

Feature Extraction

Computer vision tasks:

- Alignment
- Classification
- Recognition
- Detection
- Segmentation
- Prediction
- ...

For all these tasks, we need to determine the similarity among two images or image regions

Image similarity

Are these two images similar?

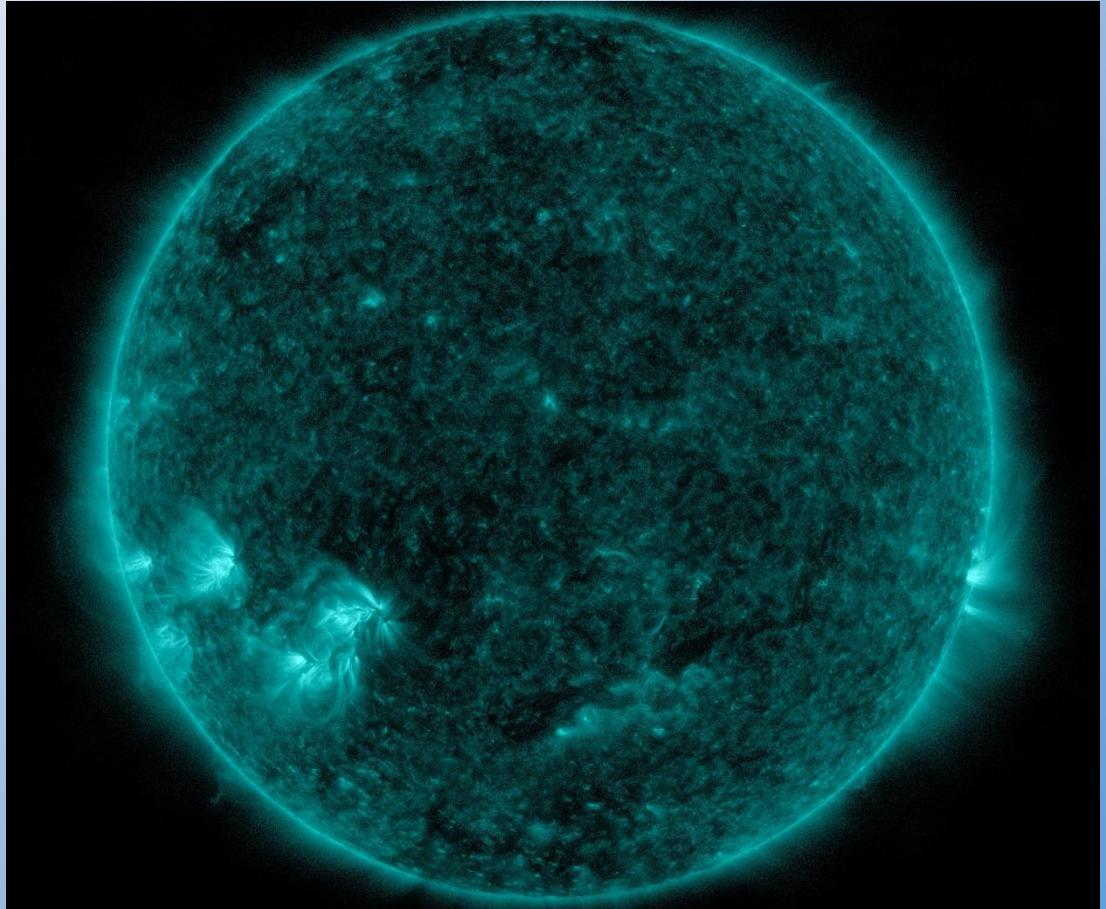
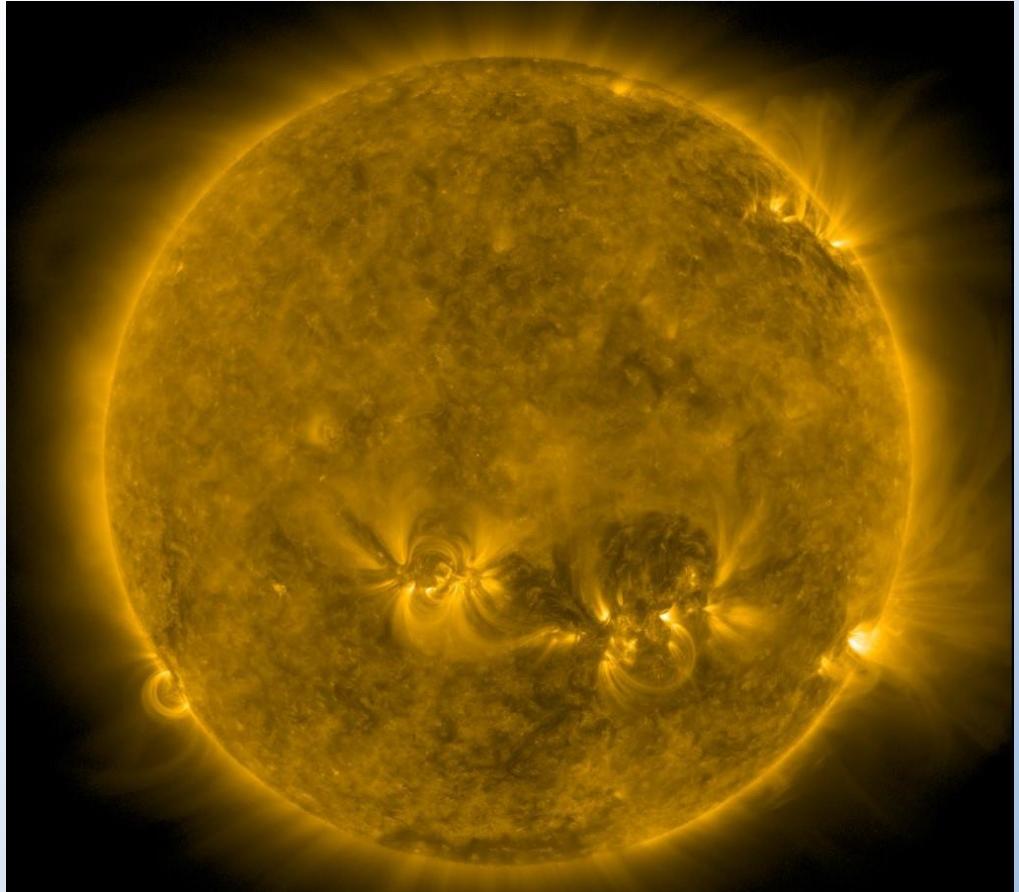


Image similarity

Are these two images similar?



Image similarity

Are these two images similar?



Image similarity

Are these two images similar?

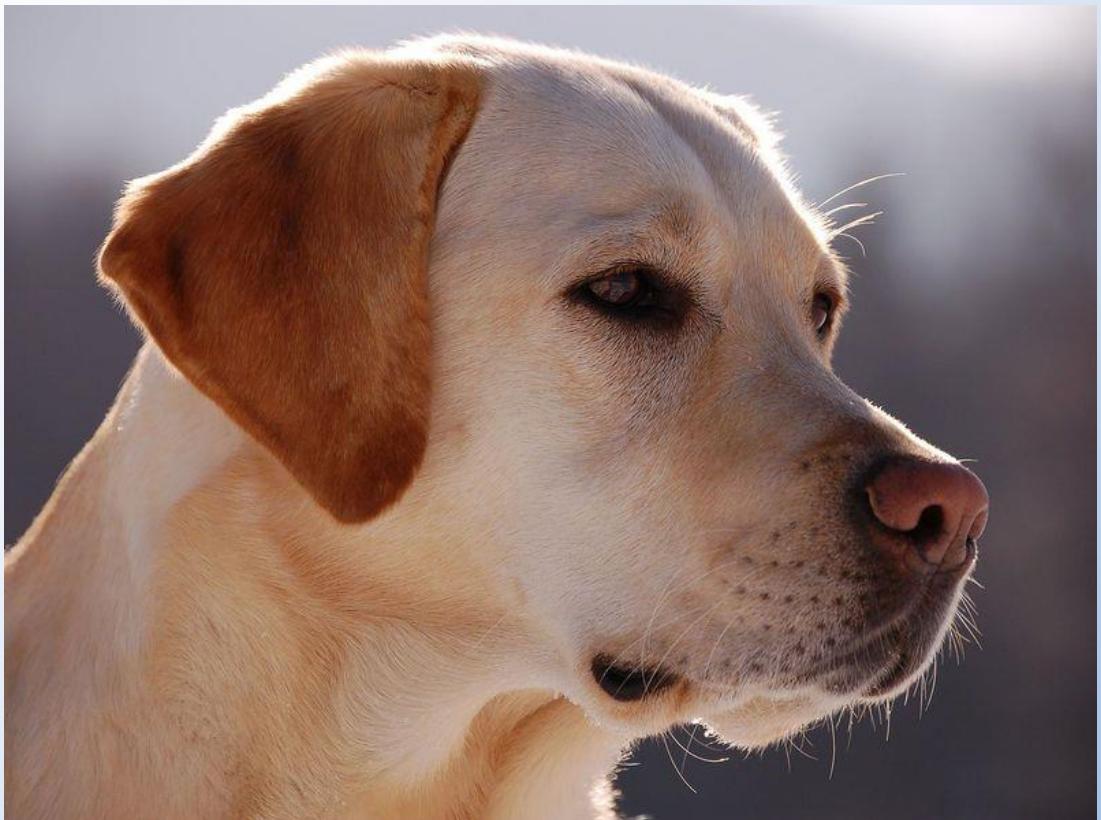


Image similarity

Are these two images similar?



Ideal Features:

- Ideally, we would like features to encode the concept of similarity as a human would evaluate it.
- Of course, this is very difficult.
- The choice of features is application dependent, but in general we want independence to:
 - Intensity transformations
 - Geometric transformations
 - Image cropping

Ideal Features:

- Depending on application, we may want features that either emphasize or deemphasize the following:
 - Texture
 - Lighting
 - Color
 - Shape
 - Higher-level information

Some Feature Sets

- Pixel intensities
- PCA
- Viola-Jones features
- Color histograms
- Histograms of gradients
- SIFT
- Visual words

Pixel intensities

- An m-by-n image is viewed as an object with $m \times n$ or $m \times n \times 3$ features
- Advantages:
 - Simplicity, features are directly available
- Disadvantages:
 - Only images of the same size can be compared (we need to crop or reshape)
 - High dimensional feature space – millions of features to describe a medium-sized image
 - Lack of generalization

Principal Component Analysis

Idea:

- View each image as a point in n dimensions (where n is the number of pixels in the image)
- Project data from n -dimensions (attributes) to m -dimensions (with $n > m$) while preserving as much information as possible

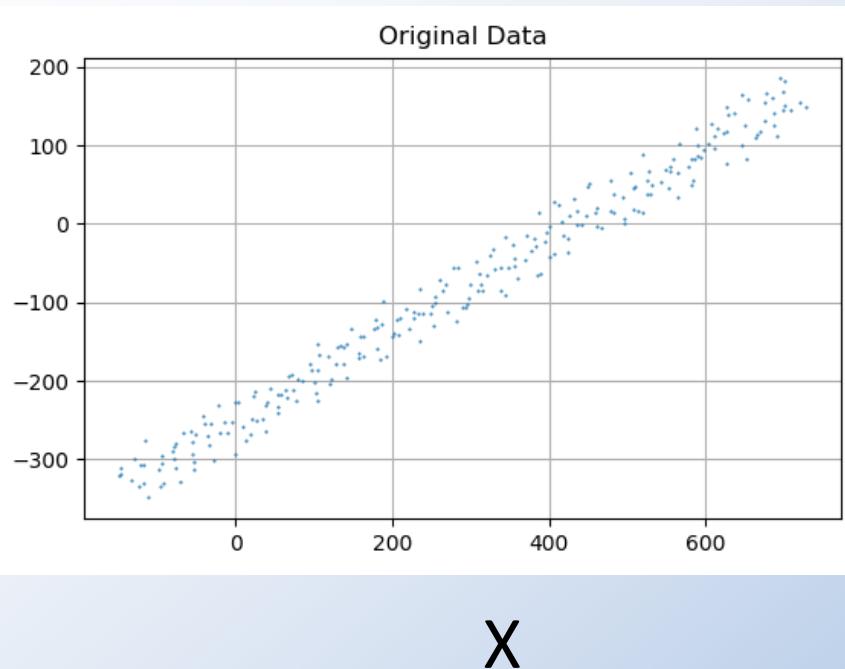
Why: Features tend to be correlated – reduce redundancy

Why: Least important features in projected space approximate noise

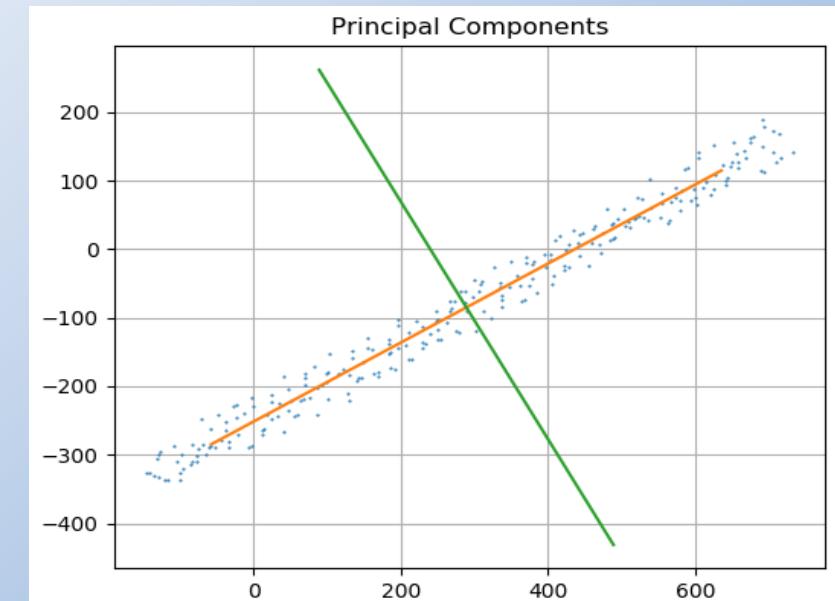
Mathematically, the principal components are the eigenvectors of the covariance matrix of the image set

Principal Component Analysis

The principal components of a dataset X are the vectors v_0, \dots, v_n such that v_i is the vector that best fits X and is perpendicular to each of v_0, \dots, v_{i-1}



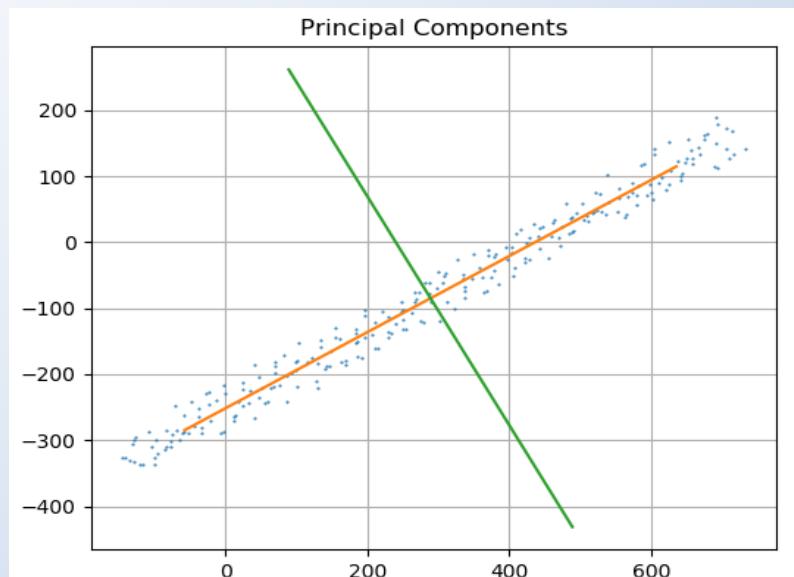
PCA
→



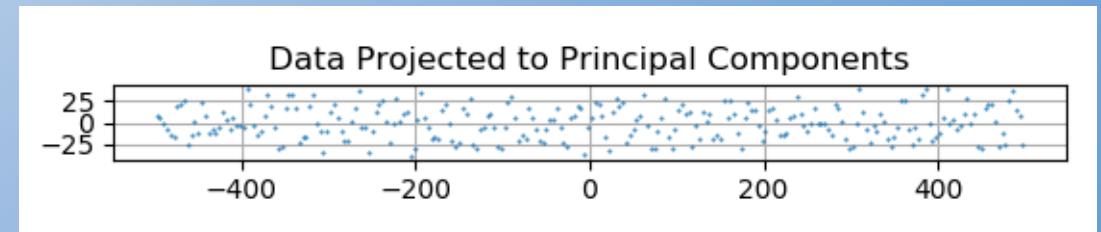
v_0 – orange, v_1 - green

Principal Component Analysis

The principal components of a dataset X are the vectors v_0, \dots, v_n such that v_i is the vector that best fits X and is perpendicular to each of v_0, \dots, v_{i-1}

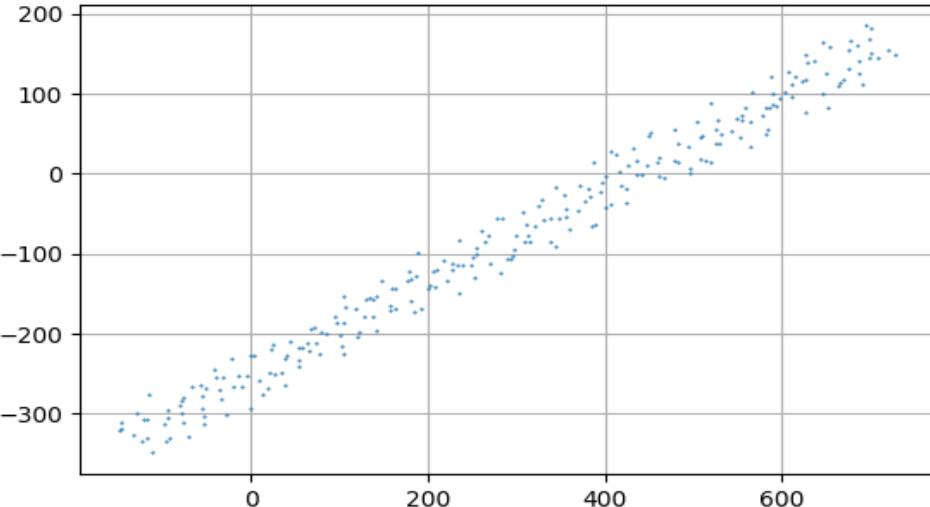


X in original feature space and principal components

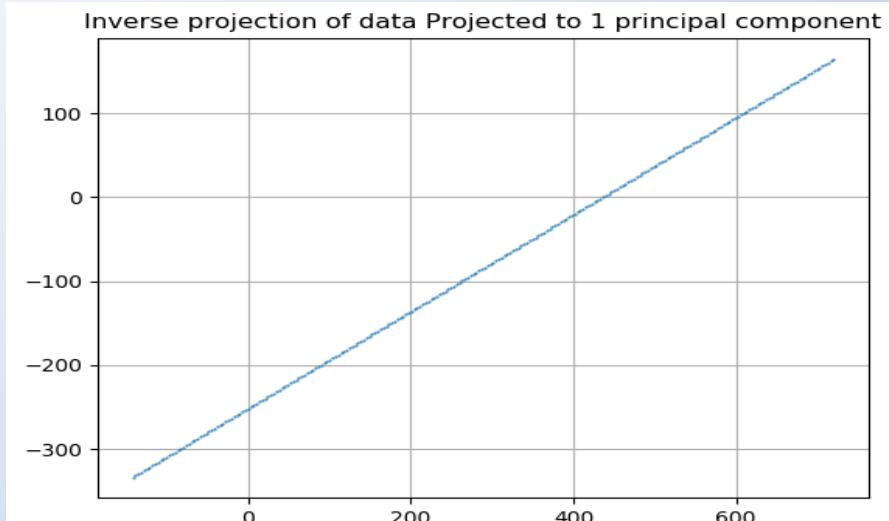
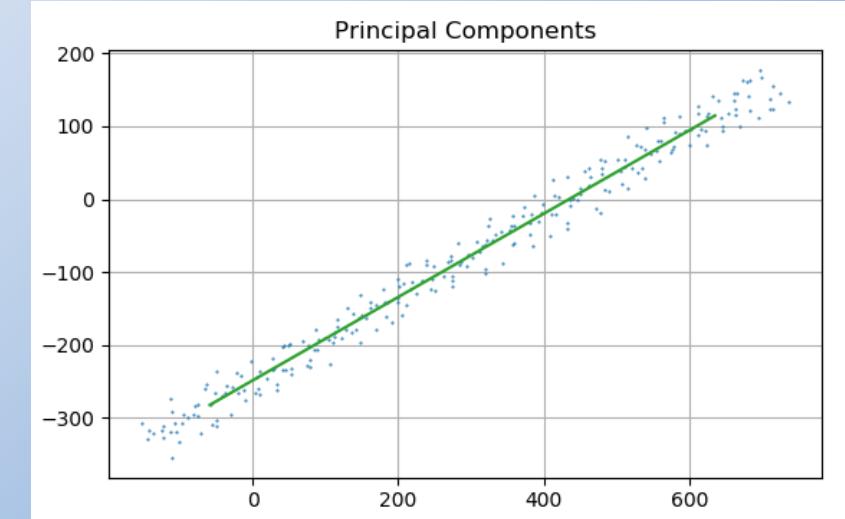


X in new feature space
Features with smallest variance can be discarded

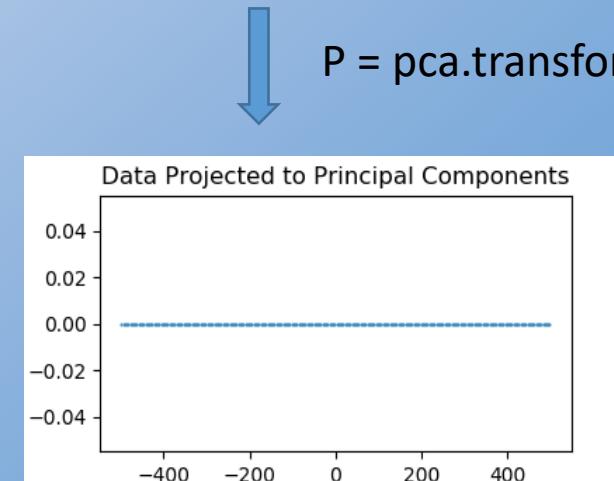
Principal Component Analysis



```
pca.fit(X,n_components=1)
```



```
Xf =  
pca.inverse_transform(P)
```



Principal Component Analysis

Let $I = \{I_1, \dots, I_n\}$ be the image set, where each images is reshaped into a vector

Find the set of eigenvectors of the covariance matrix of $I \{E_1, \dots, E_m\}$ (with $m < n$)

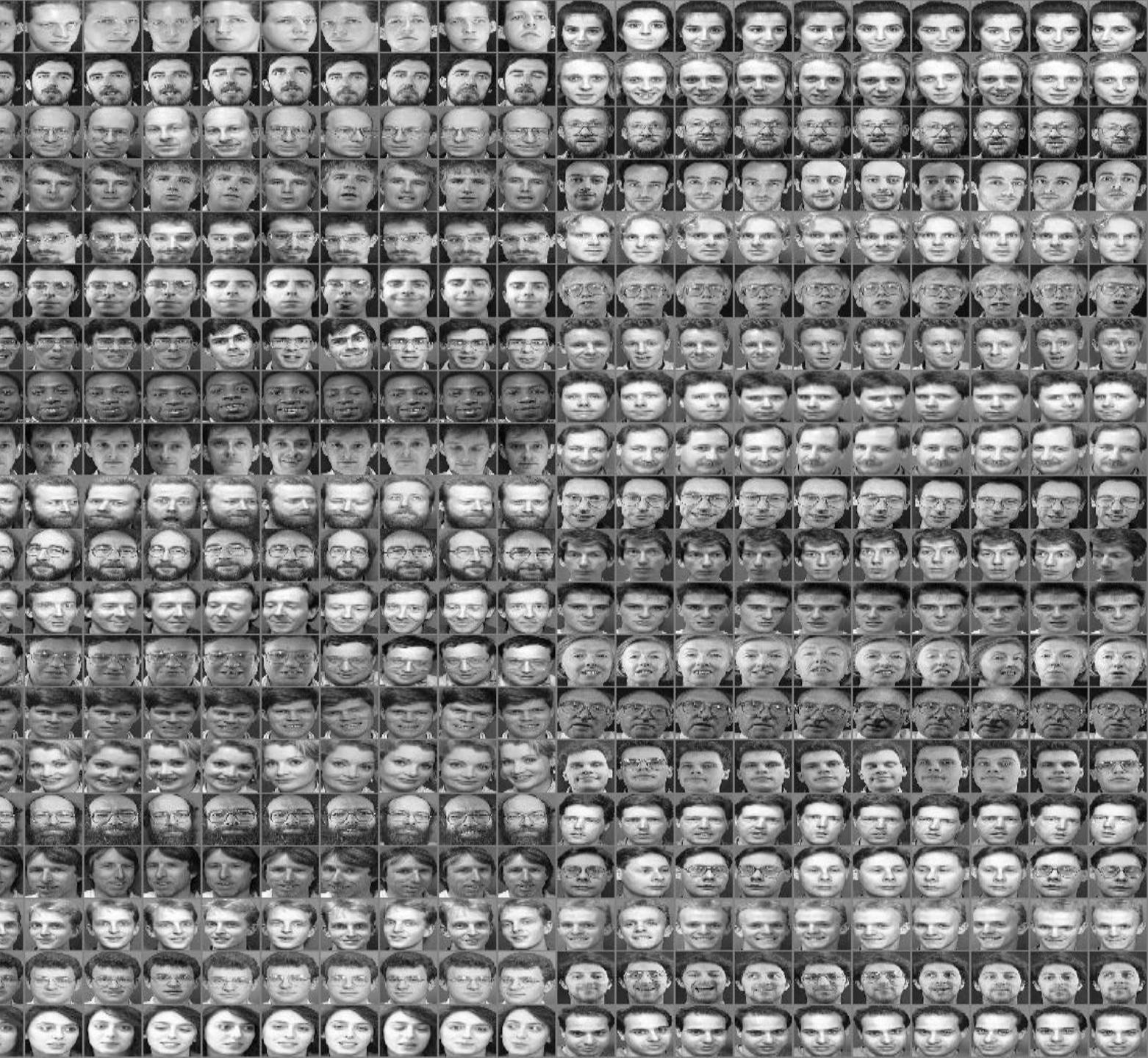
Then each image can be described as:

$$I_j = c_1 * E_1 + c_2 * E_2 + \dots + c_m * E_m + \epsilon$$

The we use the vector $[c_1, c_2, \dots, c_m]$ as the features of describing I_j

PCA Example: Eigenfaces

Original data



PCA Example: Eigenfaces

Principal
components:



Principal Component Analysis

Advantages:

- Global rather than local descriptor
- Representation has well-understood mathematical properties

Disadvantages:

- Computation is time-consuming; unfeasible for large images
- Sensitive to lighting effects

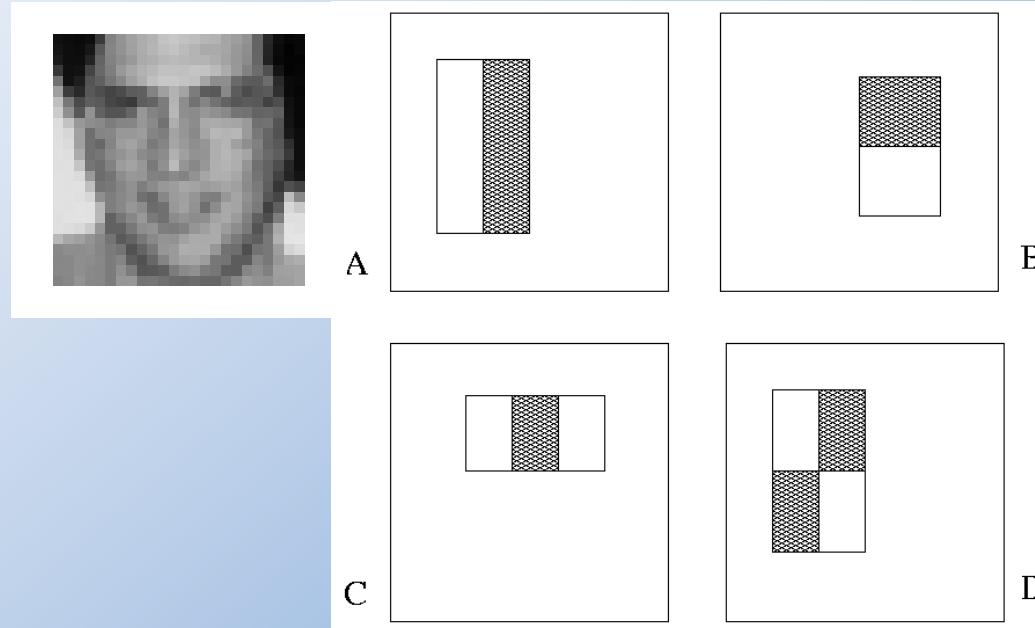
Principal Component Analysis

Final note:

We will give an F in the class to anybody who refers to this algorithm as 'Principle Components'.

Viola-Jones features

“Rectangle filters”



Value =

$$\sum (\text{pixels in white area}) - \sum (\text{pixels in black area})$$

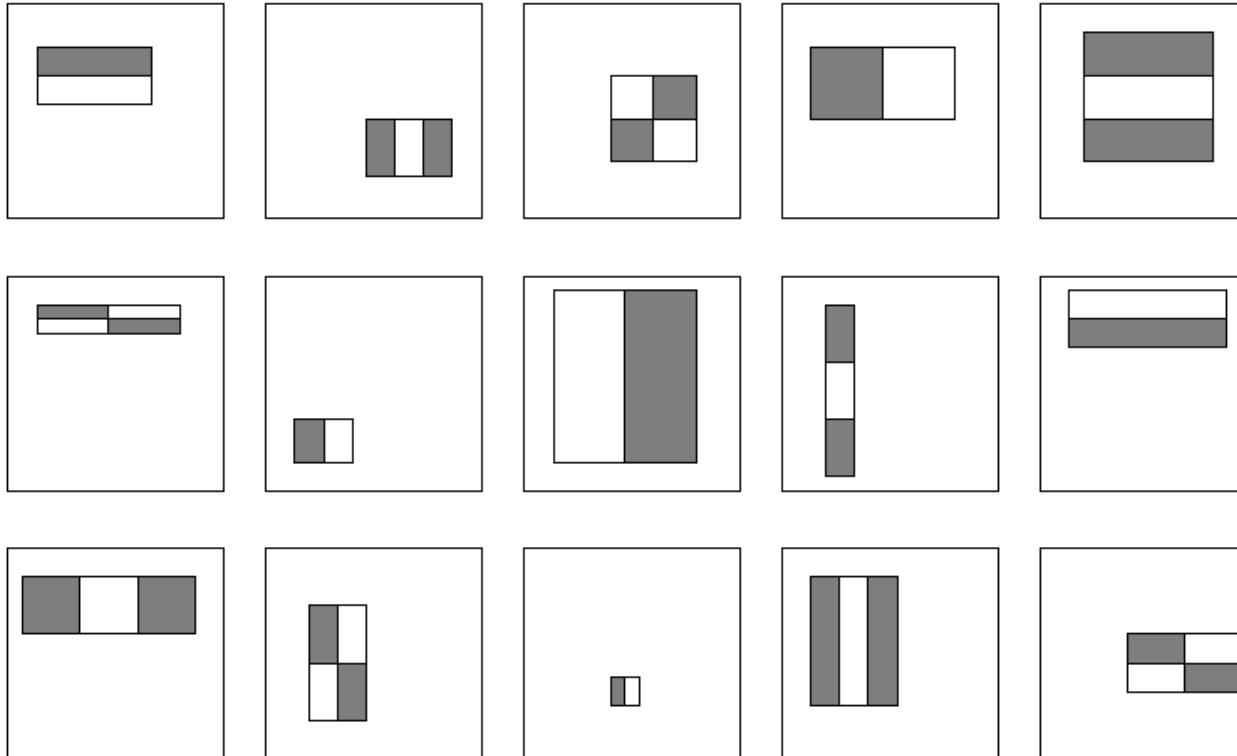
Computed efficiently using the integral image trick

Viola-Jones features

- First successful approach to object detection
(Viola and Jones, 2001)
- Designed to work on gray-level images
- Still used today (OpenCV face detector is based
on these features)

Viola-Jones features

- For a 24x24 detection region, the number of possible rectangle features is ~160,000!



Viola-Jones features

- For a 24x24 detection region, the number of possible rectangle features is around 160,000
- At test time, it is impractical to evaluate the entire feature set
- We can create a good classifier using just a small subset of all possible features

Viola-Jones features

Advantages:

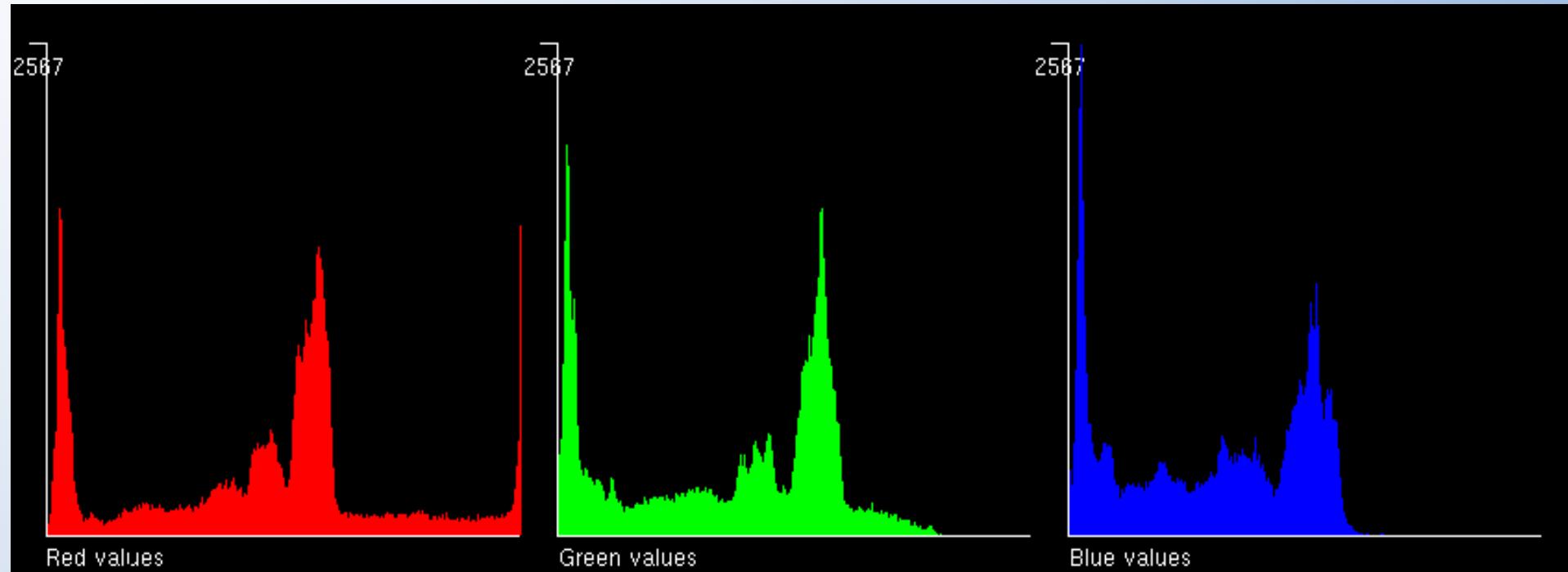
- Can be computed efficiently

Disadvantages:

- Requires a feature selection step, as the number of possible features is huge
- Tailored to grey-level images

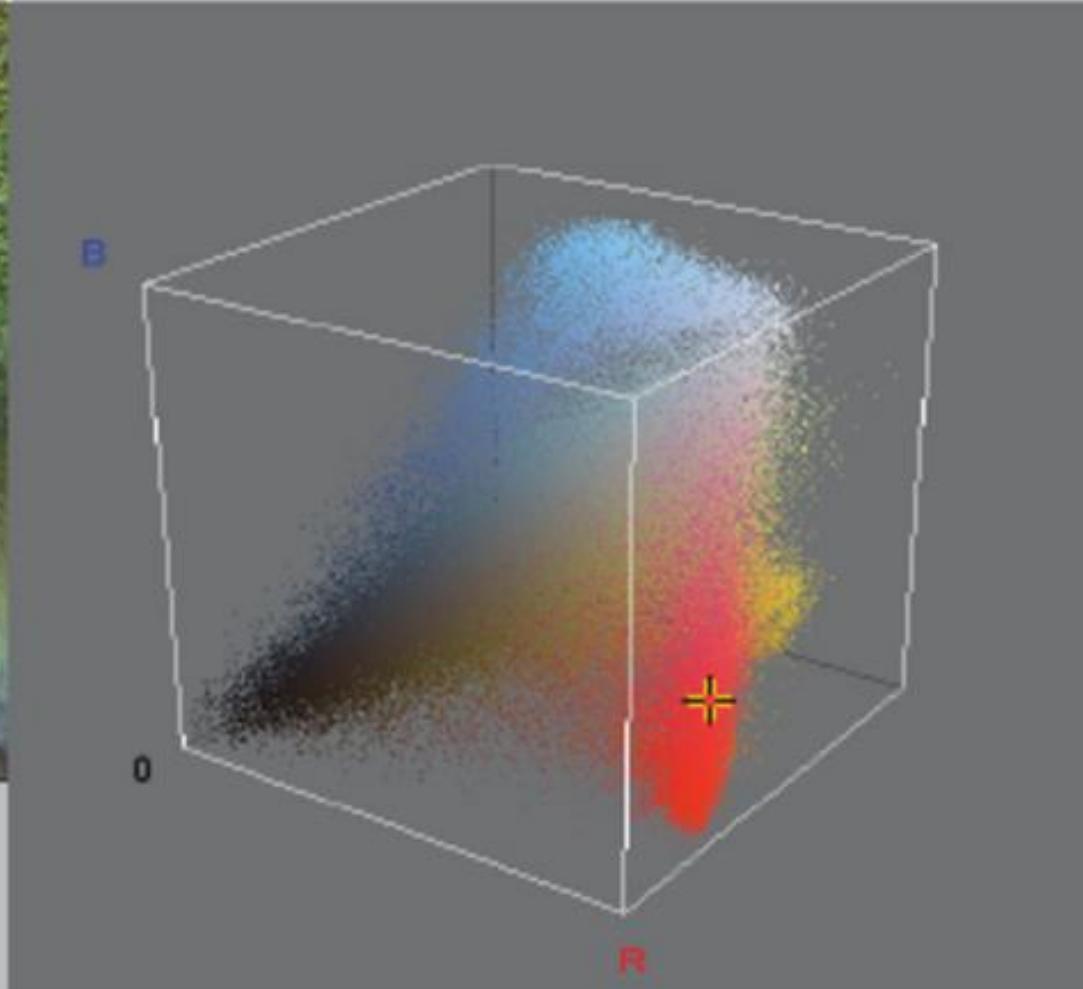
Color Histograms

- Color histograms – Three 1-D descriptors

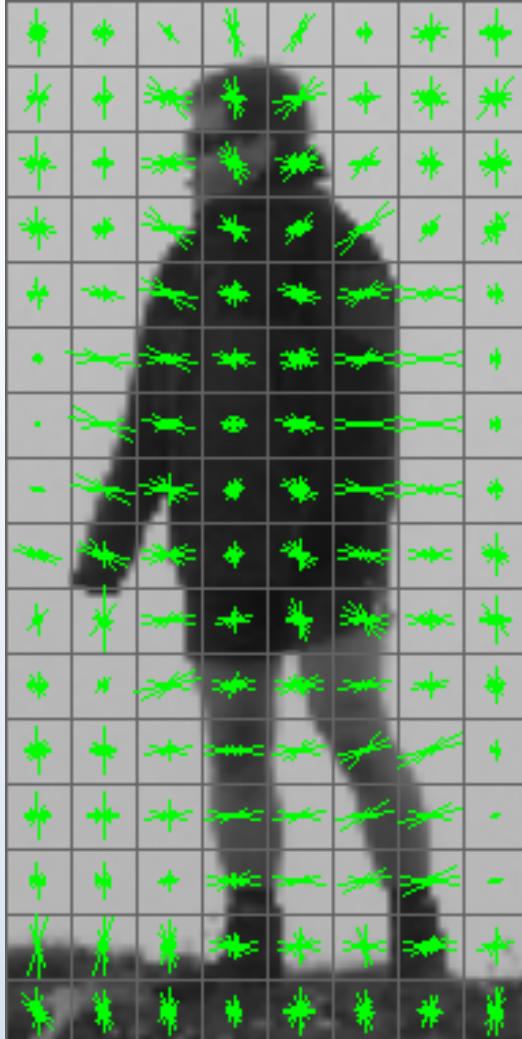


Color Histograms

- Color histograms – 3-D descriptor



Histograms of gradients (HOGs)



Find vertical and horizontal gradients in image g_v and g_h

Compute gradient magnitude

$$g_m = \sqrt{(g_v)^2 + (g_h)^2}$$

Compute gradient direction

$$g_d = \text{atan2}(g_v, g_h)$$

Split the $[-180, 180^\circ]$ range of possible gradient directions into n equally size intervals $[i_0, i_{n-1}]$

For every region, generate a descriptor vector $[h_0, h_{n-1}]$
where:

$$h_i = \sum g_m[r, c] \text{ if } g_d[r, c] \text{ in } i_i \text{ for all pixels } [r, c] \text{ in region}$$