Computer Vision and Object Recognition 2012

Instance level recognition II: Correspondence and efficient visual search

Ivan Laptev

http://www.di.ens.fr/~laptev

INRIA, WILLOW, ENS/INRIA/CNRS UMR 8548

Laboratoire d'Informatique, Ecole Normale Supérieure, Paris

With slides from: O. Chum, K. Grauman, S. Lazebnik, B. Leibe, D. Lowe, J. Philbin, J. Ponce, D. Nister, C. Schmid, N. Snavely, A. Zisserman

Outline

Part 1. Image matching and recognition with local features

- Correspondence
- Semi-local and global geometric relations
- Robust estimation – RANSAC and Hough Transform

Part 2. Going large-scale

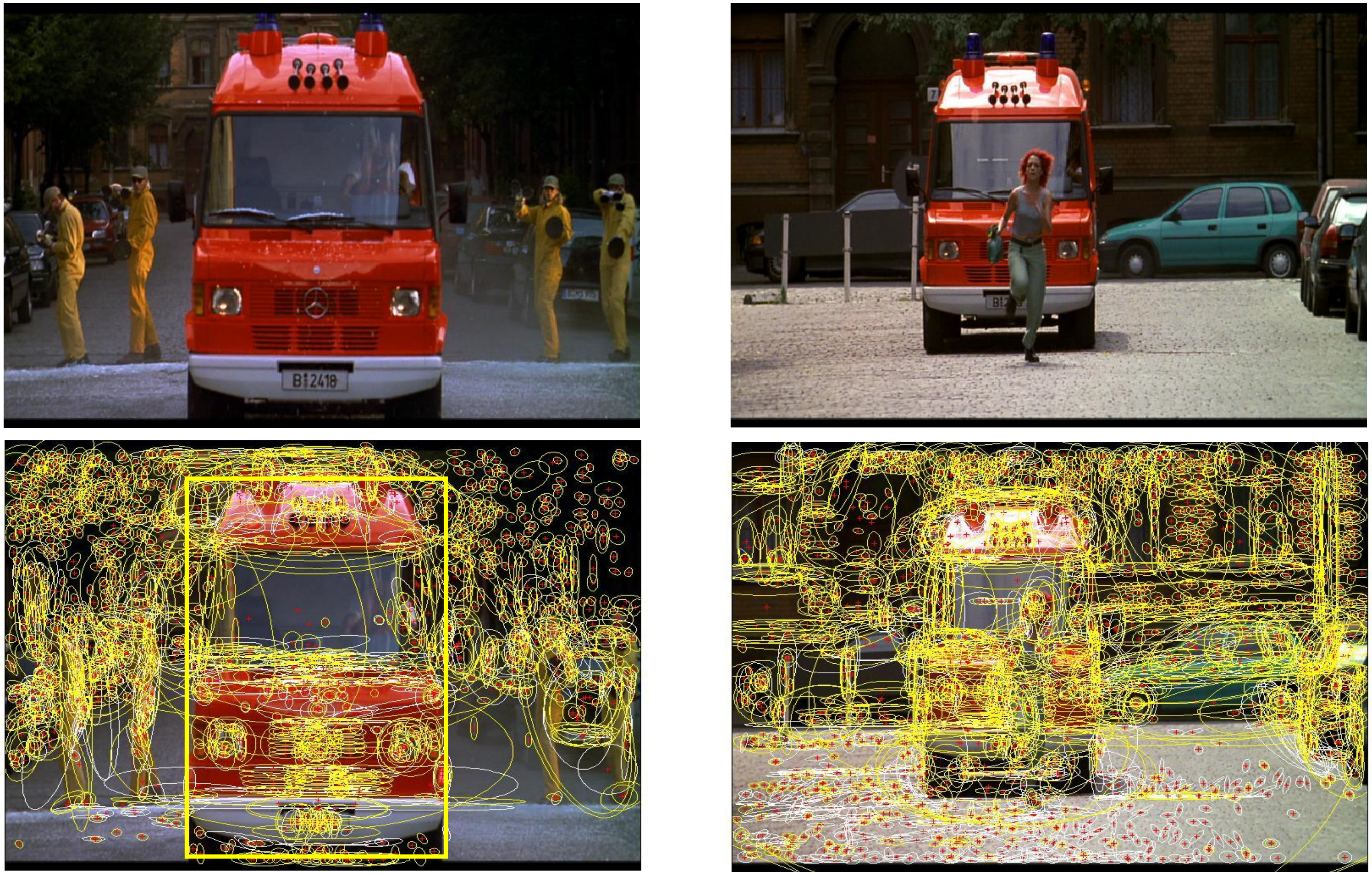
- Approximate nearest neighbour matching
- Bag-of-visual-words representation
- Efficient visual search and extensions
- Beyond bag-of-visual-words representations
- Applications

Outline

Part 2. Going large-scale

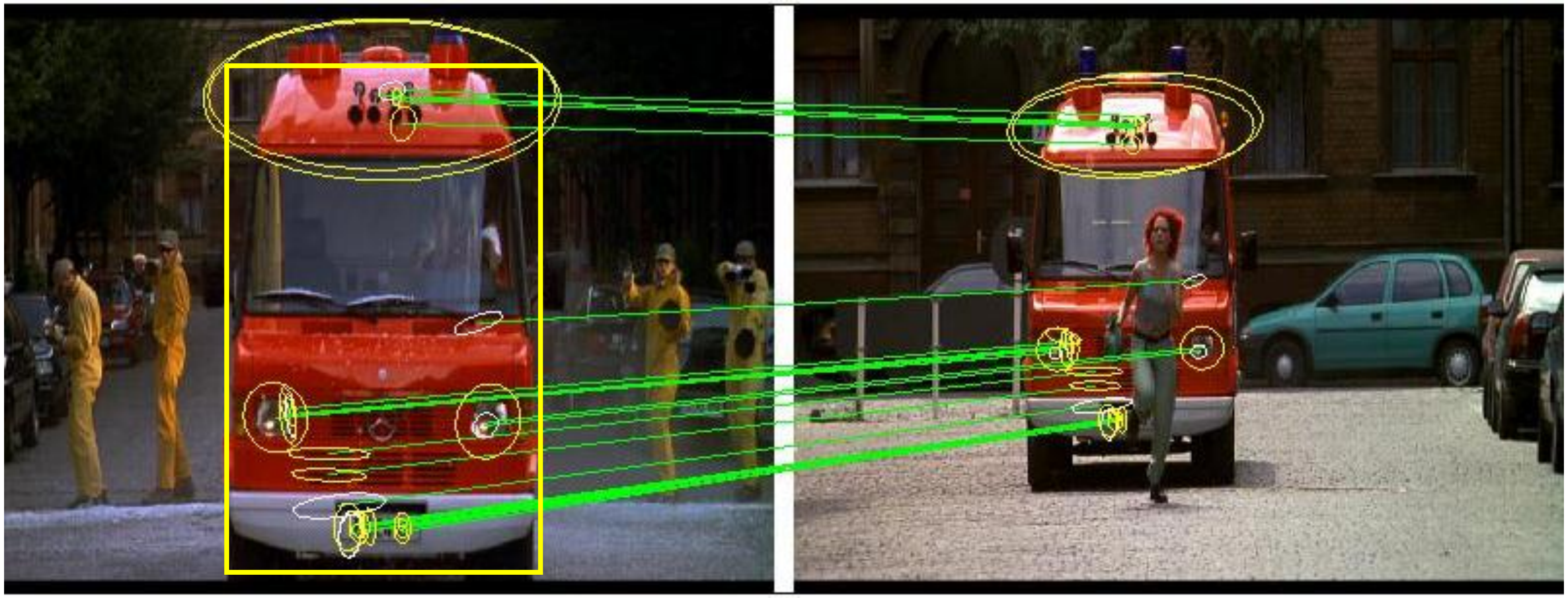
- Approximate nearest neighbour matching
- Bag-of-visual-words representation
- Efficient visual search and extensions
- Beyond bag-of-visual-words representations
- Applications

Example II: Two images again



1000+ descriptors per image

Match regions between frames using SIFT descriptors and spatial consistency



Multiple regions overcome problem of partial occlusion

Approach - review

1. Establish tentative (or putative) correspondence based on local appearance of individual features (now)

2. Verify matches based on semi-local / global geometric relations (You have just seen this).

What about multiple images?

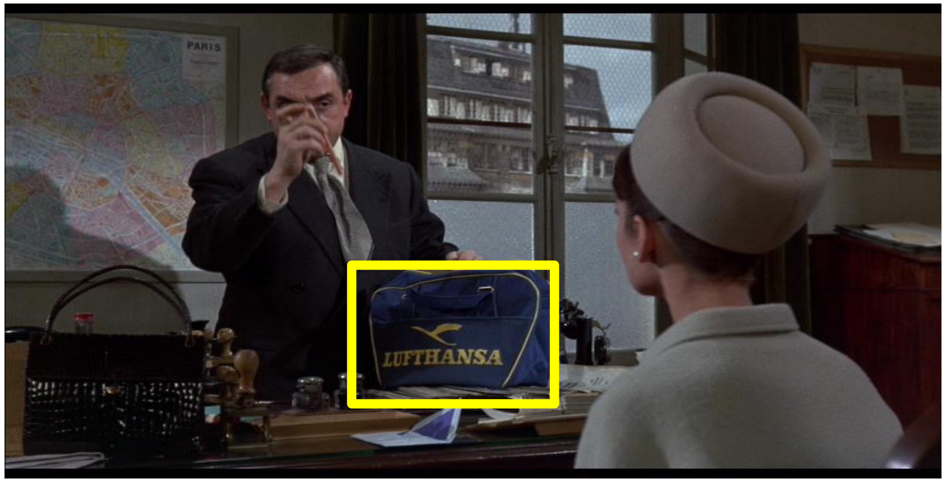
- So far, we have seen successful matching of a query image to a single target image using local features.

- How to generalize this strategy to multiple target images with reasonable complexity?

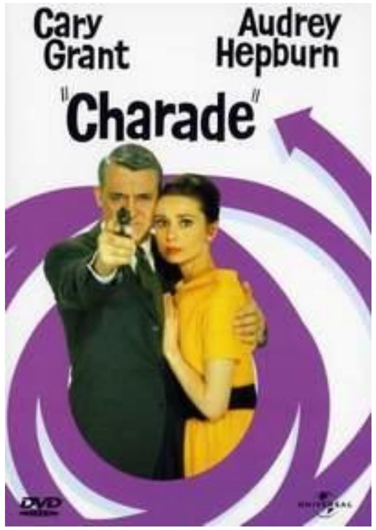
    - 10, 10^2, 10^3, …, 10^7, … 10^{10}, … images?

Example: Visual search in an entire feature length movie

Visually defined query



"Find this bag"



"Charade" [Donen, 1963]

Demo:
http://www.robots.ox.ac.uk/~vgg/research/vgoogle/index.html

History of “large scale” visual search with local regions

Schmid and Mohr ’97	– 1k images
Sivic and Zisserman’03	– 5k images
Nister and Stewenius’06	– 50k images (1M)
Philbin et al.’07	– 100k images
Chum et al.’07 + Jegou et al.’07	– 1M images
Chum et al.’08	– 5M images
Jegou et al. ’09	– 10M images
Jegou et al. ’10	– ~100M images

All on a single machine in ~ 1 second!

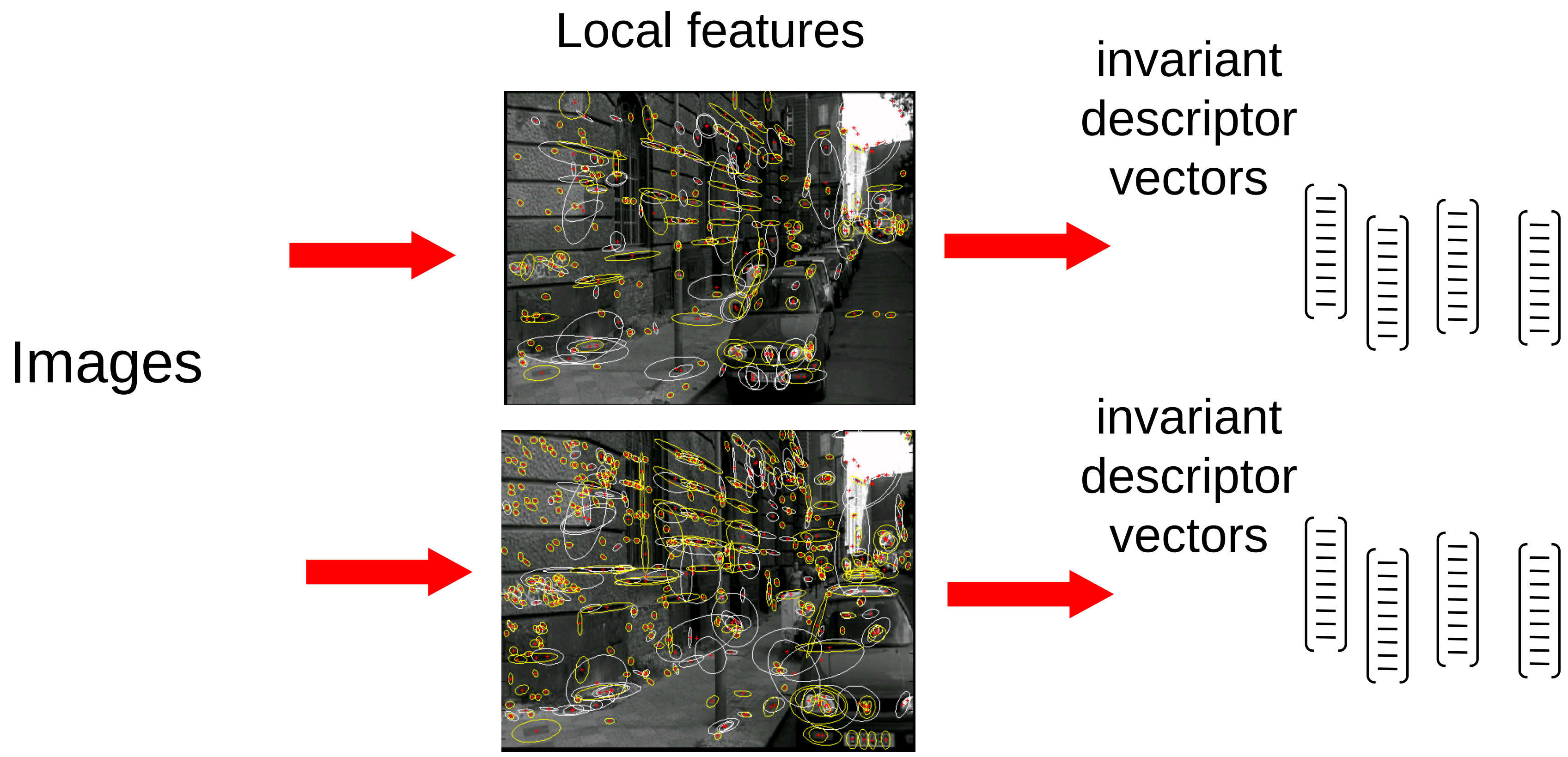
Two strategies

1. Efficient approximate nearest neighbour search on local feature descriptors.

2. Quantize descriptors into a “visual vocabulary” and use efficient techniques from text retrieval.

   (Bag-of-words representation)

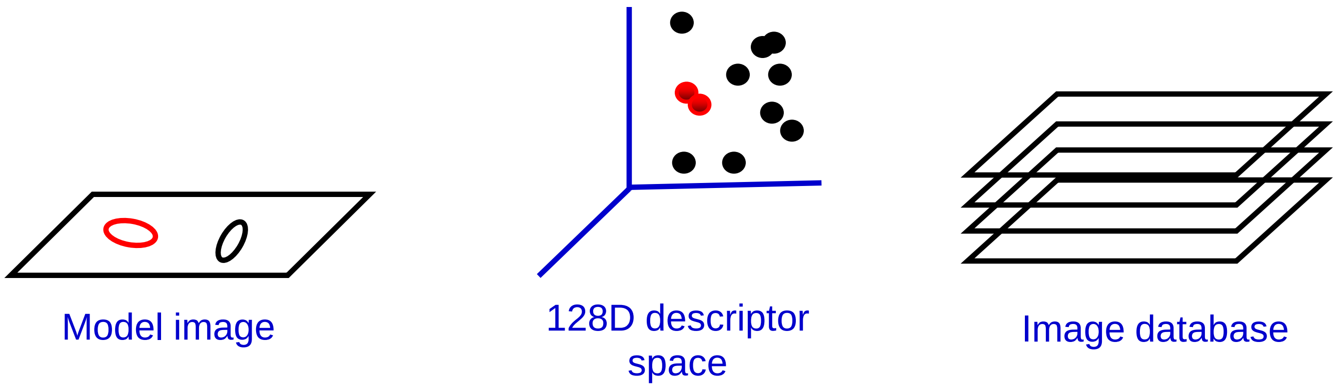
Strategy I: Efficient approximate NN search



1. Compute local features in each image independently (Part 1)
2. “Label” each feature by a descriptor vector based on its intensity (Part 1)
3. Finding corresponding features is transformed to finding nearest neighbour vectors
4. Rank matched images by number of (tentatively) corresponding regions
5. Verify top ranked images based on spatial consistency (Part 2)

Finding nearest neighbour vectors

Establish correspondences between object model image and images in the database by **nearest neighbour matching** on SIFT vectors



Solve following problem for all feature vectors, $\mathbf{x}_j \in \mathcal{R}^{128}$, in the query image:

$$\forall j \; NN(j) = \arg\min_i ||\mathbf{x}_i - \mathbf{x}_j||$$

where, $\mathbf{x}_i \in \mathcal{R}^{128}$, are features from all the database images.

Quick look at the complexity of the NN-search

N … images

M … regions per image (~1000)

D … dimension of the descriptor (~128)

Exhaustive linear search: O(M NMD)

Example:

- Matching two images (N=1), each having 1000 SIFT descriptors
  Nearest neighbors search: 0.4 s (2 GHz CPU, implemenation in C)
- Memory footprint: 1000 * 128 = 128kB / image

# of images	CPU time	Memory req.
N = 1,000 …	~7min	(~100MB)
N = 10,000 …	~1h7min	(~ 1GB)
…		
N = 10^7	~115 days	(~ 1TB)
…		
All images on Facebook:		
N = 10^{10} …	~300 years	(~ 1PB)

Nearest-neighbor matching

Solve following problem for all feature vectors, $\mathbf{x}_j$, in the query image:

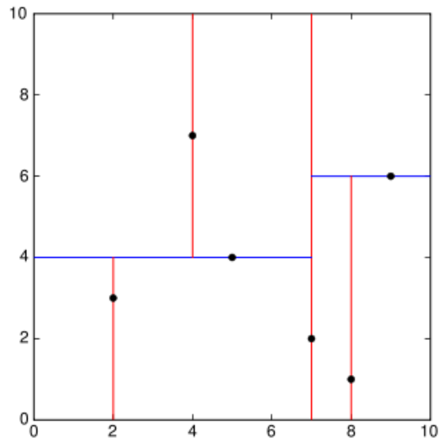
$$\forall j \; NN(j) = \arg\min_i ||\mathbf{x}_i - \mathbf{x}_j||$$

where $\mathbf{x}_i$ are features in database images.

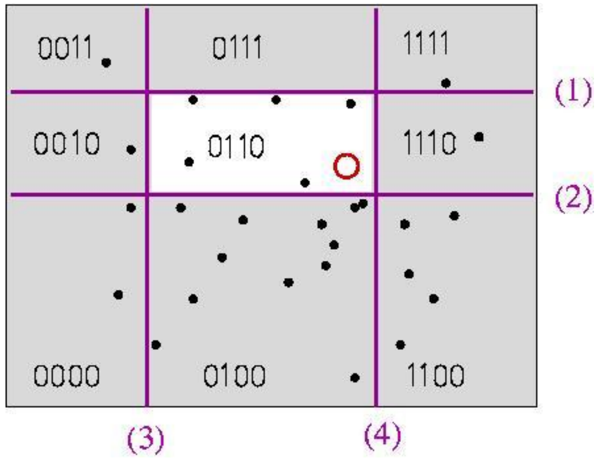
Nearest-neighbour matching is the major computational bottleneck

- Linear search performs *dn* operations for *n* features in the database and *d* dimensions
- No exact methods are faster than linear search for d>10
- Approximate methods can be much faster, but at the cost of missing some correct matches. Failure rate gets worse for large datasets.

Indexing local features: approximate nearest neighbor search



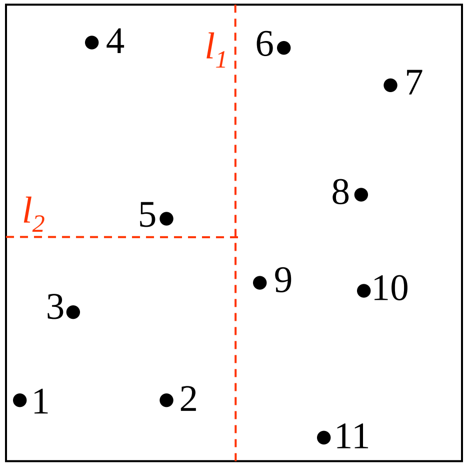
Best-Bin First (BBF), a variant of k-d trees that uses priority queue to examine most promising branches first [Beis & Lowe, CVPR 1997]

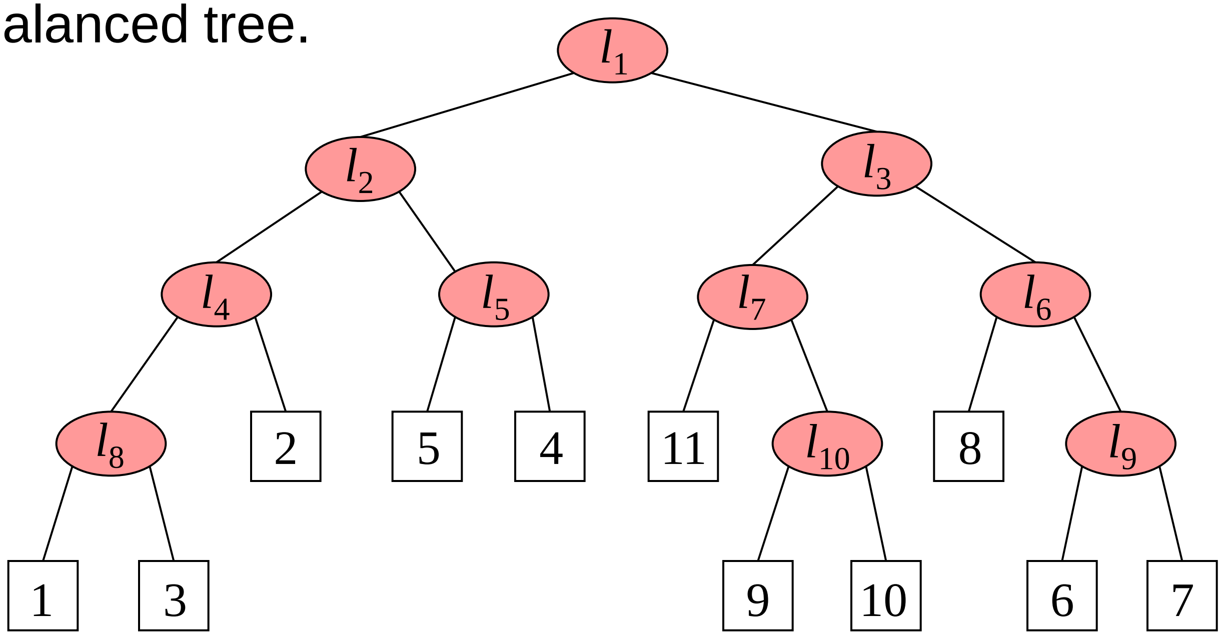


Locality-Sensitive Hashing (LSH), a randomized hashing technique using hash functions that map similar points to the same bin, with high probability [Indyk & Motwani, 1998]

K-d tree

- K-d tree is a binary tree data structure for organizing a set of points in a K-dimensional space.
- Each internal node is associated with an axis aligned hyper-plane splitting its associated points into two sub-trees.
- Dimensions with high variance are chosen first.
- Position of the splitting hyper-plane is chosen as the mean/median of the projected points – balanced tree.

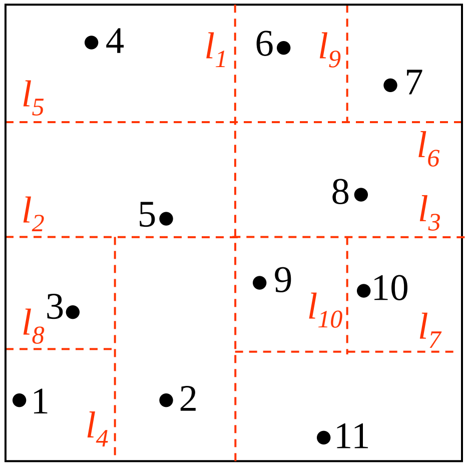


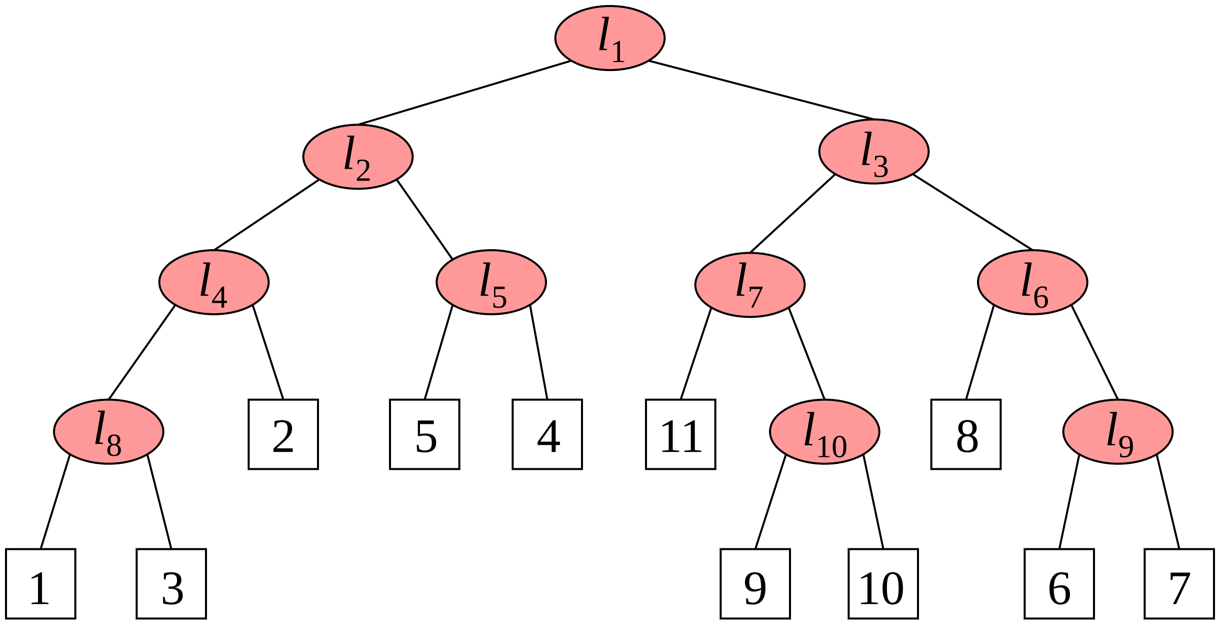


Images: Anna Atramentov

K-d tree construction

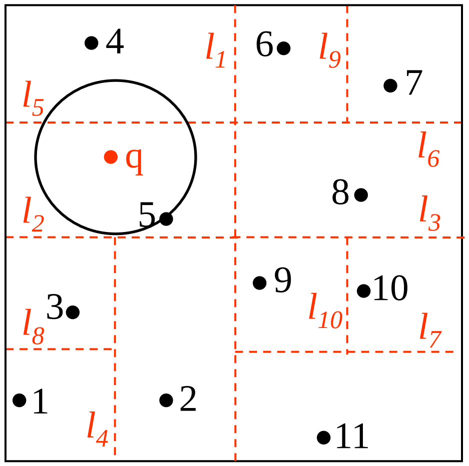
Simple 2D example

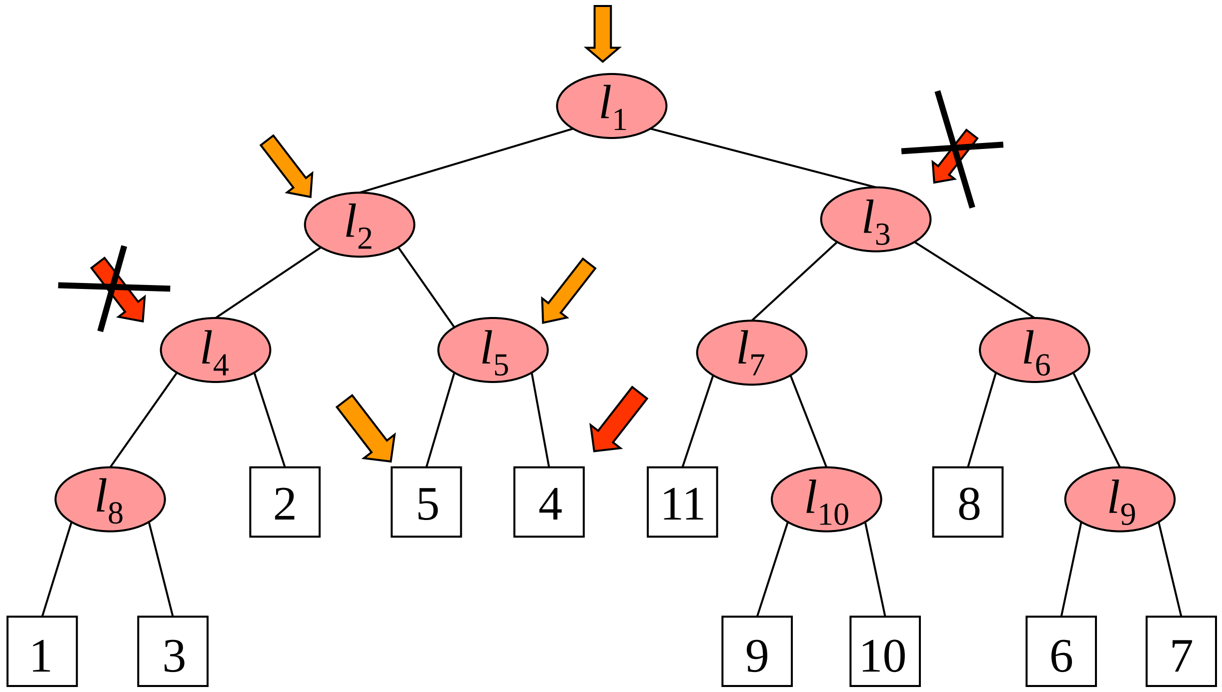




Slide credit: Anna Atramentov

K-d tree query





Slide credit: Anna Atramentov

K-d tree: Backtracking

Backtracking is necessary as the true nearest neighbor may not lie in the query cell.

But in some cases, almost all cells need to be inspected.

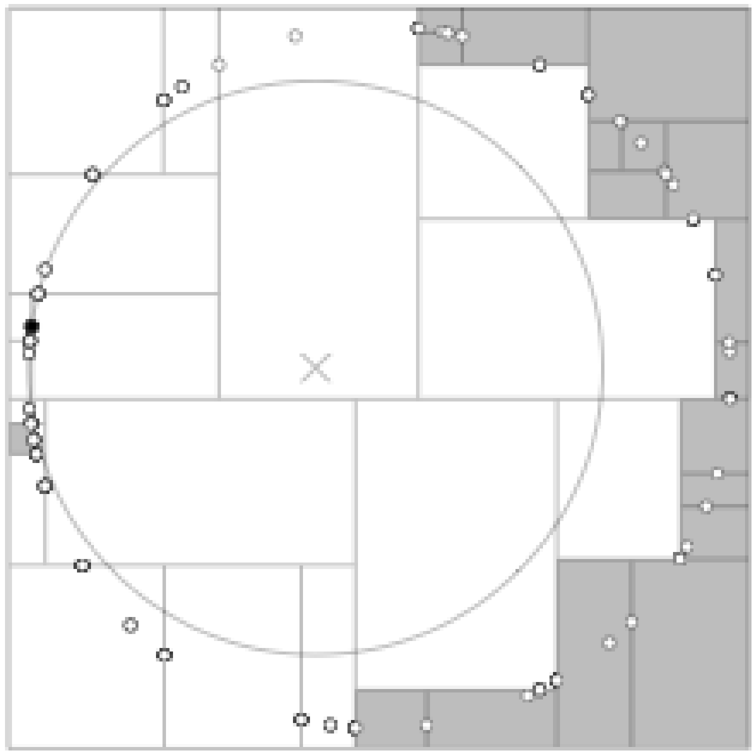


Figure 6.6

A bad distribution which forces almost all nodes to be inspected.

Figure: A. Moore

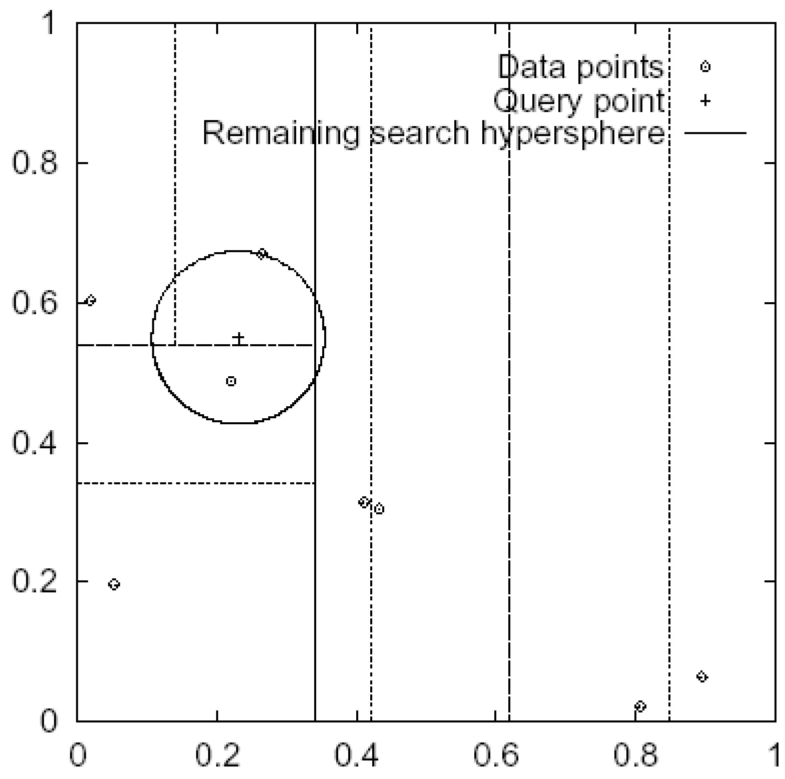
Solution: Approximate nearest neighbor K-d tree

Key ideas:

• Search k-d tree bins in order of distance from query

• Requires use of a priority queue

• Limit the number of neighbouring k-d tree bins to explore: only approximate NN is found

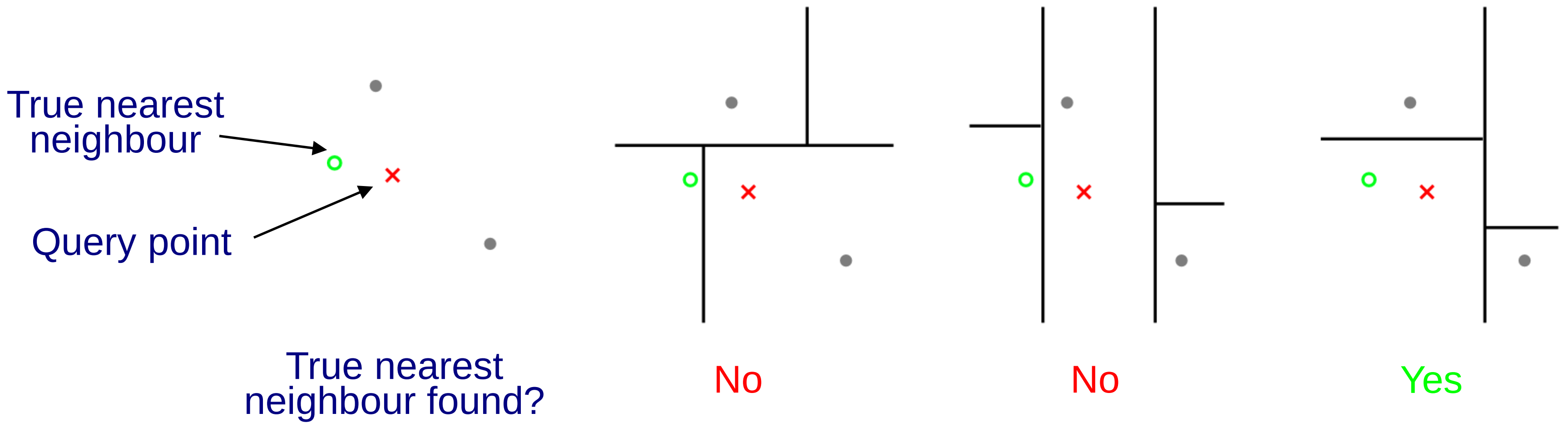


• Reduce the boundary effects by randomization

Randomized K-d trees

- How to choose the dimension to split and the splitting point?
  - Pick dimension with the highest variance
  - Split at the mean/median

- Multiple randomized trees increase the chances of finding nearby points



[Silpa-Anan and Hartley 2008, Philbin et al. 2007]

Approximate NN search using a randomized forest of K-d trees: Algorithm summary

1. Descent all (typically 8) trees to the leaf node

2. Search k-d tree bins in order of distance from query
   - Distance between the query and the bin is defined as the minimum distance between the query and any point on the bin boundary

   - Requires the use of a priority queue:
     - > During lookup an entry is added to the priority queue about the option not taken
     - > For multiple trees, the queue is shared among the trees

   - Limit the number of neighbouring K-d tree bins to explore (parameter of the algorithm, typically set to 512)

Randomized K-d trees: discussion

- Find approximate nearest neighbor in O(logN) time, where N is the number of data points.
- Increased memory requirements: needs to store multiple (~8) trees
- Good performance in practice for recognition problems (NN-search for SIFT descriptors and image patches).
- Code available online:

  http://people.cs.ubc.ca/~mariusm/index.php/FLANN/FLANN

Variation: K-means tree [Muja&Lowe, 2009]

- Partition of the space is determined by recursive application of k-means clustering.
- Cell boundaries are not axis aligned, but given by the set of cluster centers.
- Also called “tree structured vector quantization”.
- Finding nearest neighbor to a query point involves recursively finding nearest cluster center.
- Look-up complexity O(logN)
- Also used for vocabulary quantization (see later) [Nister&Stewenius’06]

Example

3-nary tree construction:

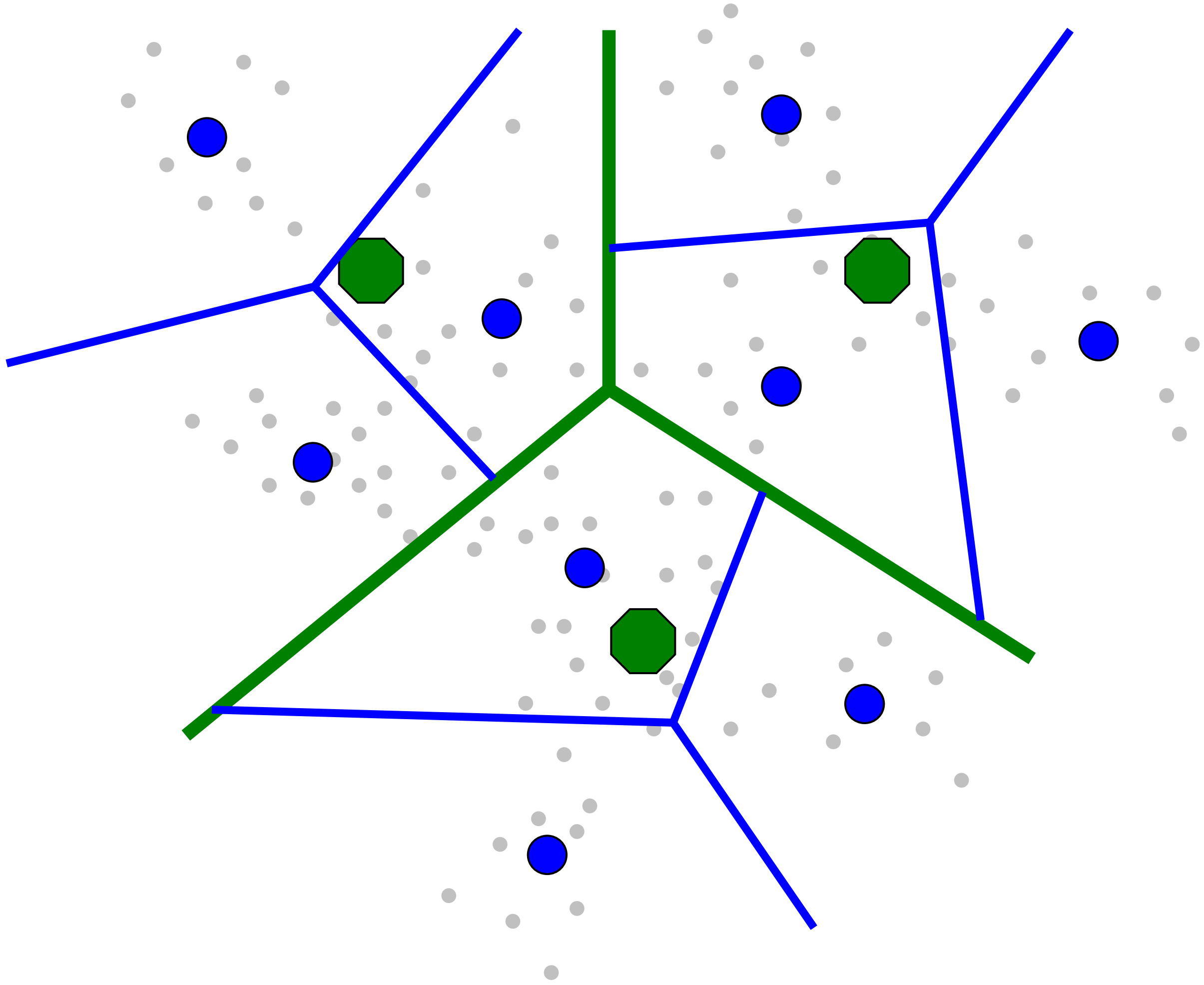


Figure credit: David Nister

Example

Query look-up:

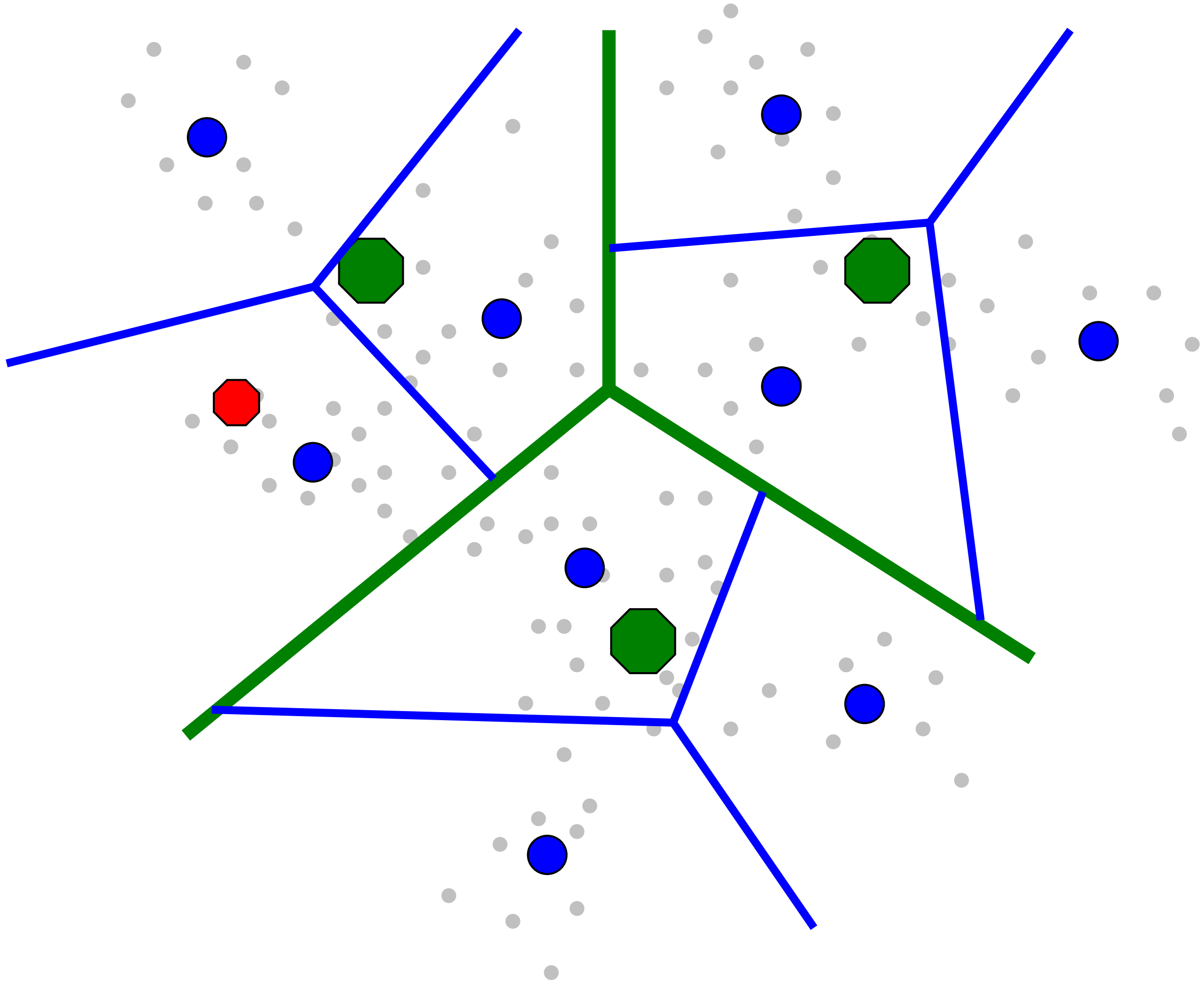
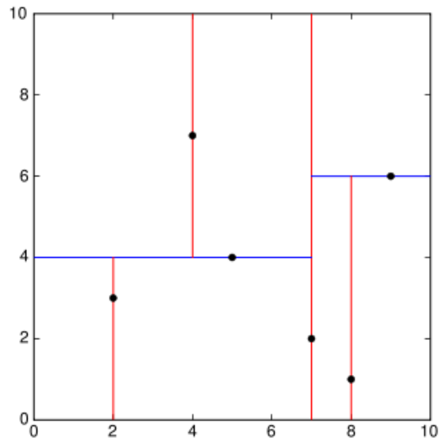
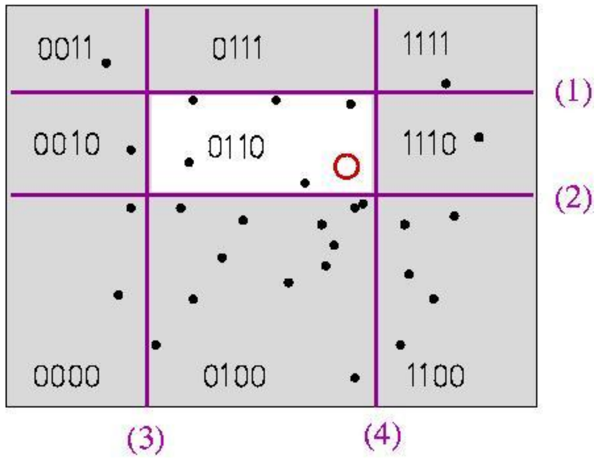


Figure credit: David Nister

Indexing local features: approximate nearest neighbor search



Best-Bin First (BBF), a variant of k-d trees that uses priority queue to examine most promising branches first [Beis & Lowe, CVPR 1997]



Locality-Sensitive Hashing (LSH), a randomized hashing technique using hash functions that map similar points to the same bin, with high probability [Indyk & Motwani, 1998]

Locality Sensitive Hashing (LSH)

Idea: construct hash functions $g: R^d \rightarrow Z^k$ such that

for any points p,q:

If $||p-q|| \leq r$, then $\Pr[g(p)=g(q)]$ is "high" or "not-so-small"
If $||p-q|| > cr$, then $\Pr[g(p)=g(q)]$ is "small"

Example of g: linear projections

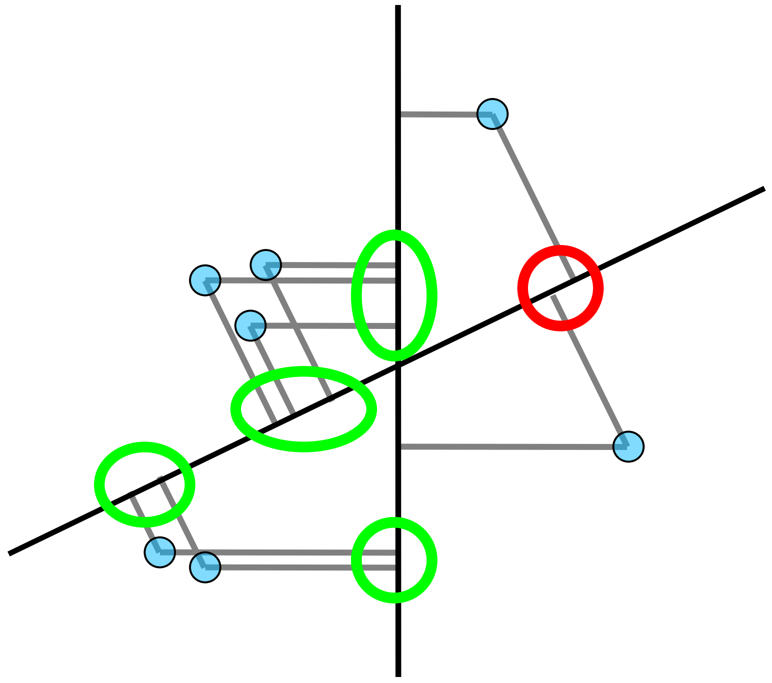
$g(p)=<h_1(p),h_2(p),\ldots,h_k(p)>$, where $h_{X,b}(p)=\lfloor (p*X+b)/w \rfloor$

$\lfloor . \rfloor$ is the "floor" operator.
X_i are sampled from a Gaussian.
w is the width of each quantization bin.
b is sampled from uniform distr. $[0,w]$.

[Datar-Immorlica-Indyk-Mirrokni'04]

Locality Sensitive Hashing (LSH)

- Choose a random projection
- Project points
- Points close in the original space remain close under the projection
- Unfortunately, converse not true



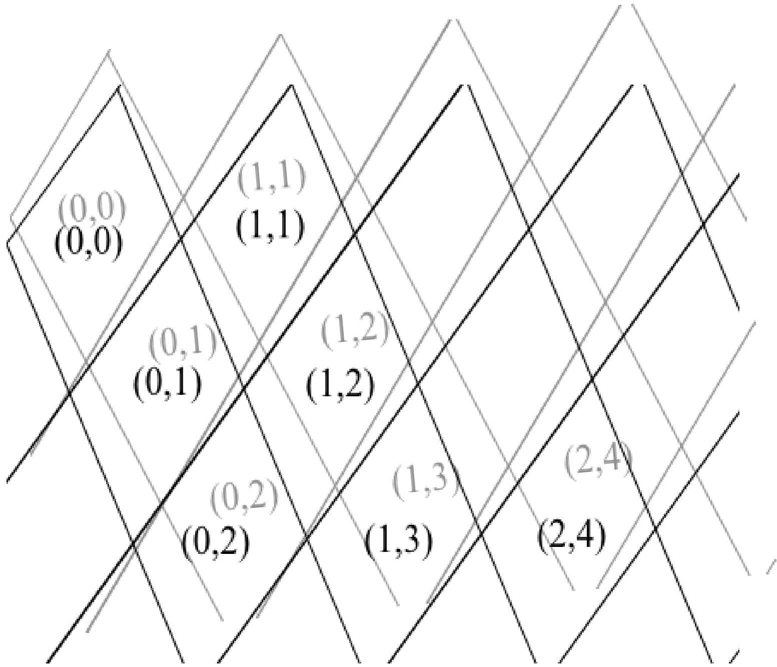
- Answer: use multiple quantized projections which define a high-dimensional "grid"

Slide: Philbin, Chum, Isard, Zissrman

Locality Sensitive Hashing (LSH)

- Cell contents can be efficiently indexed using a hash table

- Repeat to avoid quantization errors near the cell boundaries



- Point that shares at least one cell = potential candidate

- Compute distance to all candidates

Slide: Philbin, Chum, Isard, Zissrman

LSH: discussion

In theory, query time is O(kL), where k is the number of projections and L is the number of hash tables, i.e. independent of the number of points, N.

In practice, LSH has high memory requirements as large number of projections/hash tables are needed.

Code and more materials available online:

http://www.mit.edu/~andoni/LSH/

Hashing functions could be also data-dependent (PCA) or learnt from labeled point pairs (close/far).

Y. Weiss, A. Torralba, and R. Fergus, “Spectral hashing,” in *NIPS, 2008.*

R. Salakhutdinov and G. Hinton, “Semantic Hashing,” ACM SIGIR, 2007.

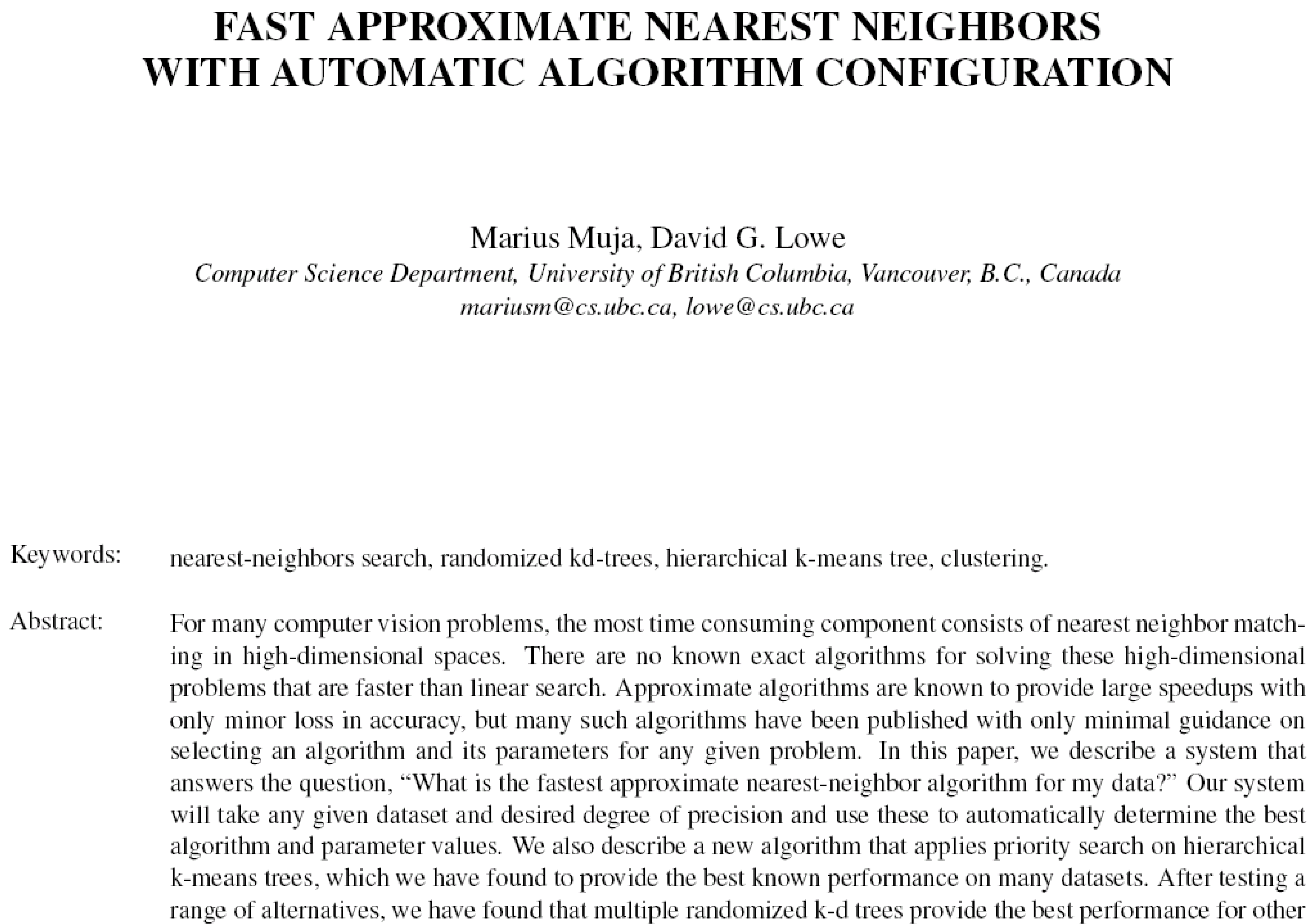
See also:

http://cobweb.ecn.purdue.edu/~malcolm/yahoo Slaney2008(LSHTutorialDraft).pdf

http://www.sanjivk.com/EECS6898/ApproxNearestNeighbors_2.pdf

Comparison of approximate NN-search methods

http://www.cs.ubc.ca/~lowe/papers/09muja.pdf



FAST APPROXIMATE NEAREST NEIGHBORS WITH AUTOMATIC ALGORITHM CONFIGURATION

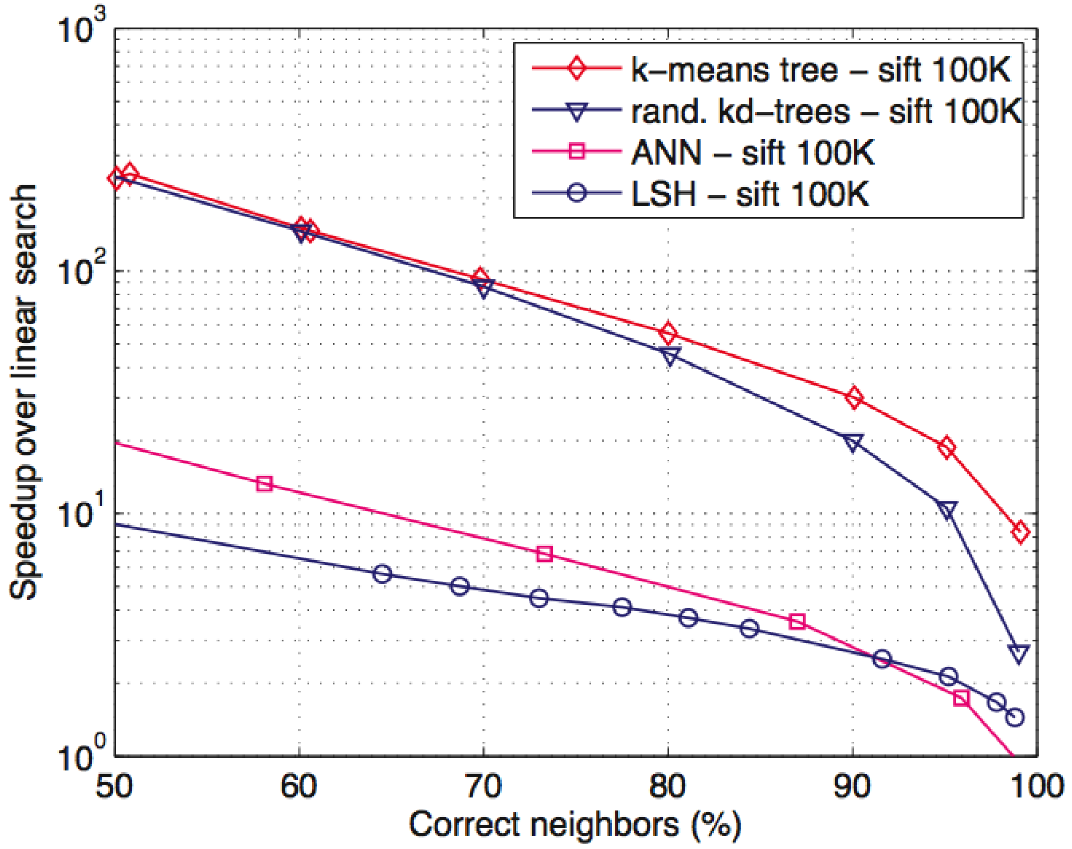
Marius Muja, David G. Lowe
Computer Science Department, University of British Columbia, Vancouver, B.C., Canada
mariusm@cs.ubc.ca, lowe@cs.ubc.ca

Keywords: nearest-neighbors search, randomized kd-trees, hierarchical k-means tree, clustering.

Abstract: For many computer vision problems, the most time consuming component consists of nearest neighbor matching in high-dimensional spaces. There are no known exact algorithms for solving these high-dimensional problems that are faster than linear search. Approximate algorithms are known to provide large speedups with only minor loss in accuracy, but many such algorithms have been published with only minimal guidance on selecting an algorithm and its parameters for any given problem. In this paper, we describe a system that answers the question, "What is the fastest approximate nearest-neighbor algorithm for my data?" Our system will take any given dataset and desired degree of precision and use these to automatically determine the best algorithm and parameter values. We also describe a new algorithm that applies priority search on hierarchical k-means trees, which we have found to provide the best known performance on many datasets. After testing a range of alternatives, we have found that multiple randomized k-d trees provide the best performance for other

Comparison of approximate NN-search methods

Dataset: 100K SIFT descriptors



Code for all methods available online, see Muja&Lowe'09

Figure: Muja&Lowe'09

Approximate nearest neighbour search (references)

J. L. Bentley. Multidimensional binary search trees used for associative searching. Comm. ACM, 18(9), 1975.

Freidman, J. H., Bentley, J. L., and Finkel, R. A. An algorithm for finding best matches in logarithmic expected time. *ACM Trans. Math. Softw., 3:209–226, 1977.*

Arya, S., Mount, D. M., Netanyahu, N. S., Silverman, R., and Wu, A. Y. An optimal algorithm for approximate nearest neighbor searching in fixed dimensions. *Journal of the ACM, 45:891–923, 1998.*

C. Silpa-Anan and R. Hartley. Optimised KD-trees for fast image descriptor matching. In CVPR, 2008.

M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In VISAPP, 2009.

P. Indyk and R. Motwani, “Approximate nearest neighbors: towards removing the curse of dimensionality,” in *Proc. of 30th ACM Symposium on Theory of Computing, 1998*

G. Shakhnarovich, P. Viola, and T. Darrell, “Fast pose estimation with parameter-sensitive hashing,” in *Proc. of the IEEE International Conference on Computer Vision, 2003.*

R. Salakhutdinov and G. Hinton, “Semantic Hashing,” ACM SIGIR, 2007.

Y. Weiss, A. Torralba, and R. Fergus, “Spectral hashing,” in *NIPS, 2008.*

ANN - search (references continued)

O. Chum, J. Philbin, and A. Zisserman. Near duplicate image detection: min-hash and tf-idf weighting. BMVC., 2008.

M. Raginsky and S. Lazebnik, “Locality-Sensitive Binary Codes from Shift-Invariant Kernels,” in *Proc. of Advances in neural information processing systems, 2009.*

B. Kulis and K. Grauman, “Kernelized locality-sensitive hashing for scalable image search,” Proc. of the IEEE International Conference on Computer Vision, 2009.

J. Wang, S. Kumar, and S.-F. Chang, “Semi-supervised hashing for scalable image retrieval,” in IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2010.

J. Wang, S. Kumar, and S.-F. Chang, “Sequential projection learning for hashing with compact codes,” in Proceedings of the 27th International Conference on Machine Learning, 2010.

So far …

- Linear exhaustive search can be prohibitively expensive for large image collections

- Answer (so far): approximate NN search methods
  - Randomized KD-trees
  - Locality sensitive hashing

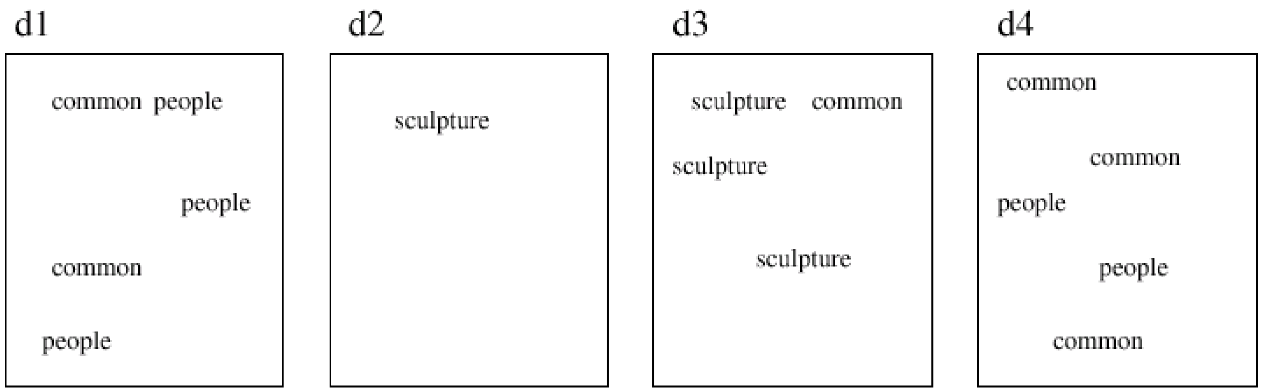
- However, memory footprint can be still high.

  Example: N = 10^7 images, 10^{10} SIFT features with 128B per feature $\Longrightarrow$ 1TB of memory

Look how text-based search engines (Google) index documents – **inverted files**.

Indexing text with inverted files

Document collection:

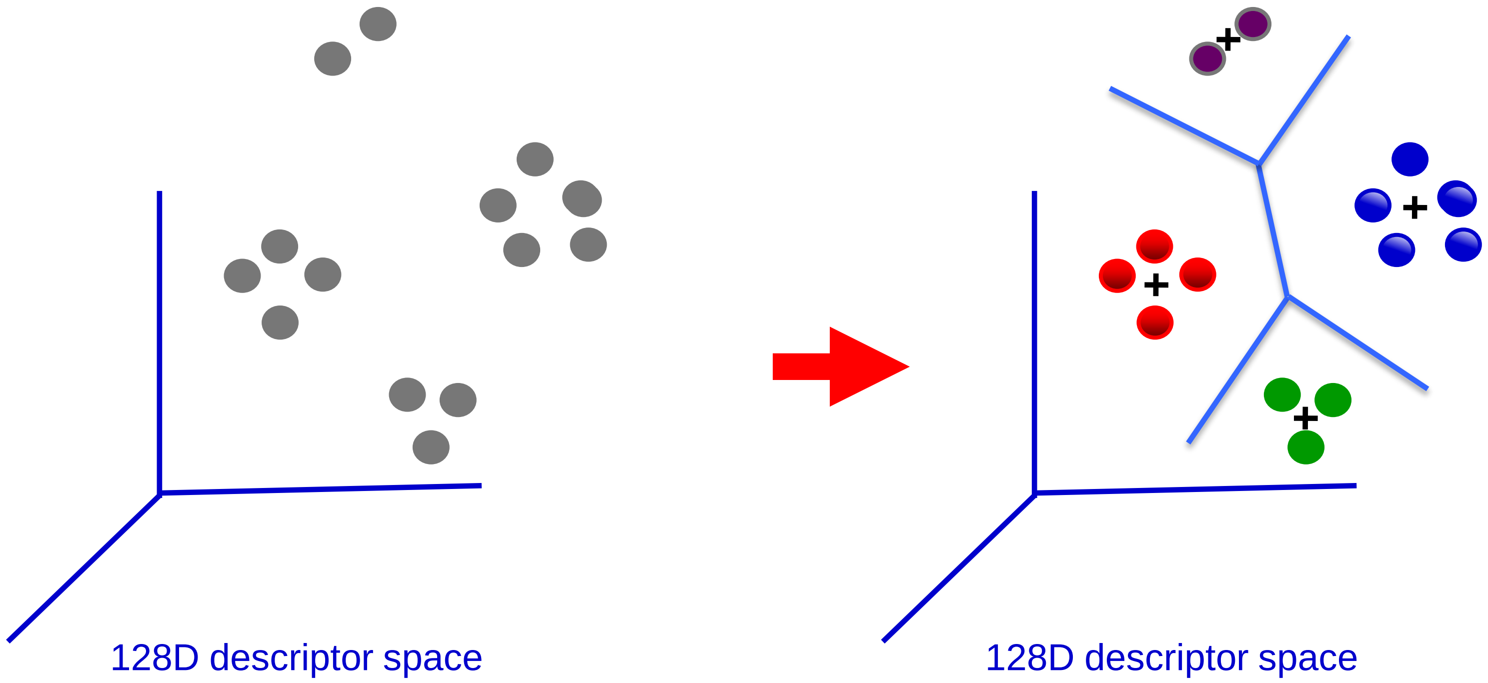


Inverted file:

Term	**List of hits** (occurrences in documents)
People	[d1:hit hit hit], [d4:hit hit] …
Common	[d1:hit hit], [d3: hit], [d4: hit hit hit] …
Sculpture	[d2:hit], [d3: hit hit hit] …

Need to map feature descriptors to “visual words”.

Build a visual vocabulary



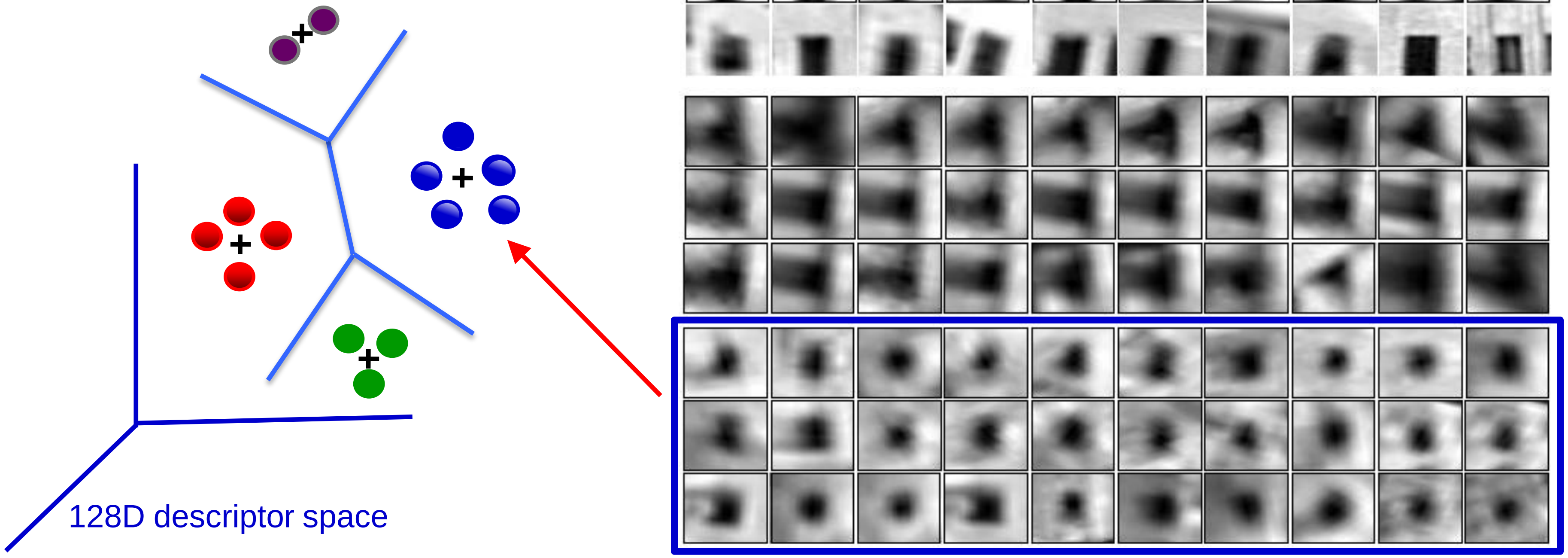
Vector quantize descriptors

- Compute SIFT features from a subset of images
- K-means clustering (need to choose K)

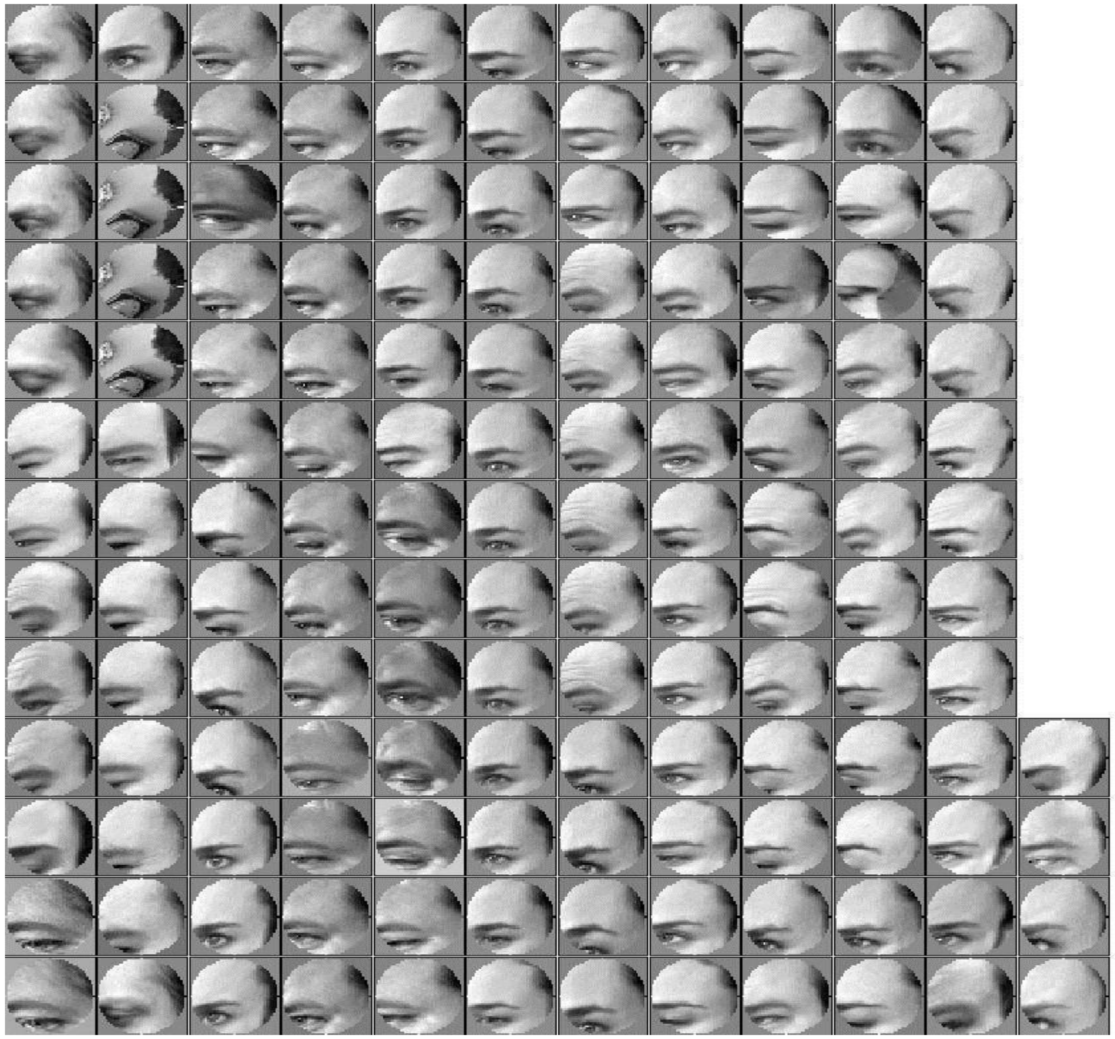
[Sivic and Zisserman, ICCV 2003]

Visual words

Example: each group of patches belongs to the same visual word

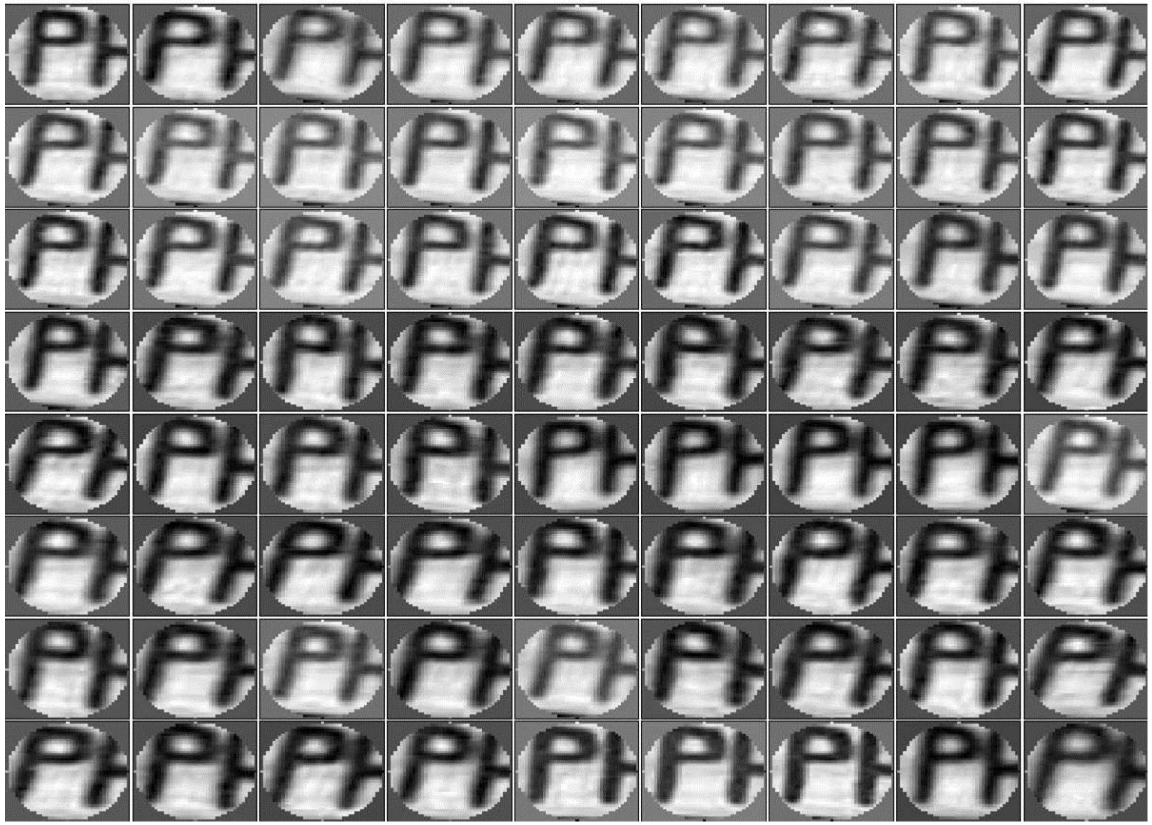


Samples of visual words (clusters on SIFT descriptors):



More specific example

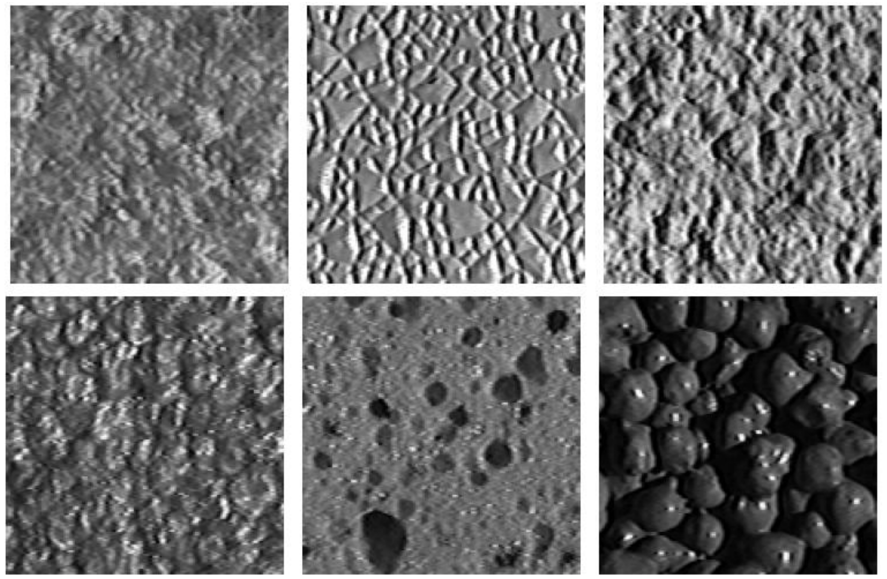
Samples of visual words (clusters on SIFT descriptors):



More specific example

Visual words

- First explored for texture and material representations
- *Texton* = cluster center of filter responses over collection of images
- Describe textures and materials based on distribution of prototypical texture elements.



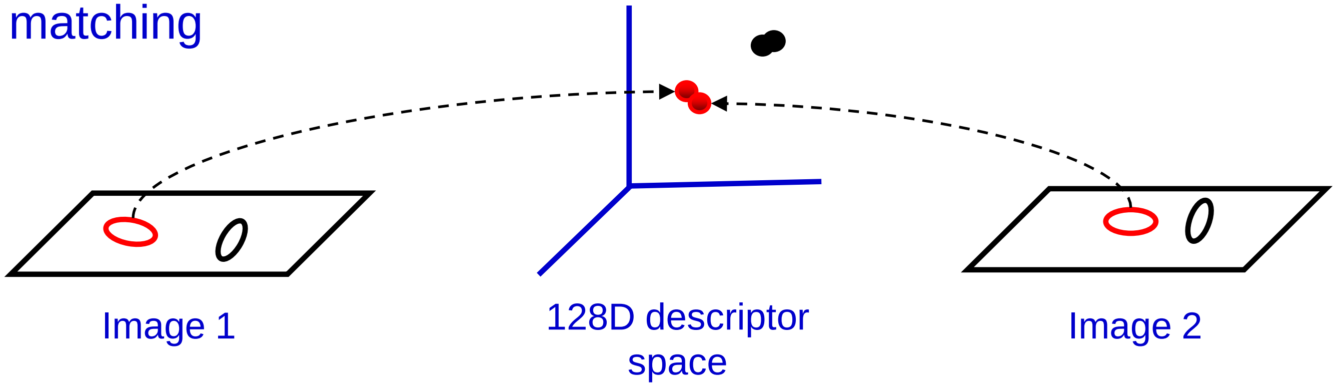
Leung & Malik 1999; Varma & Zisserman, 2002; Lazebnik, Schmid & Ponce, 2003;

Visual words: quantize descriptor space

Sivic and Zisserman, ICCV 2003

Nearest neighbour matching

- expensive to do for all frames

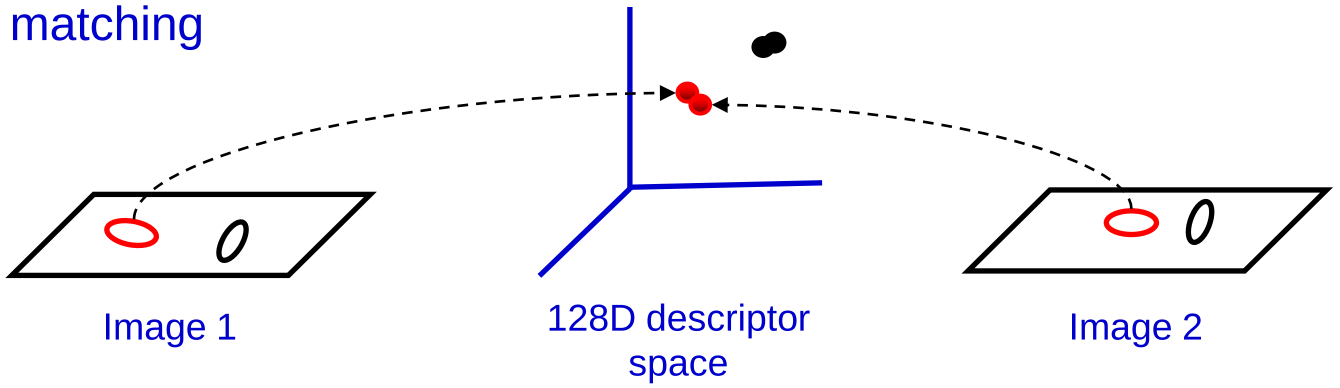


Visual words: quantize descriptor space

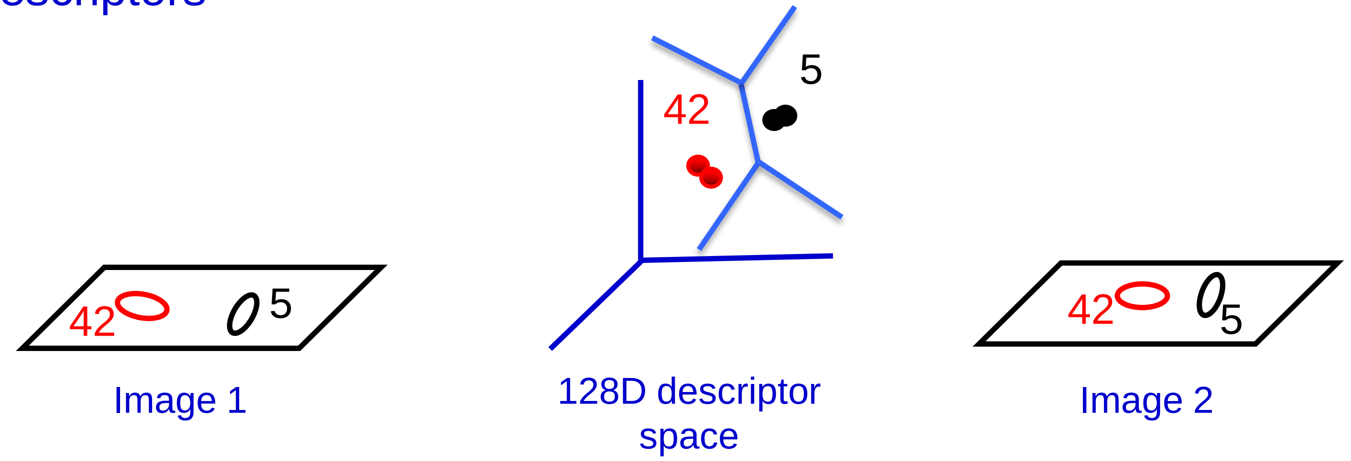
Sivic and Zisserman, ICCV 2003

Nearest neighbour matching

- expensive to do for all frames



Vector quantize descriptors

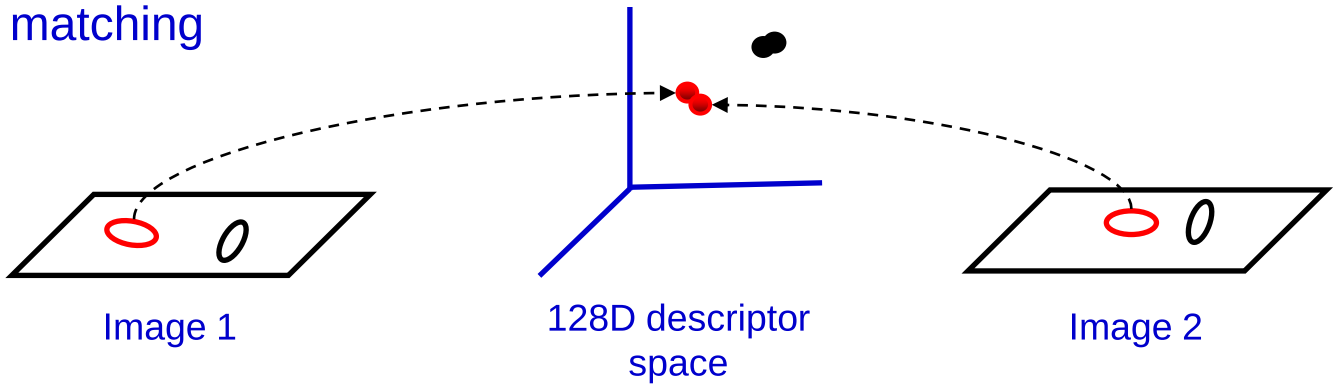


Visual words: quantize descriptor space

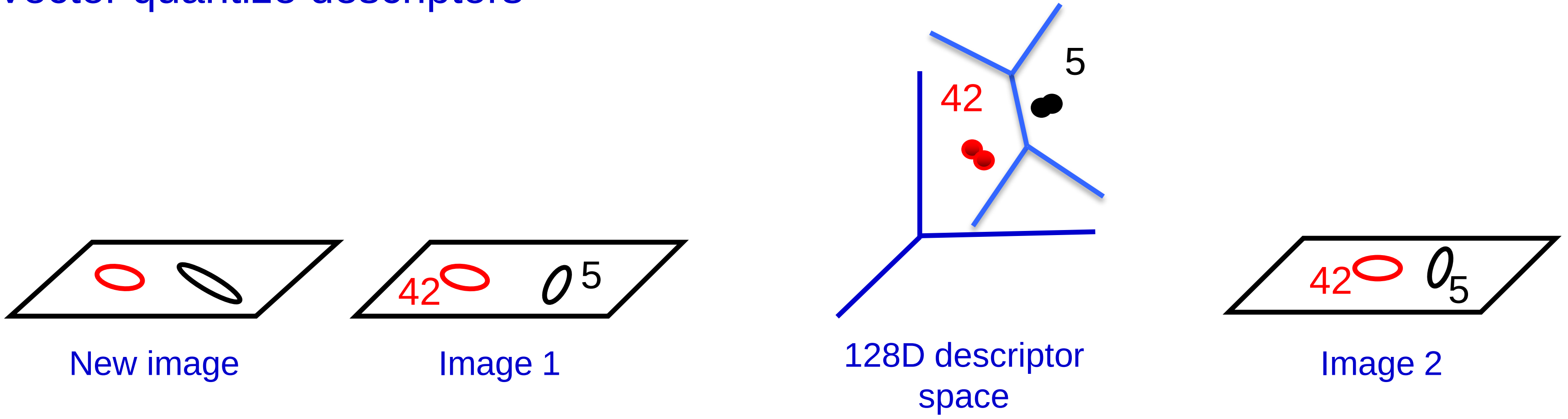
Sivic and Zisserman, ICCV 2003

Nearest neighbour matching

- expensive to do for all frames



Vector quantize descriptors

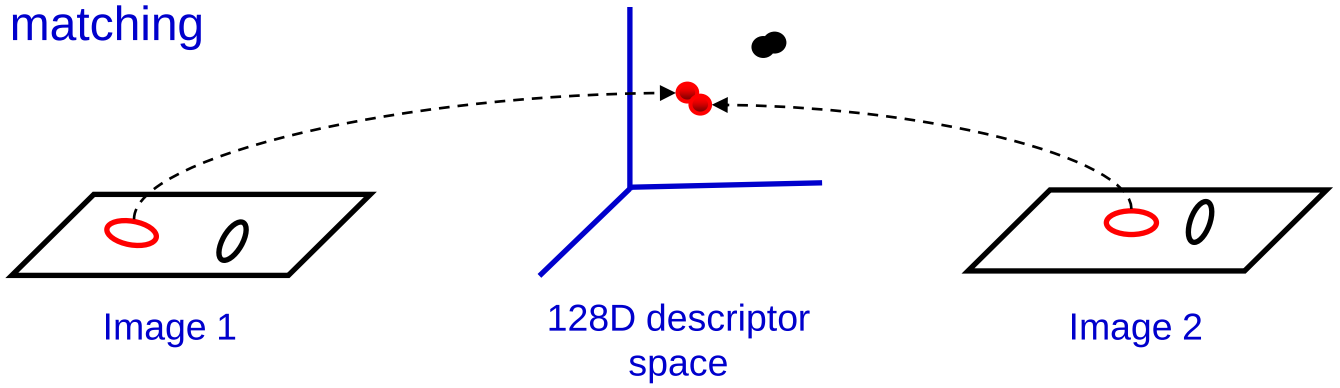


Visual words: quantize descriptor space

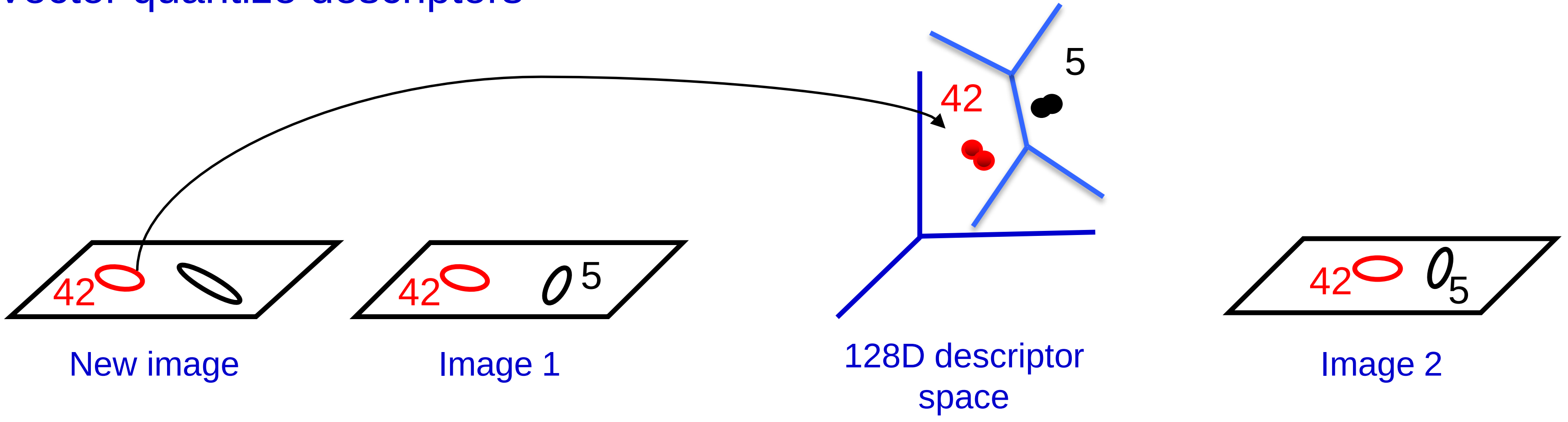
Sivic and Zisserman, ICCV 2003

Nearest neighbour matching

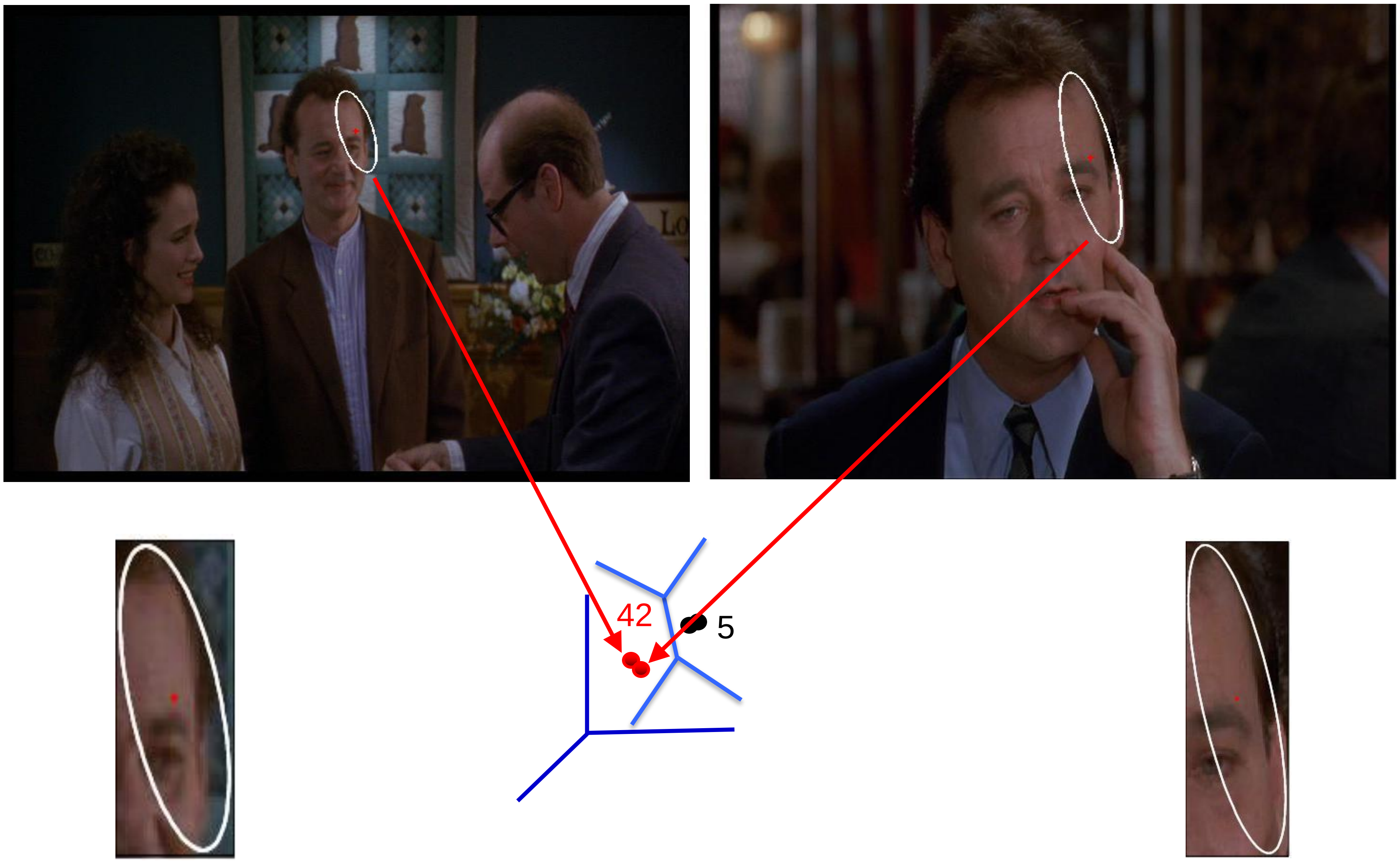
- expensive to do for all frames



Vector quantize descriptors



Vector quantize the descriptor space (SIFT)

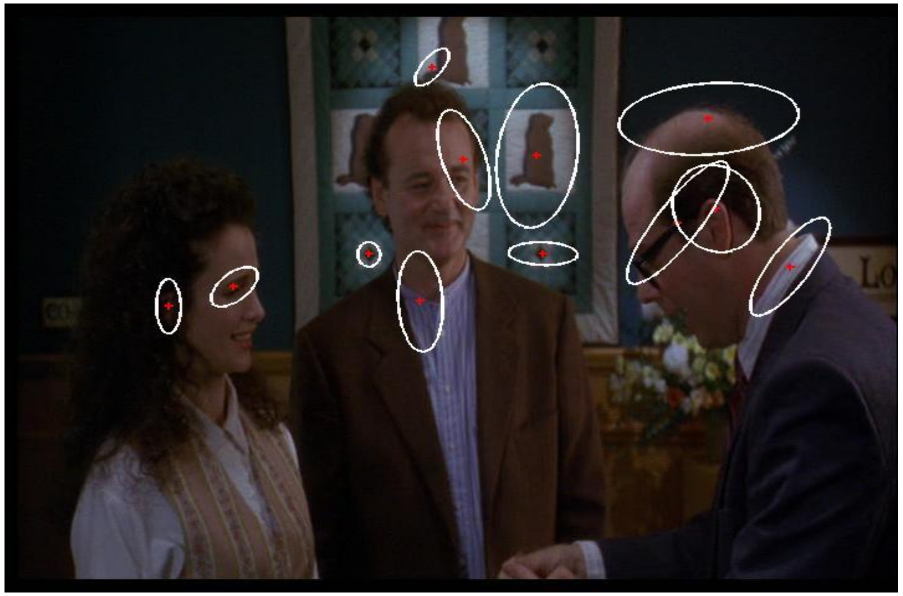


The same visual word

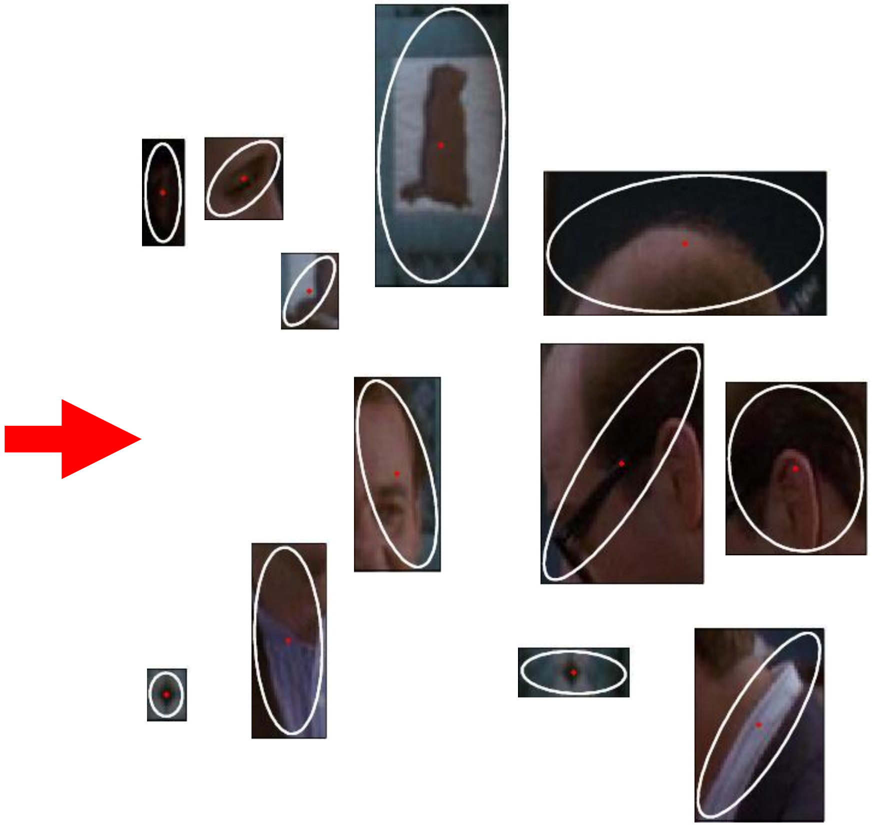
Representation: bag of (visual) words

Visual words are 'iconic' image patches or fragments

- represent their frequency of occurrence
- but not their position

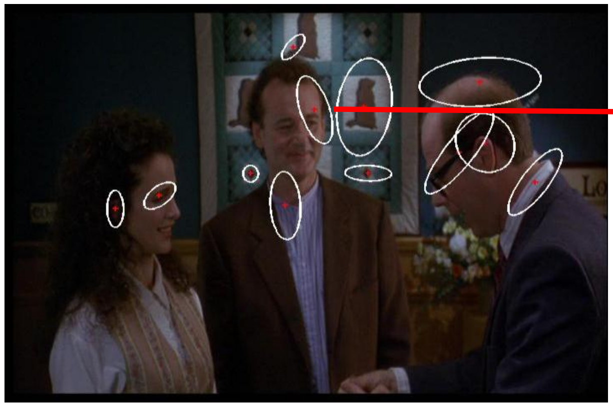


Image

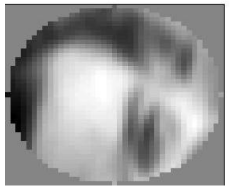


Colelction of visual words

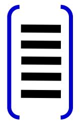
Offline: Assign visual words and compute histograms for each image



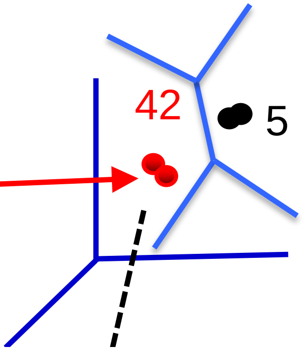
Detect patches



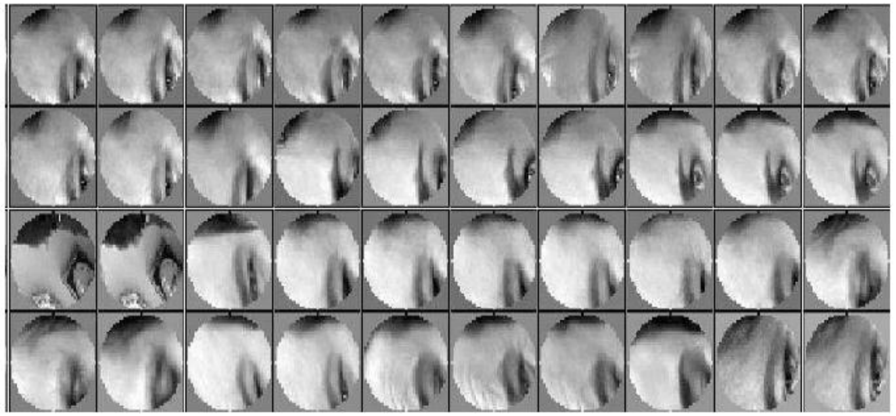
Normalize patch



Compute SIFT descriptor



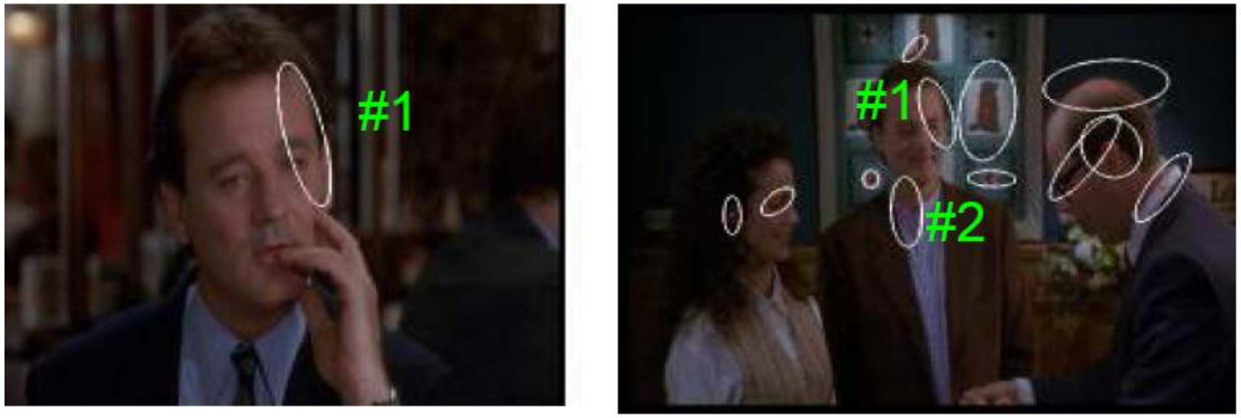
Find nearest cluster center



$$\begin{bmatrix} 2 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ \cdots \end{bmatrix}$$

Represent image as a sparse histogram of visual word occurrences

Offline: create an index



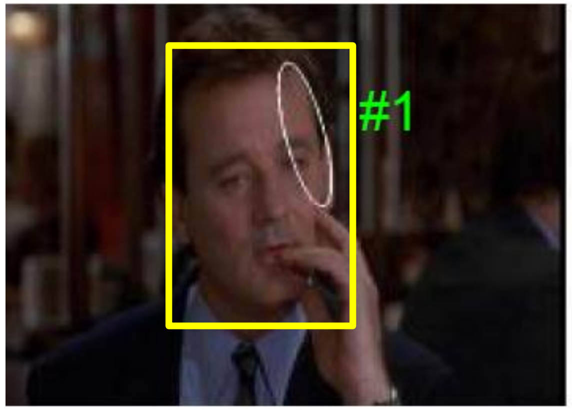
frame #5 frame #10

Word number		Posting list
1	→	5,10, ...
2	→	10,...
...		...

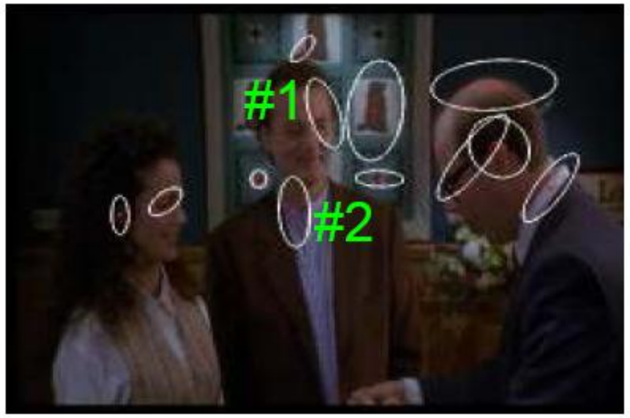
- For fast search, store a “posting list” for the dataset
- This maps visual word occurrences to the images they occur in

(i.e. like the “book index”)

At run time



frame #5

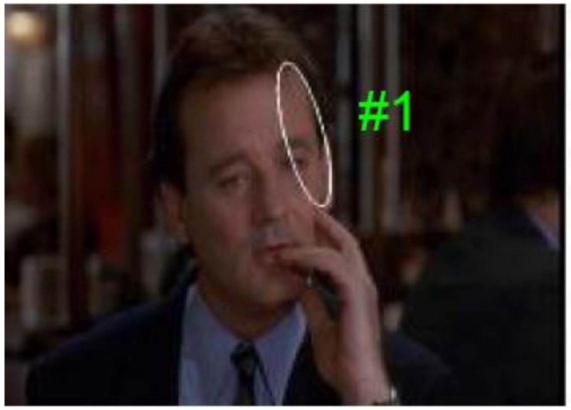


frame #10

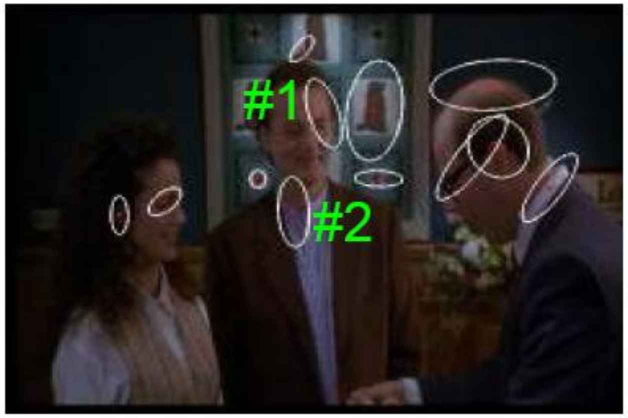
Word number	Posting list
1	5,10, ...
2	10,...
...	...

- User specifies a query region
- Generate a short-list of images using visual words in the region
    1. Accumulate all visual words within the query region
    2. Use "book index" to find other frames with these words
    3. Compute similarity for images which share at least one word

At run time



frame #5



frame #10

Word number	Posting list
1	5,10, ...
2	10,...
...	...

- Score each image by the (weighted) number of common visual words (tentative correspondences)
- Worst case complexity is linear in the number of images N
- In practice, it is linear in the length of the lists (<< N)

Another interpretation: the bag-of-visual-words model

For a vocabulary of size K, each image is represented by a K-vector

$$\mathbf{v}_d = (t_1, \ldots, t_i, \ldots, t_K)^\top$$

where t_i is the number of occurrences of visual word i.

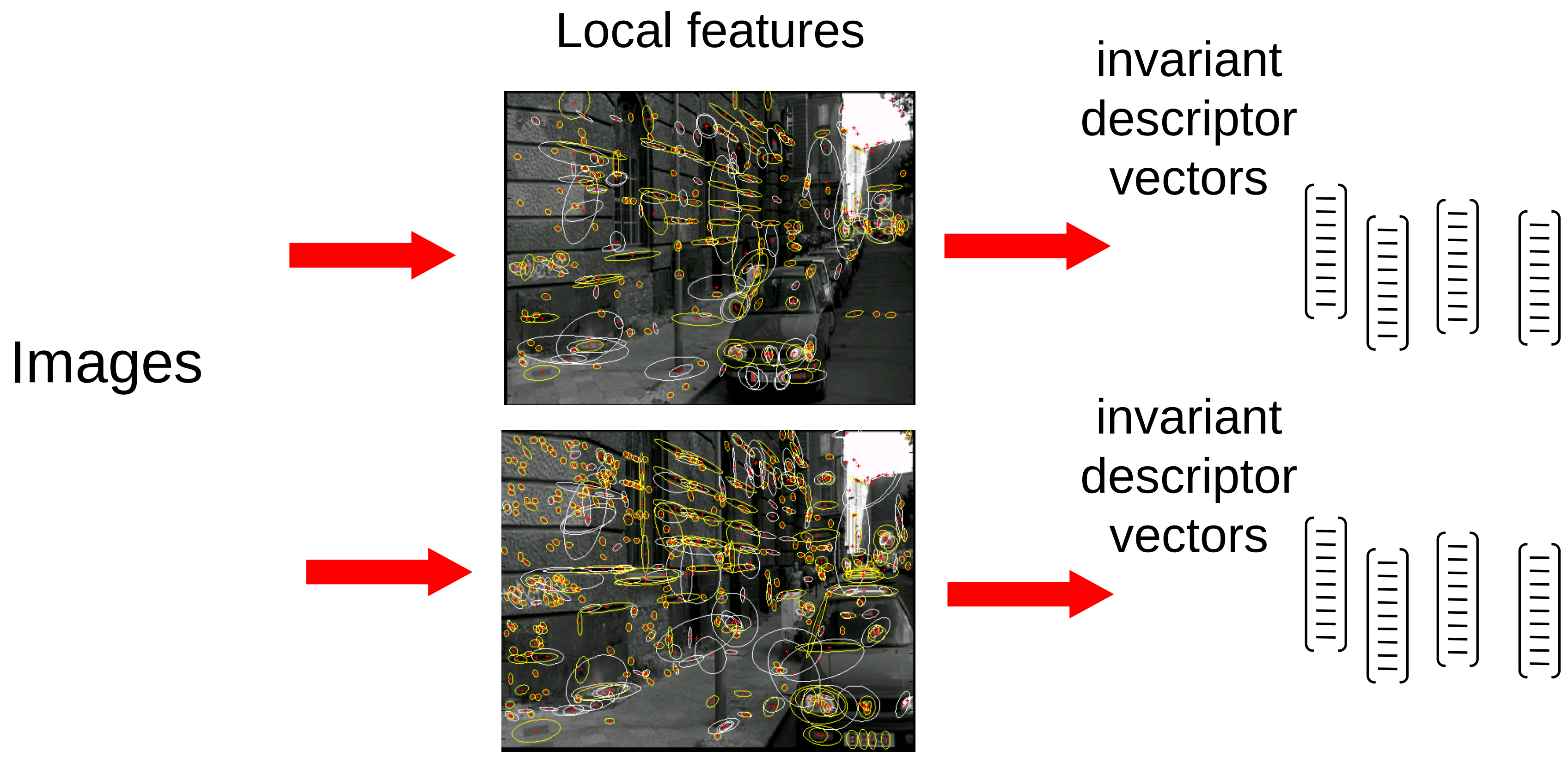
Images are ranked by the normalized scalar product between the query vector v_q and all vectors in the database v_d:

$$f_d = \frac{{\mathbf{v}_q}^\top \mathbf{v}_d}{\|\mathbf{v}_q\|_2 \; \|\mathbf{v}_d\|_2}$$

Scalar product can be computed efficiently using inverted file.

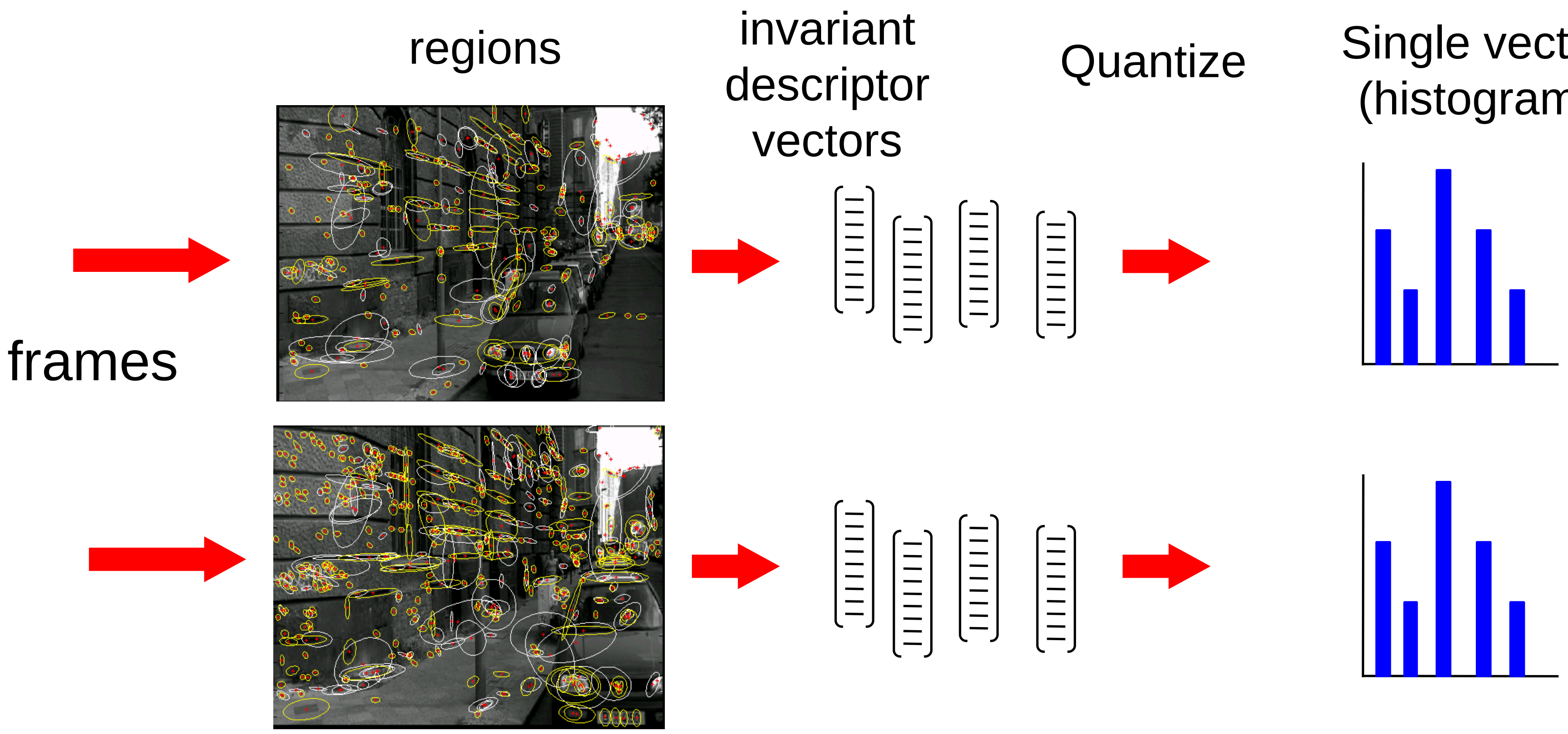
What if vectors are binary? What is the meaning of ${\mathbf{v}_q}^\top \mathbf{v}_d$?

Strategy I: Efficient approximate NN search



1. Compute local features in each image independently (offline)
2. “Label” each feature by a descriptor vector based on its intensity (offline)
3. Finding corresponding features is transformed to finding nearest neighbour vectors
4. Rank matched images by number of (tentatively) corresponding regions
5. Verify top ranked images based on spatial consistency (The first part of this lecture)

Strategy II: Match histograms of visual words



1. Compute affine covariant regions in each frame independently (offline)
2. “Label” each region by a vector of descriptors based on its intensity (offline)
3. **Build histograms of visual words by descriptor quantization (offline)**
4. **Rank retrieved frames by matching vis. word histograms using inverted files.**
5. Verify retrieved frame based on spatial consistency (The first part of the lecture)

Visual words: discussion I.

Efficiency – cost of quantization

- Need to still assign each local descriptor to one of the cluster centers. Could be prohibitive for large vocabularies (K=1M)

- Approximate NN-search still needed
    - e.g. randomized k-d trees

- True also for building the vocabulary
    - approximate k-means [Philbin et al. 2007]

Visual words: discussion II.

Generalization

• Is vocabulary/quantization learned on one dataset good for searching another dataset?

• Experimentally observe a loss in performance.

But, see also a recent work by Jegou et al.:

Hamming Embedding and Weak Geometry Consistency for Large Scale Image Search, ECCV'2008

http://lear.inrialpes.fr/pubs/2008/JDS08a/

Visual words: discussion III.

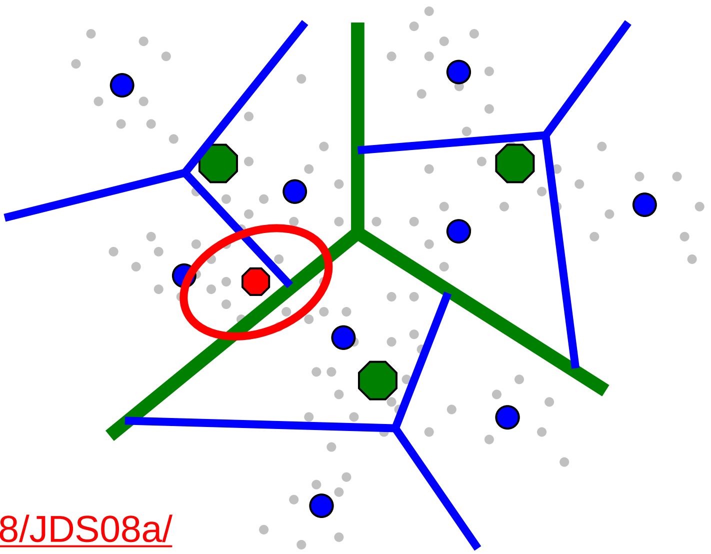
What about quantization effects?

- Visual word assignment can change due to e.g.

    noise in region detection,

    descriptor computation or

    non-modeled image variation (3D effects, lighting)



See also:

Jegou et al., ECCV'2008, http://lear.inrialpes.fr/pubs/2008/JDS08a/

Philbin et al. CVPR'08, http://www.robots.ox.ac.uk/~vgg/publications/html/philbin08-bibtex.html

Mikulik et al., ECCV'10, http://cmp.felk.cvut.cz/~chum/papers/mikulik_eccv10.pdf

Philbin et al., ECCV'10, http://www.di.ens.fr/~josef/publications/philbin10b.pdf

Visual words: discussion IV.

- Need to determine the size of the vocabulary, K.

- Other algorithms for building vocabularies, e.g.

agglomerative clustering / mean-shift, but typically more expensive.

- Supervised quantization?

Also give examples of images / descriptors which should and should not match.

E.g.:
Philbin et al. ECCV’10, http://www.robots.ox.ac.uk/~vgg/publications/html/philbin10b-bibtex.html

Visual search using local regions (references)

C. Schmid, R. Mohr, Local Greyvalue Invariants for Image Retrieval, PAMI, 1997

J. Sivic, A. Zisserman, Text retrieval approach to object matching in videos, ICCV, 2003

D. Nister, H. Stewenius, Scalable Recognition with a Vocabulary Tree, CVPR, 2006.

J. Philbin, O. Chum, M. Isard, J. Sivic, A. Zisserman, Object retrieval with large vocabularies and fast spatial matching, CVPR, 2007

O. Chum, J. Philbin, M. Isard, J. Sivic, A. Zisserman, Total Recall: Automatic Query Expansion with a Generative Feature Model for Object Retrieval, ICCV, 2007

H. Jegou, M. Douze, C. Schmid, Hamming embedding and weak geometric consistency for large scale image search, ECCV'2008

O. Chum, M. Perdoch, J. Matas: Geometric min-Hashing: Finding a (Thick) Needle in a Haystack, CVPR 2009

H. Jégou, M. Douze and C. Schmid, On the burstiness of visual elements, CVPR, 2009

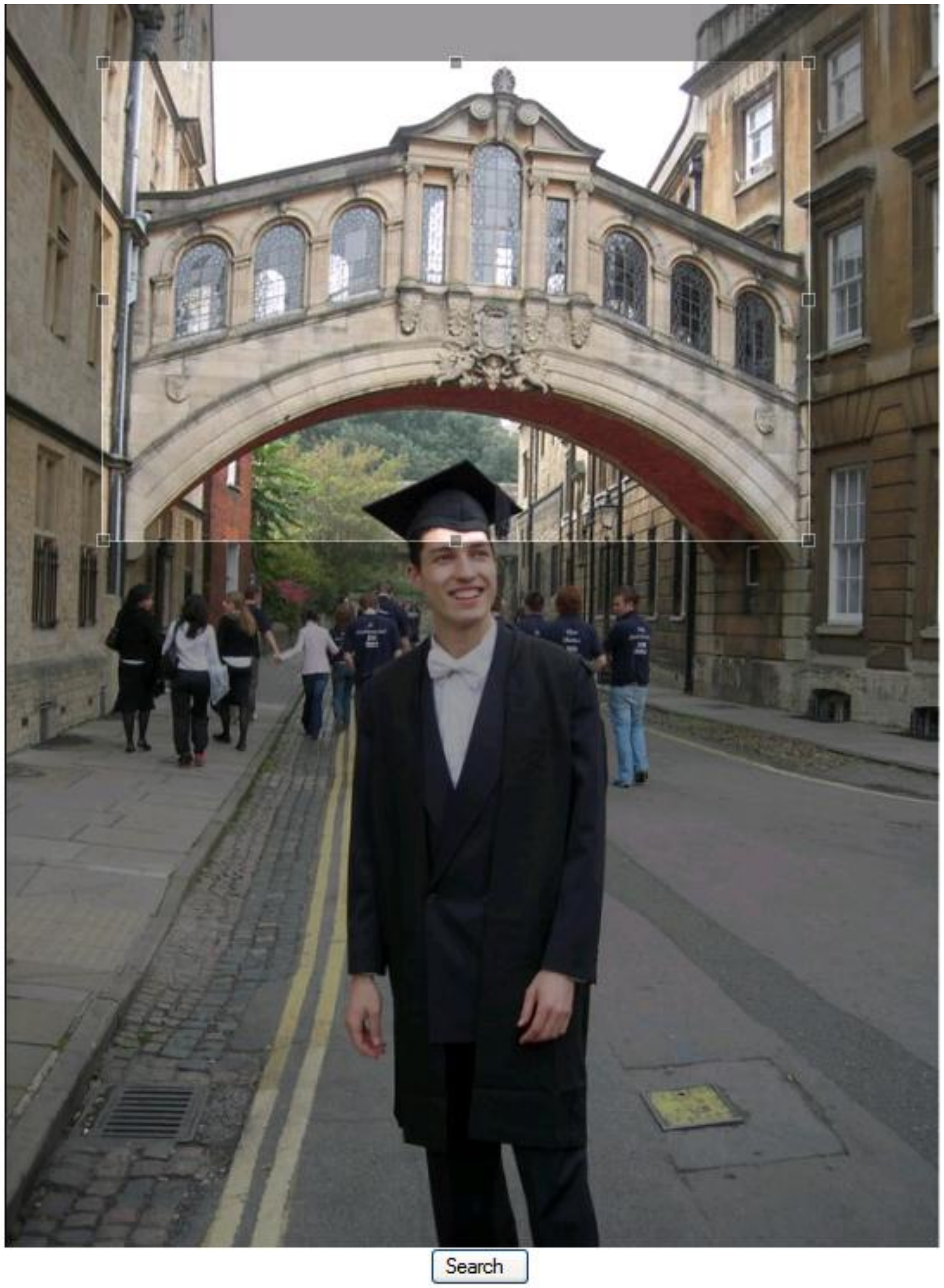
H. Jégou, M. Douze, C. Schmid and P. Pérez, Aggregating local descriptors into a compact image representation, CVPR'2010

Efficient visual search for objects and places

Oxford Buildings Search - demo

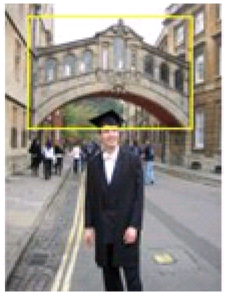
http://www.robots.ox.ac.uk/~vgg/research/oxbuildings/index.html

Example



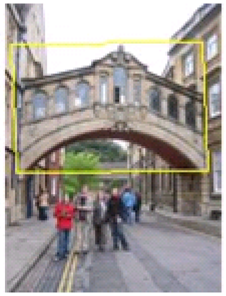
Search results 1 to 20 of 104844

1



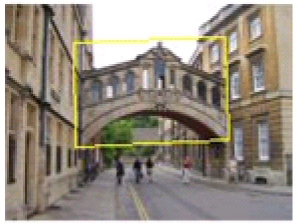
ID: oxc1_hertford_000011
Score: 1816.000000
Putative: 2325
Inliers: 1816
Hypothesis: 1.000000 0.000000 0.000015 0.000000 1.000000 0.000031
Detail

2



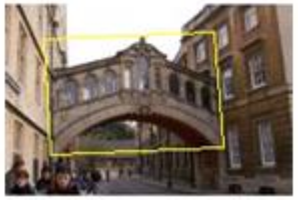
ID: oxc1_all_souls_000075
Score: 352.000000
Putative: 645
Inliers: 352
Hypothesis: 1.162245 0.041211 -70.414459 -0.012913 1.146417 91.276093
Detail

3



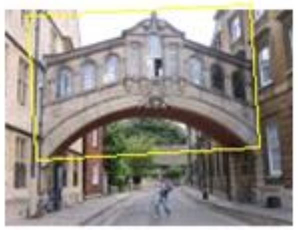
ID: oxc1_hertford_000064
Score: 278.000000
Putative: 527
Inliers: 278
Hypothesis: 0.928686 0.026134 169.954620 -0.041703 0.937558 97.962112
Detail

4



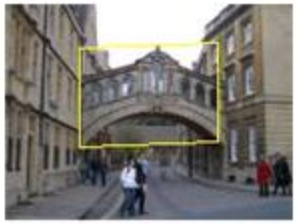
ID: oxc1_oxford_001612
Score: 252.000000
Putative: 451
Inliers: 252
Hypothesis: 1.046026 0.069416 51.576881 -0.044949 1.046938 76.264442
Detail

5



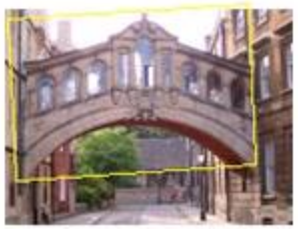
ID: oxc1_hertford_000123
Score: 225.000000
Putative: 446
Inliers: 225
Hypothesis: 1.361741 0.090413 -34.673317 -0.084659 1.301689 -32.281090
Detail

6



ID: oxc1_oxford_001085
Score: 224.000000
Putative: 389
Inliers: 224
Hypothesis: 0.848997 0.000000 195.707611 -0.031077 0.895546 114.583961
Detail

7



ID: oxc1_hertford_000077
Score: 195.000000
Putative: 386
Inliers: 195
Hypothesis: 1.465144 0.069286 -108.473091 -0.097598 1.461877 -30.205191
Detail

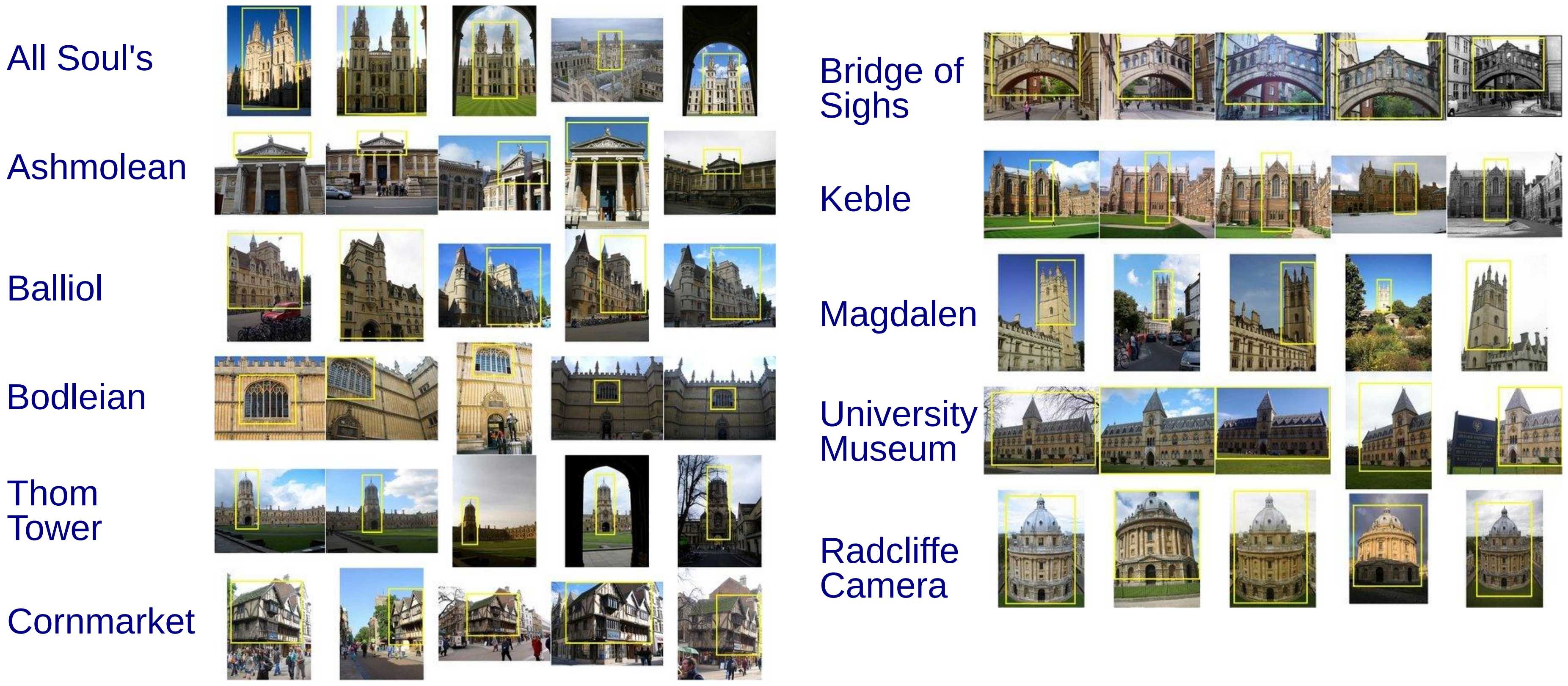
Oxford buildings dataset

- Automatically crawled from flickr
- Consists of:

Dataset	Resolution	# images	# features	Descriptor size
i	1024×768	5,062	16,334,970	1.9 GB
ii	1024×768	99,782	277,770,833	33.1 GB
iii	500×333	1,040,801	1,186,469,709	141.4 GB
Total		1,145,645	1,480,575,512	176.4 GB

Oxford buildings dataset

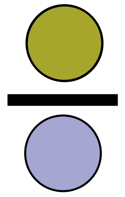
- Landmarks plus queries used for evaluation



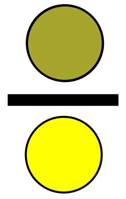
- Ground truth obtained for 11 landmarks
- Evaluate performance by mean Average Precision

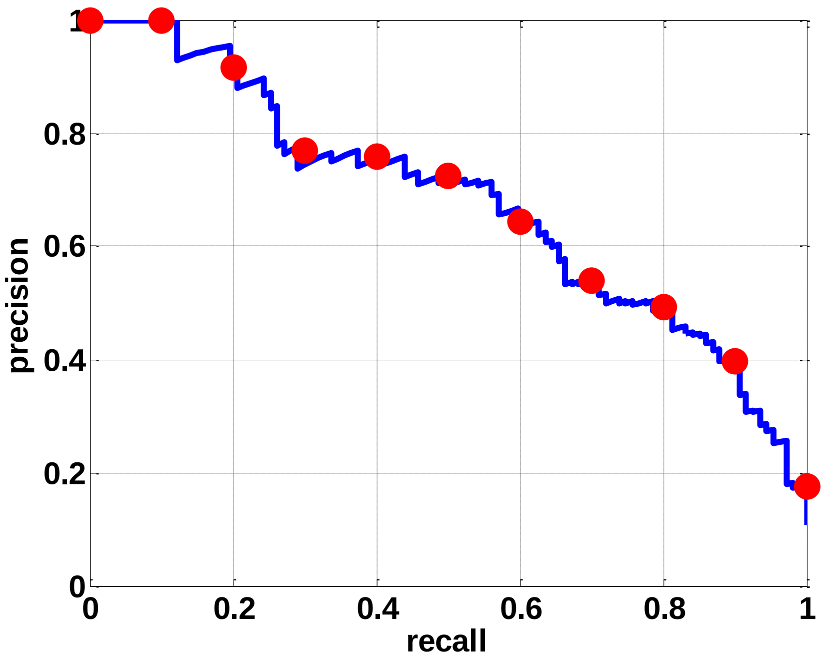
Measuring retrieval performance: Precision - Recall

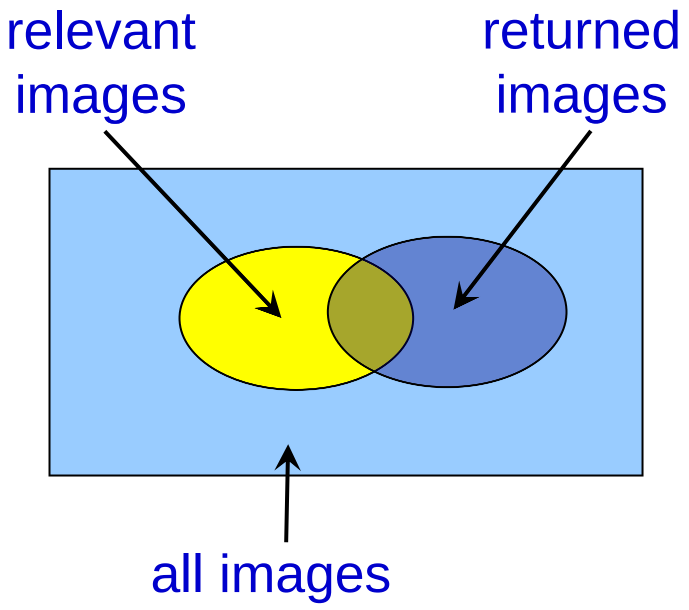
- Precision: % of returned images that are relevant



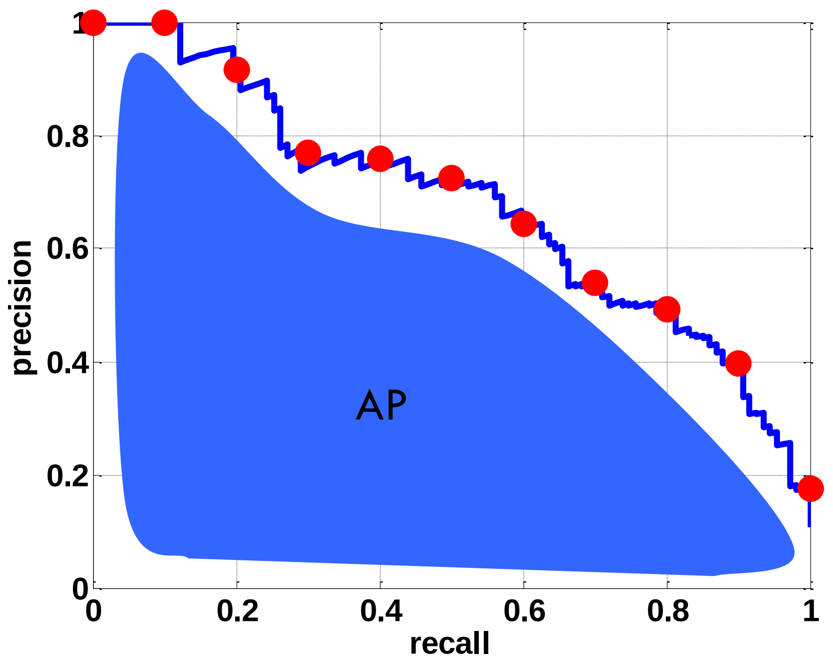
- Recall: % of relevant images that are returned





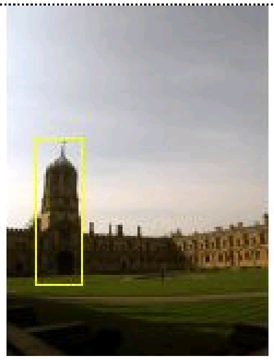


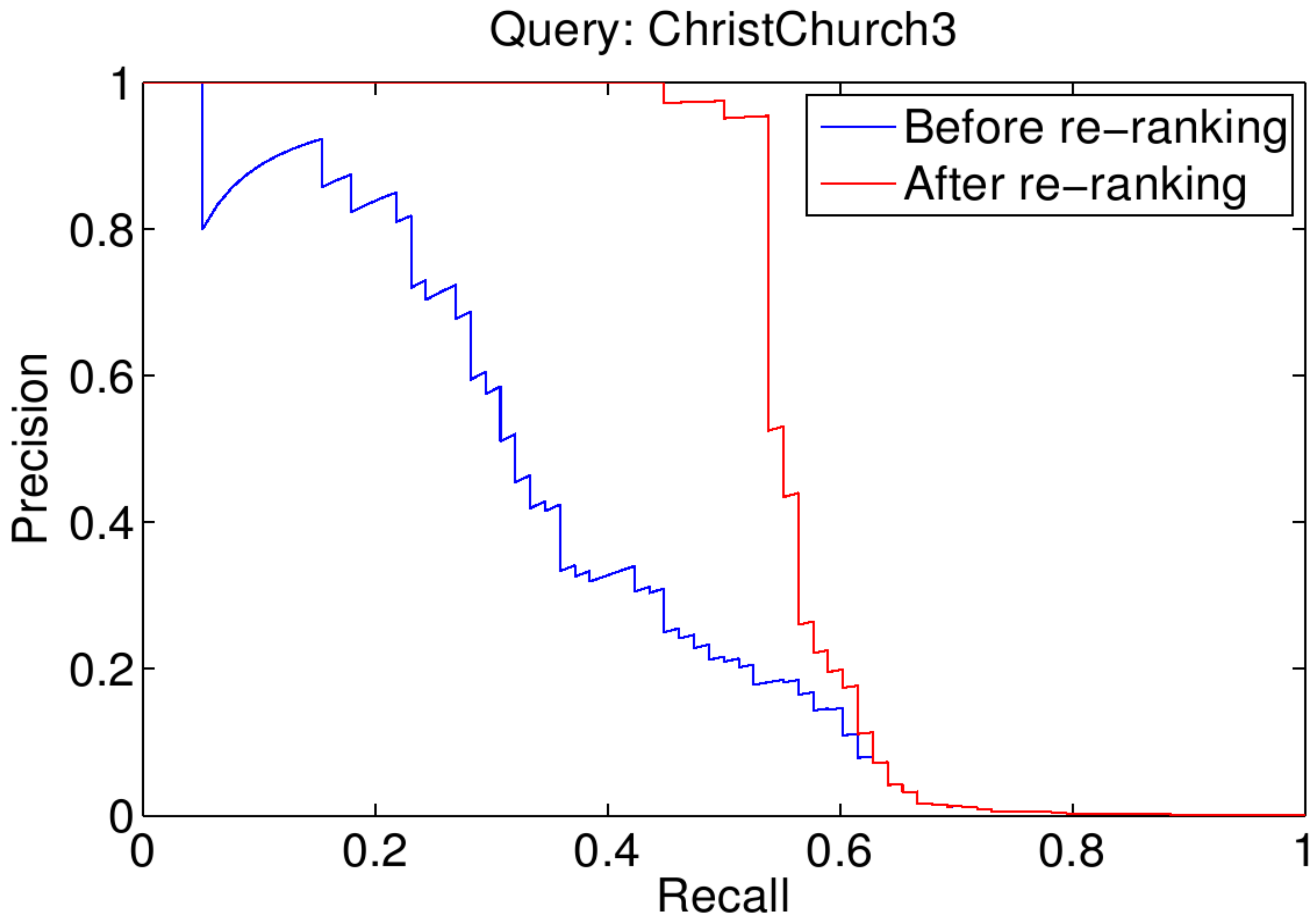
Average Precision



- A good AP score requires both high recall and high precision
- Application-independent

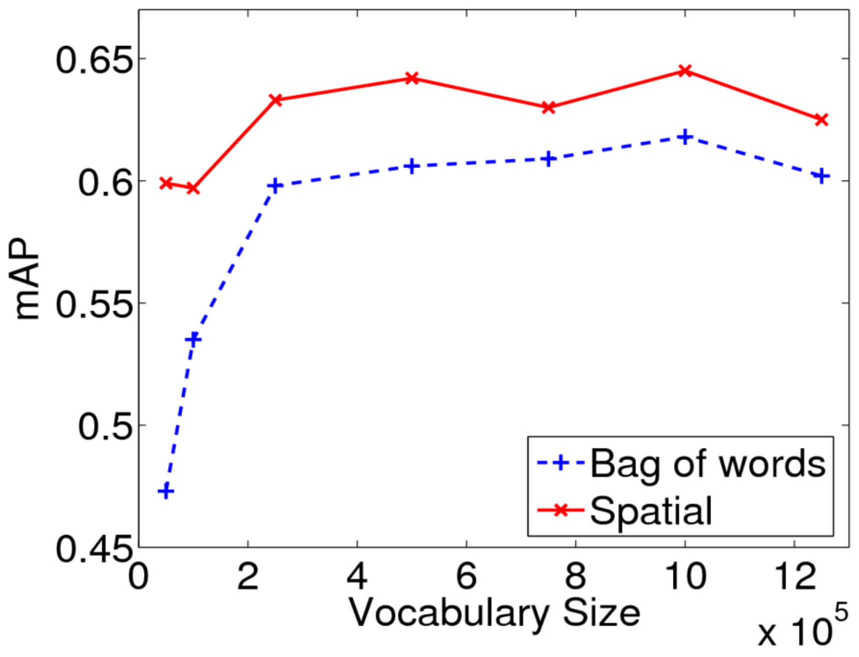
Performance measured by mean Average Precision (mAP) over 55 queries on 100K or 1.1M image datasets



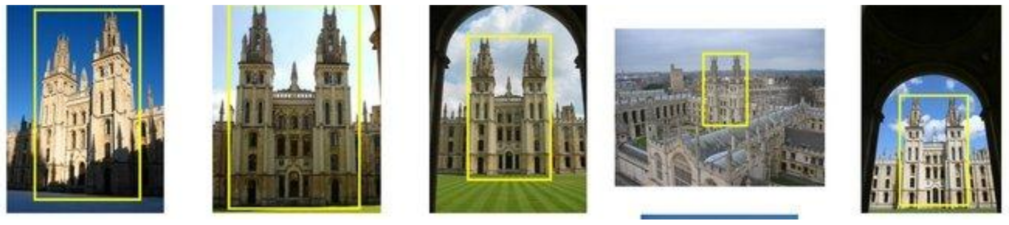

Query: ChristChurch3
Precision
Recall
Before re−ranking
After re−ranking
0
0.2
0.4
0.6
0.8
1
0
0.2
0.4
0.6
0.8
1

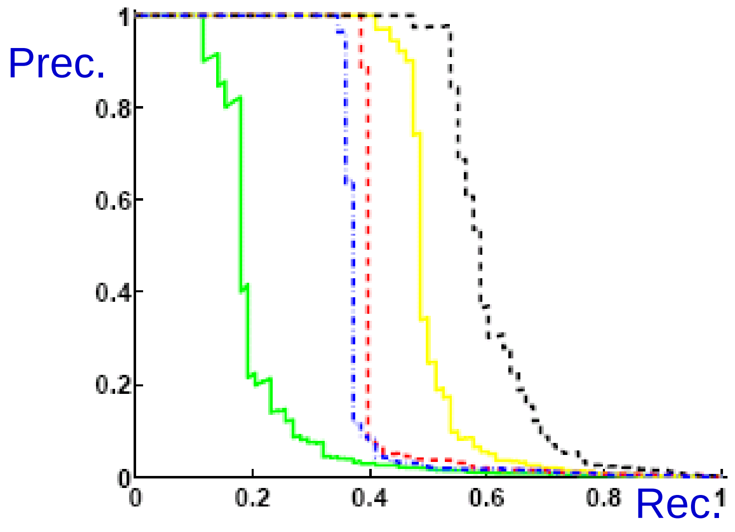
Mean Average Precision variation with vocabulary size

vocab size	bag of words	spatial
50K	0.473	0.599
100K	0.535	0.597
250K	0.598	0.633
500K	0.606	0.642
750K	0.609	0.630
1M	0.618	0.645
1.25M	0.602	0.625



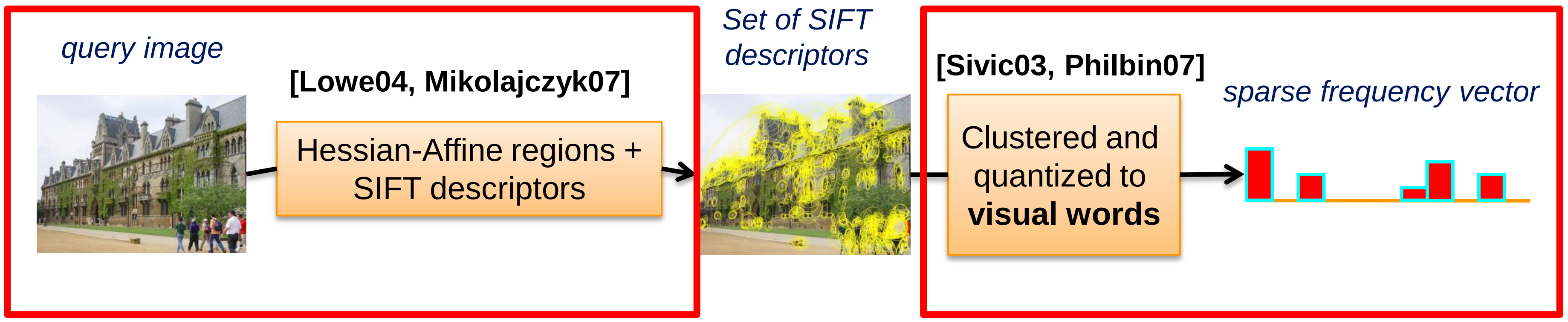
Query images





- high precision at low recall (like google)
- variation in performance over query
- none retrieve all instances

Why aren't all objects retrieved?



Obtaining visual words is like a sensor measuring the image

"noise" in the measurement process means that some visual words are missing or incorrect, e.g. due to

- Missed detections
- Changes beyond built in invariance
- Quantization effects

1. Query expansion
2. Better quantization

Consequence: Visual word in query is missing in target image

Query Expansion in text

In text :

- Reissue top n responses as queries
- Pseudo/blind relevance feedback
- Danger of topic drift

In vision:

- Reissue spatially verified image regions as queries

Query Expansion: Text

Original query: Hubble Telescope Achievements

Query expansion: Select top 20 terms from top 20 documents according to tf-idf

Added terms: Telescope, hubble, space, nasa, ultraviolet, shuttle, mirror, telescopes, earth, discovery, orbit, flaw, scientists, launch, stars, universe, mirrors, light, optical, species

Example from: Jimmy Lin, University of Maryland

Automatic query expansion

Visual word representations of two images of the same object may differ (due to e.g. detection/quantization noise) resulting in missed returns

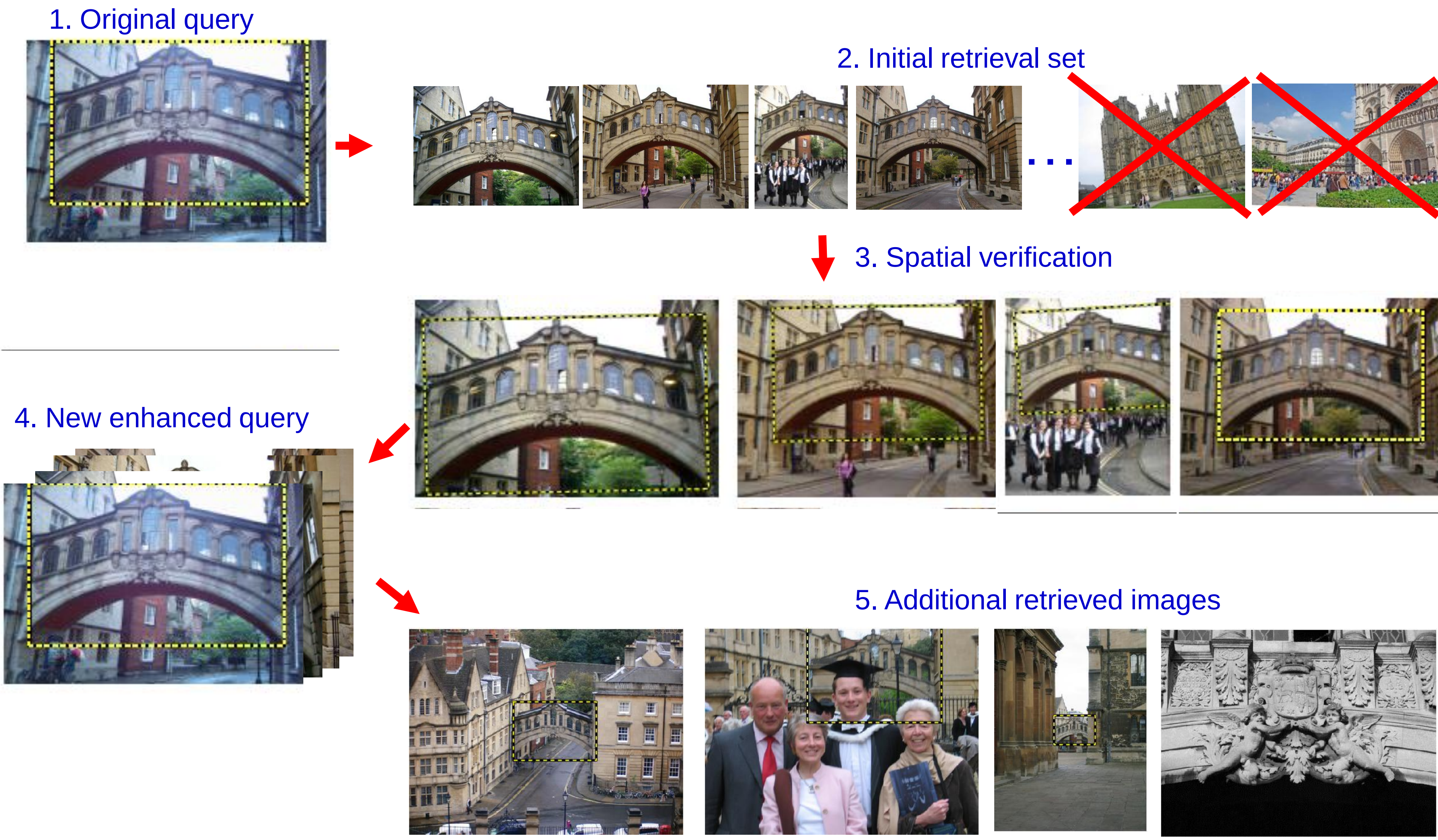
Initial returns may be used to add new relevant visual words to the query

Strong spatial model prevents ‘drift’ by discarding false positives

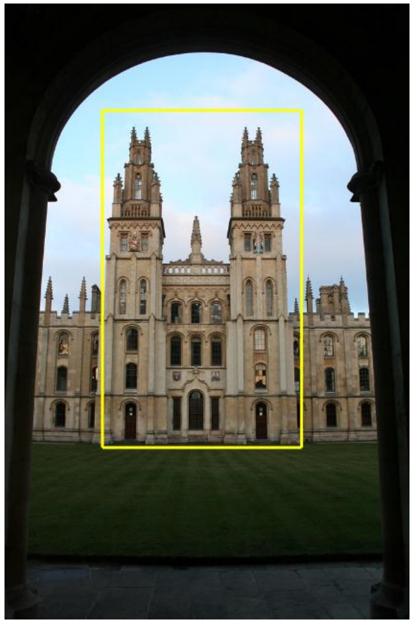
[Chum, Philbin, Sivic, Isard, Zisserman, ICCV’07;

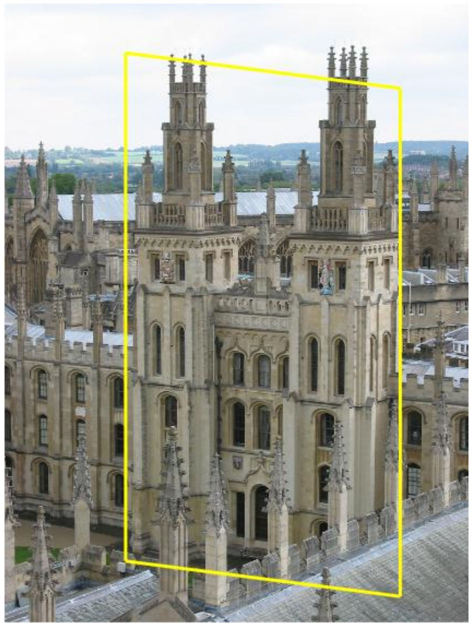
Chum, Mikulik, Perdoch, Matas, CVPR’11]

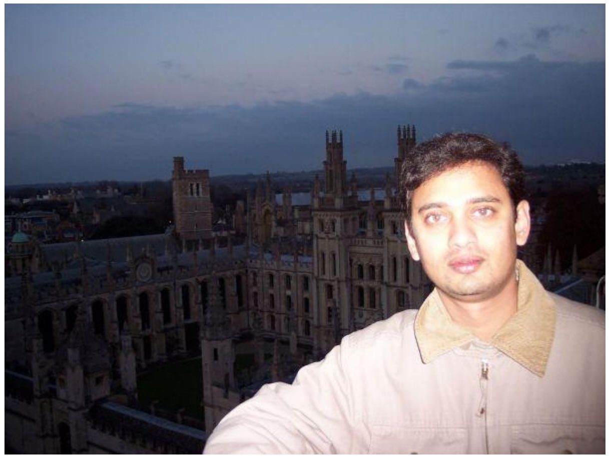
Visual query expansion - overview



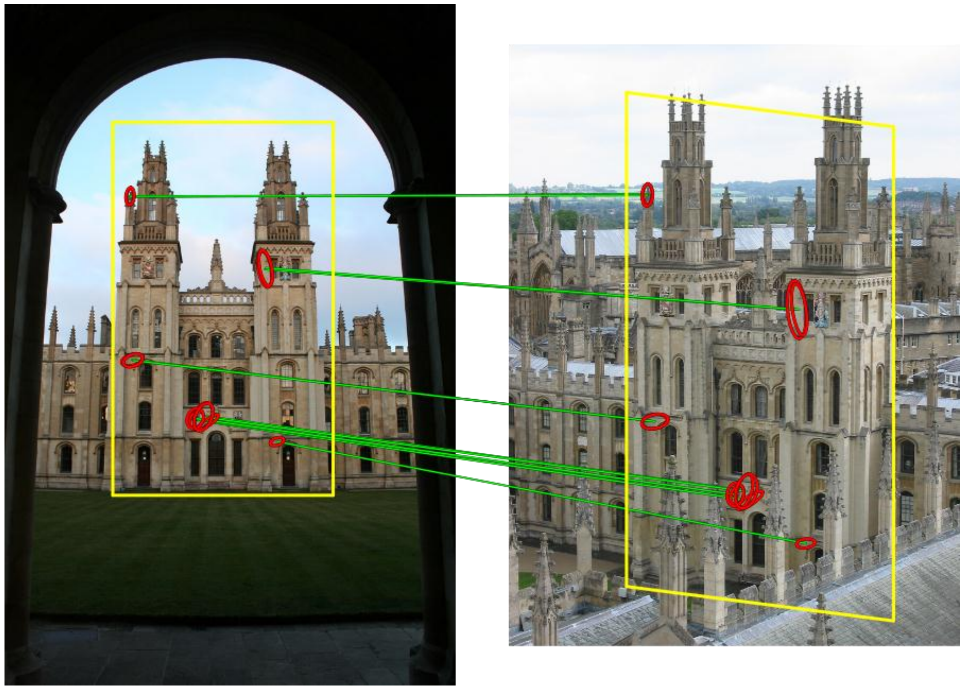
Query Expansion


Query Image

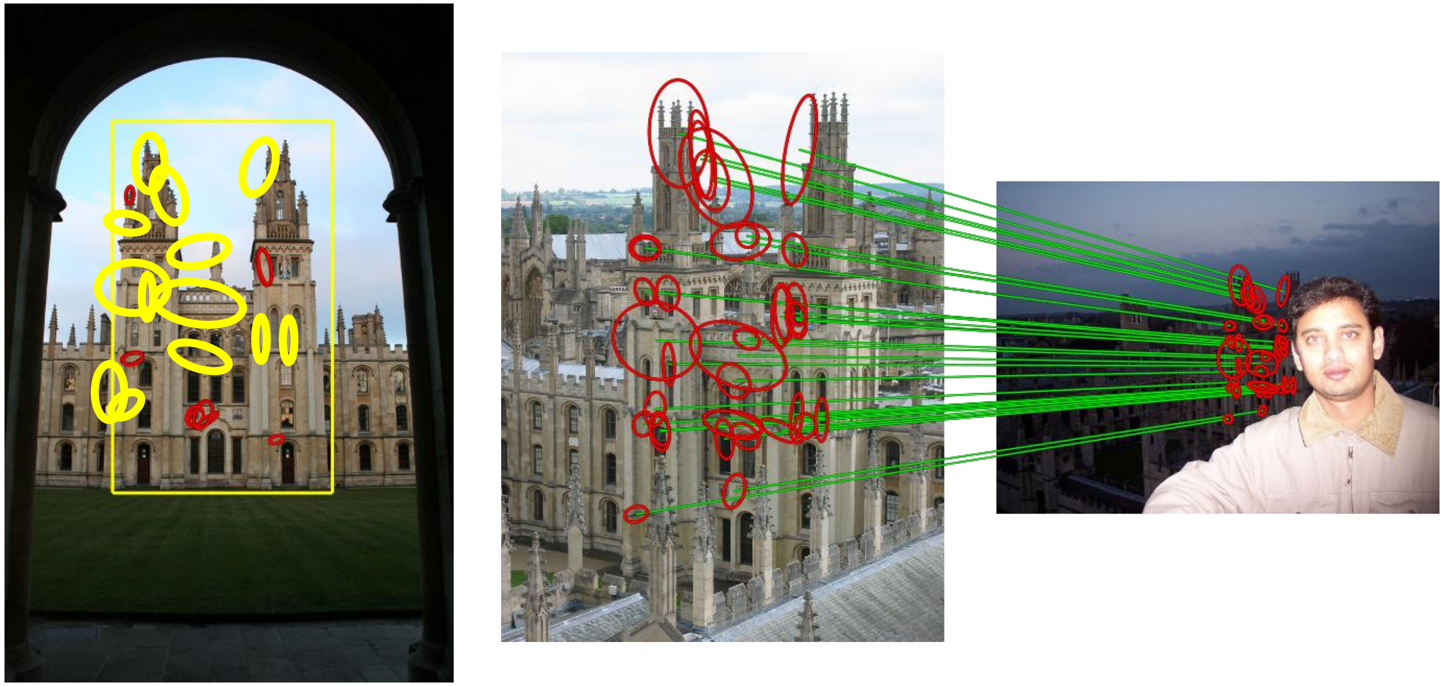

Originally retrieved image


Originally not retrieved

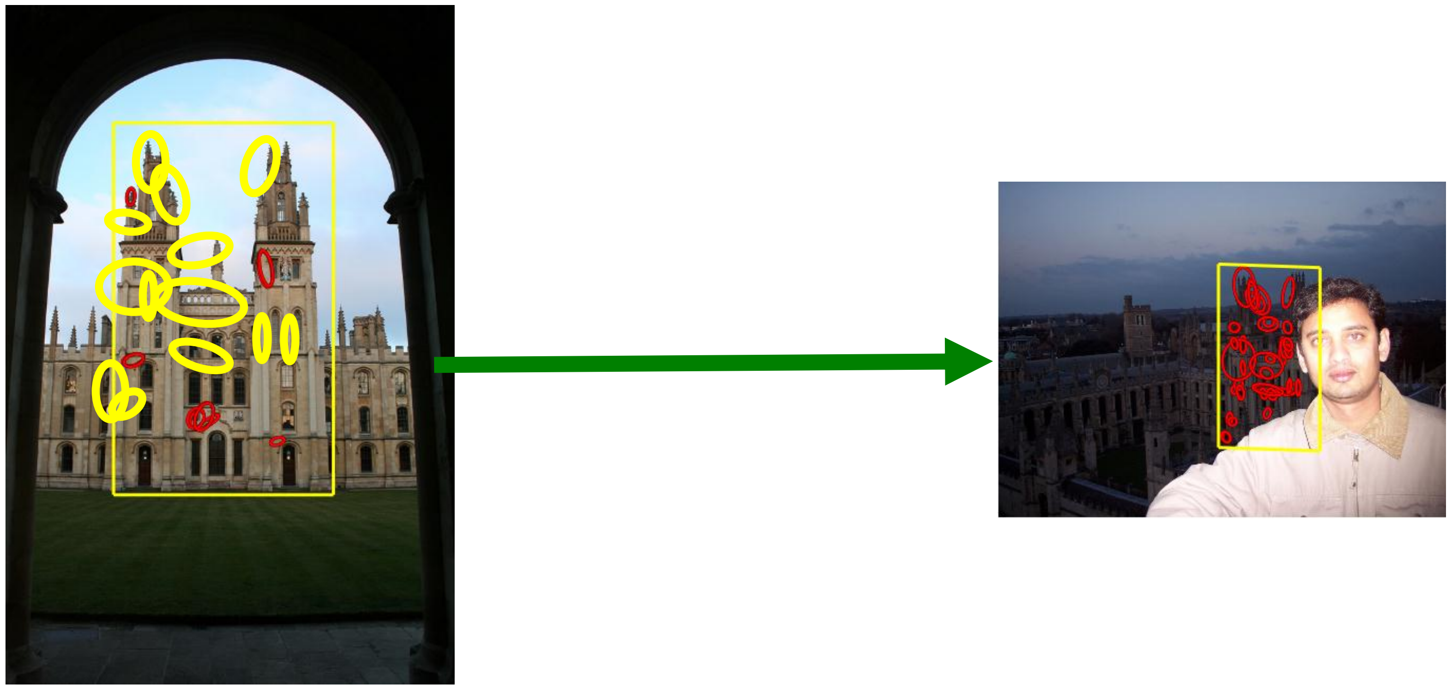
Query Expansion



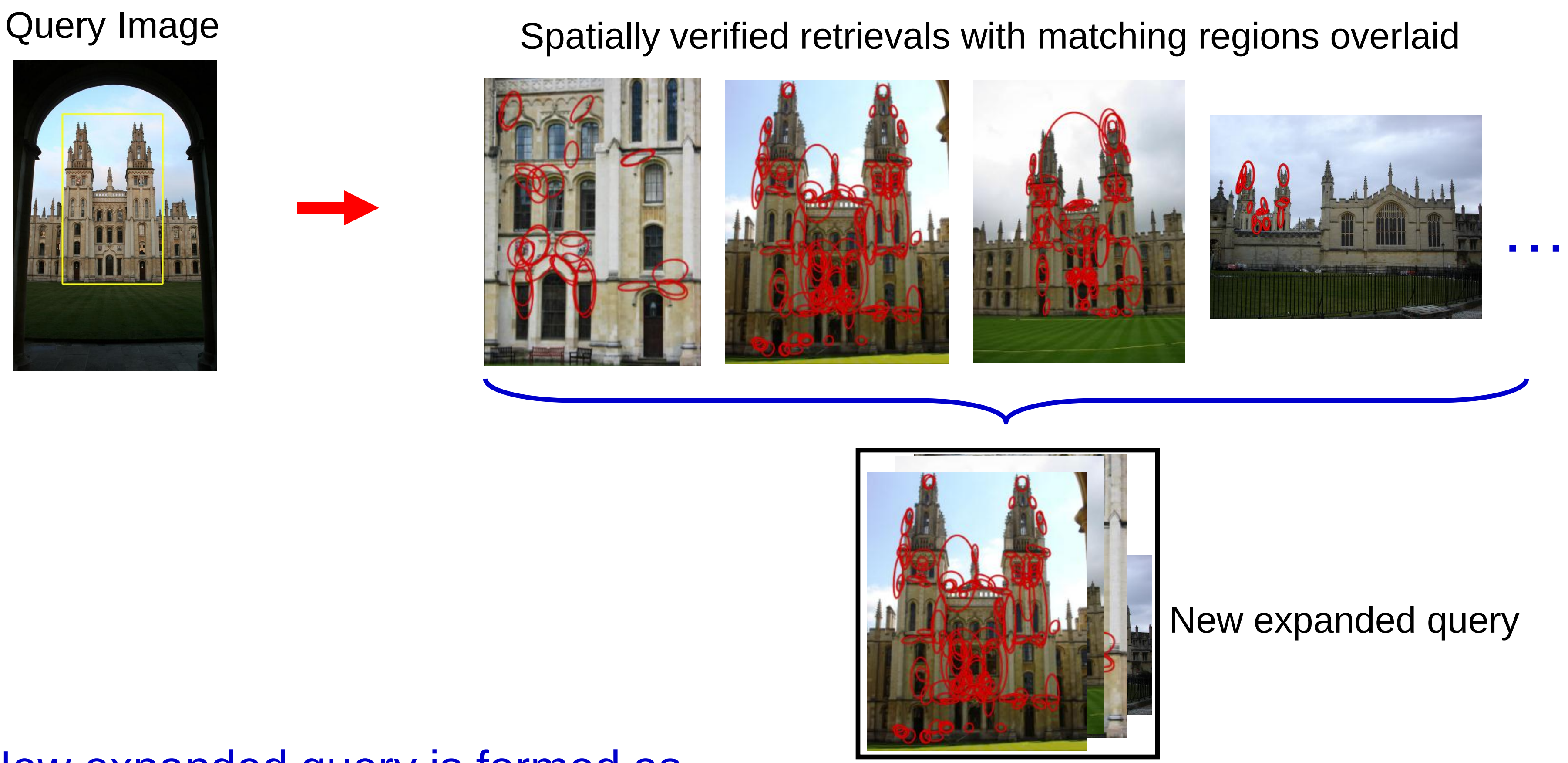
Query Expansion



Query Expansion



Query Expansion


Query Image
Spatially verified retrievals with matching regions overlaid
...
New expanded query

New expanded query is formed as

- the average of visual word vectors of spatially verified returns
- only inliers are considered
- regions are back-projected to the original query image

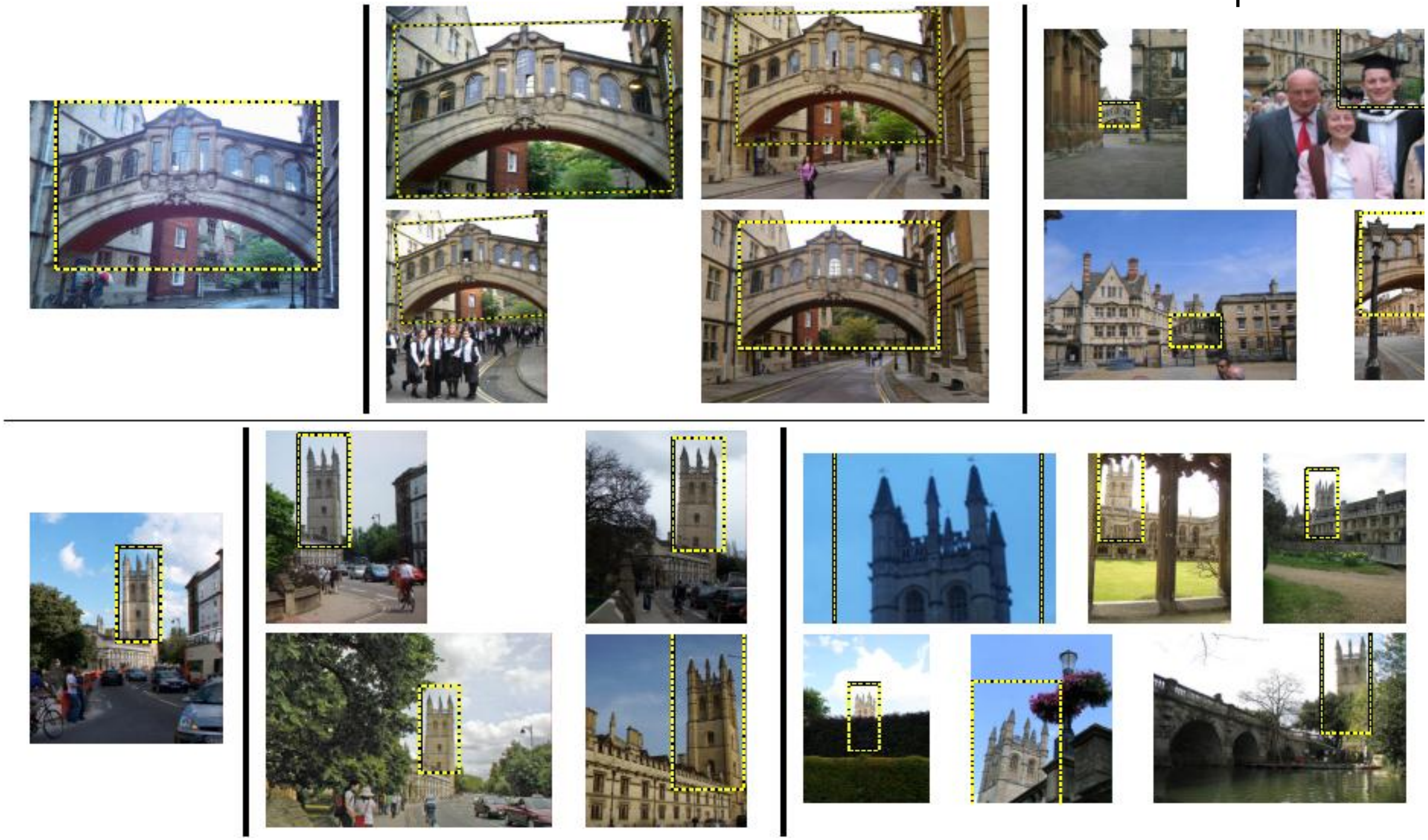
Demo

Query Expansion

Query image

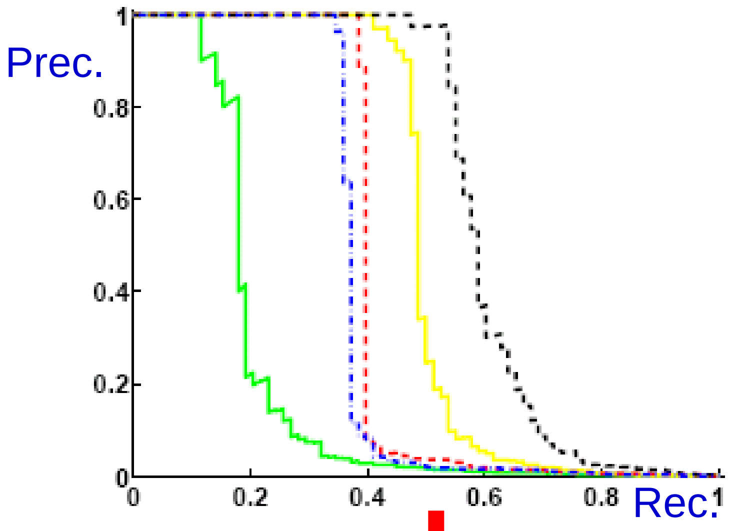
Originally retrieved

Retrieved only after expansion

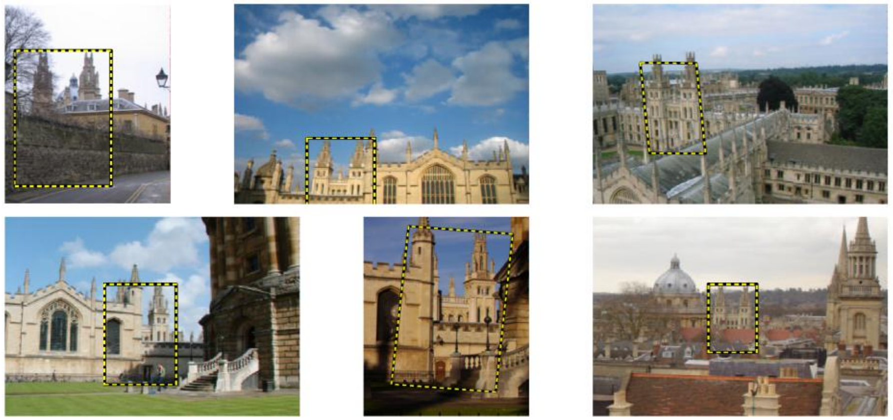


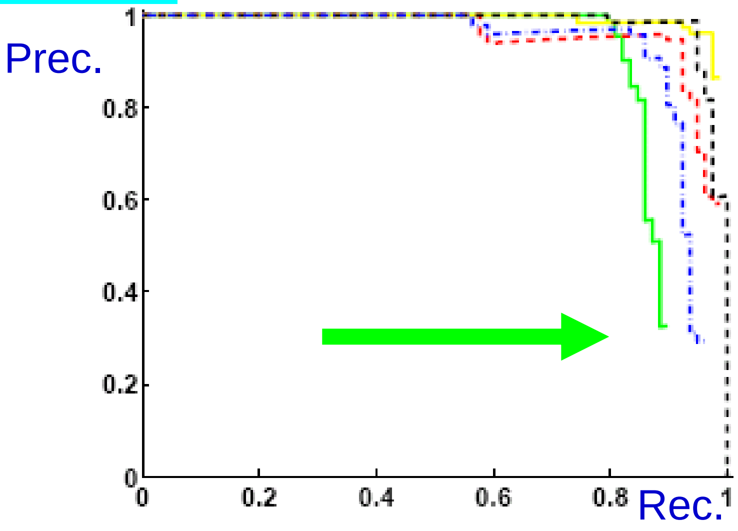
Query image

Original results (good)



Expanded results (better)

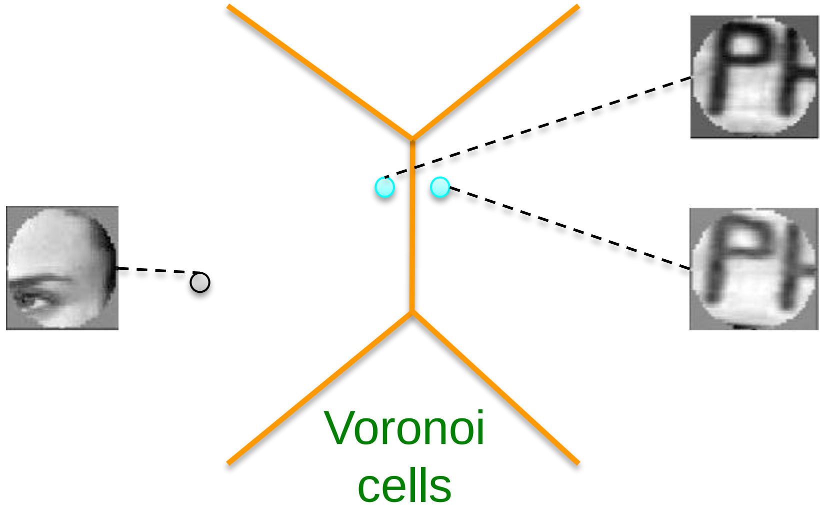




Quantization errors

Typically, quantization has a significant impact on the final performance of the system **[Sivic03,Nister06,Philbin07]**

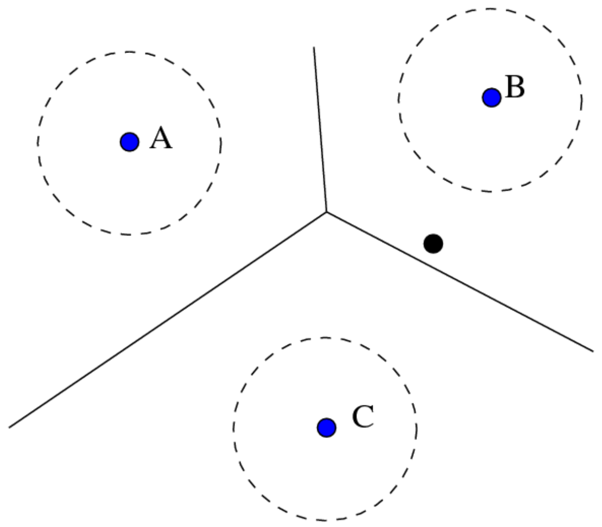
Quantization errors split features that should be grouped together and confuse features that should be separated



Overcoming quantization errors

- Soft-assign each descriptor to multiple cluster centers

[Philbin et al. 2008, Van Gemert et al. 2008]



$\left[\text{B: 1.0}\right]$ Hard Assignment

$\left[\begin{array}{l}\text{A: 0.1}\\ \text{B: 0.5}\\ \text{C: 0.4}\end{array}\right]$ Soft Assignment

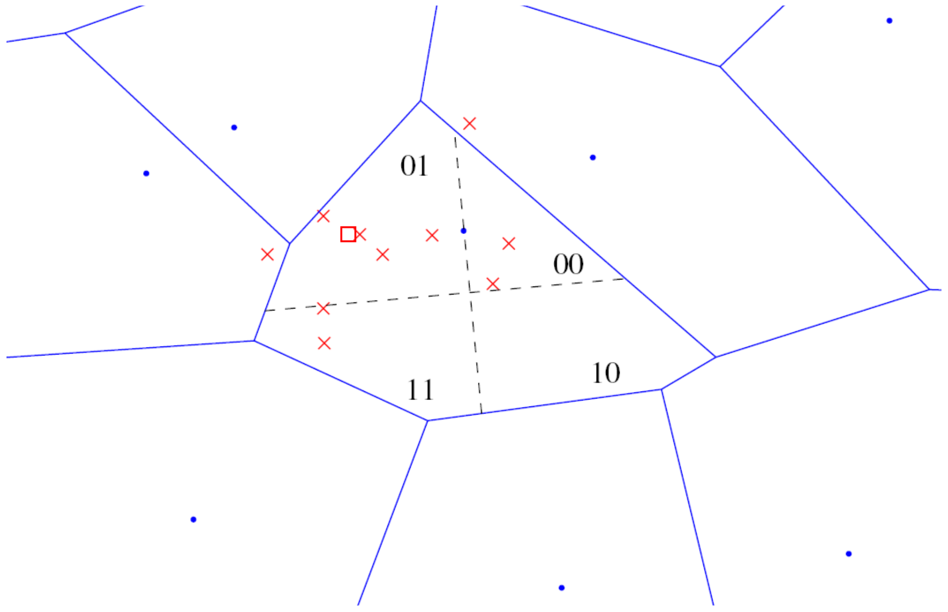
Learning a vocabulary to overcome quantization errors

[Mikulik et al. ECCV 2010, Philbin et al. ECCV 2010]

Beyond bag-of-visual-words I.

Hamming embedding [Jegou&Schmid 2008]

- Standard quantization using bag-of-visual-words
- Additional localization in the Voronoi cell by a binary signature



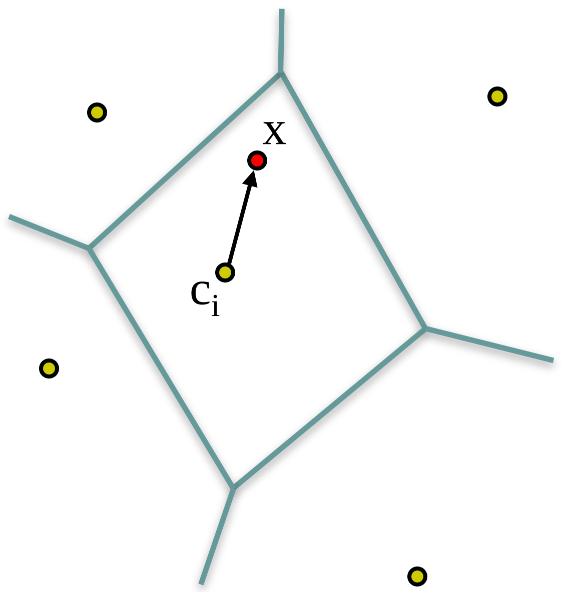
More next lecture (C. Schmid)

Beyond bag-of-visual-words II.

VLAD – Vector of locally aggregated descriptors

[Jegou et al. 2010] but see also [Perronin et al. 2010]

Measure (and quantize) the difference vectors from the cluster center.



More next lecture (C. Schmid)

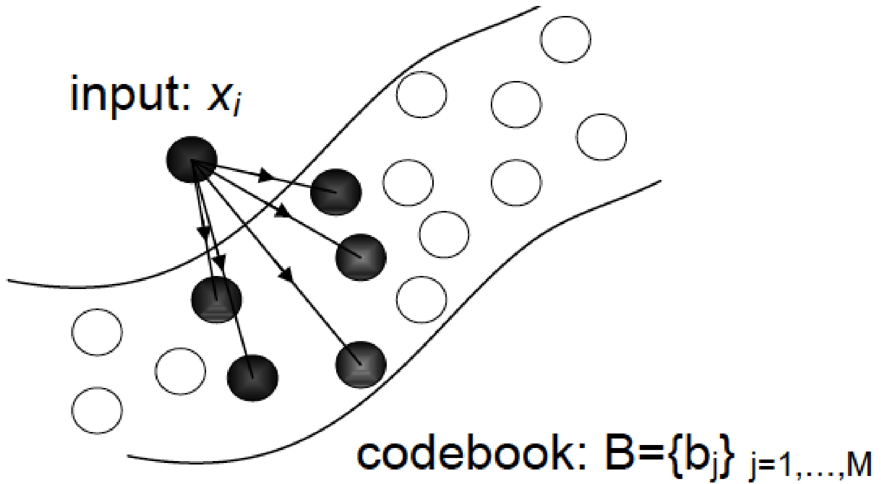
Beyond bag-of-visual-words III.

Locality-constrained linear coding.

[Wang et al. CVPR 2010]

- Represent data point as a linear combination of nearby cluster centers.
- Store the coefficients of linear combination.

Used for category-level classification.



Connection to sparse coding - more at lecture 5 (J. Ponce)

Other recent work

Learning a vocabulary to overcome quantization errors

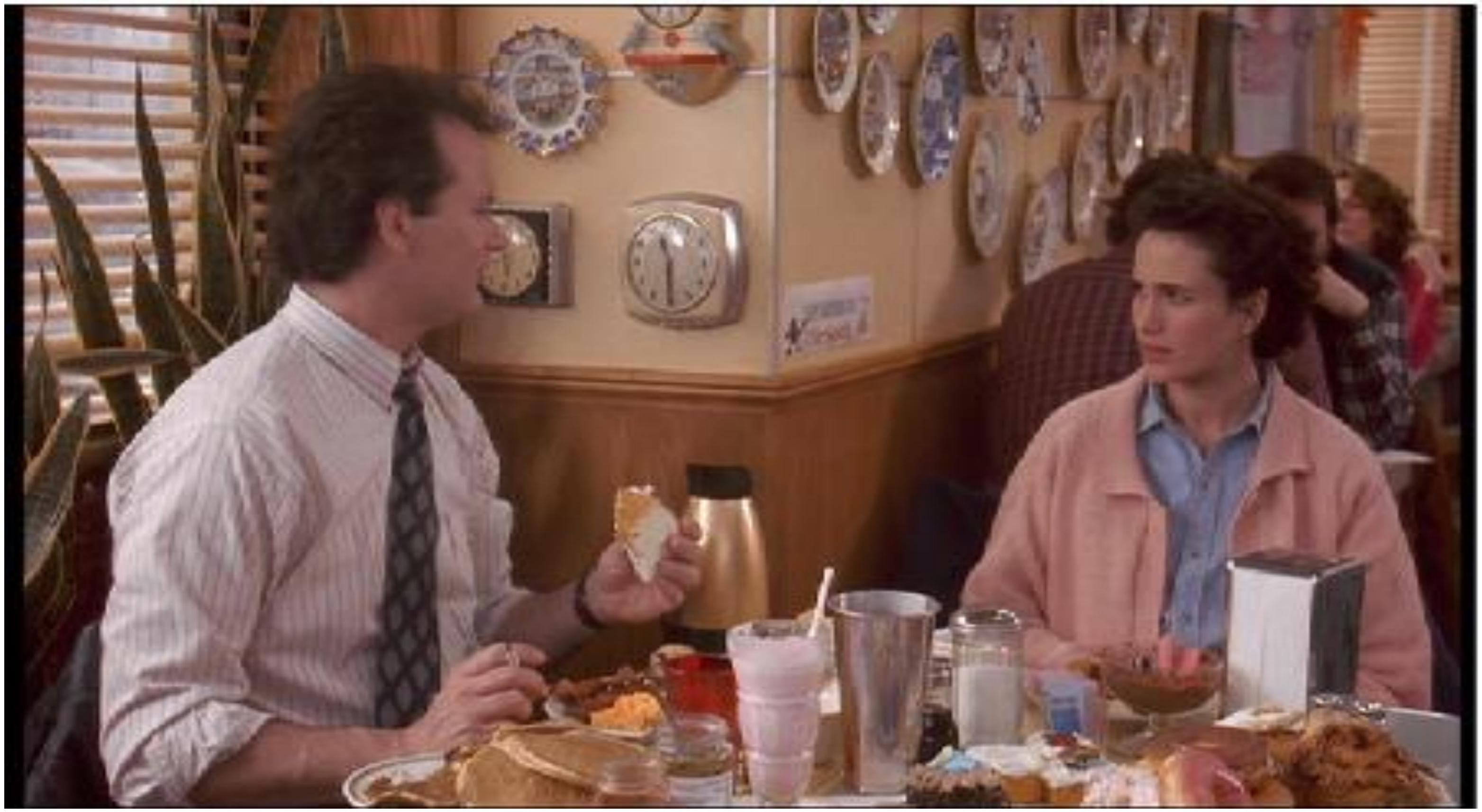
[Mikulik et al. ECCV 2010, Philbin et al. ECCV 2010]

Large scale image clustering [Chum et al. CVPR 2009, Philbin et al. IJCV 2010, Li et al., ECCV 2008]

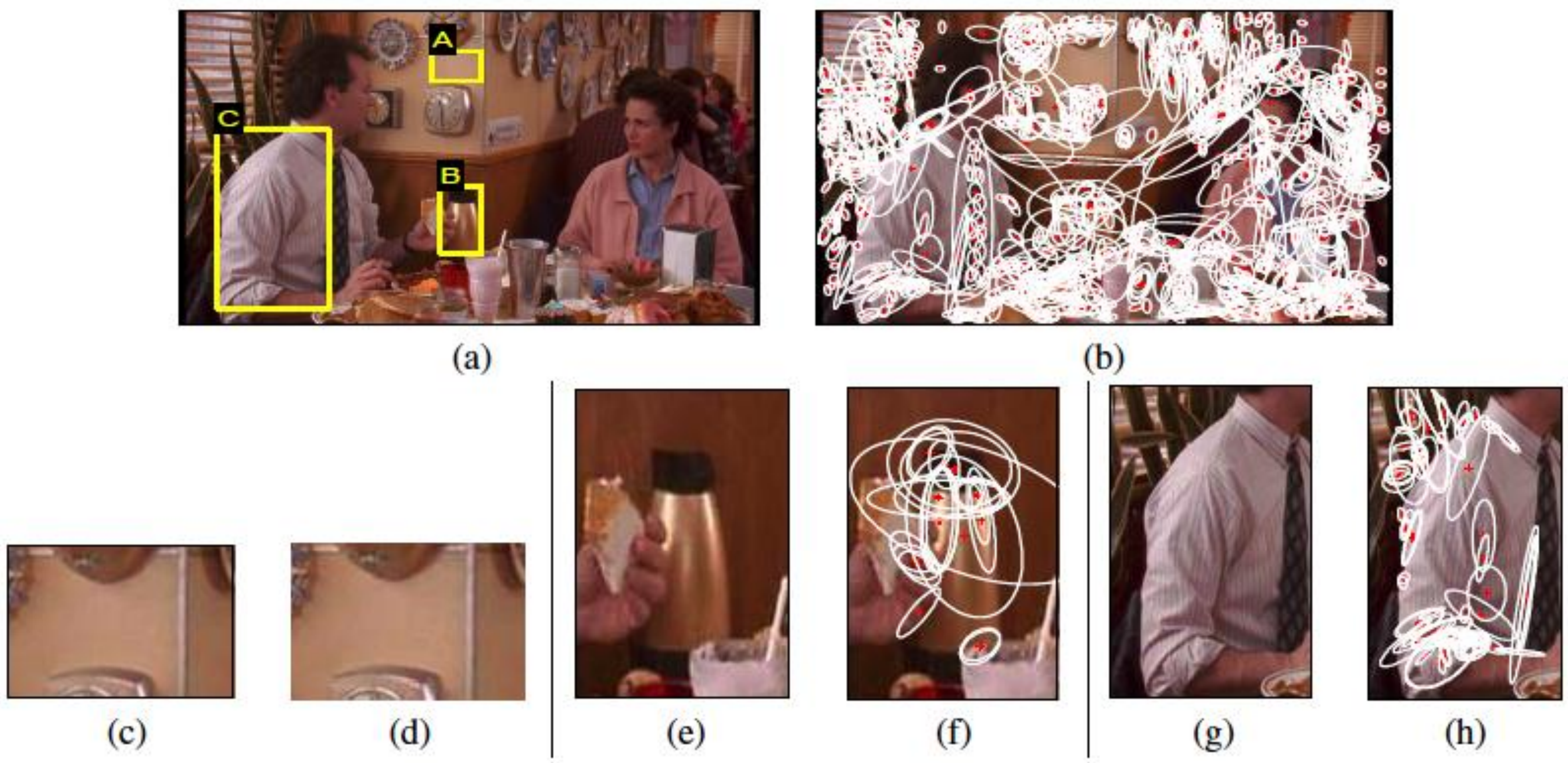
Matching in structured datasets (3D landmarks or street-view images)

[Knopp et al. ECCV 2010, Zamir&Shah ECCV 2010, Li et al. ECCV 2010, Baatz et al. ECCV 2010]

What objects/scenes local regions do not work on?



What objects/scenes local regions do not work on?



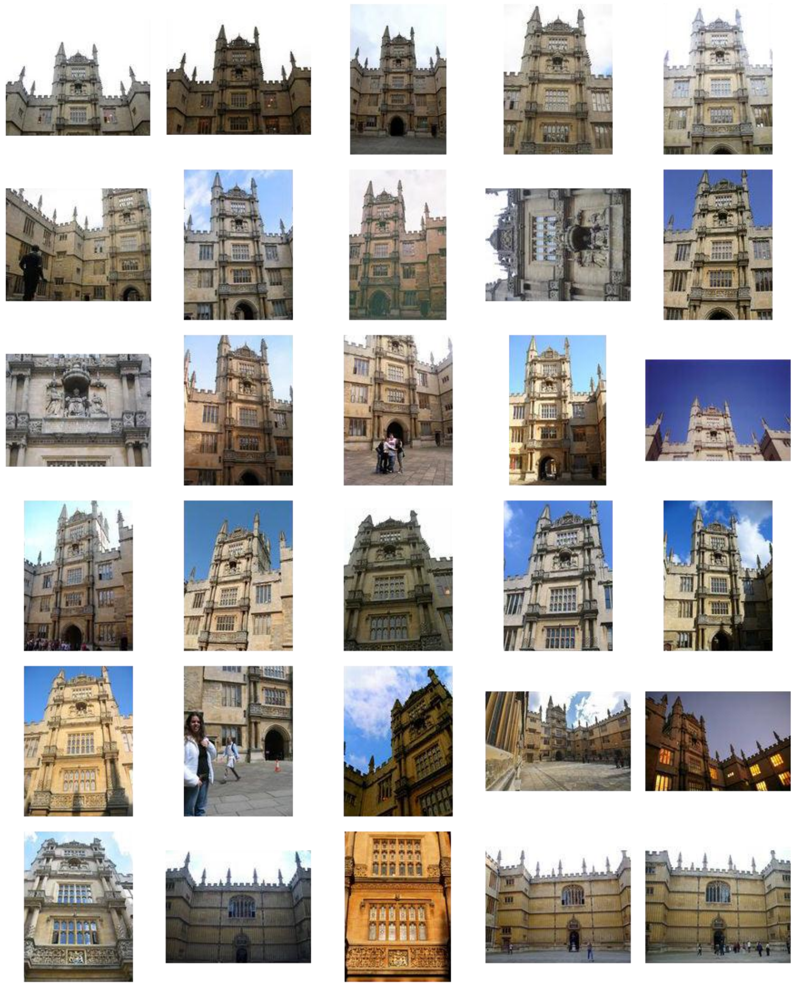
E.g. texture-less objects, objects defined by shape, deformable objects, wiry objects.

Example applications of large scale visual search and matching

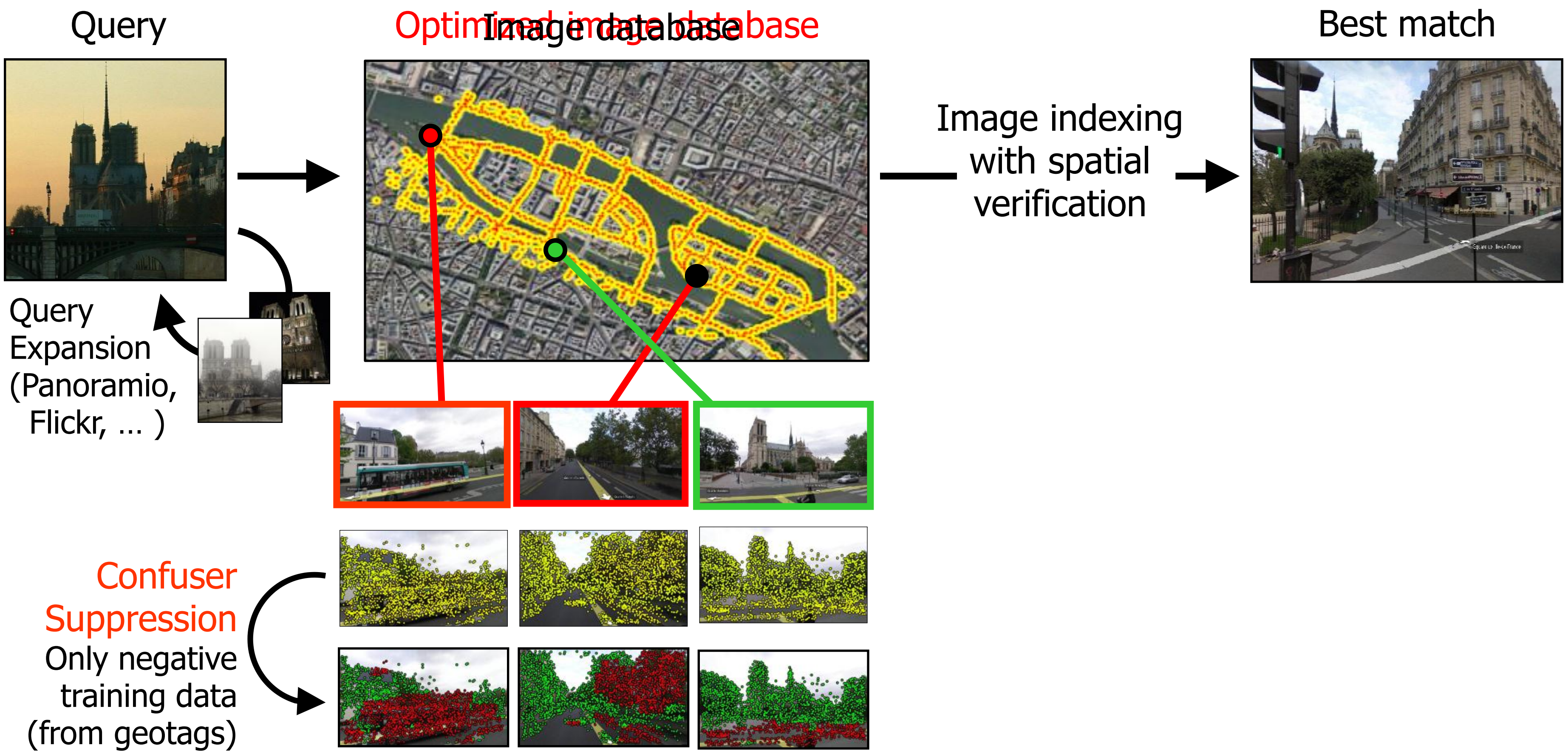
Application: Internet-based inpainting

Photo-editing using images of the same place

[Whyte, Sivic and Zisserman, 2009], but see also [Hays and Efros, 2007].

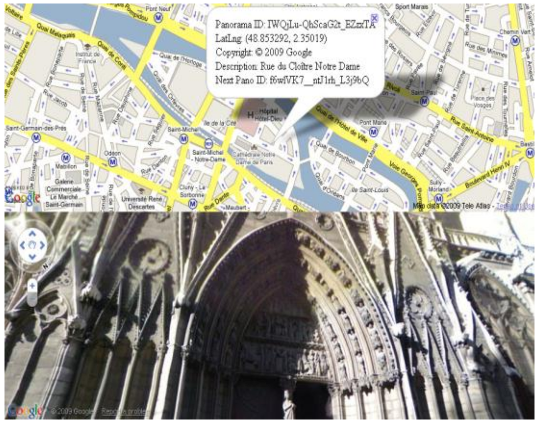


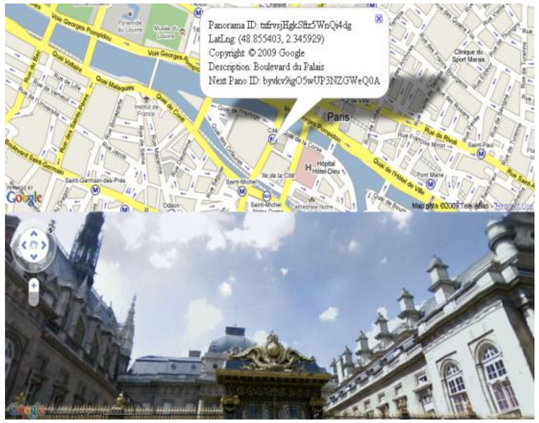
Application: place recognition (retrieval in a structured (on a map) database)

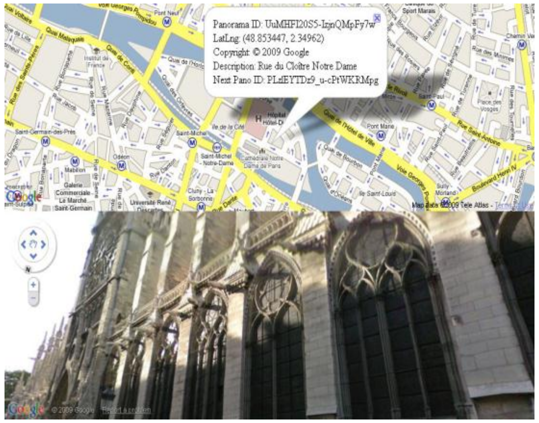

Query
Optimized image database
Image database
Best match
Image indexing with spatial verification
Query Expansion (Panoramio, Flickr, …)
Confuser Suppression
Only negative training data (from geotags)

[Knopp, Sivic, Pajdla, ECCV 2010]

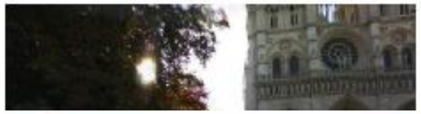
Correctly recognized examples

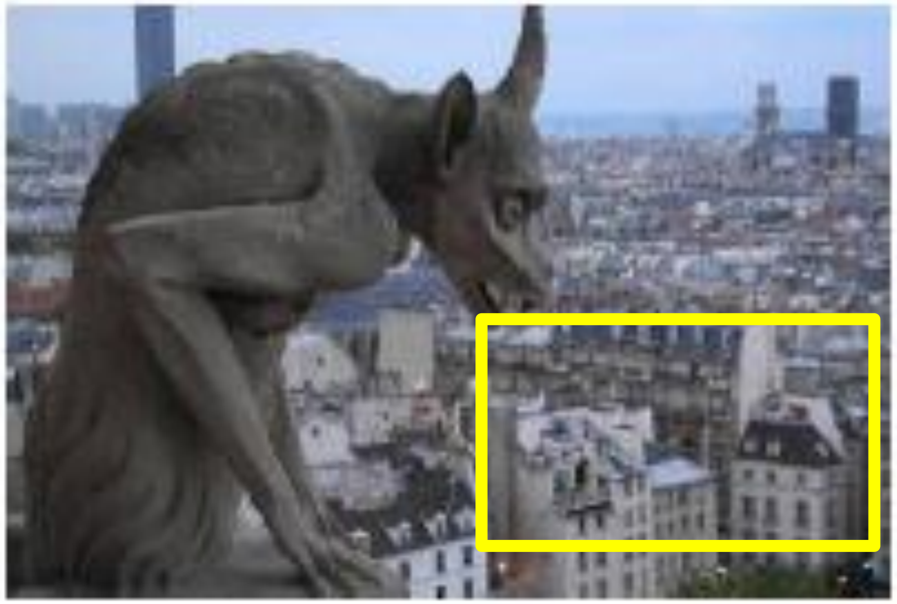

Panorama ID: IWQjLu-QhScaG2t_EZzxTA
LatLng: (48.853292, 2.35019)
Copyright: © 2009 Google
Description: Rue du Cloître Notre Dame
Next Pano ID: f6wIVK7__ntJ1rh_L3j9bQ

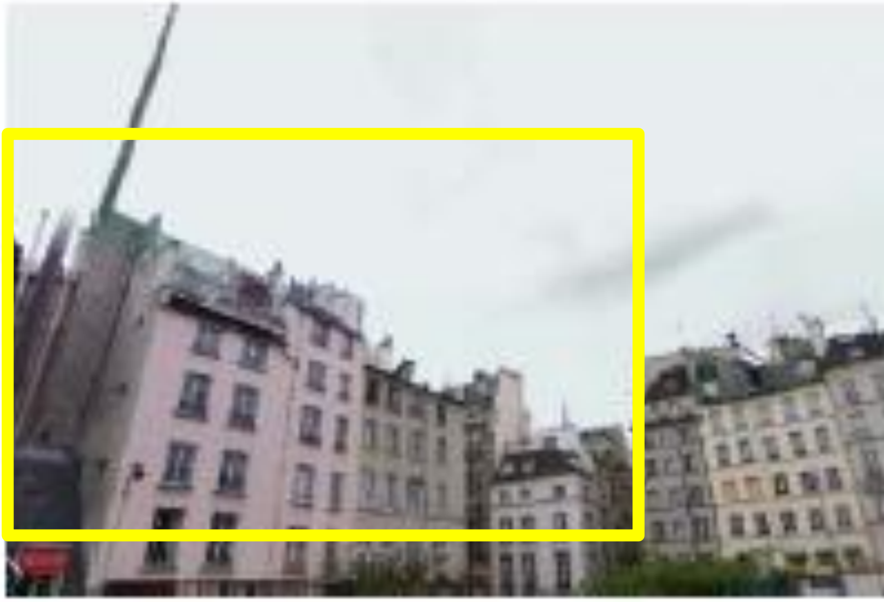

Panorama ID: txfrvsjHgkSftz5WnQi4dg
LatLng: (48.855403, 2.345929)
Copyright: © 2009 Google
Description: Boulevard du Palais
Next Pano ID: byvkv9igO5wUP3NZGWeQ0A


Panorama ID: UuMHFI20S5-IzjnQMpFy7w
LatLng: (48.853447, 2.34962)
Copyright: © 2009 Google
Description: Rue du Cloître Notre Dame
Next Pano ID: PLzlEYTDz9_u-cPtWKRMpg

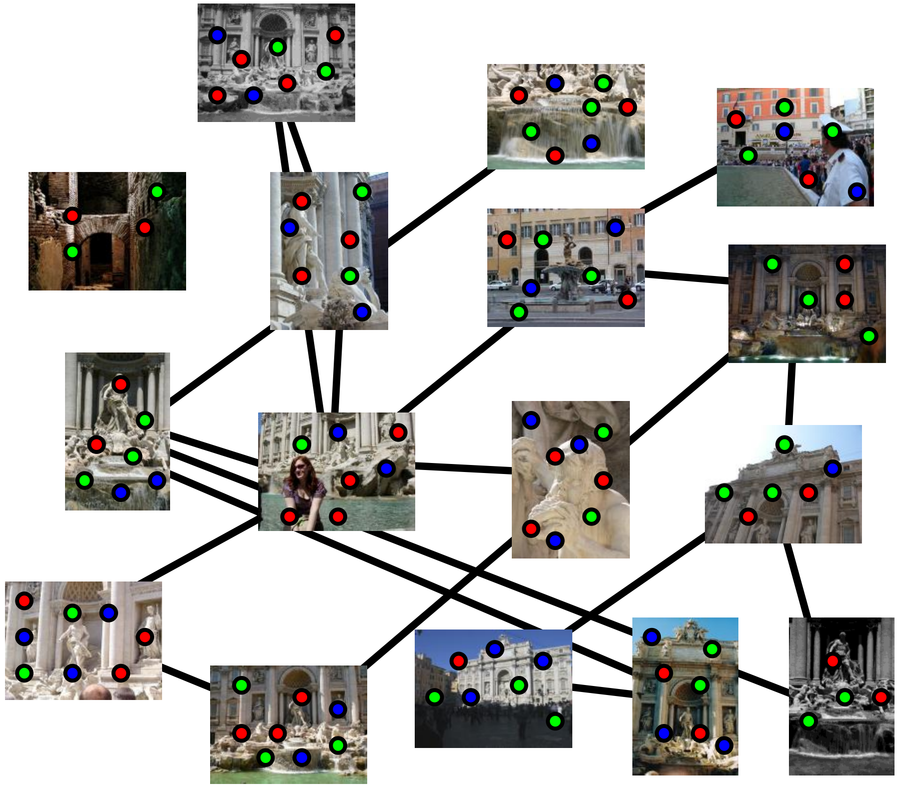
More correctly recognized examples

Query	Top ranked image	Query	Top ranked image
			





Matching and 3D reconstruction in large unstructured datasets.



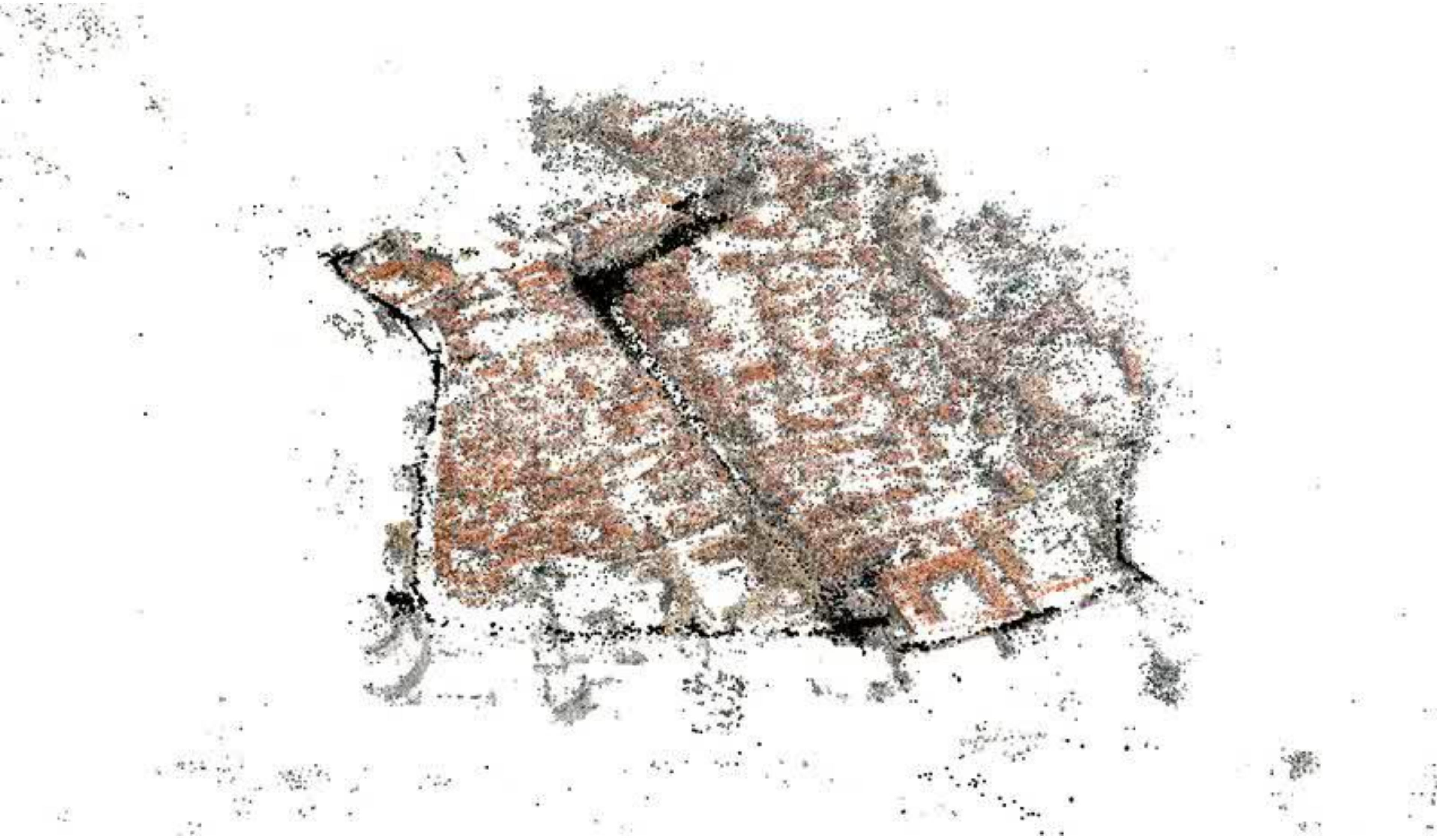
Building Rome in a Day, Sameer Agarwal, Noah Snavely, Ian Simon, Steven M. Seitz and Richard Szeliski, International Conference on Computer Vision, 2009
http://grail.cs.washington.edu/rome/

See also [Havlena, Torrii, Knopp and Pajdla, CVPR 2009].

Figure: N. Snavely

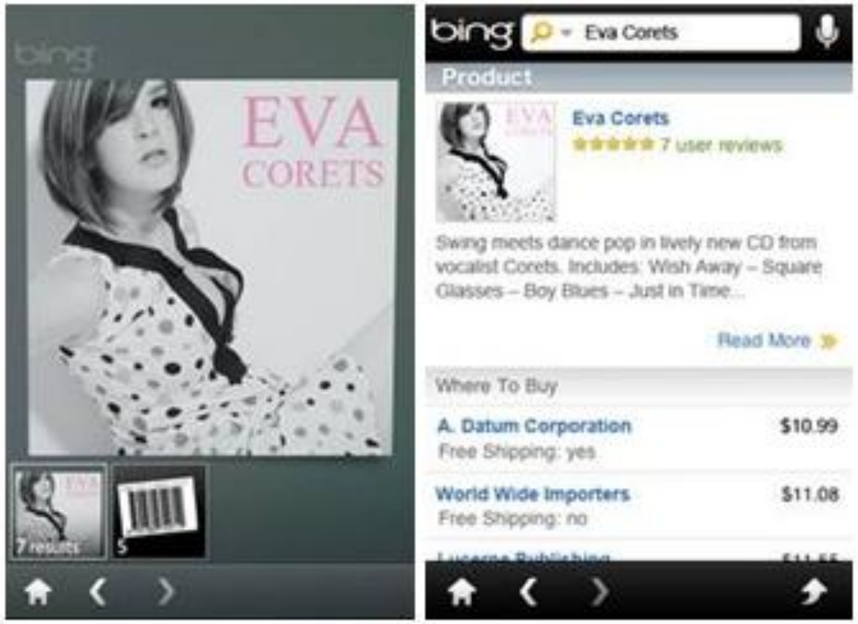
Example of the final 3D point cloud and cameras

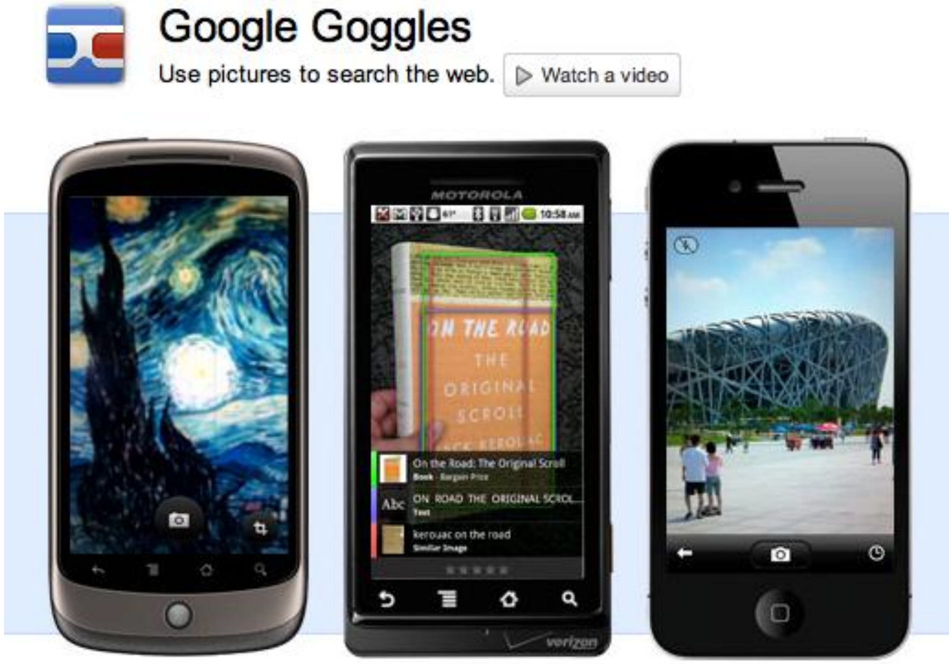
57,845 downloaded images, 11,868 registered images. This video: 4,619 images.

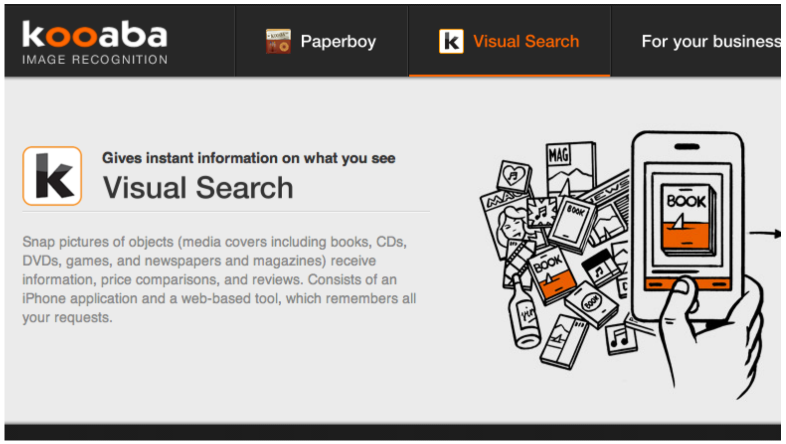


Mobile visual search

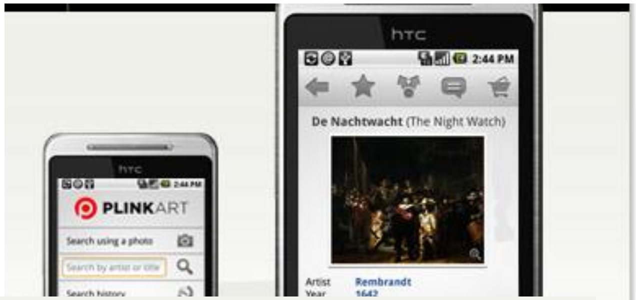
Bing visual scan


bing
Eva Corets
Product
Eva Corets
7 user reviews
Swing meets dance pop in lively new CD from vocalist Corets. Includes: Wish Away – Square Glasses – Boy Blues – Just in Time…
Read More
Where To Buy
A. Datum Corporation $10.99
Free Shipping: yes
World Wide Importers $11.08
Free Shipping: no
7 results


Google Goggles
Use pictures to search the web.
Watch a video
On the Road: The Original Scroll
ON ROAD THE ORIGINAL SCROL
kerouac on the road



Snap pictures of objects (media covers including books, CDs, DVDs, games, and newspapers and magazines) receive information, price comparisons, and reviews. Consists of an iPhone application and a web-based tool, which remembers all your requests.

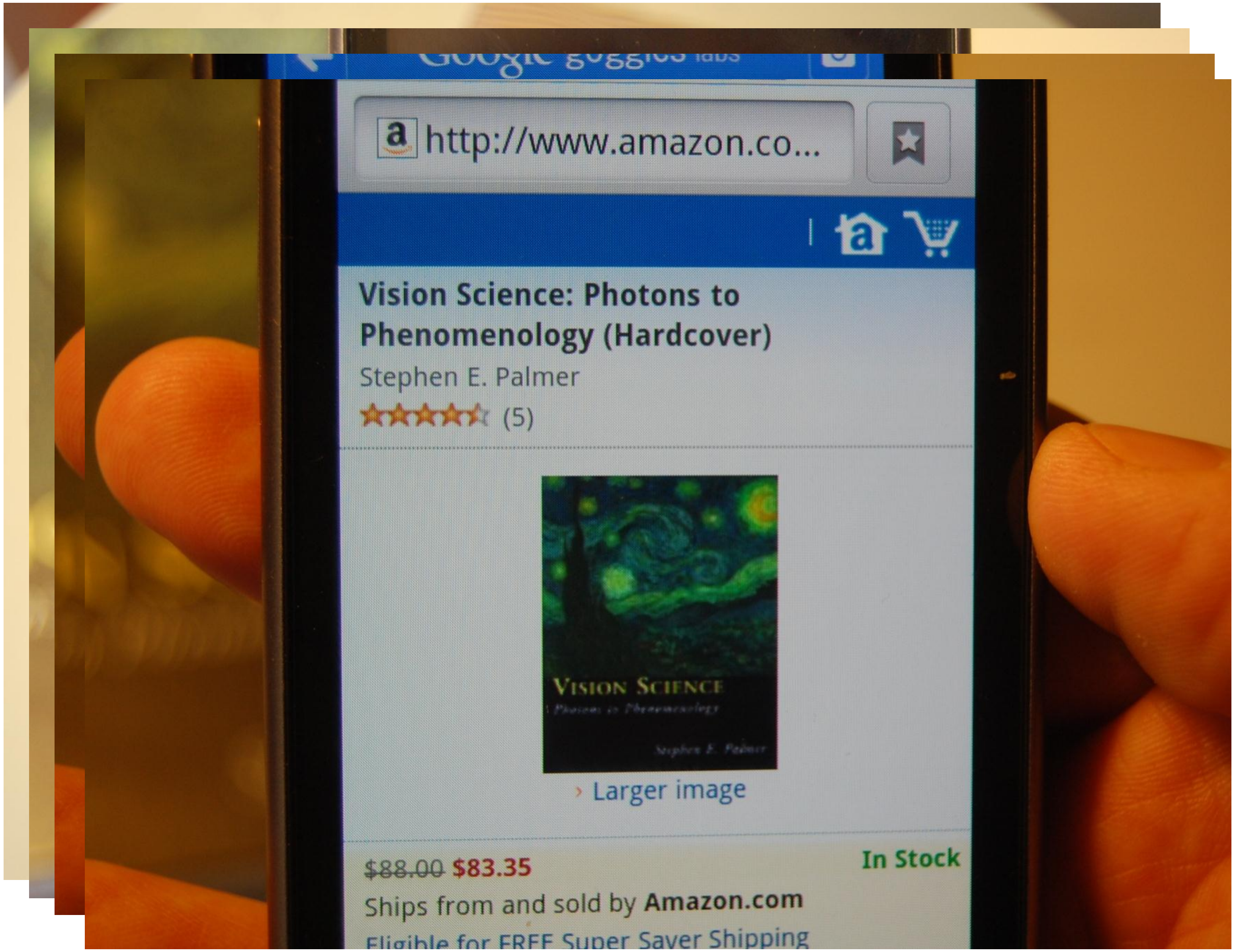


PLINKART

Plink Art is an app for your mobile phone that lets you identify almost any work of art just by taking a photo of it.

and others… Snaptell.com, Moodstocks.com

Example



Google goggles labs

http://www.amazon.co...

Vision Science: Photons to Phenomenology (Hardcover)

Stephen E. Palmer

(5)

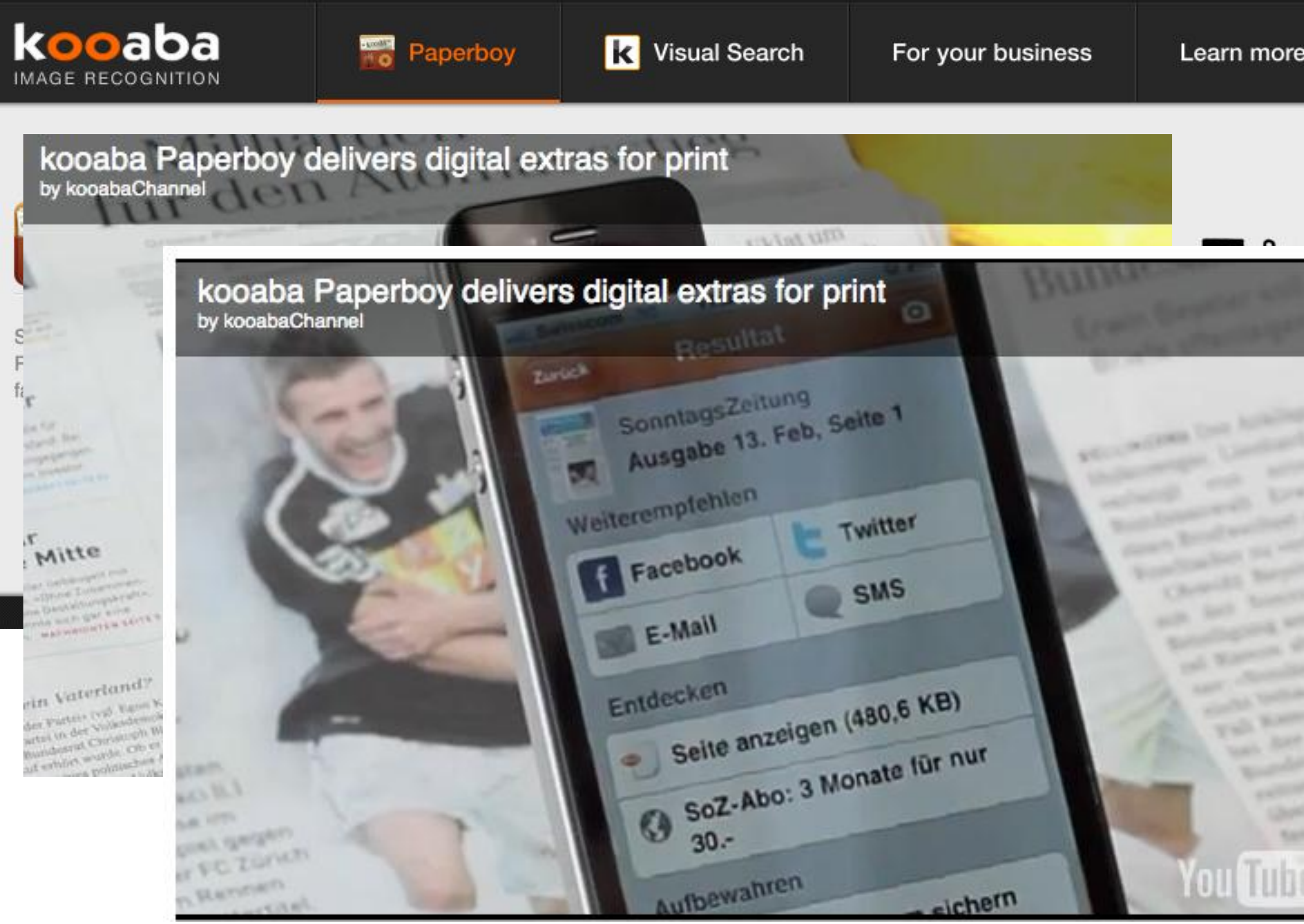
› Larger image

~~$88.00~~ $83.35 In Stock

Ships from and sold by **Amazon.com**

Eligible for FREE Super Saver Shipping

Slide credit: I. Laptev



kooaba Paperboy delivers digital extras for print
by kooabaChannel

kooaba Paperboy delivers digital extras for print
by kooabaChannel

What next?

Searchable visual memory?

Figure: iphoneography.com

What next?

Visual search for texture-less, wiry, or deformable objects..

.. also faces/people and their actions

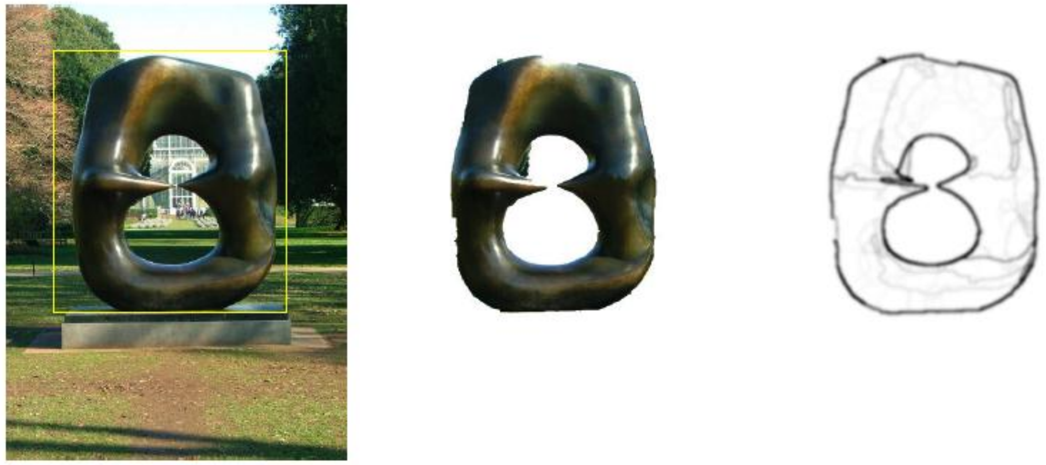
[see later lectures]

The IKEA problem

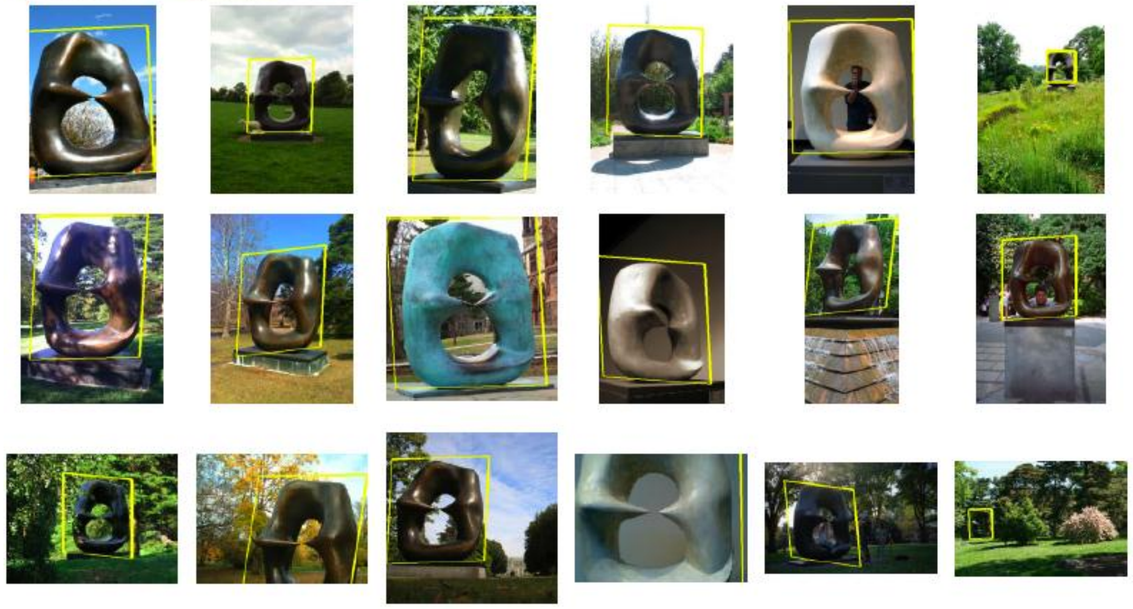
Recognize all IKEA furniture in all YouTube videos

Example a recent work:
Smooth object retrieval using a bag of boundaries
by Arandjelovic and Zisserman, ICCV 2011

Query



Retrieved matches



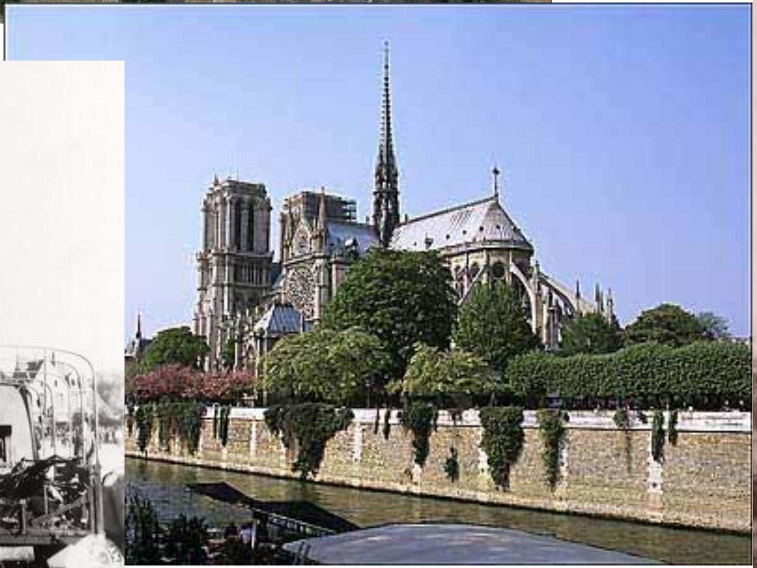
Category-level visual search [See later lectures.]

Query

same category

See also e.g. [Torresani et al. ECCV 2010]

Recognize and match non-photographic depictions

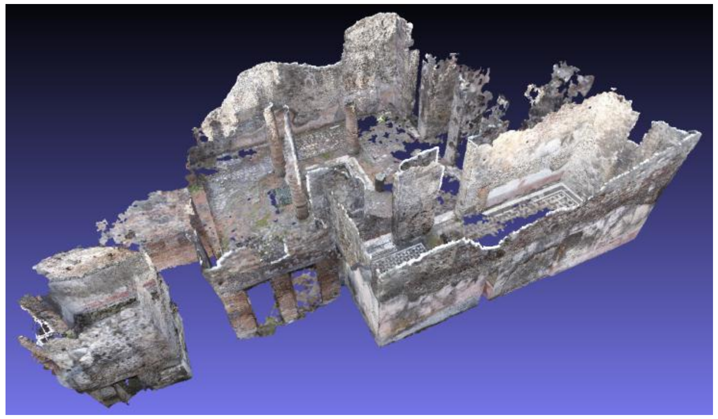


Automatic alignment of paintings and photographs depicting the same 3D scene [Russell, Sivic, Ponce and Dessales, 2011]

Inputs

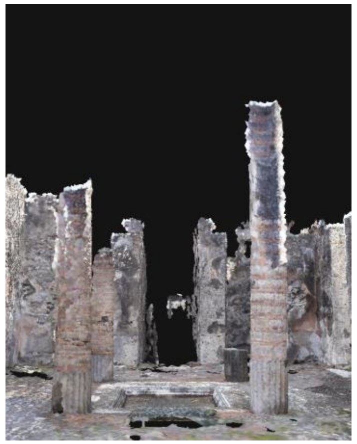
Outputs

Photographs



3D model

Painting



Viewpoint of painting