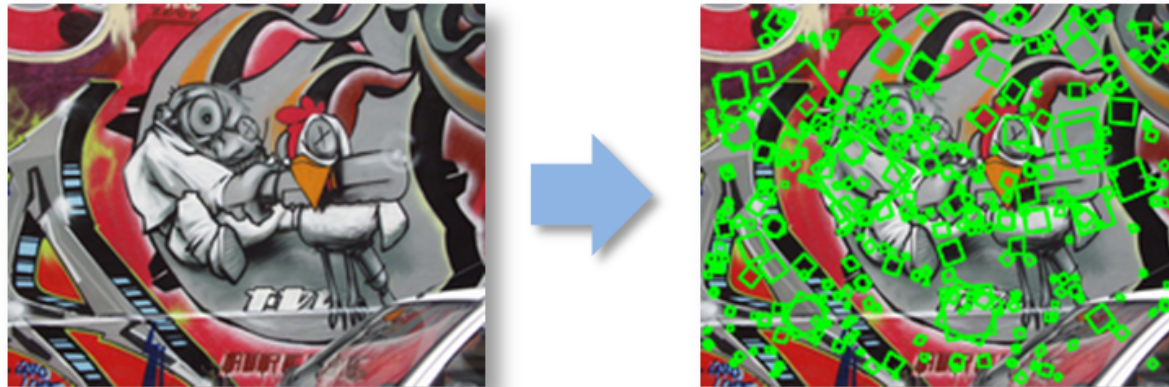


CS4670/5670: Computer Vision

Kavita Bala

Lecture 9: Feature Detection



Announcements

- Final schedule announced
- PA 1(A) is PA 1
- Will shuffle HW 1 and PA 2 a bit
 - Stay tuned
- Tape us on Friday

Reading

- Szeliski: 4.1

<http://gigapixelartzoom.com>

Motivation: Automatic panoramas



HD View

<http://research.microsoft.com/en-us/um/redmond/groups/ivm/HDView/HDGigapixel.htm>

Also see GigaPan:

<http://gigapan.org/>

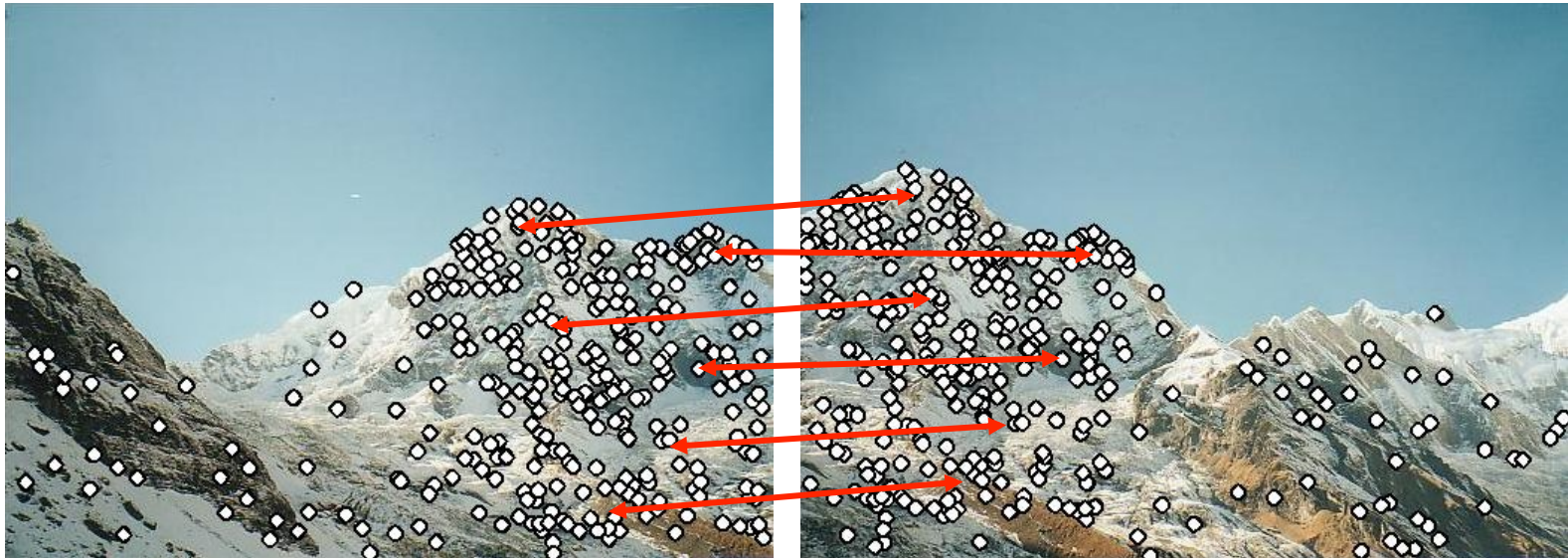
Why extract features?

- Motivation: panorama stitching
 - We have two images – how do we combine them?



Why extract features?

- Motivation: panorama stitching
 - We have two images – how do we combine them?

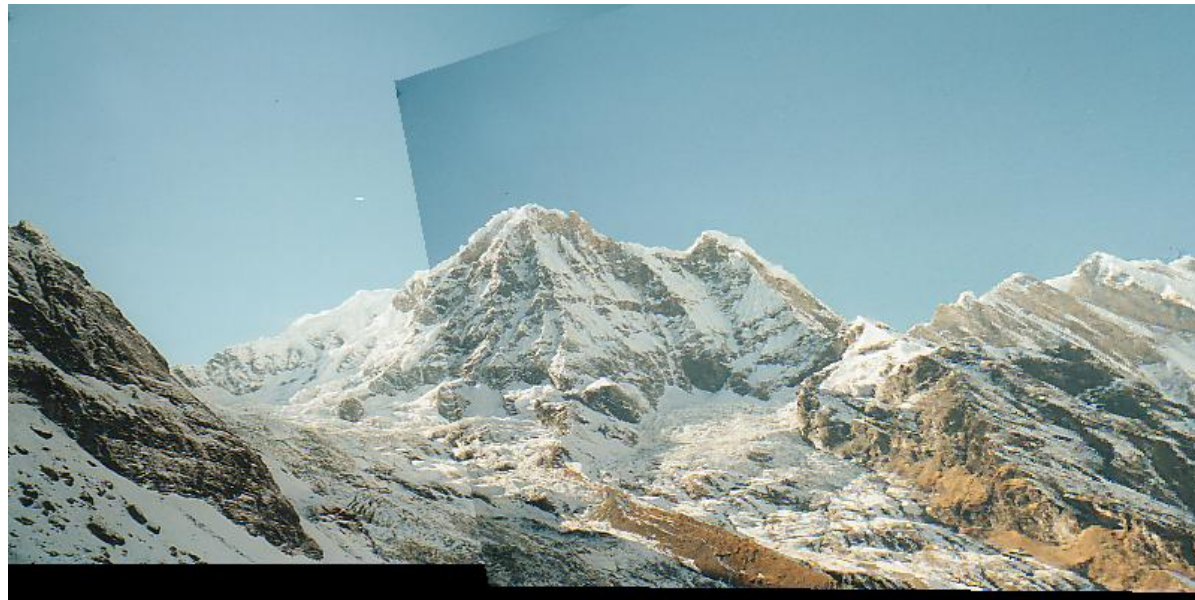


Step 1: extract features

Step 2: match features

Why extract features?

- Motivation: panorama stitching
 - We have two images – how do we combine them?



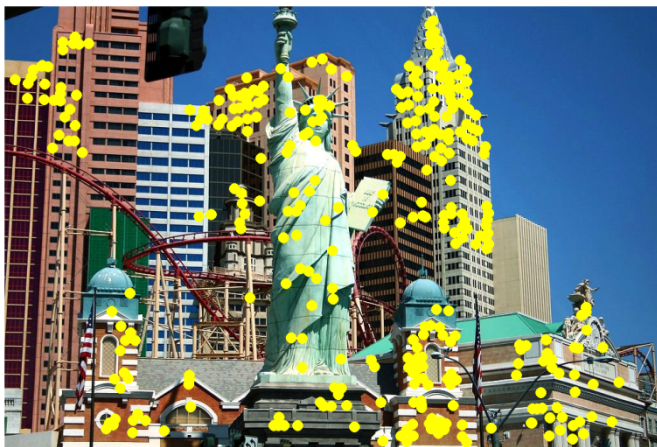
Step 1: extract features

Step 2: match features

Step 3: align images

Applications

- Feature points are used for:
 - Image alignment
 - 3D reconstruction
 - Object recognition
 - Motion tracking
 - Robot navigation



Matching can be challenging



Harder case

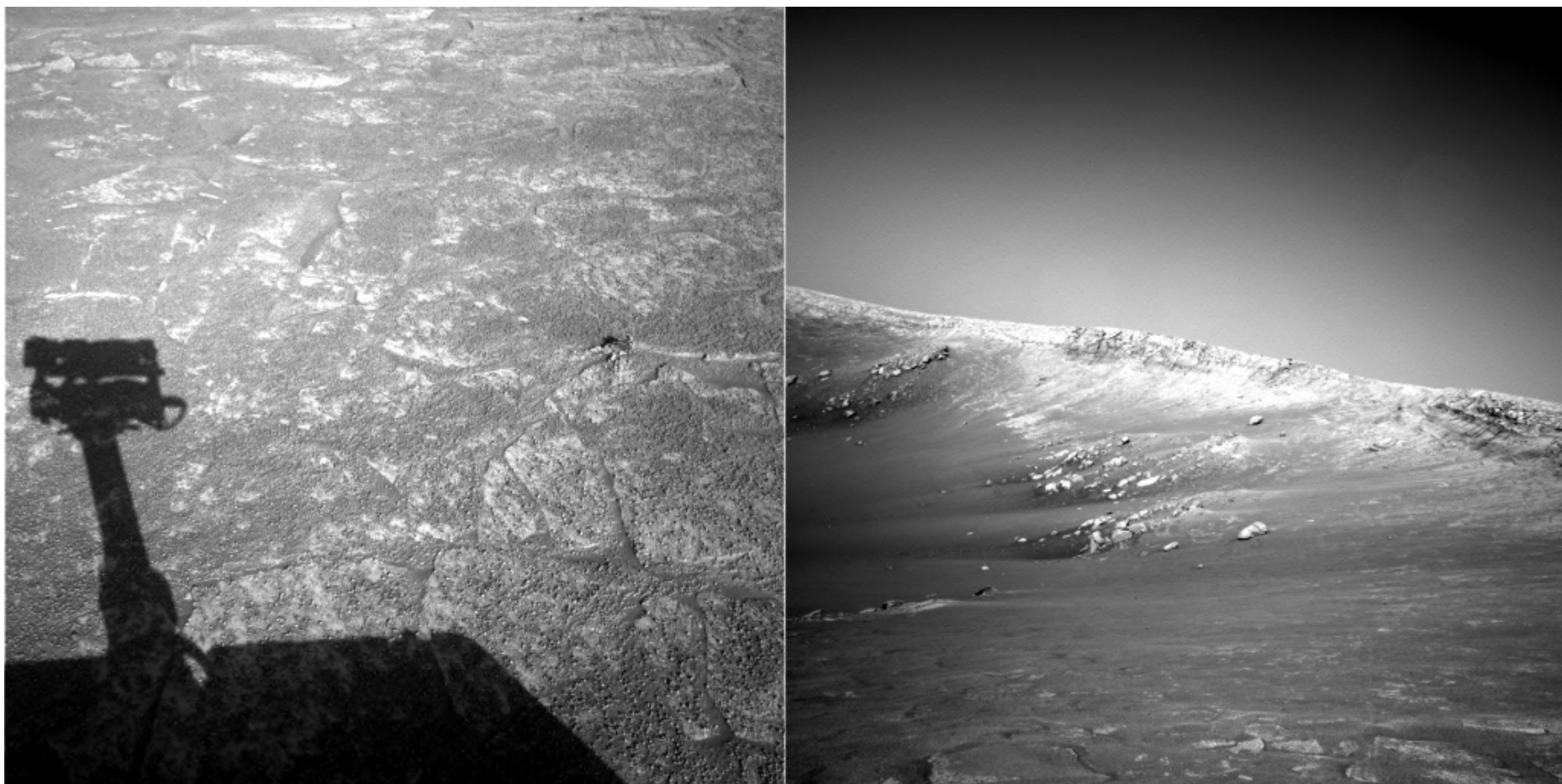


by [Diva Sian](#)

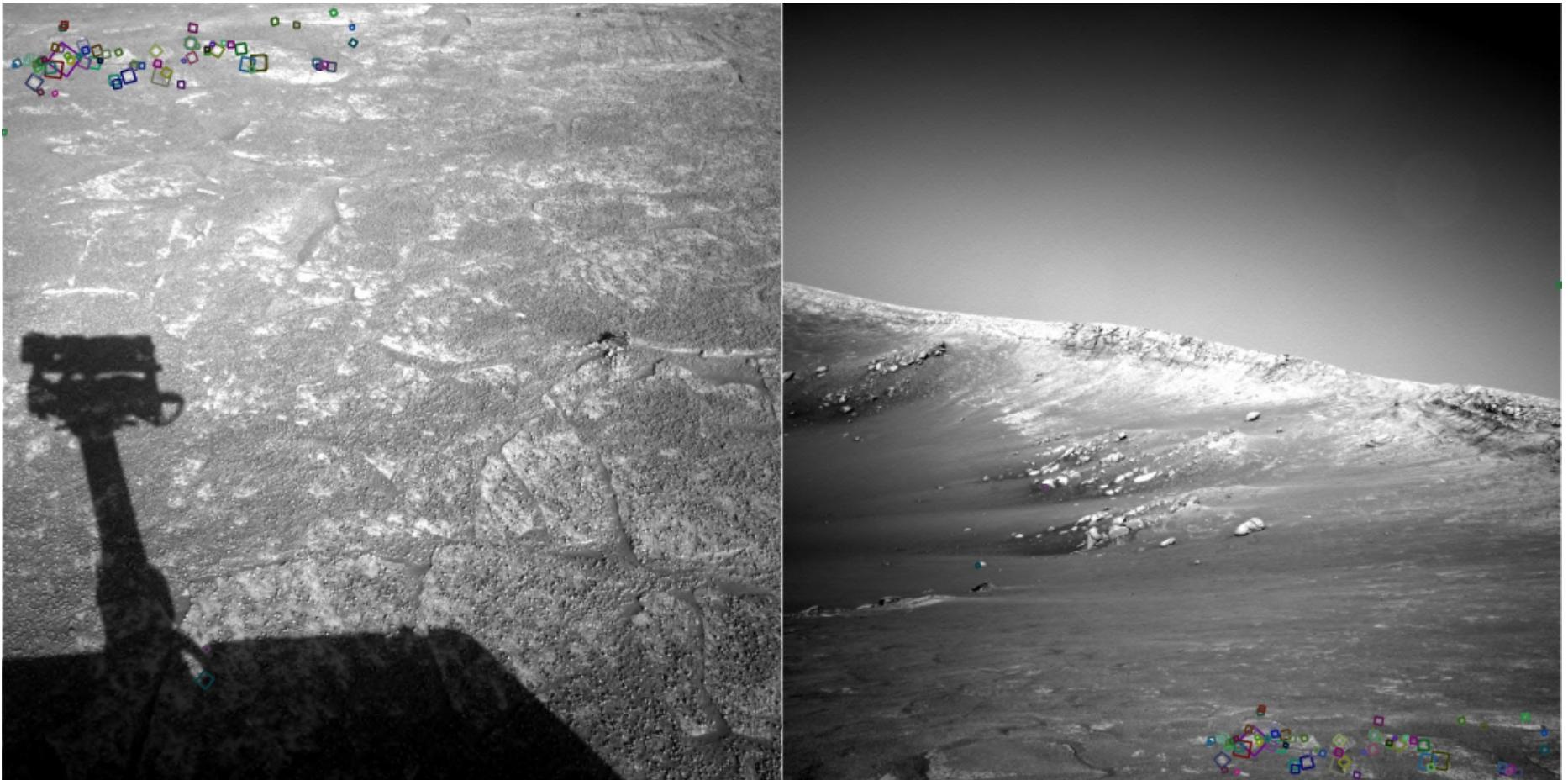


by [scgbt](#)

Harder still?



Answer below (look for tiny colored squares...)



NASA Mars Rover images
with SIFT feature matches

Approach

Feature detection: find it

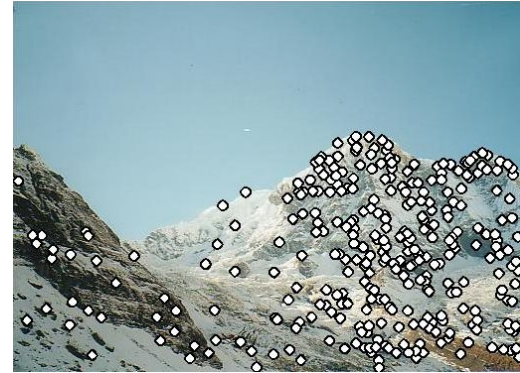
Feature descriptor: represent it

Feature matching: match it

Feature tracking: track it, when motion

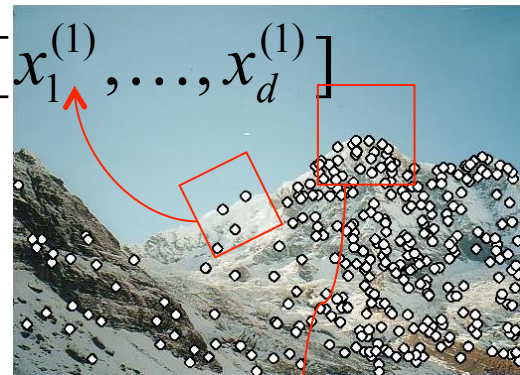
Local features: main components

1) Detection: Identify the interest points



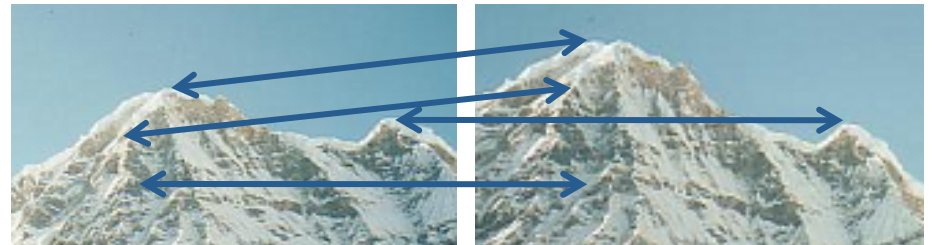
2) Description: Extract vector feature descriptor surrounding each interest point.

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

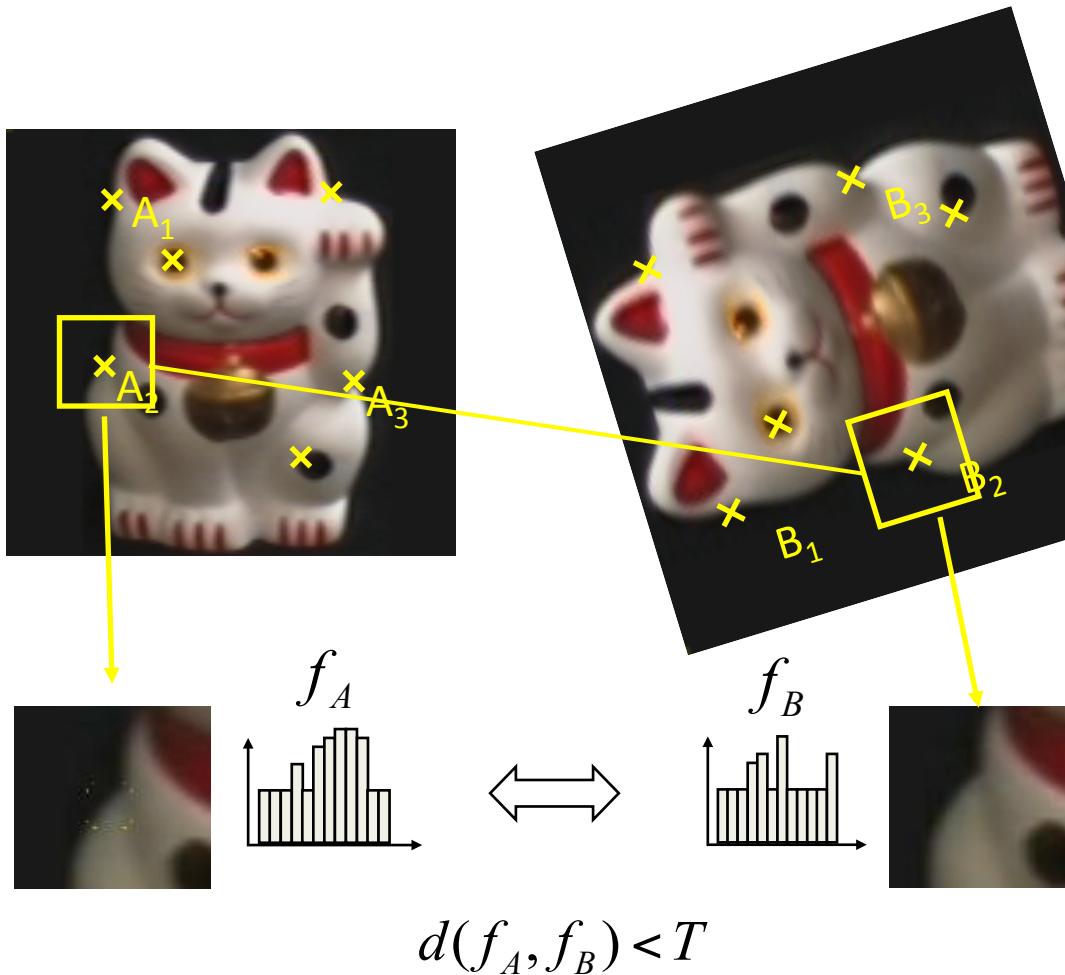


$$\mathbf{x}_2 = [x_1^{(2)}, \dots, x_d^{(2)}]$$

3) Matching: Determine correspondence between descriptors in two views



Overview



1. Detection:
distinctive features

2. Descriptor:

(a) Define region around each feature

(b) Extract and normalize region content

(c) Compute local descriptor from normalized region

3. Match local descriptors

Goals for Features



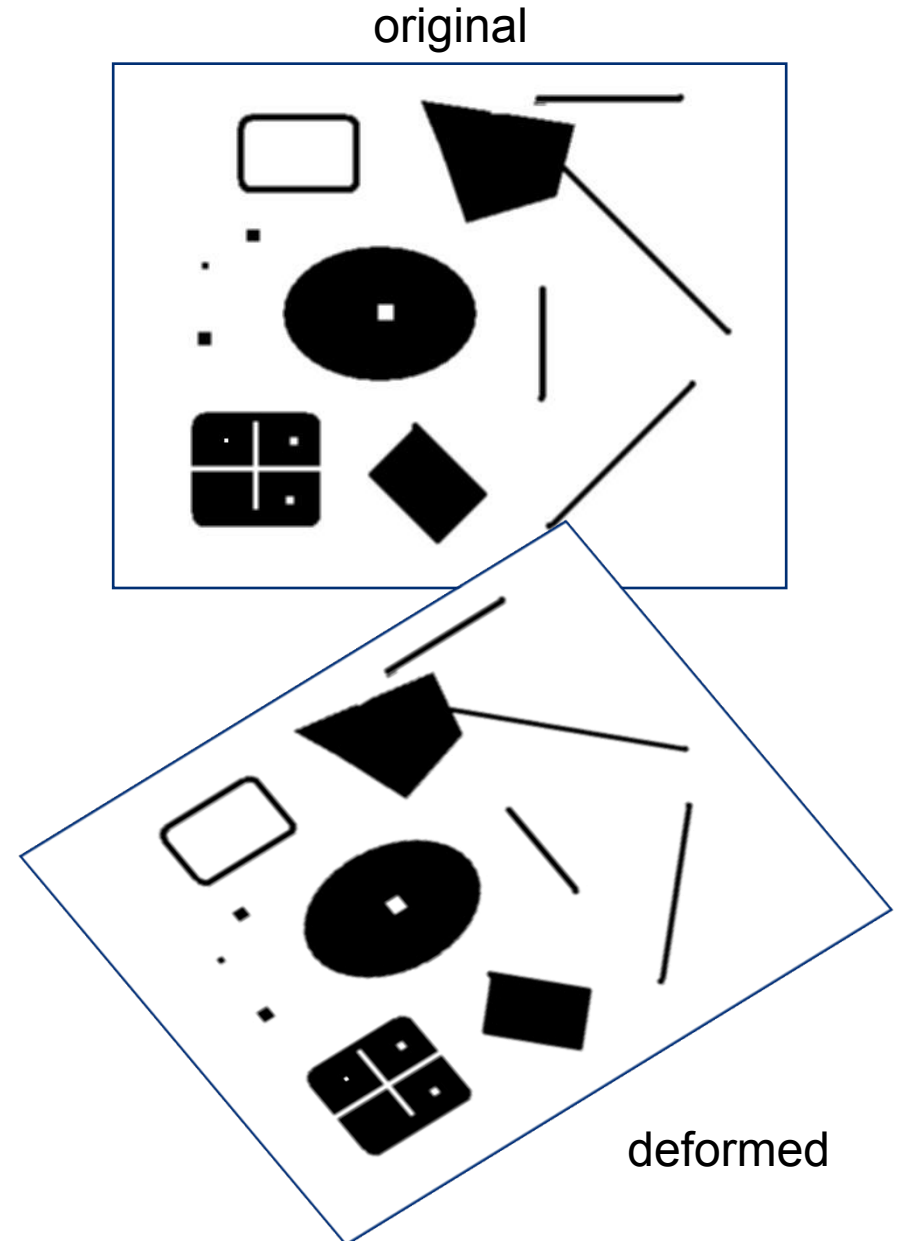
Detect points that are *repeatable* and *distinctive*

What makes a good feature?



What is a good feature?

- 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?



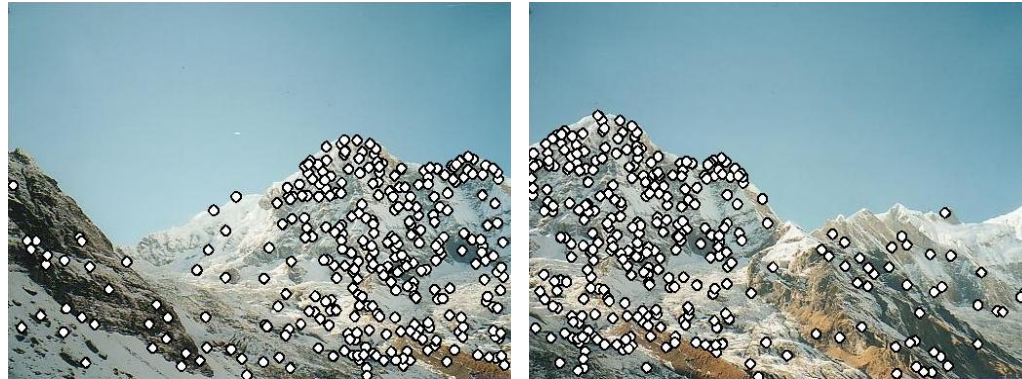
Want uniqueness

Look for image regions that are unusual

- Lead to unambiguous matches in other images

How to define “unusual”?

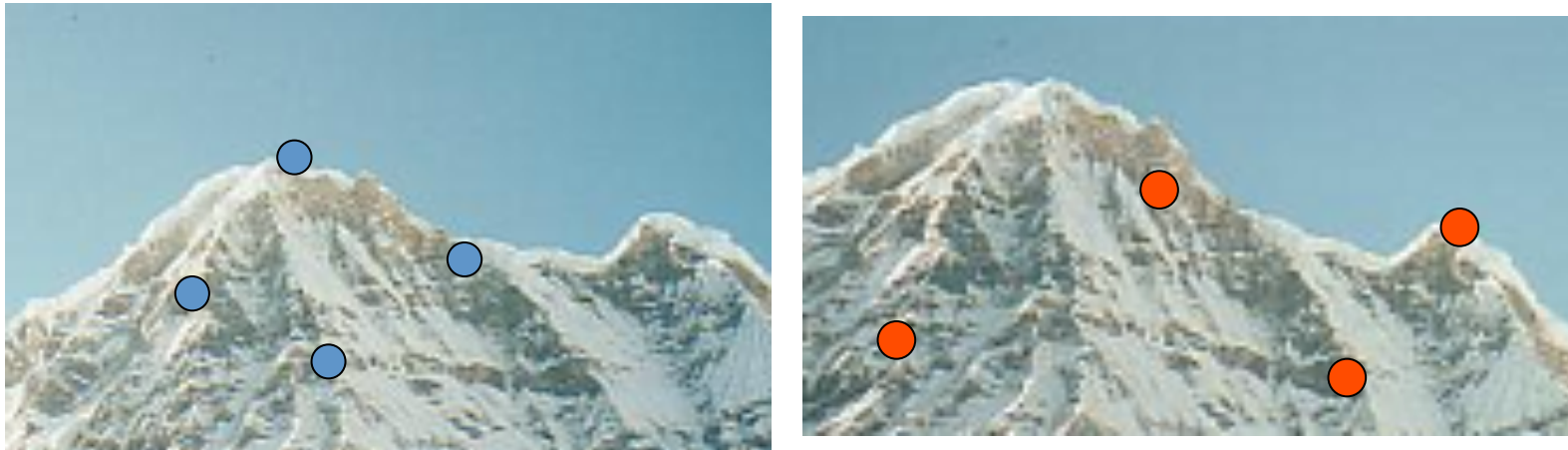
Characteristics of good features



- **Repeatability**
 - The same feature can be found in several images despite geometric and photometric transformations
- **Saliency**
 - Each feature is distinctive
- **Compactness and efficiency**
 - Many fewer features than image pixels
- **Locality**
 - A feature occupies a relatively small area of the image; robust to clutter and occlusion

Goal: repeatability

- We want to detect (at least some of) the same points in both images.

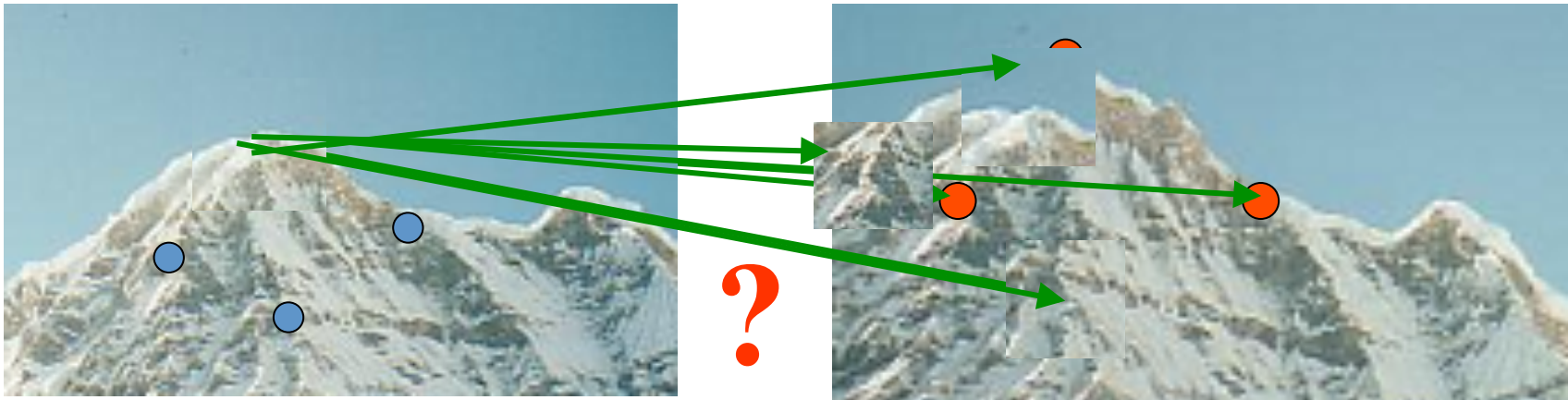


No chance to find true matches!

- Yet we have to be able to run the detection procedure *independently* per image.

Goal: descriptor distinctiveness

- We want to be able to reliably determine which point goes with which.

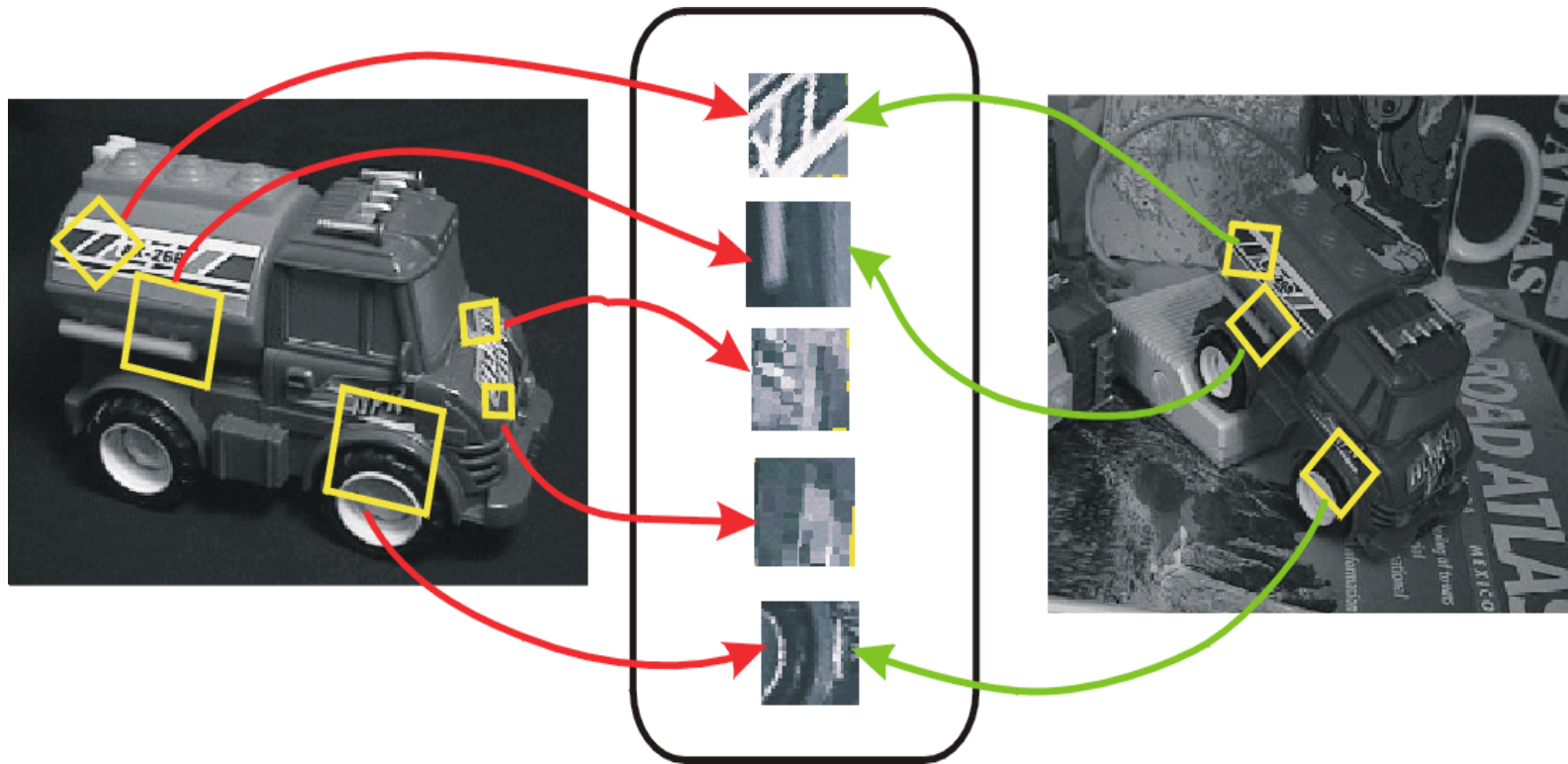


- Must provide some invariance to geometric and photometric differences between the two views.

Invariant local features

Find features that are invariant to transformations

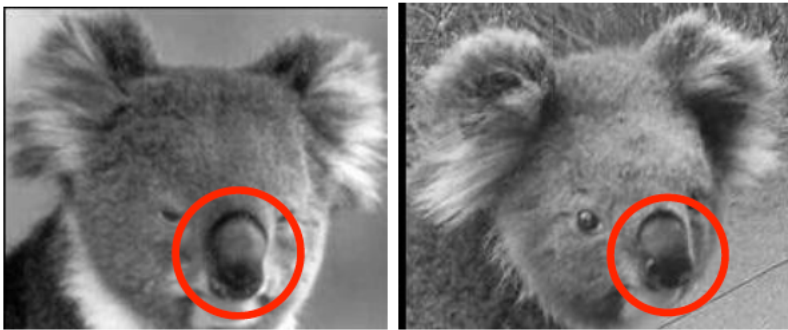
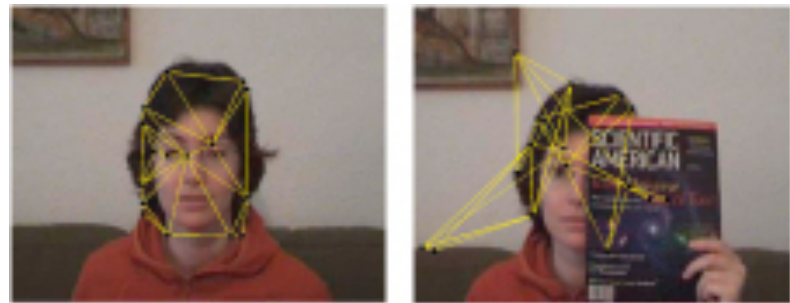
- geometric invariance: translation, rotation, scale
- photometric invariance: brightness, exposure, ...



Feature Descriptors

Features

- Global vs. local representations
- Local: describe and match local regions
- Robust to
 - Occlusions
 - Articulation
 - Intra-category variation



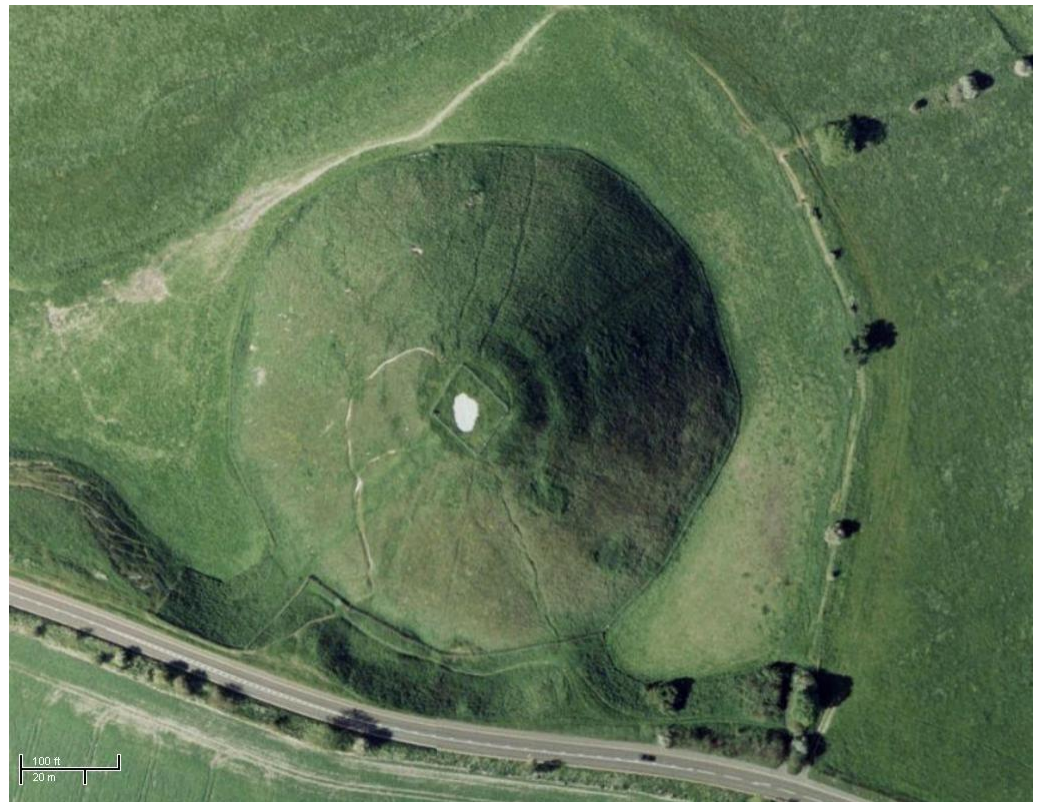
Fei-Fei Li

Feature Detection

Where would you
tell your friend to
meet you?



Where would you
tell your friend to
meet you?



Choosing distinctive interest points

- If you wanted to meet a friend would you say
 - a) “Let’s meet on campus.”
 - b) “Let’s meet on Green street.”
 - c) “Let’s meet at Green and Wright.”– Corner detection

- Or if you were in a secluded area:
 - a) “Let’s meet in the Plains of Akbar.”
 - b) “Let’s meet on the side of Mt. Doom.”
 - c) “Let’s meet on top of Mt. Doom.”– Blob (valley/peak) detection

Many Existing Detectors Available

Hessian & Harris	[Beaudet '78], [Harris '88]
Laplacian, DoG	[Lindeberg '98], [Lowe 1999]
Harris-/Hessian-Laplace	[Mikolajczyk & Schmid '01]
Harris-/Hessian-Affine	[Mikolajczyk & Schmid '04]
EBR and IBR	[Tuytelaars & Van Gool '04]
MSER	[Matas '02]
Salient Regions	[Kadir & Brady '01]
Others...	

Finding Corners

- Key property: in the region around a corner, image gradient has two or more dominant directions
- Corners are repeatable and distinctive

C.Harris and M.Stephens. 1988. ["A Combined Corner and Edge Detector."](#)
Proceedings of the 4th Alvey Vision Conference: pages 147--151.

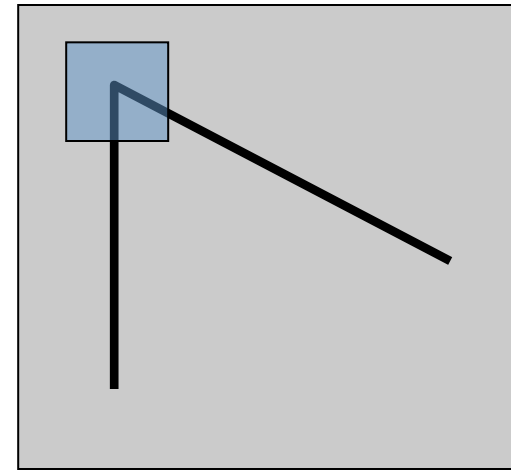
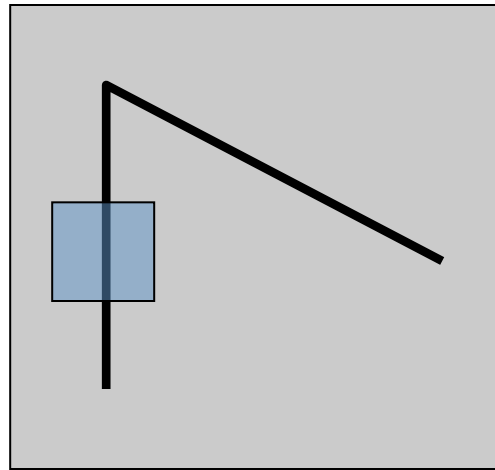
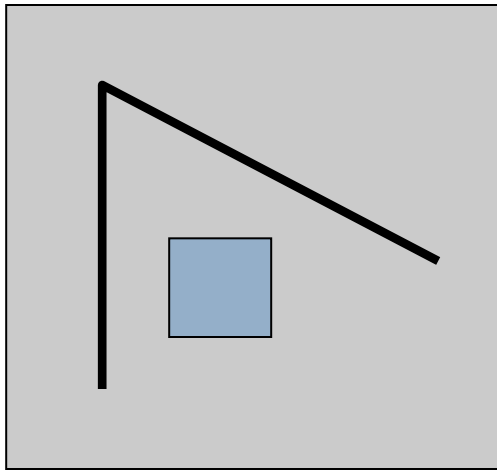
Feature extraction: Corners



Local measures of uniqueness

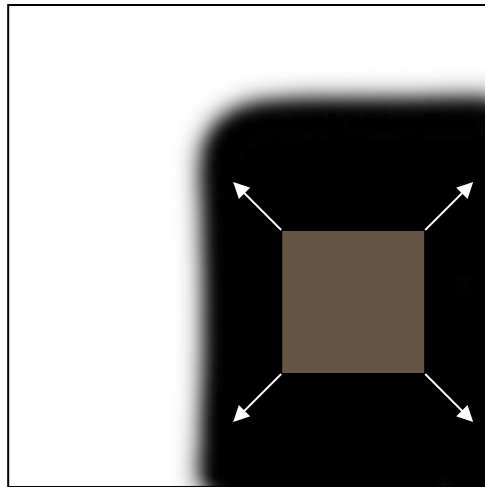
Suppose we only consider a small window of pixels

- What defines whether a feature is a good or bad candidate?

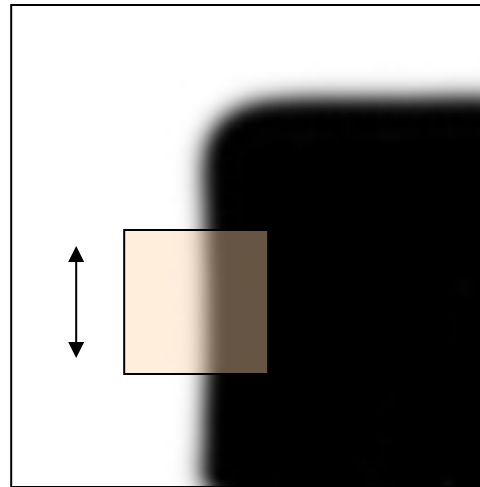


Corner Detection: Basic Idea

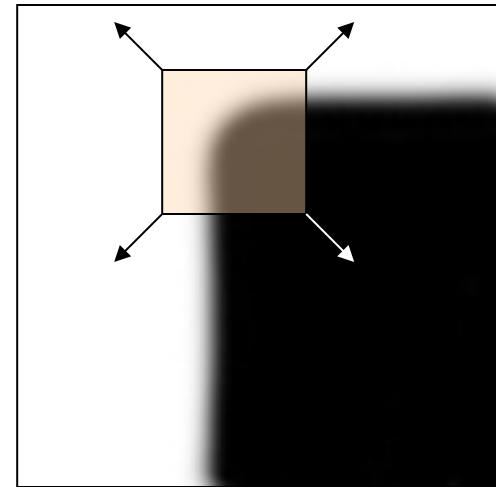
- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give a *large change* in intensity



“flat” region:
no change in
all directions



“edge”:
no change along
the edge
direction



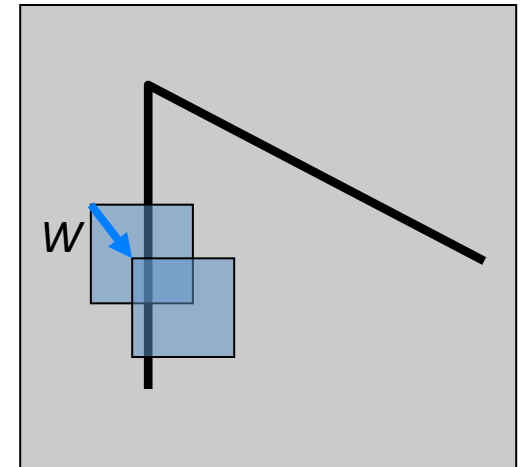
“corner”:
significant
change in all
directions

Harris corner detection: the math

Consider shifting the window W by (u, v)

- how do the pixels in W change?
- compare each pixel before and after by summing up the squared differences (SSD)
- this defines an SSD “error” $E(u, v)$:

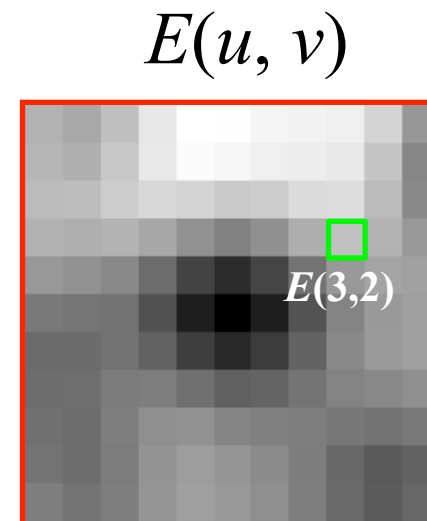
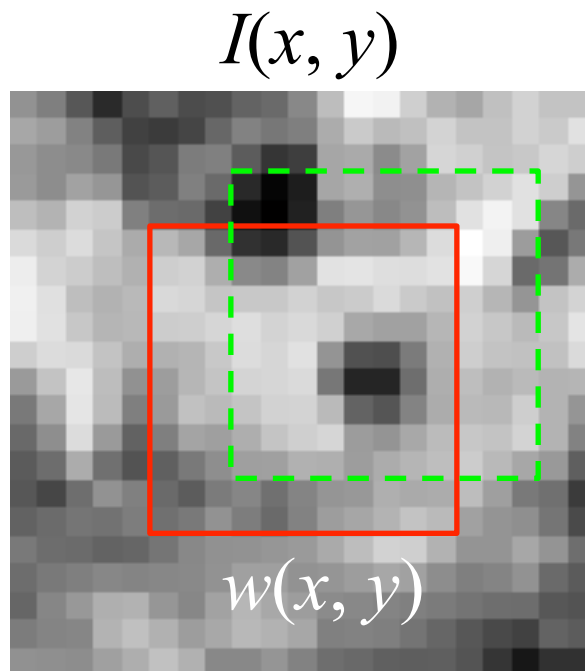
$$E(u, v) = \sum_{(x, y) \in W} [I(x + u, y + v) - I(x, y)]^2$$



Corner Detection: Mathematics

Change in appearance of window $w(x,y)$
for the shift $[u,v]$:

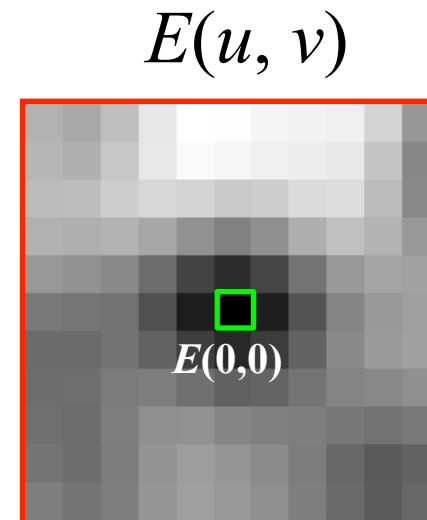
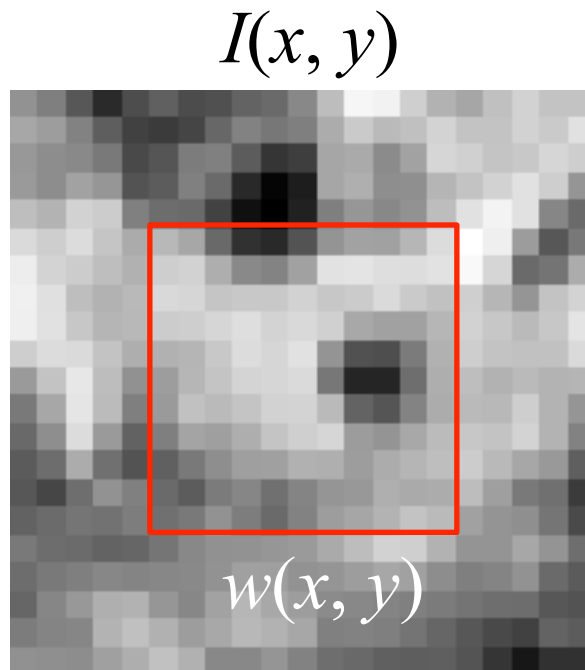
$$E(u, v) = \sum_{x,y} w(x, y) [I(x+u, y+v) - I(x, y)]^2$$



Corner Detection: Mathematics

Change in appearance of window $w(x,y)$
for the shift $[u,v]$:

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



Corner Detection: Mathematics

Change in appearance of window $w(x,y)$
for the shift $[u,v]$:

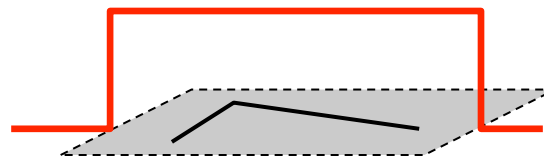
$$E(u, v) = \sum_{x,y} w(x, y) [I(x+u, y+v) - I(x, y)]^2$$

Window
function

Shifted
intensity

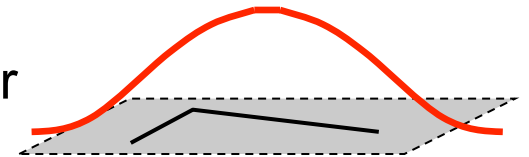
Intensity

Window function $w(x,y) =$



1 in window, 0 outside

or



Gaussian

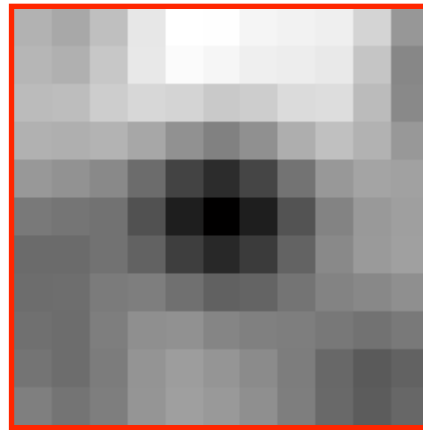
Corner Detection: Mathematics

Change in appearance of window $w(x,y)$
for the shift $[u,v]$:

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

We want to find out how this function behaves for small shifts

$E(u, v)$



Small motion assumption

Taylor Series expansion of I :

$$I(x+u, y+v) = I(x, y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \text{higher order terms}$$

If the motion (u,v) is small, then first order approximation is good

$$\begin{aligned} I(x+u, y+v) &\approx I(x, y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v \\ &\approx I(x, y) + [I_x \quad I_y] \begin{bmatrix} u \\ v \end{bmatrix} \end{aligned}$$

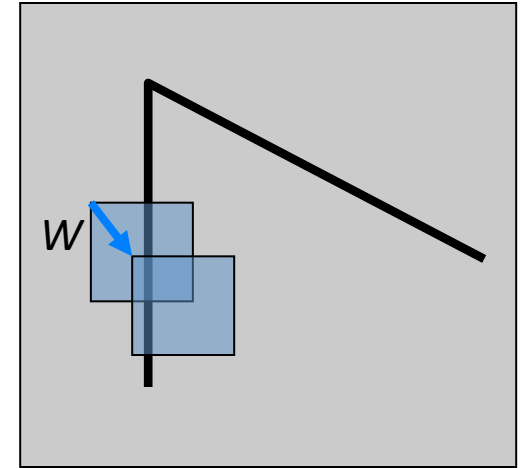
shorthand: $I_x = \frac{\partial I}{\partial x}$

Plugging this into the formula on the previous slide...

Corner detection: the math

Consider shifting the window W by (u, v)

- define an SSD “error” $E(u, v)$:



$$\begin{aligned} E(u, v) &= \sum_{(x, y) \in W} [I(x + u, y + v) - I(x, y)]^2 \\ &\approx \sum_{(x, y) \in W} [I(x, y) + I_x u + I_y v - I(x, y)]^2 \\ &\approx \sum_{(x, y) \in W} [I_x u + I_y v]^2 \end{aligned}$$

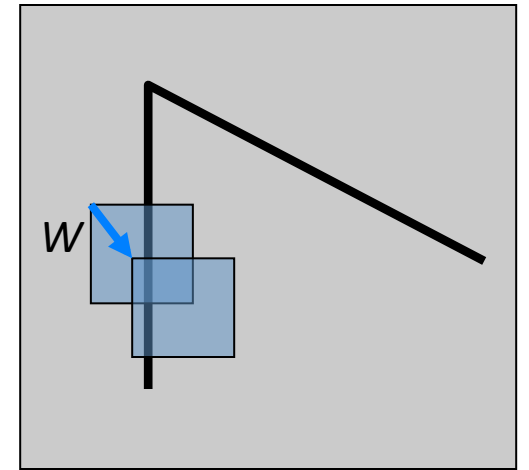
Corner detection: the math

Consider shifting the window W by (u, v)

- define an SSD “error” $E(u, v)$:

$$E(u, v) \approx \sum_{(x, y) \in W} [I_x u + I_y v]^2$$
$$\approx Au^2 + 2Buv + Cv^2$$

$$A = \sum_{(x, y) \in W} I_x^2 \quad B = \sum_{(x, y) \in W} I_x I_y \quad C = \sum_{(x, y) \in W} I_y^2$$



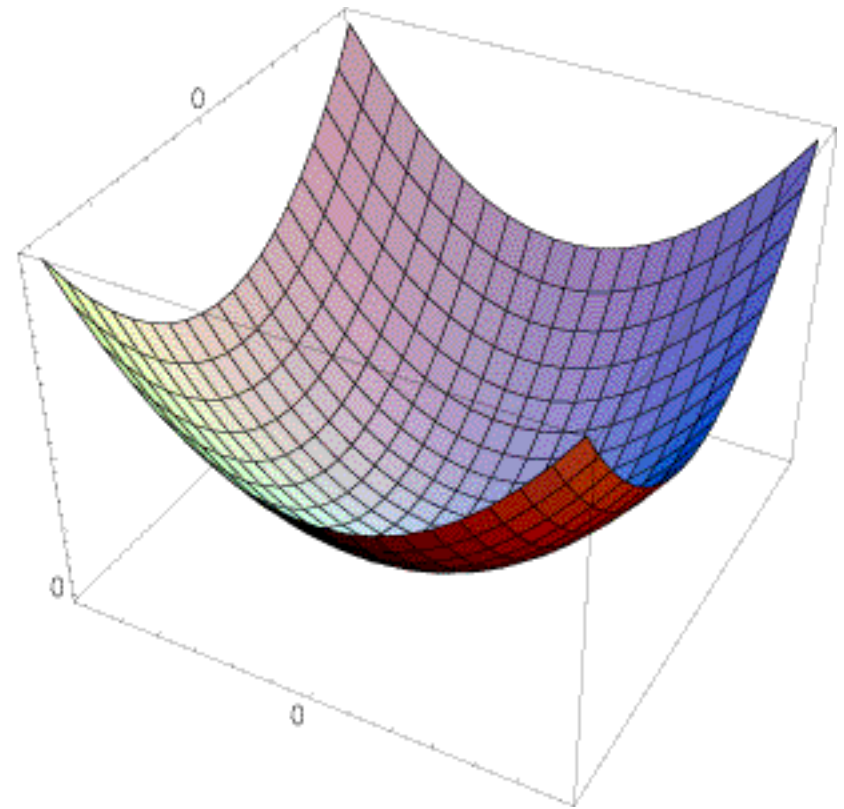
- Thus, $E(u, v)$ is locally approximated as a quadratic error function

Interpreting the second moment matrix

The surface $E(u, v)$ is locally approximated by a quadratic form. Let's try to understand its shape.

$$E(u, v) \approx [u \ v] M \begin{bmatrix} u \\ v \end{bmatrix}$$

$$M = \sum_{x, y} w(x, y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$



The second moment matrix

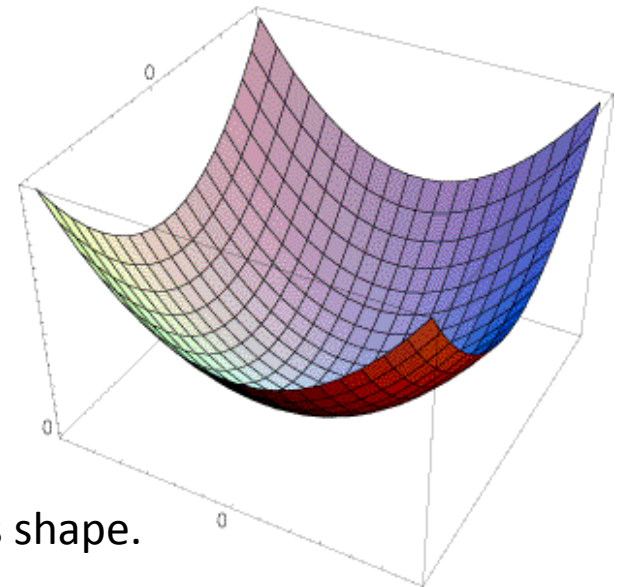
The surface $E(u,v)$ is locally approximated by a quadratic form.

$$E(u, v) \approx Au^2 + 2Buv + Cv^2$$
$$\approx \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} A & B \\ B & C \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

$$A = \sum_{(x,y) \in W} I_x^2$$

$$B = \sum_{(x,y) \in W} I_x I_y$$

$$C = \sum_{(x,y) \in W} I_y^2$$



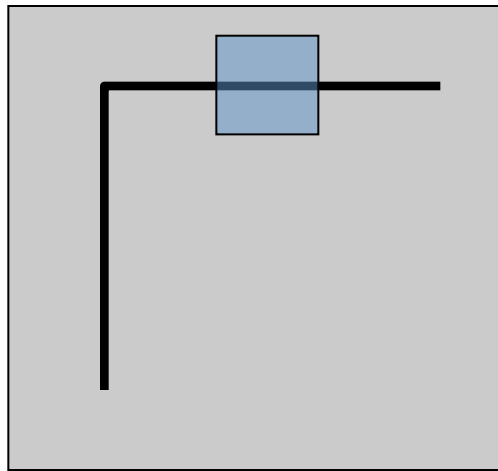
Let's try to understand its shape.

$$E(u, v) \approx \begin{bmatrix} u & v \end{bmatrix} \underbrace{\begin{bmatrix} A & B \\ B & C \end{bmatrix}}_M \begin{bmatrix} u \\ v \end{bmatrix}$$

$$A = \sum_{(x,y) \in W} I_x^2$$

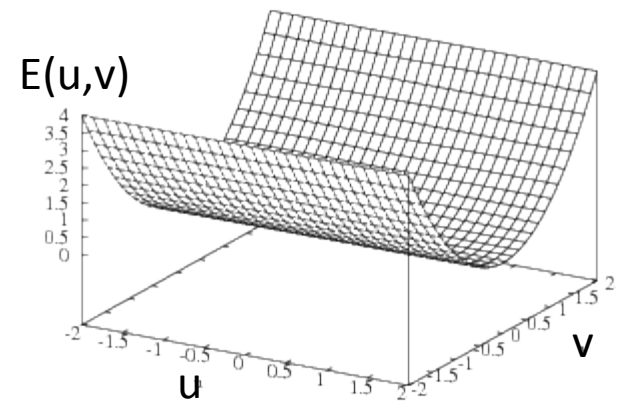
$$B = \sum_{(x,y) \in W} I_x I_y$$

$$C = \sum_{(x,y) \in W} I_y^2$$



Horizontal edge: $I_x = 0$

$$M = \begin{bmatrix} 0 & 0 \\ 0 & C \end{bmatrix}$$

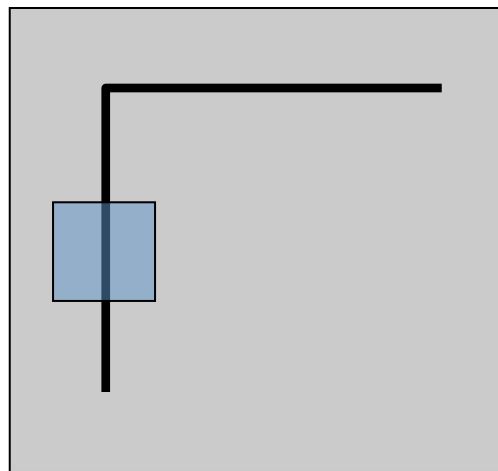


$$E(u, v) \approx \begin{bmatrix} u & v \end{bmatrix} \underbrace{\begin{bmatrix} A & B \\ B & C \end{bmatrix}}_M \begin{bmatrix} u \\ v \end{bmatrix}$$

$$A = \sum_{(x,y) \in W} I_x^2$$

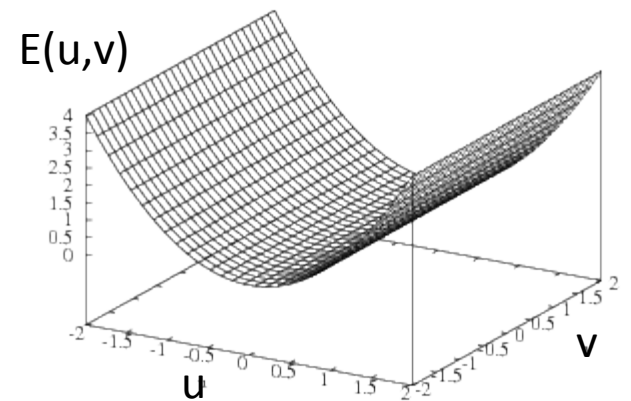
$$B = \sum_{(x,y) \in W} I_x I_y$$

$$C = \sum_{(x,y) \in W} I_y^2$$



Vertical edge: $I_y = 0$

$$M = \begin{bmatrix} A & 0 \\ 0 & 0 \end{bmatrix}$$



Interpreting the second moment matrix

Consider a horizontal “slice” of $E(u, v)$: $[u \ v] M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const}$

This is the equation of an ellipse.

