Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

CS7GV1 Computer vision

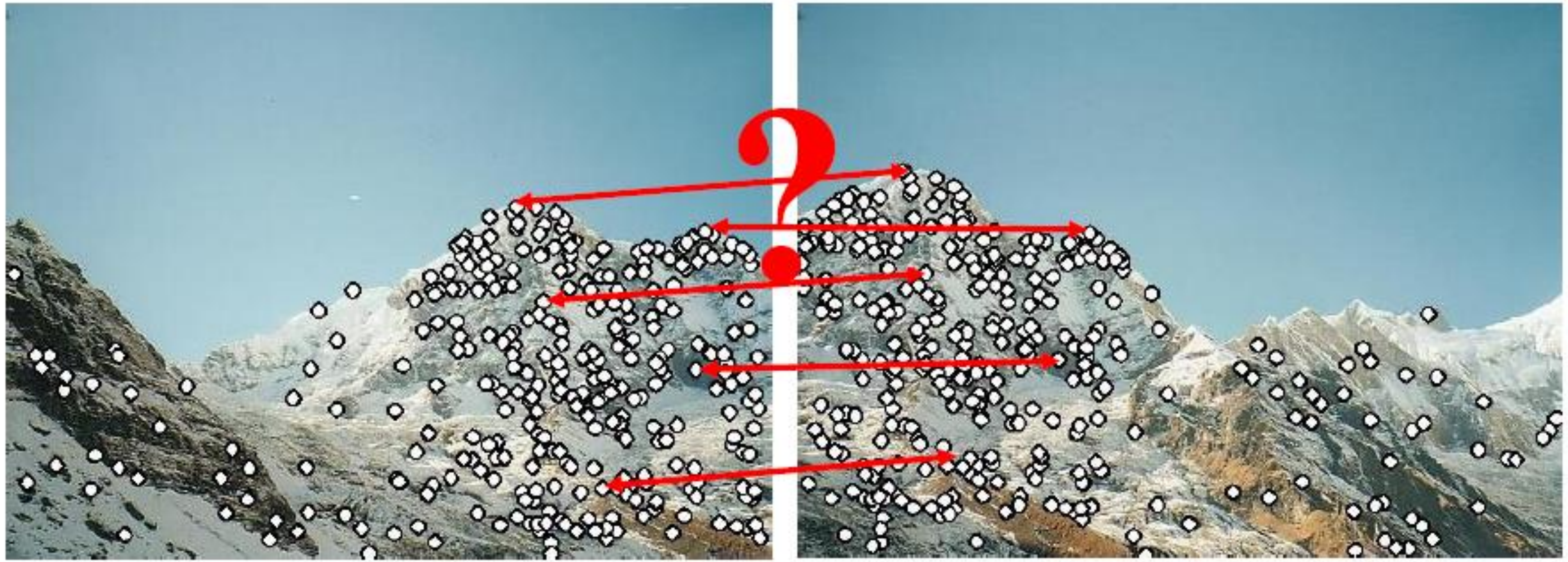
Local appearance descriptors

Dr. Martin Alain

Introduction

- Local appearance descriptors allows to describe informative areas of images

- An interesting property of interest for defining these descriptors is that they be <u>invariant</u> e.g. to scale or rotation changes.



Local approximation to surface image f

Recall that the Hessian matrix of $z = f(x, y)$ is defined to be

$$H_f(x,y) = \begin{bmatrix} f_{xx} & f_{xy} \\ f_{yx} & f_{yy} \end{bmatrix},$$

at any point at which all the second partial derivatives of f exist.

Recall that the local quadratic approximation to $z = f(x, y)$ at (x_0, y_0) is

$$f(x,y) \approx f(x_0,y_0) + \vec{\nabla} f(x_0,y_0) \cdot \begin{bmatrix} x - x_0 \\ y - y_0 \end{bmatrix} + \frac{1}{2} \begin{bmatrix} x - x_0 & y - y_0 \end{bmatrix} H_f(x_0,y_0) \begin{bmatrix} x - x_0 \\ y - y_0 \end{bmatrix},$$

https://www.iith.ac.in/~ashok/Maths_Lectures/TutorialB/Hessian_Examples.pdf

Local approximation to surface image f

Eigenvalues give information about a matrix; the Hessian matrix contains geometric information about the surface $z = f(x, y)$. We're going to use the eigenvalues of the Hessian matrix to get geometric information about the surface.

Here's the definition:

Definition 3.1. Let A be a square (that is, $n \times n$) matrix, and suppose there is a scalar λ and a vector $\vec{x}$ for which

$$A\vec{x} = \lambda\vec{x}.$$

Then

a. the ordered pair $(\lambda, \vec{x})$ is an *eigenpair of* A,

b. λ is an *eigenvalue of* A, and

c. $\vec{x}$ is an *eigenvector of* A *associated with* λ.

Quick eigenvalue/eigenvector review

The eigenvectors of a matrix A are the vectors x that satisfy:

$$Ax = \lambda x$$

$$det(A - \lambda I) = 0$$

The scalar λ is the eigenvalue corresponding to

- The eigenvalues are found by solving:

$$det \begin{bmatrix} h_{11} - \lambda & h_{12} \\ h_{21} & h_{22} - \lambda \end{bmatrix} = 0$$

- In our case, ***A*** = ***H*** is a 2x2 matrix, so we have
- The solution:

$$\lambda_{\pm} = \frac{1}{2}\left[(h_{11} + h_{22}) \pm \sqrt{4h_{12}h_{21} + (h_{11} - h_{22})^2}\right]$$

Once you know λ, you find x by solving

$$\begin{bmatrix} h_{11} - \lambda & h_{12} \\ h_{21} & h_{22} - \lambda \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 0$$

Feature detection: the math

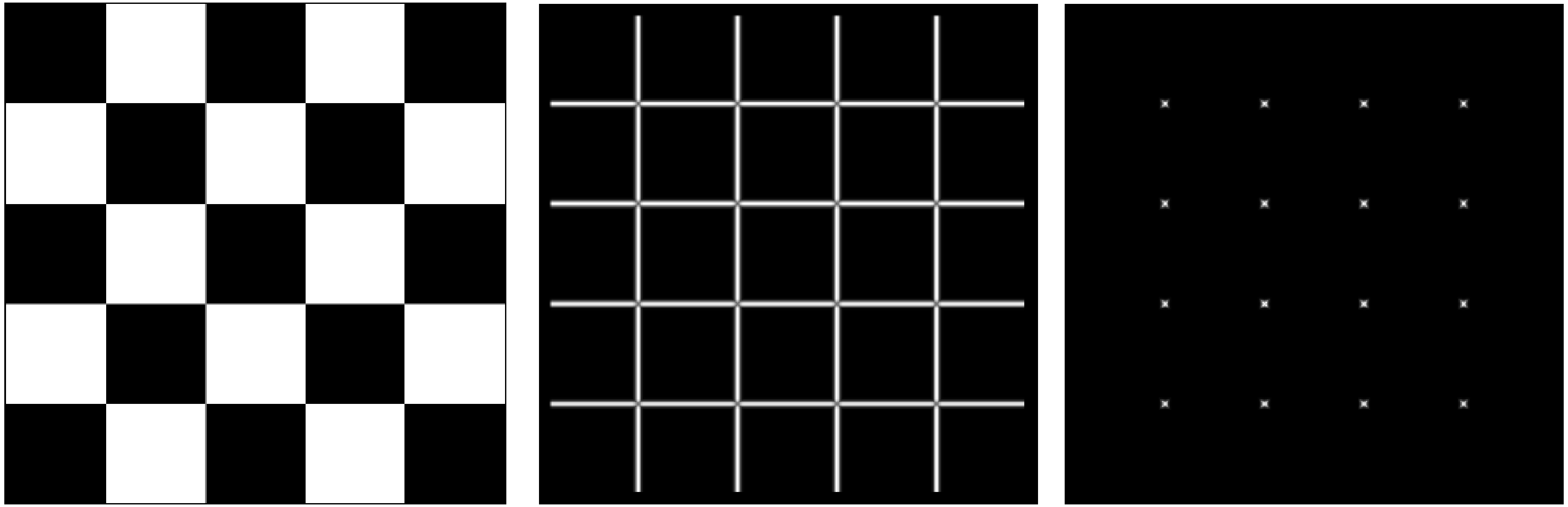
Eigenvalues and eigenvectors of H

- Define shifts with the smallest and largest change (E value)
- x_+ = direction of **largest** increase in E.
- λ_+ = amount of increase in direction x_+
- x_- = direction of **smallest** increase in E.
- λ- = amount of increase in direction x_+

$$
\begin{aligned}
Hx_+ &= \lambda_+ x_+ \\
Hx_- &= \lambda_- x_-
\end{aligned}
$$

Feature detection summary

- Compute the gradient at each point in the image
- Create the ***H*** matrix from the entries in the gradient
- Compute the eigenvalues.
- Find points with large response (λ_- > threshold)
- Choose those points where λ_- is a local maximum as features



I λ_+ λ_-

The Harris operator

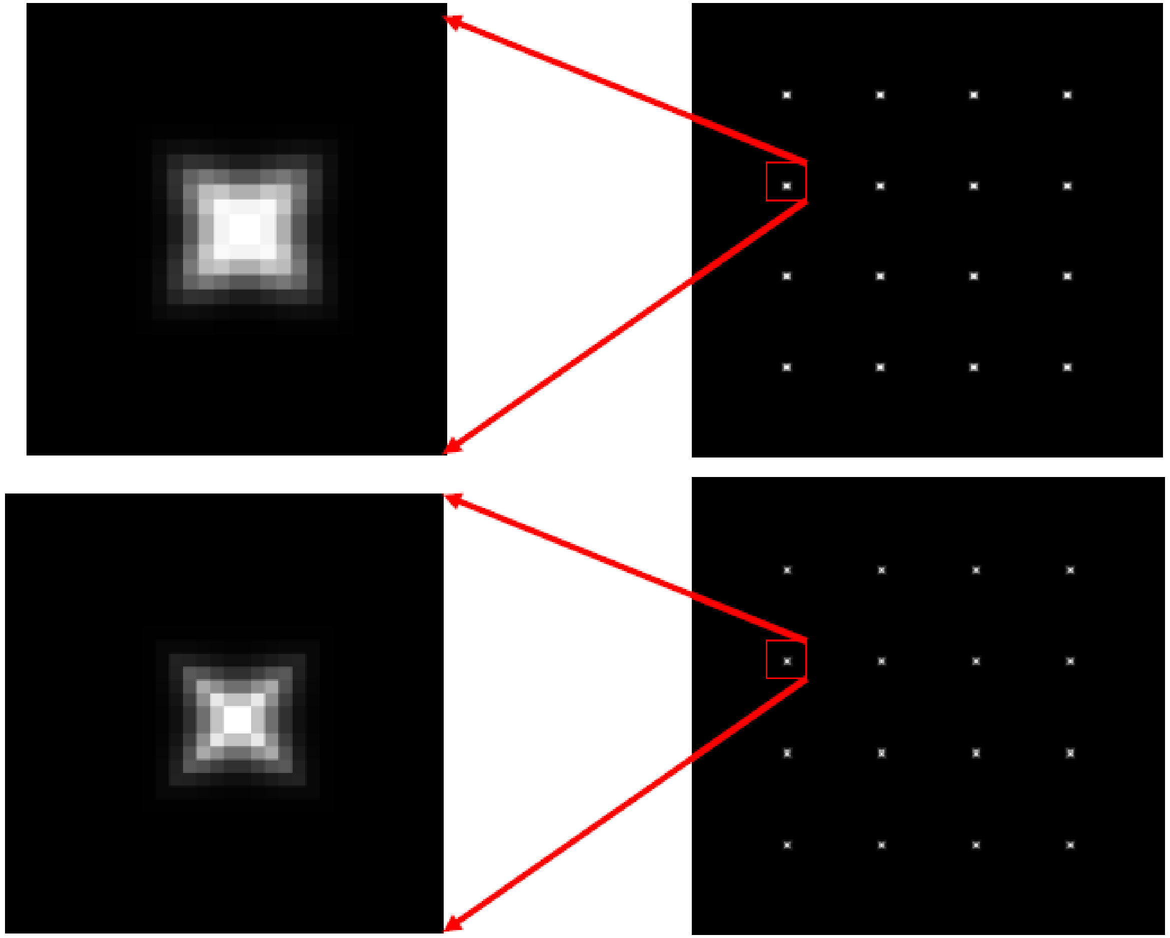
λ_- is a variant of the “Harris operator” for feature detection

$$f = \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2}$$

$$= \frac{determinant(H)}{trace(H)}$$

- The *trace* is the sum of the diagonals, i.e., $trace(H) = h_{11} + h_{22}$
- Very similar to λ_- but less expensive (no square root)
- Called the “Harris Corner Detector” or “Harris Operator”
- Lots of other detectors, this is one of the most popular

The Harris operator

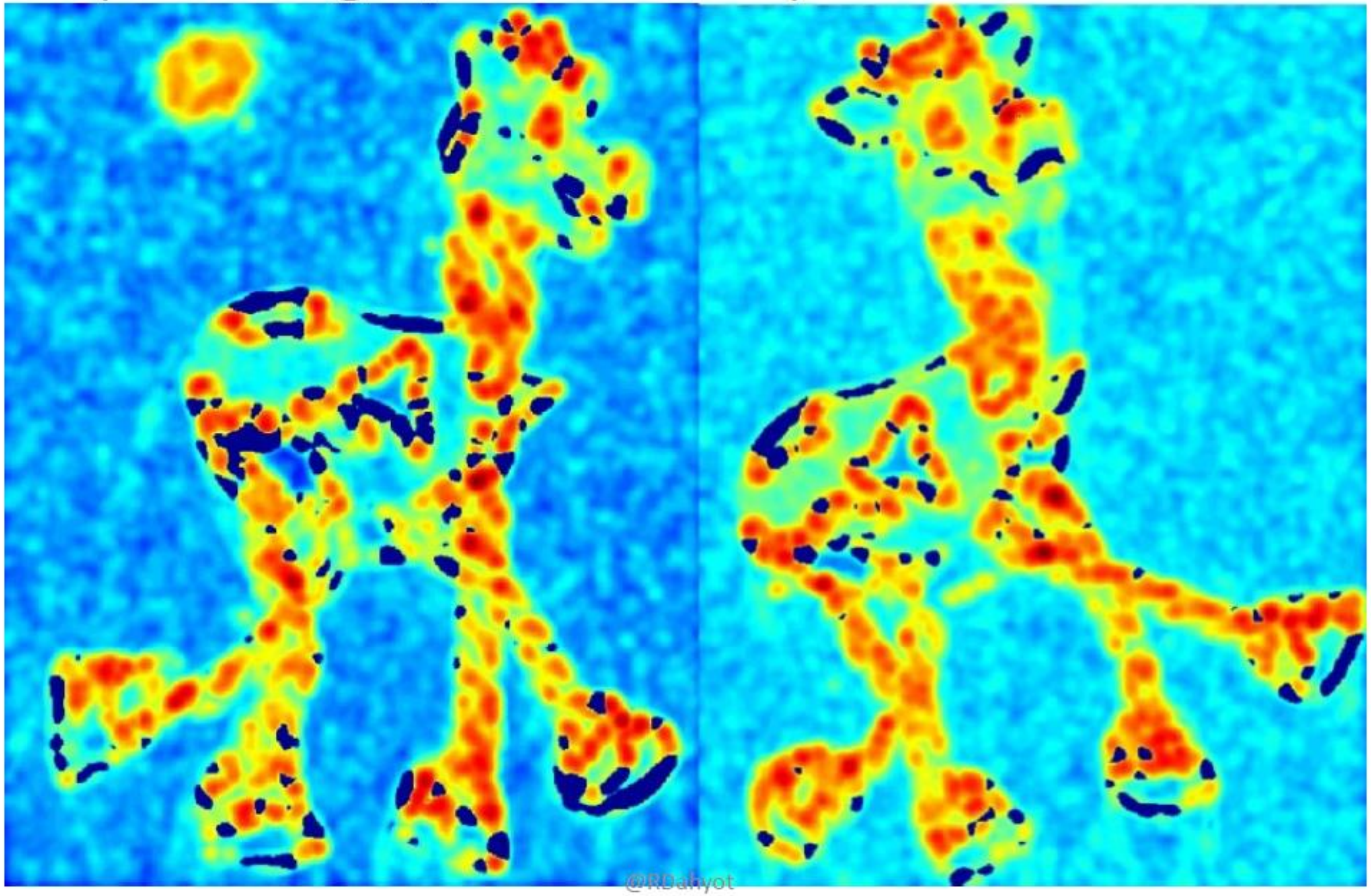


Harris operator

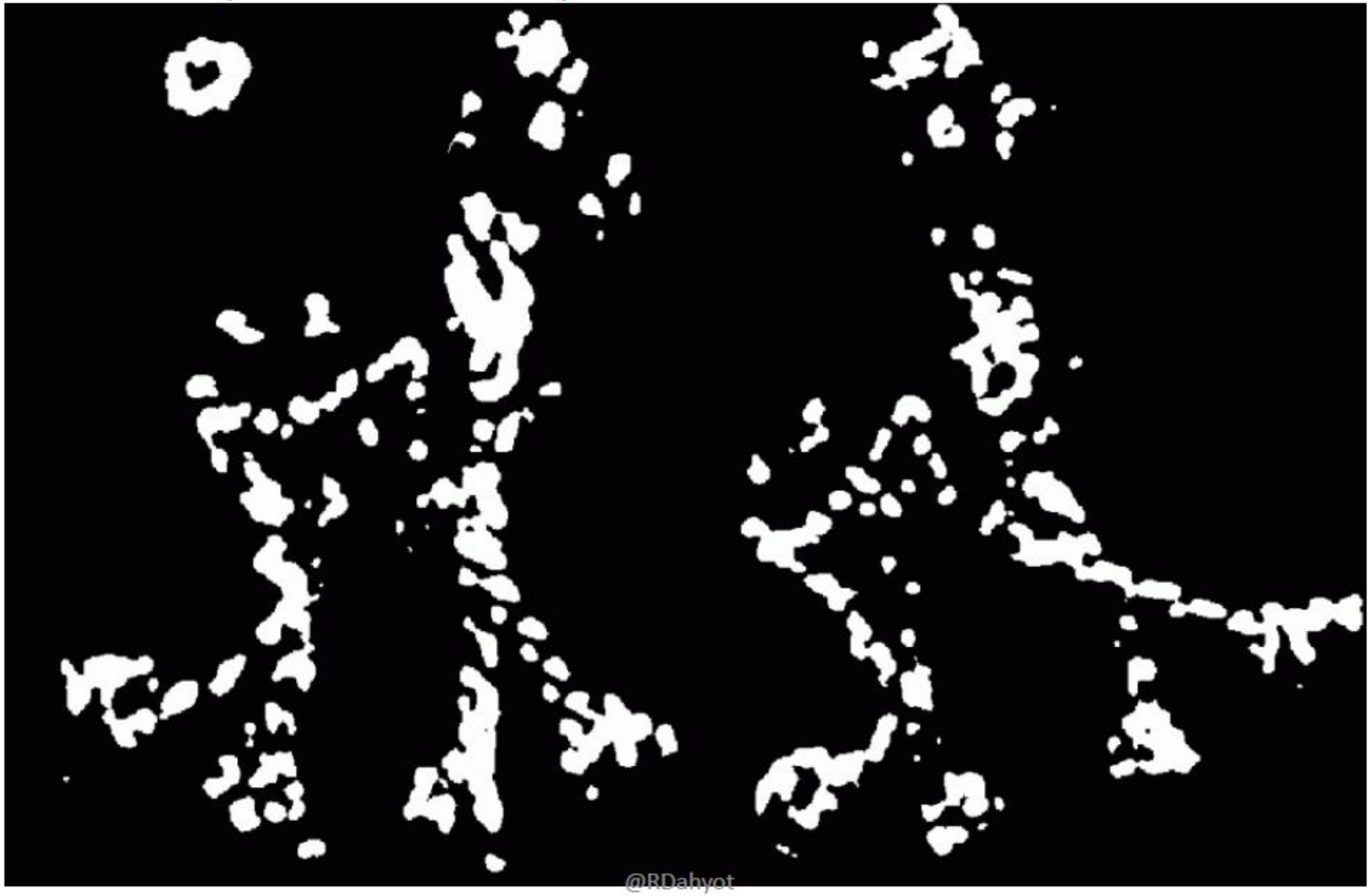
λ_-

Harris detector example

f value (red high, blue low)



Threshold (f > value)



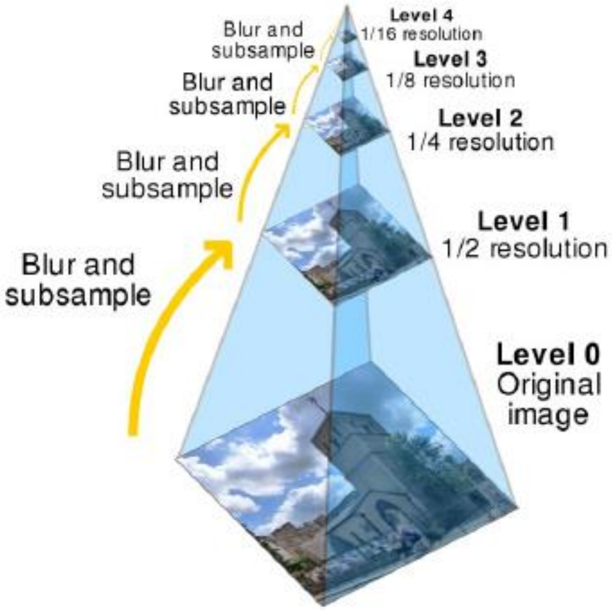
Harris features (in red)

Scale invariant interest points

How can we independently select interest points in each image, such that the detections are repeatable across different scales?

- Extract features at a **variety of scales**, e.g., by using multiple resolutions in a pyramid, and then matching features at the same level.
- When does this work?
- More efficient to extract features **stable in both location** and **scale**.
- Find scale that gives local maxima of a function f in both position and scale.

$$f(I_{i_1 \dots i_m}(x, \sigma)) = f(I_{i_1 \dots i_m}(x', \sigma'))$$



https://en.wikipedia.org/wiki/Pyramid_(image_processing)

Raquel Urtasun (lecturenotes)

https://www.cs.toronto.edu/~urtasun/courses/CV/lecture04.pdf

Automatic Scale Selection

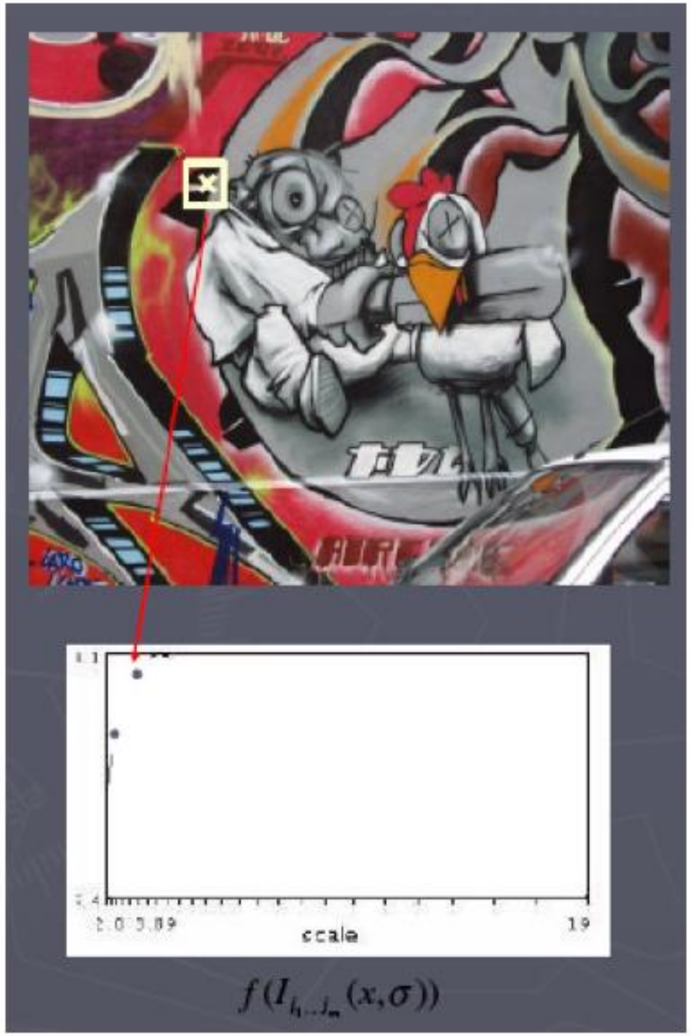
Function responses for increasing scale (scale signature).

Raquel Urtasun (lecturenotes)

https://www.cs.toronto.edu/~urtasun/courses/CV/lecture04.pdf

Automatic Scale Selection

Function responses for increasing scale (scale signature).

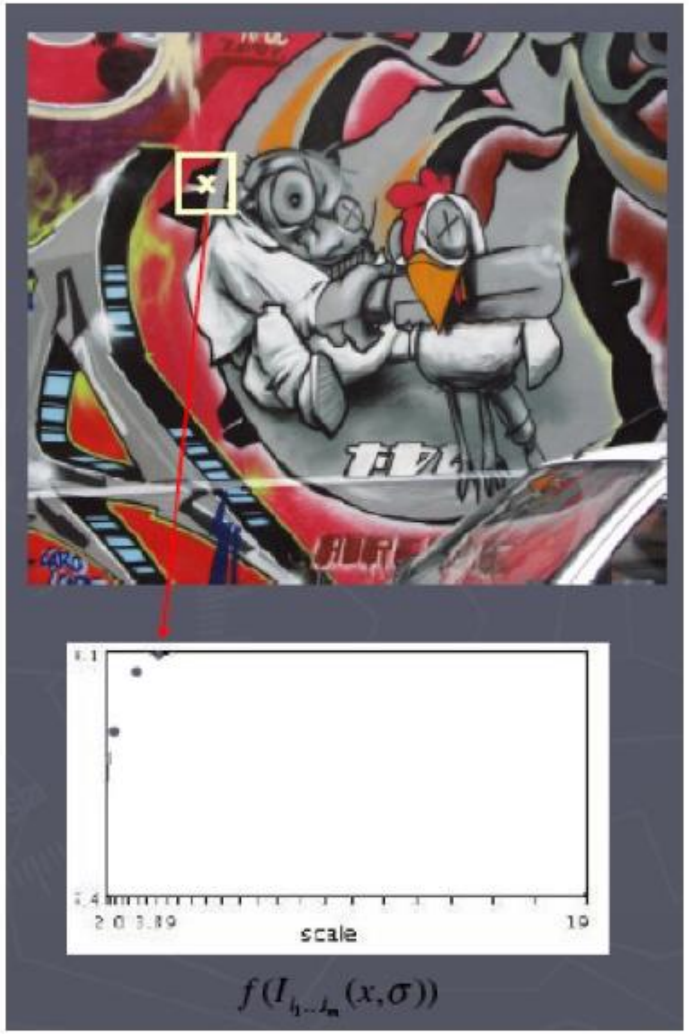


Raquel Urtasun (lecturenotes)

https://www.cs.toronto.edu/~urtasun/courses/CV/lecture04.pdf

Automatic Scale Selection

Function responses for increasing scale (scale signature).

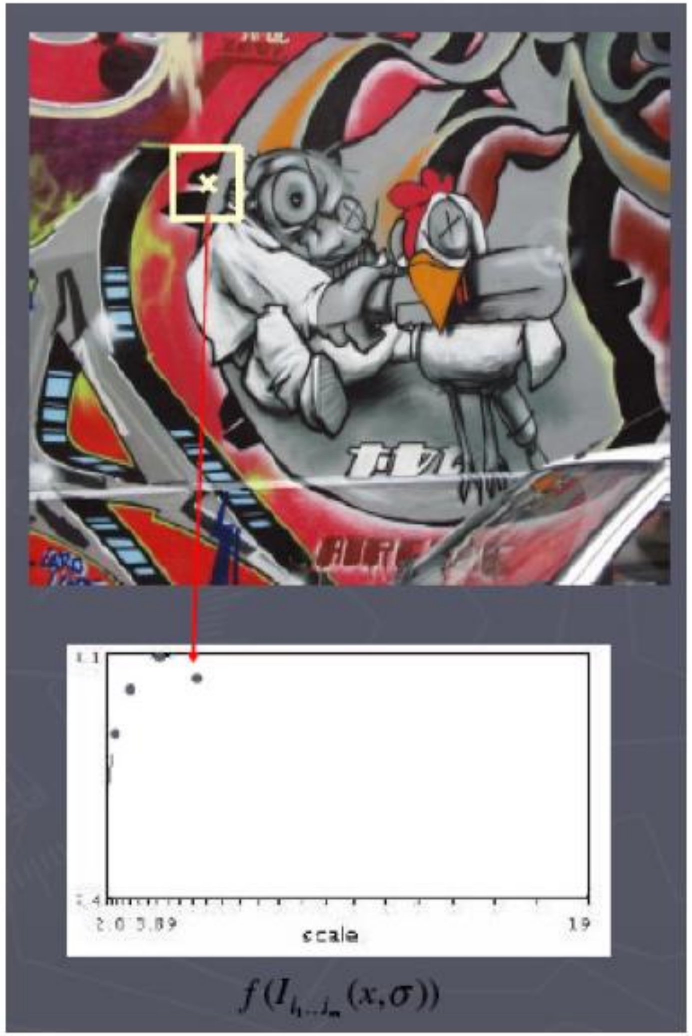


Raquel Urtasun (lecturenotes)

https://www.cs.toronto.edu/~urtasun/courses/CV/lecture04.pdf

Automatic Scale Selection

Function responses for increasing scale (scale signature).

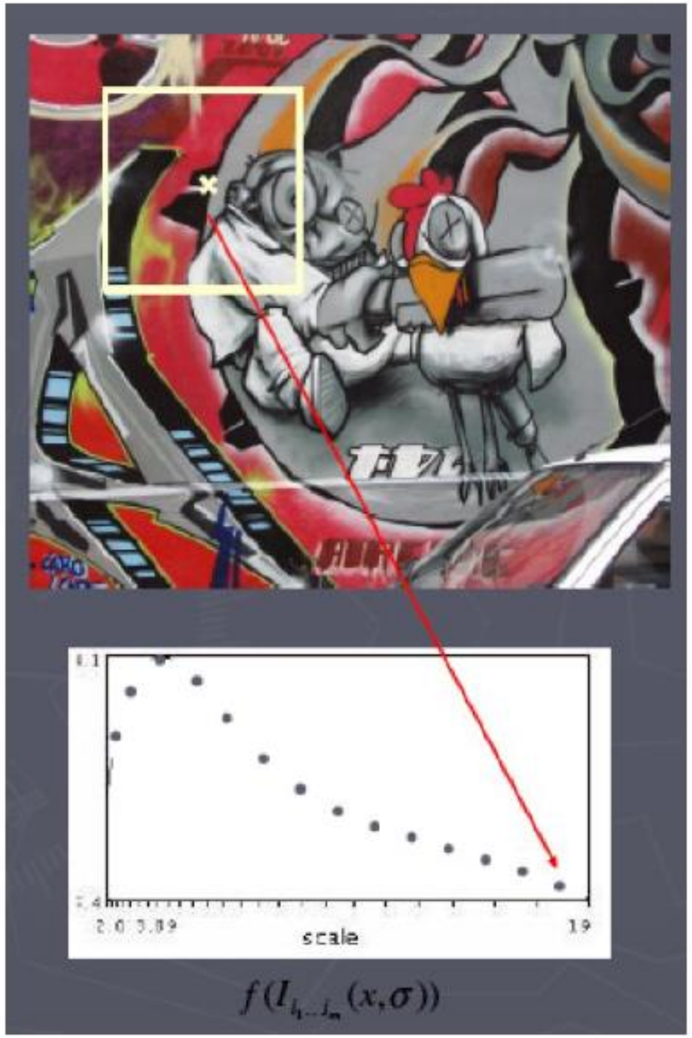


Raquel Urtasun (lecturenotes)

https://www.cs.toronto.edu/~urtasun/courses/CV/lecture04.pdf

Automatic Scale Selection

Function responses for increasing scale (scale signature).

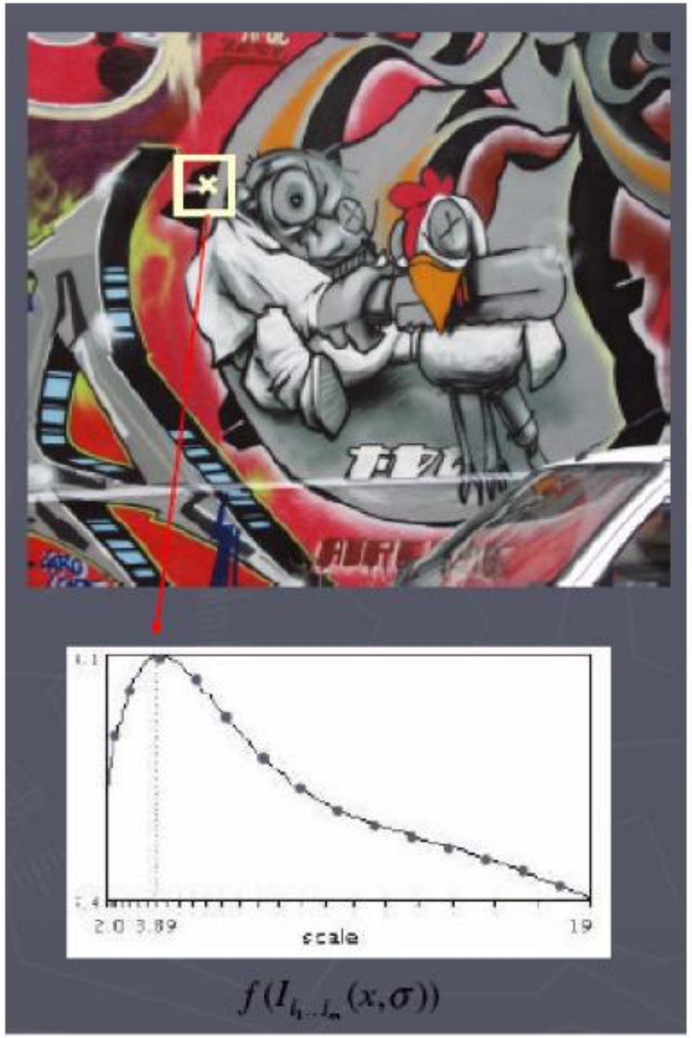


Raquel Urtasun (lecturenotes)

https://www.cs.toronto.edu/~urtasun/courses/CV/lecture04.pdf

Automatic Scale Selection

Function responses for increasing scale (scale signature).



Raquel Urtasun (lecturenotes)

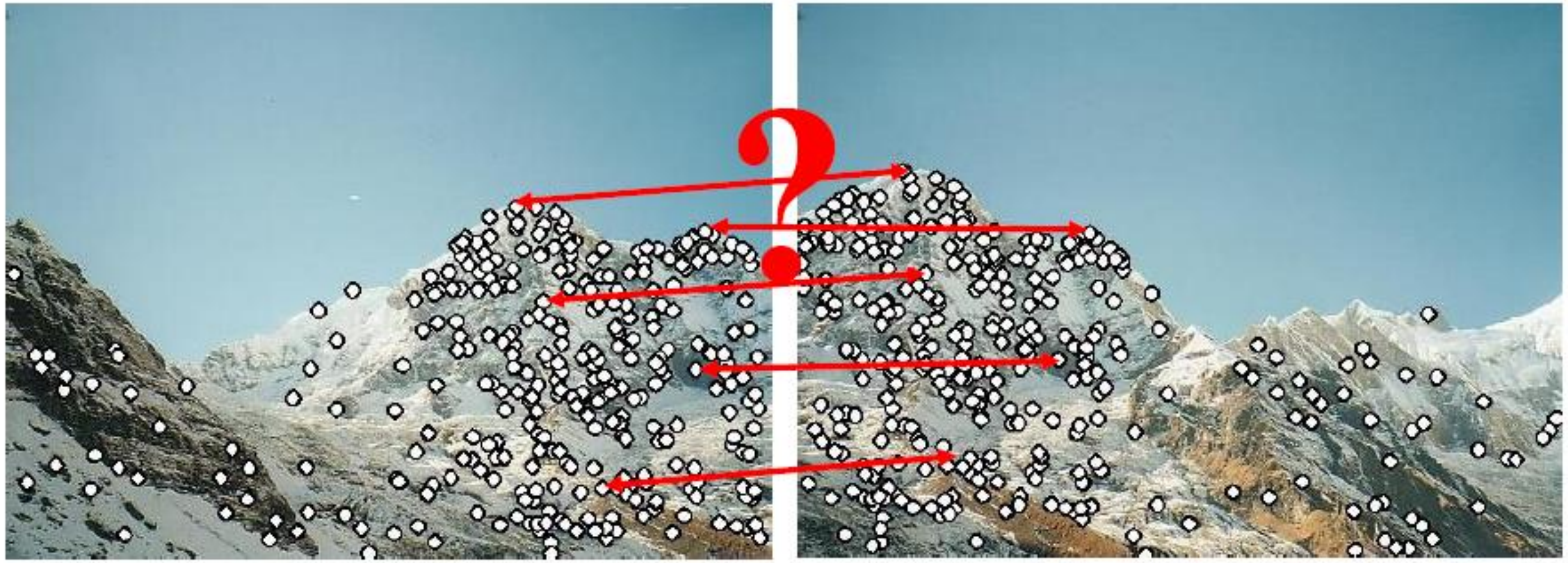
https://www.cs.toronto.edu/~urtasun/courses/CV/lecture04.pdf

Feature descriptors

We know how to detect good points
Next question: **How to match them?**

Lots of possibilities (this is a popular research area)

- Simple option: match square windows around the point
- State of the art approach: SIFT
  - David Lowe, UBC http://www.cs.ubc.ca/~lowe/keypoints/



Rotation invariance for feature descriptors

Find dominant orientation of the image patch

- This is given by $\mathbf{x}_+$, the eigenvector of $\mathbf{H}$ corresponding to λ_+
  - λ_+ is the *larger* eigenvalue
- Rotate the patch according to this angle

Figure by Matthew Brown

Detections at multiple scales

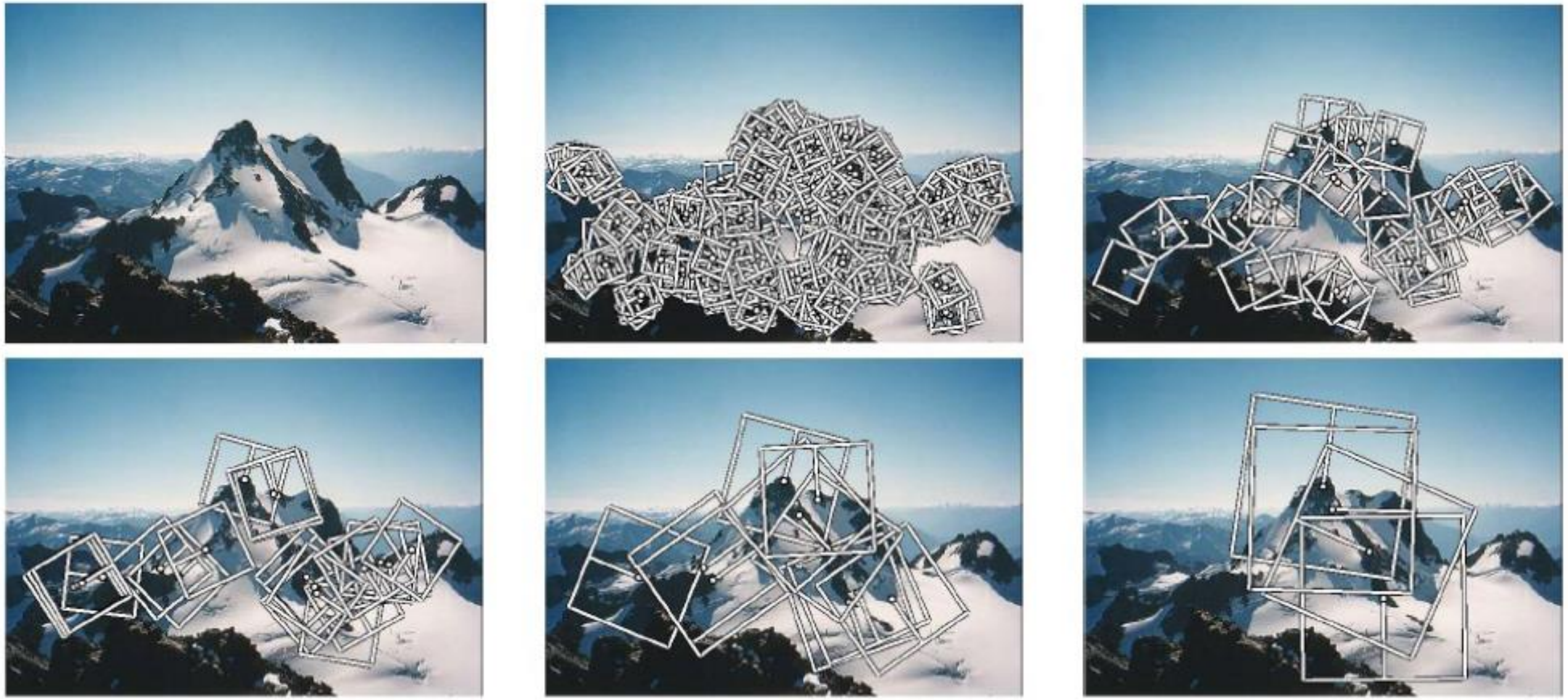


Figure 1. Multi-scale Oriented Patches (MOPS) extracted at five pyramid levels from one of the Matier images. The boxes show the feature orientation and the region from which the descriptor vector is sampled.

Scale Invariant Feature Transform: SIFT

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

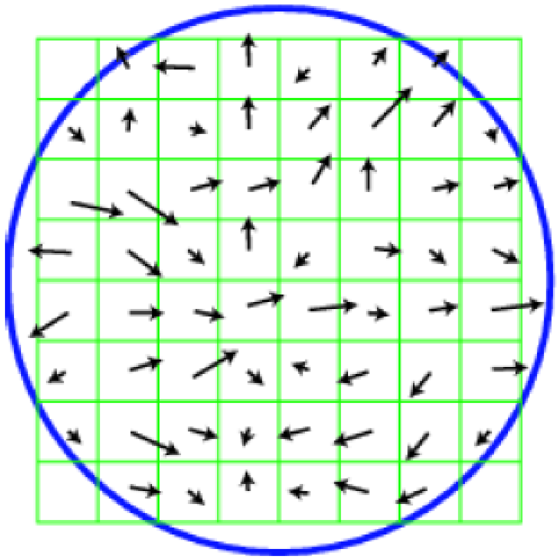
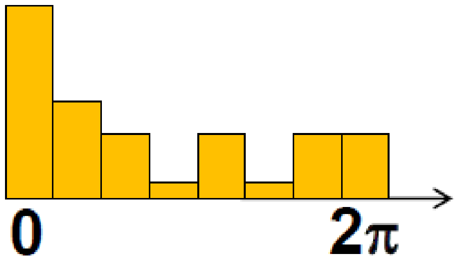
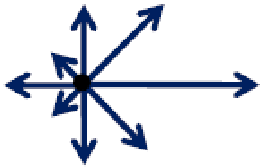


Image gradients



angle histogram



Adapted from slide by David Lowe

Scale Invariant Feature Transform: SIFT

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

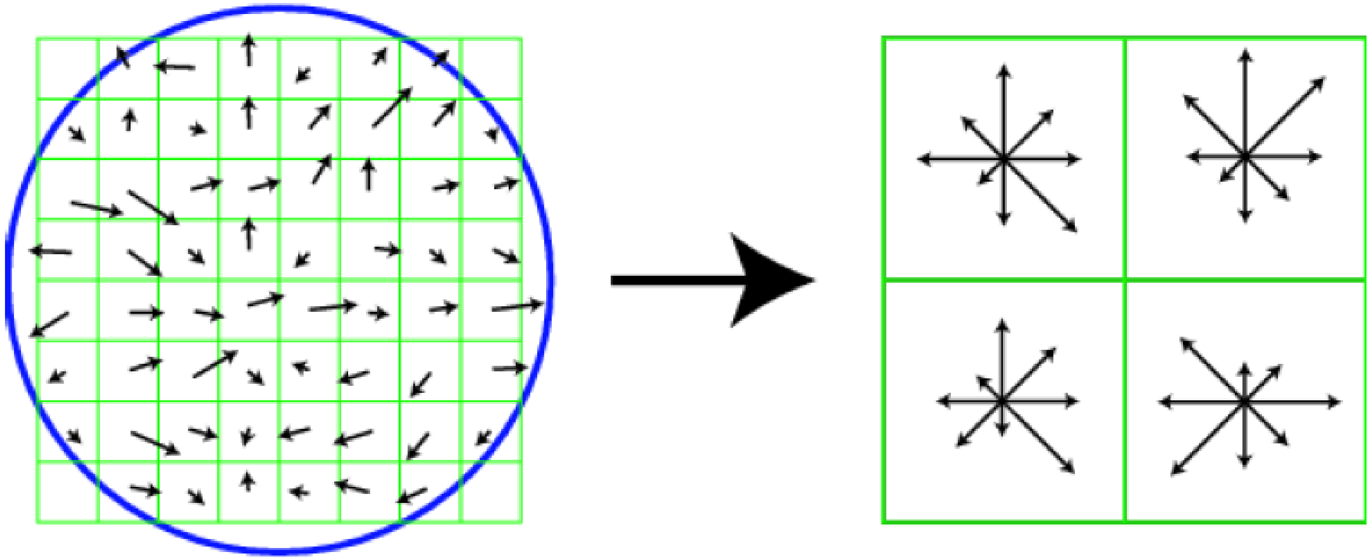


Image gradients

Keypoint descriptor

Scale Invariant Feature Transform: 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

https://www.vlfeat.org/overview/sift.html

Maximally Stable Extremal Regions: MSER

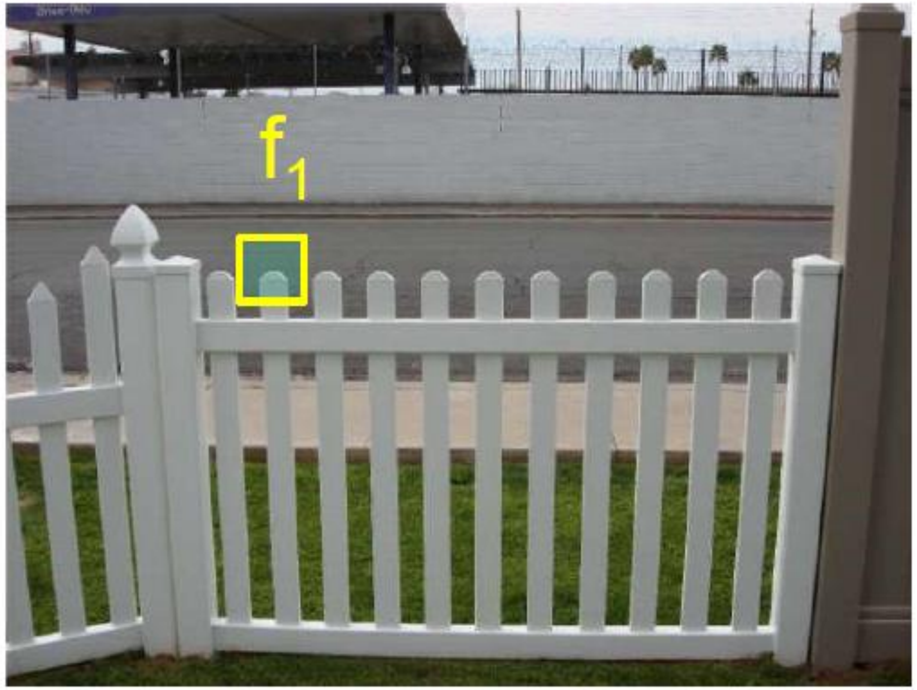
J.Matas et.al. “Distinguished Regions for Wide-baseline Stereo”. BMVC 2002.

- Maximally Stable Extremal Regions
  - *Threshold* image intensities: *I > thresh* for several increasing values of thresh
  - Extract *connected components* (“Extremal Regions”)
  - Find a threshold when region is “Maximally Stable”, i.e. *local minimum* of the relative growth
  - Approximate each region with an *ellipse*

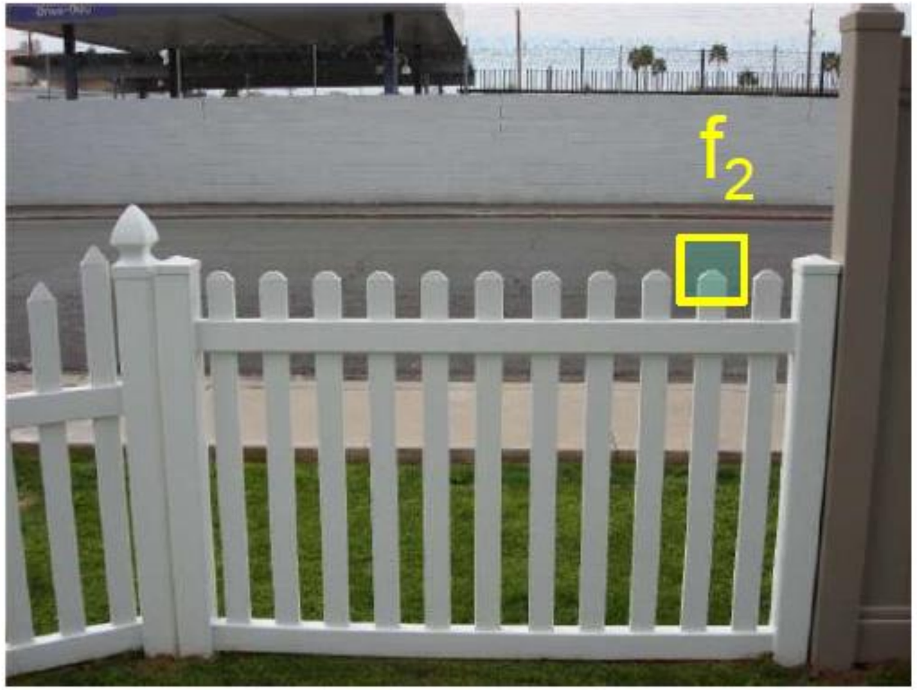
Feature distance

How to define the difference between two features f_1, f_2?

- Simple approach is SSD(f_1, f_2)
  - sum of square differences between entries of the two descriptors
  - can give good scores to very ambiguous (bad) matches



I_1

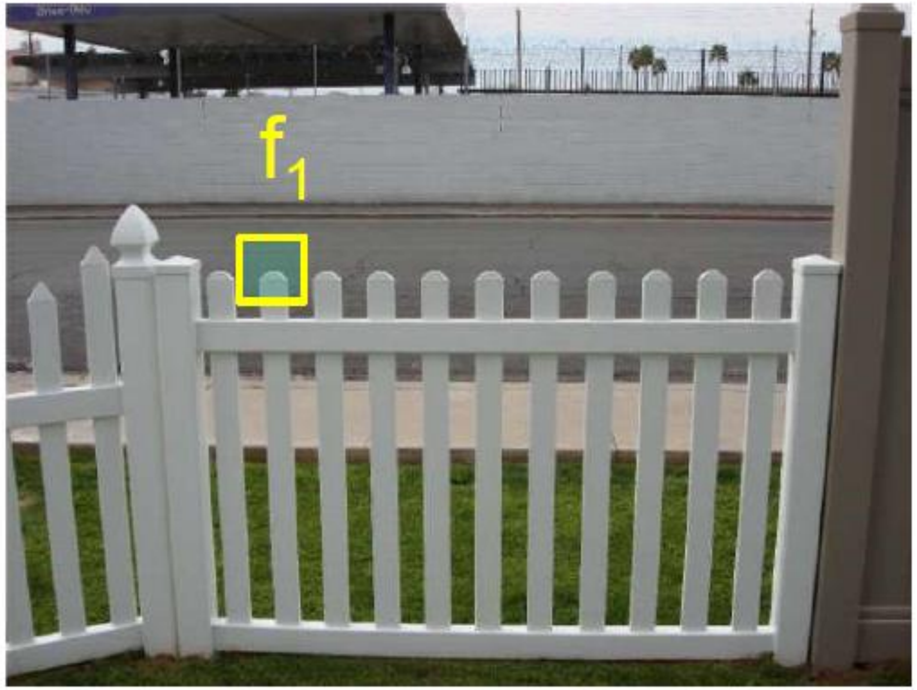


I_2

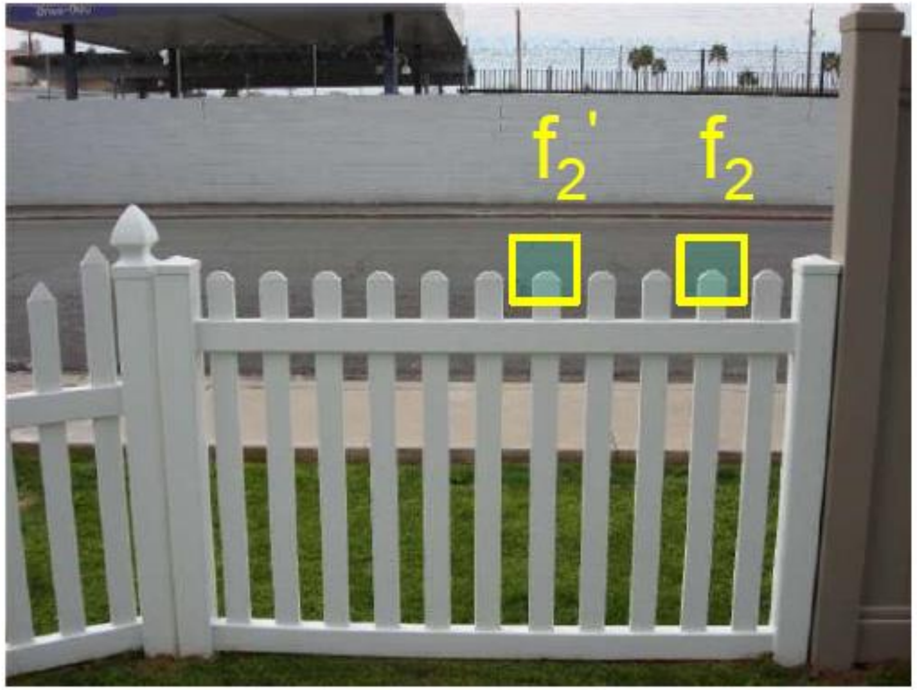
Feature distance

How to define the difference between two features f_1, f_2?

- Better approach: ratio distance = $SSD(f_1, f_2) / SSD(f_1, f_2')$
  - f_2 is best SSD match to f_1 in I_2
  - f_2' is 2^{nd} best SSD match to f_1 in I_2
  - gives small values for ambiguous matches



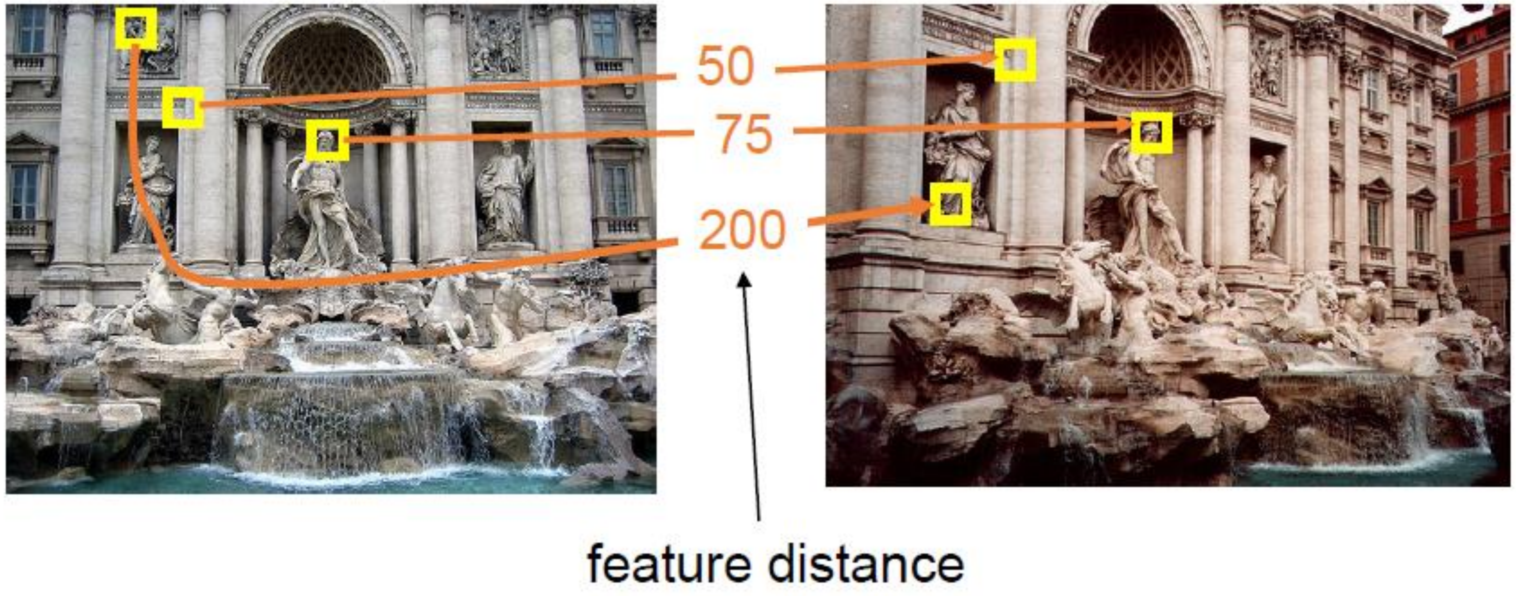
I_1



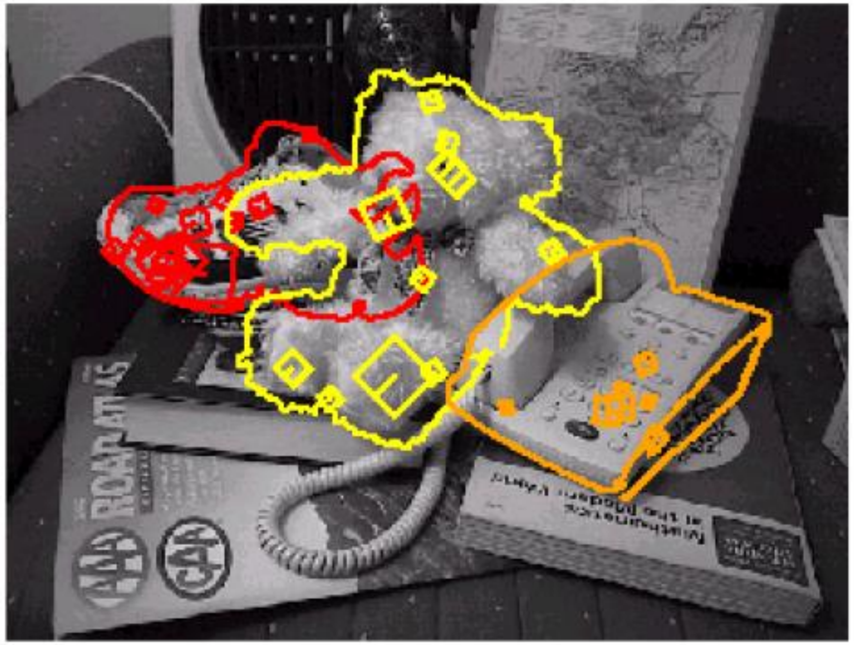
I_2

Evaluating the results

How can we measure the performance of a feature matcher?



Object recognition (David Lowe)



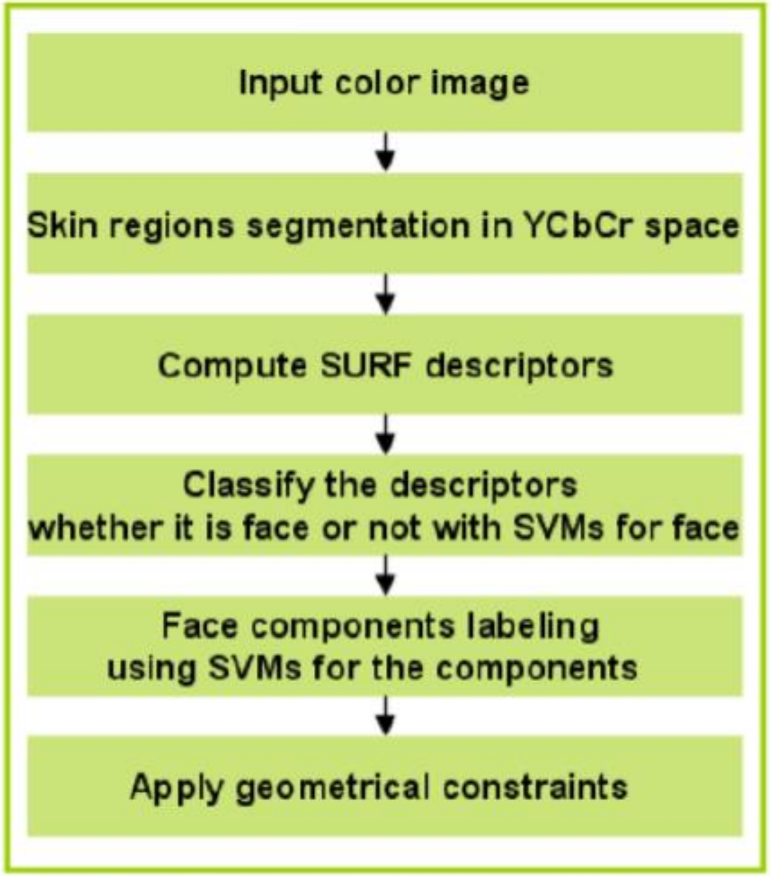


Figure 1. Overview of the proposed approach.

Figure 2. Skin color segmentation.

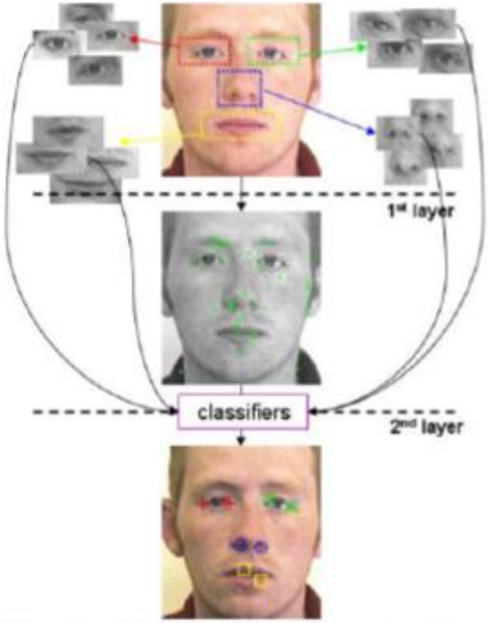


Figure 6. System overview of the facial components classifier (Red color plus: left eye, Green color cross: right eye, Blue color circle: nose, Yellow color rectangle: mouth).

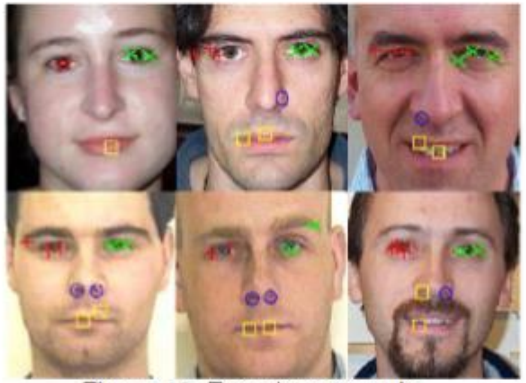
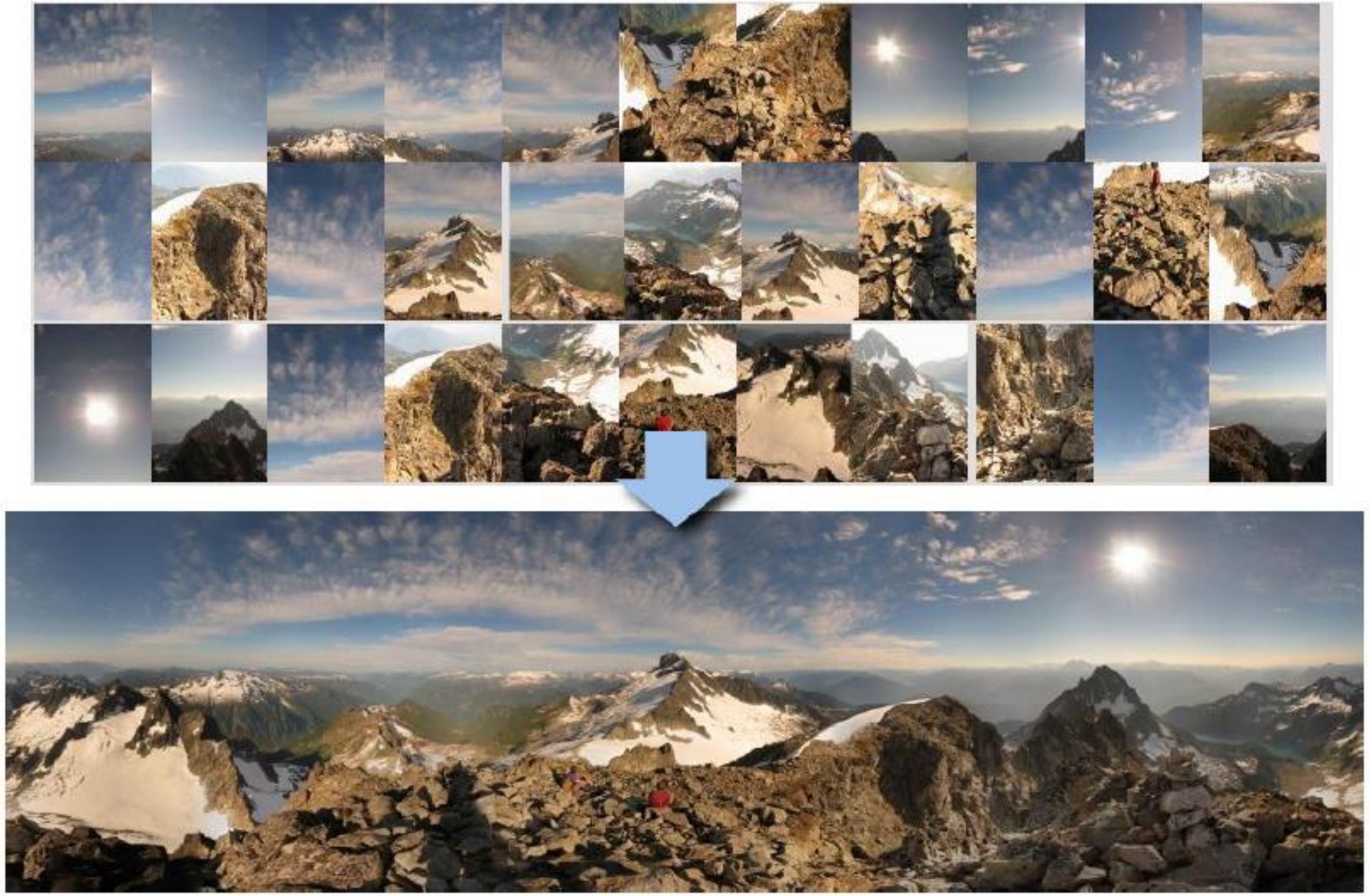


Figure 10. Experiment results

[Face components detection using SURF descriptor and SVMs](#)
Donghoon Kim and Rozenn Dahyot, International Machine Vision and Image Processing Conference, 2008 DOI:10.1109/IMVIP.2008.15

Figure 7. Geometrical constraint for nose position.

Motivation: Automatic panoramas



Credit: Matt Brown

Raquel Urtasun (lecturenotes)

https://www.cs.toronto.edu/~urtasun/courses/CV/lecture04.pdf

Why extract features?

How to combine these two images to form a panorama?

Figure: Two images

Raquel Urtasun (lecturenotes)

https://www.cs.toronto.edu/~urtasun/courses/CV/lecture04.pdf

Why extract features?

How to combine these two images to form a panorama?

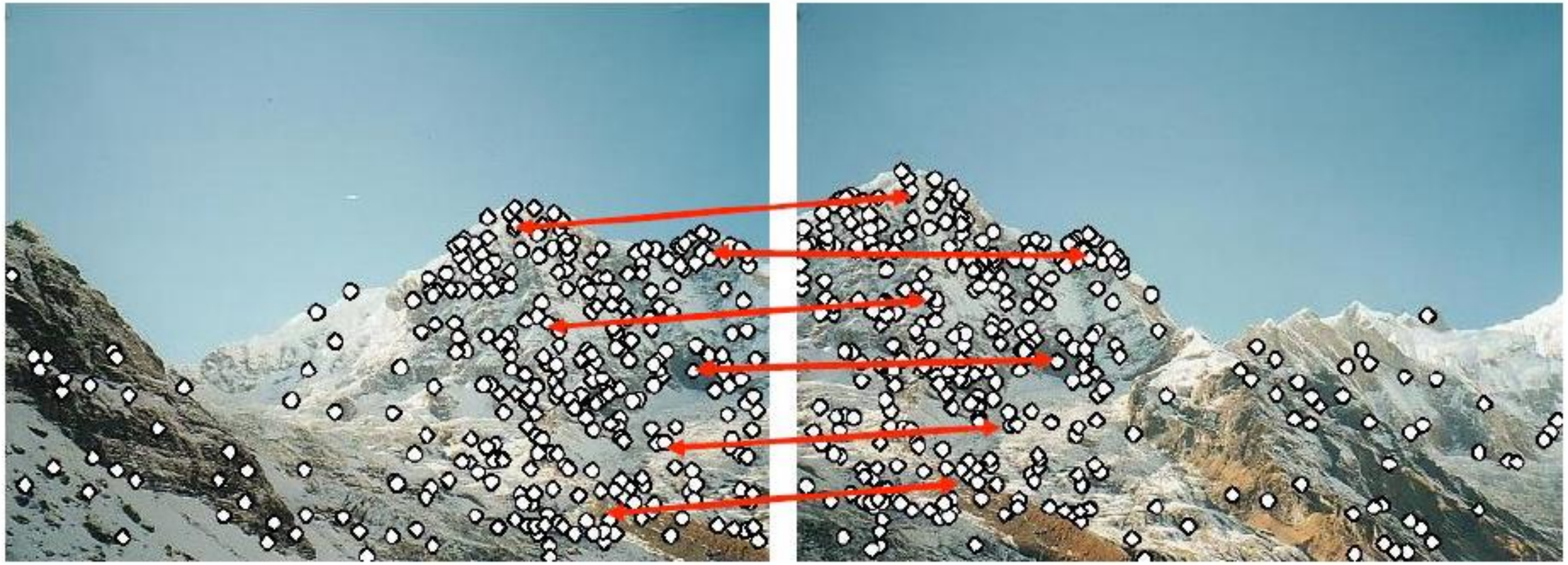


Figure: Feature extraction and matching

Raquel Urtasun (lecturenotes)

https://www.cs.toronto.edu/~urtasun/courses/CV/lecture04.pdf

Why extract features?

How to combine these two images to form a panorama?

Figure: Image aligment

Raquel Urtasun (lecturenotes)

https://www.cs.toronto.edu/~urtasun/courses/CV/lecture04.pdf

Summary: Features in computer vision

Descriptors:

- HOG: Histogram of Oriented Gradient
- Haar Wavelets
- Harris
- SIFT: Scale-invariant feature transform
- SURF: Speeded Up Robust Features https://www.vision.ee.ethz.ch/~surf/eccv06.pdf
- MSER: Maximally Stable Extremal Regions http://www.vlfeat.org/overview/mser.html
- BRISK: Binary Robust Invariant Scalable Keypoints https://www.robots.ox.ac.uk/~vgg/rg/papers/brisk.pdf

Applications:

- Image alignment (e.g., mosaics)
- 3D reconstruction
- Motion tracking
- Object recognition
- Indexing and database retrieval
- Robot navigation

...

https://github.com/cvg/Hierarchical-Localization/