SAPIENZA

Assignment 1

Student name: *Salvatori, Chirita, Trasciatti, Yahia-Messaoud*

Course: *Fundamentals of Data Science* – Professor: *Fabio Galasso*
Due date: *November 6th*

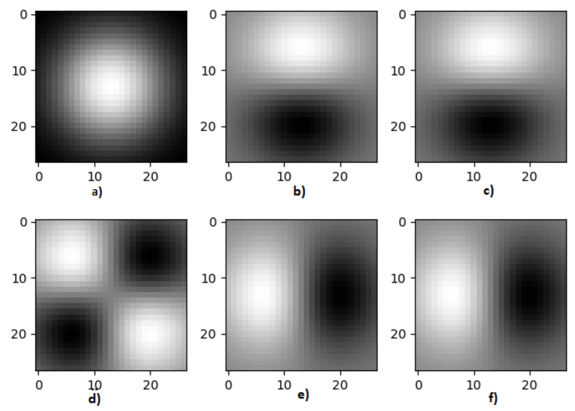
Report by Giorgia Salvatori [1763710], Gabriel Chirita [1798135], Gabriella Trasciatti [1799281] and Selim Yahia-Messaoud [1965129]

Question 1.d

Linear filtering is one of the basic operation in digital images processing, it consists in replacing each pixels by a linear combination of its neighboors.
Gaussian filter is an example of average filter in which we have a mask with positive entries that sum to one. It is a separable so :

$$(f_x \otimes f_y) \otimes I = f_x \otimes (f_y \otimes I), \tag{1}$$

where I is the image, f_x is a 1D row filter and f_y is a 1D column filter, this is good in term of computationality.
Next we have the output of question 1.d:



The original image was a white pixel in centre of a black background. In this case the image is double smoothed and we can see how the white portion increased in the center of the image and gradually, to the edges, pixels become darker.
In **b)** we first smooth, so the number of lighter pixels increases in the center of the image, and then we apply derivative along the vertical axis so horizontal edges are emphasized.
In **d)** we apply twice the derivative, along both direction. As a result we have emphasized both vertical and horizontal edges in the smoothed image.
In **g)** again, we smooth and then we apply derivative along the horizontal axis so vertical edges are emphasized.

If we look at the couples **b) c)** and **e) f)** we can observe that each couple has the same output. We expected that, since this is due to the property of associativity for the gaussian filter.

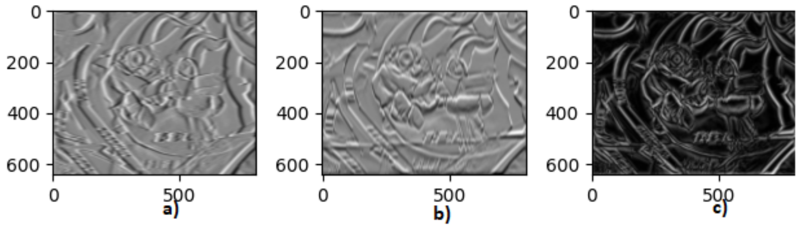
Question 1.e

In *gaussderiv* method we first smooth according to the standard deviation σ and then we apply the derivative filter in x and y direction.

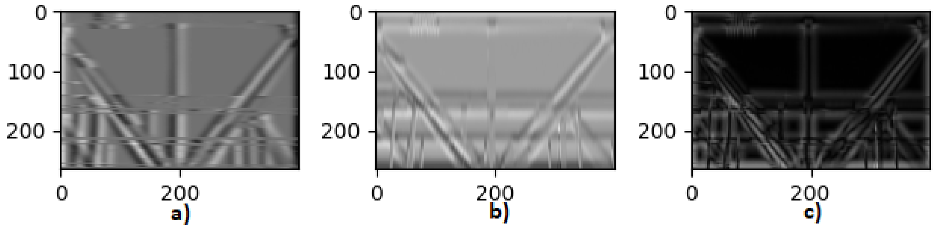
$$\frac{d}{dx}(G \otimes I) =^{1} \left(\frac{d}{dx}G\right) \otimes I \tag{2}$$

$$\frac{d}{dy}(G \otimes I) = \left(\frac{d}{dy}G\right) \otimes I \tag{3}$$

The first step is important in order to reduce the noise, in fact noise is high frequency and the first derivate amplifies frequencies. In this way becomes difficult differentiate edges, which corresponds to fast changes so high frequencies, besides noise.



In the **a)** figure we applied the derivative filter in x direction through the function *gaussdx*. This is a good method to detect vertical lines, in fact we can see more clearly vertical edges of the graf.png image.
The same reasoning was applied in **b)** but in y direction, so we have horizontal edges more defined.



In the gantrycane.png image there are few type of edges: vertical, horizontal and lines inclined by 45° degrees to the right or to the left.
In **a)** we can see that the central horizontal line is missing because we are using derivate filter along the x direction.
Likewise in **b)** we are not able to see the central vertical line.
Even not using the gradient, the oblique edges are detected because we have a strong

[1] Associative property of convolution

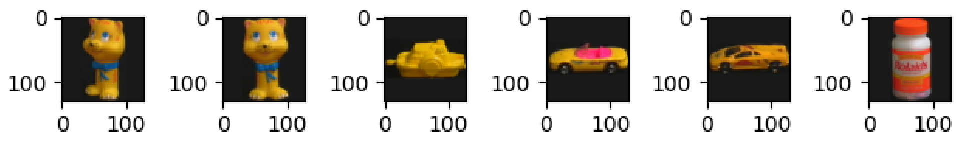
transition between the blue sky and the grey steel.
We use derivatives in image to detect edges because they became larger across them:

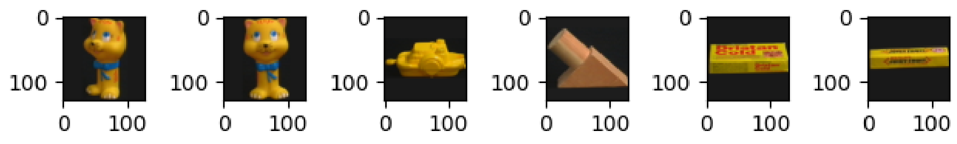
$$\frac{d}{dx}f(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h} \approx f(x+1) - f(x) \tag{4}$$

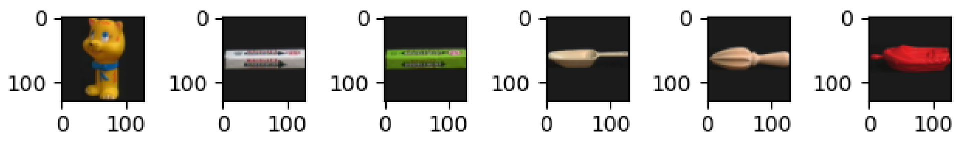
the last difference reprensent the difference between the intensity of a pixel and the intensity of the pixel next to it. The value of $f(x+1) - f(x)$ increase when there is a fast change of intensity, this is what characterize an edge. In conclusion, the local maximum and minimum of the derivate can help us dectect borders.

Question 3.c

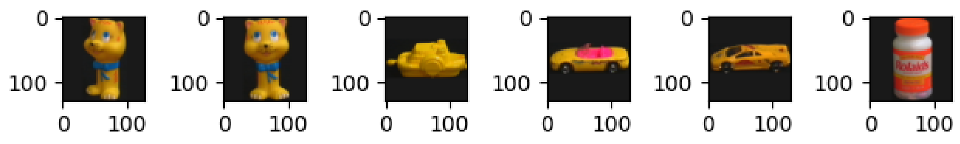
Let's start maintaining fixed *dist_type=intersect* and *hist_ type= rg* . If we decrease the value of *num_bins* the algorithm is less accurate:

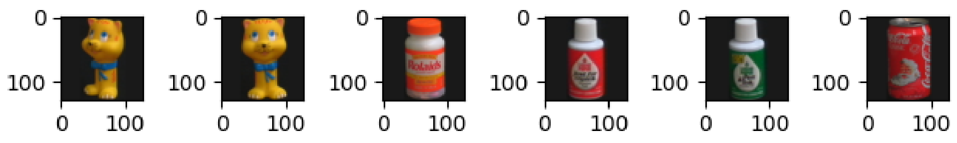

num_bins=10

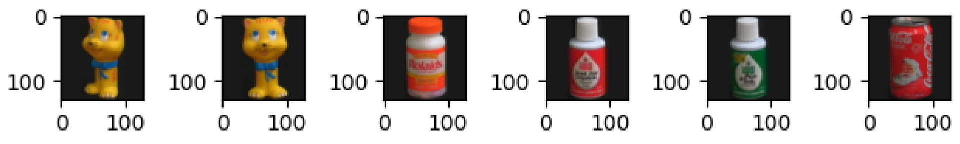

num_bins=5


num_bins=2

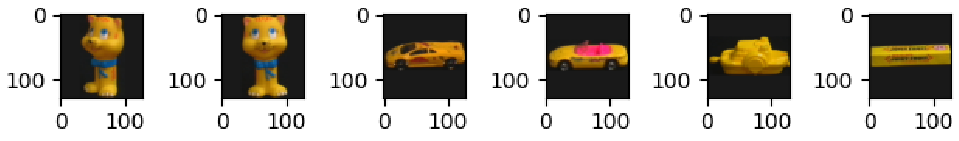
As the value of bins decreases the number of correct matches decreases: 65 out of 89 for *num_bin=30*, 62 out of 89 for *num_bin=10*, 29 out of 89 for *num_bin=2*.
This is due to the fact that if we have less bins the range of them increases and we group together more intensities of pixels, for example in the last graphic we have 256 different intensities of pixel divided into only two bins.
After a lot of trials we found out that num_bins=67 gives us the highest number of correct matches (74), over this number we have less correct matches. Now set *hist_ type= rg* and *num_bins=10* . In our case the intersection function gives us the best output:


dist_type=intersect


dist_type=chi2


dist_type=l2

Even if we have a small number of bins intersection provides us with a satisfactory result. Euclidian distance emphasizes the distance using the square and, by using it, all cells are weighted equally. We can try to solve the second issue by using the χ^2 distance and we can see, by the recognition rate, that the χ^2 distance provide us a slightly better result because the cells aren't weighted equally.
So, as we expected in the first case we have a recognition rate equals to 0.696629 in the first case, to 0.595506 in the second case and equal to 0.584270 in the last case.
Choosing *dist_type=intersect* and *num_bins=10*, we see a different result using *hist_type=rgb*:



Now the fifth best match is a yellow object of a color more similar to the cat and the ratio between number of correct matches and total number of query images is 0.786517.

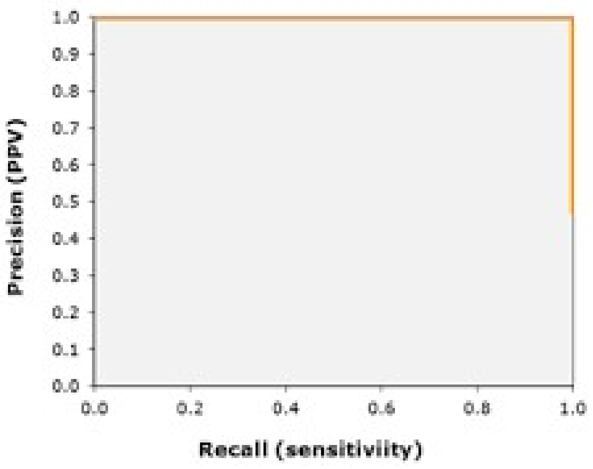
Question 4.b

Precision and recall curve (PRC) provides us a way to evaluate how good our identification algorithm is.
By definition we have:

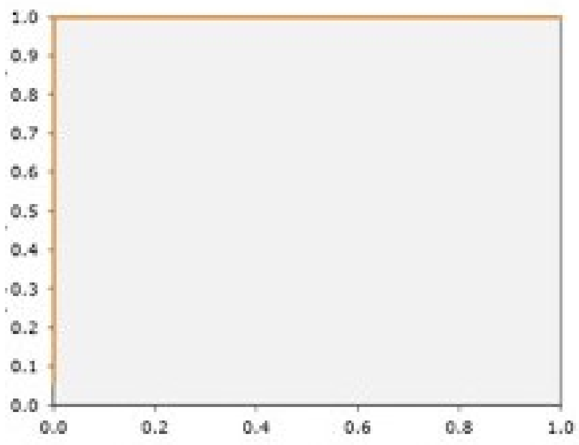
$$Precision = \frac{TP}{TP + FP} \tag{5}$$

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

In an ideal system we would like to have both precision and recall equal to 1, so as an ideal curve will be something like



In our case the labels are recall and 1−precision so in our ideal plot is



If we fix the number of bins at 20, in figure 1 we can see that, with each type of histograms, the intersection distance performs better than the euclidian distance and the χ^2 distance because the curve associated to it has a shape more similar to the ideal plot. We already knew that since intersection distance provides us the highest number of correct matches, we should observe the best performances with this method.

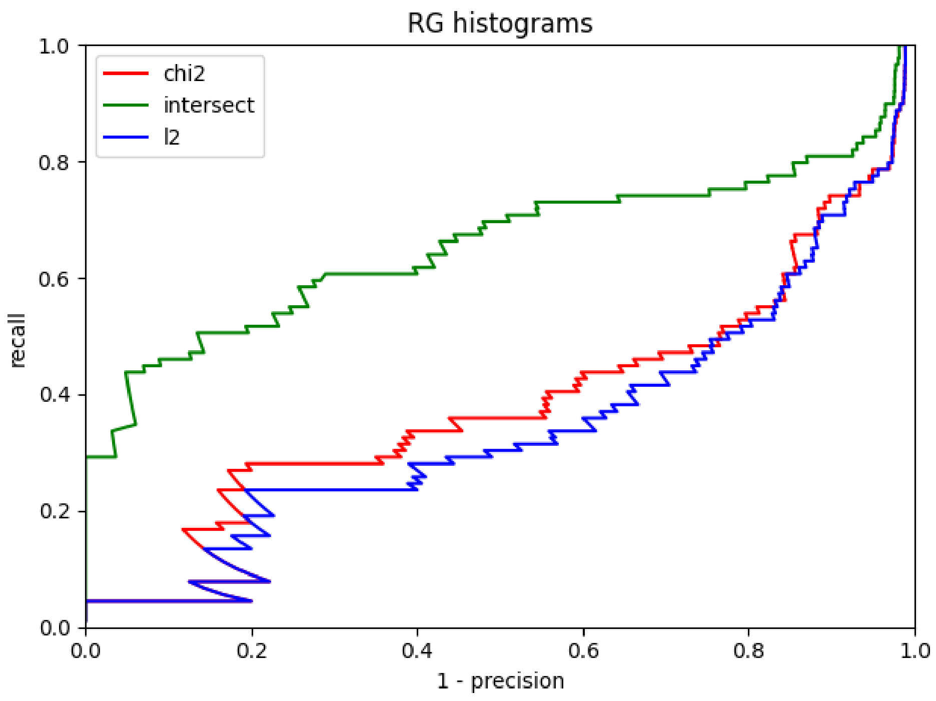
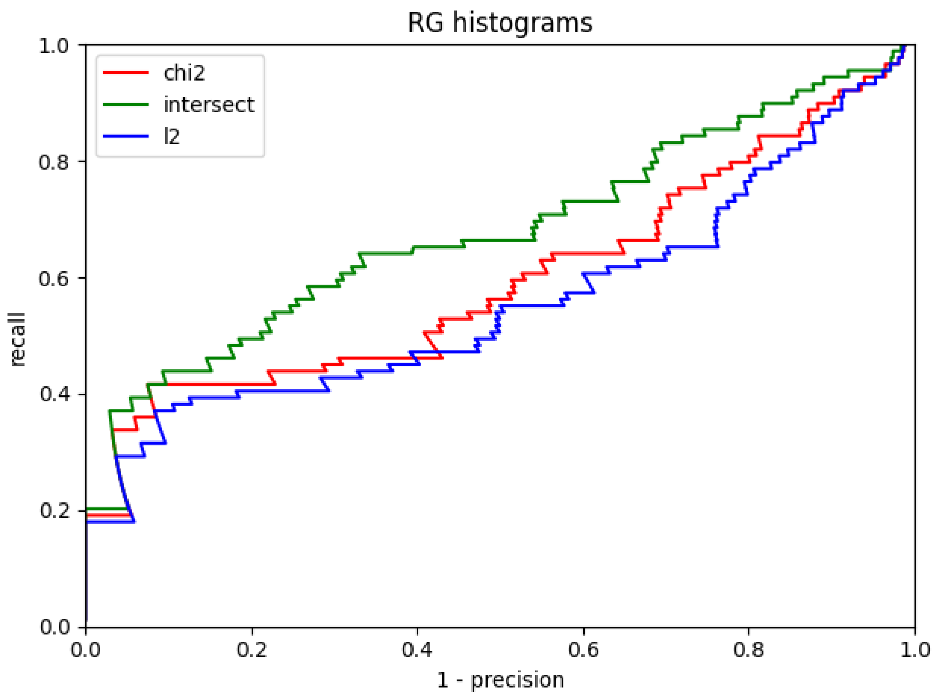
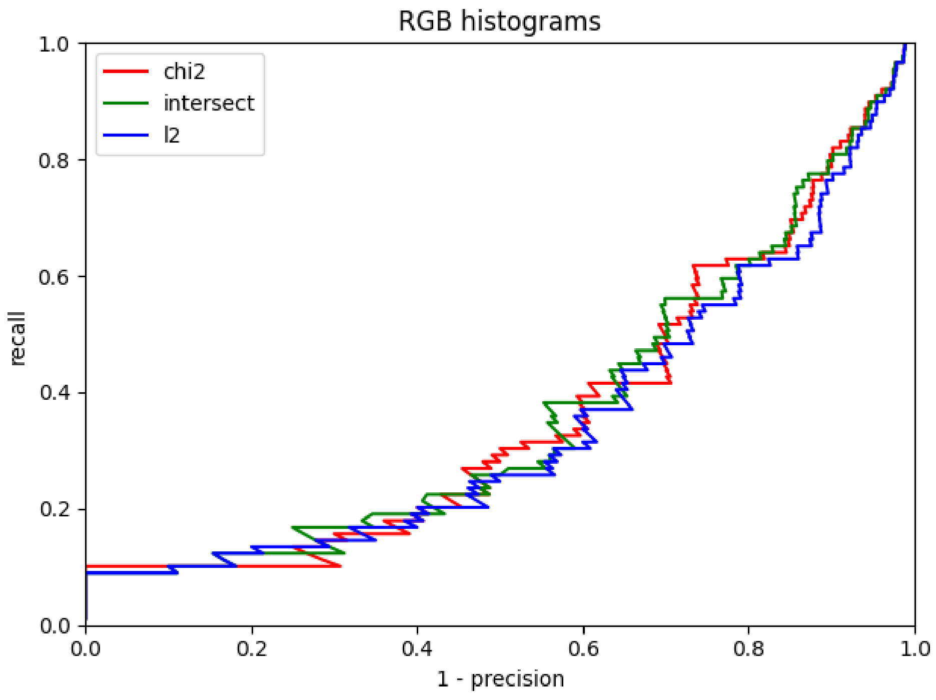


Figure 1: the green curve is above the other two, so has higher value of recall

Using 5 bins the difference among the three curves is less noticeable than before but intersection method still give us the better output.



Things change when it comes to rgb histogram since there is not a curve above the others, they intersect among each other continuously. This means that, with such a small number of bins, there isn't a better measure to use.



As we increase the number of bins the difference among the curve grows.

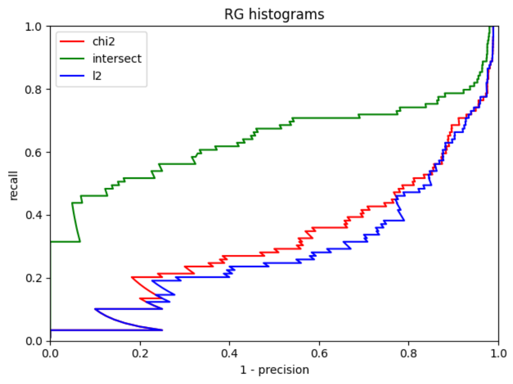


Figure 2: Plot of rg histogram with num_bins=30

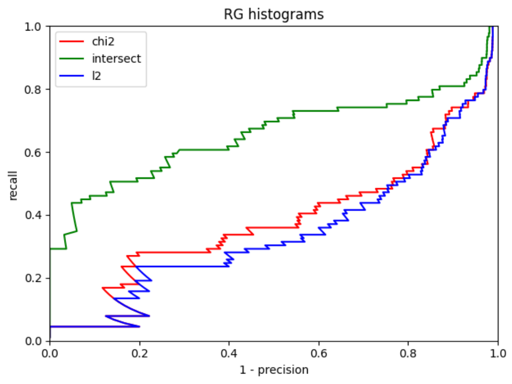


Figure 3: Plot of rg histogram with num_bins=20

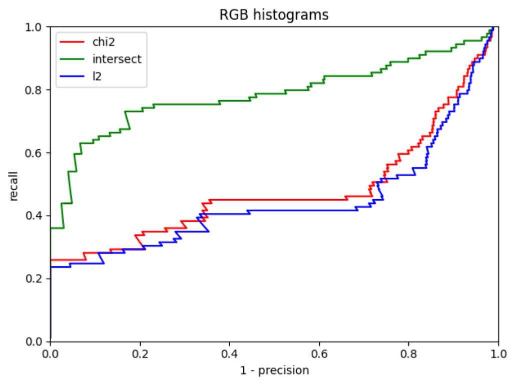


Figure 4: Plot of rgb histogram with num_bins=30

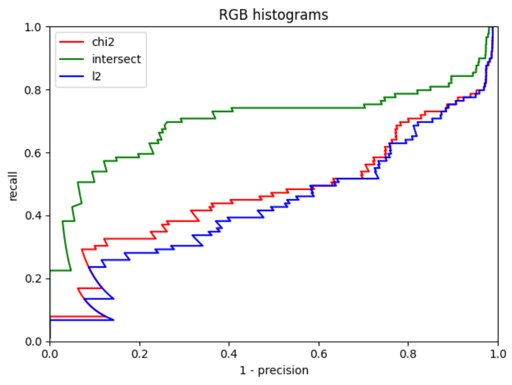


Figure 5: Plot of rgb histogram with num_bins=20

Placing our attention to the dx/dy histogram we can see that increasing the number of bins the performance of all of the distances is worse.

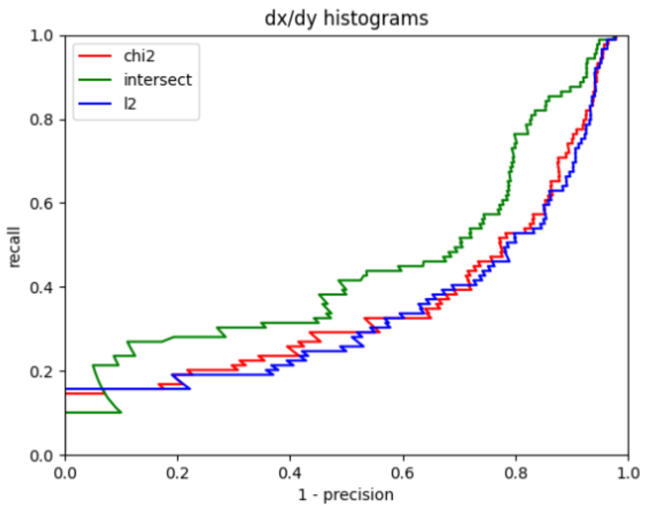


Figure 6: output with num_bins=10

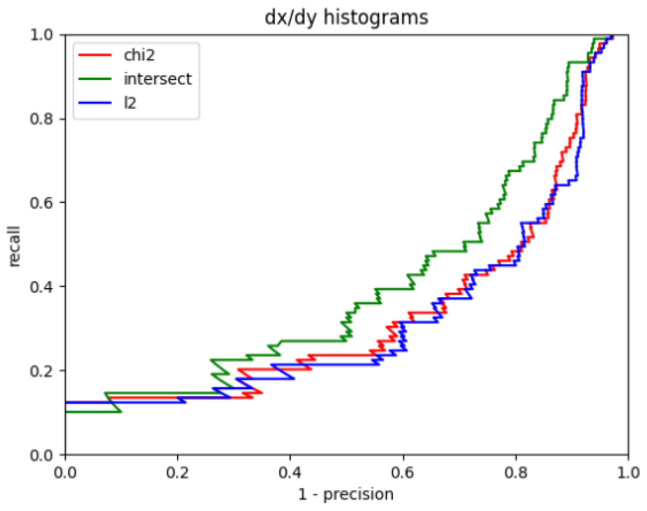


Figure 7: output with num_bins=20

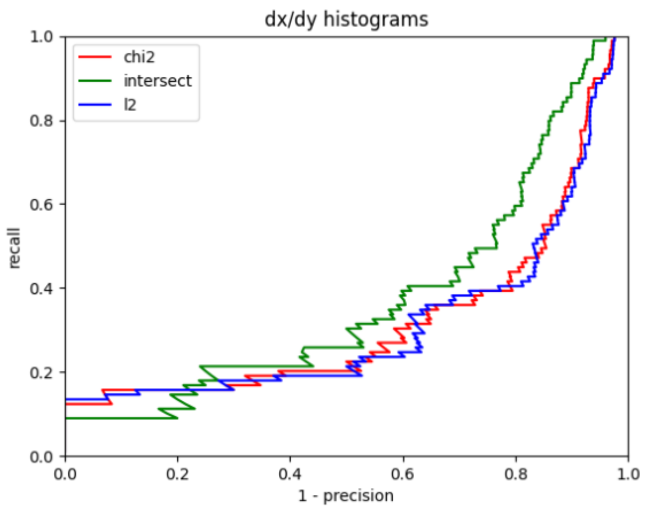


Figure 8: output with num_bins=30

As a conclusion, we obtained the best performances with the rgb histogram, the intersection distance and with a number of bins of equal to 67. To choose an optimal threshold, we look on our graph at the nearest point to the top left of the plot. Hence it offers both good precision and recall.