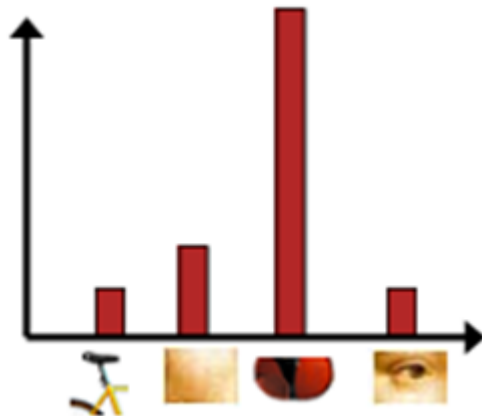


# TDS3651

## Visual Information Processing



### Visual Words: Feature Indexing Lecture 10



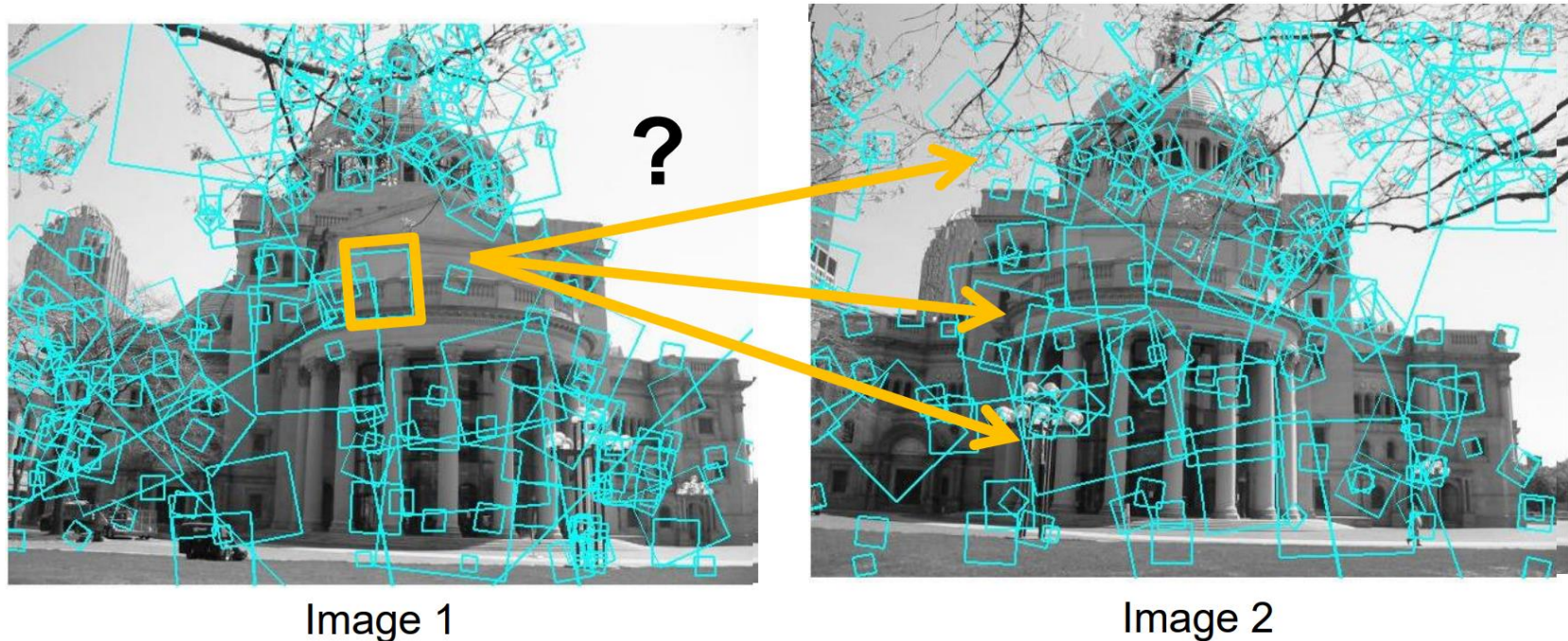
Faculty of Computing and Informatics  
Multimedia University

created by Lai-Kuan, Wong  
modified by Yuen Peng, Loh

# Lecture Outline

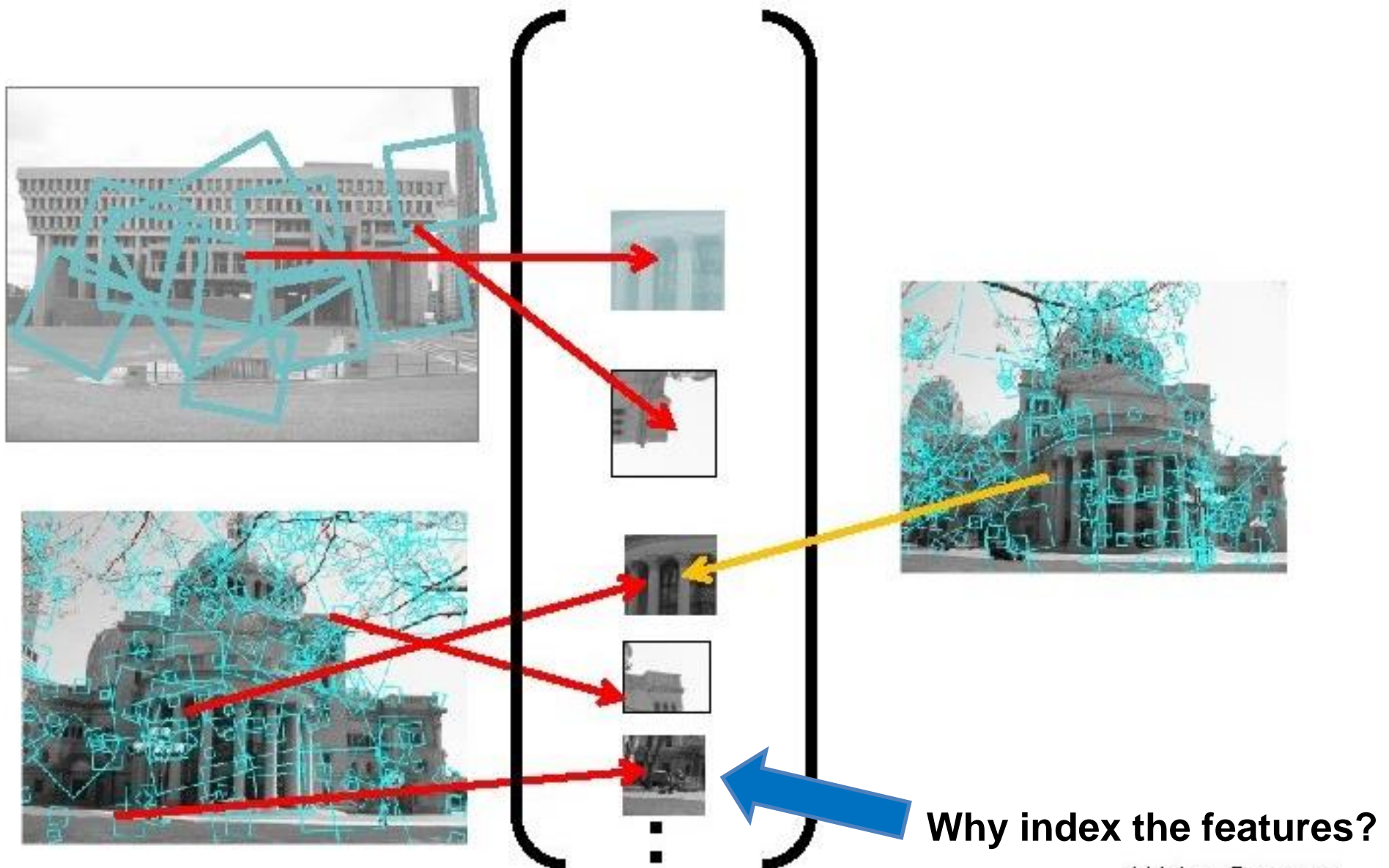
- Feature indexing – Why do it?
- “Visual words” concept
  - Bag of visual words
  - Inverted file index
  - Retrieval scoring
- Application for image retrieval

# Matching local features



- To generate candidate matches, find patches that have the most similar appearance (e.g. lowest SSD)
- Simplest approach: Compare ALL, take the closest (or closest  $k$ , or within a threshold distance)

# Indexing local features

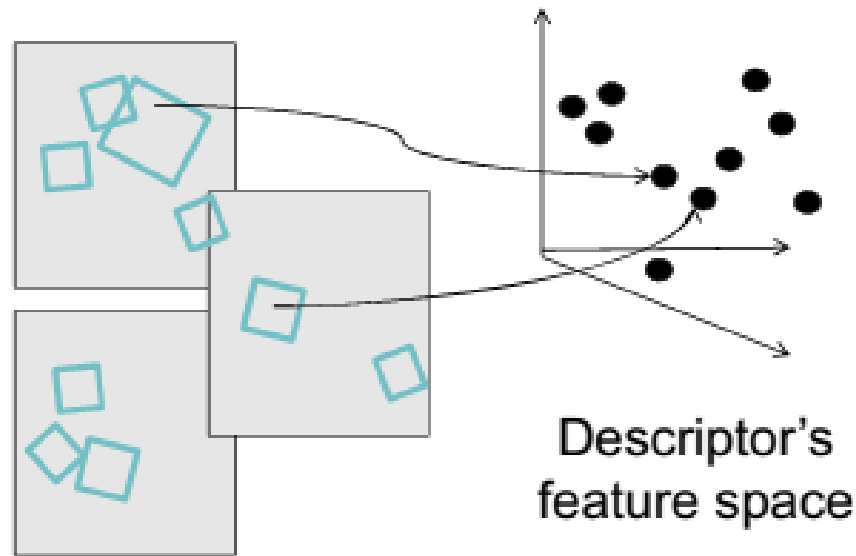


**Why index the features?**

Kristen Grauman

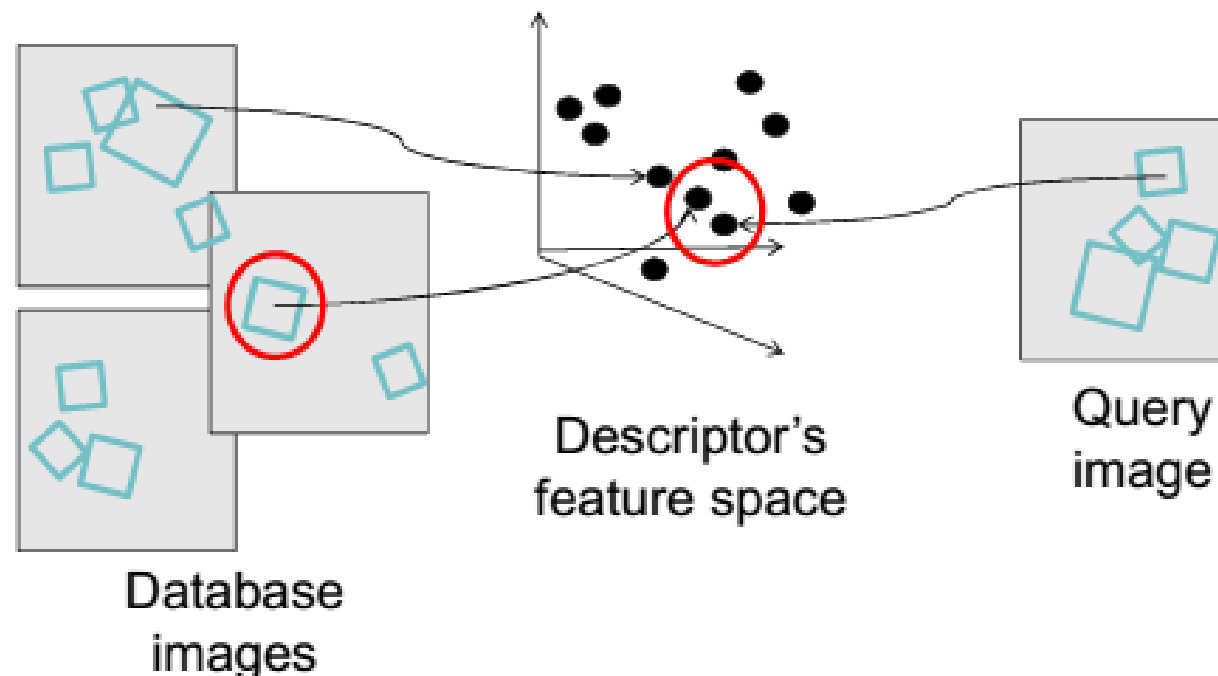
# Indexing Local Features

- Each patch/local region has a descriptor, which is a point in some high-dimensional feature space (e.g. SIFT)



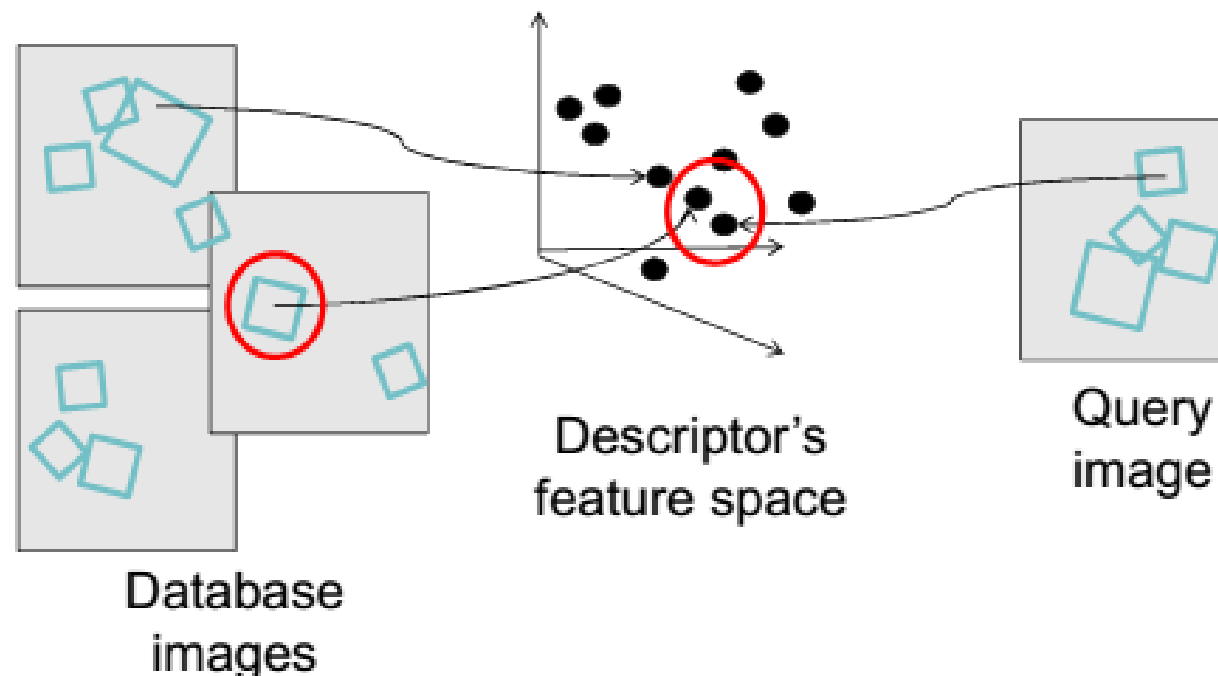
# Indexing Local Features

- When we see **close points in feature space**, we have similar descriptors, which indicates **similar local content**



# Indexing Local Features

- With potentially thousands of features per image, and hundreds to millions of images to search and match, how to efficiently find those that are relevant to a new image?





# Indexing Local Features

- For text documents, an efficient way to find all **pages** on which a **word** occurs is to use an **index**...
- Here, we want to find all **images** in which a **feature** occurs.
- To use this idea, we need to map our features to this “**index**” of visual words.



Index		
*Along I-75, From Detroit to Florida: <i>inside back cover</i>	Butterfly Center, McGuire, 134	Driving Lanes, 85
*Drive I-95, From Boston to Florida: <i>inside back cover</i>	CAA (see AAA)	Duval County, 163
1829 Spanish Trail Roadway, 101-102, 104	CCC, The, 111, 113, 115, 126, 142	Eau Gallie, 179
511 Traffic Information, 83	Ca #22a, 147	Edson, Thomas, 152
A/A (Barrier Is.) - I-95 Access, 86	Caloosahatchee River, 152	Eglin AFB, 116-118
AAA (and CAA), 83	Name, 152	Eight Reals, 176
AAA National Office, 85	Canaveral National Seashore, 173	Ellenton, 144-145
Abbreviations, 85	Cannon Creek Airport, 130	Emanuel Point Neck, 120
Colored 25 mile Maps, <i>nnnn</i>	Canopy Road, 106, 168	Emergency Carboxes, 83
Exit Services, 136	Cape Canaveral, 174	Epiphytes, 142, 148, 157, 159
Travelogue, 85	Castillo San Marcos, 169	Escambia Bay, 119
Africa, 177	Cave Diving, 131	Bridge (I-10), 119
Agricultural Inspection (Bis), 126	Cayo Costa, Name, 150	County, 120
26-Tek-Thru Museum, 160	Celebration, 90	Estero, 153
Air Conditioning, First, 112	Charlotte County, 149	Everglades, 90, 95, 139-140, 154-160
Airbus, 124	Charlotte Harbor, 150	Dawning of, 155, 161
Airbus, 132	Chautauque, 116	Middle SA, 150
County, 131	Chapley, 114	Wonder Gardens, 154
Aisla River, 143	Name, 115	Falling Waters SP, 115
Ajijah, Name, 126	Chortlewood, Name, 115	Fayer Dyles SP, 171
Alfred B. Mackay Gardens, 106	Circus Museum, Ringling, 147	Fires, Forest, 166
Alligator Alley, 154-155	Clive, 88, 97, 130, 136, 140, 160	Fires, Prescribed, 148
Alligator Hole (alligator), 157	CityPlace, W Palm Beach, 180	Fishermen's Village, 151
Alligator, Buddy, 155	City Maps, 180	Flagler County, 171
Alligators, 100, 135, 138, 147, 156	FL Lauderdale Expressway, 194-195	Flagler, Henry, 97, 155, 167, 171
Anadama Island, 170	Jacksonville, 163	Florida Aquarium, 186
Anchorage, 106-109, 146	Kissimmee Expressway, 192-193	Florida, 12,000 years ago, 167
Apalachicola River, 112	Miami Expressway, 194-195	Caverns SP, 114
Appleton Mus of Art, 156	Orlando Expressway, 192-193	Map of all Expressways, 2-3
Aquifer, 102	Pensacola, 28	Mus of Natural History, 134
Arabian Nights, 94	Tallahassee, 191	National Cemetery, 141
Art Museum, Ringling, 147	Tampa (St. Petersburg), 83	Part of Africa, 177
Audubon Beach Cafe, 183	St. Augustine, 191	Platform, 167
Audubon River Project, 106	Civil War, 102, 108, 127, 138, 141	Sheriff's Boys Camp, 126
Babcock-Ross WMA, 151	Clearwater Marine Aquarium, 167	Sports Hall of Fame, 130
Bahia Mar Marina, 184	Collier County, 154	Sun 'n Fun Museum, 97
Baker County, 99	Collier, Barron, 152	Supreme Court, 107
Bankfoot Mallen, 162	Colonial Spanish Quarters, 168	Florida's Turnpike (FTIP), 178, 189
Barge Canal, 137	Columbia County, 101, 128	25 mile Strip Maps, 86
Bee Line Expy, 80	Coquina Building Material, 165	Administration, 189
Belt Outlet Mall, 88	Corkscrew Swamp, Name, 154	Coin Systems, 190
Bennett Castro, 136	Cowboys, 95	Exit Services, 186
Bog "Y", 168	Cuba Trip II, 144	HEFT, 76, 161, 190
Big Cypress, 155, 158	Cruiser, Florida, 88, 95, 132	History, 189
Big Pond Monster, 105	Cruiseway Expy, 11, 35, 36, 143	Name, 189
Black Swamp, Galois, 160	Cuban Snack, 184	Service Plazas, 190
Blackwater River SP, 117	Cuba's Battle, 140	Spur SR91, 76
	Dade, Maj. Francis, 130-140, 161	Ticket System, 189
	Dania Beach Hurricane, 184	Toll Plazas, 190
	Daniel Boone, Florida Walk, 117	
	Daytona Beach, 172-173	

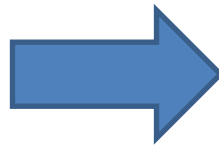


# Text retrieval vs. image search

- So, what makes the problems similar, or different?
  - A. Different because images and texts have different dimensions.
  - B. Different because images describe visual details while text describe high level concepts.
  - C. Similar because we can have a dictionary for image features like a dictionary for text words.
  - D. Similar because image features are the same vectors as text word vectors.

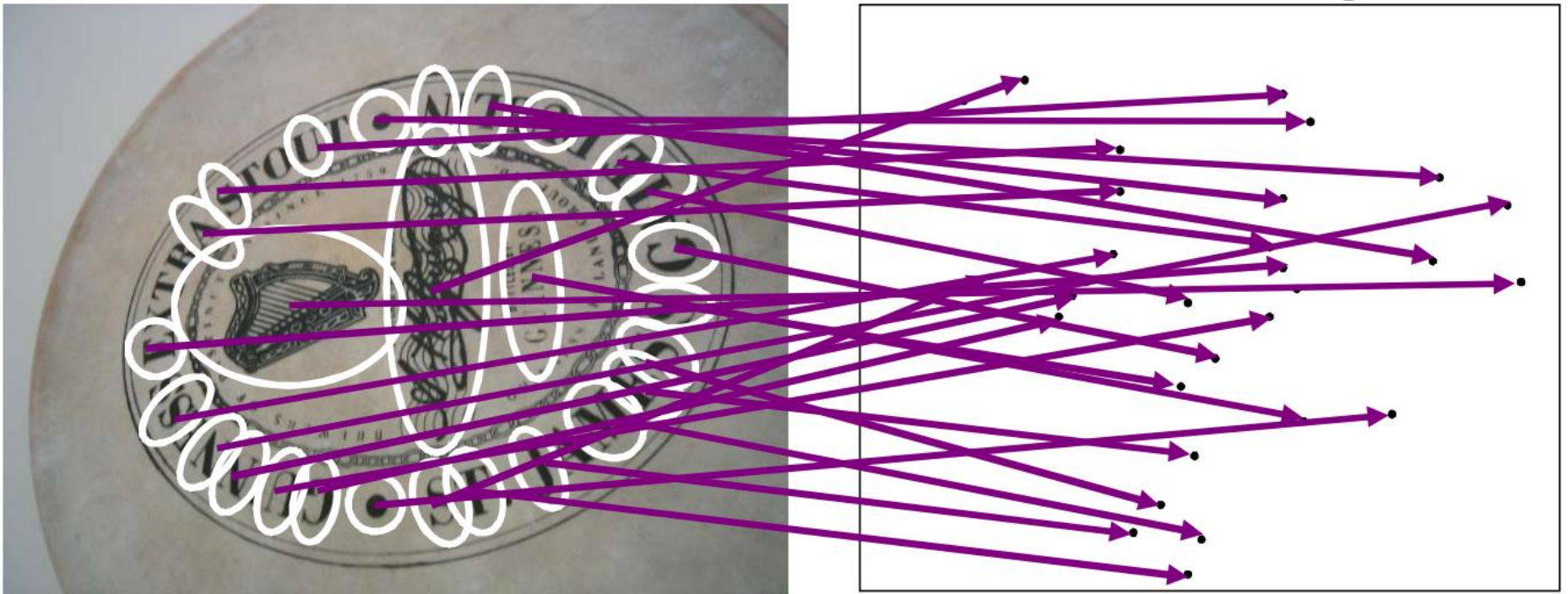
# Concept of Visual Words

# Visual words



# Visual words

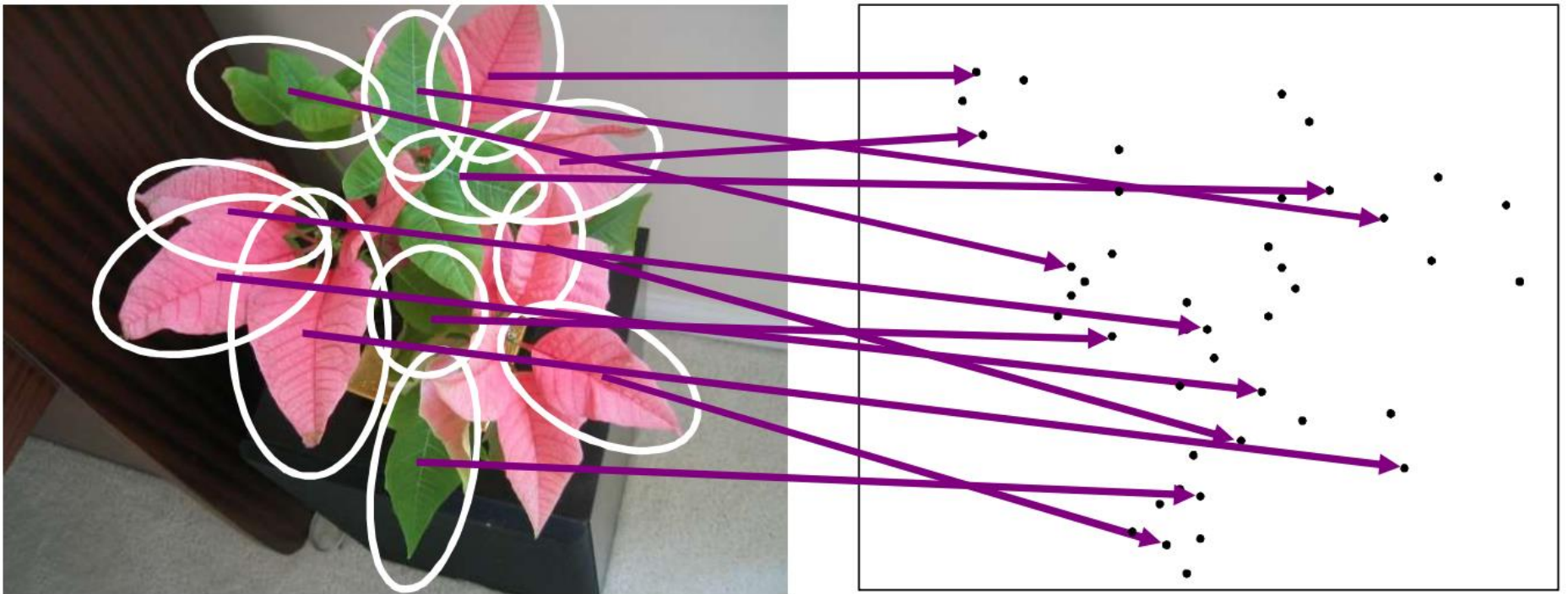
- Extract some local features from a number of images...



e.g. SIFT descriptor space: each point is 128-dimensional

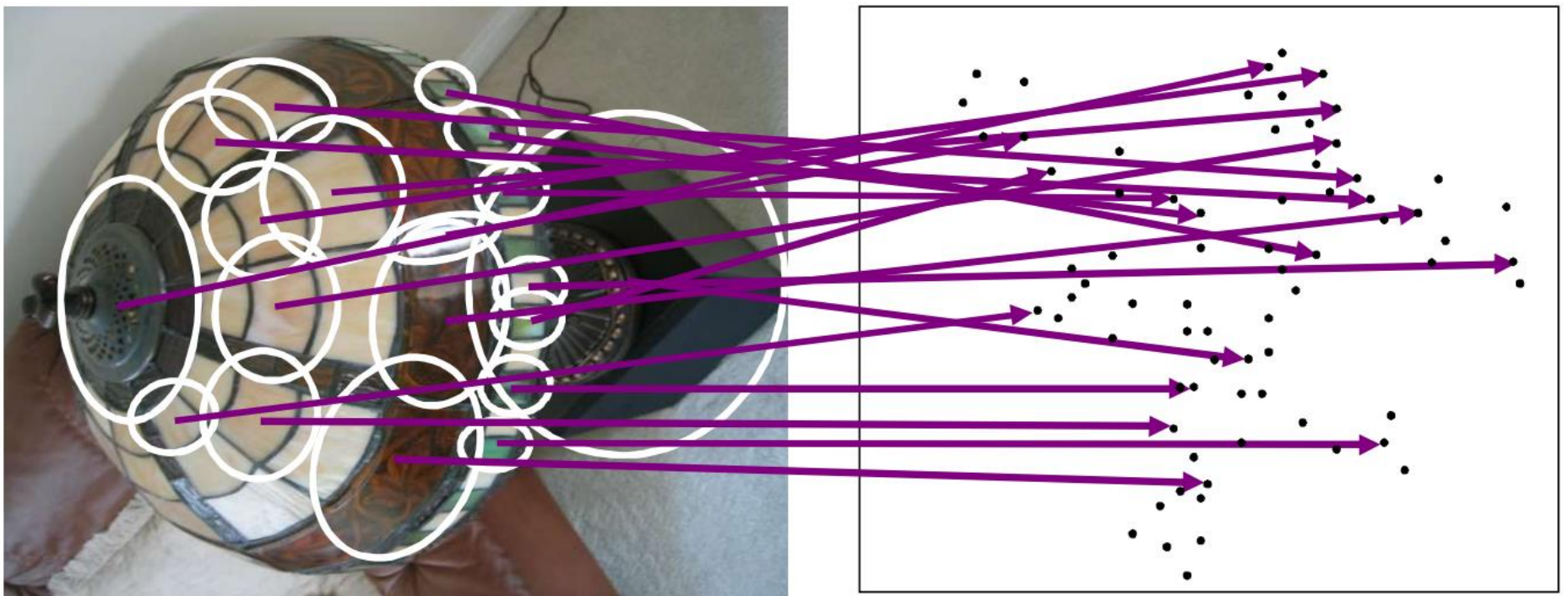
# Visual words

- Of course, it is impossible to visualize 128-dimensions!  
So, most of the time we show it in 2-D or 3-D only...





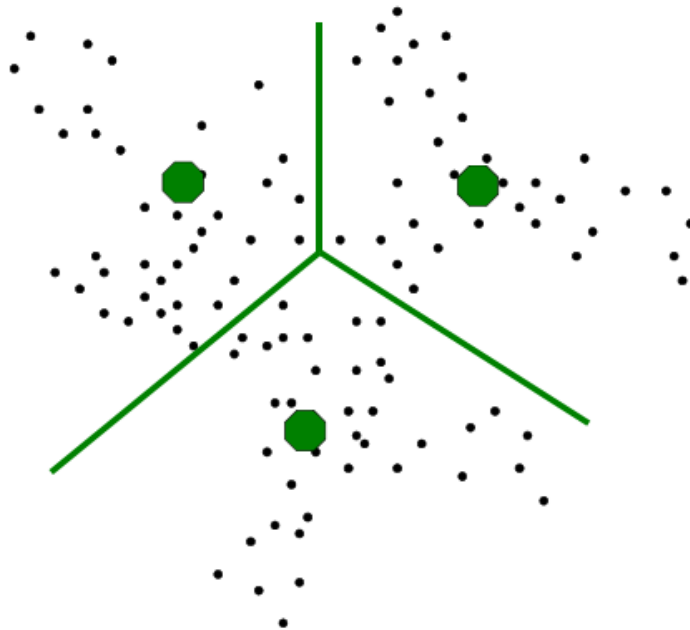
# Visual words





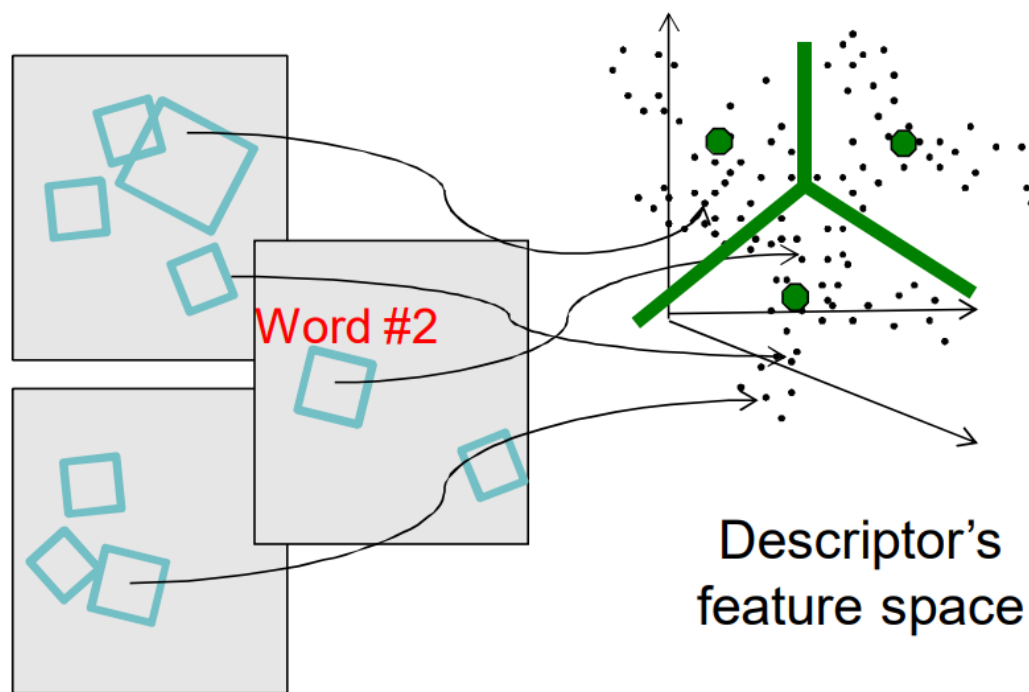
# Visual Words

- Next, we want to try group these points (each 128-dimensions) into groups which will reflect distinctive characteristics
- Solution: Use a clustering technique such as k-means



# Quantizing the feature space

- Map high-dimensional descriptors to **tokens/words** by quantizing the feature space



- Clustering:** Let **cluster centers** be the representative of the “**words**”
- Quantization:** Determine which word to assign to each image descriptor by finding the closest cluster center

# Quantizing the feature space

- Example: Each group of patches belongs to the same visual word.

Look how similar they are after performing clustering

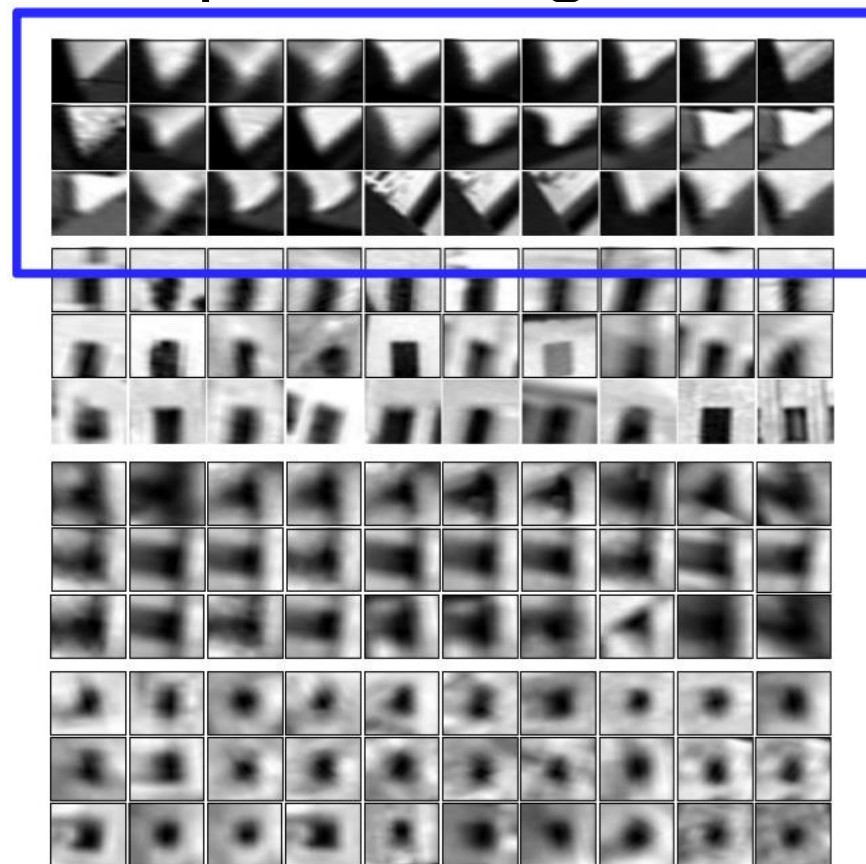
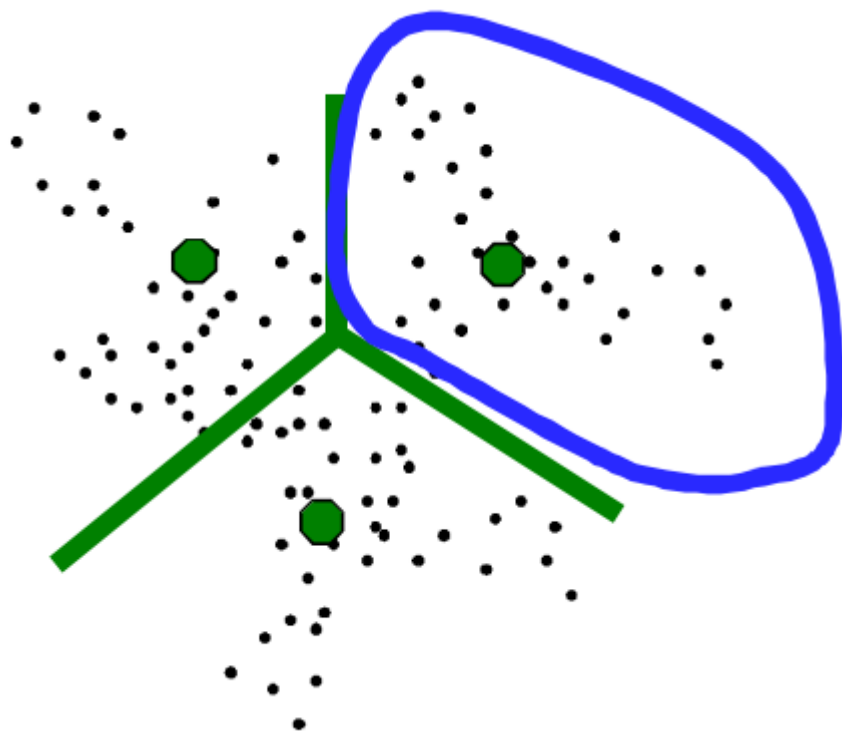
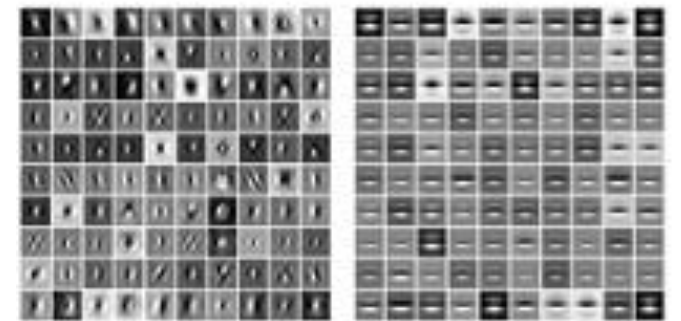
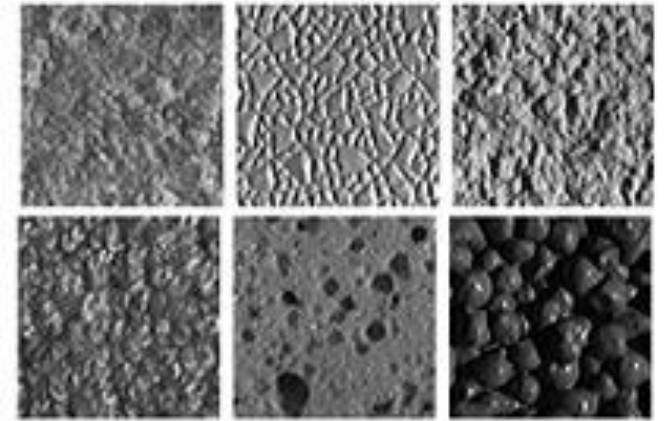


Figure from Sivic & Zisserman, ICCV 2003

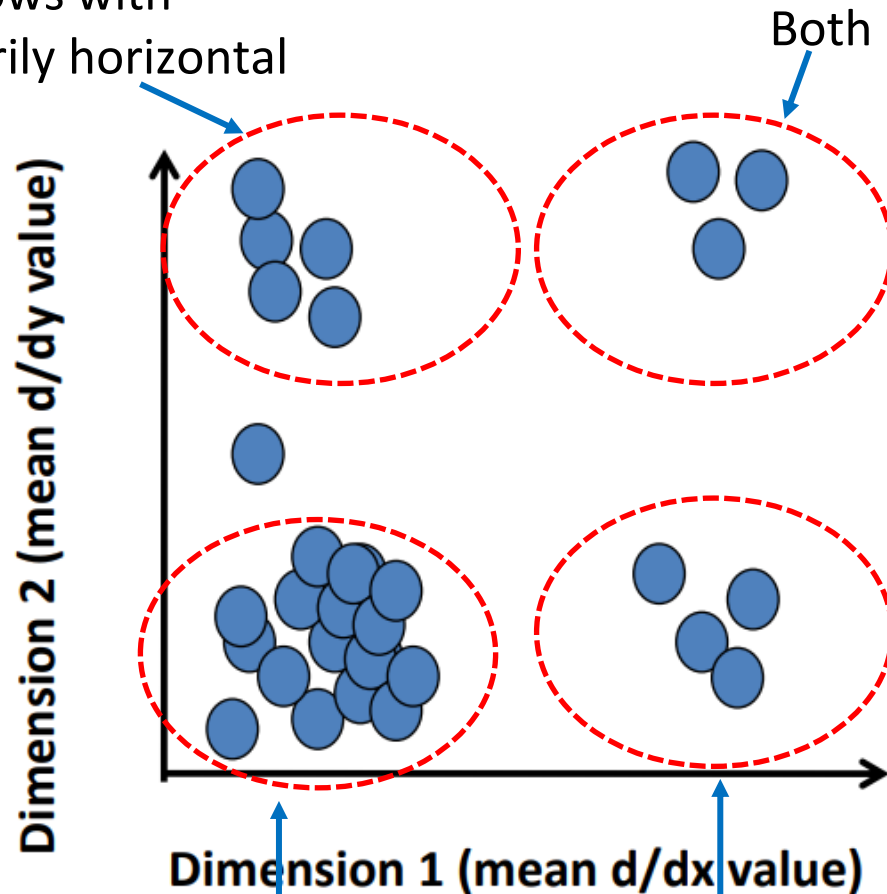
# Visual words and “textons”

- First explored in texture and material representations
- **Texton** = cluster center of filter responses over collection of images
- Describe textures and materials based on distribution of prototypical texture elements



# Recall: Texture Representation

Windows with  
primarily horizontal  
edges



Windows with  
small gradients in  
both directions

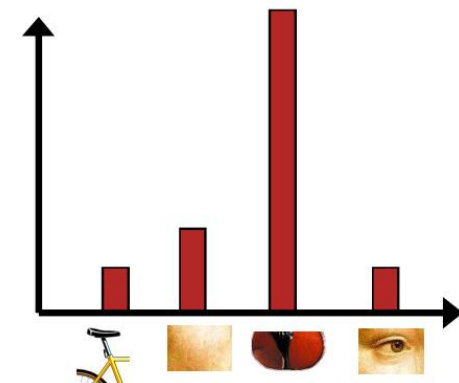
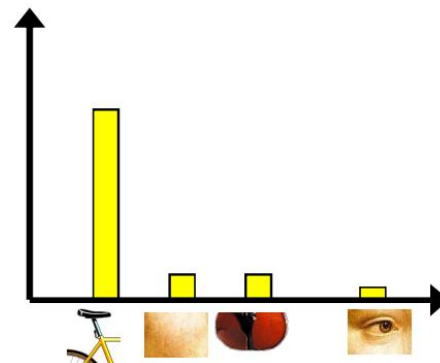
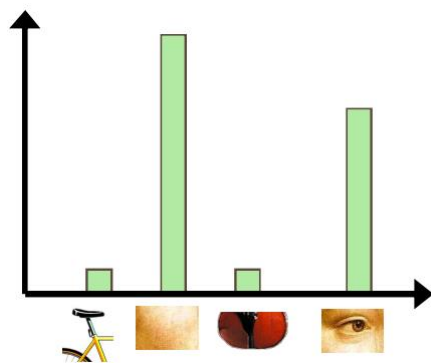
Windows with  
primarily vertical  
edges

	<u>Mean <math>d/dx</math> value</u>	<u>Mean <math>d/dy</math> value</u>
Win. #1	4	10
Win. #2	18	7
⋮	⋮	⋮
Win. #9	20	20

⋮

**statistics to  
summarize patterns  
in small windows**

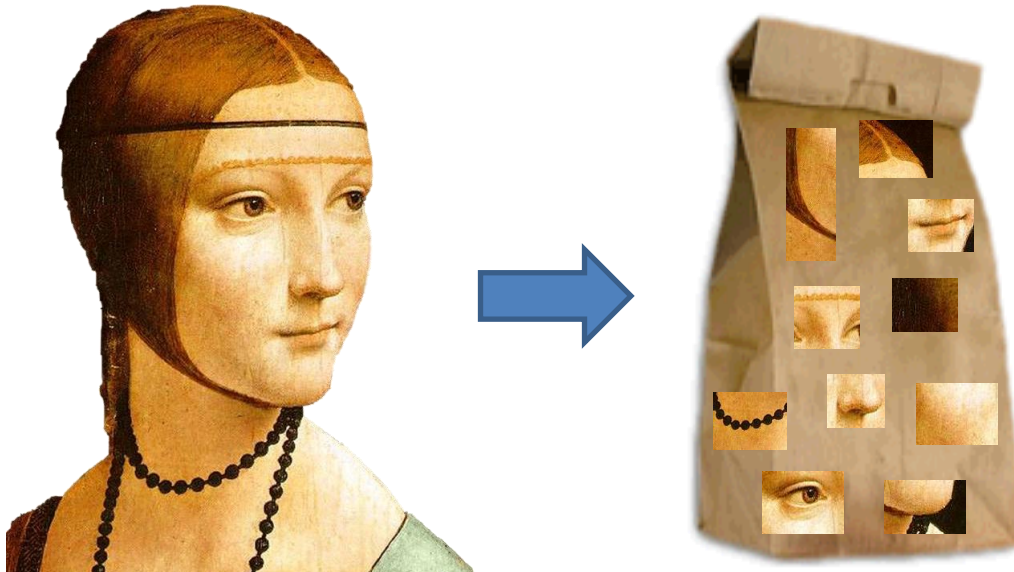
# Whole image in terms of its “parts”





# Bag of visual words

- Summarize entire image based on its distribution (histogram) of visual word occurrences
- Analogous to “bag of words” concept commonly used in documents

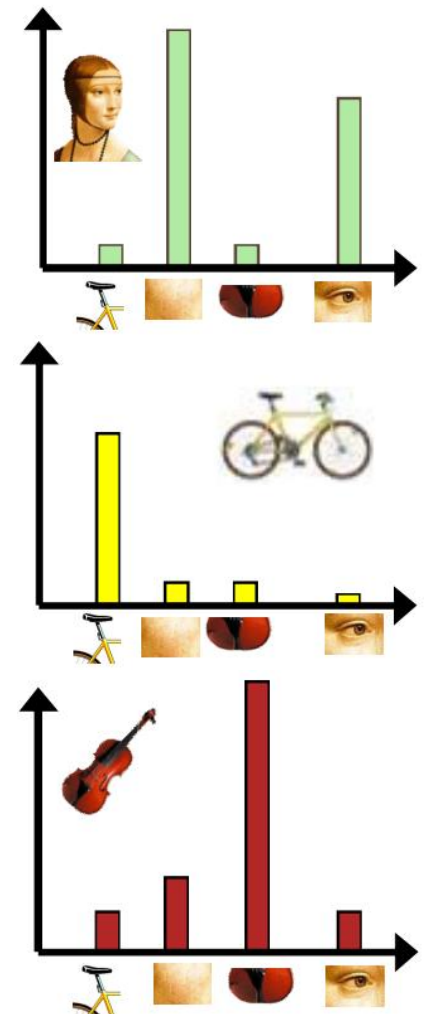


# Bag of visual words

- Summarize entire image based on its distribution (histogram) of visual word occurrences
- Analogous to “bag of words” concept commonly used in documents



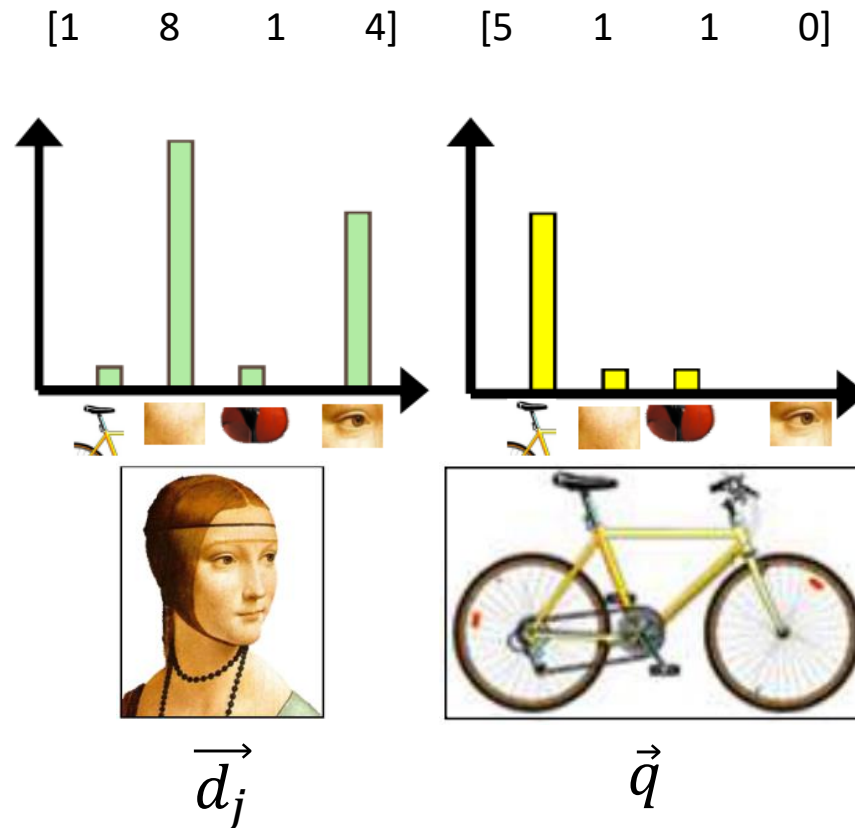
**Visual words**



**Bag of  
visual words**

# Comparing bag of words

- Rank frames by normalized scalar (dot) product between their (possibly weighted) occurrence counts
  - nearest neighbour search for similar images



Cosine similarity

$$\text{sim}(d_j, q) = \frac{\langle d_j, q \rangle}{\|d_j\| \|q\|}$$

$$= \frac{\sum_{i=1}^V d_j(i) * q(i)}{\sqrt{\sum_{i=1}^V d_j(i)^2} * \sqrt{\sum_{i=1}^V q(i)^2}}$$

for vocabulary of  $V$  words

# Visual vocabulary formation

- **Issues:**
  - Sampling strategy: where to extract features?
    - A complex image can contain features in irrelevant portions of the image
  - Clustering / quantization algorithm
    - What are some problems of k-means?
  - Vocabulary size – number of words
    - What's a good number of features to be used for representation

# Inverted File Indexing

# Representing image with visual words



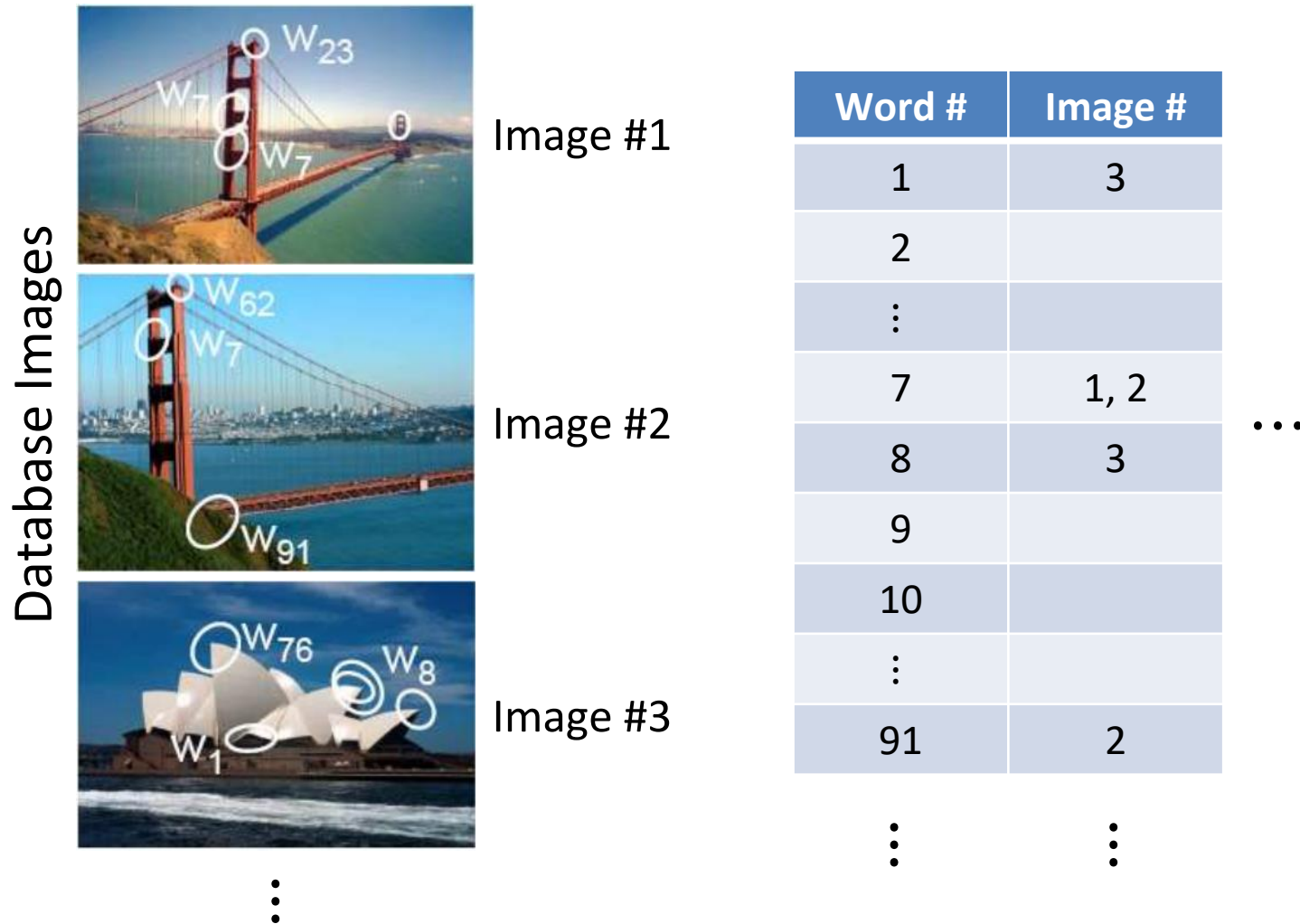
- If a local image region is a visual “word”, how can we summarize an entire image (the “document”) ?



# Analogy to documents



# Inverted file index



- Database images are loaded into the index mapping words to image numbers

# Inverted file index



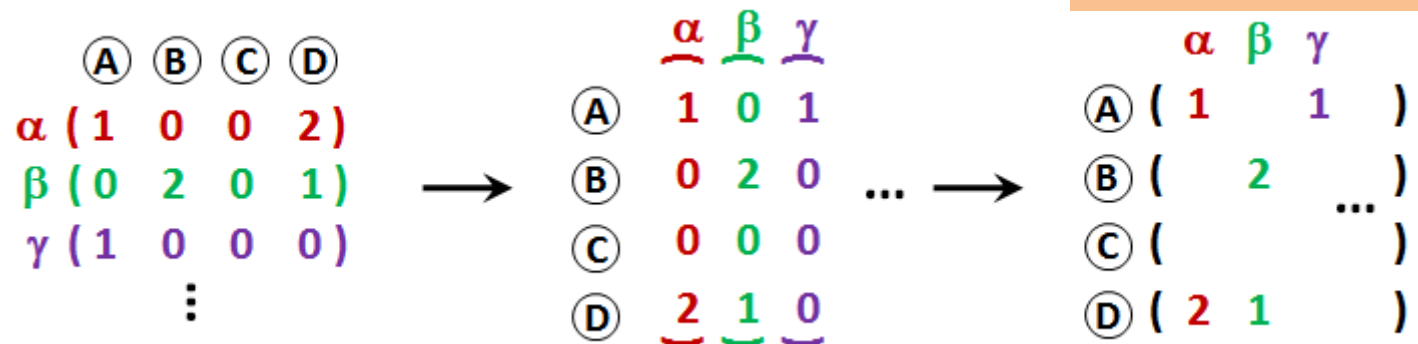
Word #	Image #
1	3
2	
$\vdots$	
7	1, 2
8	3
9	
10	
$\vdots$	
91	2
$\vdots$	$\vdots$



- A new query image is mapped to indices of database images that share a particular word

# Inverted file index

- The visual vocabulary can be very big (thousands to millions of words)
- For quick searching, use an **inverted file index** in sparse form to reduce the size of a matrix that has many zero elements



- Logically, the **weight of each word** computed from its frequency divided by length of vector of words could be useful...

# *tf-idf* weighting

- Term frequency – inverse document frequency
- Describe by frequency of each word within it, **downweight words** that appear often in database
- Standard weighting for text retrieval – can be applied to image search too

The diagram illustrates the tf-idf formula:  $t_{id} = \frac{n_{id}}{n_d} \log \frac{N}{n_i}$ . Annotations include:   
-  $n_{id}$ : Number of occurrences of word  $i$  in document  $d$    
-  $n_d$ : Number of words in document  $d$  (circled in red)   
-  $N$ : Total number of documents in database   
-  $n_i$ : Number of documents word  $i$  occurs in, in whole database   
Two explanatory boxes are present:   
1. A pink box pointing to  $n_d$  states: "Common to perform normalization later, using just the raw frequency"   
2. A blue box pointing to the denominator  $n_i$  states: "Denominator can be 0 if the word does not occur. So normally, we can use  $\log \frac{1+N}{1+n_i} + 1$  to solve this problem"

Number of occurrences of word  $i$  in document  $d$  →  $n_{id}$

Number of words in document  $d$  →  $n_d$

Total number of documents in database →  $N$

Number of documents word  $i$  occurs in, in whole database →  $n_i$

$t_{id} = \frac{n_{id}}{n_d} \log \frac{N}{n_i}$

Common to perform normalization later, using just the raw frequency

Denominator can be 0 if the word does not occur. So normally, we can use  $\log \frac{1+N}{1+n_i} + 1$  to solve this problem



# tf-idf example

- Remember: document  $\Rightarrow$  image

$$t_{id} = n_{id} \left( \log \frac{1 + N}{1 + n_i} + 1 \right)$$

$N$  = number of images in database

$n_i$  = number of images with word  $i$

$n_{id}$  = number of occurrences of word  $i$   
in image  $d$

For Image 0:

$$n_0 = 6, n_{0,0} = 3$$

$$t_{0,0} = 3 \left( \log \frac{1 + 6}{1 + 6} + 1 \right) = 3$$

$$t_{1,0} = 0 \left( \log \frac{1 + 6}{1 + 1} + 1 \right) = 0$$

$$t_{2,0} = 1 \left( \log \frac{1 + 6}{1 + 2} + 1 \right) = 1.368$$

3 words				$N = 6$
Word, $i$	0	1	2	Image, $d$
Counts =	[[3,	0,	1],	0
...	[2,	0,	0],	1
...	[3,	0,	0],	2
...	[4,	0,	0],	3
...	[3,	2,	0],	4
...	[3,	0,	2]]	5

[3,	0,	1.368]
[0.9099,	0,	0.4149]

We see an increase in this value.... Why?

Due to these changes, **normalization MUST** be performed after that, by normalizing the tf-idf value by its magnitude (**Euclidean norm**:  $\|n_{id}\|_2$ )



# *tf-idf* example

Word, $i$	0	1	2	Image, $d$
Counts =	[3,	0,	1],	0
...	[2,	0,	0],	1
...	[3,	0,	0],	2
...	[4,	0,	0],	3
...	[3,	2,	0],	4
...	[3,	0,	2]]	5

[3, 0, 1.368]

We see an increase in this value. Why?

- A. The visual word has the least number of occurrence in the database, so it is most important.
- B. The visual word has less number of occurrence but in various images, so it is useful as an index.
- C. The visual word frequency needs to be normalized hence the change in value.
- D. The frequency properly reflects the general count of words, not only in this database.



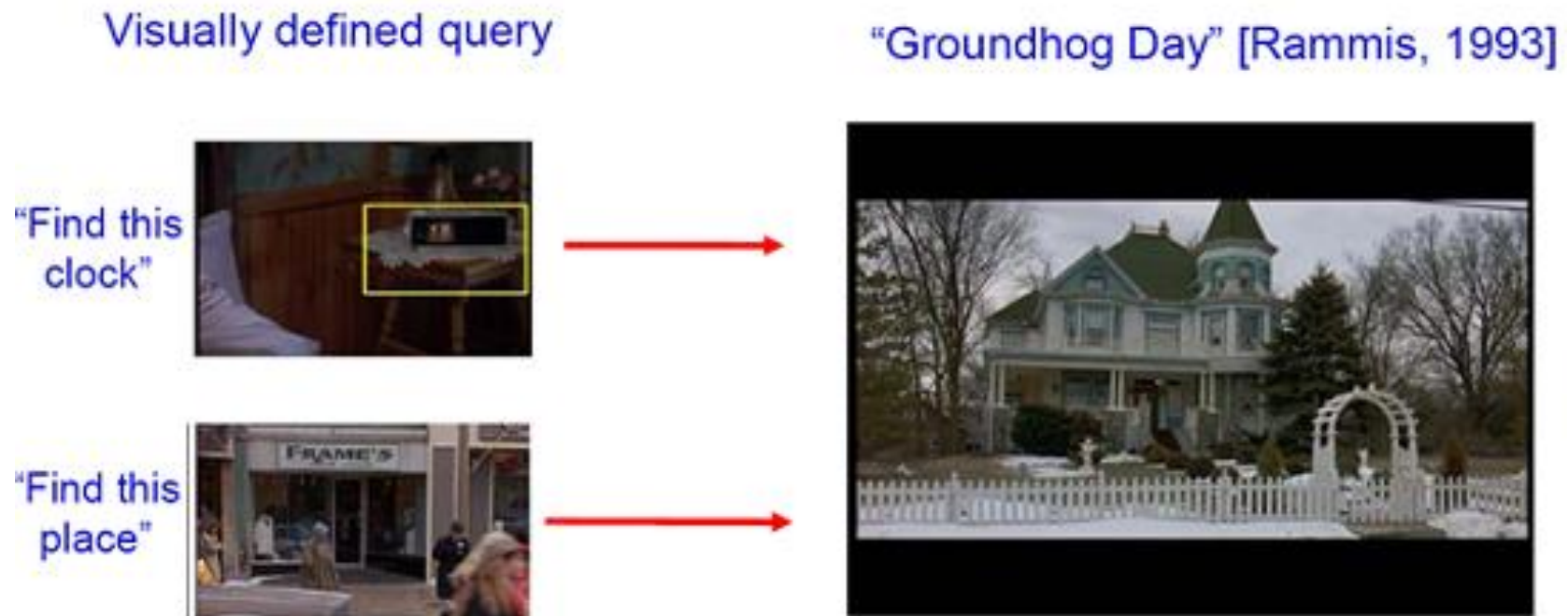
# BOW: Order-less representation

- Bag-of-Words  $\Rightarrow$  **orderless** representation  
(spatial relationships between features are gone)
- But we can use the following ideas to help...
  - **Visual “phrases”** – frequently co-occurring words  
Descriptive visual words and visual phrases for image applications
  - Let **position** be part of each feature
  - **Localize it further**: Perform BOW only within sub-grids or blocks of an image
  - After matching, **verify spatial consistency** (look at neighbours, are they same too?)

# Application and Scoring

# Application of BOW for image retrieval

- Retrieve an object from video that matches the query region



# Video Google System

- “Object matching” in videos



retrieved shots



# Video Google System

1. Collect all words within query region
2. Inverted file index to find relevant frames
3. Compare word counts
4. Spatial verification

*Sivic & Zisserman, ICCV 2003*

<http://www.robots.ox.ac.uk/~vgg/research/vgoogle/index.html>



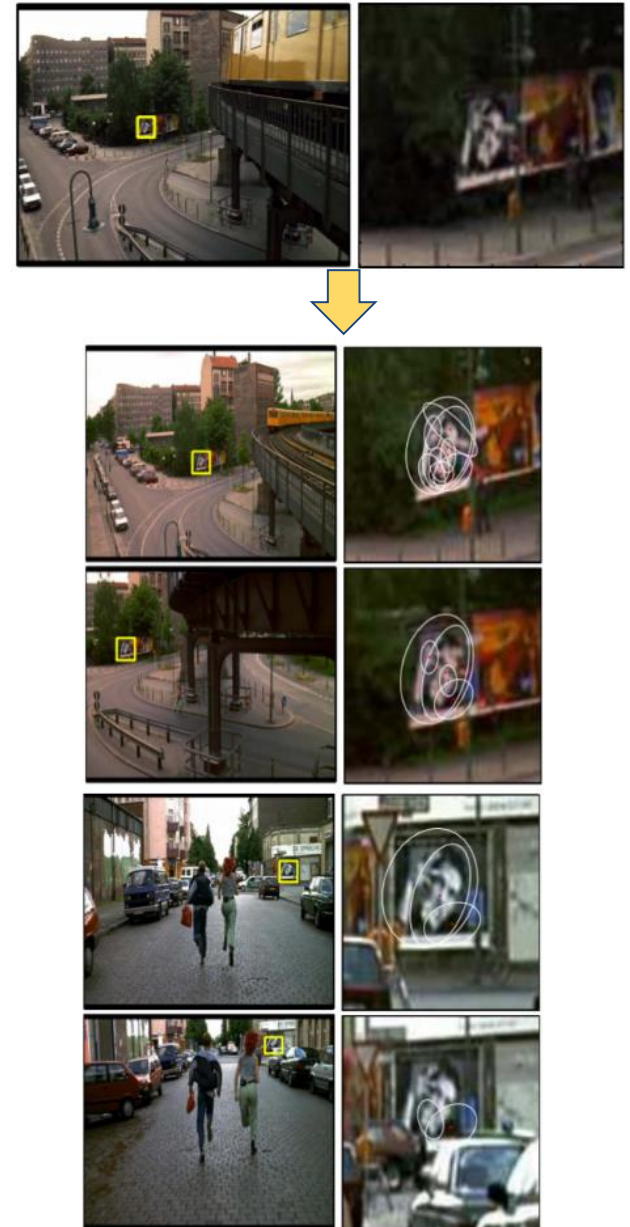


# Video Google System

1. Collect all words within query region
2. Inverted file index to find relevant frames
3. Compare word counts
4. Spatial verification

*Sivic & Zisserman, ICCV 2003*

<http://www.robots.ox.ac.uk/~vgg/research/vgoogle/index.html>





# Scoring retrieval quality

## Example:

Database size: 10 images

Relevant (total): 5 images

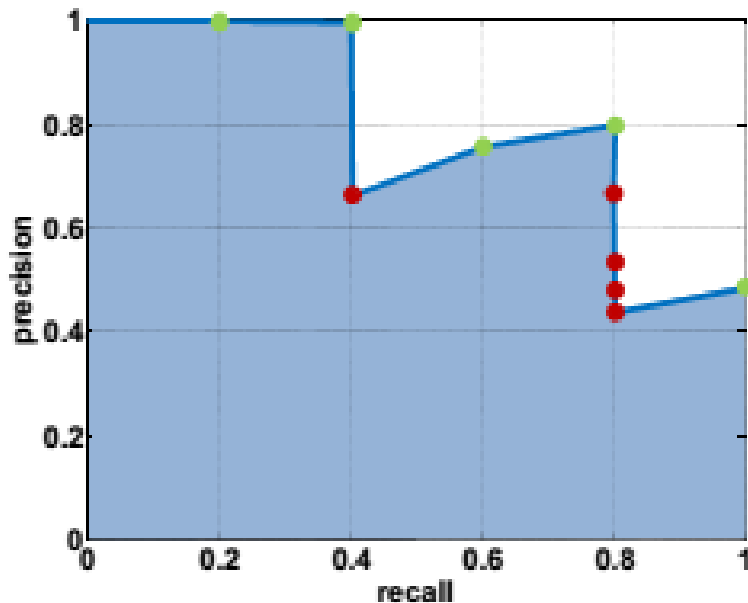


Query

Results (ordered):



Precision = # relevant / # returned  
Recall = # relevant / # total relevant



Precision-Recall curve

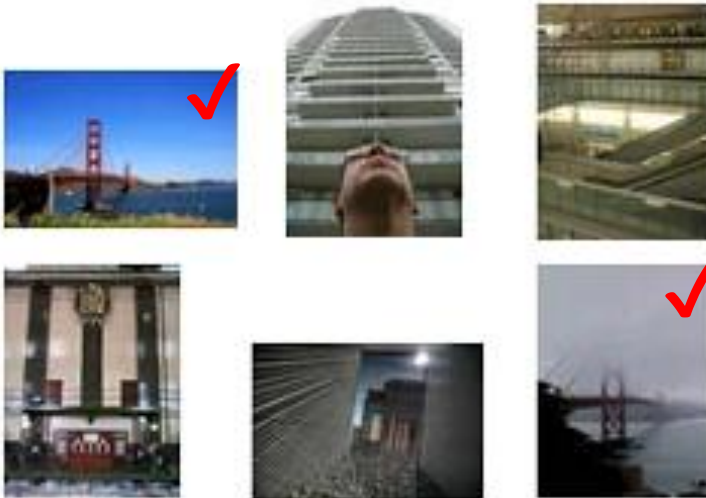
# Scoring retrieval quality

## Example



Query

Results (ordered):



Total returned images: 10

- This can be tuned by setting a threshold on the distance/similarity score

Relevant images retrieved: 5

- Based on ground truth, the number of bridge images returned.

Total relevant images: 8

- Based on ground truth, the number of the bridge images that should have been returned

Recall =  $5/8$

Precision =  $5/10$

# Bag of words: Pros and Cons

- **Pros**

- Flexible to geometry / deformations / viewpoint
- Compact summary of image content
- Provides vector representation for sets
- Very good results in practice

- **Cons**

- Basic model ignores geometry – must verify or encode via features
- Background and foreground mixed when bag covers whole image
- Optimal vocabulary formation remains unclear (how many words?)

# Summary

- Matching local invariant features
  - Useful not only to provide matches for multi-view geometry, but also to find objects and scenes in an image retrieval task
- Bag of words (BOW) representation
  - Quantize feature space to make discrete set of visual words – summarize image by distribution (or histogram) of words – then index these words
- Inverted index
  - Pre-compute index to enable faster search at query time