Digital Image Processing

Image Segmentation: Points, Lines & Edges

Contents

So far we have been considering image processing techniques used to transform images for human interpretation

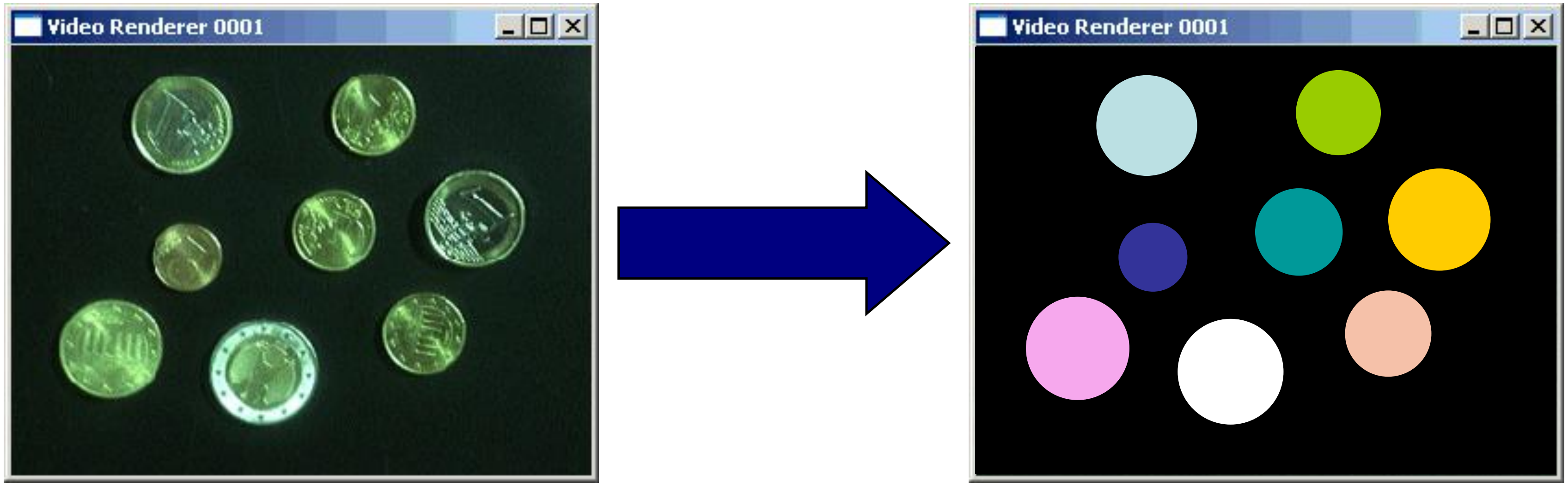
Today we will begin looking at automated image analysis by examining the thorny issue of image segmentation:

- The segmentation problem
- Finding points, lines and edges

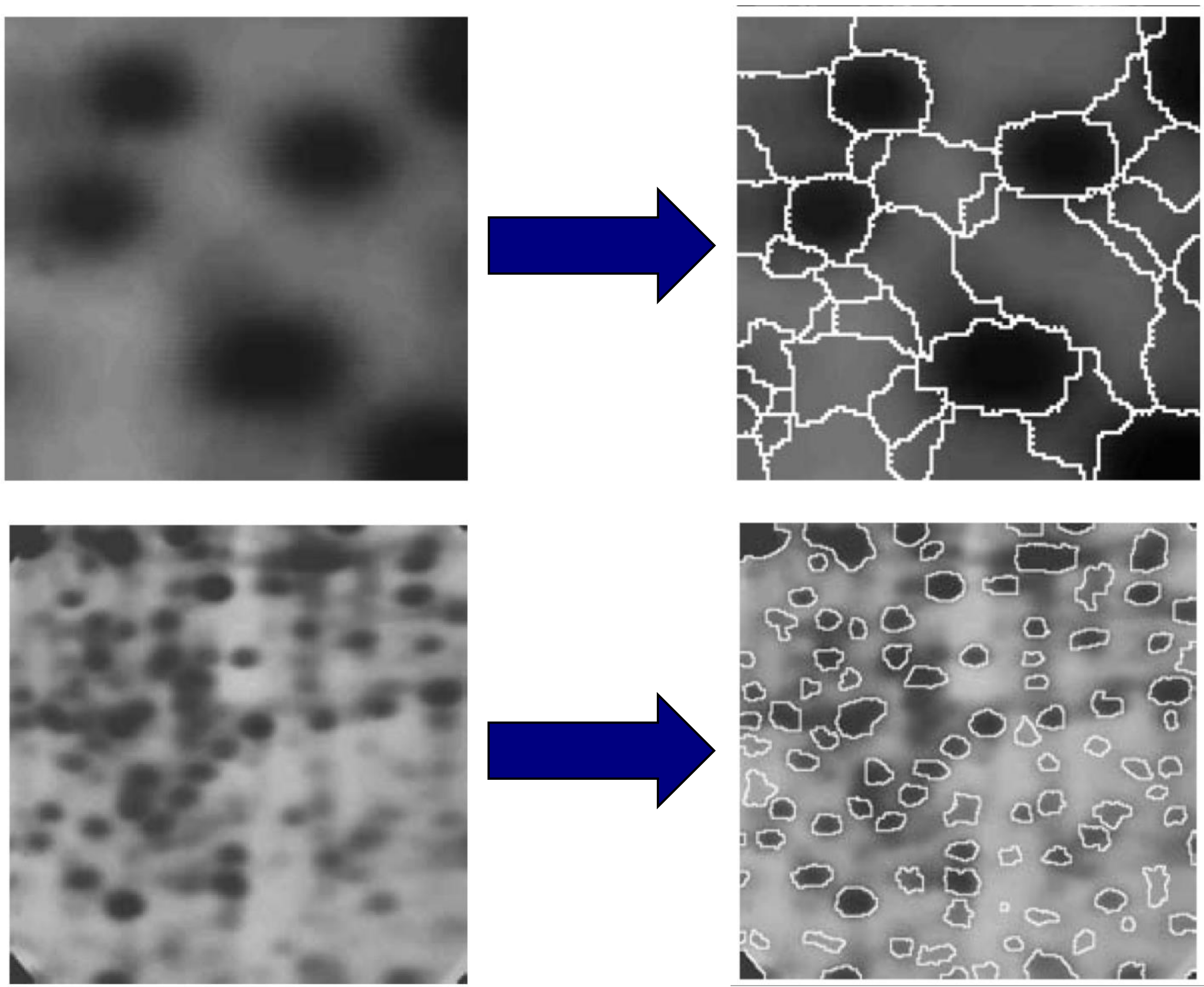
The Segmentation Problem

Segmentation attempts to partition the pixels of an image into groups that strongly correlate with the objects in an image

Typically the first step in any automated computer vision application



Segmentation Examples



Segmention

Algorithms are based on one of the 2 properties:

1. Discontinuity : partition an image based on the abrupt changes in intensity, such edges in an image.

2. Similarity : partitioning an image into regions that are similar according to a set of predefined criteria. Thresholding, region splitting and merging

Detection Of Discontinuities

There are three basic types of grey level discontinuities that we tend to look for in digital images:

- Points
- Lines
- Edges

We typically find discontinuities using masks and correlation

Point Detection

Point detection can be achieved simply using the mask below:

−1	−1	−1
−1	8	−1
−1	−1	−1

Points are detected at those pixels in the subsequent filtered image that are above a set threshold

Laplacian operations.

Point Detection

The response of the mask at any given point is given by

$$R = w_1 z_1 + w_2 z_2 + \cdots + w_9 z_9 = \sum_{i=1}^{9} w_i z_i$$

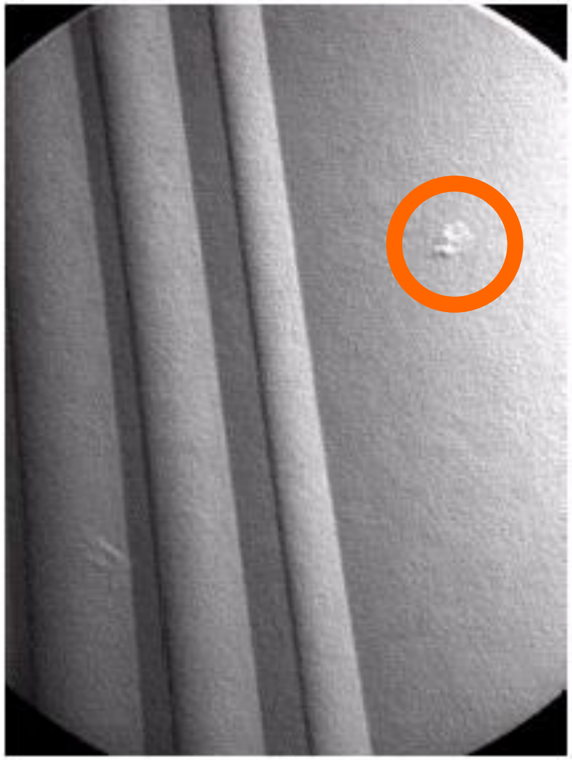
we say that a point has been detected at the location on which the mask is centered if

$$|R| \geq T$$

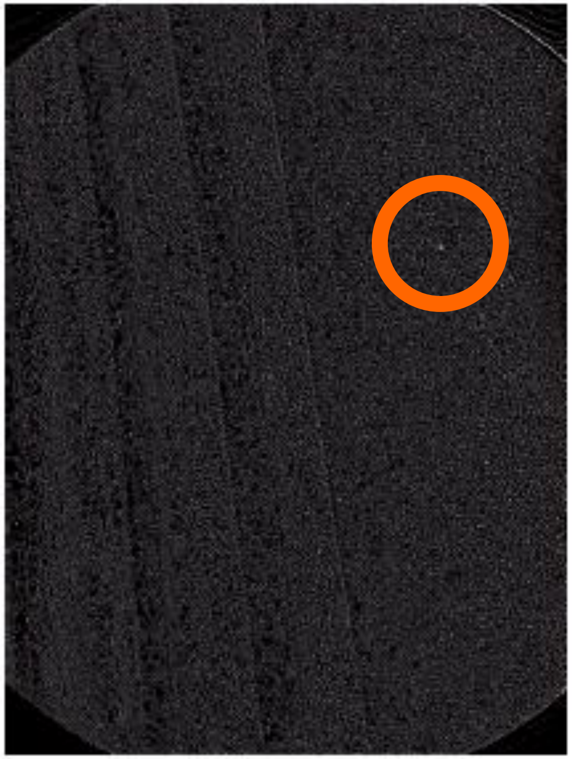
where T is a nonnegative threshold

The musk coefficients sum to zero. Mask response is zero in constant gray areas.

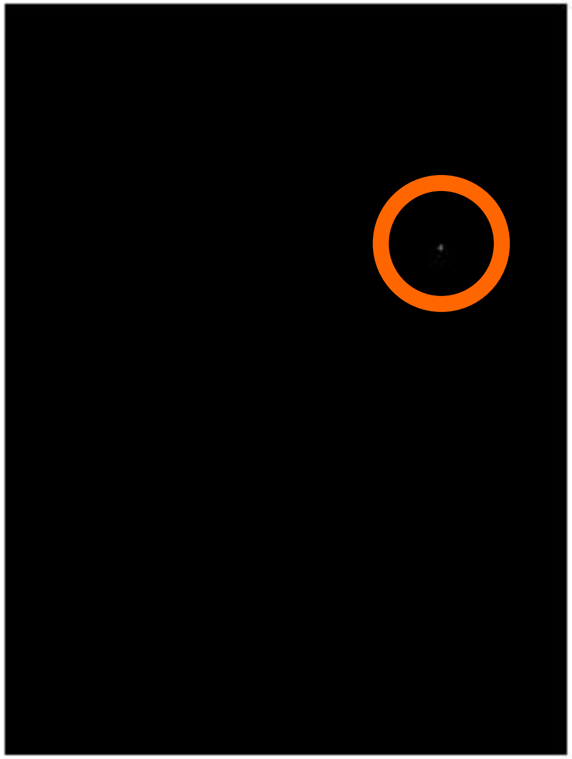
Point Detection (cont…)



X-ray image of a turbine blade



Result of point detection



Result of thresholding

Line Detection

The next level of complexity is to try to detect lines

The masks below will extract lines that are one pixel thick and running in a particular direction

−1	−1	−1
2	2	2
−1	−1	−1

Horizontal

−1	−1	2
−1	2	−1
2	−1	−1

+45°

−1	2	−1
−1	2	−1
−1	2	−1

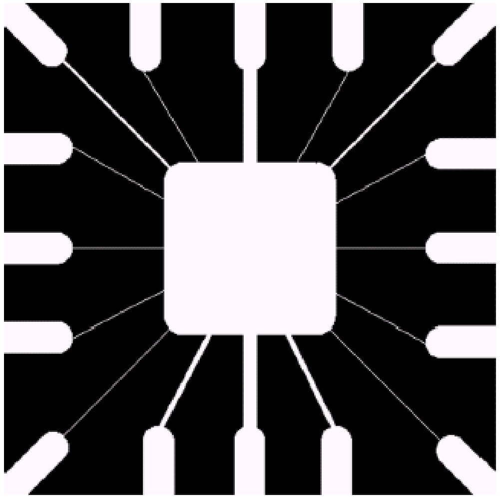
Vertical

2	−1	−1
−1	2	−1
−1	−1	2

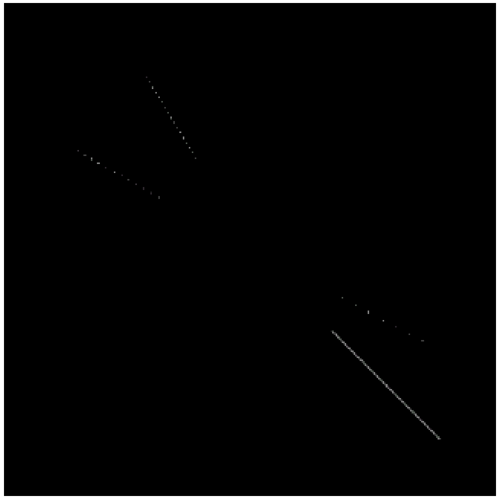
−45°

Line Detection (cont…)

Binary image of a wire bond mask



After processing with -45° line detector

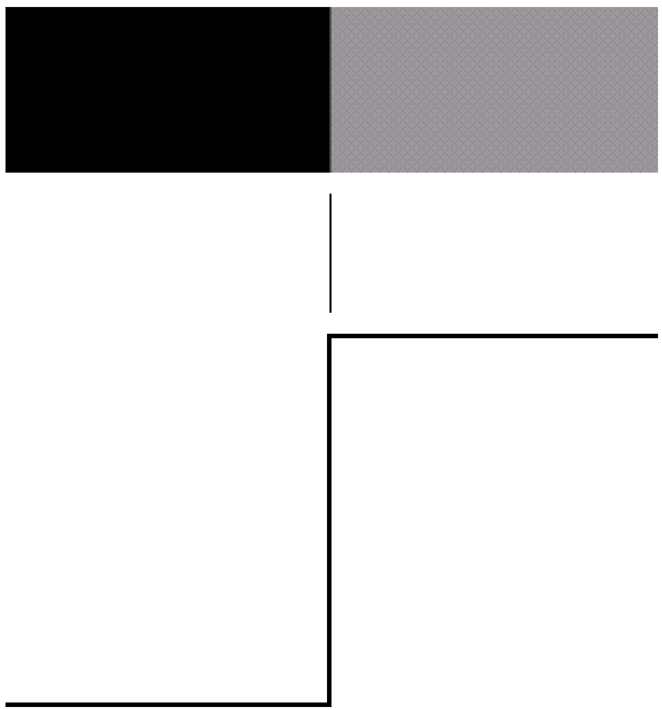


Result of thresholding filtering result

Edge Detection

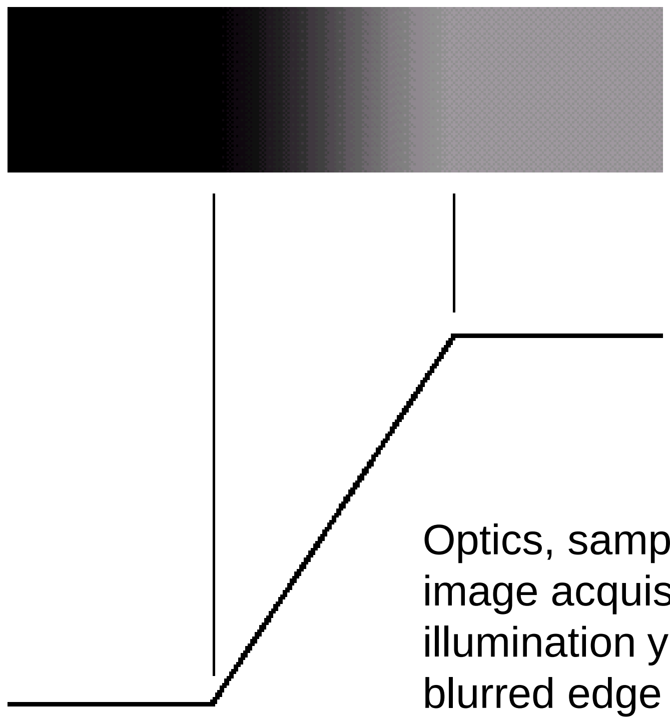
An edge is a set of connected pixels that lie on the boundary between two regions

Model of an ideal digital edge



Gray-level profile
of a horizontal line
through the image

Model of a ramp digital edge



Gray-level profile
of a horizontal line
through the image

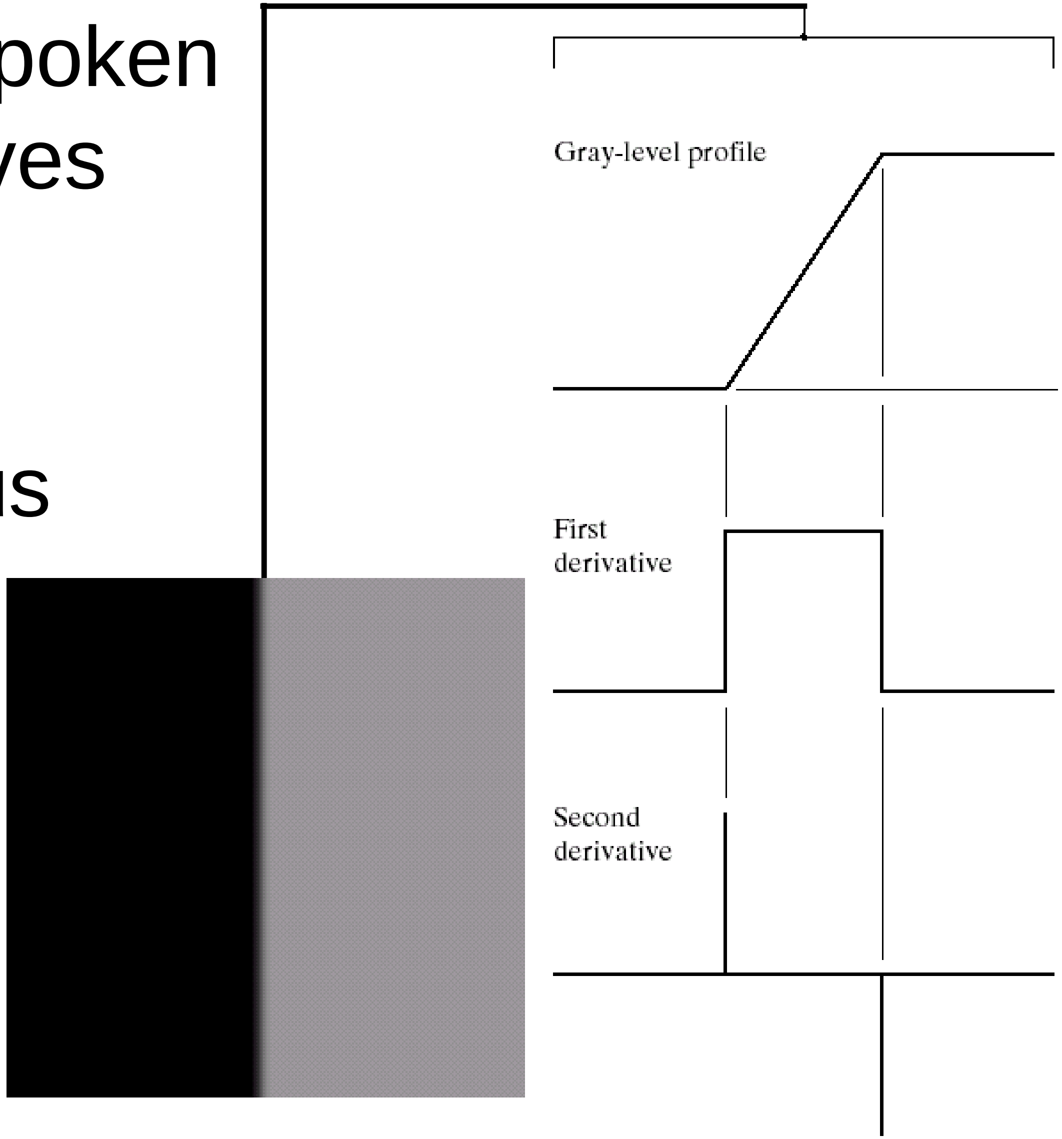
Optics, sampling,
image acquisition,
illumination yields
blurred edge

Edges & Derivatives

We have already spoken about how derivatives are used to find discontinuities

1st derivative tells us where an edge is

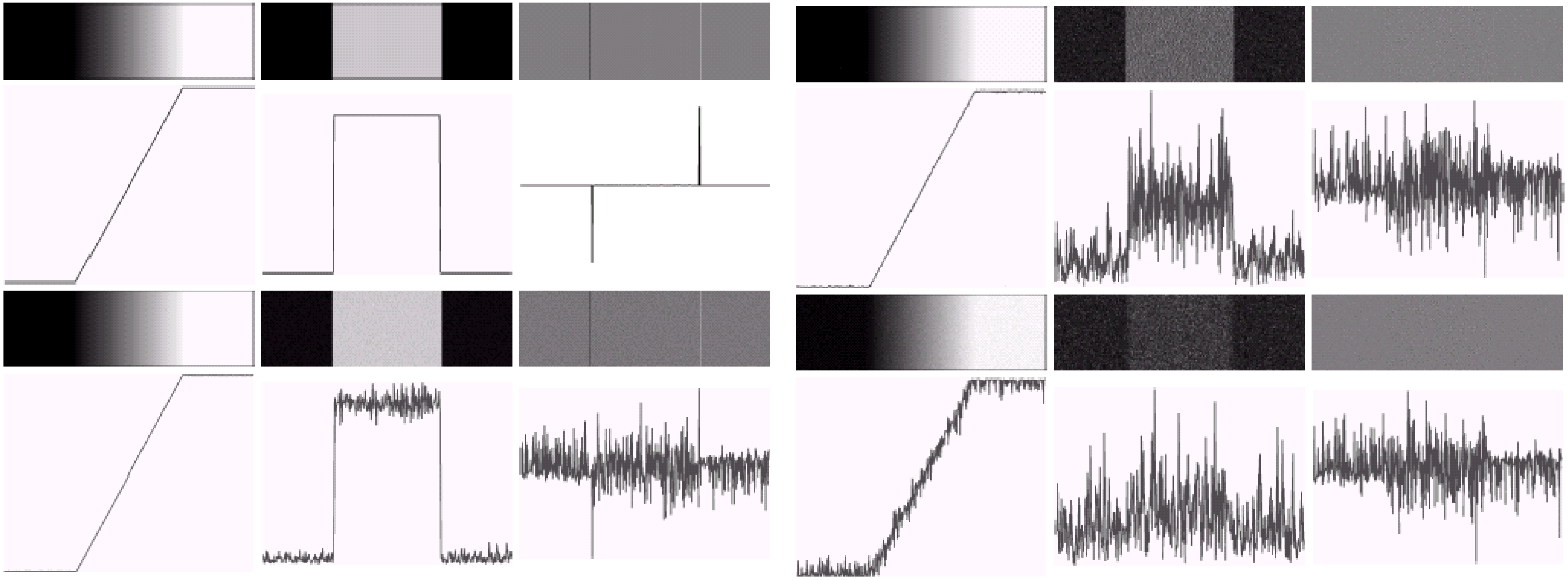
2nd derivative can be used to show edge direction



Derivatives & Noise

Derivative based edge detectors are extremely sensitive to noise

We need to keep this in mind



1st Derivative Filtering

Implementing 1st derivative filters is difficult in practice

For a function $f(x, y)$ the gradient of f at coordinates (x, y) is given as the column vector:

$$\nabla \mathrm{f} = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \dfrac{\partial f}{\partial x} \\ \dfrac{\partial f}{\partial y} \end{bmatrix}$$

1st Derivative Filtering (cont…)

The magnitude of this vector is given by:

$$\begin{aligned}\nabla f &= mag(\nabla \mathbf{f}) \\ &= \left[G_x^2 + G_y^2\right]^{1/2} \\ &= \left[\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2\right]^{1/2}\end{aligned}$$

For practical reasons this can be simplified as:

$$\nabla f \approx |G_x| + |G_y|$$

(Cont.....)

- the center point, z5 , denotes f(x, y), z1 denotes f(x-1, y-1), and so on. The simplest approximations to a first-order derivative that satisfy the conditions stated in that section are
-

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

$$G_x = (z_8 - z_5) \text{ and } G_y = (z_6 - z_5)$$

1^{st} Derivative Filtering (cont…)

Two other definitions proposed by Roberts [1965] in the early development of digital image processing use cross differences:

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

−1	0
0	1

0	−1
1	0

$$G_x = (z_9 - z_5) \quad \text{and} \quad G_y = (z_8 - z_6).$$

$$\nabla f = \left[(z_9 - z_5)^2 + (z_8 - z_6)^2\right]^{1/2}$$

$$\nabla f \approx |z_9 - z_5| + |z_8 - z_6|.$$

Common Edge Detectors

Given a 3*3 region of an image the following edge detection filters can be used

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

−1	−1	−1
0	0	0
1	1	1

−1	0	1
−1	0	1
−1	0	1

Prewitt

−1	0
0	1

0	−1
1	0

Roberts

−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

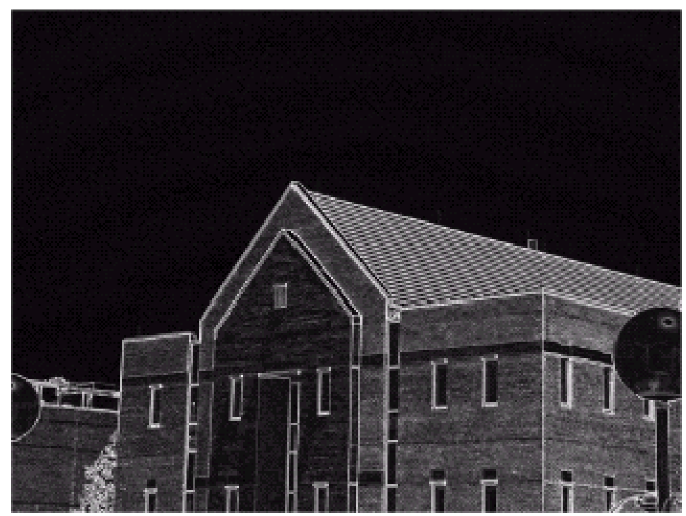
Sobel

Edge Detection Example

Original Image

Horizontal Gradient Component

Vertical Gradient Component



Combined Edge Image

Edge Detection Problems

Often, problems arise in edge detection in that there are is too much detail

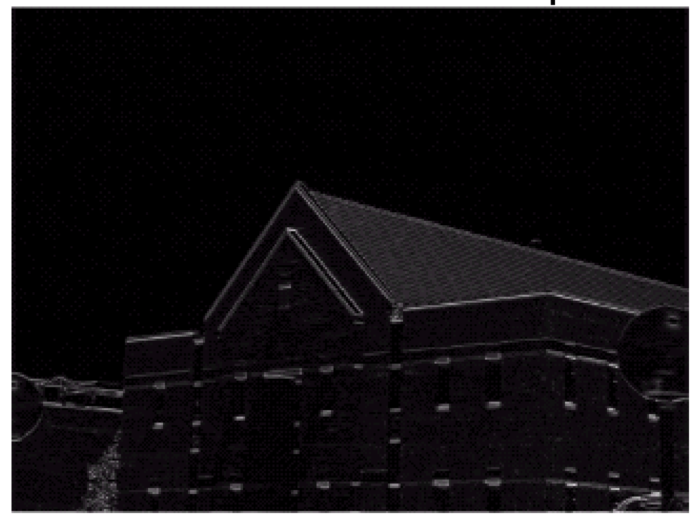
For example, the brickwork in the previous example

One way to overcome this is to smooth images prior to edge detection

Edge Detection Example With Smoothing

Original Image

Horizontal Gradient Component



Vertical Gradient Component

Combined Edge Image

The Laplacian

The Laplacian of a 2-D function $f(x, y)$ is a second-order derivative defined as

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}.$$

Which is approximated by the relation

$$\nabla^2 f = 4z_5 - (z_2 + z_4 + z_6 + z_8)$$

Or

$$\nabla^2 f = 8z_5 - (z_1 + z_2 + z_3 + z_4 + z_6 + z_7 + z_8 + z_9).$$

Laplacian Edge Detection

We encountered the 2nd-order derivative based Laplacian filter already

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

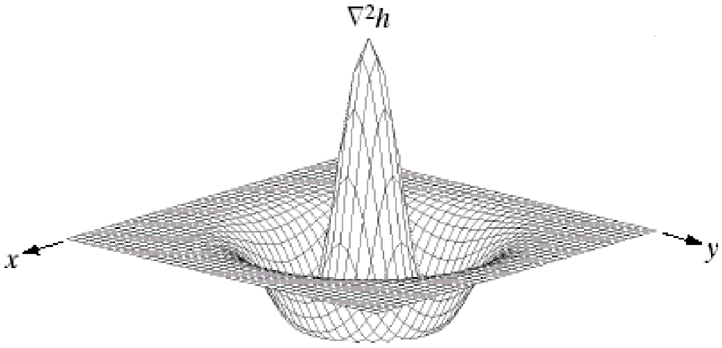
Which are isotropic for rotation increments of 90° and 45°, respectively.

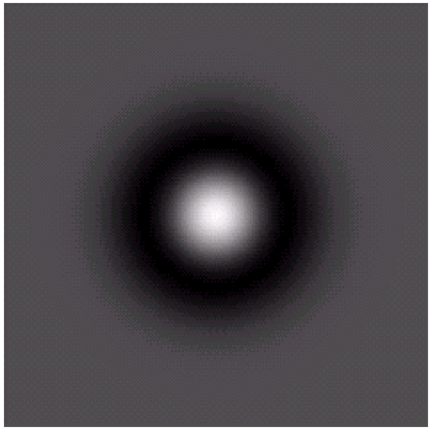
The Laplacian is typically not used by itself as it is too sensitive to noise

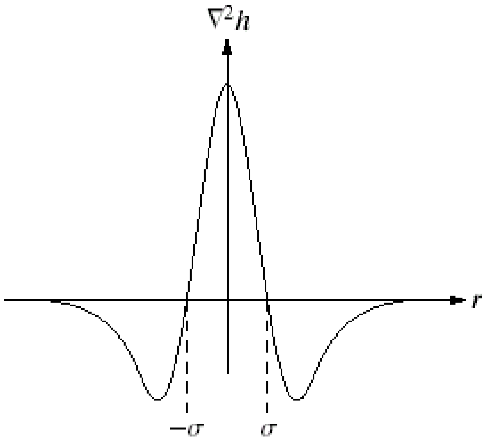
Usually hen used for edge detection the Laplacian is combined with a smoothing Gaussian filter

Laplacian Of Gaussian

The Laplacian of Gaussian (or Mexican hat) filter uses the Gaussian for noise removal and the Laplacian for edge detection

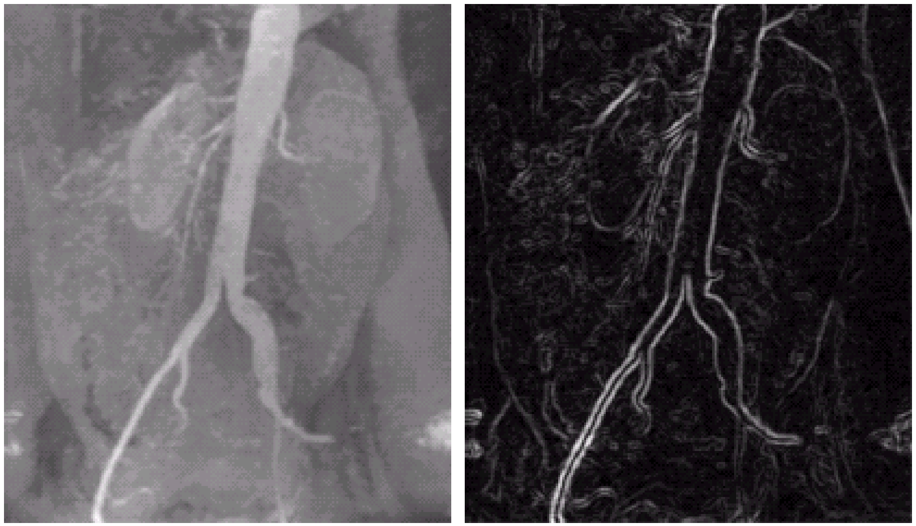


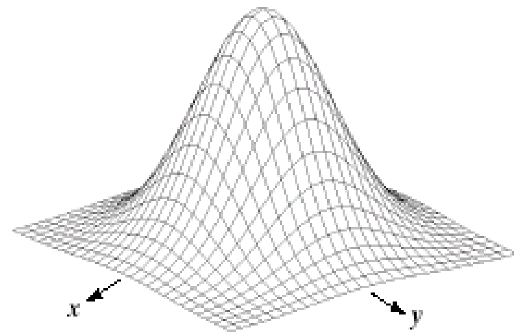




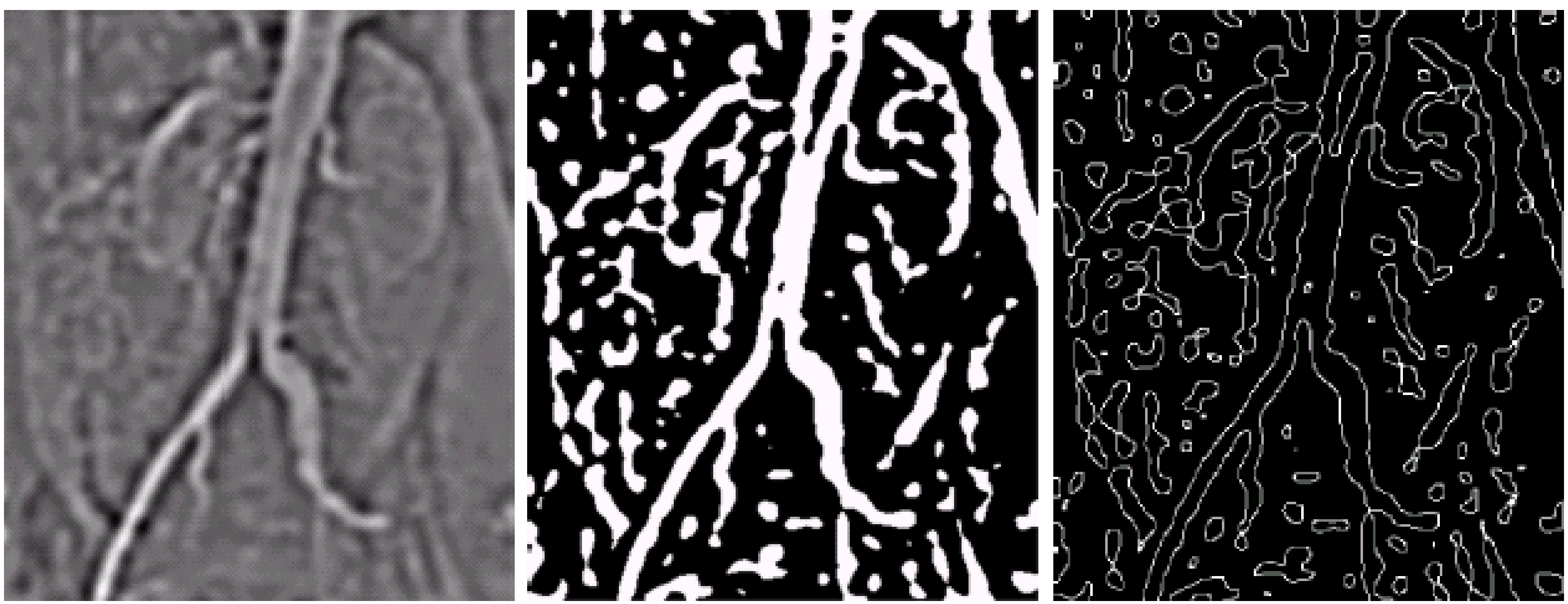
0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0

Laplacian Of Gaussian Example





−1	−1	−1
−1	8	−1
−1	−1	−1



Summary

In this lecture we have begun looking at segmentation, and in particular edge detection

Edge detection is massively important as it is in many cases the first step to object recognition