IMAGE MATCHING

- ALOK TALEKAR
- SAIRAM SUNDARESAN

: Presentation structure :

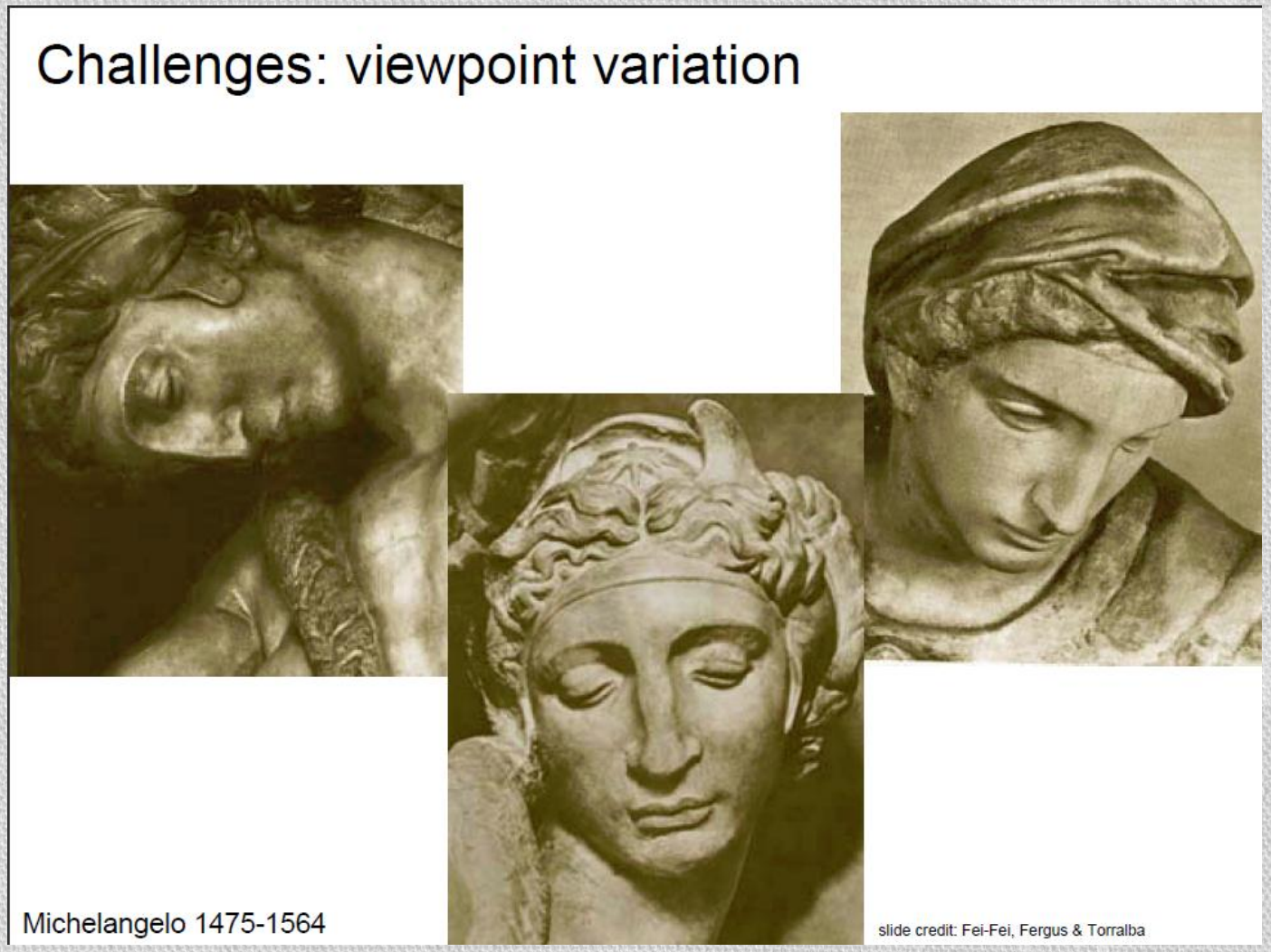
1. Brief overview of talk
2. What does Object Recognition involve?
3. The Recognition Problem
4. Mathematical background:
   1. Trees
   2. K-Means
   3. Modifications to K-Means
5. Results and Comparisions
6. Application of algorithms to mobile phones
7. Conclusion.

How many object categories are there?

Motivation for Image Recognition

- Image panoramas
- Image watermarking
- Global robot localization
- Face Detection
- Optical Character Recognition
- Manufacturing Quality Control
- Content-Based Image Indexing
- Object Counting and Monitoring
- Automated vehicle parking systems
- Visual Positioning and tracking
- Video Stabilization

What Challenges exist?



What Challenges exist?

Challenges: illumination

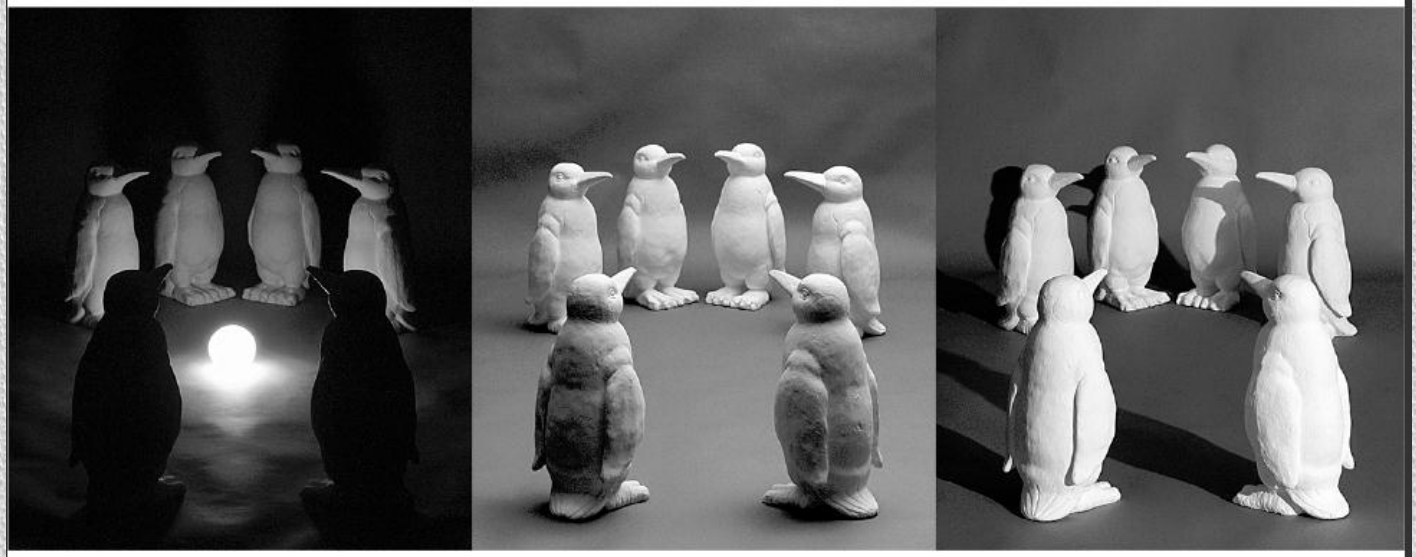
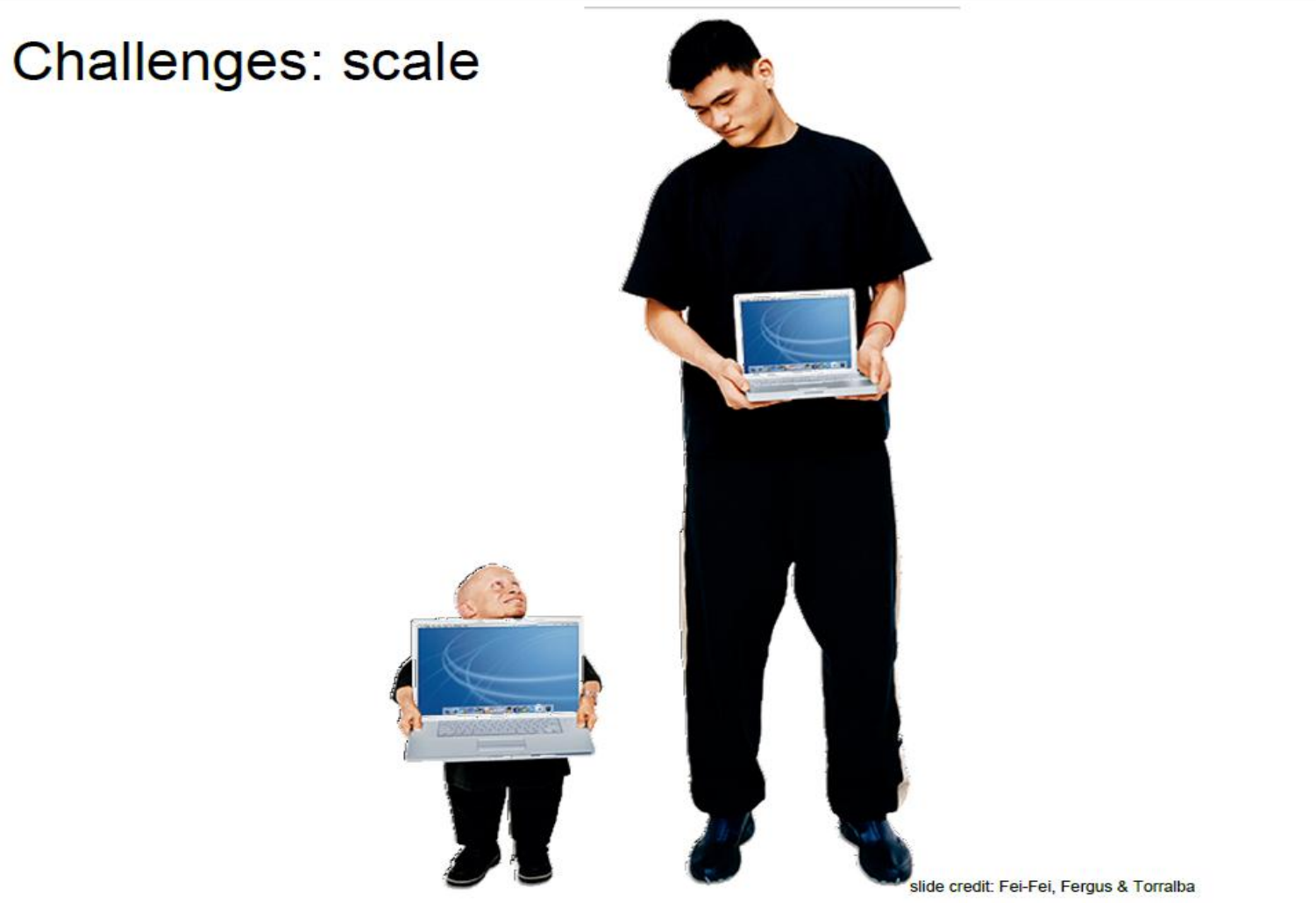


image credit: J. Koenderink

What Challenges exist?



What Challenges exist?

Challenges: deformation

What other challenges exist?

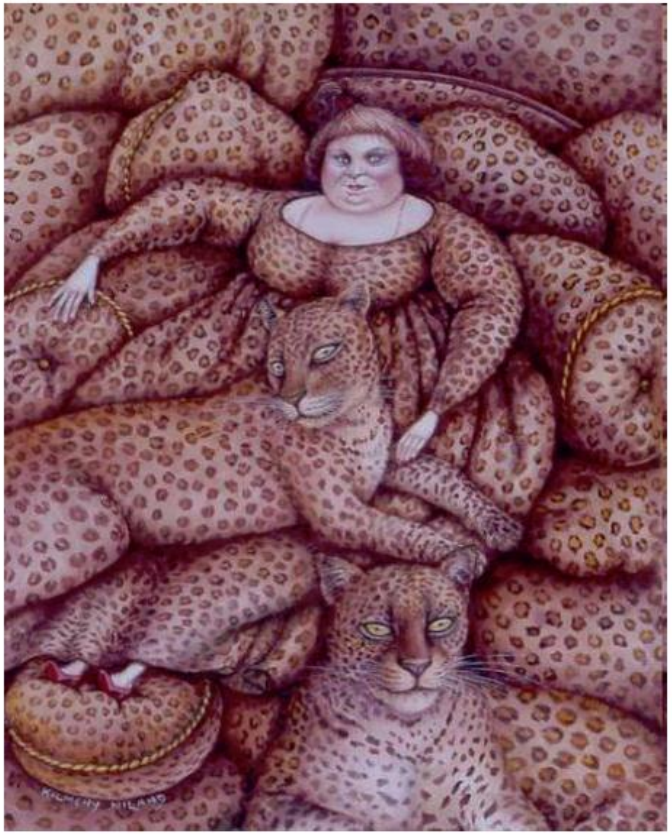
Challenges: occlusion

Magritte, 1957

slide credit: Fei-Fei, Fergus & Torralba

What other challenges exist?

Challenges: background clutter



Kilmeny Niland. 1995

What other challenges exist?

Challenges: intra-class variation






So what does object recognition involve?

What does object recognition involve?

- Identification
- Detection
- Object categorization
- Scene and context categorization
- Tracking
- Action Recognition
- Events

Object categorization: the statistical viewpoint

$$p(zebra \mid image)$$

vs.

$$p(no\ zebra \mid image)$$

- Bayes rule:

$$\underbrace{\frac{p(zebra \mid image)}{p(no\ zebra \mid image)}}_{\text{posterior ratio}} = \underbrace{\frac{p(image \mid zebra)}{p(image \mid no\ zebra)}}_{\text{likelihood ratio}} \cdot \underbrace{\frac{p(zebra)}{p(no\ zebra)}}_{\text{prior ratio}}$$

Object categorization: the statistical viewpoint

- **Discriminative methods model posterior**
- **Generative methods model likelihood and prior**

- Bayes rule:

$$\underbrace{\frac{p(zebra \mid image)}{p(no\ zebra \mid image)}}_{\text{posterior ratio}} = \underbrace{\frac{p(image \mid zebra)}{p(image \mid no\ zebra)}}_{\text{likelihood ratio}} \cdot \underbrace{\frac{p(zebra)}{p(no\ zebra)}}_{\text{prior ratio}}$$

Three main issues

- Representation
  - How to represent an object category
- Learning
  - How to form the classifier, given training data
- Recognition
  - How the classifier is to be used on novel data

Problem

- Variations in :
  - Scale
  - Illuminations
  - Rotation
  - Noise : different sources for images.

- Large number of objects and object categories.
- Large variation in “shapes”

- → **Use better features: SURF, GoH**

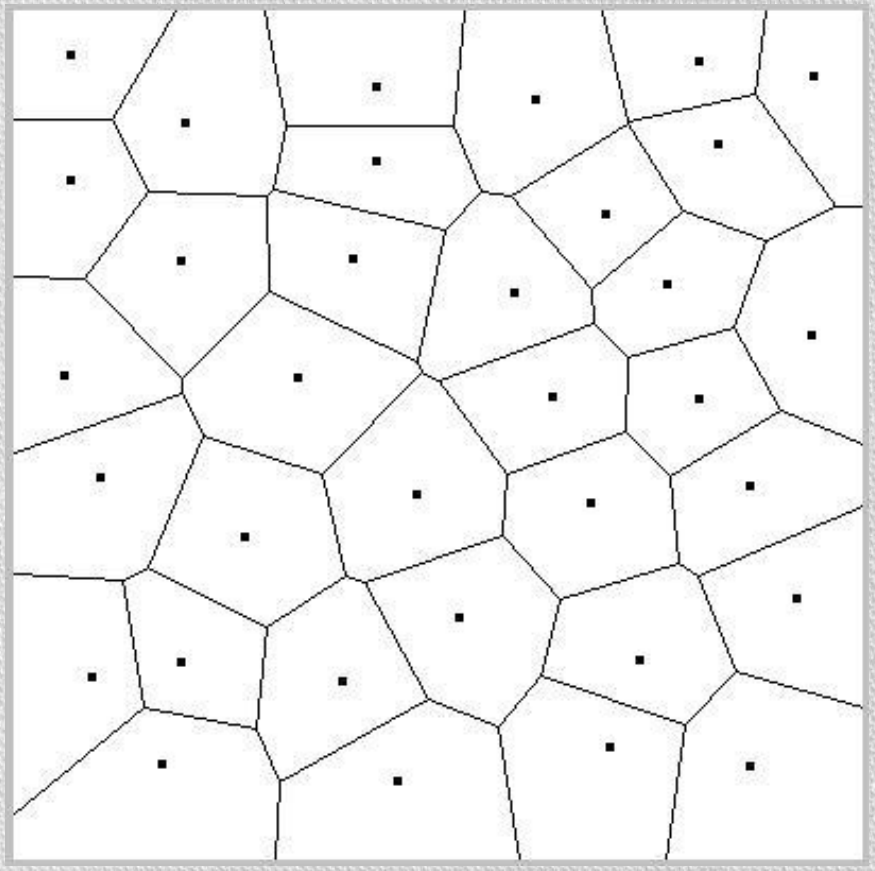
→ TODAY’S TALK : scaling + compactness

Image Recognition

Mathematics

- Voronoi diagram and vector quantization
- Trees : kd trees, vocabulary trees
- K-Means clustering
- Modifications of the k-Means.

Voronoi diagram



After Georgy Voroni

Let P be a set of n distinct points (sites) in the plane.

The Voronoi diagram of P is the subdivision of the plane into n cells, one for each site.

A point q lies in the cell corresponding to a site $p_i \in P$ *iff* Euclidean_Distance(q, p_i) < Euclidean_distance(q, p_j), for each $p_i \in P$, $j \neq i$.

Voronoi diagram

- → choose measure of distance.
- → partition space.
- → dual graph for delauney triangulation.
- → centers partitions with semi infinite edges, are the convex hull vertices.
- → implementation is O(n) using doubly linked lists. Construction requires O(n log(n)).
- kd trees are used for a similar purpose.

Clustering

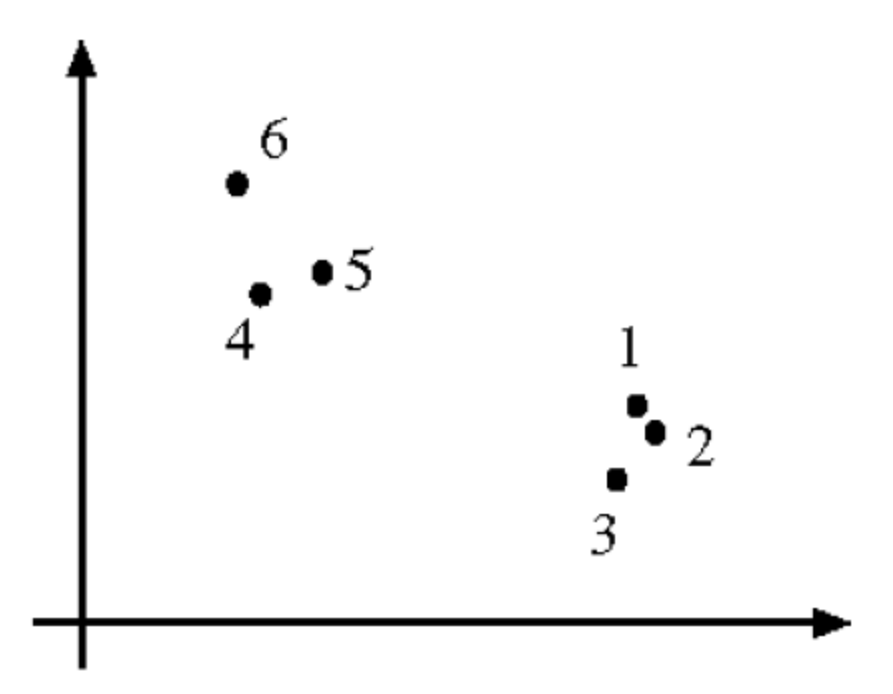
- Iterative.

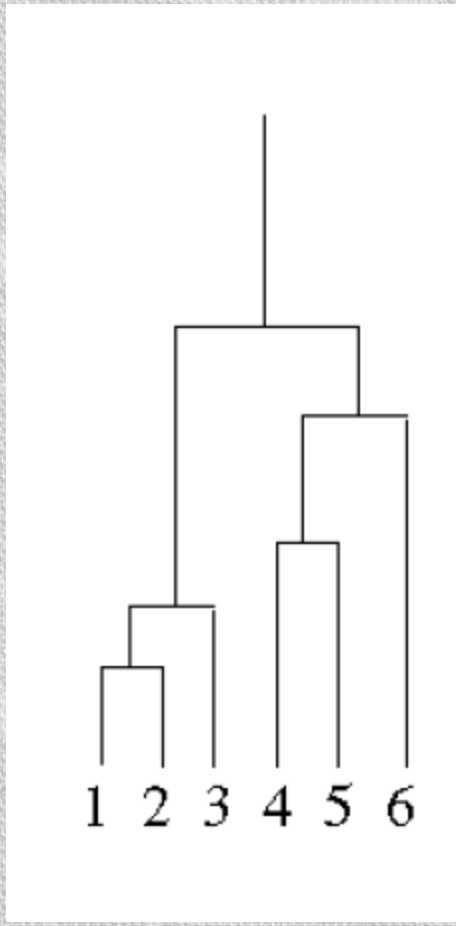
- Agglomerative clustering–Add token to cluster if token is similar enough to element of clusters
  - –Repeat

- Divisive clustering–Split cluster into subclustersif tokens are dissimilar enough within cluster
- –Boundary separates subclustersbased on similarity
  - –Repeat

Slide credit: Silvio Savarese

Hierachical structure of clusters





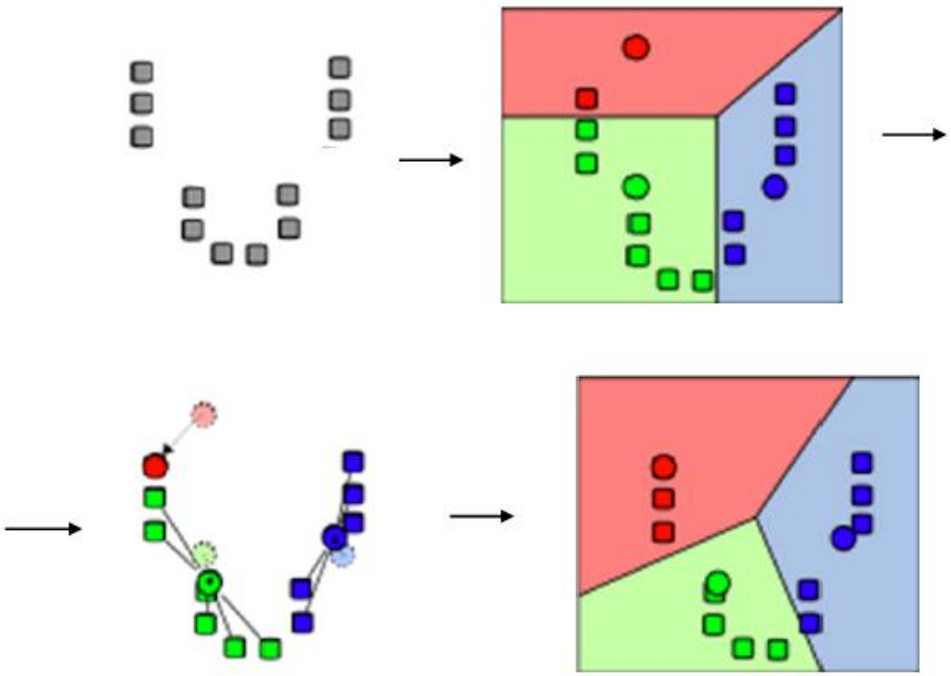
Slide credit: Silvio Savarese

K-Means Clustering

- Initialization: Given K categories, N points in feature space. Pick K points randomly; these are initial cluster centers (means) m1, …,mK. Repeat the following:
-
- 1. Assign each of the Npoints, ***xj , to clusters by nearest mi***
- 2. Re-compute mean miof each cluster from its member points
- 3. If no mean has changed more than some ε, stop

Slide credit: Christopher Rasmussen

Example: 3-means Clustering



Source: wikipedia

- Effectively carries out gradient descent to minimize:

$$e(\mathbf{m}_i) = \sum_{i=1}^{n_c} \sum_{j;c_j=i} |\mathbf{x}_j - \mathbf{m}_i|^2$$

$$\frac{\partial e}{\partial \mathbf{m}_k} = \sum_{j;c_j=k} -2(\mathbf{x}_j - \mathbf{m}_k) = 0$$

$$\mathbf{m}_k = \frac{\sum_{j;c_j=k} \mathbf{x}_j}{\sum_{j;c_j=k} 1} = \frac{1}{n_k} \sum_{j;c_j=k} \mathbf{x}_j$$

Slide credit: Silvio Savarese

Complexity

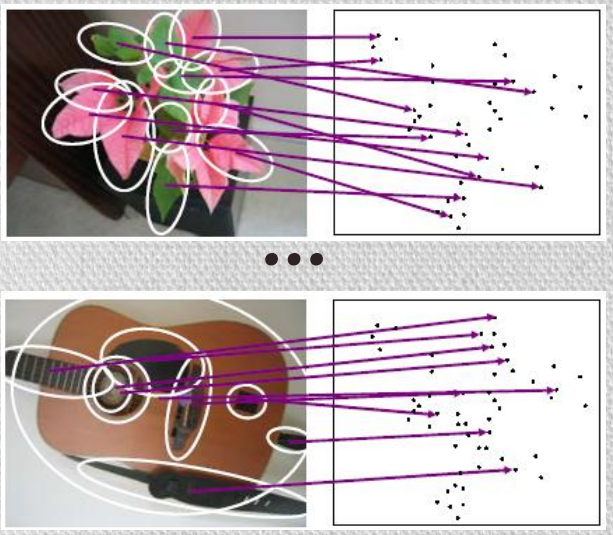
- NP-Hard. Using huristics, quick to converge.
- No guarantee that it will converge to the global optimum, and the result may depend on the initial clusters.
- As the algorithm is usually very fast, it is common to run it multiple times with different starting conditions.
- Similarity to EM, EM maintains probabilistic assignments to clusters, instead of deterministic assignments, and multivariate Gaussian distributions instead of means.

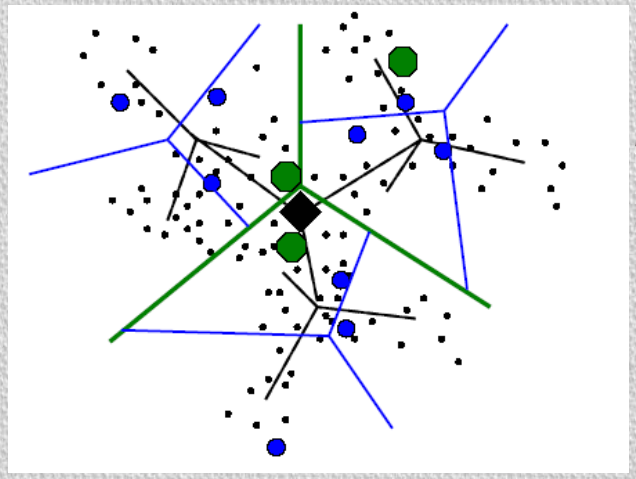
Why Trees?

- Quick to understand and implement.

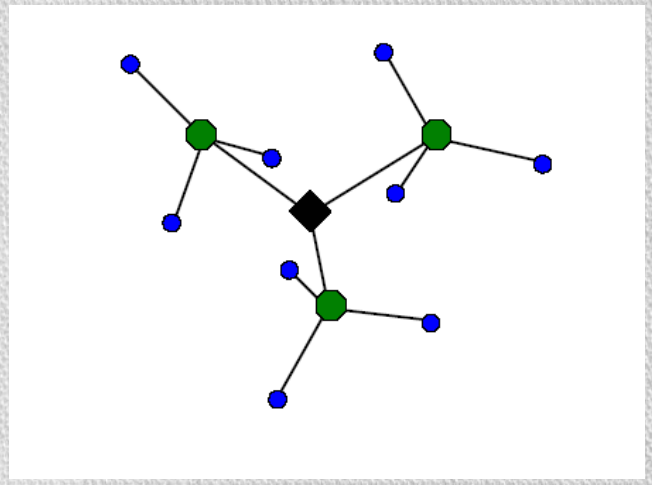
- Computational benefits:
  - Most algorithms (when implemented correctly) give a O(n log(n)) performance.
  - Speedup due to dividing the search space at each stage.

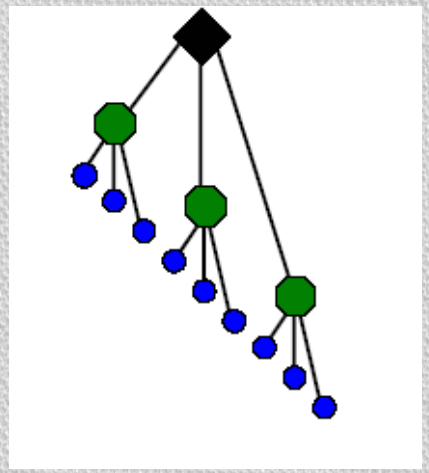
Building the Vocabulary Tree





k=3 *L*=2

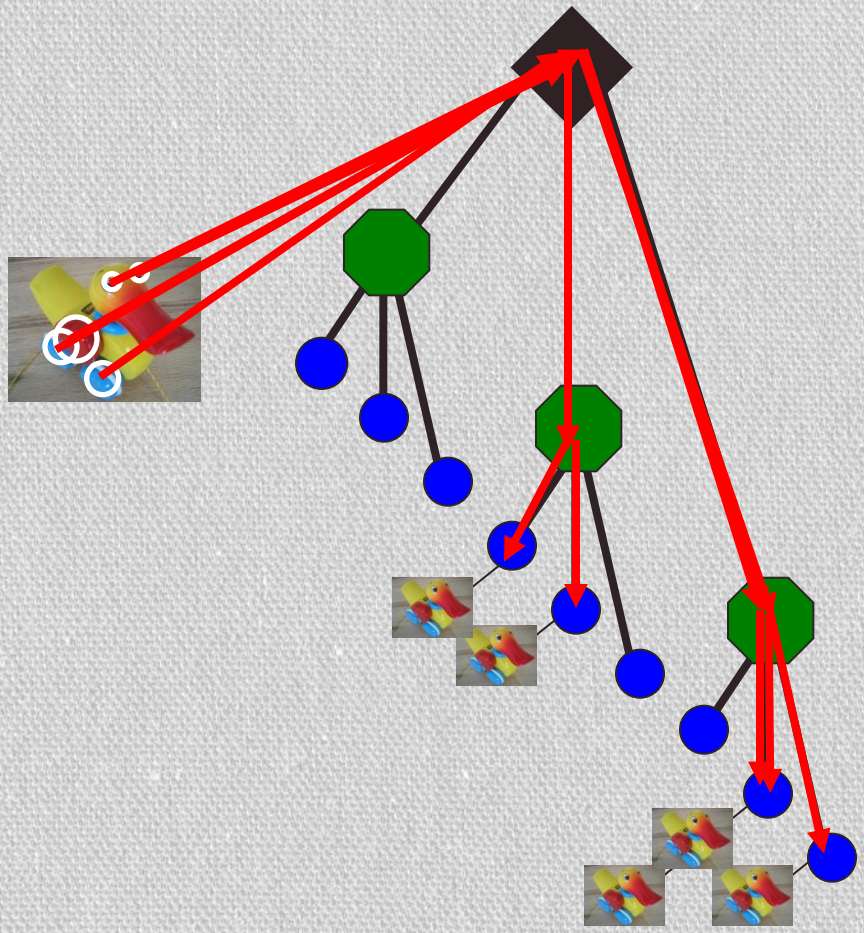




Demo

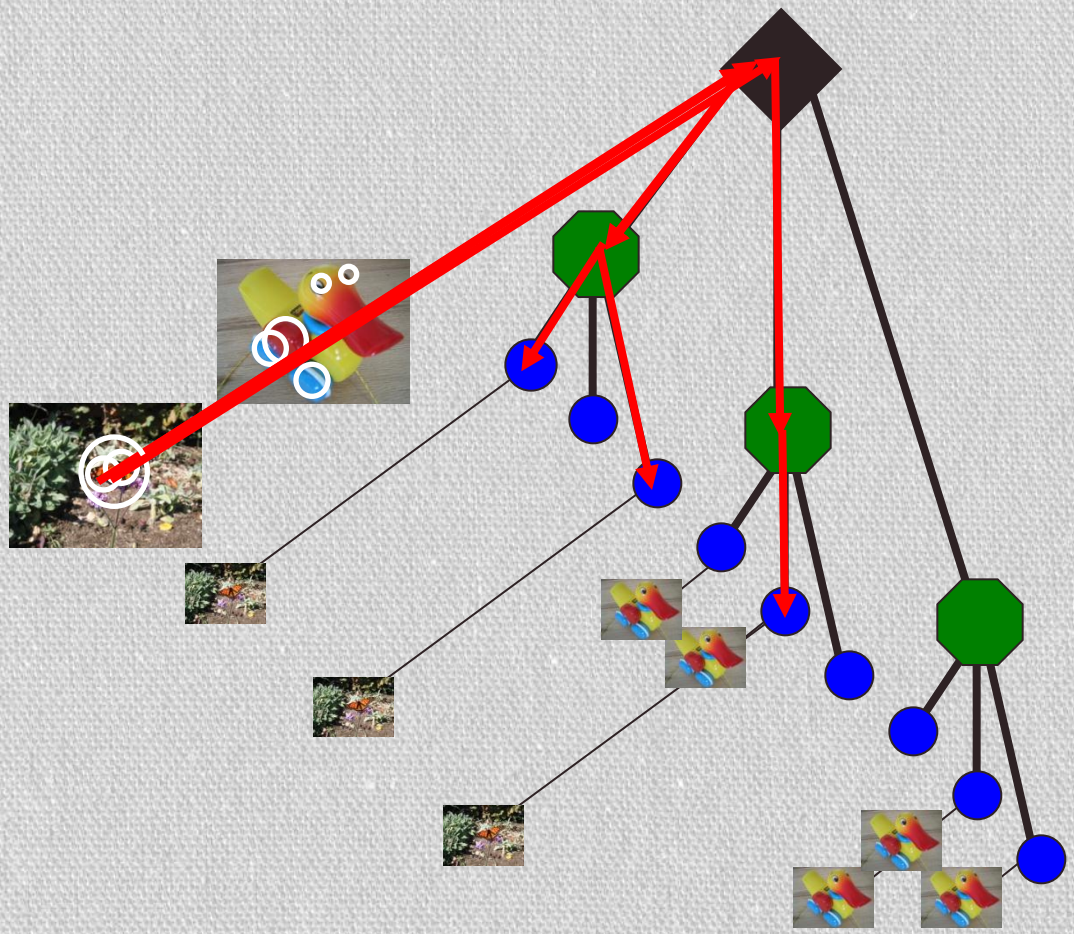
- http://www.visualthesaurus.com/

Describing an Image



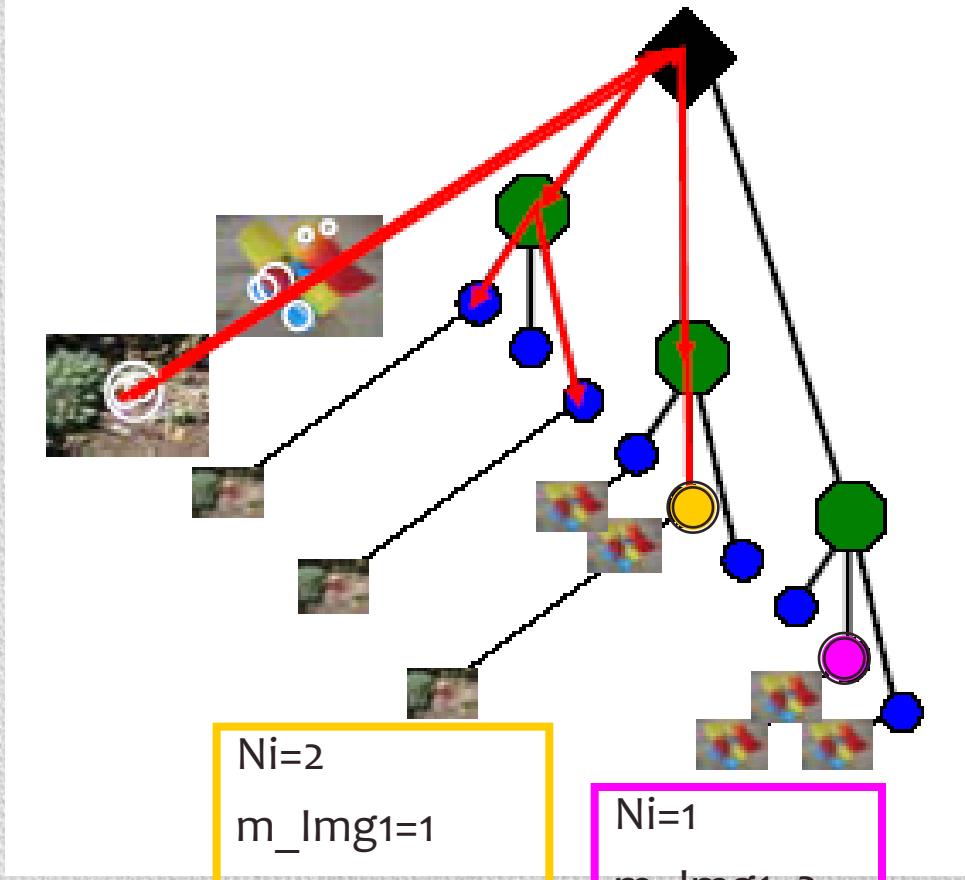
Slide credit Tatiana Tommasi - Scalable Recognition with a Vocabulary Tree

Describing an Image



Slide credit Tatiana Tommasi - Scalable Recognition with a Vocabulary Tree

Definition of Scoring



- Number of the descriptor vectors of each image with a path along the node *i* (n_i query, m_i database)
- Number of images in the database with at least one descriptor vector path through the node *i* (N_i)

Definition of Scoring

- Weights are assigned to each node (tf –idf)

$$w_i = \ln \frac{N}{N_i}$$

- Query and database vectors are defined according to their assigned weights

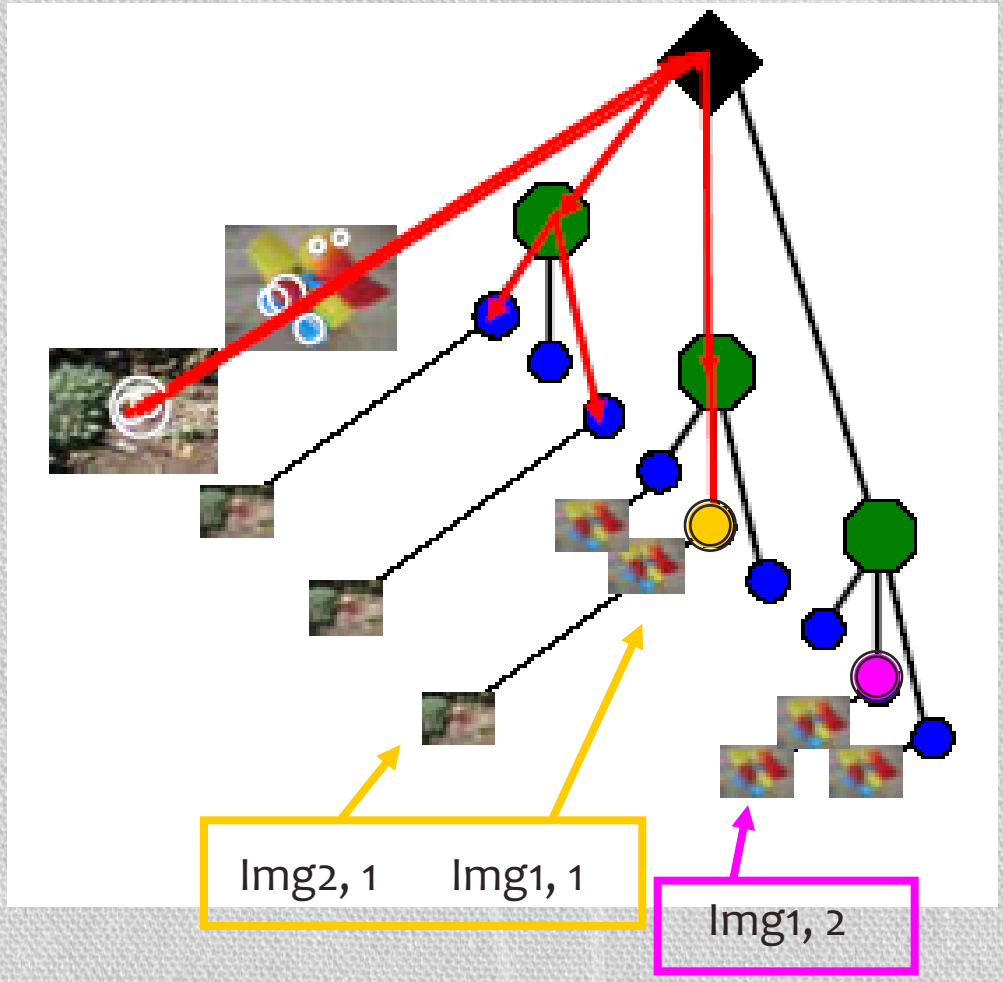
$$\begin{aligned} q_i &= n_i w_i \\ d_i &= m_i w_i \end{aligned}$$

- Each database image is given a relevance score based on the normalized difference between the query and the database vectors

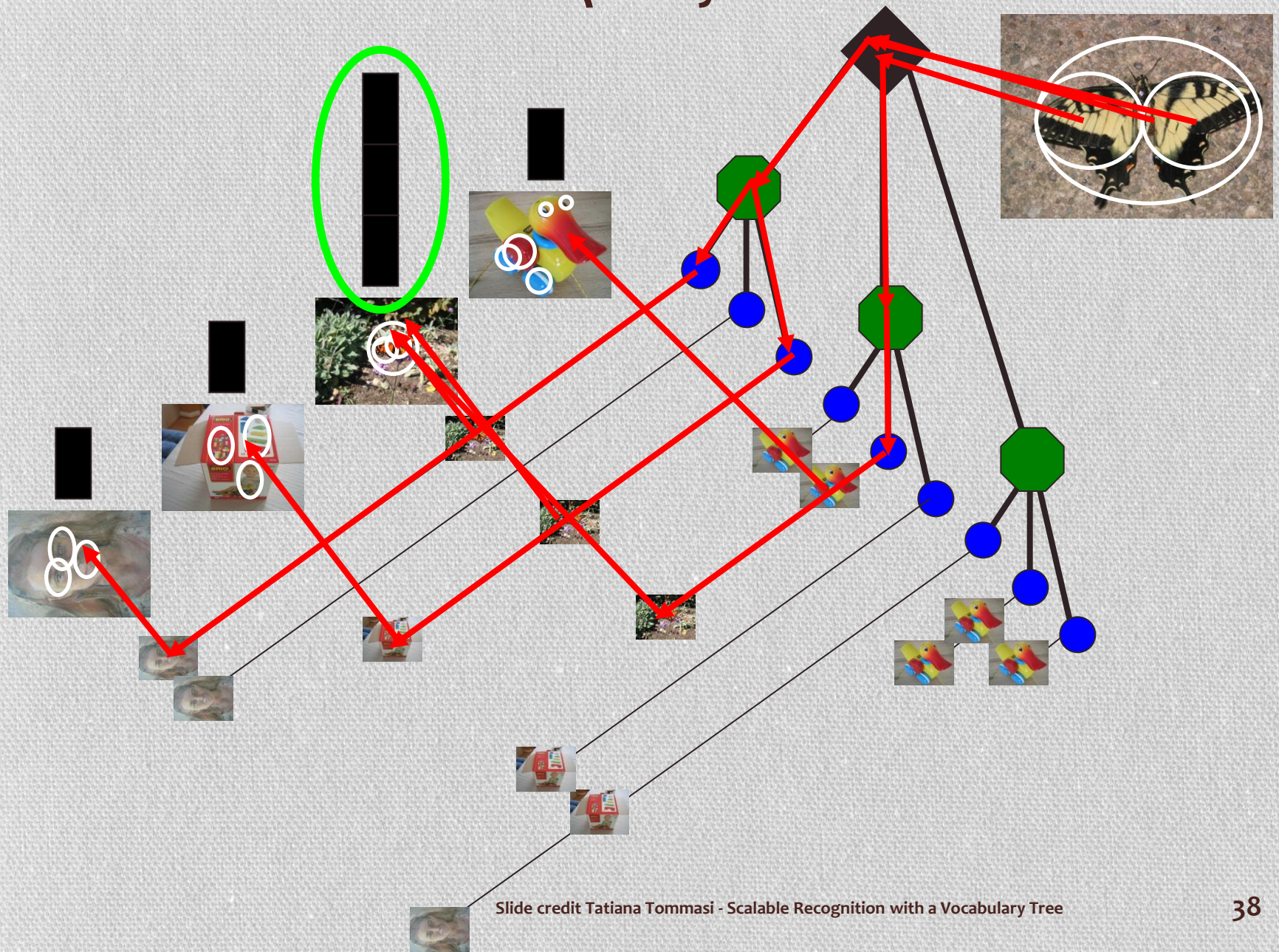
$$s(q,d) = \| \frac{q}{\| q \|} - \frac{d}{\| d \|} \|$$

Implementation of Scoring

- Every node is associated with an **inverted file**.
- Inverted files stored the **id-numbers of the images** in which a particular node occurs and the **term frequency** of that image.
- decrease the fraction of images in the database that have to be explicitely considered for a query.



Query



Slide credit Tatiana Tommasi - Scalable Recognition with a Vocabulary Tree

Database

- Ground truth database 6376 images
- Groups of four images of the same object but under different conditions
- Each image in turn is used as query image and the three remaining images from its group should be at the top of the query results

Slide credit Tatiana Tommasi - Scalable Recognition with a Vocabulary Tree

Modifications to k-Means

- Earlier approaches require additional $O(K^2)$ space for a speedup, which in general is impractical if clustering in the range of 1M.
- Two additional methods were proposed:
- → AKM: Approximate k-Means
- →HKM: Hierarchical k-Means

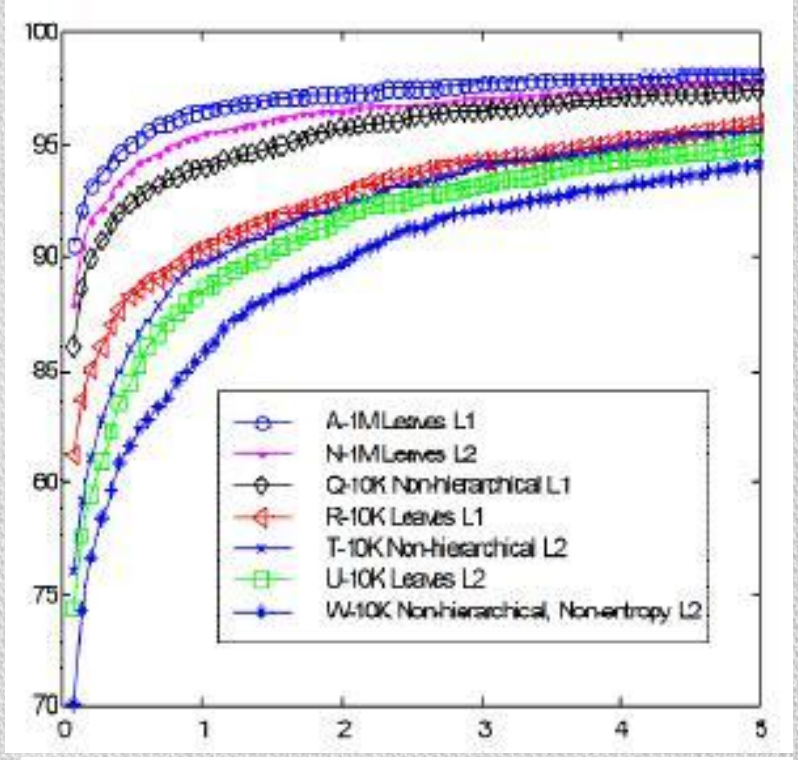
AKM

- Uses kd-trees.
- Gives almost equivalent performance in smaller scales and out performances at higher scale.
- Saves time by using elements of randomness in choosing the nearest neighbor between points and mean, rather than the exact nearest neighbor.

Hierarchical K-Means (HKM)

- Also called tree structured vector quantization.
- Here, all data points are clustered to a small number of cluster centers (say K).
- On the next level, k-means is applied again, within each of partitions independently with the same number of cluster centers.
- Thus at the nth level, there are K^n clusters.
- A new data point is assigned by descending the tree.

Results on 1400 images (Nister and Stewenius)

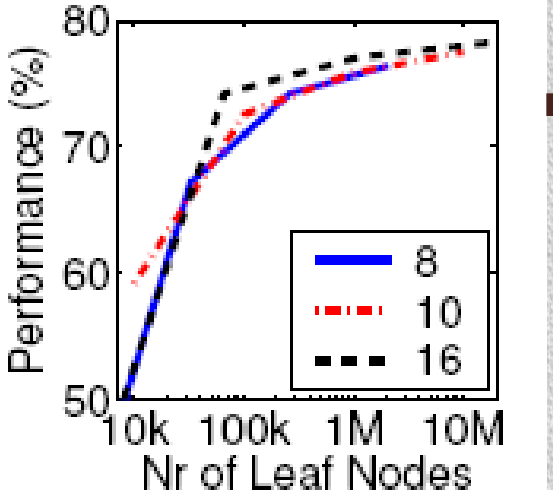


Me	En	No	S%	Voc-Tree	Le	Eb	Perf
A	y/y	L1	0	6x10=1M	1	ir	90.6
N	y/y	L2	0	6x10=1M	1	ir	87.9
Q	y/y	L1	0	1x10K=10K	1	-	86.0
R	y/y	L1	0	4x10=10K	2	ir	81.3
T	y/y	L2	0	1x10K=10K	1	-	76.0
U	y/y	L2	0	4x10=10K	1	ir	74.4
W	n/n	L2	0	1x10K=10K	1	-	70.1

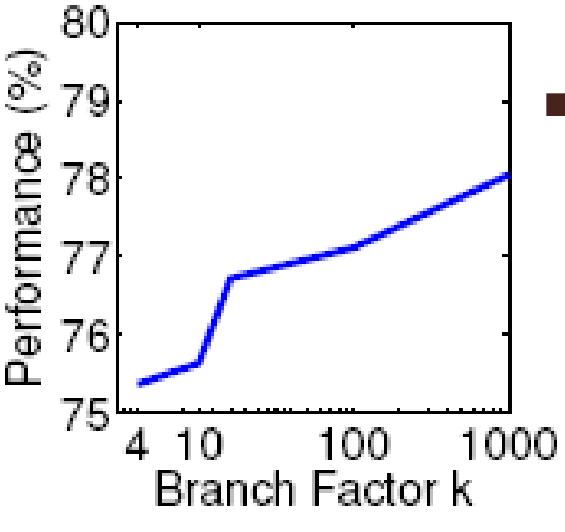
The curves show the distribution of how far the wanted images drop in the query rankings

- A larger (hierarchical) vocabulary improves retrieval performance
- L_1 norm gives better retrieval performance than L_2 norm.
- Entropy weighting is important at least for smaller vocabularies

Results on 6376 images

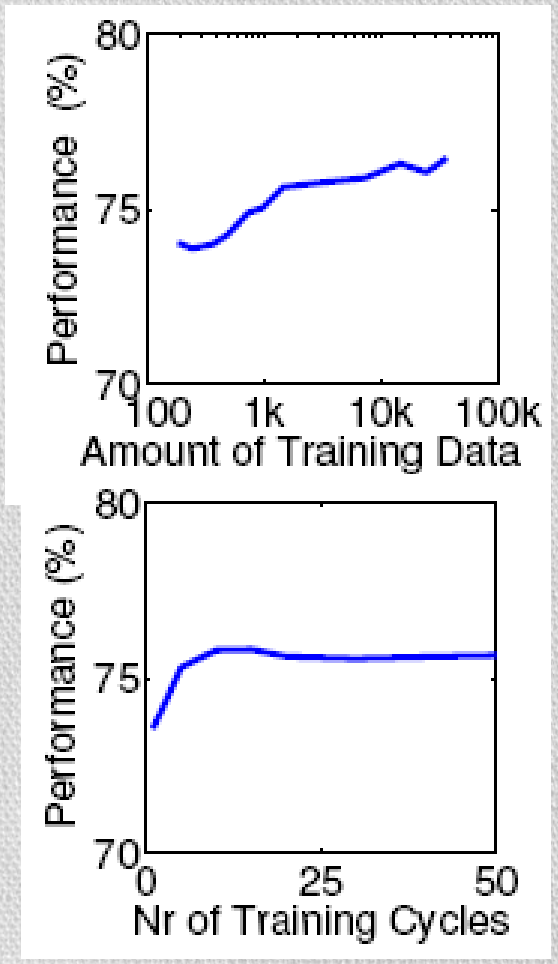


Performance increases significantly with the number of leaf nodes



Performance increases with the branch factor *k*

Results on 6376 images



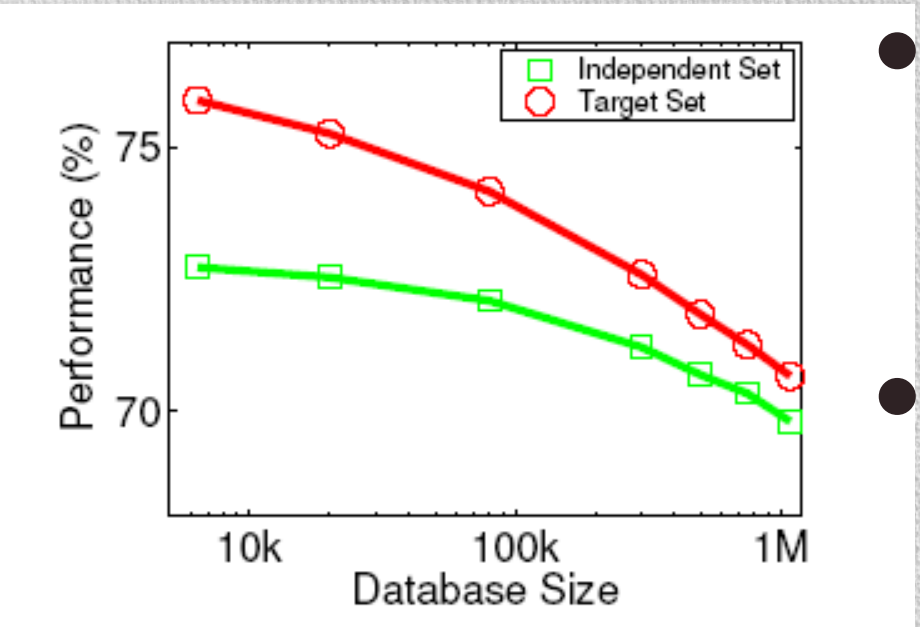
→ Performance increases when the amount of training data grows

→ Performance increases at the beginning when the number of training cycles grows, then reaches a plateaux

Results on 1 million images

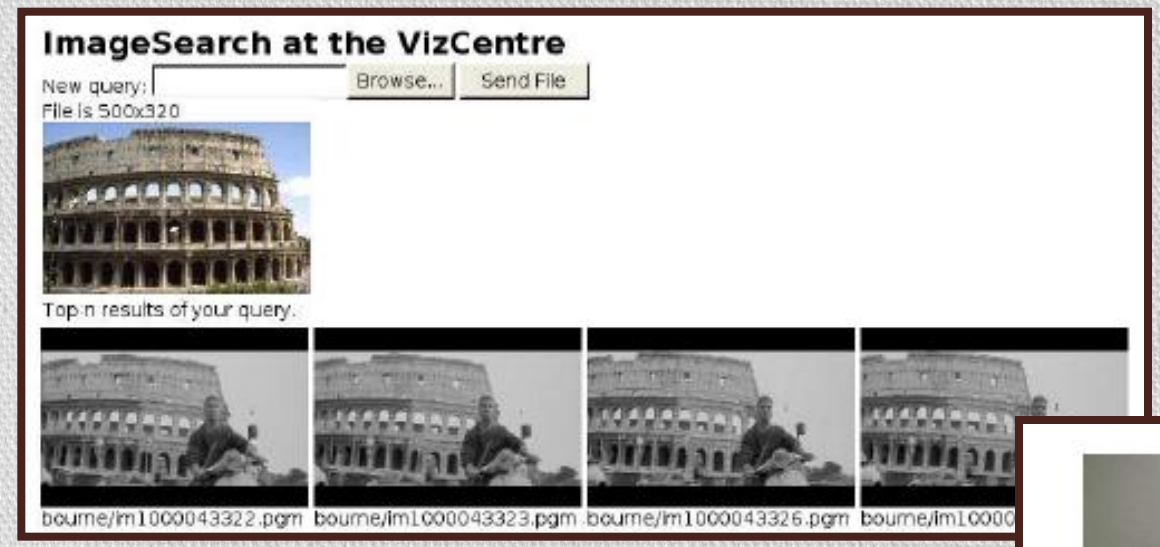
Performance with respect to increasing database size.

The vocabulary tree is defined with video frames separate from the database.



- Entropy weighting of the vocabulary tree defined with video independent from the database.
- Entropy weighting defined using the ground truth target subset of images.

Applications



Conclusions from Nister & Stewenius's work

The methodology provides the ability to make **fast searches** on **extremely large databases**.

If repeatable, discriminative features can be obtained, then recognition can scale to very large databases using the vocabulary tree and indexing approach.

Results on comparing vocabularies (Sivic)

- K-means vs AKM:
  - For the small 5K dataset, there is almost identical performance between AKM and exact K-means.
  - AKM differs in mAP by less than 1% and even outperforms k-means in two of the cases.

Clustering parameters		mAP	
# of descr.	Voc. size	k-means	AKM
800K	10K	0.355	0.358
1M	20K	0.384	0.385
5M	50K	0.464	0.453
16.7M	1M		0.618

Table 2. Comparison of the performance of exact k-means to our AKM method on the 5K dataset, using different numbers of training descriptors and clusters.

Results on Comparing Vocabularies

- HKM vs AKM:
  - Comparison is done in two ways, first, by comparing performance on the recognition benchmark introduced by Nister and Stewenius and second, by comparing the performance of the two methods on the 5K dataset.
  - In the first evaluation, 10,200 images of the same scene taken from different viewpoints are split into 4 groups. A perfect result is to return, given a query, the other three images from that query's group before images from other groups.
  - For the same number of visual words, AKM outperforms HKM.
  - The same holds true in the second evaluation, where the 5K dataset is used.

Tabulated Results

Method	Scoring Levels	Average Top
HKM	1	3.16
HKM	2	3.07
HKM	3	3.29
HKM	4	3.29
AKM		**3.45**

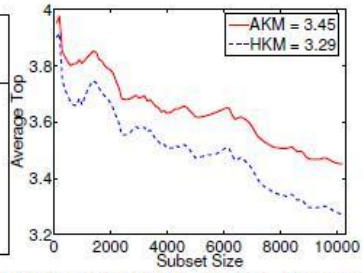


Table 3. A comparison of the AKM and HKM on the Recognition Benchmark of [20] using the descriptors for training and testing provided by the authors of [20]. "HKM" is the hierarchical k-means quantization, where the numbers are taken from [2]. "AKM" is the result of our approximate k-means clustering. Both methods use a vocabulary of 1M visual words and an L_1 distance.

Method	Dataset	mAP	
		Bag-of-words	Spatial
(a) HKM-1	5K	0.439	0.469
(b) HKM-2	5K	0.418	
(c) HKM-3	5K	0.372	
(d) HKM-4	5K	0.353	
(e) AKM	5K	0.618	0.647
(f) AKM	5K+100K	0.490	0.541
(g) AKM	5K+100K+1M	0.393	0.465

Table 4. Vocabulary comparison over the three datasets. For the HKM method, the number of levels used for scoring is listed in the method name. All methods use 1M cluster centers, generated from all 16.7M descriptors in the 5K dataset. The "spatial" method is described in section 4.

Scaling up with AKM

- The performance of AKM is evaluated for a number of different vocabulary sizes for 5K, 5K+100K, and 5K+100K+1M datasets.
- In going from the smallest dataset to the largest, there is a 226 fold increase in the number of images, but the performance falls by just over 20%.

Outdoors Augmented Reality on Mobile Phone using Loxel-Based Visual Feature Organization

- System Overview
  - Image matching on the cell phone
- Data Organization
  - Server groups images by location
- Data Reduction
  - Server clusters, prunes and compresses descriptors
- Image Matching Results
  - Qualitative and quantitative
- System Timing
  - Low latency image matching on the cell phone

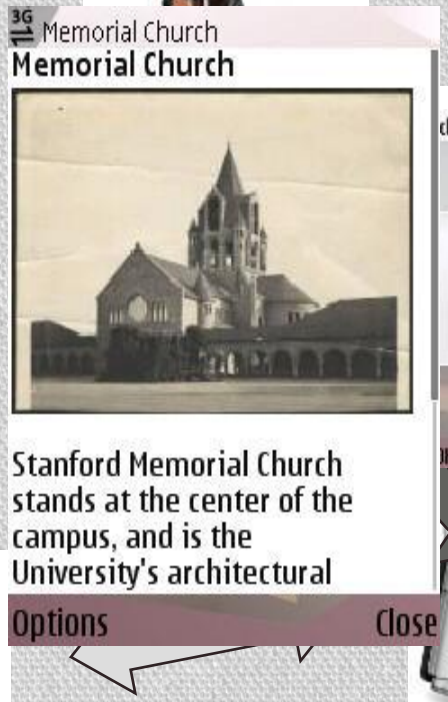
Video of Results

System Overview

- Since recognizing all the different locations just visually is a very complicated vision task, the authors **simplify** and **use rough location as a cue.**
- Phone gets **position from GPS** satellite (or other means, it's good enough to know the **rough** location within a city block or two)
- Phone **prefetches data** relevant to current position
- Then the user **snaps a photo**
- the image is then **matched** with the prefetched data
- And the **result is displayed** to the user

System Overview

GPS

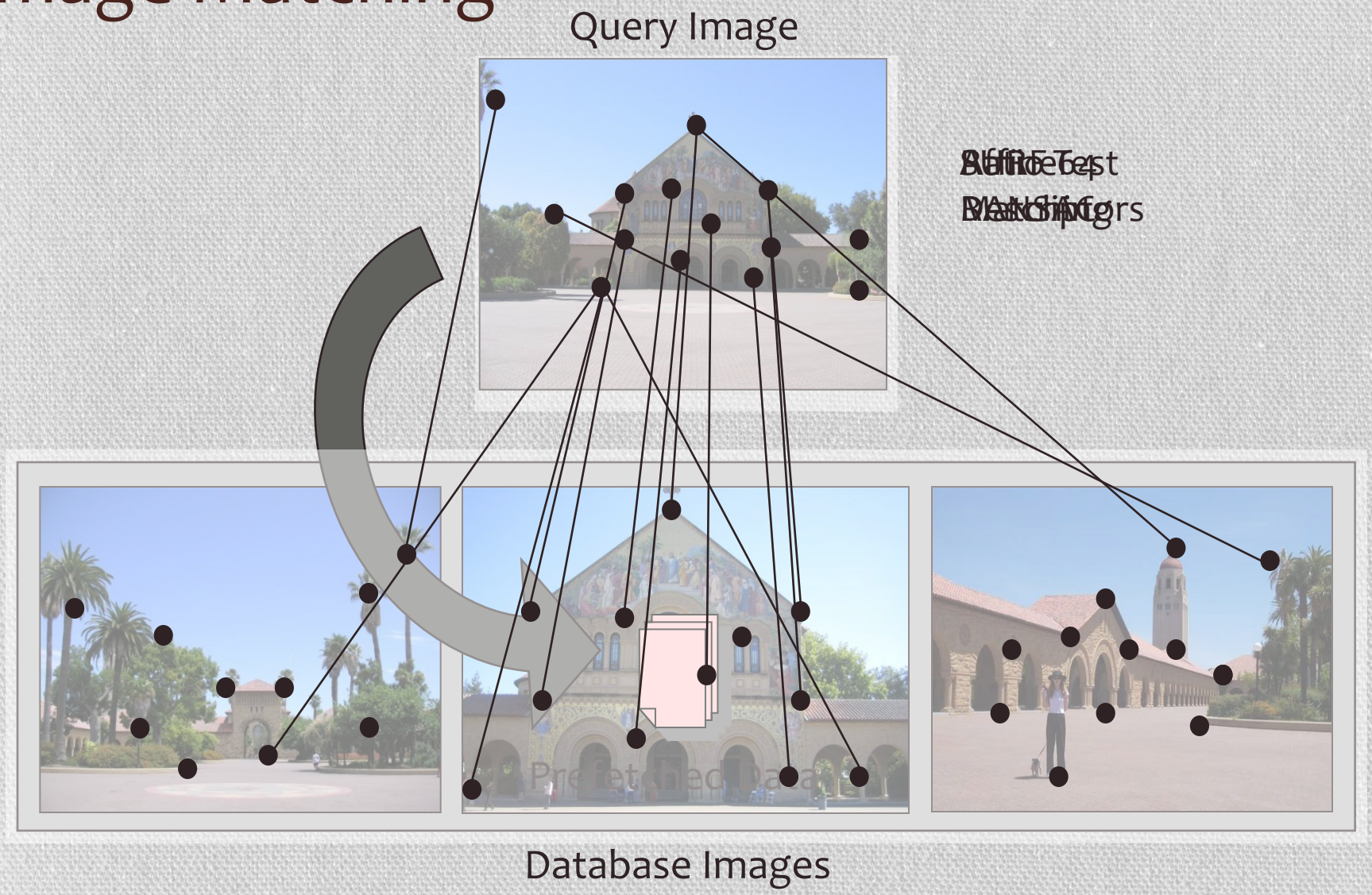


Server

Image Matching

- How does the matching work ? And what's in the pre-fetched data?
- For example, in the next slide that I'll show you, there are some database images taken from the user's location
- The prefetched data contains features from these images. features are extracted from the query image and matched against features in our database of features.
- The next step is to run geometric consistency (RANSAC) checks to eliminate false points.
- Finally, the image with the highest number of point matches is chosen.

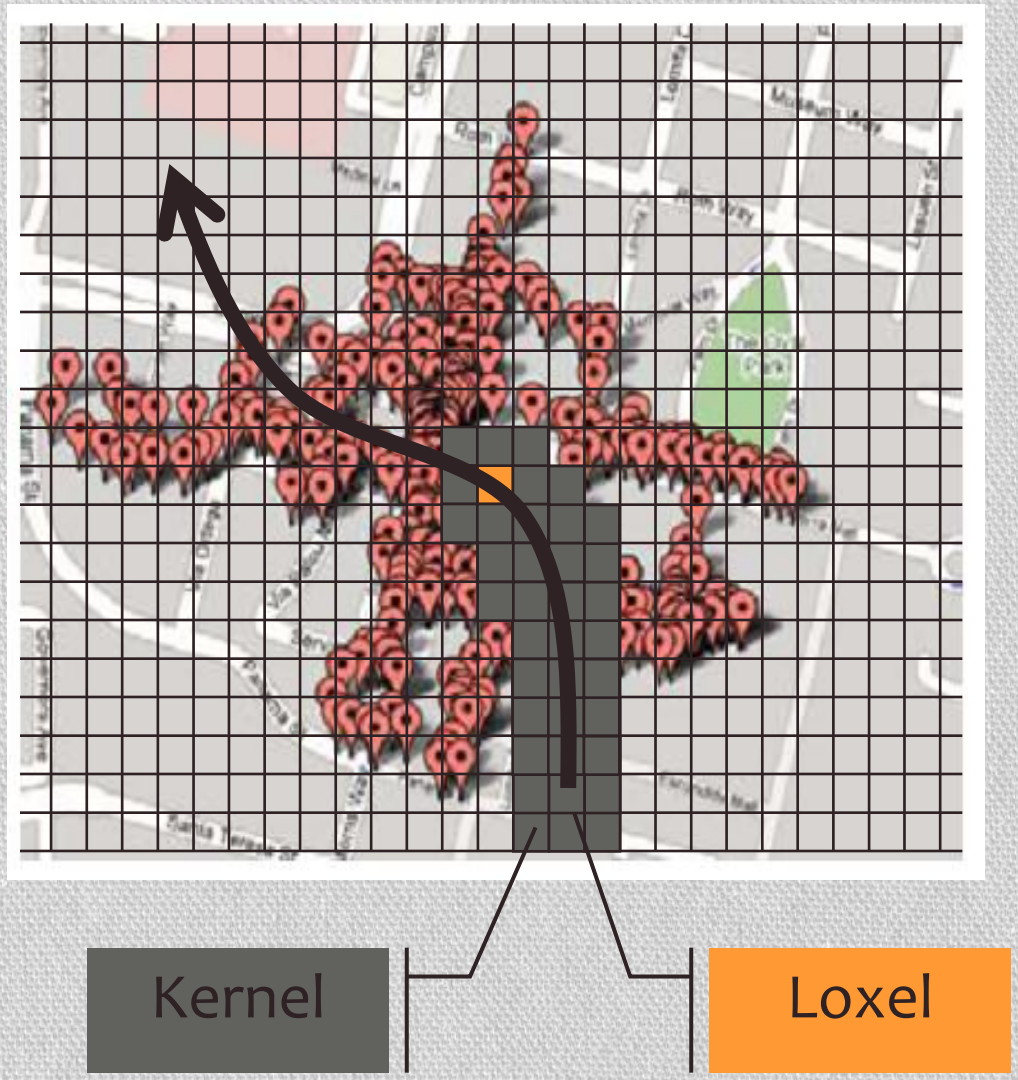
Image Matching



Data Organization & Loxels

- The map is divided into grid cells which are called loxels, location cells.
- Some of the closest loxels are kept in memory, this is called "setting a kernel". In the example shown next, a kernel is a 3x3 set of loxels, in some other situations, it is possible to have even more (e.g., 5x5).
- As the user moves, the kernel is updated by loading in some new loxels and forgetting some of the earlier loxels.

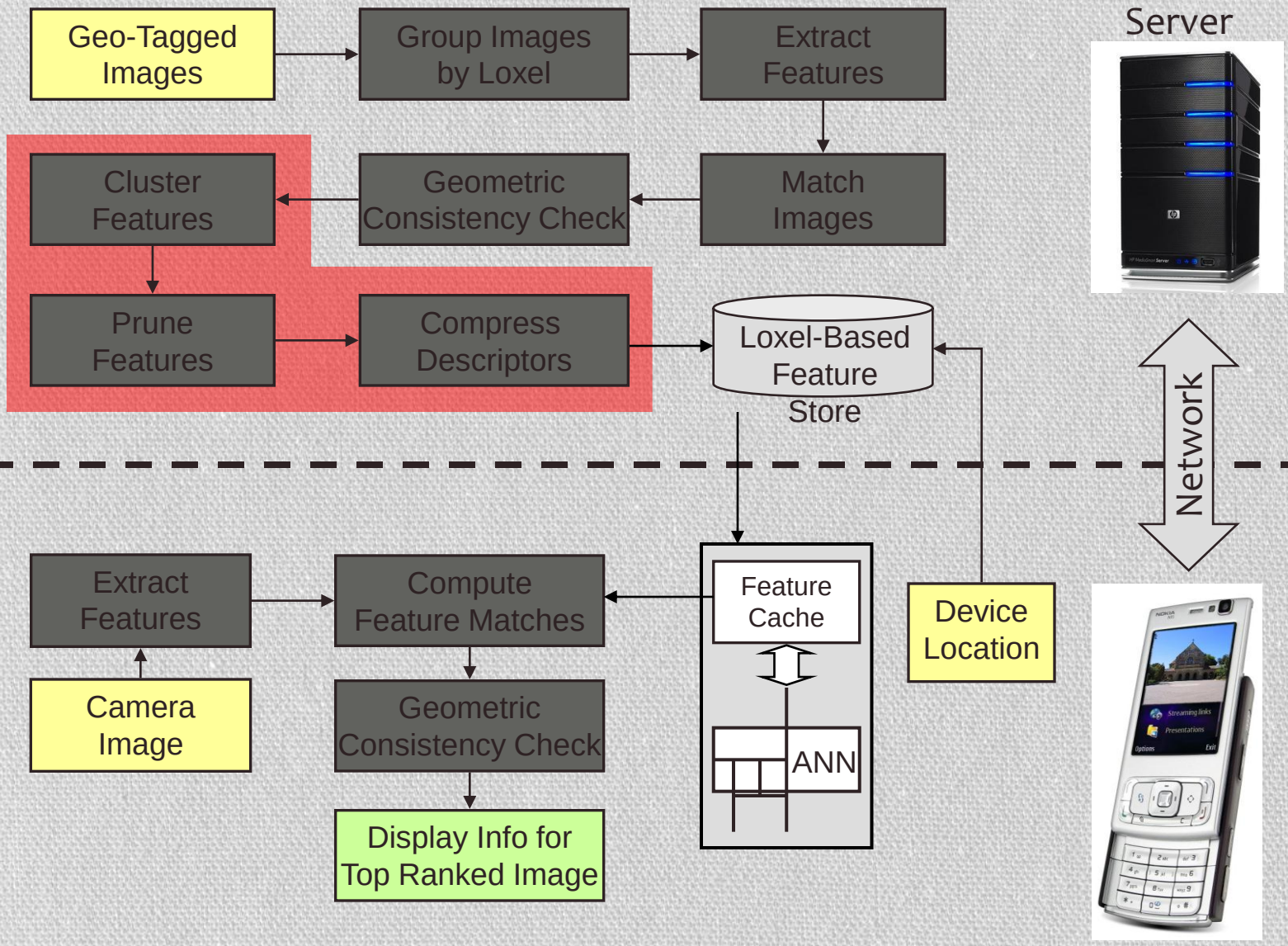
Data Organization



The Overall System

- Device sends location to server, the server returns the image features of the current kernel to the phone, where they are organized into a search structure (KD-tree, ANN = fast Approximate Nearest Neighbor search).
- Camera takes an image, extracts features, compares to features in the cache, runs RANSAC to find out whether individually matched features are consistent with other matches, and finally displays the information of the top ranked image.
- The database on the server is built by collecting a set of geotagged and POI (point-of-interest) tagged images, which are then grouped together based on their location and allocated to loxels, features are extracted, and then we try to match images to other images in the loxel.
- A similar RANSAC consistency check is run, as during run-time on the phone, cluster features that tend to easily match features in other images, and mark features that cannot be matched again as low-priority features that are sent to handset last, if at all. The features are compressed and stored in the database.

System Block Diagram

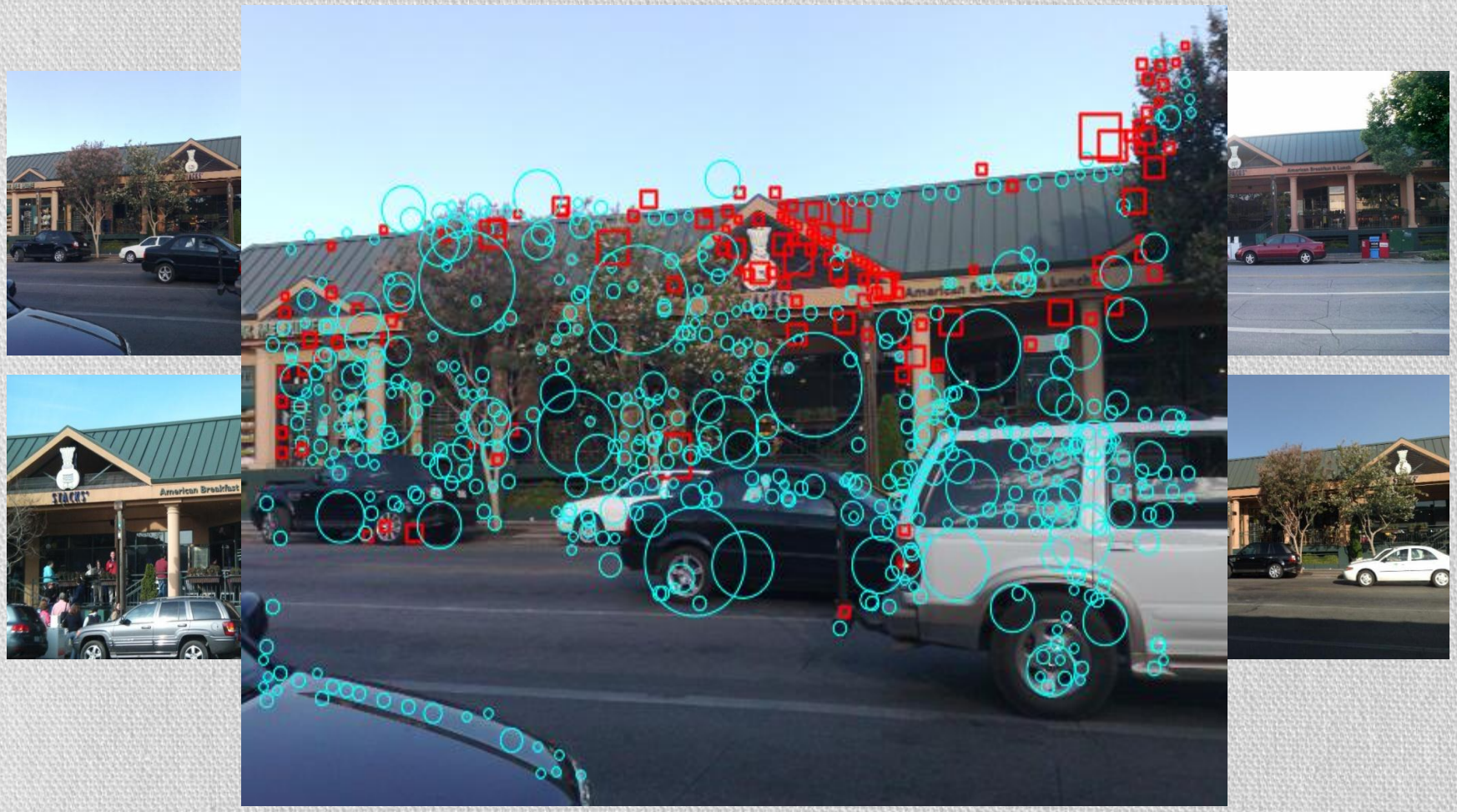


Feature Descriptor Clustering

- Match all images in loxel
- Form graph on features
- Cut graph into clusters
- Create representative meta-features

Example Feature Cluster

Feature Descriptor Clustering

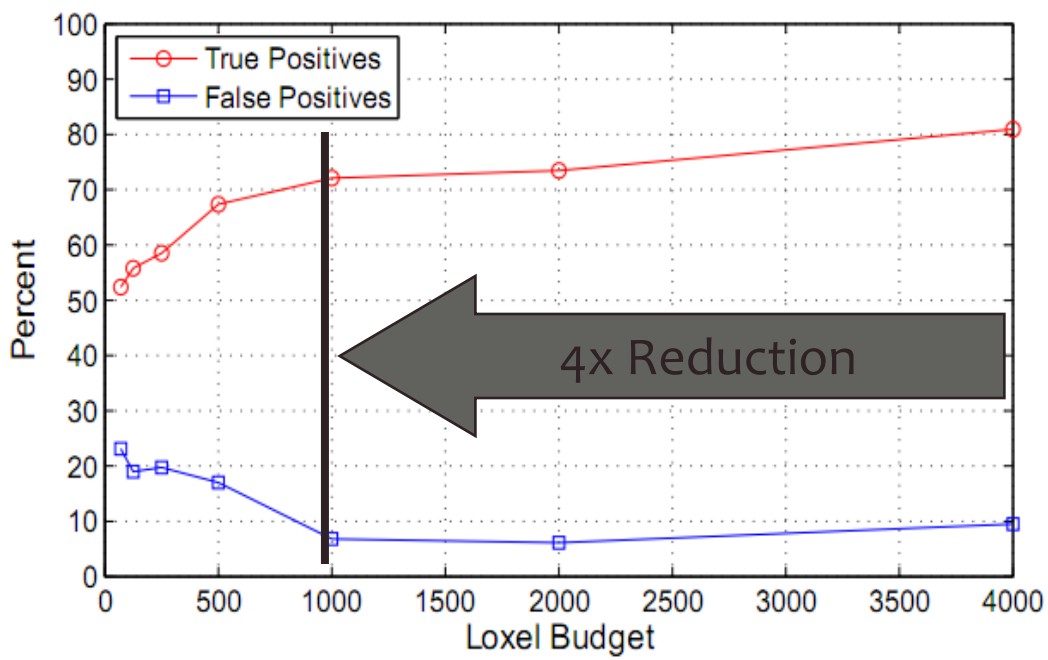


meta-feature

single feature

Database Feature Pruning

- Rank images by number of meta-features
- Allocate equal budget for each landmark
- Fill budget with meta-features by rank
- Fill any remaining budget with single features

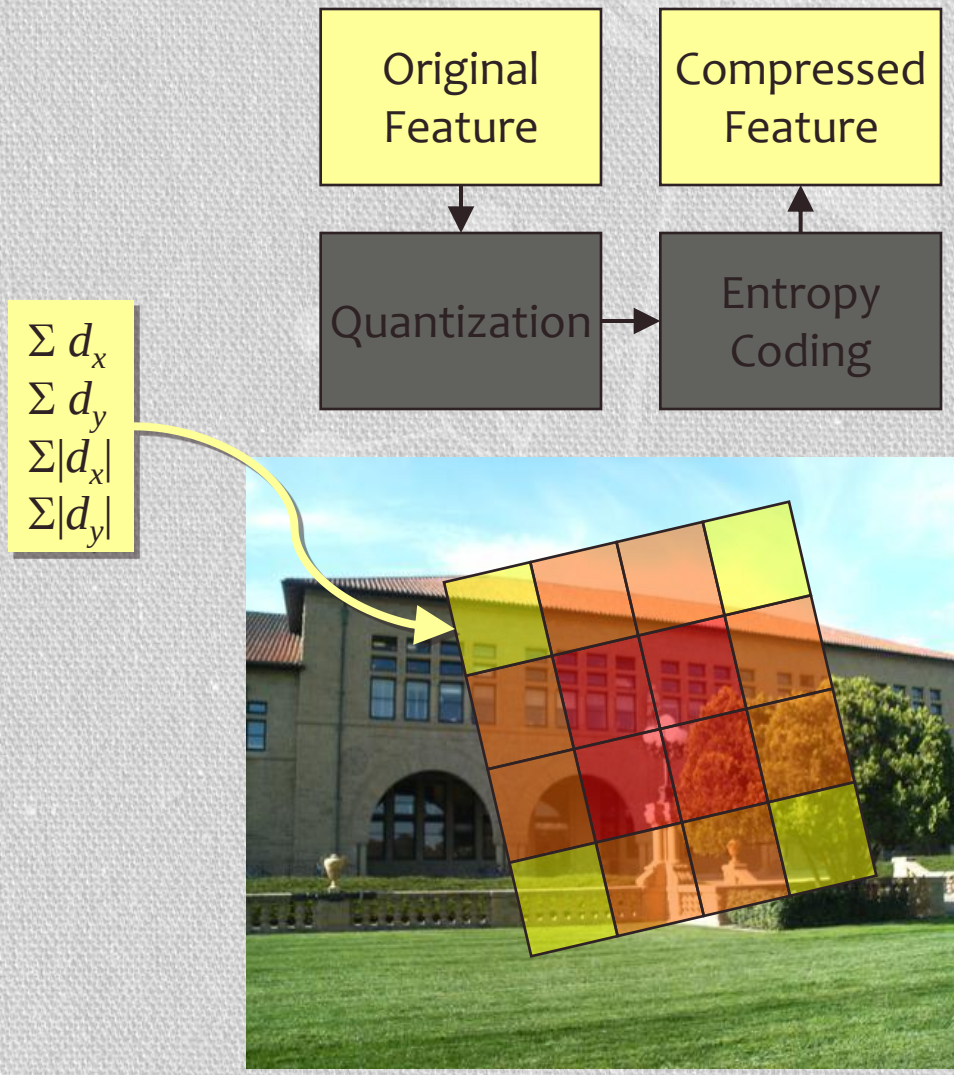


Feature Descriptor Pruning

Budget: 300

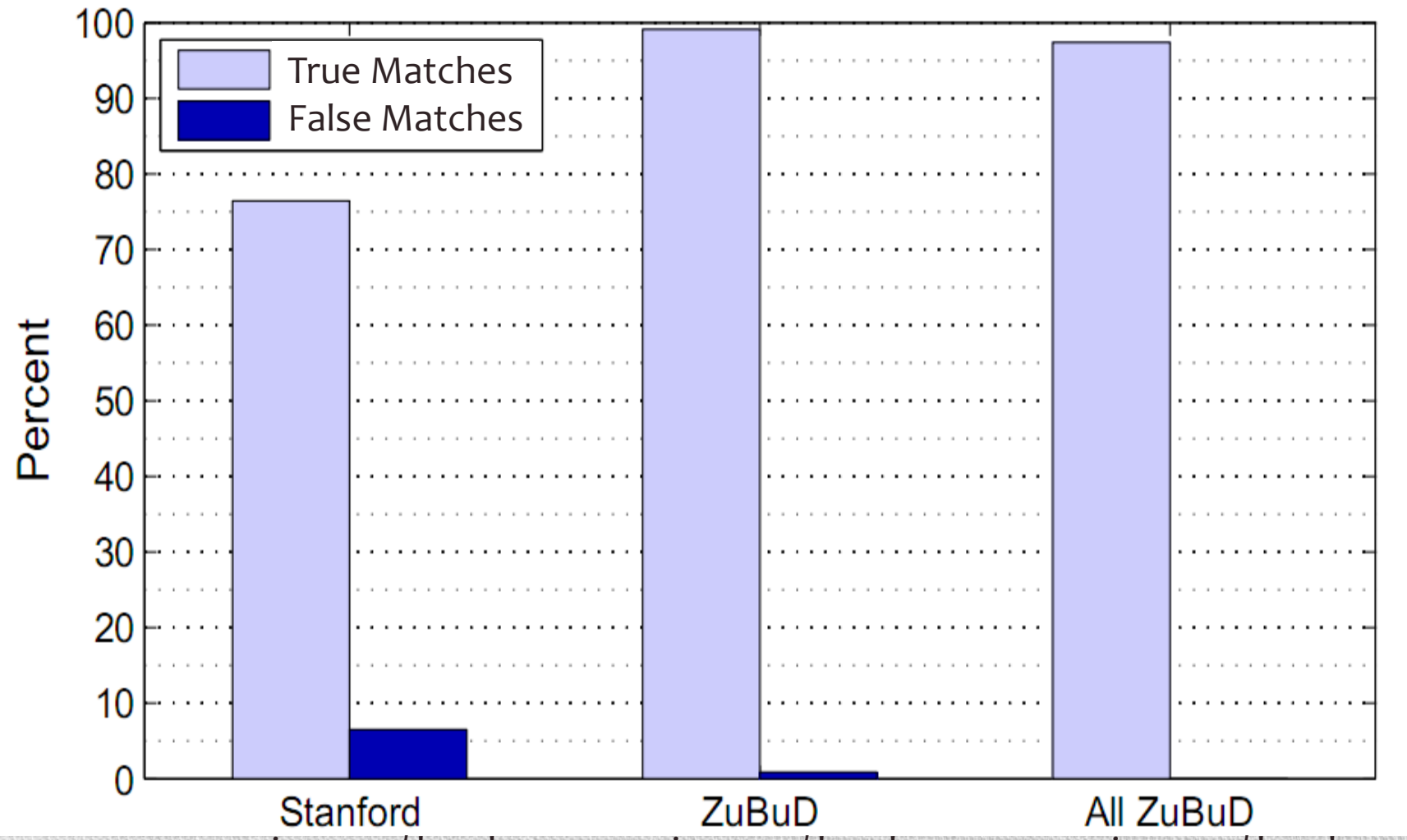
Feature Compression

- Quantization
  - Uniform, equal step-size
  - 6 bits per dimension
- Entropy coding
  - Huffman tables
  - 12 different tables
- 64-dimensional SURF
  - 256 bytes uncompressed
  - 37 bytes with compression



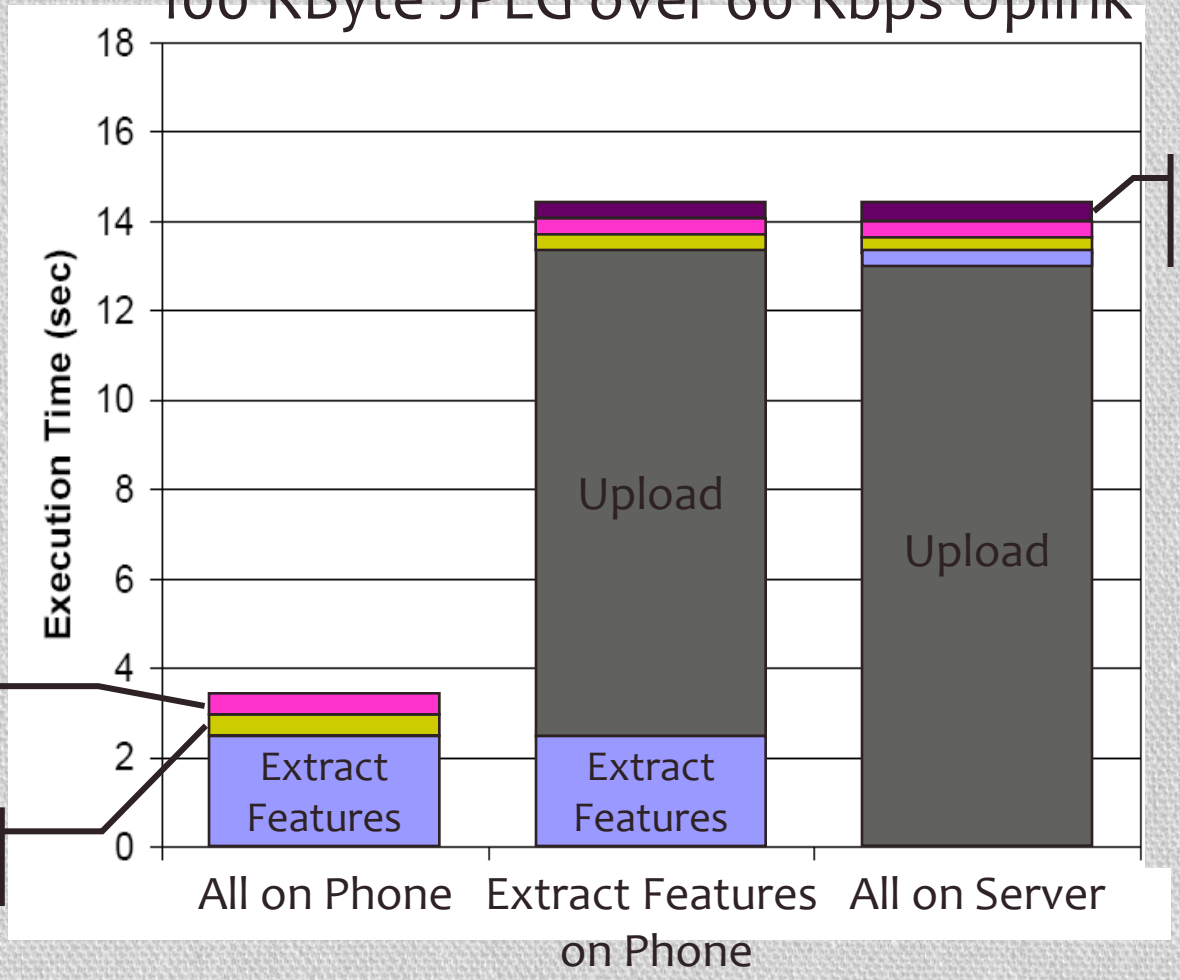
Matching Results

Matching Performance



Timing Analysis

Nokia N95
332 MHz ARM
64 MB RAM



Conclusions

- Image matching on mobile phone
- Use loxels to reduce search space
- 27x reduction in data sent to phone
  - Clustering
  - Pruning
  - Compression
- ~3 seconds for image matching on N95

The END

Questions?