Recognition: Alignment and voting

Tuesday, Nov 3

Kristen Grauman
UT-Austin

Some pset3 results!
Previously

• Local invariant features for multi-view matching
Local features: main components

1) Detection: Identify the interest points

2) Description: Extract vector feature descriptor surrounding each interest point.

3) Matching: Determine correspondence between descriptors in two views

Local features: code

- Lots of nice code / binaries available online.
- Check class page for links.
Previously

- Local invariant features for multi-view matching
- Local features for (sub-)image retrieval

Visual words

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

Figure from Sivic & Zisserman, ICCV 2003
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.

Inverted file index

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

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

Review questions

- What are the tradeoffs related to the visual vocabulary size (number of words)?
- What is the role of tf-idf weighting for a bag-of-words representation?
- If we have established a vocabulary, and get a new image with some SIFT descriptors, how do we assign its features to words?
Today

- Introduction to object recognition problem
- Recognition by alignment, voting

What does object recognition involve?
Verification: is that a lamp?

Detection: are there people?
Identification: is that Potala Palace?

Object categorization

mountain

building

street lamp

vendor

people

tree

banner
Scene and context categorization

- outdoor
- city
- ...

What could be done with recognition algorithms?

There is a wide range of applications, including...

- Autonomous robots
- Navigation, driver safety
- Situated search
- Content-based retrieval and analysis for images and videos
- Medical image analysis
Object Categorization

• Task Description
  “Given a small number of training images of a category, recognize a-priori unknown instances of that category and assign the correct category label.”

• Which categories are feasible visually?
  Extensively studied in Cognitive Psychology, e.g. [Brown’58]

Visual Object Categories

• Basic Level Categories in human categorization
  [Rosch 76, Lakoff 87]
  The highest level at which category members have similar perceived shape
  The highest level at which a single mental image reflects the entire category
  The level at which human subjects are usually fastest at identifying category members
  The first level named and understood by children
  The highest level at which a person uses similar motor actions for interaction with category members
Visual Object Categories

- Basic-level categories in humans seem to be defined predominantly visually.
- There is evidence that humans (usually) start with basic-level categorization before doing identification.
  - Basic-level categorization is easier and faster for humans than object identification!
  - Most promising starting point for visual classification

How many object categories are there?

~10,000 to 30,000

Source: Fei-Fei Li, Rob Fergus, Antonio Torralba.
Other Types of Categories

• Functional Categories
  • e.g. chairs = “something you can sit on”
Other Types of Categories

- Ad-hoc categories
  - e.g. “something you can find in an office environment”

K. Grauman, B. Leibe

Challenges: robustness

- Illumination
- Object pose
- Clutter
- Occlusions
- Intra-class appearance
- Viewpoint
Challenges: robustness

- Detection in Crowded Scenes
  - Learn object variability
  - Changes in appearance, scale, and articulation
  - Compensate for clutter, overlap, and occlusion

Challenges: context and human experience
Challenges: context and human experience

Context cues

Image credit: D. Hoeim

Challenges: learning with minimal supervision

Less
Unlabeled, multiple objects

Classes labeled, some clutter

More
Cropped to object, parts and classes labeled
This is a pottopod

Find the pottopod
Levels of Object Categorization

- Different levels of recognition
  - Which object class is in the image?
  - Where is it in the image?
  - Where exactly – which pixels?

- ⇒ Obj/Img classification
- ⇒ Detection/Localization
- ⇒ Figure/Ground segmentation

Primary steps

- How to represent a category or object
- How to perform recognition (classification, detection) with that representation
- How to learn models, new categories/objects
Coarse genres of approaches

• Alignment: hypothesize and test
  – Pose clustering with object instances
  – Indexing invariant features + verification

Recall: Alignment

• Alignment: fitting a model to a transformation between pairs of features (matches) in two images

Find transformation $T$ that minimizes

$$\sum_i \text{residual}(T(x_i), x'_i)$$
Alignment-based


Alignment-based

Huttenlocher & Ullman (1987)
Alignment-based

Projective invariants (Rothwell et al., 1992):

ACRONYM (Brooks and Binford, 1981)

Sparser patch matches : for object instances
Coarse genres of approaches

- **Alignment: hypothesize and test**
  - Pose clustering with object instances
  - Indexing invariant features + verification

- **Local features: as parts or words**
  - Part-based models
  - Bags of words models

Local feature-based:

- **bag of words models**
  - Remove spatial information, treat object as a collection of local appearance regions.
Local feature-based: constellation models

- In categorization problem, we no longer have exact correspondences...

- On a local level, we can still detect similar parts.

- Bag-of-words represents objects by their parts

- How can we improve on this?
  - Encode structure

Local feature-based: constellation models

- Fischler & Elschlager 1973

- Model has two components
  - parts (2D image fragments)
  - structure (configuration of parts)
Local feature-based: voting

- For every feature, store possible “occurrences”

- For new image, let the matched features vote for possible object positions

Coarse genres of approaches

- **Alignment: hypothesize and test**
  - Pose clustering with object instances
  - Indexing invariant features + verification

- **Local features: as parts or words**
  - Part-based models
  - Bags of words models

- **Global appearance: “texture templates”**
  - With or without a sliding window
Global appearance-based


Global appearance-based

Eigenfaces (Turk & Pentland, 1991)
Global appearance-based

Scene recognition based on global texture pattern. [Oliva & Torralba (2001)]

Global appearance-based: sliding windows

Given a binary classifier that makes a decision based on global appearance, can slide a window around

Yes, car.
No, not a car.
Global appearance-based: sliding windows

Given a binary classifier that makes a decision based on global appearance, can slide a window around.

Context can constrain a sliding window search

(b) $P(\text{person}) = \text{uniform}$  
(d) $P(\text{person} | \text{geometry})$

(i) $P(\text{person} | \text{viewpoint})$  
(g) $P(\text{person} | \text{viewpoint, geometry})$
Global appearance-based

- Appropriate for classes with more rigid structure, and when good training examples available

- But sensitive to occlusion, clutter, deformations, larger variability within the class.

What “works” today

- Reading license plates, zip codes, checks

Source: Lana Lazebnik
What “works” today

• Reading license plates, zip codes, checks
• Fingerprint recognition

Source: Lana Lazebnik

What “works” today

• Reading license plates, zip codes, checks
• Fingerprint recognition
• Face detection

Source: Lana Lazebnik
What “works” today

• Reading license plates, zip codes, checks
• Fingerprint recognition
• Face detection
• Recognition of flat textured objects (CD covers, book covers, etc.)

Source: Lana Lazebnik

Rough evolution of focus in recognition research

1980s 1990s to early 2000s Currently
Today

- Introduction to object recognition problem
- Recognition by alignment, voting

Recall: Alignment

- Alignment: fitting a model to a transformation between pairs of features (matches) in two images
- We can use this idea to recognize / verify instances of an object.

\[
\text{Find transformation } T \text{ that minimizes } \sum_i \text{residual}(T(x_i), x'_i)
\]
Recall: Alignment

- Alignment: fitting a model to a transformation between pairs of features (matches) in two images
- We can use this idea to recognize / verify instances of an object.

Hypothesize and test: main idea

- Given model of object
- New image: hypothesize object pose
- Render object
- Compare rendering to actual image: if close, good hypothesis.
How to form a hypothesis?

We want a good correspondence between model features and image features.

– Brute force?

Brute force hypothesis generation

• For every possible model, try every possible subset of image points as matches for that model’s points.
• Say we have $L$ objects with $N$ features, $M$ features in image

\[ L \text{ models} \quad \text{image} \]

\[ A \quad B \quad C \]

\[ N \text{ pts} \quad \text{M pts} \]
How to form a hypothesis?

We want a good correspondence between model features and image features.

– **Alignment:**
  • Use subsets of features to estimate larger correspondence
  • Verify

Recall: Fitting an affine transformation

• Assuming we know the correspondences, how do we get the transformation?

\[
\begin{bmatrix}
  x'_i \\
  y'_i
\end{bmatrix} = \begin{bmatrix}
  m_1 & m_2 \\
  m_3 & m_4
\end{bmatrix} \begin{bmatrix}
  x_i \\
  y_i
\end{bmatrix} + \begin{bmatrix}
  t_1 \\
  t_2
\end{bmatrix} = \begin{bmatrix}
  m_1 & m_2 & t_1 \\
  m_3 & m_4 & t_2
\end{bmatrix}
\]
Alignment: fitting

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

- 3+ matches needed to solve for the parameters
- Use local invariant features for reliable matches:
  - interest points relatively sparse
  - very distinctive descriptors

Alignment: backprojection

- 3+ matches needed to solve for the parameters
- Once we have the model parameters, can “backproject”, meaning compute the hypothesized location of any other model points.
Alignment: verification

- 3+ matches needed to solve for the parameters
- Once we have the model parameters, can “backproject”, meaning compute the hypothesized location of any other model points.
- Verification: check for total agreement (e.g., do the image edges coincide with predicted model edges?)
How to form a hypothesis?

We want a good correspondence between model features and image features.

– **Alignment:**
  • Use subsets of features to estimate larger correspondence
  • Verify

*But how to avoid checking all possible sets of correspondences?*
*We’d like to look at the most likely hypotheses first…*

How to form a hypothesis?

We want a good correspondence between model features and image features.

– **Alignment:**
  • Use subsets of features to estimate larger correspondence
  • Verify

– **Voting** (a.k.a. “pose clustering”):
  • Let features *vote* on model parameters
  • Verify those with a lot of support.
Voting: 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).

Voting: Generalized Hough Transform

- A hypothesis generated by a single match may be unreliable,
- So let each match vote for a hypothesis in Hough space
G. Hough Transform details (Lowe’s system)

- **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


Example result

Background subtract for model boundaries
Objects recognized,
Recognition in spite of occlusion

[Lowe]
Recall: difficulties 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.

Example Applications

Mobile tourist guide
- Self-localization
- Object/building recognition
- Photo/video augmentation

[Quack, Leibe, Van Gool, CIVR’08]
Application: Large-Scale Retrieval

Query Results from 5k Flickr images (demo available for 100k set)  
[Philbin CVPR'07]

Web Demo: Movie Poster Recognition

50'000 movie posters indexed

Query-by-image from mobile phone available in Switzerland

Making the Sky Searchable: Fast Geometric Hashing for Automated Astrometry

Sam Roweis, Dustin Lang & Keir Mierle
University of Toronto

David Hogg & Michael Blanton
New York University

Sam Roweis slides and the project overview available here:
http://www.astrometry.net/summary.html
Basic Problem

• I show you a picture of the night sky.

• You tell me where on the sky it came from.

Rules of the game

• We start with a catalogue of stars in the sky, and from it build an index which is used to assist us in locating (‘solving’) new test images.
Rules of the game

• We start with a catalogue of stars in the sky, and from it build an index which is used to assist us in locating (‘solving’) new test images.

• We can spend as much time as we want building the index but solving should be fast.

• Challenges:
  1) The sky is big.
  2) Both catalogues and pictures are noisy.

Distractors and Dropouts

• Bad news:
  Query images may contain some extra stars that are not in your index catalogue, and some catalogue stars may be missing from the image.

• These “distractors” & “dropouts” mean that naïve matching techniques will not work.
You try

Find this “field” on this “sky”.

http://astrometry.net  roweis@cs.toronto.edu

You try

Hint #1: Missing stars.

Find this “field” on this “sky”.

http://astrometry.net  roweis@cs.toronto.edu
You try

Hint #1: Missing stars.
Hint #2: Extra stars.

Find this “field” on this “sky”.

http://astrometry.net roweis@cs.toronto.edu
Robust Matching

• We need to do some sort of robust matching of the test image to any proposed location on the sky.
• Intuitively, we need to ask:  
  “Is there an alignment of the test image and the catalogue so that (almost*) every catalogue star in the field of view of the test image lies (almost*) exactly on top of an observed star?”

Solving the search problem

• Even if we can succeed in finding a good robust matching algorithm, there is still a huge search problem.
• Which proposed location should we match to?
• Exhaustive search? **too expensive!**

The Sky is Big™
(Inverted) Index of Features

- To solve this problem, we will employ the classic idea of an “inverted index”.
- We define a set of “features” for any particular view of the sky (image).
- Then we make an (inverted) index, telling us which views on the sky exhibit certain (combinations of) feature values.
- This is like the question: Which web pages contain the words “machine learning”?

Matching a test image

- When we see a new test image, we compute which features are present, and use our inverted index to look up which possible views from the catalogue also have those feature values.
- Each feature generates a candidate list in this way, and by intersecting the lists we can zero in on the true matching view.
Robust Features for Geometric Hashing

• In our star matching task, the features we chose must be invariant to scale, rotation and translation.

The features we use are the relative positions of nearby quadruples of stars.

Quads as Robust Features

• We encode the relative positions of nearby quadruples of stars (ABCD) using a coordinate system defined by the most widely separated pair (AB).
• Within this coordinate system, the positions of the remaining two stars form a 4-dimensional code for the shape of the quad.
Solving a new test image

- Identify objects (stars+galaxies) in the image bitmap and create a list of their 2D positions.
- Cycle through all possible valid quads (brightest first) and compute their corresponding codes.
- Look up the codes in the code KD-tree to find matches within some tolerance; this stage incurs some false positive and false negative matches.
- Each code match returns a candidate position & rotation on the sky. As soon as 2 quads agree on a candidate, we proceed to verify that candidate against all objects in the image.

http://astrometry.net roweis@cs.toronto.edu

A Real Example from SDSS

Query image (after object detection). An all-sky catalogue.

http://astrometry.net roweis@cs.toronto.edu
A Real Example from SDSS

Query image (after object detection).

Zoomed in by a factor of ~ 1 million.

http://astrometry.net
roweis@cs.toronto.edu

A Real Example from SDSS

Query image (after object detection).

The objects in our index.

http://astrometry.net
roweis@cs.toronto.edu
A Real Example from SDSS

All the quads in our index which are present in the query image.

http://astrometry.net  roweis@cs.toronto.edu

A Real Example from SDSS

A single quad which we happened to try.

http://astrometry.net  roweis@cs.toronto.edu
A Real Example from SDSS

The query image scaled, translated & rotated as specified by the quad.

http://astrometry.net roweis@cs.toronto.edu

A Real Example from SDSS

The proposed match, on which we run verification.

http://astrometry.net roweis@cs.toronto.edu
A Real Example from SDSS

http://astrometry.net  roweis@cs.toronto.edu

The verified answer, overlaid on the original catalogue.

The proposed match, on which we run verification.

Example

http://astrometry.net/gallery.html

A shot of the Great Nebula, by Jerry Lodriguss (c.2006), from astropix.com
Example

An amateur shot of M100, by Filippo Ciferri (c.2007) from flickr.com
http://astrometry.net/gallery.html

Example

A beautiful image of Bode's nebula (c.2007) by Peter Bresseler, from starlightfriend.de
http://astrometry.net/gallery.html
Summary:  
alignment-based recognition  

• Looking for object+pose that fits well with image.  
  – Use good correspondences (i.e., based on local invariant feature matches) to designate hypotheses.  
  – Can limit number of verifications performed by voting for most likely model parameters.  

• Pros:  
  – Effective when we are able to find reliable features within clutter  
  – Great results for matching specific instances  

• Cons:  
  – May not scale well with the number of models  
  – Not as suited for category-level recognition  

Coming up  

• Pset 4 is out this Thursday 11/5, due 11/24