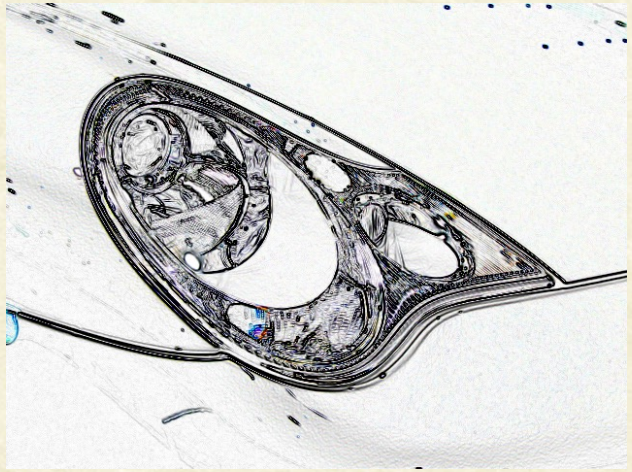
CS7.505: Computer Vision

Spring 2022: Feature Description and Matching



Anoop M. Namboodiri

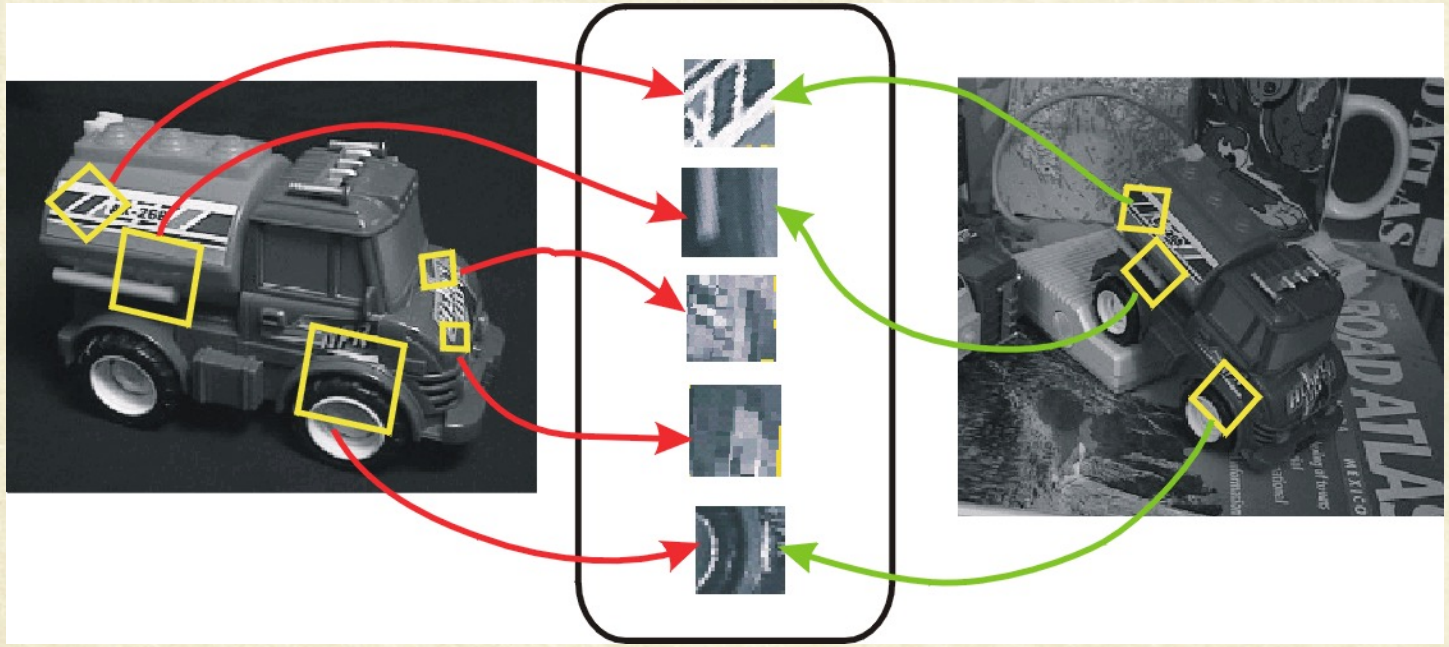
Biometrics and Secure ID Lab, CVIT,

IIIT Hyderabad

Invariant Local Features

Find features that are invariant to transformations

- Geometric invariance: Translation, rotation, scale
- Photometric invariance: Brightness, exposure, …



Feature Descriptors

Multi-Scale Oriented Patches (MOPS)

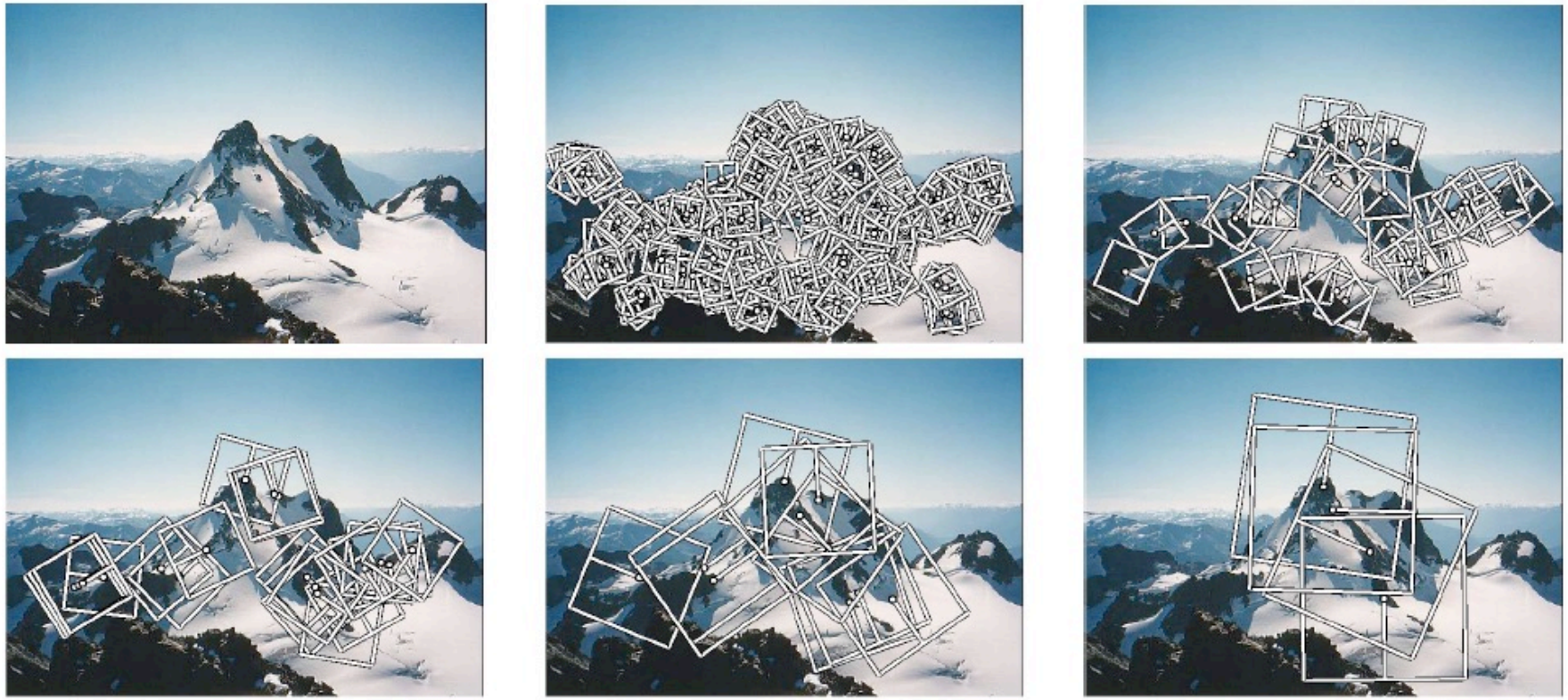
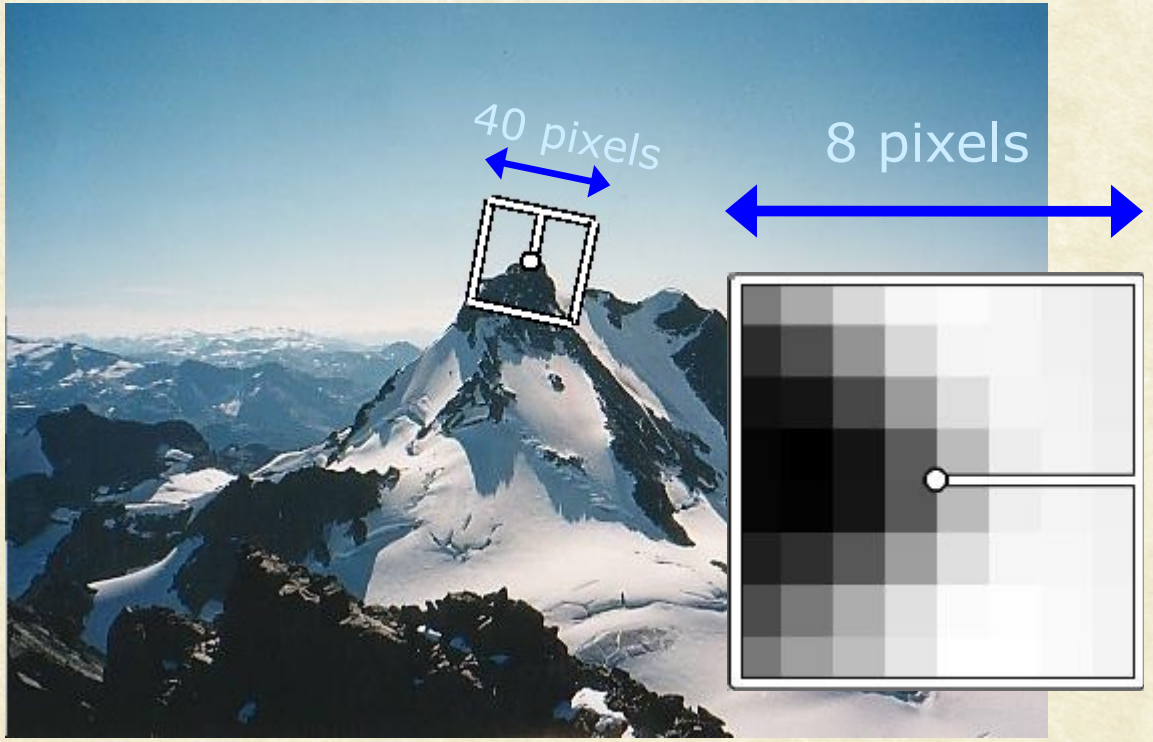


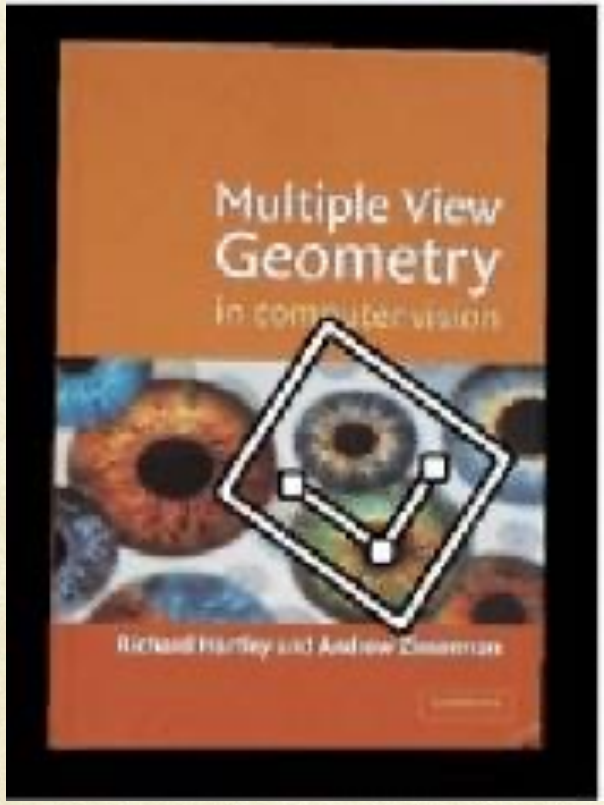
Figure 1. Multi-scale Oriented Patches (MOPS) extracted at five pyramid levels from one of the Matier images. The boxes show the feature orientation and the region from which the descriptor vector is sampled.

MOPS Descriptor Vector

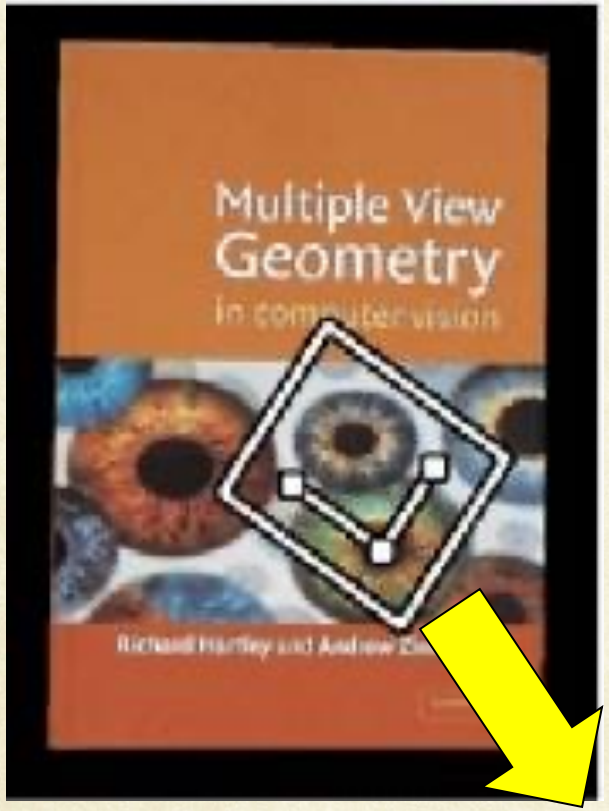
- 8x8 oriented patch
  - Sampled at 5 x scale
- Bias/gain normalisation: $I' = (I - \mu)/\sigma$

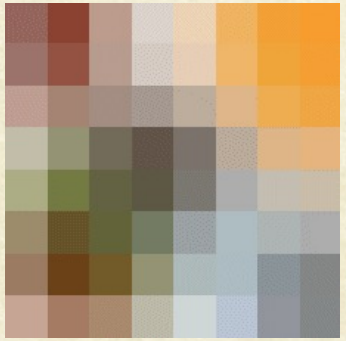


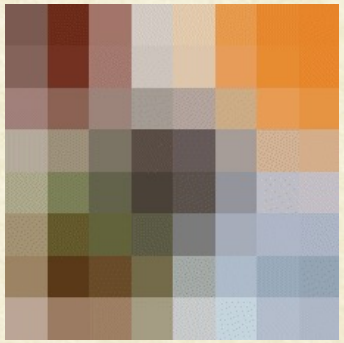
Feature Description



Feature Description







Scale Invariant Feature Transform (SIFT)

Slides by Tom Duerig

Types of Invariances

- Illumination

Types of Invariances

- Illumination
- Scale

Types of Invariances

- Illumination
- Scale
- Rotation

Types of Invariances

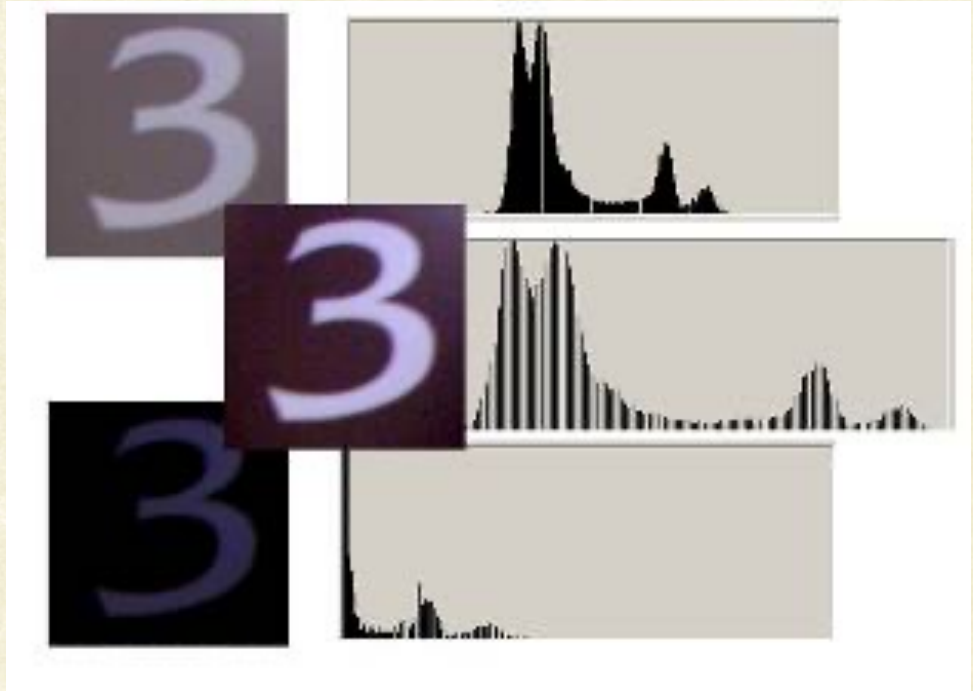
- Illumination
- Scale
- Rotation
- Affine

Types of Invariances

- Illumination
- Scale
- Rotation
- Affine
- Full Perspective

How to Achieve Illumination Invariance?

- The easy way (normalized)
- Difference based metrics (random tree, Haar, and sift)



How to achieve Scale Invariance

- Pyramids
  - Divide width and height by 2
  - Take average of 4 pixels for each pixel (or Gaussian blur)
  - Repeat until image is tiny
  - Run filter over each size image and hope its robust
- Scale Space (DOG method)

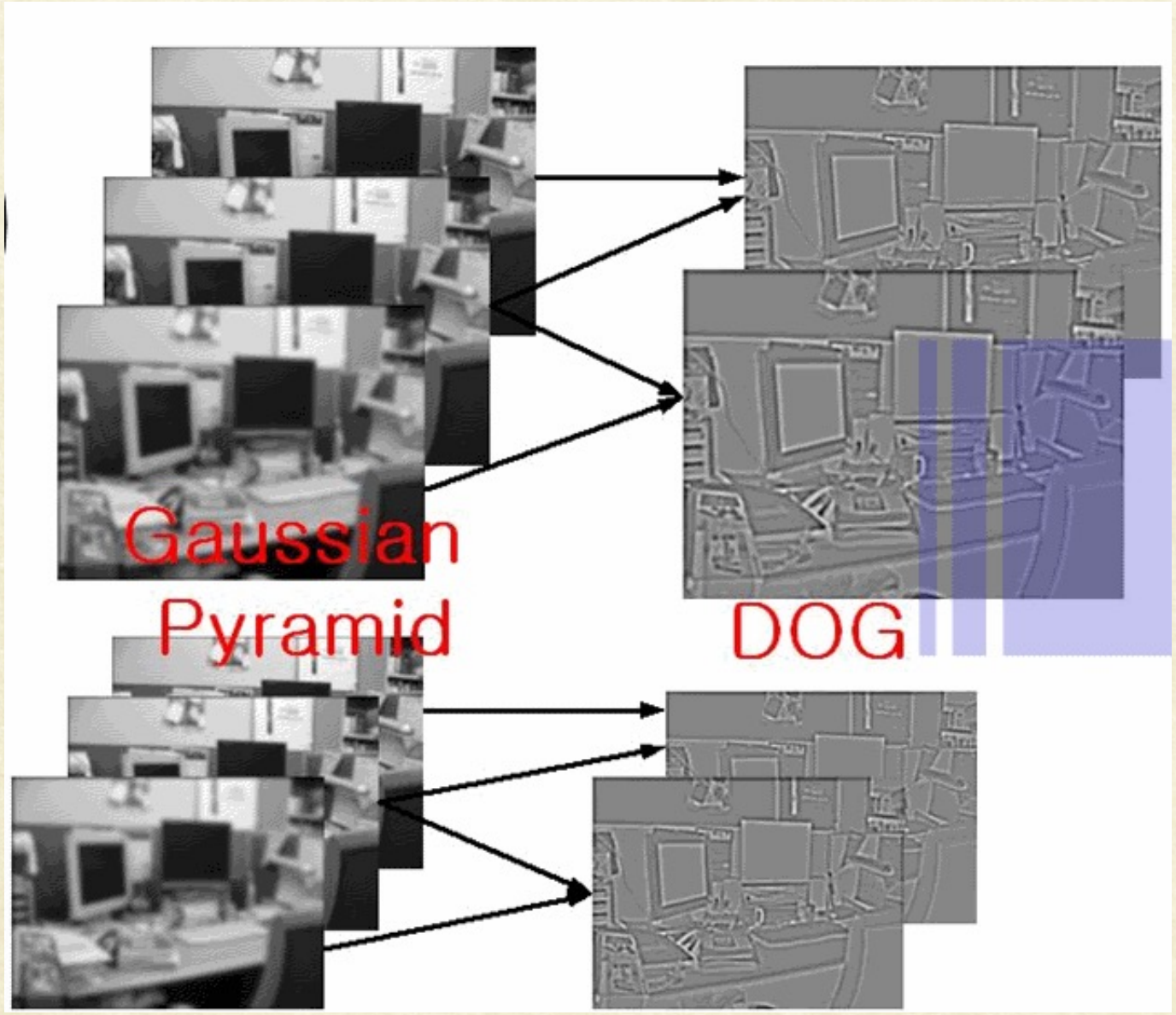
Pyramids


novelties
and
ice cream
novelties
and
ice cream
novelties
and
ice cream
novelties
and
ice cream

How to achieve Scale Invariance

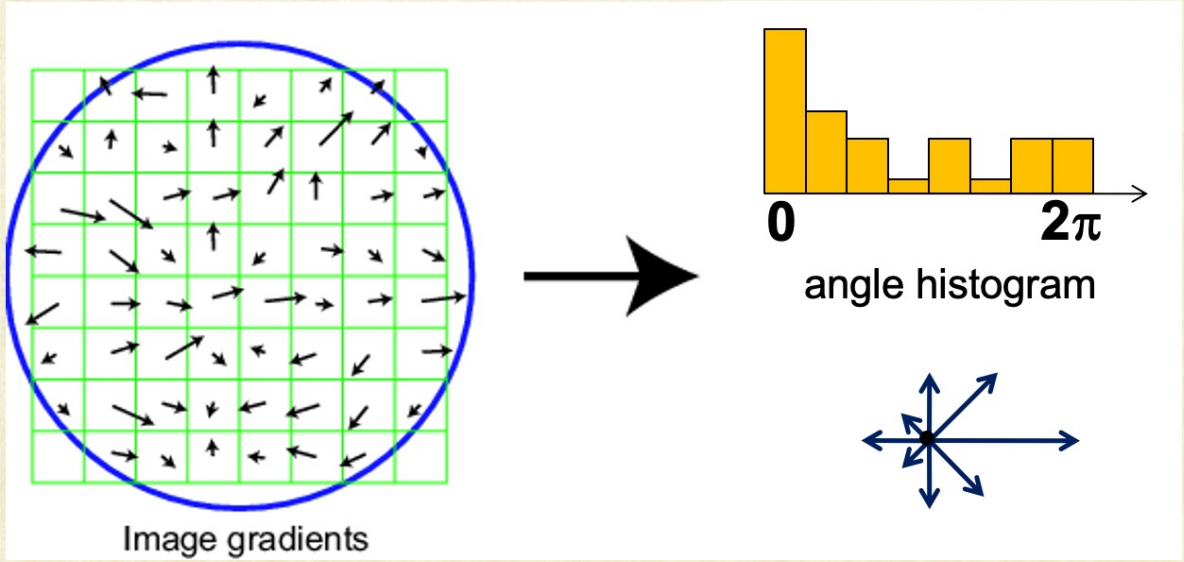
- Pyramids
- Scale Space (DOG method)
  - Pyramid but fill gaps with blurred images
  - Like having a nice linear scaling without the expense
  - Take features from differences of these images
  - If the feature is repeatably present in between Difference of Gaussians it is Scale Invariant and we should keep it.

Differences of Gaussians

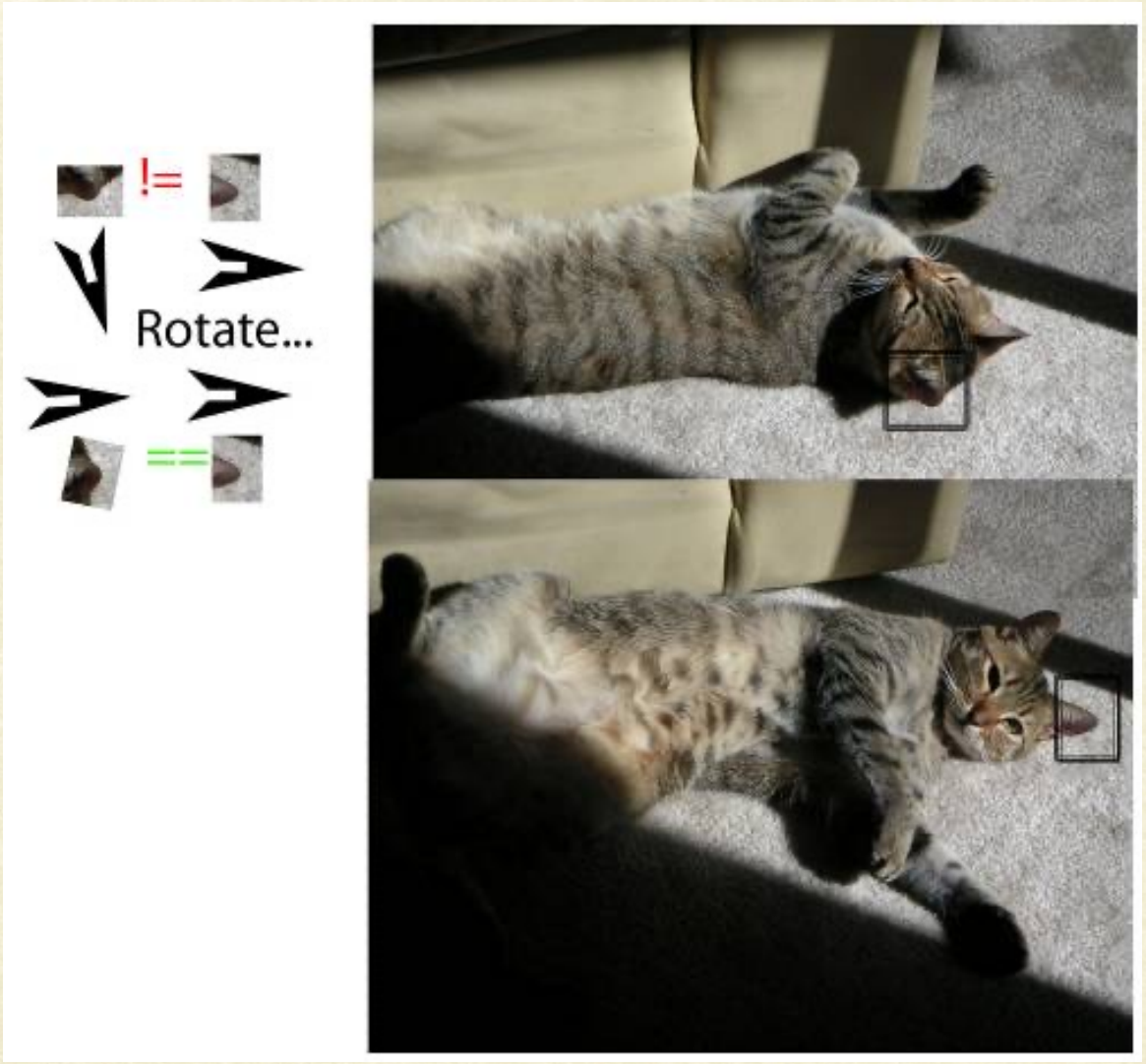


Rotation Invariance

- Rotate all features to go the same way in a determined manner
- Take histogram of Gradient directions (36 for 1 every 10 degrees)
- Rotate to most dominant (principal orientation)



Rotation Invariance

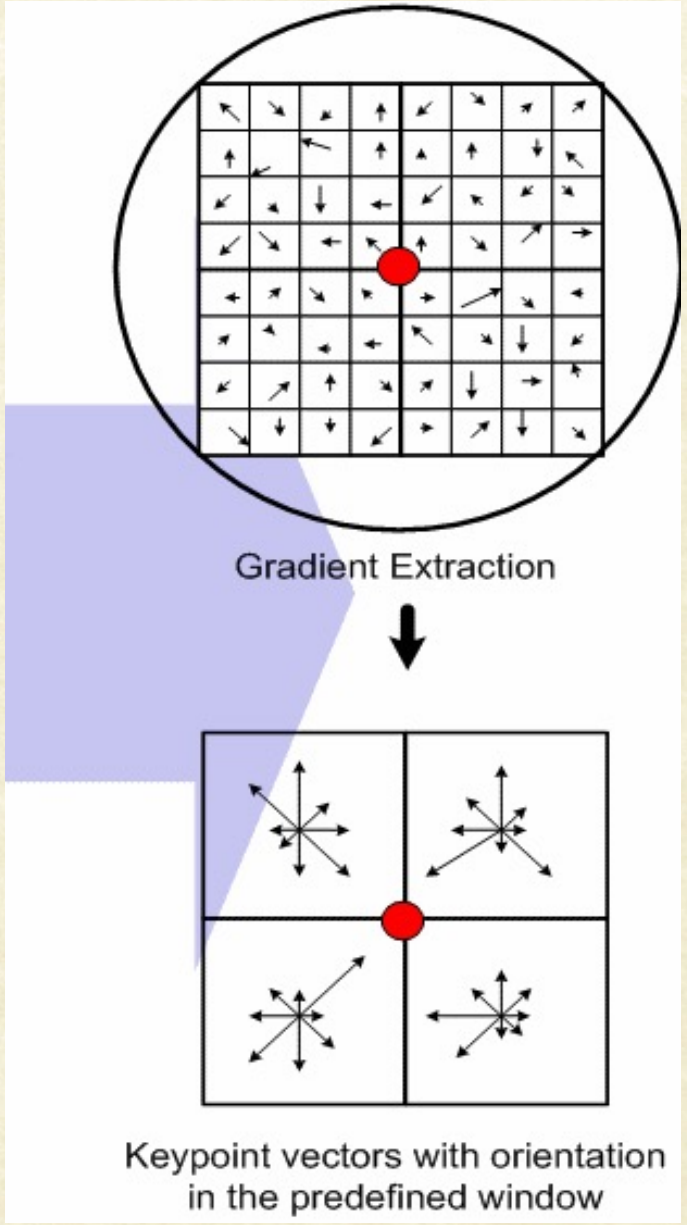


Affine Invariance

- Easy way: Warp your training and hope
- Fancy way: design your feature itself to be robust against affine transformations (SIFT method)

Actual SIFT features

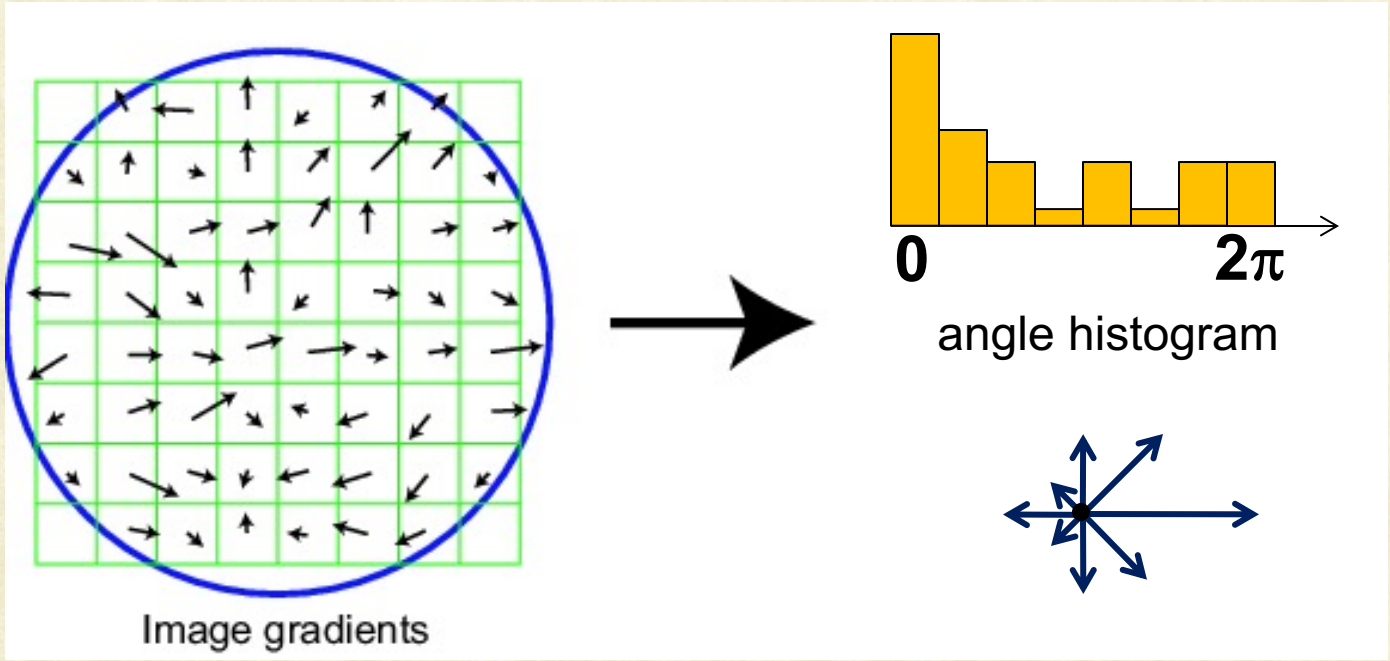
- Remember the gradient histograms we used for rotation invariance?
- Same theory, except keep N^2 histograms (4 shown, 16 used)
- Note, use weighted contributions to avoid edge nastiness



Scale Invariant Feature Transform

Basic idea:

- Take 16x16 square window around detected feature
- Compute edge orientation (angle of the gradient - 90°) for each pixel
- Throw out weak edges (threshold gradient magnitude)
- Create histogram of surviving edge orientations

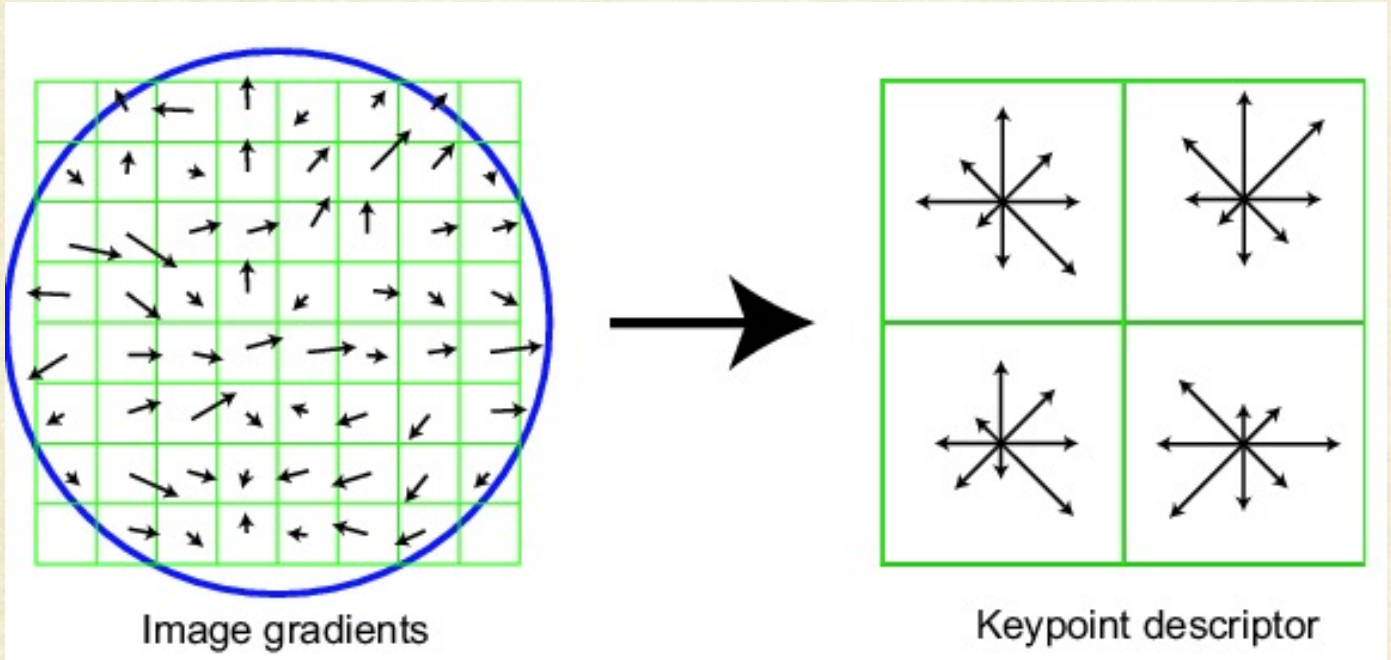


Adapted from slide by David Lowe

SIFT Descriptor

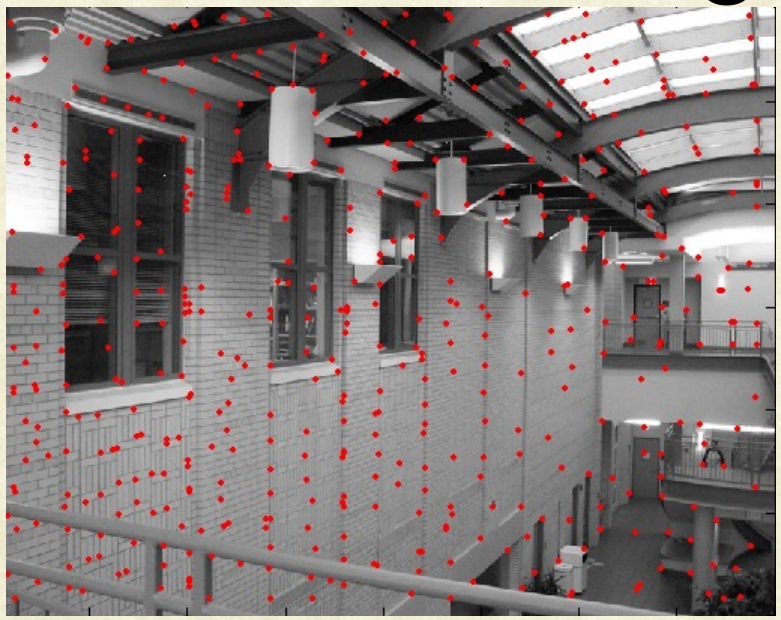
Full version

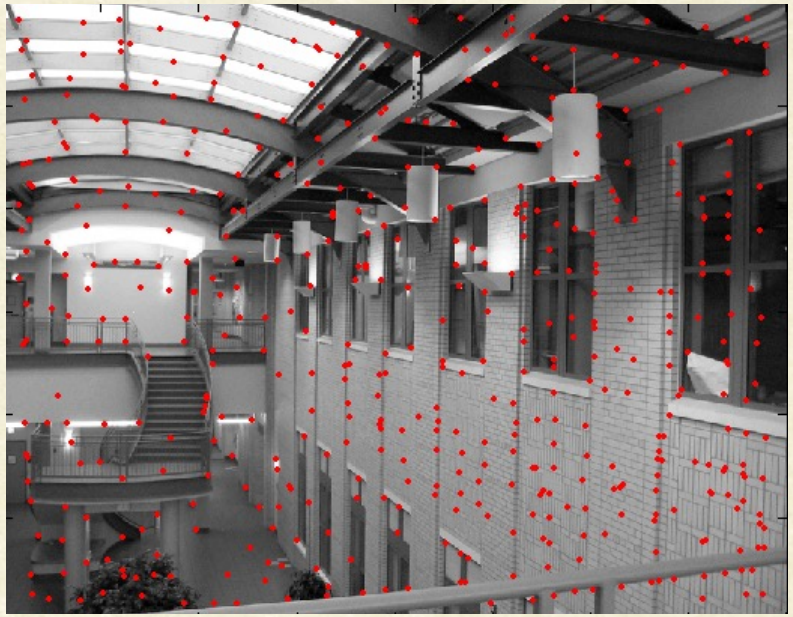
- Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below)
- Compute an orientation histogram for each cell (Use gaussian weighting)
- 16 cells * 8 orientations = 128 dimensional descriptor

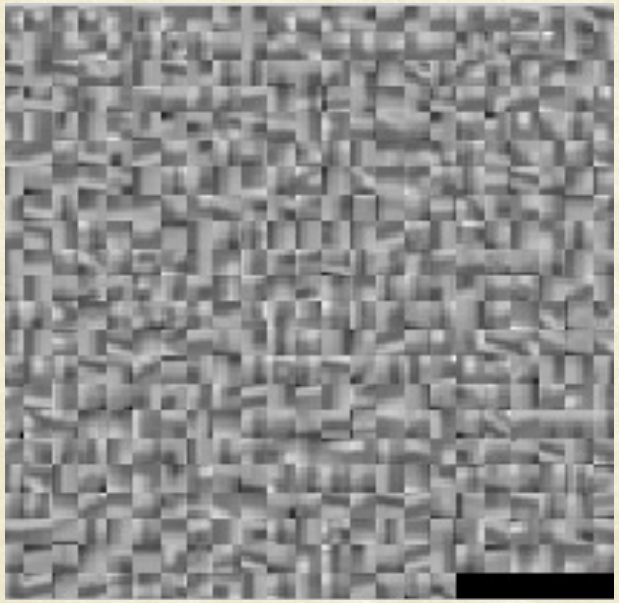


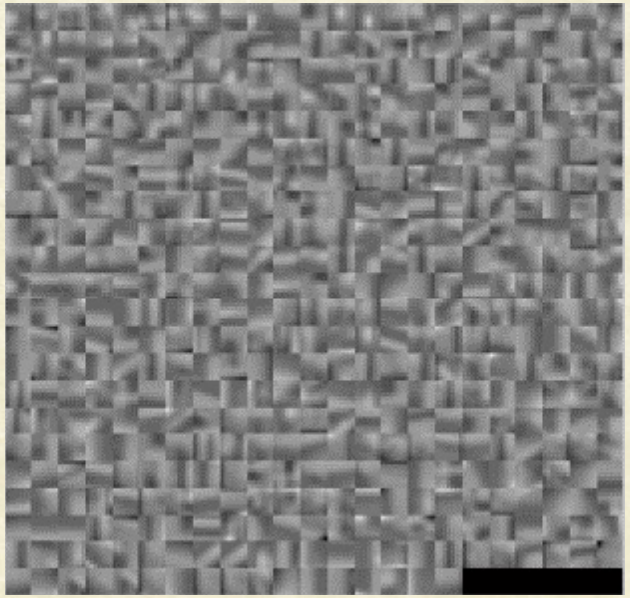
Adapted from slide by David Lowe

Feature matching

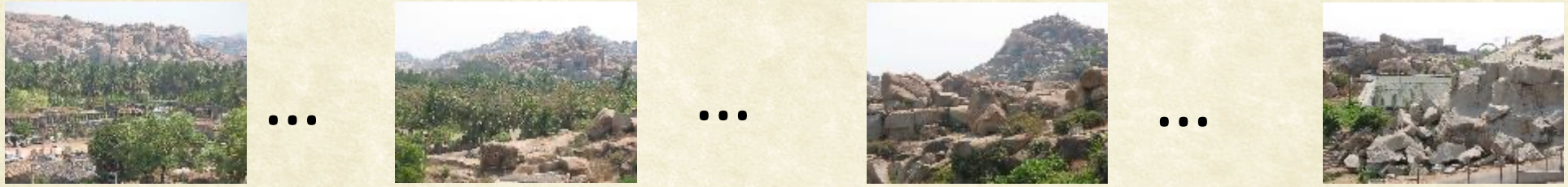








Mosaicing Example



Properties of SIFT

Extraordinarily robust matching technique

- 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 (below)
- Fast and efficient—can run in real time
- Code available: https://docs.opencv.org/3.4/da/df5/tutorial_py_sift_intro.html

Thank You