Observer motion problem

From image motion, compute:

- observer translation
  \((T_x, T_y, T_z)\)
- observer rotation
  \((R_x, R_y, R_z)\)
- depth at every location
  \(Z(x, y)\)

Observer just translates toward FOE

Directions of velocity vectors intersect at FOE

But... simple strategy doesn't work if observer also rotates

Observer motion problem, revisited

From image motion, compute:

- Observer translation
  \((T_x, T_y, T_z)\)
- Observer rotation
  \((R_x, R_y, R_z)\)
- Depth at every location
  \(Z(x, y)\)

Observer undergoes both translation + rotation

Equations of observer motion

<table>
<thead>
<tr>
<th>Translation</th>
<th>Rotation</th>
<th>Depth</th>
</tr>
</thead>
<tbody>
<tr>
<td>((T_x, T_y, T_z))</td>
<td>((R_x, R_y, R_z))</td>
<td>(Z(x, y))</td>
</tr>
</tbody>
</table>

\[
V_x = \frac{(-T_x + xT_z)}{Z} + \frac{R_x y z}{z^2}y - \frac{R_y (x^2 + 1)}{z^2} + \frac{R_z y}{z^2}
\]

\[
V_y = \frac{(-T_y + yT_z)}{Z} + \frac{R_y y z}{z^2} - \frac{R_x x z}{z^2} - \frac{R_z x}{z^2}
\]

Translational Component
Rotational Component
Along a depth discontinuity, velocity differences depend only on observer translation.

Velocity differences point to the focus of expansion.

Alignment methods

Find an object model and geometric transformation that best match the viewed image.

\[ V \] viewed object (image)
\[ M_i \] object models
\[ T_{ij} \] allowable transformations between viewed object and models
\[ F \] measure of fit between \( V \) and the expected appearance of model \( M_i \) under the transformation \( T_{ij} \)

**GOAL:** Find a combination of \( M_i \) and \( T_{ij} \) that maximizes the fit \( F \)

Alignment method: recognition process

1. Find best transformation \( T_{ij} \) for each model \( M_i \) (optimizing over possible views)
2. Find \( M_i \) whose best \( T_{ij} \) gives the best match to image \( V \)
Recognition by linear combination of views

model views

LC2 is a linear combination of M1 and M2 that best matches the novel view

Predicting object appearance

two known views of obelisk

Recognition process:
1. compute $\alpha, \beta$ that predict $P1$ & $P2$
2. use $\alpha, \beta$ to predict other points
3. evaluate fit of model to image

Why is face recognition hard?

changing posechanging illumination

aging

clutter

occlusion

changing expression

Eigenfaces for recognition (Turk & Pentland)

Principal Components Analysis (PCA)

Goal: reduce the dimensionality of the data while retaining as much information as possible in the original dataset

PCA allows us to compute a linear transformation that maps data from a high dimensional space to a lower dimensional subspace
Eigenfaces for recognition (Turk & Pentland)

Perform PCA on a large set of training images, to create a set of eigenfaces, \( E_i(x,y) \), that span the data set.

First components capture most of the variation across the data set, later components capture subtle variations.

\( \Psi(x,y) \): average face (across all faces)

Each face image \( F(x,y) \) can be expressed as a weighted combination of the eigenfaces \( E_i(x,y) \):

\[
F(x,y) = \Psi(x,y) + \sum_i w_i^* E_i(x,y)
\]

http://vismod.media.mit.edu/vismod/demos/facerec/basic.html

Representing individual faces

Each face image \( F(x,y) \) can be expressed as a weighted combination of the eigenfaces \( E_i(x,y) \):

\[
F(x,y) = \Psi(x,y) + \sum_i w_i^* E_i(x,y)
\]

Recognition process:
1. Compute weights \( w_i \) for novel face image.
2. Find image \( m \) in face database with most similar weights, e.g.

\[
\min \sum_{i=1}^{k} (w_i - w_i^m)^2
\]

Face detection: Viola & Jones

Multiple view-based classifiers based on simple features that best discriminate faces vs. non-faces.

Most discriminating features learned from thousands of samples of face and non-face image windows.

Attentional mechanism: cascade of increasingly discriminating classifiers improves performance.

Viola & Jones use simple features

Use simple rectangle features:

\[
\Sigma I(x,y) \text{ in gray area} - \Sigma I(x,y) \text{ in white area}
\]

within 24 \times 24 image sub-windows

- Initially consider 160,000 potential features per sub-window!
- Features computed very efficiently

Which features best distinguish face vs. non-face?

Learn most discriminating features from thousands of samples of face and non-face image windows.
Learning the best features

**weak classifier using one feature:**

\[ h(x, f, p, \theta) = \begin{cases} 1 & \text{if } p f(x) < p \theta \\ 0 & \text{otherwise} \end{cases} \]

\( x \) = image window
\( f \) = feature
\( p = +1 \text{ or } -1 \)
\( \theta \) = threshold

\( n \) training samples, equal weights, known classes

\[ C(x) = \begin{cases} 1 & \sum_{i=1}^{T} \alpha_i h_i(x) \geq \tau \\ 0 & \text{otherwise} \end{cases} \]

**AdaBoost:**

- use classification errors to update weights
- normalize weights
- find next best weak classifier
- weight sum (among examples)

\[ \epsilon_{t} = \min_{f,p,T} \sum_{i} w_i [ h(x_i, f, p, \theta) - y_i ] \]

\( \eta \)

~ 200 features yields good results for "monolithic" classifier

---

“Attentional cascade” of increasingly discriminating classifiers

Early classifiers use a few highly discriminating features, low threshold

- 1st classifier uses two features, removes 50% non-face windows

- later classifiers distinguish harder examples

- Increases efficiency
- Allows use of many more features

→ Cascade of 38 classifiers, using ~6000 features

---

Feature based vs. holistic processing

Tanaka & Simonyi (2016), Sinha et al. (2006)

- composite face effect
- face inversion effect
- whole-part effect

Feature based vs. holistic processing

composite face effect

face inversion effect

- identical top halves seen as different when aligned with different bottom halves
- when misaligned, top halves perceived as identical

whole-part effect

- inversion disrupts recognition of faces more than other objects
- prosopagnosics do not show effect

Identification of “studied” face is significantly better in whole vs. part condition
The power of averages, Burton et al. (2005)

Human recognition of average faces
Burton et al. (2005)

Performance: texture + shape images

Performance: shape-free images

What is an artificial neural network?

Computing in a "typical" neural network

Network of simple neuron-like computing elements...

...that can learn to associate inputs with desired outputs

How does each unit integrate its inputs to produce an output?

\[ w_0 \cdot I_0 + w_1 \cdot I_1 + w_2 \cdot I_2 + \ldots + w_n \cdot I_n > 0 \]

Activation

\[ \text{sigmoid} \]

output

sum of weighted inputs \( \rightarrow \) sigmoid function \( \rightarrow \) output between 0 and 1
Learning to recognize input patterns

feedforward processing

network weights can be learned from training examples (map inputs to correct outputs)

back-propagation:
iterative algorithm progressively reduces error between computed and desired output until performance is satisfactory

on each iteration:

• compute output of current network and assess performance
• compute weight adjustments from hidden to output layer that reduce output errors
• compute weight adjustments from input to hidden units that improve hidden layer
• change network weights, incorporating a rate parameter