MIT CSAIL

6.869: Advances in Computer Vision

MIT COMPUTER VISION

Image Features

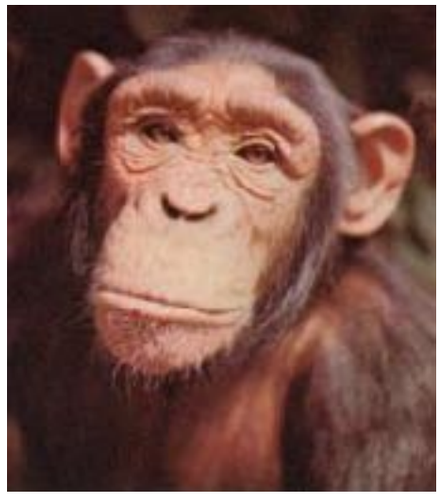
Lecture 12

Tali Dekel

Vision Systems

One eye

Two eyes



N eyes

?

The more cameras/eyes the more information about the 3D world

Camera Systems

One camera

Two cameras

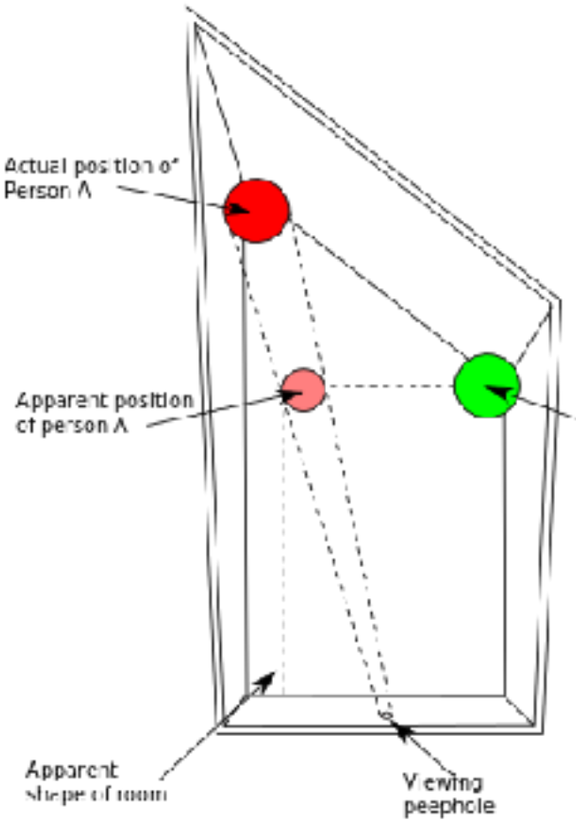
N cameras

The more cameras the more information about the 3D world

The majority of our visual data is captured with a single eye/camera…

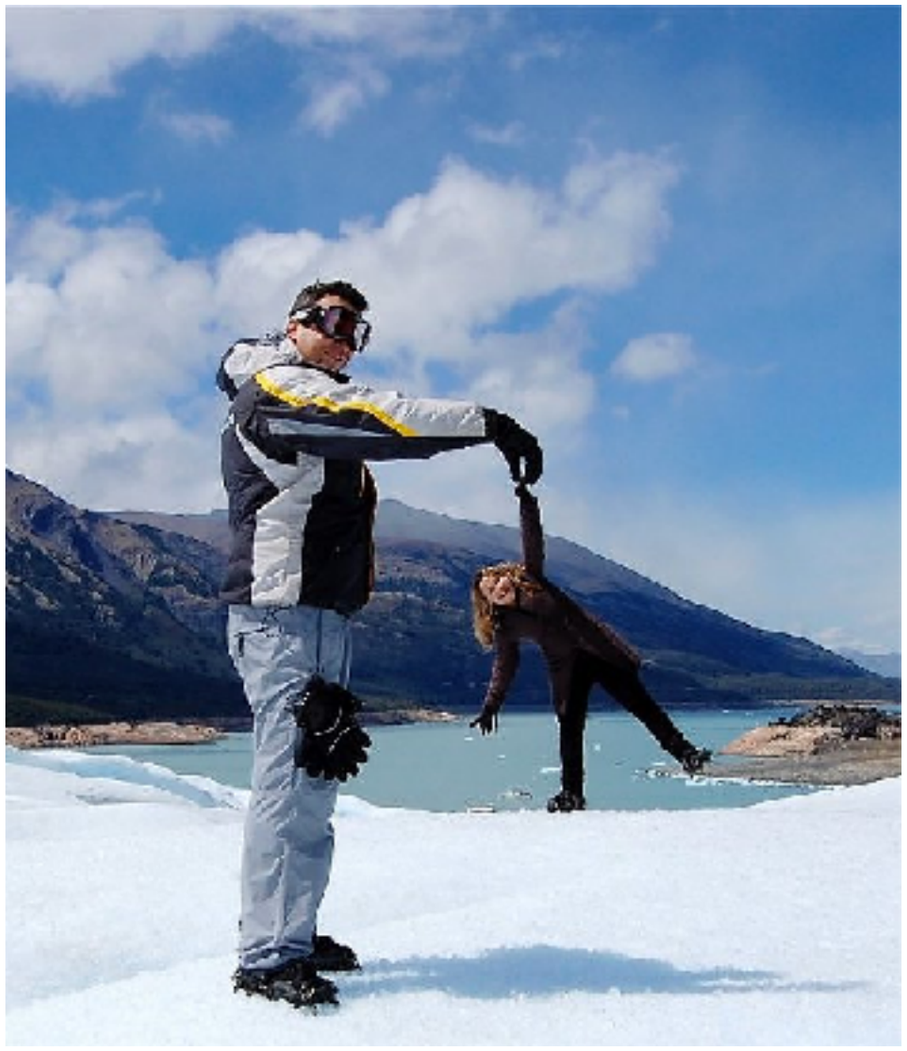
- Lost of information 3D—> 2D projection
- Limited field of view
- What information can we extract about our 3D world from a single image?

The power of linear perspective



3D percept is driven by the scene, which imposes its ruling to the objects

The power of linear perspective



http://www.opticalspy.com/opticals/forced-perspective-photos

Vanishing Points

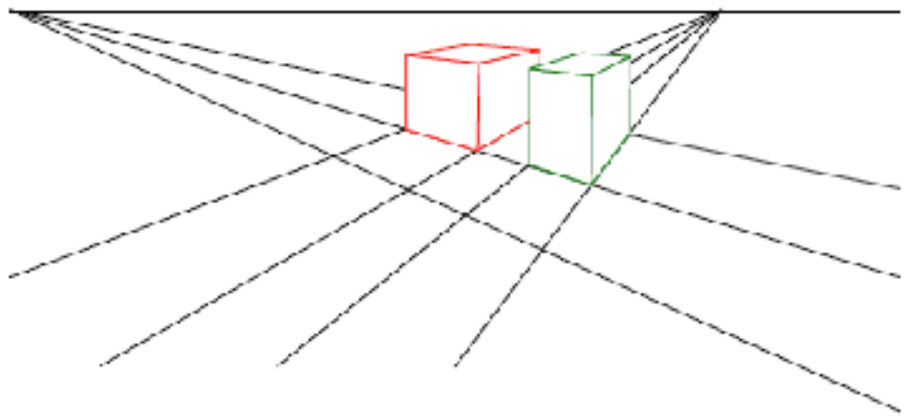
Vanishing Points

In 3D: parallel to image plane

In 2D: parallel (no VP)

Vanishing Points

- Parallel lines in 3D intersect at a single point of (vanishing point) in the image plane
- The perspective projection of a point at infinity onto the image plane



$$x(t) = x_0 + at$$
$$y(t) = y_0 + bt$$
$$z(t) = z_0 + ct$$

Line in world coordinates

$$x'(t) = \frac{fx}{z} = \frac{f(x_0 + at)}{z_0 + ct}$$
$$y'(t) = \frac{fy}{z} = \frac{f(y_0 + bt)}{z_0 + ct}$$

Perspective projection

VP depends only on the line's orientation (not position)

$$x'(t) \longrightarrow \frac{fa}{c}$$
$$y'(t) \longrightarrow \frac{fb}{c}$$

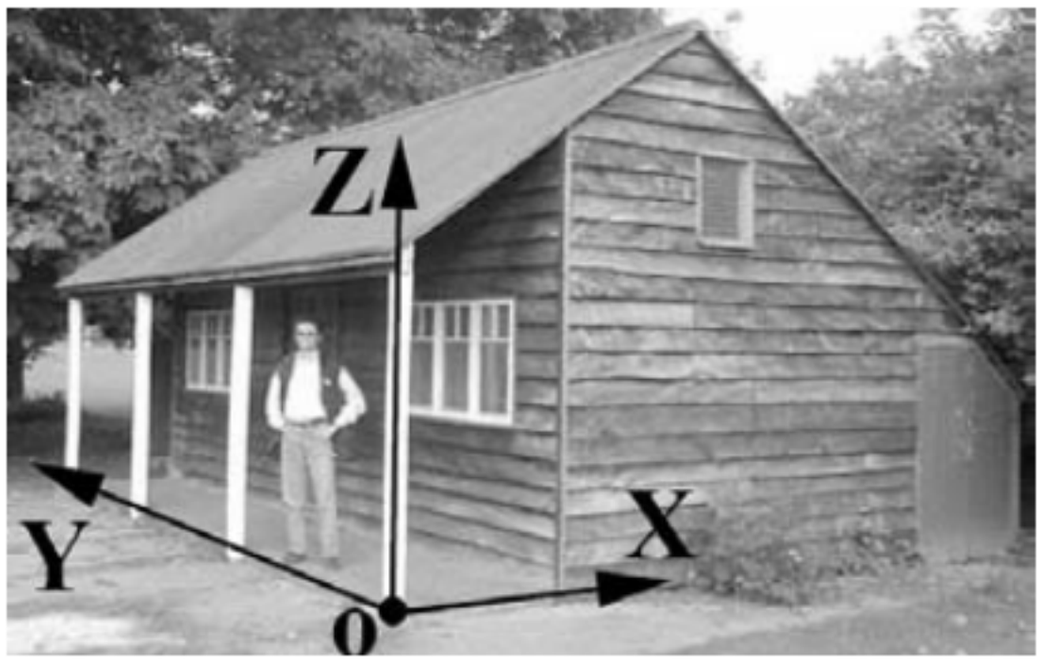
Vanishing point

Applications of liner perspective

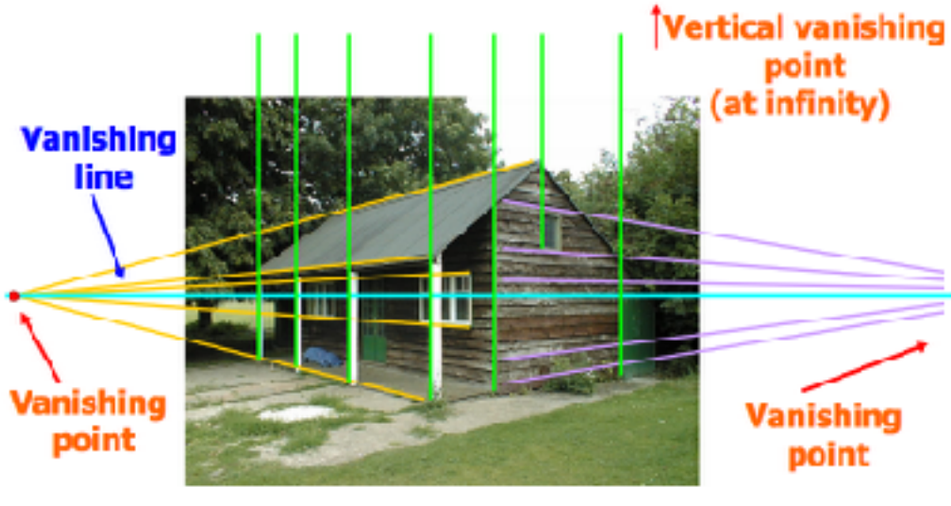
Pietro Perugino's use of perspective in the *Delivery of the Keys* fresco at the Sistine Chapel (1481–82) helped bring the Renaissance to Rome. (From Wikipedia)

Applications of liner perspective

- Single Image Metrology, *Criminisi and Zisserman, IJCV 2000*
  Interactively allows to measure heights (or other dimensions), render a new view of the scene.



- 2 orthogonal vanishing points on the ground plane
- 1 orthogonal vanishing point on the vertical axis
- Height of a reference object (given by a user)



Applications of perspective geometry

- Deviation Magnification: Revealing Departures from Ideal Geometries,
  Neal Wadhwa, Tali Dekel, Donglai Wei, Fredo Durand and William T. Freeman, SIGAsia'15

Applications of perspective geometry

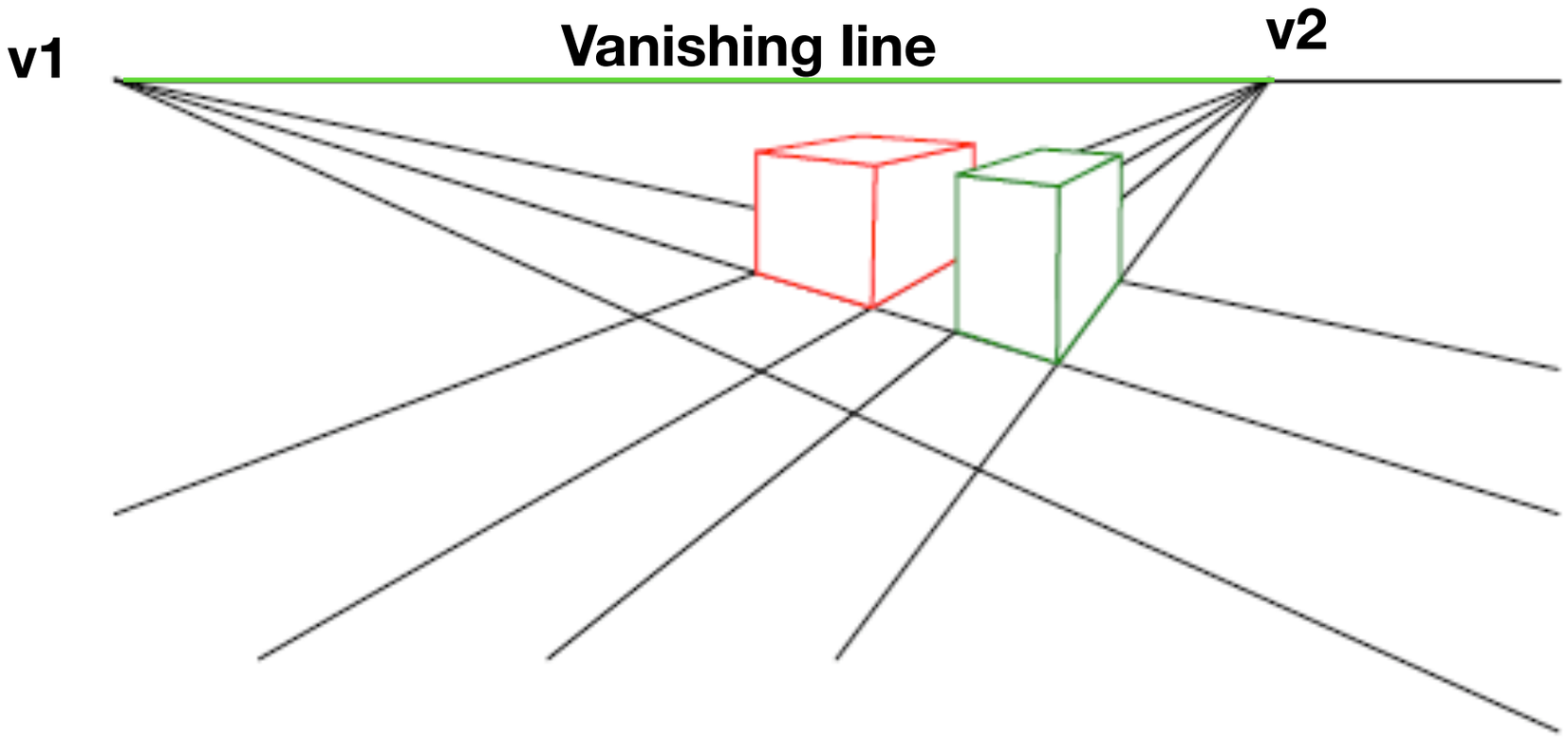
- Deviation Magnification: Revealing Departures from Ideal Geometries,
  Neal Wadhwa, Tali Dekel, Donglai Wei, Fredo Durand and William T. Freeman, SIGAsia'15

Applications of perspective geometry

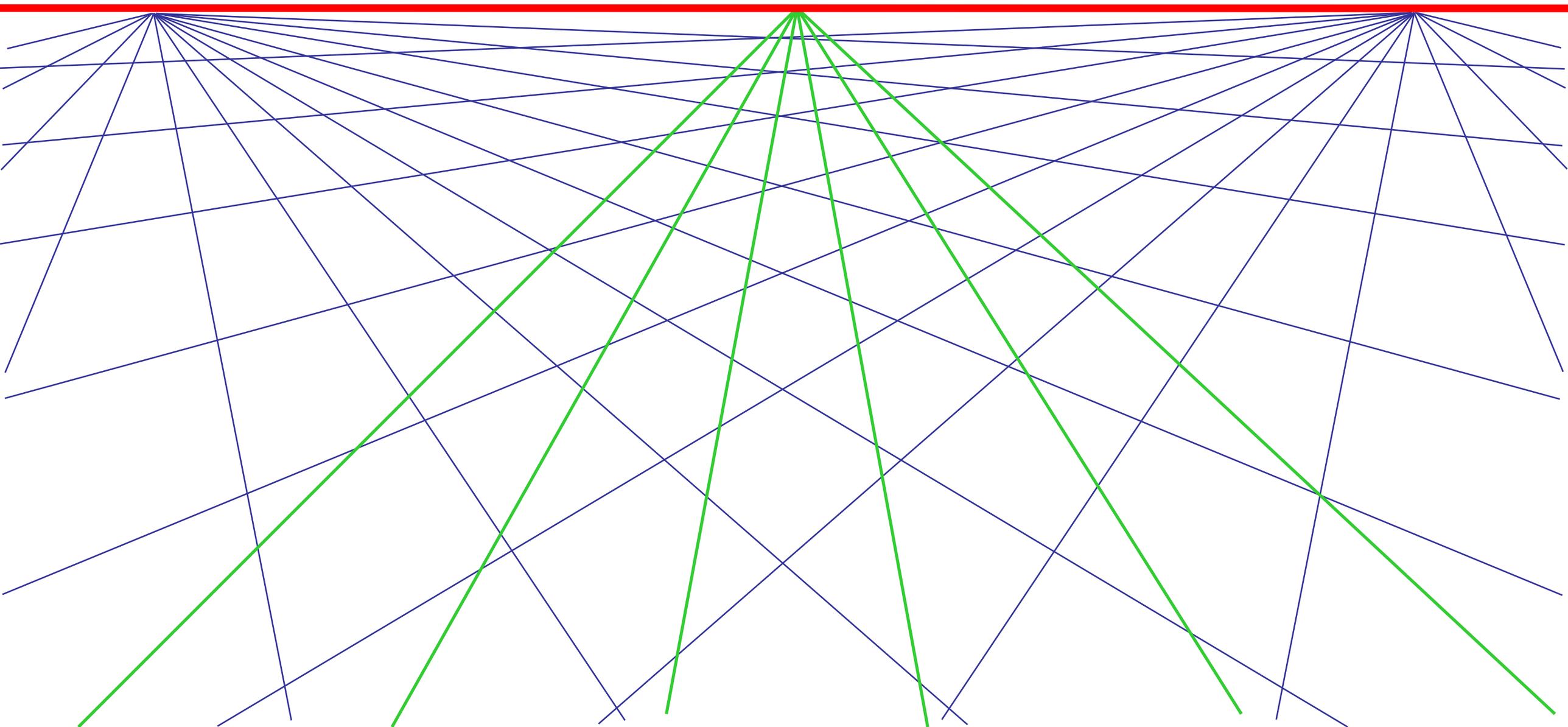
- Deviation Magnification: Revealing Departures from Ideal Geometries,
  Neal Wadhwa, Tali Dekel, Donglai Wei, Fredo Durand and William T. Freeman, SIGAsia'15

Tilting is same as measured by civil engineers!

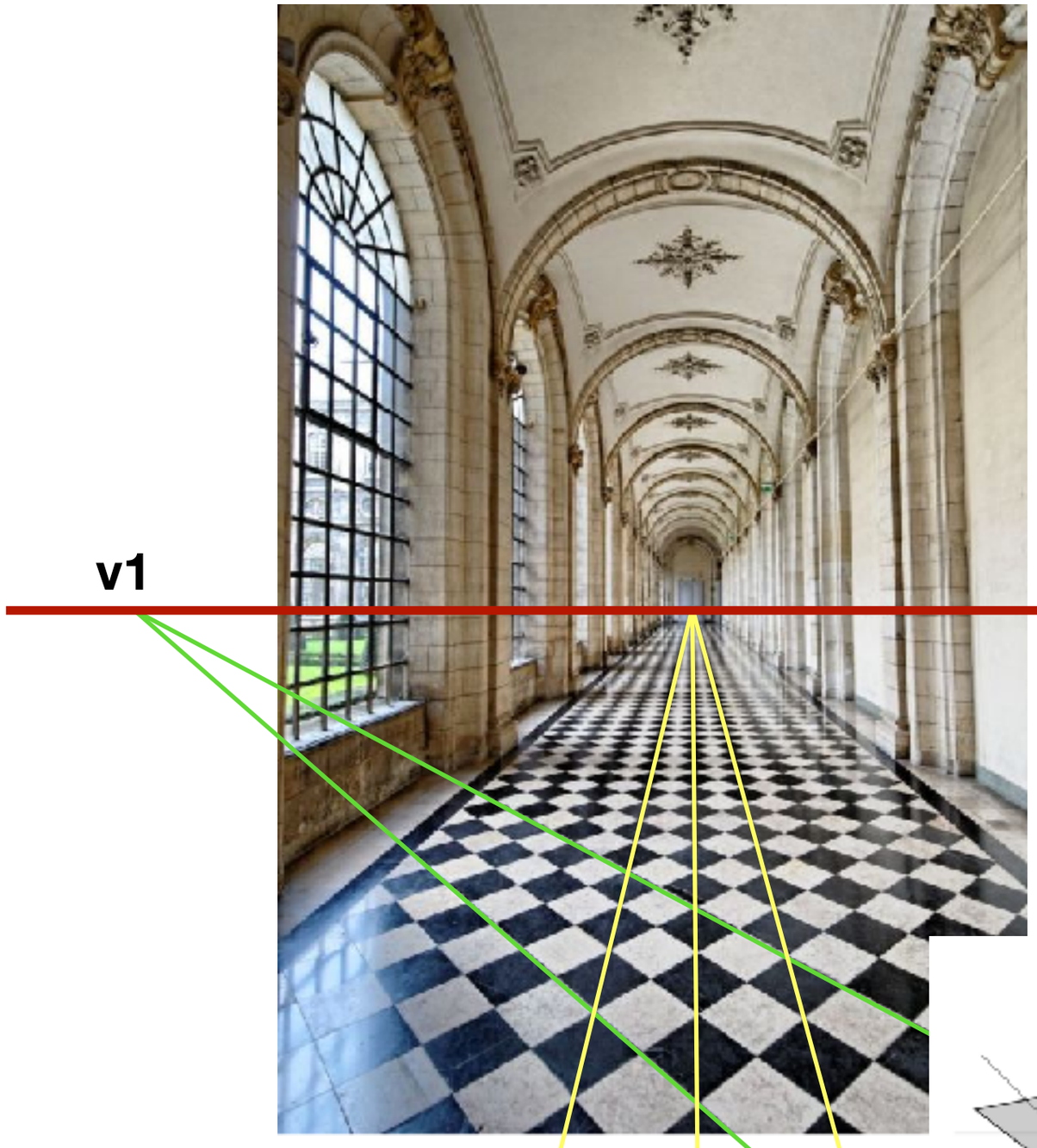
Vanishing Line

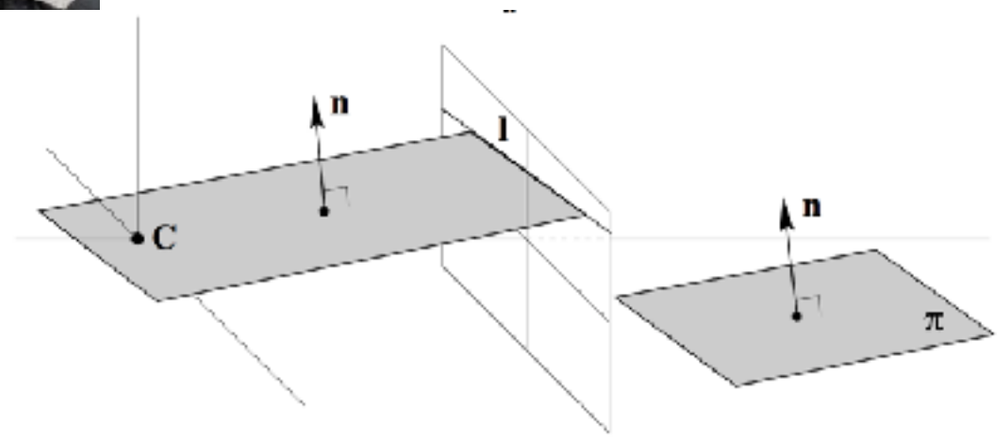


Horizon



Horizon — The vanishing line of the ground plane

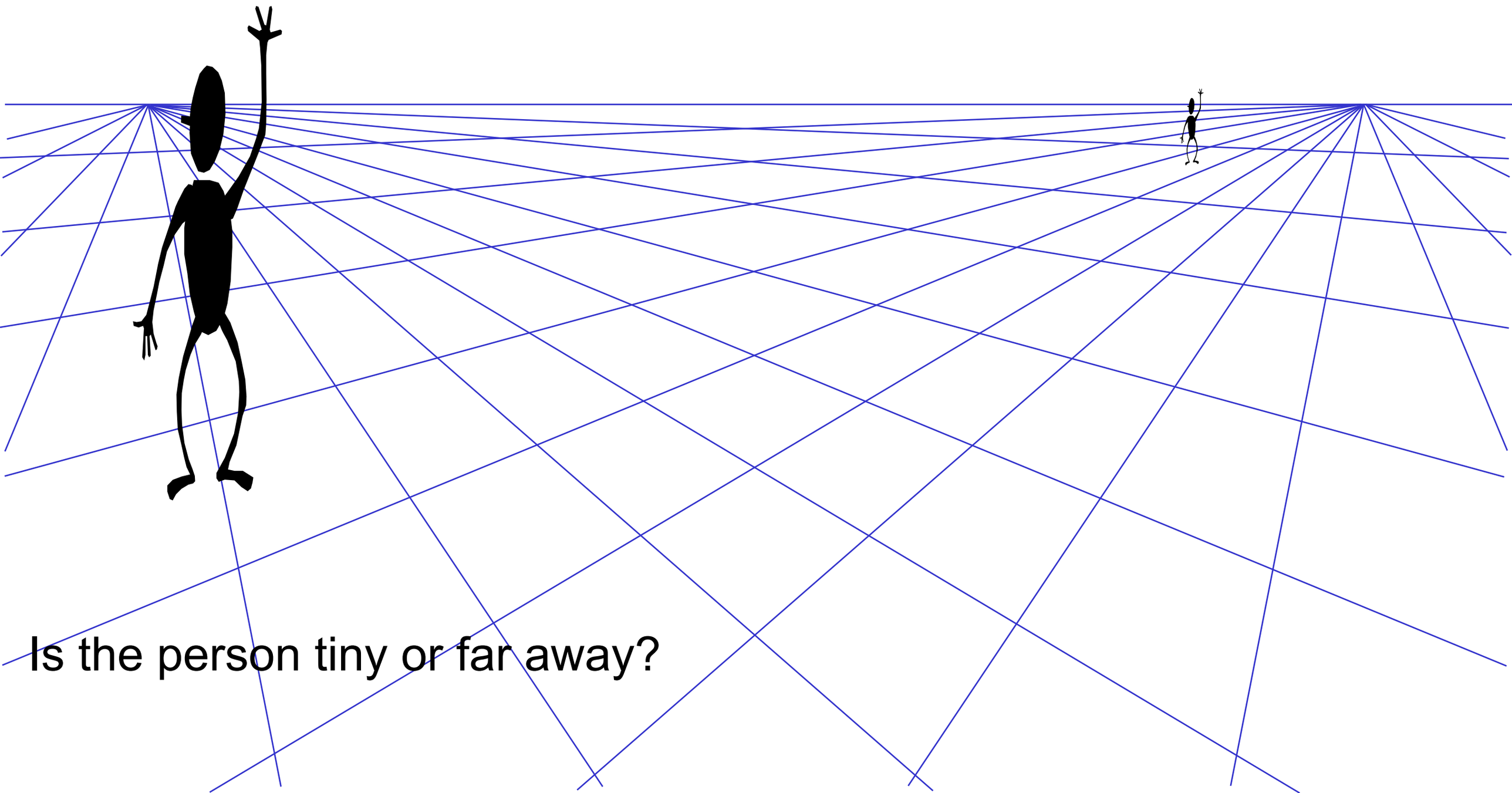




Moon Illusion

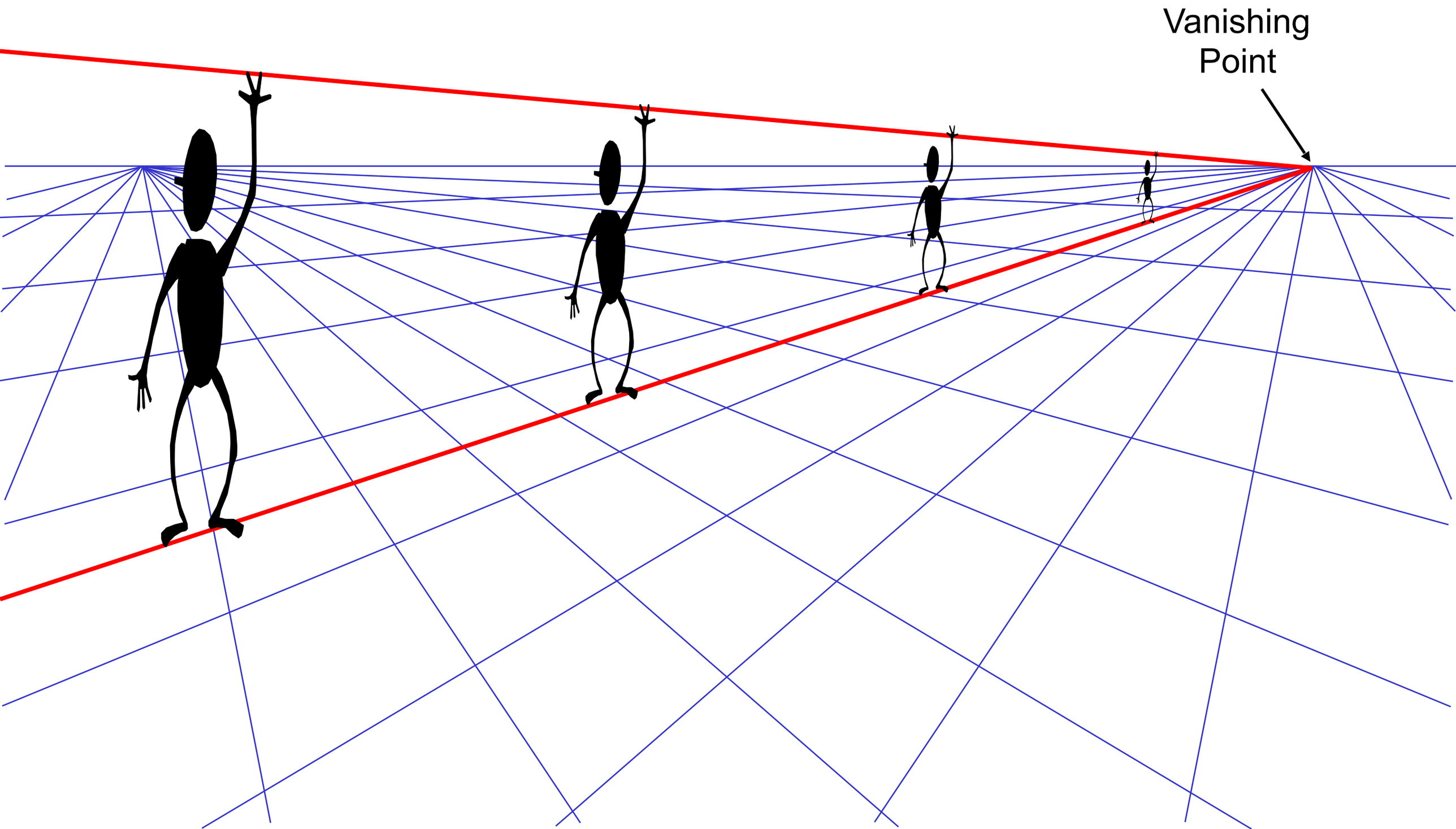
The object closer to the horizon is perceived as farther away, and the object further from the horizon is perceived as closer

Comparing heights

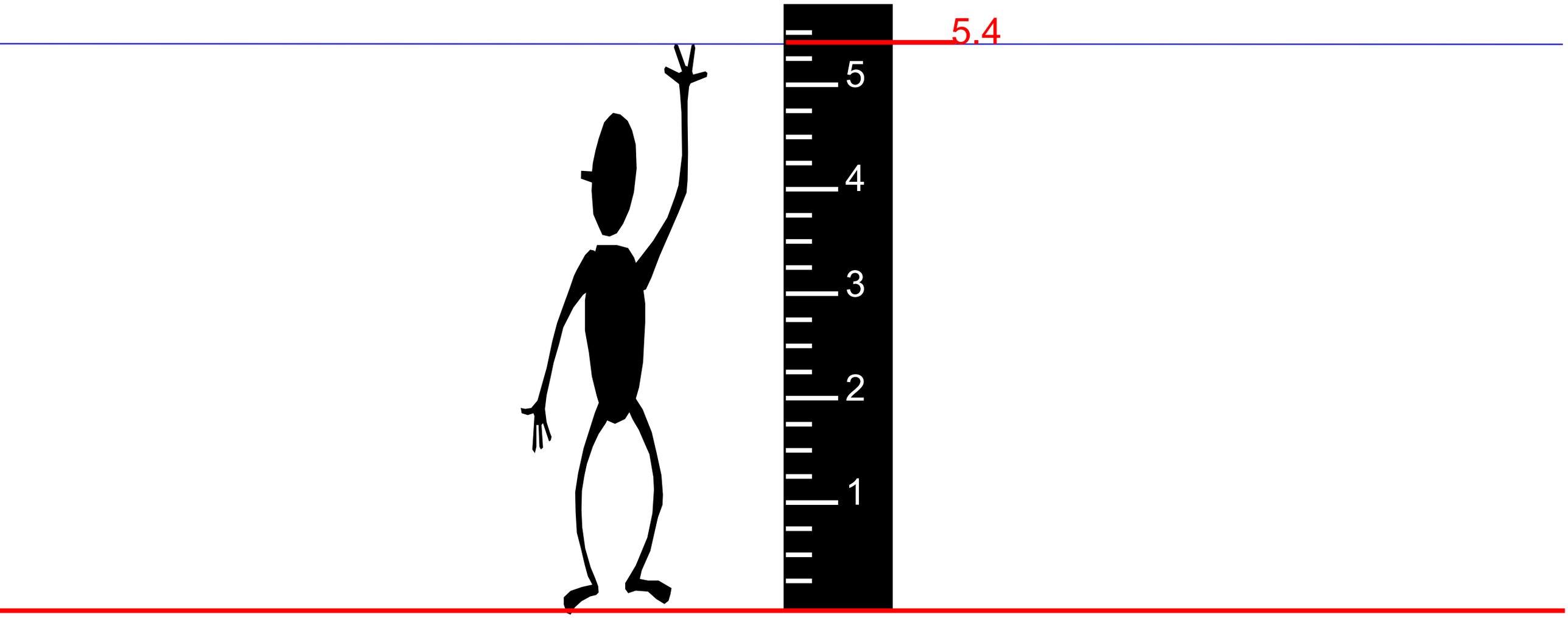


Is the person tiny or far away?

Comparing heights



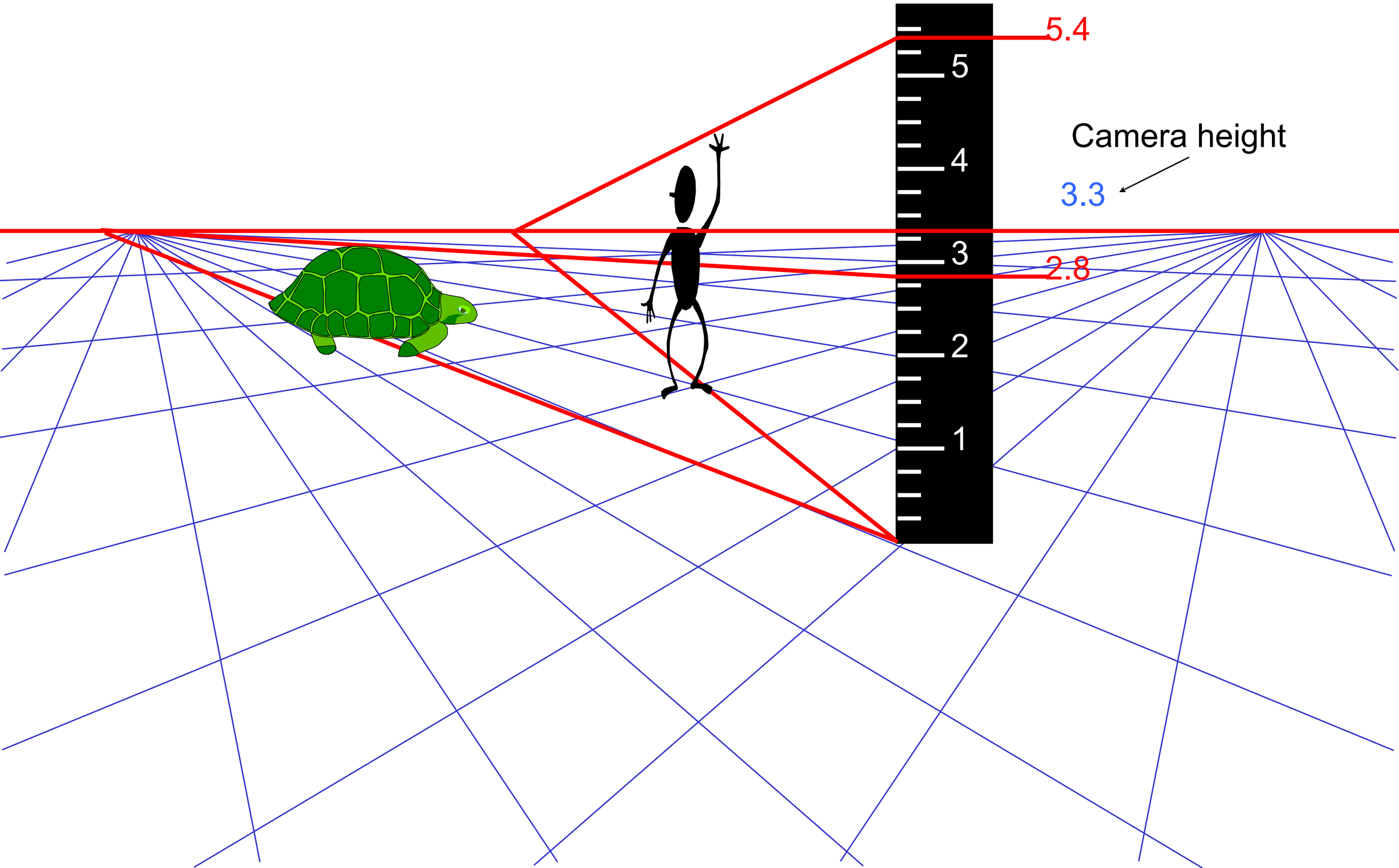
Measuring height



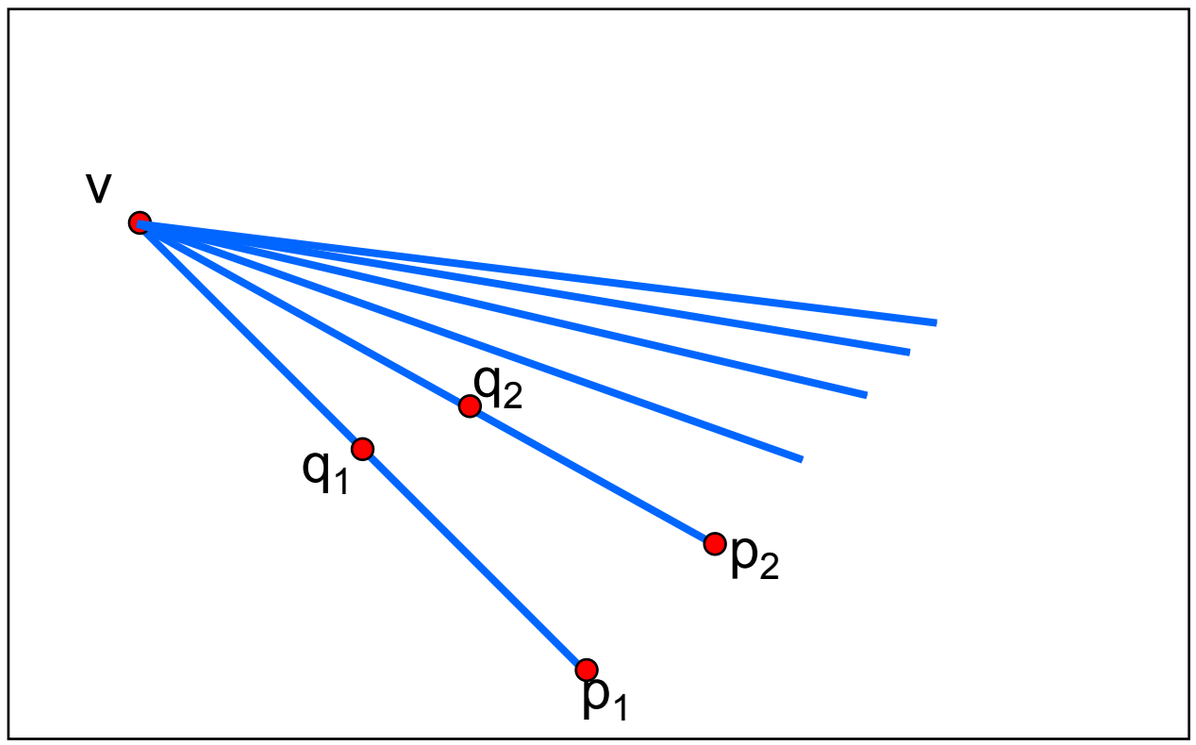
What if you can not place the ruler near the object and you have perspective projection?

If you know camera parameters: height of the camera, then we know real depth

Measuring height



Computing vanishing points (from lines)



Get lines

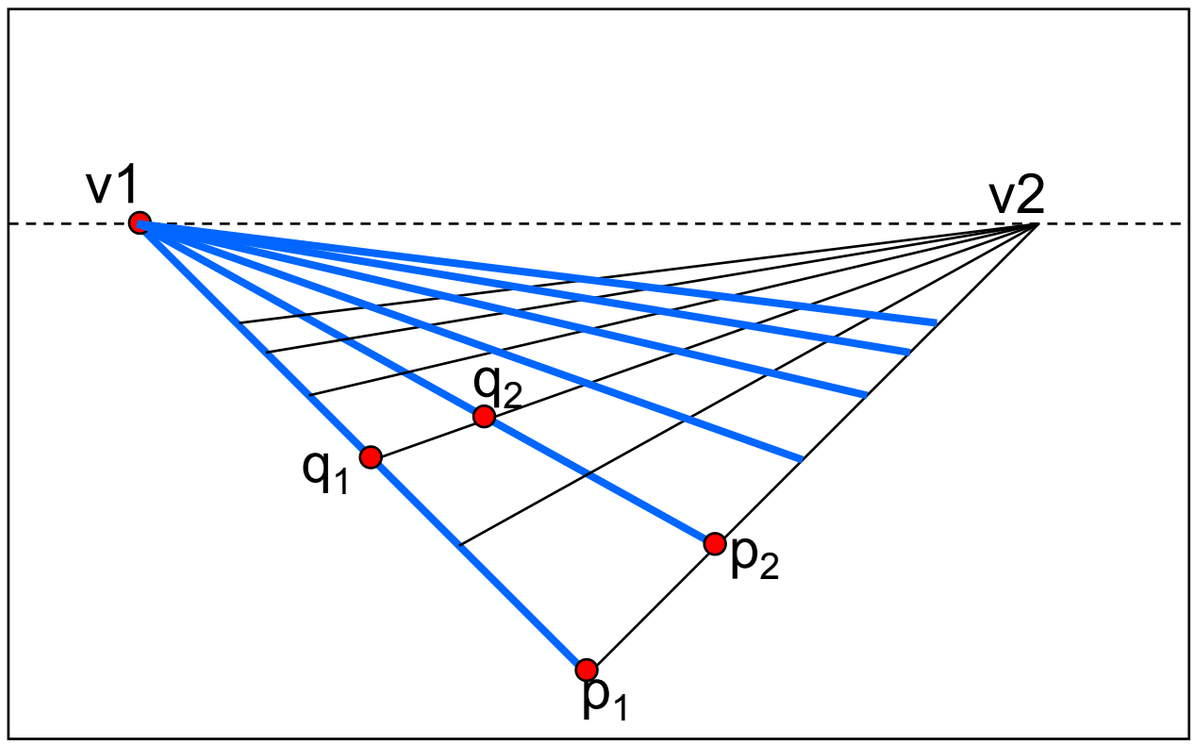
Line p1q1: $(p_1 \times q_1)$

Line p2q2: $(p_2 \times q_2)$

Intersect p_1q_1 with p_2q_2

$$v = (p_1 \times q_1) \times (p_2 \times q_2)$$

Computing the horizon line



Get two vanishing points

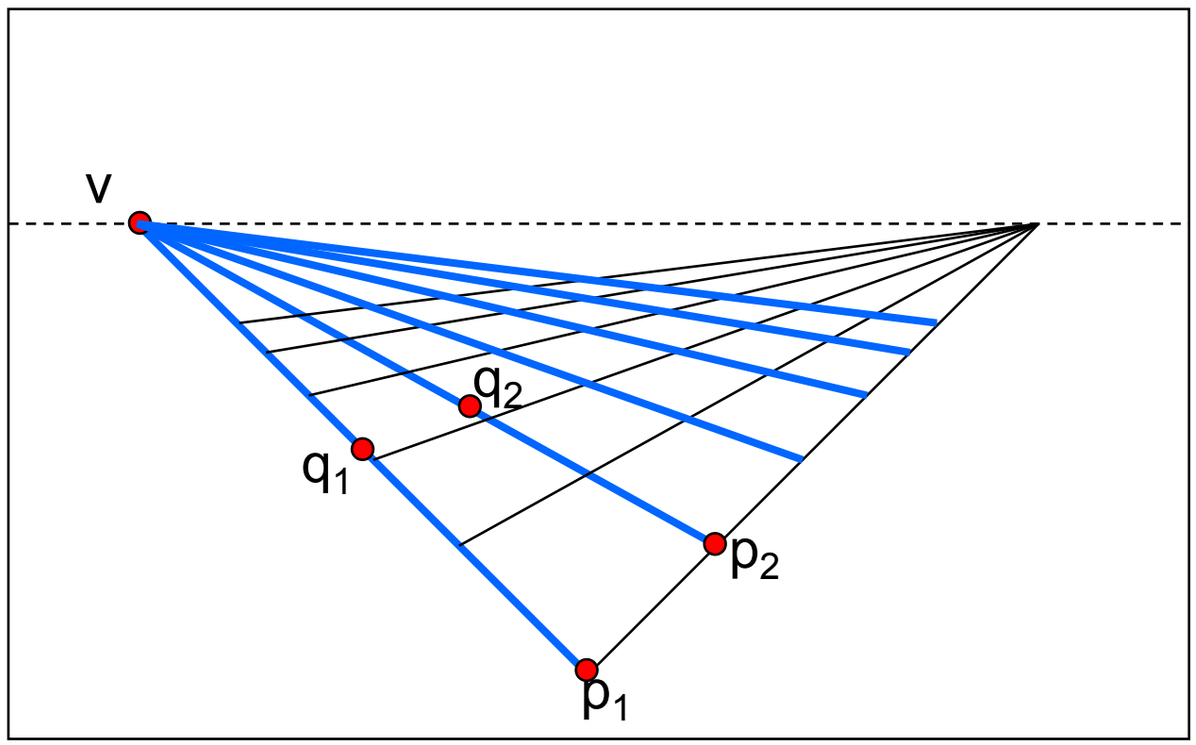
$$v_1 = (p_1 \times q_1) \times (p_2 \times q_2)$$

$$v_2 = \ldots$$

Get the line connecting both vanishing points

$$h = (v_1 \times v_2)$$

Computing vanishing points (from lines)

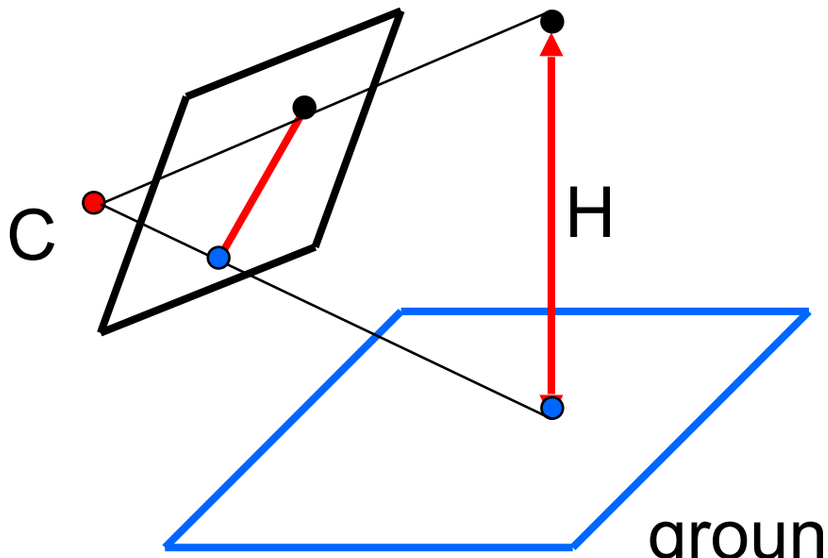


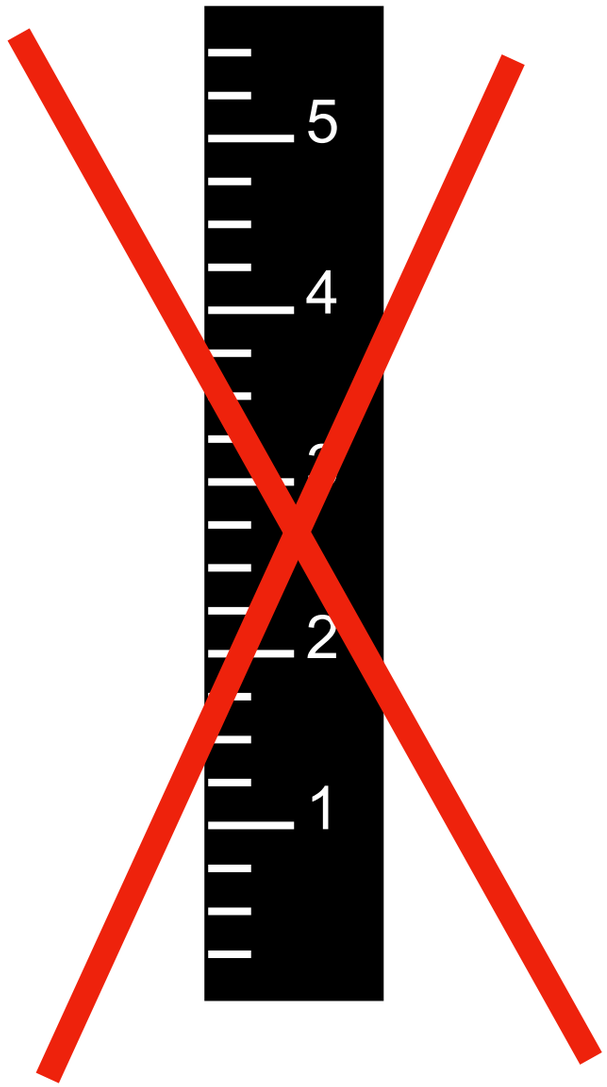
Least squares version

- Better to use more than two lines and compute the “closest” point of intersection
- See notes by Bob Collins for one good way of doing this:
  - http://www-2.cs.cmu.edu/~ph/869/www/notes/vanishing.txt

At which elevation has been taken this picture?

Measuring height without a ruler



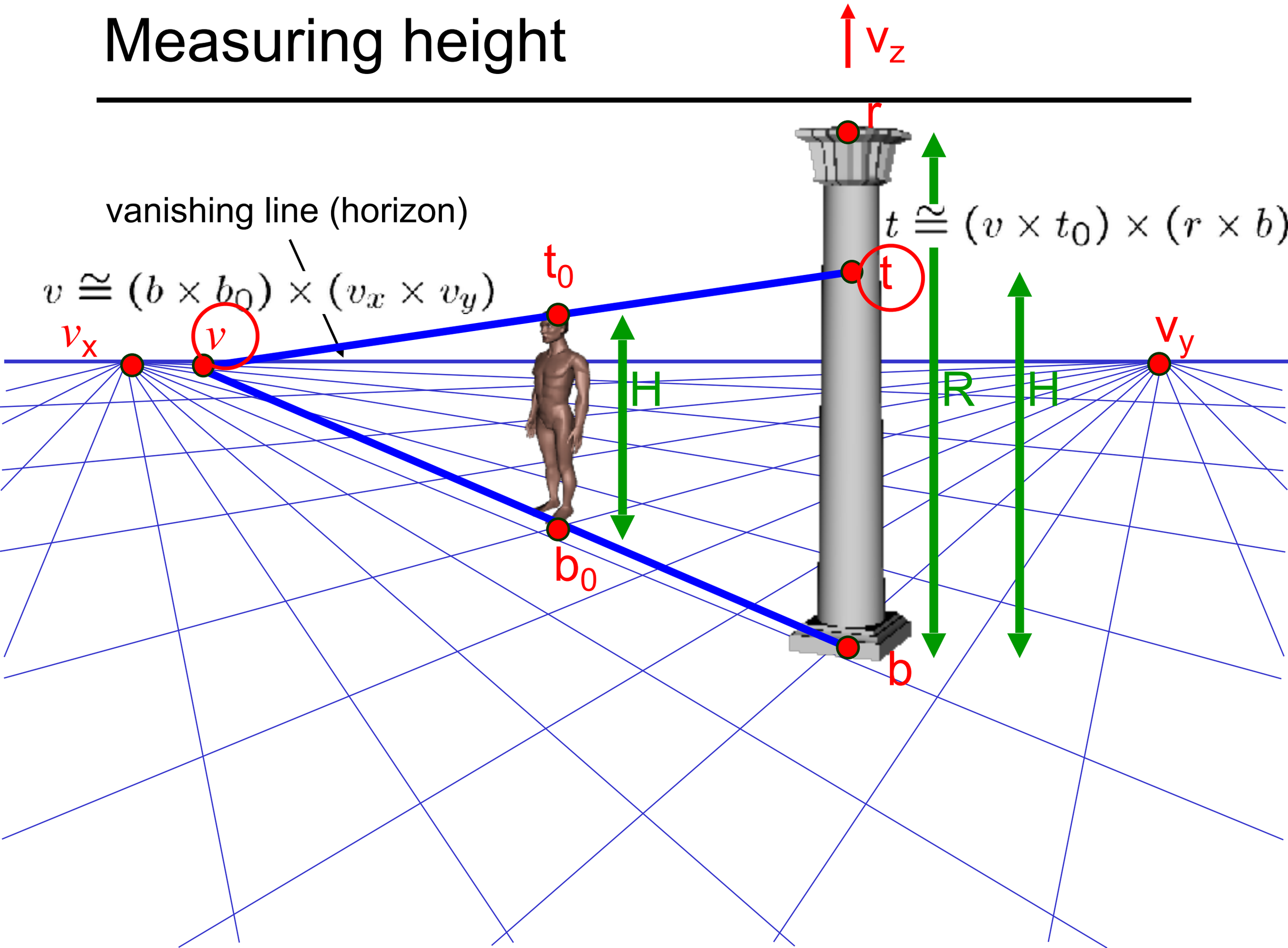


Compute H from image measurements

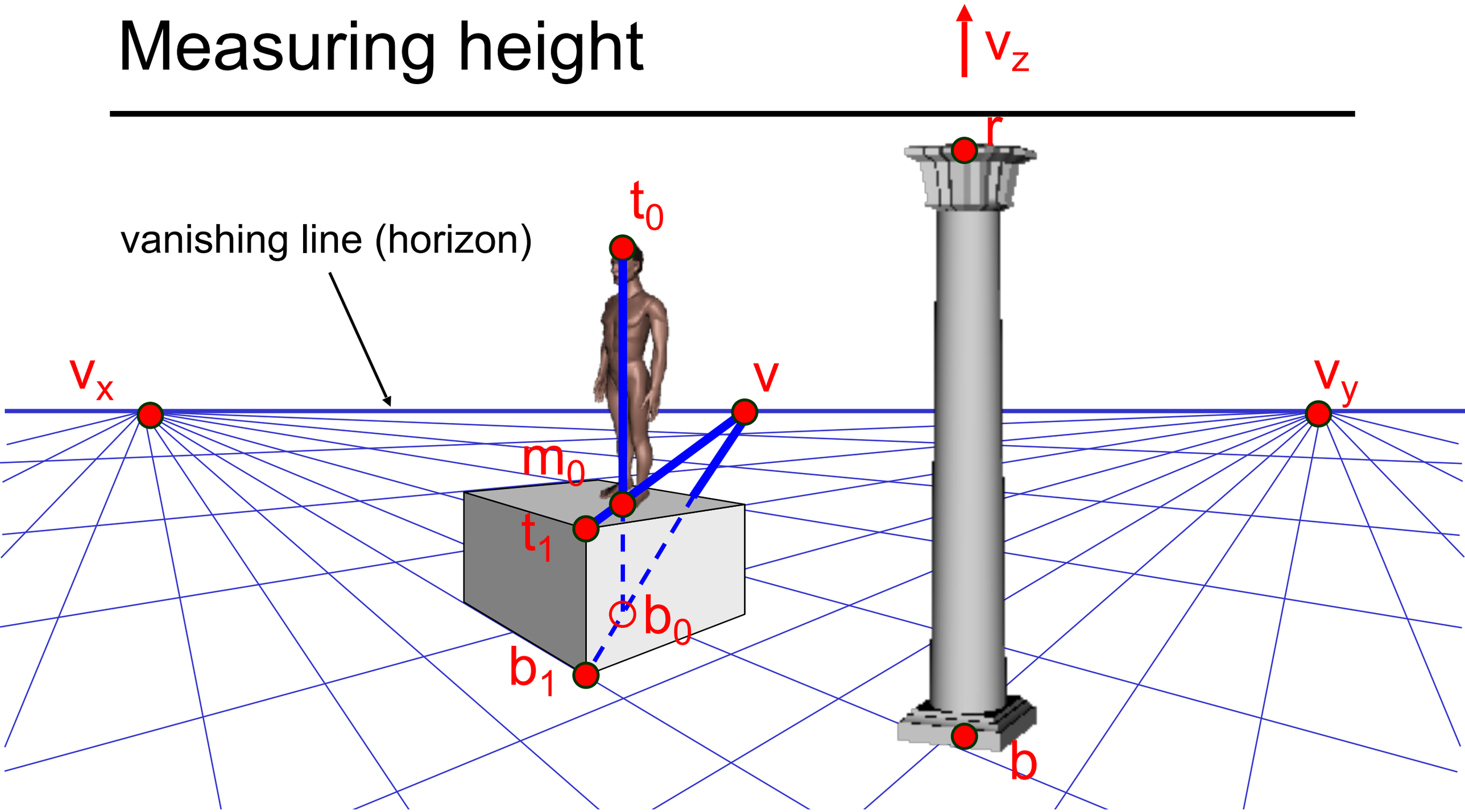
- Need more than vanishing points to do this

At which elevation has been taken this picture?

Measuring height



Measuring height



What if the point on the ground plane b_0 is not known?

- Here the guy is standing on the box
- Use one side of the box to help find b_0 as shown above

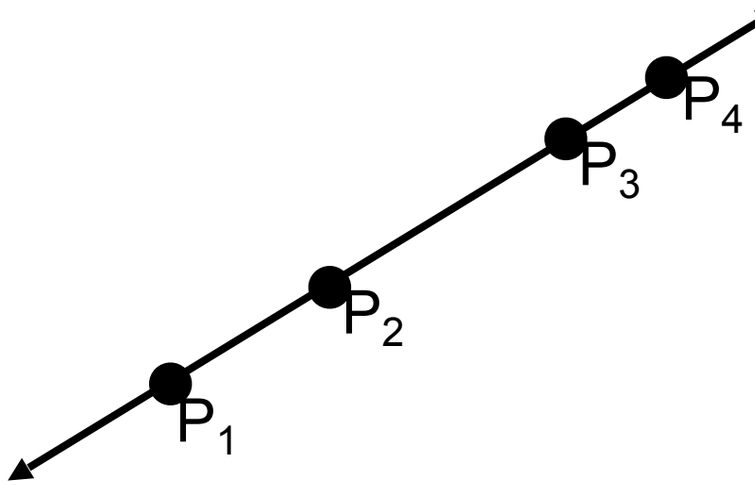
What if v_z is not infinity?

The cross ratio

A Projective Invariant

- Something that does not change under projective transformations (including perspective projection)

The cross-ratio of 4 collinear points



$$\frac{\|\mathbf{P}_3 - \mathbf{P}_1\| \, \|\mathbf{P}_4 - \mathbf{P}_2\|}{\|\mathbf{P}_3 - \mathbf{P}_2\| \, \|\mathbf{P}_4 - \mathbf{P}_1\|}$$

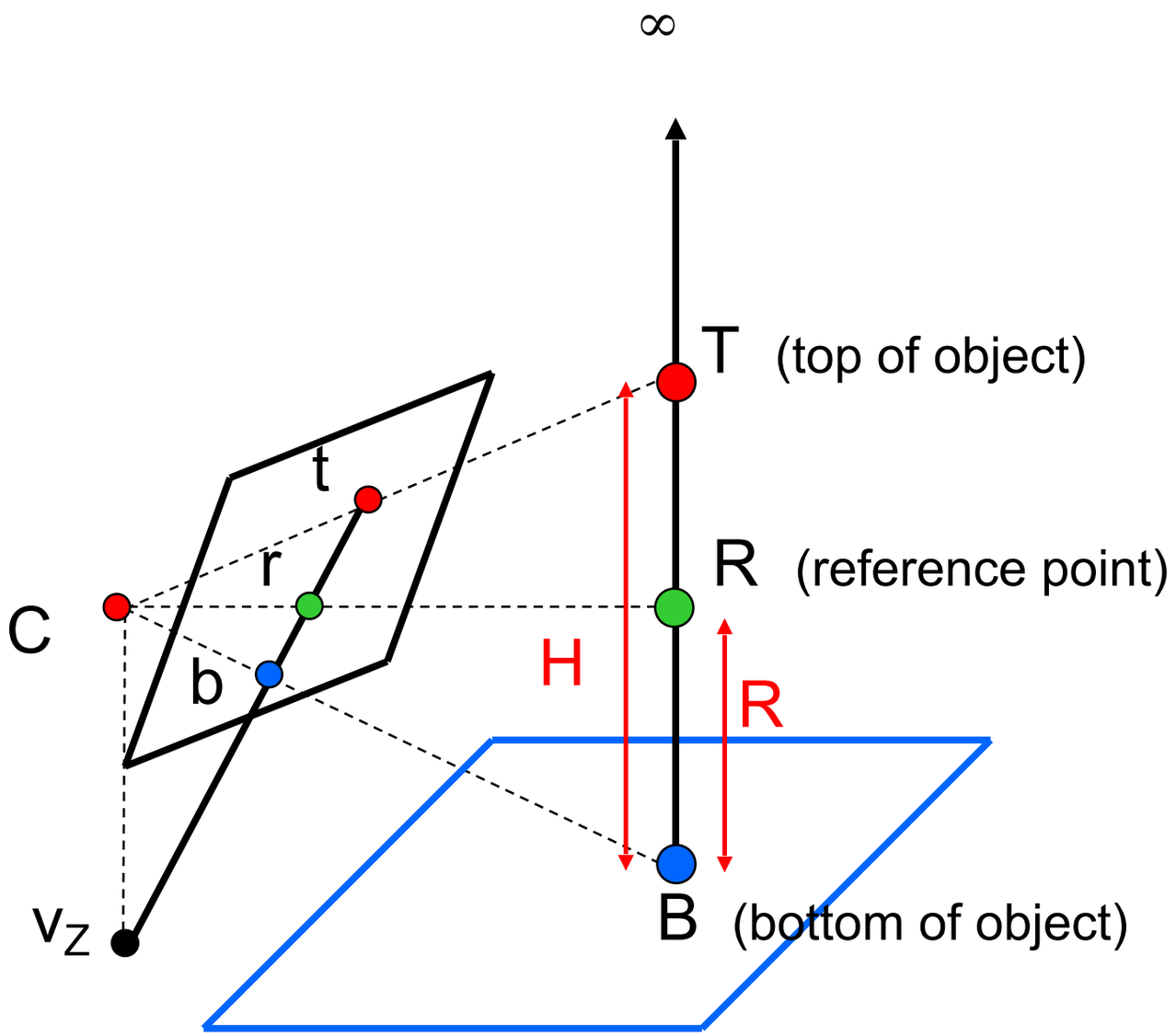
$$\mathbf{P}_i = \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

Can permute the point ordering $\frac{\|\mathbf{P}_1 - \mathbf{P}_3\| \, \|\mathbf{P}_4 - \mathbf{P}_2\|}{\|\mathbf{P}_1 - \mathbf{P}_2\| \, \|\mathbf{P}_4 - \mathbf{P}_3\|}$

- 4! = 24 different orders (but only 6 distinct values)

This is the fundamental invariant of projective geometry

Measuring height



$$\frac{\|\mathbf{T}-\mathbf{B}\|\,\|\infty-\mathbf{R}\|}{\|\mathbf{R}-\mathbf{B}\|\,\|\infty-\mathbf{T}\|}=\frac{H}{R}$$

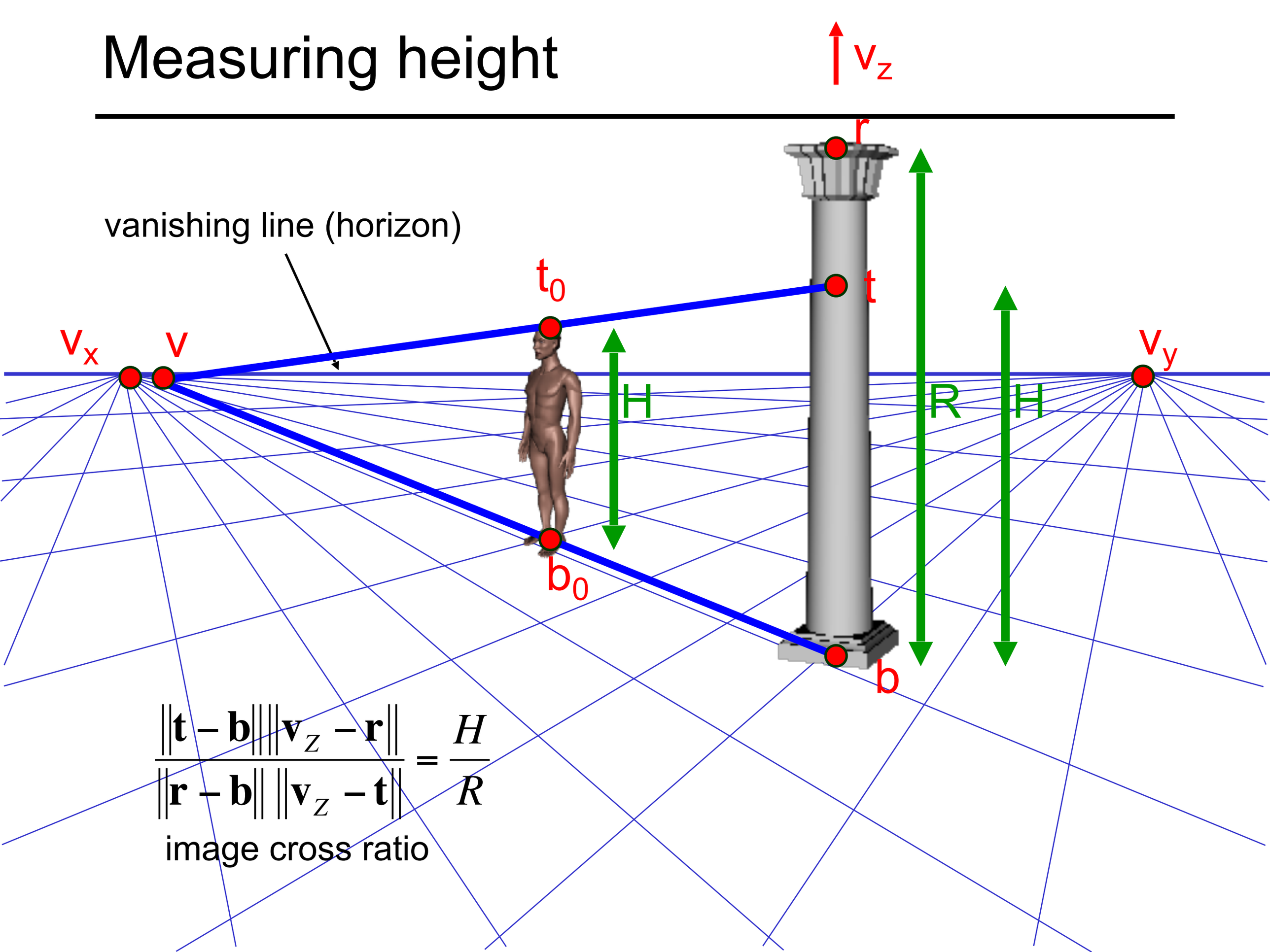
scene cross ratio

$$\frac{\|\mathbf{t}-\mathbf{b}\|\,\|\mathbf{v}_Z-\mathbf{r}\|}{\|\mathbf{r}-\mathbf{b}\|\,\|\mathbf{v}_Z-\mathbf{t}\|}=\frac{H}{R}$$

image cross ratio

scene points represented as $\mathbf{P}=\begin{bmatrix}X\\Y\\Z\\1\end{bmatrix}$ image points as $\mathbf{p}=\begin{bmatrix}x\\y\\1\end{bmatrix}$

Measuring height



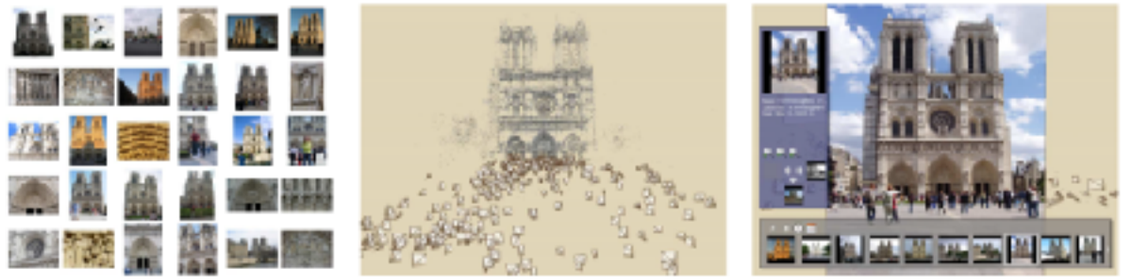
$$\frac{\|\mathbf{t}-\mathbf{b}\|\|\mathbf{v}_Z-\mathbf{r}\|}{\|\mathbf{r}-\mathbf{b}\|\,\|\mathbf{v}_Z-\mathbf{t}\|}=\frac{H}{R}$$

image cross ratio

Feature Detection and Matching

Uses in computer vision and graphics (just few of them):

Image alignment and building panoramas



3D reconstruction (Structure From Motion — SFM)

- Camera calibration
- Tracking
- Object recognition
- Indexing and database retrieval
- Video stabilization
- … other

Feature Detection and Matching

Building a Panorama

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

Building a Panorama

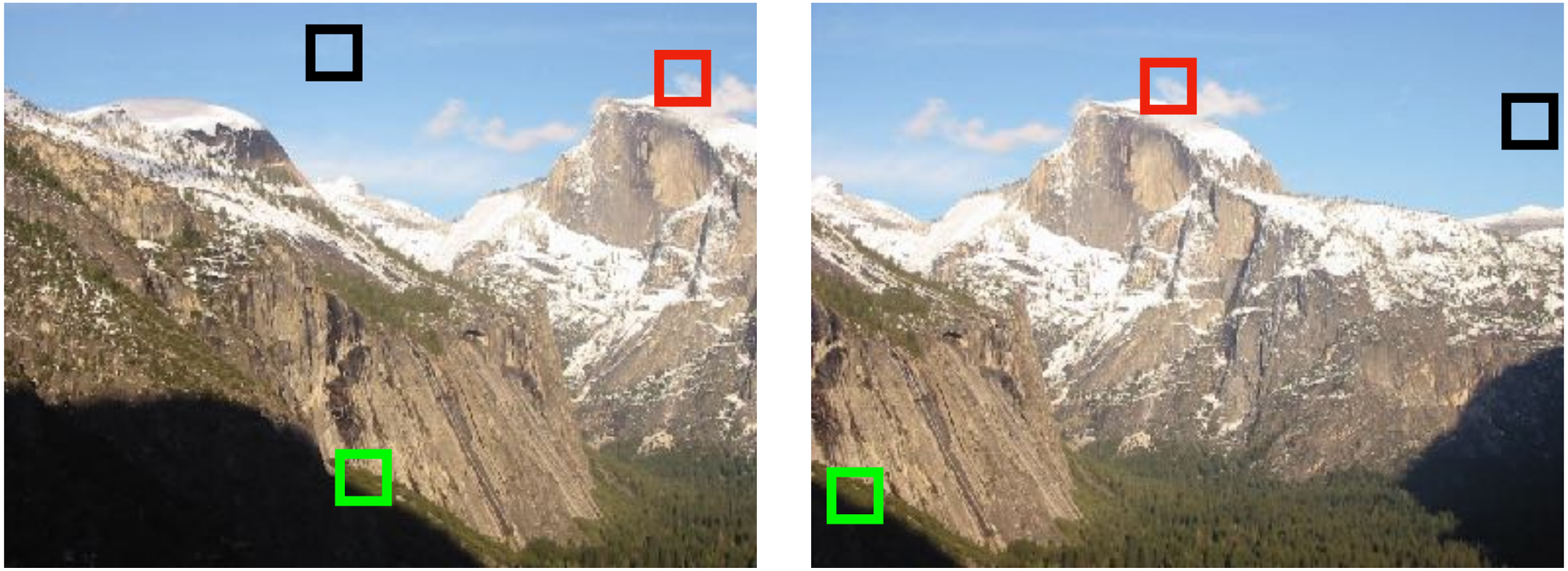
Stitching a pair of images:

- Extract a sparse set of well-localized and distinctive features in each image
- Reliably match features
- Filter matches to find inliers matches that are geometrically consistent
- Align the images and blend

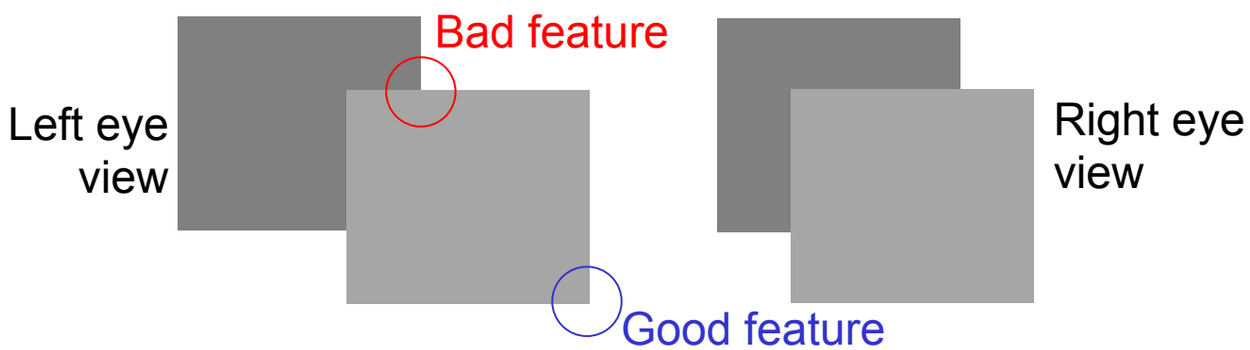
M. Brown and D. G. Lowe. Recognising Panoramas. ICCV 2003

Automatic creation of a panorama from unordered set of images

What's a "good feature"?

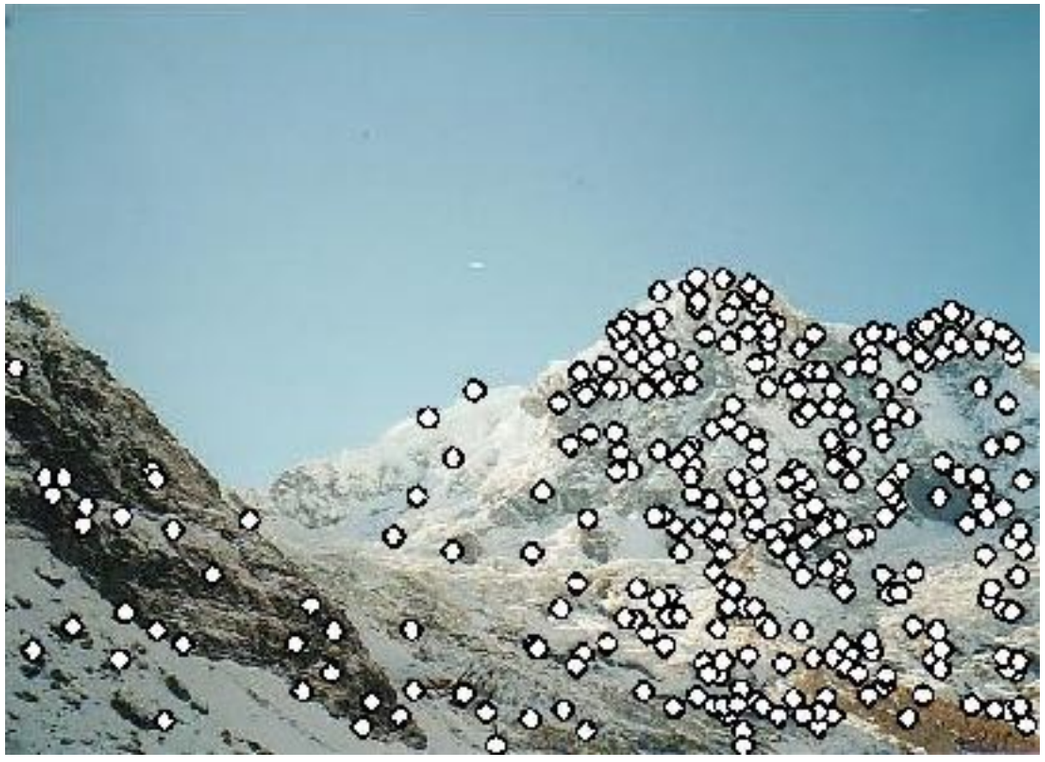


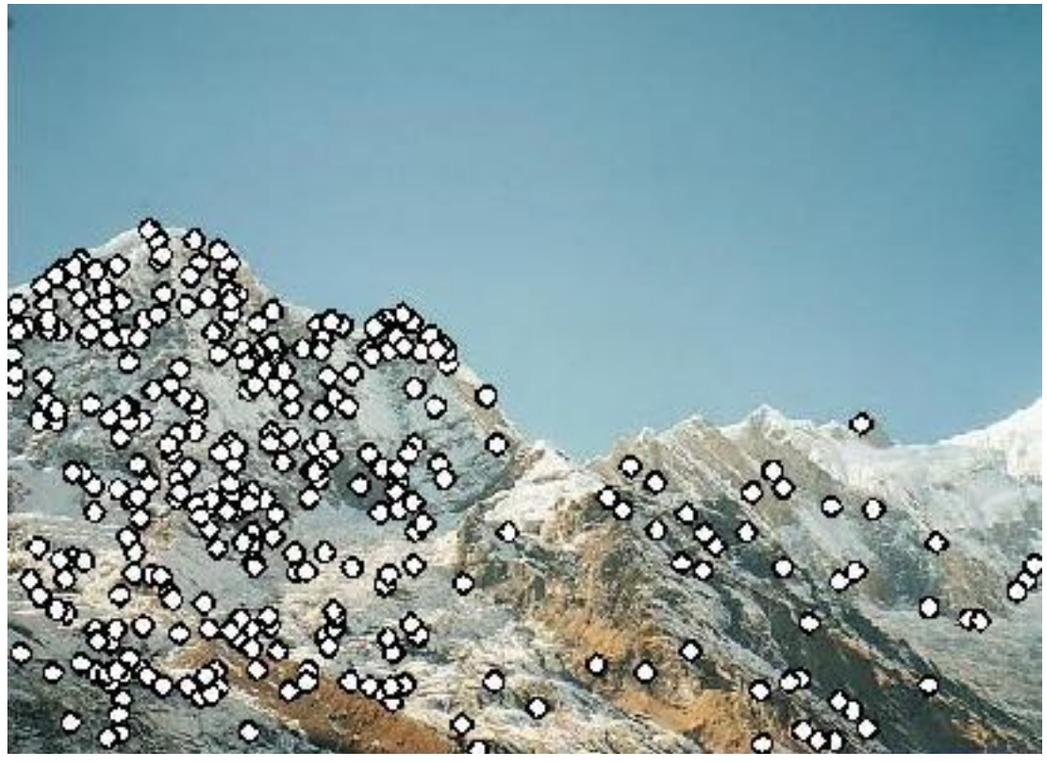
- Looks the same in both images (satisfies brightness constancy)
- Has sufficient texture variation
- Does not deform too much over time
- Corresponds to "real" surface patch



Stitching a pair of image

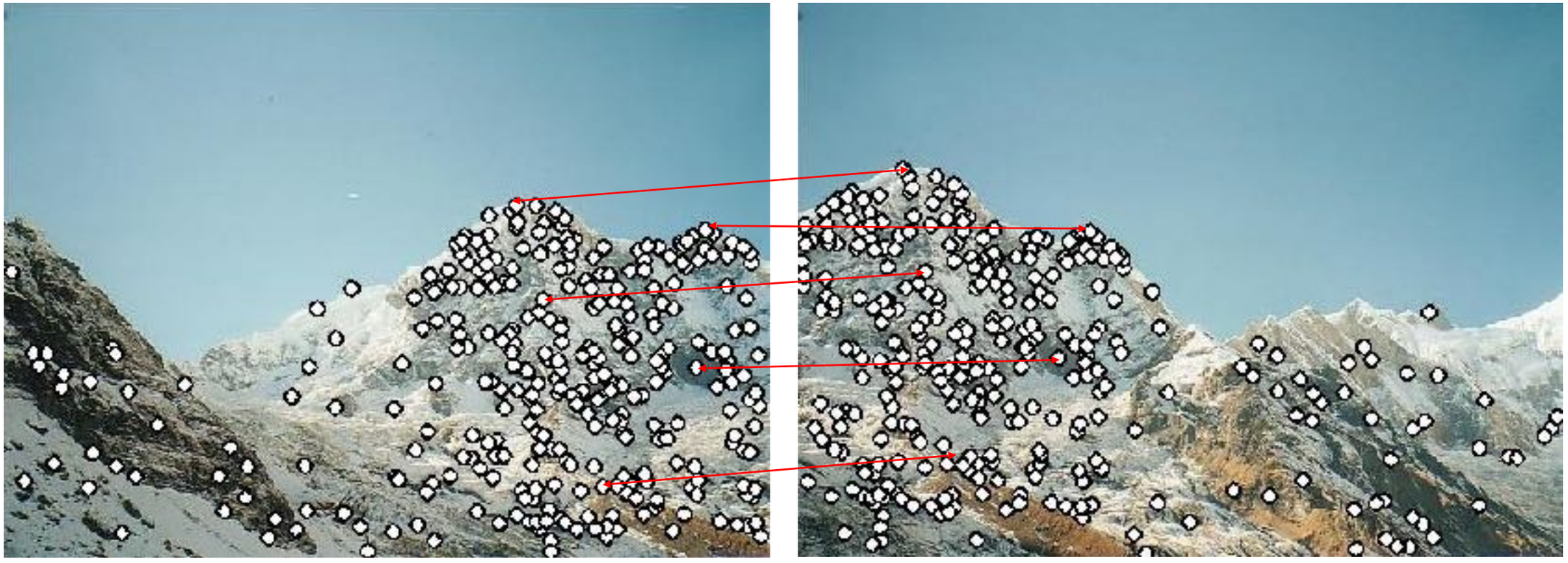
- Detect a sparse set of well-localized and distinctive features in each image





Stitching a pair of image

- Detect a sparse set of well-localized and distinctive features in each image
- Find corresponding pairs



Stitching a pair of image

- Detect a sparse set of well-localized and distinctive features in each image
- Find corresponding pairs
- Compute an homography and align the images
  (use RANSAC to filter out outliers and keep only matches that are geometrically consistent)

Matching with Features

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

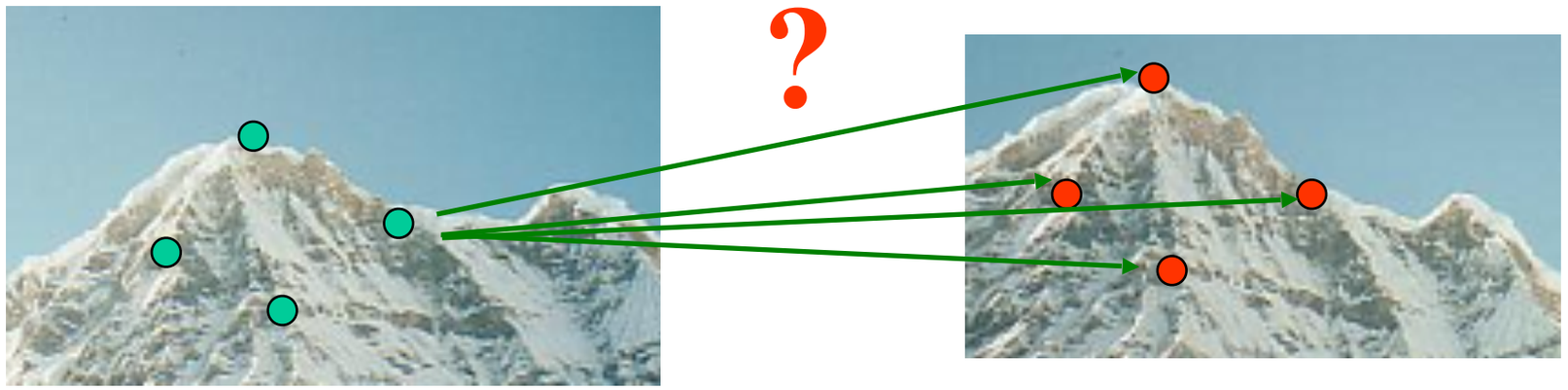
counter-example:

no chance to match!

We need a repeatable **detector**

Matching with Features

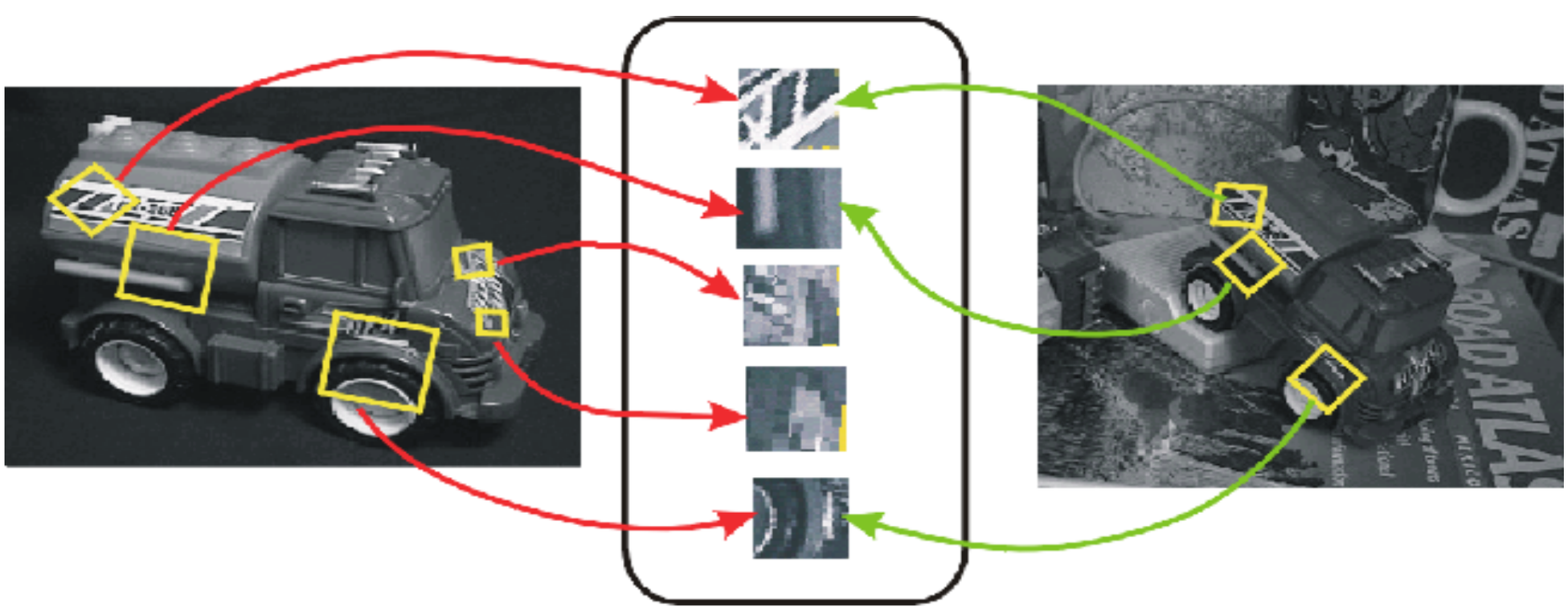
- Problem 2:
  - For each point correctly recognize the corresponding one



We need a reliable and distinctive **descriptor**

Matching with Features — Preview

- **Detector**: detect **same** scene points **independently** in both images
- **Descriptor**: encode local neighboring window
  - Note how scale & rotation of window are the same in both image (but computed independently)
- **Correspondence**: find most similar descriptor in other image



Outline

- Feature point detection
  - Harris corner detector
  - Finding a characteristic scale: DoG or Laplacian of Gaussian
- Local image description
  - SIFT features

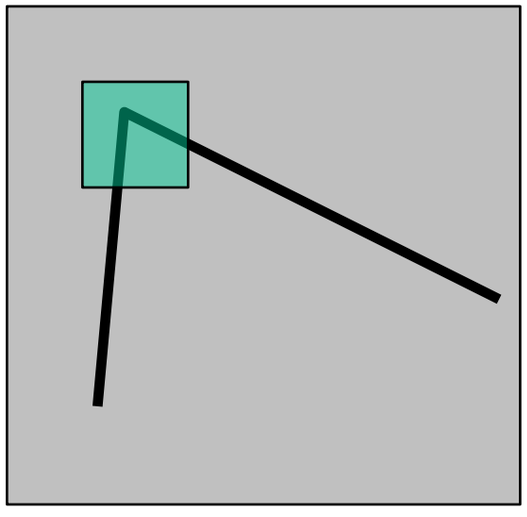
Harris Corner Detector

A COMBINED CORNER AND EDGE DETECTOR

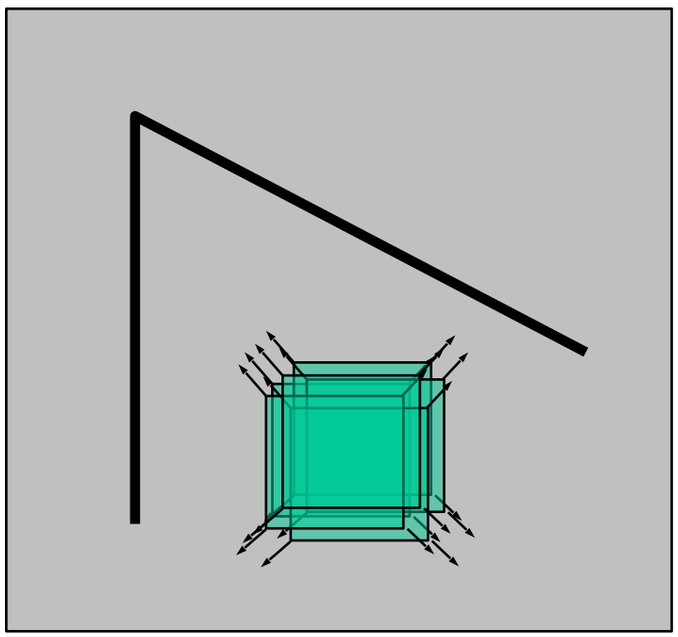
Chris Harris & Mike Stephens

Plessey Research Roke Manor, United Kingdom

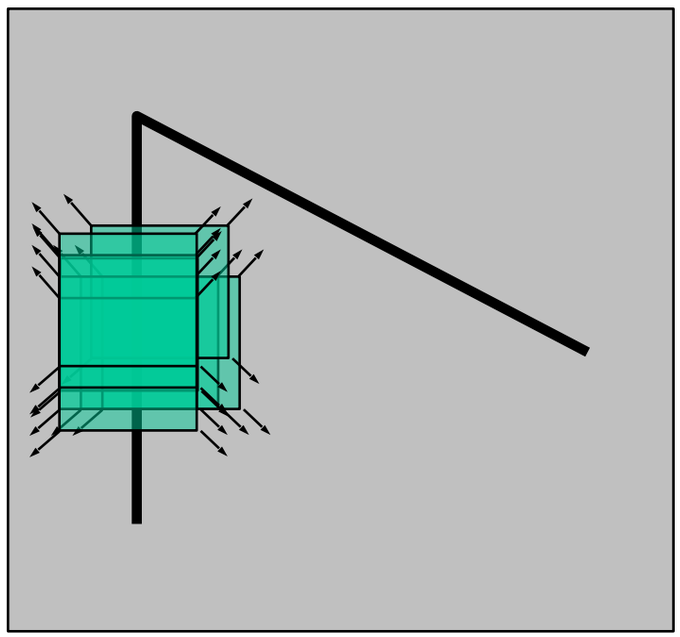
- We should easily localize the point by looking through a small window (patch)
- Shifting a window in *any direction* should give *a large change* in pixels intensities in window
  - makes location precisely define



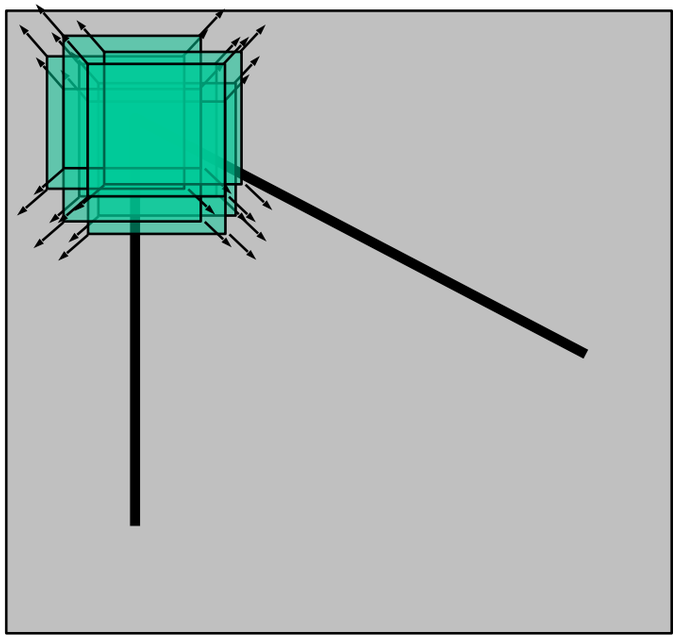
Harris Corner Detector — Basic Idea



“flat” region:
no change in all directions



“edge”:
no change along the edge direction



“corner”:
significant change in all directions

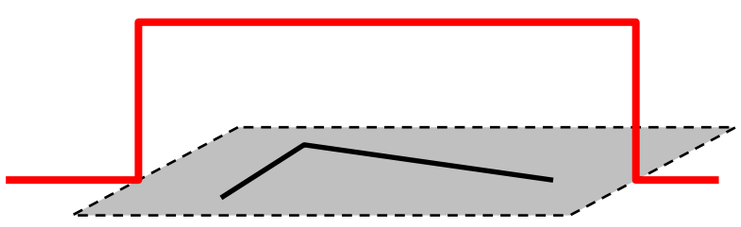
Harris Detector — Mathematics

Window-averaged squared change of intensity induced by shifting the image data by [*u,v*]:

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

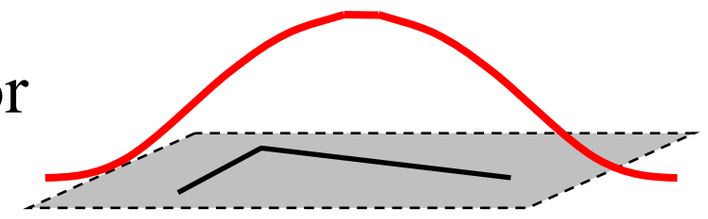
- $w(x,y)$: Window function
- $I(x+u, y+v)$: Shifted intensity
- $I(x,y)$: Intensity

Window function $w(x,y)$ =



1 in window, 0 outside

or



Gaussian

Harris Detector — Mathematics

Taylor series approximation to shifted image gives quadratic form for error as function of image shifts.

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

$$\begin{aligned} E(u,v) &\approx \sum_{x,y} w(x,y)[I(x,y) + uI_x + vI_y - I(x,y)]^2 \\ &= \sum_{x,y} w(x,y)[uI_x + vI_y]^2 \\ &= (u \quad v)\sum_{x,y} w(x,y)\begin{bmatrix} I_xI_x & I_xI_y \\ I_xI_y & I_yI_y \end{bmatrix}\begin{pmatrix} u \\ v \end{pmatrix} \end{aligned}$$

Harris Detector — Mathematics

For small shifts [*u,v*], we have a *quadratic* approximation to the error surface between a patch and itself, shifted by [*u,v*]:

$$E(u,v) \cong \begin{bmatrix} u, v \end{bmatrix} \; M \; \begin{bmatrix} u \\ v \end{bmatrix}$$

where *M* is a 2·2 matrix computed from image derivatives:

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

M is often called structure tensor

Harris Detector — Mathematics

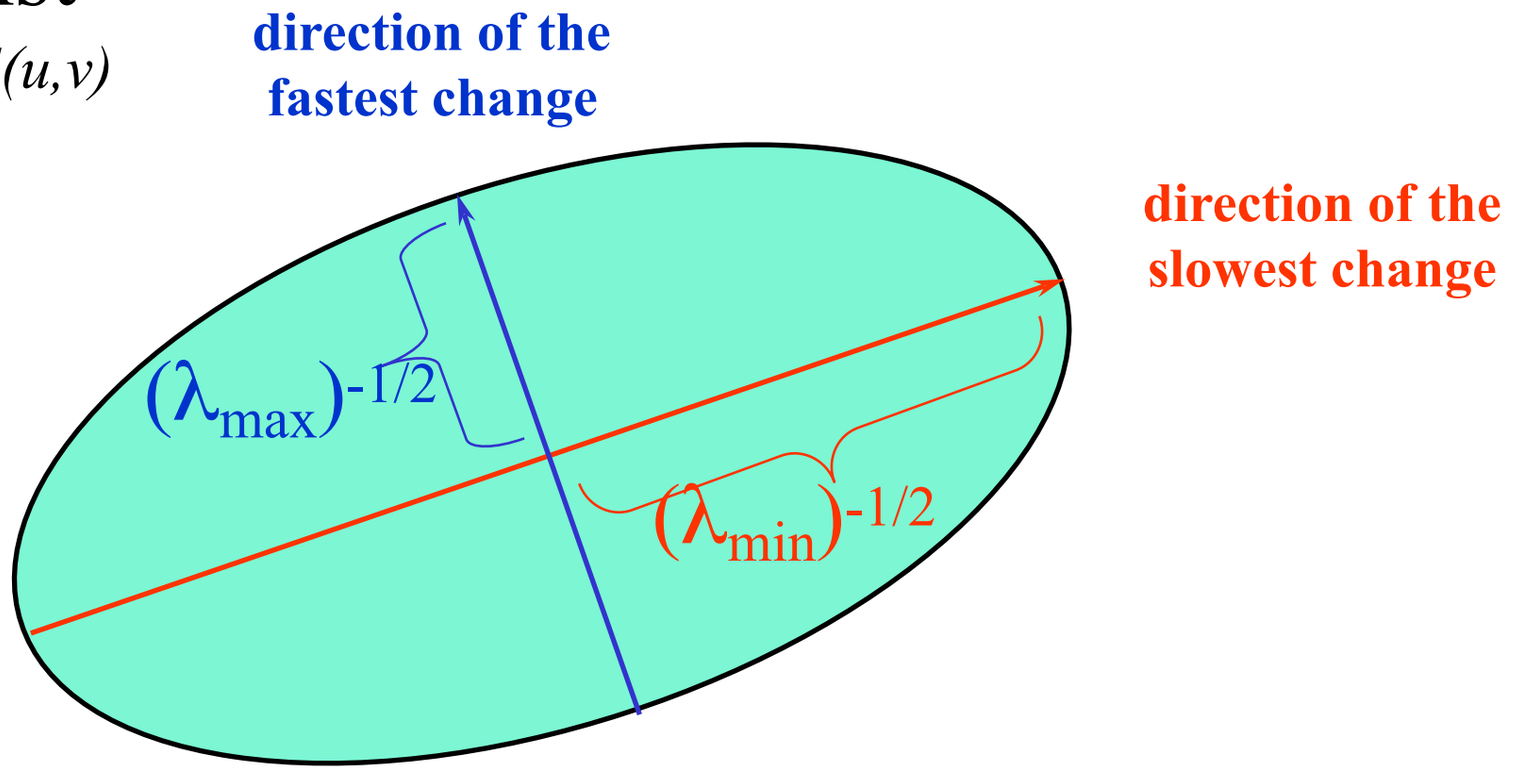
Intensity change in shifting window: eigenvalue analysis

$$E(u,v) \cong \begin{bmatrix} u, v \end{bmatrix} \; M \; \begin{bmatrix} u \\ v \end{bmatrix}$$

λ_1, λ_2 – eigenvalues of M

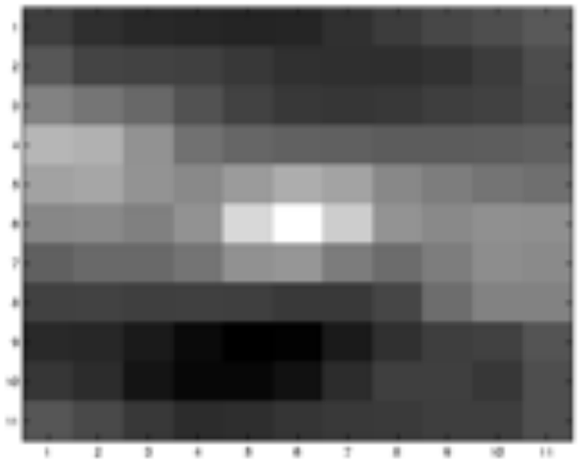
Ellipse: $E(u,v) = \text{const}$

Iso-contour of the squared error, $E(u,v)$



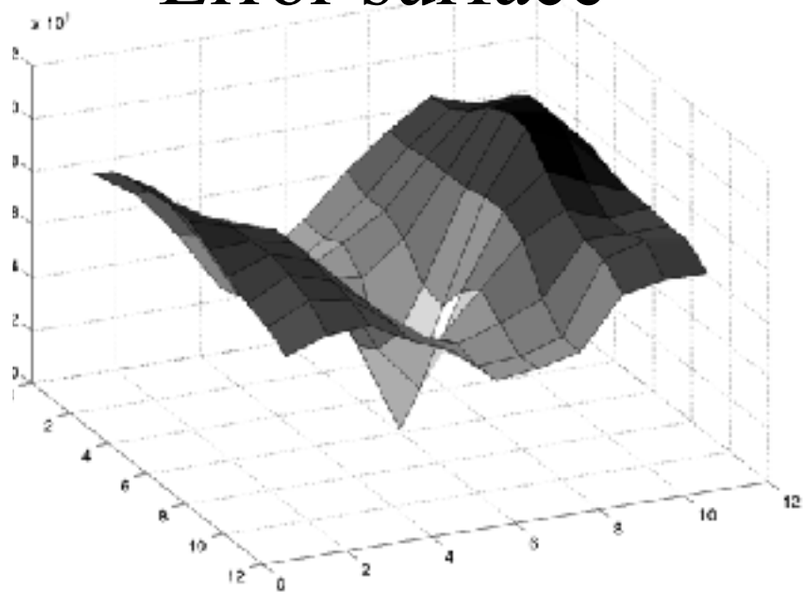
Selecting Good Features

Image patch



12x10^5

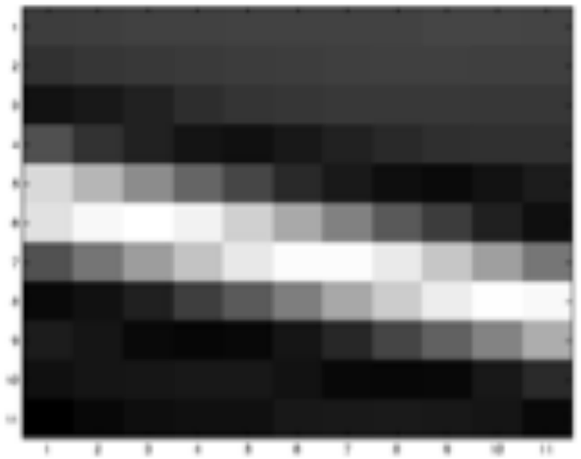
Error surface



λ_1 and λ_2 are large

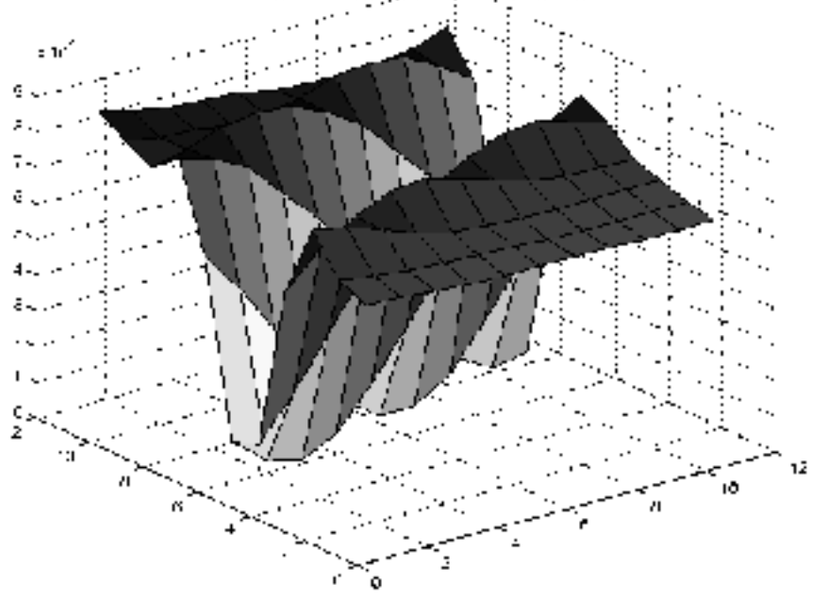
Selecting Good Features

Image patch



9x10^5

Error surface



large λ_1, small λ_2

Selecting Good Features

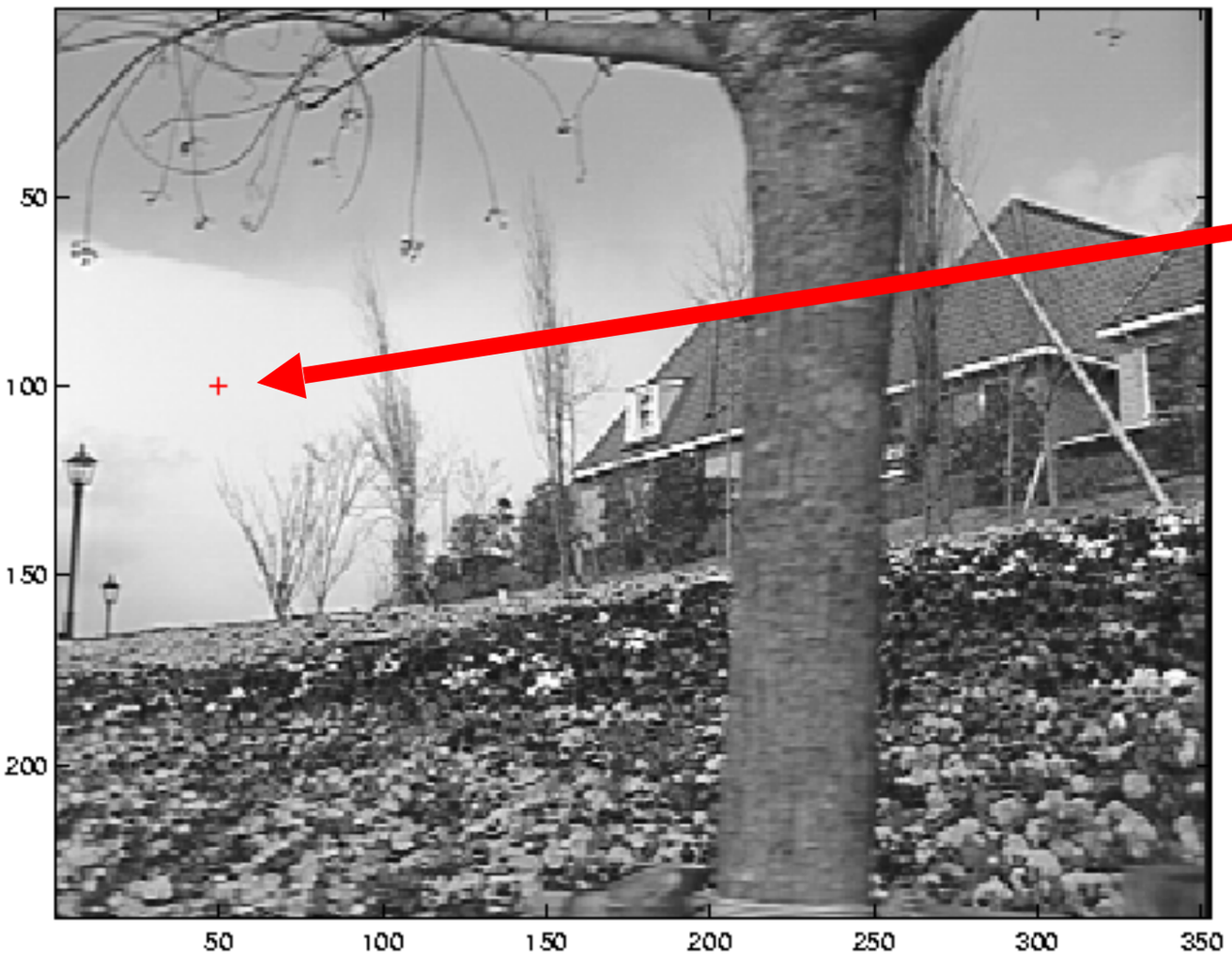
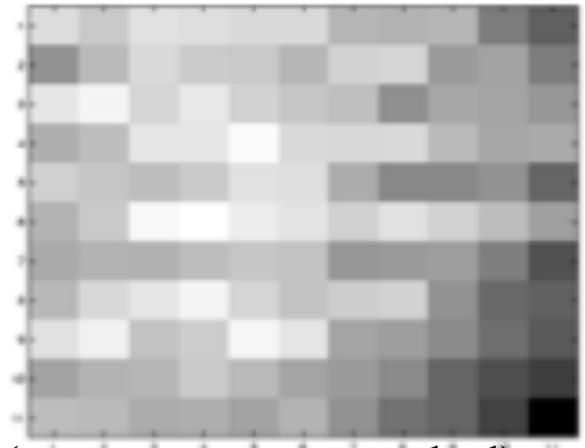
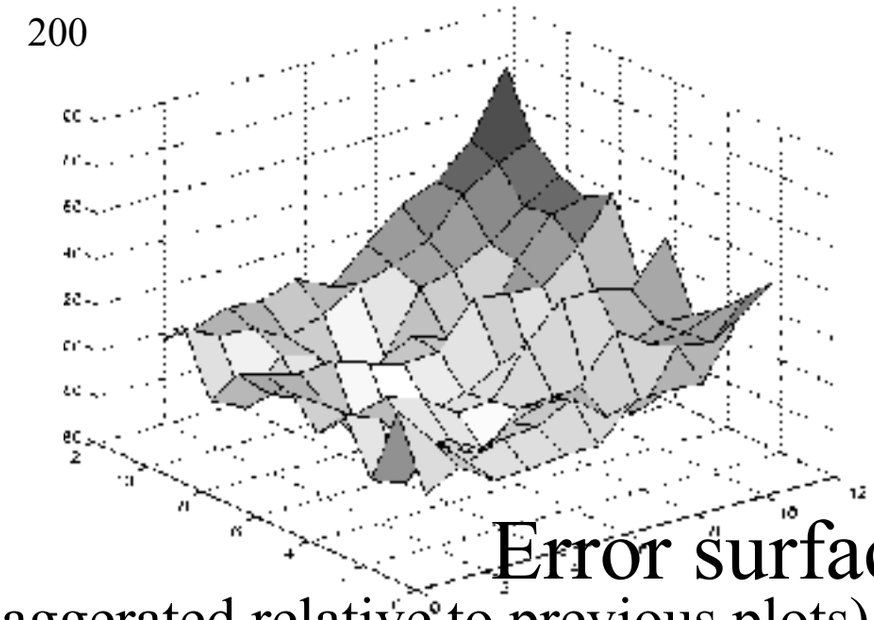


Image patch



(contrast auto-scaled)



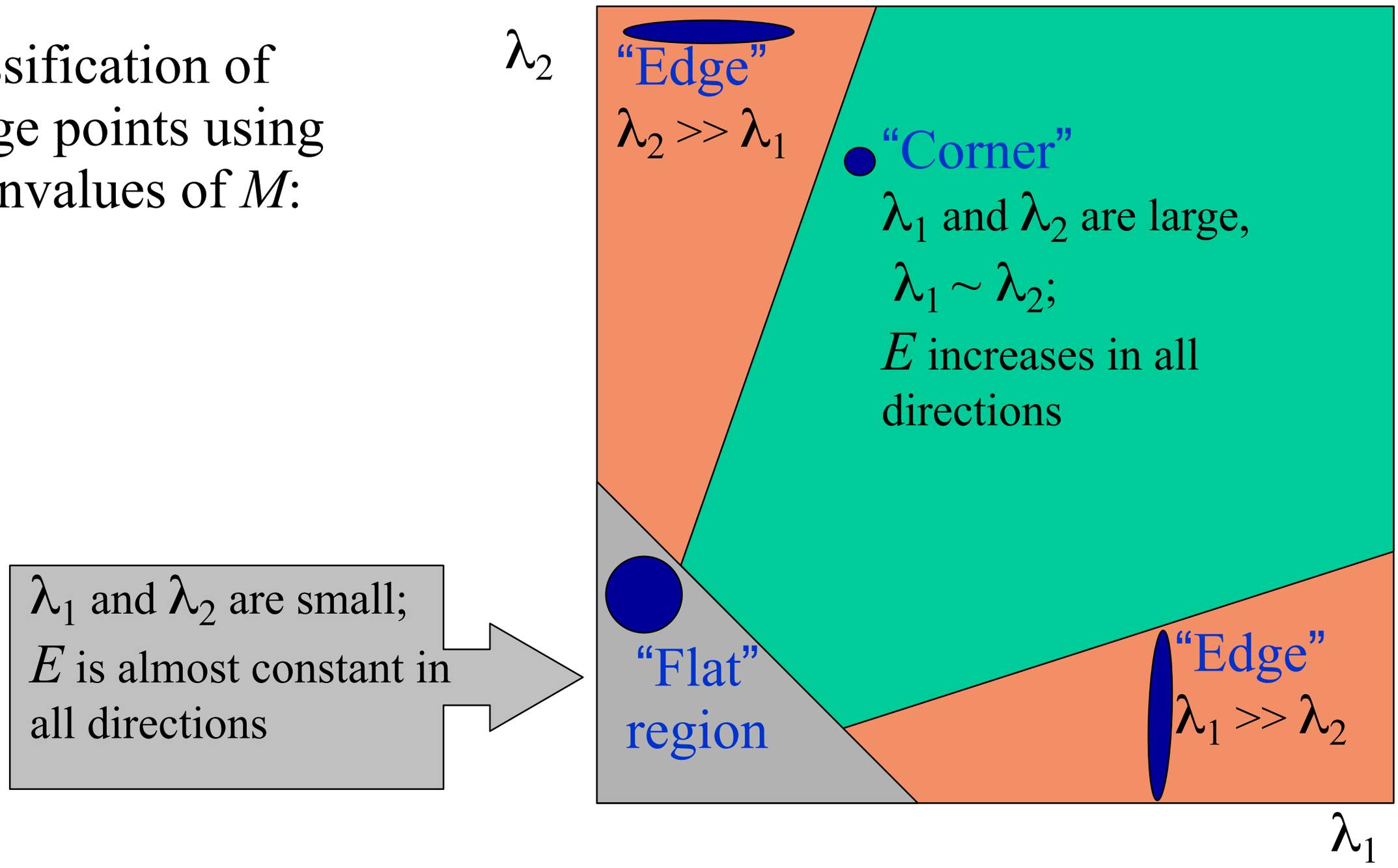
Error surface

(vertical scale exaggerated relative to previous plots)

small λ_1, small λ_2

Harris Detector: Mathematics

Classification of image points using eigenvalues of *M*:



Harris Detector: Mathematics

Measure of corner response:

$$R = \det M - k\left(\text{trace } M\right)^2$$

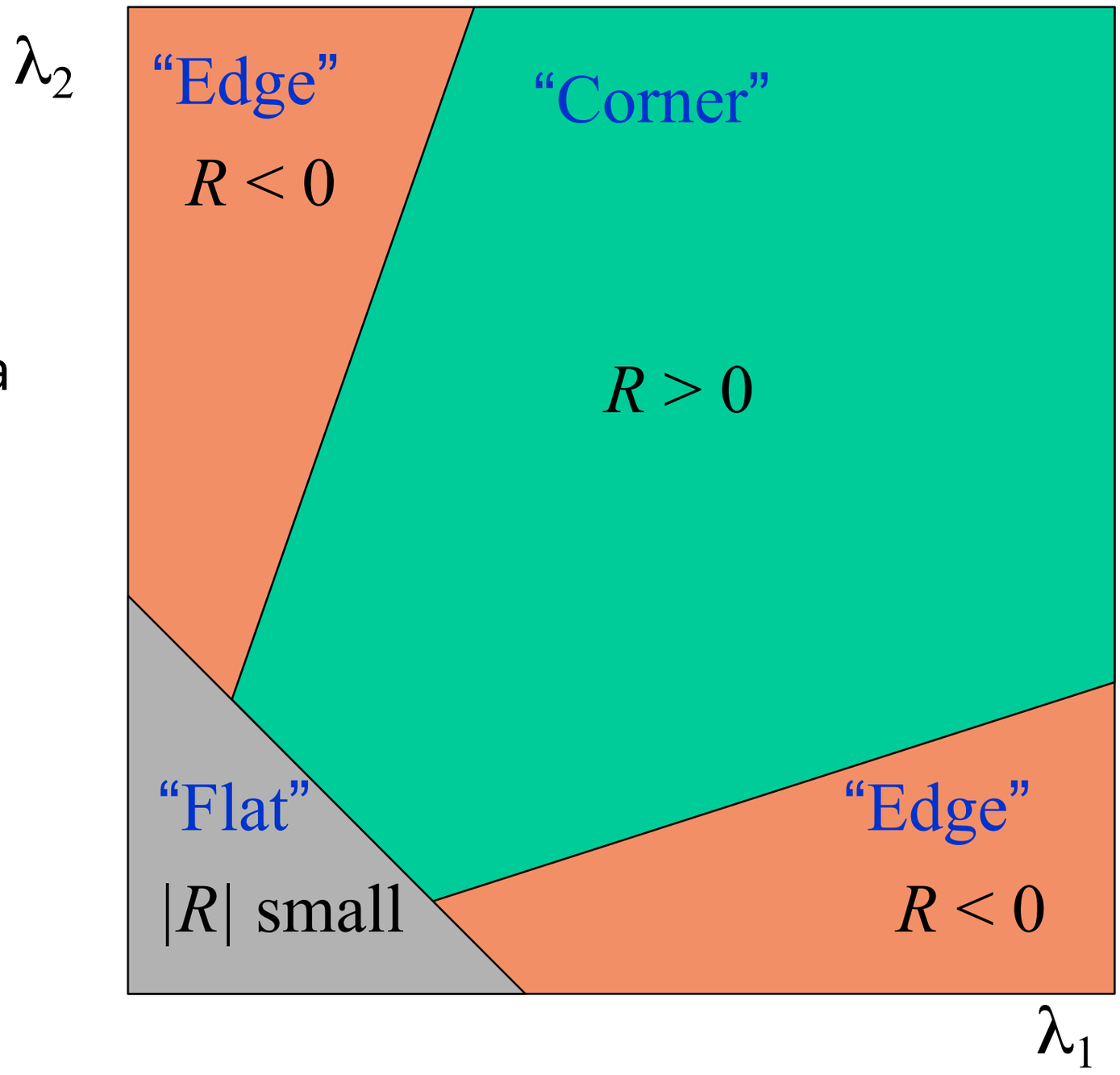
$$\det M = \lambda_1 \lambda_2$$
$$\text{trace } M = \lambda_1 + \lambda_2$$

(k – empirical constant, k = 0.04-0.06)

(Shi-Tomasi variation: use min(λ_1,λ_2) instead of R)

Harris Detector: Mathematics

- R depends only on eigenvalues of M
- R is large (positive) for a **corner**
- R is negative with large magnitude for an **edge**
- $|R|$ is small for a **flat** region



Harris Detector — Algorithm

- Compute The matrix M for each pixel:
  - Compute image gradients, I_x I_y

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

- Measure the response R for each pixel

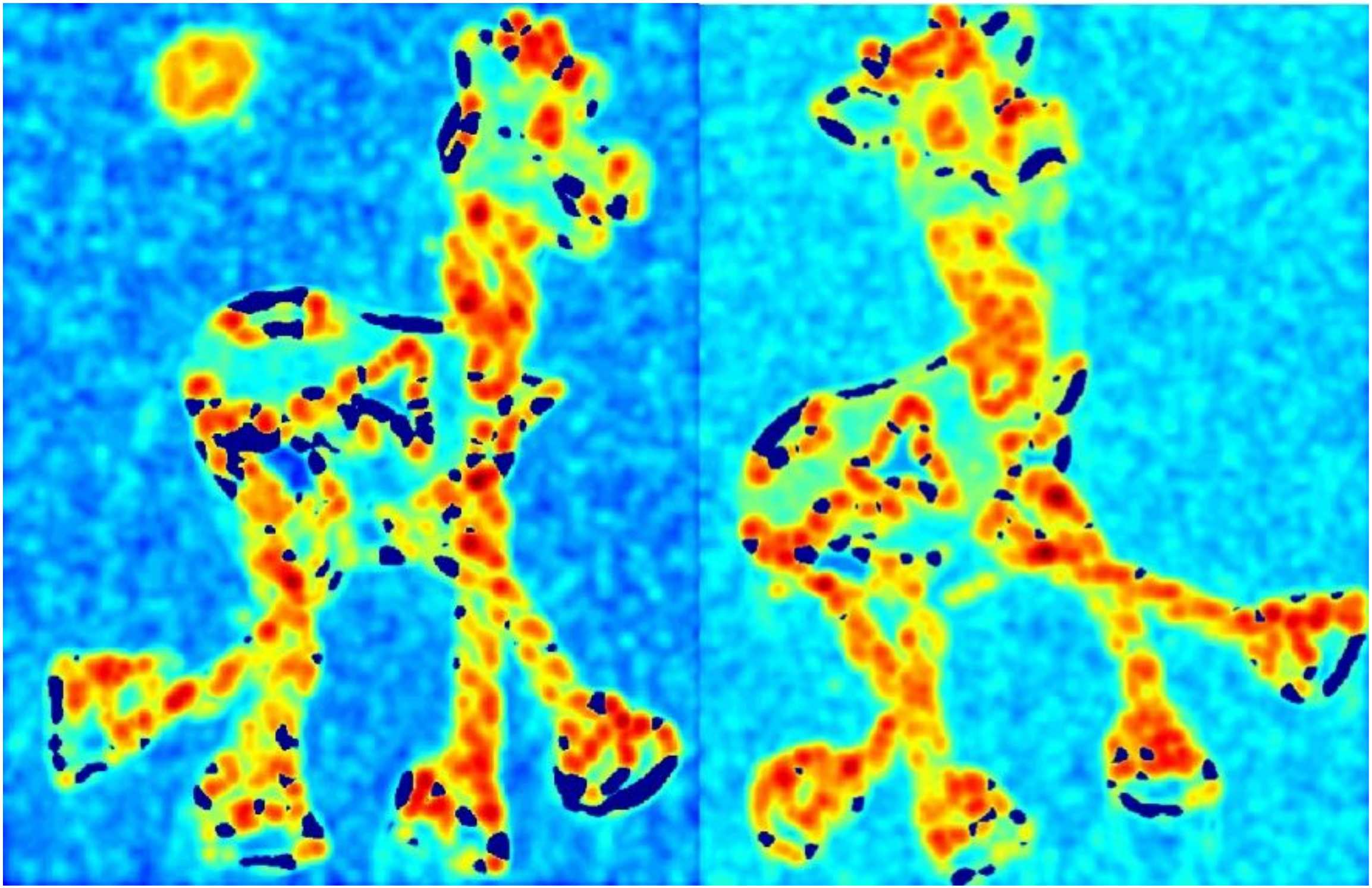
$$R = \det M - k\left(\text{trace } M\right)^2$$

- Threshold the value of R; Take points that are *local* maxima (non-max suppression, i.e, pixels where R is bigger than for all the 4 or 8 neighbors) as the detected feature points.

Harris Detector: Workflow

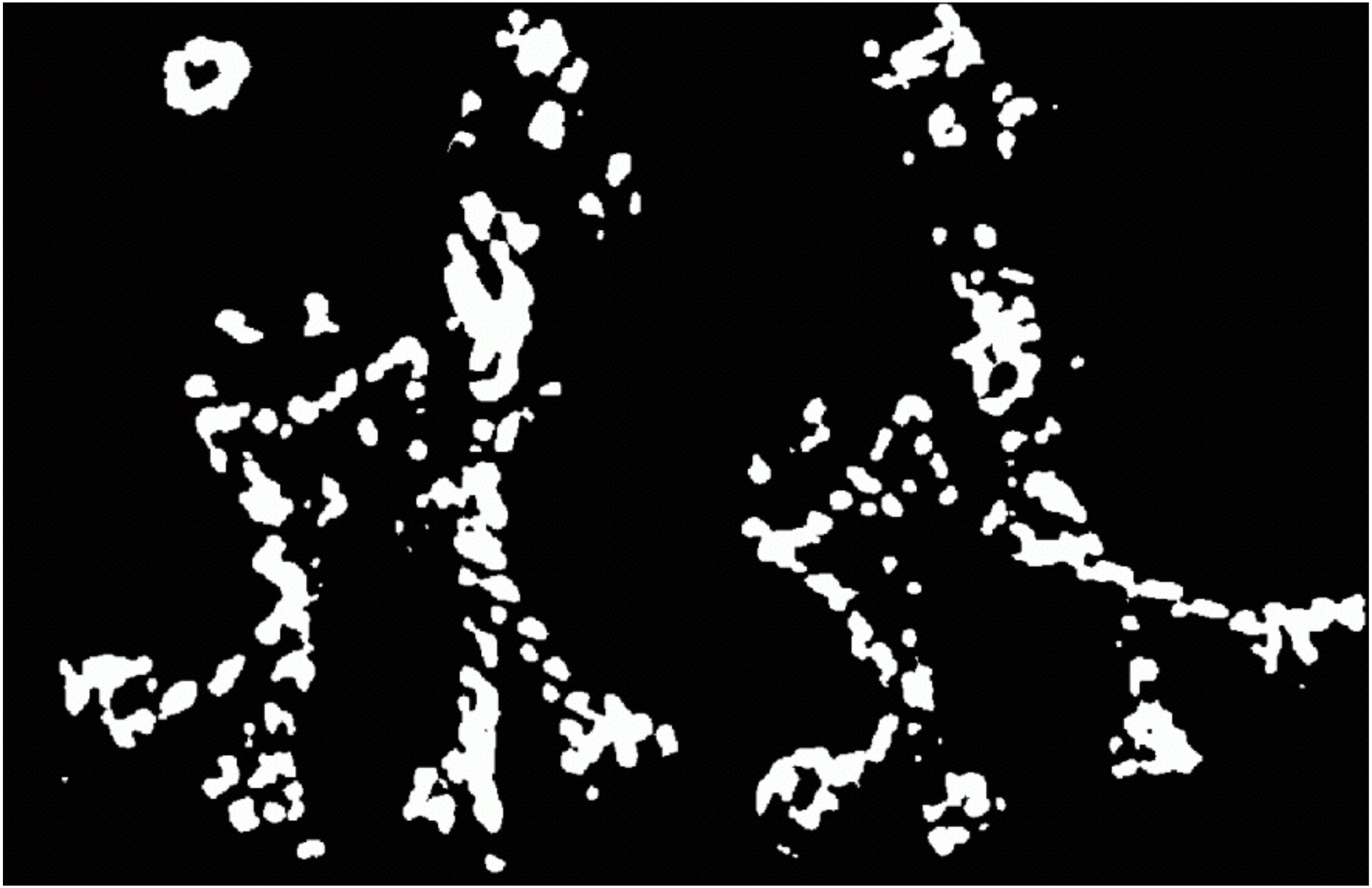
Harris Detector: Workflow

Compute corner response R



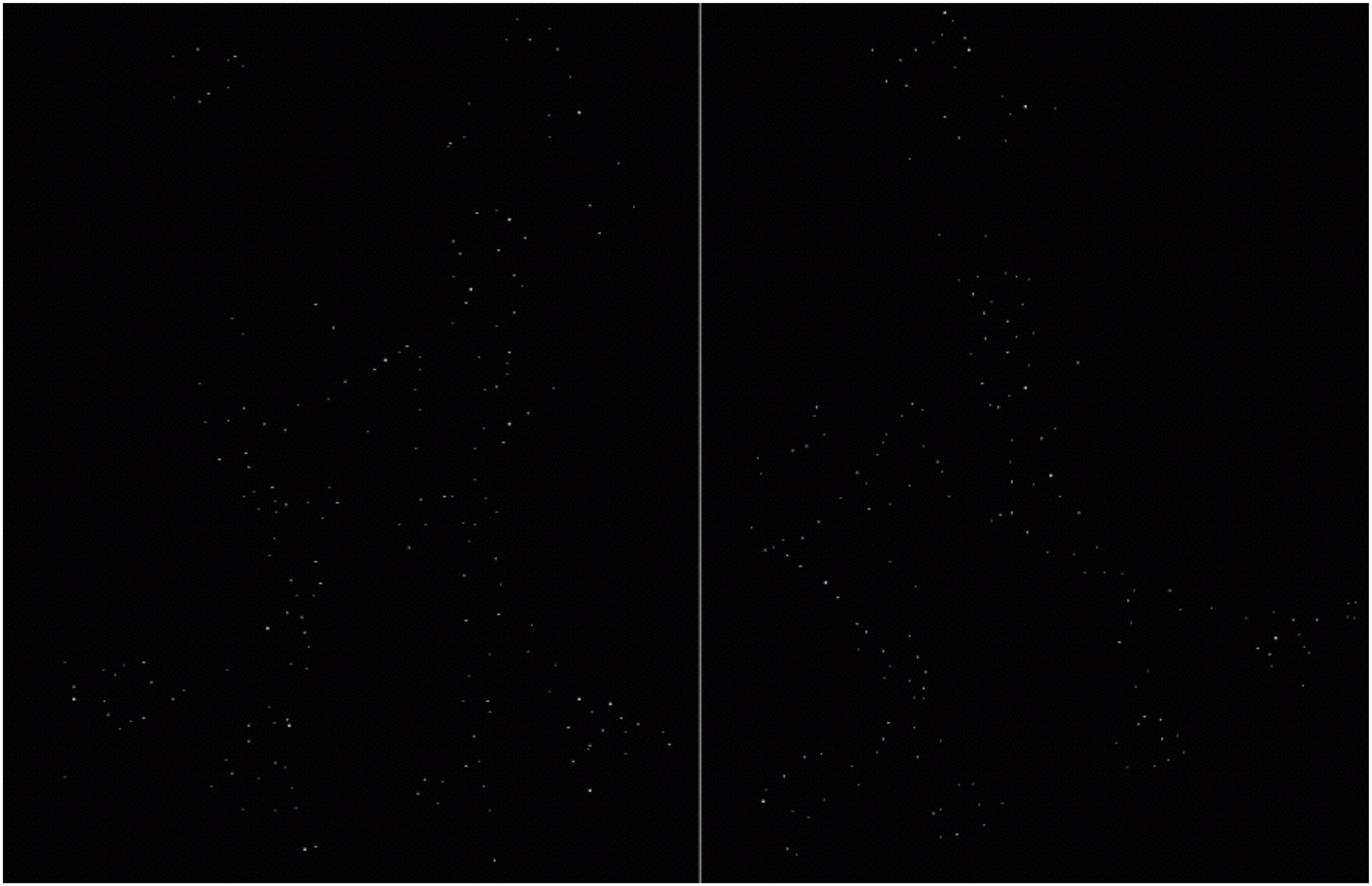
Harris Detector: Workflow

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: Analysis of invariance properties

- **Geometry**
  - rotation
  - scale
- **Photometry**
  - intensity change

Evaluation plots are from this paper

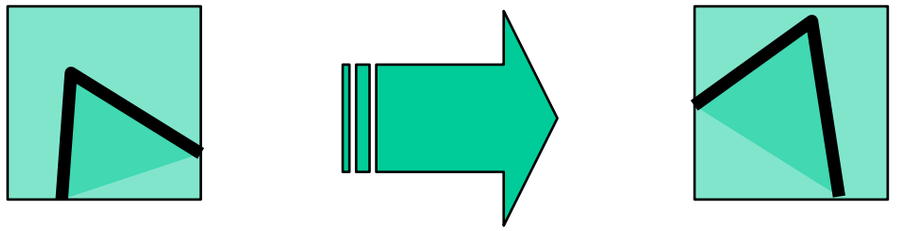
International Journal of Computer Vision 37(2), 151–172, 2000

Evaluation of Interest Point Detectors

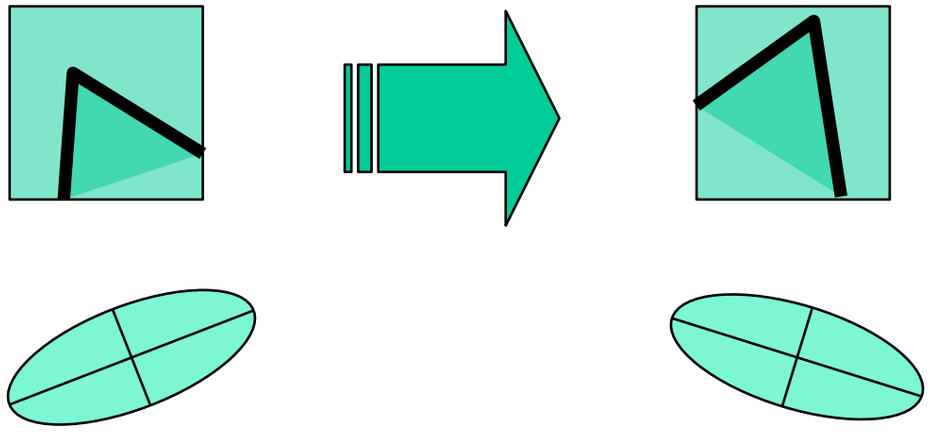
CORDELIA SCHMID, ROGER MOHR AND CHRISTIAN BAUCKHAGE
INRIA Rhône-Alpes, 655 av. de l'Europe, 38330 Montbonnot, France
Cordelia.Schmid@inrialpes.fr

Abstract. Many different low-level feature detectors exist and it is widely agreed that the evaluation of detectors is important. In this paper we introduce two evaluation criteria for interest points: repeatability rate and information content. Repeatability rate evaluates the geometric stability under different transformations. Information content measures the distinctiveness of features. Different interest point detectors are compared using these two criteria. We determine which detector gives the best results and show that it satisfies the criteria well.

Harris Detector: Rotation Invariance?



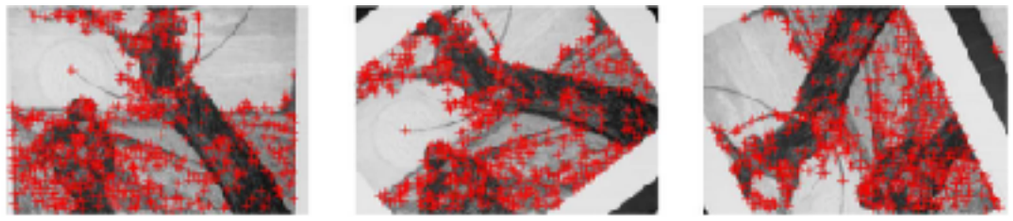
Harris Detector: Rotation Invariance?



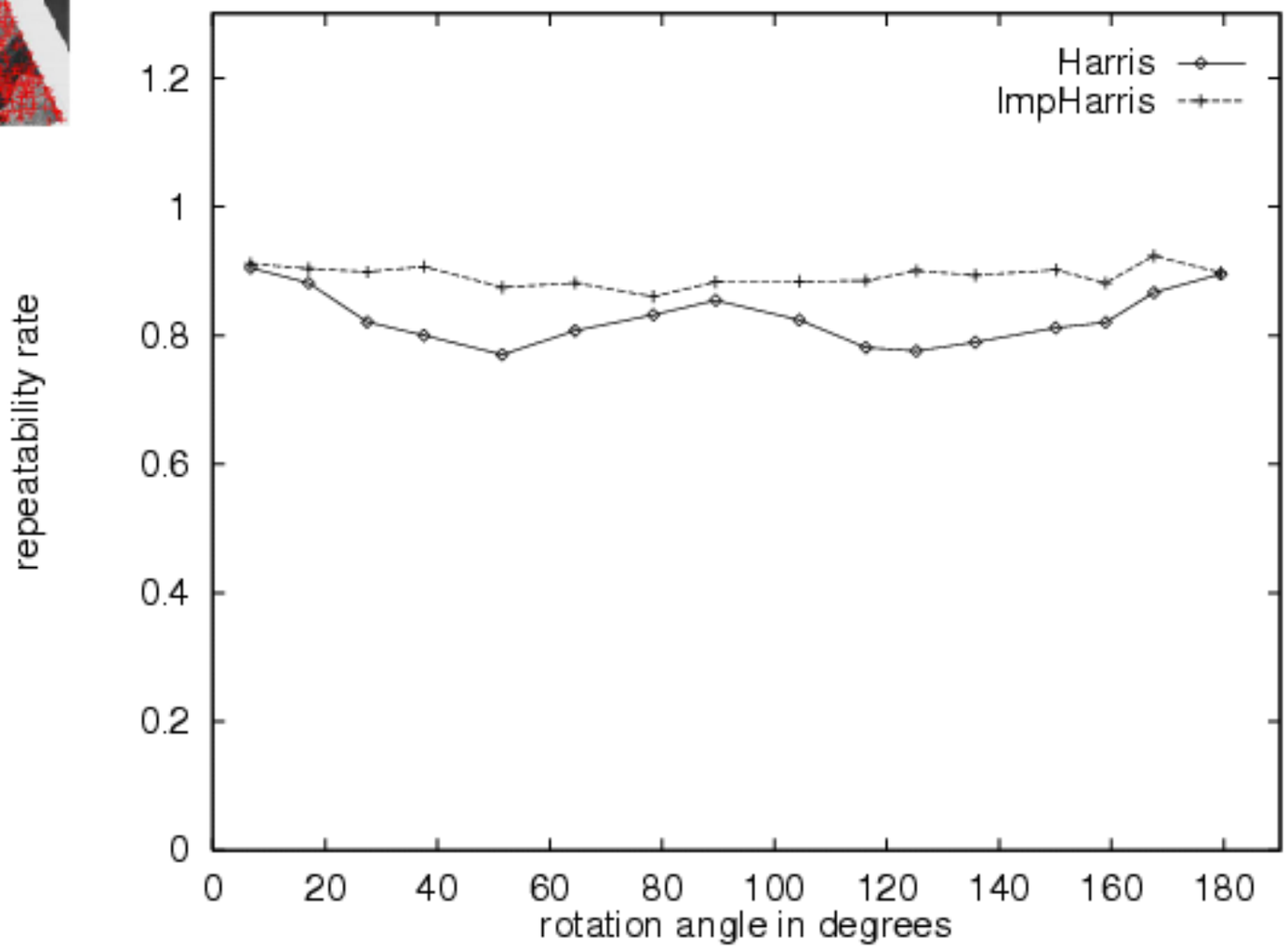
Ellipse rotates but its shape (i.e. eigenvalues) remains the same

Corner response R is invariant to image rotation

Harris Detector: Rotation Invariant Detection



This gives us rotation invariant ***detection***, but we'll need to do more to ensure a rotation invariant ***descriptor***...



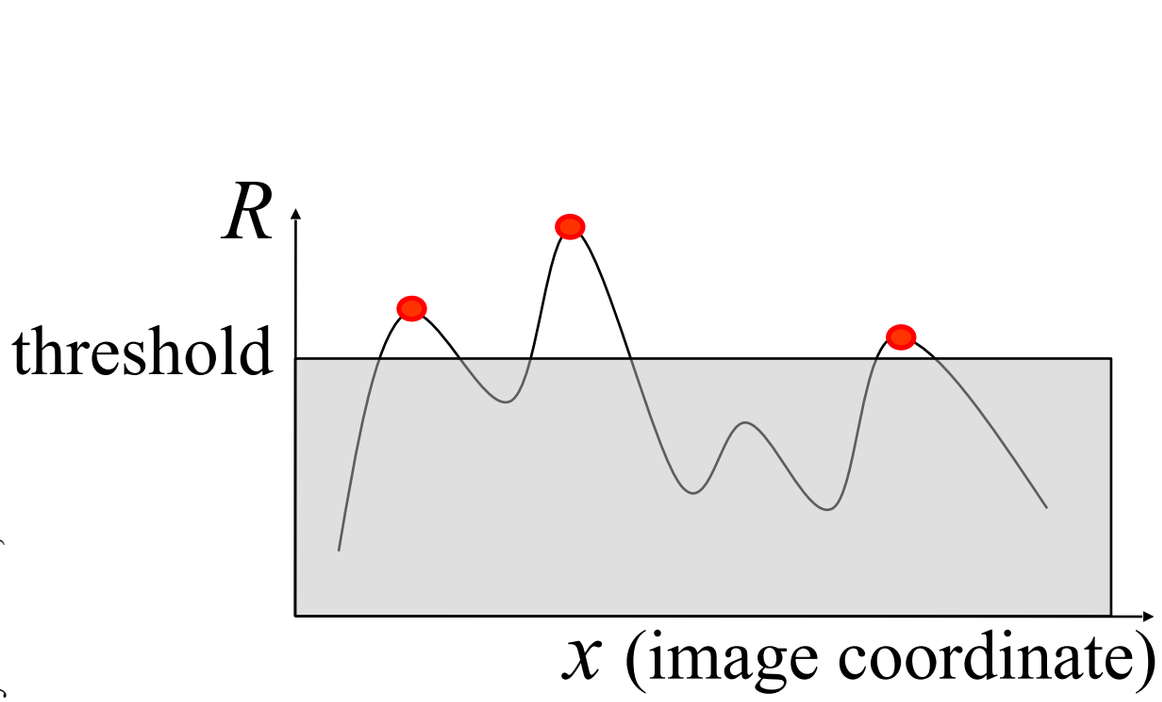
ImpHarris: derivatives are computed more precisely by replacing the [-2 -1 0 1 2] mask with derivatives of a Gaussian (sigma = 1).

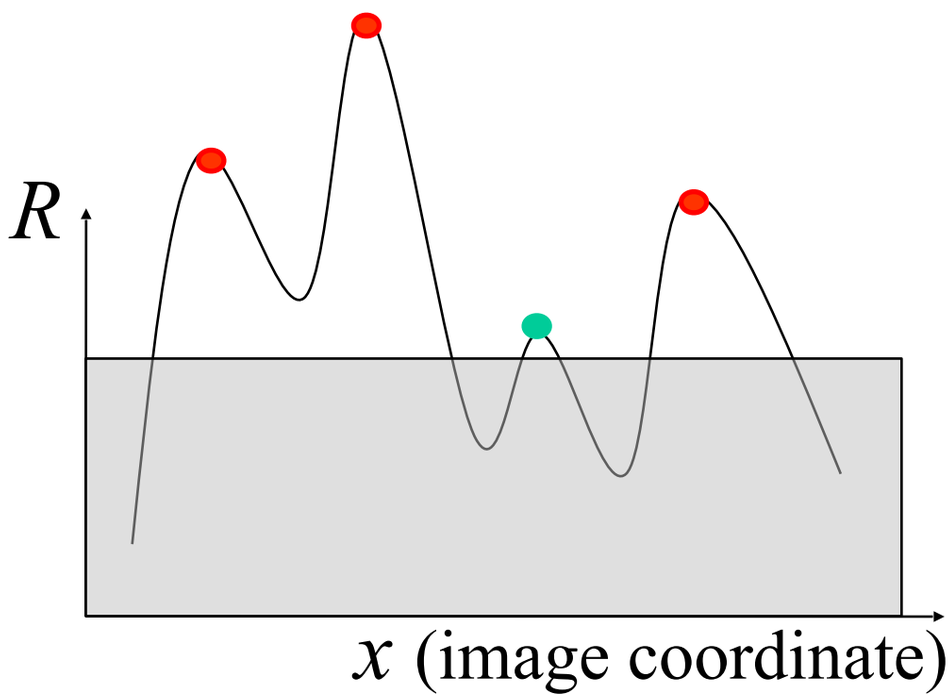
Harris Detector: Invariance to intensity change?

- Partial invariance to additive and multiplicative intensity changes

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

  ✓ Intensity scaling: $I \rightarrow a\,I$ fine, *except for the threshold that's used to specify when R is large enough.*





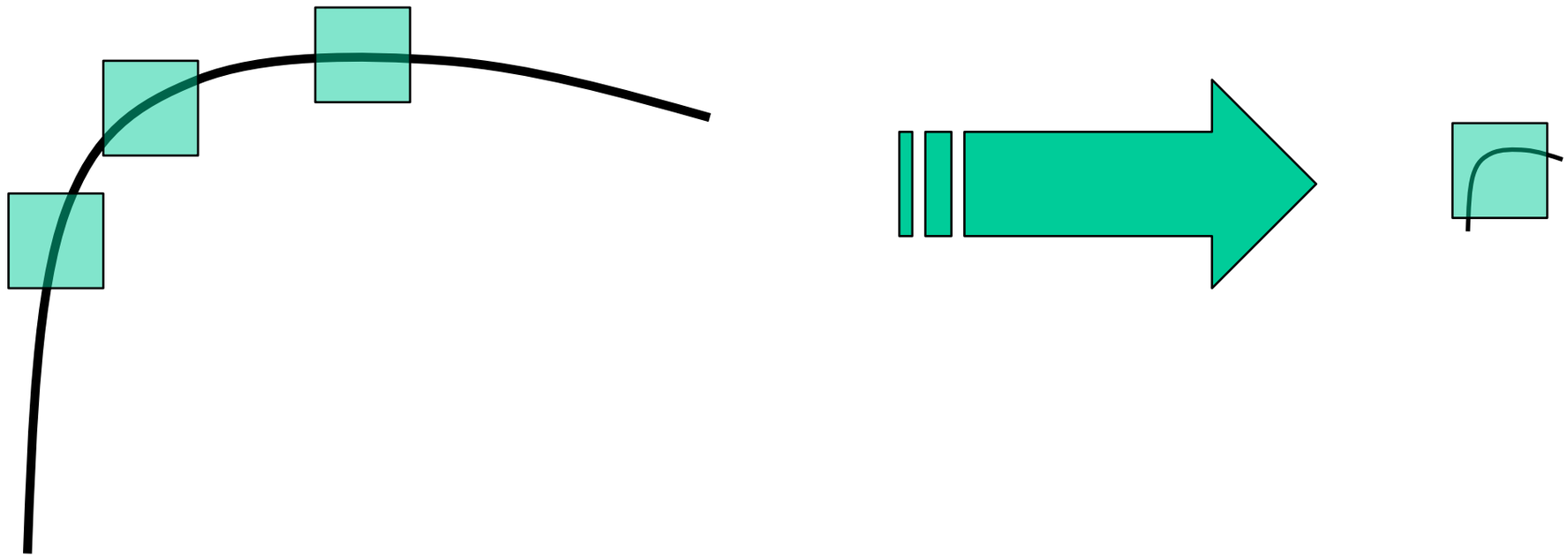
Harris Detector: Invariance to image scale?

image

zoomed image

Harris Detector: Invariance to image scale?

- Not invariant to *image scale*!



All points will be classified as edges

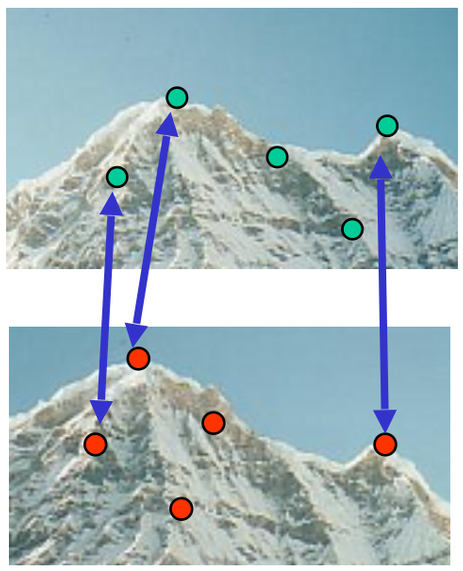
Corner !

Harris Detector: Invariance to image scale?

- Quality of Harris detector for different scale changes

Repeatability rate:

$$\frac{\#\text{ correspondences}}{\#\text{ detected points (mean in two images)}}$$

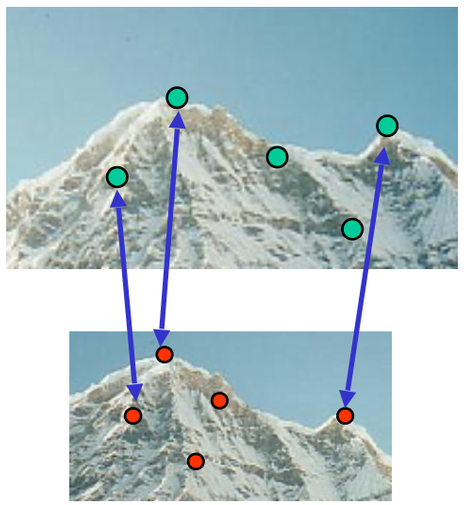


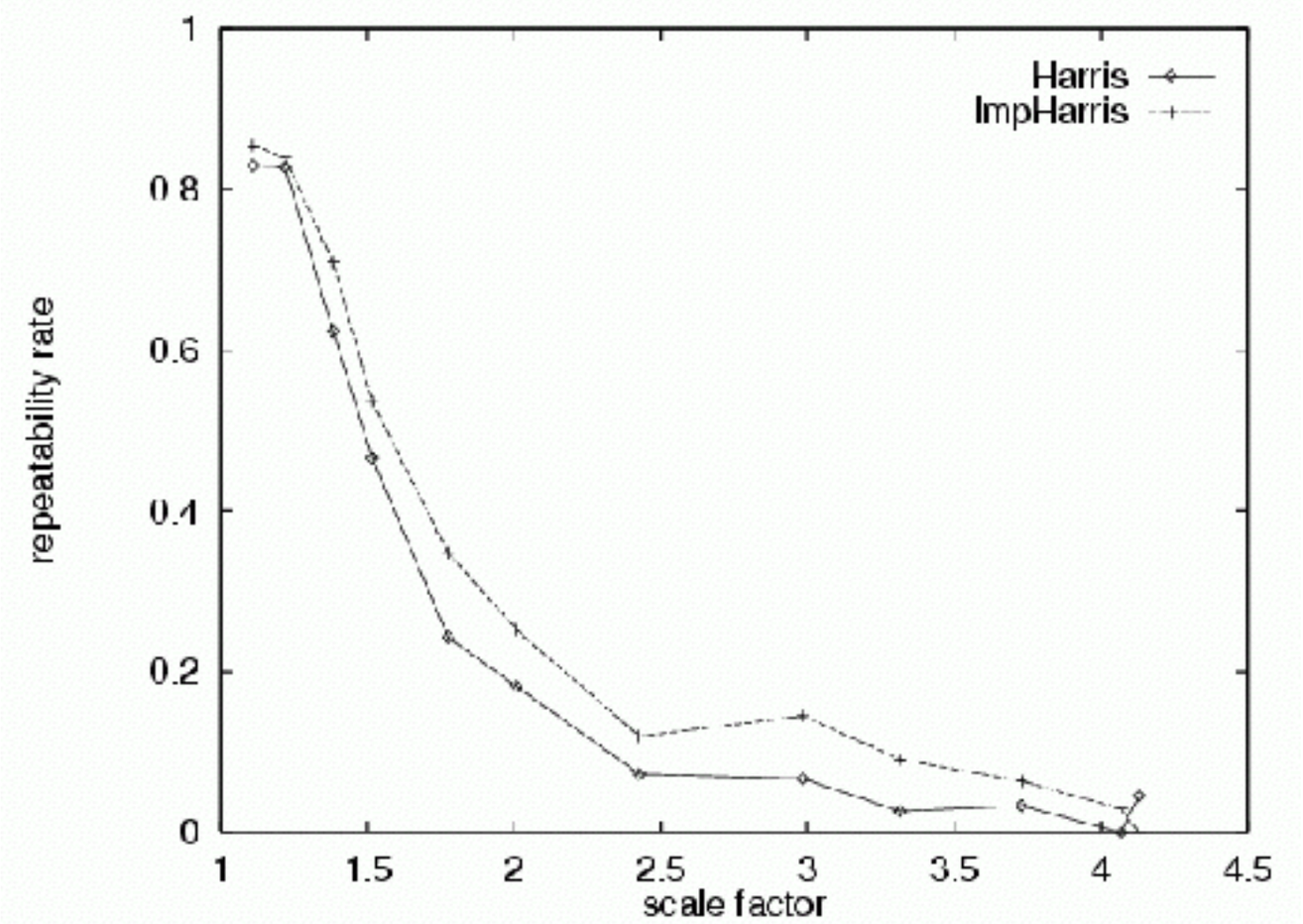
Harris Detector: Invariance to image scale?

- Quality of Harris detector for different scale changes

Repeatability rate:

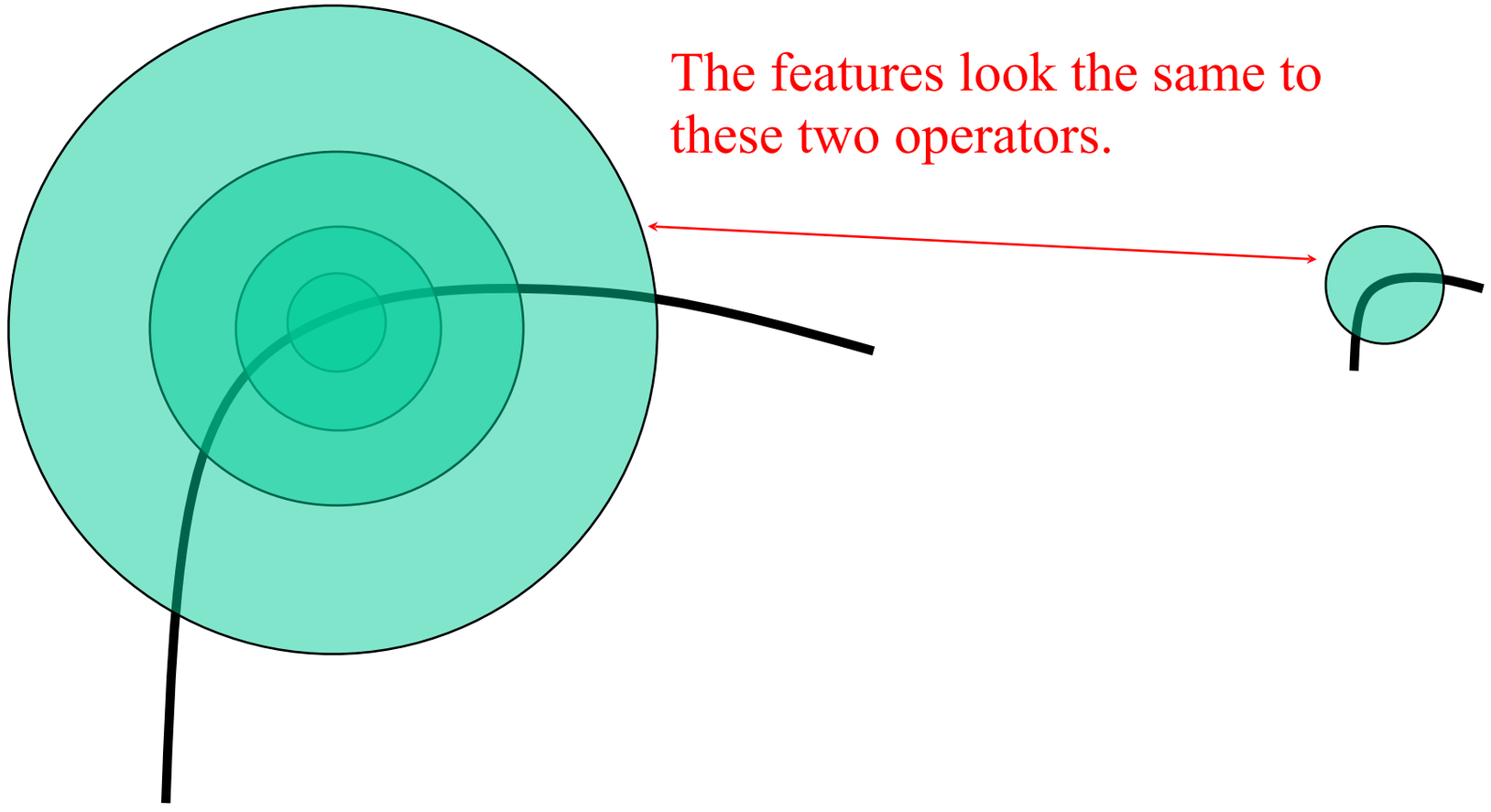
$$\frac{\#\text{ correspondences}}{\#\text{ detected points (mean in two images)}}$$





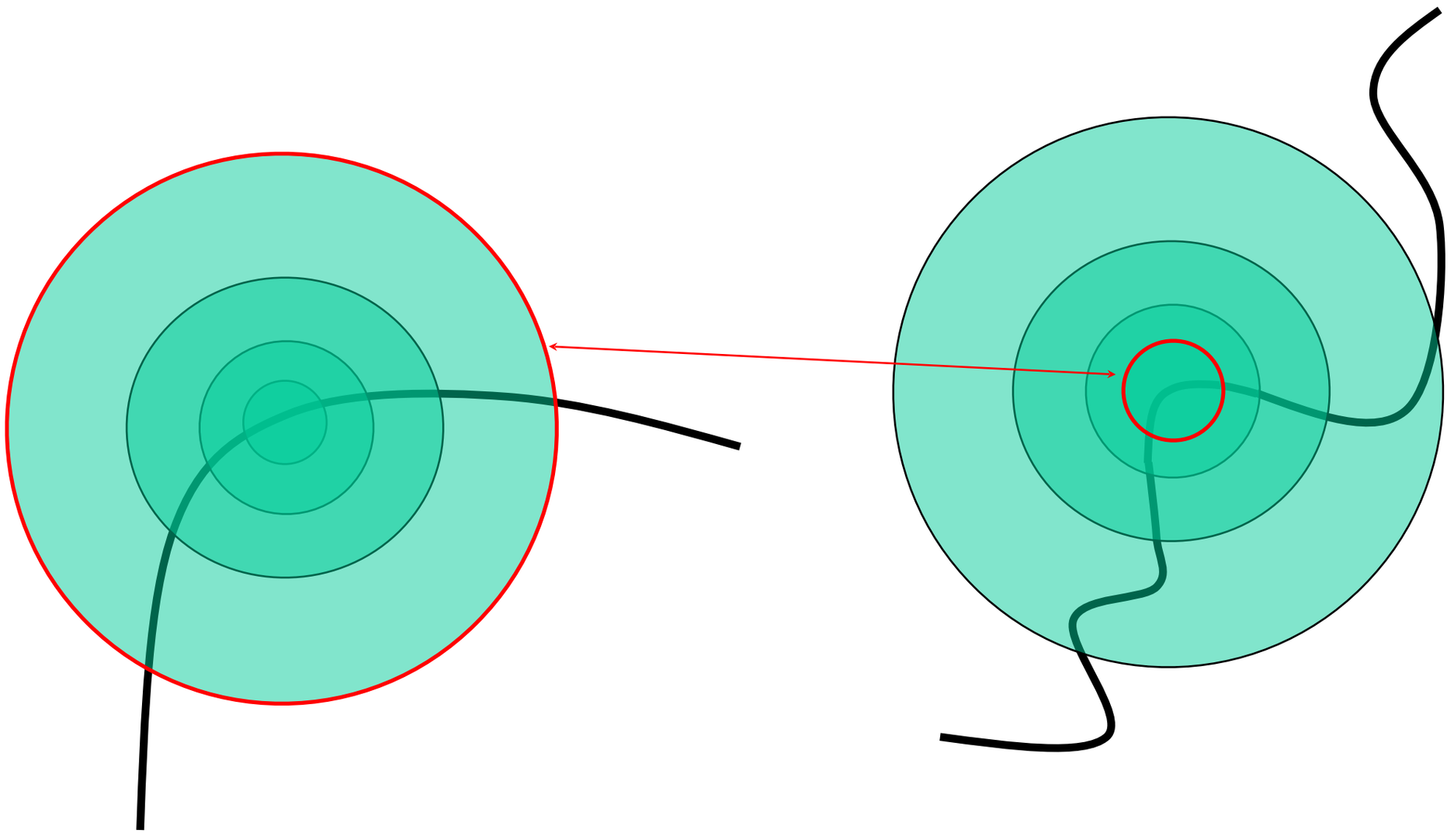
Scale Invariant Detection

- 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?
- Do objects in the image have a characteristic scale that we can identify?

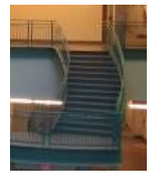


Scale Invariant Detection

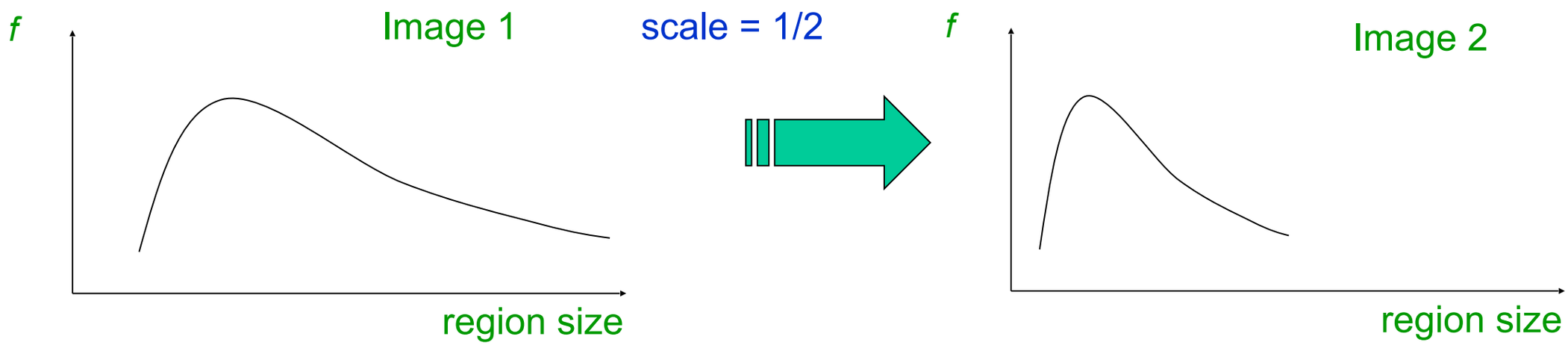
Solution:

- Design a function on the region (circle), which is “scale invariant” (the same response 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.

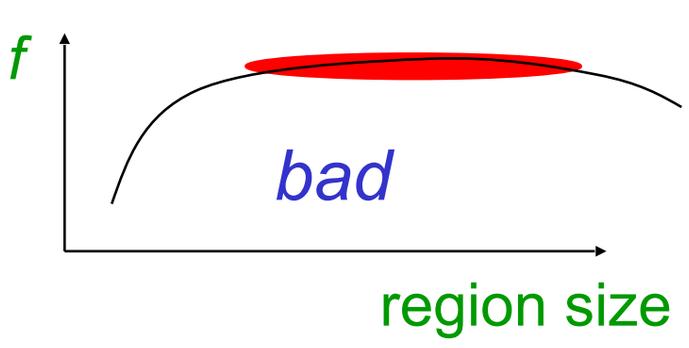


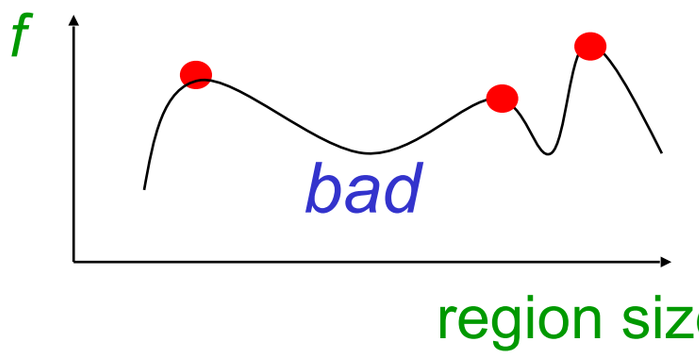
- For a point in one image, we can consider it as a function of region size (circle radius)

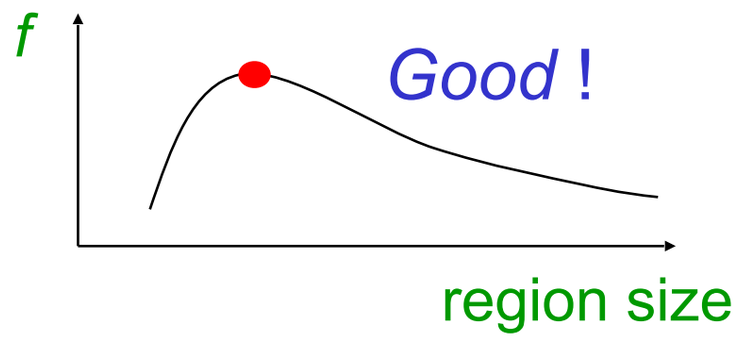


Scale Invariant Detection

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

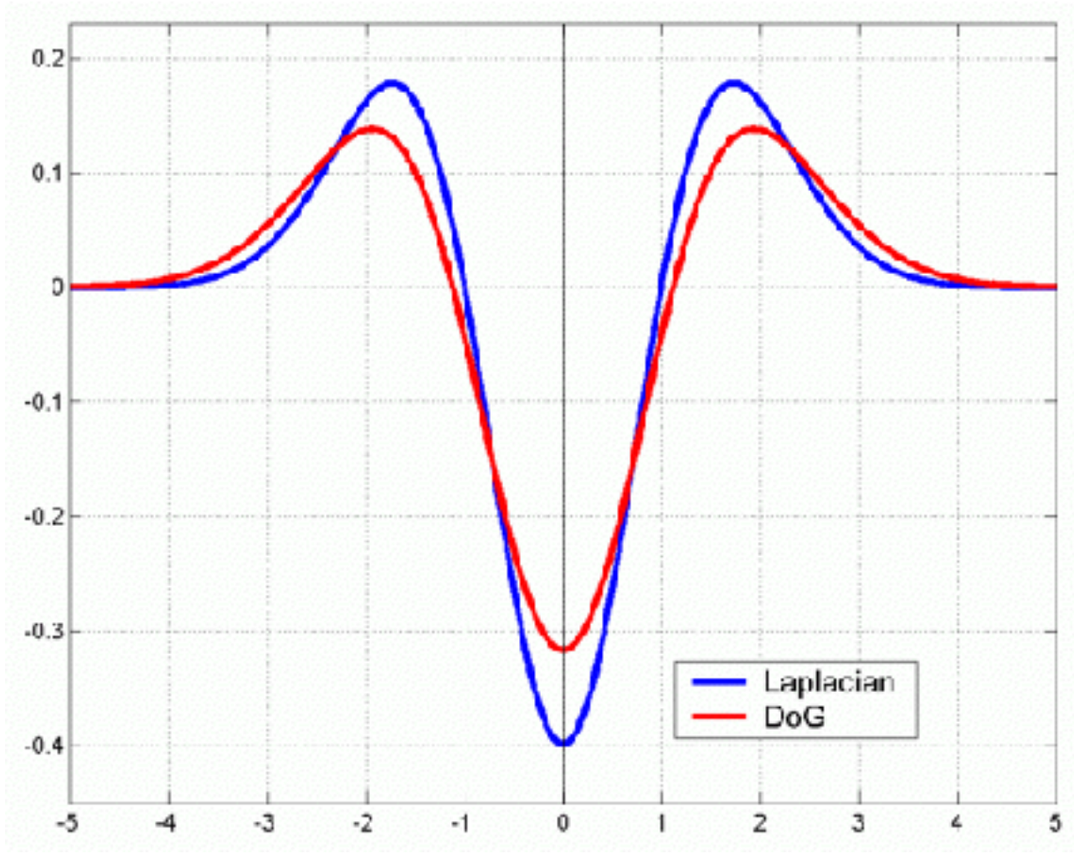
- Functions for determining scale

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

Kernels:

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

(Laplacian: 2nd derivative of Gaussian)

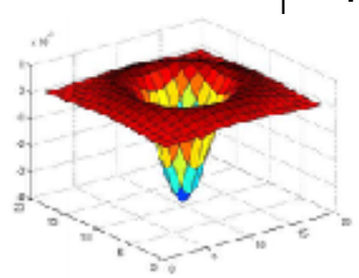


$$DoG = G(x, y, k\sigma) - G(x, y, \sigma)$$

(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*

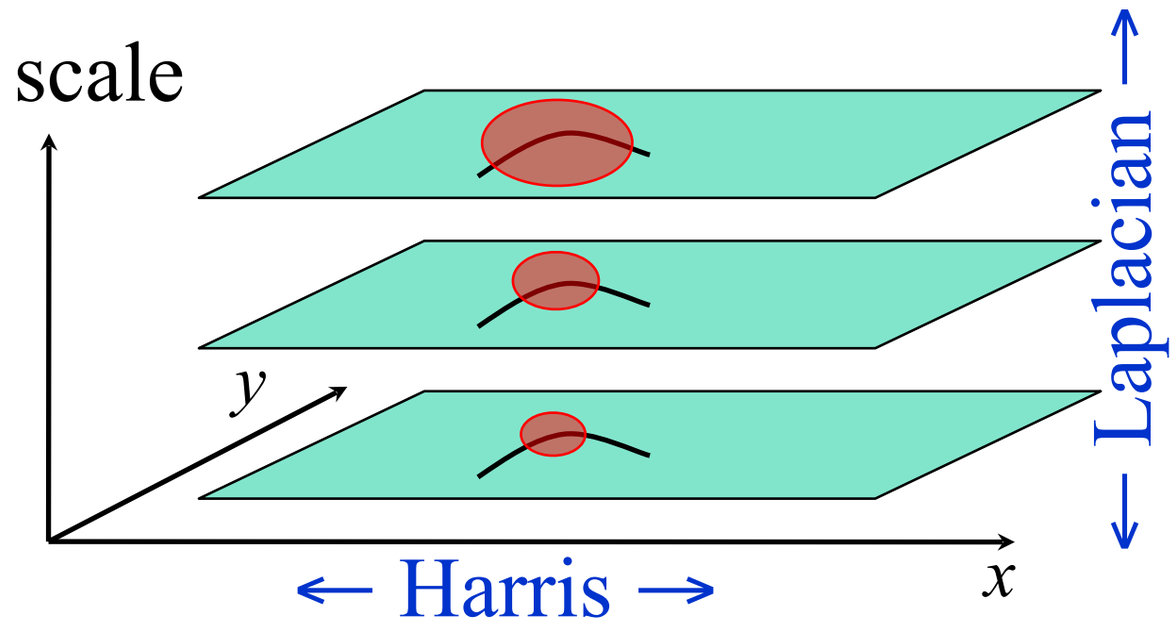
Scale Invariant Detectors

- Harris-Laplacian[1]

  Find local maximum of:

  – Harris corner detector in space (image coordinates)

  – Laplacian in scale



[1] K.Mikolajczyk, C.Schmid. “Indexing Based on Scale Invariant Interest Points”. ICCV 2001

Laplacian of Gaussian for selection of characteristic scale

$$|\mathrm{LoG}(\mathbf{x}, \sigma_n)| = \sigma_n^2 |L_{xx}(\mathbf{x}, \sigma_n) + L_{yy}(\mathbf{x}, \sigma_n)|$$

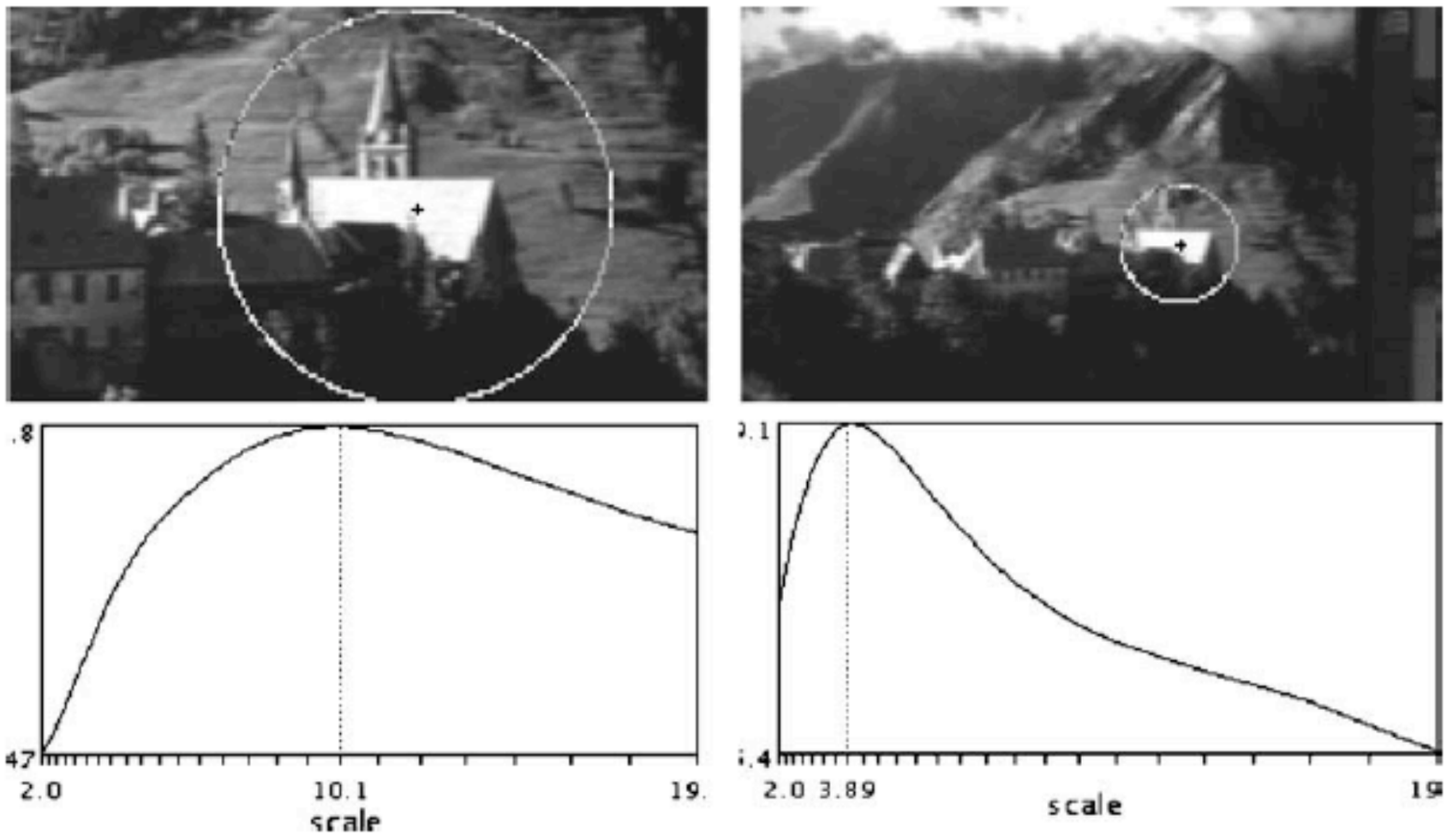


Figure 1. Example of characteristic scales. The top row shows two images taken with different focal lengths. The bottom row shows the response $F_{\mathrm{norm}}(\mathbf{x}, \sigma_n)$ over scales where F_{norm} is the normalized LoG (cf. Eq. (3)). The characteristic scales are 10.1 and 3.89 for the left and right image, respectively. The ratio of scales corresponds to the scale factor (2.5) between the two images. The radius of displayed regions in the top row is equal to 3 times the characteristic scale.

http://www.robots.ox.ac.uk/~vgg/research/affine/det_eval_files/mikolajczyk_ijcv2004.pdf

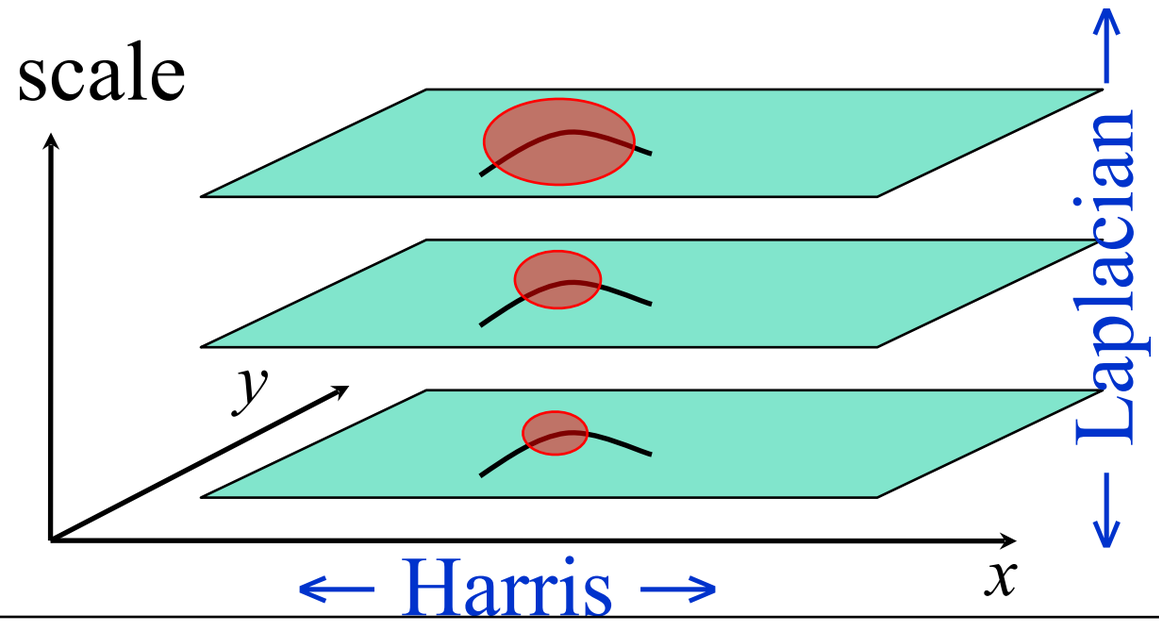
Scale Invariant Detectors

- **Harris-Laplacian**[1]

  Find local maximum of:

  – Harris corner detector in space (image coordinates)

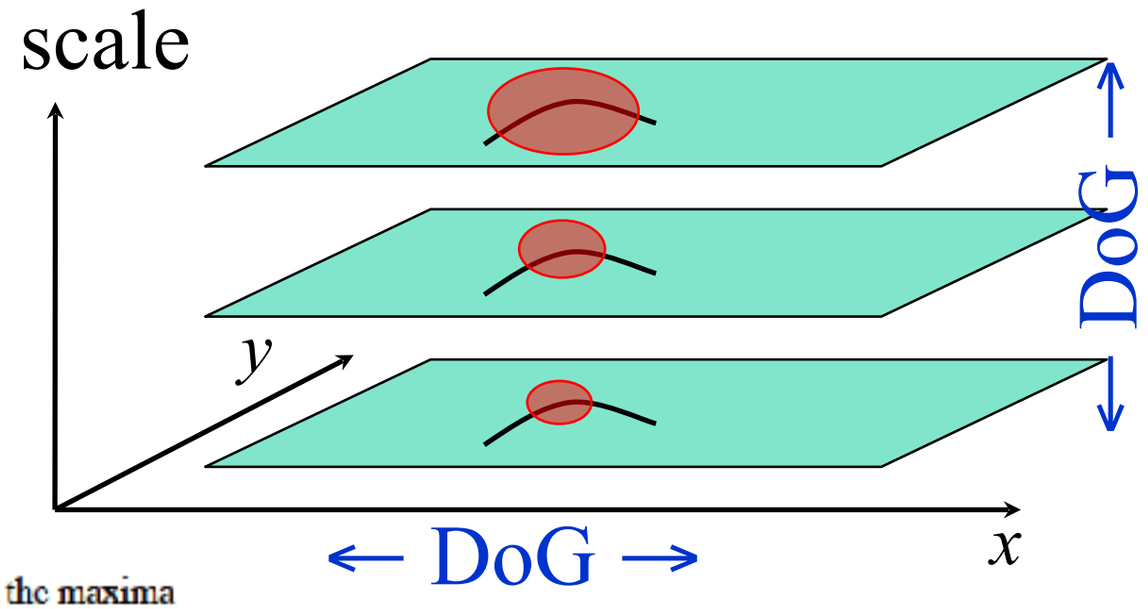
  – Laplacian in scale



- **SIFT (Lowe)**[2]

  Find local maximum (minimum) of:

  – Difference of Gaussians in space and scale



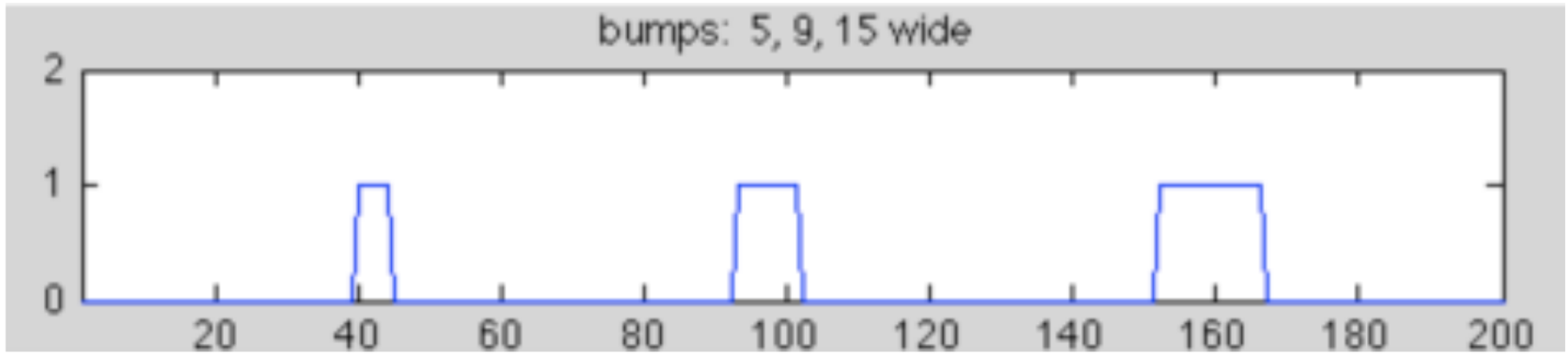
In detailed experimental comparisons, Mikolajczyk (2002) found that the maxima and minima of $\sigma^2\nabla^2 G$ produce the most stable image features compared to a range of other possible image functions, such as the gradient, Hessian, or Harris corner function.

[1] *K.Mikolajczyk, C.Schmid. "Indexing Based on Scale Invariant Interest Points". ICCV 2001*

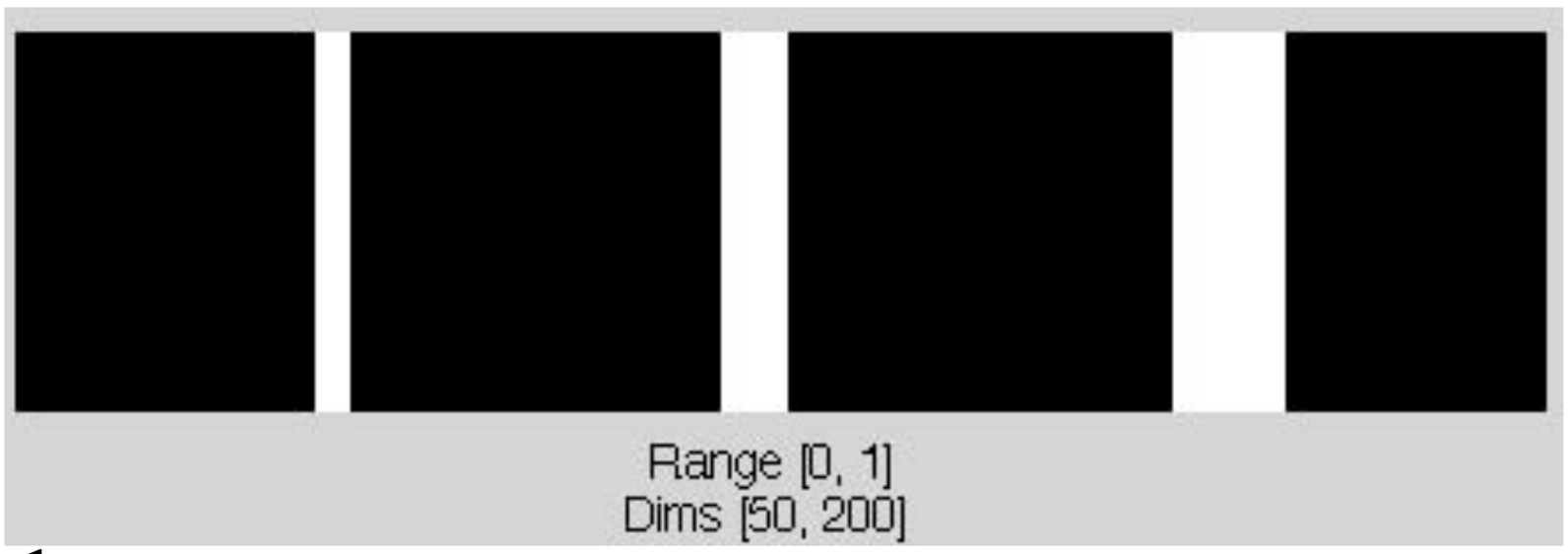
[2] *D.Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". Accepted to IJCV 2004*

Scale-space example: 3 bumps of different widths

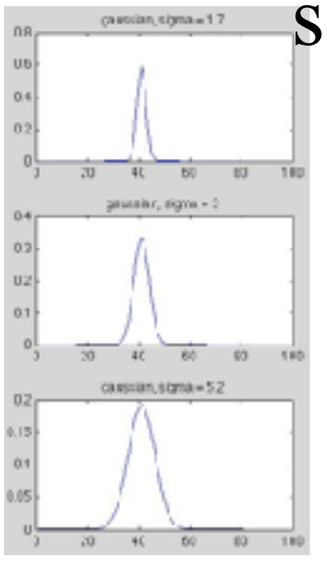
1-D bumps



Display as an image

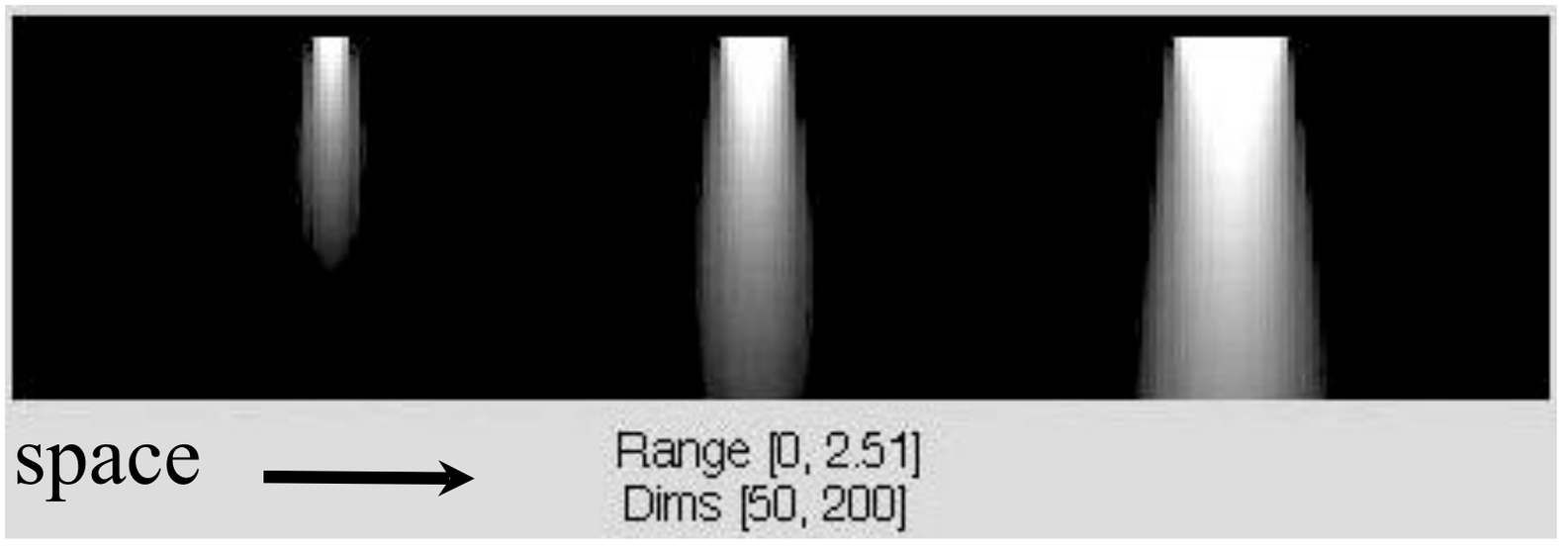


blur with Gaussians of increasing width

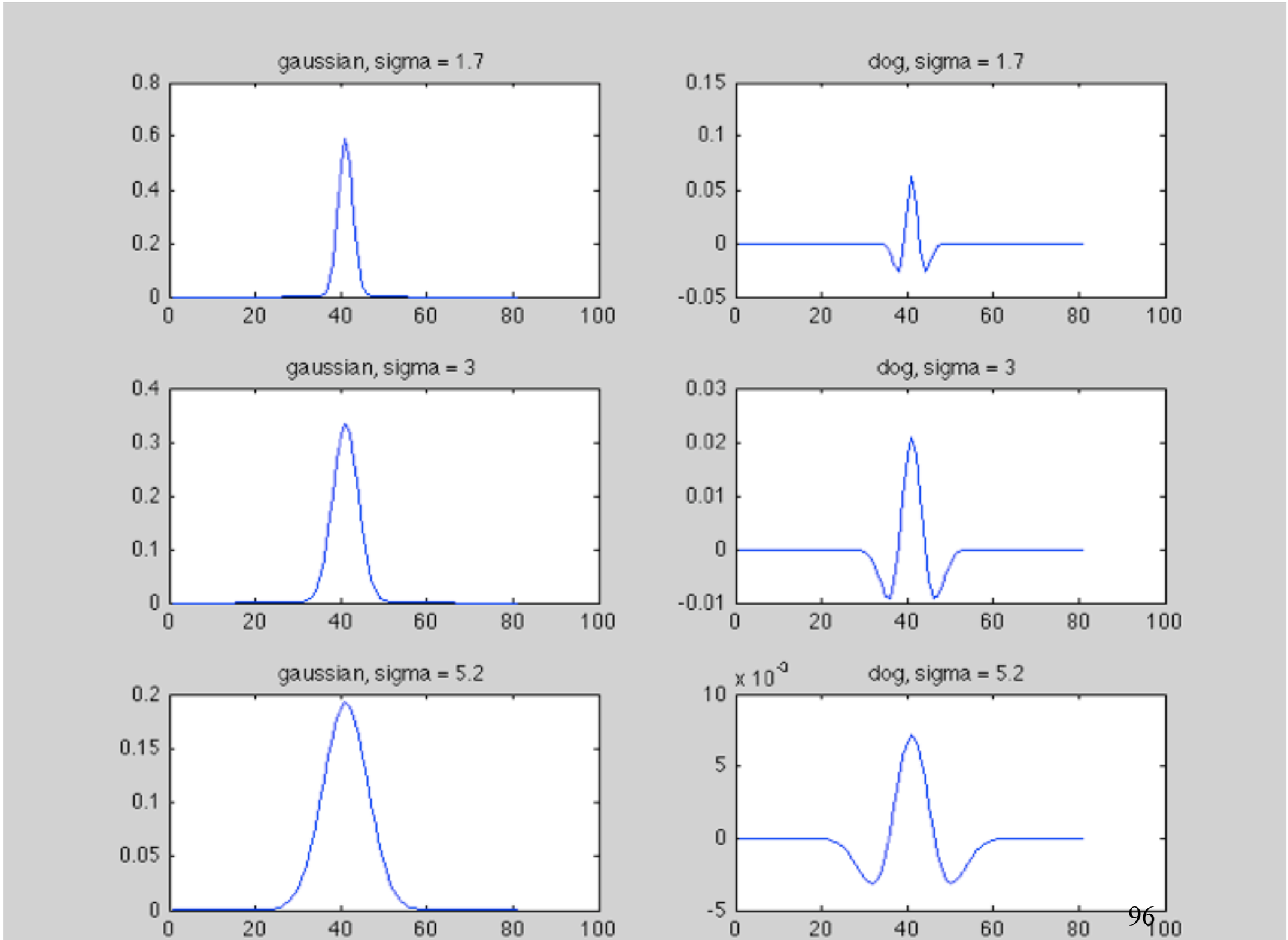


scale

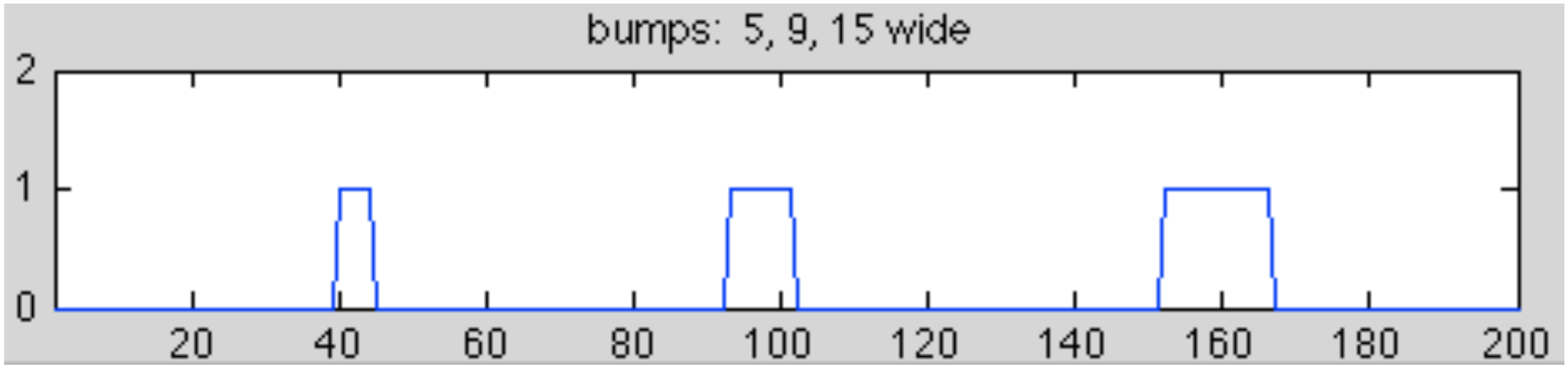
space

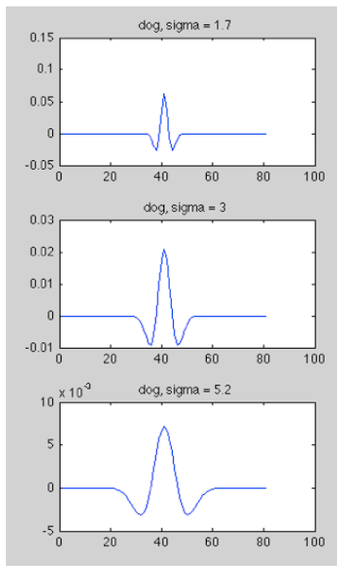


Gaussian and difference-of-Gaussian filters

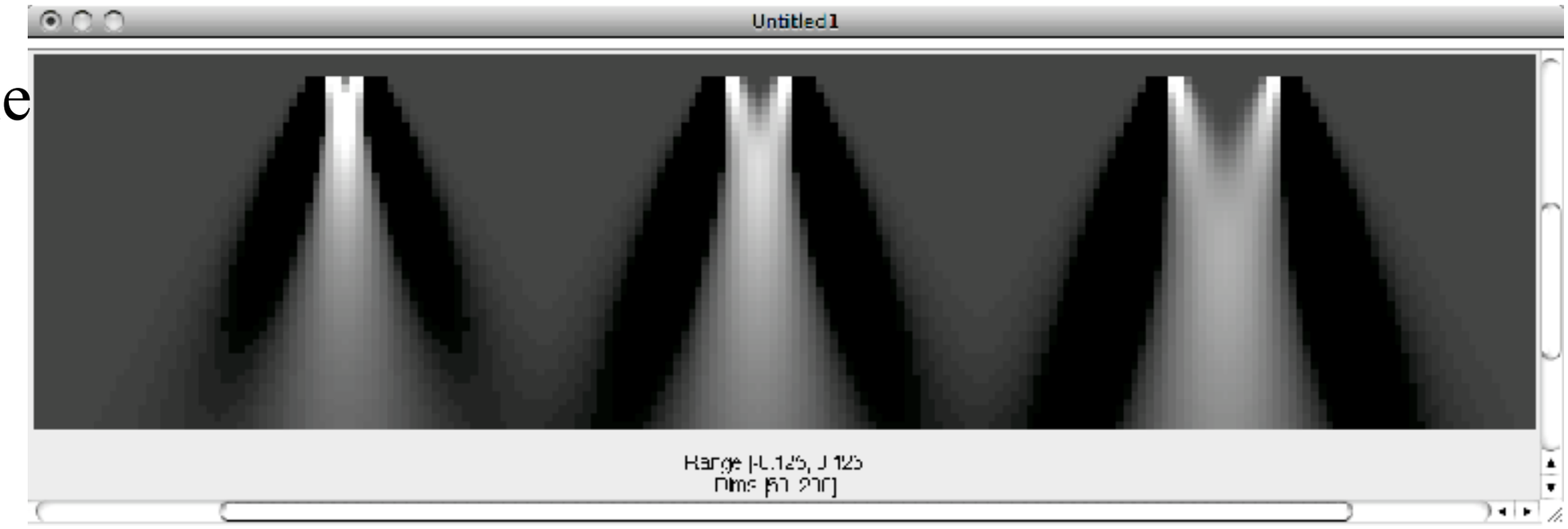


The bumps, filtered by difference-of-Gaussian filters



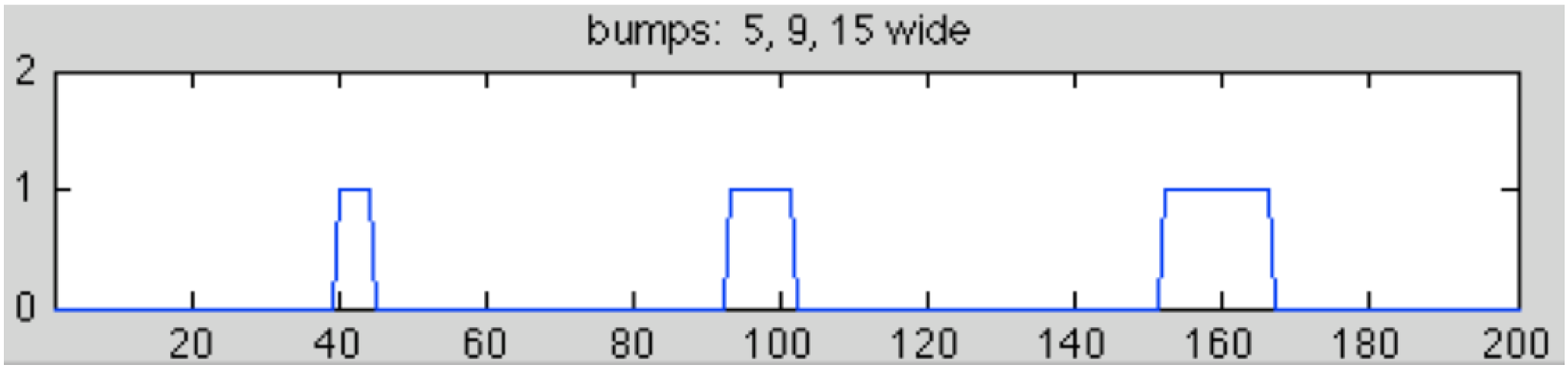


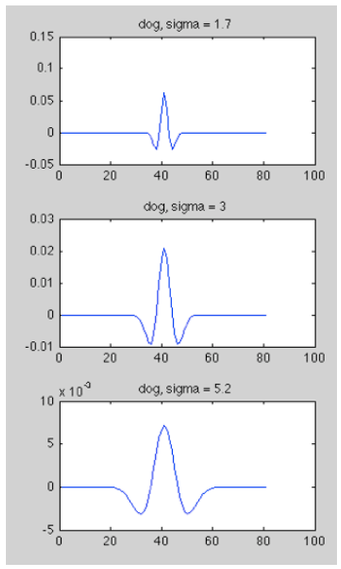
scale

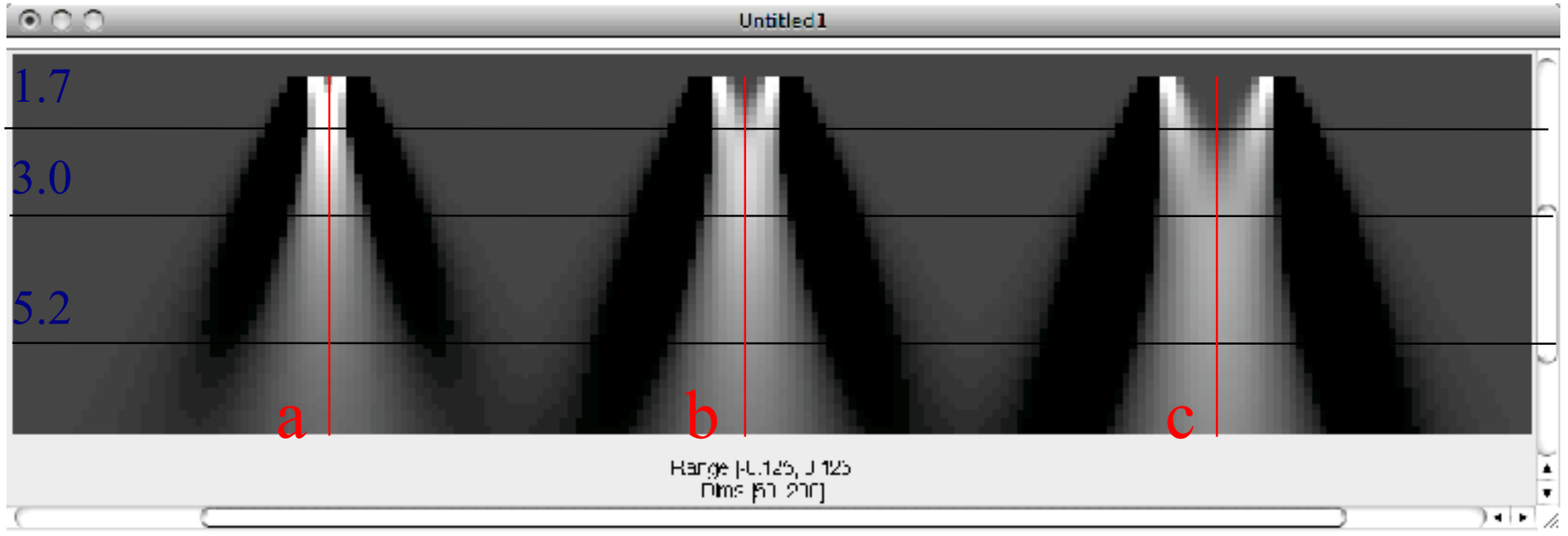


space

The bumps, filtered by difference-of-Gaussian filters

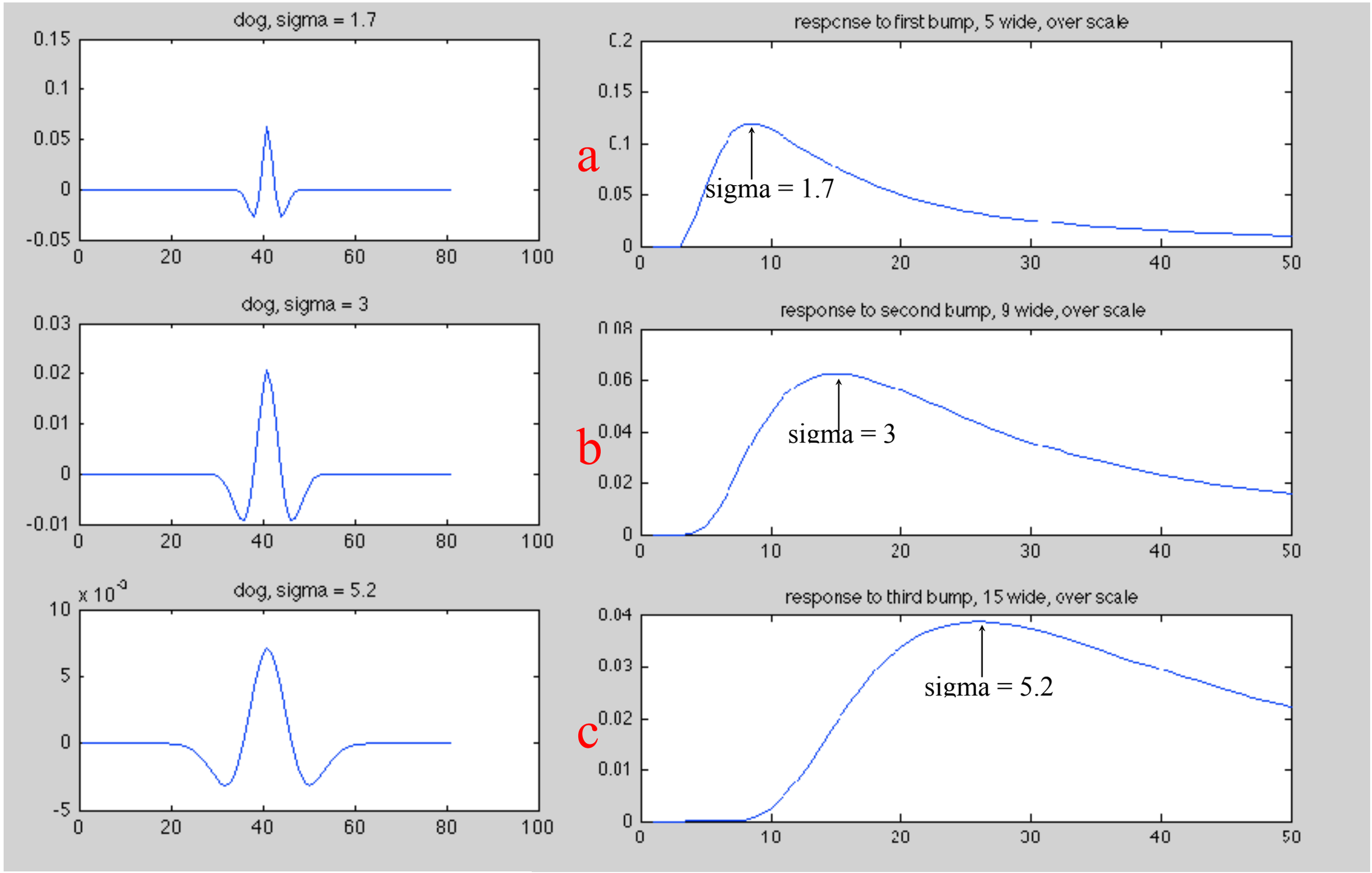




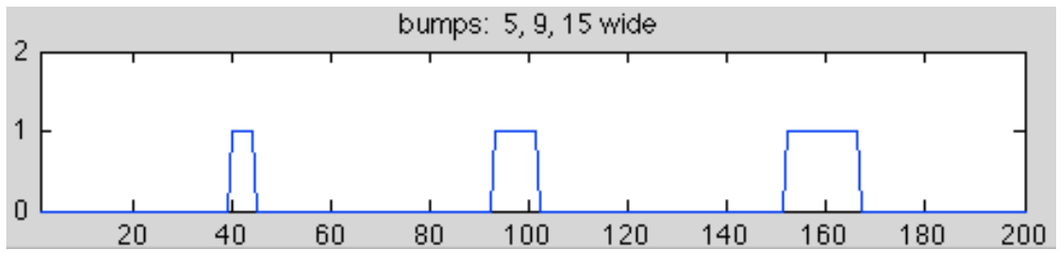


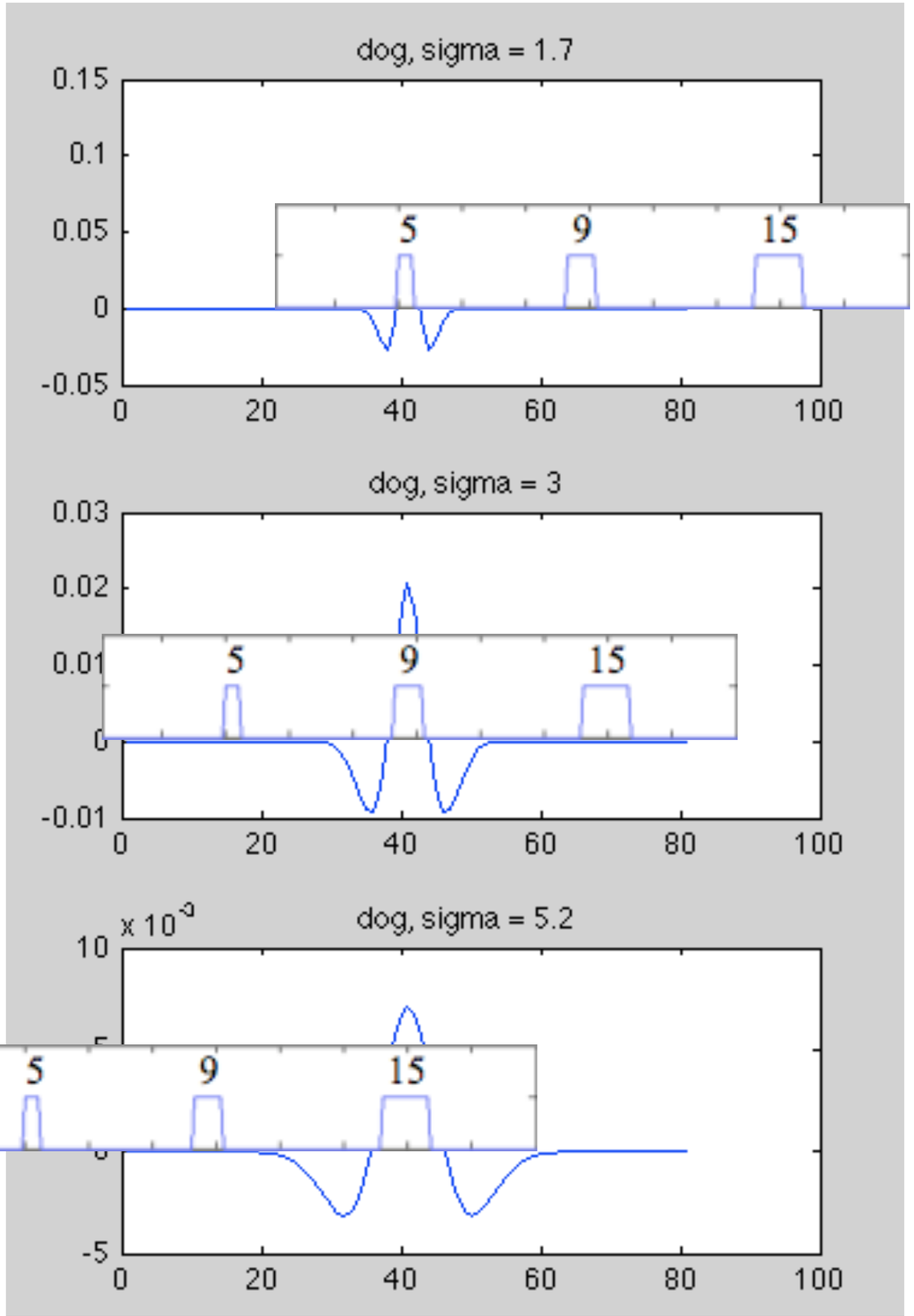
cross-sections along red lines plotted next slide

Scales of peak responses are proportional to bump width (the characteristic scale of each bump):
[1.7, 3, 5.2] ./ [5, 9, 15] = 0.3400 0.3333 0.3467



Diff of Gauss filter giving peak response





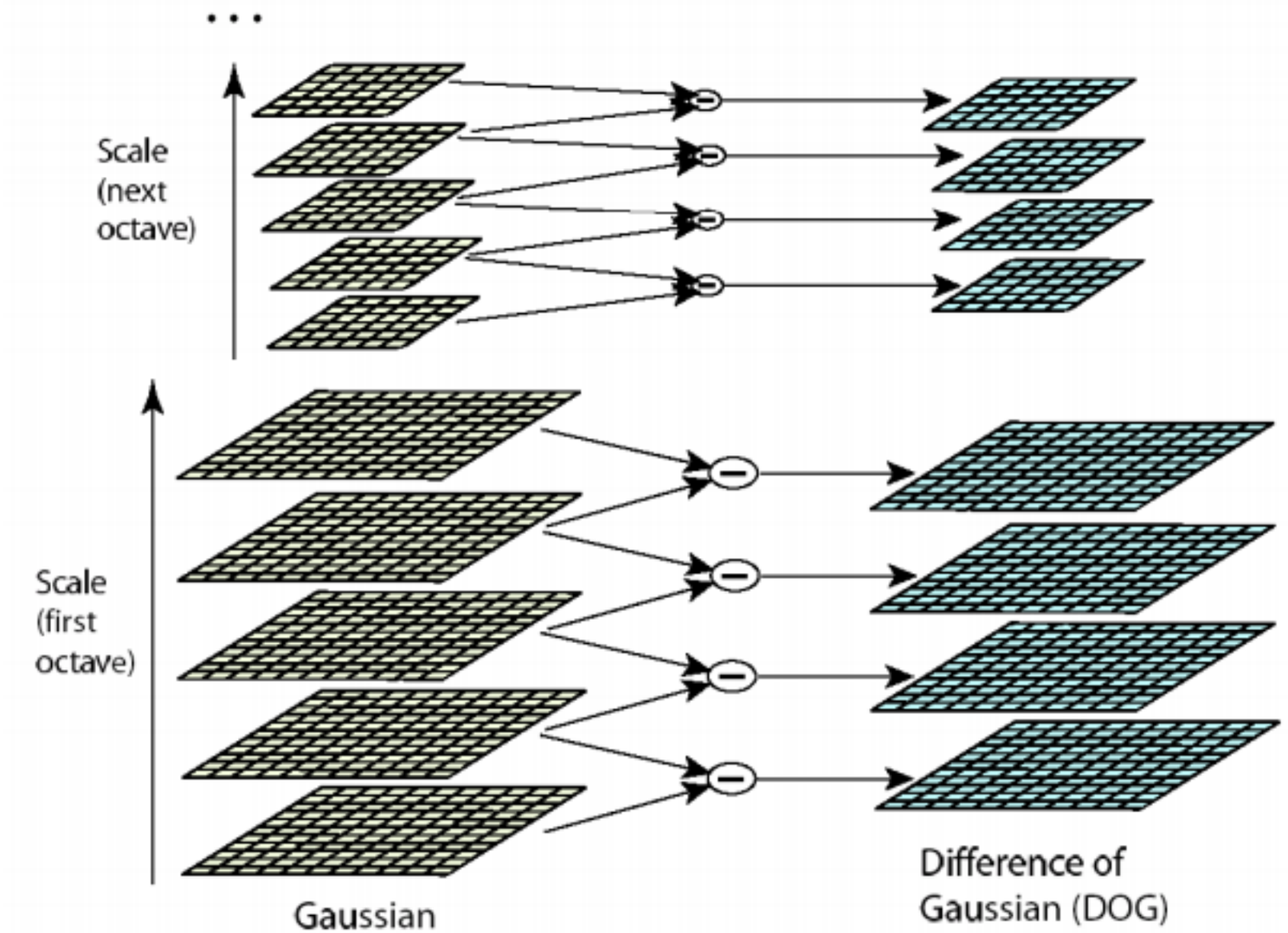
Scales of peak responses are proportional to bump width (the characteristic scale of each bump):

[1.7, 3, 5.2] ./ [5, 9, 15] = 0.3400 0.3333 0.3467

Note that the max response filters each has the same relationship to the bump that it favors (the zero crossings of the filter are about at the bump edges). So the scale space analysis correctly picks out the "characteristic scale" for each of the bumps.

More generally, this happens for the features of the images we analyze.

DOG Scale Space (Lowe 2004)



Each octave is doubling of scale – Halve image dimensions

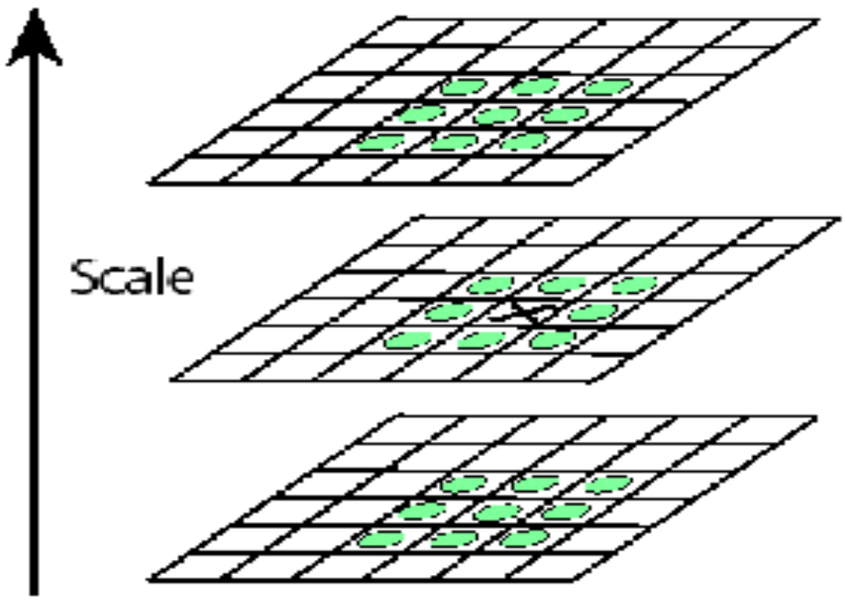
Within octave several scales – Same dimension for all images

Figure 1: For each octave of scale space, the initial image is repeatedly convolved with Gaussians to produce the set of scale space images shown on the left. Adjacent Gaussian images are subtracted to produce the difference-of-Gaussian images on the right. After each octave, the Gaussian image is down-sampled by a factor of 2, and the process repeated.

Key point localization over scale and space

- Detect extremum (maxima and minima) of difference-of-Gaussian in scale space.

  (non-maximum suppression both spatially and in scale)



- Fit a quadratic to surrounding values for sub-pixel and sub-scale interpolation (Brown & Lowe, 2002)
- Taylor expansion around point:

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

- Offset of extremum (use finite differences for derivatives):

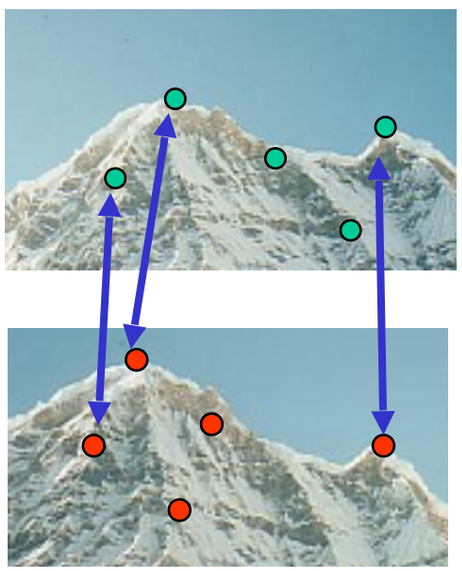
$$\hat{\mathbf{x}} = -\frac{\partial^2 D}{\partial \mathbf{x}^2}^{-1}\frac{\partial D}{\partial \mathbf{x}}$$

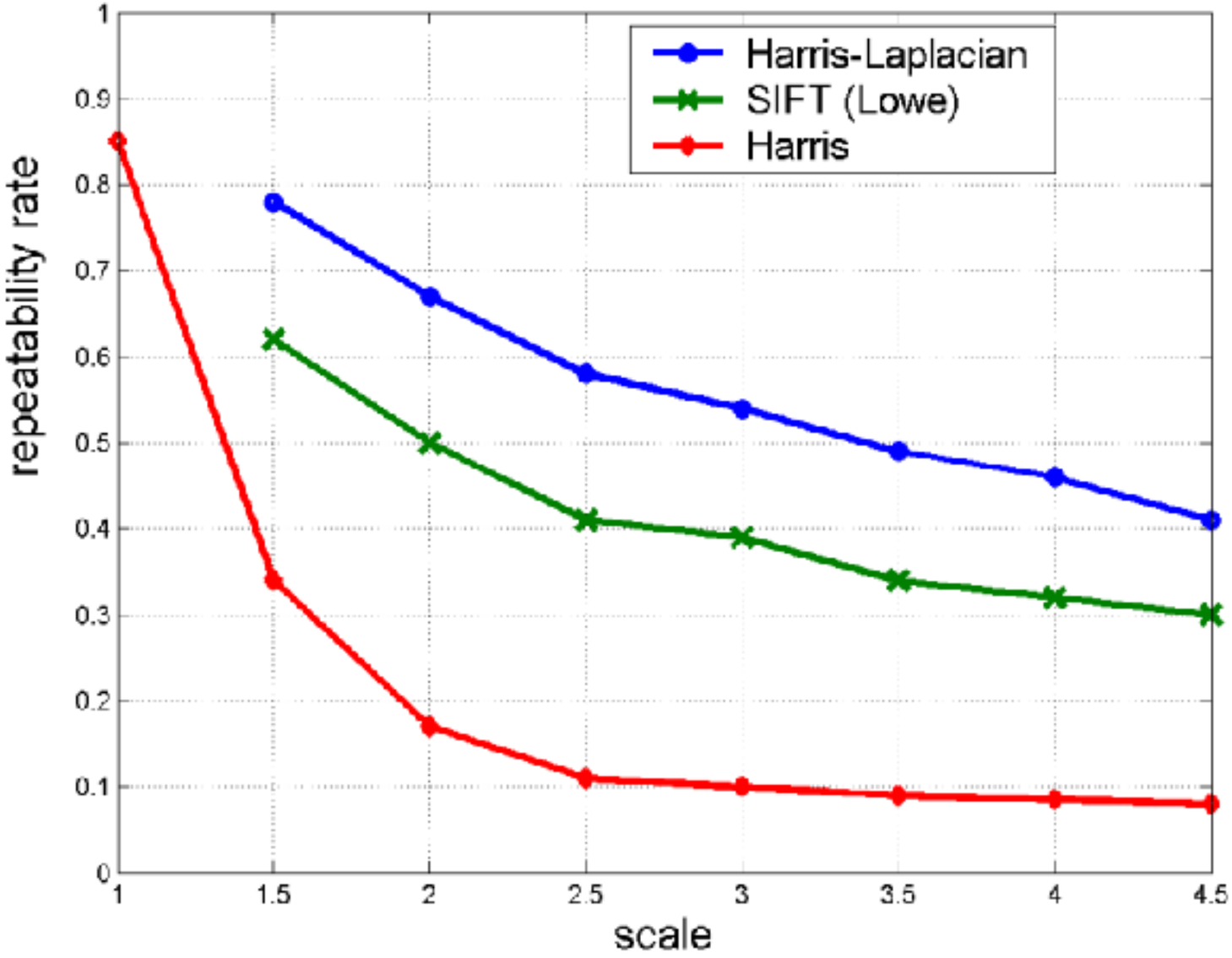
Scale Invariant Detectors

- Experimental evaluation of detectors w.r.t. scale change

Repeatability rate:

correspondences
possible correspondences





K.Mikolajczyk, C.Schmid. “Indexing Based on Scale Invariant Interest Points”. ICCV 2001

Scale and Rotation Invariant Detection: Recap

- **Given:** two images of the same scene with a large **scale difference and/or rotation** between them
- **Goal:** find **the same** interest points **independently** in each image
- **Solution:** search for **maxima** of suitable functions in **scale** and in **space** (over the image). Also, find characteristic **orientation**.
    - » finding a characteristic scale

Methods:

1. **Harris-Laplacian** [Mikolajczyk, Schmid]: maximize Laplacian over scale, Harris' measure of corner response over the image.
2. **SIFT** [Lowe]: maximize Difference of Gaussians over scale and space

Example of Keypoints Detection

(c)

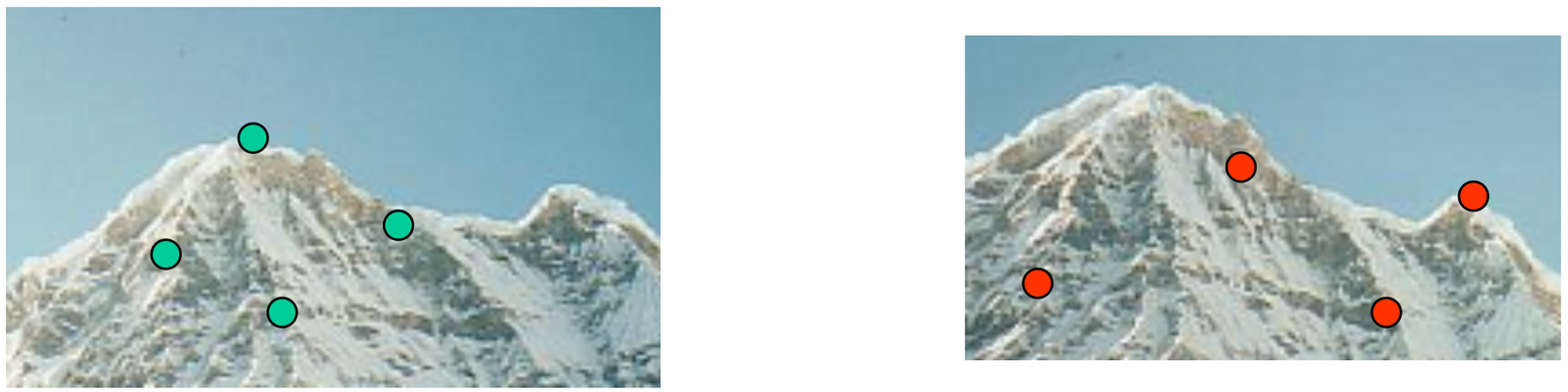
Figure 12. Robust matching: Harris-Laplace detects 190 and 213 points in the left and right images, respectively (a). 58 points are initially matched (b). There are 32 inliers to the estimated homography (c), all of which are correct. The estimated scale factor is 4.9 and the estimated rotation angle is 19 degrees.

http://www.robots.ox.ac.uk/~vgg/research/affine/det_eval_files/mikolajczyk_ijcv2004.pdf

Recall: Matching with Features

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

counter-example:

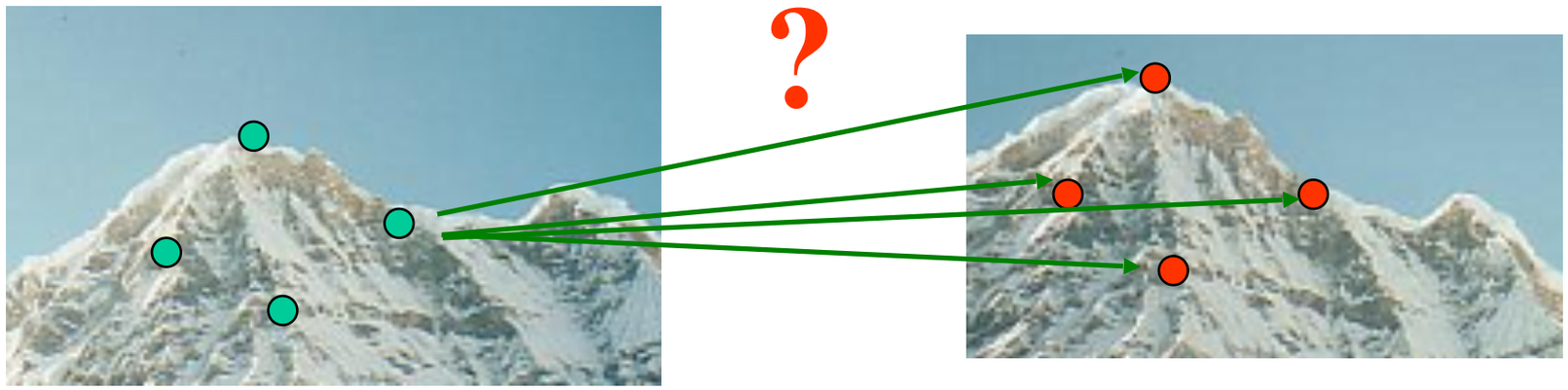


no chance to match!

We need a repeatable **detector**

Recall: Matching with Features

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



We need a reliable and distinctive **descriptor**

CVPR 2003 Tutorial

Recognition and Matching Based on Local Invariant Features

David Lowe
Computer Science Department
University of British Columbia

http://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf

Back to SIFT — Vector Formation

- Computed on rotated and scaled version of window according to computed orientation & scale
  - resample the window
- Based on gradients weighted by a Gaussian of variance half the window (for smooth falloff)

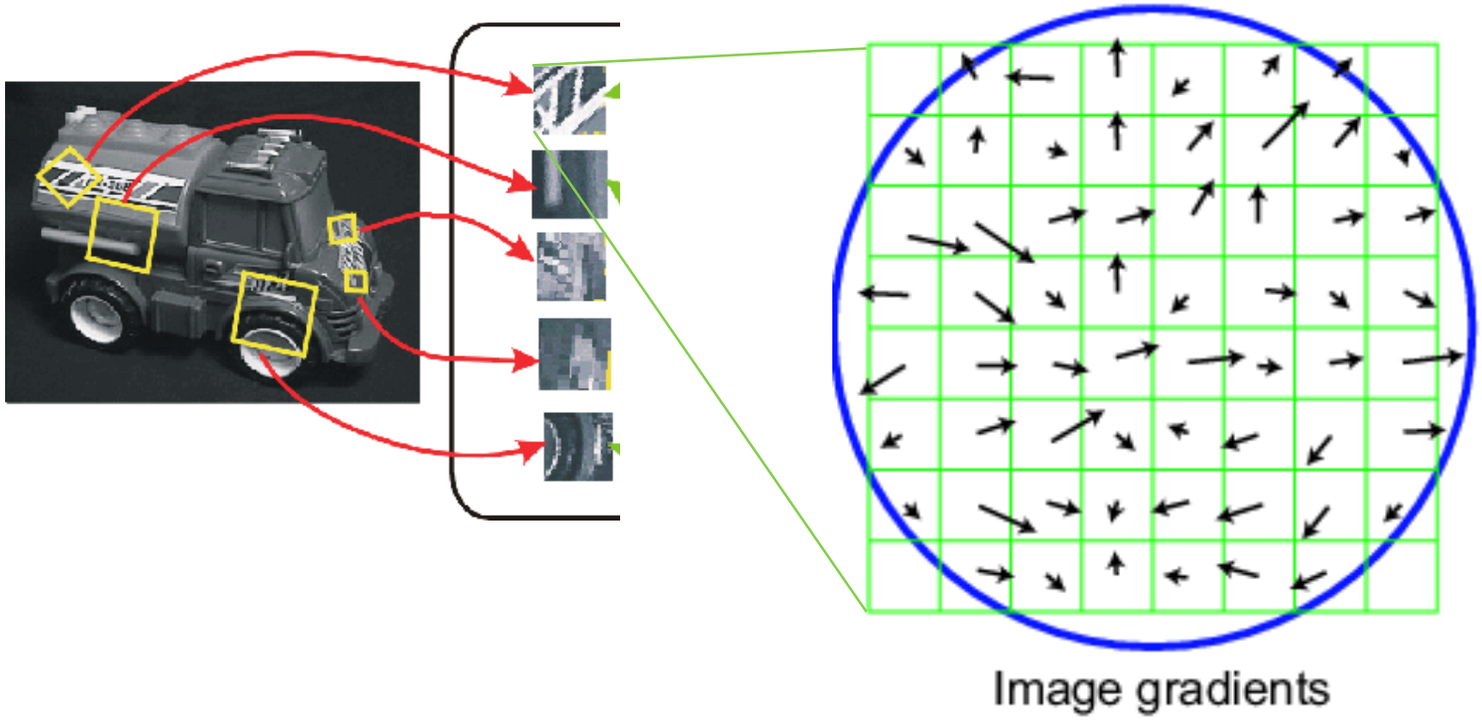
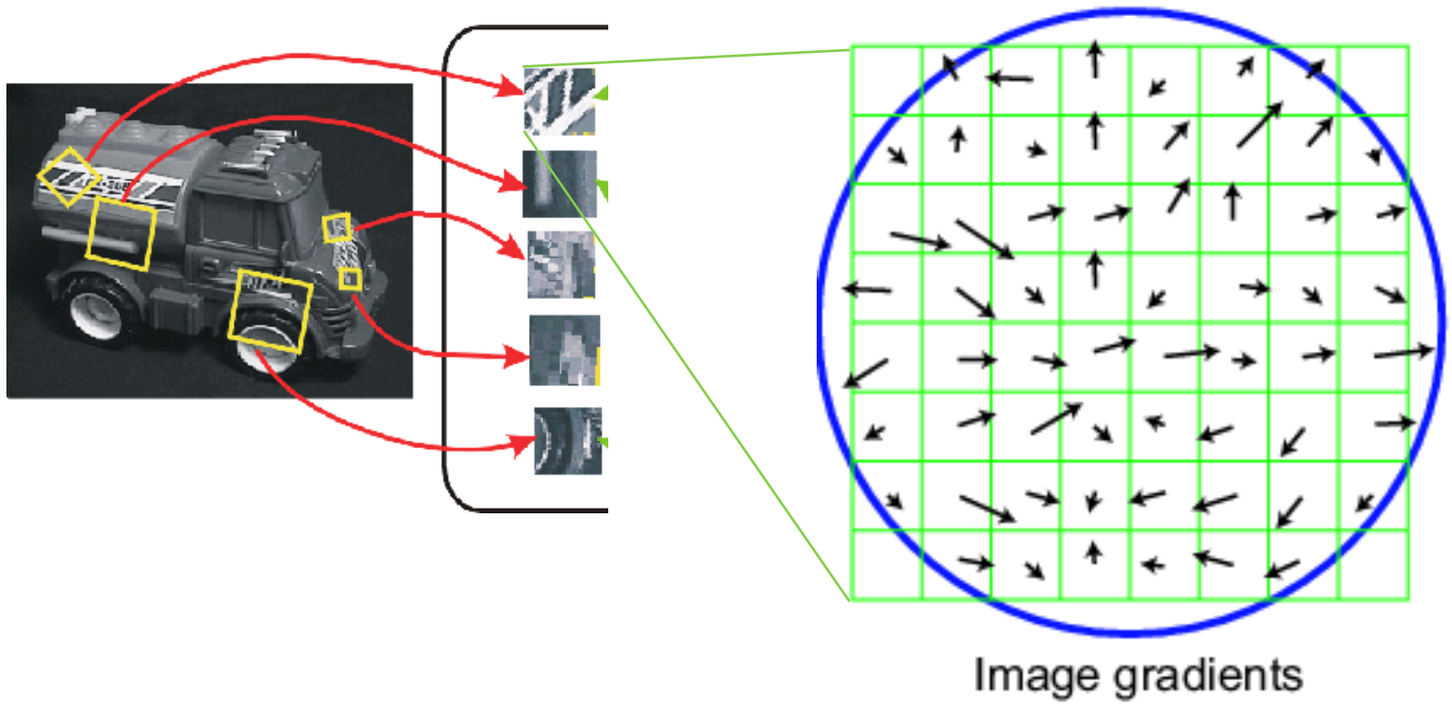


Image gradients

SIFT — Vector Formation

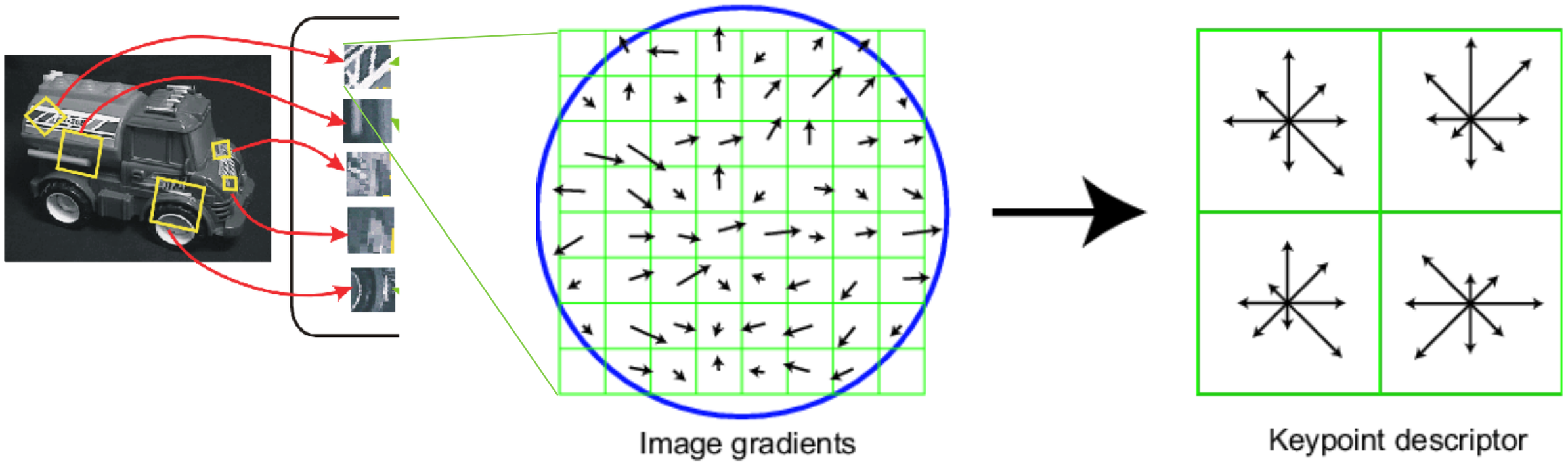
- 4x4 array of gradient orientation histograms
  - not really histogram, weighted by magnitude
- 8 orientations x 4x4 array = 128 dimensions
- Motivation: some sensitivity to spatial layout, but not too much.



showing only 2x2 here but is 4x4

Reduce Effect of Illumination

- 128-dim vector normalized to 1
- Threshold gradient magnitudes to avoid excessive influence of high gradients
  - after normalization, clamp gradients >0.2
  - renormalize

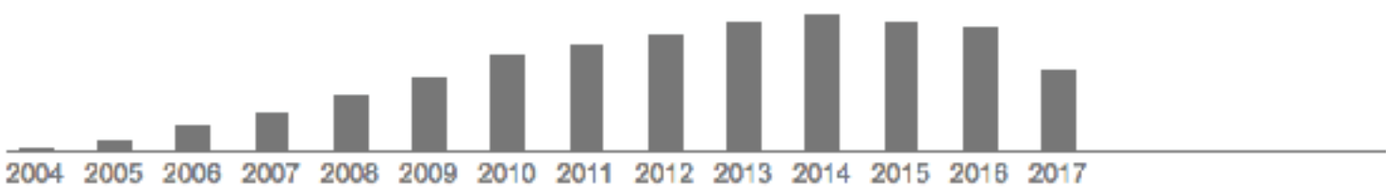


showing only 2x2 here but is 4x4

SIFT Impact

Distinctive image features from scale-invariant keypoints

Authors	David G Lowe
Publication date	2004/11/1
Journal	International journal of computer vision
Volume	60
Issue	2
Pages	91-110
Publisher	Springer Netherlands
Description	This paper presents a method for extracting distinctive invariant features from images that can be used to perform reliable matching between different views of an object or scene. The features are invariant to image scale and rotation, and are shown to provide robust matching across a substantial range of affine distortion, change in 3D viewpoint, addition of noise, and change in illumination. The features are highly distinctive, in the sense that a single feature can be correctly matched with high probability against a large database of features from ...
Total citations	Cited by 43944



A good SIFT features tutorial:

http://www.cs.toronto.edu/~jepson/csc2503/tutSIFT04.pdf

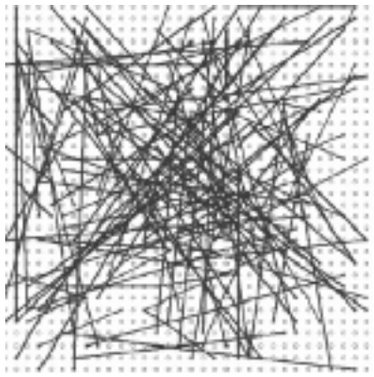
By Estrada, Jepson, and Fleet.

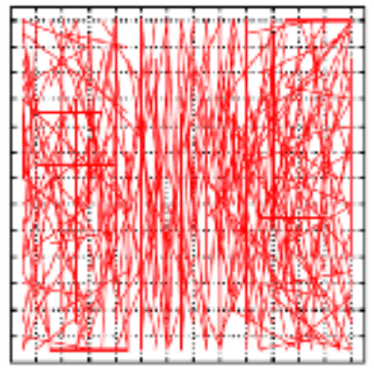
The original SIFT paper:

http://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf

Binary Descriptors

BRIEF, BRISK, ORB, FREAK

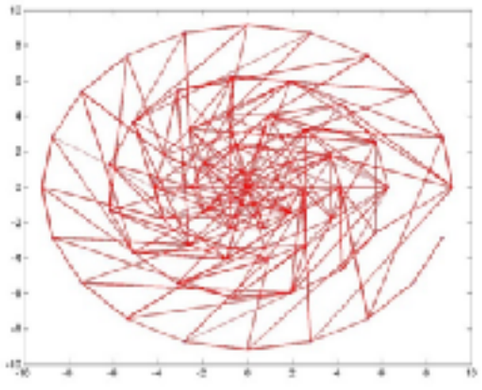




- Extremely efficient computation and comparison
- Encode a patch as a binary string using only pairwise intensity comparisons
  - Sampling pattern around each key point
  - Sampling pairs
  - Descriptor is given by a binary string:

$$F = \sum_{0 \le a \le N} 2^a T(P_a)$$

$$T(P_a) = \begin{cases} 1 & \text{if } I(P_a^{r1}) > I(P_a^{r2}) \\ 0 & \text{otherwise} \end{cases}$$



- Matching using Hamming distance: $L = \sum_{0 \le a \le N} XOR(F_a^1, F_a^2)$

Time per keypoint	SIFT	SURF	BRISK	FREAK
Description in [ms]	2.5	1.4	0.031	0.018
Matching time in [ns]	1014	566	36	25

Table 1: Computation time on 800x600 images where approximately 1500 keypoints are detected per image. The computation times correspond to the description and matching of all keypoints.

A. Alahi, R. Ortiz, and P. Vandergheynst. FREAK: Fast Retina Keypoint. In IEEE Conference on Computer Vision and Pattern Recognition,

Stitching a pair of image

We have:

- Well-localized features
- Distinctive descriptor

Now we need to:

- Match pairs of feature points in different images
- Robustly compute homographies
  (in the presence of errors/outliers)