Computer Vision

CS-E4850, 5 study credits

Lecturer: Juho Kannala

Lecture 6: Large-scale object instance recognition/retrieval

- Given a large image database of object instances, we would like to quickly recognize the objects present in a query image

- Or, given a query image of an object instance, we would like to retrieve all images of the same object from the database

Acknowledgement: many slides from James Hays, Kristen Grauman, Svetlana Lazebnik, Ondrej Chum, David Nister and others (detailed credits on individual slides)

Reading

- Szeliski's book, Section 6.1-6.2 in 2nd edition (or 14.3 in 1st edition)

- Sivic & Zisserman: Video Google, 2003
  - http://www.robots.ox.ac.uk/~vgg/research/vgoogle/

- Nister & Stewenius: Scalable recognition with a vocabulary tree, 2006
  - http://vis.uky.edu/~stewe/ukbench/

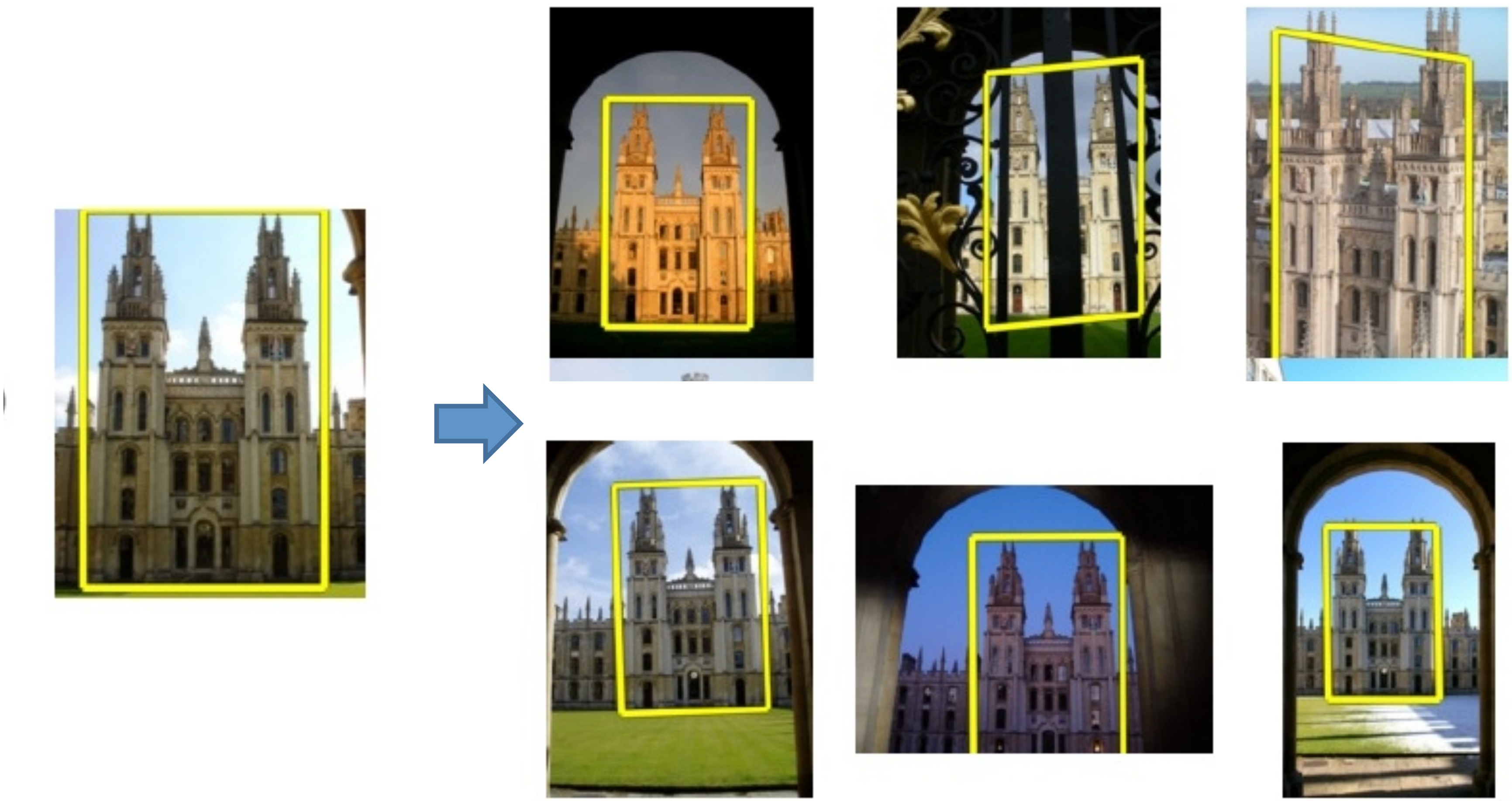
- Philbin et al.: Object retrieval with large vocabularies, 2007
  - http://www.robots.ox.ac.uk/~vgg/research/oxbuildings/index.html

- Software:
  - http://www.robots.ox.ac.uk/~vgg/practicals/instance-recognition/index.html

Local features for object instance recognition

D. Lowe (1999, 2004)

How to quickly find images in a large database that match a given image region?



Large-scale image search

Combining local features, indexing, and spatial constraints

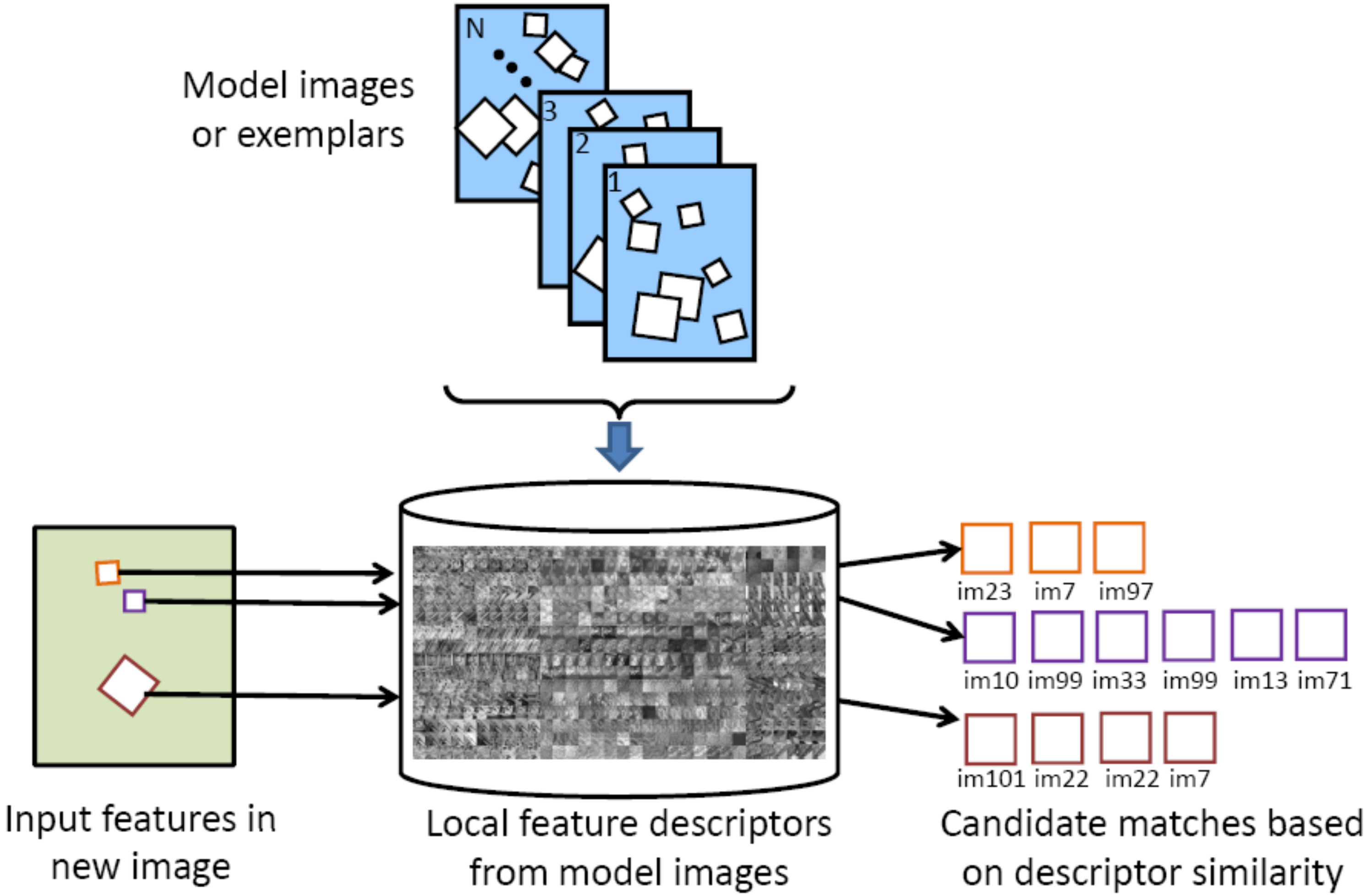


Image credit: K. Grauman and B. Leibe

Large-scale image search

Combining local features, indexing, and spatial constraints

Philbin et al. '07

Large-scale image search

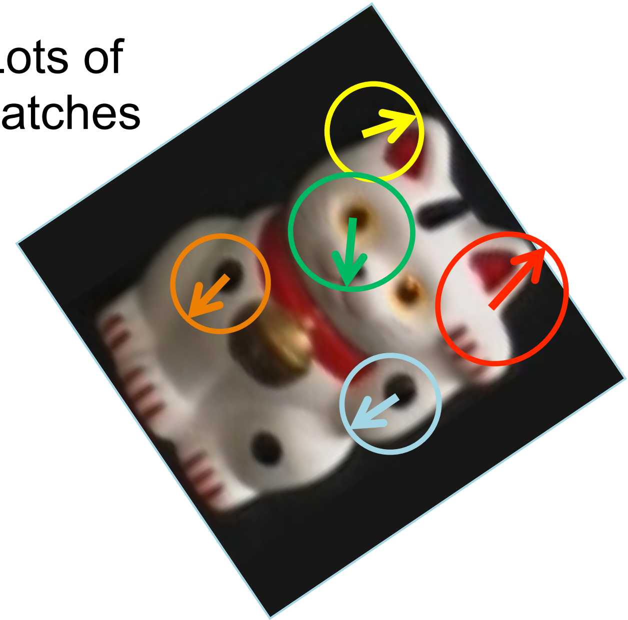
Combining local features, indexing, and spatial constraints

Svetlana Lazebnik

Simple idea

See how many keypoints are close to keypoints in each other image

Lots of Matches

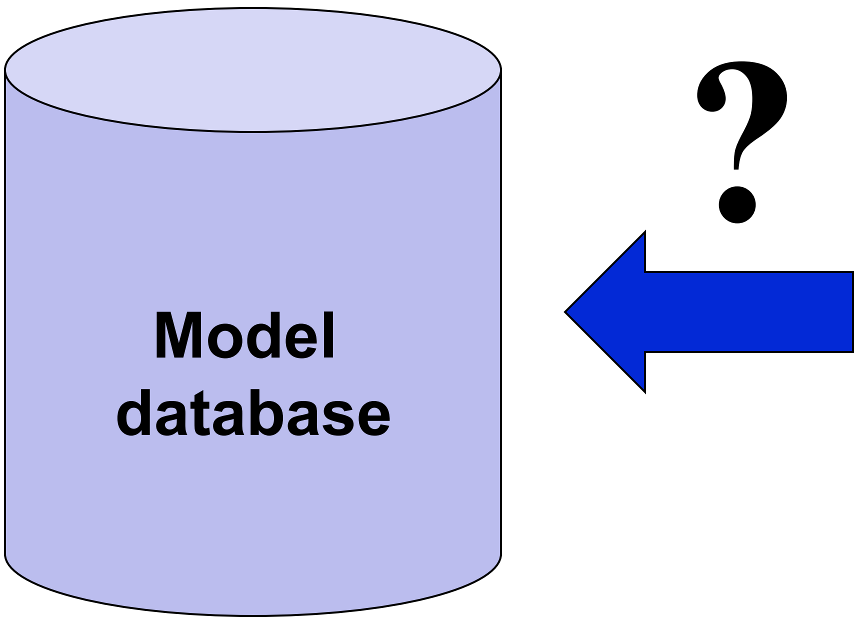


Few or No Matches

But this will be really, really slow!

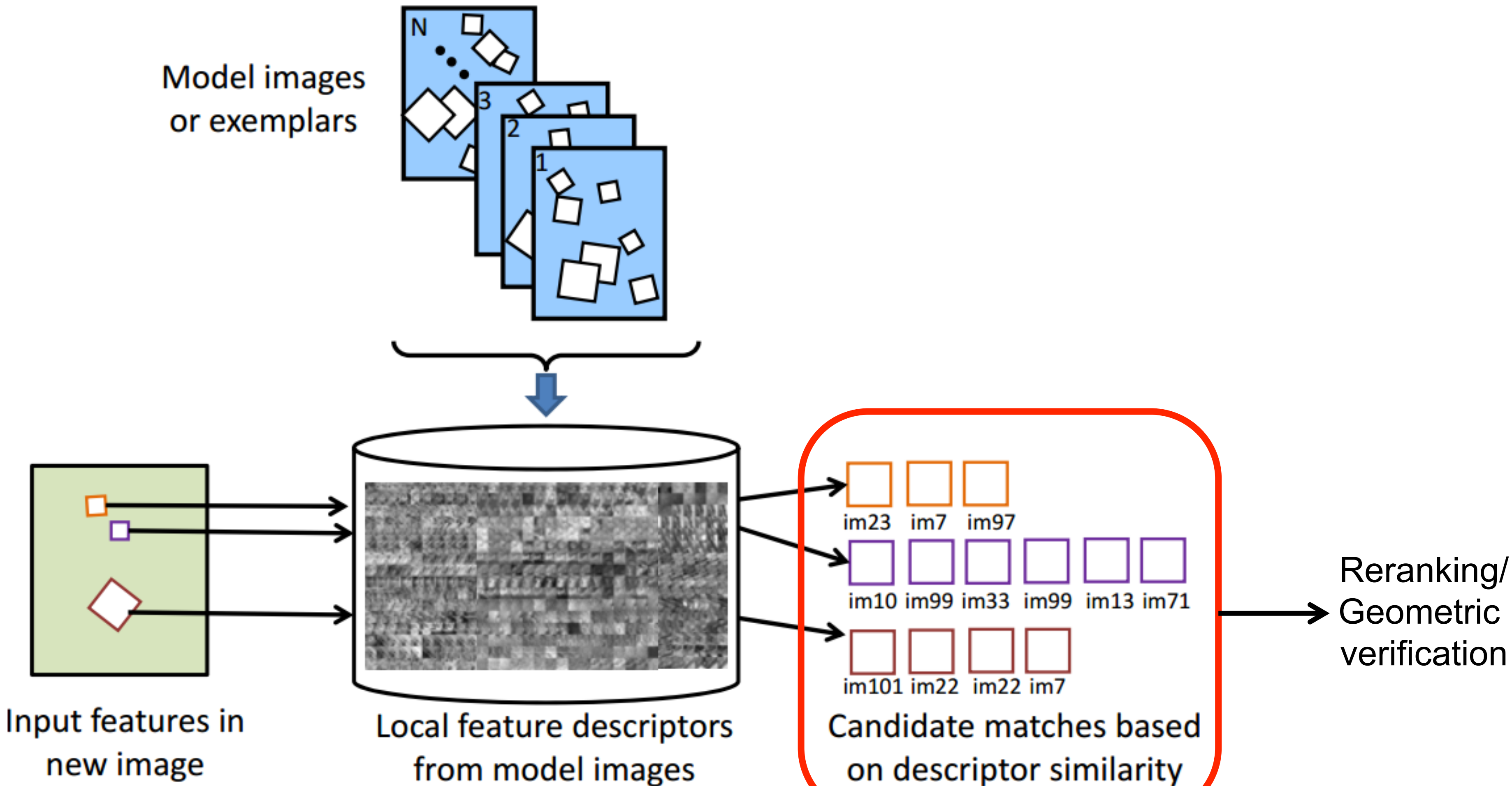
Scalability: Alignment to large databases

- What if we need to align a test image with thousands or millions of images in a model database?
  - Efficient putative match generation
    - Approximate descriptor similarity search, inverted indices



Test image

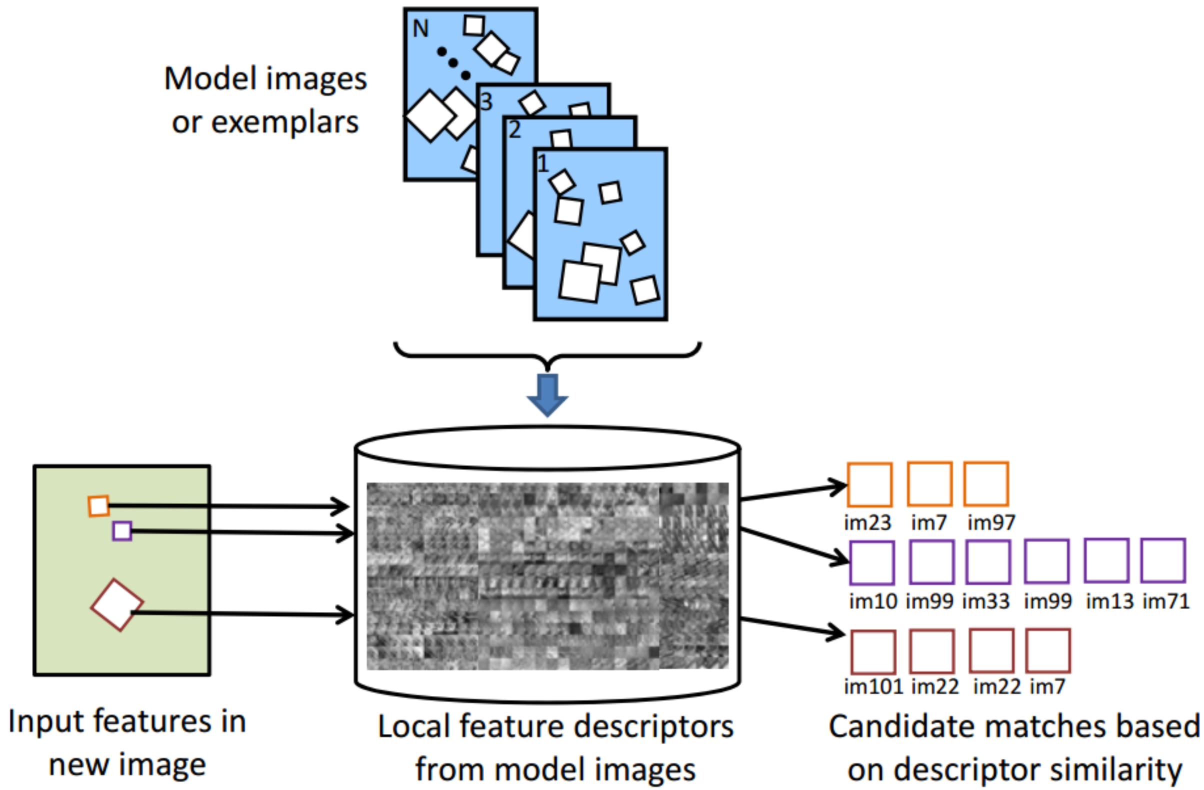
Large-scale visual search



Inverted indexing

Figure from: Kristen Grauman and Bastian Leibe, [Visual Object Recognition](), Synthesis Lectures on Artificial Intelligence and Machine Learning, April 2011, Vol. 5, No. 2, Pages 1-181

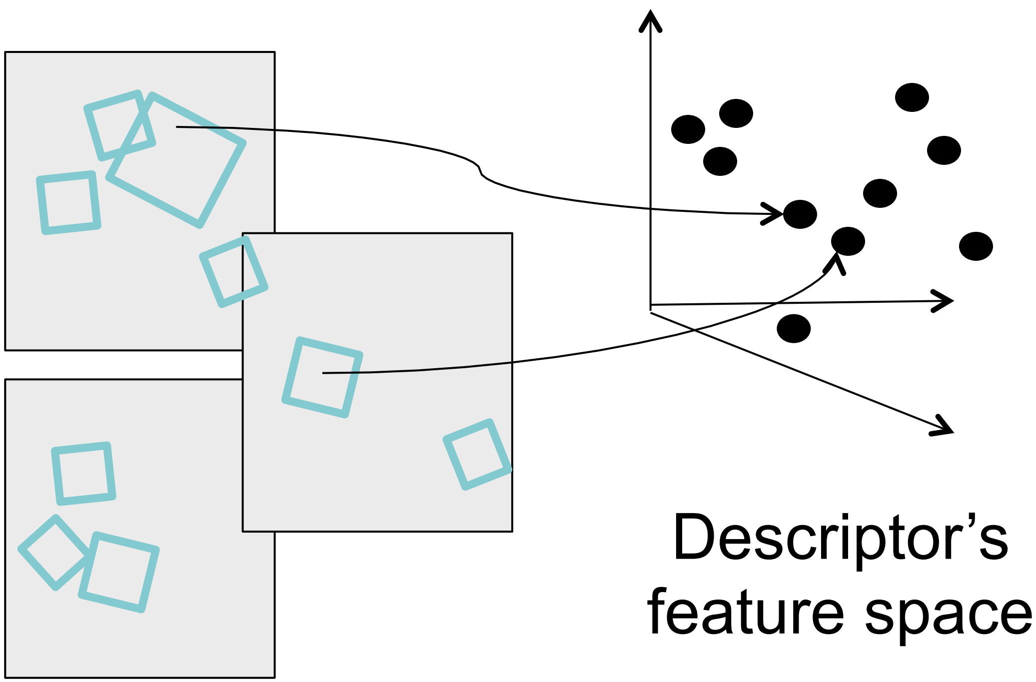
How to do the indexing?



- Idea: find a set of *visual codewords* to which descriptors can be *quantized*

Indexing local features

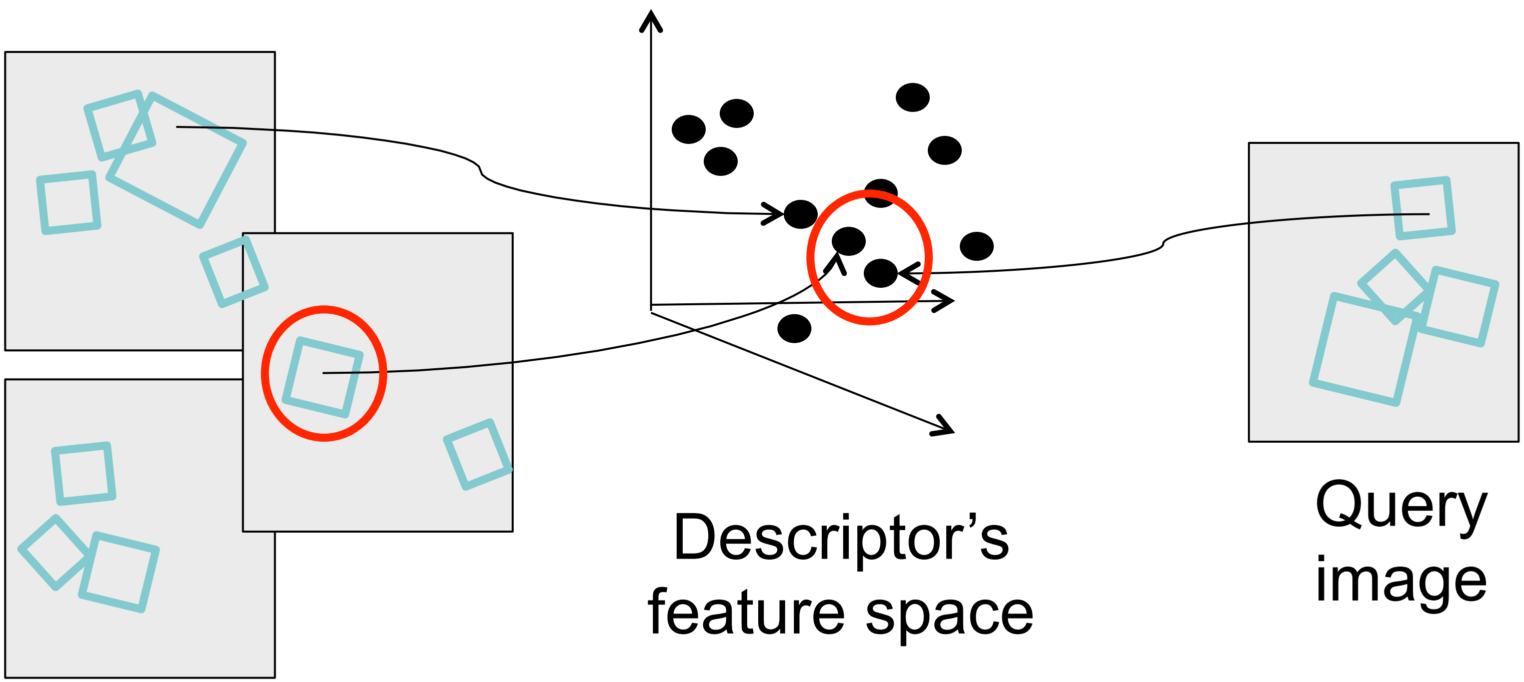
- Each patch / region has a descriptor, which is a point in some high-dimensional feature space (e.g., SIFT)



Kristen Grauman

Indexing local features

- When we see close points in feature space, we have similar descriptors, which indicates similar local content.



Database images

Descriptor's feature space

Query image

Easily can have millions of features to search!

Kristen Grau

Indexing local features: inverted file index

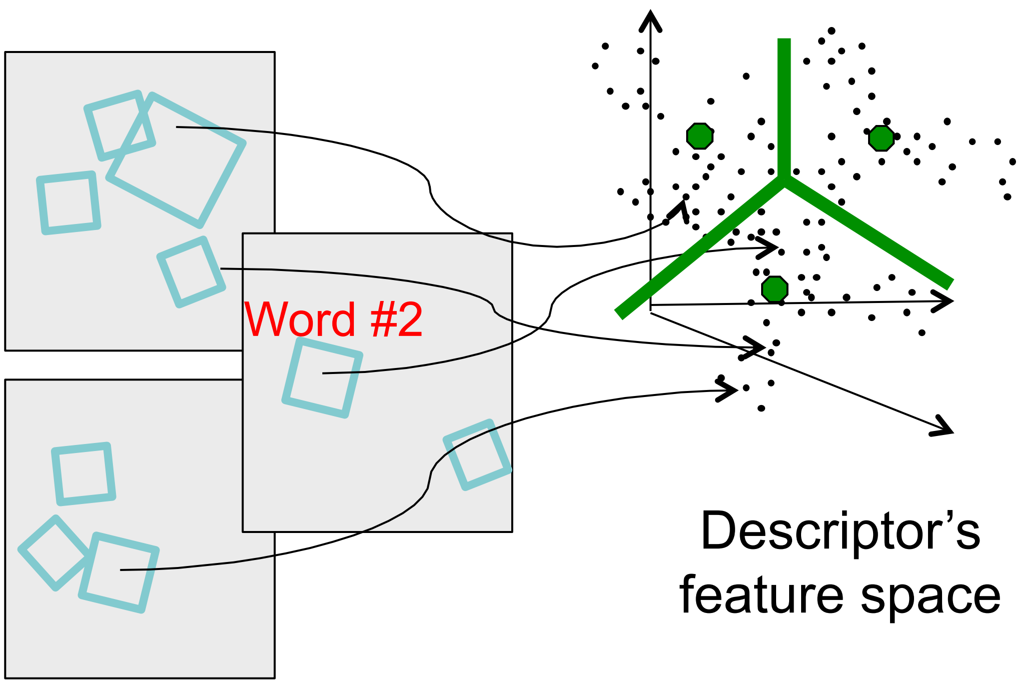
Index

- For text documents, an efficient way to find all *pages* on which a *word* occurs is to use an index…
- We want to find all *images* in which a *feature* occurs.
- To use this idea, we'll need to map our features to "visual words".

Kristen Grauman

Visual words

- Map high-dimensional descriptors to tokens/words by quantizing the feature space



- Quantize via clustering, let cluster centers be the prototype "words"
- Determine which word to assign to each new image region by finding the closest cluster center.

Kristen Grauman

K-means clustering

- Want to minimize sum of squared Euclidean distances between points $\mathbf{x}_i$ and their nearest cluster centers $\mathbf{m}_k$

$$D(X,M) = \sum_{\text{cluster } k} \sum_{\substack{\text{point } i \text{ in} \\ \text{cluster } k}} (\mathbf{x}_i - \mathbf{m}_k)^2$$

Algorithm:

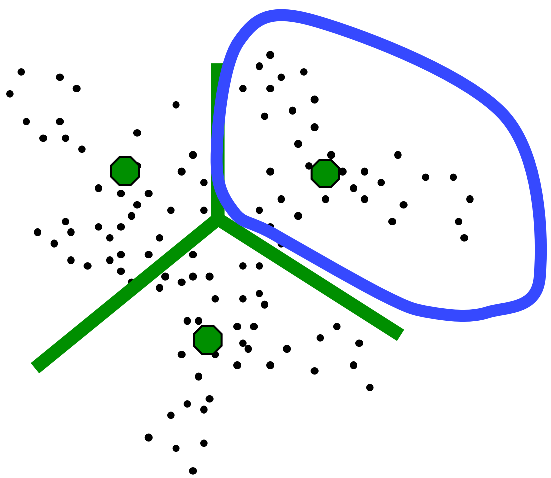
- Randomly initialize K cluster centers
- Iterate until convergence:
  - Assign each data point to the nearest center
  - Recompute each cluster center as the mean of all points assigned to it

K-means demo

Source: http://shabal.in/visuals/kmeans/1.html
Another demo: http://www.kovan.ceng.metu.edu.tr/~maya/kmeans/

Visual words

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



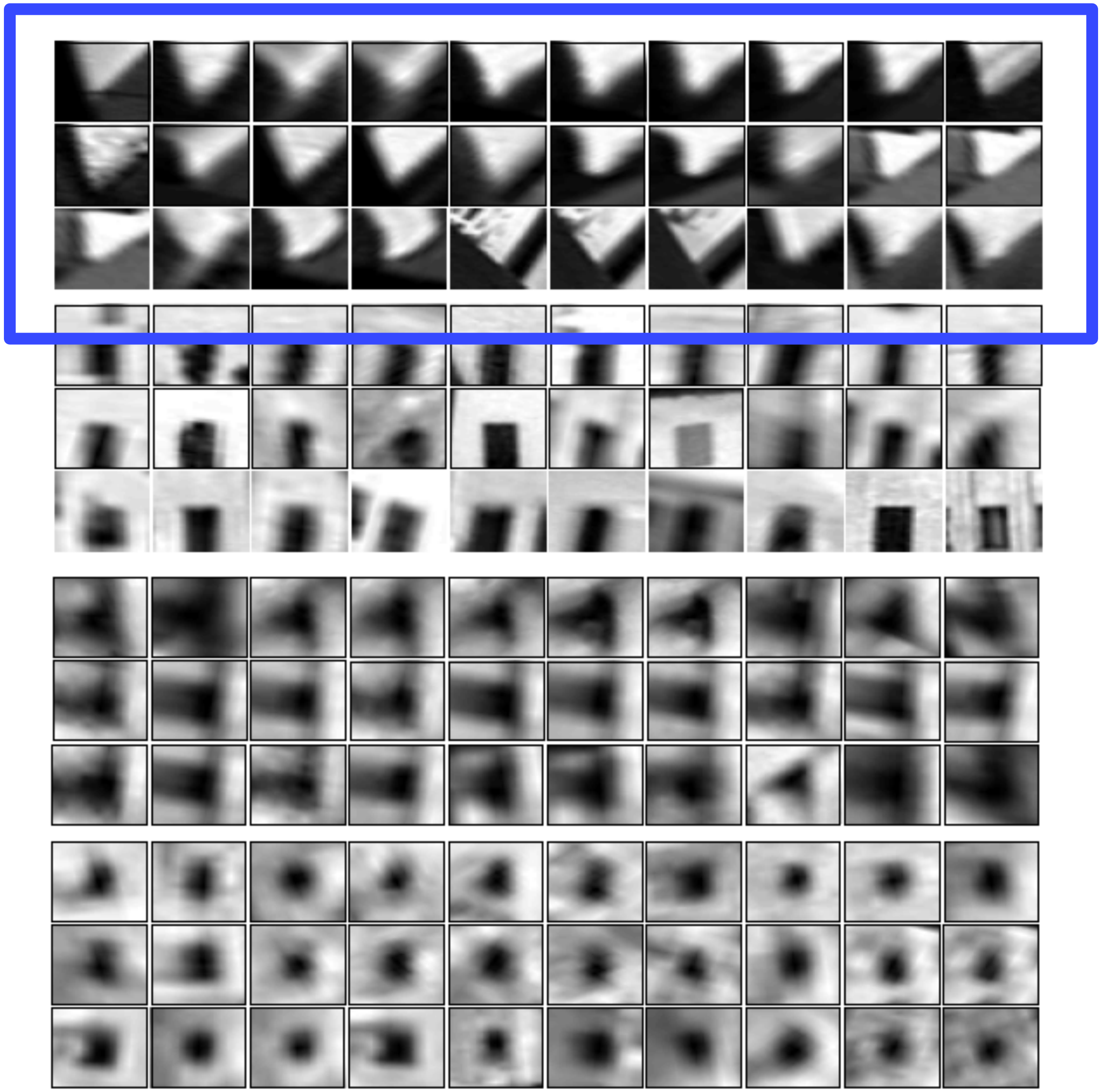
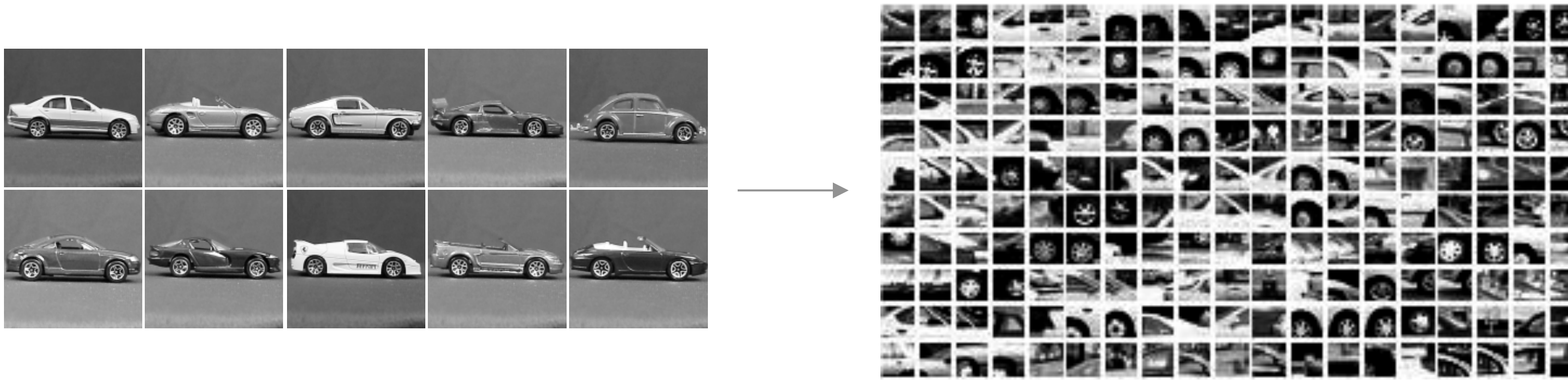
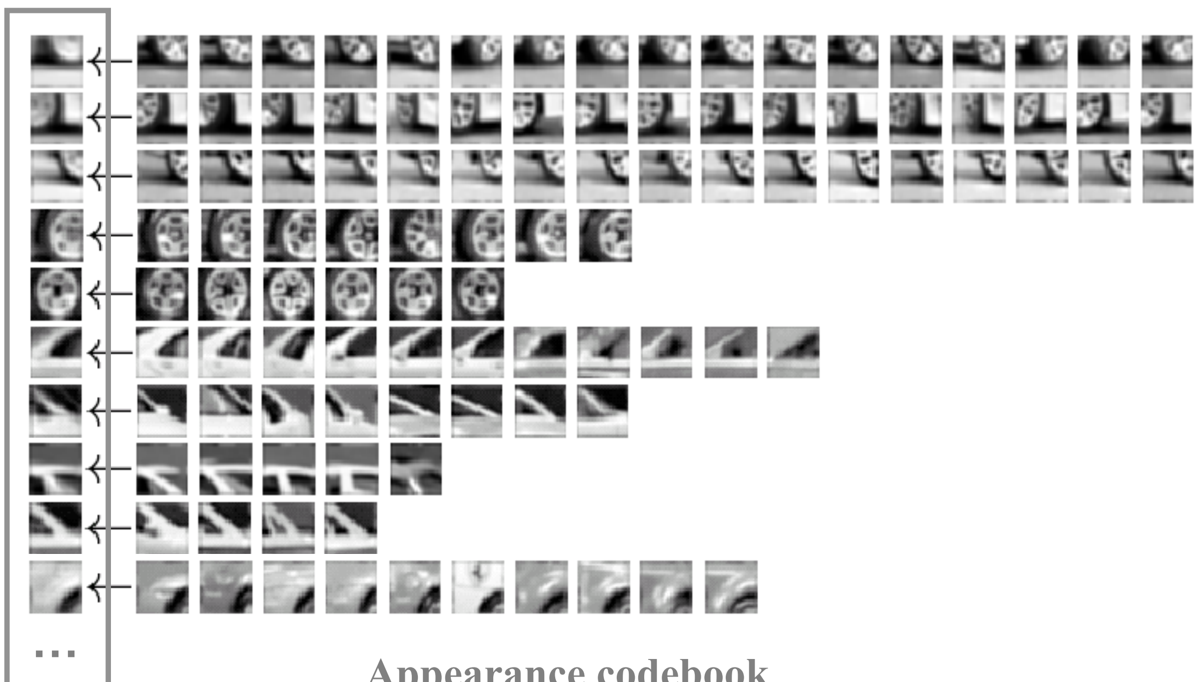


Figure from Sivic & Zisserman, ICCV 2003

Kristen Grauman

Example of a visual codebook

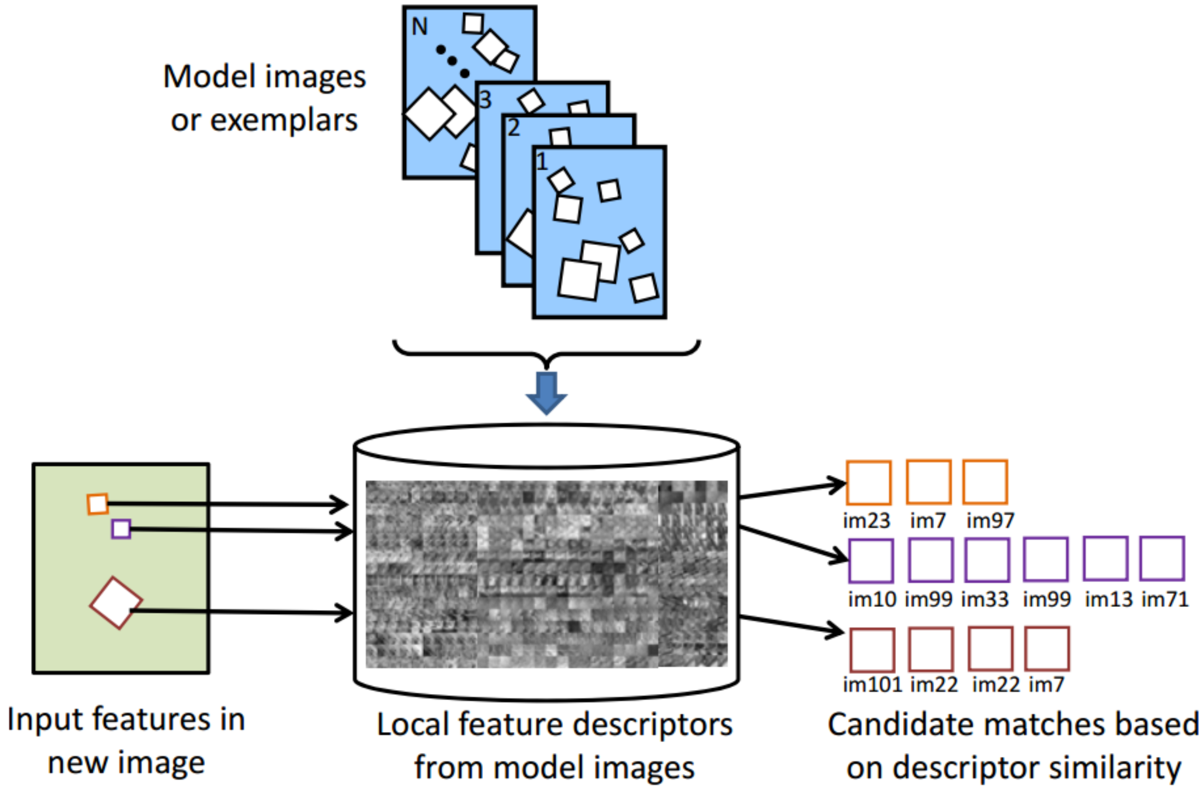




Appearance codebook

Source: B. Leibe

How to do the indexing?



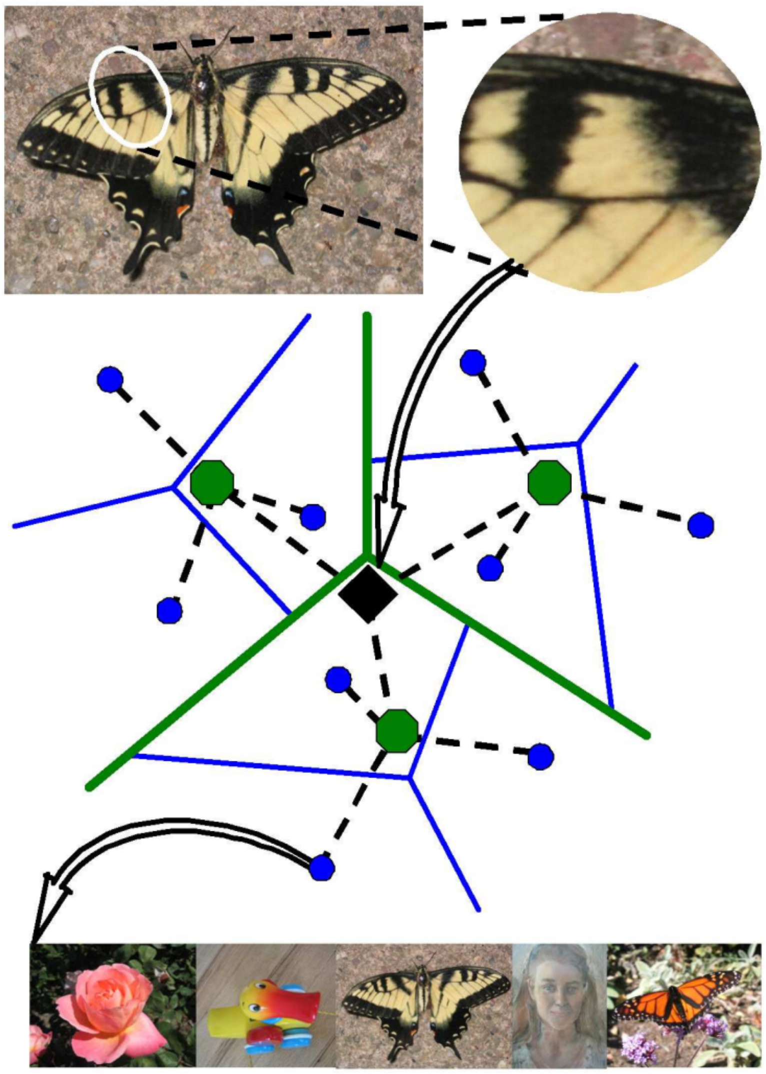
- Cluster descriptors in the database to form codebook
- At query time, quantize descriptors in query image to nearest codevectors
- Problem solved?

Efficient indexing technique: Vocabulary trees

Test image

Vocabulary tree with inverted index

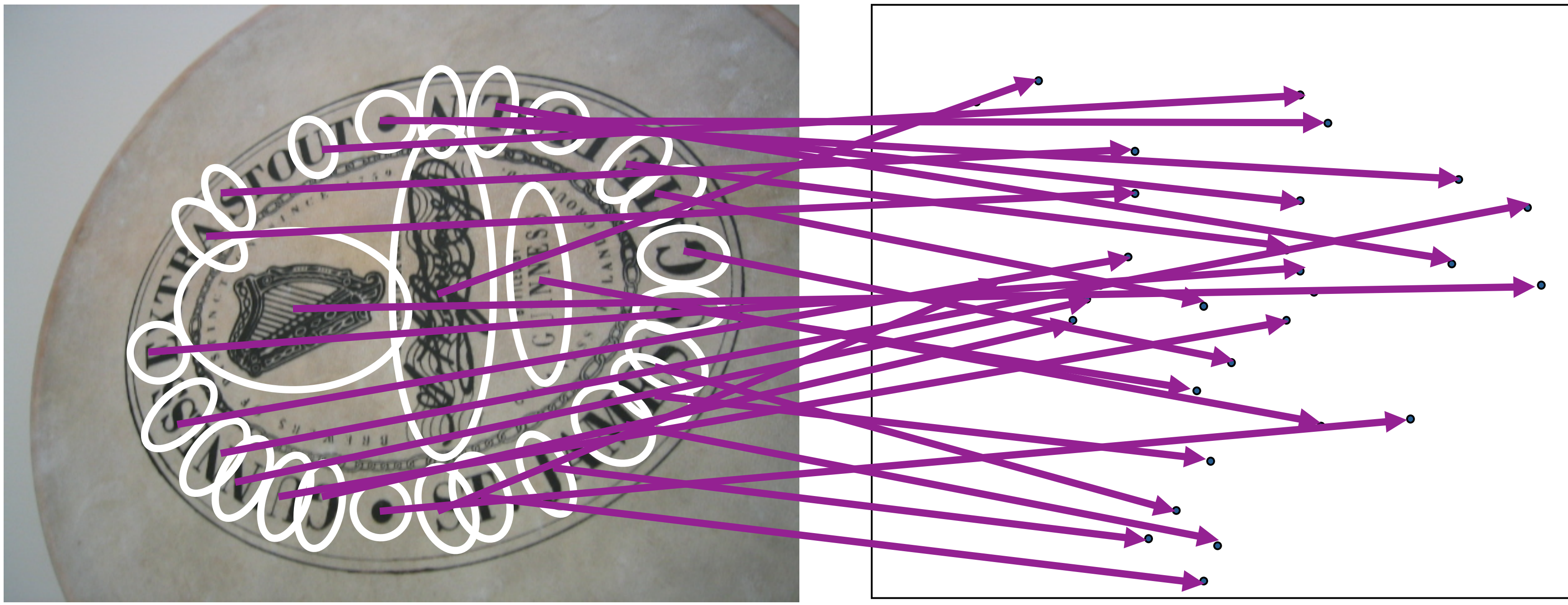
Database

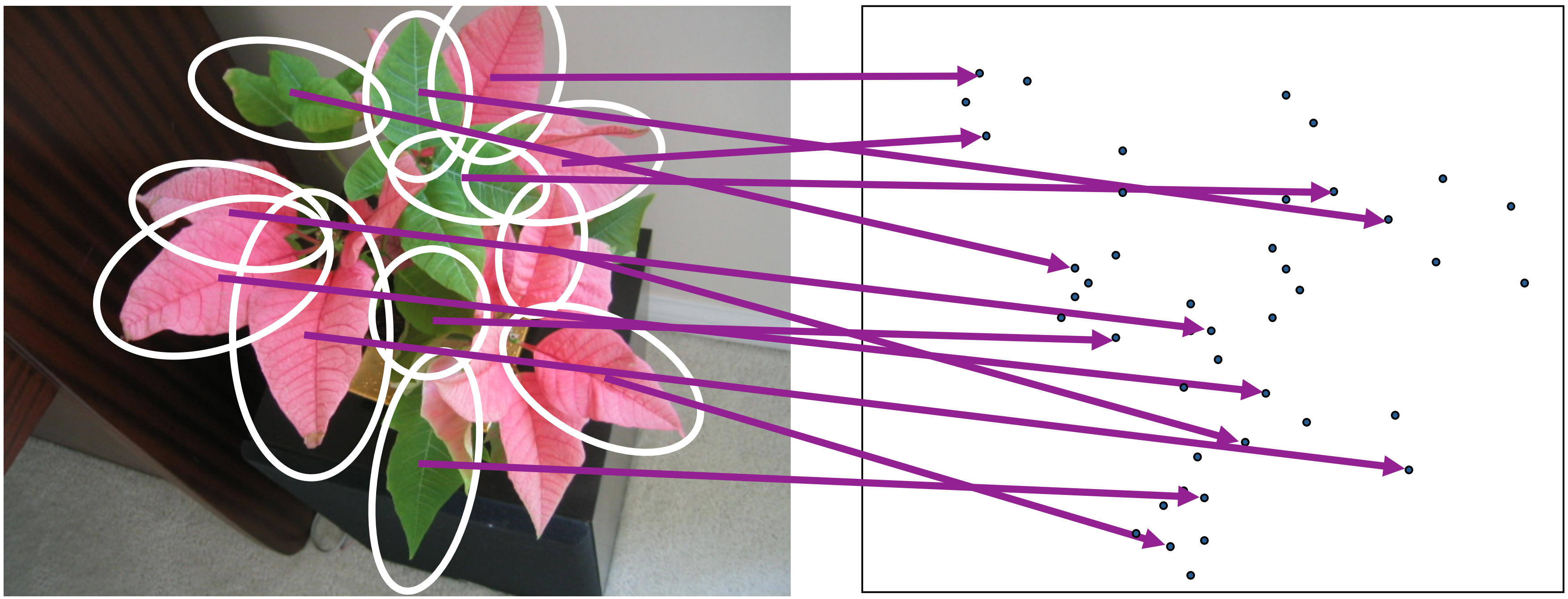


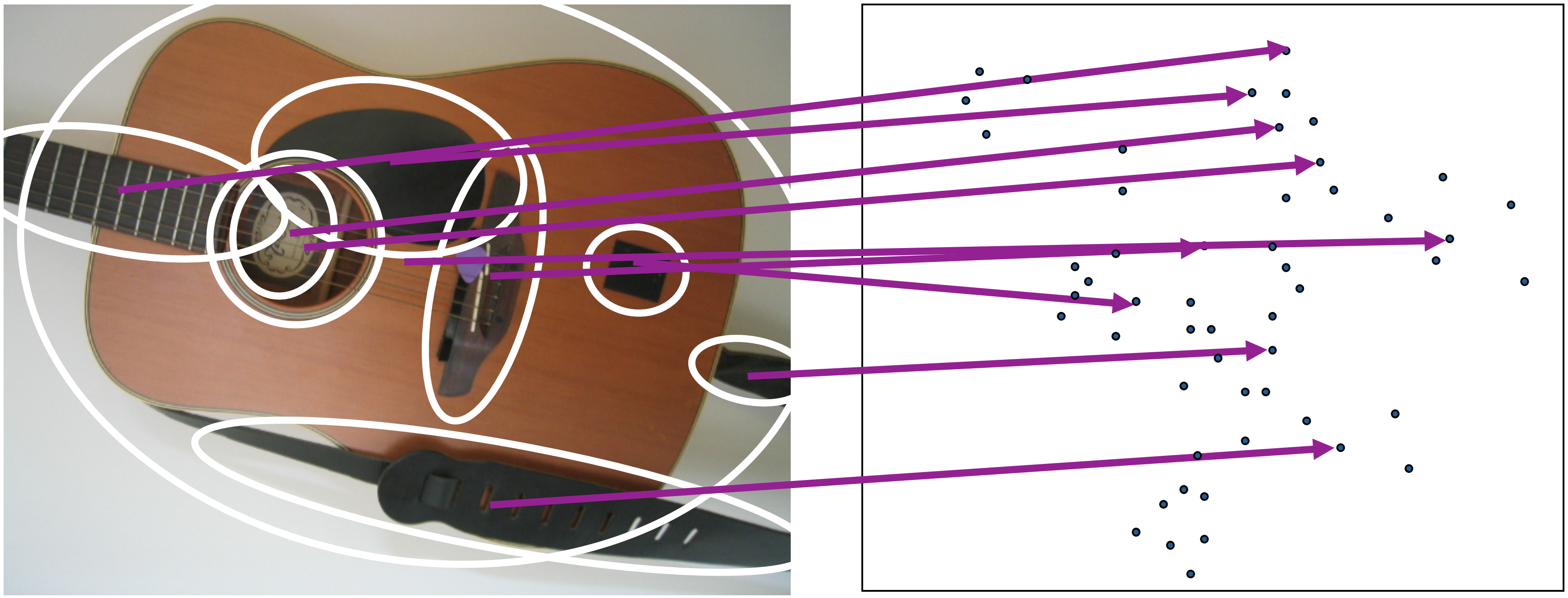
D. Nistér and H. Stewénius, [Scalable Recognition with a Vocabulary Tree](), CVPR 2006

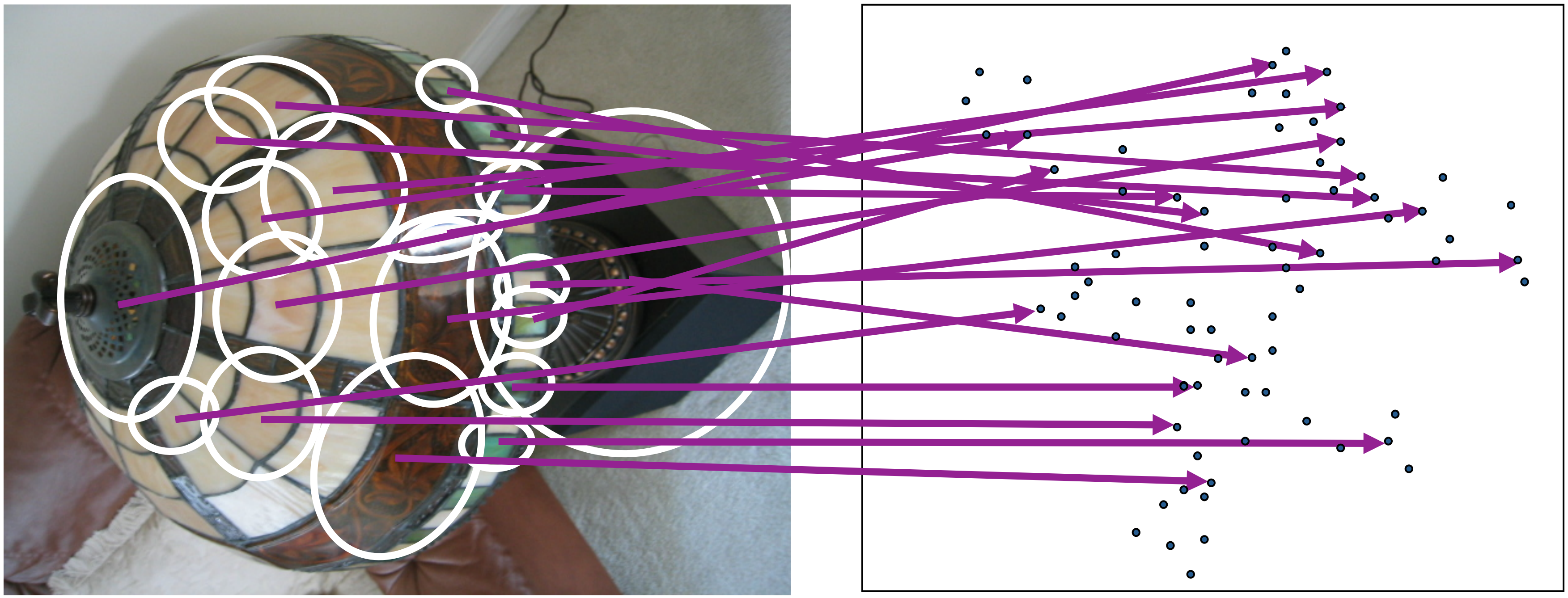
Recognition with K-tree

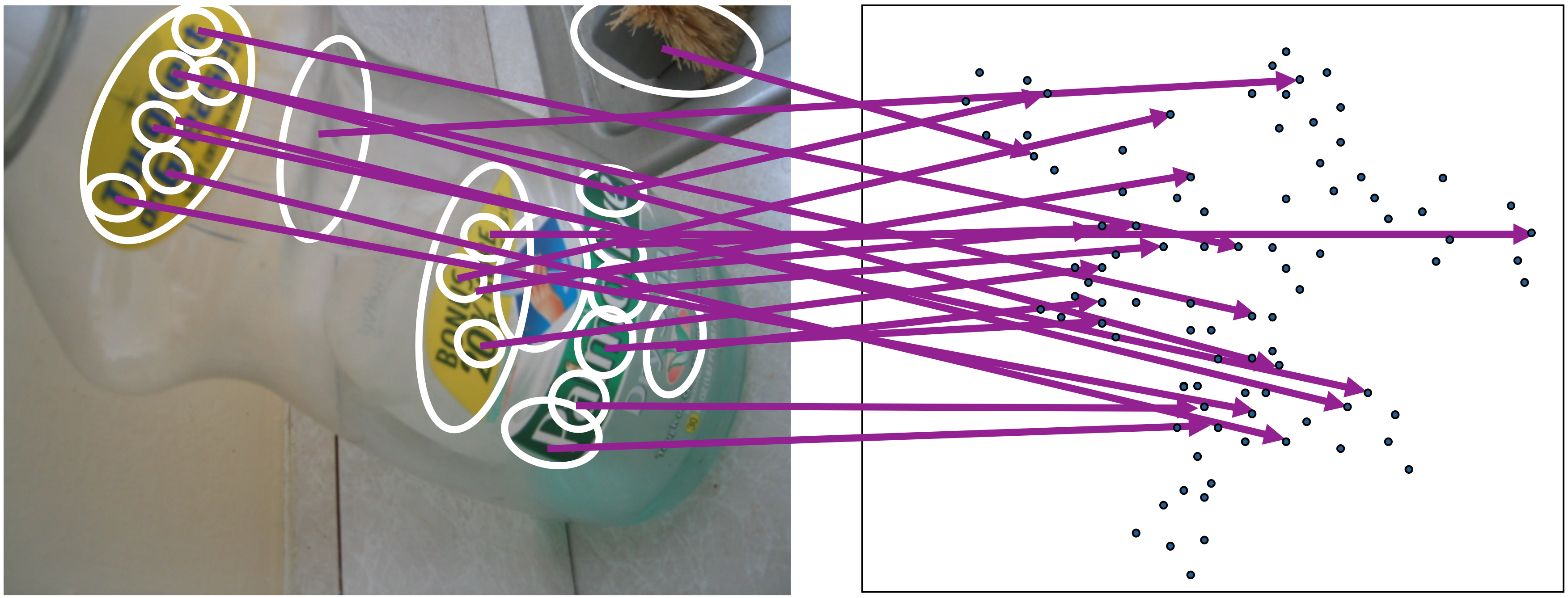
Following slides by David Nister (CVPR 2006)

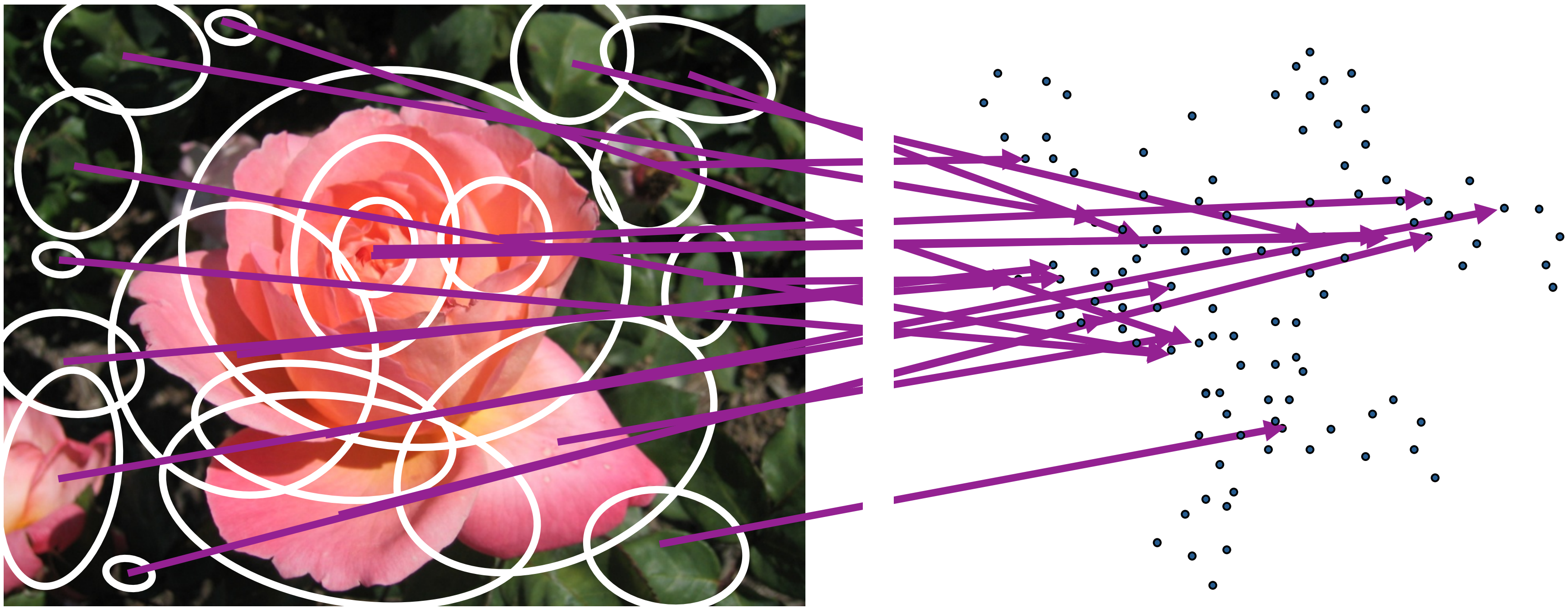


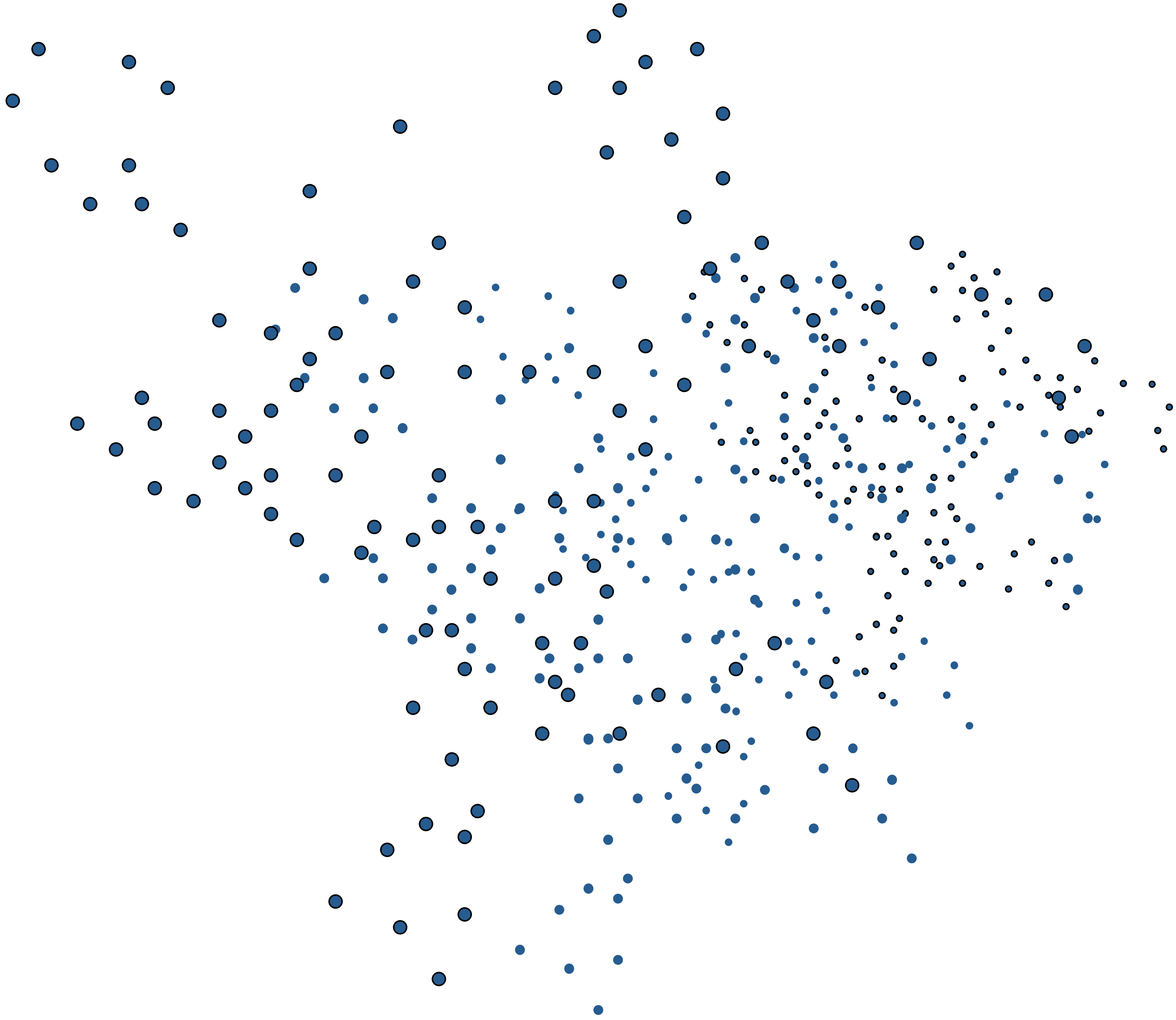


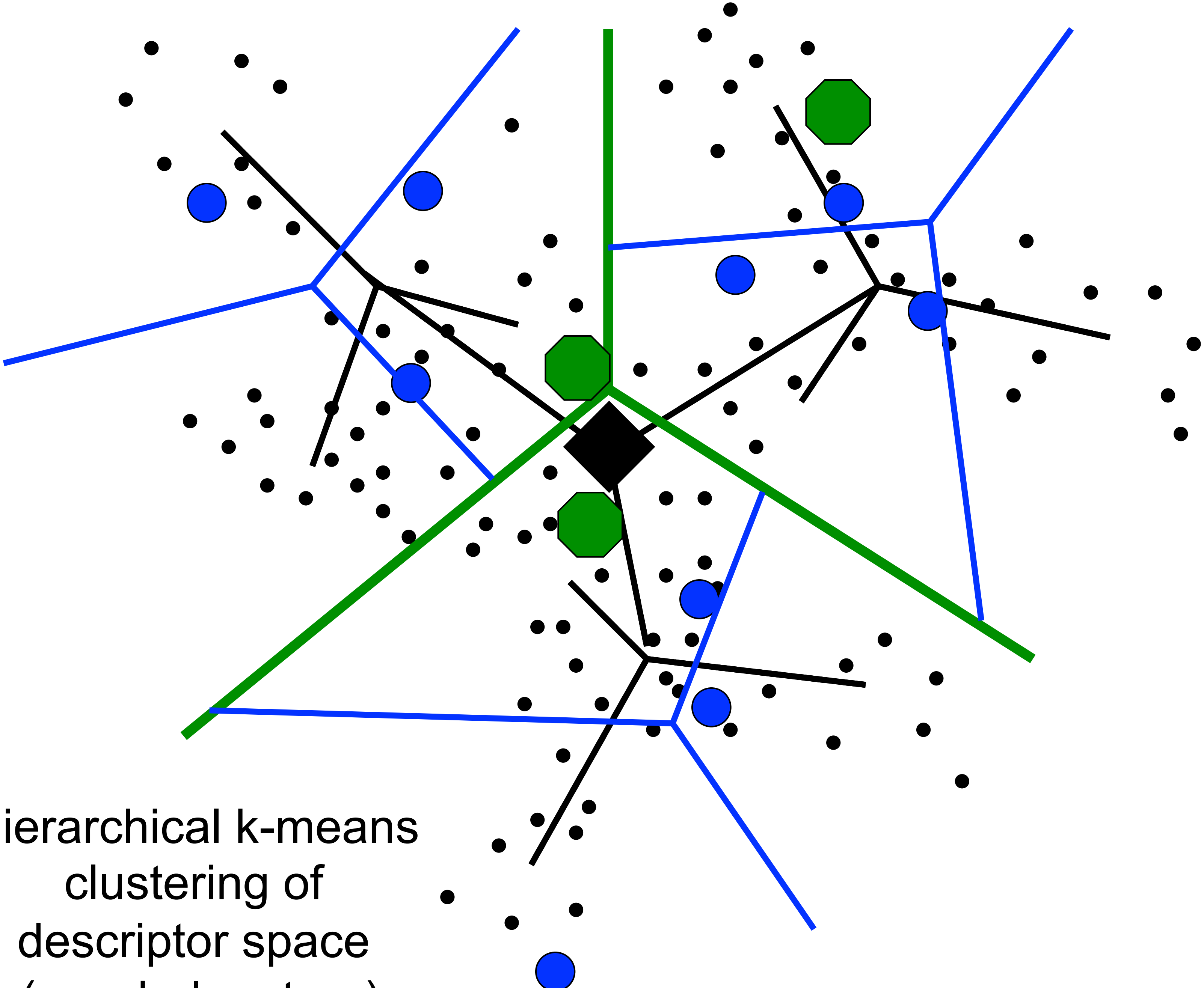






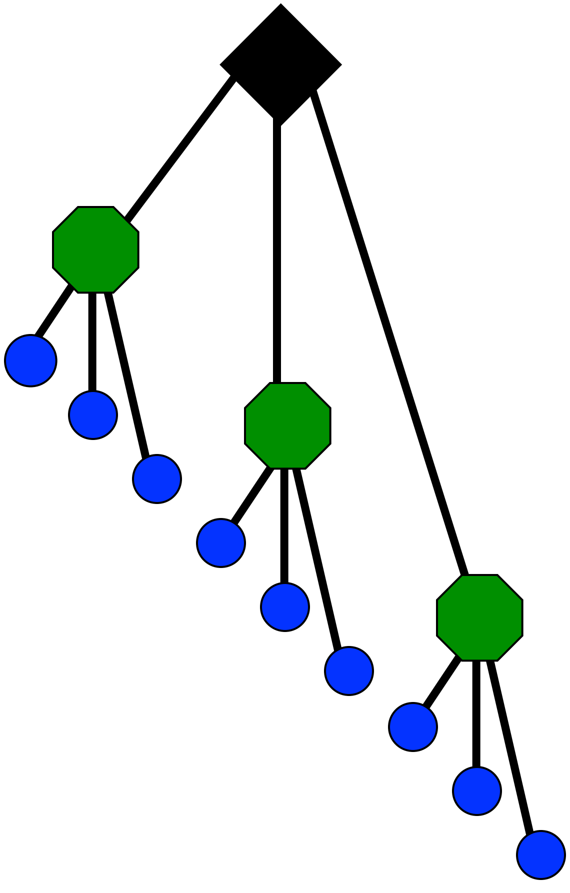






Hierarchical k-means
clustering of
descriptor space
(vocabulary tree)

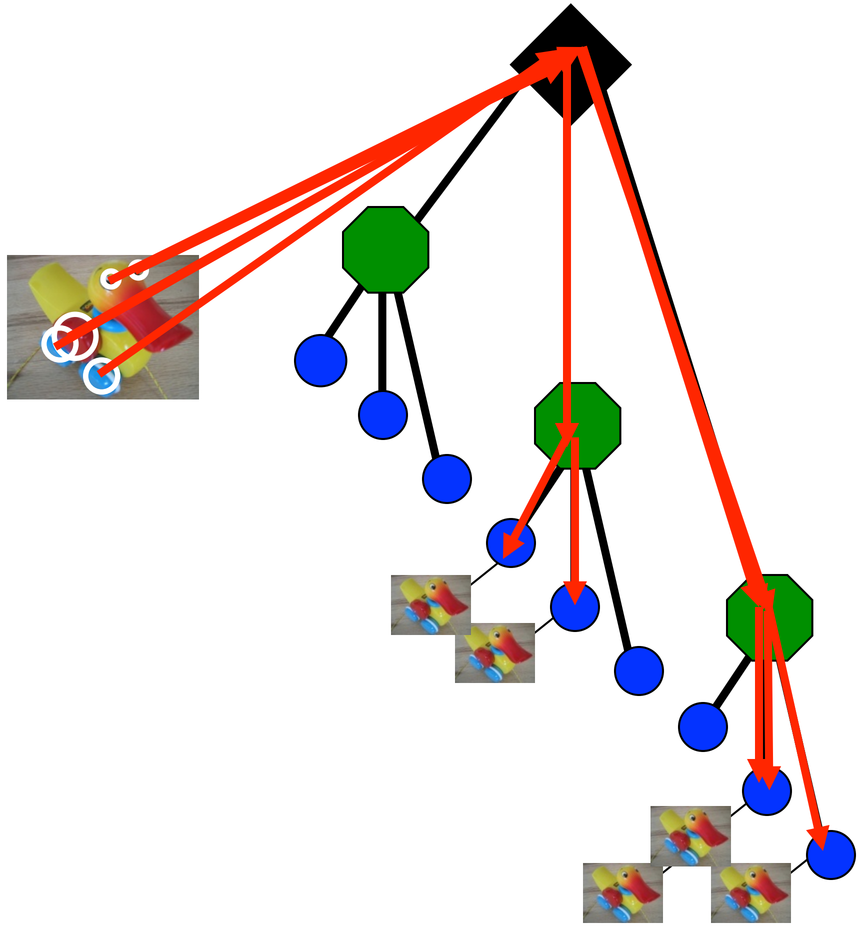
Slide credit: D. Nister



Vocabulary tree/inverted index

Slide credit: D. Nister

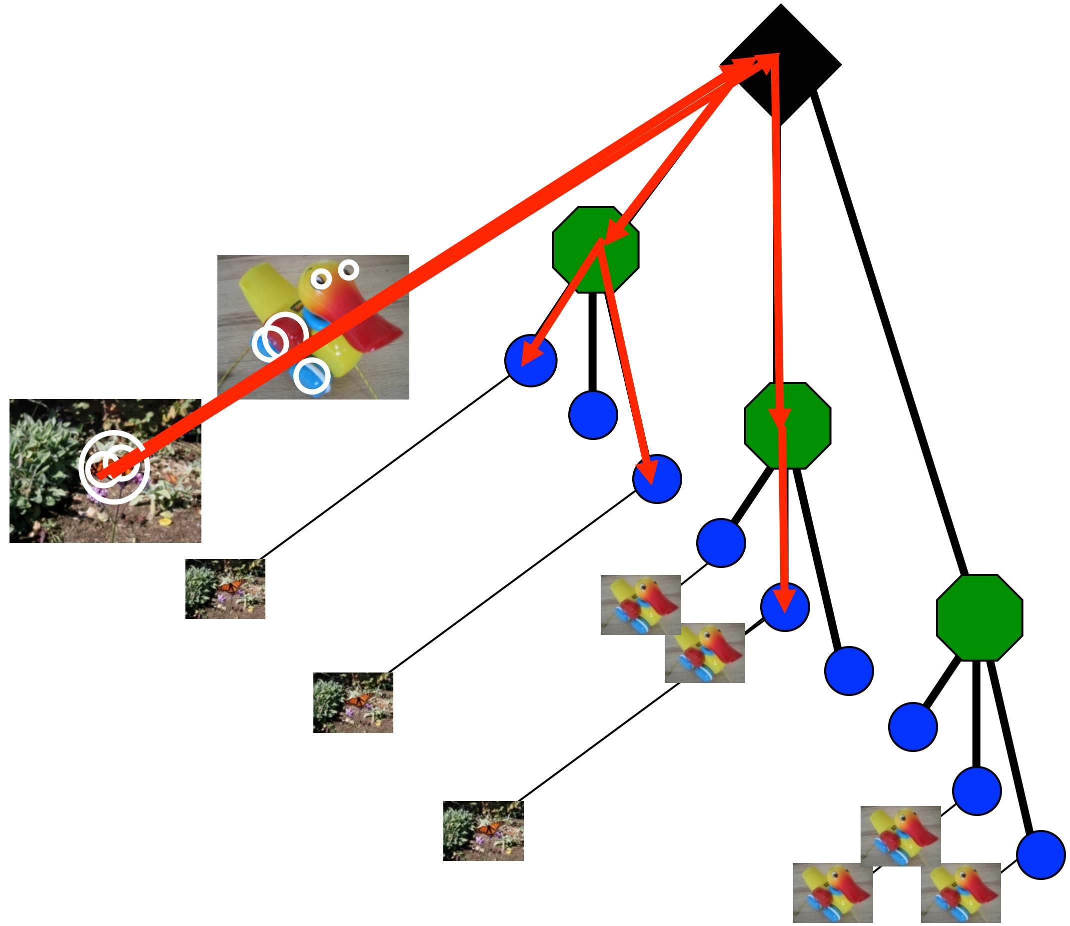
Model images



Populating the vocabulary tree/inverted index

Slide credit: D. Nister

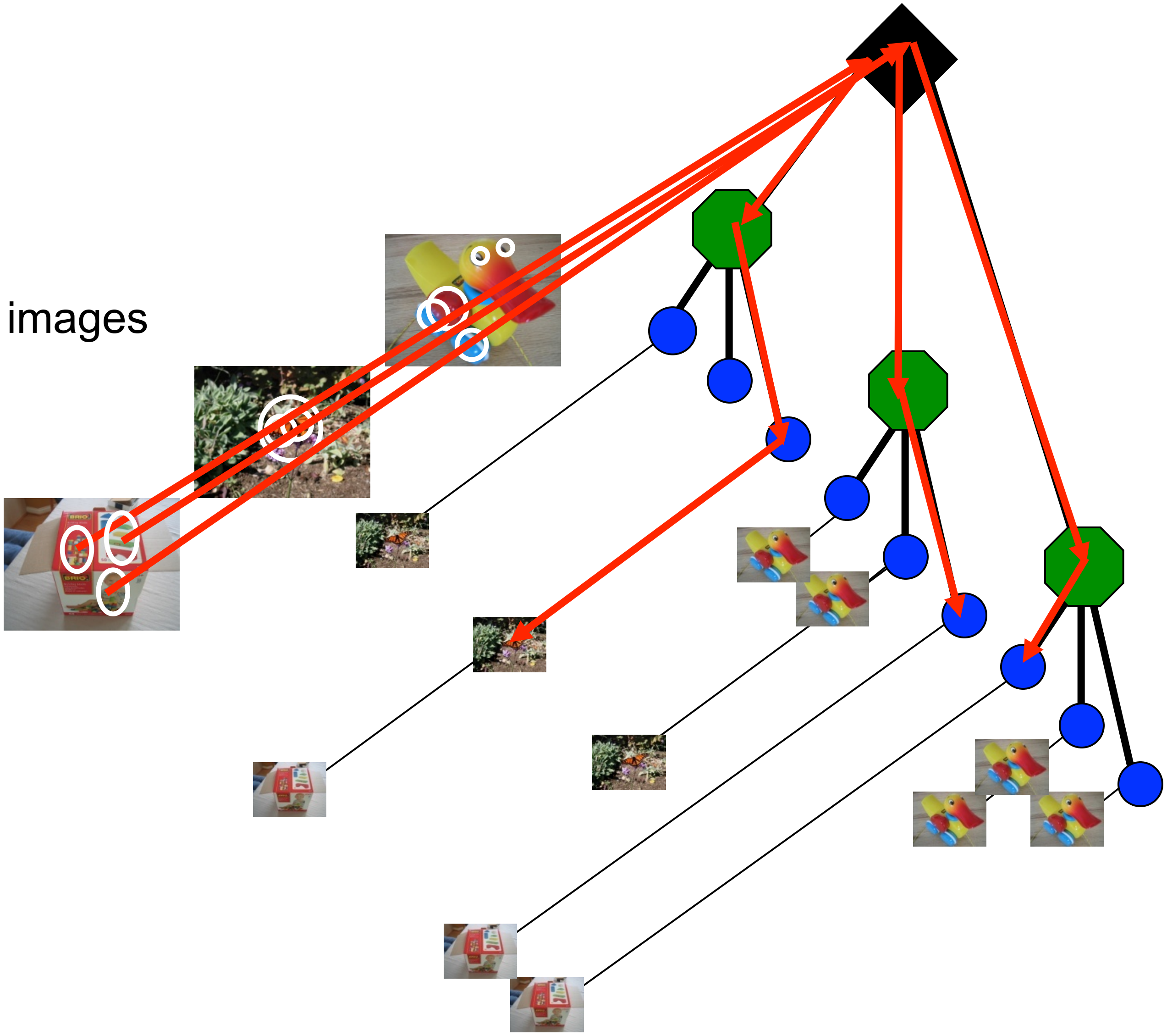
Model images



Populating the vocabulary tree/inverted index

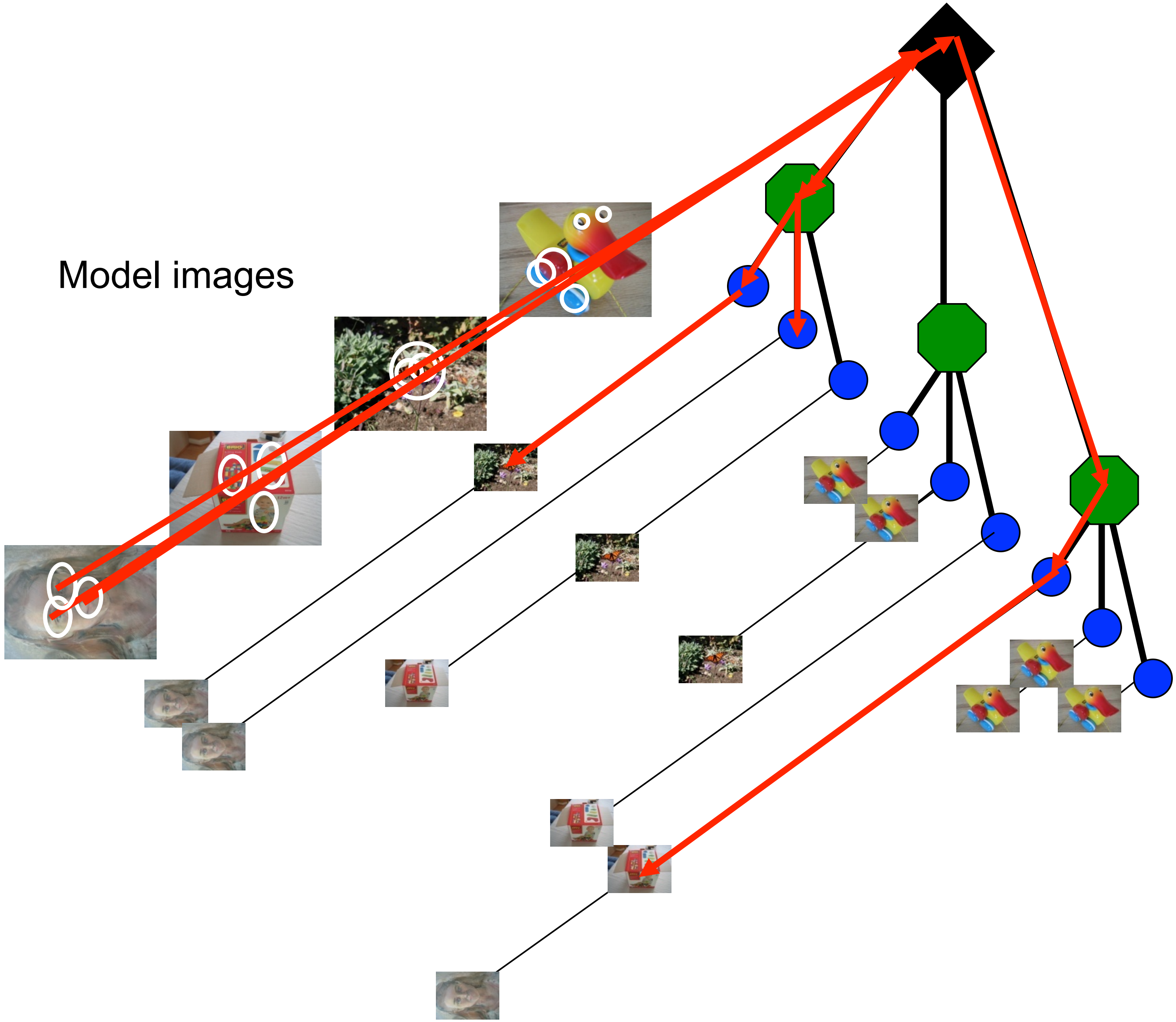
Slide credit: D. Nister

Model images



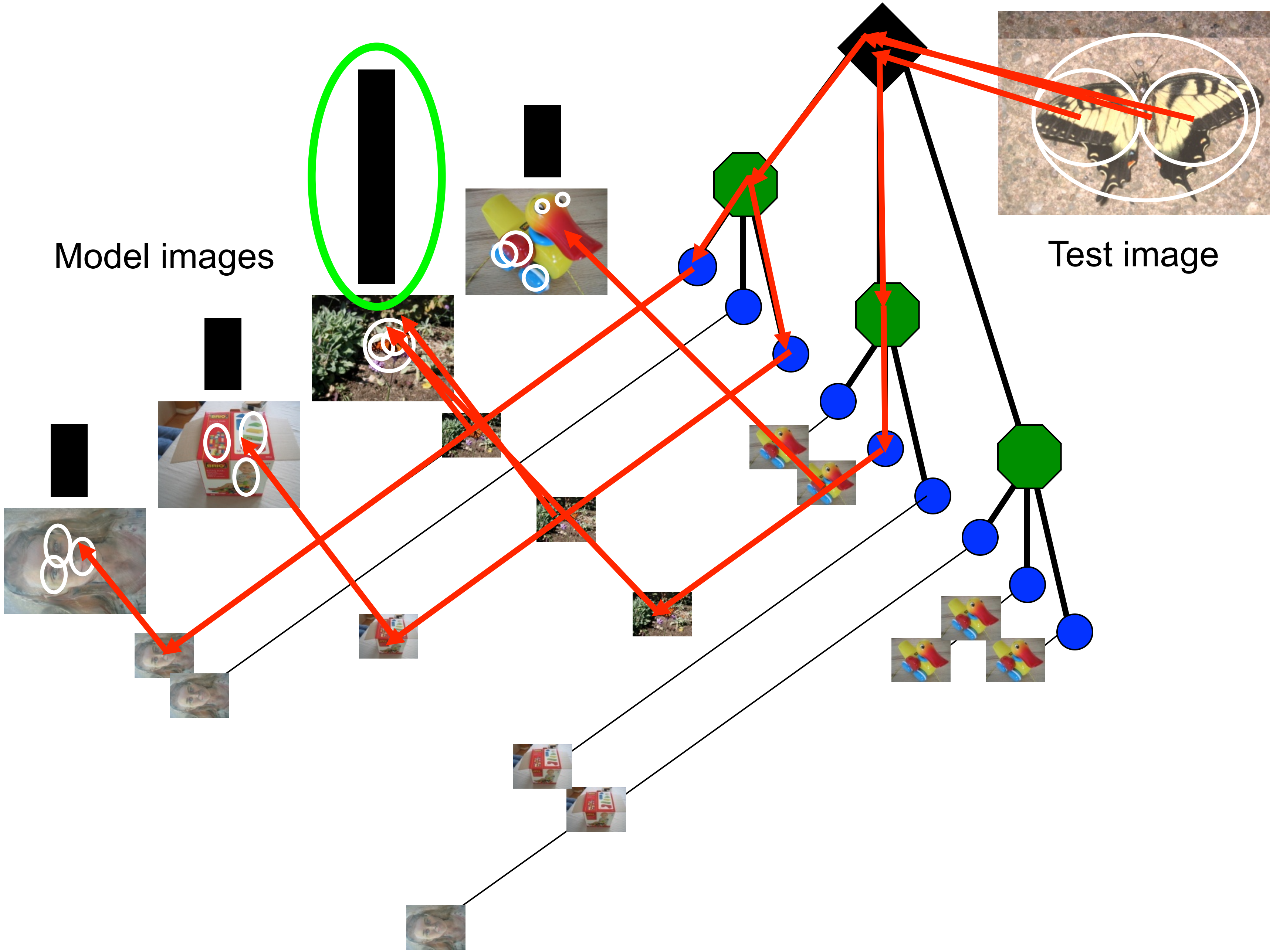
Populating the vocabulary tree/inverted index

Slide credit: D. Nister



Populating the vocabulary tree/inverted index

Slide credit: D. Nister



Looking up a test image

Slide credit: D. Nister

Inverted file index

Database images

Image #1

Image #2

Image #3

Word #	Image #
1	3
2 …	
7	1, 2
8	3
9	
10 …	
91	2

…

- Database images are loaded into the index mapping words to image numbers

Kristen Grauman

Inverted file index

New query image

Word #	Image #
1	3
2 …	
7	1, 2
8	3
9	
10 …	
91	2

⋮ ⋮

…

- New query image is mapped to indices of database images that share a word.

Kristen Grauman

Inverted file index

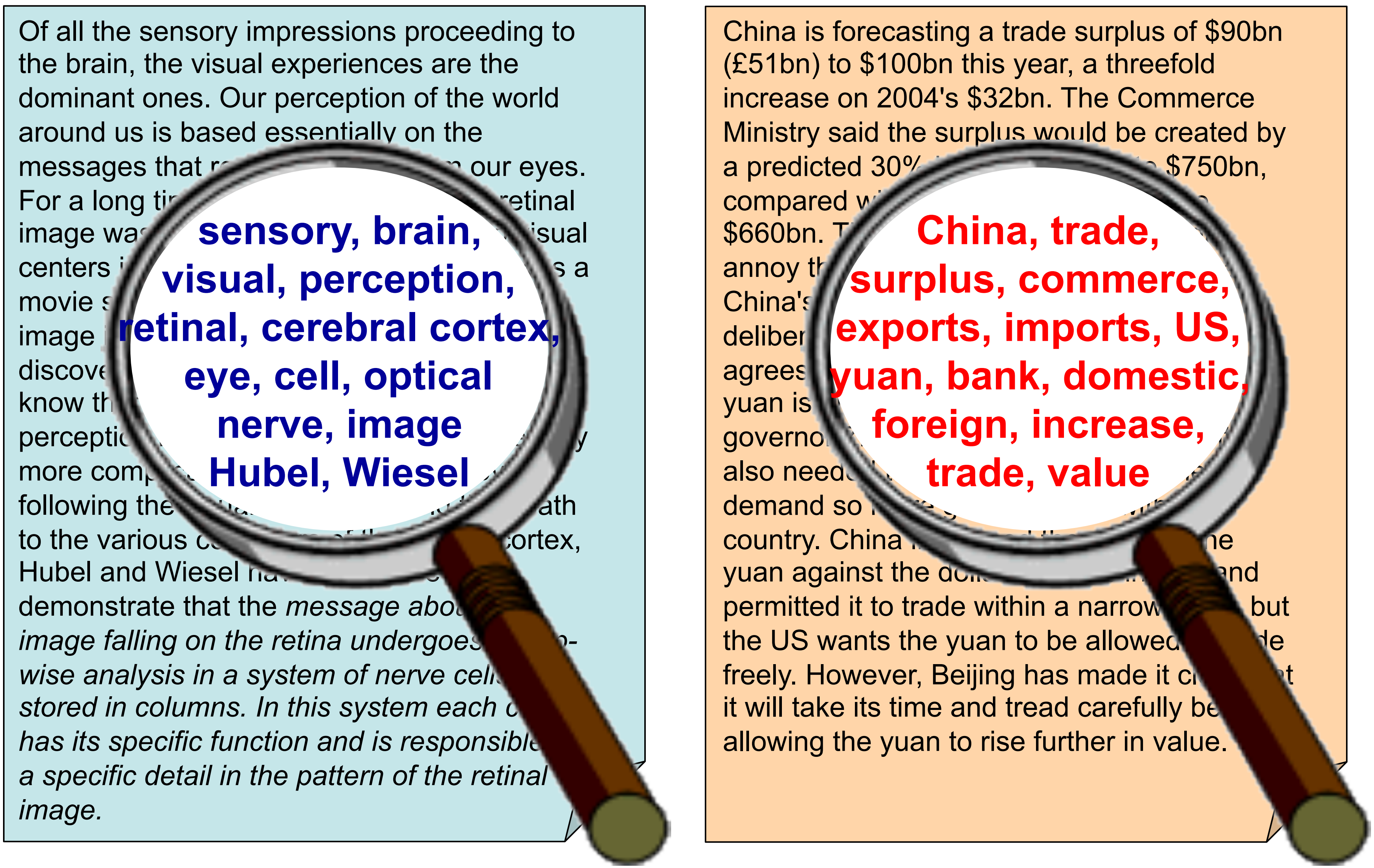
- Key requirement for inverted file index to be efficient: sparsity
- If most pages/images contain most words then you’re not better off than exhaustive search.
  - Exhaustive search would mean comparing the word distribution of a query versus every page.

Instance recognition: remaining issues

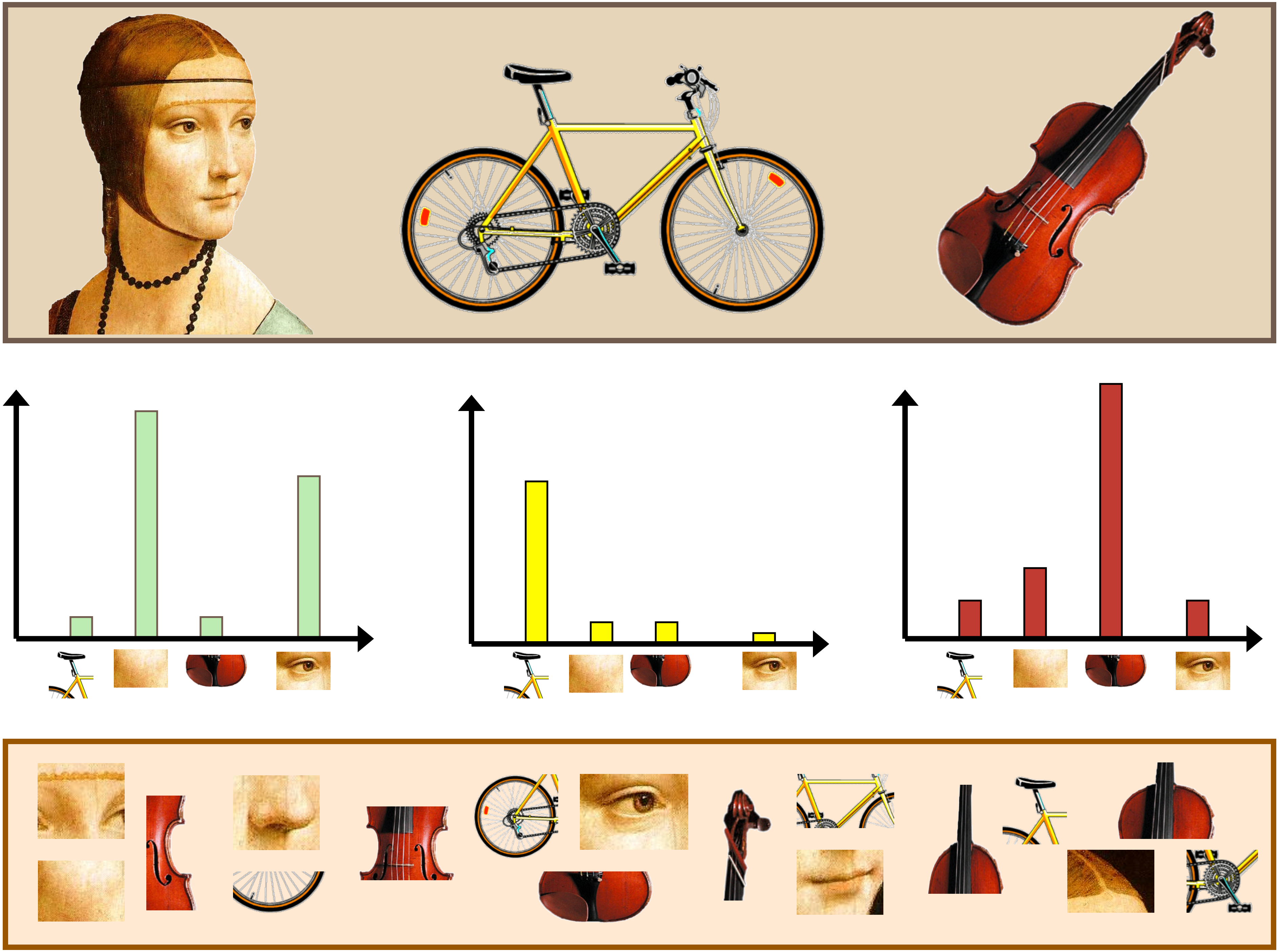
- How to summarize the content of an entire image? And gauge overall similarity?
- How large should the vocabulary be? How to perform quantization efficiently?
- Is having the same set of visual words enough to identify the object/scene? How to verify spatial agreement?
- How to score the retrieval results?

Kristen Grauman

Analogy to documents



ICCV 2005 short course, L. Fei-Fei



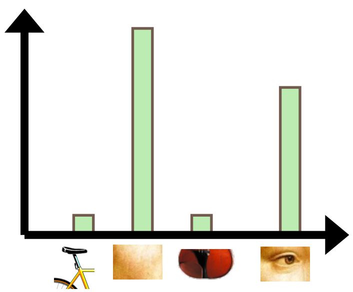
Bags of visual words

- Summarize entire image based on its distribution (histogram) of word occurrences.
- Analogous to bag of words representation commonly used for documents.

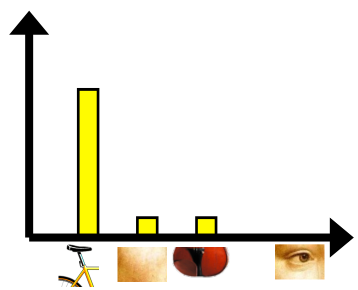
Comparing bags of words

- Rank frames by normalized scalar product between their (possibly weighted) occurrence counts (aka cosine similarity)
- This is a kind of nearest neighbor search for similar images

[1 8 1 4]

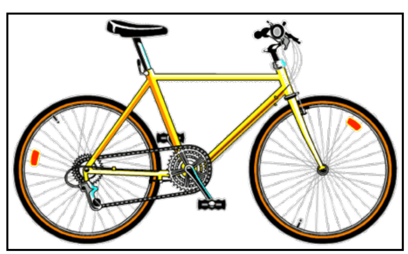


[5 1 1 0]



$$sim(d_j, q) = \frac{\langle d_j, q \rangle}{\|d_j\| \|q\|}$$

$\vec{d}_j$



$\vec{q}$

Kristen Grauman

Inverted file index and bags of words similarity

New query image

Word #	Image #
1	3
2	
…	
7	1, 2
8	3
9	
10	
…	
91	2
⋮	⋮

1. Extract words in query
2. Inverted file index to find relevant frames
3. Compare word counts

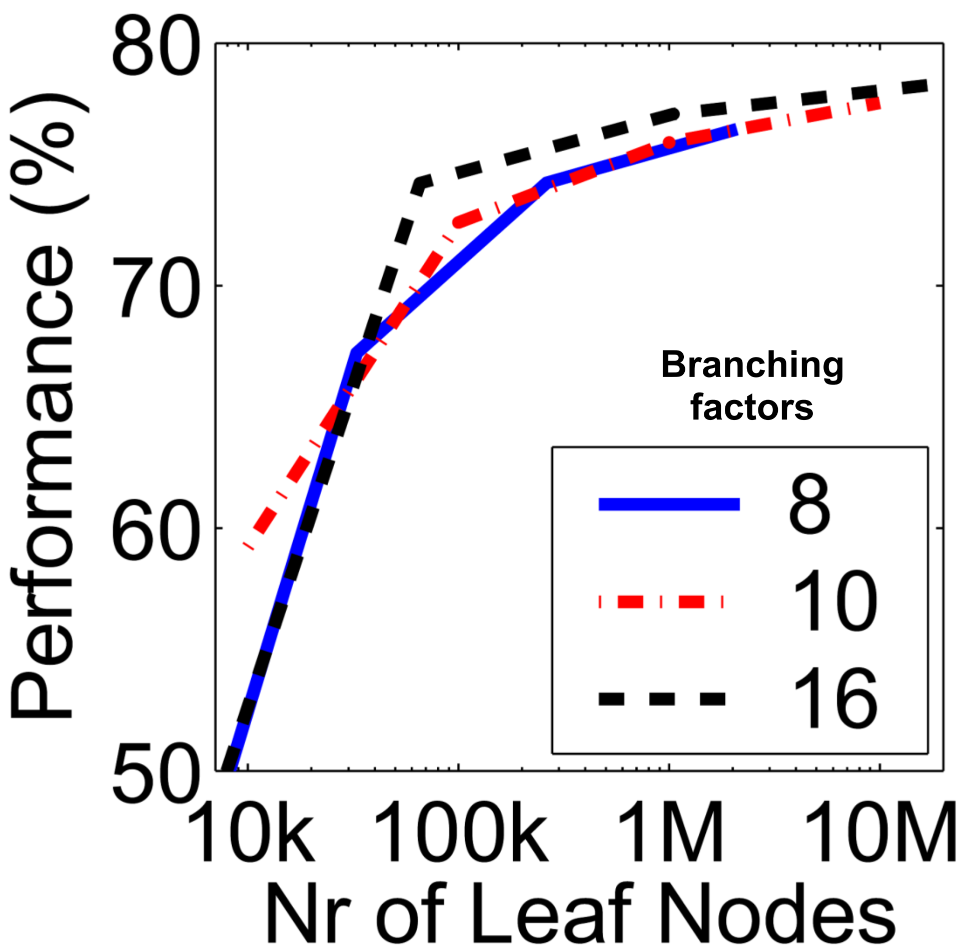
Kristen Grauman

Instance recognition: remaining issues

- How to summarize the content of an entire image? And gauge overall similarity?
- How large should the vocabulary be? How to perform quantization efficiently?
- Is having the same set of visual words enough to identify the object/scene? How to verify spatial agreement?
- How to score the retrieval results?

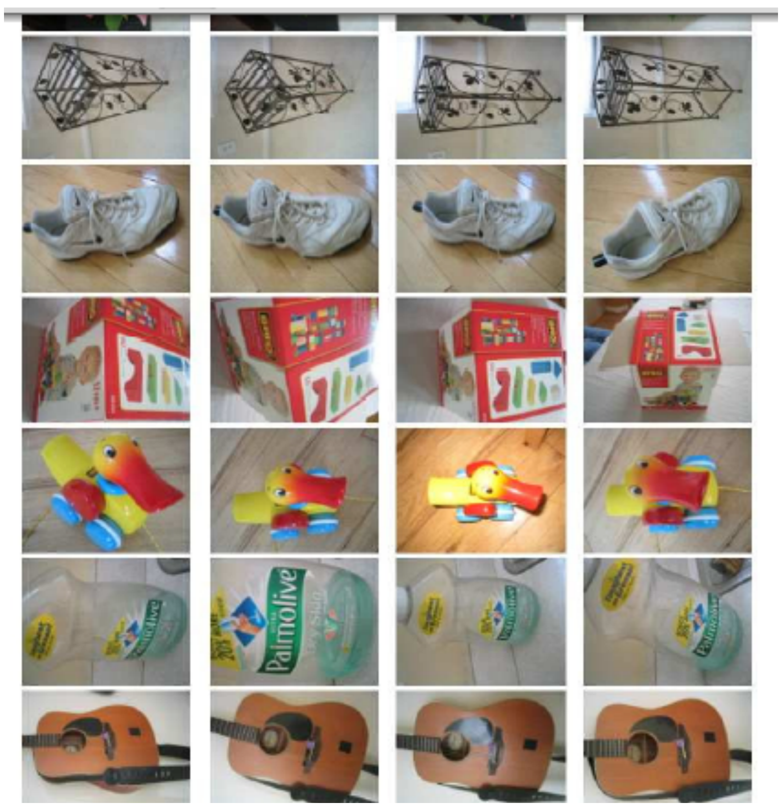
Kristen Grauman

Vocabulary size



Influence on performance, sparsity

Results for recognition task with 6347 images



Nister & Stewenius, CVPR 2006

Kristen Grauman

Vocabulary trees: complexity

Number of words given tree parameters: branching factor and number of levels

branching_factor^number_of_levels

Word assignment cost vs. flat vocabulary

O(k) for flat

$O(\log_{branching\_factor}(k) * branching\_factor)$

Is this like a kd-tree?

Yes, but with better partitioning and defeatist search.

This hierarchical data structure is lossy – you might not find your true nearest cluster.

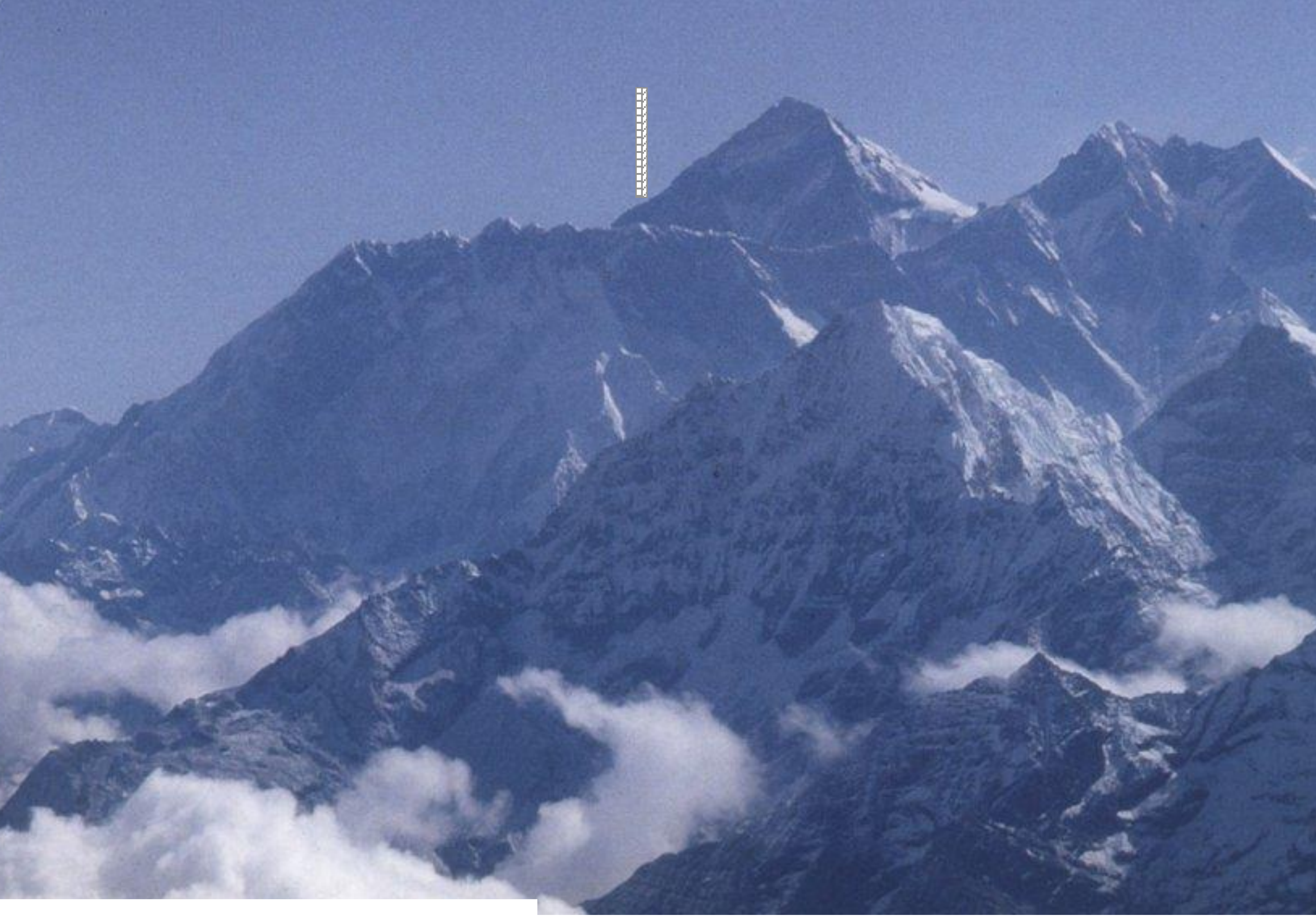
110,000,000 Images in 5.8 Seconds

Slide Credit: Nister

Slide Credit: Nister

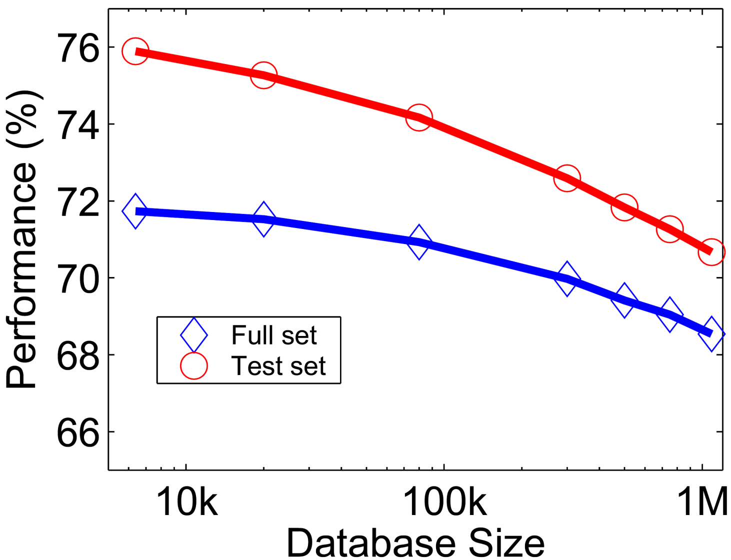
UK Center for Visualization & Virtual Environments

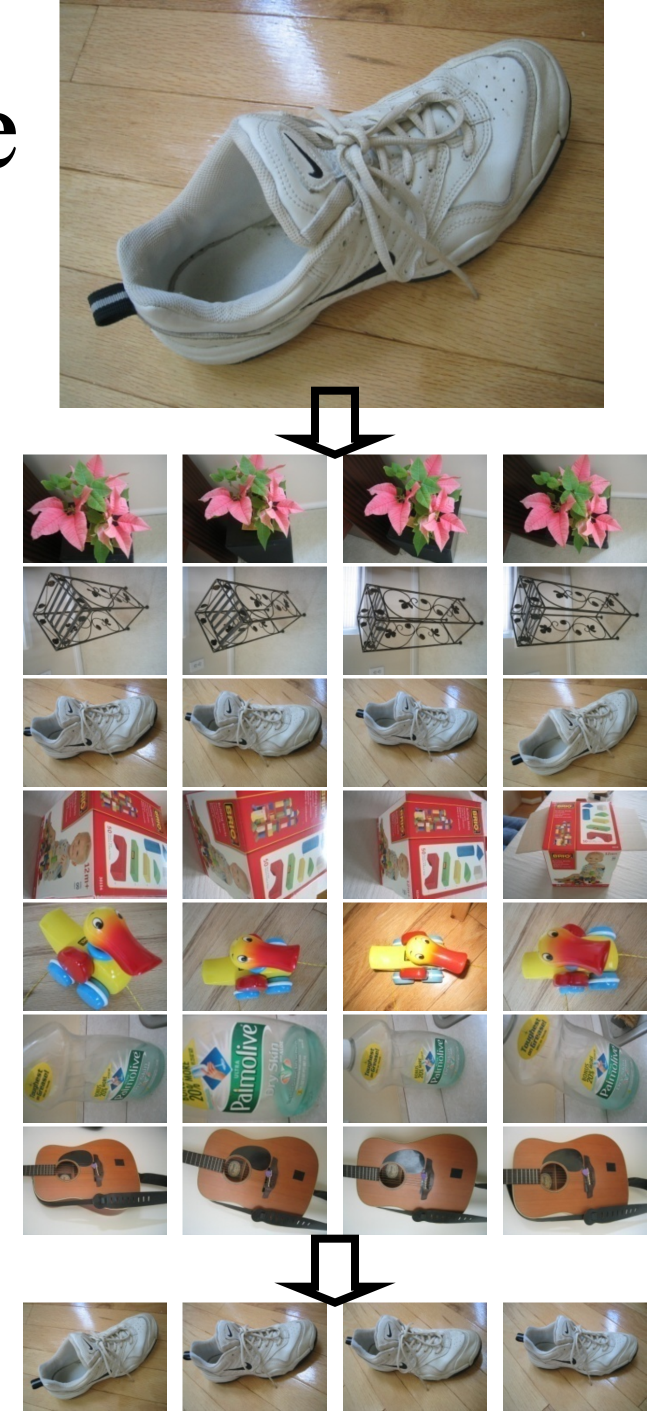
Slide Credit: Nister



Slide Credit: Nister

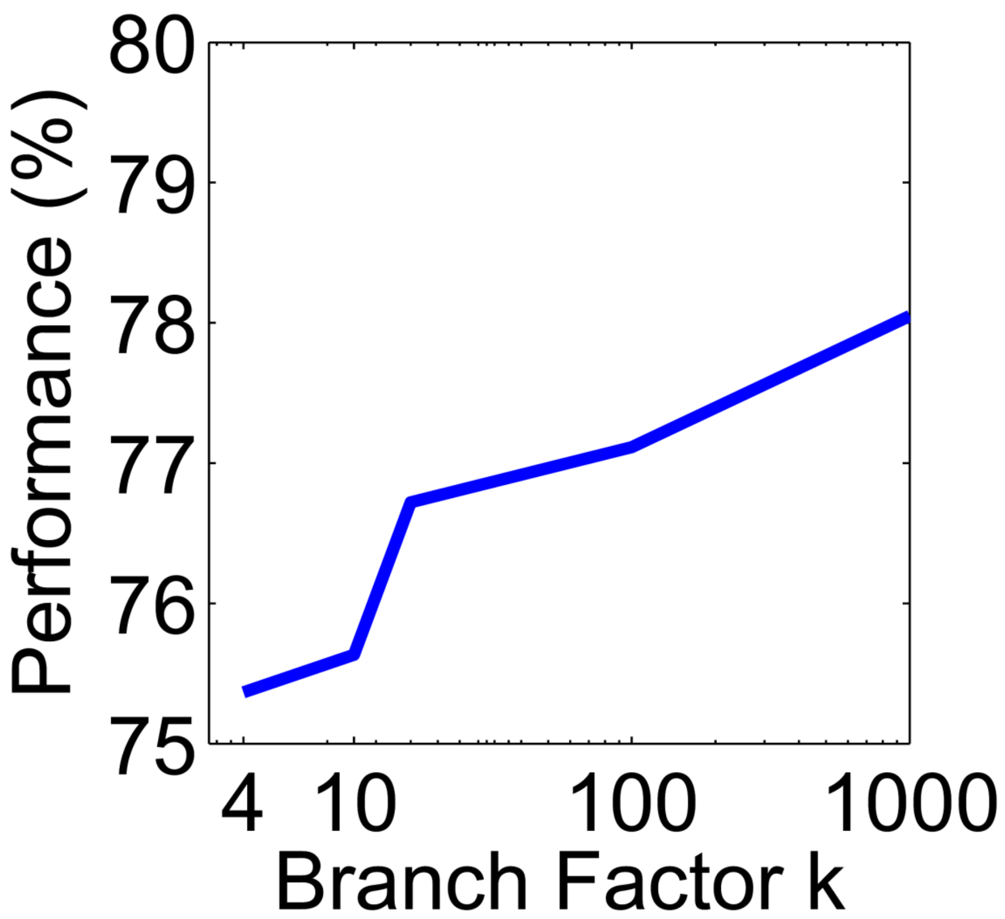
Performance





Center for Visualization & Virtual Environments

Higher branch factor works better (but slower)



Center for Visualization & Virtual Environments

Visual words/bags of words

+ flexible to geometry / deformations / viewpoint
+ compact summary of image content
+ provides fixed dimensional vector representation for sets
+ very good results in practice

- background and foreground mixed when bag covers whole image
- optimal vocabulary formation remains unclear
- basic model ignores geometry – must verify afterwards, or encode via features

Kristen Grauman

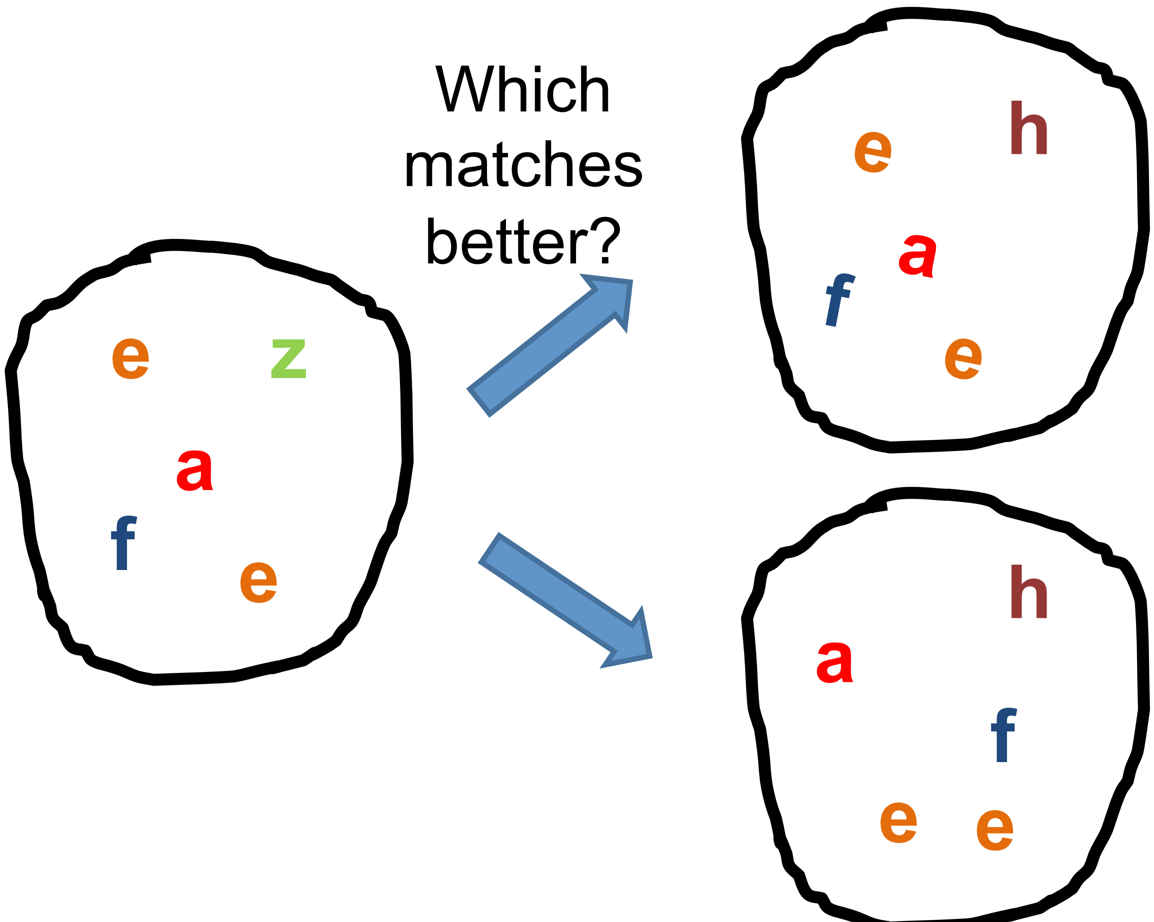
Instance recognition: remaining issues

- How to summarize the content of an entire image? And gauge overall similarity?
- How large should the vocabulary be? How to perform quantization efficiently?
- Is having the same set of visual words enough to identify the object/scene? How to verify spatial agreement?
- How to score the retrieval results?

Kristen Grauman

Can we be more accurate?

So far, we treat each image as containing a "bag of words", with no spatial information



Can we be more accurate?

So far, we treat each image as containing a "bag of words", with no spatial information

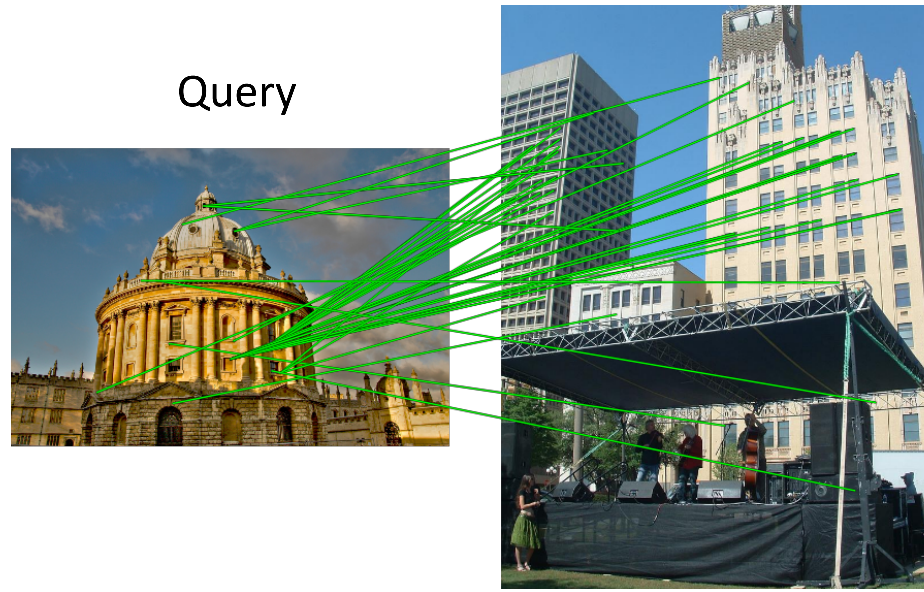
Real objects have consistent geometry

Spatial Verification

Query

DB image with high BoW similarity

Query



DB image with high BoW similarity

Both image pairs have many visual words in common.

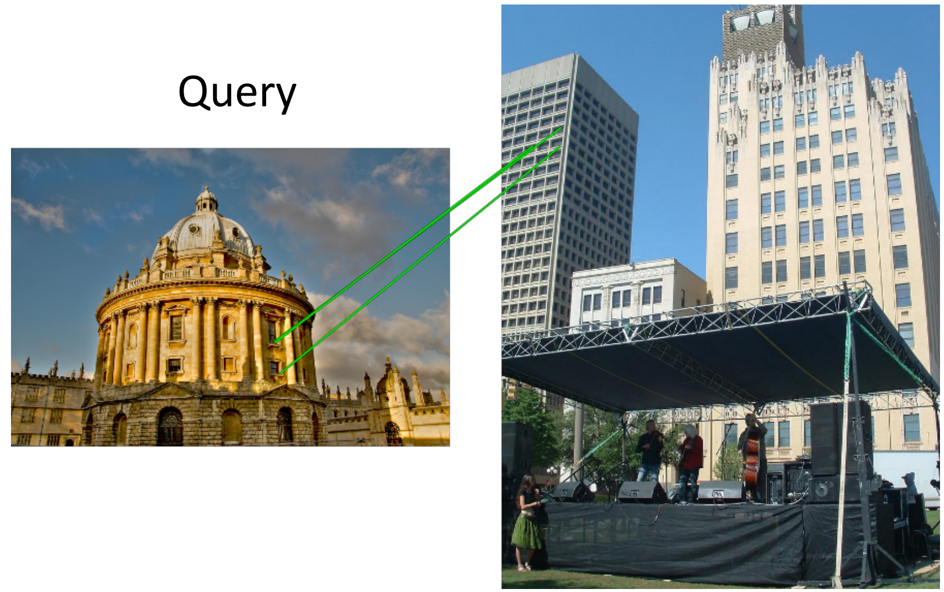
Slide credit: Ondrej Chum

Spatial Verification

Query

DB image with high BoW similarity

Query



DB image with high BoW similarity

Only some of the matches are mutually consistent

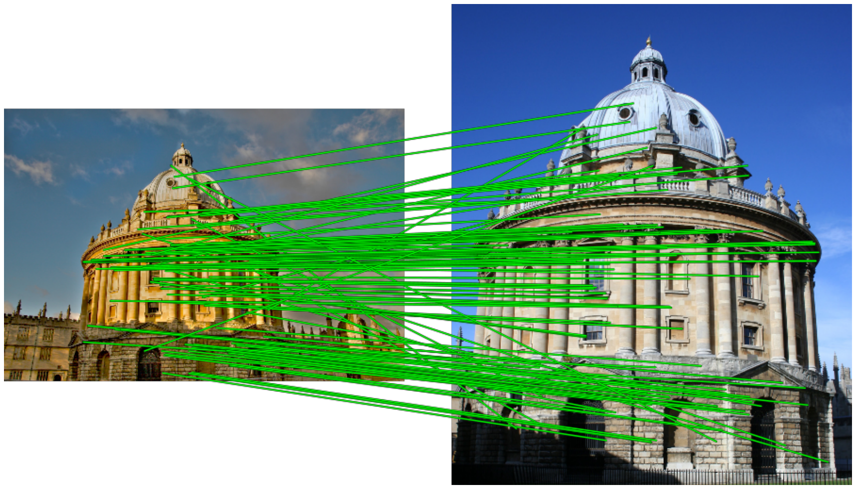
Spatial Verification: two basic strategies

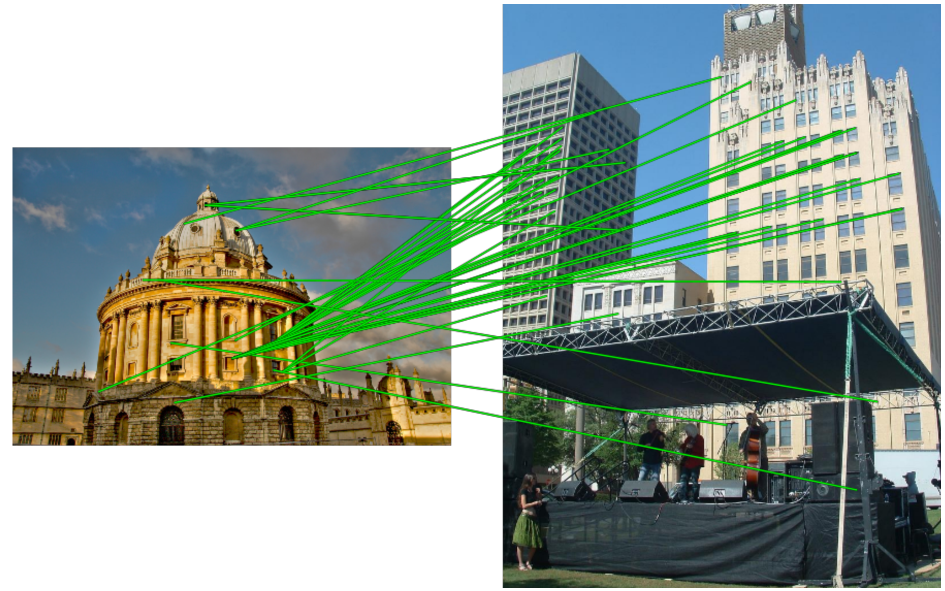
- RANSAC
  - Typically sort by BoW similarity as initial filter
  - Verify by checking support (inliers) for possible transformations
    - e.g., “success” if find a transformation with > N inlier correspondences

- Generalized Hough Transform
  - Let each matched feature cast a vote on location, scale, orientation of the model object
  - Verify parameters with enough votes

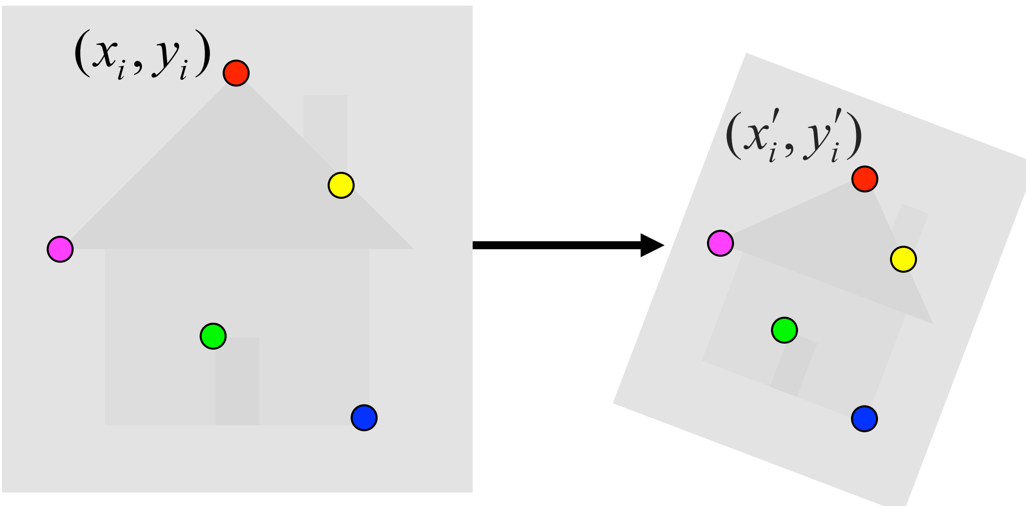
Kristen Grauman

RANSAC verification





Recall: Fitting an affine transformation

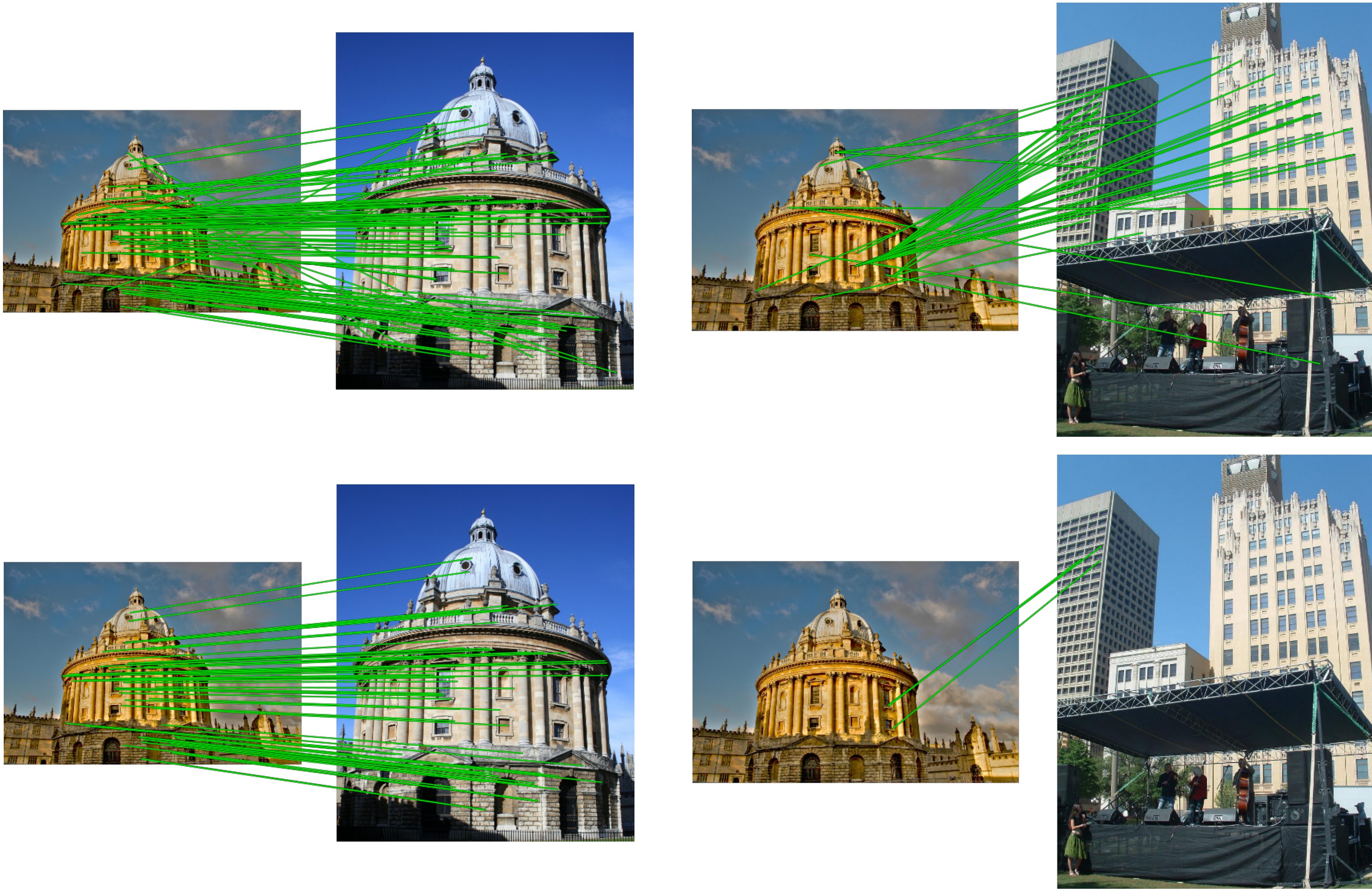


Approximates viewpoint changes for roughly planar objects and roughly orthographic cameras.

$$\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}$$

$$\begin{bmatrix} & & \cdots & & & \\ x_i & y_i & 0 & 0 & 1 & 0 \\ 0 & 0 & x_i & y_i & 0 & 1 \\ & & \cdots & & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} \cdots \\ x_i' \\ y_i' \\ \cdots \end{bmatrix}$$

RANSAC verification



Instance recognition: remaining issues

- How to summarize the content of an entire image? And gauge overall similarity?
- How large should the vocabulary be? How to perform quantization efficiently?
- Is having the same set of visual words enough to identify the object/scene? How to verify spatial agreement?
- How to score the retrieval results?

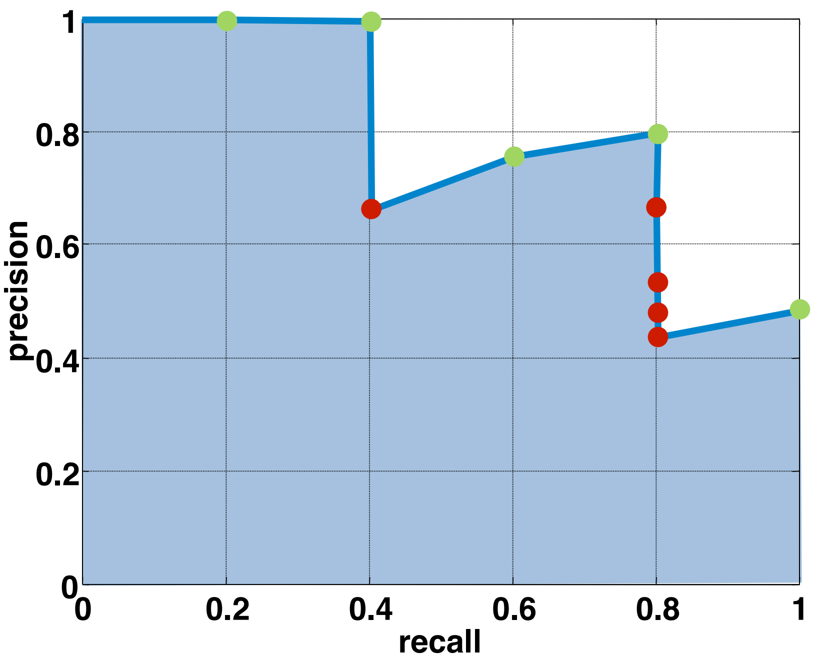
Kristen Grauman

Scoring retrieval quality

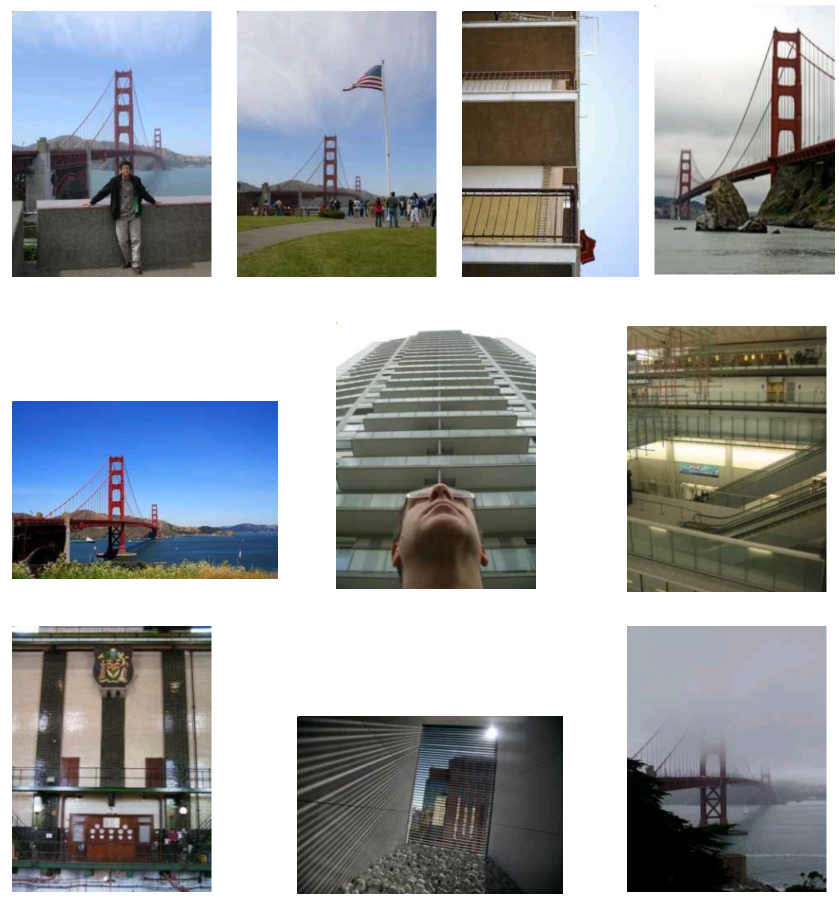
Query

Database size: 10 images
Relevant (total): 5 images

precision = #relevant / #returned
recall = #relevant / #total relevant

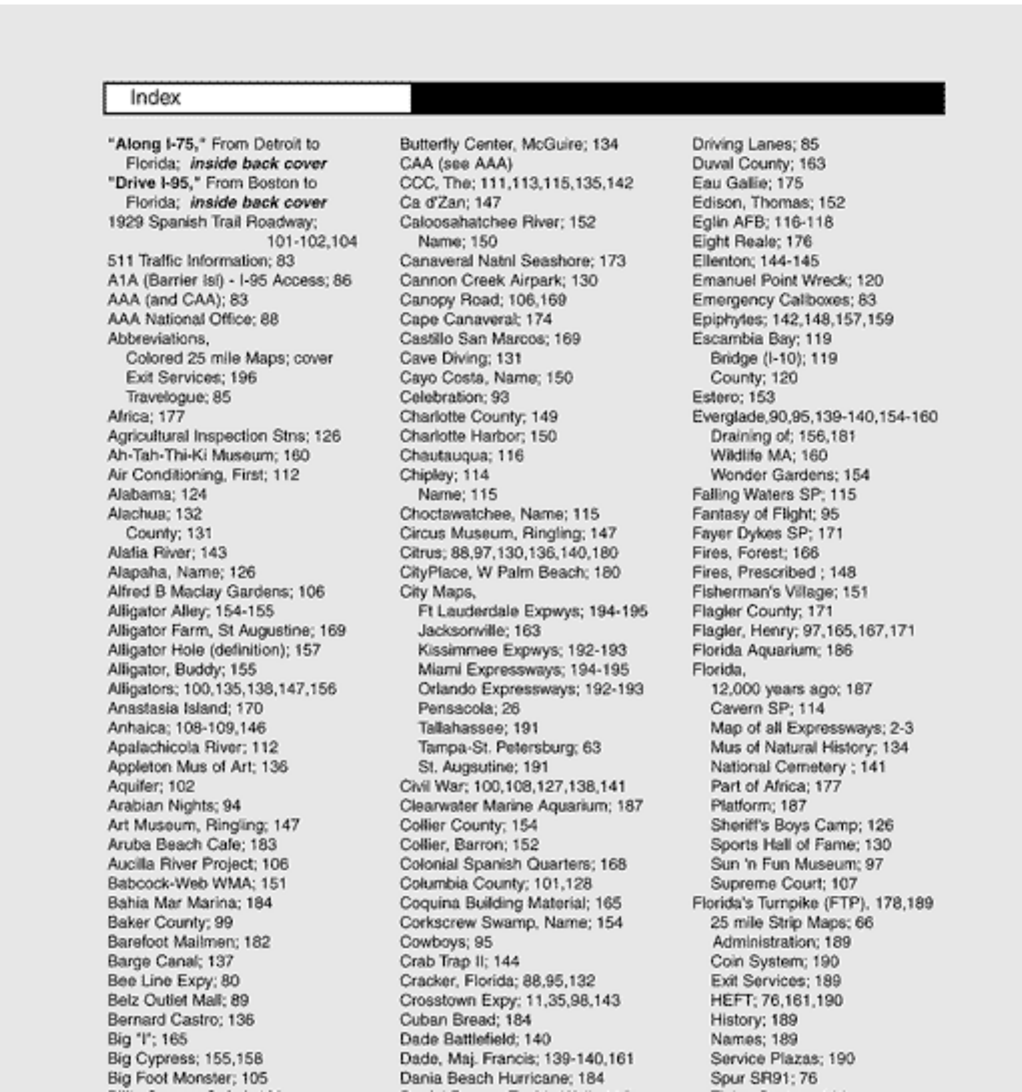


Results (ordered):



Slide credit: Ondrej Chum

What else can we borrow from text retrieval?



Index

tf-idf weighting

- **T**erm **f**requency – **i**nverse **d**ocument **f**requency
- Describe frame by frequency of each word within it, downweight words that appear often in the database
- (Standard weighting for text retrieval)

Number of occurrences of word i in document d

Number of words in document d

$$t_i = \frac{n_{id}}{n_d} \log \frac{N}{n_i}$$

Total number of documents in database

Number of documents word i occurs in, in whole database

Kristen Grauman

Recognition via alignment

Pros:

- Effective when we are able to find reliable features within clutter
- Great results for matching specific instances

Cons:

- Spatial verification as post-processing – not seamless, expensive for large-scale problems
- Not suited for category recognition.

Kristen Grauman

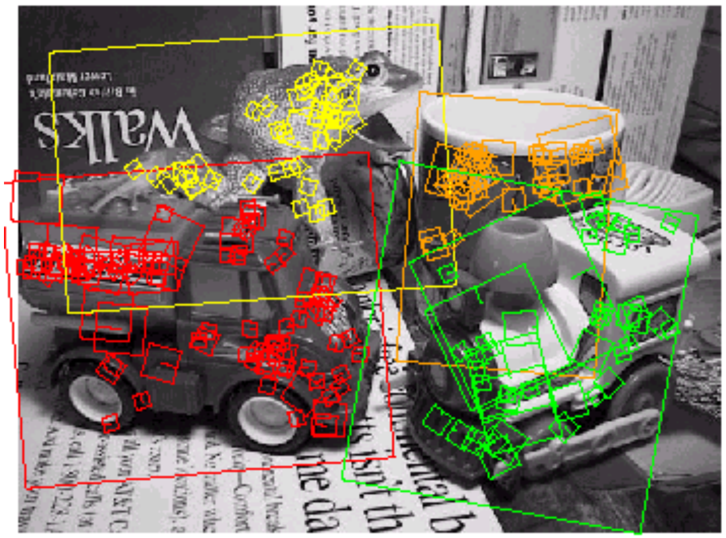
Summary

- **Matching local invariant features**
  - Useful not only to provide matches for multi-view geometry, but also to find objects and scenes.
- **Bag of words** representation: quantize feature space to make discrete set of visual words
  - Summarize image by distribution of words
  - Index individual words
- **Inverted index**: pre-compute index to enable faster search at query time
- **Recognition of instances via alignment:** matching local features followed by spatial verification
  - Robust fitting : RANSAC, GHT

Kristen Grauman

Things to remember

- Object instance recognition
  - Find keypoints, compute descriptors
  - Match descriptors
  - Vote for / fit affine parameters
  - Return object if # inliers > T



- Keys to efficiency
  - Visual words
    - Used for many applications
  - Inverse document file
    - Used for web-scale search