特征匹配与光流

陶煜波，鲍虎军，章国锋
浙江大学CAD&CG国家重点实验室

How can we find corresponding points?

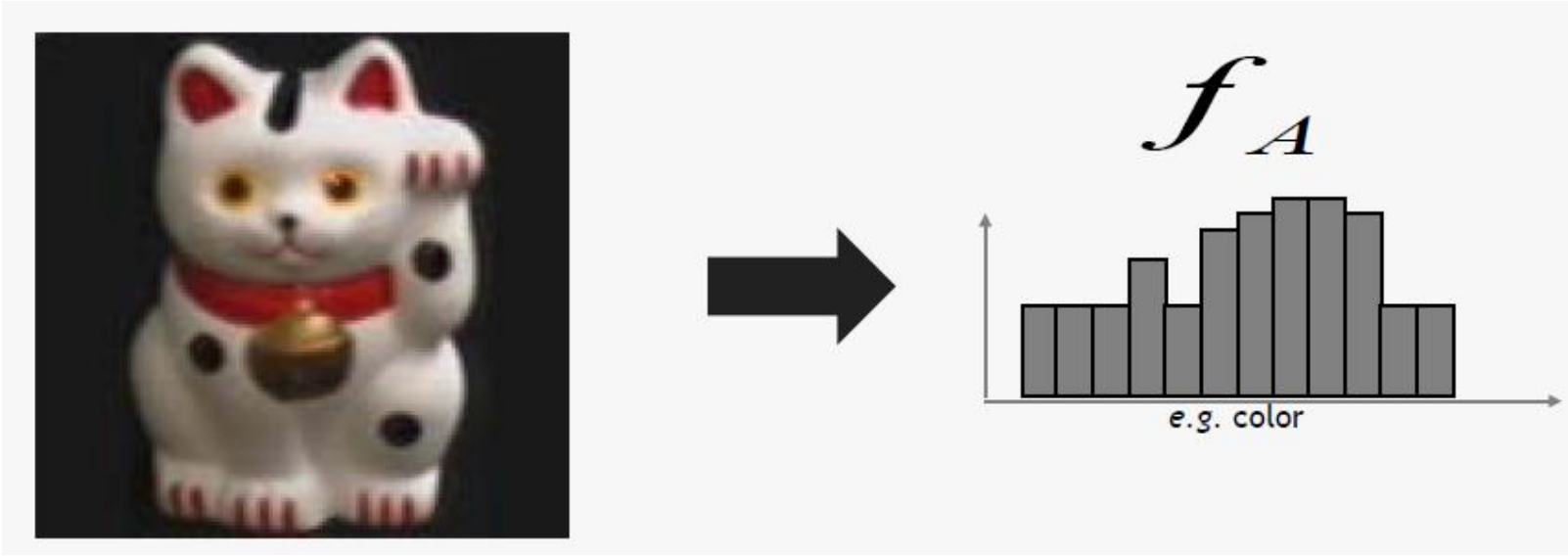
根据匹配程度分为：

- 稀疏对应——特征点匹配
- 稠密对应——光流估计

- What is a feature?
  - Point
  - Line: straight line, curve,…
  - Edge: 2D, 3D
  - Corner
  - Shape: rectangle, circle, ellipse, sphere,…
  - Texture
  - Object
  - Motion…
- Invariance
  - View point (scale, orientation, translation)
  - Lighting condition
  - Object deformations
  - Partial occlusion

What is feature extraction?

- From image space to feature space:
  - Dimensionality reduction
  - Find a latent semantic space where the pattern can be recognized easily
  - Feature detection and representation

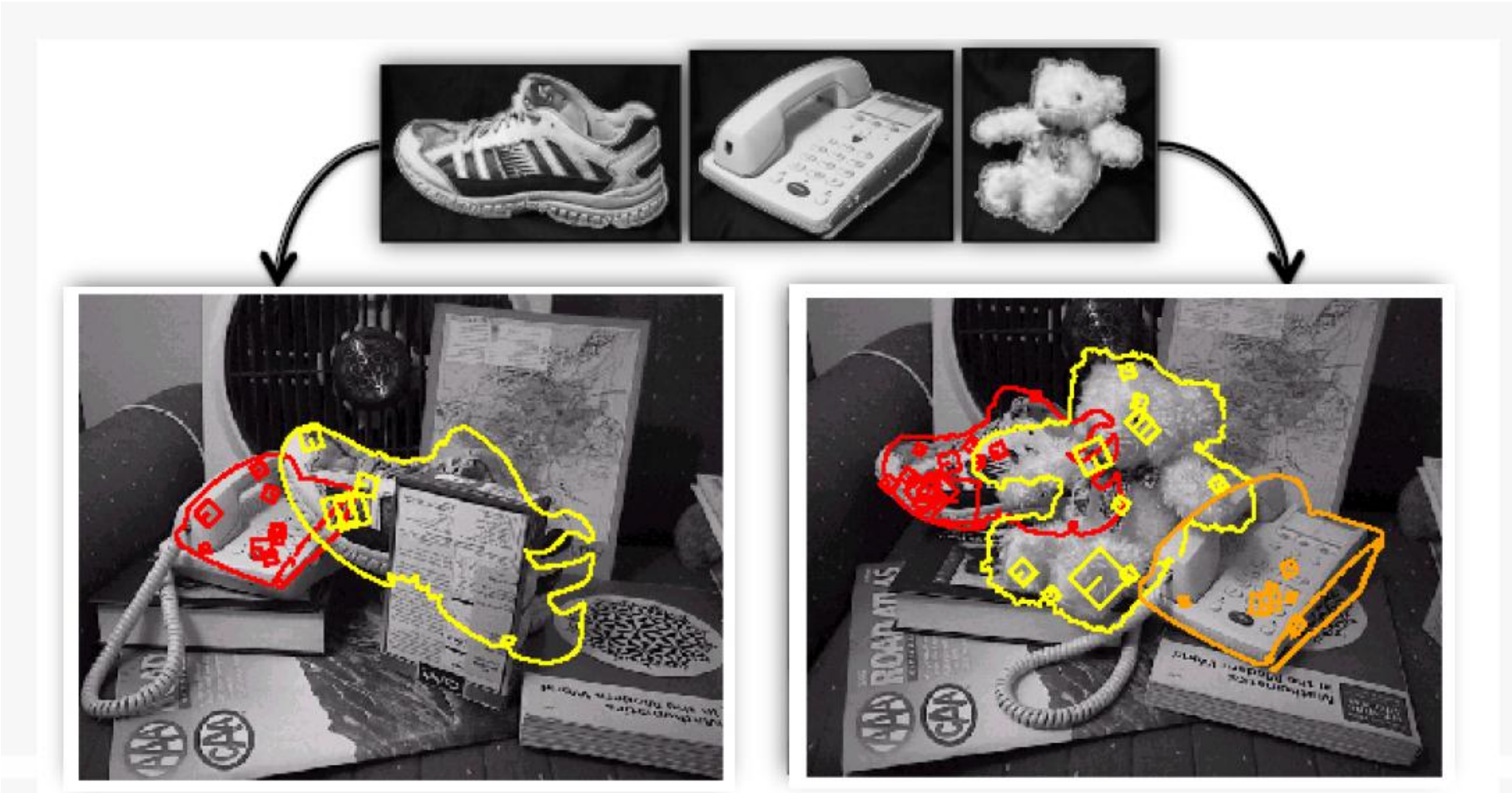


Motivation

- For object recognition

Motivation

- For object recognition

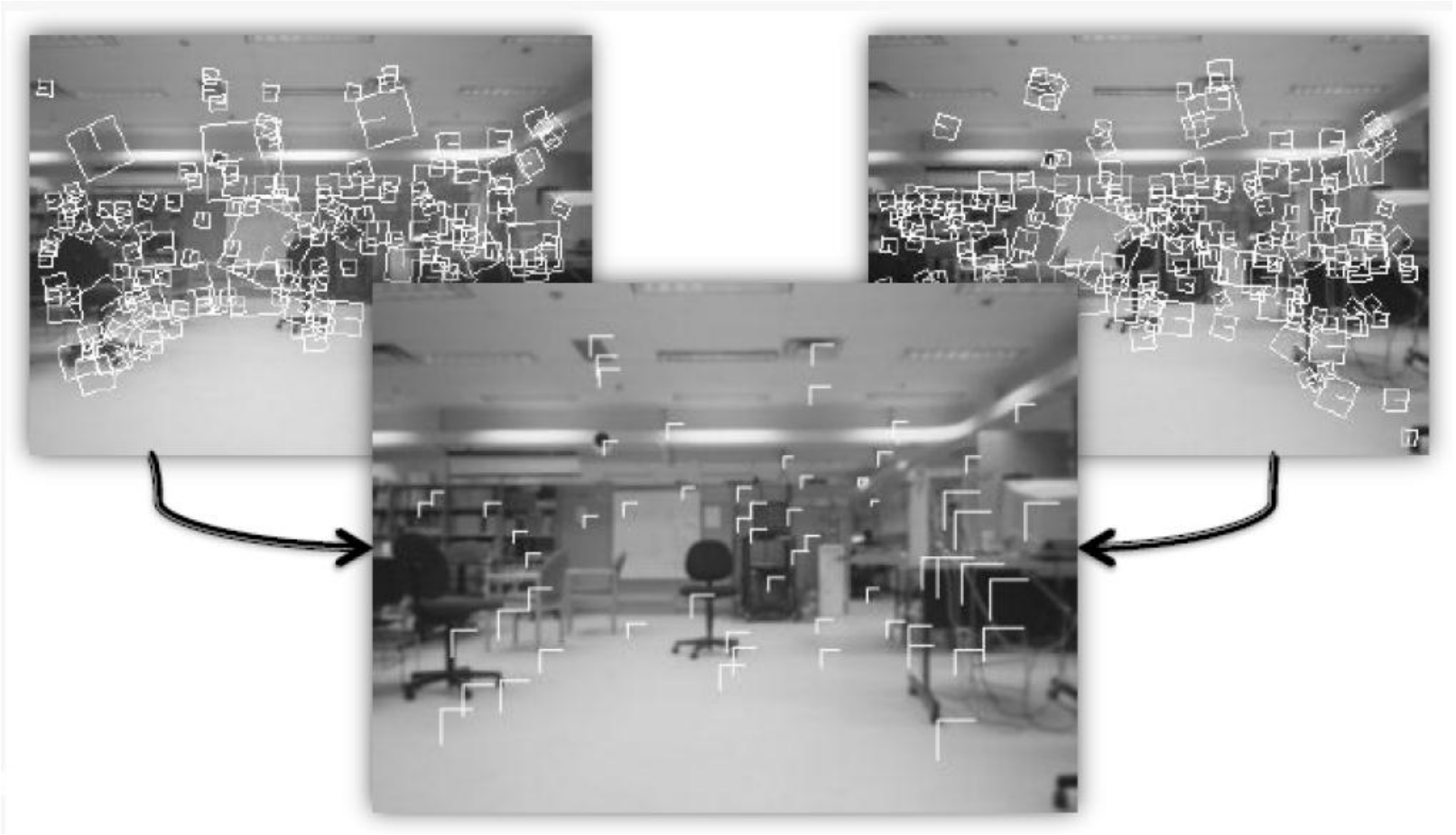


Motivation

- Panorama creation

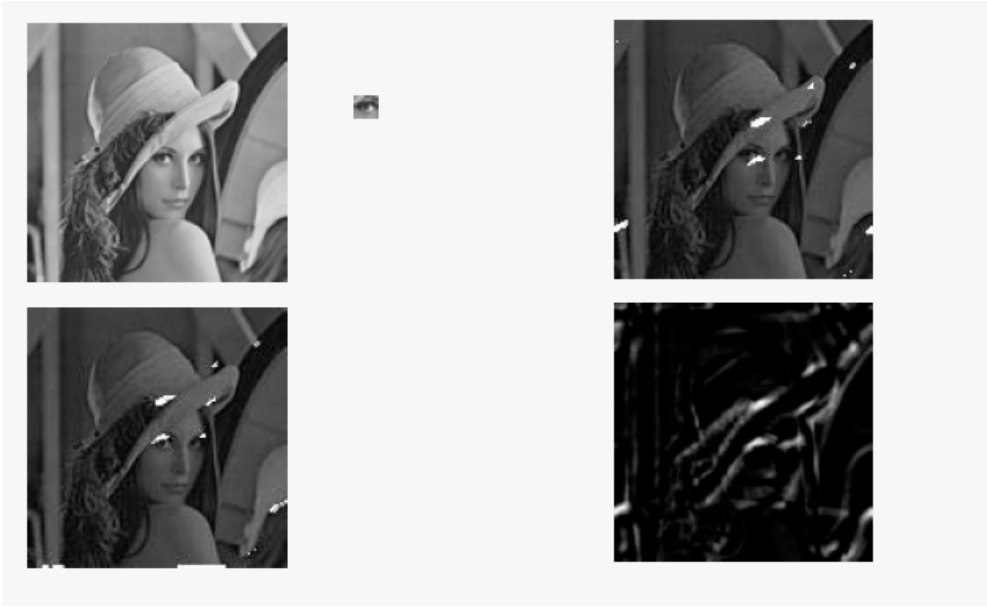
Motivation

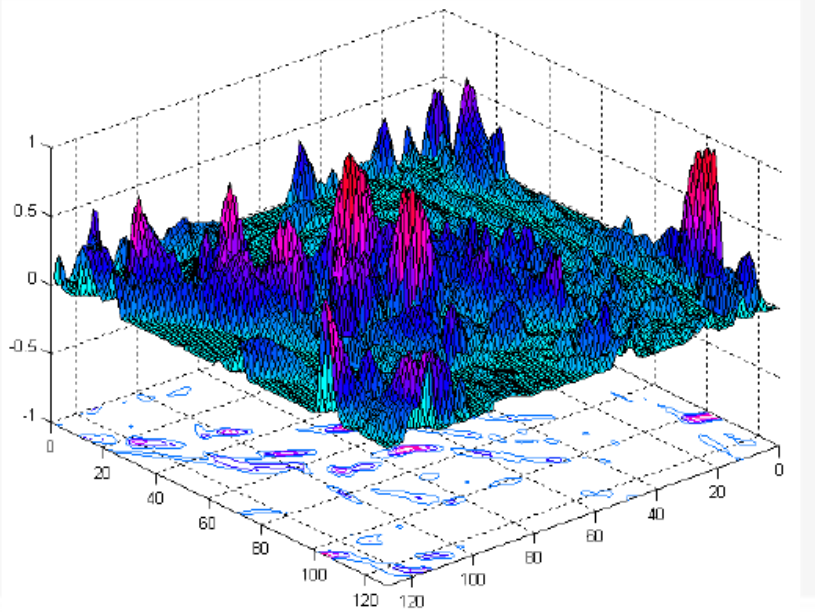
- Stereo pair matching



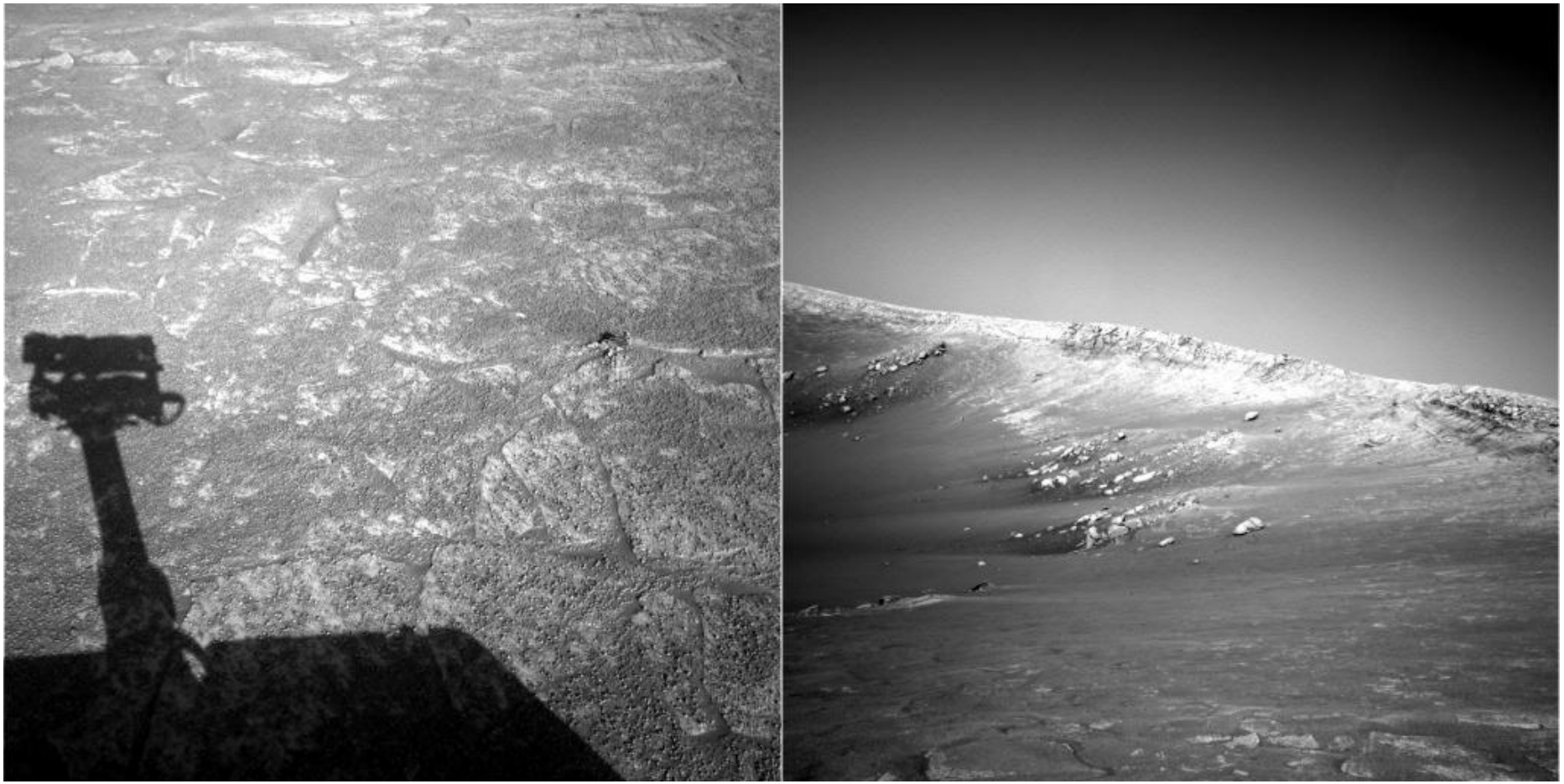
Motivation

- Template matching
  - Pick a template-rectangular/square region of an image
  - Goal: find it in the same image/images of the same from different viewpoint





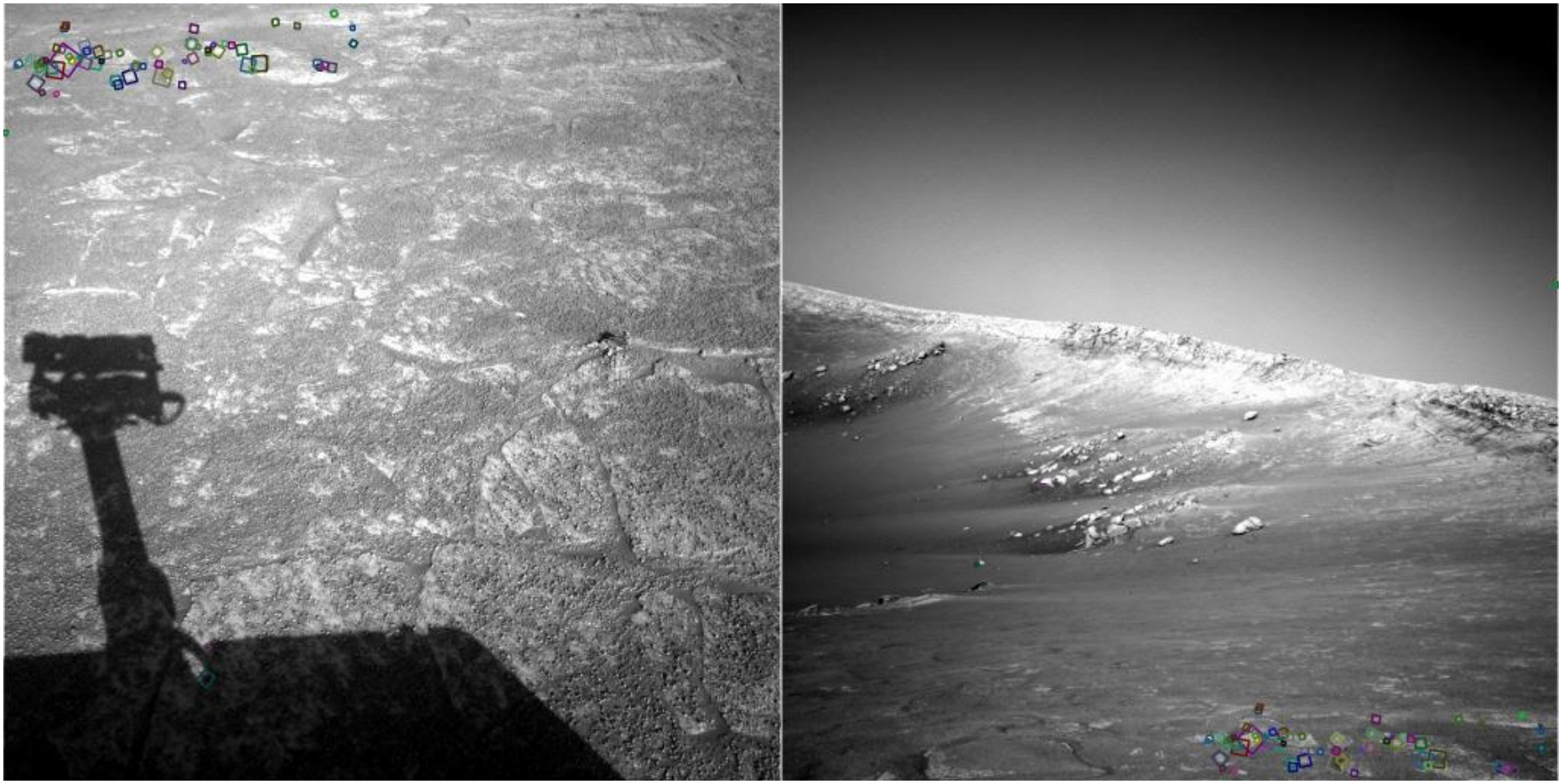
Find corresponding points-Not always easy



NASA Mars Rover images

from “http://szeliski.org/Book/”

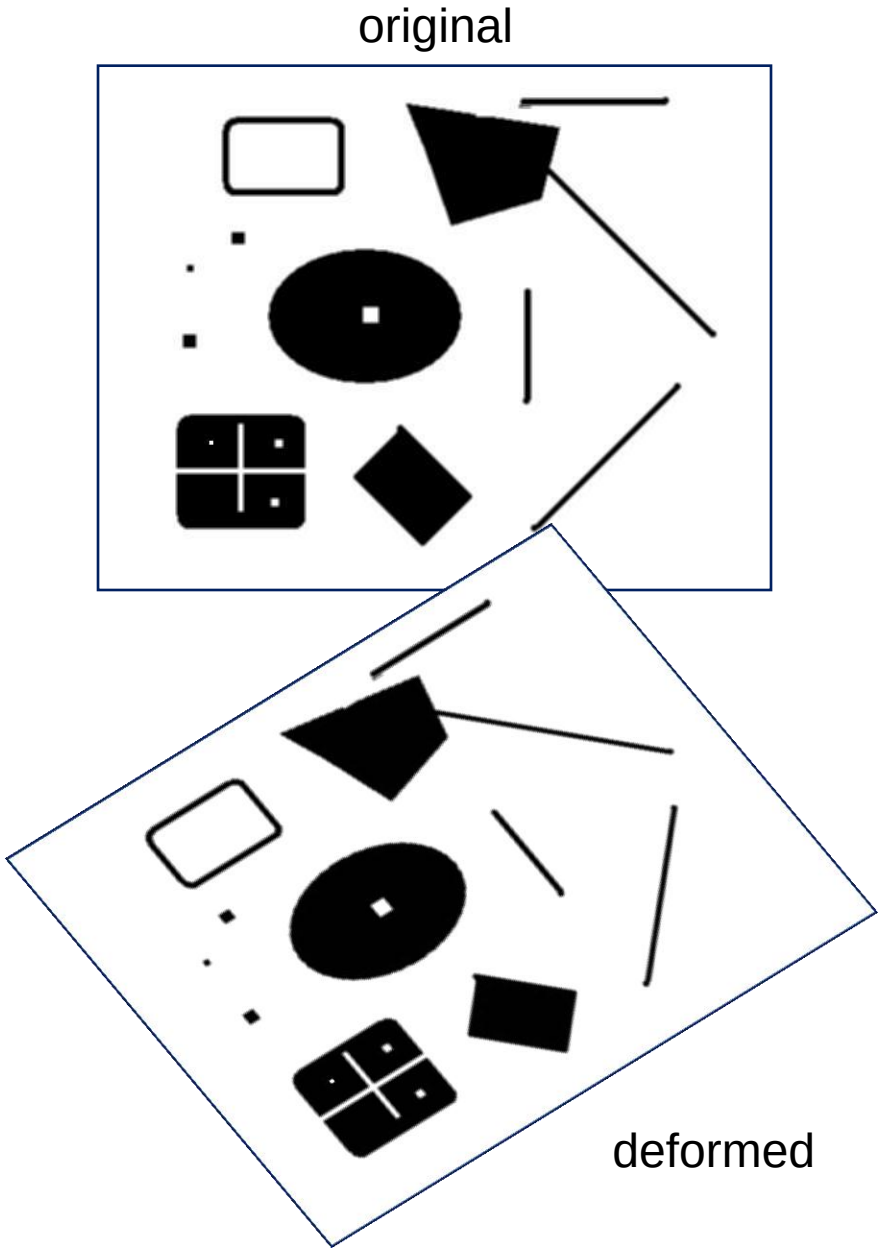
Answer below (look for tiny colored squares…)



NASA Mars Rover images
with SIFT feature matches
Figure by Noah Snavely

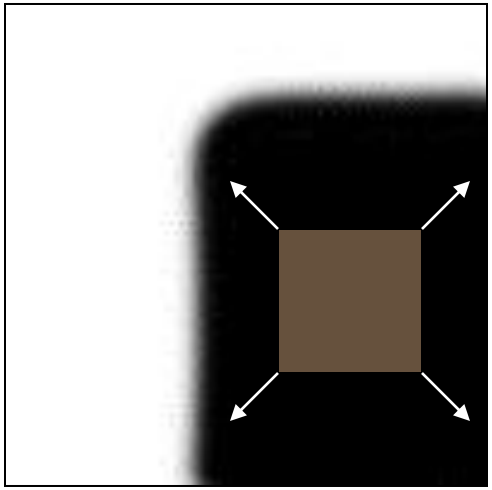
Interest points

- Suppose you have to click on some point, go away and come back after I deform the image, and click on the same points again.
  - Which points would you choose?

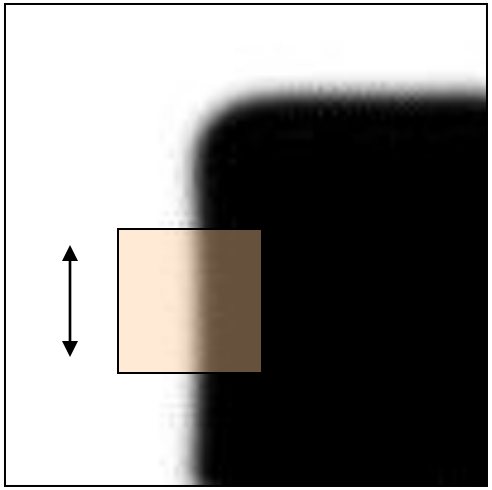


Corners

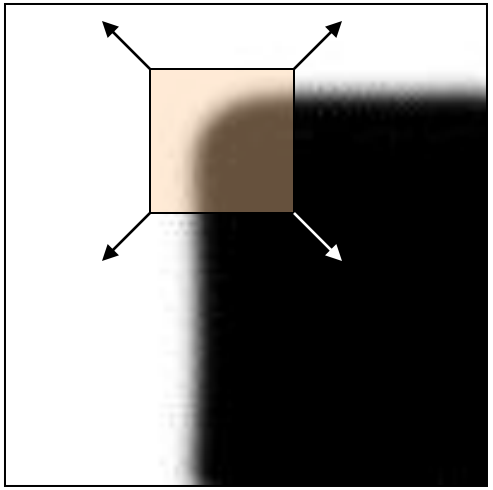
- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity



"flat" region:
no change in all directions

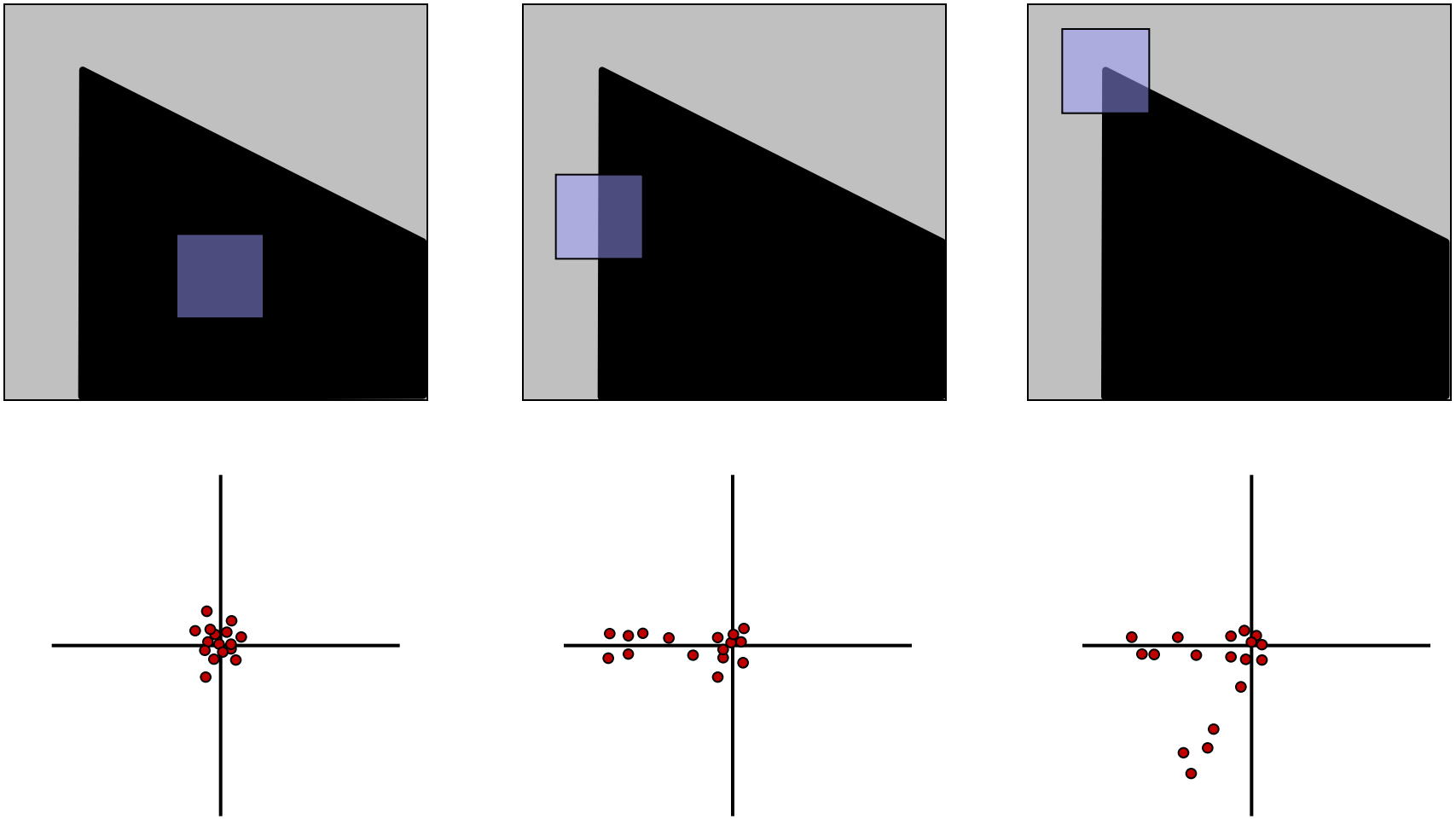


"edge":
no change along the edge direction



"corner":
significant change in all directions

Let's look at the gradient distributions



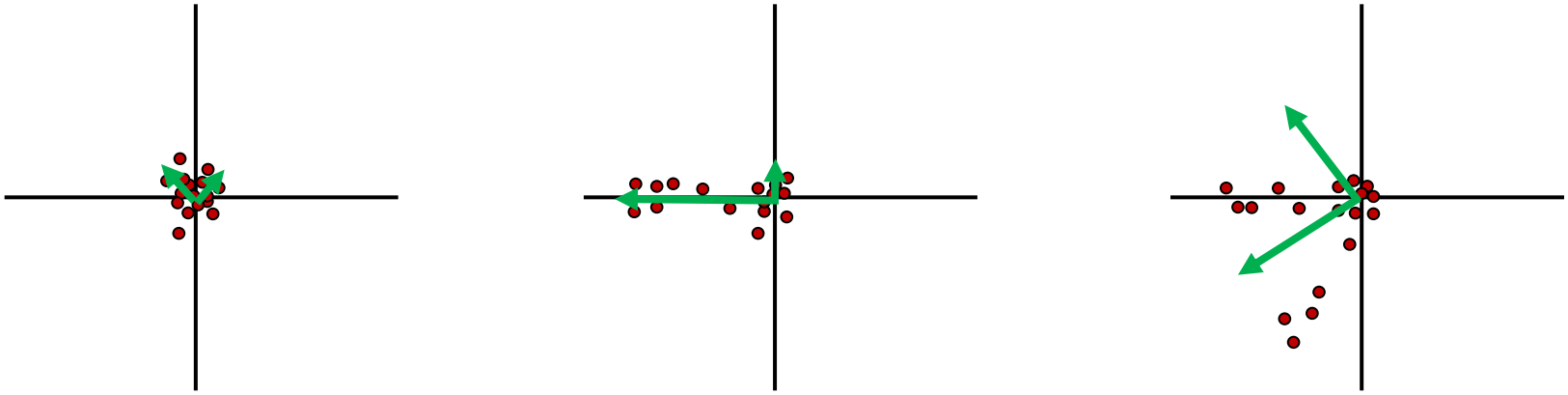
Principle Component Analysis

Principal component is the direction of highest variance.

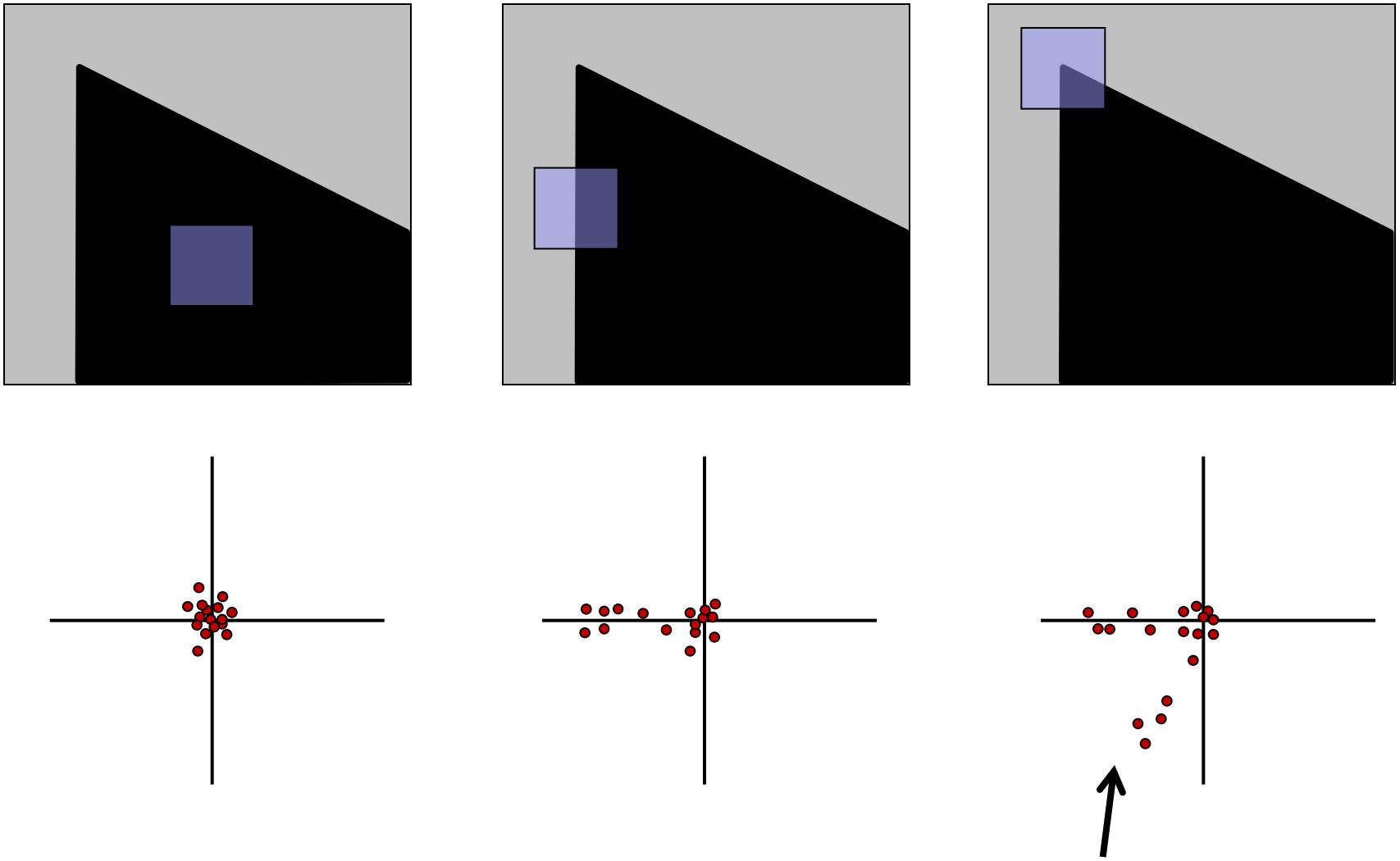
Next, highest component is the direction with highest variance *orthogonal* to the previous components.

How to compute PCA components:
1. Subtract off the mean for each data point.
2. Compute the covariance matrix.
3. Compute eigenvectors and eigenvalues.
4. The components are the eigenvectors ranked by the eigenvalues.

$$Hx = \lambda x$$



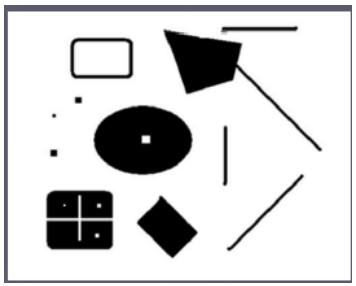
Corners have …

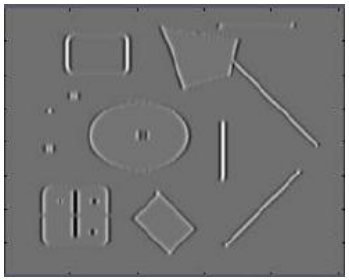


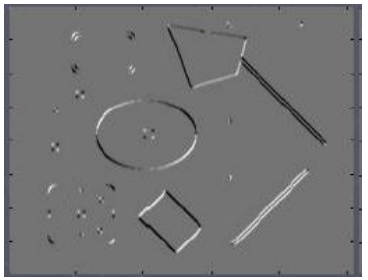
Second Moment Matrix

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

2 x 2 matrix of image derivatives (averaged in neighborhood of a point).







Notation:

$$I_x \Leftrightarrow \frac{\partial I}{\partial x} \qquad I_y \Leftrightarrow \frac{\partial I}{\partial y} \qquad I_x I_y \Leftrightarrow \frac{\partial I}{\partial x}\frac{\partial I}{\partial y}$$

The math

To compute the eigenvalues:

1. Compute the covariance matrix.

$$H = \sum_{(u,v)} w(u,v) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \qquad I_x = \frac{\partial f}{\partial x}, I_y = \frac{\partial f}{\partial y}$$

Typically Gaussian weights

2. Compute eigenvalues.

$$H = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \qquad \lambda_{\pm} = \frac{1}{2}\left((a+d) \pm \sqrt{4bc + (a-d)^2}\right)$$

Corner Response Function

- Computing eigenvalues are expensive
- Harris corner detector uses the following alternative

$$R = det(M) - \alpha \cdot trace(M)^2$$

Reminder:

$$det\left(\begin{bmatrix} a & b \\ c & d \end{bmatrix}\right) = ad - bc \qquad trace\left(\begin{bmatrix} a & b \\ c & d \end{bmatrix}\right) = a + d$$

Harris detector: Steps

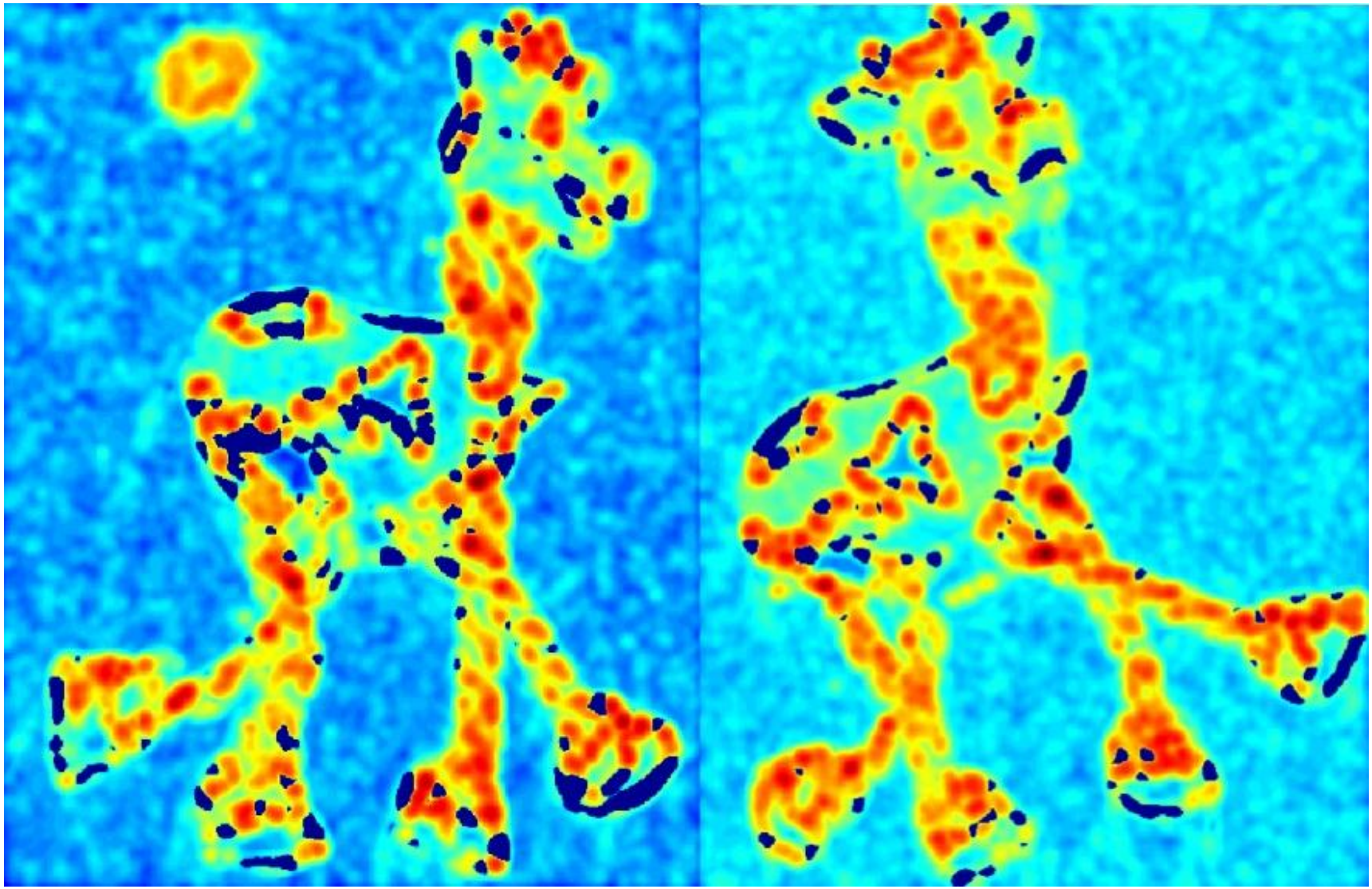
1. Compute Gaussian derivatives at each pixel
2. Compute second moment matrix *M* in a Gaussian window around each pixel
3. Compute corner response function *R*
4. Threshold *R*
5. Find local maxima of response function (nonmaximum suppression)

C.Harris and M.Stephens. "A Combined Corner and Edge Detector." *Proceedings of the 4th Alvey Vision Conference*: pages 147—151, 1988.

Harris Detector: Steps

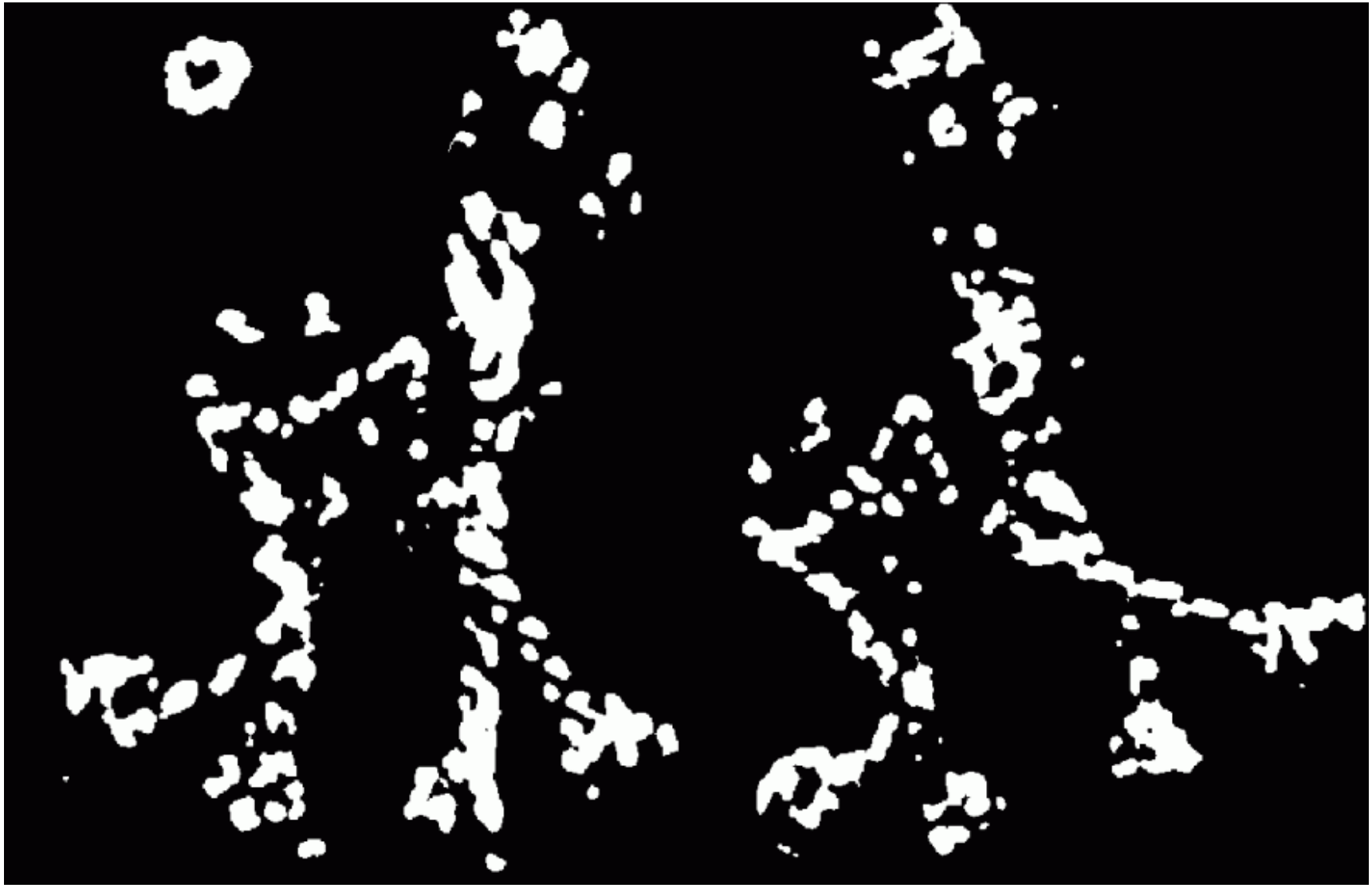
Harris Detector: Steps

Compute corner response *R*



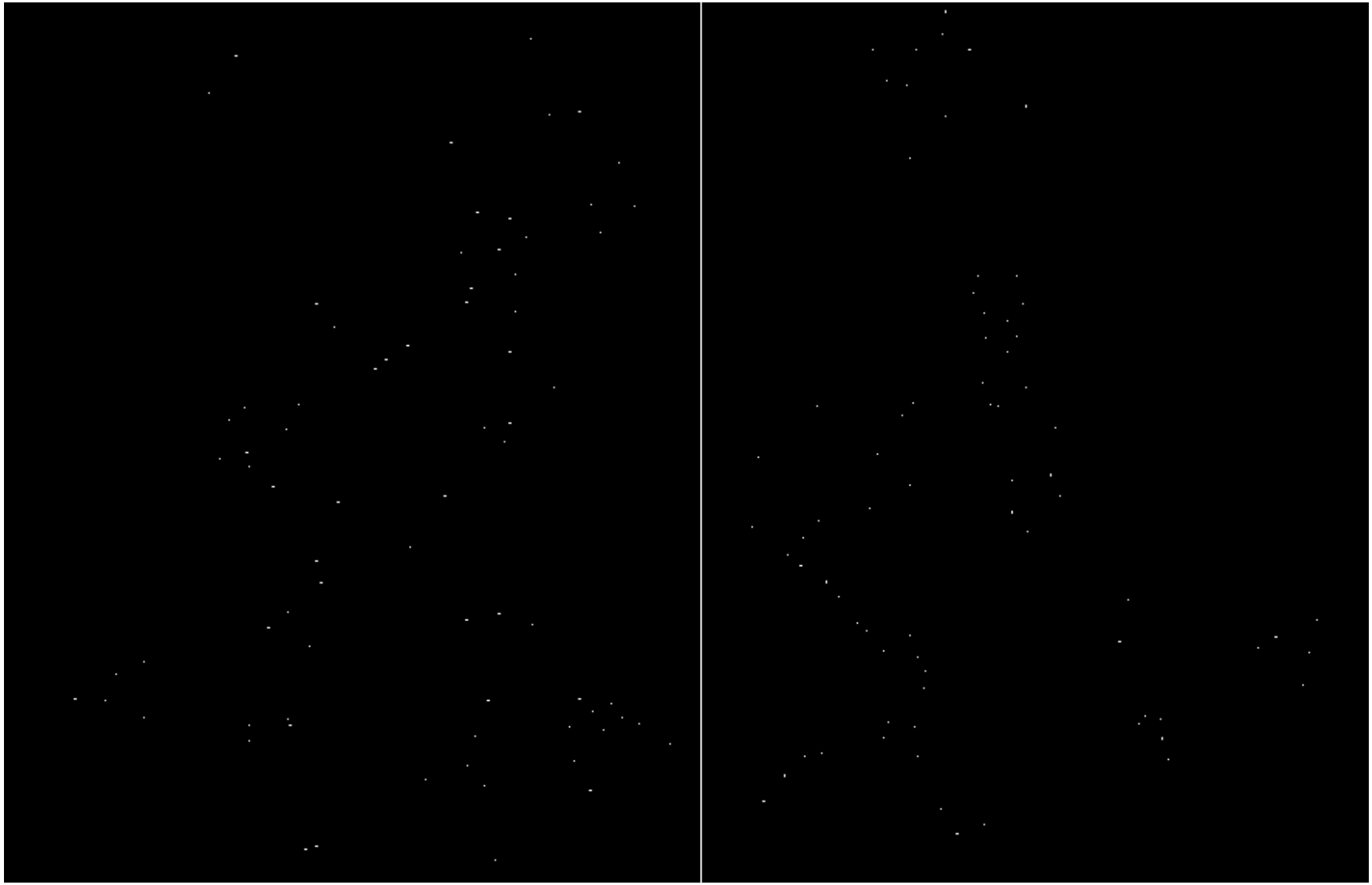
Harris Detector: Steps

Find points with large corner response: *R*>threshold



Harris Detector: Steps

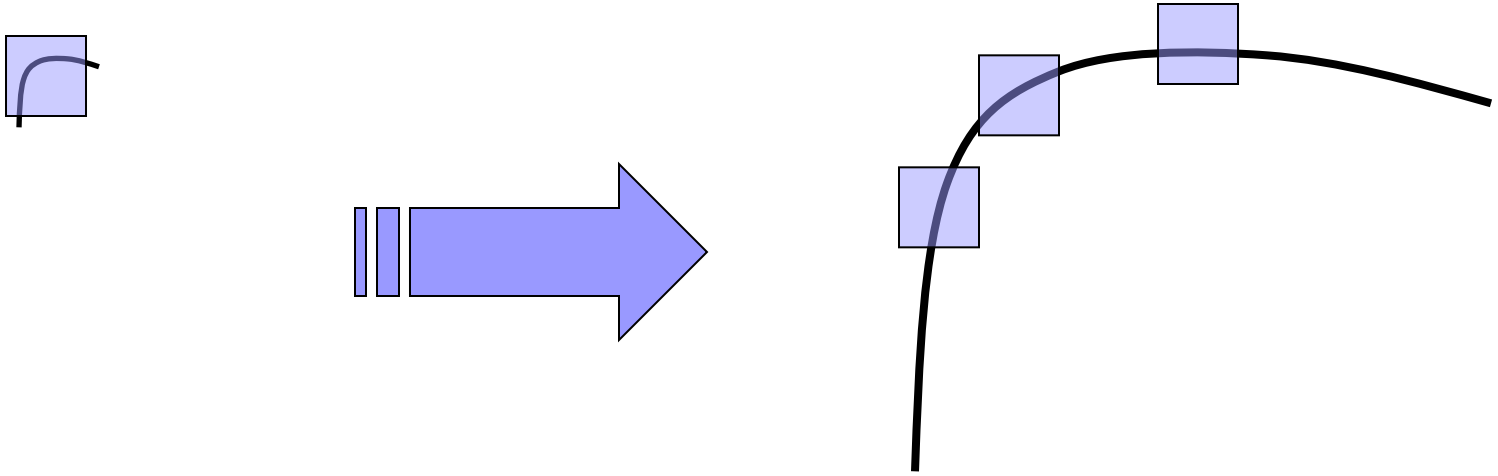
Take only the points of local maxima of *R*



Harris Detector: Steps

Properties of the Harris corner detector

- Translation invariant? Yes
- Rotation invariant? Yes
- Scale invariant? No



Corner !

All points will be classified as edges

Scale

Let's look at scale first:

What is the "best" scale?

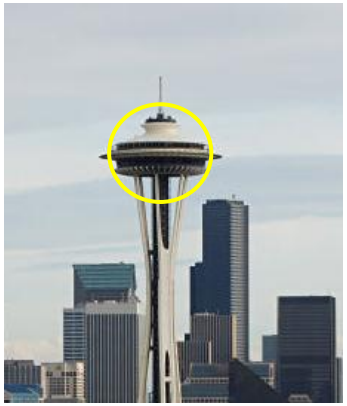
Scale Invariance

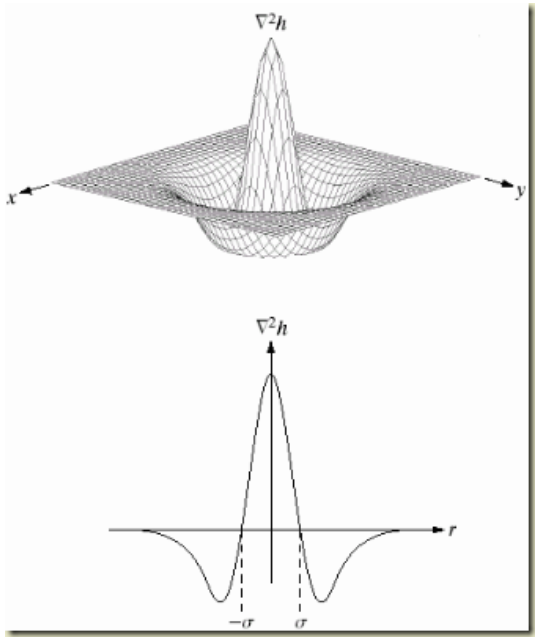
$$f(I_{i_1 \ldots i_m}(x,\sigma)) \;=\; f(I_{i_1 \ldots i_m}(x',\sigma'))$$

How can we independently select interest points in each image, such that the detections are repeatable across different scales?

K. Grauman, B. Leibe

Differences between Inside and Outside



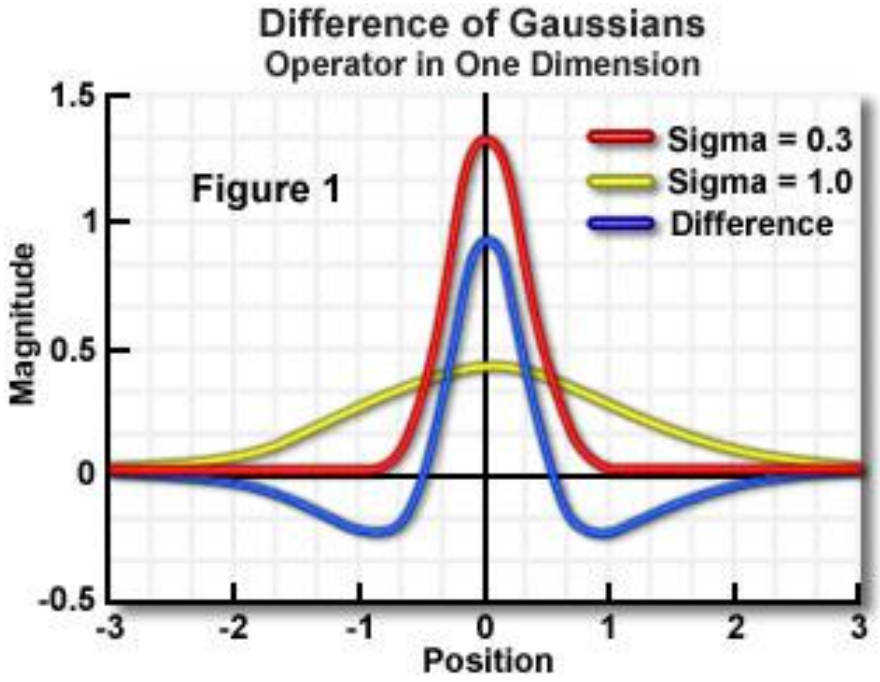


Scale

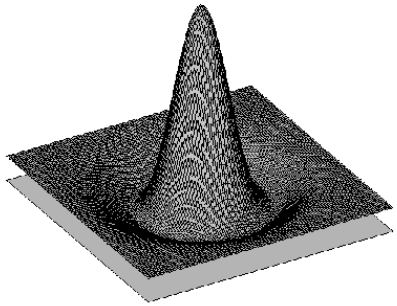
Why Gaussian?

It is invariant to scale change, i.e., $f * \mathcal{G}_\sigma * \mathcal{G}_{\sigma'} = f * \mathcal{G}_{\sigma''}$ and has several other nice properties. Lindeberg, 1994

In practice, the Laplacian is approximated using a Difference of Gaussian (DoG).

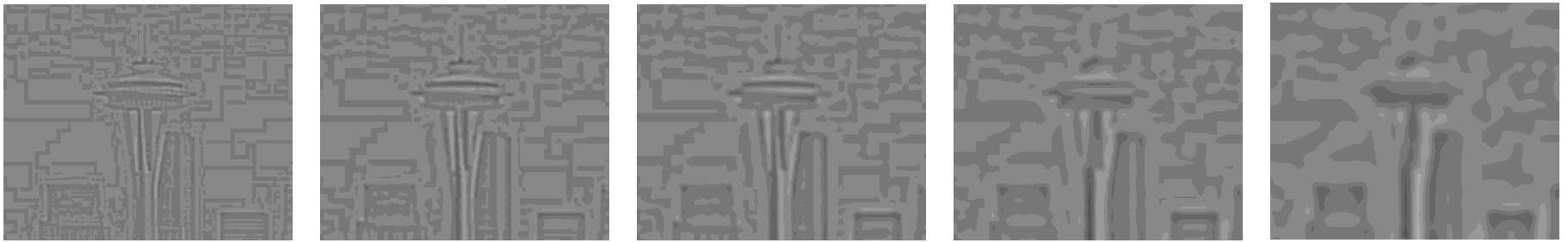


Difference-of-Gaussian (DoG)



K. Grauman, B. Leibe

DoG example



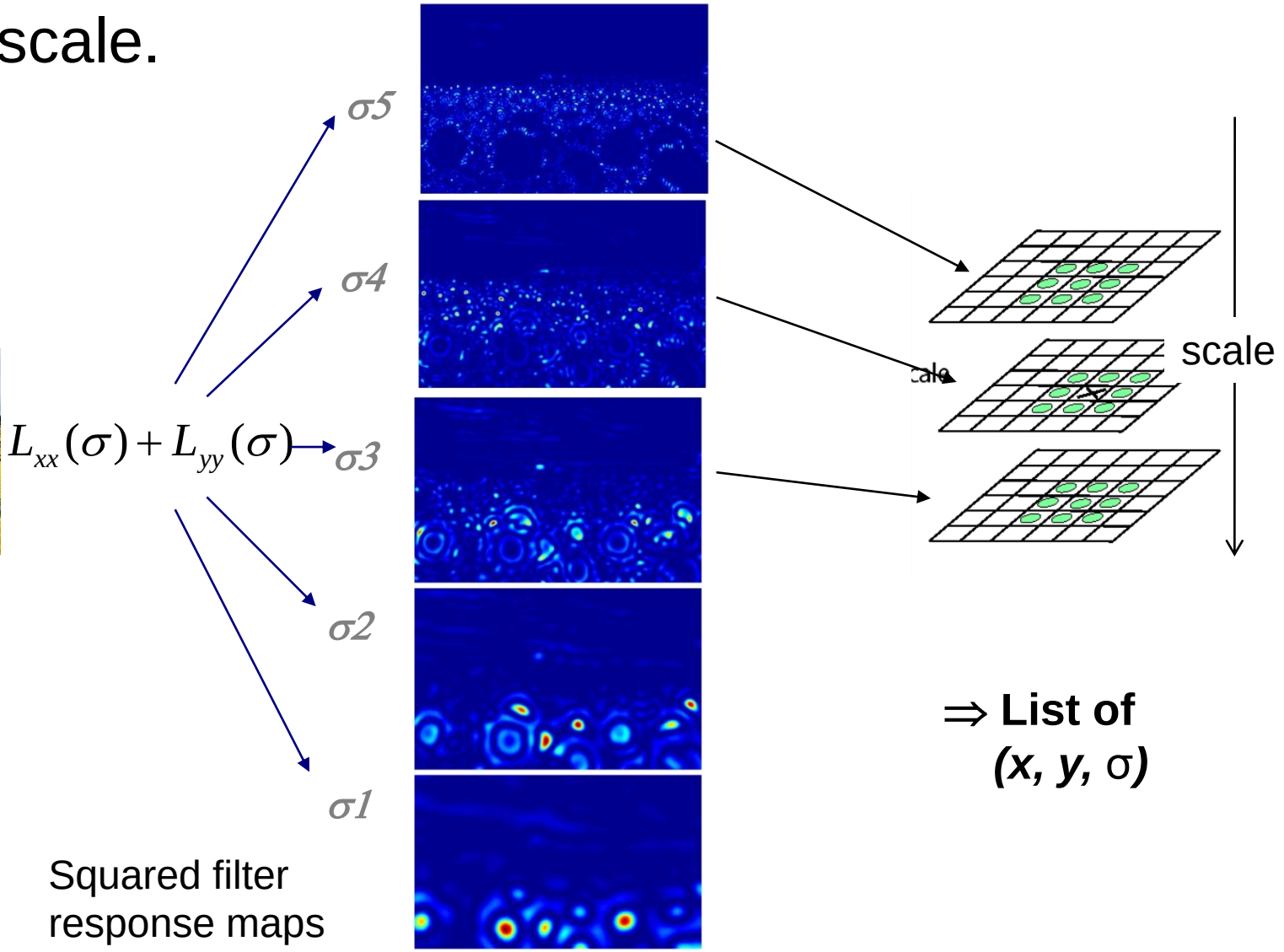
σ = 1

σ = 66

Scale invariant interest points

Interest points are local maxima in both position and scale.

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

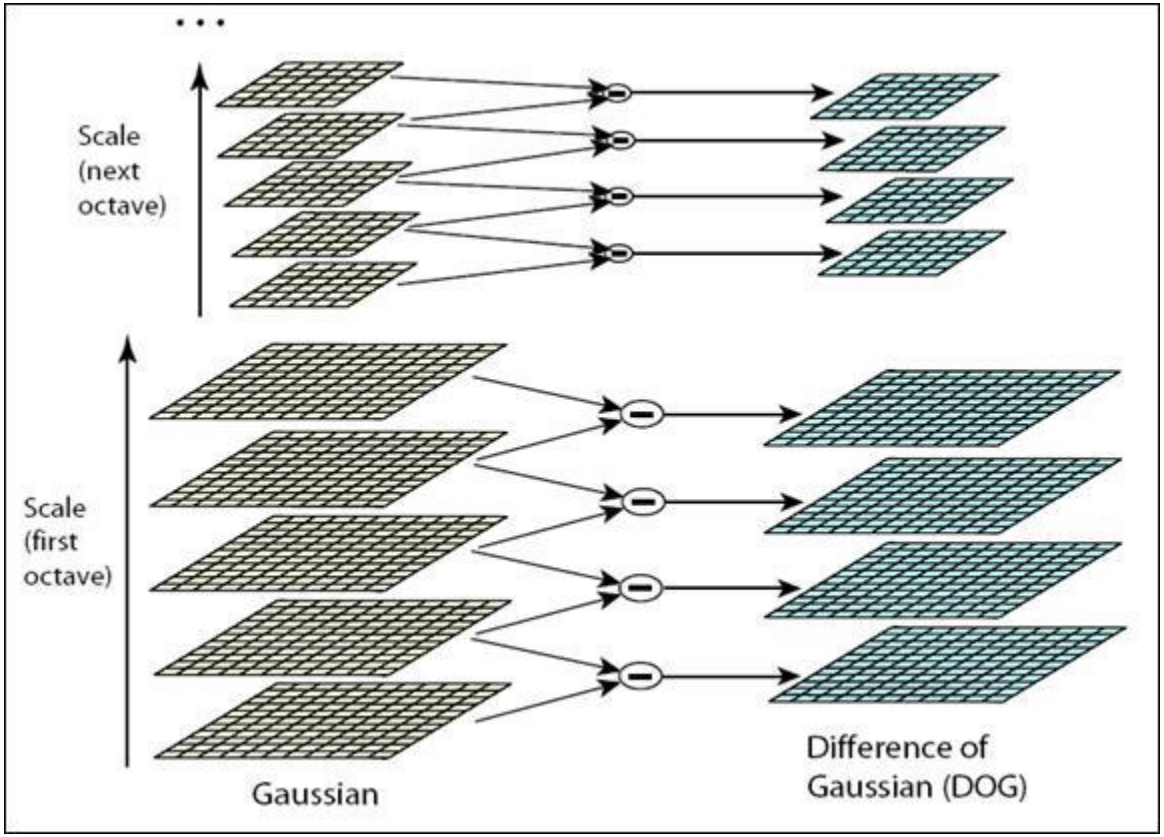


Squared filter response maps

⇒ **List of** ***(x, y, σ)***

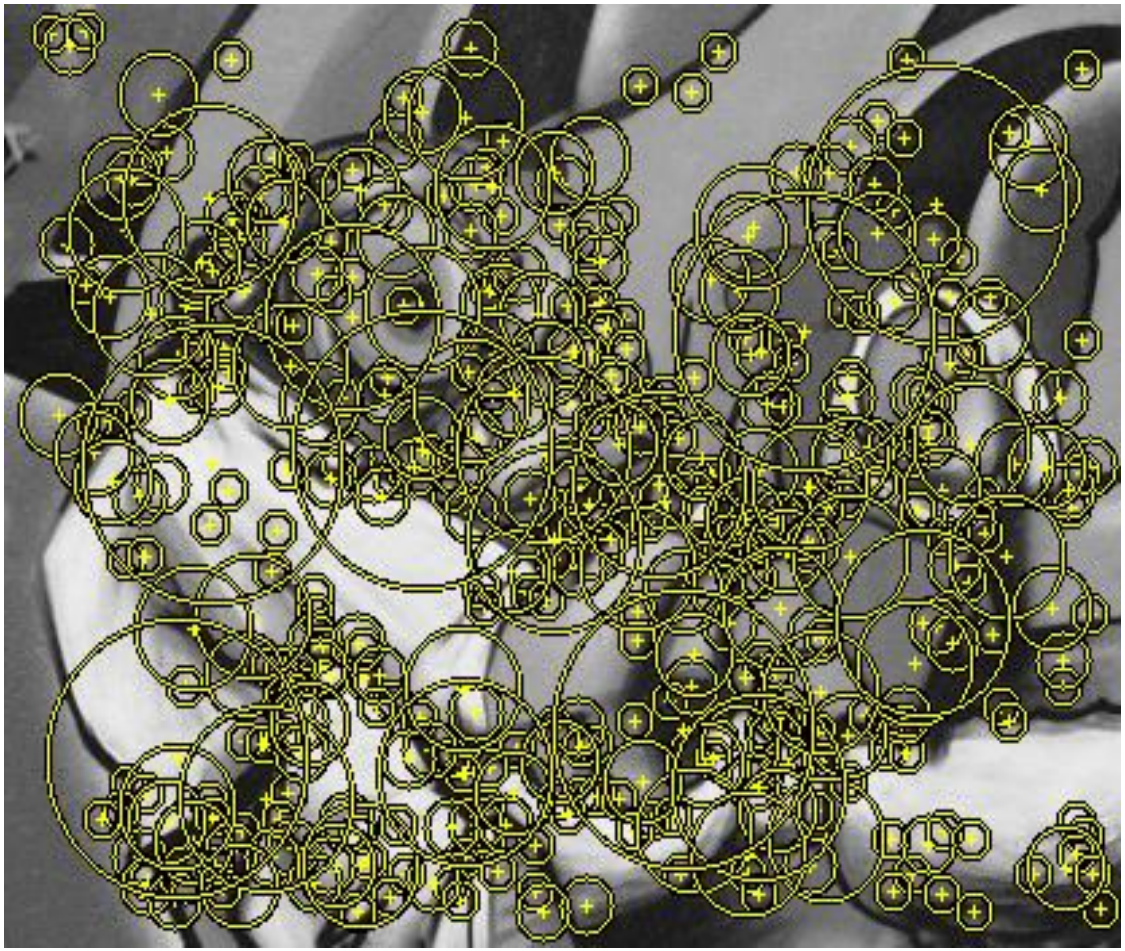
Scale

In practice the image is downsampled for larger sigmas.

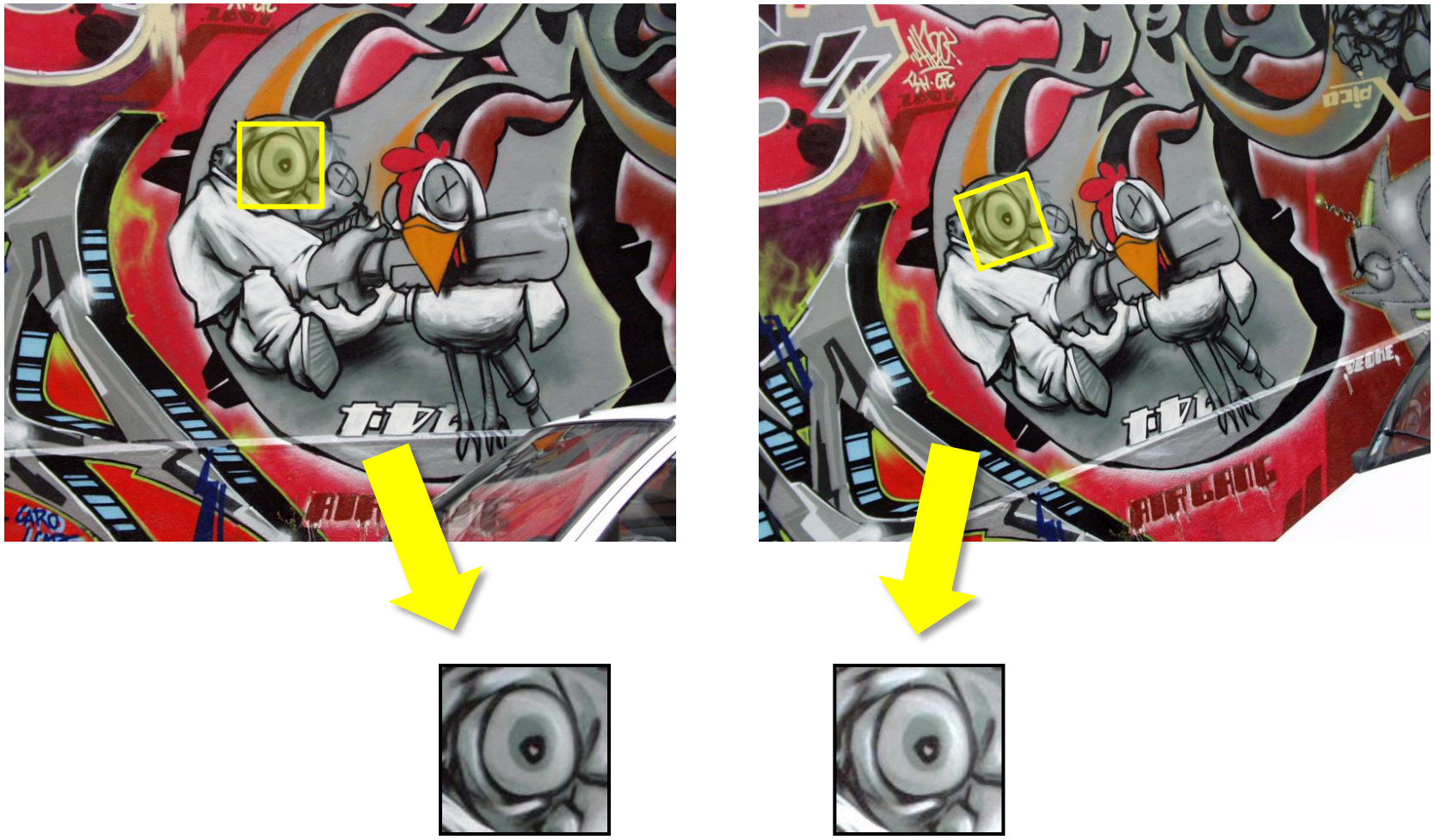


Lowe, 2004.

Results: Difference-of-Gaussian

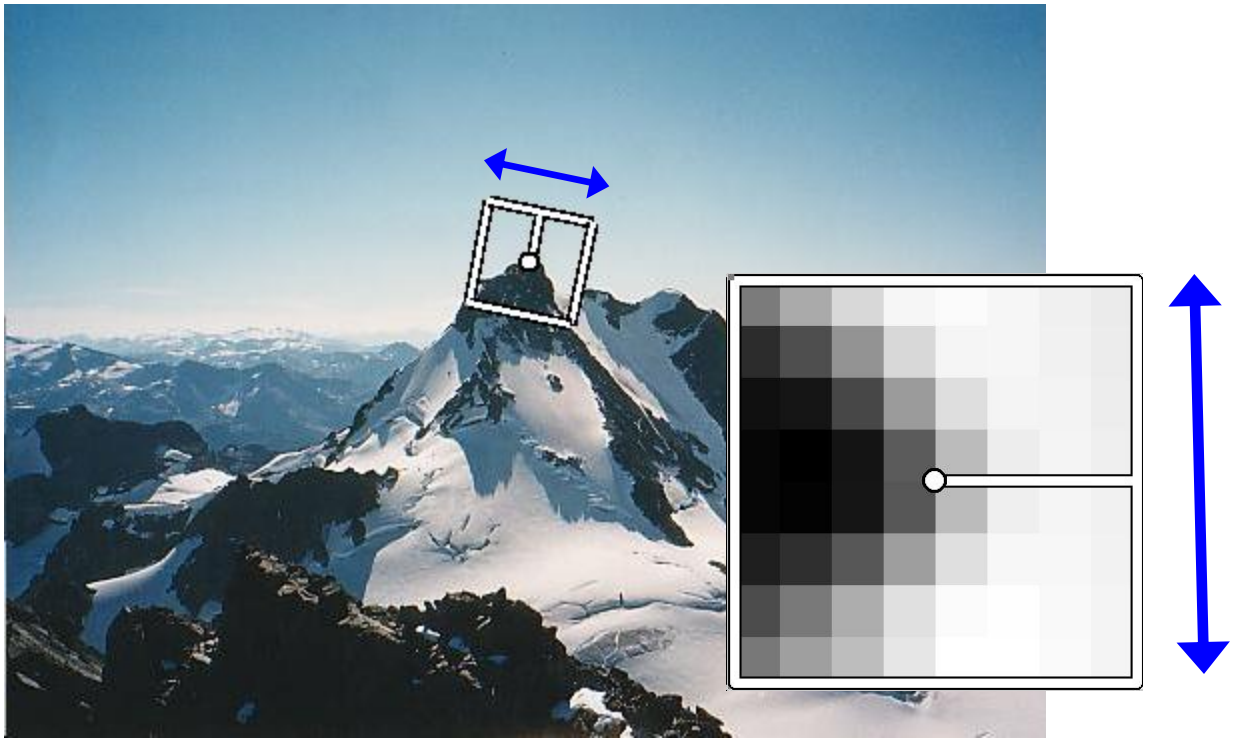


How can we find correspondences?



Similarity transform

Rotation invariance



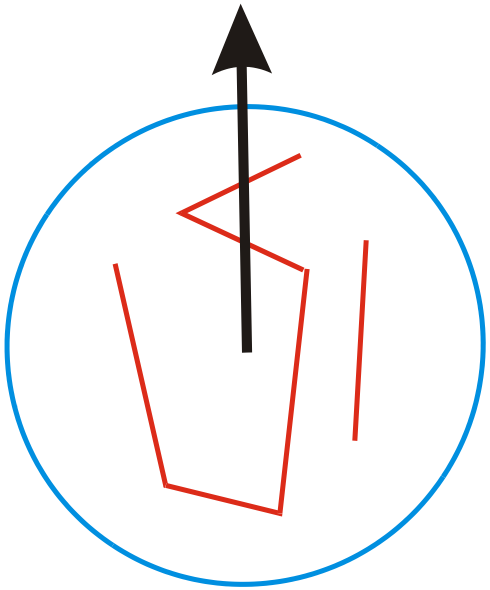
- Rotate patch according to its dominant gradient orientation
- This puts the patches into a canonical orientation.

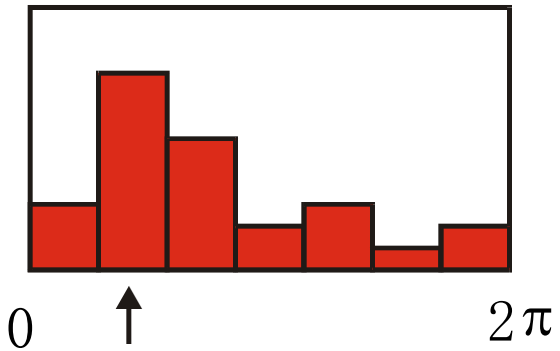
Image from Matthew Brown

Orientation Normalization

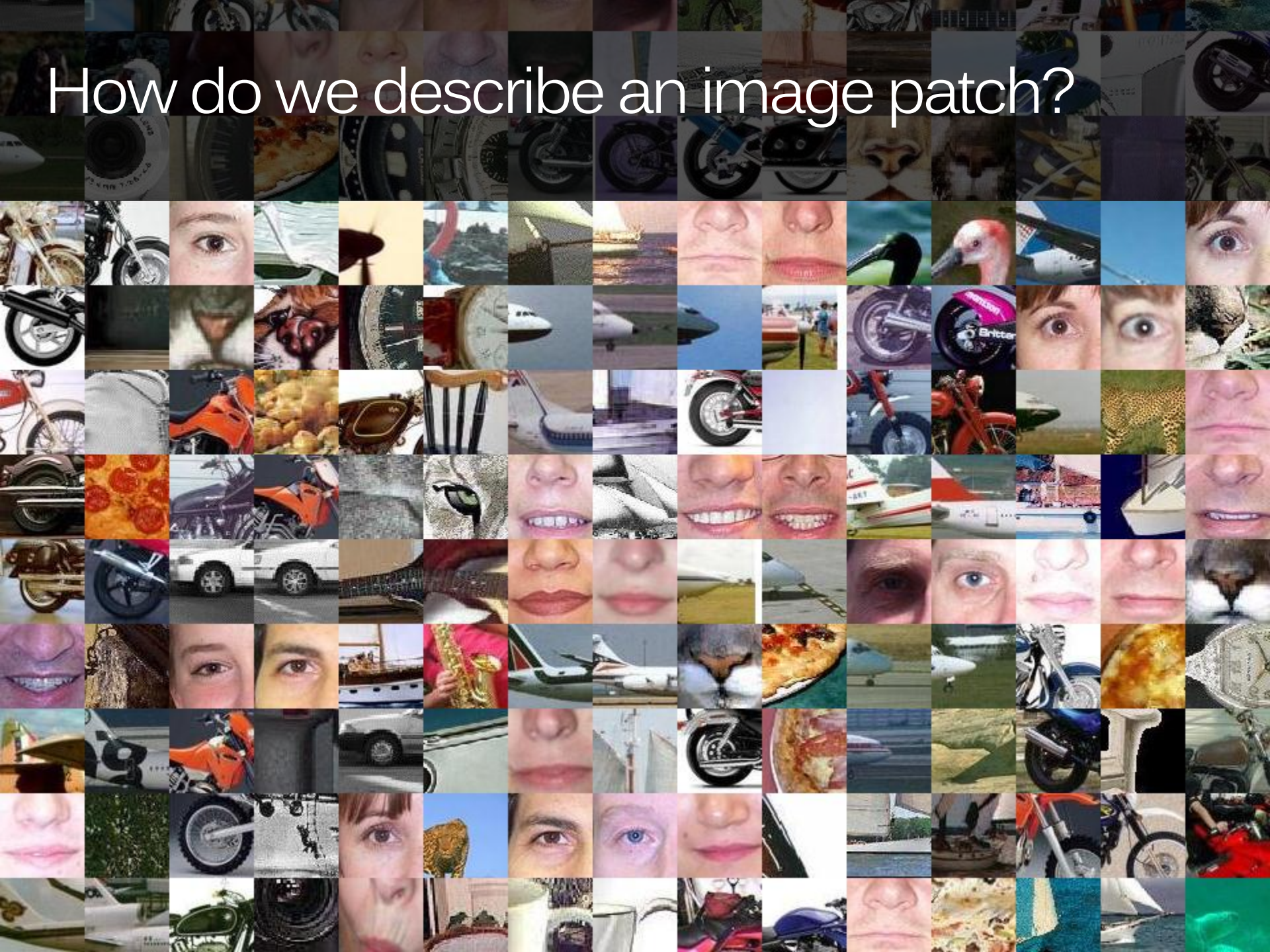
[Lowe, SIFT, 1999]

- Compute orientation histogram
- Select dominant orientation
- Normalize: rotate to fixed orientation





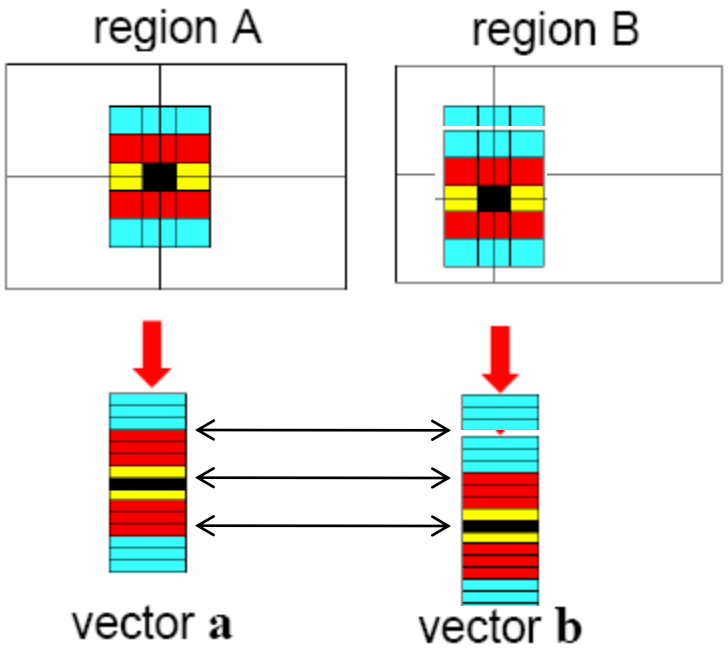
How do we describe an image patch?



How do we describe an image patch?

Patches with similar content should have similar descriptors.

Raw patches as local descriptors



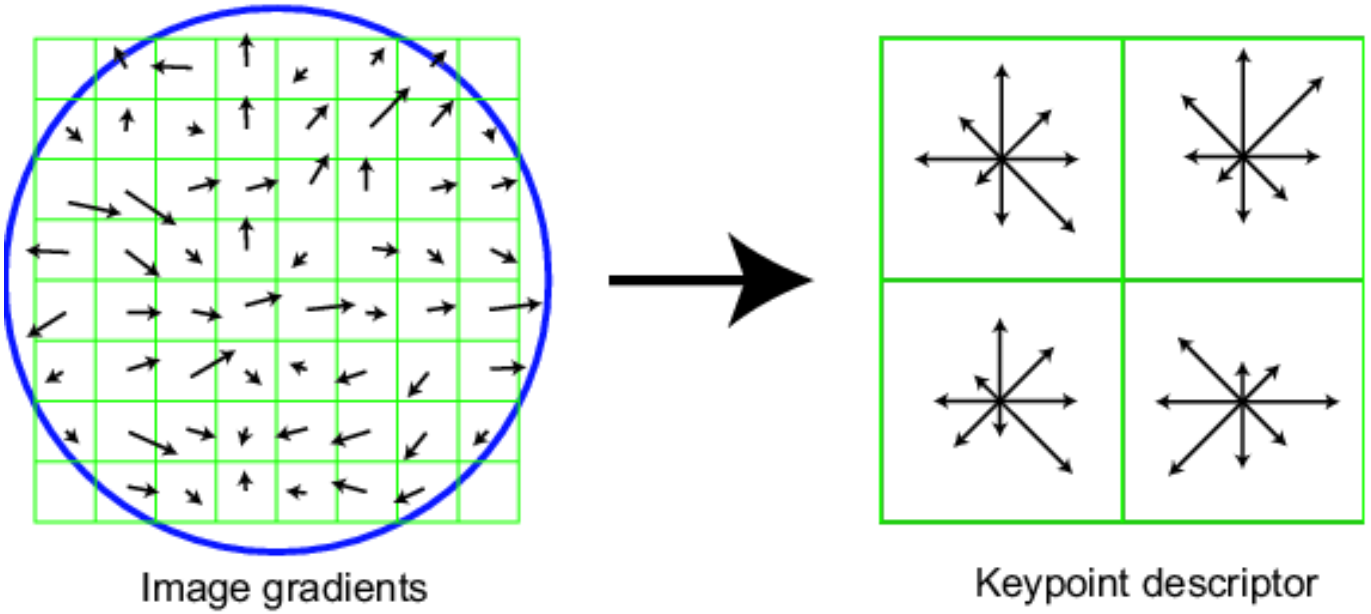
The simplest way to describe the neighborhood around an interest point is to write down the list of intensities to form a feature vector.

But this is very sensitive to even small shifts, rotations.

SIFT descriptor

Full version

- Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below)
- Compute an orientation histogram for each cell
- 16 cells * 8 orientations = 128 dimensional descriptor



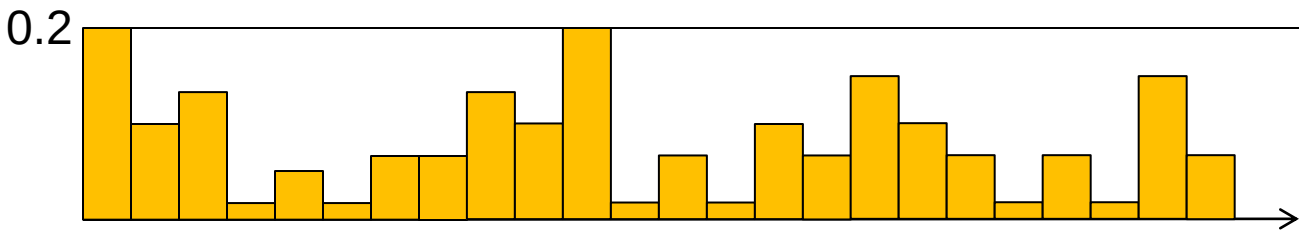
Adapted from slide by David Lowe

SIFT descriptor

Full version

- Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below)
- Compute an orientation histogram for each cell
- 16 cells * 8 orientations = 128 dimensional descriptor
- Threshold normalize the descriptor:

$$\sum_i d_i^2 = 1 \quad \text{such that:} \quad d_i < 0.2$$

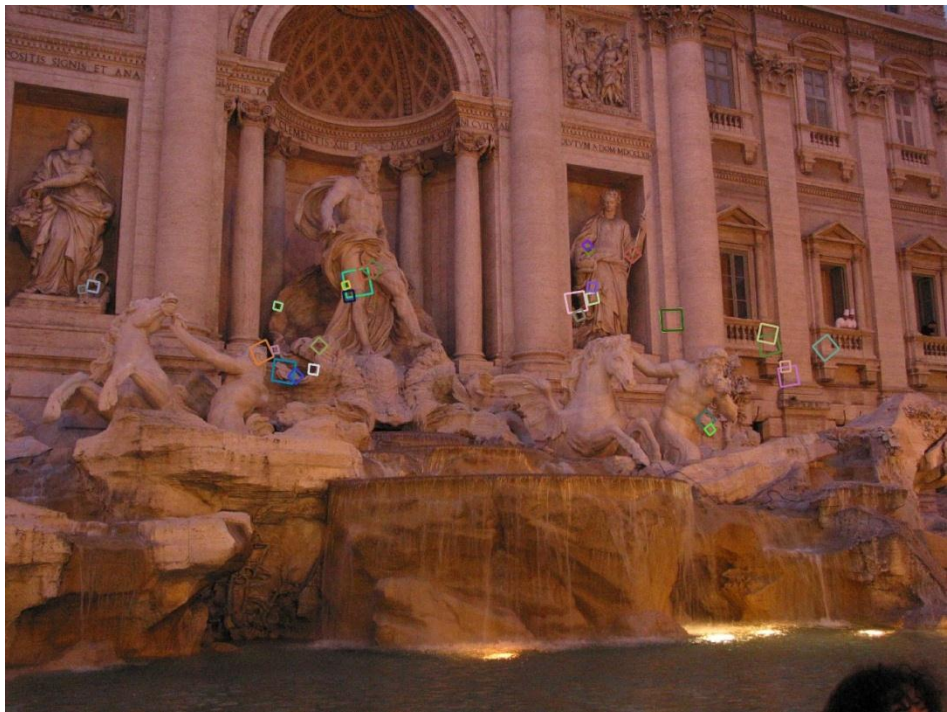


Adapted from slide by David Lowe

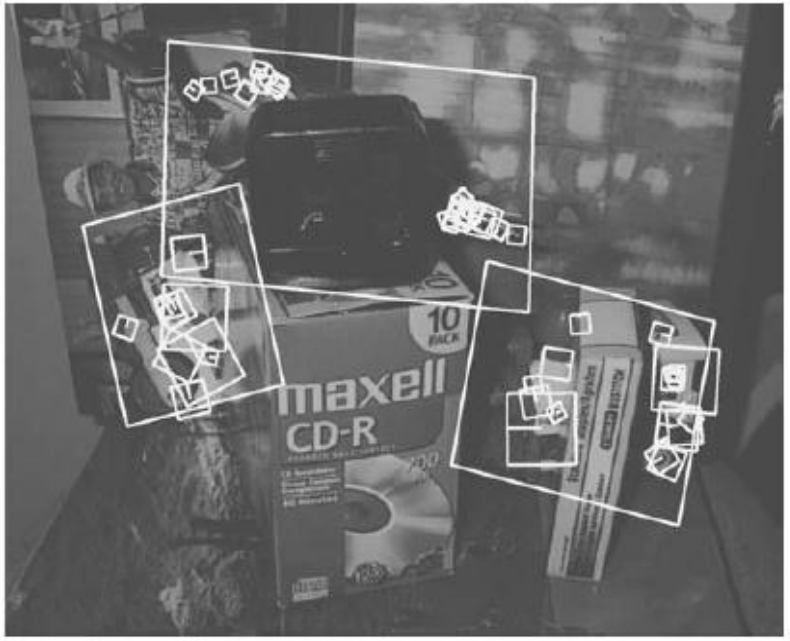
Properties of SIFT

Extraordinarily robust matching technique

- Can handle changes in viewpoint
  - Up to about 30 degree out of plane rotation
- Can handle significant changes in illumination
  - Sometimes even day vs. night (below)
- Fast and efficient—can run in real time
- Lots of code available
  - http://people.csail.mit.edu/albert/ladypack/wiki/index.php/Known_implementations_of_SIFT



Example: Object Recognition



SIFT is extremely powerful for object instance recognition, especially for well-textured objects

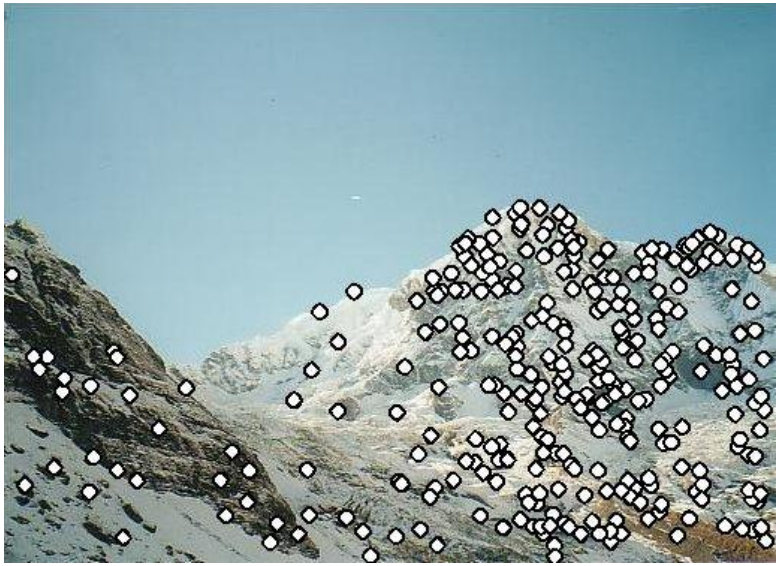
Lowe, IJCV04

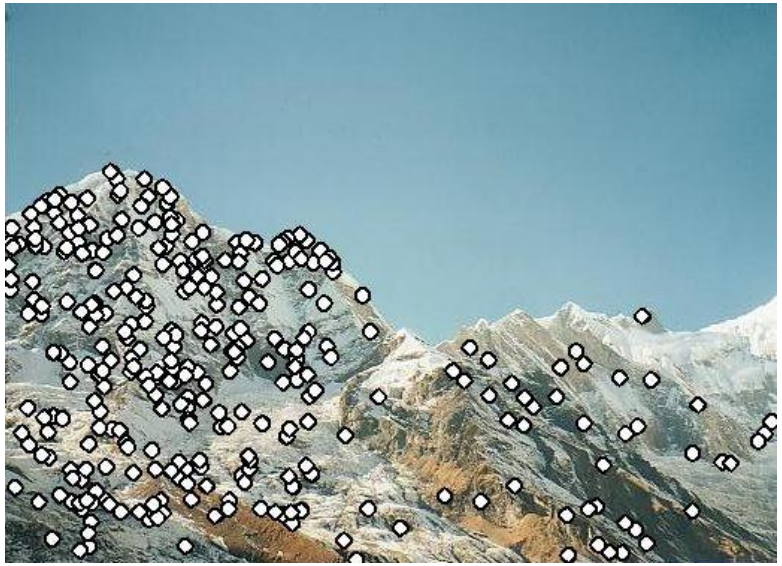
panorama?

- We need to match (align) images

Matching with Features

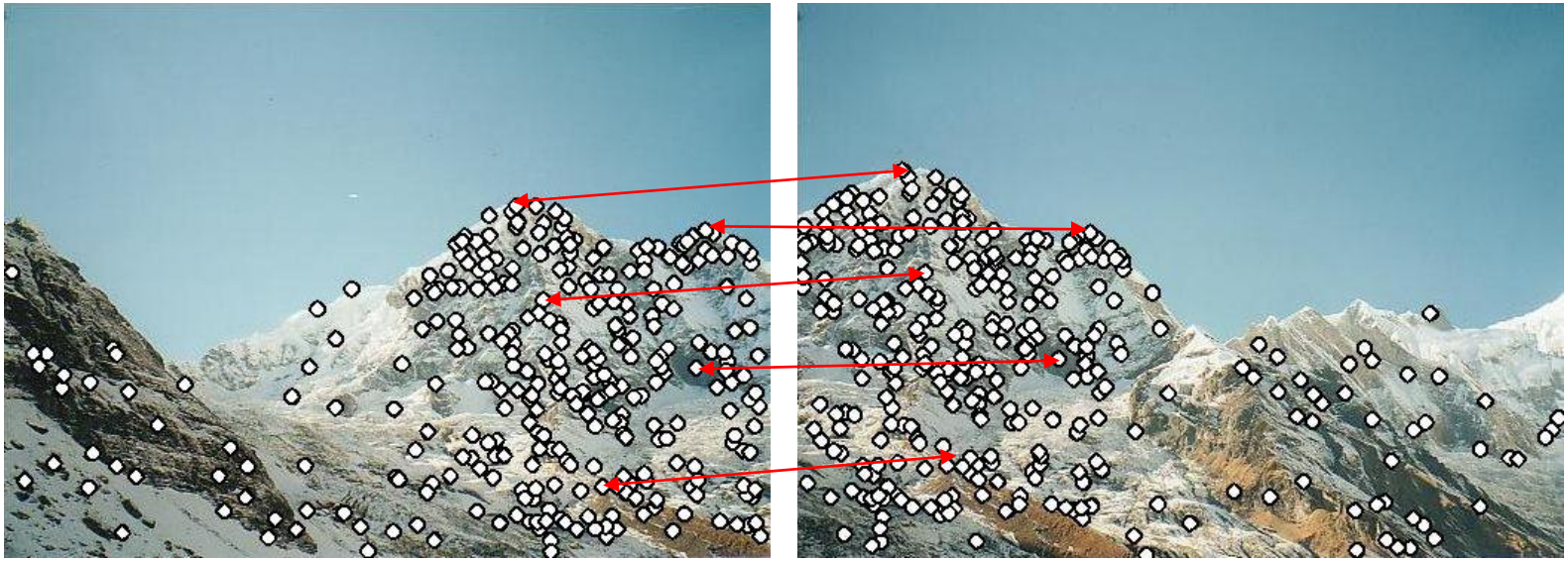
- Detect feature points in both images





Matching with Features

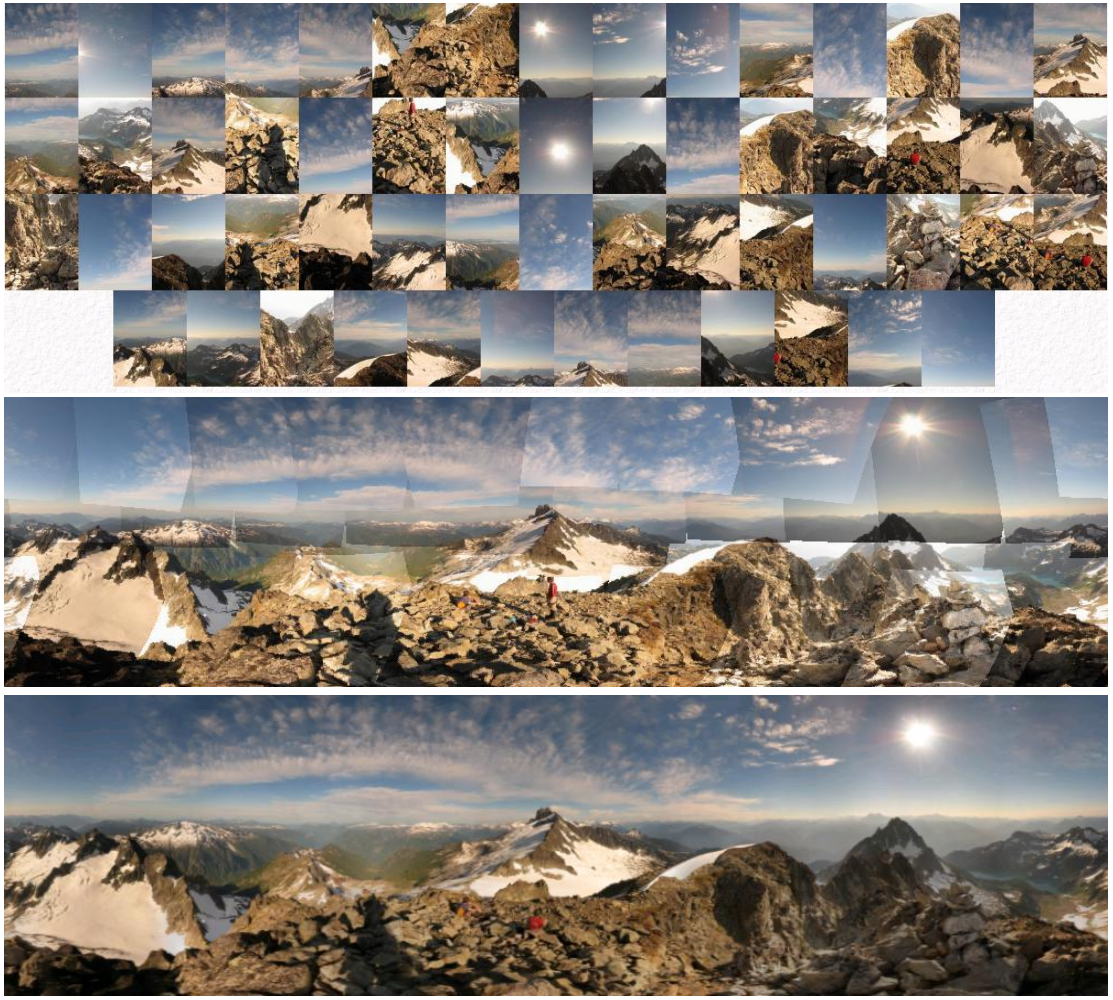
- Detect feature points in both images
- Find corresponding pairs



Matching with Features

- Detect feature points in both images
- Find corresponding pairs
- Use these matching pairs to align images - the required mapping is called a homography.

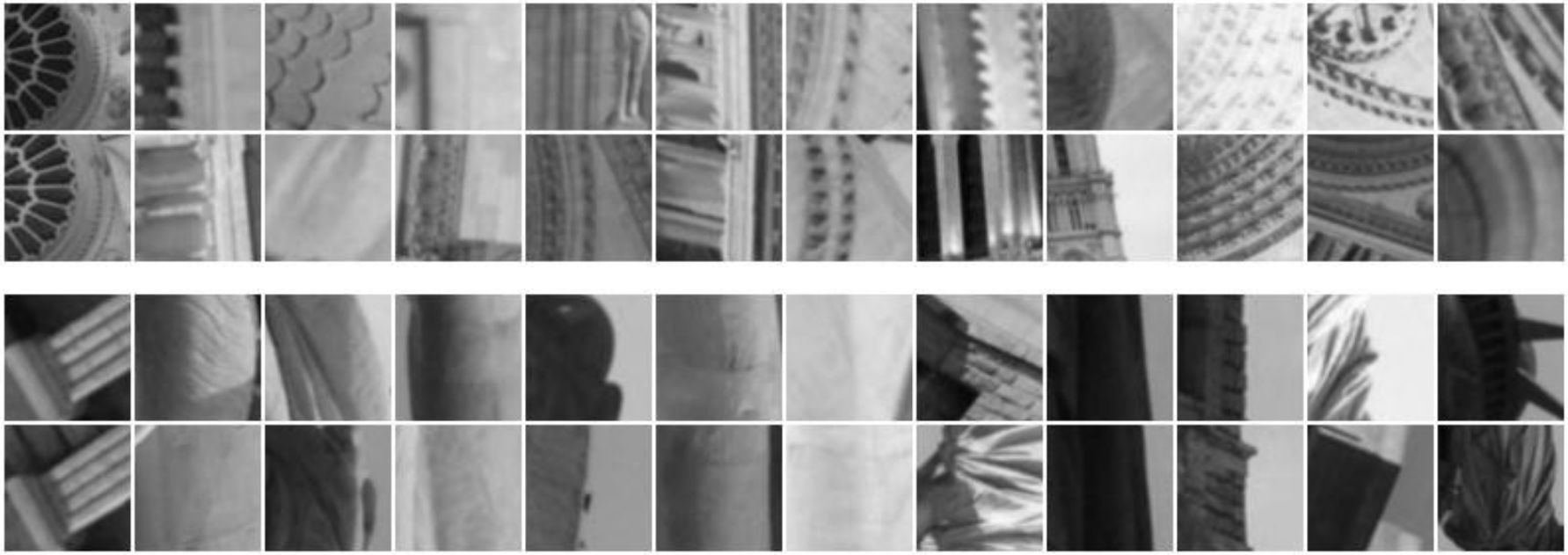
Automatic mosaicing



http://www.cs.ubc.ca/~mbrown/autostitch/autostitch.html

When does SIFT fail?

Patches SIFT thought were the same but aren't:



SURF: Speeded Up Robust Features

Background

- Local invariant Interest point detector-descriptor
  - For finding *correspondences* between two images of the same scene or object
  - Many applications, including 3D reconstruction, image retrieval and object recognition
  - SIFT is one of the best but slow
    - Image of size 1000 x 700 described in around **6** seconds (actual cost depends on the # features generated, 4000 in this case)
    - 128-D feature vectors

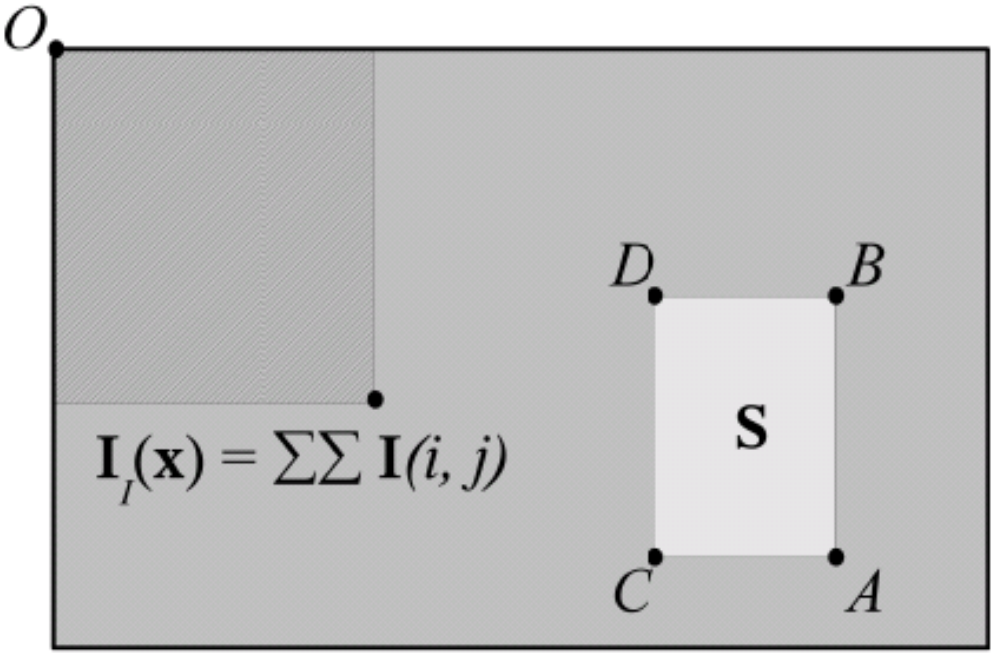
from "http://www.cse.cuhk.edu.hk/~lyu/seminar/06fall/Wyman.ppt"

Motivation

- Fast interest point **detection**
- Distinctive interest point **description**
- Speeded-up descriptor **matching**
- Invariant to common image transformations:
  - ✓ Image rotation
  - ✓ Scale changes
  - ✓ Illumination change
  - ✓ *Small* change in Viewpoint

Methodology

- Using integral images for major speed up
  - **Integral Image (summed area tables)** is an intermediate representation for the image and contains the **sum of gray scale pixel values of image**
  - Second order derivative and Haar-wavelet response



$$\mathbf{S}=A-B-C+D$$

Cost *four* additions operation only

Detection

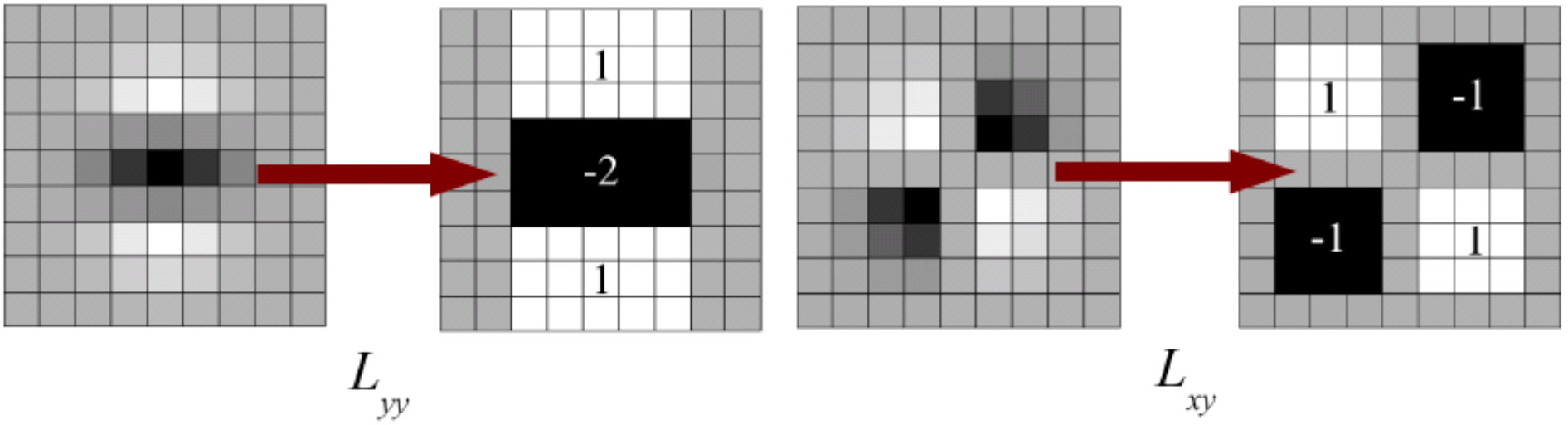
- Hessian-based interest point localization

$$H = \begin{bmatrix} L_{xx} & L_{xy} \\ L_{xy} & L_{yy} \end{bmatrix}$$

- $L_{xx}(x,y,\sigma)$ is the **Laplacian of Gaussian** of the image
- It is the convolution of the *Gaussian* second order derivative with the image
- Lindeberg showed Gaussian function is optimal for scale-space analysis
- This paper argues that Gaussian is overrated since the property that no new structures can appear while going to lower resolution is not proven in 2D case

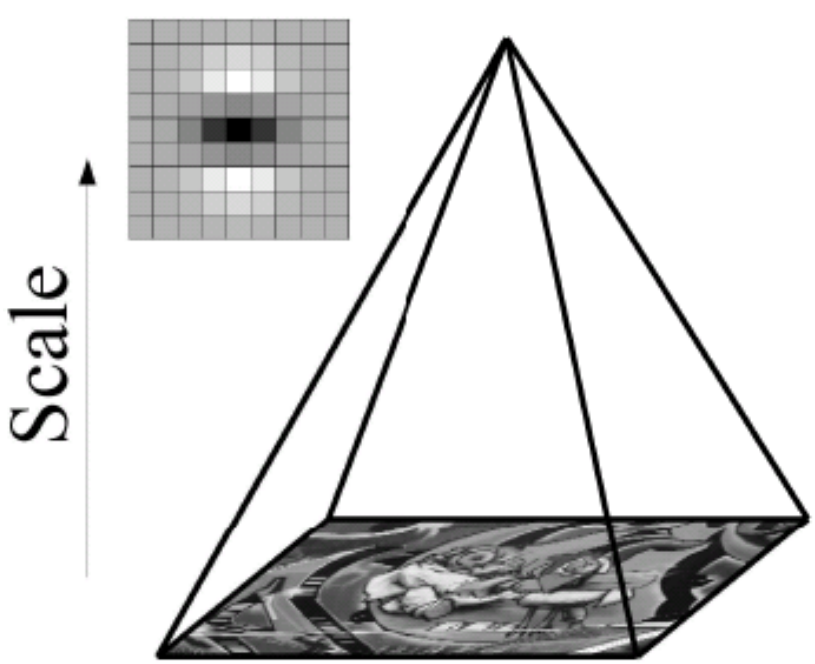
Detection

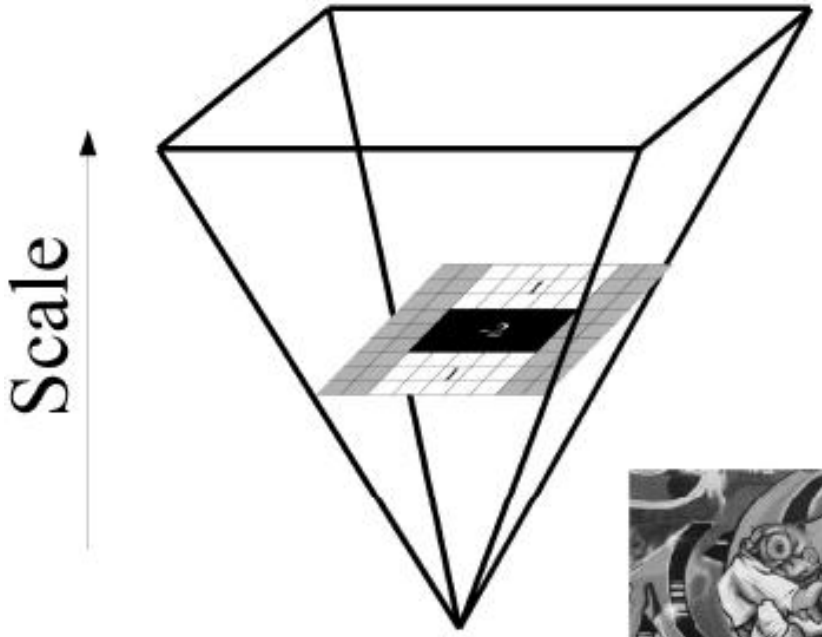
- Approximated second order derivatives with box filters (mean/average filter)



Detection

- Scale analysis with constant image size



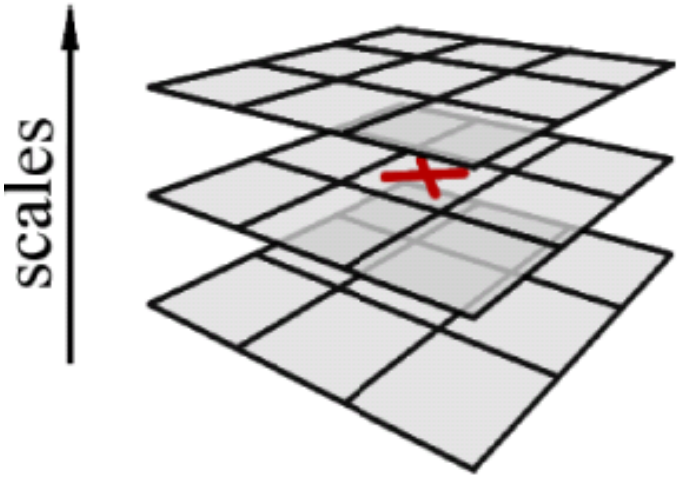


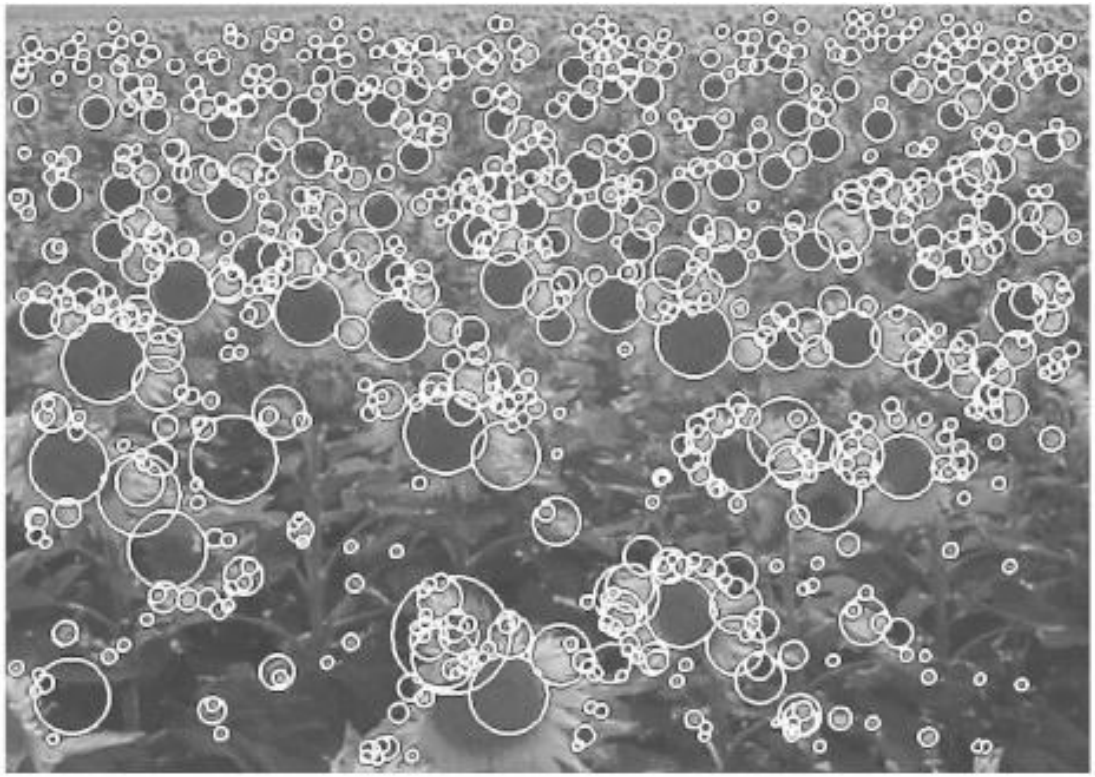
9 x 9, 15 x 15, 21 x 21, 27 x 27 → 39 x 39, 51 x 51 …

1st octave *2nd octave*

Detection

- Non-maximum suppression and interpolation
  - Blob-like feature detector

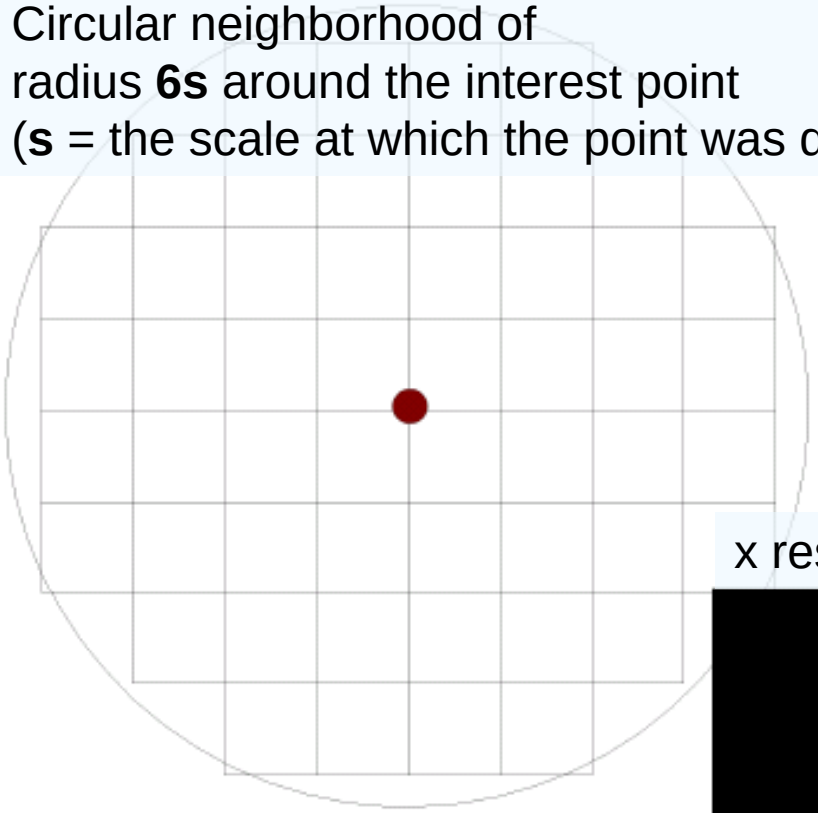


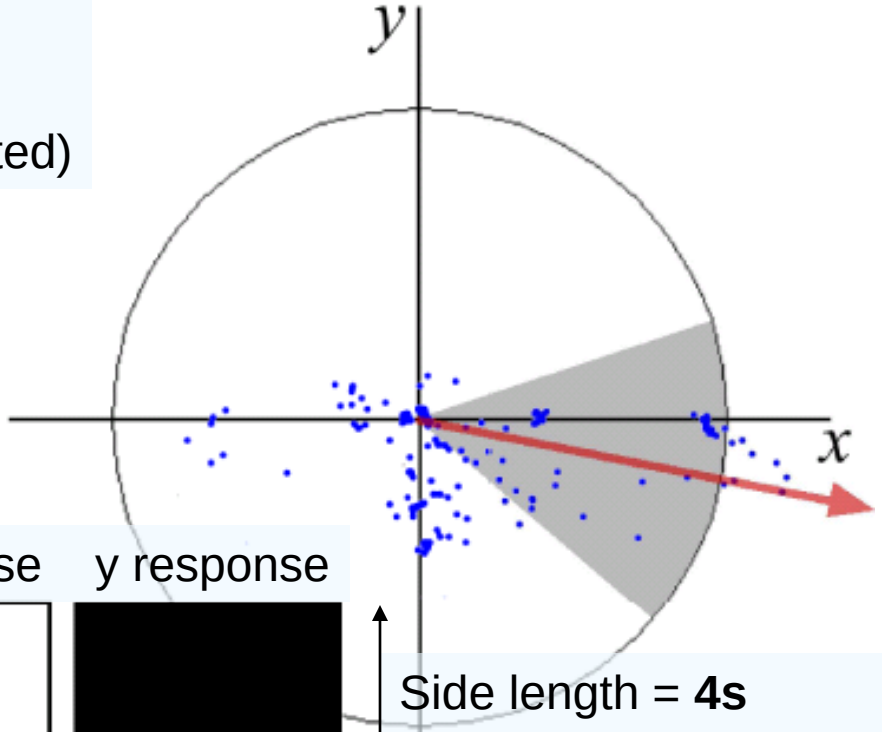


Description

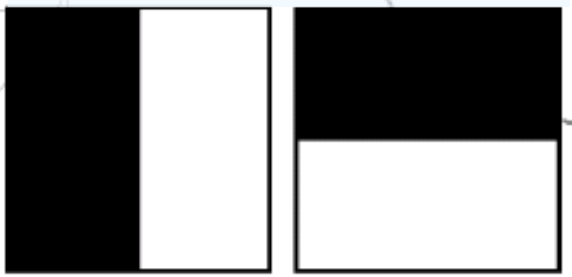
- Orientation Assignment

Circular neighborhood of
radius **6s** around the interest point
(**s** = the scale at which the point was detected)





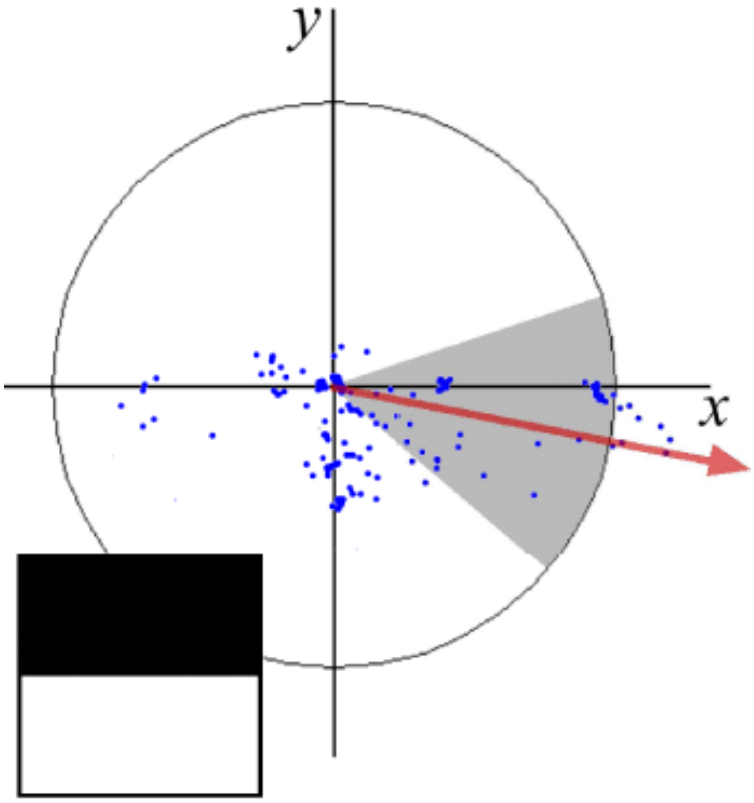
x response y response



Side length = **4s**
Cost 6 operation to
compute the response

Description

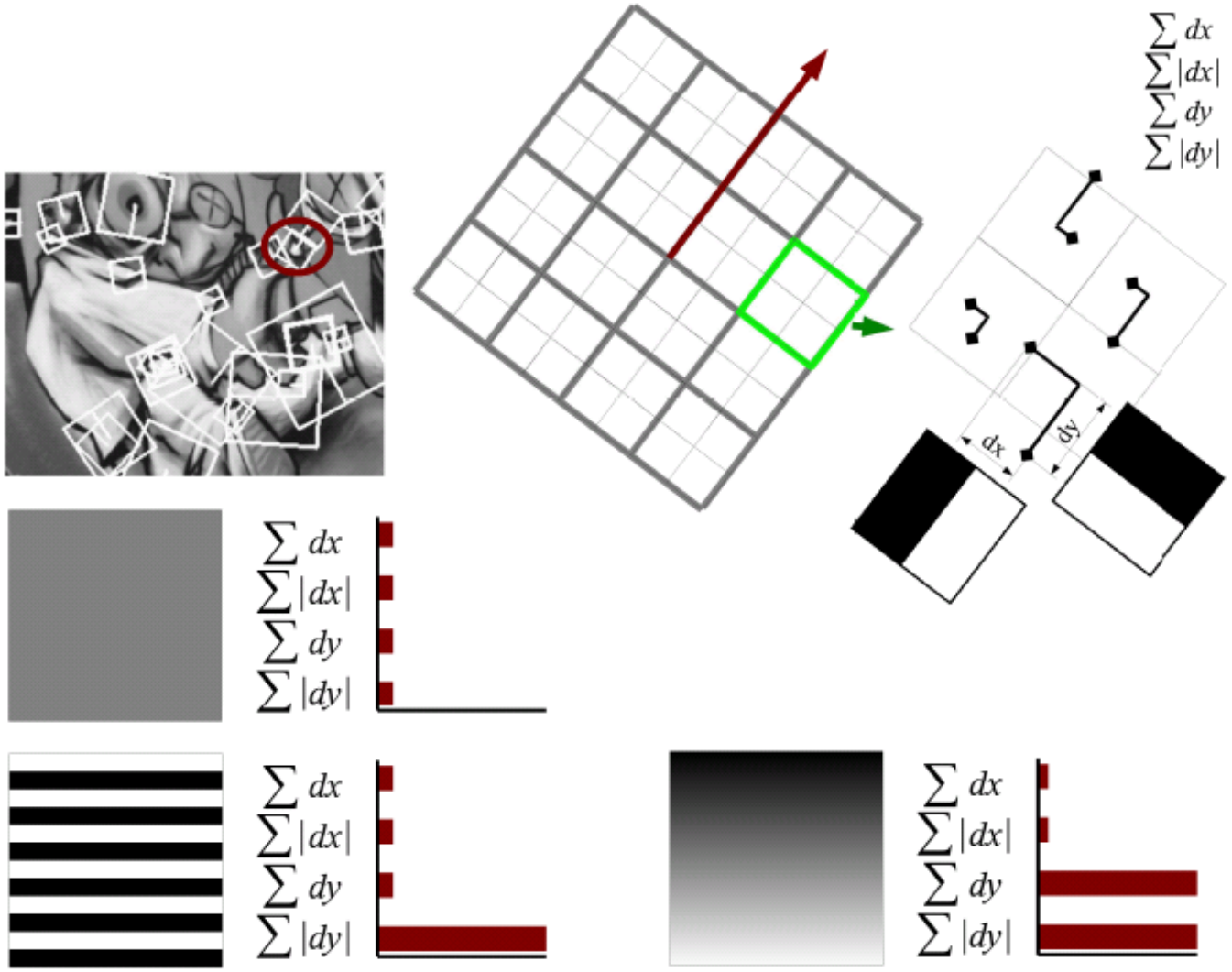
- Dominant orientation
  - The Haar wavelet responses are represented as vectors
  - Sum all responses within a sliding orientation window covering an angle of 60 degree
  - The two summed response yield a new vector
  - **The longest vector** is the dominant orientation
  - Second longest is … **ignored**



Description

- Split the interest region up into 4 x 4 square sub-regions with 5 x 5 regularly spaced sample points inside
- Calculate Haar wavelet response d_x and d_y
- Weight the response with a Gaussian kernel centered at the interest point
- Sum the response over each sub-region for d_x and d_y separately → **feature vector of length 32**
- In order to bring in information about the polarity of the intensity changes, extract the sum of absolute value of the responses → **feature vector of length 64**
- Normalize the vector into unit length

Description

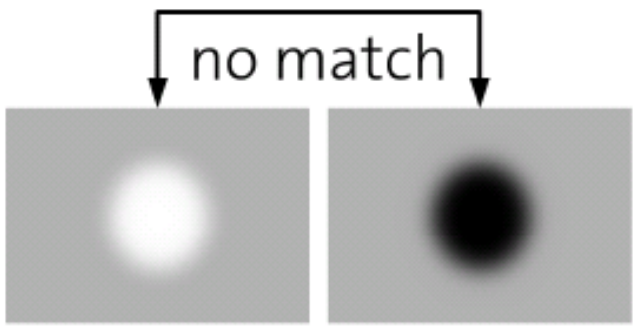


Description

- SURF-128
  - The sum of d_x and $|d_x|$ are computed separately for $d_y < 0$ and $d_y > 0$
  - Similarly for the sum of d_y and $|d_y|$
  - This doubles the length of a feature vector

Matching

- Fast indexing through the sign of the Laplacian for the underlying interest point
  - The sign of trace of the Hessian matrix
  - Trace = $L_{xx} + L_{yy}$



- Either 0 or 1 (Hard thresholding, may have boundary effect …)
- In the matching stage, compare features if they have the same type of contrast (sign)

Analysis

- SURF is good at
  - handling serious blurring
  - handling image rotation
- SURF is poor at
  - handling viewpoint change
  - handling illumination change
- SURF is always better than the SIFT implemented by VGG but not the original SIFT

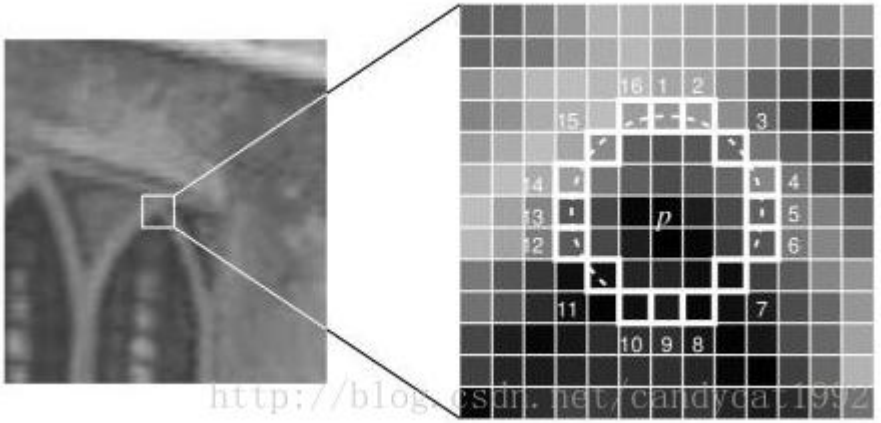
img#	Bikes	Boat	graf	leuven	wall
2	o	++	--	-	---
3	o	o	-	--	----
4	+	+++	-	--	---
5	++	+++	o	--	o
6	+++	+++	o	---	o

Conclusion

- SURF describes image faster than SIFT by 3 times
- SURF is not as well as SIFT on invariance to illumination change and viewpoint change

Other methods: Fast corner Detection

1. 从图片中选取一个像素点p，下面我们将判断它是否是一个特征点。我们首先把它的密度（即像素值）设为 Ip。
2. 设定一个合适的阈值t 。
3. 考虑该像素点周围的16个像素。
4. 如果在这个大小为16个像素的圆上有n个连续的像素点，它们的像素值要么都比Ip+t大，要么都比Ip-t小，那么它就是一个角点。（如上图中白色虚线所示）。n这里被设定为12。

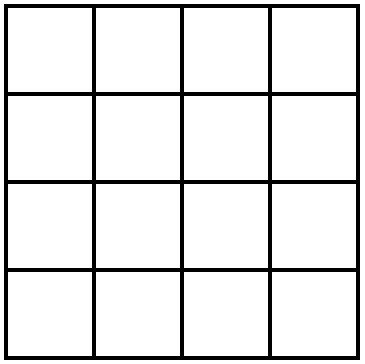


5. 检查在位置1、9、5和13四个位置的像素（首先检查1和9，看它们是否亮于或暗于阈值。如果是，再检查5和13）。如果是一个角点，那么上述四个像素点中至少有3个应该必须都大于或者小于 （因为若是一个角点，超过四分之三个圆的部分应该满足判断条件，半圆比包含上述某三个点）。如果都不满足，那么不可能是一个角点。完整的分段测试可以被用于接受的所有候选点，通过检测圆上的所有点。

总结：FAST算法比其他已知的角点检测法要快很多倍。但是当图片的噪点较多时，它的健壮性并不好。这依靠一个阈值。

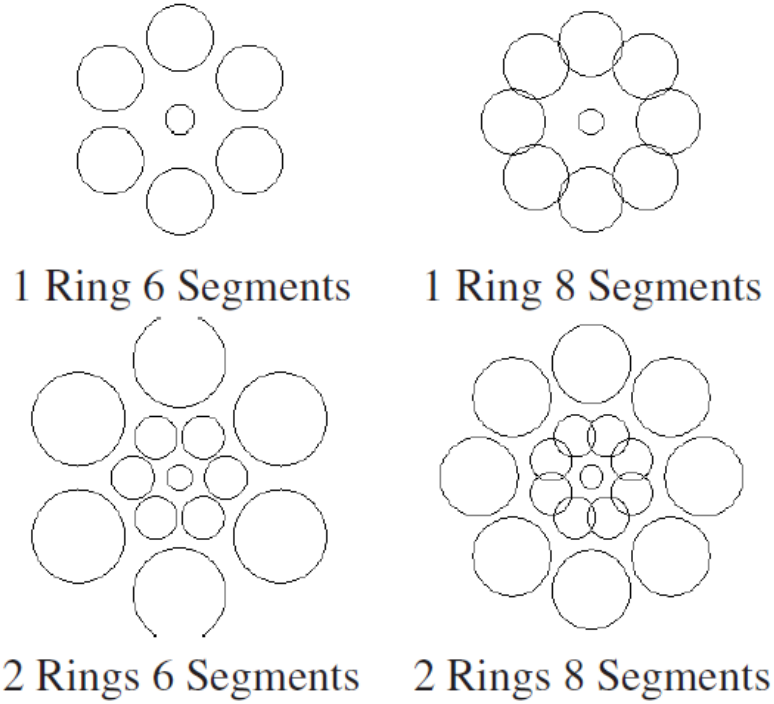
Machine learning for high-speed corner detection, ECCV06

Other methods: Daisy



SIFT

Circular gradient binning



Daisy

Picking the best DAISY, S. Winder, G. Hua, M. Brown, CVPR 09

Other methods: HOG

- Hog：Histogram of Oriented Gradient,通过计算和统计图像局部区域的梯度方向直方图来构成特征。
- **本质**：梯度的统计信息，而梯度主要存在于边缘的地方。
- **实现方法**：首先将图像分成小的连通区域，我们把它叫细胞单元。然后采集细胞单元中各像素点的梯度的或边缘的方向直方图。最后把这些直方图组合起来就可以构成特征描述器。
- **优点**：首先，由于HOG是在图像的局部方格单元上操作，所以它对图像几何的和光学的形变都能保持很好的不变性，这两种形变只会出现在更大的空间领域上。其次，在粗的空域抽样、精细的方向抽样以及较强的局部光学归一化等条件下，只要行人大体上能够保持直立的姿势，可以容许行人有一些细微的肢体动作，这些细微的动作可以被忽略而不影响检测效果。因此HOG特征是特别适合于做图像中的人体检测的。
- **应用**：Hog特征结合SVM分类器已经被广泛应用于图像识别中，尤其在行人检测中获得了极大的成功。

Histogram of Oriented Gradient for Human Detection, CVPR 2005

Other methods: BRIEF

Randomly sample pair of pixels a and b.
1 if a > b, else 0. Store binary vector.

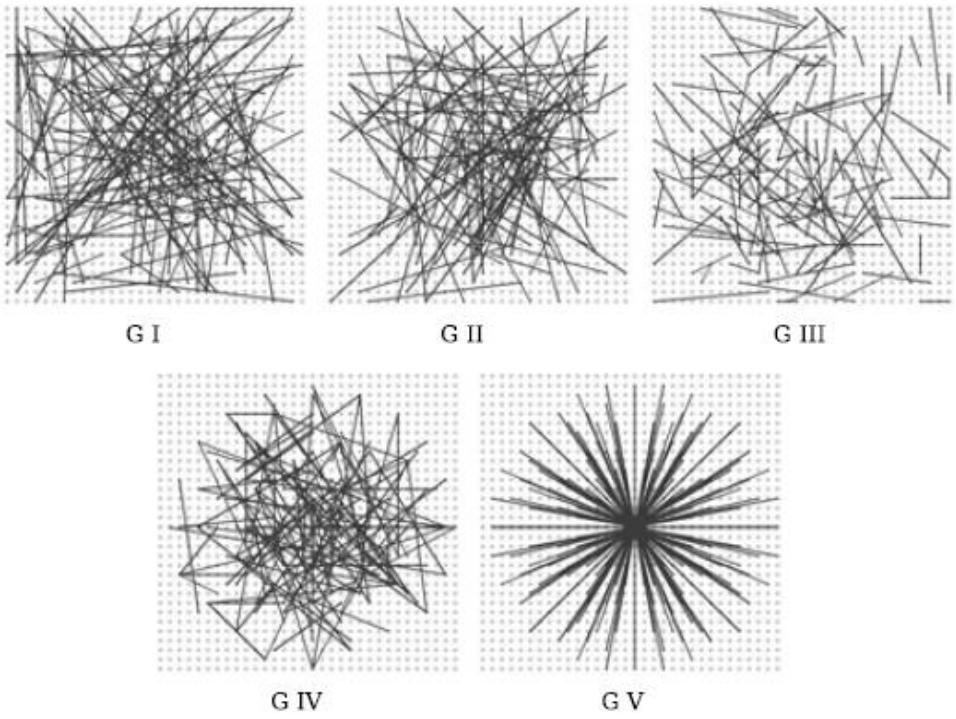


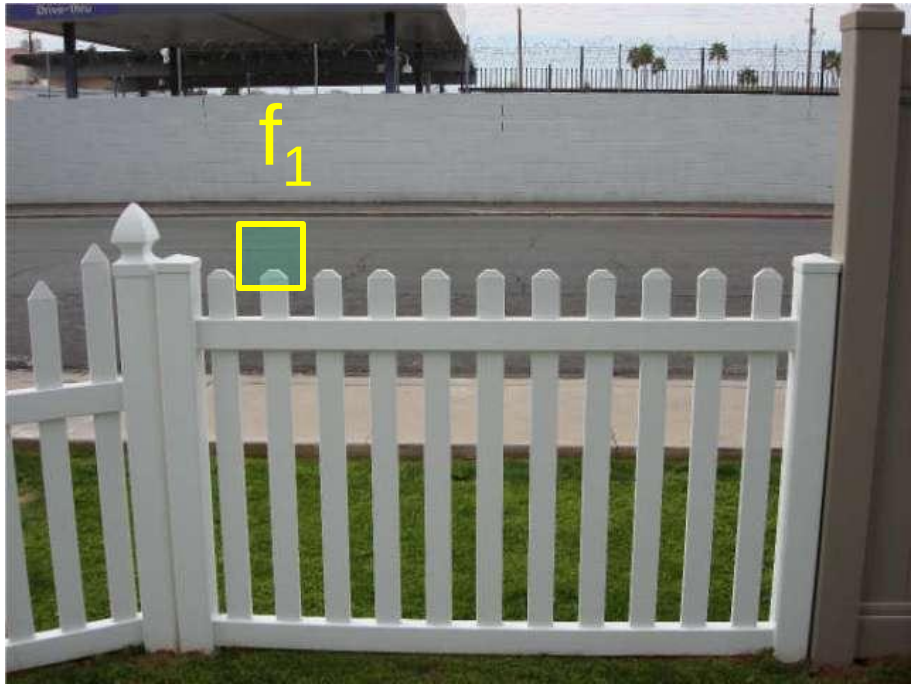
Fig. 2. Different approaches to choosing the test locations. All except the righmost one are selected by random sampling. Showing 128 tests in every image.

BRIEF: binary robust independent elementary features, Calonder, V Lepetit, C Strecha, ECCV 2010

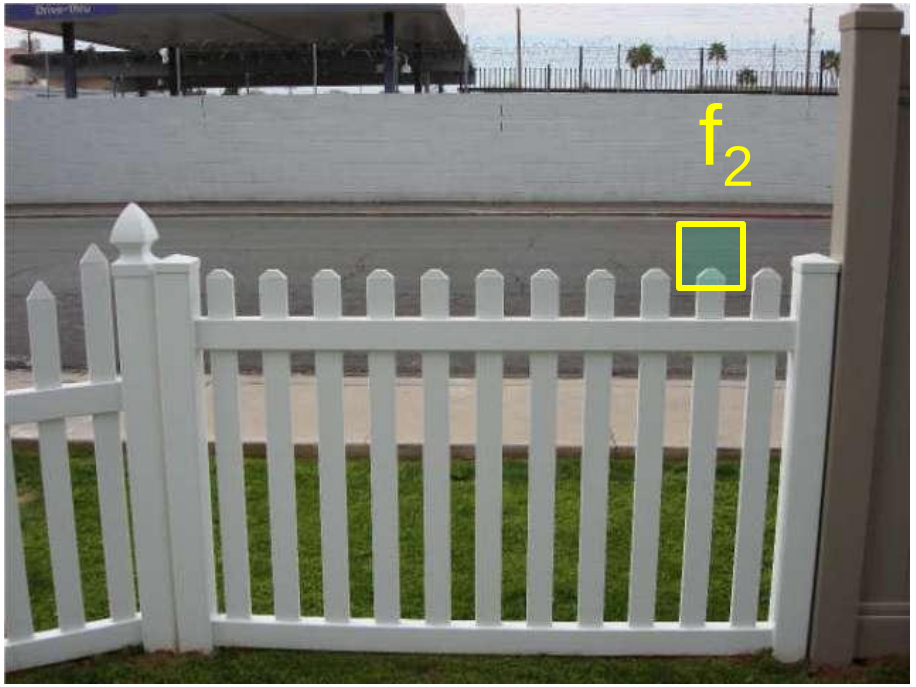
Feature distance

How to define the difference between two features f_1, f_2?

- Simple approach is SSD(f_1, f_2)
  - sum of square differences between entries of the two descriptors
  - can give good scores to very ambiguous (bad) matches



I_1

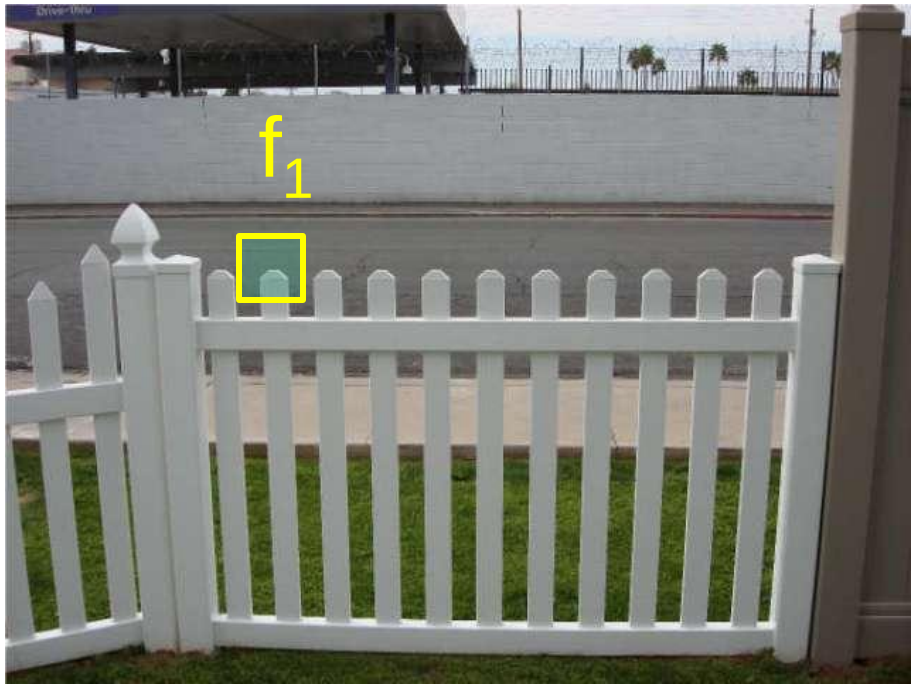


I_2

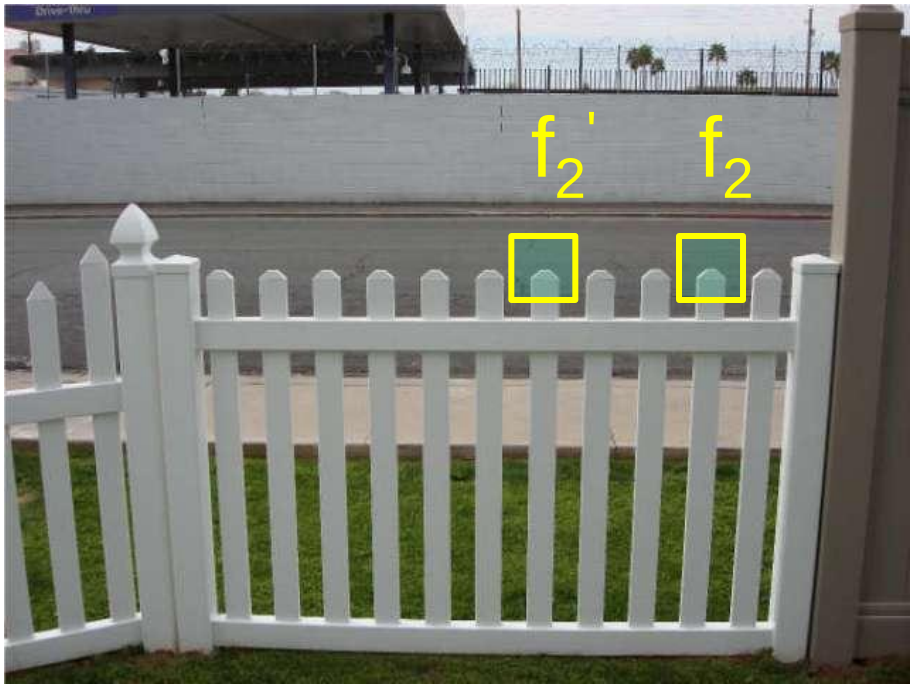
Feature distance

How to define the difference between two features f_1, f_2?

- Better approach: ratio distance = SSD(f_1, f_2) / SSD(f_1, f_2')
  - f_2 is best SSD match to f_1 in I_2
  - f_2' is 2nd best SSD match to f_1 in I_2
  - gives large values (~1) for ambiguous matches

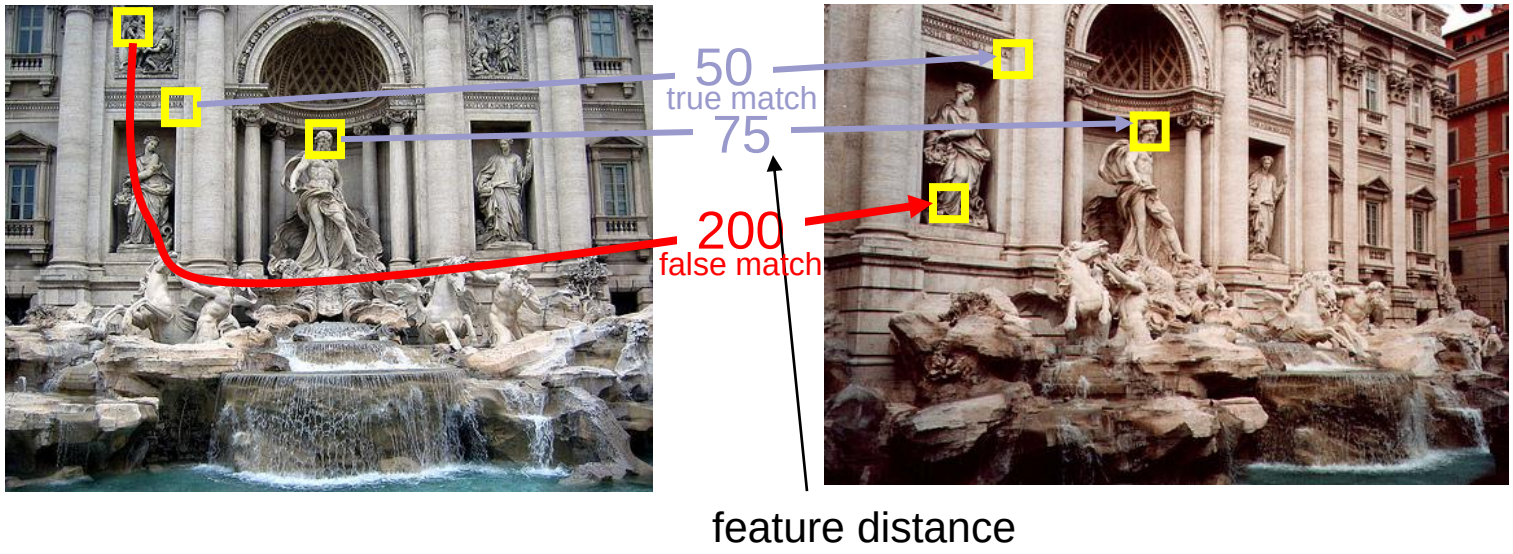


I_1



I_2

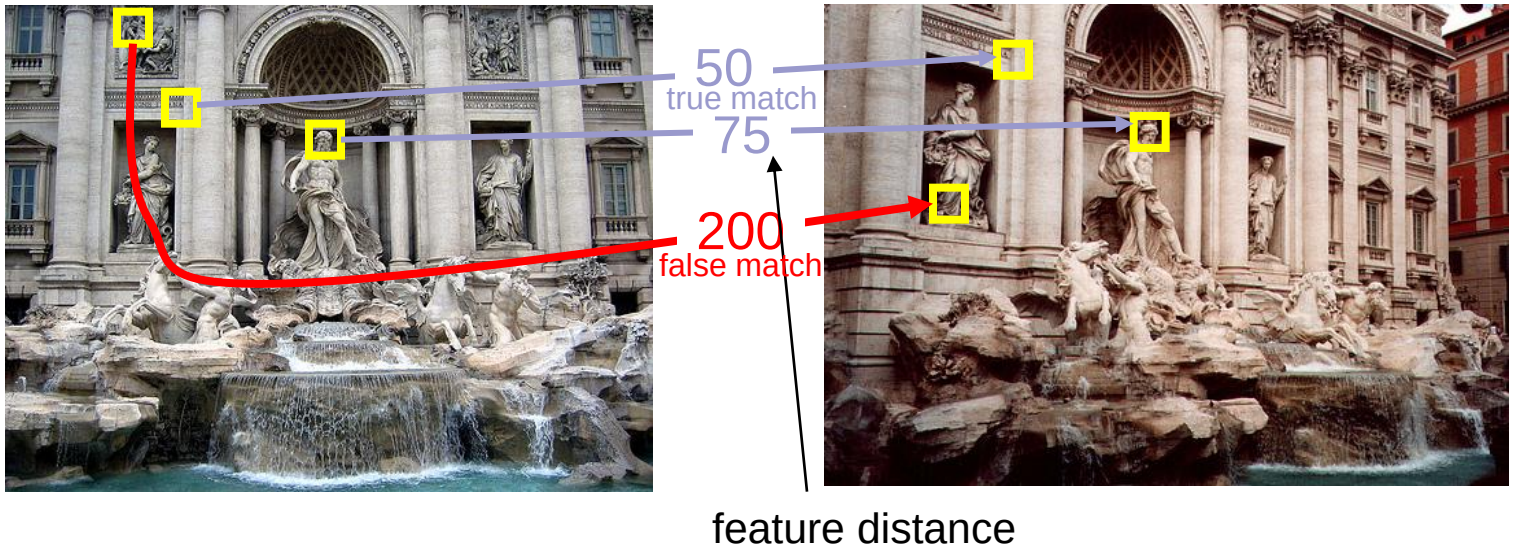
Eliminating bad matches



Throw out features with distance > threshold

- How to choose the threshold?

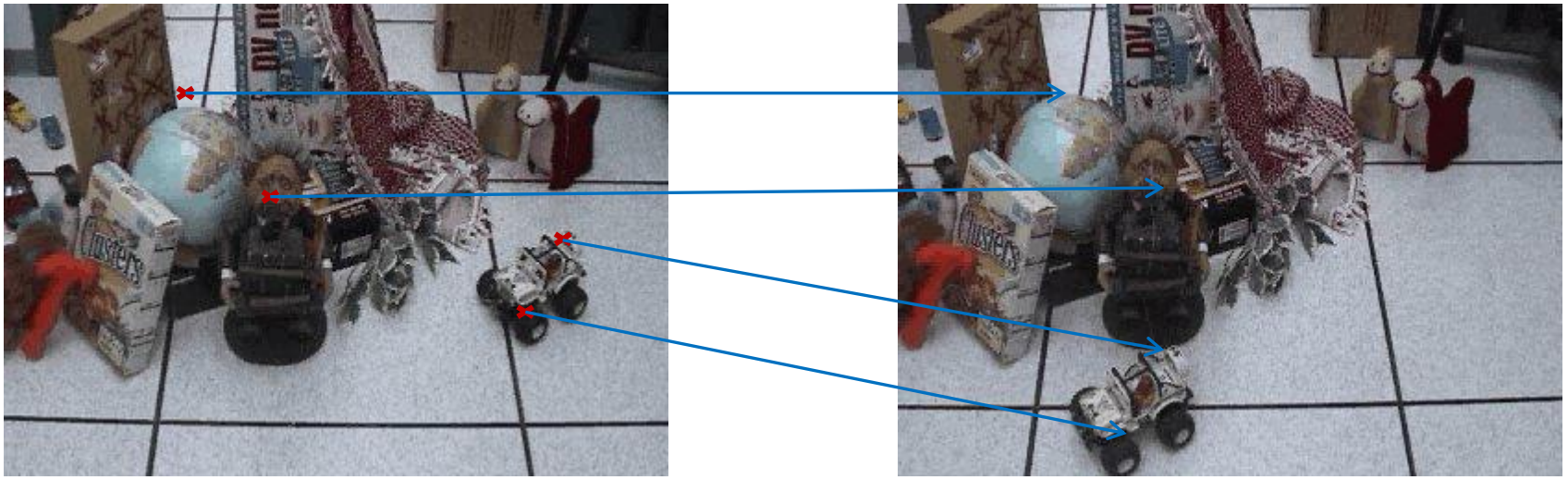
True/false positives



The distance threshold affects performance

- True positives = # of detected matches that are correct
  - Suppose we want to maximize these—how to choose threshold?
- False positives = # of detected matches that are incorrect
  - Suppose we want to minimize these—how to choose threshold?

What is Optical Flow ?



Where did **each pixel** in image 1 go to in image 2:

Dense!

Goal:

Find for **each pixel** a velocity vector $\vec{u} = (u, v)$ which says:

-How quickly(**magnitude**) is the pixel moving across the image

-In which **direction** it is moving

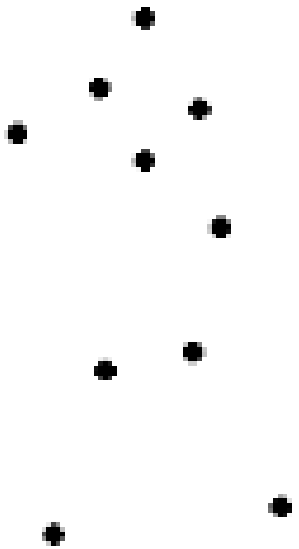
We live in a moving world

- Perceiving, understanding and predicting motion is an important part of our daily lives

from “http://szeliski.org/Book/”

Motion and perceptual organization

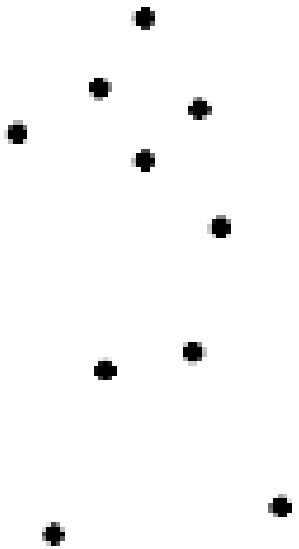
- Even “impoverished” motion data can evoke a strong percept



G. Johansson, “Visual Perception of Biological Motion and a Model For Its Analysis", *Perception and Psychophysics 14, 201-211, 1973.*

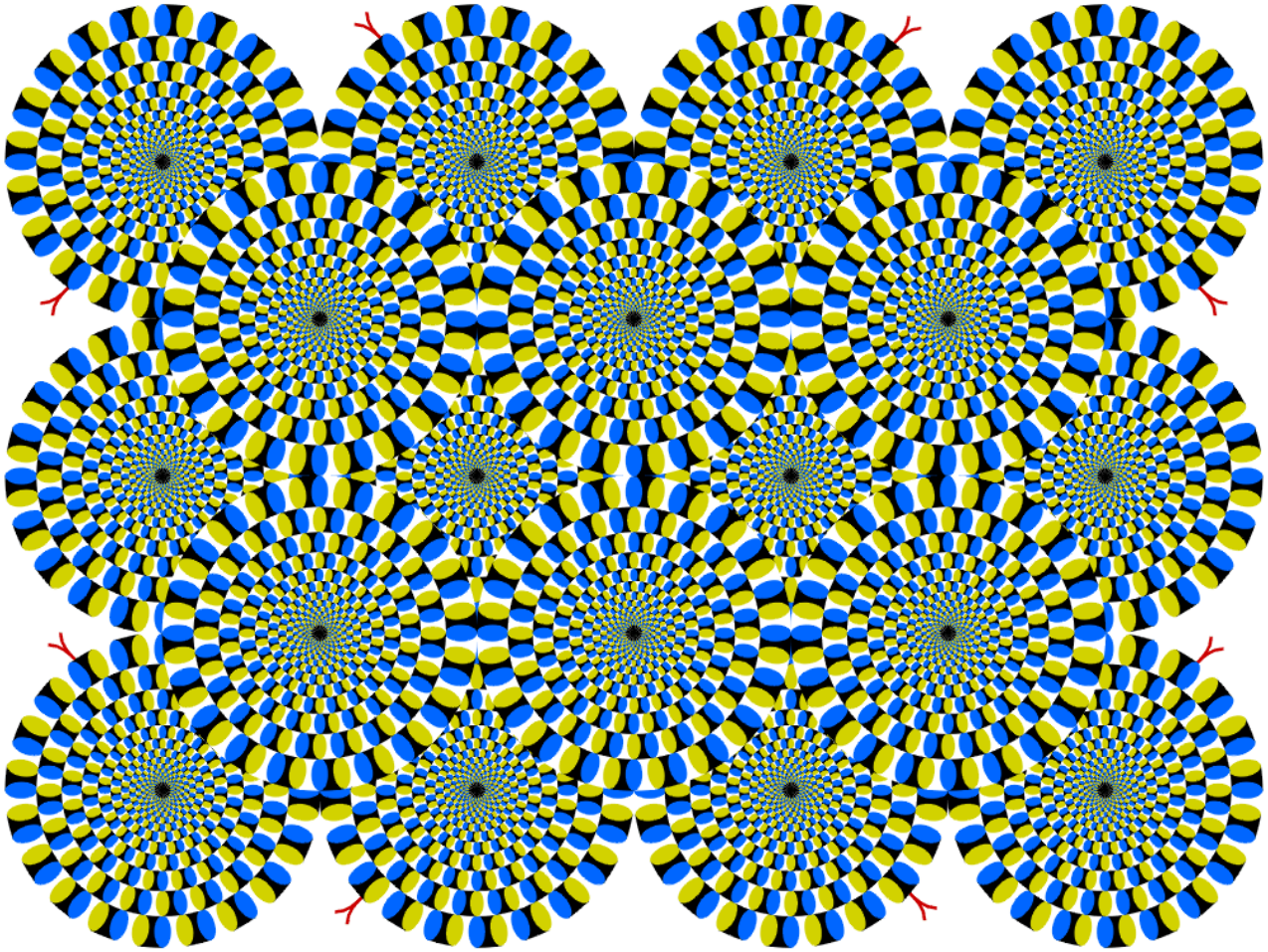
Motion and perceptual organization

- Even “impoverished” motion data can evoke a strong percept



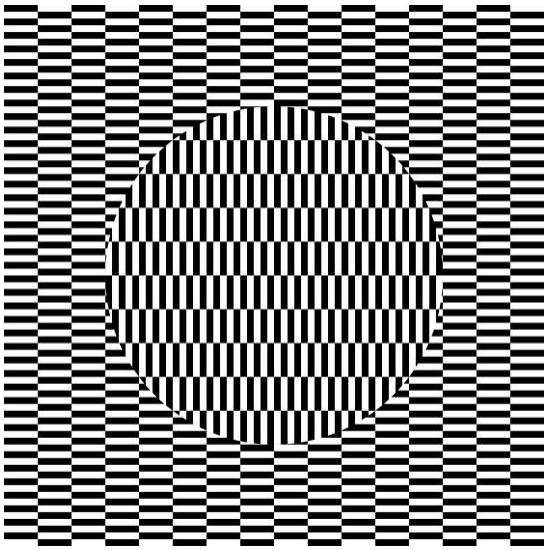
G. Johansson, “Visual Perception of Biological Motion and a Model For Its Analysis", *Perception and Psychophysics 14, 201-211, 1973.*

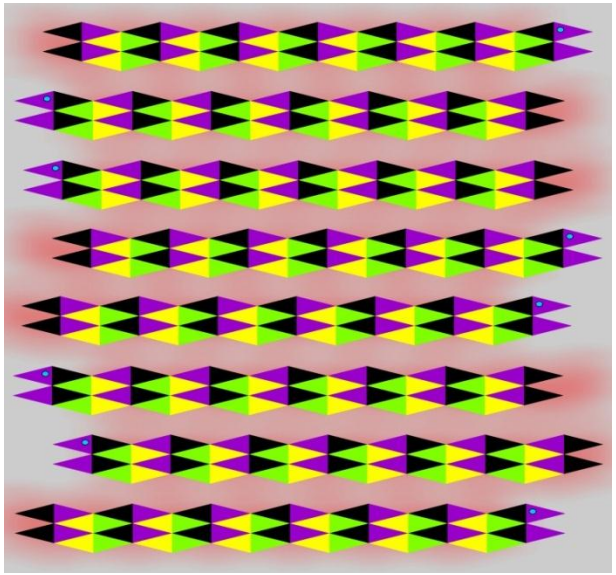
Seeing motion from a static picture?

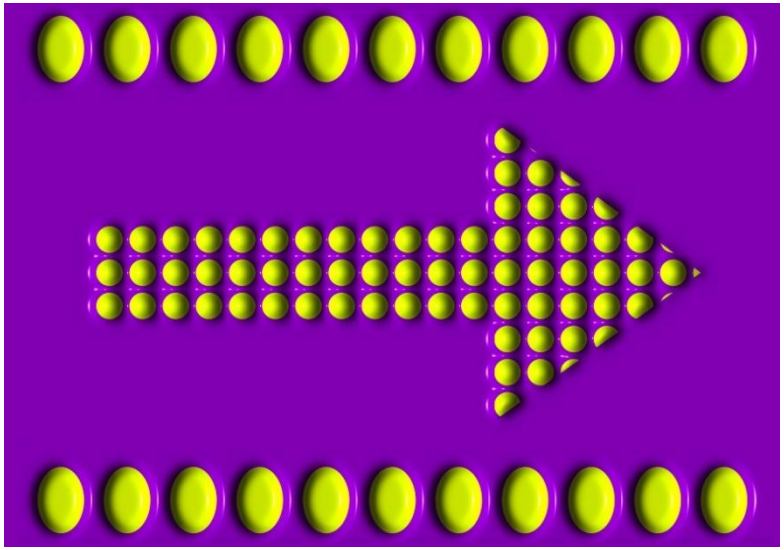


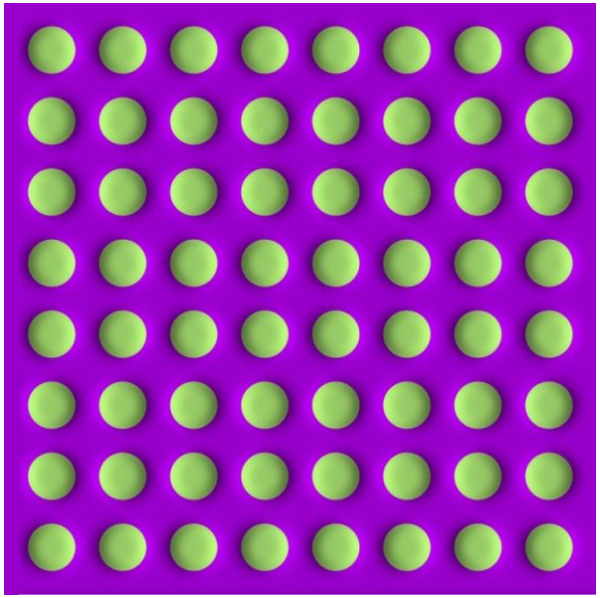
http://www.ritsumei.ac.jp/~akitaoka/index-e.html

More examples



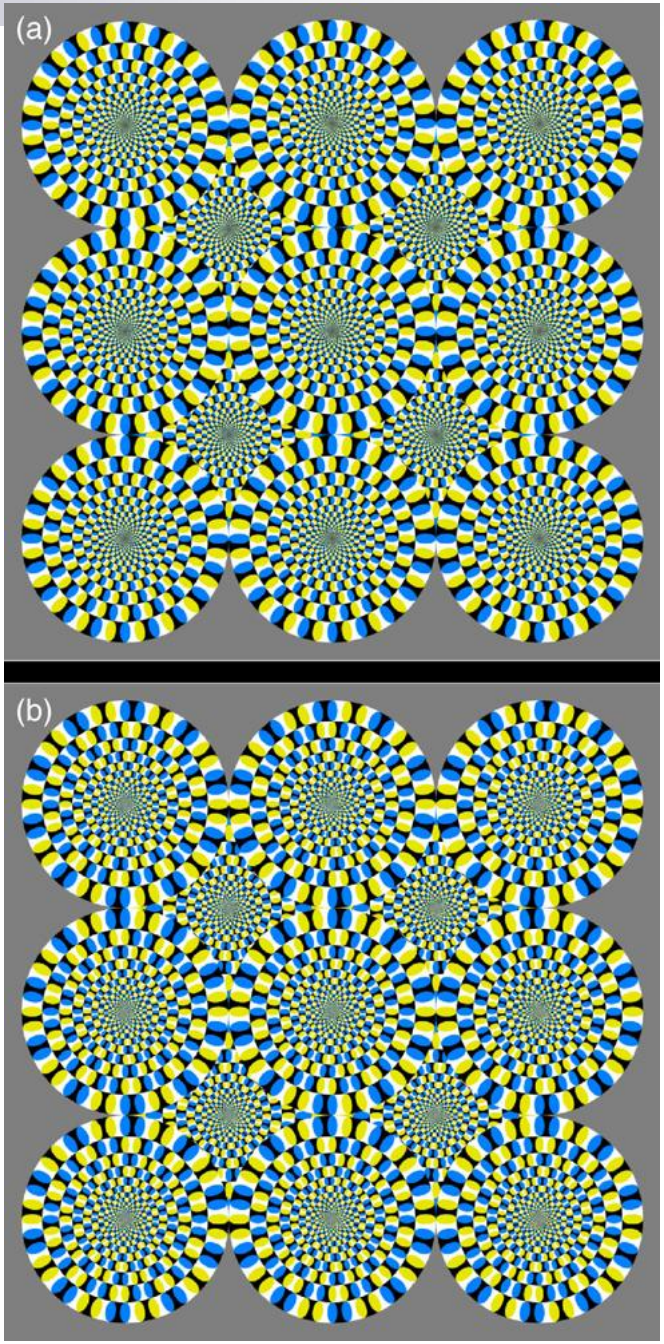






How is this possible?

- The true mechanism is to be revealed
- FMRI data suggest that illusion is related to some component of eye movements
- We don’t expect computer vision to “see” motion from these stimuli, yet



The cause of motion

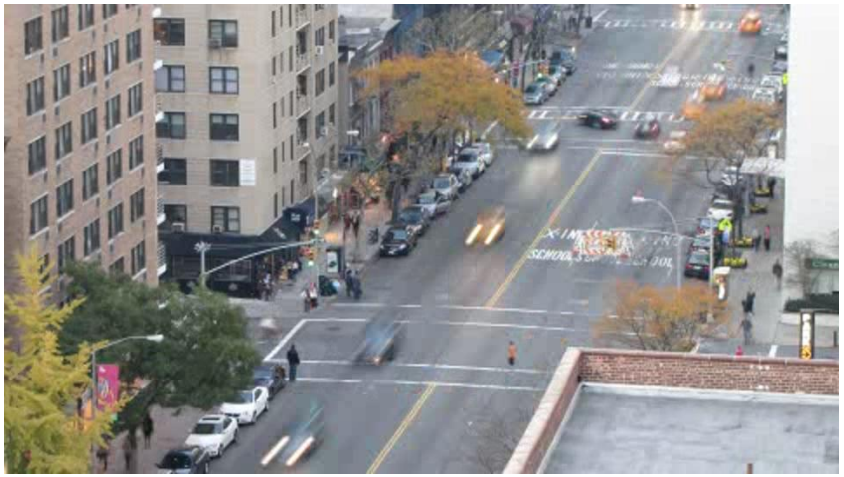
- Three factors in imaging process
  - Light
  - Object
  - Camera
- Varying either of them causes motion
  - Static camera, moving objects (surveillance)
  - Moving camera, static scene (3D capture)
  - Moving camera, moving scene (sports, movie)
  - Static camera, moving objects, moving light (time lapse)

Motion scenarios (priors)

Static camera, moving scene

Moving camera, static scene

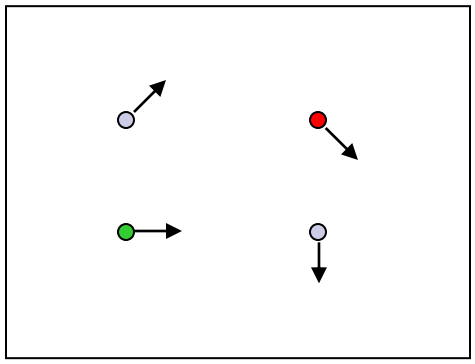
Moving camera, moving scene



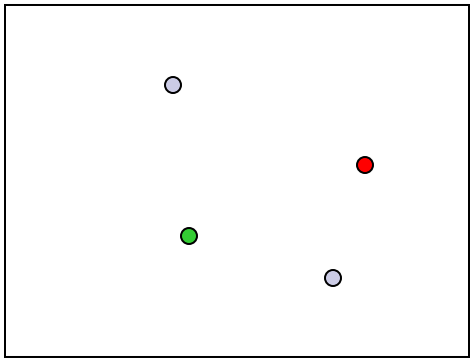
Static camera, moving scene, moving light

We still don’t touch these areas

Recovering motion



$I(x,y,t)$

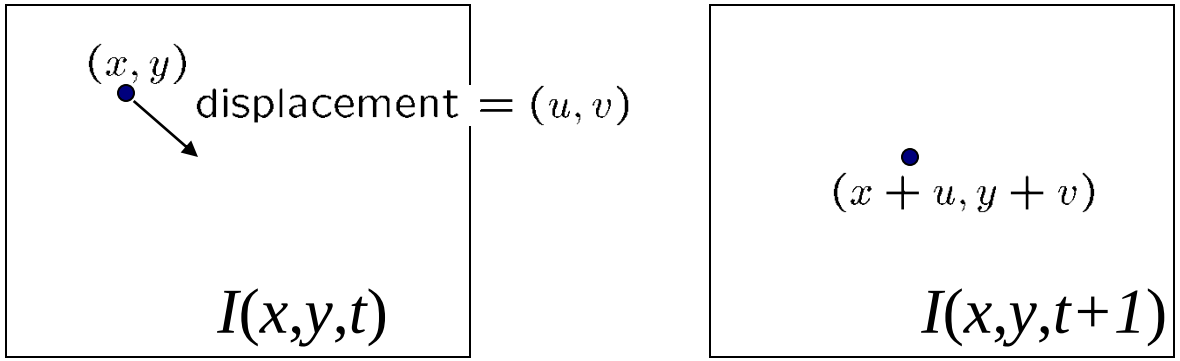


$I(x,y,t+1)$

- Given two subsequent frames, estimate the point translation
- Key assumptions of Lucas-Kanade Tracker
  - **Brightness constancy:** projection of the same point looks the same in every frame
  - **Small motion:** points do not move very far
  - **Spatial coherence:** points move like their neighbors

B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 674–679, 1981.

The brightness constancy constraint



- Brightness Constancy Equation:

$$I(x,y,t) = I(x+u, y+v, t+1)$$

Take Taylor expansion of $I(x+u, y+v, t+1)$ at (x,y,t) to linearize the right side:

Image derivative along x Difference over frames

$$I(x+u,y+v,t+1) \approx I(x,y,t) + I_x \cdot u + I_y \cdot v + I_t$$

$$I(x+u,y+v,t+1) - I(x,y,t) = I_x \times u + I_y \times v + I_t$$

So: $$I_x \cdot u + I_y \cdot v + I_t \approx 0 \quad \rightarrow \nabla \mathrm{I} \cdot \left[\mathrm{u}\ \mathrm{v}\right]^{\mathrm{T}} + \mathrm{I_t} = 0$$

The brightness constancy constraint

Can we use this equation to recover image motion (u,v) at each pixel?

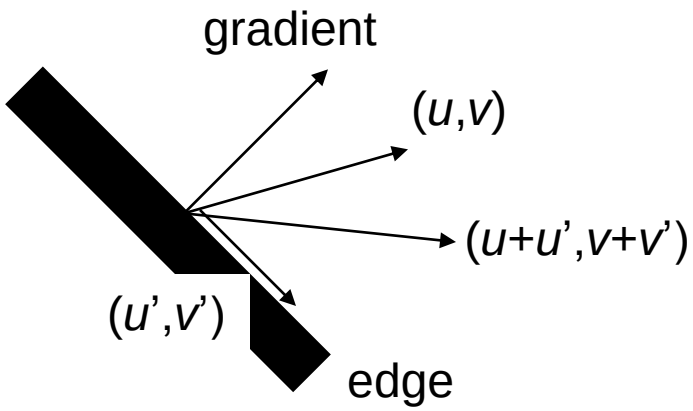
$$\nabla I \cdot \begin{bmatrix} u & v \end{bmatrix}^T + I_t = 0$$

- How many equations and unknowns per pixel?
  - One equation (this is a scalar equation!), two unknowns (u,v)

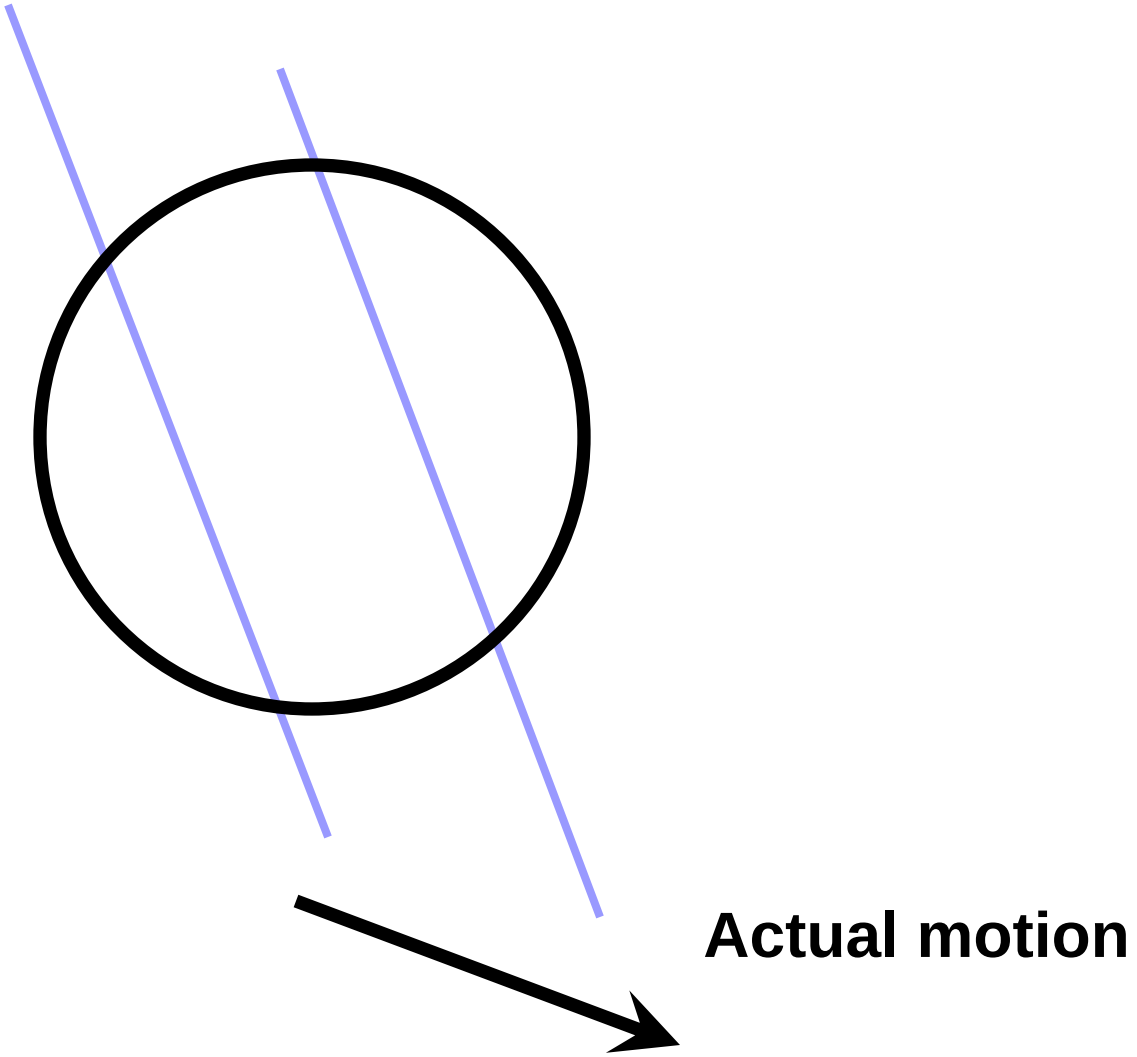
The component of the motion perpendicular to the gradient (i.e., parallel to the edge) cannot be measured

If (u, v) satisfies the equation, so does (u+u', v+v') if

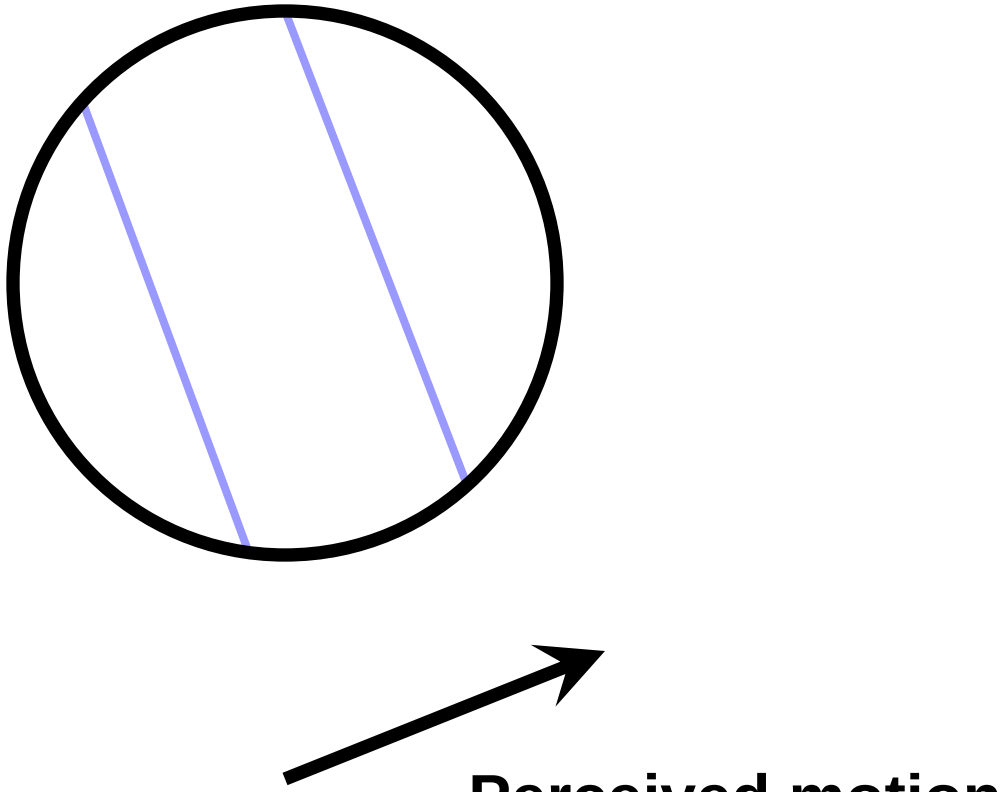
$$\nabla I \cdot \begin{bmatrix} u' & v' \end{bmatrix}^T = 0$$



The aperture problem



The aperture problem



Solving the ambiguity…

B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 674–679, 1981.

- How to get more equations for a pixel?
- **Spatial coherence constraint**
- Assume the pixel's neighbors have the same (u,v)
  - If we use a 5x5 window, that gives us 25 equations per pixel

$$0 = I_t(\mathbf{p_i}) + \nabla I(\mathbf{p_i}) \cdot [u\ v]$$

$$\begin{bmatrix} I_x(\mathbf{p_1}) & I_y(\mathbf{p_1}) \\ I_x(\mathbf{p_2}) & I_y(\mathbf{p_2}) \\ \vdots & \vdots \\ I_x(\mathbf{p_{25}}) & I_y(\mathbf{p_{25}}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p_1}) \\ I_t(\mathbf{p_2}) \\ \vdots \\ I_t(\mathbf{p_{25}}) \end{bmatrix}$$

Solving the ambiguity…

- Least squares problem:

$$\begin{bmatrix} I_x(\mathrm{p}_1) & I_y(\mathrm{p}_1) \\ I_x(\mathrm{p}_2) & I_y(\mathrm{p}_2) \\ \vdots & \vdots \\ I_x(\mathrm{p}_{25}) & I_y(\mathrm{p}_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathrm{p}_1) \\ I_t(\mathrm{p}_2) \\ \vdots \\ I_t(\mathrm{p}_{25}) \end{bmatrix}$$

$$\underset{25\text{x}2}{A} \quad \underset{2\text{x}1}{d} = \underset{25\text{x}1}{b}$$

Local Smoothness

Matching patches across images

- Overconstrained linear system

$$\begin{bmatrix} I_x(\mathbf{p}_1) & I_y(\mathbf{p}_1) \\ I_x(\mathbf{p}_2) & I_y(\mathbf{p}_2) \\ \vdots & \vdots \\ I_x(\mathbf{p}_{25}) & I_y(\mathbf{p}_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p}_1) \\ I_t(\mathbf{p}_2) \\ \vdots \\ I_t(\mathbf{p}_{25}) \end{bmatrix}$$

$$\underset{25\text{x}2}{A} \quad \underset{2\text{x}1}{d} = \underset{25\text{x}1}{b}$$

Least squares solution for *d* given by $(A^T A)\ d = A^T b$

$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

$A^T A$ $\qquad\qquad$ $A^T b$

The summations are over all pixels in the K x K window

Conditions for solvability

Optimal (u, v) satisfies Lucas-Kanade

$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

$A^T A$ $\qquad$ $A^T b$

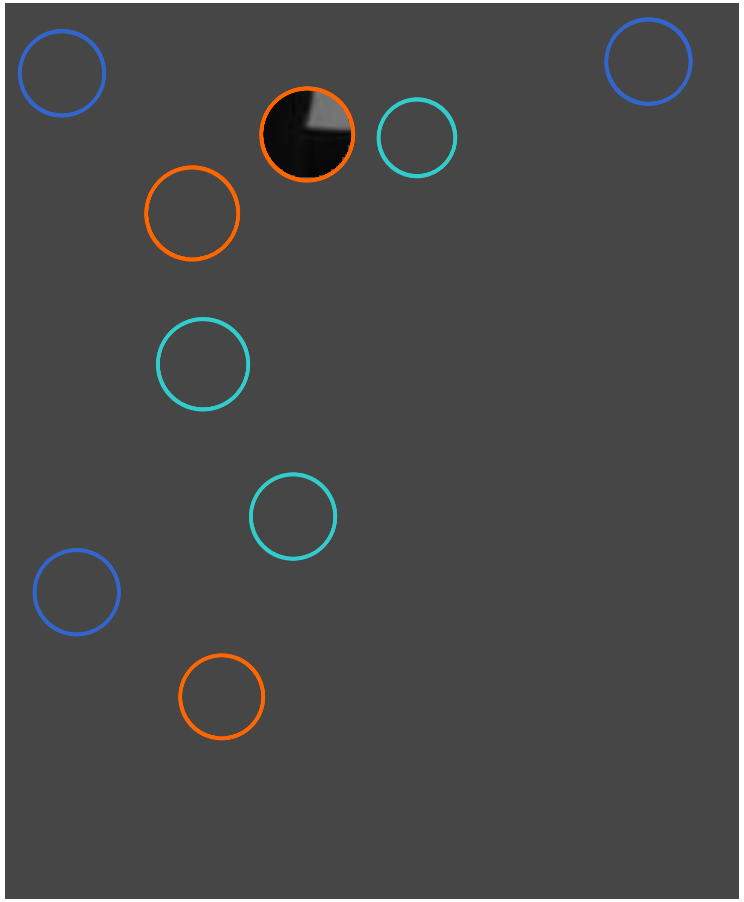
When is this solvable? I.e., what are good points to track?

- $\mathbf{A^T A}$ should be invertible
- $\mathbf{A^T A}$ should not be too small due to noise
  - eigenvalues λ_1 and λ_2 of $\mathbf{A^T A}$ should not be too small
- $\mathbf{A^T A}$ should be well-conditioned
  - λ_1 / λ_2 should not be too large (λ_1 = larger eigenvalue)

Does this remind you of anything?

Criteria for Harris corner detector

Aperture problem



Corners

Lines

Flat regions

Low-texture region

$$\sum \nabla I (\nabla I)^T$$

– gradients have small magnitude

– small λ_1, small λ_2

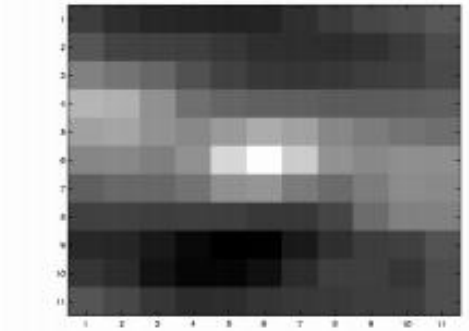
Edge

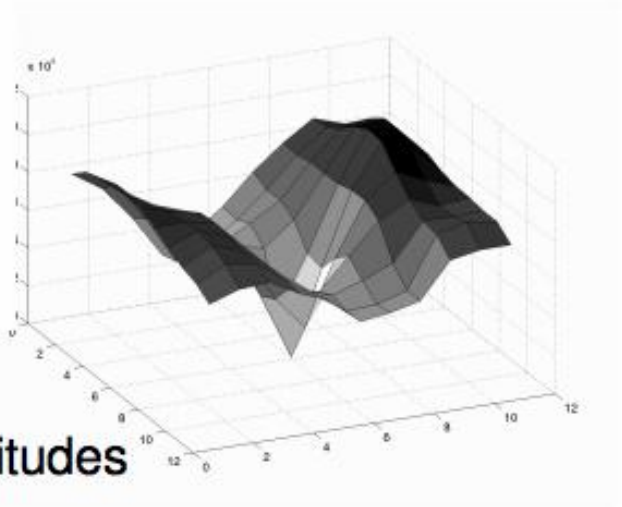
$$\sum \nabla I (\nabla I)^T$$

– gradients very large or very small

– large λ_1, small λ_2

High Texture Region



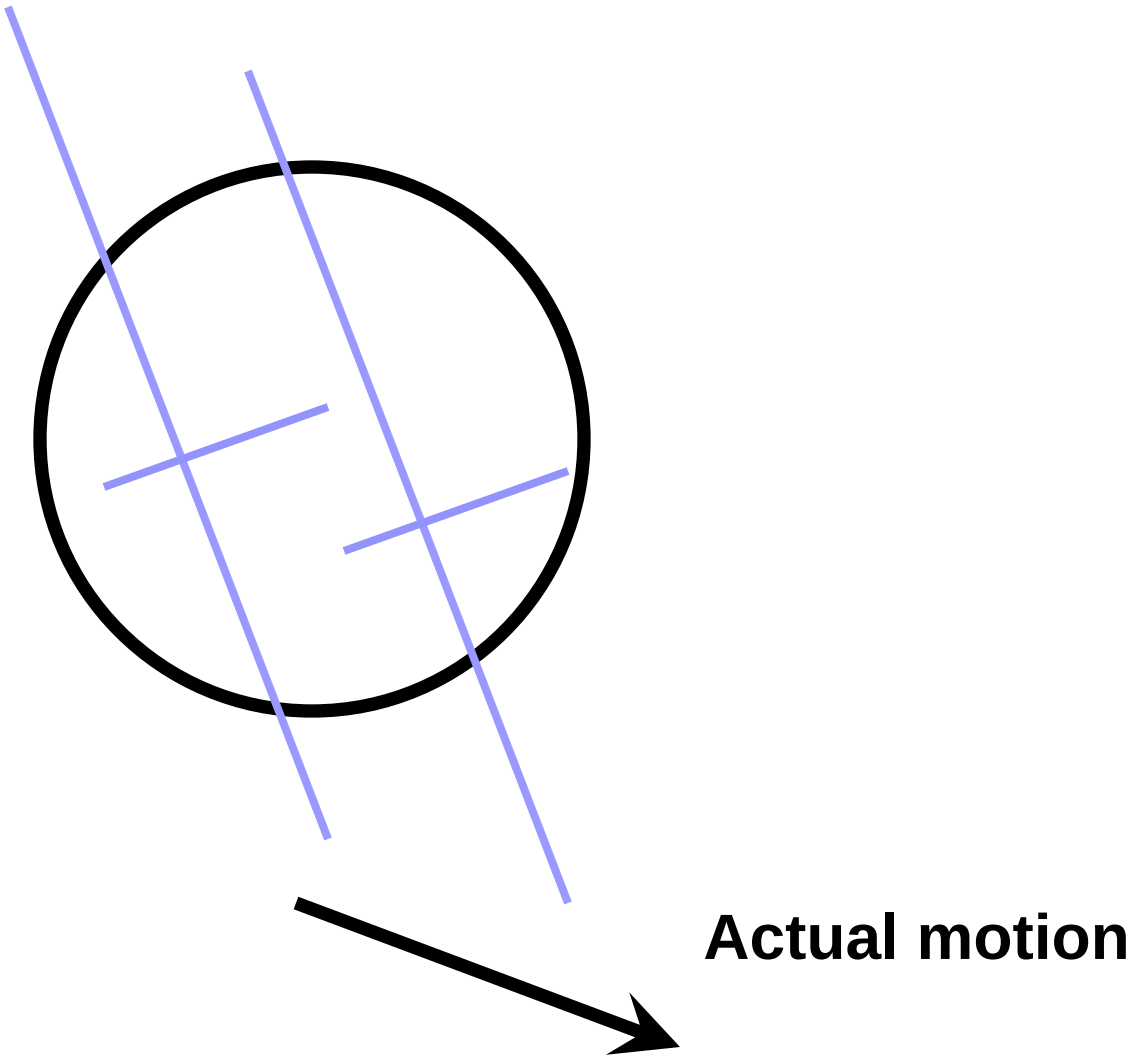


$$\sum \nabla I (\nabla I)^T$$

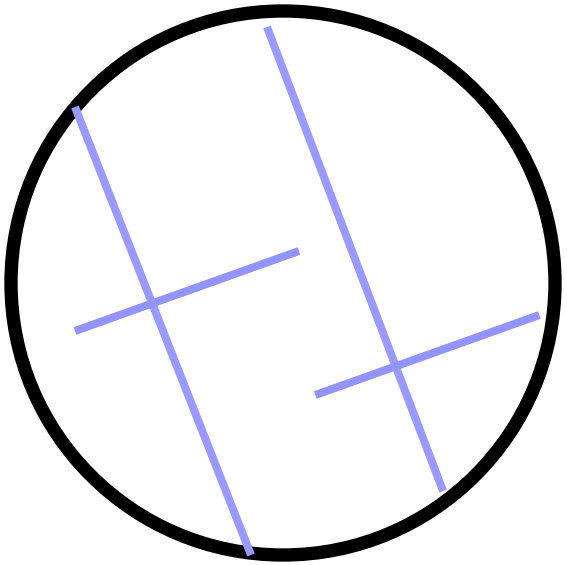
– gradients are different, large magnitudes

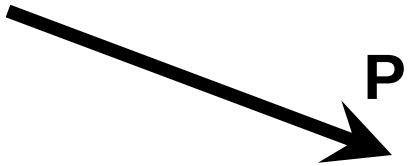
– large λ_1, large λ_2

The aperture problem resolved



The aperture problem resolved





Perceived motion

Global Smoothness Method

Determining Optical Flow, Horn B K, Schunck B G. 1981

Lucas and Kanade assume the pixel's neighbors have the same motion——local smooth method, Horn and Schunck put forward a global smooth method, formed as:

$$E = E_{data} + \lambda E_{Smooth}$$

- Data term: $E_{Data} = \sum_x \left(I_x u + I_y v + I_t \right)^2$
- Smooth term: $E_{Smooth} = \sum_x \left(u_x^2 + u_y^2 + v_x^2 + v_y^2 \right)$
- Solved with calculus of variations

Break-downs

- Brightness constancy is not perfect satisfied.
  - → Robust penalty function.
- A point does not move like its neighbors.
  - Motion discontinuity
    - → anisotropic, bilateral weight
- The motion is not small(Taylor expansion doesn’t hold).
  - → Coarse-to-fine(multi-scale) estimation.

Robust penalty function

- 1.Quadratic: $\rho(x)=x^2$
- 2.Truncated quadratic: $\rho_{\alpha,\lambda}(x)=\begin{cases}\lambda x^2 & if \quad x<\dfrac{\sqrt{\alpha}}{\sqrt{\lambda}}\\ \alpha & otherwise\end{cases}$

Calculus of variation

- 3.Lorentzian: $\rho_\sigma(x)=\log\left(1+\frac{1}{2}\left(\frac{x}{\sigma}\right)^2\right)$ → **GNC**
- 4.L1 Norm: $\rho(x)=\|x\|$ → **TV**
- 5.Charbonnier: $\rho(x)=\sqrt{\left(x^2+\varepsilon^2\right)}$
- 6.Generalized Charbonnier: $\rho(x)=\left(x^2+\varepsilon^2\right)^\alpha$

IRLS

- 7.M-estimator: $\rho(t;\sigma)=\dfrac{t^2}{t^2+\sigma^2}$

- Huber: $\rho(t;\sigma)=\begin{cases}t^2 & |t|<\sigma\\ \sigma^2 & |t|\geq\sigma\end{cases}$ $\rho(t;\sigma)=\begin{cases}t^2, & |t|<\sigma\\ \sigma/|t|, & \sigma\leq|t|<3\sigma\\ 0, & |t|\geq 3\sigma\end{cases}$

GNC：graduated non convex
TV：total variance
IRLS: iterated reweighted least square

Motion discontinuity

- 1. Anisotropic Smoothness：define a structure adaptive map according to the brightness derivatives.

$$w_{Smooth} = \exp\left(-\left\|\nabla I\right\|^{\kappa}\right)$$

"structure and motion adaptive regularization for high accuracy optical flow"(ICCV2009)

- 2.Bilateral weight:

$$w_{Smooth}\left(x,y\right) = \exp\left\{-\frac{\left\|x-y\right\|^2}{2\sigma_1^2} - \frac{\left\|I\left(x\right)-I\left(y\right)\right\|^2}{2\sigma_2^2}\right\}$$

"Efficient Nonlocal Regularization for optical flow"(ECCV2012)

Dealing with larger movements: Iterative refinement

Original (x,y) position

$I_t = I(x', y', t+1) - I(x, y, t)$

1. Initialize (x',y') = (x,y)
2. Compute (u,v) by

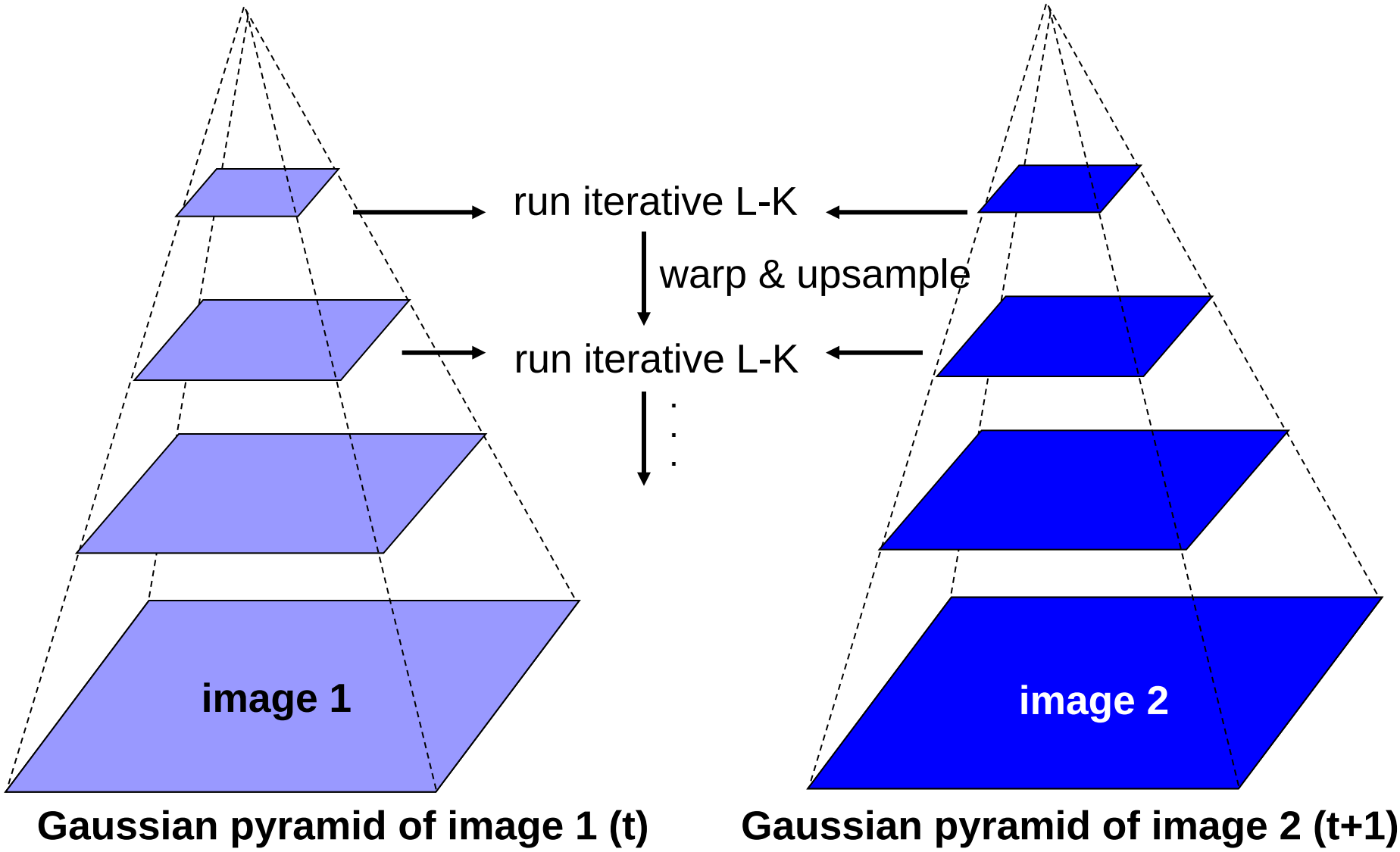
$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

2nd moment matrix for feature patch in first image

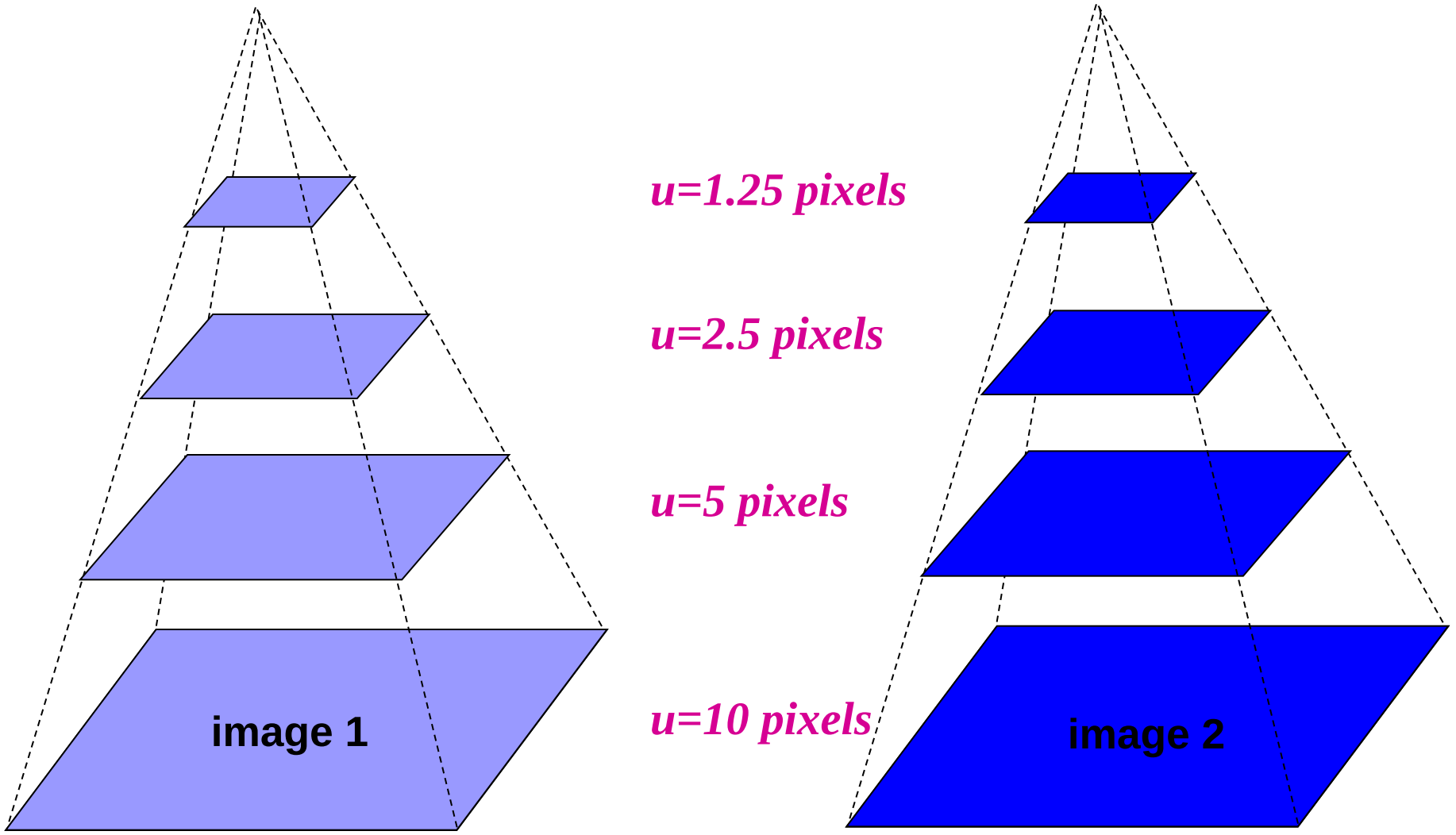
displacement

3. Shift window by (u, v): `x'=x'+u; y'=y'+v;`
4. Recalculate I_t
5. Repeat steps 2-4 until small change
   - Use interpolation for subpixel values

Coarse-to-fine optical flow estimation



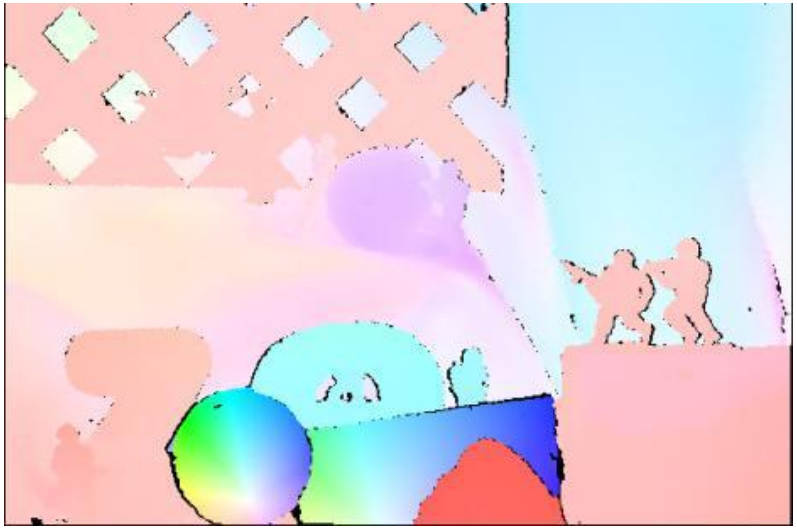
Coarse-to-fine optical flow estimation



Gaussian pyramid of image 1

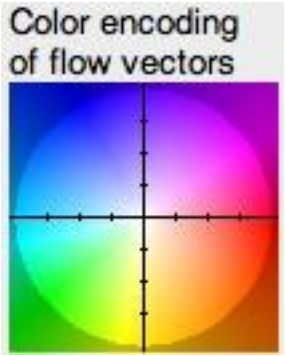
Gaussian pyramid of image 2

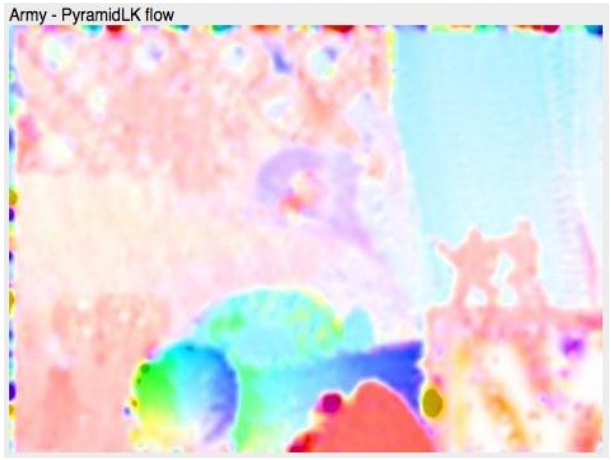
Flow quality evaluation



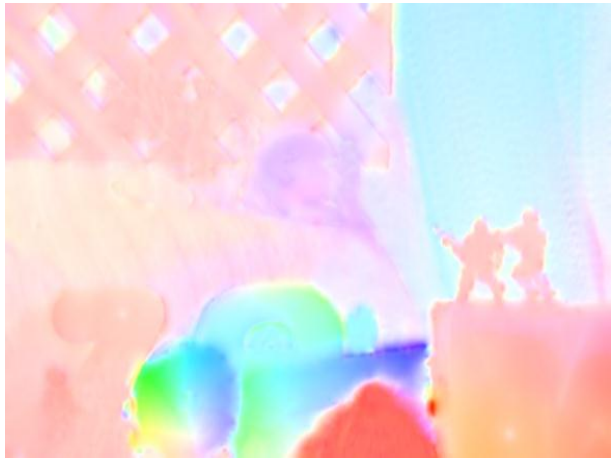
Ground Truth

- Middlebury flow page
  - http://vision.middlebury.edu/flow/





Lucas-Kanade



HS

GNC

Brox

TV-L1 Improved

MDP

Video stabilization

Video denoising

Video super resolution

Motion Detail Preserving Optical Flow Estimation

- Framework
  - Extended coarse-to-fine motion estimation for both large and small displacement optical flow
- Model
  - A new data term to selectively combine constraints
- Solver
  - Efficient numerical solver for discrete-continuous optimization

from “http://www.cse.cuhk.edu.hk/~leojia/projects/flow/”

Extended Flow Initialization

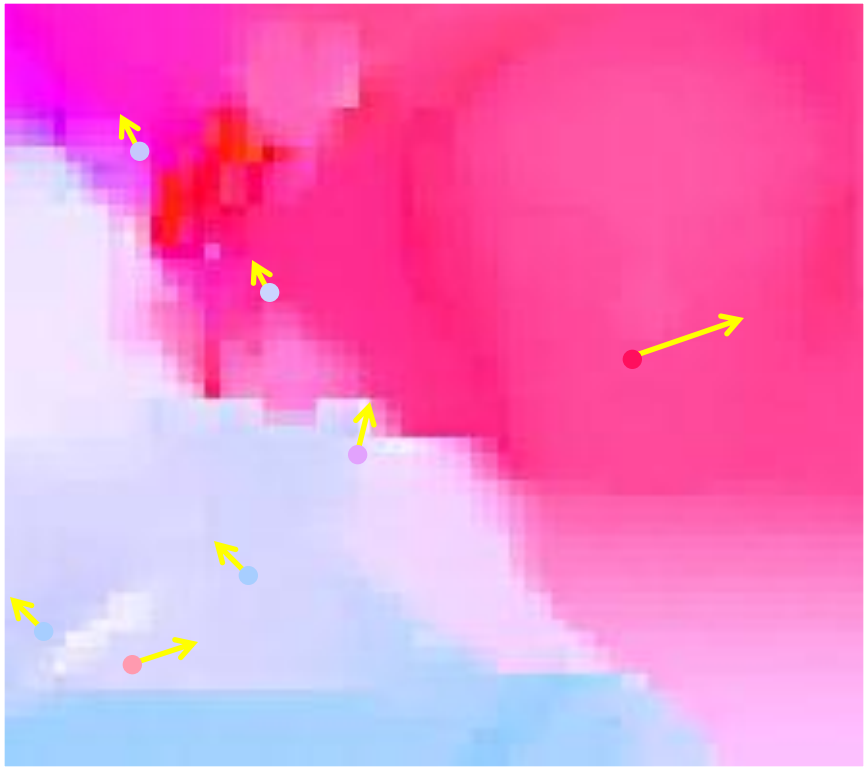
- Sparse feature matching for each level

Extended Flow Initialization

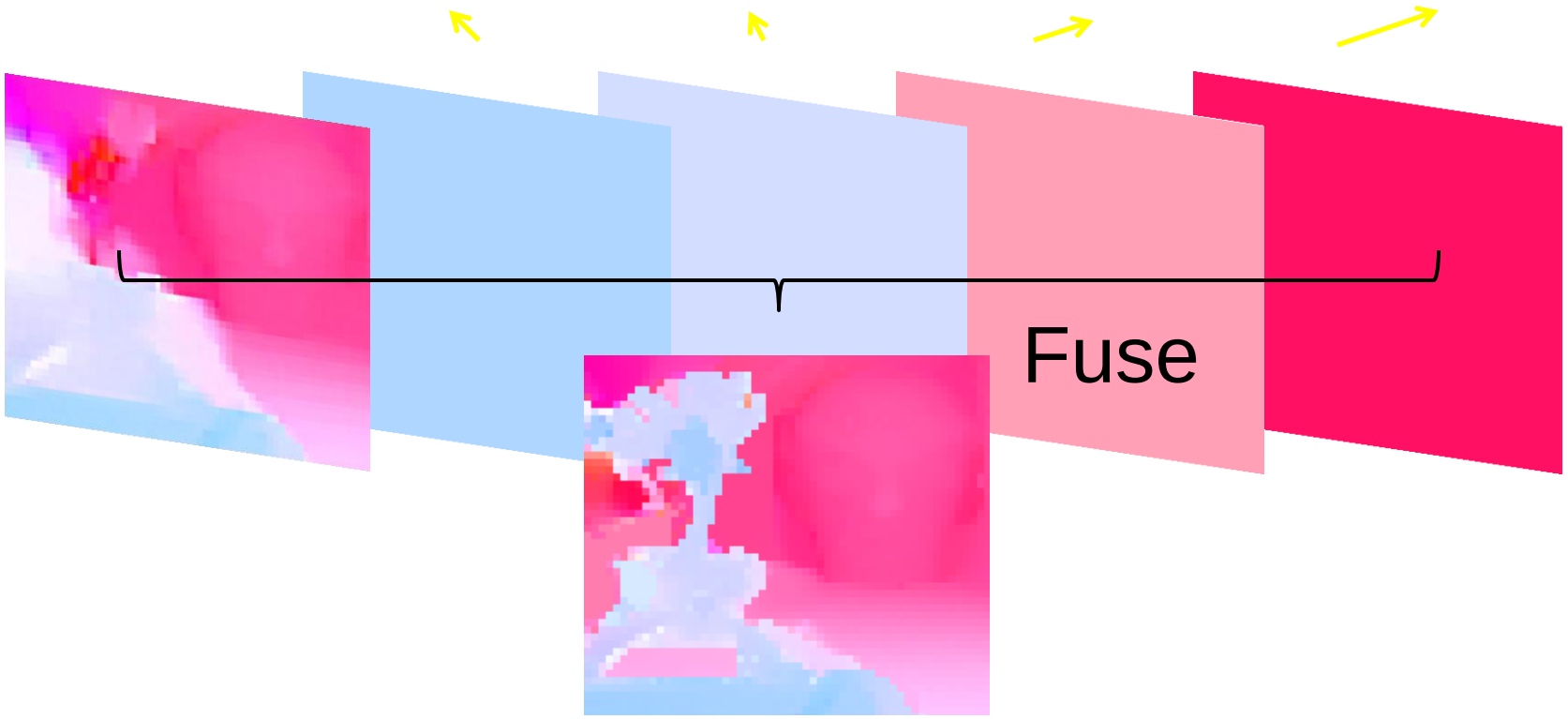
- Identify missing motion vectors

Extended Flow Initialization

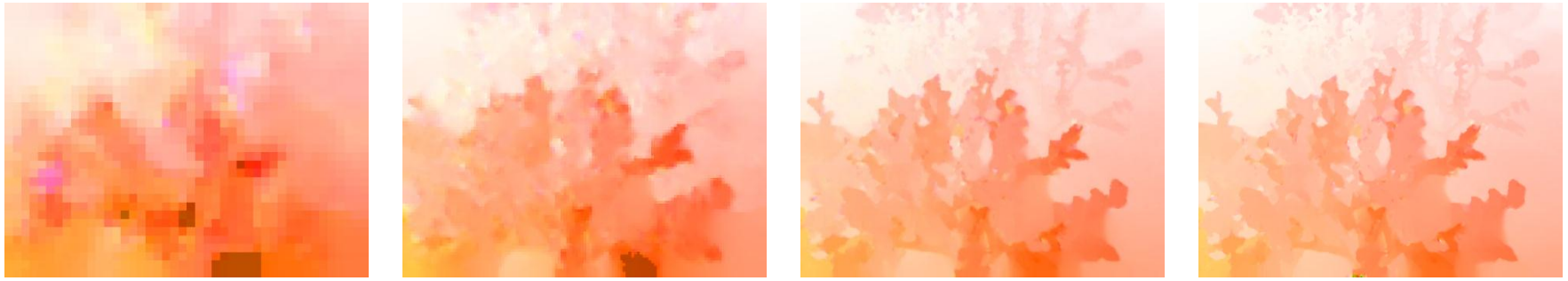
- Identify missing motion vectors



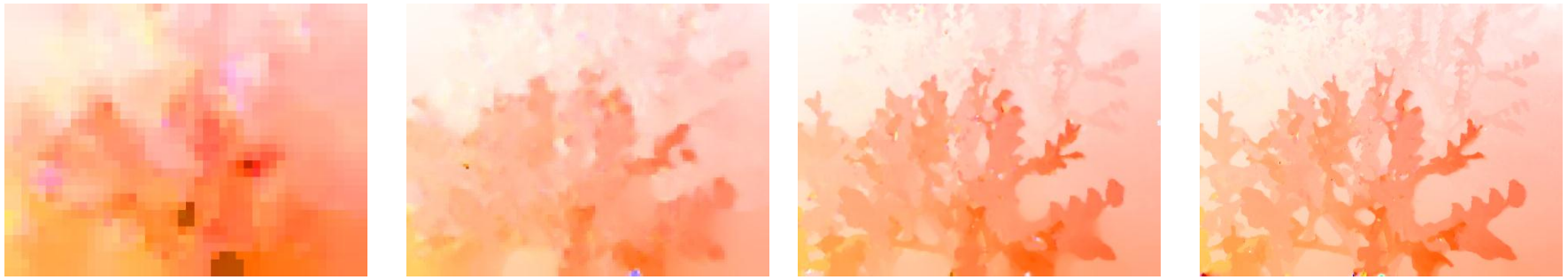
Extended Flow Initialization



Results from Different Steps



Coarse-to-fine



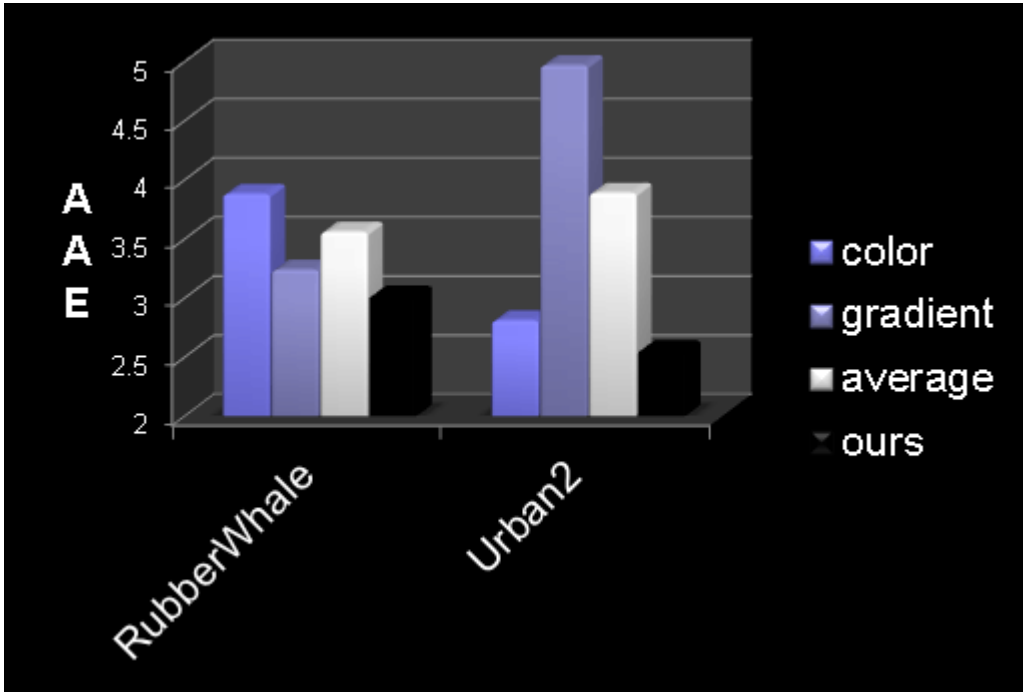
Extended coarse-to-fine

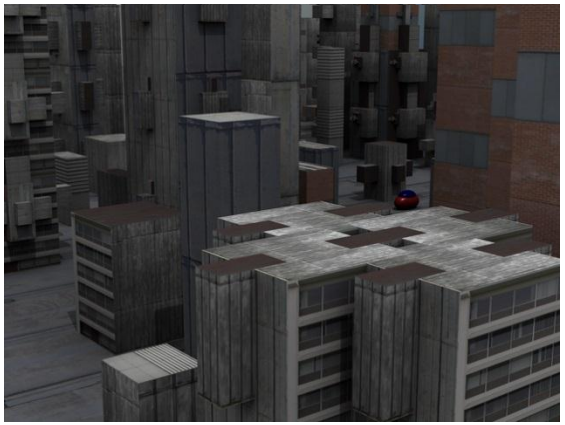
- Selectively combine the constraints

$$E_D(\mathrm{u},\alpha)=\sum_{\mathrm{x}}\alpha(\mathrm{x})D_{\mathrm{I}}(\mathrm{u},\mathrm{x})+(1-\alpha(\mathrm{x}))D_{\nabla\mathrm{I}}(\mathrm{u},\mathrm{x})$$

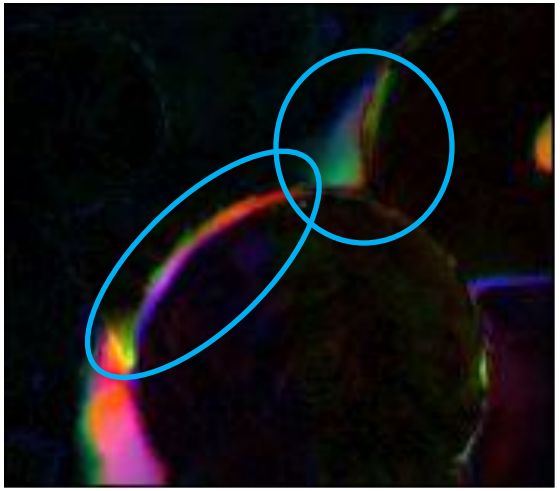
where $\alpha(\mathrm{x}):Z^2\mapsto\{0,1\}$

Comparisons





Results



Difference

Averaging constraints

Ours

More Results

Overlaid Input

Coarse-to-fine

Our Results

Challenge Problems

LARGE DISPLACEMENT **OCCLUSION**

always appear simultaneously

Average endpoint error	avg. rank	Army (Hidden texture) GT			Mequon (Hidden texture) GT			Schefflera (Hidden texture) GT			Wooden (Hidden texture) GT			Grove (Synthetic) GT			Urban (Synthetic) GT			Yosemite (Synthetic) GT			Teddy (Stereo) GT		
		all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext
IVANN [87]	2.7	0.07 1	0.20 2	.05 1	0.15 1	.51 3	0.12 5	0.18 1	0.37 1	0.14 1	0.10 2	0.49 3	0.06 2	0.41 1	0.61 1	0.21 2	0.23 2	.66 2	0.19 1	0.10 4	0.12 9	0.17 12	0.34 1	0.80 4	0.23 2
OFLAF [77]	6.9	0.08 7	0.21 3	.06 5	0.16 5	.53 4	0.12 5	0.19 2	0.37 1	0.14 1	0.14 7	0.77 23	0.07 4	0.51 4	0.78 5	0.25 3	0.31 5	.76 3	0.25 7	0.11 11	0.12 9	0.21 30	0.42 7	0.78 2	0.63 12
MDP-Flow2 [68]	7.9	0.08 7	0.21 3	.07 14	0.15 1	.48 1	0.11 1	0.20 4	0.40 4	0.14 1	0.15 18	0.80 29	0.08 10	0.63 14	0.93 14	0.43 15	0.26 3	.76 3	0.23 6	0.11 11	0.12 9	0.17 12	0.38 3	0.79 3	0.44 4
NN-field [71]	8.7	0.08 7	0.22 14	.05 1	0.17 7	.55 6	0.13 10	0.19 2	0.39 3	0.15 6	0.09 1	0.48 2	0.05 1	0.41 1	0.61 1	0.20 1	0.52 46	.64 1	0.26 10	0.13 31	0.13 26	0.20 23	0.35 2	0.83 5	0.21 1
ComponentFusion [96]	10.0	0.07 1	0.21 3	.05 1	0.16 5	.55 6	0.12 5	0.20 4	0.44 7	0.15 6	0.11 3	0.65 6	0.06 2	0.71 27	1.07 32	0.53 29	0.32 7	.06 20	0.28 13	0.11 11	0.13 26	0.15 7	0.41 6	0.88 9	0.54 5
WLIF-Flow [93]	14.7	0.08 7	0.21 3	.06 5	0.18 9	.55 6	0.15 19	0.25 13	0.56 14	0.17 12	0.14 7	0.68 7	0.08 10	0.61 12	0.91 13	0.41 13	0.43 23	.96 11	0.29 18	0.13 31	0.12 9	0.21 30	0.51 27	1.03 28	0.72 26
TC/T-Flow [76]	14.9	0.07 1	0.21 3	.05 1	0.19 14	.68 26	0.12 5	0.28 18	0.66 23	0.14 1	0.14 7	0.86 37	0.07 4	0.67 24	0.98 23	0.49 24	0.22 1	.82 5	0.19 1	0.11 11	0.11 1	0.30 67	0.50 22	1.02 25	0.64 14
Layers++ [37]	16.5	0.08 7	0.21 3	.07 14	0.19 14	.56 9	0.17 26	0.20 4	0.40 4	0.18 18	0.13 6	0.58 4	0.07 4	0.48 3	0.70 3	0.33 6	0.47 34	.01 14	0.33 35	0.15 52	0.14 47	0.24 42	0.46 12	0.88 9	0.72 26
LME [70]	17.0	0.08 7	0.22 14	.06 5	0.15 1	.49 2	0.11 1	0.30 26	0.64 18	0.31 69	0.15 18	0.78 26	0.09 23	0.66 20	0.96 19	0.53 29	0.33 8	.18 32	0.28 13	0.12 22	0.12 9	0.18 16	0.44 8	0.91 11	0.61 10
IROF++ [58]	17.4	0.08 7	0.23 19	.07 14	0.21 26	.68 26	0.17 26	0.28 18	0.63 17	0.19 30	0.15 18	0.73 19	0.09 23	0.60 10	0.89 10	0.42 14	0.43 23	.08 23	0.31 25	0.10 4	0.12 9	0.12 4	0.47 14	0.98 18	0.68 21
nLayers [57]	17.6	0.07 1	0.19 1	.06 5	0.22 32	.59 12	0.19 45	0.25 13	0.54 11	0.20 38	0.15 18	0.84 34	0.08 10	0.53 5	0.78 5	0.34 8	0.44 27	.84 6	0.30 22	0.13 31	0.13 26	0.20 23	0.47 14	0.97 17	0.67 19
FC-2Layers-FF [74]	19.6	0.08 7	0.21 3	.07 14	0.21 26	.70 30	0.17 26	0.20 4	0.40 4	0.18 18	0.15 18	0.76 22	0.08 10	0.53 5	0.77 4	0.37 9	0.49 40	.02 15	0.33 35	0.16 62	0.13 26	0.29 63	0.44 8	0.87 8	0.64 14

Recent Advances

- For Occlusion: explicitly detection and handling
  - **Cross-check**:" Fast cost volume filtering for visual correspondence and beyond"(CVPR2011)
  - **Layered model** (depth order):" Layered image motion with explicit occlusions, temporal consistency, and depth ordering"(NIPS2010)
  - **Exploit motion cues**: "Occlusion boundary detection and figure-ground assignment from optical flow"(CVPR2011)
  - **Occ layer**: "Joint Motion Estimation and Segmentation of Complex Scenes with Label Costs and Occlusion Modeling"(CVPR2012)
  - **Binary classification**: "Locally affine sparse-to-dense matching for motion and occlusion estimation"(ICCV2013)
  - **Occ relationship inference**: "Local Layering for Joint Motion Estimation and Occlusion Detection"(CVPR2014)

Recent Advances

- For Large Displacement: add correspondence constraint.
  - “Large displacement Pattern Recognition from **nearest neighbor fields**” ”(CVPR2013) **CSH**
  - “Large displacement optical flow computation without warping”(ICCV2009)
  - “Large displacement optical flow(CVPR2009)
  - “Large displacement optical flow: descriptor matching in variational motion estimation”(PAMI2011)
  - “Motion detail preserving optical flow estimation”(CVPR2010,PAMI2012):**Fusion multi-candidate flow field**
  - “Deepflow: Large displacement optical flow with **deep matching**”(ICCV2013)
  - “Fast Edge-preserving **PatchMatch** for Large Displacement Optical Flow”(CVPR2014**):** random sample, bilateral weight.

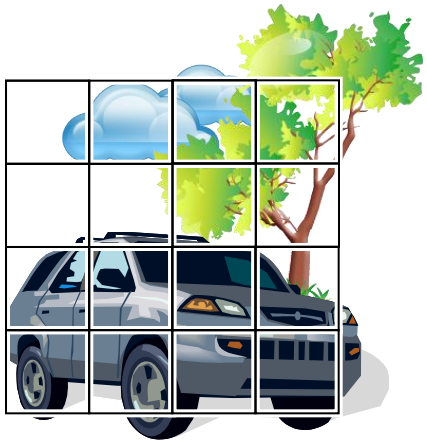
HOG

DeepFlow

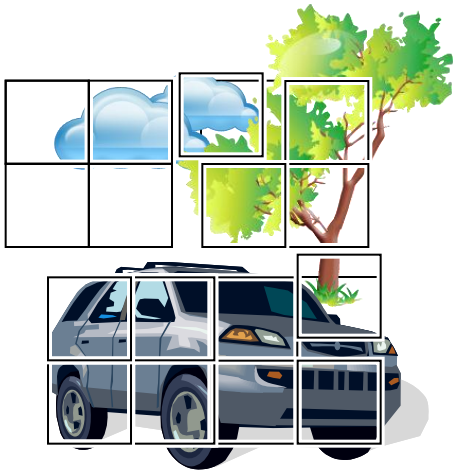
- Classical optical flow [Horn and Schunck 1981]
  - energy $E(\mathbf{w}) = \iint E_{data} + \alpha E_{smooth} \mathbf{dx}$
- Integration of Deep Matching
  - energy $E(\mathbf{w}) = \iint E_{data} + \alpha E_{smooth} + \beta E_{match} \mathbf{dx}$
  - matches guide the flow
  - similar to [Brox and Malik 2011]
- Minimization using:
  - coarse-to-fine strategy
  - fixed point iterations
  - Successive Over Relaxation (SOR)

from "http://lear.inrialpes.fr/src/deepmatching/"

Deep Matching: main idea



First image

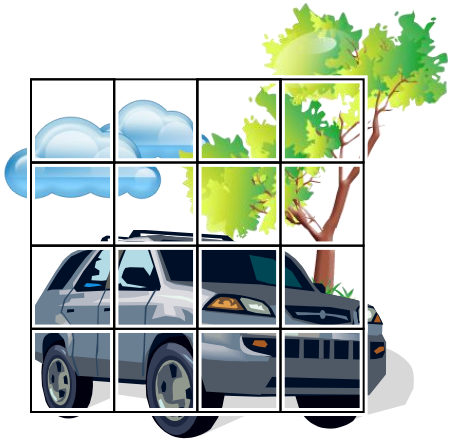


Second image

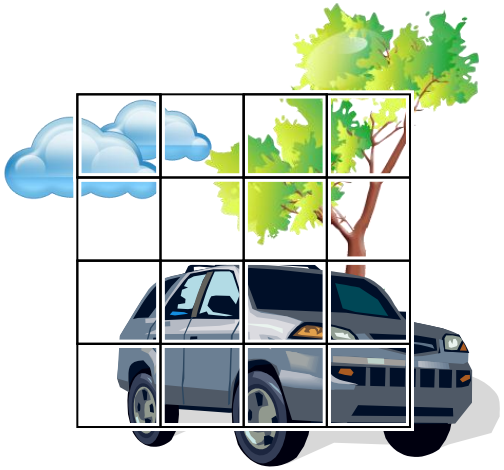
- We allow each subpatch to move:
  - independently
  - in a limited range depending on its size
- Our approach is fast to compute using convolution and max-pooling
- We apply this idea recursively

Related Work: matching

- HOG [Dalal and Triggs 2005] + Nearest Neighbor
  - rigid patches



Input patch

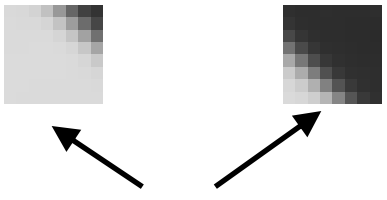


Rigid matching

- Small patches and propagation: PatchMatch [Barnes *et al.* 2010]
  - patches not sufficiently discriminative

Deep Matching (1)

Reference image

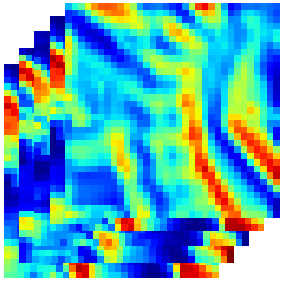


non-overlapping patches of 4x4 pixels

P:num of patches

nh: dim of desc

response maps



res:P*I

Target image

I:size of image

convolution

res:=patches * convolved desc

P * I = P * nh * nh

Image:

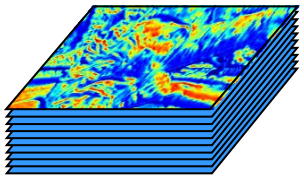
P = 1024/

res=377487360 4By =14.0625G!(225G)

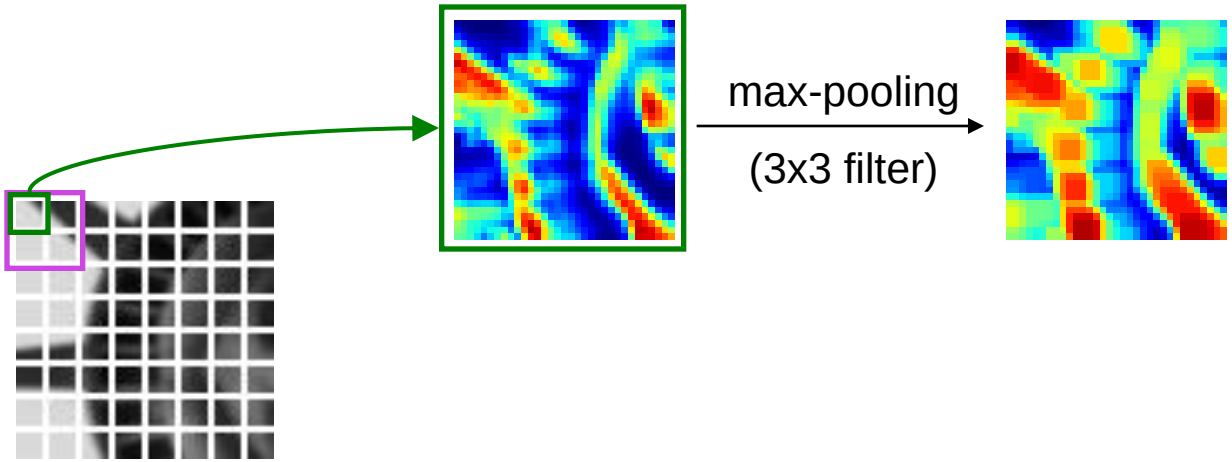
Memory cost!

Deep Matching (2)

response maps for
each 4x4 patch



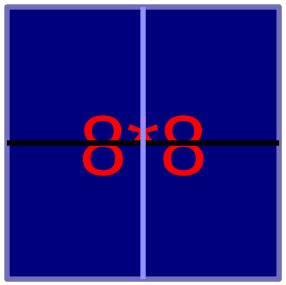
max-pooling



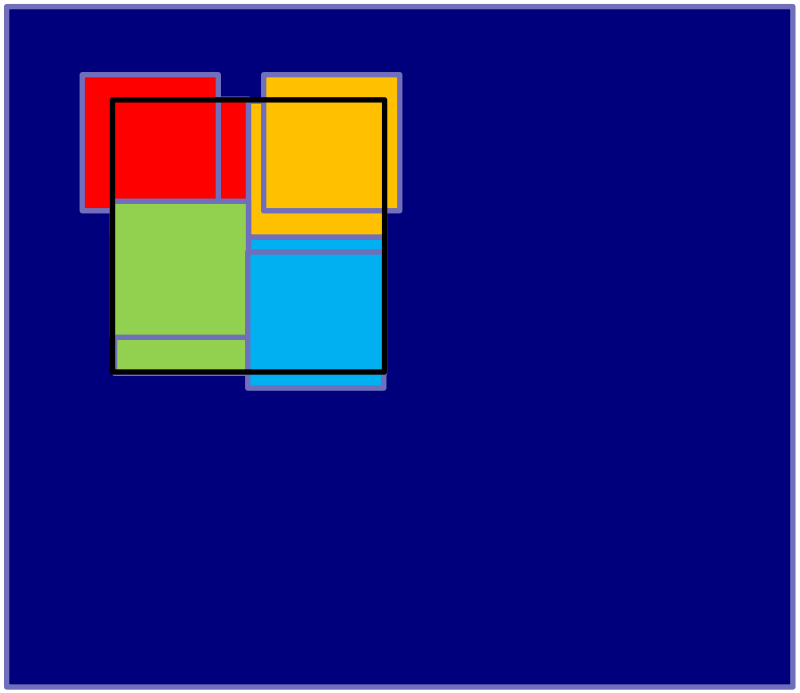
Max pooling

$$S(N) = \max_{t \in \left[-\frac{N}{8}, \frac{N}{8}\right]} S_{\text{left}}\left(\frac{N}{2}, t\right) + \max_{t \in \left[-\frac{N}{8}, \frac{N}{8}\right]} S_{\text{right}}\left(\frac{N}{2}, t\right) \cdot$$ (1D case)

4*4



patch



2D cases

For each 4*4 patch, small offset: [-1, 1] * [-1, 1], i.e:

$$\begin{Bmatrix} (-1,-1) & (-1,0) & (-1,1) \\ (0,-1) & (0,0) & (0,1) \\ (1,-1) & (1,0) & (1,1) \end{Bmatrix}$$

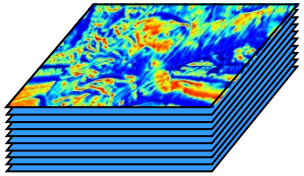
$$\rightarrow \begin{Bmatrix} -1 \\ 0 \\ 1 \end{Bmatrix} \times \{-1 \quad 0 \quad 1\}$$

$$e.g: \begin{Bmatrix} 1 & 21 & 15 \\ 27 & 40 & 50 \\ 54 & 80 & 36 \end{Bmatrix} \rightarrow \begin{Bmatrix} 27 & 40 & 50 \\ 54 & 80 & 50 \\ 54 & 80 & 50 \end{Bmatrix} \rightarrow \begin{Bmatrix} 40 & 50 & 50 \\ 80 & 80 & 80 \\ 80 & 80 & 80 \end{Bmatrix}$$

Response map

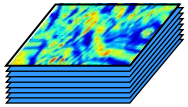
Deep Matching (2)

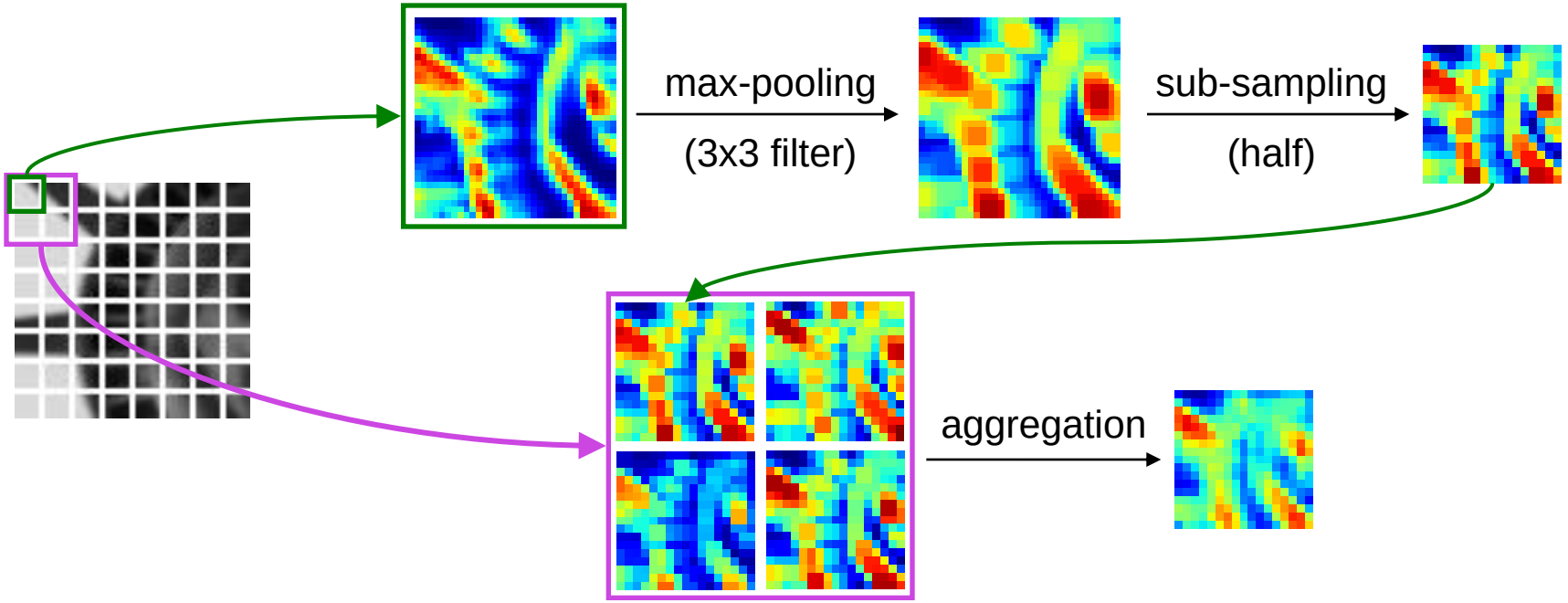
response maps for each 4x4 patch



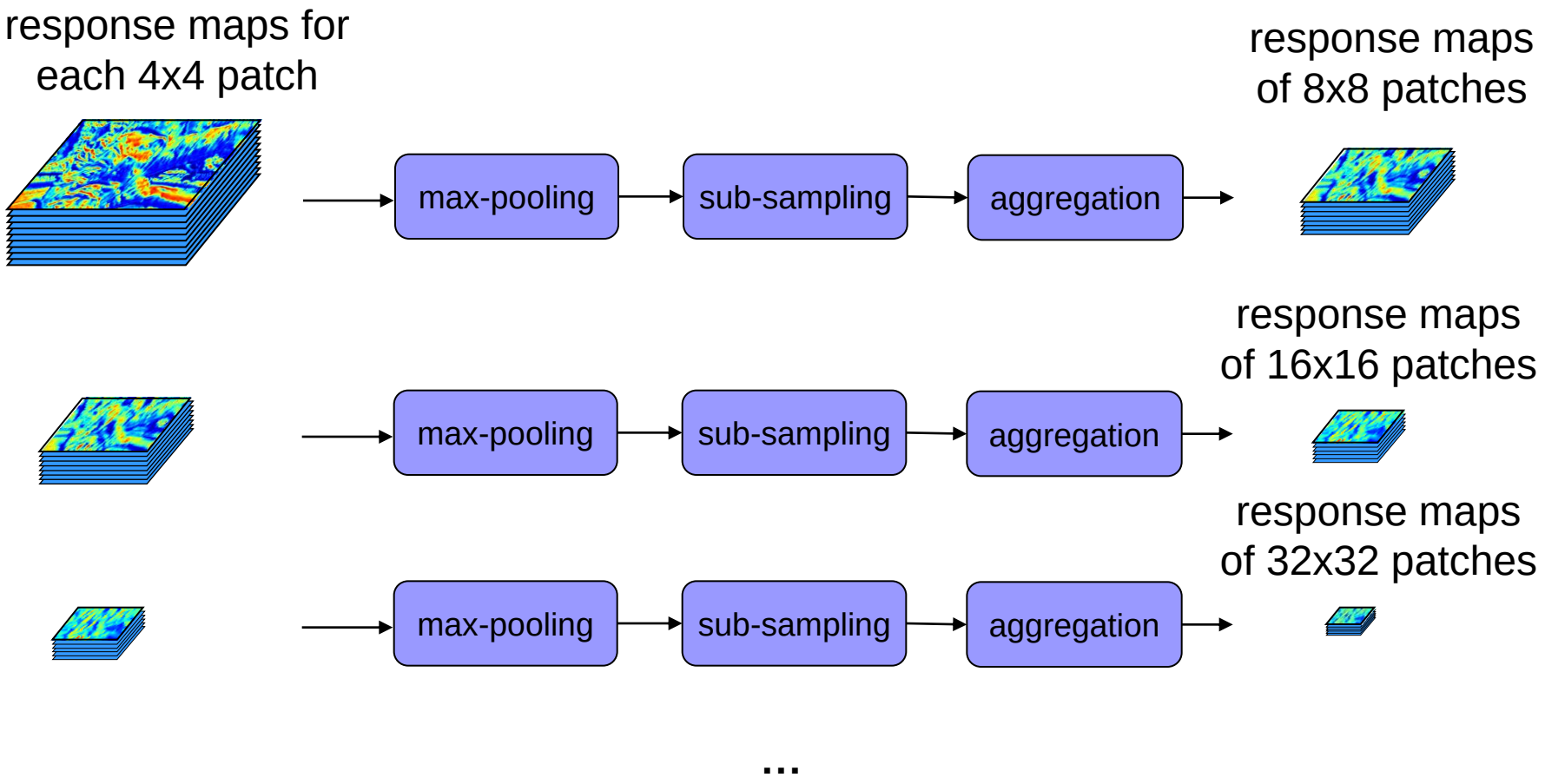
max-pooling → sub-sampling → aggregation

response maps of 8x8 patches



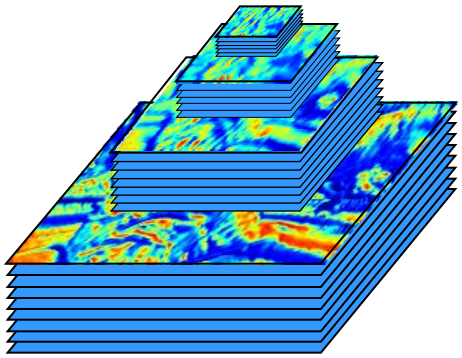


Deep Matching (2)



Pipeline similar in spirit to **deep** convolutional nets [Lecun *et al.* 1998]

Deep Matching (3)



Multi-scale response pyramid

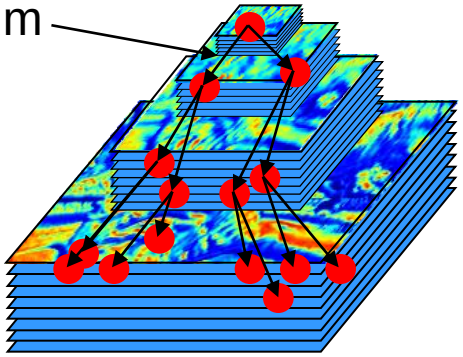
→ Extract scale-space local maxima → Backtrack quasi-dense correspondences

Bottom-up **Top-down**

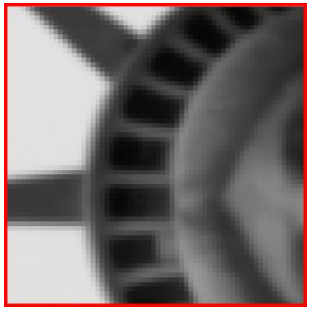
Deep Matching (3)

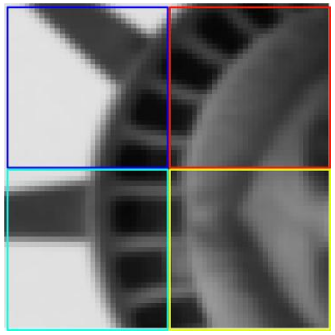
Merge the correspondences extracted from all local maxima

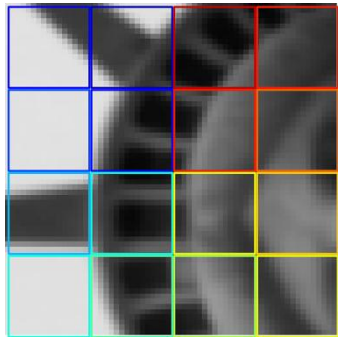
$$weight = sim(\mathrm{P}_4, \mathrm{P}_4^{'}) \cdot l \cdot S\left(w^*\right)$$

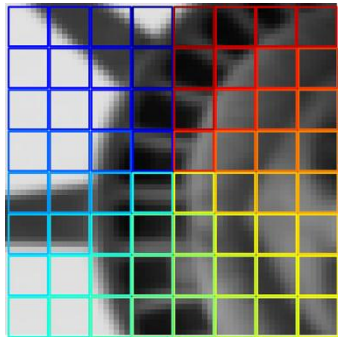


First image

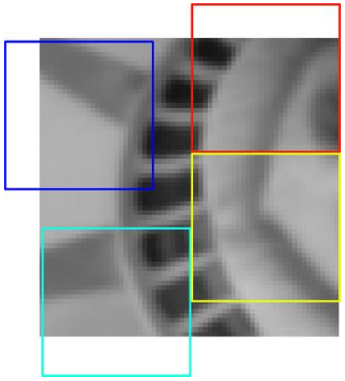


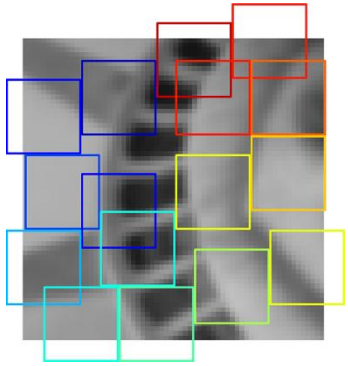


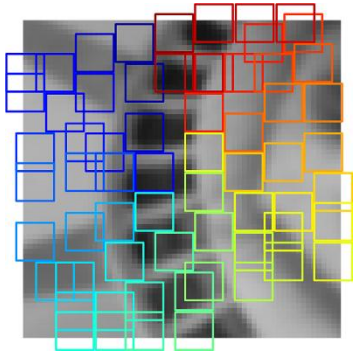




Second image



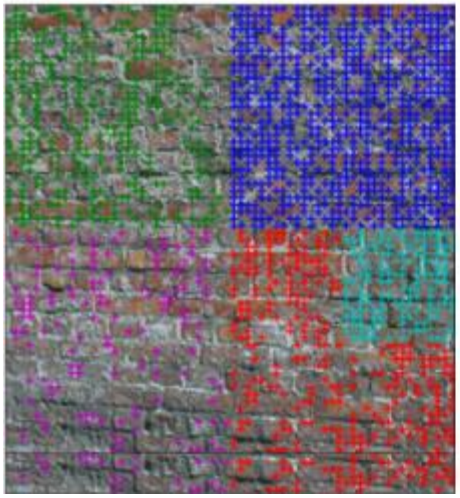


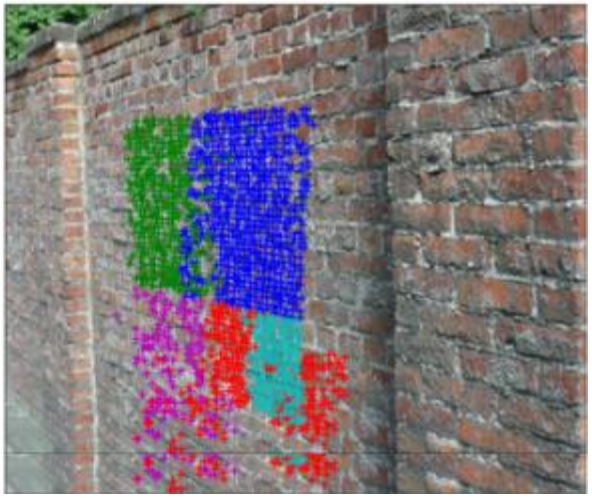


Deep Matching: example results

- Repetitive textures

First image *Second image*

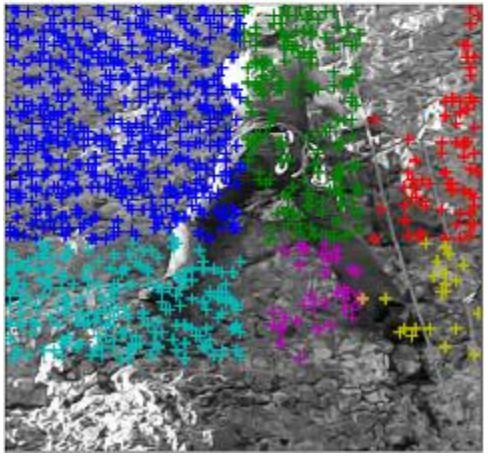


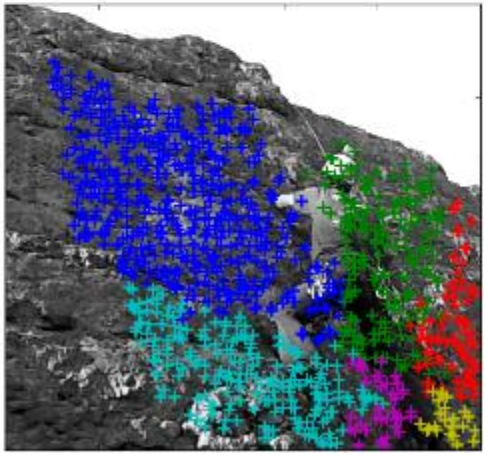


Deep Matching: example results

- Non-rigid deformation

First image *Second image*





特征匹配与光流总结

- 稀疏对应 - 特征点匹配
  - Invariance: Translation, Orientation (orientation normalization), Scale (difference of Gaussian), Lighting condition (gradient), Object deformations, Partial occlusion
  - Interest point detection, description, descriptor matching
  - Corner (spatial non-maximum suppression), SIFT (scale space), SURF, HOG, ….

特征匹配与光流总结

- 稠密对应—光流估计
  - Brightness constancy
    - Robust penalty function
  - Small motion
    - Coarse-to-fine (multi-scale) estimation (Taylor expansion doesn’t hold)
  - Spatial coherence
    - Anisotropic, bilateral weight

Thank you!