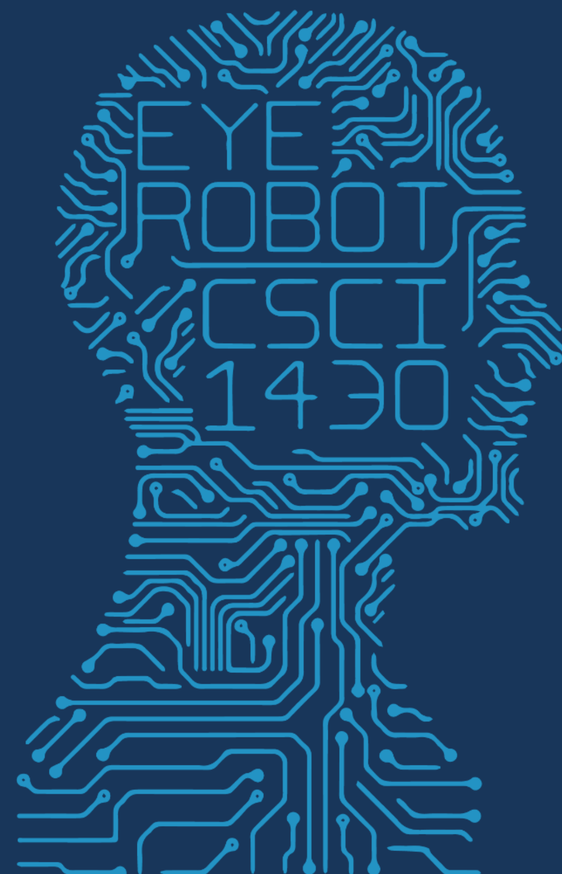
1950

Future Vision



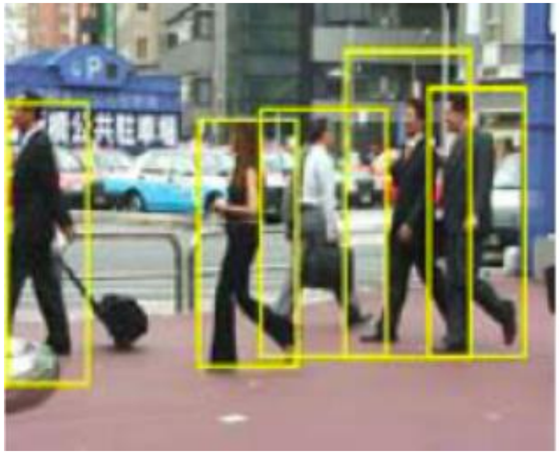
2017 MWF 1pm

Computer Vision

Category vs. instance recognition

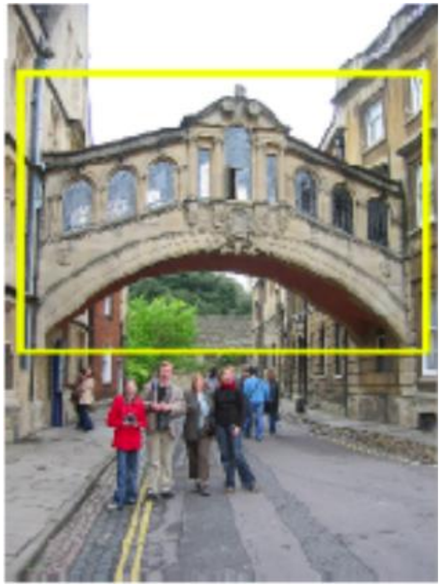
Category:

- Find all the people
- Find all the buildings
- Often within a single image
- Often ‘sliding window’

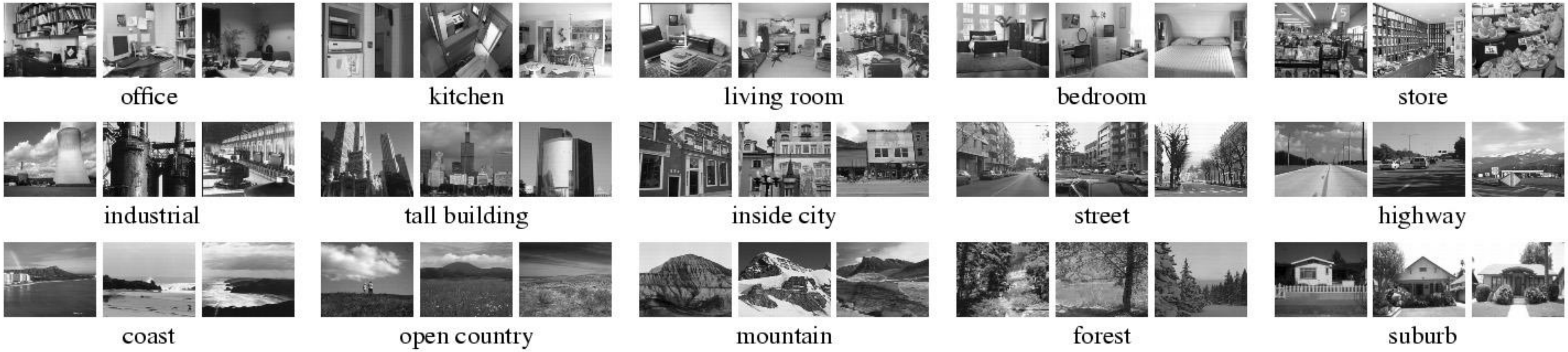


Instance:

- Is this face James?
- Find this specific famous building
- Often within a database of images

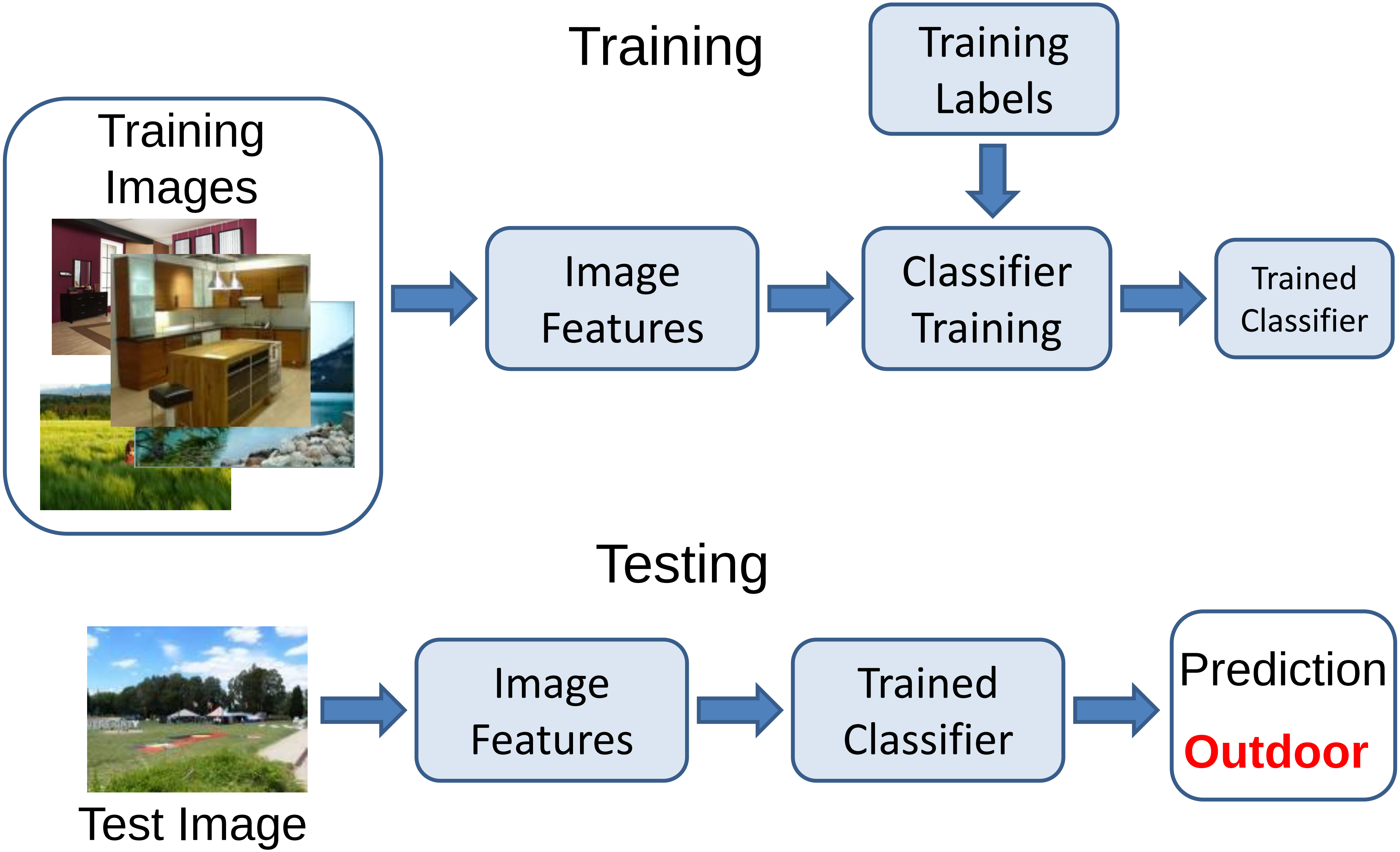


Scene recognition dataset



Instance or category?

Recognition



Recognition Issues

How to summarize the content of an entire image?
How to gauge overall similarity?

How large should the vocabulary be?
How to perform quantization efficiently?

How to score the retrieval results?

How might we add more spatial verification?

Kristen Grauman

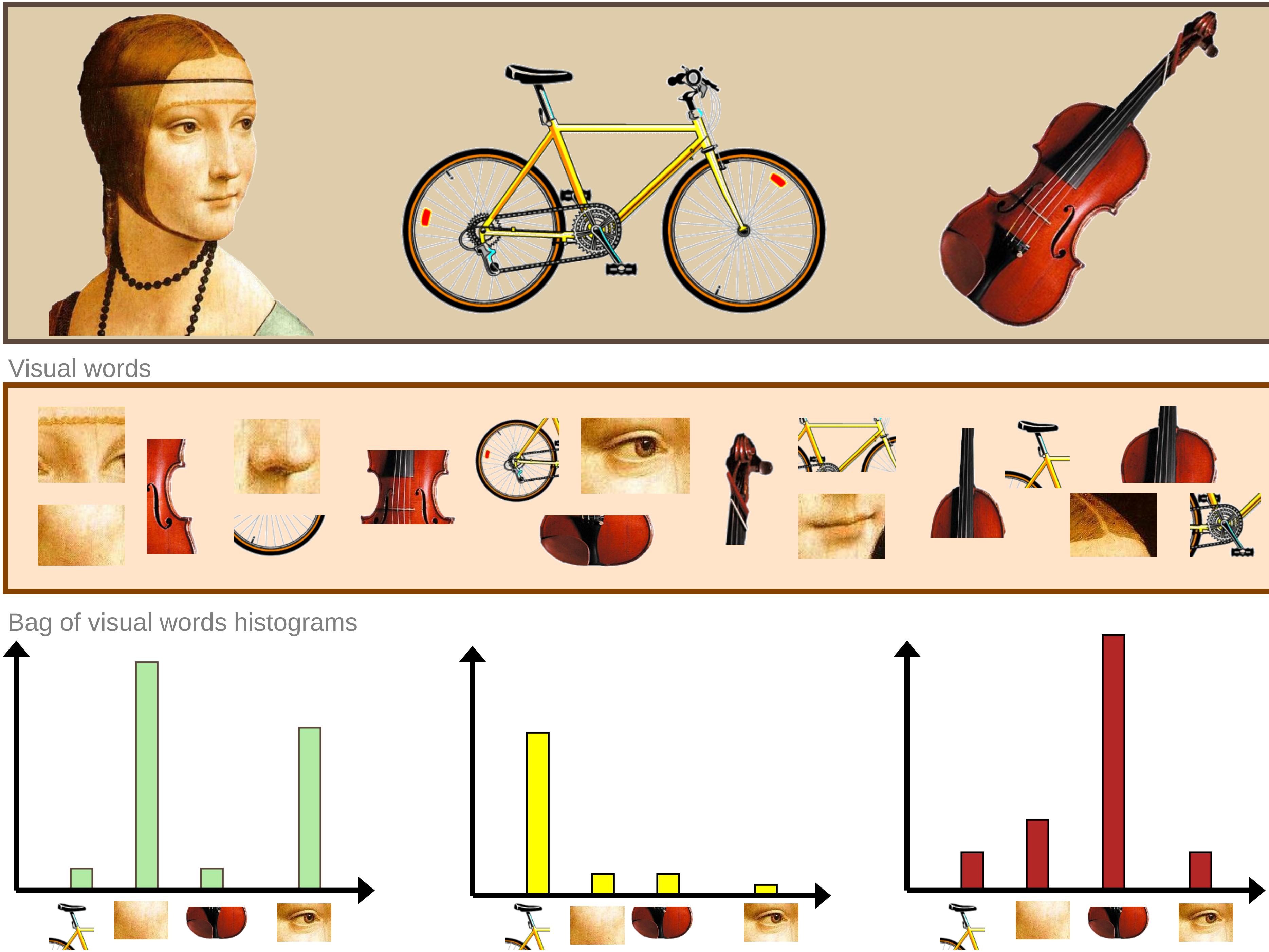
Recognition Issues

How to summarize the content of an entire image?
How to gauge overall similarity?

How large should the vocabulary be?
How to perform quantization efficiently?

How to score the retrieval results?

How might we add more spatial verification?

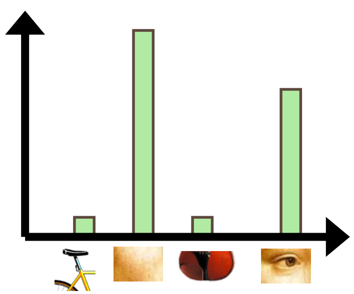

Visual words
Bag of visual words histograms

Comparing bags of words

Compute cosine similarity (normalized scalar (dot) product) between their occurrence counts, then rank and pick smallest. *Nearest neighbor* search for similar images.

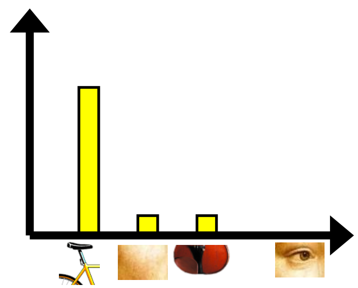
Database image

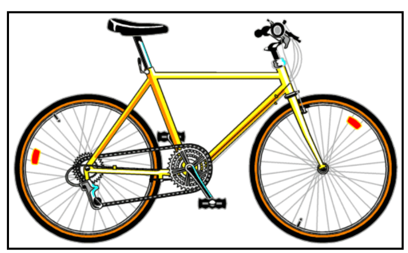
$$\vec{d}_j = [1 \quad 8 \quad 1 \quad 4]$$



Query

$$\vec{q} = [5 \quad 1 \quad 1 \quad 0]$$





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

$$= \frac{\sum_{i=1}^{V} d_j(i) \times q(i)}{\sqrt{\sum_{i=1}^{V} d_j(i)^2} \times \sqrt{\sum_{i=1}^{V} q(i)^2}}$$

for vocabulary of *V* words

Kristen Grauman

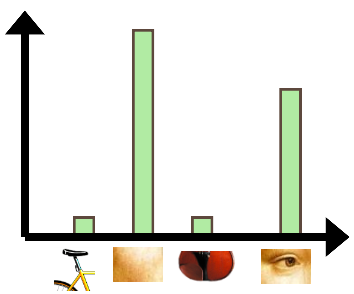
Comparing bags of words

Why might we use cosine similarity here?

What 'intuitive' effect does this provide?

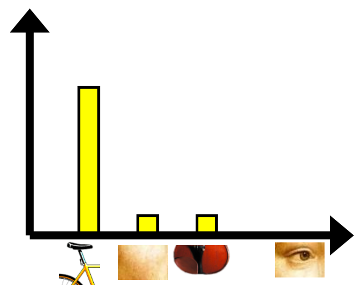
Database image

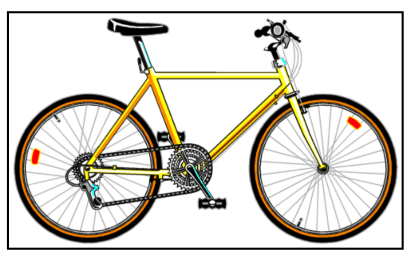
$$\vec{d}_j = [1 \quad 8 \quad 1 \quad 4]$$



Query

$$\vec{q} = [5 \quad 1 \quad 1 \quad 0]$$





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

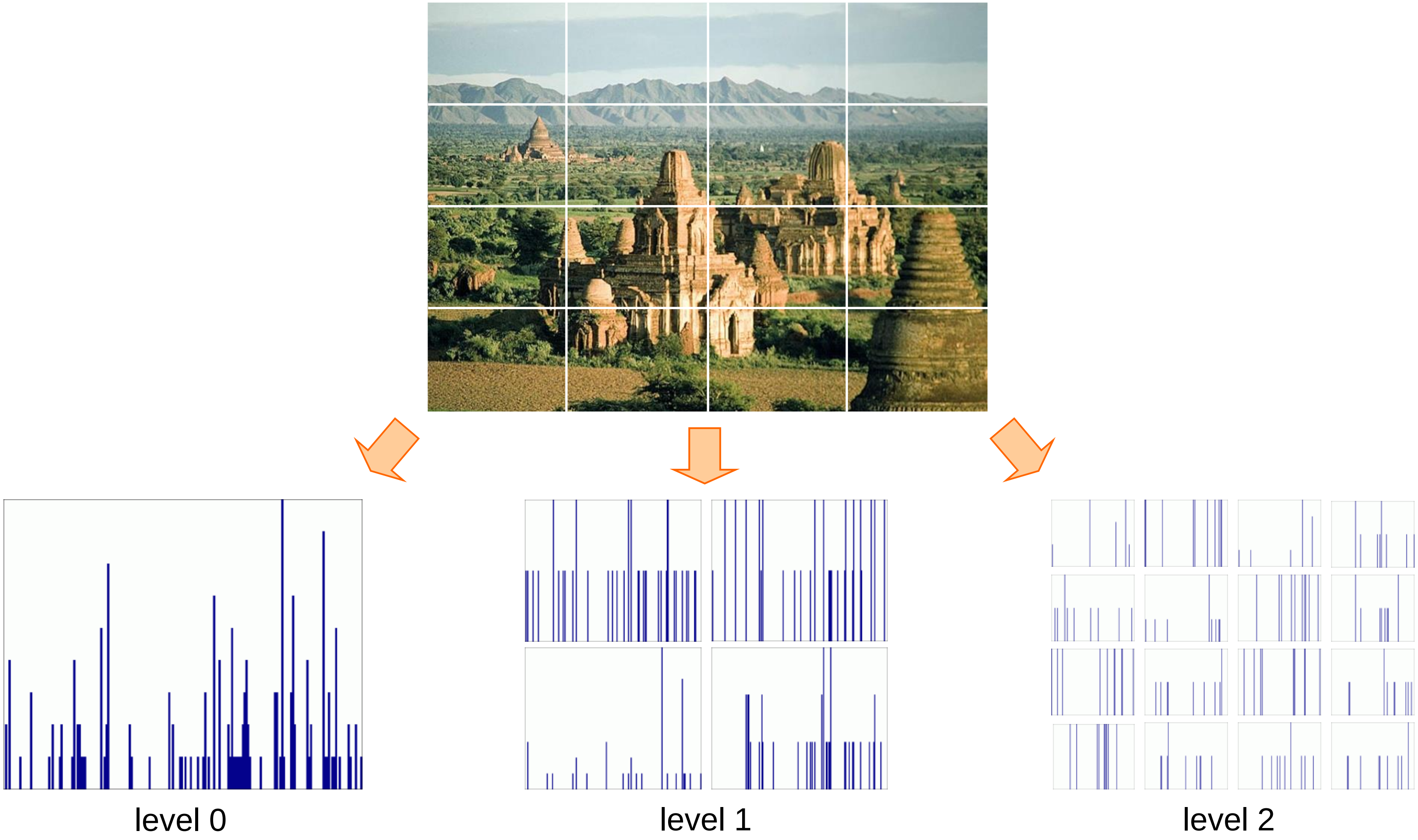
$$= \frac{\sum_{i=1}^{V} d_j(i) \times q(i)}{\sqrt{\sum_{i=1}^{V} d_j(i)^2} \times \sqrt{\sum_{i=1}^{V} q(i)^2}}$$

for vocabulary of *V* words

Kristen Grauman

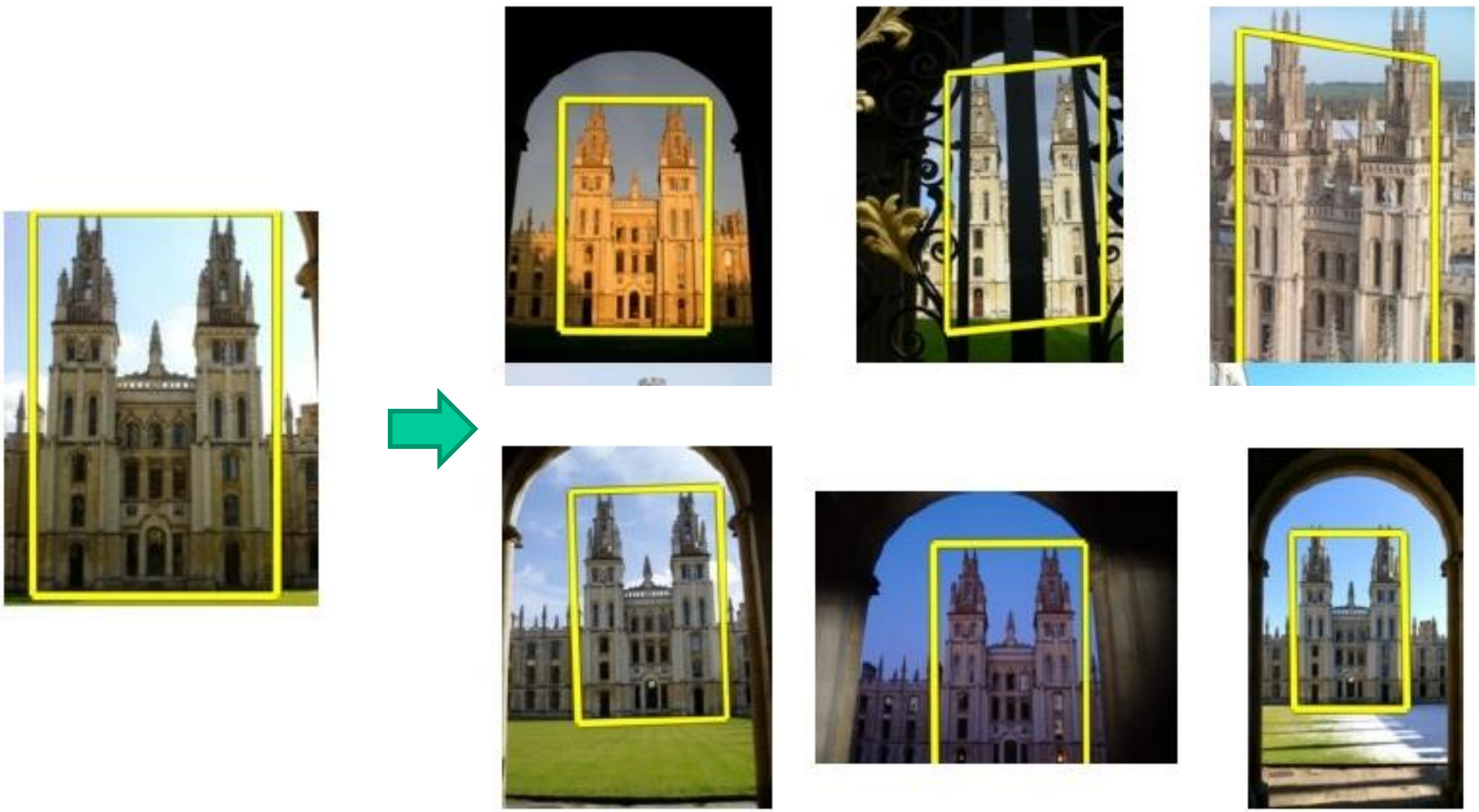
Spatial pyramid representation

- Extension of a bag of features
- Locally orderless representation at several levels of resolution



Lazebnik, Schmid & Ponce (CVPR 2006)

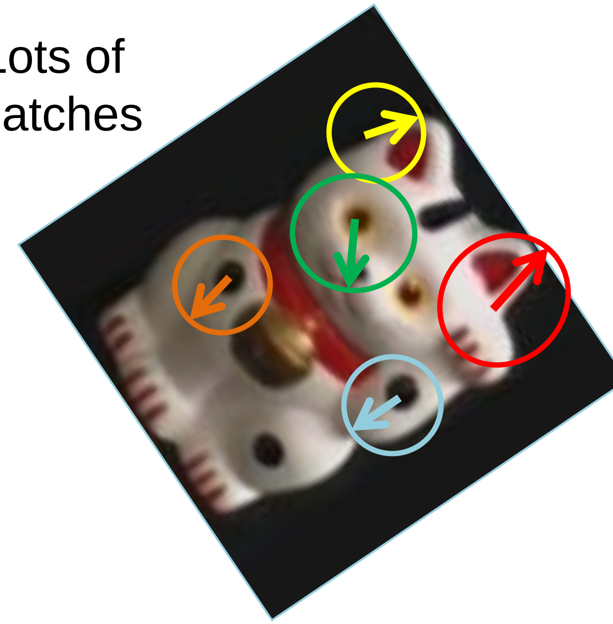
How can we quickly find images in a large database that match a given image region?



Simple idea

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

Lots of Matches

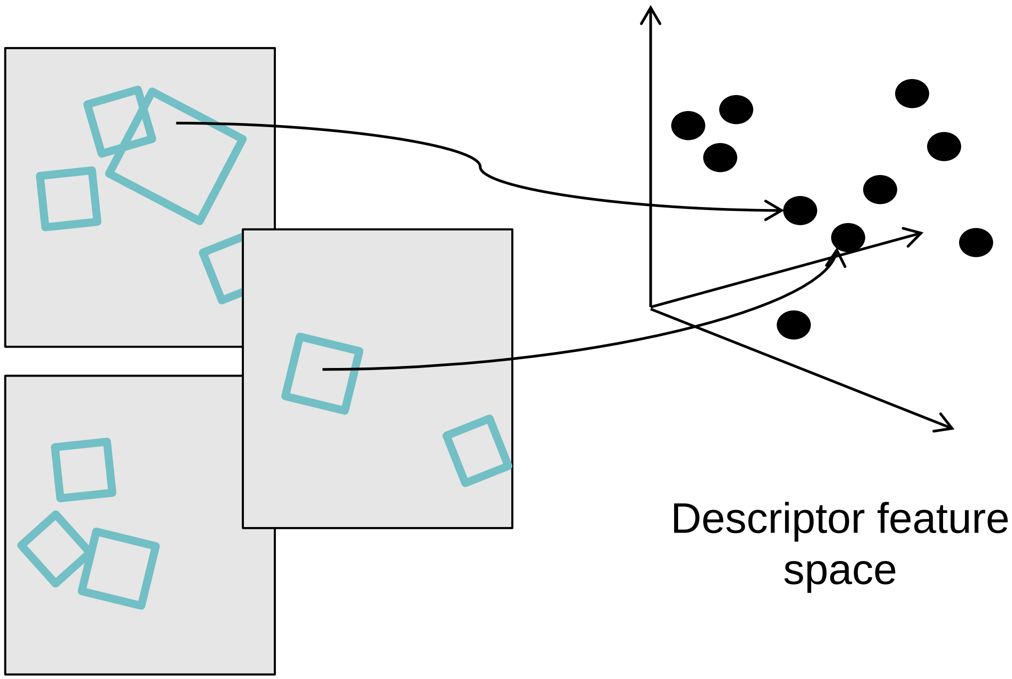


Few or No Matches

But this will be really, really slow!

Indexing local features

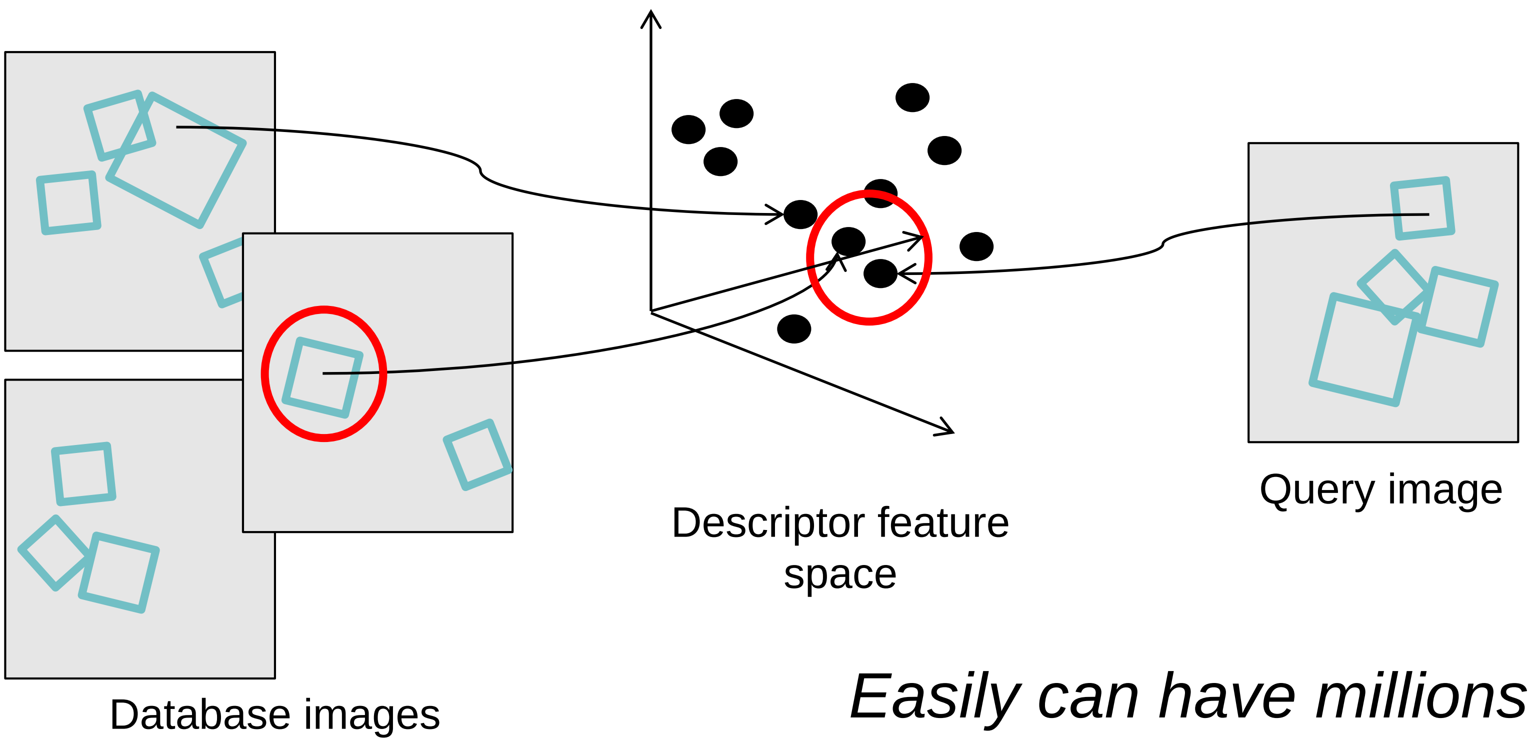
Each patch / region has a descriptor, which is a point in some high-dimensional feature space (e.g., SIFT).



Kristen Grauman

Indexing local features

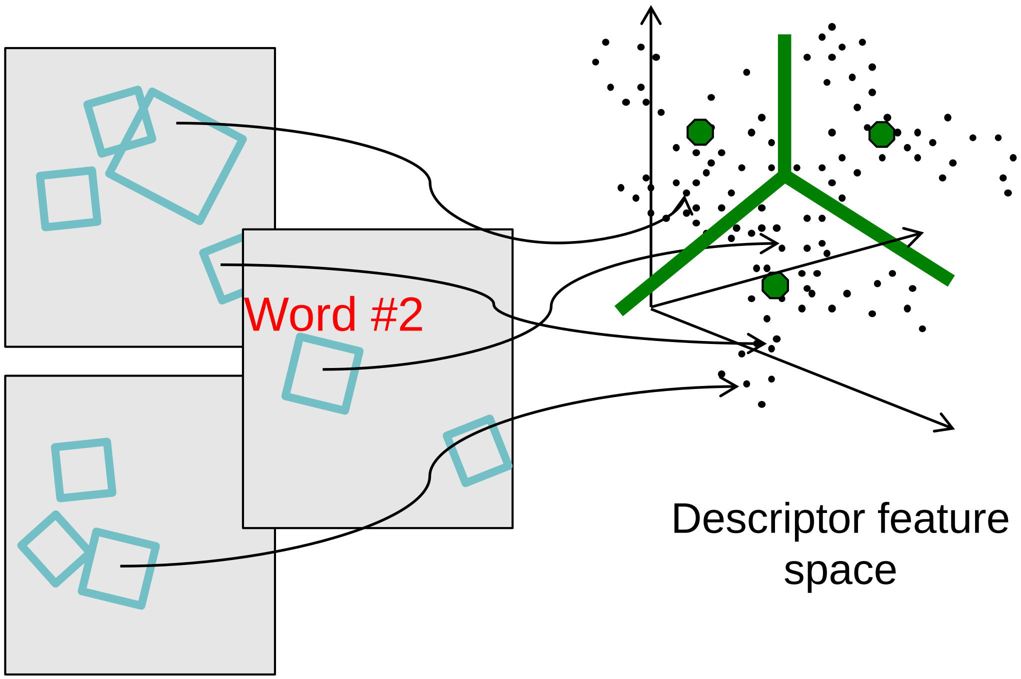
- When we see close points in feature space, we have similar descriptors, which indicates similar local content.



Easily can have millions of features to search!

Visual words

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

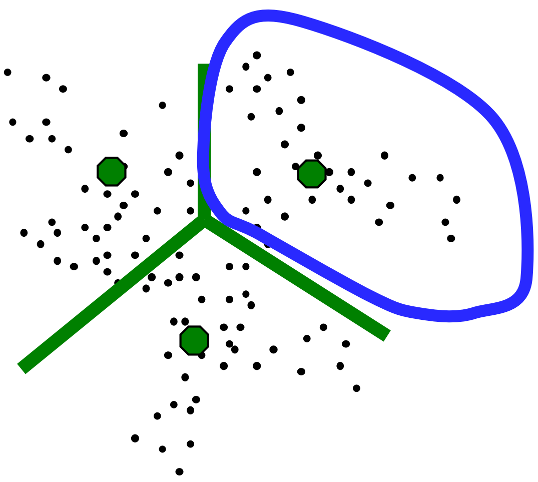


- Quantize via clustering; cluster centers are the visual “words”
- Assign word to each image region by finding the closest cluster center.

Kristen Grauman

Visual words

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



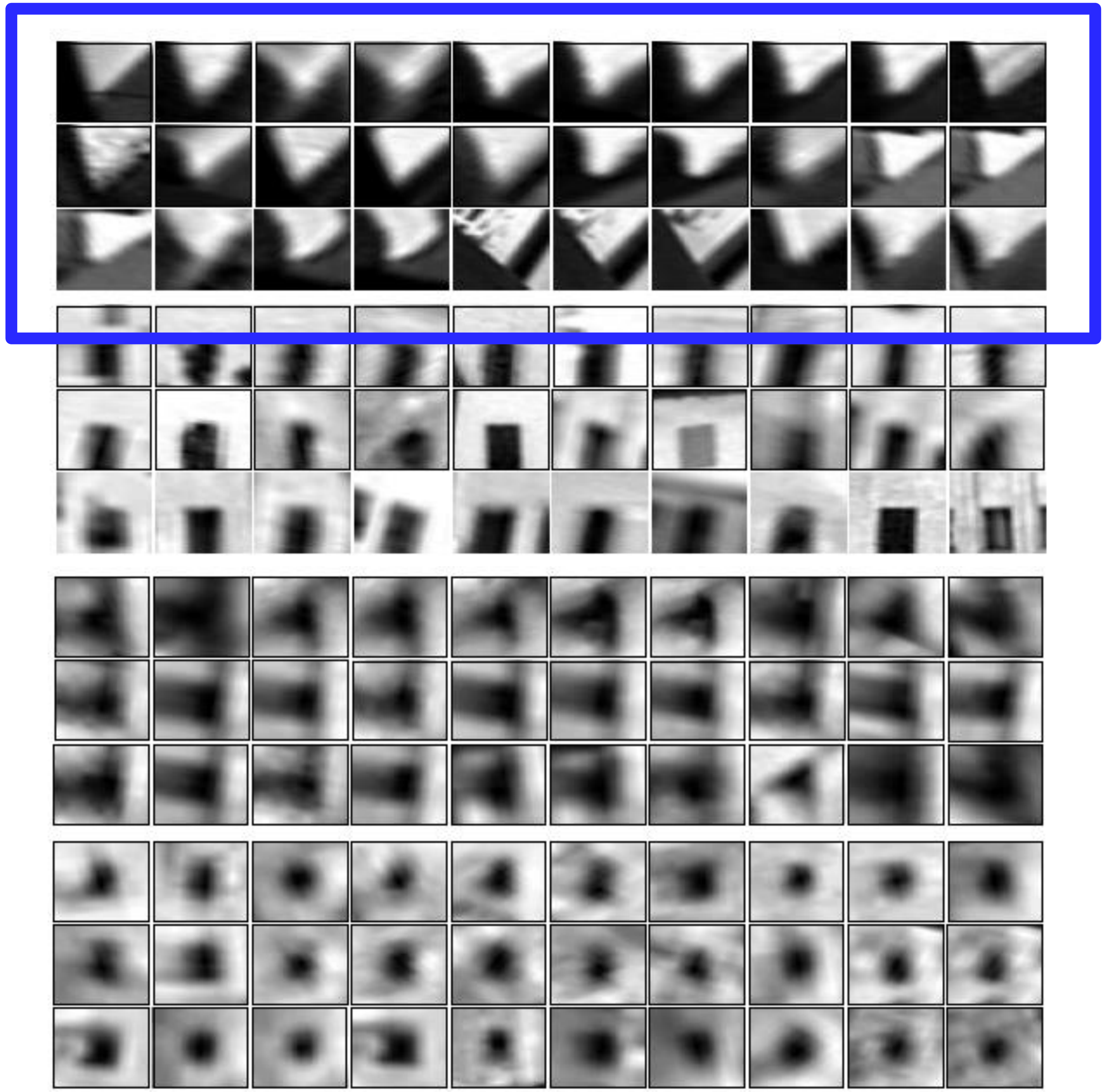
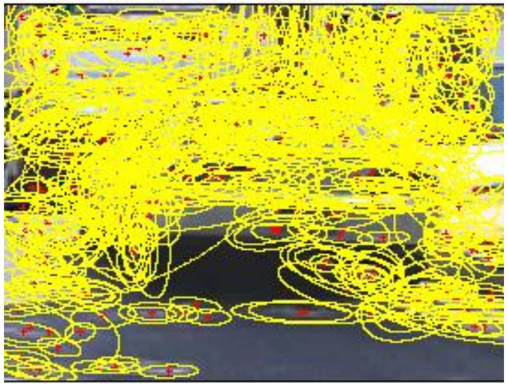


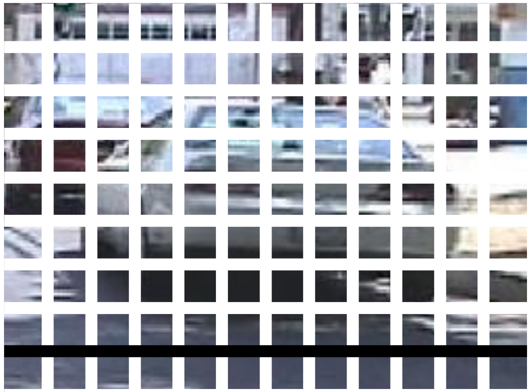
Figure from Sivic & Zisserman, ICCV 2003

Kristen Grauman

Sampling strategies

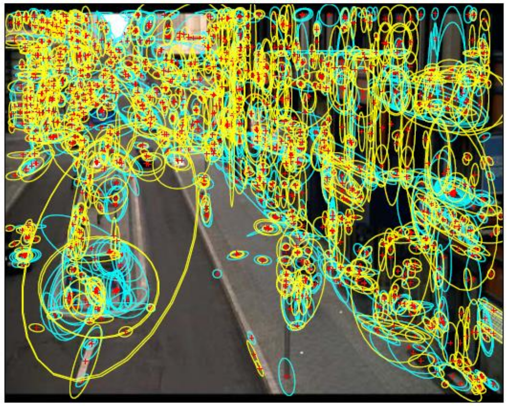


Sparse, at interest points



Dense, uniformly

Randomly



Multiple interest operators

- To find specific textured objects, sparse sampling from interest points often more reliable.
- Multiple complementary interest operators offer more image coverage.
- For object categorization, dense sampling offers better coverage.

[See Nowak, Jurie & Triggs, ECCV 2006]

Image credits: F-F. Li, E. Nowak, J. Sivic

Fast lookup: inverted 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.

Kristen Grauman

Build Inverted Index from Database

Database images

Image #1

Image #2

Image #3

⋮

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

…

Kristen Grauman

Query Inverted Index

New query image

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

…

Candidate matches

Image #1

Image #2

Kristen Grauman

Query Inverted Index

New query image

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

…

Candidate matches

Image #1

Image #2

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

Kristen Grauman

Inverted index

Key requirement: *sparsity*.

If most images contain most words, then we’re not better off than exhaustive search.

- Exhaustive search would mean comparing the visual word distribution of a query versus every page.

Recognition 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?

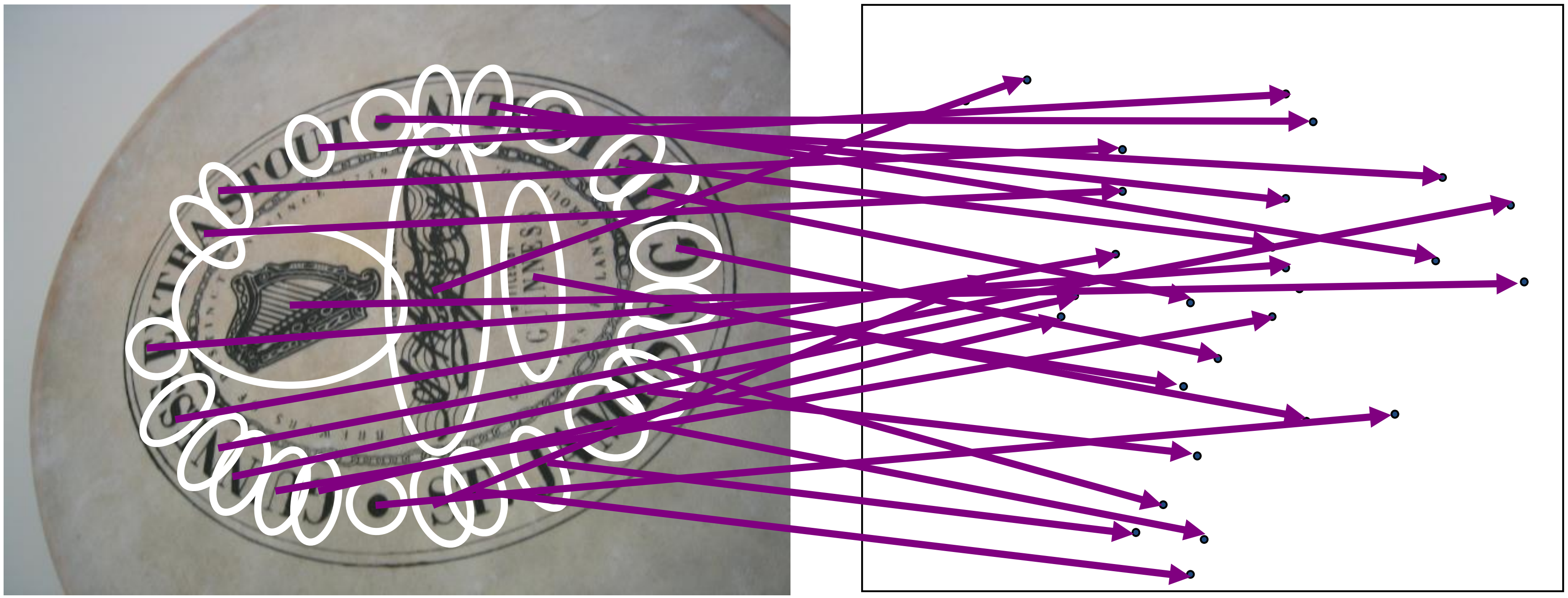
How to score the retrieval results?

How might we add more spatial verification?

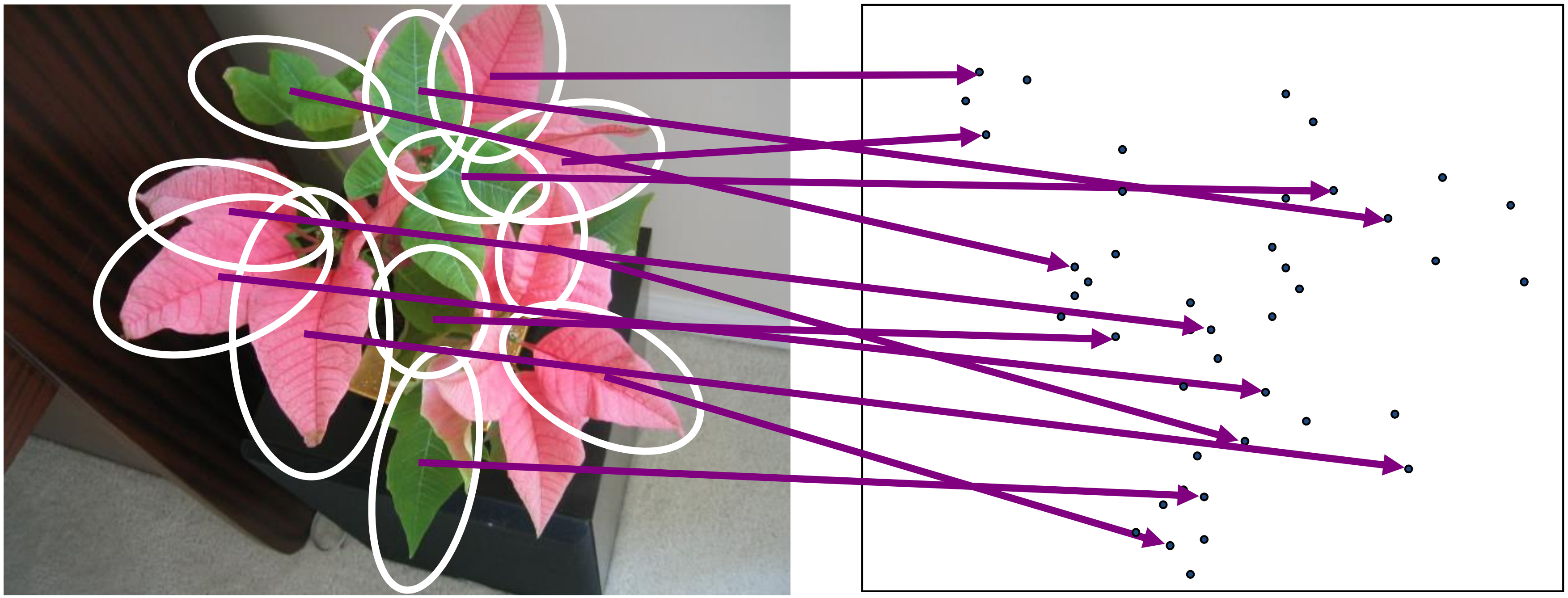
Following slides by David Nister (CVPR 2006)

Kristen Grauman

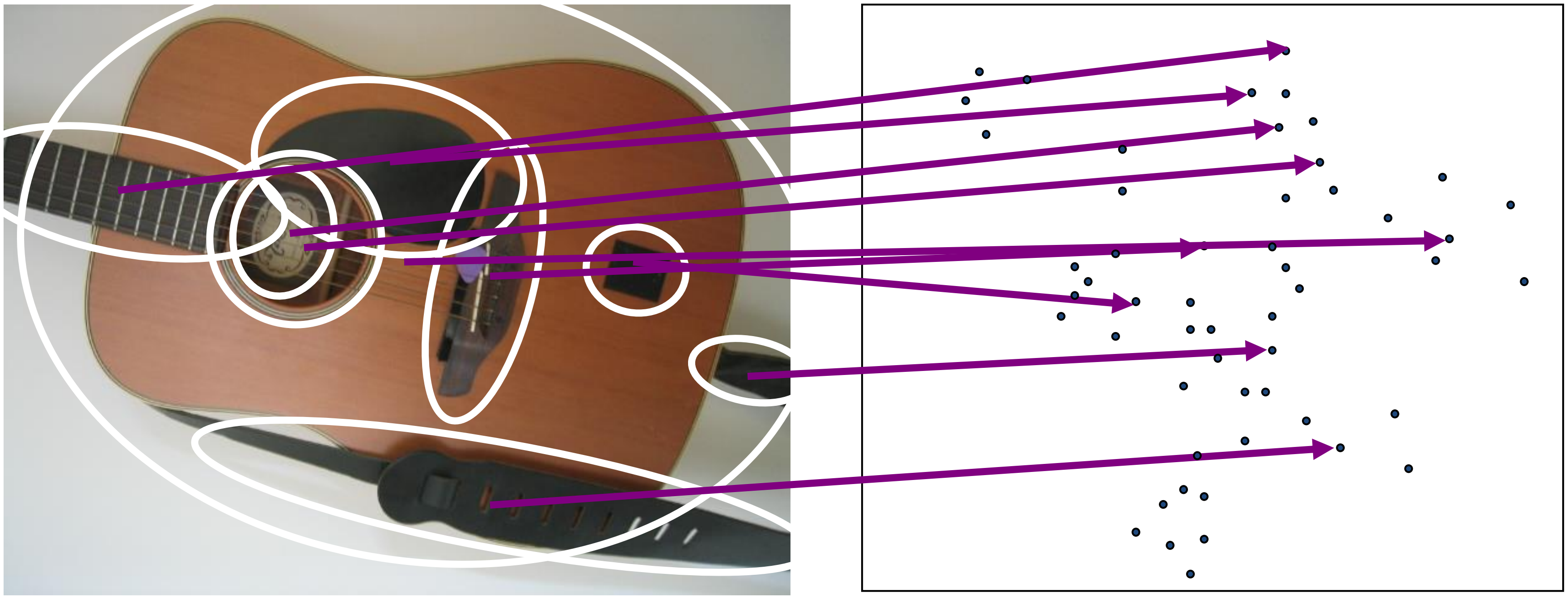
Training the vocabulary tree



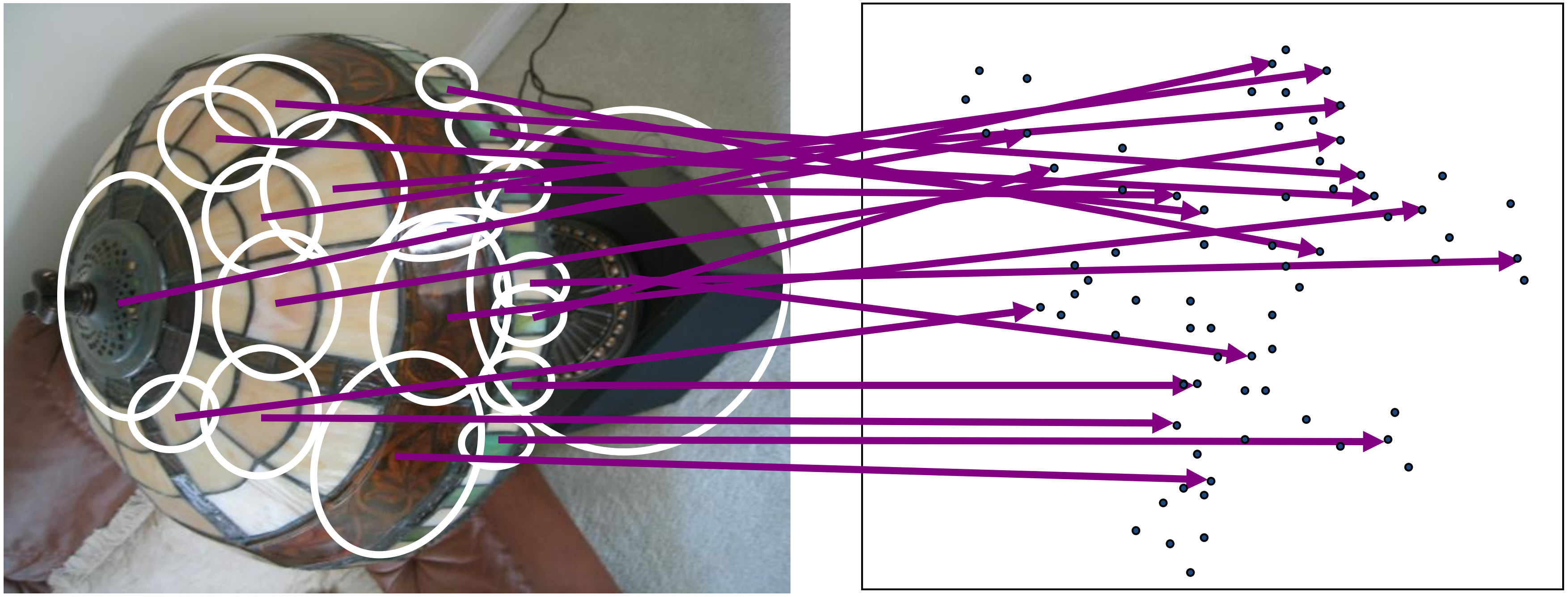
Training the vocabulary tree



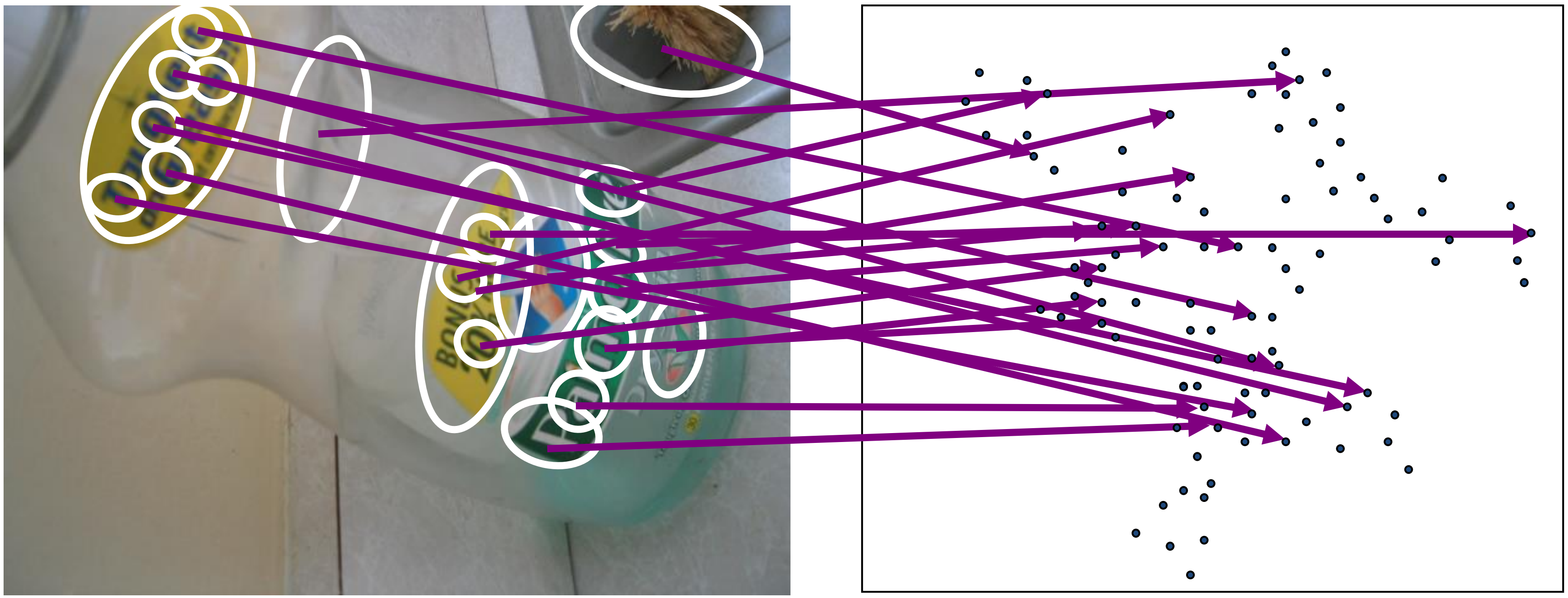
Training the vocabulary tree



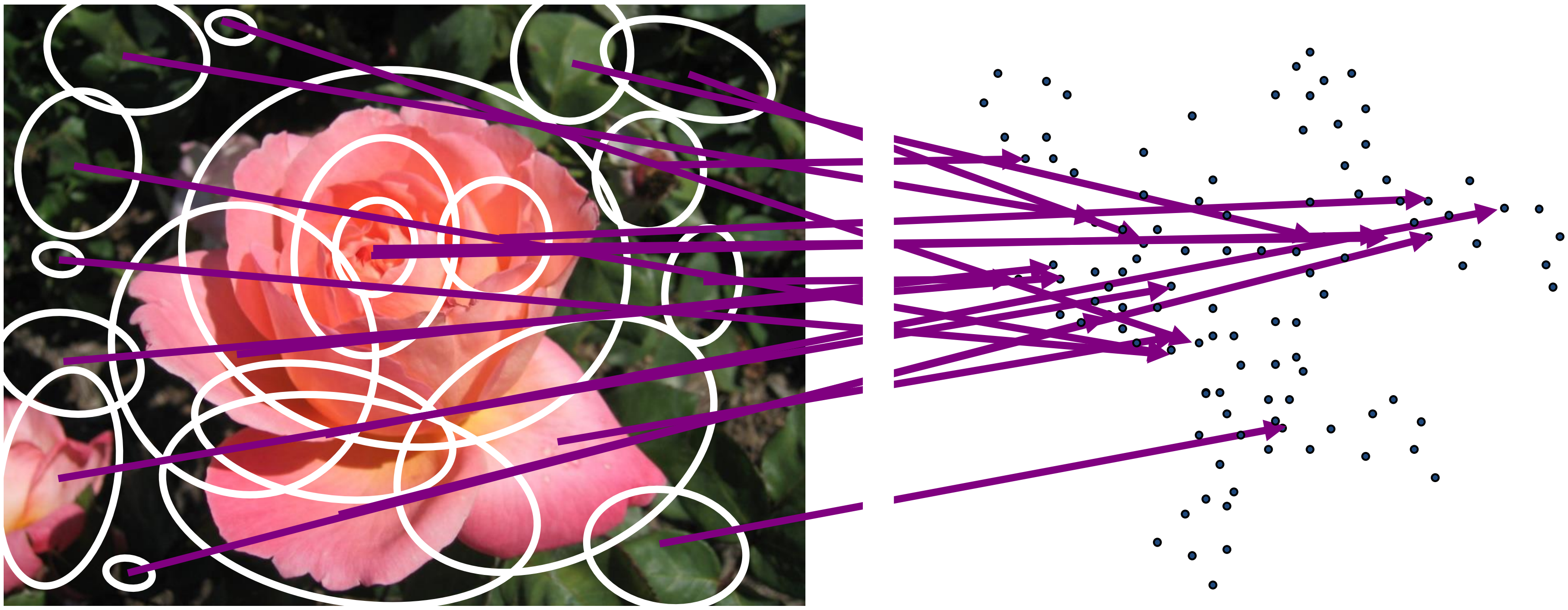
Training the vocabulary tree

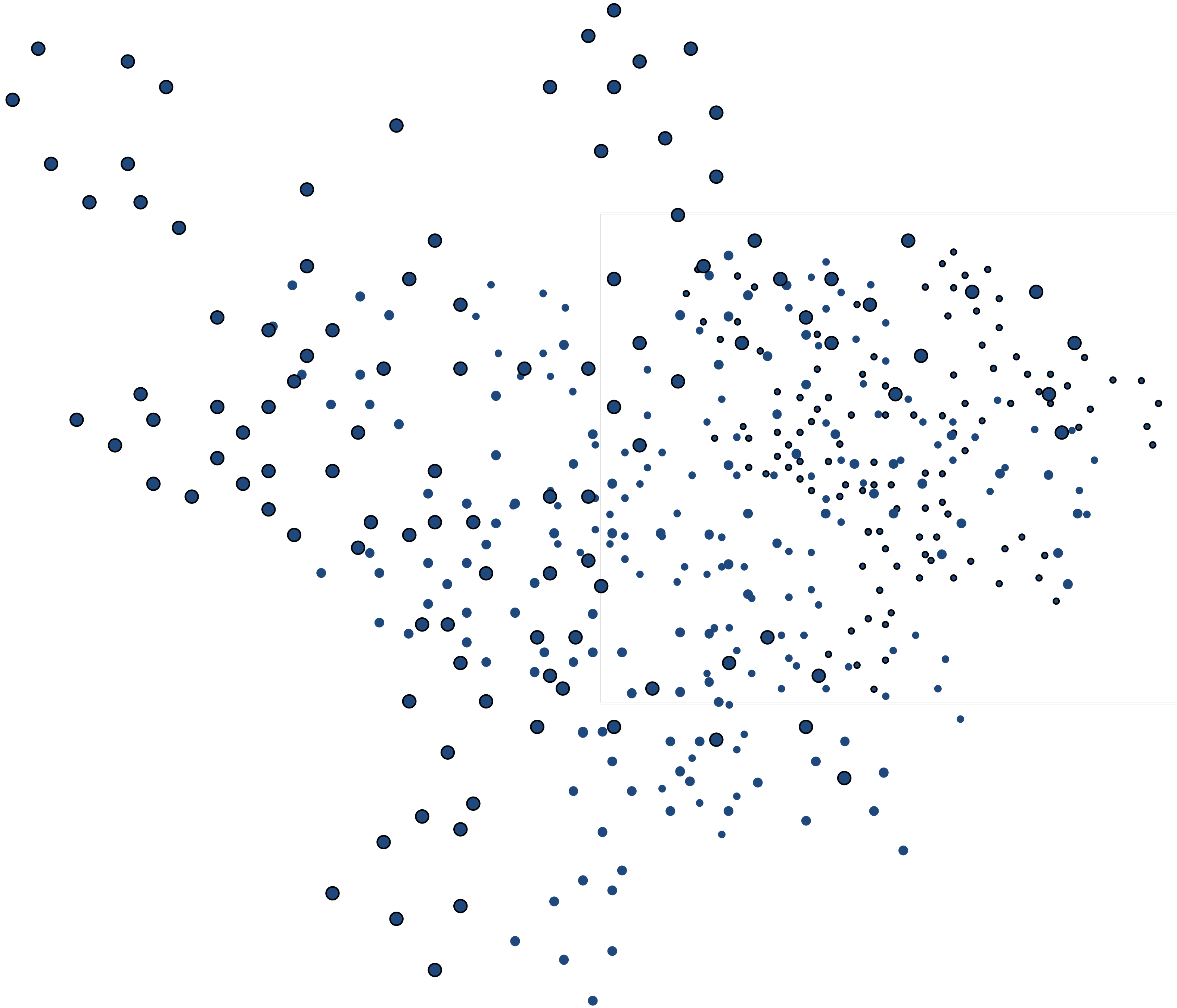


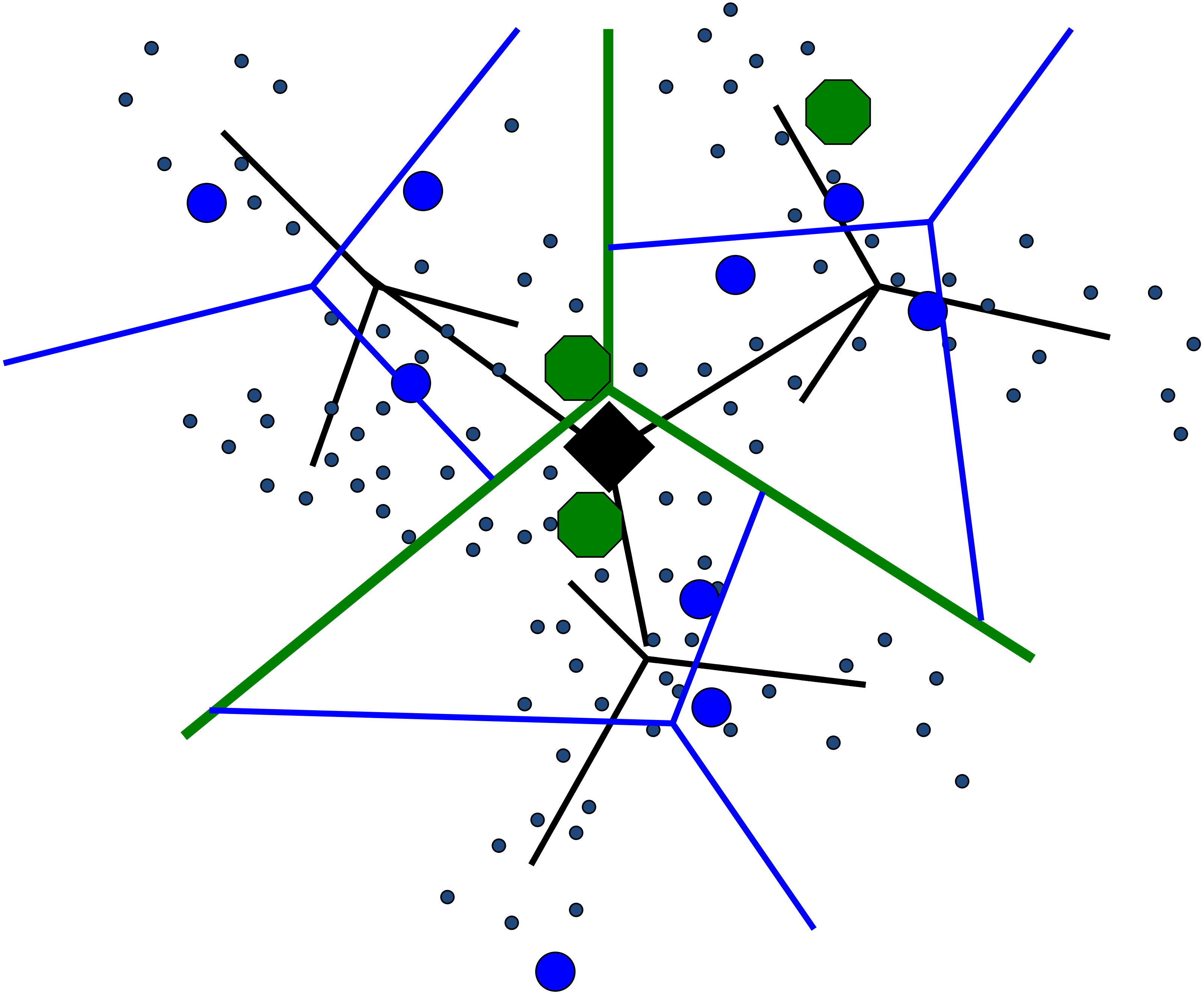
Training the vocabulary tree

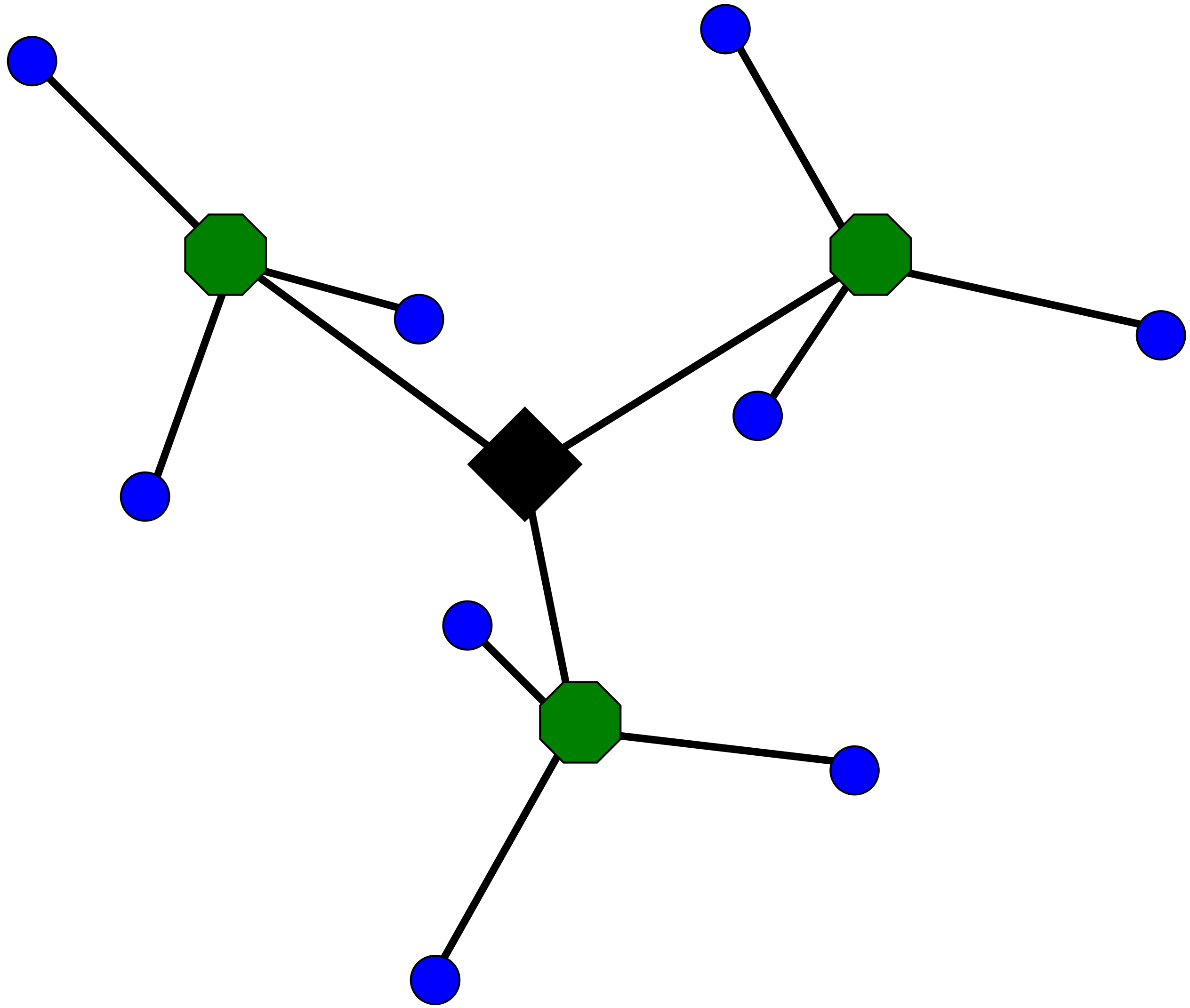


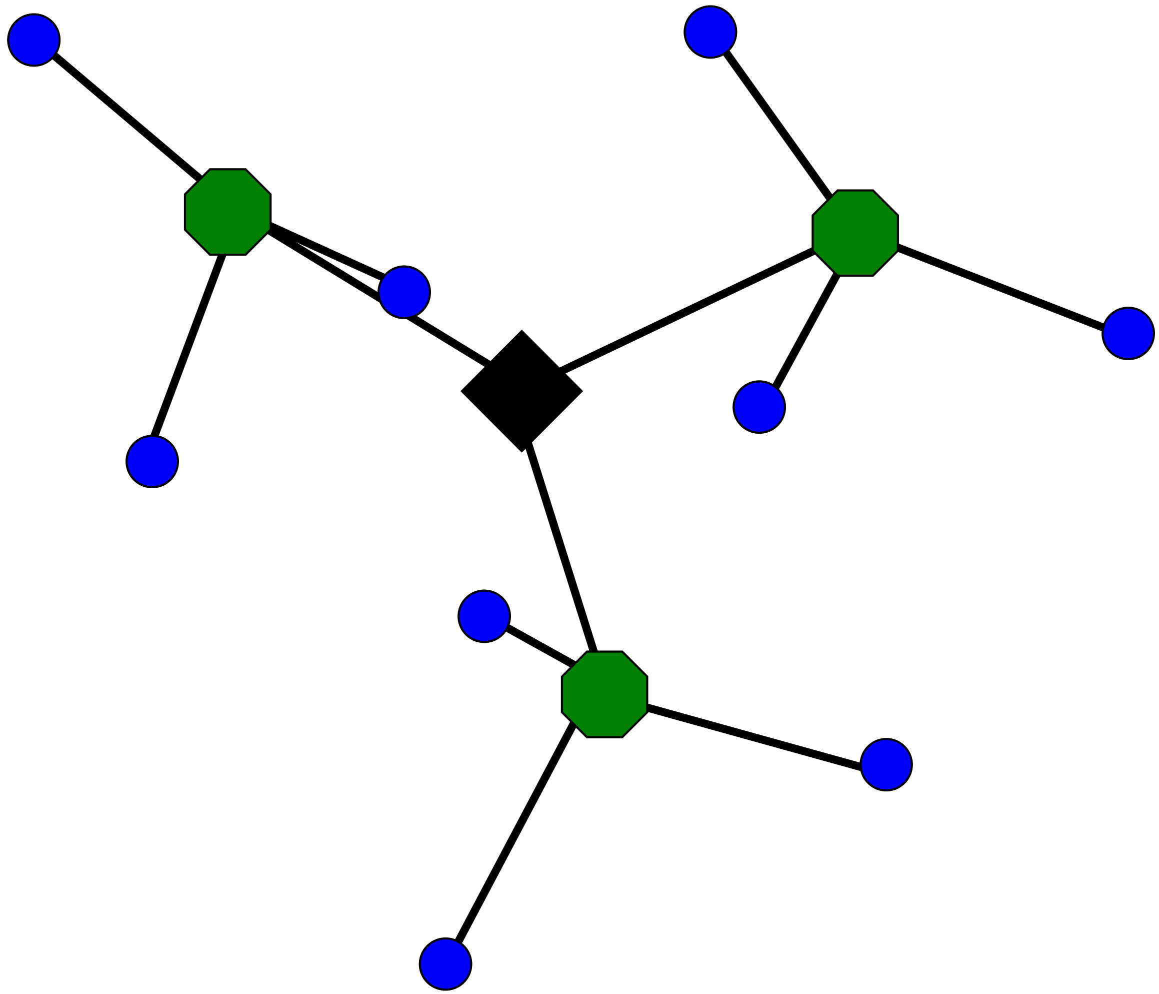
Training the vocabulary tree

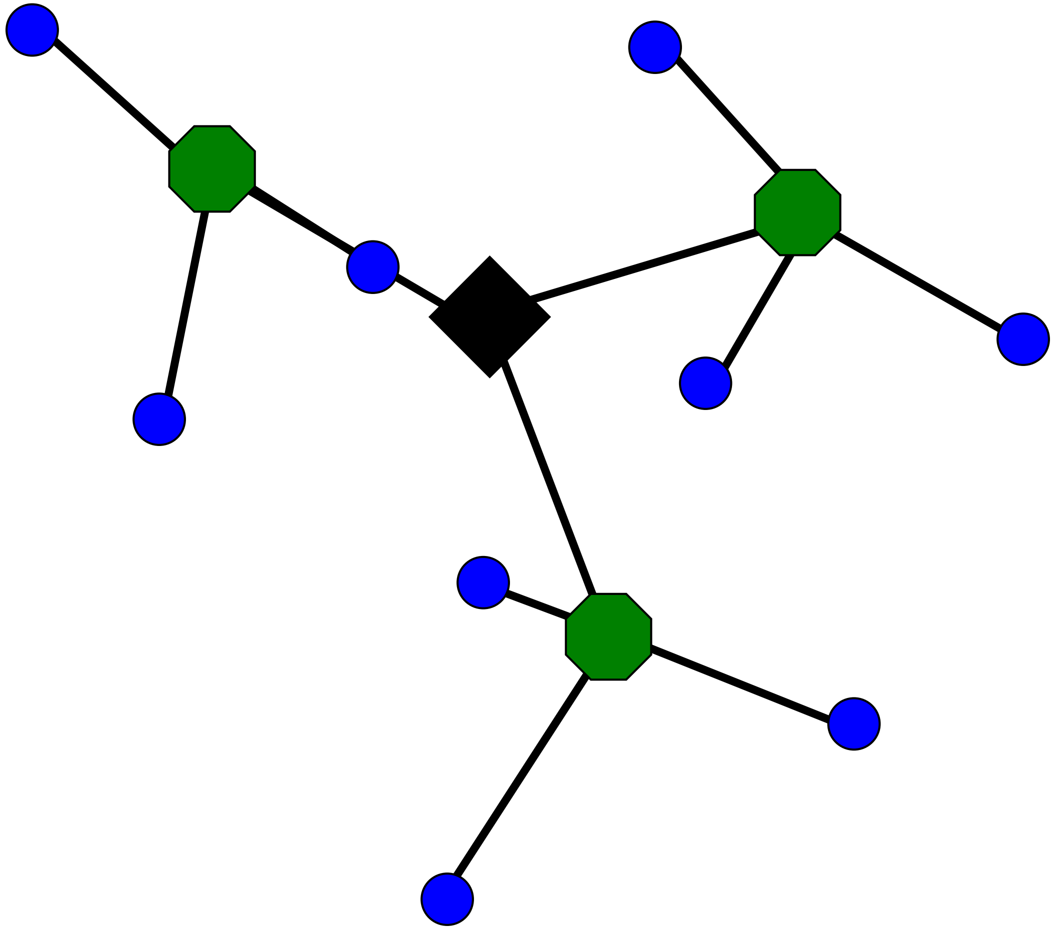


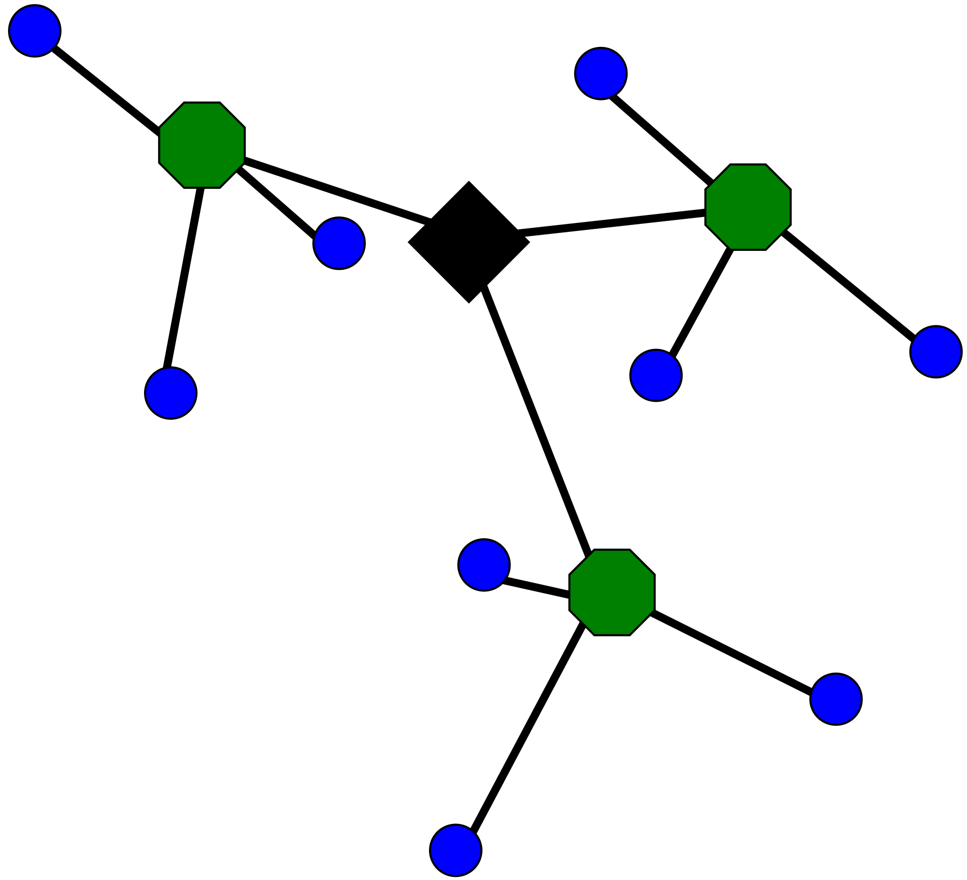


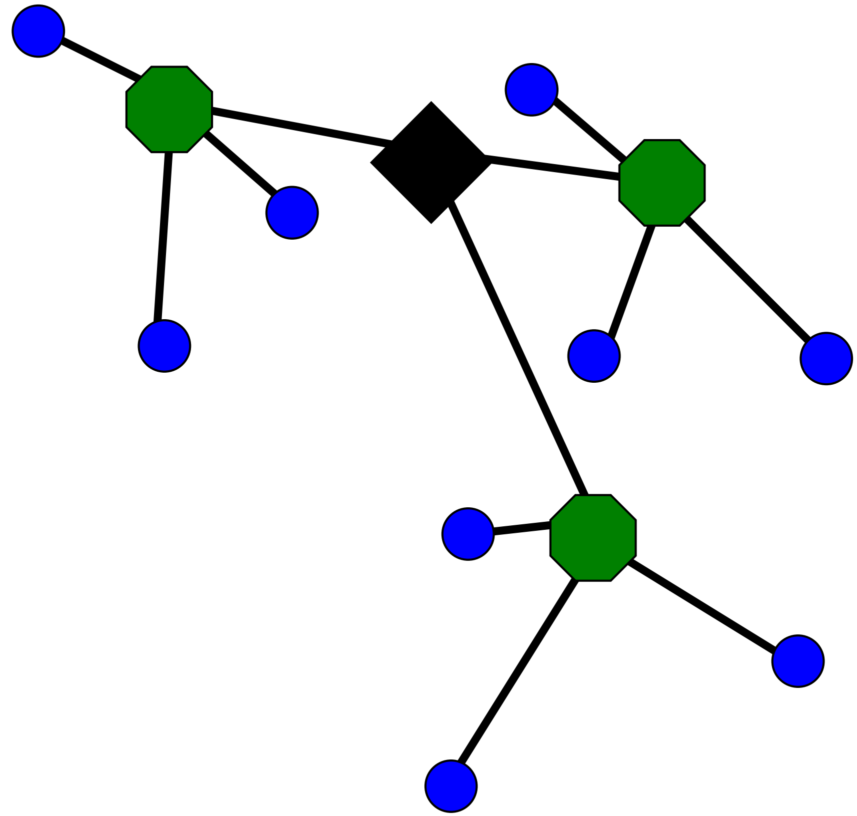


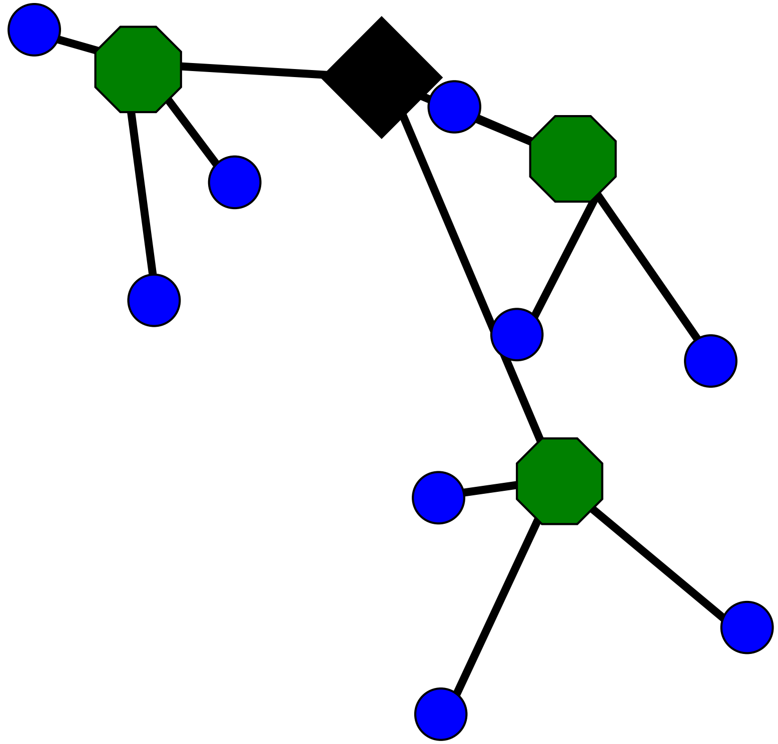


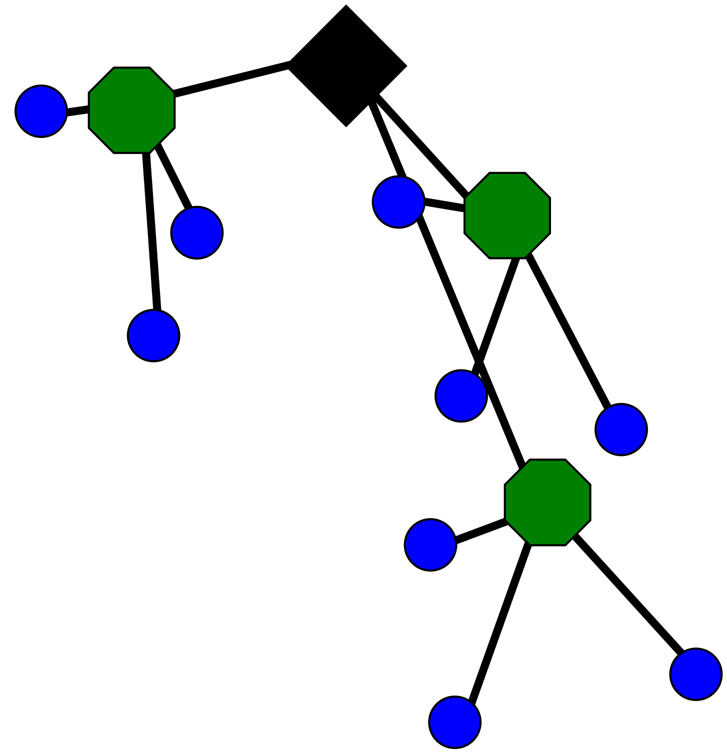


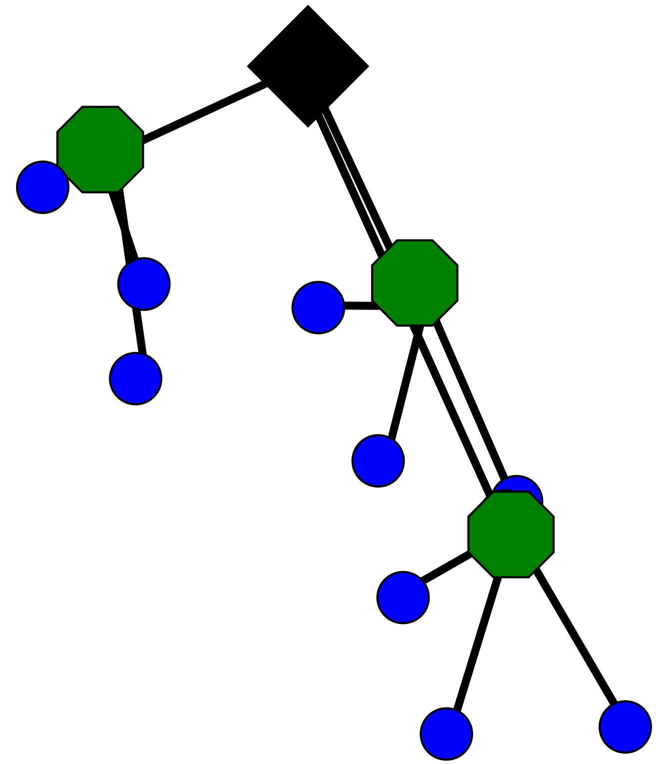


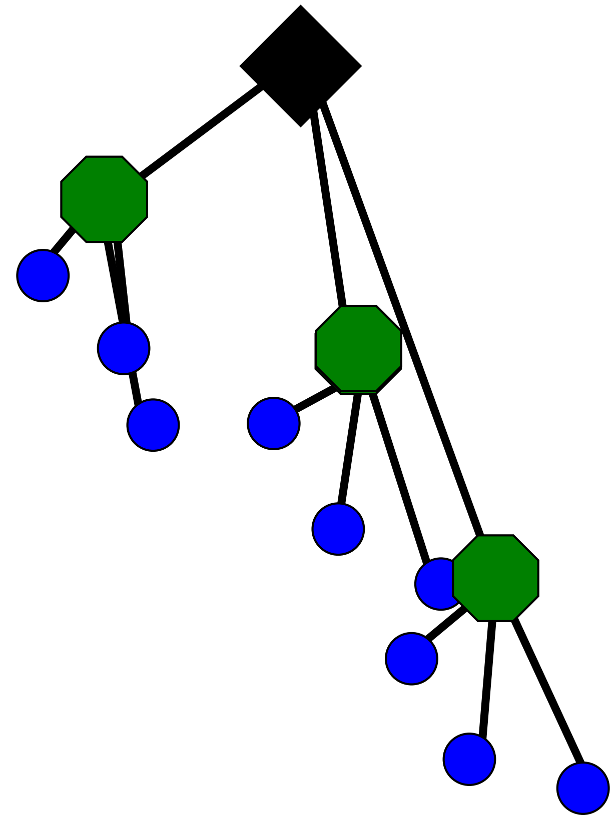




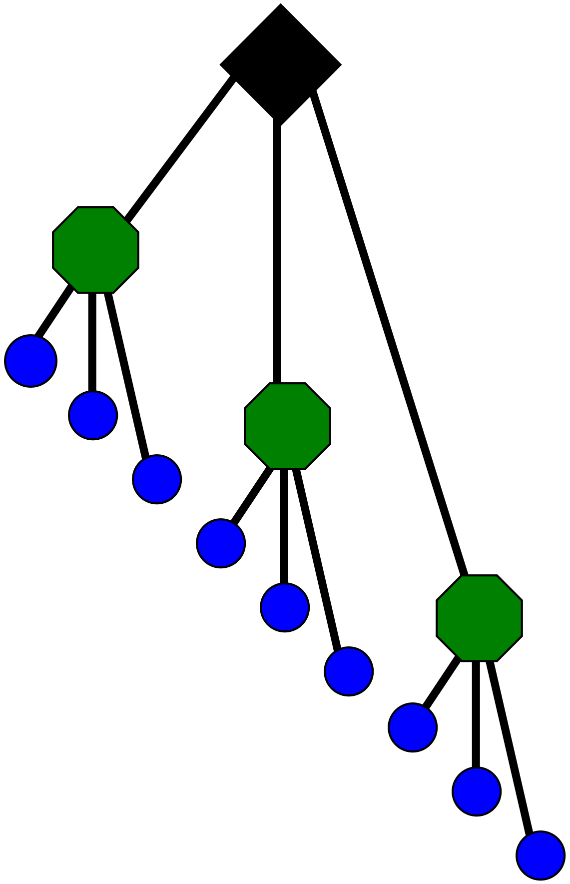




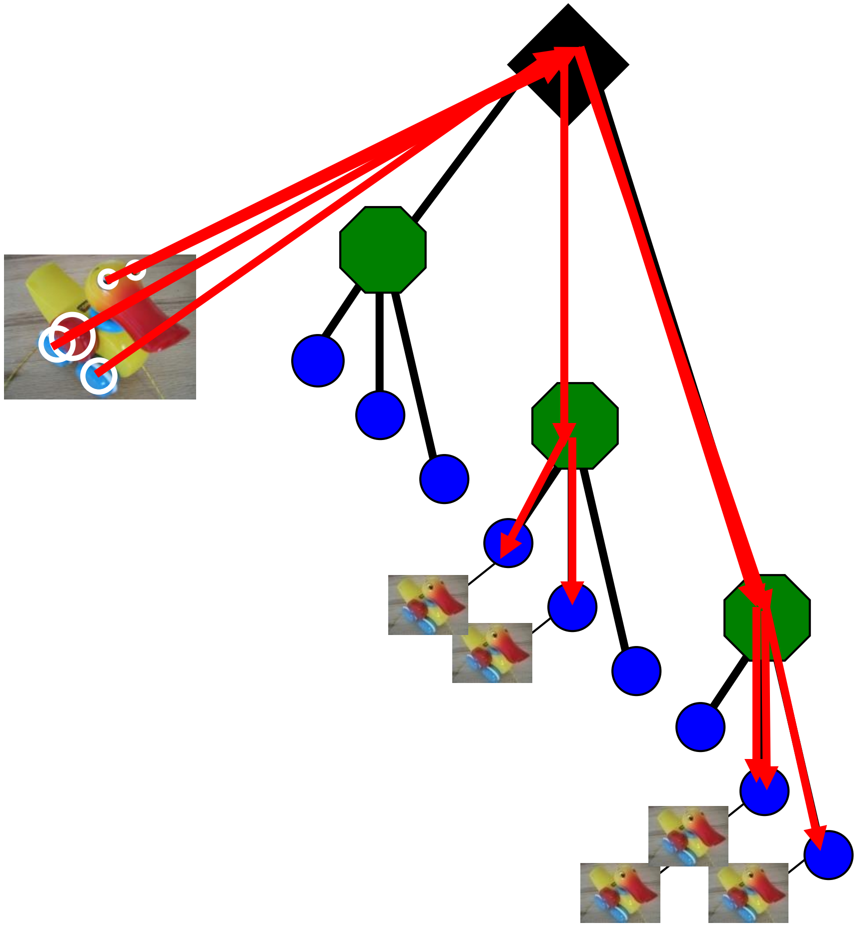


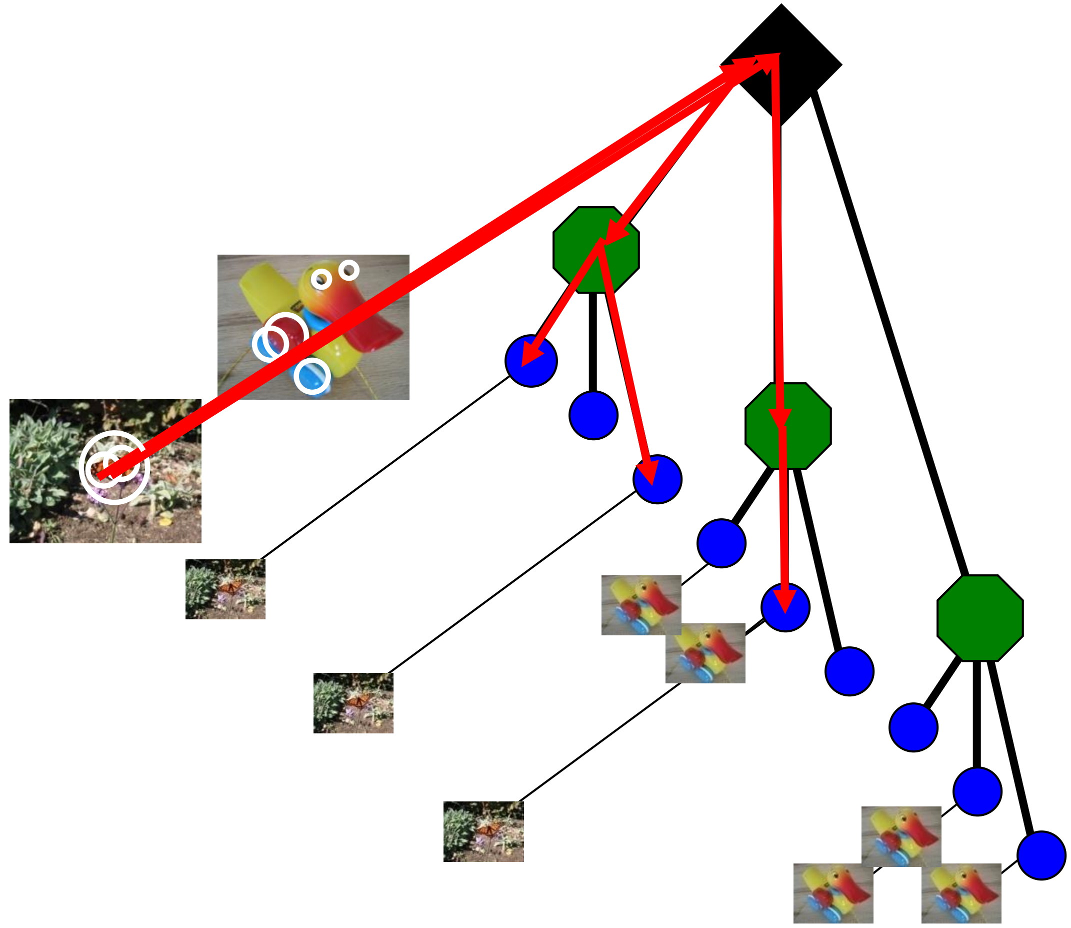


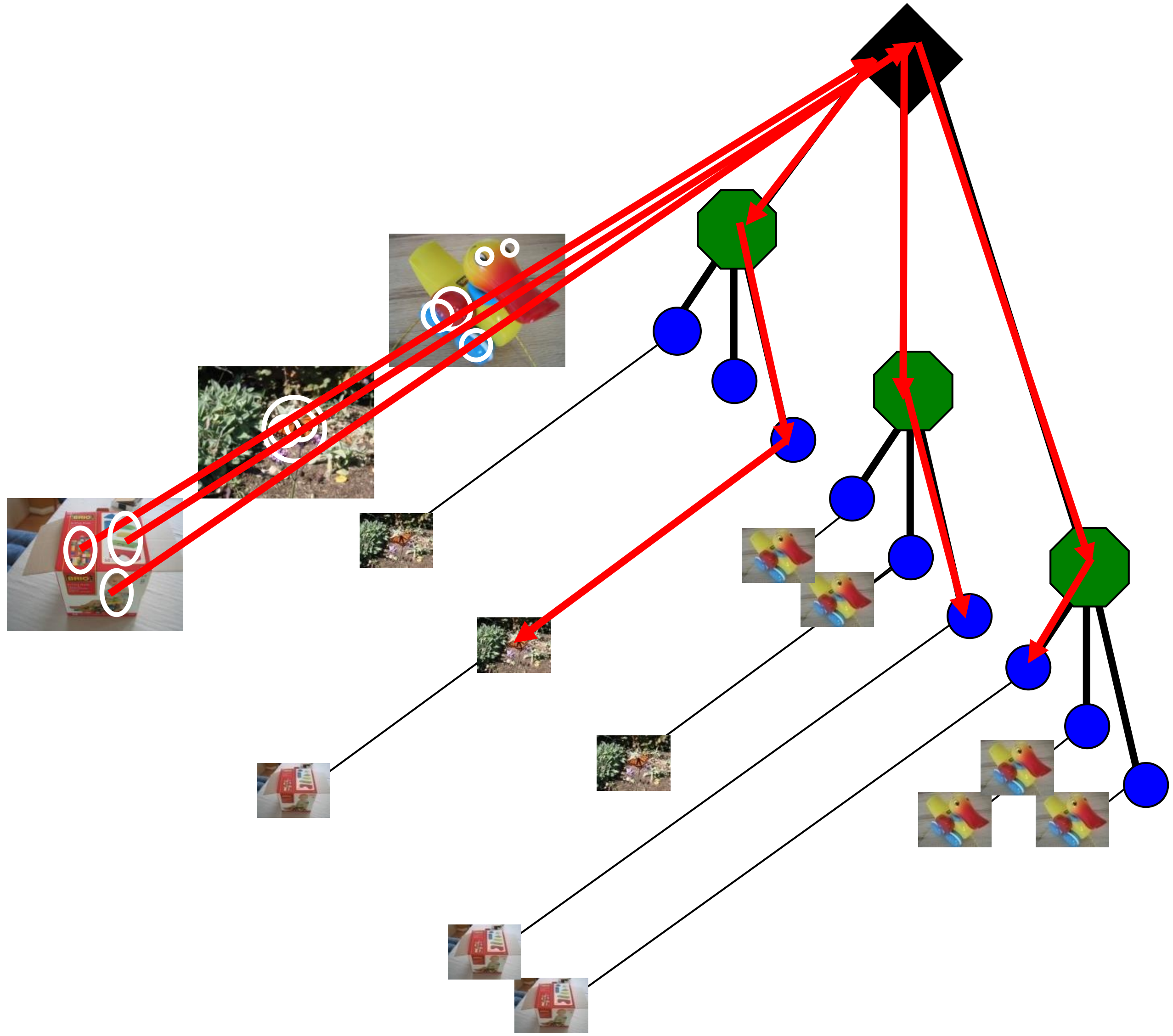
Vocabulary tree built recursively



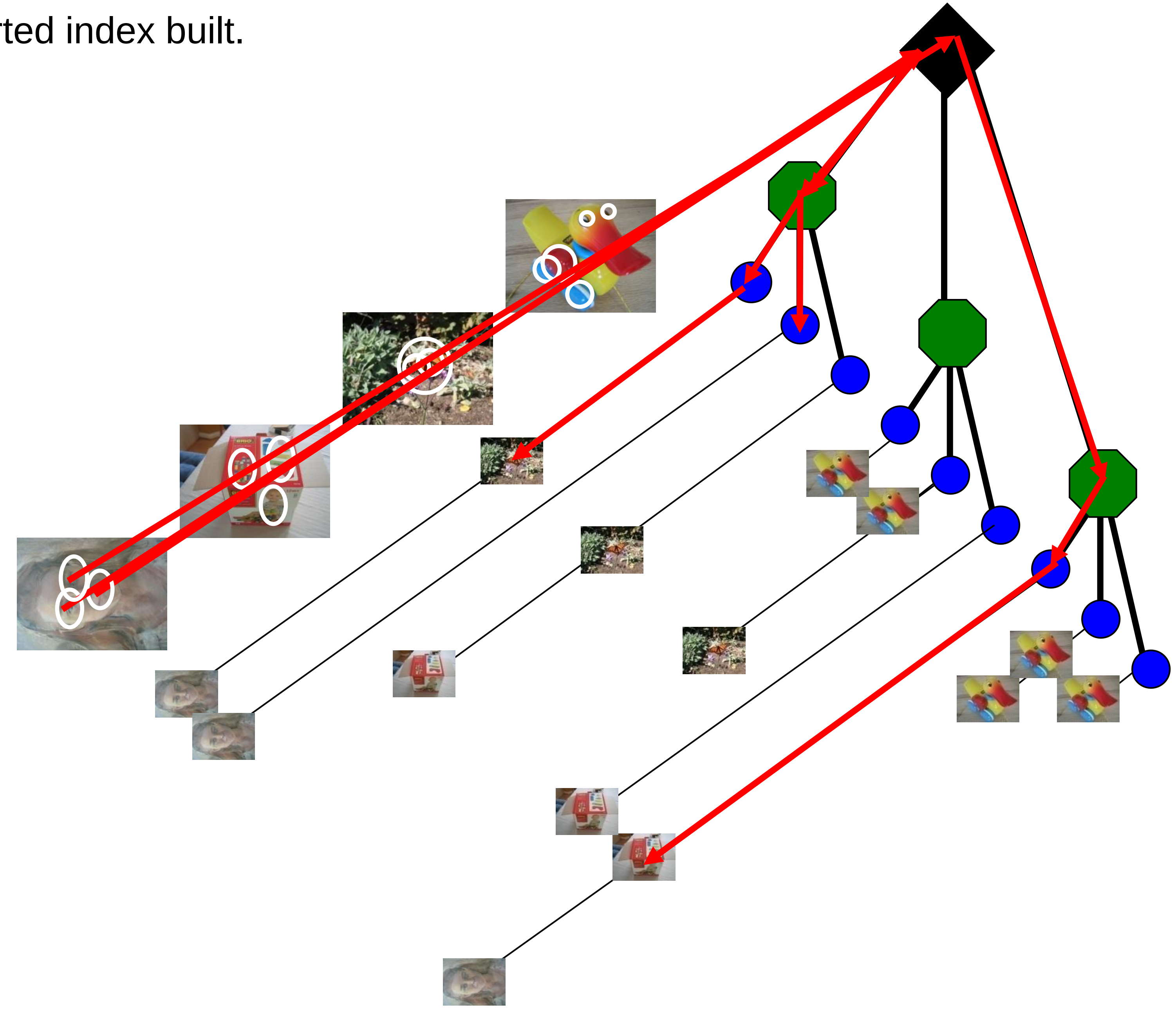
Each leaf has inverted index

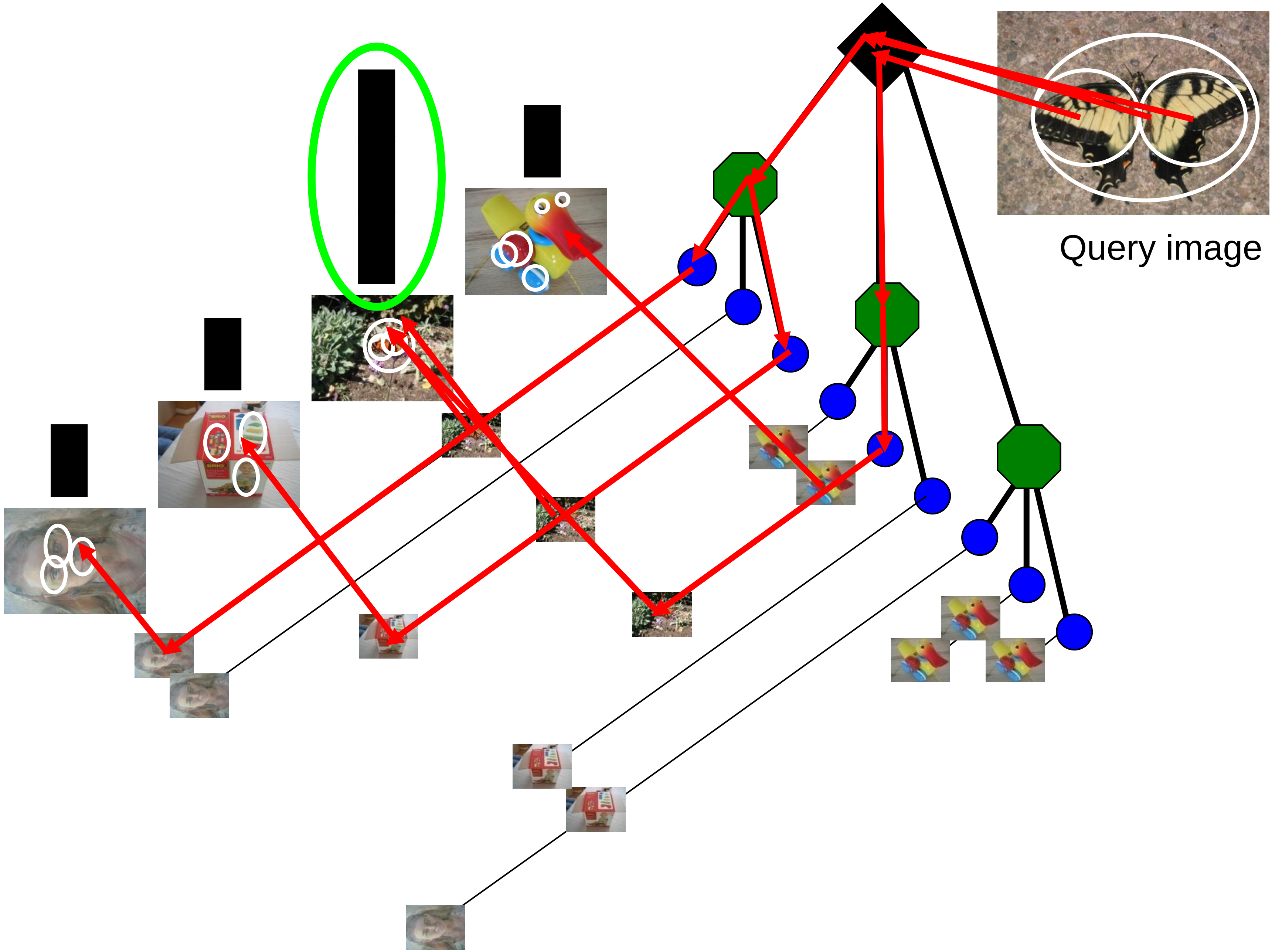




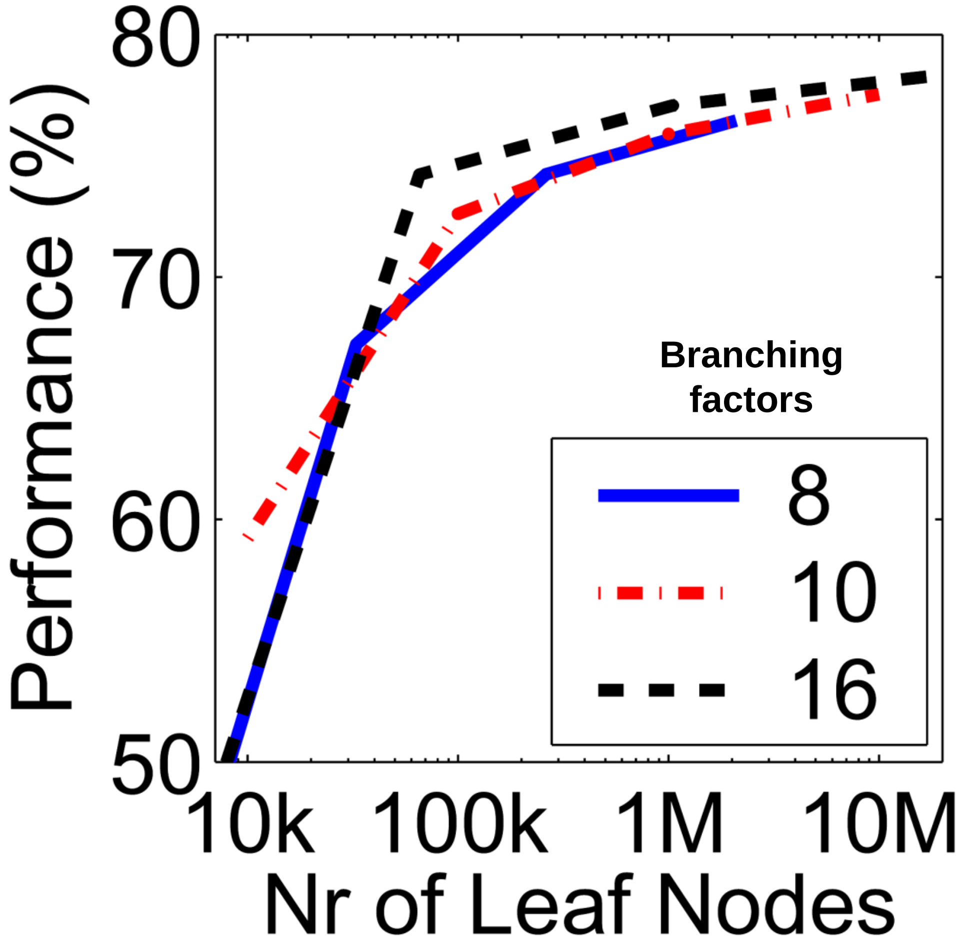


Inverted index built.

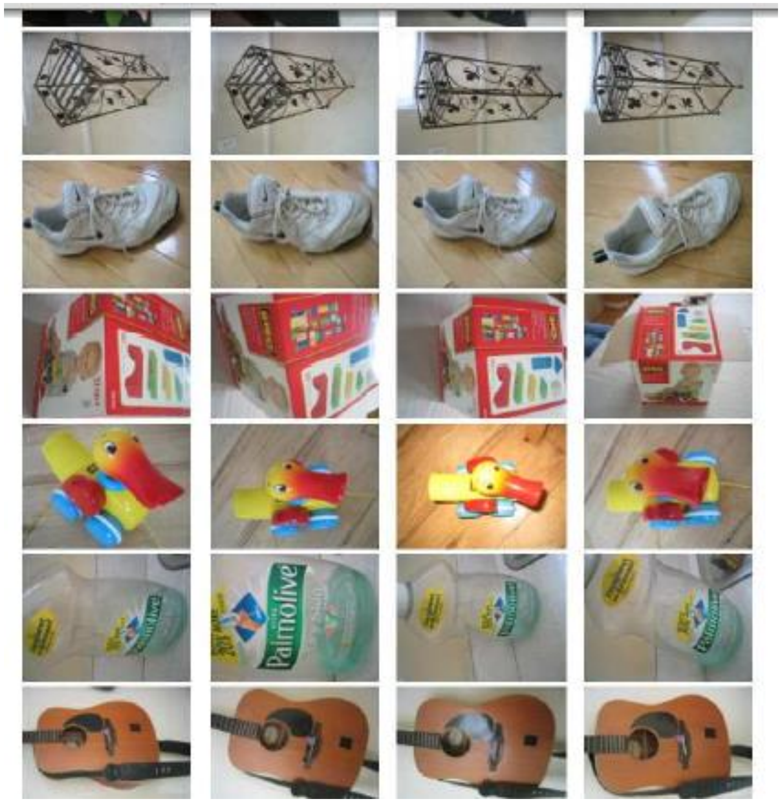



Query image

Vocabulary size



Recognition with 6347 images

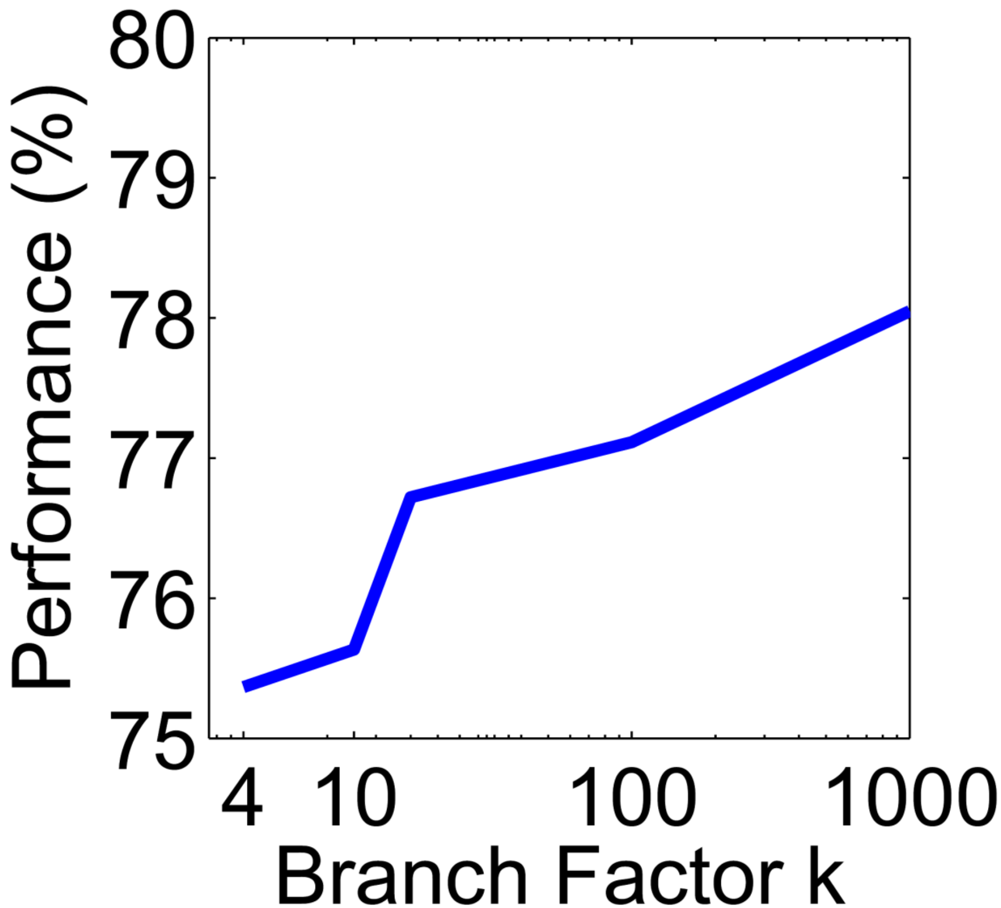


Nister & Stewenius, CVPR 2006

Influence on performance, sparsity

Kristen Grauman

Higher branch factor works better
(but slower)

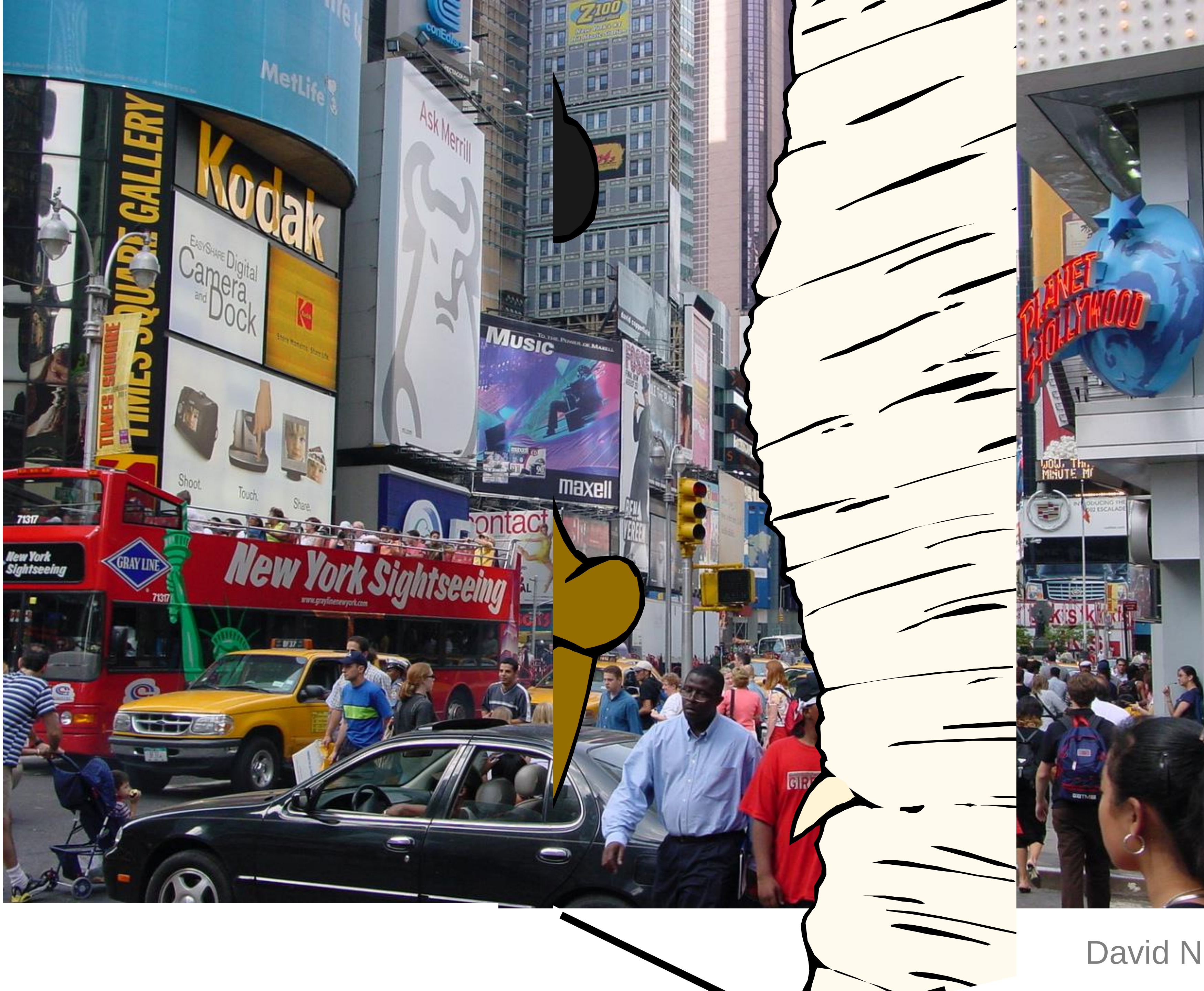


(2006) 110,000,000 images in 5.8 Seconds

UK

Center for Visualization & Virtual Environments

David Nister



David Nister

David Nister

David Nister

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 -> *is it really instance recognition?*
- optimal vocabulary formation remains unclear
- basic model ignores geometry – must verify afterwards, or encode via features

Kristen Grauman

Recognition 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?

How to score the retrieval results?

How might we add more spatial verification?

Precision and Recall

True positive (tp) – correct attribution
True negative (tn) – correct rejection

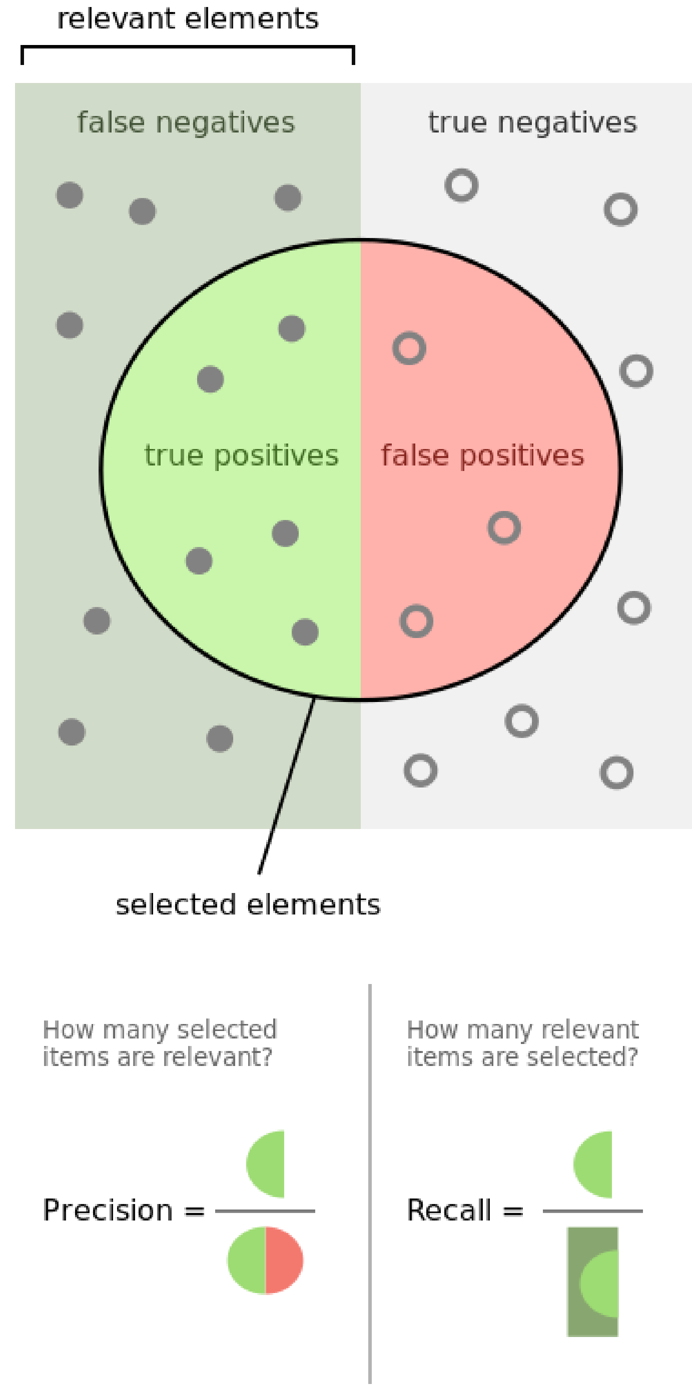
False positive (fp) – incorrect attribution
False negative (fn) – incorrect rejection

$$\text{Precision} = \frac{tp}{tp + fp}$$

Precision = #relevant / #returned

$$\text{Recall} = \frac{tp}{tp + fn}$$

Recall = #relevant / #total relevant

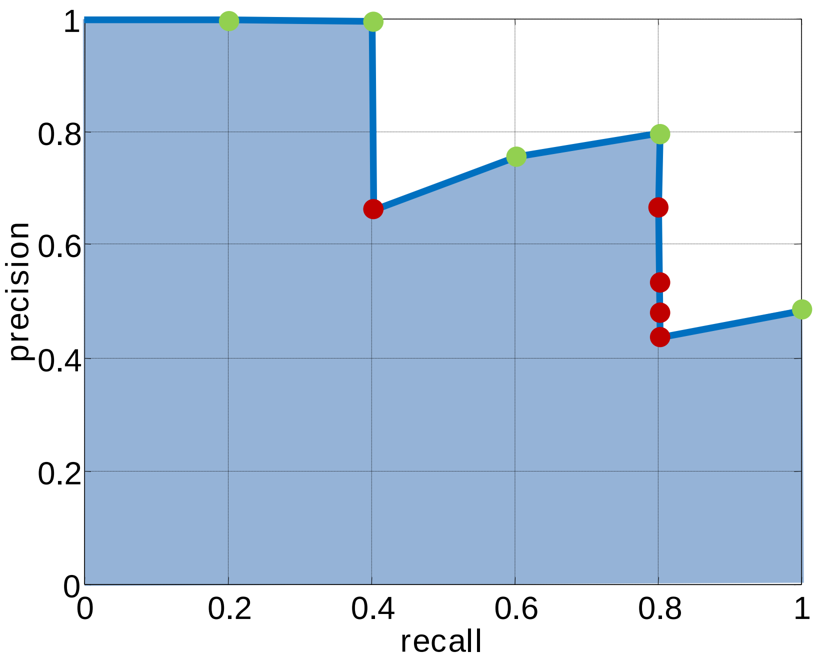


Scoring retrieval quality

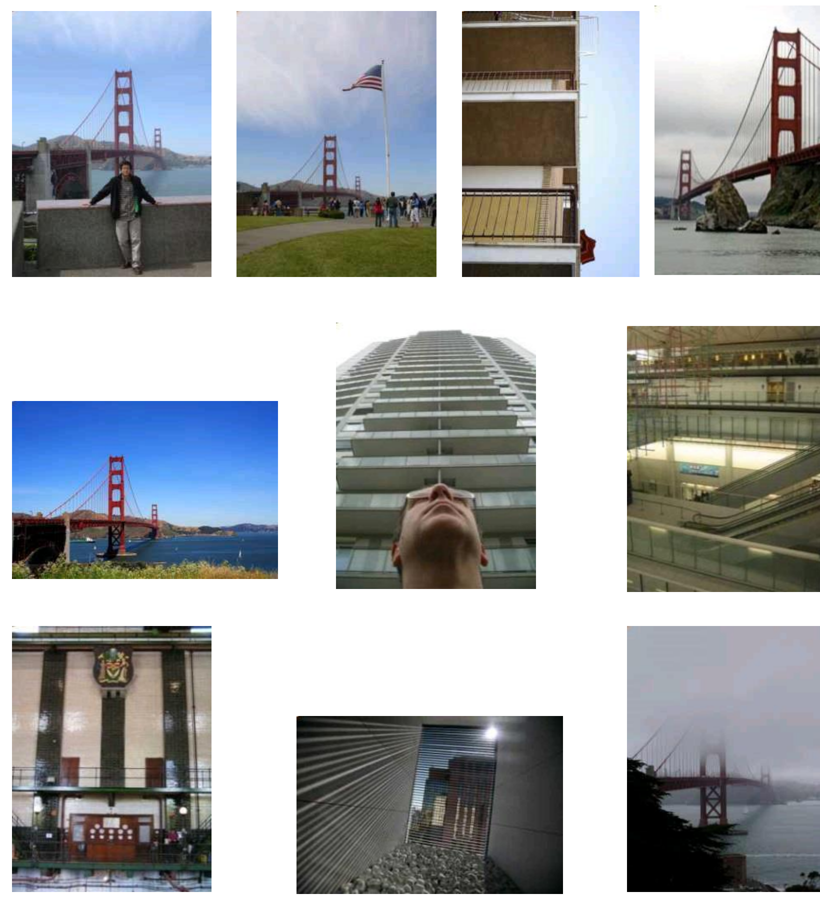
Query

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

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

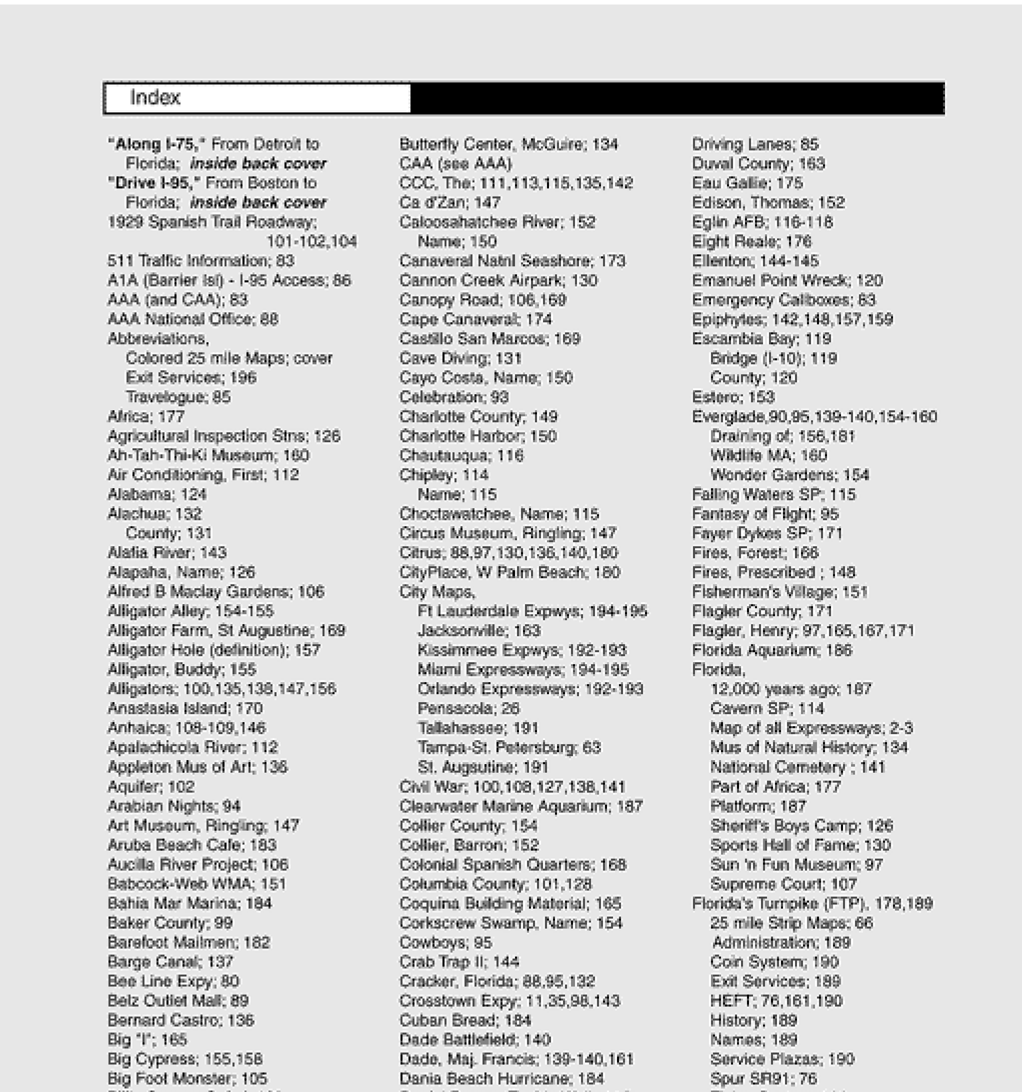


Results (ordered):



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 image 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

Total number of documents in database

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

Number of words in document d

Number of documents word i occurs in, in whole database

Kristen Grauman

Query expansion

Query: ***golf green***

Results:

- How can the grass on the ***greens*** at a ***golf*** course be so perfect?
- For example, a skilled ***golf***er expects to reach the ***green*** on a par-four hole in **...**
- Manufactures and sells synthetic ***golf*** putting ***green****s* and mats.

Irrelevant result can cause a `topic drift':

- Volkswagen ***Golf***, 1999, ***Green***, 2000cc, petrol, manual, , hatchback, 94000miles, 2.0 GTi, 2 Registered Keepers, HPI Checked, Air-Conditioning, Front and Rear Parking Sensors, ABS, Alarm, Alloy

Slide credit: Ondrej Chum

Query expansion

Chum, Philbin, Sivic, Isard, Zisserman: Total Recall..., ICCV 2007 Ondrej Chum

Recognition 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?

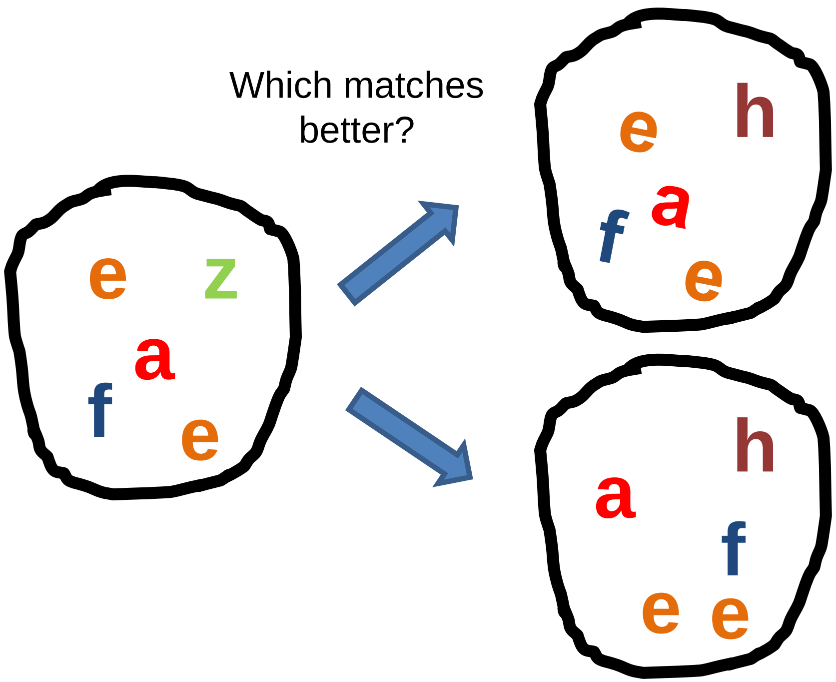
How to score the retrieval results?

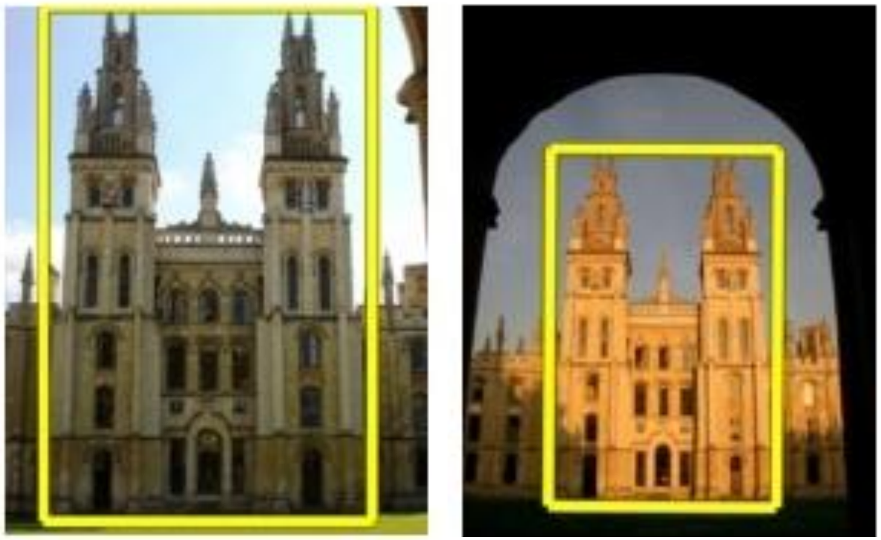
How might we add more spatial verification?

Kristen Grauman

Can we be more accurate?

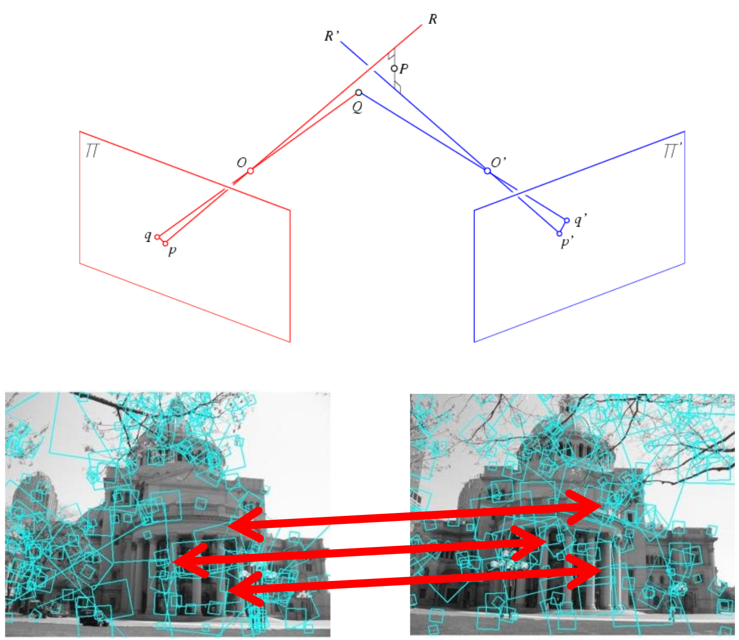
So far, we treat each image as containing a "bag of words", with no spatial information



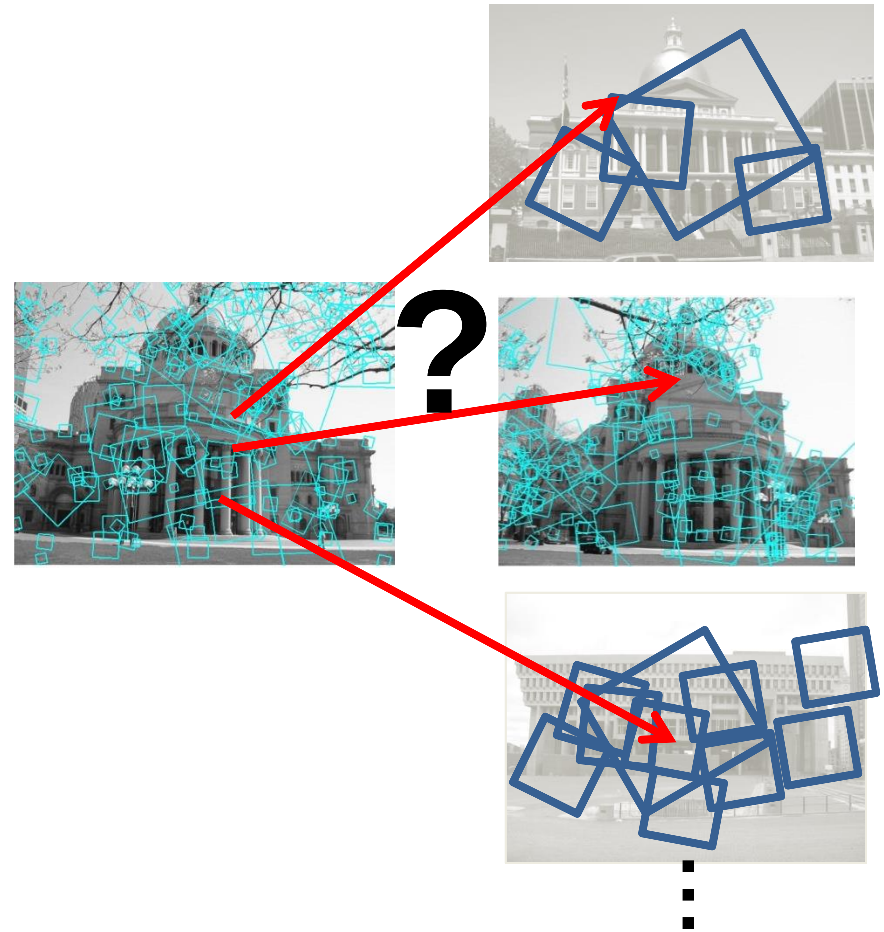


Real objects have consistent geometry

Multi-view matching



vs



Matching two given views for depth

Search for a matching view for recognition

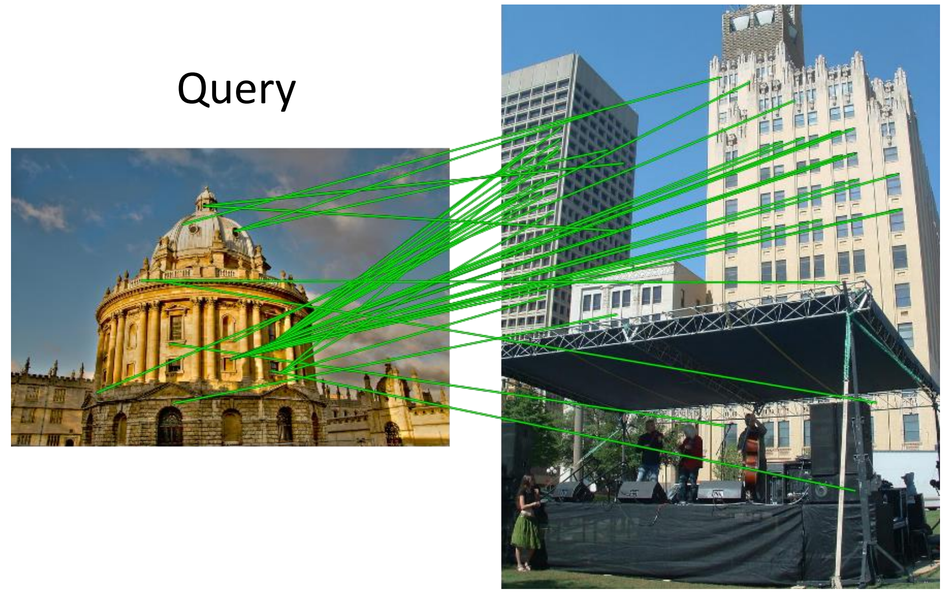
Kristen Grauman

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.

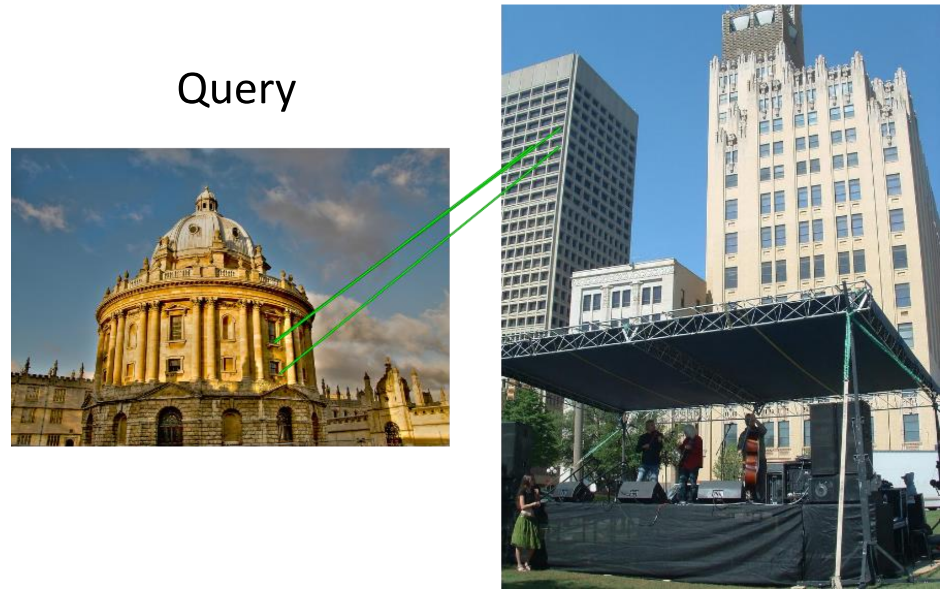
Slide credit: Ondrej Chum

Spatial Verification

Query

DB image with high BoW similarity

Query



DB image with high BoW similarity

Only some of the matches are mutually consistent with real-world geometry imaged by a camera.

Ondrej Chum

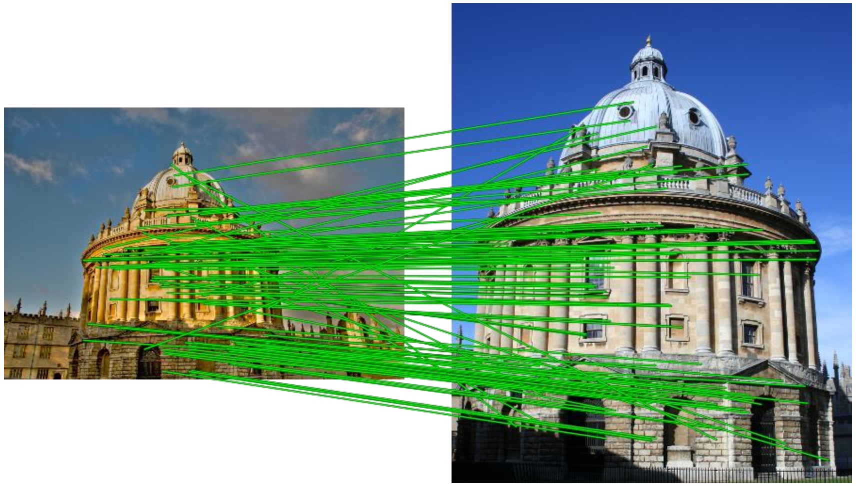
Spatial Verification: two basic strategies

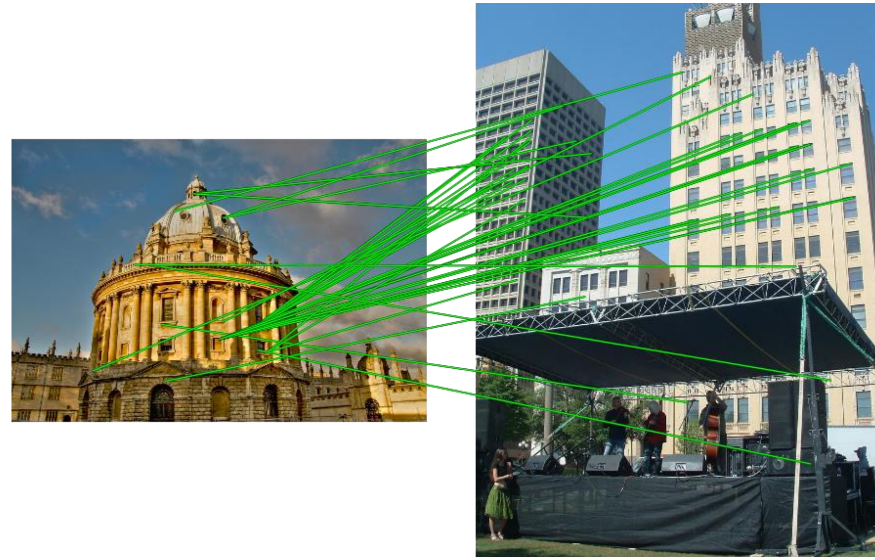
- RANSAC
  - Typically sort by BoW similarity as initial filter
  - Verify by checking support (inliers) for possible transformations
    - e.g., “success” if find a transformation with > N inlier correspondences

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

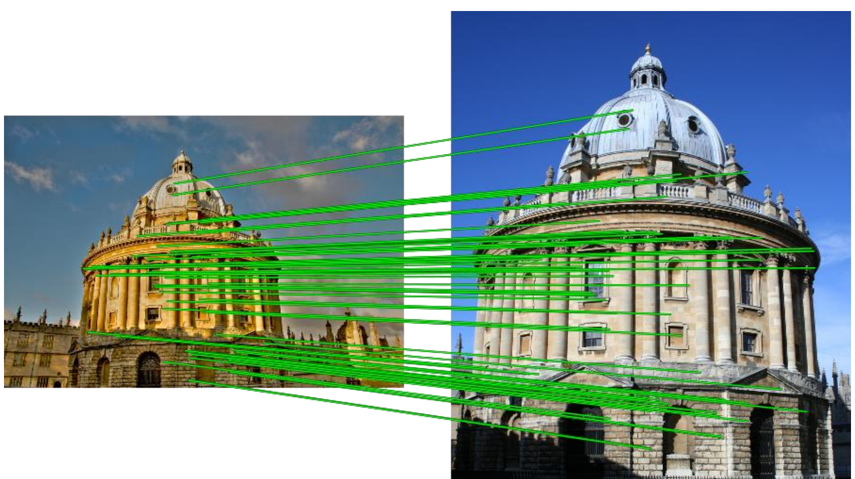
Kristen Grauman

No verification

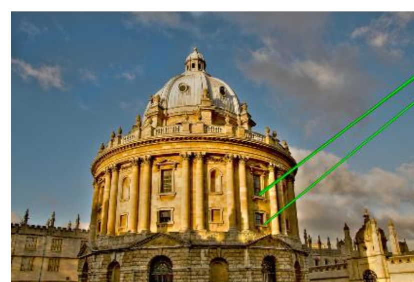




RANSAC verification



Fails to meet threshold on # inliers! Good!



Recognition via alignment

Pros:

– Effective for reliable features within clutter

– Great for matching specific instances

Cons:

– Expensive post-process (how long for proj3?!)

– Not suited for category recognition

Summary

- **Bag of words**: quantize feature space into discrete visual words
  – Summarize image by distribution of words
- **Inverted index**: visual word index for faster query time
- **Evaluation:**
- **Additional spatial verification alignment:**
  – Robust fitting : RANSAC, Generalized Hough Transform
  – We will do this in detail later on in the course

Kristen Grauman

Lessons from a decade later

For *Category* recognition (project 3)

- Bag of Feature models remained the state of the art until Deep Learning.
- Spatial layout either isn't that important or its too difficult to encode.
- Quantization error is, in fact, the bigger problem. Advanced feature encoding methods address this.
- Bag of feature models are nearly obsolete.
  At best they seem to be inspiring tweaks to deep models e.g., NetVLAD.

Lessons from a decade later

For *instance* retrieval (this lecture):

- deep learning is taking over.
- learn better local features (replace SIFT)
  e.g., MatchNet 2015
- learn better image embeddings (replace visual word histograms)
  e.g., Vo and Hays 2016.
- learn spatial verification
  e.g., DeTone, Malisiewicz, and Rabinovich 2016.
- learn a monolithic deep network to recognition all locations
  e.g., Google's PlaNet 2016.

James Hays