SAIR

Spatial AI & Robotics Lab

CSE 473/573-A

L21: RETRIEVAL

Chen Wang

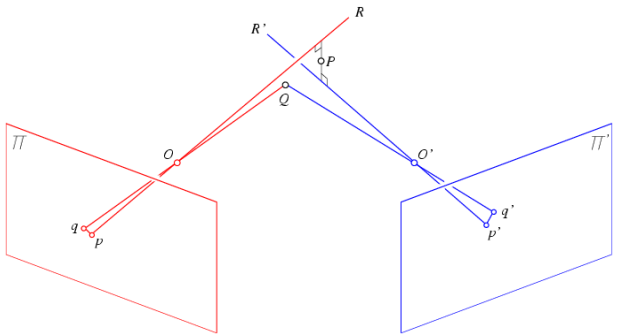
Spatial AI & Robotics Lab

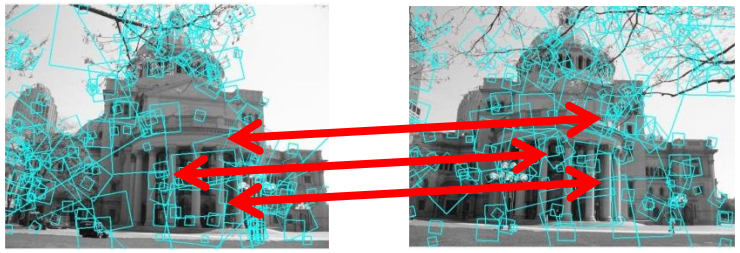
Department of Computer Science and Engineering

University at Buffalo The State University of New York

Multi-view matching (Recap)

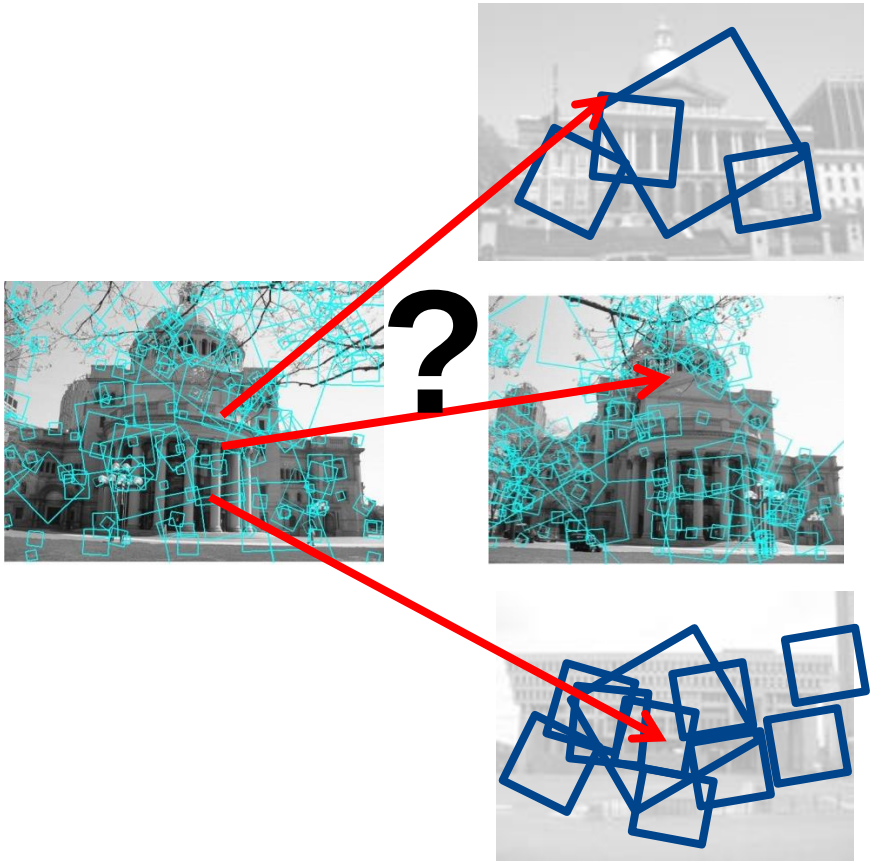
Matching two given views for depth





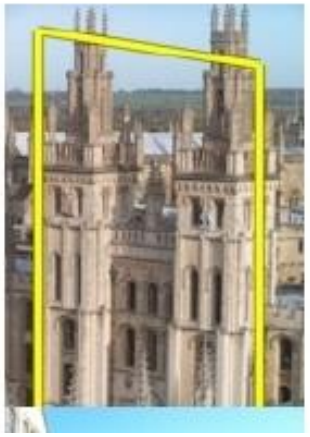
vs

Search for a matching view for recognition



Efficient Retrieval

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



Local Retrieval

1. Collect all words within query region
2. Inverted file index to find relevant frames
3. Compare word counts
4. Spatial verification

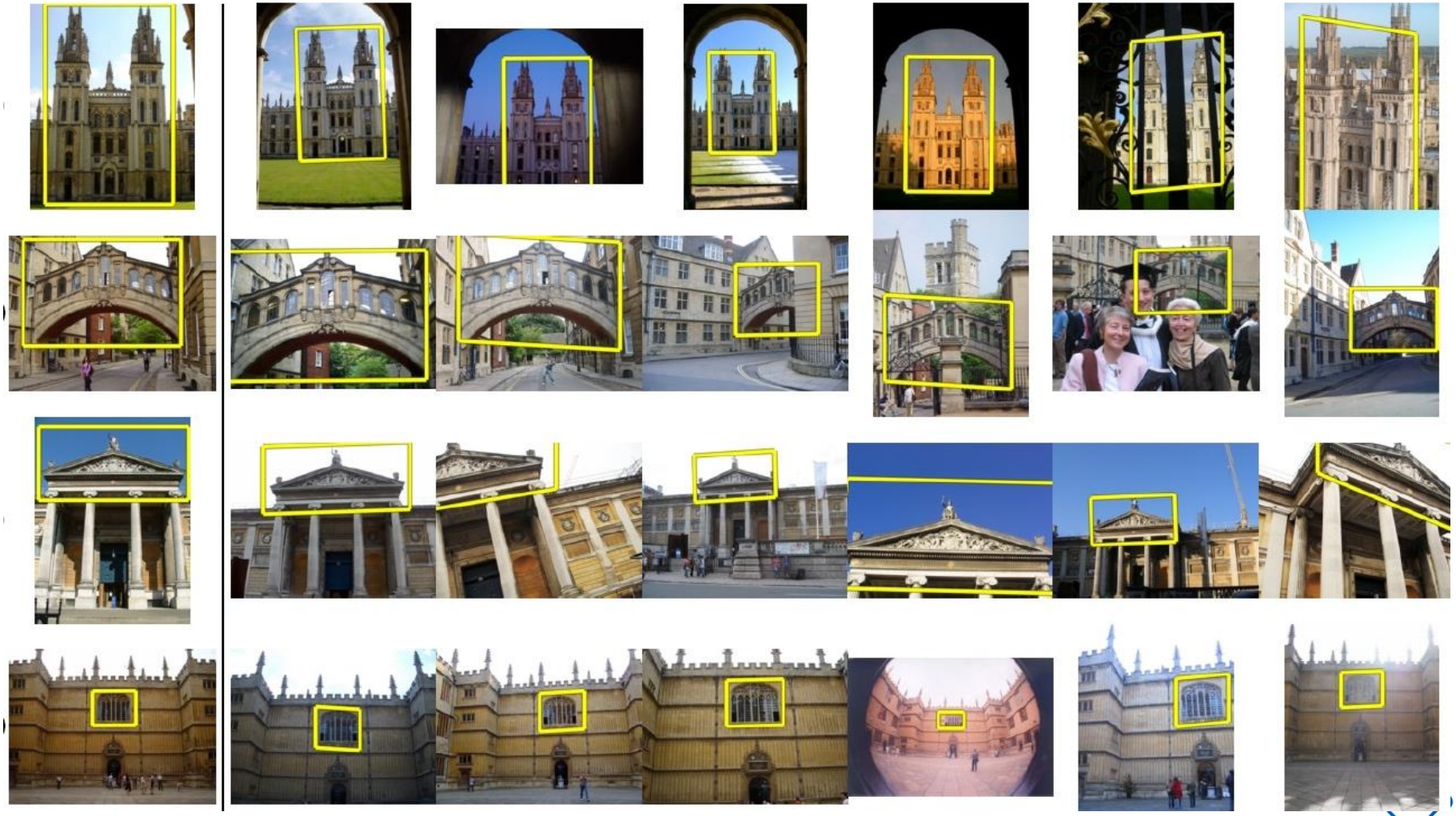
Query region

Retrieved frames

Application: Image Retrieval

Query

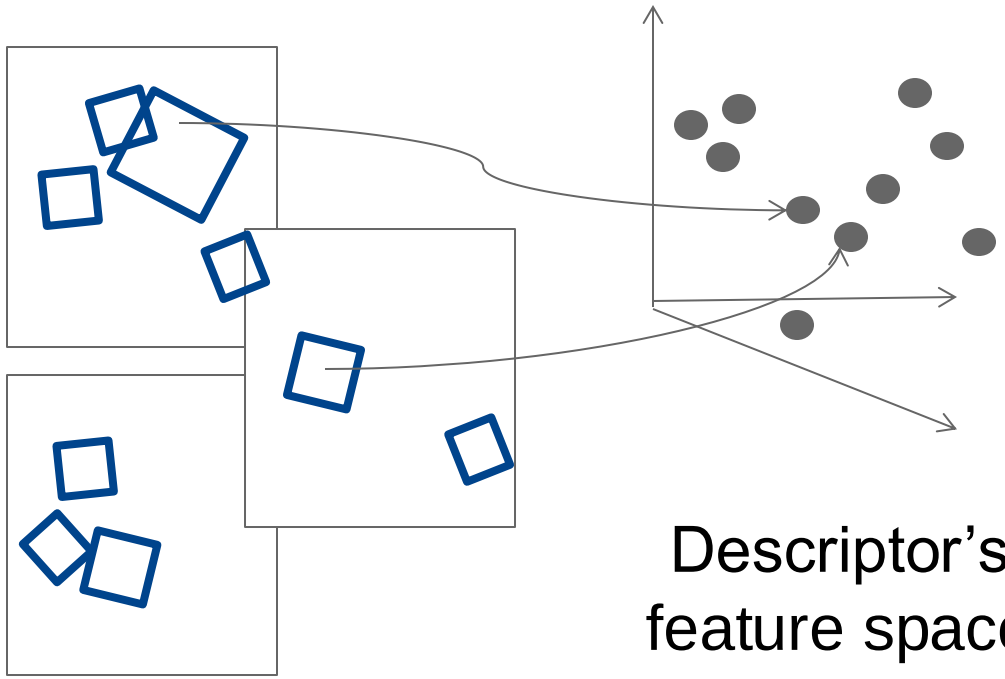
Results from 5k Flickr images (demo available for 100k set)



[Philbin CVPR'07]

Indexing local features

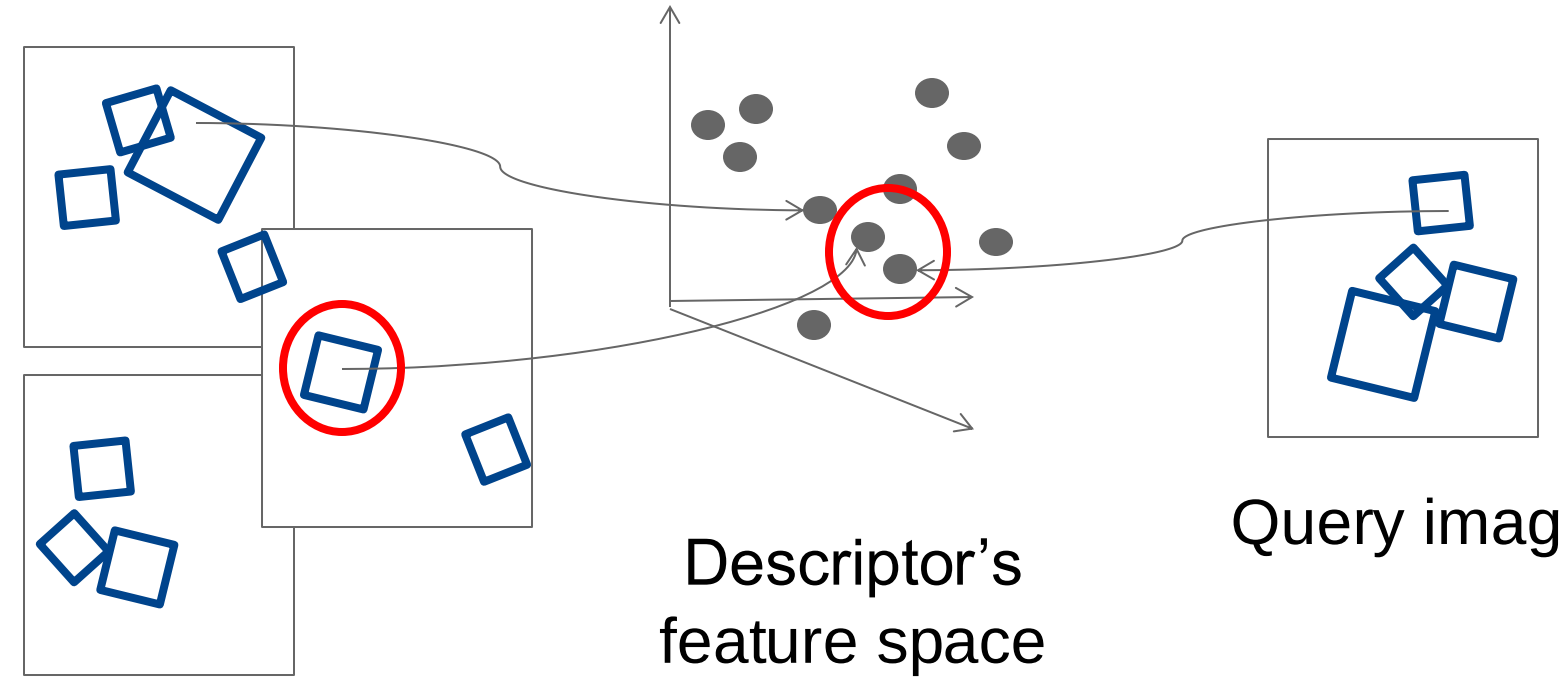
- Each patch / region has a **descriptor**, which **is a point** in some high-dimensional feature space, e.g., SIFT.



Descriptor's feature space

Indexing local features

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



Easily have millions of features to search!

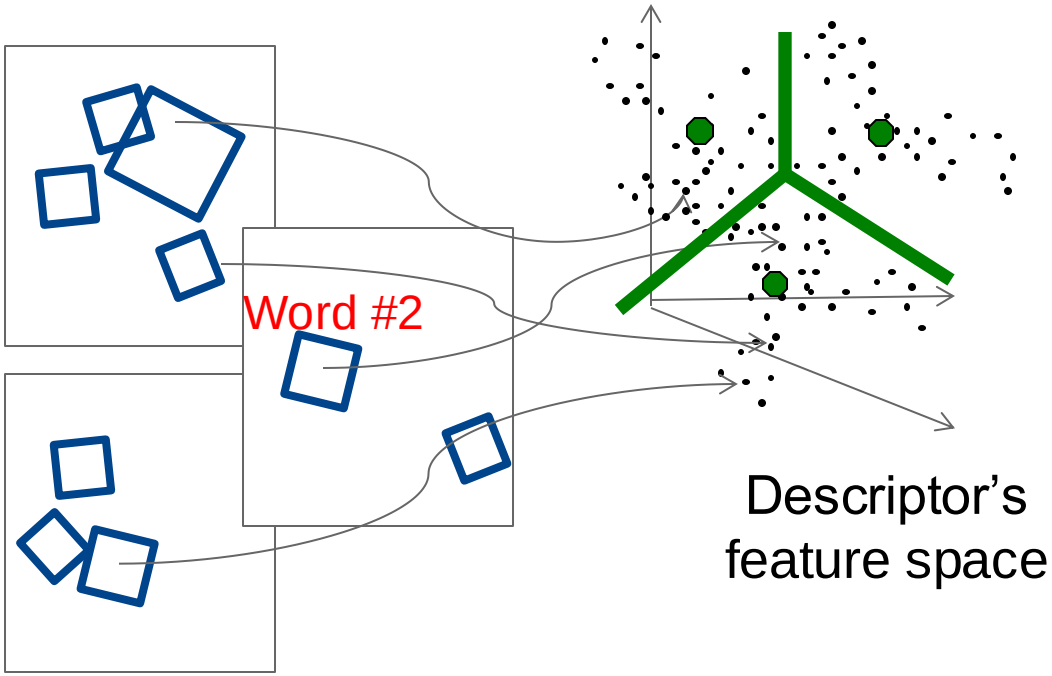
Indexing local features: inverted file index

- 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 map features to "**visual words**".

Index

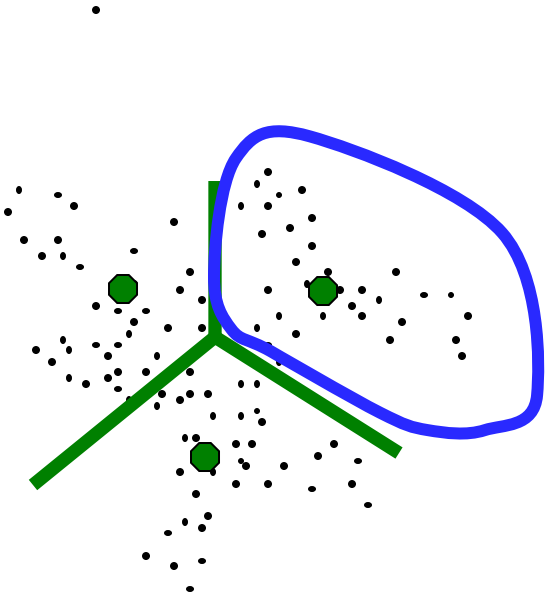
Visual Words

- Map descriptors to “words” by quantizing feature space
  - Quantize via clustering
    - Cluster centers are the prototype “words”
  - Determine which word to assign to each new image region by finding the closest cluster center.



Visual words

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



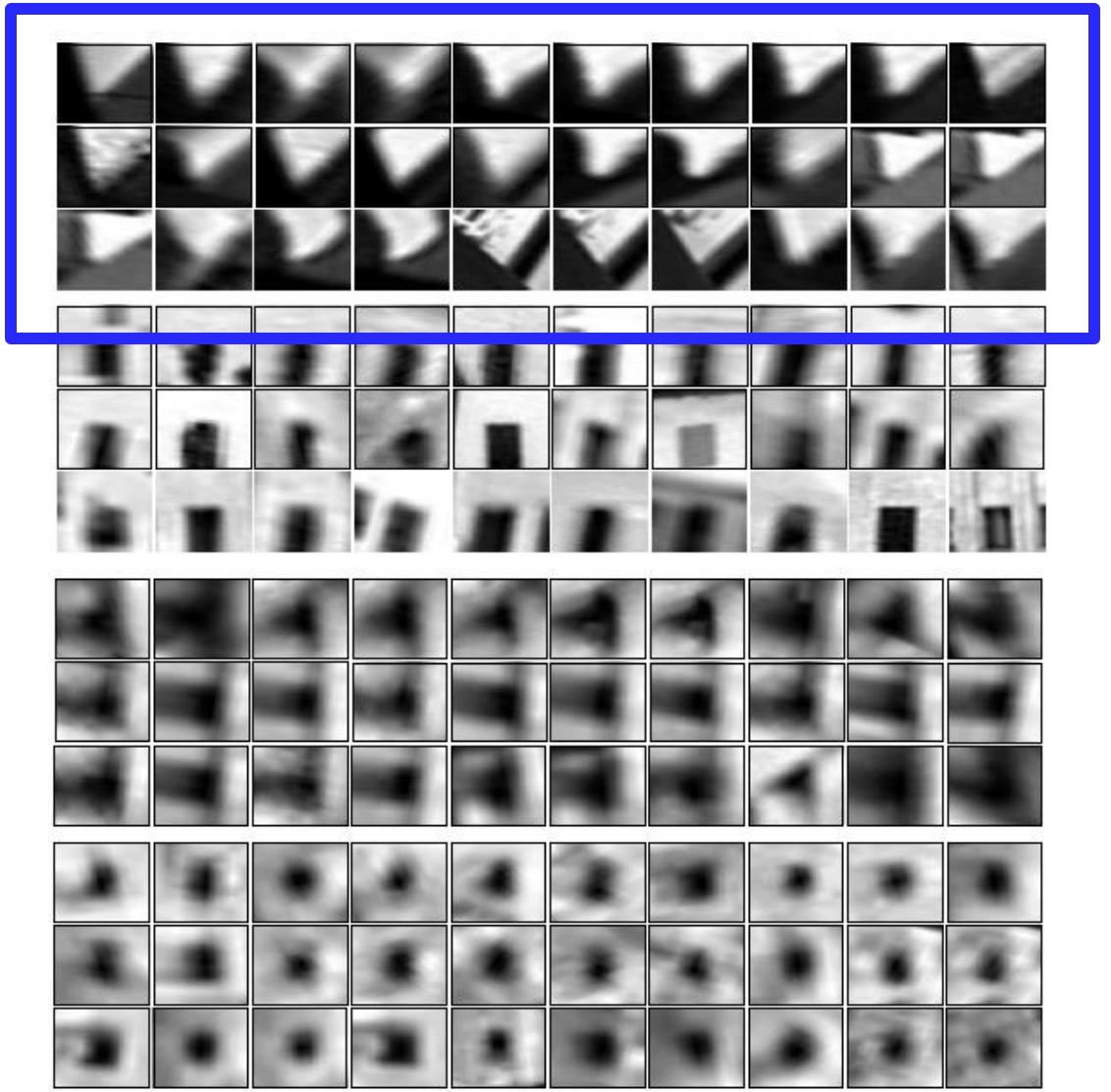
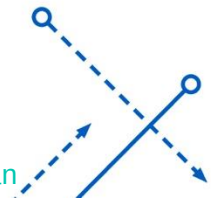


Figure from Sivic & Zisserman, ICCV 2003

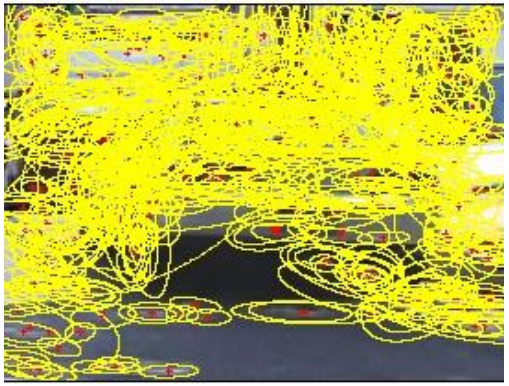
Visual vocabulary formation

Things need to consider:

- Sampling strategy: where to extract features?
- Clustering / quantization algorithm
- Unsupervised vs. supervised
- Features, vocabulary size, number of words?



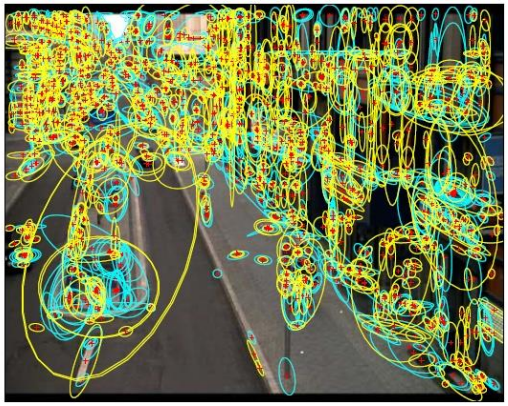
Sampling strategies



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

Inverted file index

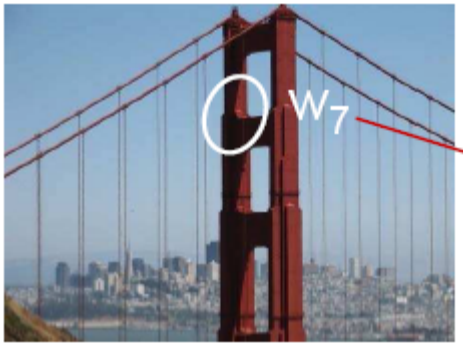
- Database images are loaded into the index mapping words to image numbers

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

…

Inverted file index

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



New query image

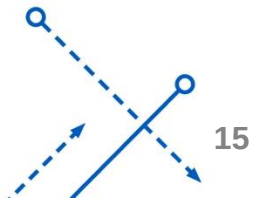
Word #	Image #
1	3
2	
…	
7	1, 2
8	3
9	
10	
…	
91	2

Image #1

Image #2

Inverted file index

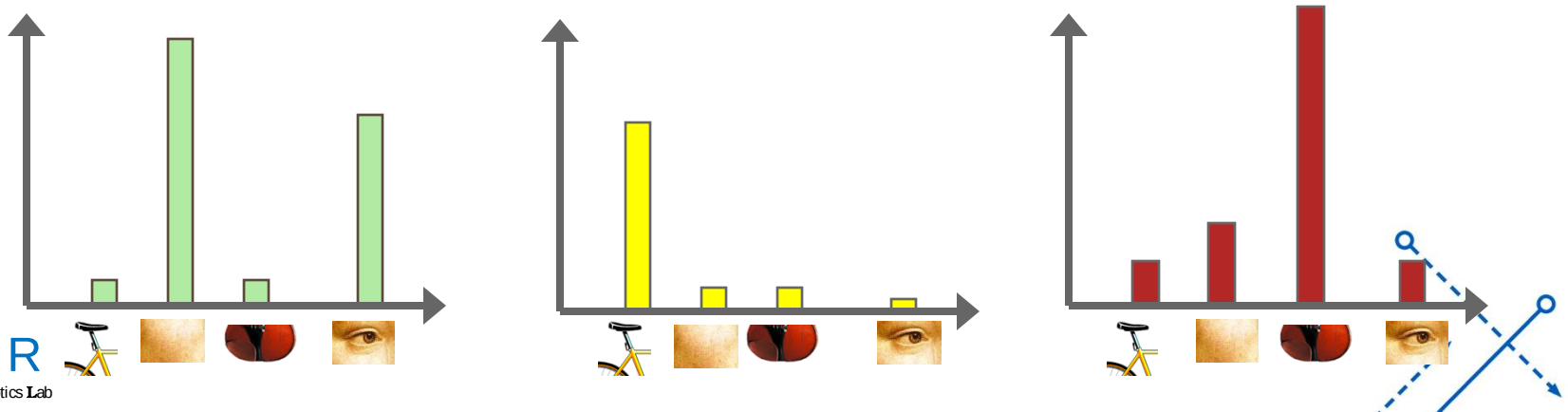
- Key requirement for inverted file index to be efficient:
  - Sparsity
  - If most pages/images contain most words, then it's no better than exhaustive search.
    - Exhaustive search would mean comparing the word distribution of a query versus every page.



Instance recognition: remaining issues

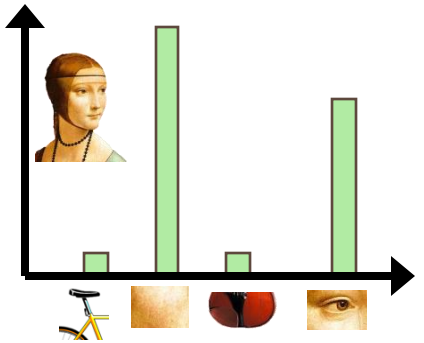
- How to summarize the content of an entire image? And estimate overall similarity?
- How large should the vocabulary be? How to perform quantization efficiently?
- Is having the same set of visual words enough to identify the object/scene? How to verify spatial agreement?
- How to score the retrieval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

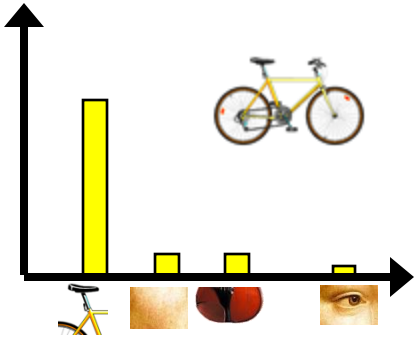
Bags of visual words (Recap)

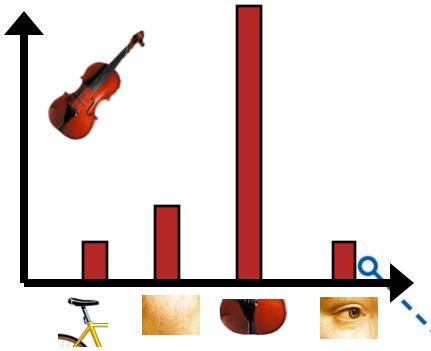


Bags of visual words (Recap)

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



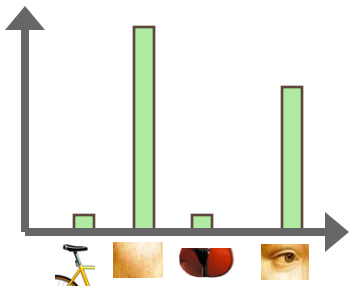




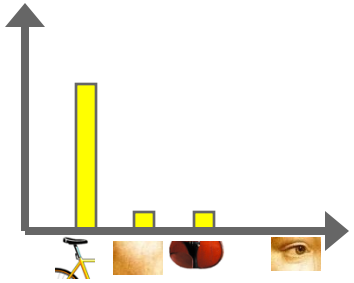
Comparing bags of words (Recap)

- Rank frames by normalized inner product between their (possibly weighted) occurrence counts---*nearest neighbor* search for similar images.

[1 8 1 4]

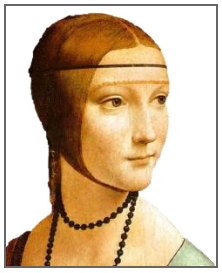


[5 1 1 0]



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

for vocabulary of *V* words

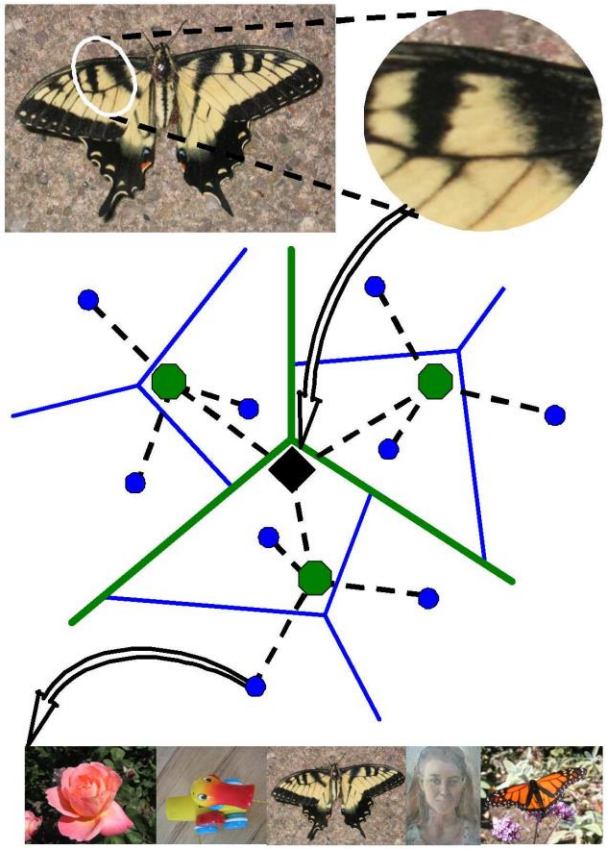


$\vec{d}_j$

$\vec{q}$

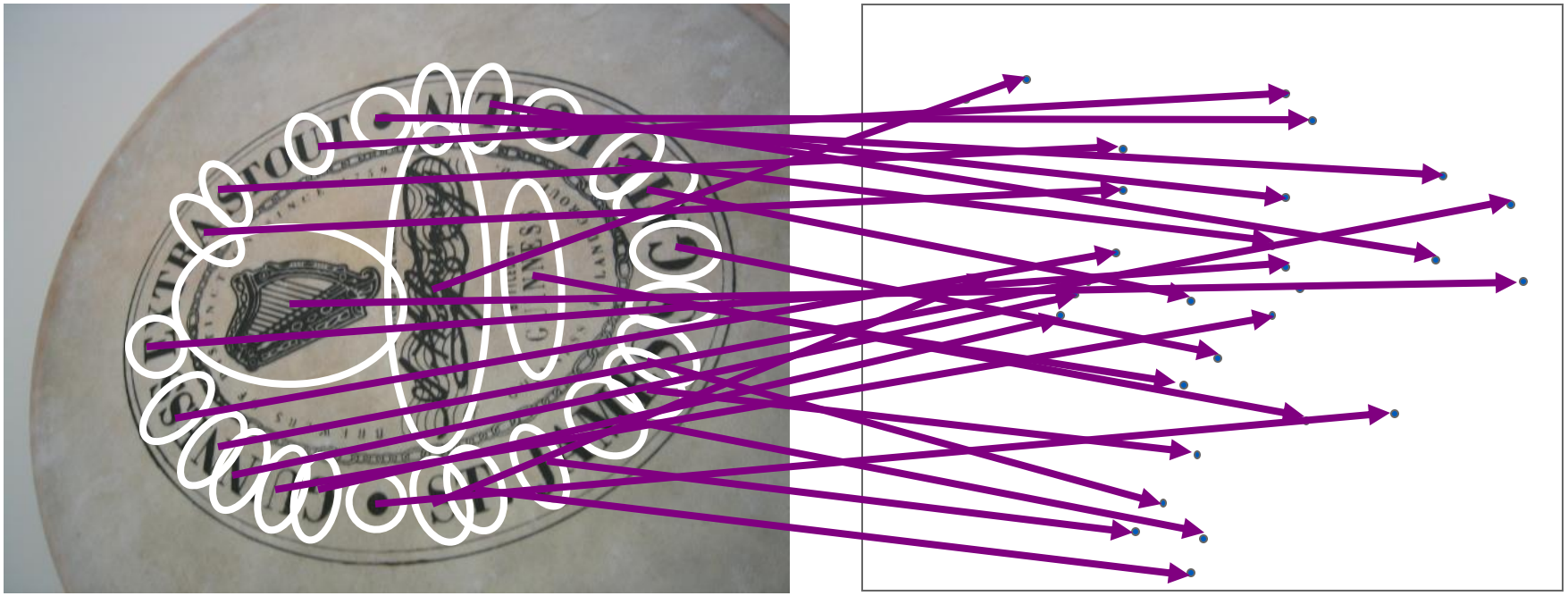
Visual vocabularies: Issues

- How to choose vocabulary size?
  - Too small: visual words not representative of all patches
  - Too large: quantization artifacts, overfitting
- Computational efficiency
  - **Vocabulary trees**



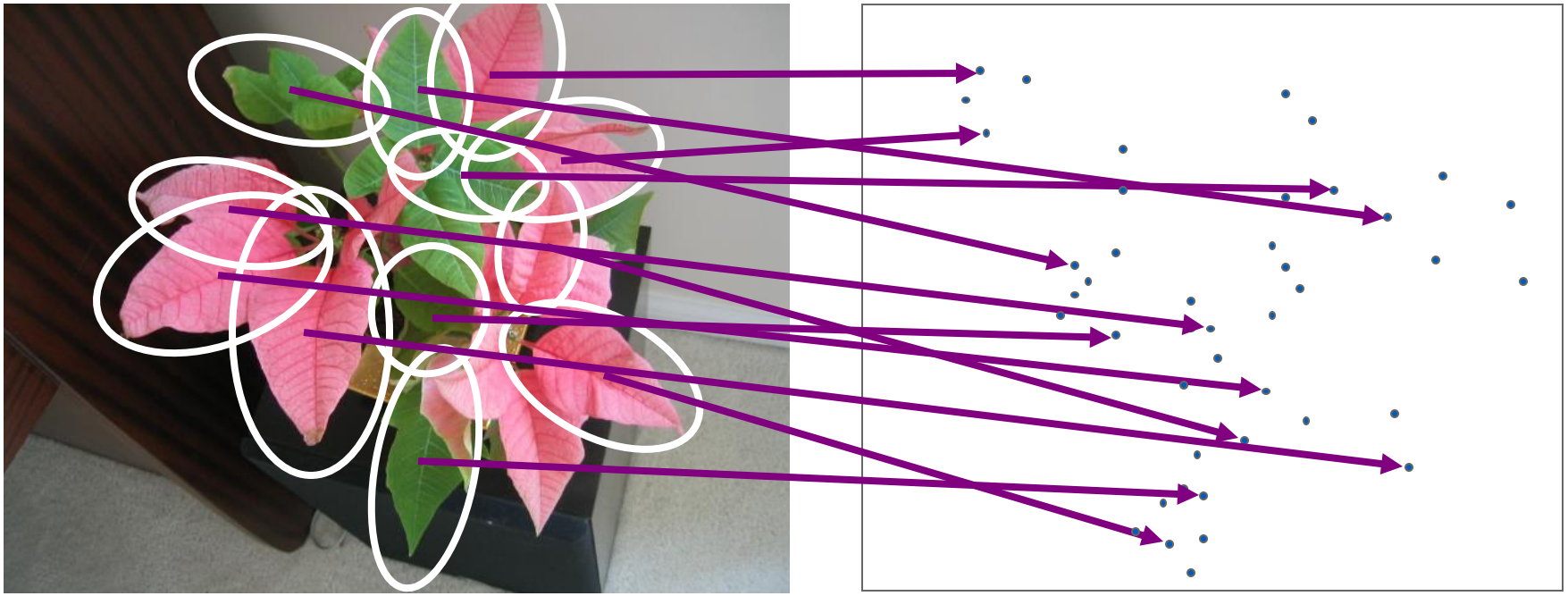
(Nister & Stewenius, 2006)

Recognition with K-d-tree



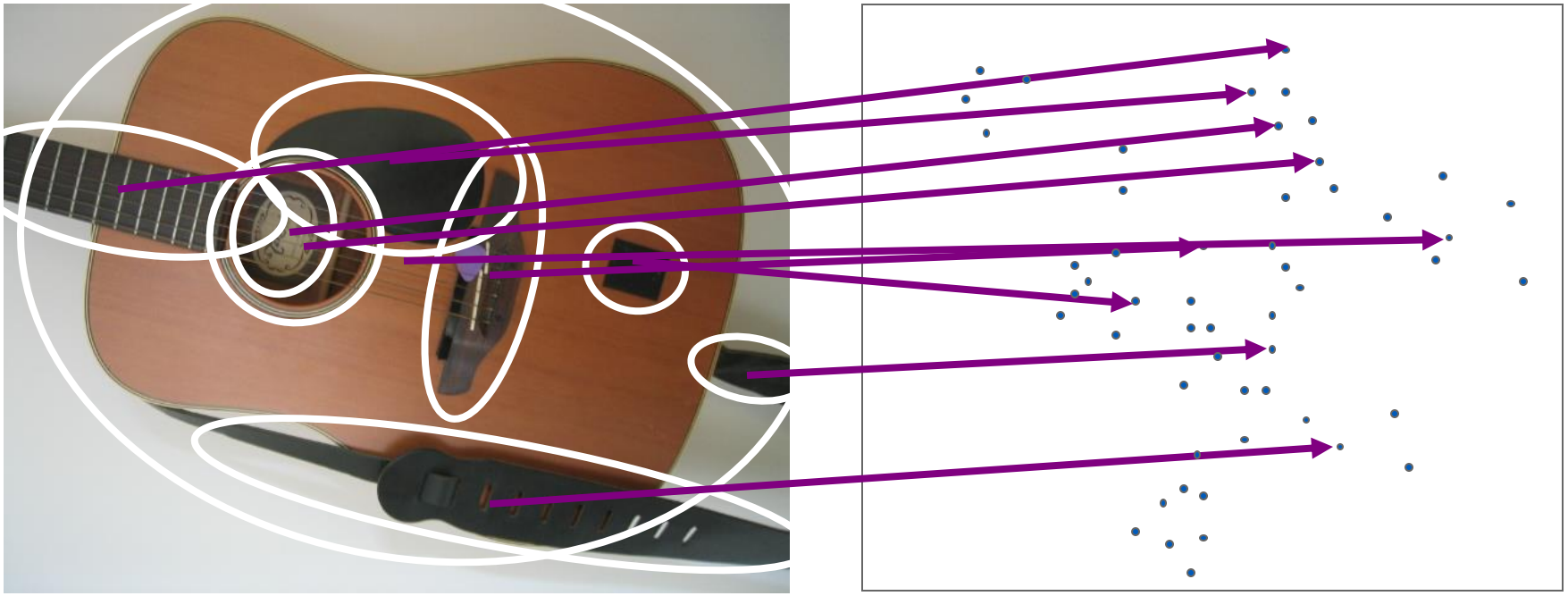
Following slides by David Nister (CVPR 2006)

Recognition with K-d-tree



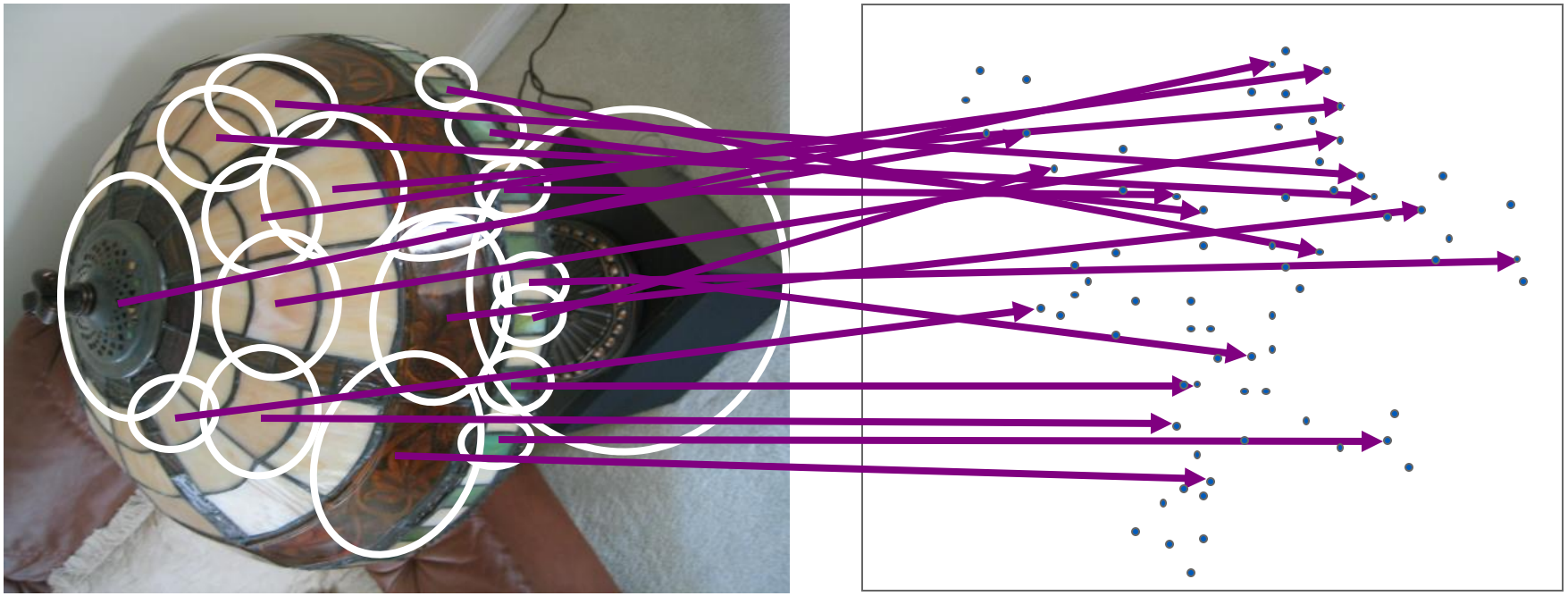
Following slides by David Nister (CVPR 2006)

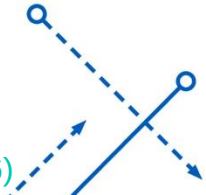
Recognition with K-d-tree



Following slides by David Nister (CVPR 2006)

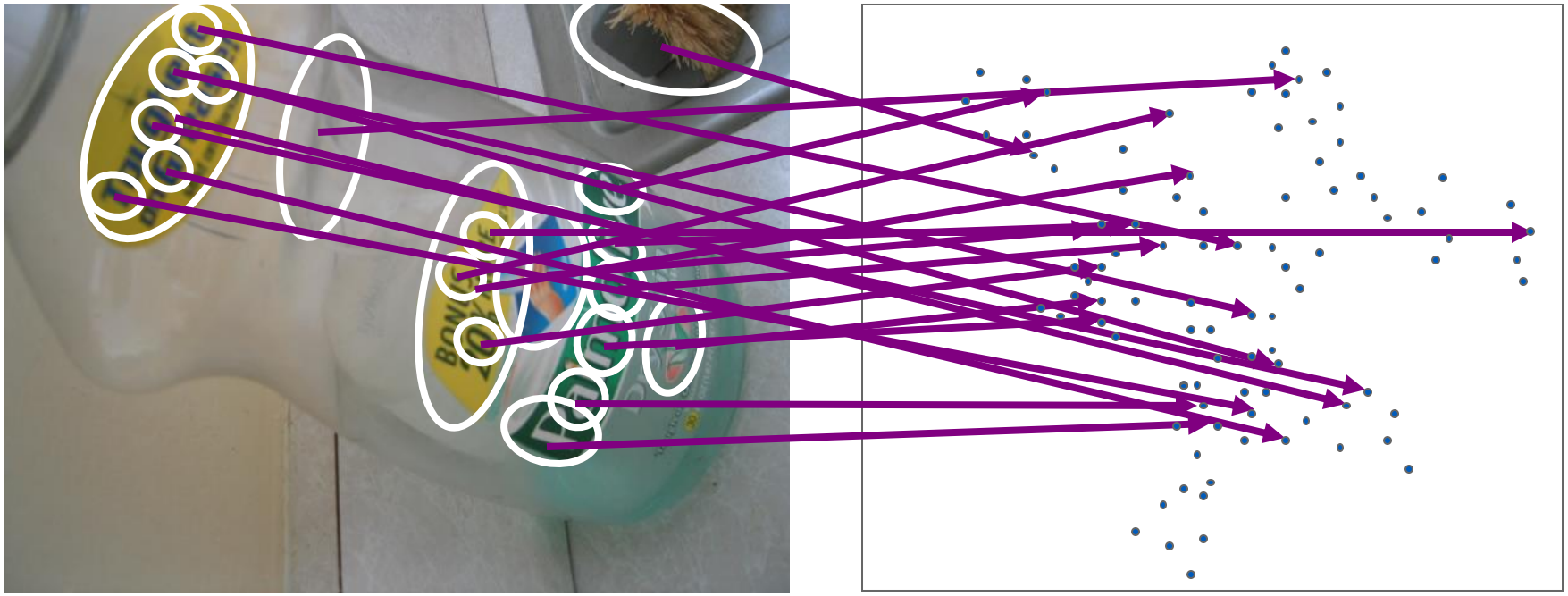
Recognition with K-d-tree

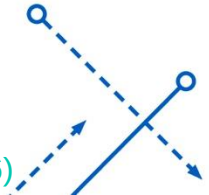




Following slides by David Nister (CVPR 2006)

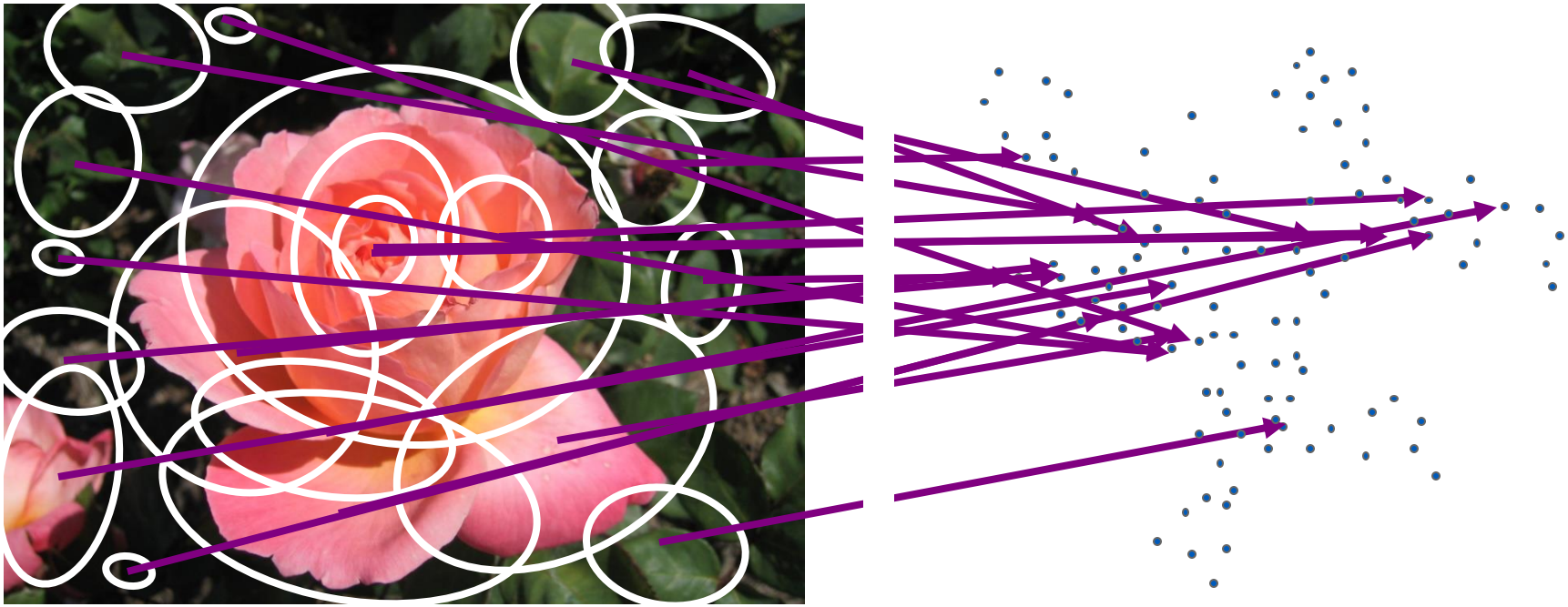
Recognition with K-d-tree

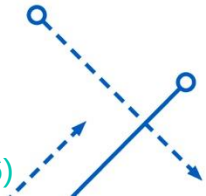




Following slides by David Nister (CVPR 2006)

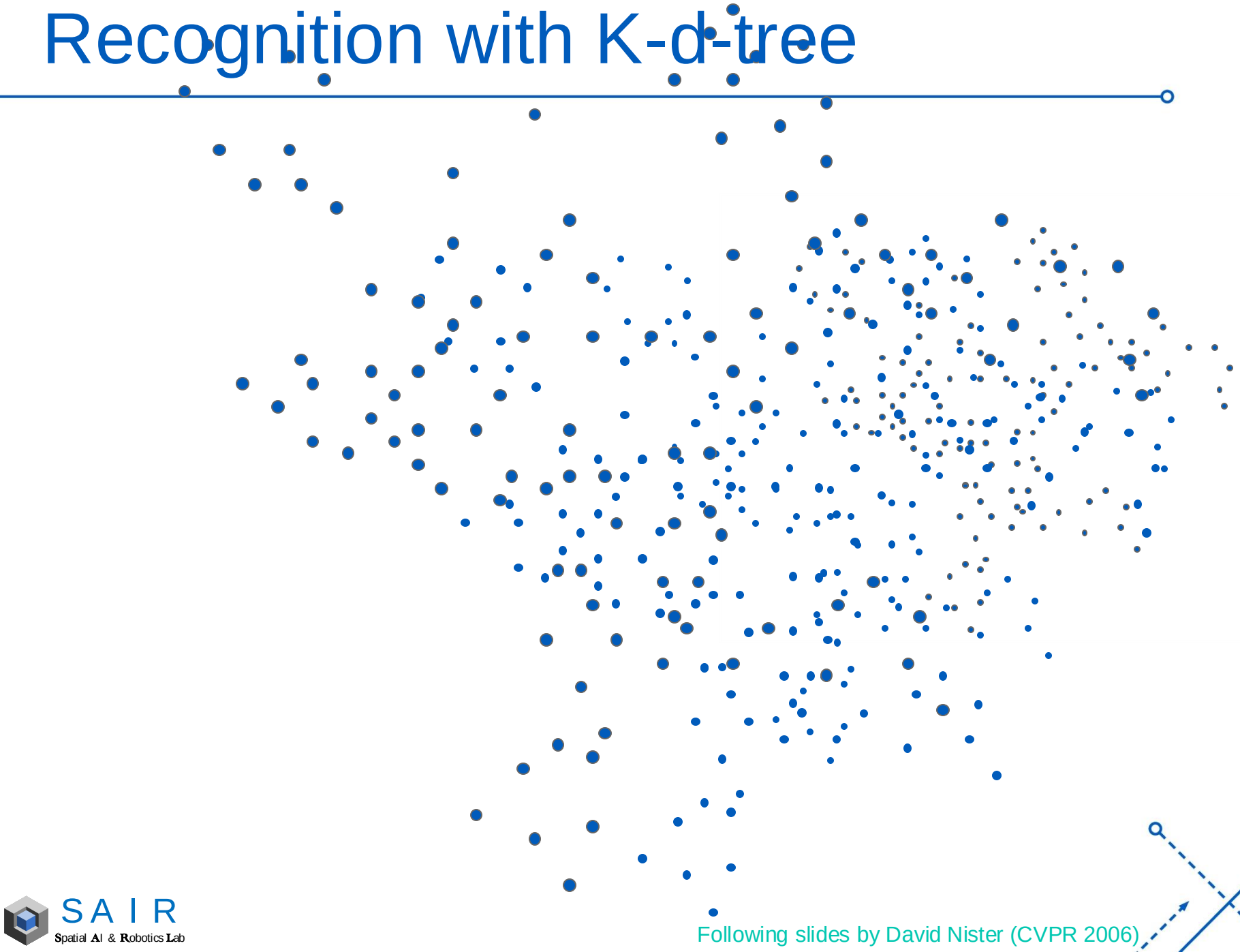
Recognition with K-d-tree





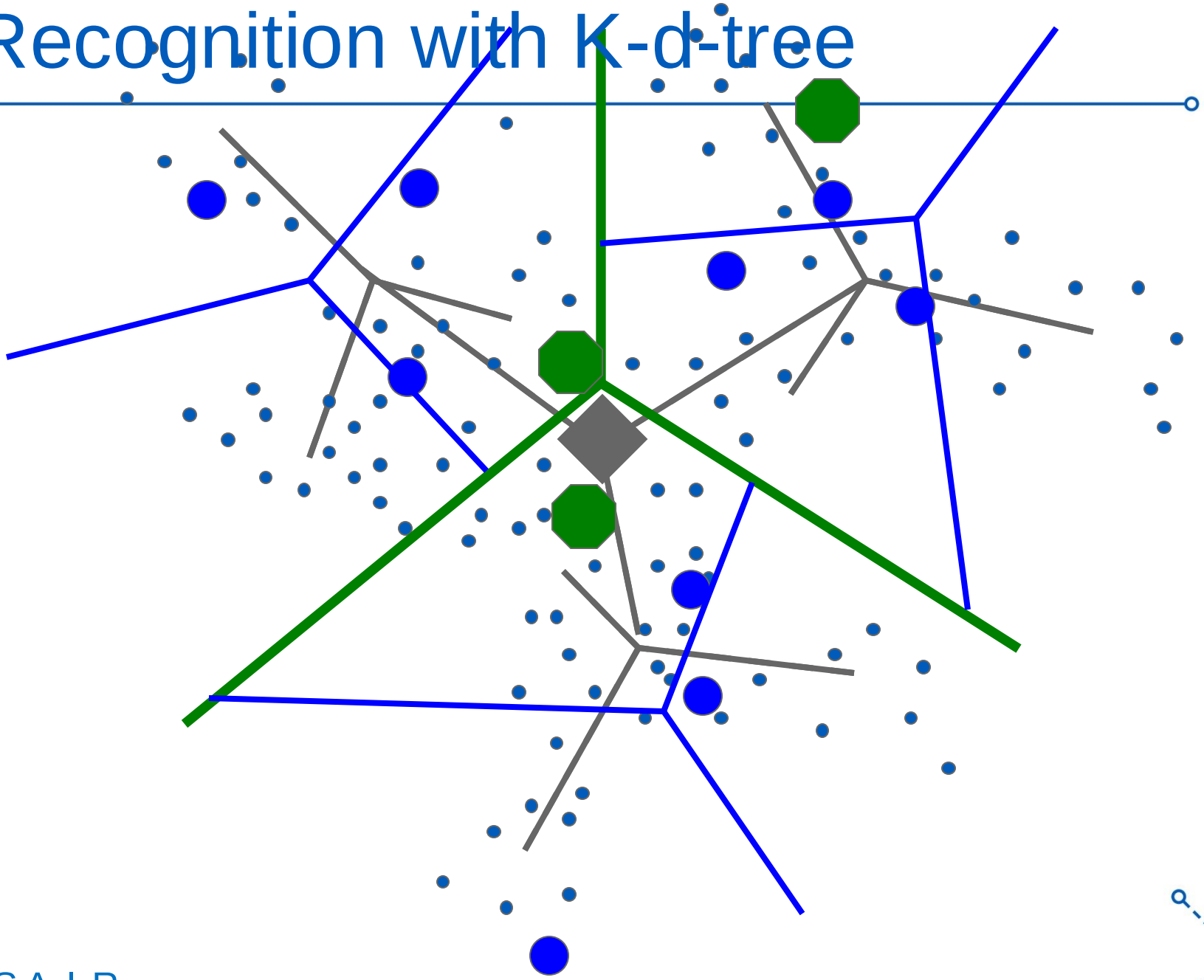
Following slides by David Nister (CVPR 2006)

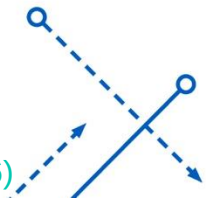
Recognition with K-d-tree



Following slides by David Nister (CVPR 2006)

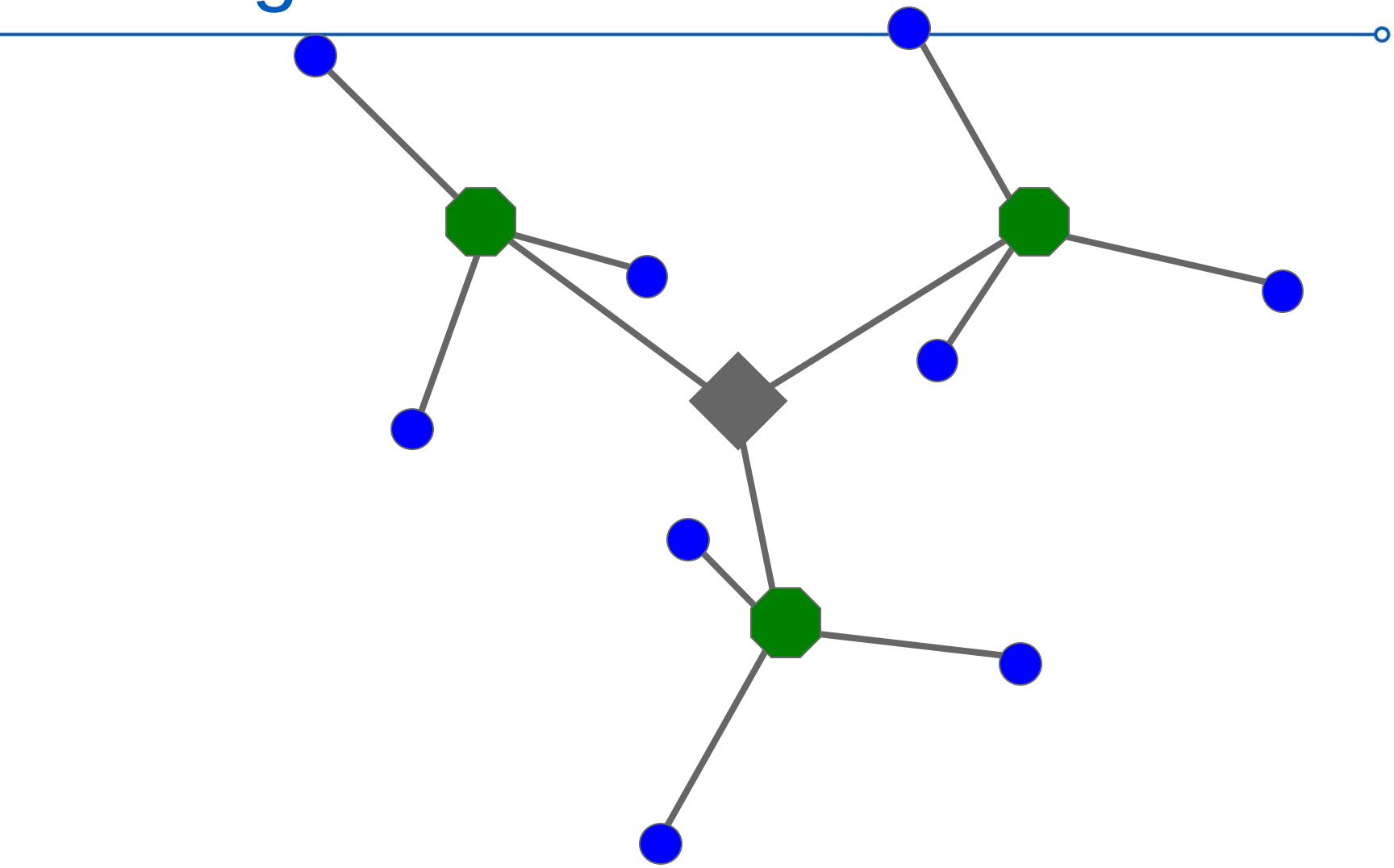
Recognition with K-d-tree





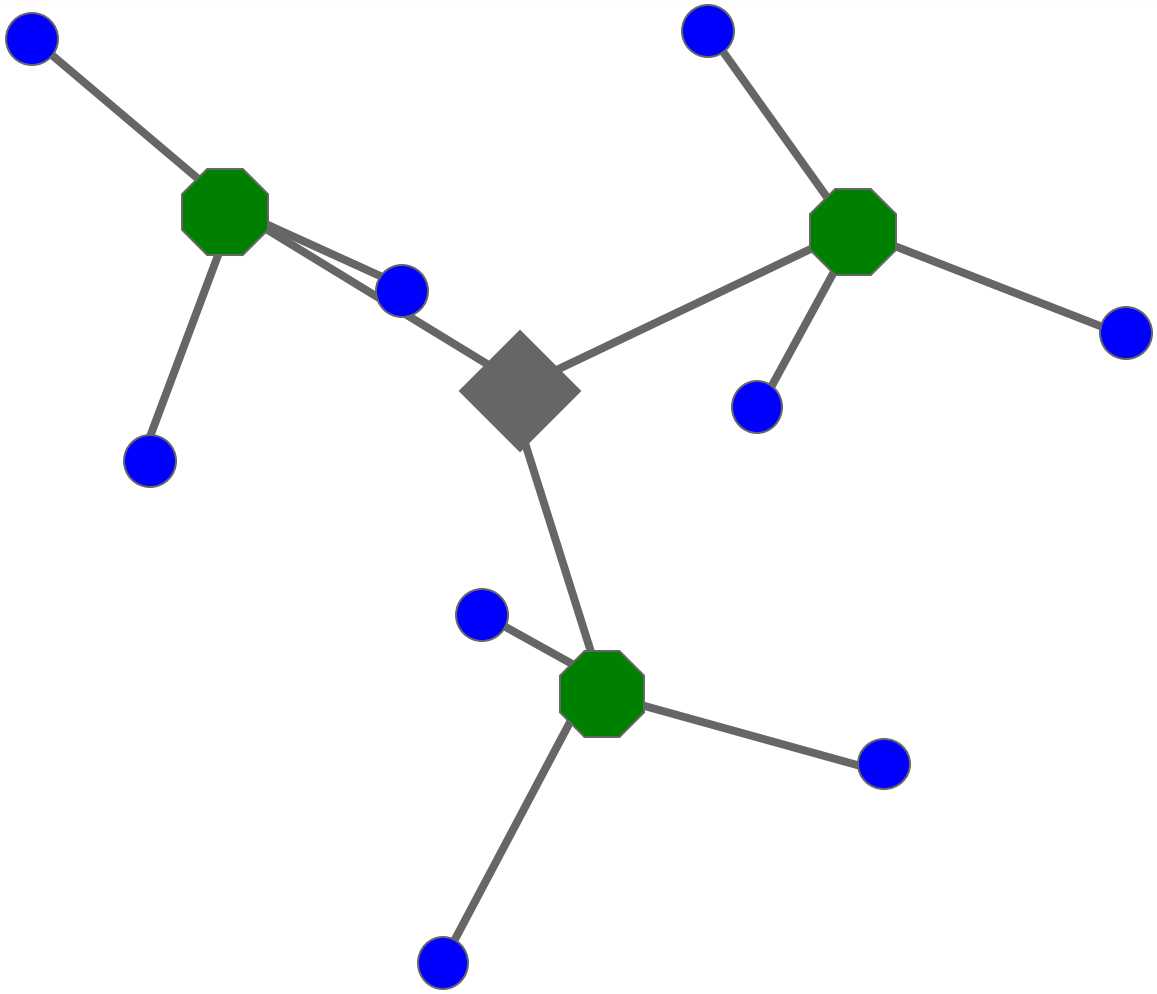
Following slides by David Nister (CVPR 2006)

Recognition with K-d-tree



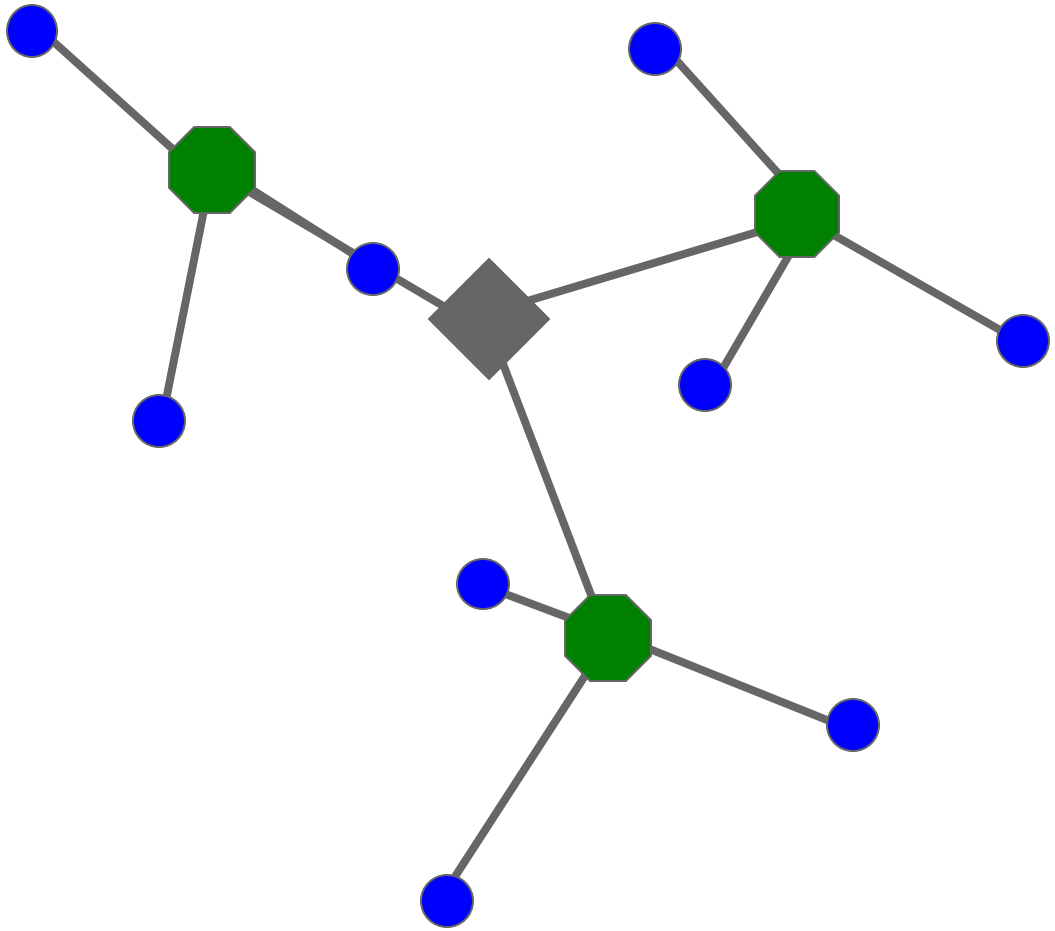
Following slides by David Nister (CVPR 2006)

Recognition with K-d-tree

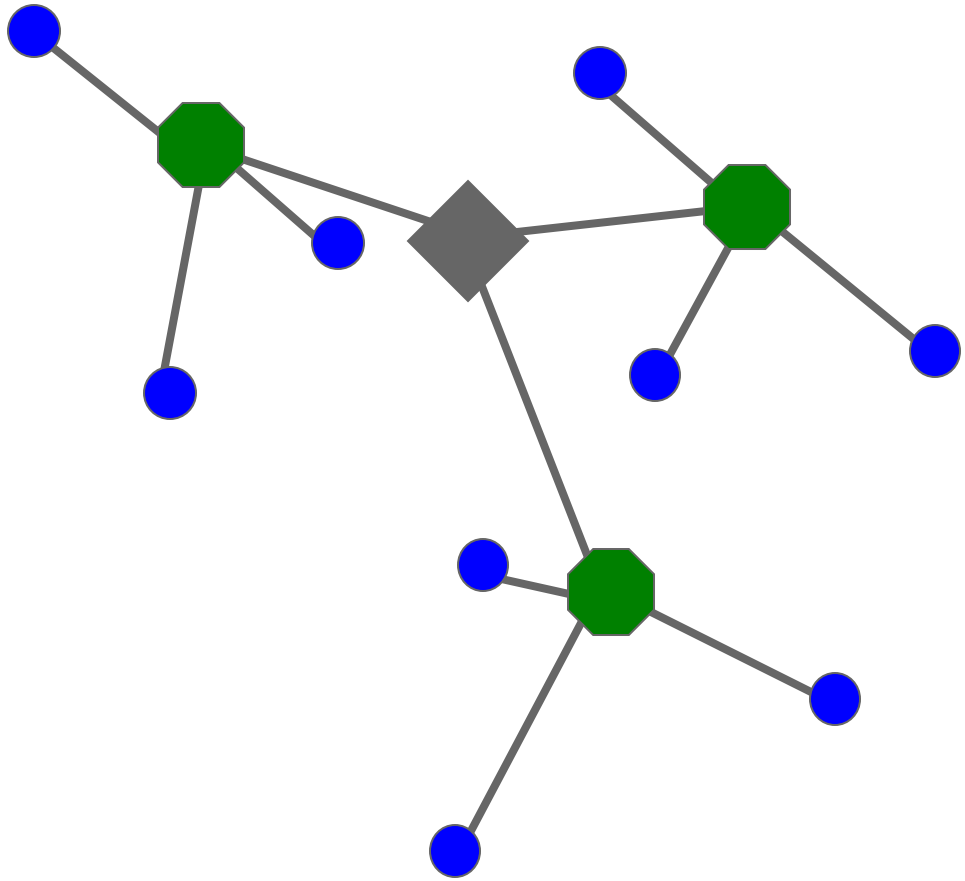


Following slides by David Nister (CVPR 2006)

Recognition with K-d-tree

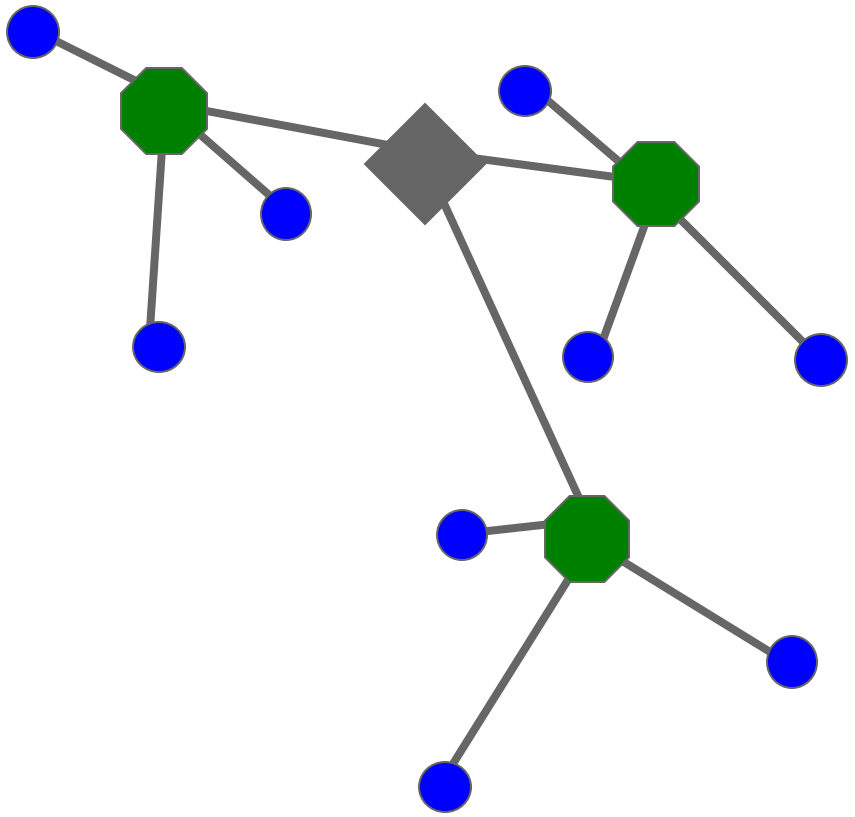


Recognition with K-d-tree



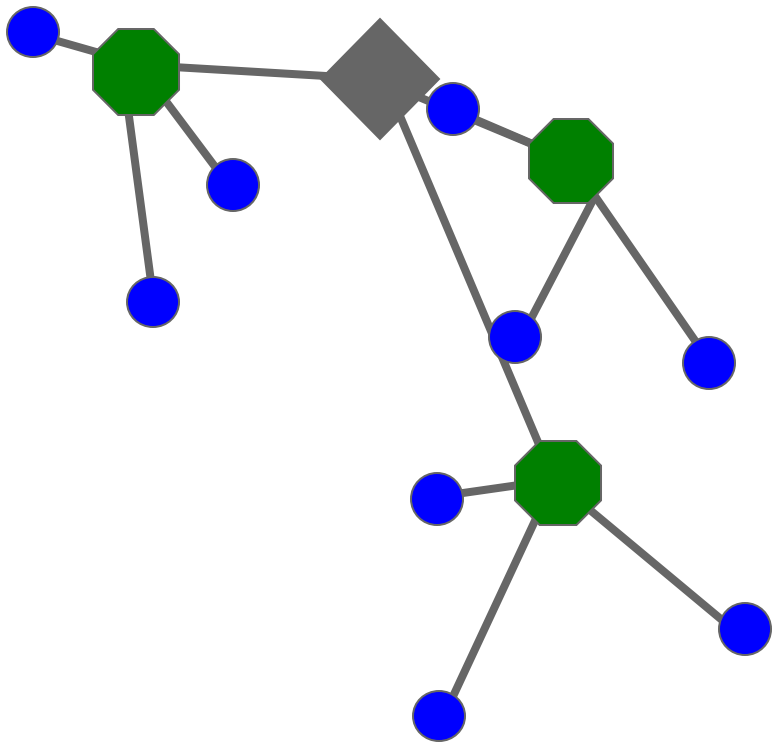
Following slides by David Nister (CVPR 2006)

Recognition with K-d-tree



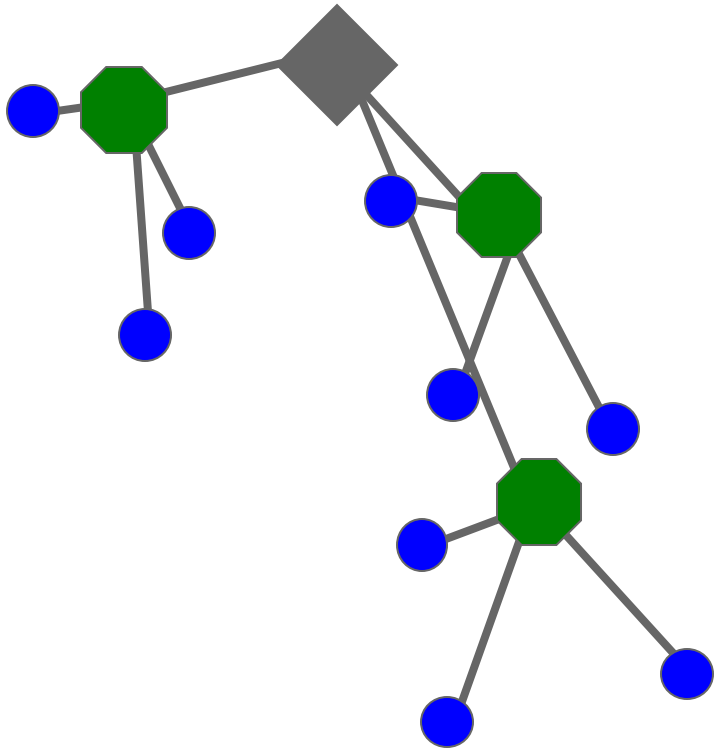
Following slides by David Nister (CVPR 2006)

Recognition with K-d-tree



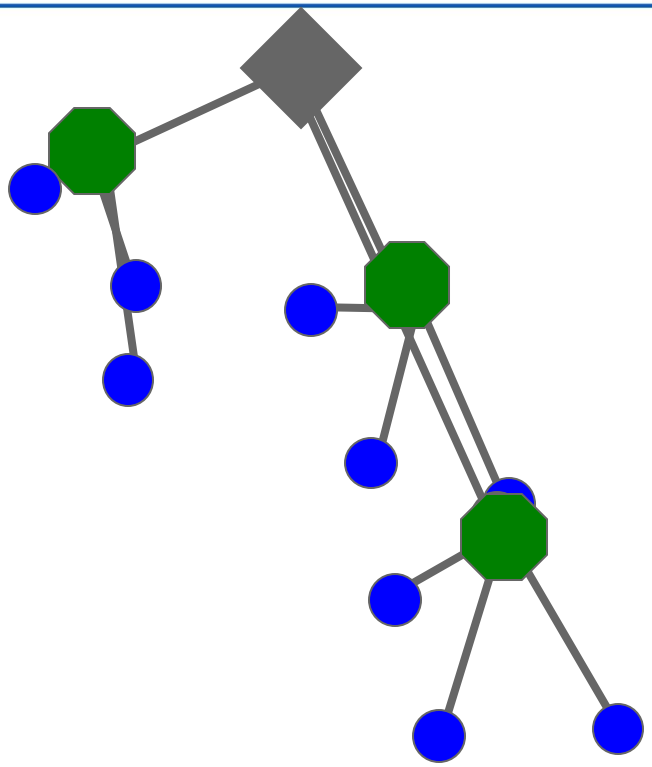
Following slides by David Nister (CVPR 2006)

Recognition with K-d-tree



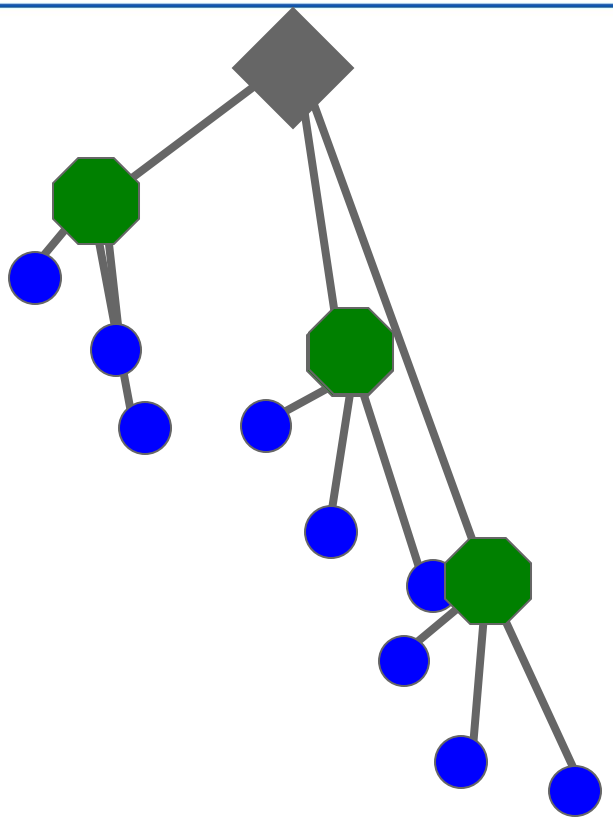
Following slides by David Nister (CVPR 2006)

Recognition with K-d-tree



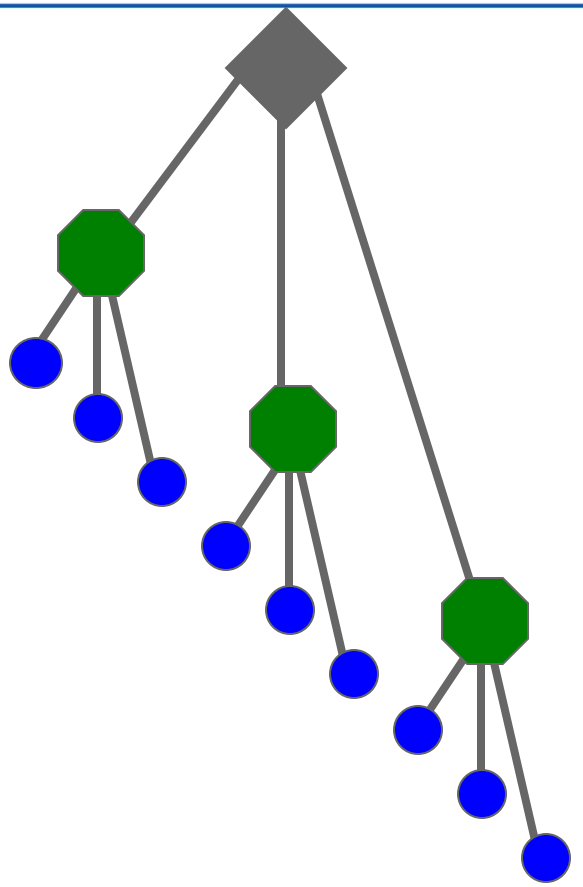
Following slides by David Nister (CVPR 2006)

Recognition with K-d-tree



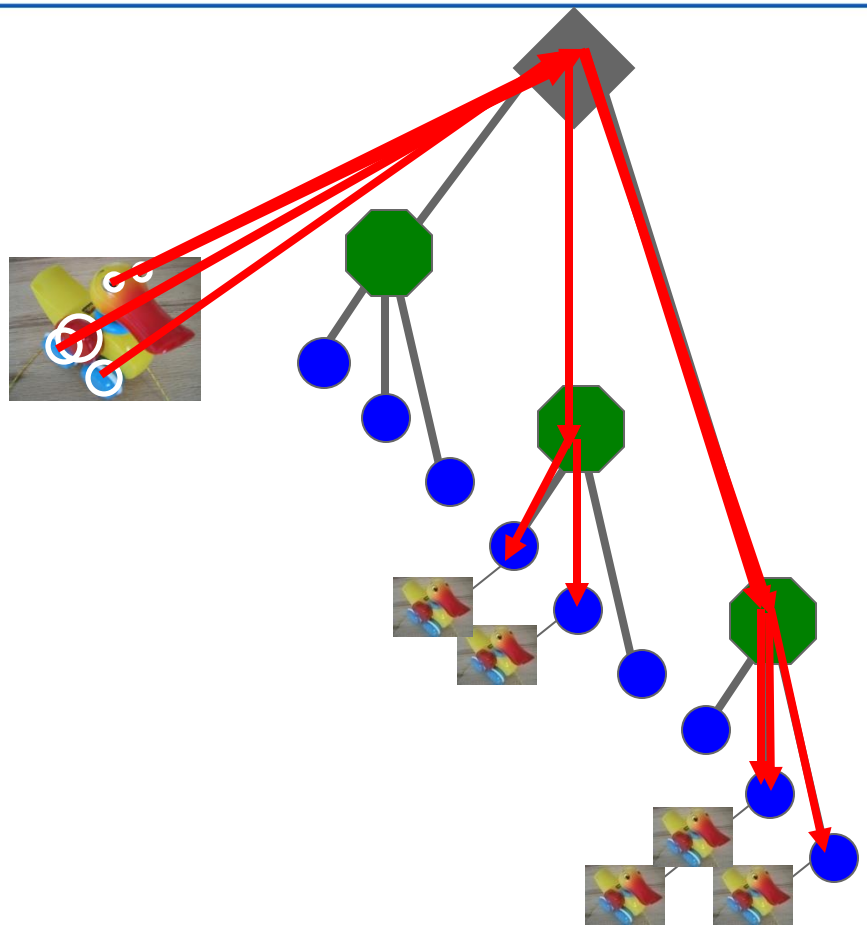
Following slides by David Nister (CVPR 2006)

Recognition with K-d-tree



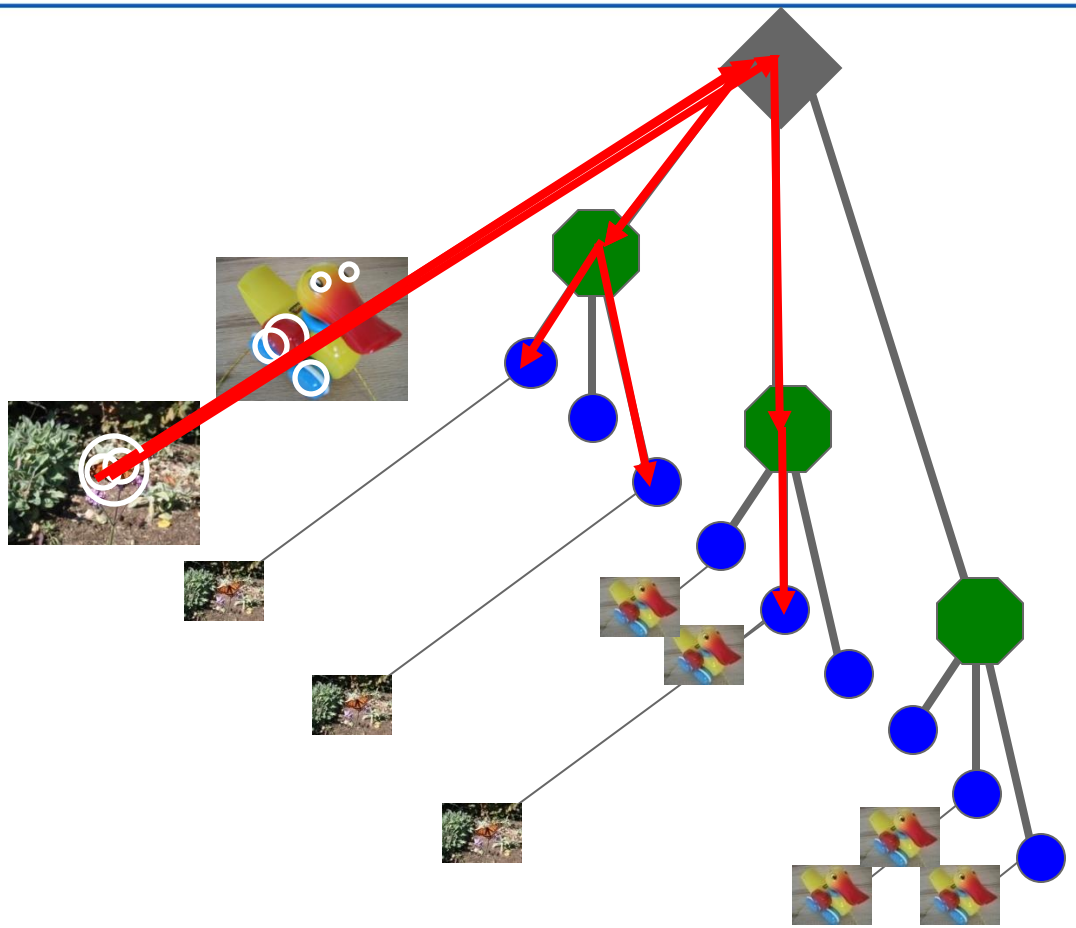
Following slides by David Nister (CVPR 2006)

Recognition with K-d-tree



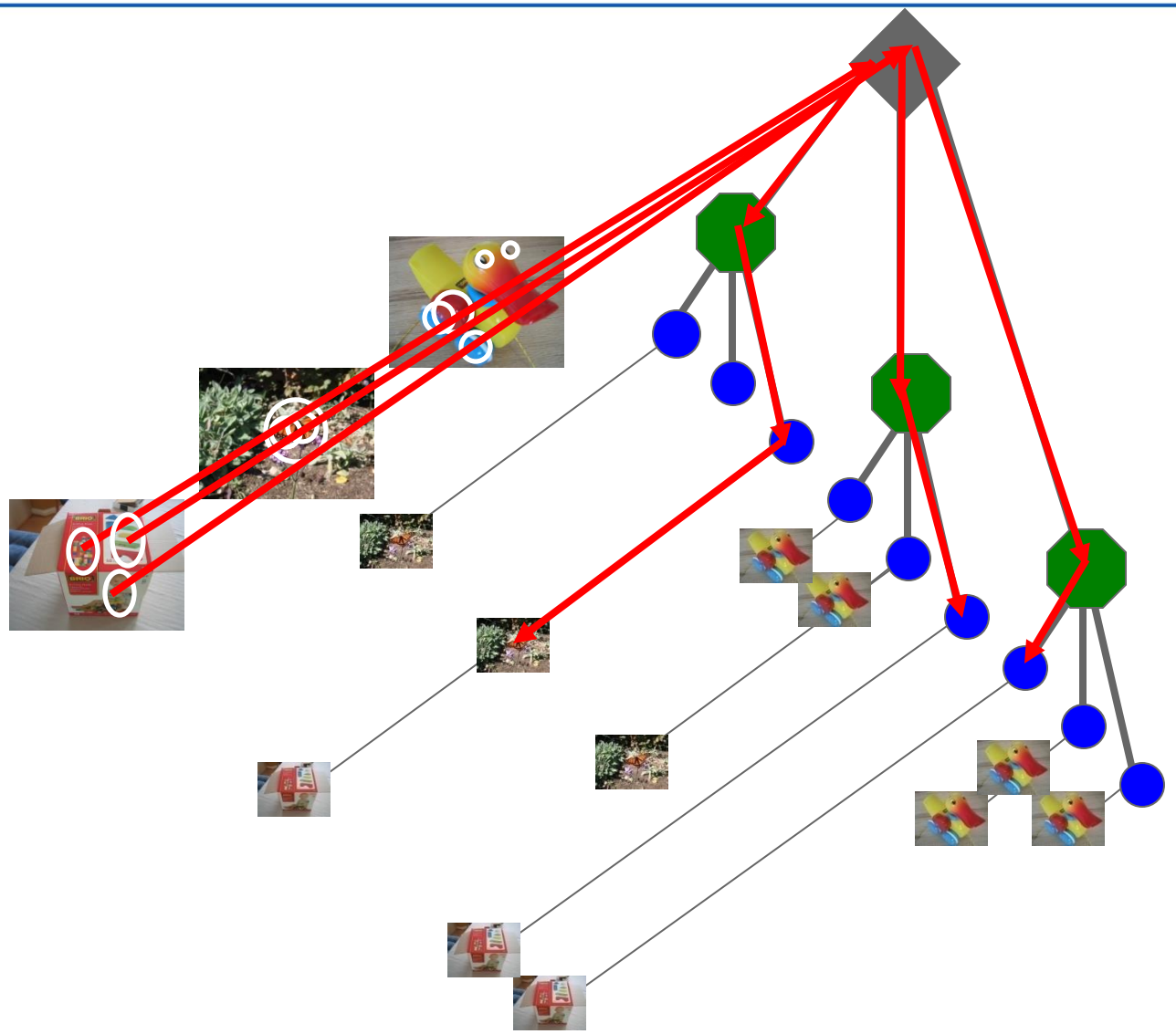
Following slides by David Nister (CVPR 2006)

Recognition with K-d-tree



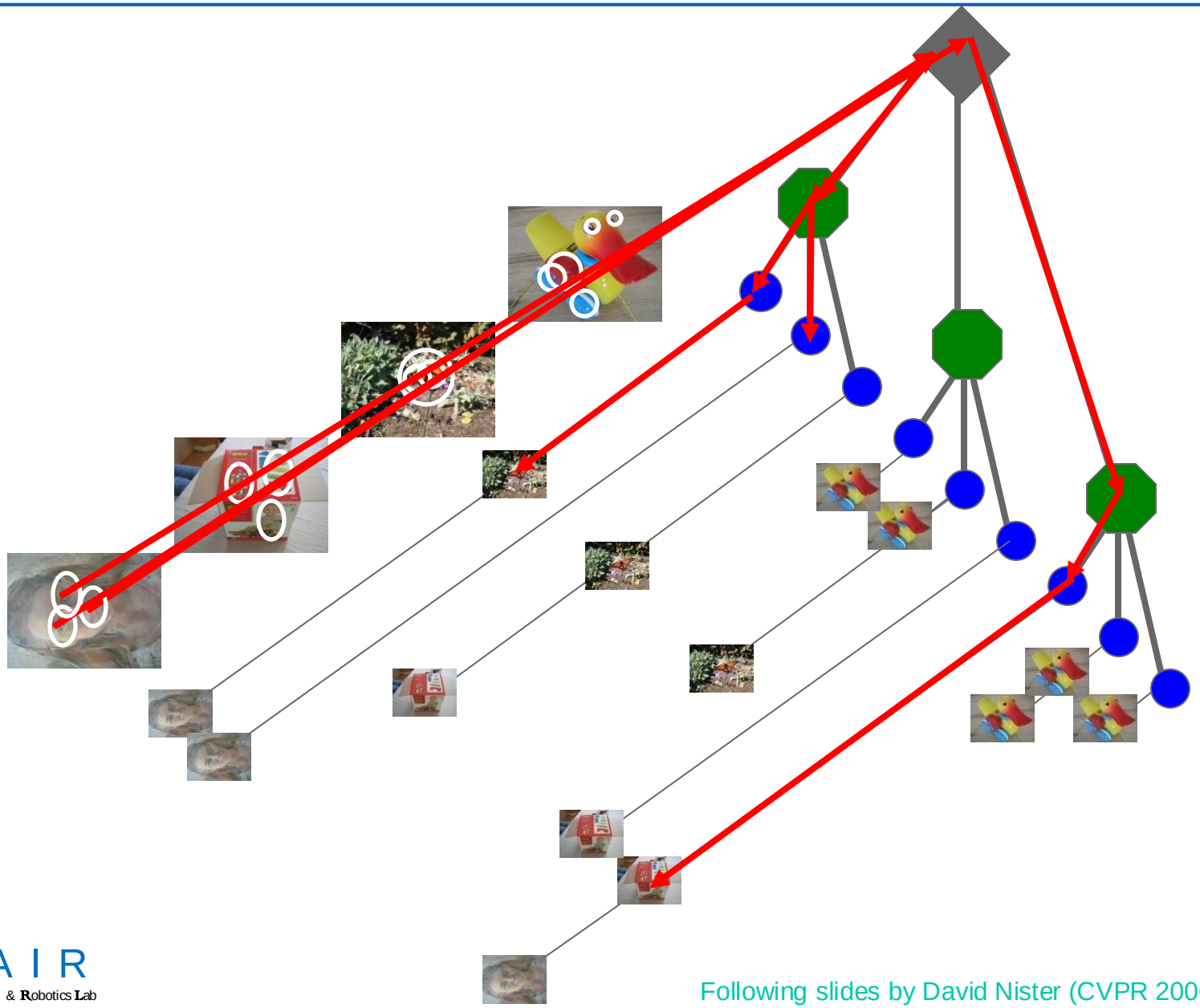
Following slides by David Nister (CVPR 2006)

Recognition with K-d-tree



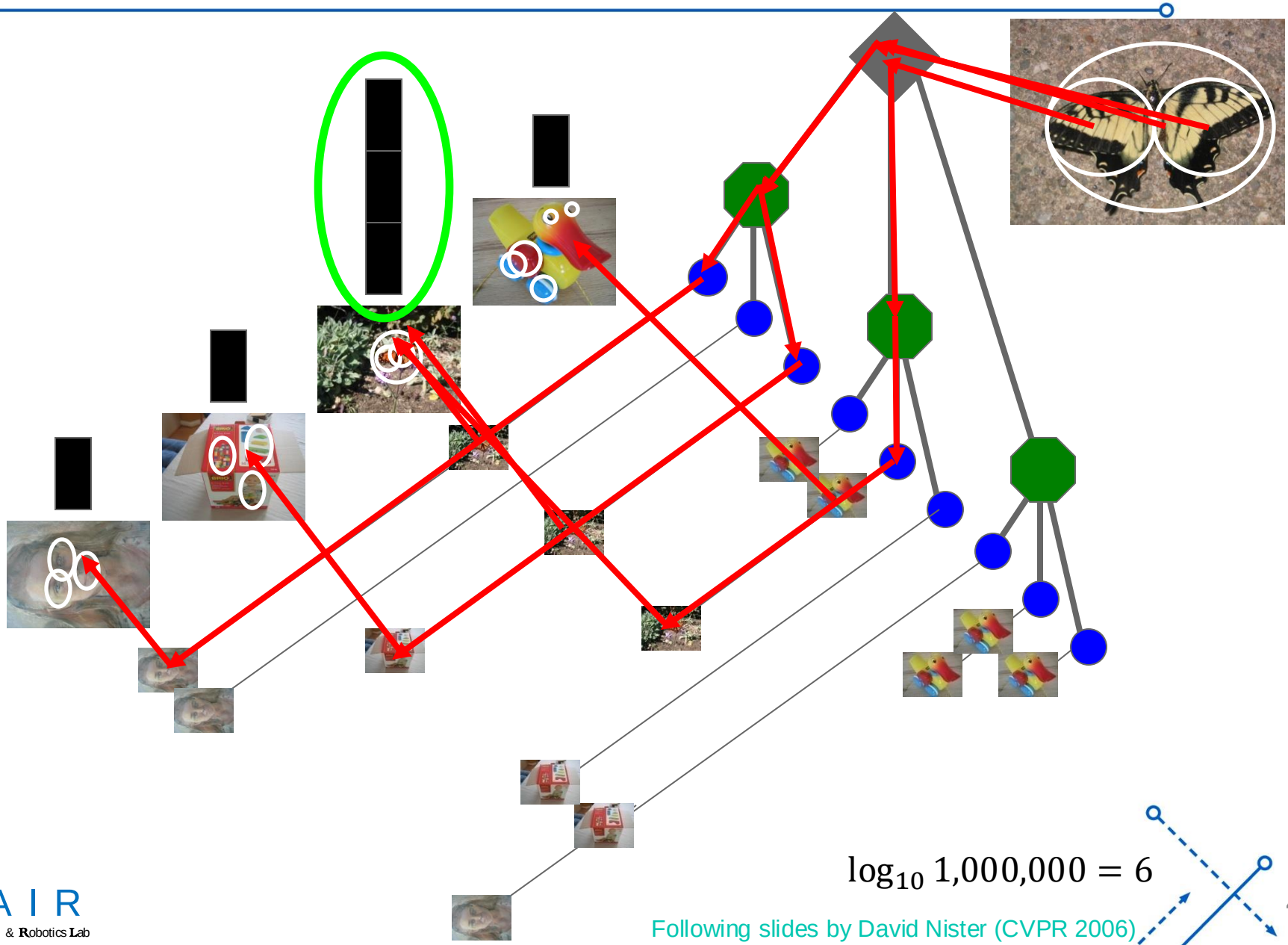
Following slides by David Nister (CVPR 2006)

Recognition with K-d-tree



Following slides by David Nister (CVPR 2006)

Recognition with K-d-tree



$$\log_{10} 1{,}000{,}000 = 6$$

Following slides by David Nister (CVPR 2006)

Vocabulary Tree: Performance

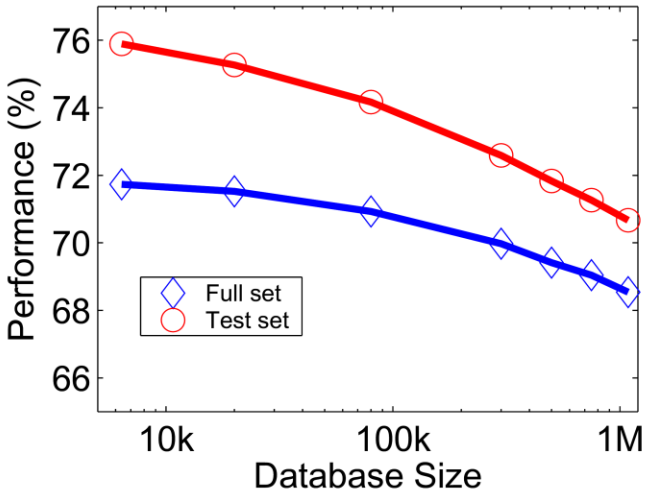
Evaluated on large databases

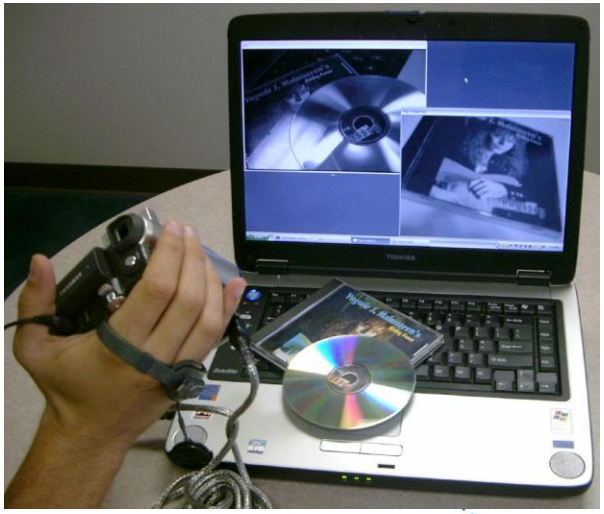
- Indexing with up to 1M images

Online recognition for database of 50,000 CD covers

- Retrieval in ~1s

Find experimentally that large vocabularies can be beneficial for recognition





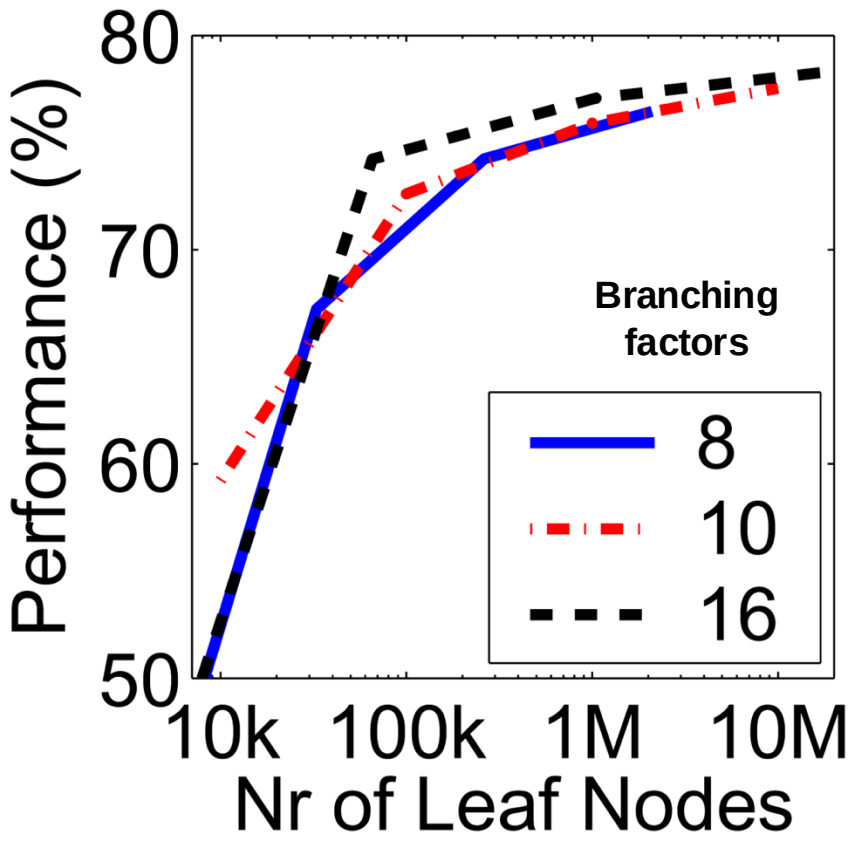
[Nister & Stewenius, CVPR'06]

Slide credit: J. Sivic

Instance recognition: remaining issues

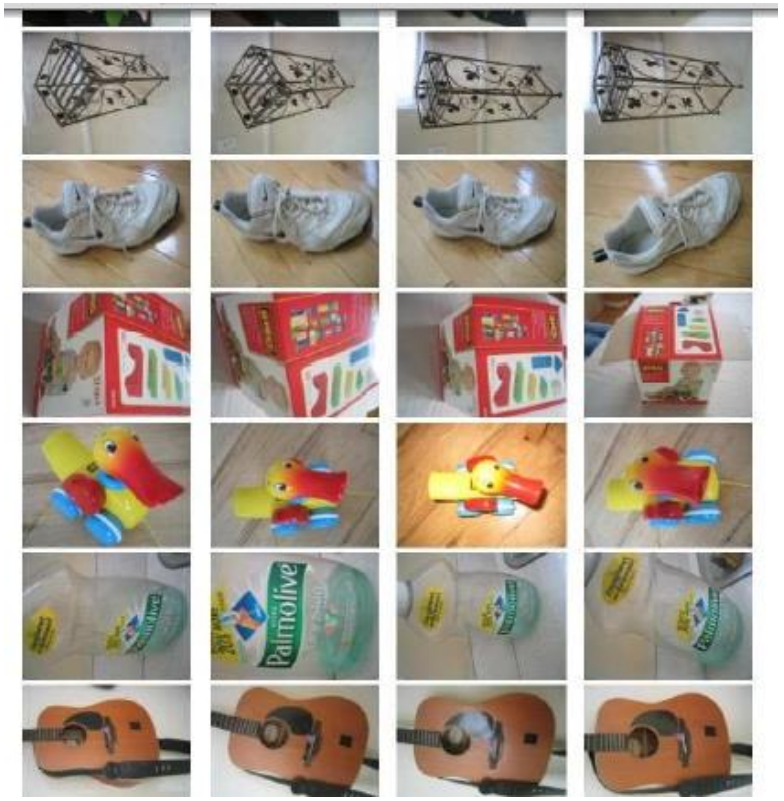
- How to summarize the content of an entire image? And estimate overall similarity?
- How large should the vocabulary be? How to perform quantization efficiently?
- Is having the same set of visual words enough to identify the object/scene? How to verify spatial agreement?
- How to score the retrieval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

Vocabulary size



Influence on performance, sparsity

Results for recognition task with 6347 images



Nister & Stewenius, CVPR 2006

Visual words/bags of words

Pro

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

Cons

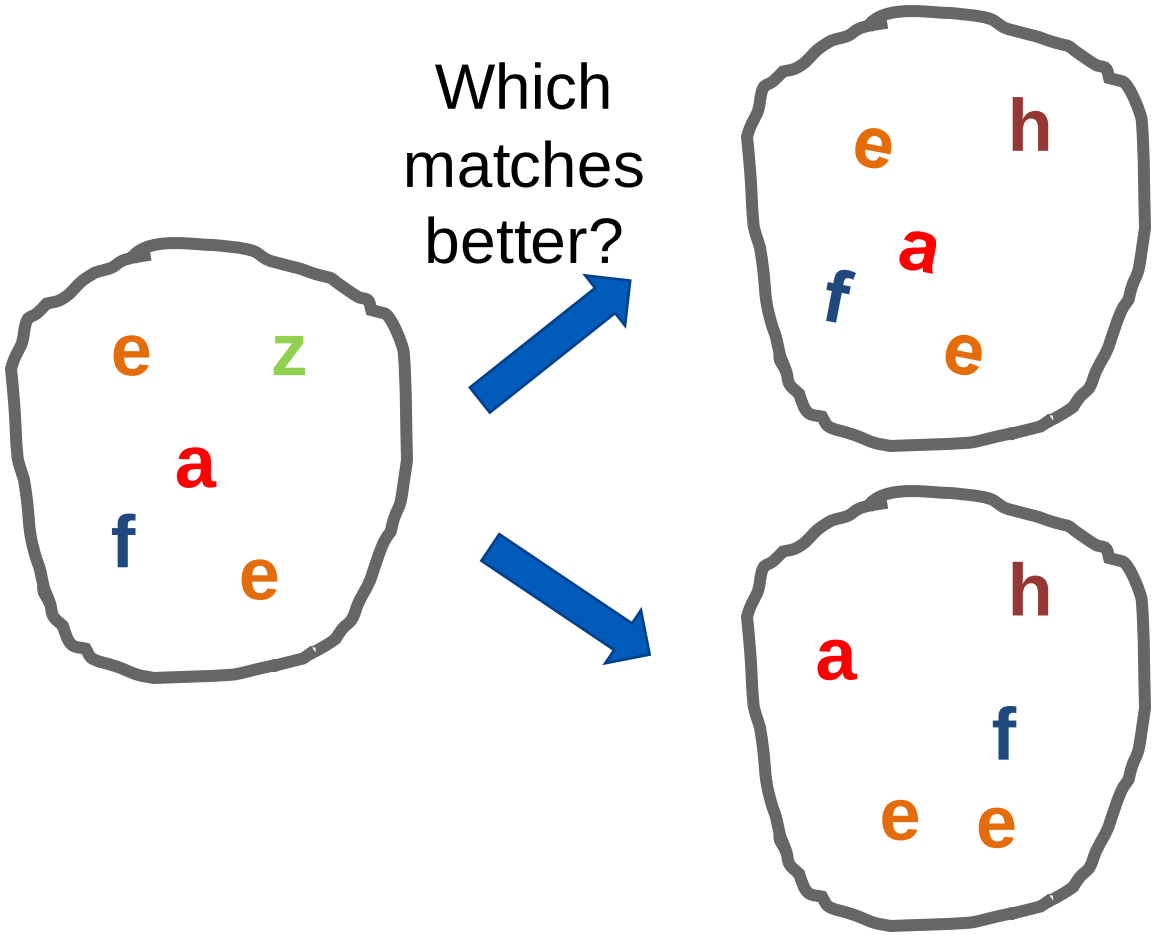
- background and foreground mixed when bag covers whole image
- optimal vocabulary formation remains unclear
- basic model ignores geometry – must verify afterwards, or encode via features

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?

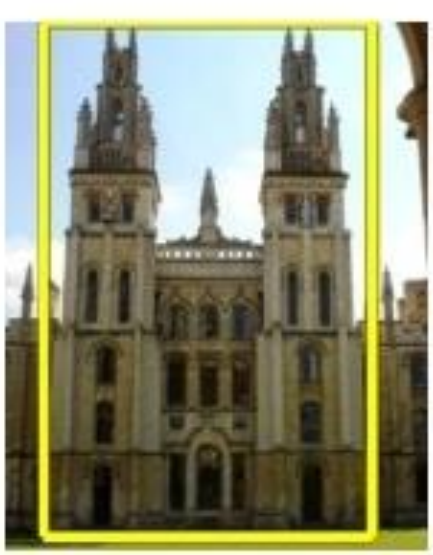
Can we be more accurate?

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



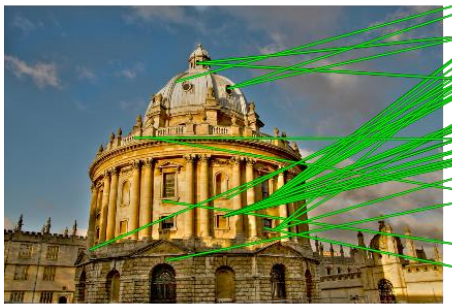
Real objects have consistent geometry

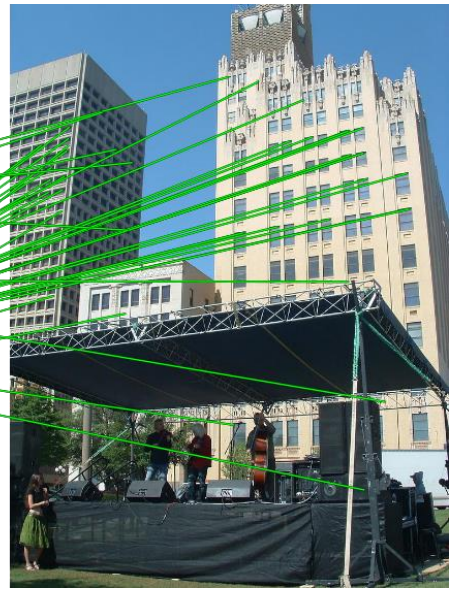
Spatial Verification

Query

DB image with high BoW similarity

Query



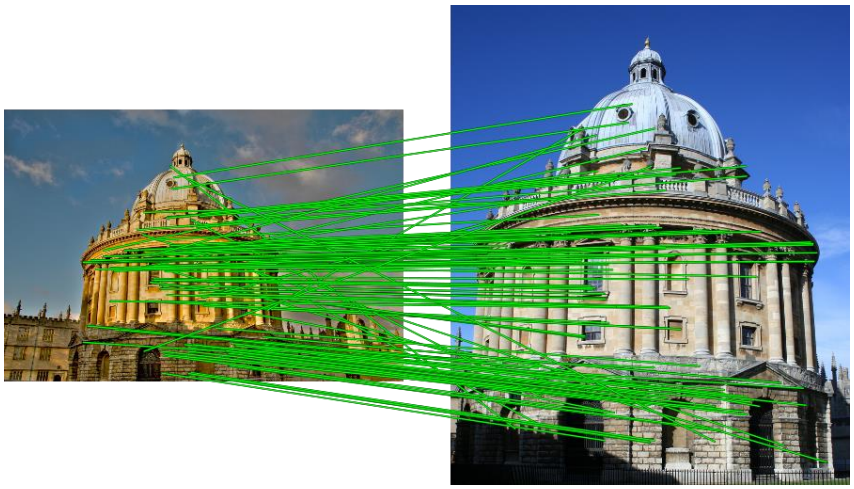


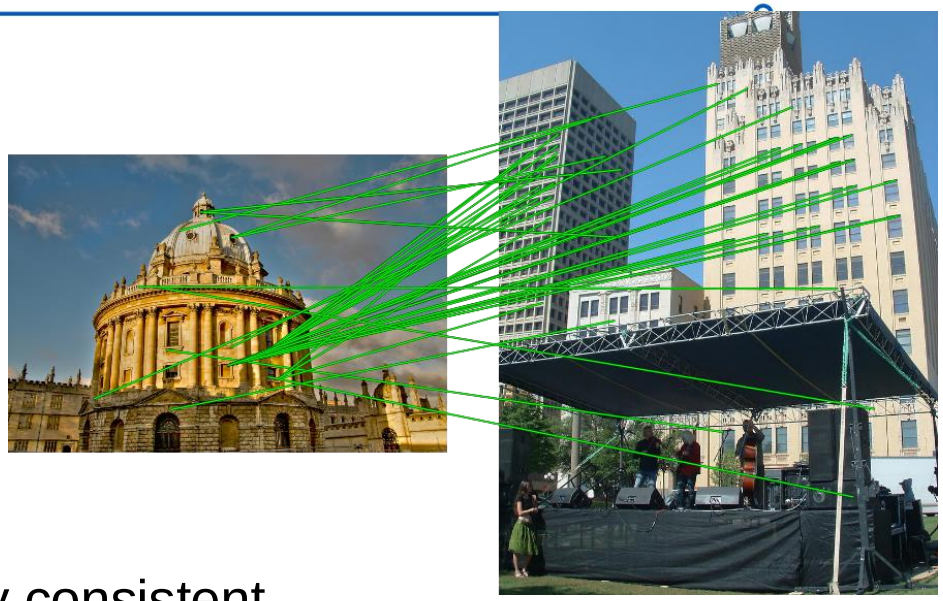
DB image with high BoW similarity

Both image pairs have many visual words in common.

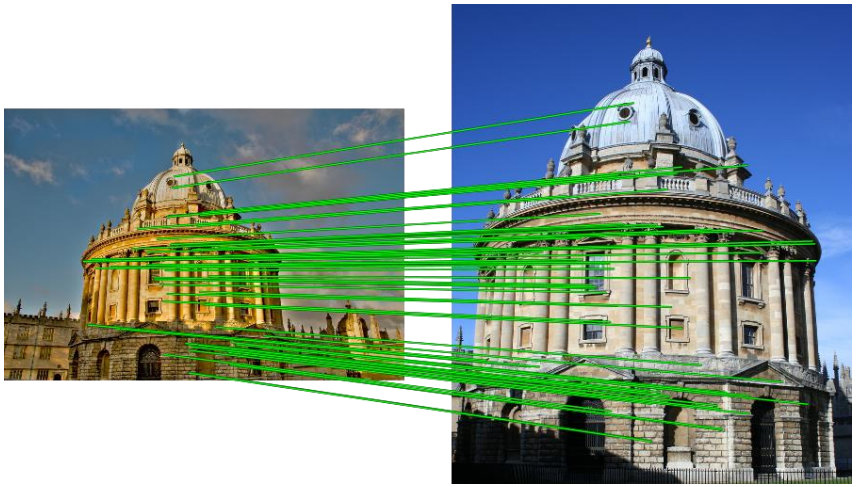
Slide credit: Ondrej Chum

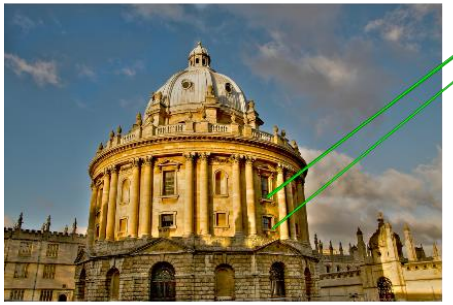
Spatial verification





Only some of the matches are mutually consistent





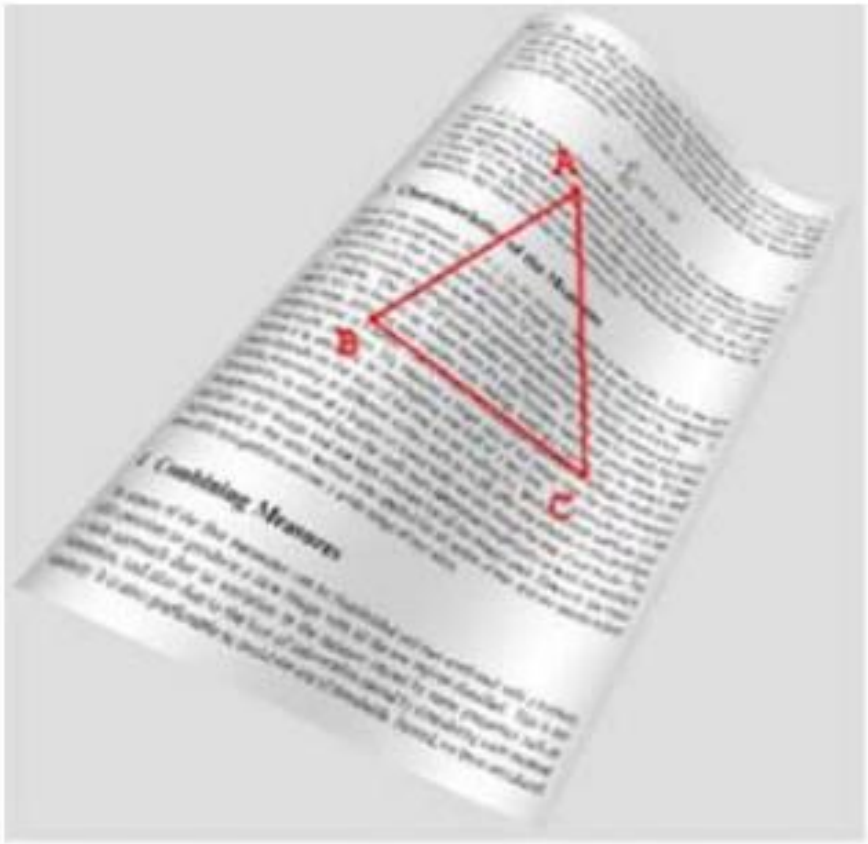
Spatial Verification: three basic strategies

- 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

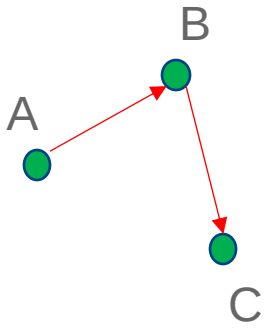
- Triplet Verification

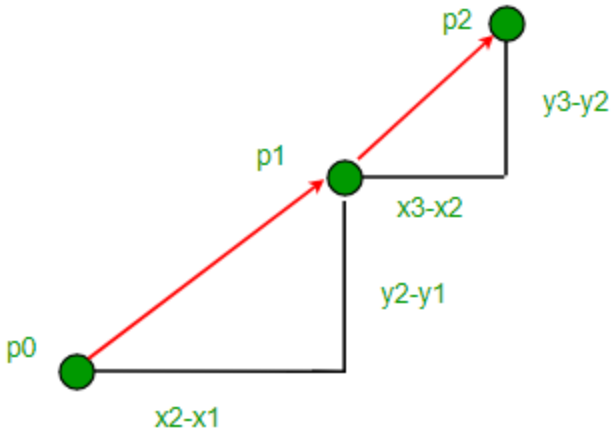
Triplet Verification



Using Slope to Determine Orientation

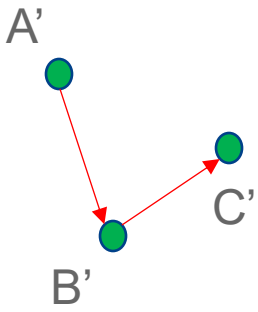
Consider the slopes….





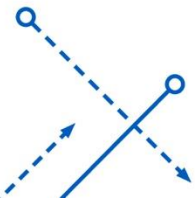
Slope(BC) – Slope(AB)?

< 0



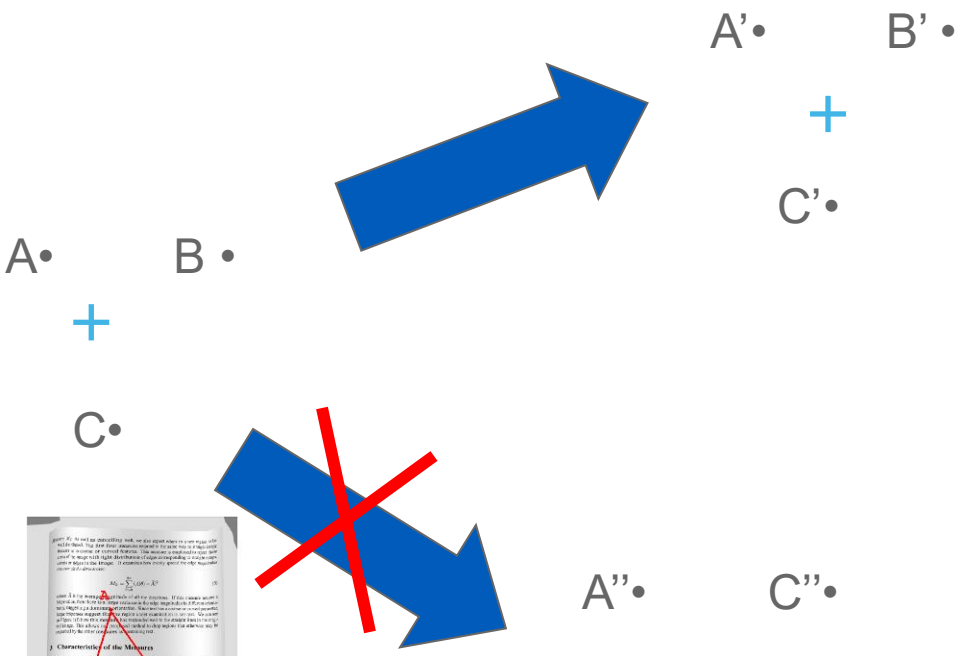
Slope(BC) – Slope(AB)?

> 0

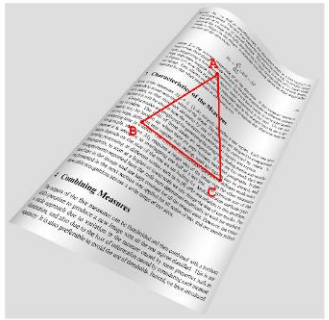


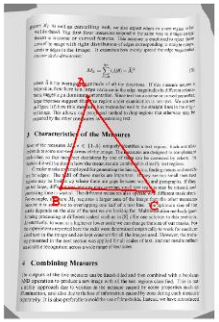
Triplet Verification

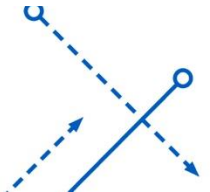
- Use the spatial relationship between “visual words”.
- Orientation of triplet is invariant under rotation, translation, scaling, warping and even crinkling



$$\begin{vmatrix} X_A & Y_A & 1 \\ X_B & Y_B & 1 \\ X_C & Y_C & 1 \end{vmatrix}$$







Triplet Verification: Scoring

- When viewed from another angle

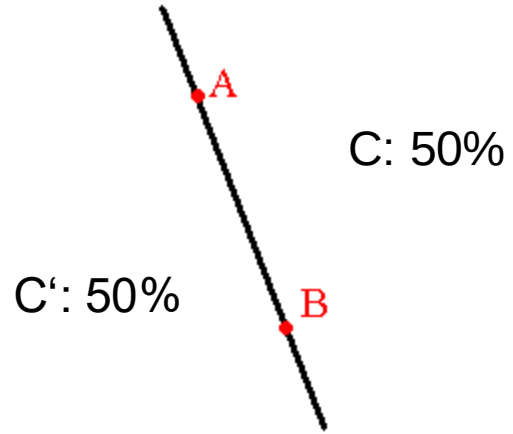
$$Sign(\begin{vmatrix} X_A & Y_A & 1 \\ X_B & Y_B & 1 \\ X_C & Y_C & 1 \end{vmatrix}) \times Sign(\begin{vmatrix} X'_A & Y'_A & 1 \\ X'_B & Y'_B & 1 \\ X'_C & Y'_C & 1 \end{vmatrix}) = 1$$

- Score is simply

$$\sum_{A,B,C \in S} (Sign(\begin{vmatrix} X_A & Y_A & 1 \\ X_B & Y_B & 1 \\ X_C & Y_C & 1 \end{vmatrix}) \times Sign(\begin{vmatrix} X'_A & Y'_A & 1 \\ X'_B & Y'_B & 1 \\ X'_C & Y'_C & 1 \end{vmatrix}))$$

Triplet Verification: Failure rate

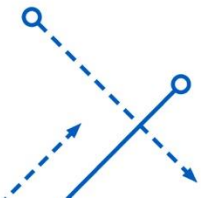
- Assume one triplet:
  - P(A,B,C clockwise) = 0.5
  - P(A,B,C anticlockwise) = 0.5



- Triplets are independent.
- Then possibility that M triplets out of N triplets accidentally satisfies orientation verification is

$$Q(N,M) = (\frac{1}{2})^N \binom{N}{M}$$

N	10	30	50	60
M	5	20	40	50
Q(N,M)	0.25	0.027	0.0001	6.5×10^{-8}



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?
  - Precision, Recall, and F1 score

Precision-Recall Curve (Recap)

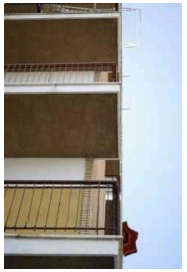
Database size: 10 images
Relevant (total): 5 images

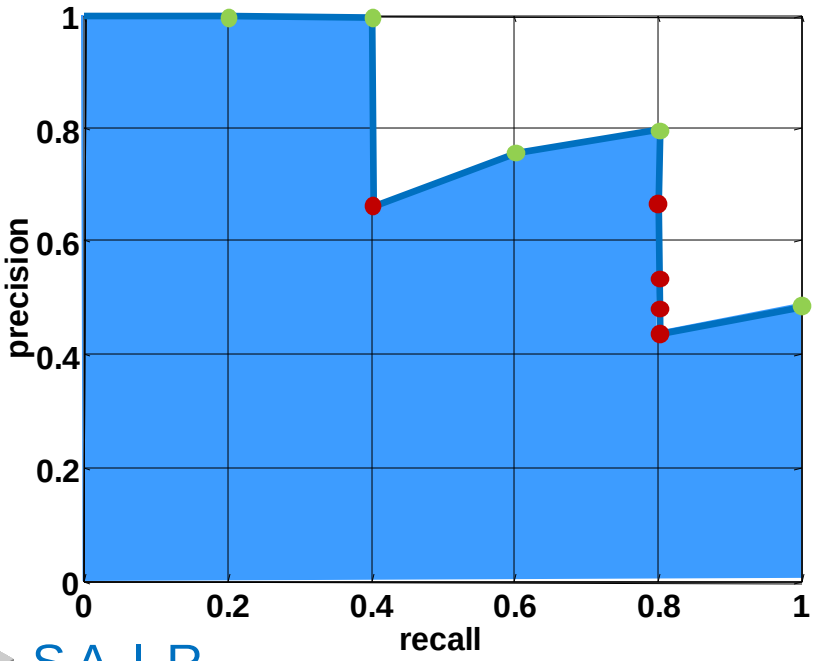
precision = #relevant / #returned
recall = #relevant / #total relevant

Query

Results (ordered):









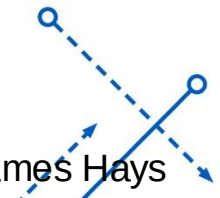

Slide credit: Ondrej Chum

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

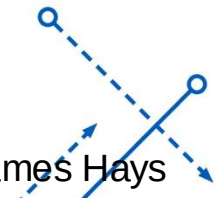
Lessons from a Decade Later

- For ***instance* retrieval** (this lecture)
  - Still widely used **in the field of simultaneously localization and mapping (SLAM)**.
  - Learn better local features (replace SIFT), e.g., MatchNet
  - or learn better image embeddings (replace the histograms of visual features), e.g., Vo and Hays 2016.
  - or learn to do spatial verification, e.g., DeTone, Malisiewicz, and Rabinovich 2016.
  - or learn a monolithic deep network to recognition all locations, e.g., Google's PlaNet 2016.



Lessons from a Decade Later

- For ***Category* recognition**
  - 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.
  - BoW is inspiring deep models, e.g., NetVLAD (deep learning model for place recognition).



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