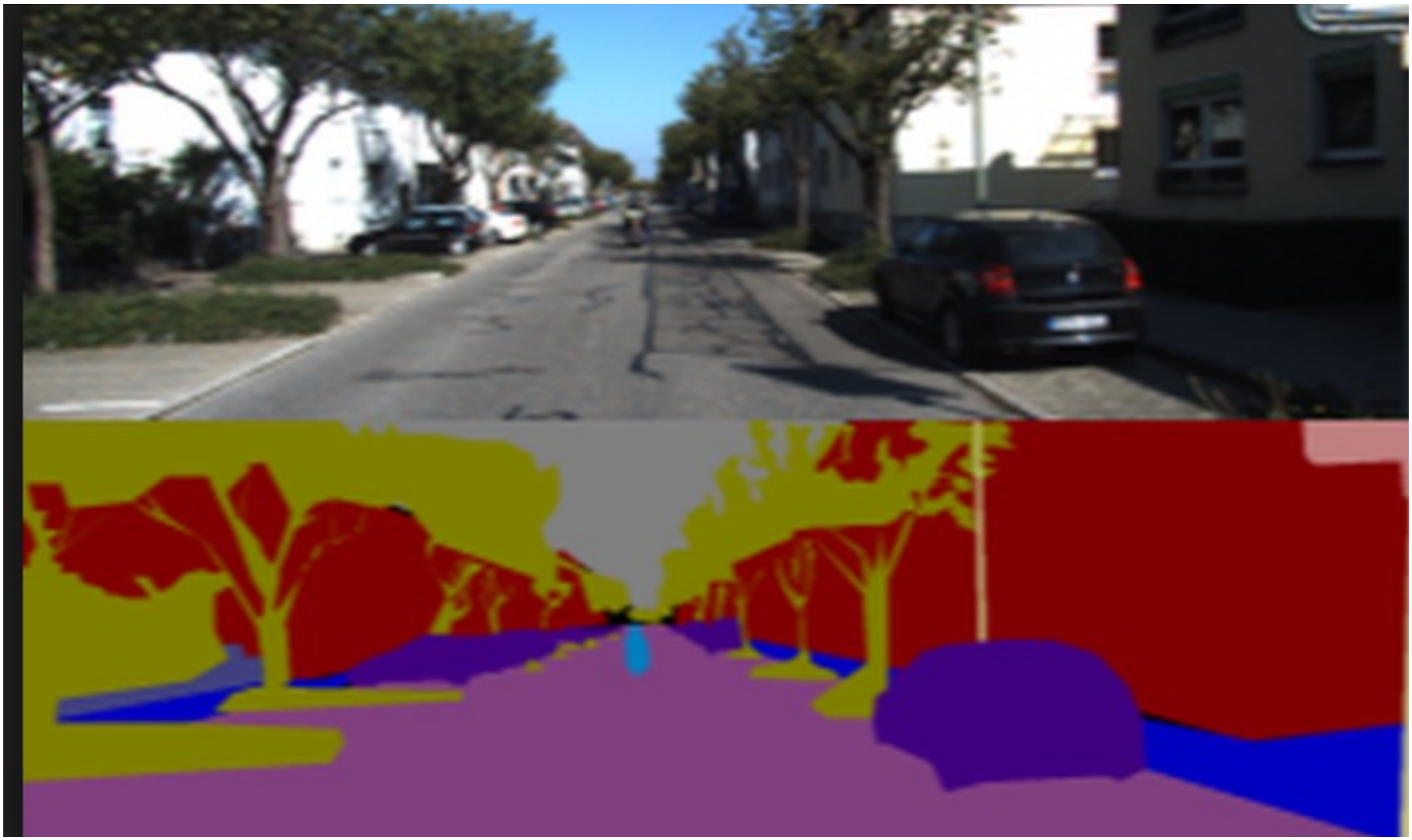
From Pixels to Edges and Keypoints

Kostas Daniilidis and Jianbo Shi

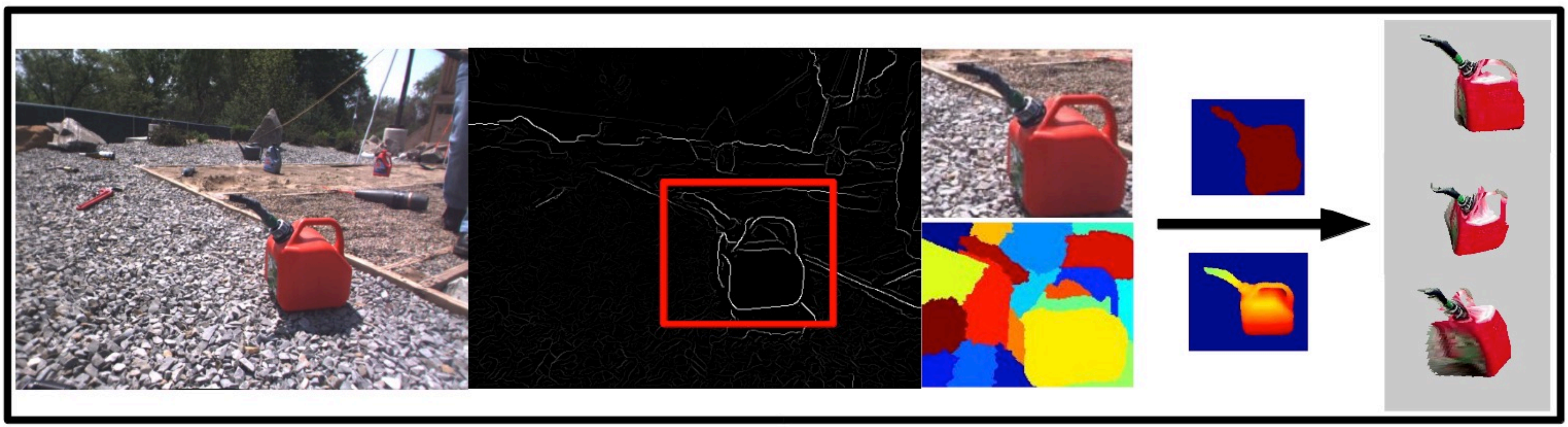
Big Picture


image
pixels → RGB
convolutions plus some non linear ops
texture
edges
lines, curves
keypoints
semantics
labels for boxes, pixels
correspondence
grouping, RANSAC
geometry

Semantic segmentation

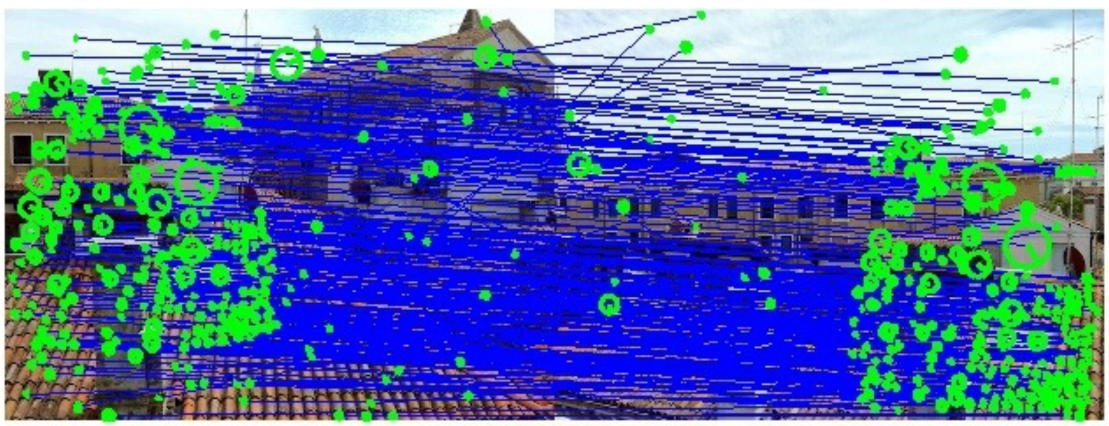


From segmentation to 3D pose



Using SIFT keypoints for image matching and SfM

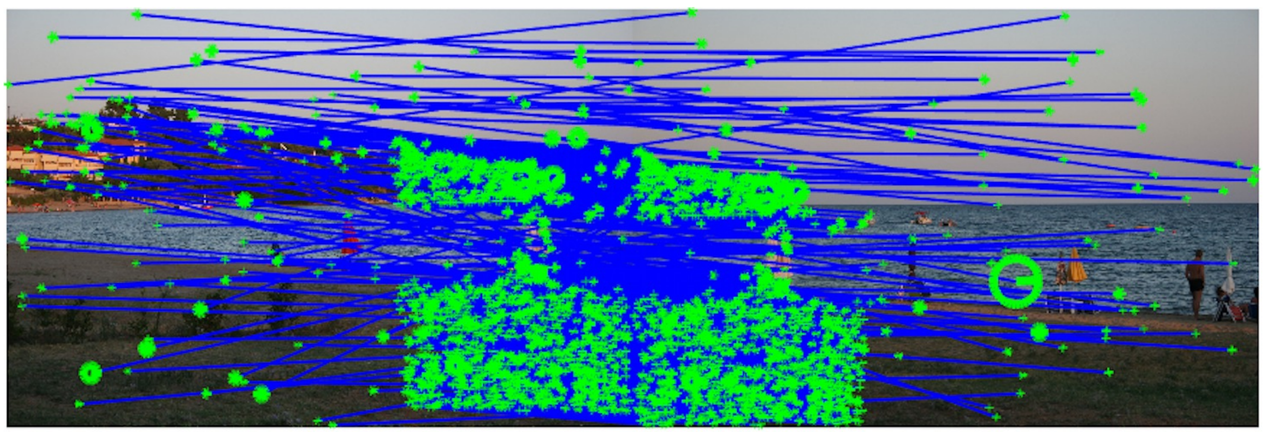
Original Image Pair



Matched features

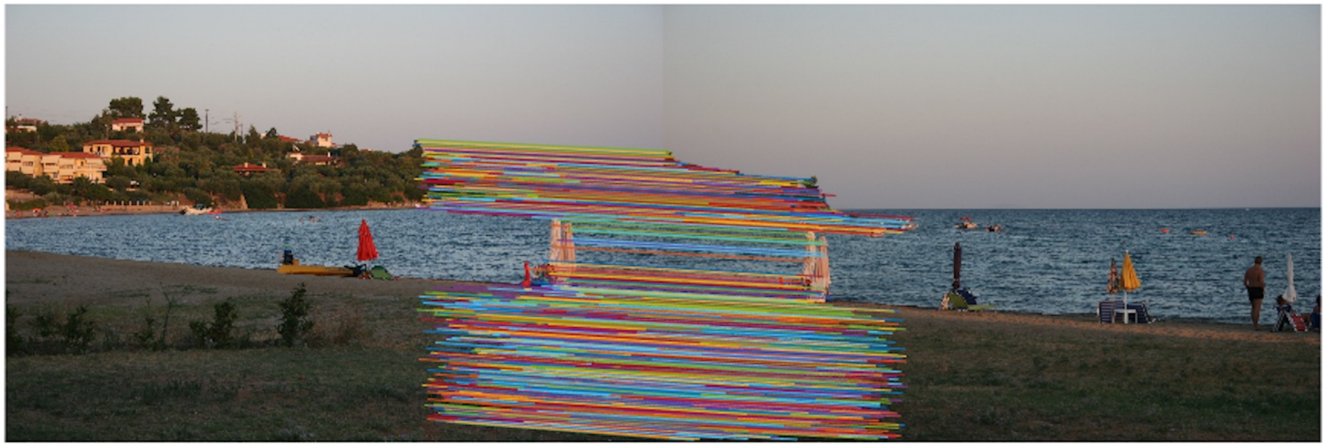
Create Image Mosaic

1. Get an image pair



2. Establish correspondences between matching features

Create Image Mosaic



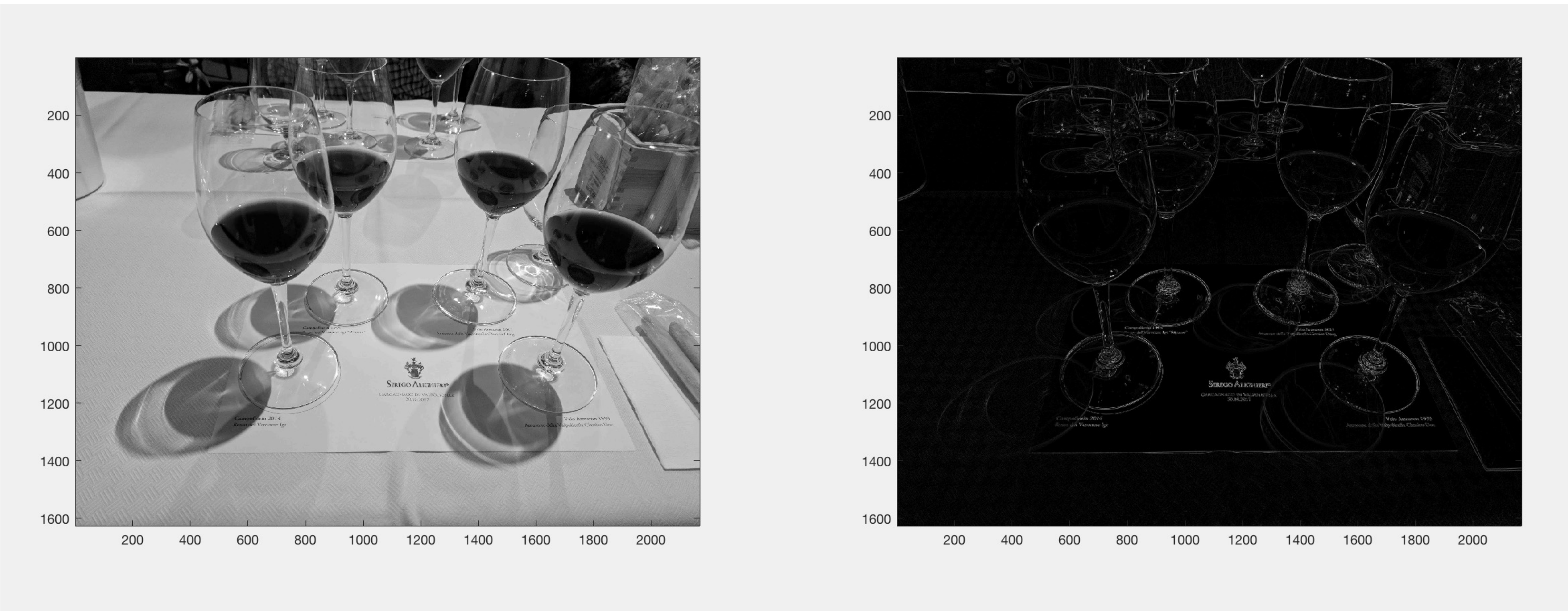
3. Keep only consistent matches (inliers)

4. Compute homography and warp 2nd image

Create Image Mosaic

5. Repeat to extend the mosaic

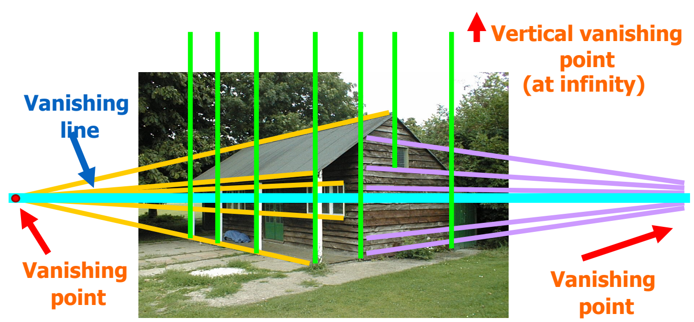
Edge detection



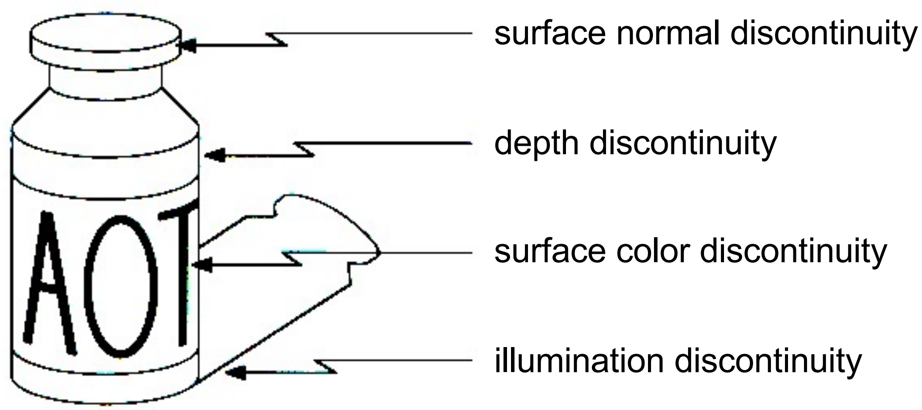
Why do we care about edges?

- Extract information, recognize objects

- Recover geometry and viewpoint



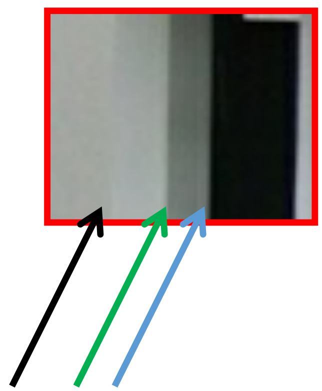
Origin of Edges



- Edges are caused by a variety of factors

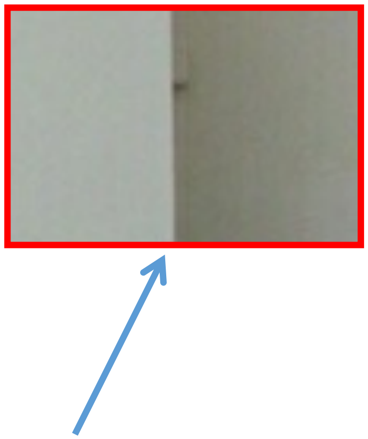
Source: Steve Seitz

Closeup of edges



Source: D. Hoiem

Closeup of edges



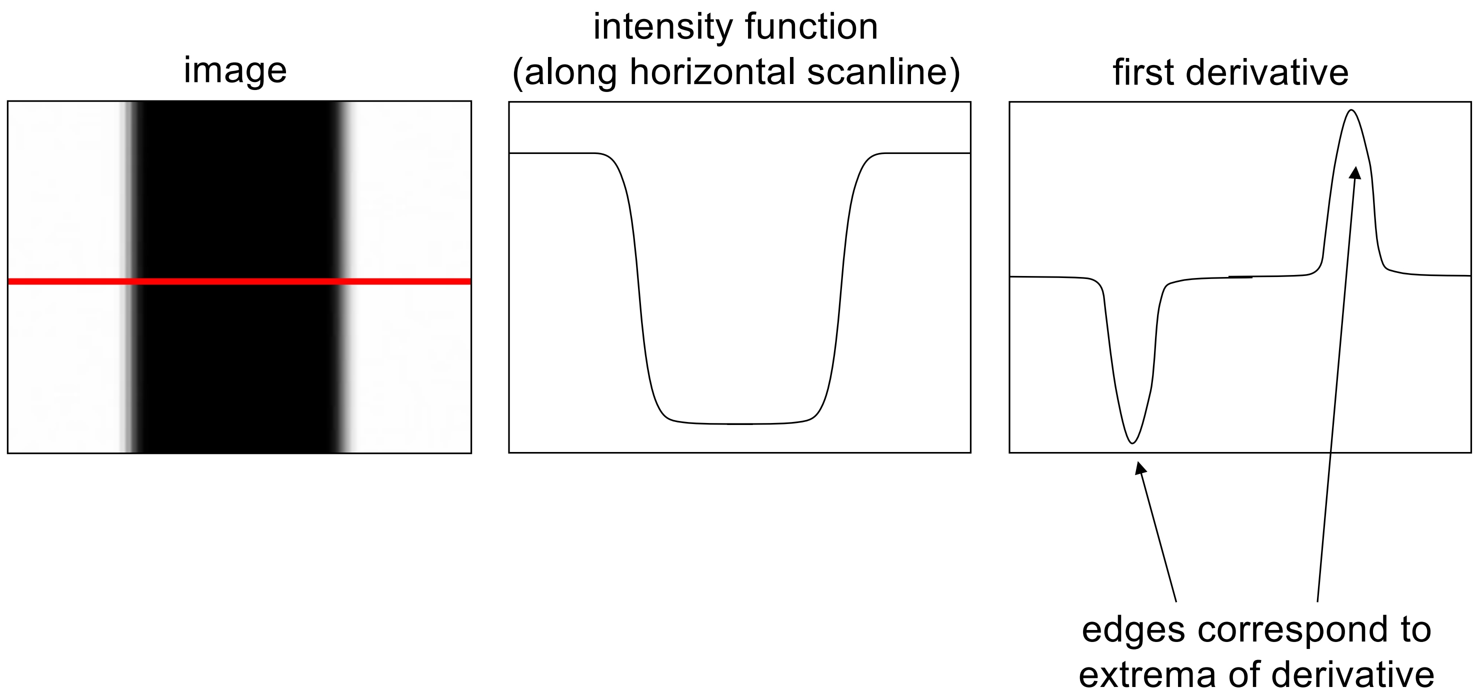
Source: D. Hoiem

Closeup of edges

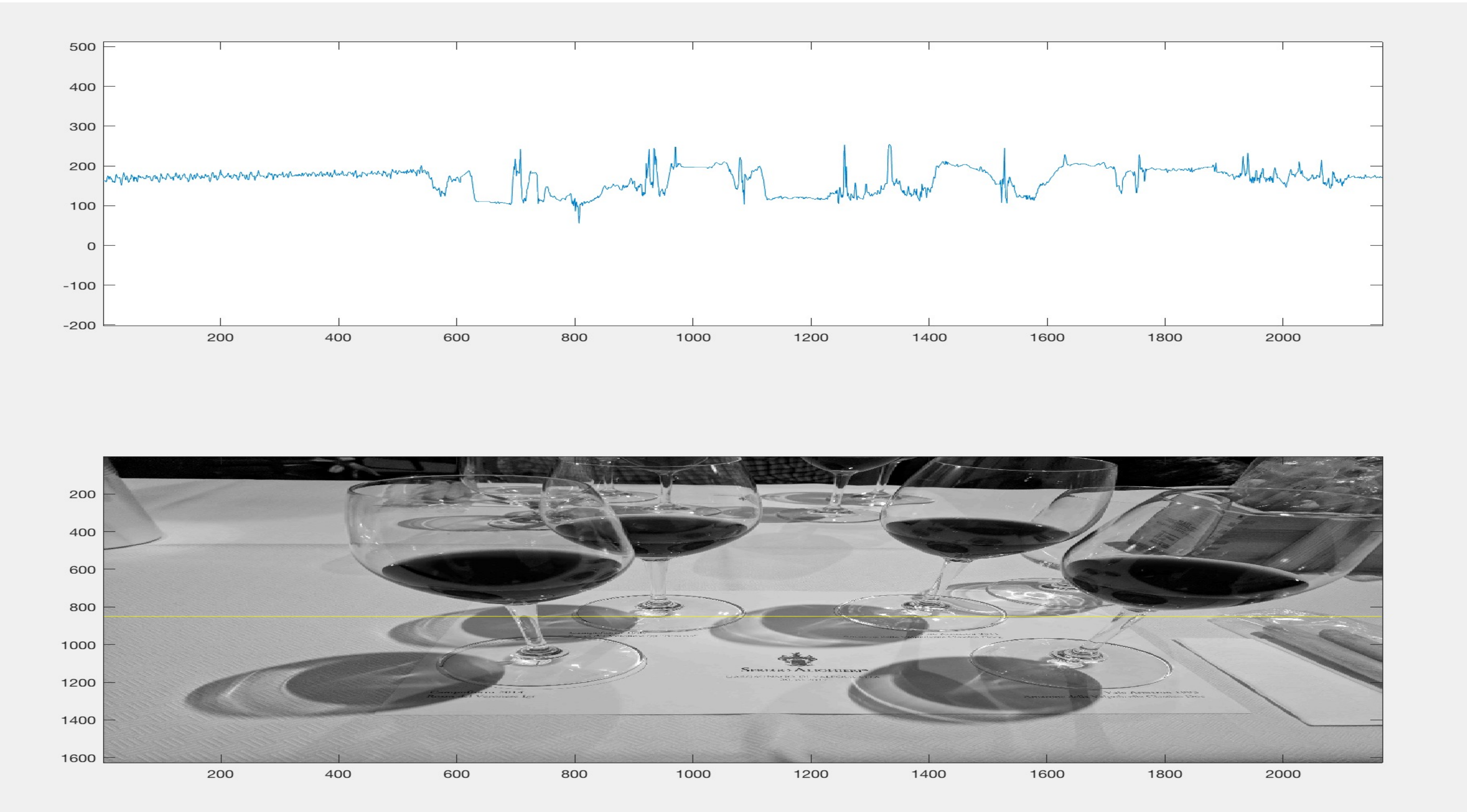
Source: D. Hoiem

Characterizing edges

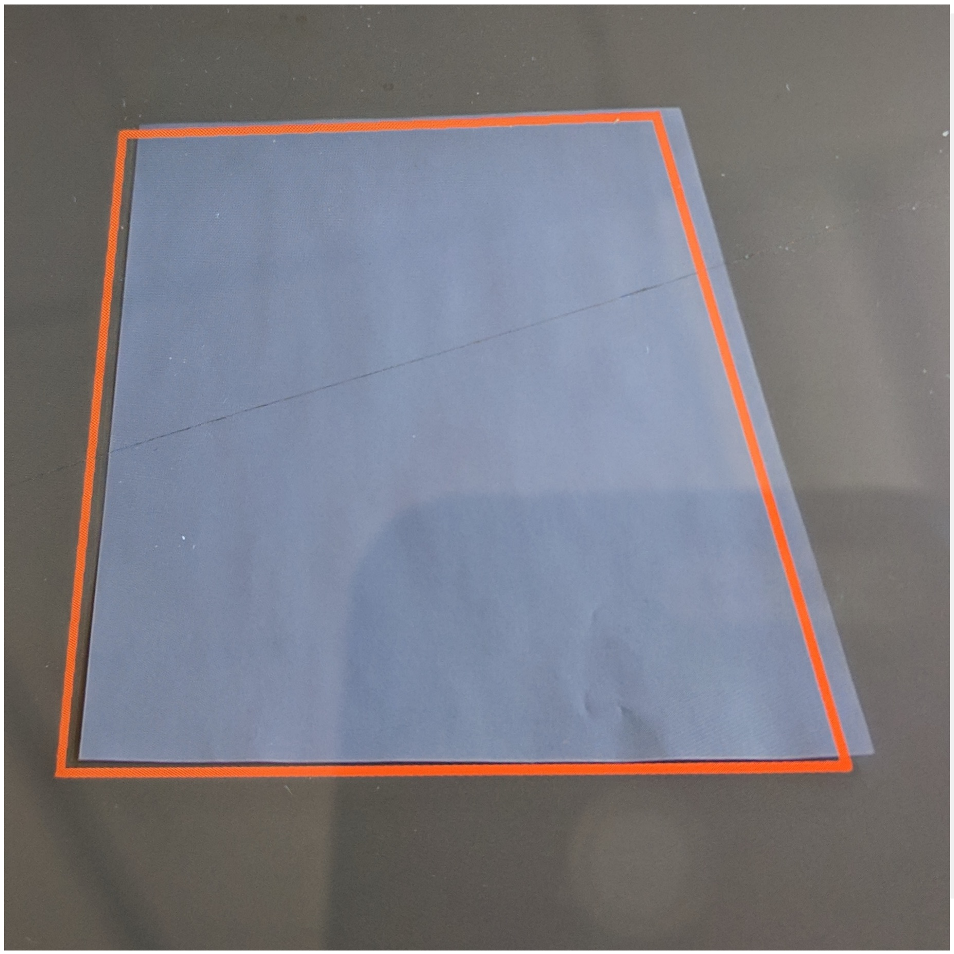
- An edge is a place of rapid change in the image intensity function

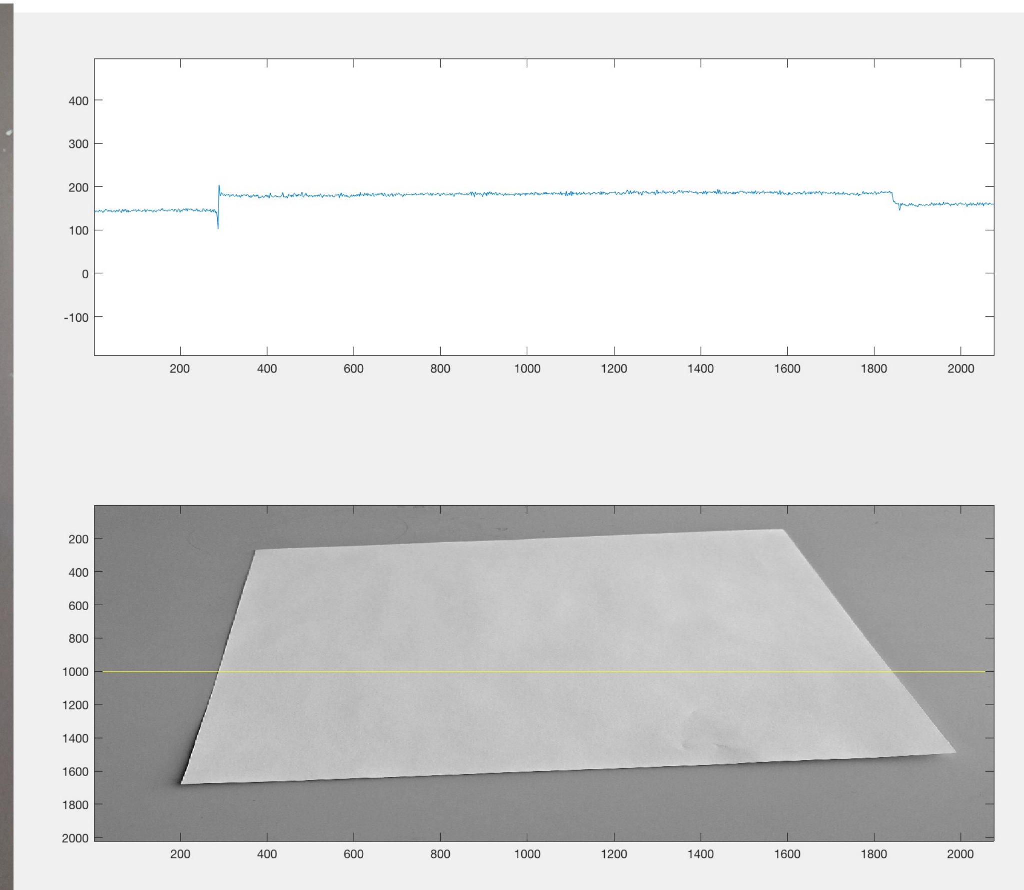


What is an edge?

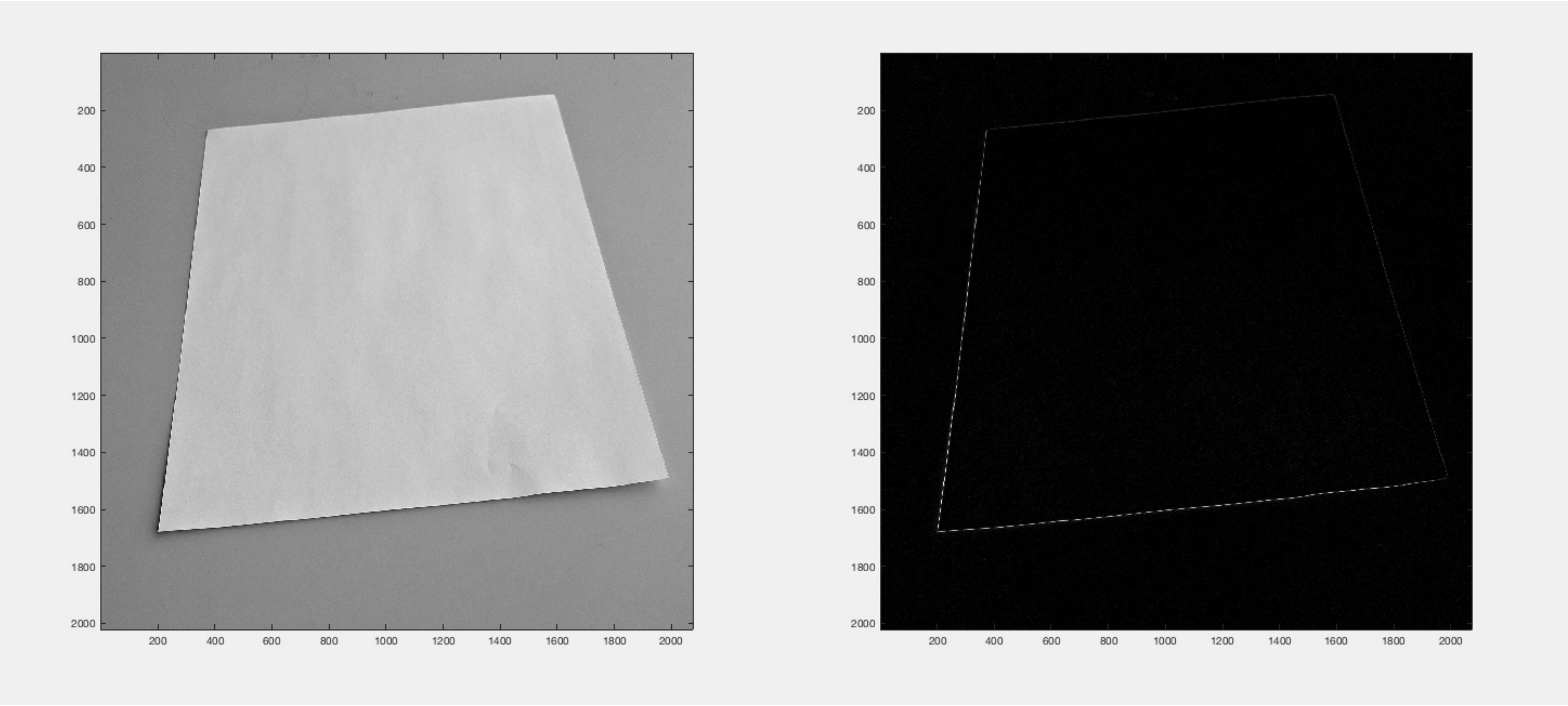


We might want to find just lines….

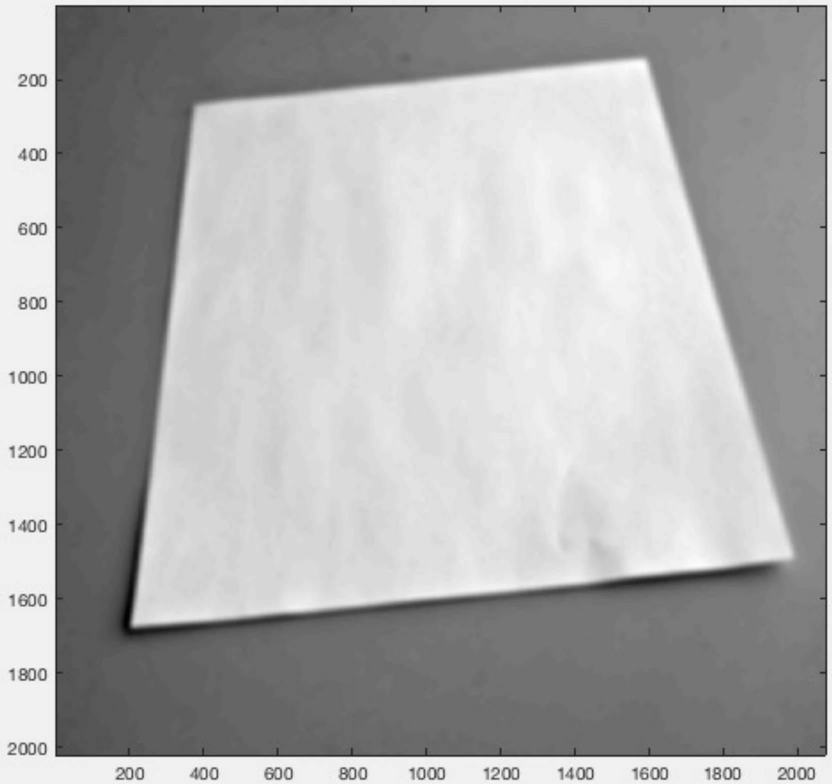


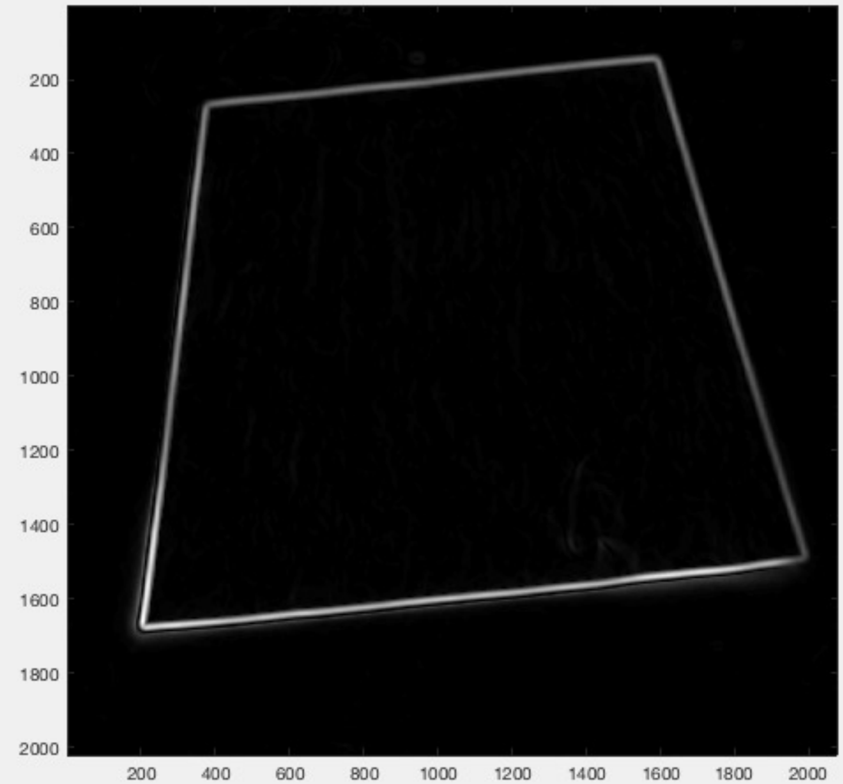


```
[Gx,Gy] = imgradientxy(I,'sobel'); %% plot magnitude
```

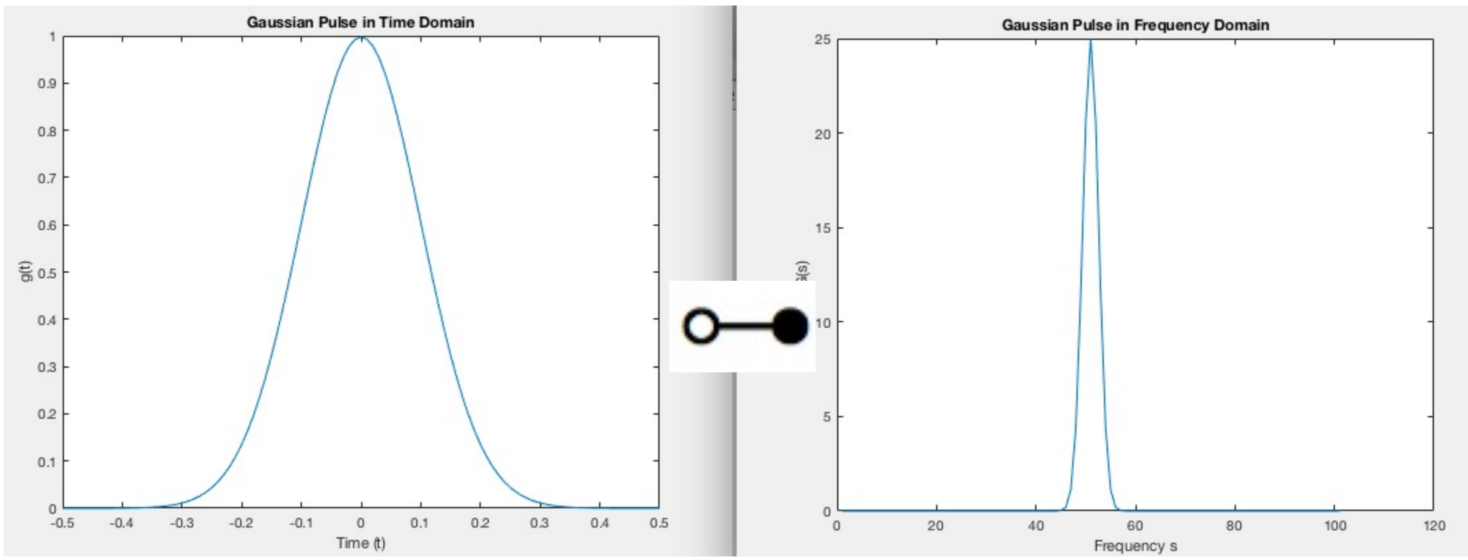


We fight aliasing by smoothing first (Shannon theorem)



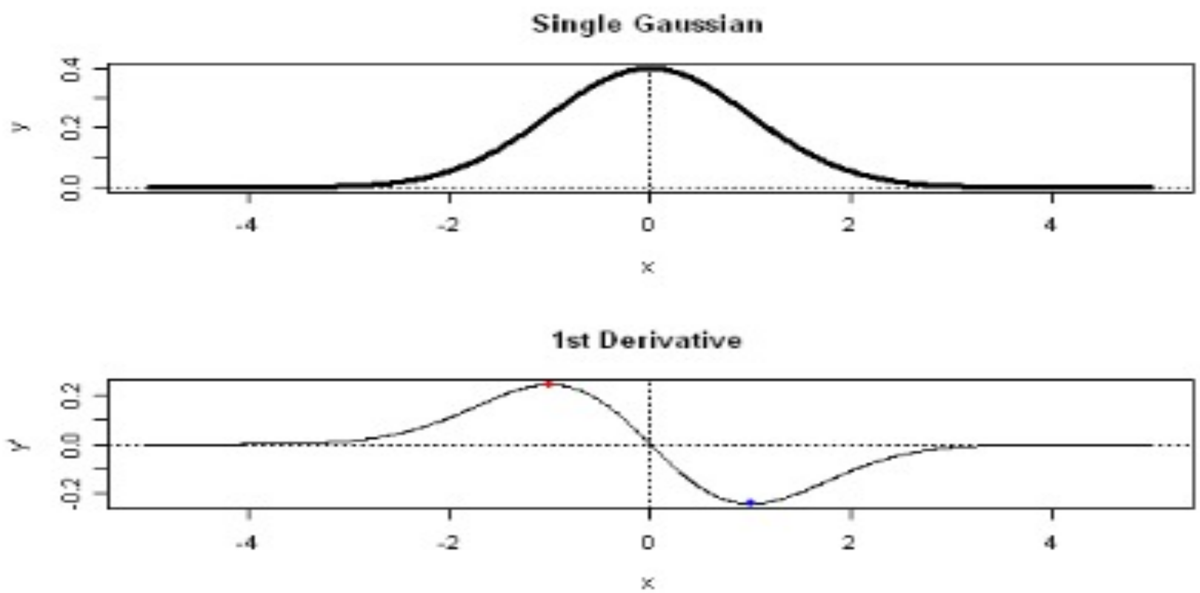


Fourier Transform of the Gaussian Function

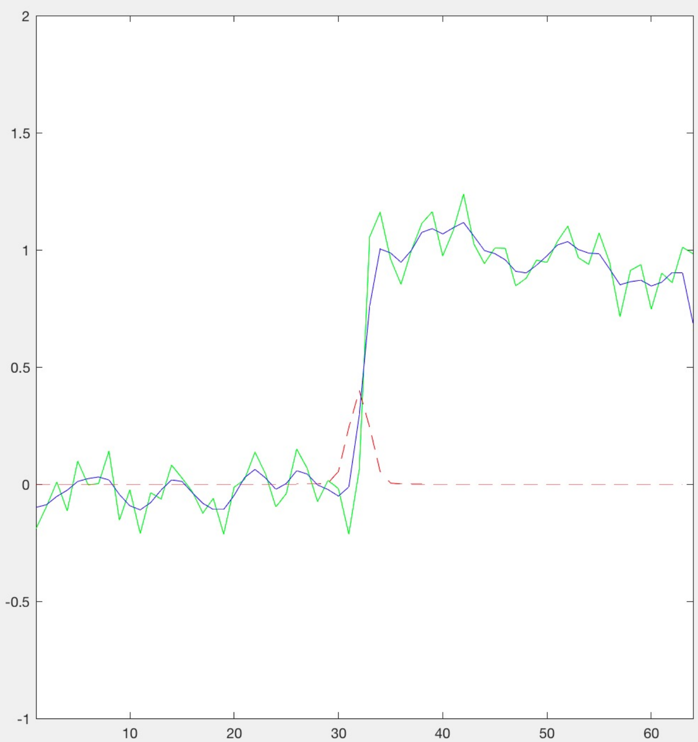


$$f(t) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{t^2}{2\sigma^2}} \quad \circ\!\!-\!\!\bullet \quad F(s) = \frac{1}{\sigma\sqrt{2\pi}} e^{-2\pi^2\sigma^2 s^2}$$

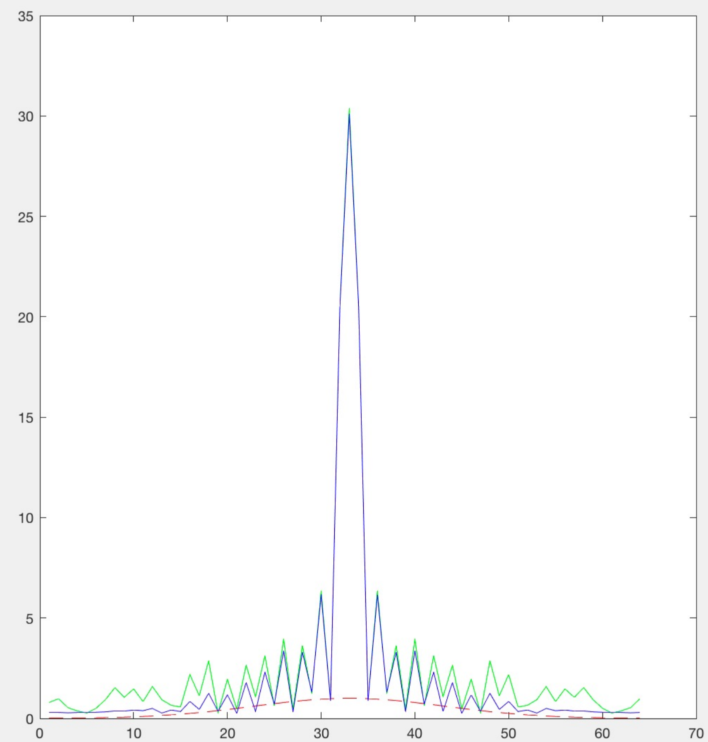
Smoothing an image and then taking derivative is the same as
convolving with first derivative of Gaussian



Smoothing an edge with a Gaussian

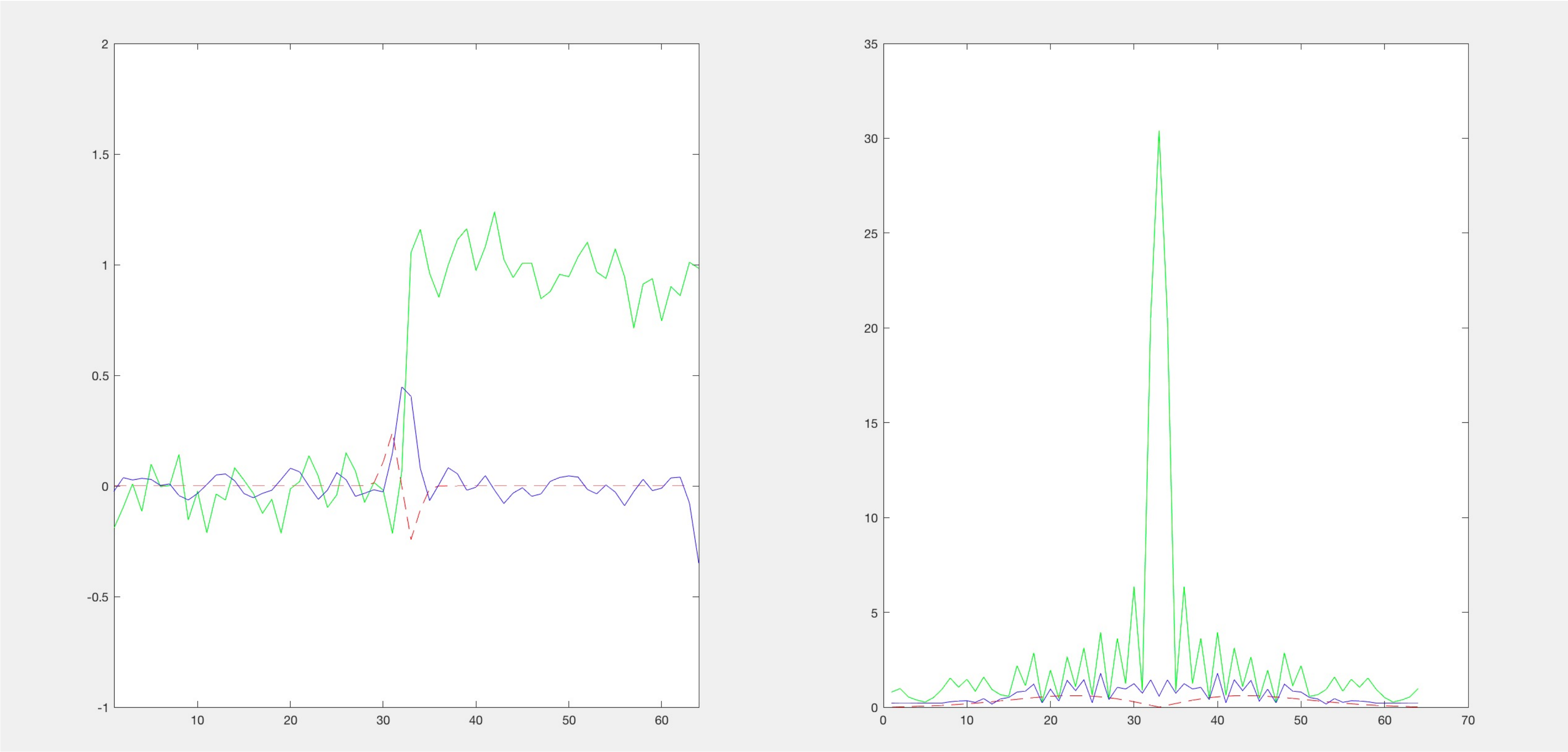


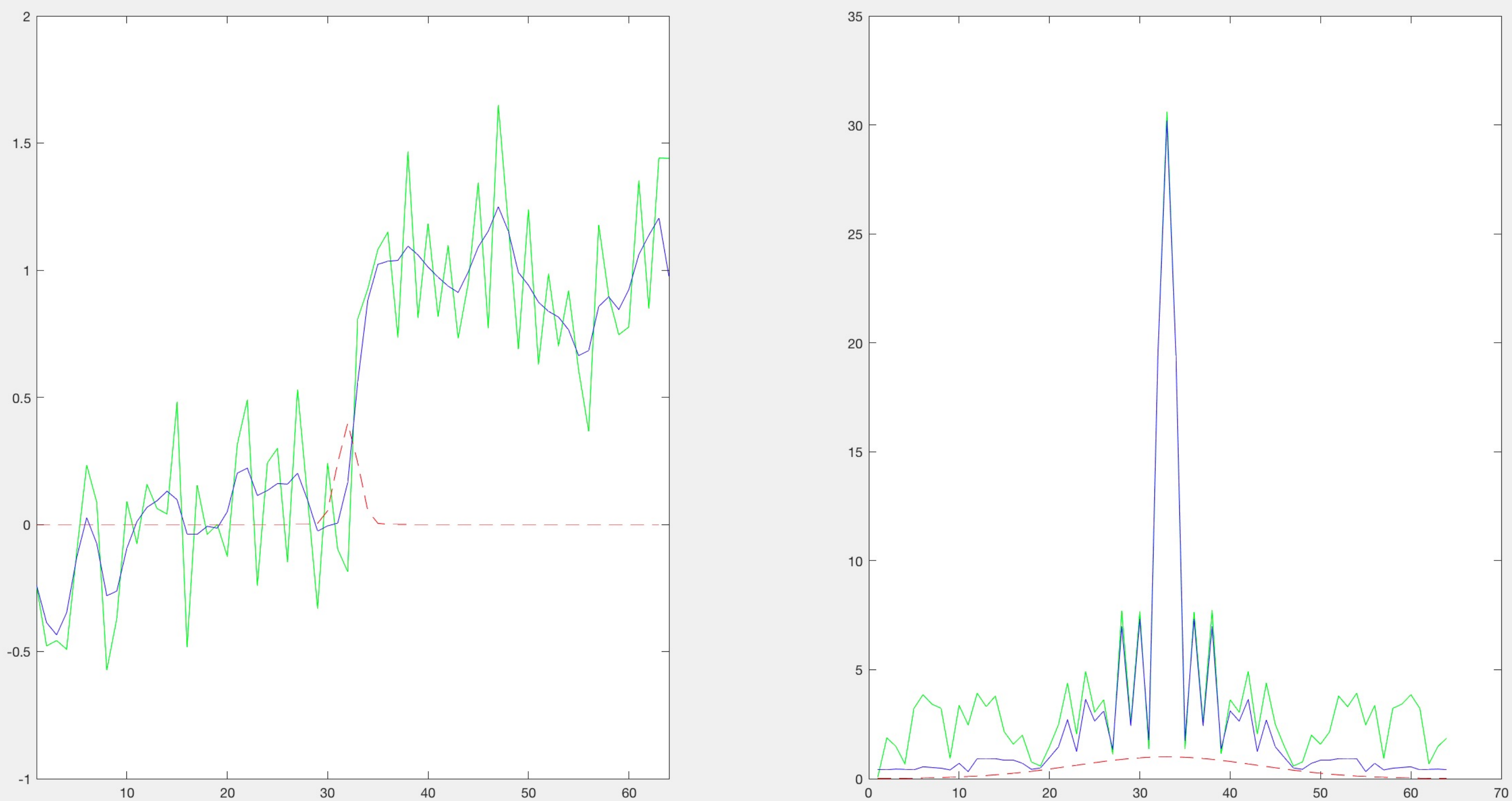
Red: filter, green: original signal, blue: signal after filtering



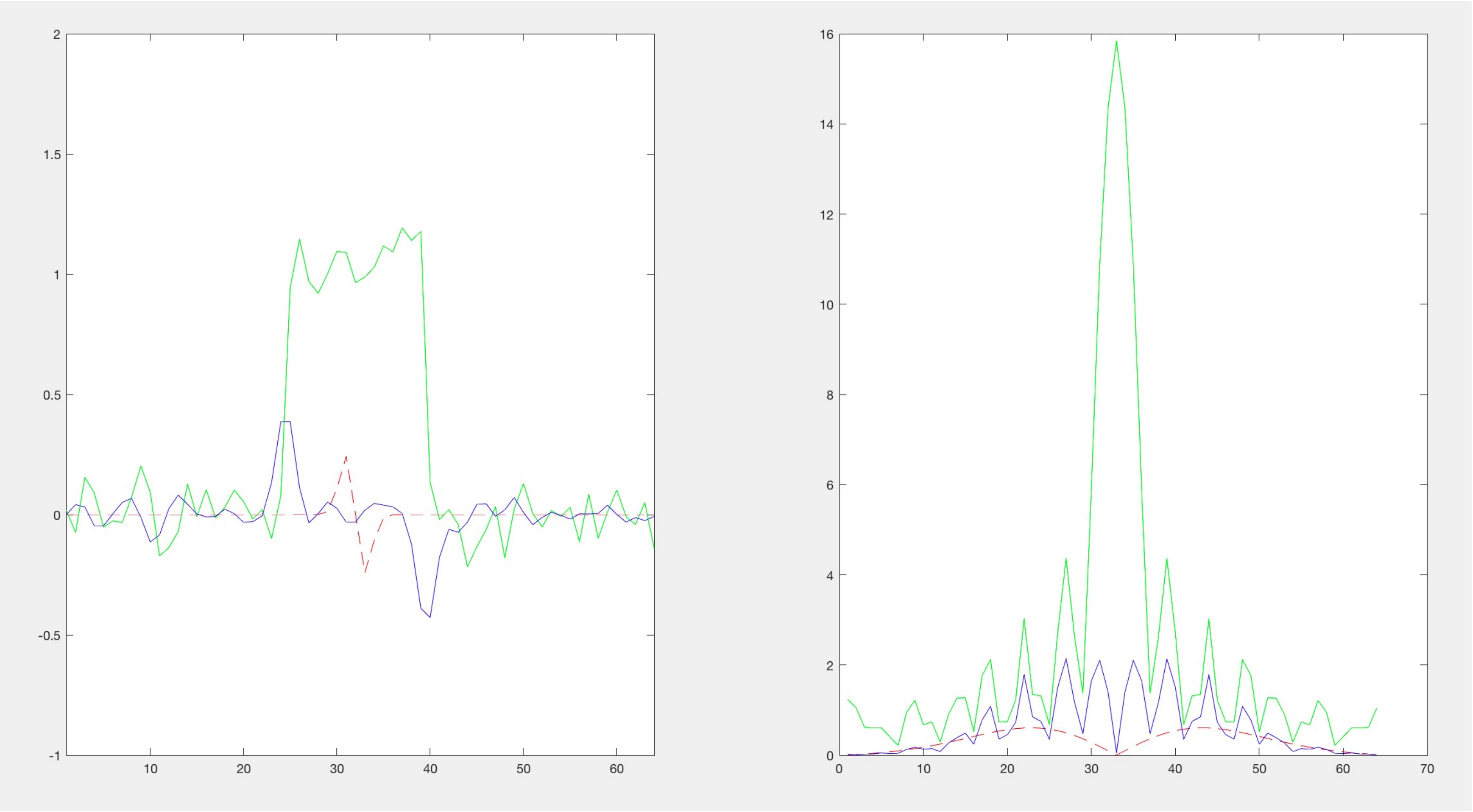
Magnitude of Fourier

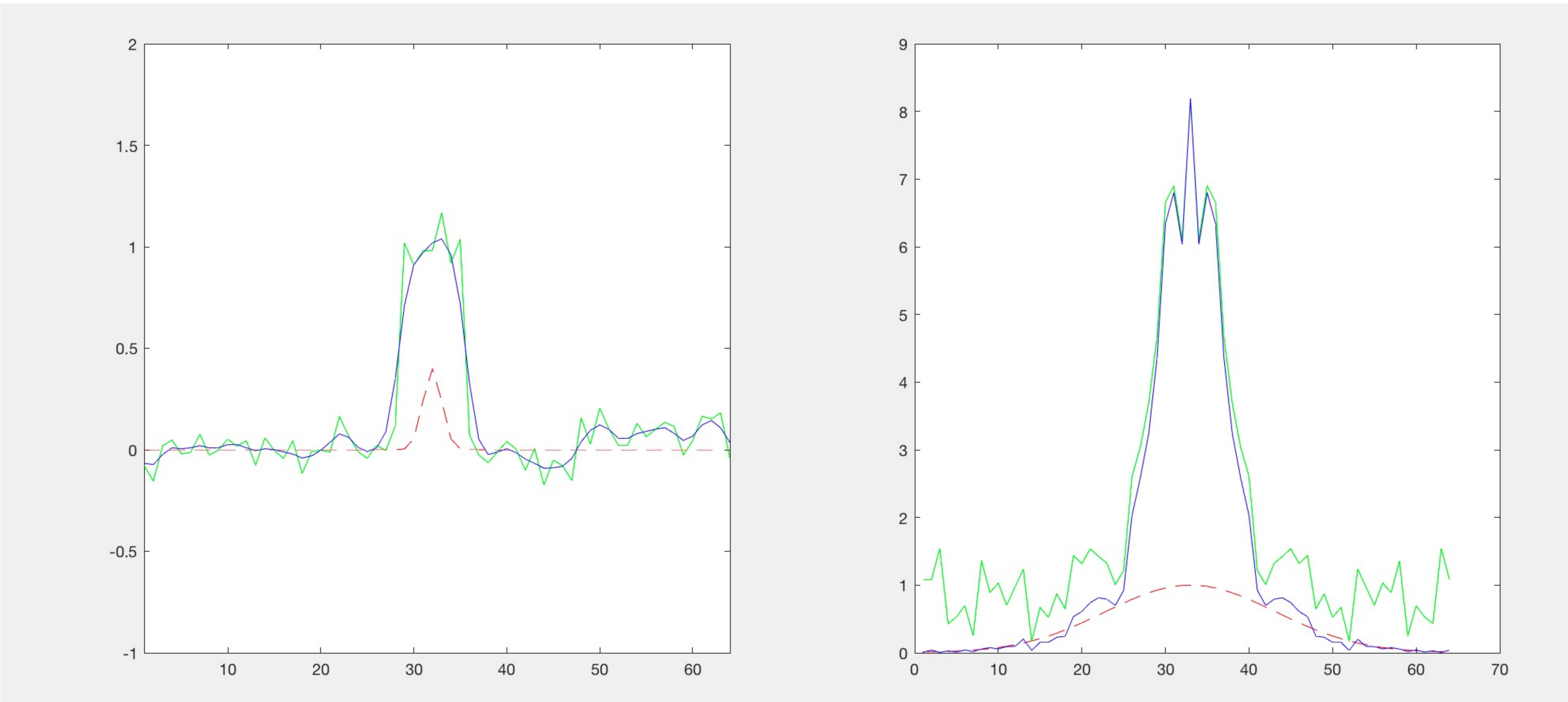
… and taking derivative and magnitude maximum



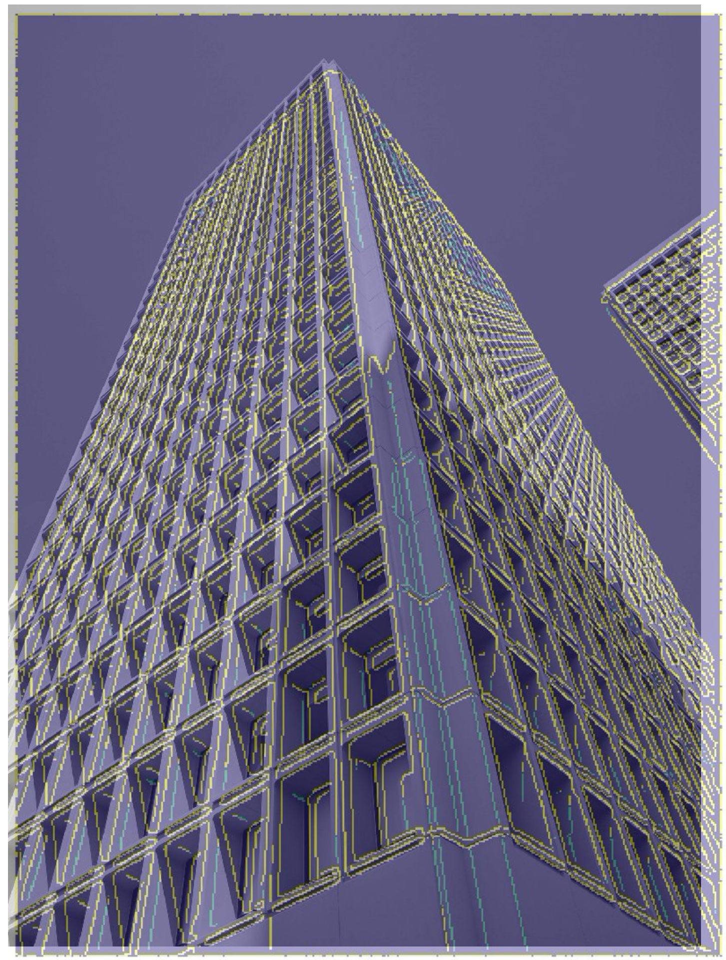

2
1.5
1
0.5
0
-0.5
-1
10
20
30
40
50
60
35
30
25
20
15
10
5
0
0
10
20
30
40
50
60
70

Ridge (or Box) Detection with the Gaussian first derivative



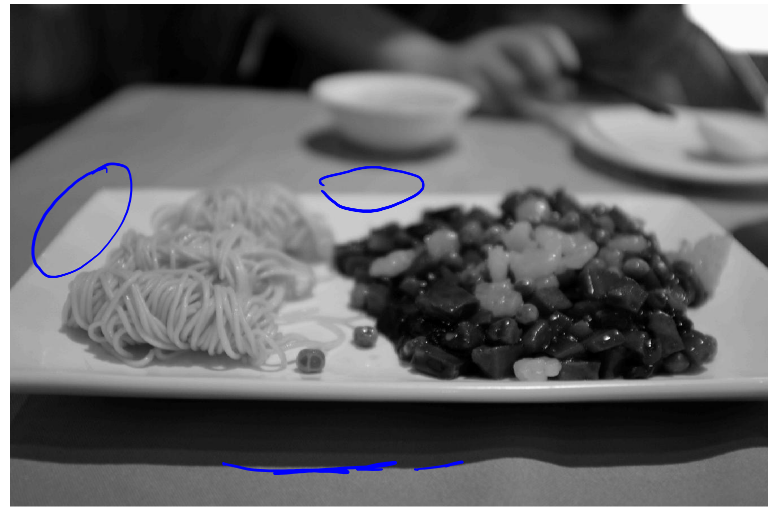


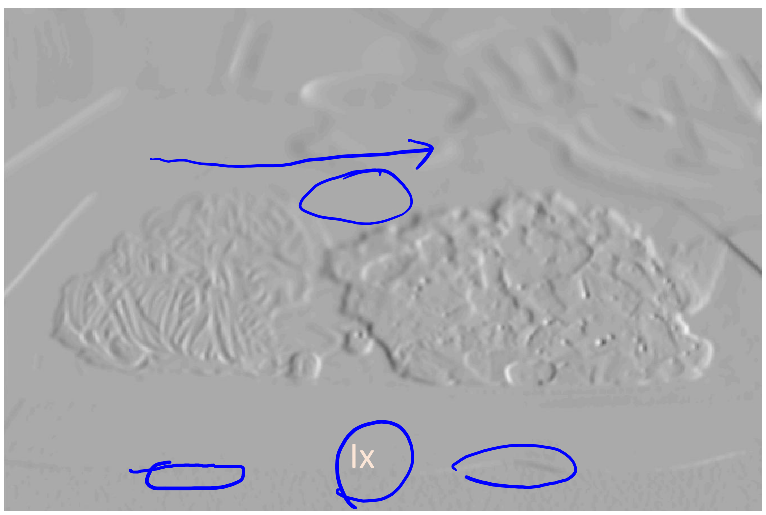
2
1.5
1
0.5
0
-0.5
-1
10
20
30
40
50
60
9
8
7
6
5
4
3
2
1
0
0
10
20
30
40
50
60
70

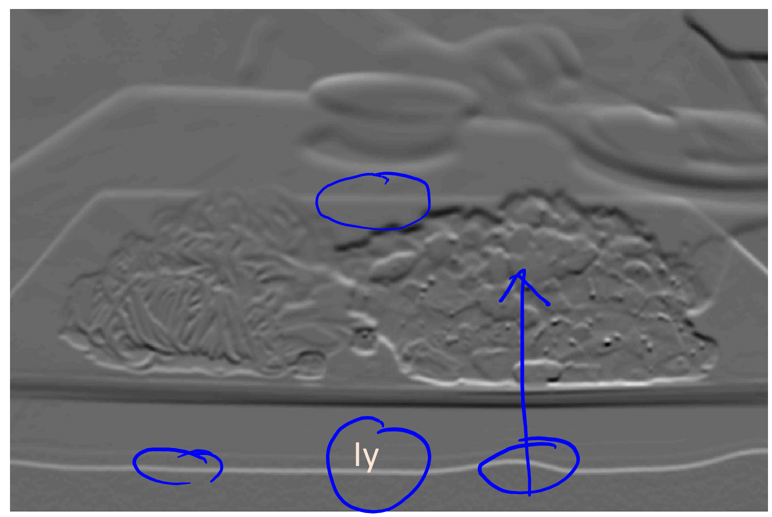


```
% % Convolution of image with Gaussian
Gx = conv2(G, dx, 'same');
Gy = conv2(G, dy, 'same');
```

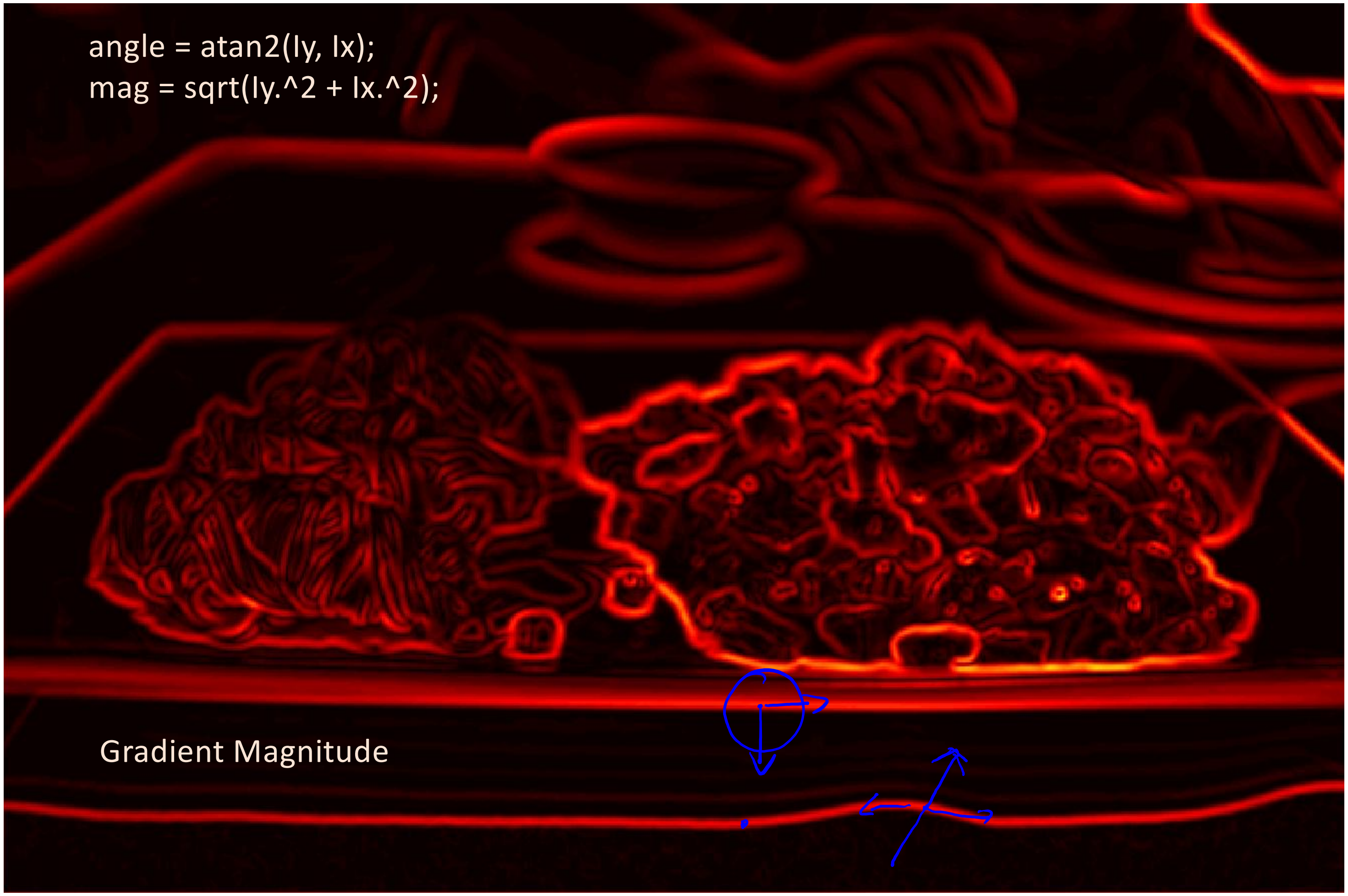
```
% Convolution of image with Gx and Gy
Ix = conv2(img, Gx, 'same');
Iy = conv2(img, Gy, 'same');
```

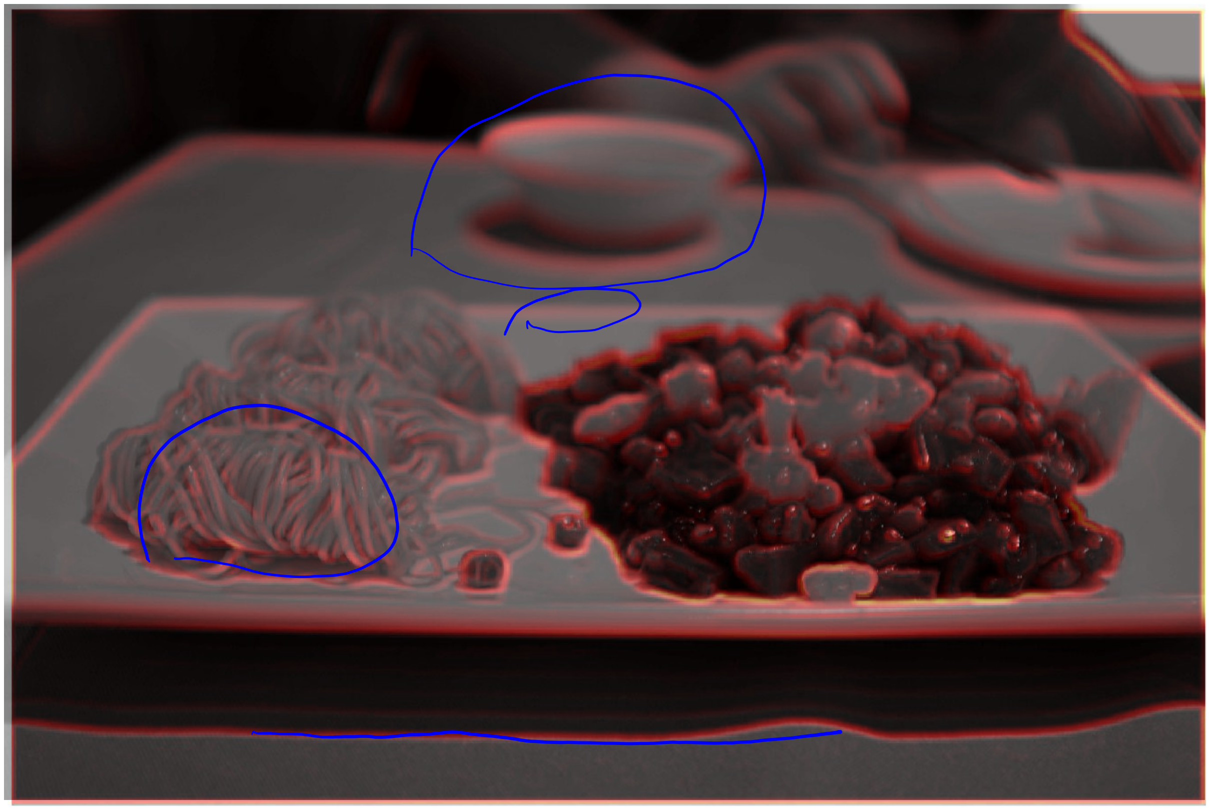






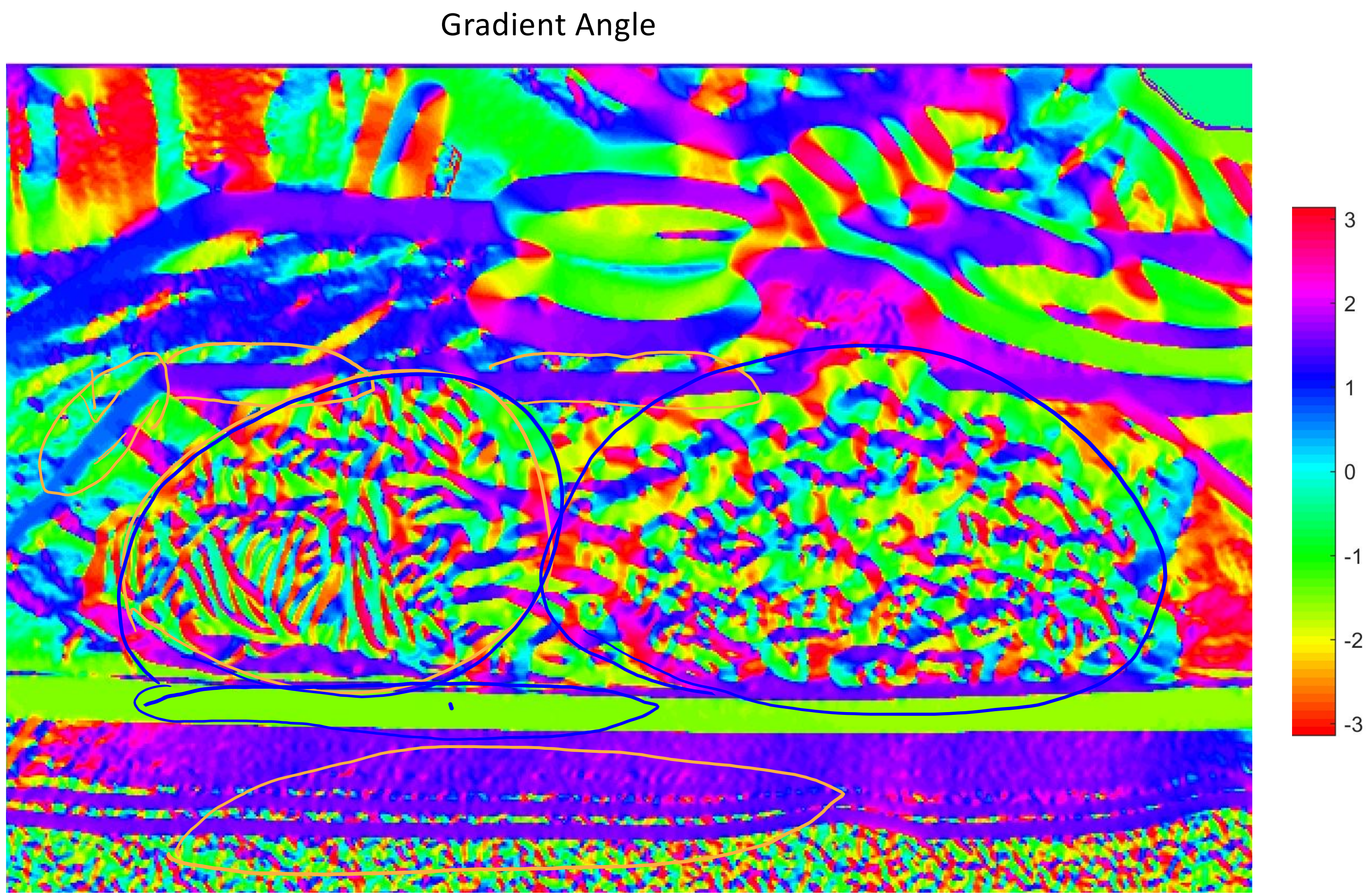
Canny Edge Detection

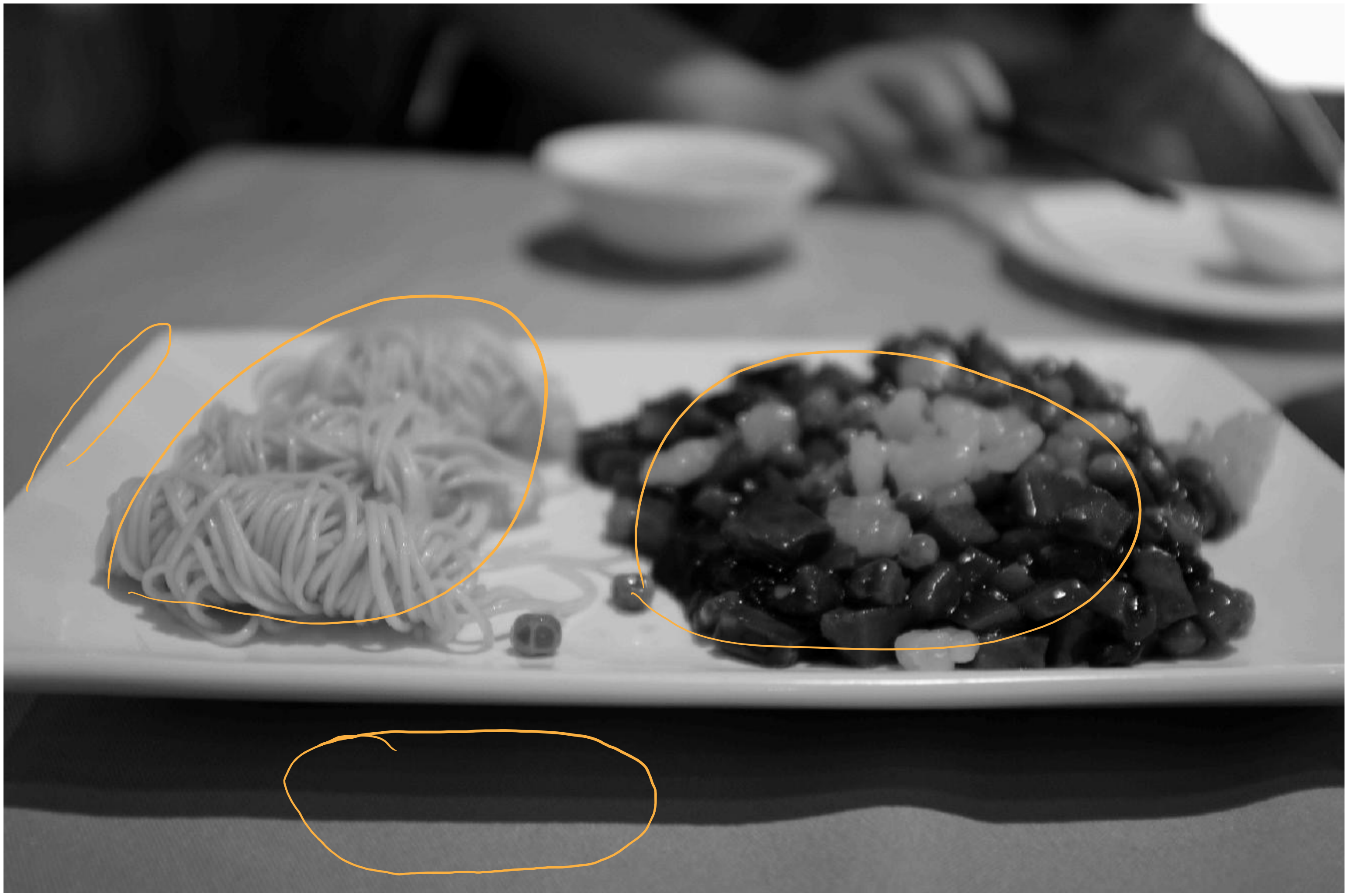


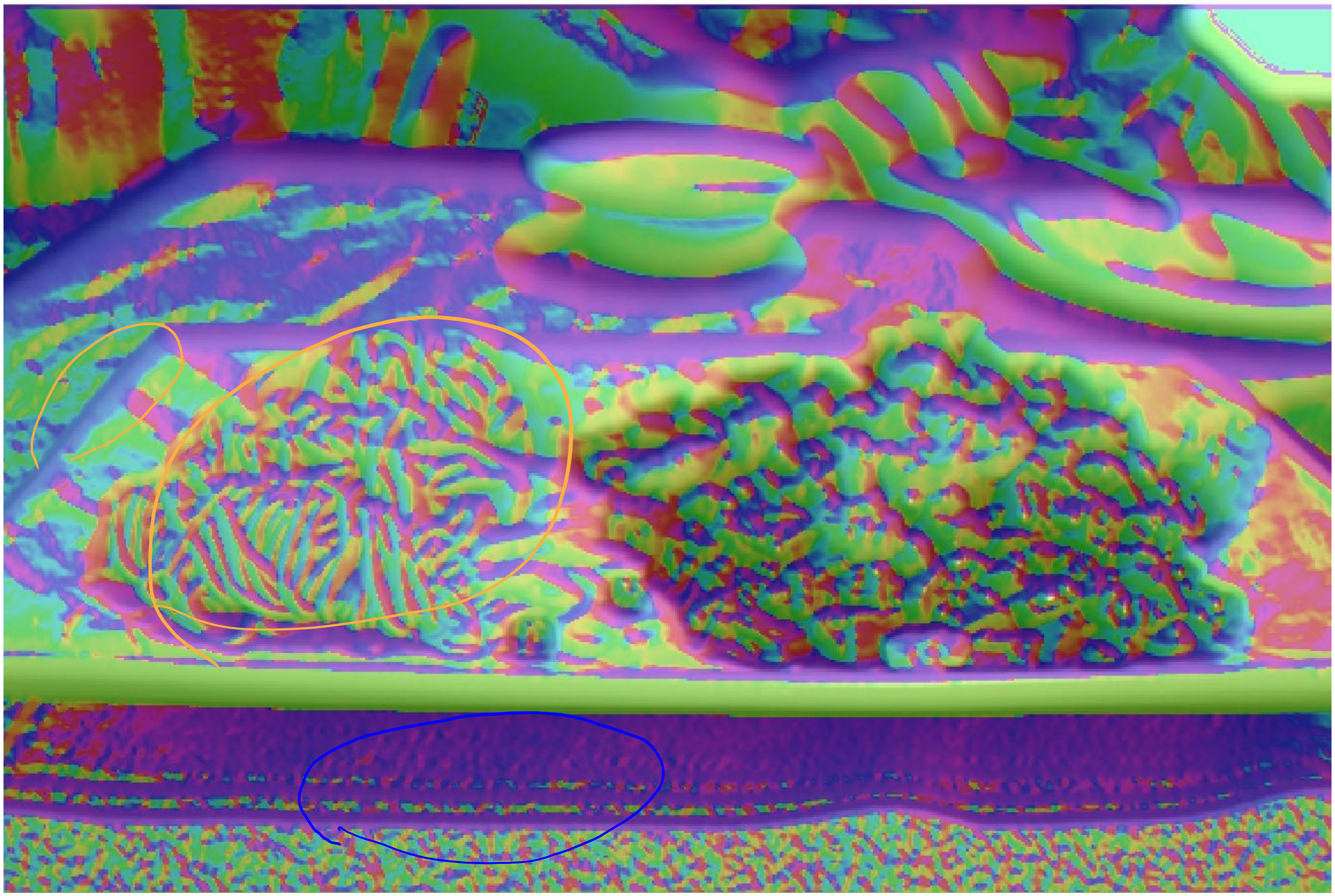


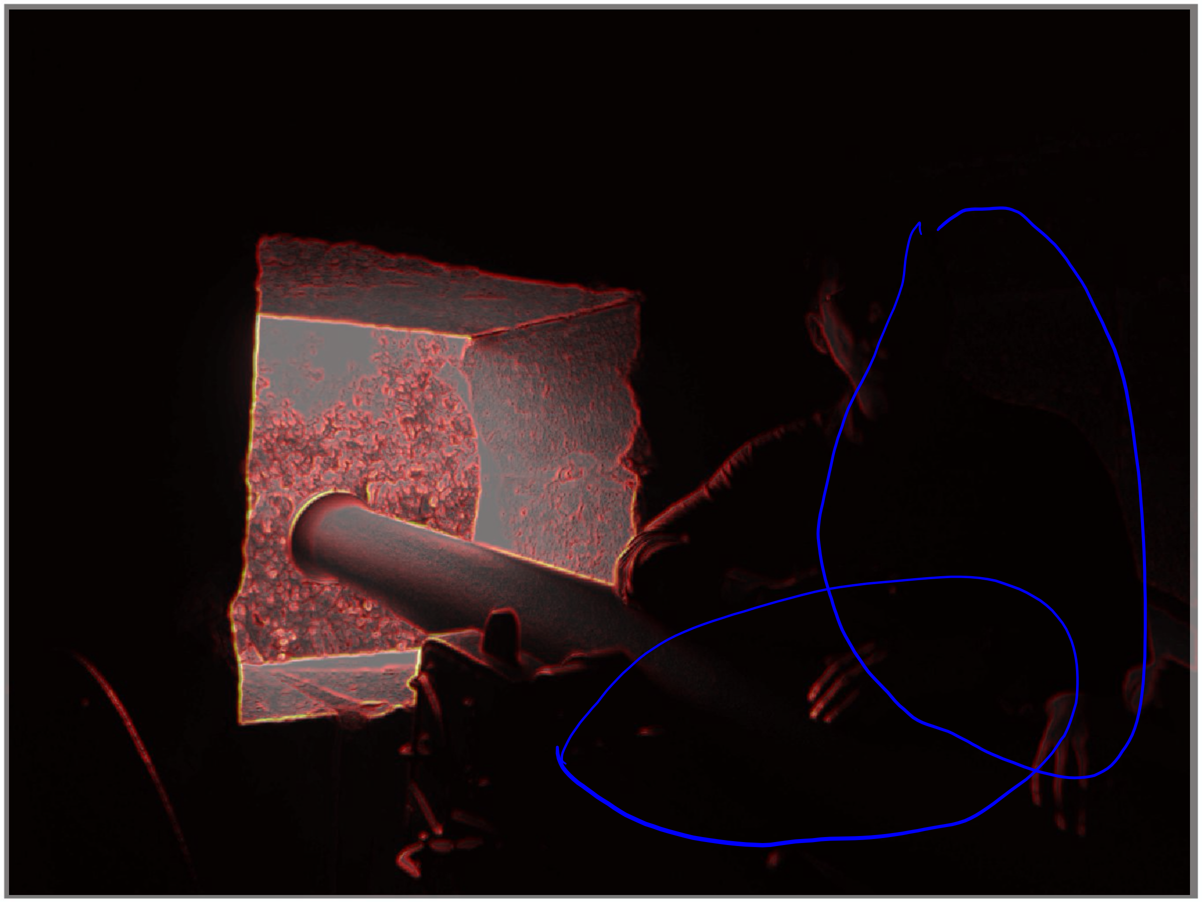
Edge Implementation

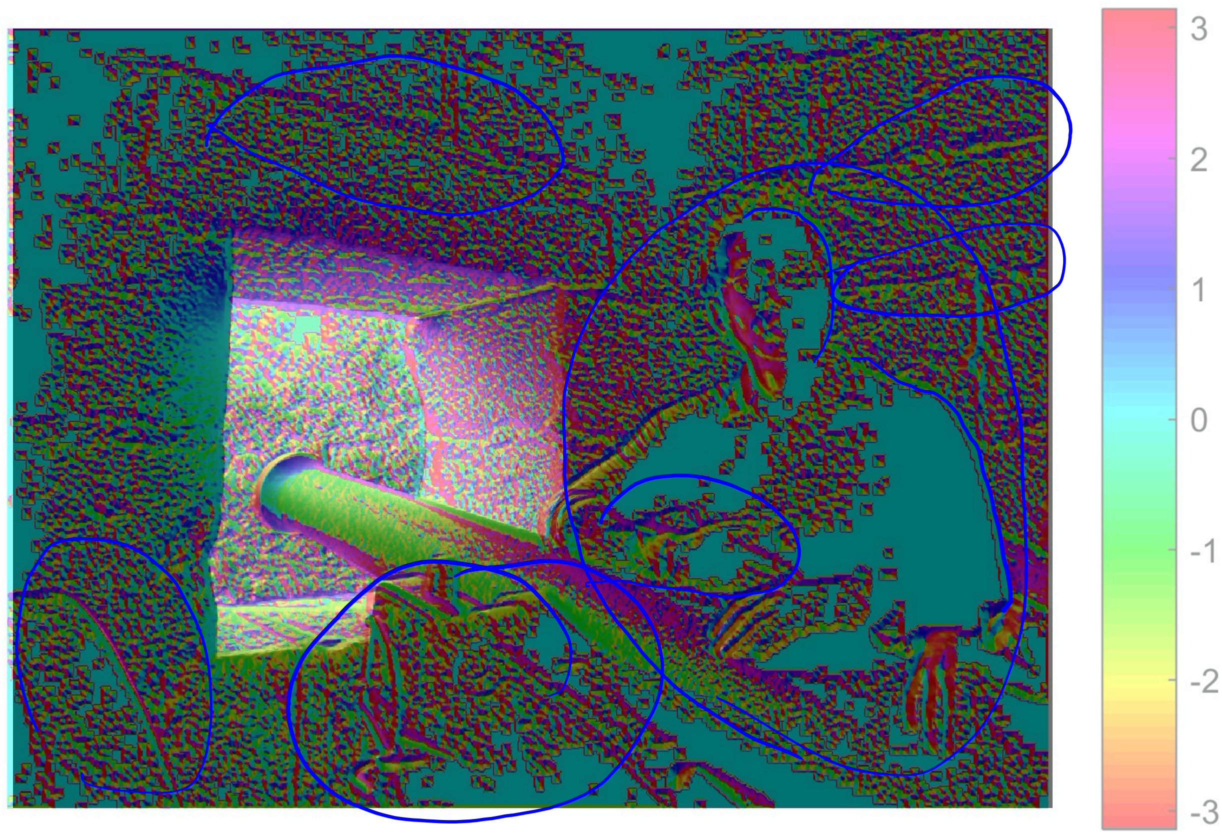
```
angle = atan2(Iy, Ix);
mag = sqrt(Iy.^2 + Ix.^2);
```

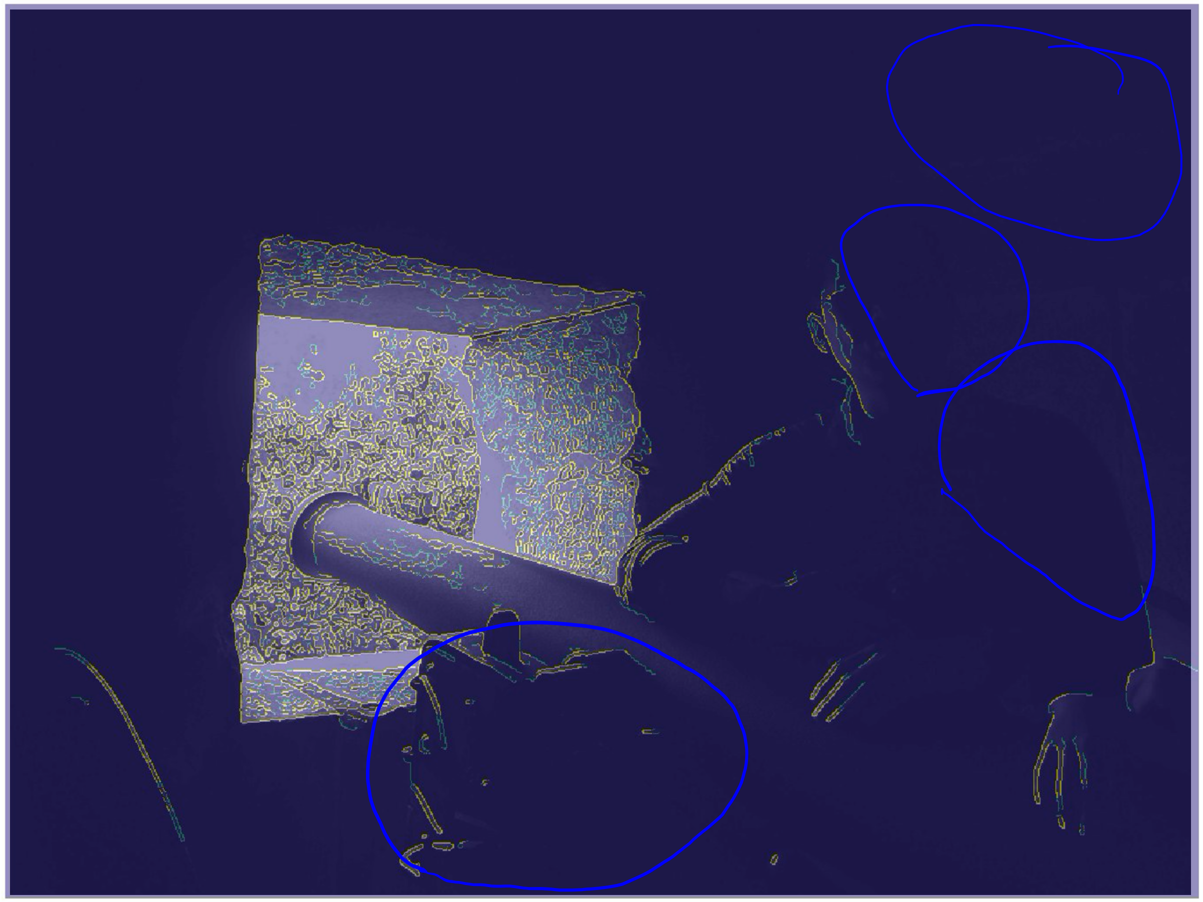






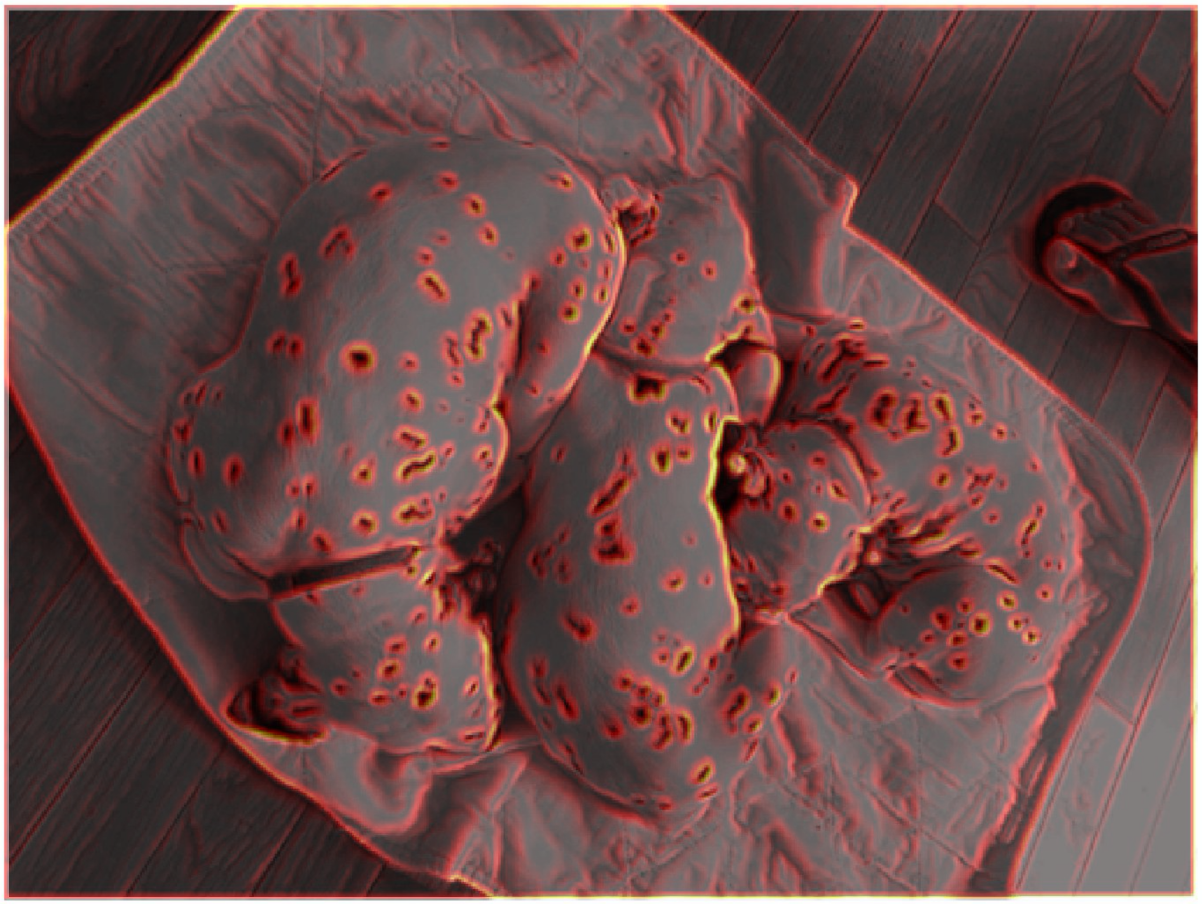


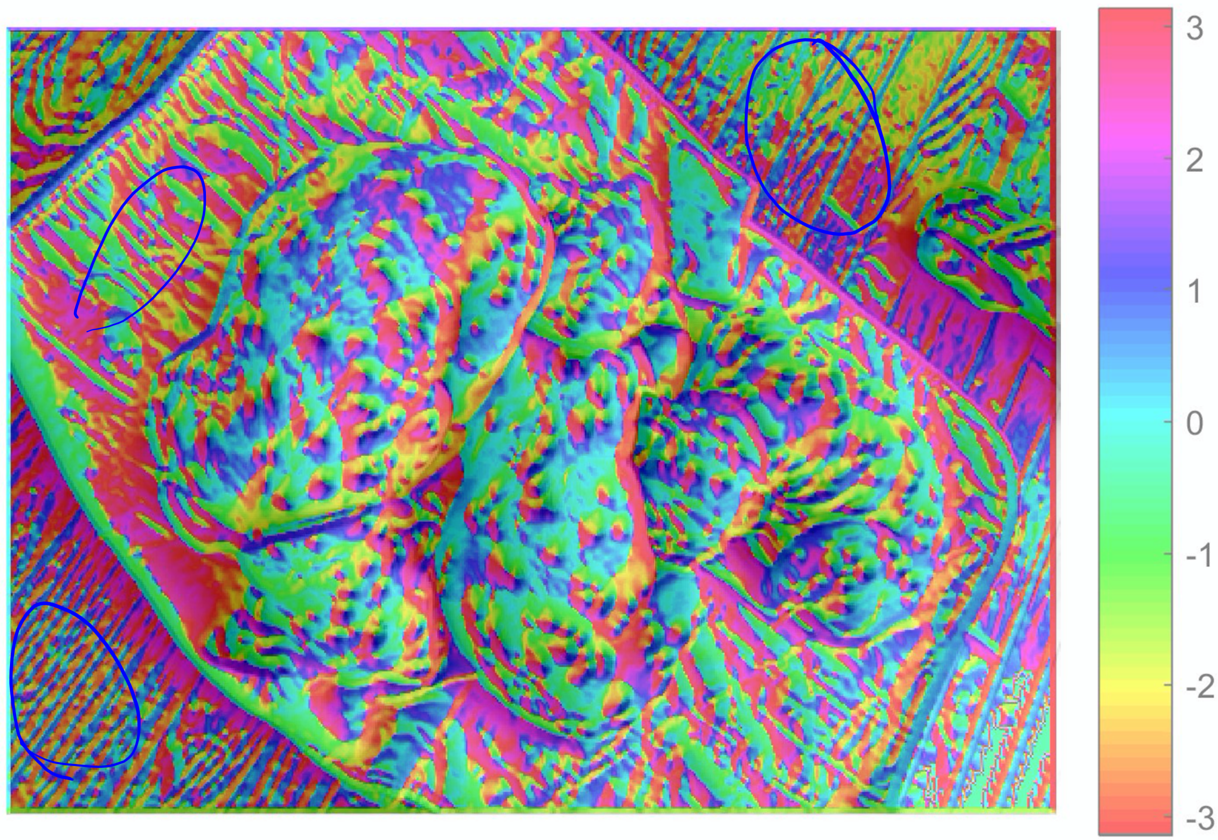

3
2
1
0
-1
-2
-3

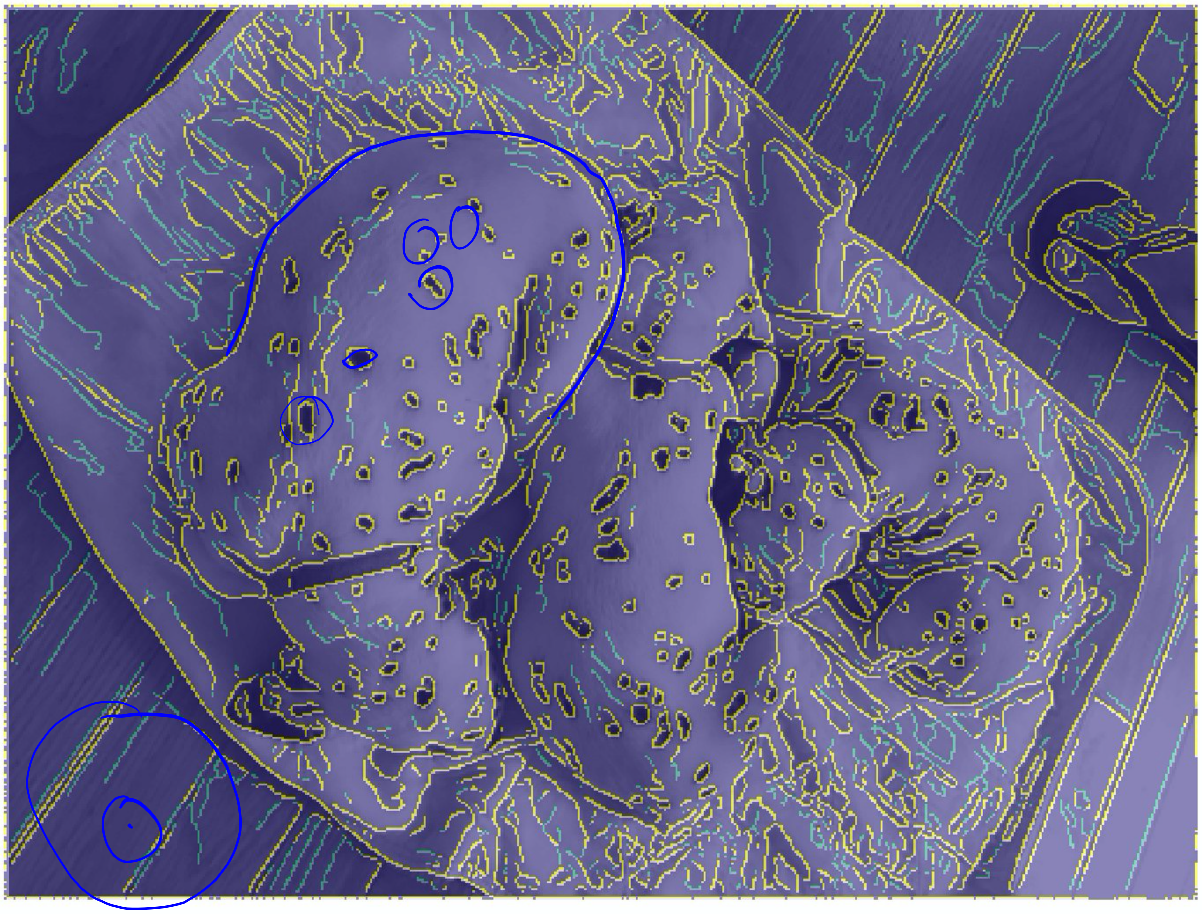


Ix

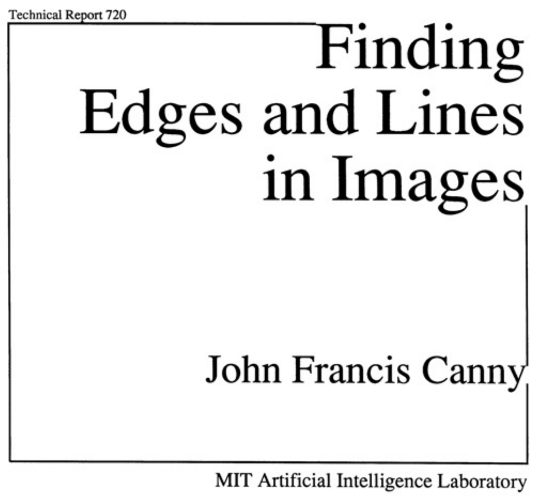
Iy




3
2
1
0
-1
-2
-3



Canny edge detection (1984 Master’s Thesis)



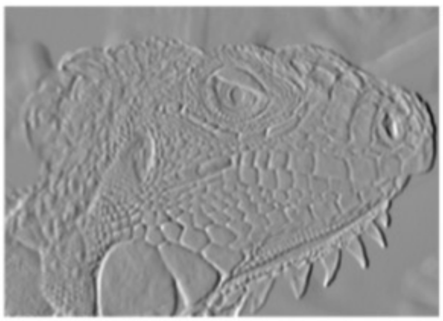
Technical Report 720

Finding Edges and Lines in Images

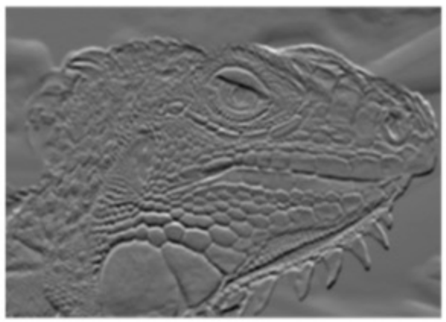
John Francis Canny

MIT Artificial Intelligence Laboratory

Compute gradient magnitude and direction

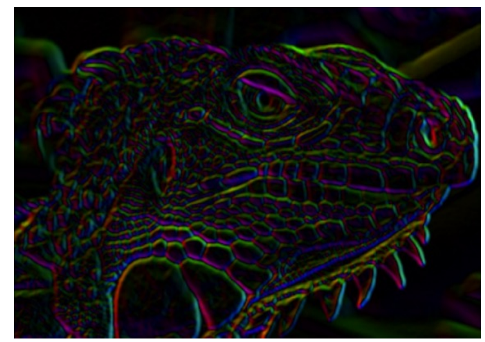


X-Derivative of Gaussian



Y-Derivative of Gaussian

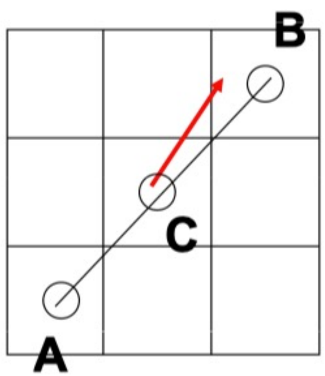
Gradient Magnitude



Gradient orientation

2D Non-Maximum Suppression

- Canny: Interpolate Gradient along gradients (plus and minus a certain distance) and check if center is larger than neighbors.
- Simplified: Test for each Gradient Magnitude pixel if neighbors along gradient direction (closest neighbors) are smaller than center: Mark C as maximum if A<C and B<C



Non-max Suppression

Before

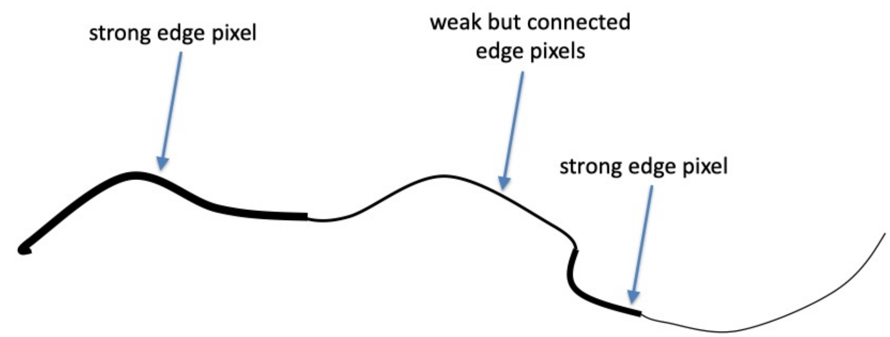
After

Hysteresis Thresholding to convert edges into connected contours (binary image)

If the gradient at a pixel is

- above High, declare it as an ‘strong edge pixel’
- below Low, declare it as a “non-edge-pixel”
- between Low and High
  - Consider its neighbors iteratively then declare it an “edge pixel” if it is connected to an ‘strong edge pixel’ directly or via pixels between Low and High

Hysteresis Thresholdi

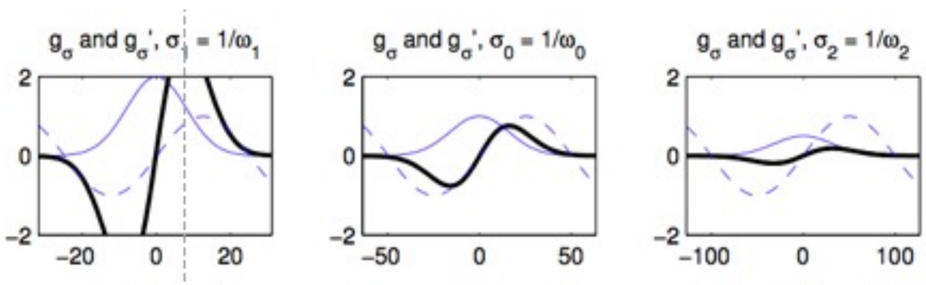


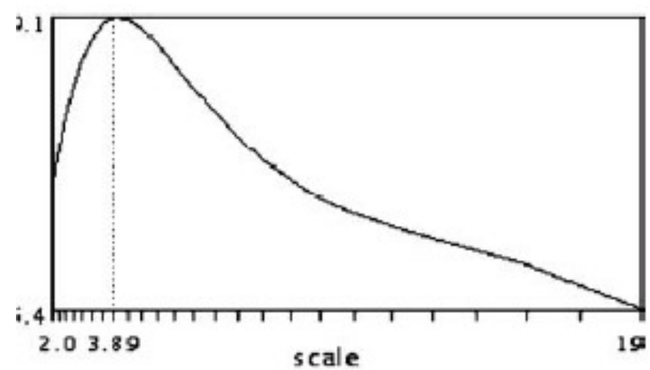
Source: S. Seitz

Detecting Keypoints
Scale Invariant Feature Transform (SIFT)

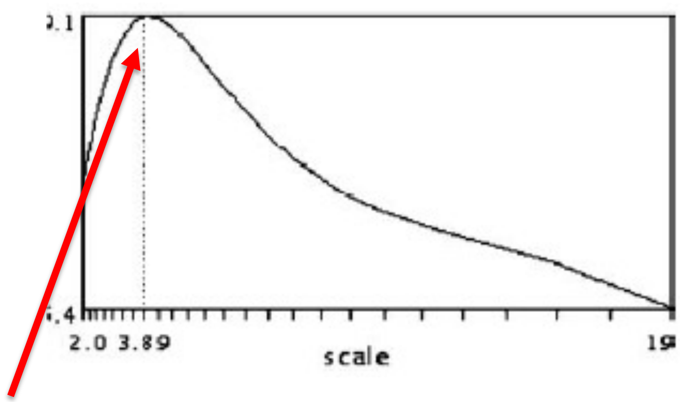
Can we find (select) the intrinsic image scale?

Yes, by taking the maximum over scale!





Scale selection



- The maximum across scale is the intrinsic scale of the image structure
- if the smoothed value is scale normalized.
- It turns out that only the derivatives of the Gaussian responses can be normalized.

Scale selection and invariance

Tony Lindeberg

Feature detection with automatic scale selection

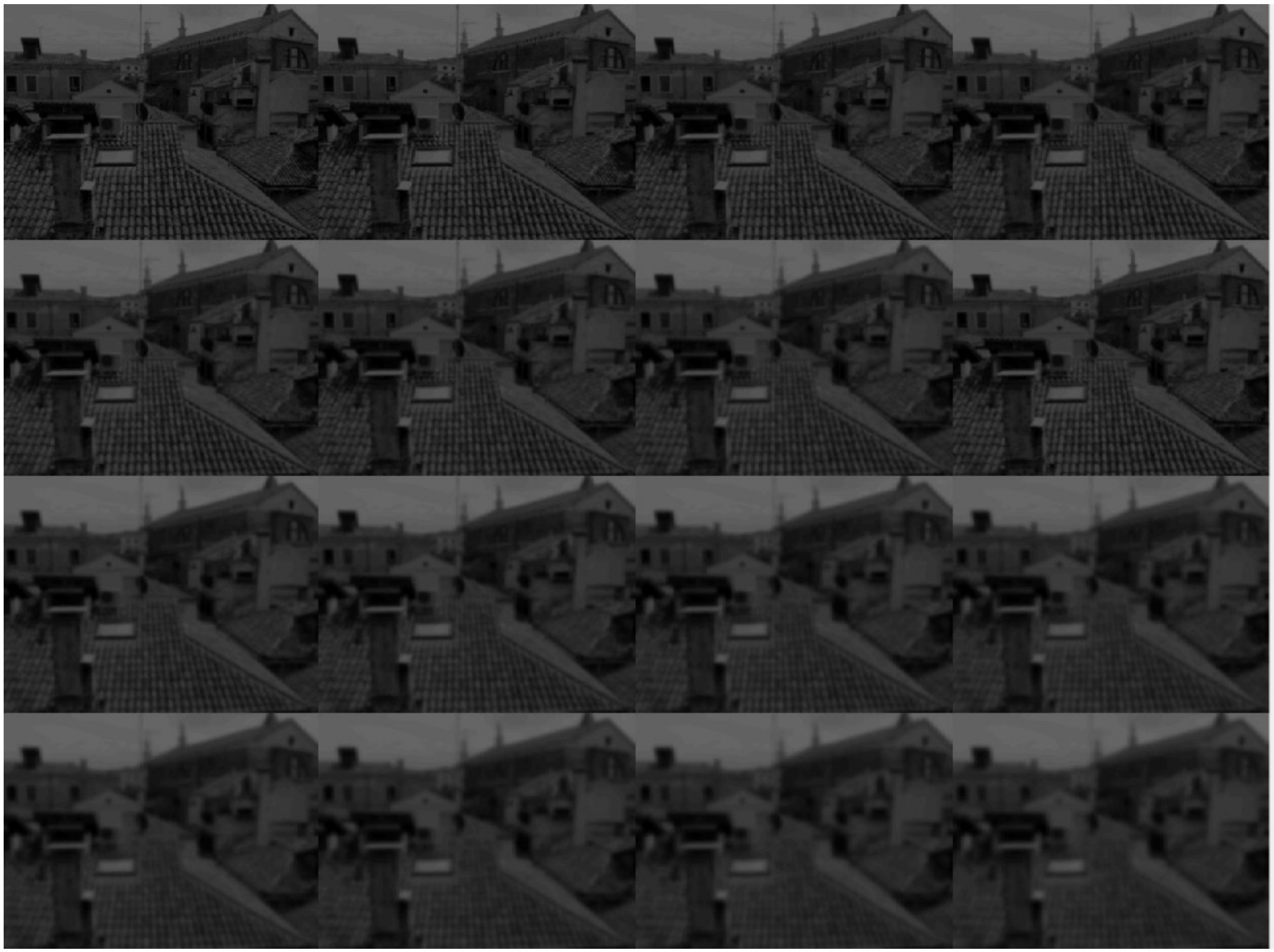
Authors	Tony Lindeberg
Publication date	1998/11/1
Journal	International Journal of Computer Vision
Volume	30
Issue	2

David Lowe

Distinctive image features from scale-invariant keypoints

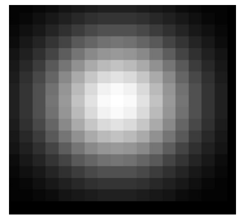
Authors	David G Lowe
Publication date	2004/11/1
Journal	International journal of computer vision
Volume	60
Issue	2

scale space without subsampling



How is scale space built?

*



2D Gaussian

=

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

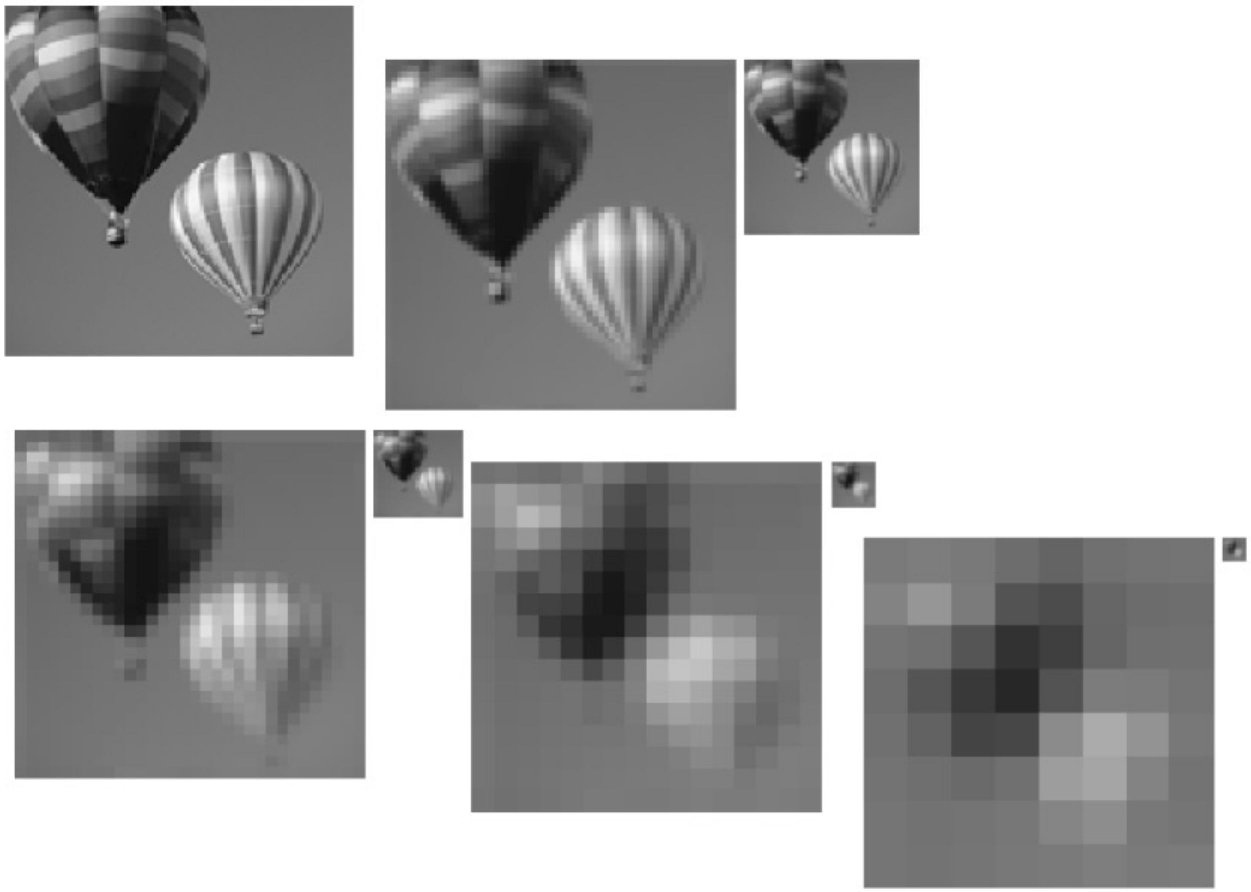
$*$ means convolution

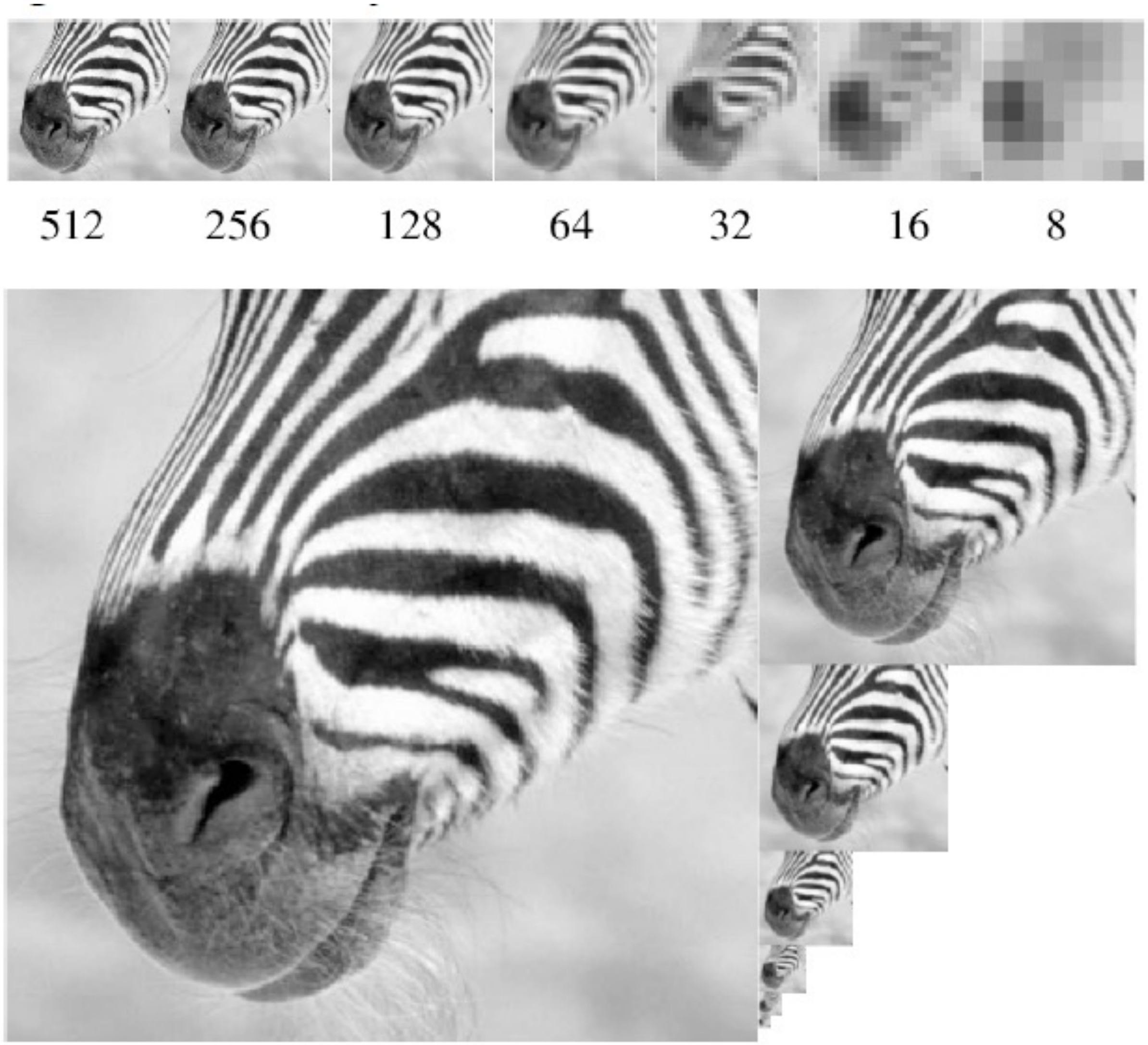
$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

The same scale but subsampled

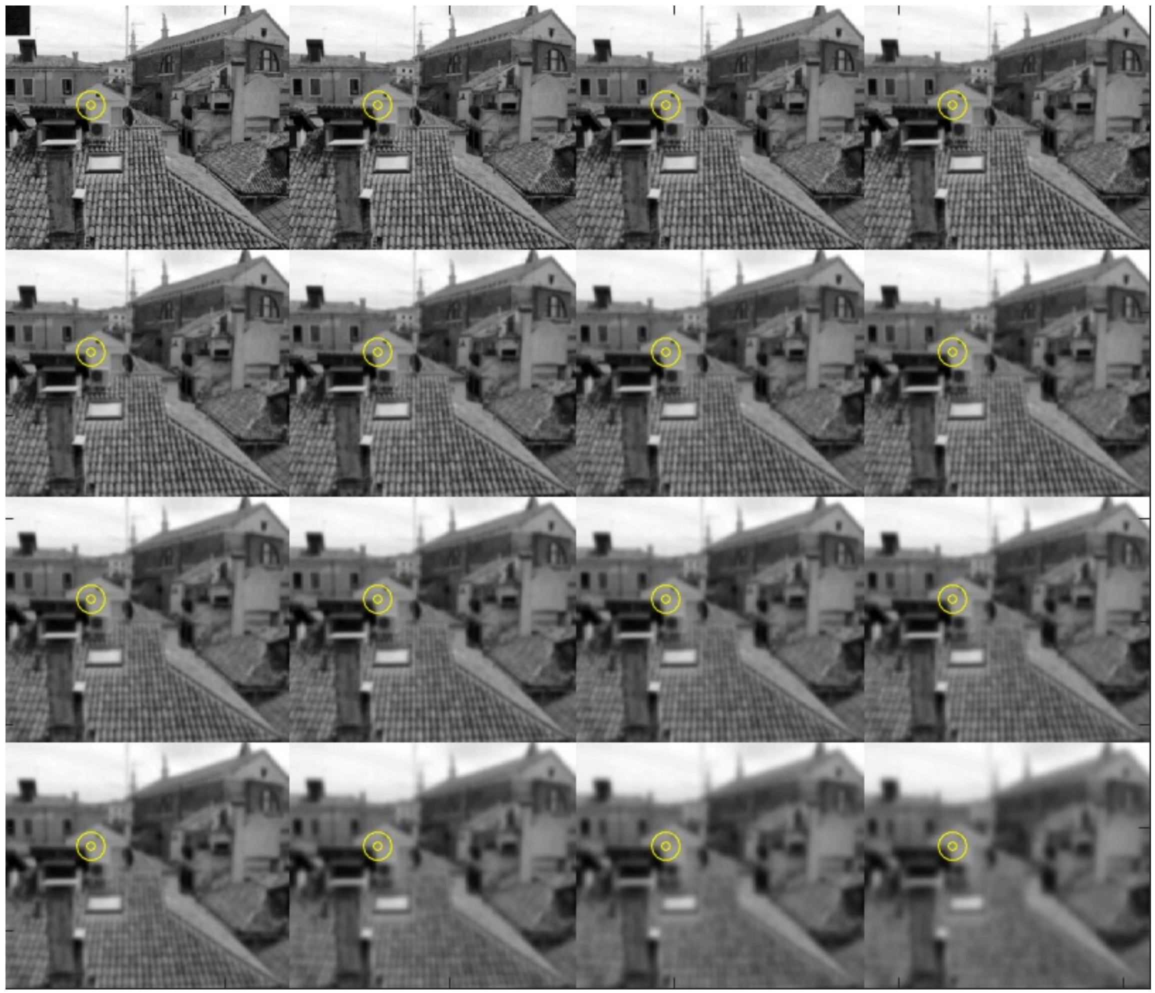
Every row has a sampling between σ and 2σ (like music notes inside an octave)
We subsampled it every time that the sigma of the Gaussian was doubled!

Look at coarser levels upsampled:

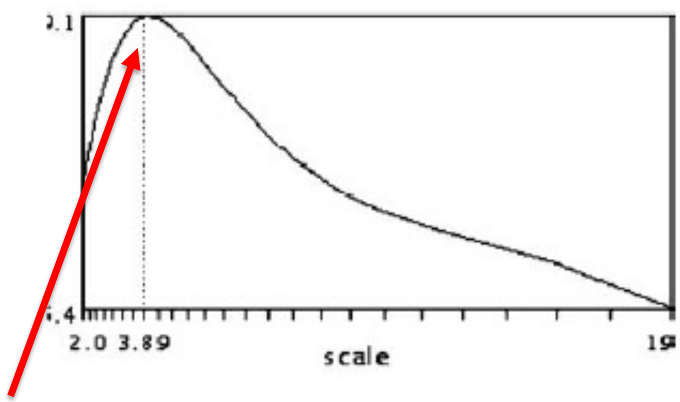



512
256
128
64
32
16
8

Now look at the same pixel across scale

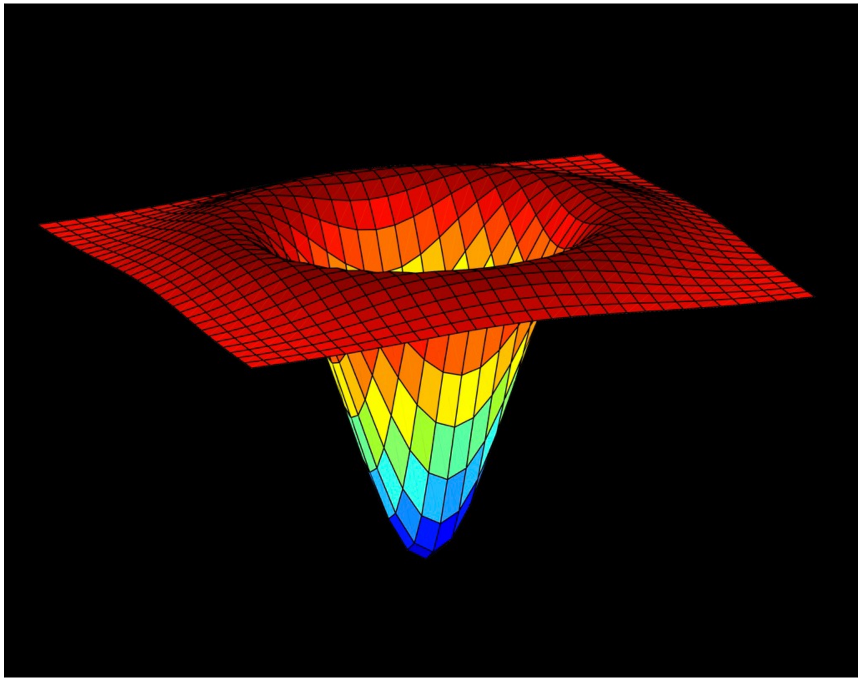


Scale selection



- The maximum across scale is the intrinsic scale of the image structure
- if the smoothed value is scale normalized.
- It turns out that only the derivatives of the Gaussian responses can be normalized.

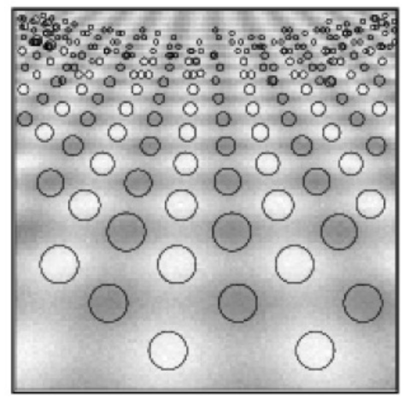
Instead of Gaussian itself we choose the 2nd derivative (trace of Hessian) , called Laplacian of Gaussian (LoG)

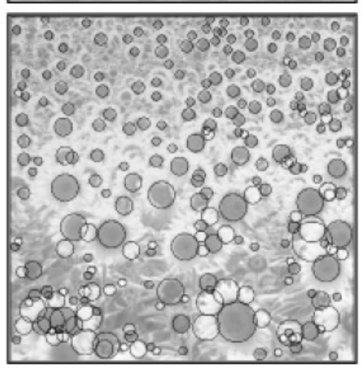


Which has the nice properties that

- it can be approximated as the difference of two Gaussians (prove)
- can detect blob like features!

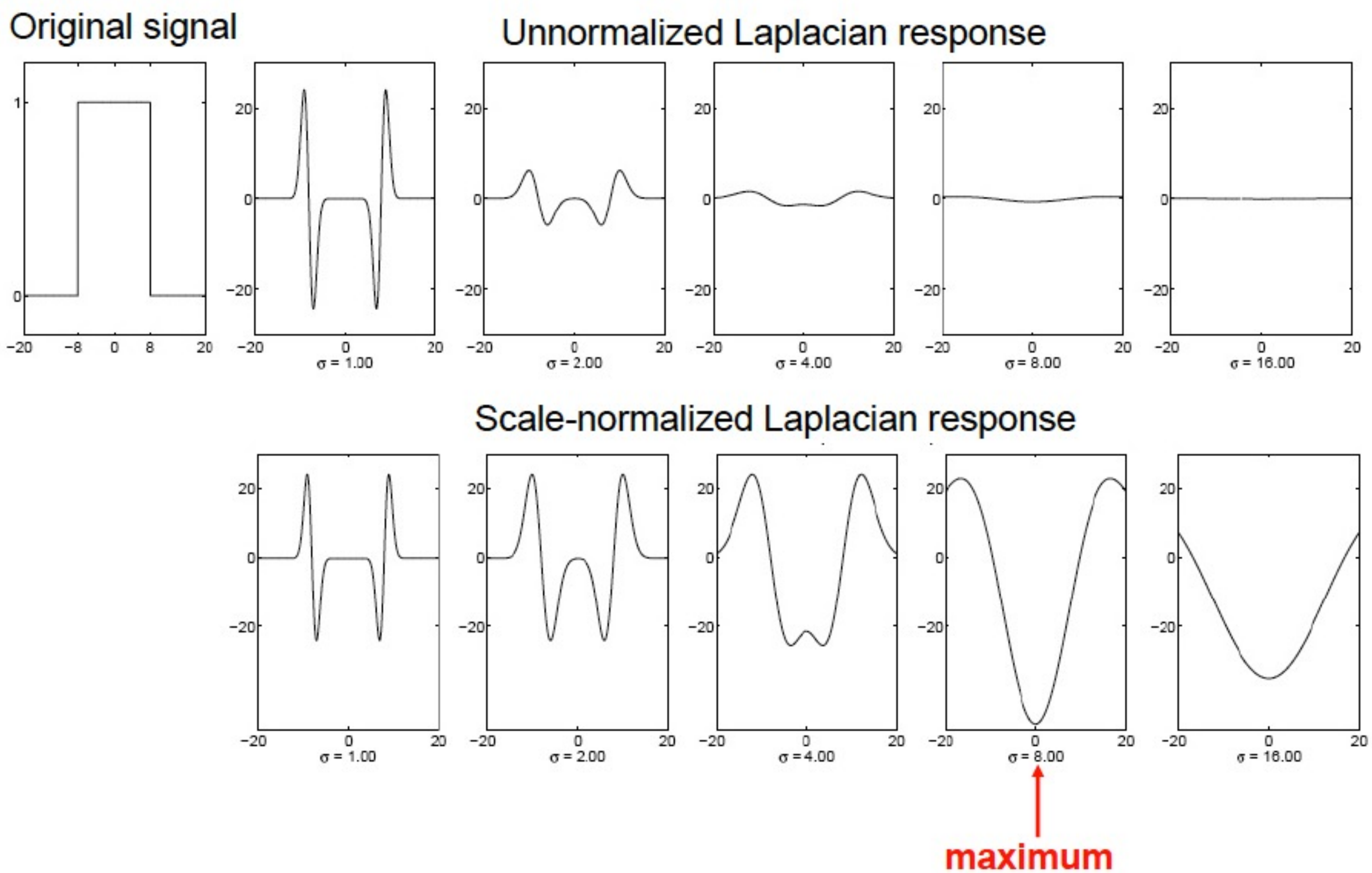
Detecting blobs at multiple scales





But first, we have to talk about detecting blobs at one scale...

Normalization of the 2nd Gaussian Derivative

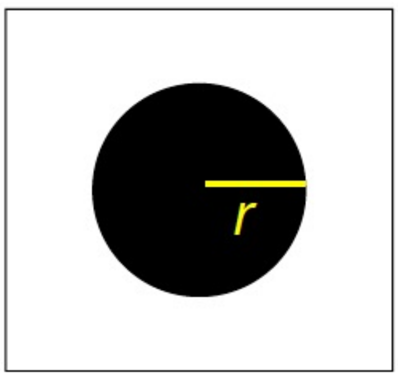


Scale selection

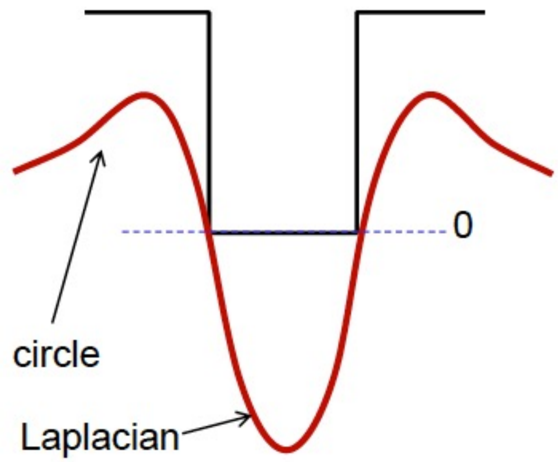
- At what scale does the Laplacian achieve a maximum response to a binary circle of radius r?
- To get maximum response, the zeros of the Laplacian have to be aligned with the circle
- The Laplacian is given by (up to scale):

$$(x^2 + y^2 - 2\sigma^2)\, e^{-(x^2+y^2)/2\sigma^2}$$

- Therefore, the maximum response occurs at $\sigma = r / \sqrt{2}.$

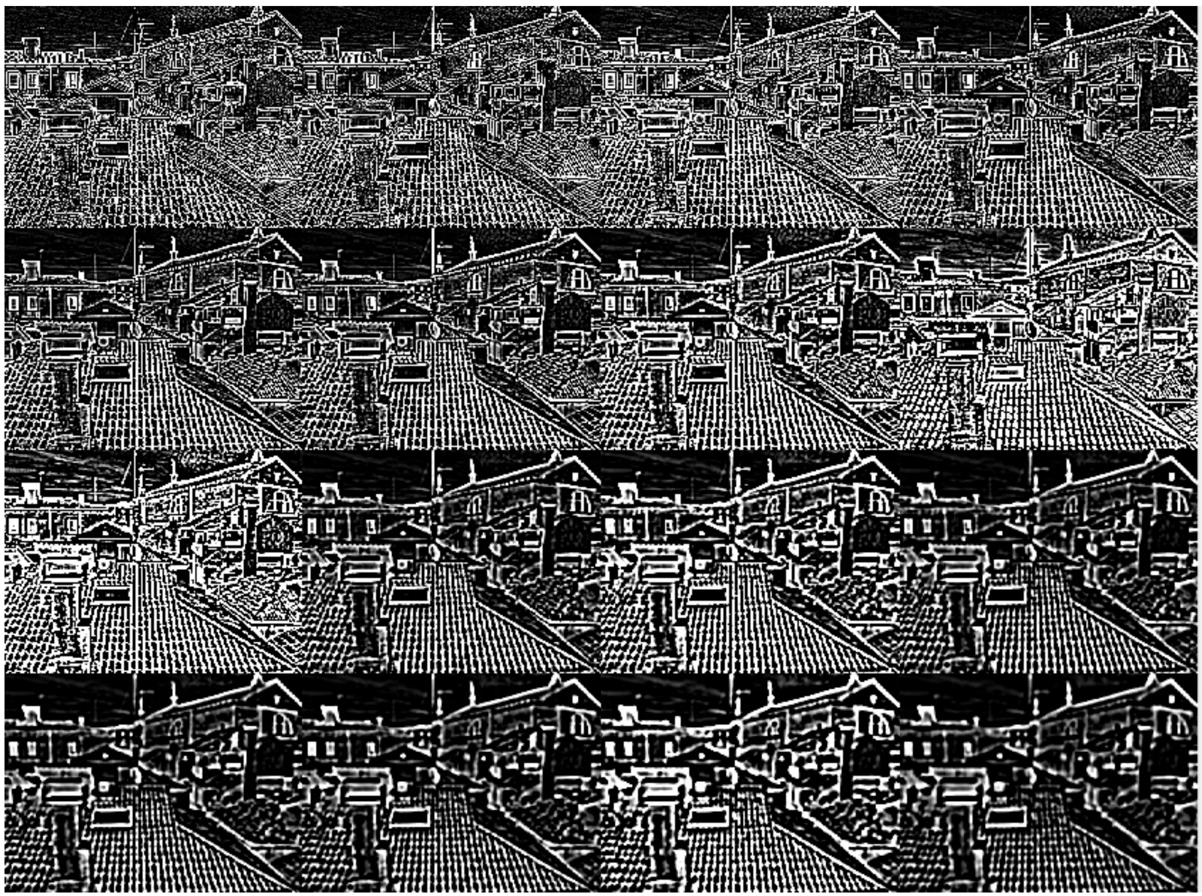


image

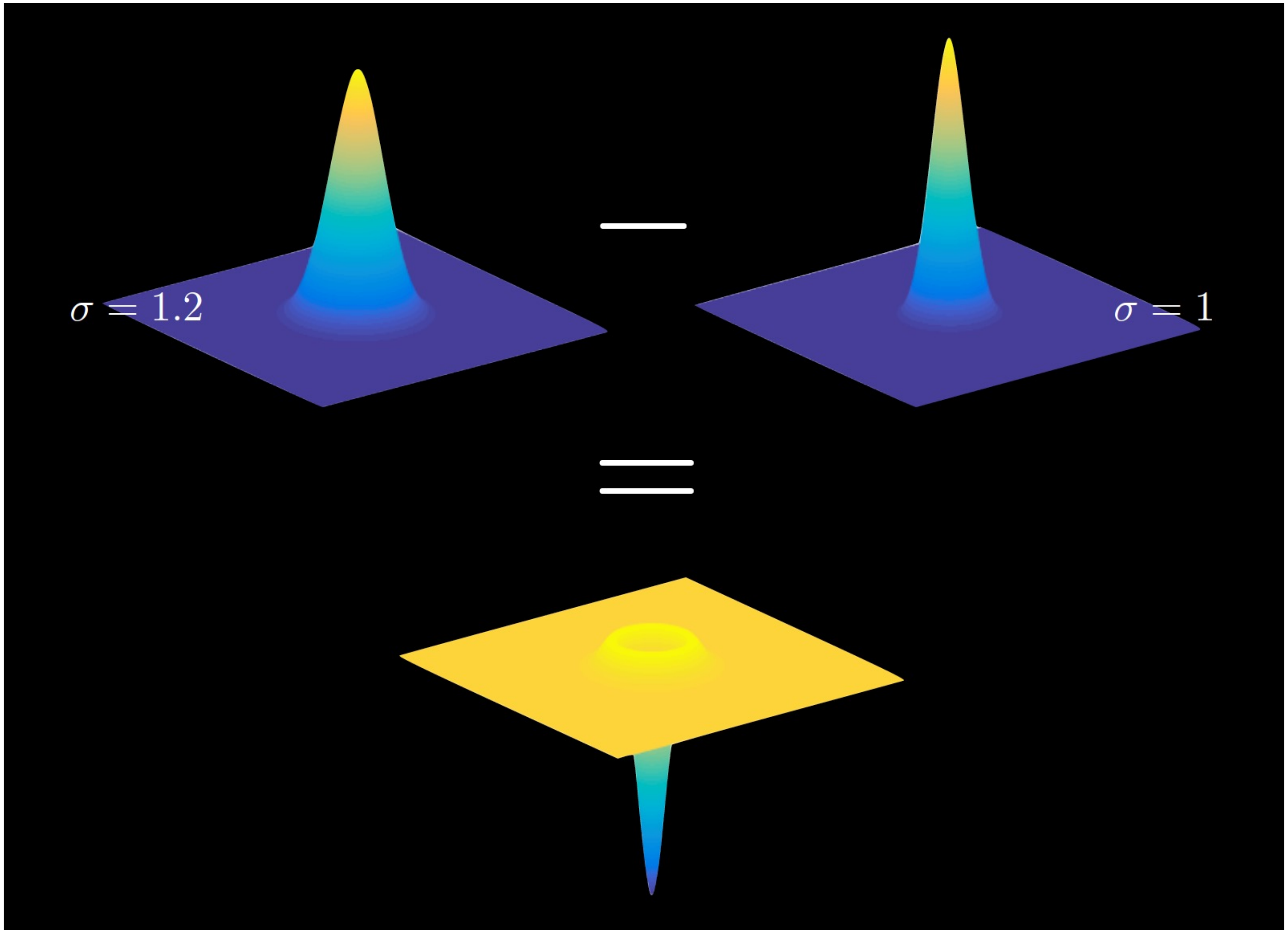


Slide from S. Lazebnik

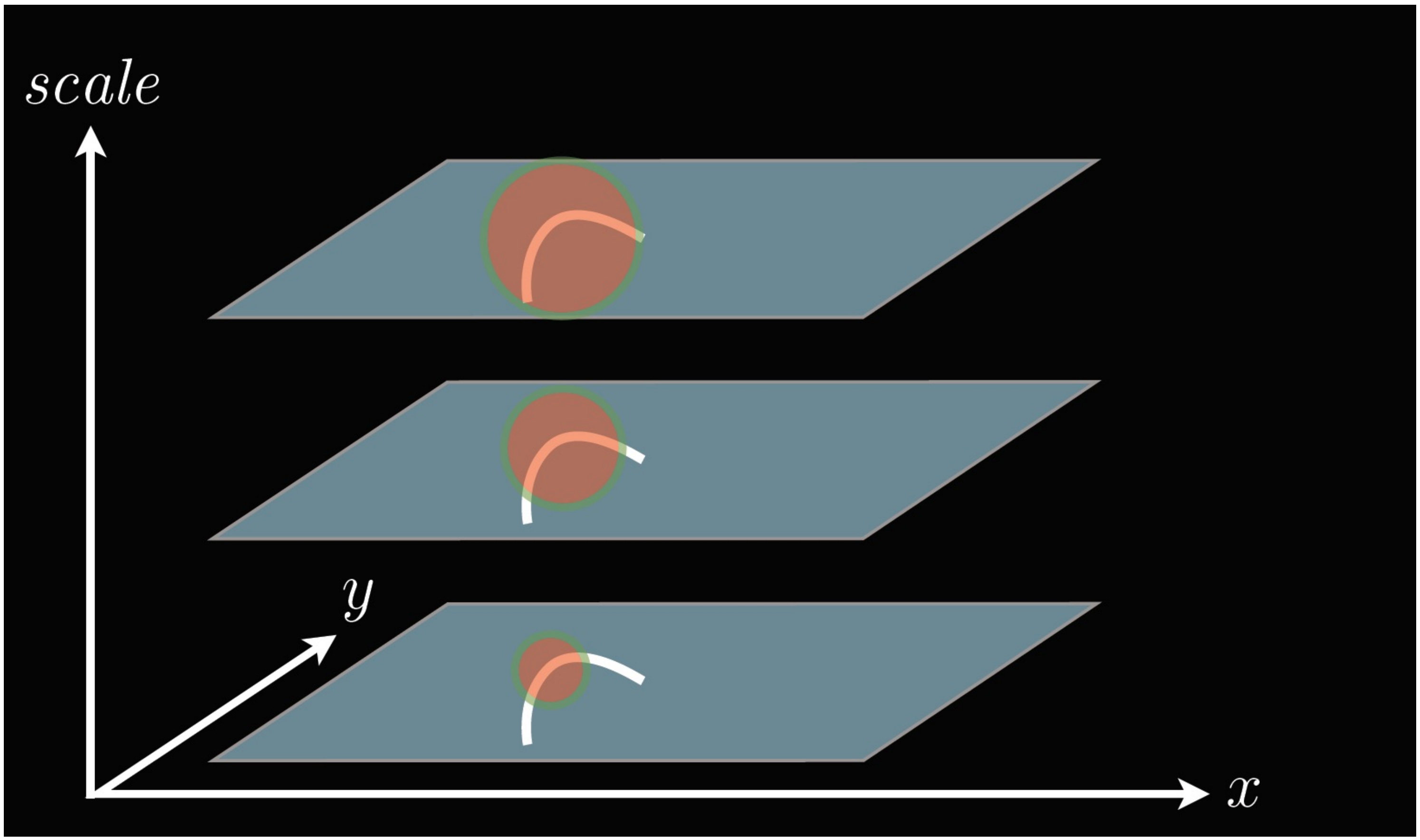
Laplacian Scale Space



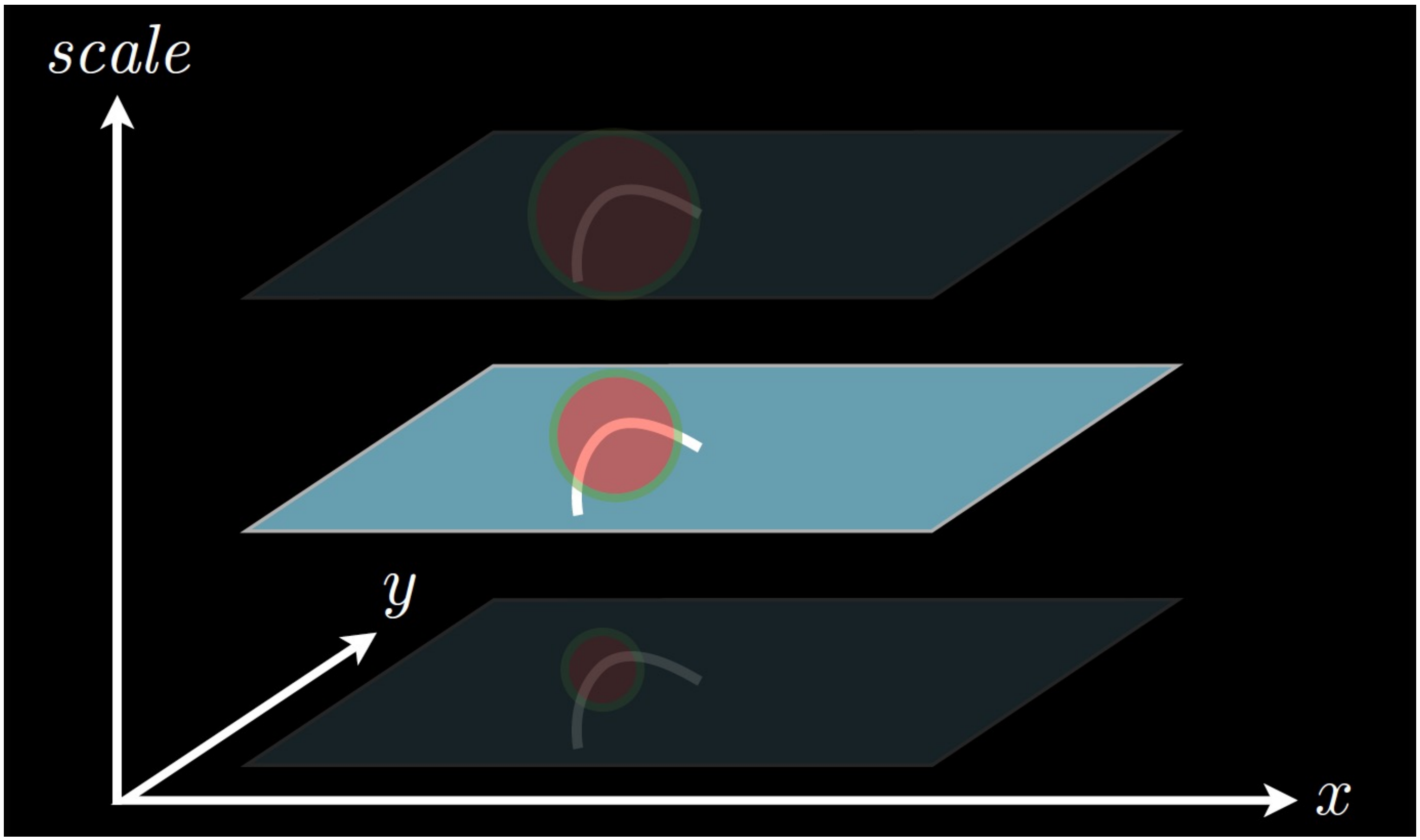
Laplacian of Gaussian (LoG) =Difference of Gaussians (DoG)



We convolve DoG across space at different scales

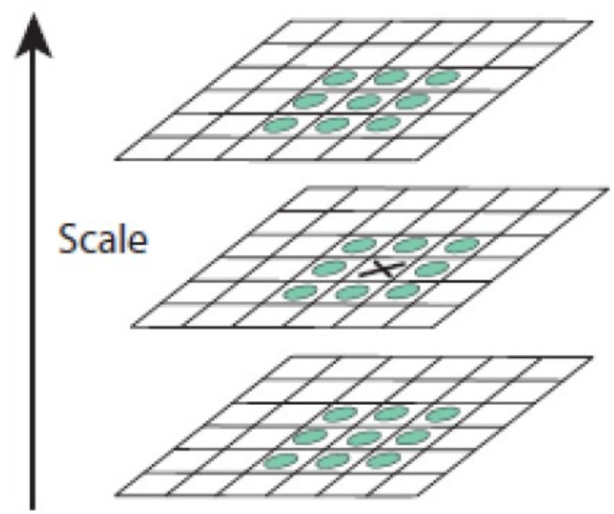


We convolve DoG across space at different scales and detect maximum



So, where is a SIFT keypoint?

- Definition of a keypoint: Maximum in the 3x3x3 (x,y,σ) region of the point.



Vedaldi’s vlfeat

VLFeat.org

Home | Download | Tutorials | Applications | Documentation

Andrea Vedaldi, Ph.D. vedaldi@robots.ox.ac.uk
Associate Professor in Engineering Science
Information Engineering Building 30.05, Parks Road, Oxford, OX1 3PJ
Visual Geometry Group (directions)
Tel. +44 1865 273 127

Résumé Google Scholar

Selected σ is visualized with a circle

Denoting the support region of the feature

Detector is rotation invariant

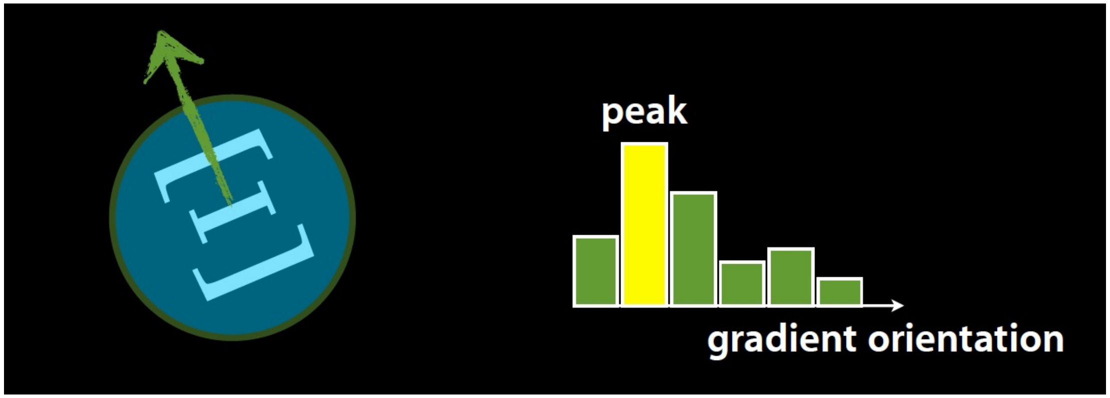
Because Laplacian is isotropic and a maximum in (x,y,σ) is invariant to rotations.

Descriptor invariance

- Since the intrinsic scale is detected (circle size) all circles will be normalized to a 16x16 region.

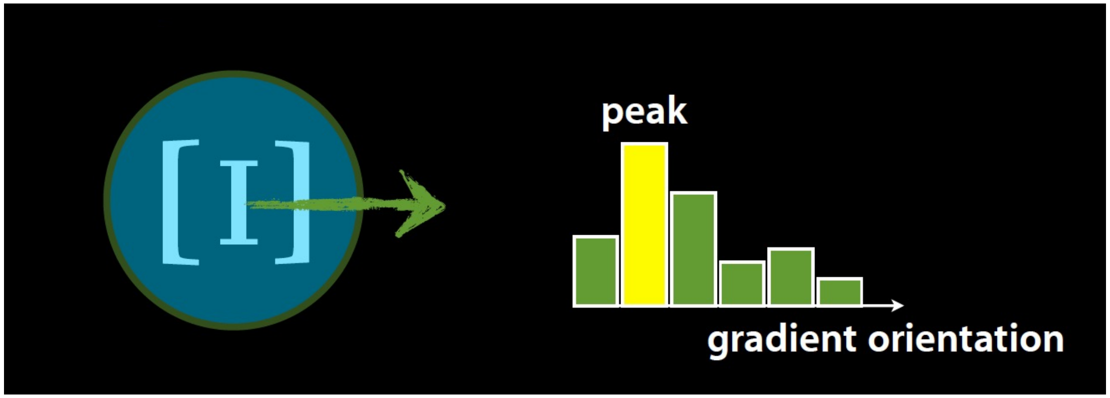
The descriptor should be also rotation invariant

- 1st Step: Find dominant orientation for the patch



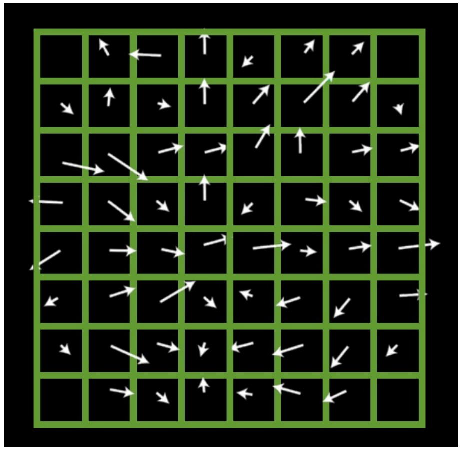
The descriptor should be rotation invariant

- 1st Step: Find dominant orientation for the patch
- 2nd Step: Rotate patch to point along x-axis



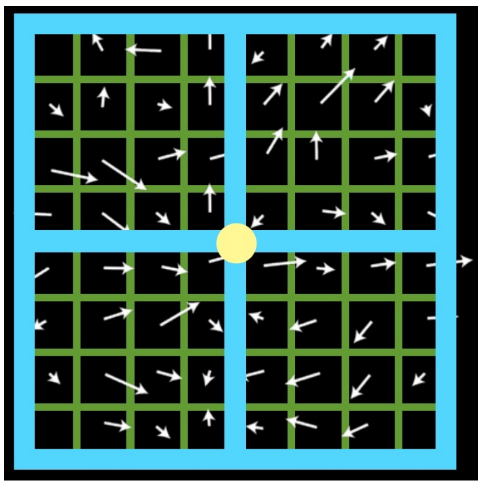
To extract a feature descriptor from a cell

- Compute Image Gradients



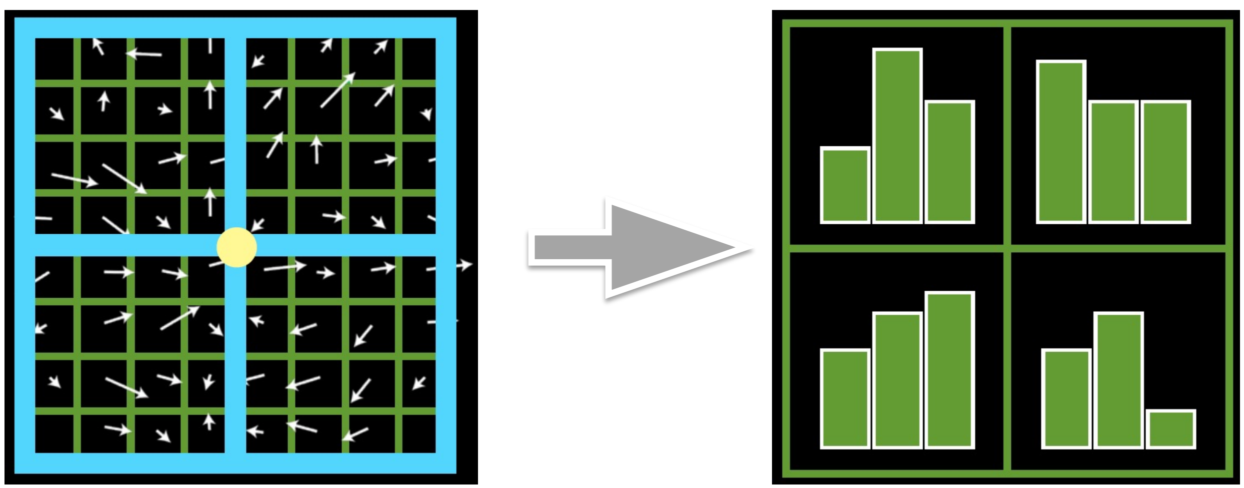
To extract a feature descriptor from a cell

- Compute Image Gradients
- Accumulate gradients along cells

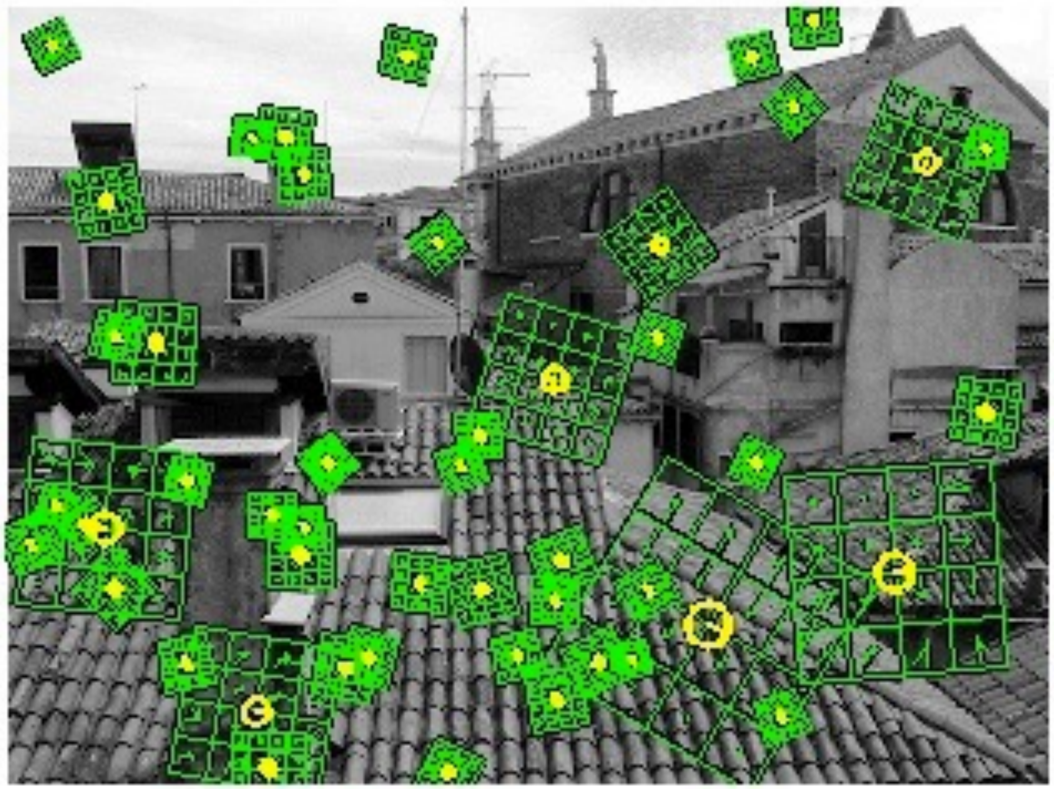


To extract a feature descriptor from a cell

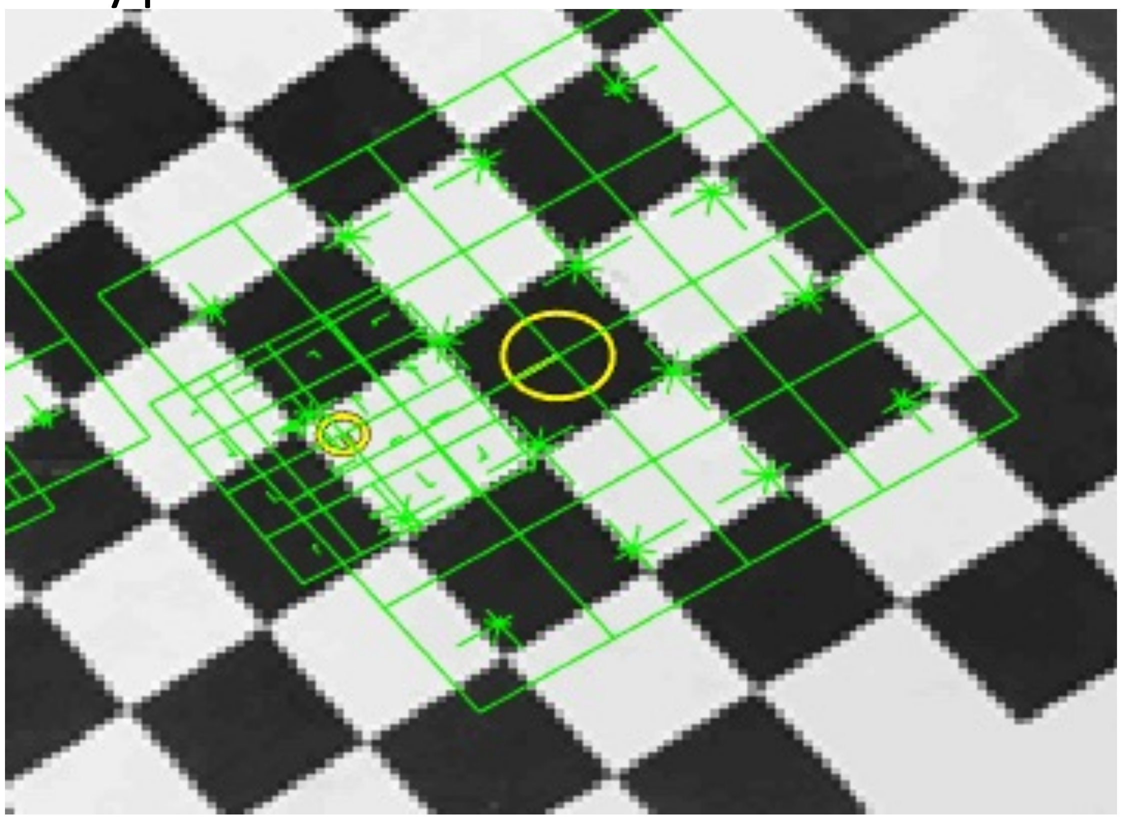
- Compute Image Gradients
- Accumulate gradients along cells
- Form image descriptor



As a matter of fact it is
a 4x4 grid of histograms at each keypoint

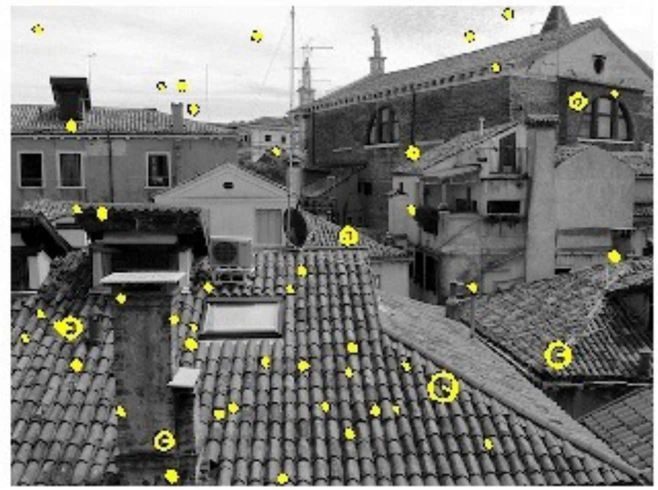


The descriptor is an 128x1 vector which together with σ,θ characterize the keypoint.



Example of SIFT detections and feature extraction

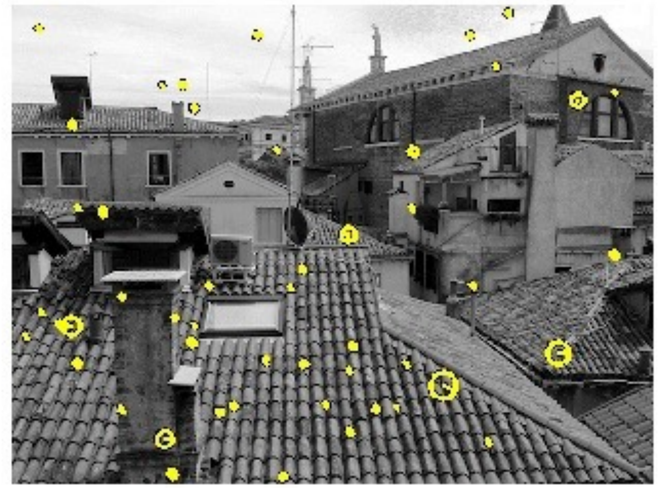
Input Image



Example Detections

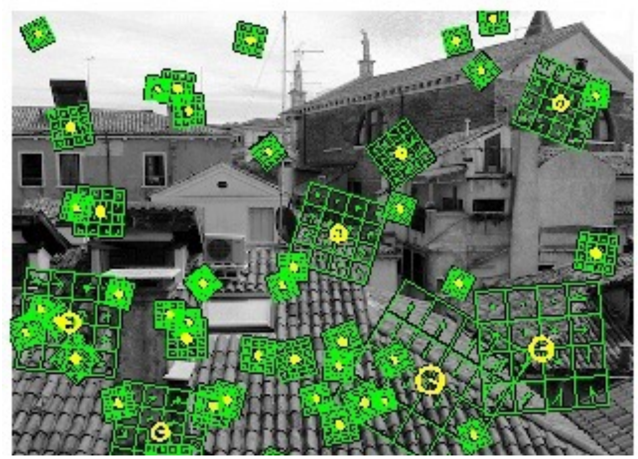
Example of SIFT detections and feature extraction

Input Image



Example Detections

Extracted Feature Descriptors

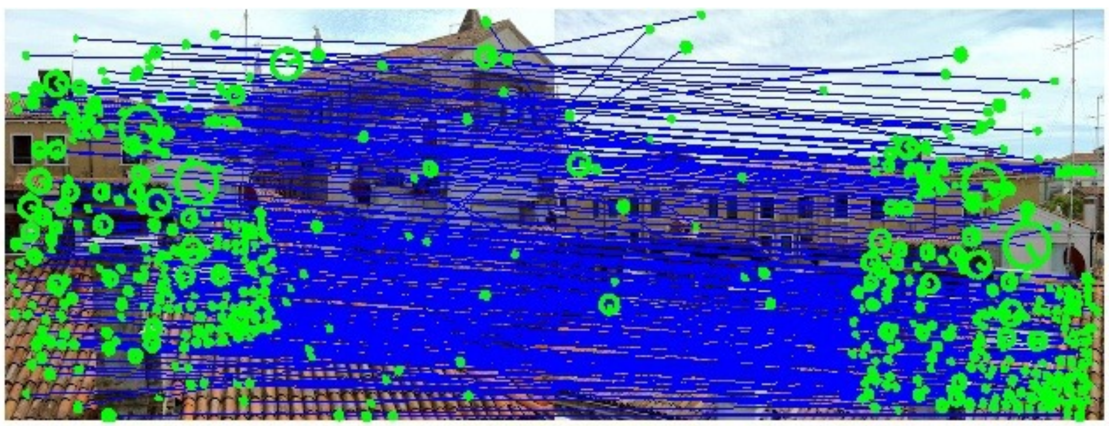


Using SIFT for image matching

Original Image Pair

Using SIFT for image matching

Original Image Pair



Matched features