# 

Text

Description automatically generated

**CE4003: Computer Vision**

**Lab 2:**

Edge Detection + Hough Transform+ Stereo Vision+ SPM

Prepared by:

Venkat Subramanian (U1921075D)

Contents

[Edge Detection 2](#_Toc118641407)

[Part a 2](#_Toc118641408)

[Part b 2](#_Toc118641409)

[Part c 3](#_Toc118641410)

[Part d 4](#_Toc118641411)

[Part e 5](#_Toc118641412)

[Hough Transform 6](#_Toc118641413)

[Part a 6](#_Toc118641414)

[Part b 7](#_Toc118641415)

[Part c 8](#_Toc118641416)

[Part d 8](#_Toc118641417)

[Part e 9](#_Toc118641418)

[Part f 9](#_Toc118641419)

[3D Stereo 10](#_Toc118641420)

[Part a 10](#_Toc118641421)

[Part b 11](#_Toc118641422)

[Part c 11](#_Toc118641423)

[Part d 12](#_Toc118641424)

[Extra Section 13](#_Toc118641425)

[Conclusion 14](#_Toc118641426)

[References 15](#_Toc118641427)

# Edge Detection

In this section, we will be exploring kernels and methods used for edge detection in an image. An edge is an image contour across which the image’s brightness changes abruptly.

## Part a

Code

Text

Description automatically generated with medium confidence

Output

A group of people running

Description automatically generated with medium confidence

The MacRitchie image is a coloured image, for edge detection we need to convert the 3 channelled images into a single channel. Therefore the image was converted to a grayscale image before edge detection.

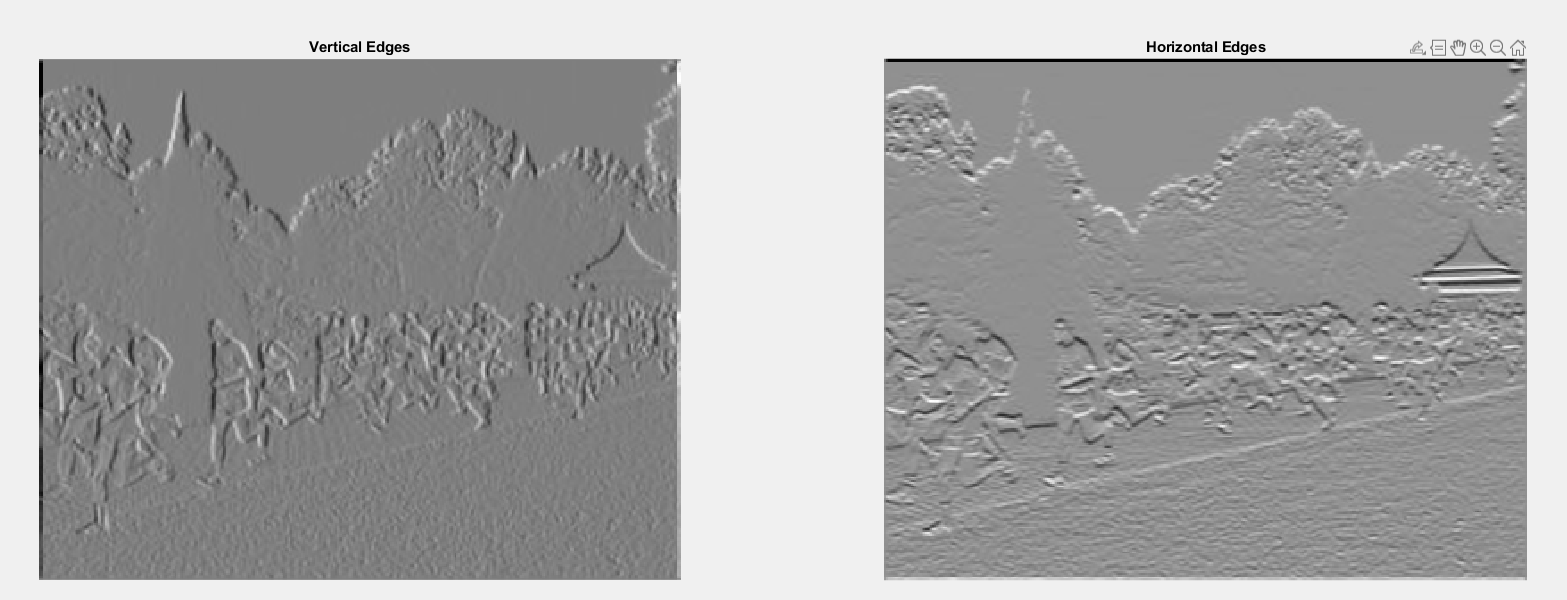
## Part b

Edges can either be a vertical or horizontal edge. For vertical edges, the gradient change is horizontal while the horizontal edges will have a vertical gradient change. A common approach to detect these edges is using either a Sobel mask or a Prewitt mask. The Sobel mask is slightly different from the Prewitt mask, the centre column of the Sobel mask has a higher weight compared to the Prewitt mask.

A picture containing white, light

Description automatically generated

To obtain vertical edges we need a vertical Sobel and a horizontal Sobel for horizontal edges. These masks are then convoluted with the original image to obtain the edges. The vertical and horizontal edges of the image are shown below



As expected the vertical Sobel can capture the vertical edges such as the trees and people. The horizontal Sobel can capture horizontal lines such as the lines in the building. Interestingly, lines that are diagonal such as the pathway is captured in both the edge map.

Square

Description automatically generatedIf we take a closer inspection of diagonal lines, only the diagonal pixels are 1 and the rest are 0s. This would mean that there is a vertical gradient change (shown in red) and a horizontal gradient change (shown in blue). As a result, the vertical Sobel detects the vertical change in gradient and the horizontal Sobel detects the horizontal change in gradient causing the diagonal line to appear in both edge maps. The same logic can be applied to non-linear lines therefore lines such as the roof of the building are captured in both edge maps.

Code for this section

Chart

Description automatically generated with low confidence

## Part c

As mentioned previously, the individual Sobel masks are only able to capture individual gradient changes but we are interested in the absolute gradient change of these edges, as a result, we have to square the individual gradients. The overall absolute gradient change of the lines is given by the formula

By right when we add the 2 respective absolute gradients together, we need to square root the addition but it is approximately the same as addition of the absolute gradient. Therefore we square the edge image and add them together.

The combined edge image is shown below, majority of the pixels are white since it contains all the horizontal, vertical and non-linear lines

A picture containing chart

Description automatically generated

## Part d

As shown in the previous part the combined edge map contains noisy edges and we are not able to get a clearer picture of the stronger lines. Some edges have a sharp gradient change while some edges have a small gradient change. Therefore to remove these noisy edges a simple threshold is introduced. Any edges that have an absolute gradient magnitude greater than the threshold will be selected and displayed. The output of the edge image with different thresholds is shown below

A picture containing text, screenshot

Description automatically generated

When we increase the threshold, the number of edges displayed decreases. The image shows the significant lines only when the threshold is greater than 10000. As the threshold increases, the noisy Ass the threshold increases edges are removed and the edges with a stark gradient change are selected. But some of these edges with a small gradient change depict the details of the image. One example is the diagonal line of the pathway, it disappeared when the threshold increased from 10000 to 100000 and some of the body parts are removed as well. In conclusion, having a lower threshold will enable us to pick the details of the image but it will also select edges that are irrelevant increasing the noise of the edge image. Having a larger threshold will enable us to select the lines that depict the trend of the image but we will lose out on the details. Therefore one should pick an optimal threshold that will remove most of the noise and keep some of the detail.

## Part e

Previously we discussed how a threshold can be used to filter out the noisy edges but we had to find the optimal threshold that can remove noise yet keep most of the details. Another edge detection method that can overcome this issue to a certain extent is the Canny edge detector. The canny edge detector makes use of a Gaussian filter to remove the noisy edges, a non-maximal suppression to deal with edges with varying thickness and Hysteresis thresholding to deal with edges that have varying strength.

The Gaussian edge filtering is filtered twice by the x and y derivatives of Gaussian. The distribution of the Gaussian will be vary based on the value of sigma used. The image below shows the effect of increasing sigma on the edge map.

A picture containing text, gallery, room

Description automatically generated

By increasing the sigma the number of lines in the edge map decreases. It removes the noisy edges and the details while keeping the trends. When the sigma increases from 1 to 5, the spread of the Gaussian distribution will increase therefore it keeps the trend and removes the details. By removing these details the location of these edgels are less accurate. Having a higher sigma will be suitable for the removal of noisy edgels but the accuracy of these edgels might decrease.

Other than the sigma value, we have to decide on a suitable upper and lower threshold for the hysteresis thresholding. In the previous section, we used 1 threshold to determine whether the edge should be selected or not. In the hysteresis threshold, any lines that have a magnitude that is greater than the upper threshold will be a 1 and anything lower than the lower threshold will be a 0. If the magnitude is in between the upper and lower threshold, we will check the neighbouring pixels that are perpendicular to the edge gradient. These pixels will follow the value of those neighbouring pixels. The edge image was tested with different lower thresholds and the output is shown below

Qr code

Description automatically generated

When we increase the lower threshold to a higher value, the number of edges shown is lesser. By increasing the lower threshold the space between the upper and lower threshold decreases, which will reduce the tiny noisy edges and find the long edges in the image. This change is very clear if we observe the edge that is within the red circle. Those edges are broken down into smaller long edges and the tiny edges that connect them are removed.

# Hough Transform

## Part a

Background pattern

Description automatically generatedIn the previous section, we experimented with different methods for edge detection. Some of these edges might belong to long edges and we are unable to identify these lines. Hough transform is a method that can be used to uncover a suitable line that connects these broken local edges. In this section, we will be exploring Hough transform on the following image

## Part b

Generally, a line can be represented using y=mx+c but in Hough transform we will represent a line using . The intuition behind this is that every vector on the line must be orthogonal to the straight line of length that comes from the origin. Therefore each line can be rewritten as rough and theta. For a given coordinate, there can be multiple pairs of values that can satisfy it. For each pixel (x,y), the Hough transform will generate new lines by going through the different rho and theta values. When these new lines pass through the accumulator’s bin, the value in that bin is incremented. Therefore the suitable rho and theta values are those that have the highest number of votes.

Chart, line chart

Description automatically generated

For radon transform, it is a projection of the image intensity along a radial line oriented at a specific angle. In MATLAB, the algorithm will split each pixel into 4 sub-pixel and project each of these subpixels separately. If the subpixel projection hits the centre point of a bin the bin will get the full value of the subpixel (which is ¼ of the pixel value). The detector line will be rotated for all possible values of theta, so if all the lines are collinear then these points will have the projection falling into the same bin (Beatty, 2012). As a result, the value of theta and t will have a higher intensity value. This method is very similar to the voting method that takes place in Hough Transform. Both the radon transform and Hough transform represent the lines using theta and t/which is the result of the point normal parameterization of a line.

Diagram

Description automatically generated

In Radon transform we consider how a data point in the destination space is obtained from the data in the source space but for Hough transform we consider how a data point in the source space maps onto the data points in the destination space (van Ginkel, Luengo Hendriks, & Van Vliet, 2004).Due to the similarities in the transformation, we can replace Hough transform with the radon transform but it is only applicable to binary images(Fernández, Flores, Alonso, & Ferrari, 2015).

Graphical user interface

Description automatically generatedThe result of the Radon Transform of the edge image in part a is shown below

## Part c

The line with the strongest edge support can be identified based on the pixel that has the maximum intensity. Since theta starts from 0 to 179 but the index in MATLAB starts from 1 we need to subtract 1 from the result.

The code to compute theta and rho:

Text

Description automatically generated with medium confidence

The theta obtained was 103 and the radius/rho obtained was -76.

## Part d

Diagram

Description automatically generatedAs mentioned in Part b, a line in Hough transform is represented using . So, comparing that equation with a standard form for linear equations (Ax+By = C), A is and B is and C is . These values can be obtained by the pol2cart function in MATLAB. The function converts the polar representation to a cartesian mapping so the vertical height will be and the horizontal height will be .

So, if we use pol2cart, we will get A as and B as which is slightly different to the Hough line equation. Therefore, we must multiply the Hough line equation with to get a similar equation result obtained from pol2cart. As a result, the new equation as . So now if we compare this equation with the standard line equation, we will get A as and B as and C as . Another thing to consider is the fact that Hough transform considers the origin as the centre of the image but the origin in MATLAB is the top left corner, so we need to translate the origin by positive 179 units in the x direction and positive 145 units in the y direction. Thus the new line equation will be . We want this line to be expressed as just so we have to move all the constants over and C is rho squared ( refer to the previous equation) resulting in the following equation . Therefore, the new value of C can be calculated by squaring rho and adding 179A and 145B.

The value obtained for A was 17.0963, B was 74.05 and C was 19573.792.

## Part e

The 2 values of y obtained were 264.324518403774 and 181.904574171948 respectively.

## Part f

The line was superimposed on the image to evaluate its accuracy. The line fits the pavement most of the time but there are times when the line deviates. These deviations are highlighted in red

****

These deviations could arise due to several errors. Some of these errors include conversion errors when we are converting from the parameter space to image space and another reason for the error could be due to the nonlinearity of the pavement. The latter half of the pavement requires a lesser theta than the current instance. But if we use a lesser theta the first half of the pavement deviates while the latter half is consistent with the pavement (the image shown on the left). One way to counter this is to take the average between this new line and the previous line obtained in part e. But even then there is some deviation (shown in the image on the right)

A group of people running

Description automatically generated with low confidence



Instead of taking equal weightage if we took 80% of the first line and 20% of the new line, we can get quite a close representation of the pavement.

A group of people running on a track

Description automatically generated with medium confidenceIf we wanted to make these steps automated or end-to-end what we could do is split the image into small segments and find the coordinates in each of these small patches. By doing so the pavement will be a representation of a piecewise function.

# 3D Stereo

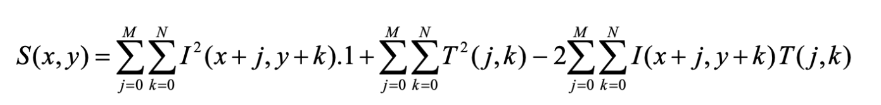
## Part a

In this section, we will be creating a function to compute the disparity map given 2 images and the template dimension. If we assume that the camera is only translated in the x direction, then we can fix the y and go through a sample of x to find the correspondence. In this case, we are taking the disparity to be within 15 pixels. Since y is fixed, the outer loop will be iterating through the possible values of y and the inner loop for possible values of x. Since the template is an 11 x 11 dimension the top corner pixel will not be used since the template cannot be placed at the centre of these pixels so we will take from index 6 to image height or width -5. During the inner iteration, the x pixels will be treated as the centre so we have to find the maximum index that we can traverse and the minimum index we can traverse. Once we have this range there will be another for loop to iterate through all these indexes in the right image to find the pixel that has the lowest SSD. The SSD can be calculated using

Text, letter

Description automatically generated

Notice in the calculation we only consider the first term and the last term of



We did not compute the middle term as it is a constant, so to decrease the computational power we can remove that term entirely from the calculation. Since the right template will be 11x11 the SSD calculated will also be 11 x 11 and we are only concerned with the value at the centre.

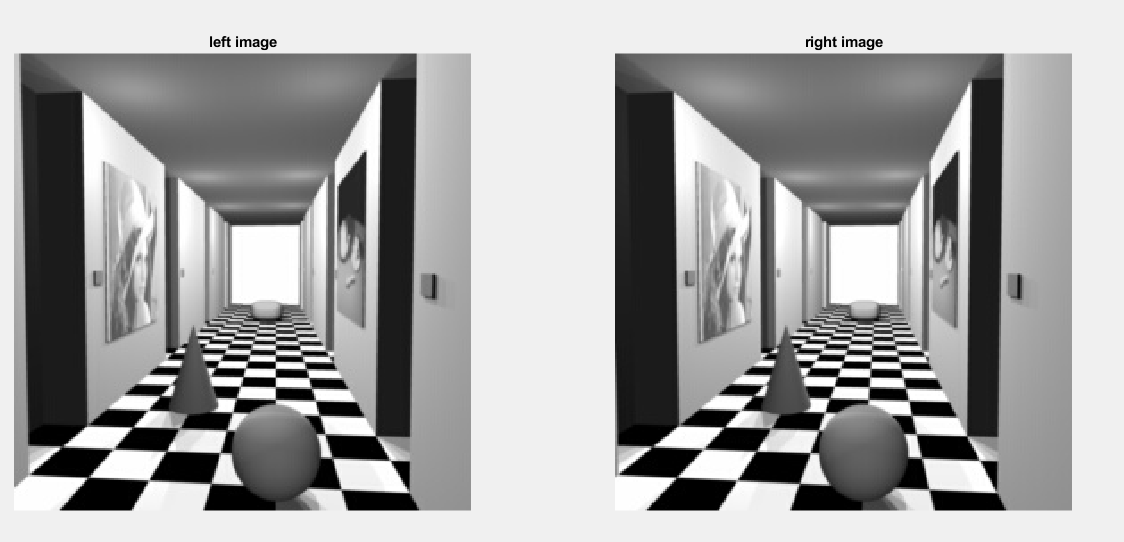
Therefore, the overall code for the function is as follows

Text

Description automatically generated

## Part b

We will be calculating the disparity map between these 2 images



## Part c

The calculated disparity map is shown below

A picture containing text, indoor, jack, electronics

Description automatically generated

Most of the calculated disparities are similar except a few regions. One of the region that differ from the expected disparity is the white wall located at the end of the aisle. The white wall is a homogenous area and for homogenous areas it is very difficult to find the exact correspondence since there can be many pixels with the similar SSD so it might take the first pixel that has the lowest SSD and the actual correspondence might be further to the right (since our function computes correspondence from the left to the right). Another error could be due the restriction in disparity. We are only searching within a radius of 15 pixel, but the correspondence might fall outside this scanline radius.

## Part d

In this section we will be testing the disparity map function with a different set of imagesA picture containing text, building, outdoor, apartment building

Description automatically generated

The computed disparity map is compared with the expected disparity map

A picture containing bathroom, indoor, sink, appliance

Description automatically generated

The calculated disparity map looks similar to the expected disparity map. But these disparity maps are different to the ones we see in part c. We would expect a disparity map that is smooth and increase as the distance increases. But if we look at the sidewalk there is a lot of fluctuation in the disparity values. This reflects poorly on the accuracy of the disparity map. If we look at the left and right image side by side, it does not seem like there is only a shift in the x axis. It appears to be a shift in the camera angle itself therefore we need to find the rectification homography to project these image plane so that they can be parallel to one another and then we can use the current function to find the disparity map since the correspondence will most likely fall onto the same scanline.

Another way is to use Feature based mapping instead of Appearance based matching. The current function is using appearance-based matching so if there are pixels that have similar appearance it might detect the correspondence wrongly resulting in an incorrect disparity map.

# Extra Section

In the lecture we have discussed Bag-Of-Words method for object detection and classification. The method involves in obtaining the feature points then converting these feature points to feature vectors. Then performing clustering on these feature vectors to reduce the number of visual words and create a histogram based on these visual words. But the limitation of this method is that it does not capture the spatial information of these features, therefore this section we will be exploring the usage of Spatial Pyramid Matching for object classification based on the Caltech-101 dataset.

We will experiment the difference between BOW and Spatial pyramid using 2 different classifiers: a linear SVM and a one vs rest SVM. The accuracy of these classifiers from the 2 methods are shown below

Linear SVM

|  |  |  |
| --- | --- | --- |
| L | Bag of Words | Spatial Pyramid |
| 0 | 0.35459183673469385 | 0.35459183673469385 |
| 1 | 0.3753644314868805 | 0.40561224489795916 |
| 2 | 0.358600583090379 | 0.42674927113702626 |

One vs Rest SVM

|  |  |  |
| --- | --- | --- |
| L | Bag of Words | Spatial Pyramid |
| 0 | 0.3432944606413994 | 0.3432944606413994 |
| 1 | 0.41180758017492713 | 0.44533527696793 |
| 2 | 0.4260204081632653 | 0.4774052478134111 |

For both classifiers when the L increases the accuracy increases. The one vs Rest SVM is more suitable than the linear SVM and Spatial pyramid always outperforms the bag of words method except when L = 0 then they have the same accuracy.

# Conclusion

In the experiment, we have discussed the different edge detection algorithm, similarities between Radon and Hough transformations, explored 3D stereo and went on to experiment with alternatives to BOG.

# References

Beatty, J. (2012). *The Radon Transform and the Mathematics of Medical Imaging*: Digital Commons @ Colby.

Fernández, A., Flores, J. L., Alonso, J. R., & Ferrari, J. A. (2015). Real-time pattern recognition using an optical generalized Hough transform. *Applied Optics, 54*(36), 10586-10591. doi:10.1364/AO.54.010586

van Ginkel, M., Luengo Hendriks, C., & Van Vliet, L. (2004). *A short introduction to the Radon and Hough transforms and how they relate to each other*.