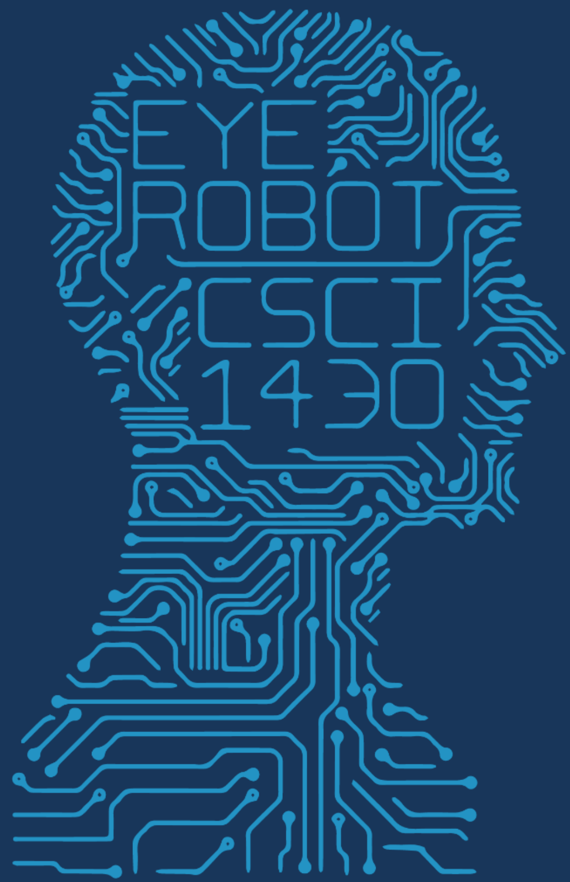
1950

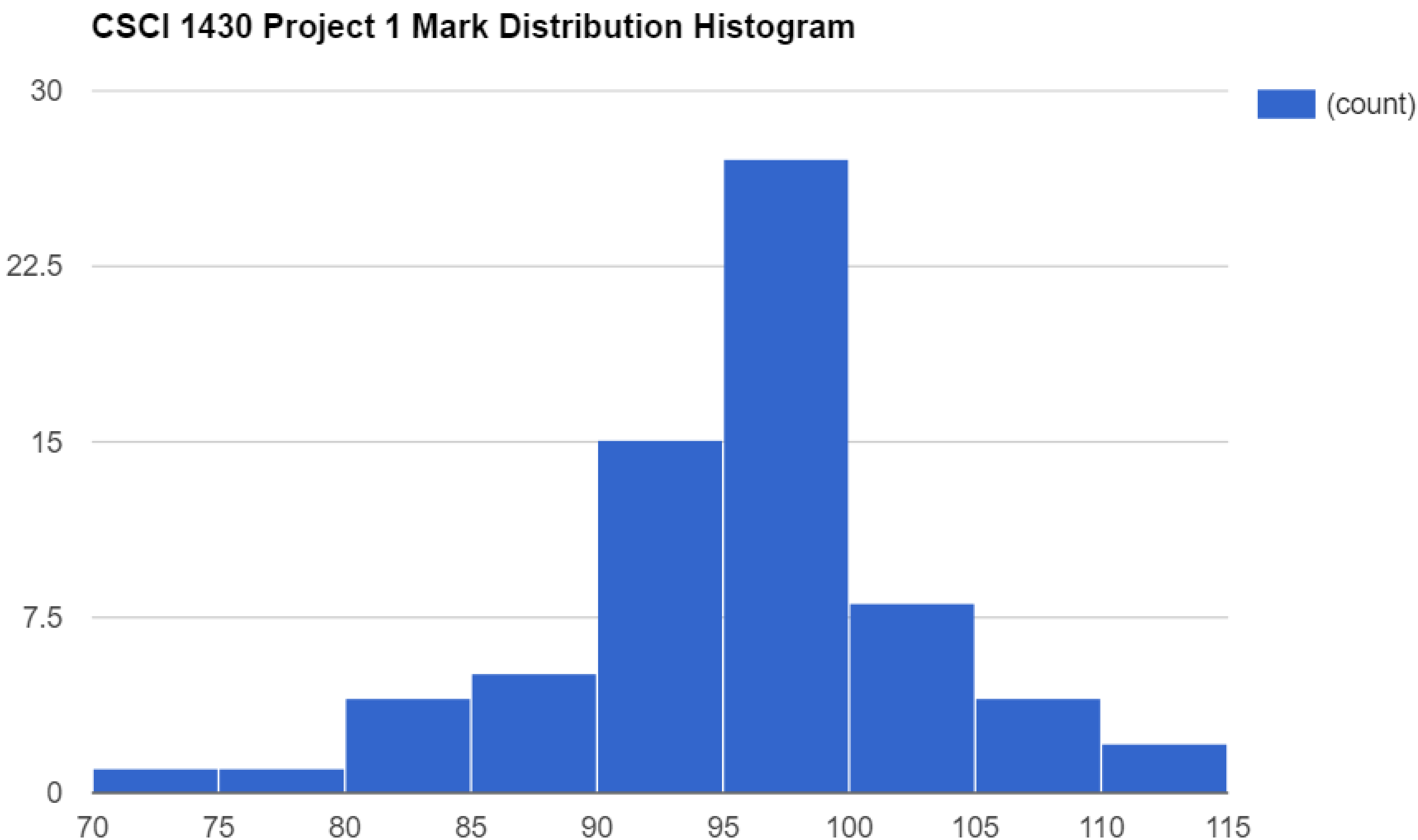
Future Vision



2017 MWF 1pm 368

Computer Vision

PROJECT 1: HYBRID IMAGES


CSCI 1430 Project 1 Mark Distribution Histogram
(count)
30
22.5
15
7.5
0
70
75
80
85
90
95
100
105
110
115

Nathaniel Parrott

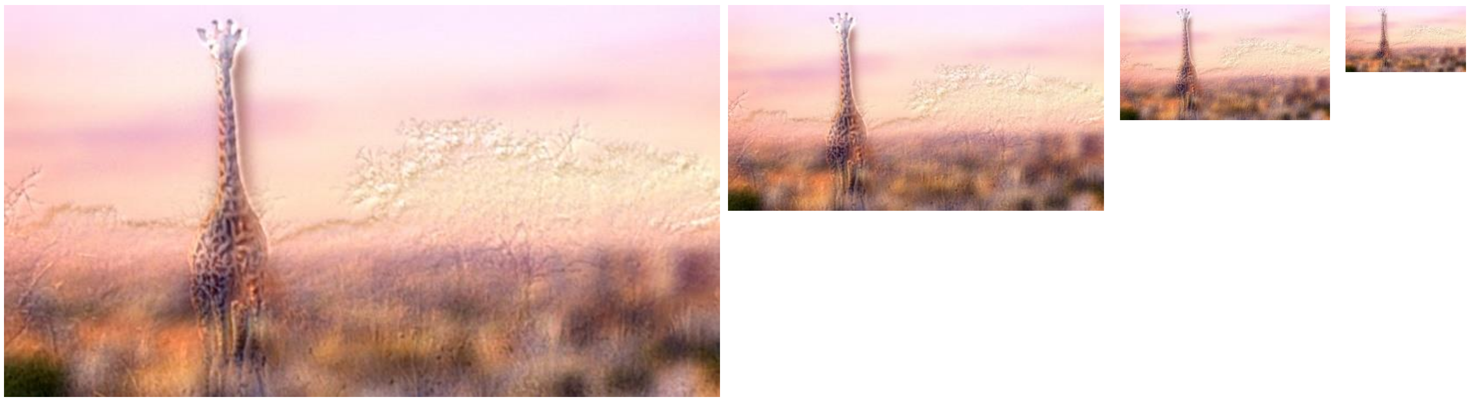
Nathaniel Parrott

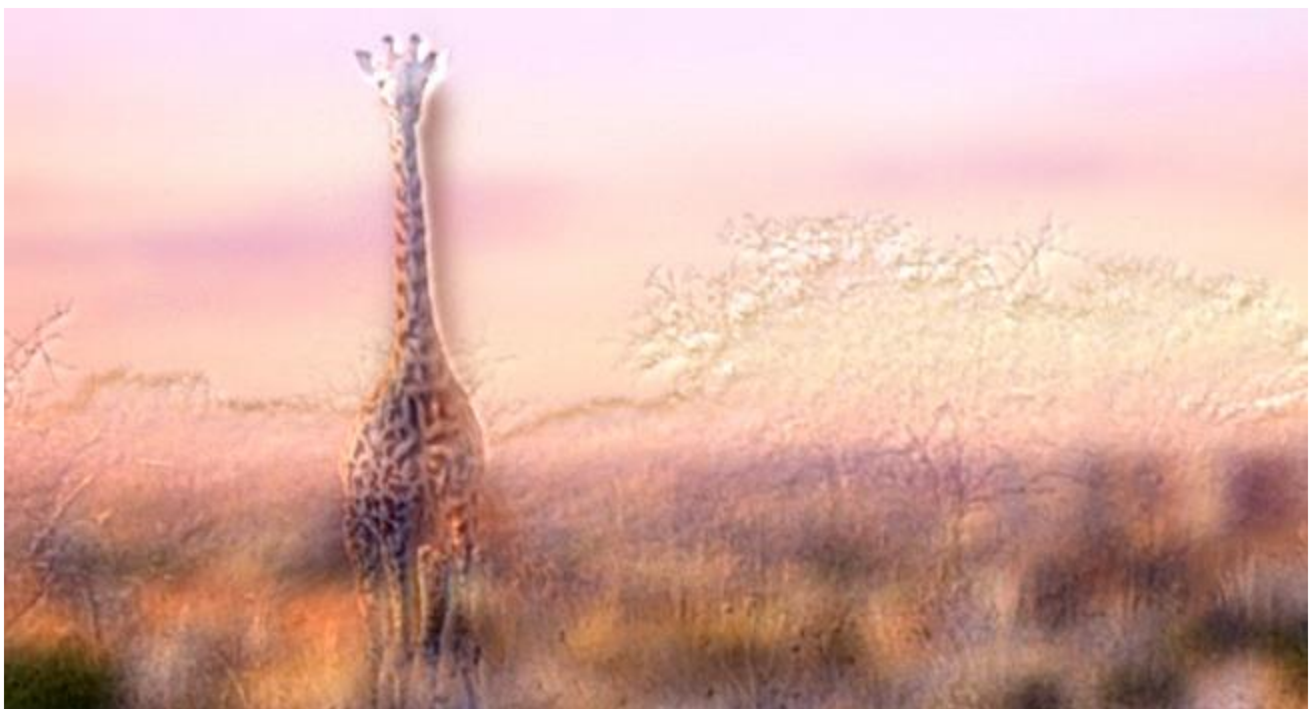
Tiffany Chen

Tiffany Chen

Tiffany Chen

Tiffany Chen

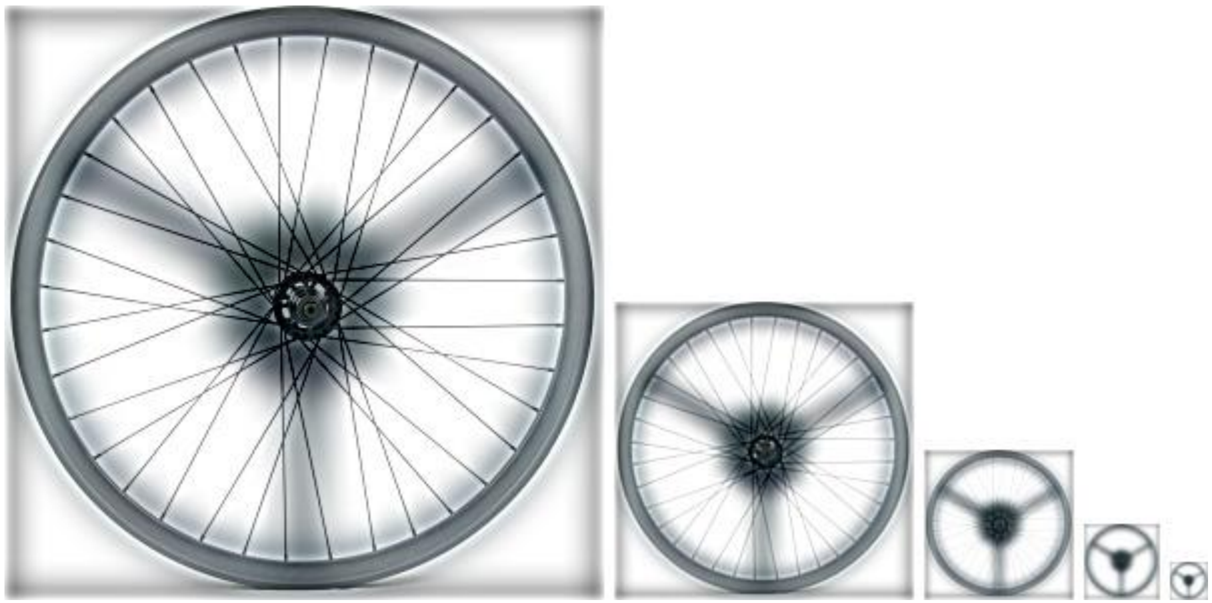




Isaiah Brand

Isaiah Brand

Eli White

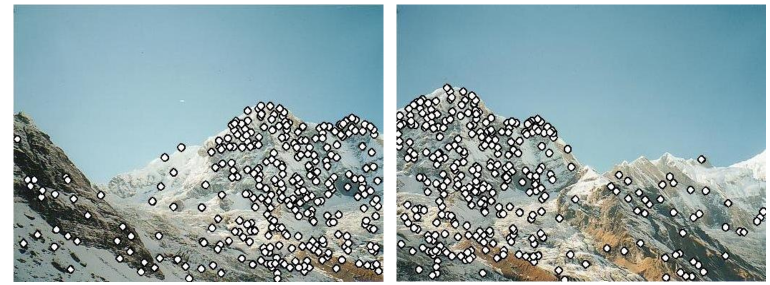


Eli White

Eli White

Local features: main components

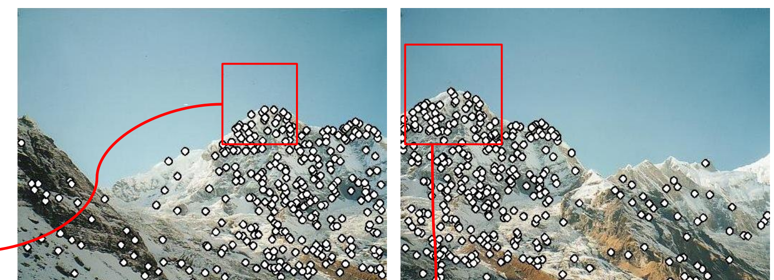
1) **Detection:**
Find a set of distinctive key points.

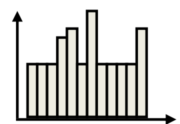


2) **Description:**
Extract feature descriptor around each interest point as vector.

x_1 

$$\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$$

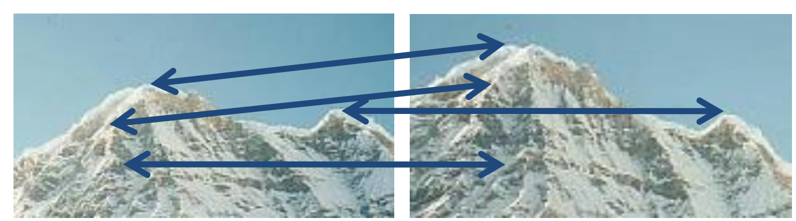


x_2 

$$\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$$

3) **Matching:**
Compute distance between feature vectors to find correspondence.

$$d(\mathrm{x}_1, \mathrm{x}_2) < T$$



Review: Interest points

- Keypoint detection: repeatable and distinctive
  - Corners, blobs, stable regions
  - Harris, DoG, MSER

(a) Gray scale input image (b) Detected MSERs

Review: Choosing an interest point detector

- Why choose?
  - Collect more points with more detectors, for more possible matches
- What do you want it for?
  - Precise localization in x-y: Harris
  - Good localization in scale: Difference of Gaussian
  - Flexible region shape: MSER
- Best choice often application dependent
  - Harris-/Hessian-Laplace/DoG work well for many natural categories
  - MSER works well for buildings and printed things
- There have been extensive evaluations/comparisons
  - [Mikolajczyk et al., IJCV'05, PAMI'05]
  - All detectors/descriptors shown here work well

Comparison of Keypoint Detectors

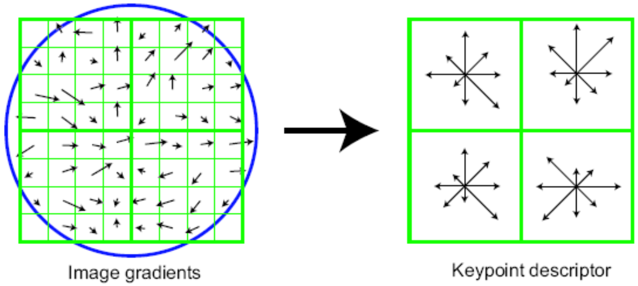
Table 7.1 Overview of feature detectors.

Feature Detector	Corner	Blob	Region	Rotation invariant	Scale invariant	Affine invariant	Repeatability	Localization accuracy	Robustness	Efficiency
Harris	√			√			+++	+++	+++	++
Hessian		√		√			++	++	++	+
SUSAN	√			√			++	++	++	+++
Harris-Laplace	√	(√)		√	√		+++	+++	++	+
Hessian-Laplace	(√)	√		√	√		+++	+++	+++	+
DoG	(√)	√		√	√		++	++	++	++
SURF	(√)	√		√	√		++	++	++	+++
Harris-Affine	√	(√)		√	√	√	+++	+++	++	++
Hessian-Affine	(√)	√		√	√	√	+++	+++	+++	++
Salient Regions	(√)	√		√	√	(√)	+	+	++	+
Edge-based	√			√	√	√	+++	+++	+	+
MSER			√	√	√	√	+++	+++	++	+++
Intensity-based			√	√	√	√	++	++	++	++
Superpixels			√	√	(√)	(√)	+	+	+	+

Tuytelaars Mikolajczyk 2008

Review: Local Descriptors

- Most features can be thought of as templates, histograms (counts), or combinations
- The ideal descriptor should be
  - Robust and Distinctive
  - Compact and Efficient



- Most available descriptors focus on edge/gradient information
  - Capture texture information
  - Color rarely used

K. Grauman, B. Leibe

Choosing a descriptor

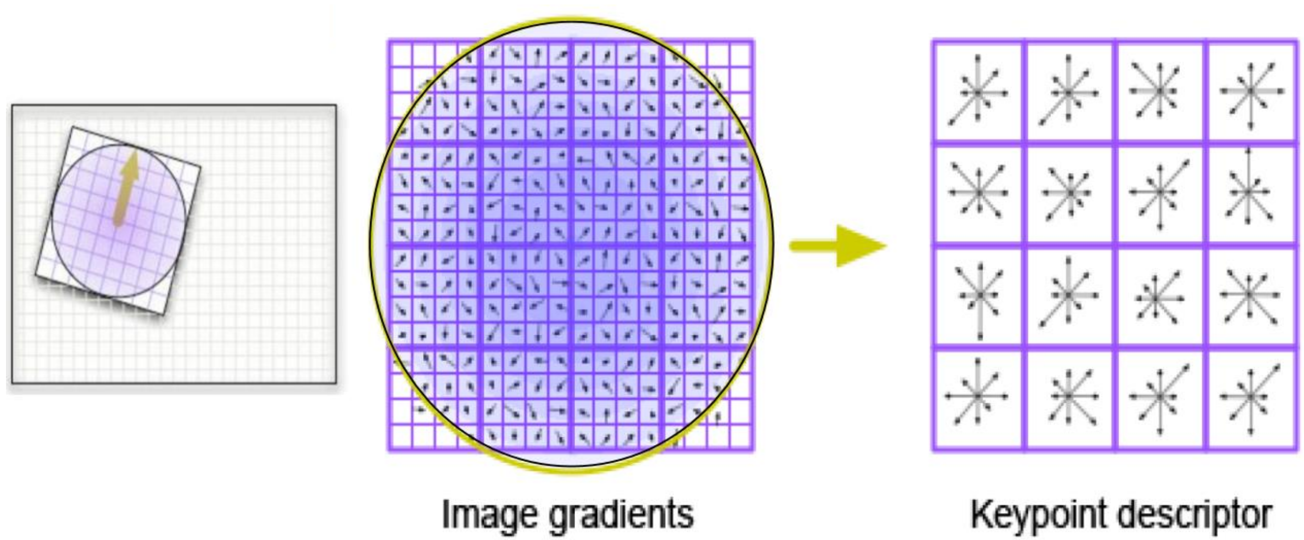
- Again, need not stick to one
- For object instance recognition or stitching, SIFT or variant is a good choice

SIFT

- Find Difference of Gaussian scale-space extrema
- Post-processing
  - Position interpolation
  - Discard low-contrast points
  - Eliminate points along edges
- Orientation estimation
- Descriptor extraction
  - Motivation: We want some sensitivity to spatial layout, and illumination, but not too much – don't want to match everything to everything!

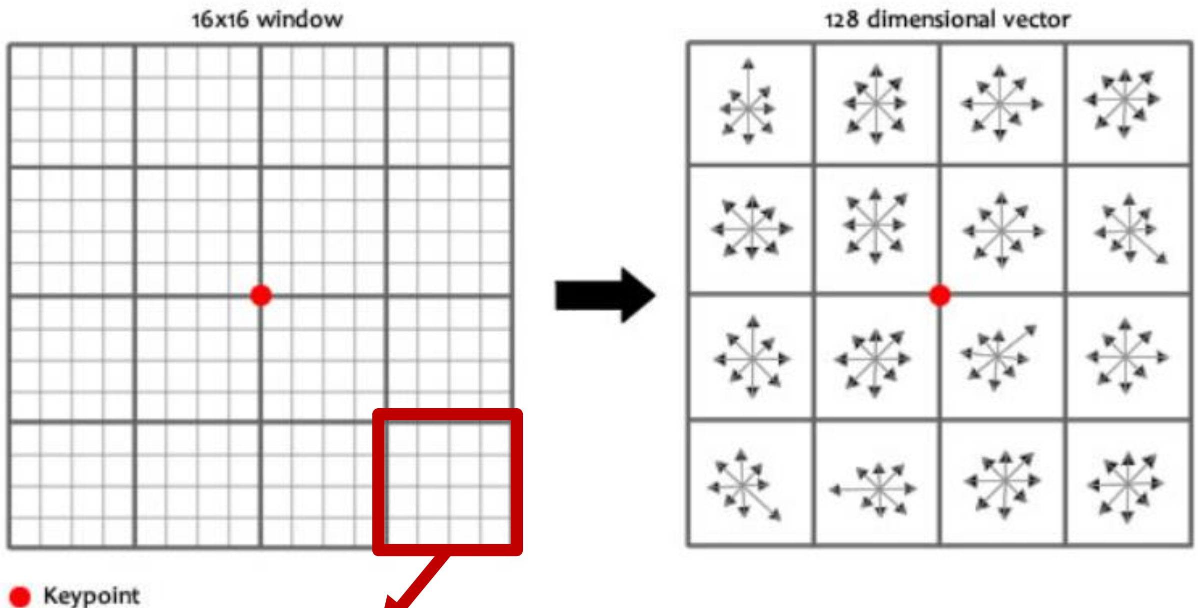
SIFT Descriptor Extraction

- Given a keypoint with scale and orientation:
  - Pick scale-space image which most closely matches estimated scale
  - Resample image to match orientation OR
  - Subtract detector orientation from vector to give invariance to general image rotation.

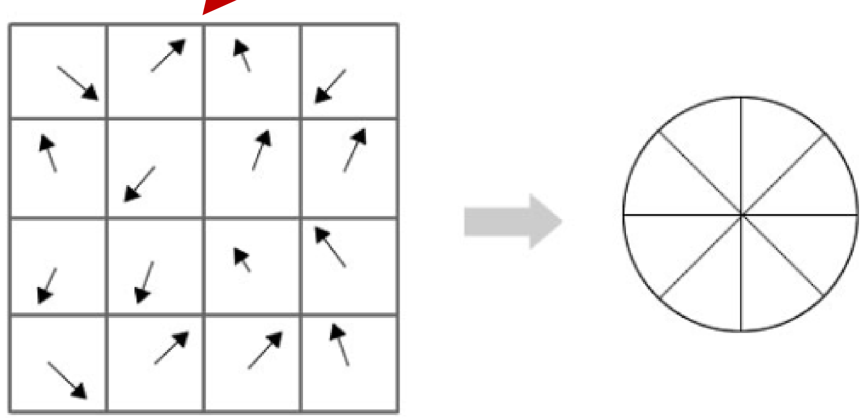


SIFT Descriptor Extraction

- Given a keypoint with scale and orientation



Gradient magnitude and orientation



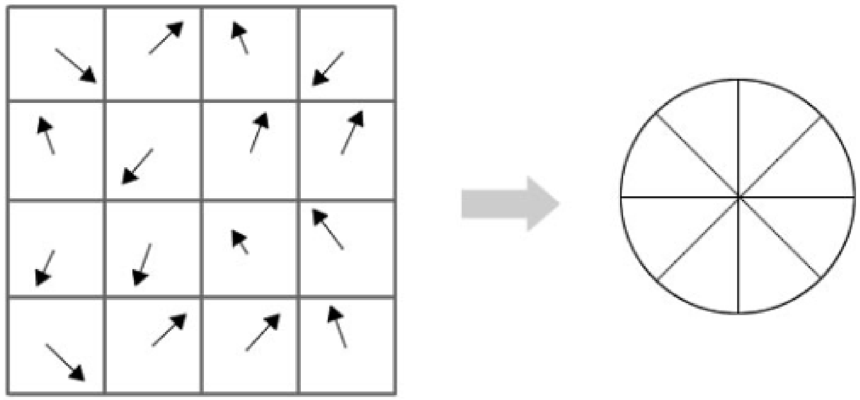
8 bin 'histogram' - add magnitude amounts!

Utkarsh Sinha

SIFT Descriptor Extraction

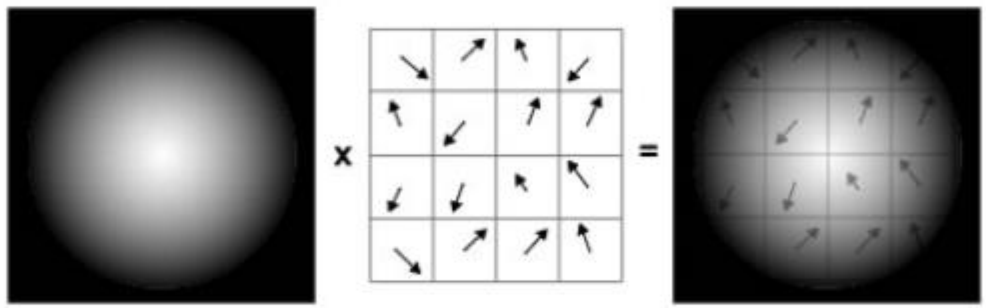
- Within each 4x4 window

Gradient magnitude and orientation



8 bin ‘histogram’ - add magnitude amounts!

Weight magnitude that is added to ‘histogram’ by Gaussian



SIFT Descriptor Extraction

- Extract 8 x 16 values into 128-dim vector
- Illumination invariance:
  - Working in gradient space, so robust to $I = I + b$
  - Normalize vector to [0…1]
    - Robust to $I = \alpha I$ brightness changes
  - Clamp all vector values > 0.2 to 0.2.
    - Robust to “non-linear illumination effects”
    - Image value saturation / specular highlights
  - Renormalize

Specular highlights

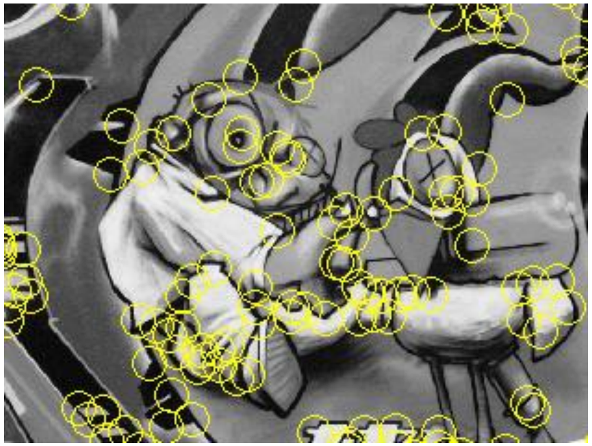
SIFT Review

- TA: Martin Zhu found this tutorial
- http://aishack.in/tutorials/sift-scale-invariant-feature-transform-features/

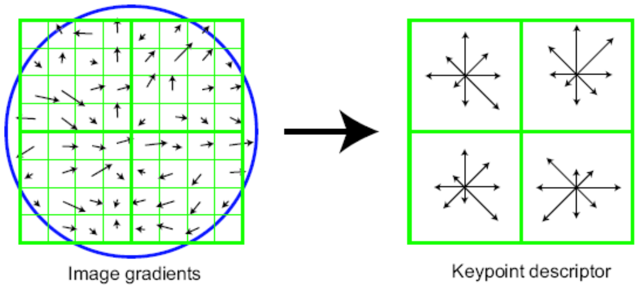
- Lowe's original paper
- http://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf

Review: Interest points

- Keypoint detection: repeatable and distinctive
  - Corners, blobs, stable regions
  - Harris, DoG



- Descriptors: robust and selective
  - Spatial histograms of orientation
  - SIFT



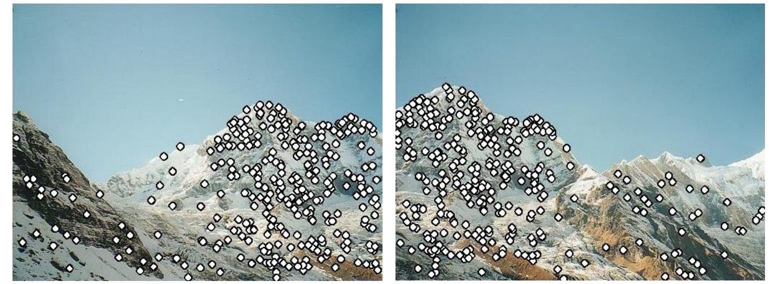
Feature Matching and Robust Fitting

Read Szeliski 4.1

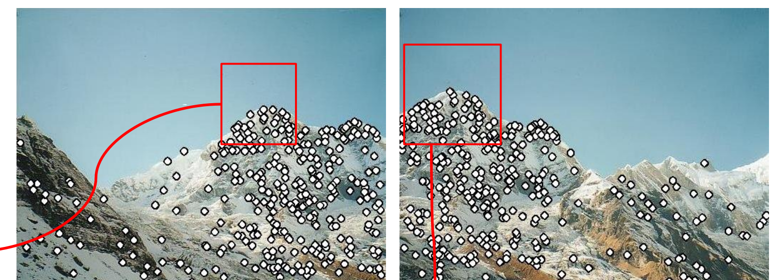
Many slides from James Hays, Derek Hoiem, and Grauman&Leibe 2008 AAAI Tutorial

Local features: main components

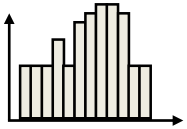
1) Detection:
   Find a set of distinctive key points.



2) Description:
   Extract feature descriptor around each interest point as vector.

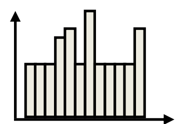


x_1



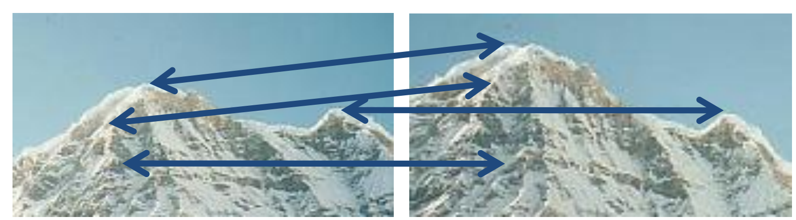
$$\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$$

x_2



$$\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$$

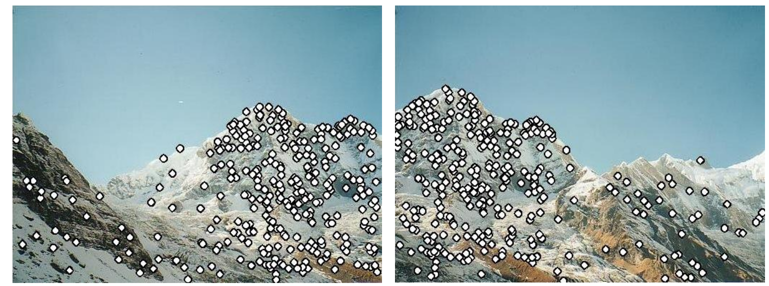
3) Matching:
   Compute distance between feature vectors to find correspondence.



K. Grauman, B. Leibe

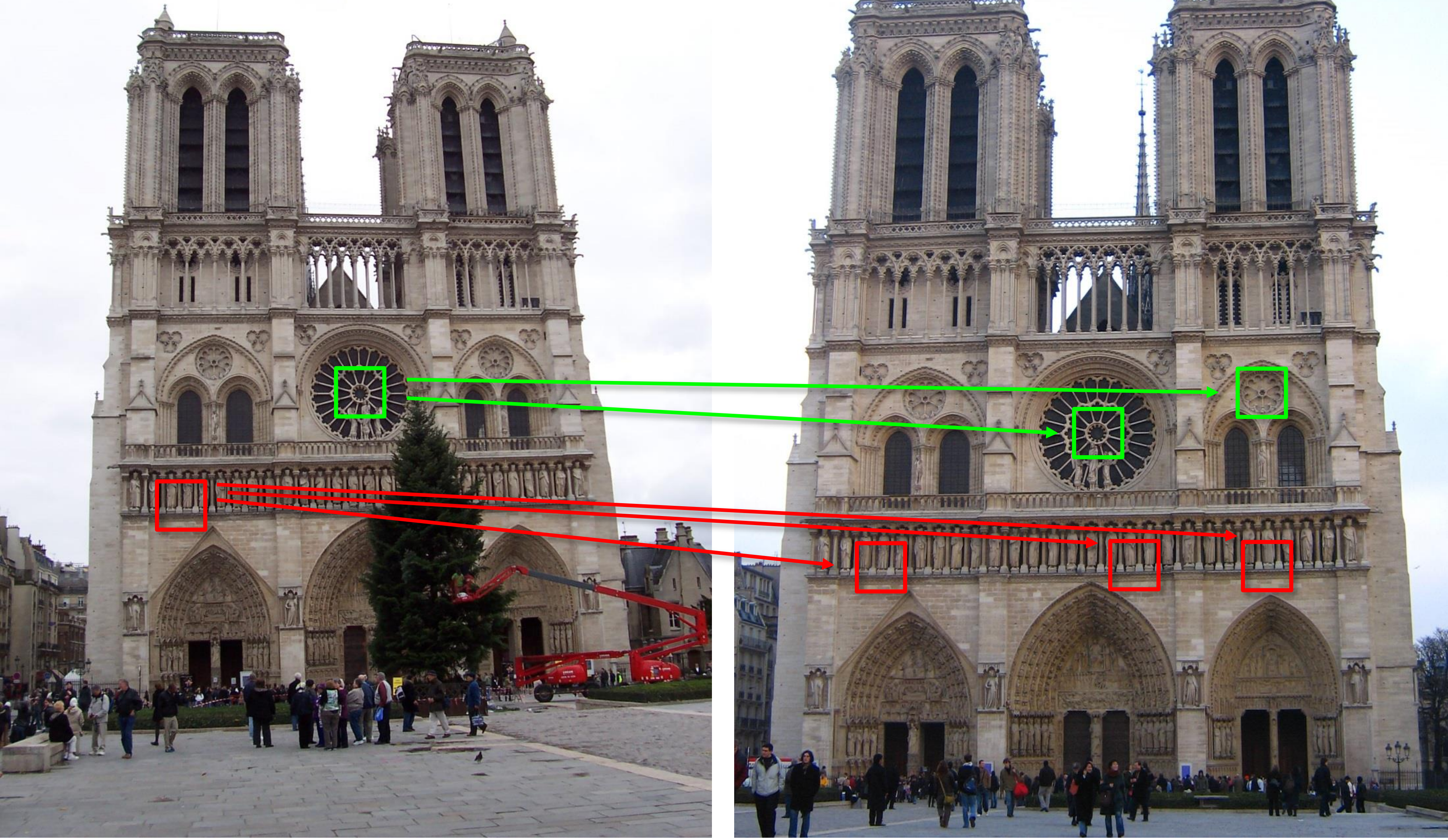
Think-Pair-Share

- *Design a feature point matching scheme.*
- Two images, I_1 and I_2



- Two sets X_1 and X_2 of feature points
  - Each feature point $\mathbf{x_1}$ has a descriptor $\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$
- Distance, bijective/injective/surjective, noise, confidence, computational complexity, generality

How do we decide which features match?



Distance: 0.34, 0.30, 0.40

Distance: 0.61, 1.22

Feature Matching

- Criteria 1:
  - Compute distance in feature space, e.g., dot product between 128-dim SIFT descriptors
  - Match point to lowest distance (nearest neighbor)

- Problems:
  - Does everything have a match?

Feature Matching

- Criteria 2:
  - Compute distance in feature space, e.g., dot product between 128-dim SIFT descriptors
  - Match point to lowest distance (nearest neighbor)
  - Ignore anything higher than threshold (no match!)

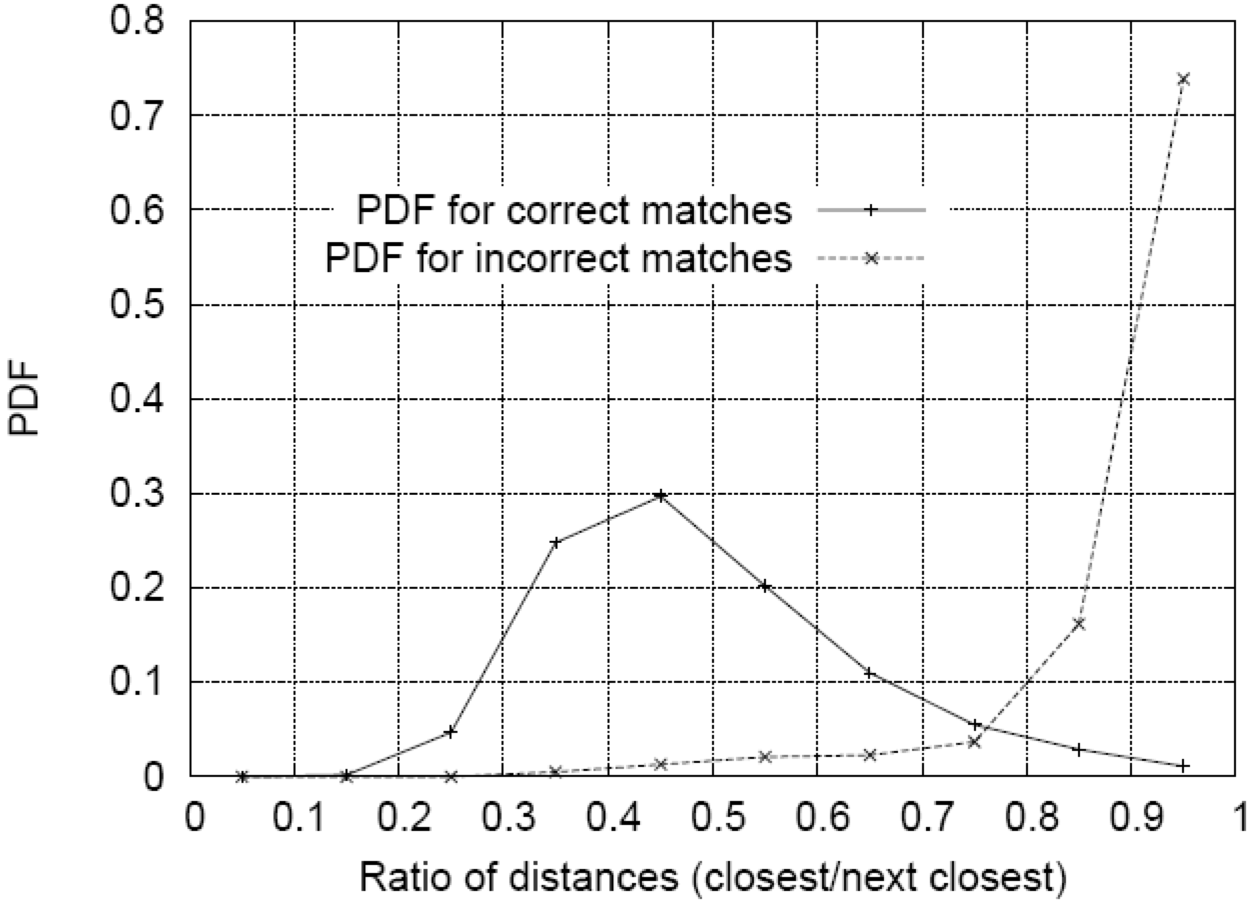
- Problems:
  - Threshold is hard to pick
  - Non-distinctive features could have lots of close matches, only one of which is correct

Nearest Neighbor Distance Ratio

- $\frac{NN1}{NN2}$ where NN1 is the distance to the first nearest neighbor and NN2 is the distance to the second nearest neighbor.
- Sorting by this ratio puts matches in order of confidence.

Matching Local Features

- Nearest neighbor (Euclidean distance)
- Threshold ratio of nearest to 2nd nearest descriptor



Lowe IJCV 2004

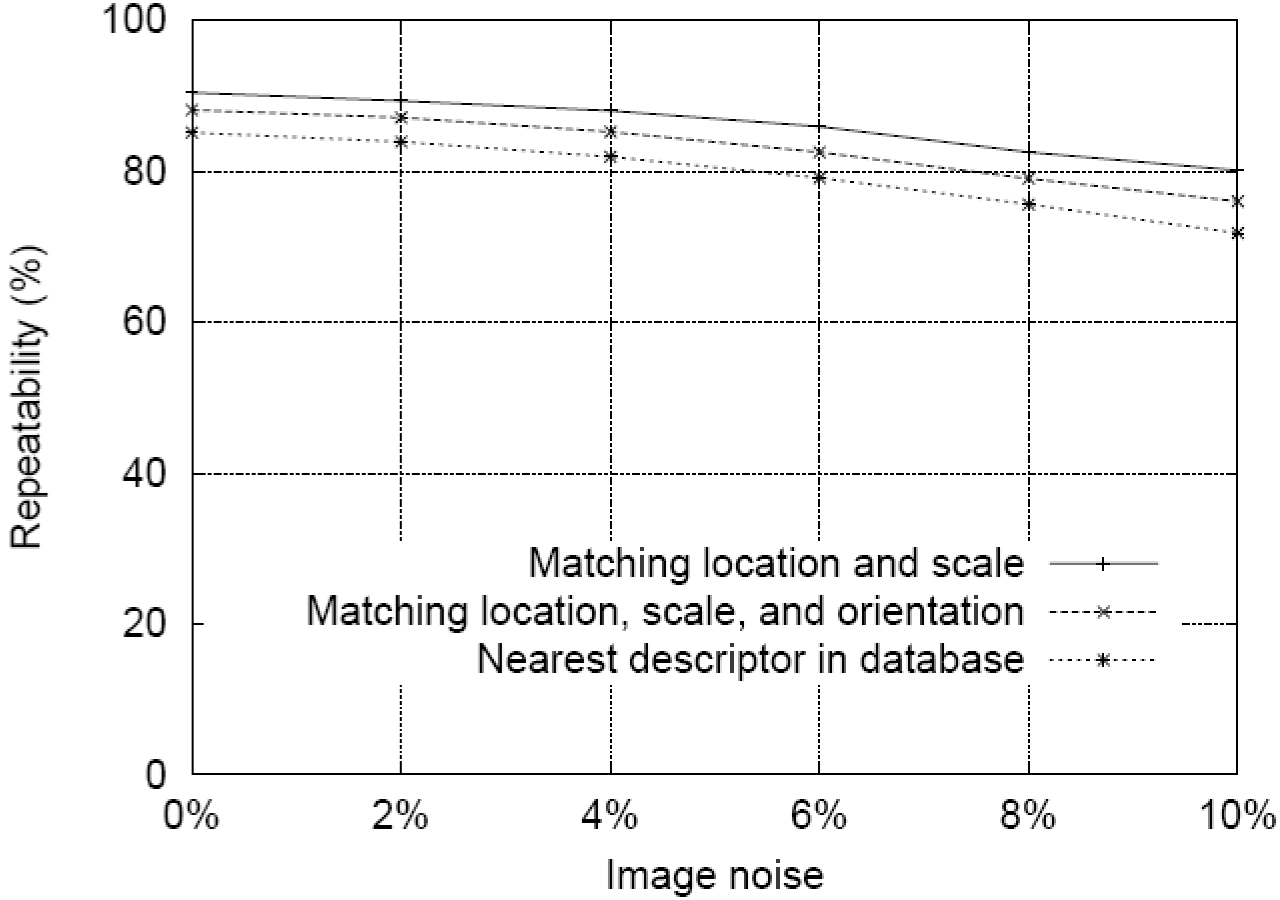
Bi-directionality / Compute cost

- Check that feature point matches hold from image 1 to image 2, and from image 2 to image 1.

- Naïve computation: Expensive
- Form all descriptors as matrix, multiply for dot products.

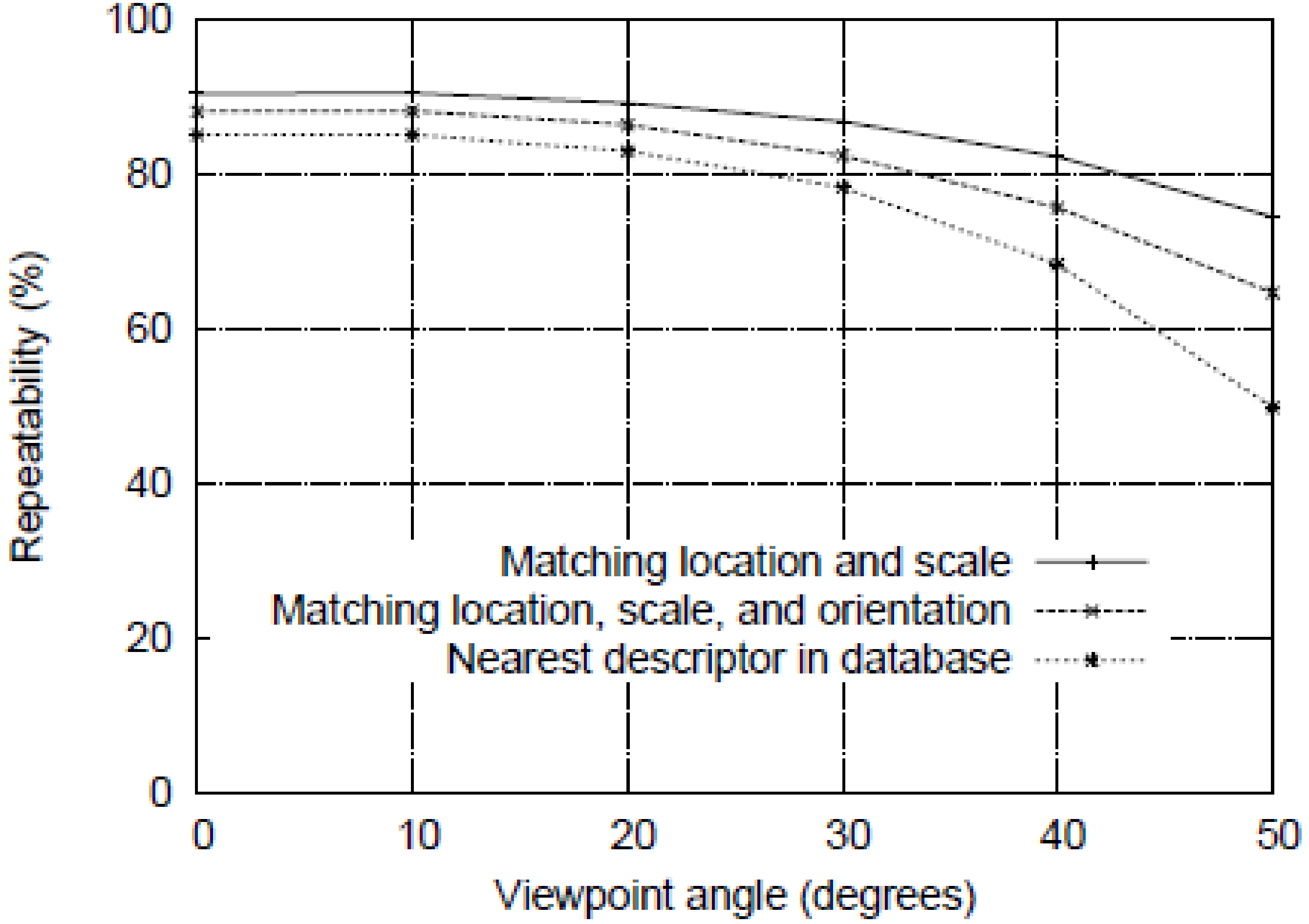
HOW GOOD IS SIFT?

SIFT Repeatability



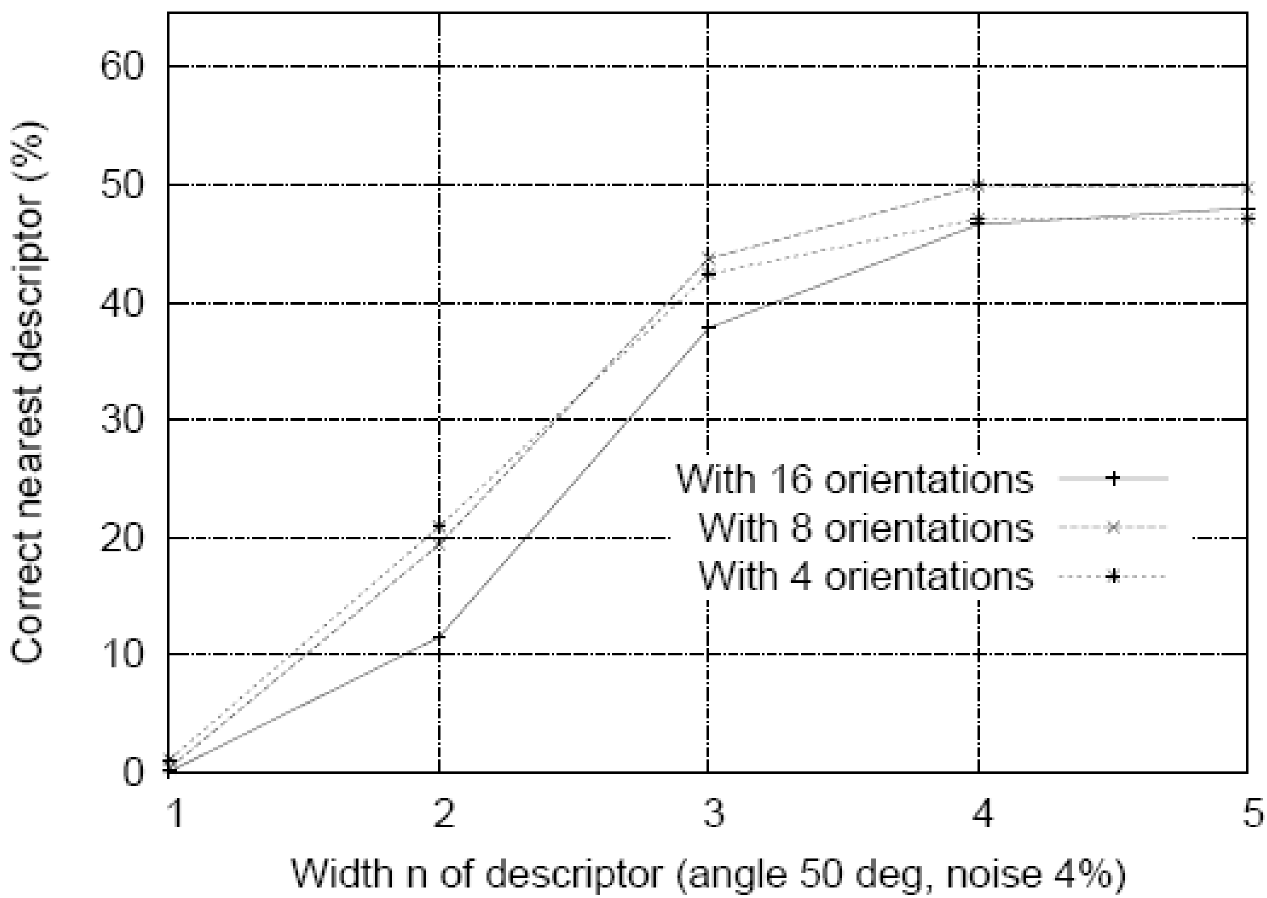
Lowe IJCV 2004

SIFT Repeatability



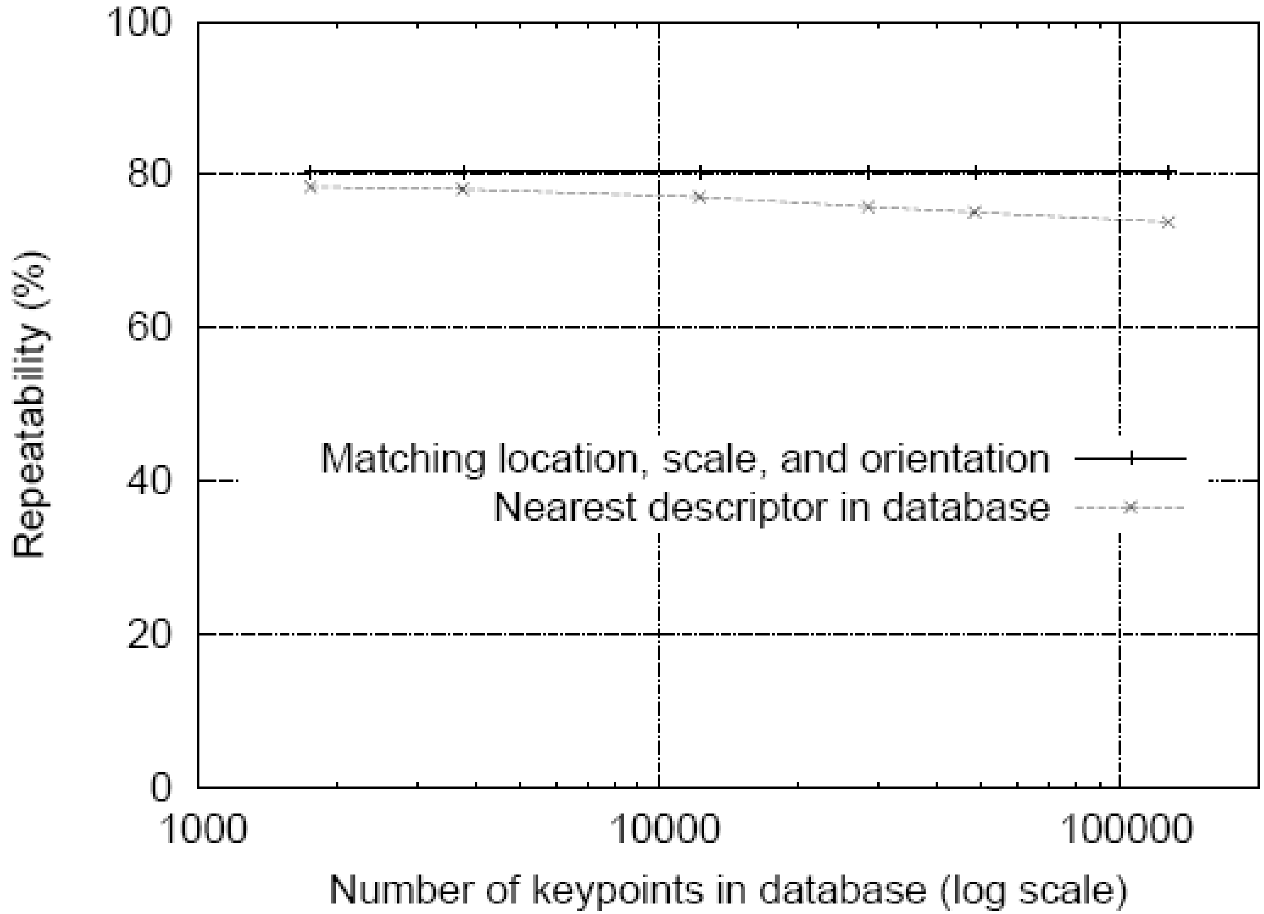
Lowe IJCV 2004

SIFT Repeatability

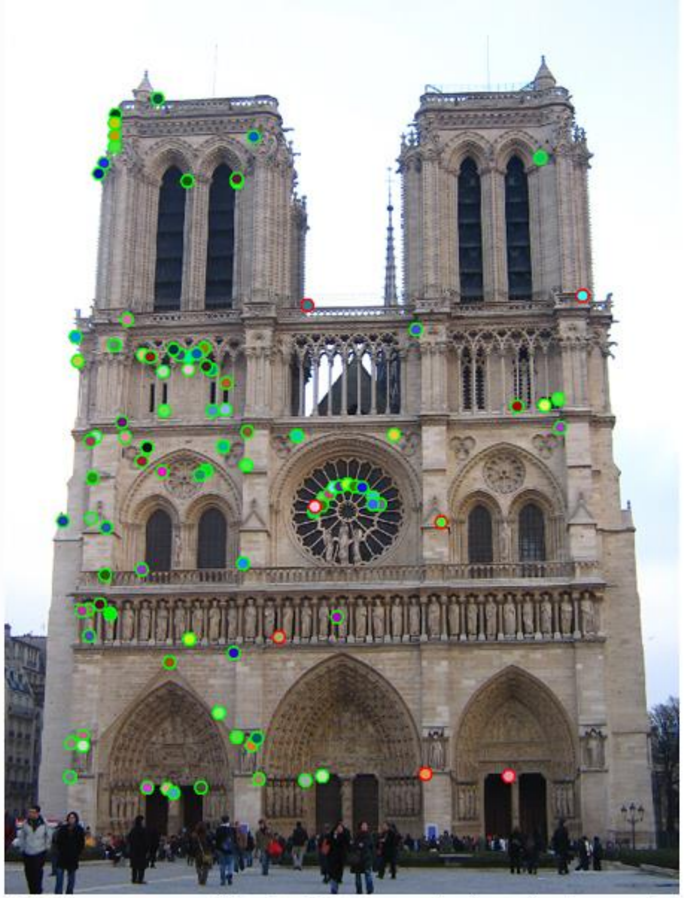


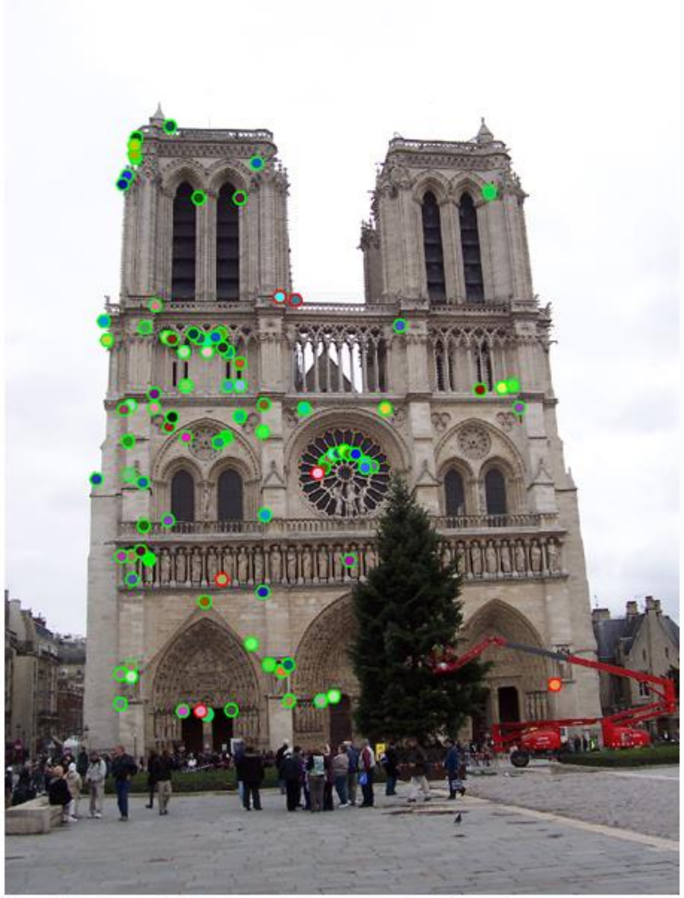
Lowe IJCV 2004

SIFT Repeatability



Lowe IJCV 2004





The top 100 most confident local feature matches from a baseline implementation of project 2. In this case, 93 were correct (highlighted in green) and 7 were incorrect (highlighted in red).

Project 2: Local Feature Matching

CSCI 1430: Introduction to Computer Vision

Brief

- Due: 9:00pm on Friday, 24th February, 2016
- Project materials including writeup template proj2.zip (6.9 MB).
- Additional scenes to test on extra_data.zip (194 MB).
- Handin: through $ cs1430_handin proj2
- Required files: README, code/, html/, html/index.html