

# CS 1674/2074: Local features: detection, description and matching

---

PhD. Nils Murrugarra-Llerena

[nem177@pitt.edu](mailto:nem177@pitt.edu)



University of  
Pittsburgh

# [Motivation] Local Features

The "Where's Waldo?" [Game](#)

The goal is to find Waldo in a crowded, complex scene.

How do you search for Waldo?

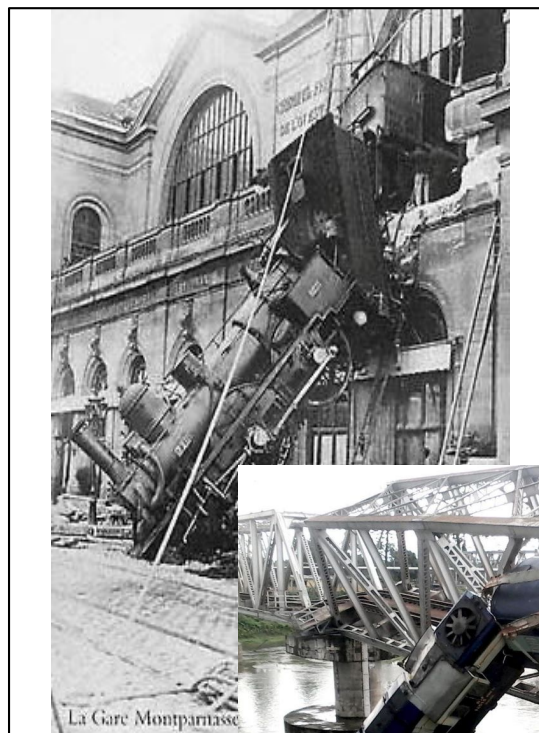


What are  
Waldo's  
distinctive  
features?

## Plan for this lecture

- Feature detection / keypoint extraction
  - Corner detection
- Feature description (of detected features)
- Matching features across images

# An image is a set of pixels



Adapted from S. Narasimhan

# Problems with pixel representation

- Not invariant to small changes
  - Translation
  - Illumination
  - etc.
- Some parts of an image are more important than others
- What do we want to represent?

# Human eye movements



Yarbus eye tracking

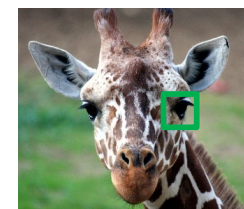
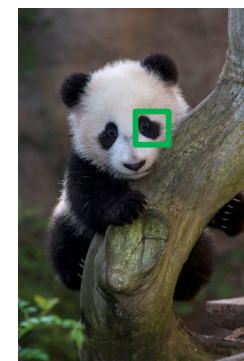
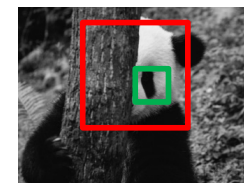


# Local features

- *Local* means that they only cover a small part of the image
- There will be many local features detected in an image; later we'll use those to compute a representation of the whole image
- *Local features* usually exploit image gradients, ignore color
- *Feature*  $\sim$  vector of gradient statistics for a window with *particular location and size*

# Local features: desired properties

- Locality
  - A feature occupies a relatively small area of the image; robust to clutter and occlusion
- Repeatability and flexibility
  - Robustness to expected variations: the **same feature can be found in several images despite geometric/photometric transformations**
  - Maximize correct matches (panda to panda)
- Distinctiveness
  - Each feature has a **distinctive description**
  - Minimize wrong matches (panda to giraffe)
- Compactness and efficiency
  - Many fewer features than image pixels



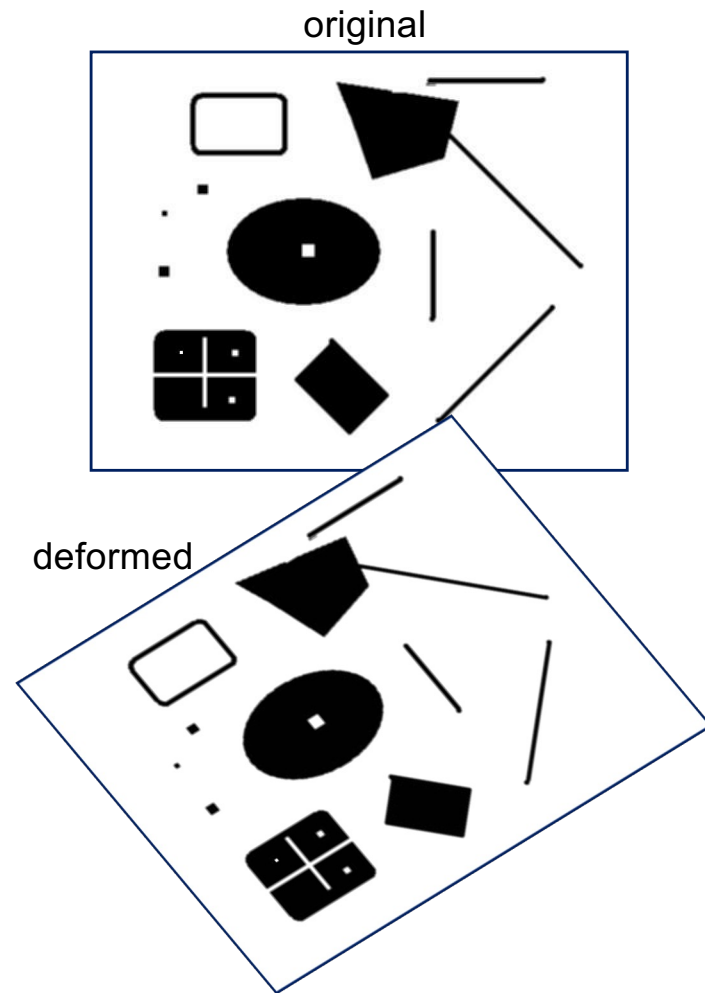


# Interest(ing) points

- Note: “interest points” = “keypoints”, also sometimes called “features”
- Many applications
  - **Recognition**: which patches are likely to tell us something about the object category?
  - **Image search**: which points would allow us to match images between query and database?
  - **3D reconstruction**: how to find correspondences across different views?
  - **Tracking**: which points are good to track?

# Interest points

- Suppose you have to click on some point, go away and come back after I deform the image, and click on the same points again.
  - Which points would you choose?



# Choosing interest points

Where would you tell  
your friend to meet you?

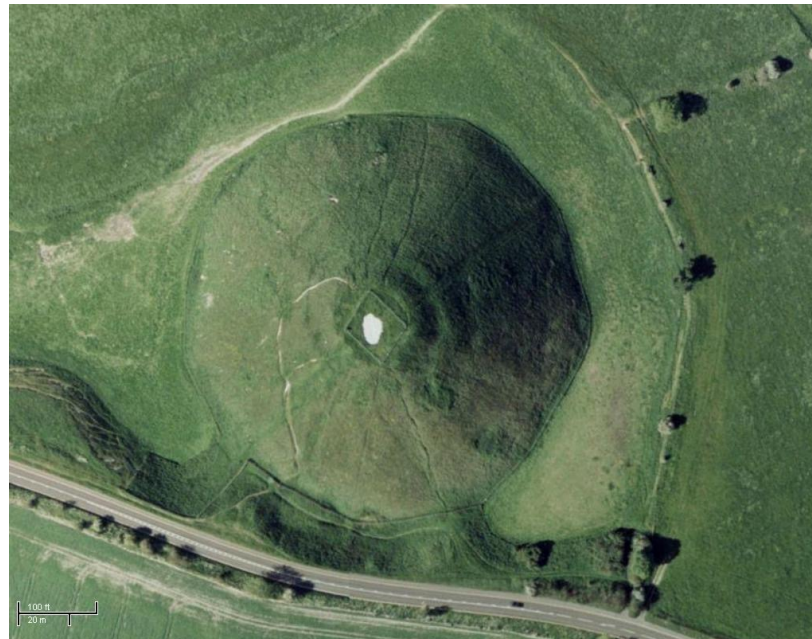
→ Corner detection



# Choosing interest points

Where would you tell  
your friend to meet you?

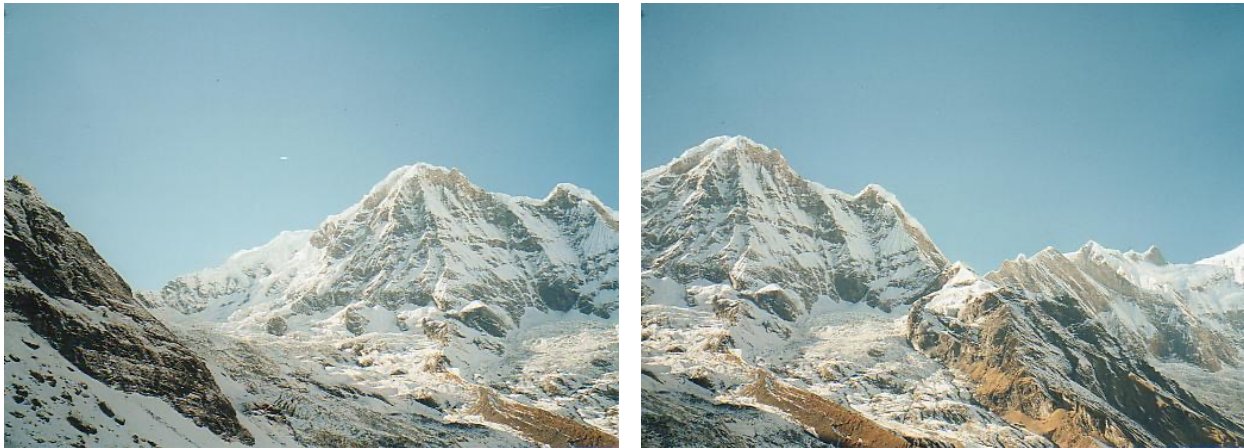
→ Blob detection



D. Hoiem

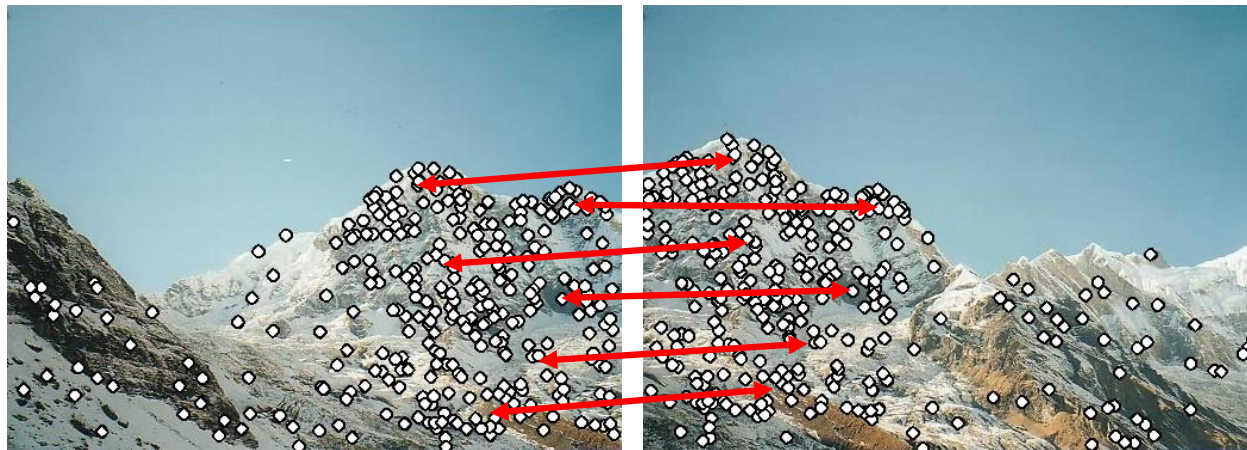
# Application: Panorama stitching

- We have two images – how do we combine them?



# Application: Panorama stitching

- We have two images – how do we combine them?



Step 1: extract features

Step 2: match features



# Application: Panorama stitching

- We have two images – how do we combine them?



Step 1: extract features

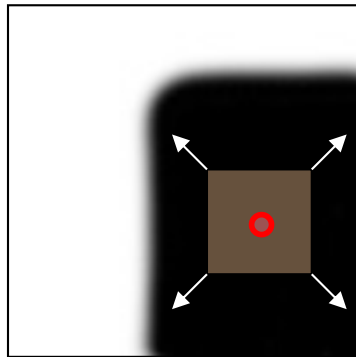
Step 2: match features

Step 3: align images

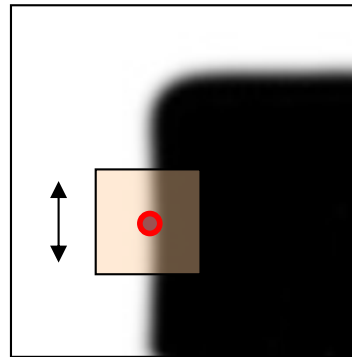
## Corners are distinctive interest points

- We should **easily recognize the keypoint** by looking through a small window
- **Shifting a window in *any direction* should give a *large change* in intensity**

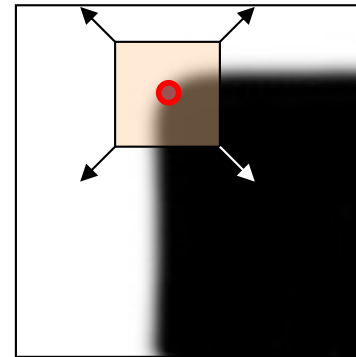
● Candidate keypoint



“flat” region:  
no change in  
all directions



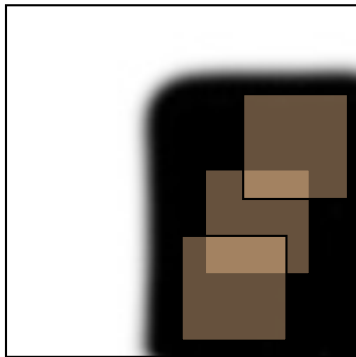
“edge”:  
no change along  
the edge direction



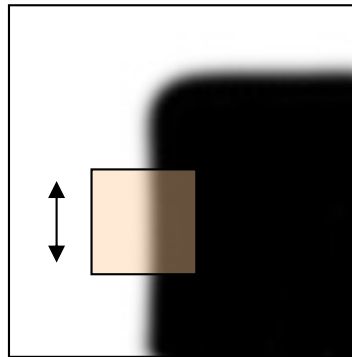
“corner”:  
significant change  
in all directions

## Corners are distinctive interest points

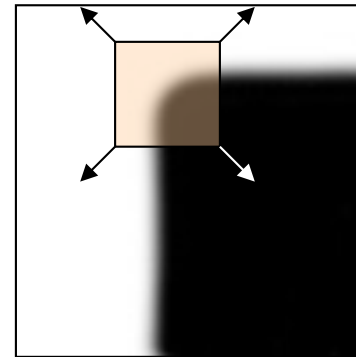
- We should easily recognize the keypoint by looking through a small window
- Shifting a window in *any direction* should give a *large change* in intensity



“flat” region:  
no change in  
all directions



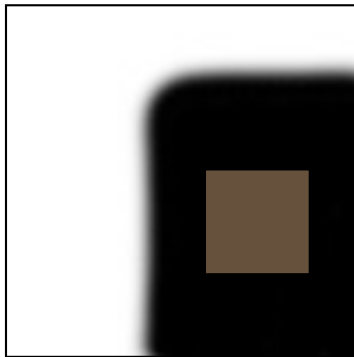
“edge”:  
no change along  
the edge direction



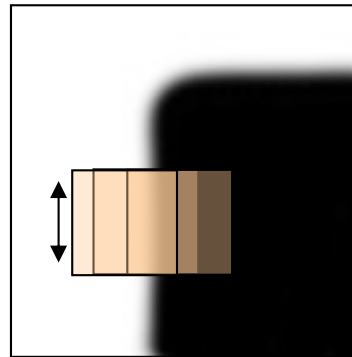
“corner”:  
significant change  
in all directions

## Corners are distinctive interest points

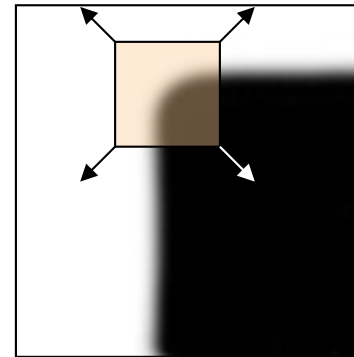
- We should easily recognize the keypoint by looking through a small window
- Shifting a window in *any direction* should give a *large change* in intensity



“flat” region:  
no change in  
all directions



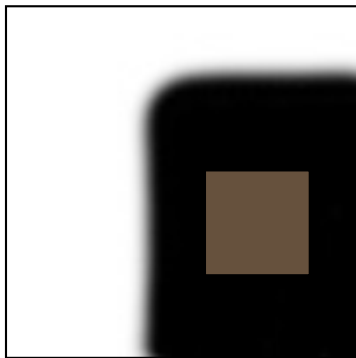
“edge”:  
no change along  
the edge direction



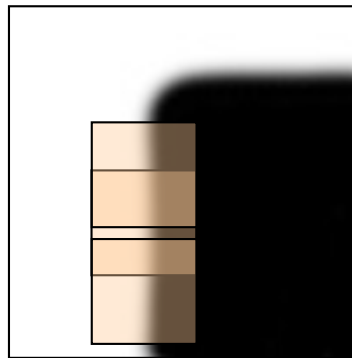
“corner”:  
significant change  
in all directions

## Corners are distinctive interest points

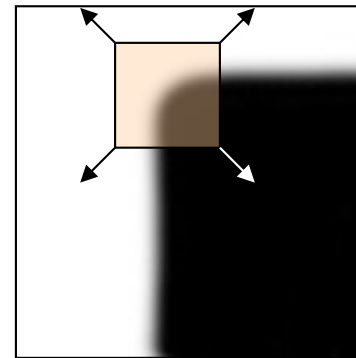
- We should easily recognize the keypoint by looking through a small window
- Shifting a window in *any direction* should give a *large change* in intensity



“flat” region:  
no change in  
all directions



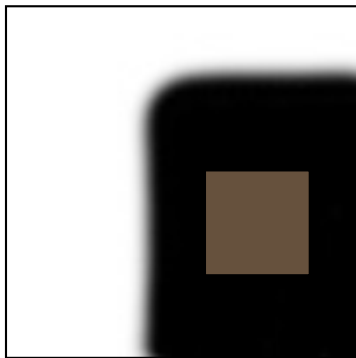
“edge”:  
no change along  
the edge direction



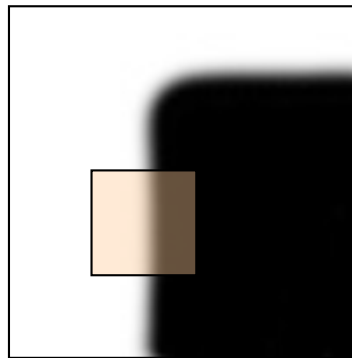
“corner”:  
significant change  
in all directions

## Corners are distinctive interest points

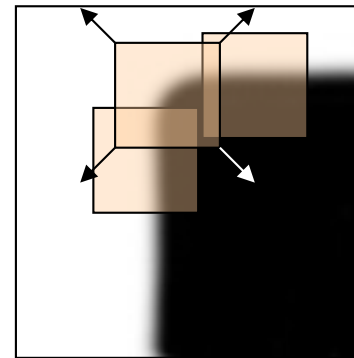
- We should easily recognize the keypoint by looking through a small window
- Shifting a window in *any direction* should give a *large change* in intensity



“flat” region:  
no change in  
all directions



“edge”:  
no change along  
the edge direction




“corner”:  
significant change  
in all directions



## Harris Detector: Mathematics

Window-averaged squared change of intensity induced by shifting the patch for a fixed candidate keypoint by  $[u,v]$ :

$$E(u, v) = \sum_{x, y} [I(x + u, y + v) - I(x, y)]^2$$


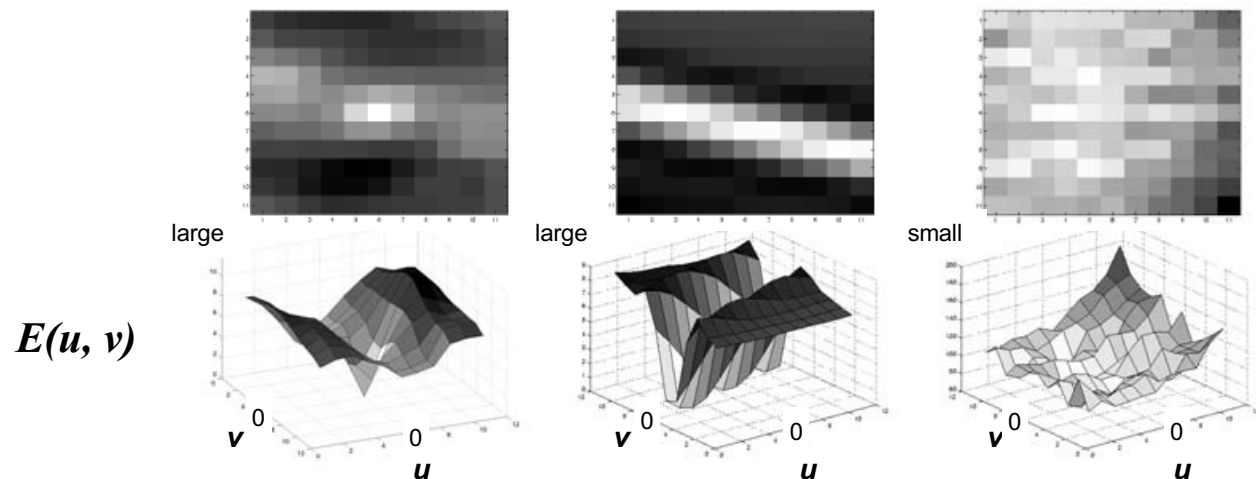
Shifted intensity

Intensity

# Harris Detector: Mathematics

Window-averaged squared change of intensity induced by shifting the patch for a fixed candidate keypoint by  $[u,v]$ :

$$E(u, v) = \sum_{x, y} [I(x + u, y + v) - I(x, y)]^2$$



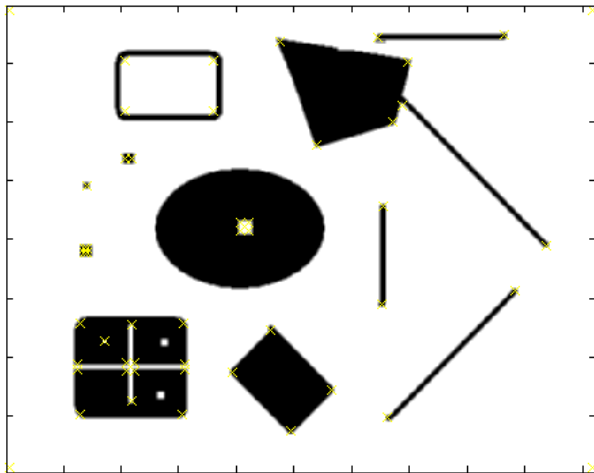
Adapted from D. Frolova, D. Simakov

# Example of Harris Application

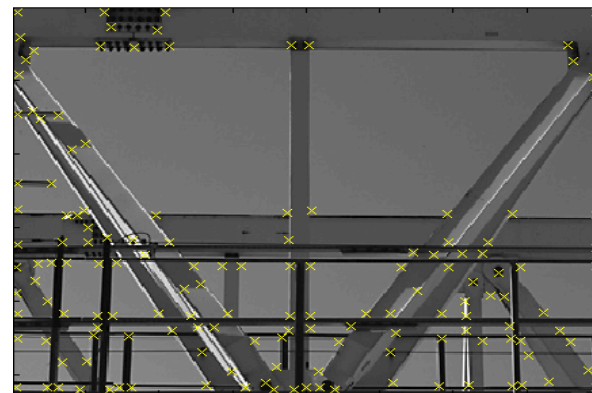
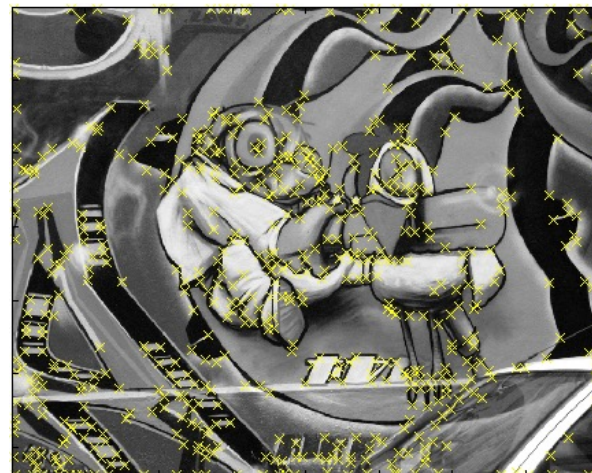


K. Grauman

## More Harris Responses



***Effect:*** A very precise corner detector.



## Plan for this lecture

- Feature detection / keypoint extraction
  - Corner detection
- Feature description (of detected features)
- Matching features across images

# Geometric transformations

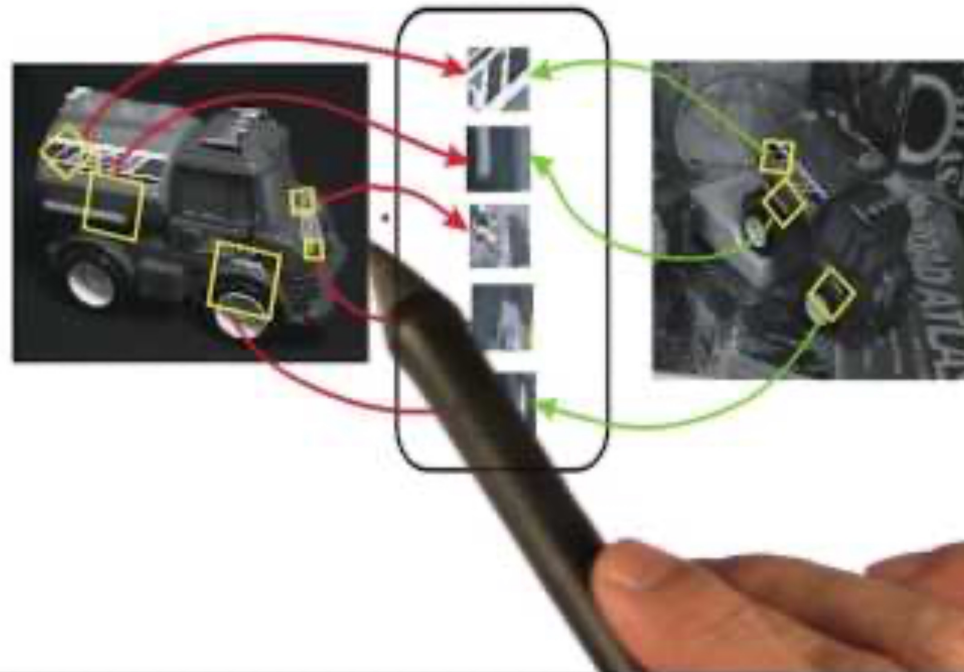




# Short Video SIFT Descriptor

---

*Invariant Local Features*

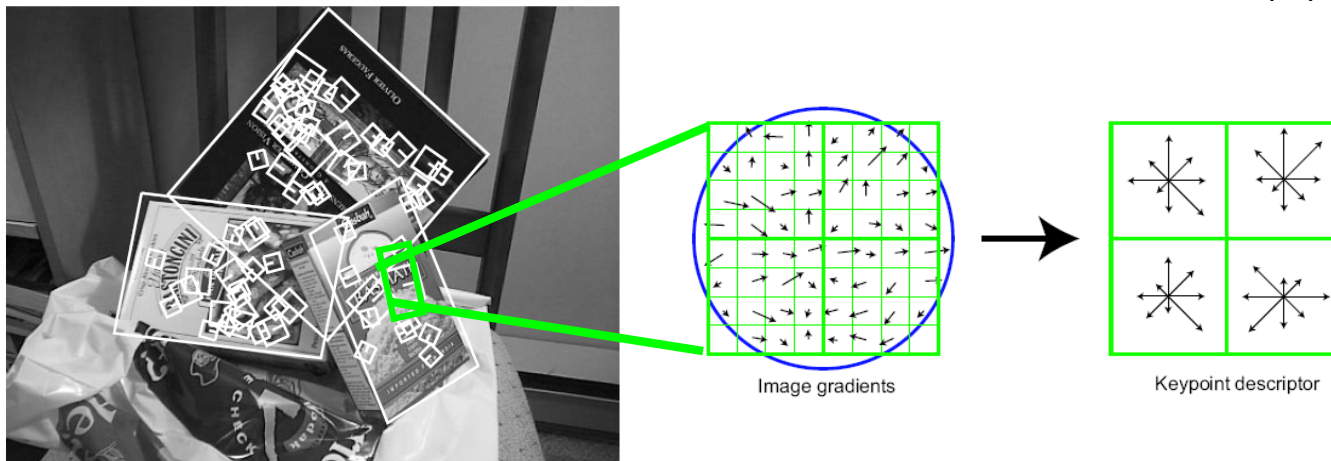


---

[https://www.youtube.com/watch?v=oKAnOzlu66c&ab\\_channel=Udacity](https://www.youtube.com/watch?v=oKAnOzlu66c&ab_channel=Udacity)

# Scale-Invariant Feature Transform (SIFT) descriptor

Journal + conference versions: 87,527 citations (AlexNet paper has 93,821)



## Histogram of oriented gradients

- Captures important texture information
- Robust to small translations / affine deformations

[Lowe, ICCV 1999]

K. Grauman, B. Leibe

# Computing gradients

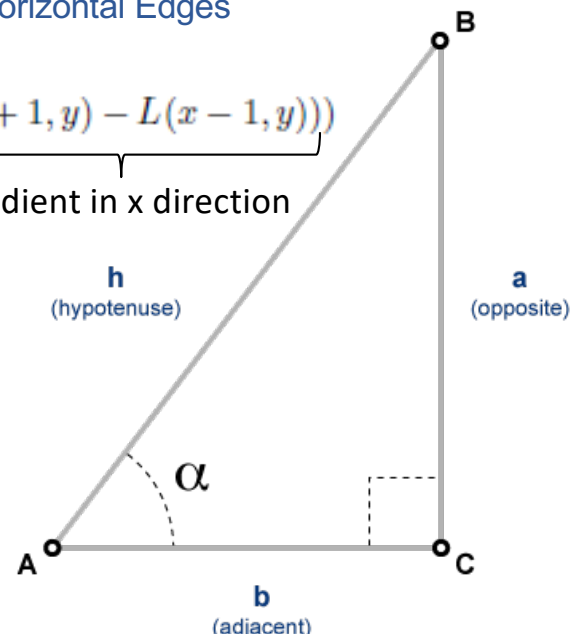
$L$  = the image intensity

$$m(x, y) = \sqrt{\underbrace{(L(x+1, y) - L(x-1, y))^2}_{\text{gradient in x direction}} + \underbrace{(L(x, y+1) - L(x, y-1))^2}_{\text{gradient in y direction}}}$$

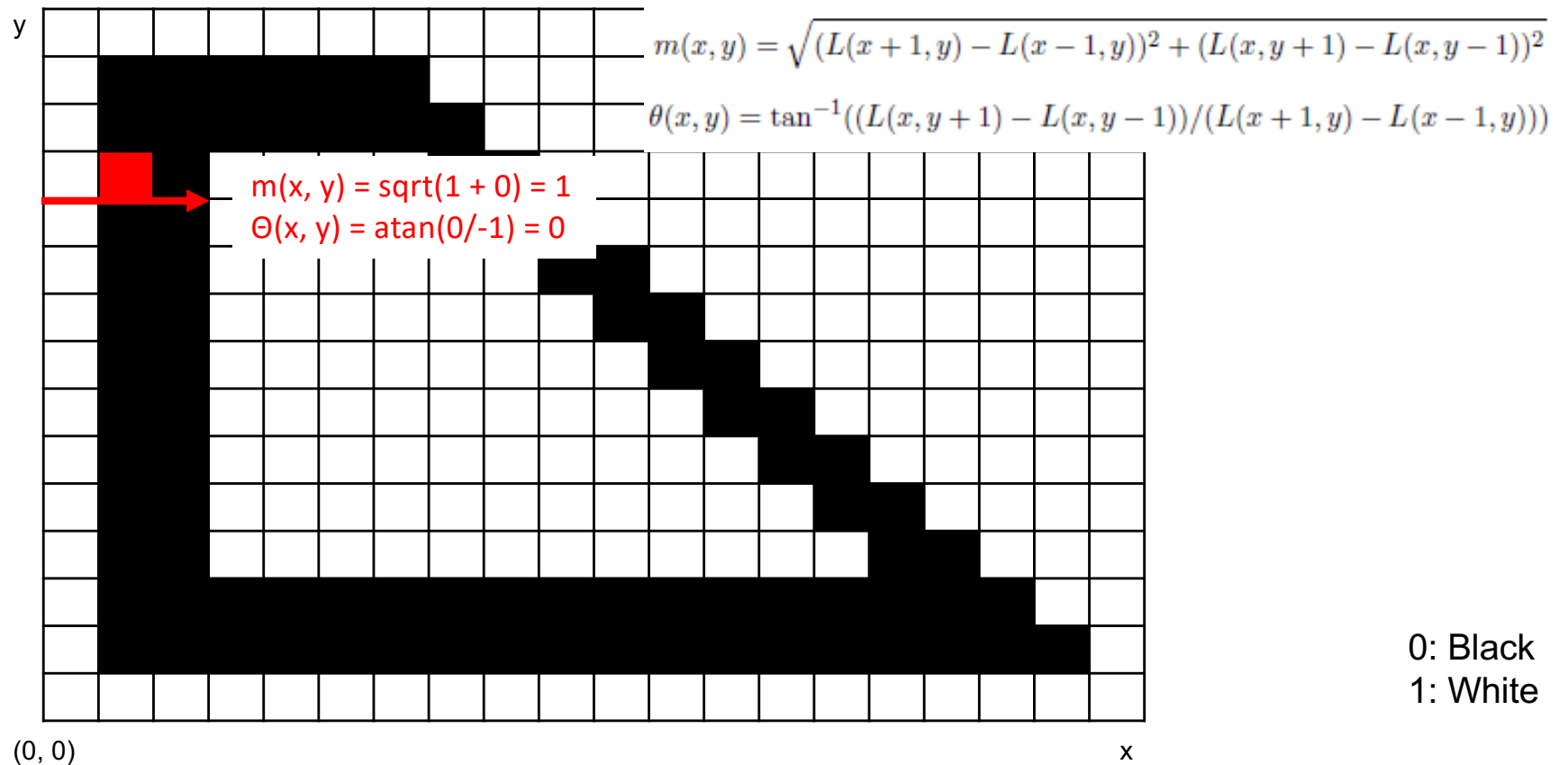
Vertical Edges                      Horizontal Edges

$$\theta(x, y) = \tan^{-1} \left( \underbrace{(L(x, y+1) - L(x, y-1))}_{\text{gradient in y direction}} / \underbrace{(L(x+1, y) - L(x-1, y))}_{\text{gradient in x direction}} \right)$$

- $\tan(\alpha) = \frac{\text{opposite side}}{\text{adjacent side}}$

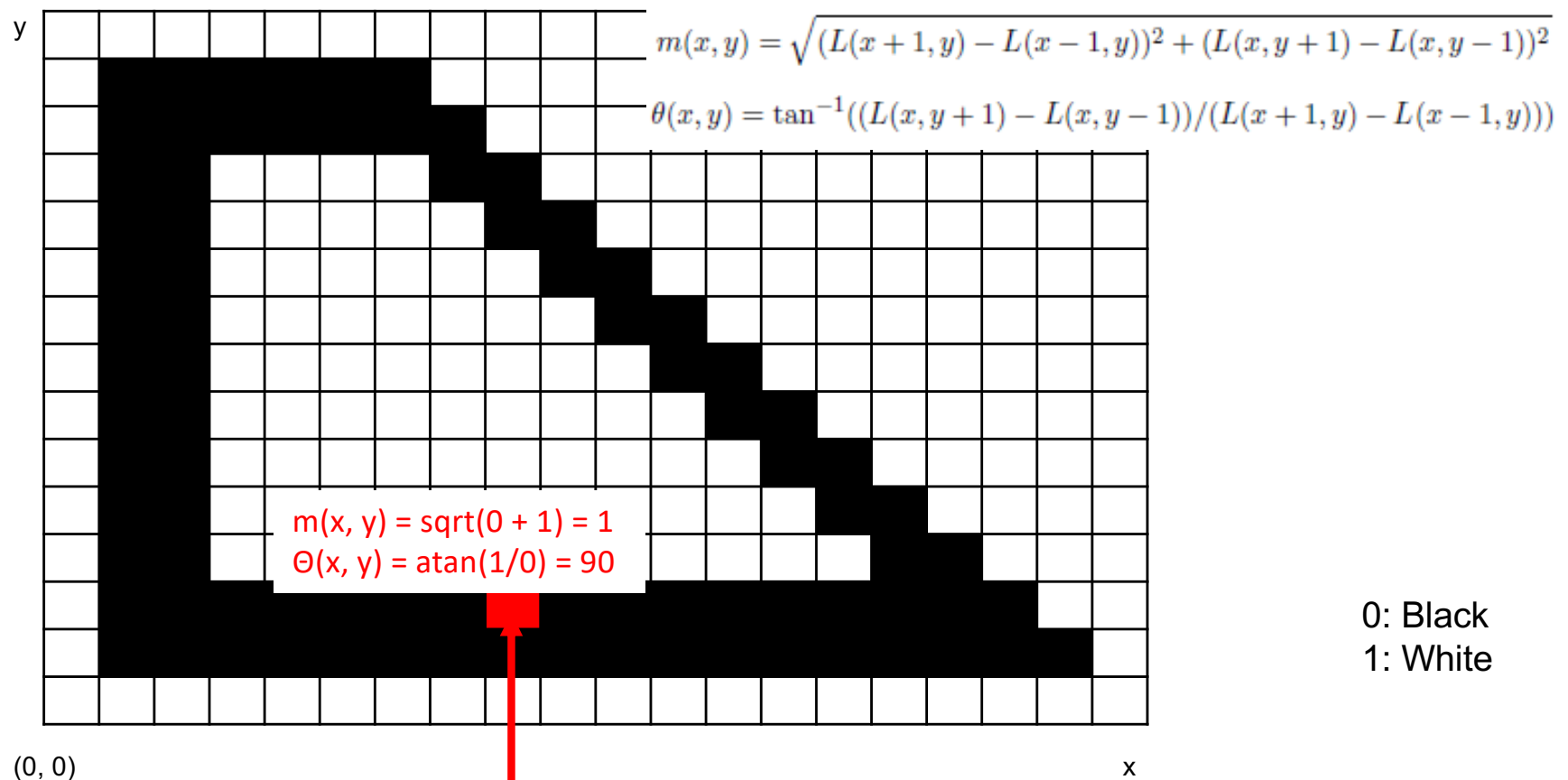


# Gradients

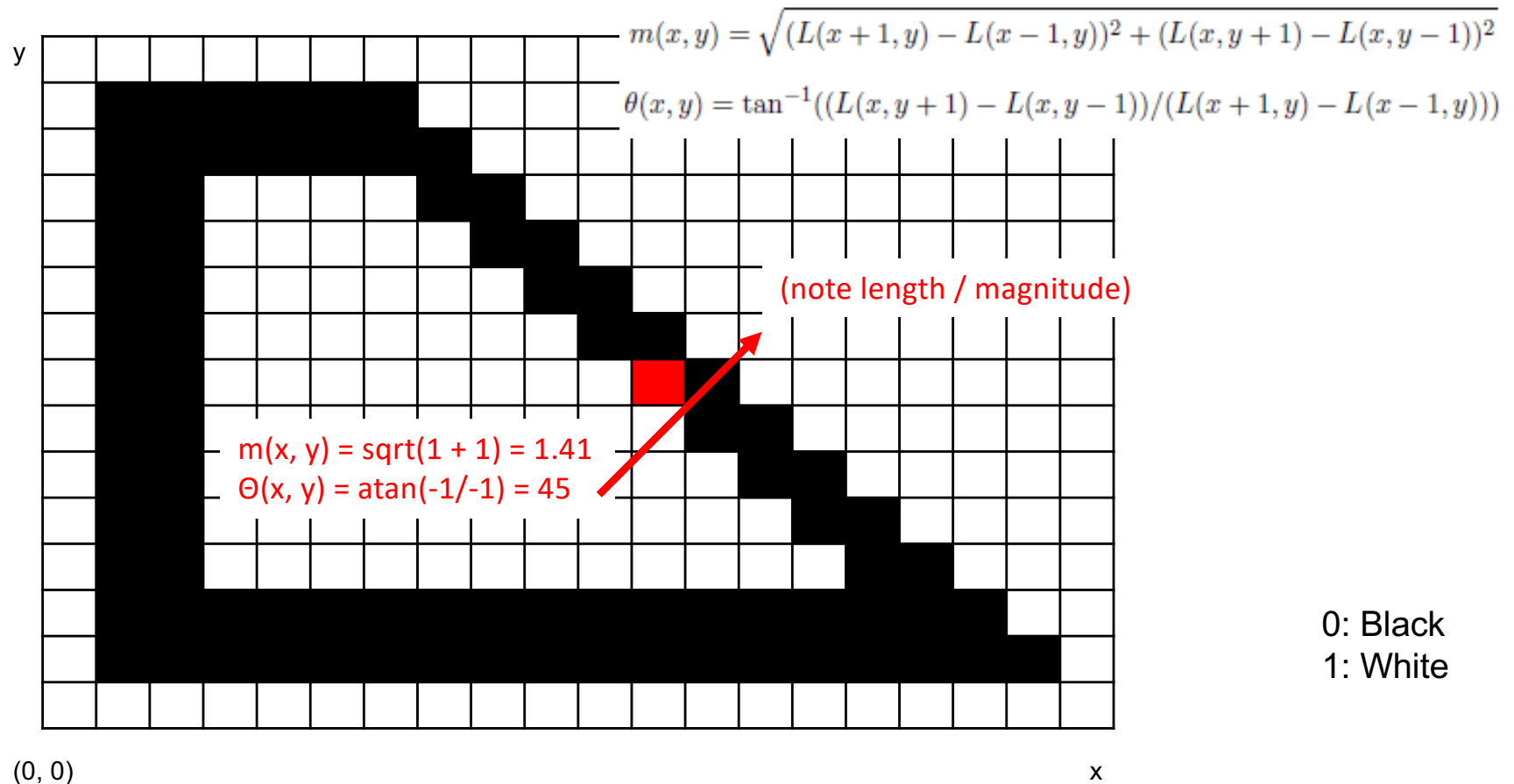


0: Black  
1: White

# Gradients



# Gradients

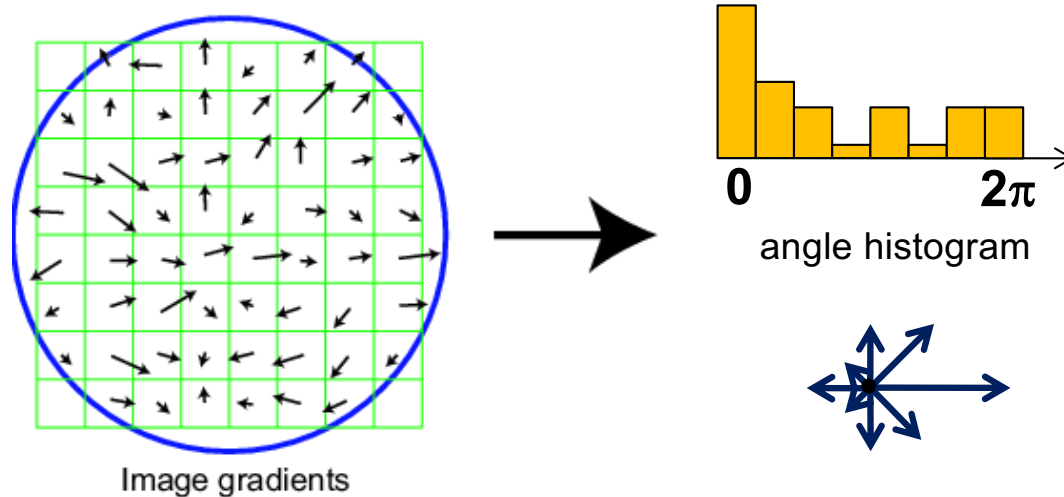




# Scale Invariant Feature Transform

Basic idea:

- Take 16x16 square window around detected feature
- Compute gradient orientation for each pixel
- Create histogram over edge orientations weighted by magnitude
- That's your feature descriptor!

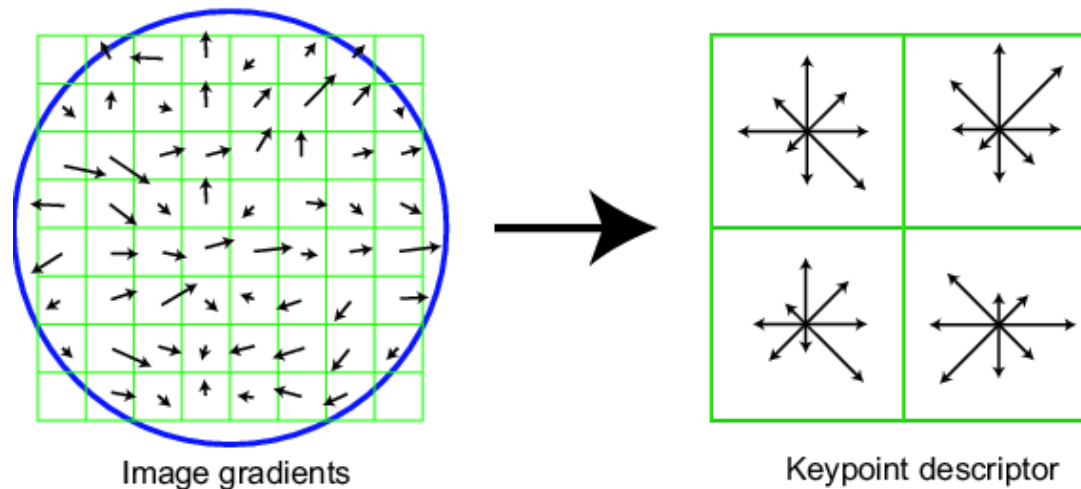


Adapted from L. Zitnick, D. Lowe










# Scale Invariant Feature Transform

## Full version

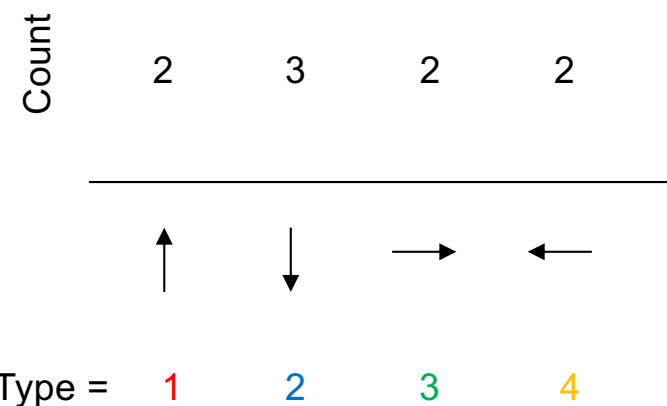
- Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below)
- Quantize the gradient orientations i.e. snap each gradient to one of 8 angles
- Each gradient contributes not just 1, but  $\text{magnitude}(\text{gradient})$  to the histogram, i.e. stronger gradients contribute more
- $16 \text{ cells} * 8 \text{ orientations} = 128 \text{ dimensional descriptor for each detected feature}$



# Scale Invariant Feature Transform

 1	 3	 1
 2	 3	 2
 4	 2	 4










Uniform weight (ignore magnitude)



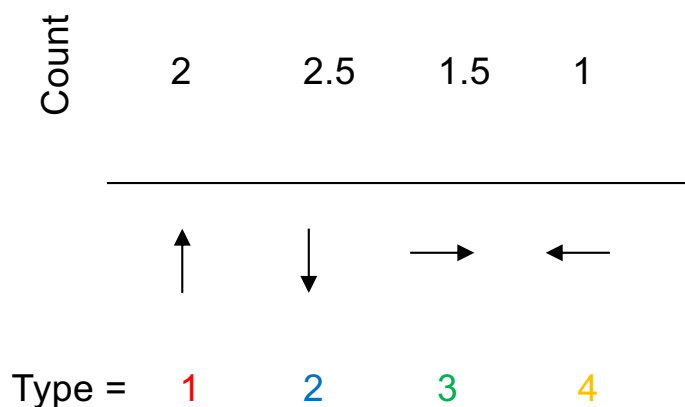
Gradients

Histogram of gradients

# Scale Invariant Feature Transform

 1	 3	 1
 2	 3	 2
 4	 2	 4

Weight contribution by magnitude  
(e.g. long = 1, short = 0.5)



Gradients

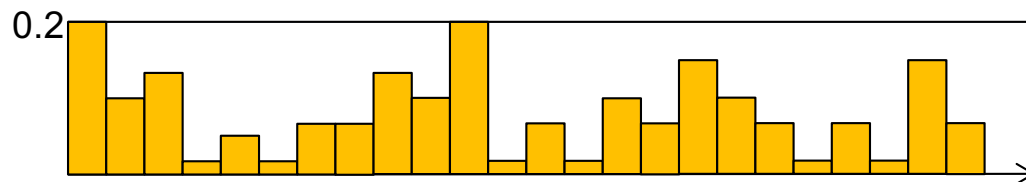
Histogram of gradients

# Scale Invariant Feature Transform

## Full version

- Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below)
- Quantize the gradient orientations i.e. snap each gradient to one of 8 angles
- Each gradient contributes not just 1, but magnitude(gradient) to the histogram, i.e. stronger gradients contribute more
- 16 cells \* 8 orientations = 128 dimensional descriptor for each detected feature
- Normalize + clip (threshold normalize to 0.2) + normalize the descriptor
- We want:

$$\sum_i d_i = 1 \quad \text{such that: } d_i < 0.2$$



# Making descriptor rotation invariant

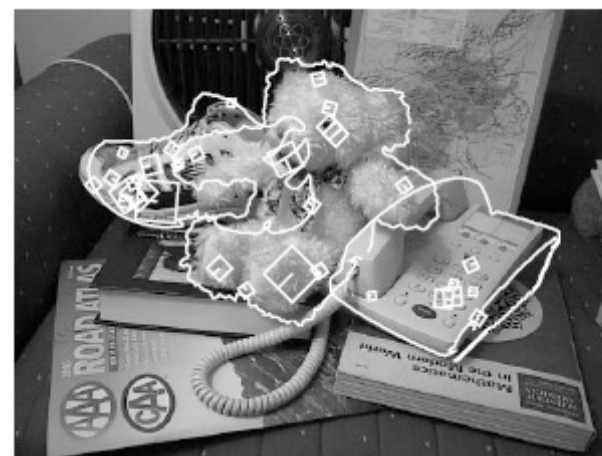


- Rotate patch according to its dominant gradient orientation
- This puts the patches into a canonical orientation

# SIFT is robust

- Can handle **changes in viewpoint**
  - Up to about 60 degree out of plane rotation
- Can handle significant **changes in illumination**
  - Sometimes even day vs. night
- **Fast and efficient**—can run in real time
- Can be made to work without feature detection, resulting in “**dense SIFT**” (more points means robustness to occlusion)
- One commonly used implementation
  - <http://www.vlfeat.org/overview/sift.html>

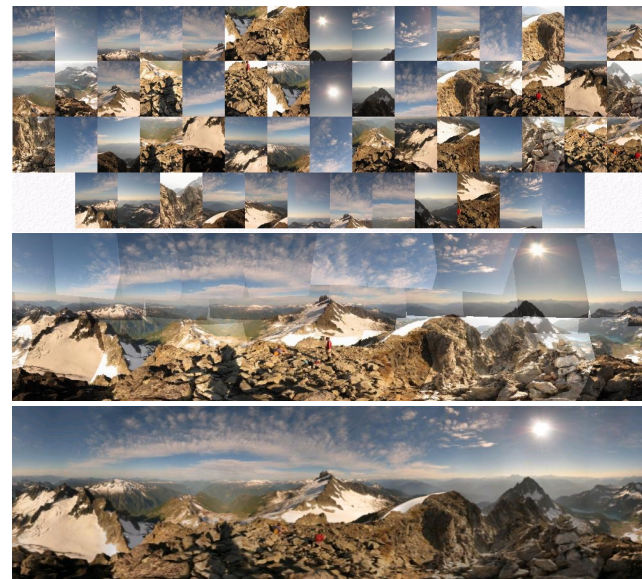
# Examples of using SIFT





# Applications of local invariant features

- Object recognition
- Indexing and retrieval
- Robot navigation
- 3D reconstruction
- Motion tracking
- Image alignment
- Panoramas and mosaics
- ...



<http://www.cs.ubc.ca/~mbrown/autostitch/autostitch.html>

## Lab 3: SIFT

Duration: 10 min



To join, go to: [ahaslides.com/OGYZC](https://ahaslides.com/OGYZC) 



**Please, select two images and draw SIFT matches among these images. Then, upload your result.**

^ Get Feedback



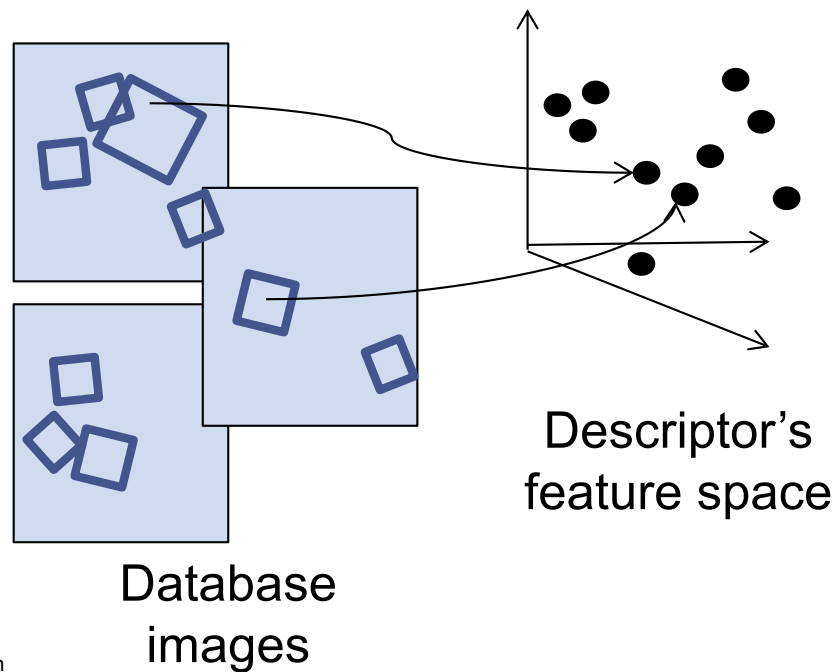
0 0/100 

## Plan for this lecture

- Feature detection / keypoint extraction
  - Corner detection
  - Blob detection
- Feature description (of detected features)
- Matching features across images

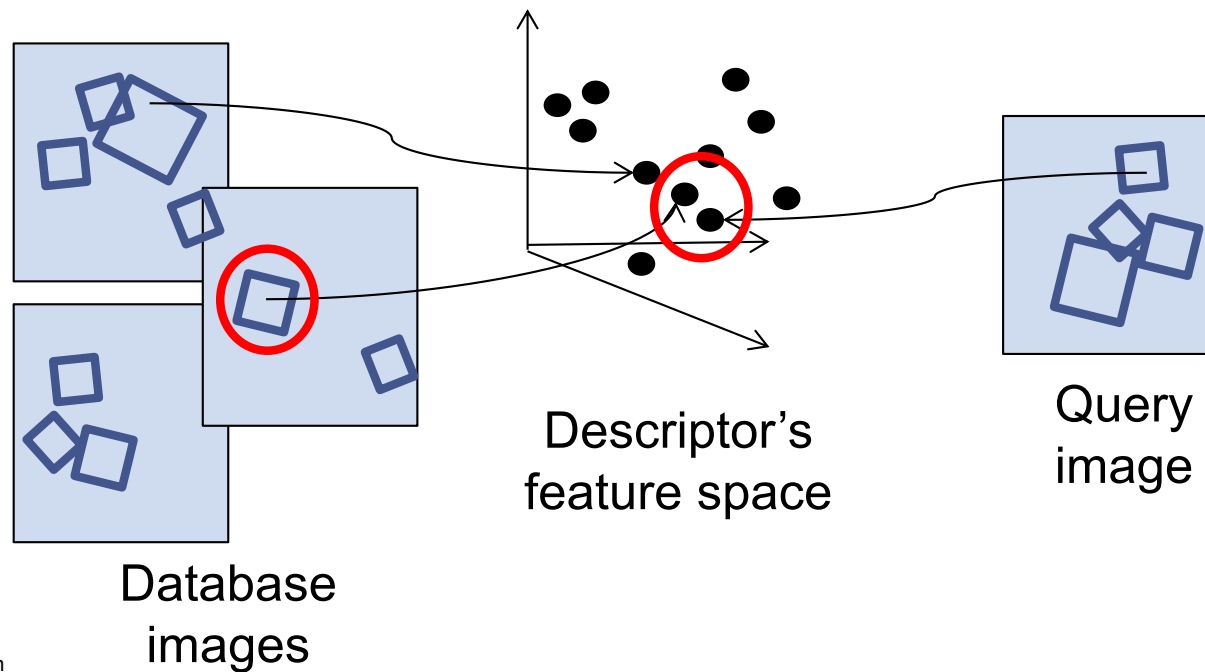
## Matching Local Features Setup

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



## Matching Local Features Setup

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



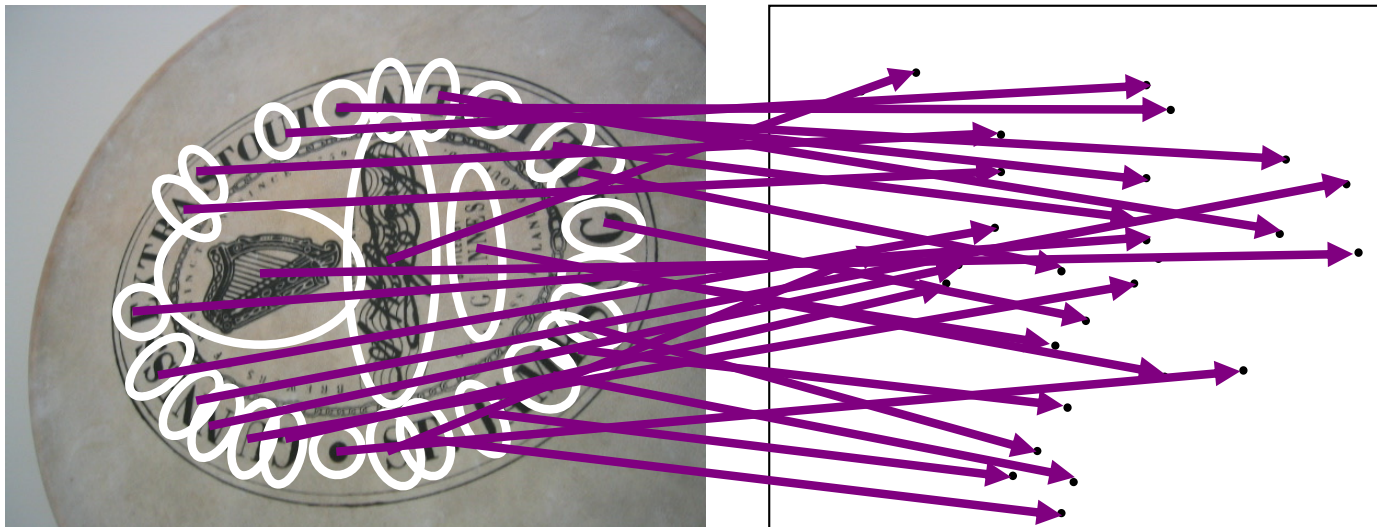
# Indexing local features

Index		
<p>*Along I-75," From Detroit to Florida; <i>inside back cover</i></p> <p>*Drive I-95," From Boston to Florida; <i>inside back cover</i></p> <p>1929 Spanish Trail Roadway; 101-102,104</p> <p>511 Traffic Information; 83</p> <p>A1A (Barrier Isl) - I-95 Access; 86</p> <p>AAA (and CAA); 83</p> <p>AAA National Office; 88</p> <p>Abbreviations,</p> <p>Colored 25 mile Maps; cover</p> <p>Exit Services; 196</p> <p>Travelogue; 85</p> <p>Africa; 177</p> <p>Agricultural Inspection Stns; 126</p> <p>Ah-Tah-Thi-Ki Museum; 160</p> <p>Air Conditioning, First; 112</p> <p>Alabama; 124</p> <p>Alachua; 132</p> <p>County; 131</p> <p>Alafia River; 143</p> <p>Alapaha, Name; 126</p> <p>Alfred B MacIay Gardens; 106</p> <p>Alligator Alley; 154-155</p> <p>Alligator Farm, St Augustine; 169</p> <p>Alligator Hole (definition); 157</p> <p>Alligator, Buddy; 155</p> <p>Alligators; 100,135,138,147,156</p> <p>Anastasia Island; 170</p> <p>Anhaica; 108-109,146</p> <p>Apalachicola River; 112</p> <p>Appleton Mus of Art; 136</p> <p>Aquifer; 102</p> <p>Arabian Nights; 94</p> <p>Art Museum, Ringling; 147</p> <p>Aruba Beach Cafe; 183</p> <p>Aucilla River Project; 106</p> <p>Babcock-Web WMA; 151</p> <p>Bahia Mar Marina; 184</p> <p>Baker County; 99</p> <p>Barfoot Mailmen; 182</p> <p>Barge Canal; 137</p> <p>Bee Line Expy; 80</p> <p>Belz Outlet Mall; 89</p> <p>Bernard Castro; 136</p> <p>Big "I"; 165</p> <p>Big Cypress; 155,158</p> <p>Big Foot Monster; 105</p> <p>Billie Swamp Safari; 160</p> <p>Blackwater River SP; 117</p> <p>Blue Angels</p> <p>A4-C Skyhawk; 117</p> <p>Atrium; 121</p> <p>Blue Springs SP; 87</p>	<p>Butterfly Center, McGuire; 134</p> <p>CAA (see AAA)</p> <p>COC, The; 111,113,115,135,142</p> <p>Ca d'Zan; 147</p> <p>Caloosahatchee River; 152</p> <p>Name; 150</p> <p>Canaveral Natl Seashore; 173</p> <p>Cannon Creek Airpark; 130</p> <p>Canopy Road; 106,169</p> <p>Cape Canaveral; 174</p> <p>Castillo San Marcos; 169</p> <p>Cave Diving; 131</p> <p>Cayo Costa, Name; 150</p> <p>Celebration; 93</p> <p>Charlotte County; 149</p> <p>Charlotte Harbor; 150</p> <p>Chautauqua; 116</p> <p>Chicley; 114</p> <p>Name; 115</p> <p>Choctawhatchee, Name; 115</p> <p>Circus Museum, Ringling; 147</p> <p>Citrus; 88,97,130,136,140,180</p> <p>CityPlace, W Palm Beach; 180</p> <p>City Maps,</p> <p>Ft Lauderdale Expwys; 194-195</p> <p>Jacksonville; 163</p> <p>Kissimmee Expwys; 192-193</p> <p>Miami Expressways; 194-195</p> <p>Orlando Expressways; 192-193</p> <p>Pensacola; 26</p> <p>Tallahassee; 191</p> <p>Tampa-St. Petersburg; 63</p> <p>St. Augustine; 191</p> <p>Civil War; 100,108,127,138,141</p> <p>Cleaver Marine Aquarium; 187</p> <p>Collier County; 154</p> <p>Collier, Barron; 152</p> <p>Colonial Spanish Quarters; 168</p> <p>Columbia County; 101,128</p> <p>Coquina Building Material; 165</p> <p>Corkscrew Swamp, Name; 154</p> <p>Cowboys; 95</p> <p>Crab Trap II; 144</p> <p>Cracker, Florida; 88,95,132</p> <p>Crosstown Expy; 11,35,98,143</p> <p>Cuban Bread; 184</p> <p>Dade Battlefield; 140</p> <p>Dade, Maj. Francis; 139-140,161</p> <p>Dania Beach Hurricane; 184</p> <p>Daniel Boone, Florida Walk; 117</p> <p>Daytona Beach; 172-173</p> <p>De Land; 87</p> <p>De Soto, Hernando,</p> <p>Anhaica; 108-109,146</p> <p>County; 149</p>	<p>Driving Lanes; 85</p> <p>Duval County; 163</p> <p>Eau Gallie; 175</p> <p>Edison, Thomas; 152</p> <p>Eglin AFB; 116-118</p> <p>Eight Reale; 176</p> <p>Ellenton; 144-145</p> <p>Emanuel Point Wreck; 120</p> <p>Emergency Callboxes; 83</p> <p>Epiphytes; 142,148,157,159</p> <p>Escambia Bay; 119</p> <p>Bridge (I-10); 119</p> <p>County; 120</p> <p>Estero; 153</p> <p>Everglade; 90,95,139-140,154-160</p> <p>Draining of; 156,181</p> <p>Wildlife MA; 160</p> <p>Wonder Gardens; 154</p> <p>Falling Waters SP; 115</p> <p>Fantasy of Flight; 95</p> <p>Fayer Dykes SP; 171</p> <p>Fires, Forest; 166</p> <p>Fires, Prescribed; 148</p> <p>Fisherman's Village; 151</p> <p>Flagler County; 171</p> <p>Flagler, Henry; 97,165,167,171</p> <p>Florida Aquarium; 186</p> <p>Florida,</p> <p>12,000 years ago; 187</p> <p>Cavern SP; 114</p> <p>Map of all Expressways; 2-3</p> <p>Mus of Natural History; 134</p> <p>National Cemetery; 141</p> <p>Part of Africa; 177</p> <p>Platform; 167</p> <p>Sheriff's Boys Camp; 126</p> <p>Sports Hall of Fame; 130</p> <p>Sun 'n Fun Museum; 97</p> <p>Supreme Court; 107</p> <p>Florida's Turnpike (FTP); 178,189</p> <p>25 mile Strip Maps; 66</p> <p>Administration; 189</p> <p>Coin System; 190</p> <p>Exit Services; 189</p> <p>HEFT; 76,161,190</p> <p>History; 189</p> <p>Names; 189</p> <p>Service Plazas; 190</p> <p>Spur SR91; 76</p> <p>Ticket System; 190</p> <p>Toll Plazas; 190</p> <p>Ford, Henry; 152</p> <p>Fort Barrancas; 122</p> <p>Buried Alive; 123</p> <p>Fort Caroline; 164</p>

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

## Visual Words: main idea

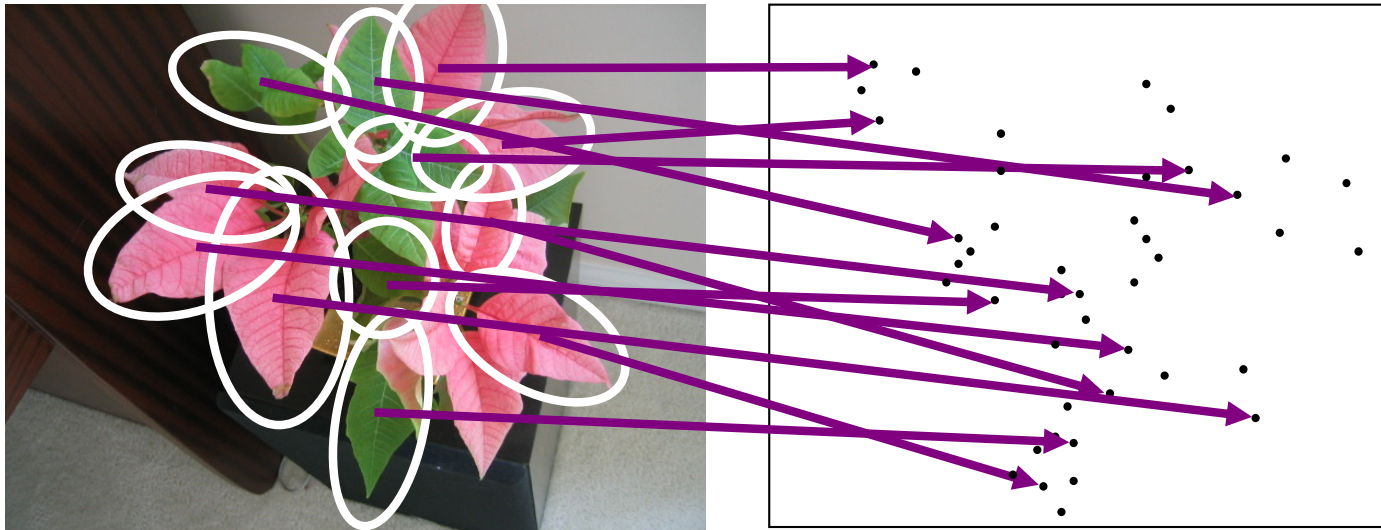
- Extract some local features from a number of images ...



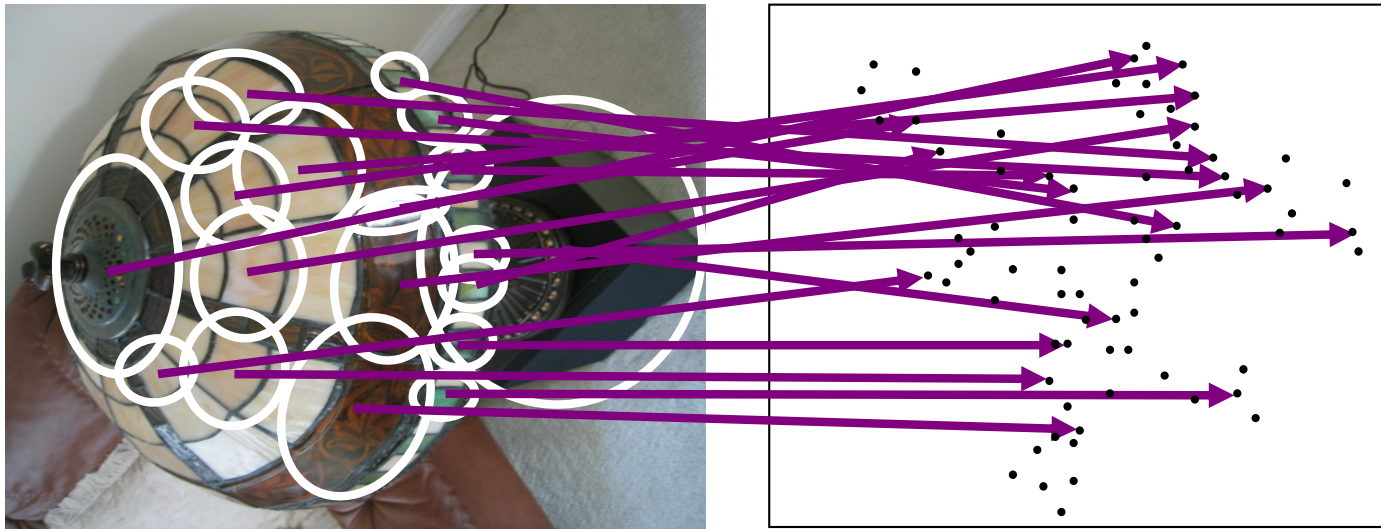
e.g., SIFT descriptor space: each point is 128-dimensional



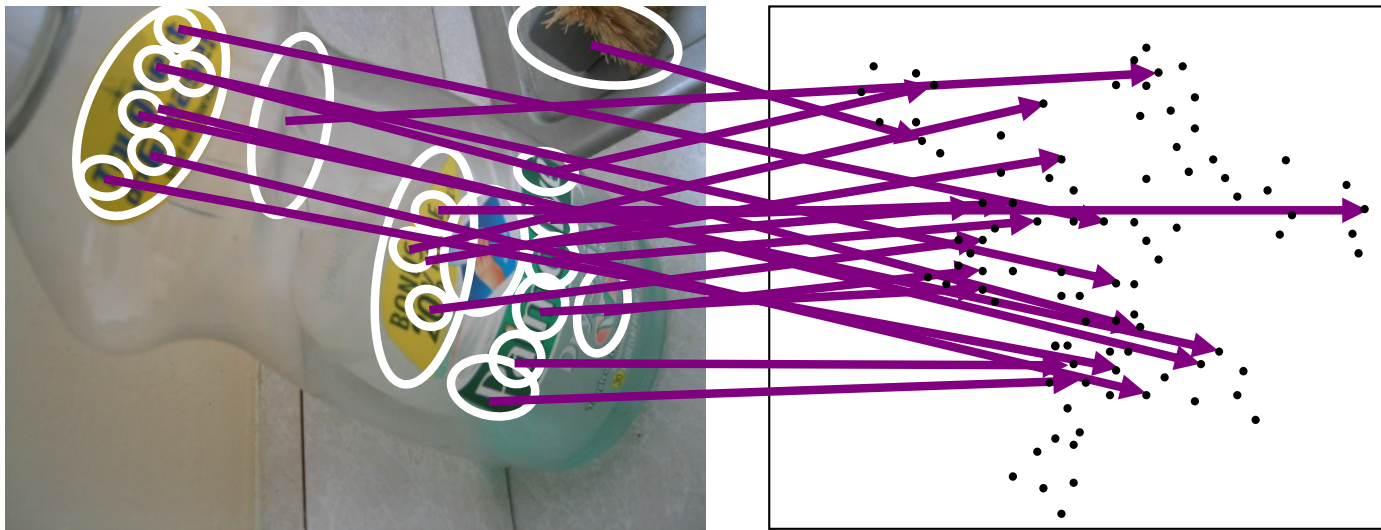
## Visual Words: main idea

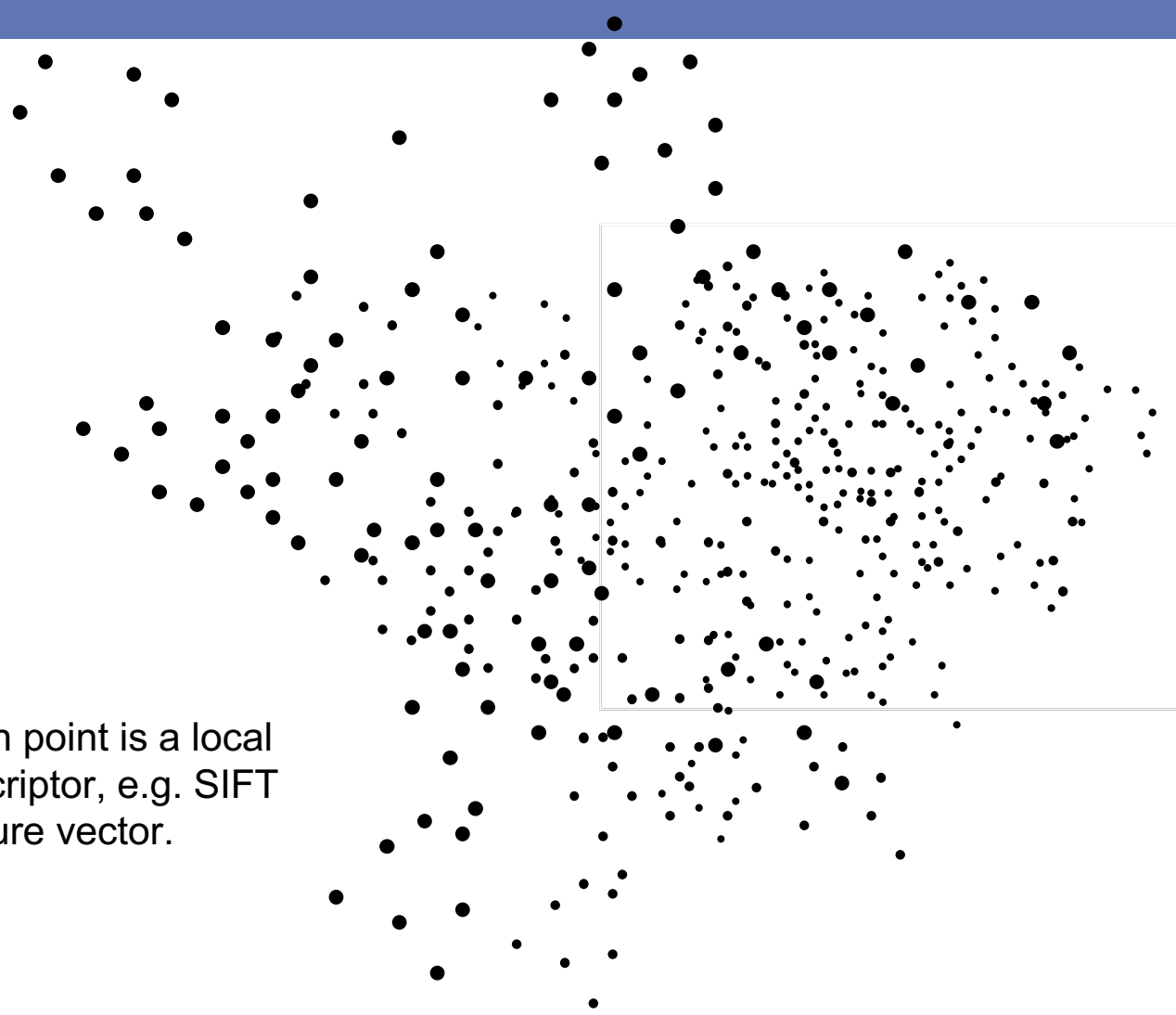


## Visual Words: main idea

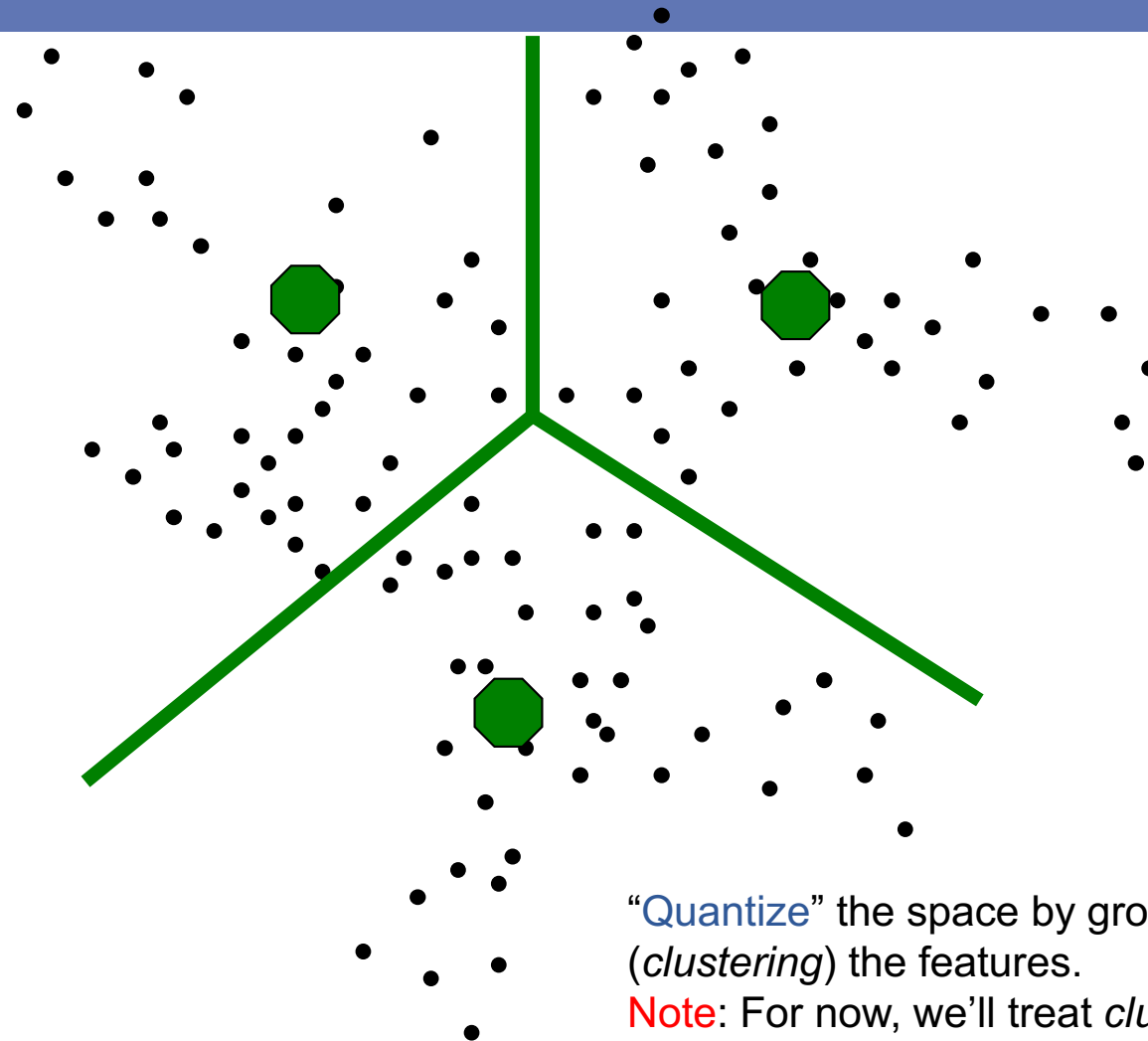


## Visual Words: main idea





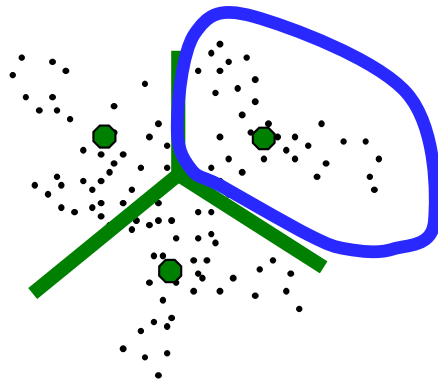
Each point is a local  
descriptor, e.g. SIFT  
feature vector.



“Quantize” the space by grouping  
(*clustering*) the features.  
**Note:** For now, we’ll treat *clustering*  
as a black box.

# Visual Words

- Patches on the right = regions used to compute SIFT
- Each group of patches belongs to the same “visual word”



Adapted from K. Grauman

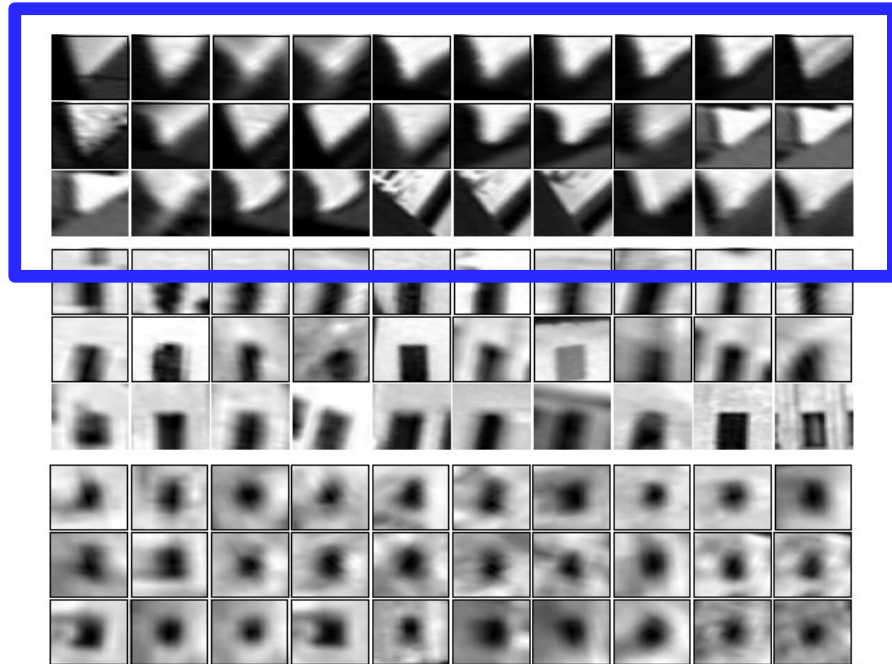
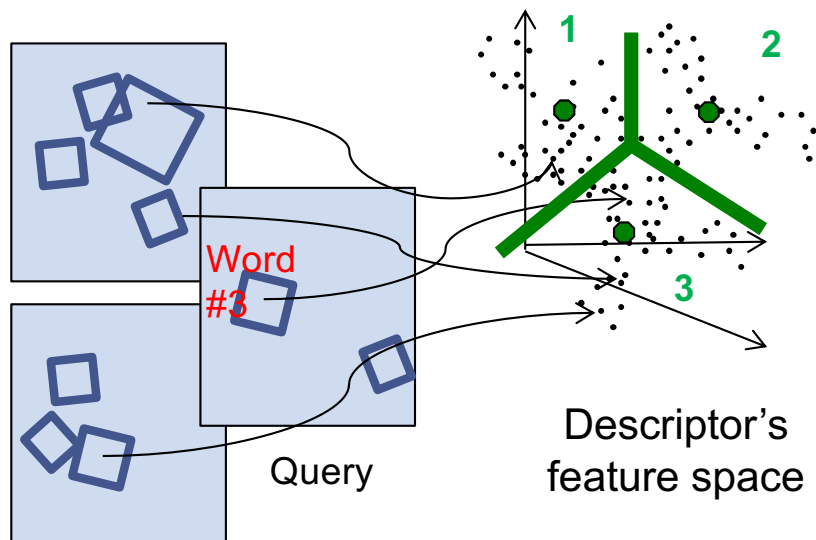


Figure from Sivic & Zisserman, ICCV 2003

# Visual Words for Indexing

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



Adapted from K. Grauman

- Each cluster has a **center**
- To determine **which word to assign to new image region** (e.q. query), **find closest cluster center**
- To compare features**: Only compare query to others in same **cluster**, or just compare word IDs
- To compare images**: see next few slides

# How to describe documents with words

Of all the sensory impressions proceeding to the brain, the visual experiences are the dominant ones. Our perception of the world around us is based essentially on the messages that reach our eyes. For a long time, the point of the retinal image was thought to be the starting point to visual perception. The visual cortex is the part of the brain upon which the visual image is projected. Wiesel and Hubel have been able to demonstrate that the message about the image falling on the retina undergoes a step-wise analysis by a system of nerve cells stored in columns. In this system each cell has its specific function and is responsible for a specific detail in the pattern of the retinal image.

**sensory, brain,  
visual, perception,  
retinal, cerebral cortex,  
eye, cell, optical  
nerve, image  
Hubel, Wiesel**

China is forecasting a trade surplus of \$90bn (£51bn) to \$100bn this year, a threefold increase on 2004's \$32bn. The Commerce Ministry said the surplus would be created by a predicted 30% increase in exports to \$750bn, compared with imports to \$660bn. Further analysis showed that China's trade surplus was a deliberate policy to encourage exports and agreed to a trade agreement with the US. The yuan is the domestic currency of the government. China also needs to increase demand for its goods and services in the country. China's trade surplus is a result of the yuan against the US dollar and permitted it to trade within a narrow band, but the US wants the yuan to be allowed to trade freely. However, Beijing has made it clear that it will take its time and tread carefully before allowing the yuan to rise further in value.

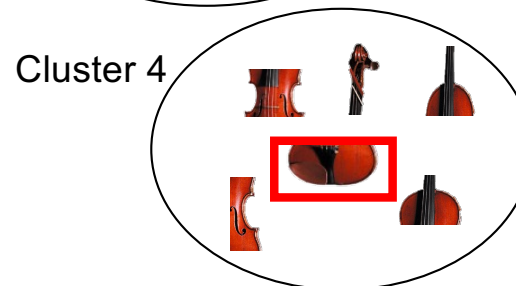
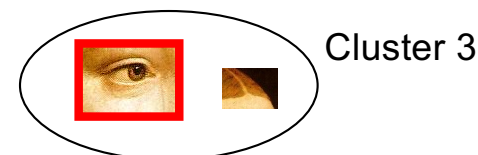
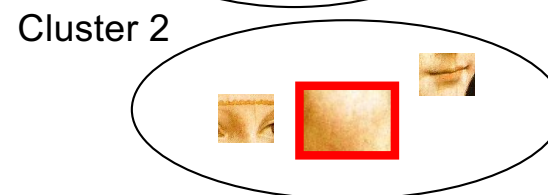
**China, trade,  
surplus, commerce,  
exports, imports, US,  
yuan, bank, domestic,  
foreign, increase,  
trade, value**



# Describing images with visual words

- Summarize entire image based on its distribution (histogram) of word occurrences
- Analogous to bag of words representation commonly used for documents

Feature patches:



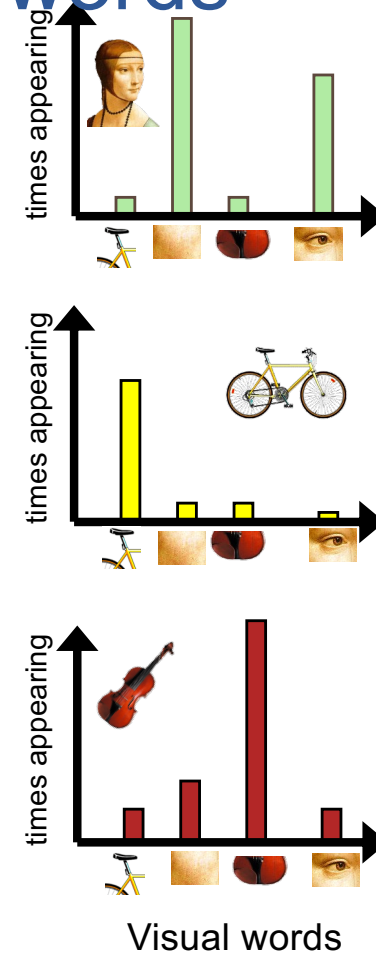
# Describing images with visual words

- Summarize entire image based on its distribution (histogram) of word occurrences
- Analogous to bag of words representation commonly used for documents

Feature patches:

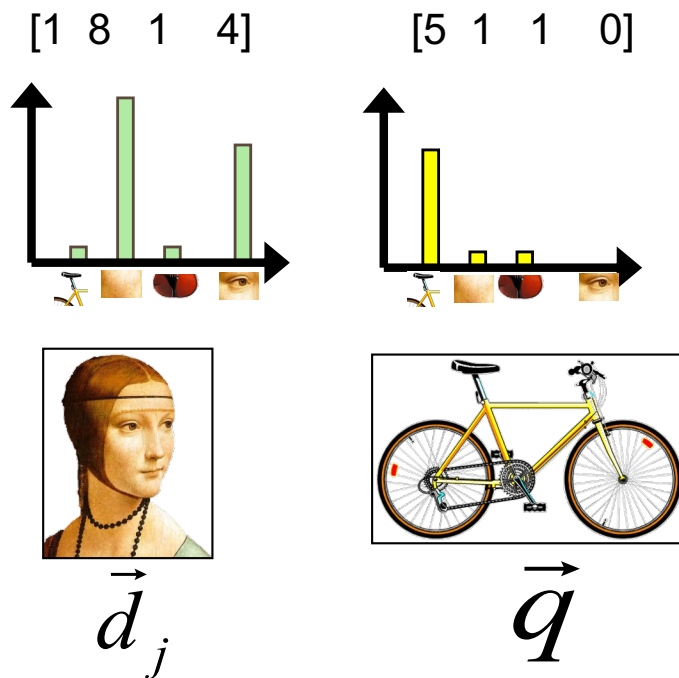


K. Grauman



# Comparing bags of words

- Similarity of images measured as normalized scalar product between their word occurrence counts
- Can be used to rank results (nearest neighbors of query)



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

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

for vocabulary of  $V$   
words

## Bags of words: pros and cons

- + Flexible to geometry / deformations / viewpoint
- + Compact summary of image content
- Basic model ignores geometry – verify afterwards
- What is the optimal vocabulary size?
- Background and foreground mixed when bag covers whole image

## Summary: Inverted file index and bags of words similarity

### Offline:

- Extract features in database images, cluster them to find words = cluster centers, make index

### Online (during search):

1. Extract words in query (extract features and map each to closest cluster center)
2. Use inverted file index to find database images relevant to query
3. Rank database images by comparing word counts of query and database image

# Summary

- Keypoint detection: repeatable and distinctive
  - Corners, blobs, stable regions
  - Laplacian of Gaussian, automatic scale selection
- Descriptors: robust and selective
  - Histograms for robustness to small shifts and translations (SIFT descriptor)
- Matching: cluster and index
  - Compare images through their feature distribution

