CS 556: Computer Vision

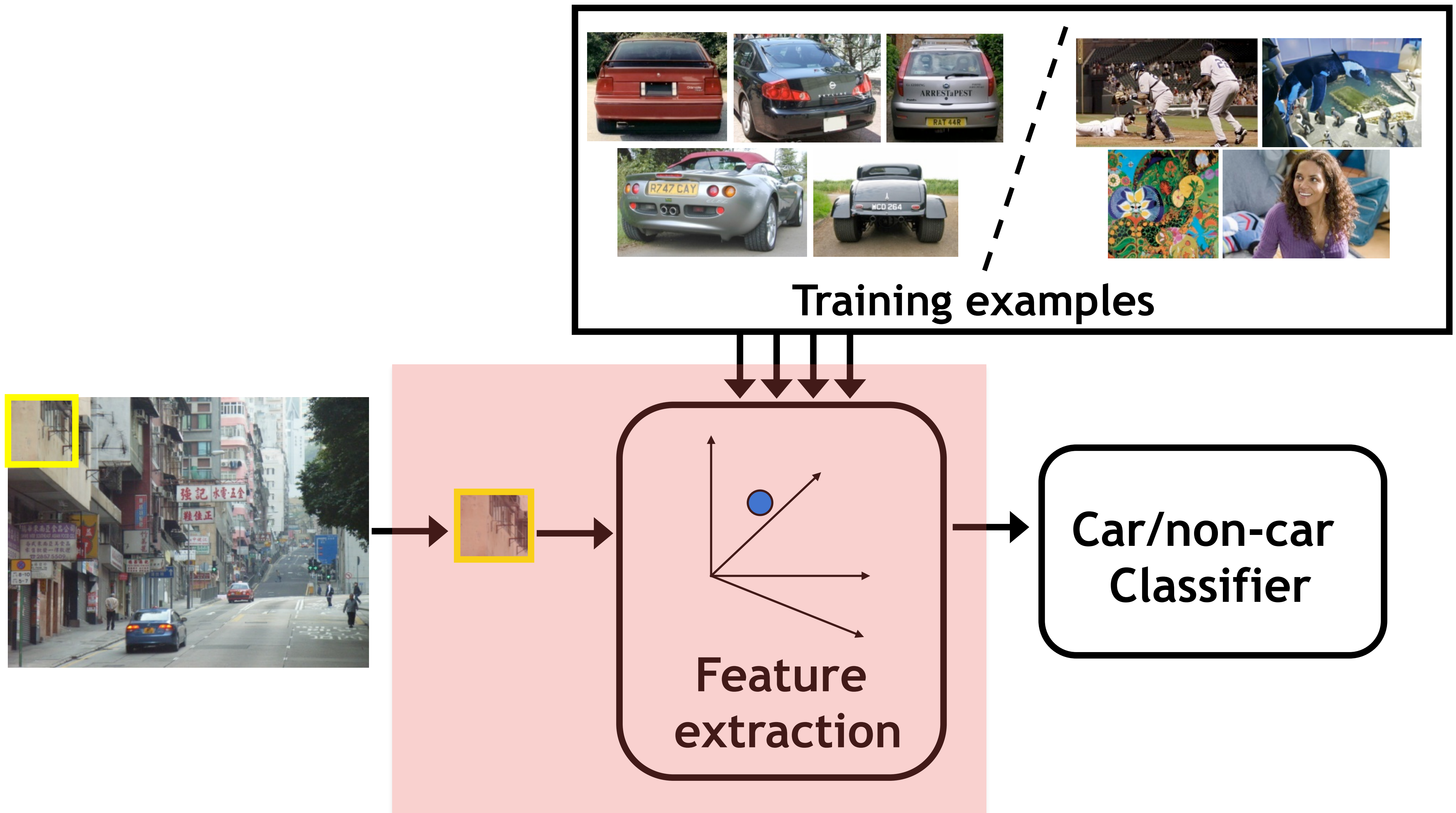
Lecture 9

Prof. Sinisa Todorovic

sinisa@eecs.oregonstate.edu

Oregon State University

Object Recognition and Image Classification

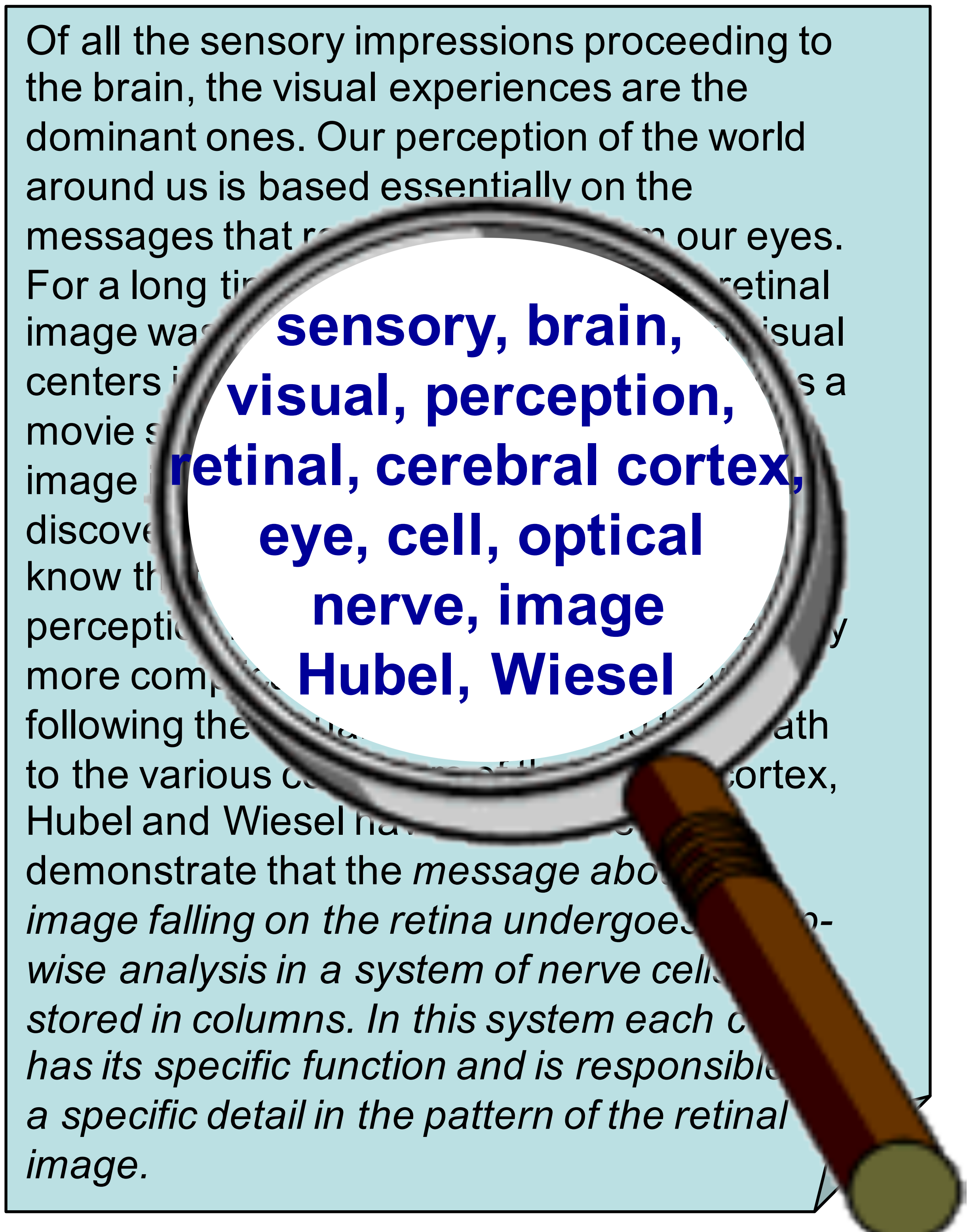


Bag-of-Words

Image → Bag of ‘words’

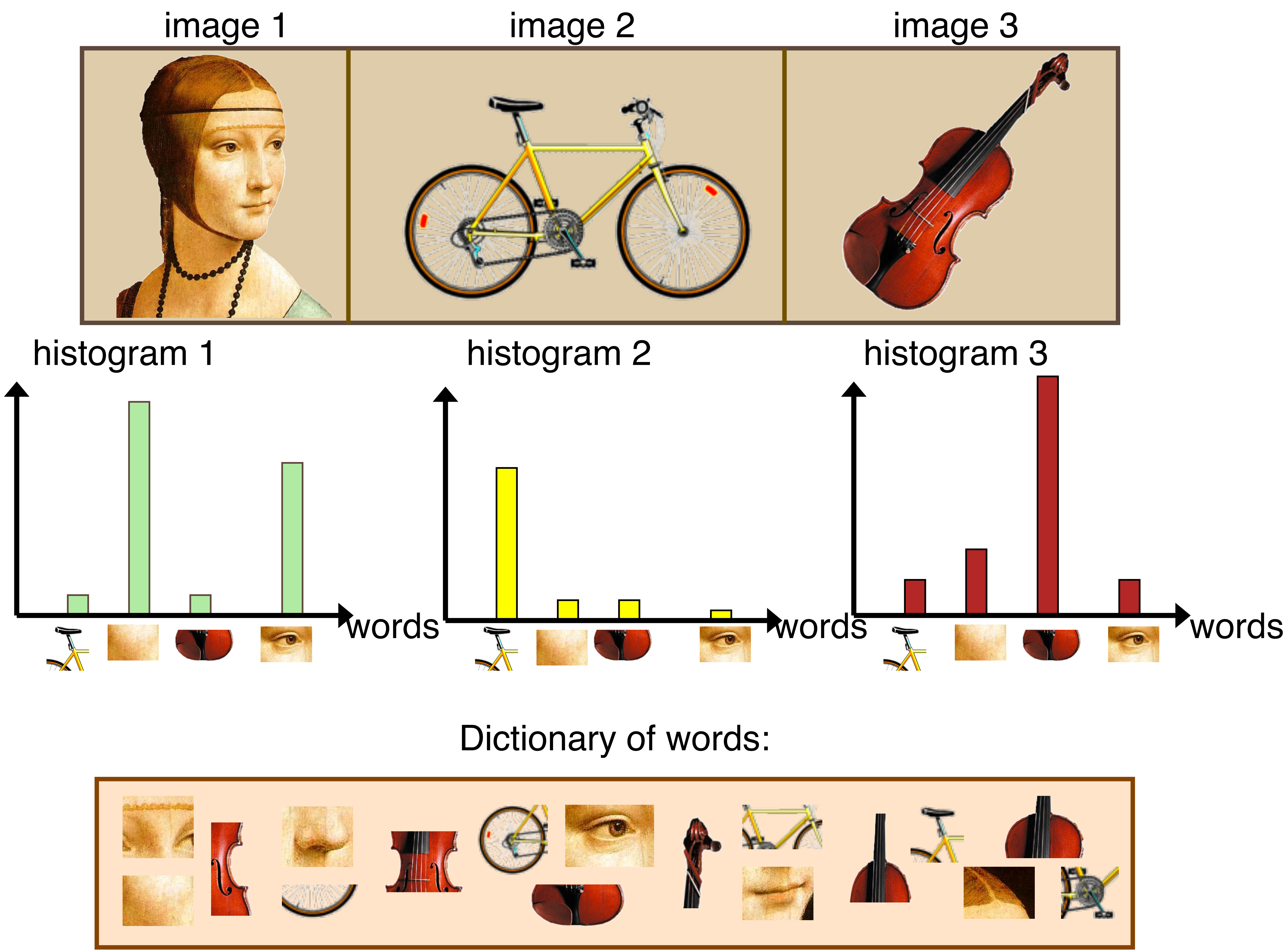
by Fei-Fei Li (Stanford)

Analogy to Documents



by Fei-Fei Li (Stanford)

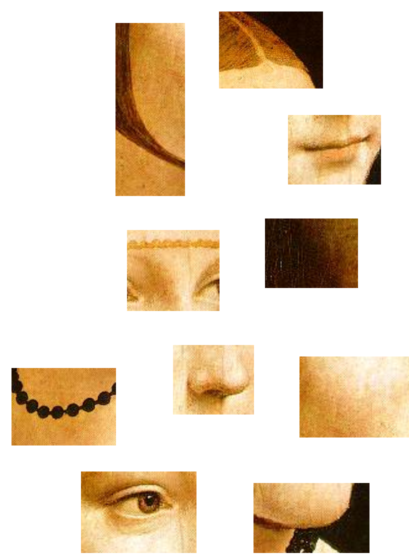
Image = Histogram of Words



by Fei-Fei Li (Stanford)

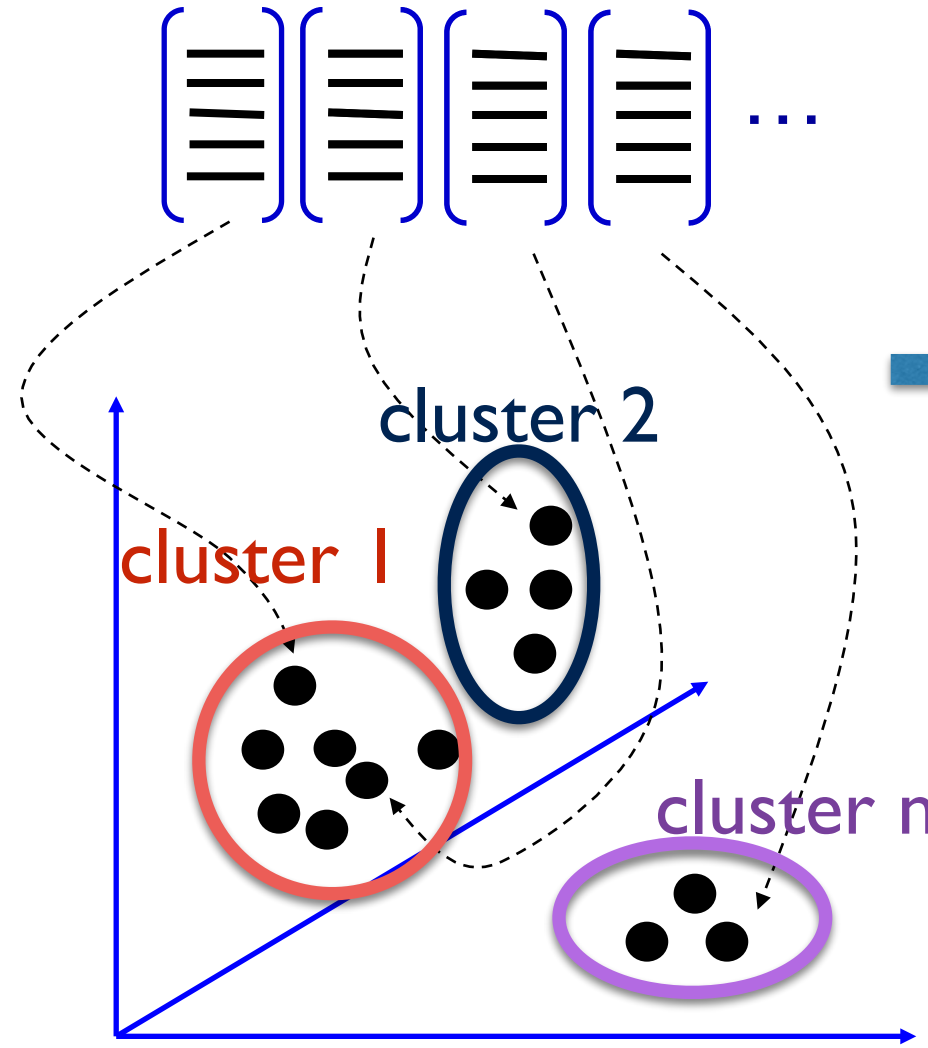
Computing BoW

image



extract
interest points
+
compute their
features

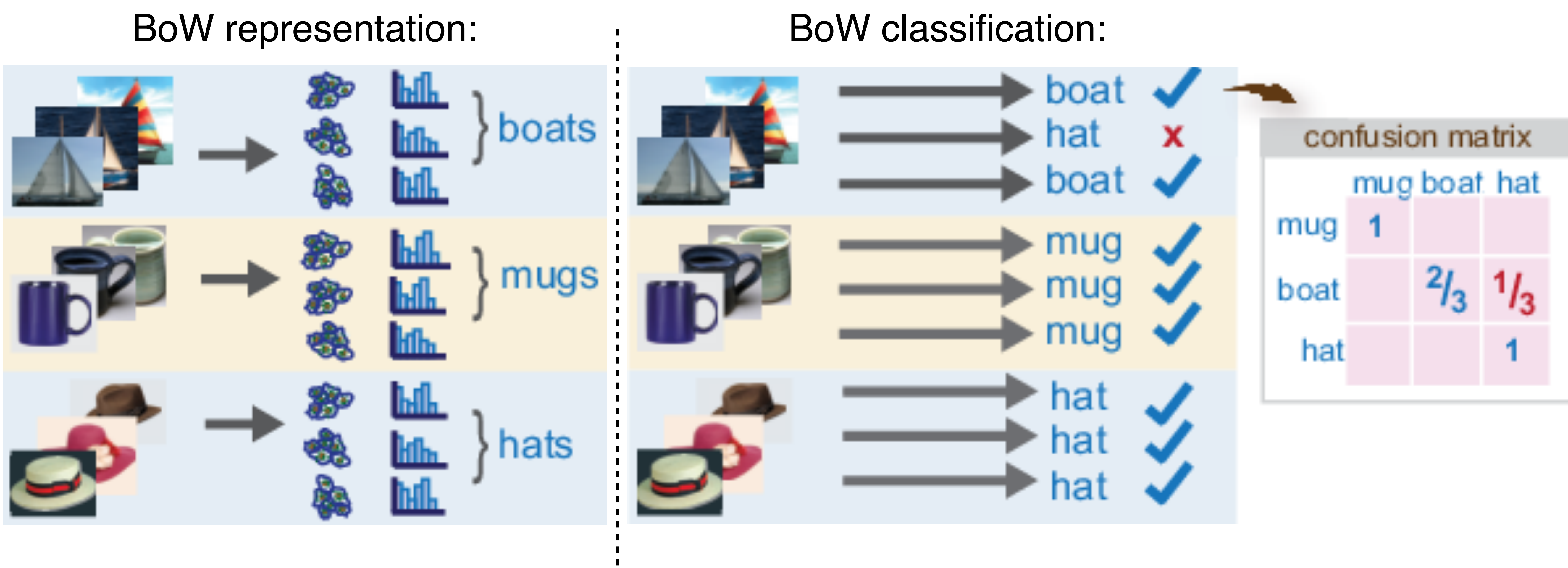
features



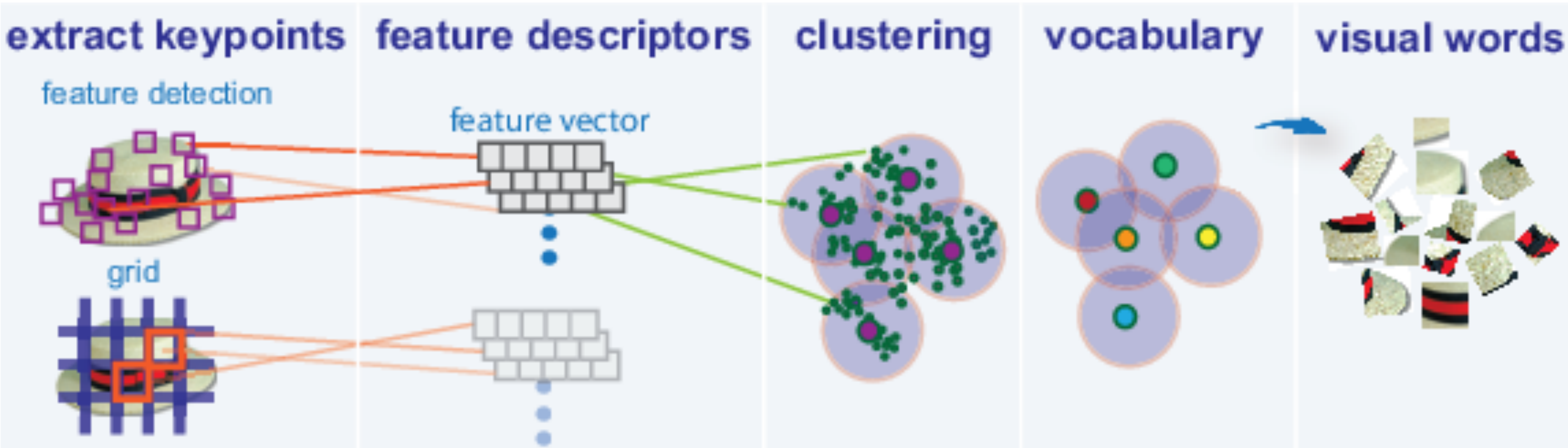
feature encoding

BoW

Image Classification using BoW



Computing a Dictionary of Words



words = centers of clusters

Clustering

- A good clustering method will produce clusters with
  - High intra-class similarity
  - Low inter-class similarity
- Similarity definition is application-dependent

K-Means

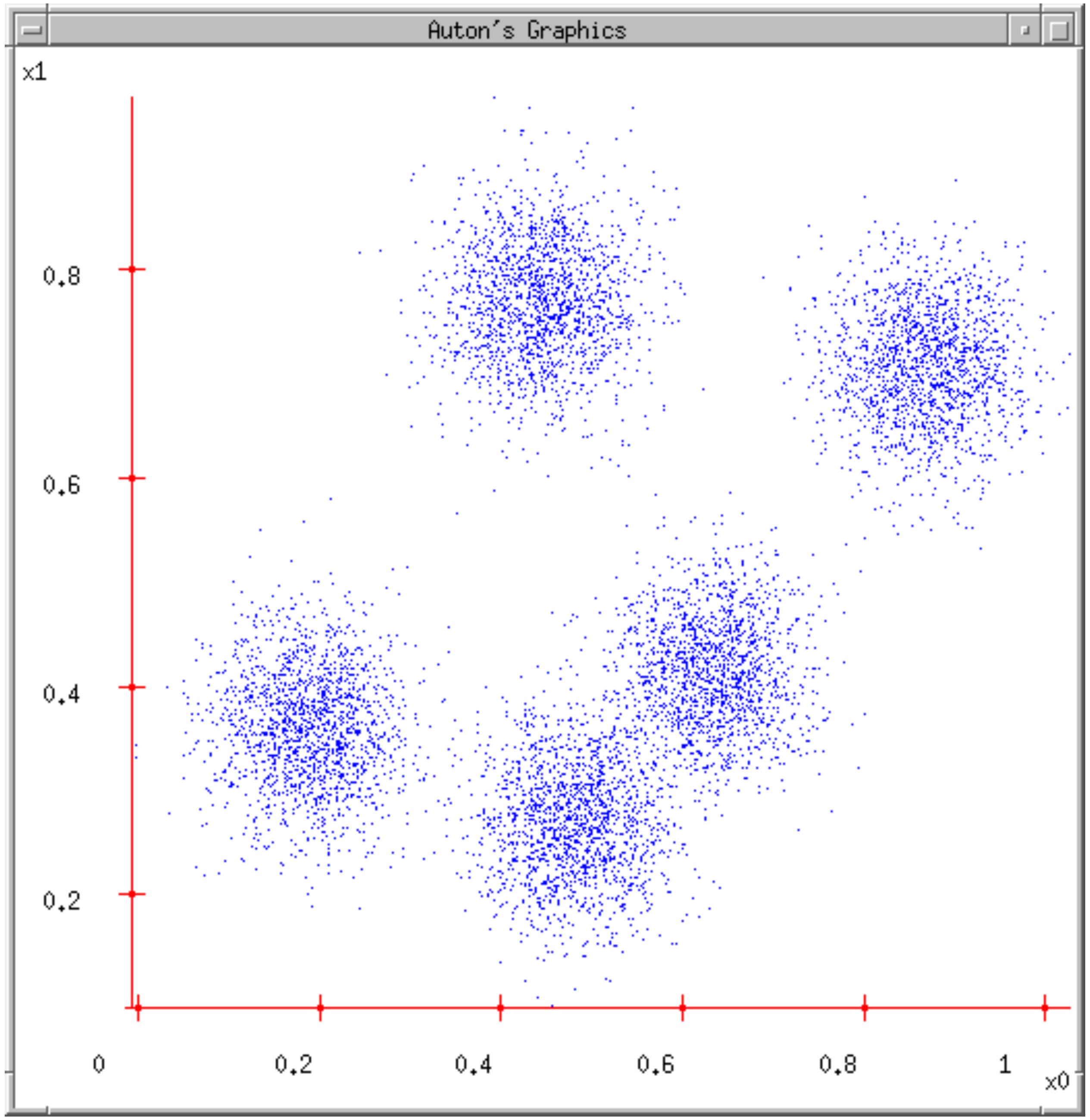
Given K, the K-means algorithm consists of four steps:

1. Select initial centroids at random.
2. Assign each point to the cluster with the nearest centroid.
3. Compute each centroid as the mean of the points assigned to it.
4. Repeat previous 2 steps until no change.

K-Means: Example

1. Set K = number of clusters

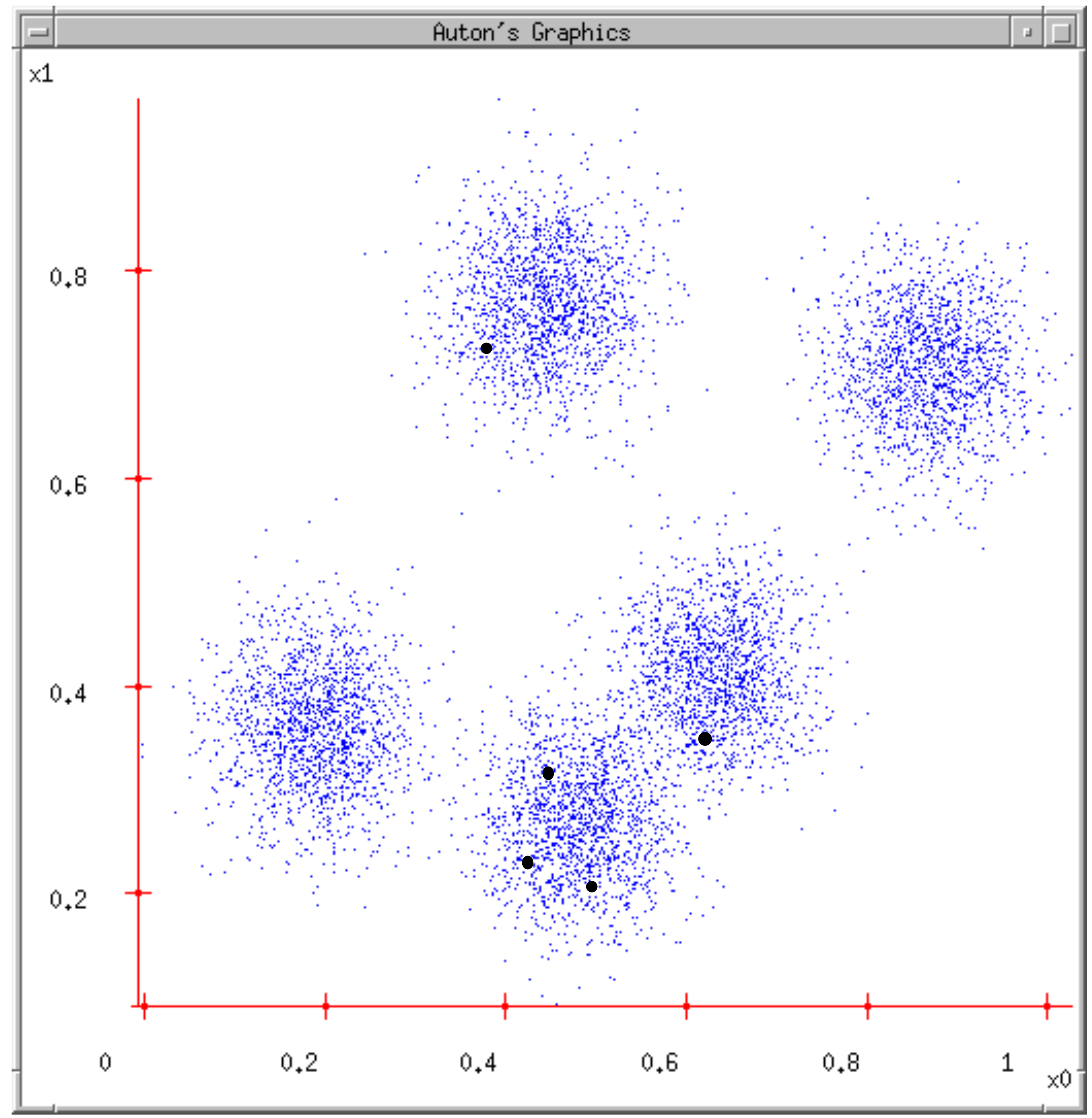
points to be clustered using K-means



K-Means: Example

1. Set K = number of clusters

2. Randomly guess K cluster centers

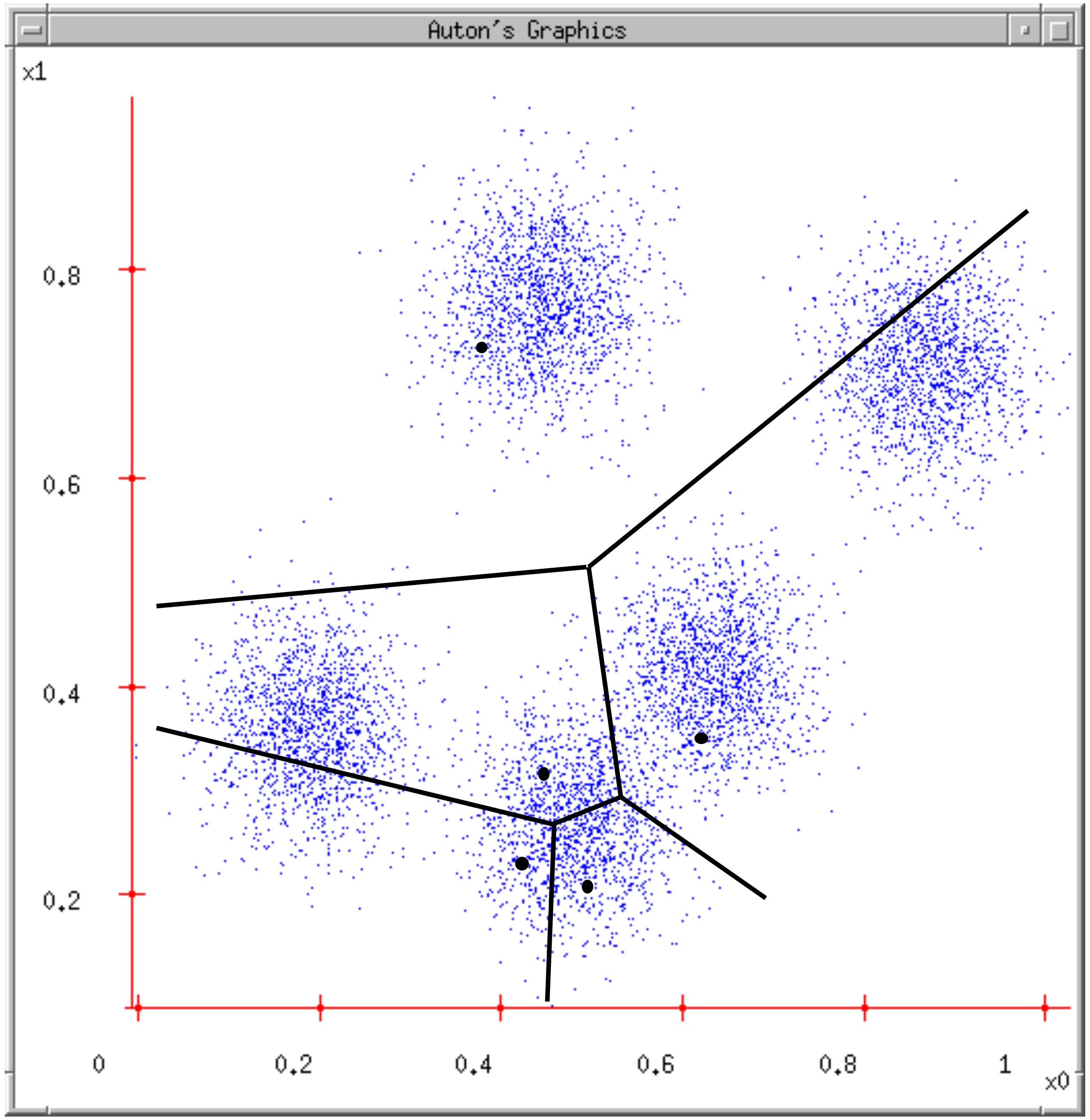


K-Means: Example

1. Set K = number of clusters

2. Randomly guess K cluster centers

3. Assign each point to the closest center, and thus form clusters



K-Means: Example

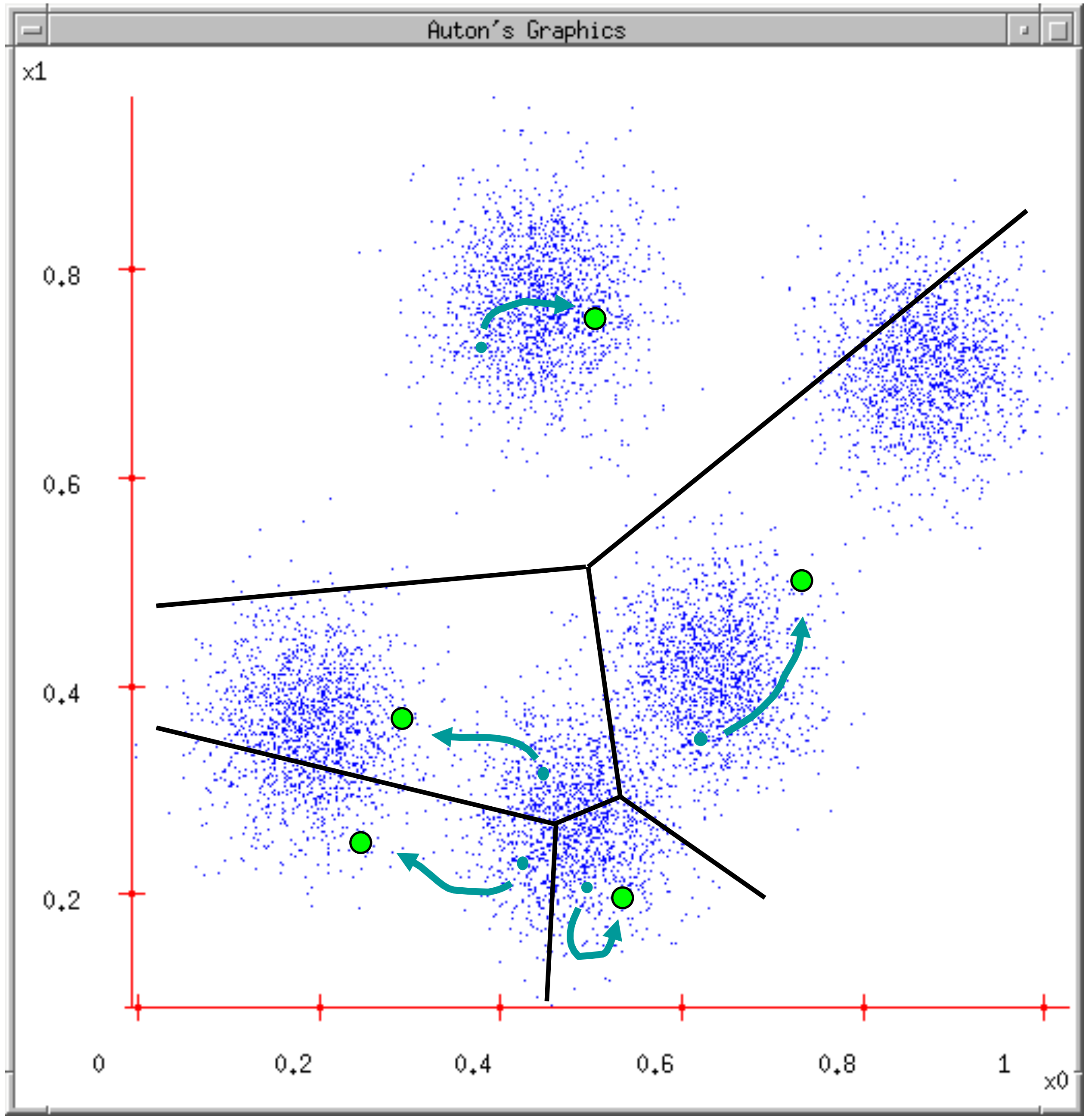
1. Set K = number of clusters

2. Randomly guess K cluster centers

3. Assign each point to the closest center, and thus form clusters

4. Update the cluster centers

5. Repeat Steps 3 and 4 until convergence



K-Means — MATLAB

```
>> K = 3;  % number of clusters
>> idx = kmeans(X,K,'Distance','cityblock');
```

cluster assignment of each feature (→ idx) rows = features (→ X)

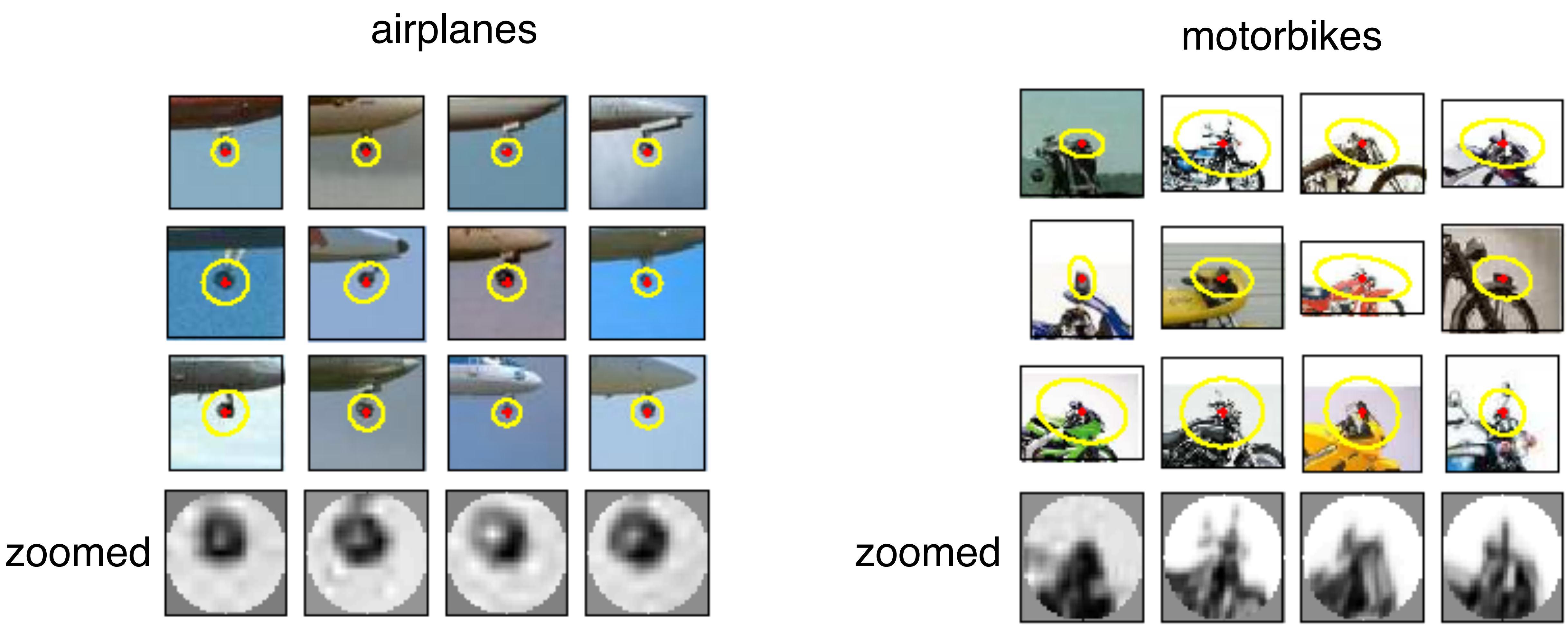
▾ `'Distance'` — Distance measure
`'sqeuclidean'` (default) | `'cityblock'` | `'cosine'` | `'correlation'` | `'hamming'`

Distance measure, in p-dimensional space, used for minimization, specified as the comma-separated pair consisting of `'Distance'` and a string.

`kmeans` computes centroid clusters differently for the different, supported distance measures. This table summarizes the available distance measures. In the formulae, *x* is an observation (that is, a row of X) and *c* is a centroid (a row vector).

Distance Measure	Description	Formula
`'sqeuclidean'`	Squared Euclidean distance (default). Each centroid is the mean of the points in that cluster.	$d(x,c) = (x-c)(x-c)'$
`'cityblock'`	Sum of absolute differences, i.e., the *L*1 distance. Each centroid is the component-wise median of the points in that cluster.	$d(x,c) = \sum_{j=1}^{p} \lvert x_j - c_j \rvert$
`'cosine'`	One minus the cosine of the included angle between points (treated as vectors). Each centroid is the mean of the points in that cluster, after normalizing those points to unit Euclidean length.	$d(x,c) = 1 - \frac{xc'}{\sqrt{(xx')(cc')}}$
`'correlation'`	One minus the sample correlation between points (treated as sequences of values). Each centroid is the component-wise mean of the points in that cluster, after centering and normalizing those points to zero mean and unit standard deviation.	$d(x,c) = 1 - \frac{(x-\vec{\bar{x}})(c-\vec{\bar{c}})'}{\sqrt{(x-\vec{\bar{x}})(x-\vec{\bar{x}})'}\sqrt{(c-\vec{\bar{c}})(c-\vec{\bar{c}})'}}$, where • $\vec{\bar{x}} = \frac{1}{p}\left(\sum_{j=1}^{p} x_j\right)\vec{1}_p$ • $\vec{\bar{c}} = \frac{1}{p}\left(\sum_{j=1}^{p} c_j\right)\vec{1}_p$ • $\vec{1}_p$ is a row vector of *p* ones.
`'hamming'`	This measure is only suitable for binary data. It is the proportion of bits that differ. Each centroid is the component-wise median of points in that cluster.	$d(x,y) = \frac{1}{p}\sum_{j=1}^{p} I\{x_j \neq y_j\}$, where *I* is the indicator function.

Example Clusters of SIFT Points



Sivic et al. 2005

Image Representation as BoW

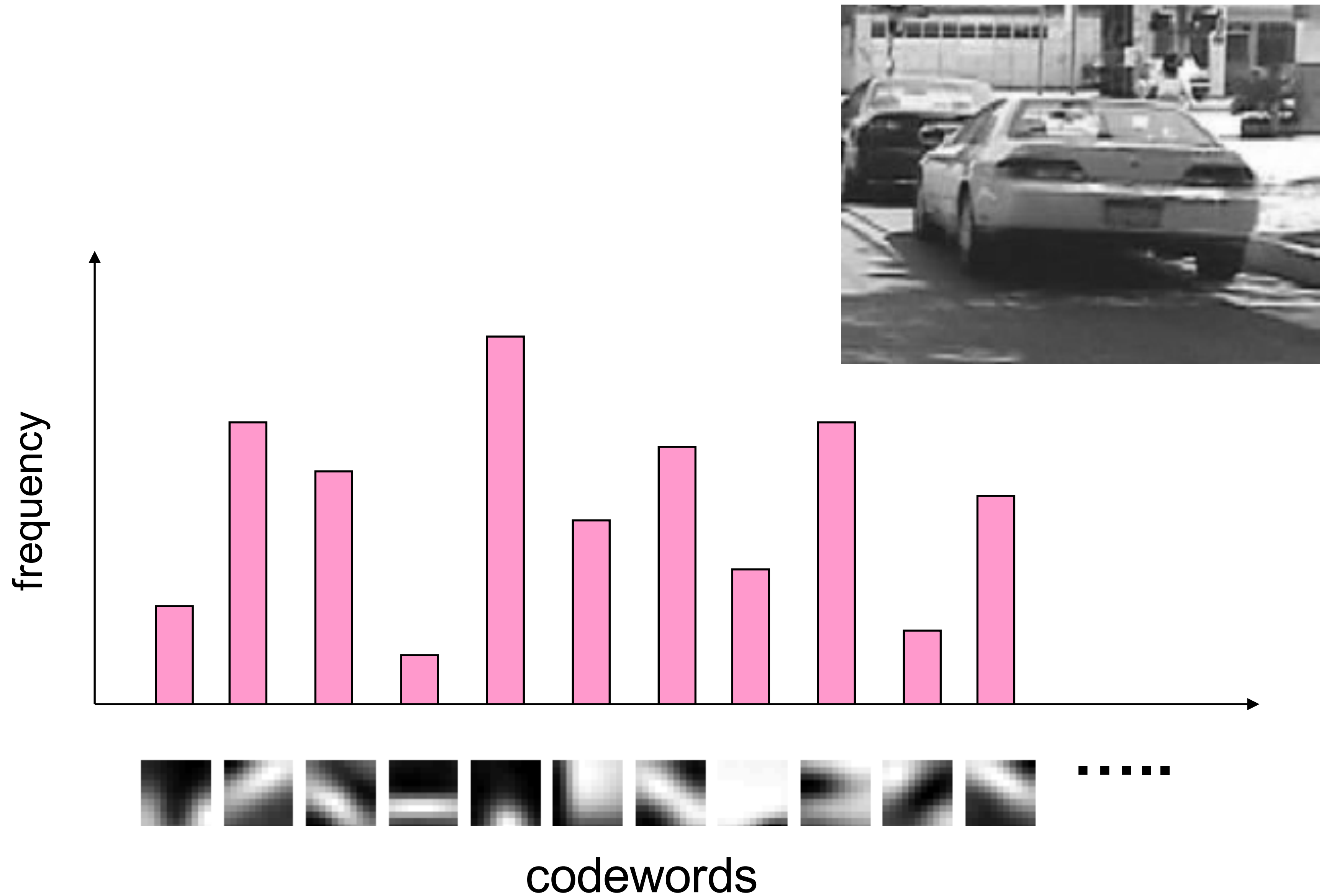


Image Representation as BoW — MATLAB

Images are in a folder organized in 5 subfolders. Each subfolder has 2 images.

```
>> setDir = fullfile('/Users/Desktop/courses/CS556/HW')

setDir =

/Users/Desktop/courses/CS556/HW

>> imgSet = imageSet(setDir, 'recursive')

imgSet =

  1x5 imageSet array with properties:

    Description
    ImageLocation
    Count

>> imshow(read(imgSet(4), 2));  %2nd image in 4th folder
```

Image Representation as BoW — MATLAB uses SURF

```
>> bag = bagOfFeatures(imgSet,'Verbose',true,'VocabularySize',500,'PointSelection','Grid');

Creating Bag-Of-Features from 5 image sets.
--------------------------------------------
* Image set 1: class1.
* Image set 2: class2.
* Image set 3: class3.
* Image set 4: class4.
* Image set 5: class5.

* Extracting SURF features using the Grid selection method.
** The GridStep is [8 8] and the BlockWidth is [32 64 96 128].

* Extracting features from 2 images in image set 1...done. Extracted 98304 features.
* Extracting features from 2 images in image set 2...done. Extracted 50760 features.
* Extracting features from 2 images in image set 3...done. Extracted 11160 features.
* Extracting features from 2 images in image set 4...done. Extracted 8512 features.
* Extracting features from 2 images in image set 5...done. Extracted 19008 features.

* Keeping 80 percent of the strongest features from each image set.

* Balancing the number of features across all image sets to improve clustering.
** Image set 4 has the least number of strongest features: 6810.
** Using the strongest 6810 features from each of the other image sets.

* Using K-Means clustering to create a 500 word visual vocabulary.
* Number of features          : 34050
* Number of clusters (K)      : 500

* Clustering...done.

* Finished creating Bag-Of-Features
```

Computing Histogram of Words for an Image — MATLAB

```
img = read(imgSet(1), 1);
featureVector = encode(bag, img);
```