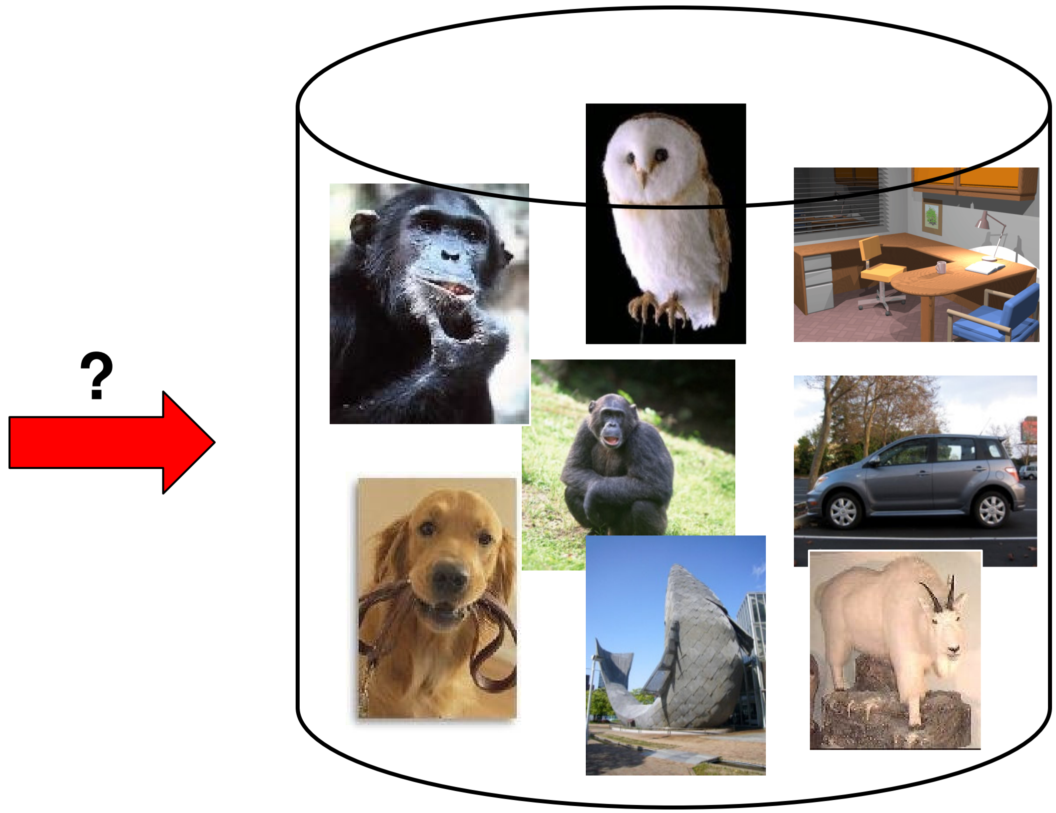
Fast Indexing and Search

Lida Huang, Ph.D.
Senior Member of Consulting Staff
Magma Design Automation

Motivation

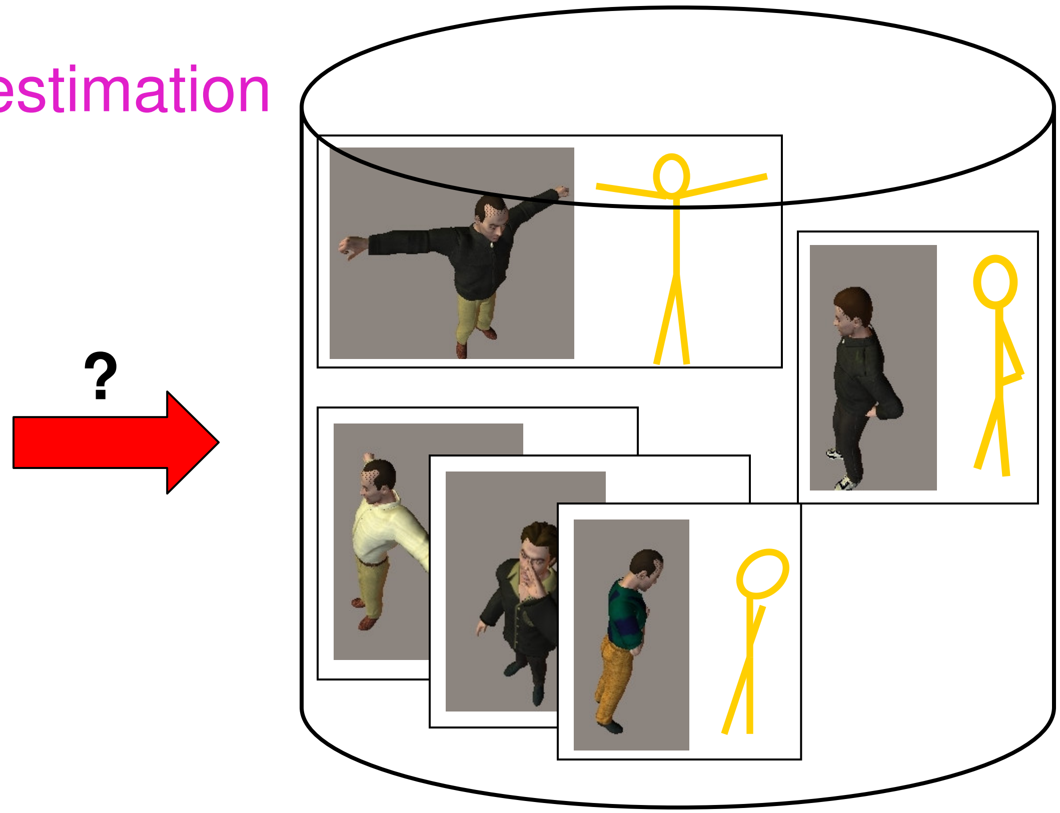
Object categorization



http://www.cs.utexas.edu/~grauman/slides/jain_et_al_cvpr2008.ppt

Motivation

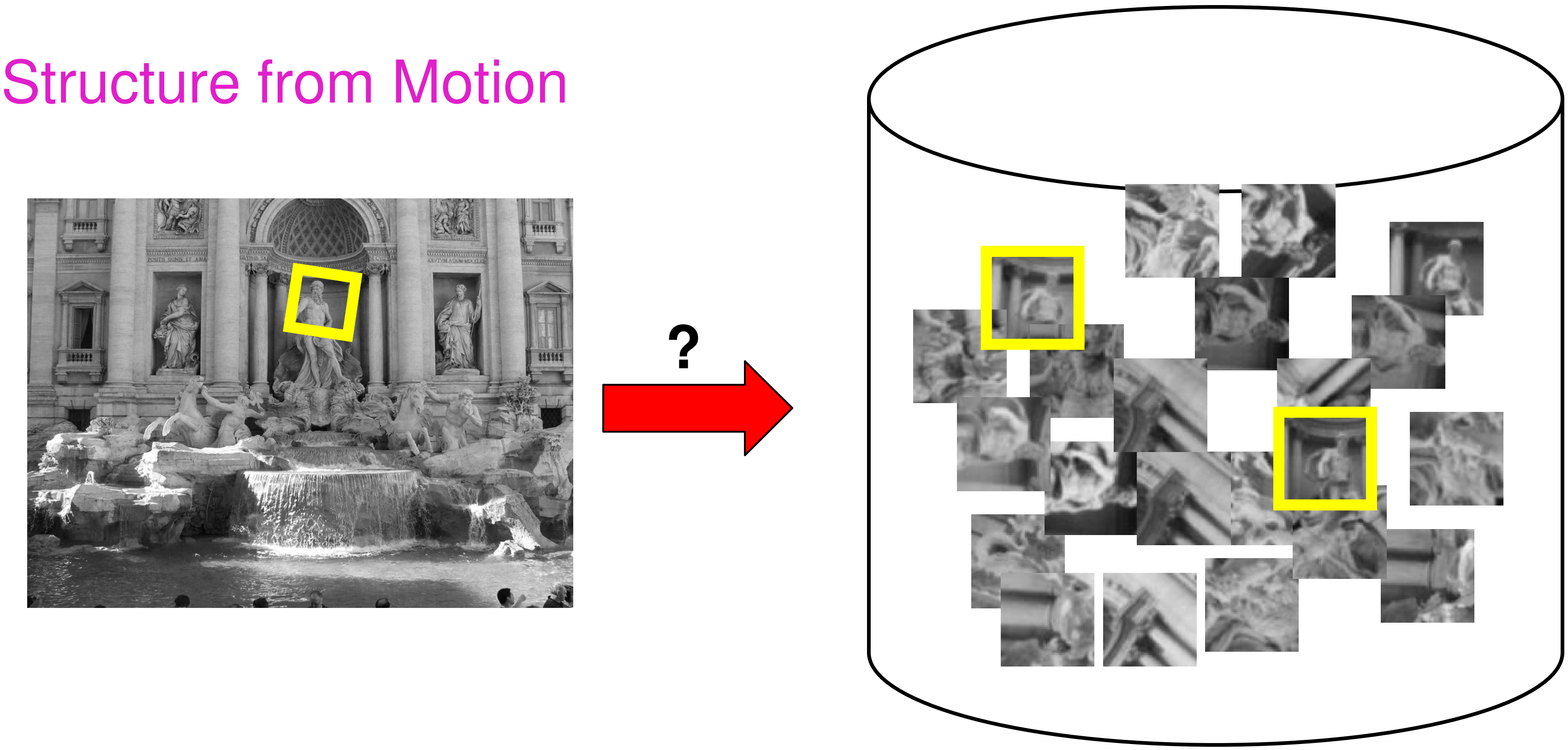
Example-based pose estimation



http://www.cs.utexas.edu/~grauman/slides/jain_et_al_cvpr2008.ppt

Motivation

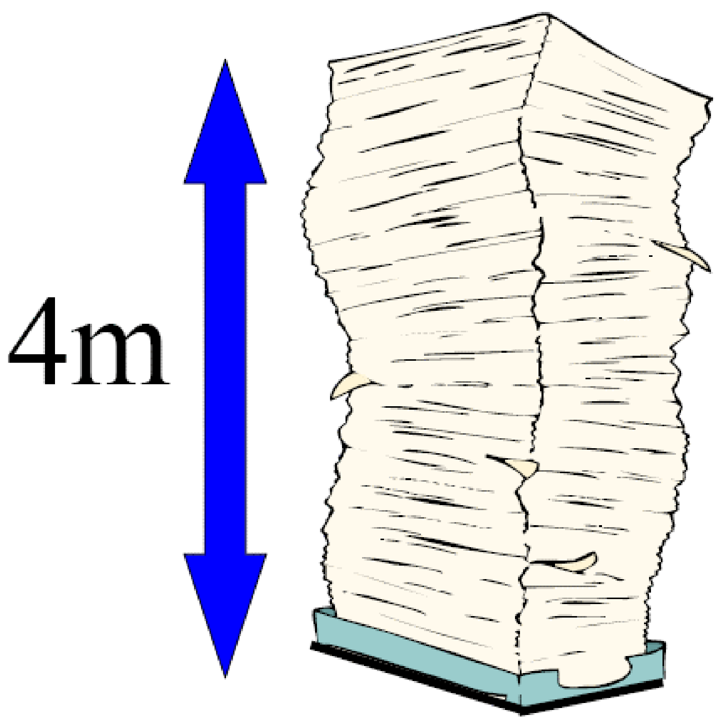
Structure from Motion



http://www.cs.utexas.edu/~grauman/slides/jain_et_al_cvpr2008.ppt

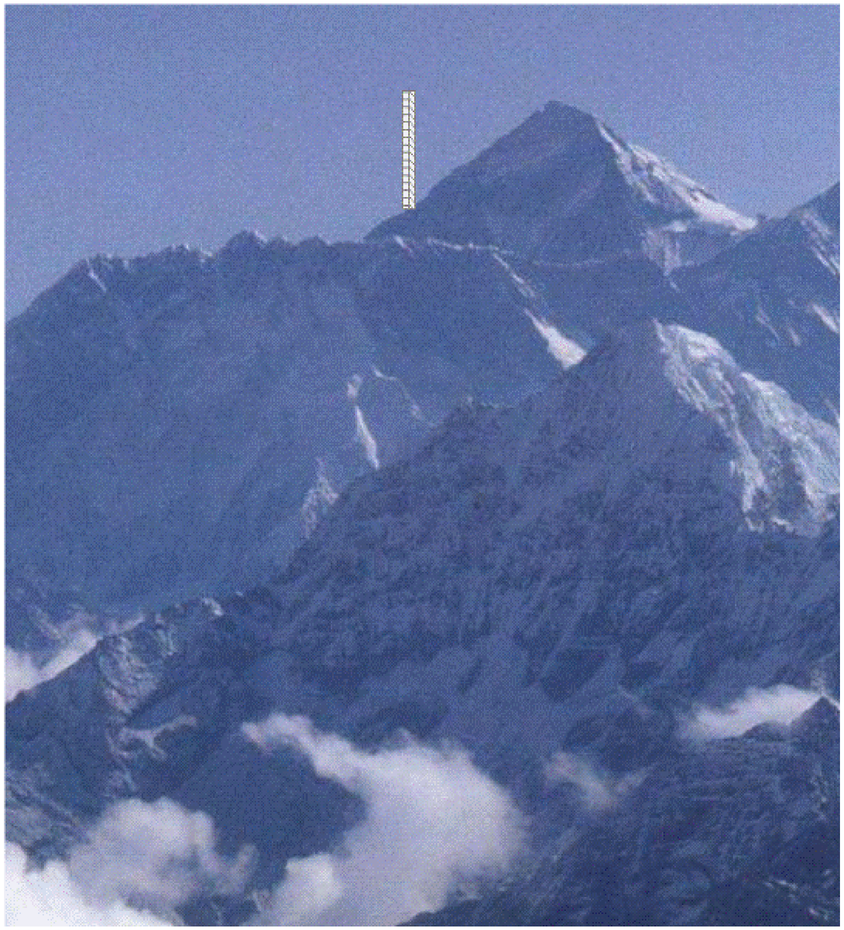
Motivation: Fast! Really Fast Needed!!!

50 Thousand Images

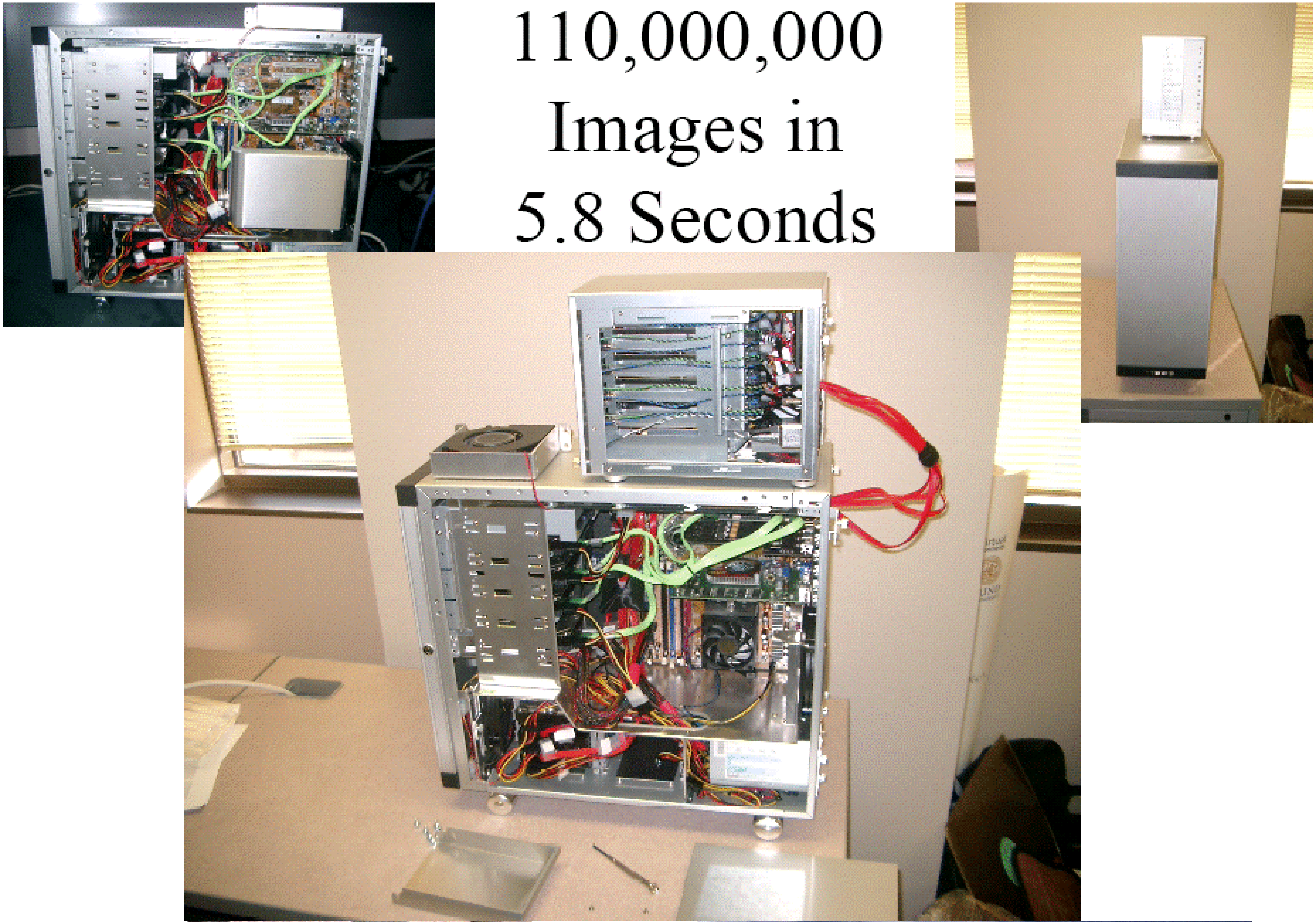


www.cs.ualberta.ca/~vis/vision06/slides/***birs2006-nister***-index.pdf

110,000,000 images Equals 8,800 Meters



110,000,000 Images in 5.8 Seconds



www.cs.ualberta.ca/~vis/vision06/slides/**birs2006-nister**-index.pdf

Overview

- Efficient Near-Duplicate Detection and Sub-Image Retrieval (Multimedia 2004)
  - Parts Based representation
  - LSH
  - Optimize the access on the Hard-Disk
- Scalable Recognition with a Vocabulary Tree (CVPR 2006)
  - Hierarchically quantized in vocabulary tree
- Fast Image Search for Learned Metrics (CVPR 2008)
  - Similarity and Dissimilarity Constraints
  - Learned Mahalanobis distance
  - Randomized locality-sensitive hash function.

Efficient Near-Duplicate Detection and Sub-Image Retrieval

- Part-Based Representation of Images
- Locality-Sensitive Hashing(LSH)
- Layout of the Data on the Hard-Disk

http://www.cs.cmu.edu/~yke/retrieval/mm2004-retrieval.pdf

Photographer: Lieutenant Ivor Castle

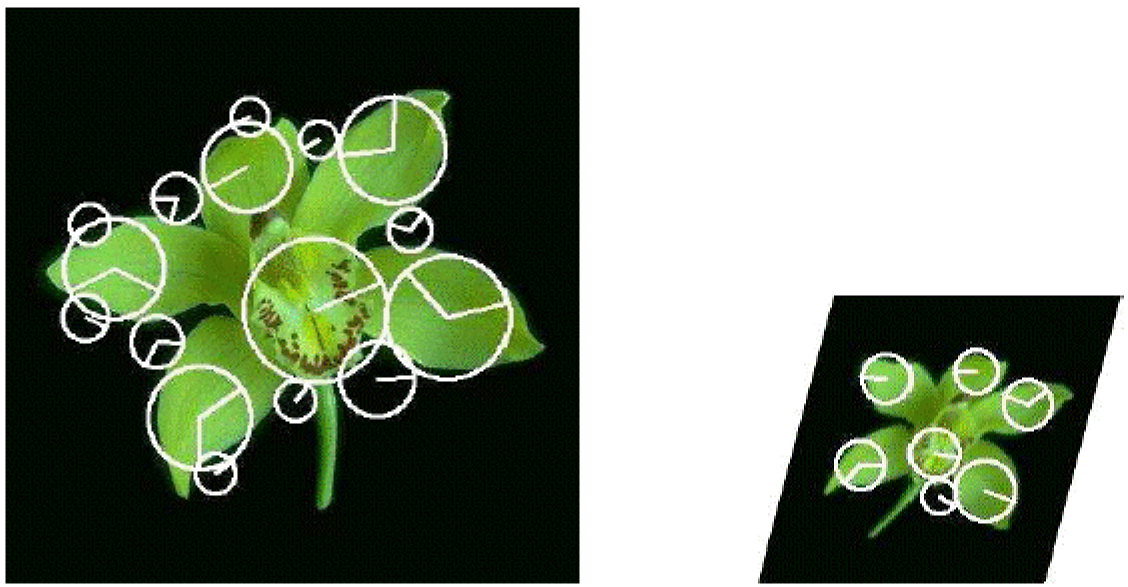
29th Infantry Battalion advancing over No Man's Land during the Battle of Vimy Ridge, 1917

http://www.collectionscanada.gc.ca/forgery/002035-200-e.html#a

Original Film

http://www.collectionscanada.gc.ca/forgery/002035-200-e.html#a

Extracted Key Features



(a) Original image (b) Rotated, scaled, and sheared

http://www.cs.cmu.edu/~yke/retrieval/mm2004-retrieval.pdf

Automatically Generated Near-Duplicates

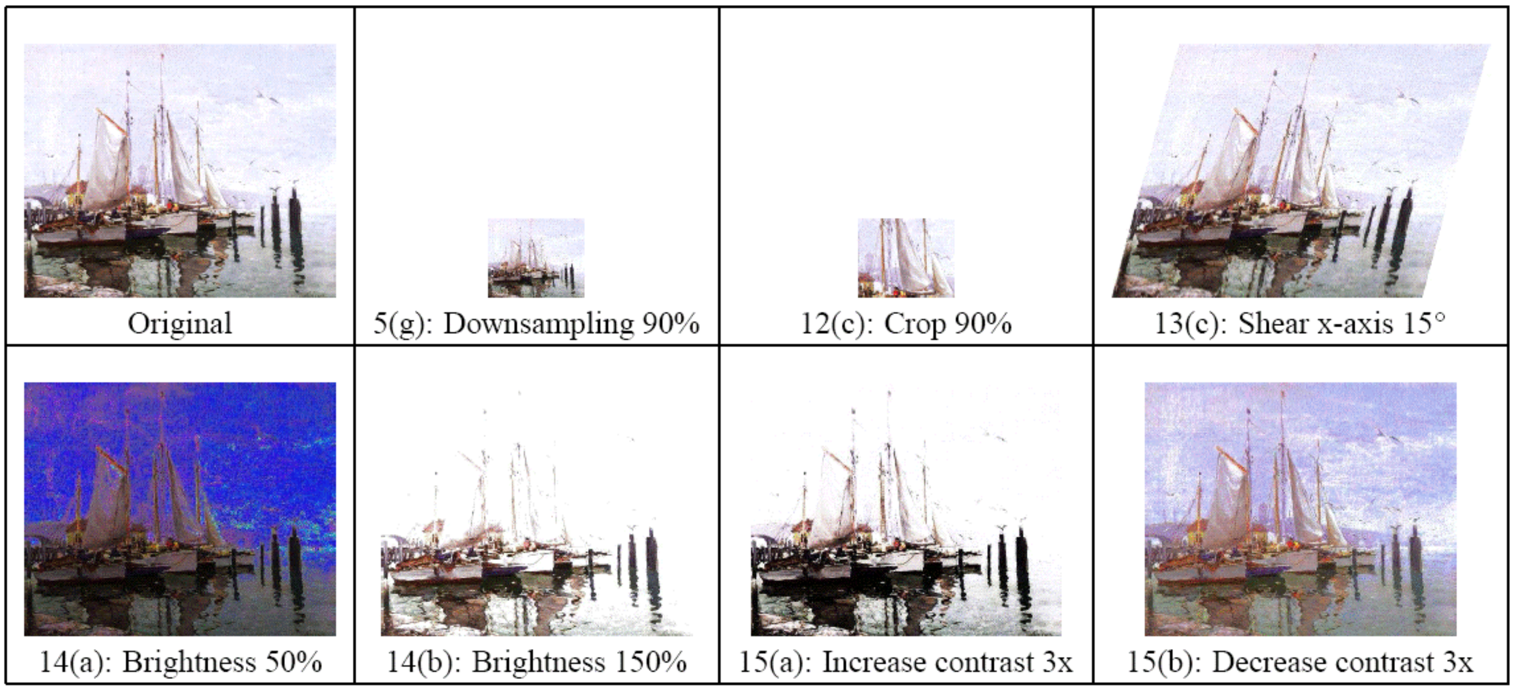


Figure 5: Examples of automatically-generated near-duplicates. Only 7 out of 50 transforms are shown; all were correctly identified.

http://www.cs.cmu.edu/~yke/retrieval/mm2004-retrieval.pdf

50,000 images
1000 keypoing/image
==
50M keypoints

LSH build ℓ independent Hashtables

Each hash table:
1M keypoints

Index Construction

1. For each image in the gallery:
2.   Find keypoints using DoG detector
3.   Build PCA-SIFT local descriptors for each keypoint
4. Build and store file name table (FT)
5. Build and store keypoint table (KT)
6. For each of the l hash tables (HTs):
7.   For each keypoint:
8.   Hash keypoint and store id in table (in memory)
9.   Store hash table (HT) on disk

Database Query

1. Find keypoints in query image using DoG detector
2. For each keypoint:
3.   Build its PCA-SIFT local descriptor
4.   Compute the l LSH hashes for the descriptor
5. Sort hashes by bucket id, scan hash tables (HTs)
6. Sort returned keypoint ids and scan KT linearly
7. For each returned image:
8.   Determine best affine transform using RANSAC
9.   Discard if a valid transform was not found
10. Print matched file names by reading FT

http://www.cs.cmu.edu/~yke/retrieval/mm2004-retrieval.pdf

File Name Table (FT)

	Byte 1	Byte 2	…	Byte 256
ID	Len	File name		
1	xxx	File 1		
2	xxx	File 2		
…	…	…		

Keypoint Table (KT)

	Bytes 1-4	Bytes 5-8	Bytes 9-12	Bytes 13-16	Bytes 17-20	Bytes 21-92
ID	File ID	X	Y	Size	Orien.	Local Descr.
1	aaa					
2	bbb					
…	…	…	…	…	…	…

Layout of *one* hash table (HT)

	Bytes 1-4	Bytes 5-8	Bytes 9-12	Bytes 13-16	…	
	Keypoint 1		Keypoint 2			
Bucket ID	Key ID	Hash Val	Key ID	Hash Val		
1						
2						
…	…	…	…	…	…	…

Figure 3: Format of the disk-based data structures.

http://www.cs.cmu.edu/~yke/retrieval/mm2004-retrieval.pdf

Evaluation Metrics

$$recall = \frac{\text{number of } \textit{correct-positives}}{\text{total number of positives}}$$

and

$$precision = \frac{\text{number of } \textit{correct-positives}}{\text{total number of matches (correct or false)}}.$$

http://www.cs.cmu.edu/~yke/retrieval/mm2004-retrieval.pdf

Retrieval Results

Table 1: Recall-Precision for standard transformations.

recall	precision
Baseline - select 40 random images	
0.3%	0.01%
Weighted Sampling Threshold method from [16]	
90%	67%
100%	6%
Our method on art database of 12,000 images	
99.85%	100%

147 global color, texture
and shape features
132 color-based local features
Total **279** features

Table 2: Recall-Precision for difficult transformations.

	recall	precision
Art database of 7,611 images		
	98.40%	99.86%
MM270K database of 18,722 images		
Original	96.78%	88.78%
Same scene removed	96.78%	96.12%

http://www.cs.cmu.edu/~yke/retrieval/mm2004-retrieval.pdf

Recall Precision and RunTime

Table 3: Recall-Precision for composite images.

recall	precision
98.85%	99.65%

Table 4: Efficiency of LSH versus linear search.

	linear search	LSH
Running time in sec. (σ)	80.3 (0.06)	0.97 (0.04)
Pairs of keys checked	268 million	2656
Pairs of keys matched	5464	1611

http://www.cs.cmu.edu/~yke/retrieval/mm2004-retrieval.pdf

L1 in LSH
L2 in PCA-SIFT

Table 5: Inefficiency due to L1 assumption.

	No. of keys
Checked by LSH	2656
Matched under L1 ($d \leq 18000$)	1674
Matched under L2 ($d \leq 3000$)	1611
Checked because of hash table collision	982
Matched under L1 but not L2	63

Table 6: Importance of building hash table in memory.

	Running time in sec. (σ)
Build directly on disk	325 (1.8)
Build in memory, stream to disk	48 (0.1)

http://www.cs.cmu.edu/~yke/retrieval/mm2004-retrieval.pdf

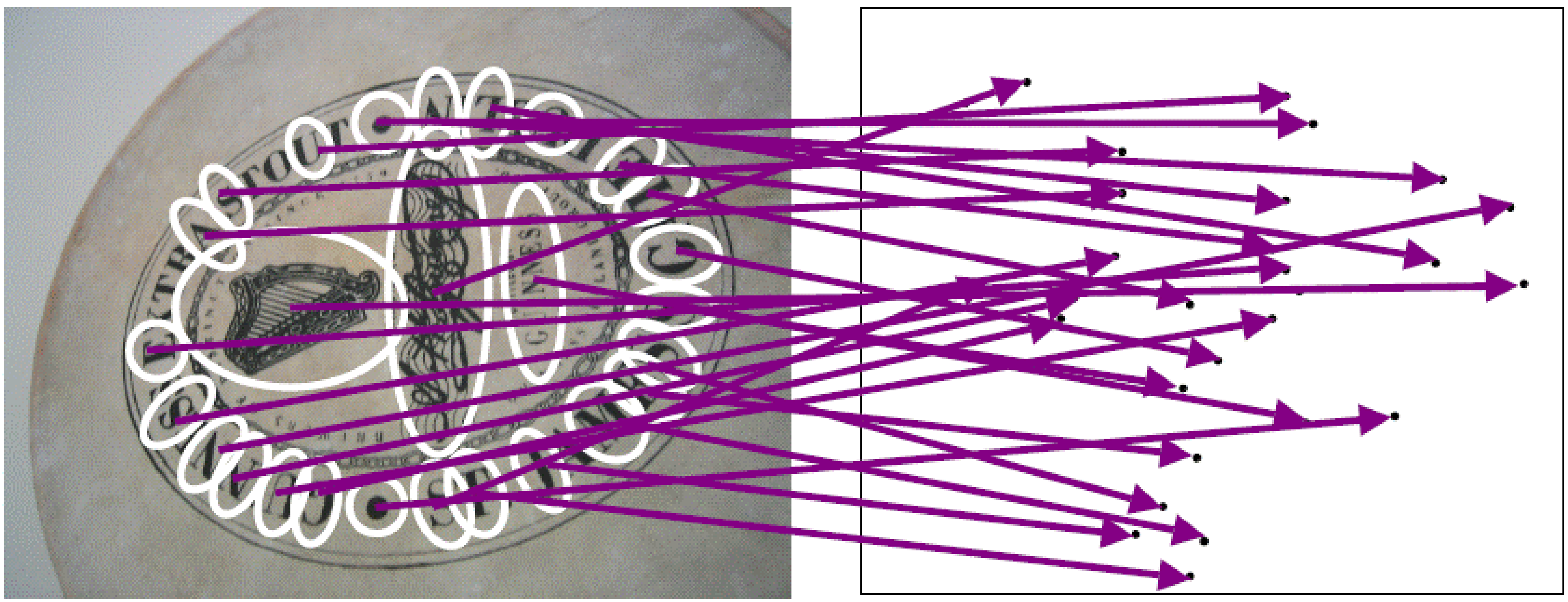
Comments and Discussions

- Proved the local features could be used more effective and robust in image matching.
- Scalability?
- Efficiency?
- Pointed out the implementation details.

Repeatable Discriminative Features

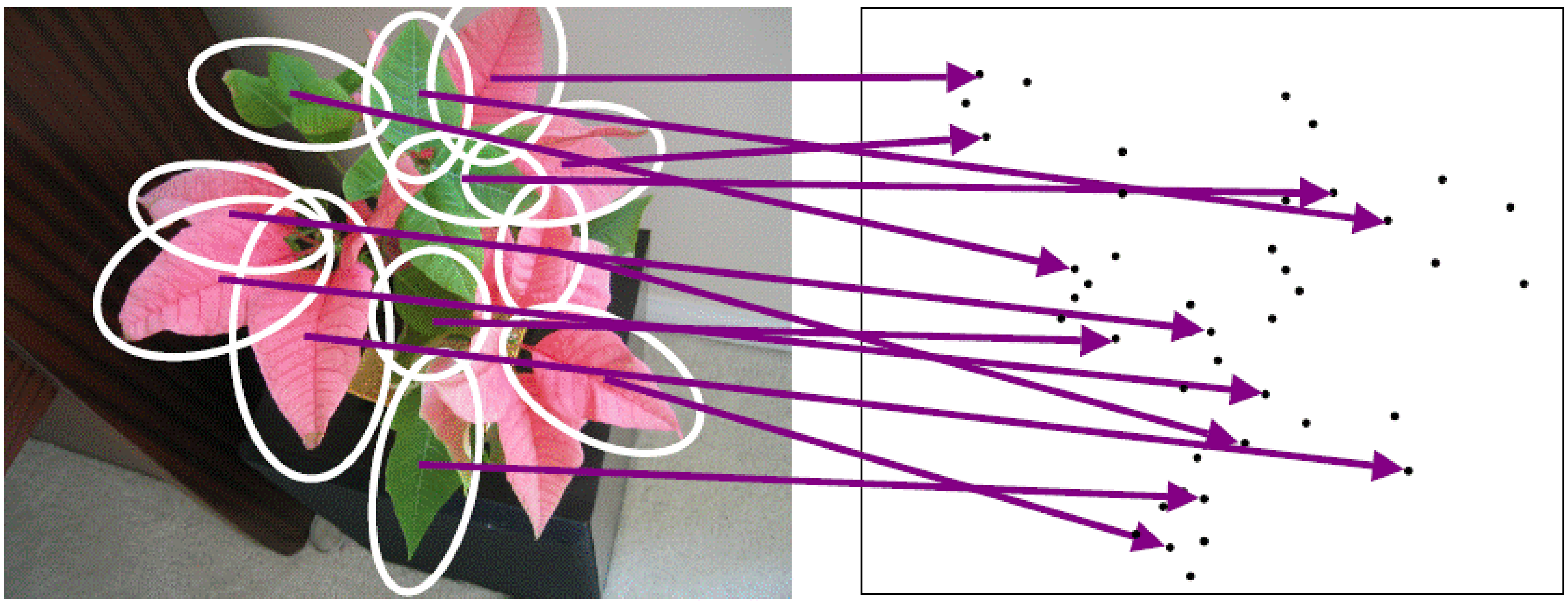
- Scalable Recognition with a Vocabulary Tree (CVPR 2006)
- Find out the most efficient way to represent the images.
- Reuse as much as possible.
- Simply saying: Restructure it well. Coding Theory.

Visual Words



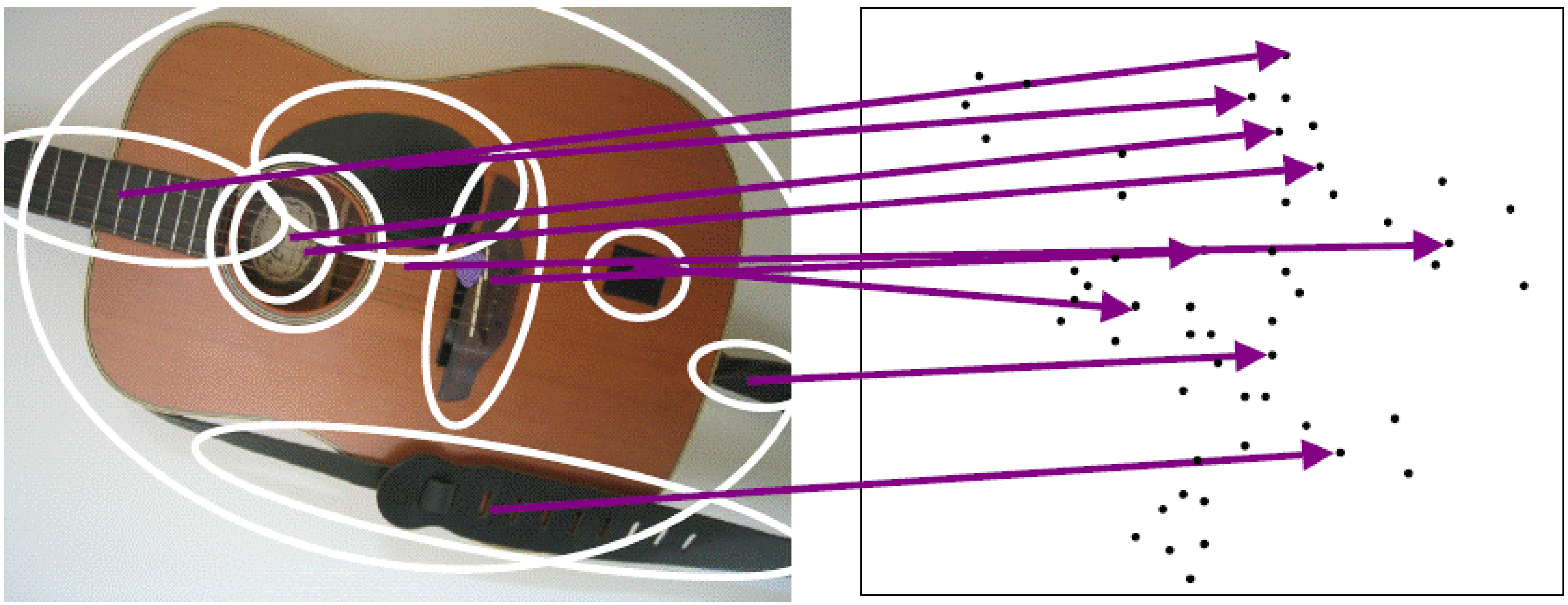
*www.cs.ualberta.ca/~vis/vision06/slides/**birs2006-nister**-index.pdf*

Visual Words



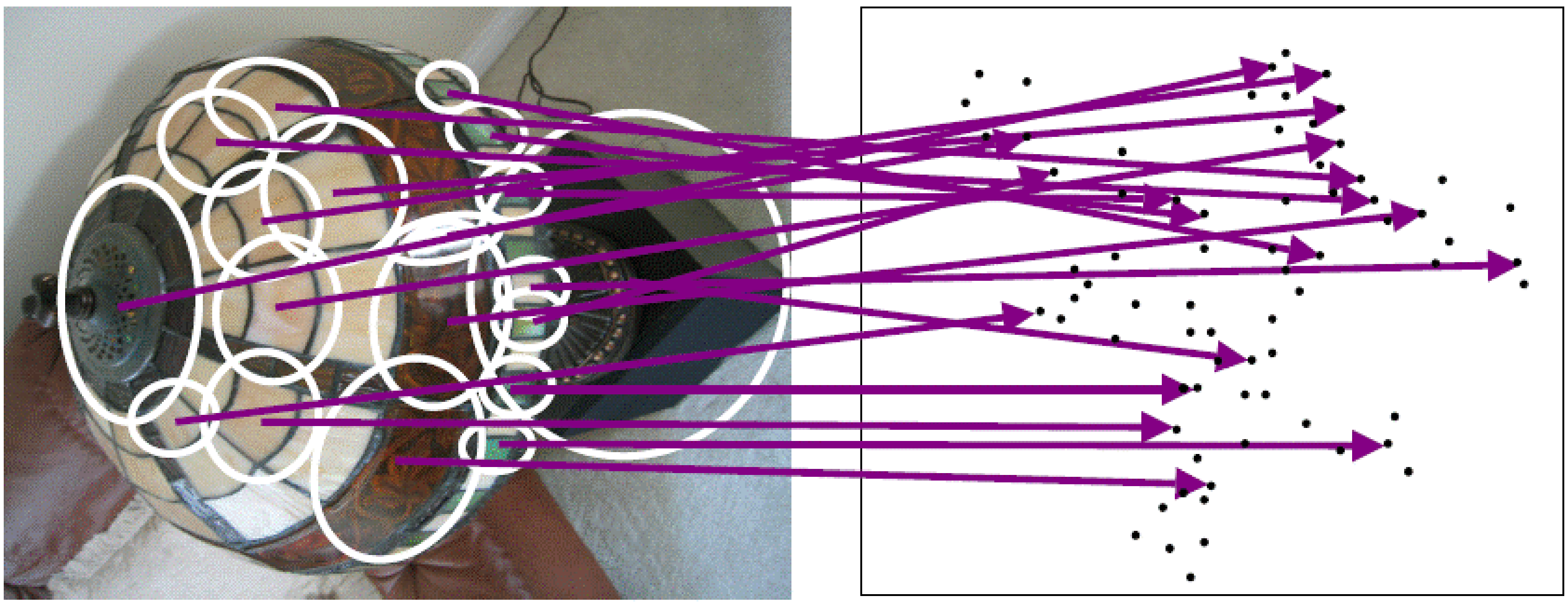
*www.cs.ualberta.ca/~vis/vision06/slides/**birs2006-nister**-index.pdf*

Visual Words



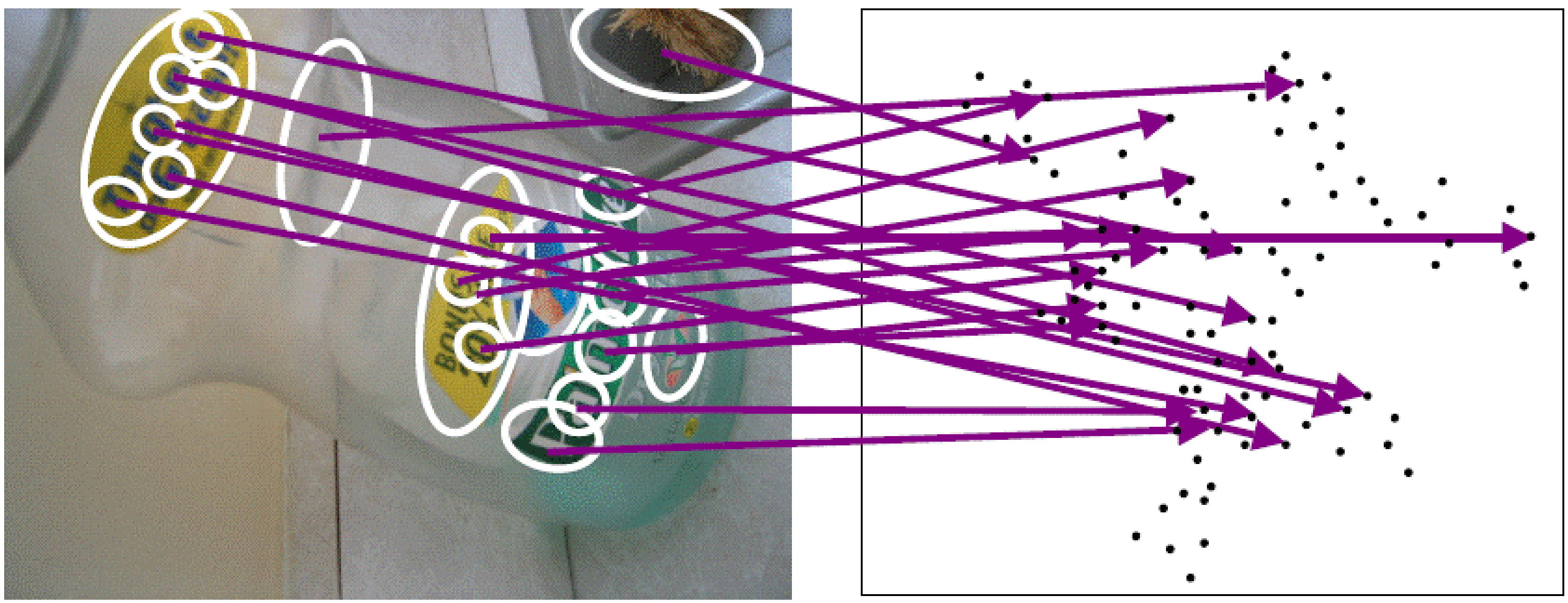
*www.cs.ualberta.ca/~vis/vision06/slides/**birs2006-nister**-index.pdf*

Visual Words



*www.cs.ualberta.ca/~vis/vision06/slides/**birs2006-nister**-index.pdf*

Visual Words

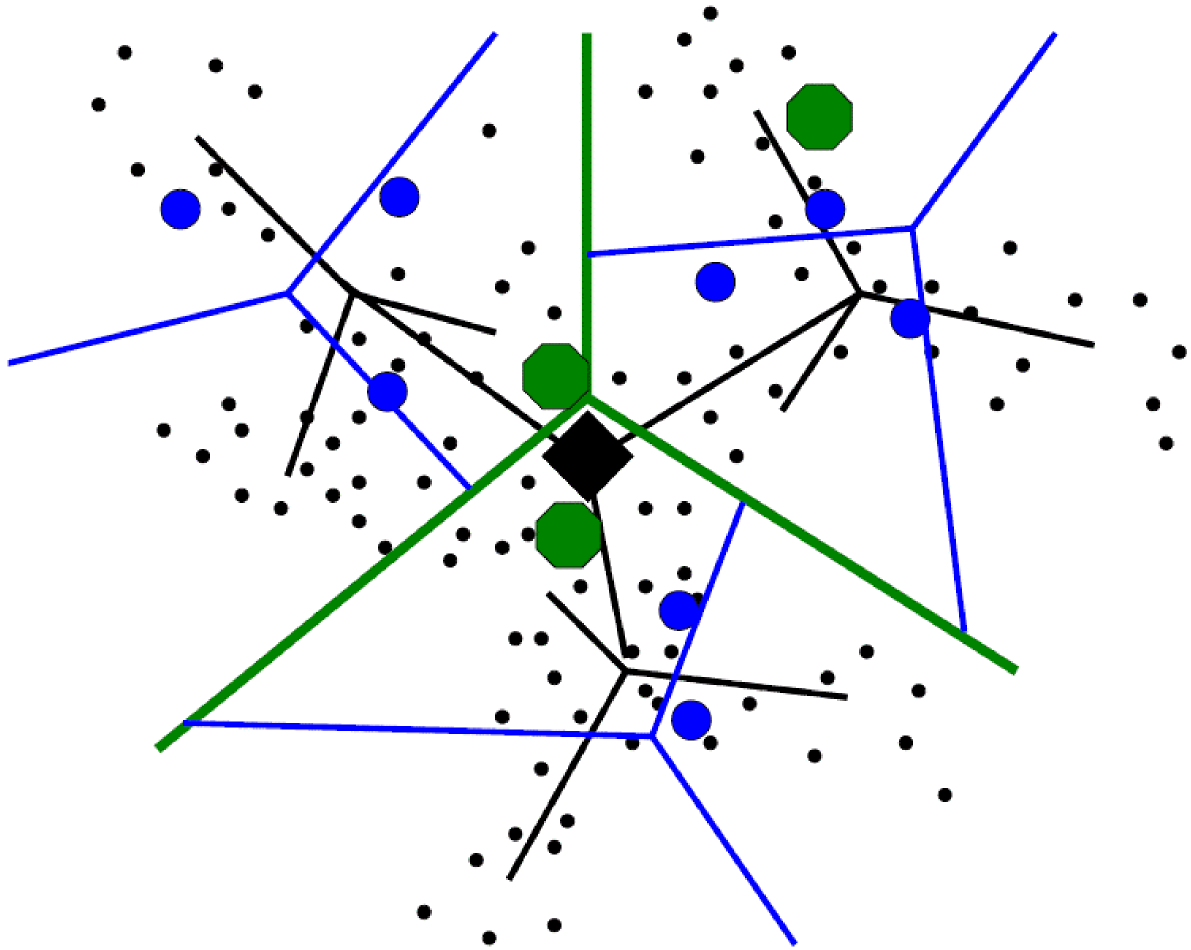


*www.cs.ualberta.ca/~vis/vision06/slides/**birs2006-nister**-index.pdf*

Visual Words Cluster Naturally

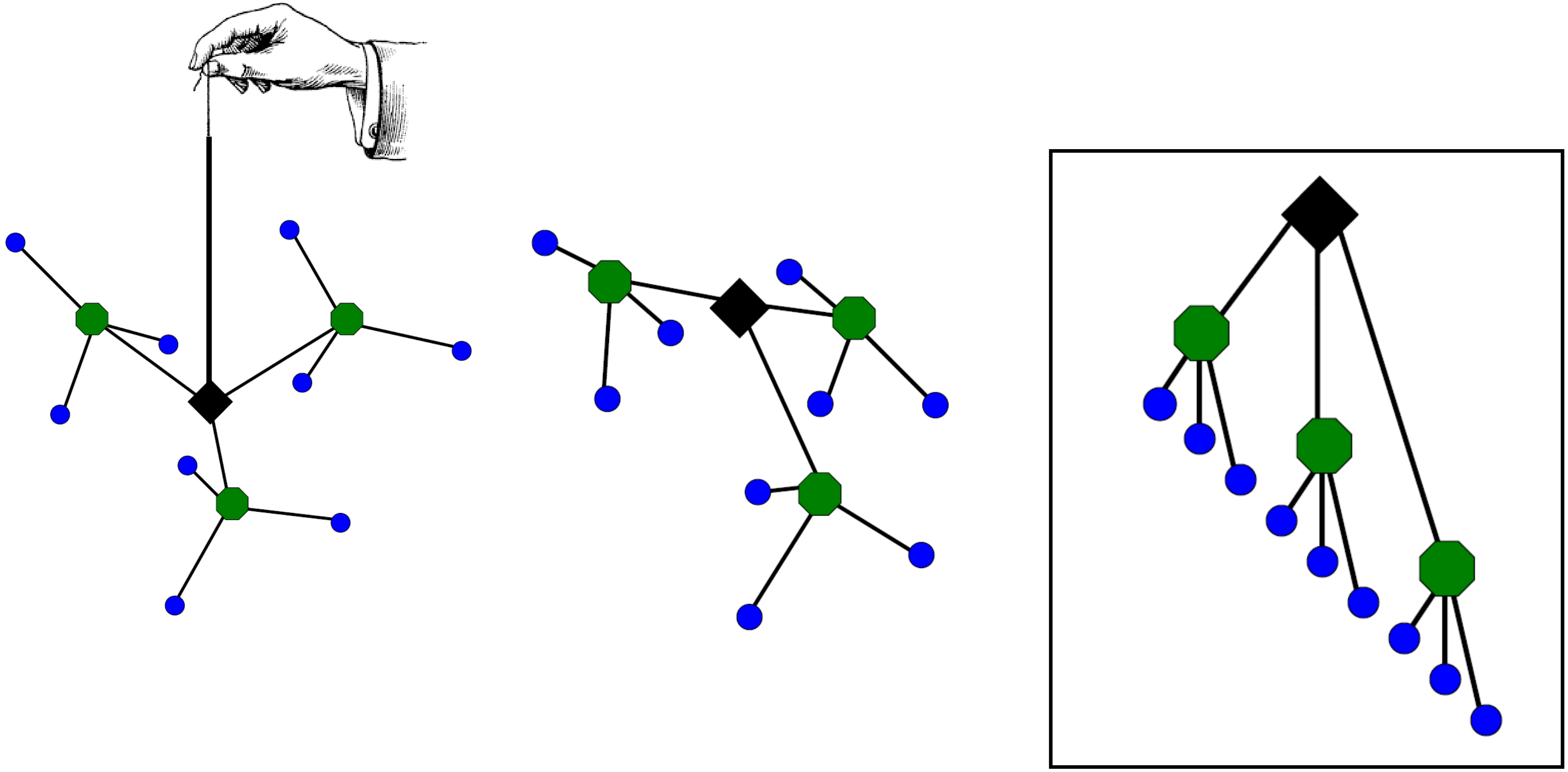
*www.cs.ualberta.ca/~vis/vision06/slides/**birs2006-nister**-index.pdf*

Hierarchically Clustering

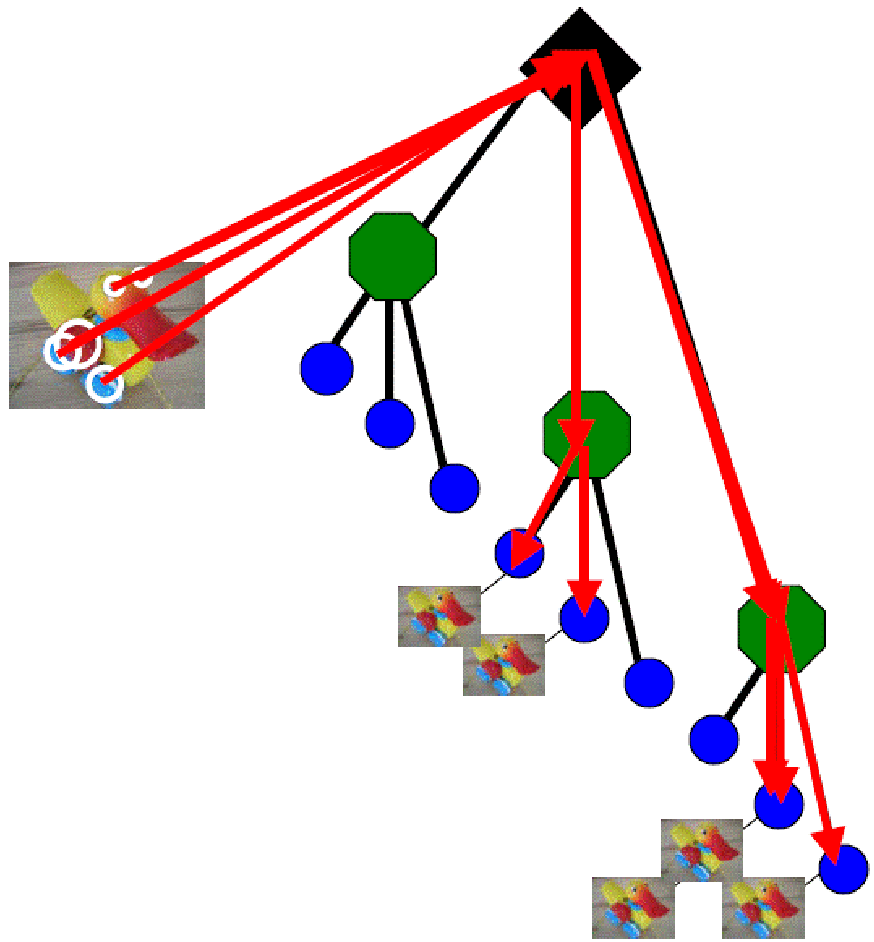


*www.cs.ualberta.ca/~vis/vision06/slides/**birs2006-nister**-index.pdf*

Visualized as a Tree

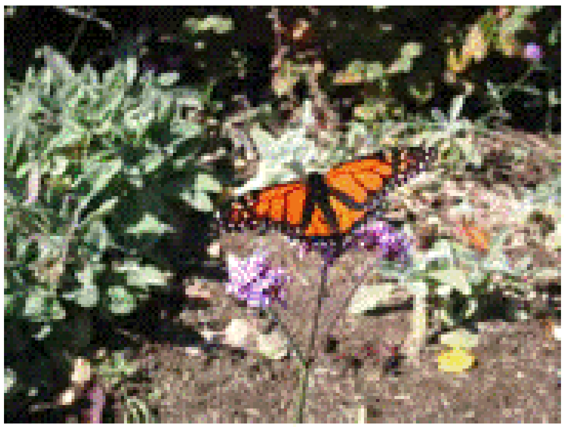


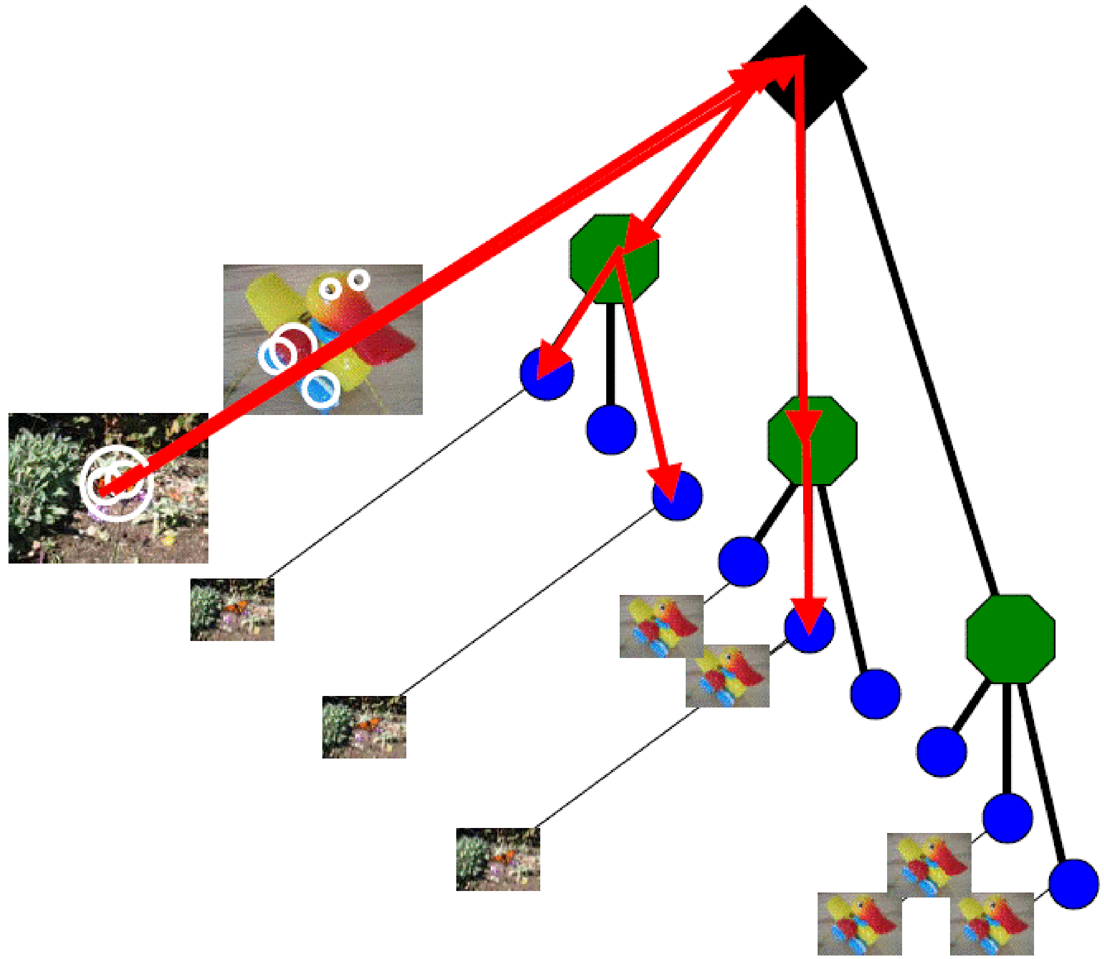
Item Added



www.cs.ualberta.ca/~vis/vision06/slides/***birs2006-nister***-index.pdf

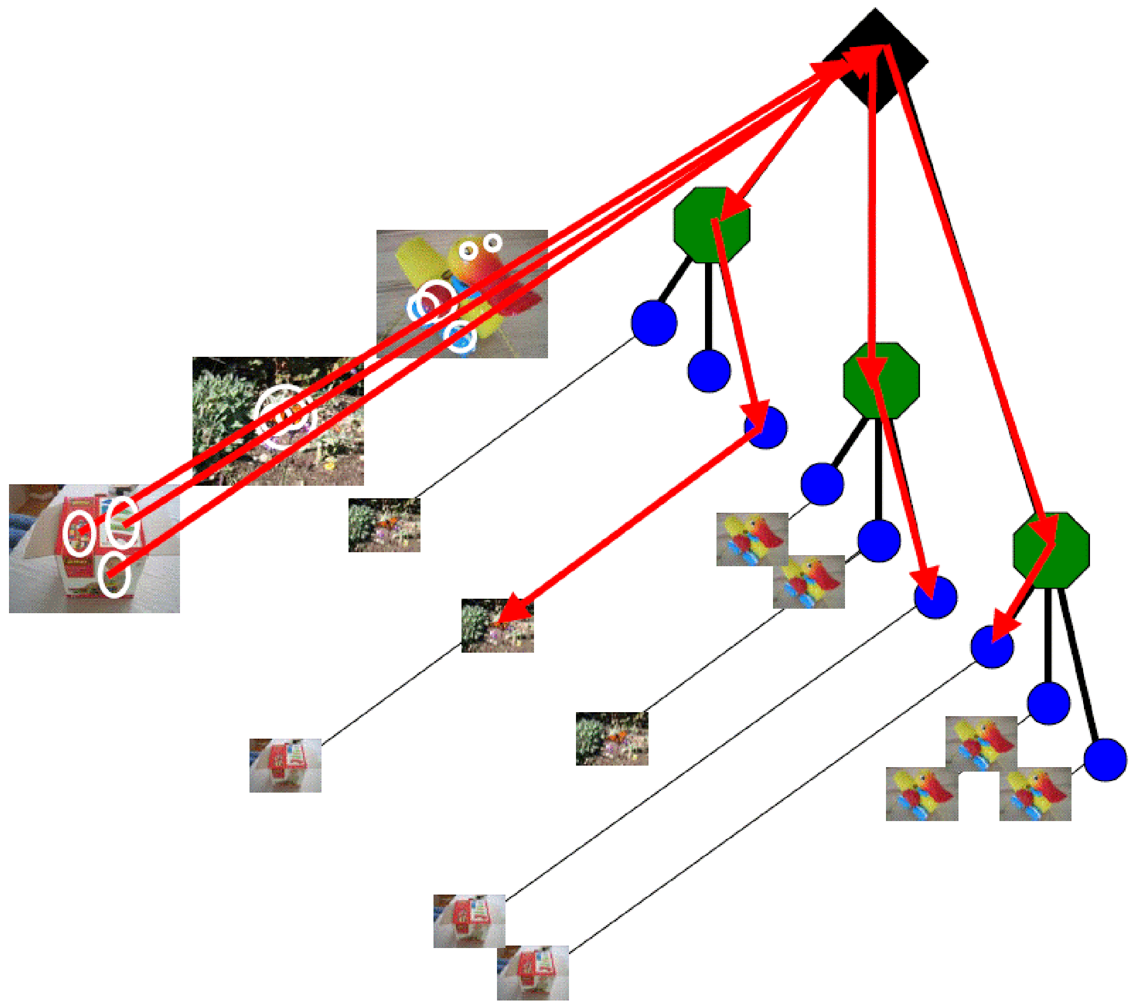
Item Added





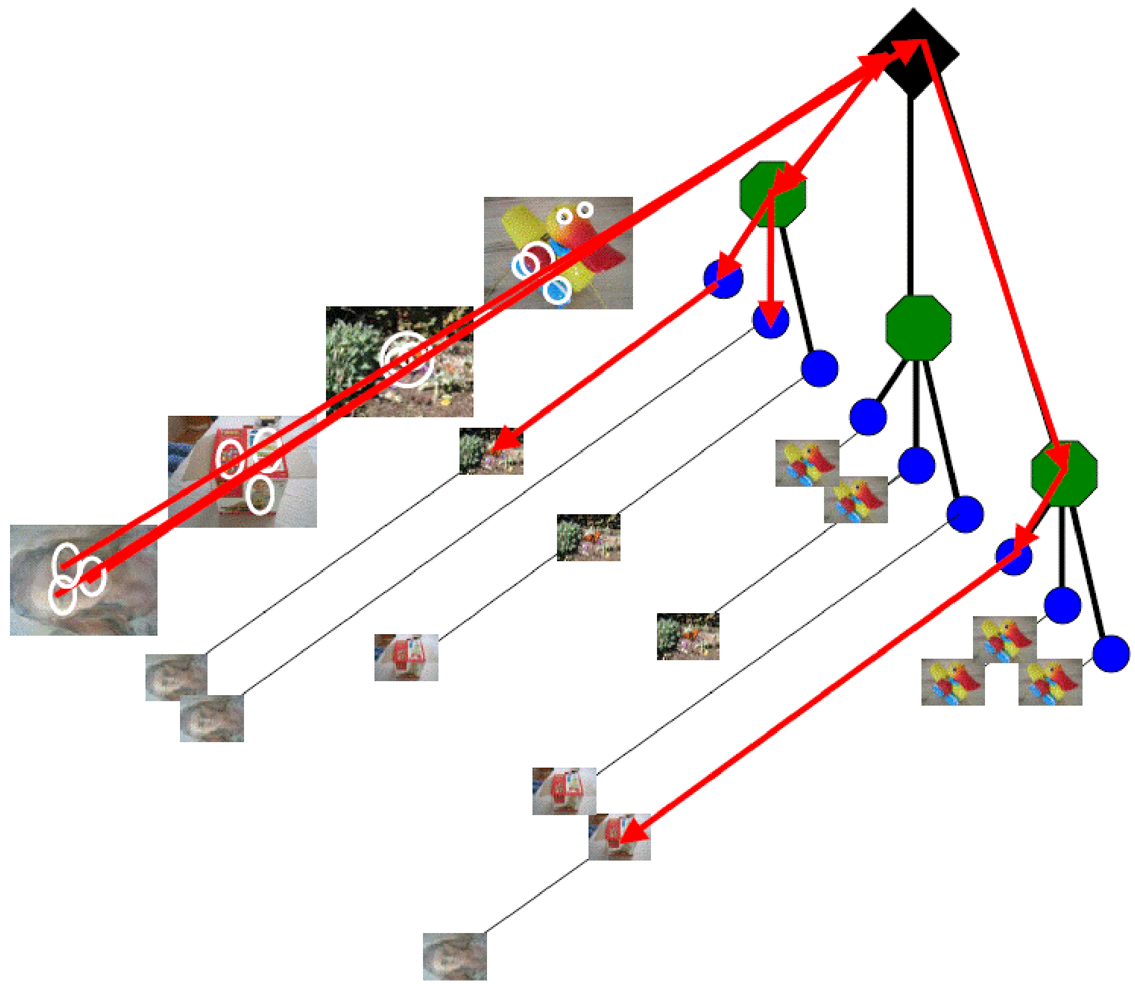
www.cs.ualberta.ca/~vis/vision06/slides/***birs2006-nister***-index.pdf

Item Added



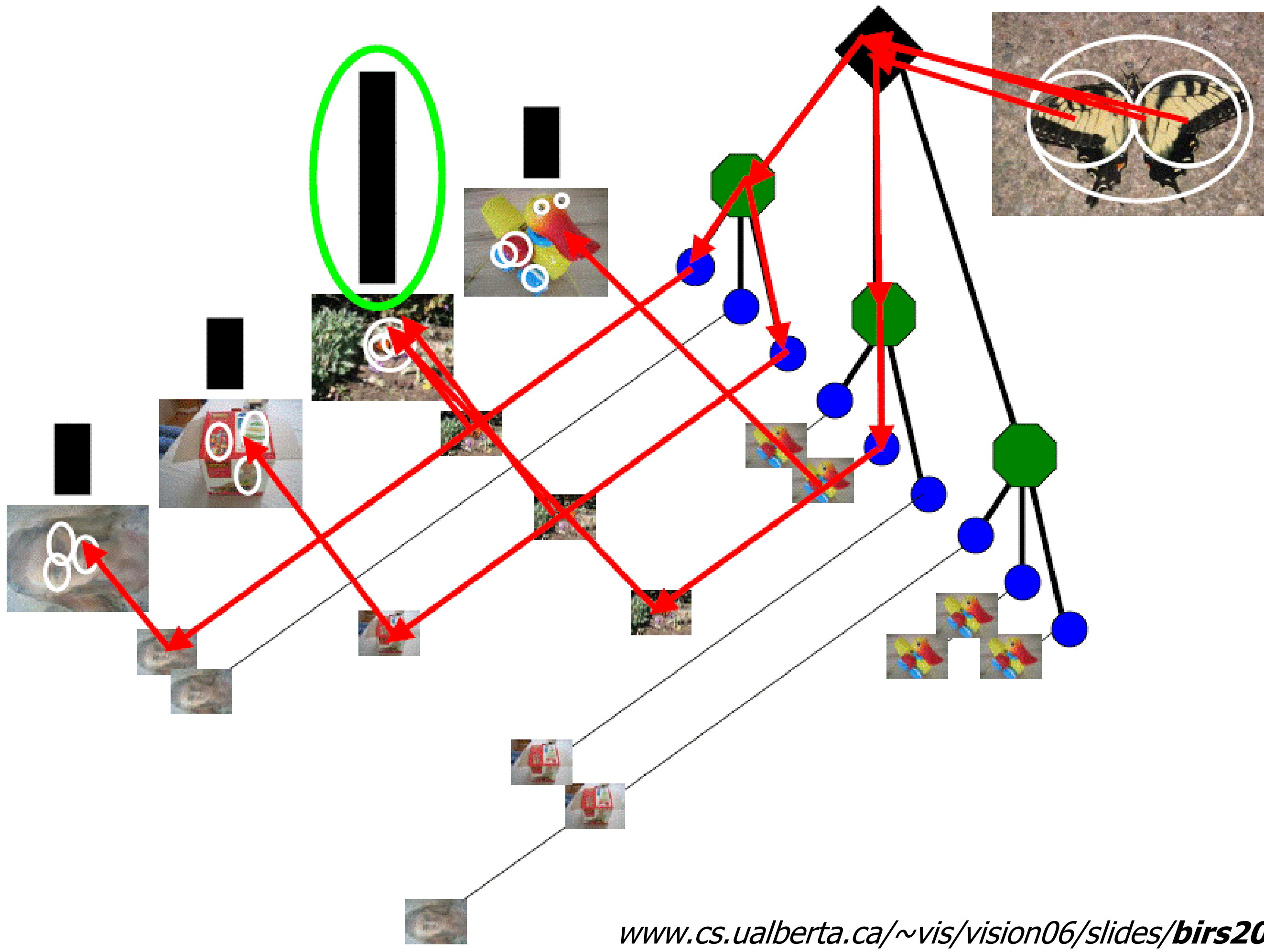
*www.cs.ualberta.ca/~vis/vision06/slides/**birs2006-nister**-index.pdf*

Item Added



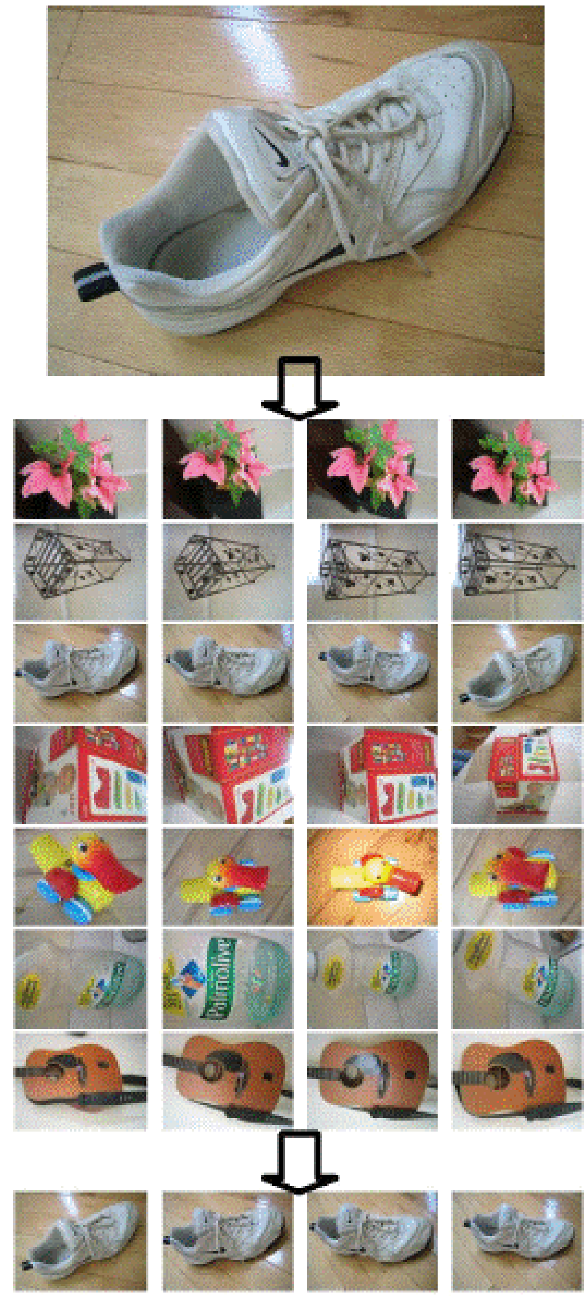
*www.cs.ualberta.ca/~vis/vision06/slides/**birs2006-nister**-index.pdf*

Item Queried

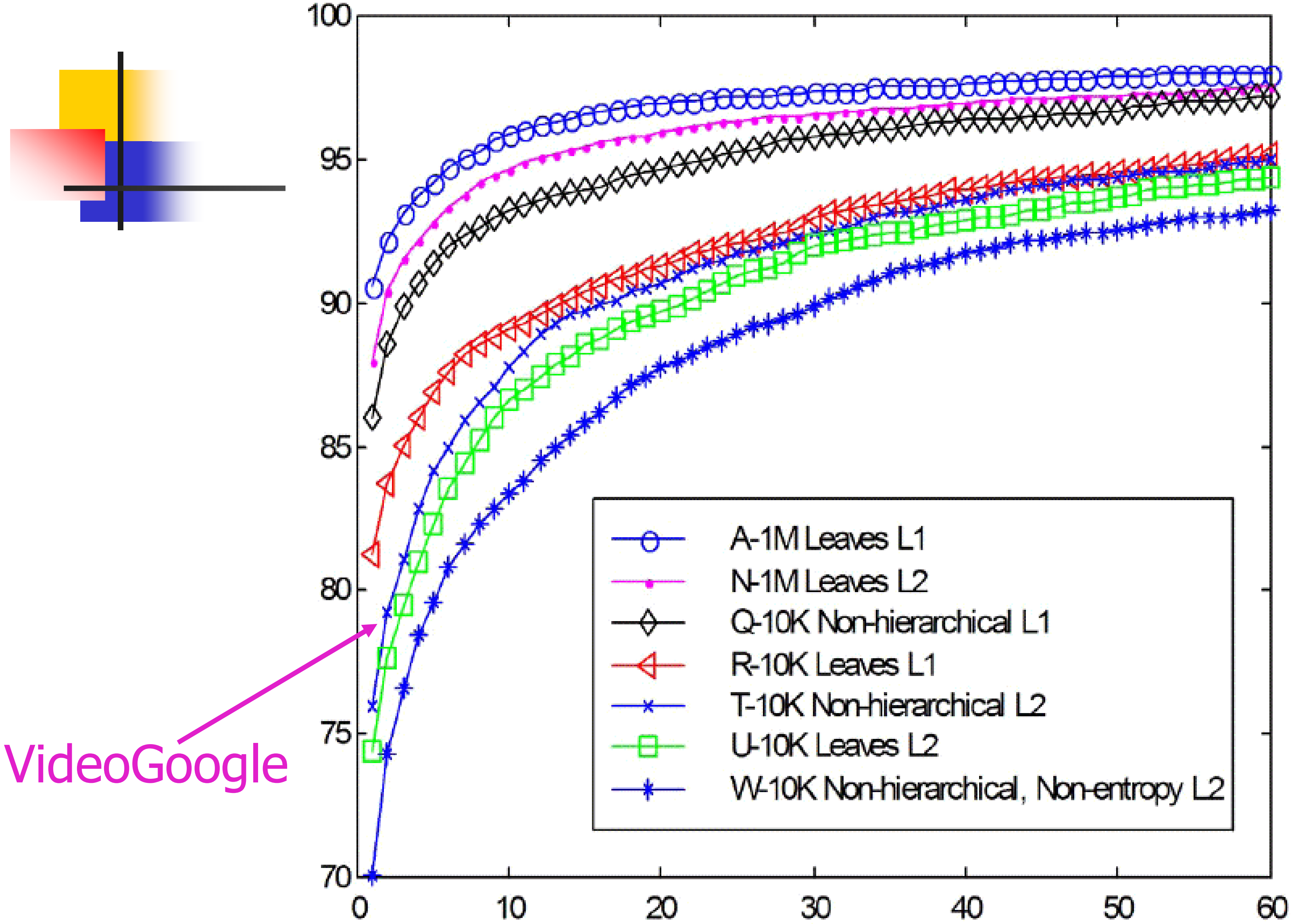


*www.cs.ualberta.ca/~vis/vision06/slides/**birs2006-nister**-index.pdf*

Ground Truth Database
6376 images
In groups of four



www.cs.ualberta.ca/~vis/vision06/slides/***birs2006-nister***-index.pdf

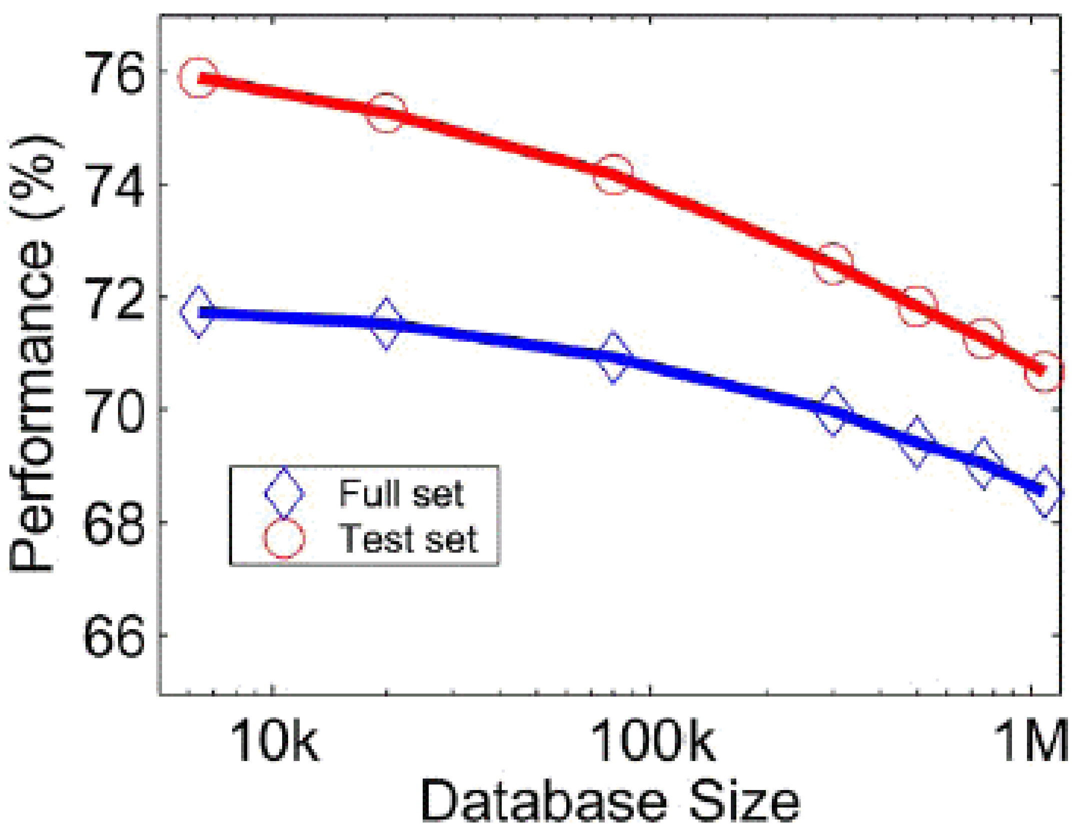


*www.cs.ualberta.ca/~vis/vision06/slides/**birs2006-nister**-index.pdf*

Me	En	No	S%	Voc-Tree	Le	Eb	Perf
A	**y/y**	**L1**	**0**	**6x10=1M**	**1**	**ir**	**90.6**
B	y/y	L1	0	6x10=1M	1	vr	90.6
C	y/y	L1	0	6x10=1M	2	ir	90.4
D	n/y	L1	0	6x10=1M	2	ir	90.4
E	y/n	L1	0	6x10=1M	2	ir	90.4
F	n/n	L1	0	6x10=1M	2	ir	90.4
G	n/n	L1	0	6x10=1M	1	ir	90.2
H	y/y	L1	m2	6x10=1M	1	ir	90.0
I	y/y	L1	0	6x10=1M	3	ir	89.9
J	y/y	L1	0	6x10=1M	4	ir	89.9
K	y/y	L1	0	6x10=1M	2	vr	89.8
L	y/y	L1	0	6x10=1M	2	ip	89.0
M	y/y	L1	m5	6x10=1M	1	ir	89.1
N	**y/y**	**L2**	**0**	**6x10=1M**	**1**	**ir**	**87.9**
O	y/y	L2	0	6x10=1M	2	ir	86.6
P	y/y	L1	110	6x10=1M	2	ir	86.5
Q	**y/y**	**L1**	**0**	**1x10K=10K**	**1**	**-**	**86.0**
R	**y/y**	**L1**	**0**	**4x10=10K**	**2**	**ir**	**81.3**
S	y/y	L1	0	4x10=10K	1	ir	80.9
T	**y/y**	**L2**	**0**	**1x10K=10K**	**1**	**-**	**76.0**
U	**y/y**	**L2**	**0**	**4x10=10K**	**1**	**ir**	**74.4**
V	y/y	L2	0	4x10=10K	2	ir	72.5
W	**n/n**	**L2**	**0**	**1x10K=10K**	**1**	**-**	**70.1**

VideoGoogle

Database Size Performance

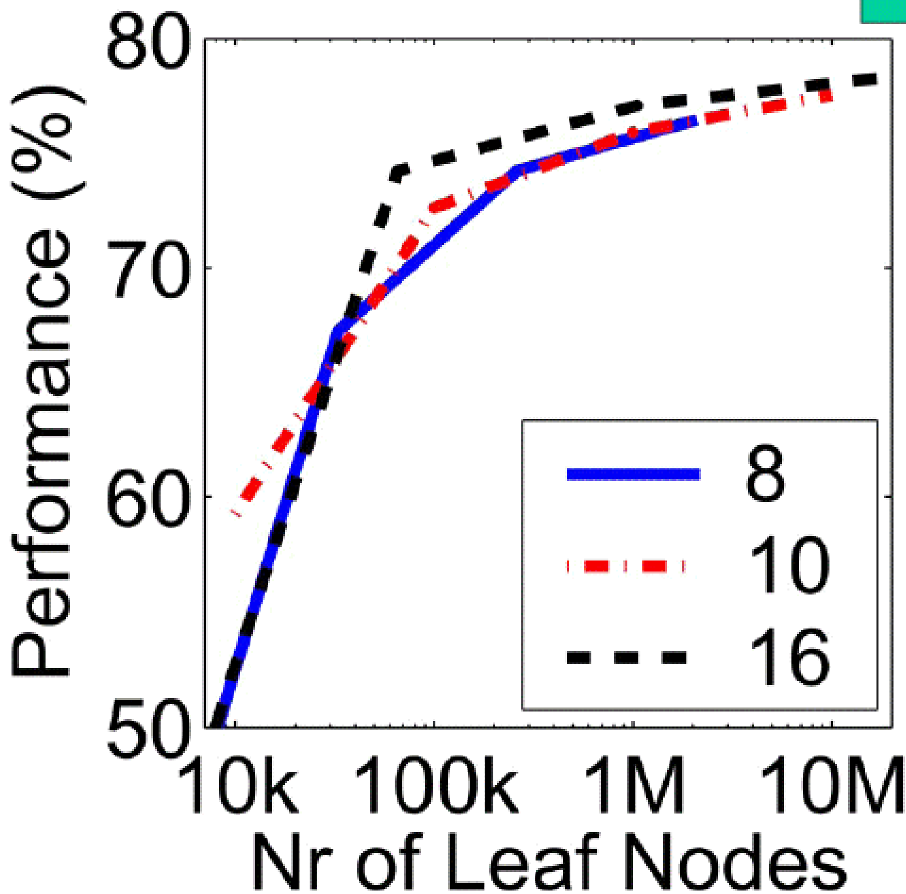


*www.cs.ualberta.ca/~vis/vision06/slides/**birs2006-nister**-index.pdf*

Size Matters

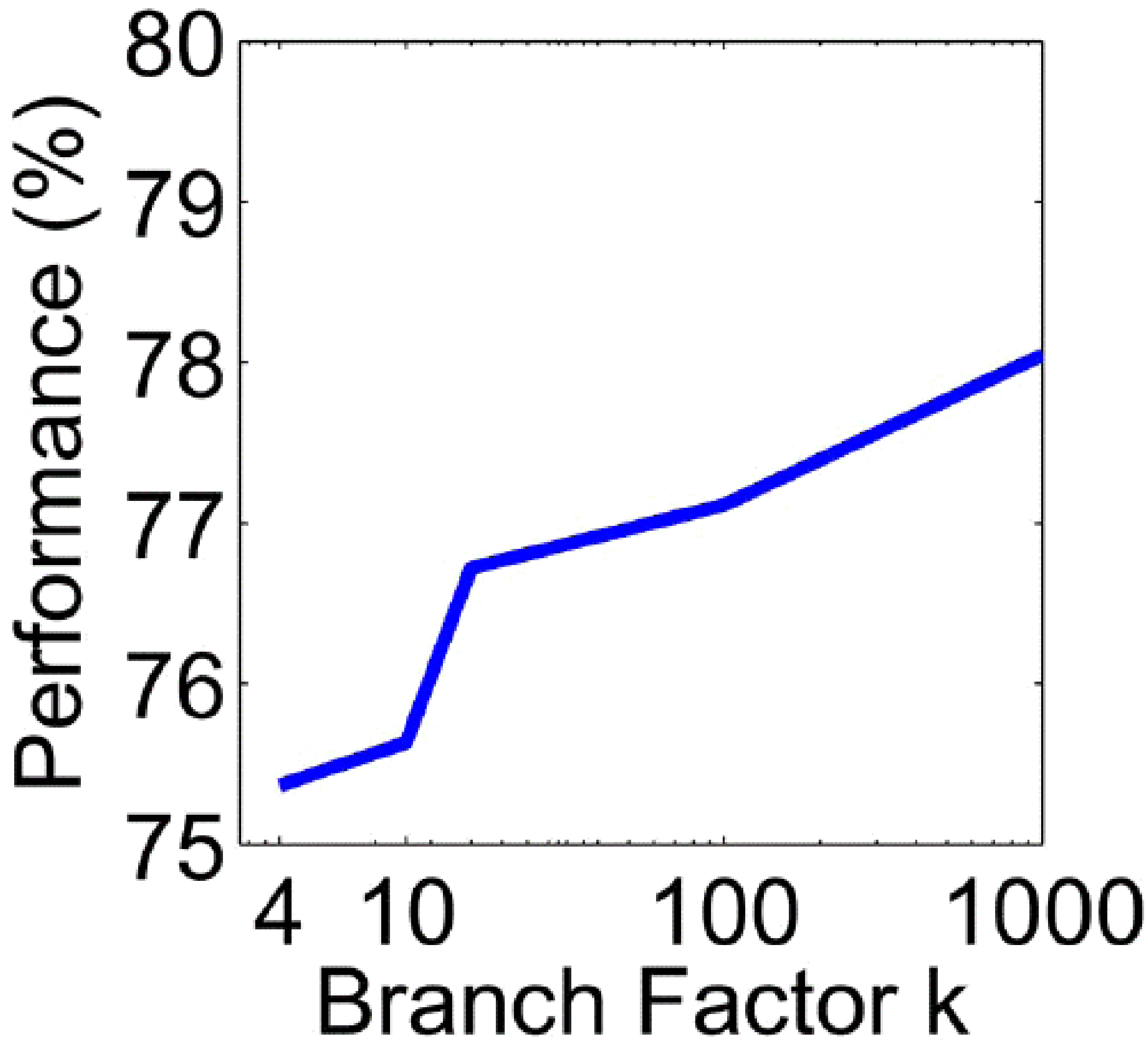
Improves Retrieval

Improves Speed

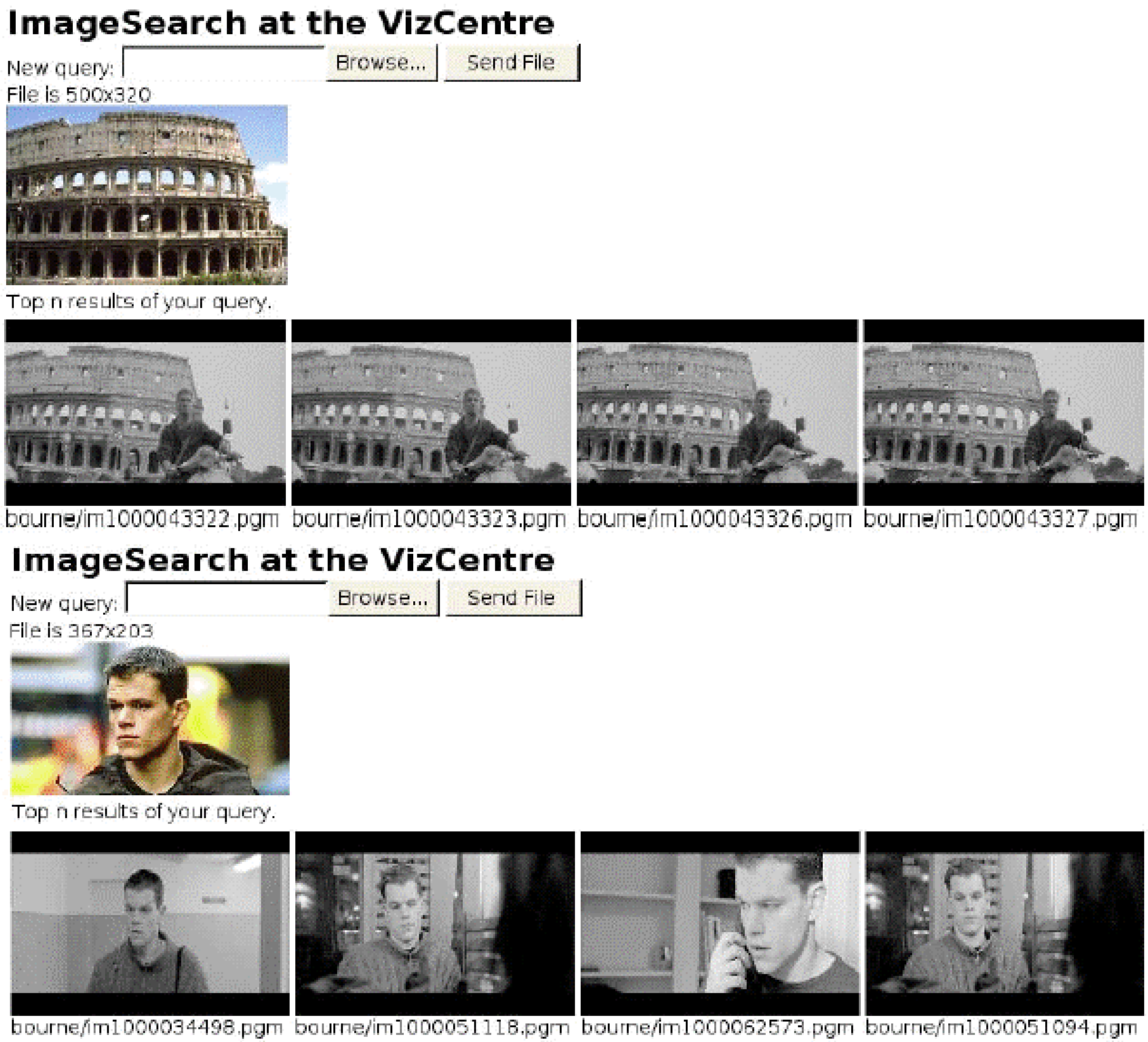


*www.cs.ualberta.ca/~vis/vision06/slides/**birs2006-nister**-index.pdf*

Given 1M leave nodes



*www.cs.ualberta.ca/~vis/vision06/slides/**birs2006-nister**-index.pdf*


ImageSearch at the VizCentre
New query:
Browse...
Send File
File is 500x320
Top n results of your query.
bourne/im1000043322.pgm
bourne/im1000043323.pgm
bourne/im1000043326.pgm
bourne/im1000043327.pgm
ImageSearch at the VizCentre
New query:
Browse...
Send File
File is 367x203
Top n results of your query.
bourne/im1000034498.pgm
bourne/im1000051118.pgm
bourne/im1000062573.pgm
bourne/im1000051094.pgm

Comments and Discussion

- Is it easy to dynamically change the tree structure?
- Is the metric system effective/accurate enough?
  - Good for rigid objects.
  - Bad for faces, animals…
- Can the metric be learned from the specific data base?

Fast Image Search for Learned Metrics

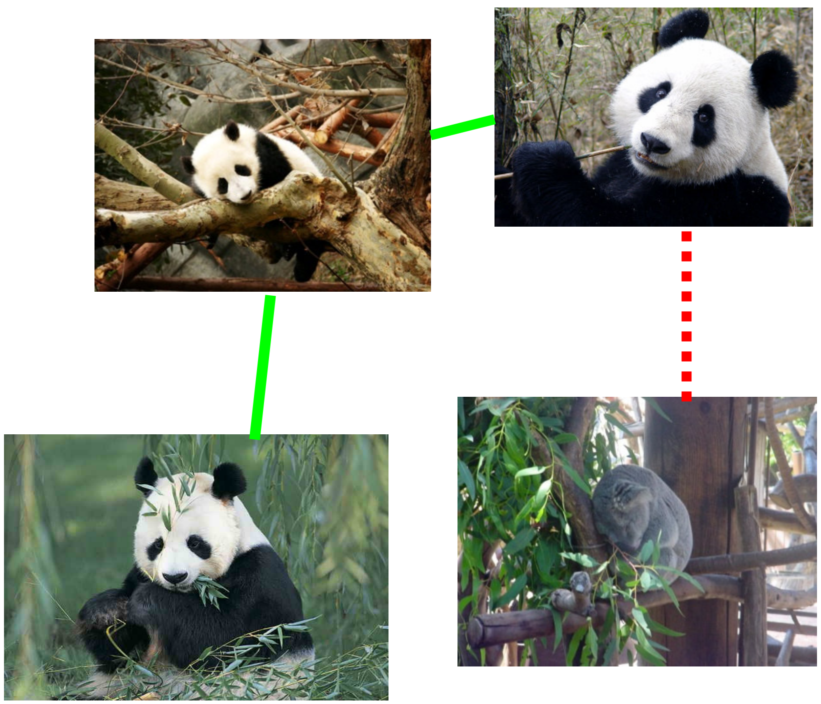
- Fast and Accurate
- Fast: Generic or Low-Dimension metric
  - Not accurate for many cases
- Accurate: *Learned* Metrics. Specific for some certain tasks.
  - No guarantee to be fast. Could be deteriorated to linear search.

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

http://www.cs.utexas.edu/~grauman/slides/jain_et_al_cvpr2008.ppt

Metric learning

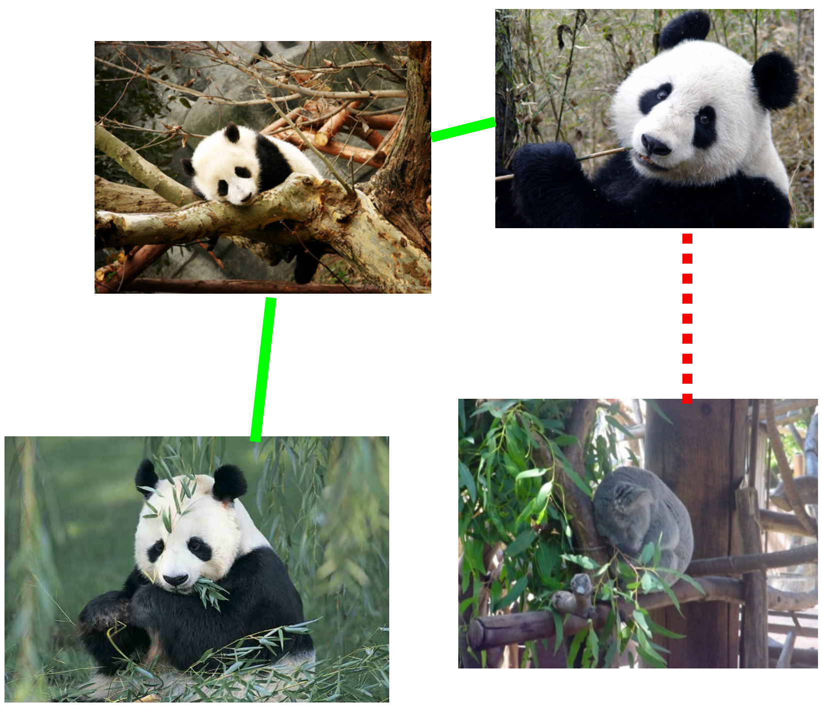


There are various ways to judge appearance/shape similarity…

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

http://www.cs.utexas.edu/~grauman/slides/jain_et_al_cvpr2008.ppt

Metric learning

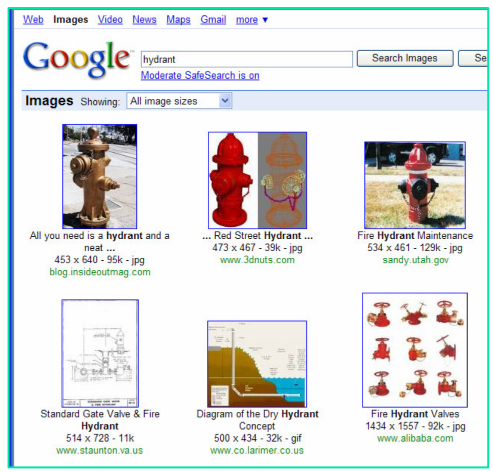


- Exploit partially labeled data and/or (dis)similarity constraints to construct more useful distance function
- Various existing techniques

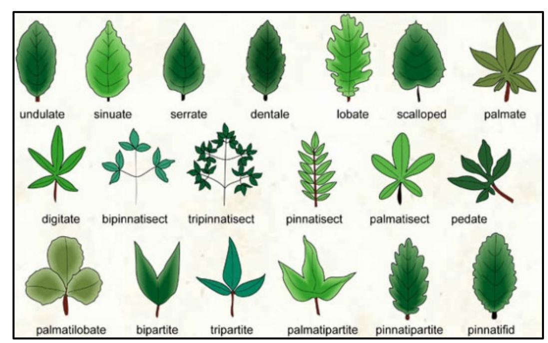
Example sources of similarity constraints

Partially labeled image databases

Detected video shots, tracked objects



User feedback



Problem-specific knowledge

Fully labeled image databases

http://www.cs.utexas.edu/~grauman/slides/jain_et_al_cvpr2008.ppt

Problem: How to guarantee fast search for a learned metric?

Exact search methods break down in high-d spaces, rely on good partitioning heuristics, and can degenerate to linear scan in worst case.

Approximate search techniques are defined only for particular "generic" metrics, e.g. Hamming distance, L_p norms, inner product.

http://www.cs.utexas.edu/~grauman/slides/jain_et_al_cvpr2008.ppt

Mahalanobis distances

- Distance parameterized by p.d. $d \times d$ 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)

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

http://www.cs.utexas.edu/~grauman/slides/jain_et_al_cvpr2008.ppt

Information-theoretic (LogDet) metric learning

- Formulation:

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

$$
D_{\ell \mathrm{d}}(A, A_0) = \mathrm{tr}(AA_0^{-1}) - \log\det(AA_0^{-1}) - d,
$$

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

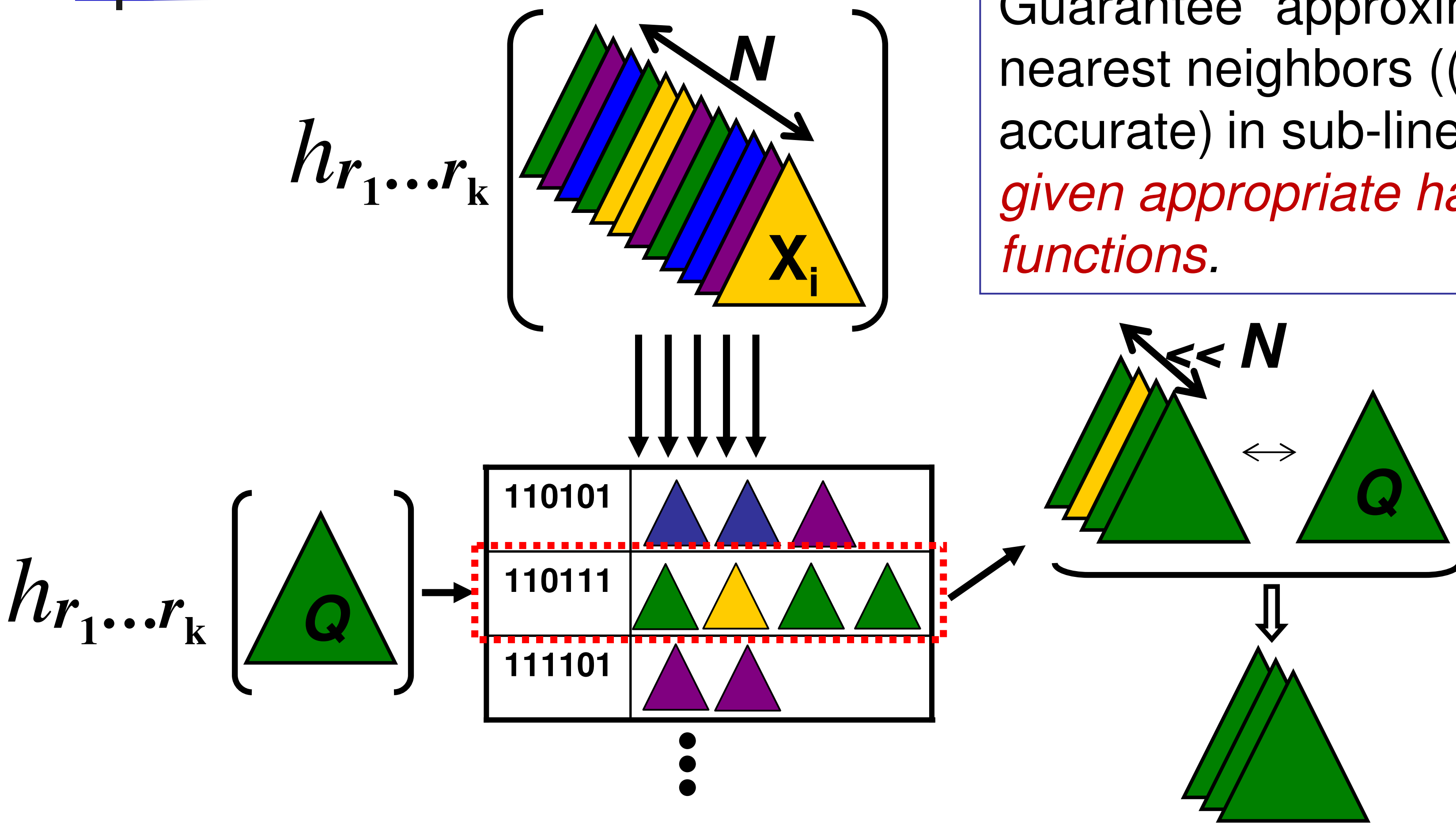
[Davis, Kulis, Jain, Sra, and Dhillon, ICML 2007]

http://www.cs.utexas.edu/~grauman/slides/jain_et_al_cvpr2008.ppt

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*.



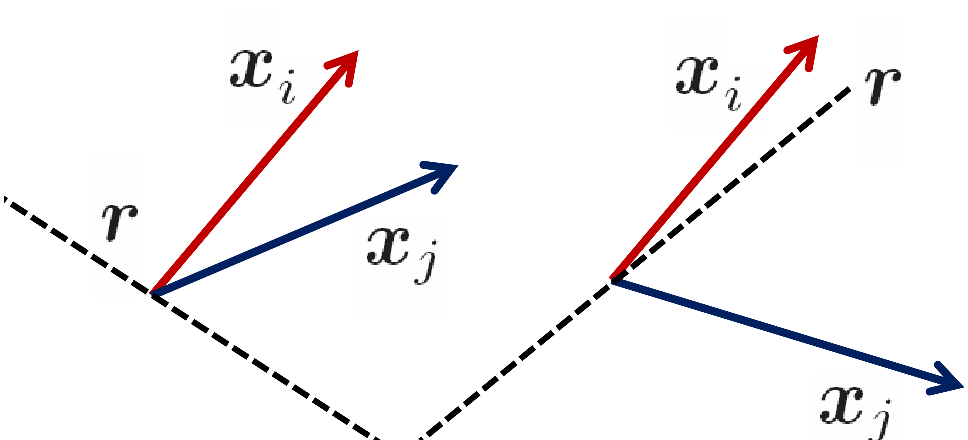
[Indyk and Motwani 1998, Charikar 2002]

http://www.cs.utexas.edu/~grauman/slides/jain_et_al_cvpr2008.ppt

LSH functions for dot products

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

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



High dot product: unlikely to split

Lower dot product: likely to split

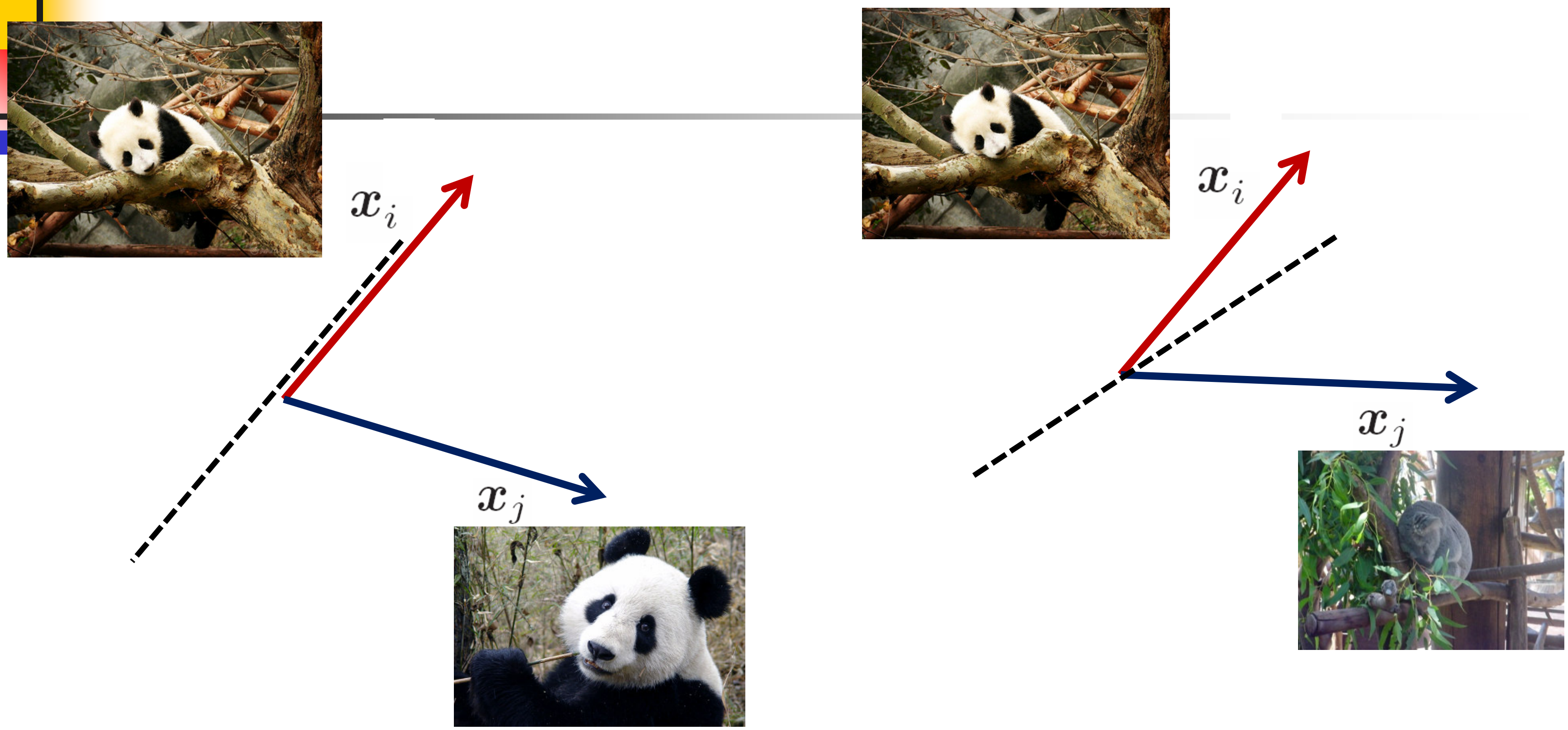
Corresponding hash function:

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

http://www.cs.utexas.edu/~grauman/slides/ja _et_al_cvpr2008.ppt

[Goemans and Williamson 1995, Charikar 2004]

LSH functions for learned metrics



It should be unlikely that a hash function will split examples like those having similarity constraints…

…but likely that it splits those having dissimilarity constraints.

http://www.cs.utexas.edu/~grauman/slides/jain_et_al_cvpr2008.ppt

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 → can't work with $r^T G x$.
- We derive an implicit update rule that simultaneously updates metric and hash function parameters.
- **Integrates** metric learning and hashing

http://www.cs.utexas.edu/~grauman/slides/jain_et_al_cvpr2008.ppt

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.

http://www.cs.utexas.edu/~grauman/slides/jain_et_al_cvpr2008.ppt

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.

http://www.cs.utexas.edu/~grauman/slides/jain_et_al_cvpr2008.ppt

Results

Object Categorization

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

Pose Estimation

Poser data, 24k dimensions, .5 million points

Patch Indexing

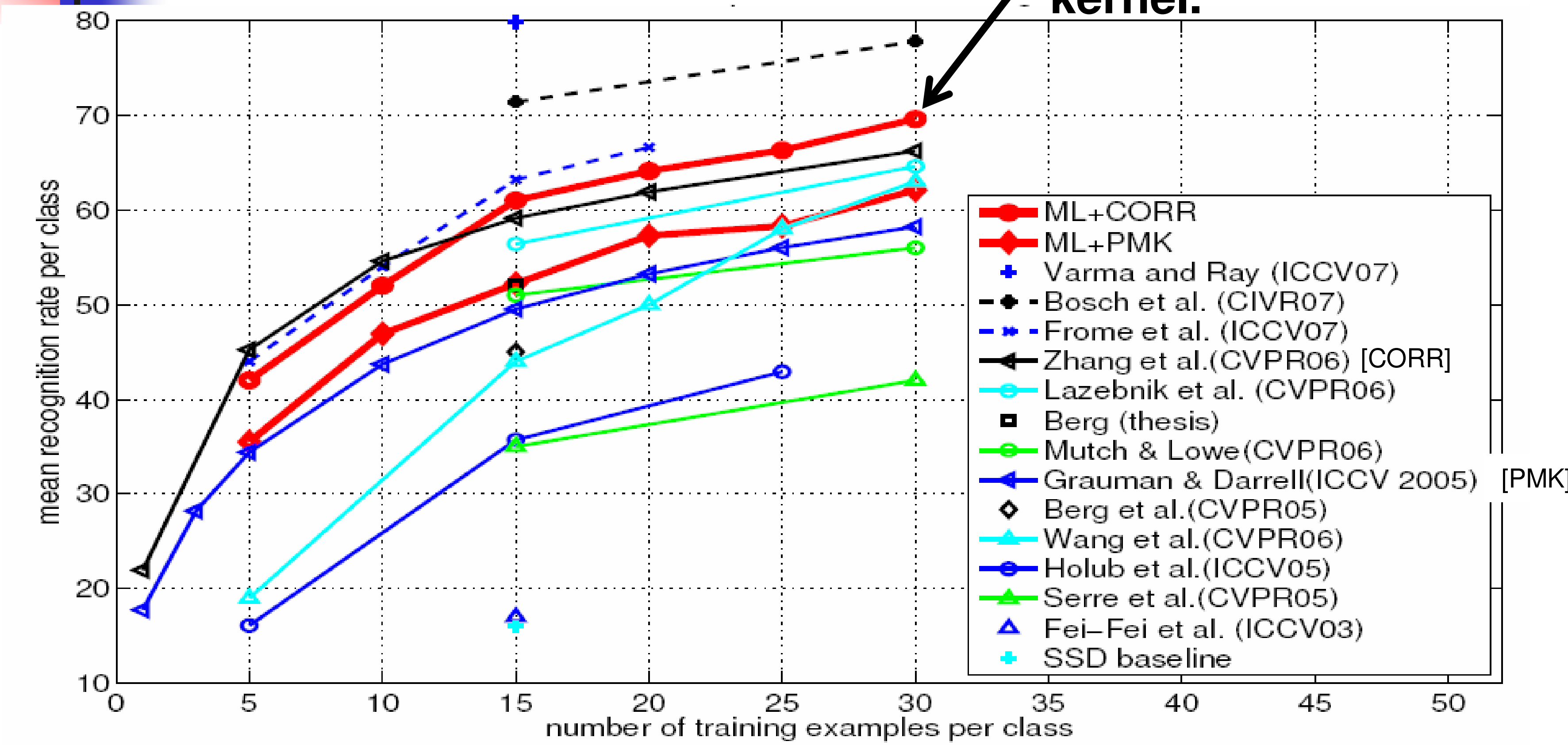
Photo Tourism data, 4096 dimensions, 300k points

http://www.cs.utexas.edu/~grauman/slides/jain_et_al_cvpr2008.ppt

Caltech-101

Results: object categorization

Best accuracy to date with a single metric / kernel.

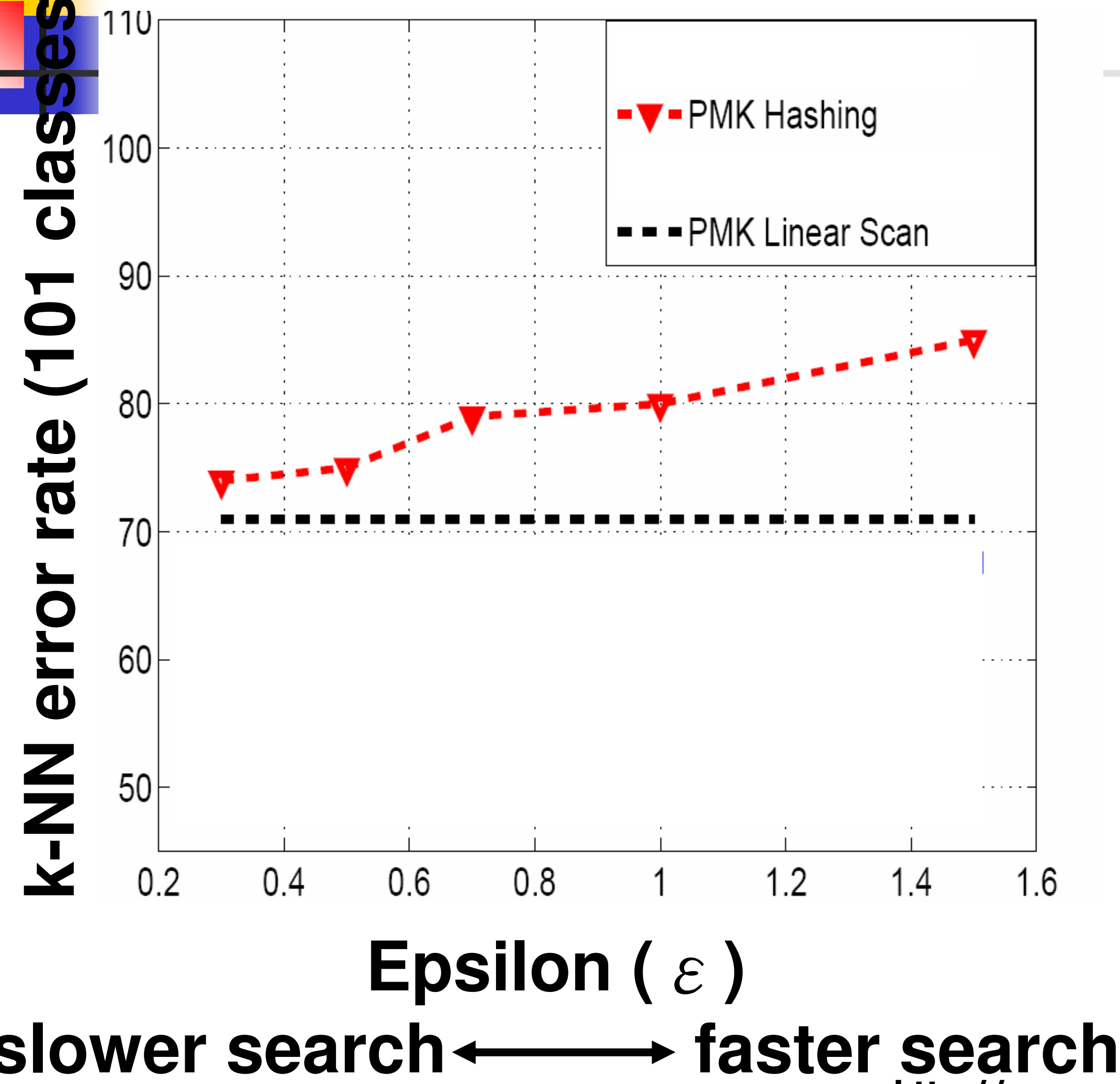


Caltech-101 database

http://www.cs.utexas.edu/~grauman/slides/jain_et_al_cvpr2008.ppt

ML = metric learning

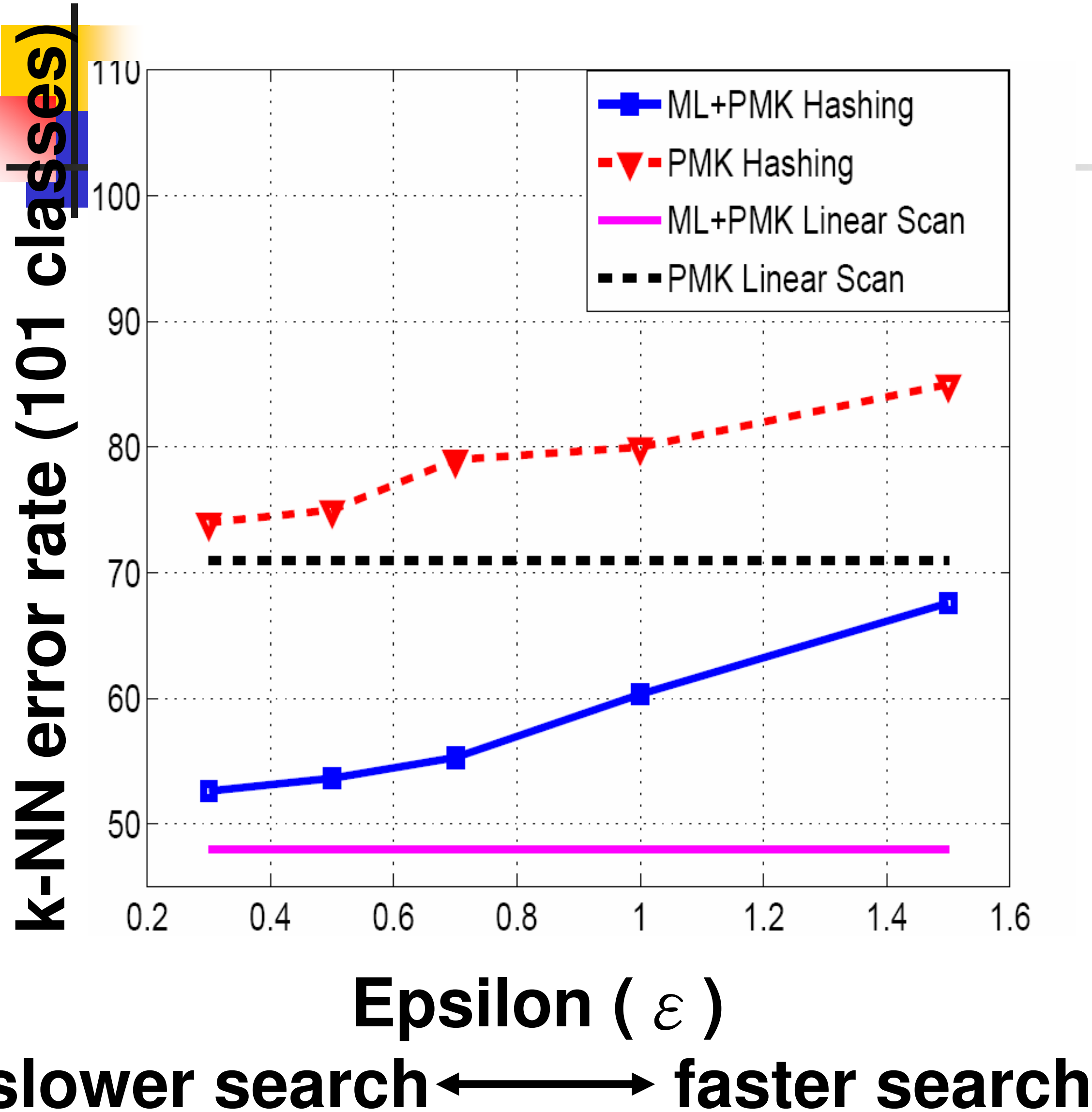
Results: object categorization



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

http://www.cs.utexas.edu/~grauman/slides/jain_et_a

Results: object categorization



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

http://www.cs.utexas.edu/~grauman/slides/jain_et_al_cvpr2008.ppt

Results: pose estimation

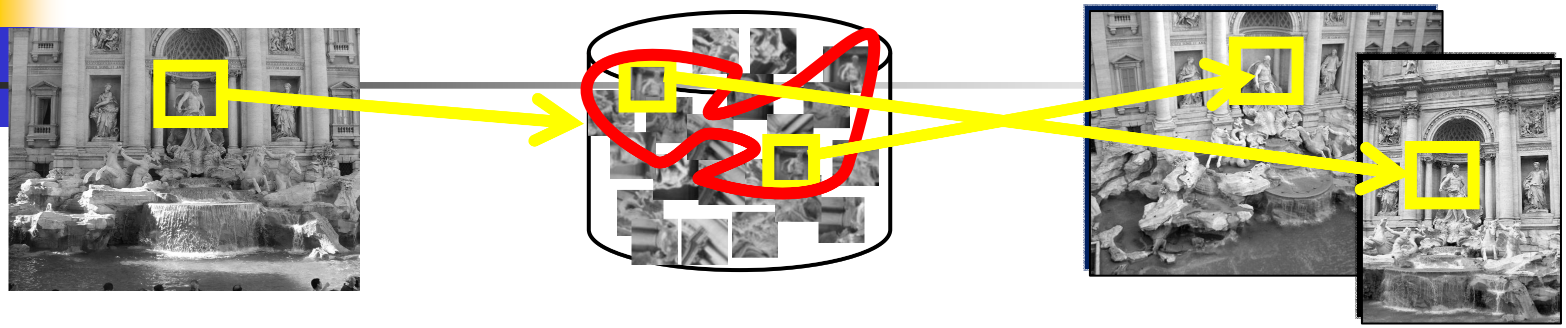
 

- 500,000 synthetic images
- Measure mean error per joint between query and NN
  - Random 2 database images: 34.5 cm between each joint
- Average query time:
  - ML linear scan: 433.25 sec
  - ML hashing: 1.39 sec

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

http://www.cs.utexas.edu/~grauman/slides/jain_et_al_cvpr2008.ppt

Results: patch indexing



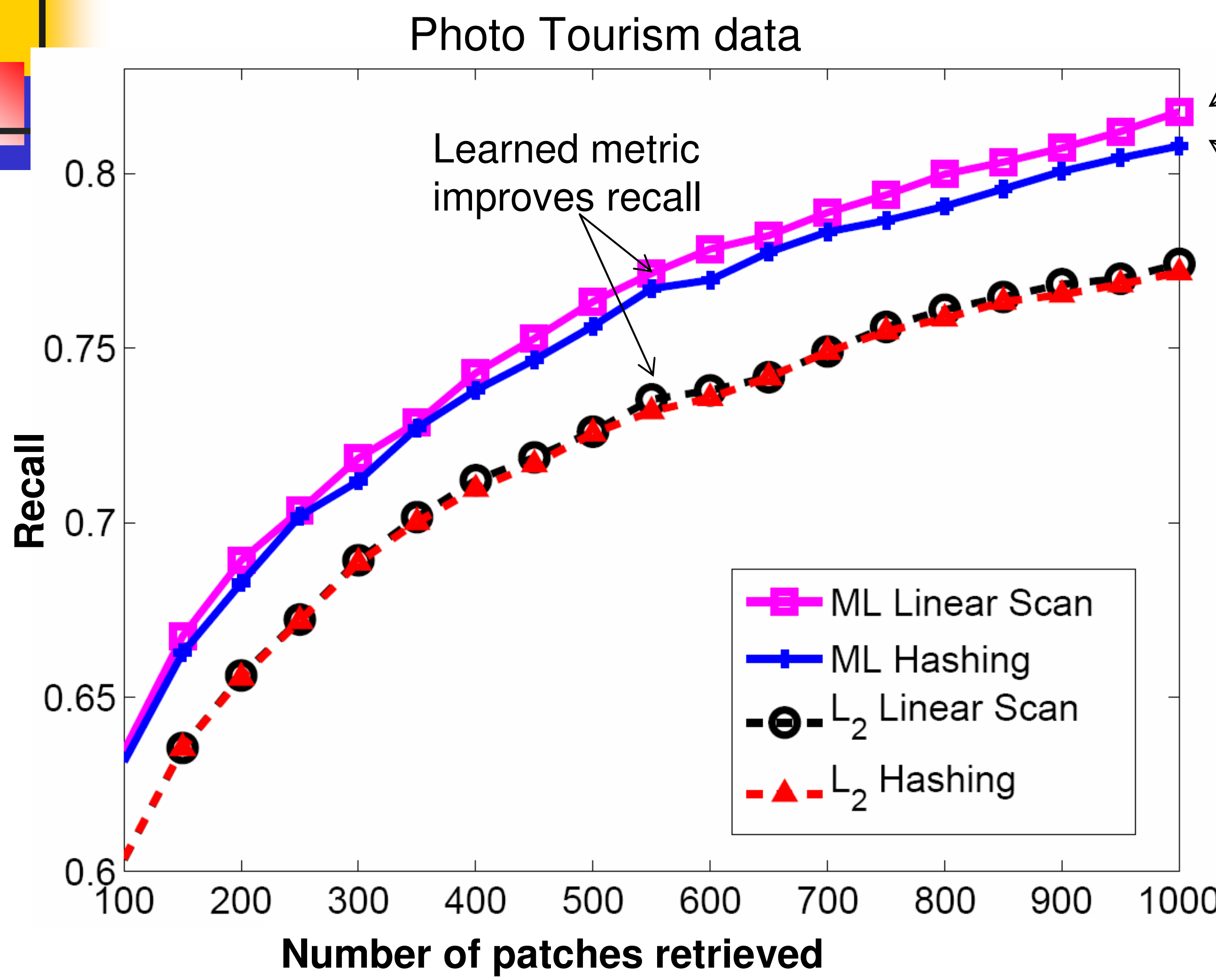
O(10^5) patches

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

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

http://www.cs.utexas.edu/~grauman/slides/jain_et_al_cvpr2008.ppt

Results: patch indexing



Search 100% of data

Search 0.8% of data

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

http://www.cs.utexas.edu/~grauman/slides/jain_et_al_cvpr2008.ppt

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.

http://www.cs.utexas.edu/~grauman/slides/jain_et_al_cvpr2008.ppt

Comments and Discussion

- Scalable to multi-millions images?
- Flexible to expand to cove more constraints?
- Hybrid system:
  - Hierarchy vocabulary tree
  - Learned metric with LSH

Conclusions

- The technologies have been into the practically usable.
- Implementation details could differentiate further.
- Find out the appealing daily life applications. $$$$$ ☺
- Machines how to evolve?
  - Teach them to ask the key questions
  - Process on the key questions.