CS GY 6643 - Computer Vision, Fall 2024 **Guido Gerig**

Homework 2:

Out: Wed Feb-28-2024
Due: Tue-March-5-2024
This is a pencil/paper homework that we use in preparation for the midterm exam scheduled for March 14. The workload will be about 1-2hrs.

Late submissions: Late submissions result in 10% deduction for each day. The assignment will no longer be accepted 3 days after the deadline.

Office hours:

		Mon	Tue	Wed	Thur	Fri
Guido Gerig	gerig@nyu.edu				2-3pm (ZOOM)	
Pragnavi Ravuluri Sai	pr2370@nyu.edu					8am – 10am EST
Sai Rajeev Koppuravuri	rk4305@nyu.edu			12.00-13.00 EST		

Please remember that we also use campuswire for communication on homeworks.

Please submit a **pdf document** where you combine the questions with your answers. You can use pencil&paper for handwritten answers or some other tools for graphing of plots or lists, then scan all pages with an app such as CamScanner or similar, and finally combine into a single pdf document. Please make sure that your answers are well readable for our grading (contrast, hand-writing, etc.).

This is a project to be solved by **each student individually**. Solutions and reports that may indicate copying materials from other students or from web resources are considered plagiarism and subject to violation of the honor code.

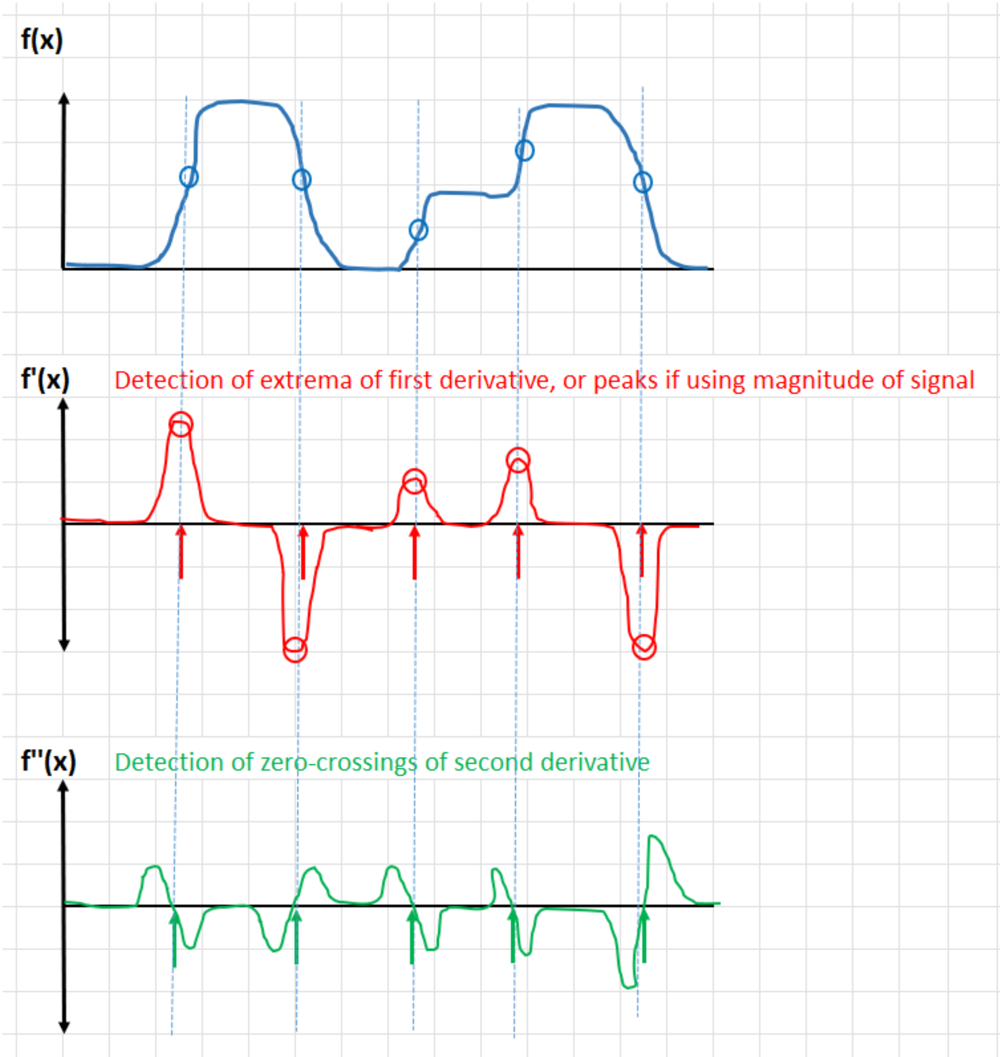
---

1. Edge Detection

1.1 Edge Detection Concept

We have discussed edge detection by applying the first derivative in x and y coordinates to the image f(x,y) and then building the edge magnitude image, followed by processing of the results. In the following, we will study 1D edge detection:

a. Using the sketch of 1-D step-edges in intensity, sketch the first and second derivatives, respectively.
b. Mark the locations of the edges, and briefly explain what scheme you may apply in order to detect the edge locations.



Discussion:

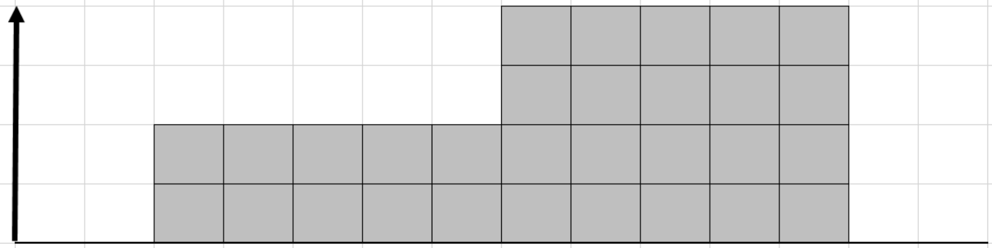
We can detect edges by first derivative of the image signal, and then characterizing the extremal locations (middle pixel larger or smaller than neighbors, e.g.). Alternatively, and simpler, we may apply a second derivative filtering and detect zero-crossings where the sign flips from positive to negative or vice-versa. The slope at the zero-crossings is proportional to the strength/steepness of the edge, and can be additionally used to filter strong edges from noise.

1.2 Edge Detection via first and second derivatives and postprocessing

Given the following 1-D pixel image:

Image Signal

0	0	2	2	2	2	2	4	4	4	4	4	0	0



1.2.a) Apply a first derivative filter to the image signal. The filter is given by the mask shown below. You can center the filter on the right pixel of the mask. Please note that we do not filter the boundary.

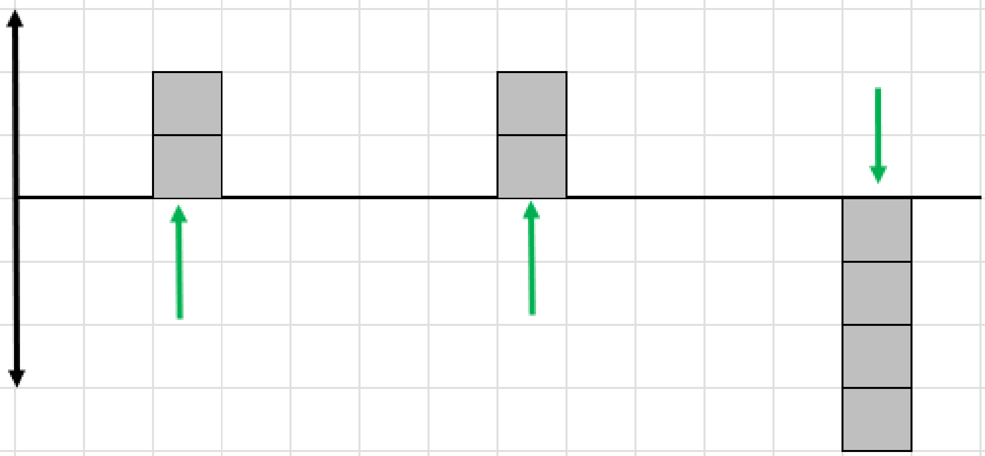
Mark detected edges by proper procedure following 1st derivative filtering

First Derivative

	0	2	0	0	0	0	2	0	0	0	0	-4	0

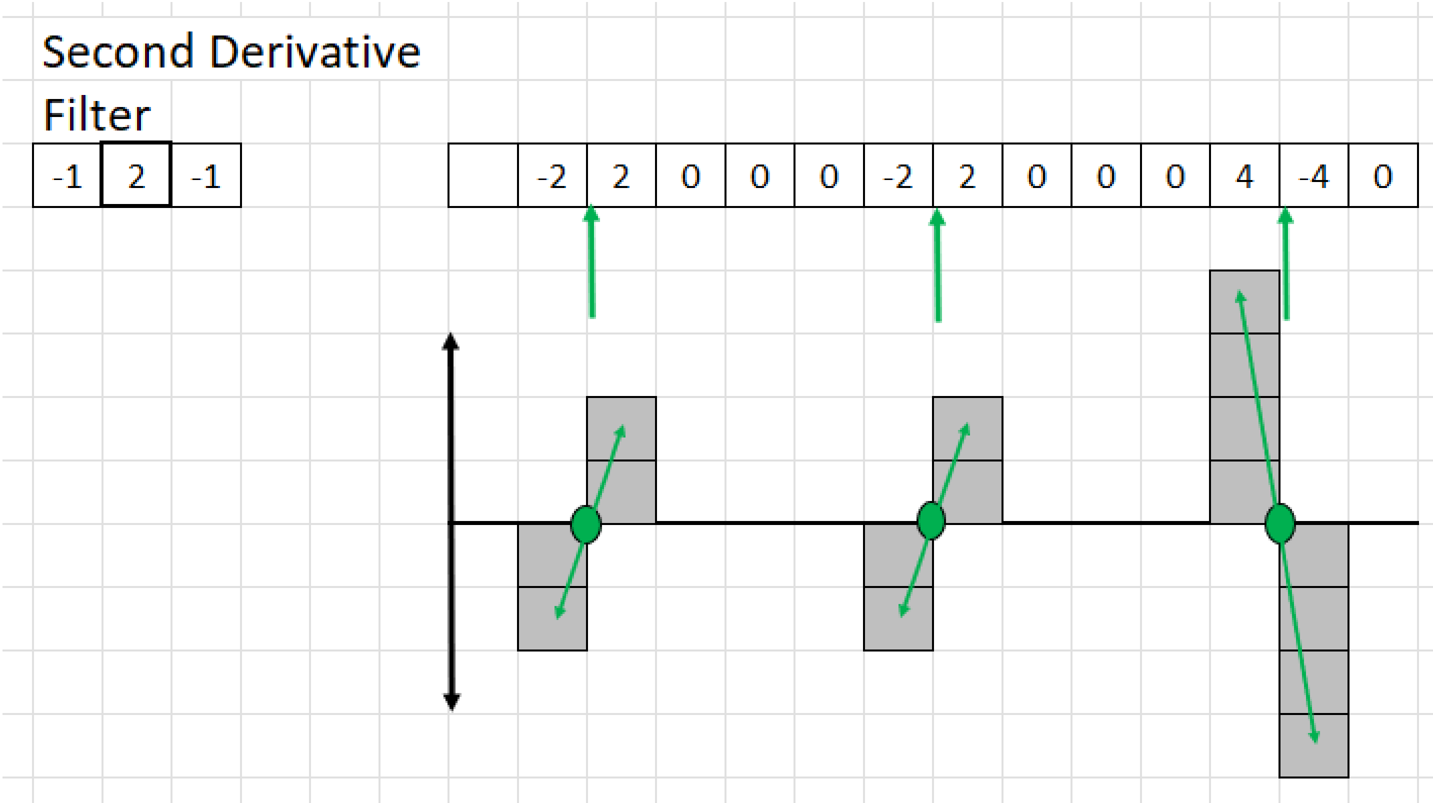
Filter

-1	1

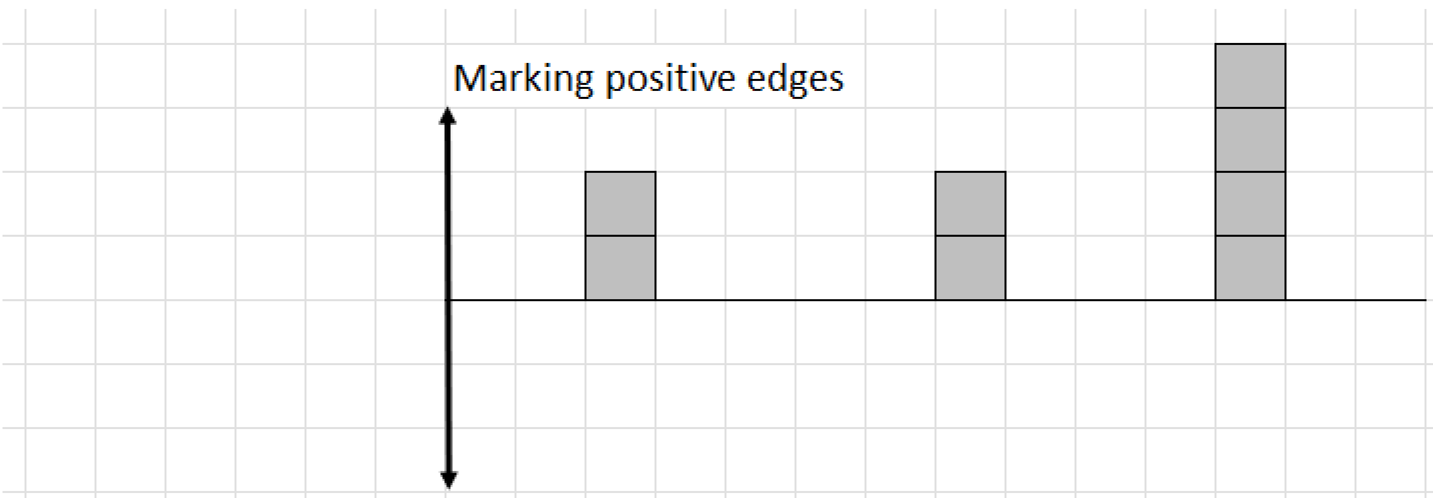


1.2.b) Apply a second derivative filter to the image signal. The filter is given by the mask shown below. You have to center the filter at the middle pixel of the mask. Please note that we do not filter the boundary.

Mark detected edges by proper procedure following 2nd derivative filtering, which is detection of locations of 0-crossings.



Since zero-crossings are "between" pixels, you either have to mark them as the interior side of a structure, or the exterior side. Here, we can mark the positive edges and get the following:



1.2.c) Discuss results that you get in both edge detection schemes. Do you get what you anticipated, or do you see any issues?

Edges can be detected by peaks/extrema of the first derivative, or zero-crossings of the 2nd derivative. The "strength" of the zero-crossings is proportional to the strength of the edge and can be used to filter important versus spurious edges.

2. Correlation Properties

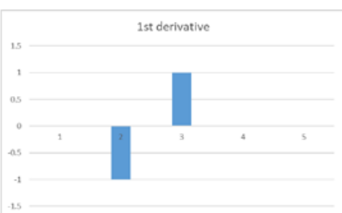
Lecture 3 discusses that the correlation operation is a linear and shift invariant operation (LSI).

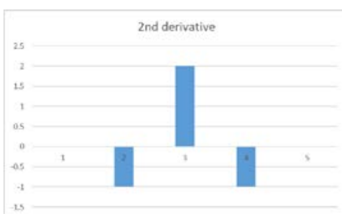
2.1: Linearity: Cascading of filter operations:

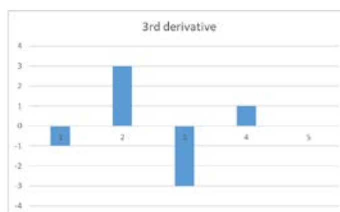
We know that a second derivative of a function f(x) is simply another derivative d/dx of a first derivative of f(x), i.e. f”” =d/dx f’(x) = d/dx d/dx f(x).
In a discrete setting, discussed that the first derivative of a discrete 1D signal can be approximated by a filter o [-1,1], so that your filter operation can be written as *[-1,1] o I(x),* where o stands for the correlation operation.

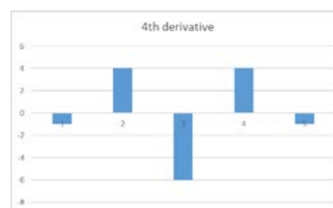
- Using the linearity property, we can write the second derivative as *[-1,1] o [-1,1] o I(x)*, either by applying a first derivative first, and then another first derivative second: *[-1,1] o **([-1,1] o I(x])***, or by first correlating the two filter operations and then applying it to the image: ***([-1,1] o [-1,1])** o I(x)*.
- Derive the discrete 2^{nd} derivative operator via correlating the first derivative operator by itself: (*[-1,1] o [-1,1]):*
-
- *[-1,1] o [-1,1] = [-1,2,-1]*
-

- Using the same concept, derive the discrete operators for 3^{rd} and 4^{th} derivative kernels. Disregard any normalization of filter weights.
- Sketch/plot the 1^{st}, 2^{nd}, 3^{rd} and 4^{th} derivative kernel weights via bar graphs.









	-1	1		

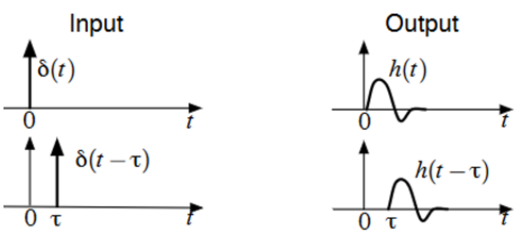
	-1	2	-1	

-1	3	-3	1	

-1	4	-6	4	-1

2.2: Shift Invariance:

Show that correlation of a function with a δ function simply reproduces the original function. Now show that correlation of a function with a shifted δ function shifts the function. δ stands for the delta function, an infinite delta peak in the real world.



In the discrete world, let us use a discrete δ function to demonstrate the concept. Given a discrete function [0,0,1,3,0,0]

0	0	1	3	0	0

2.2.a) Correlate the function with a δ kernel [0,**1**,0].

	0	1	3	0	

2.2.b) Correlate the function with a δ kernel [1,**0**,0], a filter shifted by +1.

	0	0	1	3	

Here, the original function is shifted by +1, which confirms the shift invariance theorem.

3. Linear versus Median Filtering

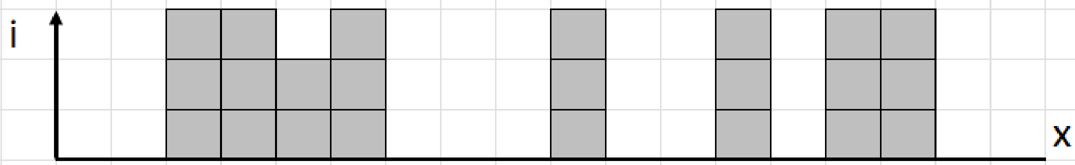
Consider the following 1-D pixel images, shown with numbers and also a plot of x versus i(x).

Filter

1/3	1	1	1

Image Signal

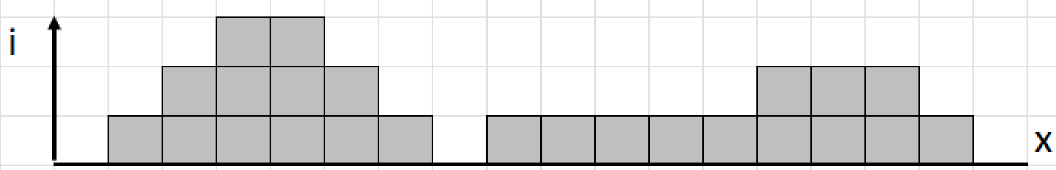
0	0	3	3	2	3	0	0	0	3	0	0	3	0	3	3	0	0



3.1) Apply a linear filter as shown above to the left to the pixel image. Provide the result as pixel numbers and also a plot of x versus i(x).

Linear Filter Result

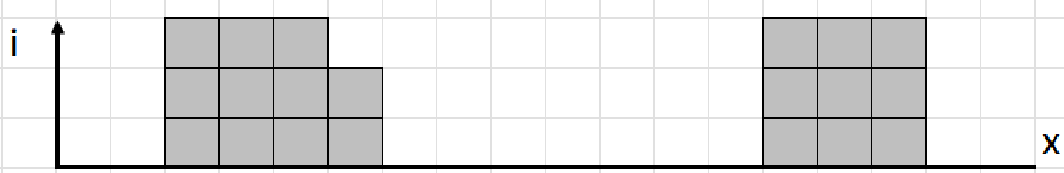
	1	2	3	3	2	1	0	1	1	1	1	1	2	2	2	1	



3.2) Apply a median filter of width 3 to the image signal. Provide the result as pixel numbers and also a plot of x versus i(x).

Median Filter Result

	0	3	3	3	2	0	0	0	0	0	0	0	3	3	3	0	



3.3) Briefly compare and discuss the two results.

The linear filter blurs the signal, edges and lines are not preserved. The median filter keeps the major blocks of the signal intact, and strong edges are preserved. Single lines are removed (“noise”).