

Visual Recognition and Machine Learning Summer School

Grenoble 2012

Large scale visual search

Josef Sivic

<http://www.di.ens.fr/~josef>

INRIA, WILLOW, ENS/INRIA/CNRS UMR 8548

Laboratoire d'Informatique, Ecole Normale Supérieure, Paris

Slides with A. Zisserman

Also with some slides from: O. Chum, K. Grauman, I. Laptev, S. Lazebnik, B. Leibe, D. Lowe, J. Philbin, J. Ponce, D. Nister, C. Schmid, N. Snavely

Outline

1. Local invariant features (45 mins, C. Schmid)
2. Matching and recognition with local features (45 mins, J. Sivic)
- 3. Large scale visual search (45 mins, J. Sivic)**
4. Very large scale visual indexing (45 mins, C. Schmid)

Practical session – Instance-level recognition and search (60 mins)

Outline

Efficient visual search

Approximate nearest neighbour matching

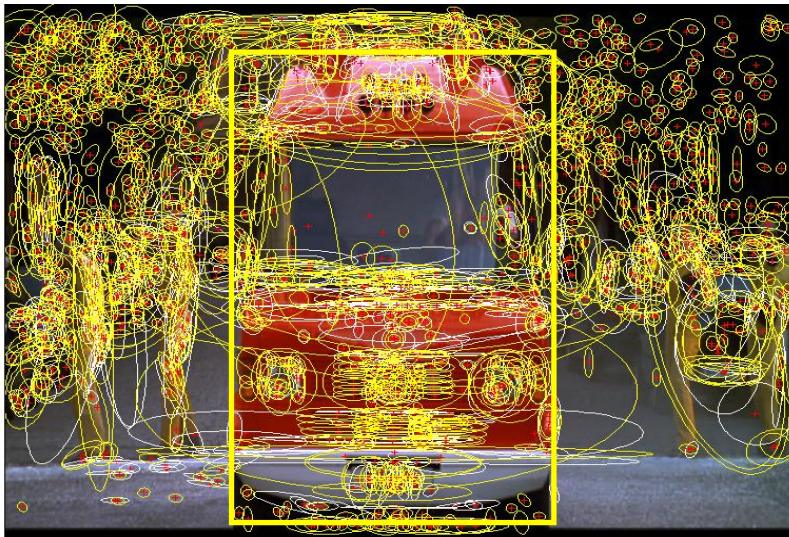
Bag-of-visual-words representation

Efficient visual search and extensions

Beyond bag-of-visual-words representations

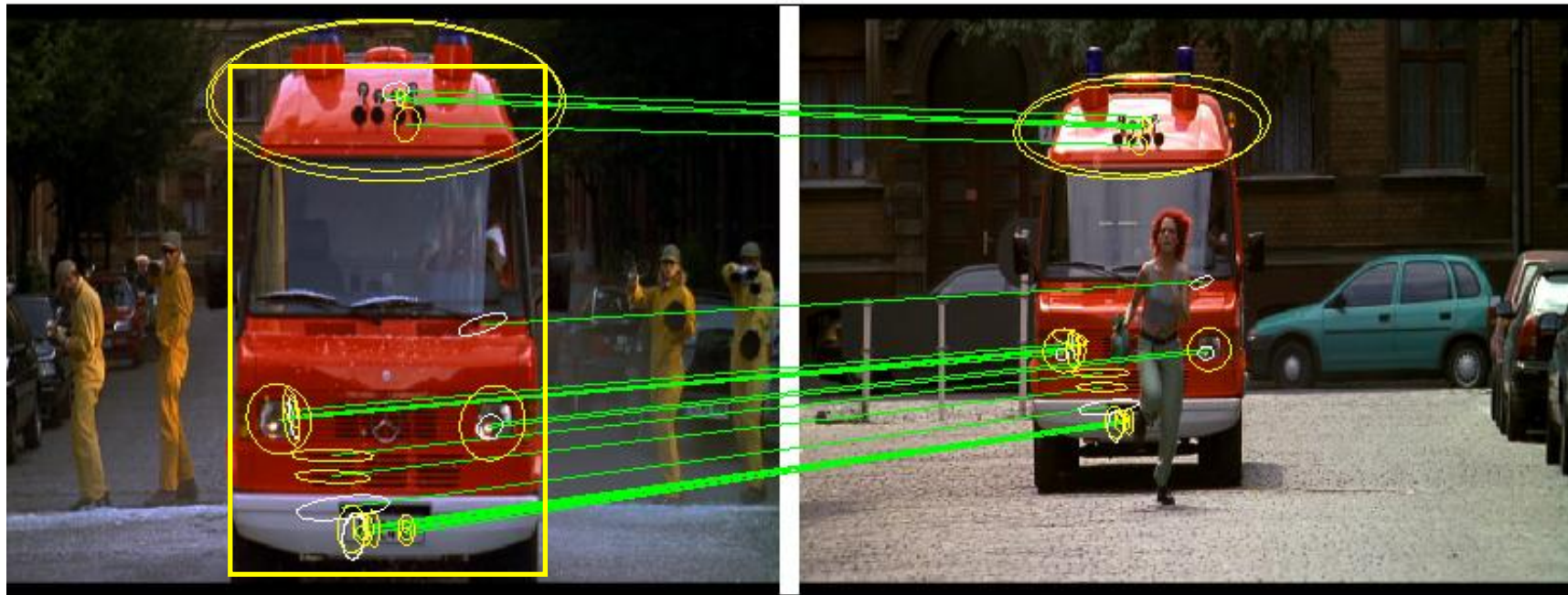
Applications

Example: 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?

History of “large scale” visual search with local regions

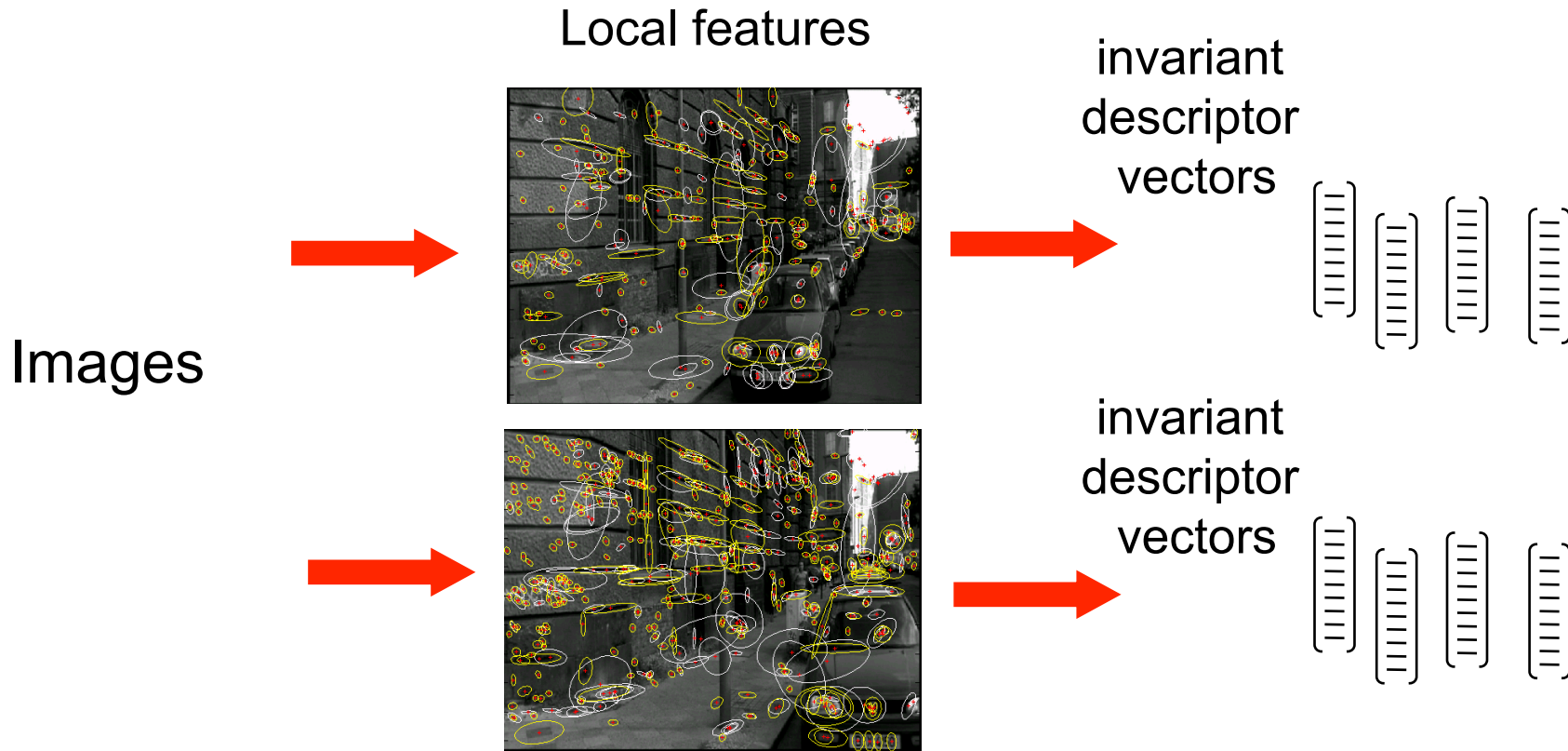
Schmid and Mohr '97	– 1k images
Sivic and Zisserman'03	– 5k images
Nister and Stewenius'06	– 50k images (1M)
Philbin et al.'07	– 100k images
Chum et al.'07 + Jegou et al.'07	– 1M images
Chum et al.'08	– 5M images
Jegou et al. '09	– 10M images
Jegou et al. '10	– ~100M images

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)

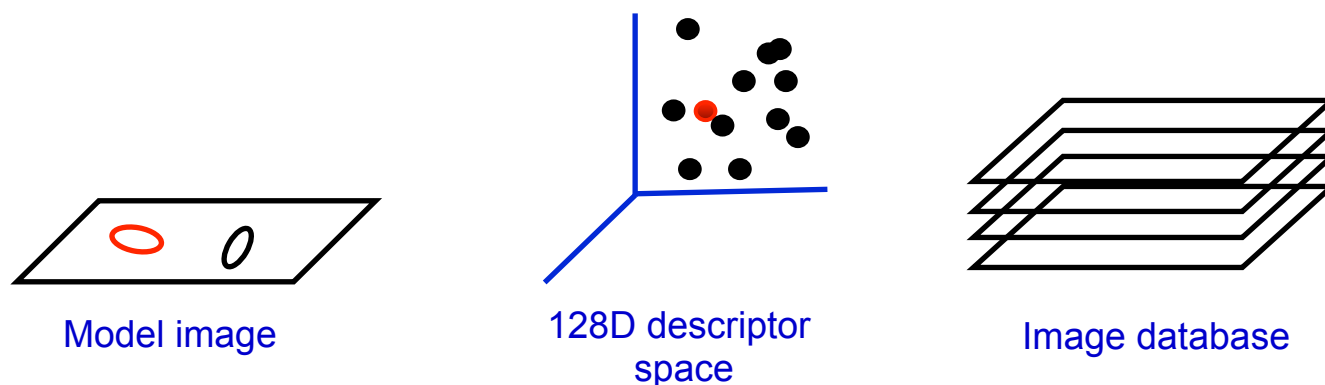
Strategy I: Efficient approximate NN search



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:

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

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: $O(M \cdot N \cdot D)$

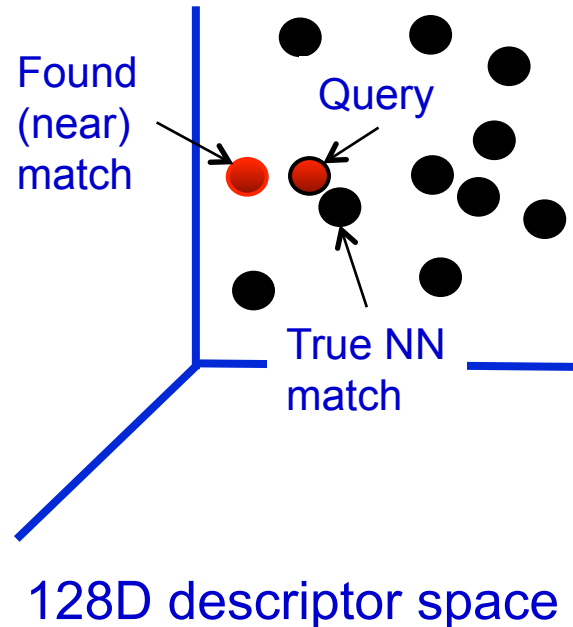
Example:

- Matching two images ($N=1$), each having 1000 SIFT descriptors
Nearest neighbors search: 0.4 s (2 GHz CPU, implementation in C)
- Memory footprint: $1000 \cdot 128 = 128\text{kB}$ / image

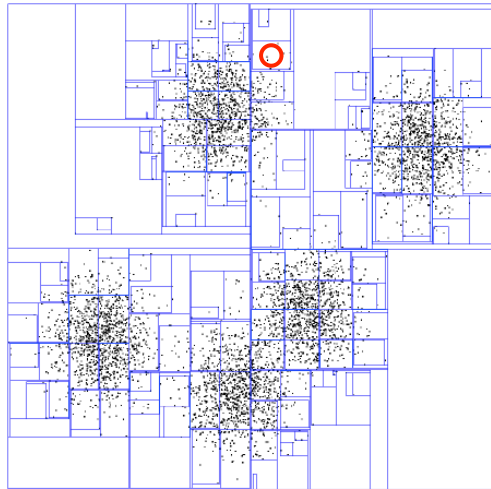
# of images	CPU time	Memory req.
N = 1,000 ...	~7min	(~100MB)
N = 10,000 ...	~1h7min	(~ 1GB)
...		
N = 10^7	~115 days	(~ 1TB)
...		
All images on Facebook:		
N = 10^{10} ...	~300 years	(~ 1PB)

Finding *approximate* nearest neighbour vectors

- Approximate method is not guaranteed to find the nearest neighbour.
- Can be much faster, but at the cost of missing some nearest matches

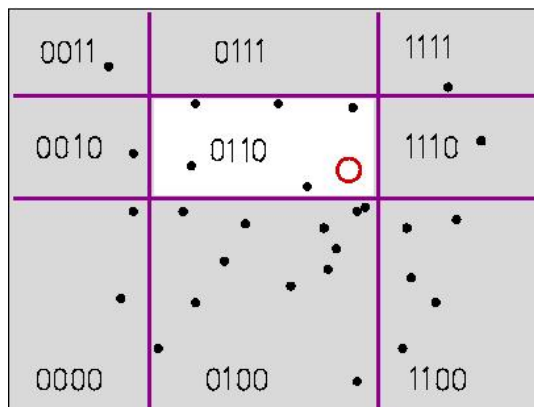


Approximate nearest neighbor search



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

Extended to multiple randomized trees in : [Muja & Lowe, 2009]



(1)

(2)

(3)

(4)

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]

Can reduce the complexity of the search, e.g. $O(\log N)$ for k-d tree.

But at the cost of missing some nearest matches.

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

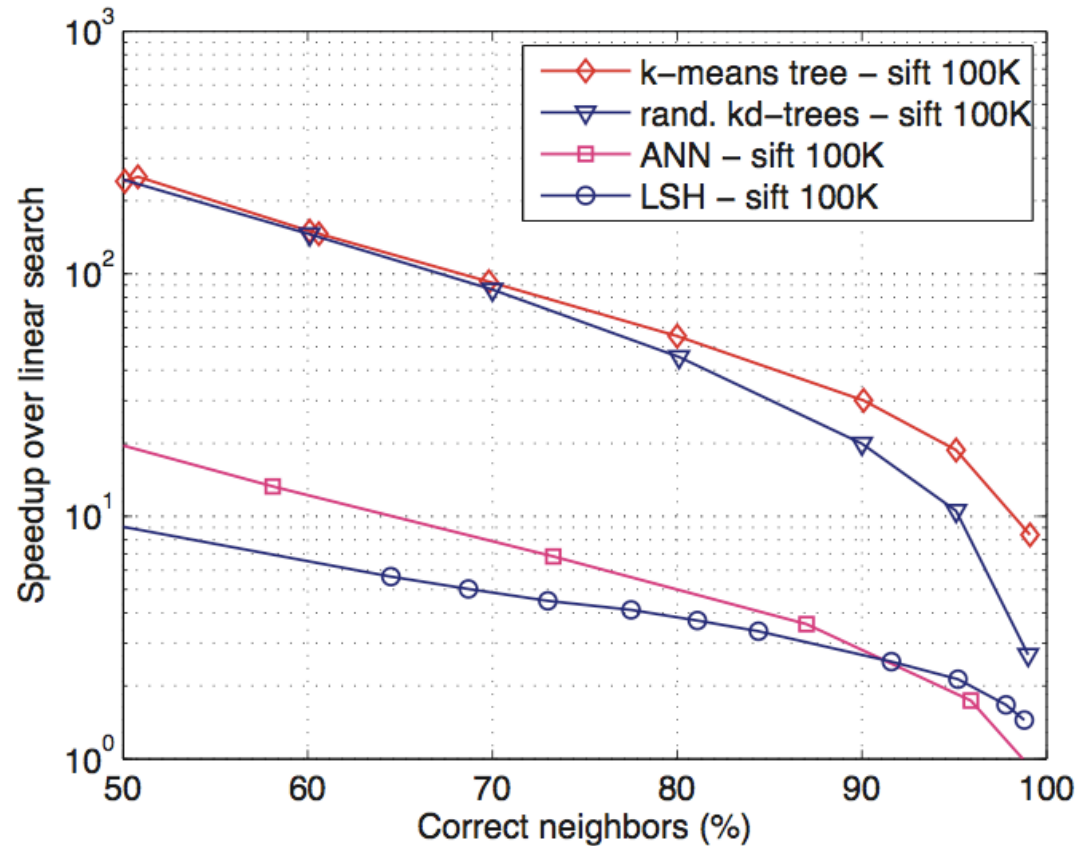
mariusm@cs.ubc.ca, lowe@cs.ubc.ca

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*.

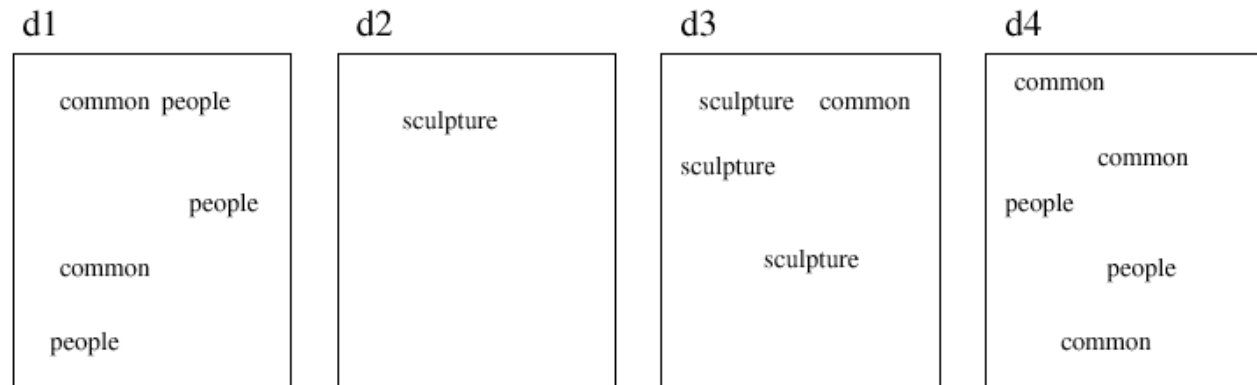
So far ...

- 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 \implies 1TB of memory

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

Indexing text with inverted files

Document collection:

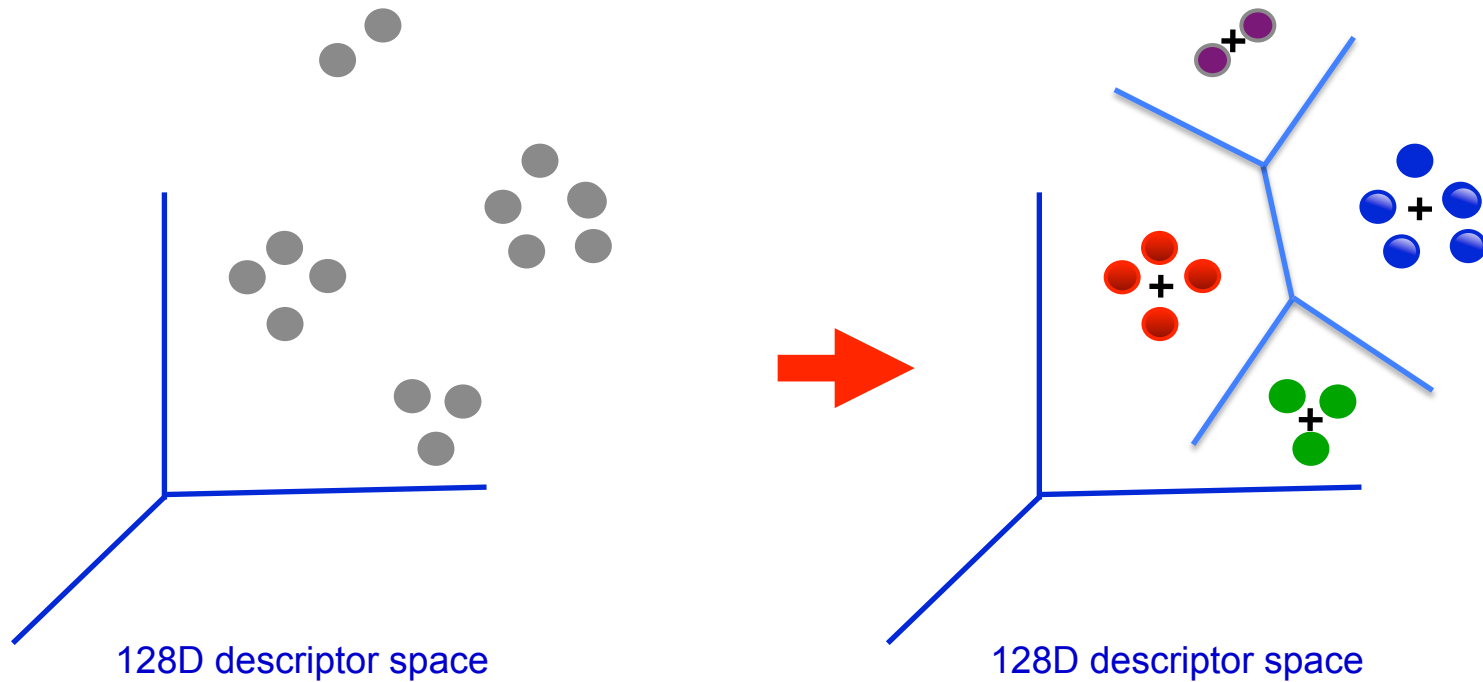


Inverted file:

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



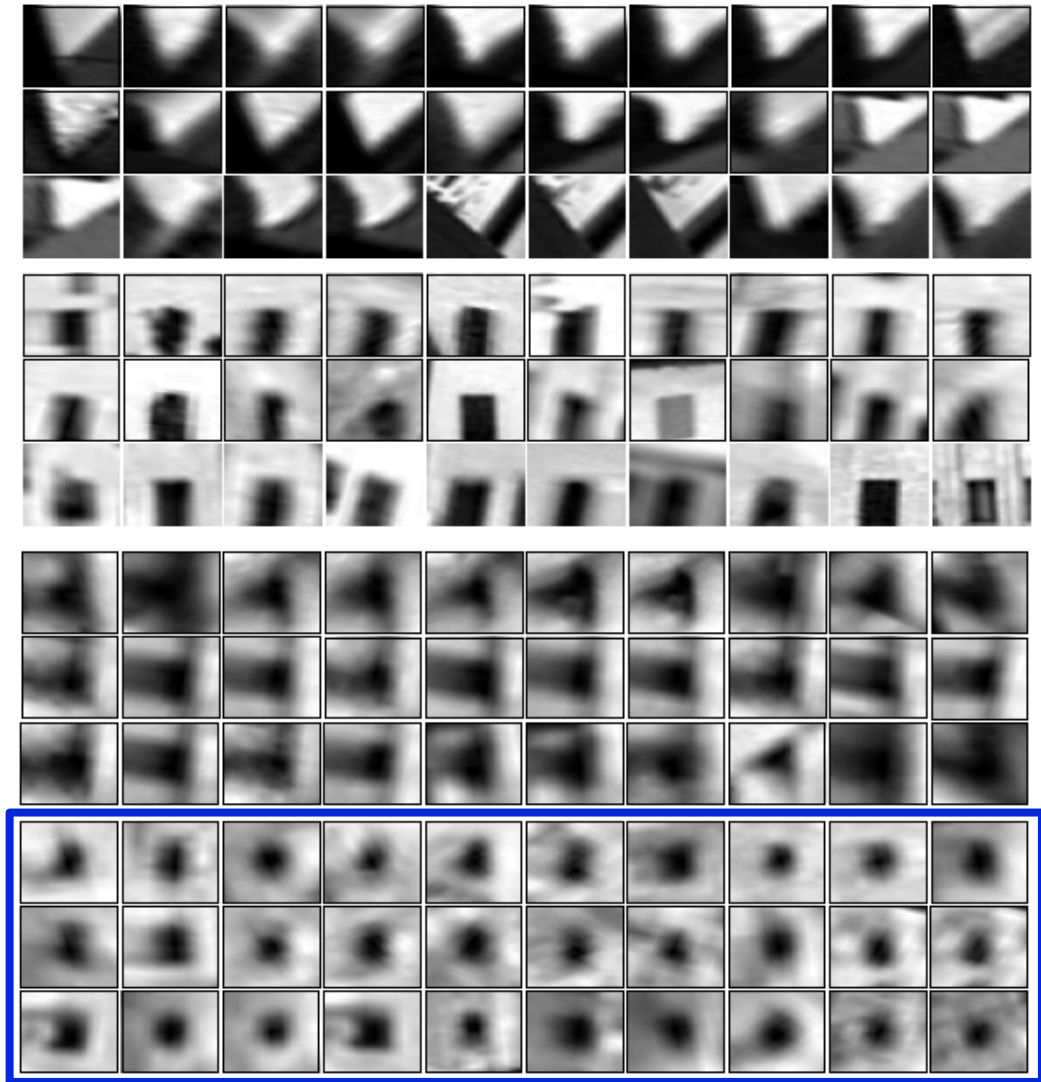
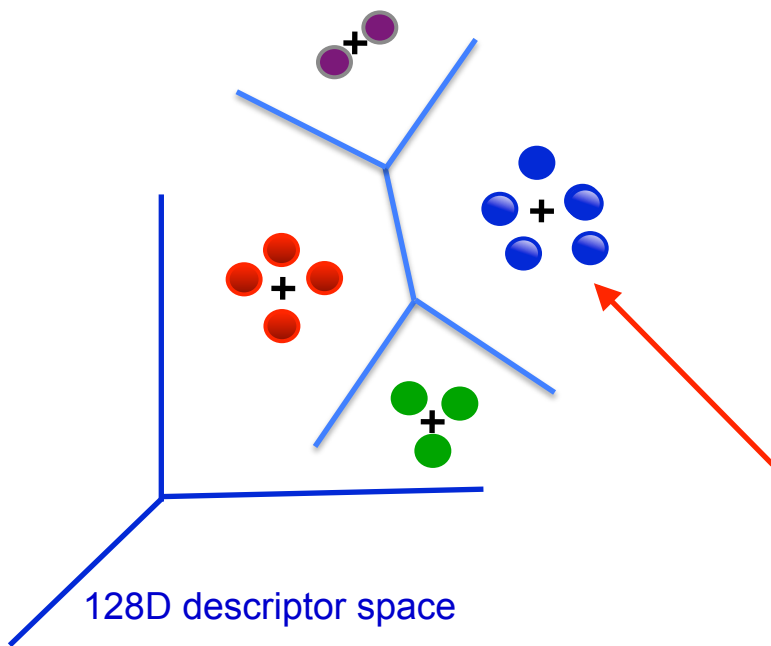
Vector quantize descriptors

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

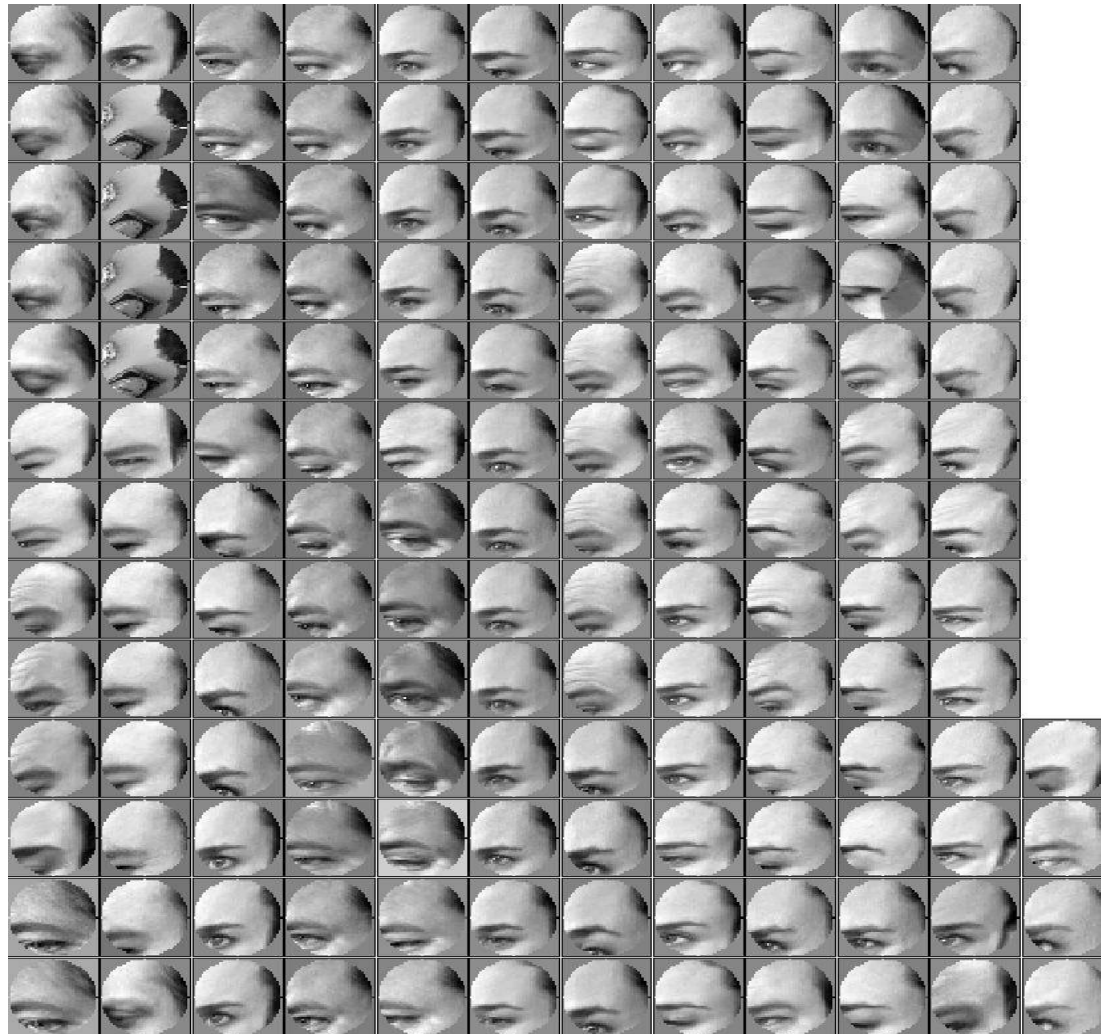
[Sivic and Zisserman, ICCV 2003]

Visual words

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

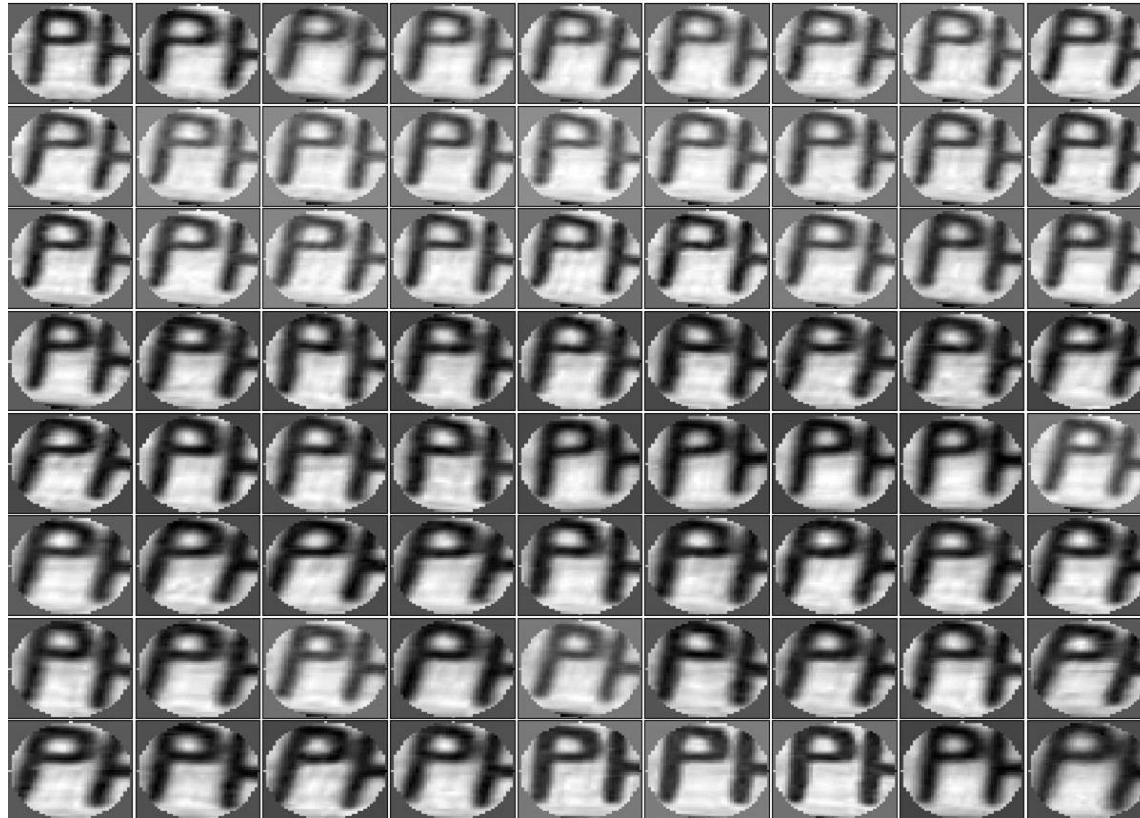


Samples of visual words (clusters on SIFT descriptors):



More specific example

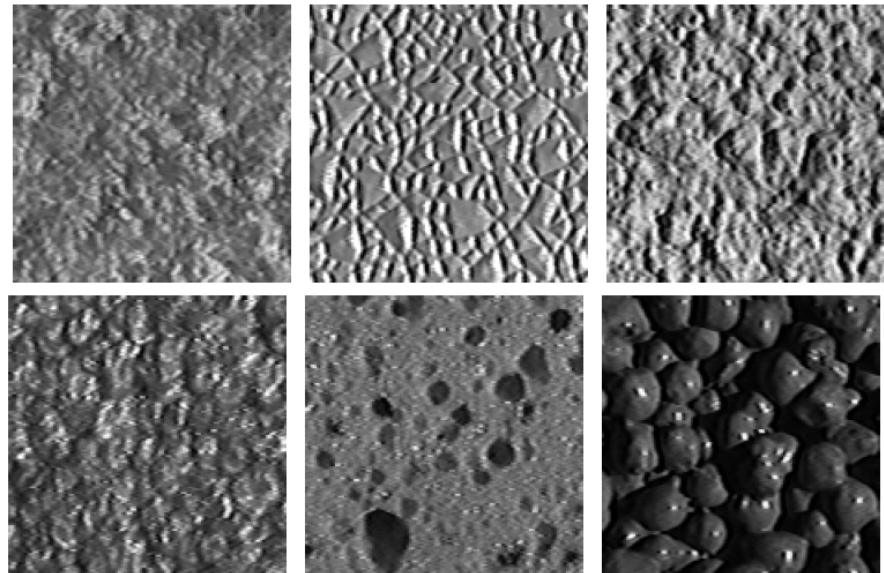
Samples of visual words (clusters on SIFT descriptors):



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.



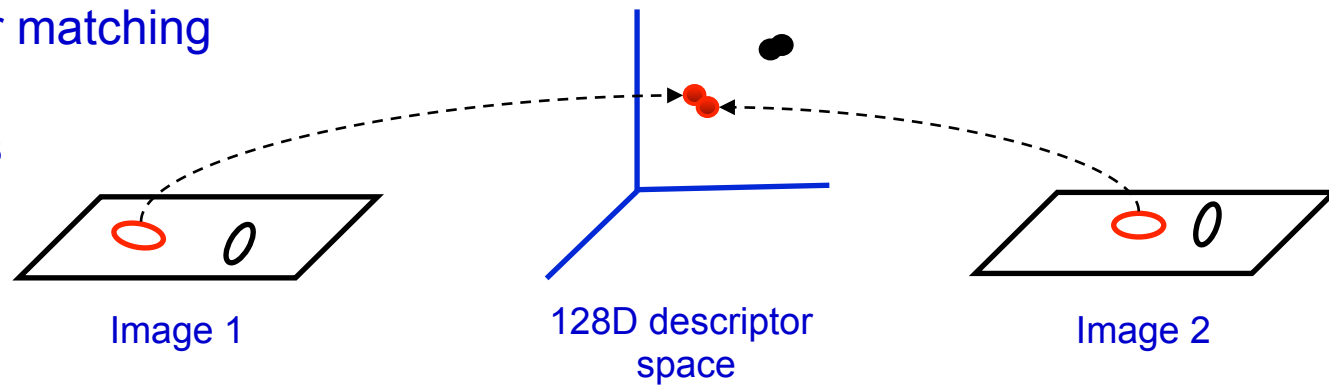
Leung & Malik 1999; Varma & Zisserman, 2002; Lazebnik, Schmid & Ponce, 2003;

Visual words: quantize descriptor space

Sivic and Zisserman, ICCV 2003

Nearest neighbour matching

- expensive to do for all frames

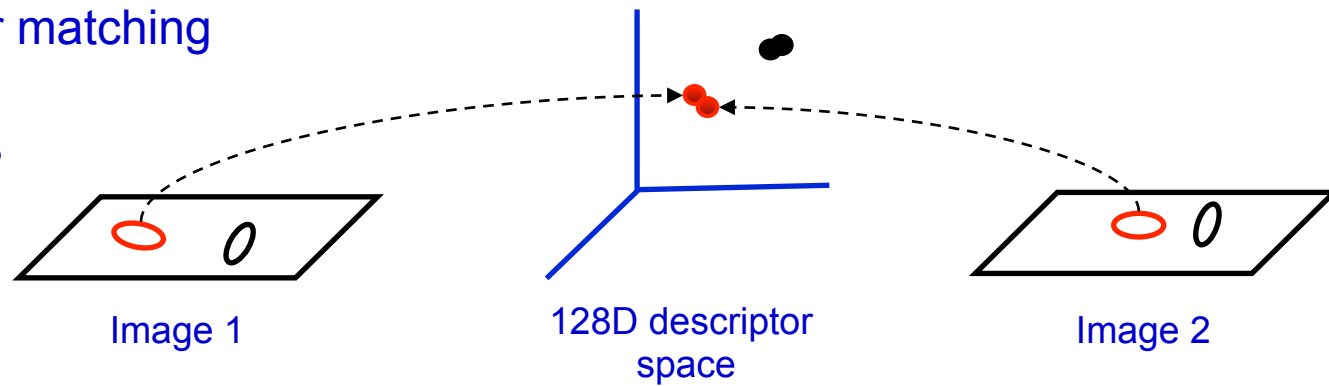


Visual words: quantize descriptor space

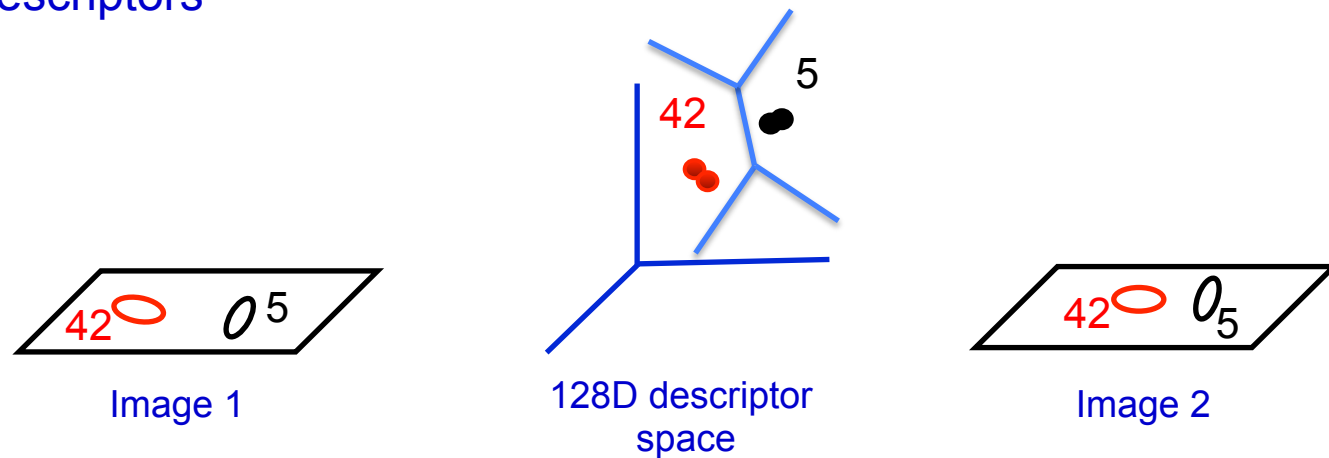
Sivic and Zisserman, ICCV 2003

Nearest neighbour matching

- expensive to do for all frames



Vector quantize descriptors

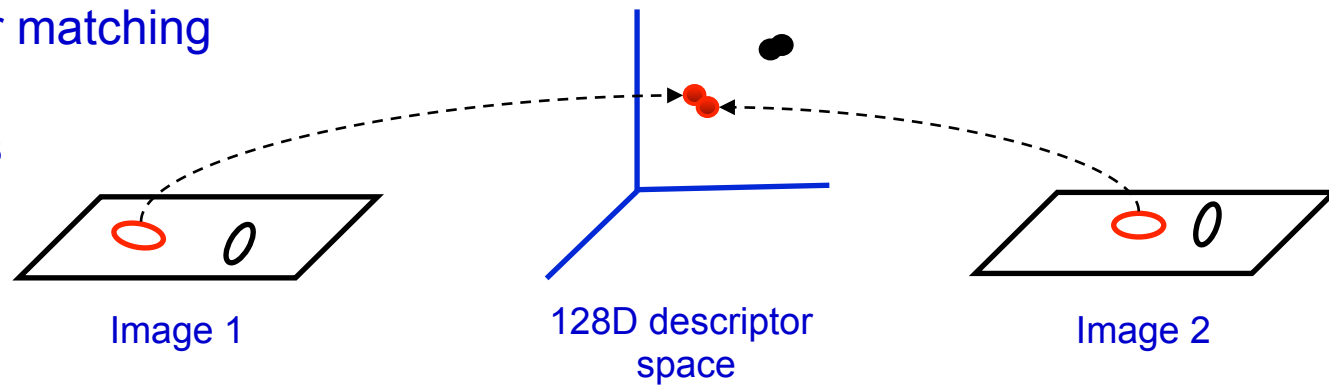


Visual words: quantize descriptor space

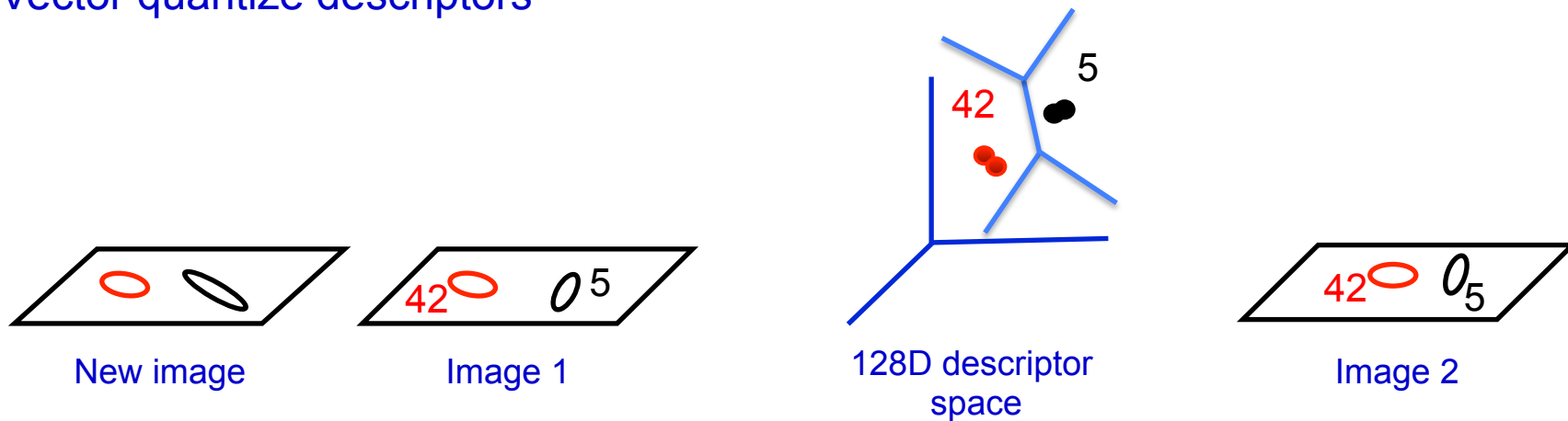
Sivic and Zisserman, ICCV 2003

Nearest neighbour matching

- expensive to do for all frames



Vector quantize descriptors

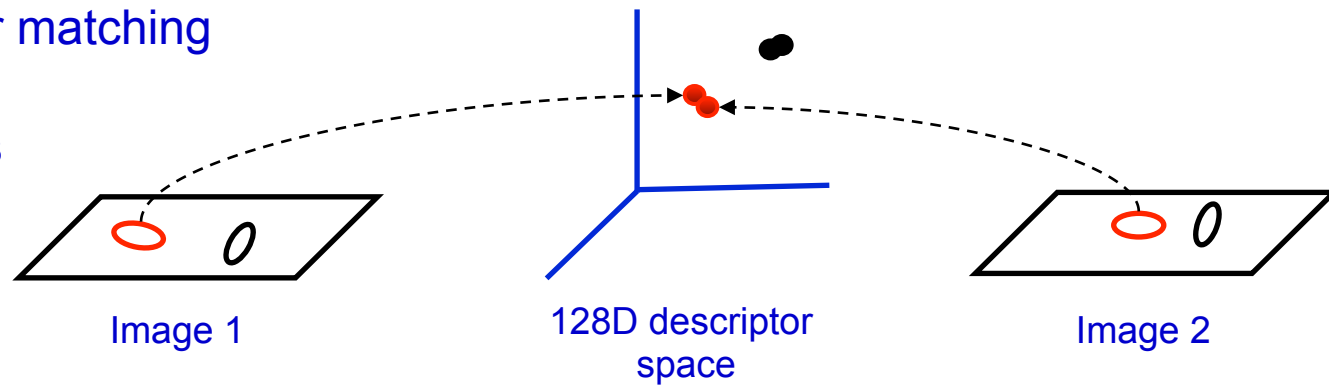


Visual words: quantize descriptor space

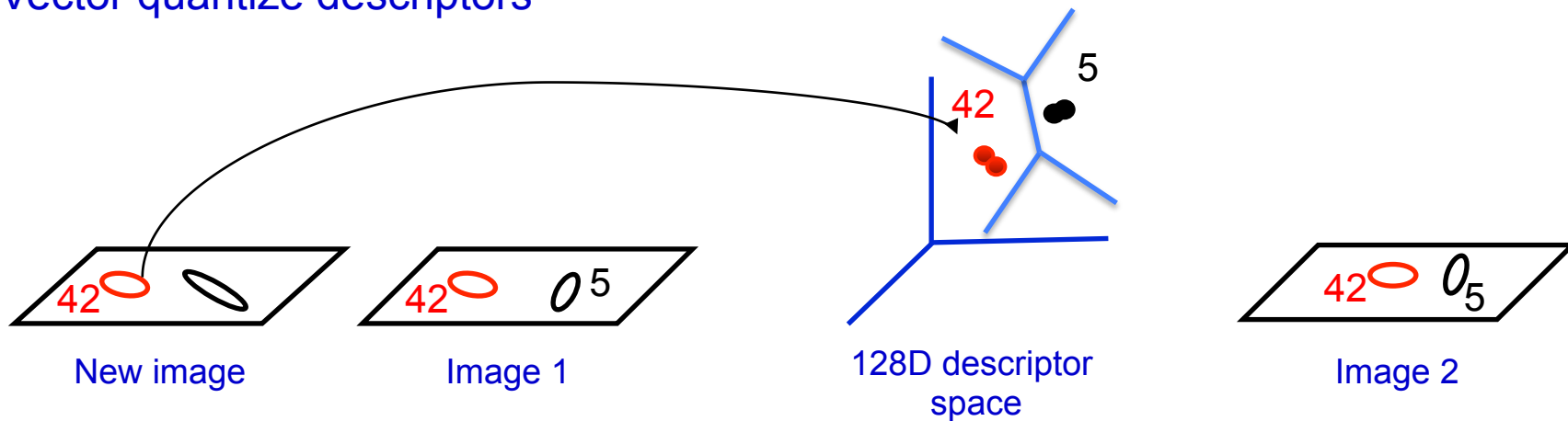
Sivic and Zisserman, ICCV 2003

Nearest neighbour matching

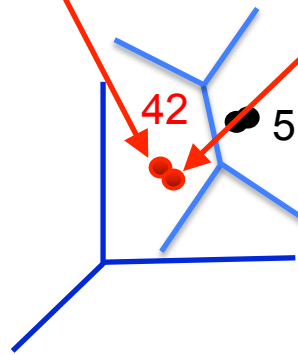
- expensive to do for all frames



Vector quantize descriptors



Vector quantize the descriptor space (SIFT)

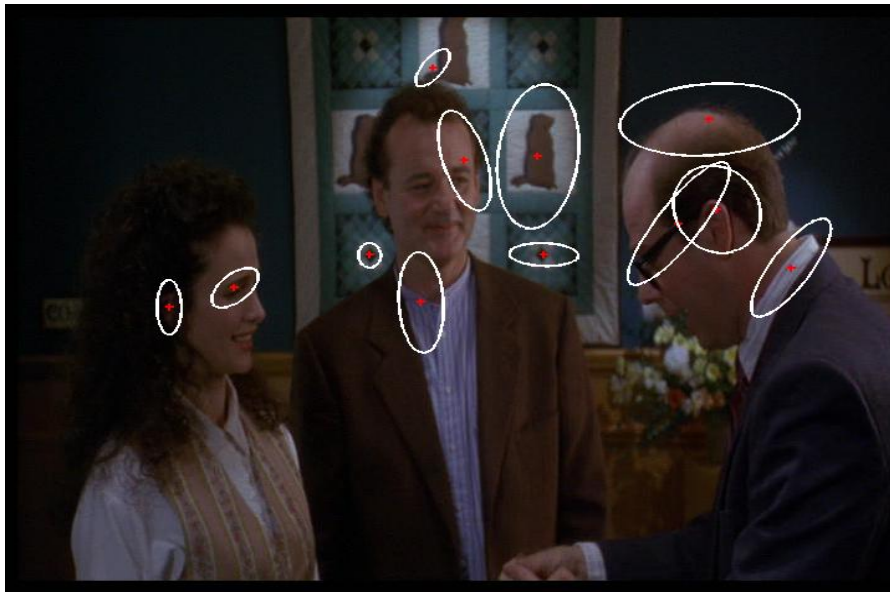


The same visual word

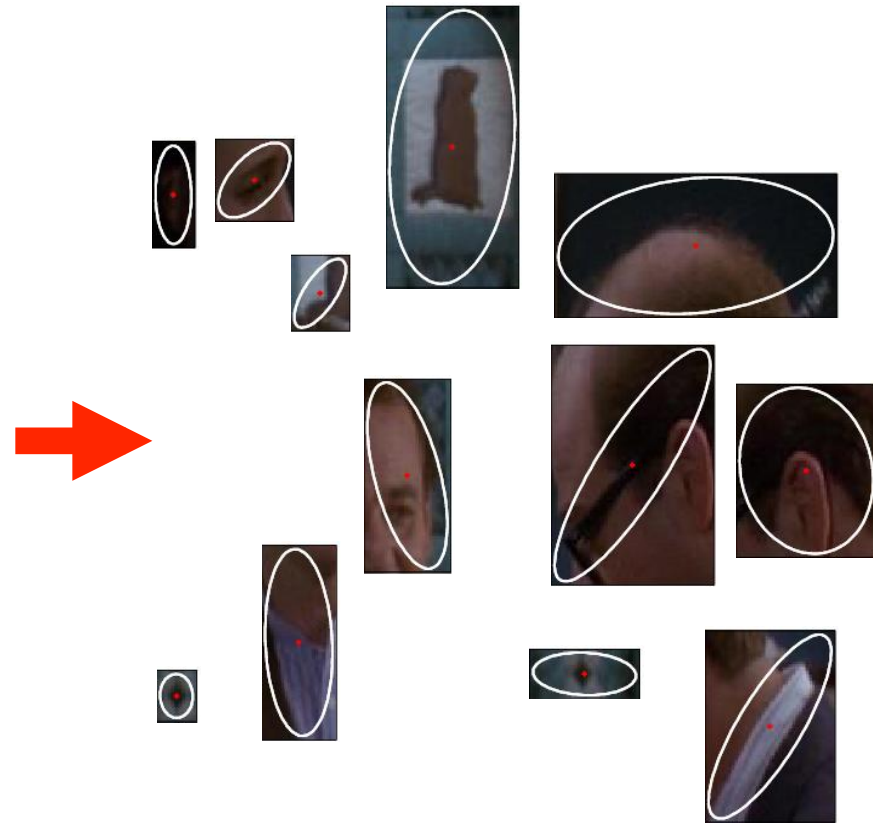
Representation: bag of (visual) words

Visual words are 'iconic' image patches or fragments

- represent their frequency of occurrence
- but not their position

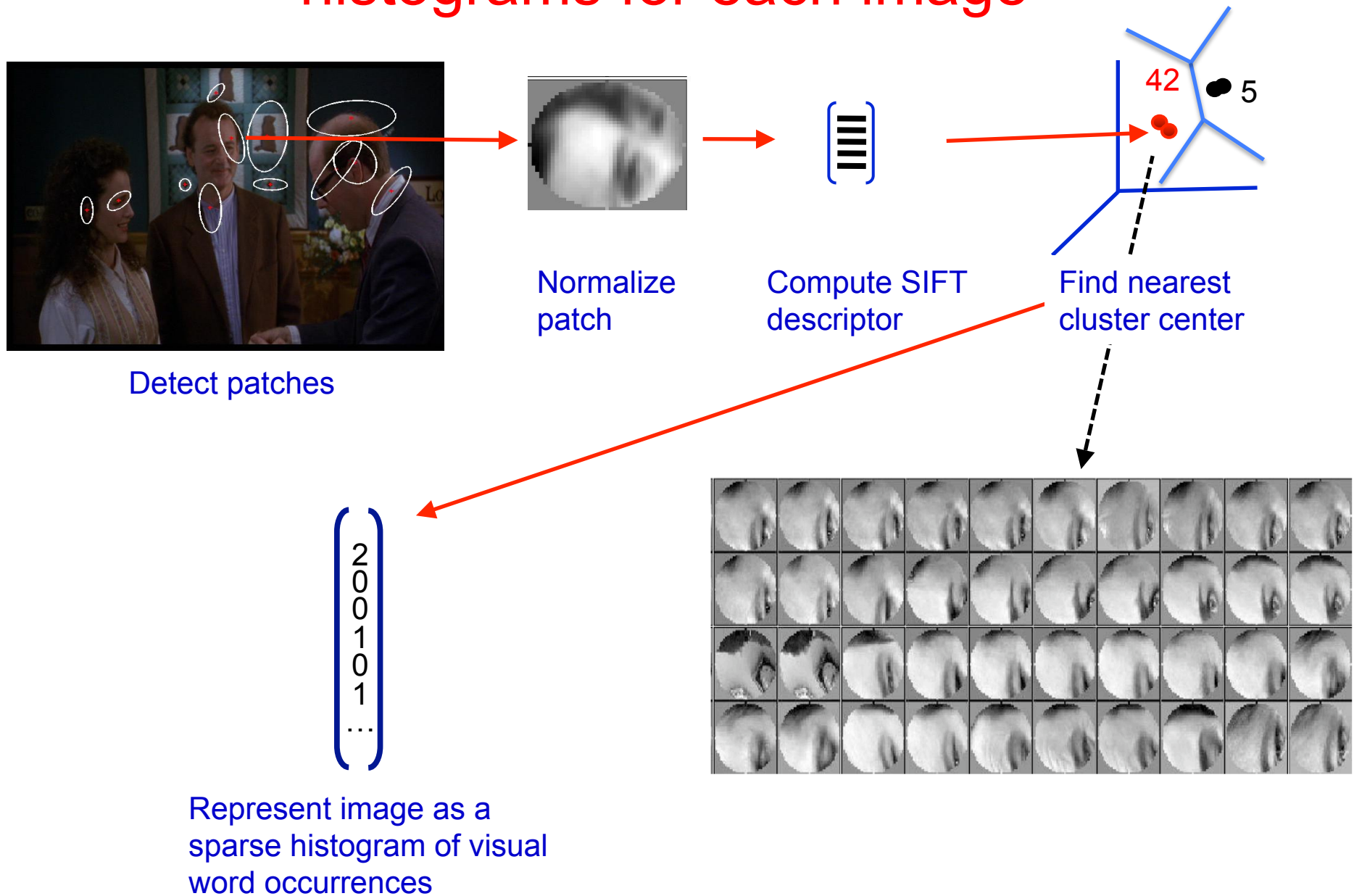


Image



Collection of visual words

Offline: Assign visual words and compute histograms for each image



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 that 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 ($\ll N$)

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

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

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

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

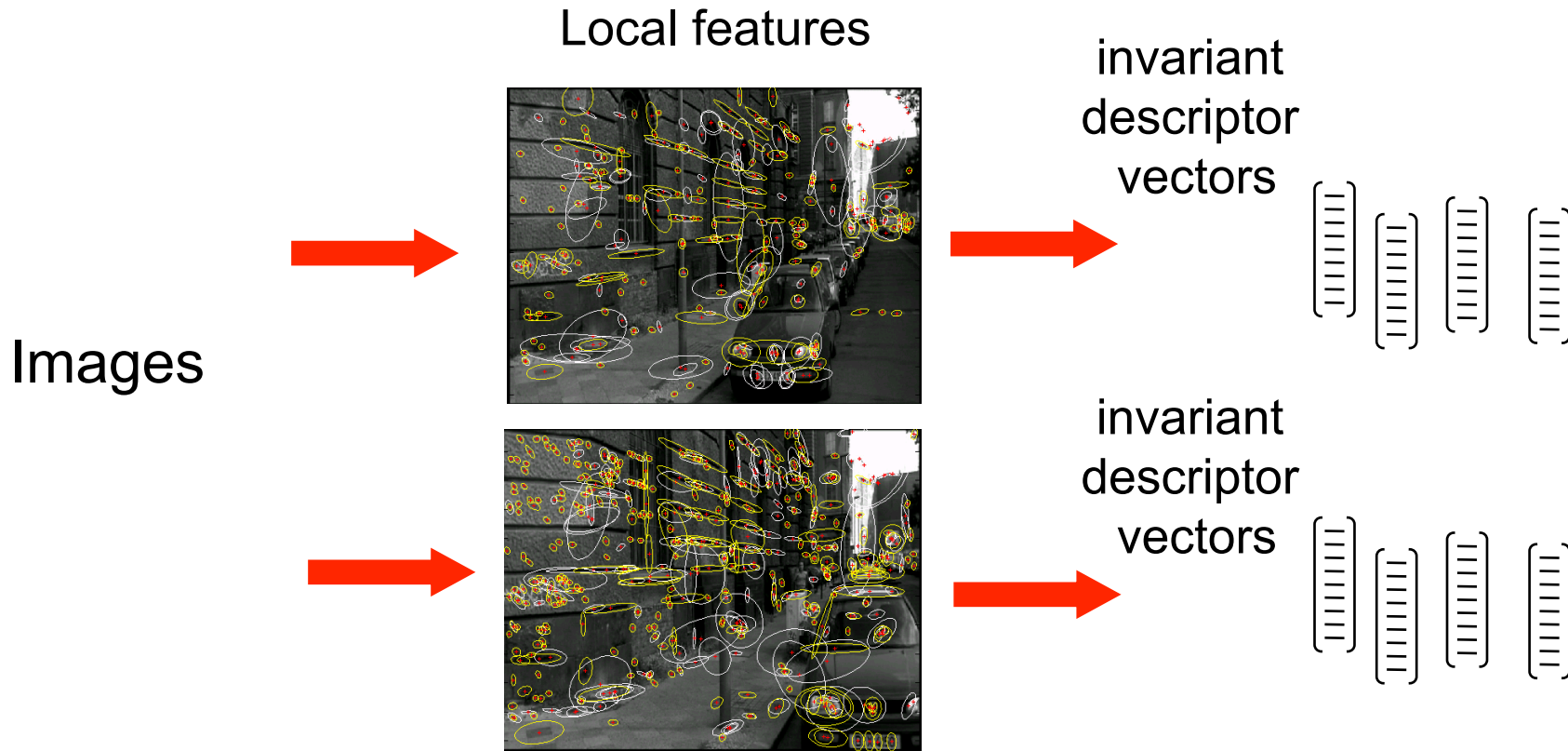
Images are ranked by the normalized scalar product between the query vector \mathbf{v}_q and all vectors in the database \mathbf{v}_d :

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

Scalar product can be computed efficiently using inverted file.

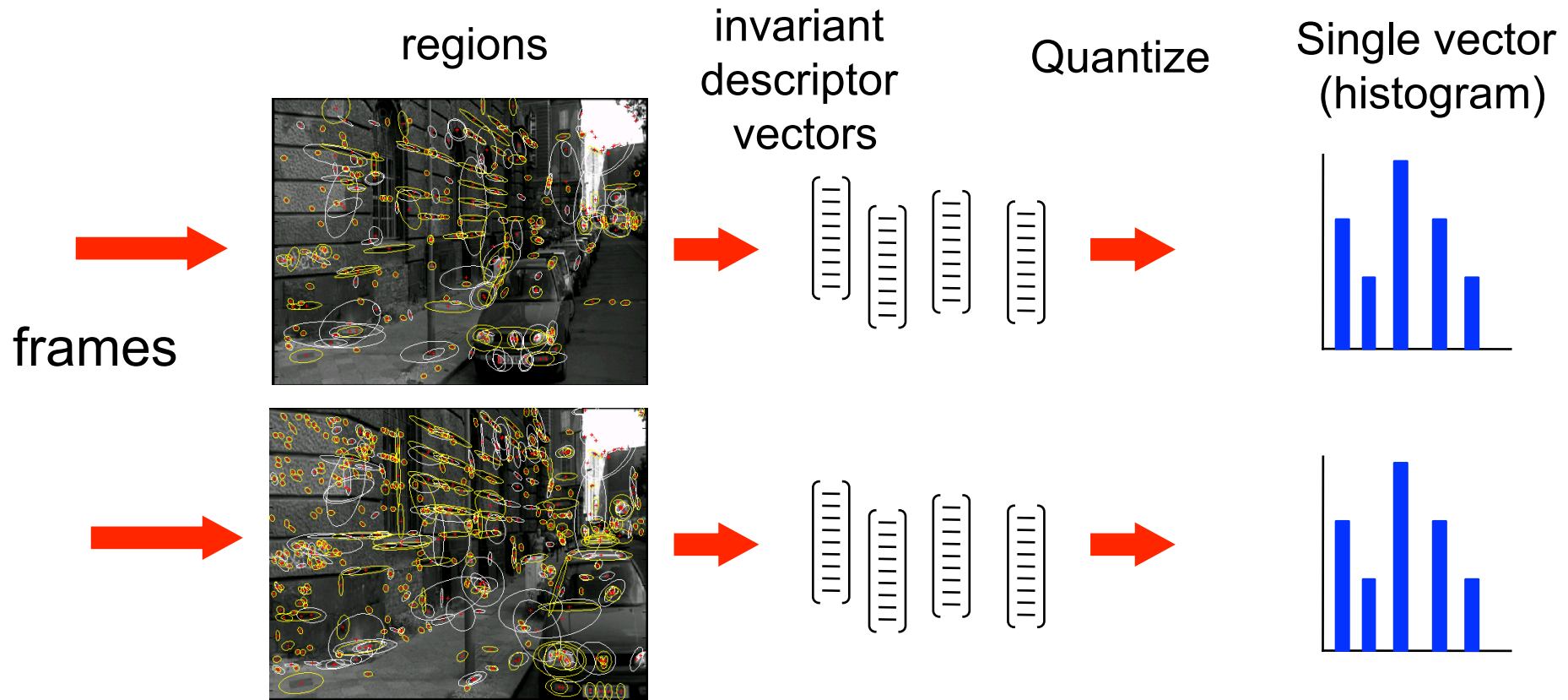
What if vectors are binary? What is the meaning of $\mathbf{v}_q^\top \mathbf{v}_d$?

Strategy I: Efficient approximate NN search



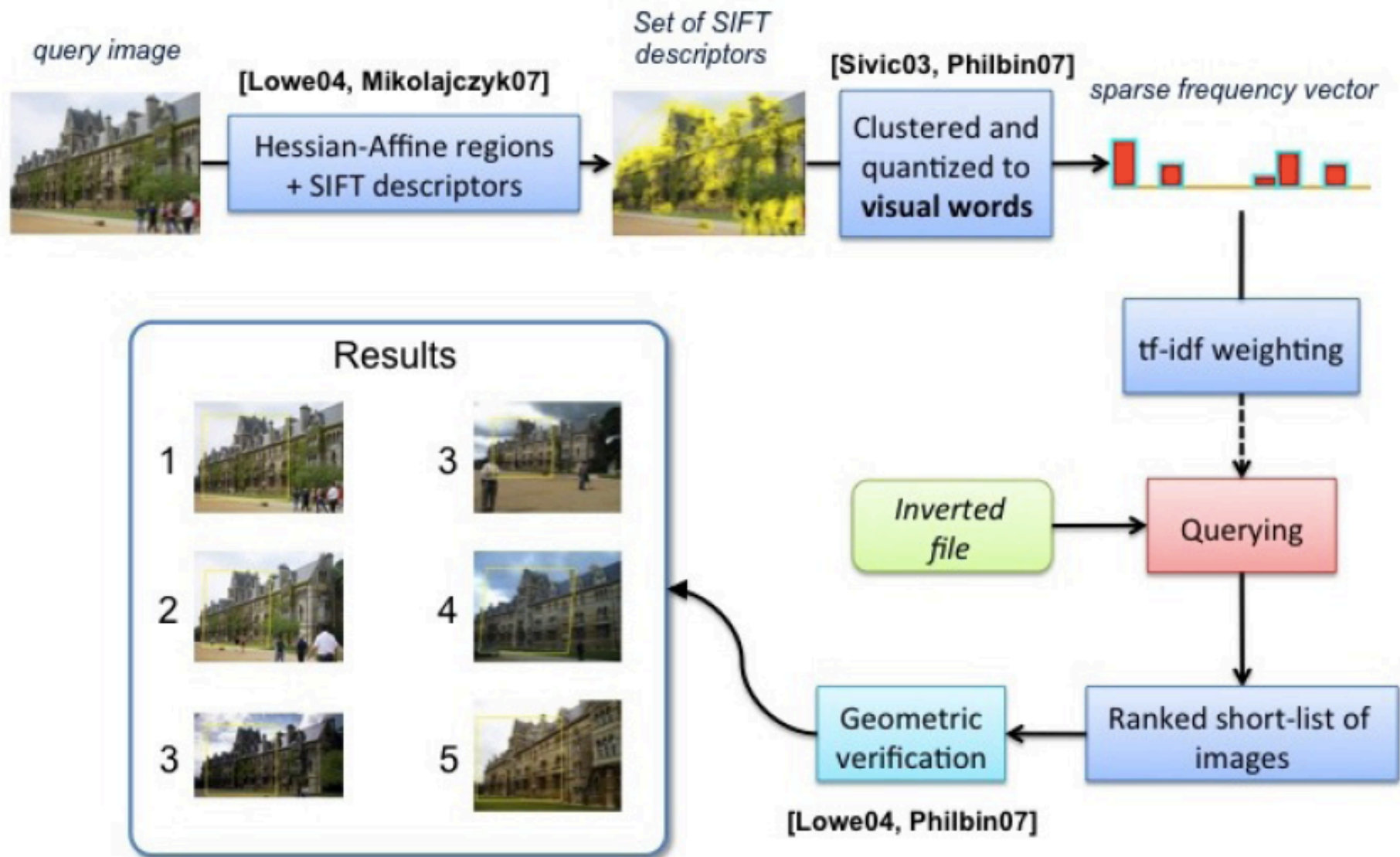
1. Compute local features in each image independently (offline)
2. “Label” each feature by a descriptor vector based on its intensity (offline)
3. Finding corresponding features is transformed to **finding nearest neighbour vectors**
4. Rank matched images by number of (tentatively) corresponding regions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

Overview of the retrieval system



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 [Muja&Lowe 2009]
- True also for building the vocabulary
 - approximate k-means [Philbin et al. 2007]
 - Reduce k-means cost from $O(NK)$ to $O(N \log K)$
 - Can scale to very large K .

Visual words: discussion II.

- 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](http://www.robots.ox.ac.uk/~vgg/research/oxbuildings/index.html)

Oxford buildings dataset

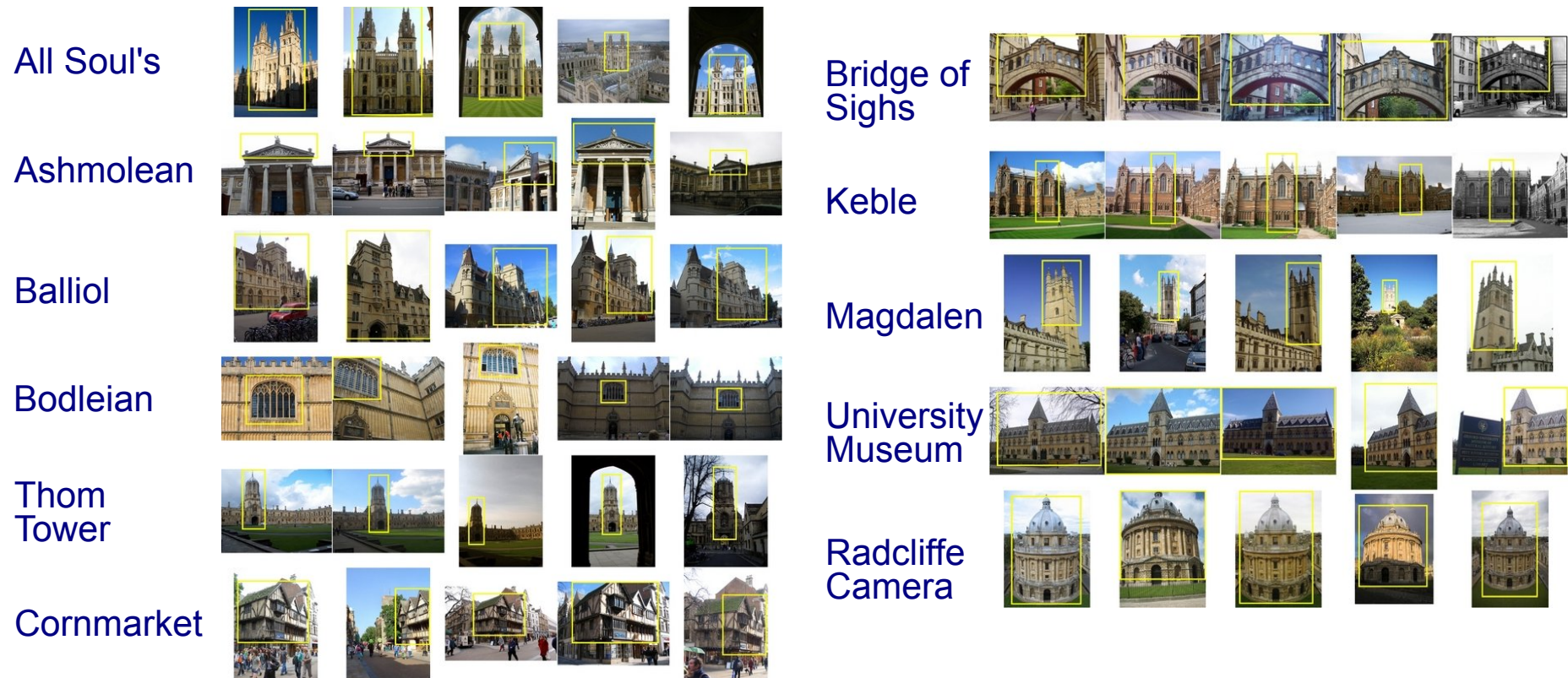
- Automatically crawled from **flickr**
- Consists of:

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

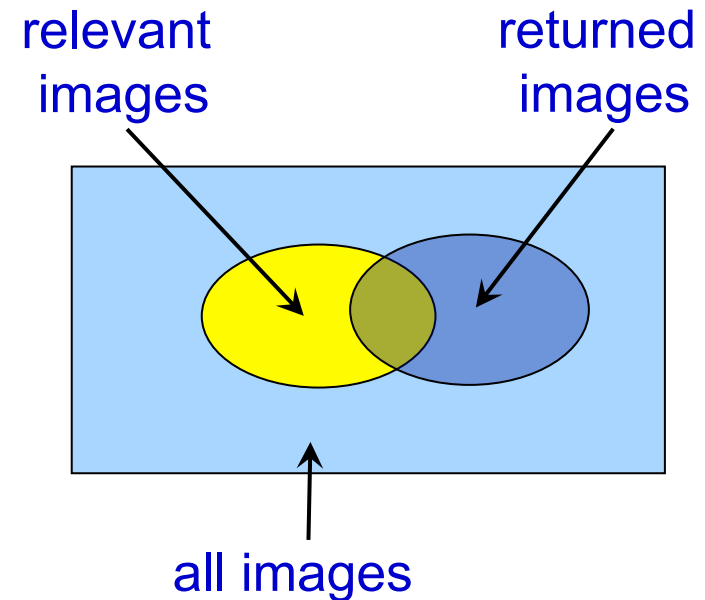
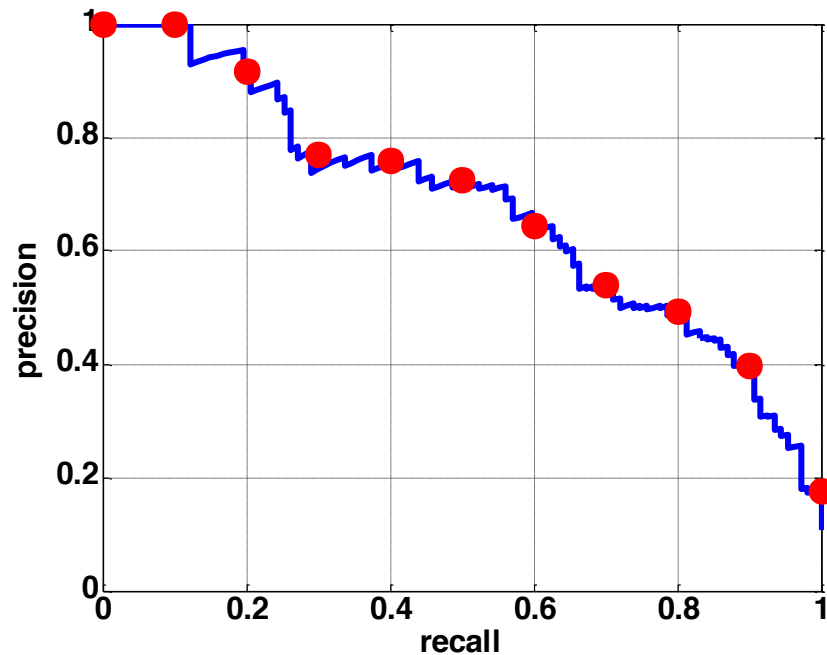
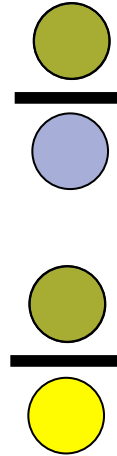
- Landmarks plus queries used for evaluation



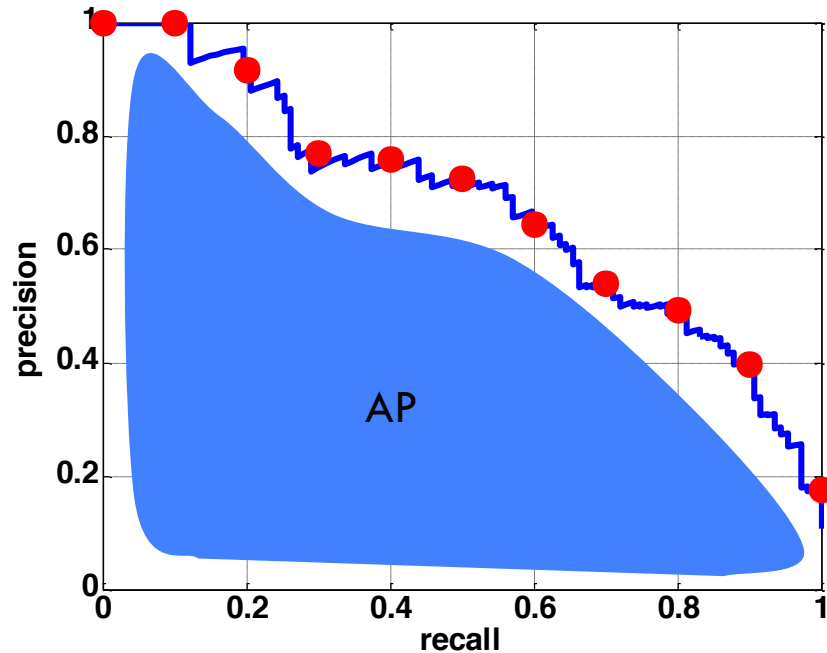
- Ground truth obtained for 11 landmarks
- Evaluate performance by mean Average Precision

Measuring retrieval performance: Precision - Recall

- **Precision:** % of returned images that are relevant
- **Recall:** % of relevant images that are returned

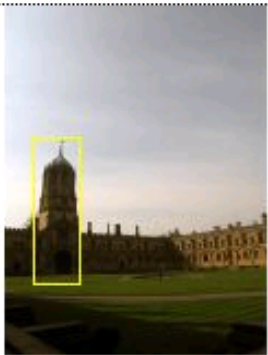


Average Precision

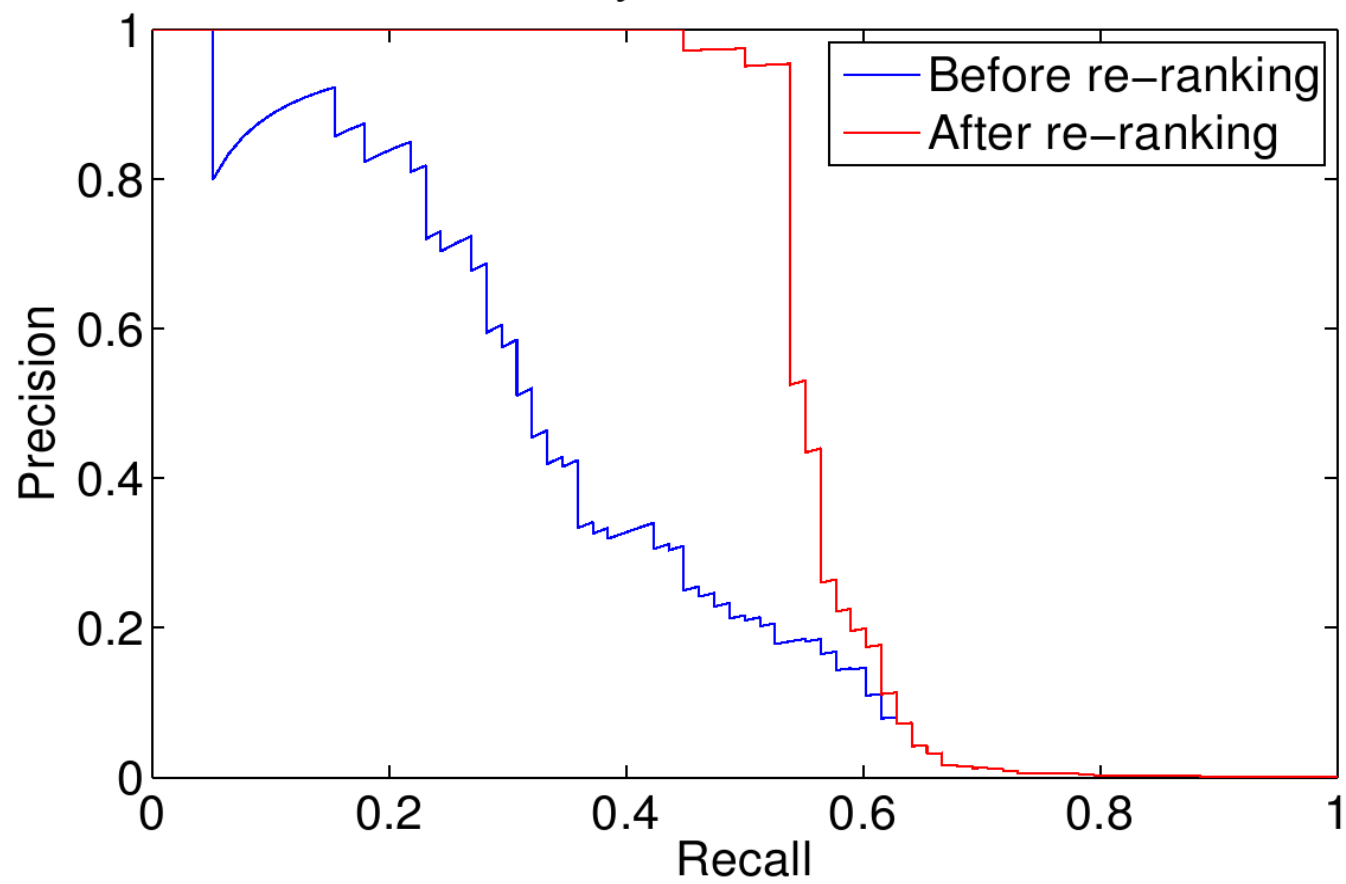


- A good AP score requires both high recall **and** high precision
- Application-independent

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

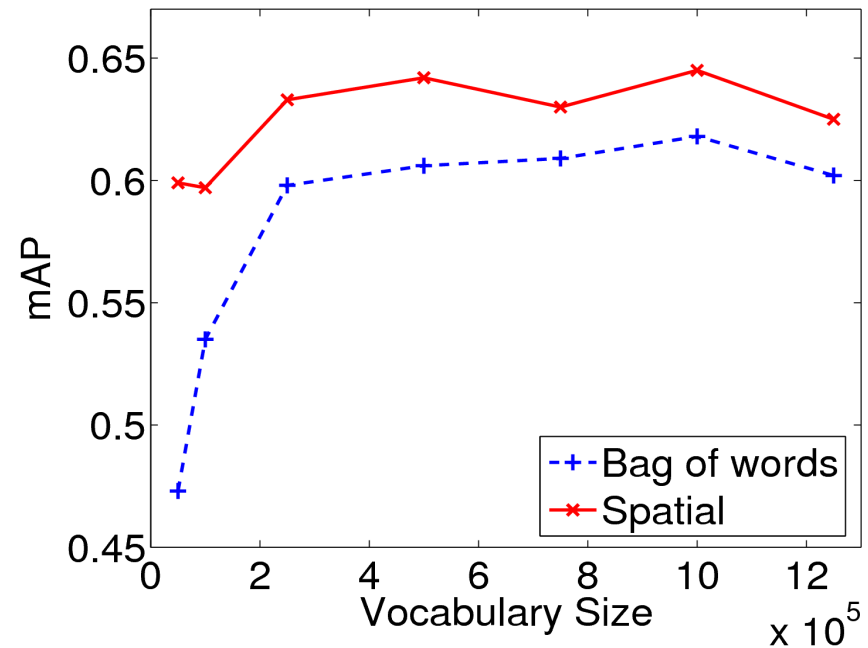


Query: ChristChurch3

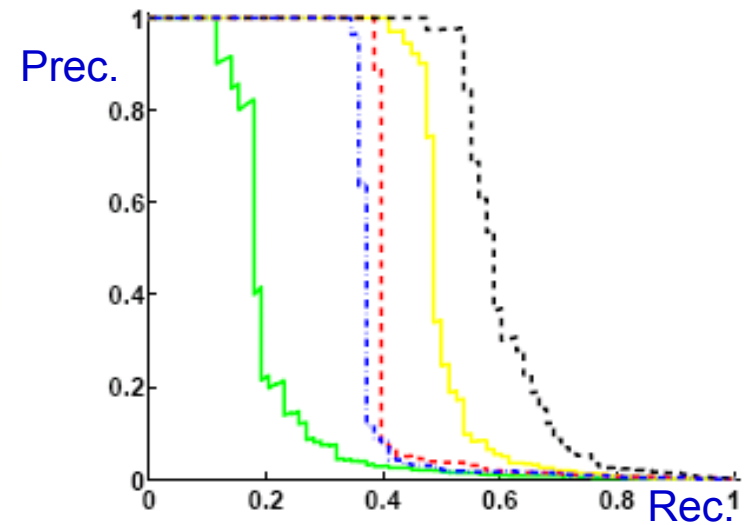
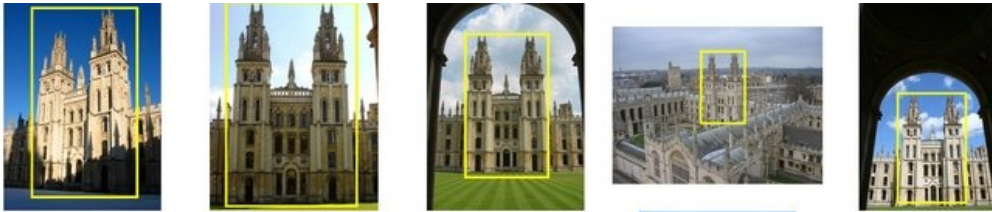


Mean Average Precision variation with vocabulary size

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

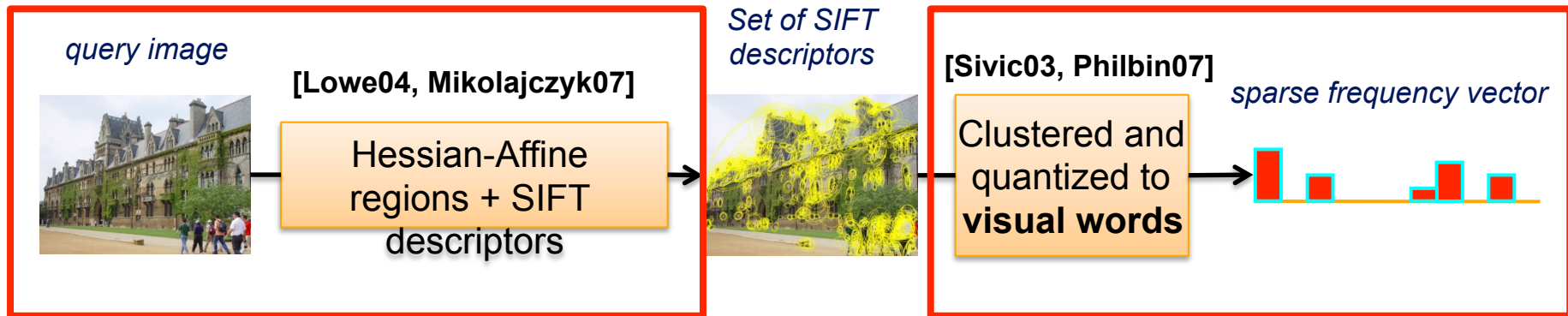


Query images



- 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

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

1. Original query



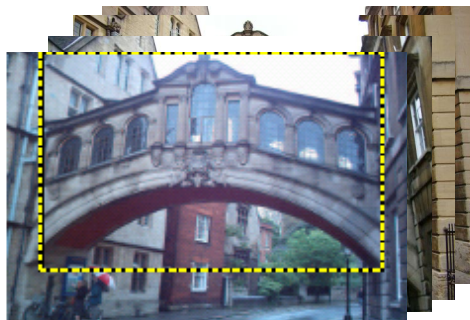
2. Initial retrieval set



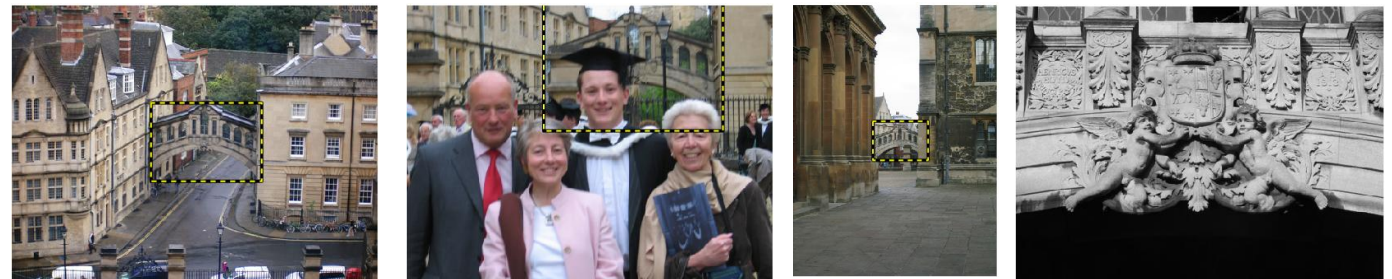
3. Spatial verification



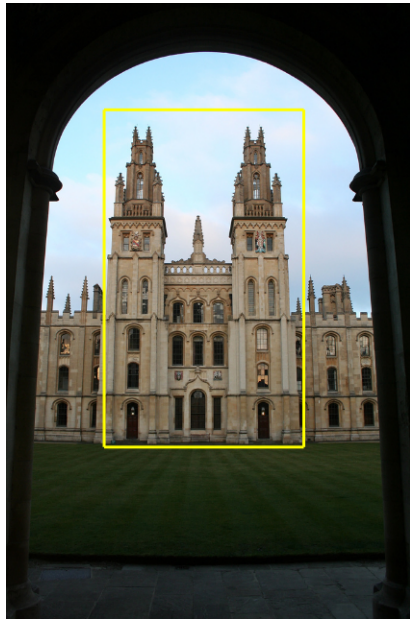
4. New enhanced query



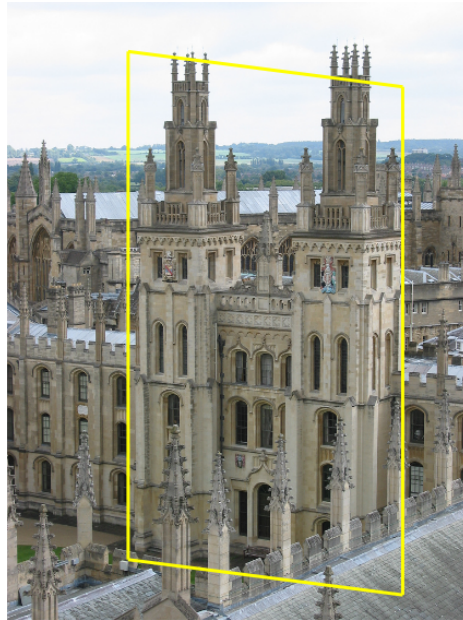
5. Additional retrieved images



Query Expansion



Query Image

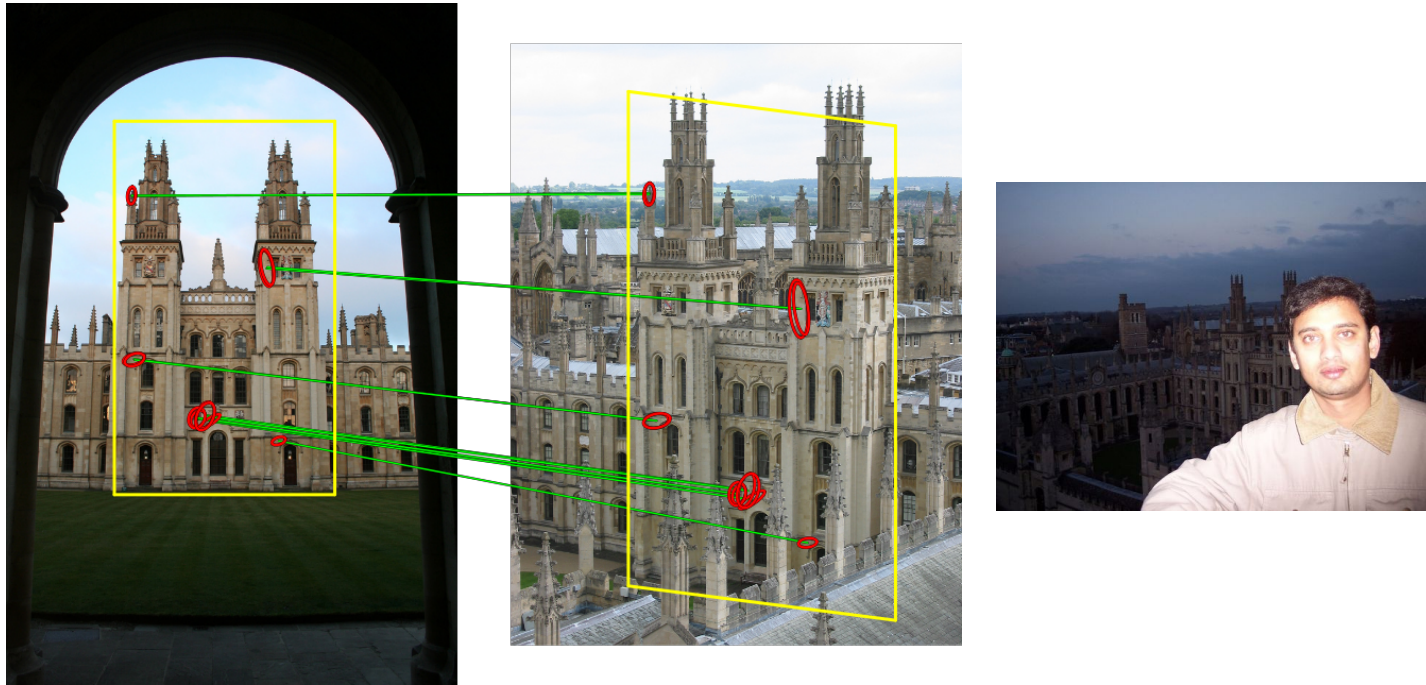


Originally retrieved image

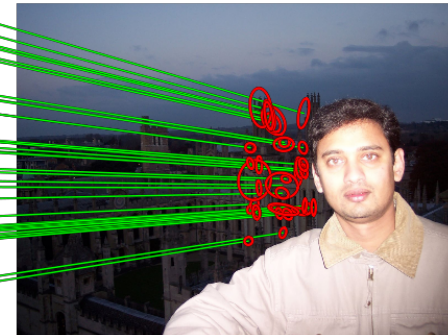
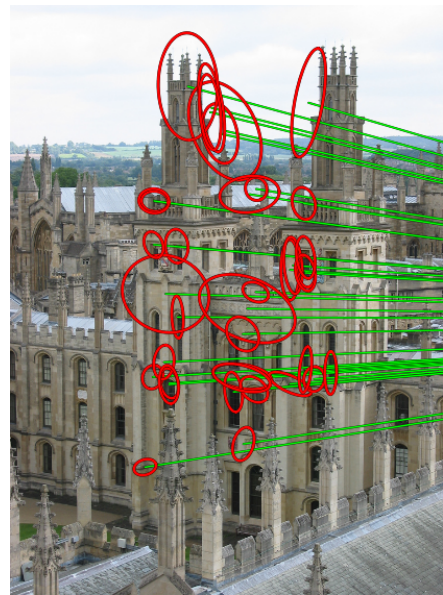
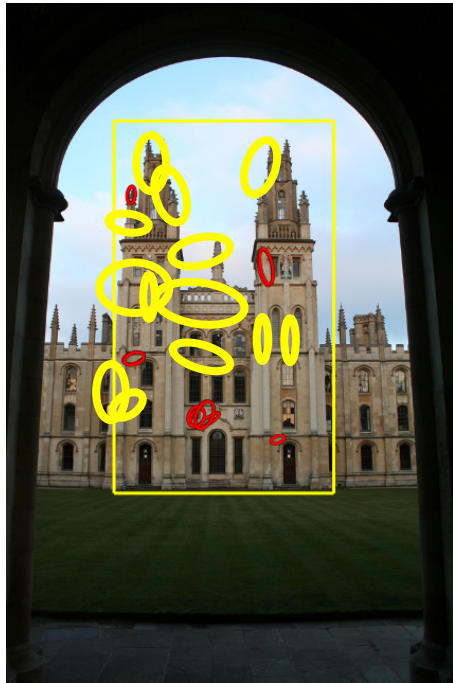


Originally not retrieved

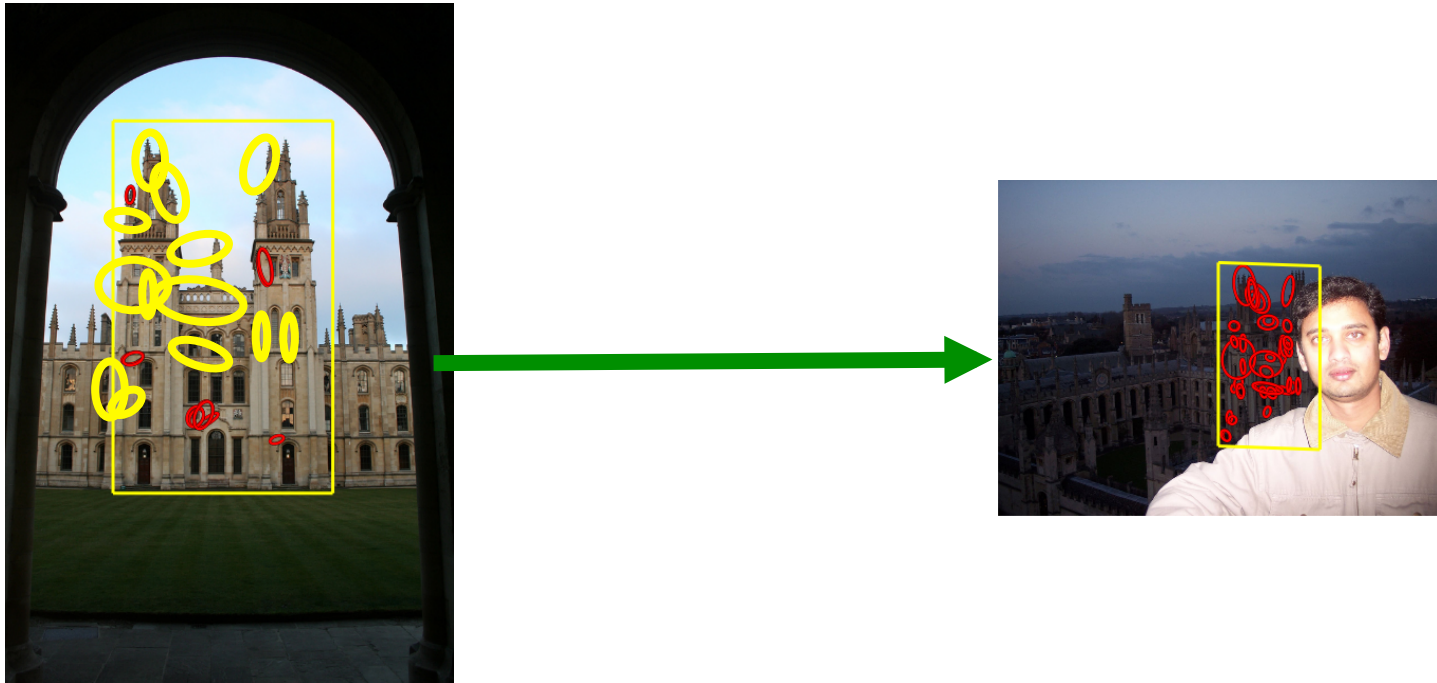
Query Expansion



Query Expansion



Query Expansion



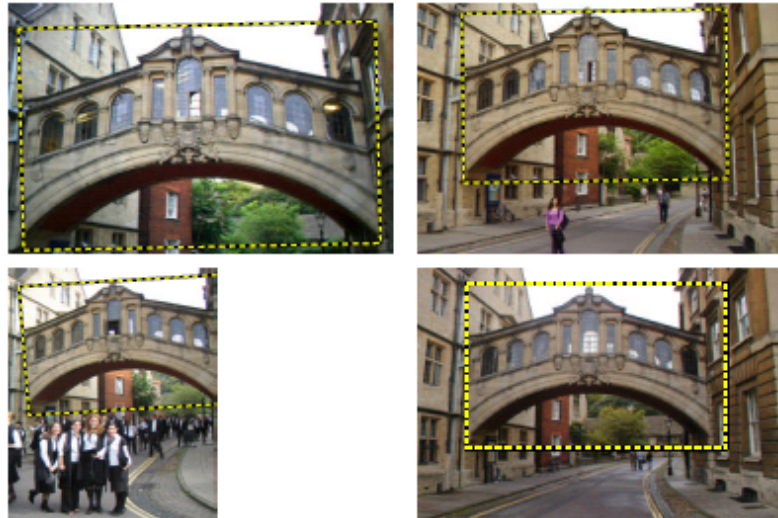
Demo

Query Expansion

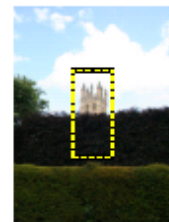
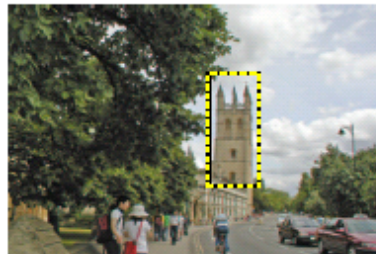
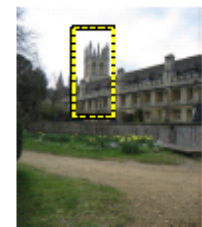
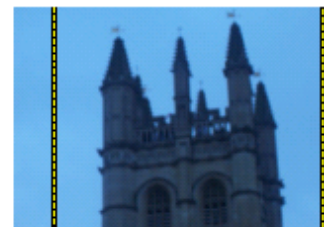
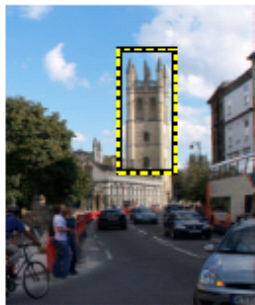
Query image

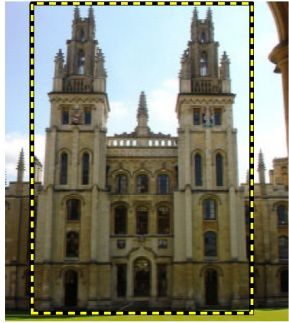


Originally retrieved



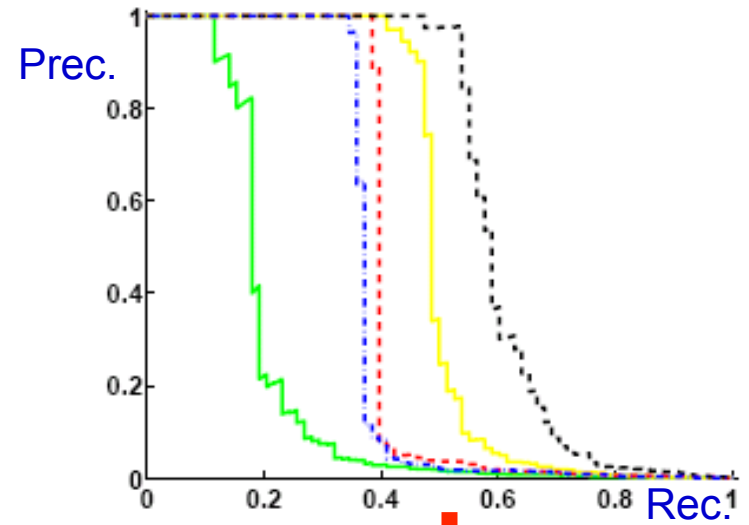
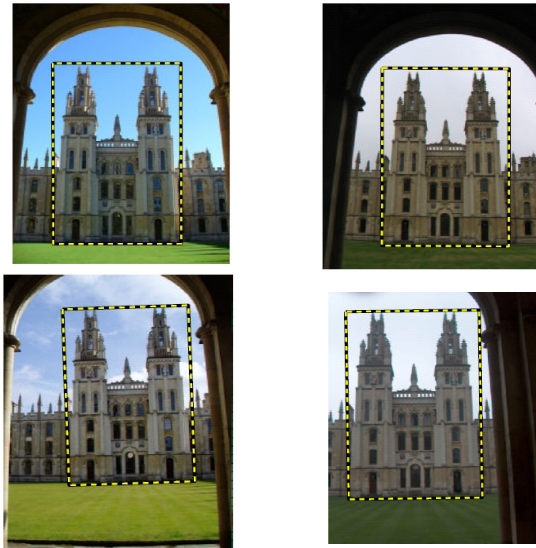
Retrieved only after expansion



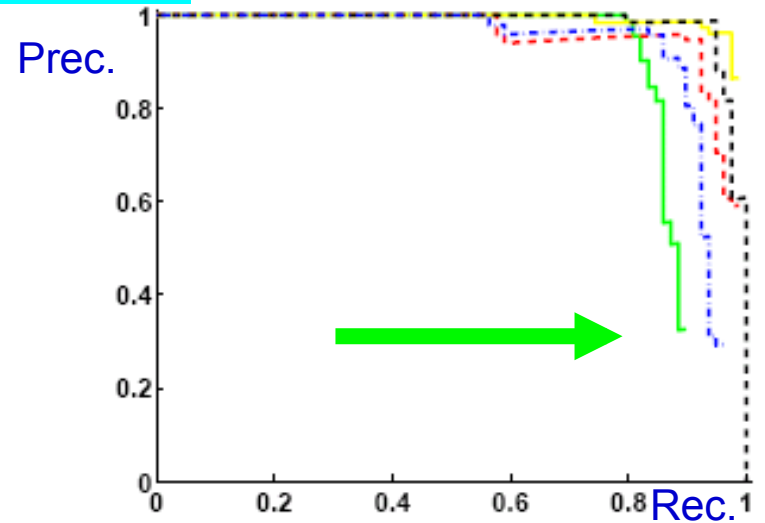


Query image

Original results (good)



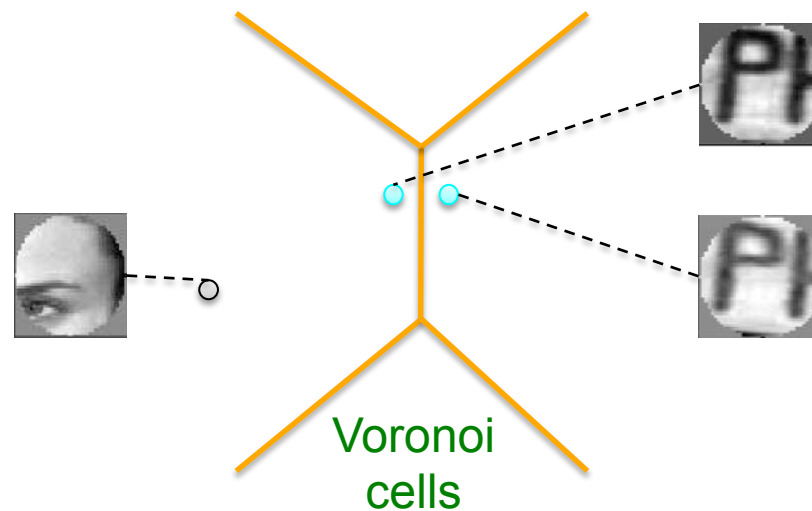
Expanded results (better)



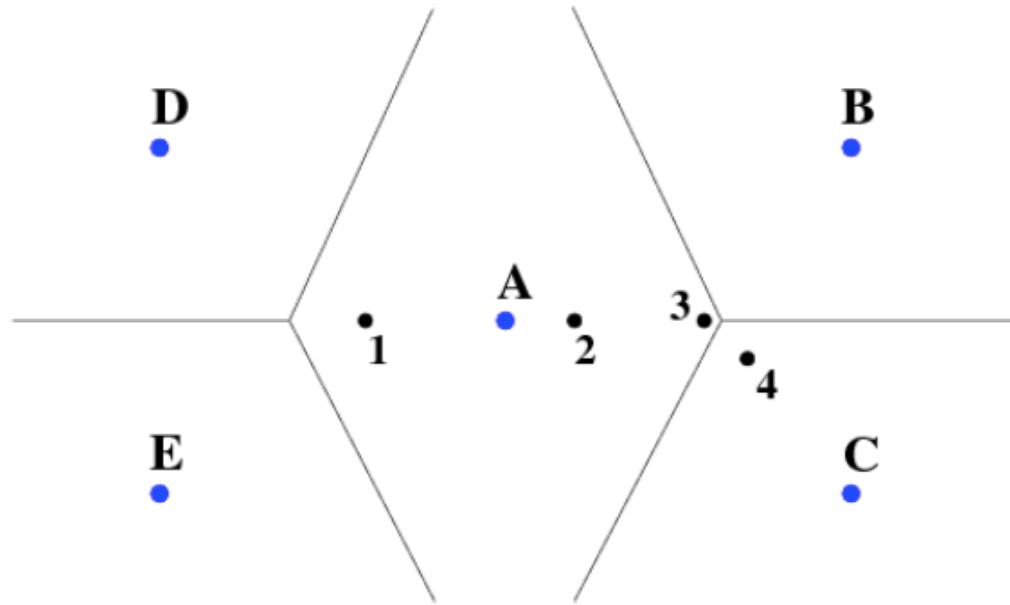
Quantization errors

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



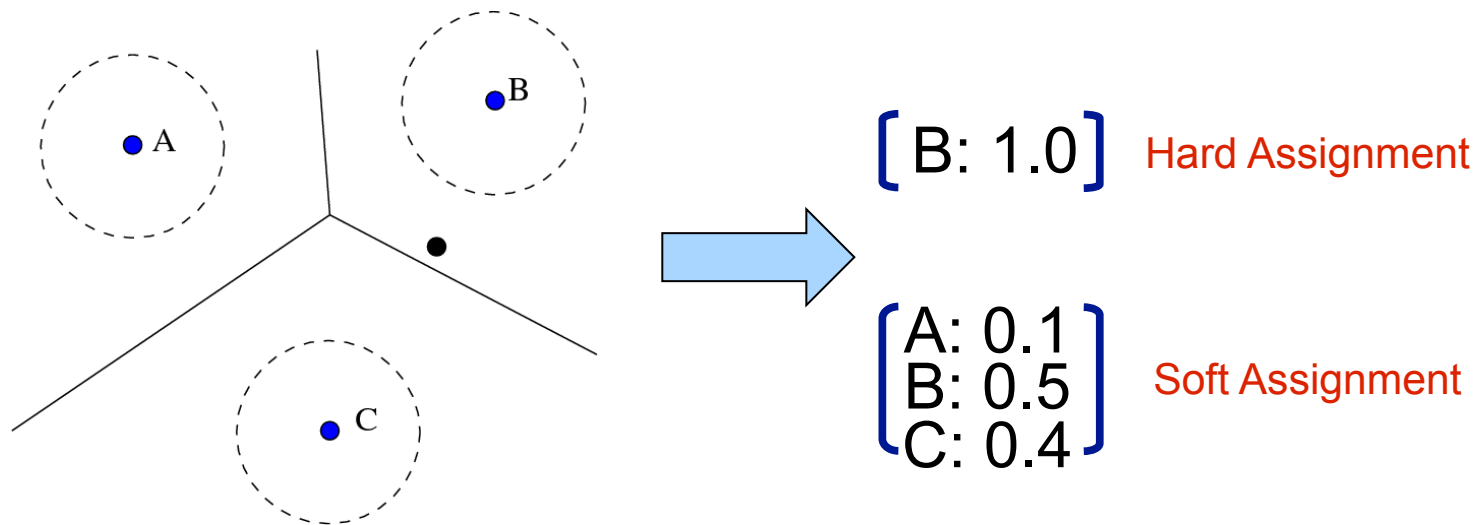
But also..



- (1) Points 3 and 4 are close but not matched.
- (2) Points 1, 2 and 3 are matched equally.

Overcoming quantization errors

- Soft-assign each descriptor to multiple cluster centers [Philbin et al. 2008, Van Gemert et al. 2008]



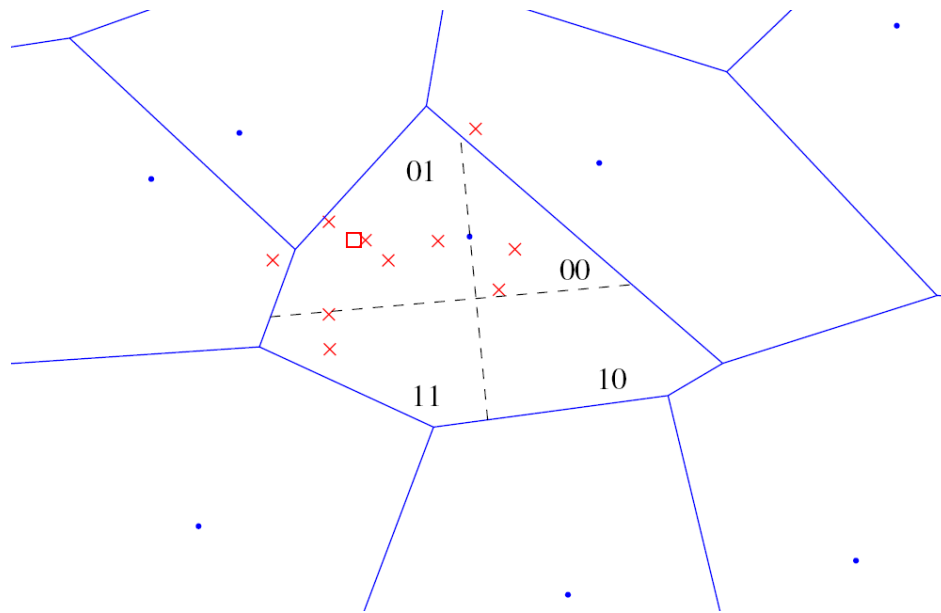
Learning a vocabulary to overcome quantization errors

[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

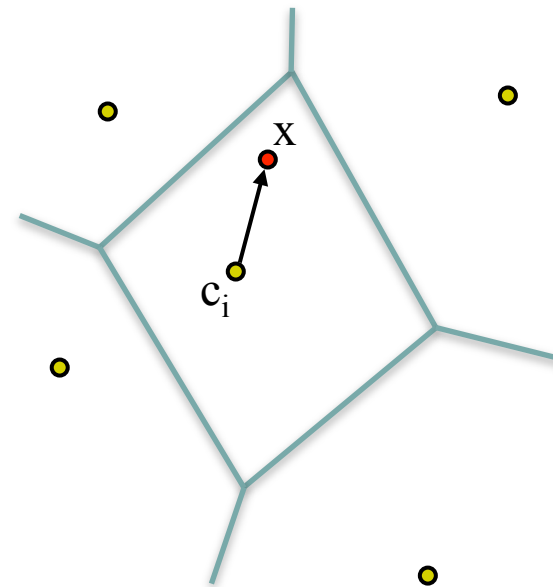


More next lecture (C. Schmid)

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)

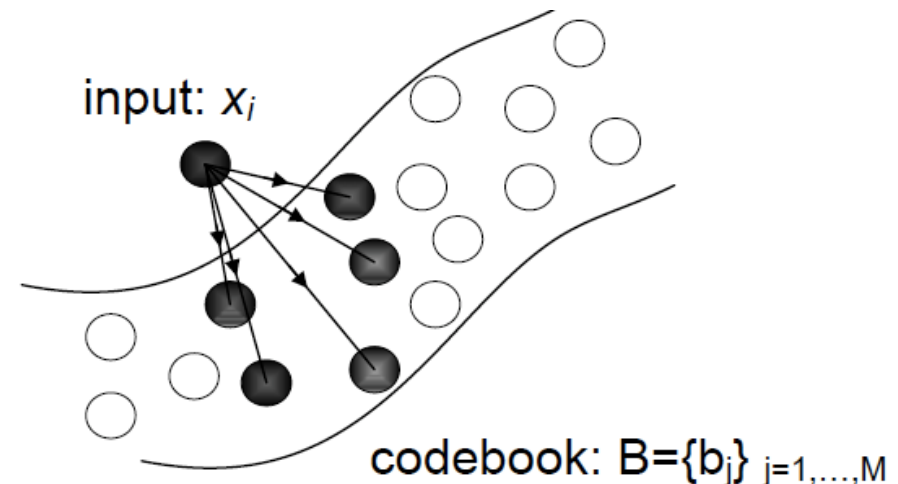
Beyond bag-of-visual-words III.

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 the lecture of 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]

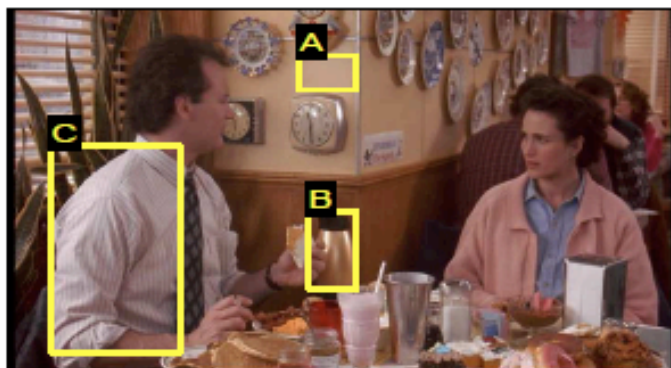
Matching in structured datasets (3D landmarks or street-view images)

[Cummins and Newman 2009, Knopp et al. ECCV 2010, Zamir&Shah ECCV 2010, Li et al. ECCV 2010, Baatz et al. ECCV 2010, Chen et al. CVPR 2011, Sattler et al. CVPR 2011]

What objects/scenes local regions do not work on?



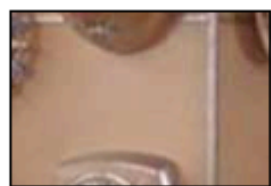
What objects/scenes local regions do not work on?



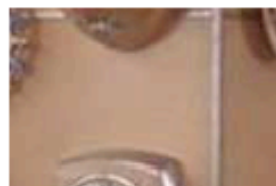
(a)



(b)



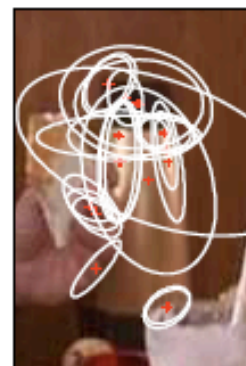
(c)



(d)



(e)



(f)



(g)



(h)

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



(e)



(f)



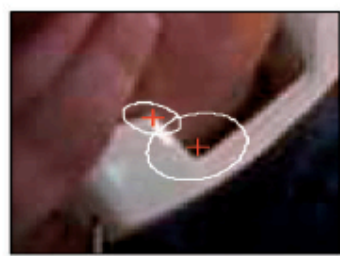
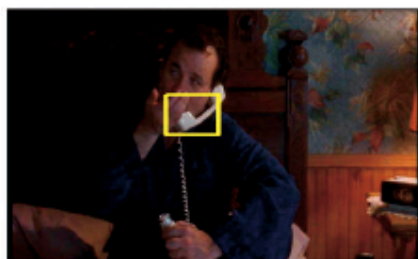
(g)



(h)



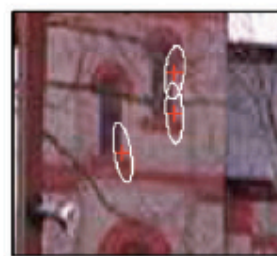
(i)



(j)



(k)



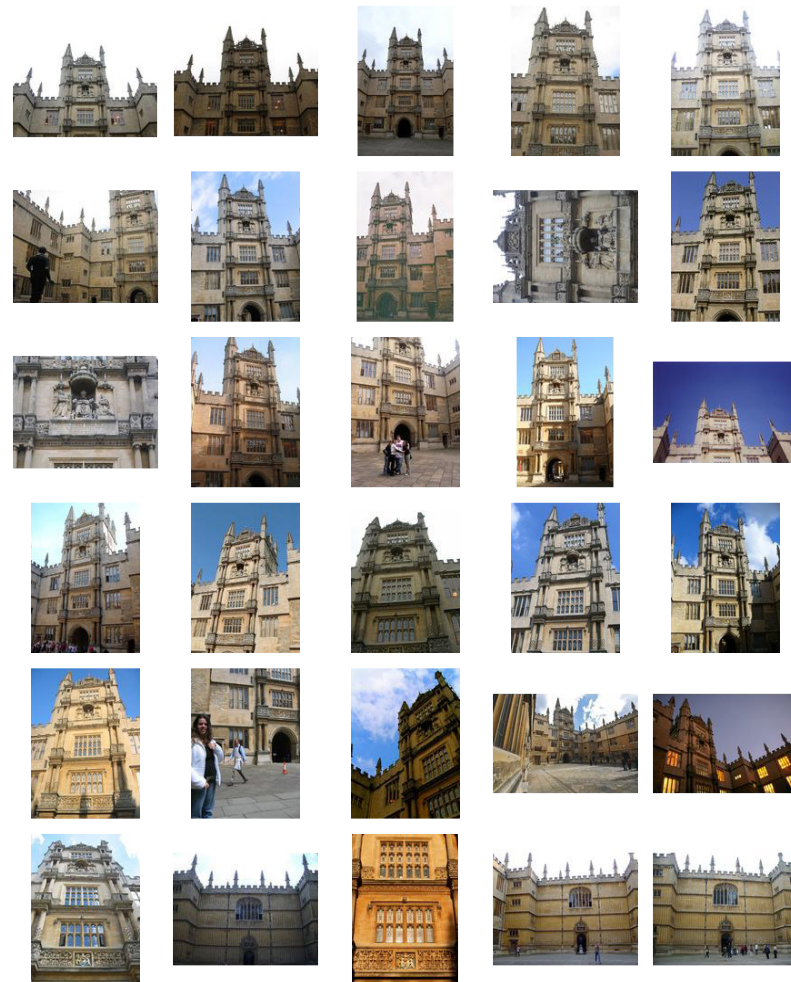
(l)

Example applications of large scale visual search and matching

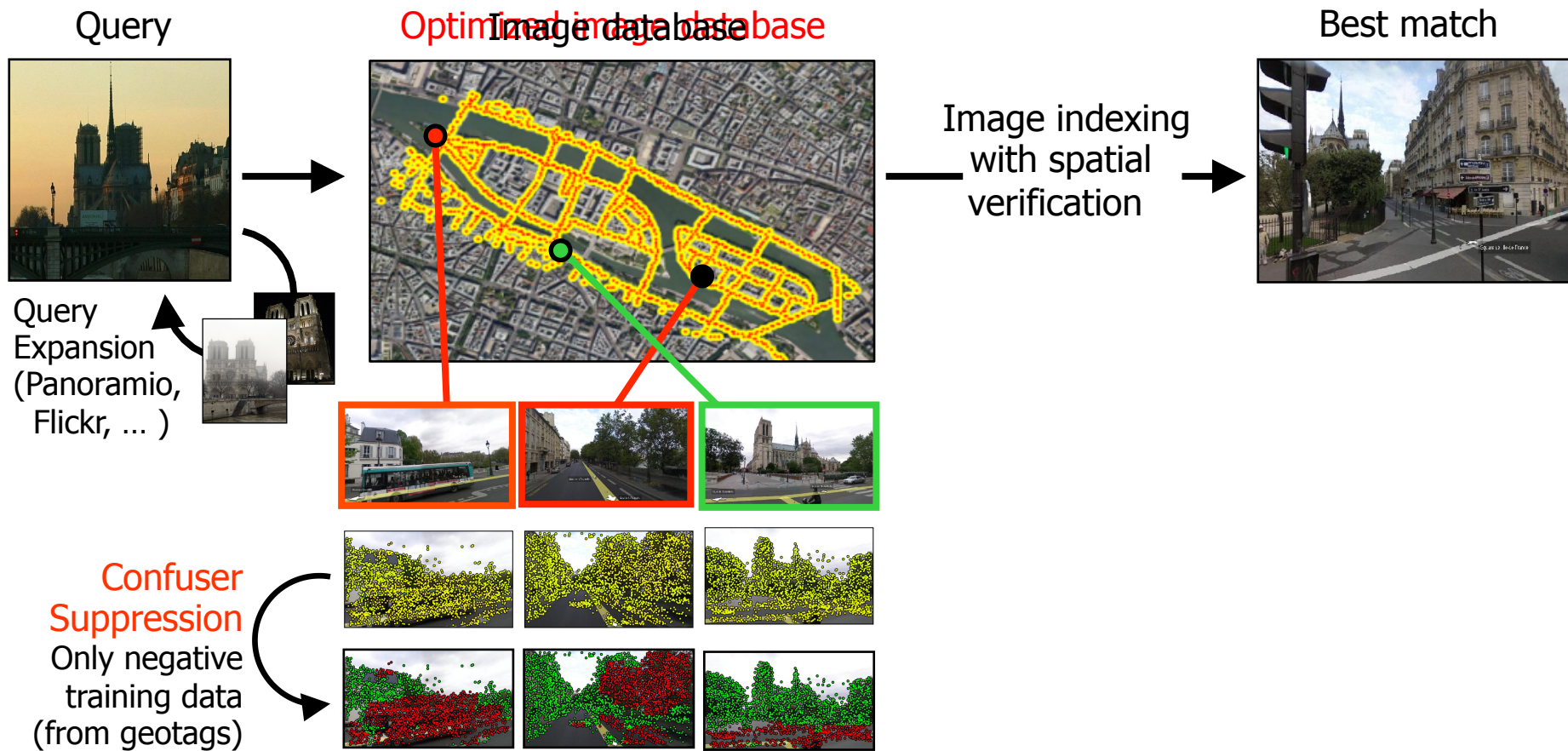
Application: Internet-based inpainting

Photo-editing using images of the same place

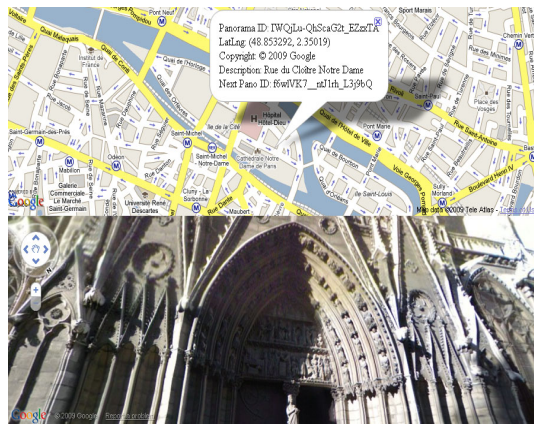
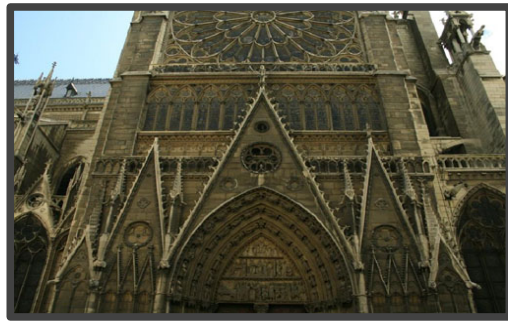
[Whyte, Sivic and Zisserman, 2009], but see also [Hays and Efros, 2007].







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

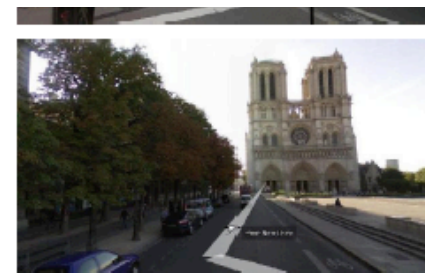
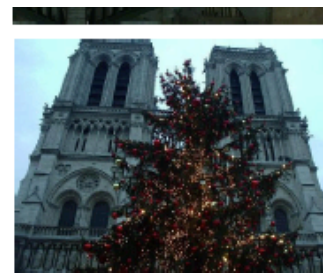
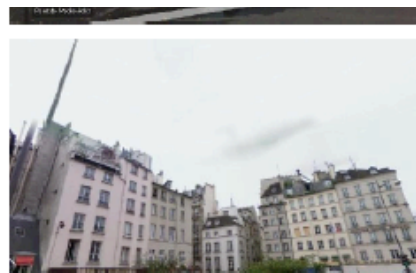
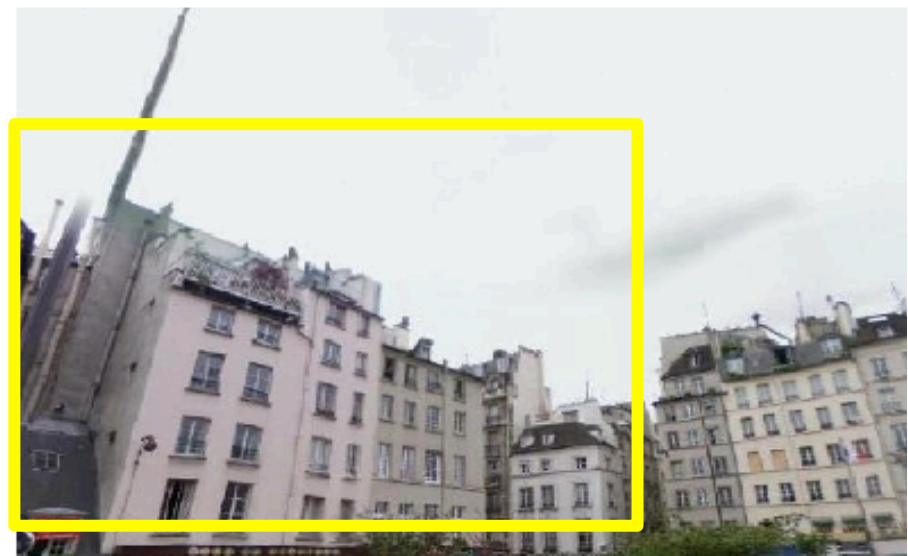


Correctly recognized examples

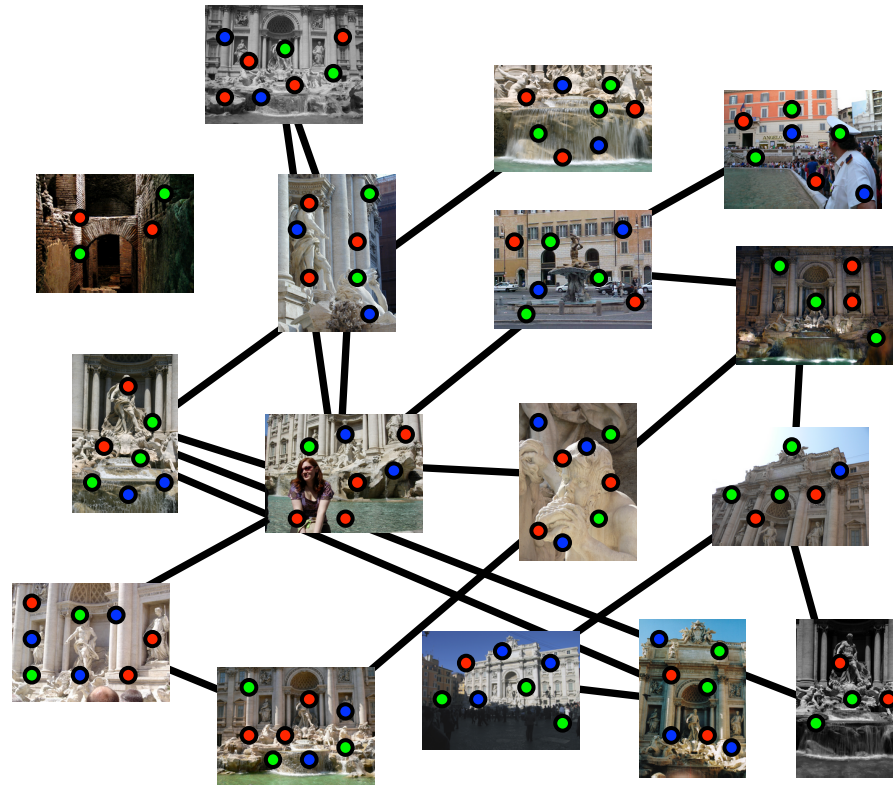


More correctly recognized examples

Query	Top ranked image	Query	Top ranked image
			



Matching and 3D reconstruction in large unstructured datasets.



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



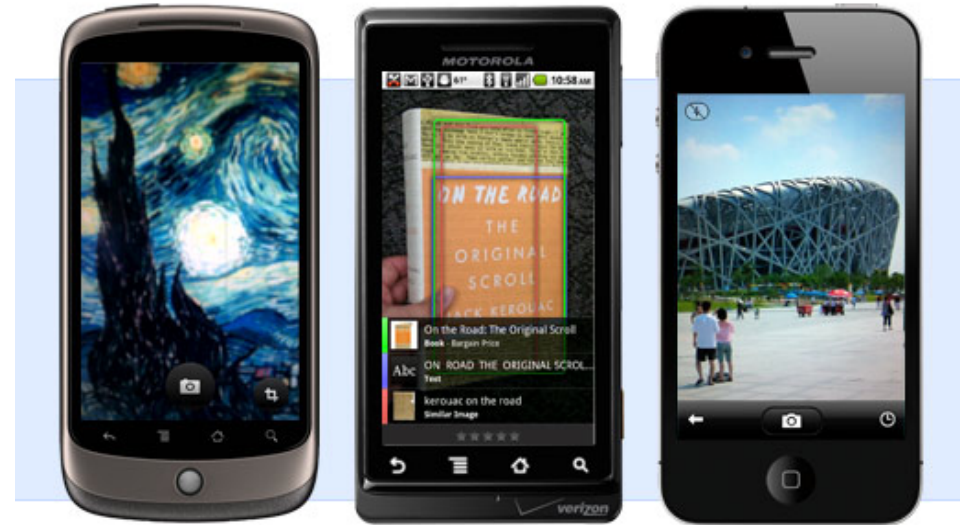
Mobile visual search

Bing visual scan



Google Goggles

Use pictures to search the web. [▶ Watch a video](#)



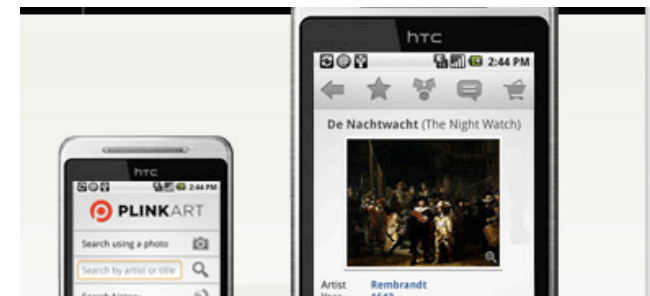
kooba
IMAGE RECOGNITION
 Paperboy
 Visual Search
For your business

Gives instant information on what you see

Visual Search

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.

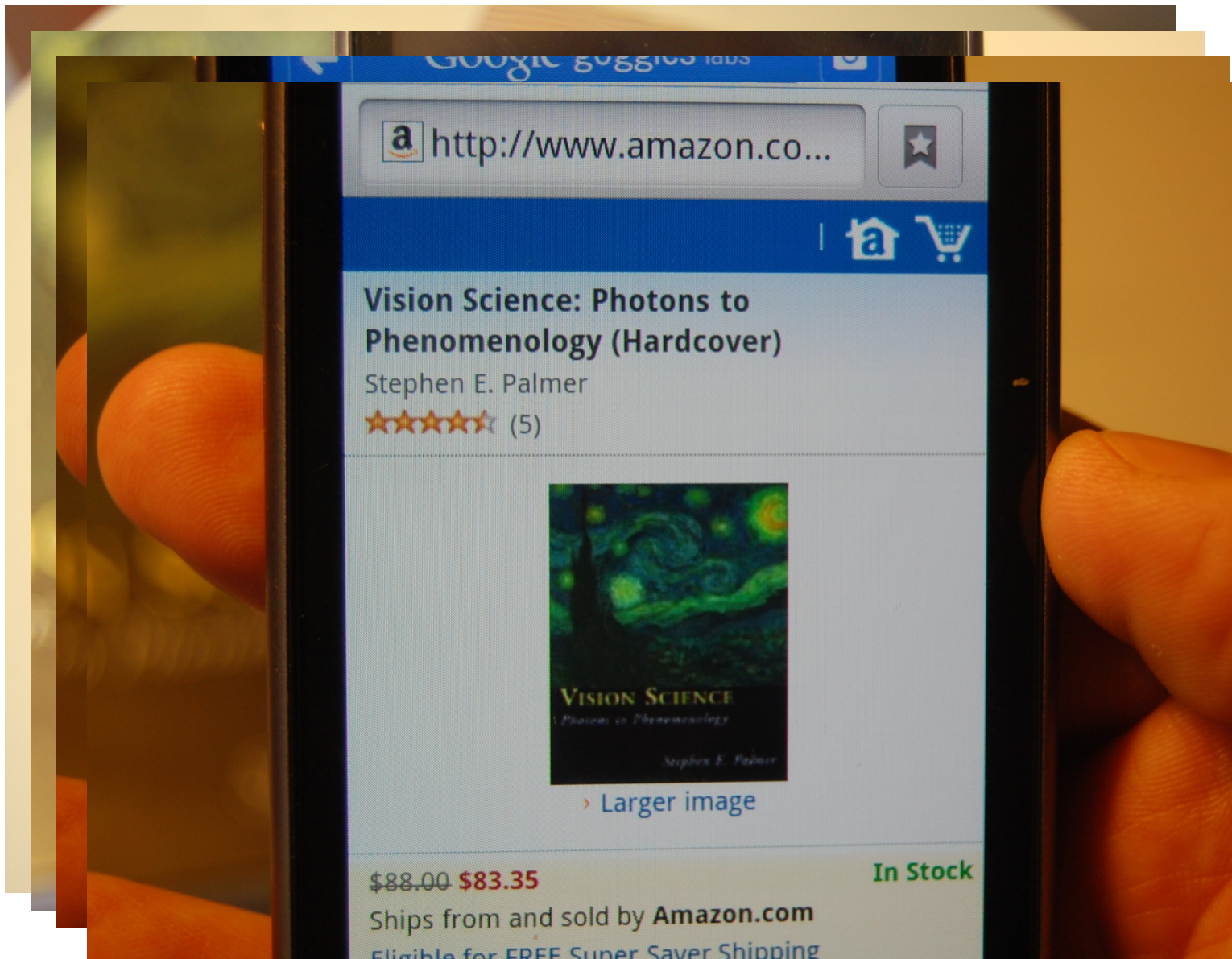
and others... Snaptell.com, Millpix.com



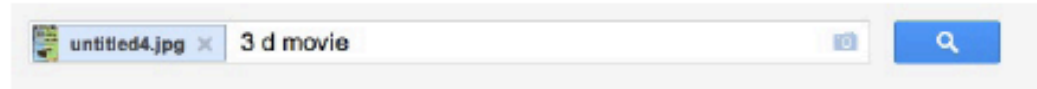
PLINKART

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.

Example



Google images



About 123 results (0.65 seconds)



Image size:
765 x 1024

No other sizes of this image found.

Best guess for this image: [3 d movie](#)

[3D film - Wikipedia, the free encyclopedia](#)

en.wikipedia.org/wiki/3D_film

The following are some of the technical details and methodologies employed in some of the more notable 3-D movie systems that have been developed. 6 images

[3D Movies, 3D Blu-ray Movies, 3D Blu-ray Players - Blu-ray.com](#)

www.blu-ray.com/3d/

3D movies and 3D reviews, 3D Blu-ray release dates for new and upcoming 3D movies, 3D Blu-ray players, 3D HDTVs, 3D projectors and 3D receivers. 75 images

[Visually similar images](#) - [Report images](#)



Pages that include matching images



300 x 300

[Amazon.com: 3D Movie Making: Stereoscopic Digital Cinema from ...](#)

www.amazon.com › ... › Programming › Graphics & Multimedia

Amazon.com: 3D Movie Making: Stereoscopic Digital Cinema from Script to Screen (9780240811376): Bernard Mendiburu: Books.



122 x 148

[3D Movie Making - Elsevier](#)

www.elsevier.com/wps/product/cws_home/717332

26 Dec 2011 – 3D Movie Making-Hollywood is going 3D! Join the revolution with this primer to all of the essential skills for live action 3D, from preproduction ...

kooaba Paperboy delivers digital extras for print

by kooabaChannel

kooaba Paperboy delivers digital extras for print

by kooabaChannel



What next?

Searchable visual memory?



Google Glass

I took ~3K images in one year.

What next?

Searchable visual memory?



Google Glass, 12 hrs/day @ 24 fps: Capture 379M frames in one year

What if the current 1 Billion* smartphone users switch to Glass?

Are the current methods going to scale to this size of the data?

New representations are likely to be needed.

* Source: <http://www.digitalbuzzblog.com/2011-mobile-statistics-stats-facts-marketing-infographic/>

What next?

Searchable visual memory?



What do we want to search for?

Current visual descriptors work well only for certain classes of objects.

What next?

Visual search for texture-less, wiry, deformable and 3D objects..



The IKEA problem

Recognize all IKEA furniture in all YouTube videos



Figure: IKEA

Example:
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



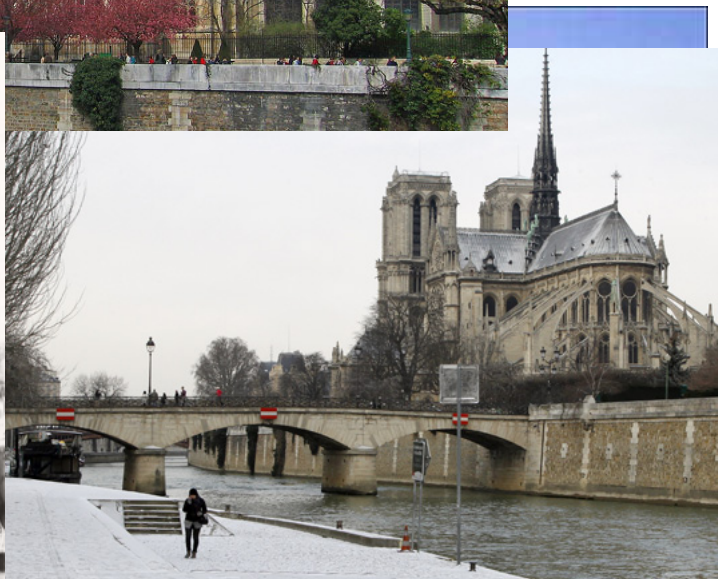
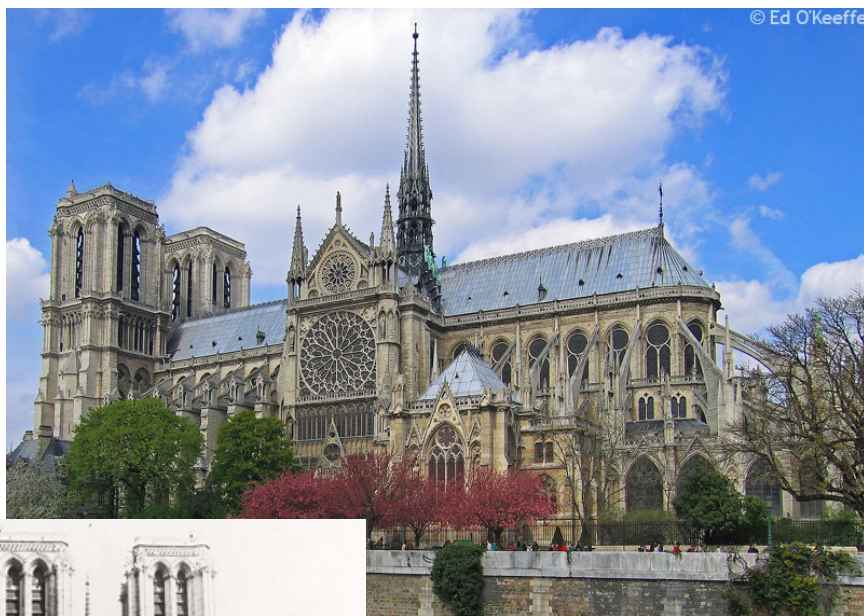
same category



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

Recognize and match the same scene/object across large appearance changes.

Examples: non-photographic depictions, degradation over time, change of season, ...



Example: Automatic alignment of paintings and photographs depicting the same 3D scene

[Russell, Sivic, Ponce and Dessales, 2011]



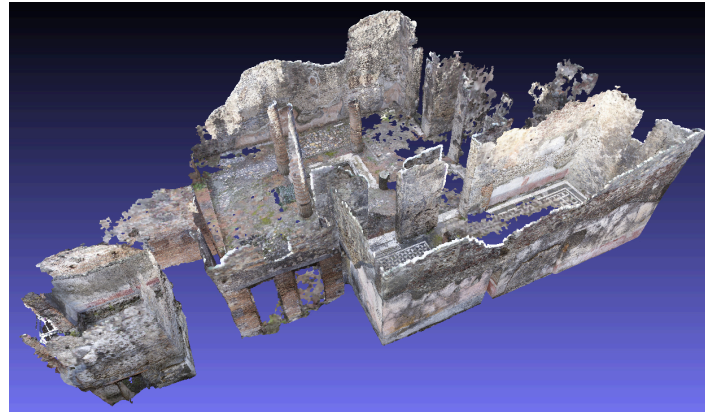
Automatic alignment of paintings and photographs depicting the same 3D scene [Russell, Sivic, Ponce and Dessales, 2011]

Inputs



Photographs

Outputs



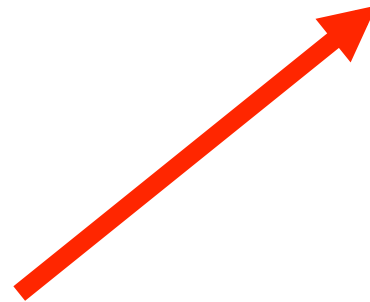
3D model



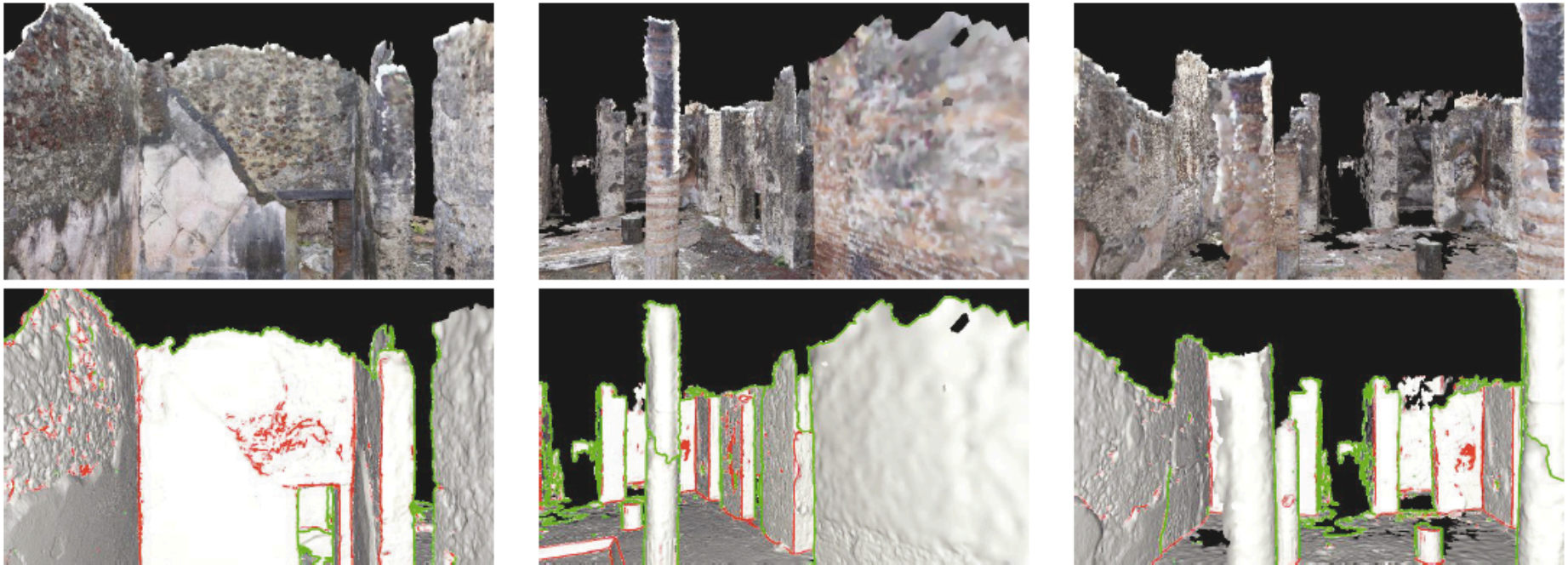
Painting



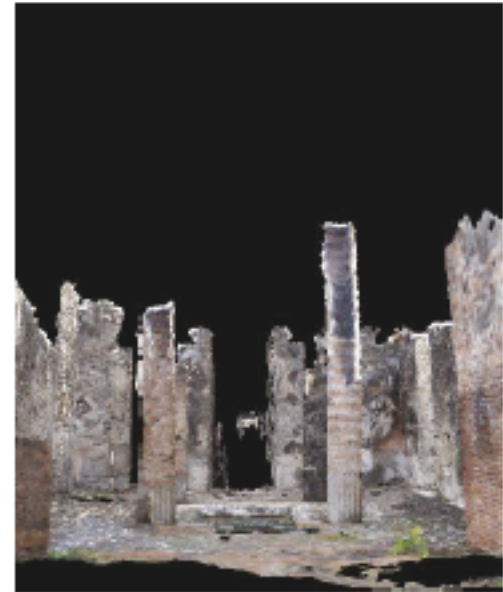
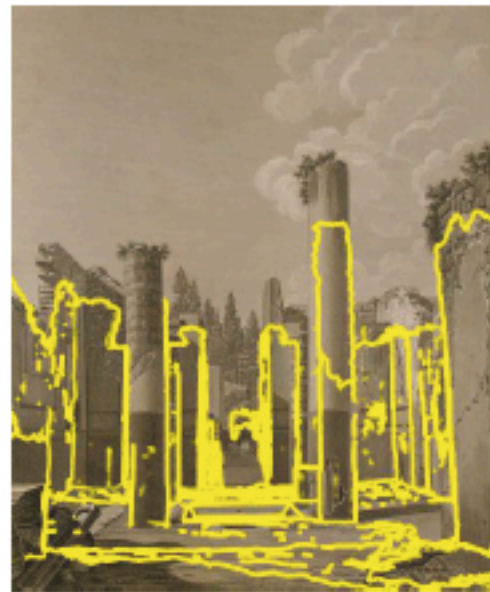
Viewpoint of painting



Contour based alignment



Detected contours on the 3D model





Architecture retrieval

Goal: Given a query from one city, find places in another city with a similar layout of architectural elements (e.g. windows, balconies, doors, etc.)



Query Image in Prague

Retrieved Images in Paris

[Doersch, Singh, Gupta, Sivic, Efros, SIGGRAPH 2012]

Architecture retrieval

- Build visual vocabulary of architectural elements from Street-view images
- Use geotags as a form of supervision



Paris, France

Prague, Czech Republic

London, England

Architecture retrieval

- Detect visual elements in all images
- Match on visual elements and their spatial layout

Applications:



Questions?