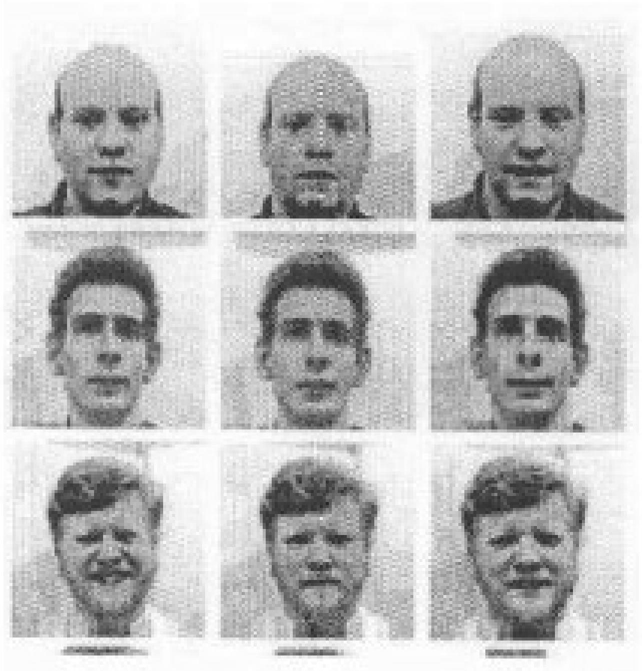
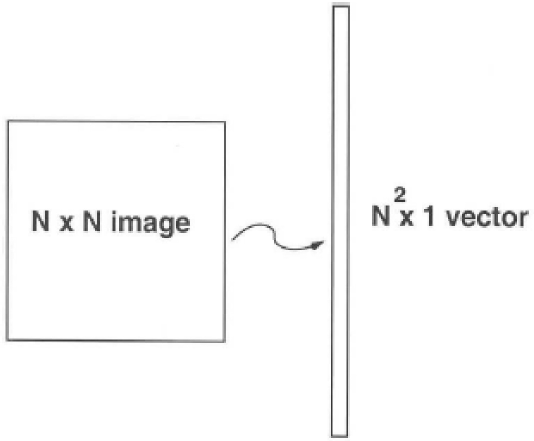
Application to Faces (from slides of CS479/679 Pattern Recognition Dr. George Bebis)

- Computation of low-dimensional basis (i.e.,eigenfaces):

Step 1: obtain face images I_1, I_2, ..., I_M (training faces)

(**very important:** the face images must be *centered* and of the same *size*)





Step 2: represent every image I_i as a vector Γ_i

- Computation of the eigenfaces – cont.

Step 3: compute the average face vector Ψ:

$$\Psi = \frac{1}{M} \sum_{i=1}^{M} \Gamma_i$$

Step 4: subtract the mean face:

$$\Phi_i = \Gamma_i - \Psi$$

Step 5: compute the covariance matrix C:

$$C = \frac{1}{M} \sum_{n=1}^{M} \Phi_n \Phi_n^T = \frac{AA^T}{M} \quad (N^2 \mathrm{x} N^2 \text{ matrix})$$

where $A = [\Phi_1 \ \Phi_2 \cdots \Phi_M] \quad (N^2 \mathrm{x} M \text{ matrix})$

- Computation of the eigenfaces – cont.

Step 6: compute the eigenvectors u_i of AA^T $\Rightarrow$ $AA^T u_i = \lambda_i u_i$

The matrix AA^T is very large --> not practical !!

Step 6.1: consider the matrix $A^T A$ (M x M matrix)

Step 6.2: compute the eigenvectors v_i of $A^T A$

$$A^T A v_i = \mu_i v_i$$

What is the relationship between us_i and v_i?

$A^T A v_i = \mu_i v_i \Rightarrow AA^T A v_i = \mu_i A v_i \Rightarrow$

$CAv_i = \mu_i A v_i$ or $Cu_i = \mu_i u_i$ where $u_i = Av_i$

$$u_i = Av_i \quad and \quad \lambda_i = \mu_i$$

- Computation of the eigenfaces – cont.

Note 1: AA^T can have up to N^2 eigenvalues and eigenvectors.

Note 2: $A^T A$ can have up to M eigenvalues and eigenvectors.

Note 3: The M eigenvalues of $A^T A$ (along with their corresponding eigenvectors) correspond to the M *largest* eigenvalues of AA^T (along with their corresponding eigenvectors).

Step 6.3: compute the M best eigenvectors of AA^T: $u_i = Av_i$ ***(i.e., using A^TA)***

(**important:** normalize u_i such that $\|u_i\| = 1$)

Step 7: keep only K eigenvectors (corresponding to the K largest eigenvalues)

each face Φ_i can be represented as follows:

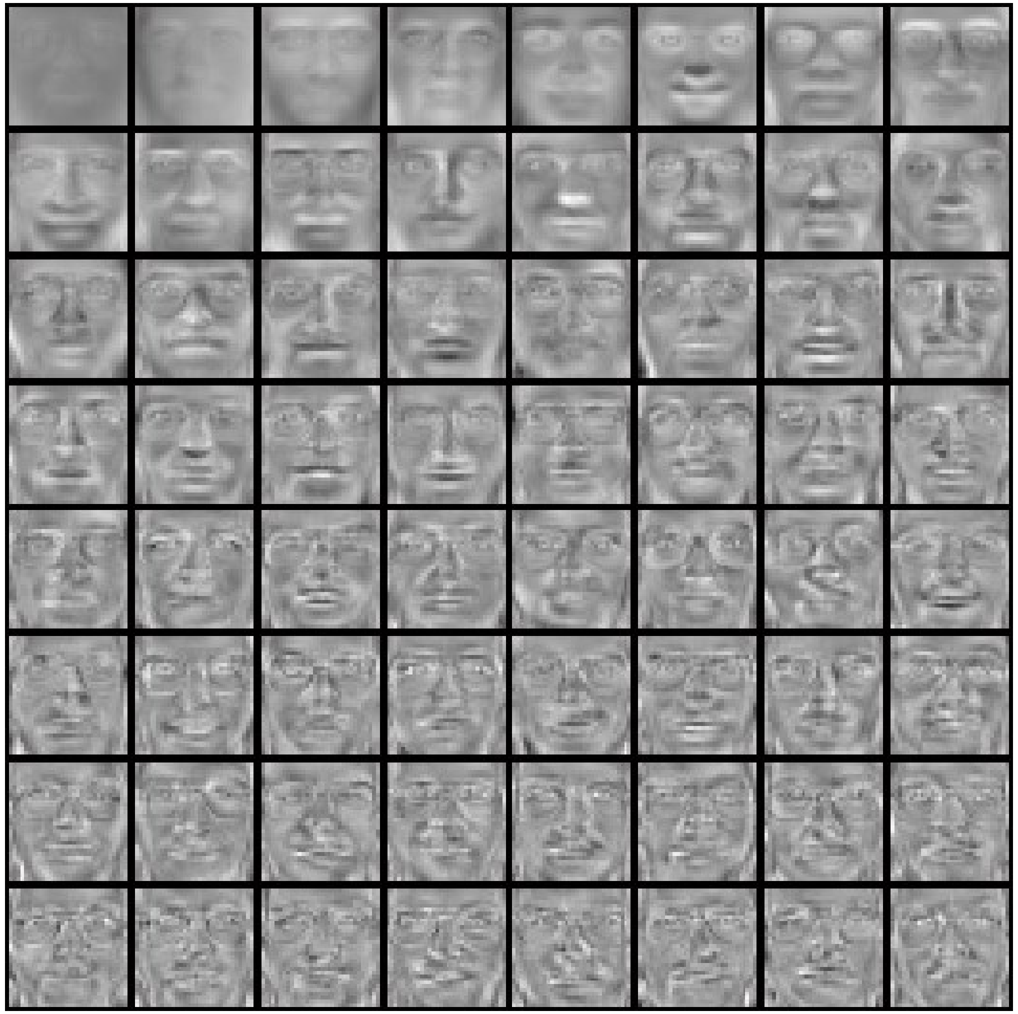
$$\hat{\Phi}_i - mean = \sum_{j=1}^{K} w_j u_j, \quad (w_j = u_j^T \Phi_i) \Rightarrow \Omega = \begin{bmatrix} w_1 \\ w_2 \\ \cdots \\ w_K \end{bmatrix}$$

Example

Example (cont'd)

Mean: μ

Top eigenvectors: $u_1, \ldots u_k$



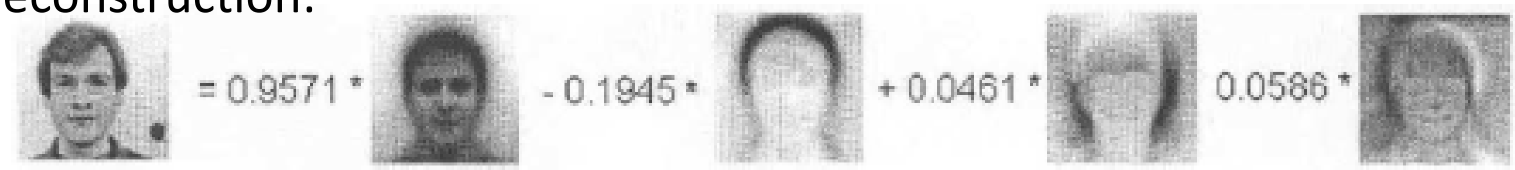
- Representing faces onto this basis

- Each face (minus the mean) Φ_i in the training set can be represented as a linear combination of the best K eigenvectors:

$$\hat{\Phi}_i - mean = \sum_{j=1}^{K} w_j u_j, \quad (w_j = u_j^T \Phi_i) \quad (where \| u_j \| = 1)$$

(we call the u_j's *eigenfaces*)

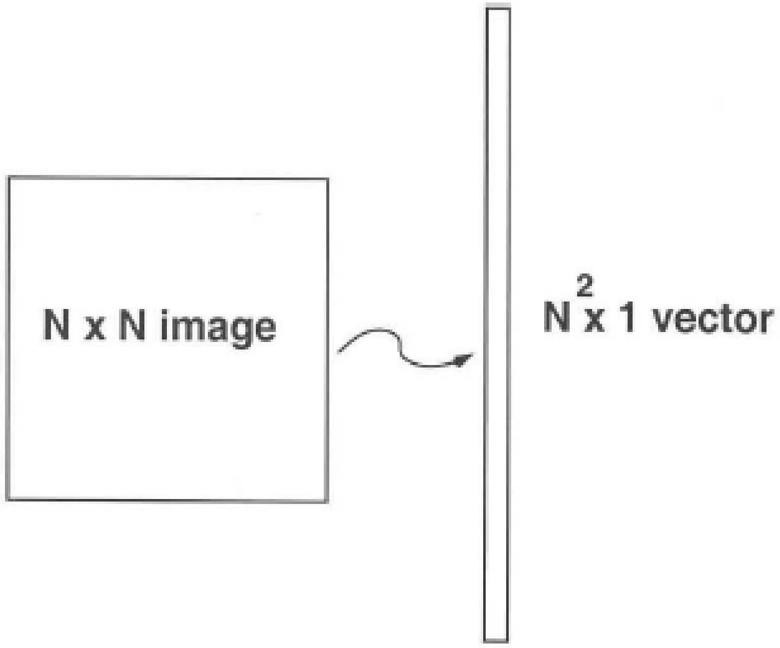
Face reconstruction:



Case Study: Eigenfaces for Face Detection/Recognition

- M. Turk, A. Pentland, "Eigenfaces for Recognition", *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71-86, 1991.

- Face Recognition
  - The simplest approach is to think of it as a template matching problem.
  - Problems arise when performing recognition in a high-dimensional space.
  - Use *dimensionality reduction*!



Eigenfaces

- Face Recognition Using Eigenfaces

- Given an unknown face image Γ (centered and of the same size like the training faces) follow these steps:

Step 1: normalize Γ: $\Phi = \Gamma - \Psi$

Step 2: project on the eigenspace

$$\hat{\Phi} = \sum_{i=1}^{K} w_i u_i \quad (w_i = u_i^T \Phi) \qquad (where \| u_i \| = 1)$$

Step 3: represent Φ as: $\Omega = \begin{bmatrix} w_1 \\ w_2 \\ \cdots \\ w_K \end{bmatrix}$

Step 4: find $e_r = \min_l \|\Omega - \Omega^l\|$ where $\| \Omega - \Omega^l \| = \sum_{i=1}^{K} (w_i - w_i^l)^2$

Step 5: if $e_r < T_r$, then Γ is recognized as face l from the training set.

The distance e_r is called *distance in face space (difs)*

Eigenfaces

- Face Detection Using Eigenfaces

- Given an unknown image Γ

<u>Step 1:</u> compute $\Phi = \Gamma - \Psi$

<u>Step 2:</u> compute $\hat{\Phi} = \sum_{i=1}^{K} w_i u_i \quad (w_i = u_i^T \Phi) \qquad (where \| u_i \| = 1)$

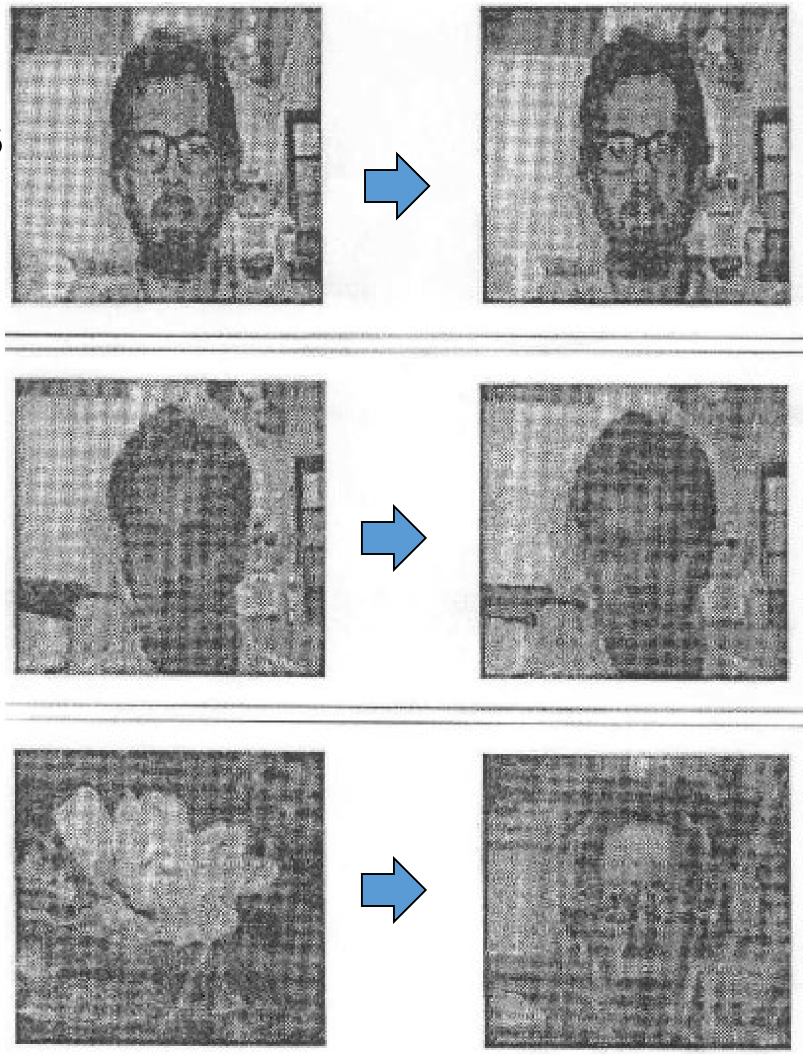
<u>Step 3:</u> compute $e_d = \|\Phi - \hat{\Phi}\|$

<u>Step 4:</u> if $e_d < T_d$, then Γ is a face.

- The distance e_d is called <u>distance from face space (dffs)</u>

Eigenfaces

Input Reconstructed



Reconstructed image looks like a face.

Reconstructed image looks like a face.

Reconstructed image looks like a face again!

Reconstruction using partial information

- Robust to partial face occlusion.

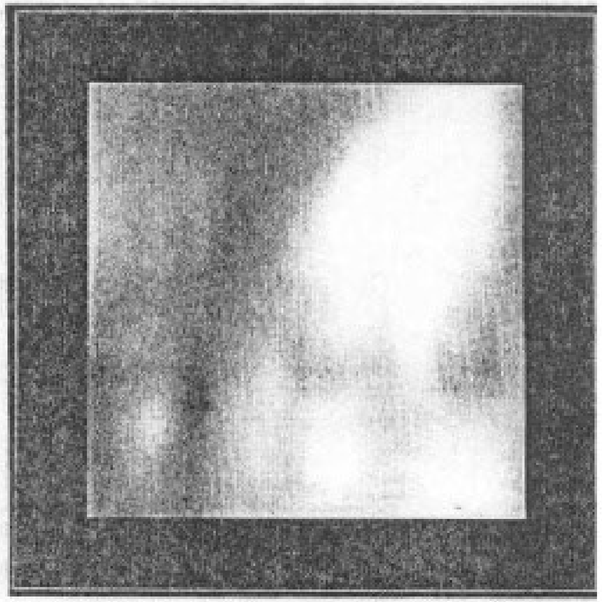
Input Reconstructed

Eigenfaces

- Face detection, tracking, and recognition

Visualize dffs:

$$e_d = \|\Phi - \hat{\Phi}\|$$



Limitations

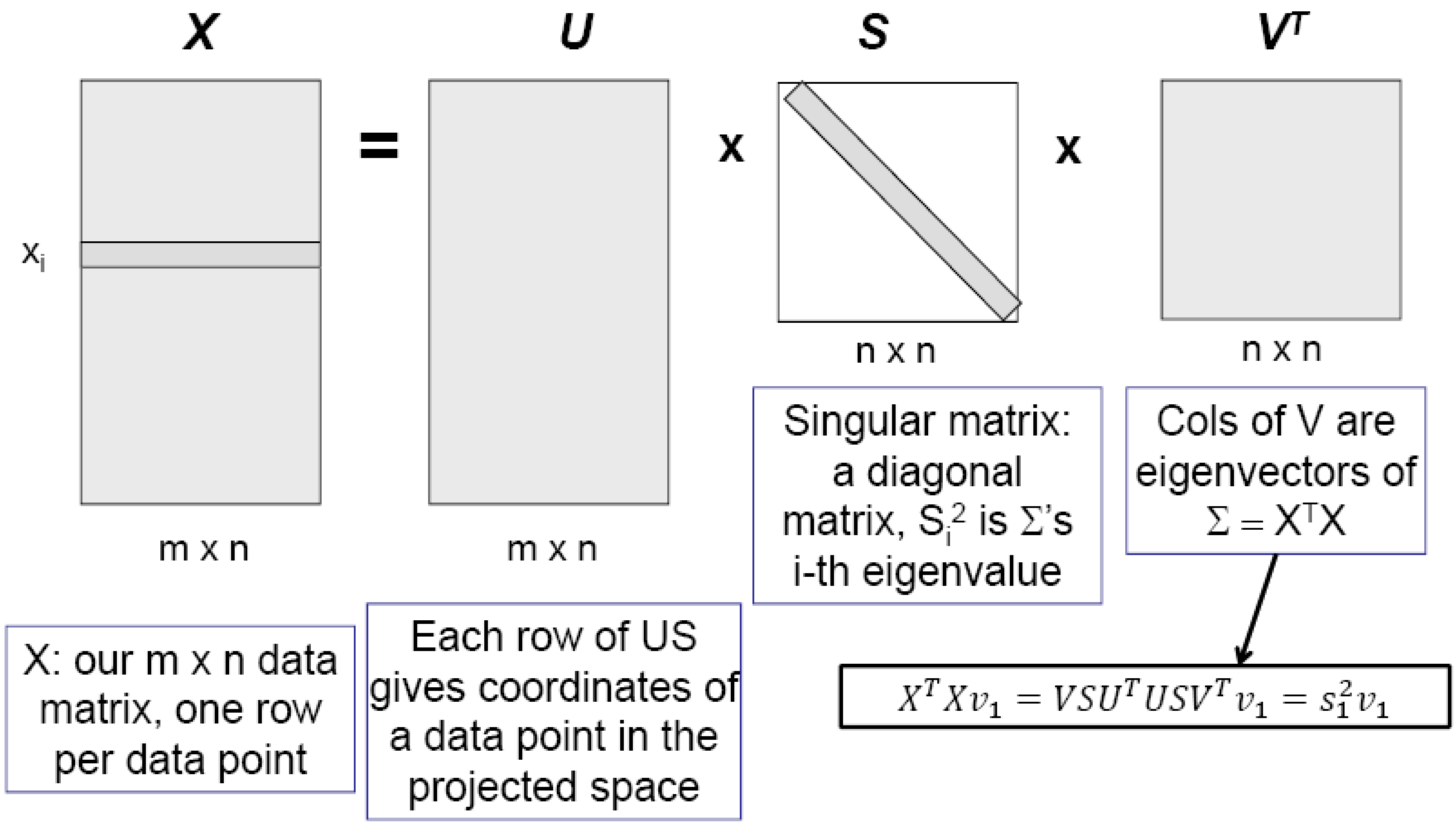
- Background changes cause problems
  - De-emphasize the outside of the face (e.g., by multiplying the input image by a 2D Gaussian window centered on the face).
- Light changes degrade performance
  - Light normalization might help but this is a challenging issue.
- Performance decreases quickly with changes to face size
  - Scale input image to multiple sizes.
  - Multi-scale eigenspaces.
- Performance decreases with changes to face orientation (but not as fast as with scale changes)
  - Out-of-plane rotations are more difficult to handle.
  - Multi-orientation eigenspaces.

Limitations (cont’d)

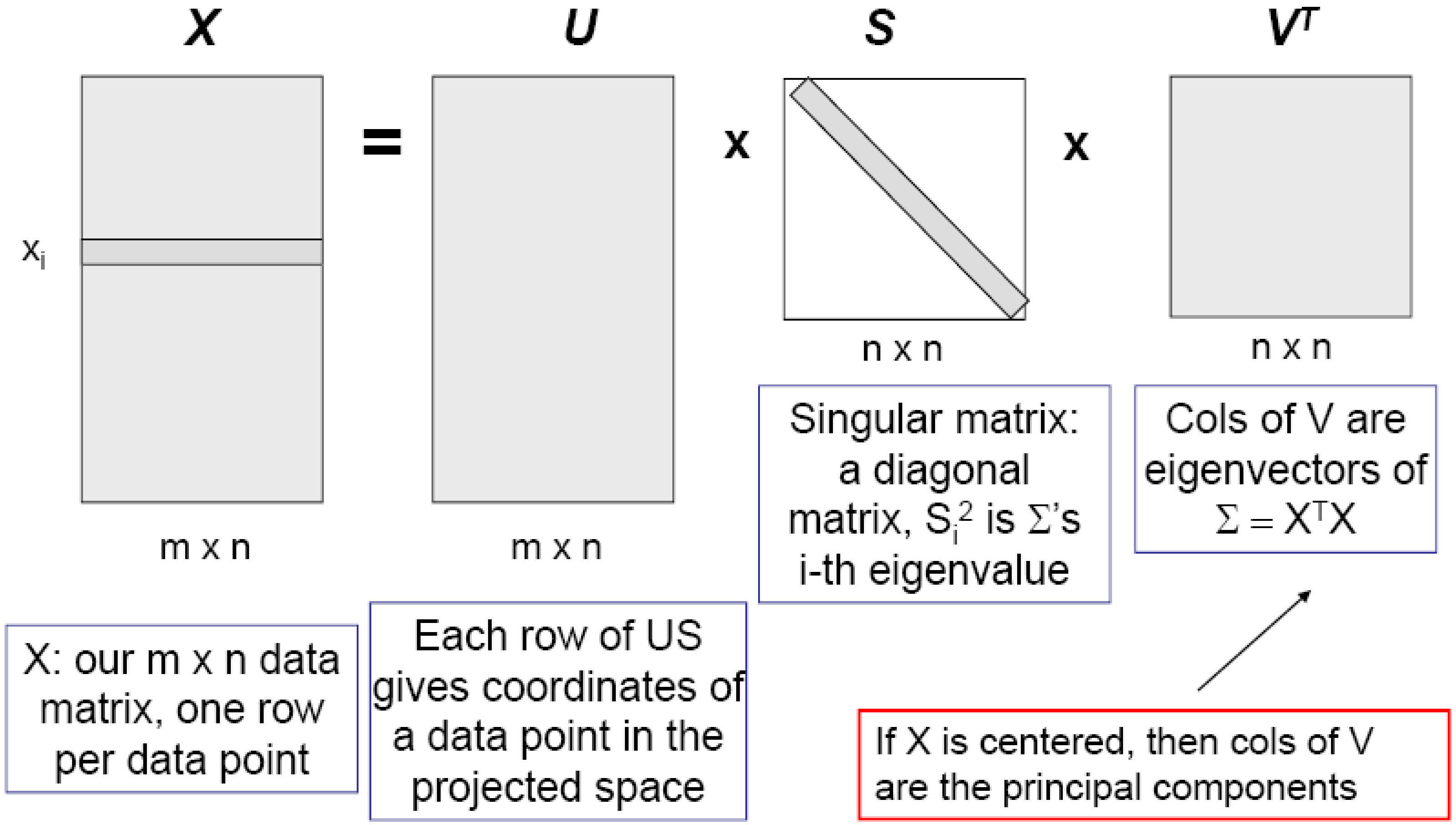
- Not robust to misalignment.

For non square matrix: Singular value decomposition (SVD)

$$\mathbf{X = U \cdot S \cdot V^T}$$



$$X = U \cdot S \cdot V^T$$



SVD for PCA

- Create mean-centered data matrix $\mathbf{X}$.
- Solve SVD: $\mathbf{X} = \mathbf{U} \cdot \mathbf{S} \cdot \mathbf{V}^T$.
- Columns of $\mathbf{V}$ are the eigenvectors of $\mathbf{\Sigma}$ sorted from largest to smallest eigenvalues.
- Select the first k columns as our k principal components.