



UNIVERSITÀ DI PARMA

# Features Extraction



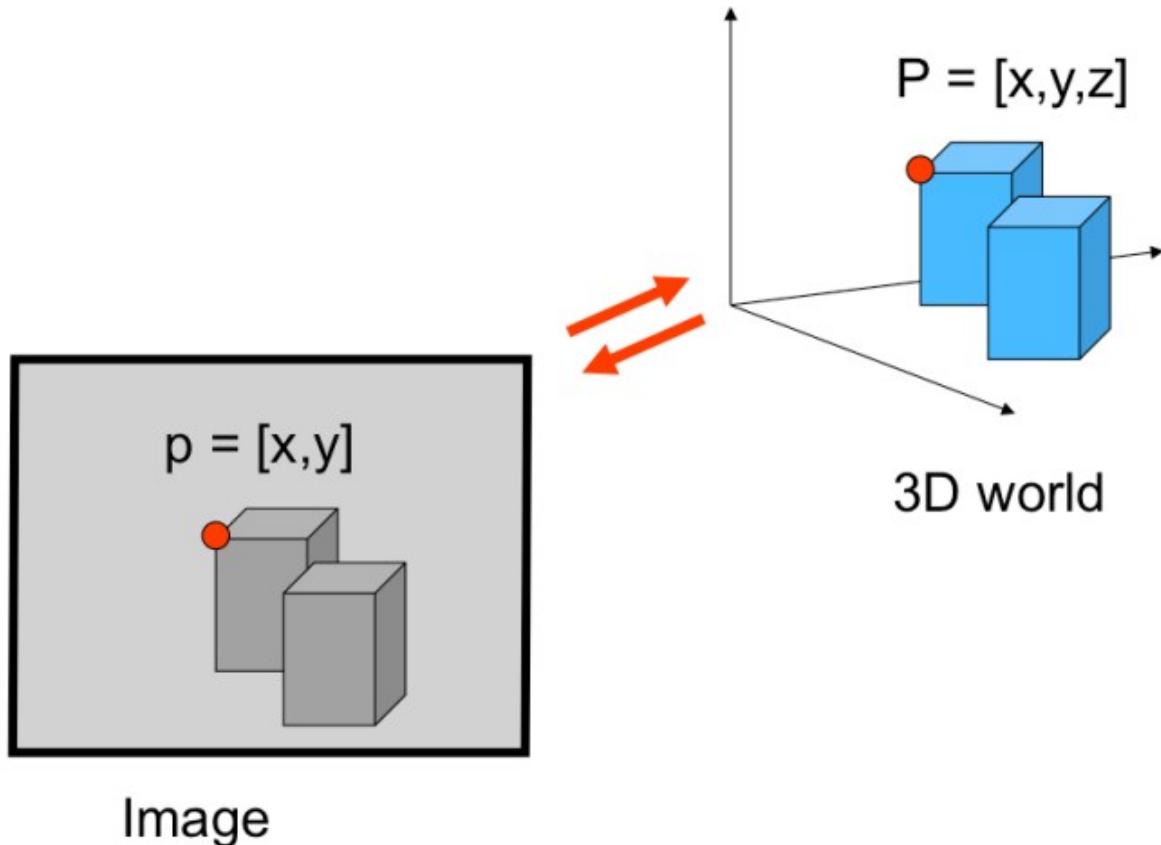
# Summary



- What is a feature?
- Keypoints ideal characteristics
- Harris Corners
- Scale Invariant Detectors

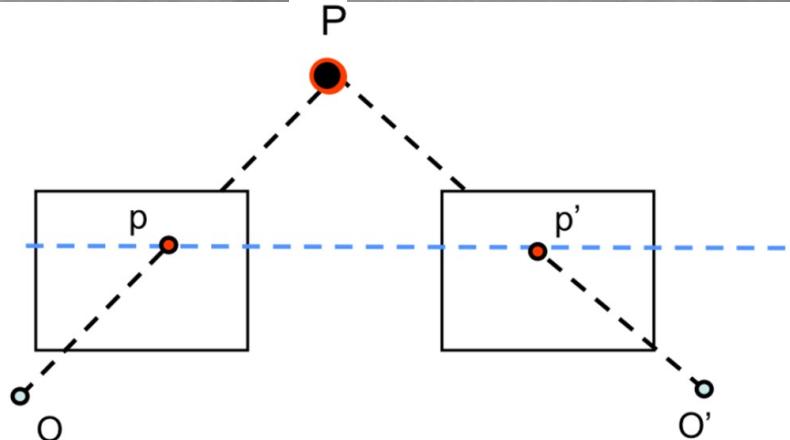
- [FP] D. A. Forsyth and J. Ponce. **Computer Vision: A Modern Approach** (2nd Edition). Prentice Hall, 2011.
- **CS231A · Computer Vision: from 3D reconstruction to recognition**, Prof. Silvio Savarese – Stanford University
- **CS131 · Computer Vision: Foundations and Applications**, Prof. Fei-Fei Li – Stanford University
- **Elementi di Analisi per Visione Artificiale**
  - Paolo Medici <http://www.ce.unipr.it/people/medici>

# Small Recap



- We found a geometrical relation between world and image

# Small Recap



- We found a geometrical relation between world and image

# Beyond Stereo Vision

UNIVERSITÀ  
DI PARMA



# Beyond Stereo Vision

UNIVERSITÀ  
DI PARMA



by [Diva Sian](#)



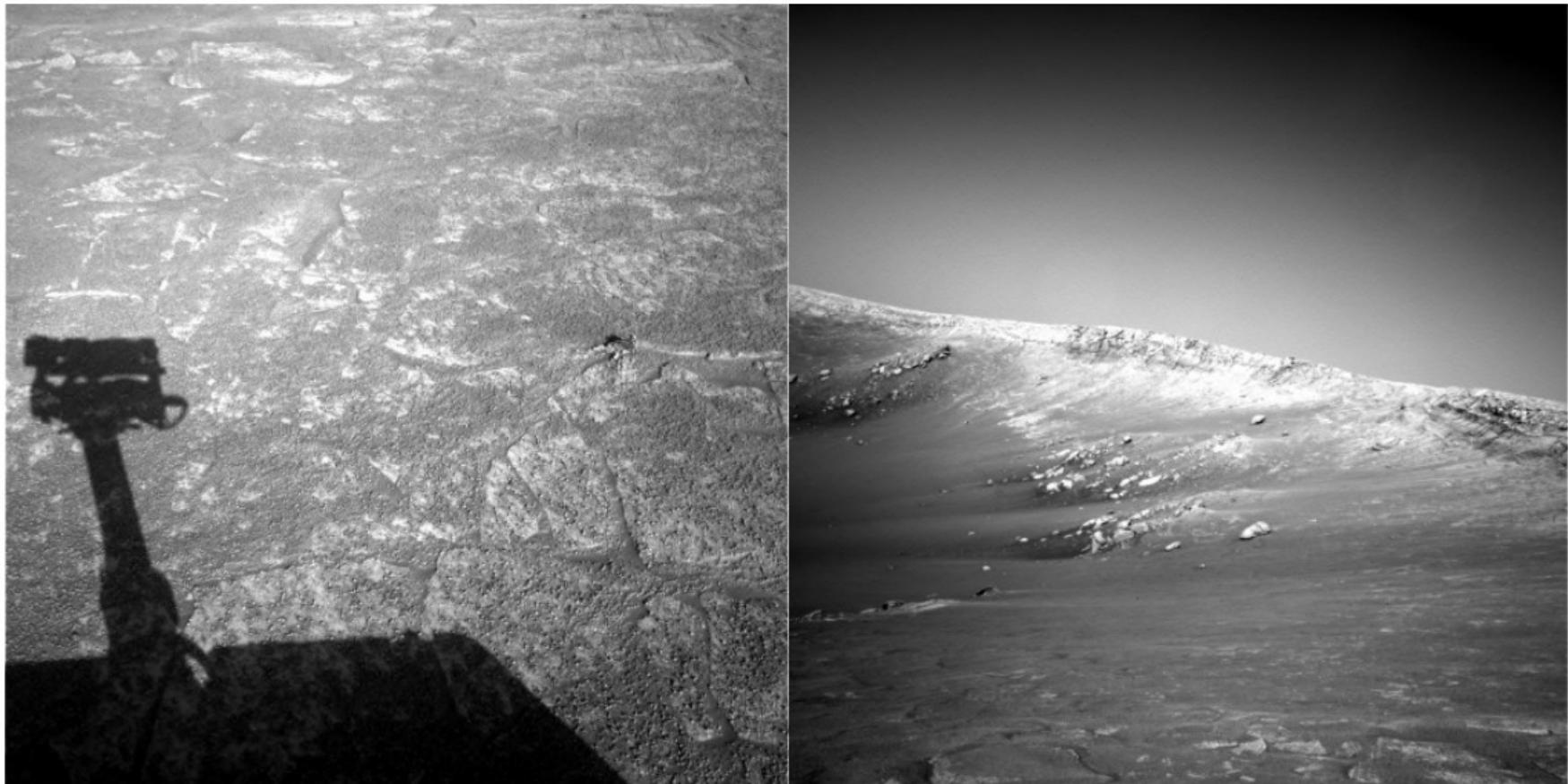
# Beyond Stereo Vision

UNIVERSITÀ  
DI PARMA



# Beyond Stereo Vision

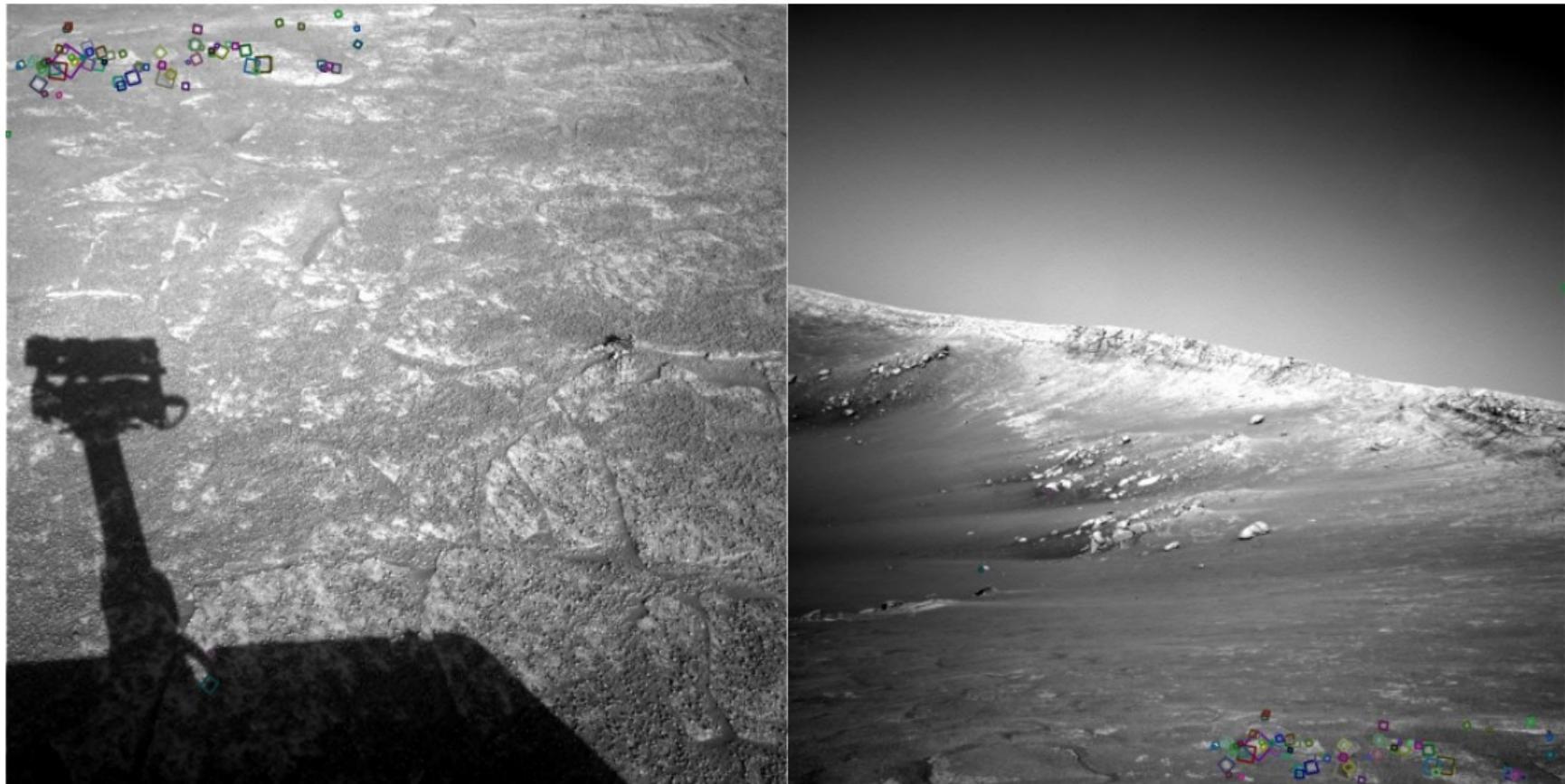
UNIVERSITÀ  
DI PARMA



NASA Mars Rover images

# Beyond Stereo Vision

UNIVERSITÀ  
DI PARMA

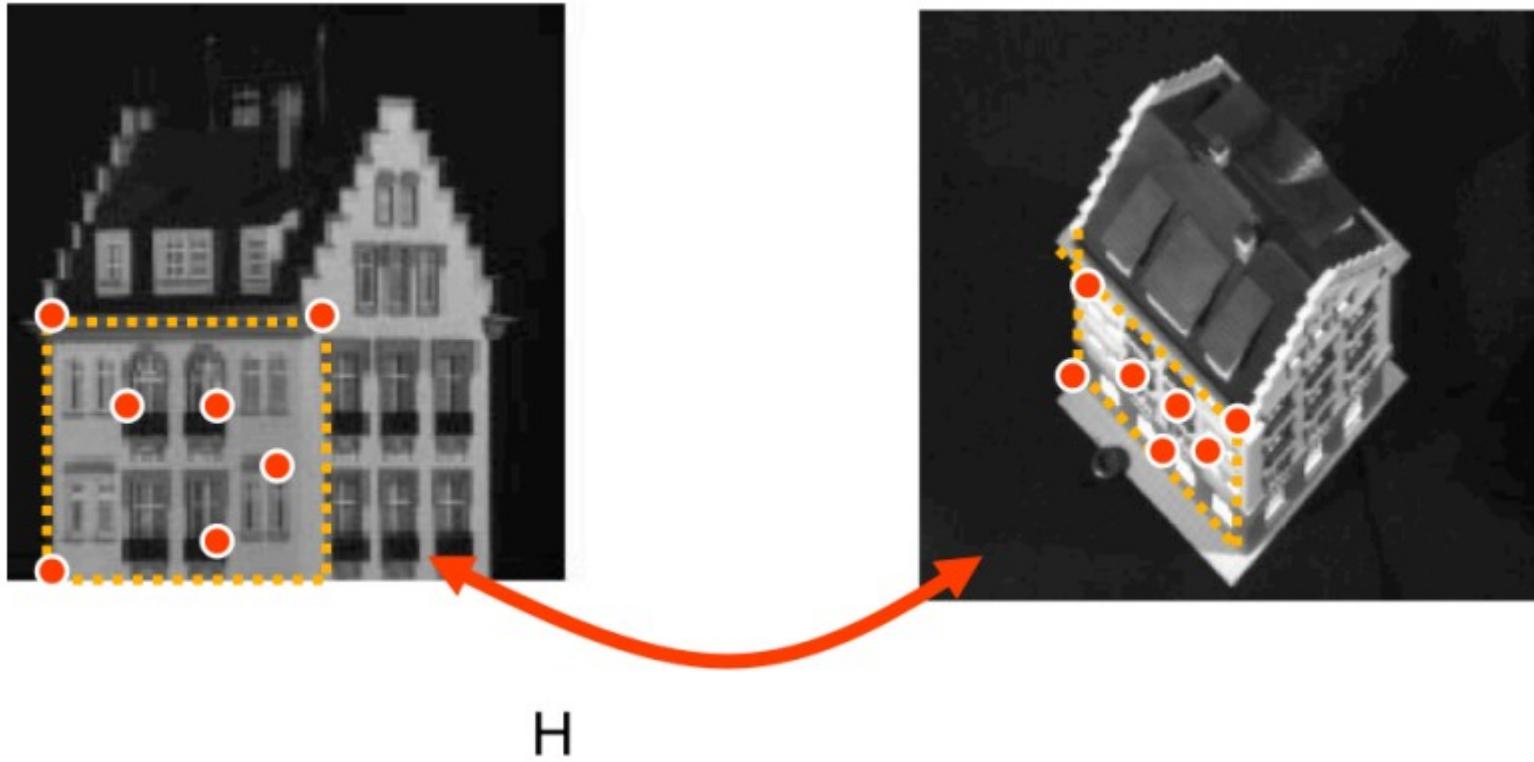


NASA Mars Rover images with SIFT feature matches

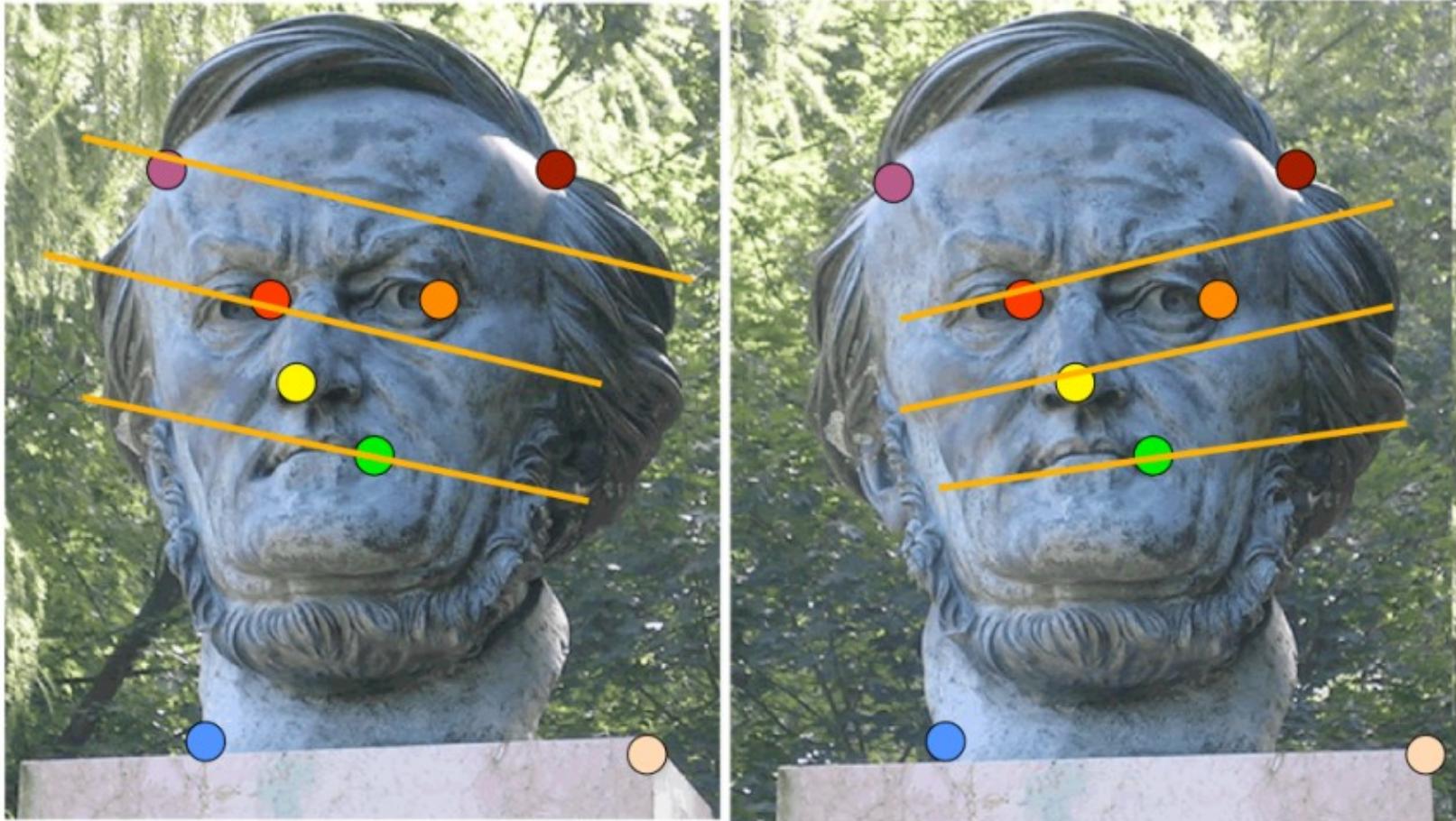
- During this lesson and the following one we are going to learn:
  - How to extract “features” from images
  - How to match them among all images

- Potential outcomes are:
  - Fit or estimate models that describe an image or a portion of it
  - Match or index images
  - Detect objects or actions from images
  - “Combine” different images

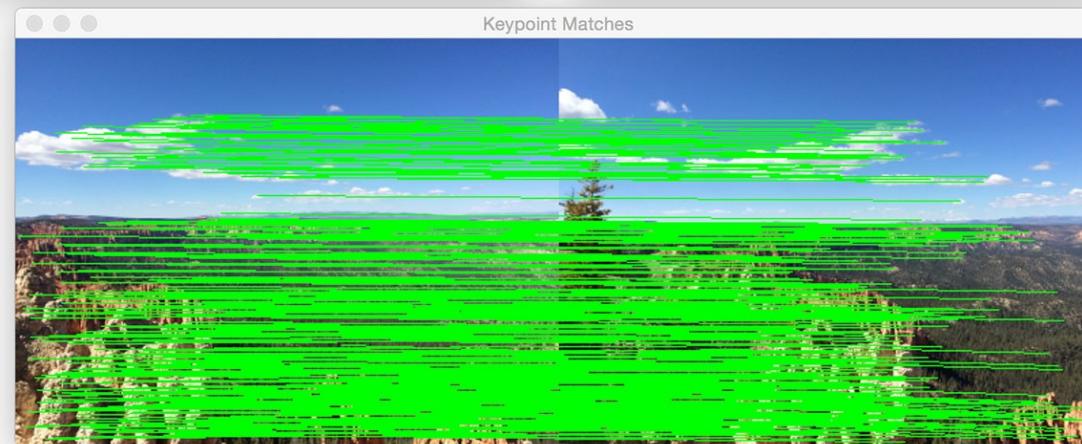
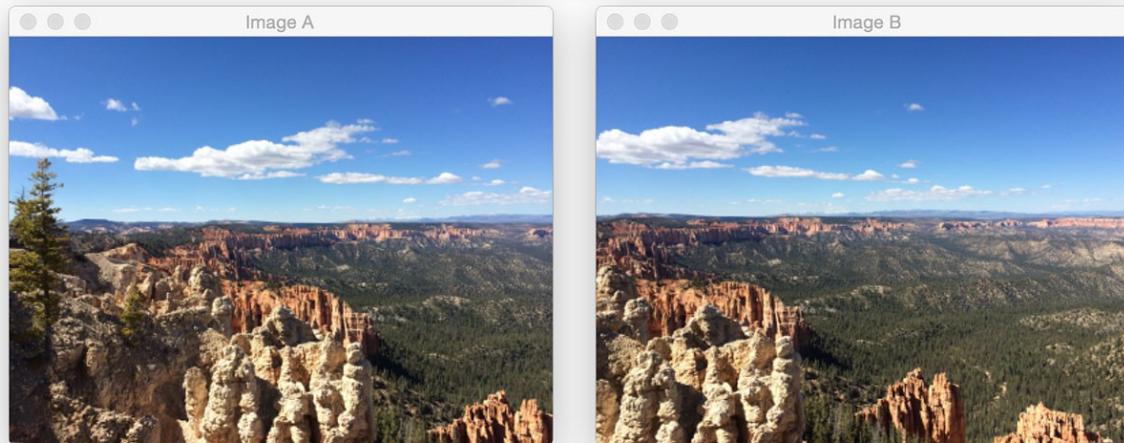
# Applications



# Applications

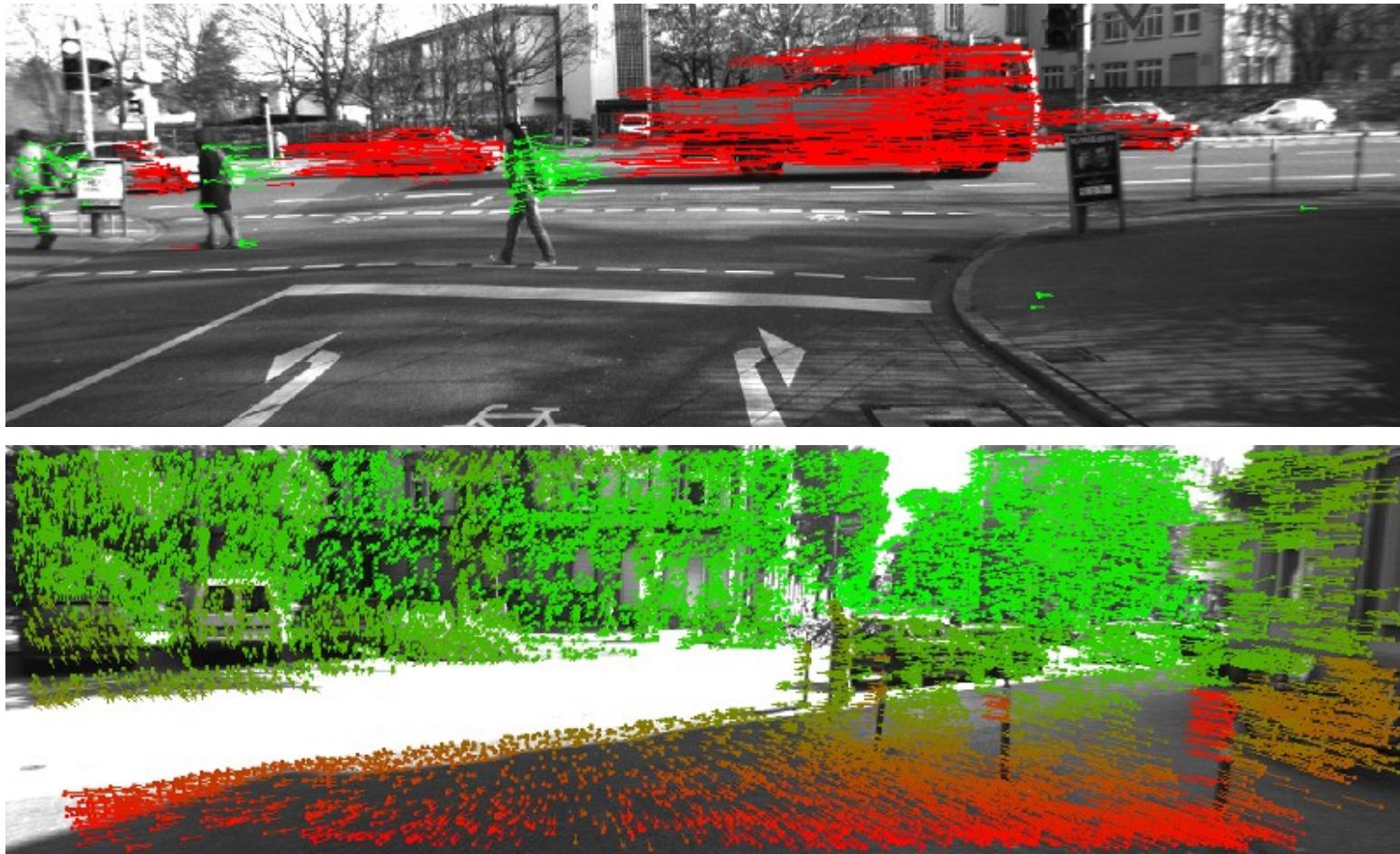


# Applications



# Applications

UNIVERSITÀ  
DI PARMA



# Applications

UNIVERSITÀ  
DI PARMA



# Applications

UNIVERSITÀ  
DI PARMA



# How to do this?

- Theoretically we could use stereo vision techniques
  - Too much expensive!
  - Moreover, we do not have to obtain a full 3D reconstruction
- Just look at the following case
  - Joining of different images...

# Example



- How to combine following images?



# Example



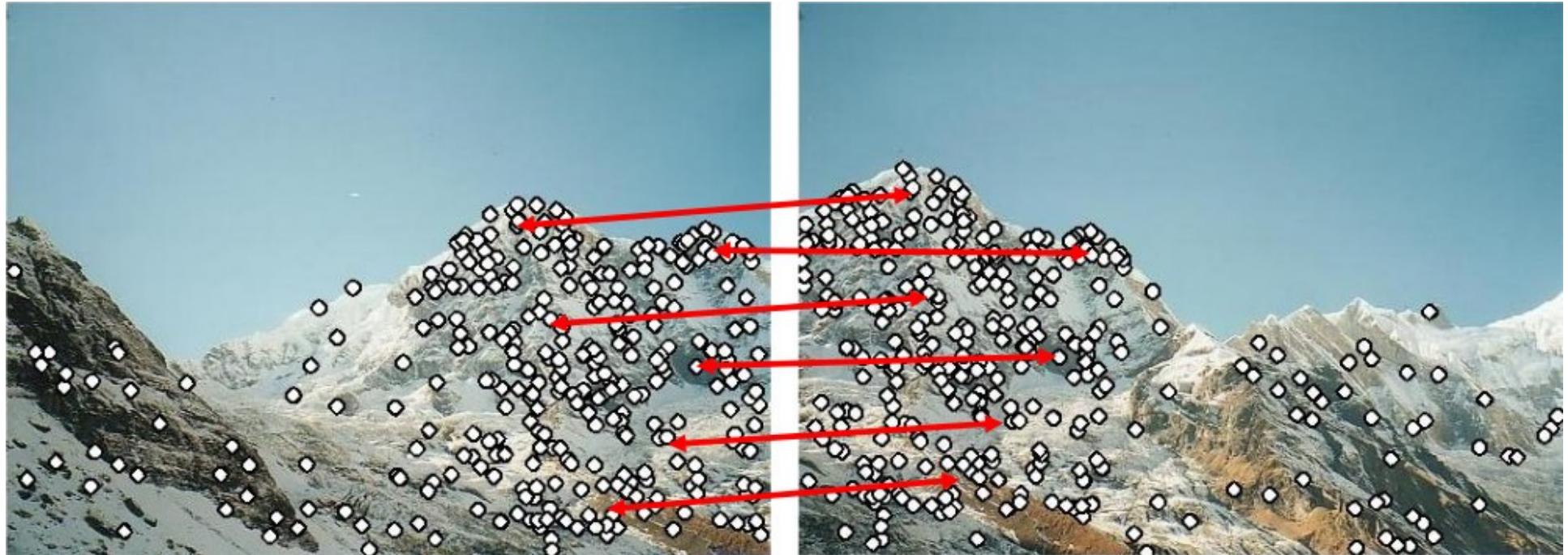
- 1. select “specific” points



# Example



- 2. find potential correspondances



# Example



- 3. compute alignment



# Example recap

- Select specific points → keypoints or features → **features extraction**
- Find potential correspondences → **features matching**
- Application dependent step
  - Matching
  - Indexing
  - Detection
  - Image alignment

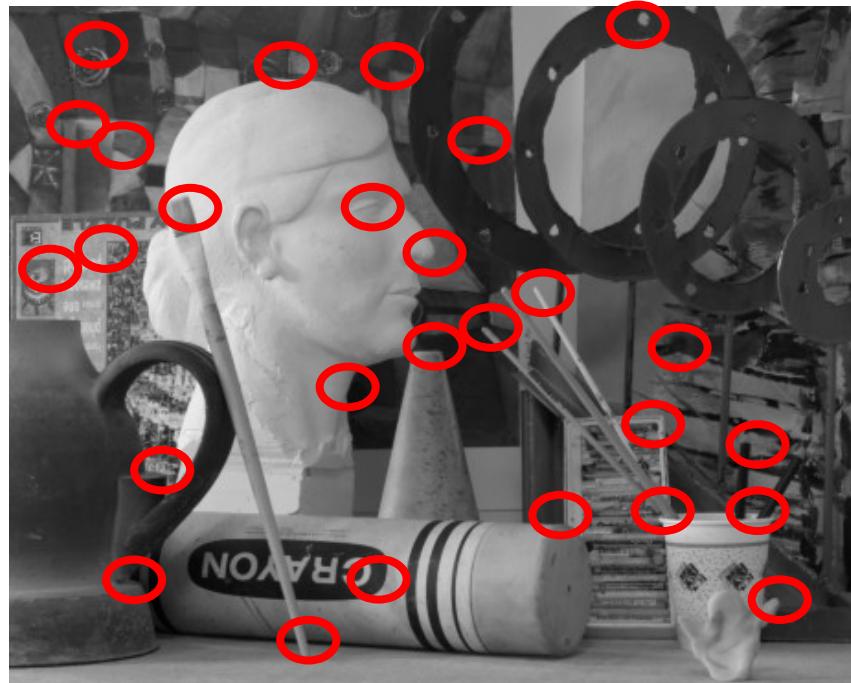
# Keypoints or Features

- For Stereo Vision
  - Window match → SAD
- In a more general situation
  - Different size
  - Different orientation
- A complete match is impracticable
  - Only keypoints → patches

# Keypoints or Features



- What keypoints I have to select?



# Characteristics of a good feature



- **Repeatability:** to be able to find the same features on even very different images
- **Saliency:** the feature contains information
- **Locality:** relatively small area of the image, namely more robust wrt occlusions and clutter

# Repeatability



Illumination  
invariance



Scale  
invariance



Pose invariance  
•Rotation  
•Affine

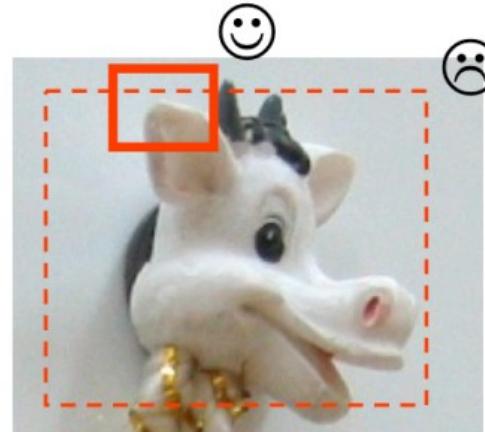
# Saliency & Locality



- Saliency

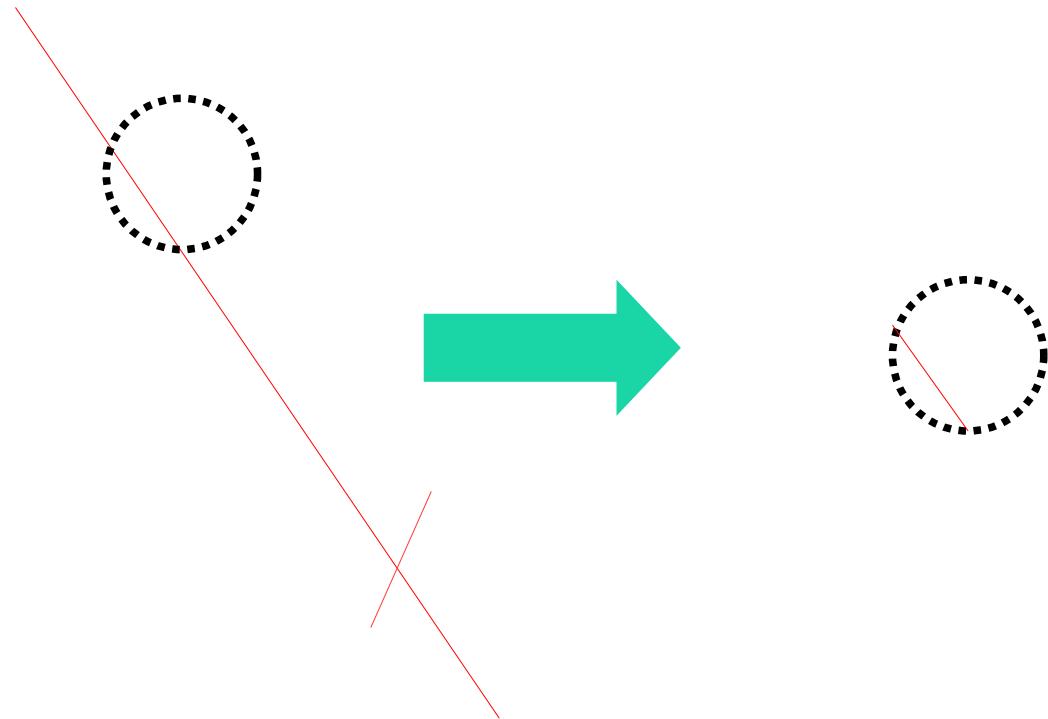


- Locality

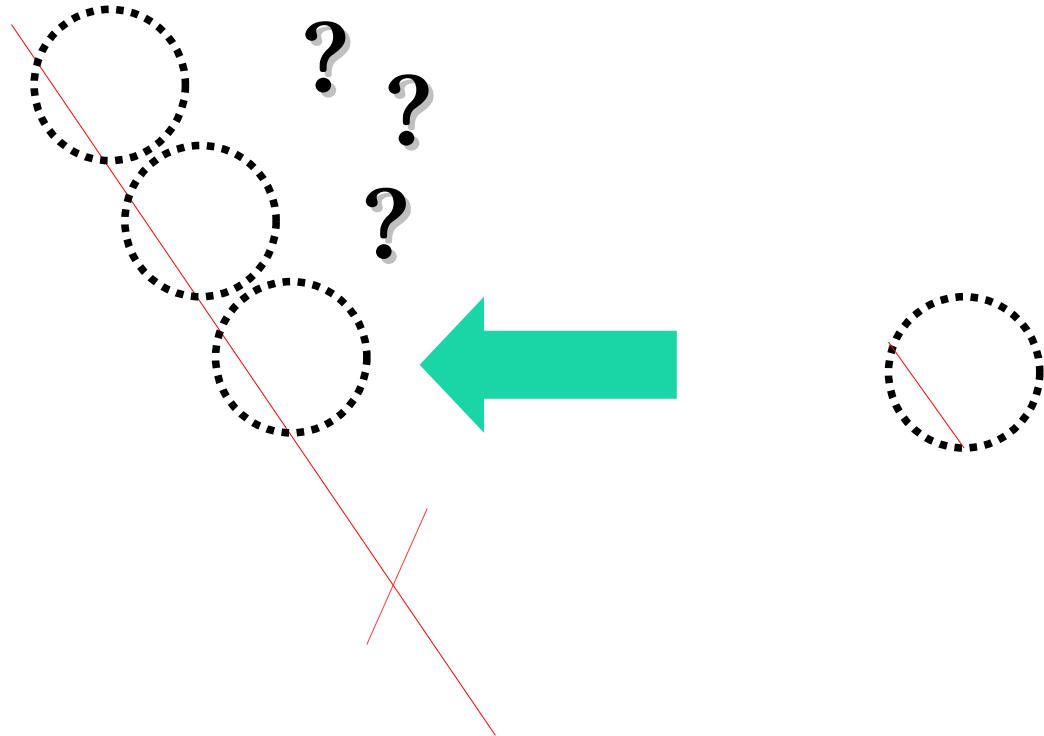


- Textureless patches are nearly impossible to match
- Patches with large contrast changes (gradients) are easier to localize
  - We want edges!
- Are straight edges a good option?
  - Aperture Problem

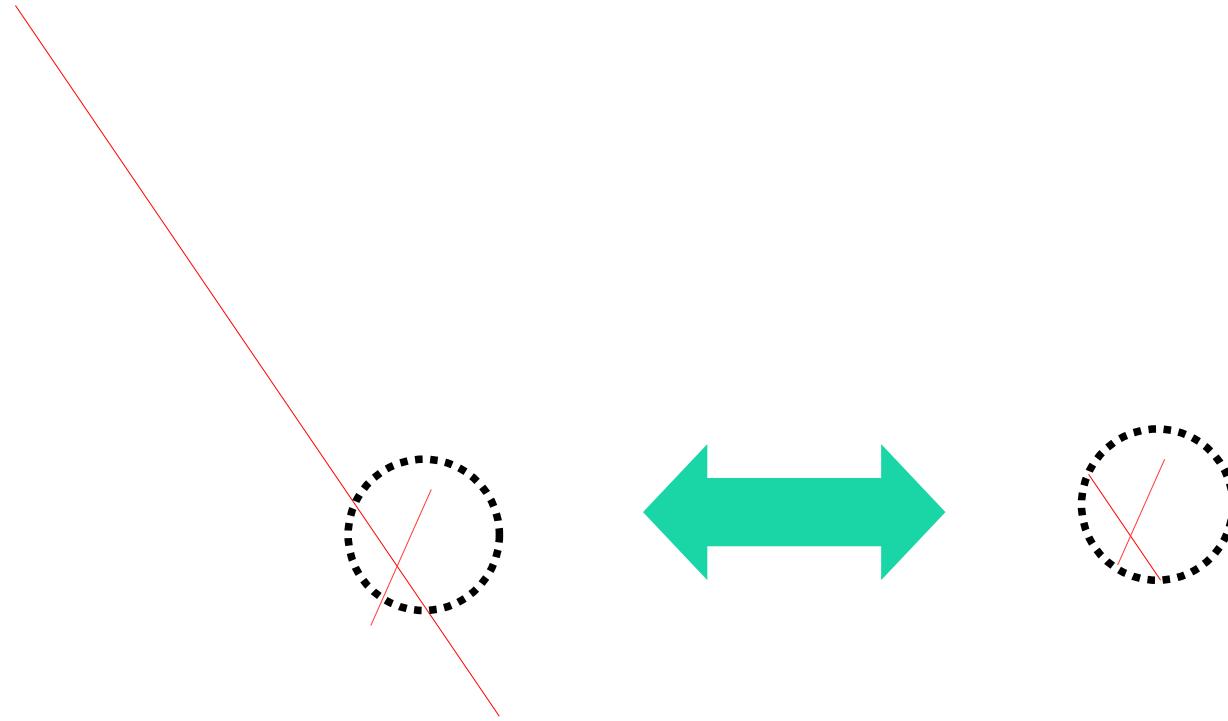
# Aperture problem



# Aperture problem



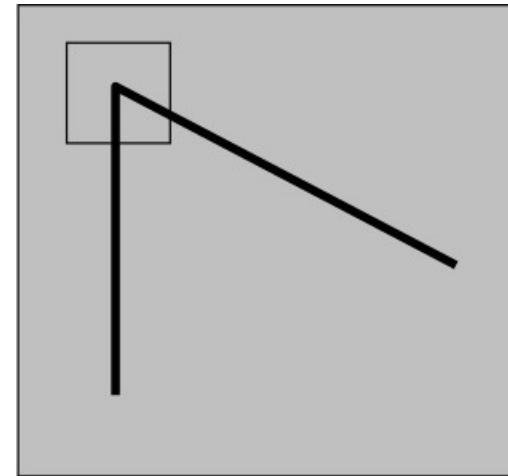
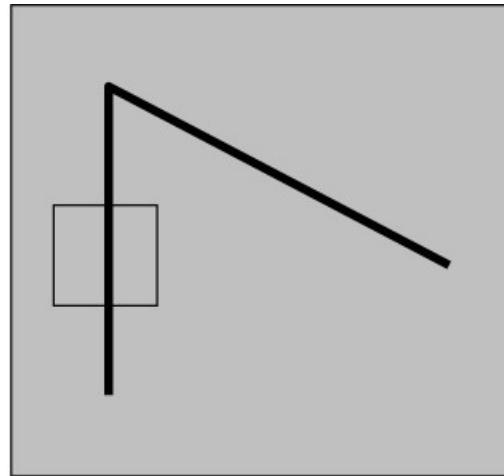
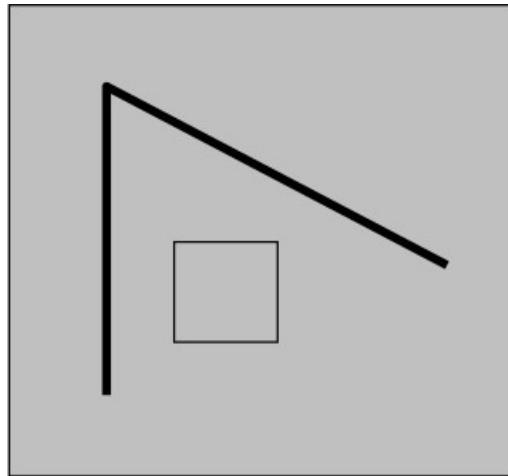
# Aperture problem



# Local Measure of Uniqueness



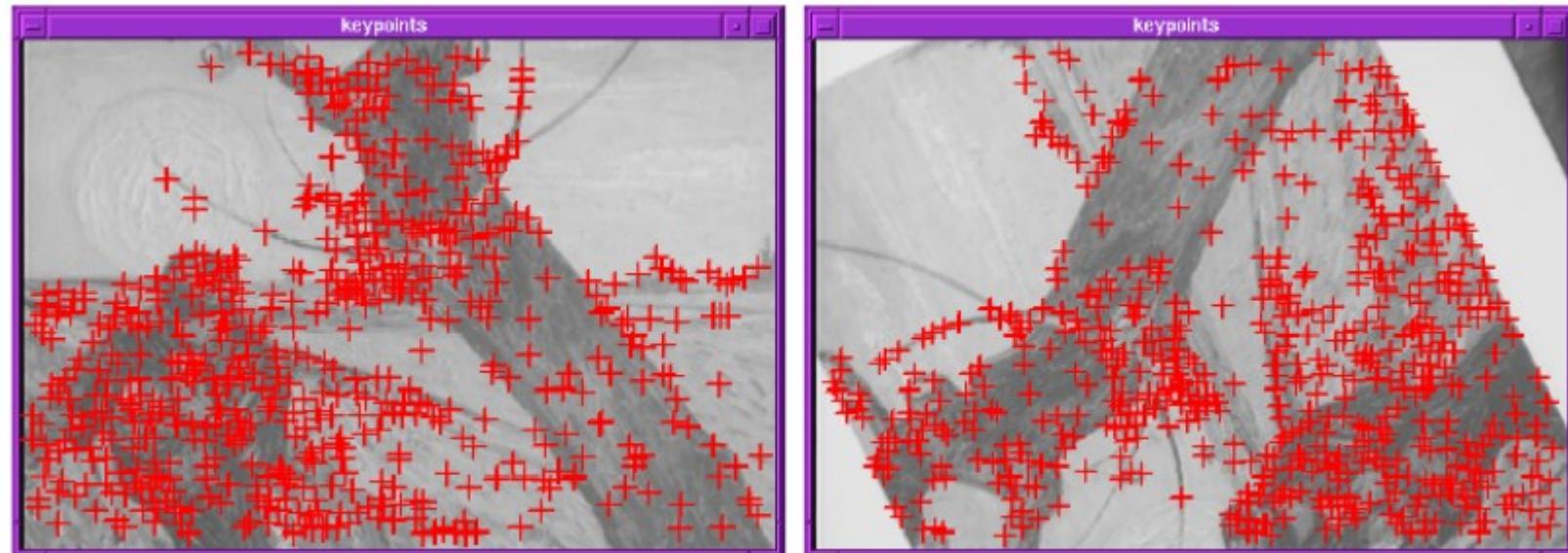
- What defines whether our patch is a good or bad candidate?



# Corner Detection



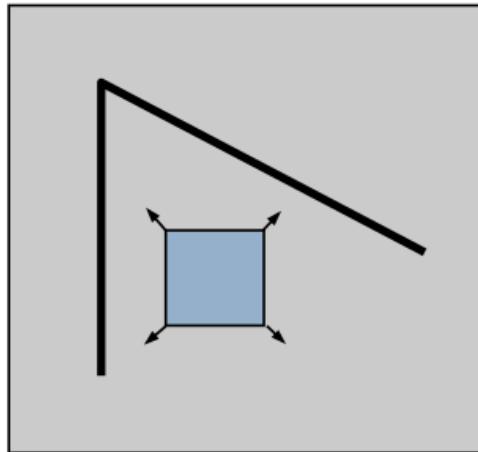
- In “real” images a corner is easier to match
  - 2 “strong” gradients



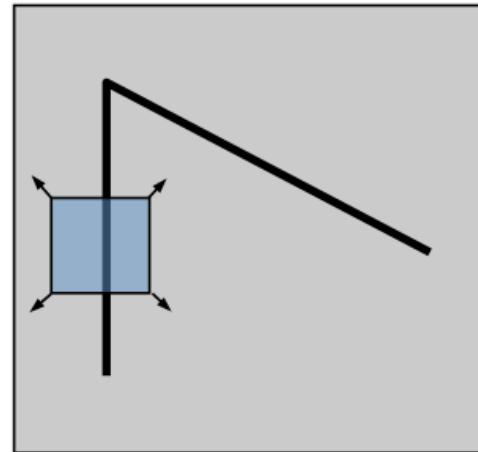
**Repeatability – Saliency – Locality**

# Corner Detection

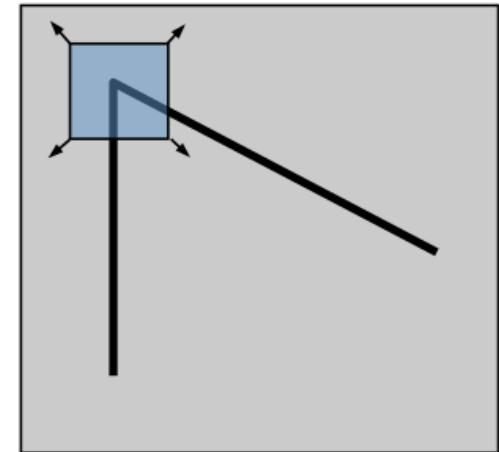
- Let's move a small window ( $W$ ) on the image
- How the content is changing?



**"flat"** region:  
no change in all  
directions



**"edge"**:  
no change along the  
edge direction

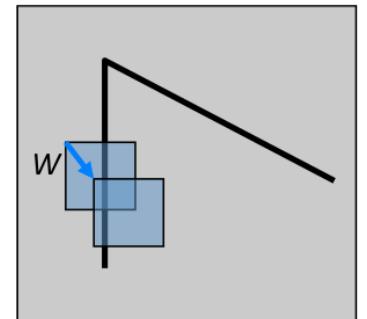


**"corner"**:  
significant change in  
all directions

# Corner Detection

- Let's move window W by (u,v)
  - u and v are small...
- A good estimation can be the Sum of Squared Differences (SSD)
- The SSD error  $E(u, v)$  can be defined as

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



- Using a Taylor Expansion

$$I_x = \frac{\partial I(x, y)}{\partial x} \quad I_y = \frac{\partial I(x, y)}{\partial y}$$

$$I(x+u, y+v) \approx I(x, y) + I_x u + I_y v + \text{higher order terms}$$

$$\approx I(x, y) + I_x u + I_y v$$

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

- Using a Taylor Expansion

$$I_x = \frac{\partial I(x, y)}{\partial x} \quad I_y = \frac{\partial I(x, y)}{\partial y}$$

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

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

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

# Corner Detector

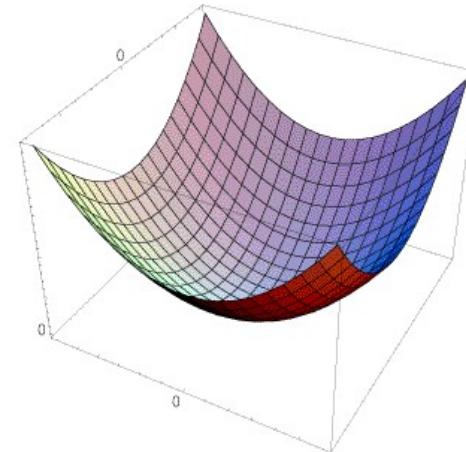


- Using a Taylor Expansion

$$I_x = \frac{\partial I(x, y)}{\partial x} \quad I_y = \frac{\partial I(x, y)}{\partial y}$$

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

$$\approx A u^2 + 2 B u v + C v^2$$



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

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

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

# Corner Detector

- Using a Taylor Expansion

$$I_x = \frac{\partial I(x, y)}{\partial x} \quad I_y = \frac{\partial I(x, y)}{\partial y}$$

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

$$\approx A u^2 + 2 B u v + C v^2$$

$$\approx [u \quad v] \begin{bmatrix} A & B \\ B & C \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = [u \quad v] H \begin{bmatrix} u \\ v \end{bmatrix}$$

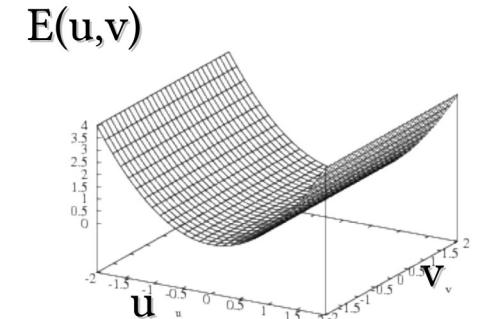
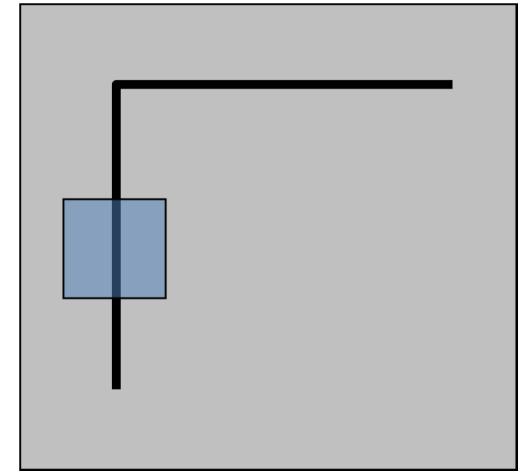
# Corner Detector



- Vertical Edge  $\rightarrow I_y=0$

$$E(u, v) \approx [u \quad v] \begin{bmatrix} A & B \\ B & C \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

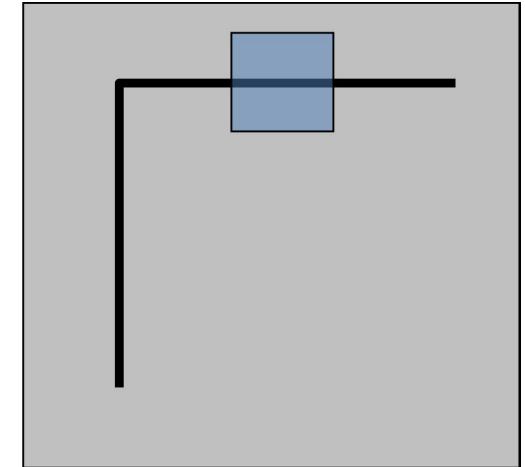
$$\approx [u \quad v] \begin{bmatrix} A & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$



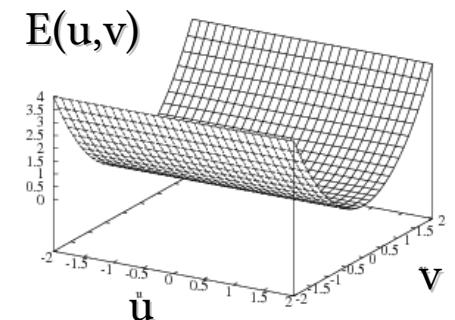
# Corner Detector

- Horizontal Edge  $\rightarrow I_x=0$

$$E(u, v) \approx [u \quad v] \begin{bmatrix} A & B \\ B & C \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$



$$\approx [u \quad v] \begin{bmatrix} 0 & 0 \\ 0 & C \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

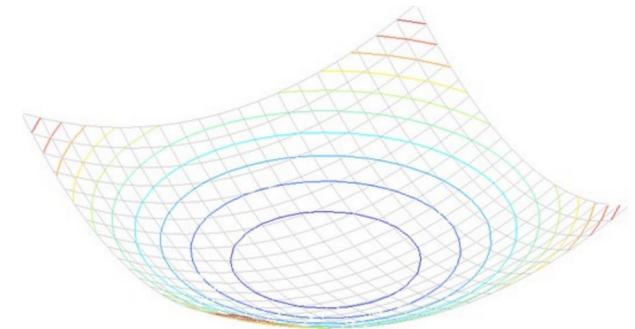




- General Case

$$E(u, v) \approx [u \quad v] \begin{bmatrix} A & B \\ B & C \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = [u \quad v] H \begin{bmatrix} u \\ v \end{bmatrix}$$

- $H$  can be considered as an ellipse with
  - Axis length →  $H$  eigenvalues
  - Axis orientation →  $H$  eigenvectors



# Corner Detector



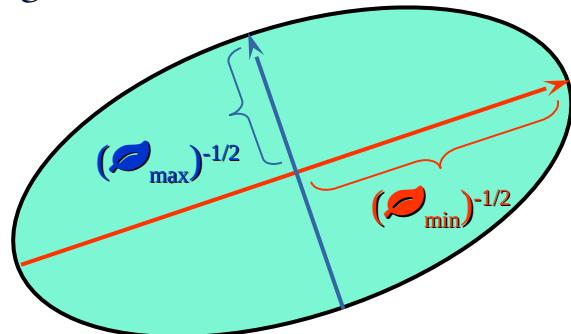
- $H$  can be visualized as an ellipse with
  - Axis length  $\rightarrow$   $H$  eigenvalues
  - Axis orientation  $\rightarrow$   $H$  eigenvectors

$$H = R \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R^{-1}$$

direction of the fastest  
change

leaf<sub>max</sub>, leaf<sub>min</sub> : eigenvalues of  $H$

direction of the slowest change



# Eigenvector/Eigenvalues review

- Given a matrix  $A$  eigenvectors  $x$  and eigenvalues  $\lambda$  satisfy:

$$Ax = \lambda x$$

- Eigenvalues can be found solving

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

- For a  $2 \times 2$  matrix like  $H$  we can find  $\lambda \rightarrow$

- This leads to a **quadratic** equation

- i.e.  $\lambda_1$  and  $\lambda_2$

- Once found  $\lambda$ s,  $x$  can be computed as  $\rightarrow$

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

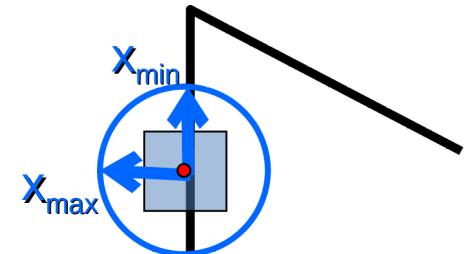
# Corner Detector



$$E(u, v) \approx [u \quad v] \begin{bmatrix} A & B \\ B & C \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = [u \quad v] H \begin{bmatrix} u \\ v \end{bmatrix}$$

$$\begin{aligned} Hx_{max} &= \lambda_{max} x_{max} \\ Hx_{min} &= \lambda_{min} x_{min} \end{aligned}$$

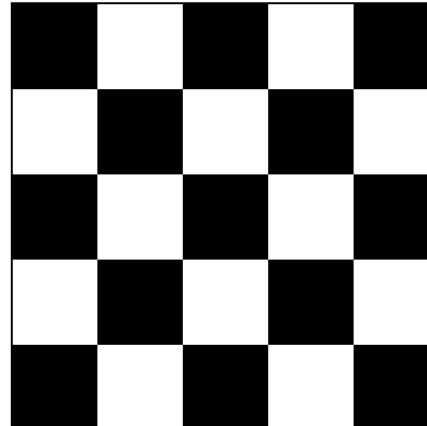
- $x_{max} \rightarrow$  direction of largest increase for  $E(u, v)$
- $x_{min} \rightarrow$  direction of smallest increase for  $E(u, v)$
- $\lambda_{max} \rightarrow$  amount of increase along  $x_{max}$
- $\lambda_{min} \rightarrow$  amount of increase along  $x_{min}$



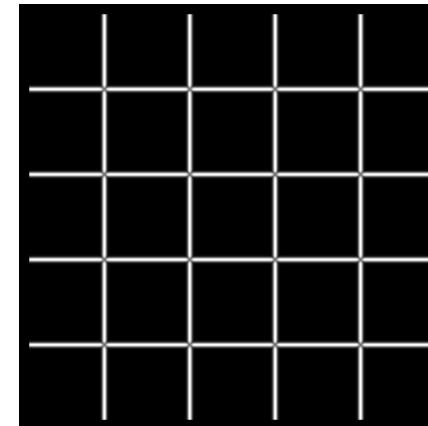
# Corner Detector



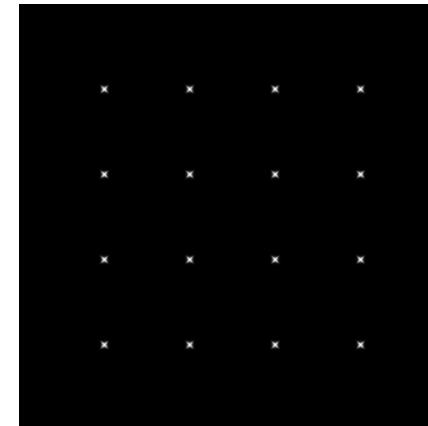
- How  $\lambda_{\min}$  and  $\lambda_{\max}$  behave on corners?
- $\lambda_{\max}$  is “big” when we have at least one “strong” gradient
- $\lambda_{\min}$  encodes the strength of the orthogonal gradient



$I$



$\lambda_{\max}$



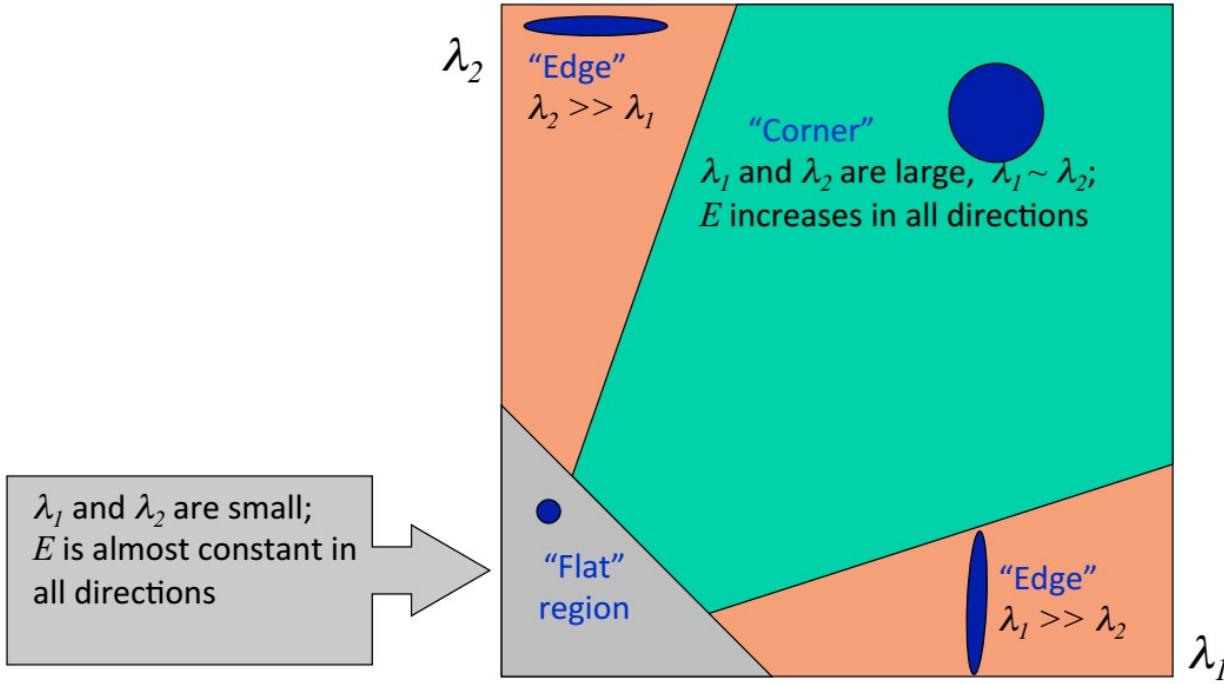
$\lambda_{\min}$

# Corner Detector



- In a corner both  $\lambda_1$  and  $\lambda_2$  are “large”

Slide credit: Kristen Grauman

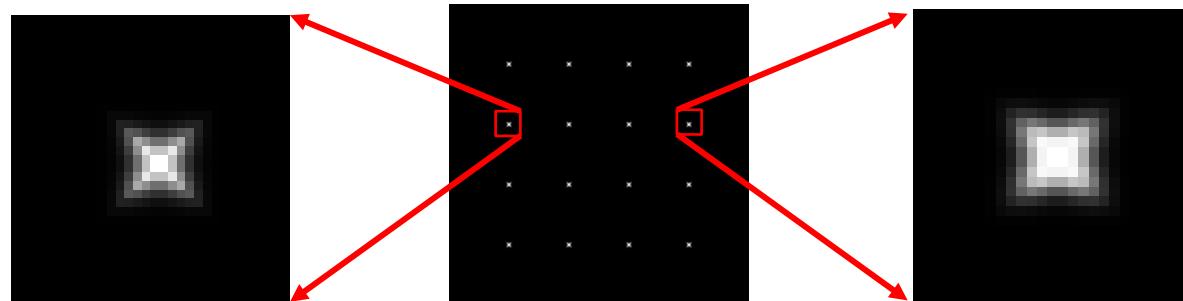


# Harris Corner Detector



- “Harris” is a little variant of previous detector
  - No need to compute  $\lambda$ s  $\rightarrow$  no square roots
  - Weight function  $\rightarrow$  based on the distance from the center pixel

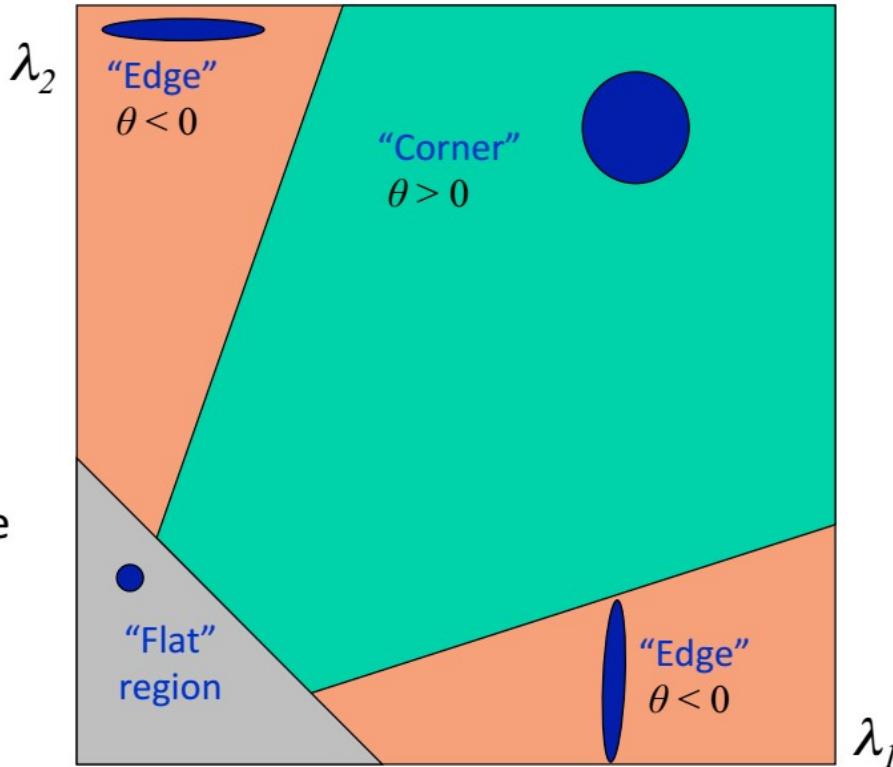
$$\theta = \det(H) - \alpha \operatorname{trace}(H)^2$$



# Harris Corner Detector



$$\theta = \det(M) - \alpha \operatorname{trace}(M)^2 = \lambda_1 \lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$$



- Fast approximation
  - Avoid computing the eigenvalues
  - $\alpha$ : constant (0.04 to 0.06)

Slide credit: Kristen Grauman

# Corner Detectors

- Shi et al. (1994) →  $\lambda_{min}^{\frac{1}{2}}$
- Harris et al. (1998) →  $\theta = \det(H) - \alpha \text{trace}(H)^2 = \lambda_0 \lambda_1 - \alpha (\lambda_0 + \lambda_1)^2$
- Triggs et al. (2004) →  $\lambda_{min} - \alpha \lambda_{max}$
- Brown et al. (2005) →  $\lambda_{min} \approx \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2} = \frac{\det(H)}{\text{trace}(H)}$

# Window Function

- Harris also introduced a window function
  - The idea is to weight distance

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

Window  
Function

Shifted  
Intensity

Intensity

# Window function



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

- Option 1: uniform window

- Sum over square window

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

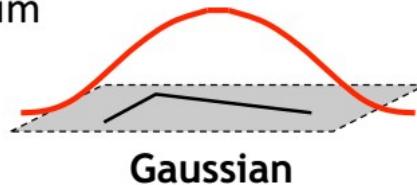


- Problem: not rotation invariant

- Option 2: Smooth with Gaussian

- Gaussian already performs weighted sum

$$M = g(\sigma) * \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$



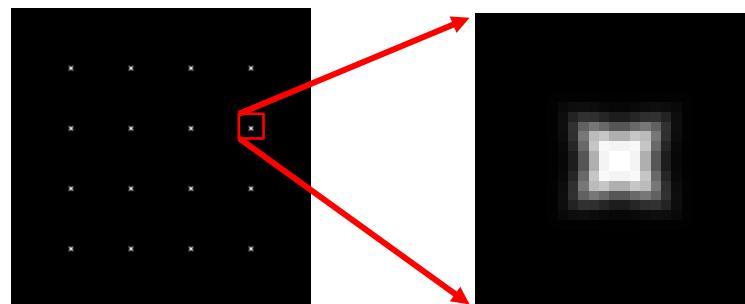
- Result is rotation invariant

# Harris Corner Detector



- Summary
  - Compute horizontal and vertical gradients
  - Apply window function
  - For each pixel
    - Create H matrix from gradients
    - Compute  $\theta$
    - Discard pixels with  $\theta < 0$  or anyway below a given threshold
  - Apply a NMS approach to find pixels where  $\lambda_{\min}$  is a local maximum

$$H(\sigma_I, \sigma_D) = g(\sigma_I) \begin{bmatrix} \sum_{(x,y) \in W} I_x^2(\sigma_D) & \sum_{(x,y) \in W} I_x I_y(\sigma_D) \\ \sum_{(x,y) \in W} I_x I_y(\sigma_D) & \sum_{(x,y) \in W} I_y^2(\sigma_D) \end{bmatrix}$$

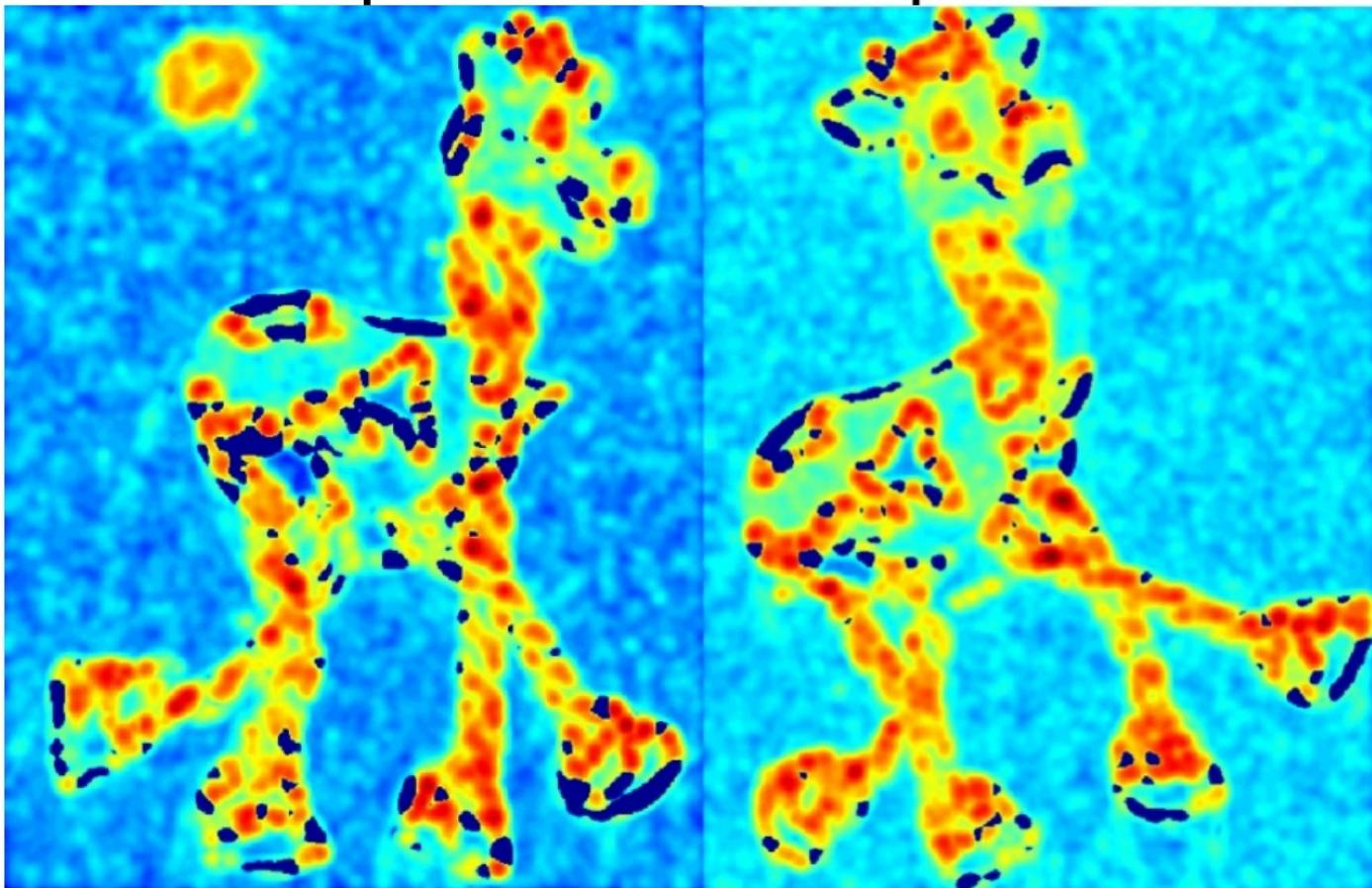


# Harris Corner Detector steps



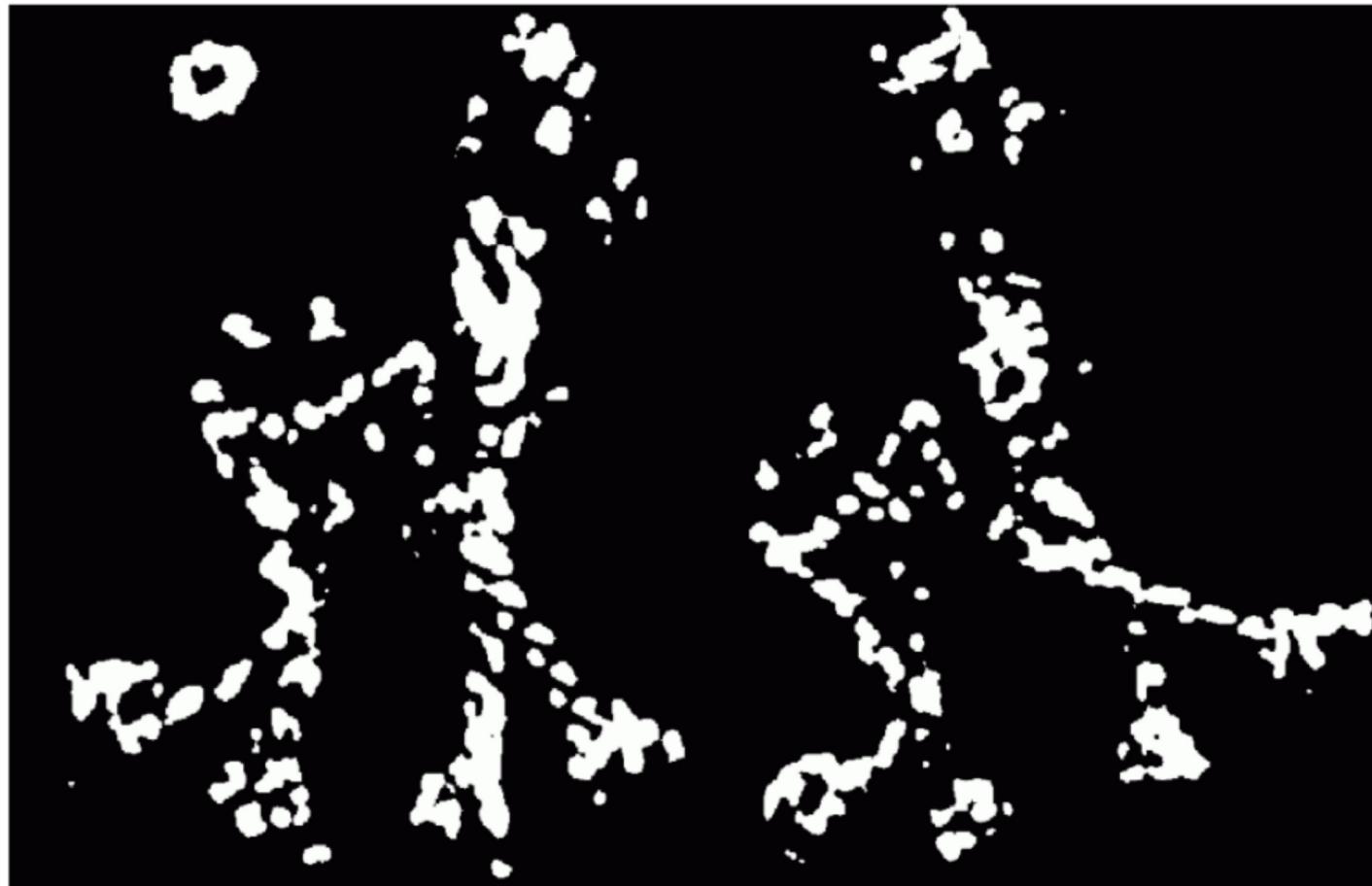
Slide adapted from Darya Frolova, Denis Simakov

# Harris Corner Detector steps



Slide adapted from Darya Frolova, Denis Simakov

# Harris Corner Detector steps



# Harris Corner Detector steps



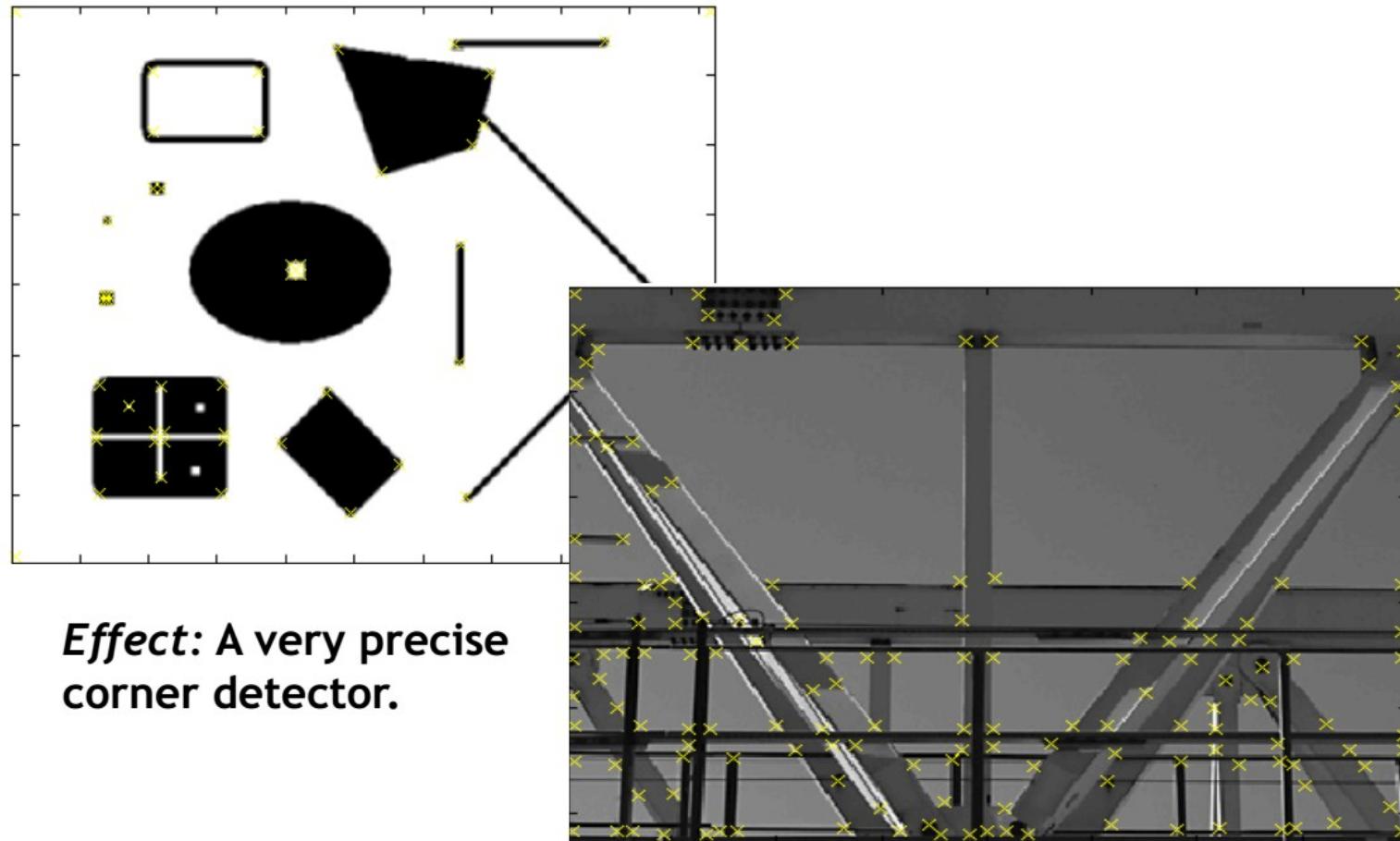
Slide adapted from Darya Frolova, Denis Simakov

# Harris Corner Detector steps



Slide adapted from Darya Frolova, Denis Simakov

# Examples



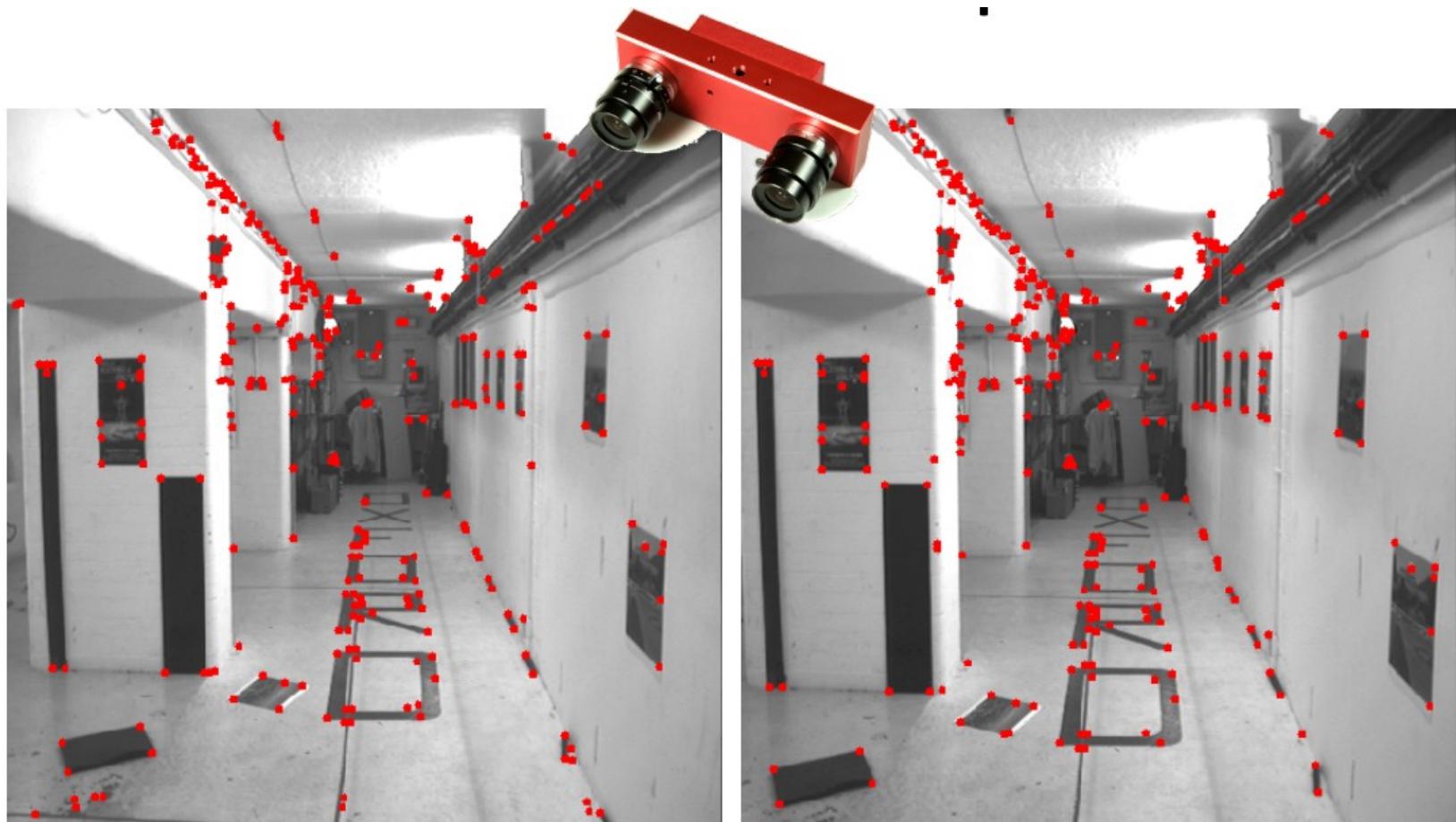
Slide credit: Krystian Mikolajczyk

# Examples



Slide credit: Krystian Mikołajczyk

# Sparse Stereo



# Invariance to transformations



- How much keypoints extraction is robust to image transformations?

- Rotation (traslation)



- Intensity change

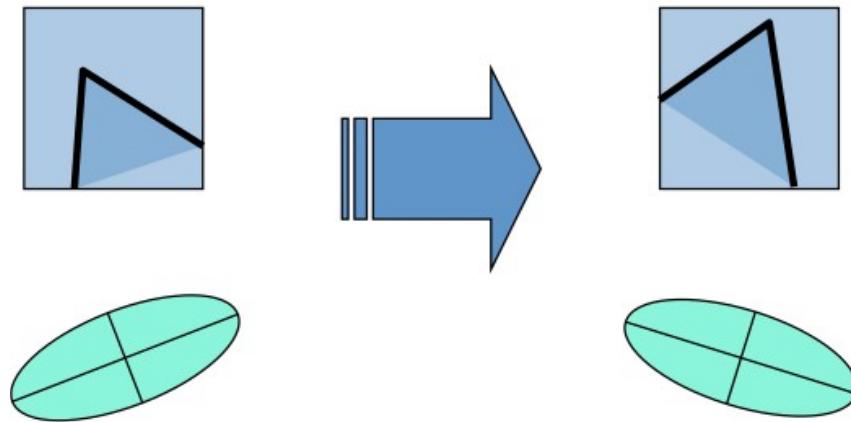


- Scale



- Ellipse rotates but eigenvalues are the same

$$H = R \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R^{-1}$$

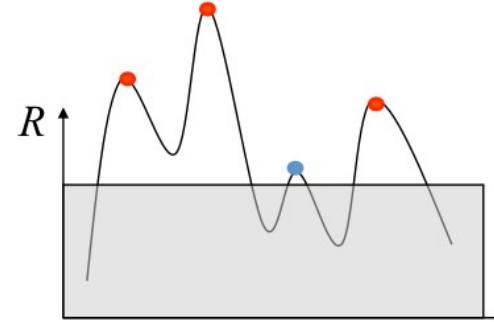
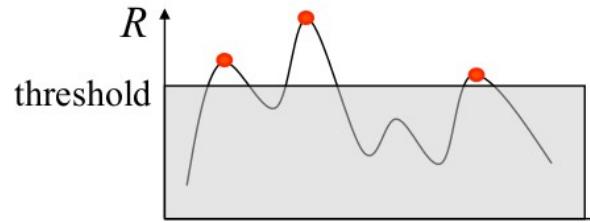


Invariant to Rotation

# Intensity Change



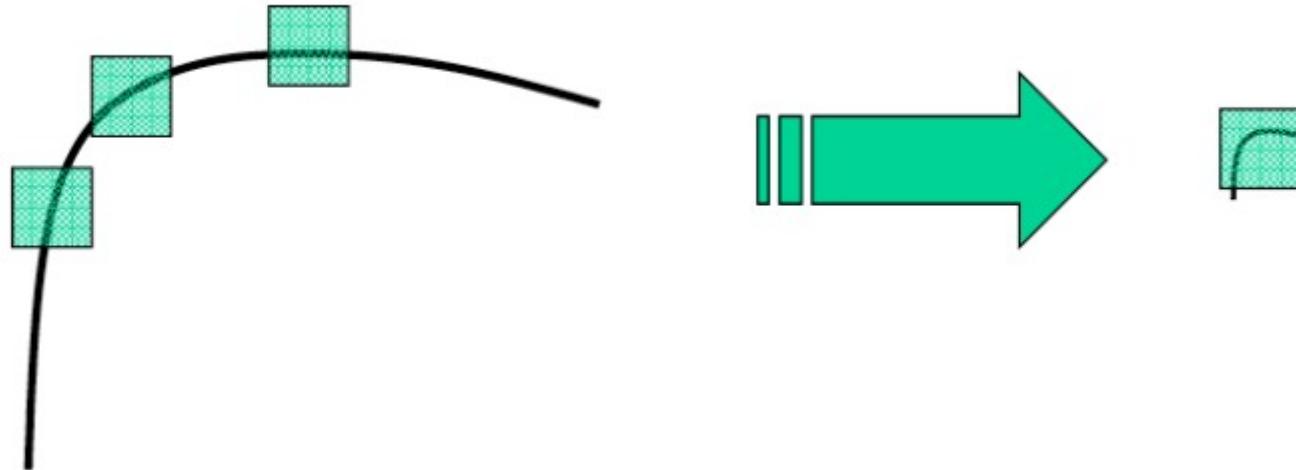
- Intensity change can be modeled as  $g(r, c) = a \cdot f(r, c) + b$
- Derivative operations make result invariant wrt  $b$
- Not so true about  $a$  ...



**Partially Invariant to Intensity Changes**



- Right → corner
- Left → corner



Not Invariant to Scale Changes

# Scale



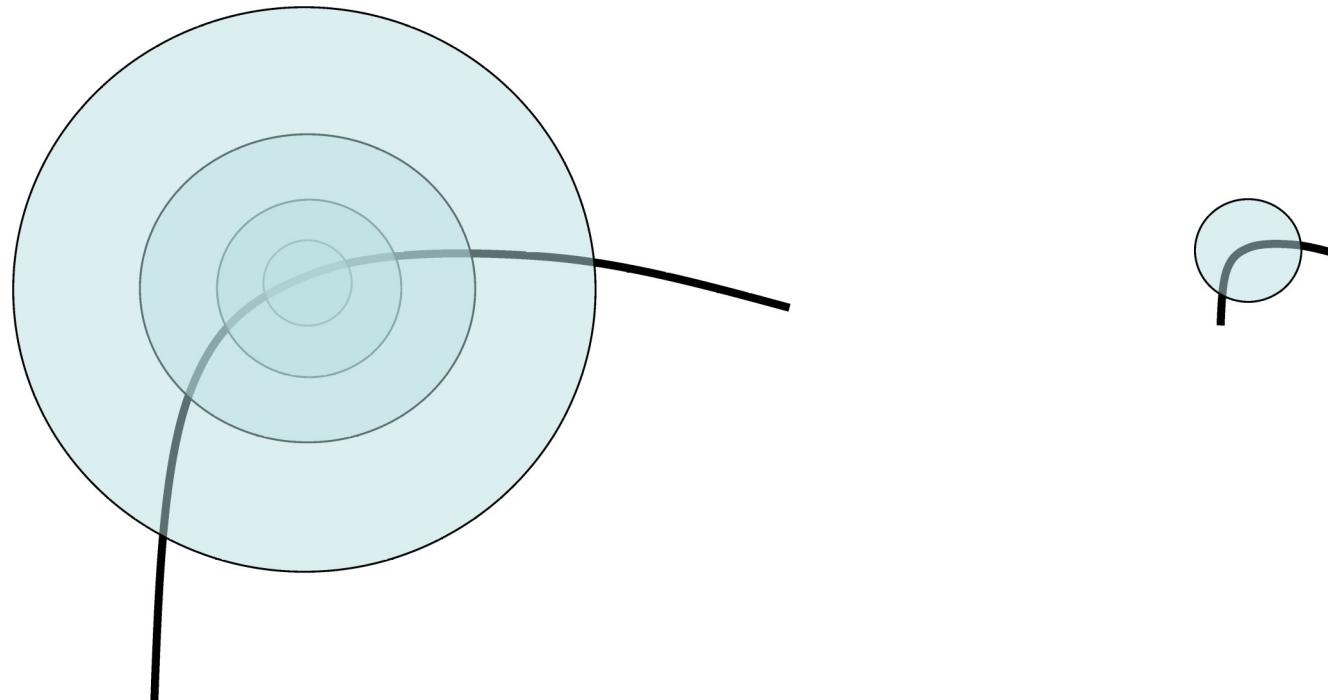
- The keypoints are highly dependent on window size



# Scale invariant detector



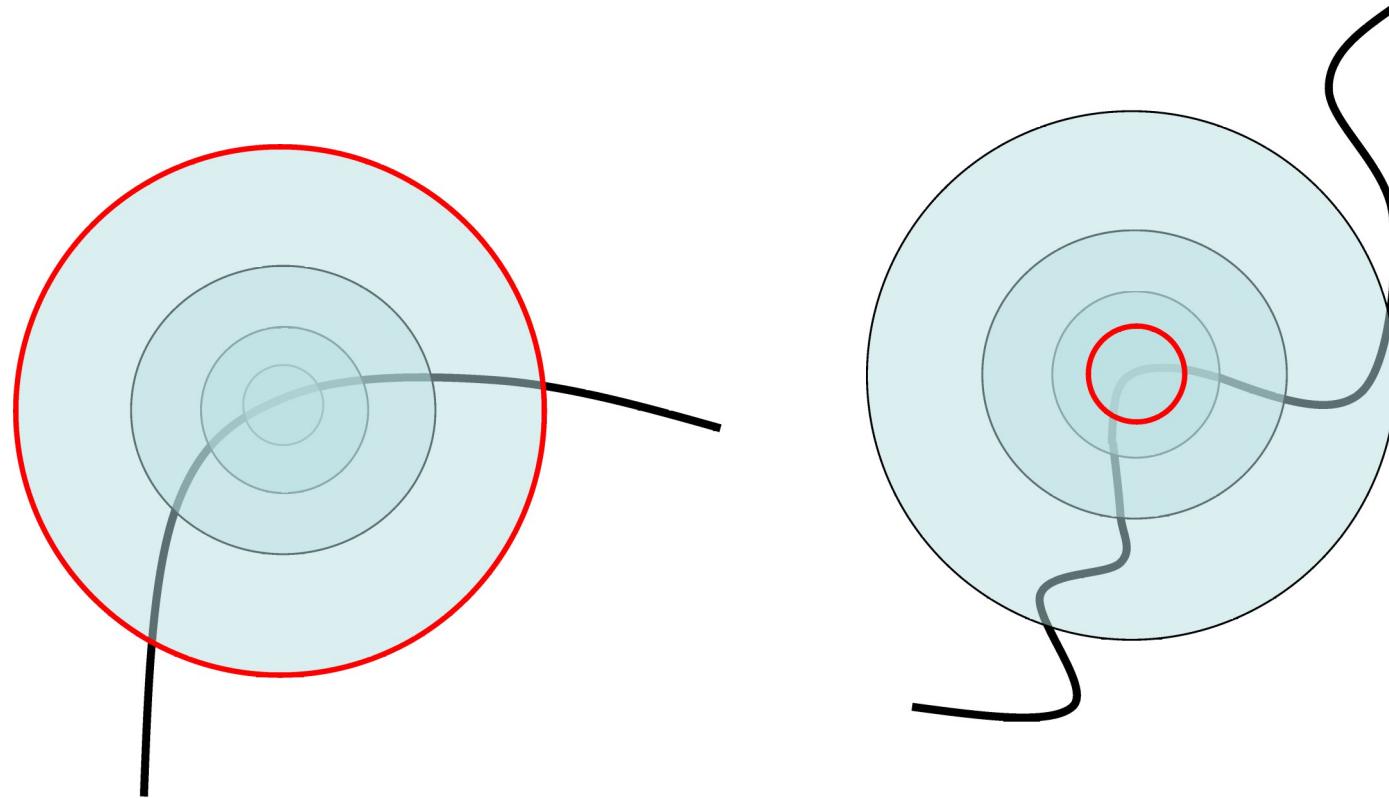
- Consider circles of different size around a point
- We can find a size that look similar in both images



# Scale invariant detector



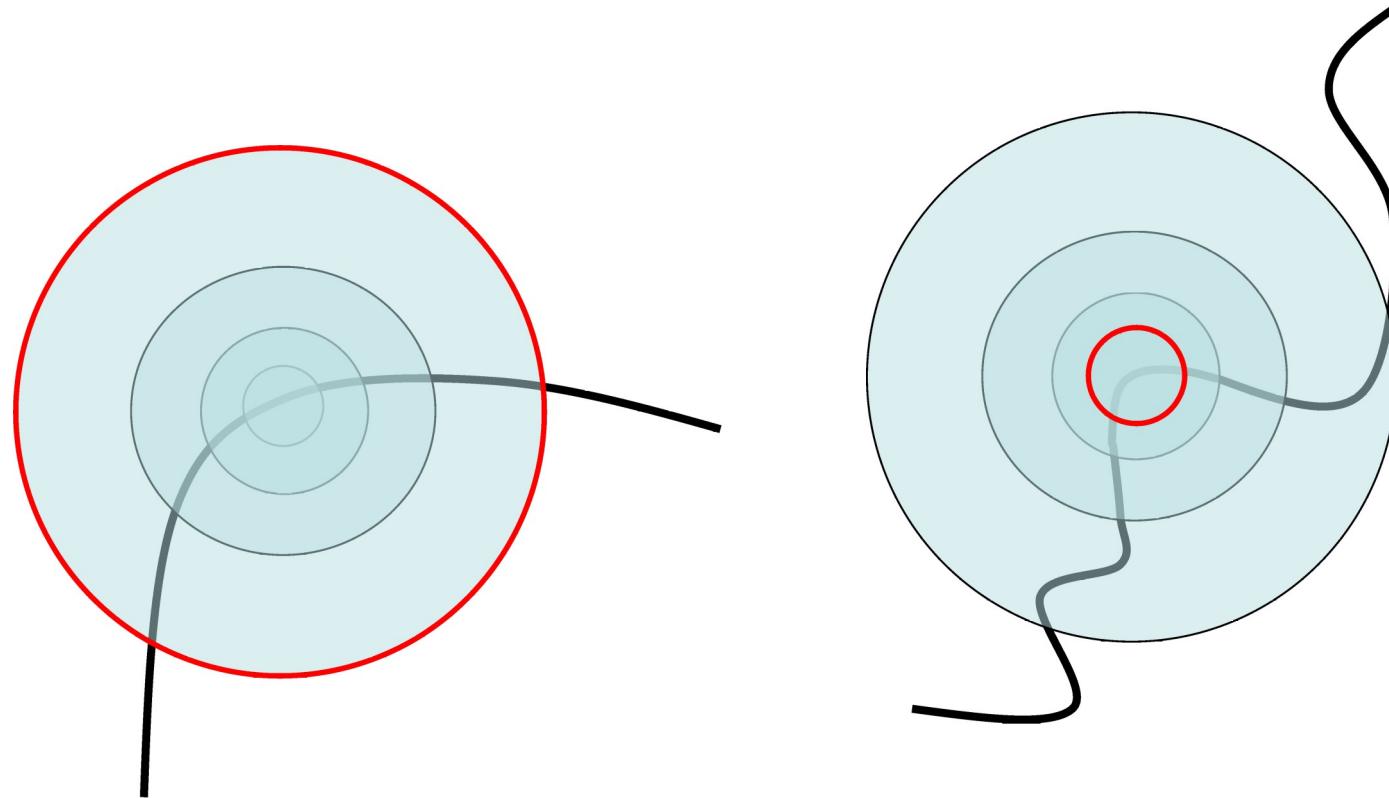
- How do we select appropriate circles size in both images?



# Scale invariant detector



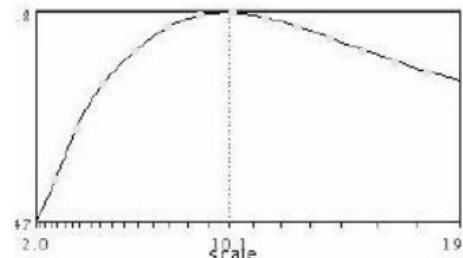
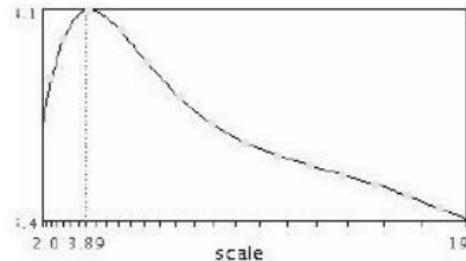
- We can select the window size that **maximize** Harris operator



# Scale invariant detector



- Example



# Scale invariant detector



- Instead computing  $\theta$  for larger and larger windows a pyramidal approach can be used



(sometimes need to create in-between levels, e.g. a  $\frac{3}{4}$ -size image)

# Other scale invariant detectors

- **HARRIS-LAPLACIAN**
  - K. Mikolajczyk, C. Schmid, Indexing based on scale invariant interest points, ICCV 2001
- **SIFT: Scale Invariant Feature Transform**
  - D. Lowe. "Distinctive Image Features from Scale-Invariant Keypoints", IJCV 2004
- **FAST: Features from Accelerated Segment Test**
  - Edward Rosten and Tom Drummond, "Machine learning for high speed corner detection" in 9th European Conference on Computer Vision, vol. 1, 2006
- **SURF: Speeded-Up Robust Features**
  - Herbert Bay, Andreas Ess, Tinne Tuytelaars, Luc Van Gool, "SURF: Speeded Up Robust Features", Computer Vision and Image Understanding (CVIU), 2008
- **MSER: Maximally Stable Extremal Regions**
  - J. Matas, O. Chum, M. Urban, and T. Pajdla. "Robust wide baseline stereo from maximally stable extremal regions." Proc. of British Machine Vision Conference, 2002.
- **BRISK: Binary Robust Invariant Scalable Keypoint**
  - Stefan Leutenegger ; Margarita Chli ; Roland Y. Siegwart, 2011 International Conference on Computer Vision



UNIVERSITÀ DI PARMA

# Features Extraction

Question time!

