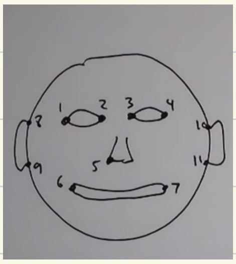
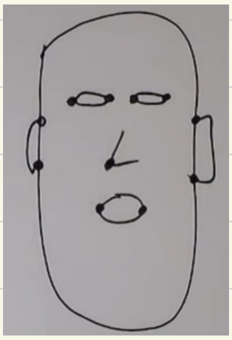
active shape models

a different kind of object detection

(not template based)

training data

establish correspondence for each training example (mannual)

$\{(x_1^j, y_1^j), (x_2^j, y_2^j), \ldots, (x_n^j, y_n^j)\}$ n feature points for person j $\quad z^j \quad 2n\times 1$

algorithm 1: alignment

1) translate all shapes to be centered at (0,0)
2) fix one shape z', scale so $\|z'\| = 1$
3) scale and rotate all other shapes to align with this shape

$$a_j = \frac{z^j \cdot z'}{\|z^j\|^2} \qquad b_j = \frac{\sum_{i=1}^{n} (x_i^j y_i' - x_i' y_i^j)}{\|z^j\|^2}$$

$$s_j = \sqrt{(a_j)^2 + (b_j)^2} \qquad \theta^j = \tan^{-1}(b_j / a_j)$$

$$\begin{bmatrix} \hat{x}_i^j \\ \hat{y}_i^j \end{bmatrix} = s_j \begin{bmatrix} \cos\theta^j & \sin\theta^j \\ -\sin\theta^j & \cos\theta^j \end{bmatrix} \begin{bmatrix} x_i^j \\ y_i^j \end{bmatrix}$$

procrustes analysis

Now we have S sets of aligned training shapes.

Each is described by a 2n-vector of feature points

we want to reduce the dimensionality of this set to a number k << 2n

core idea: principal component analysis (PCA)

algorithm 2: PCA

1) compute mean of data $\mu = \frac{1}{S} \sum_{i=1}^{S} z^i \quad 2n\times 1$
2) compute co-variance

$$\Sigma = \frac{1}{S-1} \sum_{i=1}^{S} (z^i - \mu)(z^i - \mu)^T$$

unbiased $\quad 2n \times 2n$

3) compute eigenvalues and eigenvector of Σ

(λ_j, v_j) s.t $\lambda_1 > \lambda_2 > \lambda_3 \ldots$ $\quad \Sigma v_j = \lambda_j v_j$

4) each eigenvalue λ_j gives variance of data in the direction v_j

compute total variance as $T = \sum_{j=1}^{S} \lambda_j$

choose the k largest eigenvalues to account for p% of the total variance

$$\frac{\sum_{j=1}^{k} \lambda_j}{T} \geq 0.99$$

we can approximate any shape z as

$z = \mu + pb$, k << 2n

$z: 2n\times 1 \quad p: 2n\times k \quad b: k\times 1$

$v_1 \cdots v_k$

the k-dimensional vector b defines a small number of parameters for the deformable shape model

choosing b corresponds to a candidate shape.

The small set of values in the b-vector are like "knobs" we can turn to get the best fit.

The "modes" of v1...vk are often intuitive. eg. for faces, they may correspond to shaking, nodding, smiling, etc

fitting the model to new data.

assume we have a 2n-vector Y, we want to fit the model - find the best translation, rotation, scale (t,s,θ) and model parameter b

min $\|Y - M(\mu - Pb)\|^2$

$$M(x) = s \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} x + t$$

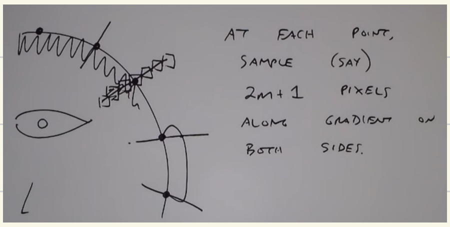
algorithm 3: matching model to target

1) init b = 0
2) generate model points $x = \mu + Pb$
3) find s,θ,t to best fit Y to x
   (algo 1) - gives the new Y
4) project Y into x space ($y = M^{-1}Y$)
5) update model parameters $b = P^T(y - \mu)$
6) go to step2, iterate until convergence

How do we know what image points Y should belong to the model?

algo4: active shape model
1) init $b = 0$, $x = \mu$
2) search around each x_i for best nearby image point y_i
3) fit new parameters (s,t,θ,b) to y_i (algo 3)
4) enforce constraint that $|b_i| < 3\lambda_i$
(so shapes are “reasonable”)
5) iterate

we can do better in step2



for each point, both (x,y) and a 2m+1 grayscale vector - do PCA on the whole concatenated vector