Object Recognition

Computer Vision I

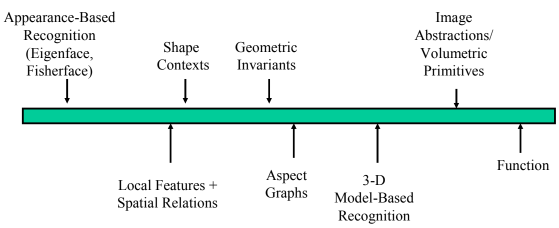
CSE252A

Lecture 20

Announcements

- HW 4 Due Date Extended to Friday. Leave report in my mailbox or under my door in AP&M.
- Today:
  1. Appearance-based vision
  2. Final Exam
  3 Model-Based Vision

A Rough Recognition Spectrum



A Rough Recognition Spectrum

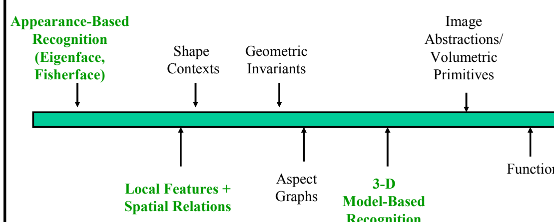
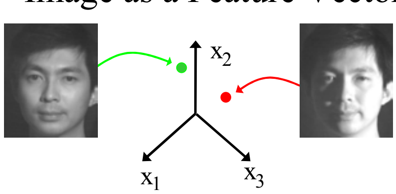


Image as a Feature Vector



- Consider an n-pixel image to be a point in an n-dimensional space, $\mathbf{x} \in \mathbf{R}^n$.
- Each pixel value is a coordinate of $\mathbf{x}$.

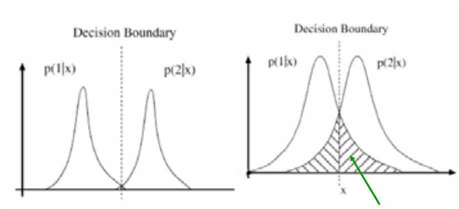
Bayes Decision Theory

Class: ω
Feature vector: $\mathbf{x}$

Prior probability: $p(\omega_1)$, $p(\omega_2)$
Class conditional density $p(\mathbf{x} \mid \omega_1)$, $p(\mathbf{x} \mid \omega_2)$,

Posterior: $p(\omega_1 \mid \mathbf{x}) = p(\mathbf{x} \mid \omega_1)\, p(\omega 1)/p(\mathbf{x})$,

Decide ω_1 when $p(\omega_1 \mid \mathbf{x}) > p(\omega_2 \mid \mathbf{x})$



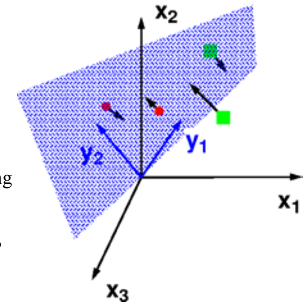
Eigenfaces: linear projection

- An n-pixel image $\mathbf{x} \in \mathbf{R}^n$ can be projected to a low-dimensional feature space $\mathbf{y} \in \mathbf{R}^m$ by

$$\mathbf{y} = W\mathbf{x}$$

where W is an n by m matrix.

- Recognition is performed using nearest neighbor in $\mathbf{R}^m$.
- How do we choose a good W?



Eigenfaces: Principal Component Analysis (PCA)

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

$$\boldsymbol{\mu} = \frac{1}{n}\sum_i \boldsymbol{x}_i$$

$$\Sigma = \frac{1}{n-1}\sum_i (\boldsymbol{x}_i - \boldsymbol{\mu})(\boldsymbol{x}_i - \boldsymbol{\mu})^T$$

The unit eigenvectors of Σ — which we write as $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_d$, where the order is given by the size of the eigenvalue and $\boldsymbol{v}_1$ has the largest eigenvalue — give a set of features with the following properties:

- They are independent.
- Projection onto the basis $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k\}$ gives the k-dimensional set of linear features that preserves the most variance.

Algorithm 22.5: *Principal components analysis identifies a collection of linear features that are independent, and capture as much variance as possible from a dataset.*

Some details: Use Singular value decomposition, "trick" described in text to compute basis when n<<d

Singular Value Decomposition

[Repeated from Lecture 7]

- Any m by n matrix $\boldsymbol{A}$ may be factored such that

$$\boldsymbol{A} = \boldsymbol{U}\Sigma\boldsymbol{V}^T$$

$$[m \times n] = [m \times m][m \times n][n \times n]$$

- $\boldsymbol{U}$: m by m, orthogonal matrix
  - Columns of $\mathbf{U}$ are the eigenvectors of $\boldsymbol{AA}^T$
- $\boldsymbol{V}$: n by n, orthogonal matrix,
  - columns are the eigenvectors of $\boldsymbol{A}^T\boldsymbol{A}$
- **Σ:** m by n, diagonal with non-negative entries $(\sigma_1, \sigma_2, \ldots, \sigma_s)$ with s=min(m,n) are called the called the singular values
  - **Singular values are the square roots of eigenvalues of both $\boldsymbol{AA}^T$ and $\boldsymbol{A}^T\boldsymbol{A}$ & *Columns of U are corresponding Eigenvectors***
  - *Result of SVD algorithm:* $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_s$

SVD Properties

- In Matlab [u s v] = svd(A), and you can verify that: A=u*s*v'
- r=Rank($\boldsymbol{A}$) = # of non-zero singular values.
- $\boldsymbol{U}$, $\boldsymbol{V}$ give us orthonormal bases for the subspaces of $\boldsymbol{A}$:
  - 1st r columns of $\boldsymbol{U}$: Column space of $\boldsymbol{A}$
  - Last $m - r$ columns of $\boldsymbol{U}$: Left nullspace of $\boldsymbol{A}$
  - 1st r columns of $\boldsymbol{V}$: Row space of $\boldsymbol{A}$
  - 1st $n - r$ columns of $\boldsymbol{V}$: Nullspace of $\boldsymbol{A}$
- *For d<= r, the first d column of U provide the best d-dimensional basis for columns of A in least squares sense.*

Thin SVD

- Any m by n matrix $\boldsymbol{A}$ may be factored such that

$$\boldsymbol{A} = \boldsymbol{U}\Sigma\boldsymbol{V}^T$$

$$[m \times n] = [m \times m][m \times n][n \times n]$$

- If m>n, then one can view Σ as:

$$\begin{bmatrix} \Sigma' \\ 0 \end{bmatrix}$$

- Where $\Sigma'=\mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_s)$ with s=min(m,n), and lower matrix is (n-m by m) of zeros.
- Alternatively, you can write:

$$\boldsymbol{A} = \boldsymbol{U}'\Sigma'\boldsymbol{V}^T$$

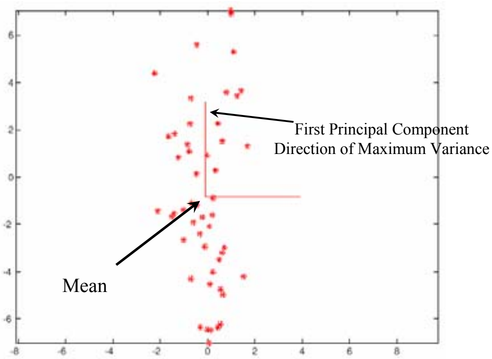
- In Matlab, thin SVD is:[U S V] = svds(A)

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
- And $\sum_{i=1}^{n} a_i a_i^T = [a_1 \quad a_2 \quad \cdots \quad a_n][a_1 \quad a_2 \quad \cdots \quad a_n]^T = AA^T$
- Covariance matrix is:

$$\Sigma = \frac{1}{n}\sum_{i=1}^{n}(\vec{x}_i - \vec{\mu})(\vec{x}_i - \vec{\mu})^T$$

- So, ignoring 1/n subtract mean image μ from each input image, create data matrix, and perform thin SVD on the data matrix.



Eigenfaces

- Modeling
  1. Given a collection of n labeled training images,
  2. Compute mean image and covariance matrix.
  3. Compute k Eigenvectors (note that these are images) of covariance matrix corresponding to k largest Eigenvalues. (Or perform using SVD!!)
  4. Project the training images to the k-dimensional Eigenspace.
- Recognition
  1. Given a test image, project to Eigenspace.
  2. Perform classification to the projected training images.

Eigenfaces: Training Images

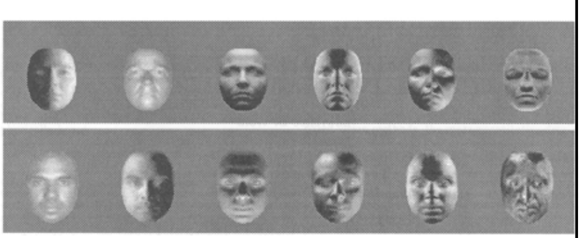
[Turk, Pentland 01

Eigenfaces

Mean Image

Basis Images

Eigenfaces for a single individual, observed under variable lighting

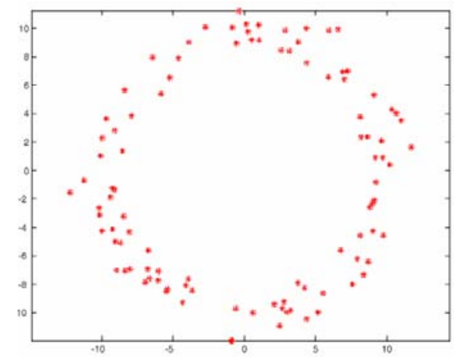


Underlying assumptions

- Background is not cluttered (or else only looking at interior of object
- Lighting in test image is similar to that in training image.
- No occlusion
- Size of training image (window) same as window in test image.

Difficulties with PCA

- Projection may suppress important detail
  - smallest variance directions may not be unimportant
- Method does not take discriminative task into account
  - typically, we wish to compute features that allow good discrimination
  - not the same as largest variance or minimizing reconstruction error.



Fisherfaces: Class specific linear projection

P. Belhumeur, J. Hespanha, D. Kriegman, *Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection*, PAMI, July 1997, pp. 711--720.

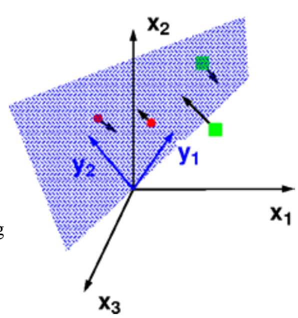
- An n-pixel image $\mathbf{x} \in \mathbf{R}^n$ can be projected to a low-dimensional feature space $\mathbf{y} \in \mathbf{R}^m$ by

$$\mathbf{y} = W\mathbf{x}$$

where W is an n by m matrix.

- Recognition is performed using nearest neighbor in $\mathbf{R}^m$.

- How do we choose a good W?



PCA & Fisher's Linear Discriminant

- Between-class scatter

$$S_B = \sum_{i=1}^{c} |\chi_i| (\mu_i - \mu)(\mu_i - \mu)^T$$

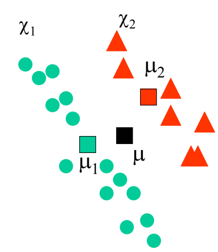
- Within-class scatter

$$S_W = \sum_{i=1}^{c} \sum_{x_k \in \chi_i} (x_k - \mu_i)(x_k - \mu_i)^T$$

- Total scatter

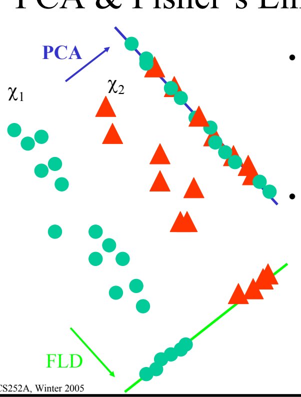
$$S_T = \sum_{i=1}^{c} \sum_{x_k \in \chi_i} (x_k - \mu)(x_k - \mu)^T = S_B + S_W$$

- Where
  - c is the number of classes
  - μ_i is the mean of class χ_i
  - $|\chi_i|$ is number of samples of χ_i.



- If the data points are projected by y=Wx and scatter of points is S, then the scatter of the projected points is $W^T S W$

PCA & Fisher's Linear Discriminant



- PCA (Eigenfaces)

$$W_{PCA} = \arg\max_W \left| W^T S_T W \right|$$

Maximizes projected total scatter

- Fisher's Linear Discriminant

$$W_{fld} = \arg\max_W \frac{\left| W^T S_B W \right|}{\left| W^T S_W W \right|}$$

Maximizes ratio of projected between-class to projected within-class scatter

Computing the Fisher Projection Matrix

$$W_{opt} = \arg\max_W \frac{\left| W^T S_B W \right|}{\left| W^T S_W W \right|} = \left[\mathbf{w}_1 \ \mathbf{w}_2 \ \ldots \ \mathbf{w}_m \right] \qquad (4)$$

where $\{\mathbf{w}_i \mid i = 1, 2, \ldots, m\}$ is the set of generalized eigenvectors of S_B and S_W corresponding to the m largest generalized eigenvalues $\{\lambda_i \mid i = 1, 2, \ldots, m\}$, i.e.,

$$S_B \mathbf{w}_i = \lambda_i S_W \mathbf{w}_i, \qquad i = 1, 2, \ldots, m$$

- The w_i are orthonormal
- There are at most c-1 non-zero generalized Eigenvalues, so m <= c-1
- Can be computed with *eig* in Matlab

Fisherfaces

$$W = W_{fld} W_{PCA}$$

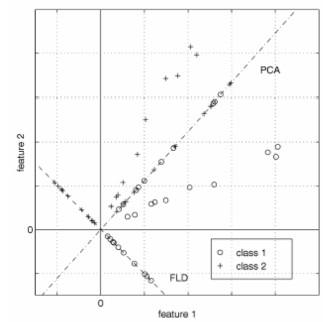
$$W_{PCA} = \arg\max_W \left| W^T S_T W \right|$$

$$W_{fld} = \arg\max_W \frac{\left| W^T W_{PCA}^T S_B W_{PCA} W \right|}{\left| W^T W_{PCA}^T S_W W_{PCA} W \right|}$$

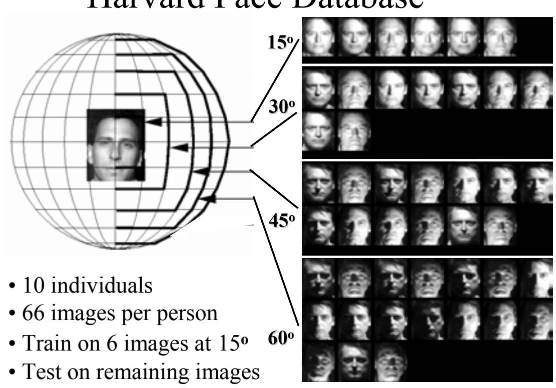
- Since S_W is rank N-c, project training set to subspace spanned by first *N-c* principal components of the training set.
- Apply FLD to *N-c* dimensional subspace yielding *c-1* dimensional feature space.

- 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.

PCA vs. FLD

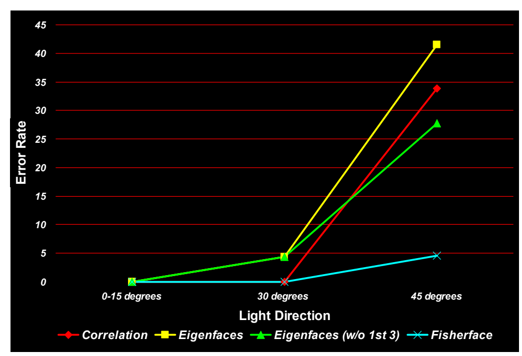


Harvard Face Database



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

Recognition Results: Lighting Extrapolation



Support Vector Machines

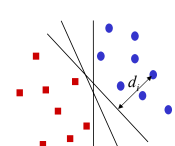
- Bayes classifiers and generative approaches in general try to model of the posterior, p(ω|**x**)
- Instead, try to obtain the decision boundary directly
  - potentially easier, because we need to encode only the geometry of the boundary, not any irrelevant wiggles in the posterior.
  - Not all points affect the decision boundary

Support Vector Machines

- Set S of points $x_i \in R^n$, each x_i belongs to one of two classes $y_i \in \{-1,1\}$
- The goals is to find a hyperplane that divides S in these two classes

S is separable if $\exists w \in R^n, b \in R$

$$y_i(\mathrm{w}.\mathrm{x}_i + b) \geq 1$$



Separating hyperplanes

$$\mathrm{w}.\mathrm{x} + b = 0$$

$$d_i = \frac{\mathrm{w}.\mathrm{x}_i + b}{w} \qquad w = \|\mathrm{w}\|$$

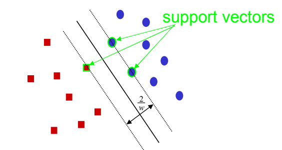
Closest point

$$y_i d_i = \frac{1}{w}$$

Support Vector Machines

- Optimal separating hyperplane maximizes $\frac{1}{w}$

Problem 1:
Minimize $\frac{1}{2}\mathrm{w}.\mathrm{w}$
Subject to $y_i(\mathrm{w}.\mathrm{x}_i + b) \geq 1, \; i = 1,2,\ldots,N$



Optimal separating hyperplane (OSH)

Solve using Lagrange multipliers

- Lagrangian

$$L(\mathrm{w}, b, \alpha) = \tfrac{1}{2}\mathrm{w}.\mathrm{w} - \sum_{i=1}^{N} \alpha_i \{ y_i(\mathrm{w}.\mathrm{x}_i + b) - 1 \}$$

– at solution

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{N} y_i \alpha_i = 0$$

$$\frac{\partial L}{\partial \mathrm{w}} = \mathrm{w} - \sum_{i=1}^{N} \alpha_i y_i \mathrm{x}_i = 0$$

– therefore

$$L(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j \mathrm{x}_i . \mathrm{x}_j$$

$$\boldsymbol{\alpha} \geq 0$$

Decision function

- Once w and b have been computed the classification decision for input x is given by

$$f(\mathrm{x}) = \mathrm{sign}(\mathrm{w}.\mathrm{x} + b)$$

- Note that the globally optimal solution can always be obtained (convex problem)

Non-linear SVMs

- Non-linear separation surfaces can be obtained by non-linearly mapping the data to a high dimensional space and then applying the linear SVM technique
- Note that data only appears through vector product
- Need for vector product in high-dimension can be avoided by using Mercer kernels:

$$K(\mathrm{x}_i, \mathrm{x}_i) = \Phi(\mathrm{x}_i)\Phi(\mathrm{x}_i)$$

e.g. $K(\mathrm{x}, \mathrm{y}) = (\mathrm{x}.\mathrm{y})^p$ (Polynomial kernel)

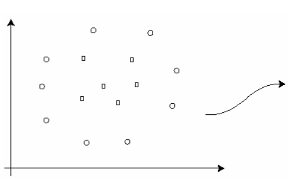
$K(\mathrm{x}, \mathrm{y}) = (\mathrm{x}.\mathrm{y})^2 = x_1^2 y_1^2 + x_1 x_2 y_1 y_2 + x_2^2 y_2^2$

$\Phi(\mathrm{x}) = (x_1^2, x_1 x_2, x_2^2)$

$K(\mathrm{x}, \mathrm{y}) = \exp\left(\frac{-\|\mathrm{x} - \mathrm{y}\|^2}{\sigma^2} \right)$ (Radial Basis Function)

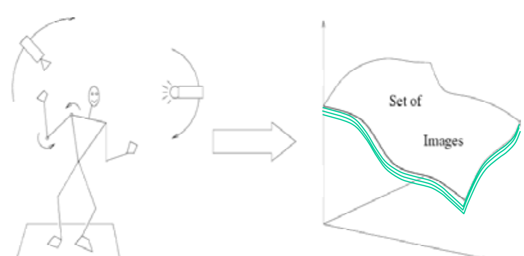
$K(\mathrm{x}, \mathrm{y}) = \tanh(\kappa\, \mathrm{x}.\mathrm{y} - \delta)$ (Sigmoïdal function)

$$(x, y) \rightarrow (x^2, xy, y^2, x, y) = (u_0, u_1, u_2, u_3, u_4)$$



Space in which decision boundary is linear - a conic in the original space has the form

$$au_0 + bu_1 + cu_2 + du_3 + eu_4 + f = 0$$



Variability: Camera position
Illumination
Internal parameters

Within-class variations

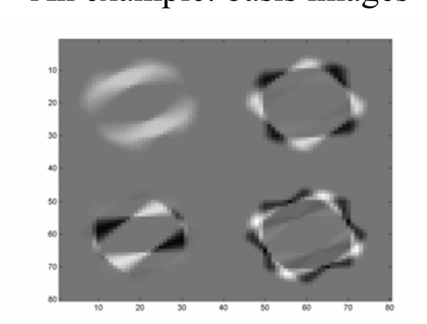
Appearance manifold approach

(Nayar et al. '96)

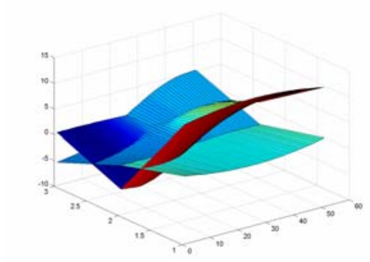
- for every object
  1. sample the set of viewing conditions
  2. Crop & scale images to standard size
  3. Use as feature vector
- apply a PCA over all the images
- keep the dominant PCs
- Set of views for one object is represented as a manifold in the projected space
- Recognition: What is nearest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

An example: input images

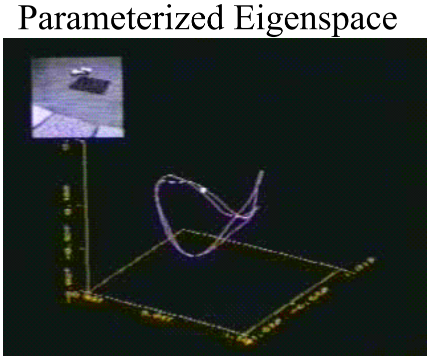
An example: basis images



An example: surfaces of first 3 coefficients

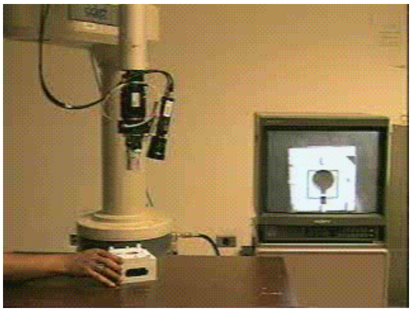


Parameterized Eigenspace



Recognition

Appearance-based vision for robot control



[Nayar, Nene, Murase 1994]

Limitations of these approaches

- Object must be segmented from background (How would one do this in non-trivial situations?
- Occlusion?
- The variability (dimension) in images is large, so is sampling feasible?
- How can one generalize to classes of objects?

Appearance-Based Vision: Lessons

Strengths

- Posing the recognition metric in the image space rather than a derived representation is more powerful than expected.
- Modeling objects from many images is not unreasonable given hardware developments.
- The data (images) may provide a better representations than abstractions for many tasks.

Appearance-Based Vision: Lessons

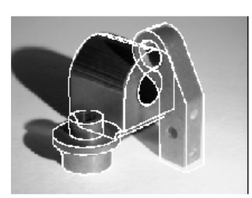
Weaknesses

- Segmentation or object detection is still an issue.
- To train the method, objects have to be observed under a wide range of conditions (e.g. pose, lighting, shape deformation).
- Limited power to extrapolate or generalize (abstract) to novel conditions.

Final Exam

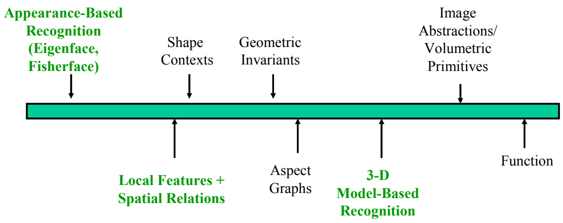
- Closed book
- One cheat sheet
  - Single piece of paper, handwritten, no photocopying, no physical cut & paste.
- What to study
  - Basically material presented in class, and supporting material from text
  - If it was in text, but NEVER mentioned in class, it is very unlikely to be on the exam
- Question style:
  - Short answer
  - Some longer problems to be worked out.

Model-Based Vision



- Given 3-D models of each object
- Detect image features (often edges, line segments, conic sections)
- Establish correspondence between model &image features
- Estimate pose
- Consistency of projected model with image.

A Rough Recognition Spectrum



Recognition by Hypothesize and Test

- General idea
  - Hypothesize object identity and pose
  - Recover camera parameters (widely known as backprojection)
  - Render object using camera parameters
  - Compare to image
- Issues
  - where do the hypotheses come from?
  - How do we compare to image (verification)?
- Simplest approach
  - Construct a correspondence for all object features to every correctly sized subset of image points
    - These are the hypotheses
  - Expensive search, which is also redundant.

Pose consistency

- Correspondences between image features and model features are not independent.
- A small number of correspondences yields a camera matrix --- the others correspondences must be consistent with this.
- Strategy:
  - Generate hypotheses using small numbers of correspondences (e.g. triples of points for a calibrated perspective camera, etc., etc.)
  - Backproject and verify


```
For all object frame groups O
  For all image frame groups F
    For all correspondences C between
      elements of F and elements
      of O

      Use F, C and O to infer the missing parameters
      in a camera model

      Use the camera model estimate to render the object

      If the rendering conforms to the image,
        the object is present
    end
  end
end
```


Figure from "Object recognition using alignment," D.P. Huttenlocher and S. Ullman, Proc. Int. Conf. Computer Vision, 1986, copyright IEEE, 1986

Voting on Pose

- Each model leads to many correct sets of correspondences, each of which has the same pose
  - Vote on pose, in an accumulator array
  - This is a hough transform, with all it's issues.

```
For all objects O
  For all object frame groups F(O)
    For all image frame groups F(I)
      For all correspondences C between
        elements of F(I) and elements
        of F(O)

        Use F(I), F(O) and C to infer object pose P(O)

        Add a vote to O's pose space at the bucket
        corresponding to P(O).
      end
    end
  end
end
For all objects O
  For all elements P(O) of O's pose space that have
    enough votes

    Use the P(O) and the
    camera model estimate to render the object

    If the rendering conforms to the image,
    the object is present
  end
end
```

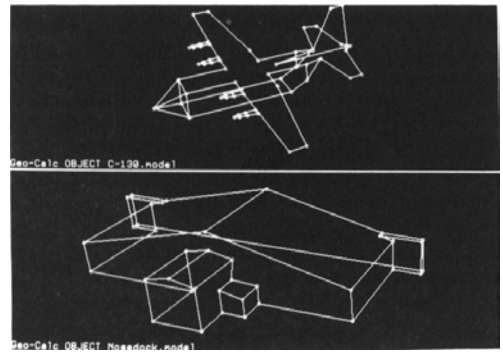



Figure from “The evolution and testing of a model-based object recognition system”, J.L. Mundy and A. Heller, Proc. Int. Conf. Computer Vision, 1990 copyright 1990 IEEE

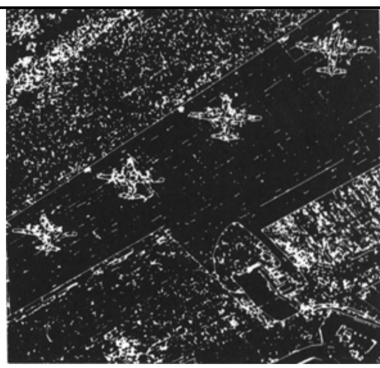


Figure from “The evolution and testing of a model-based object recognition system”, J.L. Mundy and A. Heller, Proc. Int. Conf. Computer Vision, 1990 copyright 1990 IEEE

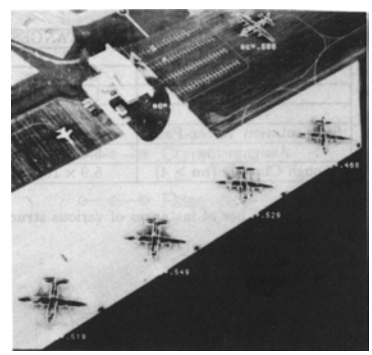


Figure from “The evolution and testing of a model-based object recognition system”, J.L. Mundy and A. Heller, Proc. Int. Conf. Computer Vision, 1990 copyright 1990 IEEE

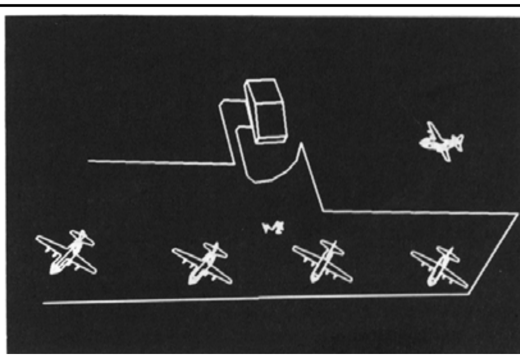


Figure from “The evolution and testing of a model-based object recognition system”, J.L. Mundy and A. Heller, Proc. Int. Conf. Computer Vision, 1990 copyright 1990 IEEE

Invariance

- Properties or measures that are independent of some group of transformation (e.g., rigid, affine, projective, etc.)
- For example, under affine transformations:
  - Collinearity
  - Parallelism
  - Intersection
  - Distance ratio along a line
  - Angle ratios of tree intersecting lines
  - Affine coordinates

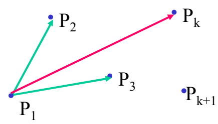
Invariance - 1

- There are geometric properties that are invariant to camera transformations
- Easiest case: view a plane object in scaled orthography.
- Assume we have three base points P_i (i=1..3) on the object
  - then any other point on the object can be written as

$$P_k = P_1 + \mu_{ka}(P_2 - P_1) + \mu_{kb}(P_3 - P_1)$$

- Now image points are obtained by multiplying by a plane affine transformation, so

$$\begin{aligned} p_k &= AP_k \\ &= A\left(P_1 + \mu_{ka}(P_2 - P_1) + \mu_{kb}(P_3 - P_1)\right) \\ &= p_1 + \mu_{ka}(p_2 - p_1) + \mu_{kb}(p_3 - p_1) \end{aligned}$$



Geometric hashing

- Vote on identity and correspondence using invariants
  - Take hypotheses with large enough votes
- Building a table:
  - Take all triplets of points in on model image to be base points P_1, P_2, P_3.
  - Take ever fourth point and compute μ's
  - Fill up a table, indexed by μ's, with
    - the base points and fourth point that yield those μ's
    - the object identity

Algorithm 18.3: Geometric hashing: voting on identity and point labels

```
For all groups of three image points T(I)
  For every other image point p
    Compute the μ's from p and T(I)
    Obtain the table entry at these values
      if there is one, it will label the three points in T(I)
      with the name of the object
      and the names of these particular points.
    Cluster these labels;
      if there are enough labels, backproject and verify
    end
  end
end
```

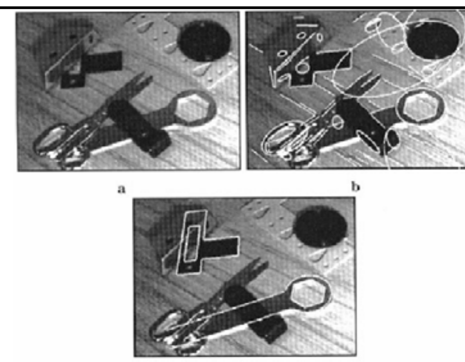
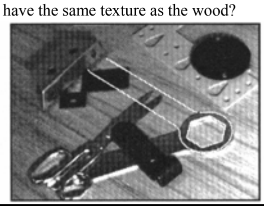



Figure from "Efficient model library access by projectively invariant indexing functions," by C.A. Rothwell et al., Proc. Computer Vision and Pattern Recognition, 1992, copyright 1992, IEEE

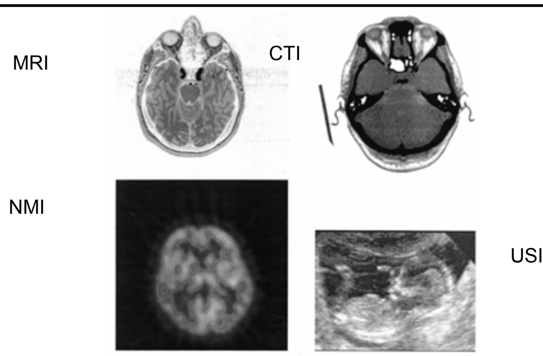
Verification

- Edge score
  - are there image edges near predicted object edges?
  - very unreliable; in texture, answer is usually yes
- Oriented edge score
  - are there image edges near predicted object edges with the right orientation?
  - better, but still hard to do well (see next slide)
- Texture
  - e.g. does the spanner have the same texture as the wood?



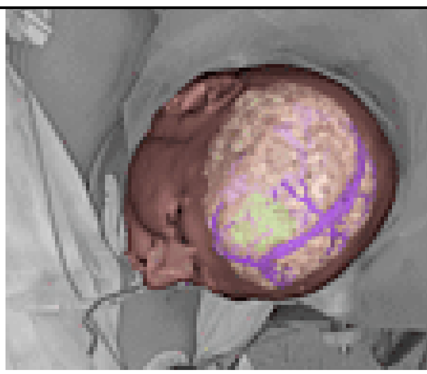
Application: Surgery

- To minimize damage by operation planning
- To reduce number of operations by planning surgery
- To remove only affected tissue
- Problem
  - ensure that the model with the operations planned on it and the information about the affected tissue lines up with the patient
  - display model information supervised on view of patient
  - **Big Issue**: coordinate alignment, as above



Reprinted from Image and Vision Computing, v. 13, N. Ayache, "Medical computer vision, virtual reality and robotics", Page 296, copyright, (1995), with permission from Elsevier Science

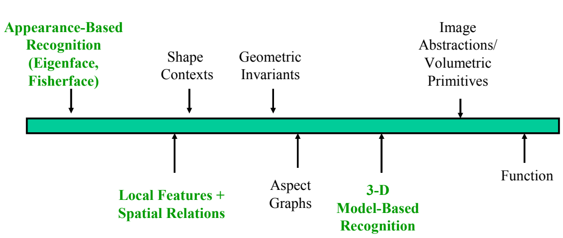
Figures by kind permission of Eric Grimson; further information can be obtained from his web site http://www.ai.mit.edu/people/welg/welg.html.



Figures by kind permission of Eric Grimson; further information can be obtained from his web site http://www.ai.mit.edu/people/welg/welg.html.

Figures by kind permission of Eric Grimson; further information can be obtained from his web site http://www.ai.mit.edu/people/welg/welg.html.

A Rough Recognition Spectrum



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, affine-invariant 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.

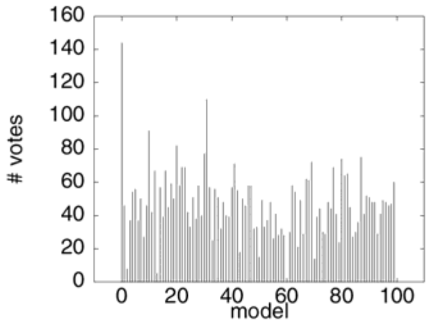


Figure from "Local grayvalue invariants for image retrieval," by C. Schmid and R. Mohr, IEEE Trans. Pattern Analysis and Machine Intelligence, 1997 copyright 1997, IEEE

Probabilistic interpretation

- Write

$$P\{\text{patch of type } i \text{ appears in image}|j\text{'th pattern is present}\} = p_{ij}$$

$$P\{\text{patch of type } i|\text{no pattern is present}\} = p_{ix}$$

- Assume $p_{ij} = \mu$ if the pattern can produce this patch and 0 otherwise

$p_{ix} = \lambda < \mu$ for all i

- Lik that n_p patches came from that pattern and $n_i - n_p$ patches come from noise, is

$$P(\text{interpretation}|\text{pattern}) = \lambda^{n_p}\mu^{(n_i-n_p)}$$

Employ spatial relations

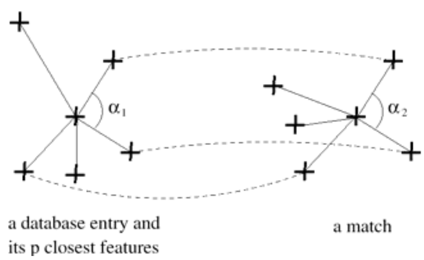


Figure from "Local grayvalue invariants for image retrieval," by C. Schmid and R. Mohr, IEEE Trans. Pattern Analysis and Machine Intelligence, 1997 copyright 1997, IEEE

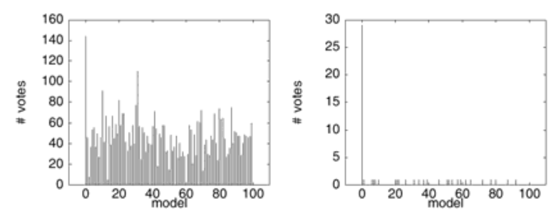
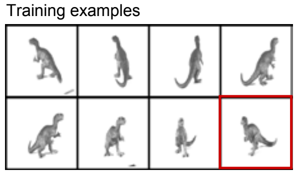


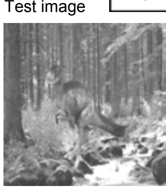
Figure from "Local grayvalue invariants for image retrieval," by C. Schmid and R. Mohr, IEEE Trans. Pattern Analysis and Machine Intelligence, 1997 copyright 1997, IEEE

Example

Training examples



Test image



Finding faces using relations

- Strategy:
  - Face is eyes, nose, mouth, etc. with appropriate relations between them
  - build a specialised detector for each of these (template matching) and look for groups with the right internal structure
  - Once we've found enough of a face, there is little uncertainty about where the other bits could be

Finding faces using relations

- Strategy: compare

$$P(\text{one face at } \boldsymbol{F} | \boldsymbol{X}_{\text{le}} = \boldsymbol{x}_1, \boldsymbol{X}_{\text{re}} = \boldsymbol{x}_2, \boldsymbol{X}_{\text{m}} = \boldsymbol{x}_3, \boldsymbol{X}_{\text{n}} = \boldsymbol{x}_4, \text{all other responses})$$

with

$$P(\text{no face} | \boldsymbol{X}_{\text{le}} = \boldsymbol{x}_1, \boldsymbol{X}_{\text{re}} = \boldsymbol{x}_2, \boldsymbol{X}_{\text{m}} = \boldsymbol{x}_3, \boldsymbol{X}_{\text{n}} = \boldsymbol{x}_4, \text{all other responses})$$

Notice that once some facial features have been found, the position of the rest is quite strongly constrained.

Figure from, "Finding faces in cluttered scenes using random labelled graph matching," by Leung, T. ;Burl, M and Perona, P., Proc. Int. Conf. on Computer Vision, 1995 copyright 1995, IEEE

Figure from, "Finding faces in cluttered scenes using random labelled graph matching," by Leung, T. ;Burl, M and Perona, P., Proc. Int. Conf. on Computer Vision, 1995 copyright 1995, IEEE

Even without shading, shape reveals a lot - line drawings

Scene Interpretation

"The Swing"
Fragonard, 1766