



UNIVERSITÀ DEGLI STUDI DI PADOVA

Your first edge detection algorithm

Stefano Ghidoni

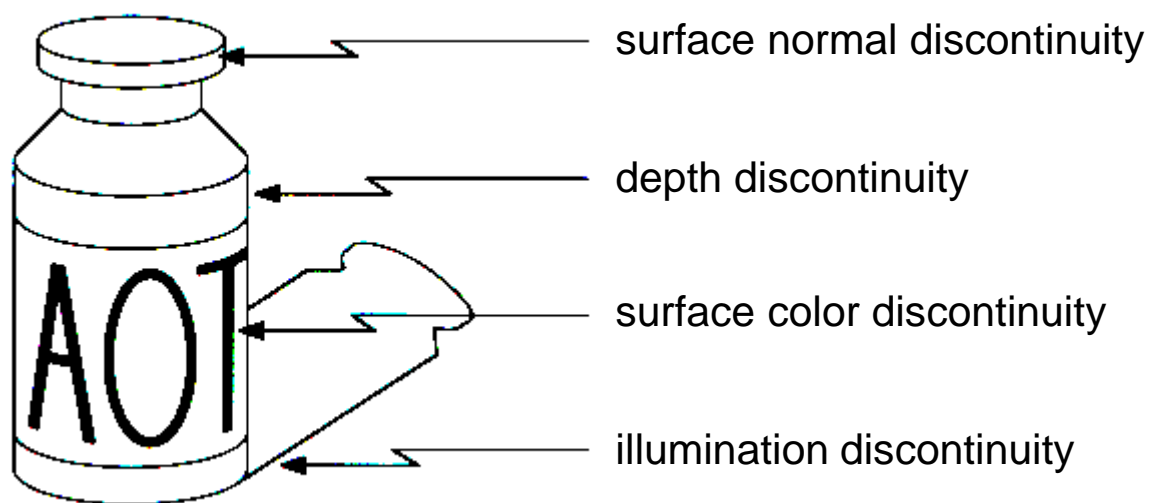




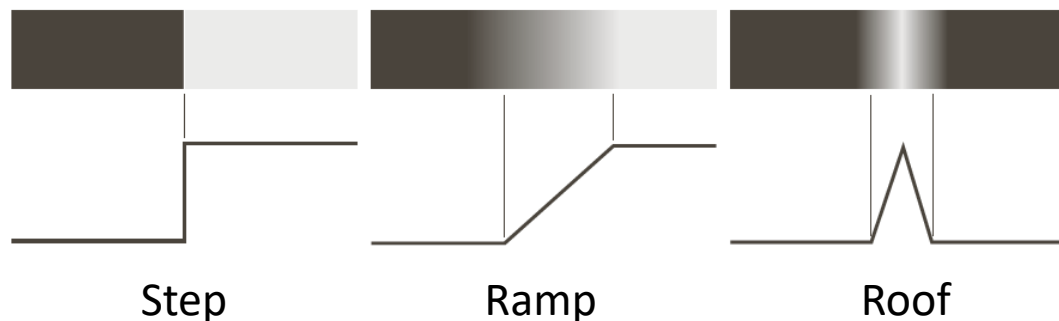
- How to find edges?
- Interactive activity: combine filters to create an edge detection algorithm

Edge detection: problem formulation

- Edges are caused by a variety of factors



- Several types of edges
 - Different *gradients*



The tools



- We have already analyzed several tools that can be useful to solve our problem
 - Gradient
 - Laplacian
 - Smoothing filters



- We have already analyzed several tools that can be useful to solve our problem
 - **Gradient**
 - Laplacian
 - Smoothing filters



- We have already analyzed several tools that can be useful to solve our problem
- Discrete gradient: formulation

$$g_x = f(x + 1, y) - f(x, y)$$

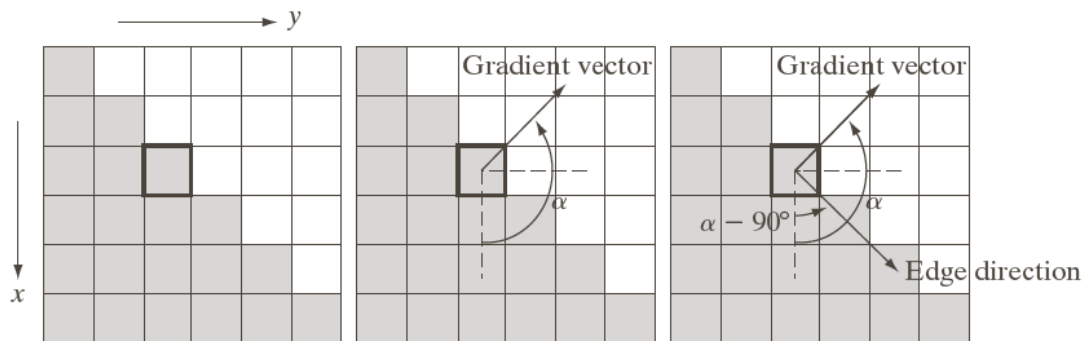
$$g_y = f(x, y + 1) - f(x, y)$$

- The gradient is a vector pointing towards the fastest varying direction
- We get information about edge features considering:
 - Magnitude (edge strength)

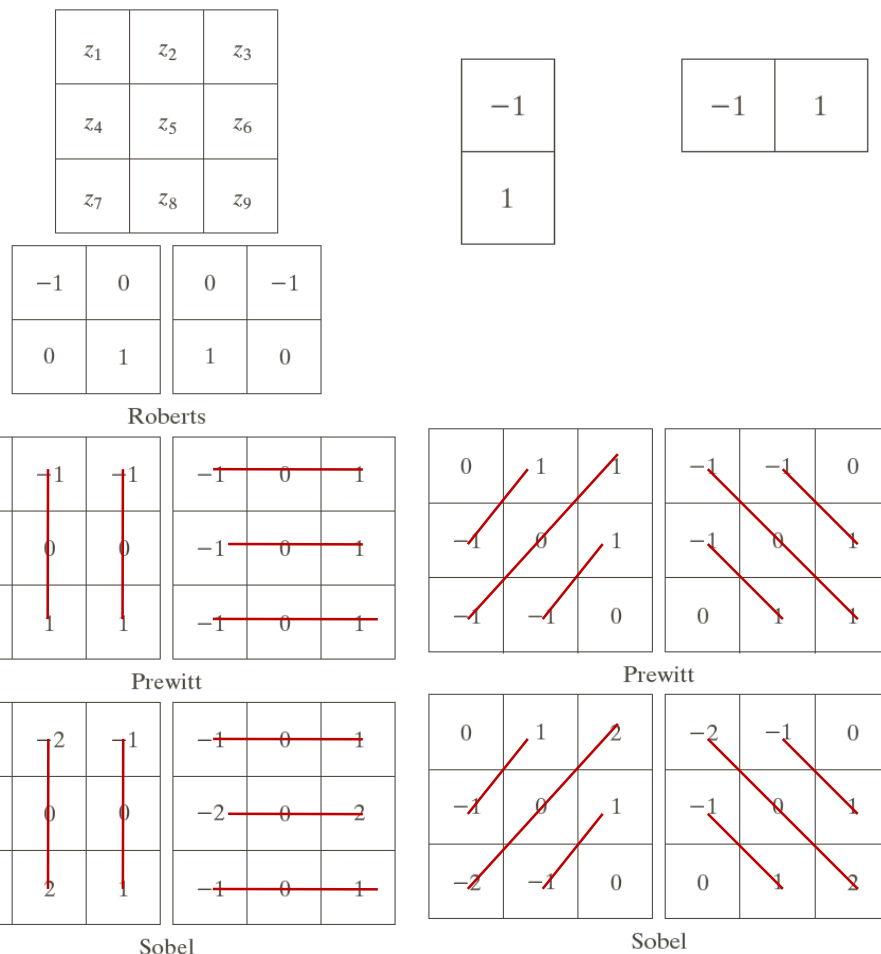
$$\|\nabla f(x, y)\| = \sqrt{g_x^2 + g_y^2}$$

- Phase (fastest varying direction)

$$\alpha \triangleq \theta = \tan^{-1} \left(\frac{g_y}{g_x} \right)$$



- We already introduced the masks for edge detection
 - The Sobel operator is the most widely used
 - Evaluated in four directions



- Gradient magnitude

Is it a good edge detection? No there aren't only the edges in the photo, there's also different colors



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

Sobel

- Gradient angle

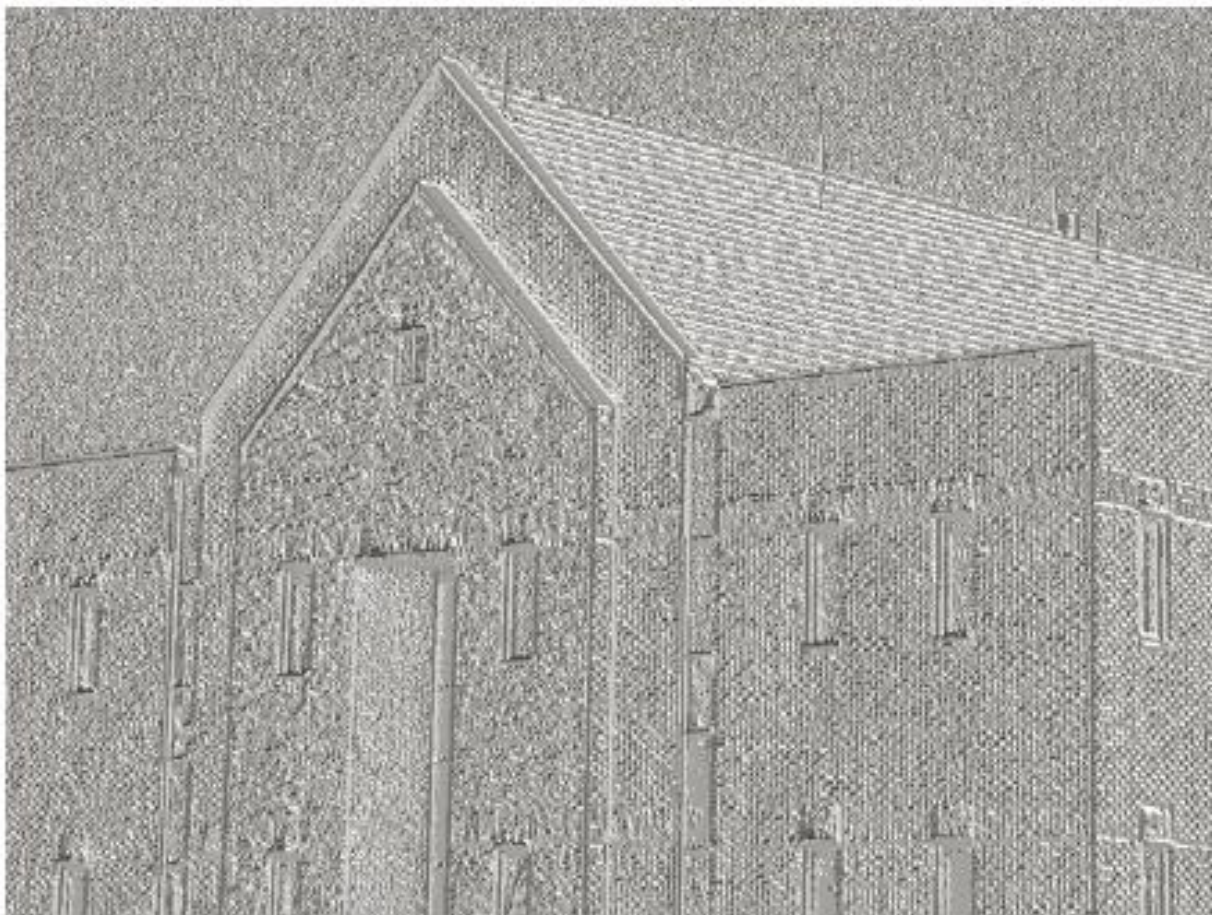


FIGURE 10.17

Gradient angle
image computed
using

Eq. (10.2-11).

Areas of constant
intensity in this
image indicate
that the direction
of the gradient
vector is the same
at all the pixel
locations in those
regions.



- We have already analyzed several tools that can be useful to solve our problem
 - **Gradient**
 - Laplacian
 - Smoothing filters



- We have already analyzed several tools that can be useful to solve our problem
 - Gradient
 - **Laplacian**
 - Smoothing filters

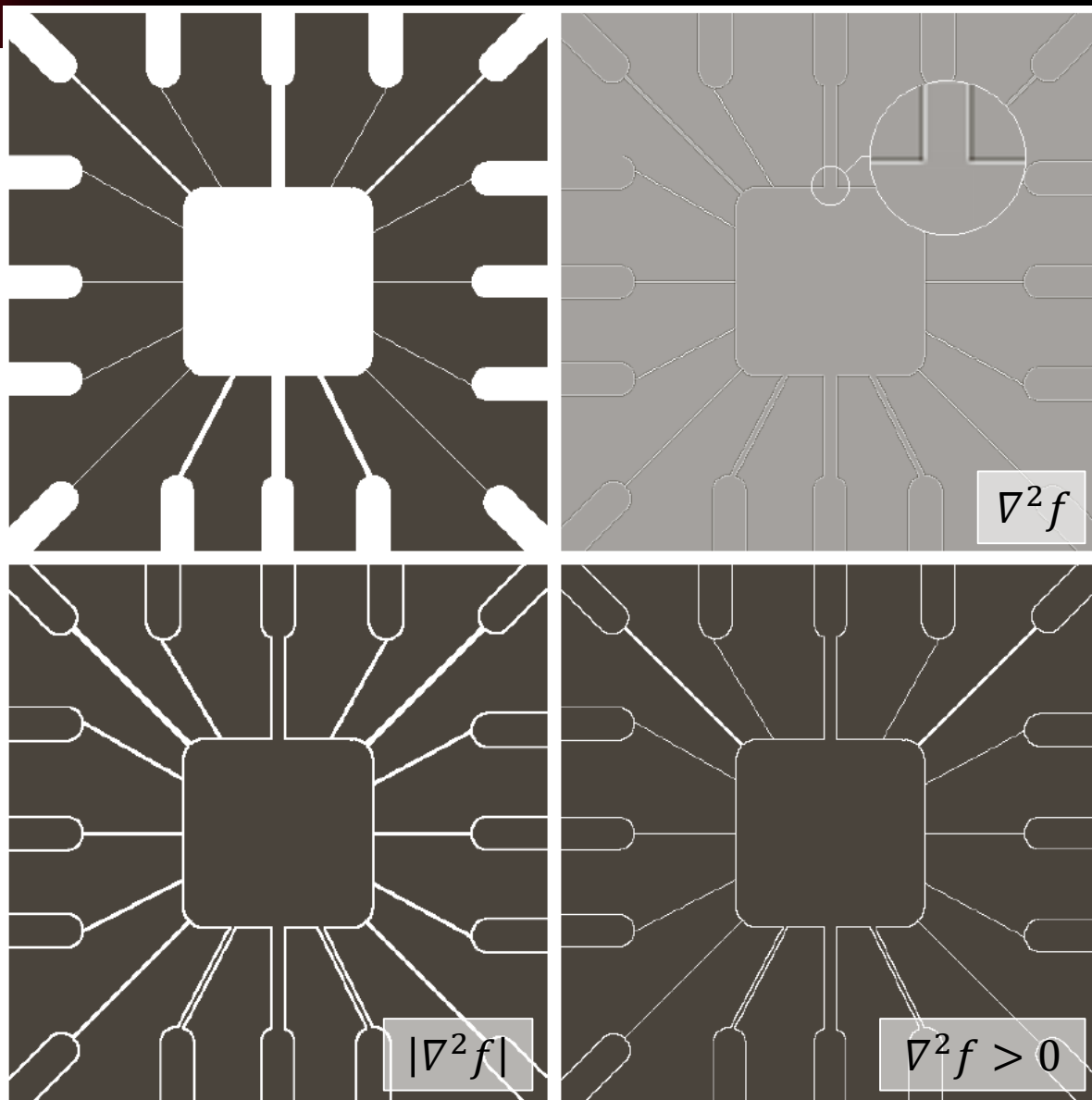


- Recall: Laplacian in discrete domain

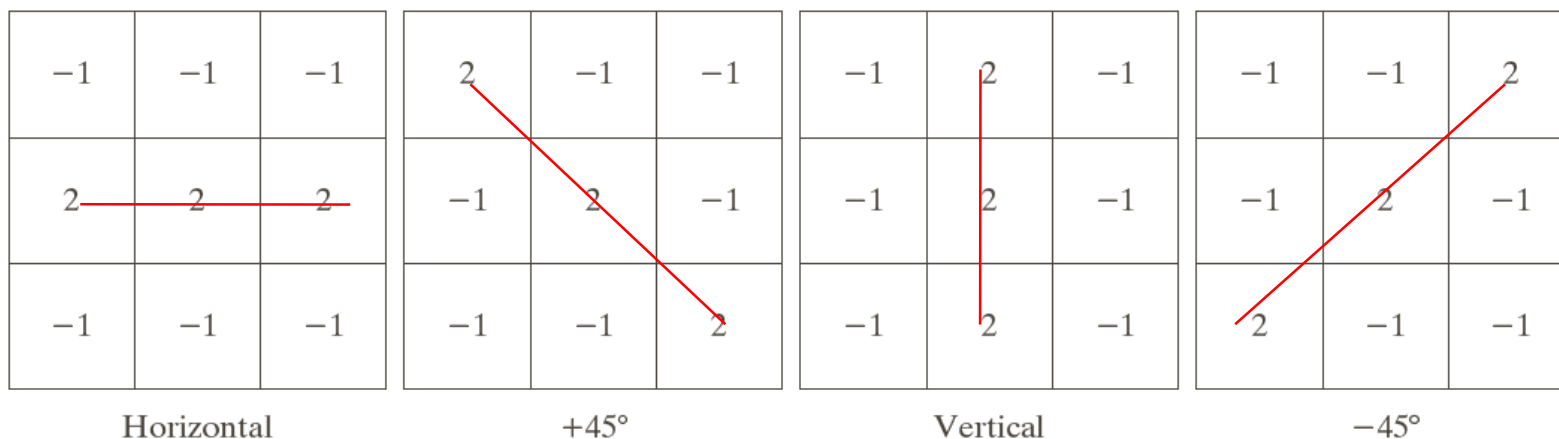
$$\nabla^2 f(x, y) = f(x + 1, y) + f(x - 1, y) + f(x, y + 1) + f(x, y - 1) - 4f(x, y)$$

Laplacian

- Recall:
 - Isotropic detector
 - Double-line effect

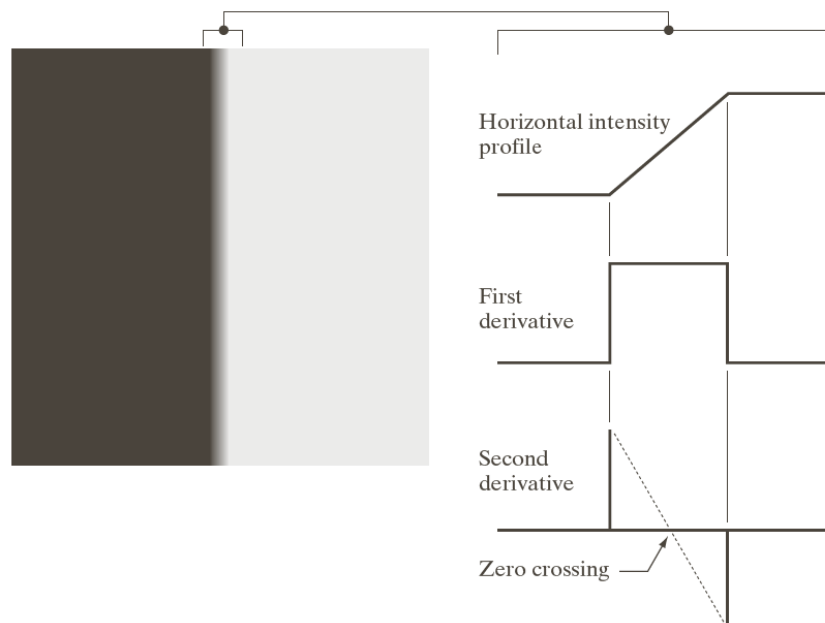


- Recall: anisotropic detectors **derived from** the laplacian
 - Anisotropic: four directions available



- Recap: first and second order derivatives

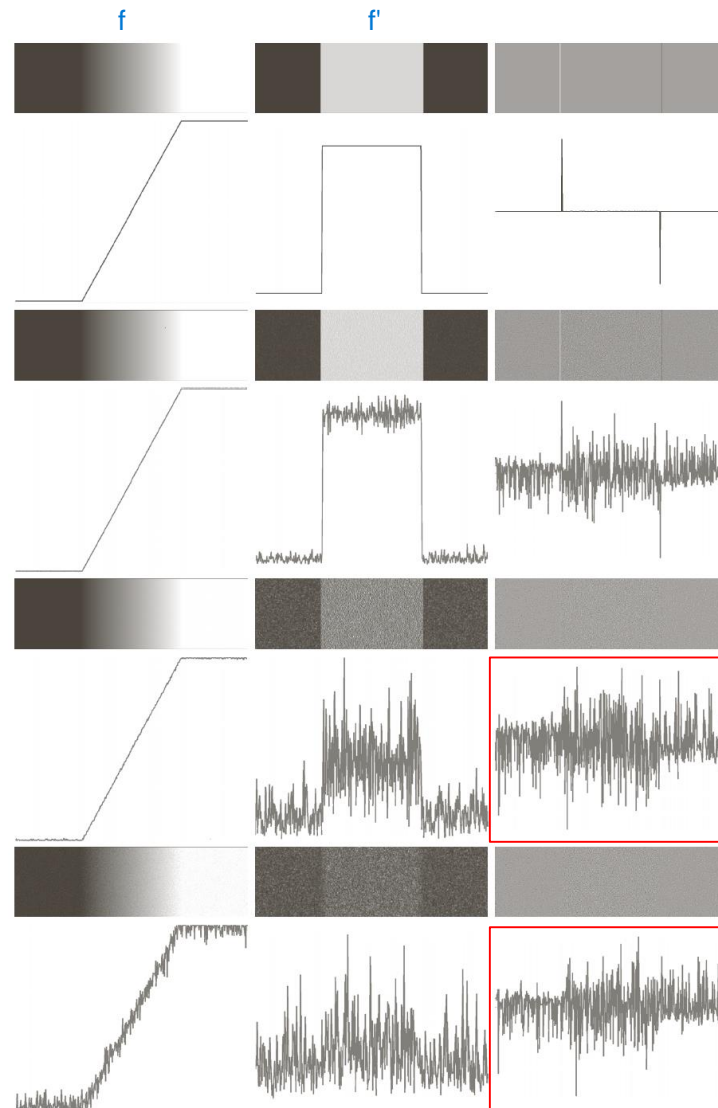
	<i>1st order derivative</i>	<i>2nd order derivative</i>
Large values	On all the edge	Start and end of the edge
Edge type	Single	Double
Edge thickness	Thick	Thin
Noise sensitivity	Moderate	High





- Are derivatives sensitive to noise?

- Are derivatives sensitive to noise?
- Derivatives amplify noise
 - 2nd derivative noisier than 1st



The approach



- Let's discuss a possible algorithm for detecting edges
 - Without looking at the next slides!
- Example: how to reduce the effect of noise on edge detection?
 - This shall be done combining some techniques we have discussed in the previous lectures



- Anti spoiler 😊

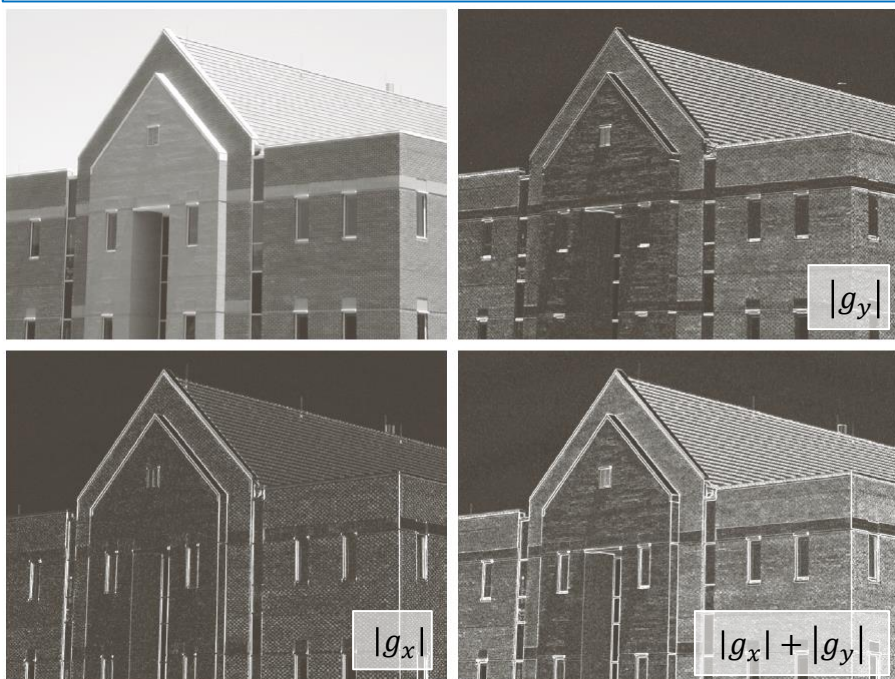


Steps of a possible approach:

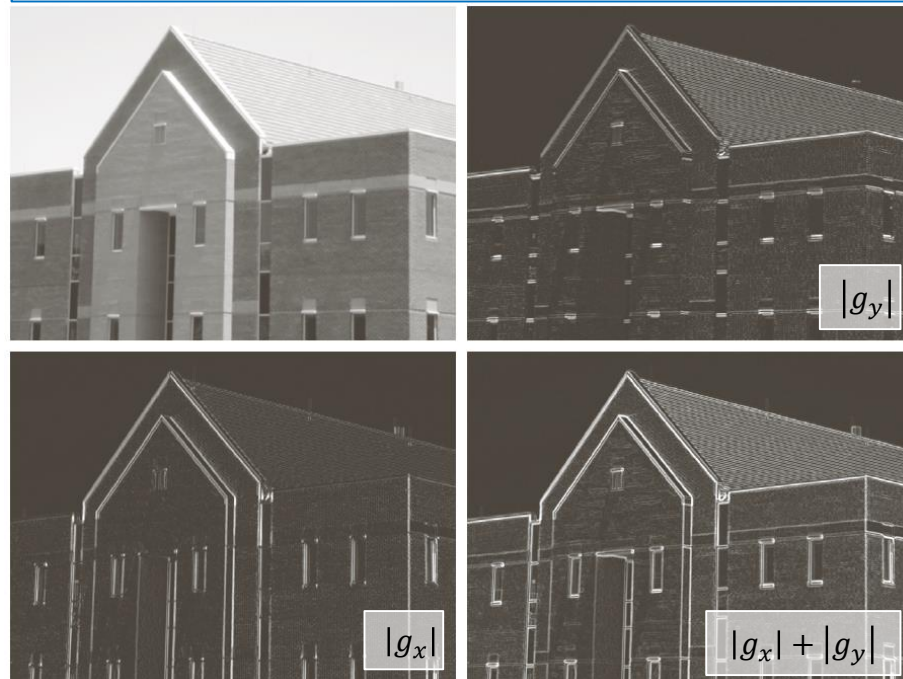
1. Low-pass filter for noise removal
2. Gradient calculation
3. Gradient thresholding – $|\nabla f| > T$

- Edge images are noisy
 - Smoothing (low-pass) often useful prior to calculation
 - In which part of the image can you observe the effect of noise reduction?

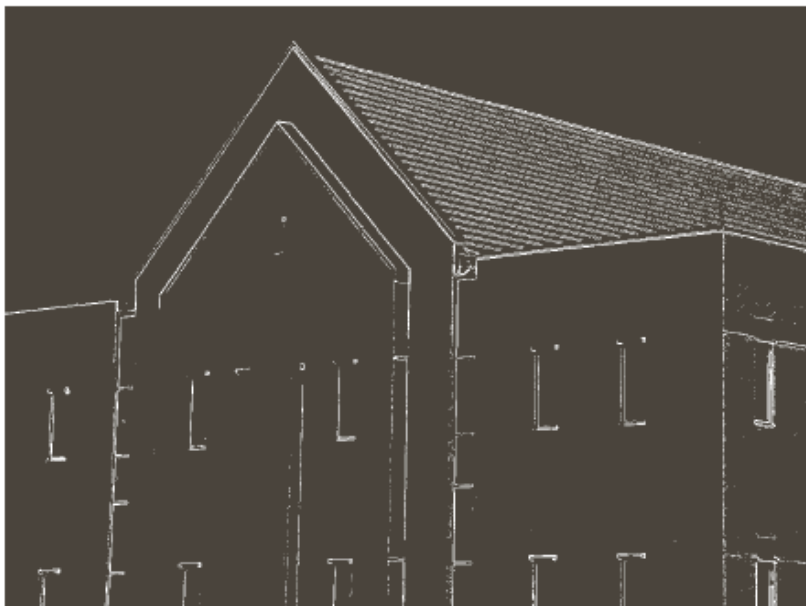
Without smoothing



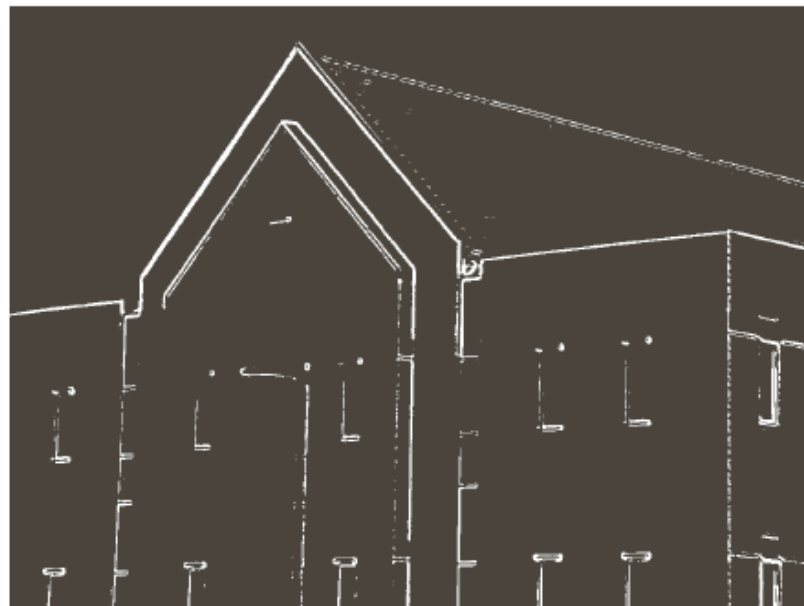
With smoothing: 5x5 averaging filter



- Thresholding selects only the strongest edges



Thresholded (33% of highest value)
no smoothing



Thresholded (33% of highest value)
5×5 smoothing filter

- Results of diagonal edges computation



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

Sobel



- Good! You designed your first computer vision *algorithm*



UNIVERSITÀ DEGLI STUDI DI PADOVA

Your first edge detection algorithm

Stefano Ghidoni

