Canny Edge Detector

Bruno Kewitz Demarchi[#1], Felipe Pilon[*2]

[#]*Systems and Computing Department, FURB*
Blumenau - Brazil

[1]demarchi.sd@gmail.com

[2]pilonfelipe@gmail.com

***Abstract*— The edge detection is widely used in imaging processing area. John F. Canny has developed what is considered the best algorithm for this task nowadays. In this article we're going to detail our algorithm implementation step by step, based on the stages defined by Canny.**

***Keywords*— Detection, edges, Canny, Sobel, Gaussian**

I. Introduction

The main reason of edge localization is the reduction of information in the image, preparing it for the subsequent operations on it, with the least as possible loss of their structural characteristics. There are many algorithms that do this work, but in this article we are going to focus particularly the one developed by John F. Canny in his PhD thesis, in 1986.

Even though it's an quite old algorithm, it has suffered many improvements, and it's still being used a lot in academic researches. It's seen by many authors as the most consistent technique in the edge localization area [2].

By a variety of uses, we can mention some examples of articles that used the edge localization in some step:

1. **Iris authentication**: used to localize the iris and the pupil in the image [3].
2. **Fingerprints authentication**: used to reduce the amount of information in the image, keeping just what really matters [4].
3. **Facial blemish removal**: used to localize the blemishes in the person's face [5].

The Canny algorithm is basically a convolution filter that uses the **first derivate**. While it's executing, it mixes noise smoothing with edge detection, combining a differential operator with a Gaussian filter.

Canny has defined that a good edge detection algorithm should achieve for the maximum three criteria:

1. **Detection**: detecting real edge points should be maximized, while the probability of detecting false edges (noise) should be minimized.
2. **Localization**: the localized edges should be as close as possible to the real edges.
3. **Number of responses**: one real edge should not result in more than one detected edge.

For that, Canny has developed an algorithm separated in five sequential steps, they are:

1. **Smoothing**: blurs the image to reduce the quantity of noise.
2. **Finding gradients**: the edges should be marked where the gradients of the image has high magnitudes.
3. **non-Maximum Suppression**: only local máxima should be marked as edges.
4. **Double thresholding**: two thresholds are used to remove the still existing noise.
5. **Edge tracking by hysteresis:** by the edges localized in the previous step, the "low" ones that are not connected to at least a "high" one should be suppressed.

Besides this five steps, our implementation also contains the image transformation to grayscale. The result can be seen in **Figure 1**.

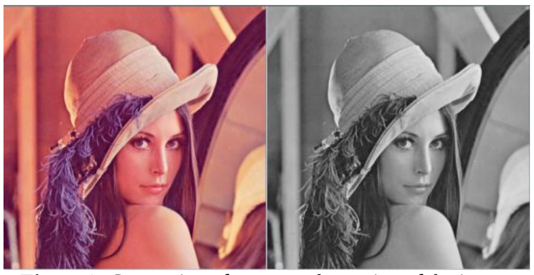

***Figure 1:** Generation of a grayscale version of the image.*

The steps defined by Canny and used in our implementation will be detailed next.

II. Smoothing

It's inevitable that images contain some amount of noise, and with the goal to avoid that they are jumbled with real edges, the image is blurred with a Gaussian filter with deviation **5**. The generated kernel is shown in **Figure 2.**

$$\left[\frac{1}{375}, \frac{1}{74}, \frac{1}{21}, \frac{1}{9}, \frac{1}{5}, \frac{1}{4}, \frac{1}{5}, \frac{1}{9}, \frac{1}{9}, \frac{1}{21}, \frac{1}{74}, \frac{1}{376}\right]$$

***Figure 2: The k**ernel used by the Gaussian filter.*

The Gaussian filter is similar to the mean filter, however it's smoother and preserves more the edges of the image because they have different weight. The result will be smoother as the deviation used grows. The result image using Gaussian filter with deviation **5** can be seen in **Figure 3**.

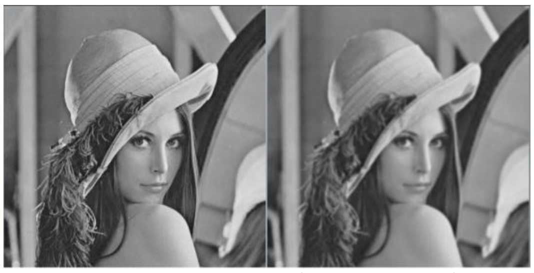
Figure 3: *Image after the Gaussian filter.*

III. Finding Gradients

In this step the algorithm searches for the biggest changes in the image's grayscale, and then the gradients of the image will be detached. For the resolution of this step the Sobel operator is used.

The Sobel operator identifies the gradient's intensity in each area of the image, giving the direction of the bigger variation of light to dark. For that we apply to the image the kernel shown in the **Figure 4**.

$$K_{\mathrm{GX}} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

$$K_{\mathrm{GY}} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Figure 4: *Kernel used by the Sobel operator.*

By this way it's possible to calculate the gradient's magnitude and also its direction, using Euclidian distance and arctangent, respectively. Both are shown in **Figure 5**.

$$|G| = \sqrt{G_x^2 + G_y^2}$$

$$\theta = \arctan\left(\frac{|G_{\mathrm{y}}|}{|G_{\mathrm{x}}|}\right)$$

Figura 5: *Euclidian distance (first) and arctangent (second)*

The result after applying the Sobel operator can be viewed in **Figure 6**.

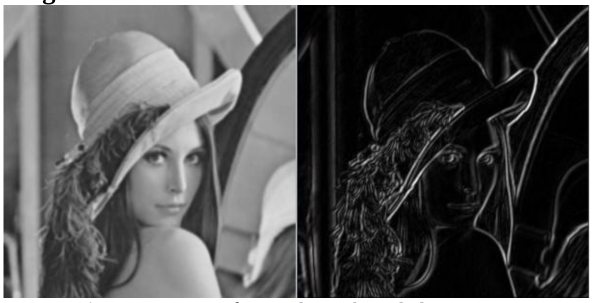
Figure 6: *Image after applying the Sobel operator.*

IV. Nom-Maximum Suppression

How the name already says, with the non-maximum suppression the points that aren't local maxima are discarded. This is done by rounding the gradient direction (obtained in the previous step) nearest to 45° and comparing the pixels in the positive and negative gradient direction (I.e. if the gradient direction is 0°, the pixel will be compared with his left and right neighbor). If the pixel isn't bigger than both of its neighbors, it will be suppressed. We have an illustration of how it works in **figure 7**. The arrow represents the direction (in this case most of them are pointing north), and the not detached pixels are the suppressed ones.

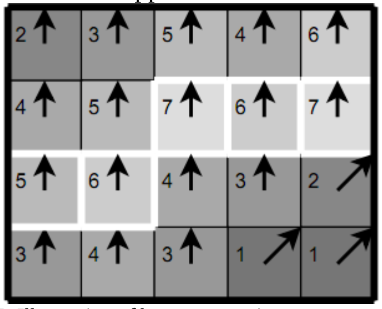

Figure 7: *Illustration of how non-maximum suppression works*

The result of this operation in the gradient image can be seen in **Figure 8.**

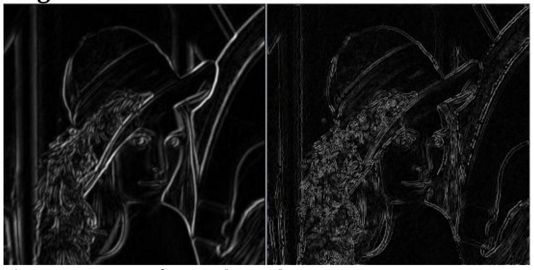
Figure 8: *Image after applying the non-maximum suppression.*

V. Double Thresholding

In this penultimate step the goal is similar to the one in the first step: reduce the amount of noise considered edge. For that we use two thresholds: one with the low value, and the other with the high value.

Basically, the image is traveled by pixel to pixel, and for each the following operation is made: if the intensity of the pixel is bigger than the high threshold, then the pixel is considered strong. But, if the intensity is in between the high threshold and the low threshold, the pixel is considered weak. Finally, if the intensity is lower than the low threshold, the pixel is suppressed.

In the example of the **Figure 9**, just one pixel, colored in white, is considered strong. The pixels that have intensity between the high and low threshold, considered weak, are colored in grey. The pixels with intensity lower than the low threshold, the ones that are suppressed, are detached in black.

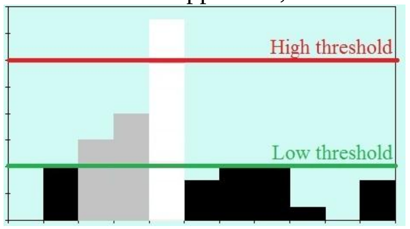

Figure 9: *Histogram representing the Double Thresholding*

The result of the Double Thresholding application in the non-Maximum suppression image is in the **Figure 10**.

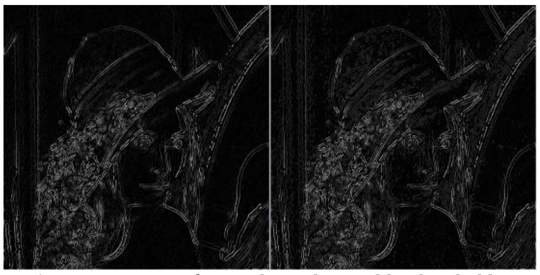
Figure 10: *Image after applying the Double Thresholding*

VI. Edge Tracking by Hysteresis

In the image resulting of the Double Thresholding, there still might be pixels that are marked as edge that shouldn't be. There might be noises that were detected as edges, for that in the Edge Tracking by Hysteresis step a tracking is done, so these false edges are removed from the image.

The tracking is done checking if the edges considered weak are directly connected with the strong edges. This is done by BLOB-Analysis, where the image is travelled by 8-connected groups. If in the group analyzed there is at least one strong pixel, the weak pixels of that group are considered as real edges. However, if there is a group with some weak pixels, but there is no strong pixel, the weak pixels are discarded.

In the example of **Figure 11**, in the first image the gray pixels (weak ones) are discarded. But, in the second image, they are considered real edge pixels because the group also contains white pixels (strong ones).

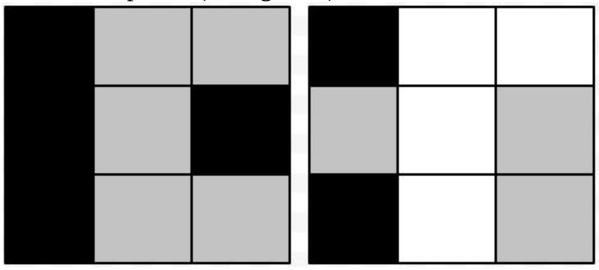
Figure 11: *Edge Tracking by Hysteresis representation*

After the application of the Edge Tracking by Hysteresis, we obtain the final image containing the edges of the original image. The result can be seen in the **Figure 12**.

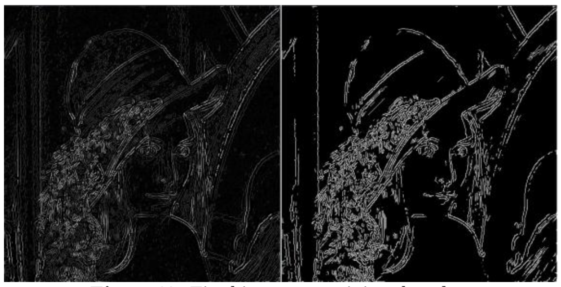
Figure 12: *Final image containing the edges*

VII. Conclusions

During the implementation, the greatest difficulty was to align the Sobel operator and the non-Maximum Suppression methods. In some codes studied for the implementation, like the example developed by Gibara [1], these two steps are made in a single block, where the non local maxima pixels localization is made during the identification of the gradients of the image, using then the immediately result.

After the application of all the steps, the result obtained has served the items **1** and **2** of the criteria defined by Canny for a good edge detection algorithm, because the level of noise wrongly detected as edge were low and the edges localized were as close as possible to the real edges. However, the item **3** was not attended, since the quantity of double edges generated was high. The tests were made with many images having different characteristics, and, even by this, in all of the tests the edges of the images where localized.

References

[1] - T. Gibara. (2009) Canny Edge Detector Implementation. [Online]. Available: http://www.tomgibara.com/computer-vision/canny-edge-detector

[2] - F. Silva and C. Alves. (2008) Aplicação de técnicas de processamento de imagens digitais em imagens geradas por ultra-som. [Online]. Available: http://www.dimap.ufrn.br/~sbmac/ermac2008/Anais/Resumos%20Estendidos/aplica%E7%E3o%20de%20tecnicas%20de%20PS_Silva.pdf

[3] - E. Delgado. (2009) Reconhecimento de Íris Usando Transformada de Wavelet e Zero-Crossing. [Online]. Available: http://www.pee.ufrj.br/teses/textocompleto/2009082801.pdf

[4] - M. Boldischar and C. Moua. [Online]. Available: http://www2.uwstout.edu/content/rs/2007/Edge%20Detection.pdf

[5] - J. Marino and G. Yoblin. (2005). [Online]. Available: http://homepages.cae.wisc.edu/~ece533/project/f05/Marino_Yoblin_Report.pdf