Digital Image Processing

Image Segmentation: Thresholding

Abdullah Al Shiam
Assistant Professor
Computer Science and Engineering
Netrokona University
shiam.cse@neu.ac.bd

Contents

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

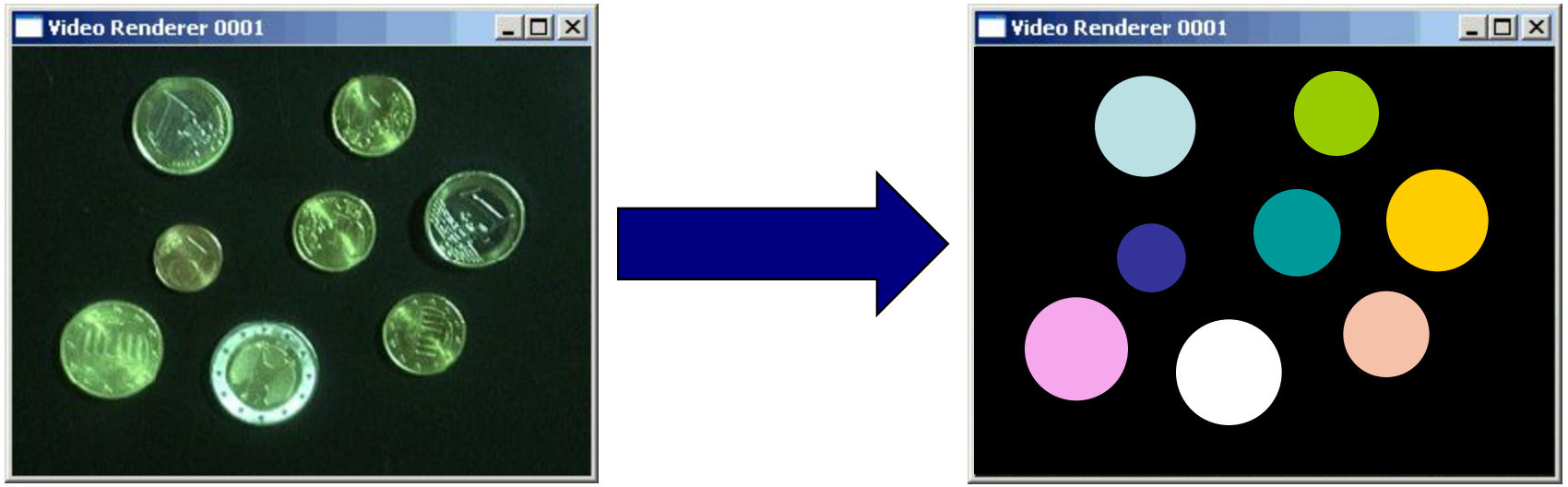
Today we will begin looking at automated image analysis by examining the thorny issue of image segmentation:

- The segmentation problem
- Finding points, lines and edges

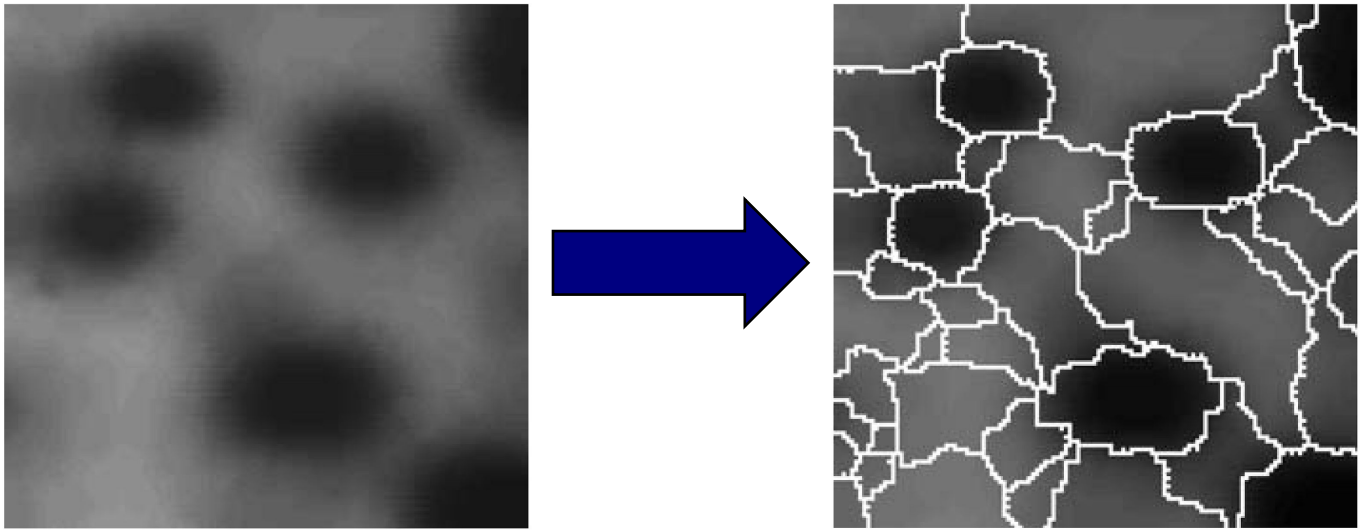
The Segmentation Problem

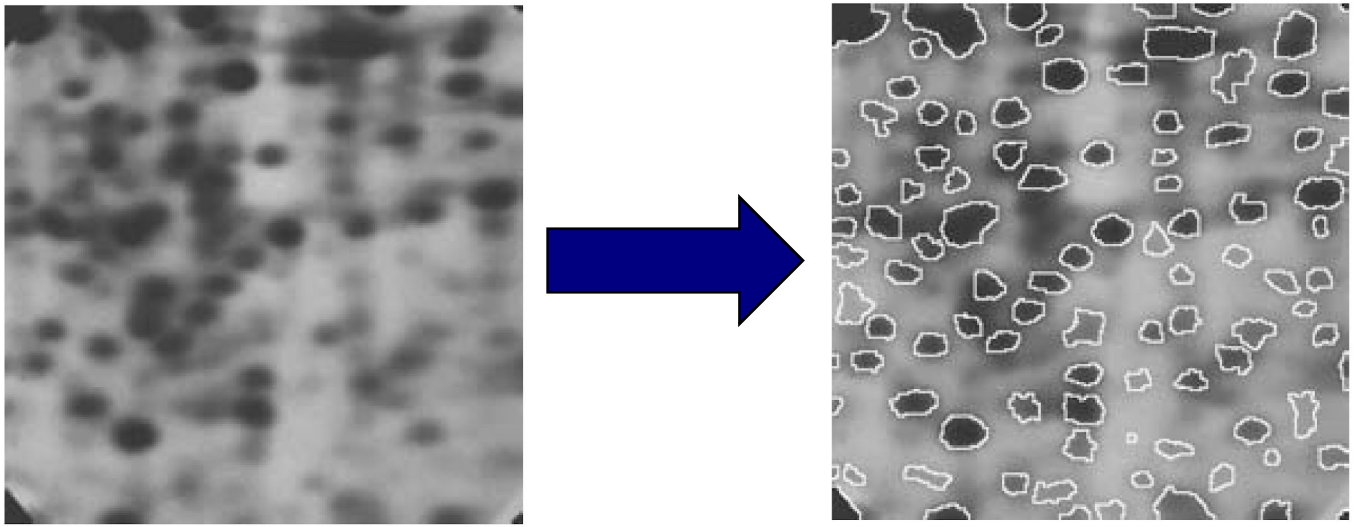
Segmentation attempts to partition the pixels of an image into groups that strongly correlate with the objects in an image

Typically the first step in any automated computer vision application



Segmentation Examples





Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Detection Of Discontinuities

- Image segmentation algorithms generally are based on one of two basic properties of intensity values: discontinuity and similarity.

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

Discontinuity based approach: Partition an image based on abrupt changes in intensity.

- Similarity based approach: Partition an image based on regions that are similar according to a set of predefined criteria.
  - Thresholding
  - Region growing
  - Region splitting and merging

Point Detection

This procedure involves computing the sum of products of the coefficients with the gray levels contained in the region encompassed by the mask. That is, the response of the mask at any point in the image

Where is the gray level of the pixel associated with mask coefficient As Wi. usual, the response of the mask is defined with respect to its center location. When the mask is centered on a boundary pixel, the response is computed by using the appropriated partial neighborhood.

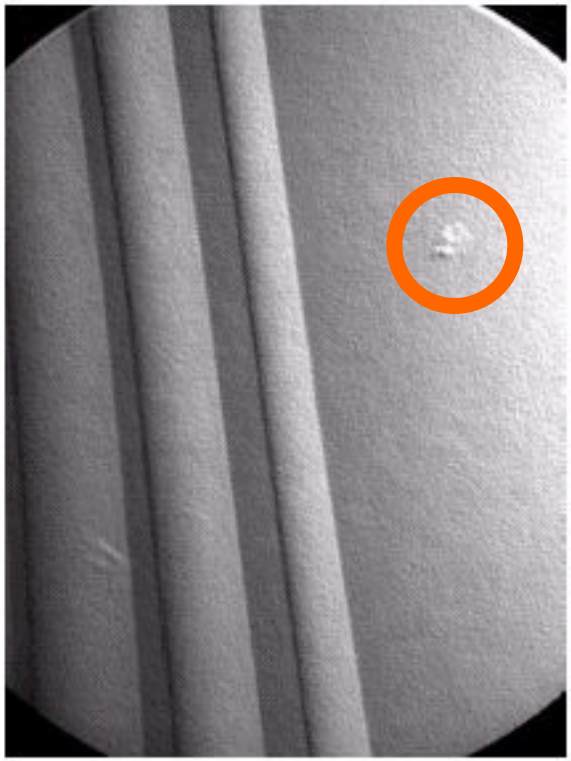
Point detection can be achieved simply using the mask below:

−1	−1	−1
−1	8	−1
−1	−1	−1

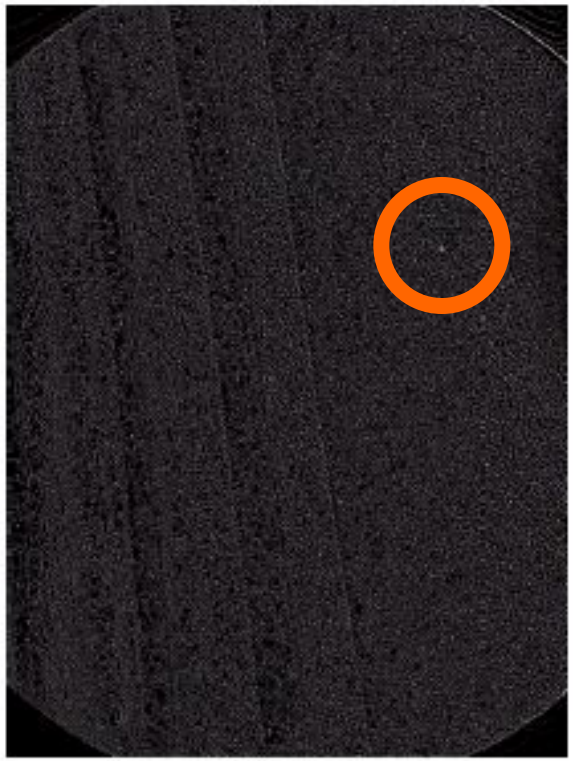
A point has been detected at the location p(i, j) on which the mask is centered if |R |>T, where T is a nonnegative threshold, and R is the response of the mask at any point in the image.

Basically all that this formulation does is measure the weighted differences between the center point and its neighbors. The idea is that the gray level of an isolated point will be quite different from the gray level of its neighbors.

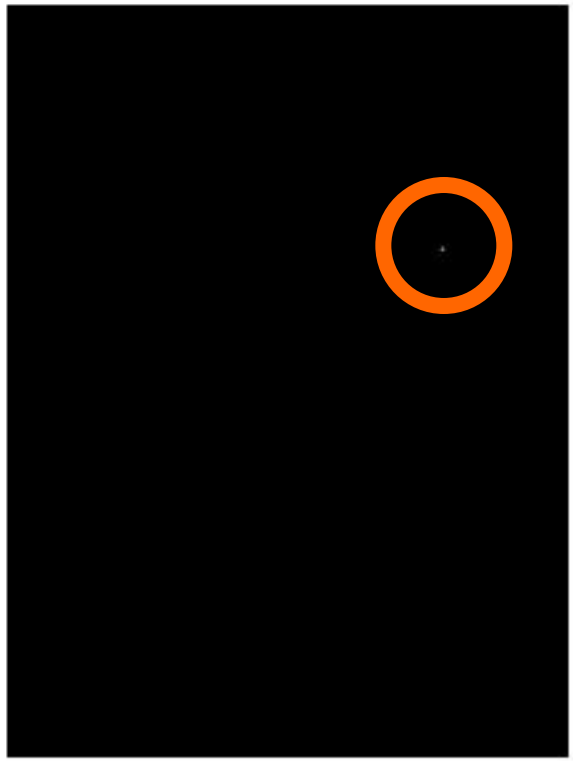
Point Detection (cont…)



X-ray image of
a turbine blade



Result of point
detection



Result of
thresholding

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Line Detection

This procedure involves computing the sum of products of the coefficients with the gray levels contained in the region encompassed by the mask. That is, the response of the mask at any point in the image

Where is the gray level of the pixel associated with mask coefficient As Wi. usual, the response of the mask is defined with respect to its center location. When the mask is centered on a boundary pixel, the response is computed by using the appropriated partial neighborhood.

Most of the earlier methods for detecting lines were based on pattern matching. These pattern templates are designed with suitable coefficients and are applied at each point in an image. A set of such templates is shown in. **Line detection can be achieved simply using the mask below:**

−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

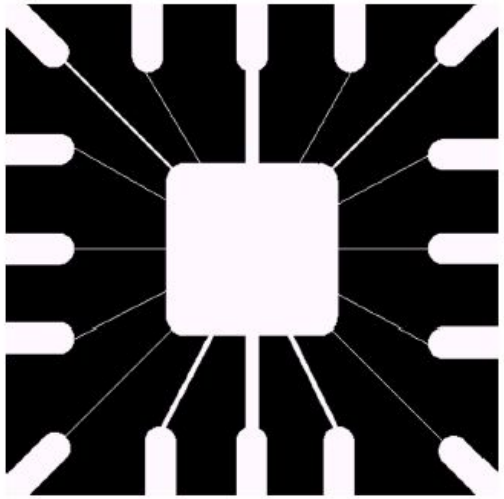
2	−1	−1
−1	2	−1
−1	−1	2

−45°

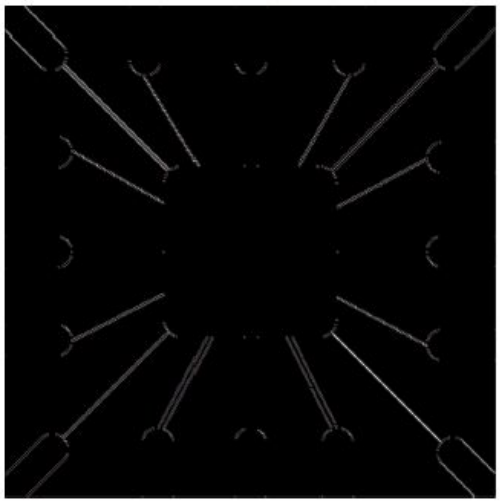
If the first mask were moved around an image, it would respond more strongly to lines oriented horizontally. With constant background, the maximum response would result when the line passed through the middle row of the mask. This is easily verified by sketching a simple array of 1's with a line of a different gray level running horizontally through the array. A similar experiment would show that the second mask in responds best to lines oriented at +45; the third mask to vertical lines; and the fourth mask to lines in the – 45 direction. These directions can also be established by refer that the preferred direction of each mask is weighted with larger coefficient i.e., 2 than other possible directions.

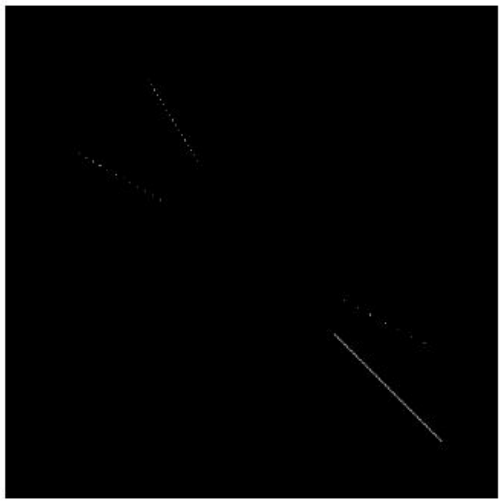
Line Detection (cont…)

Binary image of a wire bond mask



After processing with -45° line detector





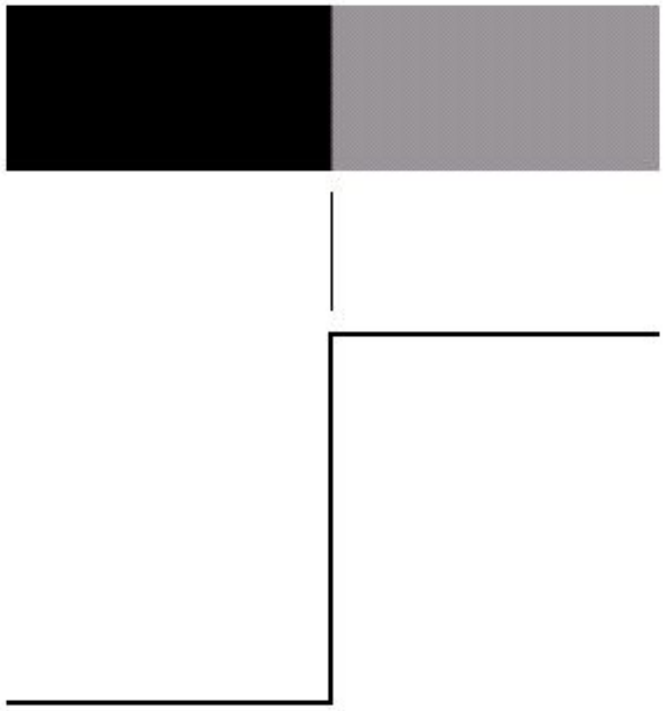
Result of thresholding filtering result

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Edge Detection

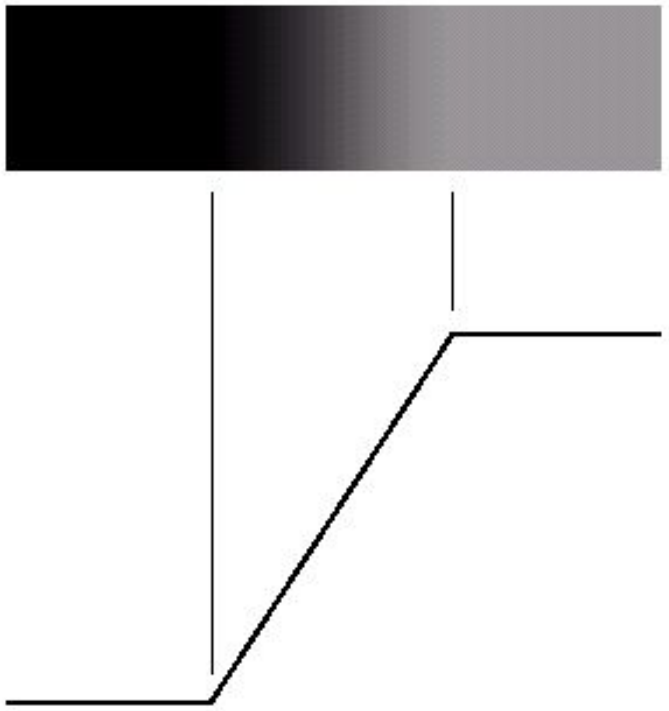
An edge is a set of connected pixels that lie on the boundary between two regions. boundary between two regions with relatively distinct gray levels.

Model of an ideal digital edge



Gray-level profile of a horizontal line through the image

Model of a ramp digital edge



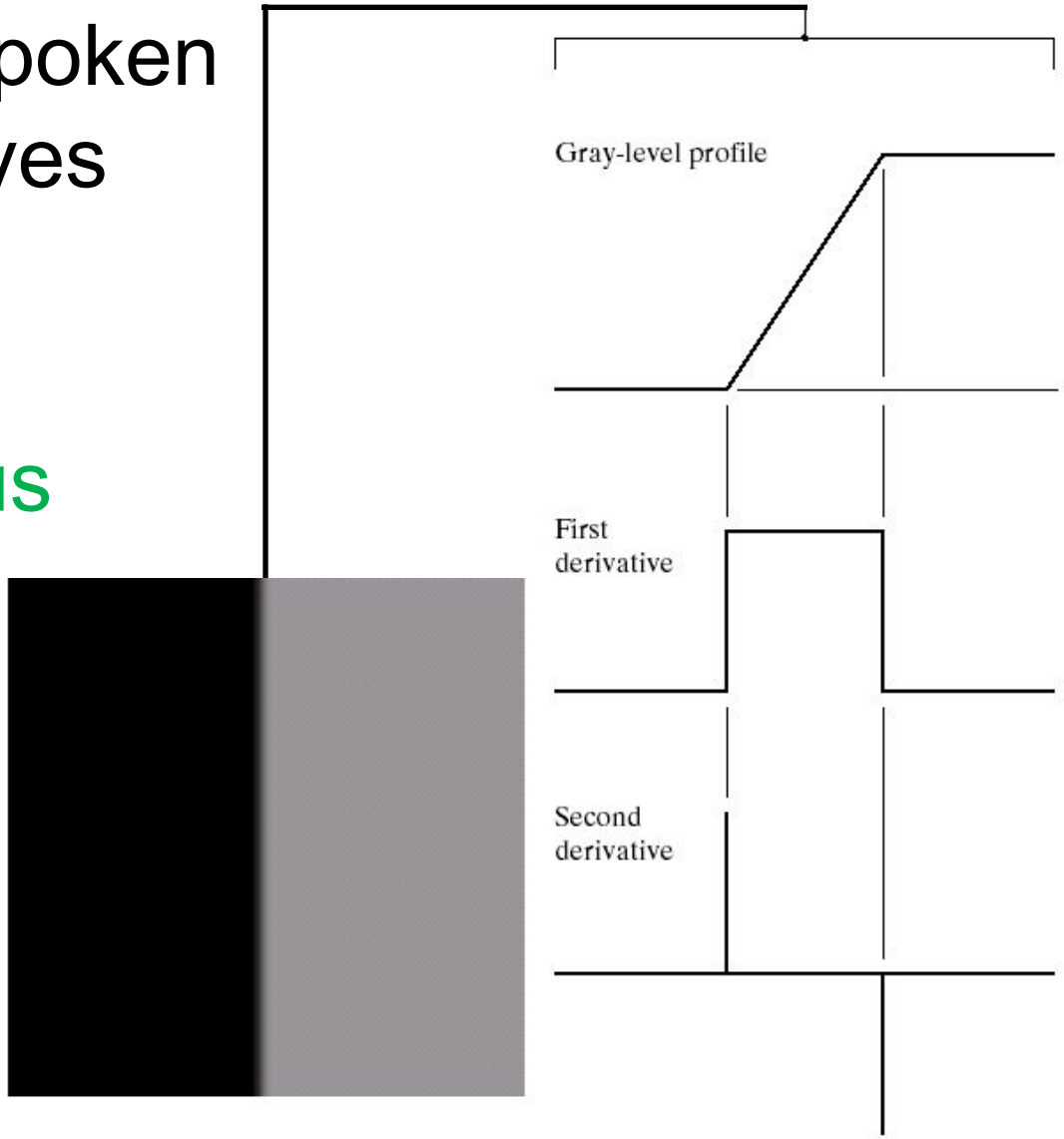
Gray-level profile of a horizontal line through the image

Edges & Derivatives

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

1st derivative tells us where an edge is

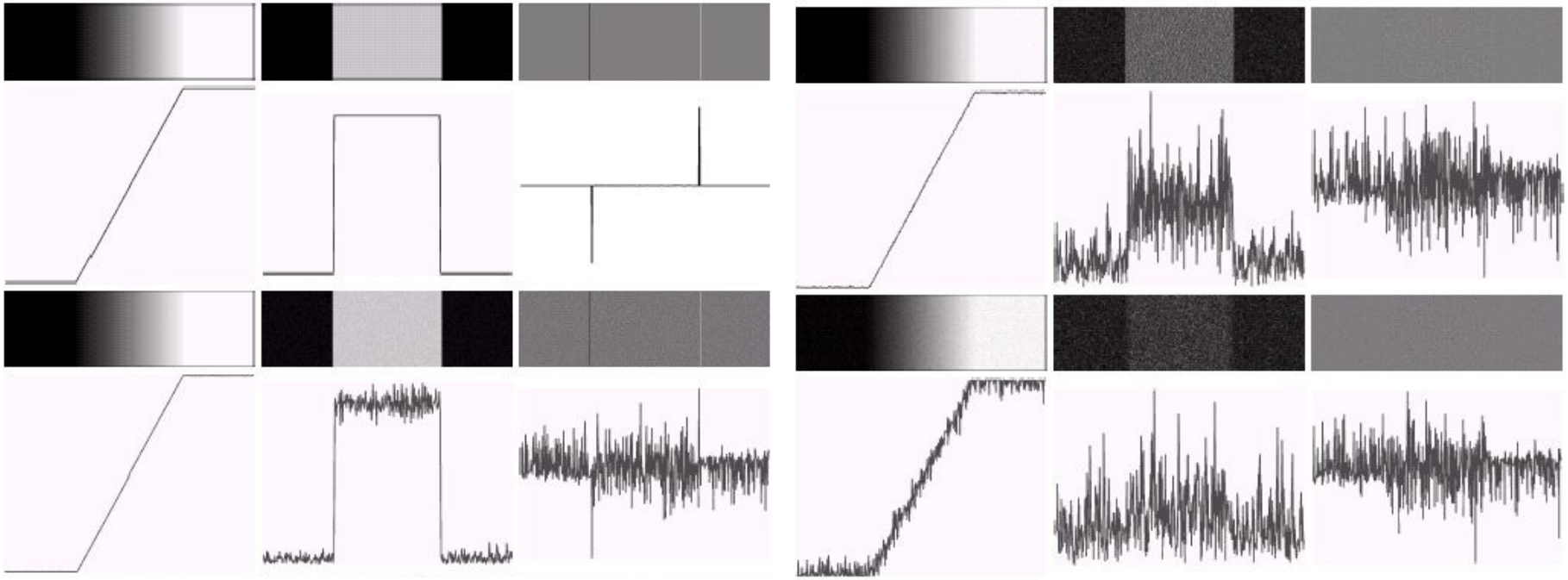
2nd derivative can be used to show edge direction



Derivatives & Noise

Derivative based edge detectors are extremely sensitive to noise

We need to keep this in mind



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

Prewitt Operator

Prewitt operator is used for edge detection in an image. It detects two types of edges.

1.Horizontal edges

2.Vertical Edges

Edges are calculated by using difference between corresponding pixel intensities of an image. All the masks that are used for edge detection are also known as derivative masks

Vertical direction:

−1	0	1
−1	0	1
−1	0	1

Above mask will find the edges in vertical direction and it is because the zeros column in the vertical direction. When you will convolve this mask on an image, it will give you the vertical edges in an image.

How it works:

When we apply this mask on the image it distinguished vertical edges. It simply works like as first order derivate and calculates the difference of pixel intensities in a edge region. As the center column is of zero so it does not include the original values of an image but rather it calculates the difference of right and left pixel values around that edge. This increase the edge intensity and it become enhanced comparatively to the original image.

Horizontal Direction

−1	−1	−1
0	0	0
1	1	1

Above mask will find edges in horizontal direction and it is because that zeros column is in horizontal direction. When you will convolve this mask onto an image it would distinguished horizontal edges in the image.

How it works:

This mask will distinguished the horizontal edges in an image. It also works on the principle of above mask and calculates difference among the pixel intensities of a particular edge. As the center row of mask is consist of zeros so it does not include the original values of edge in the image but rather it calculate the difference of above and below pixel intensities of the particular edge. Thus increasing the sudden change of intensities and making the edge more visible. Both the above masks follow the principle of derivate mask.

Sobel Operators

Based on the previous equations we can derive the *Sobel Operators*

-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

To filter an image it is filtered using both operators the results of which are added together

Sobel Operators

- The sobel operator is very similar to Prewitt operator. It is also a derivate mask and is used for edge detection. Like Prewitt operator sobel operator is also used to detect two kinds of edges in an image:
- Vertical direction
- Horizontal direction
- Following is the vertical Mask of Sobel Operator:

-1	0	1
-2	0	2
-1	0	1

- When applied on an image this mask will highlight the vertical edges

How it works

When we apply this mask on the image it prominent vertical edges. It simply works like as first order derivate and calculates the difference of pixel intensities in a edge region.

As the center column is of zero so it does not include the original values of an image but rather it calculates the difference of right and left pixel values around that edge. Also the center values of both the first and third column is 2 and -2 respectively.

This give more weight age to the pixel values around the edge region. This increase the edge intensity and it become enhanced comparatively to the original image.

- Following is the horizontal Mask of Sobel Operator

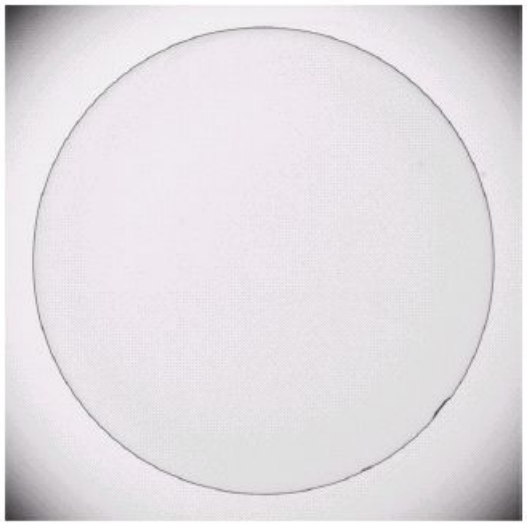
-1	-2	-1
0	0	0
1	2	1

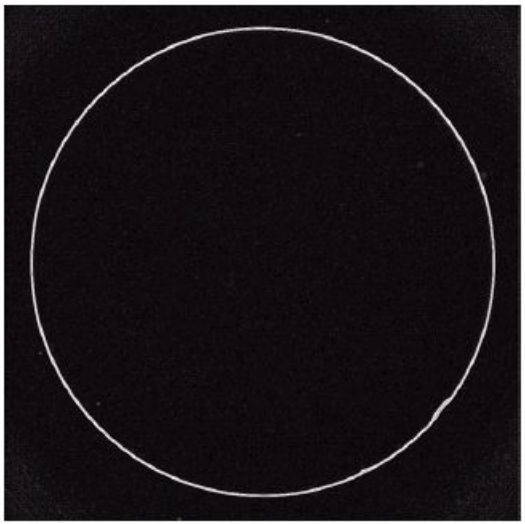
- Above mask will find edges in horizontal direction and it is because that zeros column is in horizontal direction. When you will convolve this mask onto an image it would prominent horizontal edges in the image. The only difference between it is that it have 2 and -2 as a center element of first and third row.

How it works

- This mask will prominent the horizontal edges in an image. It also works on the principle of above mask and calculates difference among the pixel intensities of a particular edge. As the center row of mask is consist of zeros so it does not include the original values of edge in the image but rather it calculate the difference of above and below pixel intensities of the particular edge. Thus increasing the sudden change of intensities and making the edge more visible.

Sobel Example



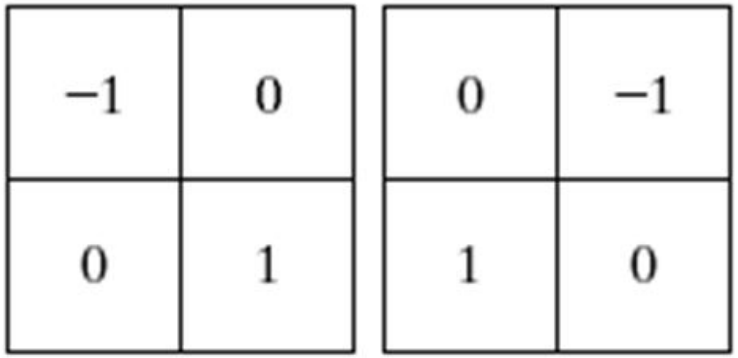


An image of a contact lens which is enhanced in order to make defects (at four and five o'clock in the image) more obvious

Sobel filters are typically used for edge detection

Roberts Operator

- The **Roberts cross** operator is used in image processing operator is used in image processing and computer vision operator is used in image processing and computer vision for edge detection. It was one of the first edge detectors.
- As a differential operatorAs a differential operator, the idea behind the Roberts cross operator is to approximate the gradient of an image through discrete differentiation which is achieved by computing the sum of the squares of the differences between diagonally adjacent pixels. The results of this operation will highlight changes in intensity in a diagonal direction. One of the most appealing aspects of this operation is its simplicity; the kernel is small and contains only integers.



Roberts

Difference Sobel and Prewitt Operator

- The major difference is that in sobel operator the coefficients of masks are not fixed and they can be adjusted according to our requirement unless they do not violate any property of derivative masks.

- There is only one difference that is it has “2” and “-2” values in center of first and third column.

- sobel operator finds more edges or make edges more visible as compared to Prewitt Operator.
- Sobel operator give the more weight to the edges than prewitt operator..

Edge Detection Example

Original Image

Horizontal Gradient Component

Vertical Gradient Component

Combined Edge Image

Edge Detection Example

Edge Detection Example

Edge Detection Example

Edge Detection Example

Edge Detection Problems

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

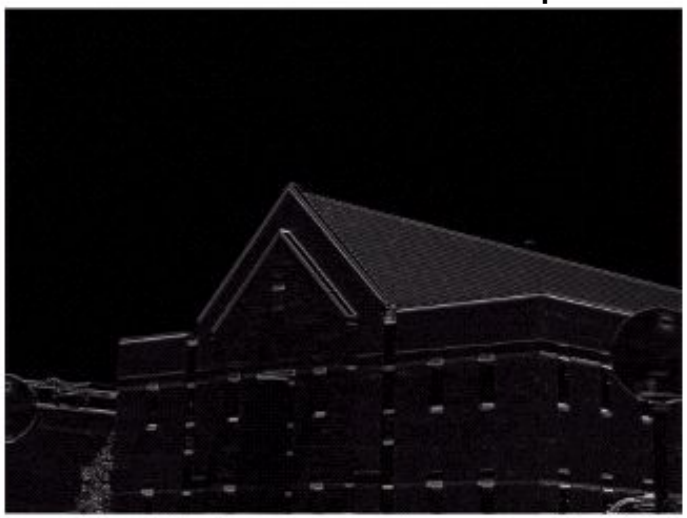
For example, the brickwork in the previous example

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

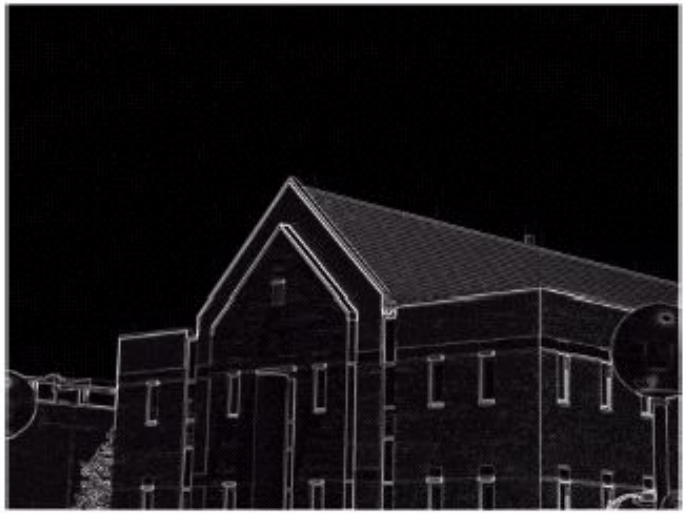
Edge Detection Example With Smoothing

Original Image

Horizontal Gradient Component



Vertical Gradient Component



Combined Edge Image

Laplacian Edge Detection

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

0	−1	0
−1	4	−1
0	−1	0

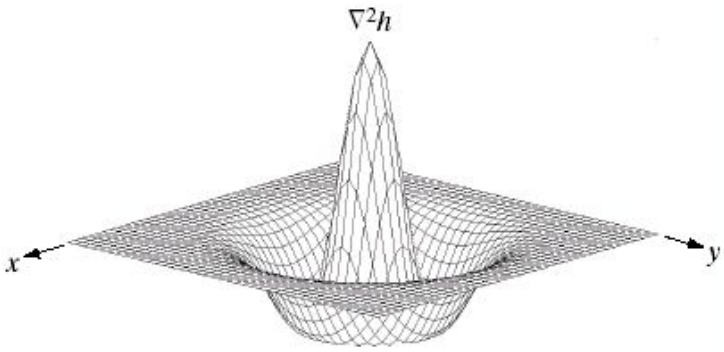
−1	−1	−1
−1	8	−1
−1	−1	−1

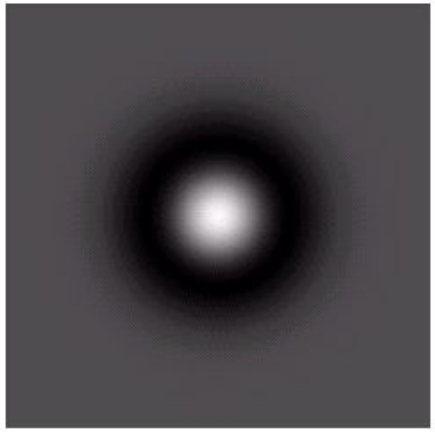
The Laplacian is typically not used by itself as it is too sensitive to noise

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

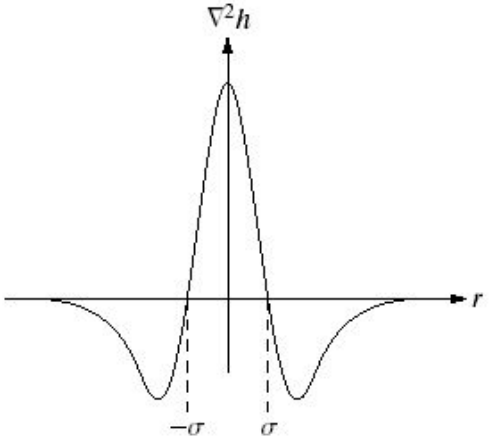
Laplacian Of Gaussian

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 filters are derivative filters used to find areas of rapid change (edges) in images. Since derivative filters are very sensitive to noise, it is common to smooth the image (e.g., using a Gaussian filter) before applying the Laplacian. This two-step process is call the Laplacian of Gaussian (LoG) operation.

$$L(x,y) = \nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2}$$

There are different ways to find an approximate discrete convolution kernel that approximates the effect of the Laplacian. A possible kernel is

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

This is called a positive Laplacian because the central peak is negative.

To include a smoothing Gaussian filter, combine the Laplacian and Gaussian functions to obtain a single equation:

A discrete kernel for the case of σ = 1.4 is given by

$$\mathrm{LoG}(x,y) = -\frac{1}{\pi\sigma^4}\left[1-\frac{x^2+y^2}{2\sigma^2}\right]e^{-\frac{x^2+y^2}{2\sigma^2}}$$

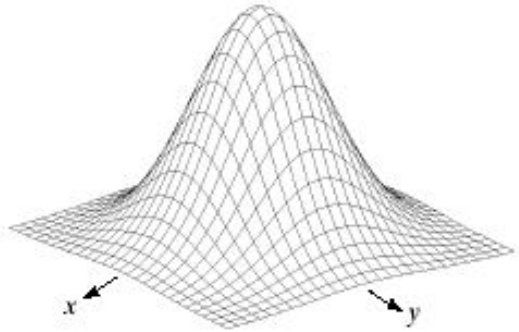
0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0

Wherever a change occurs, the LoG will give a positive response on the darker side and a negative response on the lighter side. At a sharp edge between two regions, the response will be

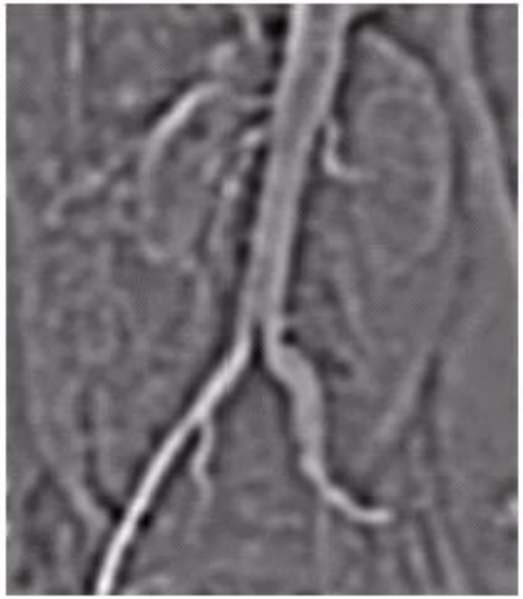
- zero away from the edge
- positive just to one side
- negative just to the other side
- zero at some point in between on the edge itself

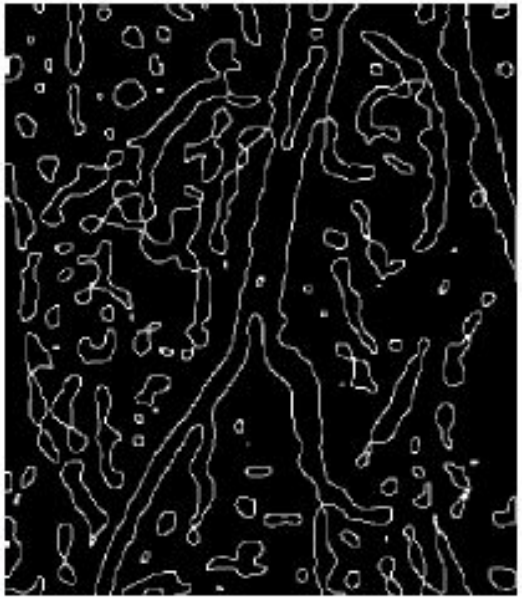
When using the filter given above, the output can contain values that are quite large and may be negative, so it is important to use an image type that supports negatives and a large range, and then scale the output. Alternatively, a scaling factor can be used on the filter to restrict the range of values.

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