# Assignment 2: Filtering implementation

For this assignment we were too design and implement a python program to run a photo of ours through a selection of the filtering methods we have learned so far. For this task in the program, I utilized a python dictionary to store and assign the various kernels for filtering. The median, and gradient edge did require their own implementation as the extraction of the filtered image isn’t as straight forward as the other filters. For this I also utilized a mix of OpenCV and scipy.signal each had either easier extraction or more desirable output( the scipy for gradient edge); so a mix felt appropriate. As for the report I would like to structure this in the method of going through analyzing each output as I called it in the program. -> Average, Sobel’s, Gradient, Median, Laplacian, Gaussian, and then the two-user defined for me was another Gaussian Kernel, and finally a large special filter Kernel. At the conclusion I will talk about the work I did with filtering and experimenting with the pupil image. For reference I will start with my sample photo, (Provided with consent of my roommate) this is Kuma!

A close up of a dog

Description automatically generated. A picture containing text, dog, indoor, black

Description automatically generated

Next to it we see the first filter I applied(after converting to grayscale) the Average filter. The softening of all the general features was the thing most notable to me. I really enjoy the zoom feature as the image is a bit large zooming helps to really encapsulate the subtle change the filter had.

For the next grouping we will be looking at the Sobel filters and the gradient edge. These in particular I found really interesting. Kuma is a great subject with the contrast in her coat that has come to distinguish her. The Sobel filters did a great job of picking this up; and when we see the final gradient edge you really get a feel for how interesting and precise a tool like this can be!

The gradient edge here is really the cool transformation. With details down to light from outside catching in the eye and the dispersal of black fur around her nose. This was a particular favorite of Kuma’s owner as well!

A picture containing text

Description automatically generated A picture containing graphical user interface

Description automatically generated

A picture containing diagram

Description automatically generated

Our next filter will look at the median filter. This one required special implementation as it uses a non-linear method of approximation. Upon close inspection I believe this filter did more in the way of softening than the average filter. I will note though that the size of the photo can make those subtleties more difficult to spot. That and choosing Scipy as my output, the OpenCV output was easier to detect the filtering, but the window couldn’t contain the entire sample. Thus, I utilized Scipy as my sole output generator.

A picture containing text, black, indoor, looking

Description automatically generated

Next we take a look at the Laplacian filter for detecting edges, I found that the return image had notable weaker contrast on detected edges than that of the Sobel filters for this sample. However when It came time to work with the Iris my Sobel filters would miss a section of the pupil boundary, one the Laplacian had an easier time detecting. Each to its own uses I suppose.

A picture containing graphical user interface

Description automatically generated

Finally we get to what I believe was the most visually pleasing grouping of filters. We have the two Gaussian filters and a large kernel special filter. Again notes on sizing and detection of filtration. But overall I found both of the Gaussian filters to give me the most aesthetically accurate results. The photo felt softened but in a gentle more reactive way. Where the results of the Spatial filter are just an overall general smoothing of everything.A picture containing text, dog, looking, indoor

Description automatically generated A picture containing text, dog, looking, black

Description automatically generated

A picture containing text, indoor, black, looking

Description automatically generated

For the final section of this report we will look at the filtering efforts made on the Iris image. I messed around with this section for a good while with the goal of really highlighting the pupil boundary. I however didn’t obtain a center point of the pupil(not to say I wont attempt to in my own time), just enjoyed the filter manipulation more. One of the tricks I really enjoyed while manipulating the image was to add or subtract the filtered results from the original image. This in my opinion often yielded interesting and dynamic images. Here is a sample of the code that brought about the images. Followed by my favorite final resulting image.

detect = filter((filter((filter(filter(pic2, 'Gaussian'), 'Laplacian') \* pic2), 'Average') - pic2), 'Laplacian')  
detect = filter(filter(filter(detect, 'Gaussian'), 'Gaussian'), 'Spacial-large')

A picture containing graphical user interface

Description automatically generatedA picture containing graphical user interface

Description automatically generated

A picture containing chart

Description automatically generated

Overall this was a fun project I feel I have a confident understanding of these filters and how there implementations take form in the code and in utilization. Thanks again to Kuma and her mom Arianna for the sample!