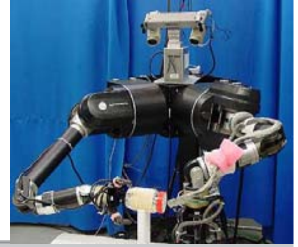
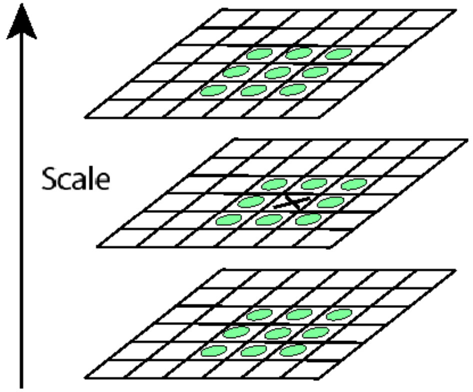
Einführung in Visual Computing

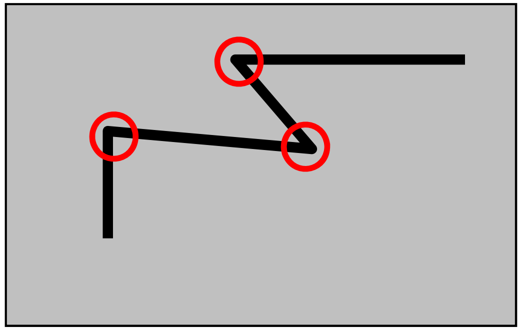
Unit 17: Image Features - Interest Points

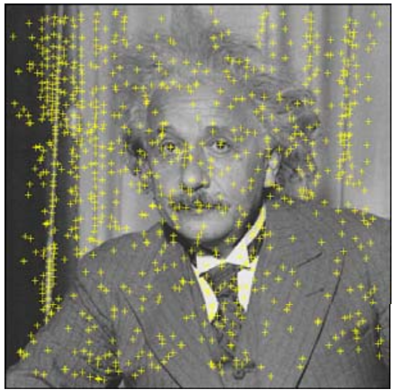


http://www.caa.tuwien.ac.at/cvl/teaching/sommersemester/evc

- Content:
  - Introduction to Image Features
  - Edges
  - Corners
    - Moravec Corner Detector
    - Harris Corner Detector
  - SIFT







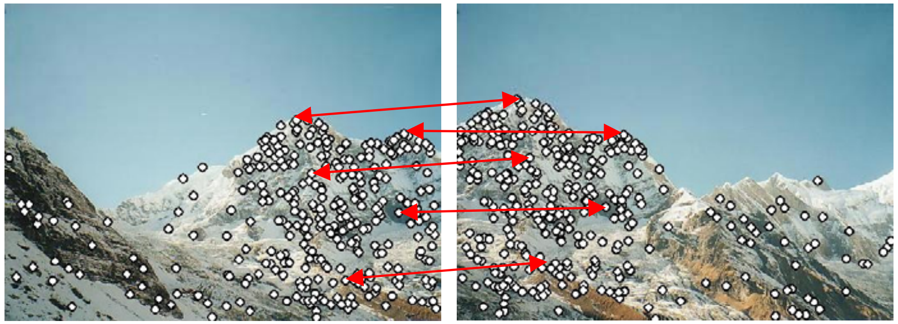
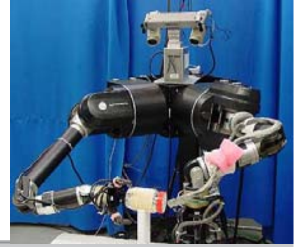
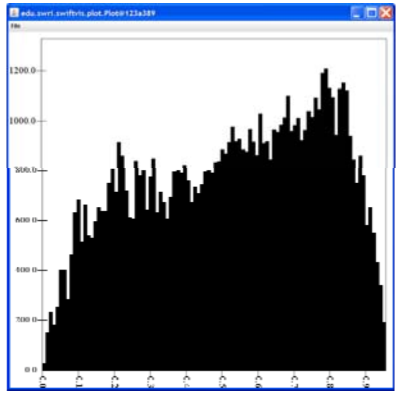


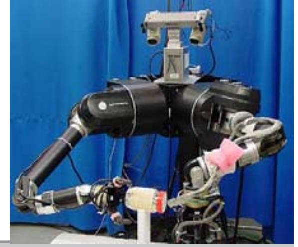
Image Features



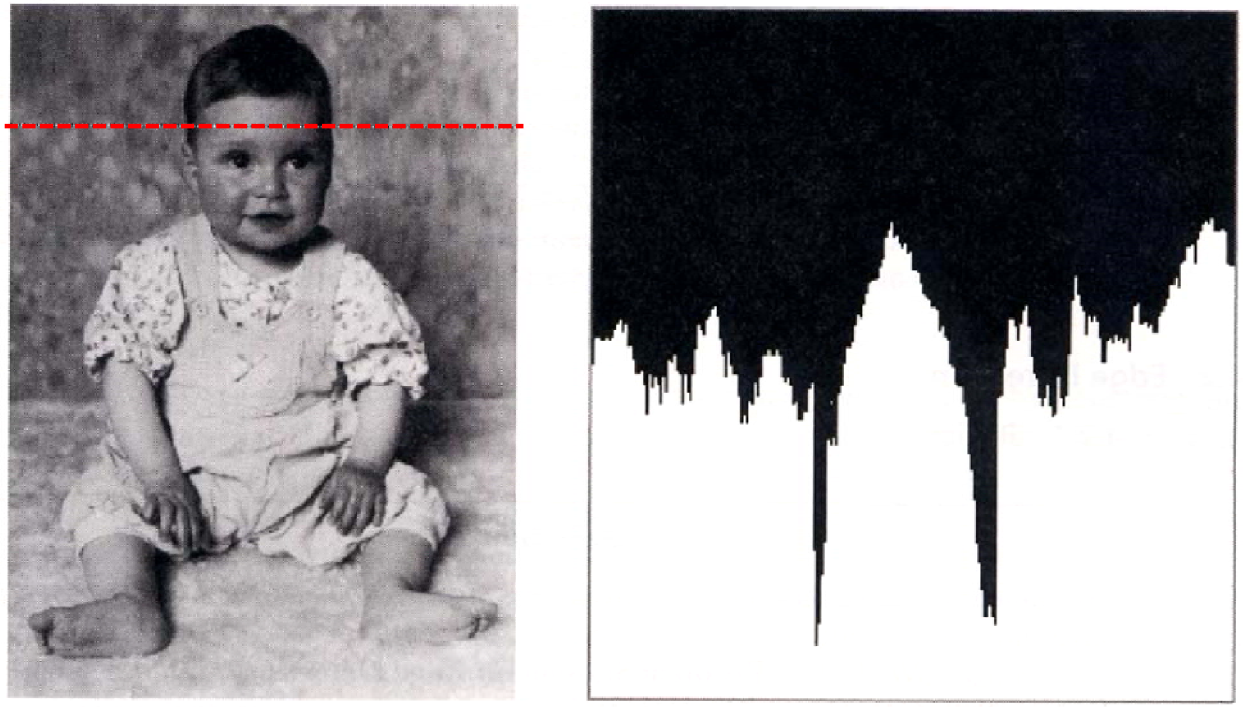
- Global features – global properties of an image, including intensity histogram, frequency domain descriptors, covariance matrix and high order statistics, etc
- Local features – local regions with special properties, including edges, corners, lines, curves, regions with special properties, etc
- Depending on applications, various features are useful. We will focus on edges, corners and so called Interest Points within SIFT



Edges

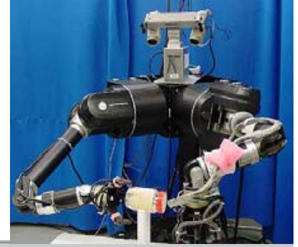


- Edge points are pixels at or around which the image values undergo a sharp variation – pixels with large gradient



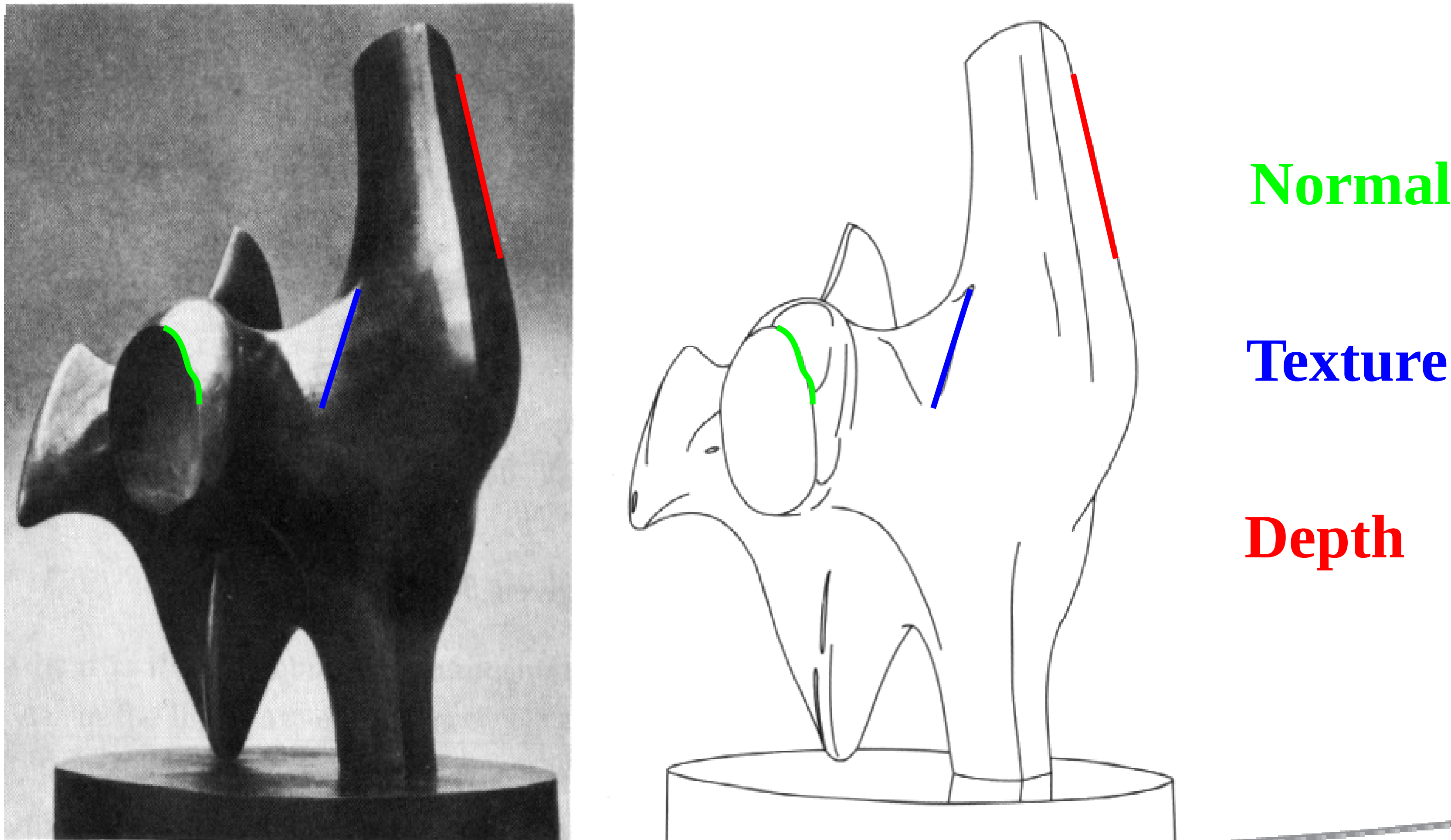
(a) (b)

Figure 4.1 (a) A 325 × 237-pixel image, with scanline $i = 56$ highlighted. (b) The intensity profile along the highlighted scanline. Notice how the main intensity variations indicate the borders of the hair region along the scanline.



Edges

- Colloquially edges describe the boundary of a surface or a significant change in orientation



Edge Detection Example

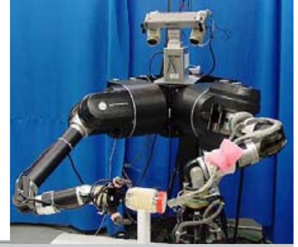
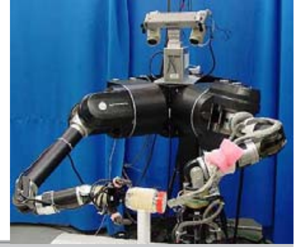


Figure 4.6 Output of HYSTERESIS_THRESH run on Figure 4.5, showing the effect of varying the filter's size. Left to right: $\sigma_f = 1, 2, 3$ pixel. The grey levels has been inverted (black on white) for clarity.

Building a Panorama

M. Brown and D. G. Lowe. Recognising Panoramas. ICCV 2003

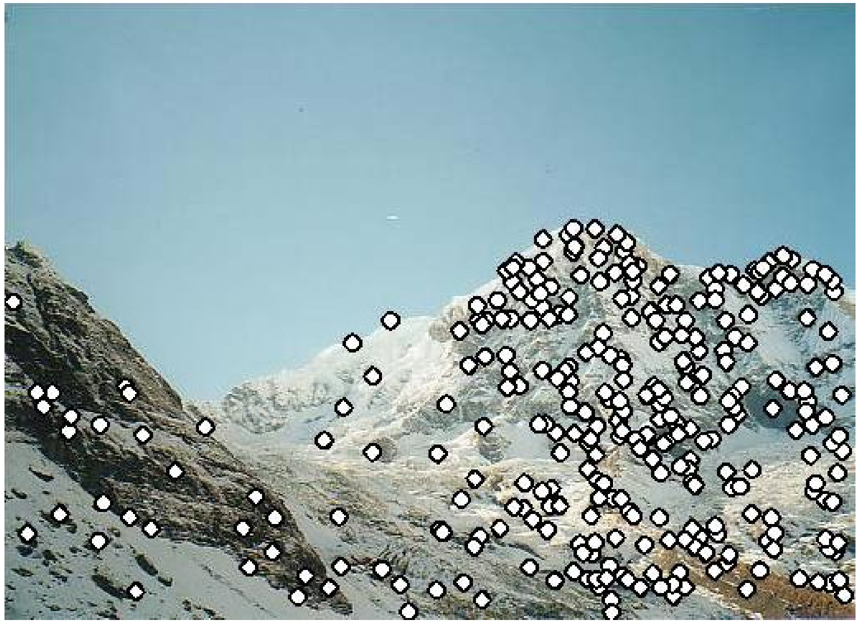
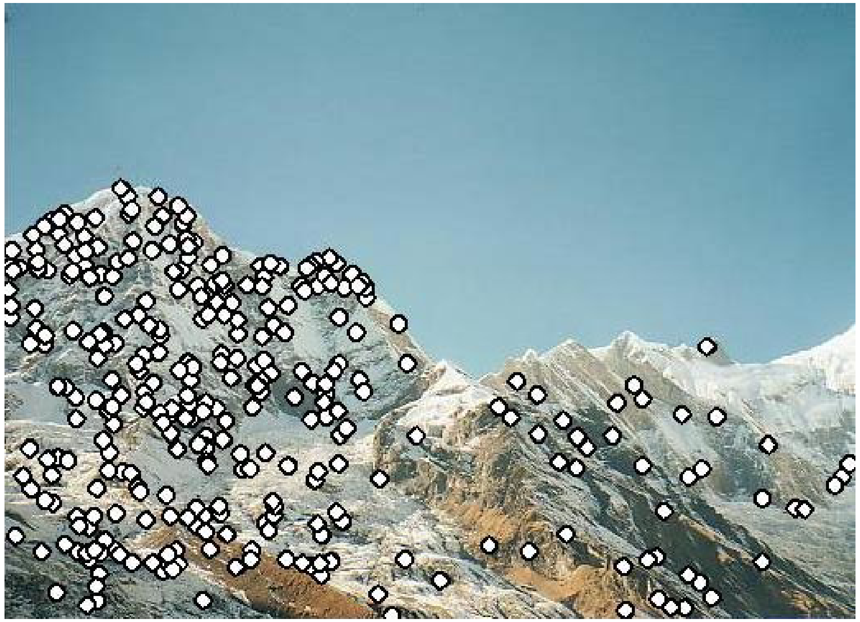


How do we build a Panorama?

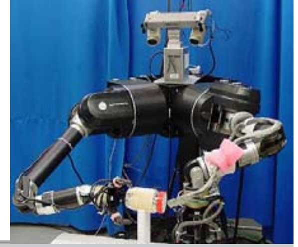
- We need to match (align) images
- Global methods sensitive to occlusion, lighting, parallax effects. So look for local features that match well.
- How would you do it by eye?

Matching with Features

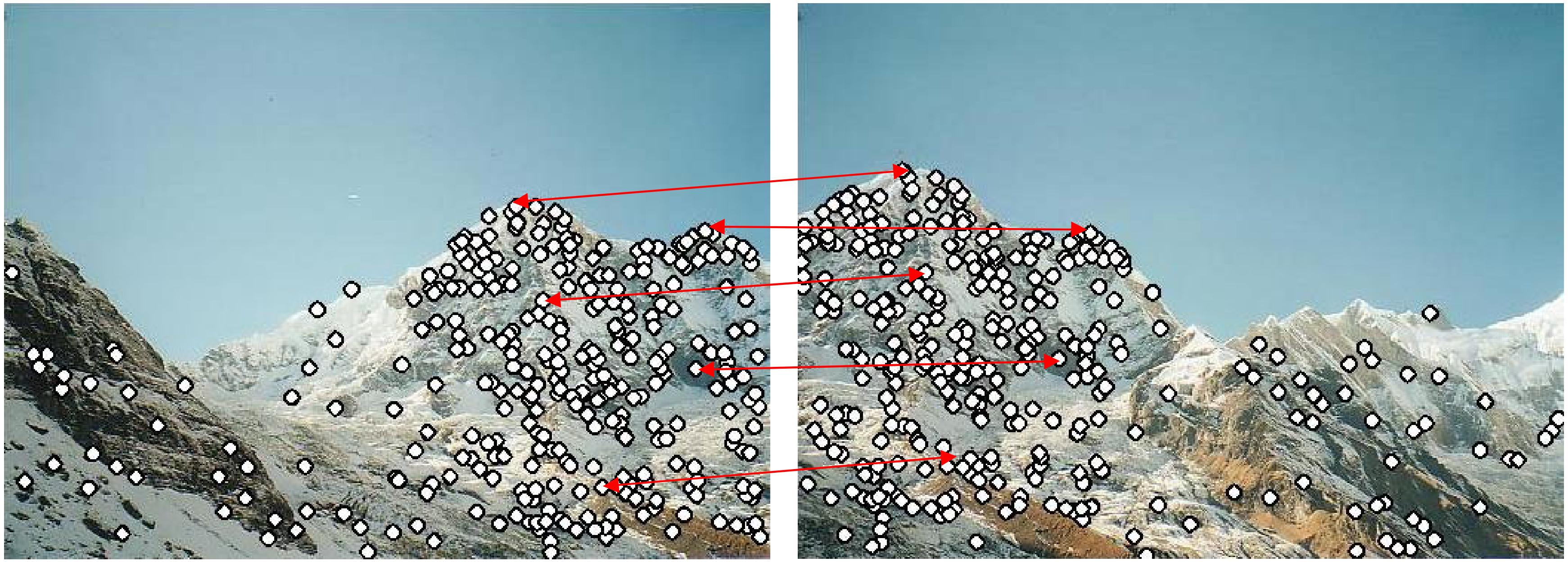
- Detect feature points in both images

Matching with Features



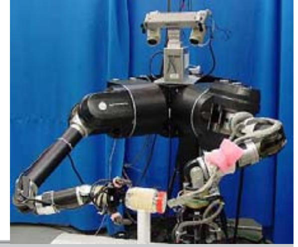
- Detect feature points in both images
- Find corresponding pairs



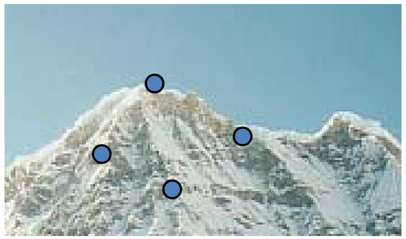
Matching with Features

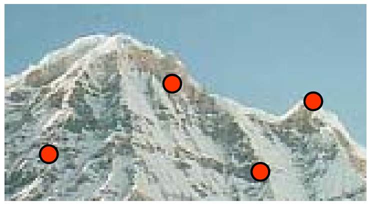
- Detect feature points in both images
- Find corresponding pairs
- Use these pairs to align images

Matching with Features



- Problem 1:
  - Detect the *same* point *independently* in both images



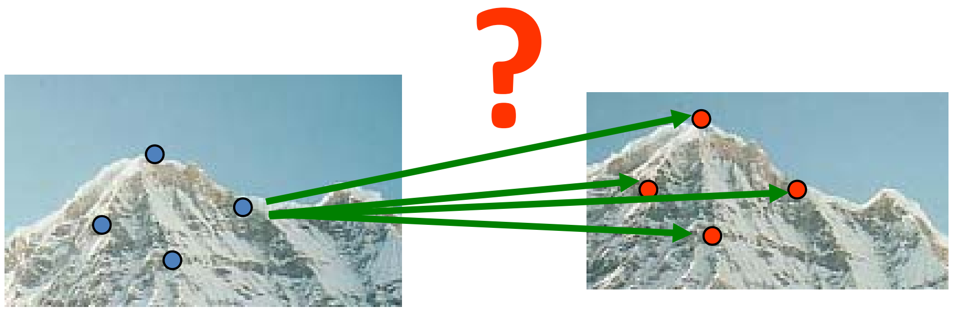


no chance to match!

We need a repeatable detector

Matching with Features

- Problem 2:
  - For each point correctly recognize the corresponding one

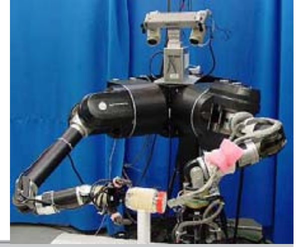


We need a reliable and distinctive descriptor

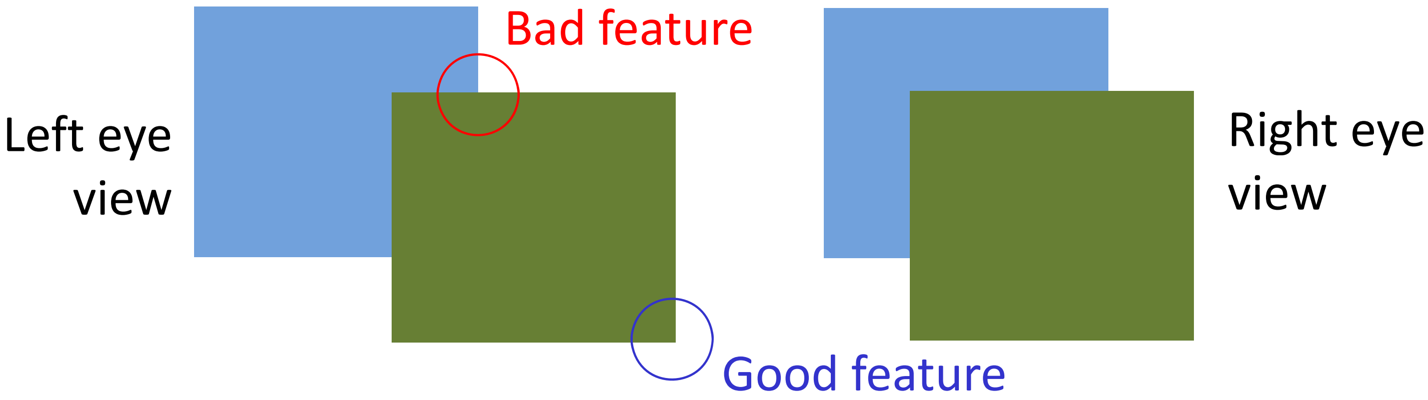
More Motivation…

- Feature points are used also for:
  - Image alignment (homography, fundamental matrix)
  - 3D reconstruction
  - Motion tracking
  - Object recognition
  - Indexing and database retrieval
  - Robot navigation
  - … other

Selecting Good Features

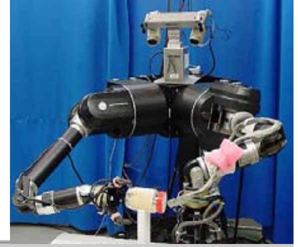


- What's a "good feature"?
  - Satisfies brightness constancy—looks the same in both images
  - Has sufficient texture variation
  - Does not have too much texture variation
  - Corresponds to a "real" surface patch—see below:



  - Does not deform too much over time

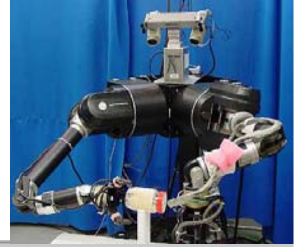
Interest Points



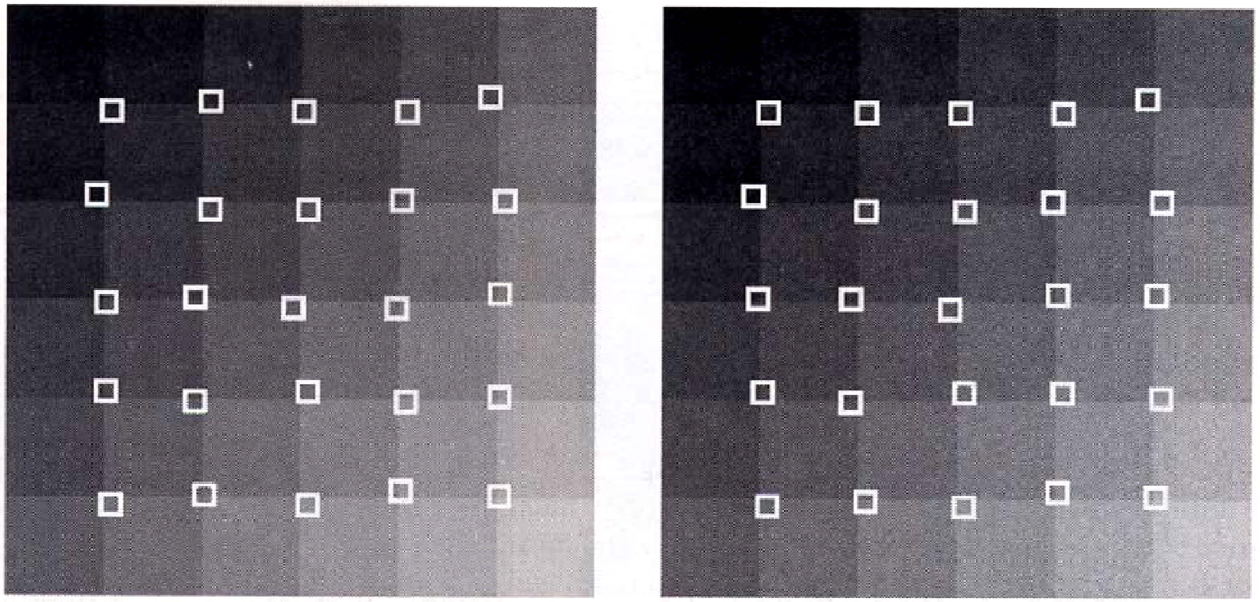
An **interest point** is a point in the image which in general can be characterized as follows:

1. it has a clear, preferably **mathematically well-founded, definition**,
2. it has a **well-defined position** in image space,
3. the **local image structure** around the interest point **is rich** in terms of local information contents, such that the use of interest points simplify further processing in the vision system,
4. it is **stable** under **local and global perturbations** in the image domain, including deformations as those arising from perspective transformations (sometimes reduced to affine transformations, scale changes, rotations and/or translations) as well as illumination/brightness variations,
5. The notion of interest point should include an **attribute of scale**.

Corner Features



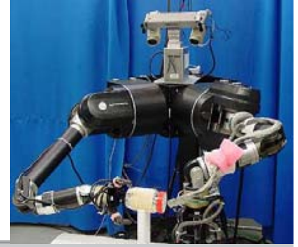
- Sources: intersection of image lines, corner patterns in the images, etc
- Stable between and across sequences of images



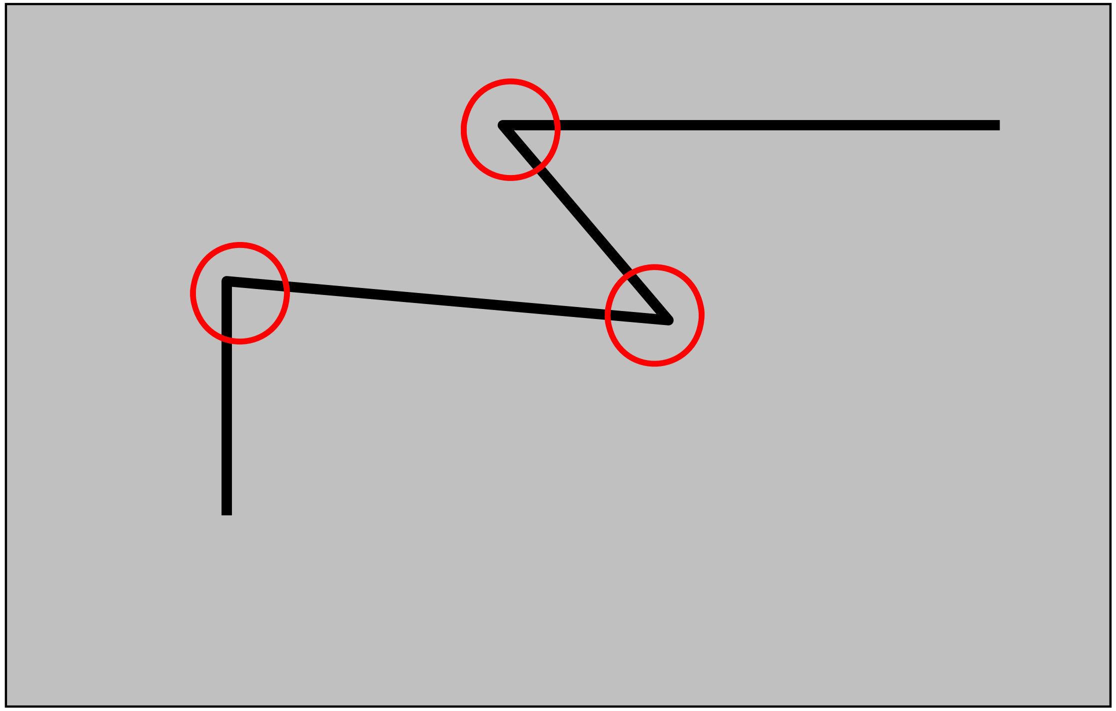
(a) (b)

Figure 4.8 Corners found in a 8-bit, synthetic checkerboard image, corrupted by two realizations of synthetic Gaussian noise of standard deviation 2. The corner is the bottom right point of each 15 × 15 neighbourhood (highlighted).

An introductory example:



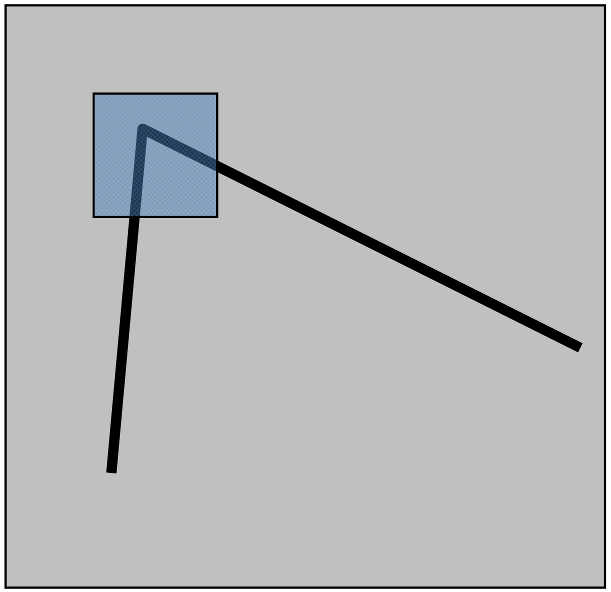
- Corner detector



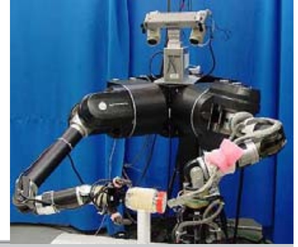
H. Moravec. Obstacle avoidance and navigation in the real world by a seeing robot rover. Technical Report CMU-RI-TR-3, Carnegie-Mellon University, Robotics Institute, 1980.

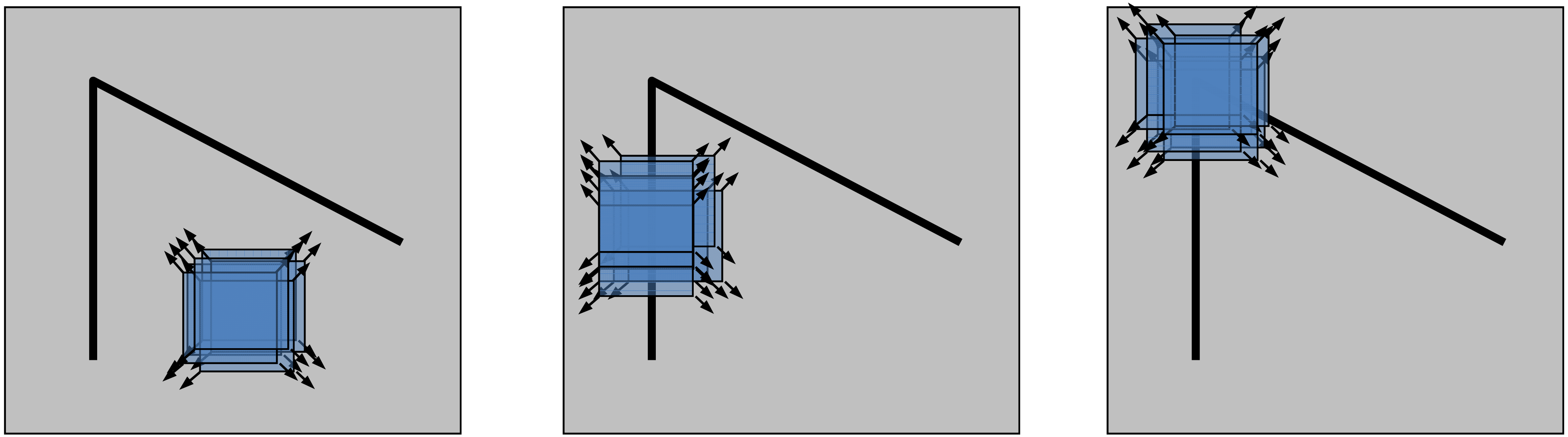
The Basic Idea

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



Corner Detector: Basic Idea



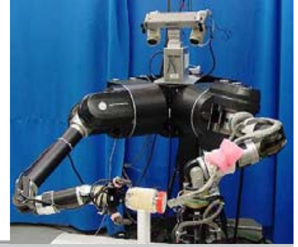


"flat" region:
no change as shift window in all directions

"edge":
no change as shift window along the edge direction

"corner":
significant change as shift window in all directions

Moravec Corner Detector



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

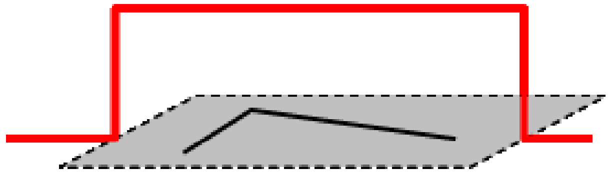
$$E(u,v)=\sum_{x,y} w(x,y)\left[I(x+u,y+v)-I(x,y)\right]^2$$

Window function → $w(x,y)$

Shifted intensity → $I(x+u,y+v)$

Intensity → $I(x,y)$

Window function $w(x,y)$ =

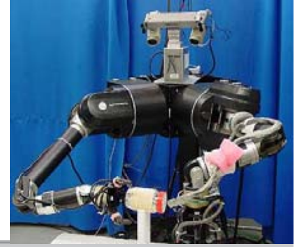


1 in window, 0 outside

Four shifts: (u,v) = (1,0), (1,1), (0,1), (-1, 1)

Look for local maxima in min{E}

Problems of Moravec Detector

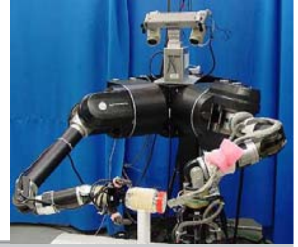


- Noisy response due to a binary window function
- Only a set of shifts at every 45 degree is considered
- Responds too strong for edges because only minimum of E is taken into account

⇨ Harris corner detector (1988) solves these problems.

C. Harris, M. Stephens. "A Combined Corner and Edge Detector",
Alvey Vision Conference, pp.147-151, 1988

Harris Corner Detector

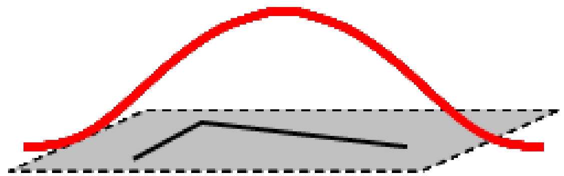


Noisy response due to a binary window function

➢ Use a Gaussian function

$$w(x,y) = \exp\left(-\frac{(x^2+y^2)}{2\sigma^2}\right)$$

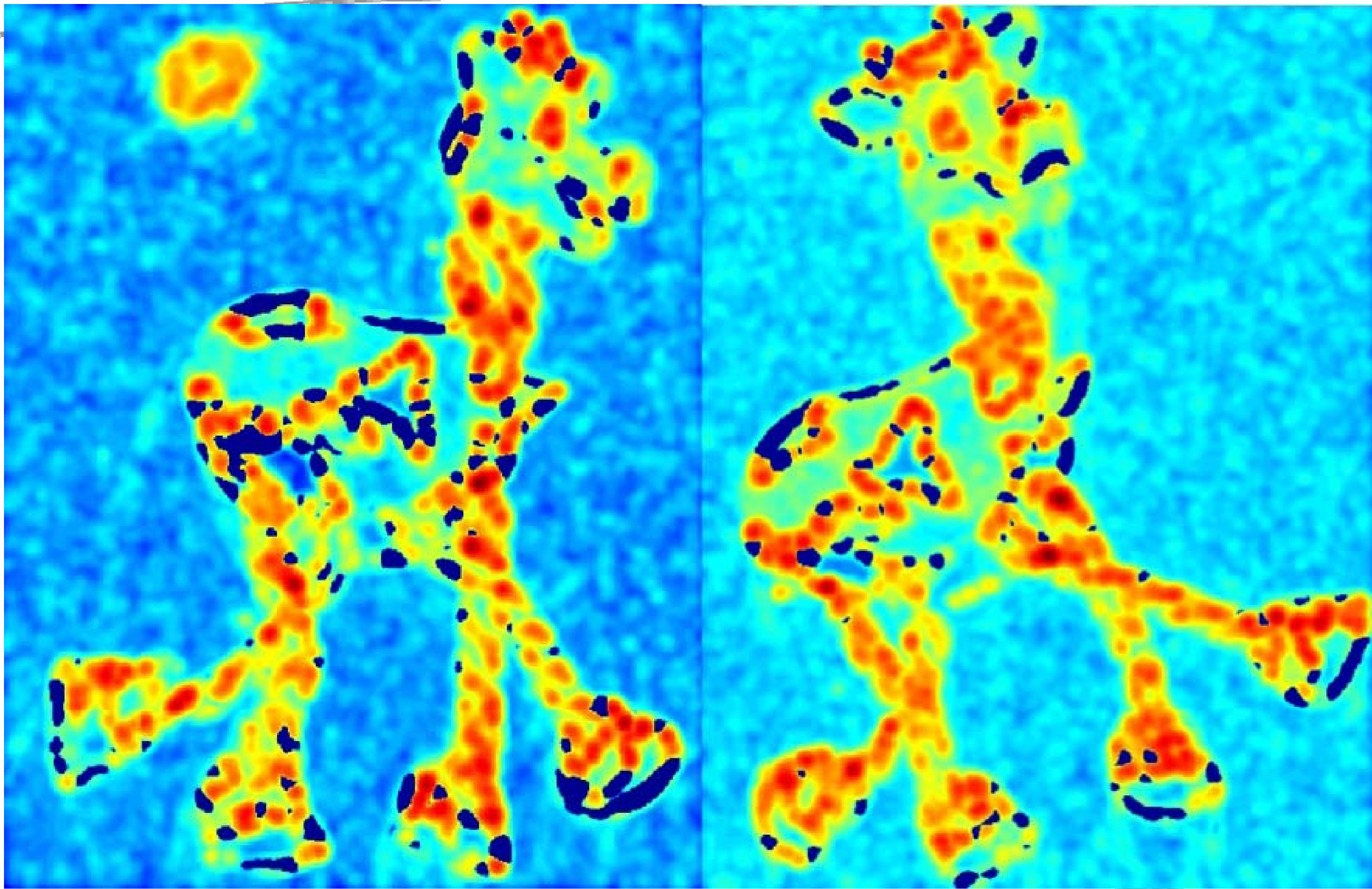
Window function $w(x,y)$ =



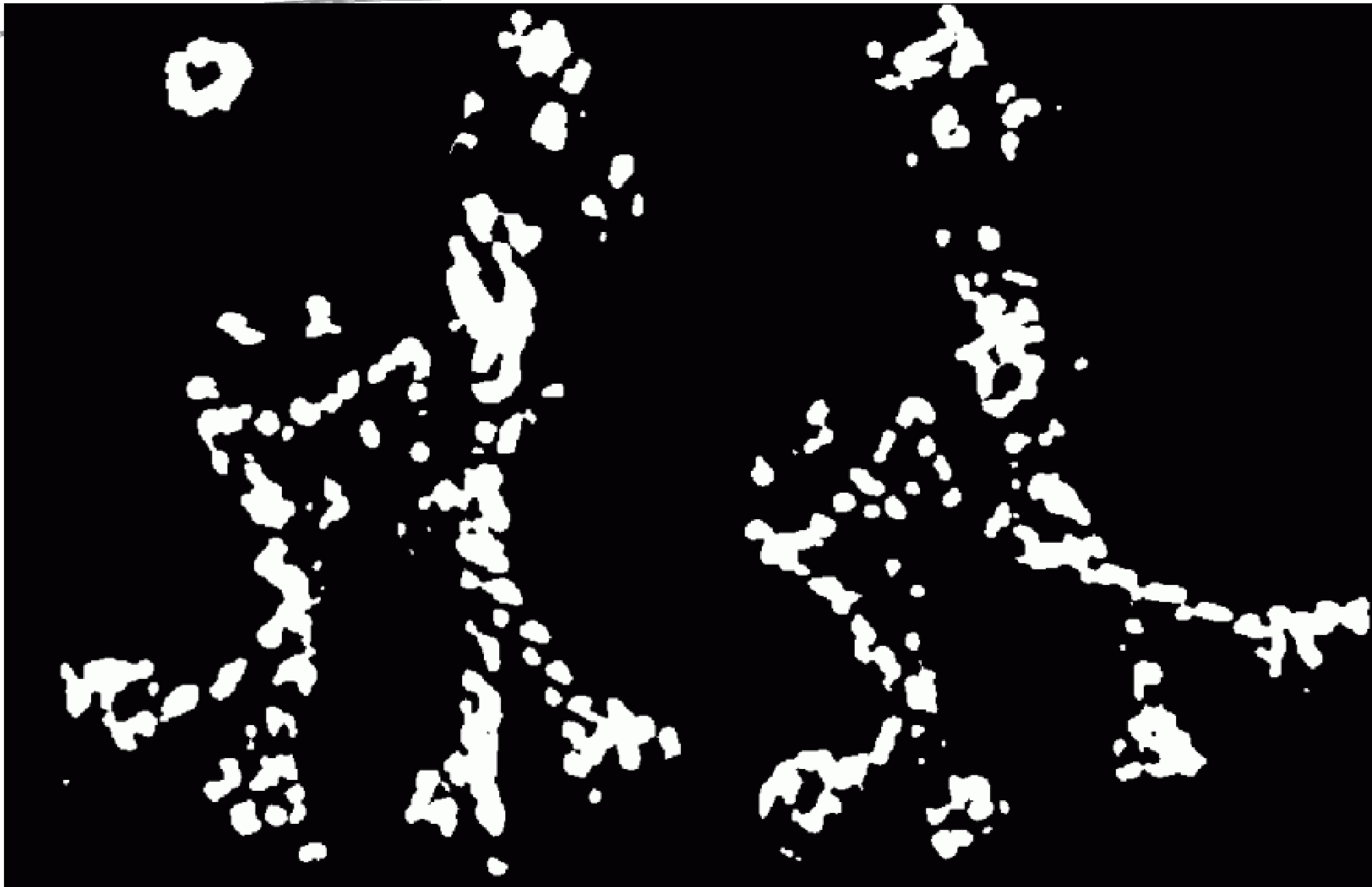
Gaussian

Harris Corner Detector (input)

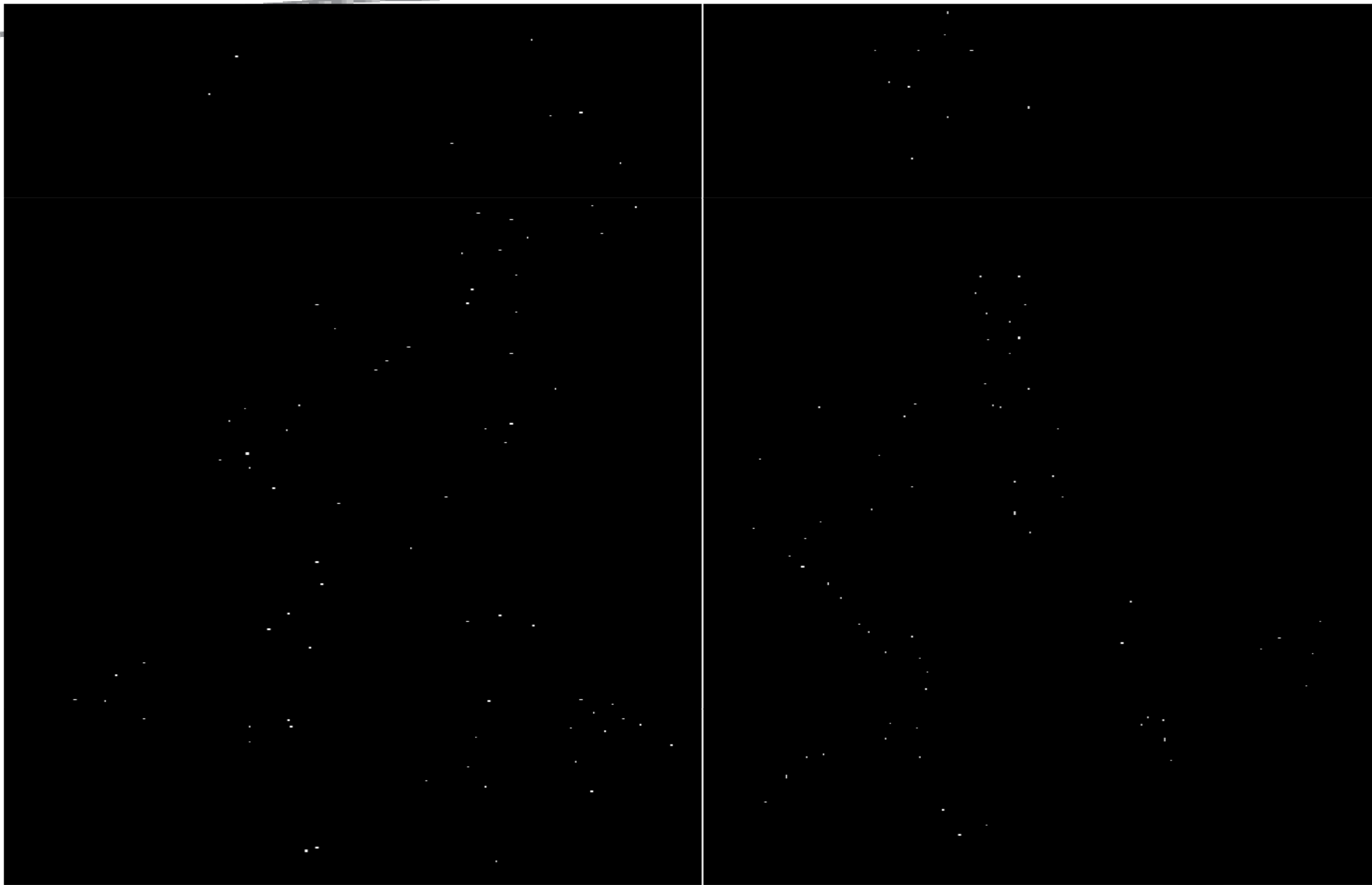
Corner Response R



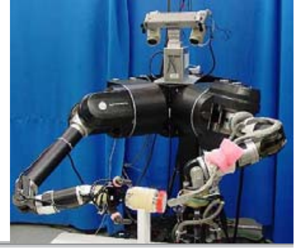
Threshold on R

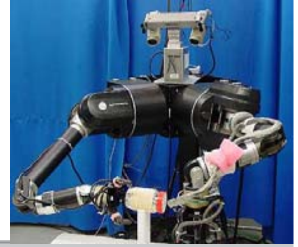


Local Maximum of R



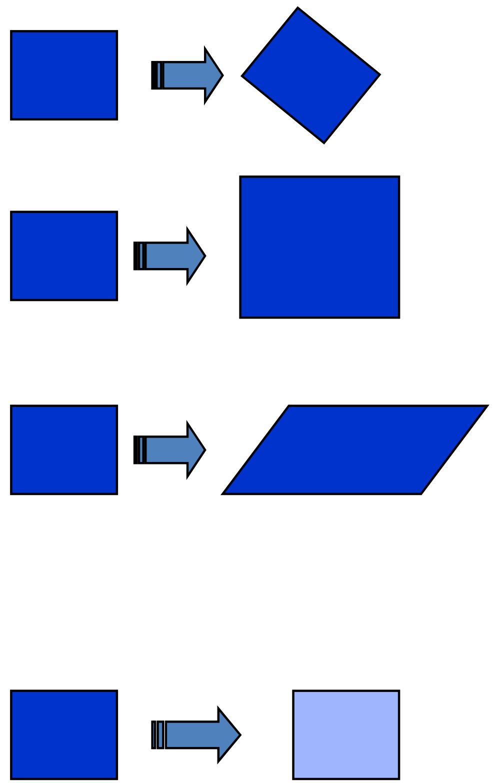
Harris Corner Detector



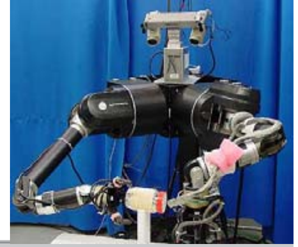


Models of Image Change

- Geometry
  - Rotation
  - Similarity (rotation + uniform scale)
  - Affine (scale dependent on direction)
    valid for: orthographic camera, locally planar object
- Photometry
  - Affine intensity change ($I \rightarrow a\,I + b$)



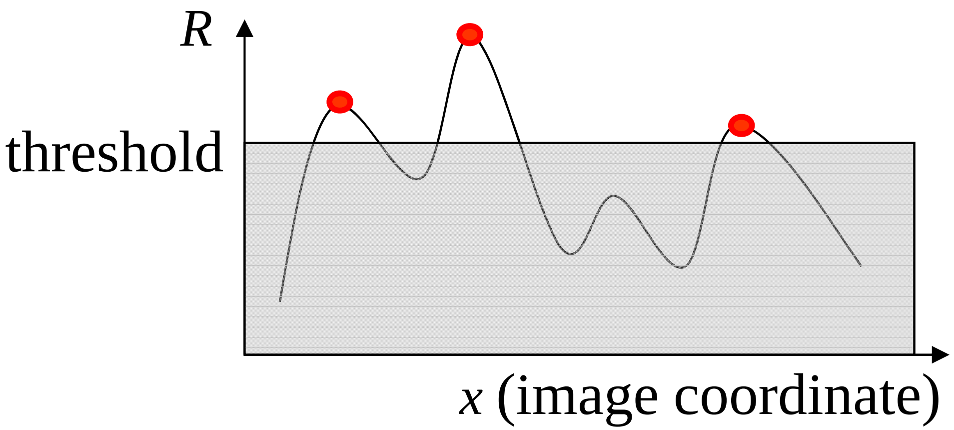
Harris Detector: Some Properties

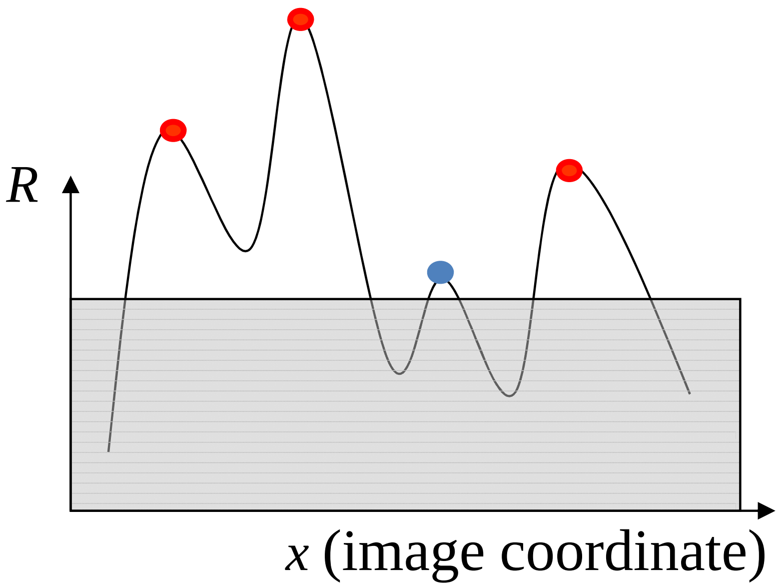


- Partial invariance to *affine intensity* change

  ✓ Only derivatives are used => invariance to intensity shift $I \rightarrow I + b$

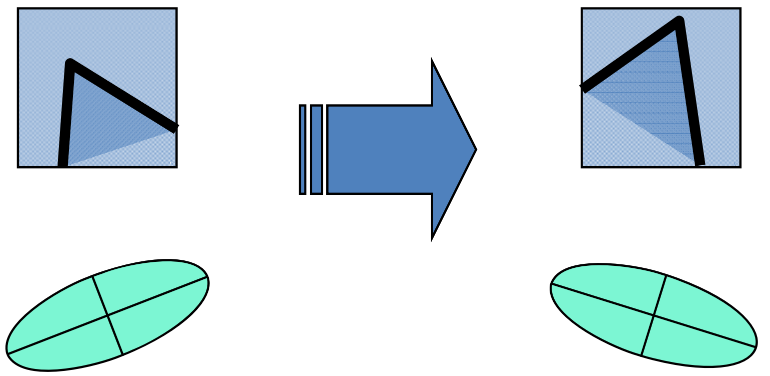
  ✓ Intensity scale: $I \rightarrow a\,I$





Harris Detector: Some Properties

- Rotation invariance

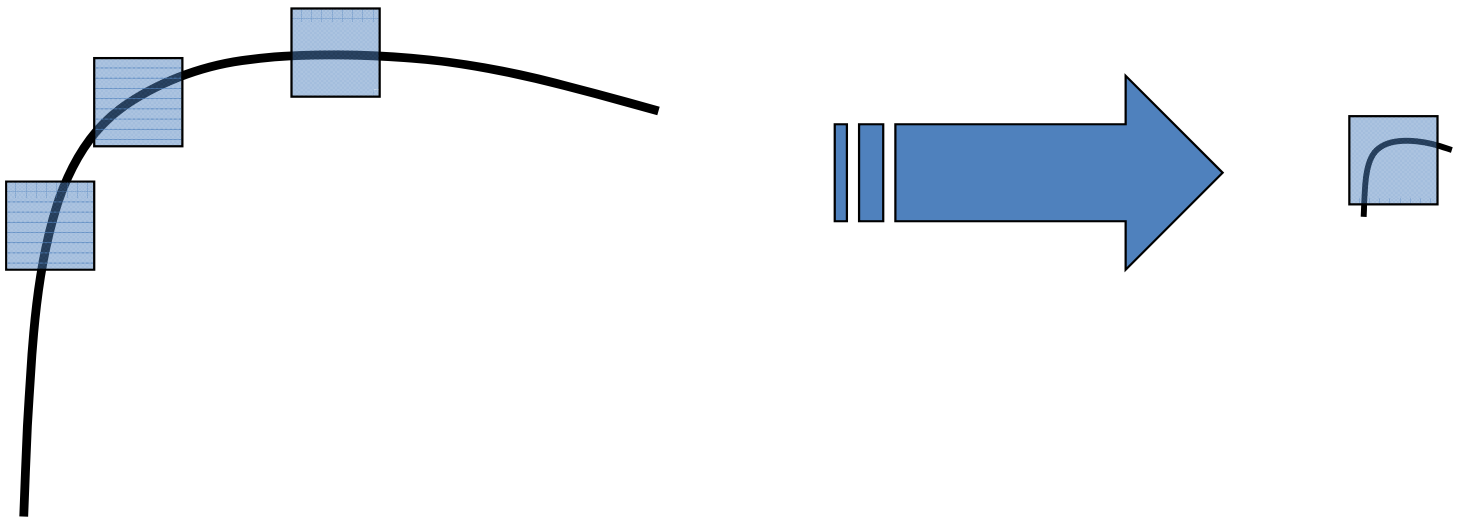


Ellipse rotates but its shape (i.e. eigenvalues) remains the same

Corner response R is invariant to image rotation

Harris Detector: Some Properties

- But: non-invariant to image scale!

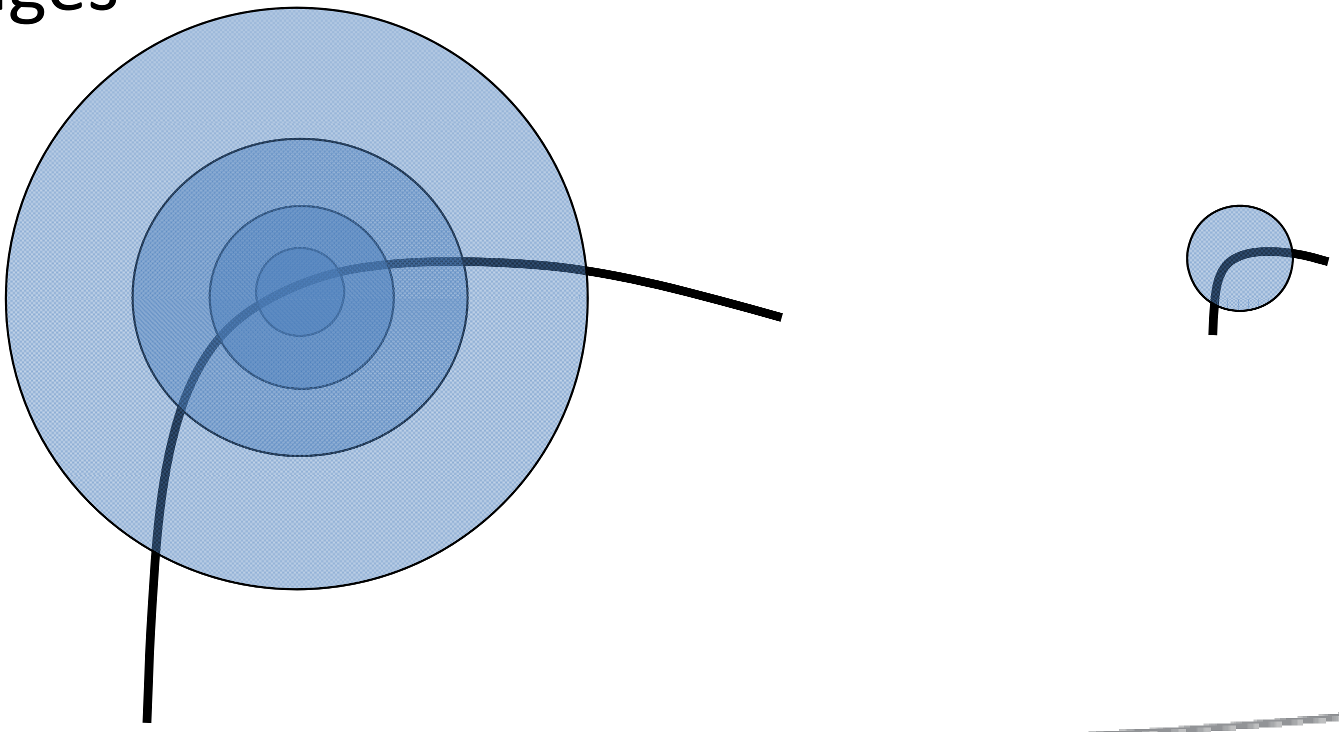


All points will be classified as edges

Corner !

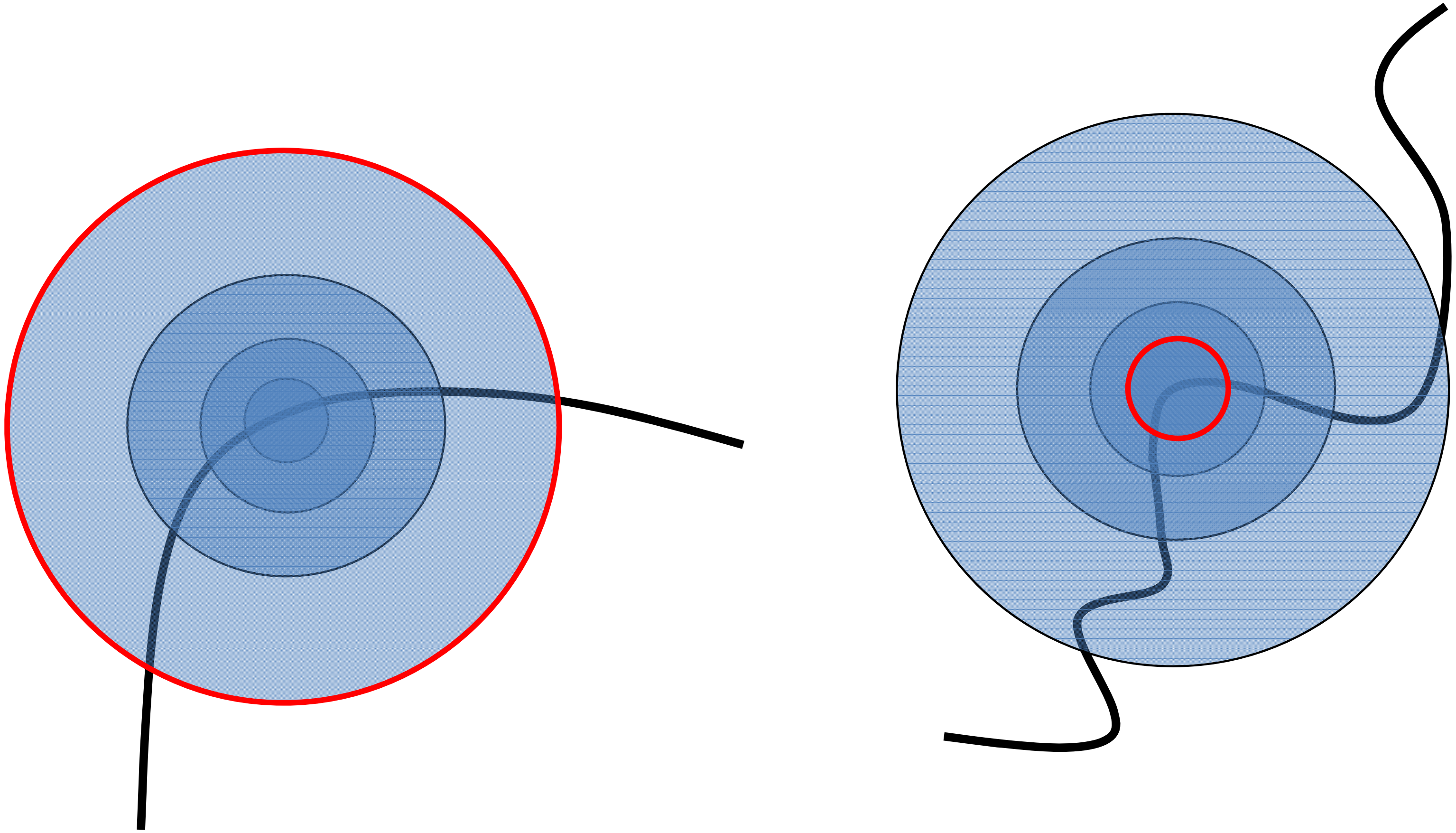
Scale Invariant Detection

- Consider regions (e.g. circles) of different sizes around a point
- Regions of corresponding sizes will look the same in both images

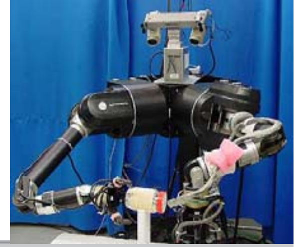


Scale Invariant Detection

- The problem: how do we choose corresponding circles **independently** in each image?



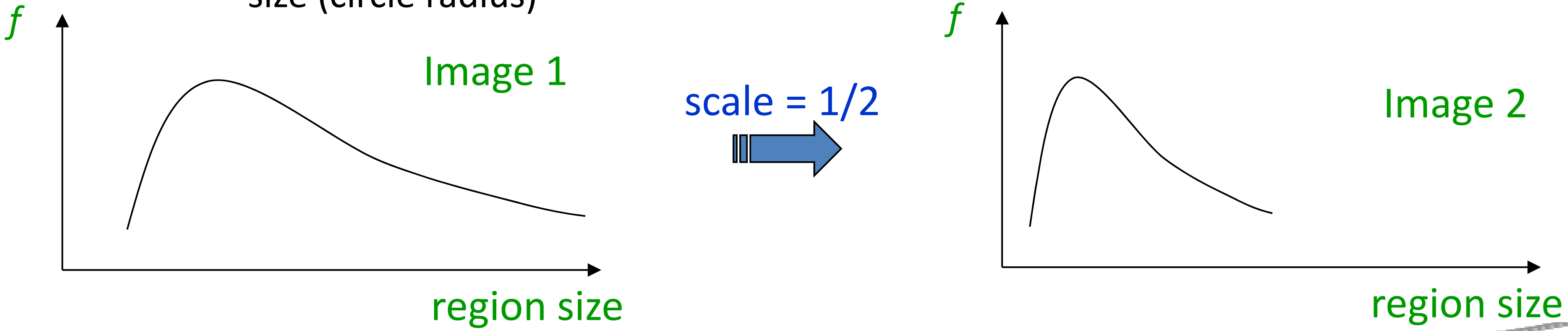
Scale Invariant Detection



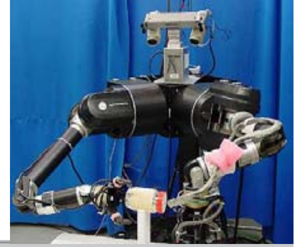
- Solution:
  - Design a function on the region (circle), which is “scale invariant” (the same for corresponding regions, even if they are at different scales)

Example: average intensity. For corresponding regions (even of different sizes) it will be the same.

– For a point in one image, we can consider it as a function of region size (circle radius)



Scale Invariant Detection

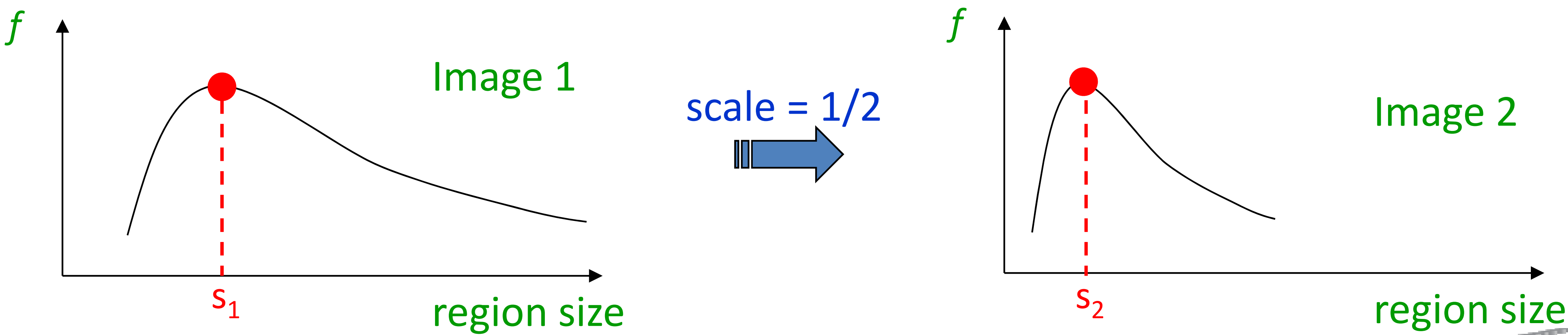


- Common approach:

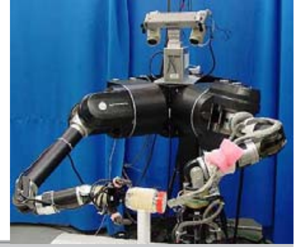
  Take a local maximum of this function

  Observation: region size, for which the maximum is achieved, should be *invariant* to image scale.

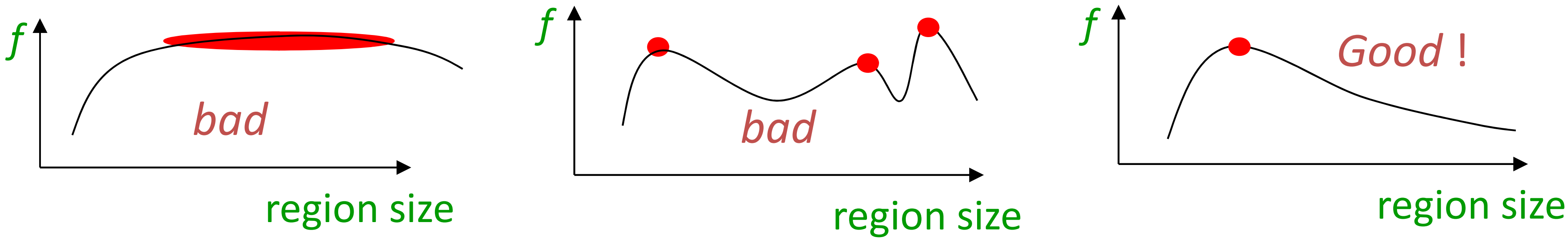
  Important: this scale invariant region size is found in each image independently!



Scale Invariant Detection

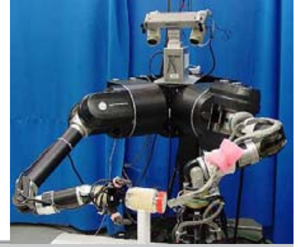


- A "good" function for scale detection: has one stable sharp peak



- For usual images: a good function would be a one which responds to contrast (sharp local intensity change)

Scale Invariant Detection



- Functions for determining scale

$$f = \text{Kernel} * \text{Image}$$

Kernels:

$$L = \sigma^2 \left(G_{xx}(x,y,\sigma) + G_{yy}(x,y,\sigma) \right)$$

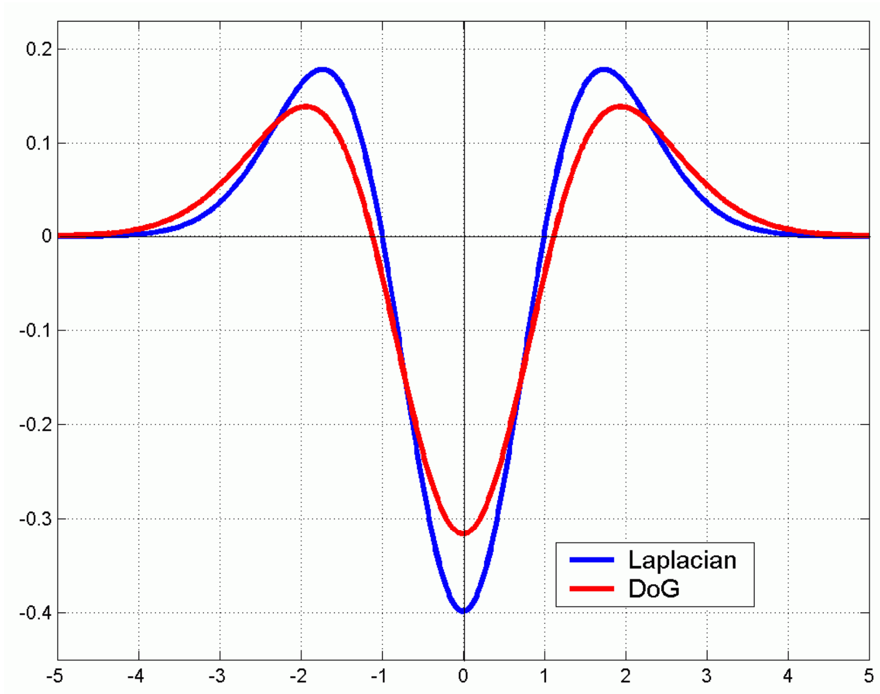
(Laplacian)

$$DoG = G(x,y,k\sigma) - G(x,y,\sigma)$$

(Difference of Gaussians)

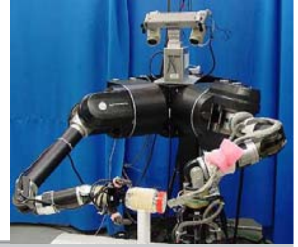
where Gaussian

$$G(x,y,\sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

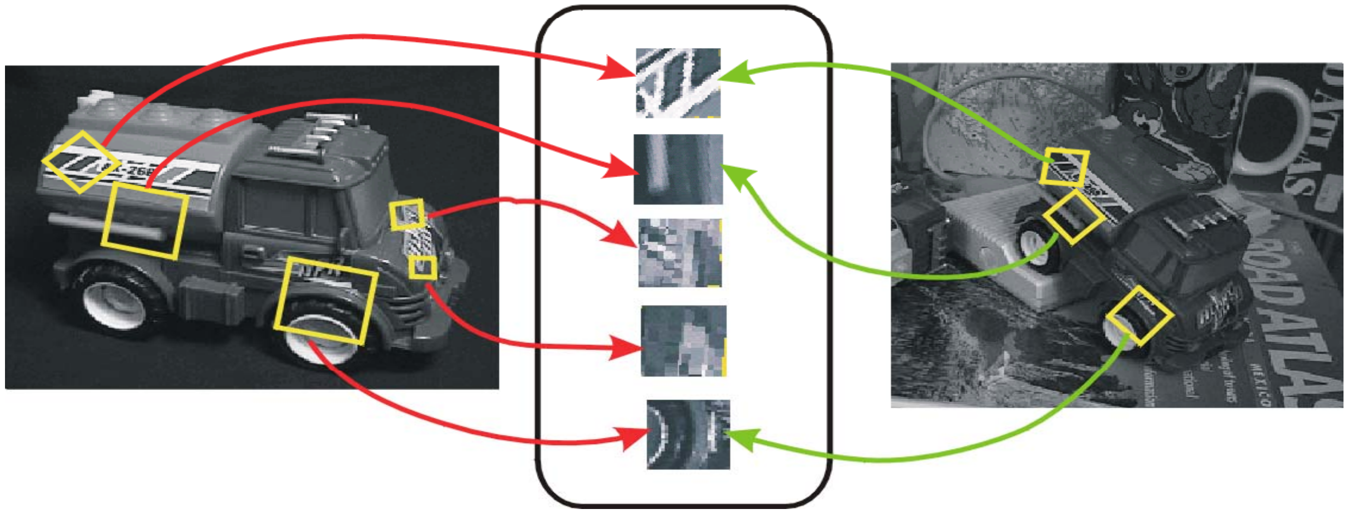


Note: both kernels are invariant to *scale* and *rotation*

SIFT (Scale Invariant Feature Transform)[1]

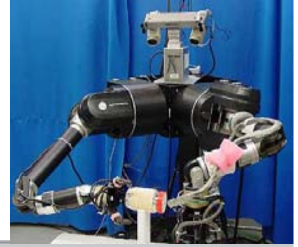


- SIFT is an carefully designed procedure with empirically determined parameters for the invariant and distinctive features
- Empirically found[2] to show very good performance, invariant to *image rotation*, *scale*, *intensity change*, and to moderate *affine* transformations
- Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters



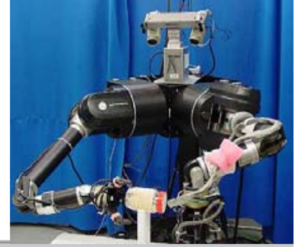
[1] D.Lowe. “Distinctive Image Features from Scale-Invariant Keypoints”. IJCV 2004
[2] K.Mikolajczyk, C.Schmid. “A Performance Evaluation of Local Descriptors”. CVPR 2003

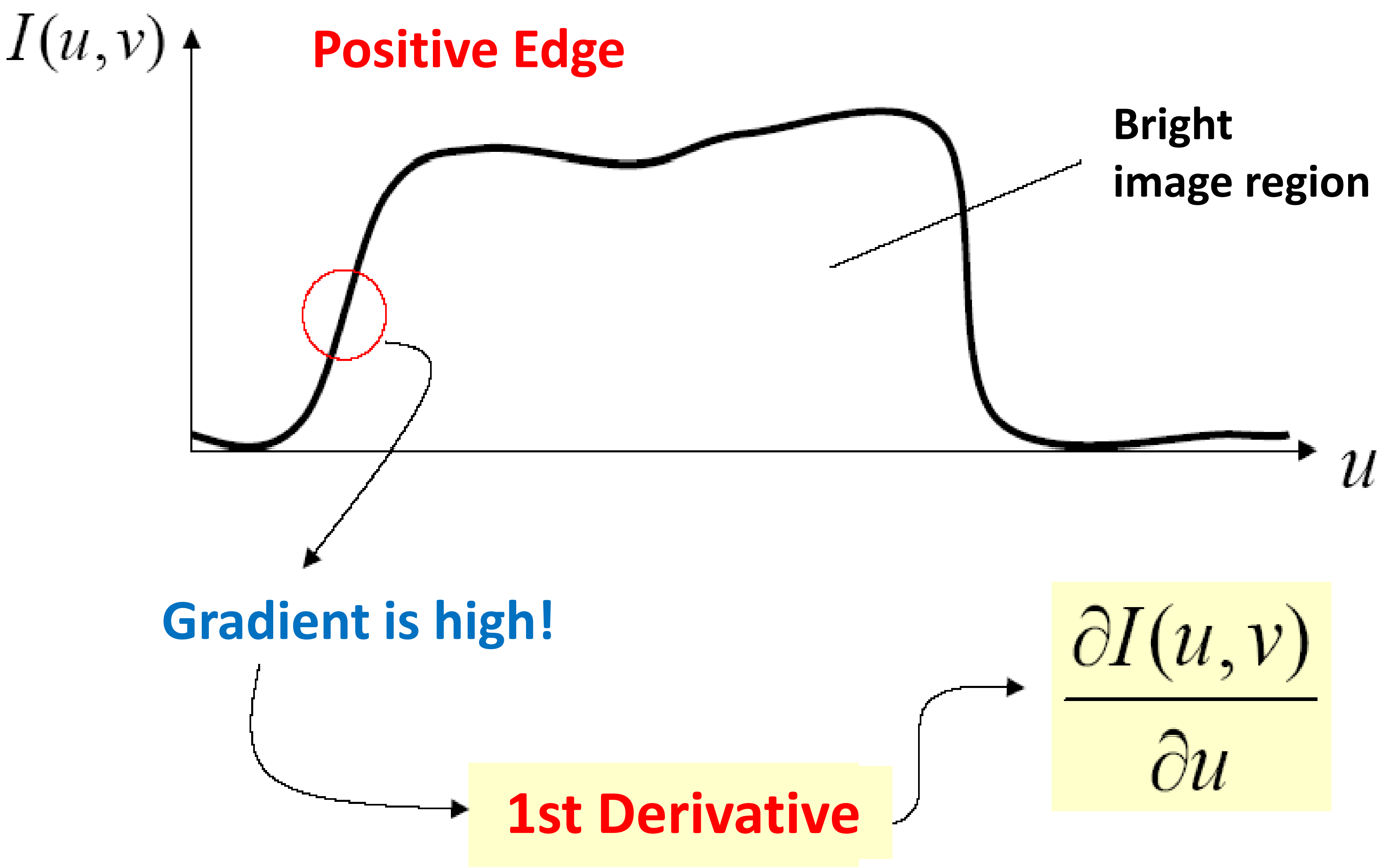
Introduction



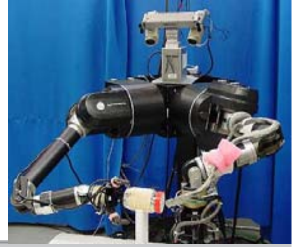
- SIFT in one sentence
  - Histogram of gradients @ Harris-corner-like
- Extract features
  - Find keypoints
    - Scale, Location
    - Orientation
  - Create signature
- Match features

What happens at an edge?





SIFT stages:

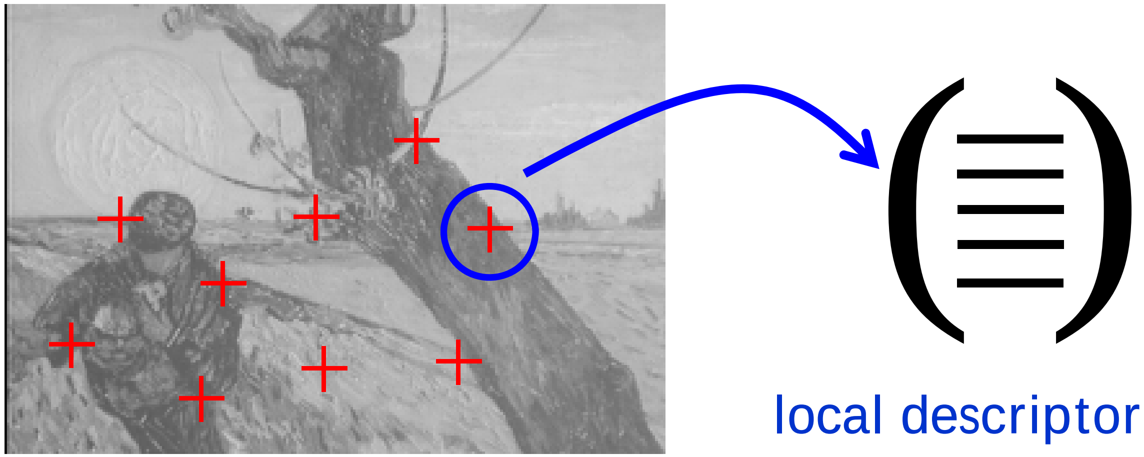


- Scale-space extrema detection
- Keypoint localization

detector

- Orientation assignment
- Keypoint descriptor

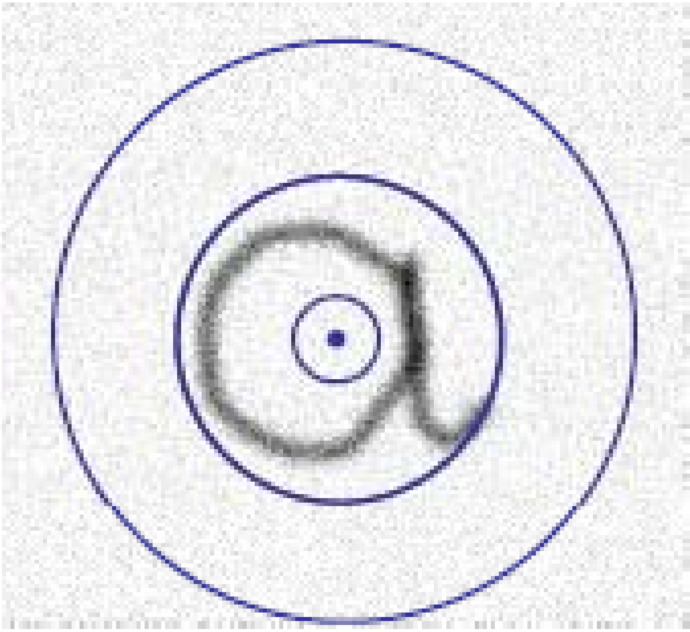
descriptor

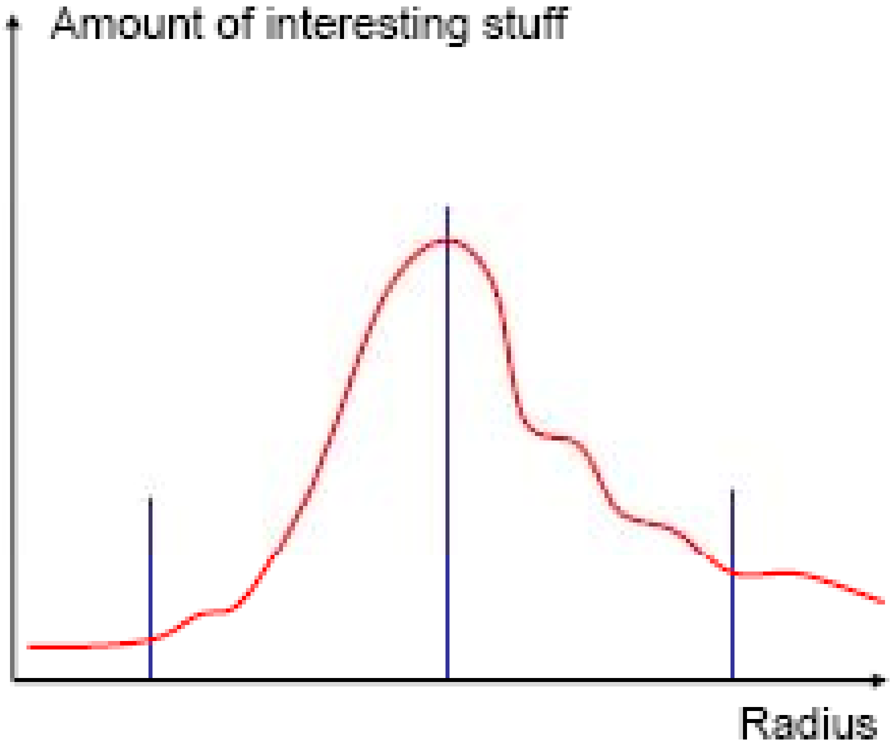


A 500x500 image gives about 2000 features

Finding Keypoints – Scale, Location

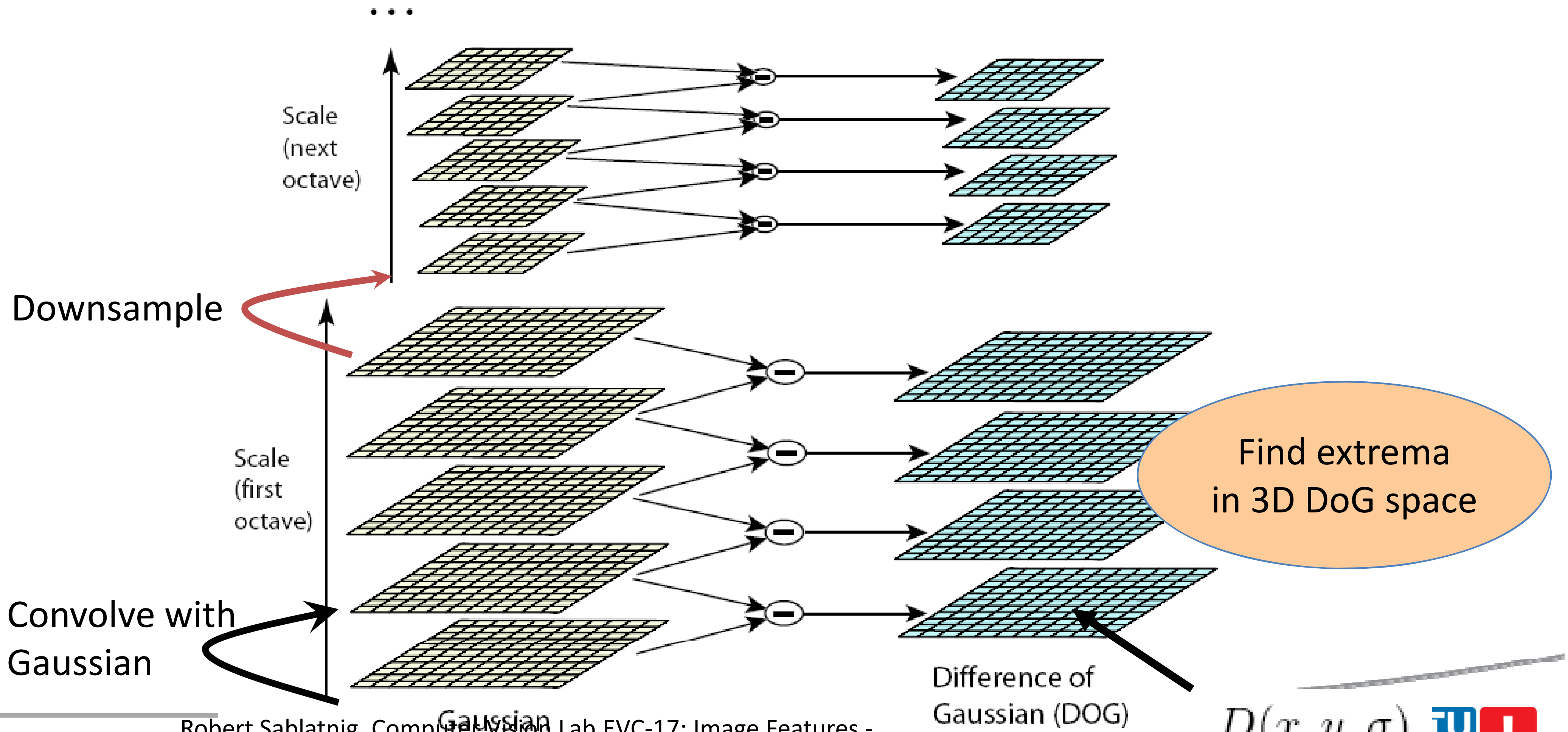
- How do we choose scale?





1. Finding Keypoints – Scale

- Scale selection principle (T. Lindeberg ’94)
  - ➔ Maxima/minima of Difference of Gaussian



$D(x, y, \sigma)$

DoG Filtering

- Convolution with a variable-scale Gaussian

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y),$$

$$G(x, y, \sigma) = 1/(2\pi\sigma^2) \exp^{-(x^2+y^2)/\sigma^2}$$

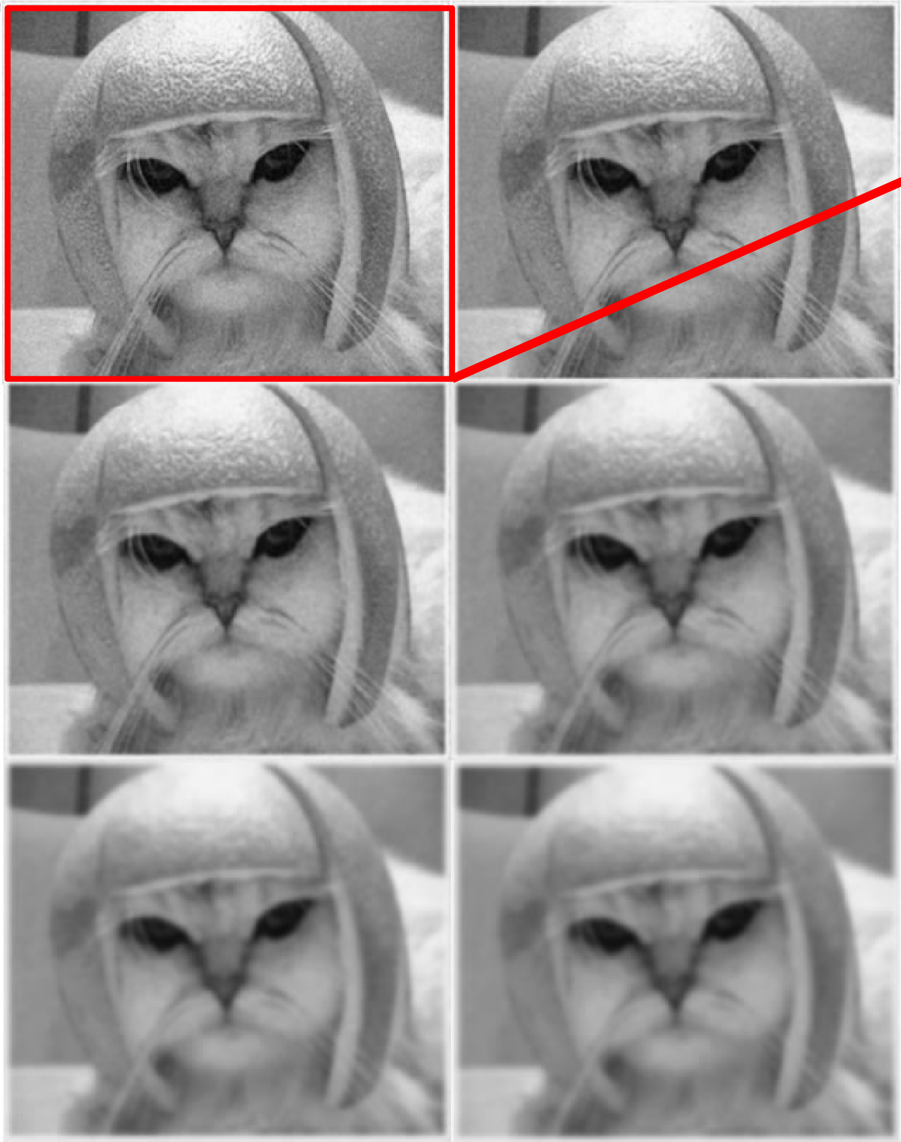
- Difference-of-Gaussian (DoG) filter

$$G(x, y, k\sigma) - G(x, y, \sigma)$$

- Convolution with the DoG filter

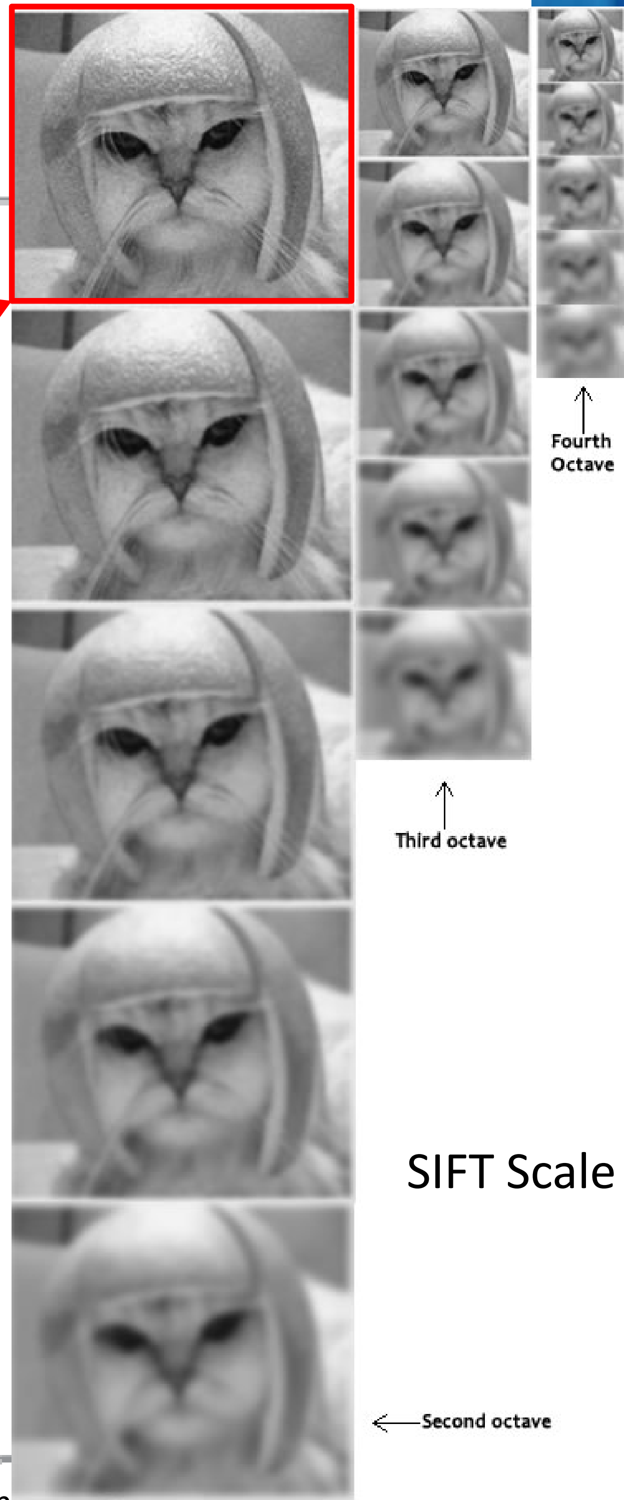
$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma)$$

Finding Keypoints – Scale



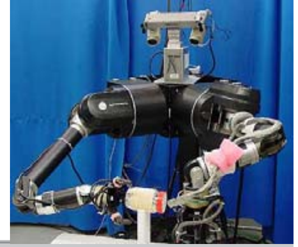
Scale Space

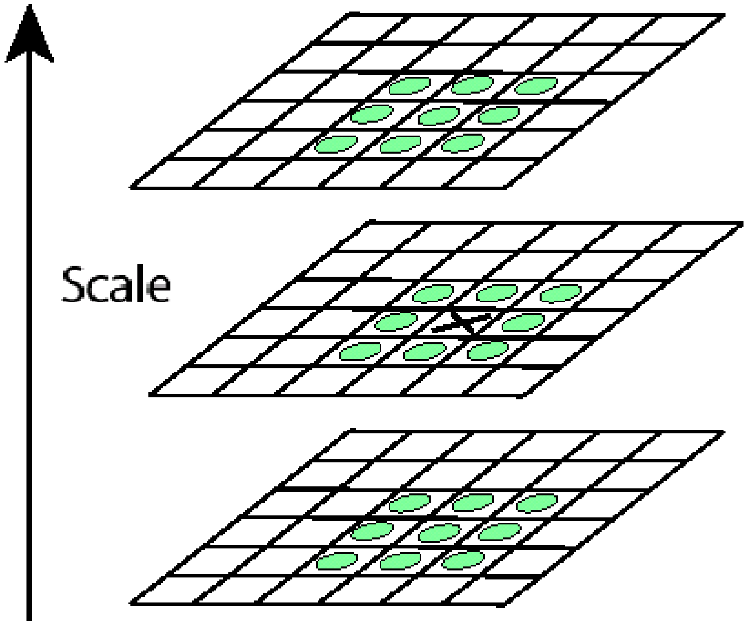
Reduce by 50%



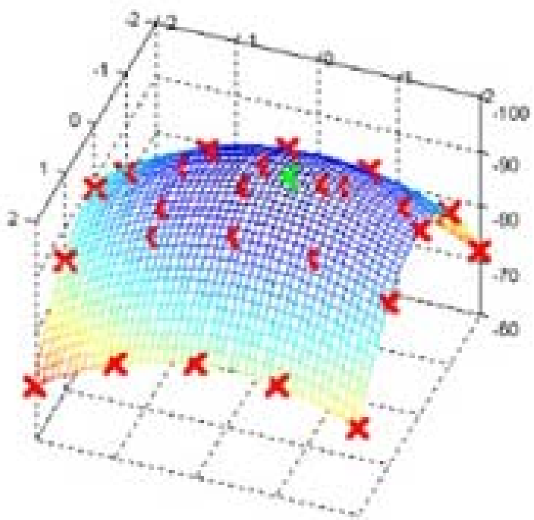
SIFT Scale Space

2. Finding Keypoints – Localization





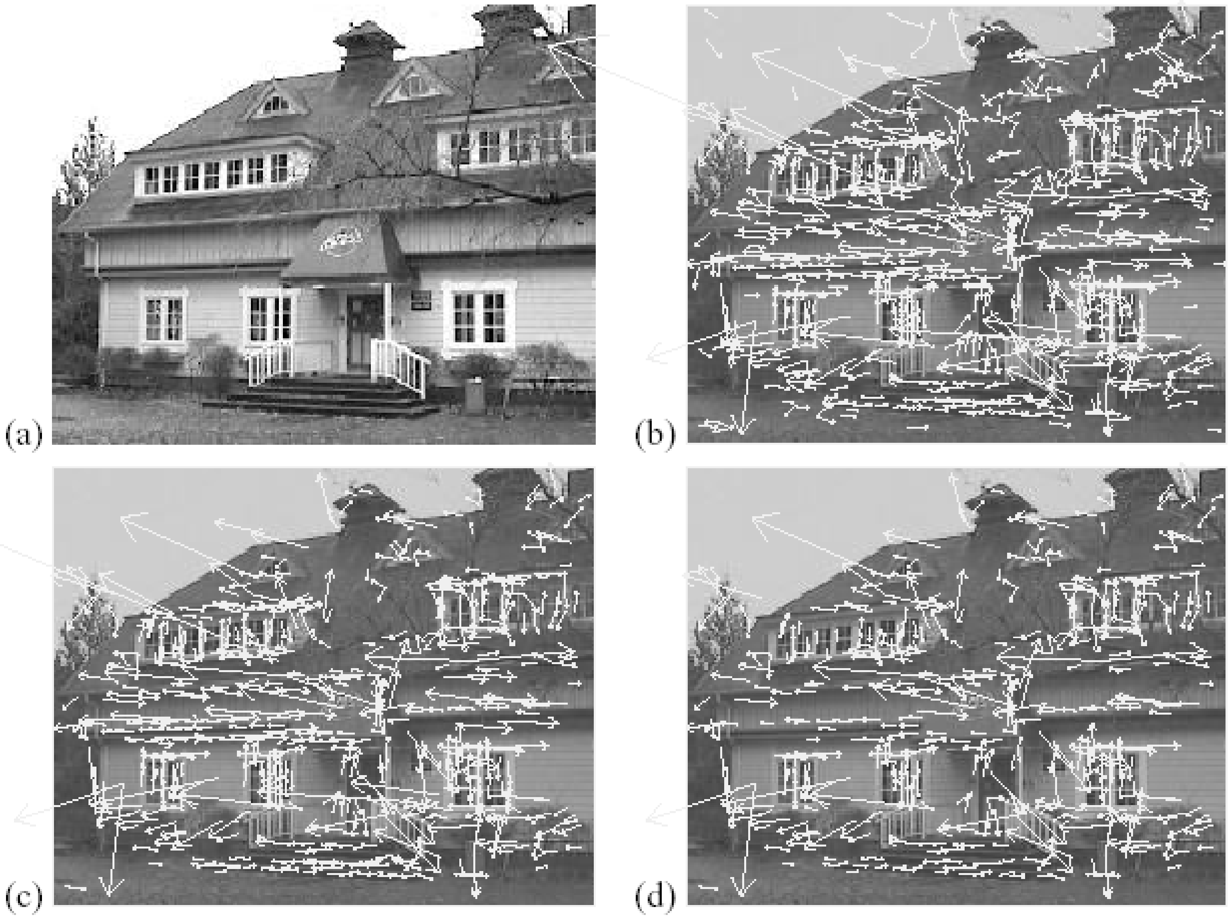
X is selected if it is larger or smaller than all 26 neighbors



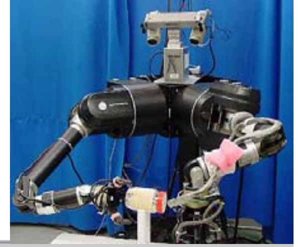
Using the available pixel data, subpixel values are generated. This is done by the Taylor expansion of the image around the approximate key point

Keypoint Localization

- Reject points with low contrast and poorly localized along an edge
- Fit a 3D quadratic function for sub-pixel maxima

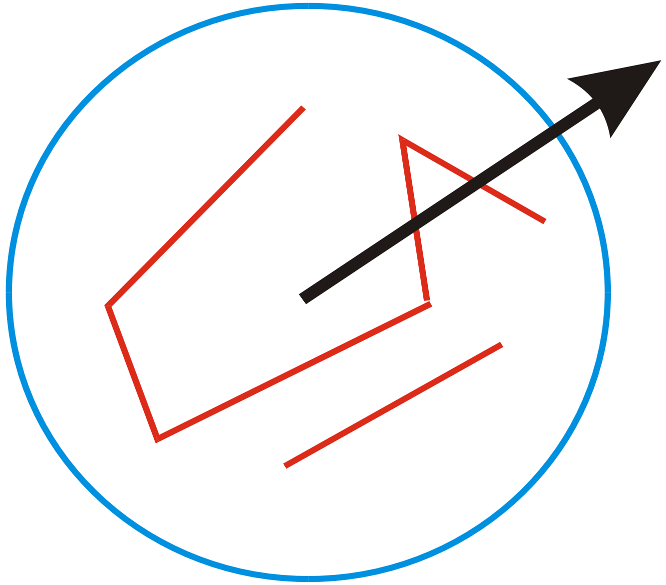


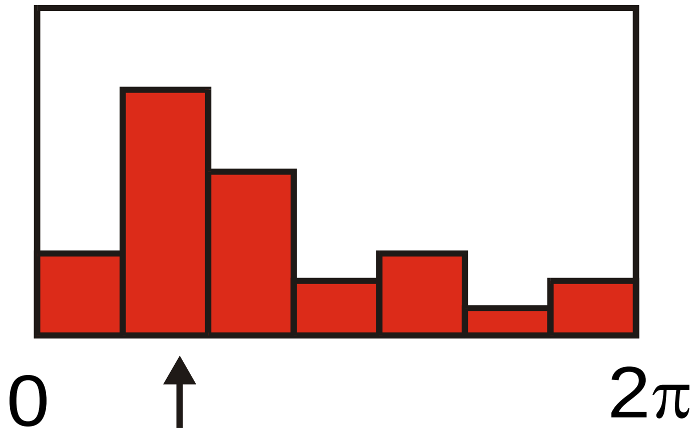
(a) 233x189 image
(b) 832 DOG extrema
(c) 729 left after peak value threshold
(d) 536 left after testing ratio of principle curvatures



3. Finding Keypoints – Orientation

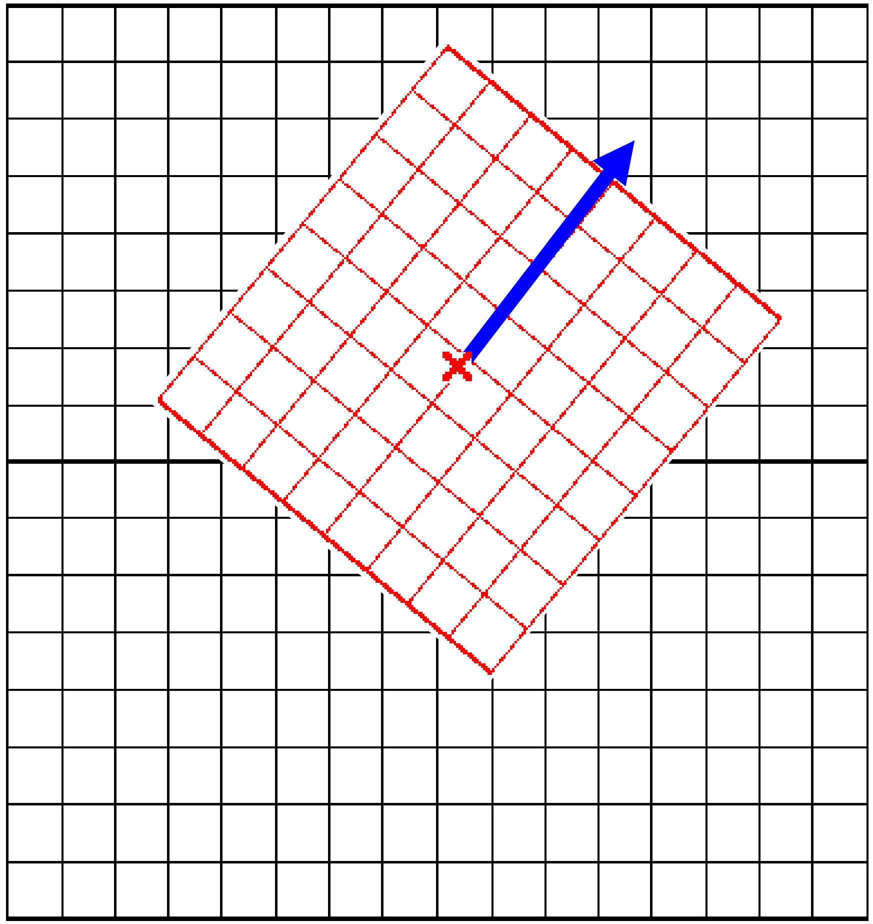
- Create histogram of local gradient directions computed at selected scale
- Assign canonical orientation at peak of smoothed histogram
- Each key specifies stable 2D coordinates (x, y, scale, orientation)



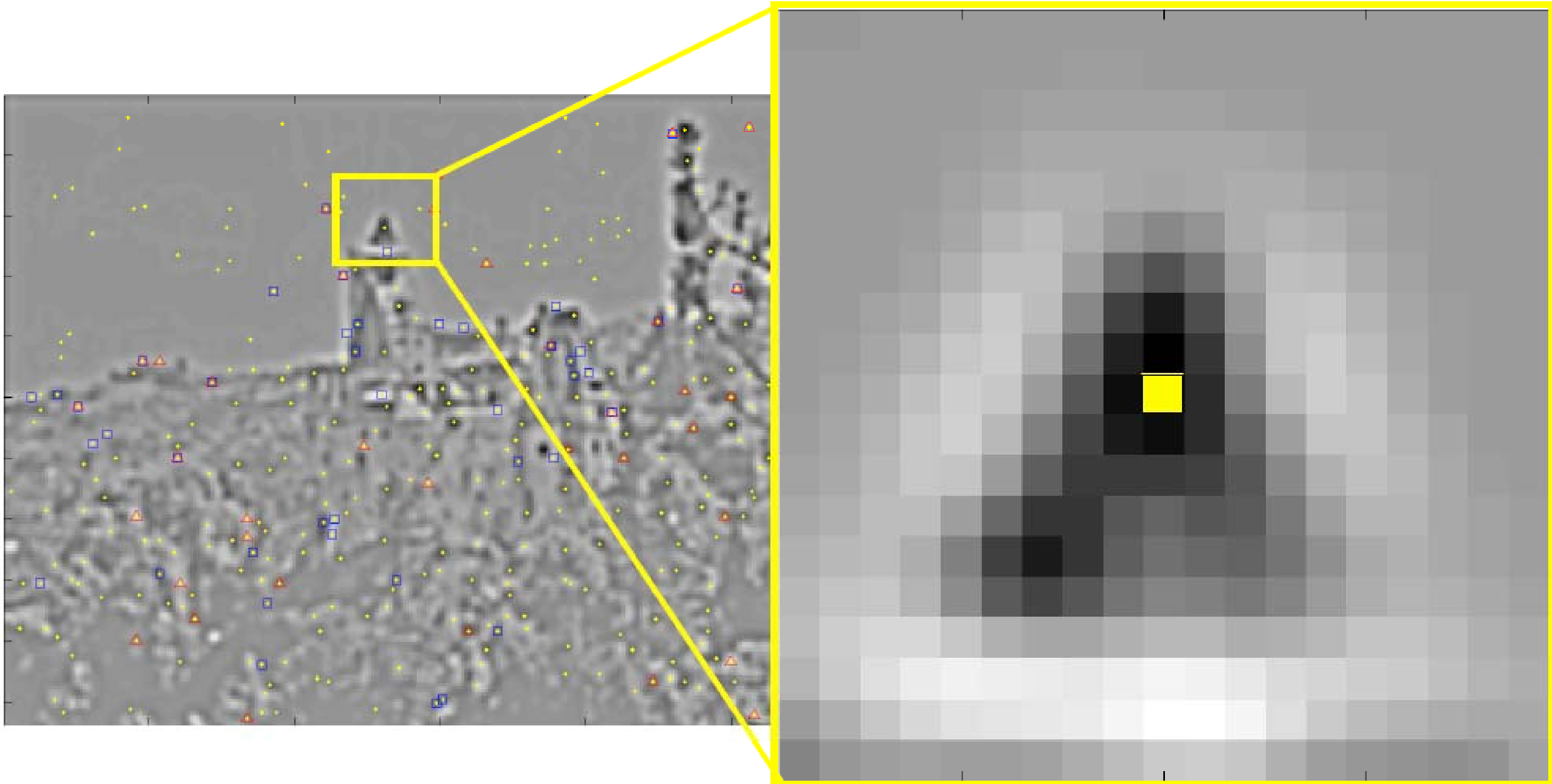


Finding Keypoints – Orientation

- Assign dominant orientation as the orientation of the keypoint

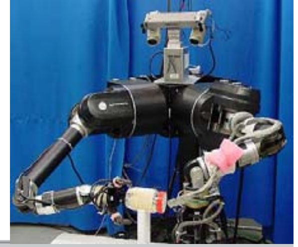


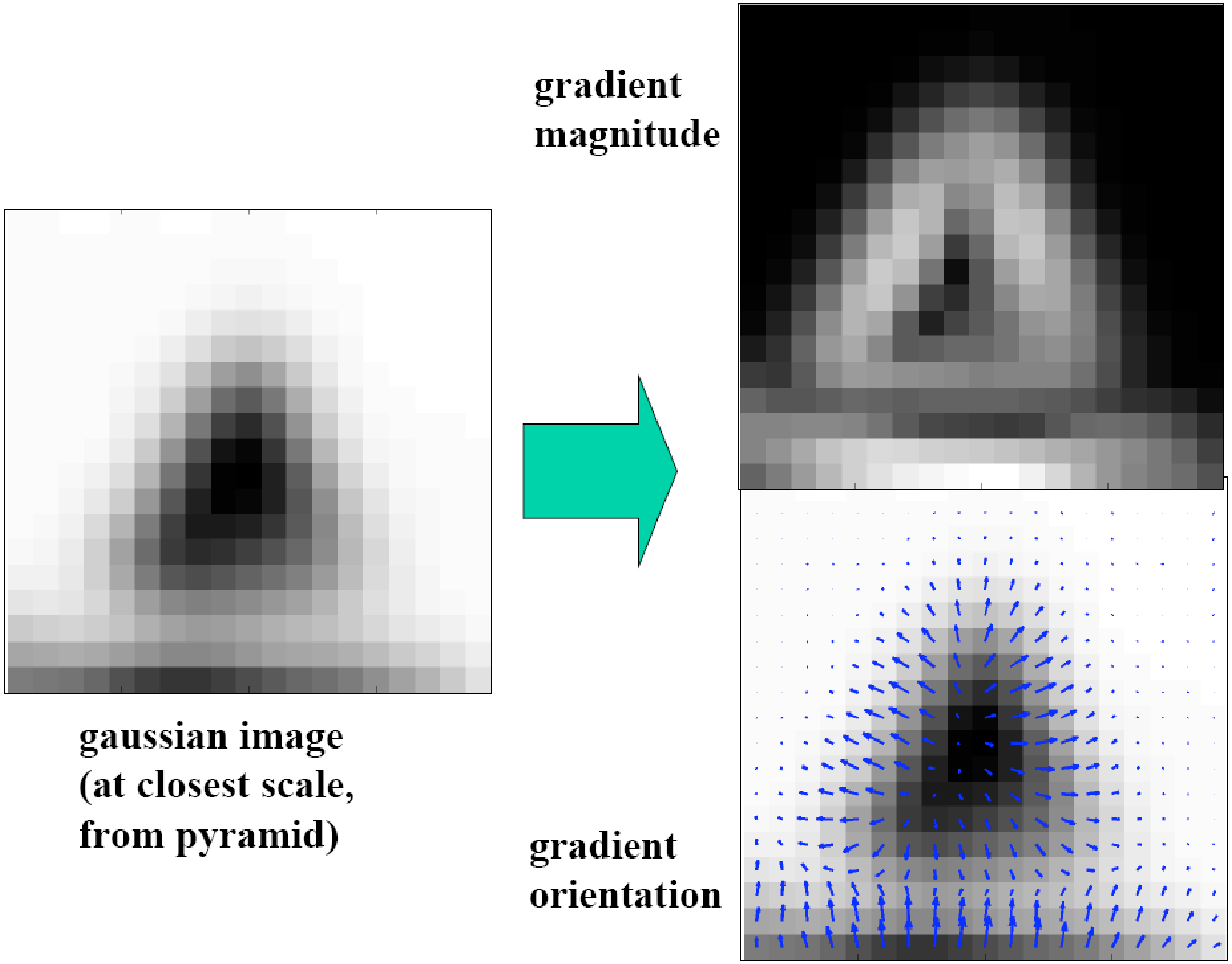
Orientation Assignment



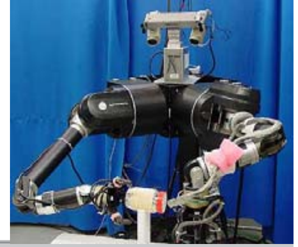
- Keypoint location = extrema location
- Keypoint scale is scale of the DOG image

Orientation Assignment

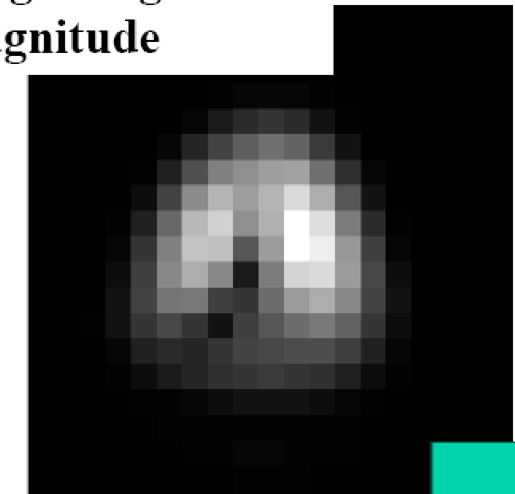




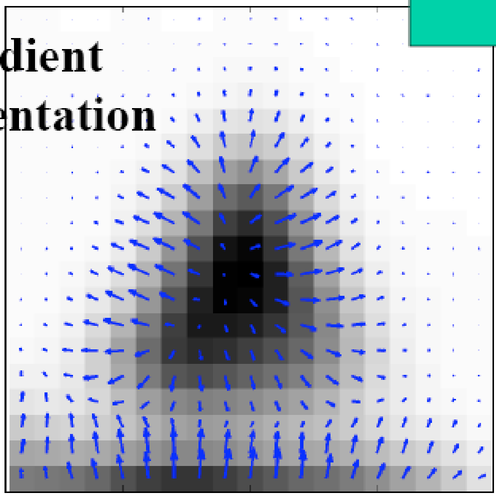
Orientation Assignment



weighted gradient magnitude

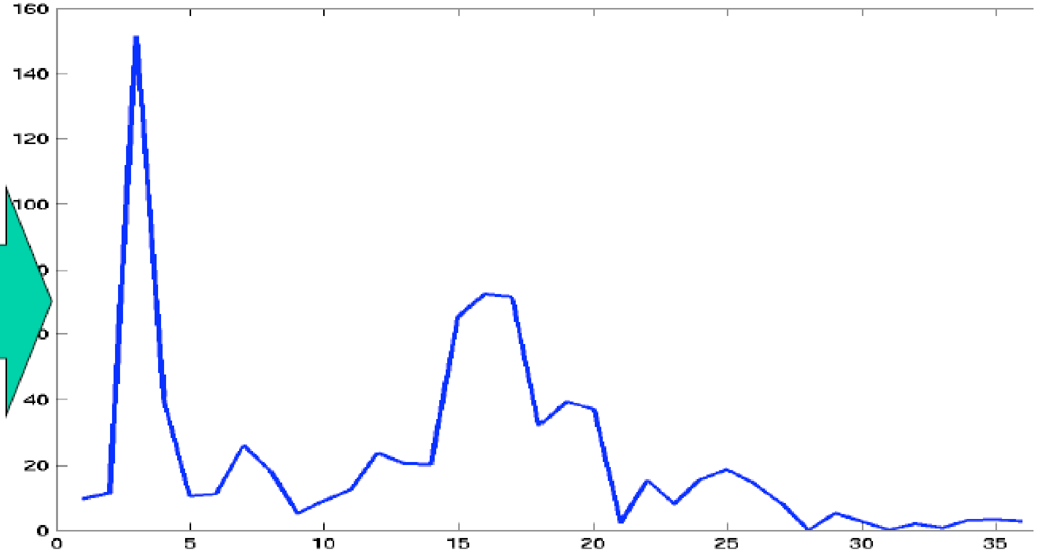


gradient orientation



weighted orientation histogram.

Each bucket contains sum of weighted gradient magnitudes corresponding to angles that fall within that bucket.



36 buckets

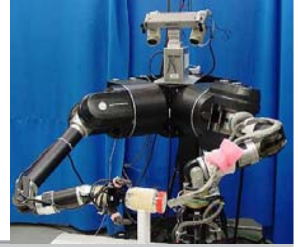
10 degree range of angles in each bucket, i.e.

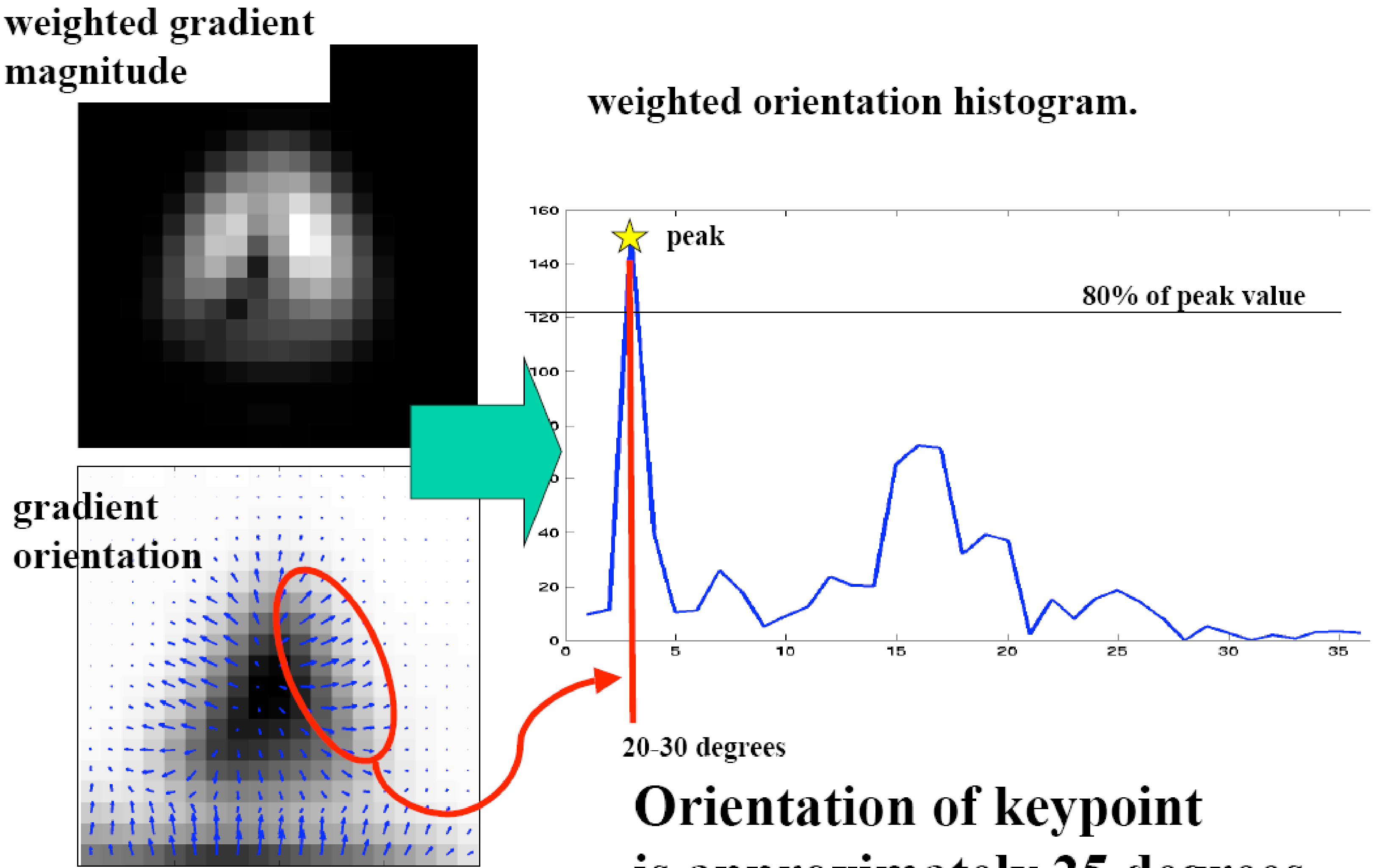
0 <=ang<10 : bucket 1

10<=ang<20 : bucket 2

20<=ang<30 : bucket 3 …

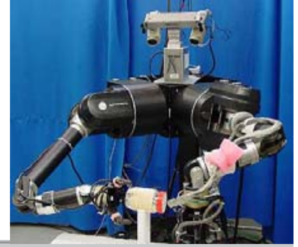
Orientation Assignment





Orientation of keypoint is approximately 25 degrees

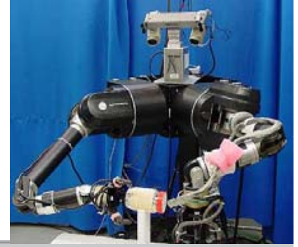
Finding Keypoints



- So far, we found…
  - where interesting things are happening
  - and its orientation

- With the hope of
  - Same keypoints being found, even under some scale, rotation, illumination variation.

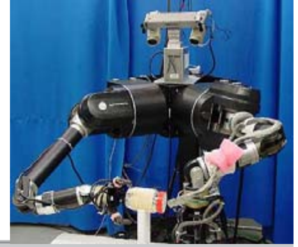
Finding Keypoints



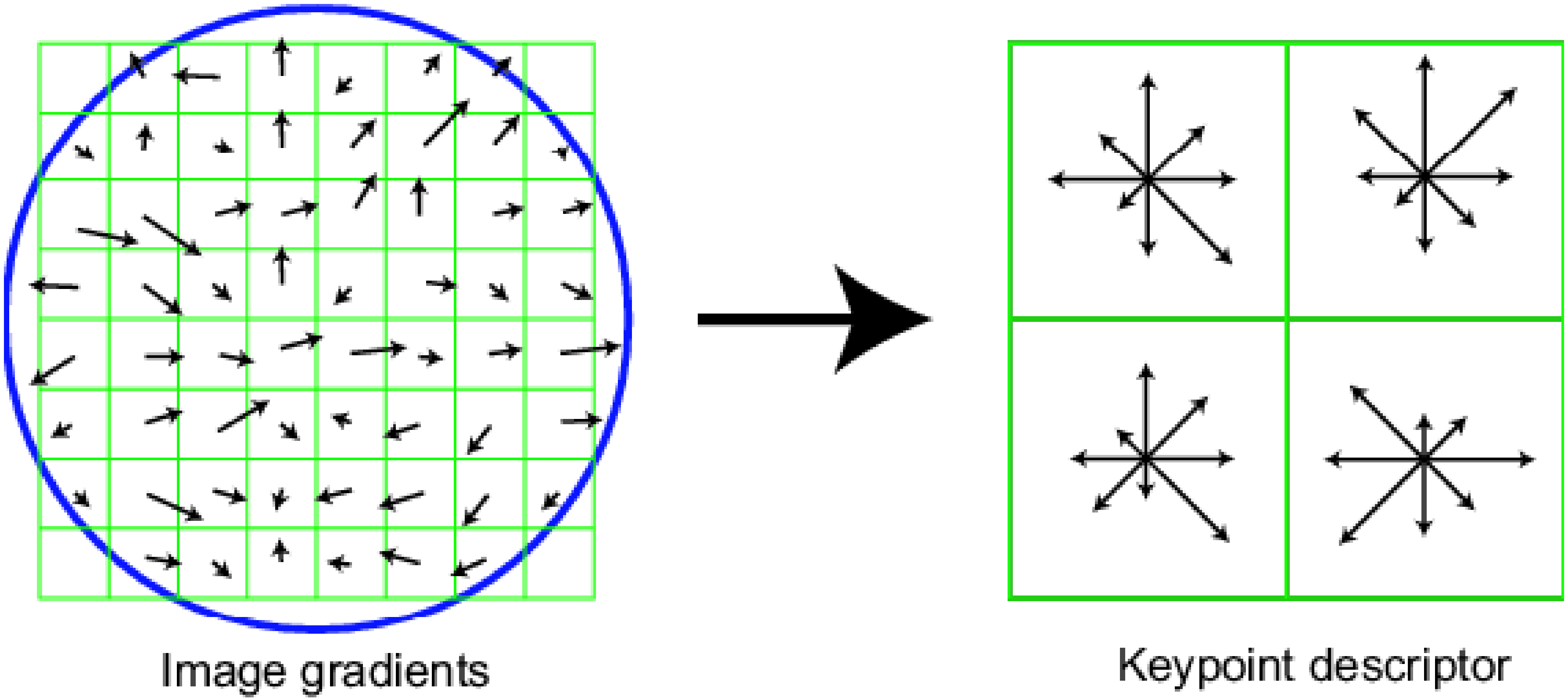
- Extract features
  - ☑ Find keypoints
    - ☑ Scale, Location
    - ☑ Orientation
  - Create signature

- Match features

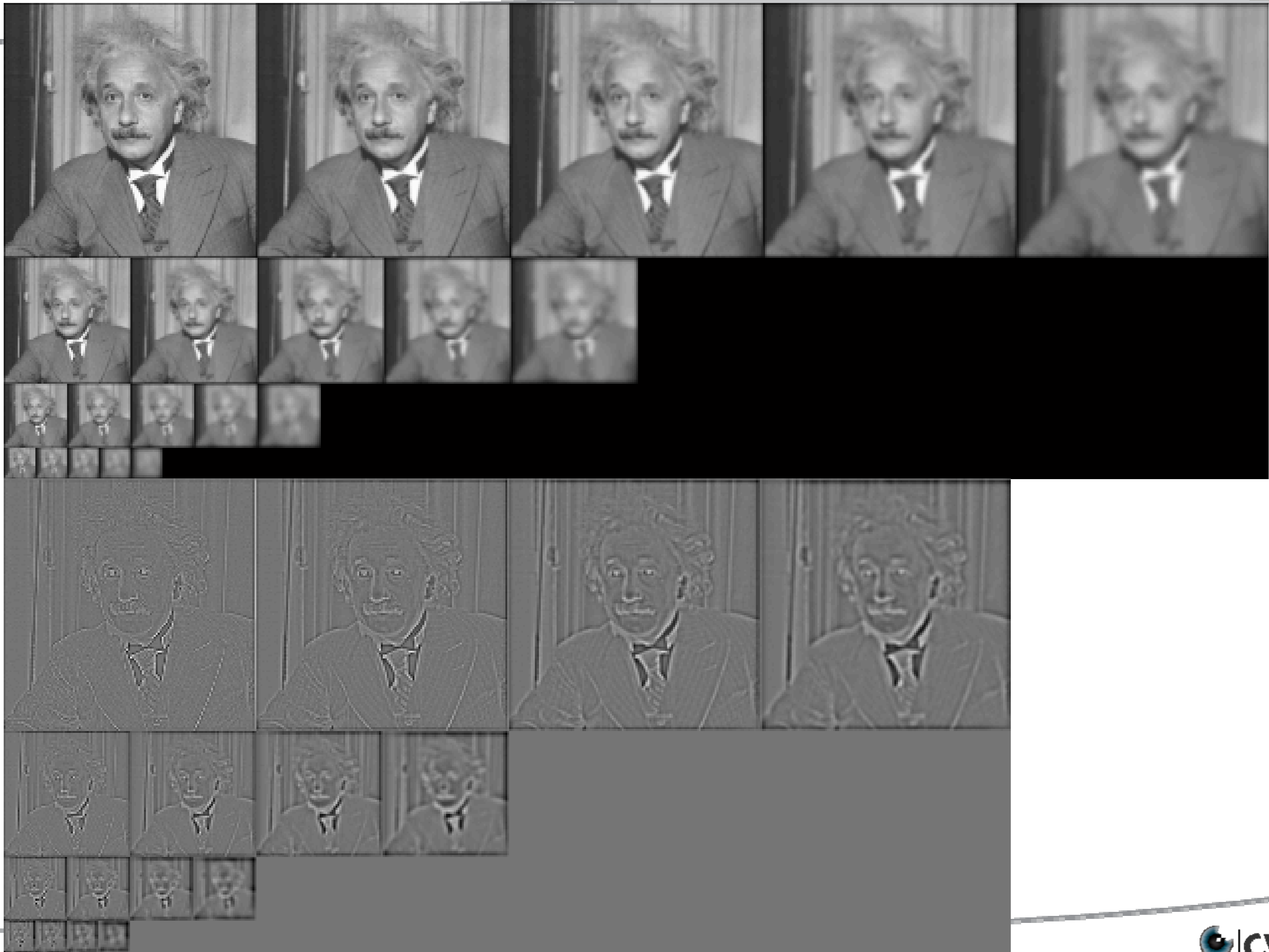
4. Creating Signature



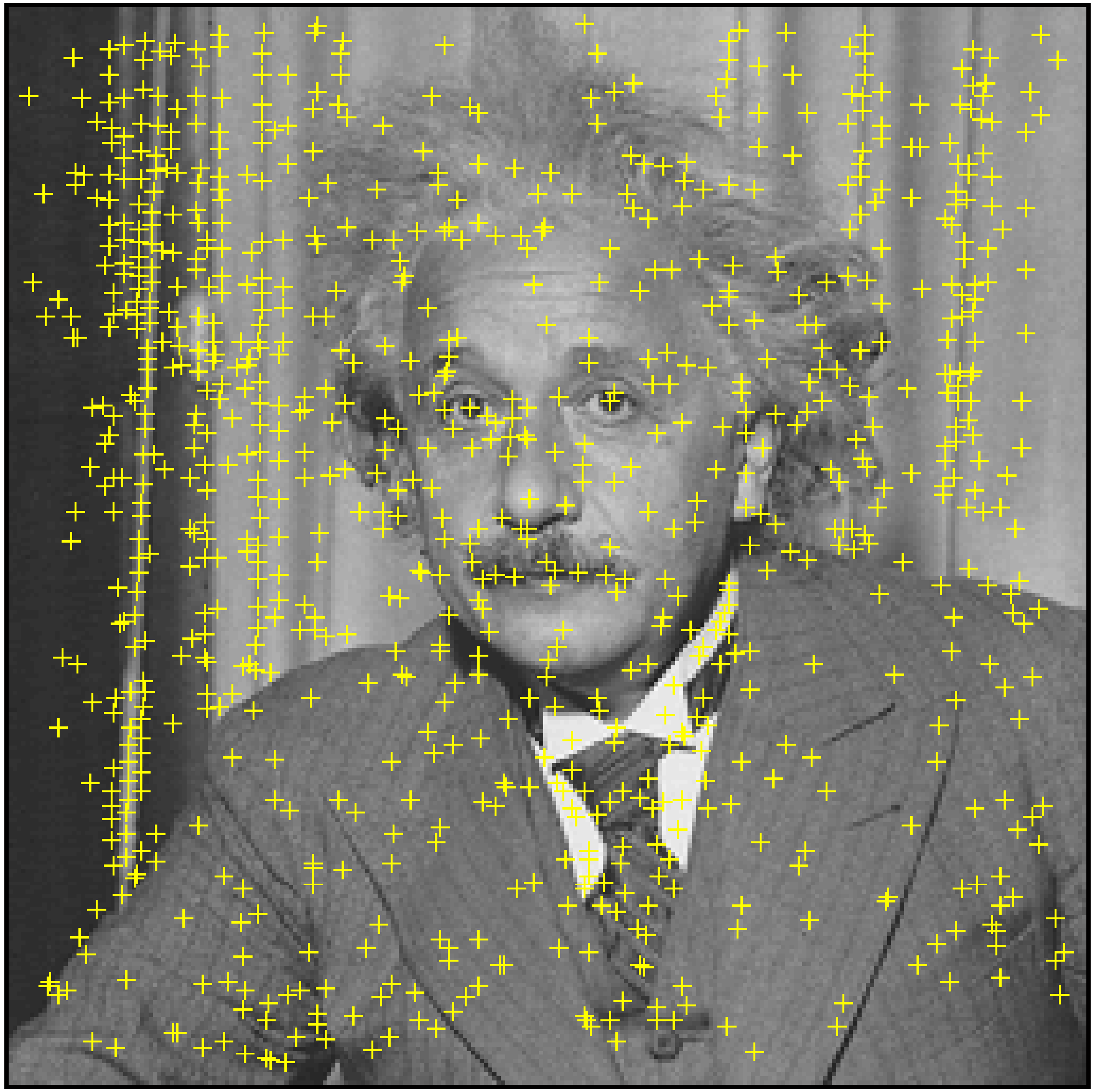
- Thresholded image gradients are sampled over 16x16 array of locations in scale space
- Create array of orientation histograms
- 8 orientations x 4x4 histogram array = 128 dimensions of feature vector



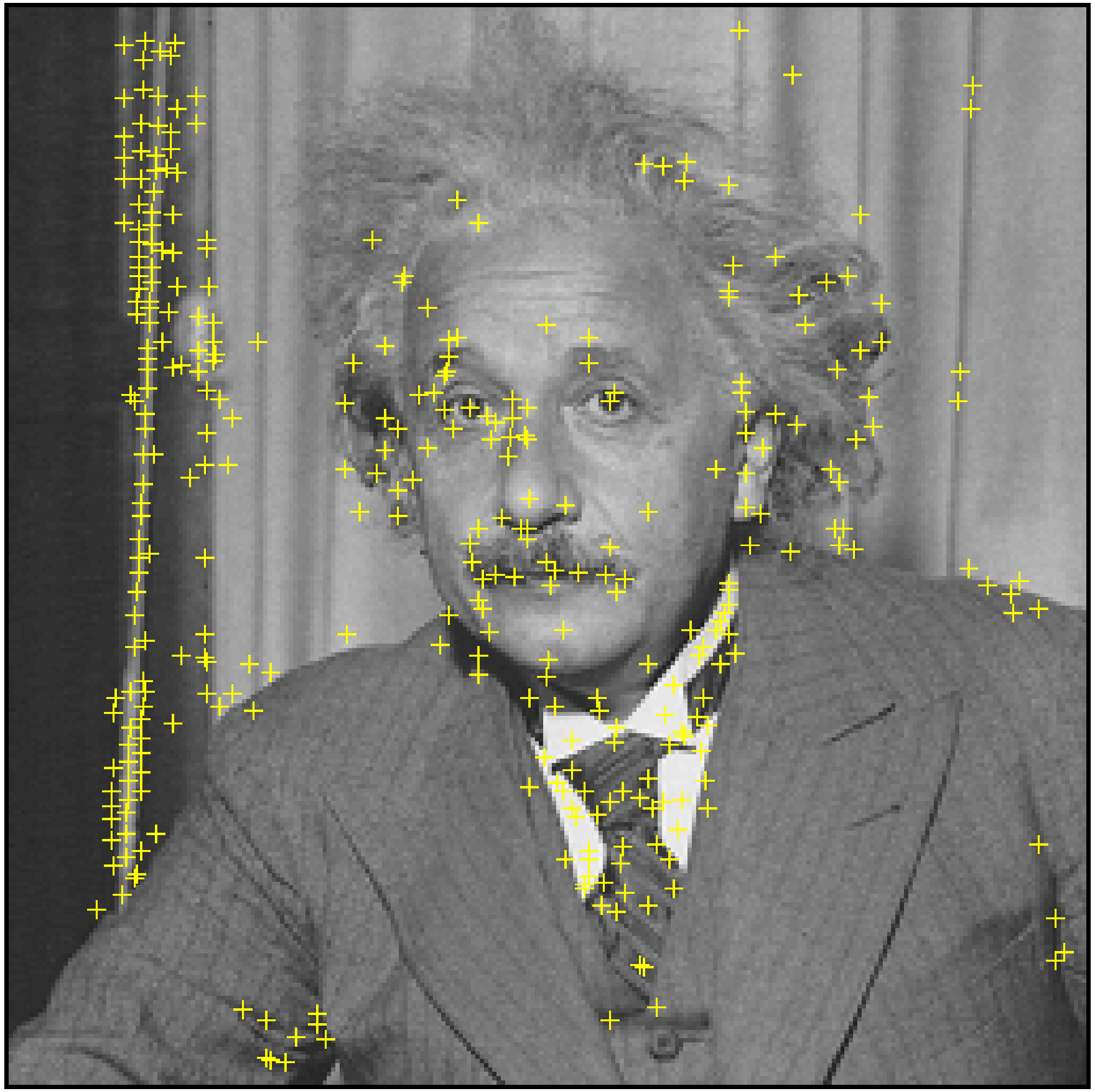
SIFT Example



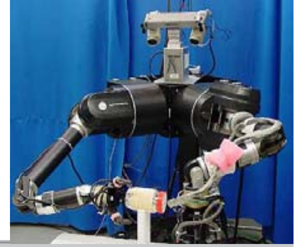
Maxima in D

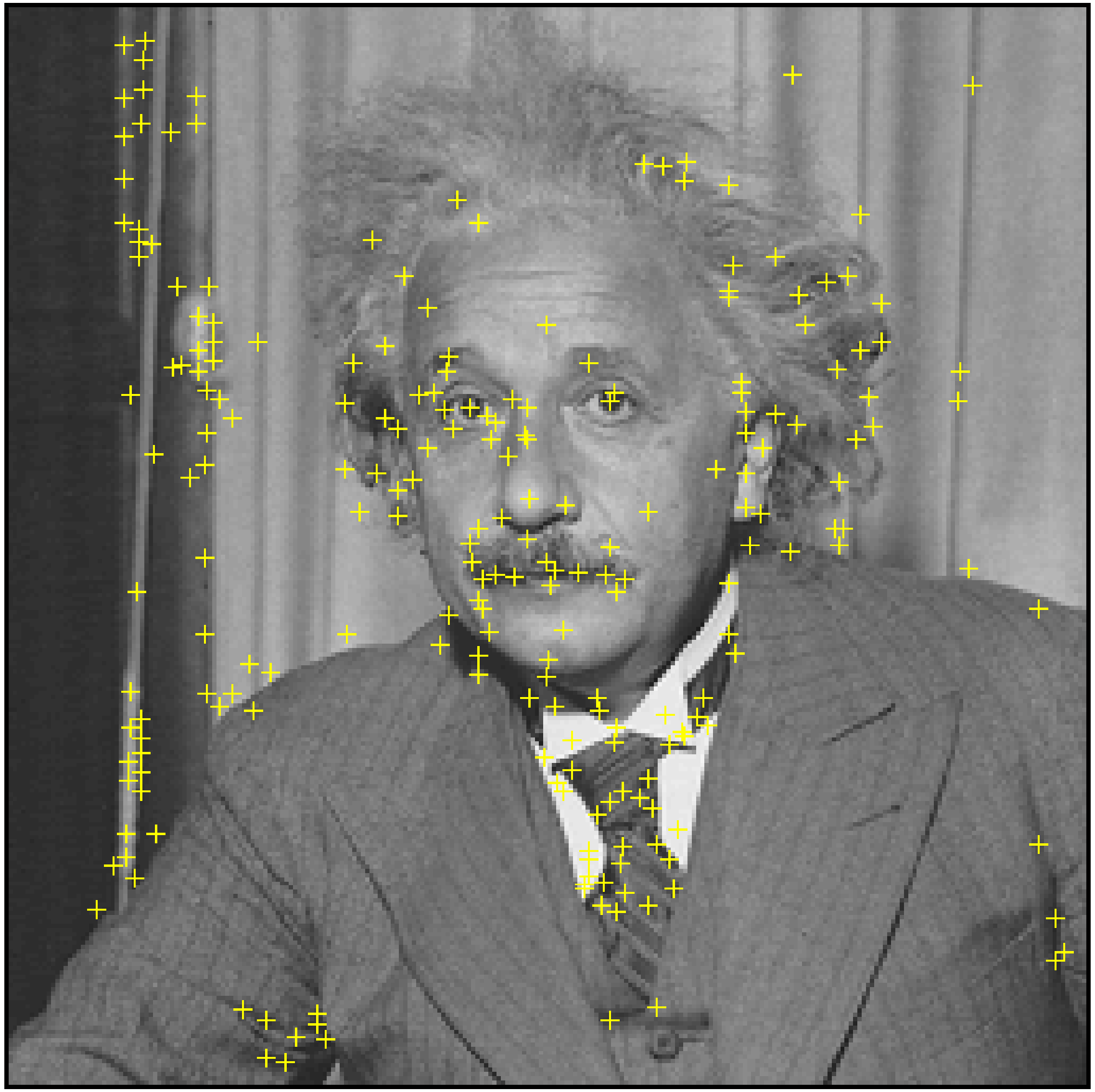


Remove Low Contrast

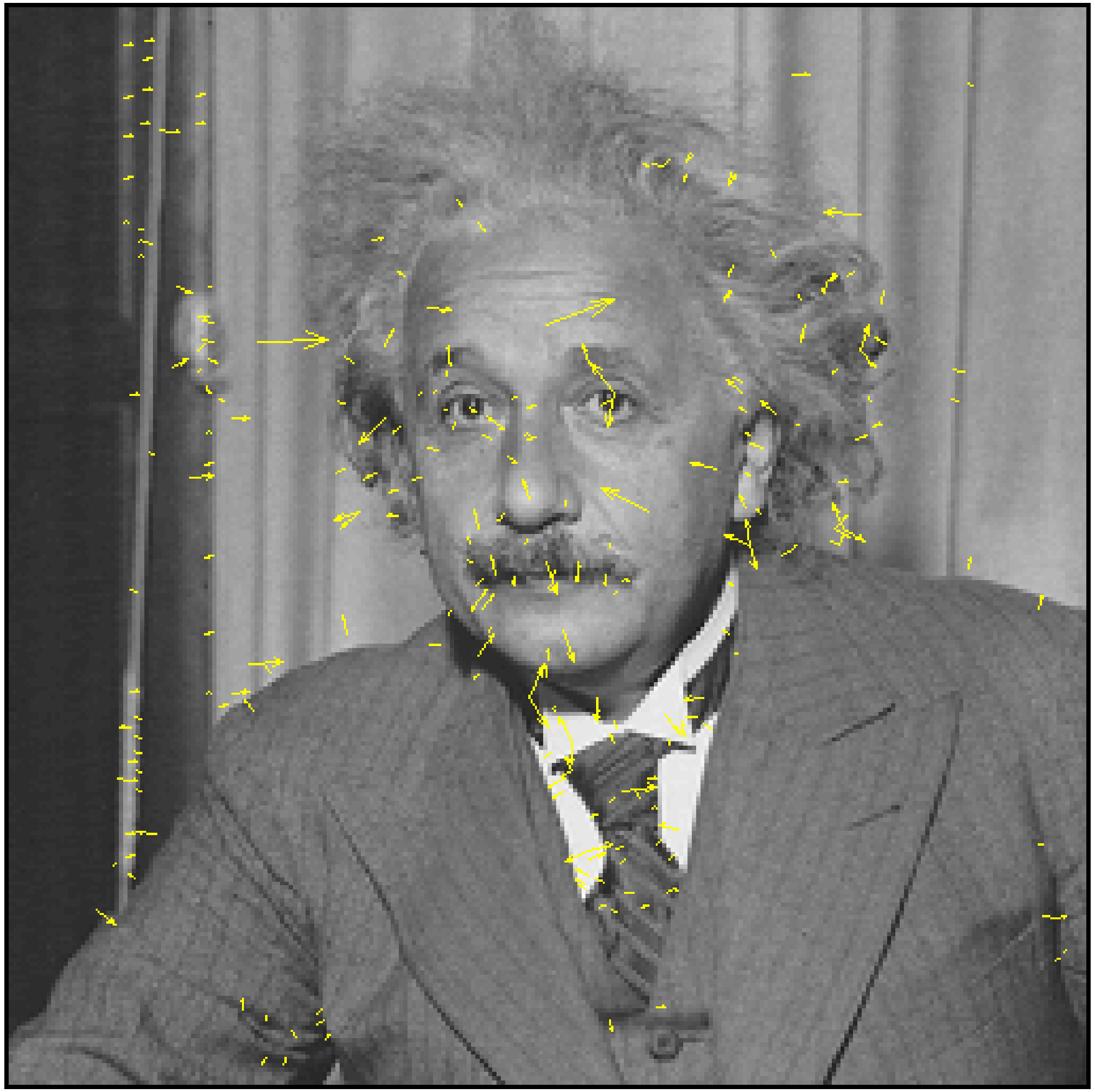


Remove Edges





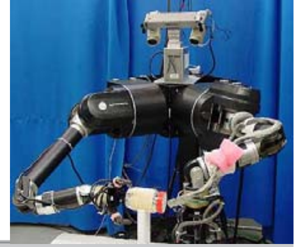
SIFT Descriptor



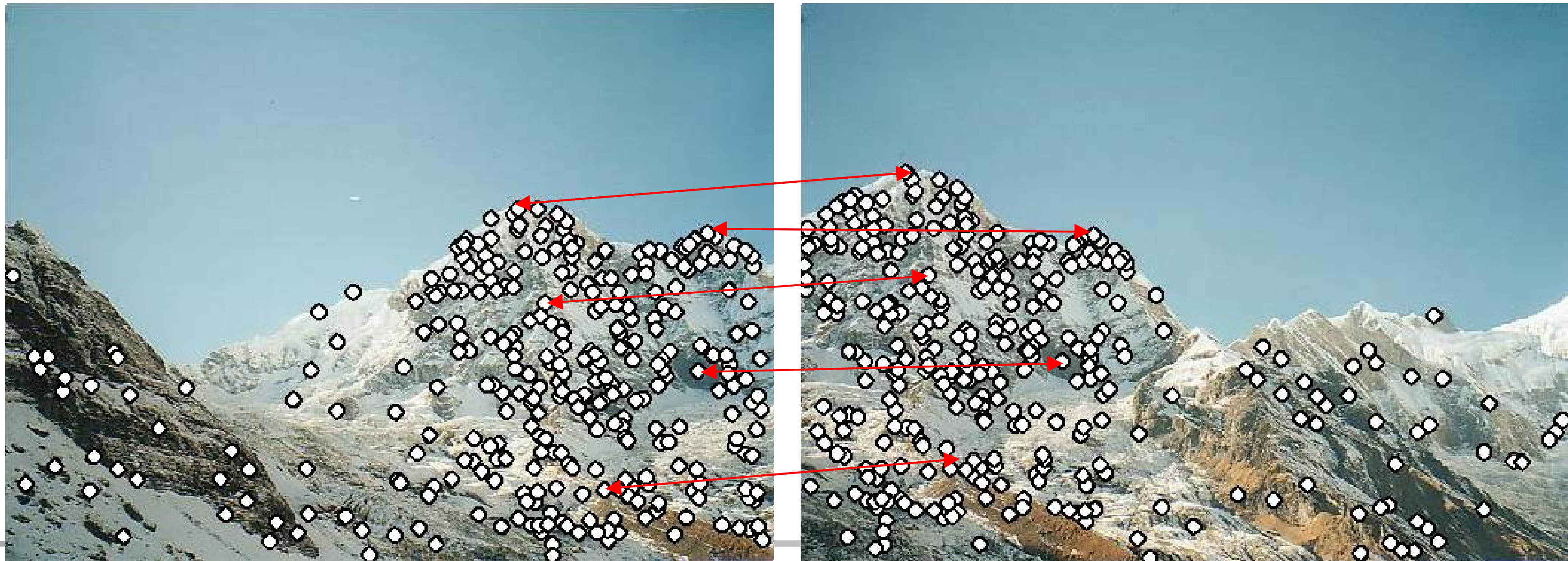
SIFT

- Extract features
  - ☑ Find keypoints
    - ☑ Scale, Location
    - ☑ Orientation
  - ☑ Create signature

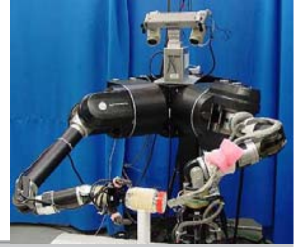
- Match features
  - Nearest neighbor, Hough voting, Least-square affine parameter fit

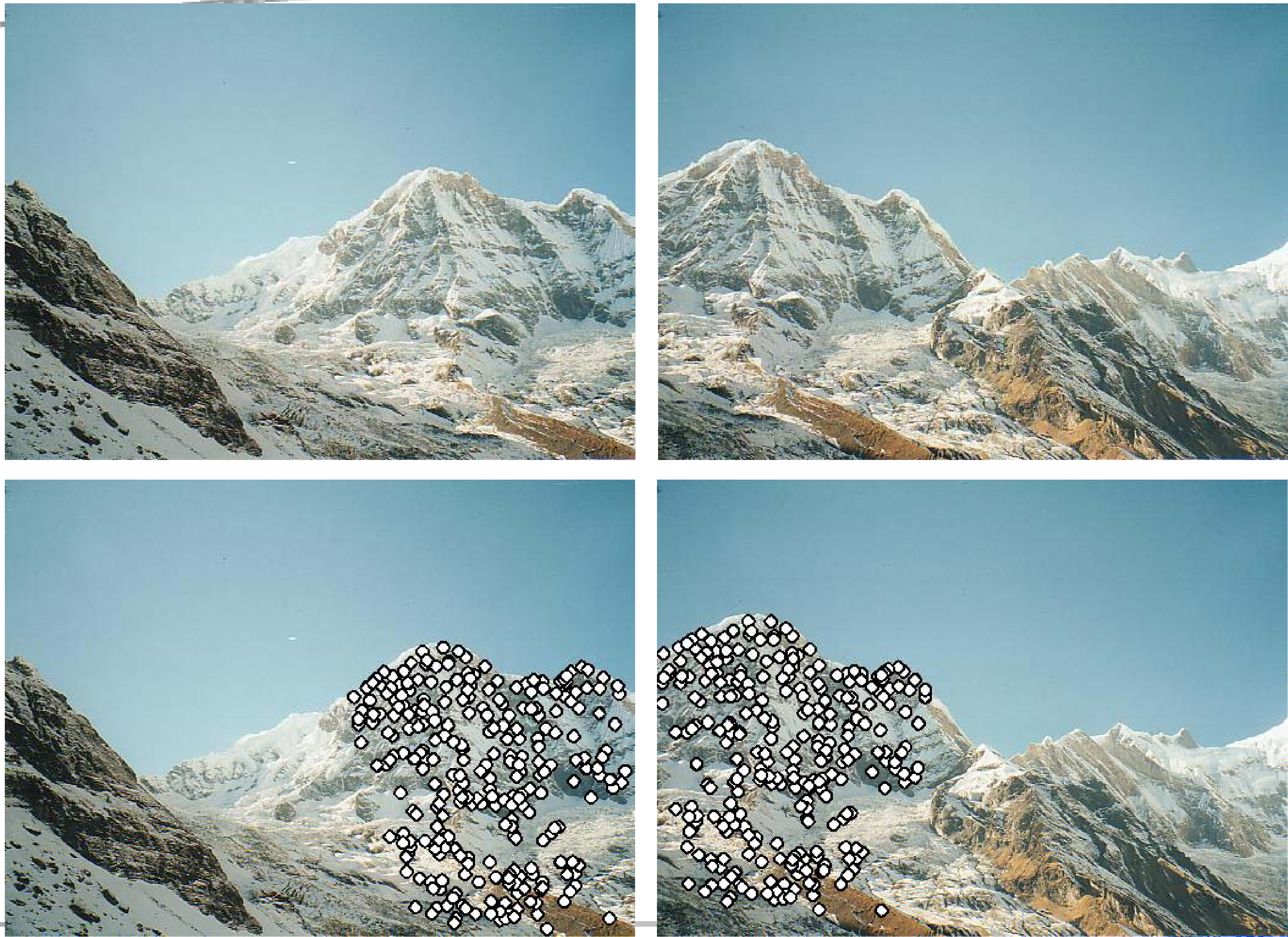


Example: Automatic Image Stitching

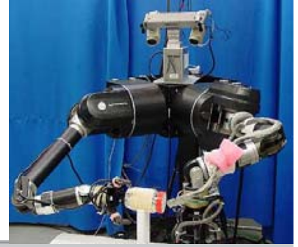


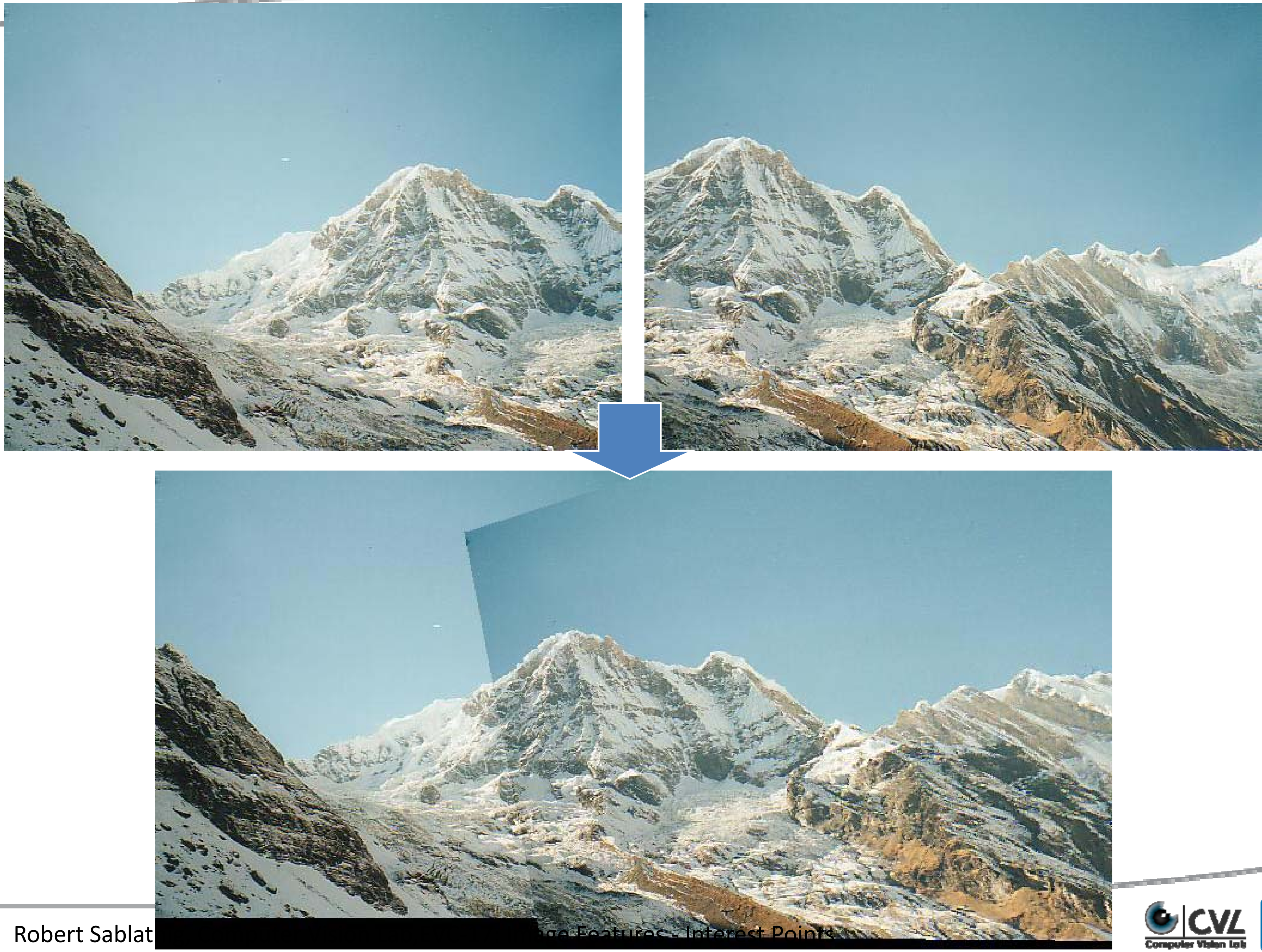
Example: Automatic Image Stitching



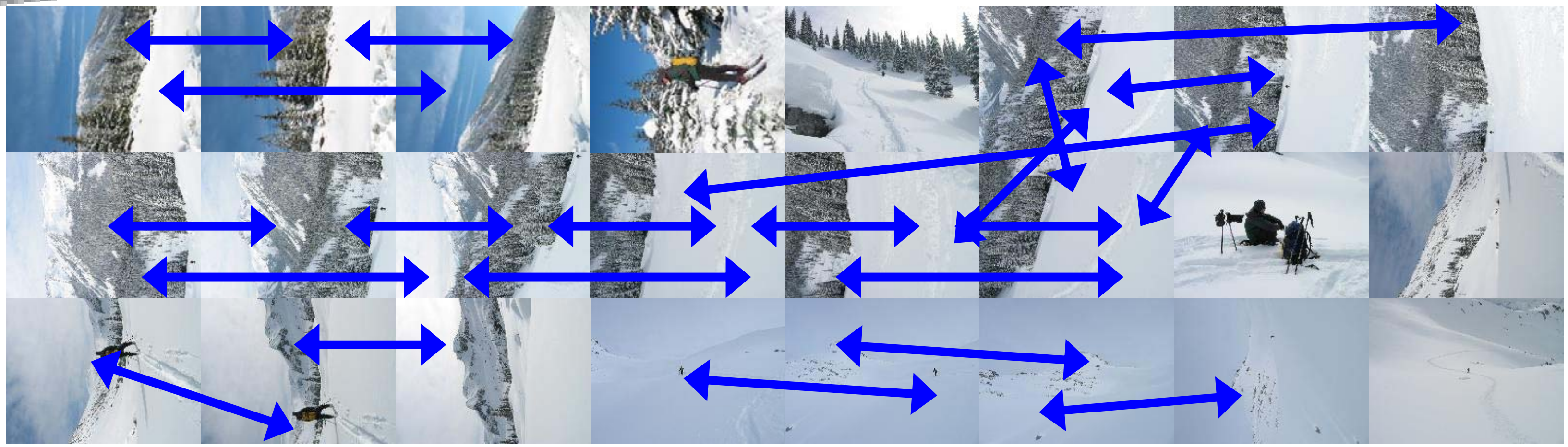


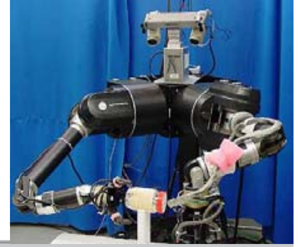
Example: Automatic Image Stitching



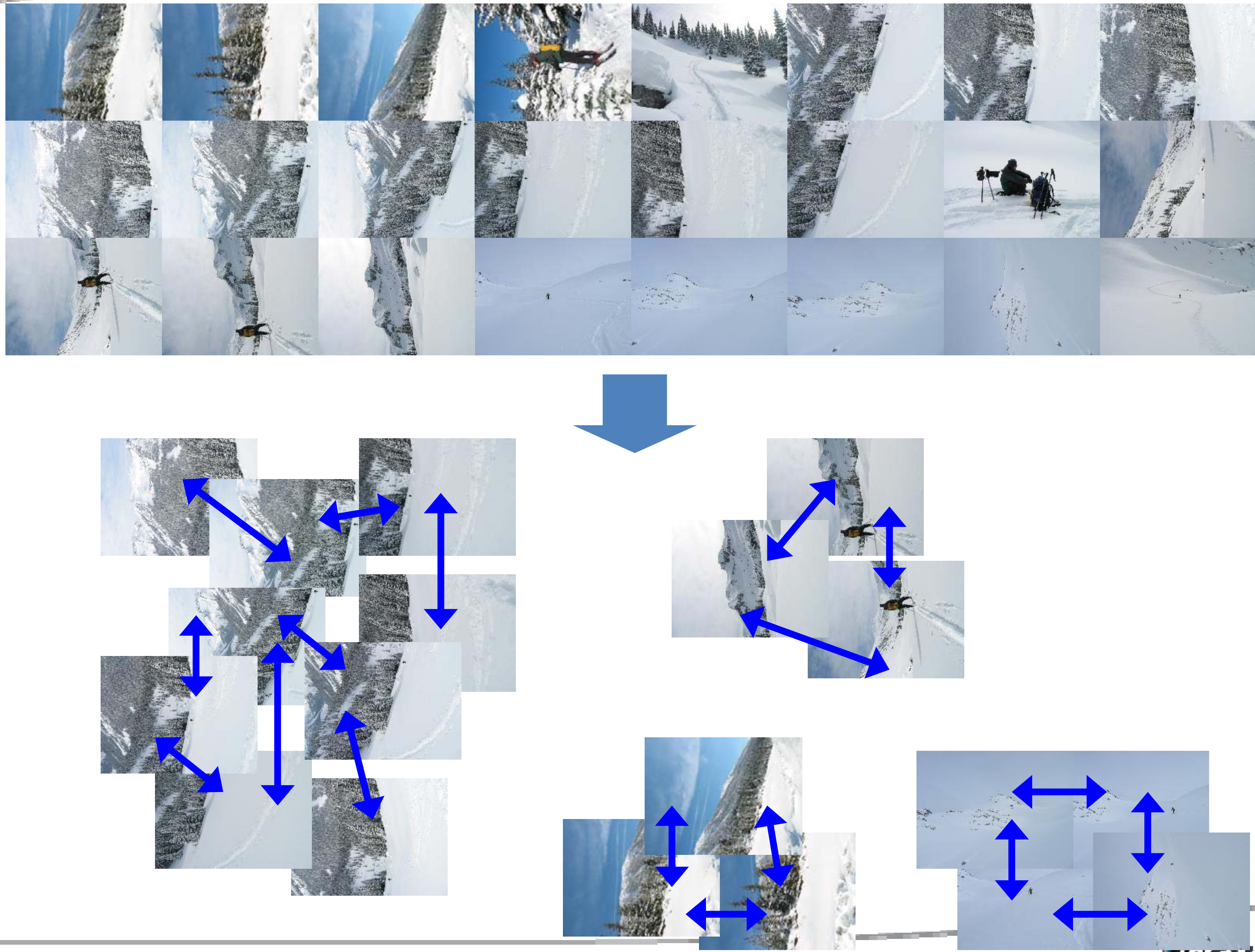


Example: Automatic Image Stitching





Example: Automatic Image Stitching



Example: Automatic Image Stitching