Project #2

Fall 2021

Willie Chang

Abstract

In the second half of the course, I have learned about the ideas and algorithms of *Image Enhancement* techniques. Here we apply *"Robert's Gradient Convolution"* as the Kernel of size 2×2 with L2 norm, which is a cross-gradient operator based on the diagonal differences.

In this project, I am going to write my own algorithms without the built-in functions in Matlab for convolution processes. By convolving the kernels to the image, I can then get the edge enhanced image with five different conditions which will be shown in the following topics.

Introduction

Edge detection is one of the most important methods in image processing. To find the edges in the image, we need to look for the large differences from the intensities of local pixels. Those kind of huge variations from locals are usually the edges of the objects (except for the noise).

Since we want to find the edges in the input image, shown in Fig. 1, we apply the *"Robert's Gradient"* method with the five given conditions.

Fig. 1. Input image examples for wavelet denoising process

Process

Firstly, read the image and turn the data types into *double* so that the Matlab system can do further convolution processing.

Robert's gradient masks is then given as:

$$H_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

$$H_2 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

For the convolution process, I move the mask from the first pixel $(1,1)$ to the last pixel $(512,512)$. The expanded one row and one column copy the former row and column from the last of them from preventing the distortion. By convolving the kernels with each local pixels, we can get the gradient values of the local pixels.

Lastly, I compare the gradient values with the threshold $T = 25$ which is the given condition to generate the gradient array of image, and the results are shown below:

(i) Method 1

$$g(x,y) = G[f(x,y)]$$

Since $g(x,y)$ is the gradient value of local pixels, the smooth part of the image will appear dark, while the edges will appear brighter.

This shows the details *(high frequencies)* of the image but removing the background.

(a) Original image (b) Edge enhancement image

(ii) Method 2

$$g(x,y) = \begin{cases} G[f(x,y)], & if\ G[f(x,y)] \geq T \\ f(x,y), & otherwise \end{cases}$$

If the calculated gradient is larger than the given threshold, then emphasize those edges without changing the values of the smooth part from the original image.

This makes the object clearer with the highlighted edges.

(a) Original image (b) Edge enhancement image

(iii) Method 3

$$g(x,y) = \begin{cases} L_G, & if\ G[f(x,y)] \geq T \\ f(x,y), & otherwise \end{cases}$$

If the calculated gradient is larger than the given threshold, then modify those edges to a given gray level value L_G *(in this project, we set as white)* without changing the smooth part from the original image.

In the following result, we can still see some edges that are set as white color, but since the background are mostly light colors, it is not as easy as the previous methods to see the highlighted edges.

(a) Original image

(b) Edge enhancement image

(iv) Method 4

$$g(x,y) = \begin{cases} G[f(x,y)], & if\ G[f(x,y)] \geq T \\ L_B, & otherwise \end{cases}$$

If the calculated gradient is larger than the given threshold, then emphasize those edges while modify the smooth part to a given value L_B (*in this project, we set as black)*.

We can see that the highlighted edges are clear to distinguish from the background.

(a) Original image

(b) Edge enhancement image

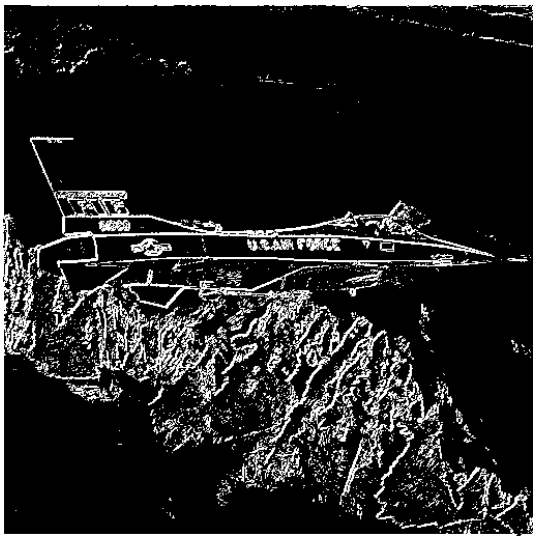
(V) Method 5

$$g(x,y) = \begin{cases} L_G, & if\ G[f(x,y)] \geq T \\ L_B, & otherwise \end{cases}$$

If the calculated gradient is larger than the given threshold, then we set the values to L_G, while modify the smooth part to a given value L_B (*in this project,* $L_G = 255, L_B = 0$*)*.

In the following result, the edges are clearly shown due to the high contrast of white and black.

(a) Original image



(b) Reconstruction image

Conclusion

For the results of each different situation, we can see that the edges are highlighted in different ways. Even though the edges are emphasized, "*Robert's Gradient Convolution Masks"* are not useful for computing the edges for the kernels that are symmetric to their centers. Besides, there are even more different techniques and masks for edge enhancements.

For the future works, fast convolution methods should be implemented due to the large dataset that might be encountered. Different edge enhancement techniques should also be presented for different conditions, computation complexity, and enhanced performance.