Corners and Other Salient Features

More Possibilities

- 2D analysis is more than just grouping and segmentation
- For use in video – tracking
- For use in multi-view analysis – model building, panorama building
- Detect and match features across multiple frames
- What constitute a good feature to track and match?
  - Uniqueness
  - Invariance

Image features

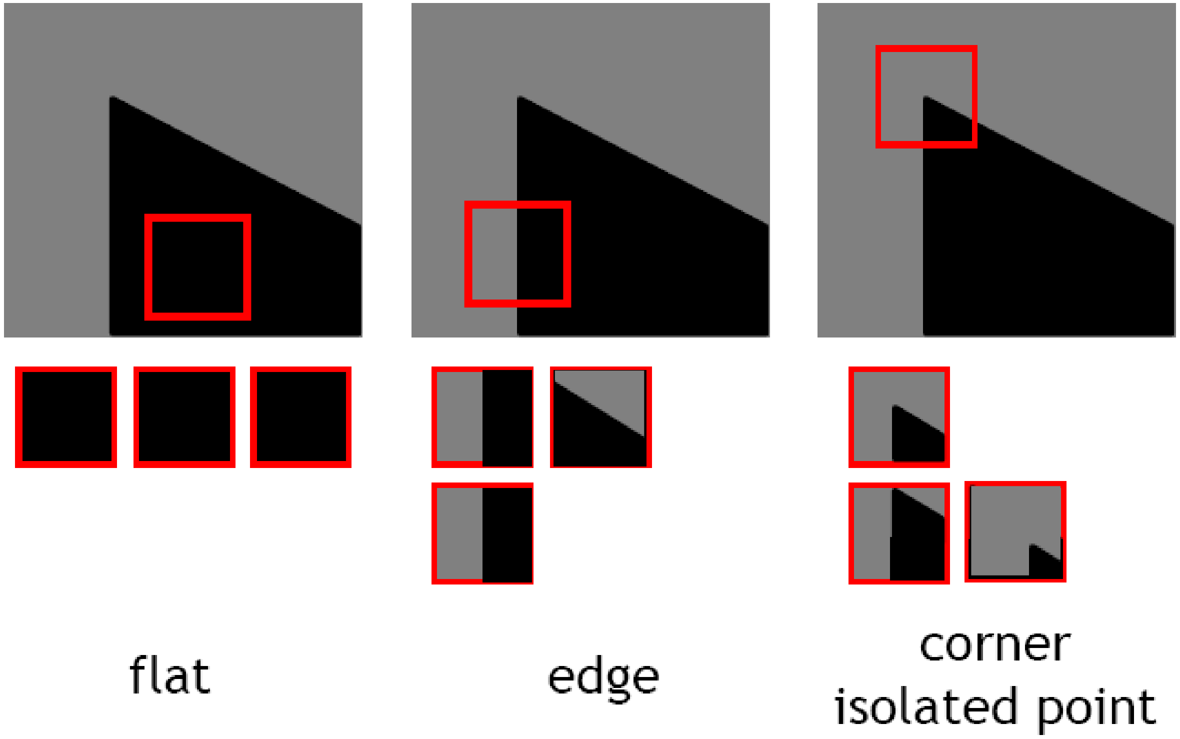
- Low-level features
  - Meaningful or “interesting” points, local features:
    - Edges, corners, salient textures
  - Desirable properties?
    - Easy to compute
    - Relatively robust
      - To noise, variations in illumination, variations in viewpoint and pose, different sensors/cameras, ...
    - High detection rate, low false positive rate
- Mid-level features
  - Lines, curves, contours, ellipses
  - Groups of features
    - Parallel lines, related corners, clusters of low-level features, ...
- High-level (Semantic) features
  - Faces, telephones, tooth brushes, etc.

Image Features

- Low-level features
  - Simple to detect & describe
  - Precise localization
  - Many, harder to match
  -
- High-level features
  - Sophisticated detection
  - Vague localization
  - Unique, easier to match

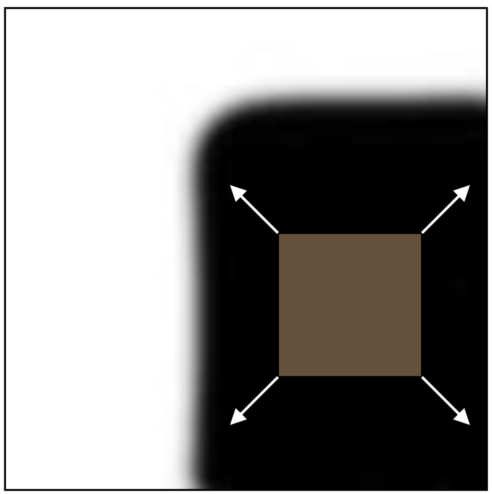
Corner detectors

- Why might a corner be more useful than an edge?
  - Edge: Constrained along 1D
  - Corner: Specific, fixed 2D location

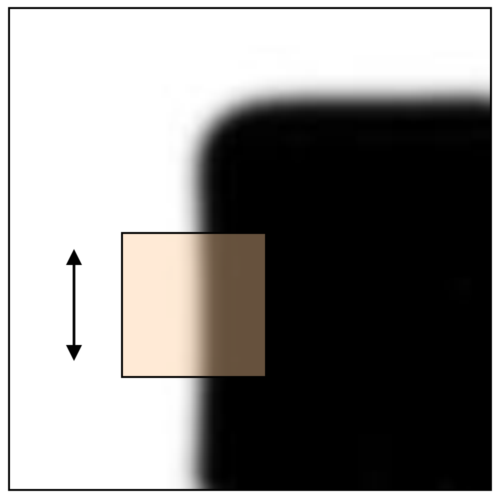


Corners as distinctive interest points

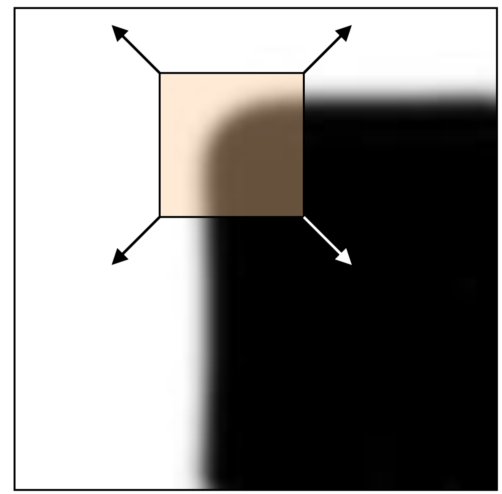
- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity



"flat" region:
no change in all directions



"edge":
no change along the edge direction

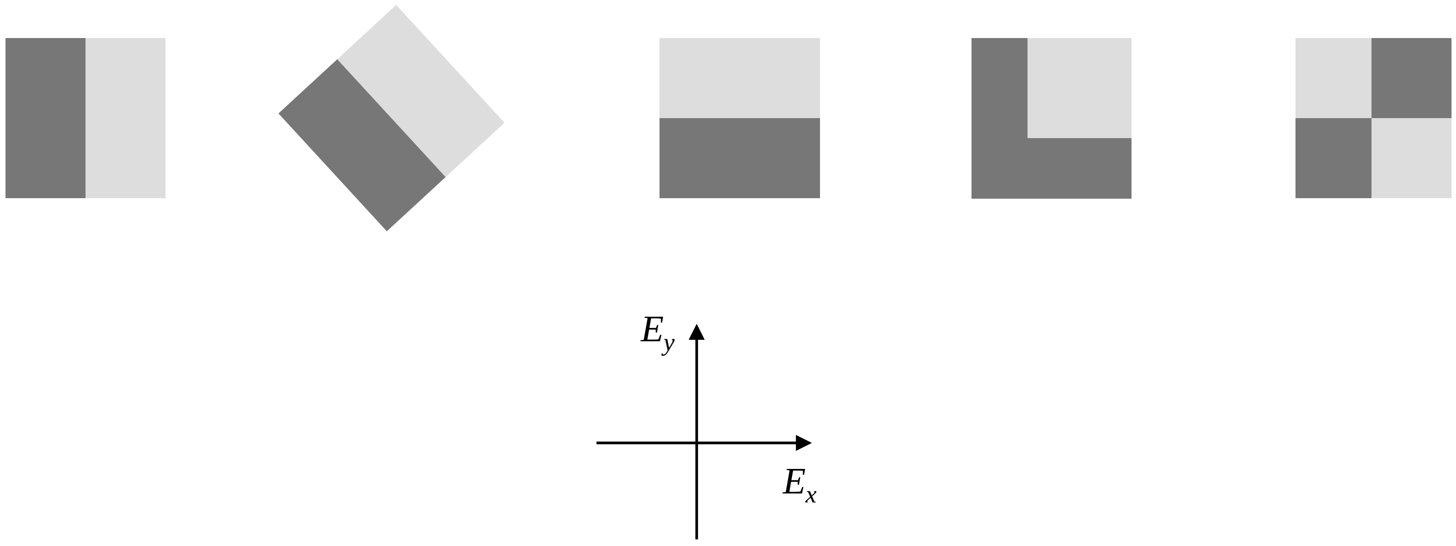


"corner":
significant change in all directions

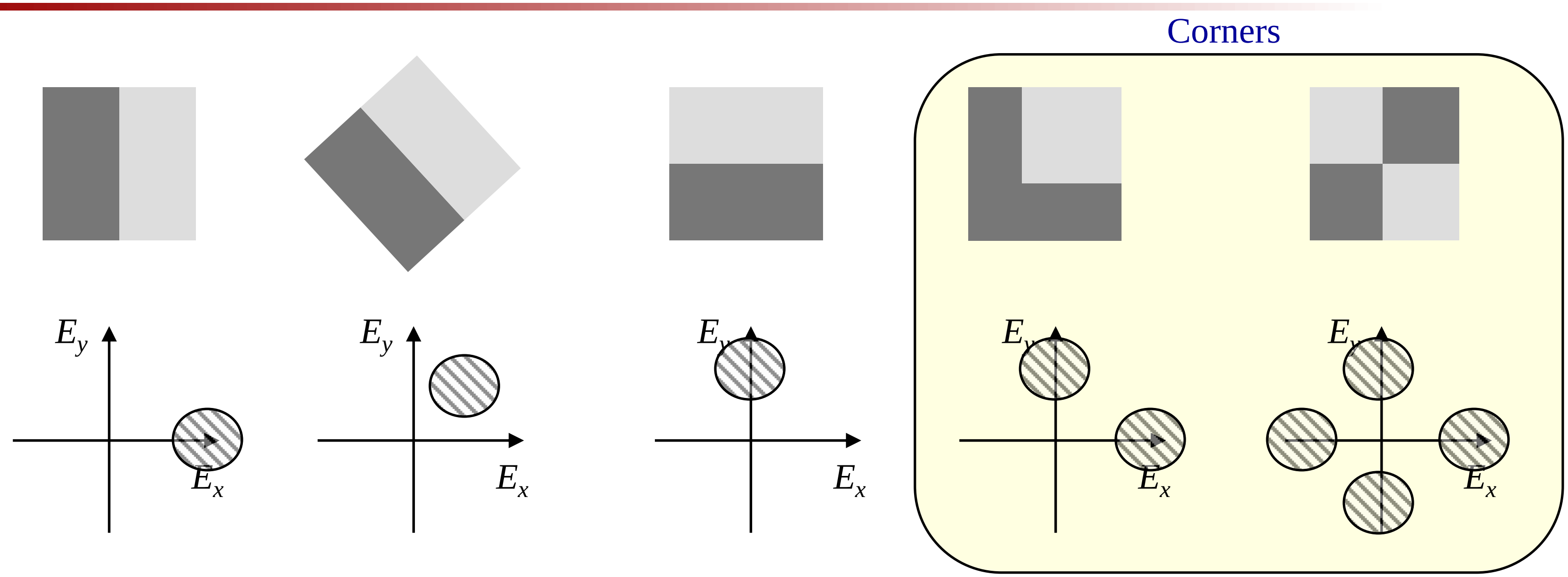
Source: A. Efros

Corner detectors

- One way to detect a corner:
  - Find an image patch where image gradients in both *x* and *y* directions are significant

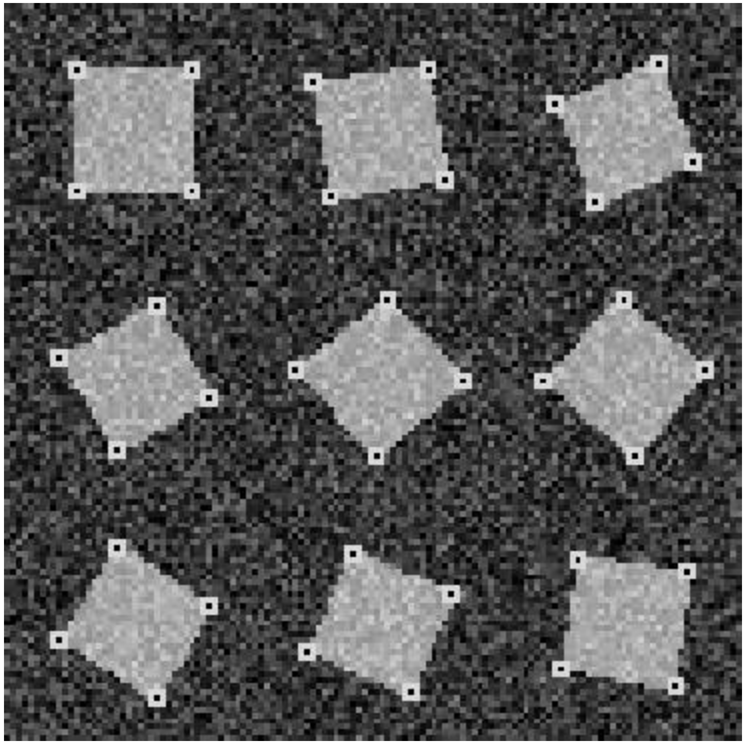


Corner detection

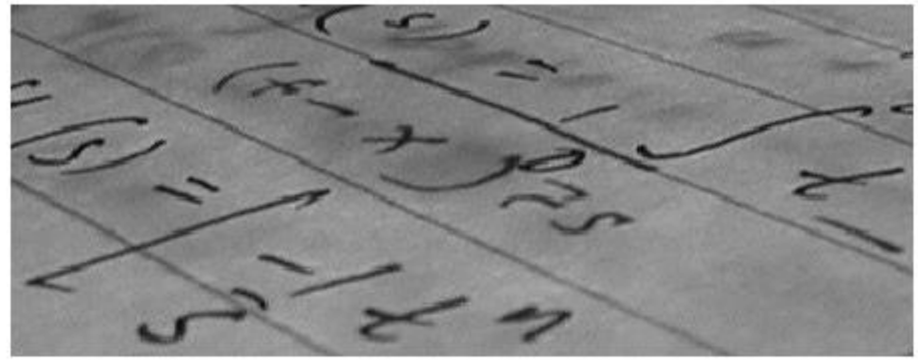


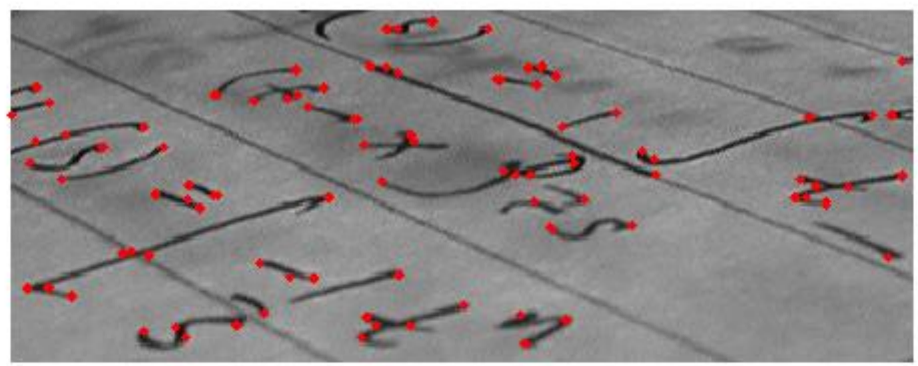
- We can create a corner detector by computing edge strength in *x* and *y* and then looking for certain combinations that describe a corner (e.g., via eigenvector analysis of the (E_x, E_y) space)
- Some detected corners will be spurious (not useful), but many will be meaningful

Examples



Examples

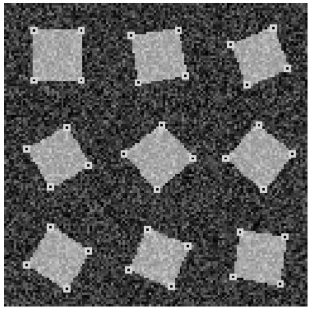




Why corners?

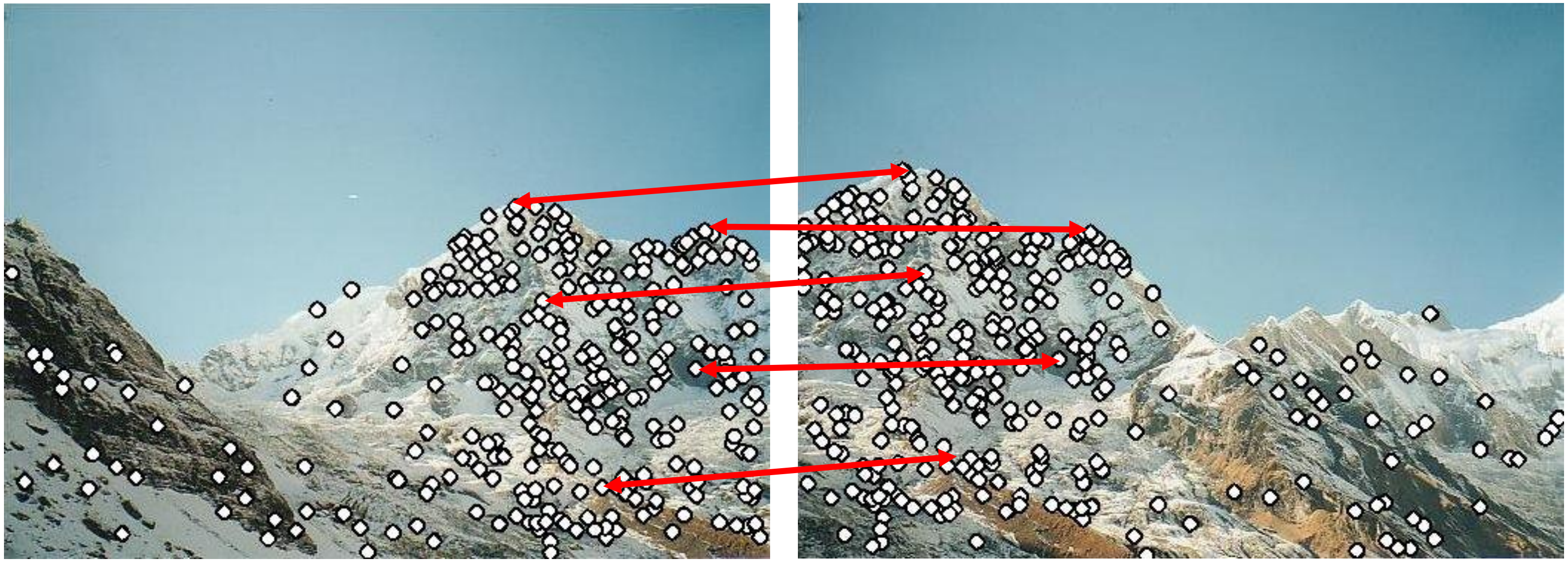
- Corners are typically discernable locations in images that correspond to meaningful aspects of the scene
  - Object corners, occlusion boundaries, sharp intensity changes, etc.

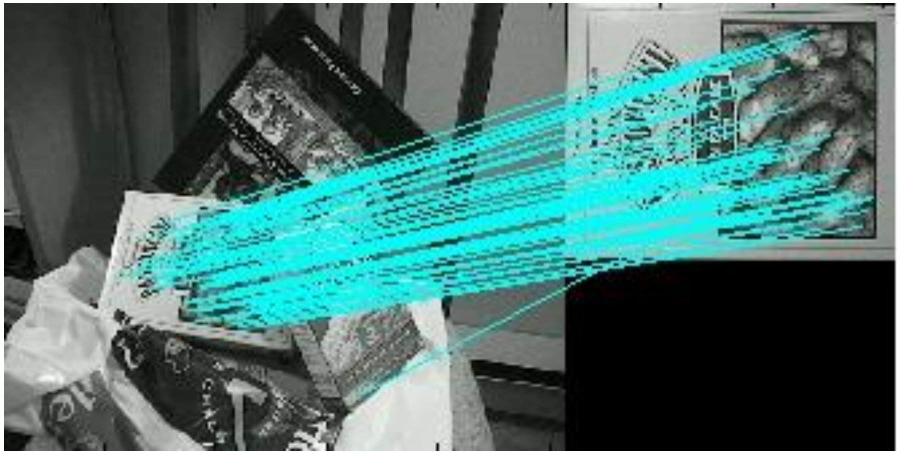
- They can help to form a description of an object or scene



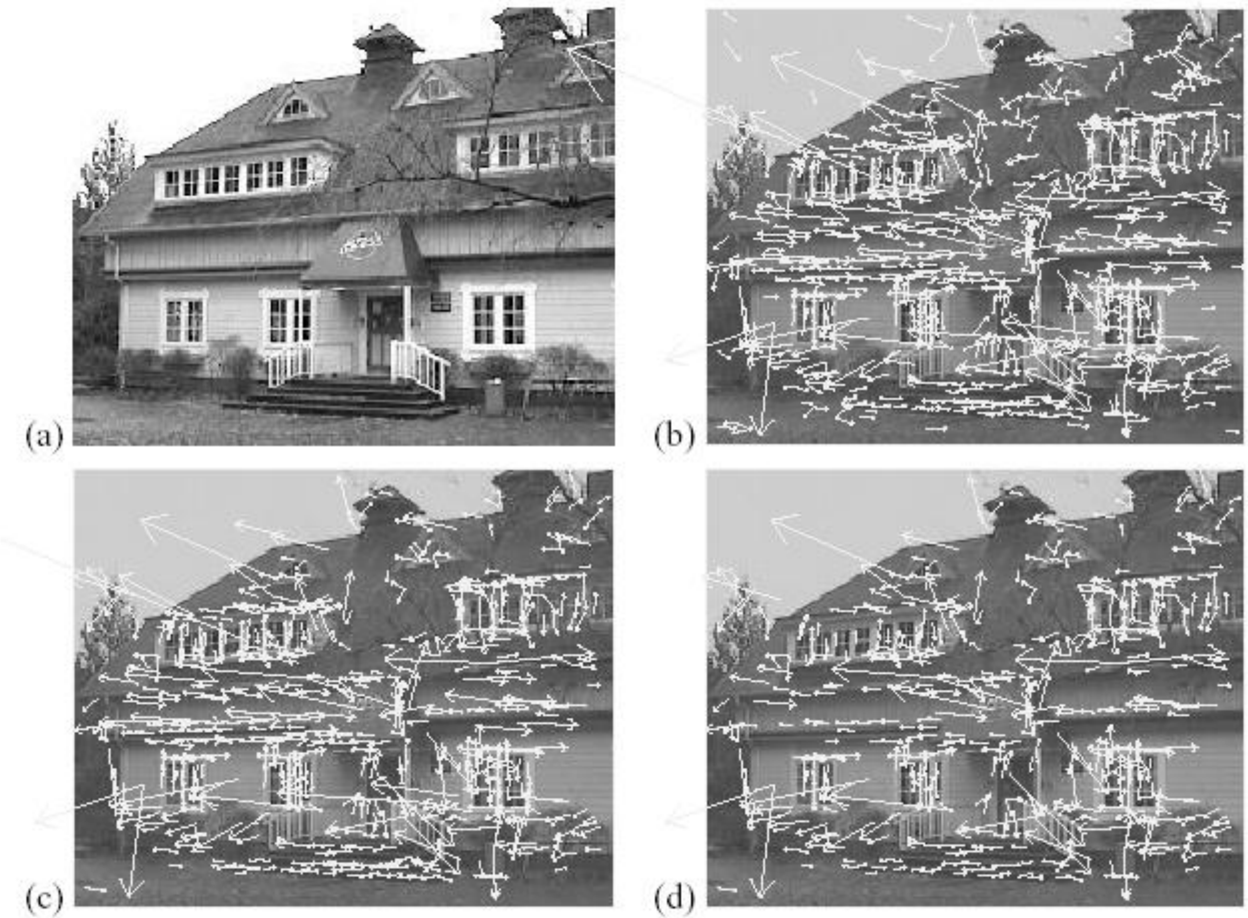
- They can help to make correspondence from one frame to another in multiple frames
  - Stereo and motion computation, tracking, image stitching, …

Matching corners?



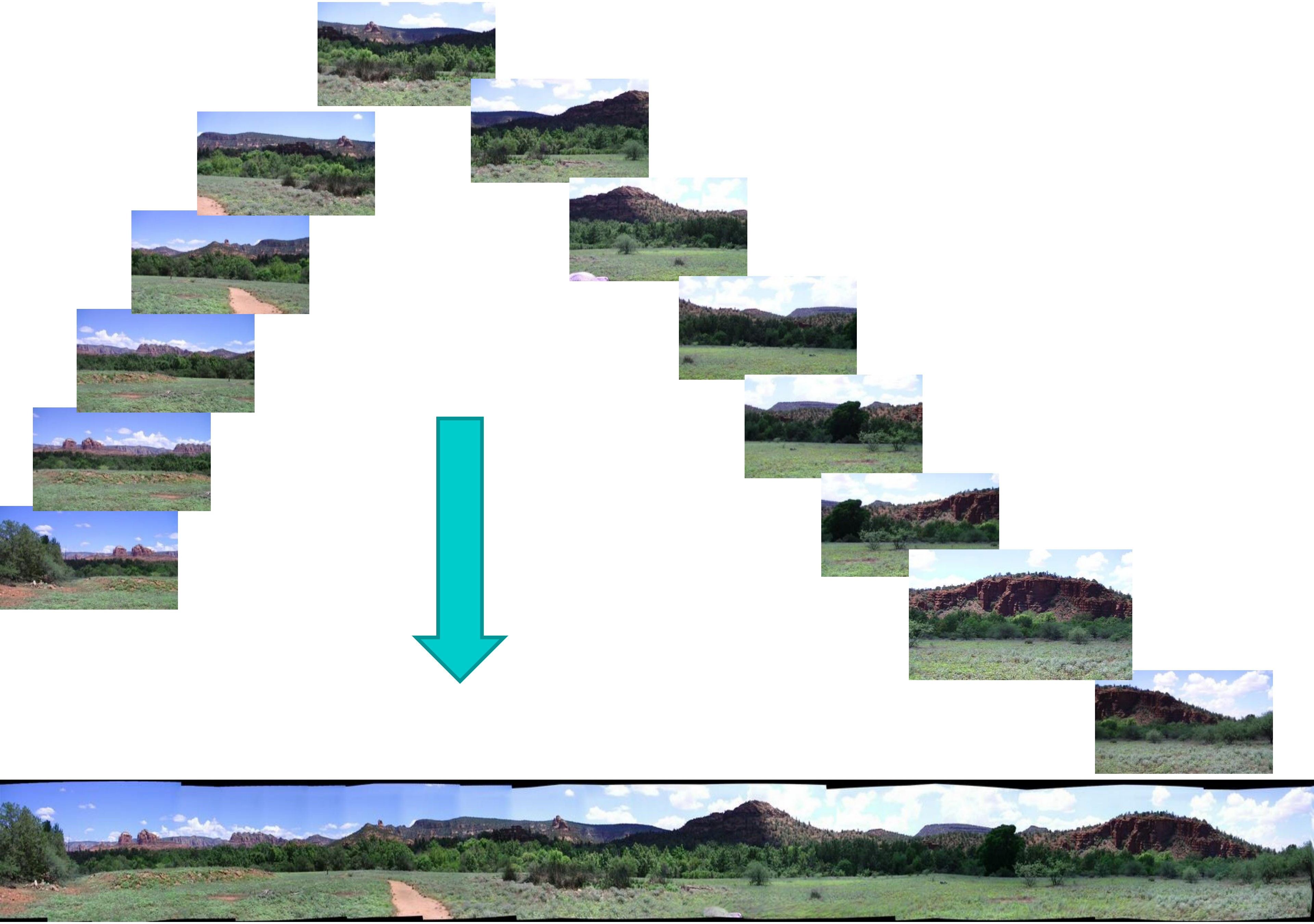


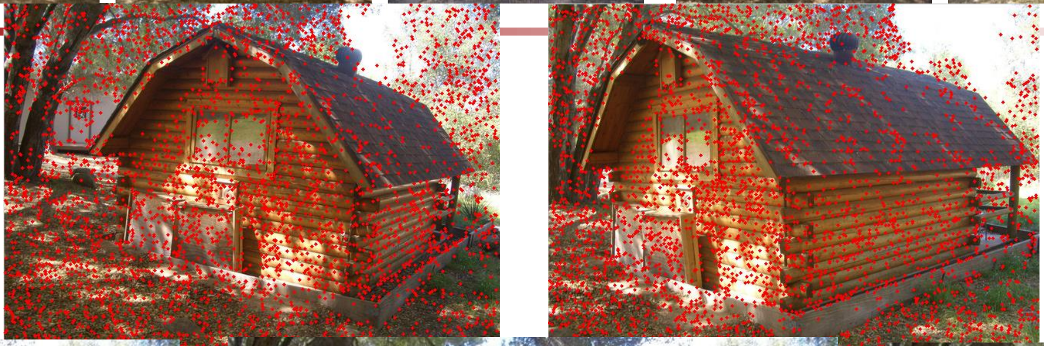
Example of keypoint detection



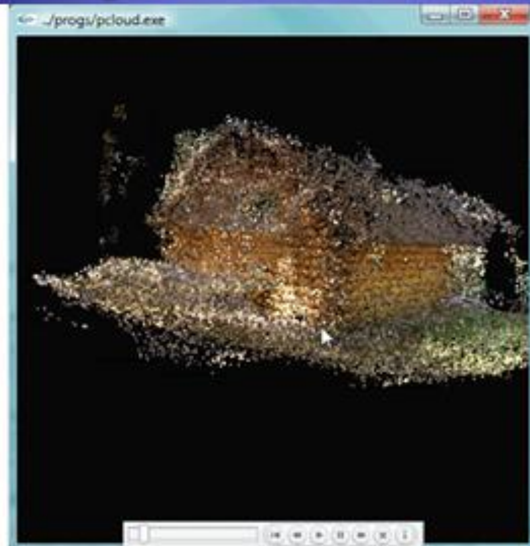
(a) (b) (c) (d)

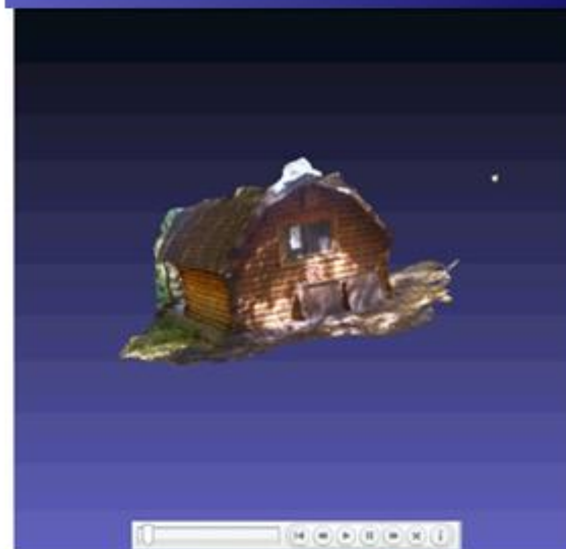
(a) 233x189 image
(b) 832 DOG extrema
(c) 729 left after peak value threshold
(d) 536 left after testing ratio of principle curvatures (removing edge responses)



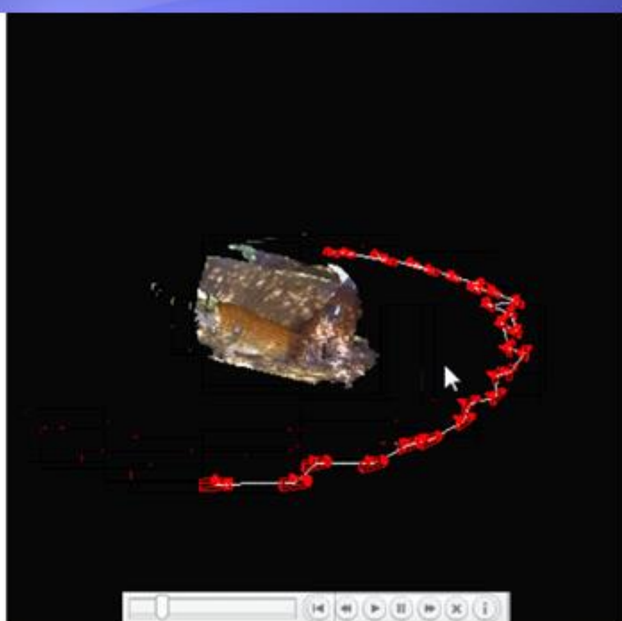



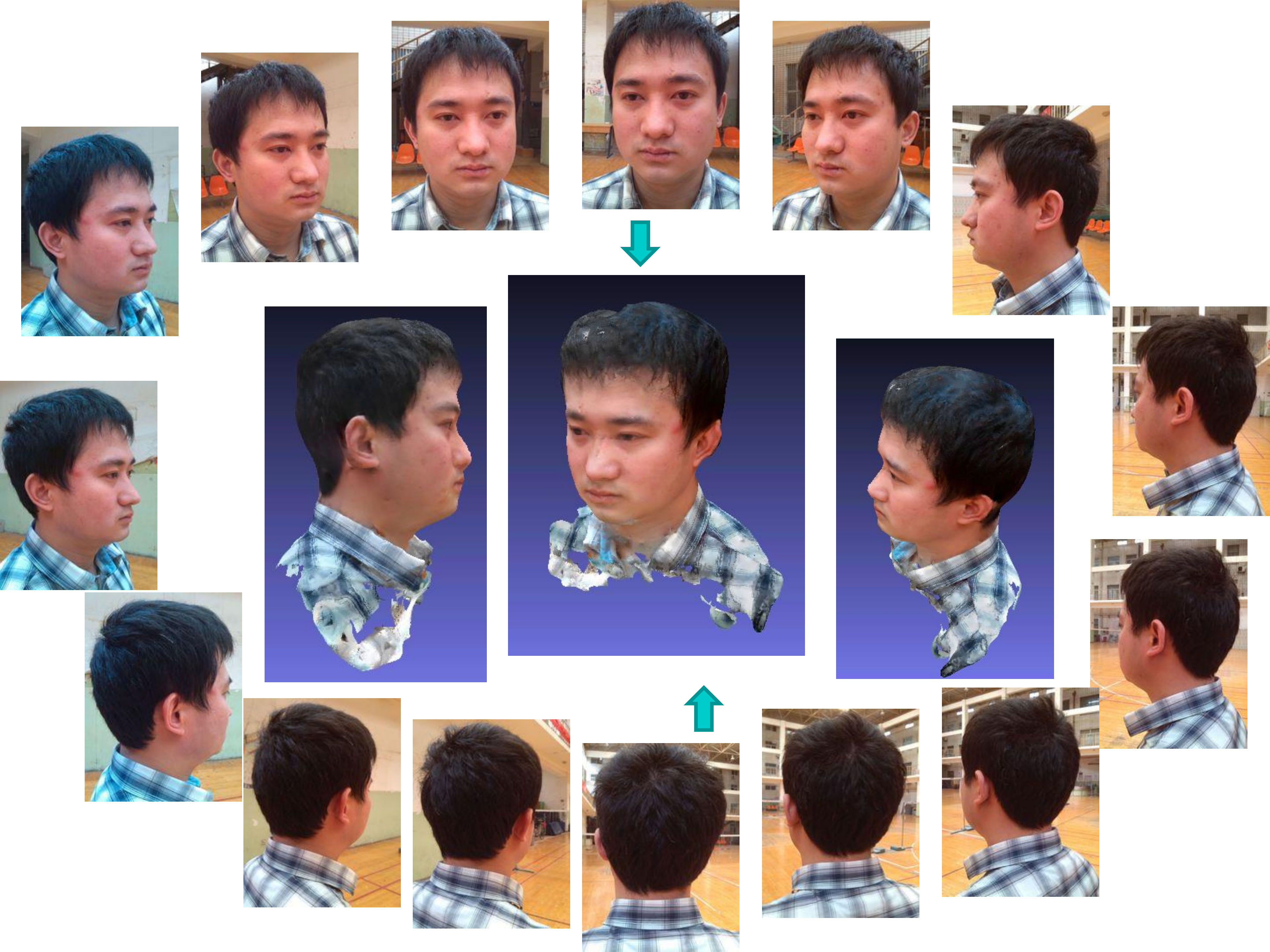

Object Structure

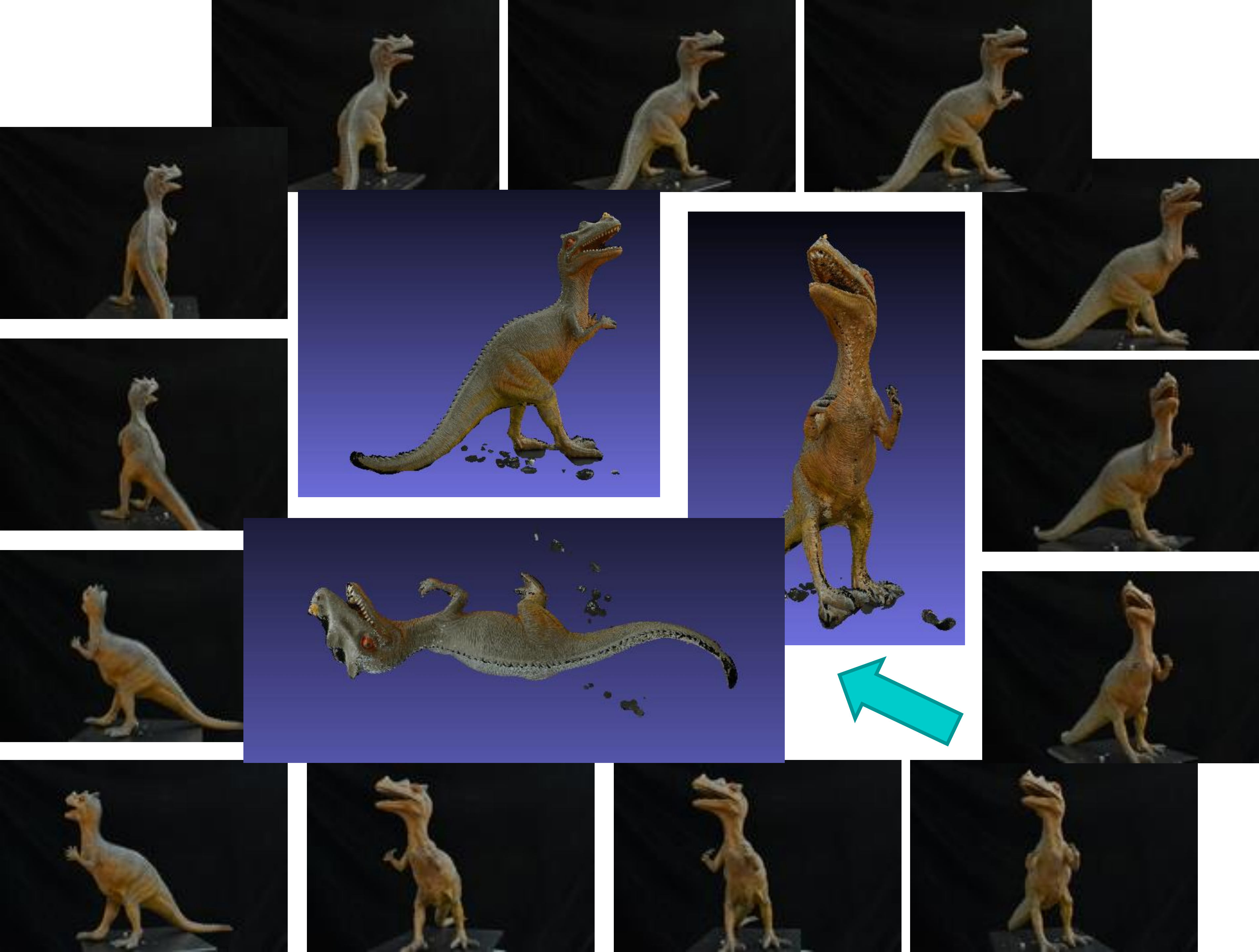




Camera Motion





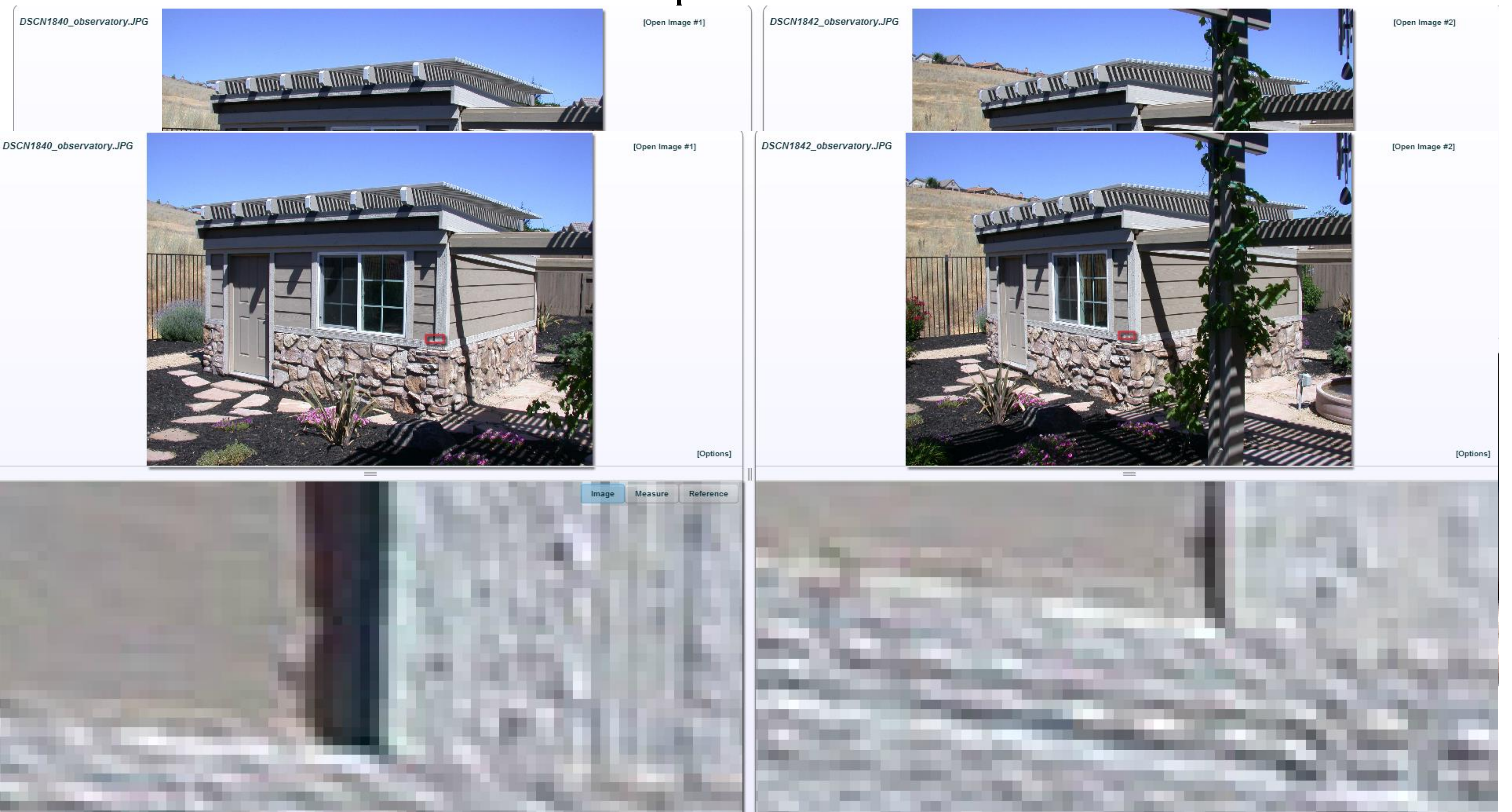


Interest point detection & description

- Corner detection is an example of interest point detection
- Ideally, an interest point:
  - Has a clear, mathematically described, definition
  - Has a well-defined position in the image space
  - Is computable from local information
  - Is stable under global and local perturbations of the image (changes in illumination, pose, scale, etc.)
- Interest points can include not only location (where is the point in the image?) but also a rich description that helps subsequent matching of interest points across images (what it is?)
- I.e., both questions:
  - ***Where*** it is?
  - ***What*** it is? Need answers

Why is this hard?

- Because a CV program doesn't have "visual common sense" and has a small aperture



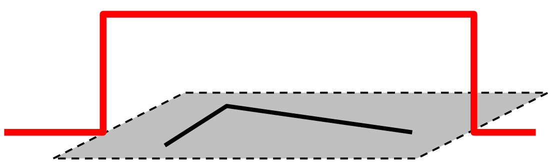
Harris Detector formulation

Change of intensity for the shift [u,v]:

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

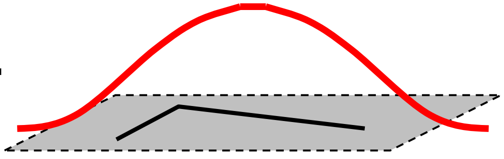
Window function — Shifted intensity — Intensity

Window function $w(x,y)$ =



1 in window, 0 outside

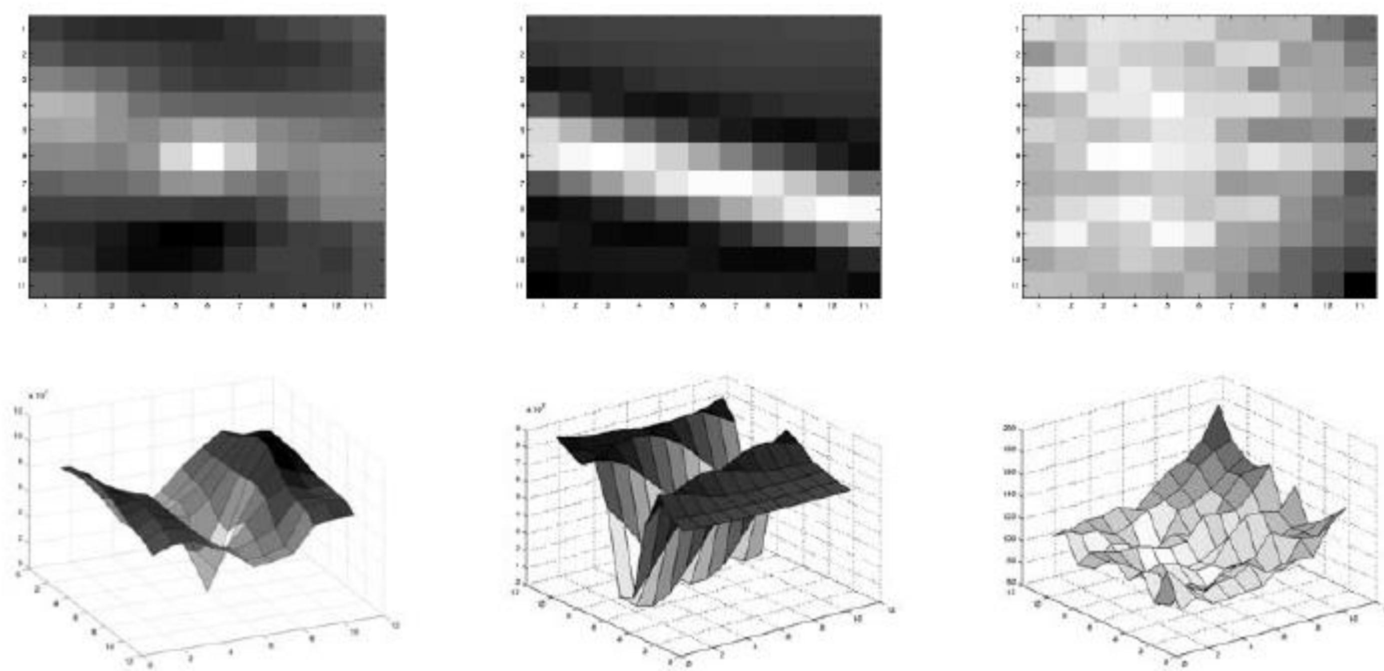
or



Gaussian

Source: R. Szeliski

(a)



Harris Detector formulation

This measure of change can be approximated by:

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

where M is a 2×2 matrix computed from image derivatives:

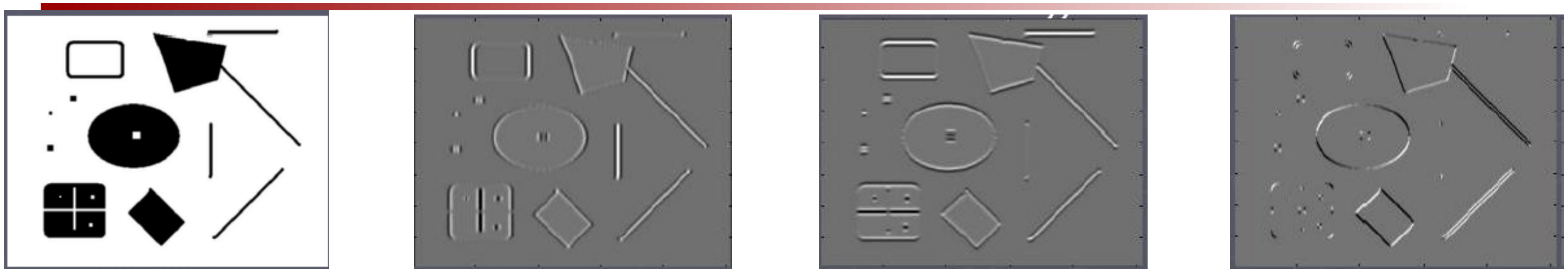
$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Gradient with respect to x, times gradient with respect to y

Sum over image region – area we are checking for corner

$$M = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \ I_y]$$

Harris Detector formulation



where M is a 2×2 matrix computed from image derivatives:

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

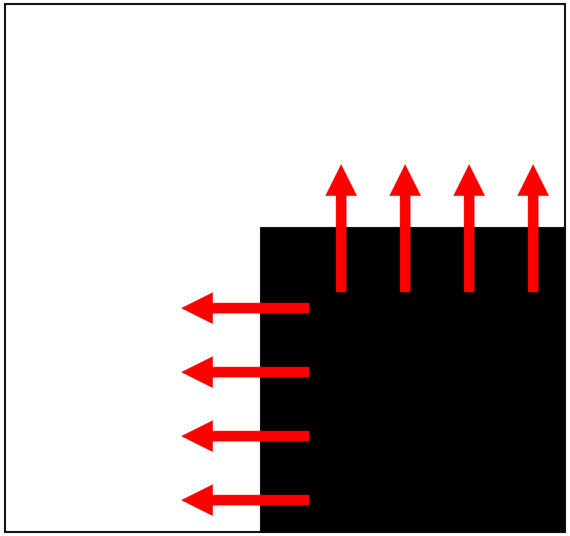
Gradient with respect to x, times gradient with respect to y

Sum over image region – area we are checking for corner

$$M = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \ I_y]$$

What does this matrix reveal?

First, consider an axis-aligned corner:



What does this matrix reveal?

First, consider an axis-aligned corner:

$$M = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

This means dominant gradient directions align with x or y axis

If either λ is close to 0, then this is **not** a corner, so look for locations where both are large.

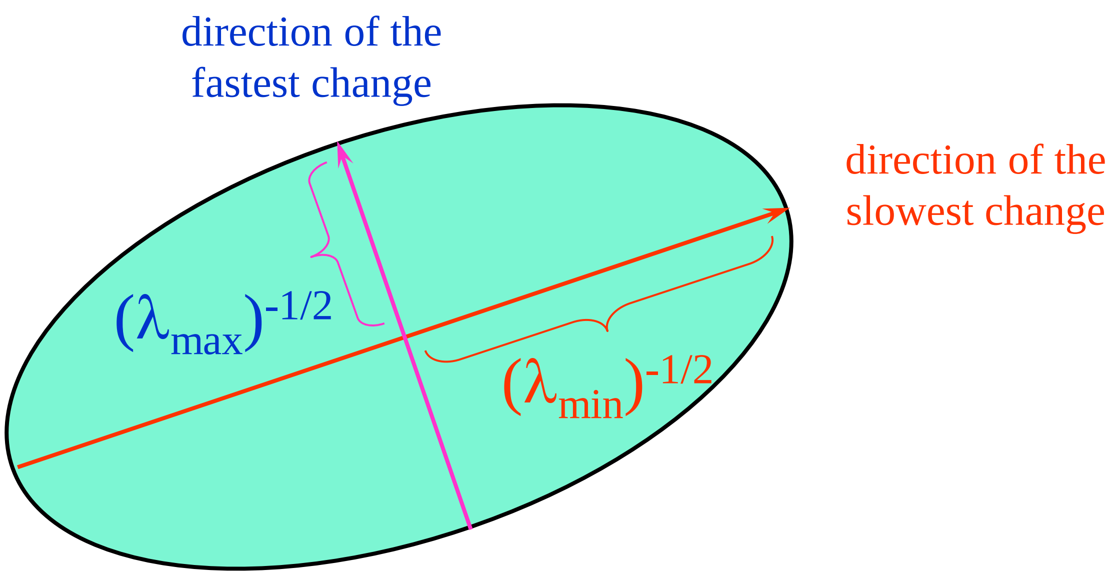
What if we have a corner that is not aligned with the image axes?

Slide credit: David Jacobs

General Case

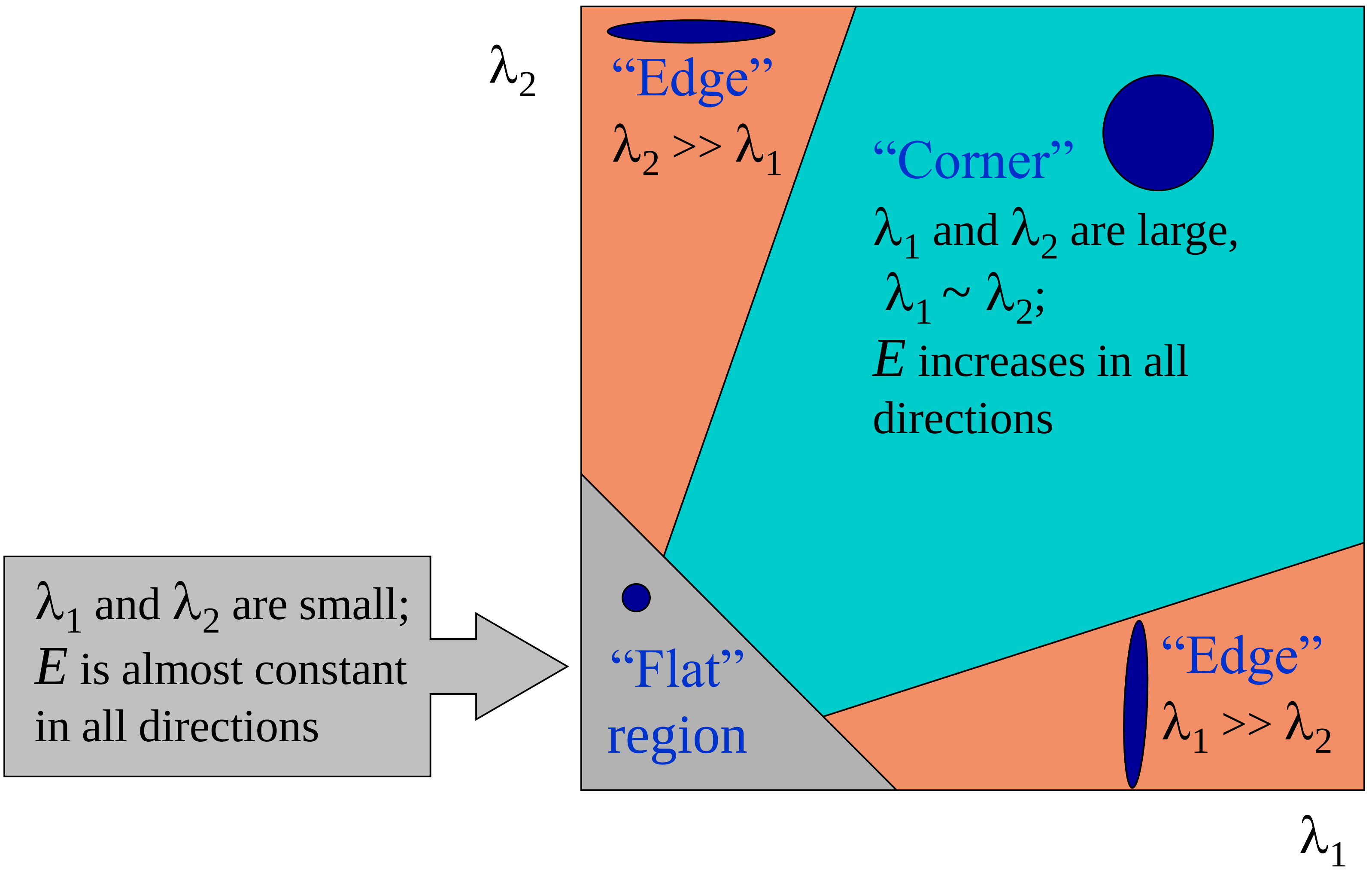
Since M is symmetric, we have $$M = R^{-1}\begin{bmatrix}\lambda_1 & 0 \\ 0 & \lambda_2\end{bmatrix}R$$

We can visualize *M* as an ellipse with axis lengths determined by the eigenvalues and orientation determined by *R*



Interpreting the eigenvalues

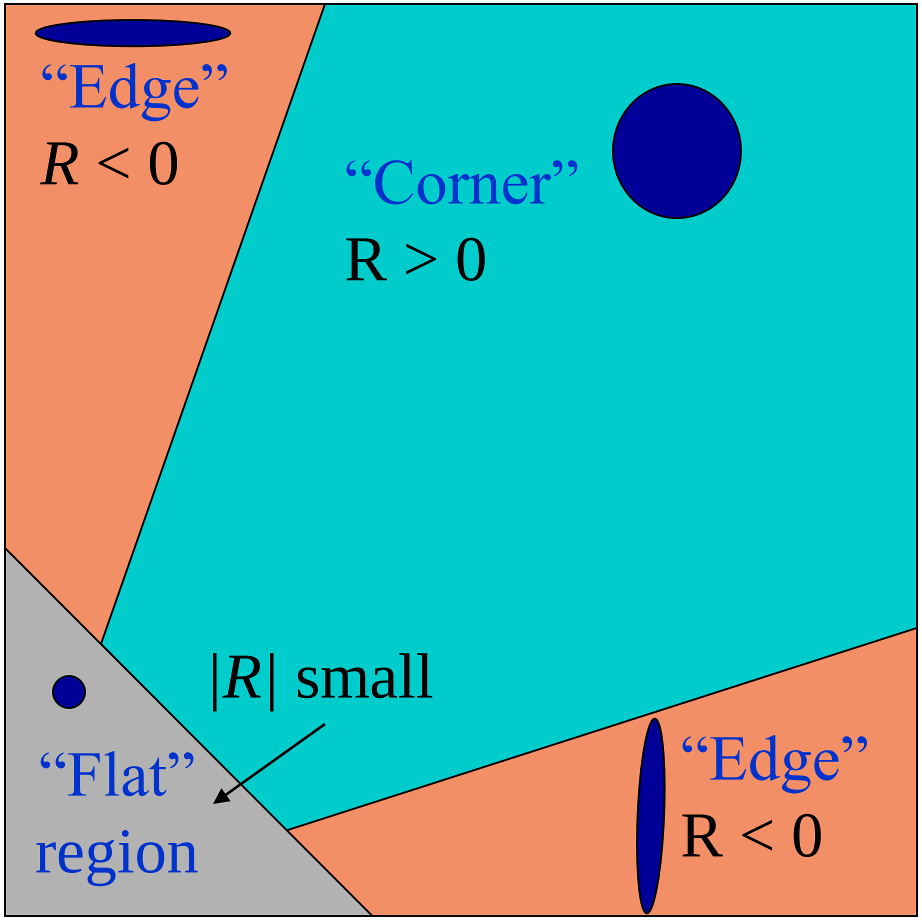
Classification of image points using eigenvalues of *M*:



Corner response function

$$R = \det(M) - \alpha\, \mathrm{trace}(M)^2 = \lambda_1 \lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$$

α: constant (0.04 to 0.06)



Harris Corner Detector

- Algorithm steps:
  - Compute M matrix within all image windows to get their R scores
  - Find points with large corner response

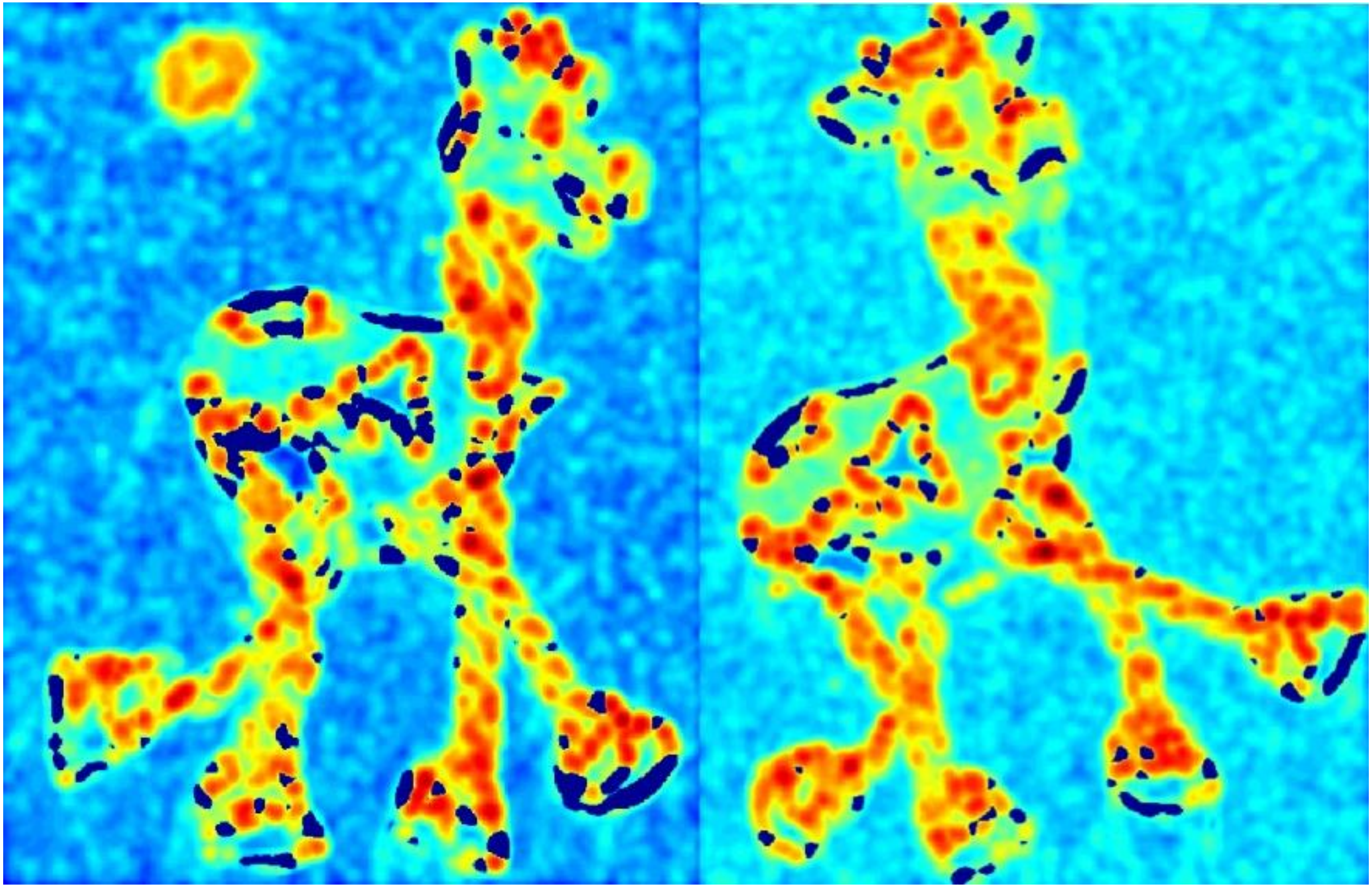
    (R > threshold)
  - Take the points of local maxima of R

Harris Detector: Workflow

Slide adapted form Darya Frolova, Denis Simakov, Weizmann Institute.

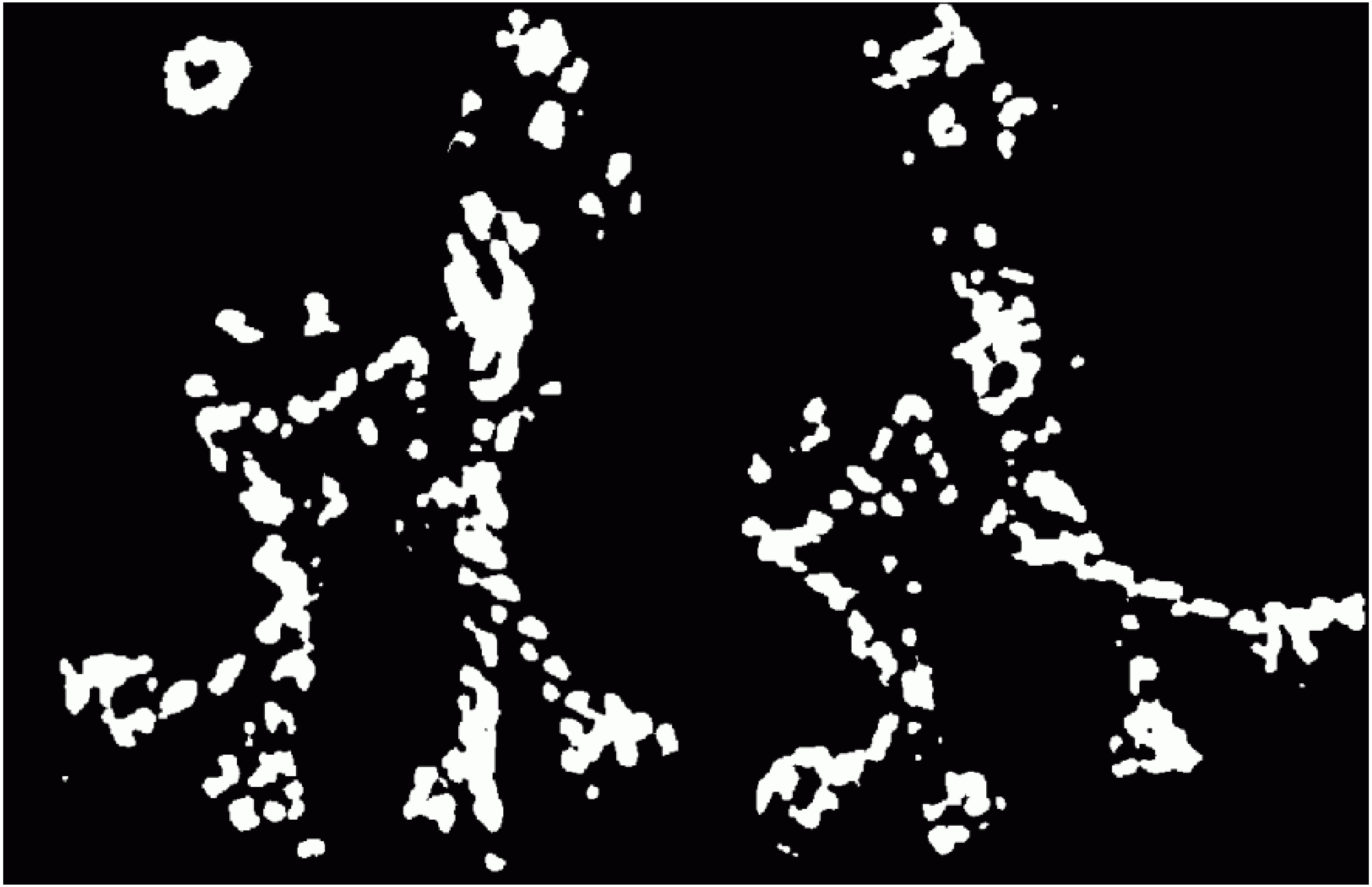
Harris Detector: Workflow

Compute corner response R



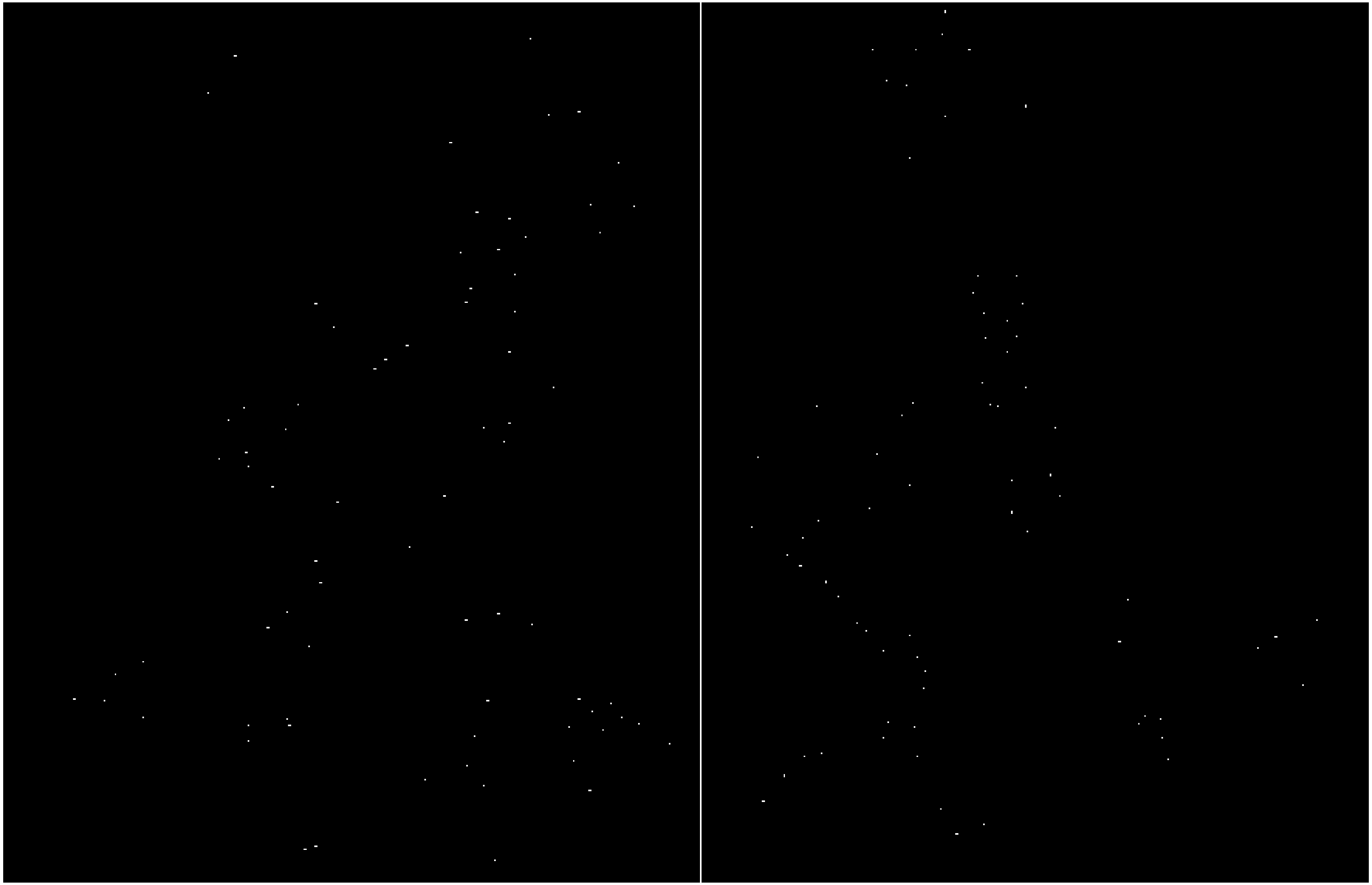
Harris Detector: Workflow

Find points with large corner response: R>threshold



Harris Detector: Workflow

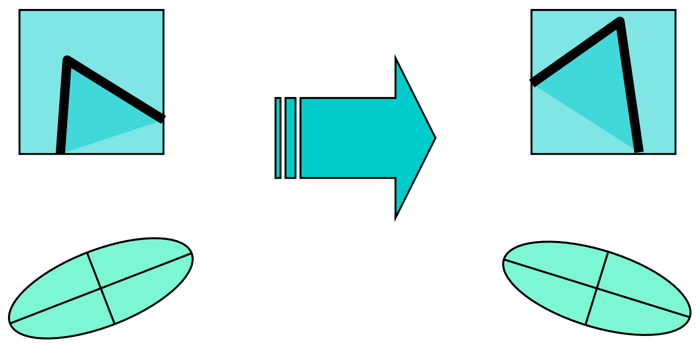
Take only the points of local maxima of R



Harris Detector: Workflow

Harris Detector: Properties

- Rotation invariance

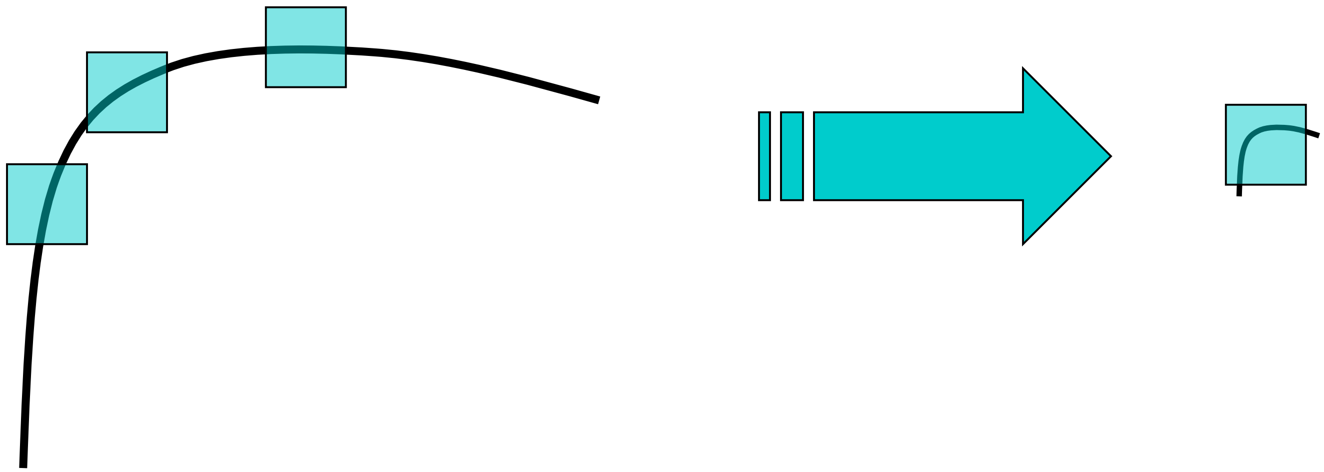


Ellipse rotates but its shape (i.e. eigenvalues) remains the same

Corner response R is invariant to image rotation

Harris Detector: Properties

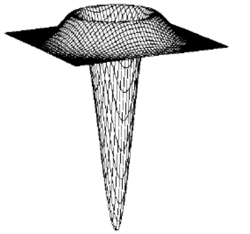
- Not invariant to image scale



All points will be classified as edges

Corner !

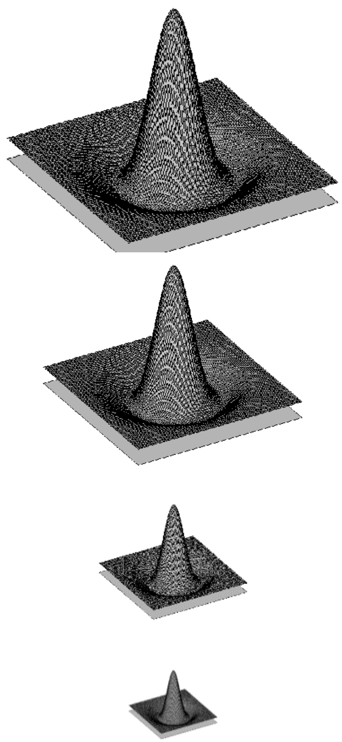
SIFT

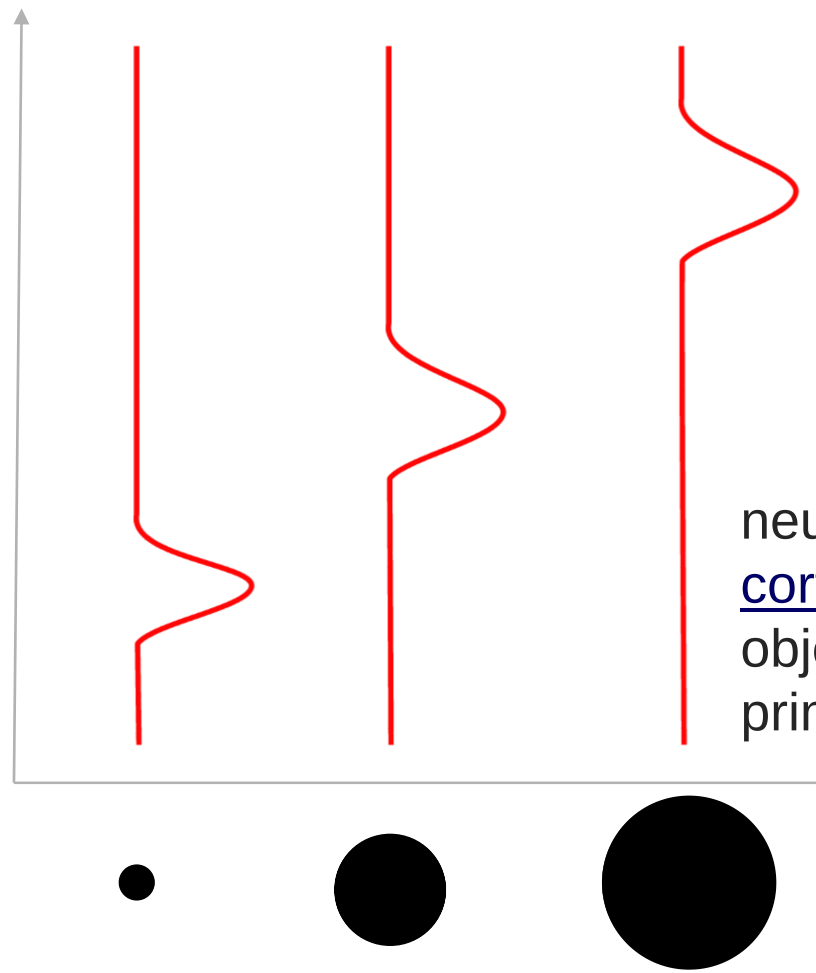


- Scale-invariant feature transform (SIFT) is an algorithm to detect and describe local features
- SIFT features are:
  - Invariant to image scale and in-plane rotation
  - Robust to changes in illumination, noise, and minor changes in viewpoint
  - Highly distinctive, relatively easy to extract
- The SIFT algorithm:
  - Detect extrema (max and min) after filtering with a Difference of Gaussian (DoG) at multiple scales
  - Eliminate unstable and weak points and localize (get the accurate position of) the good points
  - Assign orientation(s) to the points
  - Compute a full descriptor vector (128 elements) for each point

What Is A Useful Signature Function?

- Laplacian-of-Gaussian = "blob" detector

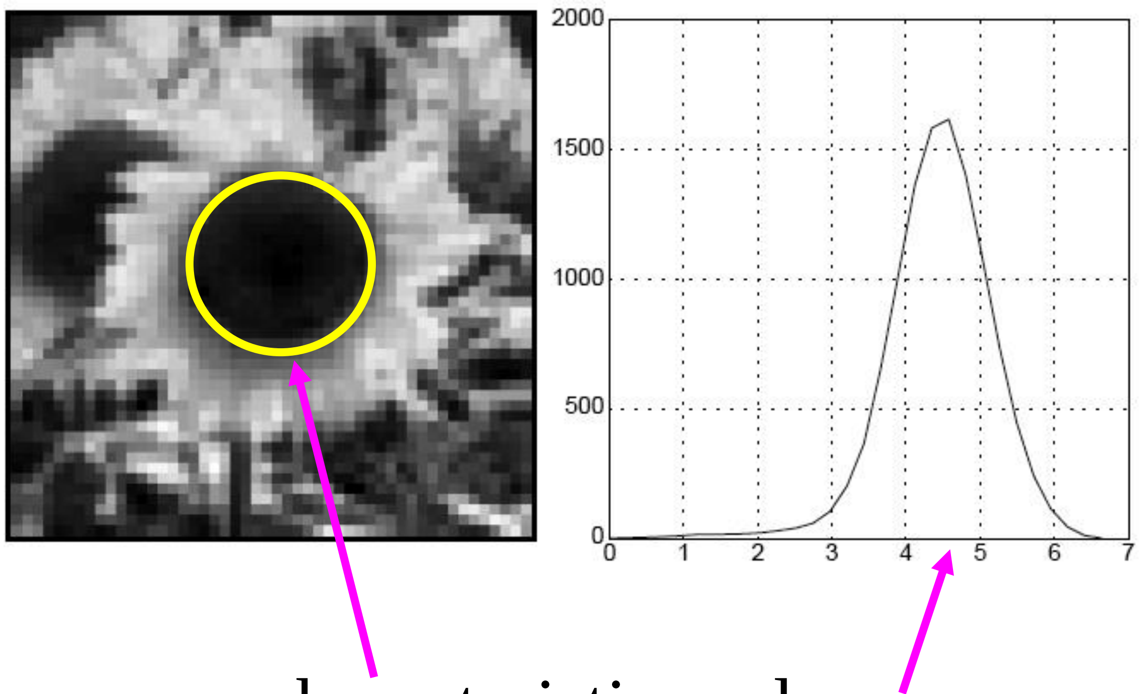




neurons in inferior temp
cortex that are used fo
object recognition in
primate vision

Characteristic scale

- We define the *characteristic scale* as the scale that produces peak of Laplacian response



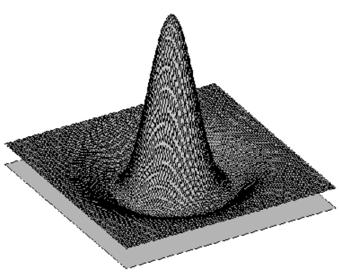
characteristic scale

T. Lindeberg (1998). "Feature detection with automatic scale selection." *International Journal of Computer Vision* **30** (2): pp 77--116.

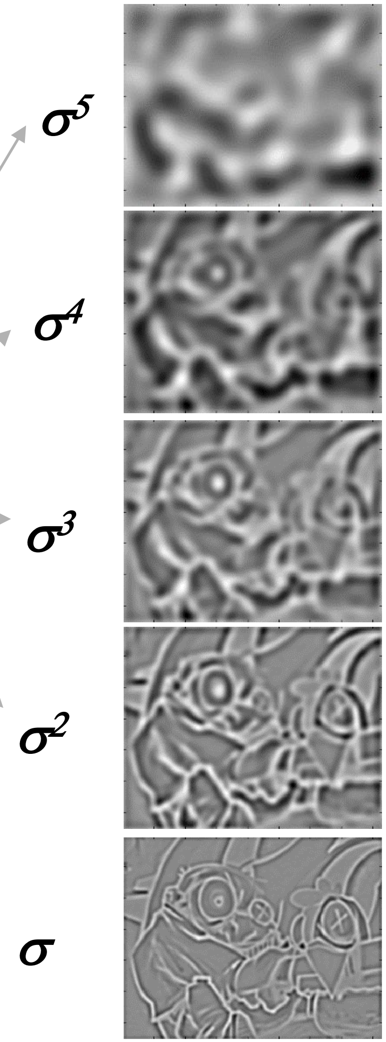
Laplacian-of-Gaussian (LoG)

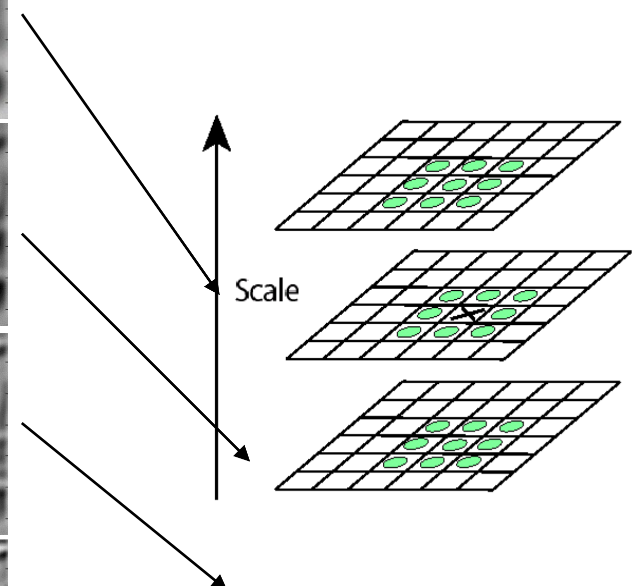
- Interest points:

  Local maxima in scale space of Laplacian-of-Gaussian



$$L_{xx}(\sigma) + L_{yy}(\sigma)$$





⇒ List of *(x, y, σ)*

Scale-space blob detector: Example

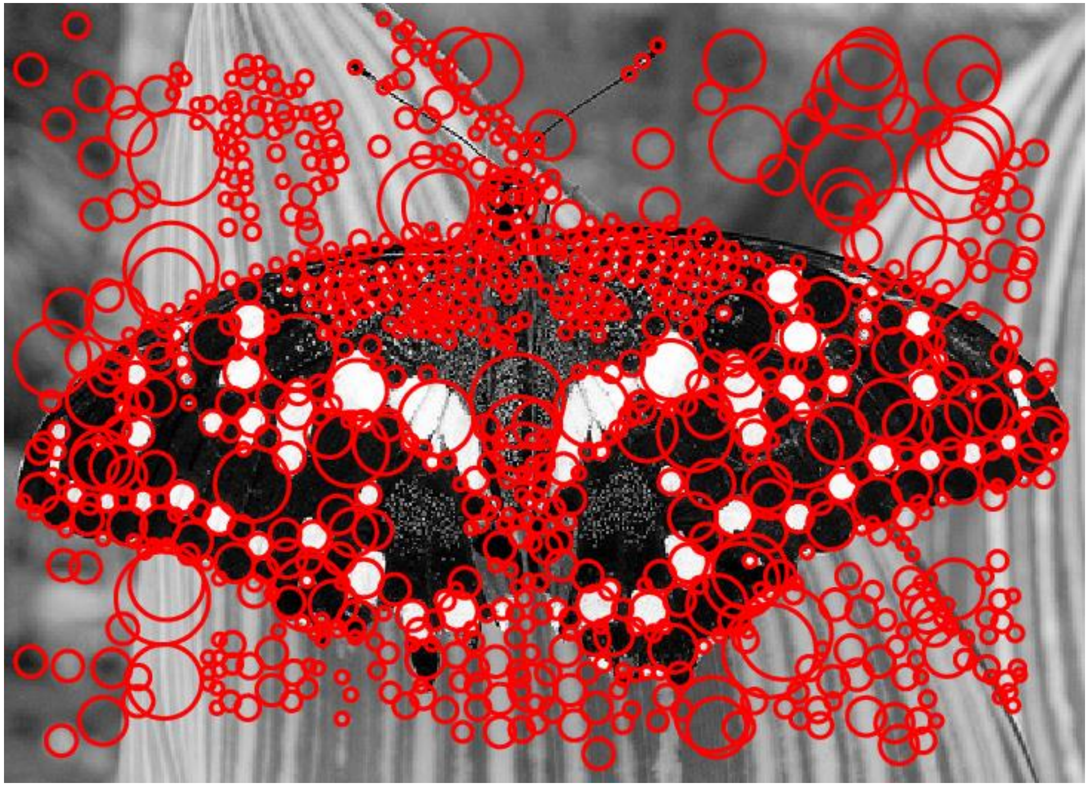
Source: Lana Lazebnik

Scale-space blob detector: Example

sigma = 11.9912

Source: Lana Lazebnik

Scale-space blob detector: Example



Source: Lana Lazebnik

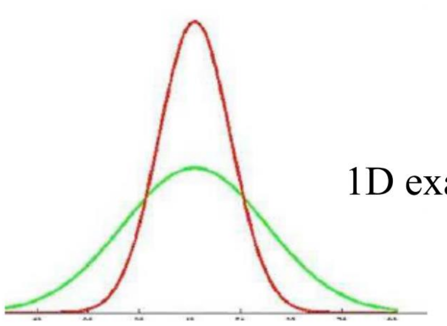
Key point localization with DoG

- Detect maxima of difference-of-Gaussian (DoG) in scale space
- Then reject points with low contrast (threshold)
- Eliminate edge responses

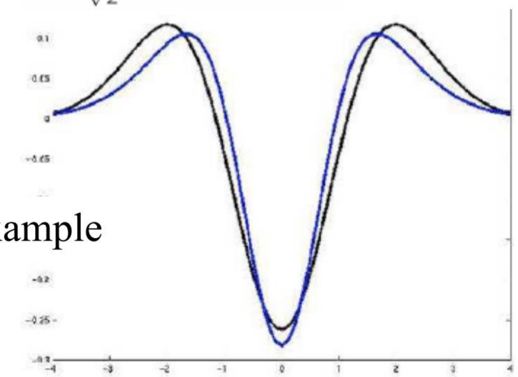
LoG can be approximate by a Difference of two Gaussians (DoG) at different scales

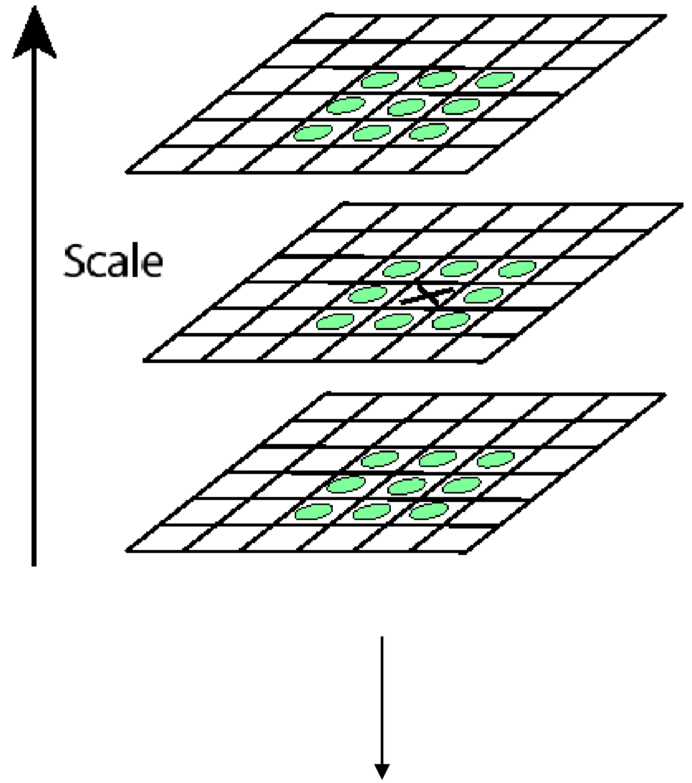
$$\nabla^2 G_\sigma \approx G_{\sigma_1} - G_{\sigma_2}$$

Best approximation when:
$\sigma_1 = \frac{\sigma}{\sqrt{2}}, \; \sigma_2 = \sqrt{2}\sigma$



1D example





Candidate keypoints: list of (x,y,σ)

Provide *scale invariance*

Key point localization w. Interpolation

- DoG (difference of Gaussian) in spatial (x,y) and scale space (σ) as input

$$D(\mathbf{x}) = D + \frac{\partial D}{\partial \mathbf{x}}^T \mathbf{x} + \frac{1}{2}\mathbf{x}^T \frac{\partial^2 D}{\partial \mathbf{x}^2}\mathbf{x}$$

- **x**= (x,y, σ)
- Derivatives calculated at the candidate key point
- Provide *subpixel precision*

Orientation assignment

- L(x,y) at key point scale σ

$$m(x,y) = \sqrt{\left(L(x+1,y) - L(x-1,y)\right)^2 + \left(L(x,y+1) - L(x,y-1)\right)^2}$$

$$\theta(x,y) = \operatorname{atan2}\left(L(x,y+1) - L(x,y-1),\, L(x+1,y) - L(x-1,y)\right)$$

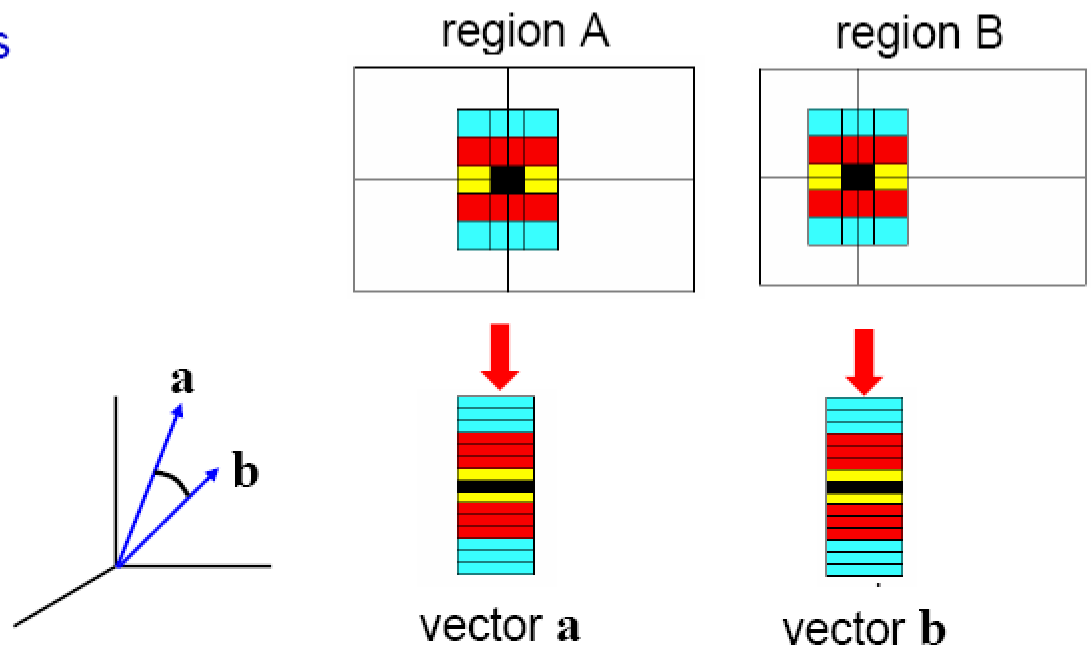
- m (magnitude), θ (angle) computed around key point neighborhood, using histogram
  - 36 bins (10 degree each)
  - Added by magnitude and Gaussian weighted distance from center
- Provide *rotation invariance*

Local descriptors

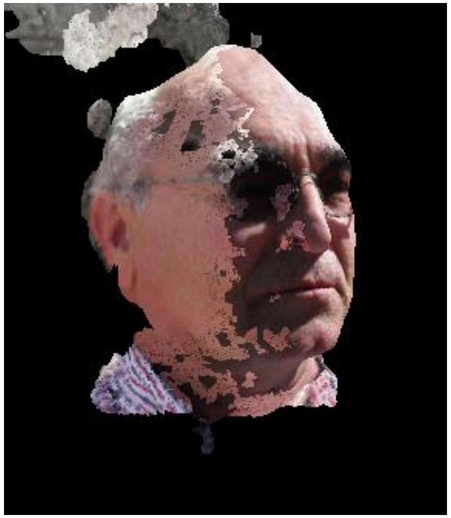
- Simplest descriptor: list of intensities or colors within a patch.
- What is this going to be invariant to?

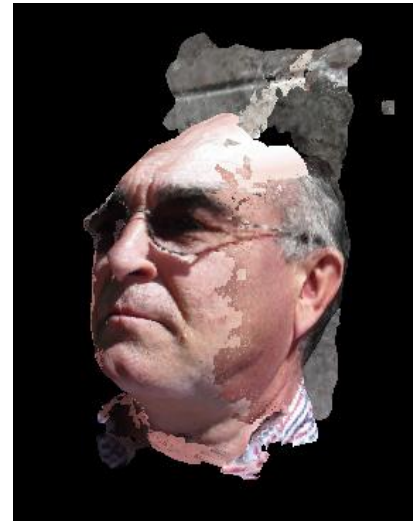
Write regions as vectors

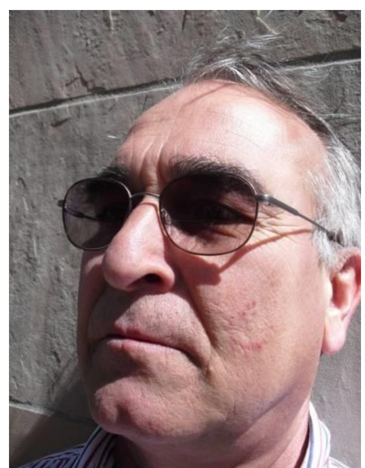
$$A \rightarrow \mathbf{a}, \;\; B \rightarrow \mathbf{b}$$



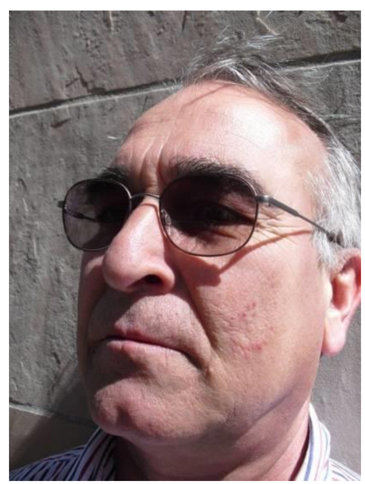
Is Color Invariant (outdoors)?

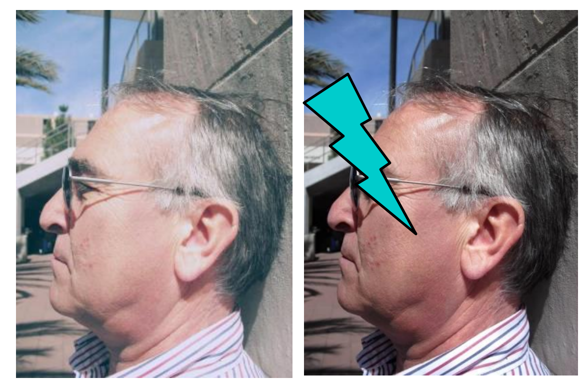







Radiometry Calibration

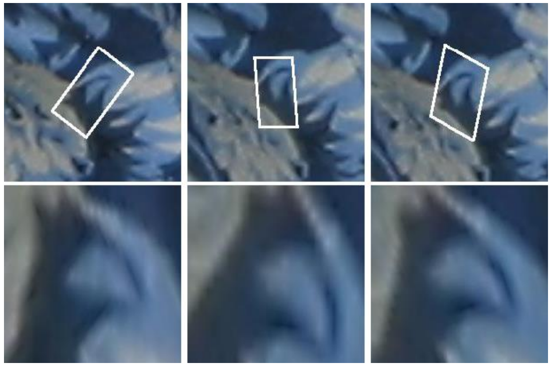




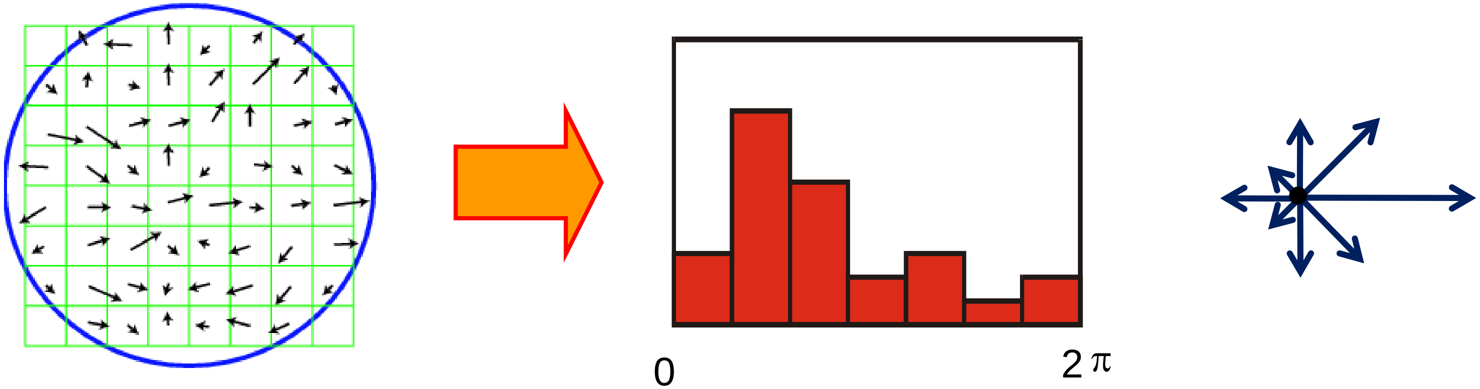
Is color invariant (indoors)?

Feature descriptors

- Disadvantage of patches as descriptors:
  - Small shifts can affect matching score a lot



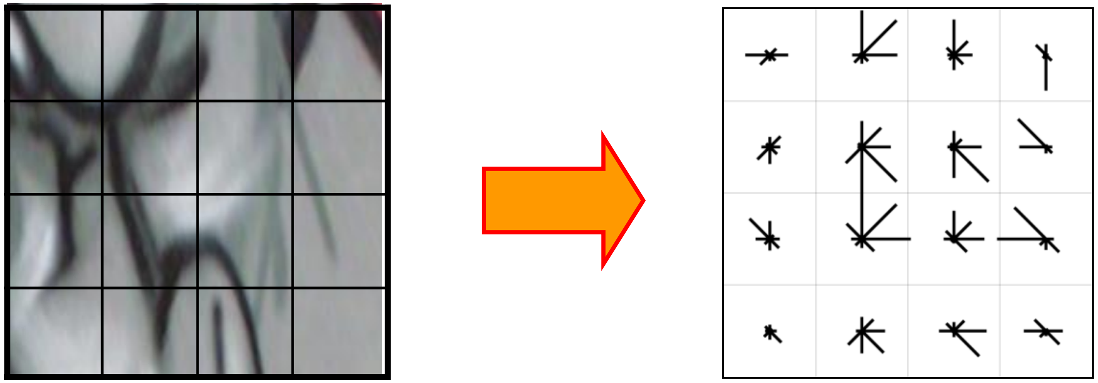
- Solution: histograms



Source: Lana Lazebnik

Feature descriptors: SIFT

- **S**cale **I**nvariant **F**eature **T**ransform
- Descriptor computation:
  - Divide patch into 4x4 sub-patches: 16 cells
  - Compute histogram of gradient orientations (8 reference angles) for all pixels inside each sub-patch
  - Resulting descriptor: 4x4x8 = 128 dimensions

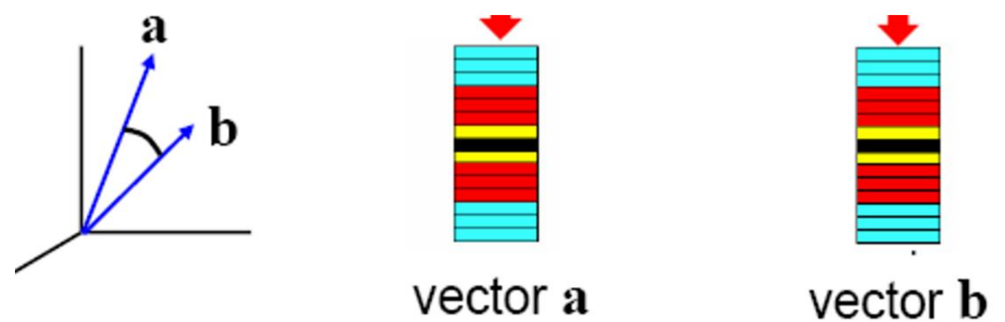


David G. Lowe. "Distinctive image features from scale-invariant keypoints." *IJCV* 60 (2), pp. 91-110, 2004.

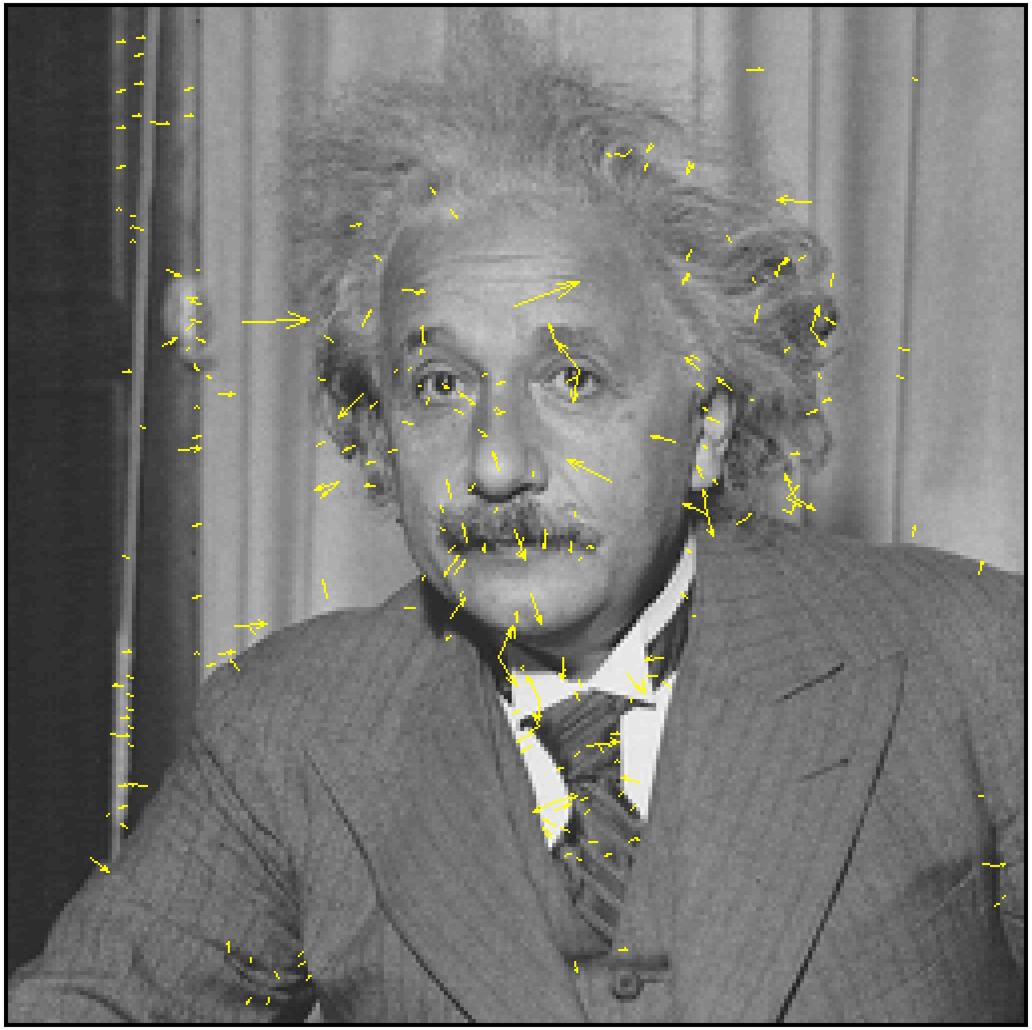
Source: Lana Lazebnik

Matching descriptors

- Euclidian distance between 2 128-bit vectors
  - No color
  - No intensity
  - Gradient is more robust
  - Multiple local histograms (not a single big one) -> tolerate certain occlusion
- Caveat
  - Not enough to say vector a is close to vector b (how about it is also close to vector c?)
  - Closest vector must beat out 2^{nd} closest vector with margin to spare

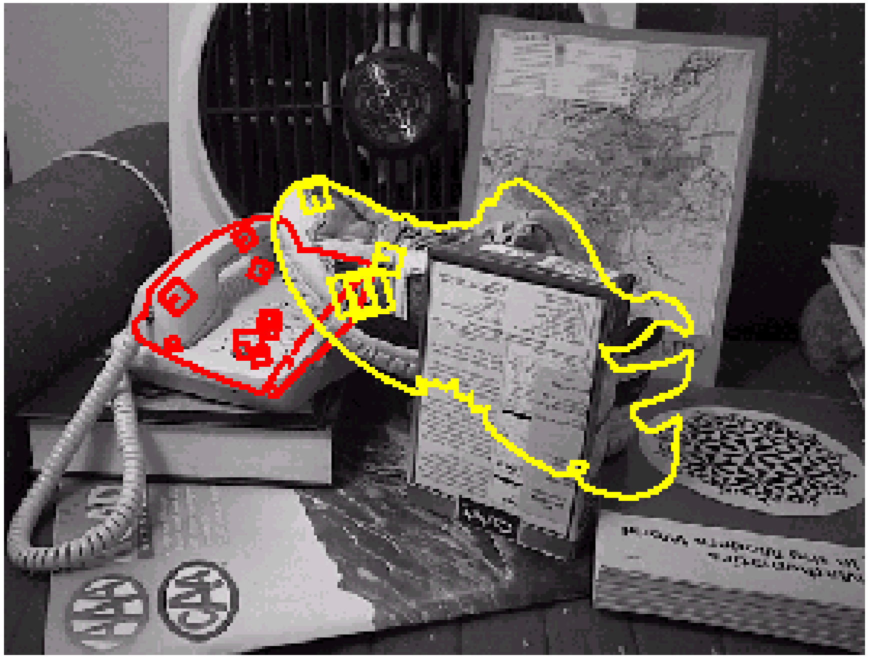


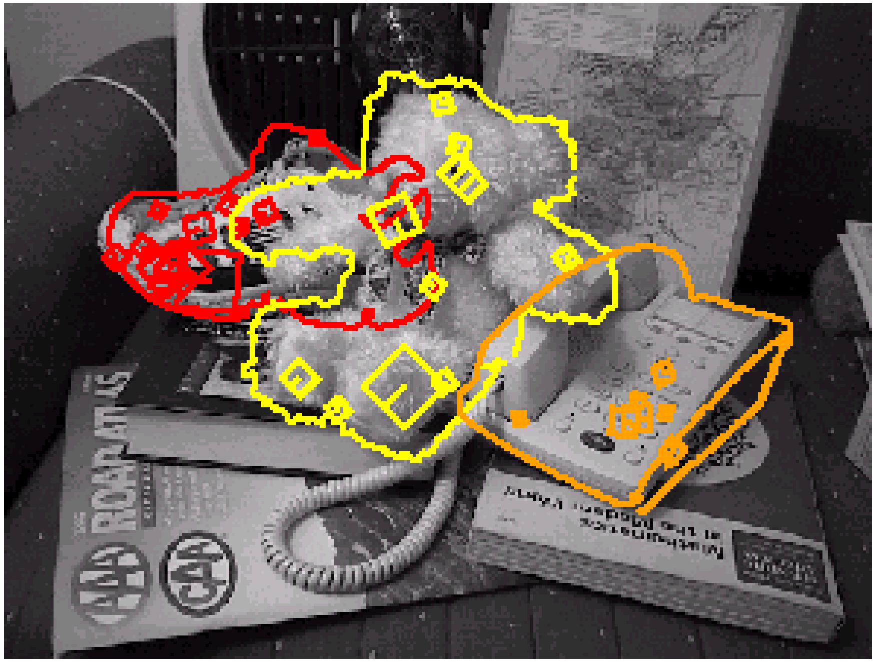
SIFT examples



Using SIFT to describe and recognize objects

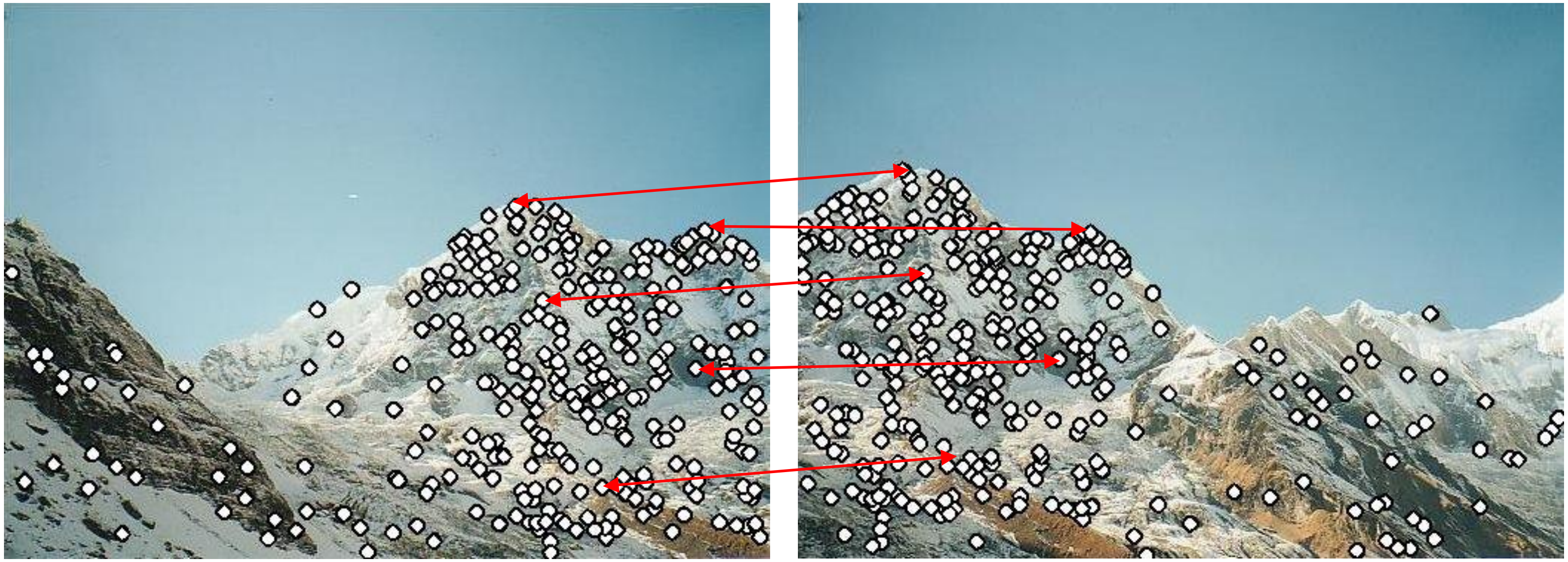
…and in the presence of occlusion





Local features and alignment

- Detect feature points in both images
- Find corresponding pairs



[Darya Frolova and Denis Simakov]

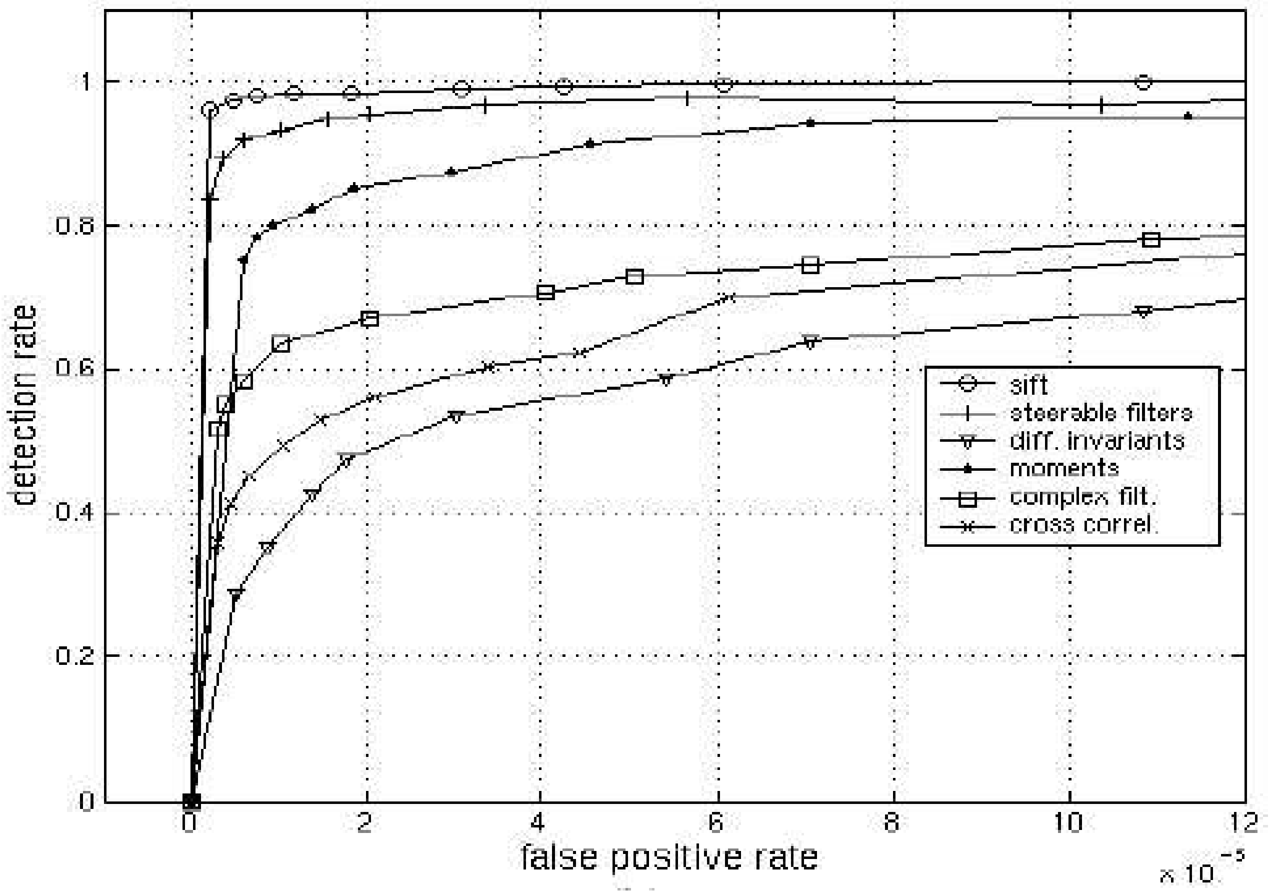
Local features and alignment

- Detect feature points in both images
- Find corresponding pairs
- Use these pairs to align images

[Darya Frolova and Denis Simakov]

Using SIFT for panorama stitching

ROC comparing SIFT performance to others



Hand-crafted Features

- SIFT and SURF have seen tremendous success in applications
- Followed up by FAST and BRIFF (simple and fast)
- Designed by Experts who “know what they are doing”
- How general are they? How good are they for different applications?
- Intuition: Fourier bases can be applied everywhere, there might be better bases (wavelets) tuned for different applications