**Recognition III**

Introduction to Computer Vision
CSE 152
Lecture 19

**Announcements**

- HW 4 has been posted. Due 06/08
- Lecture plan – Wrap up Recognition and on tomotion.
- Final exam: June 12, 8AM-11AM, cumulative

---

**Linear Subspaces & Linear Projection**

- An \( n \)-pixel image \( x \in \mathbb{R}^d \) can be projected to a low-dimensional feature space \( y \in \mathbb{R}^k \) by
  \[
  y = Wx
  \]
  where \( W \) is an \( k \) by \( d \) matrix.

- Recognition is performed in \( \mathbb{R}^k \) using, for example, nearest neighbor.

- How do we choose a good \( W \)?

**Distance to Linear Subspace**

- An \( n \)-pixel image \( x \in \mathbb{R}^d \) can be projected to a low-dimensional feature space \( y \in \mathbb{R}^k \) by
  \[
  y = Wx
  \]

- From \( y \in \mathbb{R}^k \), the reconstruction of the point in \( \mathbb{R}^d \) is \( W^T y = W^T Wx \).

- The error of the reconstruction, or the distance from \( x \) to the subspace spanned by \( W \) is:
  \[
  ||x - W^T Wx||
  \]

**Distance to Affine Subspace**

(i.e., Distance to Face Space)

- Represented by mean vector \( \mu \) and basis images \( W \).

- An \( n \)-pixel image \( x \in \mathbb{R}^d \) can be projected to a low-dimensional feature space \( y \in \mathbb{R}^k \) by
  \[
  y = W(x - \mu)
  \]

- From \( y \in \mathbb{R}^k \), the reconstruction of the point in \( \mathbb{R}^d \) is
  \[
  W^T y + \mu = W^T W(x - \mu) + \mu
  \]

- The error of the reconstruction, or the distance from \( x \) to the affine is:
  \[
  ||x - W^T W(x - \mu) - \mu|| = ||(I - W^T W)(x - \mu)||
  \]

**Application 1:**

Face detection using “distance to face space”

- Scan a window \( \omega \) across the image, and classify the window as face/not face as follows:
  - Project window to subspace, and reconstruct as described earlier.
  - Compute distance between \( \omega \) and reconstruction.

- Local minima of distance over all image locations less than some threshold are taken as locations of faces.

- Repeat at different scales.

- Possibly normalize windows intensity so that \( ||\omega|| = 1 \).
An important footnote:
We don’t really implement PCA by constructing a covariance matrix!

Why?
1. How big is $\Sigma$?
   - $n \times n$ where $n$ is the number of pixels in an image!!
2. You only need the first $k$ Eigenvectors

Assume we have a set of $n$ feature vectors $x_i (i=1, ..., n)$ in $\mathbb{R}^d$. Write

$$\mathbf{X} = \frac{1}{n} \sum_{i=1}^{n} (x_i - \mu)(x_i - \mu)^T$$

The unit eigenvectors of $\mathbf{X}$, which we write as $v_1, v_2, ..., v_n$, where the order is given by the size of the eigenvalue and $v_1$ has the largest eigenvalue $= \sigma_1$ gives us a set of features with the following representation:

Singular Value Decomposition

- Any $m \times n$ matrix $A$ may be factored such that
  $$A = UV^T$$
  $[m \times n] = [m \times m][m \times n][n \times n]$
- $U$: $m \times m$, orthogonal matrix
  - Columns of $U$ are the eigenvectors of $AA^T$
- $V$: $n \times n$, orthogonal matrix
  - Columns are the eigenvectors of $A^TA$
- $\Sigma$: $m \times n$, diagonal with non-negative entries ($\sigma_1, \sigma_2, ..., \sigma_s$) with $s=\min(m,n)$ are called the singular values.

Important property
- Singular values are the square roots of Eigenvalues of both $AA^T$ and $A^TA$ & Columns of $U$ are corresponding Eigenvectors!!

Performing PCA with SVD

- Singular values of $A$ are the square roots of eigenvalues of both $AA^T$ and $A^TA$
- Columns of $U$ are corresponding Eigenvectors
- Covariance matrix is:
  $$\Sigma = \frac{1}{n} \sum_{i=1}^{n} (x_i - \mu)(x_i - \mu)^T$$
- So, ignoring $1/n$ subtract mean image $\mu$ from each input image, create a $d \times n$ data matrix, and perform thin SVD on the data matrix. $D=[x_1-\mu \mid x_2-\mu \mid ... \mid x_n-\mu]$

PCA & Fisher’s Linear Discriminant

- Between-class scatter
  $$S_b = \sum_{i=1}^{c} k_i (\mu_i - \mu)(\mu_i - \mu)^T$$
- Within-class scatter
  $$S_w = \sum_{i=1}^{c} \sum_{x_i \in \chi_i} (x_i - \mu_i)(x_i - \mu_i)^T$$
- Total scatter
  $$S = S_w + S_b$$
- Where
  - $c$ is the number of classes
  - $\mu_i$ is the mean of class $\chi_i$
  - $|\chi_i|$ is number of samples of $\chi_i$.

If the data points $x_i$ are projected by $y_i=Wx_i$ and the scatter of $x_i$ is $S$, the scatter of the projected points $\chi_i$ is $W^TSW$.

Computing the Fisher Projection Matrix

The $w_i$ are orthonormal
- There are at most $c-1$ non-zero generalized Eigenvalues, so $m \leq c-1$
- Can be computed with eig in Matlab
Fisherfaces

\[ W = W_f W_{PC} \]
\[ W_{PC} = \arg \max_{W} \left| W^T S_r W \right| \]
\[ W_f = \arg \max_{W} \left| W^T W_{PC}^2 W_{PC} W \right| \]

- Since \( S_r \) is rank \( N-c \), project training set to subspace spanned by first \( N-c \) principal components of the training set.
- Apply FLD to \( N-c \) dimensional subspace yielding \( c-1 \) dimensional feature space.

- Fisher’s Linear Discriminant projects away the within-class variation (lighting, expressions) found in training set.
- Fisher’s Linear Discriminant preserves the separability of the classes.

Harvard Face Database

- 10 individuals
- 66 images per person
- Train on 6 images at 15°
- Test on remaining images

Recognition Results: Lighting Extrapolation

Variability:
- Camera position
- Illumination
- Internal parameters
- Within-class variations

Appearance manifold approach
(Nayar et al. '96)

- for every object
  1. sample the set of viewing conditions
  2. Crop & scale images to standard size
  3. Use as feature vector
- apply a PCA over all the images
- keep the dominant PCs
- Set of views for one object is represented as a manifold in the projected space
- Recognition: What is nearest manifold for a given test image?

Parameterized Eigenspace
Recognition in Cluttered Scenes
Interest Points + Feature Descriptors + Relations

Matching using Local Image features
Simple approach
- Detect corners in image (e.g. Harris corner detector).
- Represent neighborhood of corner by a feature vector produced by Gabor Filters, K-jets, SIFT features, etc.
- Modeling: Given an training image of an object w/o clutter, detect corners, compute feature descriptors, store these.
- Recognition time: Given test image with possible clutter, detect corners and compute features. Find models with same feature descriptors (hashing) and vote.

Employ spatial relations

Figure from “Local grayvalue invariants for image retrieval,” by C. Schmid and R. Mohr, IEEE Trans. Pattern Analysis and Machine Intelligence, 1997 copyright 1997, IEEE.
Even without shading, shape reveals a lot

Motion

Some problems of motion
1. Correspondence: Where have elements of the image moved between image frames
2. Reconstruction: Given correspondence, what is 3-D geometry of scene
3. Ego Motion: How has the camera moved.
4. Segmentation: What are regions of image corresponding to different moving objects
5. Tracking: Where have objects moved in the image? related to correspondence and segmentation.

Variations:
- Small motion (video),
- Wide-baseline (multi-view)

Structure-from-Motion (SFM)
Goal: Take as input two or more images or video w/o any information on camera position/motion, and estimate camera position and 3-D structure of scene.

Two Approaches
1. Discrete motion (wide baseline)
   1. Orthographic (affine) vs. Perspective
   2. Two view vs. Multi-view
   3. Calibrated vs. Uncalibrated
2. Continuous (Infinitesimal) motion

Discrete Motion: Some Counting
Consider $M$ images of $N$ points, how many unknowns?
1. Camera locations: Affix coordinate system to location of first camera location: $(M-1)*6$ Unknowns
2. 3-D Structure: $3*N$ Unknowns
3. Can only recover structure and motion up to scale. Why?

Total number of unknowns: $(M-1)*6+3*N-1$
Total number of measurements: $2*M*N$
Solution is possible when $(M-1)*6+3*N-1 \leq 2*M*N$

$M=2 \Rightarrow N \geq 5$
$M=3 \Rightarrow N \geq 4$
Sketch of Two View SFM Algorithm

1. Detect feature points
2. Find 8 matching feature points (easier said than done, but usually done with RANSAC Algorithm)
3. Compute the Essential Matrix $E$ using Normalized 8-point Algorithm
4. Compute $R$ and $T$ (recall that $E = RS$ where $S$ is skew symmetric matrix)
5. Perform stereo matching using recovered epipolar geometry expressed via $E$.
6. Reconstruct 3-D geometry of corresponding points.

Feature matching

Evaluate normalized cross correlation (or sum of squared differences) for all features with similar coordinates e.g. $(x', y') = \begin{bmatrix} x' & x'' \\ y' & y'' \end{bmatrix}$

Keep mutual best matches
Still many wrong matches!

Comments

- Greedy Algorithm:
  - Given feature in one image, find best match in second image irrespective of other matches.
  - OK for small motions, little rotation, small search window
- Otherwise
  - Must compare descriptor over rotation
  - Can’t consider $O(n^2)$ potential pairings (way too many), so
    * Manual correspondence (e.g., façade, photogrammetry).
    * Use random sampling (RANSAC)
    * More descriptive features (line segments, SIFT, larger regions, color).
  - Use video sequence to track, but perform SFM w/ first and last image.