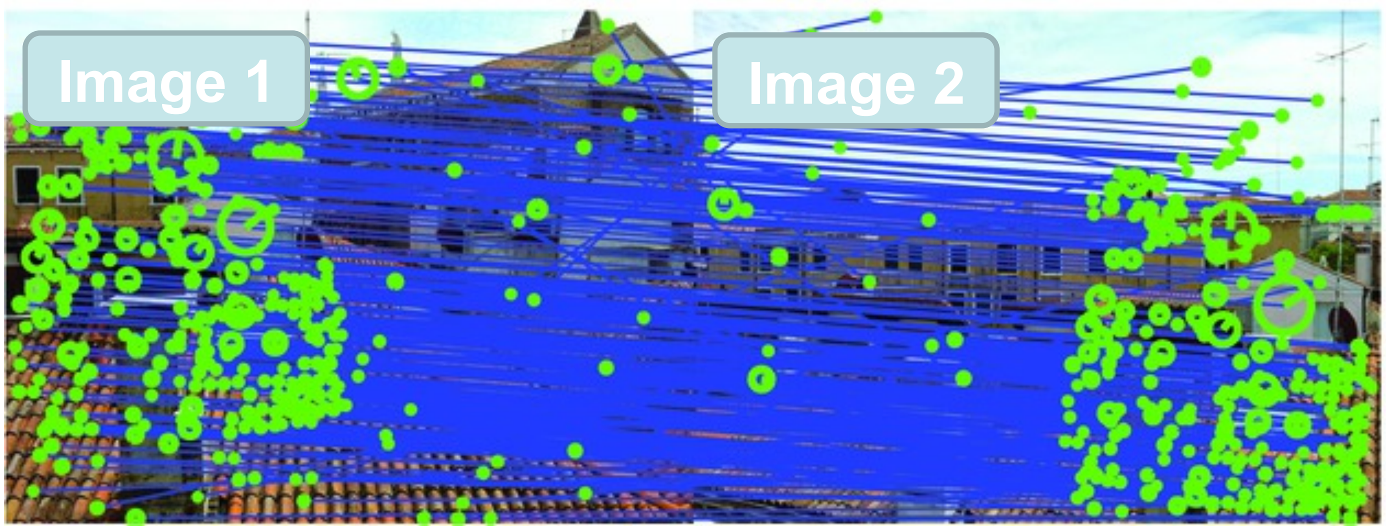
Invariance to: scale, orientation, multiplicative & additive changes

Application: Image Matching

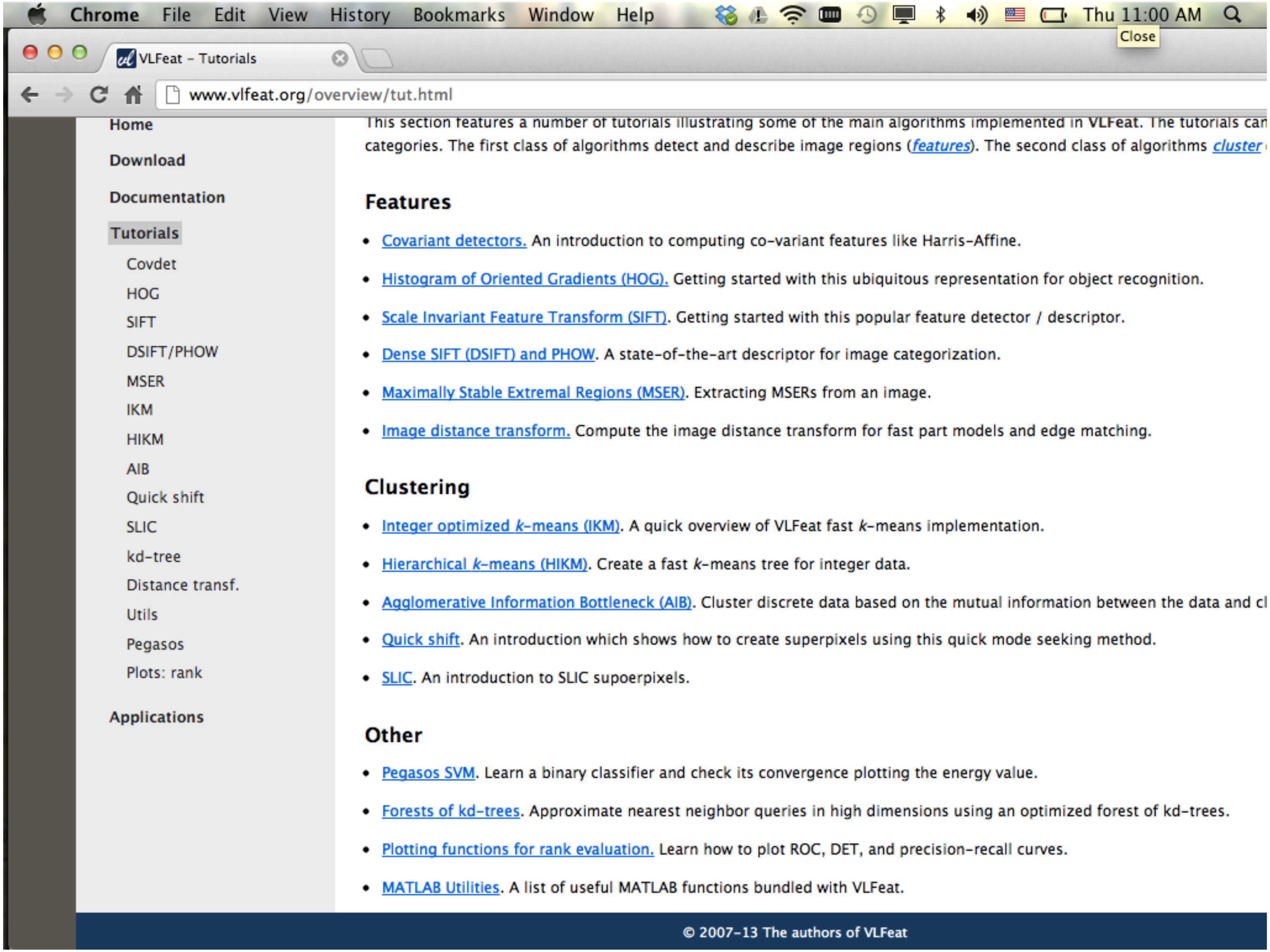
Assumption: images undergo global deformations with a few degrees-of-freedom (e.g. scaling, rotation)

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





Open source implementation: www.vlfeat.org



Chrome File Edit View History Bookmarks Window Help Thu 11:00 AM

Close

VLFeat – Tutorials

www.vlfeat.org/overview/tut.html

- Home
- Download
- Documentation
- Tutorials
  - Covdet
  - HOG
  - SIFT
  - DSIFT/PHOW
  - MSER
  - IKM
  - HIKM
  - AIB
  - Quick shift
  - SLIC
  - kd-tree
  - Distance transf.
  - Utils
  - Pegasos
  - Plots: rank
- Applications

This section features a number of tutorials illustrating some of the main algorithms implemented in VLFeat. The tutorials can categories. The first class of algorithms detect and describe image regions (*features*). The second class of algorithms *cluster*

Features

- Covariant detectors. An introduction to computing co-variant 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 cl
- Quick shift. An introduction which shows how to create superpixels using this quick mode seeking method.
- SLIC. An introduction to SLIC supoerpixels.

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.

Further reading (literature 'seeds')

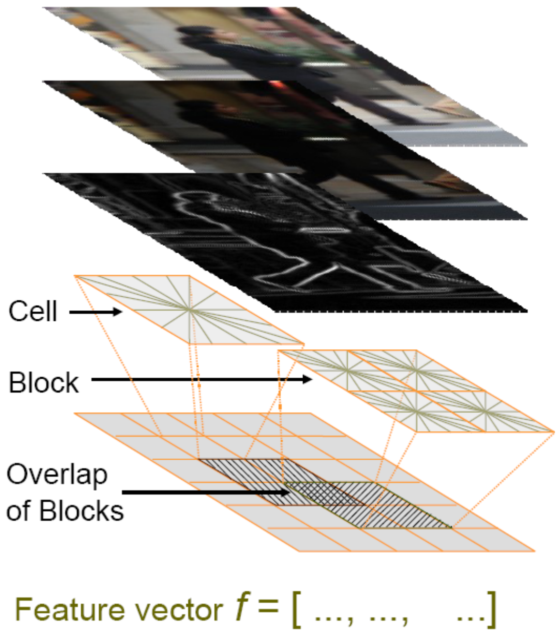
- **Compact Codes & Large-scale Retrieval**
  - J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. ICCV, 2003.
  - Nister, D., Stewenius, H.: Scalable recognition with a vocabulary tree. CVPR. (2006)
  - M. Perdoch, O. Chum, and J. Matas. Efficient representation 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')

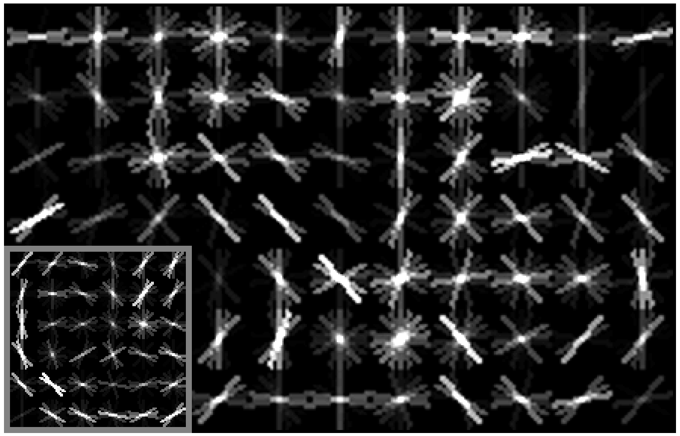
- **Feature encoding**
  - Improving the fisher kernel for large-scale image classification, F. Perronnin, J. Sánchez, and T. Mensink. In Proc. ECCV, 2010.
  - The devil is in the details: an evaluation of recent feature encoding methods, K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman, BMVC, 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

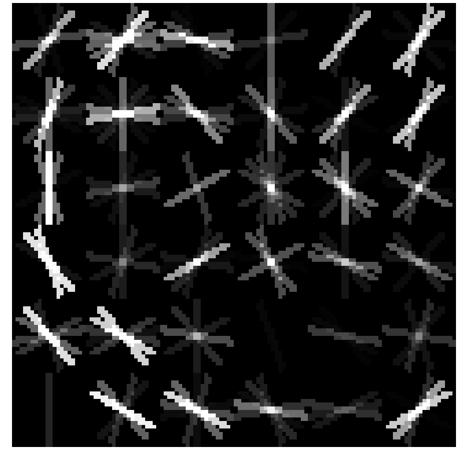
Histogram of Orientated Gradients (HOG) descriptor

- 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

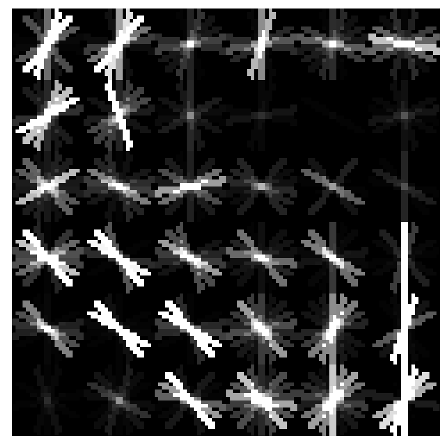


Part score computation





$\mathbf{w}[y]$

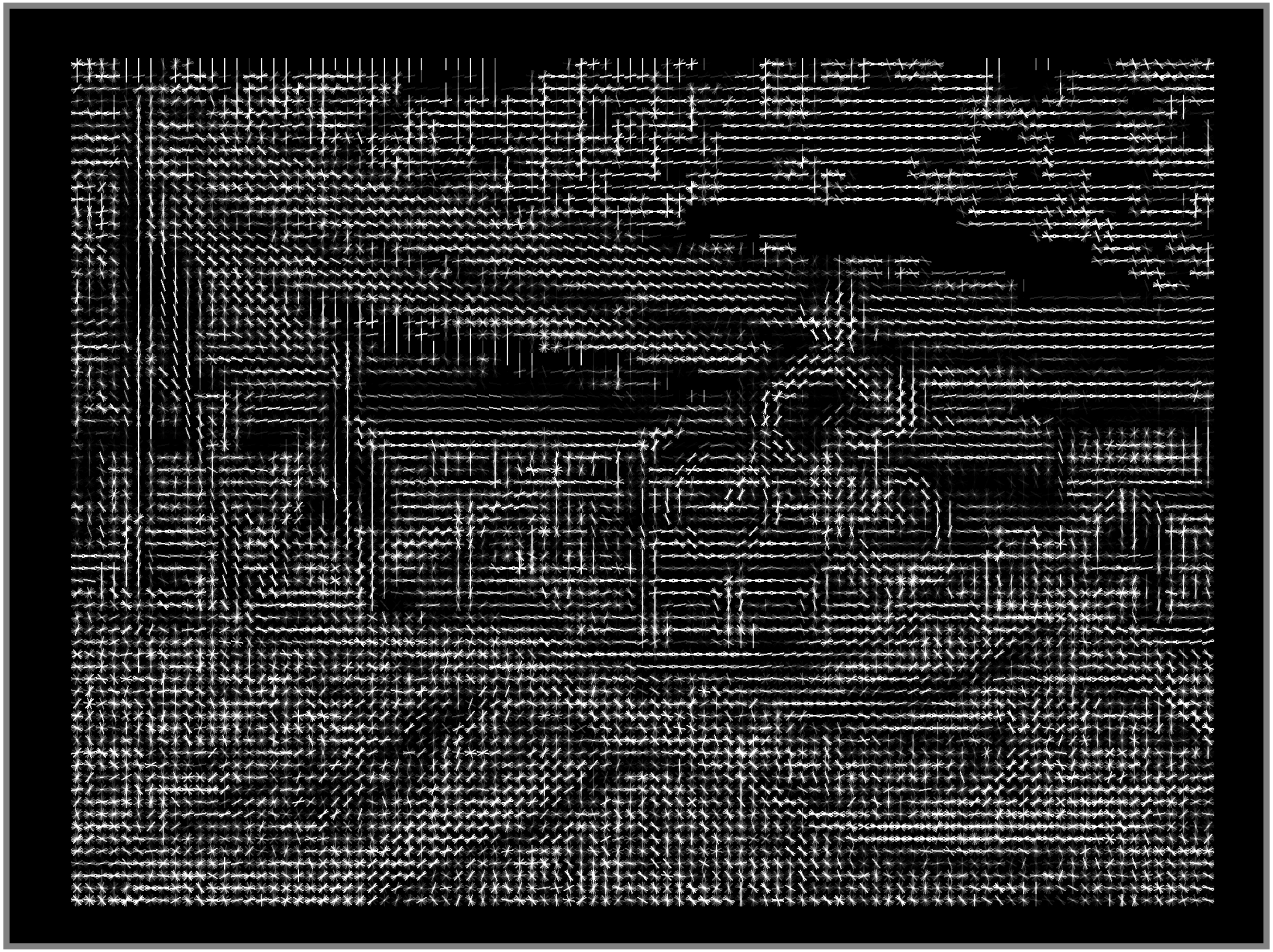


$\mathbf{h}[x+y]$

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

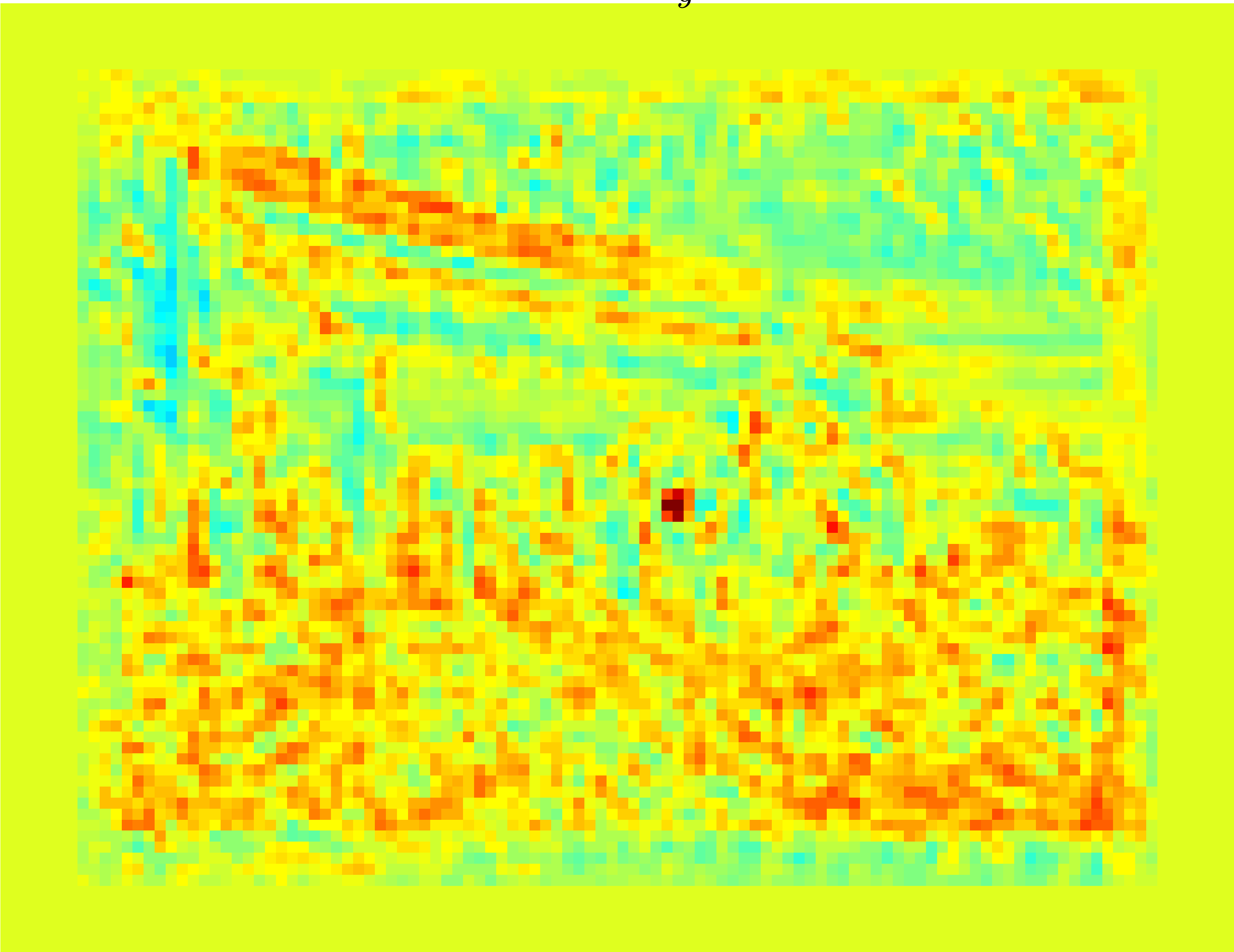
Part score

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

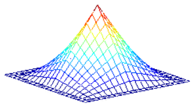


Part score

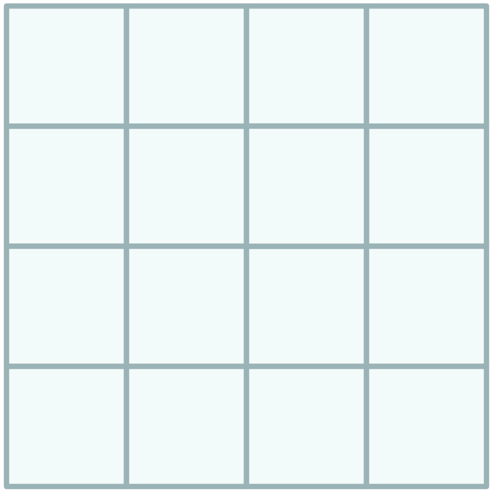
$$\mathbf{h}[x] \qquad s[x] = \sum_y \langle \mathbf{h}[x+y], \mathbf{w}[y] \rangle$$



SIFT-> DAISY



SIFT

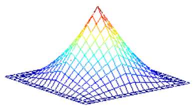


+ Good

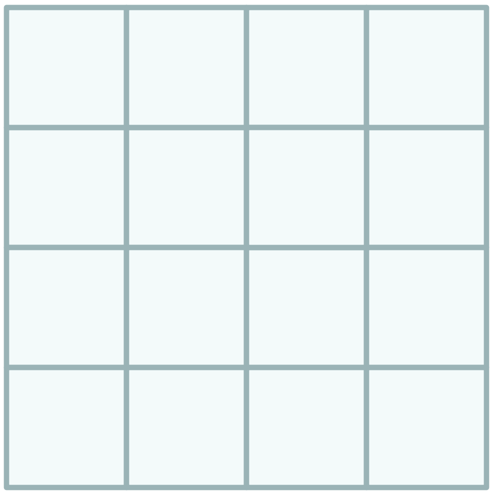
- Not suitable for dense computation

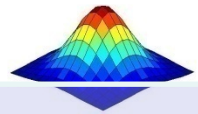
DAISY: An Efficient Dense Descriptor Applied to Wide Baseline Stereo, E. Tola, V. Lepetit, P. Fua, PAMI 10

SIFT-> DAISY

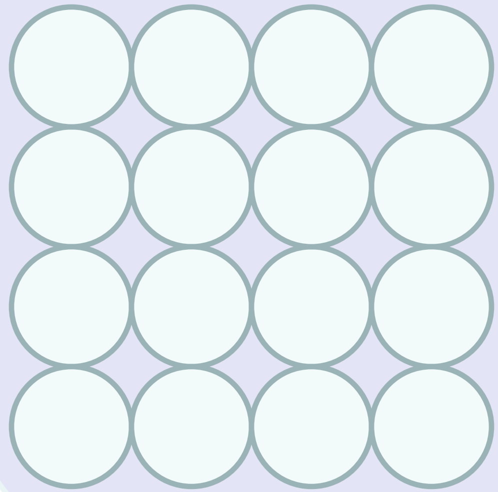


SIFT

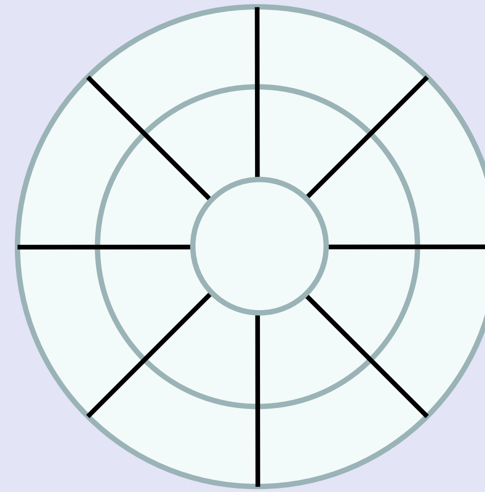




Sym.SIFT



GLOH*



+ Good

- Not suitable for dense computation

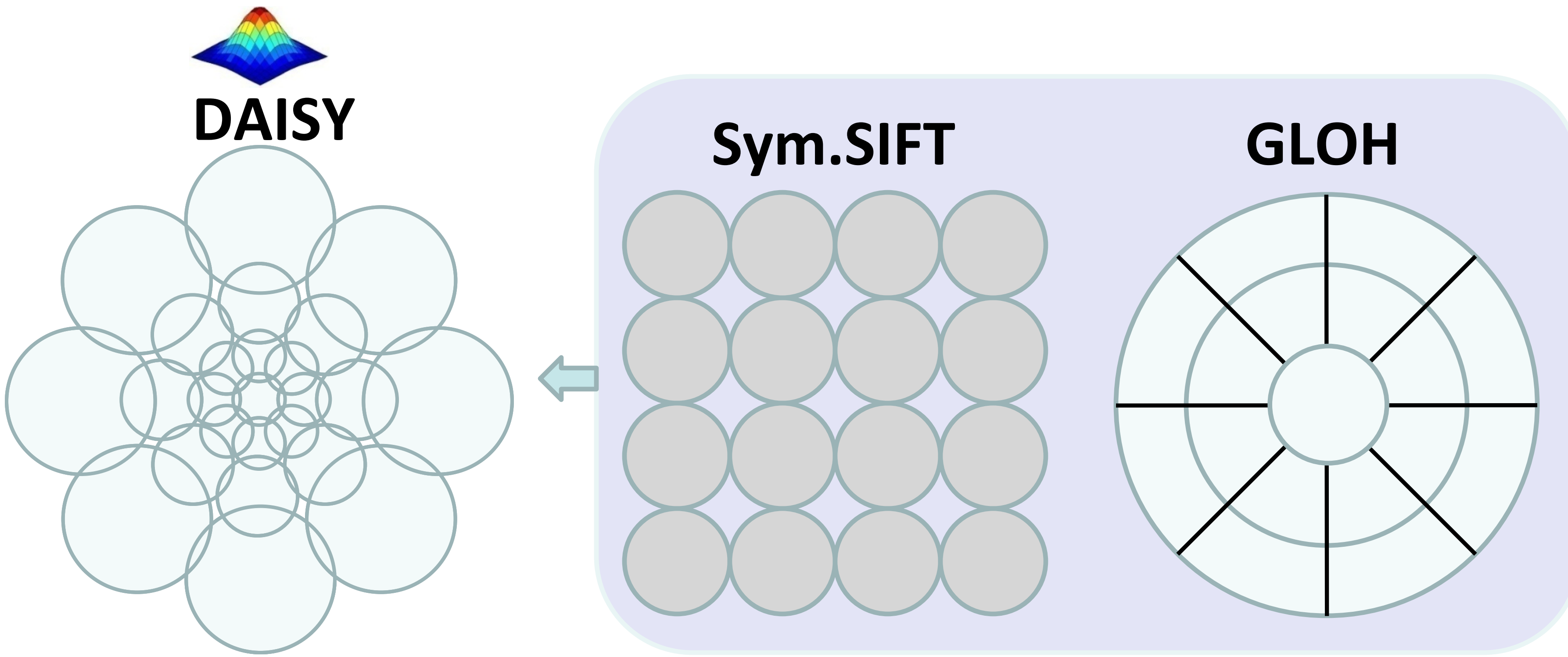
+ Gaussian Kernels : Suitable for Dense Computation

+ Good Performance

- Not suitable for dense computation

* K. Mikolajczyk and C. Schmid. A Performance Evaluation of Local Descriptors. PAMI’04.

SIFT-> DAISY



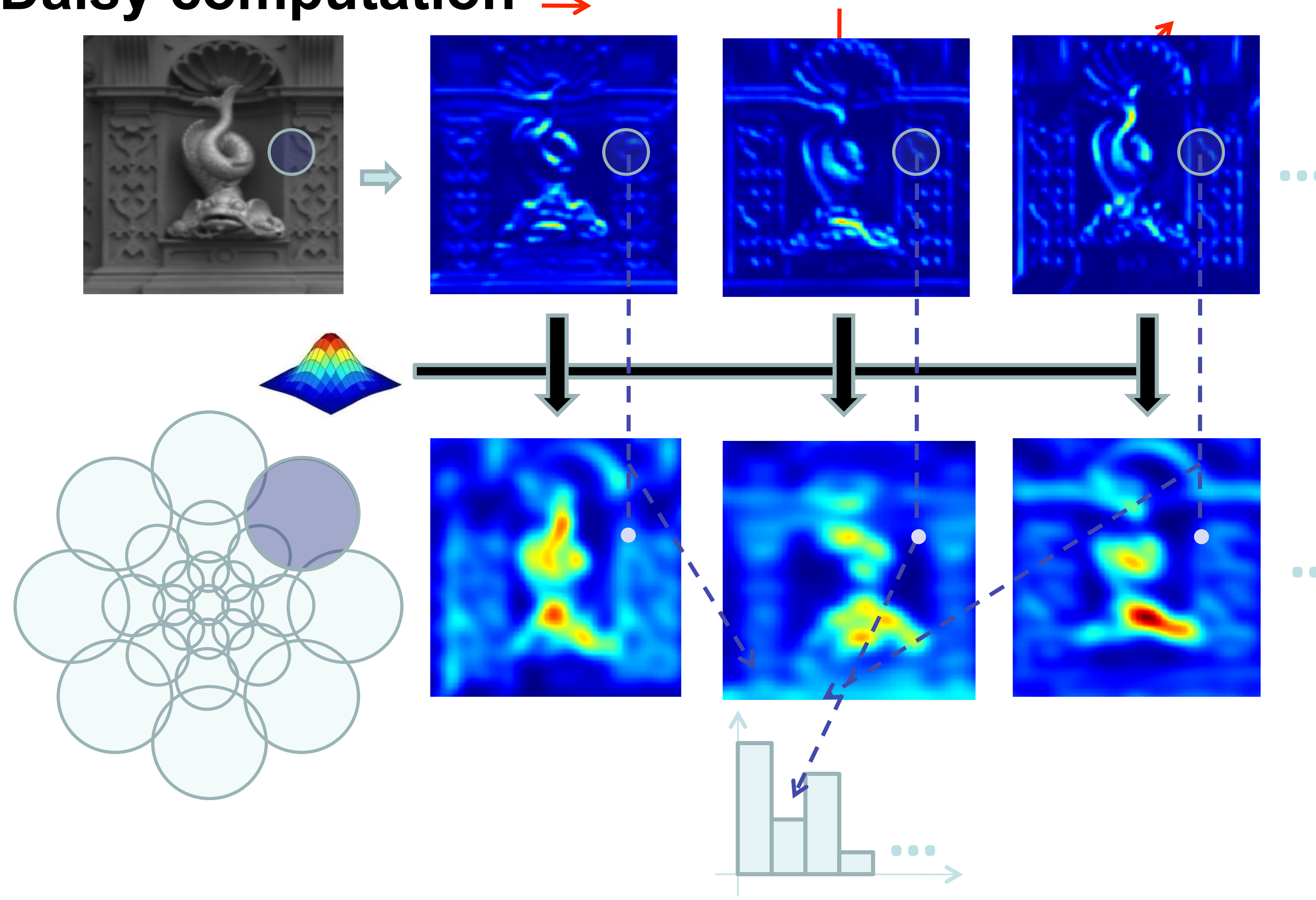
Suitable for dense computation
+ Improved performance:*
+ Precise localization
+ Rotational Robustness

+ Suitable for Dense Computation

+ Good Performance
- Not suitable for dense computation

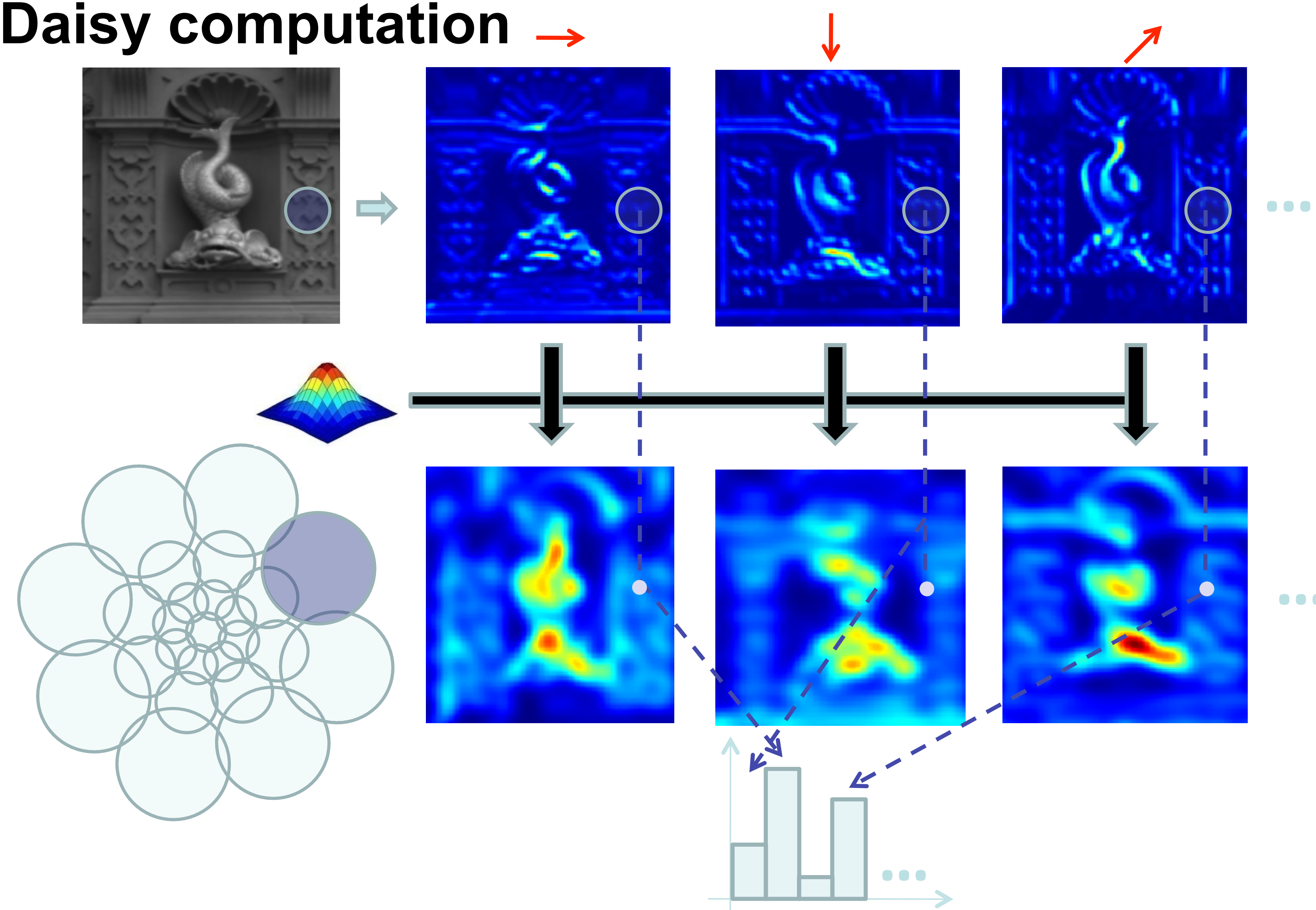
* S. Winder and M. Brown. *Learning Local Image Descriptors* in CVPR'07

Daisy computation



DAISY: An Efficient Dense Descriptor Applied to Wide Baseline Stereo, E. Tola, V. Lepetit, P. Fua, PAMI, 10

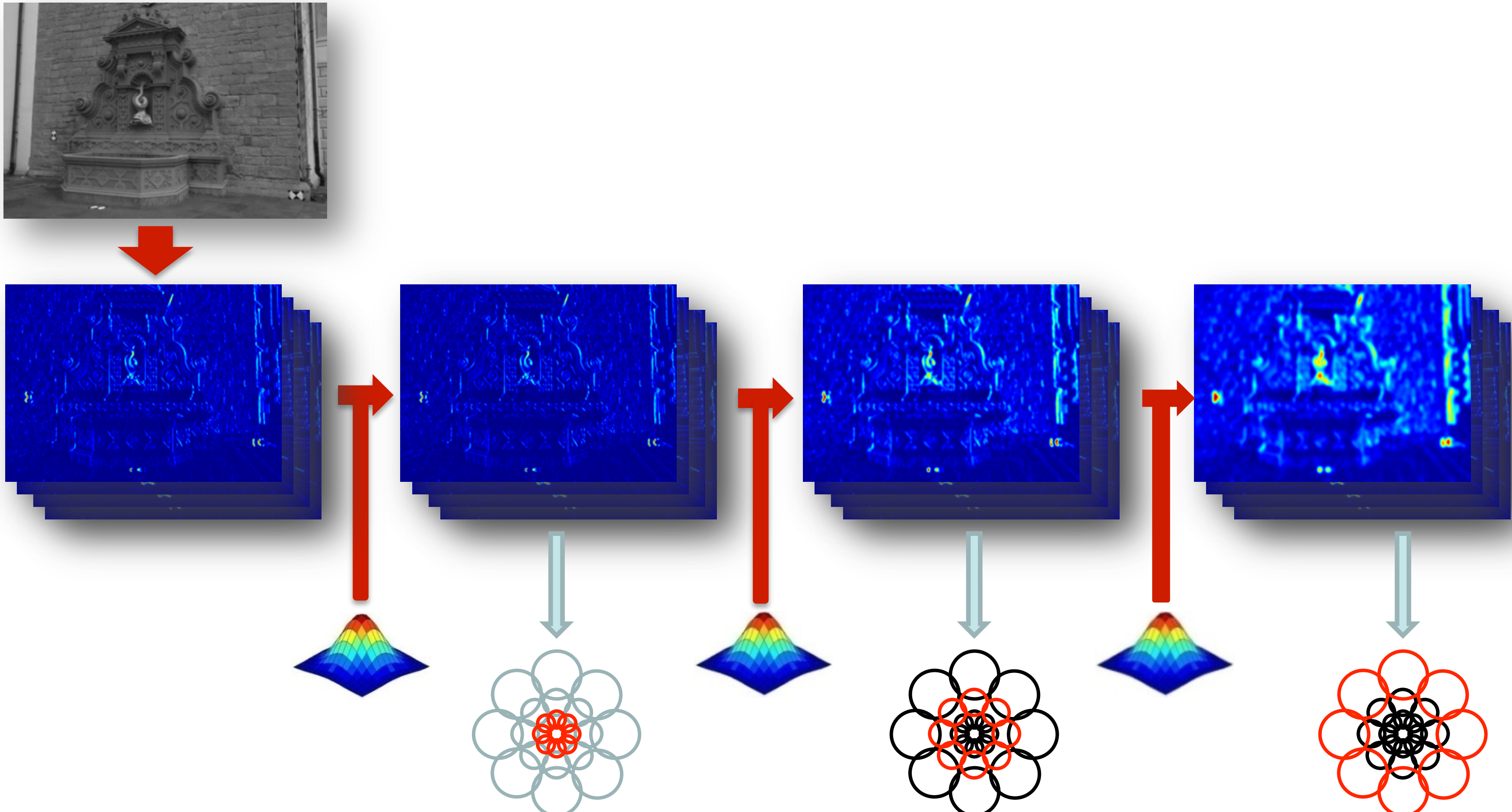
Daisy computation



DAISY: An Efficient Dense Descriptor Applied to Wide Baseline Stereo, E. Tola, V. Lepetit, P. Fua, PAMI, 10

Daisy computation

DAISY : 5s
SIFT : 250s



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

DAISY: An Efficient Dense Descriptor Applied to Wide Baseline Stereo, E. Tola, V. Lepetit, P. Fua, PAMI, 10

Scale- and rotation- invariance & Fourier

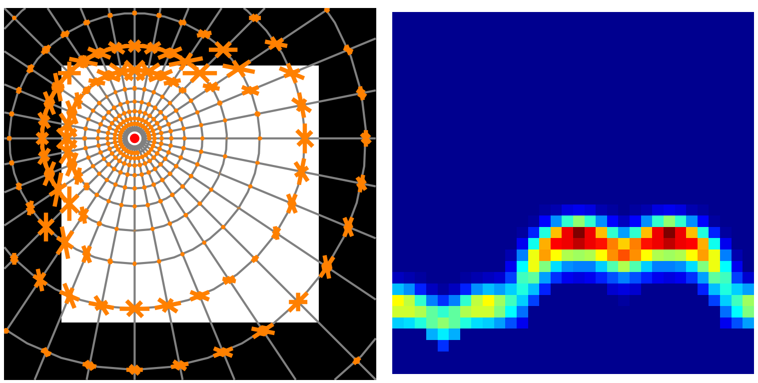
Fact 1: *Signal translation does not affect the signal's Fourier Transform Magnitude*:

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

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

Original

Image + grid@point descriptor

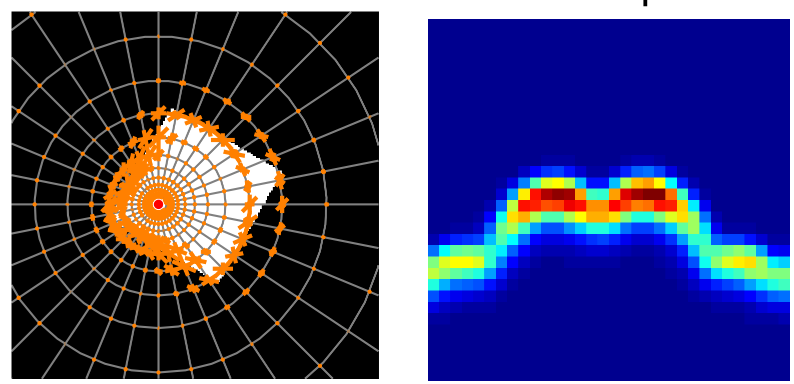


scale

angle

Scaled

Image + grid@point descriptor



scale

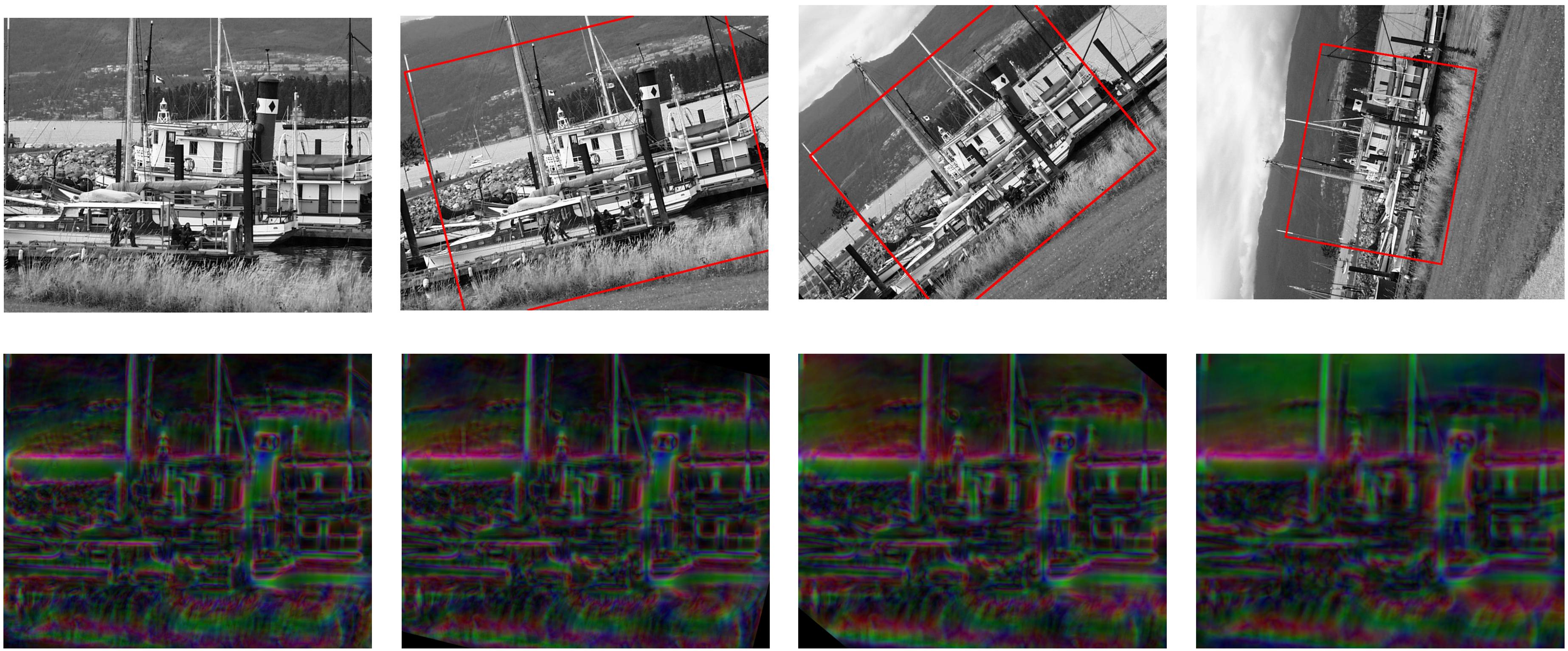
angle

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

Problem: How can a computer find cars (or faces, hands..) in images?

1980's pixels → edge → texton → motif → part → object

2000-2010

→ SIFT/HOG → K-Means/ pooling → classifier → "car"

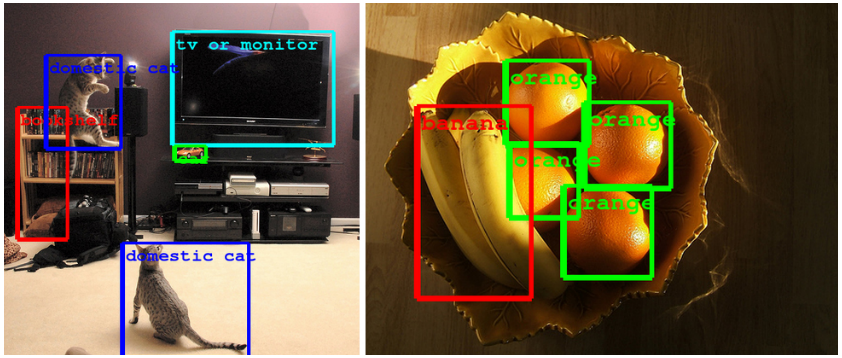
fixed | unsupervised | supervised

2010+

$\xrightarrow{x}$ $\boxed{max(0, W^1 x)}$ $\xrightarrow{h^1}$ $\boxed{max(0, W^2 h^1)}$ $\xrightarrow{h^2}$ $\boxed{W^3 h^2}$ $\xrightarrow{o}$

Deep Learning: Breakthroughs in speech & vision: classification, detection, recognition,...

Results from GoogLeNet, 2014
ECP entry: 7^{th} out of 38



Lecture summary

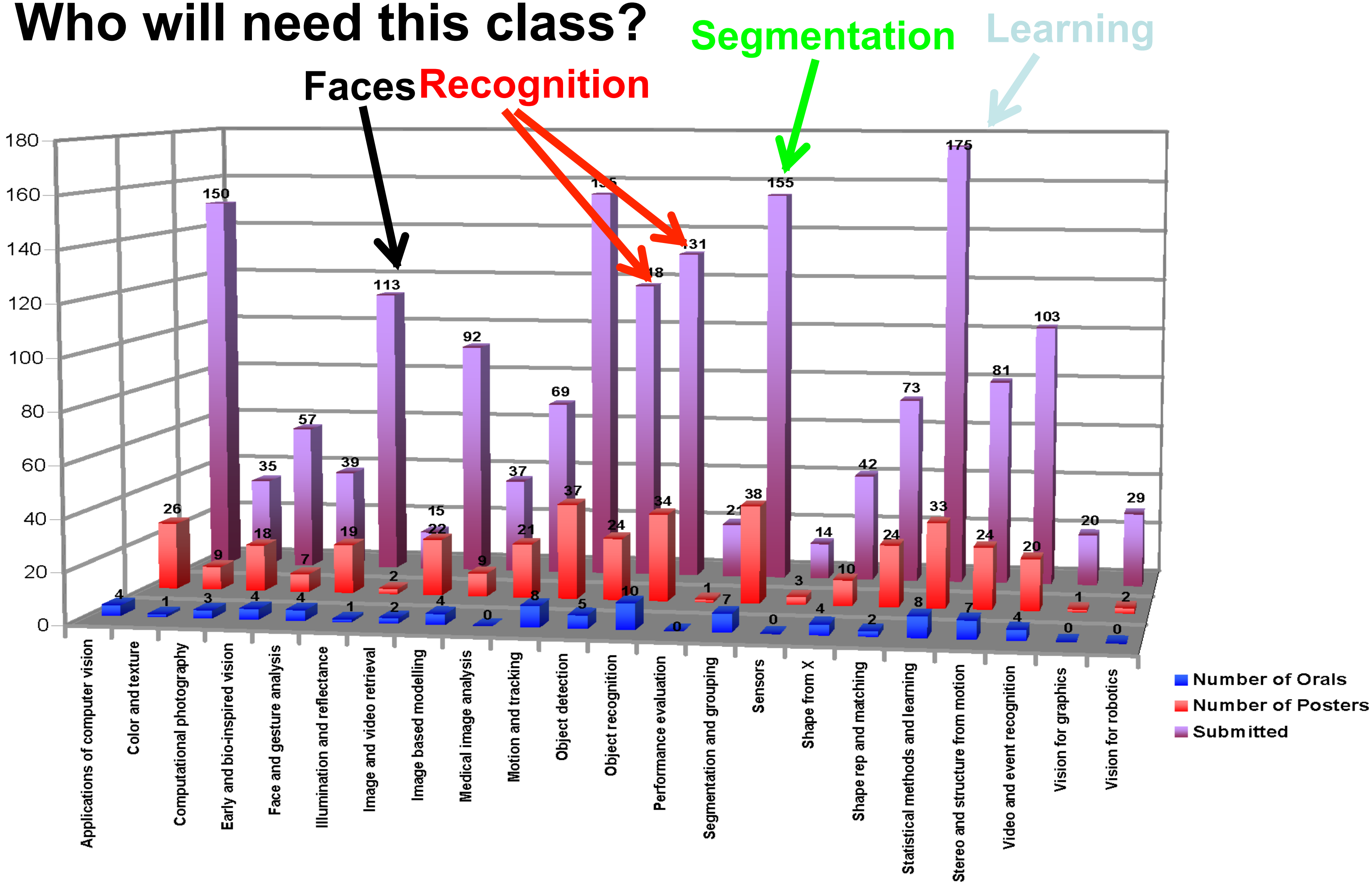
Introduction to the class

Introduction to the problem of classification

Linear classifiers

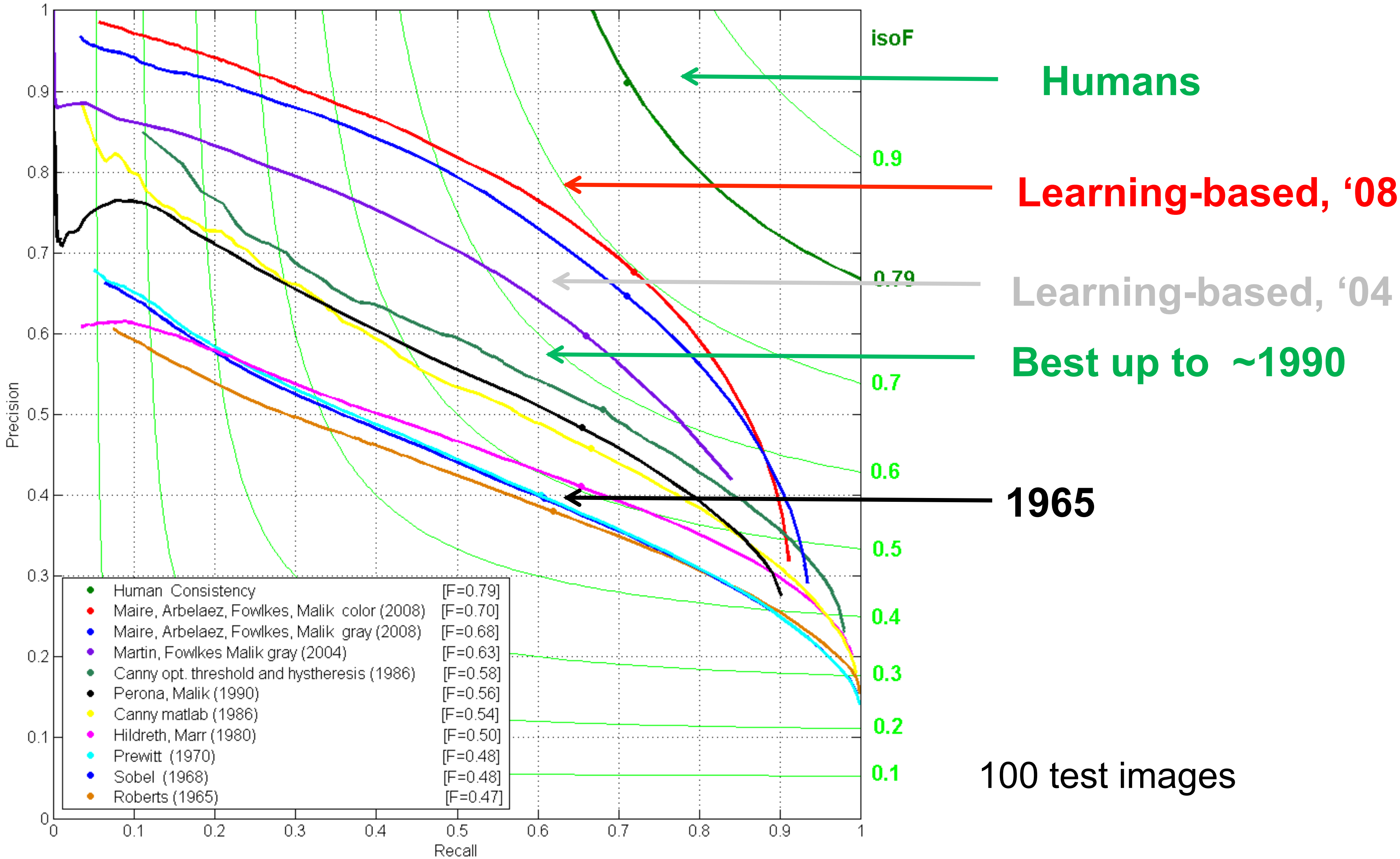
Image-based features

Who will need this class?



Submission/Acceptance Statistics from CVPR 2010

Progress during the last 4 decades



100 test images

Lecture summary

Introduction to the class

Introduction to the problem of classification

Linear classifiers

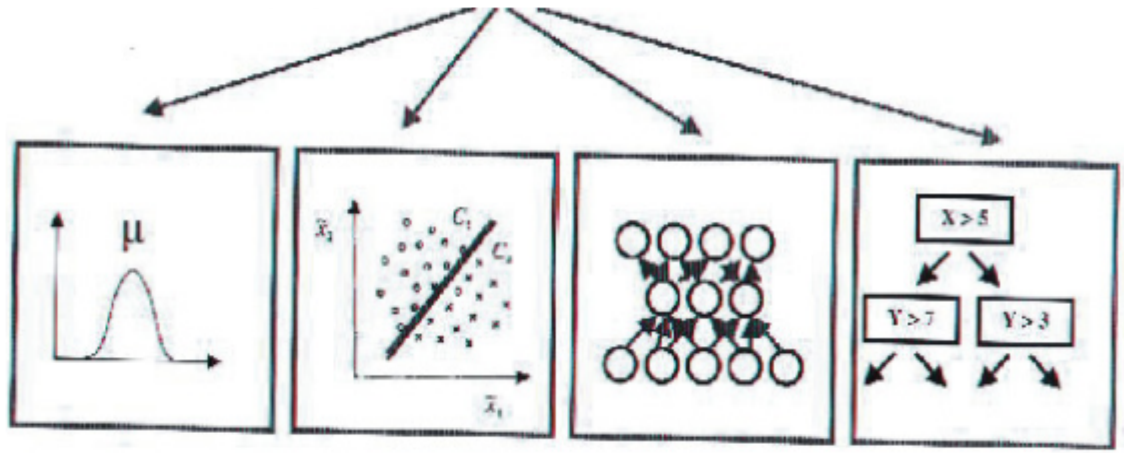
Image-based features

Classifier function

- Input-output mapping

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

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

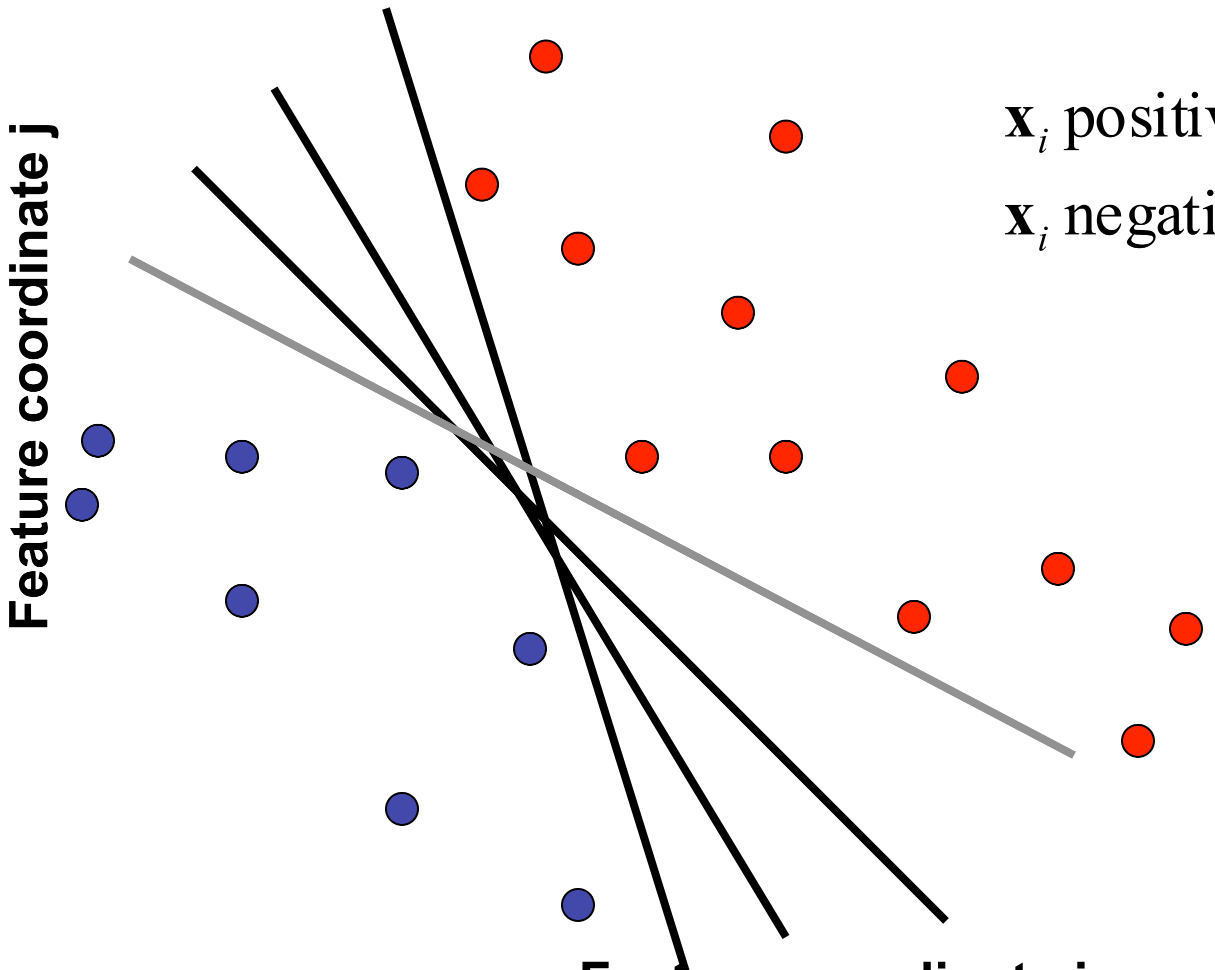
Introduction to the problem of classification

Linear classifiers

Image-based features

Linear Classifiers

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



$$\mathbf{x}_i \text{ positive}: \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 0$$

$$\mathbf{x}_i \text{ negative}: \quad \mathbf{x}_i \cdot \mathbf{w} + b < 0$$

Each data point has a class label:

$$y_t = \begin{cases} +1 \; (\;) \\ -1 \; (\;) \end{cases}$$

Linear regression

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

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

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

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

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

data term complexity term

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

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

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

Full-rank matrix

So how do we set λ ?

What is a good tradeoff between accuracy and complexity?

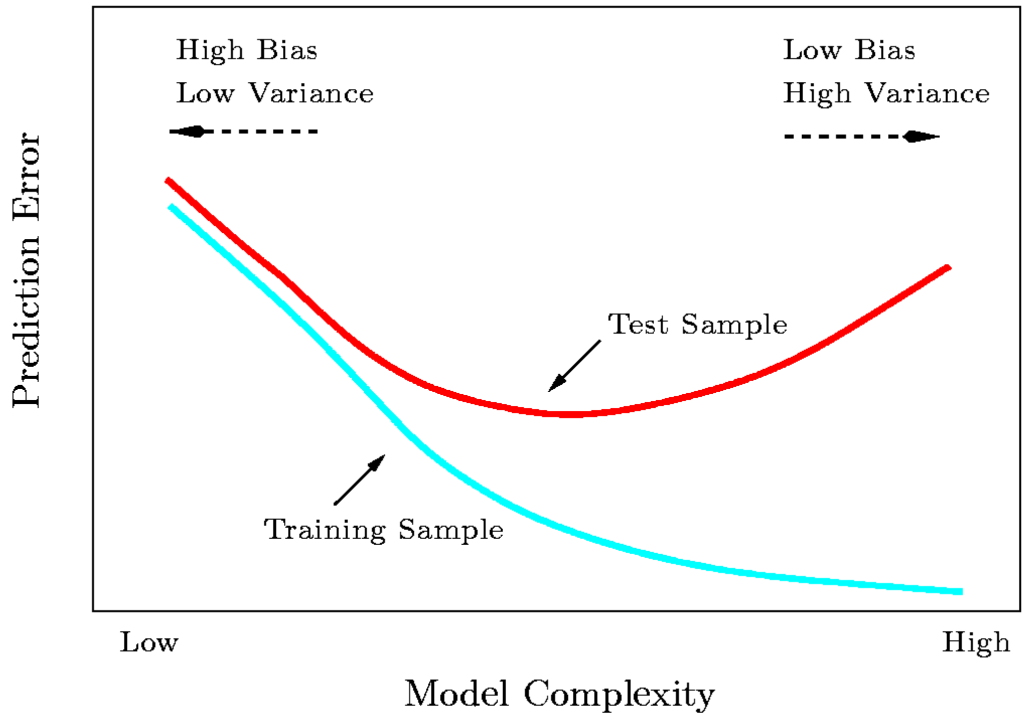
Tuning the model's complexity

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

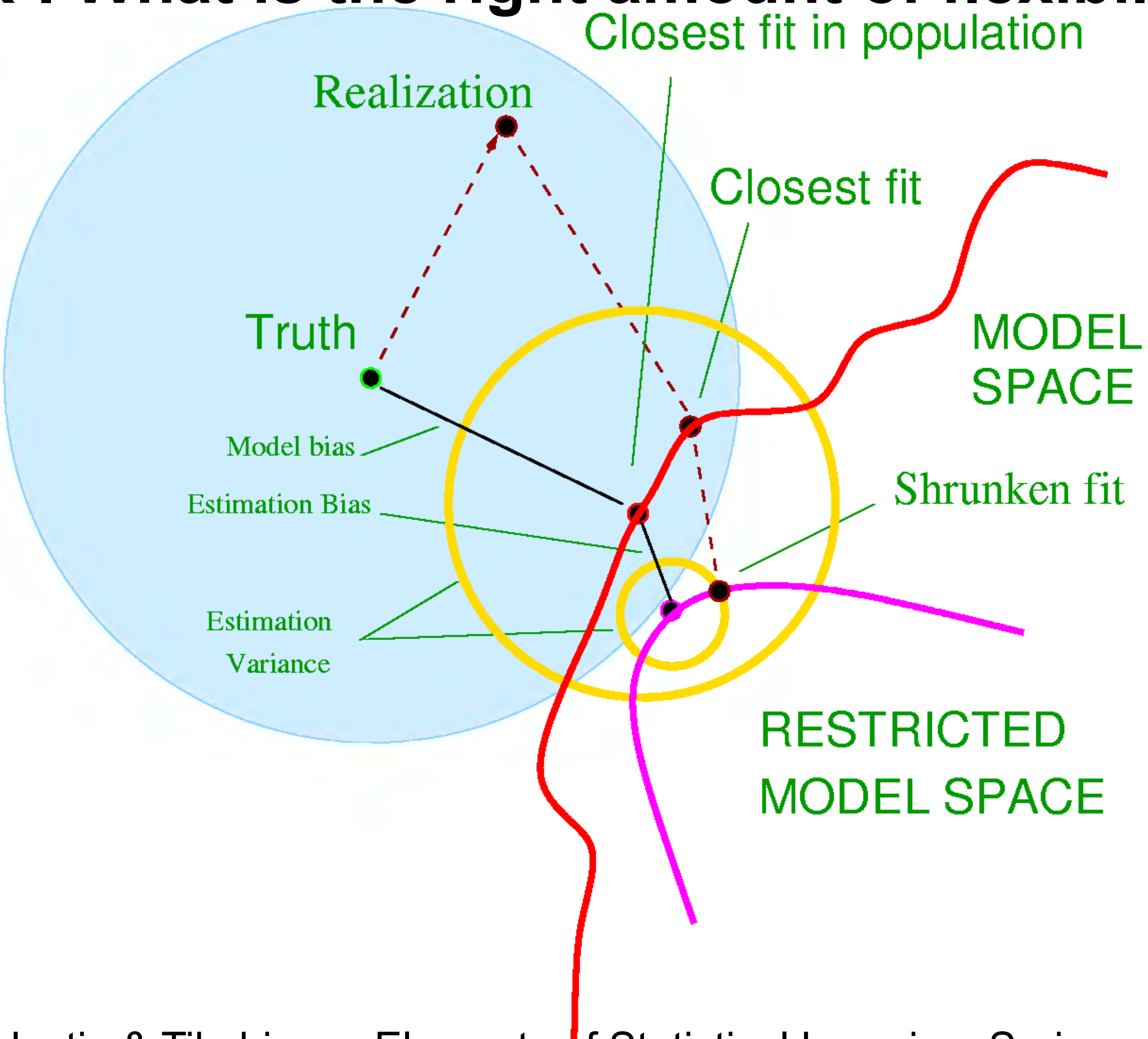
degree of invariance

task-specific performance

analytical tractability

...

Appendix-I What is the right amount of flexibility?



Slide credit: Hastie & Tibshirany, Elements of Statistical Learning, Springer 2001

Bias-Variance-I

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

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

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 :

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

Appendix-II: Ridge regression = parameter shrinkage

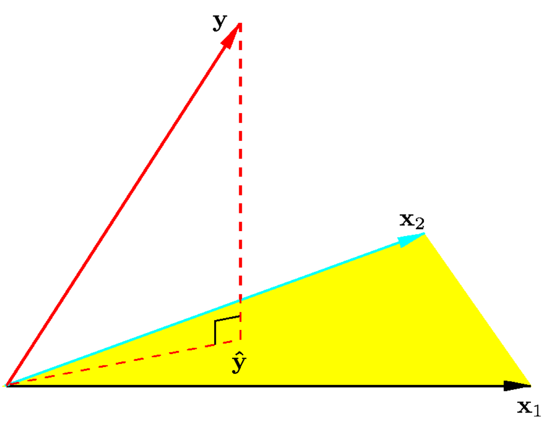
Reference: Hastie & Tibshirani, Elements of Statistical Learning, Springer 2001

Least squares parameter estimation: minimization of

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

$$\begin{aligned} \frac{\partial RSS(w)}{\partial w} &= 2\mathbf{X}^T(\mathbf{y} - \mathbf{X}w) \\ \mathbf{X}^T\mathbf{y} &= \mathbf{X}^T\mathbf{X}\hat{w} \\ \hat{w} &= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \end{aligned}$$

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



SVD-based interpretation of least squares

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

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

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

Reconstruction of y on the subspace spanned by X's columns

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

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

$\mathbf{U}$ orthonormal basis for $d-$dimensional subspace of R^M.

$\mathbf{U^Ty}$ expansion coefficients for projection of $\mathbf{Y}$ onto this basis.

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

- SVD interpretation

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

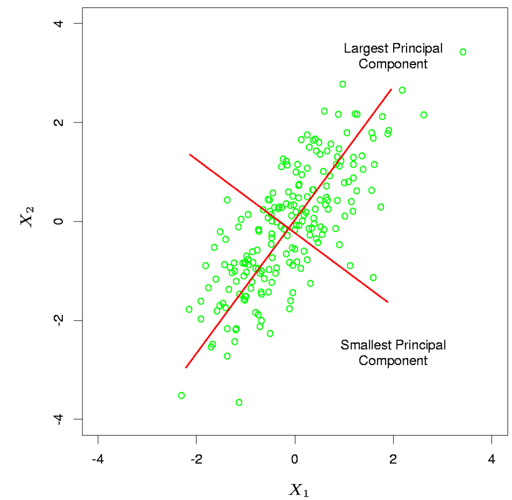
$$= \mathbf{U}\mathbf{D}\left(\mathbf{D}^2 + \lambda\mathbf{I}\right)^{-1}\mathbf{D}\mathbf{U}^{\mathbf{T}}\mathbf{y}$$

- `Shrinkage'

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

Feature Space Interpretation of ridge regression

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

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

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

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