COMP316 – 18A Assignment 3 Report

Yunhao Fu
ID: 1255469

Abstract

This report is for COMP316-18A assignment 3. This is an assignment about computer vision by applying different filters on images to generate a mathematical result or description of that image.

For the question of this report is that *does any pre-processing improving the quality of the image descriptor?*

This report tries to answer this question by analysing the results between pre-processing and non-pre-processing image. Before comparing two different value, it needs to come with some new methods of pre-processing.

Pre-processing

The way to pre-processing is decided to be blurring image using linear filter, smoothing image using Gaussian smoothing filter and Laplace filter and thresholding with median of all pixels where a0 being Q1 and a1 being Q3.

$$\begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix} \begin{bmatrix} 0.075 & 0.125 & 0.075 \\ 0.125 & 0.200 & 0.125 \\ 0.075 & 0.125 & 0.075 \end{bmatrix} \begin{bmatrix} 0.25 & 0.5 & 0.25 \\ 0.5 & -3 & 0.5 \\ 0.25 & 0.5 & 0.25 \end{bmatrix} \quad f_t(a) = \begin{cases} a_0 \; a < a_{th} \\ a_1 \; a > a_{th} \end{cases}$$

Linear Filter, Gaussian Smooth Filter, Laplace Filter & Thresholding

The method to construct and apply filter is like what for kirsch filters except choosing which kirsch filter to apply. That is before targeting to one exact pixel for point operation, a more general method would be called to handle file IO and construct a nested loop to iteratively go through all pixels in the image.

To be detailed, the general method is named, for example, *blur*. It should have a filename as an argument. This method should call *read-image* method to read the image file and then assign data to a local variable declared by let function. Create an empty image with the same size of input file as well. Next, a nested loop function will be executed to conduct operation on each pixel point one by one.

The point operation function is named something-po like *smooth-po*. General idea is to generate a vector of grey data from the targeted pixel and the pixels around it. Since filter is a vector (or nested-vector) containing 9 elements, the grey data vector can easily be utilized with filter. This operation handled by a method named *two-vec-muladd* which indicating multiply two vectors and then add them up. New grey value can be generated though this way, the last step is to save the value into the pixel and return the image after processing.

Pre-processing Results

This image below is what the non-pre-processing image looks like. Originally it is in 8-bit grey.

Figure 1: Example Image – Car1.jpg

After processing, this image looks like these below.

Figure 2 & 3: Linear Filter & Gaussian Filter Applied

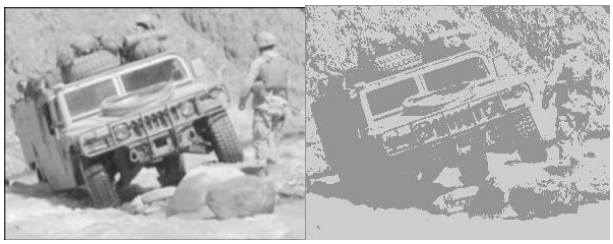

Figure 4 & 5: Laplace Filter Applied & Threshold Applied

There are three blurred images generated by applying different filters. It is quite hard to see any difference amongst them using naked-eye. However, using magnifier software can easily see the pixels are blurring around (See Figure 6). The last image using thresholding looks different.

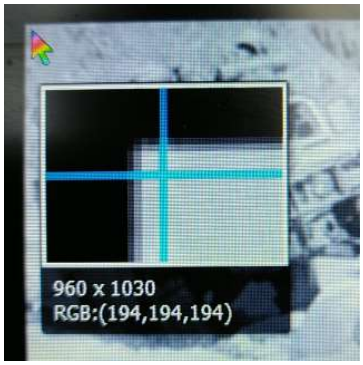

Figure 6: Details from Magnifier Software Detection

Image Description Results

Use *image-description* method to calculate each image's mathematical description. The following line charts are generated by all results.

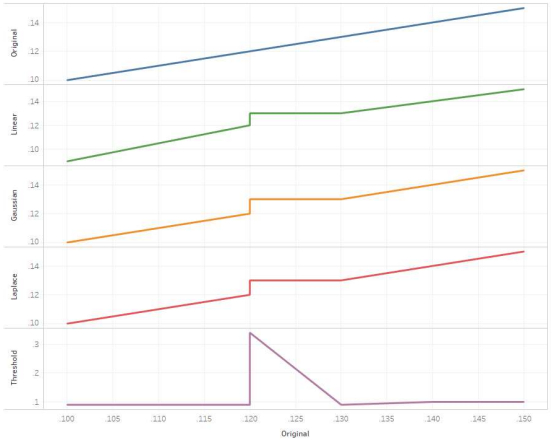

Figure 7: Edge Direction Line Chart

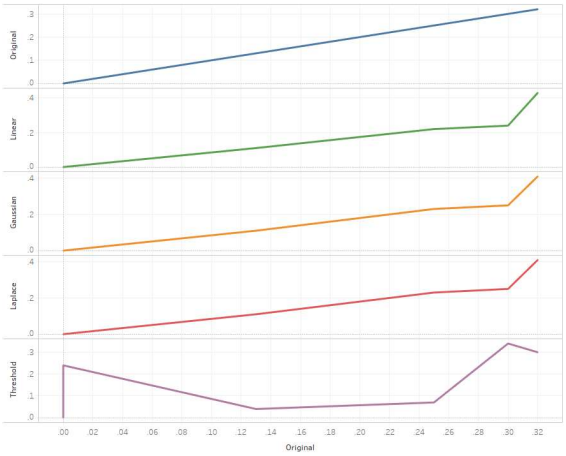

Figure 8: Edge Magnitude Line Chart

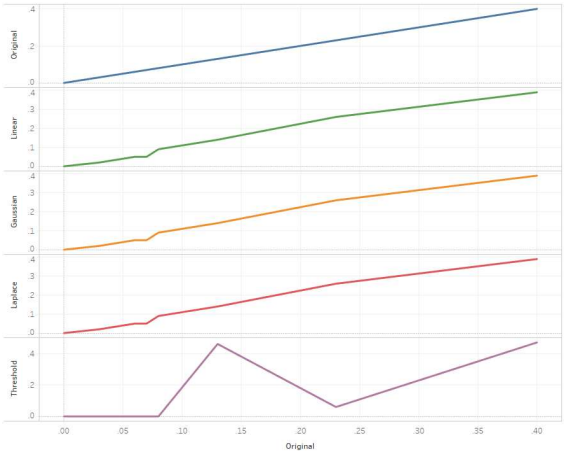

Figure 9: Normalised Intensity Line Chart

From the top to bottom, Original image, Linear, Gaussian, Laplace filter and threshold vs. Original image data are compared. Each line chart is constructed based on different attributes. First part in each line chart is perfectly linear line. Then, three different blurring filters are quite similar. But the threshold line is identically different.

Quality of Image Description

The quality of image description can be found by comparing the result of *image-similarity* function. There are six sets of images being selected and compared. A table is given below to show all details. Since the threshold function has identical difference, in this section, pre-processing method would be threshold.

Images Pair	Non-pre-processing	Pre-processing	Better?
Car10 vs. Car16	0.9067	0.9435	Yes
Car1 vs. Car4	0.7201	0.5135	Yes
Plane6 vs. Plane2	0.9502	0.9733	Yes
Plane4 vs. Plane5	0.7469	0.6201	Yes
Train18 vs. Train20	0.8468	0.8033	Kind of*
Train6 vs. Train9	0.8468	0.7501	Yes

Table 1: Comparison on Non-pre-processing & Pre-processing Images

**Since train18 and train20 are not quite similar originally, similarity datum should be reduced but not as dramatically as pair of train6 and train9.*

After compared similar and not similar image pairs, a table containing similarity and observation data generated above. The conclusion could be drawn as using threshold as pre-processing method could improve the quality of image description.
For more general extension, the following tables below are similarity data by applying different blur filters.

Images Pair	Linear Filter	Gaussian Filter	Laplace Filter
Car10 vs. Car16	0.9169	0.9169	0.9169
Car1 vs. Car4	0.7399	0.7399	0.7399
Plane6 vs. Plane2	0.9502	0.9567	0.9567
Plane4 vs. Plane5	0.7467	0.7467	0.7467
Train18 vs. Train20	0.8502	0.8502	0.8502
Train6 vs. Train9	0.8735	0.8768	0.8768

Table 2: Similarity Data after Applying Different Filters

Based on the table above, it is hard to say any pre-processing methods could make image description better. Generally, the data after applying blurring filters are slightly greater than non-pre-processing data in table 1.

Conclusion

As for conclusion, pre-processing could generally influence the image description. But as for a better quality of description is really varying. Threshold technology could make the quality of description better and clearer by binarization of intensity. In contrast, blurring methods may not work quite well to make description better.

End of Report