UNIVERSIDAD DE LOS ANDES
Facultad de Ingeniería y Ciencias Aplicadas
Digital Image Processing 2019-20

Homework N°2: Edge Detection

Tarik S. Riadi

Submission Date:
09/04/2019

I Introduction

In this document, we will be discussing methods to detect edges in images. These can be achieved by either calculating the variance of an image using a sliding window or by calculating the gradient of the image in a similar fashion. Both types of methods allow us to compare how "fast" the image changes along its pixels. Low changes tend to represent flat or blurry sections and high values of variance or gradient represent edges. Finally, we will contrast the proposed methods while explaining each one of them. All the implementations were done in Python 3 using the libraries OpenCV and Numpy.

II Standard Deviation and Variance Filters

The *standard deviation* (denoted by the Greek letter σ and also known as SD) is an indicator that measures the dispersion of a data distribution; its square is known as *variance* (σ^2). Variance, on the other hand, measures how spread out a data set is from its own mean. The implementation of these indicators is identical, as one is only the square of the other; however, they both differ significantly from one another because of the same reason.

The equations [1] that define them are presented below.

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2} \tag{1}$$

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2 \tag{2}$$

Additionally, the bias or unbiased factor may be accounted for when introducing these indicators. Bias, simply put, distinguished if one is modeling around a sample (biased) or the whole population (unbiased) [2]. For example, equations (1) are (2) are unbiased; their biased counterparts are described by the following equations.

$$\sigma = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu)^2} \tag{3}$$

$$\sigma^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu)^2 \tag{4}$$

1 Implementation

To implement[1] a filter around one of these indicators, one must first start by arbitrarily defining a window size. This window will slide along the width and height of the image that is to be filtered. For each iteration, that is, for each "stop" the window does along the image, the calculation of the indicator (either SD or variance) is made along the values within the window.

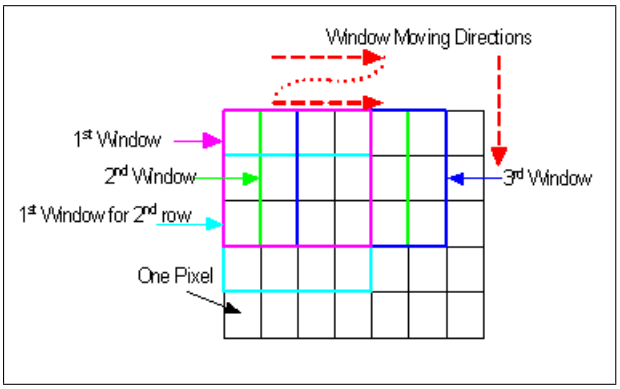


Figure 1: Sliding window along an image.[5]

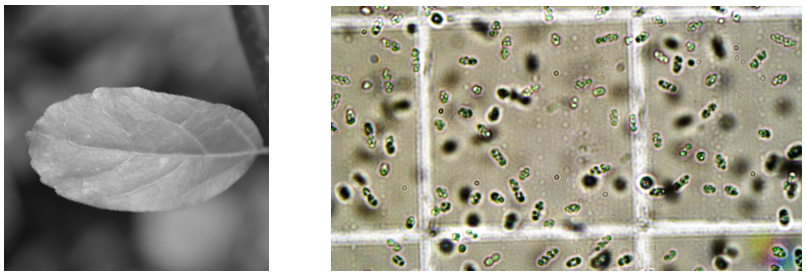
This calculation consists of getting the intensity values for each pixel of the image within the window and computing the variance of this data set. Keep in mind that all values will be an integer number between 0 and 255. Finally, the pixel's value corresponding to the central position of the window will be replaced by this result. For either one of the indicators, what this effectively does is to highlight the edges, or high gradient zones of an image, because of what these indicators do. When the SD or variance, which simply is the square of SD, is large, it means that the value of the pixels within the neighborhood vary a lot from each other, which means that the result of either indicator will be high; a high value will signify a closer-to-white color in the resulting image. Furthermore, when pixels within the window are similar

[1]To implement these filters in python, the functions numpy.var [3] and numpy.std [4] were used to calculate these indicators.

to each other, as it could happen in a greyed blurry background, the indicator value will be closer to 0, which corresponds to a color closer-to-black color.

Because of the mathematical nature of the variance, its filter is non-linear, which prevent the possibility of building a kernel with default numbers (in a fashion similar to the Sobel filter).

These filters were applied to two images: one of a focused leaf with an out-of-focus background (from now on leaf's image) and a picture taken with a microscope of some bacteria (from now on bacteria's image).

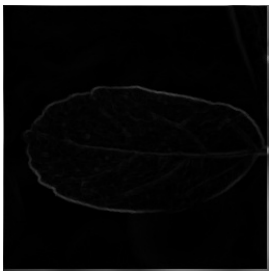


(a) Leaf's image (b) Bacteria's image

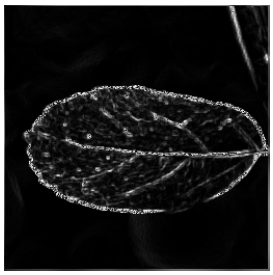
Figure 2: Original images upon the which all filters were applied.

After applying the filters based on each of the indicators, the difference becomes more evident: The σ, because is just the square root of the variance, is not as hard on the detection, therefore only highlighting the most violent changes. In contrast, the variance filter highlights more subtle edges than the SD filter. These changes can also be appreciated in the bacteria's image, even though the SD filter for this case does not stand out enough from the black background. The results of applying these filters with various window size are shown in the next page.

From these results, it is possible to conclude that the difference between using a biased and unbiased variance filter is that the first kind tend to be darker than the latter ones. This makes sense because of the equations (2)

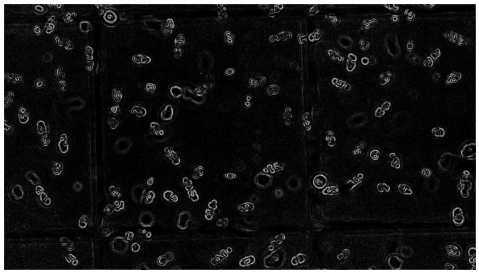


(a) SD filter in the leaf's image.



(b) Unbiased variance Filter in the leaf's image.

(c) SD Filter in the bacteria's image.

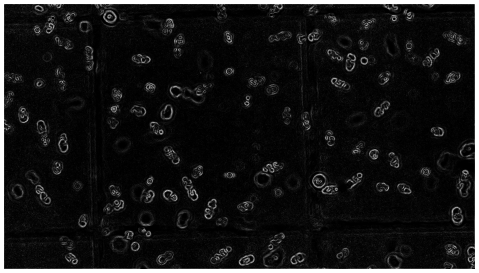


(d) Variance filter in the bacteria's image.

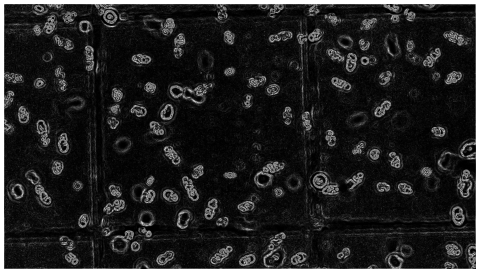
Figure 3: SD and unbiased variance filters applied to the original images using a 3x3 window.

(a) 3x3 window size in the leaf's image.

(b) 5x5 window size in the leaf's image.



(c) 3x3 window size in the bacteria's image.



(d) 5x5 window size in the bacteria's image.

Figure 4: Biased variance filters using different window sizes for both original images.

and (4): A larger denominator in the fraction makes for a smaller number, which is the case of the biased definitions. Consequently, a smaller number is closer to black than white in opencv.

2 The Relevance of Window Size

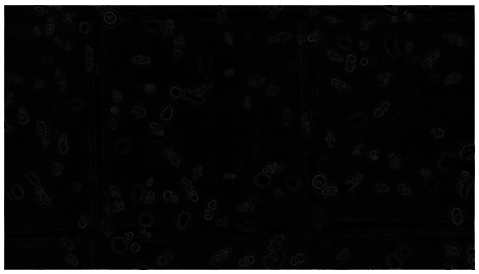
Window size will have a direct impact on the result of the detection because of the amount of pixels included in the calculation at a given time. Because of how the variance and SD filter work, the calculation will include more or less pixels to the comparison, depending on the window size; therefore, the impact of pixel in a neighborhood will be inversely proportional to the size of said neighborhood.

To better understand this, we may exaggerate a comparison between a window of size 3x3 and another of the same size of the input image. In the first case, the window will have to slide along the image, calculating the variance (or SD, depending on the filter applied) of each neighborhood. By the end, for a given neighborhood, the variance will be more representative than if we were to apply the second window. In the latter, the variance obtained will be taken on the whole image, therefore it will not be as representative of the various neighborhoods of the image, specially when it is not uniform, as could be one that has a focused object in the front and a blurry background.

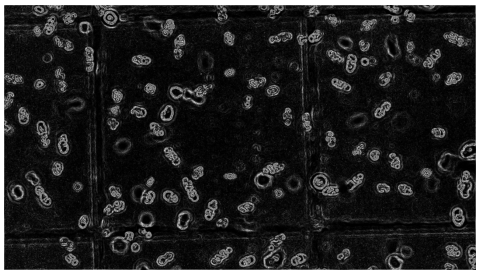
In the next page, results of applying different window sizes to the original images shown in [Figure 2].

(a) SD filter in the leaf's image.

(b) Unbiased variance Filter in the leaf's image.



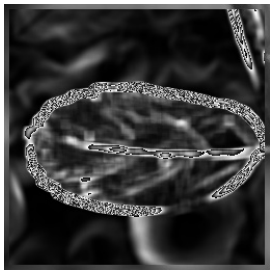
(c) SD Filter in the bacteria's image.



(d) Variance filter in the bacteria's image.

Figure 5: SD and Variance filters applied to the original images using a 5x5 window.

(a) SD filter in the leaf's image.



(b) Unbiased variance Filter in the leaf's image.

Figure 6: SD and Variance filters applied to the leaf's image using a 10x10 window.

As it can be observed, a smaller window size is better than a large one to avoid false positives in the image; however, a window size of 5x5 may be more beneficial than the 3x3 size in images like the leaf one, because it highlights better the leaf's edge. Nevertheless, this cannot be generalized, because in the case of the bacteria's image, increasing the size of the window also highlights undesired parts of the background. It can be concluded that, for different detection purposes, different sizes may be more or less useful; it will always depend on the image's composition and desired detection.

III Sobel Filter

The Sobel filter[2] works by detecting gradient changes in two directions: horizontal and vertical. Afterwards, using a gradient norm method, the previously generated images (one for each direction edge detection) are combined into a single image to create a complete image detector. Sobel uses a kernel, this means that it uses a matrix with pre-loaded values and it convolves this kernel with the image whose edges want to be found (in this case, the leaf and bacteria images).

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \tag{5}$$

$$G_y = (G_x)^T \tag{6}$$

The previous equations show the pre-loaded kernels of the Sobel filter for a kernel size of 3x3.

The results of using a Sobel filter for the detection of the original images are shown below.

[2]The sobel filter was implemented in python using the incorporated function cv2.Sobel [6].

(a) 3x3 kernel applied to the leaf's image.

(b) 5x5 kernel applied to the leaf's image.

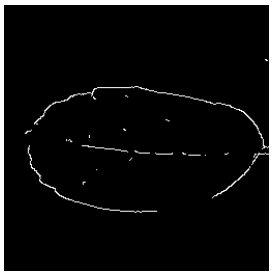
(c) 3x3 kernel applied to the bacteria's image

(d) 5x5 kernel applied to the bacteria's image

Figure 7: Sobel filters with different kernel sizes.

IV Canny Edge Filter

Another widely used filter for edge detection is the "Canny Edge Detector" [7] which works without the need of a window or kernel, but instead computes the gradient along both of the image's directions. Nonetheless, this method does require to set an hysteresis [8] lower and upper bound to set gradient limits to the detection. Wider margins between both bounds will produce more detected images in the image, whereas the opposite will happen when the margin is reduced.



(a) 3x3 kernel applied to the leaf's image.

(b) 5x5 kernel applied to the leaf's image.

Figure 8: Canny edge detectors with hysteresis bounds set to 150-200.

V Conclusion

Summing up, it is clear that we can control more accurately the edge detection using standard deviation and variance filters than with the other methods proposed, which may be helpful for more complex images like the bacteria one. However, the already implemented Sobel and Canny detectors are faster to implement than the first detectors and can obtain similar results for simpler images like the leaf's one.

References

[1] A. Fabijańska, "Variance filter for edge detection and edge-based image segmentation," *2011 Proceedings of 7th International Conference on Perspective Technologies and Methods in MEMS Design, MEMSTECH 2011*, 07 2011.

[2] mathisfun.com, "Standard deviation and variance." `https://www.mathsisfun.com/data/standard-deviation.html`, 2017. Accessed on 2019-09-03.

[3] SciPy, "Scipy variance documentation." `https://docs.scipy.org/doc/numpy/reference/generated/numpy.var.html`, July 2019. Accessed on 2019-09-03.

[4] SciPy, "Scipy standard deviation documentation." `https://docs.scipy.org/doc/numpy/reference/generated/numpy.std.html#numpy.std`, July 2019. Accessed on 2019-09-03.

[5] G. Sites, "Computer vision." `https://sites.google.com/site/computervision2014pbl/computer-vision/implementation`, 2013. Accessed on 2019-09-03.

[6] OpenCV, "Sobel derivatives documentation." https://docs.opencv.org/3.4/d2/d2c/tutorial_sobel_derivatives.html. Accessed on 2019-09-03.

[7] OpenCV, "Canny edge detector documentation." https://docs.opencv.org/3.4/da/d5c/tutorial_canny_detector.html, July 2019. Accessed on 2019-09-03.

[8] Wikipedia, "Hysteresis." https://en.wikipedia.org/wiki/Hysteresis, June 2019. Accessed on 2019-09-03.