**LAPORAN PRAKTIKUM PENGOLAHAN CITRA DIGITAL**

**18. EDGE DETECTION**



**Disusun oleh :**

**Nama : Garcia Bryan Farrel**

**NPM : 2327250026**

**Kelas : IF41**

**PROGRAM STUDI INFORMATIKA**

**FAKULTAS ILMU KOMPUTER DAN REKAYASA**

**UNIVERSITAS MULTI DATA PALEMBANG**

**2024**

**TUTORIAL : EDGE DETECTION**

**Goal**

The goal of this tutorial is to learn how to implement edge detection and associated

techniques in MATLAB.

**Objectives**

* Learn how to use the IPT edge function.
* Explore the most popular first-derivative edge detectors: Roberts, Sobel, and Prewitt.
* Explore the Marr–Hildreth Laplacian of Gaussian edge detector.
* Explore the Canny edge detector.
* Learn how to implement edge detection with compass masks (Kirsch and Robinson).

**What You Will Need**

* lenna.tif
* mandrill.tif

**Procedure**

First-order edge detection methods, such as Prewitt and Sobel, are defined as convolution kernels. As we have seen in previous tutorials, convolution can be executed in MATLAB using the imfilter function. Although we could use this function to implement edge detection, we would be required to perform additional tasks, such as convolving the image twice (once for horizontal edges, and another for vertical ones) and adding the absolute values of these results to yield the final image. In practice, this image might also then be thresholded to produce a binary image where white pixels would represent edges. The function edge will do all this for us and will even determine a threshold value if we choose not to specify one.

**Edge Detection Using the Prewitt Operator**

1. Load and display the test image.



|  |  |
| --- | --- |
|  |  |

1. Extract the edges in the image using the Prewitt operator.



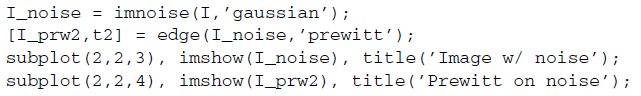
|  |  |
| --- | --- |
|  |  |

**Question 1** What does the t1 variable represent?

|  |
| --- |
| t1 adalah nilai threshold otomatis yang digunakan oleh metode Prewitt untuk mendeteksi tepi pada citra. |

Edge detection methods are often compared by their ability to detect edges in noisy images. Let us perform the Prewitt operator on the Lenna image with additive Gaussian noise.

1. Add noise to the test image and extract its edges.



|  |
| --- |
|  |
|  |

**Question 2** How did the Prewitt edge detector perform in the presence of noise (compared to no noise)?

|  |
| --- |
| hasil deteksi tepi pada gambar dengan noise akan tampak lebih berantakan, tidak rapi, dan terdapat banyak titik-titik tepi yang tidak relevan dibandingkan dengan gambar tanpa noise. |

**Question 3** Did MATLAB use a different threshold value for the noisy image?

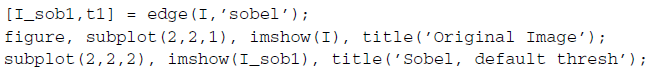
|  |
| --- |
| Iya menghitung nilai threshold secara otomatis jika kita tidak menentukan nilai threshold secara manual. Jadi nilai t2 threshold untuk gambar dengan noise berbeda dari t1 threshold untuk gambar asli. |

**Question 4** Try using different threshold values. Do these different values affect the operator’s response to noise? How does the threshold value affect the edges of the object?

|  |
| --- |
| Semakin kecil nilai threshold lebih banyak tepi yang terdeteksi, termasuk noise.  Semakin besar nilai threshold hanya tepi dengan perubahan intensitas lebih rapi. |

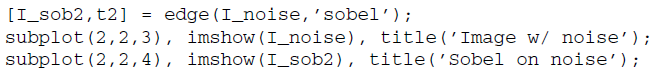
**Edge Detection Using the Sobel Operator**

1. Extract the edges from the test image using the Sobel edge detector.



|  |  |
| --- | --- |
|  |  |

1. Extract the edges from the test image with Gaussian noise using the Sobel edge detector.



|  |  |
| --- | --- |
|  |  |

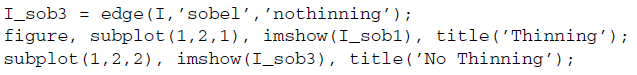
**Question 5** How does the Sobel operator compare with the Prewitt operator with and without noise?

|  |
| --- |
| Tanpa noise hasil cukup baik, tepi jelas sedikit lebih tajam dan halus. Dengan noise lebih banyak tepi palsu dan hasil terlihat lebih tahan terhadap noise. |

Another feature of the edge function is ’thinning’, which reduces the thickness

of the detected edges. Although this feature is turned on by default, it can be turned off, which results in faster edge detection.

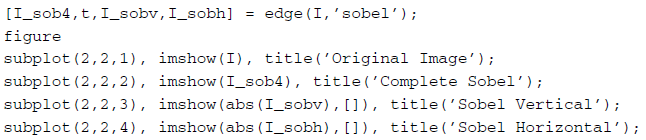
1. Extract the edges from the test image with the Sobel operator with no thinning.



|  |  |
| --- | --- |
|  |  |

As you already know, the Sobel operator actually performs two convolutions (horizontal and vertical). These individual images can be obtained by using additional output parameters.

1. Display the horizontal and vertical convolution results from the Sobel operator.



|  |
| --- |
|  |
|  |

**Question 6** Why do we display the absolute value of the vertical and horizontal images? Hint: Inspect the minimum and maximum values of these images.

|  |
| --- |
| Gradien vertikal ‘I\_sobv’ mendeteksi perubahan dari atas ke bawah.  Gradien horizontal ‘I\_sobh’ mendeteksi perubahan dari kiri ke kanan. |

**Question 7** Change the code in step 7 to display thresholded (binarized), not

thinned, versions of all images.

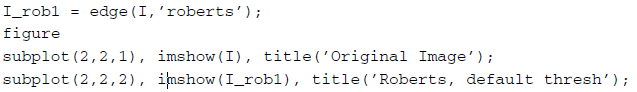
|  |
| --- |
| memberi nilai threshold t dan komponen vertikal ‘I\_sobv’ dan horizontal  ‘I\_sobh’. |

As you may have noticed, the edge function returns the vertical and horizontal images before any thresholding takes place.

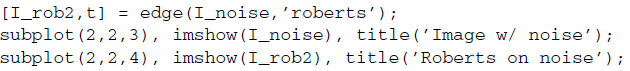
**Edge Detection with the Roberts Operator**

Similar options are available with the edge function when the Roberts operator is used.

1. Extract the edges from the original image using the Roberts operator.



1. Apply the Roberts operator to a noisy image.



|  |
| --- |
|  |
|  |

**Question 8** Compare the Roberts operator with the Sobel and Prewitt operators. How does it hold up to noise?

|  |
| --- |
| Tidak bisa saat ada noise, dan menghasilkan banyak tepi palsu di gambar .  Sobel lebih tahan terhadap noise karena kernel lebih besar.  Prewitt sedikit lebih sensitif daripada sobel tapi masih lebih baik dari Roberts. |

**Question 9** If we were to adjust the threshold, would we get better results when filtering the noisy image?

|  |
| --- |
| Tepi asli yang penting bisa hilang dan banyak noise di anggap tepi. |

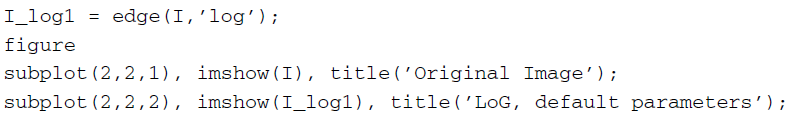
**Question 10** Suggest a method to reduce the noise in the image before performing edge detection.

|  |  |
| --- | --- |
|  |  |

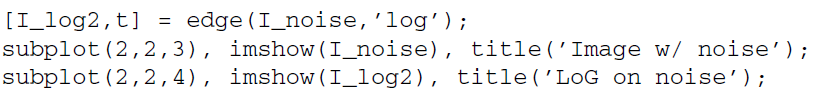
**Edge Detection with the Laplacian of a Gaussian Operator**

The LoG edge detector can be implemented with the edge function as well. Let us see its results.

1. Extract the edges from the original image using the LoG edge detector.



1. Apply the LoG edge detector to the noisy image.



|  |
| --- |
|  |
|  |

**Question 11** By default, the LoG edge detector uses a value of 2 for σ (the standard deviation of the filter). What happens when we increase this value?

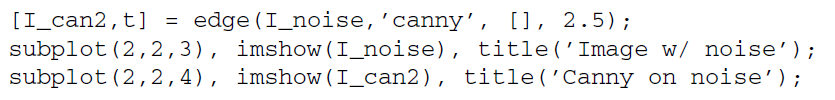
|  |  |
| --- | --- |
| Meningkatkan σ pada LoG akan mengurangi noise dan tepi kecil. |  |

**Edge Detection with the Canny Operator**

1. Extract the edges from the original image using the Canny edge detector.



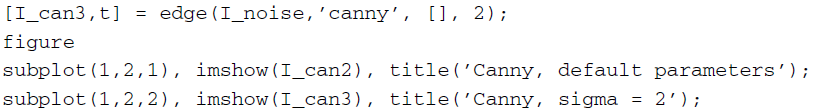
1. Apply the filter to the noisy image.



|  |
| --- |
|  |
|  |

As you know, the Canny detector first applies a Gaussian smoothing function to the image, followed by edge enhancement. To achieve better results on the noisy image, we can increase the size of the Gaussian smoothing filter through the sigma parameter.

1. Apply the Canny detector on the noisy image where sigma = 2.



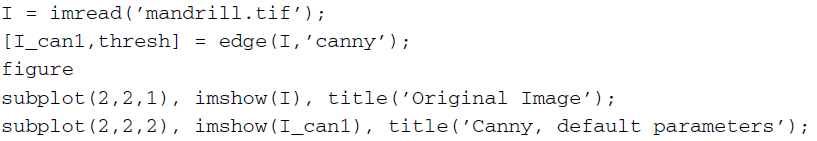
|  |
| --- |
|  |
|  |

**Question 12** Does increasing the value of sigma give us better results when using the Canny detector on a noisy image?

|  |
| --- |
| Tidak dia malah mengabil noise lebih banyak sehingga gambar sedikit rusak. |

Another parameter of the Canny detector is the threshold value, which affects the sensitivity of the detector.

1. Close any open figures and clear all workspace variables.
2. Load the mandrill image and perform the Canny edge detector with default parameters.



1. Inspect the contents of variable thresh.
2. Use a threshold value higher than the one in variable thresh.



1. Use a threshold value lower than the one in variable thresh.



|  |
| --- |
|  |
|  |

**Question 13** How does the sensitivity of the Canny edge detector change when the threshold value is increased?

|  |
| --- |
| Iya pakai canny mengurangi sensitivitas sehingga hanya mendeteksi tepi yang kuat. Menurunkan threshold meningkatkan sensitivitas, tetapi bisa menyebabkan deteksi tepi yang berlebihan. |

**Edge Detection with the Kirsch Operator**

The remaining edge detection techniques discussed in this tutorial are not included

in the current implementation of the edge function, so we must implement them as

they are defined. We will begin with the Kirsch operator.

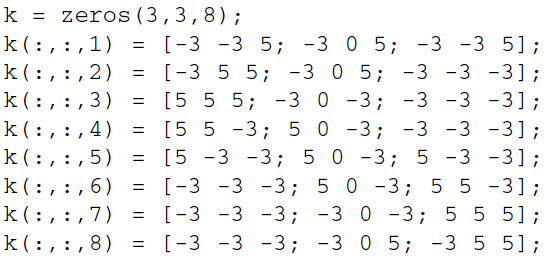
1. Close any open figures and clear all workspace variables.
2. Load the mandrill image and convert it to double format.

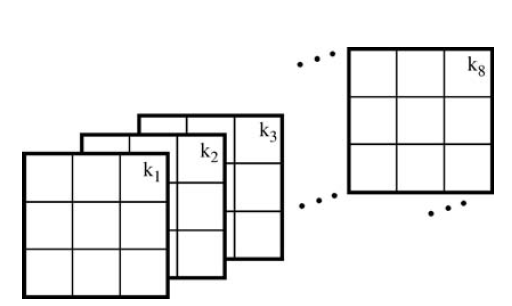


Previously, when we were using the edge function, we did not need to convert the image to class double because the function took care of this for us automatically. Since now we are implementing the remaining edge detectors on our own, we must perform the class conversion to properly handle negative values (preventing unwanted truncation).

Next we will define the eight Kirsch masks. For ease of implementation, we will store all eight masks in a 3 × 3 × 8 matrix. Figure 18.1 illustrates this storage format.

1. Create the Kirsch masks and store them in a preallocated matrix.

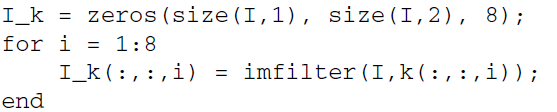




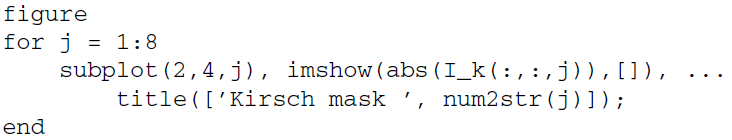
**FIGURE 18.1** Kirsch masks stored in a 3 × 3 × 8 matrix.

Next we must convolve each mask on the image, generating eight images.We will store these images in a three-dimensional matrix just as we did for the masks. Because all the masks are stored in one matrix, we can use a for loop to perform all eight convolutions with less lines of code.

1. Convolve each mask with the image using a for loop.



1. Display the resulting images.



|  |
| --- |
|  |
|  |

**Question 14** Why are we required to display the absolute value of each mask? Hint: Inspect the minimum and maximum values.

|  |
| --- |
| Hasil konvolusi dengan mask (filter) bisa berupa nilai positif dan negatif.  Nilai negatif muncul karena filter mendeteksi perubahan intensitas ke arah tertentu (misalnya tepi yang gelap-ke-terang vs terang-ke-gelap). |

**Question 15** How did we dynamically display the mask number when displaying all eight images?

|  |
| --- |
| Pakai code ini  Jadi, num2str(i) mengubah angka i menjadi string yang bisa digabungkan dengan teks untuk judul subplot. |

Next we must find the maximum value of all the images for each pixel. Again,

because they are all stored in one matrix, we can do this in one line of code.

1. Find the maximum values.



|  |  |
| --- | --- |
|  |  |

**Question 16** When calculating the maximum values, what does the last parameter in the max function call mean?

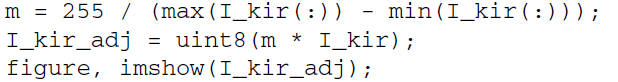
|  |
| --- |
| Parameter 3 menunjukkan dimensi sepanjang mana operasi max dilakukan. jadi I\_k adalah array dengan ukuran baris, kolom, 8, dimana dimensi ke3 adalah jumlah mask/filter. Jadi, max(I\_k, [], 3) artinya cari nilai maksimum di sepanjang dimensi ke3 yaitu di antara 1-8 hasil filter untuk setiap piksel. Kesimpulannya untuk setiap piksel ambil nilai terbesar dari 8 hasil filter. |

**Question 17** Why are we required to scale the image when displaying it?

|  |
| --- |
| Melakukan scaling saat menampilkan gambar agar rentang nilai piksel yang tidak standar (misalnya nilai negatif atau lebih besar dari 1) bisa dipetakan dengan benar ke rentang warna, sehingga gambar tampil dengan kontras yang sesuai dan detailnya terlihat jelas. |

In the previous step we scaled the result for display purposes. If we wish to threshold the image (as we did with all previous edge detectors), we must first scale the image so that its values are within the range [0, 255] as well as convert to class uint8. To do so, we can create a linear transformation function that maps all current values to values within the range we want.

1. Create a transformation function to map the image to the grayscale range and perform the transformation.



|  |  |
| --- | --- |
|  |  |

**Question 18** Why is it not necessary to scale this image (I\_kir\_adj) when displaying it?

|  |
| --- |
| Karena gambar I\_kir\_adj sudah dikonversi menjadi tipe uint8, yang secara otomatis memiliki rentang nilai piksel 0–255. |

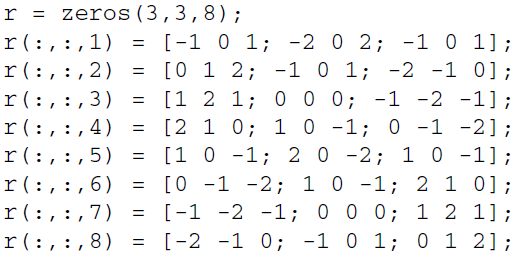
**Question 19** Make a copy of the mandrill image and add Gaussian noise to it. Then perform the Kirsch edge detector on it. Comment on its performance when noise is present.

|  |  |
| --- | --- |
| Kirsch cukup sensitif terhadap noise. Noise kecil pun bisa menghasilkan nilai gradien tinggi banyak tepi palsu muncul akibat noise. |  |
|  |  |

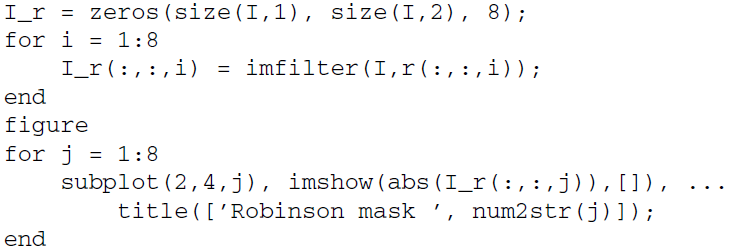
**Edge Detection with the Robinson Operator**

The Robinson edge detector can be implemented in the same manner as the Kirsch detector. The only difference is the masks.

1. Generate the Robinson masks.



1. Filter the image with the eight Robinson masks and display the output.



|  |
| --- |
|  |
|  |

1. Calculate the max of all eight images and display the result.



|  |  |
| --- | --- |
|  |  |

**Question 20** How does the Robinson edge detector compare with the Kirsch detector?

|  |
| --- |
| Kirsch cocok digunakan ketika ingin mendeteksi tepi yang kuat dan tajam, tetapi kurang cocok untuk gambar yang mengandung banyak noise.  Robinson lebih baik untuk menghasilkan deteksi tepi yang lebih halus dan stabil terutama ketika gambar mengandung noise kecil. |