Visual Recognition: Instances and Categories

Raquel Urtasun

TTI Chicago

Jan 24, 2012

Instance-level recognition

Instance recognition

- Motivation – visual search
- Visual words: quantization, inverted index, bags of words
- Spatial verification: RANSAC, Hough
- Other text retrieval tools: tf-idf
- Example applications

Recognizing or retrieving specific objects

- Example: Visual search in feature films

Visually defined query

“Groundhog Day” [Rammis, 1993]

“Find this clock”

“Find this place”

[Source: J. Sivic]

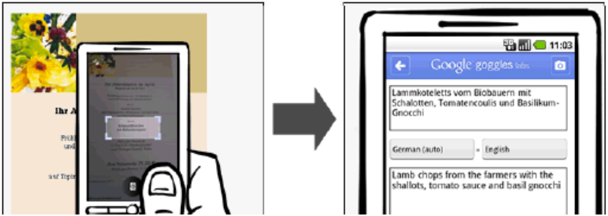
Recognizing or retrieving specific objects

- Example: Search photos on the web for particular places

Find these landmarks ...in these images and 1M more

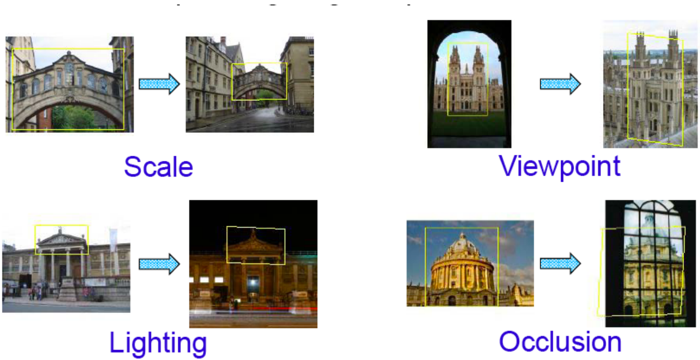
[Source: J. Sivic]


Google Goggles
Use pictures to search the web.
Watch a video
Get Google Goggles
Android (1.6+ required)
Download from Android Market.
Send Goggles to Android phone
New! iPhone (iOS 4.0 required)
Download from the App Store.
Send Goggles to iPhone
New!
Text
Landmarks
Books
Contact Info
Artwork
Wine
Logos


11:03
Google goggles labs
Lammkoteletts vom Biobauern mit Schalotten, Tomatencoulis und Basilikum-Gnocchi
German (auto)
English
Lamb chops from the farmers with the shallots, tomato sauce and basil gnocchi

Why is it difficult?

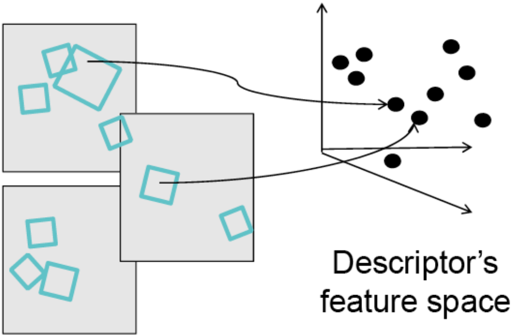
- Want to find the object despite possibly large changes in scale, viewpoint, lighting and partial occlusion.
- We can't expect to match such varied instances with a single global template...



[Source: J. Sivic]

Indexing local features

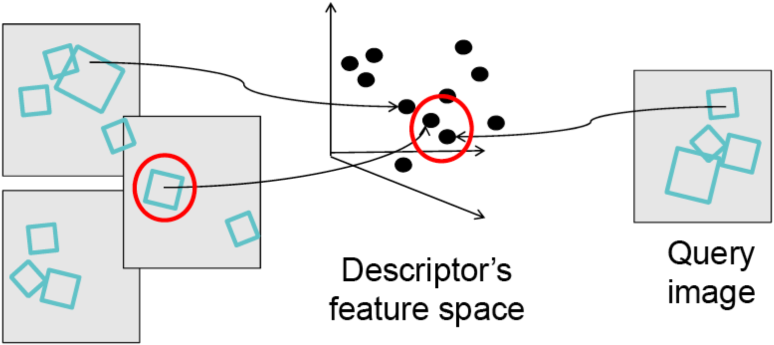
- Each patch / region has a descriptor, which is a point in some high-dimensional feature space (e.g., SIFT)



[Source: K. Grauman]

Indexing local features

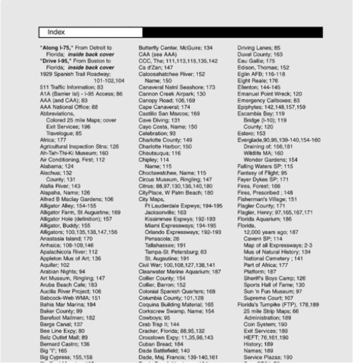
- It can have millions of features to search.



[Source: K. Grauman]

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, well need to map our features to visual words.
- Why?



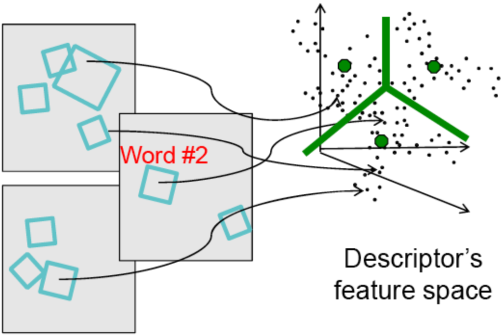
Index

"Along I-75," From Detroit to
  Florida; ***inside back cover***
"Drive I-95," From Boston to
  Florida; ***inside back cover***
1929 Spanish Trail Roadway; 101-102,104
511 Traffic Information; 83
A1A (Barrier Isl) - I-95 Access; 86
AAA (and CAA); 83
AAA National Office; 88
Abbreviations,
  Colored 25 mile Maps; cover
  Exit Services; 196
  Travelogue; 85
Africa; 177
Agricultural Inspection Stns; 126
Ah-Tah-Thi-Ki Museum; 160
Air Conditioning, First; 112
Alabama; 124
Alachua; 132
  County; 131
Alafia River; 143
Alapaha, Name; 126
Alfred B Maclay Gardens; 106
Alligator Alley; 154-155
Alligator Farm, St Augustine; 169
Alligator Hole (definition); 157
Alligator, Buddy; 155
Alligators; 100,135,138,147,156
Anastasia Island; 170
Anhaica; 108-109,146
Apalachicola River; 112
Appleton Mus of Art; 136
Aquifer; 102
Arabian Nights; 94
Art Museum, Ringling; 147
Aruba Beach Cafe; 183
Aucilla River Project; 106
Babcock-Web WMA; 151
Bahia Mar Marina; 184
Baker County; 99
Barefoot Mailmen; 182
Barge Canal; 137
Bee Line Expy; 80
Belz Outlet Mall; 89
Bernard Castro; 136
Big "I"; 165
Big Cypress; 155,158
Butterfly Center, McGuire; 134
CAA (see AAA)
CCC, The; 111,113,115,135,142
Ca d'Zan; 147
Caloosahatchee River; 152
  Name; 150
Canaveral Natnl Seashore; 173
Cannon Creek Airpark; 130
Canopy Road; 106,169
Cape Canaveral; 174
Castillo San Marcos; 169
Cave Diving; 131
Cayo Costa, Name; 150
Celebration; 93
Charlotte County; 149
Charlotte Harbor; 150
Chautauqua; 116
Chipley; 114
  Name; 115
Choctawatchee, Name; 115
Circus Museum, Ringling; 147
Citrus; 88,97,130,136,140,180
CityPlace, W Palm Beach; 180
City Maps,
  Ft Lauderdale Expwys; 194-195
  Jacksonville; 163
  Kissimmee Expwys; 192-193
  Miami Expressways; 194-195
  Orlando Expressways; 192-193
  Pensacola; 26
  Tallahassee; 191
  Tampa-St. Petersburg; 63
  St. Augsutine; 191
Civil War; 100,108,127,138,141
Clearwater Marine Aquarium; 187
Collier County; 154
Collier, Barron; 152
Colonial Spanish Quarters; 168
Columbia County; 101,128
Coquina Building Material; 165
Corkscrew Swamp, Name; 154
Cowboys; 95
Crab Trap II; 144
Cracker, Florida; 88,95,132
Crosstown Expy; 11,35,98,143
Cuban Bread; 184
Dade Battlefield; 140
Dade, Maj. Francis; 139-140,161
Driving Lanes; 85
Duval County; 163
Eau Gallie; 175
Edison, Thomas; 152
Eglin AFB; 116-118
Eight Reale; 176
Ellenton; 144-145
Emanuel Point Wreck; 120
Emergency Callboxes; 83
Epiphytes; 142,148,157,159
Escambia Bay; 119
  Bridge (I-10); 119
  County; 120
Estero; 153
Everglade,90,95,139-140,154-160
  Draining of; 156,181
  Wildlife MA; 160
  Wonder Gardens; 154
Falling Waters SP; 115
Fantasy of Flight; 95
Fayer Dykes SP; 171
Fires, Forest; 166
Fires, Prescribed ; 148
Fisherman's Village; 151
Flagler County; 171
Flagler, Henry; 97,165,167,171
Florida Aquarium; 186
Florida,
  12,000 years ago; 187
  Cavern SP; 114
  Map of all Expressways; 2-3
  Mus of Natural History; 134
  National Cemetery ; 141
  Part of Africa; 177
  Platform; 187
  Sheriff's Boys Camp; 126
  Sports Hall of Fame; 130
  Sun 'n Fun Museum; 97
  Supreme Court; 107
Florida's Turnpike (FTP), 178,189
  25 mile Strip Maps; 66
  Administration; 189
  Coin System; 190
  Exit Services; 189
  HEFT; 76,161,190
  History; 189
  Names; 189
  Service Plazas; 190

[Source: K. Grauman]

Indexing local features: inverted file index

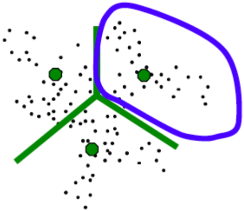
- Map high-dimensional descriptors to tokens/words by quantizing the feature space.
- Quantize via clustering, let cluster centers be the prototype words.
- Determine which word to assign to each new image region by finding the closest cluster.

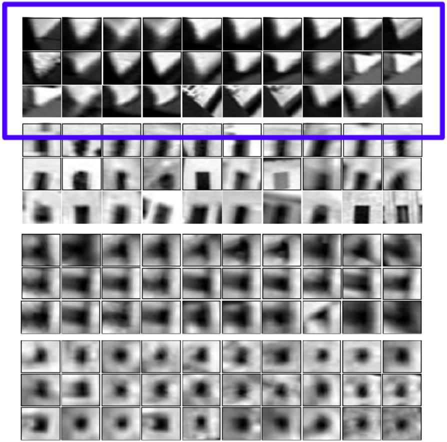


[Source: K. Grauman]

Visual words

- Each group of patches belongs to the same visual word.





Visual vocabulary formation issues

- Vocabulary size, number of words.
- Sampling strategy: where to extract features?

Visual vocabulary formation issues

- Vocabulary size, number of words.
- Sampling strategy: where to extract features?
- Clustering / quantization algorithm.

Visual vocabulary formation issues

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

Visual vocabulary formation issues

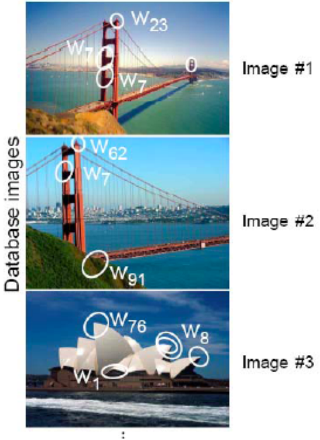
- Vocabulary size, number of words.
- Sampling strategy: where to extract features?
- Clustering / quantization algorithm.
- Unsupervised vs. supervised.
- What corpus provides features (universal vocabulary?)

Visual vocabulary formation issues

- Vocabulary size, number of words.
- Sampling strategy: where to extract features?
- Clustering / quantization algorithm.
- Unsupervised vs. supervised.
- What corpus provides features (universal vocabulary?)

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

...

What else do we need to do?

- How to summarize the content of an entire image? And gauge overall similarity?
- How large should the vocabulary be? How to perform quantization efficiently?

What else do we need to do?

- 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?

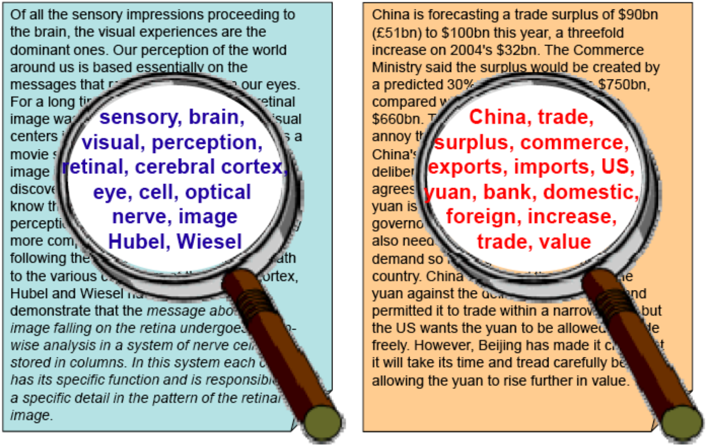
What else do we need to do?

- 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?

What else do we need to do?

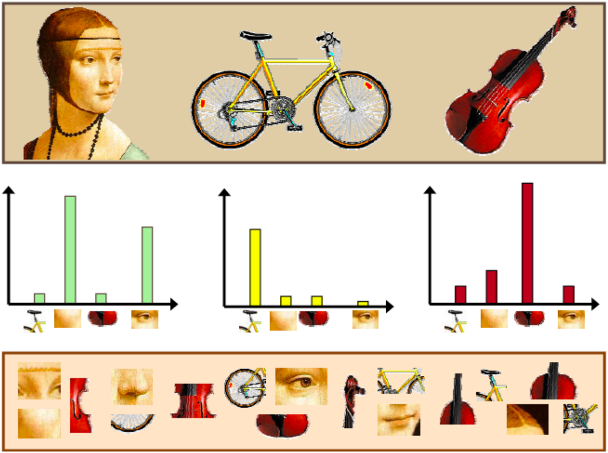
- How to summarize the content of an entire image? And gauge overall similarity?
- How large should the vocabulary be? How to perform quantization efficiently?
- Is having the same set of visual words enough to identify the object/scene? How to verify spatial agreement?
- How to score the retrieval results?

Relation to Documents



Bags of visual words

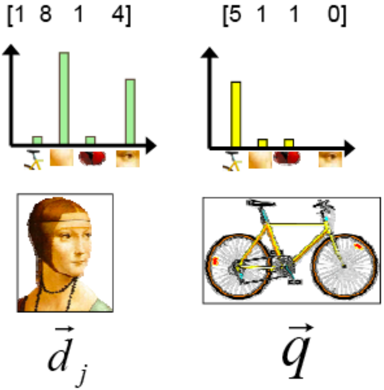
- Summarize entire image based on its distribution (histogram) of word occurrences.
- Analogous to bag of words representation commonly used for documents.



Comparing visual Words

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

$$\text{sim}(d_j, q) = \frac{< d_j, q >}{||d_j|| \cdot ||q||}$$



What else do we need to do?

- 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?

Vocabulary Trees

- Hierarchical clustering for large vocabularies, [Nister et al., 06].
- k defines the branch factor (number of children of each node) of the tree.

Vocabulary Trees

- Hierarchical clustering for large vocabularies, [Nister et al., 06].
- k defines the branch factor (number of children of each node) of the tree.
- First, an initial k- means process is run on the training data, defining k cluster centers.

Vocabulary Trees

- Hierarchical clustering for large vocabularies, [Nister et al., 06].
- k defines the branch factor (number of children of each node) of the tree.
- First, an initial k- means process is run on the training data, defining k cluster centers.
- The same process is then recursively applied to each group.

Vocabulary Trees

- Hierarchical clustering for large vocabularies, [Nister et al., 06].
- k defines the branch factor (number of children of each node) of the tree.
- First, an initial k- means process is run on the training data, defining k cluster centers.
- The same process is then recursively applied to each group.
- The tree is determined level by level, up to some maximum number of levels L.

Vocabulary Trees

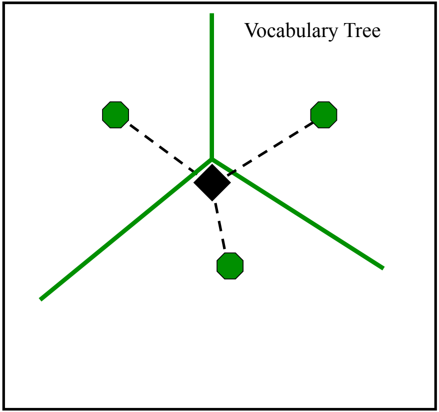
- Hierarchical clustering for large vocabularies, [Nister et al., 06].
- k defines the branch factor (number of children of each node) of the tree.
- First, an initial k- means process is run on the training data, defining k cluster centers.
- The same process is then recursively applied to each group.
- The tree is determined level by level, up to some maximum number of levels L.
- Each division into k parts is only defined by the distribution of the descriptor vectors that belong to the parent quantization cell.

Vocabulary Trees

- Hierarchical clustering for large vocabularies, [Nister et al., 06].
- k defines the branch factor (number of children of each node) of the tree.
- First, an initial k- means process is run on the training data, defining k cluster centers.
- The same process is then recursively applied to each group.
- The tree is determined level by level, up to some maximum number of levels L.
- Each division into k parts is only defined by the distribution of the descriptor vectors that belong to the parent quantization cell.

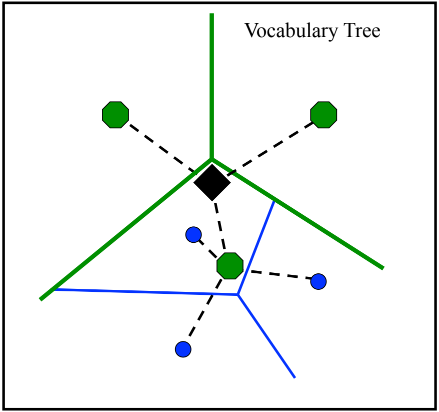
Constructing the tree

- Offline phase: hierarchical clustering.



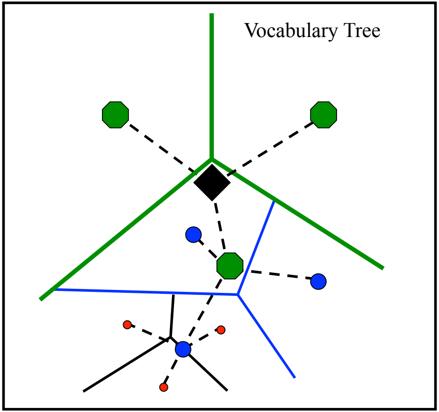
Constructing the tree

- Offline phase: hierarchical clustering.



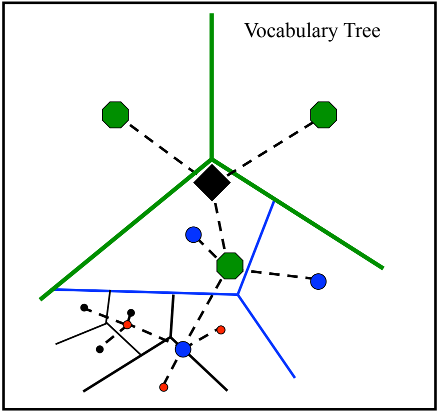
Constructing the tree

- Offline phase: hierarchical clustering.



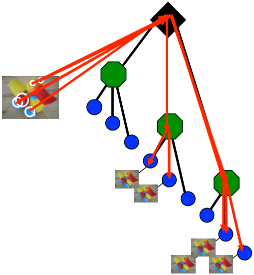
Constructing the tree

- Offline phase: hierarchical clustering.



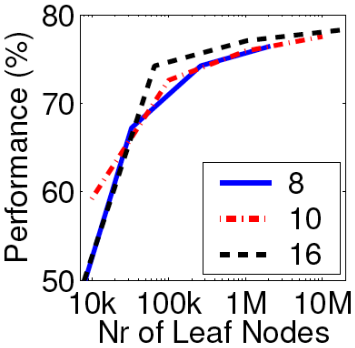
Parsing the tree

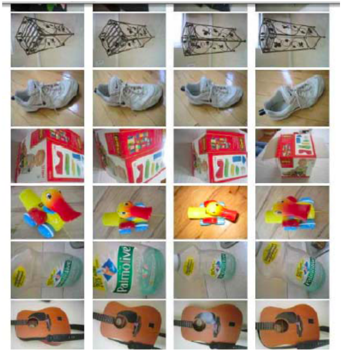
- Online phase: each descriptor vector is propagated down the tree by at each level comparing the descriptor vector to the k candidate cluster centers (represented by k children in the tree) and choosing the closest one.
- The tree directly defines the visual vocabulary and an efficient search procedure in an integrated manner.
- Every node in the vocabulary tree is associated with an inverted file.
- The inverted files of inner nodes are the concatenation of the inverted files of the leaf nodes (virtual).



Vocabulary size

- Complexity: branching factor and number of levels
- Most important for the retrieval quality is to have a large vocabulary





Visual words/bags of words

Good

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

Bad

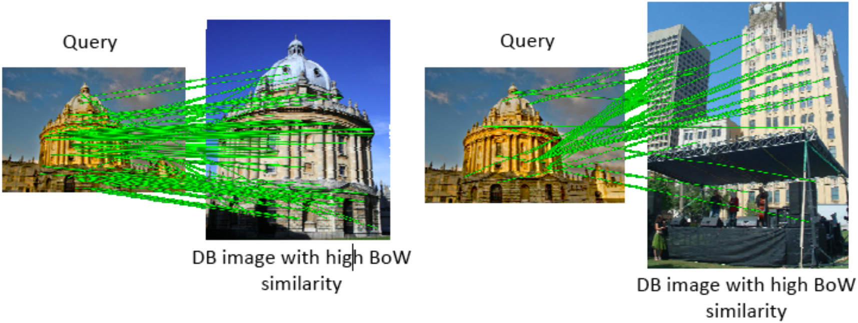
- background and foreground mixed when bag covers whole image
- optimal vocabulary formation remains unclear
- basic model ignores geometry must verify afterwards, or encode via features

What else do we need to do?

- 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?

Spatial Verification

- Both image pairs have many visual words in common
- Only some of the matches are mutually consistent



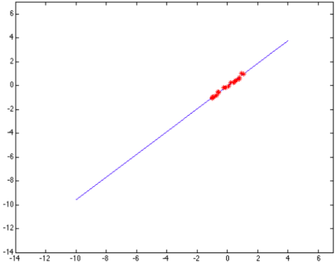
[Source: O. Chum]

Spatial Verification

Two basic strategies

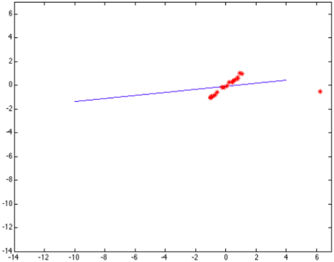
- RANSAC
- Generalized Hough Transform

Illustration: Least Squares Fit



[Source: K. Grauman]

Illustration: Least Squares Fit



[Source: K. Grauman]

RANSAC

- RANdom Sample Consensus.
- Approach: we want to avoid the impact of outliers, so lets look for inliers, and use those only.
- Intuition: if an outlier is chosen to compute the current fit, then the resulting line wont have much support from rest of the points.

RANSAC: General form

Loop

- Randomly select a seed group of points on which to base transformation estimate
- Compute model from seed group
- Find inliers to this transformation
- If the number of inliers is sufficiently large, re-compute estimate of model on all of the inliers

Keep the model with the largest number of inliers

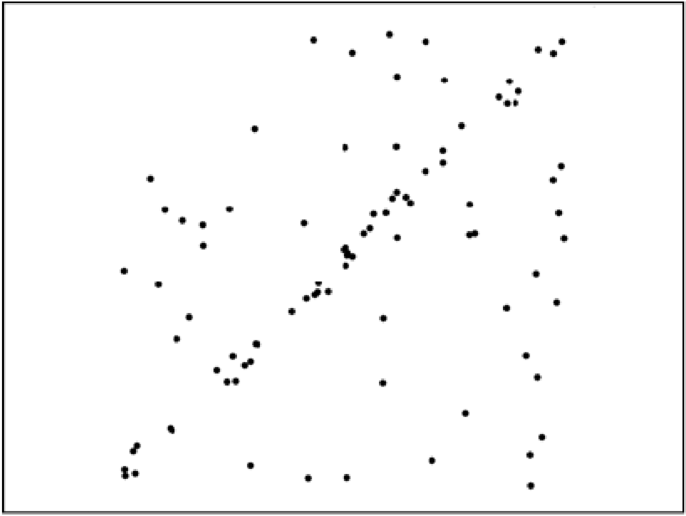
RANSAC for line fitting

Repeat:

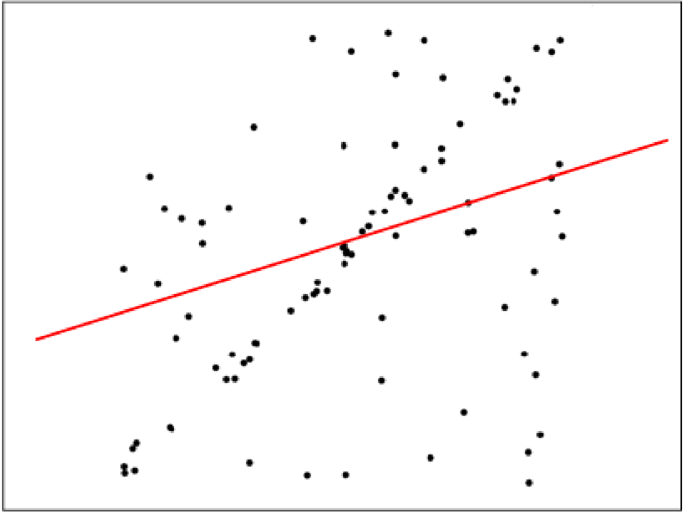
- Draw s points uniformly at random
- Fit line to these s points
- Find inliers to this line among the remaining points (i.e., points whose distance from the line is less than t)
- If there are d or more inliers, accept the line and refit using all inliers

[S. Lazebnik]

Example of line fitting

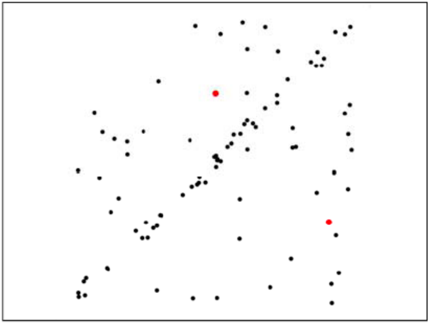


Example of line fitting



RANSAC for line fitting example

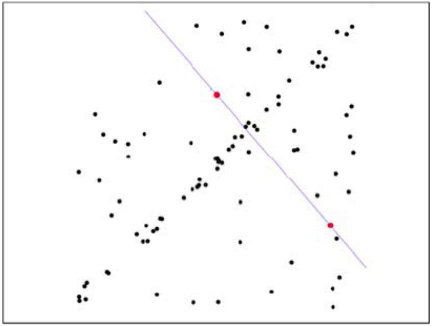
1. Randomly select minimal subset of points
2. Hypothesize a model



[Source: R. Raguram]

RANSAC for line fitting example

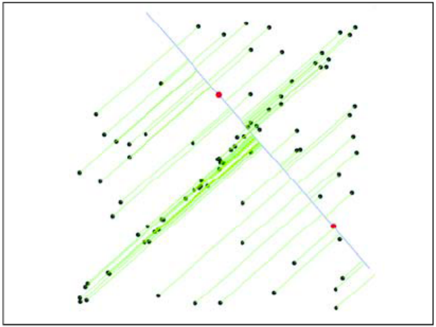
1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error function



[Source: R. Raguram]

RANSAC for line fitting example

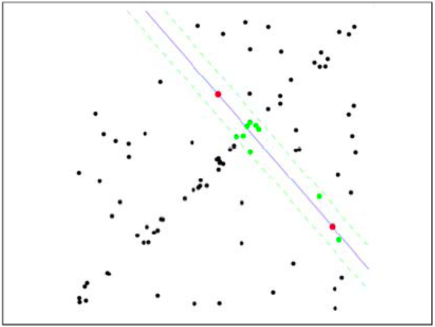
1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error function
4. Select points consistent with model



[Source: R. Raguram]

RANSAC for line fitting example

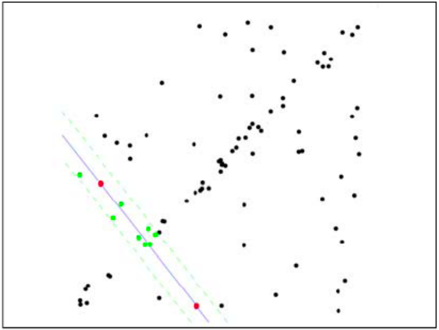
1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error function
4. Select points consistent with model
5. Repeat hypothesize and verify loop



[Source: R. Raguram]

RANSAC for line fitting example

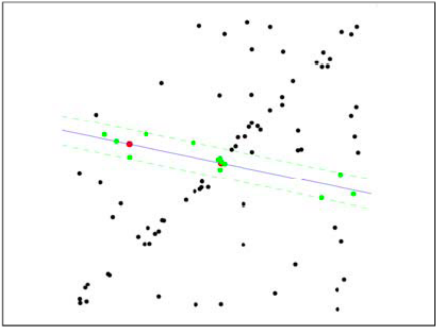
1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error function
4. Select points consistent with model
5. Repeat hypothesize and verify loop



[Source: R. Raguram]

RANSAC for line fitting example

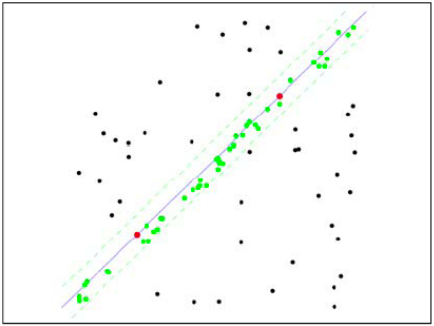
1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error function
4. Select points consistent with model
5. Repeat hypothesize and verify loop



[Source: R. Raguram]

RANSAC for line fitting example

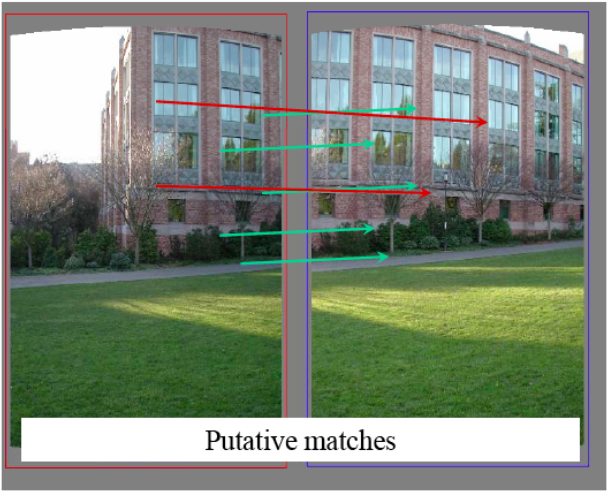
1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error function
4. Select points consistent with model
5. Repeat hypothesize and verify loop



[Source: R. Raguram]

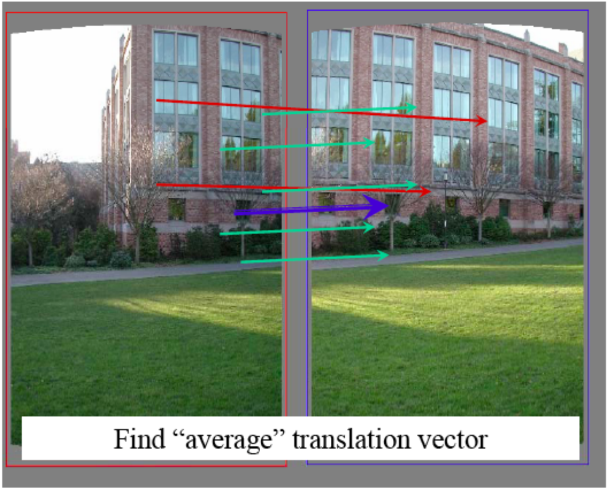
What about fitting a transformation?

- Select one match, count inliers



What about fitting a transformation?

- Select one match, count inliers



RANSAC

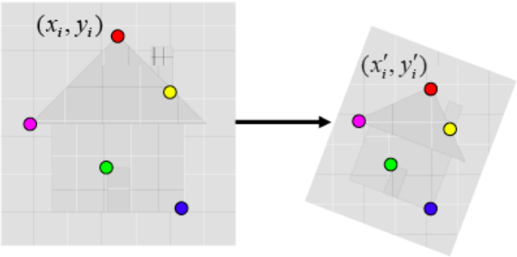
- Typically sort by BoW similarity as initial filter
- Verify by checking support (inliers) for possible transformations
- Success if find a transformation with $> N$ inlier correspondences

Fitting an affine transformation

- Approximates viewpoint changes for roughly planar objects and roughly orthographic cameras
- **Affine** is $\mathbf{p}' = \mathbf{A}\bar{\mathbf{p}}$, with $\mathbf{A}$ an arbitrary 2×3 matrix, i.e.,

$$\mathbf{p}' = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \end{bmatrix} \bar{\mathbf{p}}$$

- Parallel lines remain parallel under affine transformations.



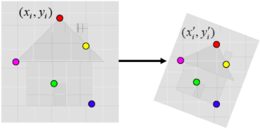
Fitting an affine transformation

- For all points

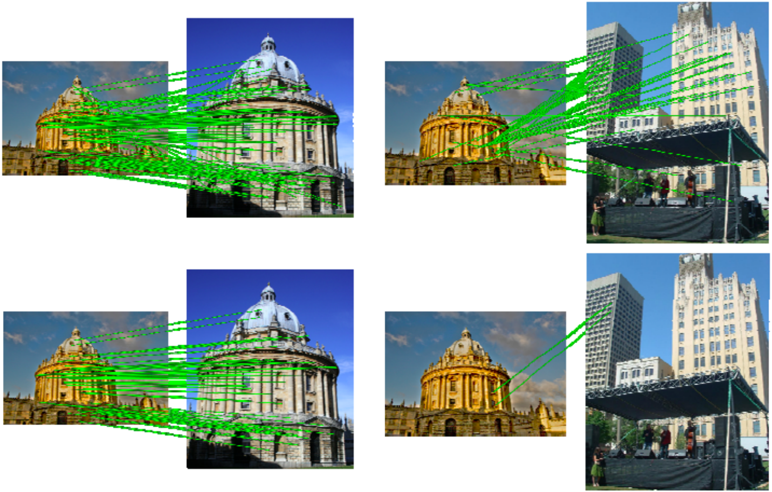
$$\underbrace{\begin{bmatrix} & & \vdots & & & \\ x_i & y_i & 0 & 0 & 1 & 0 \\ 0 & 0 & x_i & y_i & 0 & 1 \\ & & \vdots & & & \end{bmatrix}}_{\mathbf{P}} \underbrace{\begin{bmatrix} a_{00} \\ a_{01} \\ a_{02} \\ a_{10} \\ a_{11} \\ a_{12} \end{bmatrix}}_{\mathbf{a}} = \underbrace{\begin{bmatrix} \vdots \\ x_i' \\ y_i' \\ \vdots \end{bmatrix}}_{\mathbf{P}'}$$

- Least-squares fitting

$$\min_{a_{00},\cdots,a_{12}} ||\mathbf{Pa} - \mathbf{P}'||_2^2$$



Ransac Verification



[Source: K. Grauman]

Spatial Verification

Generalized Hough Transform

- Its not feasible to check all combinations of features by fitting a model to each possible subset.
- First, cycle through features, cast votes for model parameters: location, scale, orientation of the model object.

Spatial Verification

Generalized Hough Transform

- Its not feasible to check all combinations of features by fitting a model to each possible subset.
- First, cycle through features, cast votes for model parameters: location, scale, orientation of the model object.
- Look for model parameters that receive a lot of votes, and verify them.

Spatial Verification

Generalized Hough Transform

- Its not feasible to check all combinations of features by fitting a model to each possible subset.
- First, cycle through features, cast votes for model parameters: location, scale, orientation of the model object.
- Look for model parameters that receive a lot of votes, and verify them.
- Noise & clutter features will cast votes too, but their votes should be inconsistent with the majority of good features.

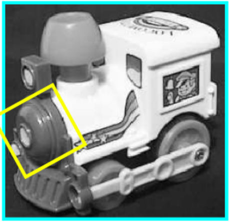
Spatial Verification

Generalized Hough Transform

- Its not feasible to check all combinations of features by fitting a model to each possible subset.
- First, cycle through features, cast votes for model parameters: location, scale, orientation of the model object.
- Look for model parameters that receive a lot of votes, and verify them.
- Noise & clutter features will cast votes too, but their votes should be inconsistent with the majority of good features.

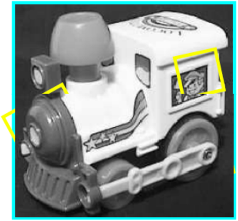
Generalized Hough Transform

- If we use scale, rotation, and translation invariant local features, then each feature match gives an alignment hypothesis (for scale, translation, and orientation of model in image).



[Source: S. Lazebnik]

- A hypothesis generated by a single match its in general unreliable,
- Let each match vote for a hypothesis in Hough space.



[Source: K. Grauman]

[Lowe 04]

- Training phase: For each model feature, record 2D location, scale, and orientation of model (relative to normalized feature frame)
- Test phase: Let each match between a test SIFT feature and a model feature vote in a 4D Hough space
  - Use broad bin sizes of 30 degrees for orientation, a factor of 2 for scale, and 0.25 times image size for location
  - Vote for two closest bins in each dimension

[Lowe 04]

- Training phase: For each model feature, record 2D location, scale, and orientation of model (relative to normalized feature frame)
- Test phase: Let each match between a test SIFT feature and a model feature vote in a 4D Hough space
  - Use broad bin sizes of 30 degrees for orientation, a factor of 2 for scale, and 0.25 times image size for location
  - Vote for two closest bins in each dimension
- Find all bins with at least three votes and perform geometric verification
  - Estimate least squares affine transformation
  - Search for additional features that agree with the alignment

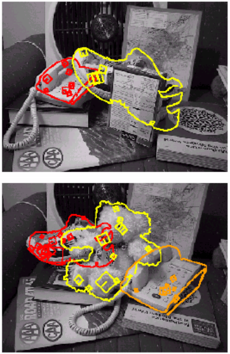
[Lowe 04]

- Training phase: For each model feature, record 2D location, scale, and orientation of model (relative to normalized feature frame)
- Test phase: Let each match between a test SIFT feature and a model feature vote in a 4D Hough space
  - Use broad bin sizes of 30 degrees for orientation, a factor of 2 for scale, and 0.25 times image size for location
  - Vote for two closest bins in each dimension
- Find all bins with at least three votes and perform geometric verification
  - Estimate least squares affine transformation
  - Search for additional features that agree with the alignment

Recognition Example

Background subtract for model boundaries

Objects recognized,



Recognition in spite of occlusion

Problems of Voting

- Noise/clutter can lead to as many votes as true target
- Bin size for the accumulator array must be chosen carefully

Problems of Voting

- Noise/clutter can lead to as many votes as true target
- Bin size for the accumulator array must be chosen carefully
- In practice, good idea to make broad bins and spread votes to nearby bins, since verification stage can prune bad vote peaks

Problems of Voting

- Noise/clutter can lead to as many votes as true target
- Bin size for the accumulator array must be chosen carefully
- In practice, good idea to make broad bins and spread votes to nearby bins, since verification stage can prune bad vote peaks

Comparison Verification

Generlized Hough Transform

- Each single correspondence votes for all consistent parameters
- Represents uncertainty on the parameter space
- Complexity: Beyond 4D space is impractical
- Can handle high outlier/inlier ratio

Ransac

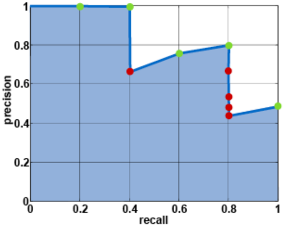
- Minimal subset of correspondences to estimate the model, then count inliers
- Represent uncertainty in image space
- Must look at all points to check for inliers at each iteration
- Scales better with high dimensionality of parameter space.

Scoring retrieval quality

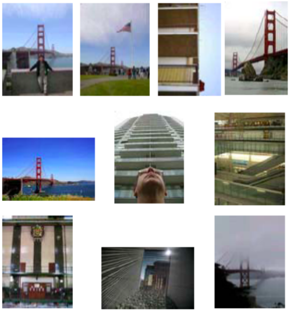
Query

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

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



Results (ordered):



[Source: O. Chum]

More from NLP community

tf-idf weighting

- Term frequency – inverse document frequency
- Describe frame by frequency of each word within it, downweight words that appear often in the database

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

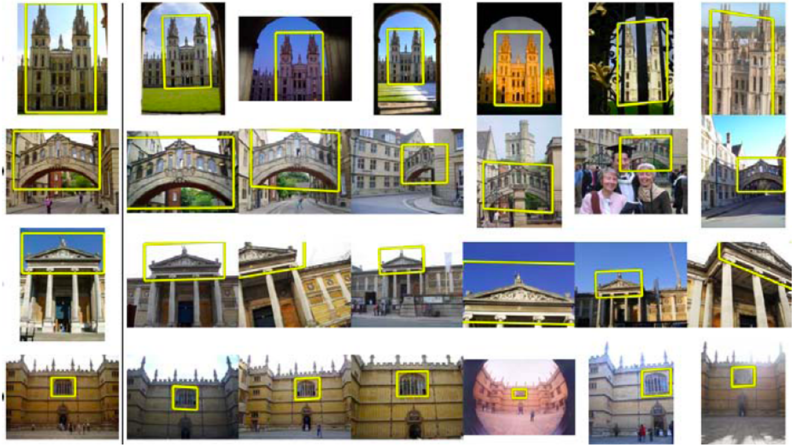
- n_{id} : number of occurrences of word i in document d
- n_d : number of words in document d
- N : total number of documents in the dataset
- n_i : number of documents word i occurs in (in the whole dataset)

Video Google System

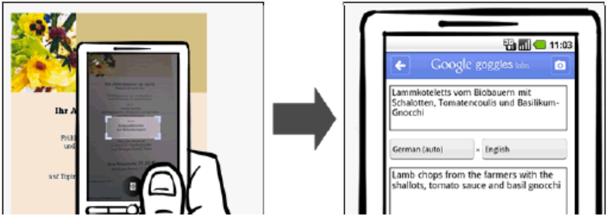
- Collect all words within query region
- Inverted file index to find relevant frames
- Compare word counts
- Spatial verification

[Philbin 07]

- Object retrieval with large vocabularies and fast spatial matching
- Results from 5k Flickr images (demo available for 100k set)




Google Goggles
Use pictures to search the web.
Watch a video
Get Google Goggles
Android (1.6+ required)
Download from Android Market.
Send Goggles to Android phone
New! iPhone (iOS 4.0 required)
Download from the App Store.
Send Goggles to iPhone
New!
Text
Landmarks
Books
Contact Info
Artwork
Wine
Logos


11:03
Google goggles
Lammkoteletts vom Biobauern mit Schalotten, Tomatencoulis und Basilikum-Gnocchi
German (auto)
English
Lamb chops from the farmers with the shallots, tomato sauce and basil gnocchi

Recognition via feature matching + spatial verification

Pros:

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

Cons:

- Scaling with number of models
- Spatial verification as post-processing – expensive for large-scale problems
- Not suited for category recognition

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

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

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, Generalized Hough Transform

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, Generalized Hough Transform

Category-level recognition

General recognition problem

Challenges

- Realistic scenes are crowded, cluttered, have overlapping objects

Recognition framework

- Build/train object model
  - Choose a representation
  - Learn or fit parameters of model / classifier
- Generate candidates in new image: only one for global scene classifiers
- Score the candidates

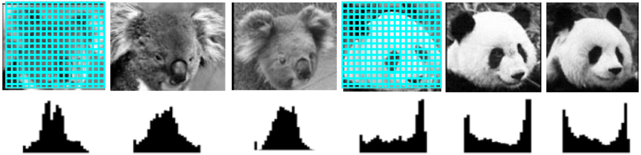
Representation Choice

Models can be divided on

- Window-based models: reason about the full object
- Part-based models: reason about parts and compose the information

Window-based model

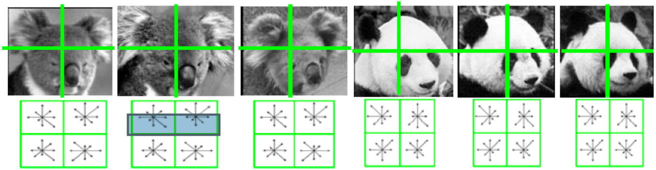
1. Holistic: vector of pixel intensities –template matching
2. Holistic: grayscale/color histogram



- Pixel-based representations sensitive to small shifts
- Color or grayscale-based appearance description can be sensitive to illumination and intra-class appearance variation
- Possible solution: Consider edges, contours, and (oriented) intensity gradients

Window-based model

1. Summarize local distribution of gradients with histogram
   - Locally orderless: offers invariance to small shifts and rotations
   - Contrast-normalization: try to correct for variable illumination



Which Classifier to use?

So many choices

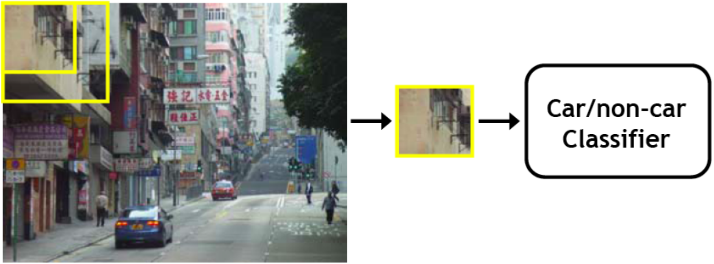
- Nearest Neighbors (NN)
- Support Vector Machines (SVMs)
- Gaussian processes (GPs)
- Boosting
- Neural networks
- Conditional Random Fields (CRFs)
- etc

Recognition framework

- Build/train object model
  - Choose a representation
  - Learn or fit parameters of model / classifier
- Generate candidates in new image: only one for global scene classifiers
- Score the candidates

Generating and scoring candidates

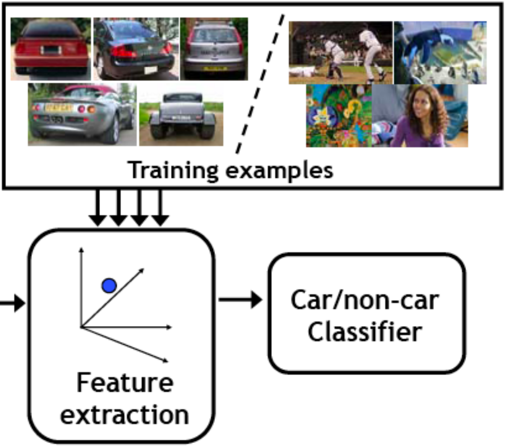
- Try every possible location: not very efficient.
- Work at different scales



Sliding Window Recap

Training:
1. Obtain training data
2. Define features
3. Define classifier

Given new image:
1. Slide window
2. Score by classifier



[Source: K. Grauman]

Choices and Issues

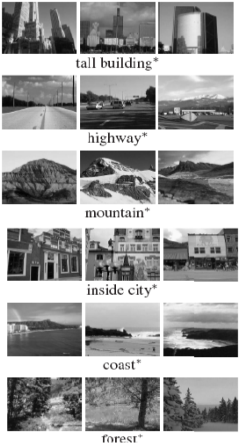
What classifier?

- Generative or discriminative model?
- Data resources – how much training data?
- How is the labeled data prepared?
- Training time allowance
- Test time requirements – real-time?
- Fit with the representation

Choices and Issues

- What classifier?
- What features or representations?
- How to make it affordable?
- What categories are amenable?
    - Similar to specific object matching, we expect spatial layout to be fairly rigidly preserved.
    - Unlike specific object matching, by training classifiers we attempt to capture intra-class variation or determine discriminative features.

What categories work well with sliding window?

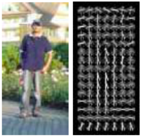


Which detectors?

Window-based

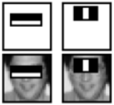
NN + scene Gist classification

e.g., Hays & Efros



SVM + person detection

e.g., Dalal & Triggs



Boosting + face detection

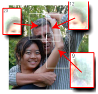
Viola & Jones

Part-based

BOW, pyramids
e.g., [Grauman et al.]

ISM: voting
e.g., [Leibe & Shiele]

deformable parts
e.g., [Felzenszwalb et al.]



poselets
[Bourdev et al.]

Example: Global representation

IM2GPS: estimating geographic information from a single image

James Hays and Alexei A. Efros
Carnegie Mellon University

Abstract

Estimating geographic information from an image is an excellent, difficult high-level computer vision problem whose time has come. The emergence of vast amounts of geographically-calibrated image data is a great reason for computer vision to start looking globally – on the scale of the entire planet! In this paper, we propose a simple algorithm for estimating a distribution over geographic locations from a single image using a purely data-driven scene matching approach. For this task, we will leverage a dataset of over 6 million GPS-tagged images from the Internet. We represent the estimated image location as a probability distribution over the Earth's surface. We quantitatively evaluate our approach in several geolocation tasks and demonstrate encouraging performance (up to 30 times better than chance). We show that geolocation estimates can provide the basis for numerous other image understanding tasks such as population density estimation, land cover estimation or urban/rural classification.

1. Introduction

Consider the photographs in Figure 1. What can you say about where they were taken? The first one is easy – it's an iconic image of the Notre Dame cathedral in Paris. The middle photo looks vaguely Mediterranean, perhaps a small

Figure 1. What can you say about where these photos were taken?

ical sea, sand and palm trees, we would simply remember: "I have seen something similar on a trip to Hawaii!". Note that although the original picture is unlikely to actually be from Hawaii, this association is still extremely valuable in helping to implicitly define the *type* of place that the photo belongs to.

Of course, computationally we are quite far from being able to semantically reason about a photograph (although encouraging progress is being made). On the other hand, the recent availability of truly gigantic image collections has made data association, such as brute-force scene matching, quite feasible [17, 4].

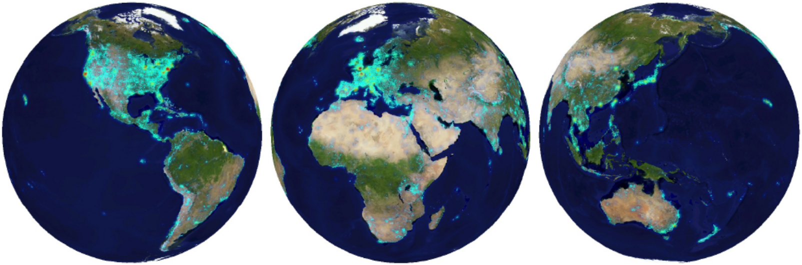
In this paper, we propose an algorithm for estimating a distribution over geographic locations from an image using a purely data-driven scene matching approach. For this task, we leverage a dataset of over 6 million GPS-tagged images from the Flickr online photo collection. We represent the es-

Where was this taken in the world?



Distribution of images

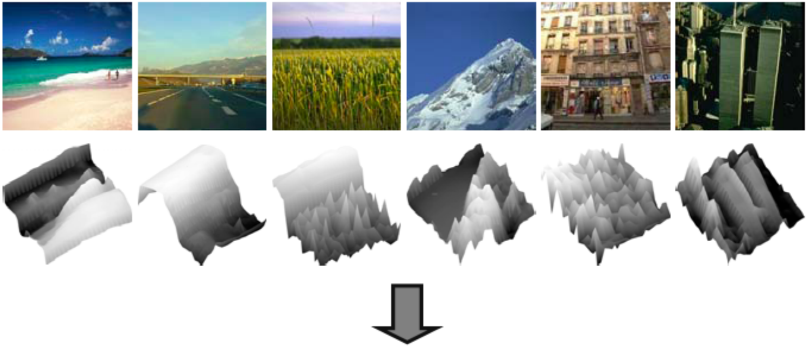
- Large collection of images from Flickr
- 6+ million geotagged photos by 109,788 photographers



Representation

- Color Histograms – L*A*B* 4x14x14 histograms, total of 784 dimensions.
- Texton Histograms – 512 entry, bank of filters with 8 orientations, 2 scales, and 2 elongations. For each image we then build a 512 dimensional histogram by assigning each pixel's set of filter responses to the nearest texton dictionary entry.
- Line Features – Histograms of straight line stats (line angles and line lenghts) to distinguishing between natural and man-made.
- Geometric context – compute the geometric class probabilities for image regions.
- Gist scene descriptor – 5 by 5 spatial resolution where each bin contains that image regions average response to steerable filters at 6 orientations and 4 scales.

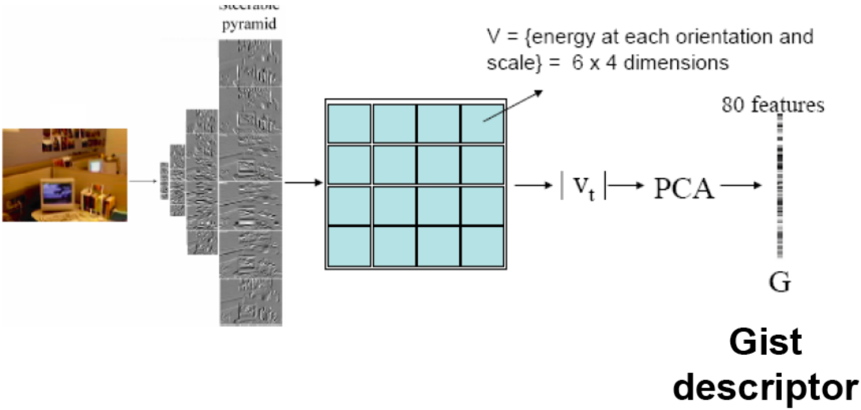
Spatial Envelope Theory of Scene Representation



A scene is a single surface that can be represented by global (statistical) descriptors

[Source: A. Oliva]

Spatial Envelope Theory of Scene Representation

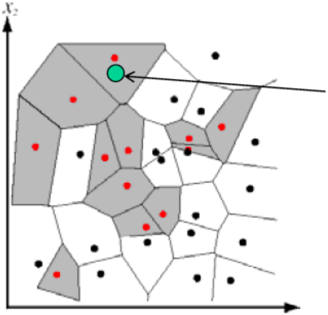


[Source: A. Oliva]

Classifier

- Assign label of nearest training data point to each test data point
- Voronoi partitioning of feature space for 2-category 2D data

Black = negative
Red = positive



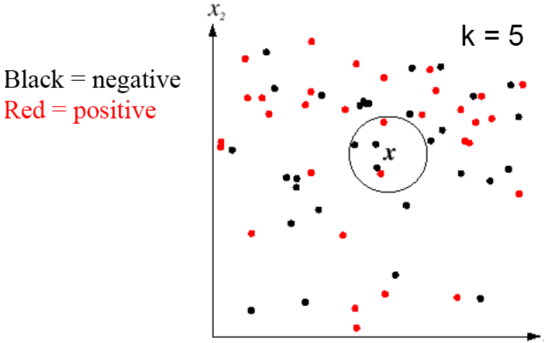
from Duda *et al.*

Novel test example

Closest to a positive example from the training set, so classify it as positive.

Classifier improvement

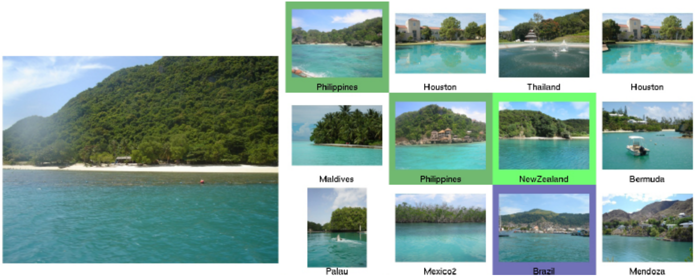
- For a new point, find the k closest points from training data
- Labels of the k points vote to classify



Qualitative Results

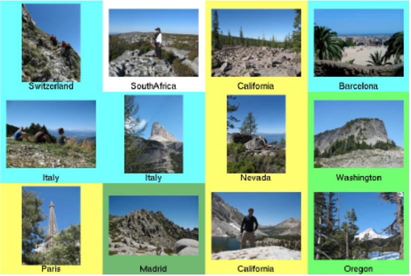
Qualitative Results

Qualitative Results

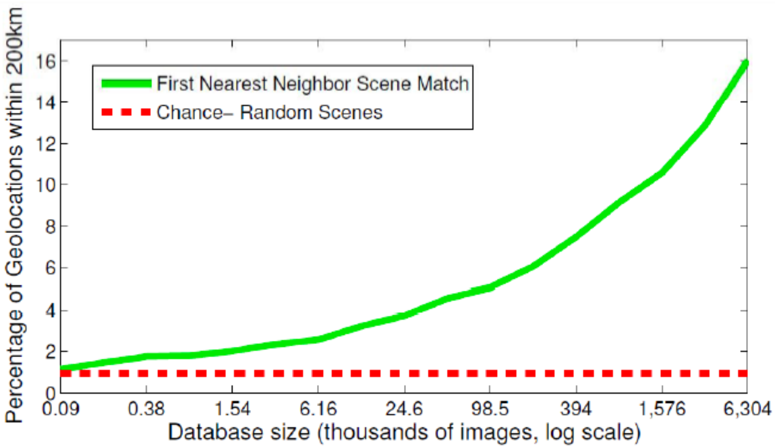


Qualitative Results

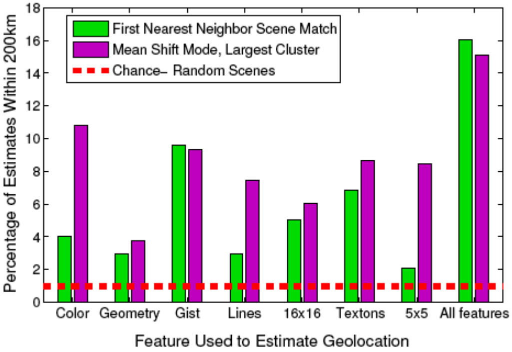
Qualitative Results



Results: size matters



- Multi-features: We scale each features distances so that their standard deviations are roughly the same and thus they influence the ordering of scene matches equally.



NN

Pros:

- Simple to implement
- Flexible to feature / distance choices
- Naturally handles multi-class cases
- Can do well in practice with enough representative data

Cons:

- Large search problem to find nearest neighbors, e.g., KD-trees, hashing, etc.
- Storage of data: non-parametric, we keep everything.
- Must know we have a meaningful distance function: metric learning

Example: Dalal & Triggs

Histograms of Oriented Gradients for Human Detection

Navneet Dalal and Bill Triggs
INRIA Rhône-Alps, 655 avenue de l'Europe, Montbonnot 38334, France
{Navneet.Dalal,Bill.Triggs}@inrialpes.fr, http://lear.inrialpes.fr

Abstract

We study the question of feature sets for robust visual object recognition, adopting linear SVM based human detection as a test case. After reviewing existing edge and gradient based descriptors, we show experimentally that grids of Histograms of Oriented Gradient (HOG) descriptors significantly outperform existing feature sets for human detection. We study the influence of each stage of the computation on performance, concluding that fine-scale gradients, fine orientation binning, relatively coarse spatial binning, and high-quality local contrast normalization in overlapping descriptor blocks are all important for good results. The new approach gives near-perfect separation on the original MIT pedestrian database, so we introduce a more challenging dataset containing over 1800 annotated human images with a large range of pose variations and backgrounds.

1 Introduction

We briefly discuss previous work on human detection in §2, give an overview of our method §3, describe our data sets in §4 and give a detailed description and experimental evaluation of each stage of the process in §5–6. The main conclusions are summarized in §7.

2 Previous Work

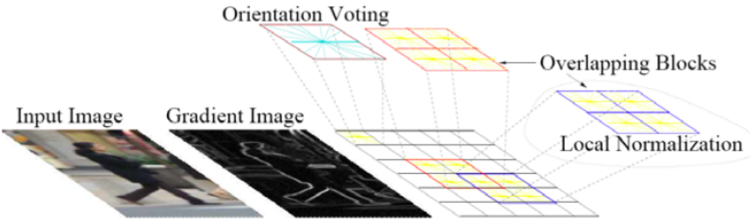
There is an extensive literature on object detection, but here we mention just a few relevant papers on human detection [18,17,22,16,20]. See [6] for a survey. Papageorgiou *et al* [18] describe a pedestrian detector based on a polynomial SVM using rectified Haar wavelets as input descriptors, with a parts (subwindow) based variant in [17]. Depoortere *et al* give an optimized version of this [2]. Gavrila & Philomen [8] take a more direct approach, extracting edge images and matching them to a set of learned exemplars using chamfer distance. This has been used in a practical real-time pedestrian detection system [7]. Viola *et al* [22] build an efficient

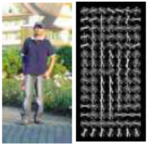
Task to solve

- Pedestrian detection

Representation

- Histogram of gradients: [Schiele & Crowley, Freeman & Roth]
- Code available: http://pascal.inrialpes.fr/soft/olt/



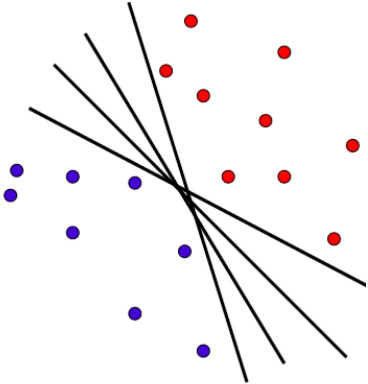


Linear Classifier

- Find linear function to separate positive and negative examples

$$f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$$

- $f(\mathbf{x}) > 0$ if $\mathbf{x}$ is a positive example.
- $f(\mathbf{x}) < 0$ if $\mathbf{x}$ is a positive example.



Learning Setup

- Input $\mathbf{x} \in \Re^D$, and outputs $y_i \in \{-1, 1\}$
- General setup: training set sampled i.i.d. from $p(\mathbf{x}, y)$, we want to find parametric predictor $f \in \mathcal{F}$ that minimizes

$$R(f) = E_{\mathbf{x}_0, y_0}[L(f(\mathbf{x}_0; \Theta), y_0)]$$

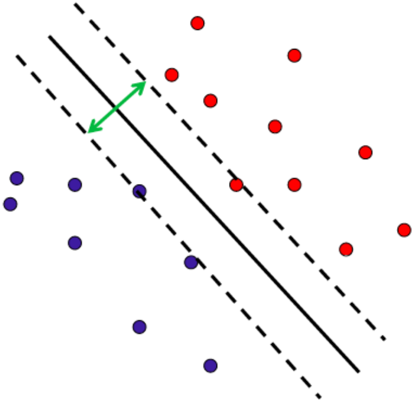
  with L the loss
- Regularized ERM:

$$\widehat{\theta} = \underset{\theta}{\text{argmin}} \sum_{i=1}^{N} L(f(\mathbf{x}_i; \theta), y_i) + R(\theta)$$

- Loss L: square loss (ridge regression, GP), hinge (SVM), log loss (logistic regression)

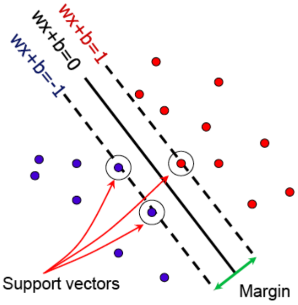
Linear Classifier

- Discriminative classifier based on optimal separating hyperplane
- Maximize the margin between the positive and negative training examples



Support Vector machines

- Maximize the margin between the positive and negative training examples



- Positive $\mathbf{y}_i = 1$: $\mathbf{w}^T\mathbf{x}_i + b \geq 1$
- Negative $\mathbf{y}_i = -1$: $\mathbf{w}^T\mathbf{x}_i + b \leq 1$
- Support vector: $\mathbf{x}_i \cdot \mathbf{w} + b =??1$
- Point line distance: $\frac{\mathbf{y}(\mathbf{w}^T\mathbf{x}+b)}{||\mathbf{w}||}$
- For support vectors: $\frac{1}{||\mathbf{w}||}$
- Margin $M = \frac{2}{||\mathbf{w}||}$

Find the max margin hyperplane

- Maximize the margin and classify all the points
- Quadratic optimization problem

$$\min_{\mathbf{w}} \quad \frac{1}{2}\|\mathbf{w}\|^2$$
$$\text{subject to} \;\; y_i(b + \mathbf{w}^T\mathbf{x}_i) - 1 \;\geq\; 0, \quad i = 1, \ldots, N.$$

- We will associate with each constraint the loss

$$\max_{\alpha \geq 0} \alpha \left[1 - y_i(b + \mathbf{w}^T\mathbf{x}_i)\right] = \begin{cases} 0, & \text{if } y_i \left(w_0 + \mathbf{w}^T\mathbf{x}_i\right) - 1 \geq 0, \\ \infty & \text{otherwise (constraint violated).} \end{cases}$$

- We can reformulate our problem now:

$$\min_{\mathbf{w}} \left\{ \frac{1}{2}\|\mathbf{w}\|^2 + \sum_{i=1}^{N} \max_{\alpha_i \geq 0} \alpha_i \left[1 - y_i(b + \mathbf{w}^T\mathbf{x}_i)\right] \right\}$$

Find the max margin hyperplane

- Maximize the margin and classify all the points
- Quadratic optimization problem

$$\min_{\mathbf{w}} \quad \frac{1}{2}\|\mathbf{w}\|^2$$
$$\text{subject to} \;\; y_i(b + \mathbf{w}^T\mathbf{x}_i) - 1 \;\geq\; 0, \quad i = 1, \ldots, N.$$

- We will associate with each constraint the loss

$$\max_{\alpha \geq 0} \alpha \left[1 - y_i(b + \mathbf{w}^T\mathbf{x}_i)\right] = \begin{cases} 0, & \text{if } y_i\left(w_0 + \mathbf{w}^T\mathbf{x}_i\right) - 1 \geq 0, \\ \infty & \text{otherwise (constraint violated).} \end{cases}$$

- We can reformulate our problem now:

$$\min_{\mathbf{w}} \left\{ \frac{1}{2}\|\mathbf{w}\|^2 + \sum_{i=1}^{N} \max_{\alpha_i \geq 0} \alpha_i \left[1 - y_i(b + \mathbf{w}^T\mathbf{x}_i)\right] \right\}$$