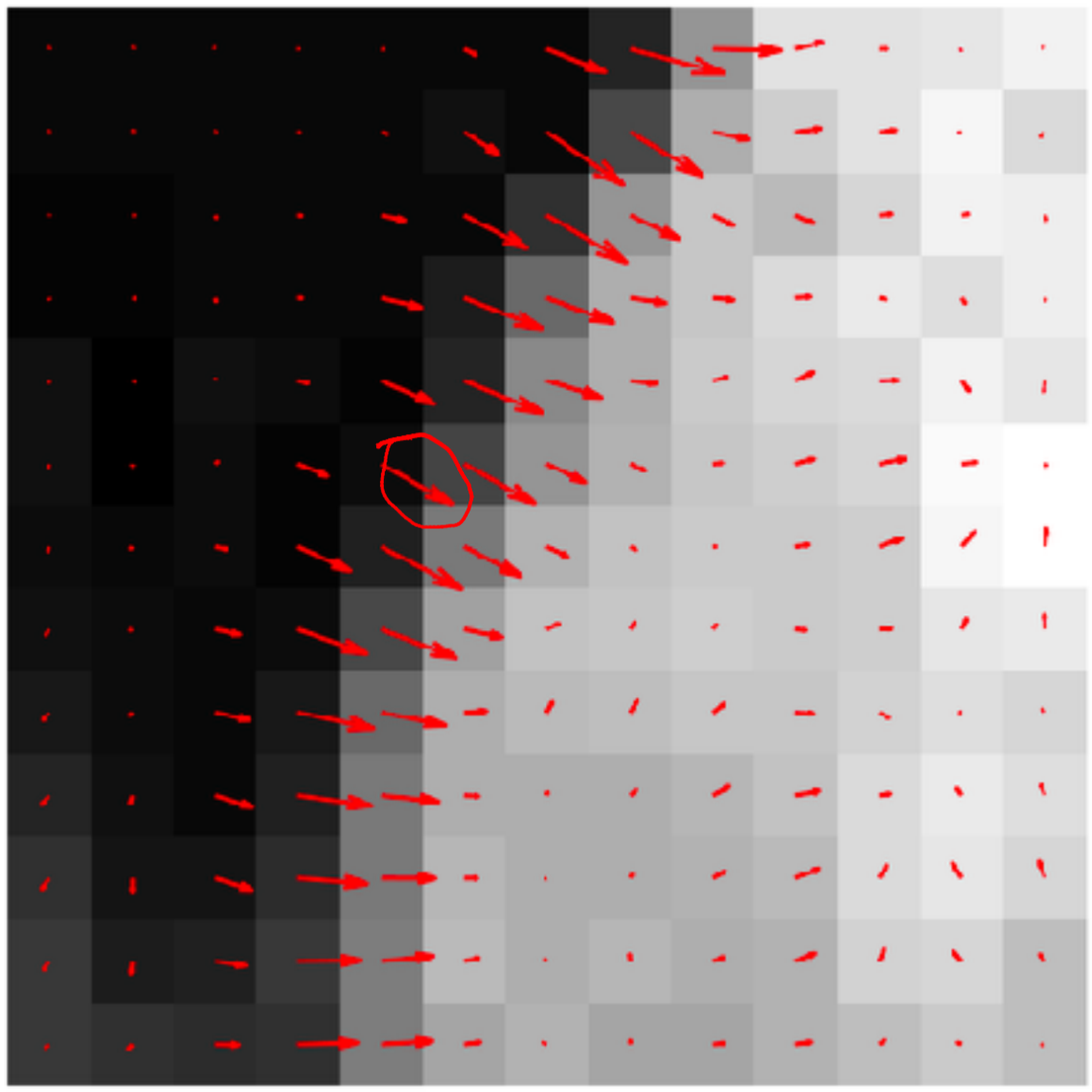
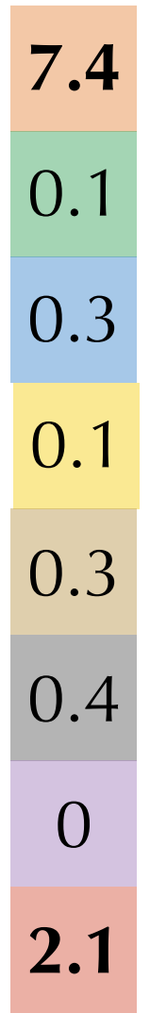
SSY097 - Image Analysis

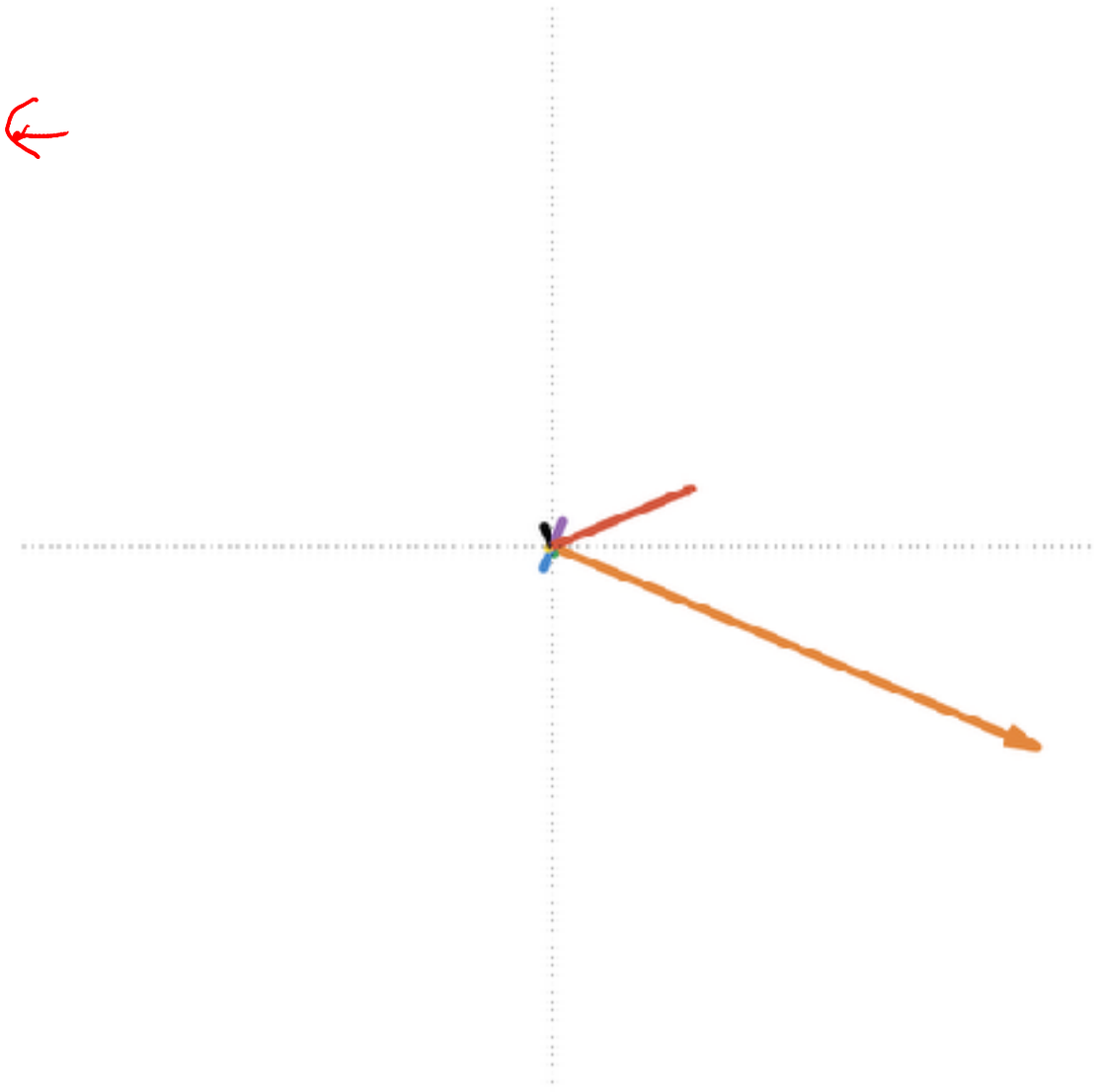
Lecture 3 - Scale-Invariant Local Features

Torsten Sattler
(slides adapted from Olof Enqvist)

Last Lecture

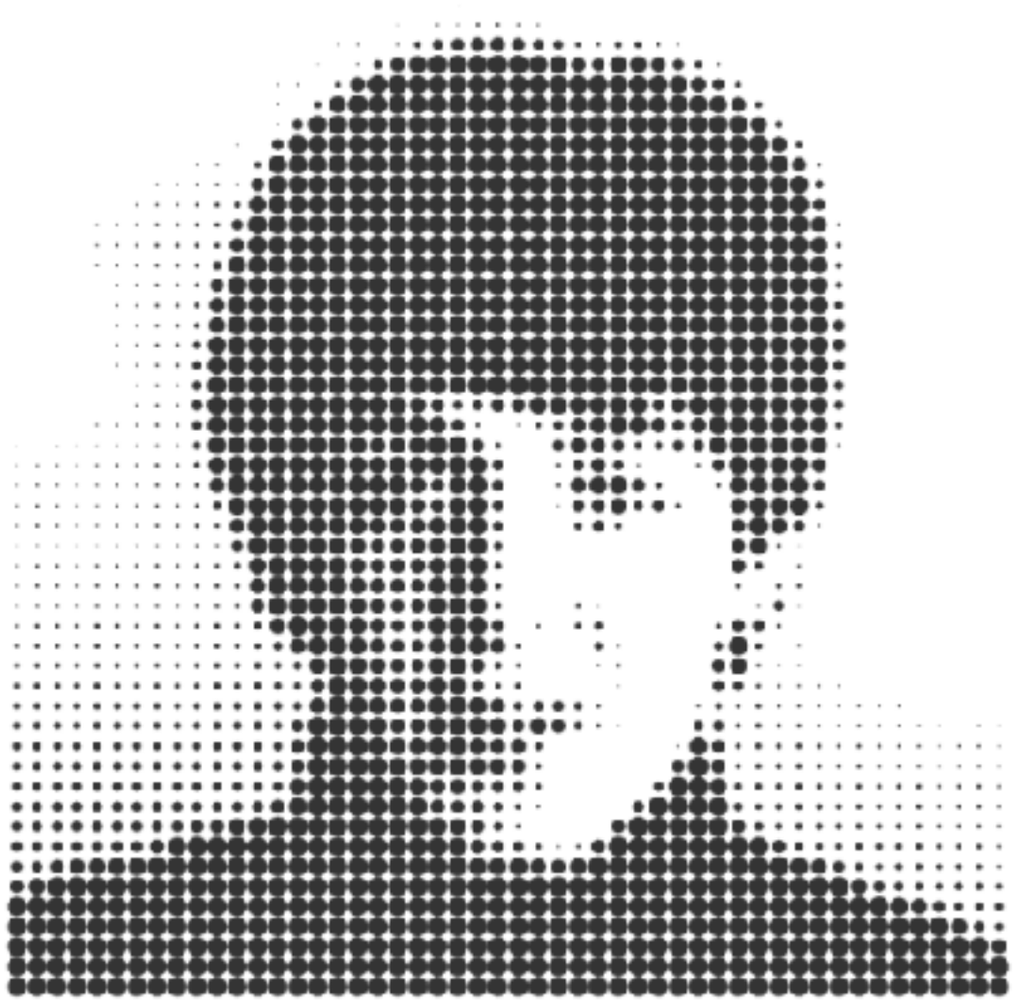






Gradient histograms

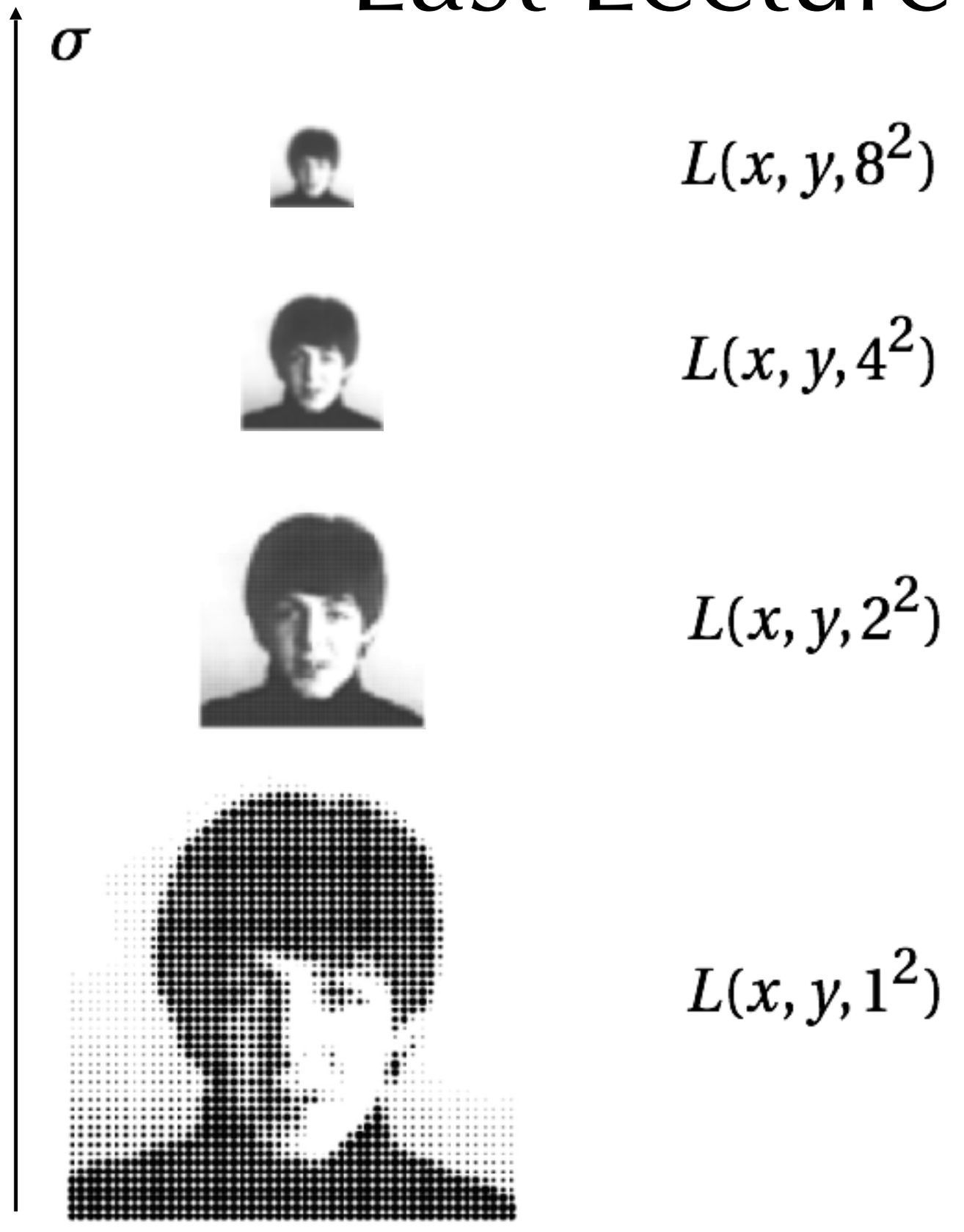
Last Lecture



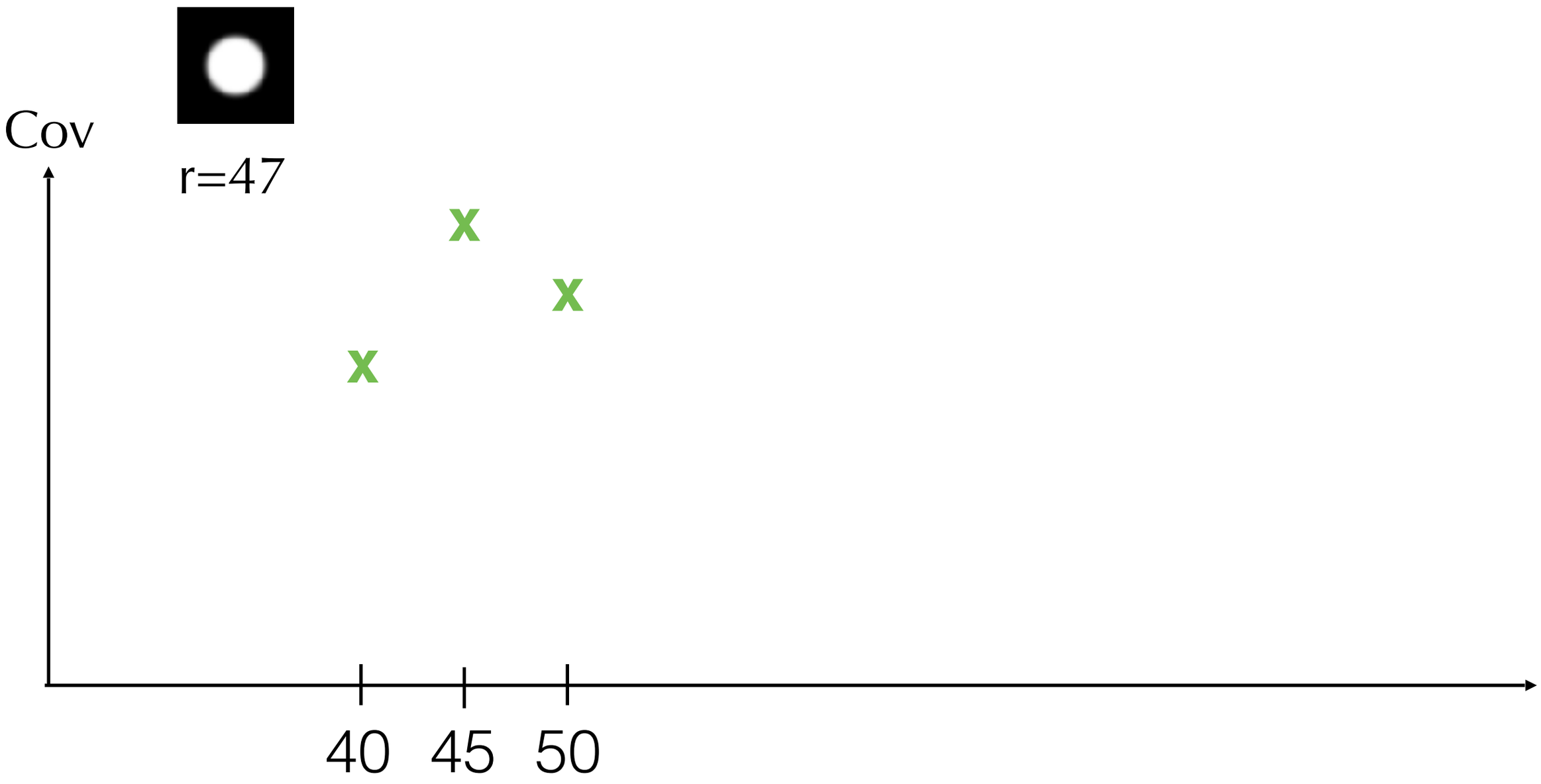
Object detection at different scales

Last Lecture

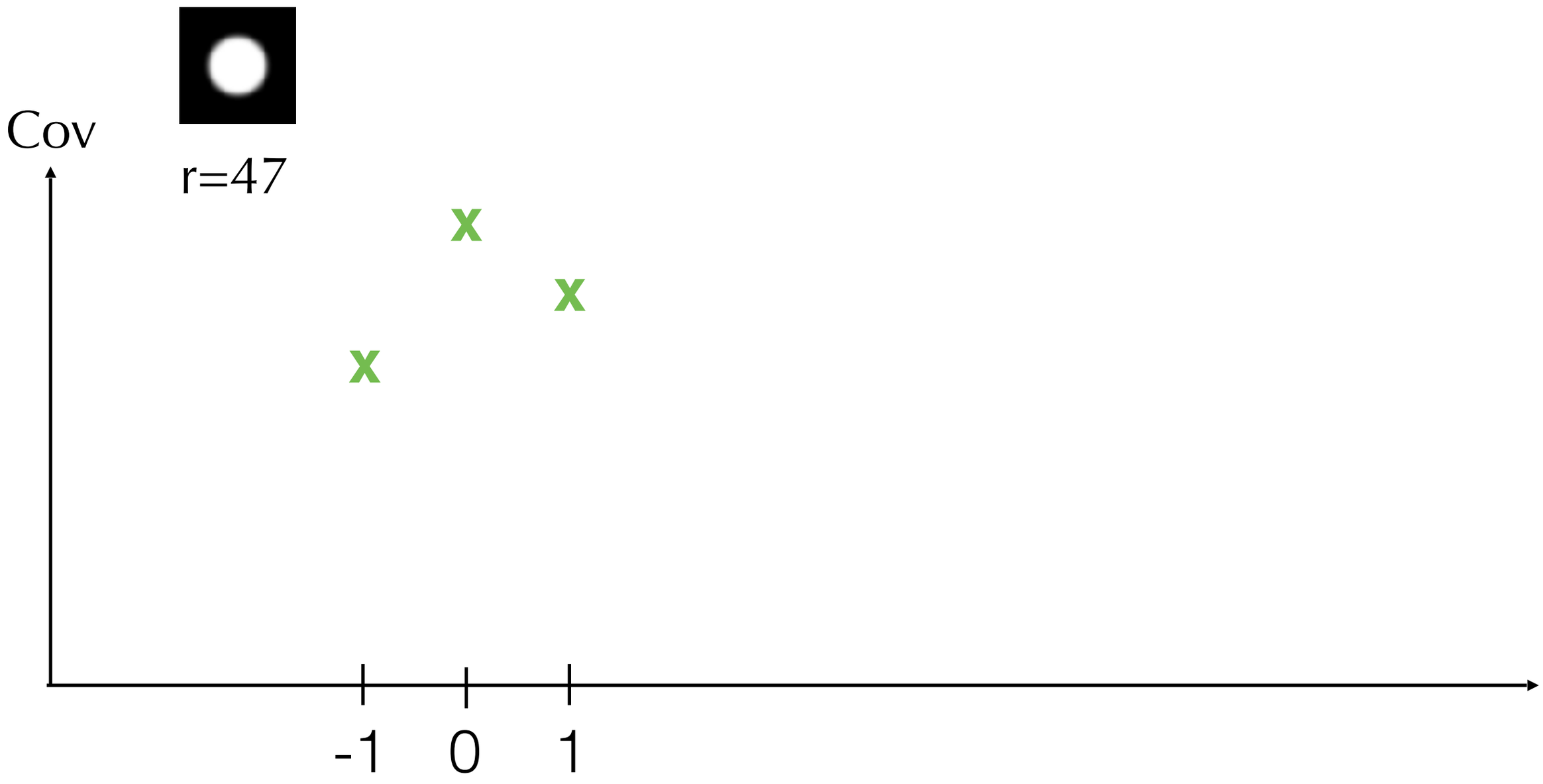
Scale space



Last Lecture



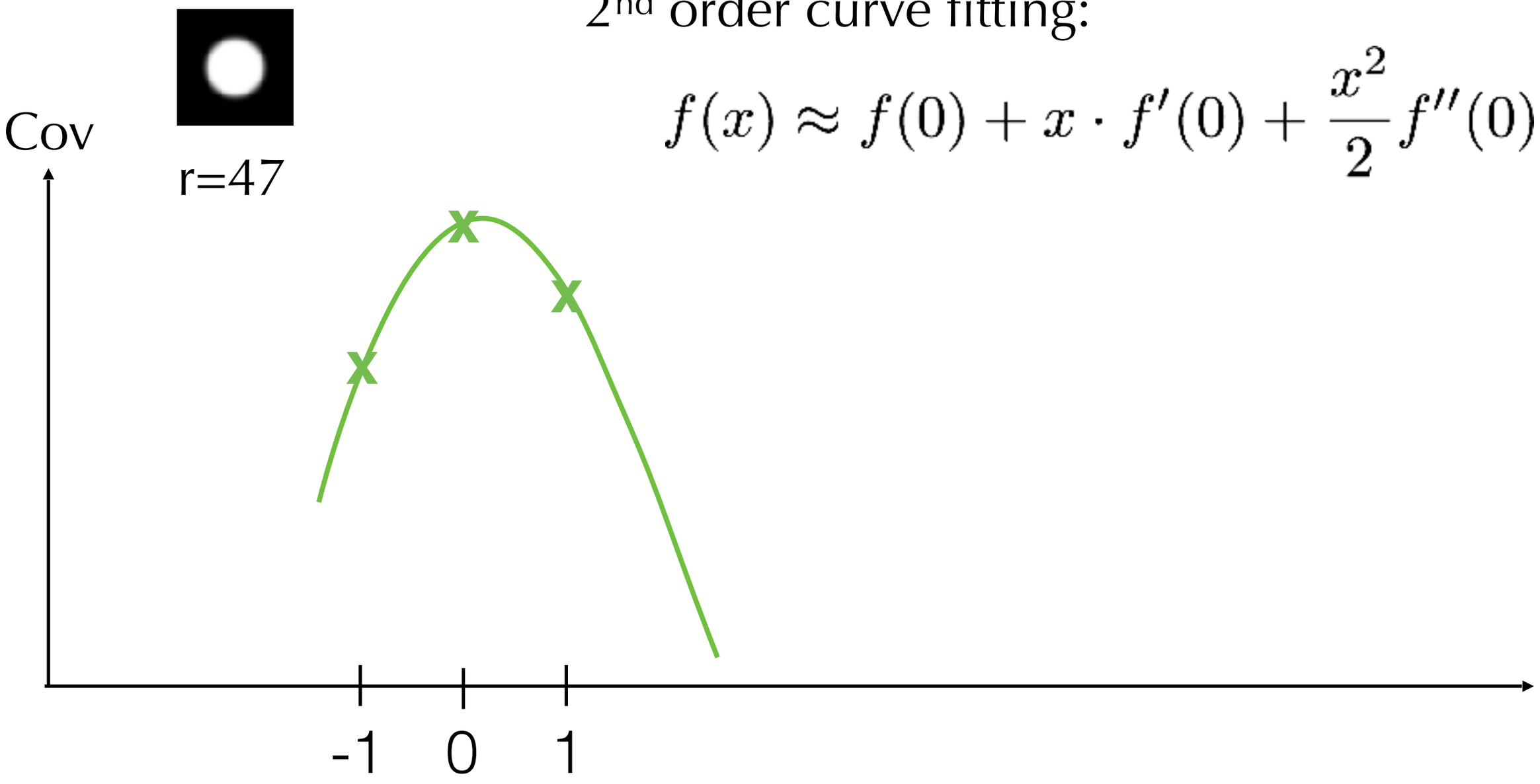
Last Lecture



Last Lecture

2nd order curve fitting:

$$f(x) \approx f(0) + x \cdot f'(0) + \frac{x^2}{2} f''(0)$$

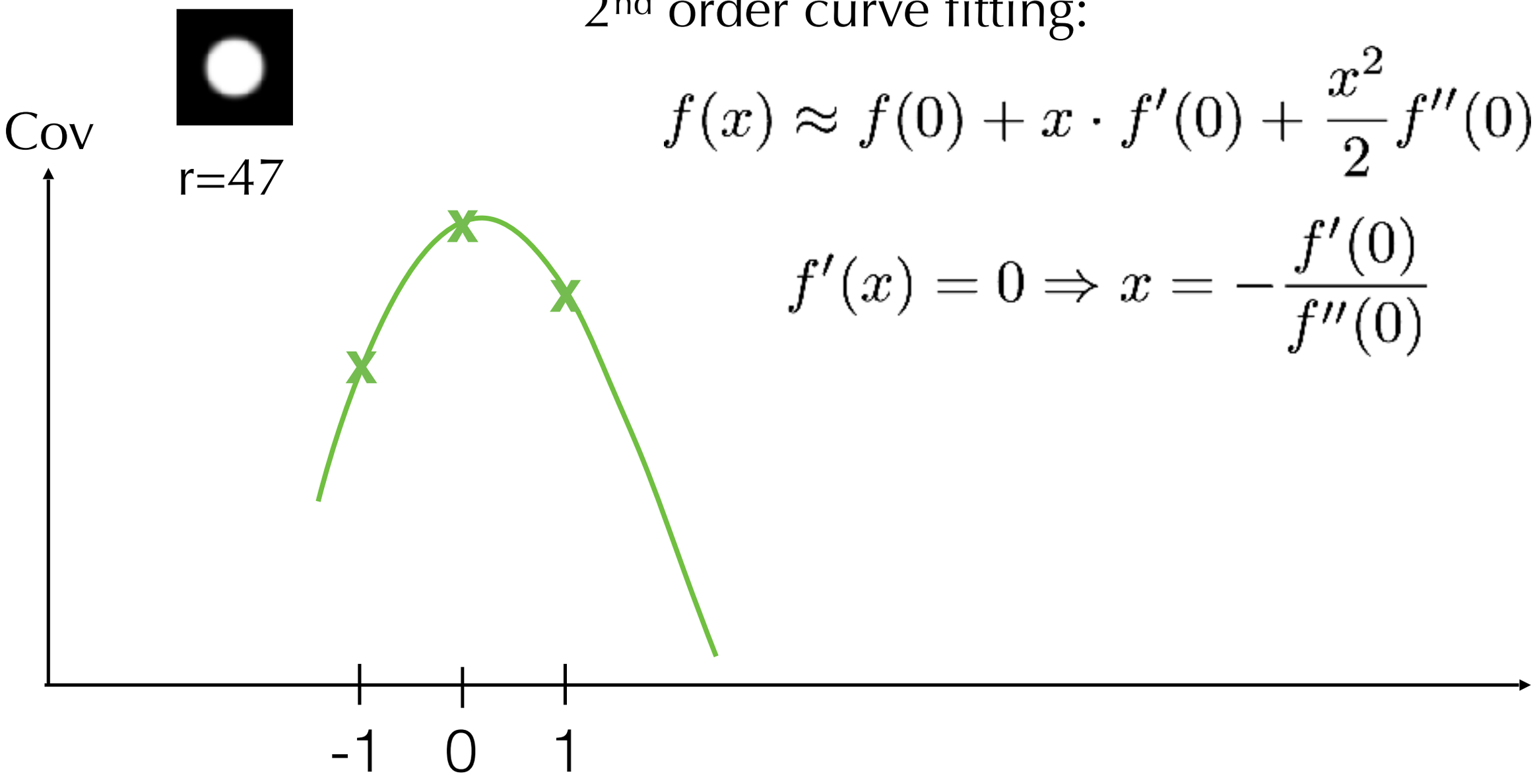


Last Lecture

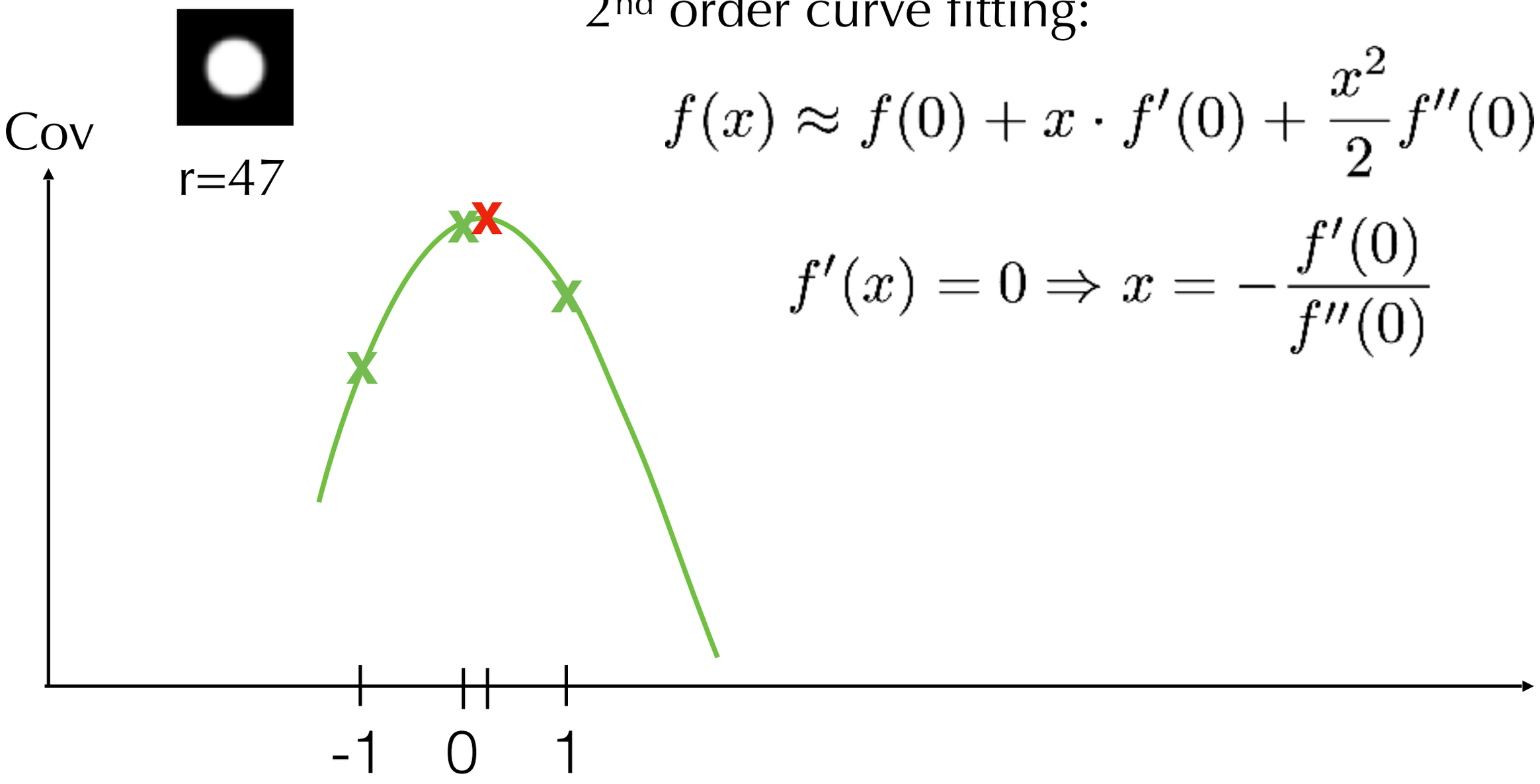
2nd order curve fitting:

$$f(x) \approx f(0) + x \cdot f'(0) + \frac{x^2}{2} f''(0)$$

$$f'(x) = 0 \Rightarrow x = -\frac{f'(0)}{f''(0)}$$



Last Lecture



Scale refinement

2nd order curve fitting:

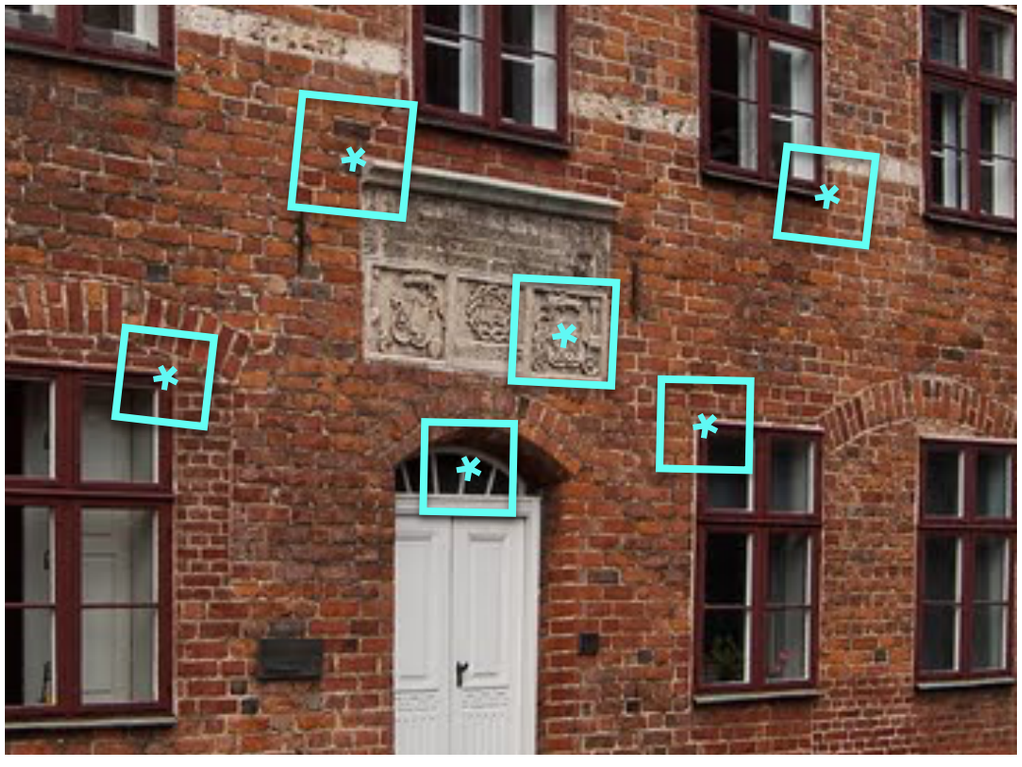
$$f(x) \approx f(0) + x \cdot f'(0) + \frac{x^2}{2} f''(0)$$

$$f'(x) = 0 \Rightarrow x = -\frac{f'(0)}{f''(0)}$$

Today

- Scale-Invariant Local Features
- Invariances*

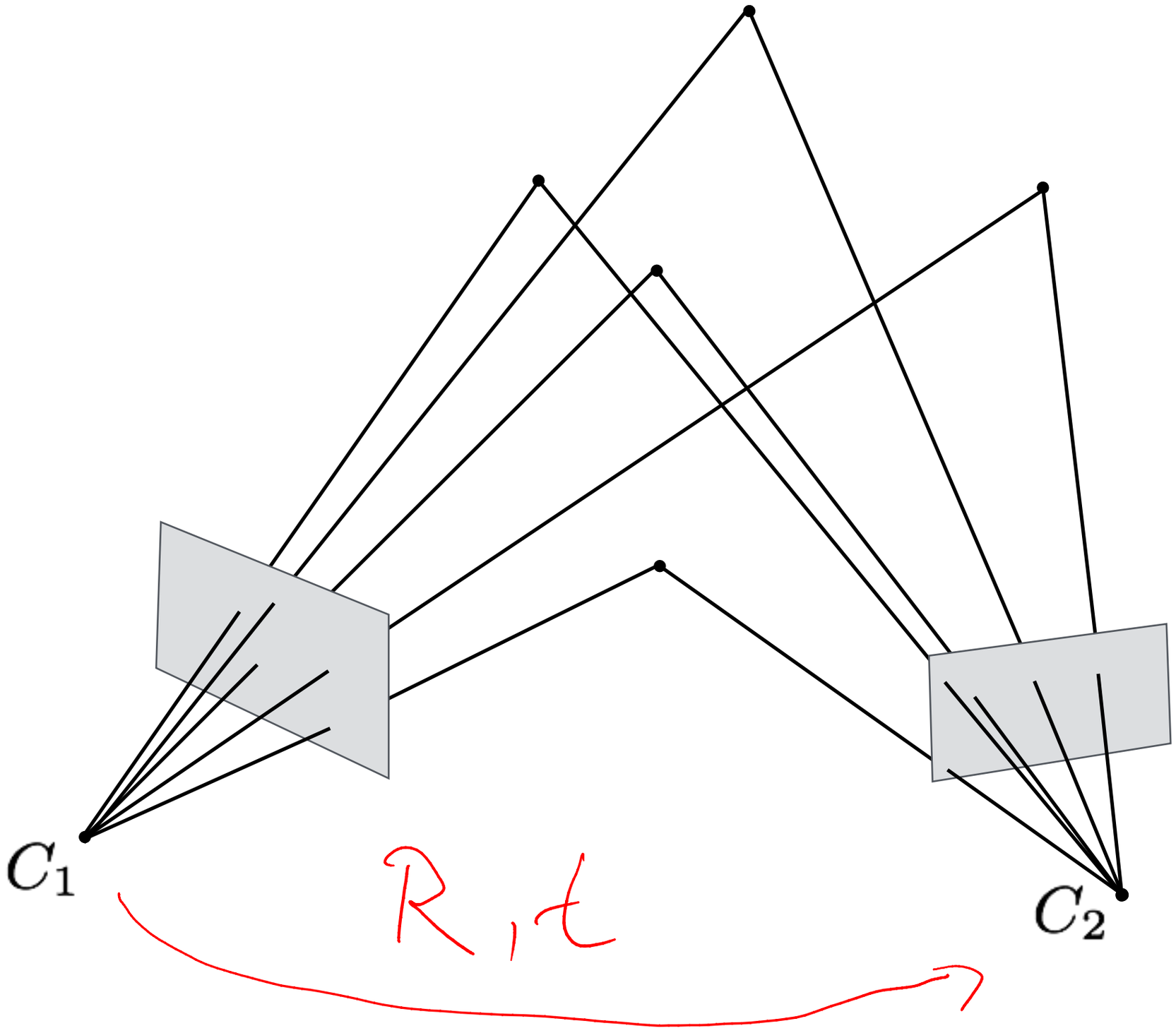
Why?



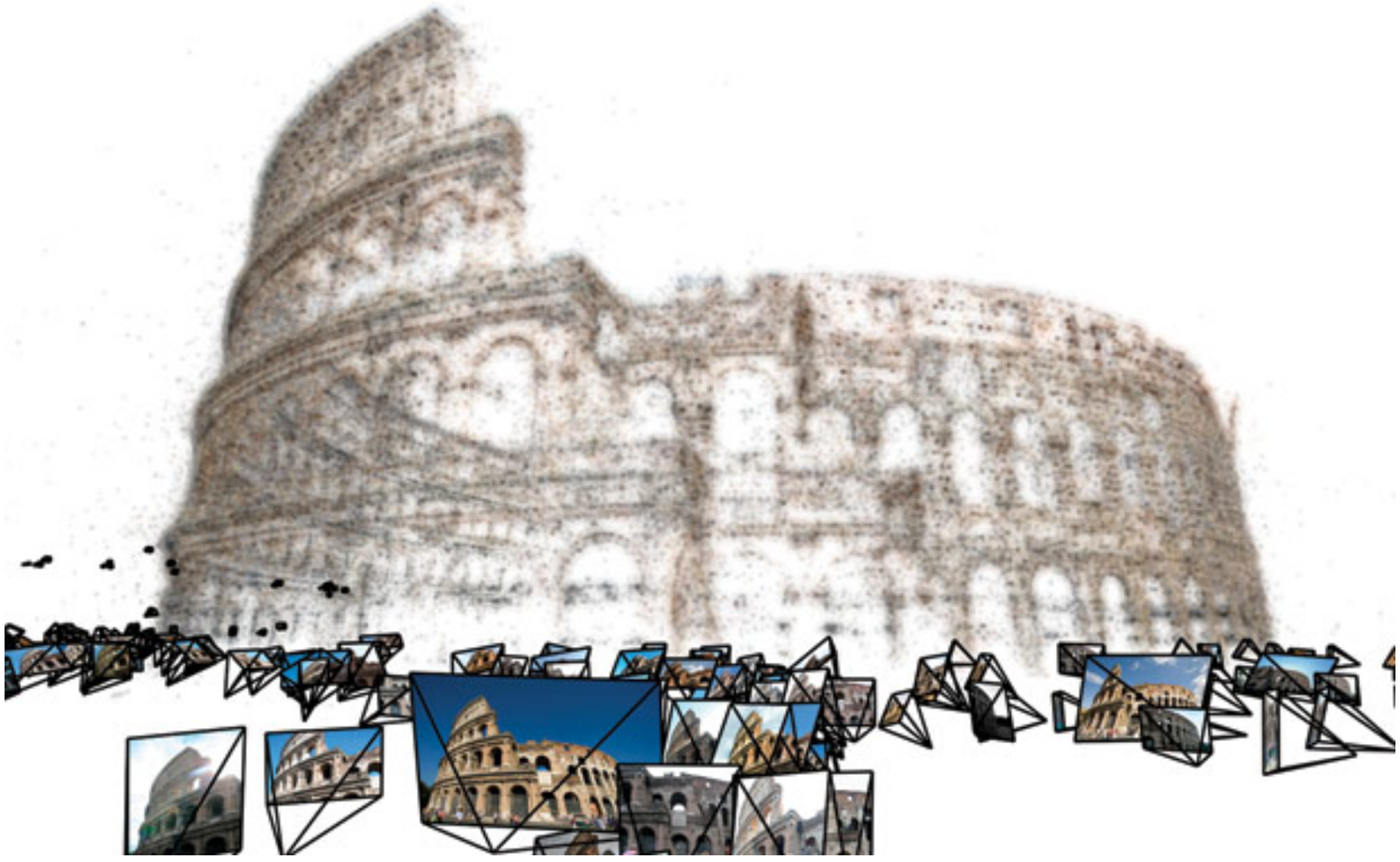
Patch matching

Applications: Image Stitching

Relative Pose Estimation



Applications: 3D Reconstruction



Snavely et al.

Scale-Invariant Feature Transform (SIFT) Features

[Lowe, Distinctive Image Features from Scale-Invariant Keypoints, IJCV 2004]

Scale-Invariant Feature Transform (SIFT) Features

- Find a set of *interest* points

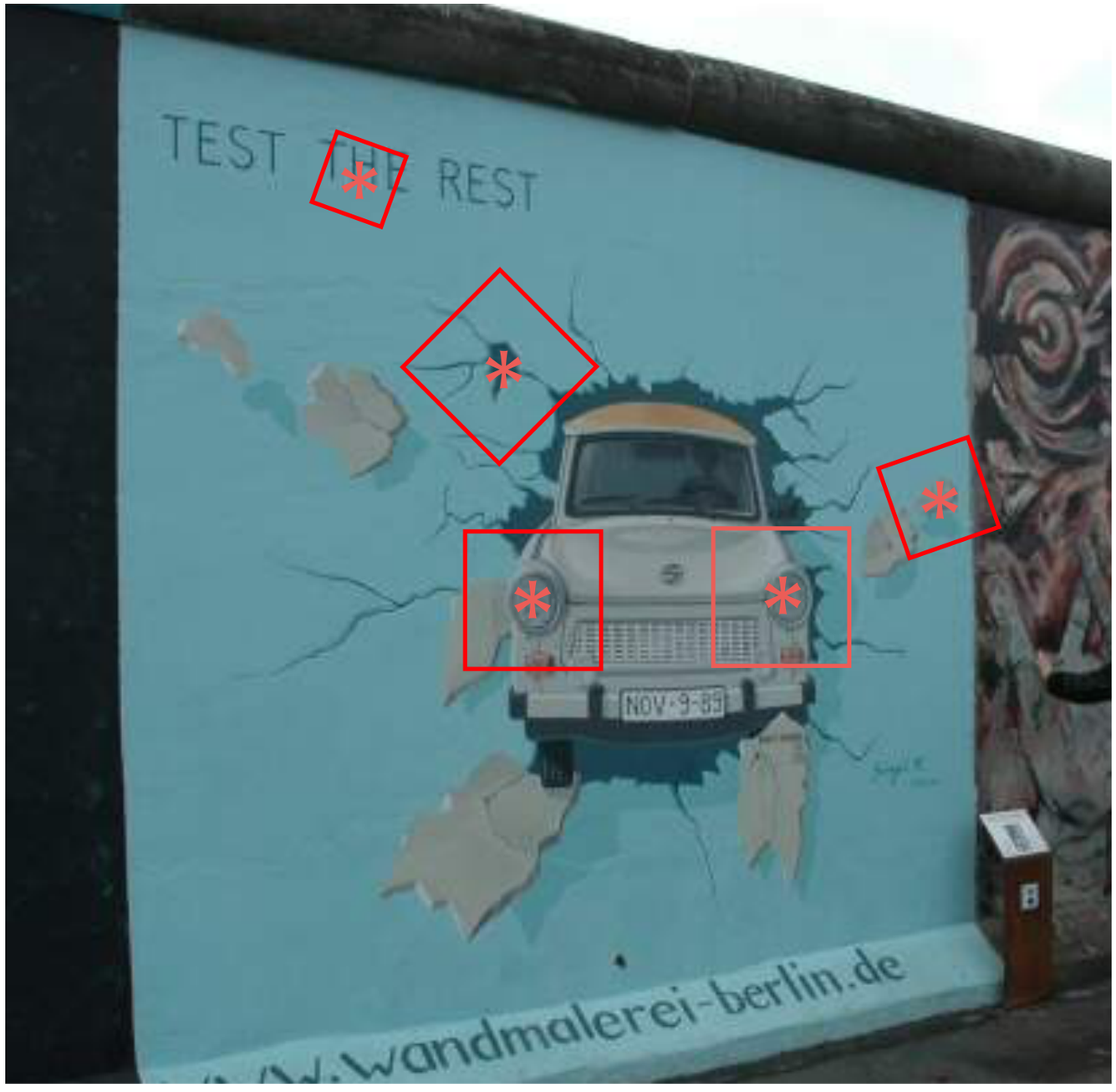
[Lowe, Distinctive Image Features from Scale-Invariant Keypoints, IJCV 2004]

Scale-Invariant Feature Transform (SIFT) Features

- Find a set of *interest* points
- Extract oriented patch around each interest point

[Lowe, Distinctive Image Features from Scale-Invariant Keypoints, IJCV 2004]

Scale-Invariant Feature Transform (SIFT) Features



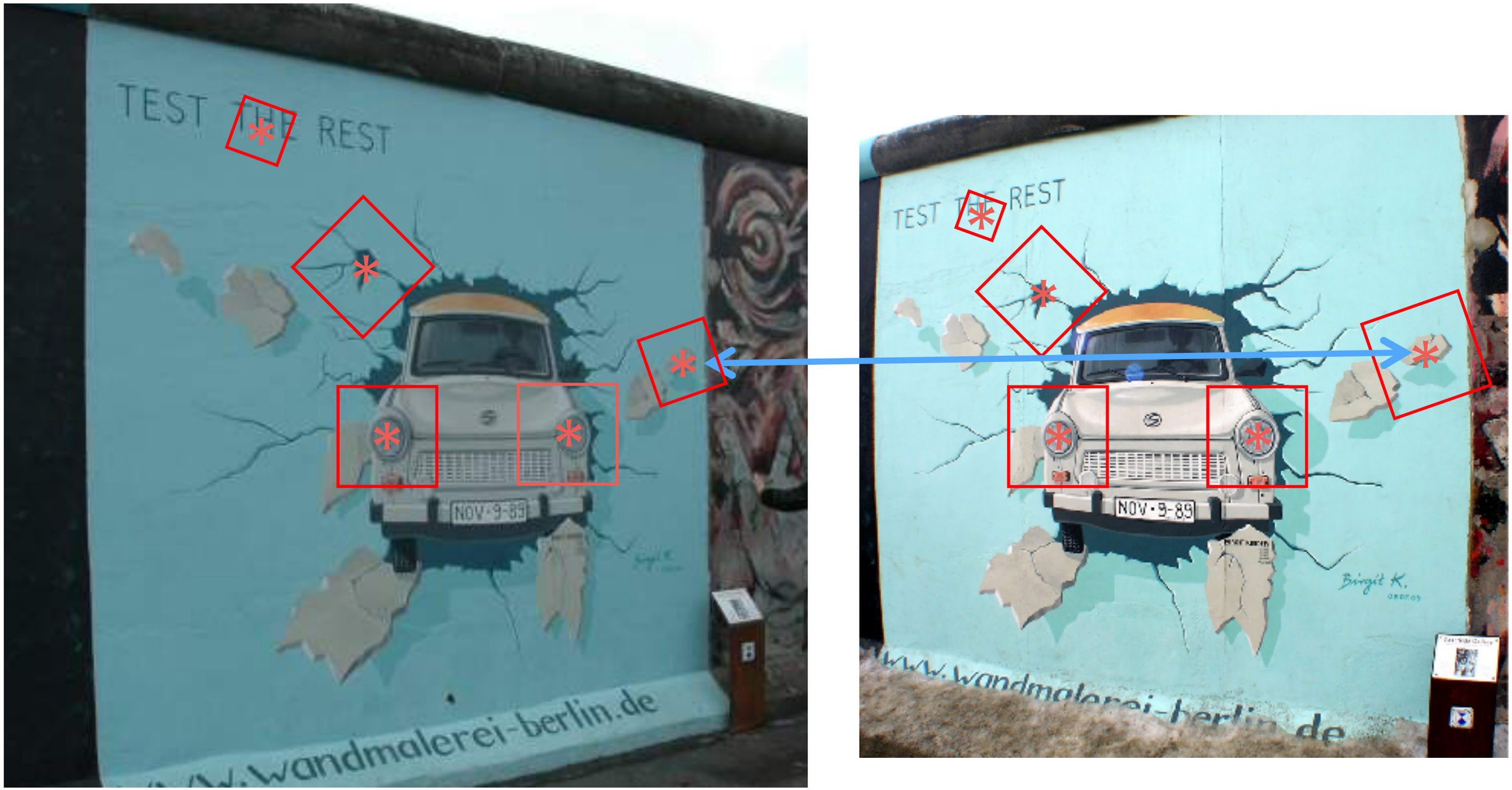
- Find a set of *interest* points
- Extract oriented patch around each interest point
- Compute SIFT descriptor for each patch

[Lowe, Distinctive Image Features from Scale-Invariant Keypoints, IJCV 2004]

Scale-Invariant Feature Transform (SIFT) Features

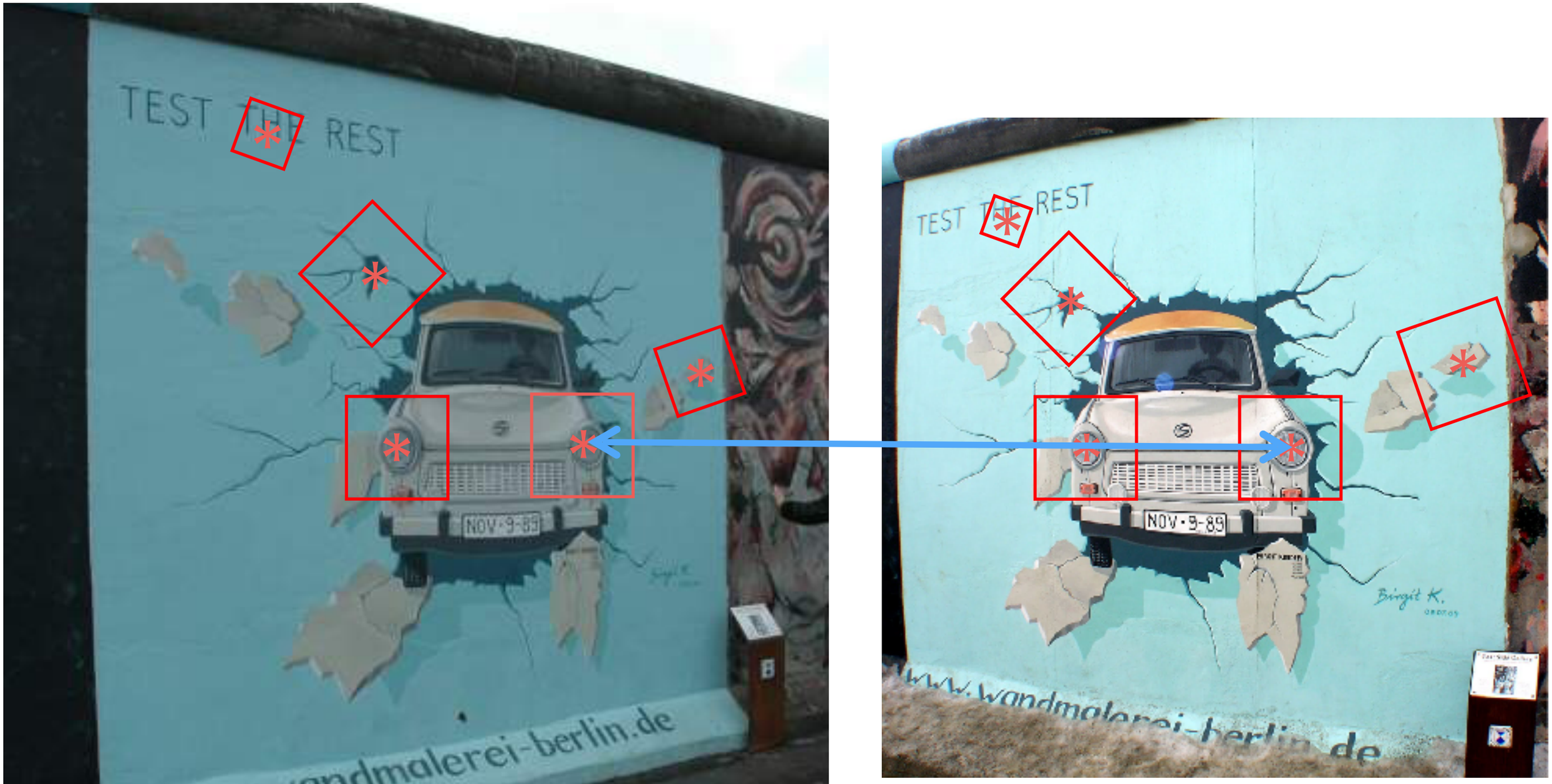
Use descriptor similarity (L_2 distance) to find corresponding points in different images

Scale-Invariant Feature Transform (SIFT) Features



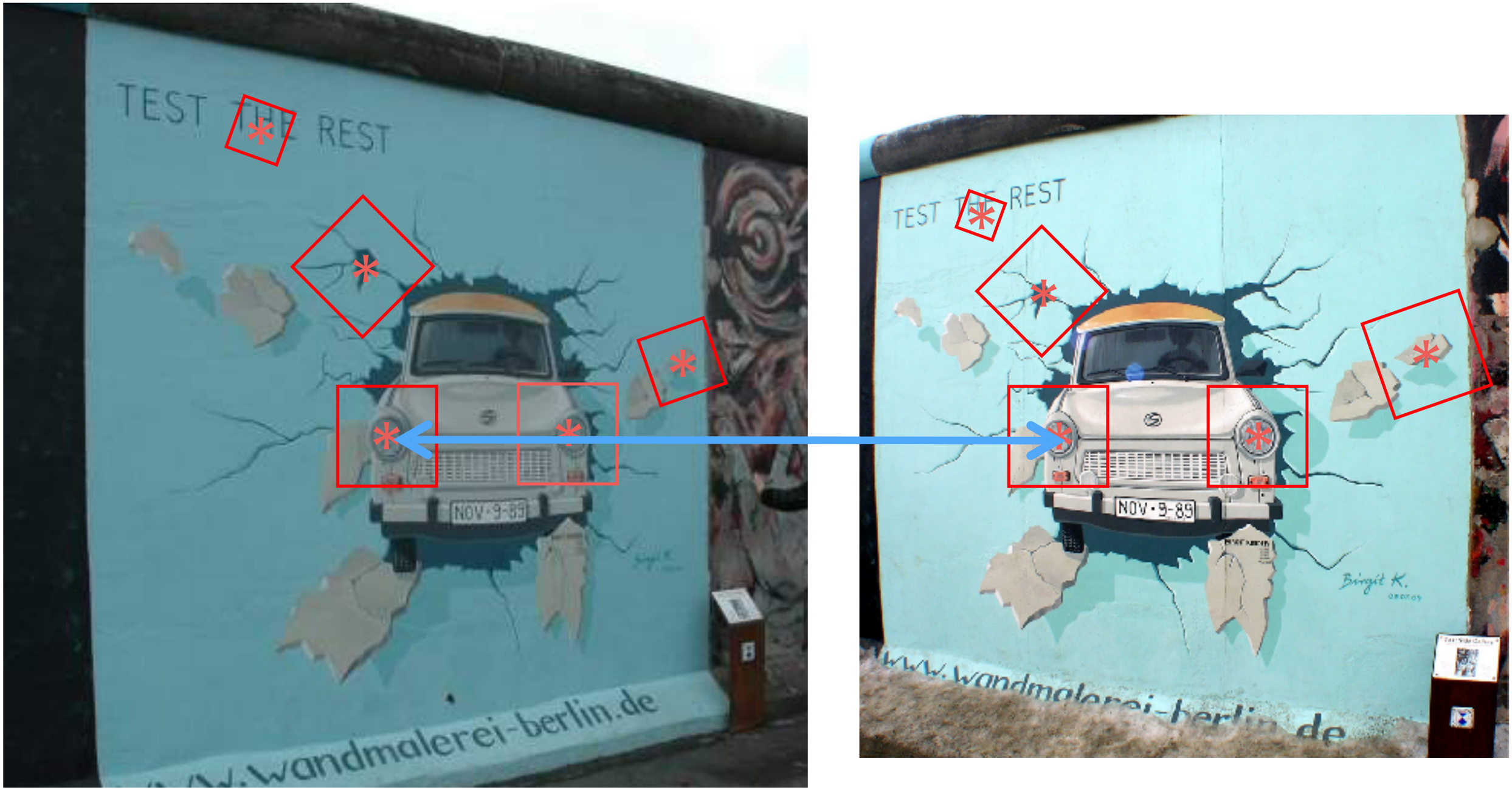
Use descriptor similarity (L_2 distance) to find corresponding points in different images

Scale-Invariant Feature Transform (SIFT) Features



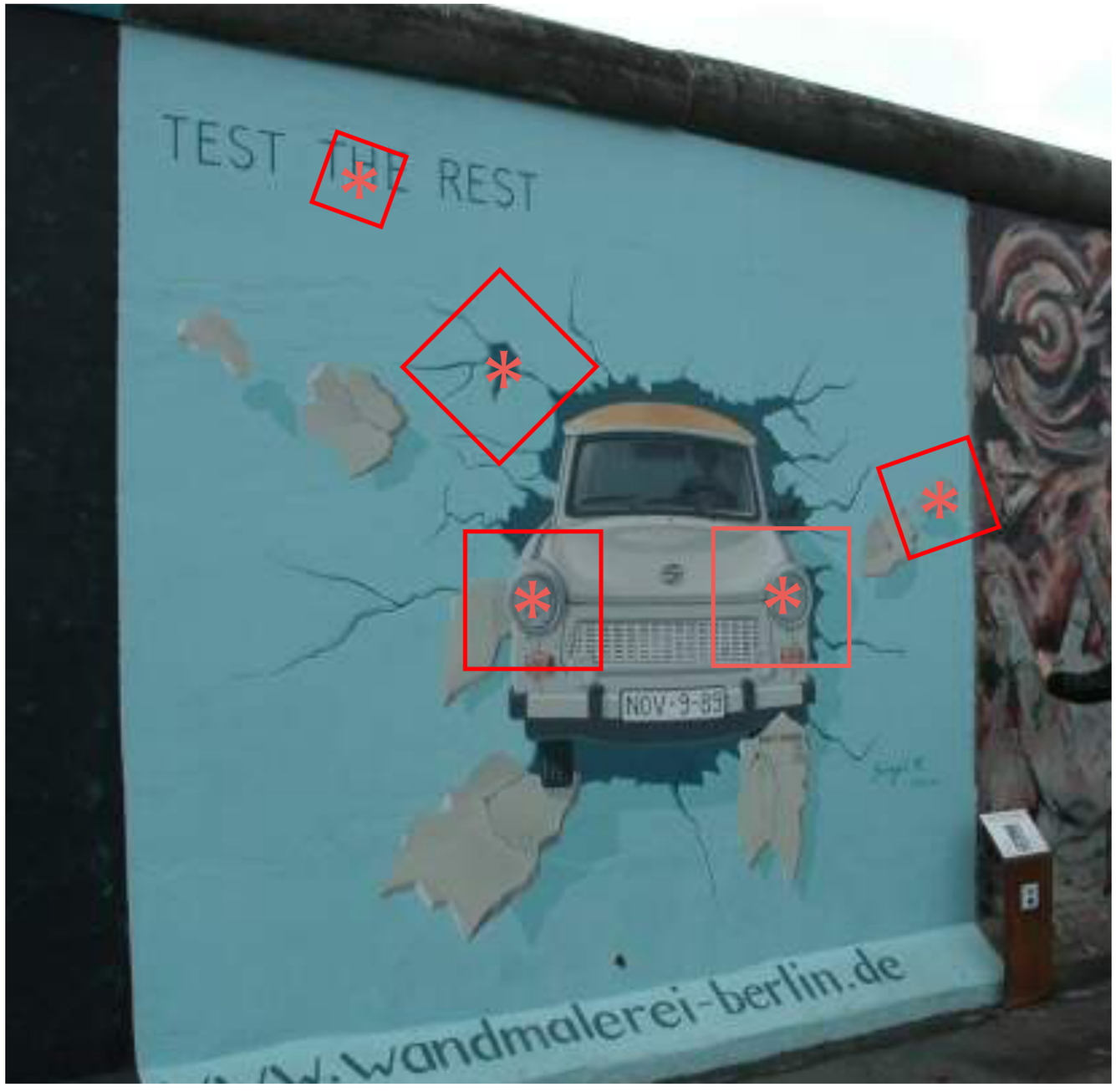
Use descriptor similarity (L_2 distance) to find corresponding points in different images

Scale-Invariant Feature Transform (SIFT) Features



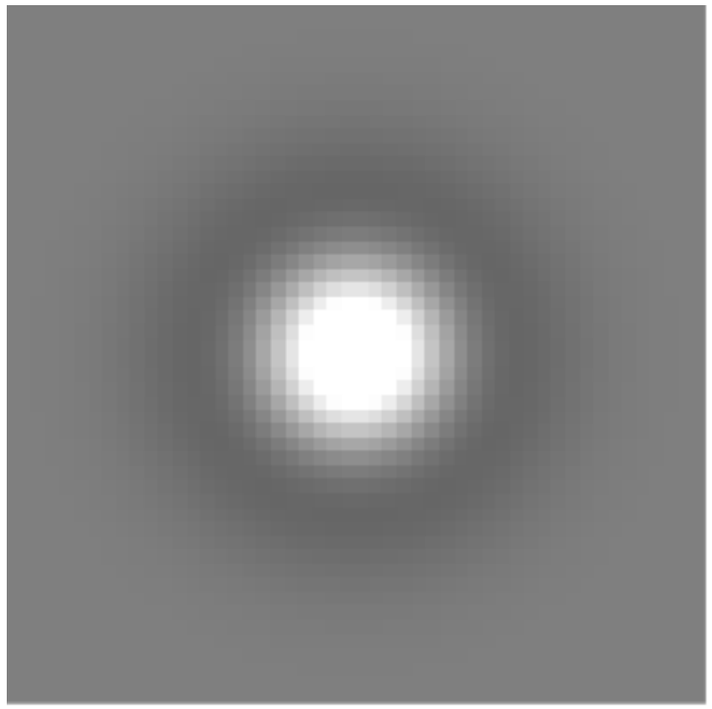
Use descriptor similarity (L_2 distance) to find corresponding points in different images

Scale-Invariant Feature Transform (SIFT) Features

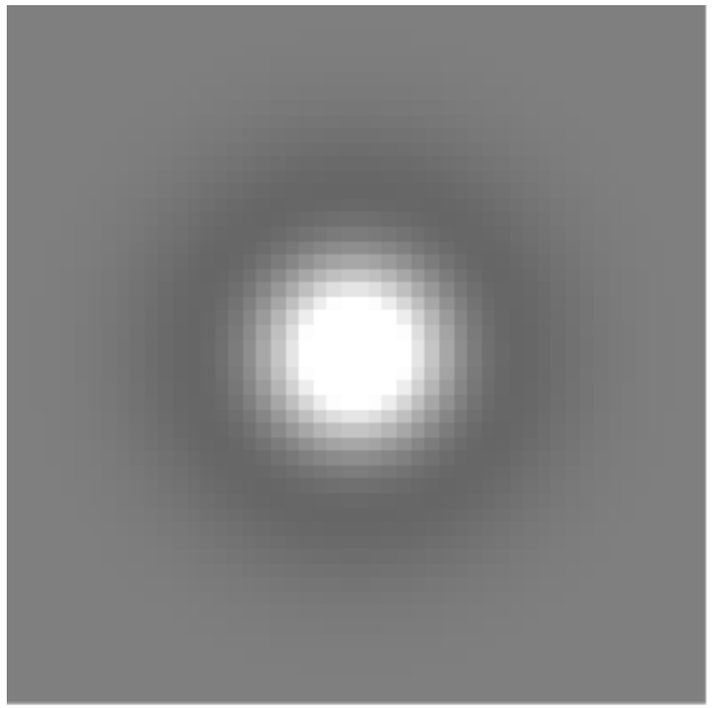


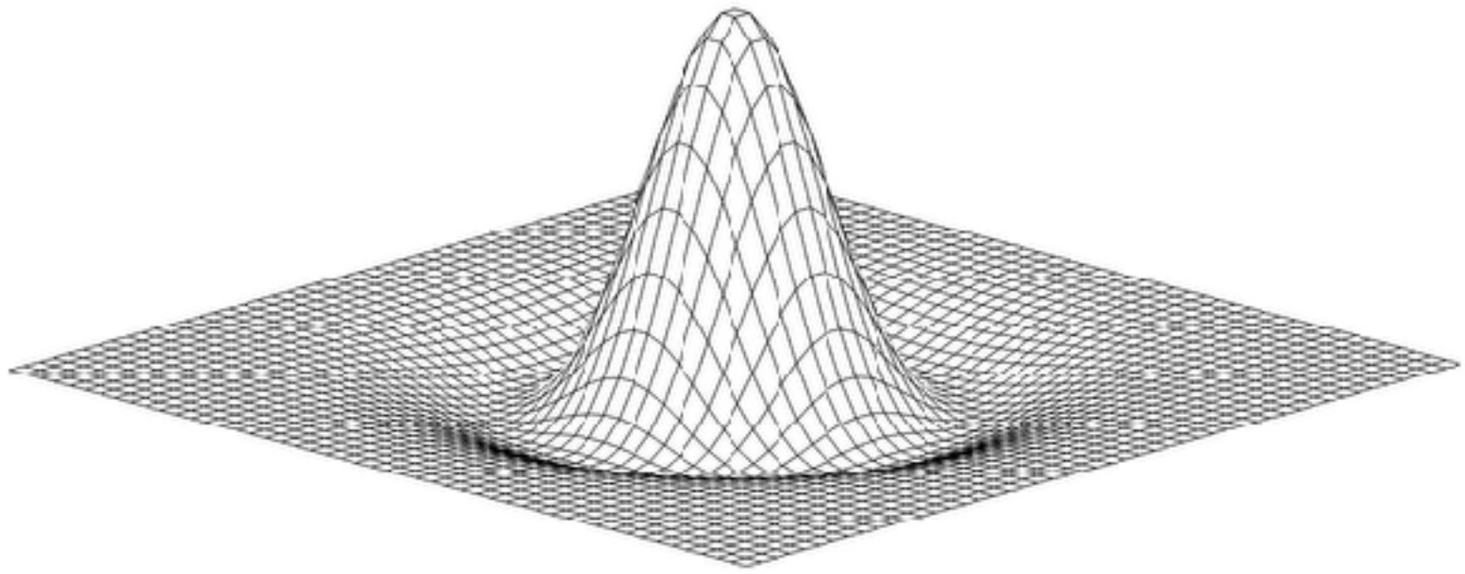
- **Find a set of *interest* points**
- Extract oriented patch around each interest point
- Compute SIFT descriptor for each patch

Difference of Gaussians (DoG)



Difference of Gaussians (DoG)





Difference of two Gaussian filters

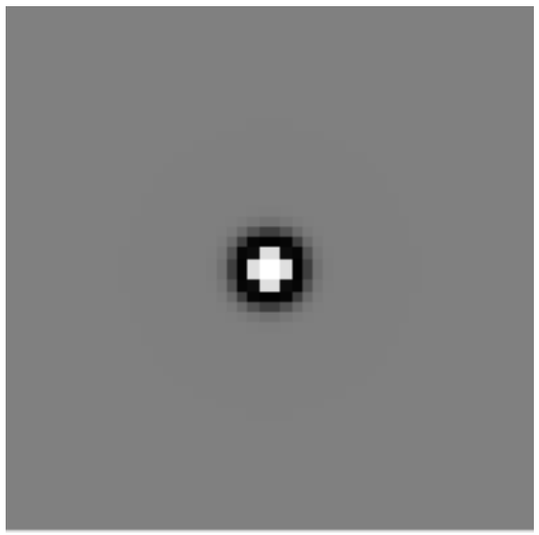
- Local maxima at bright blobs
- Local minima at dark blobs

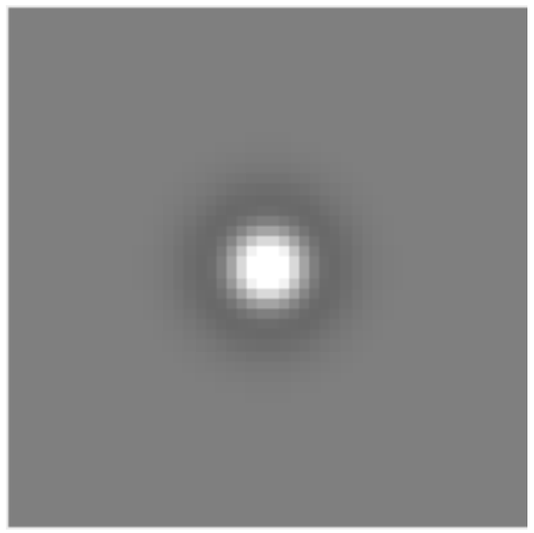
The Need for Multi-Scale Detections

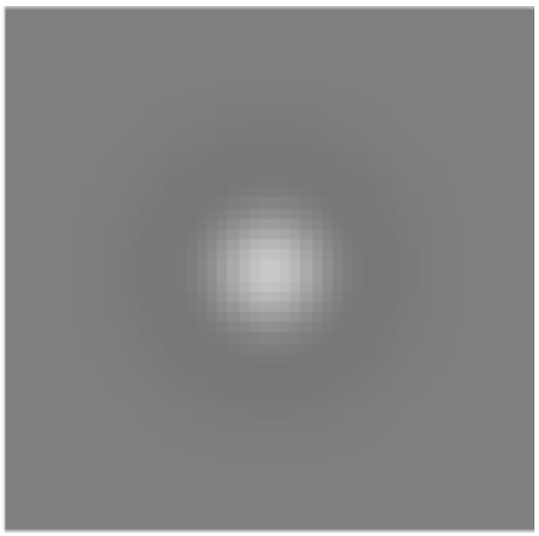
The Need for Multi-Scale Detections

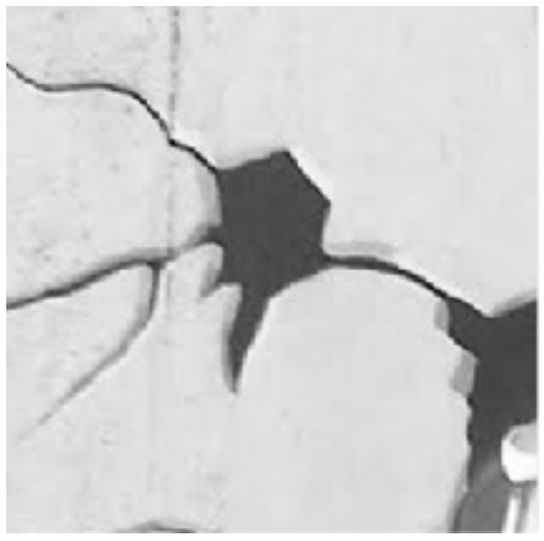
Difference of Gaussians (DoG)

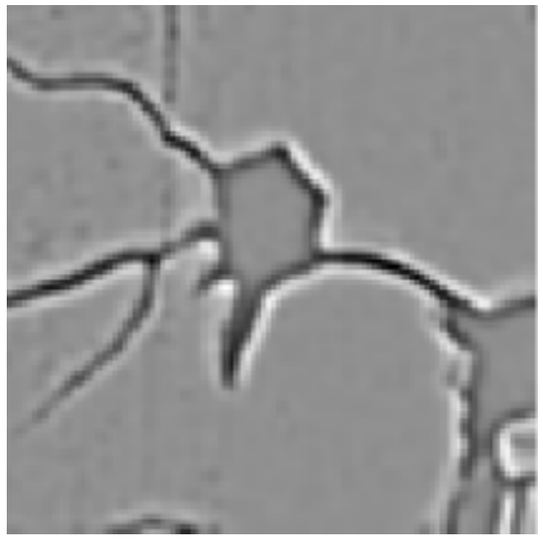
Use filters of different size

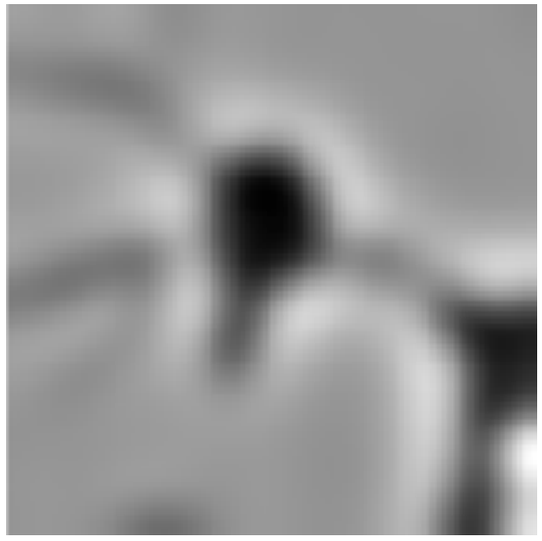


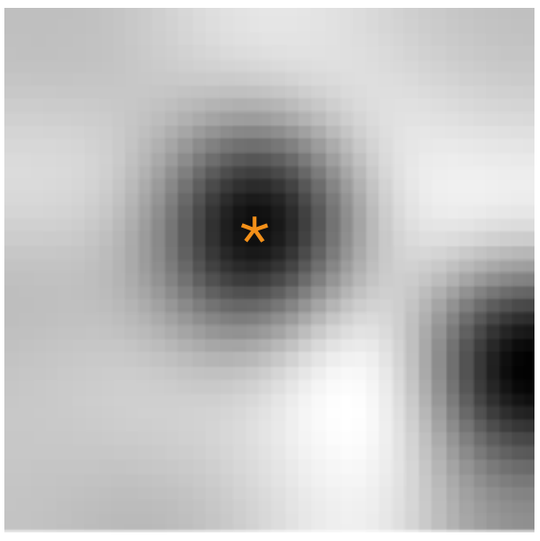




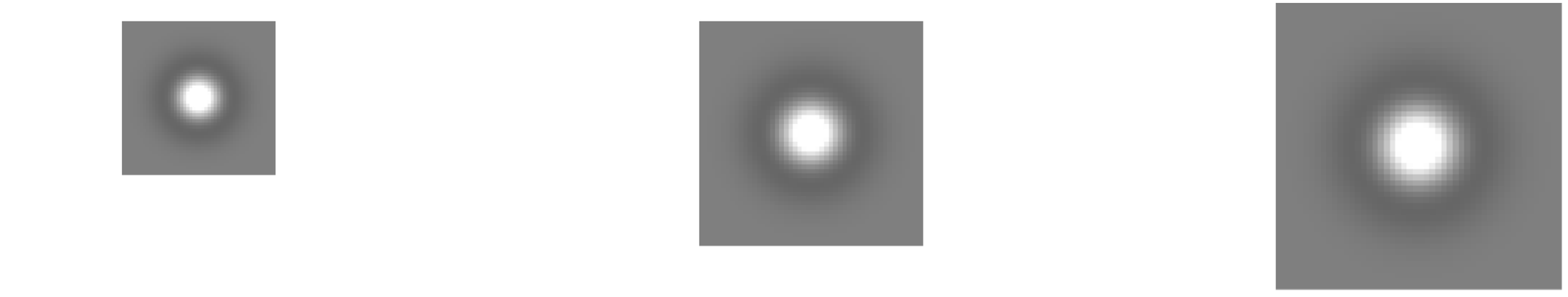




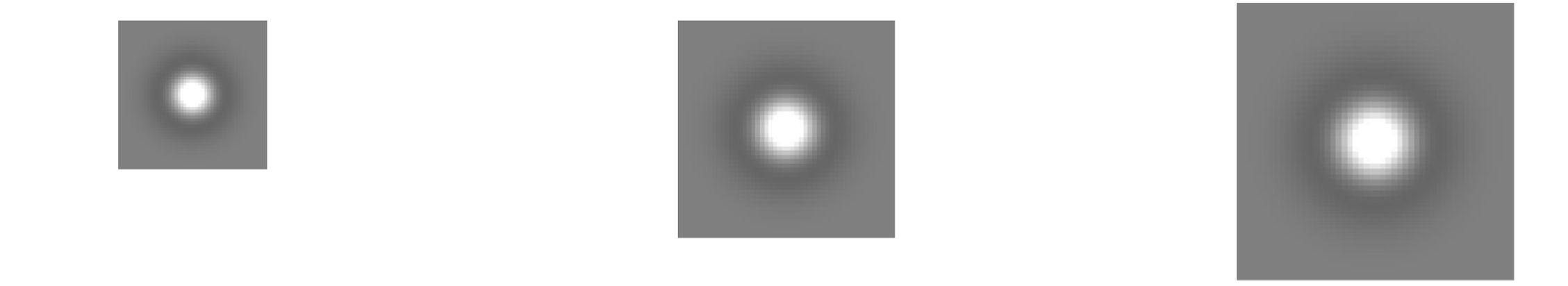




Detect Scale and Position



Detect Scale and Position

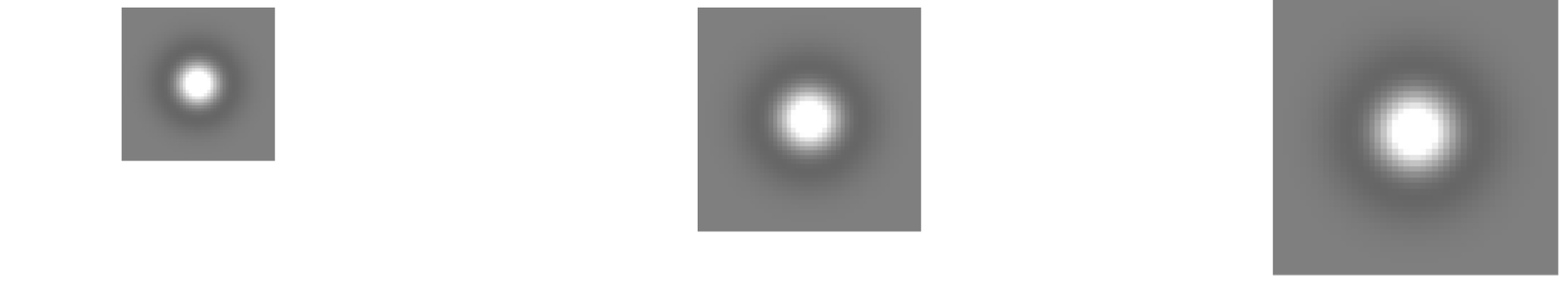


1	0	1	2	3	4	2
2	7	8	2	3	3	3
2	3	1	4	5	6	2
2	7	12	15	20	7	2
2	8	12	25	19	11	3
3	9	11	18	16	7	1
1	2	3	2	1	3	1

2	1	2	2	2	3	3
1	2	3	2	2	3	4
5	11	9	8	9	8	1
1	11	15	20	22	9	5
2	12	16	35	22	11	4
3	9	15	18	21	7	1
2	2	9	8	7	9	1

3	3	3	3	3	3	3
3	4	4	4	4	4	3
3	5	6	6	6	5	2
2	7	12	12	12	7	2
2	5	12	20	13	11	3
3	9	12	13	13	9	1
1	2	5	5	5	3	1

Detect Scale and Position



1	0	1	2	3	4	2
2	7	8	2	3	3	3
2	3	1	4	5	6	2
2	7	12	15	20	7	2
2	8	12	25	19	11	3
3	9	11	18	16	7	1
1	2	3	2	1	3	1

2	1	2	2	2	3	3
1	2	3	2	2	3	4
5	11	9	8	9	8	1
1	11	15	20	22	9	5
2	12	16	35	22	11	4
3	9	15	18	21	7	1
2	2	9	8	7	9	1

3	3	3	3	3	3	3
3	4	4	4	4	4	3
3	5	6	6	6	5	2
2	7	12	12	12	7	2
2	5	12	20	13	11	3
3	9	12	13	13	9	1
1	2	5	5	5	3	1

Non-maximum suppression in image **and scale**!

Sub-Pixel Refinement

2nd order curve fitting (1D):

$$f(x) \approx f(0) + x \cdot f'(0) + \frac{x^2}{2} f''(0)$$

Sub-Pixel Refinement

2nd order curve fitting (1D):

$$f(x) \approx f(0) + x \cdot f'(0) + \frac{x^2}{2} f''(0)$$

2nd order surface fitting (3D):

$$f(x, y, \sigma) \approx f(0, 0, 0) + \begin{pmatrix} x & y & \sigma \end{pmatrix} \cdot \nabla f(0, 0, 0) + \frac{1}{2} \begin{pmatrix} x & y & \sigma \end{pmatrix} \mathrm{H}(0, 0, 0) \begin{pmatrix} x \\ y \\ \sigma \end{pmatrix}$$

Sub-Pixel Refinement

2nd order curve fitting (1D):

$$f(x) \approx f(0) + x \cdot f'(0) + \frac{x^2}{2} f''(0)$$

2nd order surface fitting (3D):

$$f(x, y, \sigma) \approx f(0,0,0) + \begin{pmatrix} x & y & \sigma \end{pmatrix} \cdot \nabla f(0,0,0) + \frac{1}{2} \begin{pmatrix} x & y & \sigma \end{pmatrix} \mathrm{H}(0,0,0) \begin{pmatrix} x \\ y \\ \sigma \end{pmatrix}$$

Hessian matrix:

$$\mathrm{H}(0,0,0) = \begin{pmatrix} f''_{xx}(0,0,0) & f''_{xy}(0,0,0) & f''_{x\sigma}(0,0,0) \\ f''_{xy}(0,0,0) & f''_{yy}(0,0,0) & f''_{y\sigma}(0,0,0) \\ f''_{x\sigma}(0,0,0) & f''_{y\sigma}(0,0,0) & f''_{\sigma\sigma}(0,0,0) \end{pmatrix}$$

Sub-Pixel Refinement

2nd order surface fitting (3D):

$$f(x,y,\sigma) \approx f(0,0,0) + \begin{pmatrix} x & y & \sigma \end{pmatrix} \cdot \nabla f(0,0,0) + \frac{1}{2} \begin{pmatrix} x & y & \sigma \end{pmatrix} \mathrm{H}(0,0,0) \begin{pmatrix} x \\ y \\ \sigma \end{pmatrix}$$

Sub-Pixel Refinement

2nd order surface fitting (3D):

$$f(x,y,\sigma) \approx f(0,0,0) + \begin{pmatrix} x & y & \sigma \end{pmatrix} \cdot \nabla f(0,0,0) + \frac{1}{2} \begin{pmatrix} x & y & \sigma \end{pmatrix} \mathrm{H}(0,0,0) \begin{pmatrix} x \\ y \\ \sigma \end{pmatrix}$$

Set derivative to 0:

$$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} = \nabla f(0,0,0) + \mathrm{H}(0,0,0) \begin{pmatrix} x \\ y \\ \sigma \end{pmatrix}$$

offset

Sub-Pixel Refinement

2nd order surface fitting (3D):

$$f(x, y, \sigma) \approx f(0,0,0) + \begin{pmatrix} x & y & \sigma \end{pmatrix} \cdot \nabla f(0,0,0) + \frac{1}{2} \begin{pmatrix} x & y & \sigma \end{pmatrix} \mathrm{H}(0,0,0) \begin{pmatrix} x \\ y \\ \sigma \end{pmatrix}$$

Set derivative to 0:

$$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} = \nabla f(0,0,0) + \mathrm{H}(0,0,0) \begin{pmatrix} x \\ y \\ \sigma \end{pmatrix}$$

Solve small linear system:

$$\begin{pmatrix} x \\ y \\ \sigma \end{pmatrix} = -\mathrm{H}(0,0,0)^{-1} \nabla f(0,0,0)$$

Sub-Pixel Refinement

2nd order surface fitting (3D):

$$f(x,y,\sigma) \approx f(0,0,0) + \begin{pmatrix} x & y & \sigma \end{pmatrix} \cdot \nabla f(0,0,0) + \frac{1}{2} \begin{pmatrix} x & y & \sigma \end{pmatrix} \mathrm{H}(0,0,0) \begin{pmatrix} x \\ y \\ \sigma \end{pmatrix}$$

Set derivative to 0:

$$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} = \nabla f(0,0,0) + \mathrm{H}(0,0,0) \begin{pmatrix} x \\ y \\ \sigma \end{pmatrix}$$

Solve small linear system:

$$\begin{pmatrix} x \\ y \\ \sigma \end{pmatrix} = -\mathrm{H}(0,0,0)^{-1} \nabla f(0,0,0)$$

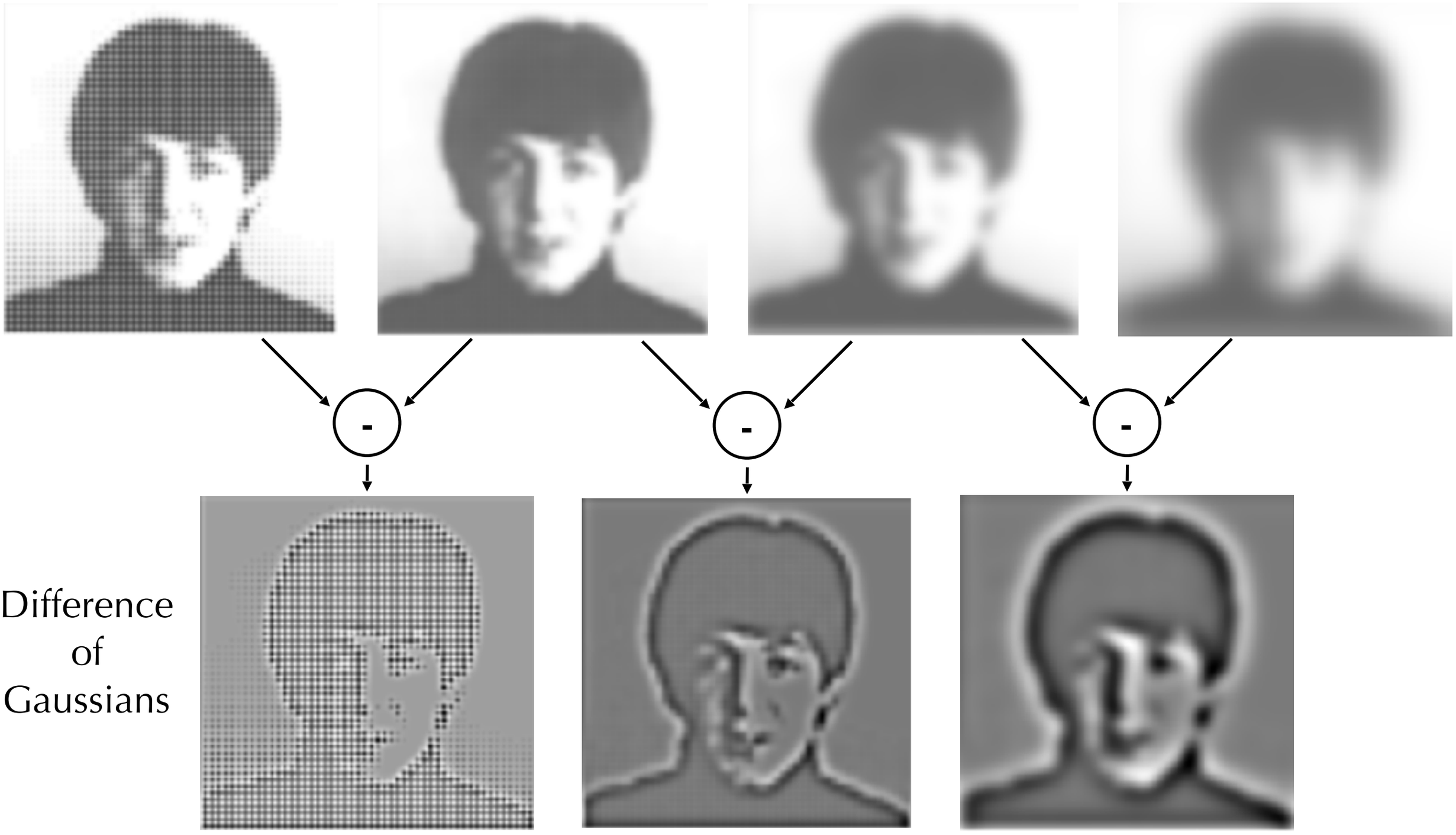
Potentially repeat for a few iterations

Efficient implementation

Gaussian filtering

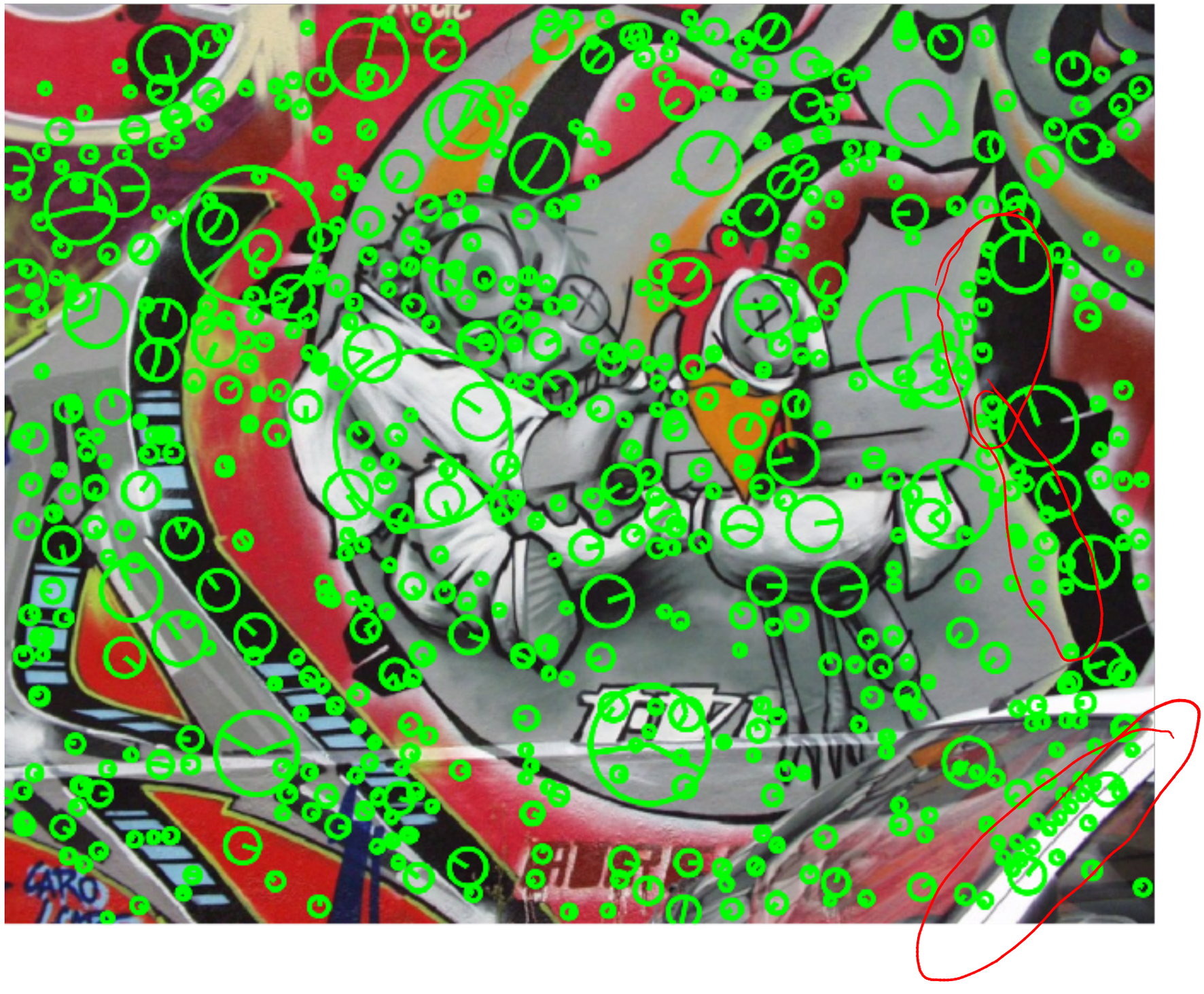
Efficient implementation

Gaussian filtering

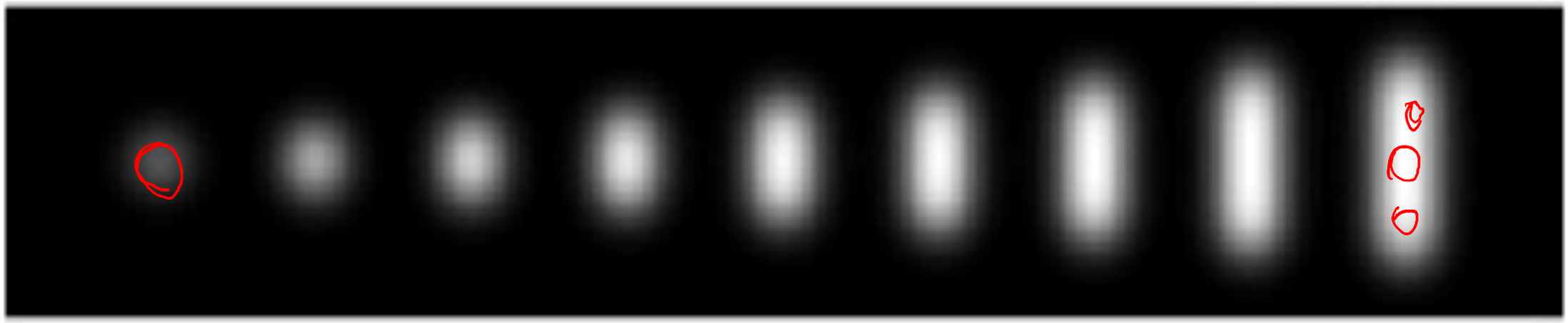


Difference of Gaussians

Avoiding Edges



Avoiding Edges

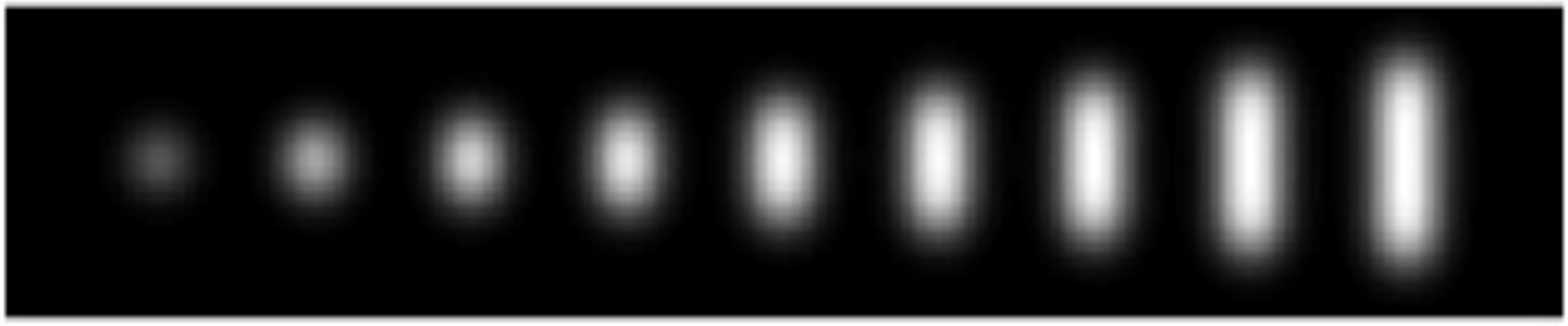


well-localized blob

edge

image source: vlfeat.org

Avoiding Edges



well-localized blob

edge

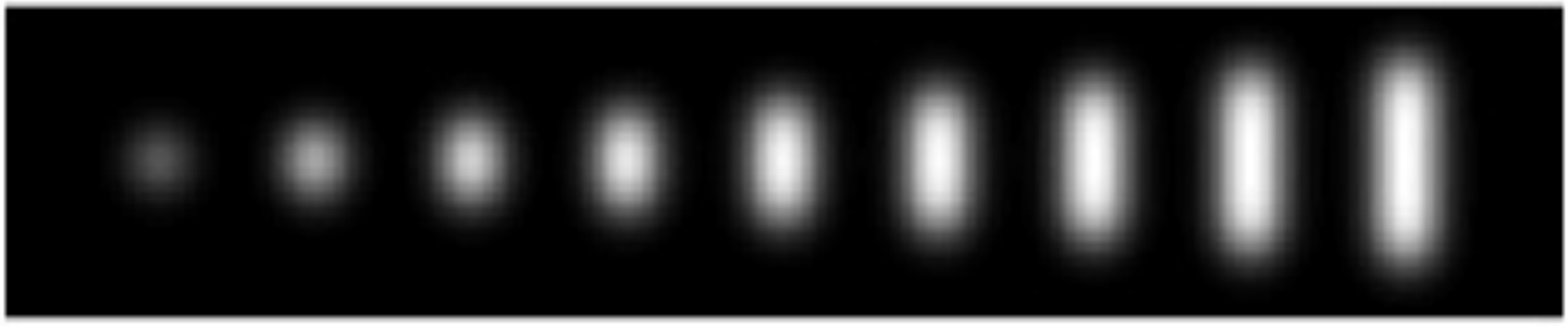
Consider 2D Hessian around detected keypoint (after refinement):

$$\mathrm{H} = \begin{pmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{pmatrix}$$

2nd order derivatives

image source: vlfeat.org

Avoiding Edges



well-localized blob

edge

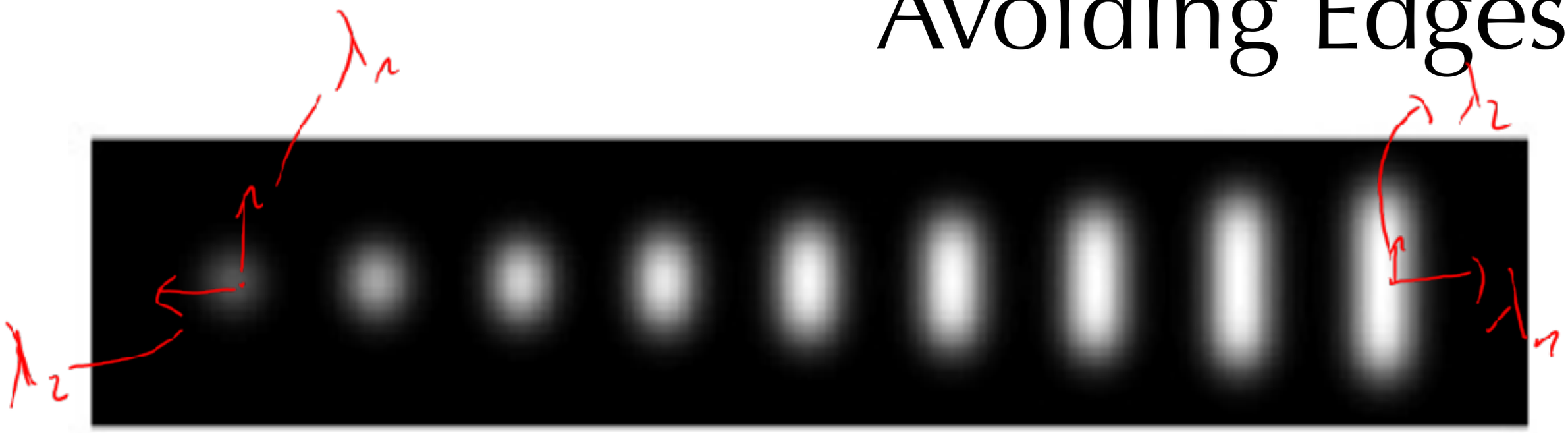
Consider 2D Hessian around detected keypoint (after refinement):

$$\mathbf{H} = \begin{pmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{pmatrix}$$

Eigenvalues $\lambda_1 \geq \lambda_2$ of $\mathbf{H}$ proportional to *principal curvature*

image source: vlfeat.org

Avoiding Edges

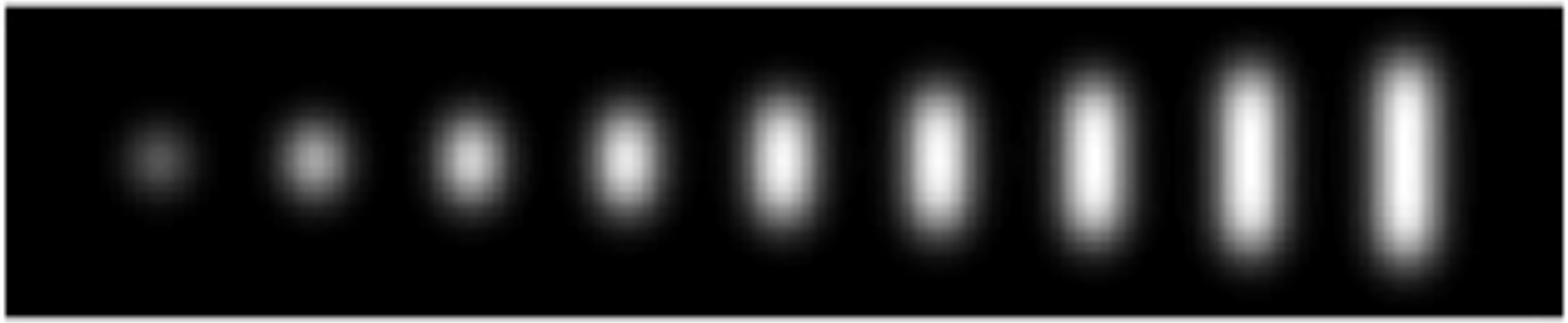


well-localized blob

edge

image source: vlfeat.org

Avoiding Edges



well-localized blob

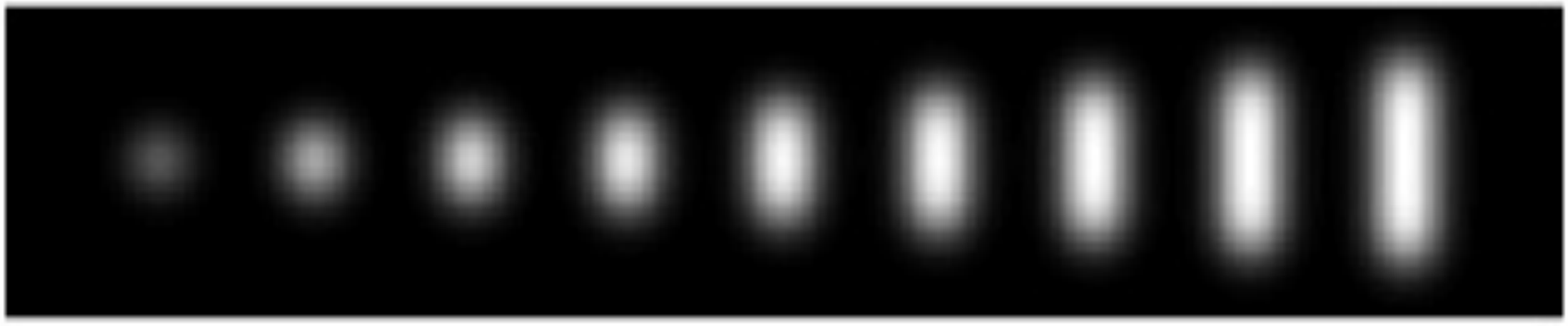
edge

Detecting edges: Look at ratio of Eigenvalues ($\lambda_1 \geq \lambda_2$)

$$\frac{\lambda_1}{\lambda_2} < r$$

image source: vlfeat.org

Avoiding Edges



well-localized blob

edge

Detecting edges: Look at ratio of Eigenvalues ($\lambda_1 \geq \lambda_2$)

$$\frac{\lambda_1}{\lambda_2} < r$$

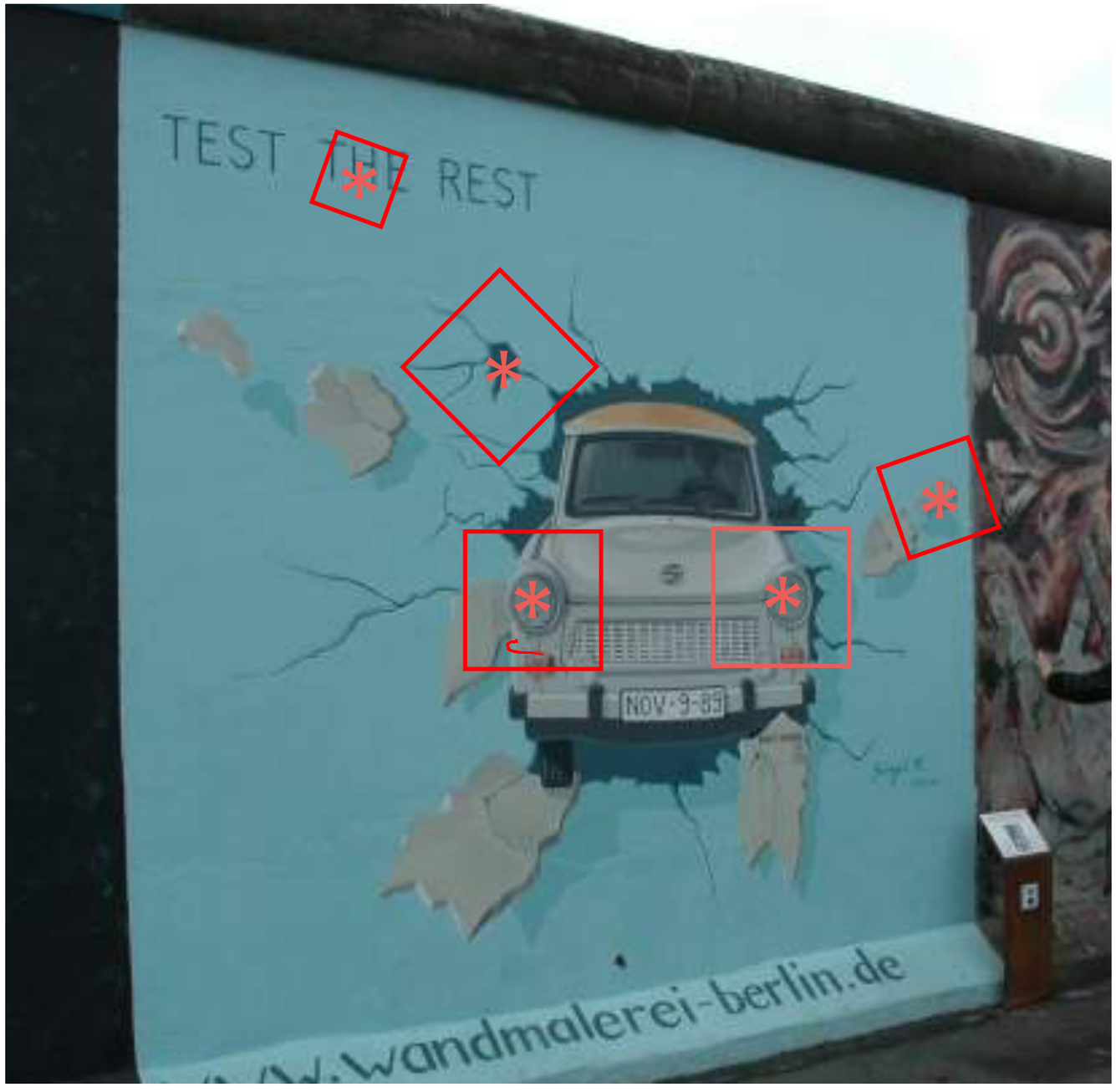
No need to explicitly compute Eigenvalues (see Lowe's paper)

image source: vlfeat.org

Avoiding Edges

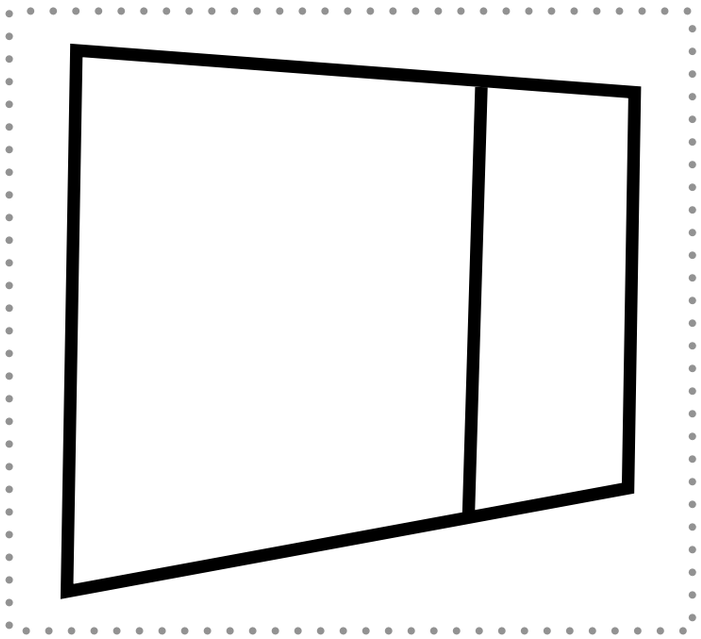
Avoiding Edges

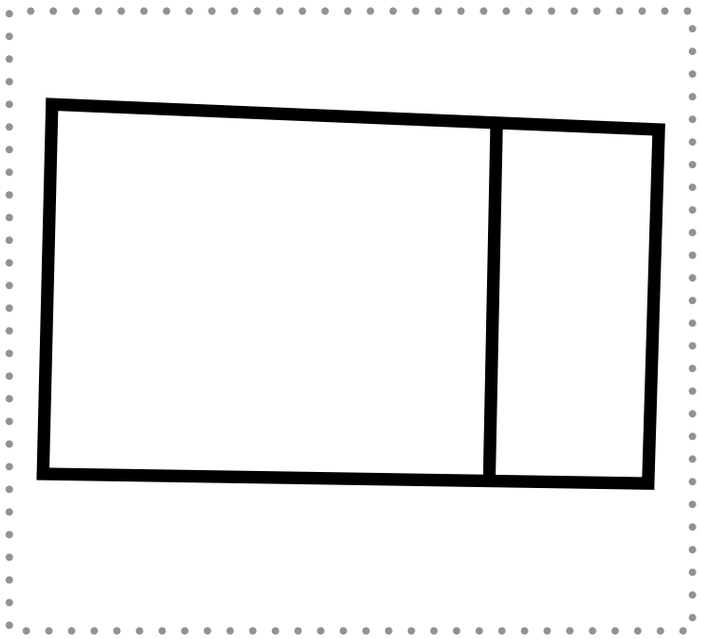
Scale-Invariant Feature Transform (SIFT) Features

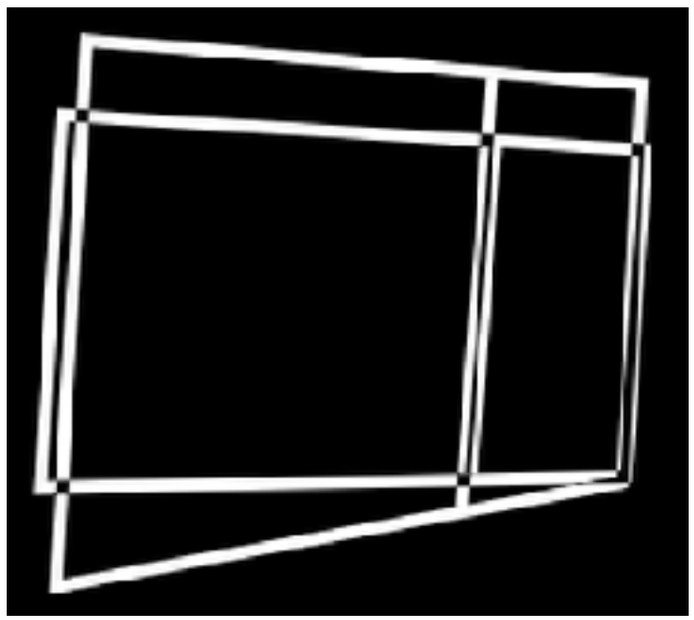


- Find a set of *interest* points
- Extract oriented patch around each interest point
- **Compute SIFT descriptor for each patch**

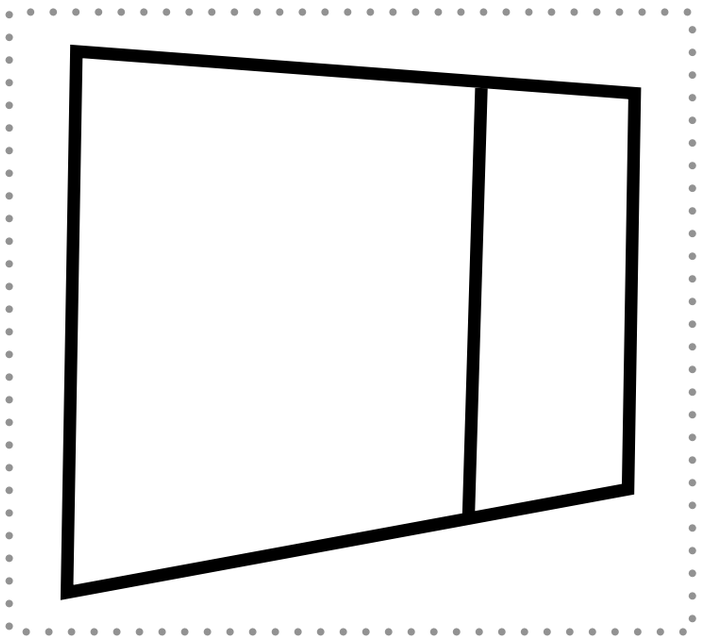
Perspective Distortion

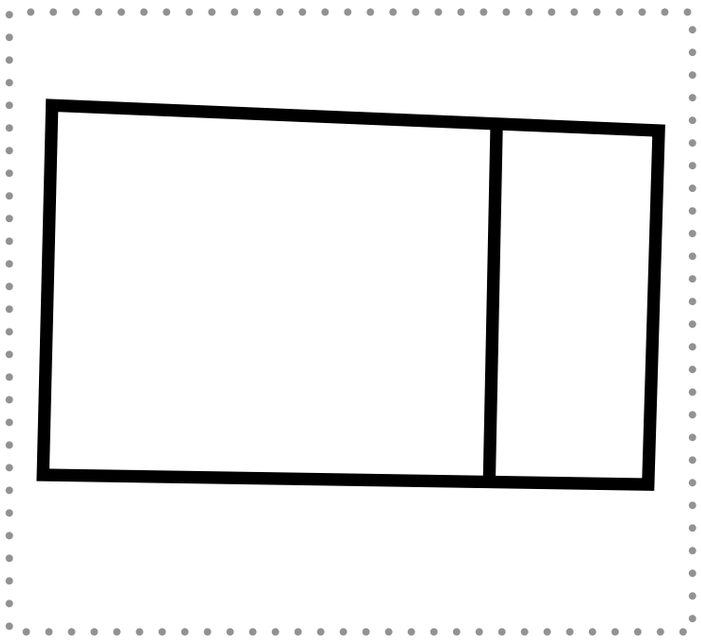


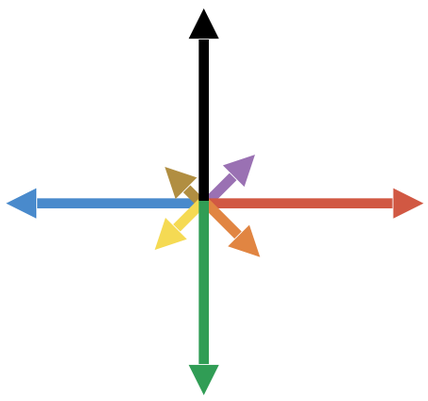


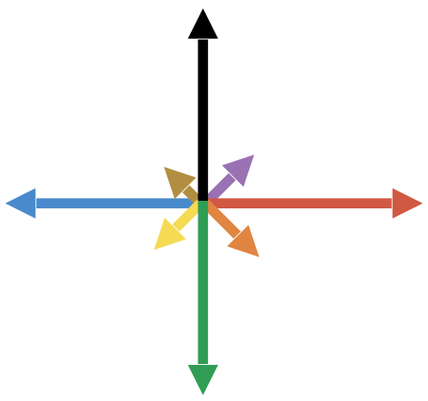


Perspective Distortion

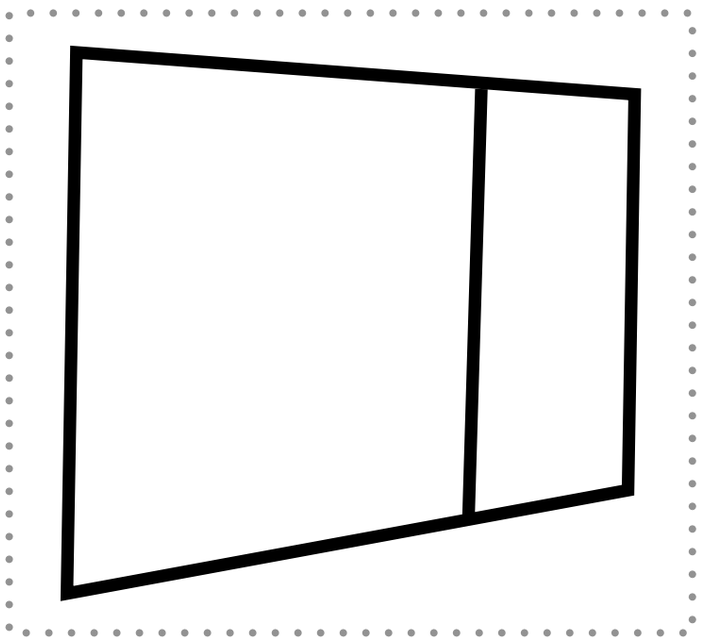


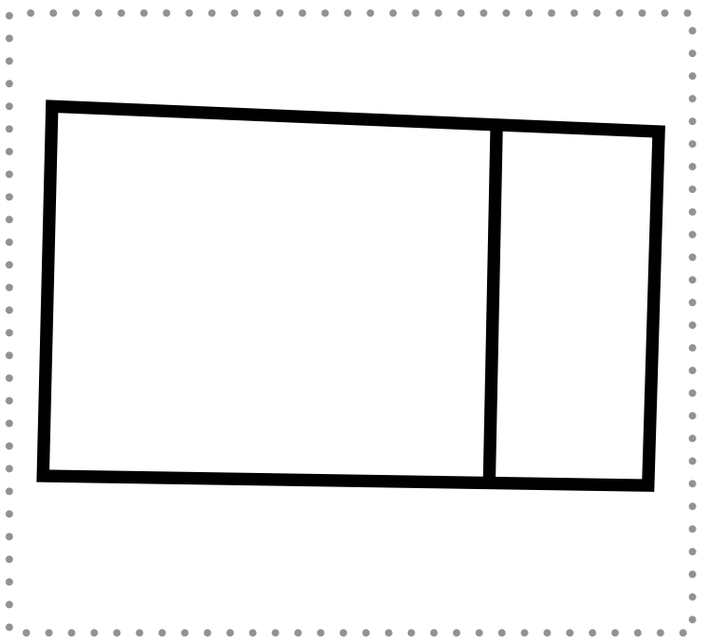


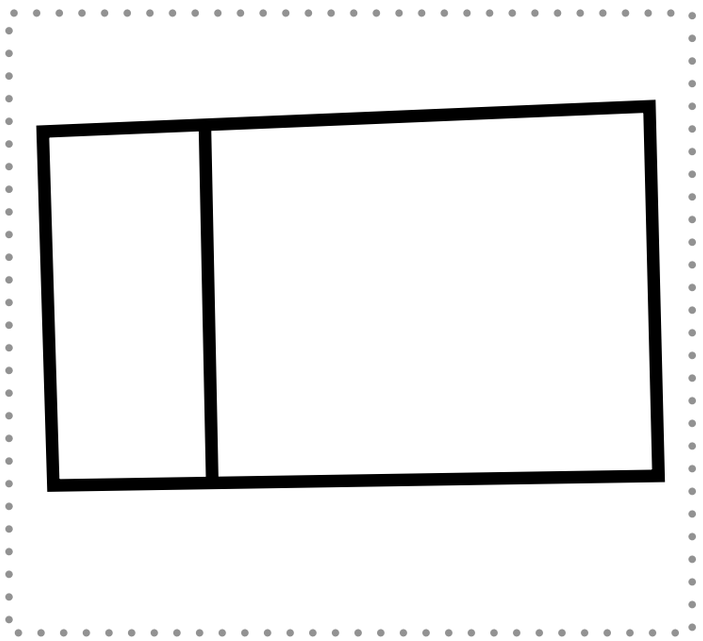


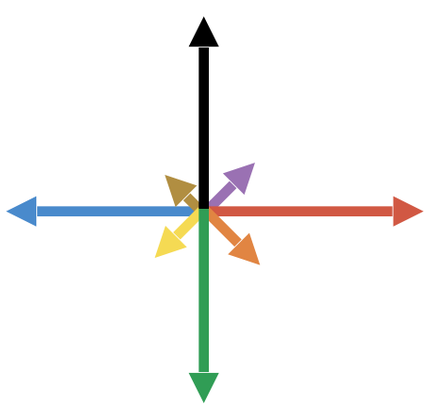


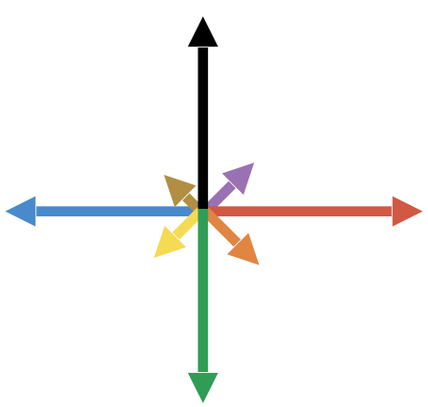
Perspective Distortion

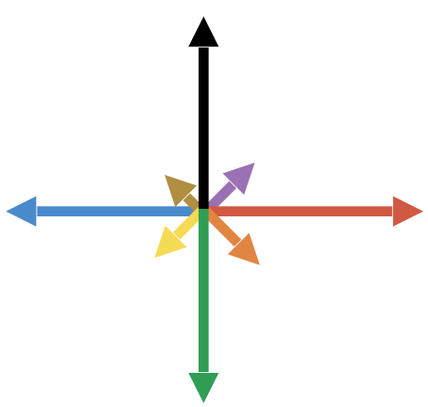




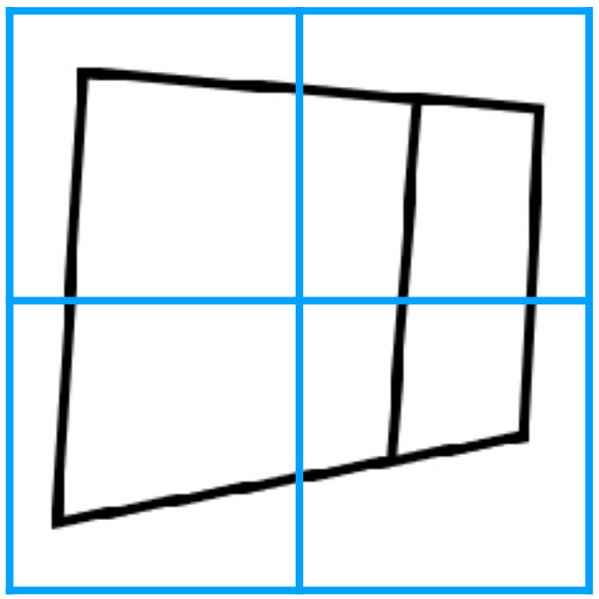






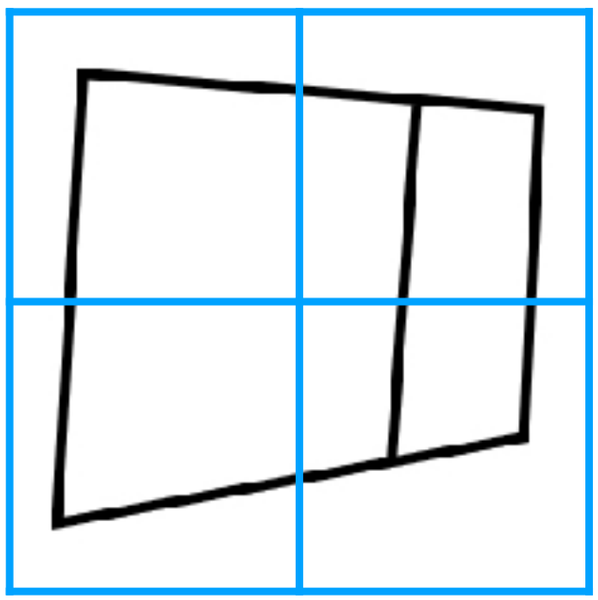


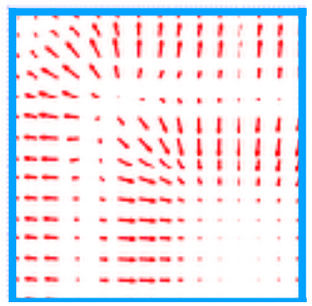
SIFT Descriptor (Idea)



Split patch in 2x2 regions
(SIFT actually uses 4x4 regions)

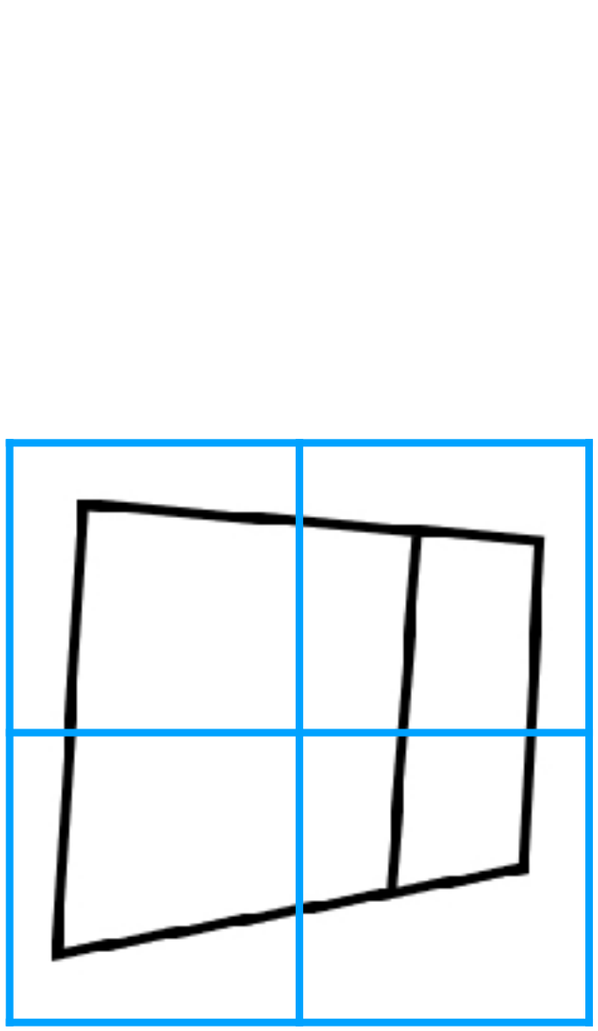
SIFT Descriptor (Idea)

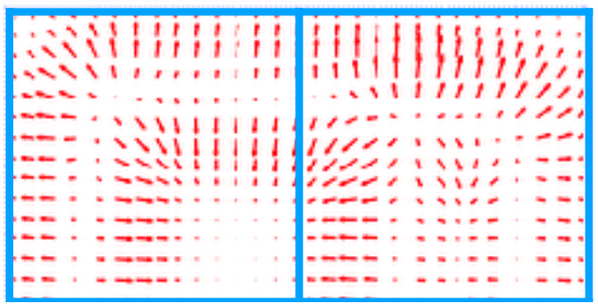


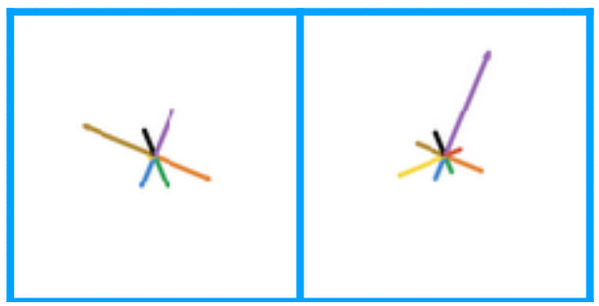


Compute a 8-bin gradient histogram for each region

SIFT Descriptor (Idea)

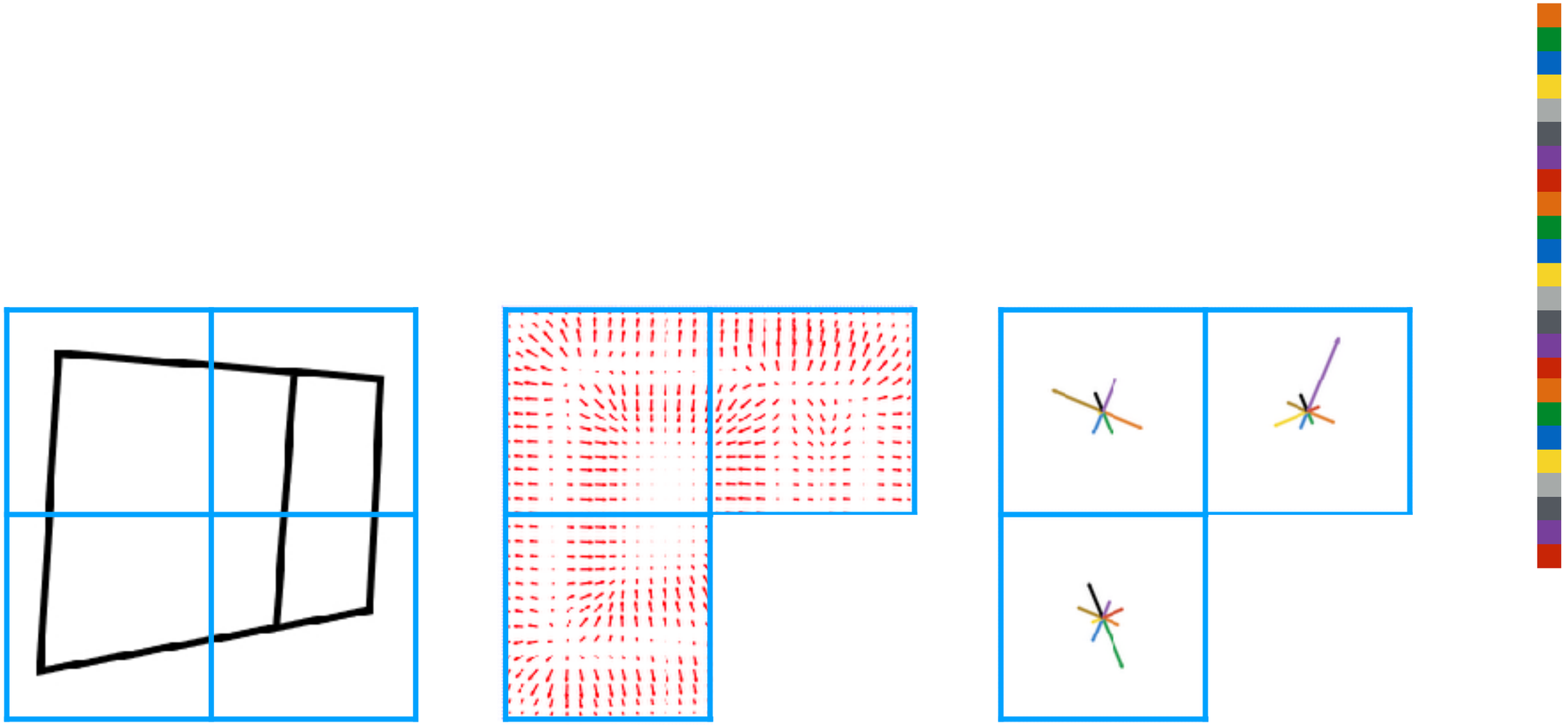






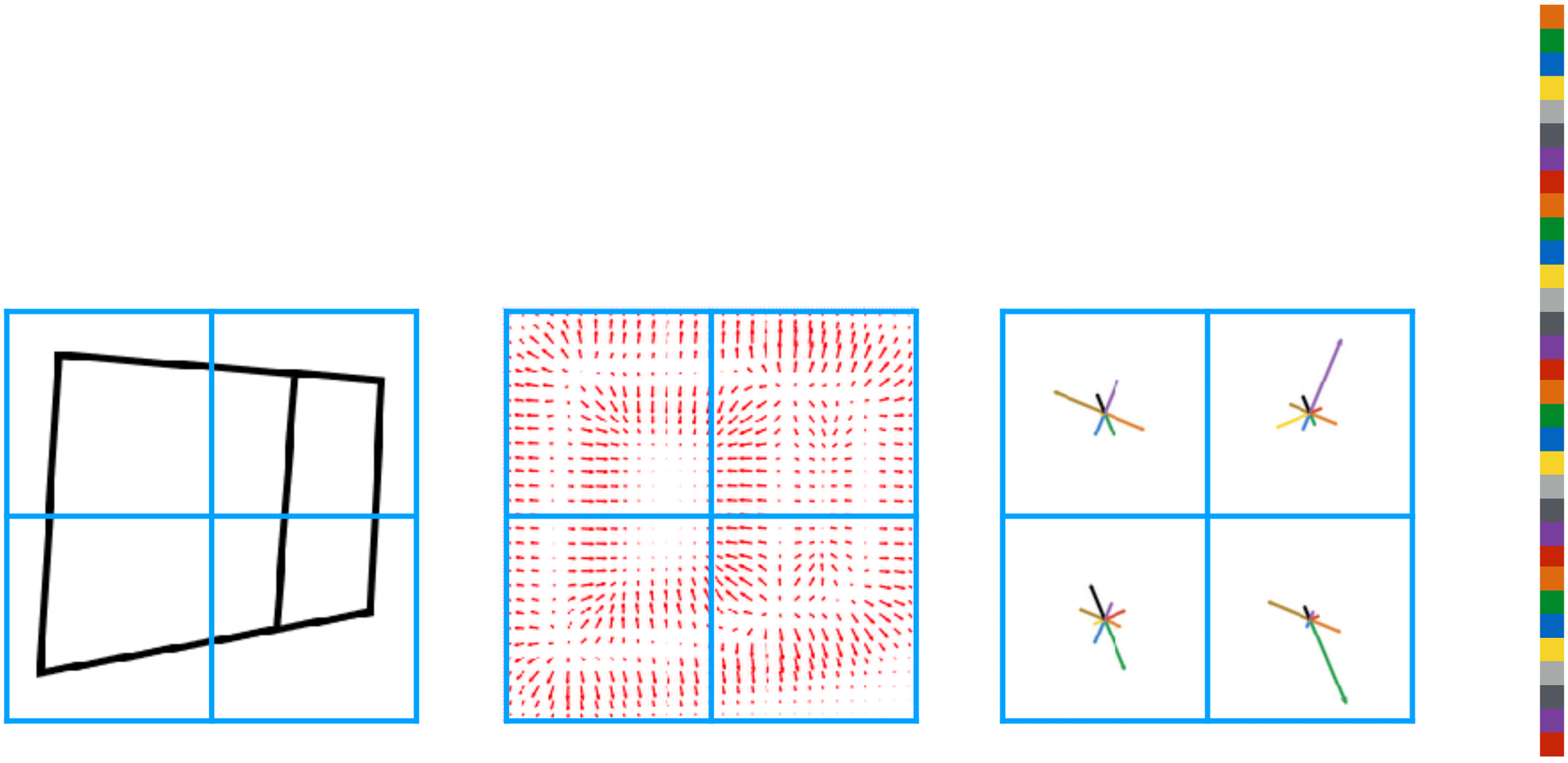
Compute a 8-bin gradient histogram for each region

SIFT Descriptor (Idea)

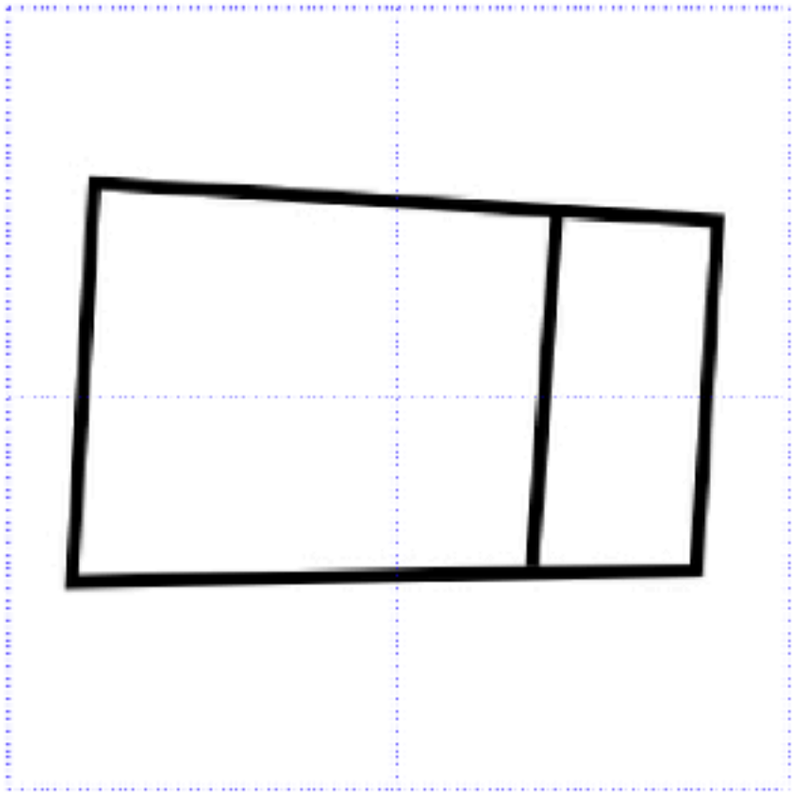


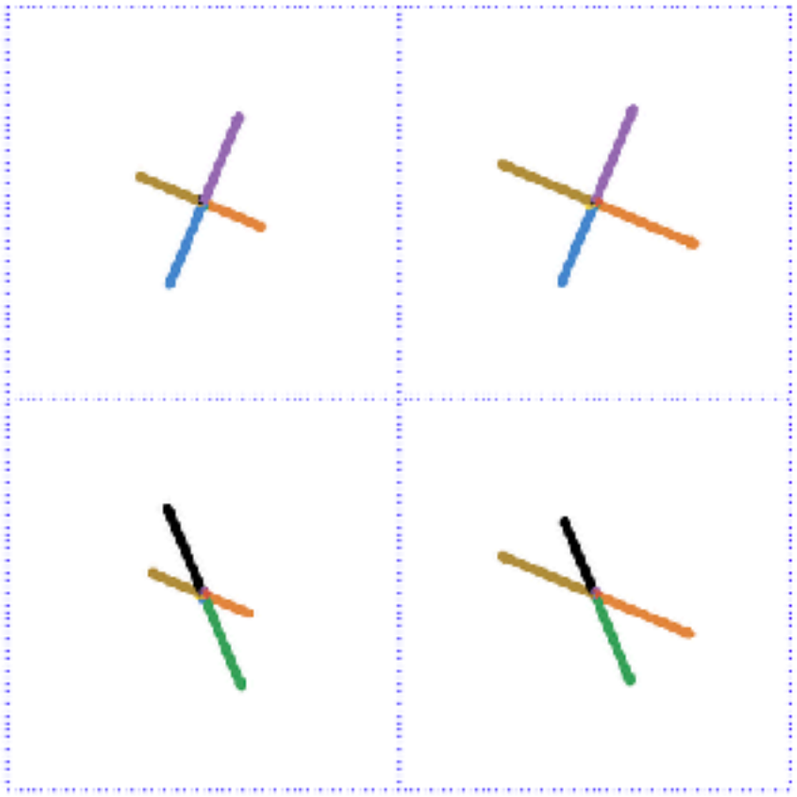
Compute a 8-bin gradient histogram for each region

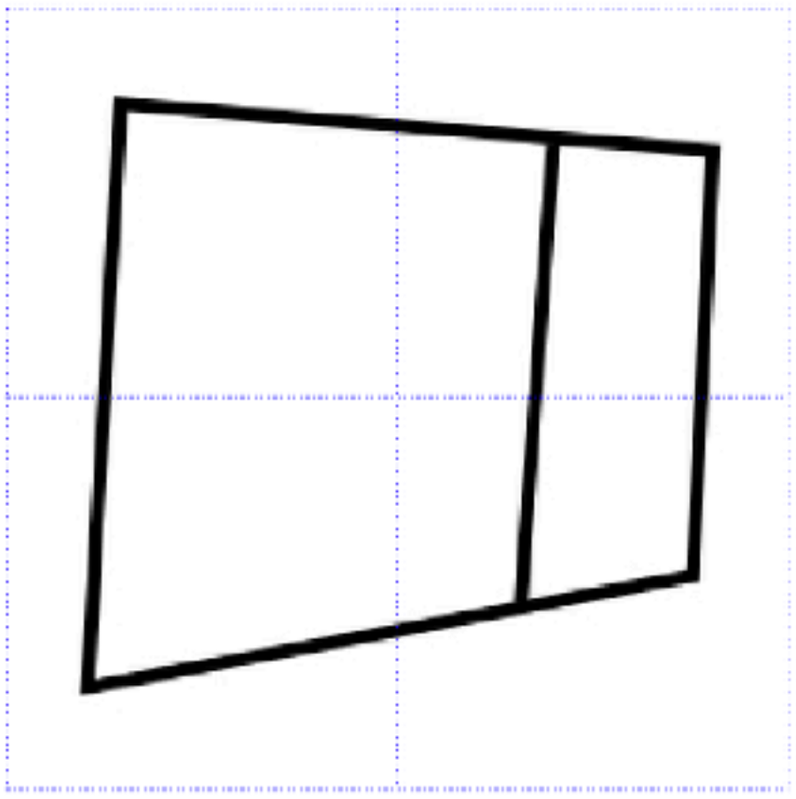
SIFT Descriptor (Idea)

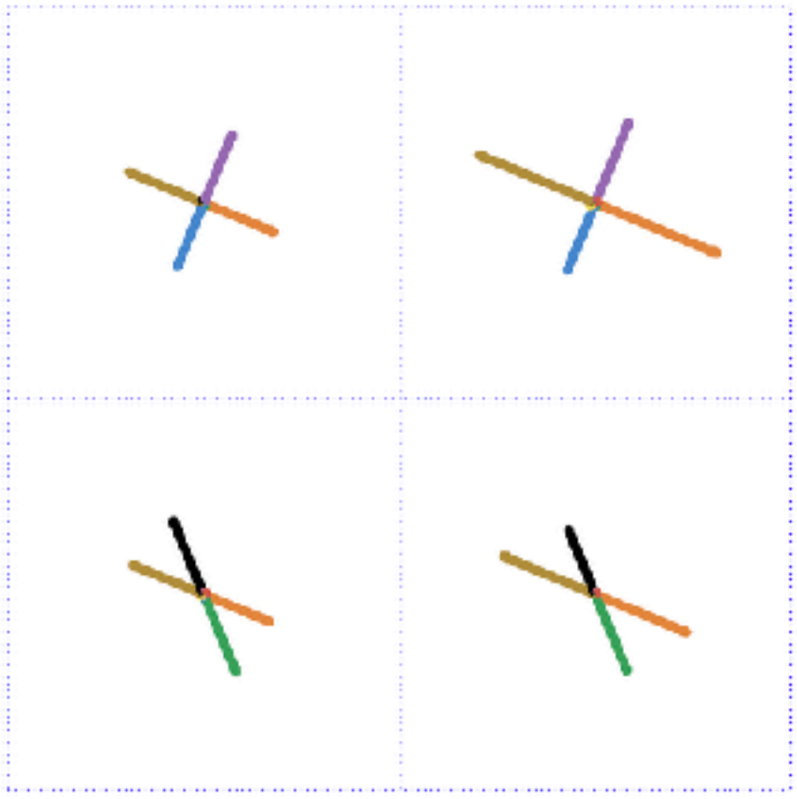


L_2 Normalize the descriptor

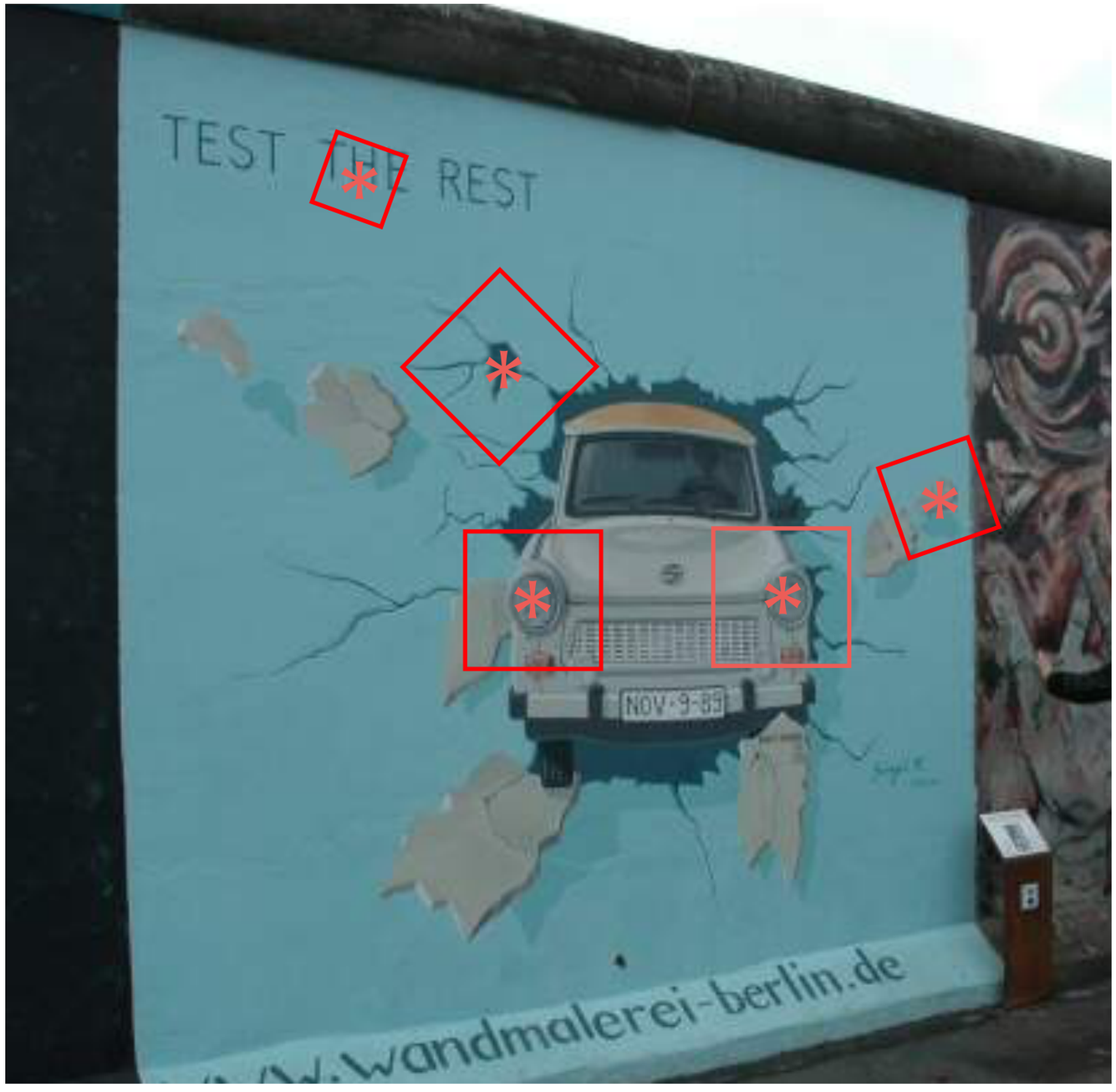








Scale-Invariant Feature Transform (SIFT) Features



- Find a set of *interest* points
- **Extract oriented patch around each interest point**
- Compute SIFT descriptor for each patch

Scale Invariance

Detect a size s, for each point

Scale Invariance

Detect a size s, for each point

Scale Invariance

Detect a size s, , for each point
Scale patch size to fit.
Compute gradients in $L(x, y, s^2)$

Scale Invariance

size=5

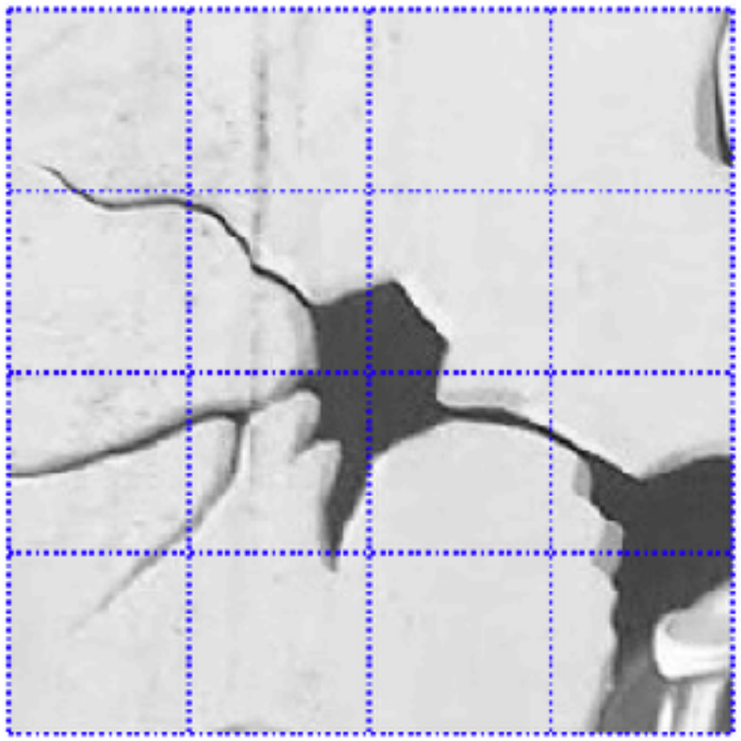
size=0.5

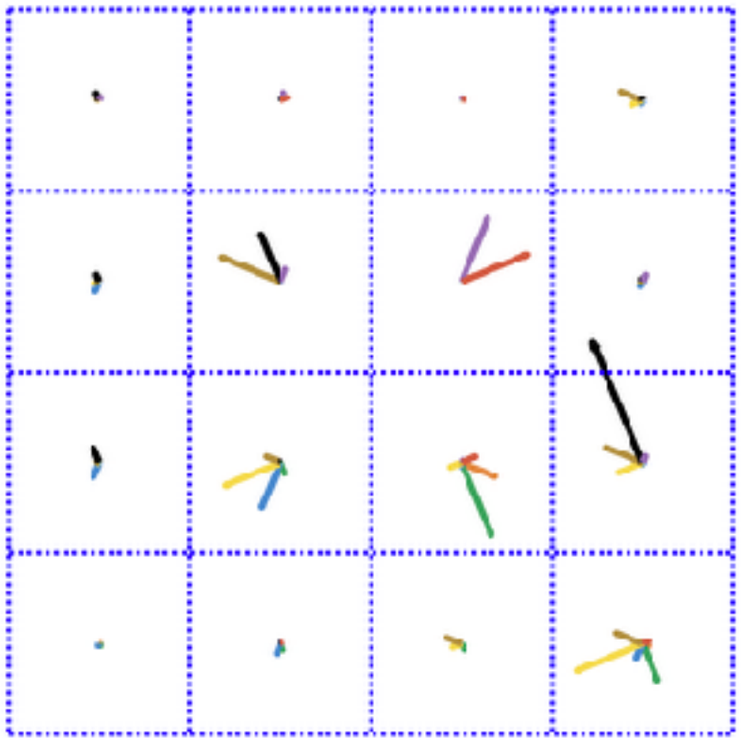
Scale Invariance

size=5 size=0.5

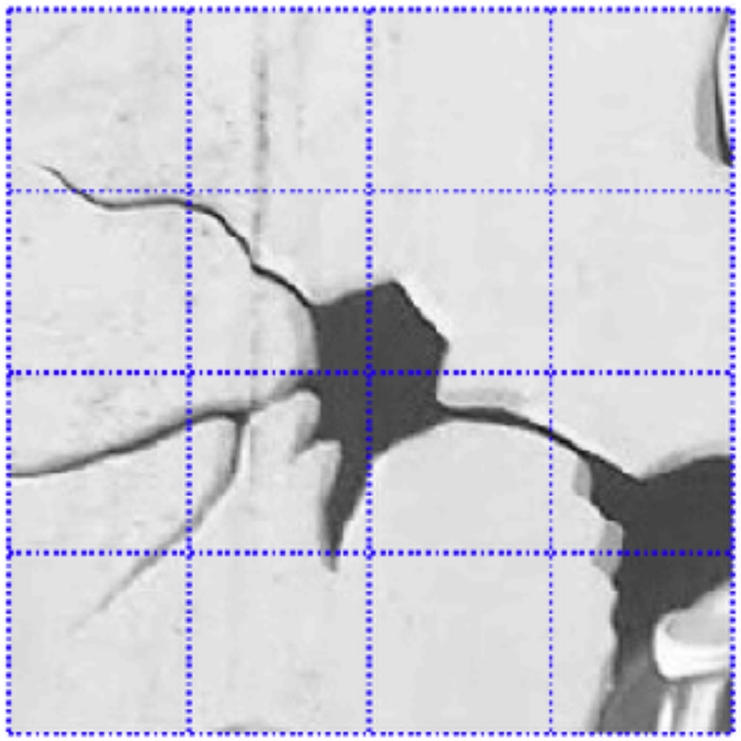
Map to canonical patch before descriptor extraction

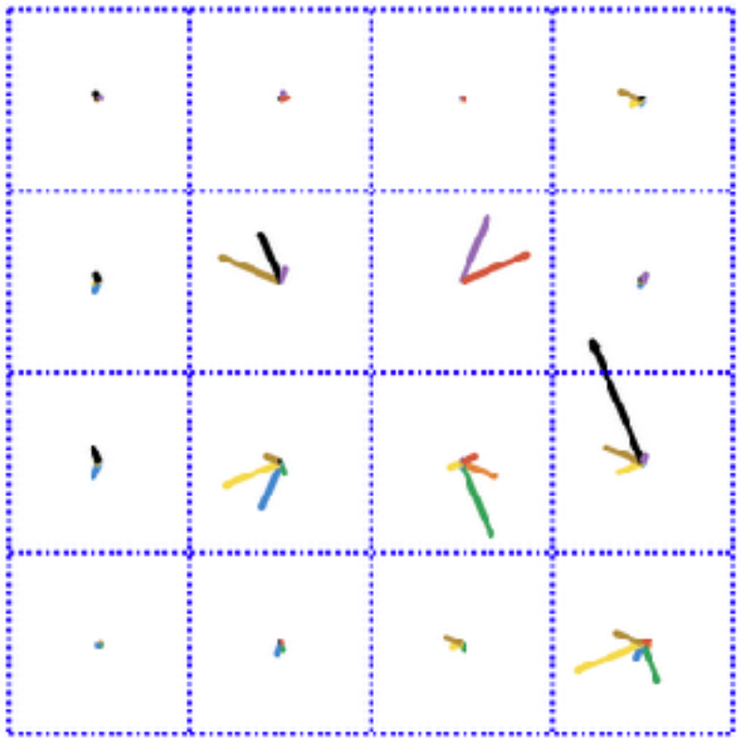
Canonical Patches

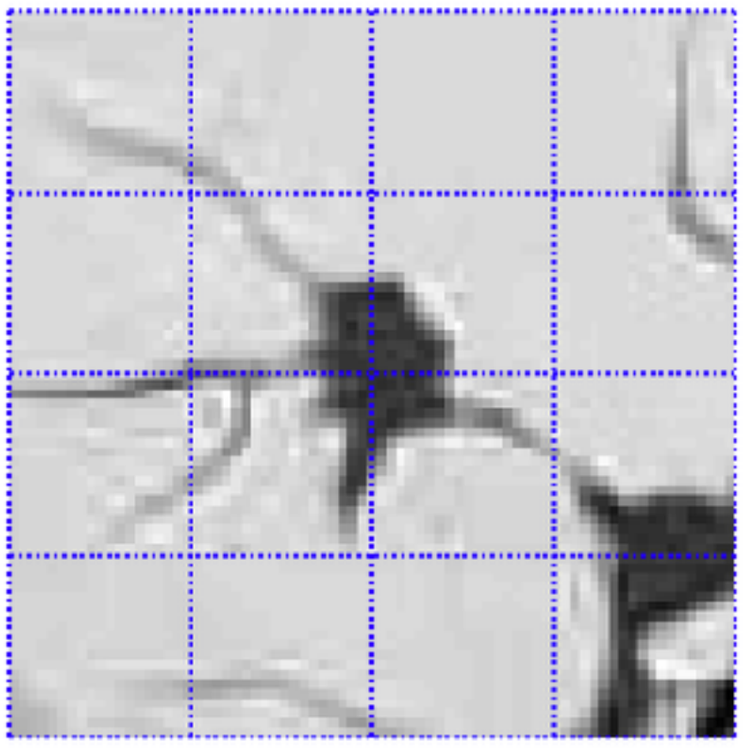


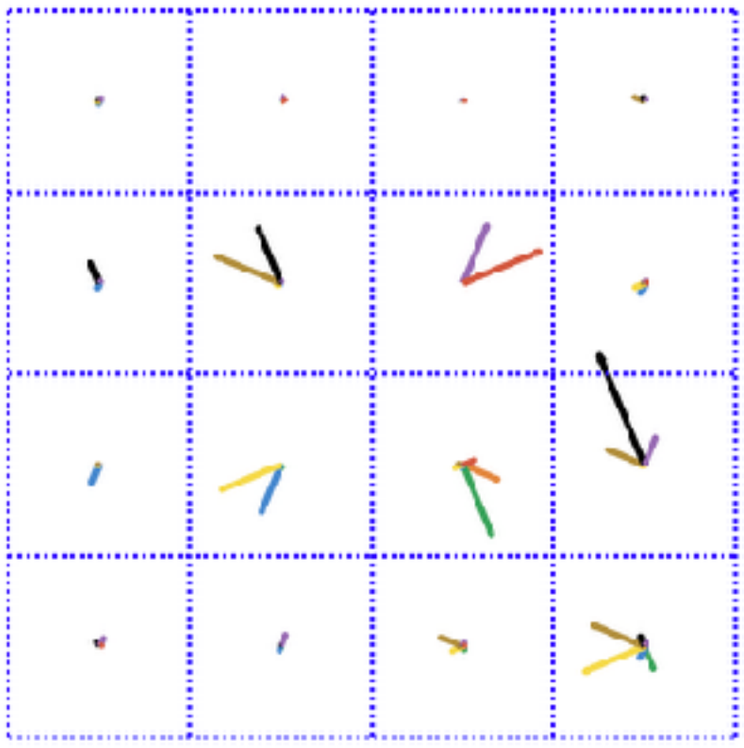


Canonical Patches

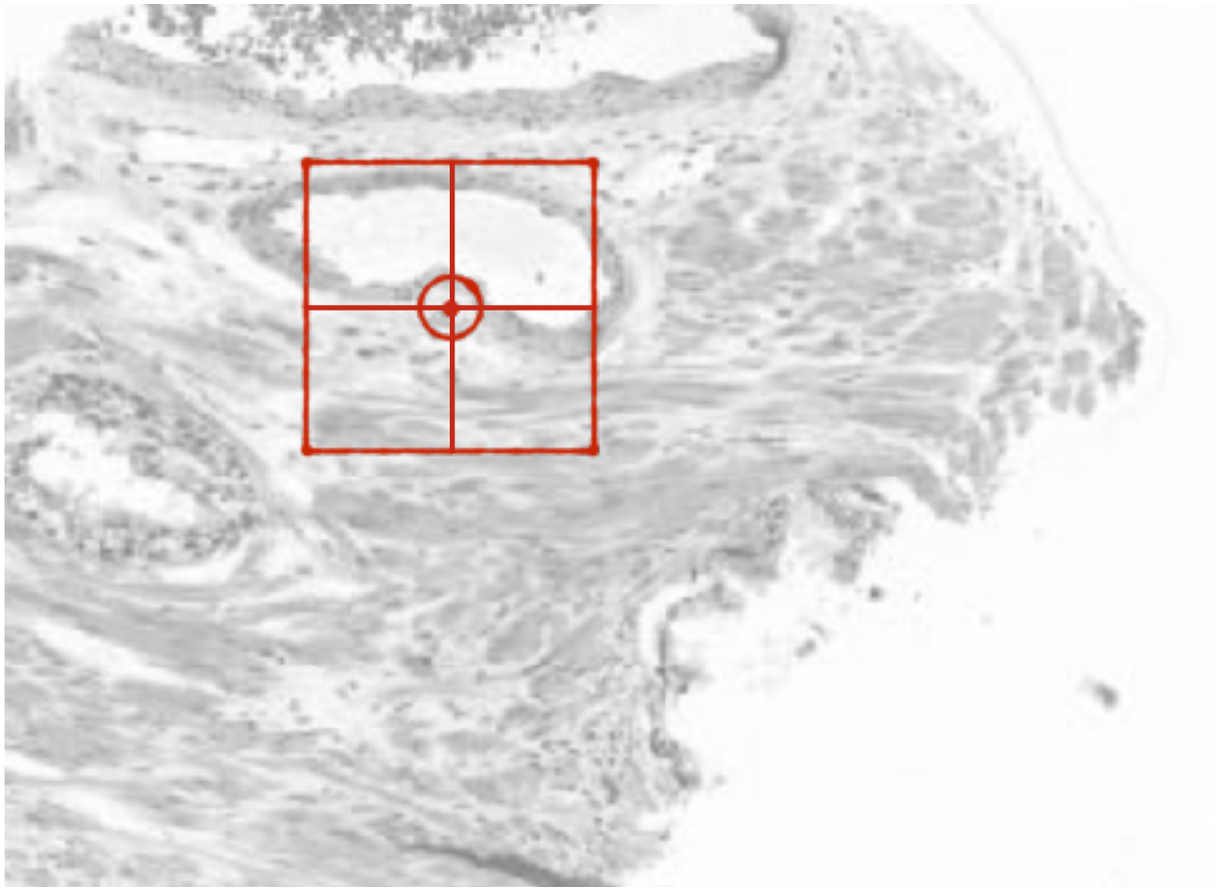


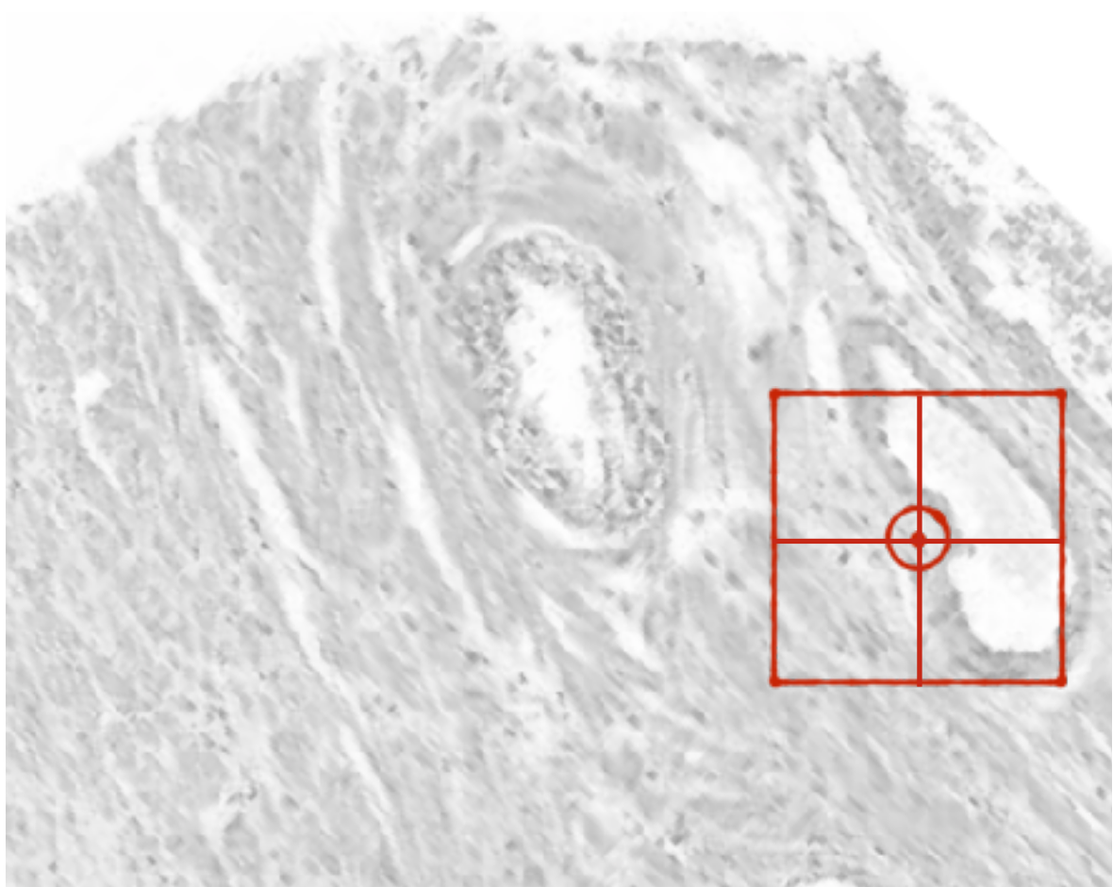




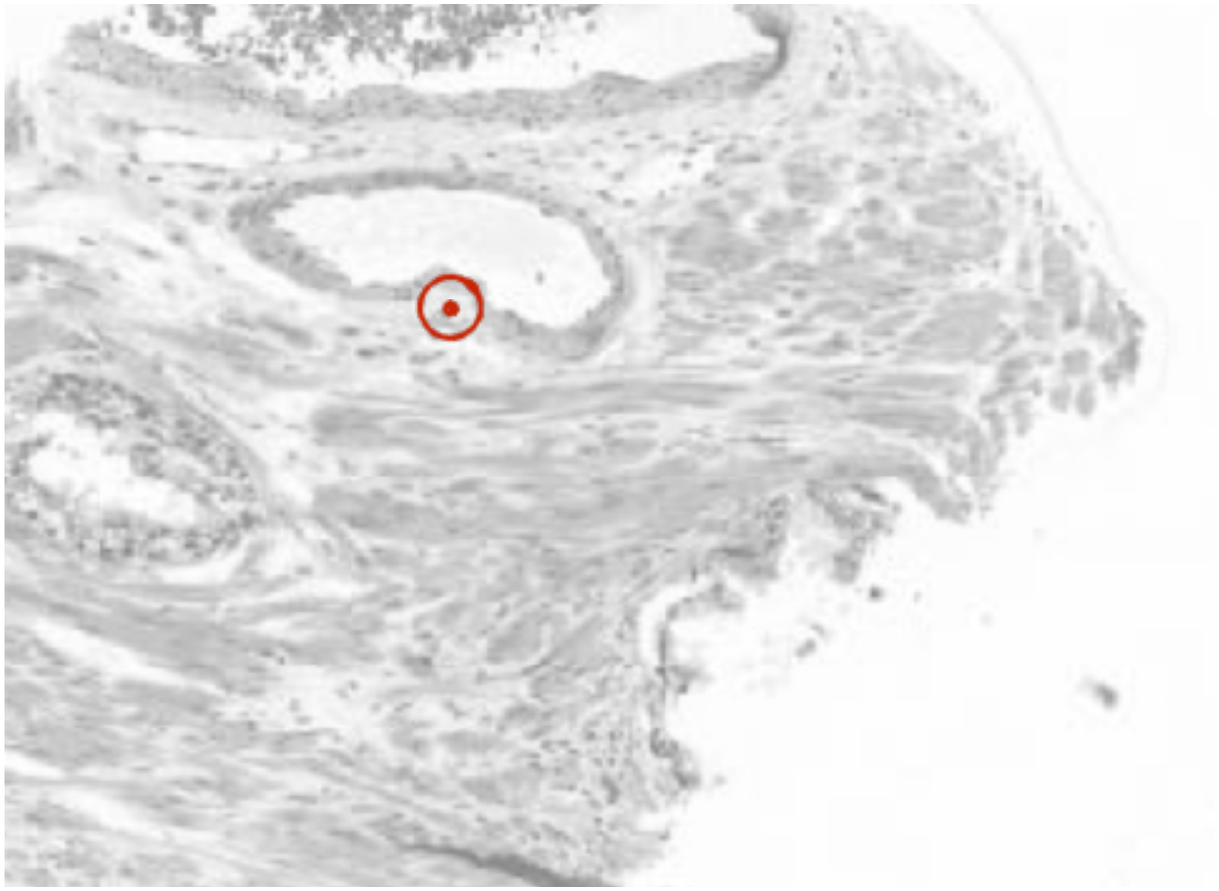


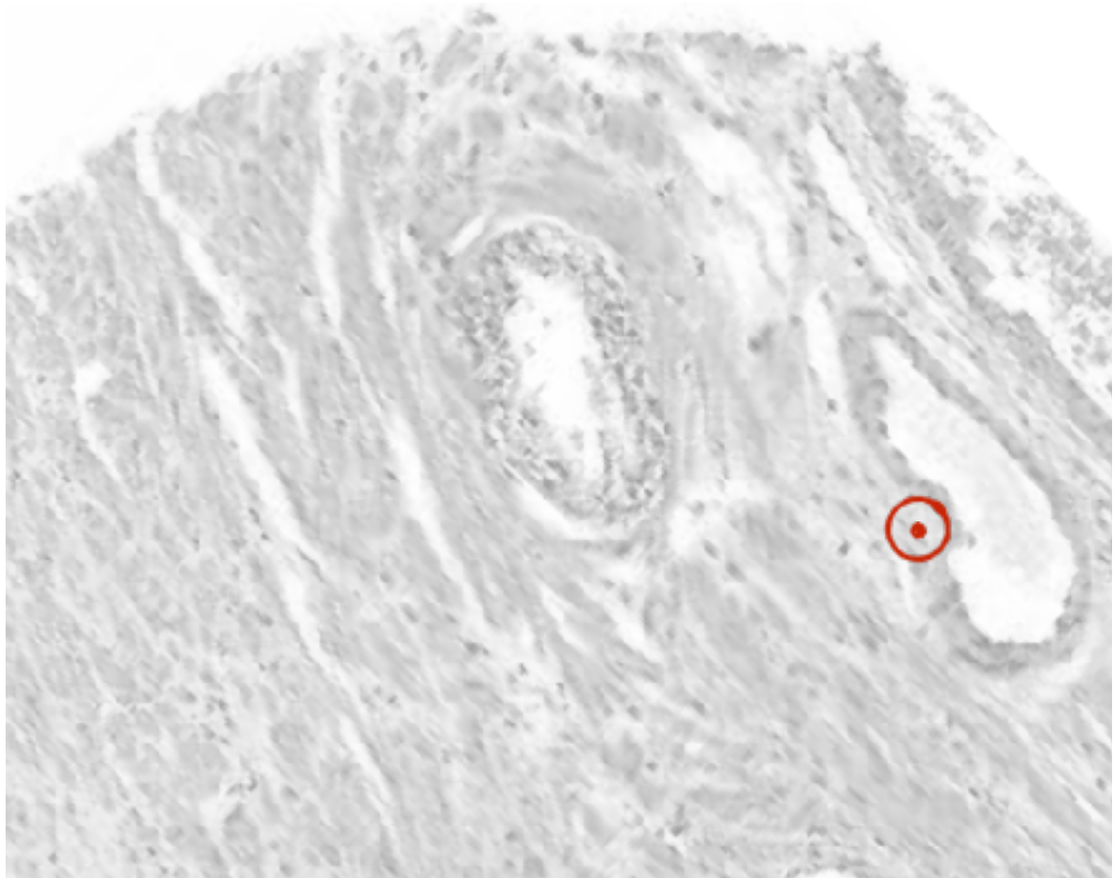
Challenge: Rotations





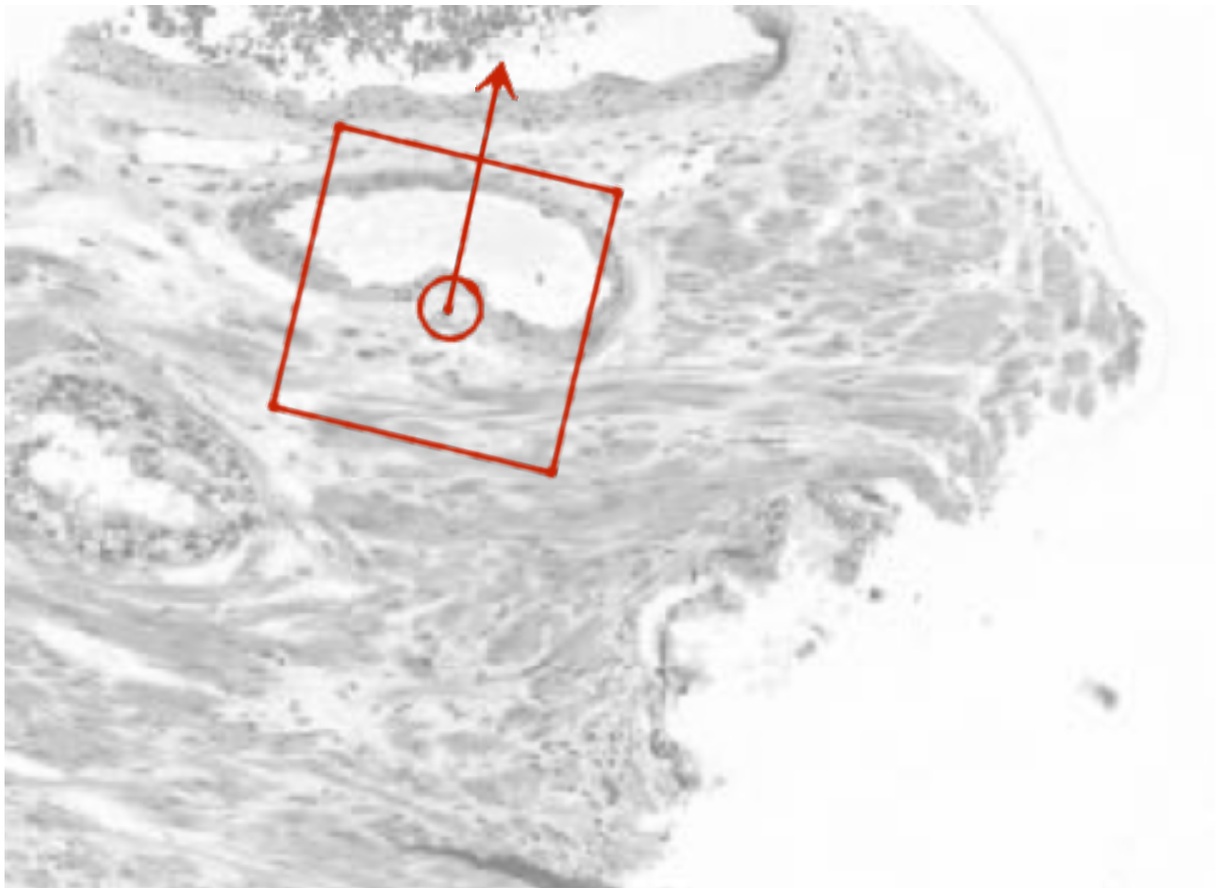
Challenge: Rotations

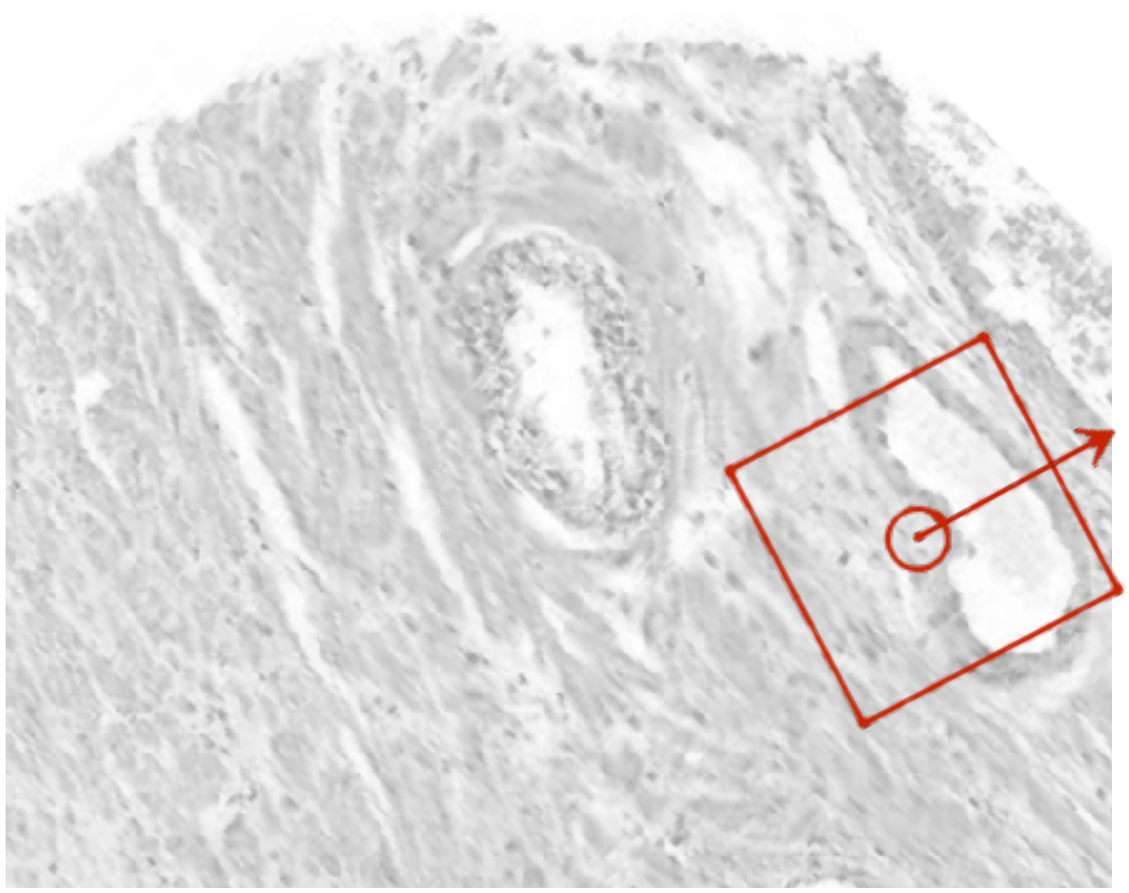




Estimate a dominant direction

Challenge: Rotations

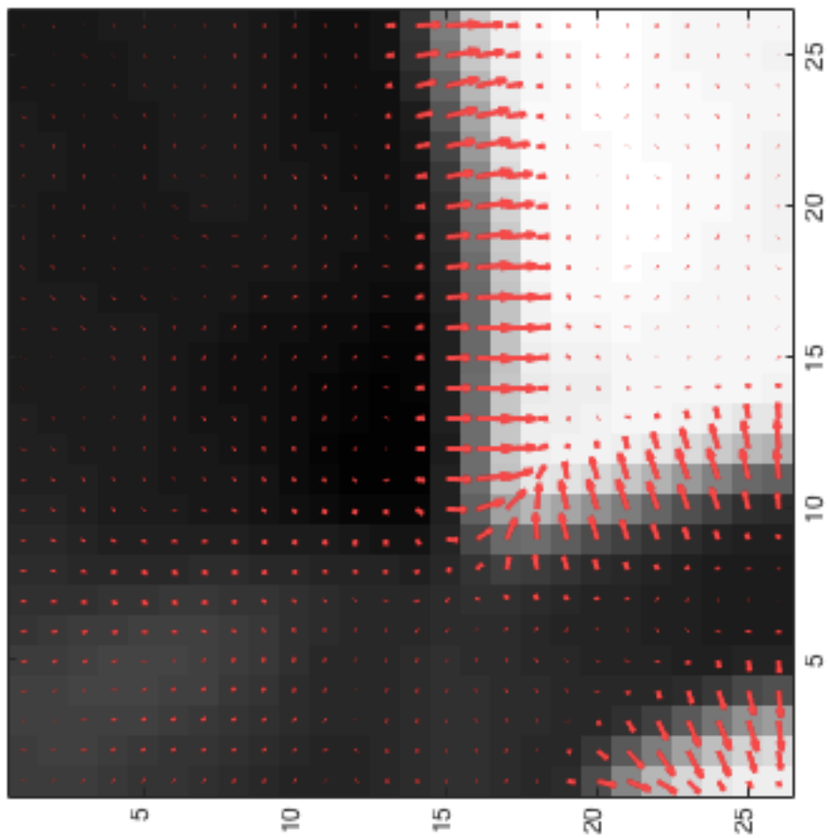


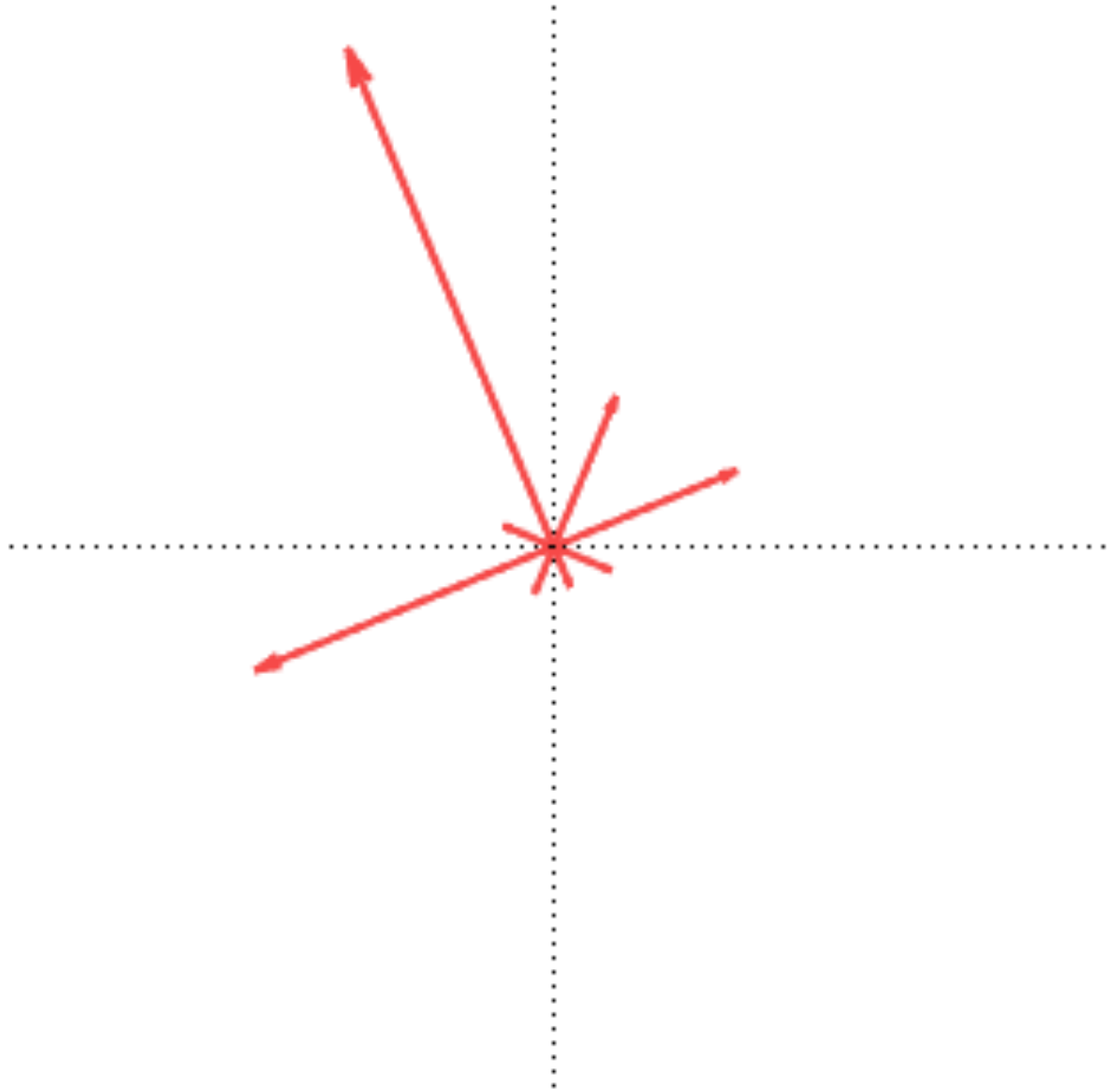


Estimate a dominant direction

Align patches to this direction

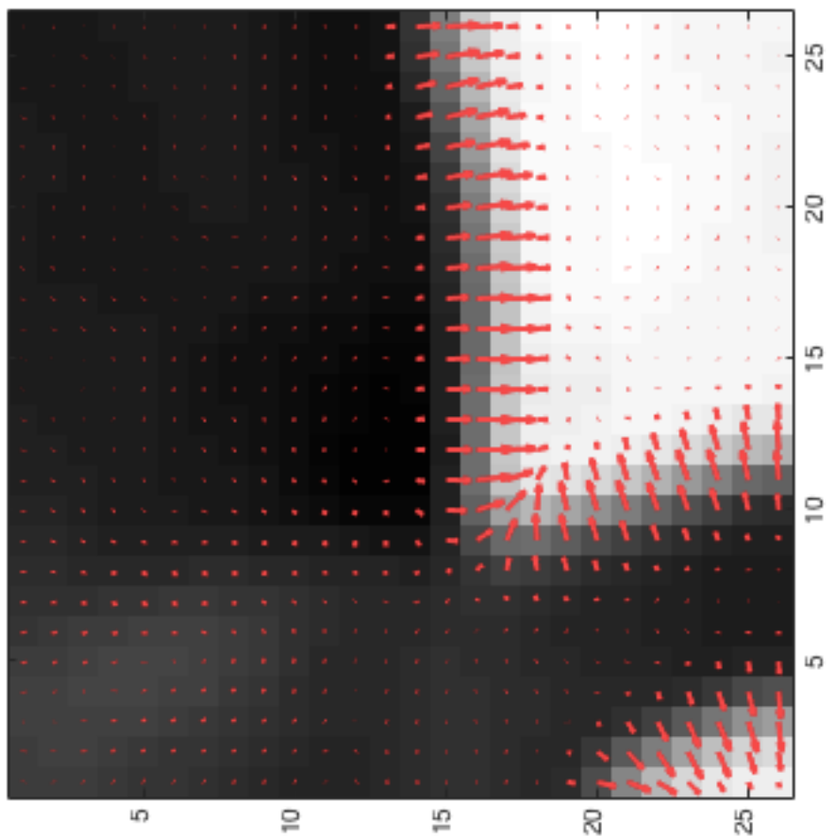
Orientation Assignment

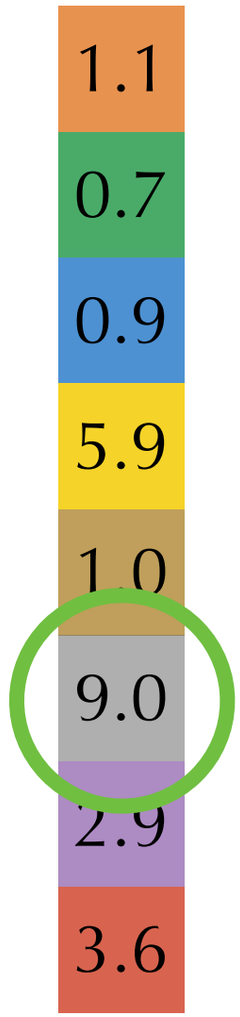


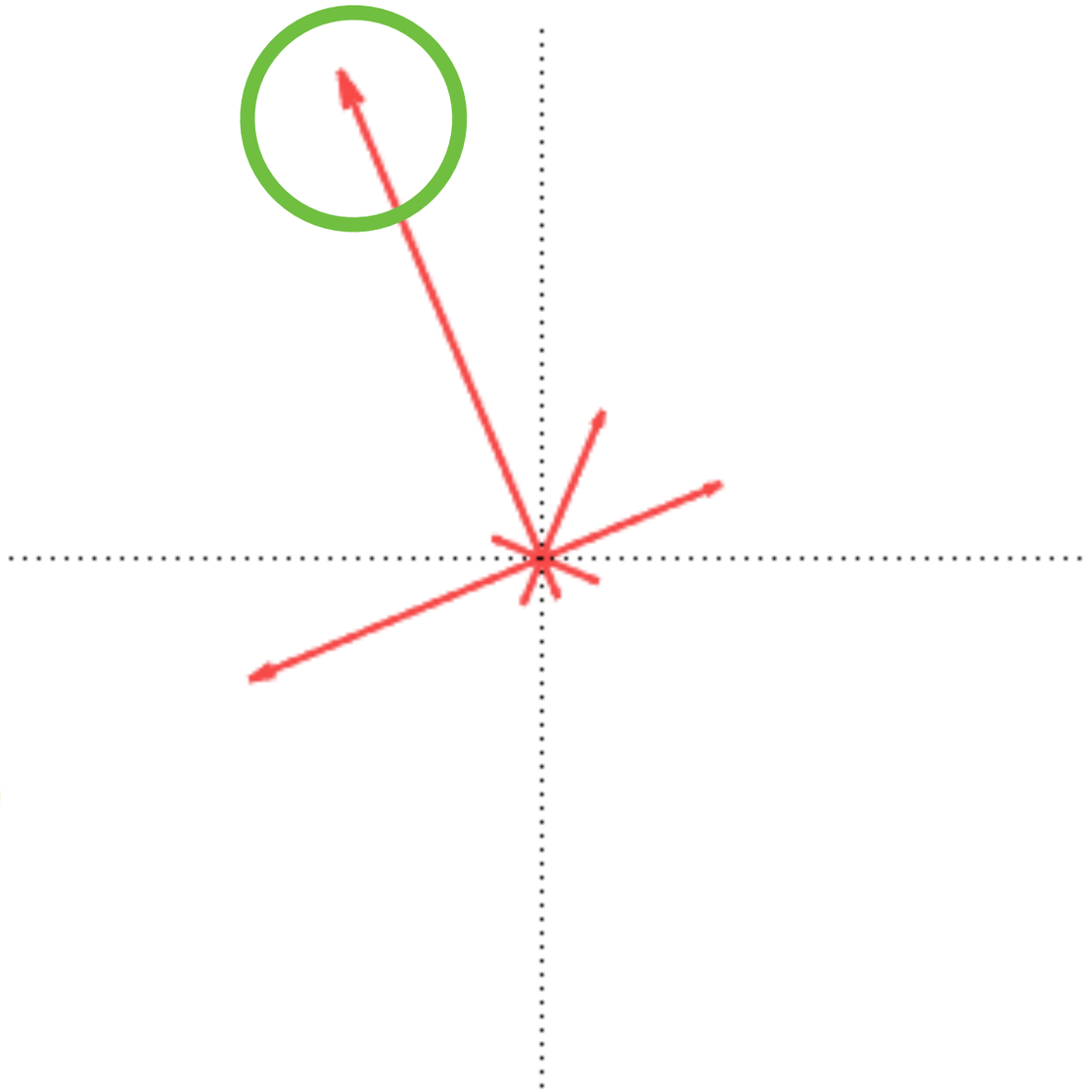


Histogram of gradients around the point.
Use the dominant orientation.

Orientation Assignment

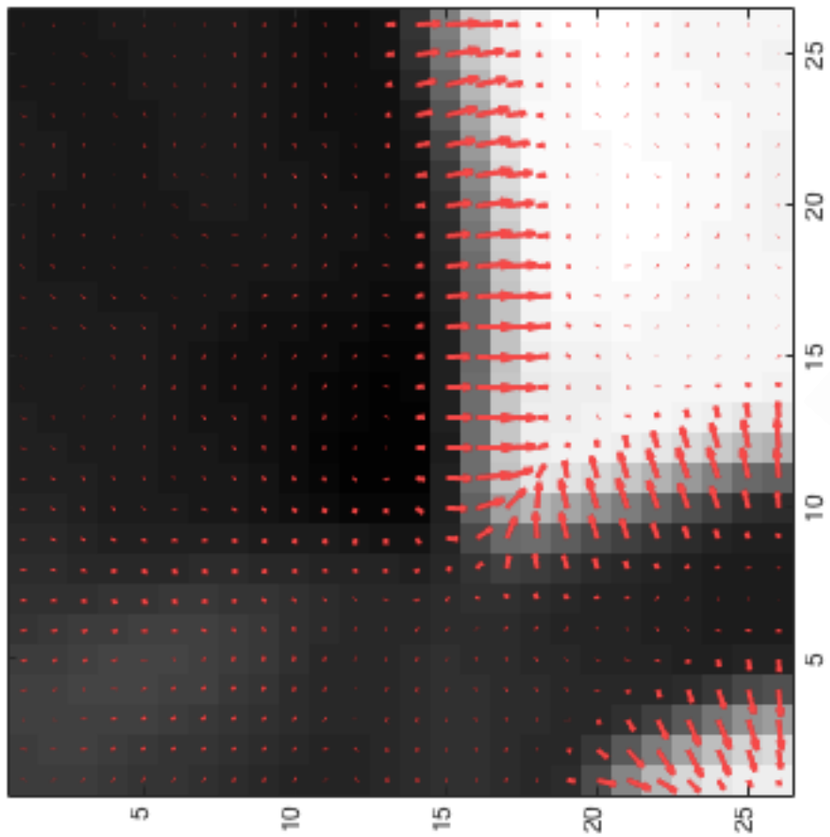


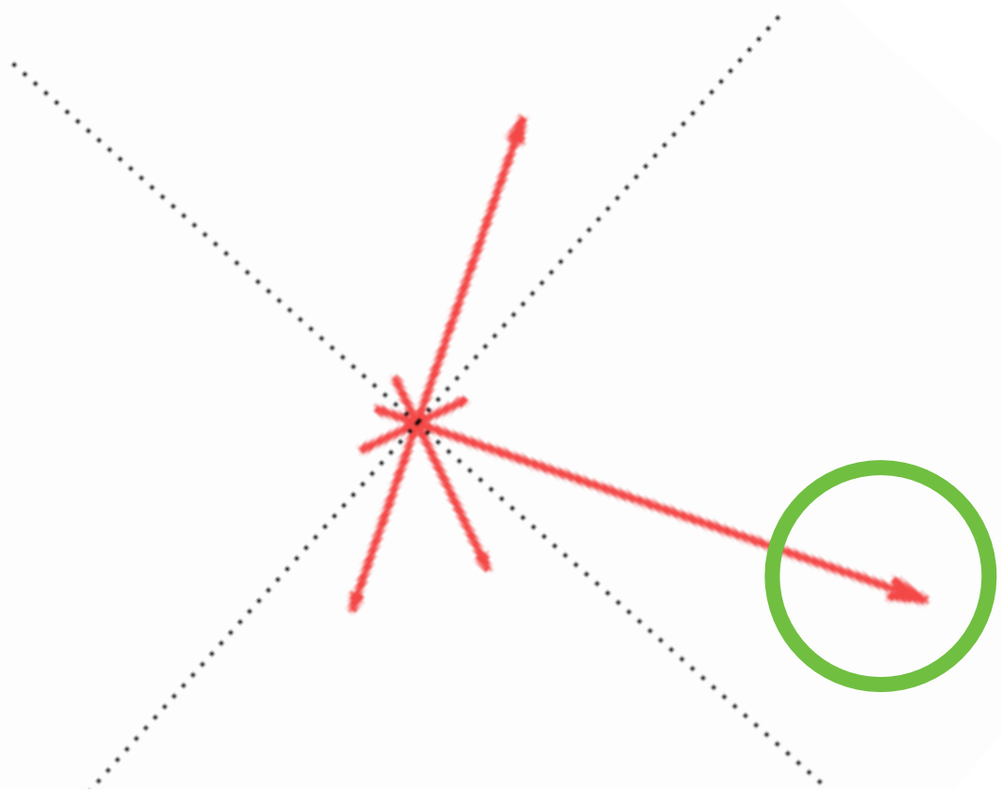




Histogram of gradients around the point.
Use the dominant orientation.

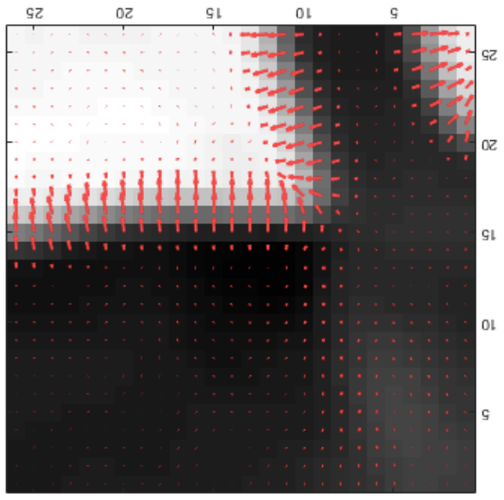
Orientation Assignment

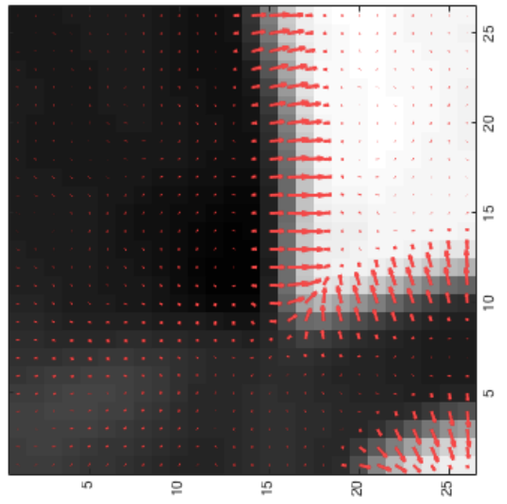




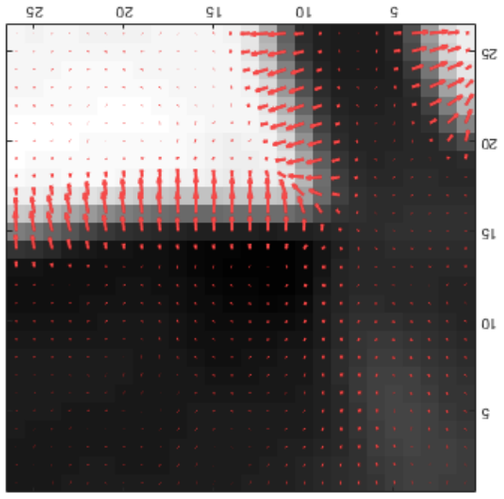
Histogram of gradients around the point.
Use the dominant orientation.

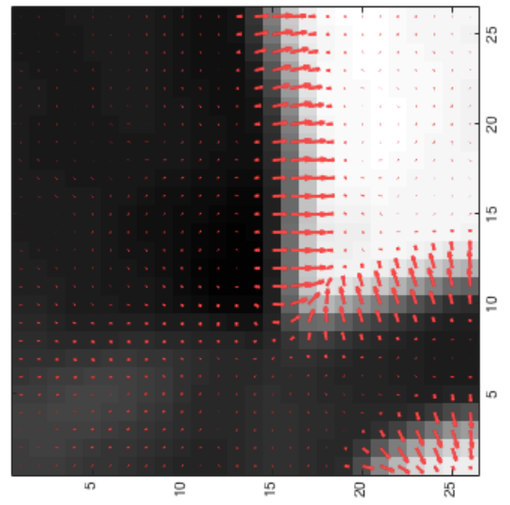
Orientation Assignment

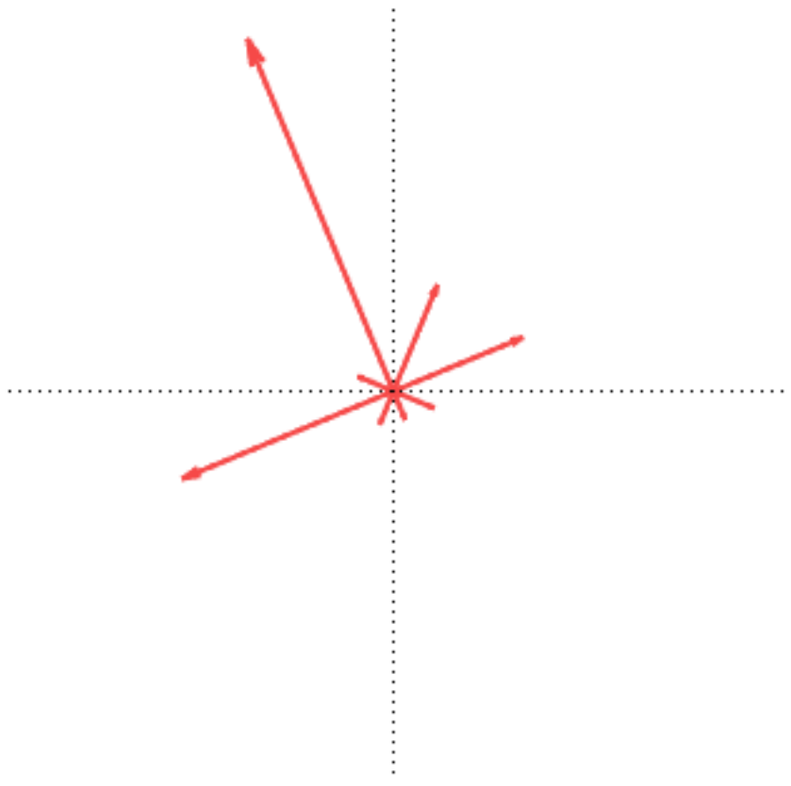


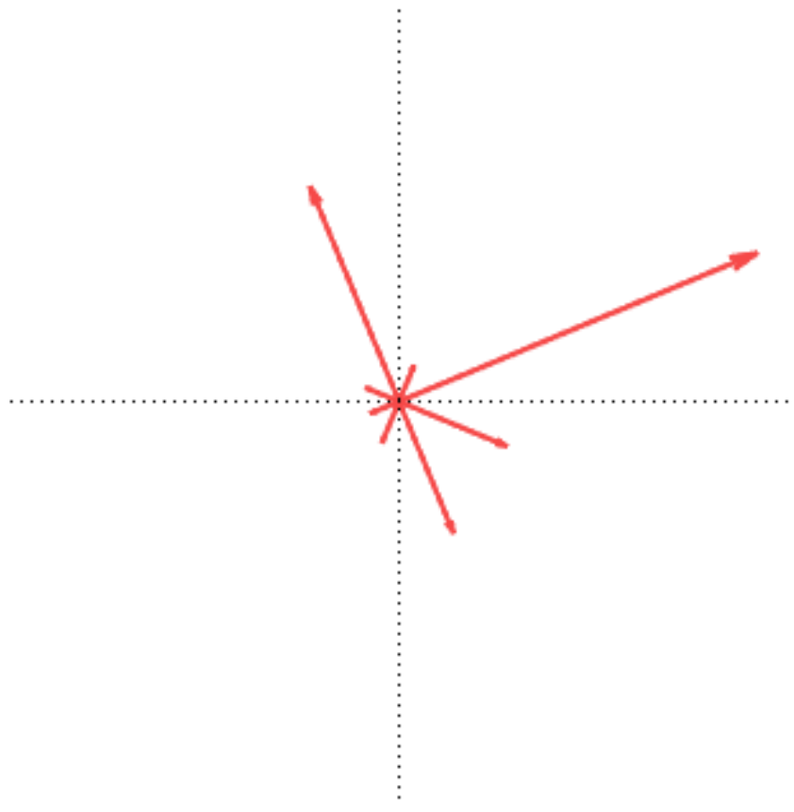


Orientation Assignment

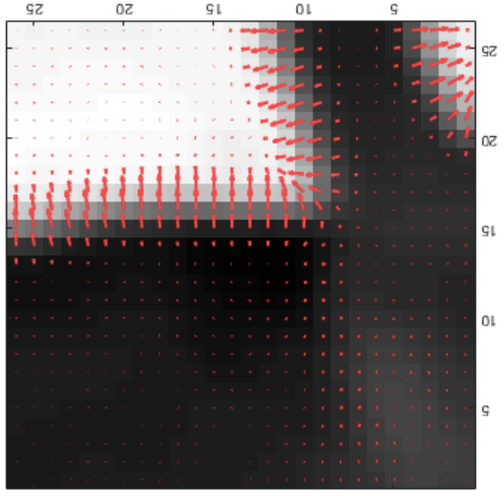


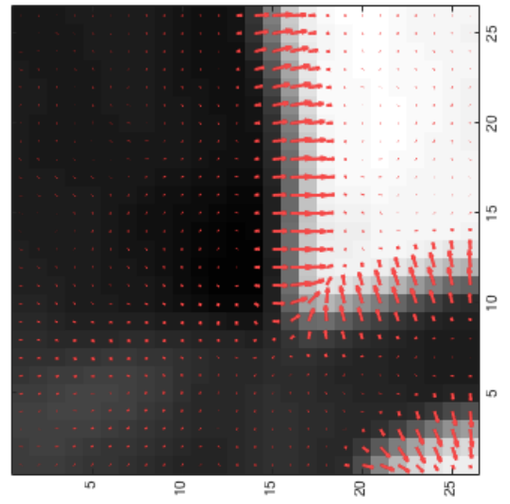


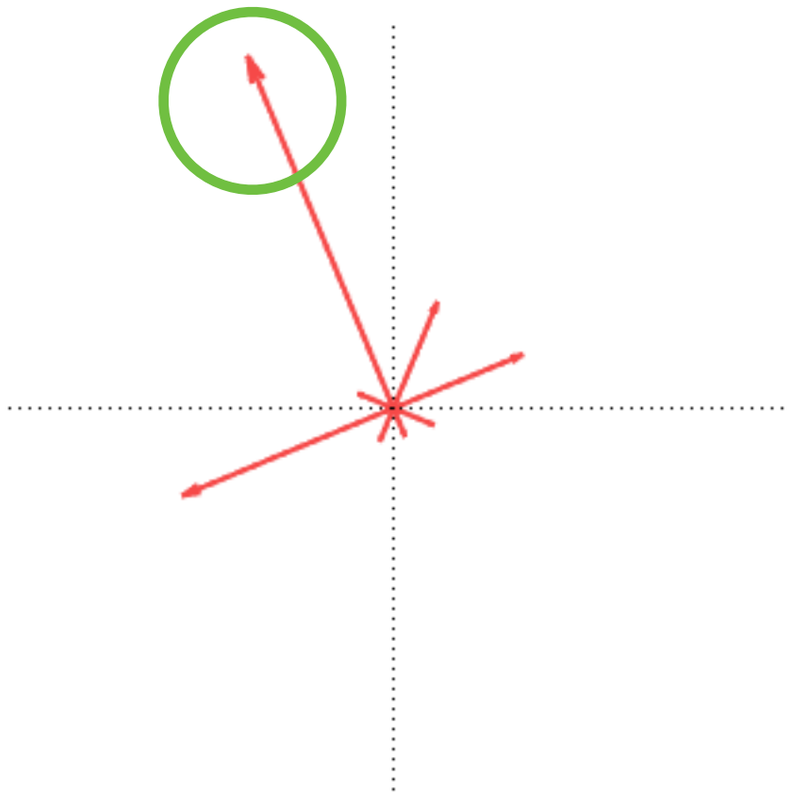


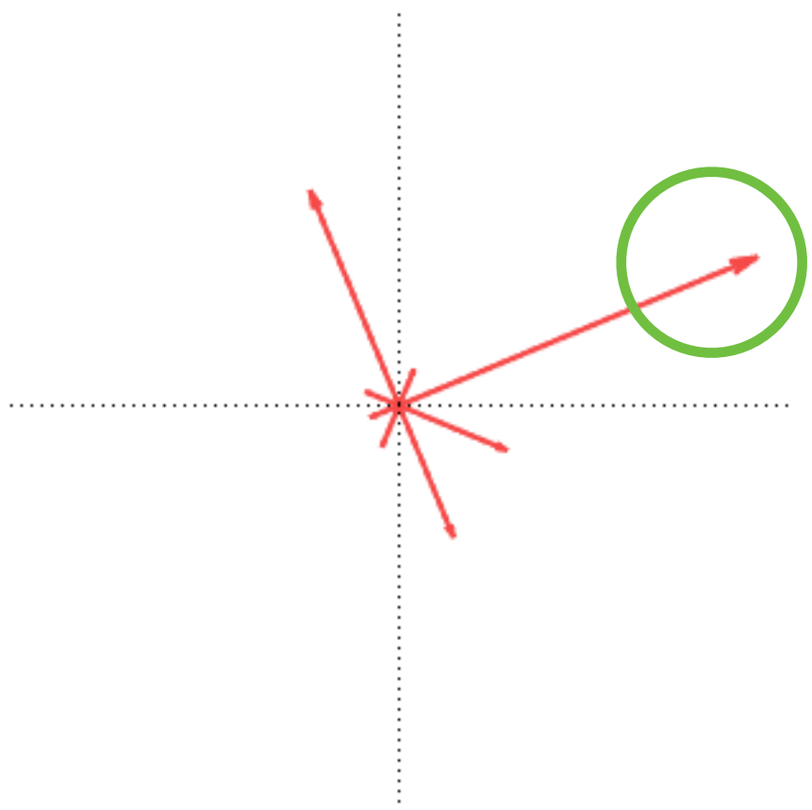


Orientation Assignment

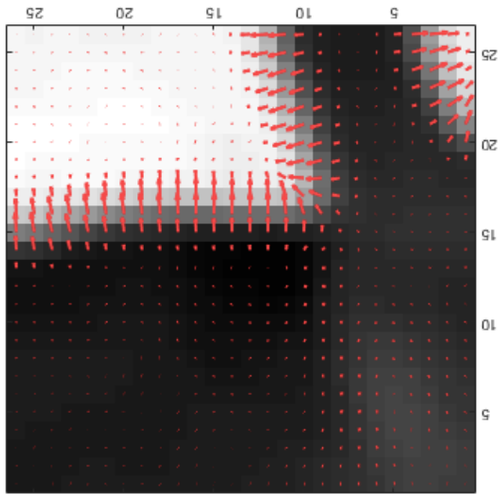


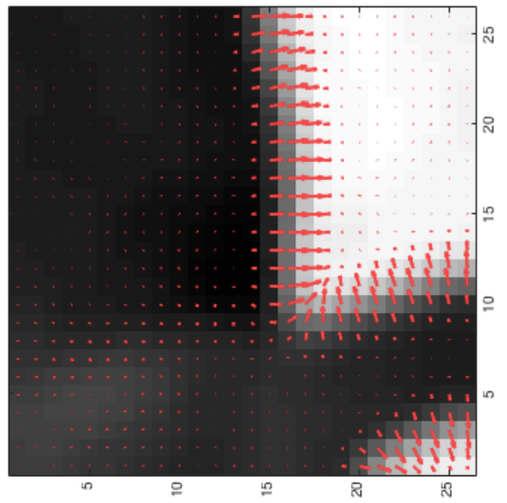


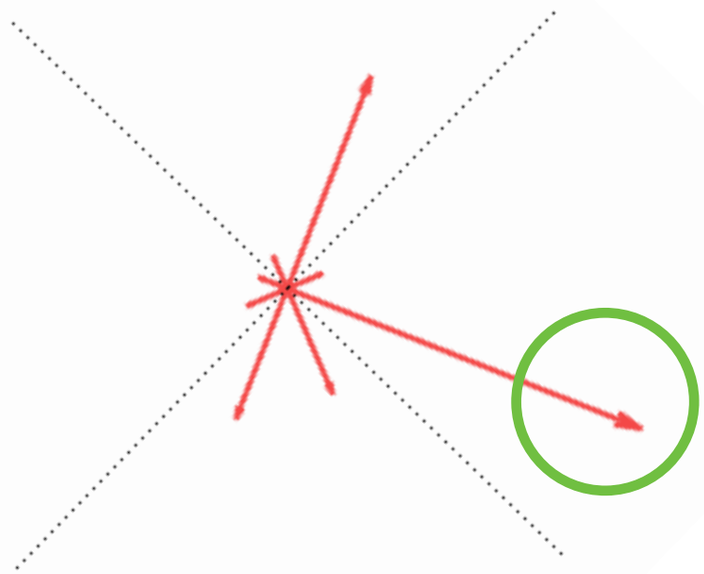


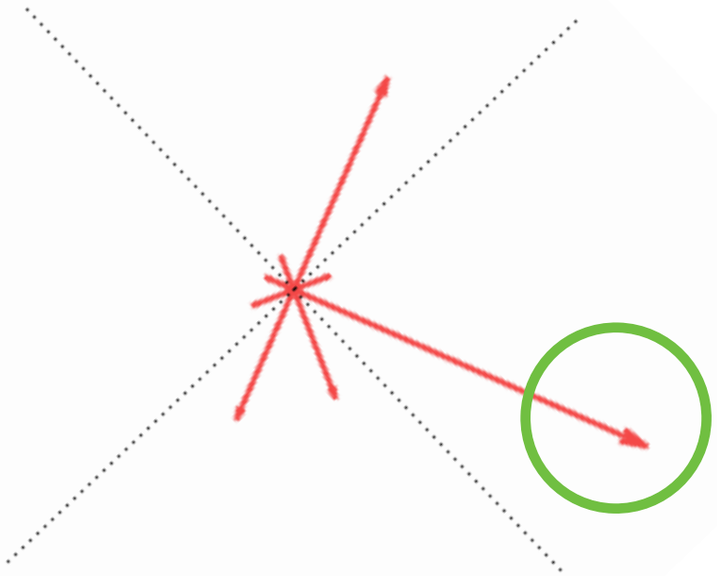


Orientation Assignment









The Perils of Rotation Invariance

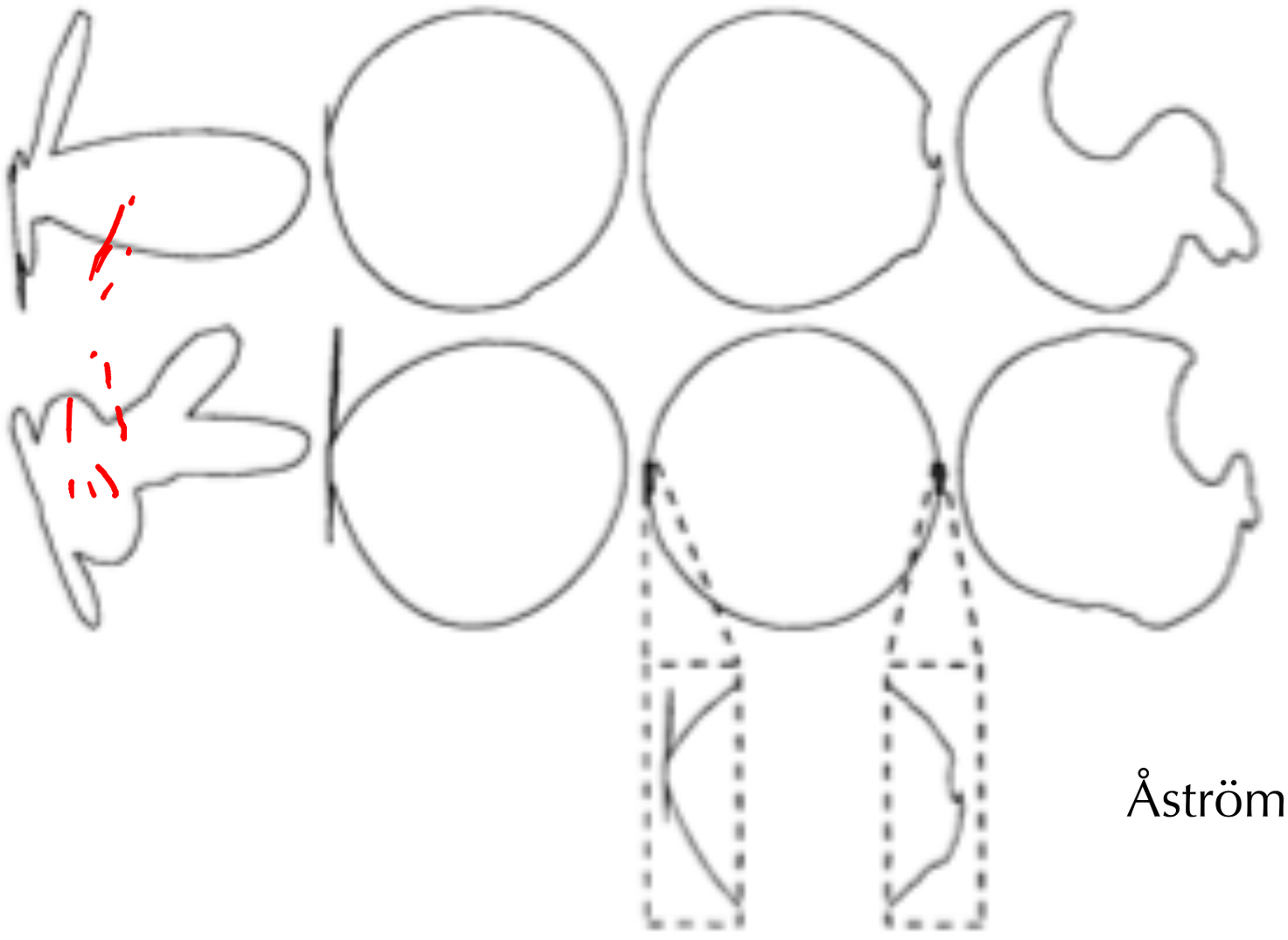
The Perils of Rotation Invariance

Is Your Visual System Rotation Invariant?

Is Your Visual System Rotation Invariant?

Thatcher illusion

Invariance vs. Robustness

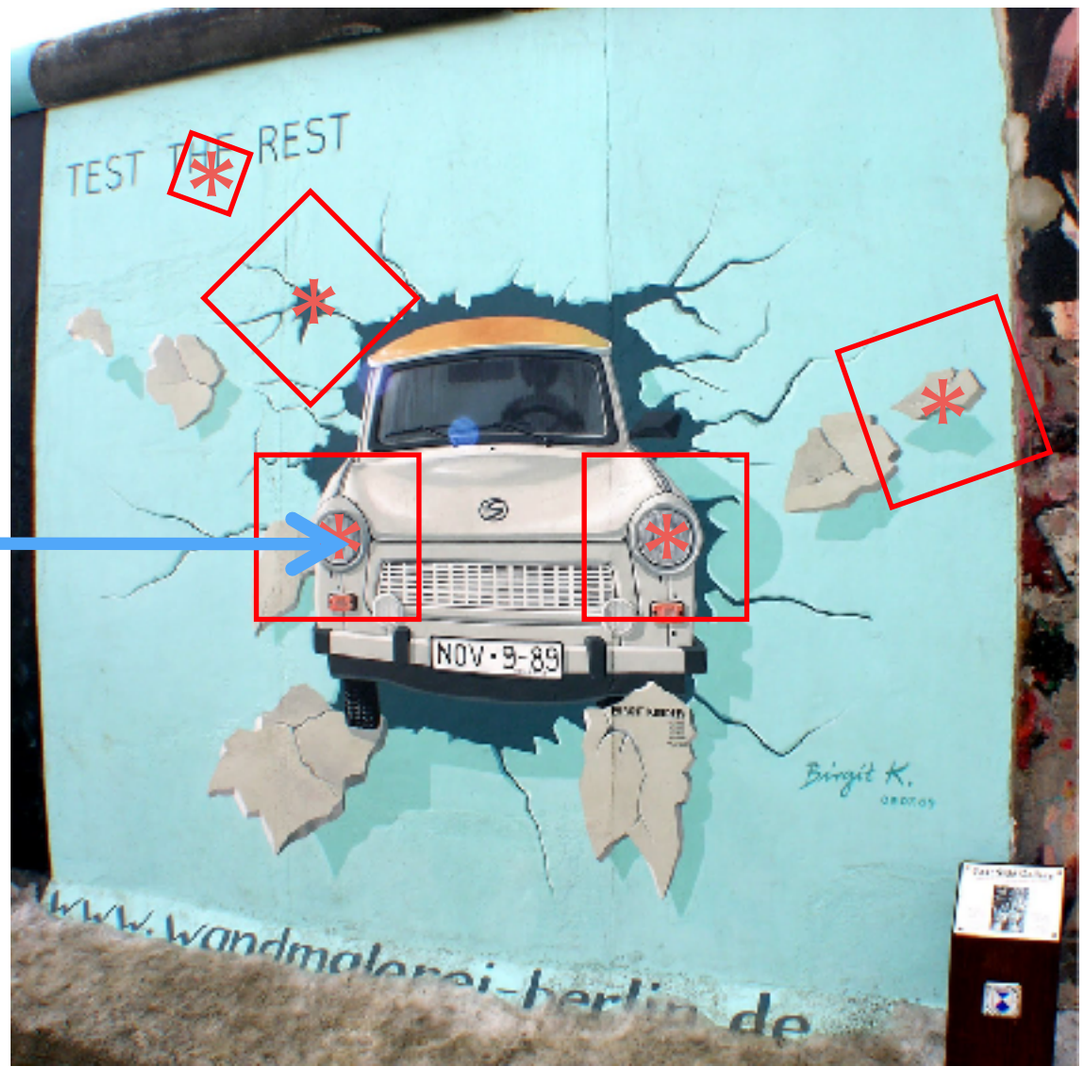


Åström-95

Invariance = no influence on result

Robustness = we can deal with a moderate amount

Scale-Invariant Feature Transform (SIFT) Features

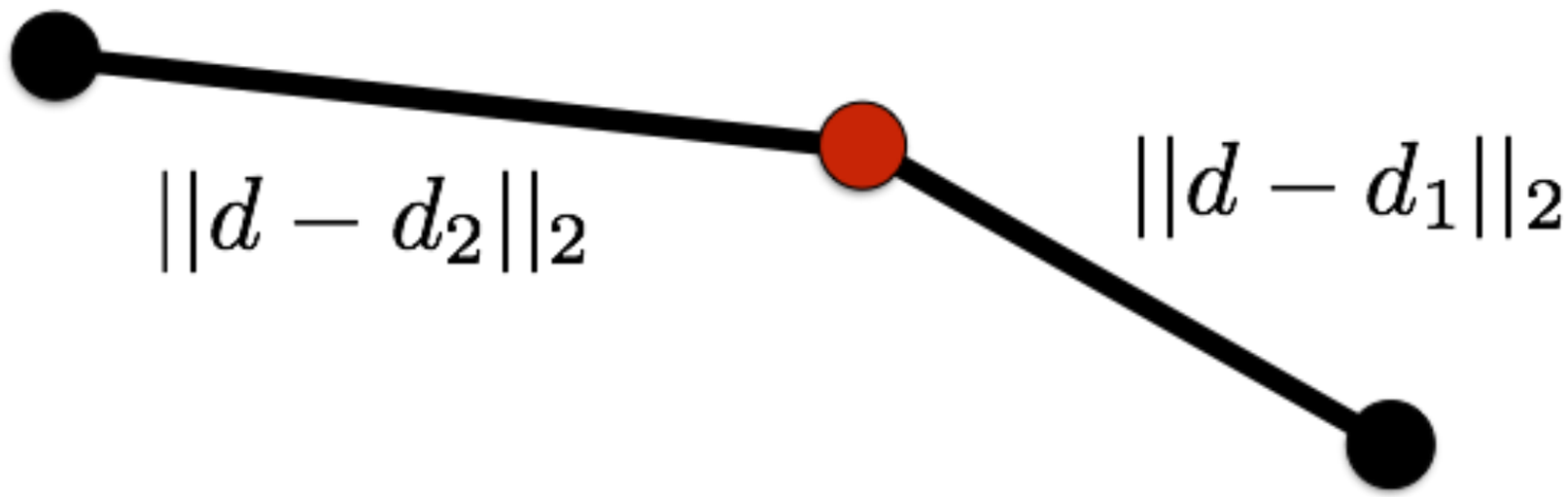


Use descriptor similarity (L_2 distance) to find corresponding points in different images

L_2 Similar = Correct?

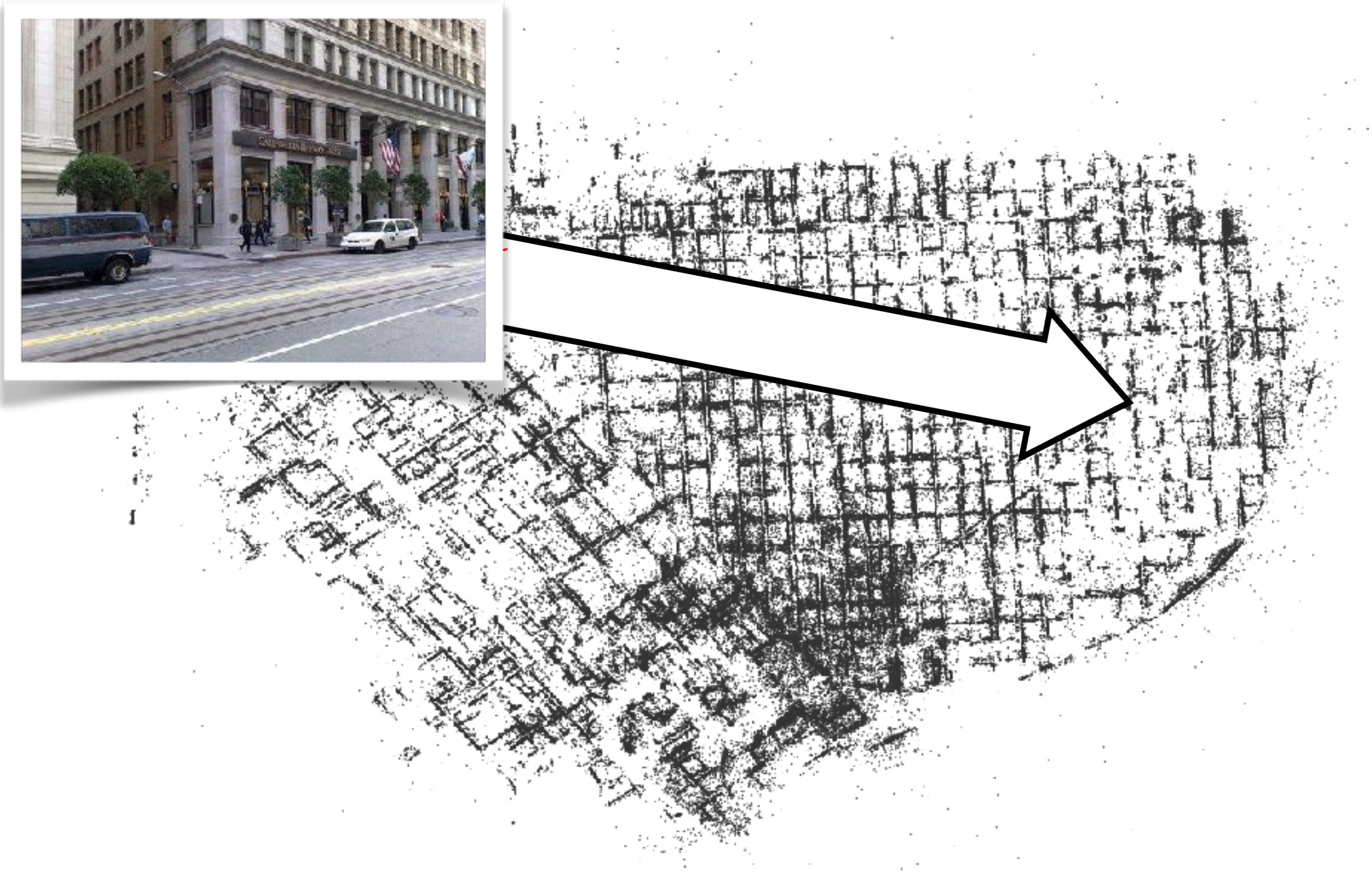
query:

Lowe's Ratio Test



Accept match with 1st neighbor if $\dfrac{||d - d_1||_2}{||d - d_2||_2} < 0.8$

City-Scale Localization



[Svärm, Enqvist, Kahl, Oskarsson, City-Scale Localization for Cameras with Known Vertical Direction, IJCV 2017 / CVPR 2014]
[Zeisl, Sattler, Pollefeys, Camera Pose Voting for Large-Scale Image-Based Localization, CVPR 2015]

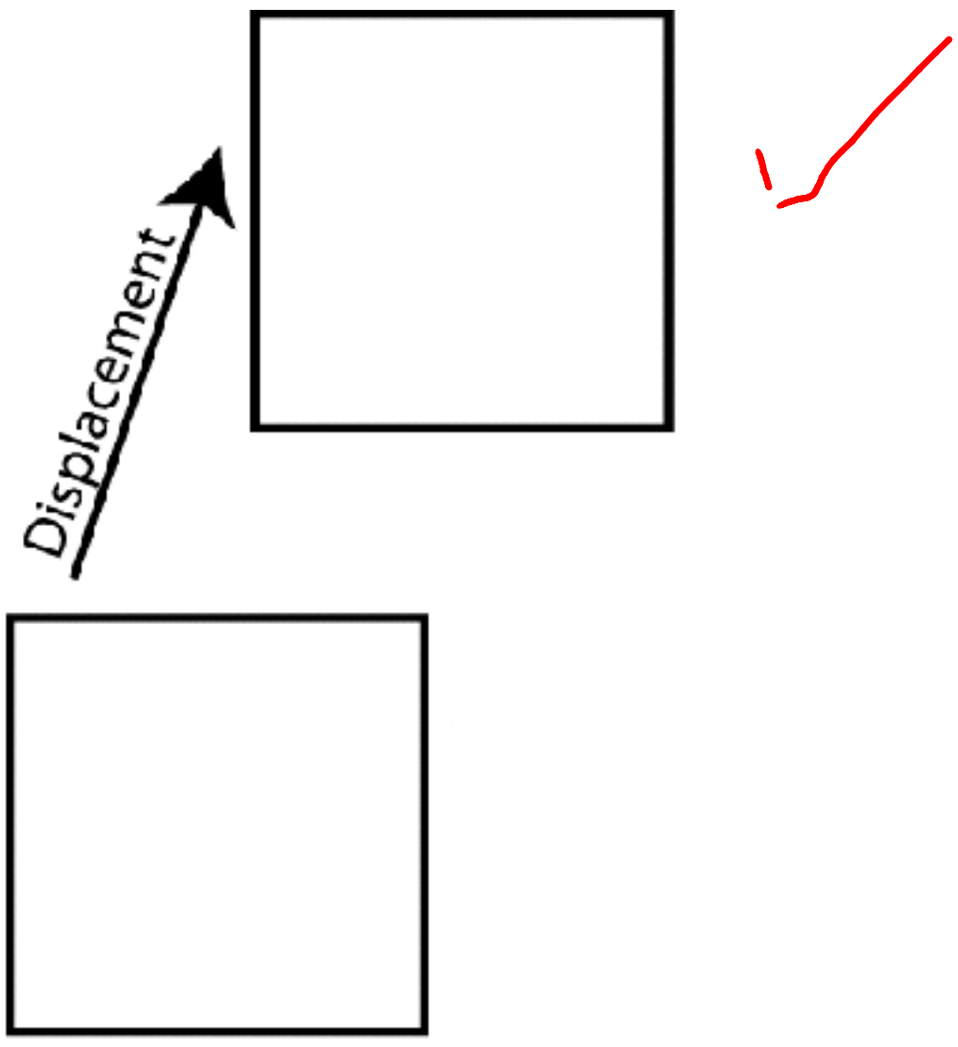
SIFT Features

- Skipped some details:
  - Down-weighting of pixels far away from center of spatial bin
  - Bilinear interpolation between histogram bins
  - Bilinear interpolation between spatial bins
  - Thresholding on histogram entries & re-normalization
  - For further details, read Lowe's 2004 IJCV paper

Today

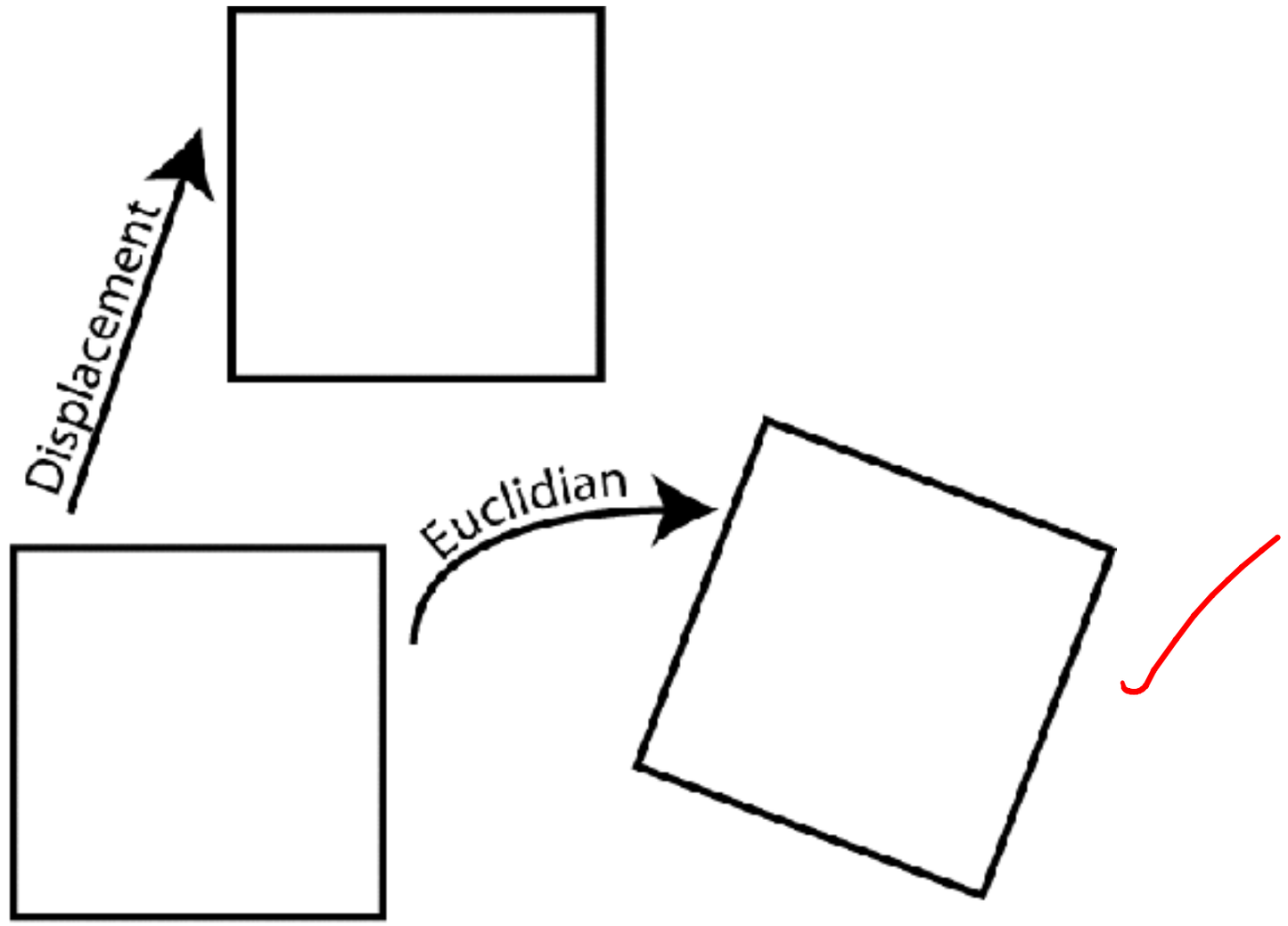
- Scale-Invariant Local Features
- Invariances*

Patch Transformations



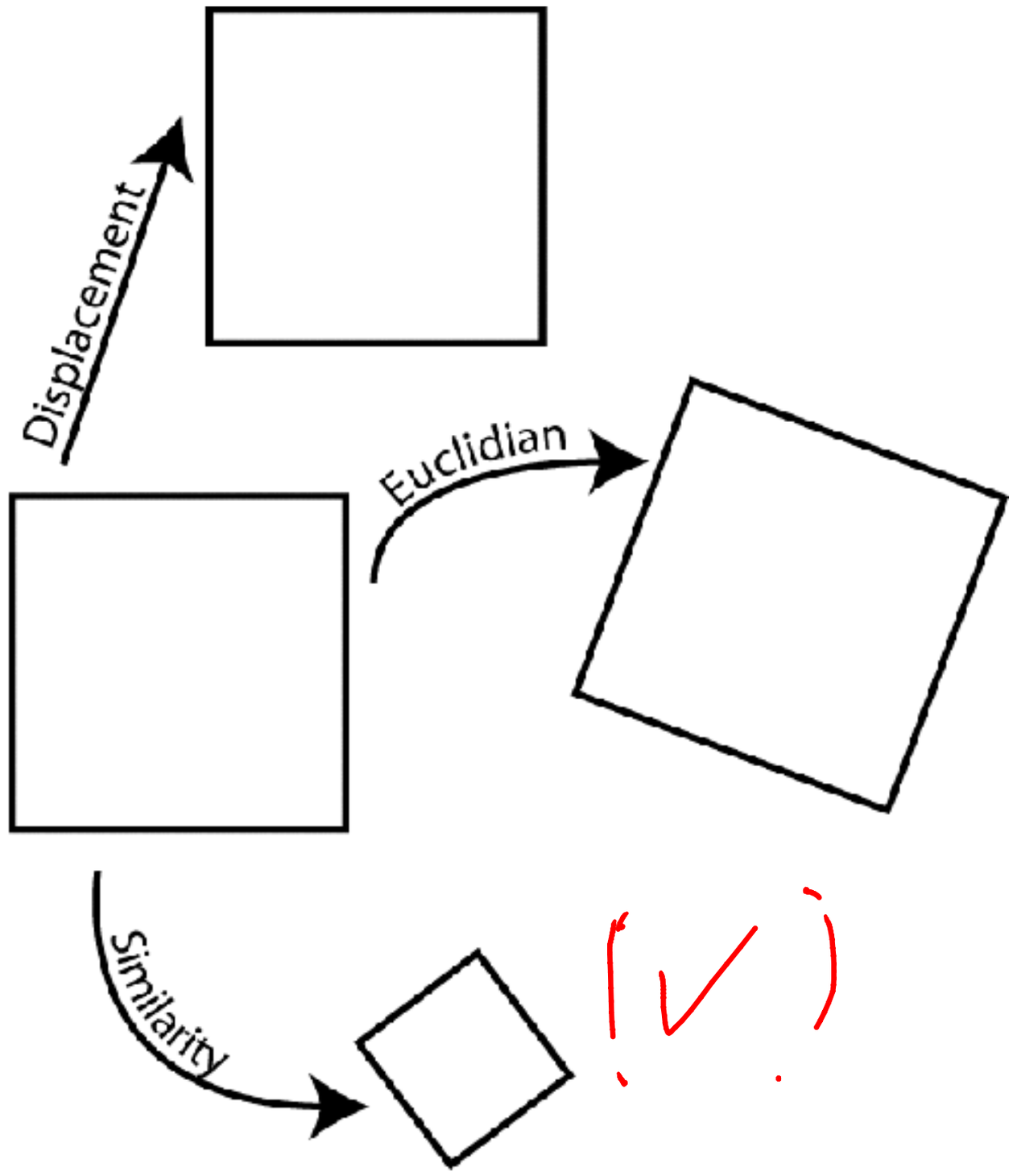
slide credit: Marc Pollefeys

Patch Transformations



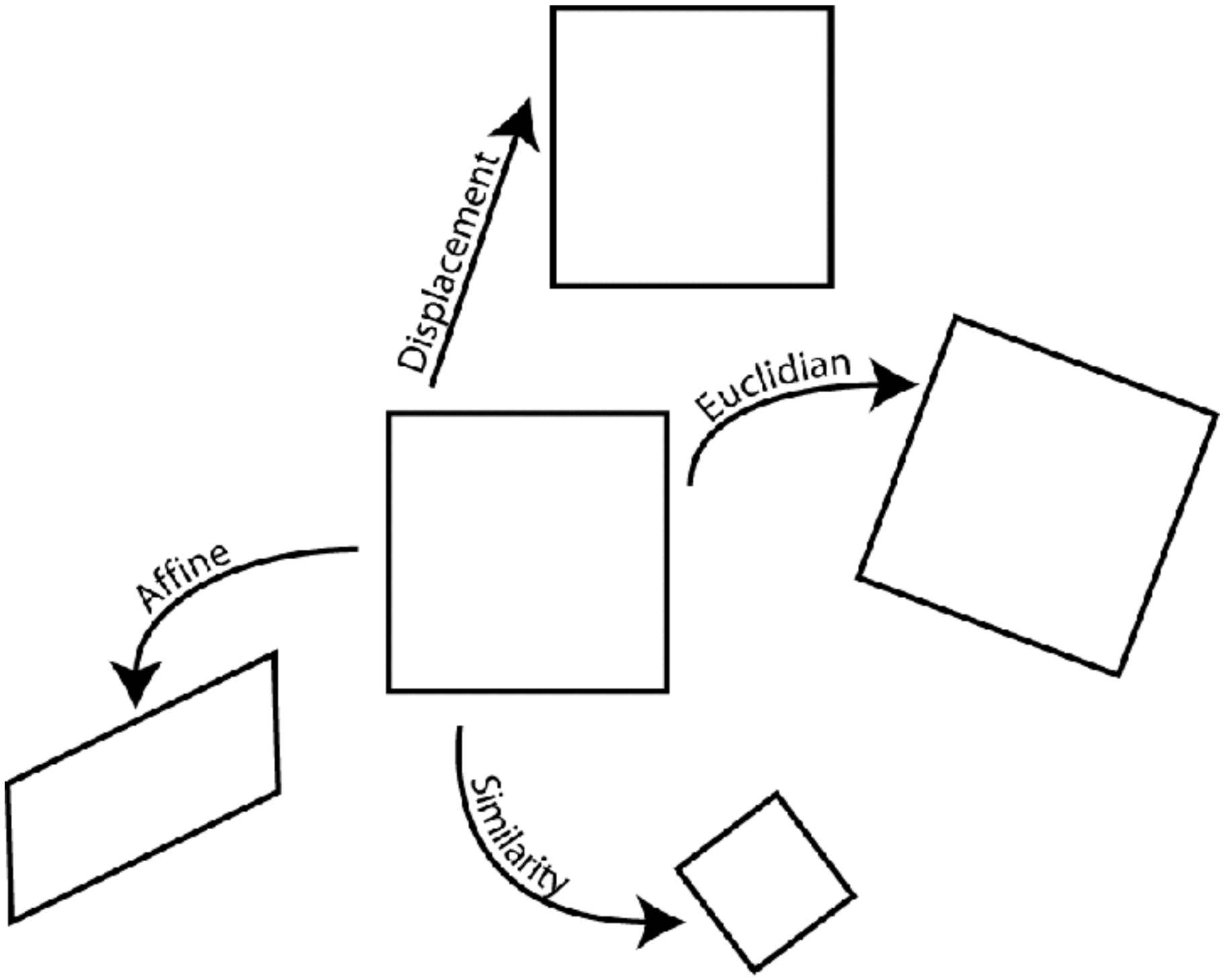
slide credit: Marc Pollefeys

Patch Transformations



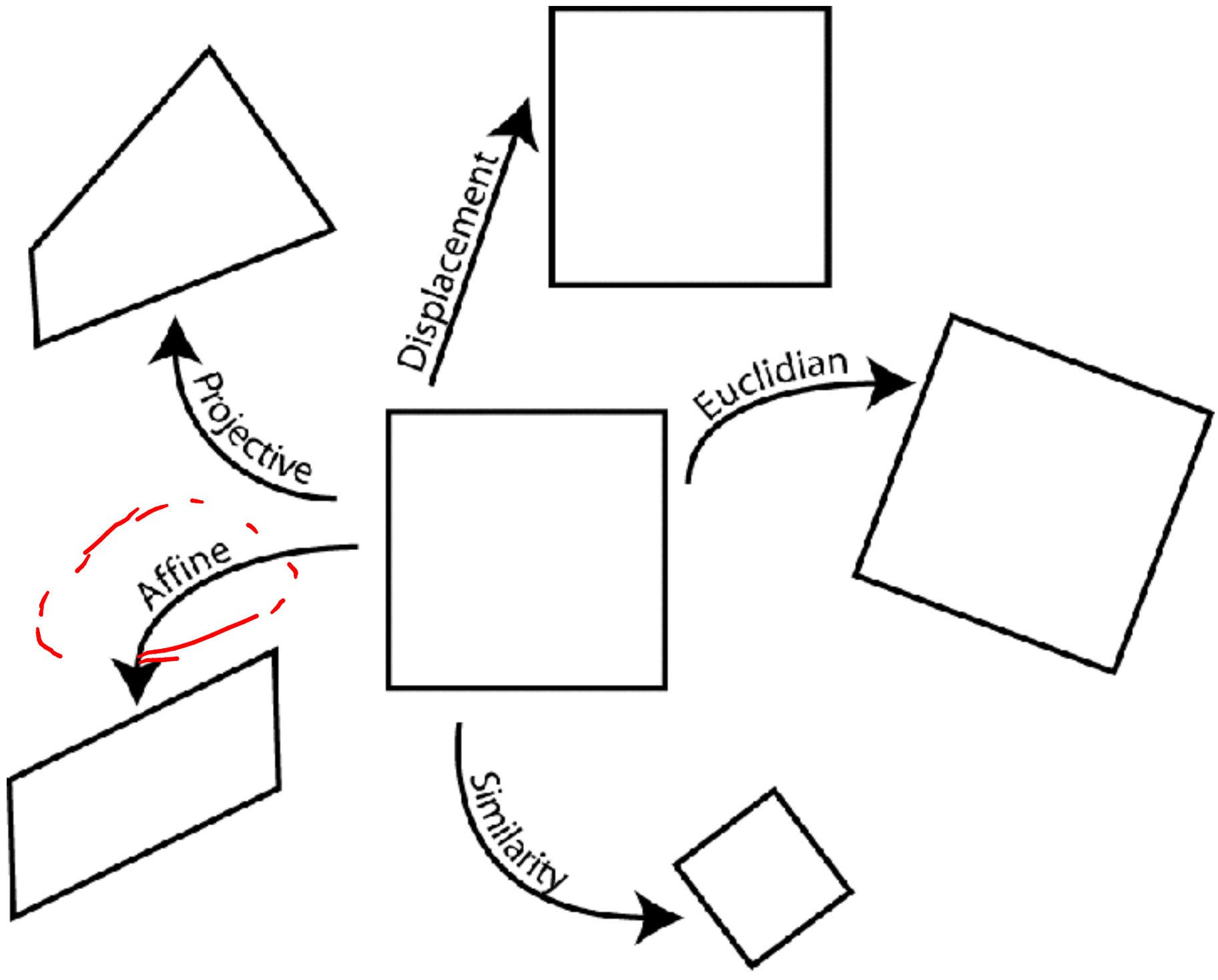
slide credit: Marc Pollefeys

Patch Transformations



slide credit: Marc Pollefeys

Patch Transformations



slide credit: Marc Pollefeys

Projective Transformations

Affine transformations can locally approximate perspective transformations quite well

slide credit: Marc Pollefeys

Projective Transformations

Affine transformations can locally approximate perspective transformations quite well

slide credit: Marc Pollefeys

Projective Transformations

Affine transformations can locally approximate perspective transformations quite well

slide credit: Marc Pollefeys

Lessons Learned

- Main lessons from this lecture
  - SIFT feature detector: Difference of Gaussians over scale space
  - Extraction of oriented and scale-dependent patches
  - SIFT feature descriptor: Concatenation of gradient histograms

Lessons Learned

- Main lessons from this lecture
  - SIFT feature detector: Difference of Gaussians over scale space
  - Extraction of oriented and scale-dependent patches
  - SIFT feature descriptor: Concatenation of gradient histograms
- Next lecture: Learning classifiers