Advanced Machine Learning
Exercise 1
Image Filtering and Object Identification

Abbonato Diletta, Pezone Francesco, Testa Lucia

25 March 2020

1 Question 1: Image Filtering

1.1 1.d

We want to study the effect of applying a filter to an image by looking at its impulse response.

First of all we must note that the convolution is both commutative and associative, therefore we have:

$$\begin{cases} f * g = g * f \\ \\ (f * g) * h = f * (g * h) \end{cases} \tag{1}$$

The other important aspect is that a Gaussian filter G is separable, which means that we can write it as $G = uv^T$ with u a row vector and v^T a column vector.
So apply first a 1d gaussian kernel to a picture and after its transpose vertion give the same result of apply a 2d kernel, we can see this from the top left image in Figure 1.

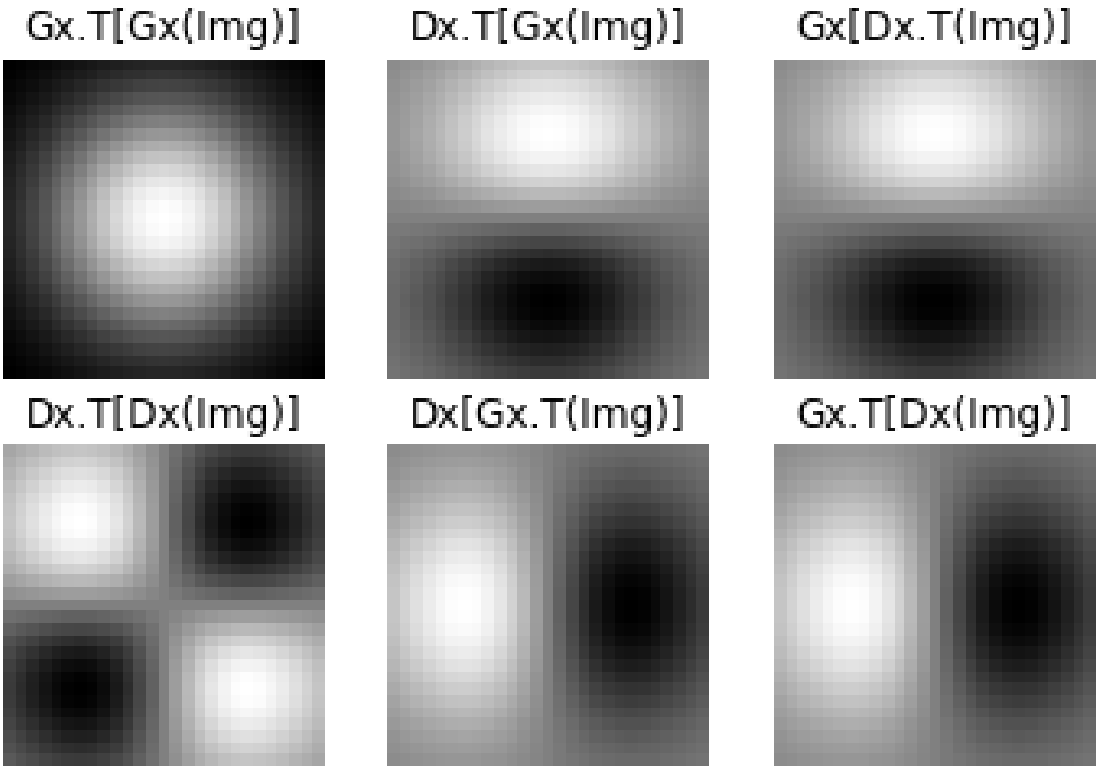


Figure 1: Results of differents combinations of pairs of 1d filters for a impulsive signal

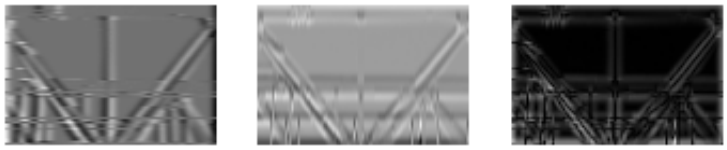
We use the separated version of the 2d gaussian kernel since he require a small number of product: the 2d kernel require k^2n^2 operation, if k is the size of the kernel and n is the size of the image.
The separable version require only $2kn^2$ operations. Now let's see the photo in the upper right and the one in the center. Both filter are equal for the commutativity of the convolution; here we are smoothing along one direction (the x axis) and trying to identify the edges along another (the y axis). The same thing but with the inverted axes it happens in the images below those of before.
With the filter at the bottom left we are unable to detect edges parallel to the axes (x and y), we can only identify edges along mixed directions (f(x,y)) or corners (of course a single pixel is a kind of "corner").

1.2 1.e

Let's start by commenting on the two images on which the directional derivativeskernels have been applied. To better understand, let's focus on the image 'gantrycrane.png', since it has many vertical and horizontal lines. These lines are highlighted in two different ways by the two filters: the contours of the vertical pole in the center of the photo are seen only through the filter with the derivative along the x axis and vice versa.

The important part is to note that the image filtered in this way is strongly conditioned by the presence of noise. To limit this dependence, it is useful to filter the image previously with a Gaussian filter; let's see the difference below.

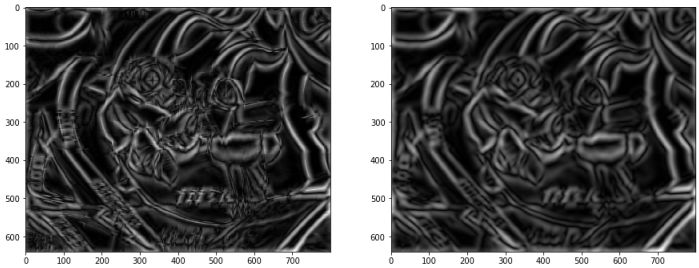
(a) graph.png filtered along different axes in the first two images (x and y respectively) and subsequently with the Pythagorean theorem to match the derivatives of both directions



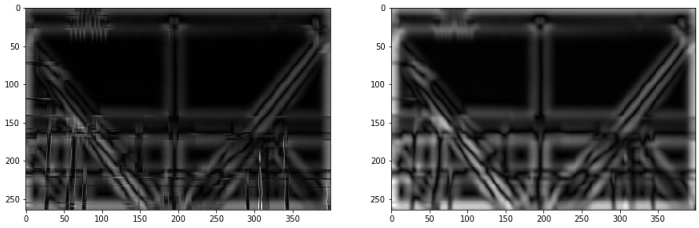
(b) gantrycrane.png filtered along different axes in the first two images (x and y respectively) and subsequently with the Pythagorean theorem to match the derivatives of both directions

Figure 2: Filtered images

As we can see, in Figure 1 the images on the left have a lot of noise (in the form of very thin net lines) but as soon as we apply a Gaussian filter this noise is smoothed and these lines disappear.
In general, we always do this: first we filter to reduce noise and then we filter to detect edges.



(a) graph.pngfiltered directly with the derivative kernel, on the left. On the right we first apply a gaussian filter with $\sigma = 7$.



(b) gantrycrane.png filtered directly with the derivative kernel, on the left. On the right we first apply a gaussian filter with $\sigma = 7$.

Figure 3: Derivate filter with or witout pre gaussian smoothing

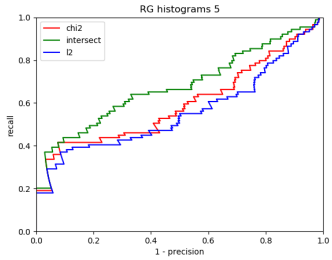
2 Question 3: Object Identification

From the following table we can study in more detail the ability of each type of histogram and distance in comparing and recognizing images. We can see that in the first entries of the table the type of distance is the intersection, while the type of histogram is RGB. Regardless of the number of bins, which in the case of our analysis were 5,10,20,30,40. Below we can find some of the results of the rg histogram, and we can verify that the best performance is obtained by the intersection distance. The distance ^{2} also does not exceed a correctness of 61%, while in $l2$ it does not exceed 57%. The best results with regards to the number of bins seem to be the intermediate values. We find that the best performance for RGB has 30 bins, for RG 40 and for dxdy 20. Finally we can see how the results for dxdy are the worst, since we have a maximum of 60% (dxdy, intersect, 20 bins) at a minimum of 35% (dxdy, l2, 5 bins)

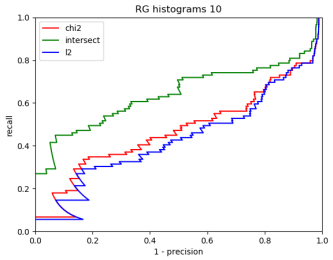
hist type	dist type	num bins	num of correct over 100
rgb	intersect	30	72
rgb	intersect	20	71
rgb	intersect	40	70
rgb	intersect	10	70
rgb	intersect	5	69
rg	intersect	40	67
rg	intersect	30	65
rg	intersect	20	65
rg	intersect	5	63
rg	intersect	10	62
rgb	chi2	5	61
rgb	chi2	10	60
dxdy	intersect	20	60
dxdy	intersect	40	58
rg	chi2	5	58
dxdy	intersect	30	57
rgb	l2	5	57
rg	l2	5	55
rgb	l2	10	54
dxdy	intersect	10	53
rg	chi2	10	53
rg	l2	10	52
rg	chi2	20	51
rgb	chi2	20	48
dxdy	chi2	10	44
dxdy	intersect	5	43
rg	l2	20	43
rgb	l2	20	42
rg	chi2	30	42
dxdy	chi2	20	41
rg	chi2	40	40
dxdy	chi2	30	40
dxdy	l2	10	40
dxdy	l2	30	40
rg	l2	30	39
dxdy	chi2	5	39
dxdy	chi2	40	39
dxdy	l2	20	39
rgb	chi2	30	38
dxdy	l2	40	38
rg	l2	40	37
dxdy	l2	5	35
rgb	l2	30	34
rgb	chi2	40	34
rgb	l2	40	30

3 Question 4: Performance Evaluation

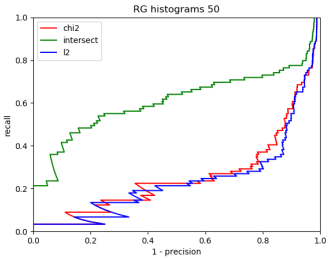
3.1 RG histograms generated with different number of bins



(a) RG histograms $bin = 5$



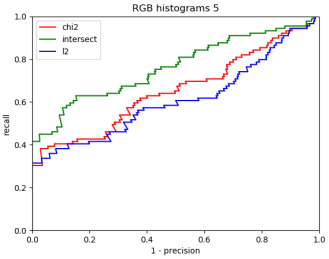
(b) RG histograms $bin = 10$



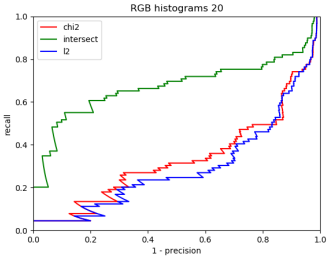
(c) RG histograms $bin = 50$

Figure 4: RPC for RG histograms with a variable bin number

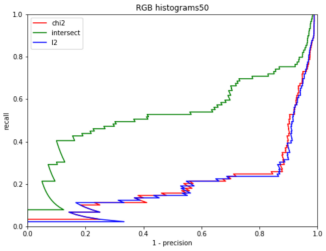
3.2 RGB histograms generated with different number of bins



(a) RGB histograms $bin = 5$



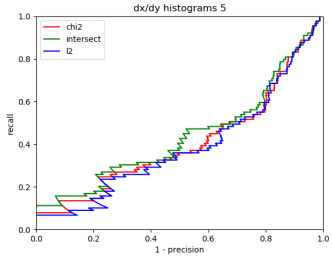
(b) RGB histograms $bin = 20$



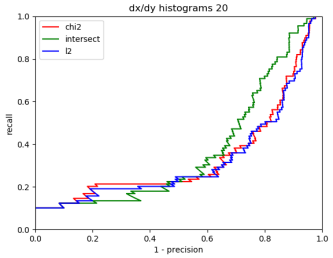
(c) RGB histograms $bin = 50$

Figure 5: RPC for RGB histograms with a variable bin number

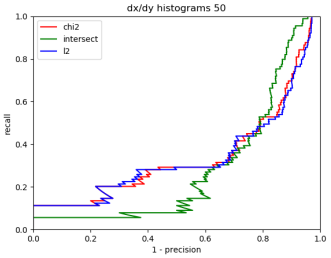
3.3 Dxdy histograms generated with different number of bins



(a) dxdy histograms $bin = 5$



(b) dxdy histograms $bin = 20$



(c) dxdy histograms $bin = 50$

Figure 6: RPC for dxdy histograms with a variable bin number

With reference to the plots obtained by varying the histogram type and at the same time the number of bins per histogram, it is possible to note the consistency of the results according to the output of point 3.c. In fact, there are better performances with the use of the RGB histogram. Furthermore, by paying attention to the plot, it's possible to notice that the intersect distance is the best as regards the use of the RGB itself. In general, we can also say that the distances l2 and chi2 are less performing than the intersect mode, and this could be well understood from the results obtained previously in which the scores for the distance l2 and the distance chi2 are lower.

Finally, the most significant result is the one concerning edge detection, using the *dxdy_histogram*. There was already a very low rate in the results of 3.c, but the precision recall plots confirm these results in which there is a trend that starts from the bottom and continues up very slowly. In addition, we can notice that the three distances do not particularly prevail over each other. This is an indication of the poor effectiveness of the type of histogram, regardless of the *dist_mode* used.