Classical Computer Vision: Feature Engineering

Connelly Barnes

With slides from Alexei Efros, James Hays, Antonio Torralba, Jean-Francois Lalonde, Frederic Heger, Steve Seitz, Rick Szeliski, Fredo Durand, Kristin Grauman
Big Problems in Computer Vision

- Find correspondences between the same object in 2 photos

Photo from: [Vedaldi and Zisserman](http://vedaldi.org/brain/)
Big Problems in Computer Vision

• What category is this image? (classification/recognition)

Better than human performance reported on ImageNet large-scale challenge.
Big Problems in Computer Vision

• Are two photos the same?
Classical Approach

• Manually engineer features to detect and describe different regions of the image.

• A feature is just a vector in $\mathbb{R}^n$.

• It could represent the entire image...

• …Or just a local region

• To find similar features, use a distance metric such as Euclidean distance.
How big is Flickr?

100M photos updated \textit{daily}

6B photos as of August 2011!

• ~3B public photos

Credit: Franck_Michel (http://www.flickr.com/photos/franckmichel/)
How Annotated is Flickr? (tag search)

Party – 23,416,126
Paris – 11,163,625
Pittsburgh – 1,152,829
Chair – 1,893,203
Violin – 233,661
Trashcan – 31,200
“Trashcan” Results

http://www.flickr.com/search/?q=trashcan+NOT+party&m=tags&z=t&page=5
Big Issues

If we could harness all this data, we could use it.

What is out there on the Internet?
How do we get it?
What can we do with it?

• Let’s see a motivating example...
Scene Completion
Scene Matching
Scene Descriptor
Scene Descriptor

Scene Gist Descriptor
( Oliva and Torralba 2001 )
Scene Descriptor

Scene Gist Descriptor

(Oliva and Torralba 2001)
2 Million Flickr Images
Context Matching
Graph cut + Poisson blending
Nearest neighbors from a collection of 20 thousand images
Nearest neighbors from a collection of 2 million images
"Unreasonable Effectiveness of Data"

Parts of our world can be explained by elegant mathematics

• physics, chemistry, astronomy, etc.

But much cannot

• psychology, economics, genetics, etc.

Enter The Data!

• Great advances in several fields:
  – e.g. speech recognition, machine translation
  – Case study: Google

[Halevy, Norvig, Pereira 2009]
A.I. for the postmodern world:

- all questions have already been answered…many times, in many ways
- Google is dumb, the “intelligence” is in the data
How about *visual* data?

Text is simple:

- clean, segmented, compact, 1D

Visual data is much harder:

- Noisy, unsegmented, high entropy, 2D/3D
Distance Metrics

- Euclidian distance of 5 units

- Gray value distance of 50 values

- ?
SSD says these are not similar
Image Descriptors

- Blur + SSD
- Gist descriptor (average edge response in a coarse spatial grid)
- Color histograms
- Filter response histograms
- Invariant detectors and descriptors (SIFT)
- Convolutional neural networks (CNNs) – later classes
Image Descriptors

- Blur + SSD
- Gist descriptor (average edge response in a coarse spatial grid)
- **Color histograms**
- Filter response histograms
- Invariant detectors and descriptors (SIFT)
- Convolutional neural networks (CNNs) – later classes
Image Representations: Histograms

global histogram
Represent distribution of features
Color, texture, depth, …
Image Representations: Histograms

- **Joint histogram**
  - Requires lots of data
  - Loss of resolution to avoid empty bins

- **Marginal histogram**
  - Requires independent features
  - More data/bin than joint histogram

Images from Dave Kauchak
Image Representations: Histograms

Adaptive binning

- Better data/bin distribution, fewer empty bins
- Can adapt available resolution to relative feature importance
Image Representations: Histograms

Clusters / Signatures

- “super-adaptive” binning
- Does not require discretization along any fixed axis
Issue: How to Compare Histograms?

Bin-by-bin comparison
Sensitive to bin size.
Could use wider bins ...
... but at a loss of resolution

Cross-bin comparison
How much cross-bin influence is necessary/sufficient?
Red Car Retrievals (Color histograms)

\[
\chi^2(h_i, h_j) = \frac{1}{2} \sum_{m=1}^{K} \frac{[h_i(m) - h_j(m)]^2}{h_i(m) + h_j(m)}
\]

Histogram matching distance
Image Descriptors

- Blur + SSD
- Gist descriptor (average edge response in a coarse spatial grid)
- Color histograms
- **Filter response histograms**
- Invariant detectors and descriptors (SIFT)
- Convolutional neural networks (CNNs) – later classes
Capturing the “essence” of texture

...for real images

We don’t want an actual texture realization, we want a texture invariant.

What are the tools for capturing statistical properties of some signal?
But first...

How to filter an image?
Convolution takes a windowed average of an image $F$ with a filter $H$, where the filter is flipped horizontally and vertically before being applied:

$$G[i, j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u, v] F[i - u, j - v]$$

$$G = H \ast F$$
Convolution is nice!

• Notation: \( b = c \ast a \)

• Convolution is a multiplication-like operation
  – commutative \( a \ast b = b \ast a \)
  – associative \( a \ast (b \ast c) = (a \ast b) \ast c \)
  – distributes over addition \( a \ast (b + c) = a \ast b + a \ast c \)
  – scalars factor out \( \alpha a \ast b = a \ast \alpha b = \alpha(a \ast b) \)
  – identity: unit impulse \( e = [..., 0, 0, 1, 0, 0, ...] \)
    \( a \ast e = a \)

• Conceptually no distinction between filter and signal

• Usefulness of associativity
  – often apply several filters one after another: \(((a \ast b_1) \ast b_2) \ast b_3)\)
  – this is equivalent to applying one filter: \(a \ast (b_1 \ast b_2 \ast b_3)\)
Practice with linear filters

Original

0 0 0 0
0 1 0 0
0 0 0

Source: D. Lowe
Practice with linear filters

Original

Filtered
(no change)

Source: D. Lowe
Practice with linear filters

Original

Source: D. Lowe
Practice with linear filters

Original

Shifted left
By 1 pixel

Source: D. Lowe
Other filters

Separable (show on board)
Other filters

Sobel

Vertical Edge (absolute value)
Other filters

Sobel

Separable (show on board)
Other filters

Sobel

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>-2</td>
<td>-1</td>
</tr>
</tbody>
</table>

Separable (show on board)

Horizontal Edge (absolute value)
How to use filters to describe texture?
Representing textures

Textures are made up of quite stylised subelements, repeated in meaningful ways

**Representation:**
- find the subelements, and represent their statistics

But what are the subelements, and how do we find them?
- find subelements by applying filters, looking at the magnitude of the response

**What filters?**
- experience suggests spots and oriented bars at a variety of different scales

**What statistics?**
- within reason, the more the merrier.
- At least, mean and standard deviation
- better, various conditional histograms.
Gabor Filter

- Rotated Gaussian filter times cosine wave.

Real

\[ g(x, y; \lambda, \theta, \psi, \sigma, \gamma) = \exp \left( - \frac{x'^2 + \gamma^2 y'^2}{2\sigma^2} \right) \cos \left( 2\pi \frac{x'}{\lambda} + \psi \right) \]
Multi-scale filter decomposition

Gabor Filter bank

Input image
Filter response histograms
Threshold squared, blurred responses, then categorize texture based on those two bits.
Start with a noise image as output

Main loop:

- Match *pixel* histogram of output image to input
- Decompose input and output images using multi-scale filter bank (Steerable Pyramid)
- Match sub-band histograms of input and output pyramids
- Reconstruct input and output images (collapse the pyramids)
Image Descriptors

- Blur + SSD
- Gist descriptor (average edge response in a coarse spatial grid)
- Color histograms
- Filter response histograms
- Invariant detectors and descriptors (SIFT)
- Convolutional neural networks (CNNs) – later classes
Feature Detection
Feature Matching

How do we match the features between the images?

• Need a way to describe a region around each feature
  – e.g. image patch around each feature

• Use successful matches to estimate models of objects/scene
  – Need to do something to get rid of outliers

Issues:

• What if the image patches for several interest points look similar?
  – Make patch size bigger

• What if the image patches for the same feature look different due to scale, rotation, exposure etc.
  – Need an invariant descriptor
Invariant Feature Descriptors

Applications

Feature points are used for:

• Image alignment (homography, fundamental matrix)
• 3D reconstruction
• Motion tracking
• Object recognition
• Scene categorization
• Indexing and database retrieval
• Robot navigation
• … other
Feature Detectors and Descriptors

• Feature detector
  • scale invariant Harris corners
• Feature descriptor
  • patches, oriented patches

Reading:
David Lowe 2004,
Distinctive Image Features from Scale-Invariant Keypoints
Harris corner detector

The Basic Idea

We should easily recognize the point by looking through a small window. Shifting a window in *any direction* should give a *large change* in intensity.
Harris Detector: Basic Idea

“flat” region:
no change in all directions

“edge”:
no change along the edge direction

“corner”:
significant change in all directions
Change of intensity for the shift $[u,v]$:

$$E(u,v) = \sum_{x,y} w(x, y) [I(x+u, y+v) - I(x, y)]^2$$

Window function $w(x,y)$ =
- 1 in window, 0 outside
- Gaussian
Harris Detector: Mathematics

For small shifts \([u, v]\) we have a \textit{bilinear} approximation:

\[
E(u, v) \approx \begin{bmatrix} u \\ v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix}
\]

where \(M\) is a \(2 \times 2\) matrix computed from image derivatives:

\[
M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}
\]

\[
A^TA = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x I_y] = \sum \nabla I(\nabla I)^T
\]
Classification of image points using eigenvalues of $M$:

- **Corner**
  - $\lambda_1$ and $\lambda_2$ are large,
  - $\lambda_1 \sim \lambda_2$;
  - $E$ increases in all directions

- **Edge**
  - $\lambda_2 \gg \lambda_1$

- **Flat** region
  - $\lambda_1$ and $\lambda_2$ are small;
  - $E$ is almost constant in all directions

But eigenvalues are expensive to compute.
Harris Detector: Mathematics

Measure of corner response:

\[ R = \frac{\det M}{\text{Trace } M} \]

\[ \det M = \lambda_1 \lambda_2 \]
\[ \text{trace } M = \lambda_1 + \lambda_2 \]

Algorithm: collect local maxima of R (above a threshold).
DoG Feature Detector ("Blob detection")
Idea: Find blob regions, scale invariant

Approach:
Run linear filter (Difference of Gaussians)
At different resolutions of image

Often used for computing SIFT.
"SIFT" = DoG detector + SIFT descriptor
Difference of Gaussians

Minus

Equals
Key point localization

Detect maxima and minima of difference-of-Gaussian in scale space
Example of keypoint detection

(a) 233x189 image
(b) 832 DOG extrema
Feature descriptors

We know how to detect points
Next question: **How to match them?**

Point descriptor should be:
1. Invariant
2. Distinctive
Descriptors Invariant to Rotation

Find local orientation

Dominant direction of gradient

- Extract image patches relative to this orientation
Descriptor Vector

Orientation = dominant gradient direction

Rotation Invariant Frame

- Scale-space position \((x, y, s)\) + orientation \((\theta)\)
SIFT vector formation

Thresholded image gradients are sampled over 16x16 array of locations in scale space
Create array of orientation histograms
8 orientations x 4x4 histogram array = 128
SIFT local feature descriptor

Based on 16*16 patches
4*4 subregions
8 bins in each subregion
4*4*8=128 dimensions in total
SIFT vs CNNs

SIFT descriptor is outperformed by CNN features.

[Discriminative Unsupervised Feature Learning... 2015]
Feature matching

Use k-nearest neighbors
What about outliers?
Feature-space outlier rejection

- 1-NN: SSD of the closest match
- 2-NN: SSD of the second-closest match
- Look at how much better 1-NN is than 2-NN, e.g. 1-NN/2-NN
- That is, is our best match so much better than the rest?
Image Descriptors

- Blur + SSD
- Gist descriptor (average edge response in a coarse spatial grid)
- Color histograms
- Filter response histograms
- Invariant detectors and descriptors (SIFT)
- Convolutional neural networks (CNNs) – later classes