Final Project

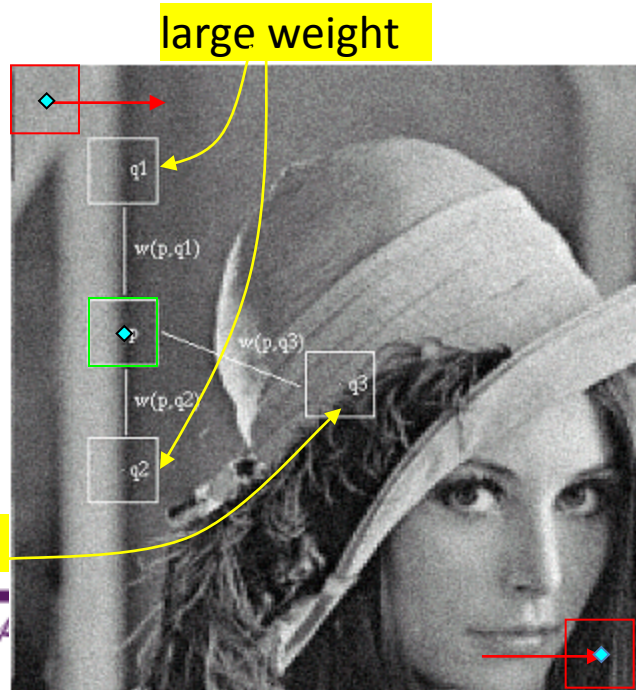
CS3570 introduction to multimedia

Non-linear Image Denoising

- Given a noisy image $v=\{v(i)\,|\,i\in I\}$, the estimated value $NL[v](i)$, for a pixel i, is computed as a weighted average of all the pixels in the image

$$NL[v](i)=\sum_{j\in I}\omega(i,j)v(j)$$

- where the family weights $\{w(i,j)\}_j$ depend on the similarity between the pixels i and j, and satisfy the usual conditions $0\le w(i,j)\le 1$ and $\sum_{j\in I} w(i,j)=1$.



Non-linear Image Denoising

- The similarity between two pixels *i* and *j* depends on the similarity of the intensity gray level vectors *v*(*Ni*) and *v*(*Nj*)
- This similarity is measured as a decreasing function of the weighted Euclidean distance $\|v(\mathcal{N}_i) - v(\tilde{\mathcal{N}}_j)\|_{2,a}^2$, where $a > 0$ is the standard deviation of the Gaussian kernel.
- Weight function

$$w(i,j) = \frac{1}{Z(i)} e^{-\frac{||v(\mathcal{N}_i) - v(\mathcal{N}_j)||_{2,a}^2}{h^2}},$$

where *Z*(*i*) is the normalizing constant

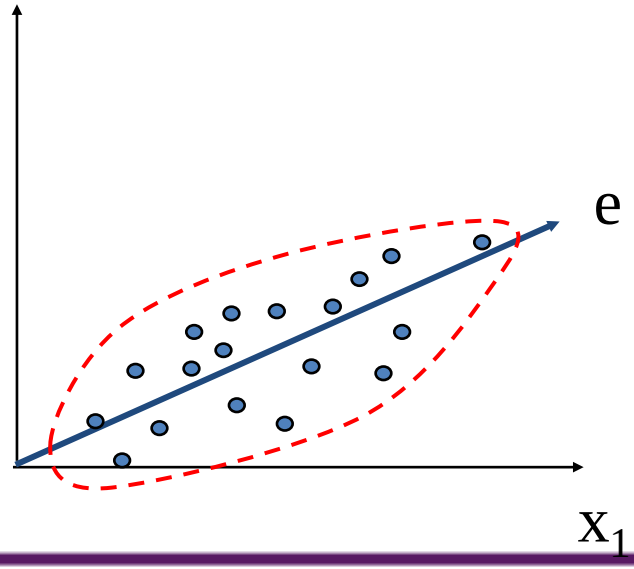
$$Z(i) = \sum_j e^{-\frac{||v(\mathcal{N}_i) - v(\mathcal{N}_i)||_{2,a}^2}{h^2}}$$

Non-linear Image Denoising – comparison with other methods

Figure 5. Denoising experience on a natural image. From left to right and from top to bottom: noisy image (standard deviation 20), Gauss filtering, anisotropic filtering, Total variation, Neighborhood filtering and NL-means algorithm. The removed details must be compared with the method noise experience, Figure 4.

Face Recognition Using Eigenfaces

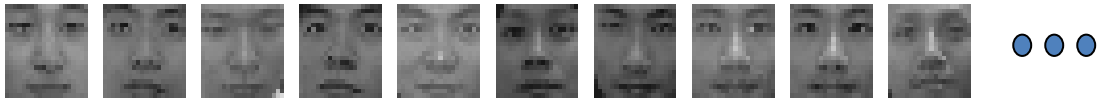
- Face Images are projected into a feature space ("Face Space") that best encodes the variation among known face images.
- The face space is defined by the "eigenfaces", which are the eigenvectors of the set of faces.
- Eigen Vectors show the direction of axes of a fitted ellipsoid
- Eigen Values show the significance of the corresponding axis



PCA face model

- Face model

Training images



$$F = (g_1, g_2, \cdots g_n)^T$$

Normalized face data $$D = \left[F_1 - \overline{F}, F_2 - \overline{F}, ..., F_m - \overline{F}\right]$$

Covariance matrix $$C = \frac{1}{m-1} DD^T$$ SVD $$I = \overline{I} + \sum_{i=1}^{e} \alpha_i v_i$$

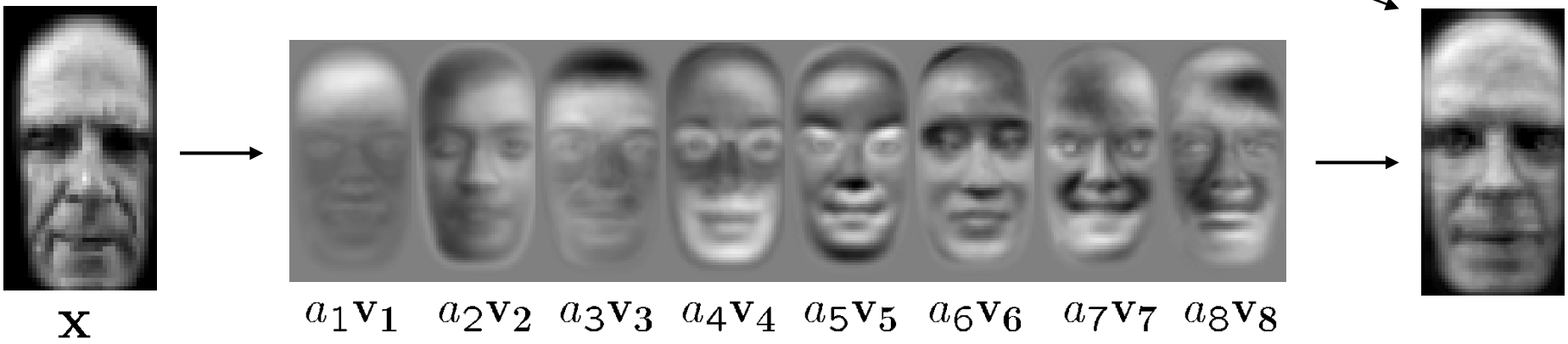
PCA model

eigenfaces

- The eigenfaces $\mathbf{v_1}$, ..., $\mathbf{v_K}$ span the space of faces
  - A face is converted to eigenface coordinates by

$$\mathbf{x} \rightarrow (\underbrace{(\mathbf{x}-\overline{\mathbf{x}})\cdot\mathbf{v_1}}_{a_1}, \underbrace{(\mathbf{x}-\overline{\mathbf{x}})\cdot\mathbf{v_2}}_{a_2}, \ldots, \underbrace{(\mathbf{x}-\overline{\mathbf{x}})\cdot\mathbf{v_K}}_{a_K})$$

$$\mathbf{x} \approx \overline{\mathbf{x}} + a_1\mathbf{v_1} + a_2\mathbf{v_2} + \ldots + a_K\mathbf{v_K}$$



Background subtraction / synthesis

- Moving object detection in video sequences is one of the main tasks in many computer vision applications.
- Background subtraction is an common approach for this task. The idea is to compare the current image against background model which learned by GMM
- C. Staufer, W.E.L. Grimson, “Adaptive background mixture models for real-time tracking,”*CVPR*, Vol. 2, pp. 246-252, June 1999.

Background subtraction

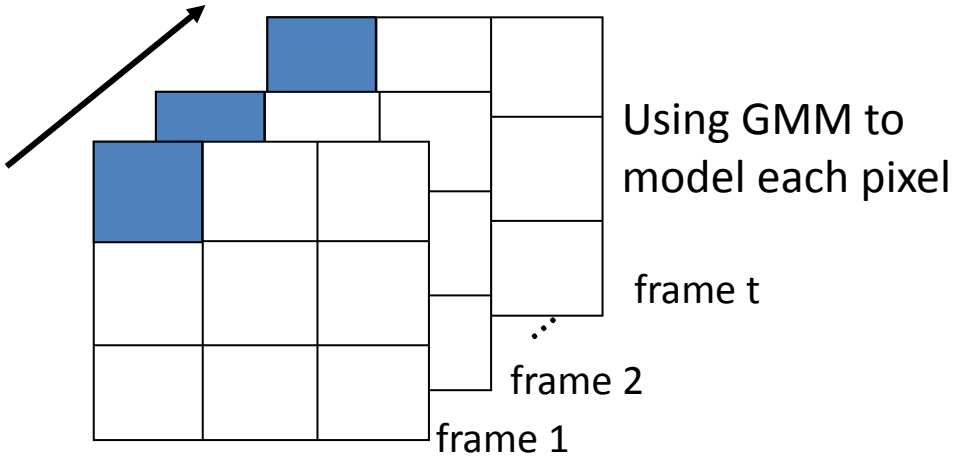
synthesis

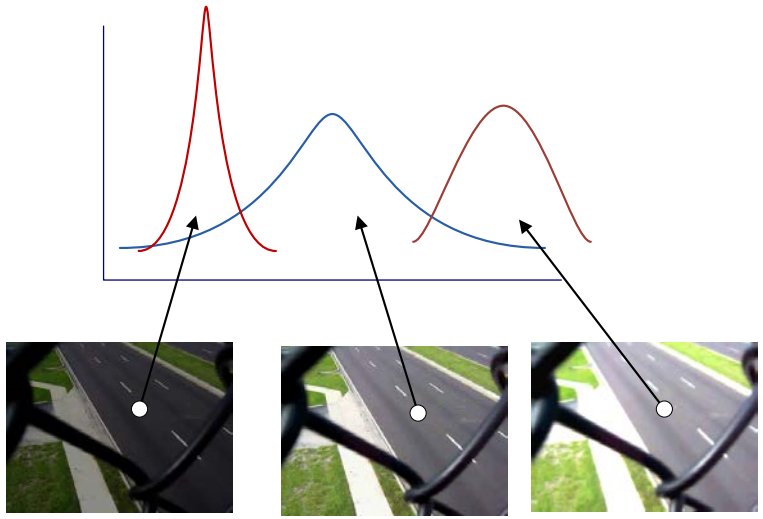
Background subtraction / synthesis

- Each pixel modeled with a mixture of Gaussians

$$P(.) = \sum_{i=1}^{K} w_i \eta(., \mu_i, \Sigma_i)$$

$$\eta(X_t, \mu, \Sigma) = \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} e^{-1/2(X_t-\mu)^T \Sigma^{-1}(X_t-\mu)}$$





Background subtraction / synthesis

- Updating the GMM background model

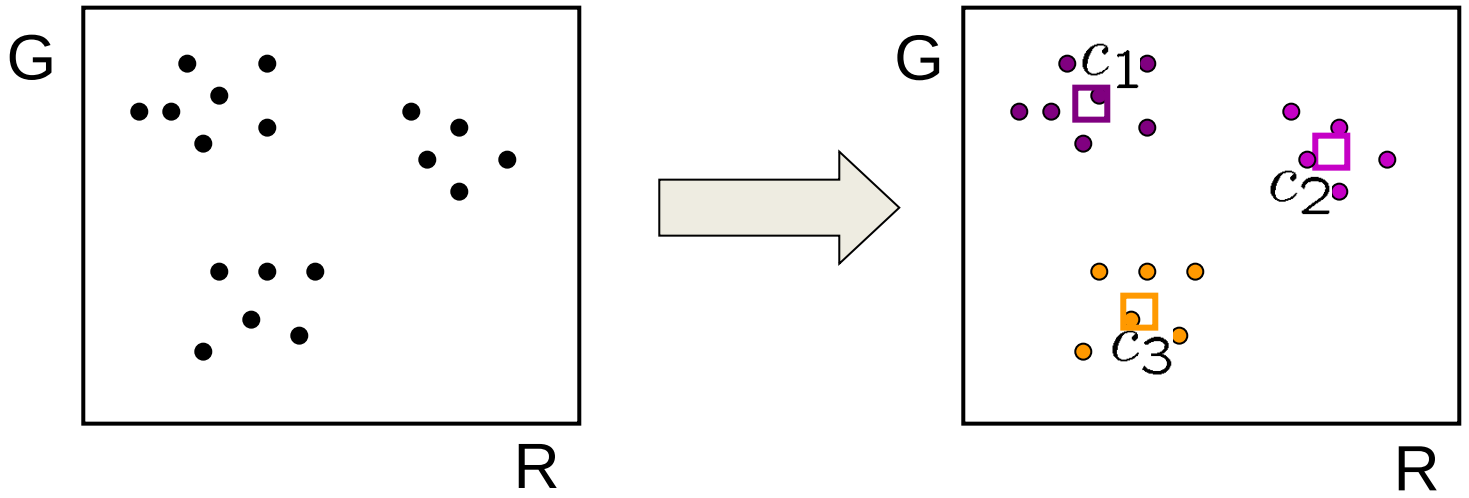
$$\omega_{k,t} = (1-\alpha)\omega_{k,t-1} + \alpha M_{k,t}$$
$$\mu_t = (1-\rho)\mu_{t-1} + \rho X_t$$
$$\sigma_t^{\ 2} = (1-\rho)\sigma^2{}_{t-1} + \rho(X_t - \mu_t)^T(X_t - \mu_t)$$
$$\rho = \alpha\eta(X_t \mid \mu_k, \sigma_k)$$

- First B states are labeled as background states

$$B = \underset{b}{argmin}\left(\sum_{k=1}^{b}\omega_k > T\right)$$

Clustering

- How to choose the representative colors?
  - This is a clustering problem!



Objective

- Each point should be as close as possible to a cluster center
  - Minimize sum squared distance of each point to closest center

$$\sum_{\text{clusters } i} \quad \sum_{\text{points p in cluster } i} \|p - c_i\|^2$$

K-means clustering

- K-means clustering algorithm
  1. Randomly initialize the cluster centers, c_1, ..., c_K
  2. Given cluster centers, determine points in each cluster
     - For each point p, find the closest c_i. Put p into cluster i
  3. Given points in each cluster, update c_i to be the mean of all points in cluster i
  4. If c_i have changed, repeat Step 2

- Properties
  - Will always converge to *some* solution
  - Can be a "local minimum"
    - does not always find the global minimum of objective function: $\sum_{\text{clusters } i} \sum_{\text{points p in cluster } i} \|p - c_i\|^2$

8. Background subtraction / synthesis

- In Image composition, a new image *I(x,y)* can be blended from a background image *B(x,y)* and foreground image *F(x,y)* with its alpha matte α*(x,y)* $I = \alpha F + (1-\alpha)B$
- Automatically replace the background region by another background image in the input image