CS294-43: Visual Object and Activity Recognition

Prof. Trevor Darrell

Feb 24th: Voting and Hashing techniques

Today

- Demos
- Niebles Presentation from last week [Daphna B.]

- ISM (Leibe) – *Mario Fritz guest lecture*
- Learning Distances (Frome) – [Ashley E.]
- Hashing with Metric Learning – *Brian Kulis guest lecture*
- Semantic Texton Forests (Shotten)
- Forests/Ferns for Keypoint Localization (Leptit)
- Small Codes/Large Image Datasets (Torralba)

Today

- Demos
- Niebles Presentation from last week [Daphna B.]

- **ISM (Leibe) – *Mario Fritz guest lecture***
- Learning Distances (Frome) – [Ashley E.]
- Hashing with Metric Learning – *Brian Kulis guest lecture*
- Semantic Texton Forests (Shotten)
- Forests/Ferns for Keypoint Localization (Leptit)
- Small Codes/Large Image Datasets (Torralba)

Implicit Shape Model

[Leibe,Schiele04]

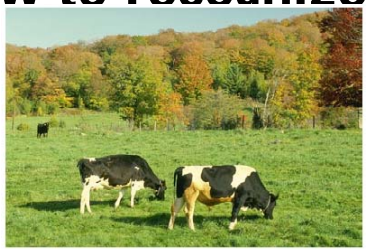
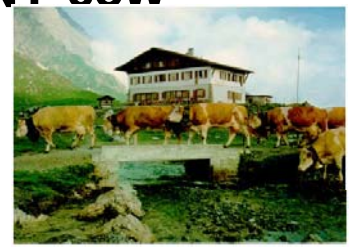
Object Categorization in Real-World Scenes

- How to recognize ANY car

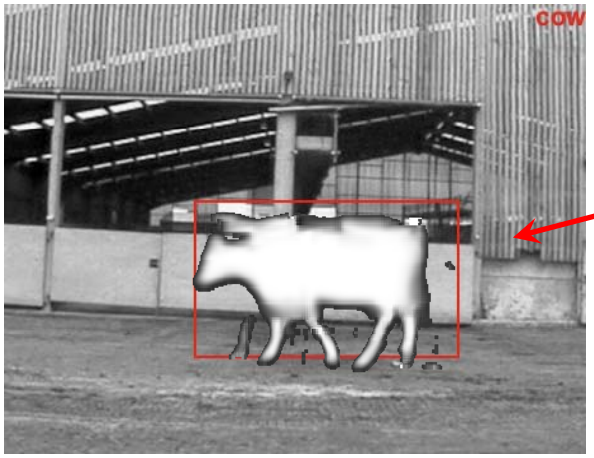


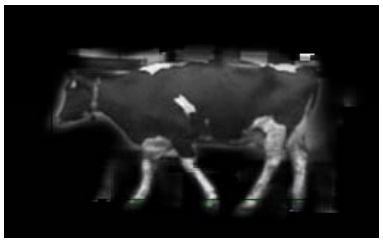


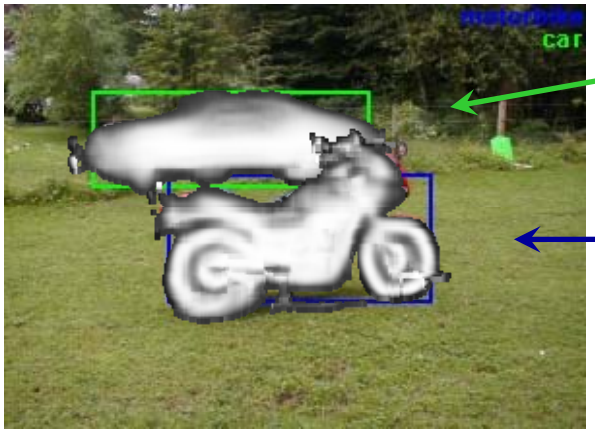
- How to recognize ANY cow

Object Categorization and Segmentation

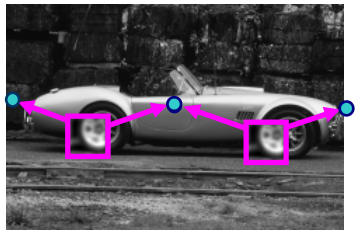






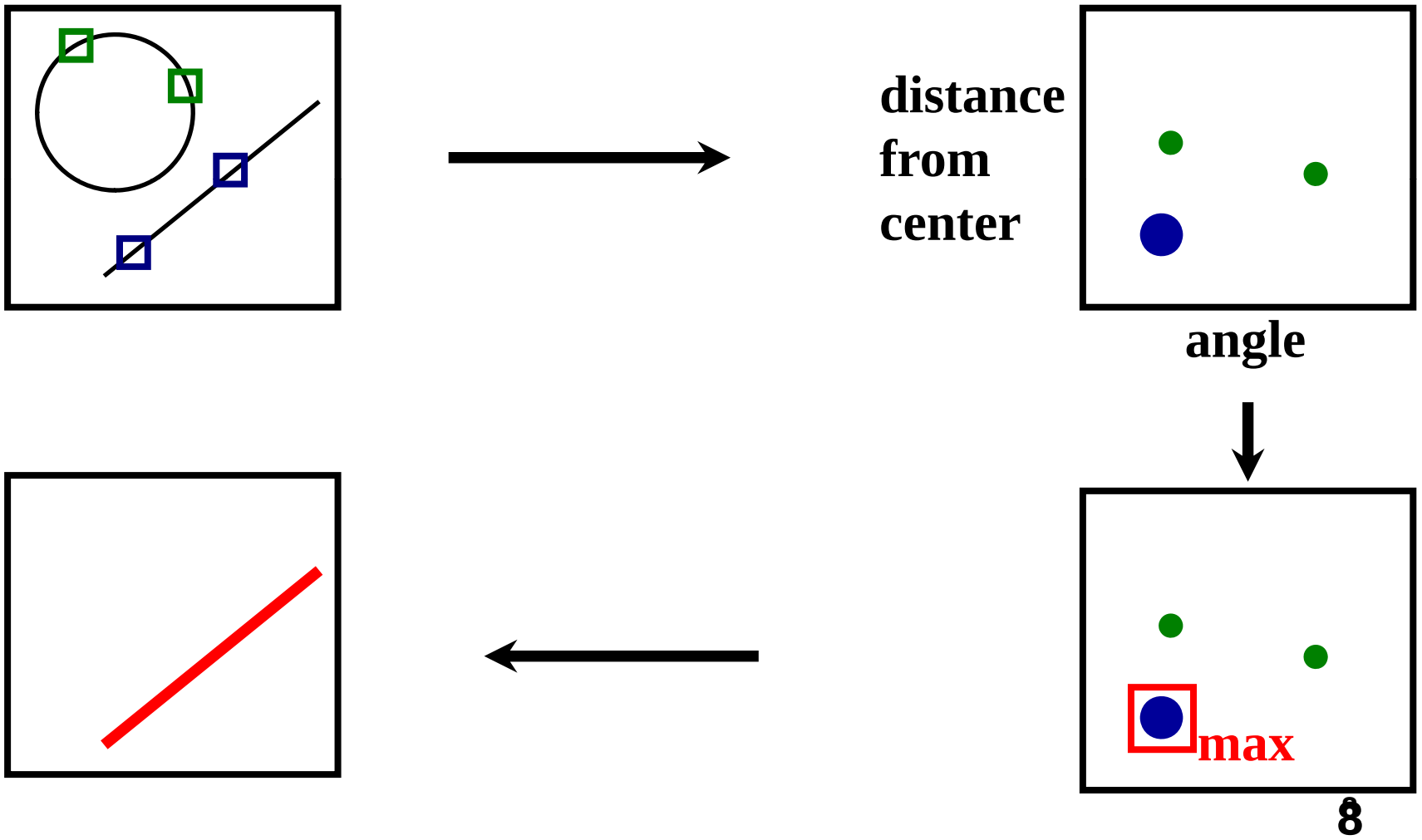
Overview

- **Implicit Shape Model:**
  - **Hough transform idea**
  - **Non-parametric object model**
  - **Voting scheme for detection**
  - **Detection and segmentation**
  - **Limitations and outlook**



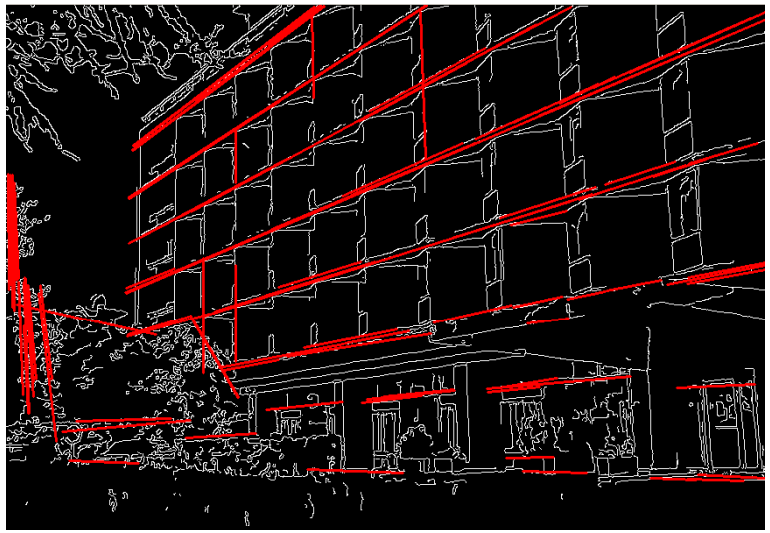
Hough Transform

- **Simple Examle: find lines in image**



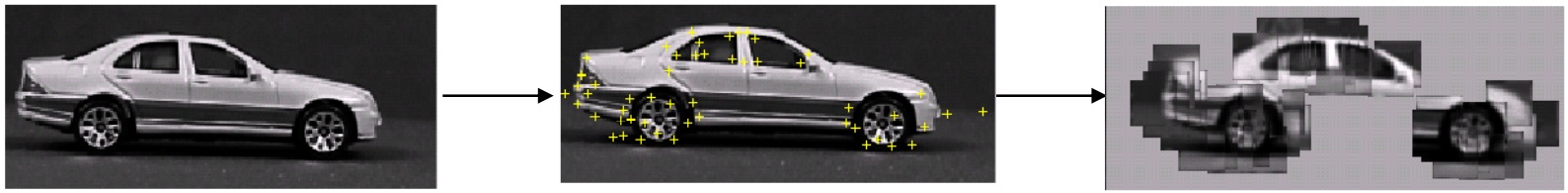
Hough Transform

- **example:**



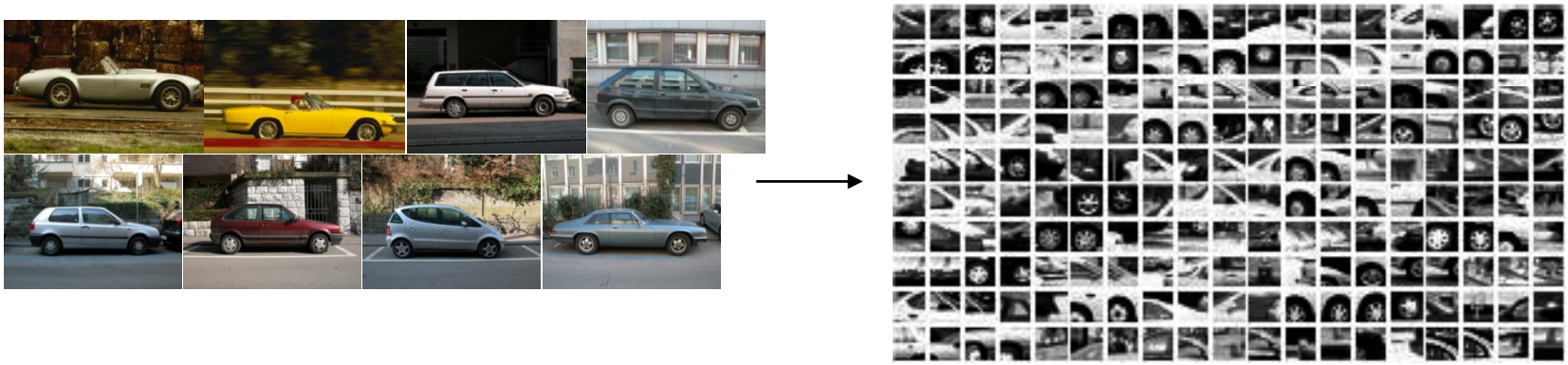
Codebook Representation

- **Extraction of local object patches**
  - **scale-invariant interest points (difference of gaussian)**

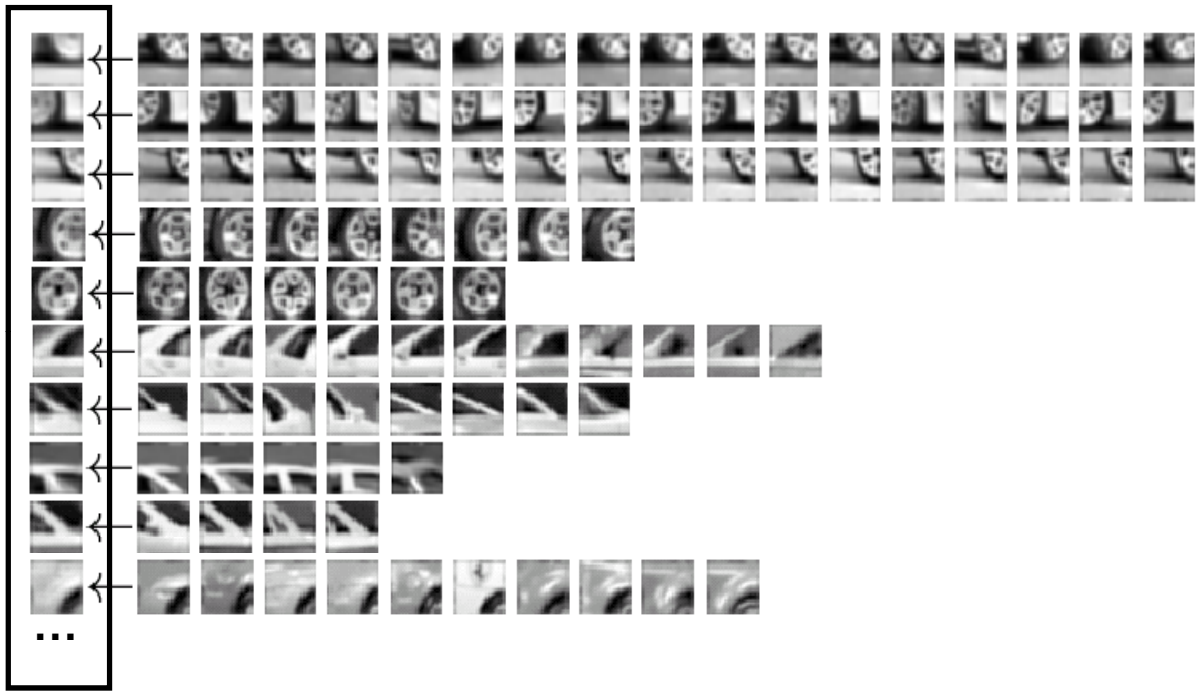


- **Collect patches from whole training set**

- **Example:**



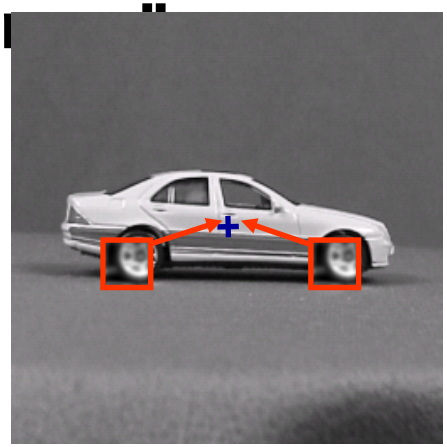
Codebook Representation



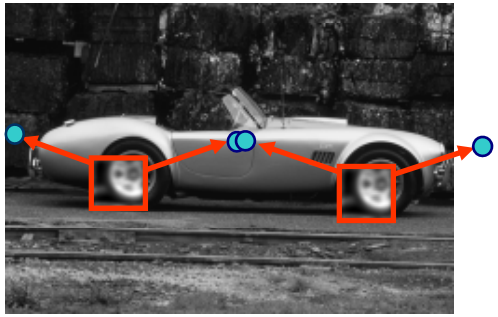
- **50 car images**
- **only side views were used**

Implicit Shape Model (ISM)

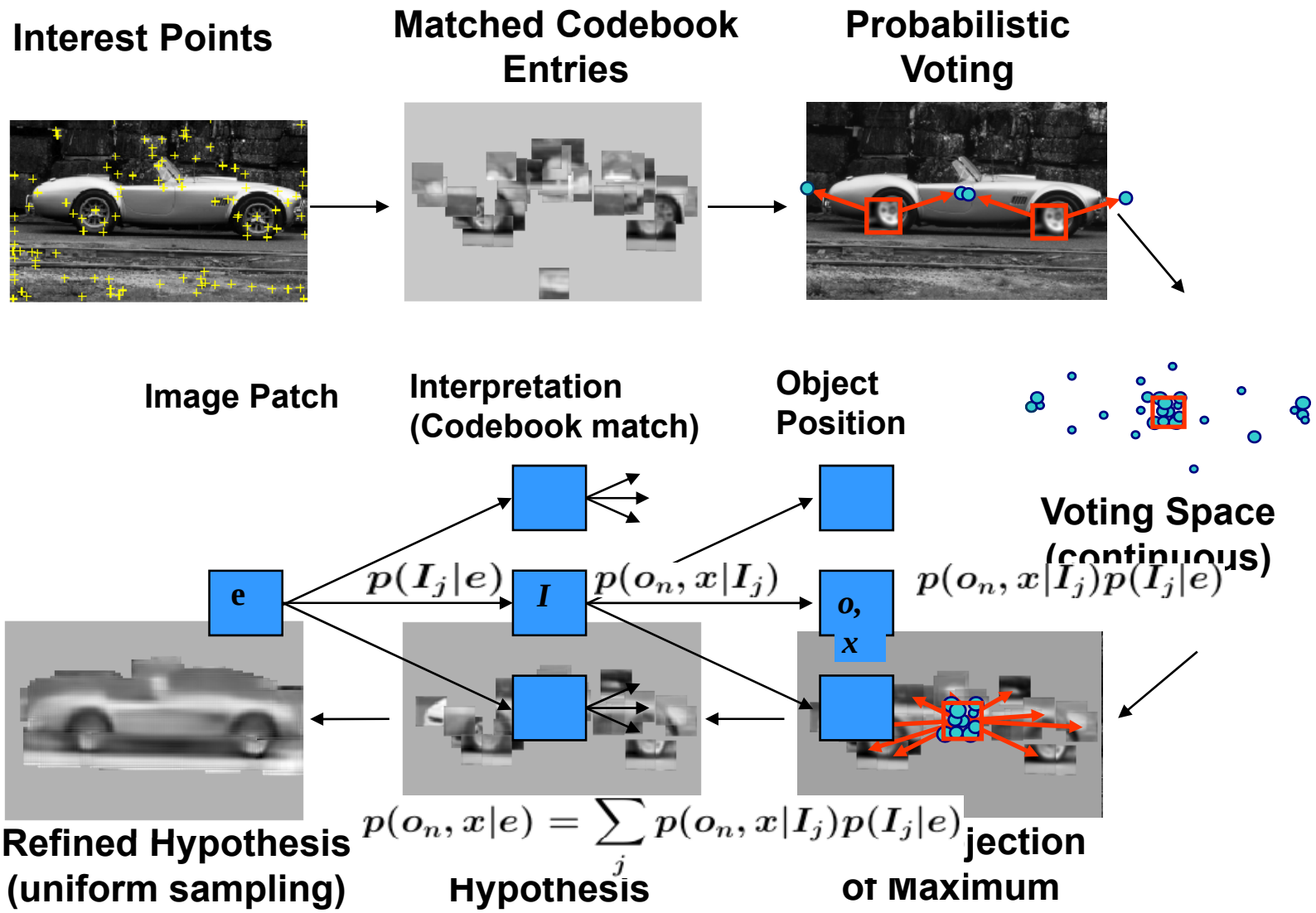
- For every codebook entry, store possible "occurre



- For new image, let the matched patches vote for possible object positions



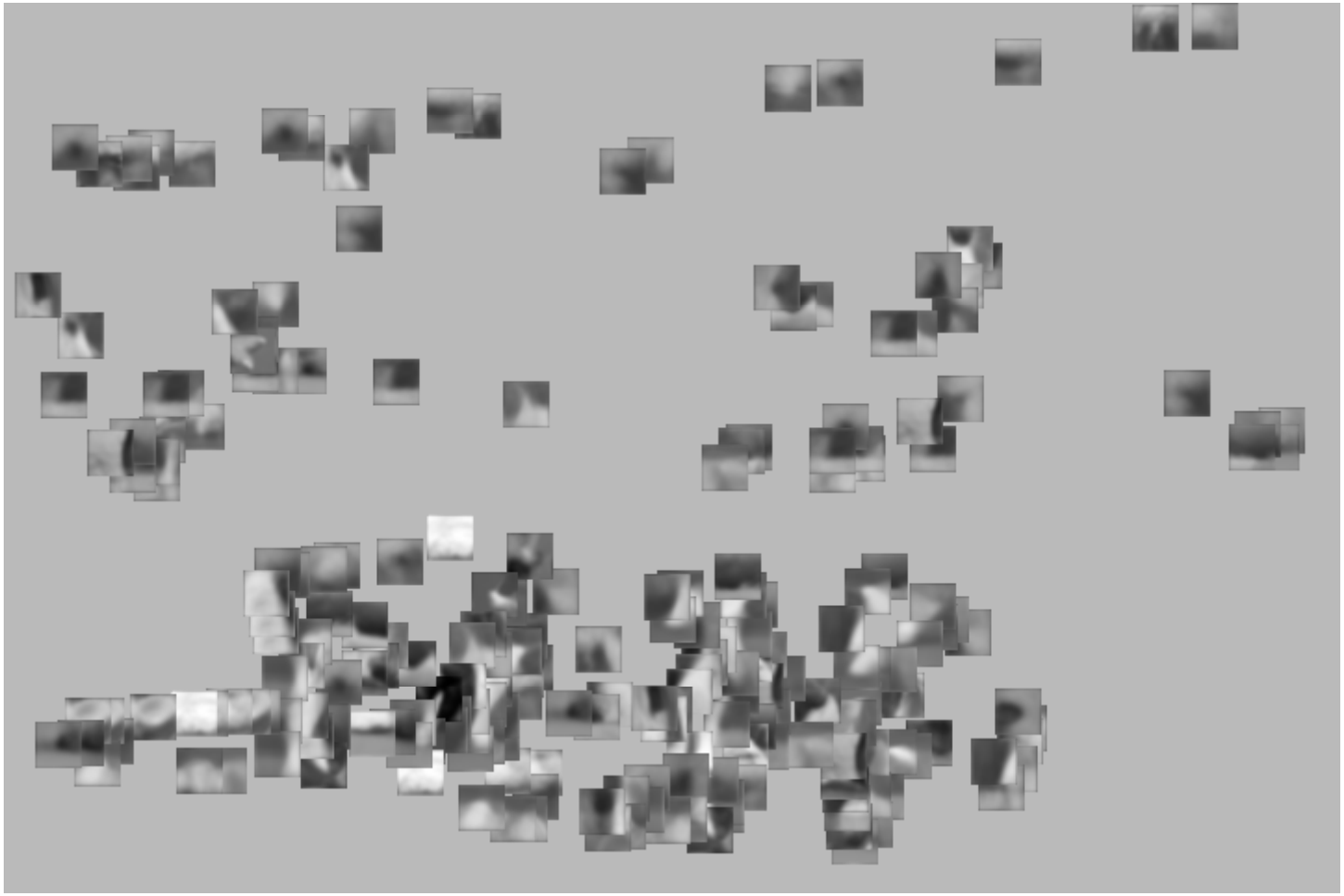
Object Categorization Procedure



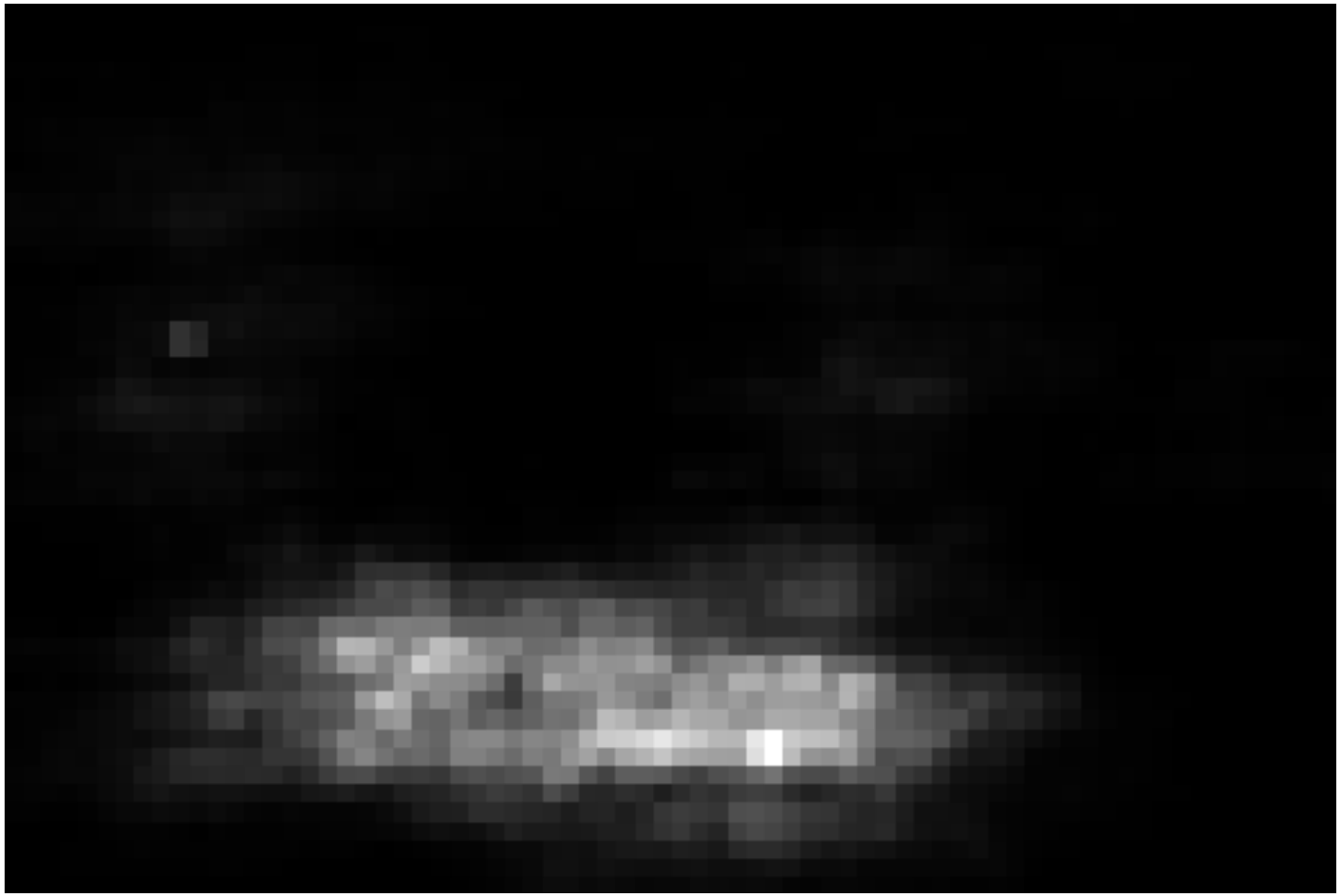
Results on Cows

Results on Cows

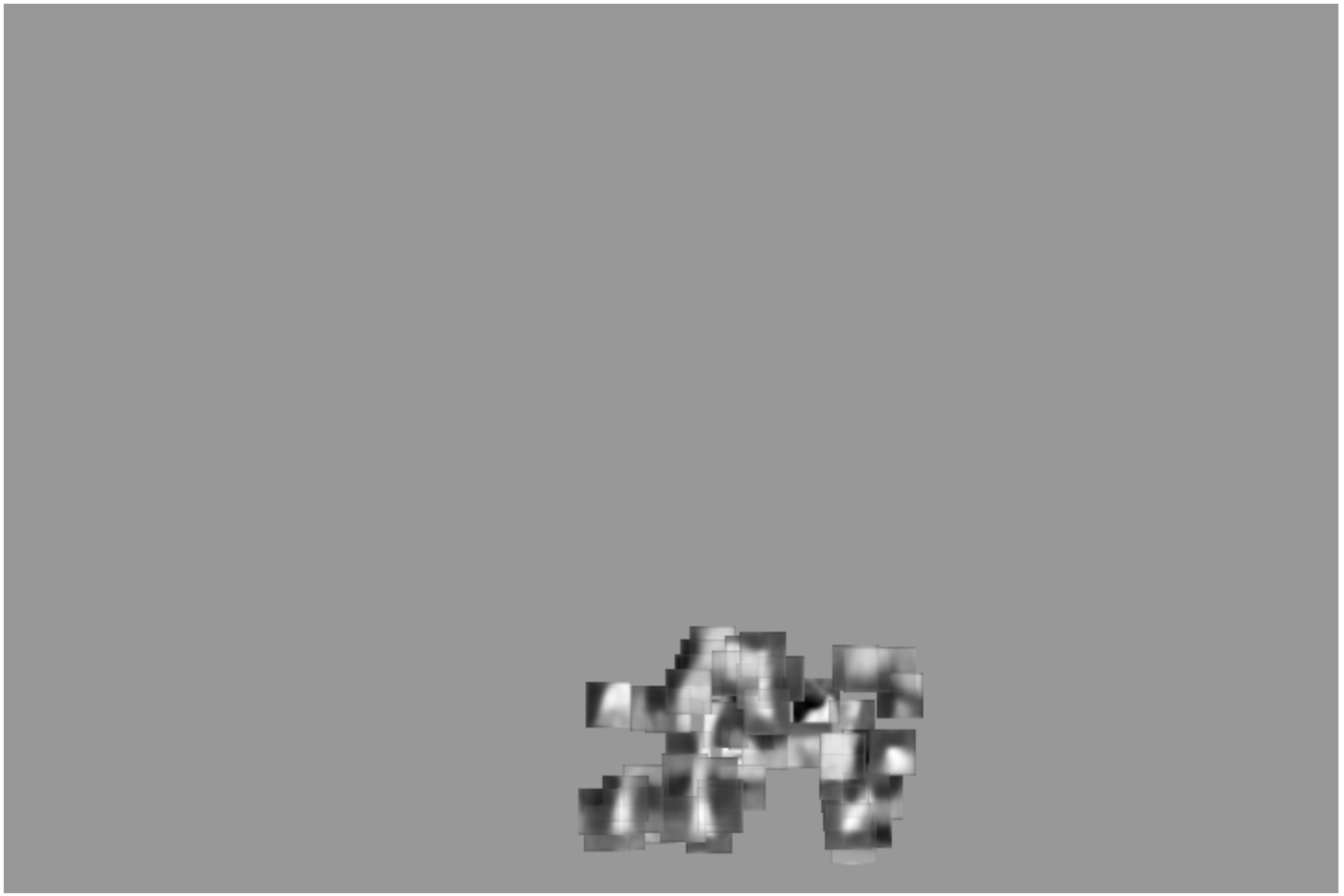
Results on Cows



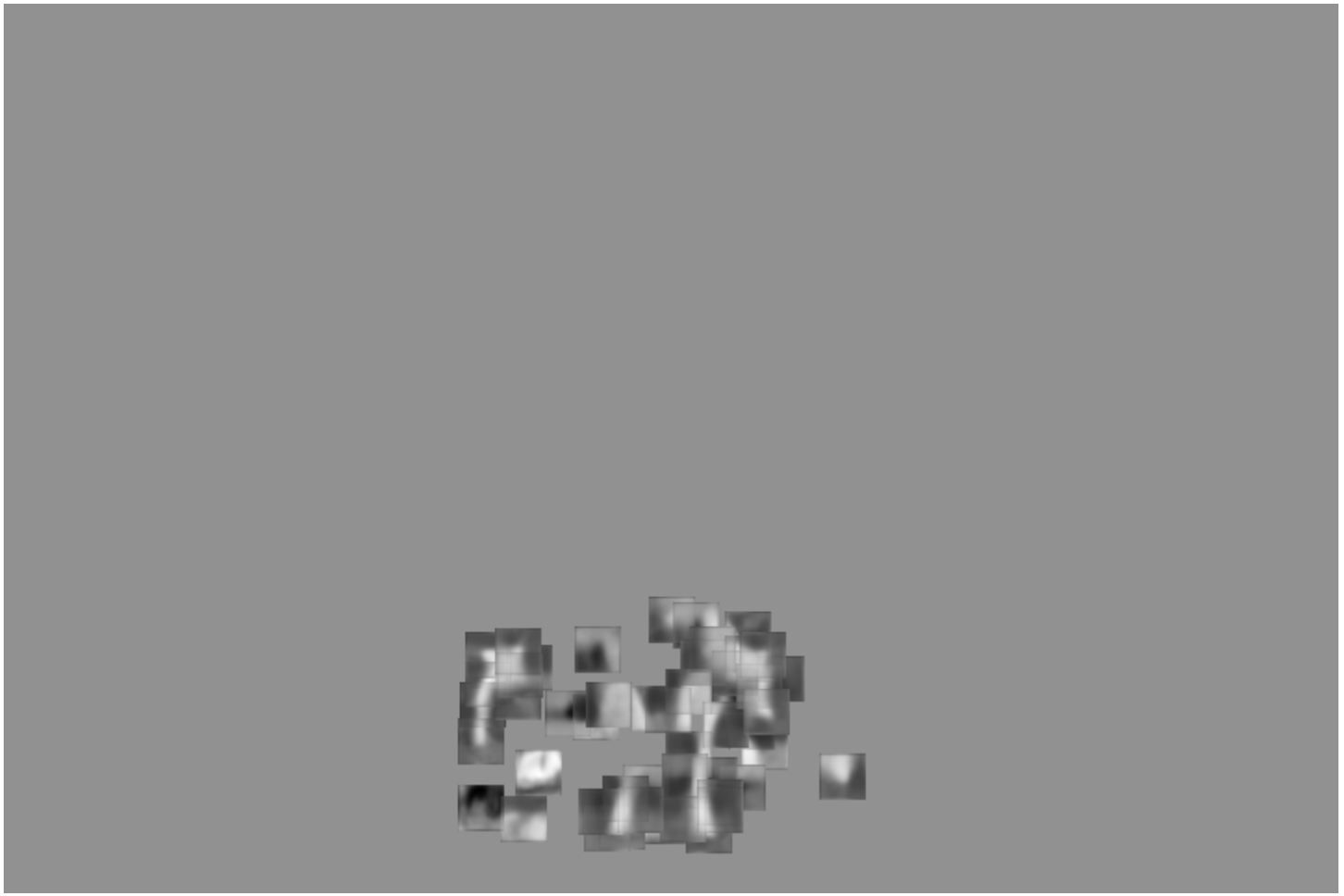
Results on Cows



Results on Cows

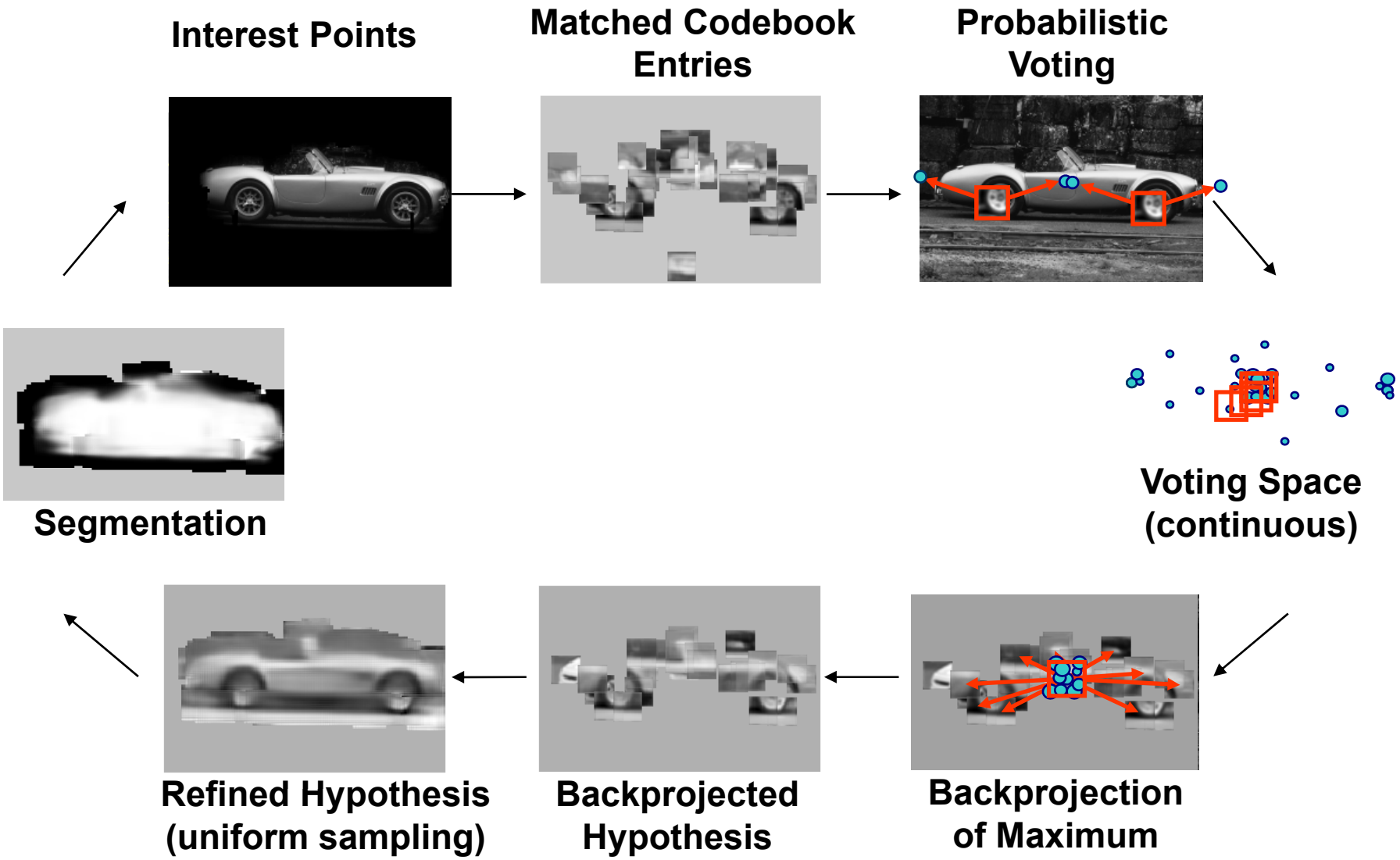


Results on Cows

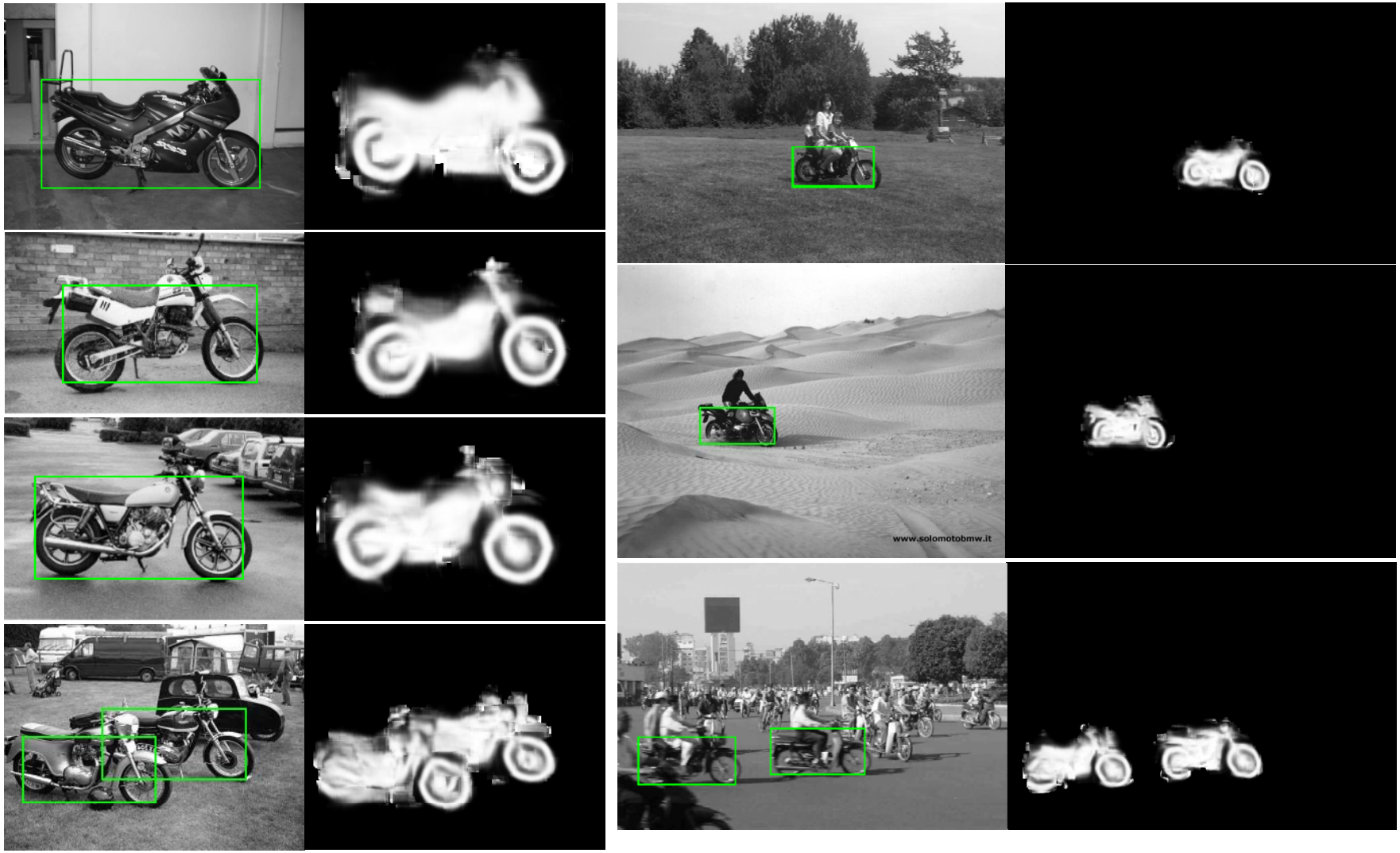


Results on Cows

Object Categorization Procedure



Motorbikes: Detection/Segmentation Results



Results on New Sequences

- **Object Detections**

QuickTime™ and a
YUV420 codec decompressor
are needed to see this picture.

Results on New Sequences

- **Segmentation**

QuickTime™ and a
YUV420 codec decompressor
are needed to see this picture.

Segmentation Based Hypothesis Verification

- **Secondary hypotheses**
  - **Desired property of algorithm! ® robustness to occlusion**
  - **Standard solution: reject based on bounding box**
  - **® Problematic - may lead to missing detections!**
  - **® Use segmentations to resolve ambiguities instead defining costs and savings for acceptance of hypotheses**

Formalization in MDL Framework

- **Savings of a hypothesis**

$$S_h = K_0 S_{area} - K_1 S_{model} - K_2 S_{error}$$

- **with**
  - S_{area} **: #pixels** N **in segmentation**
  - S_{model}**: model cost, assumed constant**
  - S_{error} **: estimate of error, according to**

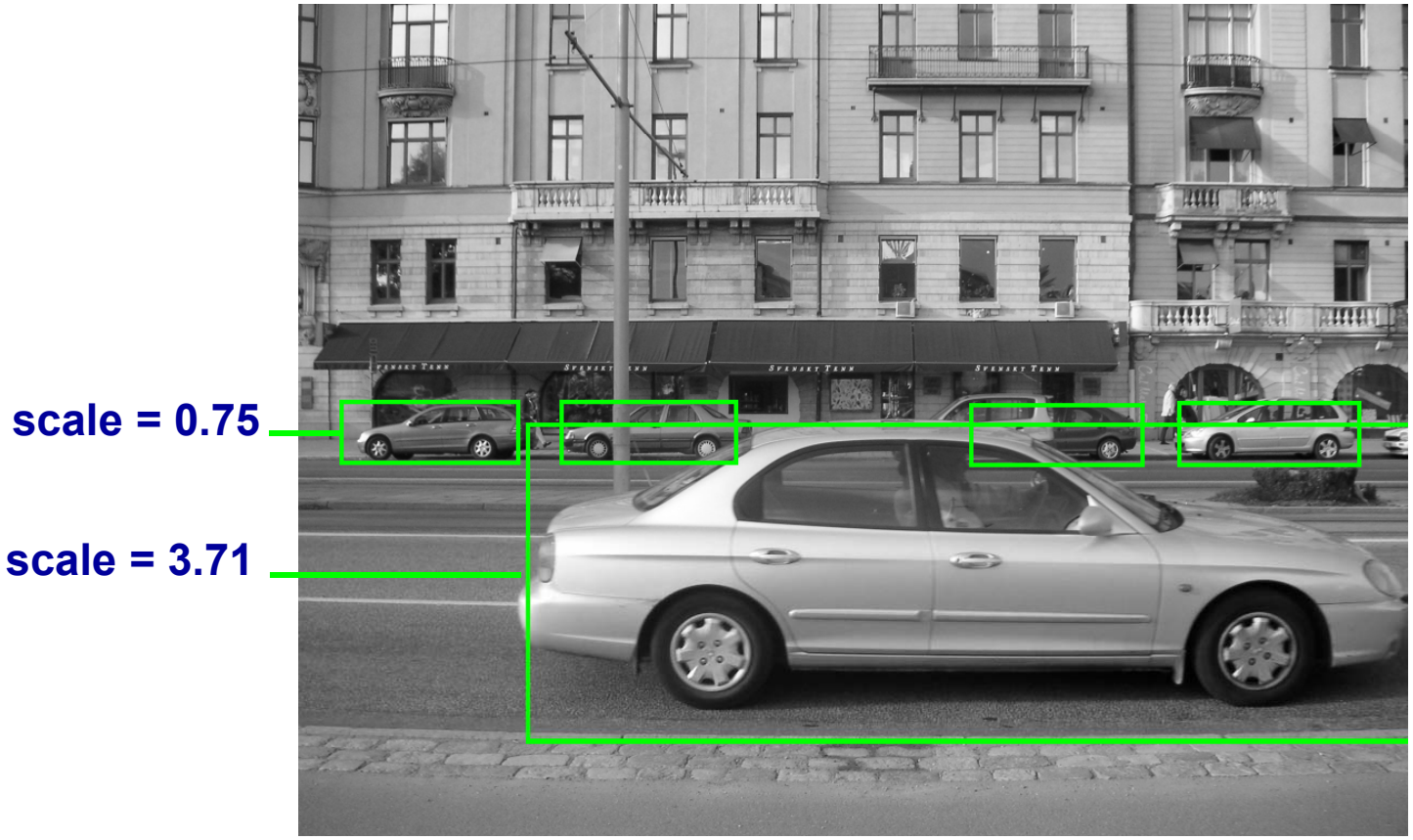
$$S_{error} = \sum_{\mathbf{p} \in Seg(h)} (1 - p(\mathbf{p} = figure|h))$$

- **Savings of** *combined* **hypothesis**

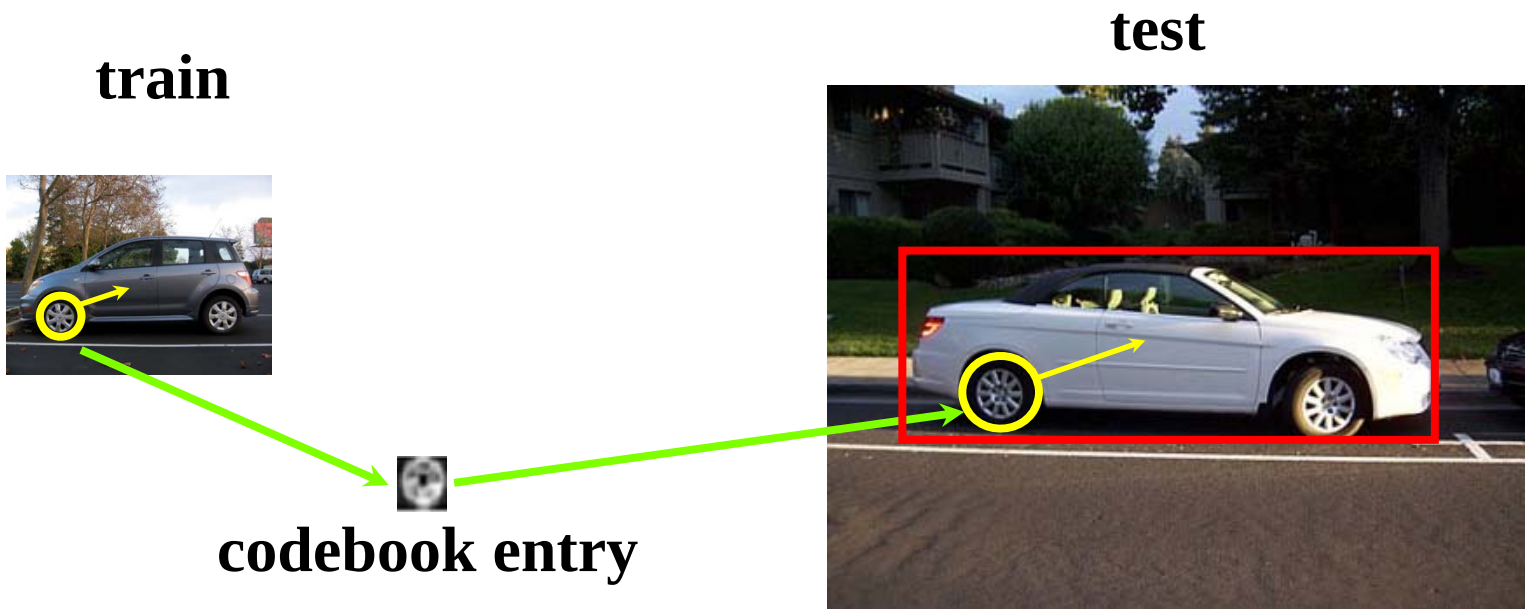
$$S_{h_1 \cup h_2} = S_{h_1} + S_{h_2} - S_{area}(h_1 \cap h_2) + S_{error}(h_1 \cap h_2)$$

- **-> greedy optimization of total savings**

Extension to Scale Invariance

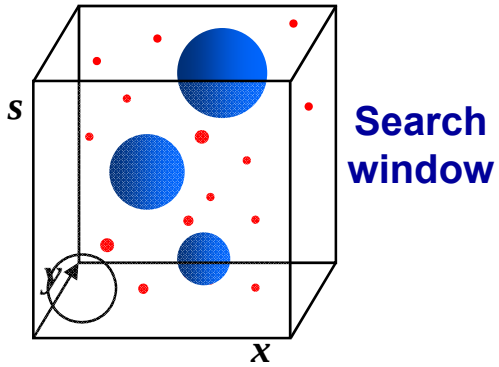


Extensions to Scale Invariance



- **Generate scale votes**
  - **Scale as 3rd dimension in voting space**

$$x_{vote} = x_{img} - x_{occ}(s_{img}/s_{occ})$$
$$y_{vote} = x_{img} - y_{occ}(s_{img}/s_{occ})$$
$$s_{vote} = (s_{img}/s_{occ})$$

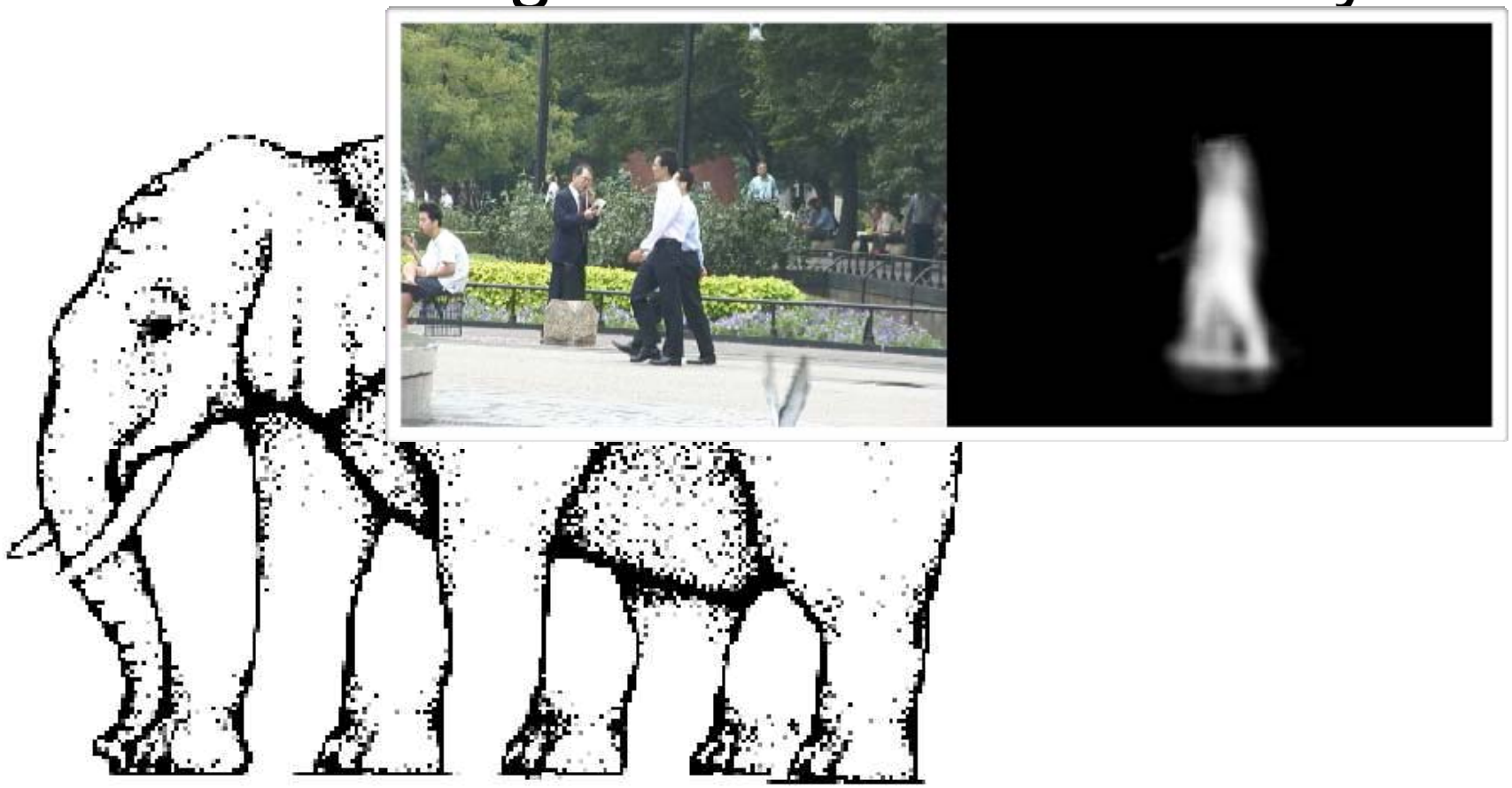


  - **Search for maxima in 3D voting space**

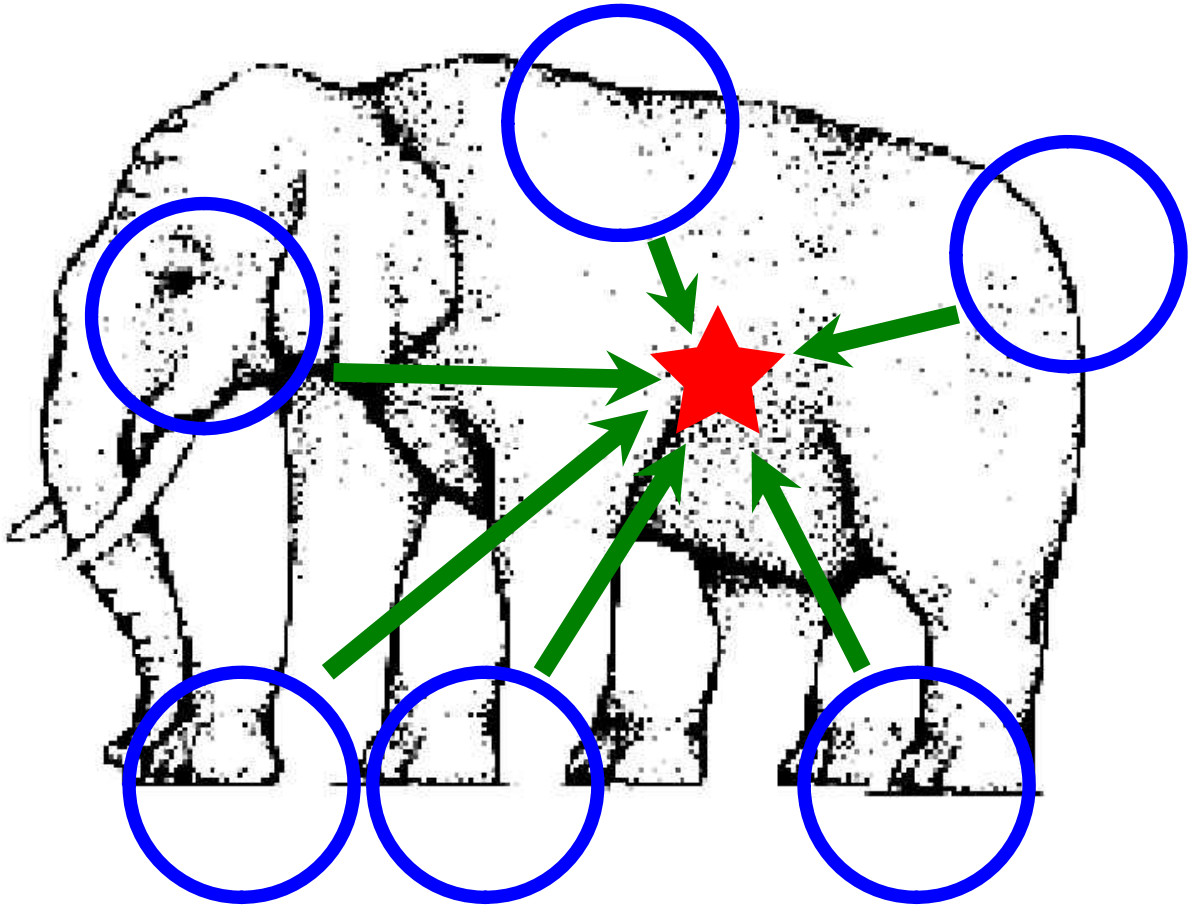
Extension to Rotation Invariance [Mikolajczyk06]

QuickTime™ and a
Microsoft Video 1 decompressor
are needed to see this picture.

Complexity of Recognition: Local Voting vs. Global Cosistency



Complexity of Recognition: Local vs. Global



star model

Outlook to Lecture on 3rd March

- Recovering global consistency
- Adding discriminance to the model

Today

- Demos
- Niebles Presentation from last week [Daphna B.]

- ISM (Leibe) – *Mario Fritz guest lecture*
- **Learning Distances (Frome) – [Ashley E.]**
- Hashing with Metric Learning – *Brian Kulis guest lecture*
- Semantic Texton Forests (Shotten)
- Forests/Ferns for Keypoint Localization (Leptit)
- Small Codes/Large Image Datasets (Torralba)

Frome Presentation Slides

- See http://www.eecs.berkeley.edu/Research/Projects/CS/vision/shape/papers/FromeSingerShaMalikICCV07_talk.pdf

Today

- Demos
- Niebles Presentation from last week [Daphna B.]

- ISM (Leibe) – *Mario Fritz guest lecture*
- Learning Distances (Frome) – [Ashley E.]
- **Hashing with Metric Learning – *Brian Kulis guest lecture***
- Semantic Texton Forests (Shotten)
- Forests/Ferns for Keypoint Localization (Leptit)
- Small Codes/Large Image Datasets (Torralba)

Fast Image Search for Learned Metrics

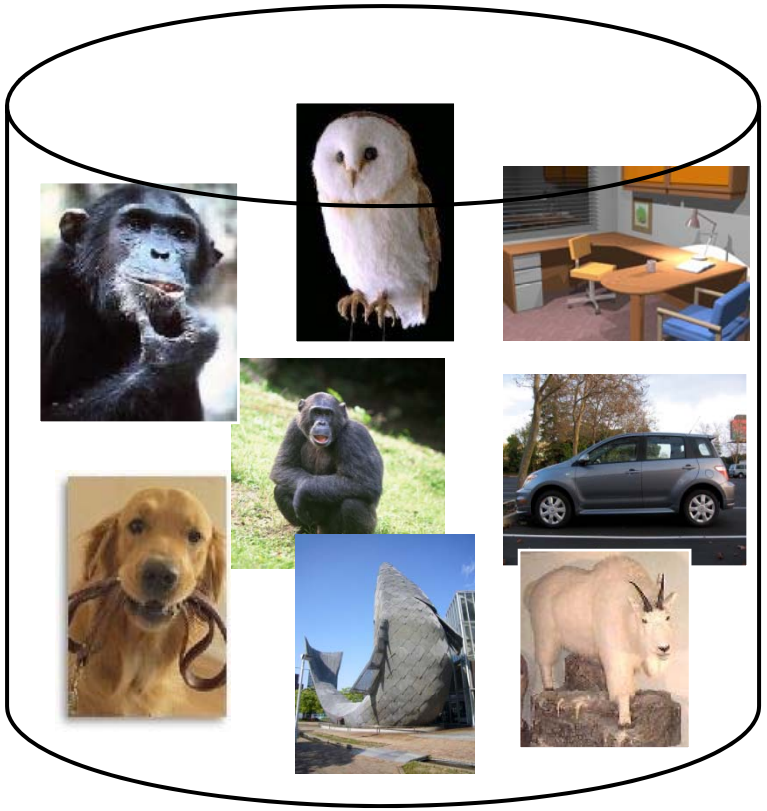
Prateek Jain, Brian Kulis, and Kristen Grauman

Motivation

- Fast image search is a useful component for a number of vision problems.

Object categorization

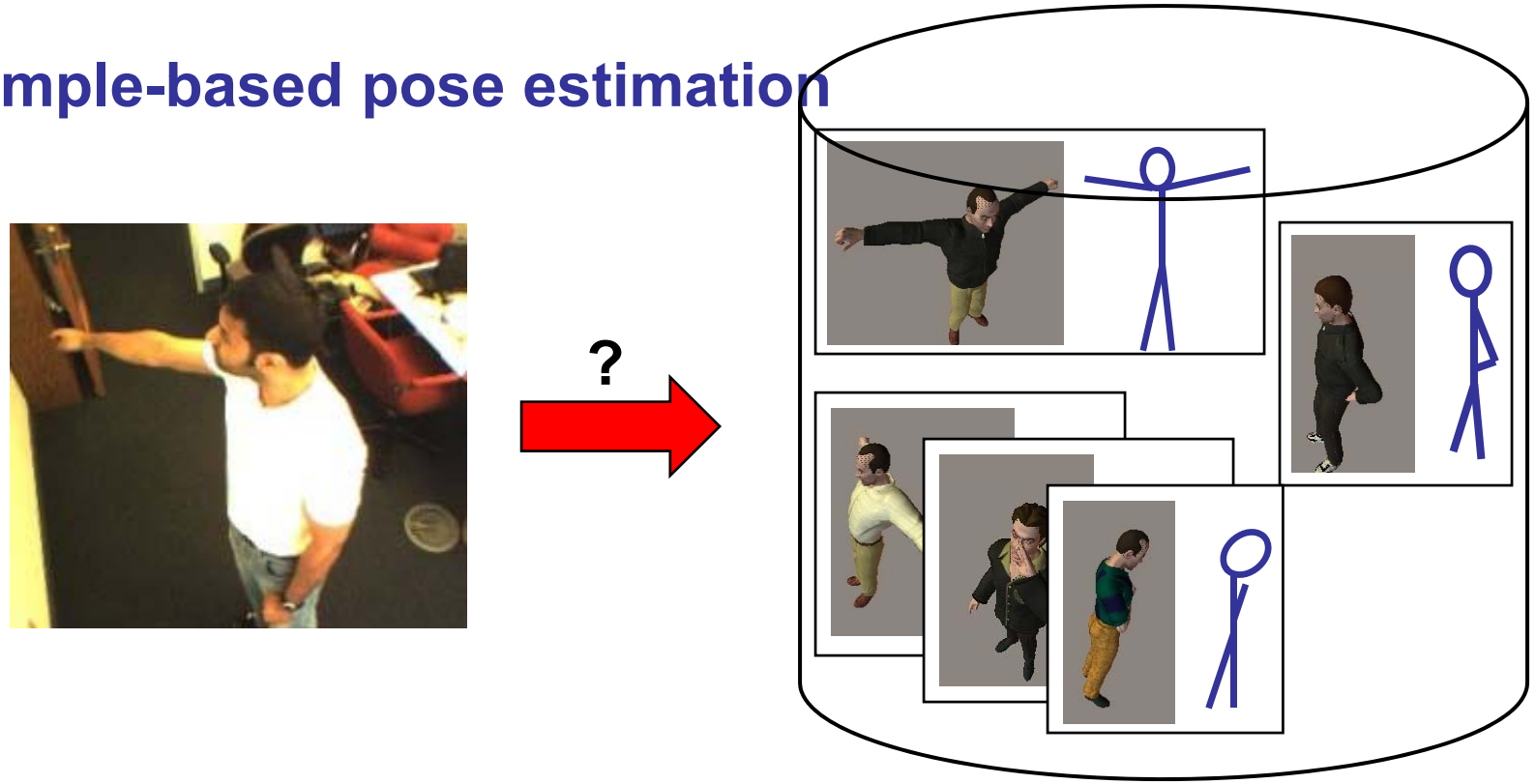
?



Motivation

- **Fast image search is a useful component for a number of vision problems.**

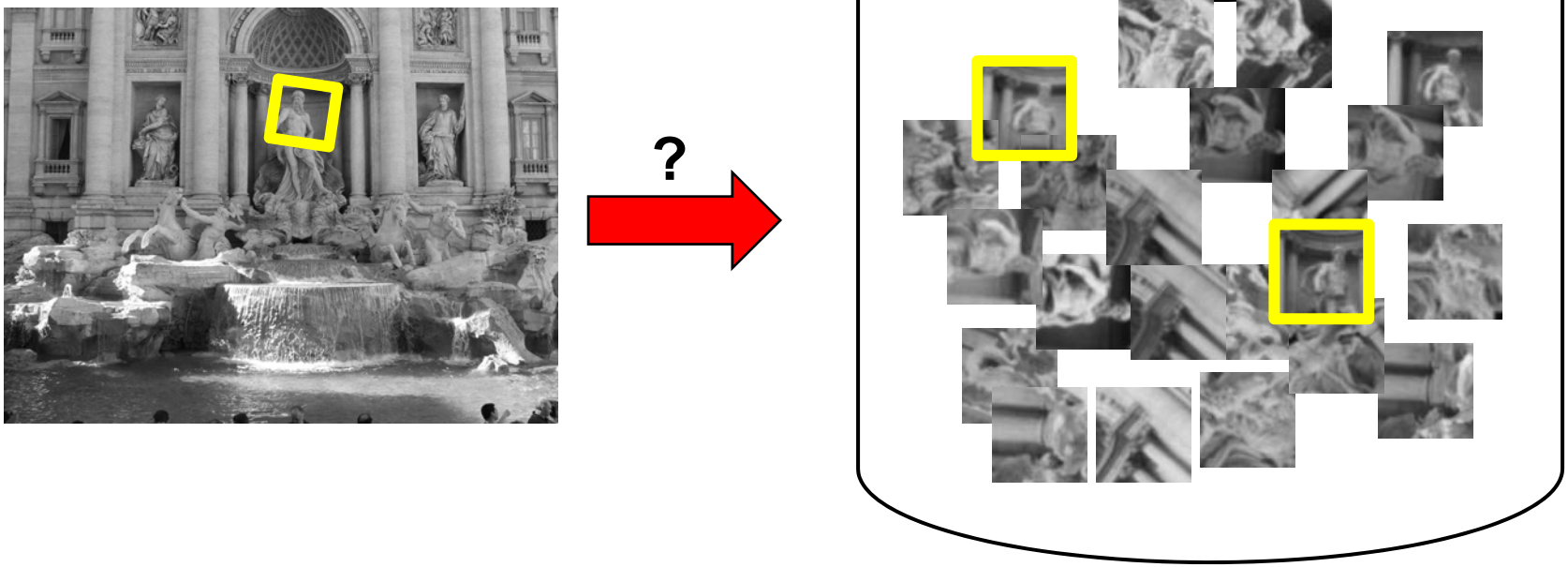
Example-based pose estimation



Motivation

- **Fast image search is a useful component for a number of vision problems.**

Structure from Motion



Problem

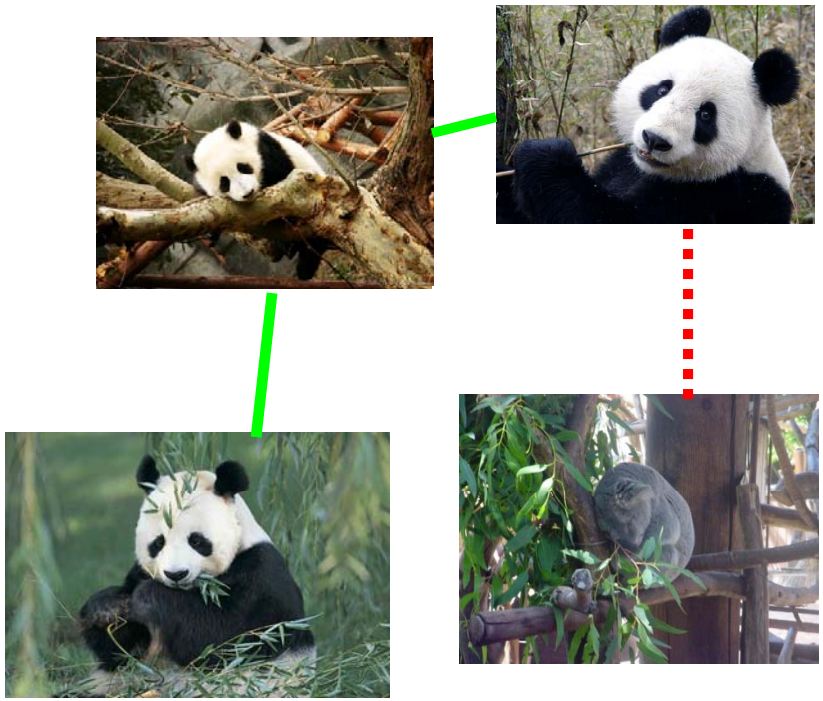
- Search must be both **fast** and **accurate**
  - "Generic" distances or low-dimensional representations amenable to fast search, but may be inaccurate for a given problem.
  - *Learned* task-specific distance functions more accurate, but current methods cannot guarantee fast search for them.
- Our approach:
  - Develop approximate similarity search method for learned metrics
  - Encode side-information into randomized locality-sensitive hash functions
  - Applicable for a variety of image search tasks

Related work

- Metric learning for image distances
  - Weinberger et al. 2004, Hertz et al. 2004, Frome et al. 2007, Varma & Ray 2007
- Embedding functions to reduce cost of expensive distances
  - Athitsos et al. 2004, Grauman & Darrell 2005, Torralba et al. 2008
- Search structures based on spatial partitioning and recursive decompositions
  - Beis & Lowe 1997, Obdrzalek & Matas 2005, Nister & Stewenius 2006, Uhlmann 1991

- **Locality-sensitive hashing (LSH) for vision applications**
  - **Shakhnarovich et al. 2003, Frome et al. 2004, Grauman & Darrell 2004**
- **Data-dependent variants of LSH**
  - **Shakhnarovich et al. 2003, Georgescu et al. 2003**

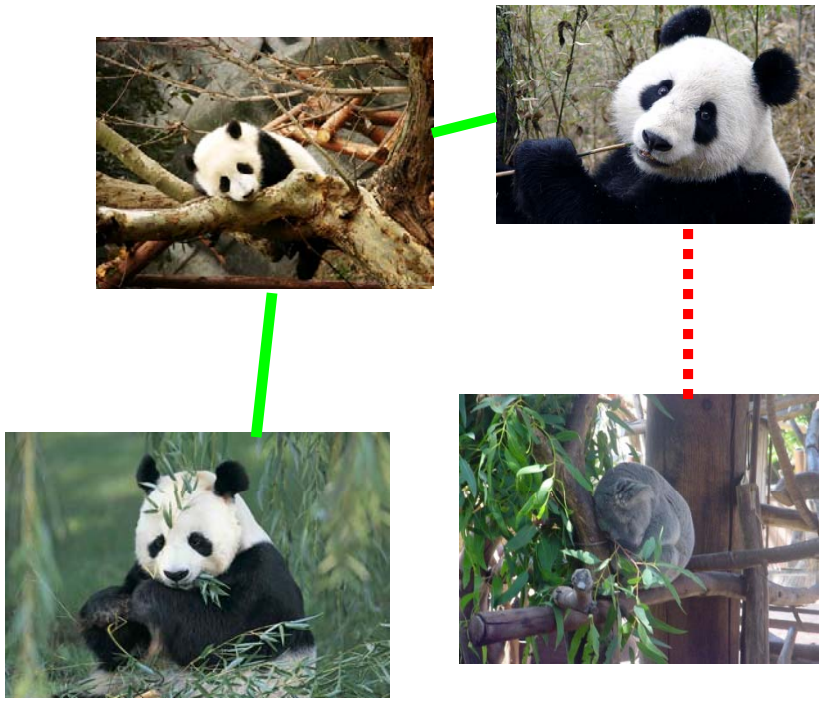
Metric learning



There are various ways to judge appearance/shape similarity…

but often we know more about (some) data than just their appearance.

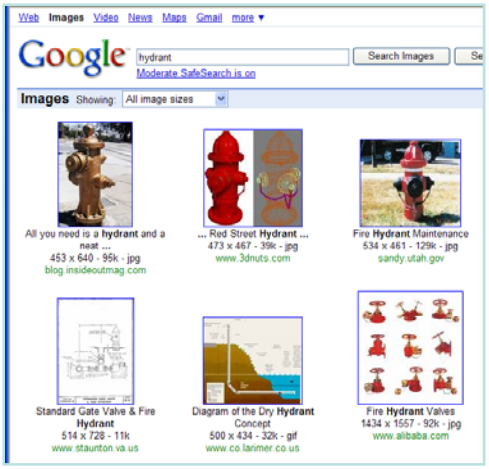
Metric learning



- **Exploit partially labeled data and/or (dis)similarity constraints to construct more useful distance function**
- **Various existing techniques**

Example sources of similarity constraints

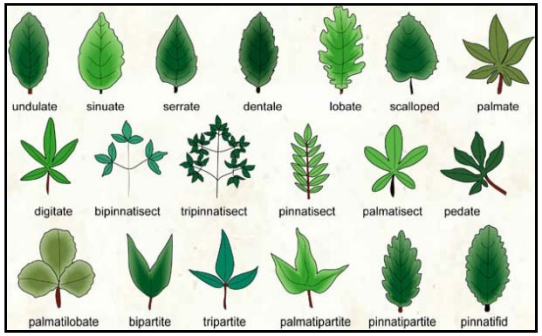
Partially labeled image databases



User feedback

Detected video shots, tracked objects

Fully labeled image databases



Problem-specific knowledge

Mahalanobis distances

- Distance parameterized by p.d. *d* × *d* matrix *A*:

$$d_A(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i - \boldsymbol{x}_j)^T A(\boldsymbol{x}_i - \boldsymbol{x}_j)$$

- Similarity measure is associated generalized inner product (kernel)

$$s_A(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i^T A \boldsymbol{x}_j.$$

Information-theoretic (LogDet) metric learning

- Formulation:

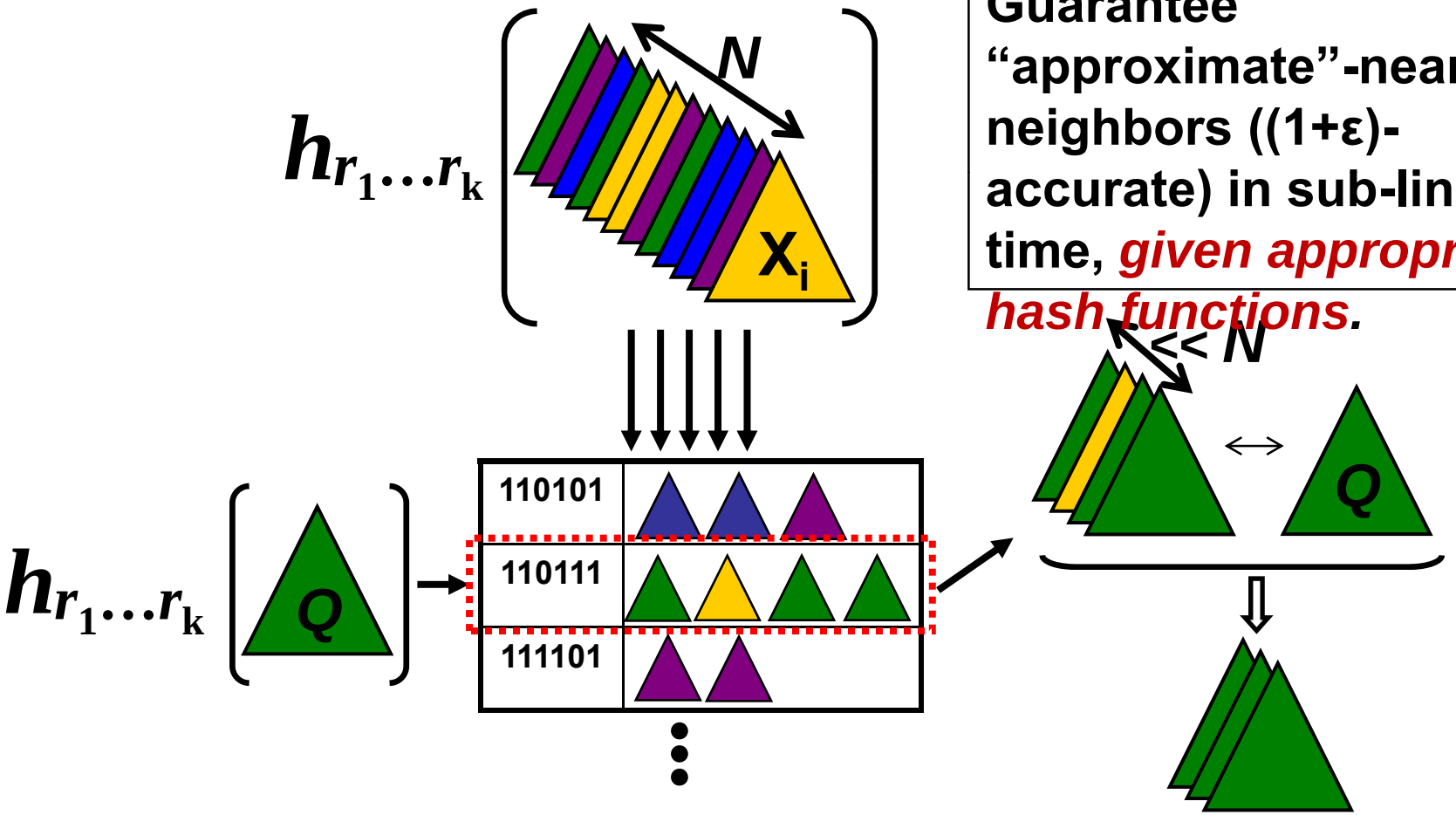
$$
\begin{aligned}
\min_A \quad & D_{\ell d}(A, A_0) \\
\text{s.t.} \quad & (\mathbf{x}_i - \mathbf{x}_j)^T A (\mathbf{x}_i - \mathbf{x}_j) \leq u \quad \text{if } (i,j) \in \mathcal{S} \text{ [similarity constraints]} \\
& (\mathbf{x}_i - \mathbf{x}_j)^T A (\mathbf{x}_i - \mathbf{x}_j) \geq \ell \quad \text{if } (i,j) \in \mathcal{D} \text{ [dissimilarity constraints]}
\end{aligned}
$$

- **Advantages:**
  - **-Simple, efficient algorithm**
  - **-Can be applied in kernel space**

[Davis, Kulis, Jain, Sra, and Dhillon, ICML 2007]

Locality Sensitive Hashing (LSH)

$$\Pr_{h \in \mathcal{F}}[h(x) = h(y)] = sim(x, y)$$



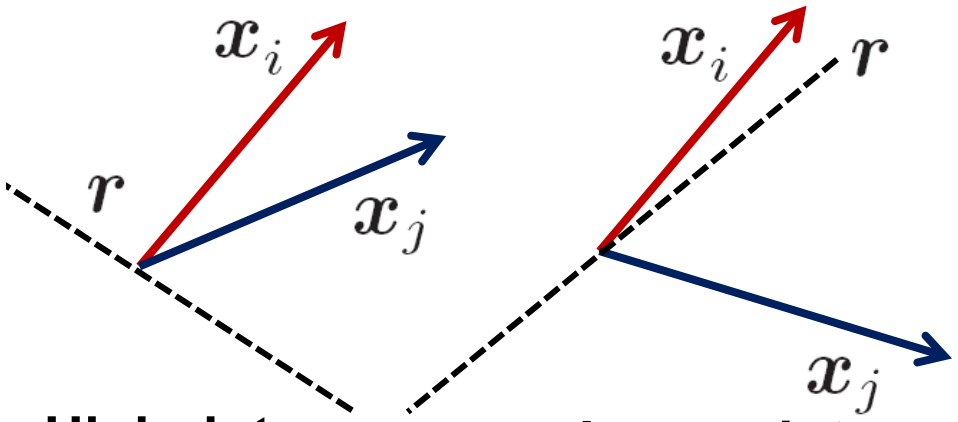
Guarantee "approximate"-nearest neighbors ((1+ε)-accurate) in sub-linear time, *given appropriate hash functions*.

[Indyk and Motwani 1998, Charikar

LSH functions for dot products

The probability that a *random hyperplane* separates two unit vectors depends on the angle between them:

$$\Pr[\text{sign}(\boldsymbol{x}_i^T \boldsymbol{r}) = \text{sign}(\boldsymbol{x}_j^T \boldsymbol{r})] = 1 - \frac{1}{\pi}\cos^{-1}(\boldsymbol{x}_i^T \boldsymbol{x}_j)$$



High dot product: unlikely to split

Lower dot product: likely to split

Corresponding hash function:

$$h_{\boldsymbol{r}}(\boldsymbol{x}) = \begin{cases} 1, & \text{if } \boldsymbol{r}^T \boldsymbol{x} \geq 0 \\ 0, & \text{otherwise} \end{cases}$$

[Goemans and Williamson 1995, Charika

LSH functions for learned metrics

- Given learned metric with $A = G^T G$
- We generate parameterized hash functions for $s_A(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i^T A \boldsymbol{x}_j$:

$$h_{\boldsymbol{r},A}(\boldsymbol{x}) = \begin{cases} 1, & \text{if } \boldsymbol{r}^T G \boldsymbol{x} \geq 0 \\ 0, & \text{otherwise} \end{cases}$$

This satisfies the locality-sensitivity condition:

$$\Pr\left[h_{\boldsymbol{r},A}(\boldsymbol{x}_i) = h_{\boldsymbol{r},A}(\boldsymbol{x}_j)\right] = 1 - \frac{1}{\pi}\cos^{-1}\left(\frac{\boldsymbol{x}_i^T A \boldsymbol{x}_j}{\sqrt{|G\boldsymbol{x}_i||G\boldsymbol{x}_j|}}\right)$$

Implicit hashing formulation

- Image data often high-dimensional—must work in kernel space
- High-d inputs are sparse, but $A = G^T G$ may be dense $\longrightarrow$ can't work with $\boldsymbol{r}^T G \boldsymbol{x}$.

- We derive an implicit update rule that simultaneously updates metric and hash function parameters.
- **Integrates** metric learning and hashing

Implicit hashing formulation

We show that the same hash function can be computed indirectly via:

$$G = I + XSX^T$$

Possible due to property of information-theoretic metric learning

S is c x c matrix of coefficients that determine how much weight each pair of the *c* constrained inputs contributes to learned parameters.

Recap: data flow

1. Receive constraints and base metric.
2. Learning stage: simultaneously update metric and hash functions.
3. Hash database examples into table.
4. When a query arrives, hash into existing table for approximate neighbors under learned metric.

Results

Object Categorization

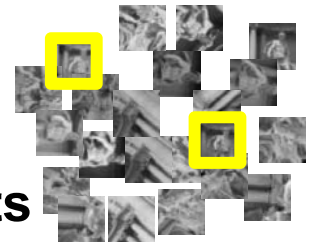
Caltech 101, O(10^6) dimensions, 4k points

Pose Estimation

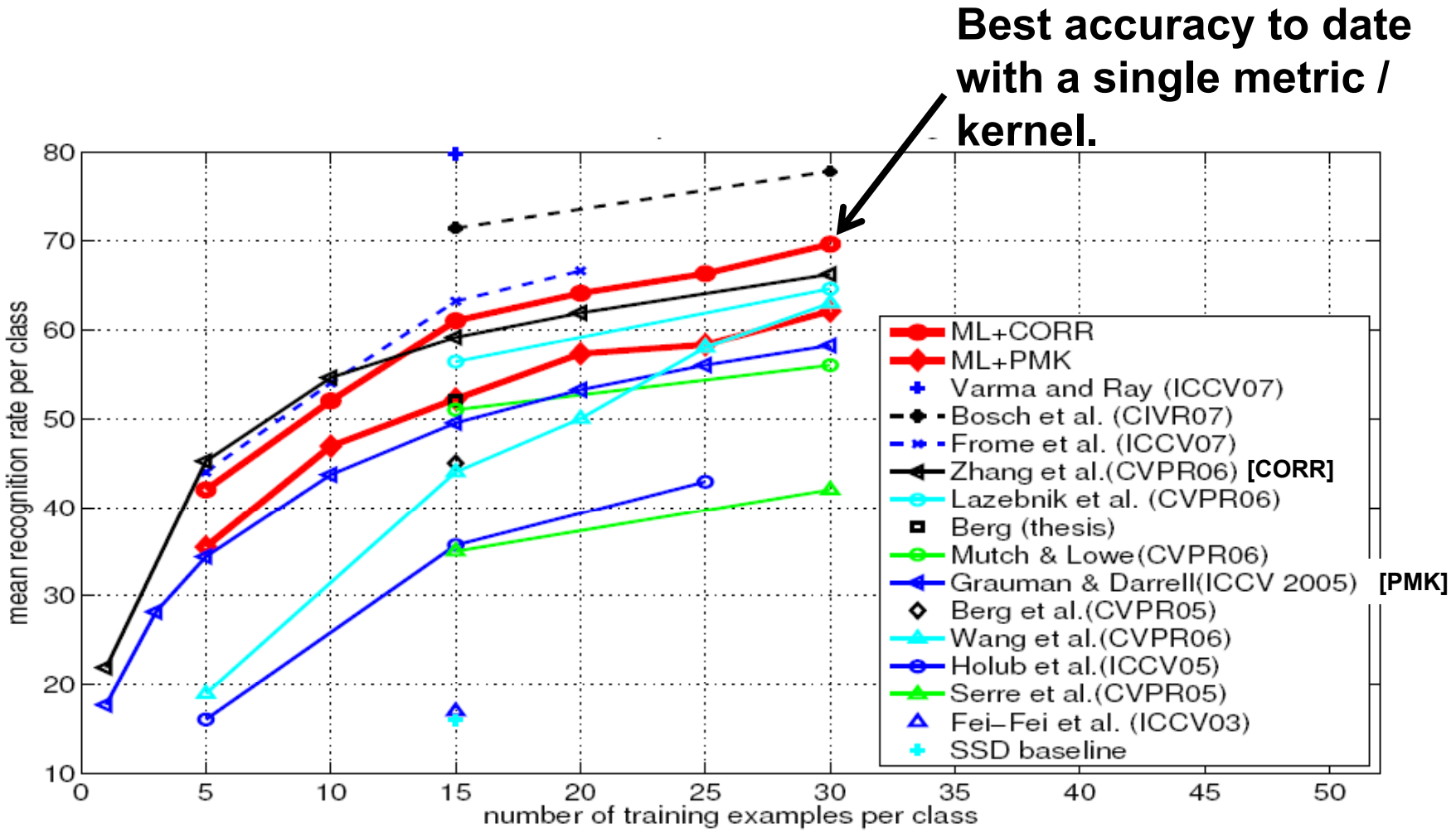
Poser data, 24k dimensions, .5 million points

Patch Indexing

Photo Tourism data, 4096 dimensions, 300k points



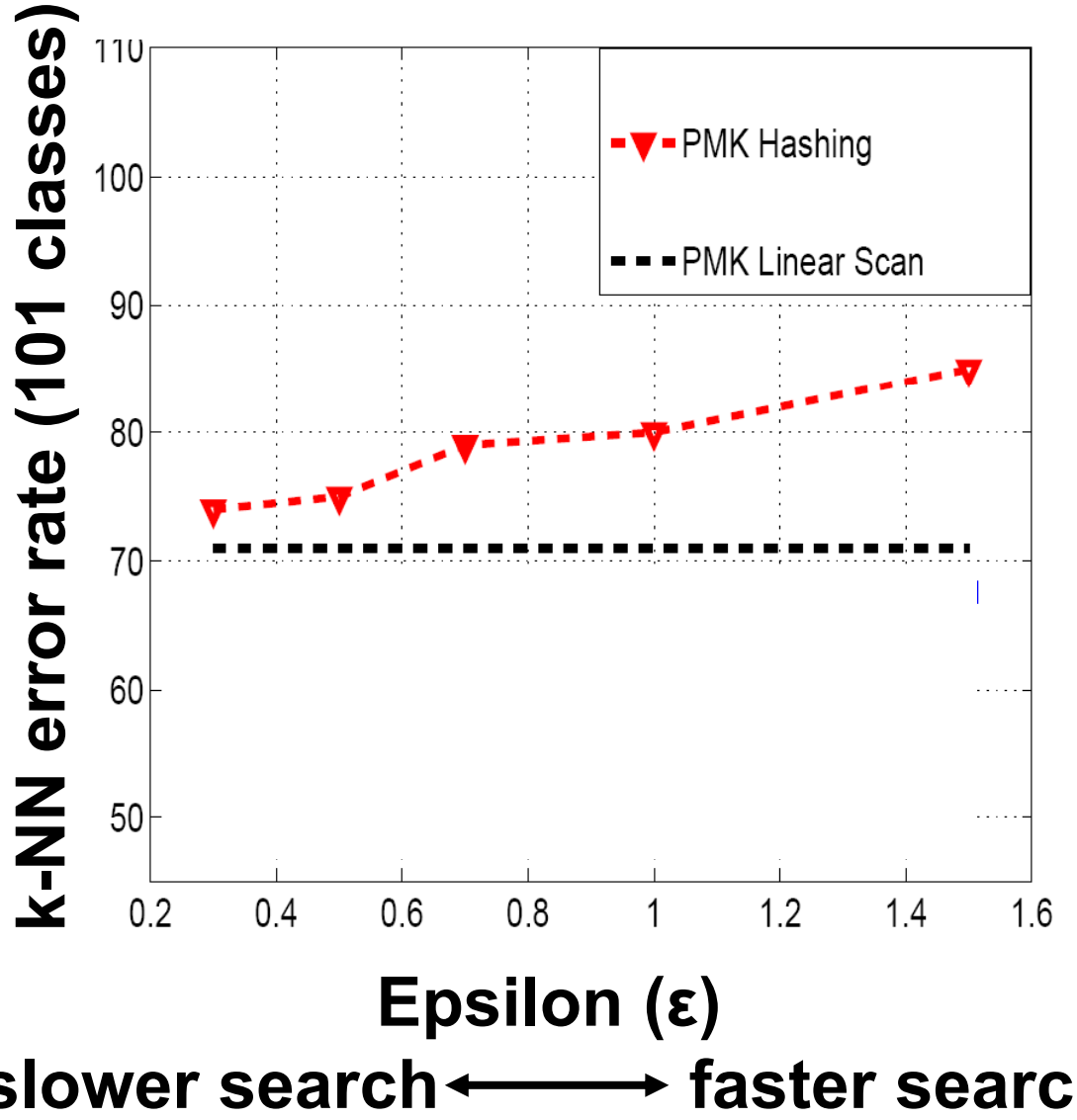
Results: object categorization



Caltech-101 database

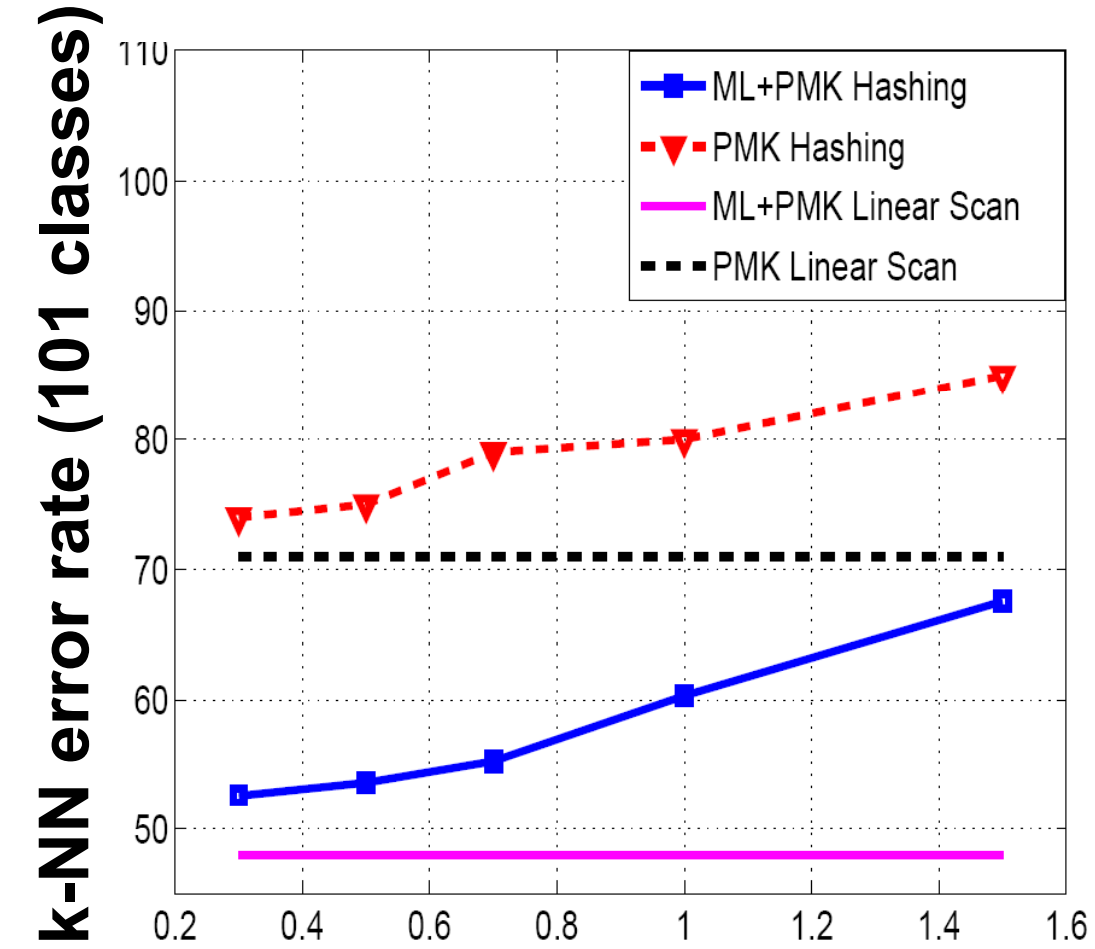
ML = metric learn

Results: object categorization



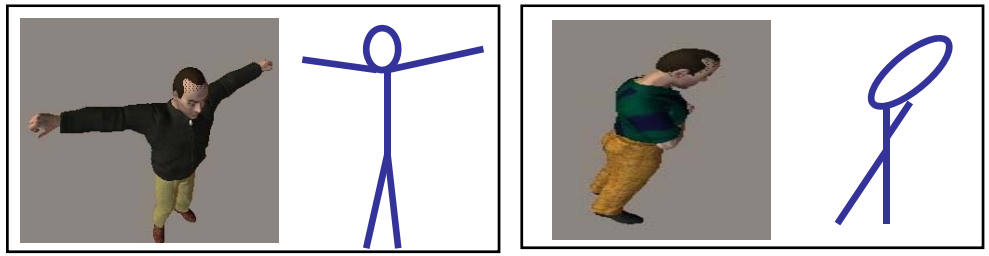
- **Query time controlled by required accuracy**
- **e.g., search less than 2% of database examples for accuracy close to linear scan**

Results: object categorization



- **Query time controlled by required accuracy**
- **e.g., search less than 2% of database examples for accuracy close to linear scan**

Results: pose estimation



- 500,000 synthetic images
- Measure mean error per joint between query and NN
  - Random 2 database images: 34.5 cm between each joint
- Average query time:
  - ML linear scan: 433.25 sec
  - ML hashing: 1.39 sec

Method	d	Error (cm)
L_2 linear scan	24K	8.9
L_2 hashing	24K	9.4
PSH, linear scan	1.5K	9.4
PCA, linear scan	60	13.5
ML PCA, lin. scan	60	13.1
ML linear scan	24K	8.4
ML hashing	24K	8.8

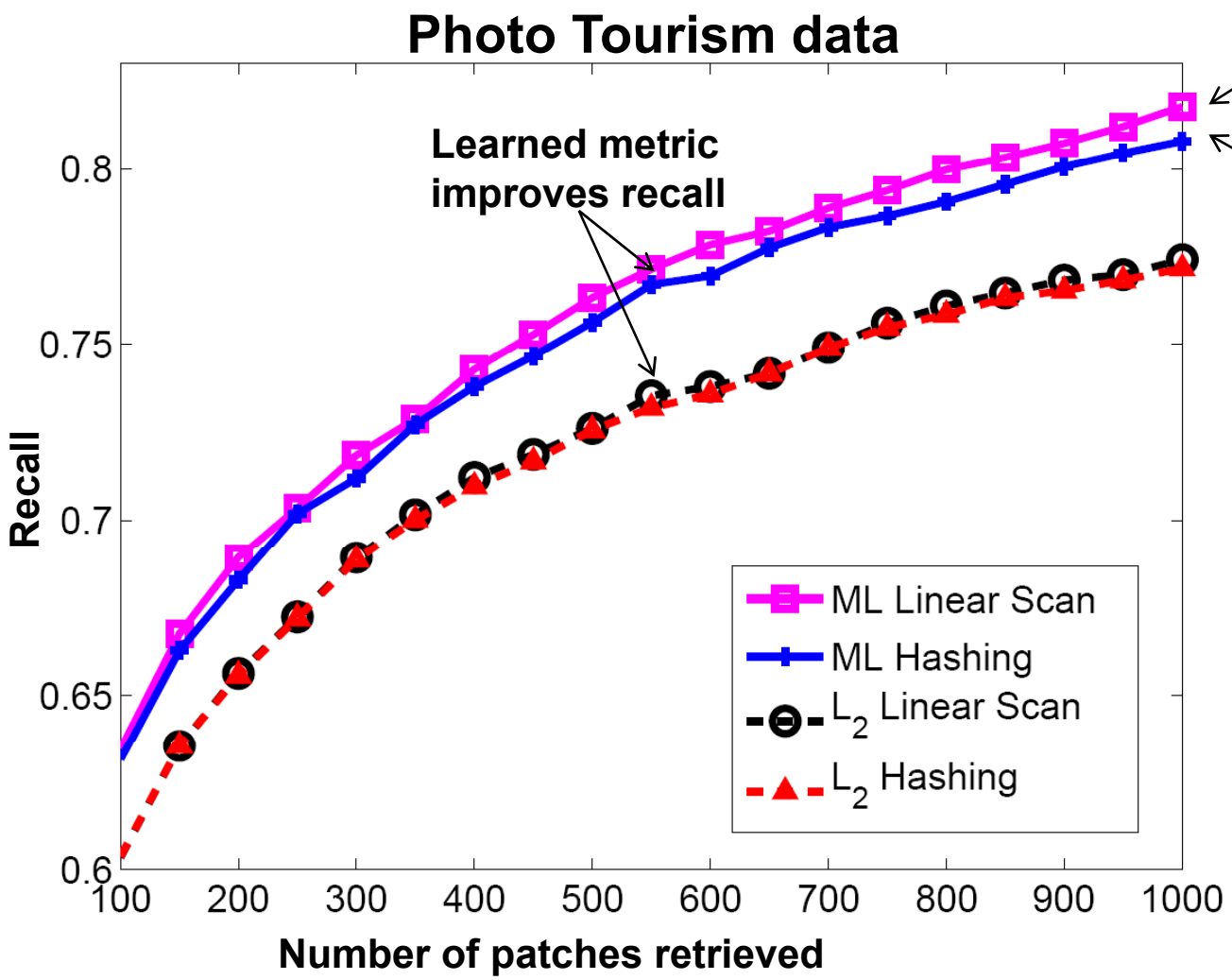
Results: patch indexing

O(10^5) patches

- Photo Tourism data: goal is to match patches that correspond to same point on 3d object
- More accurate matches → better reconstruction
- Huge search pool

[Photo Tourism data provided by Snavely, Seitz, Szeliski, Winder & Brown]

Results: patch indexing



Our technique maintains accuracy while searching less than 1% of the database.

Summary

- Content-based queries demand fast search algorithms for useful image metrics.
- Contributions:
  - Semi-supervised hash functions for class of learned metrics and kernels
  - Theoretical guarantees of accuracy on nearest neighbor searches
  - Validation with pose estimation, object categorization, and patch indexing tasks.

Today

- Demos
- Niebles Presentation from last week [Daphna B.]

- ISM (Leibe) – *Mario Fritz guest lecture*
- Learning Distances (Frome) – [Ashley E.]
- Hashing with Metric Learning – *Brian Kulis guest lecture*
- **Semantic Texton Forests (Shotten)**
- Forests/Ferns for Keypoint Localization (Leptit)
- Small Codes/Large Image Datasets (Torralba)

Semantic Texton Forests for Image Categorization and Segmentation

Jamie Shotton **Matthew Johnson** **Roberto Cipolla**

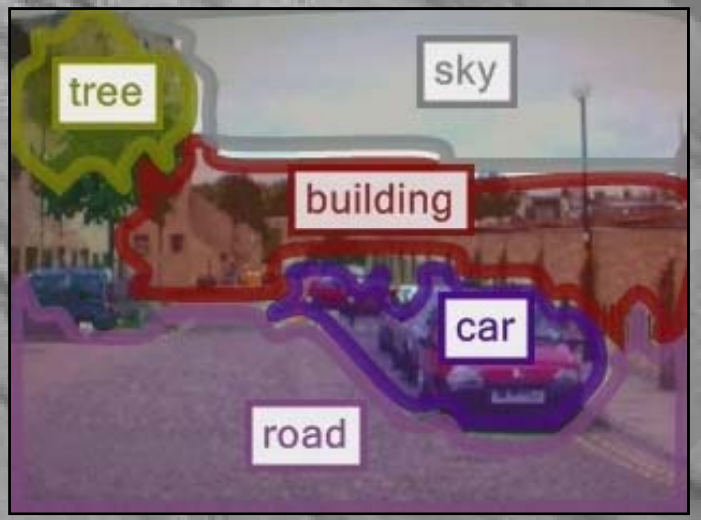
Microsoft Research Cambridge

UNIVERSITY OF CAMBRIDGE

TOSHIBA
Leading Innovation

Introduction

- **Aim – improved visual vocabulary**
  - image categorization
    - *does this image contain cows, trees, etc.*?
  - object segmentation
    - *draw and label the outlines of the cow, grass, etc.*



- **Design goals**
  - **fast and accurate**
  - **learned**
  - **semantic information**

Real-Time Object Segmentation

- **Trained on MSRC 21-class dataset**
  - sheep, grass, sky, etc.
- **Test on unseen images**
- **About 8 fps on 2.6GHz laptop**
  - all on CPU

Live Demo

- **See demo stand Wednesday 8.30am – 12.30am**

Object Recognition Pipeline

extract features

SIFT, filter bank

clustering

k-means

assignment

nearest neighbour

hand-crafted

unsupervised

classification algorithm

SVM, decision forest, boosting

supervised

Object Recognition Pipeline

STF

clustering ‘semantic textons’

+

local classification

Semantic Texton Forest (STF)

- decision forest for both clustering & classification
- tree nodes have learned object category associations

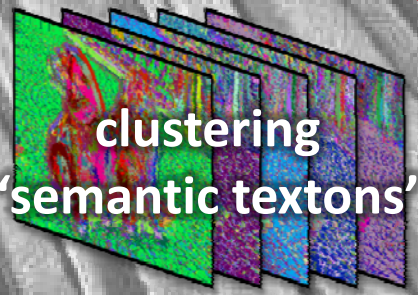
classification algorithm

SVM, decision forest, boosting

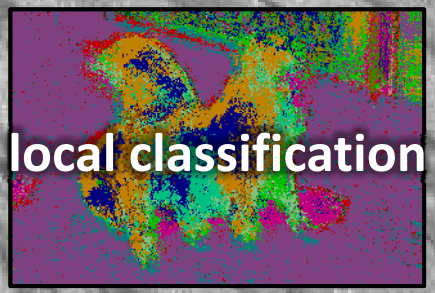
} supervised

Object Recognition Pipeline

STF



+



SVM

SF

building
dog
road

image categorization

object segmentation

Semantic Texton Forest (STF)

- decision forest for both clustering & classification
- tree nodes have learned object category associations

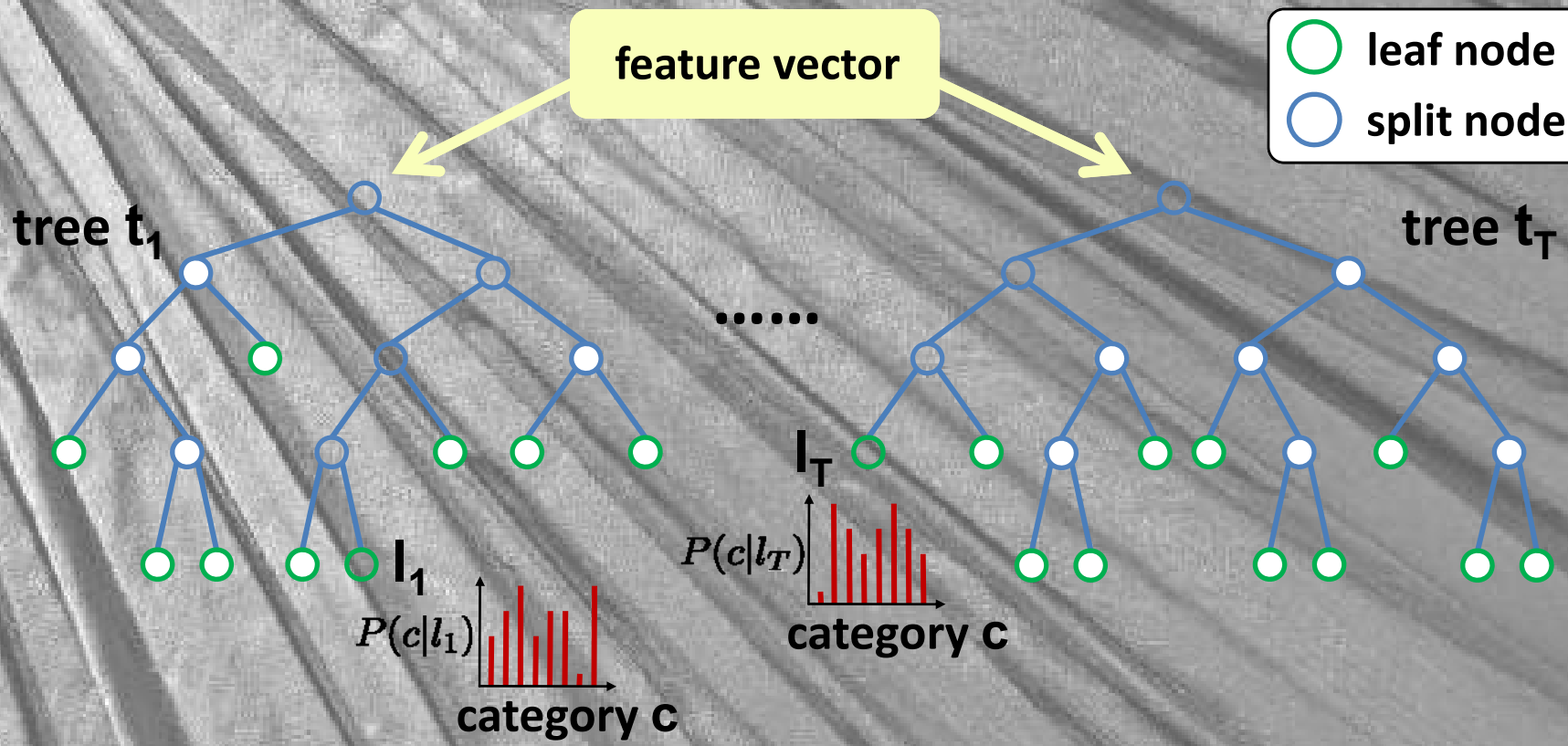
Support Vector Machine (SVM)

- pyramid match kernel in learned tree hierarchies

Segmentation Forest (SF)

- second decision forest
- features use layout & context
- semantic context

Decision Forests Review



- **Forest is ensemble of T trees**
  - classification is $P(c|L) = \sum_{t=1}^{T} P(c|l_t)$

[Amit & Geman 97]
[Lepetit *et al.* 06]

Decision Forests Review [Geurts *et al.* 06]

- **Recursively split training data I_n on feature function** f

left split

$$I_l = \{i \in I_n \mid f_i < t\}$$

right split

$$I_r = I_n \setminus I_l$$

threshold

function value at example i

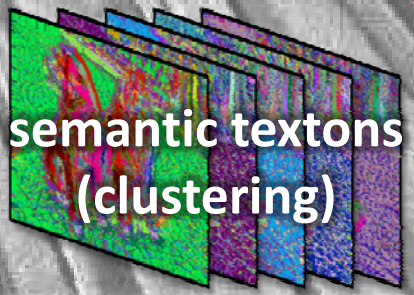
- **Try several random feature functions** f **and thresholds** t
  - maximize gain in information:

$$\Delta E = -\frac{|I_l|}{|I_n|} E(I_l) - \frac{|I_r|}{|I_n|} E(I_r)$$

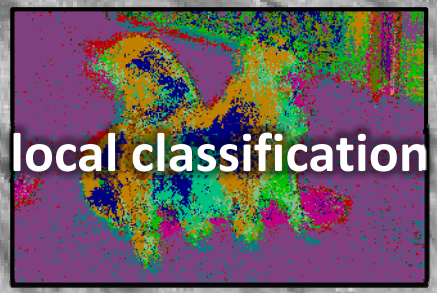
- **Randomization for generalization**
  - training examples
  - feature functions f
  - thresholds t

Main Contributions

STF



+



SVM SF

building
dog
road

image categorization

object segmentation

Semantic Texton Forest (STF)

- decision forest for both clustering & classification
- tree nodes have learned object category associations

Support Vector Machine (SVM)

- pyramid match kernel in learned tree hierarchies

Segmentation Forest (SF)

- second decision forest
- features use layout & context

Textons & Visual Words

- **Textons [Julesz 81]**
  - computed densely
  - clustered filter-bank responses
  - used for object recognition

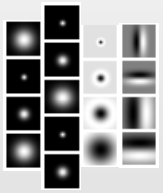
e.g. references
[Malik 01] [Varma 05]
[Winn 05] [Shotton 06]

- **Visual words**
  - usually computed sparsely
  - clustered descriptors
  - used for object recognition

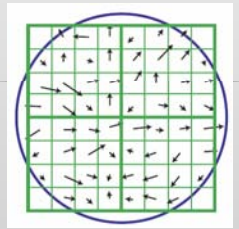
[Mikolacjzyk 04]
[Lowe 04]
[Sivic 03] [Csurka 04]

extract features

filter bank



local descriptors
e.g. [Lowe 04]



clu…ent

…neighbour

Expensive!

Semantic Texton Forests (STF)

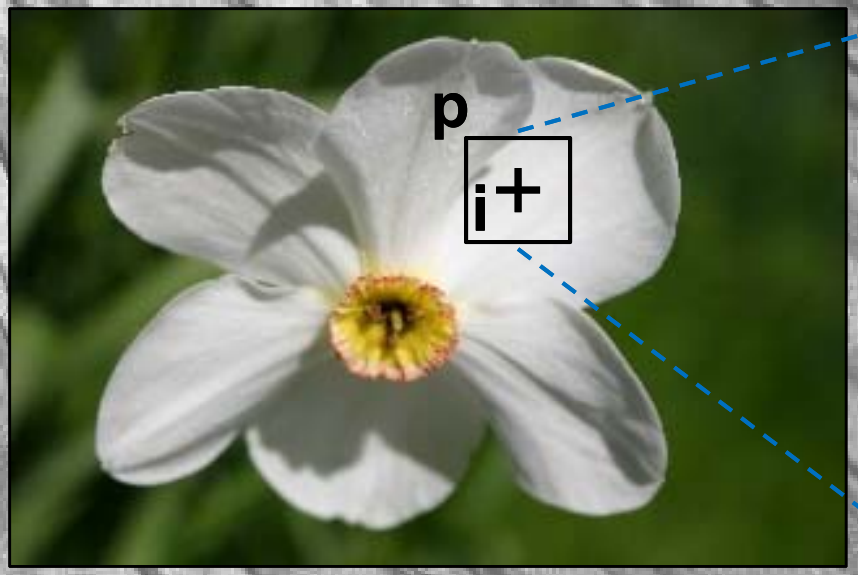
- **A STF is**
  - a decision forest applied at each image pixel
  - simple pixel-based features

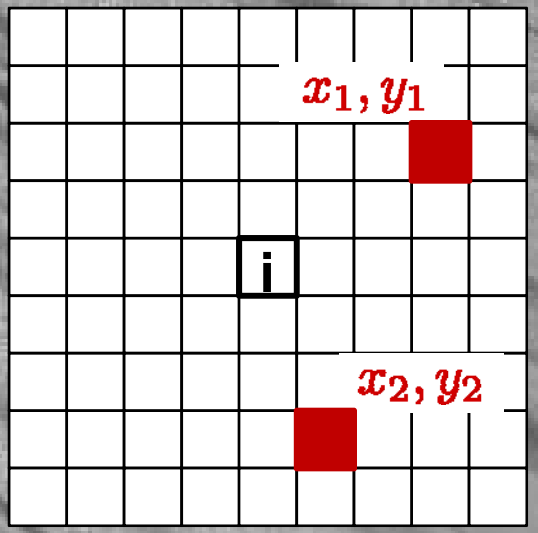
- **How is this new?**
  - no descriptors or filter-banks
  - decision forest
    - **fast clustering** & **assignment**
    - **local classification**

learned semantic information

Very Fast

Image Patch Features





Pixel i gives patch p

(21x21 pixels in experiments)

$$f(\mathbf{p}) = p_{x_1,y_1,c_1}$$
$$f(\mathbf{p}) = p_{x_1,y_1,c_1} + p_{x_2,y_2,c_2}$$
$$f(\mathbf{p}) = p_{x_1,y_1,c_1} - p_{x_2,y_2,c_2}$$
$$f(\mathbf{p}) = |\, p_{x_1,y_1,c_1} - p_{x_2,y_2,c_2} \,|$$

f(p) $\gtrless$ learned threshold

tree split function

Leaf Node Visualization

- **Average of all training patches at each leaf node**

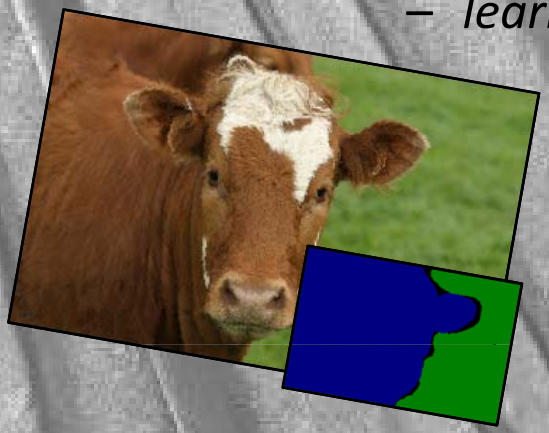
STF Training Examples

- **Supervised training**



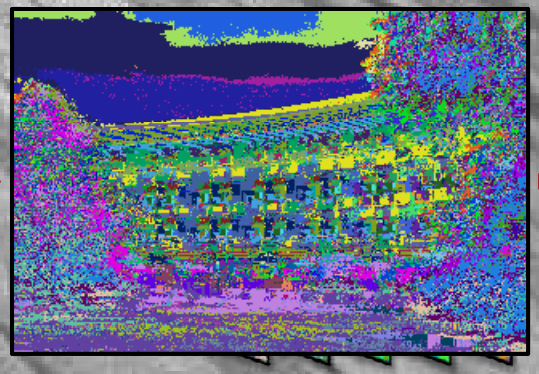
(GT colors ⇔ categories)

- **Regular grid**
- **Random transformations**
  - *learn* invariances [Lepetit *et al.* 06]

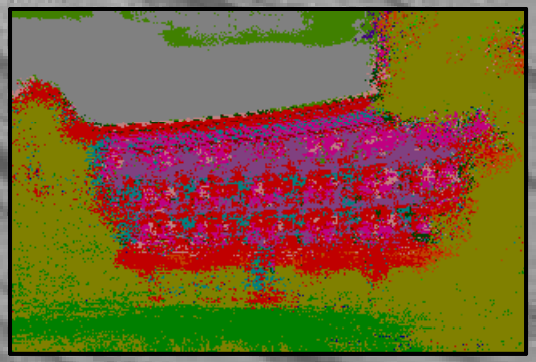


Semantic Textons & Local Classification

test image



semantic textons
(color ⇔ leaf node index)

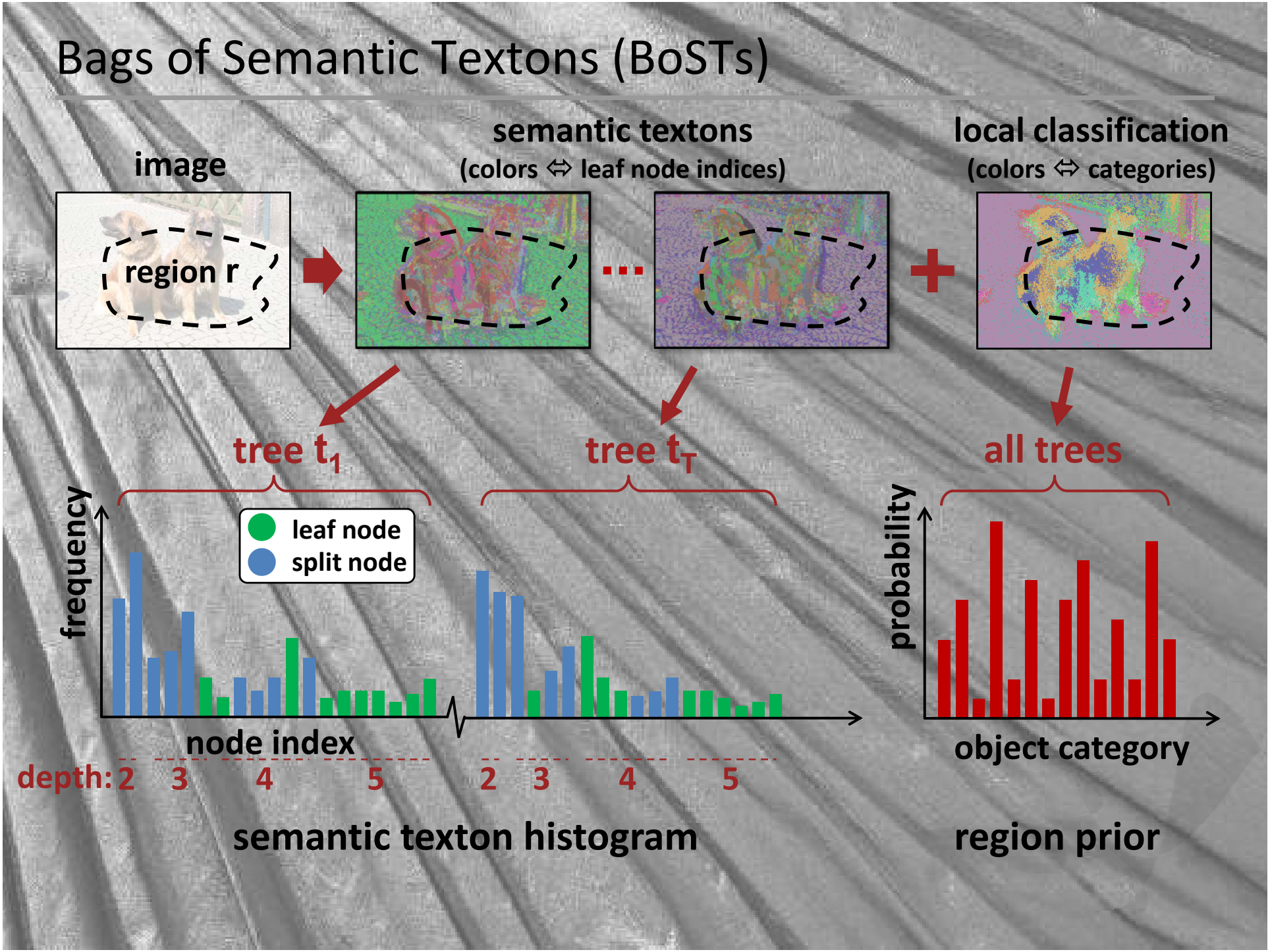


local classification
(color ⇔ most likely category)

ground truth
(for reference)

comparable

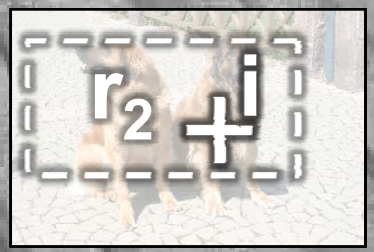
Bags of Semantic Textons (BoSTs)



Choice of Regions for BoSTs

- **Image categorization**
  - region r = whole image

- **Object segmentation**
  - many image regions r



....

Main Contributions

STF

+

SVM

SF

building
dog
road

image categorization

object segmentation

Semantic Texton Forest (STF)

- decision forest for both clustering & classification
- tree nodes have learned object category associations

Support Vector Machine (SVM)

- **pyramid match kernel in learned tree hierarchies**

More on PMK in two weeks…

Segmentation Forest (SF)

- second decision forest
- features use layout & context

Main Contributions

STF

+

SVM

SF

object segmentation

Segmentation Forest (SF)

- second decision forest
- features use layout & context

Semantic Texton Forest (STF)

- decision forest for both clustering & classification
- tree nodes have learned object category associations

Support Vector Machine (SVM)

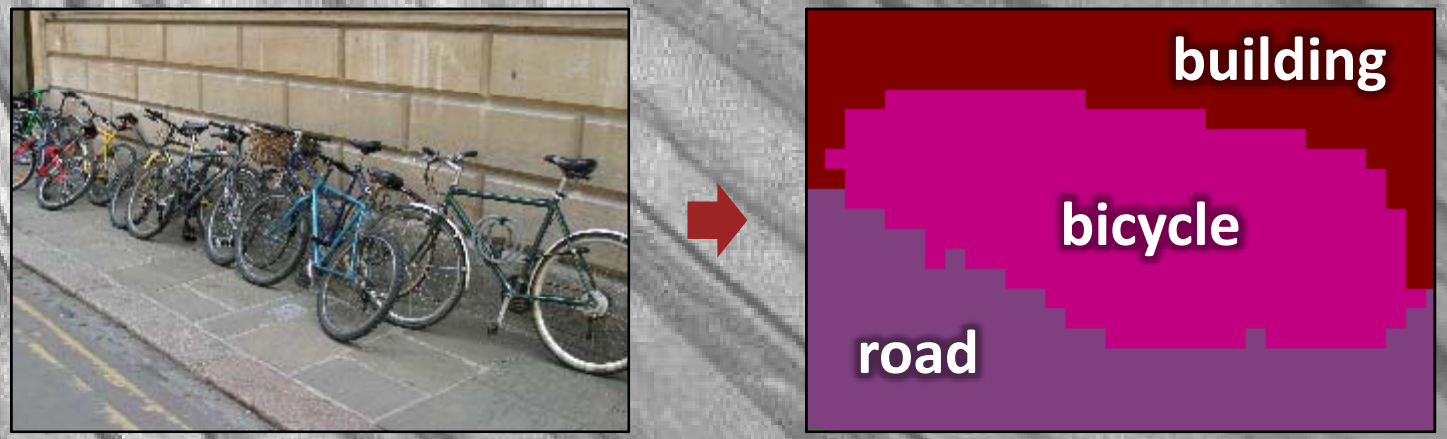
- pyramid match kernel in learned tree hierarchies

building
dog
road

image categorization

Segmentation Forest

- **Object segmentation**

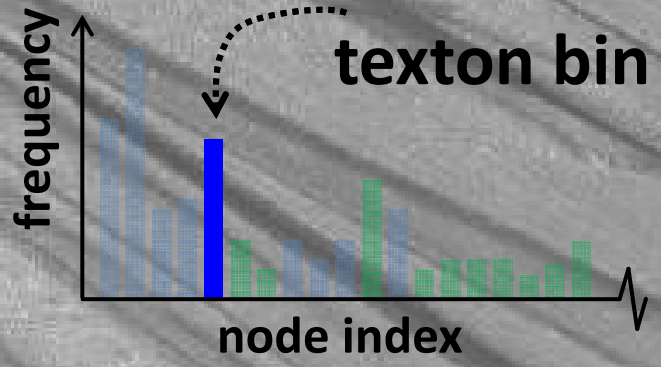


- **Adapt TextonBoost [Shotton *et al.* 06]**
  - boosted classifier → randomized decision forest
    textons → semantic textons + region priors
  - no conditional random field

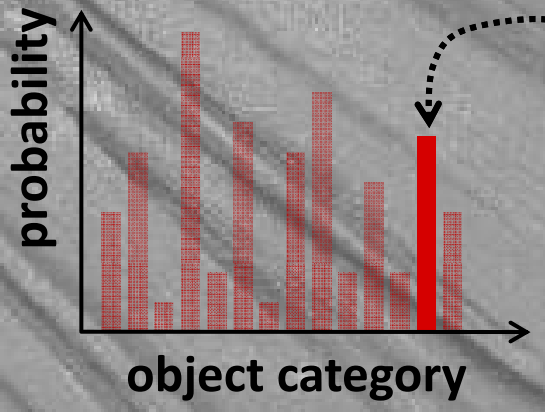
Features in Segmentation Forest

offset rectangle r

+



or



bin count $\geq$ learned threshold ?

tree split function

Features in Segmentation Forest

- **Learning the randomized forest**
  - regular grid
  - discriminative pairs of region r and BoST bin
- **Properties**
  - features exploit texture, layout, context [Shotton *et al.* 06]
  - region prior allows *semantic* context
- **Efficient calculation**
  - compute bins only as required
  - use integral images [Viola & Jones 01]
  - sub-sample integral images

Image-Level Prior (ILP)

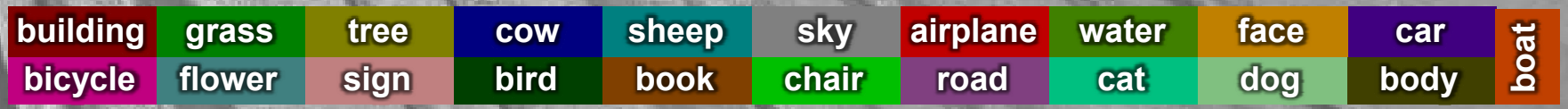
- **Combine**
  - image categorization (SVM with learned PMK)
  - object segmentation (decision forest)

image categorization posterior → **prior for segmentation**

$$P'(c|i) = \underbrace{P(c|i)}_{\text{SF}} \times \underbrace{P(c)^{\overbrace{\alpha}^{\text{weighting}}}}_{\text{ILP}}$$

See also [Verbeek *et al.* NIPS 07]

MSRC Segmentation Results



Conclusions

- **Semantic texton forests**
  - effective alternative to textons for recognition
- **Image-level prior**
  - significantly improves segmentation
  - uses identical image features
- **Memory**
  - high memory requirements for training
- **Efficiency**
  - very fast on CPU
  - GPU friendly [Sharp, ECCV 08]

Today

- Demos
- Niebles Presentation from last week [Daphna B.]

- ISM (Leibe) – *Mario Fritz guest lecture*
- Learning Distances (Frome) – [Ashley E.]
- Hashing with Metric Learning – *Brian Kulis guest lecture*
- Semantic Texton Forests (Shotten)
- **Forests/Ferns for Keypoint Localization (Leptit)**
- Small Codes/Large Image Datasets (Torralba)

Ferns for 3D Detection

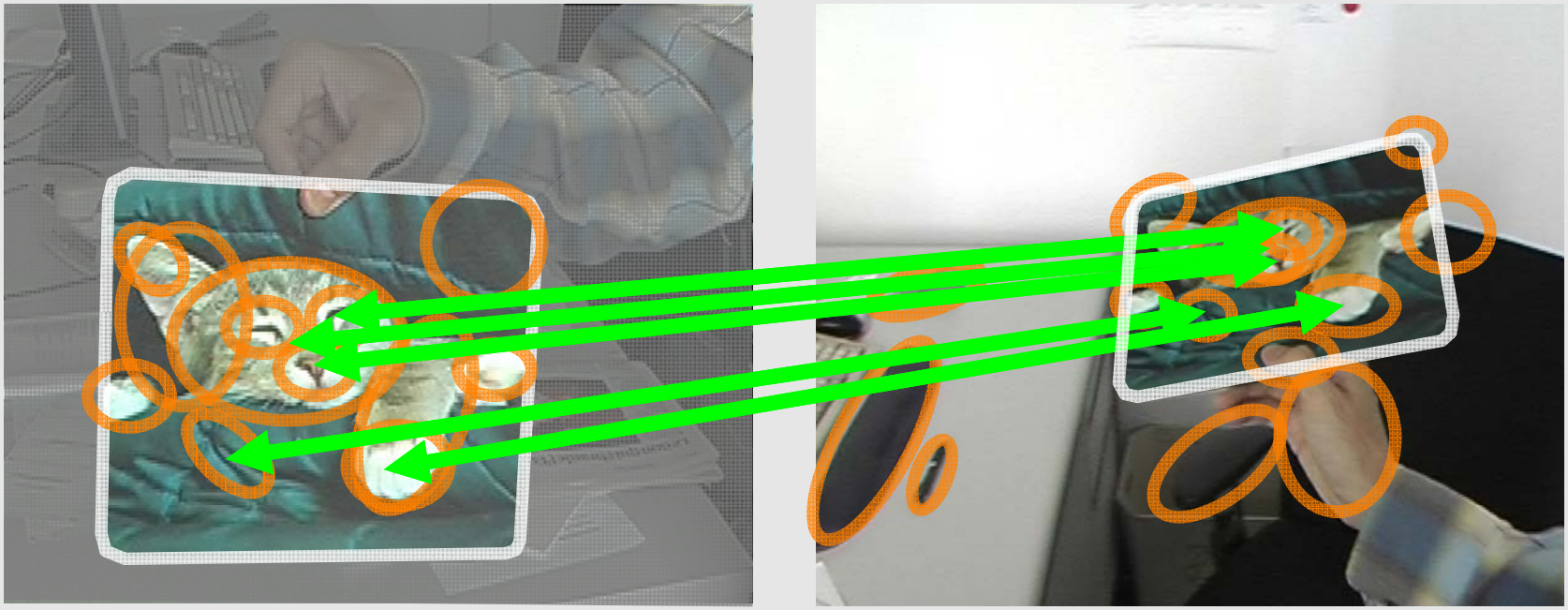
Vincent Lepetit
CVLab - EPFL

3D Object Detection

Keypoint detection (Harris, extrema of Laplacian, affine regions,...);

Keypoint recognition (descriptor matching or classification);

Robust pose estimation (RANSAC+P3P, ...).



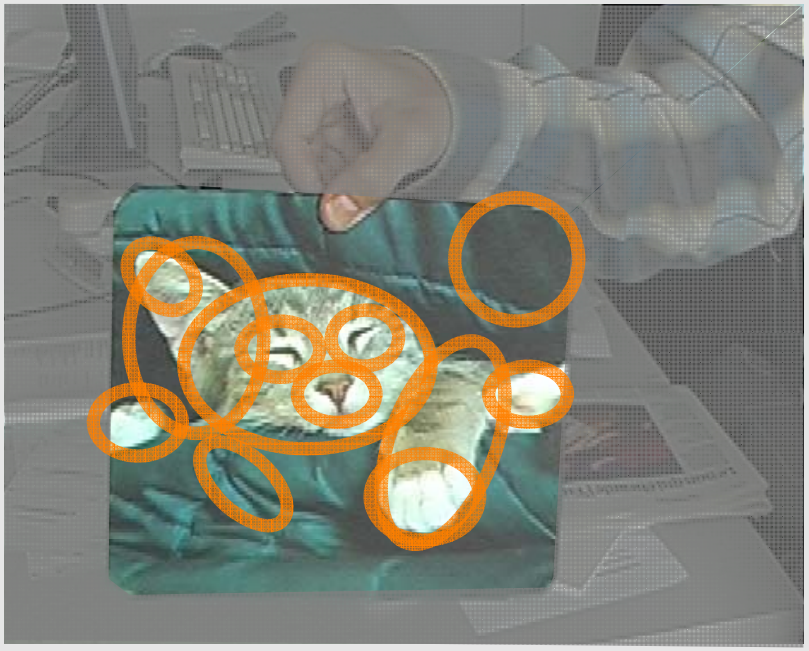
Registered image(s) of the object to detect

Input image

Standard Approach to Keypoint-Based Object Detection

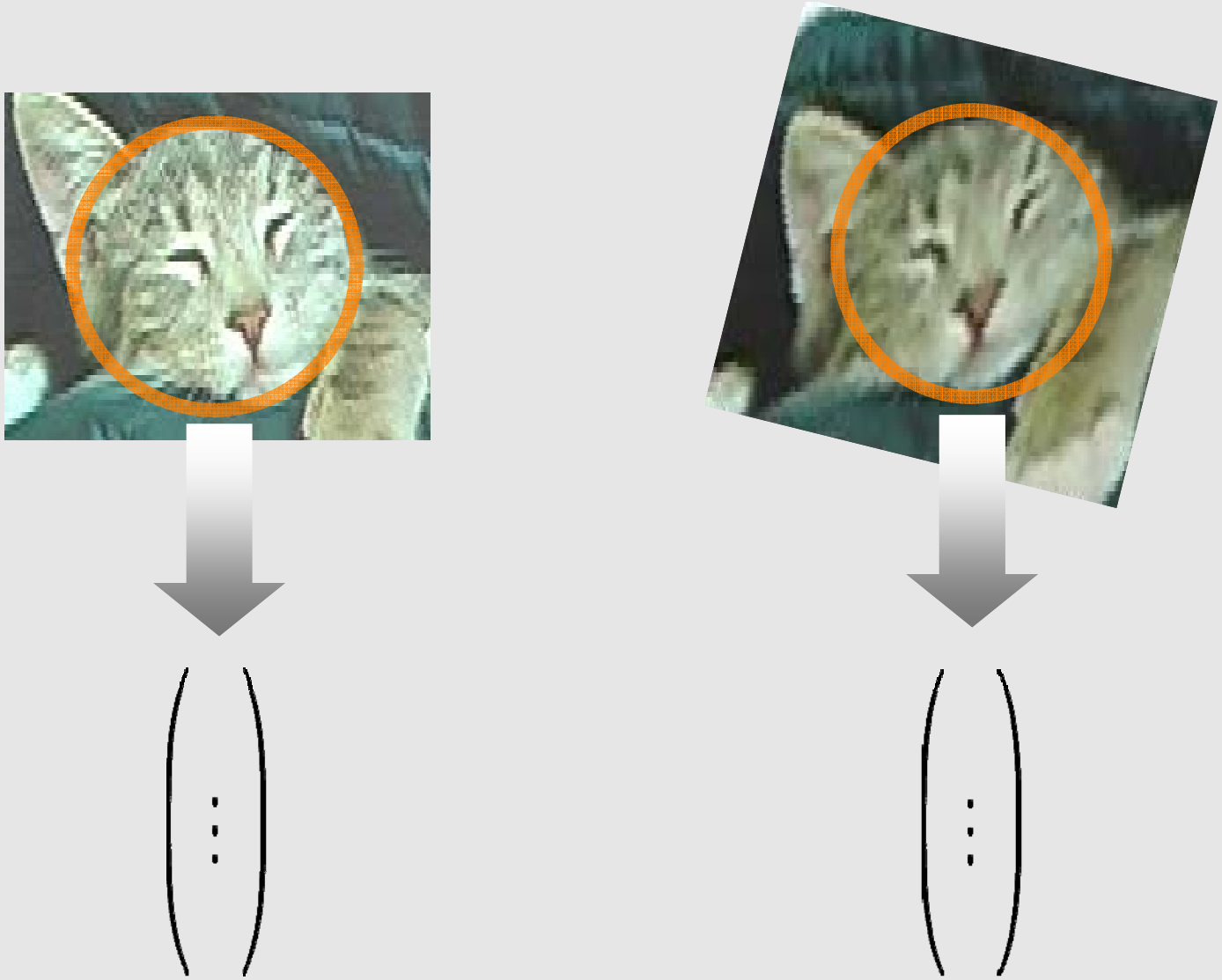
Standard Approach

Step 1: Detection invariant to scale and rotation, or perspective transformation

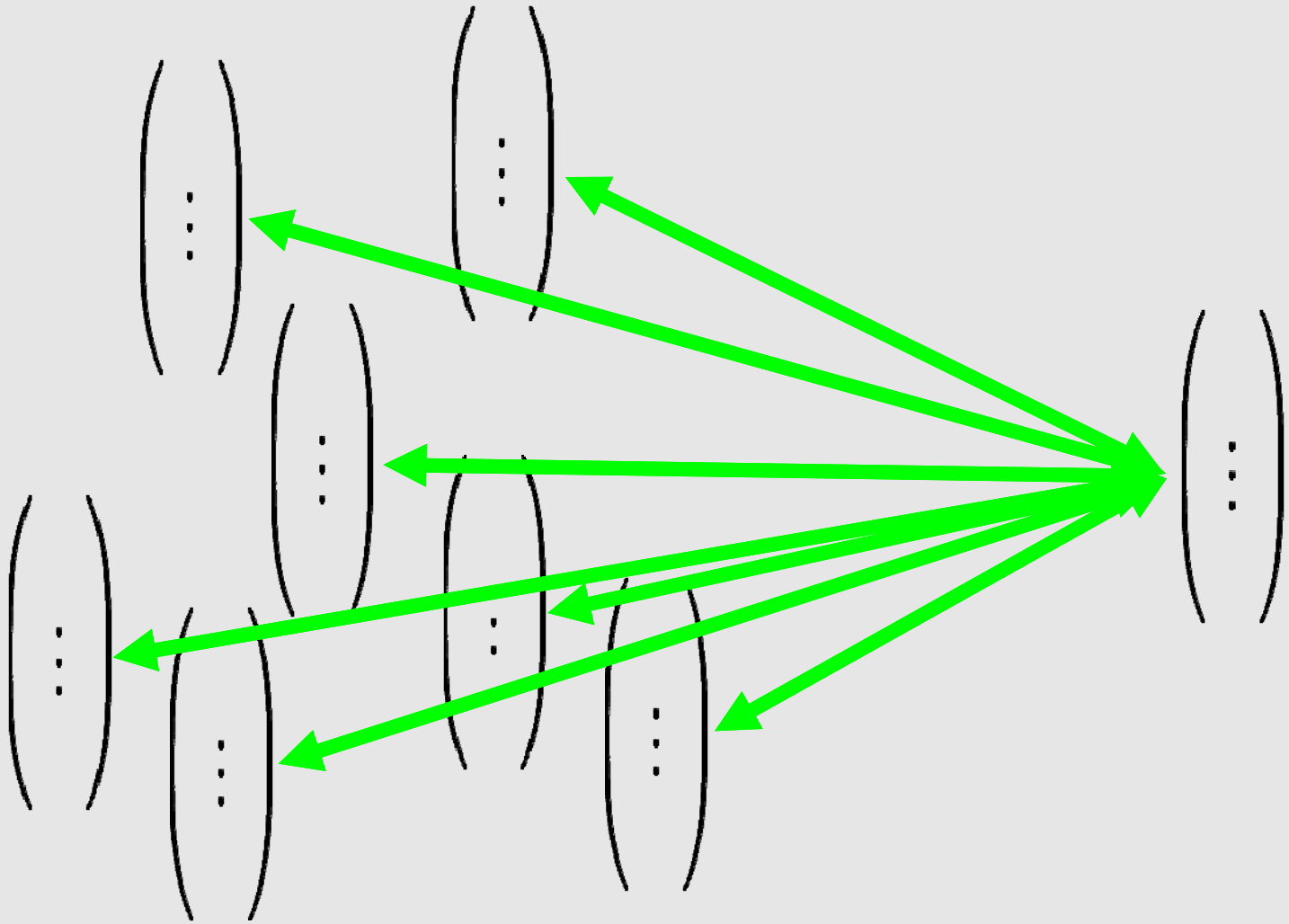


Standard Approach
Step 2: Patch rectification

Standard Approach
Step 3: Build description vector

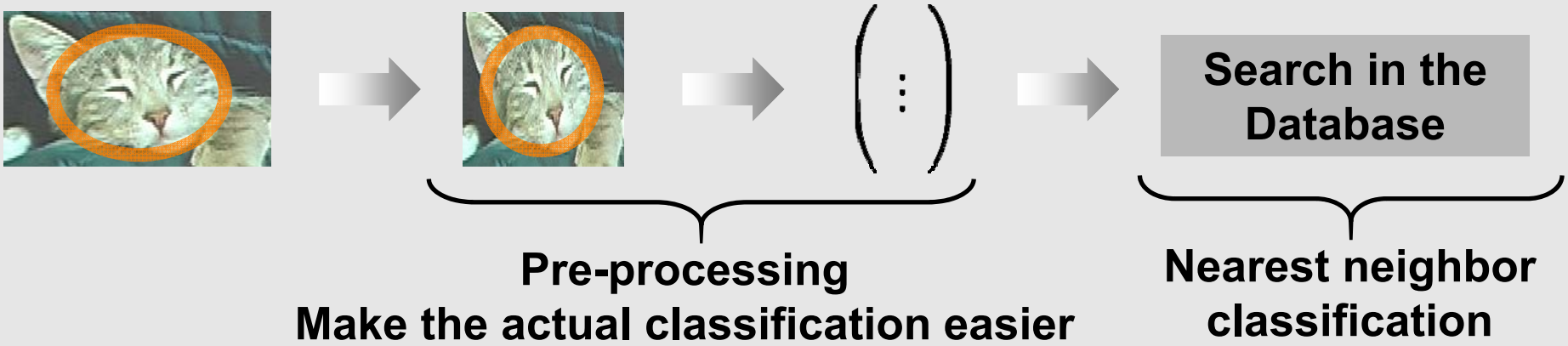


Standard Approach
Step 4: Match description vectors

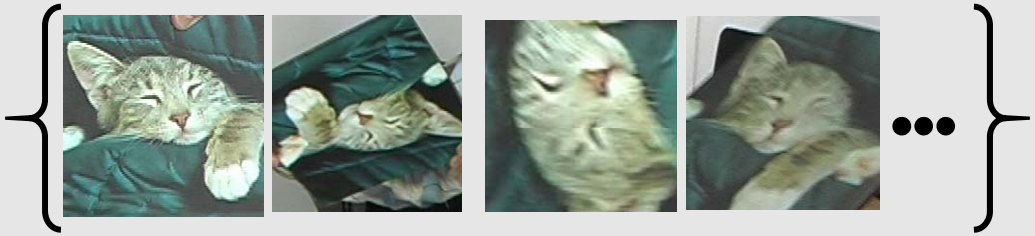


Keypoint Recognition

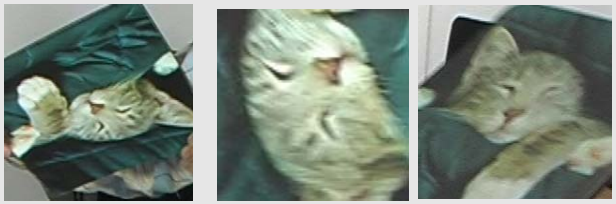
The standard approach is a particular case of classification:



One class per keypoint: the set of the keypoint's possible appearances under various perspective, lighting, noise...



Training phase



Classifier

Used at run-time to recognize the keypoints

We are looking for $\arg\max_i P(C = c_i \mid \textbf{patch})$

If patch can be represented by a set of image features $\{f_i\}$:

$$P(C = c_i \mid \textbf{patch}) = P(C = c_i \mid f_1, f_2, \ldots f_n, f_{n+1}, \ldots \ldots f_N)$$

which is proportional to

$$P(f_1, f_2, \cdots f_n, f_{n+1}, \cdots \cdots f_N \mid C = c_i)$$

but complete representation of the joint distribution infeasible.

Naive Bayesian ignores the correlation:

$$\approx \prod_j P(f_j \mid C = c_i)$$

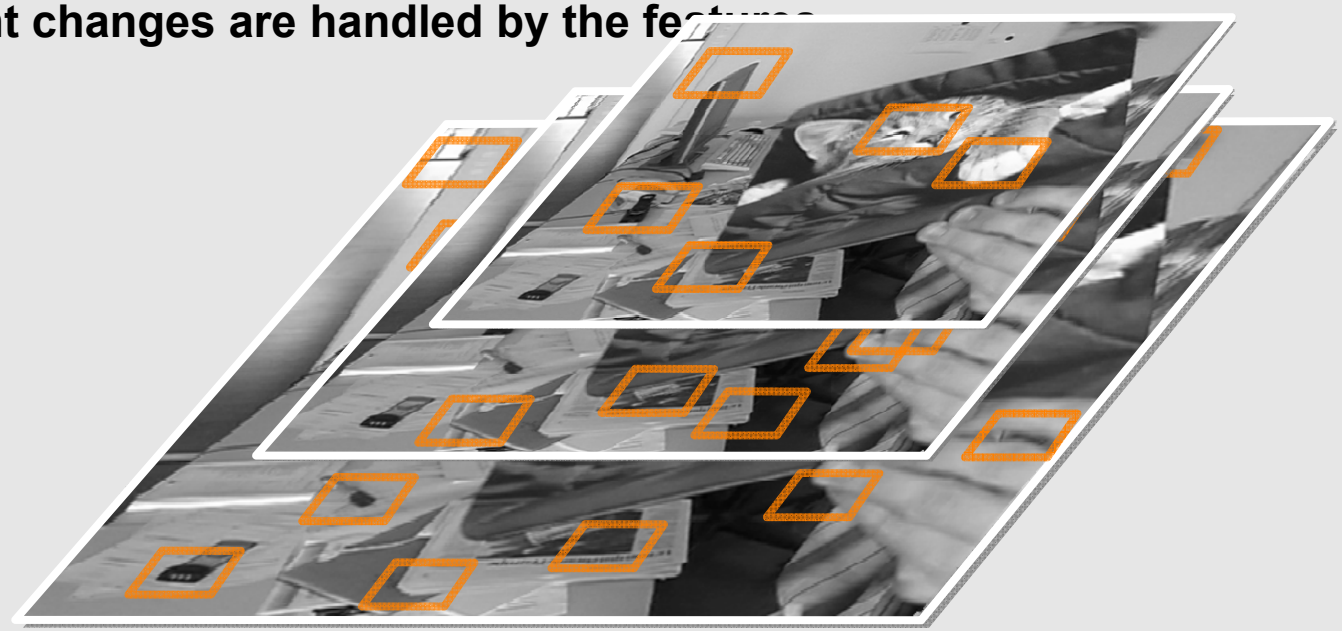
Compromise:

$$\approx P(f_1, f_2, \cdots f_n \mid C = c_i) \times P(f_{n+1}, \cdots f_{2n} \mid C = c_i) \times \cdots$$

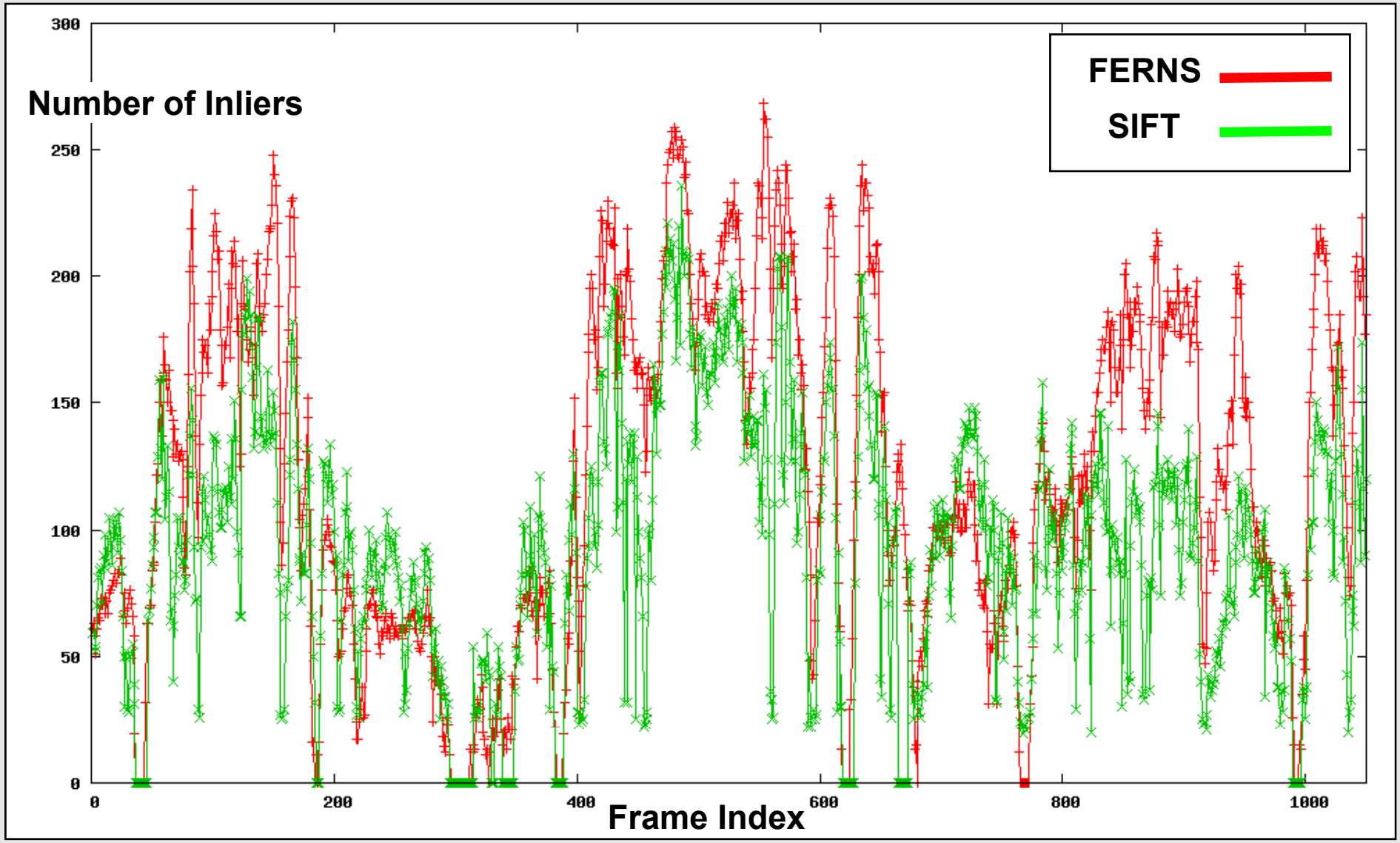
Handling rotation, scale, perspective, light changes

We detect keypoints as extrema of Laplacian in several downscaled versions of the input image.

- **Rotation and perspective are entirely handled by the classifier;**
- **Scale changes over an octave are handled by the classifier; the keypoint detector and the classifier work in tandem to recognize the keypoints over large scale changes;**
- **Light changes are handled by the fe**



Comparison with SIFT
Recognition rate



Comparison with SIFT
Computation time

- **SIFT: 1 ms to compute the descriptor of a keypoint (without including convolution);**

- **FERNS: 13.5 micro-second to classify one keypoint into 200 classes.**

Keypoint Recognition in Ten Lines of Code

```
 1: for(int i = 0; i < H; i++) P[i ] = 0.;
 2: for(int k = 0; k < M; k++) {
 3:   int index = 0, * d = D + k * 2 * S;
 4:   for(int j = 0; j < S; j++) {
 5:     index <<= 1;
 6:     if (*(K + d[0]) < *(K + d[1]))
 7:       index++;
 8:     d += 2;
      }
 9:   p = PF + k * shift2 + index * shift1;
10:   for(int i = 0; i < H; i++) P[i] += p[i];
     }
```

Very simple to implement;
No need for orientation nor perspective correction;
(Almost) no parameters to tune;
Very fast.

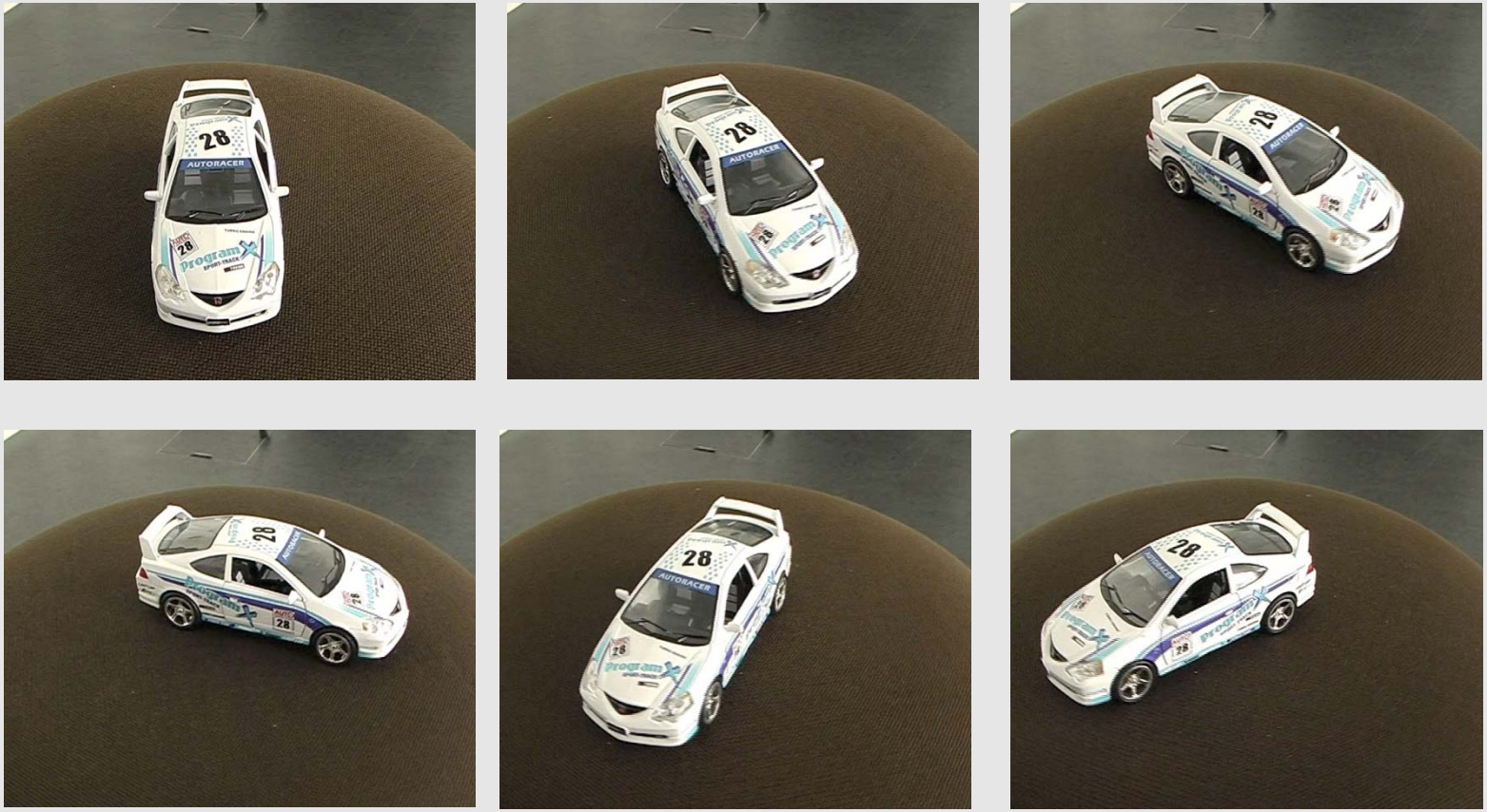
Ferns Tuning

- The number of ferns, and
- The number of tests per ferns

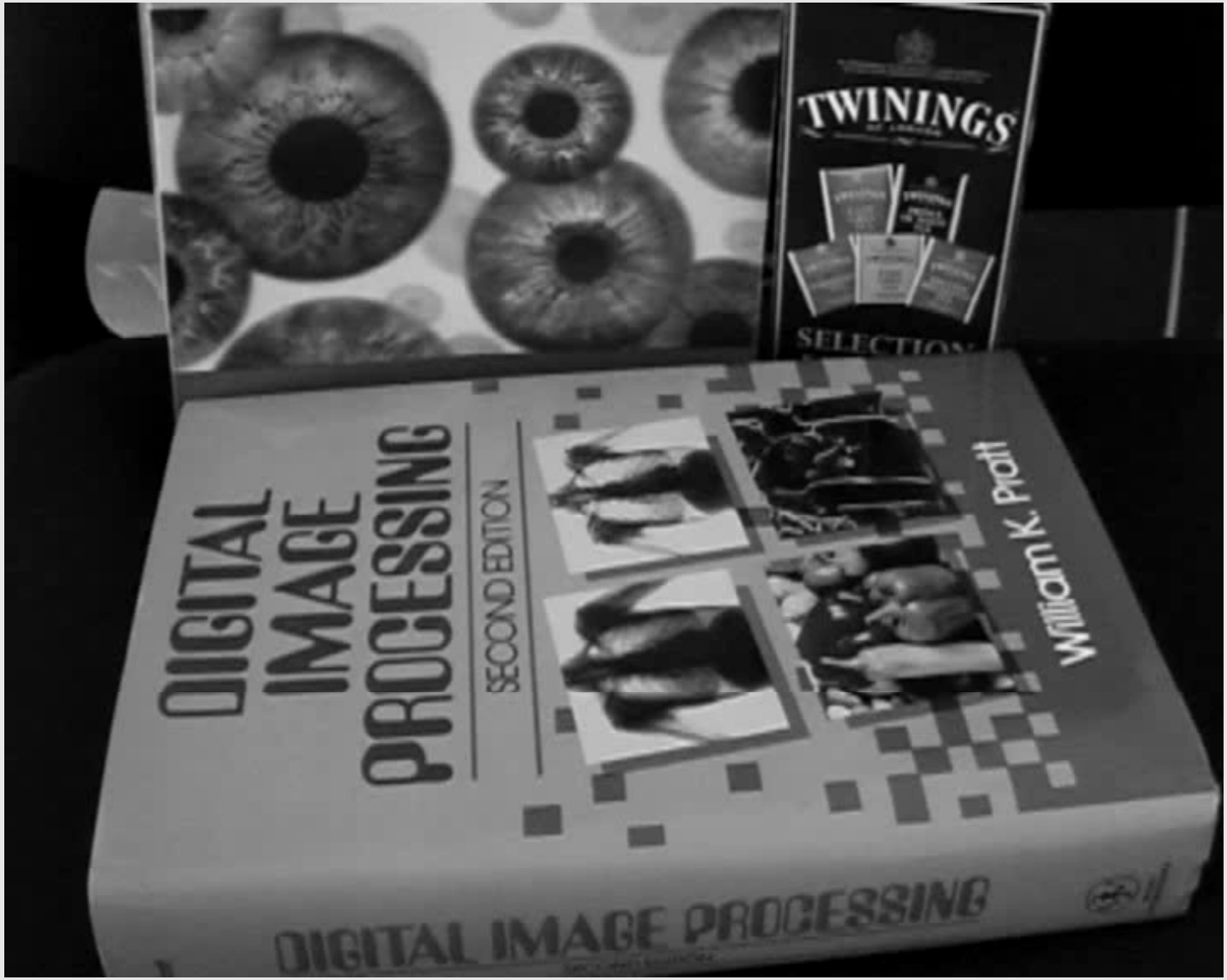
can be tuned to adapt to the hardware in terms of CPU power and memory size.

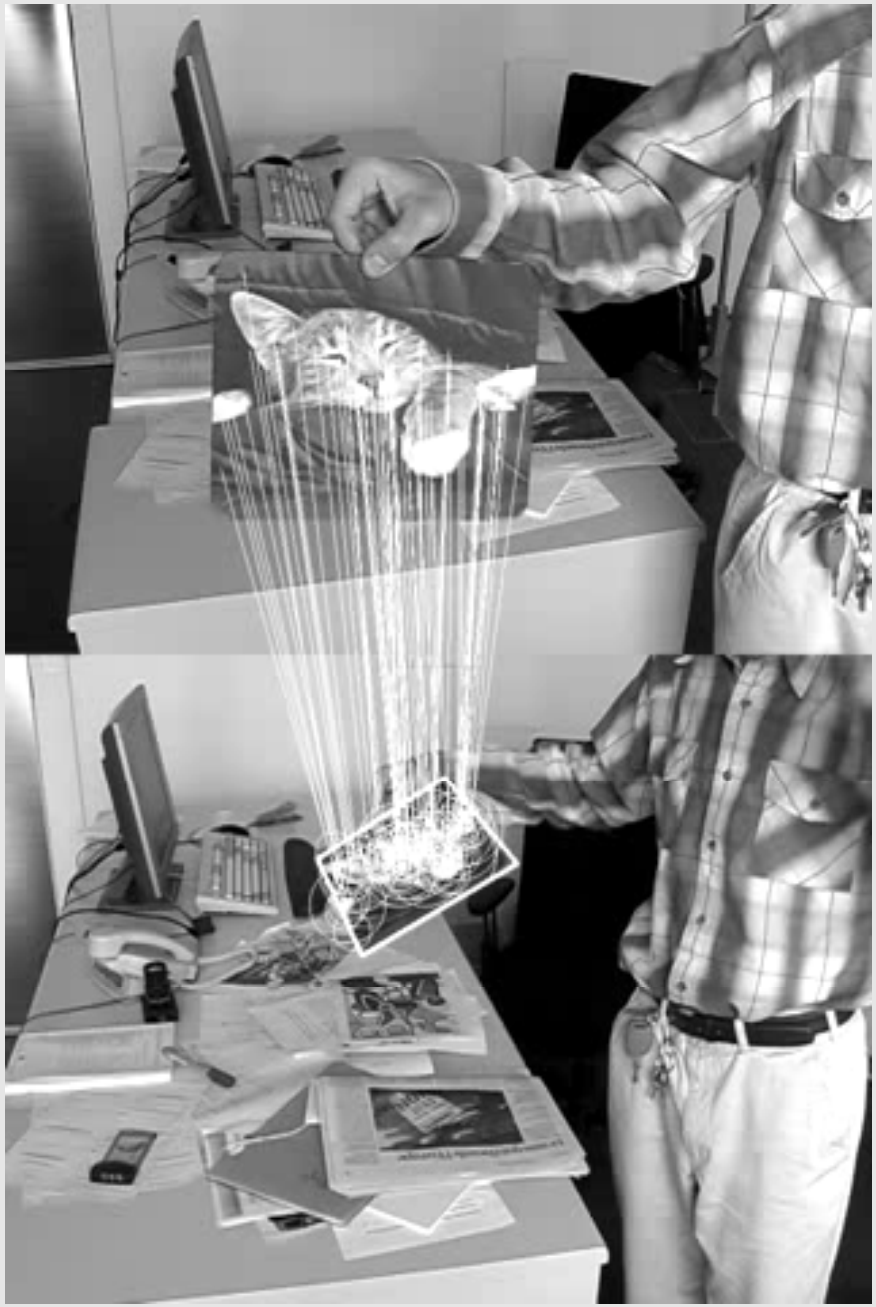
Feature Harvesting

Estimate the posterior probabilities from a training video sequence:



Test Sequence





Related Work:

A. Bosch, A. Zisserman, and X. Munoz. Image classification using random forests. In International Conference on Computer Vision, 2007.

N.E. Apostoloff and A. Zisserman. Who are you? Real-time person identification. In British Machine Vision Conference, 2007

A. Boffy, Y. Tsin and Y. Genc. Real-Time Feature Matching using Adaptive and Spatially Distributed Classification Trees. In British Machine Vision Conference, 2007.

Y. Tsin, Y. Genc, Y. Zhu, and V. Ramesh. Learn To Track Edges. In International Conference on Computer Vision, 2007.

B. Williams, G. Klein, and I. Reid: Real-time SLAM Relocalisation. In International Conference on Computer Vision, 2007.

References:

F. Moreno-Noguer, V. Lepetit, and P. Fua, Accurate Non-Iterative $O(n)$ Solution to the PnP Problem. In Proceedings of International Conference on Computer Vision, 2007.

M. Ozuysal, P. Fua, and V. Lepetit, Fast Keypoint Recognition in Ten Lines of Code. In Proceedings of Conference on Computer Vision and Pattern Recognition, 2007.

M. Ozuysal, V. Lepetit, F. Fleuret, and P. Fua, Feature Harvesting for Tracking-by-Detection. In Proceedings of European Conference on Computer Vision, 2006.

V. Lepetit and P. Fua, Monocular Model-Based 3D Tracking of Rigid Objects: A Survey. Foundations and Trends in Computer Graphics and Vision, Vol. 1, Nr. 1, pp. 1-89, October 2005.

Today

- Demos
- Niebles Presentation from last week [Daphna B.]

- ISM (Leibe) – *Mario Fritz guest lecture*
- Learning Distances (Frome) – [Ashley E.]
- Hashing with Metric Learning – *Brian Kulis guest lecture*
- Semantic Texton Forests (Shotten)
- Forests/Ferns for Keypoint Localization (Leptit)
- **Small Codes/Large Image Datasets (Torralba)**

Dealing with millions of images

Input image

Binary codes for global scene representation

- Short codes allow for storing millions of images
- Efficient search: hamming distance (search millions of images in few microseconds)
- Internet scale experiments: compute nearest neighbors between all images in the internet

512 bits

Binary codes for images

- Want images with similar content to have similar binary codes

- Use Hamming distance between codes
  - Number of bit flips
  - E.g.: **Ham_Dist(10001010,10001110)=1**

    Ham_Dist(10001010,11101110)=3

- Semantic Hashing [Salakhutdinov & Hinton, 2007]
  - Text documents

Slide Rob Fergus

Binary codes for images

- Permits fast lookup via hashing
  - Each code is a memory address
  - Find neighbors by exploring Hamming ball around query address
  - Lookup time depends on radius of ball, NOT on # data points

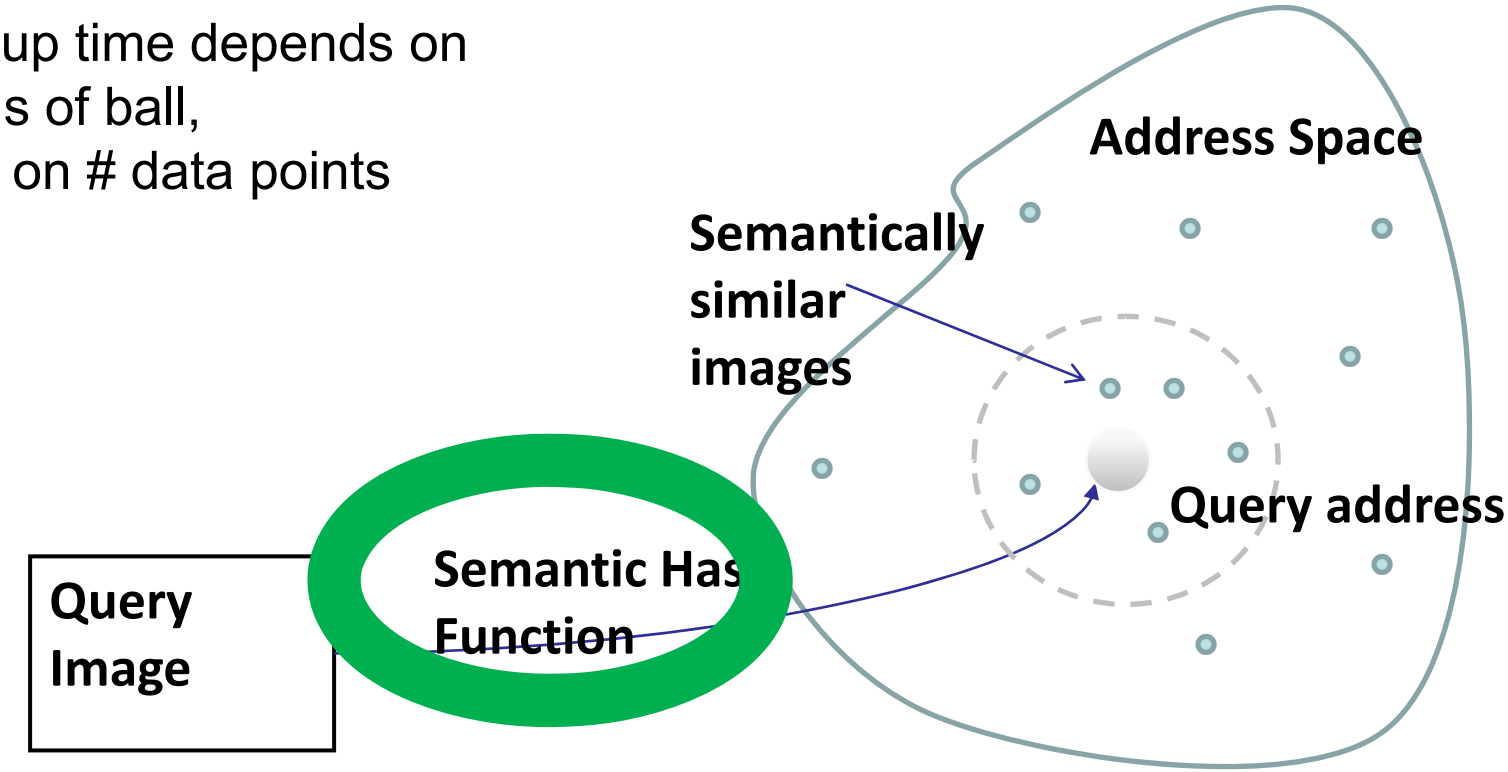


Figure adapted from Salakhutdinov & Hinton '07

Slide Rob Fergus

Compact Binary Codes

- Google has few billion images (10^9)
- Big PC has ~10 Gbytes (10^{11} bits)
- Codes must fit in memory (disk too slow)

→ Budget of 10^2 bits/image

- **1 Megapixel image is 10^7 bits**
- **32x32 color image is 10^4 bits**

→ **Semantic hash function must also reduce dimensionality**

Slide Rob Ferg

How many bits do we need?

Goal: to decide when two images are similar (two images are similar if they contain the same large object classes in similar spatial configurations)

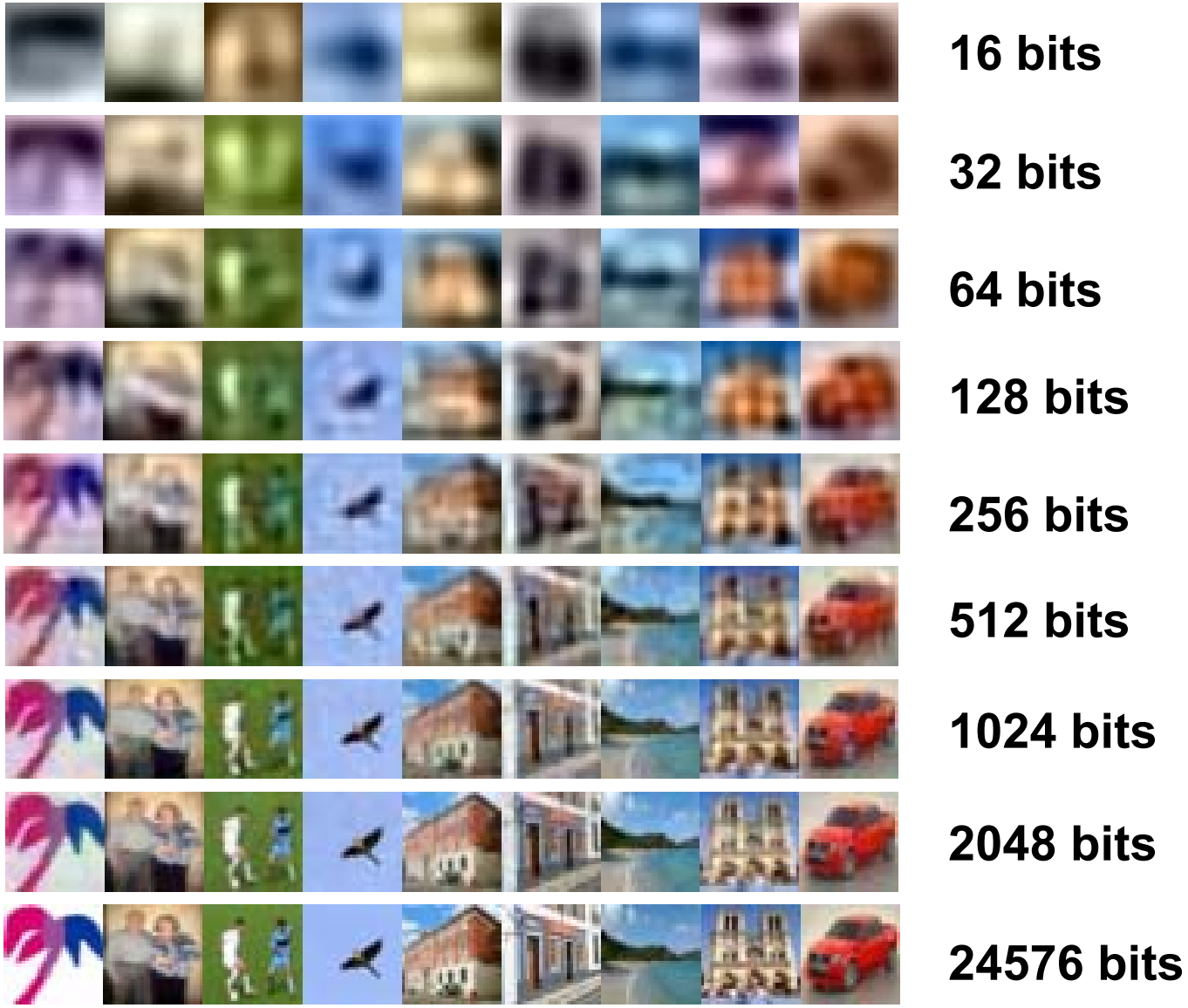
First, let's use some hand-wavy arguments to gain some intuition about how many bits do we need. If we have 1000 object categories, then we would need 1000 bits to tell if an object is present or not (assuming independent objects). However, if we assume that objects are sparse, and that, on average, there are only 5 important object present on an image, then we would need only 50 bits to describe the image content. Adding spatial location can add another 8*5 = 40 bits.

Short image codes

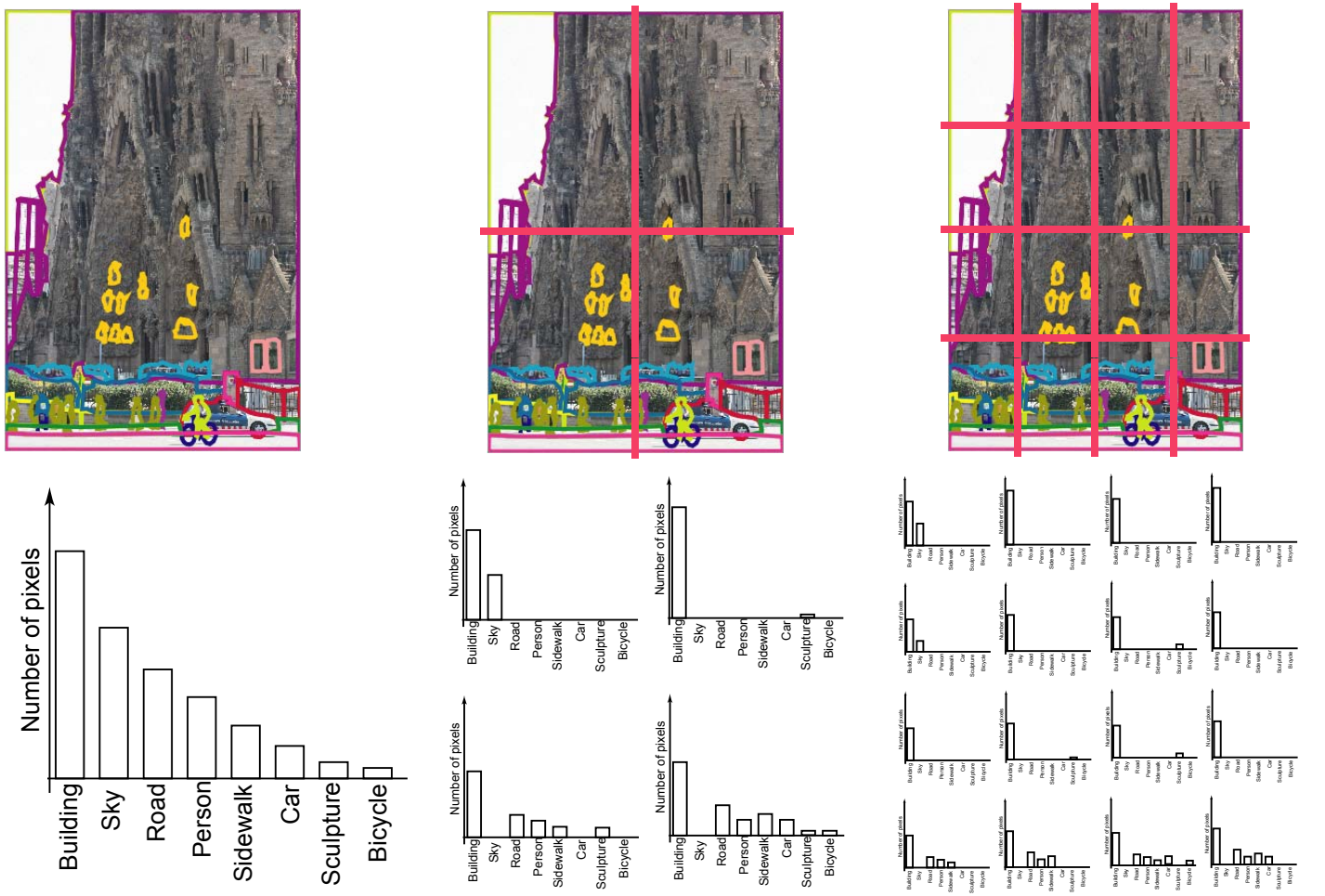
3 mega pixels
8 million bits

1000 pixels
512 bits

How many bits do we need?

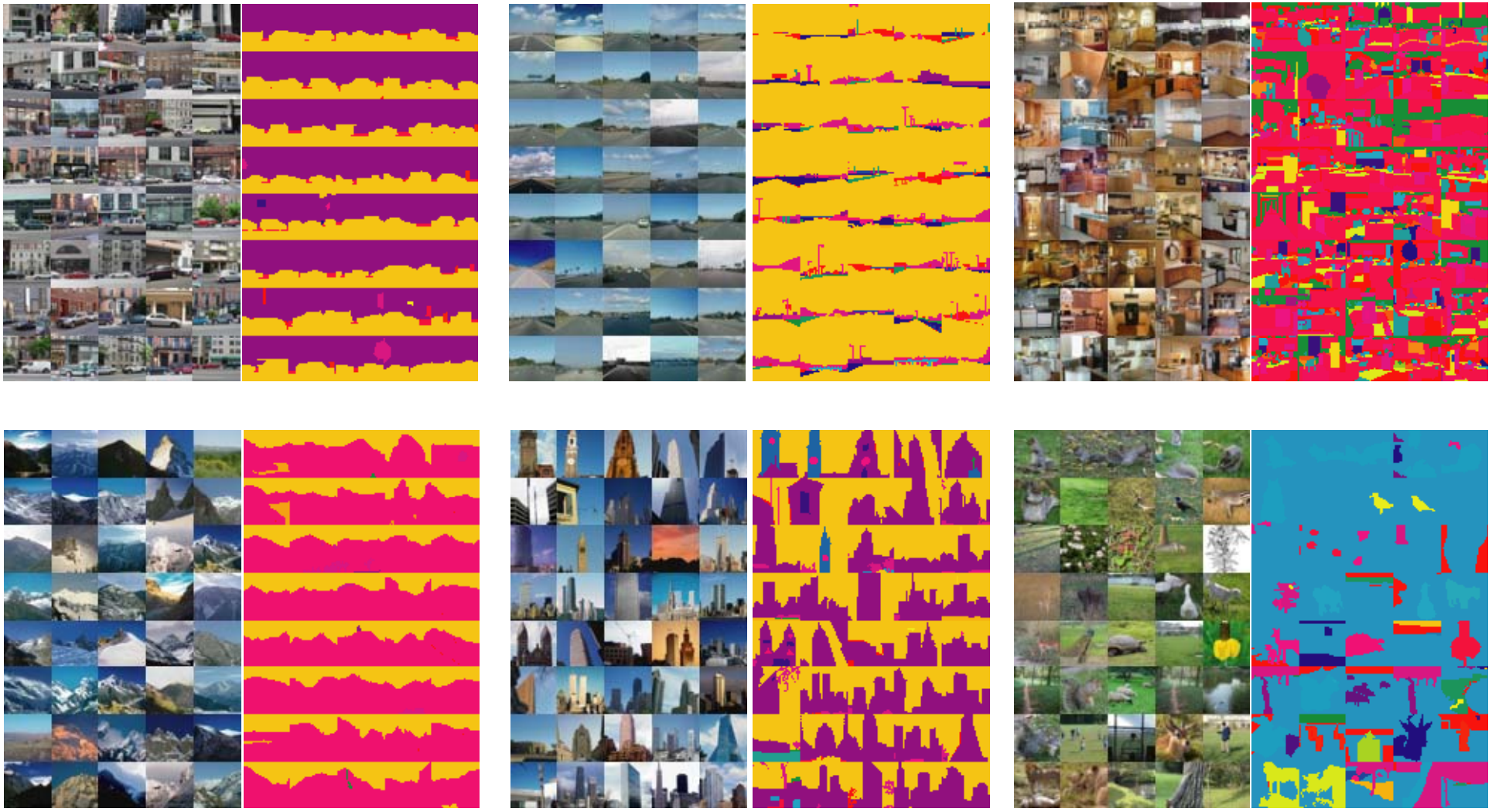


Measuring image similarity with annotated data



$$S(h_1,h_2) = \text{sum}(\min(h_1,h_2))$$

Spatial pyramid matching [Lazebnik06, Grauman07]



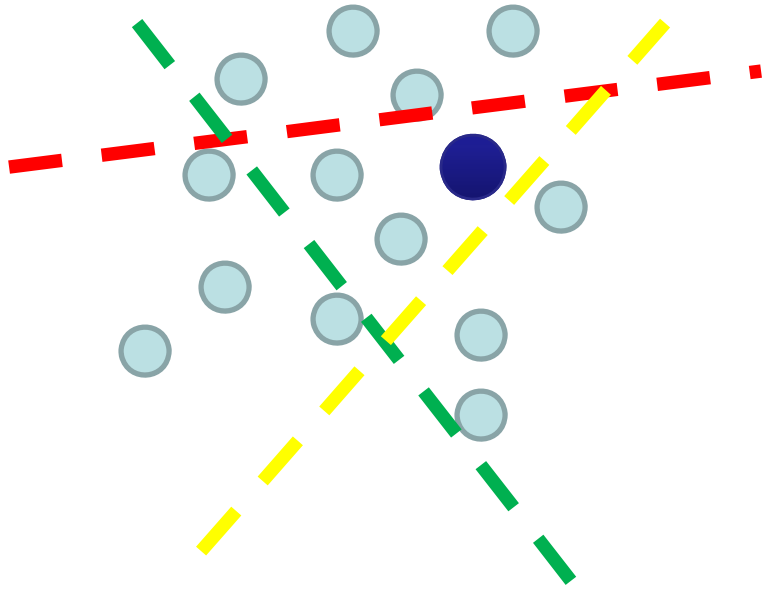
Hashing

We consider the following learning problem - given a database of images $\{x_i\}$ and a distance function D(i, j) we seek a binary feature vector $y_i = f(x_i)$ that preserves the nearest neighbor relationships using a Hamming distance.

Salakhutdinov and Hinton [SIGIR 2007], Shakhnarovich et al [ICCV 2003], Athitsos et al. [ICDE 2008], Grauman et al [CVPR 2007], Nascimentio et al [ACM Smyp. App. Computing 2002], Wang [ICME 2006], Wang [PAMI 2008],

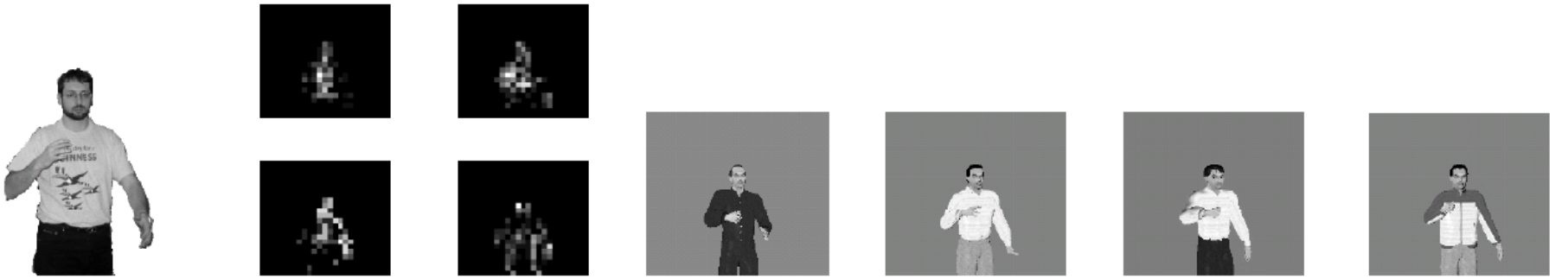
Locality Sensitive Hashing

- Gionis, A. & Indyk, P. & Motwani, R. (1999)
- Take random projections of data
- Quantize each projection with few bits



•For our N bit code:
  – Compute first N PCA components of data
  – Each random projection must be linear combination of the N PCA components

Previously used for Pose estimation



(a) Input (b) A subset of the features (c) 3 top matches in PSII, left to right (d) Robust LWR

Figure 1. Pose estimation with parameter-sensitive hashing and local regression.

Fast Pose Estimation with Parameter Sensitive Hashing. Shakhnarovich, Viola, Darrell. ICCV 2003

Learning hamming distances with boosting

Each image is represented by a binary vector with M bits

$$y = [h_1(x), h_2(x), ..., h_M(x)]$$

- **x = vector of image features**
- **h_i = function with binary output**
- **y = binary vector**

Distance between two images is given by a weighted Hamming distance

$$D(i, j) = \sum_{n=1}^{M} \alpha_n |h_n(x_i) - h_n(x_j)|$$

The weights α_i and the functions $h_n(x_i)$ that map the input vector x_i into binary features are learned.

Shaknarovich and Darrell

Learning

Positive examples are pairs of images x_i, x_j so that x_j is one of the N nearest neighbors of x_i. Negative examples are pairs of images that are not neighbors.

In BoostS:

$$f_n(x_i, x_j) = \alpha_n \left[(e_n^T x_i > T_n) = (e_n^T x_j > T_n) \right] + \beta_n$$

$h_n(x_i)$

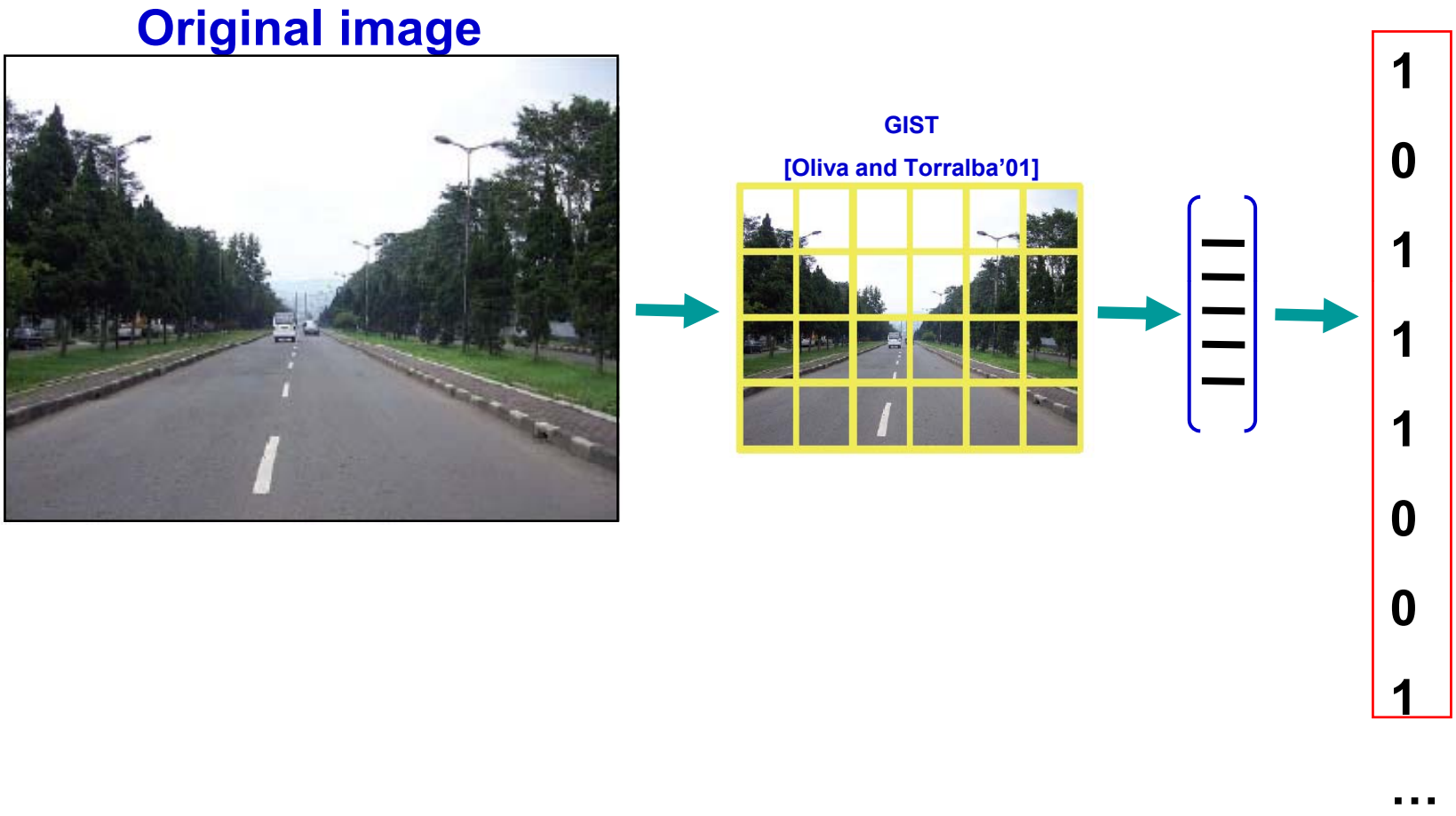
At each iteration n we select the parameters of f_n, the regression coefficients (α_n, β_n), the stump parameters (where e_n is a unit vector, so that $e^T_n x$ returns the k-th component of x, and T_n is a threshold), to minimiz

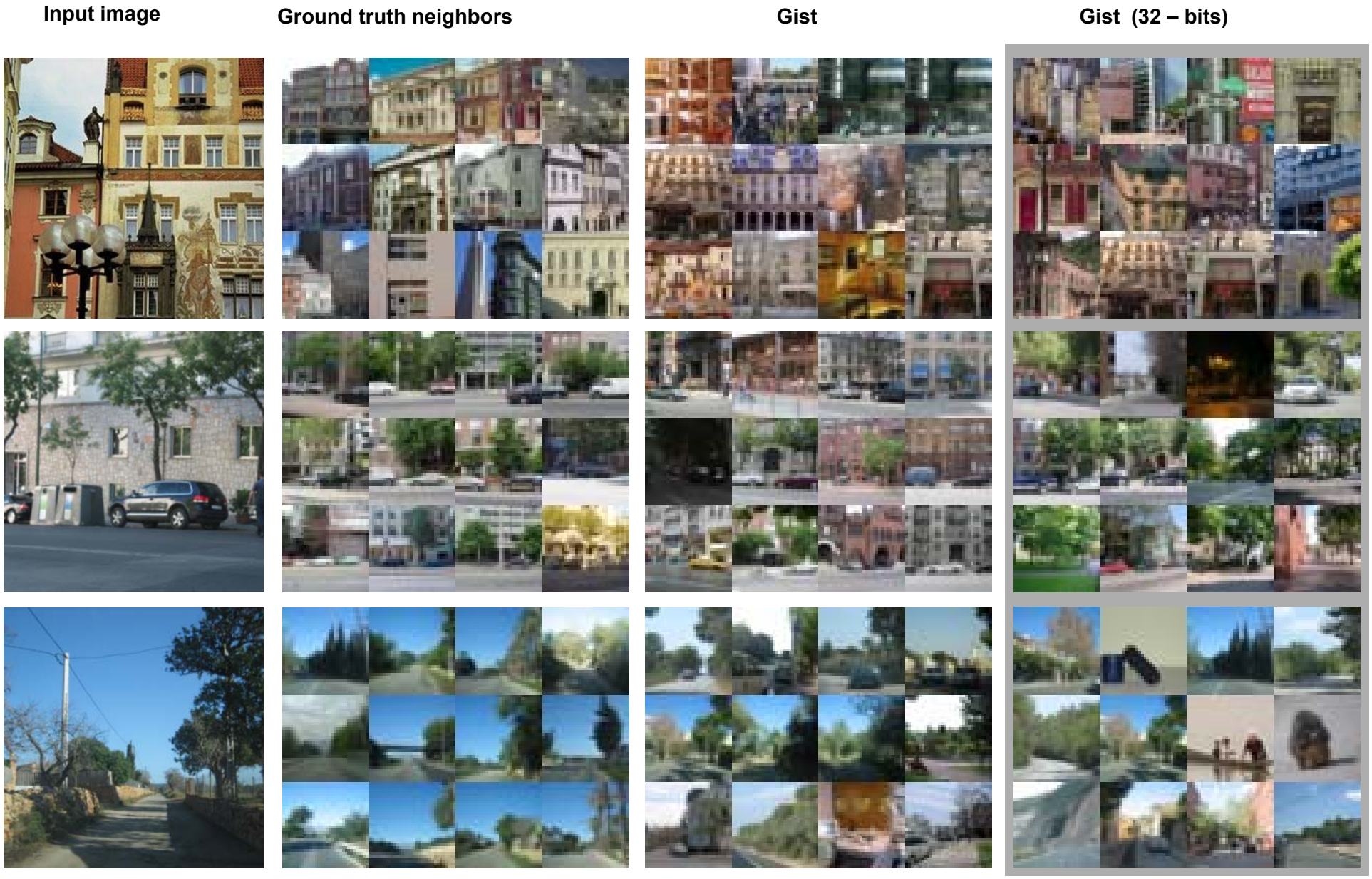
$$\sum_{k=1}^{K} w_n^k (z_k - f_n(x_i^k, x_j^k))^2$$

Where K is the number of training pairs, z_k is the neighborhood label ($z_k = 1$ if the two images are neighbors and $z_k = -1$ otherwise), and w^k_n is the weight for each training pair at iteration n given by

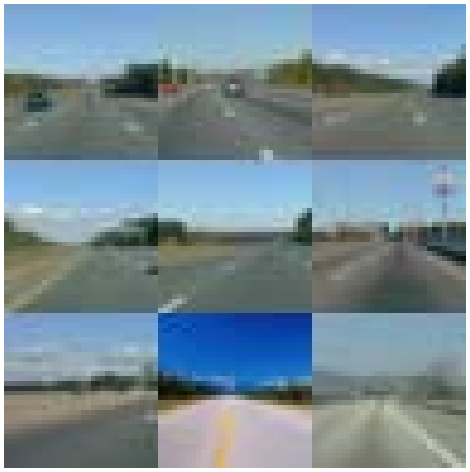
$$w_n^k = \exp(-z_k \textstyle\sum_{t=1}^{n-1} f_t(x_i^k, x_j^k))$$

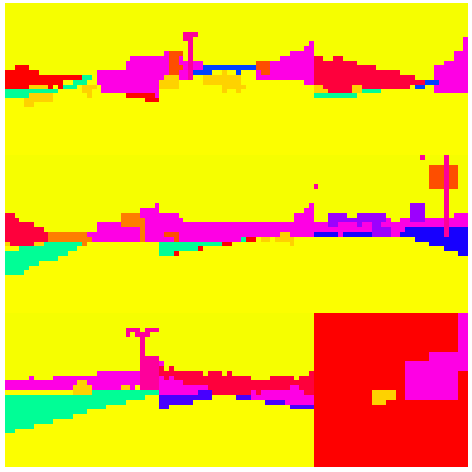
Compressing the gist descriptor




Input image
Ground truth neighbors
Gist
Gist (32 – bits)

How many bits do we need?



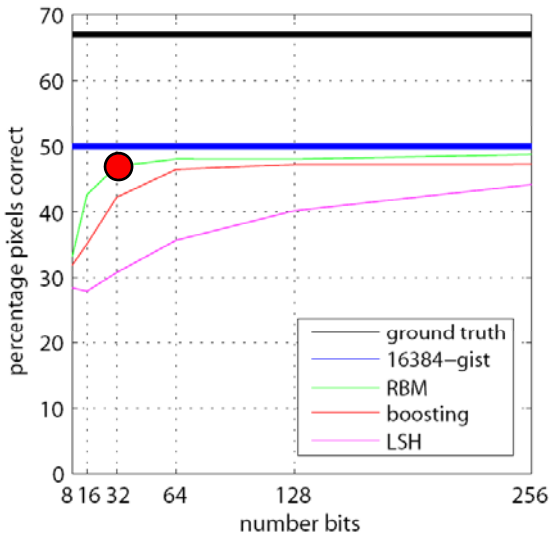


LabelMe: 22,000 images

32 bits

Tiny Images: 10^7 images

256 bits



Dataset	LabelMe	Web
# images	2×10^4	1.29×10^7
Gist vector dim.	512	384
Method	**Time (s)**	**Time (s)**
Spill tree - Gist vector	1.05	-
Brute force - Gist vector	0.38	-
Brute force - 30 bit binary	4.3×10^{-4}	0.146
" - 30 bit binary, M/T	2.7×10^{-4}	0.074
Brute force - 256 bit binary	1.4×10^{-3}	0.75
" - 256 bit binary, M/T	4.7×10^{-4}	0.23
Hashing - 30 bit binary	6×10^{-6}	6×10^{-6}

Today

- Demos
- Niebles Presentation from last week [Daphna B.]

- ISM (Leibe) – *Mario Fritz guest lecture*
- Learning Distances (Frome) – [Ashley E.]
- Hashing with Metric Learning – *Brian Kulis guest lecture*
- Semantic Texton Forests (Shotten)
- Forests/Ferns for Keypoint Localization (Leptit)
- Small Codes/Large Image Datasets (Torralba)

March 3rd – Discriminative approaches *(note revised reading list)*

- C. Dance, J. Willamowski, L. Fan, C. Bray, and G. Csurka, "Visual categorization with bags of keypoints," in ECCV International Workshop on Statistical Learning in Computer Vision, 2004. Available: http://www.xrce.xerox.com/Publications/Attachments/2004-010/2004_010.pdf
- M. Fritz; B. Leibe; B. Caputo; B. Schiele: Integrating Representative and Discriminant Models for Object Category Detection, ICCV'05, Beijing, China, 2005. Available: http://www.mis.informatik.tu-darmstadt.de/People/mfritz/fritz05iccv.pdf
- H. Zhang, A. C. Berg, M. Maire, and J. Malik, "Svm-knn: Discriminative nearest neighbor classification for visual category recognition," in CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Washington, DC, USA: IEEE Computer Society, 2006, pp. 2126-2136. Available: http://dx.doi.org/10.1109/CVPR.2006.301
- P. Felzenszwalb, D. Mcallester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in IEEE International Conference on Computer Vision and Pattern Recognition (CVPR) Anchorage, Alaska, June 2008., June 2008. Available: http://www.ics.uci.edu/~dramanan/papers/latent.pdf

Optional Readings:

- Y. Wang and G. Mori, “Learning a Discriminative Hidden Part Model for Human Action Recognition”, Advances in Neural Information Processing Systems (NIPS), 2008; http://www.sfu.ca/~ywang12/papers/nips.pdf