Task 1

1. Sobel operator:

Sobel Operator is basically 2D spatial gradient (Horizontal & Vertical direction) measurements for edge detection using the 1st order derivatives. It is a basic operator which is used widely to detect the edge. sobel operator is very slow than robert’s operator. But if we are using larger kernel size, that will make operator less sensitive to noise. Sobel operator is using when the image edge has sufficient contrast from the background

Matlab function: I = edge(Image,’sobel’,Threshold,Direction)

In my understanding, it is simple matlab function where we have to give threshold value (it is basically used to cutoff some of the edge detection data which is not in the consideration like noise). Here My threshold is 0.03. And the direction is used to give user input for in which direction the gradient is needed? (horizontal or vertical or both).By default matlab will take both directions (also for prewitt & roberts). In fig 1, we can say that the edge details (Hair & Eye) on the girl is very good as compare to background. And some part of the background is not detected (High-lighted circle). This happening due to on that edge, they have very less contrast compare to background (surroundings). If I tried to go below this threshold, it gives some noise(Surrounding details) details in the background. And my major concern is to detect the small details on the girl. Here the noise means we have some background information which is not the major concern; my concern is to detect the edge which is perceived easily by human eyes So that small parts in the background consider as noise in my next writing.I have used the zero pad of [10 10] from both the side of the image.

G:\hemal_spring_19\image processing\Hw\HW 6\6S.tif

Fig 1: Sobel operator.

1. Prewitt Operator:

Prewitt Operator is working same as Robert’s. But the main difference is the mask of gradient Prewitt operator is giving same weight on whole mask.

Matlab function : I = edge(Image,’prewitt’,Threshold,Direction.)

Here for this operator my threshold is 0.035. In the fig 2, this threshold will eliminate the detection of noise in the background. The major thing is it is detect same as sobel. The tiny details on hat, face and hair is easily visible. But due to less contrast of the background edge, we cannot not detect the highlighted area in the background. Because this operator works good when we have good contrast between the edge and background.it is good that it still detect the edge in the mirror.

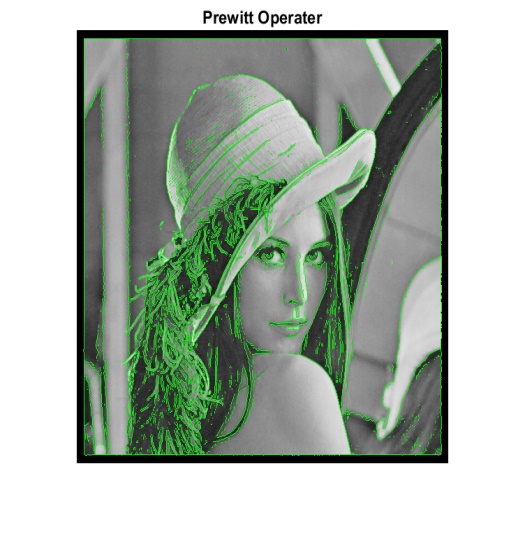


Fig 2 : Prewitt Operator.

1. Robert’s Operator:

Robert’s operate is the simplest, efficient and 1st edge detection operator. It works well as compare to sobel and prewitt. Because it can give the edge detection that what human can perceive easily from the necked eyes by directly seeing the image.

Matlab function : I = edge(Image,’roberts’,Threshold,Direction.)

Here the threshold is choosing 0.035. In fig 3, it can detect good details in hair face and cap with less noise. But still it is unable to detect the background highlight circle in fig 3, due to less contrast between edge and surrounding of the edge.

G:\hemal_spring_19\image processing\Hw\HW 6\6R.tif

Fig 3: Robert’s operator.

1. Canny Operator:

Canny Operator is very famous for detecting wide range of edge that is using basically for detecting structural information from the image. This operator can detect the edge with less noise because it approximated with 1st derivative of the Gaussian.

Matlab Function : I = edge(Image,'canny',threshold,sigma);

Here I have used threshold 0.06 and sigma is by default sqrt(2). In my understanding canny works well to detect most of the edges perfectly. As in Fig 4, we can see the smooth detection of edge in all over the image. We can see fine detailed structure in hat, hair and face. In my observation it is detecting most of all the edges in this image. For 1st 3 operators, due to less contrast, some edge details are not detectable. But with the canny we can detect that all the edge in the background.

G:\hemal_spring_19\image processing\Hw\HW 6\6C.tif

Fig 4 : Canny Operator.

1. LoG operator:

LoG operator is working well with the detection of edge as well as noise in the background. So algorithms basically smooth the image to suppress the noise before the convolution.

Matlab Function: I = edge(Image,'log',Threshold, sigma)

Here I have used threshold 0.02 and sigma is 1. In my understanding using less sigma will give smoother small details of the edge. Because using high sigma will give too much details of noise in the background of the image. In fig 5, we can see that the small details on girl in the image are efficiently detected like in the hat and face (Highlighted with the circle). But the background edge information is not optimal due to less sigma.

G:\hemal_spring_19\image processing\Hw\HW 6\6L.tif

Fig 5: LoG Operator.

1. Zero crossing method:

Zero crossing detector looks for the places where the intensity of the image is changing rapidly.it is basically known for the feature detector. It is also using LoG for filtering the noise first.

Matlab Funcrion: BW5 = edge(Image,'zerocross',threshold);

Here I am using threshold is 0.002. In the fig 6, the most of the smooth details of the hair, cap and face is detectable (Highlighted with the circle). But some of the details in the background are missing. If I will use the lower threshold than it is detectable but background noise is detectable. That is not our major concern.

G:\hemal_spring_19\image processing\Hw\HW 6\6Z.tif

Fig 6: Zero crossing operator.

Task 2:

For my own edge detection Matlab code, the Frei and Chen’s edge algorithm is implemented. The major reason behind to do it is it can find the Edges and lines separately. So the Frei and Chen’s edge detection algorithm is in the form of a complex set of basis vectors that will represent a weighted sum of the 9 different masks. The starting four masks (shown below) are suitable for detecting the edge and second four masks are suitable for detecting lines. The second four masks are proportional to the average of the pixels in the region at which the mask is located in the image. For the implementation procedure is explained below:

Step 1: Read the Image and do the padding. Here I am using 3 x 3 kernel, So I have padded my image according to that.

Step 2: Frei – Chen operator used total 9 different kernel. Each kernel is working independently.

Step 3: Applying Column gradient for the detection of horizontal edge.(Gradient direction is shown with dashed line)

\*1/(2 + )

Step 4: Applying row gradient for the detection of Vertical edge. Gradient direction is shown with dashed line)

\*1/(2 + )

Step 5: Applying the gradient for the detection of -45 degree edge. Gradient direction is shown with dashed line)

\*1/(2 + )

Step 6: Applying the gradient for the detection of 45 degree edge. Gradient direction is shown with dashed line)

\*1/(2 + )

Step 7: Applying the next 4 gradient Detection kernel to detect line subspace.

\*(1/2)

\*(1/2)

\*(1/6)

\*(1/6)

Step 8: this is the average gradient filter on whole Image.

\*(1/3)

Step 9: Finding the weighted sum of all the masks. I have tried the approach of find the maximum value by doing point to point matrix operation on all 8 masks. But I have followed the theory of how they are finding the final edge details. I have eliminated the x9 mask from the finding weighted sum because the averaging mask will give me fine edge details(Like the hair) instead of that it is giving me all over edge thing(only face cut, some one can assume that this mask will not detect the inside of the hair detail bur detect only outer portion of the hair). For the convolution I am directly using imfilter matlab command.

(Here Image = our original Image and doing the convolution)

Step 10: Convert the Image into binary image and created a mask with thresholding (factor is 0.09). This factor is chosen because if I am going below this factor, the background noise is coming in the Image (we can see the background noise as an edge). Ansd that is not the major concern. And the BW mast clearly shows the most of the edges in the background as well as the object (girl).

Step 11: I am using bwmorph (with input ‘remove’) matlab function to see the edge clearly.in my understanding the input ‘remove’ is used to leave only the boundary pixels on (here our boundary pixel is edge). Thus it will remove the interior pixels of the thick edge and give us a nice thin edge.

The major thing I observed is I can see it is not differentiate the Overlapping edge, so at that time the two edges will consider as 1 edge. So sometimes some of the minor details we cannot see. See the highlighted in the image 7. But after all if you can see other filters(Fig 1 - 6), my filter can differentiate the 2 identical edges(side by side) easily (Highlighted with blue in fig 7.). This will give good results where we have 2 edges that are very near to each other. Because this filter is the combination of edge with line detection. SO it is working efficiently on this domain. But still the background results are missing. If I tried to decrease my threshold, will give unnecessary details in the background.

Results:

G:\hemal_spring_19\image processing\Hw\HW 6\M.tif

Fig 7: Overlayed Image with threshold = 0.09

G:\hemal_spring_19\image processing\Hw\HW 6\BW_mask.tif

Fig 8 : BW mask of Edge