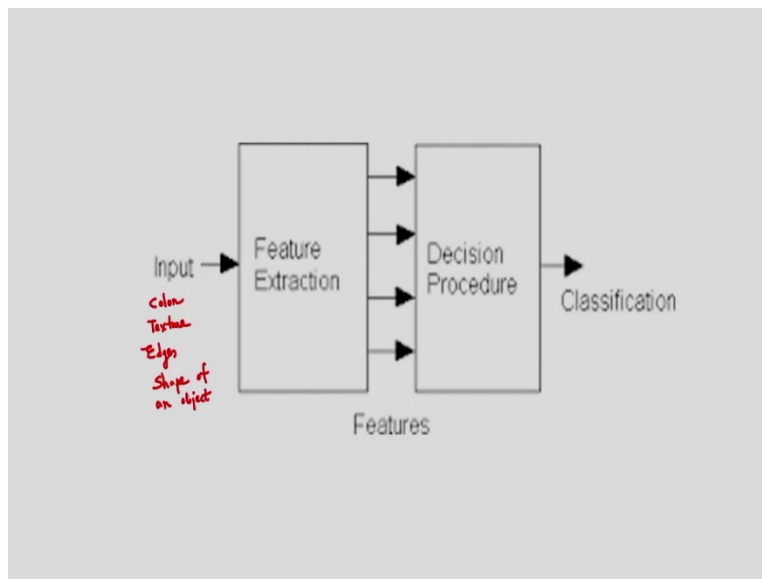


Computer Vision and Image Processing - Fundamentals and Applications
Professor Doctor MK Bhuyan
Department of Electronics and Electrical Engineering
Indian Institute of Technology, Guwahati
Lecture 21
Image Features and Edge Detection

Welcome to NPTEL MOOCs course on Computer Vision and Image Processing - Fundamentals and Applications. Up till now I have discussed about some image processing concepts, so now I will discuss about image features. I can give some examples of image features; colour, textures, edges or the boundaries are the examples of image features. So, how to extract these features, so I am going to discuss and these features can be extracted in spatial domain or in frequency domain. So, today first I will discuss about colour features and after this I will discuss the concept of edge detection. So, let us see what are the colour features.

(Refer Slide Time: 1:16)



So, here you can see, I have shown the input image and I have shown a block that it is a feature extraction, so I can extract features like may be the colour is a feature or may be the texture I can consider or may be the edges or the boundaries I can consider or may be the shape of an object I can consider as image features. And after extracting image features, I can go for decision making that is pattern classification. So, I can classify different images based on these features. So, first let us discuss about the colour features.

(Refer Slide Time: 2:03)

Colour Features

□ **Colour histogram:** It defines the image colour distribution. Colour histogram characterizes the global distribution of colours in an image.

$$d = \sum_{j=1}^B |h_j^{(M)} - h_j^{(N)}|$$

In this, $\underline{H}^{(M)} = \{h_1^{(M)}, h_2^{(M)}, \dots, h_K^{(M)}\}$ and $\underline{H}^{(N)} = \{h_1^{(N)}, h_2^{(N)}, \dots, h_K^{(N)}\}$ are two feature vectors extracted from the colour histograms of two images \underline{M} and \underline{N} , where $h_j^{(M)}$ and $h_j^{(N)}$ are the count of pixels in the j^{th} bin of the two histograms, respectively. In this, B is the number of bins in each histogram.

So, the first feature is the colour histogram, so in my lecture of image enhancement, I discussed about how to determine histogram of an image, Similarly, I can determine the colour histogram of an image, so it defines the image colour distribution and mainly the colour histogram characterize the global distribution in an image. I can compare two images by histograms. So, in this case I can consider the colour models like HSI colour model, Y CbCr colour model or maybe the lab colour models I can use. Because they give better results as compared to the RGB colour space.

So, this colour histogram characterizes the global distribution of colours in an image. And already I told you that it can be used as an image feature. So, for colour histogram that means to extract colour histogram from an image, and the color space can be quantize into a finite number of discrete levels and after this, each of these quantization levels is been in the histogram. So, that concept already we know, how to calculate, how to determine histogram of an image and after this we can compare two images based on this colour histogram.

So, here you can see I am doing the comparison between the two images by considering colour histogram. So, I am considering two images, one image is M and another image is N. And I am defining the histogram for the first image and the histogram for the second image I am defining like this. Then in this case, I can find the dissimilarity or maybe the similarity between two

histograms and based on this I can discriminate or maybe I can compare two images. So, this is about the colour histogram.

(Refer Slide Time: 4:10)

❑ **Colour correlogram:** One problem with the colour histogram is that the global colour distribution does not indicate the spatial distribution of the colour pixels. A **colour correlogram** expresses how the spatial correlation of pairs of colours changes with distance. Correlogram is a variant of histogram that accounts for the local spatial correlation of colours.

❑ **Colour moments:** It is a compact representation of colour features of an image.

$$\mu_c = \frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N p_{ij}^c$$

$$\sigma_c = \sqrt{\left[\frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N (p_{ij}^c - \mu_c)^2 \right]}$$

$$\theta_c = \sqrt{\left[\frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N (p_{ij}^c - \mu_c)^3 \right]}$$

The another point is; the another feature is the colour correlogram. In case of the colour histogram, the spatial information is not available, so that is why we consider this feature, that feature is the colour correlogram. The spatial correlation of pairs of colours that changes with distance, that I am considering for this feature. In case of the colour histogram, that information is not available, that is the spatial information is not available.

So, in colour correlogram we consider that information, the information is the spatial correlation of pairs of colours that changes with distance and in this case it is mainly the histogram but we consider the spatial information. The another colour feature is the colour moments, so in the colour moments, I can determine the first order moment, so you can see I can determine the first order moment. The second order moment also I can consider, that is nothing but the standard deviation I can determine. And also I can determine the third order moment that is nothing but skewness of a colour.

So, in this case what is this P_{ij}^c , so this is the value of the C_{th} colour component of the colour pixel in the i_{th} row and j_{th} column of an image. So, P_{ij}^c it is represented like this, that is the value of the C_{th} colour component of the colour pixel in the i_{th} row and j_{th} column of the image. So, this

colour moments, I can extract, the first order moment, the second moment, the third order moment and that I can determine. So, this is nothing but the skewness, skewness of a colour.

(Refer Slide Time: 6:32)

- ❑ **The MPEG-7 colour descriptors** comprise of **histogram descriptors**, a **dominant colour descriptor**, and a **colour layout descriptor (CLD)**.
 - ❑ For dominant colour descriptor, a set of dominant colours in an image or a region of interest is selected.
 - ❑ For colour space descriptor, a short description of the most widely used colour spaces/models is defined. $\rightarrow YCbCr$
 - ❑ The CLD is a compact MPEG-7 visual descriptor which is designed to represent the spatial distribution of colour in the Y CbCr colour space. $8 \times 8 = 64 \text{ blocks}$
- ❑ **Scalable colour descriptor (SCD)** is a Harr-transform-based encoding scheme that measures colour distribution over an image. The HSI colour space is used and it is quantized uniformly to 256 bins. The histograms are encoded using a Harr transform, which allows desired scalability.

The another, the descriptor, the colour descriptor is the MPEG-7, the MPEG-7 colour descriptor, the Moving Picture Experts Group – 7 the colour descriptors. So, it mainly comprises of the histogram descriptors, a dominant colour descriptor and a Colour Layout Descriptor. This is about the colour descriptors, the MPEG-7 colour descriptors. So, for dominant colour descriptors, a set of dominant colours in an image or a particular region of interest is selected. And in this case for colour descriptors, a short description of the most widely used colour space a model is defined.

Generally, we in this case we consider the Y CbCr model is considered. So, the CLD that is CLD means the Colour Layout Descriptor, a CLD is a compact MPEG-7 visual descriptor and in this case it is designed to represent the spatial distribution of the colour in the Y CbCr colour space, that we consider this the CLD. For this what we have to do, the given image or a region of interest of an image is divided into 8 by 8, that is 8 by 8 means, 8 by 8, into 8 by 8 means the 64 blocks. So, that means the given image or a region of interest of an image is divided into 64 blocks.

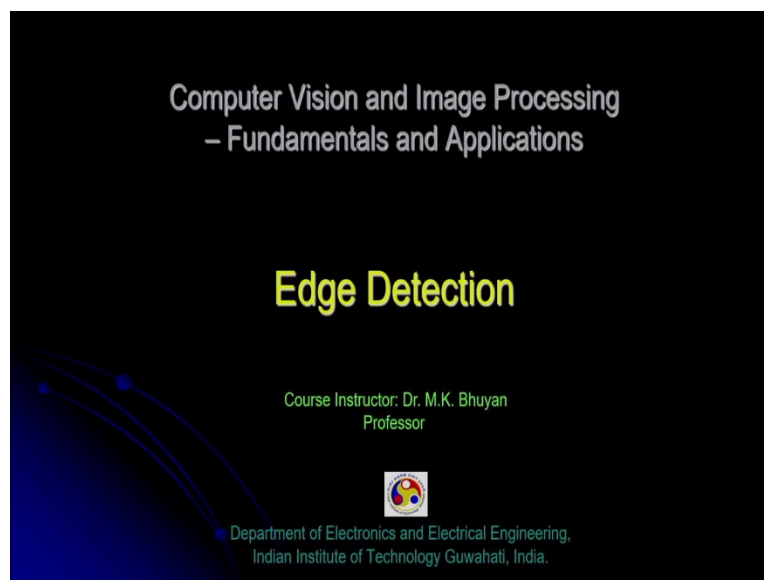
And the average colour of each block is calculated as its representative colour. And finally what we can do, finally the discrete cosine transform is perform in the series of the average colours

and I can consider few low frequency coefficients, that means we can select few low frequency coefficients are selected. The CLD is from after quantization of the remaining coefficients, this is about the Colour Layout Descriptor. So, I am not explaining in details, so if you want to read in details you can see the books or maybe the papers, the research papers for MPEG-7 colour descriptors.

Another one is scalable colour descriptors. So, it is mainly the Harr-transform-based encoding scheme and that is used to measure colour distribution over an image. So, for this the HSI colour space is used and it is quantized uniformly to 256 bins. And finally the histograms are encoded using a Harr transformation. So, if I use the Harr transformation, that means we have to, we can consider the scalability. A scalability is important.

So, again I am not explaining in details, but for understanding of this concept, you can see the papers, the research papers and the books. One is the MPEG-7 colour descriptors, another one is scalable color descriptors. So, this is about the colour features, so we have considered colour histogram, colour correlogram, colour moments and regarding the descriptors we have considered MPEG-7 colour descriptors and the scalable colour descriptors.

(Refer Slide Time: 10:01)



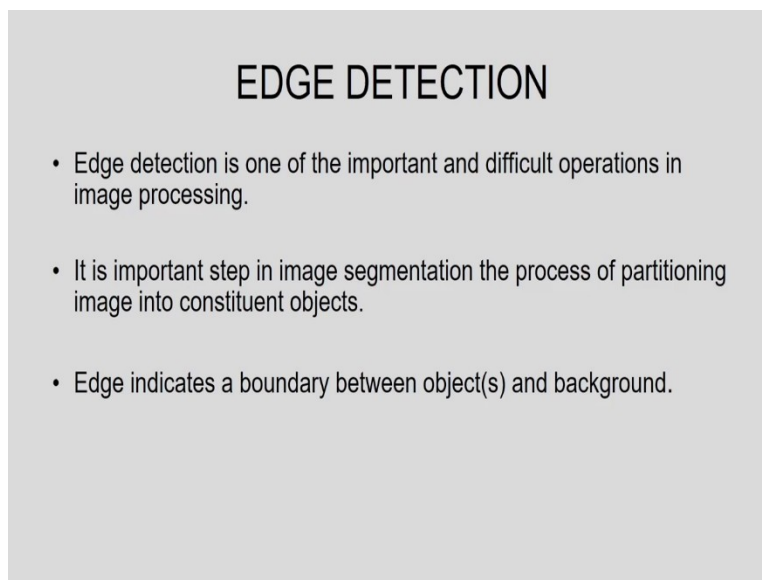
The next topic is the concept of the edge detection. So, edge detection is a very important image processing step that how to detect the edges and the boundaries. But it is a very difficult image processing task because mainly it depends on many cases like I can give one example, in

different illumination conditions, the number of edges in an image will be different. Suppose, if I consider a low light condition and if I consider a very good light condition, the number of edges, the number of edge pixels will be different in both the cases.

So, that is why the edge detection is a very difficult task in image processing. Another point is, suppose for image segmentation, the edge detection is quite important. Image segmentation means the partitioning of an image into connected homogeneous region. So, I can determine the edges, I can determine the boundaries and based on this I can do the image segmentation. And mainly suppose if I want to consider the separation of the foreground and the background, so based on the edge detection principles, I can do this.

I can determine the edges and I can determine the boundaries and after this I can do the segmentation. So, edge detection is a very important image processing step and edges I can consider as a feature. Now, let us see how to determine the edges, how to detect the edges. So, before explaining this concept, I can show the concept of the edges and the boundaries.

(Refer Slide Time: 11:40)

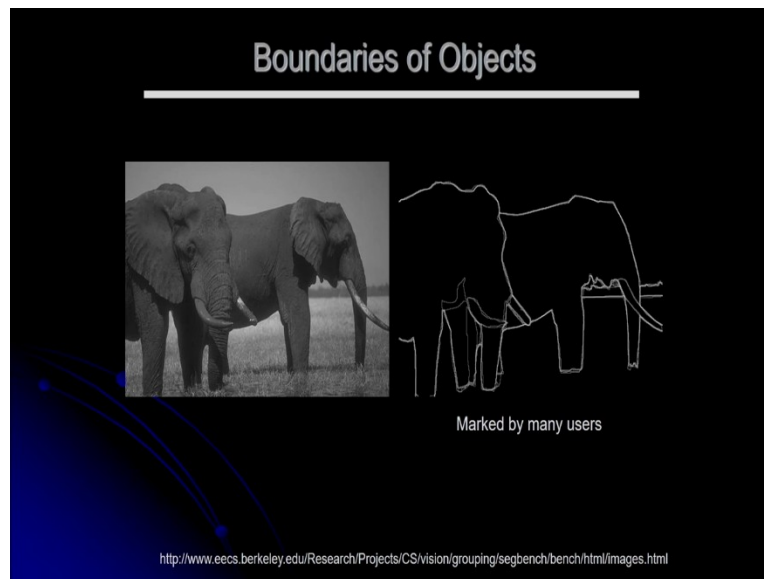


EDGE DETECTION

- Edge detection is one of the important and difficult operations in image processing.
- It is important step in image segmentation the process of partitioning image into constituent objects.
- Edge indicates a boundary between object(s) and background.

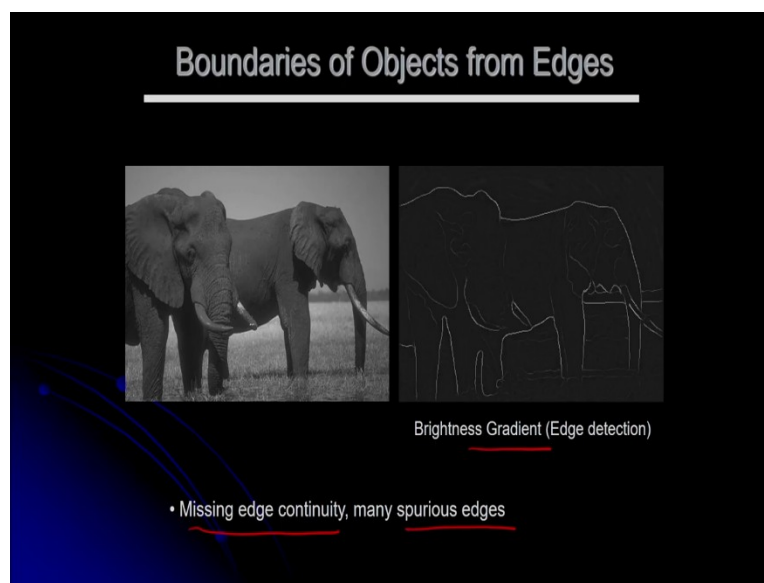
So, here you can see, the edge detection is one of the important and difficult operations in image processing. And in this case already I have explained this one that is for image segmentation is nothing but the partitioning of the image into connected homogeneous region. So, for this also the edge detection is quite important. An edge indicates a boundary between objects and background.

(Refer Slide Time: 12:10)



So, you can see here in this example, the boundaries of objects that is marked by many users, you can see. So, it is very difficult to find the accurate boundary.

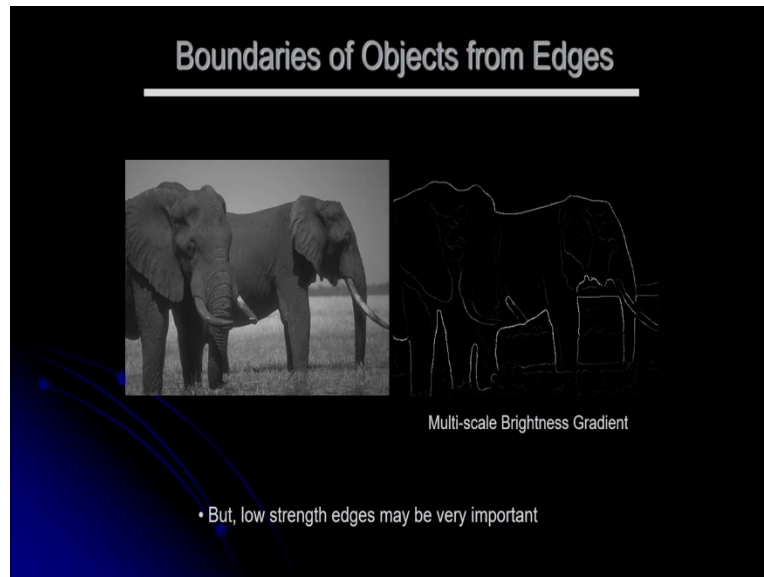
(Refer Slide Time: 12:22)



The next example I can show you, the boundaries of objects from edges. So, in this case, I can determine edges of an image by using some gradient principle, that is the brightness gradient I can determine. This concept I am going to explain after some time, so based on the gradient information, I can determine the edges and from this I can determine the boundary. But the main problem is missing edge continuity, that is one problem and many unwanted edges I am having

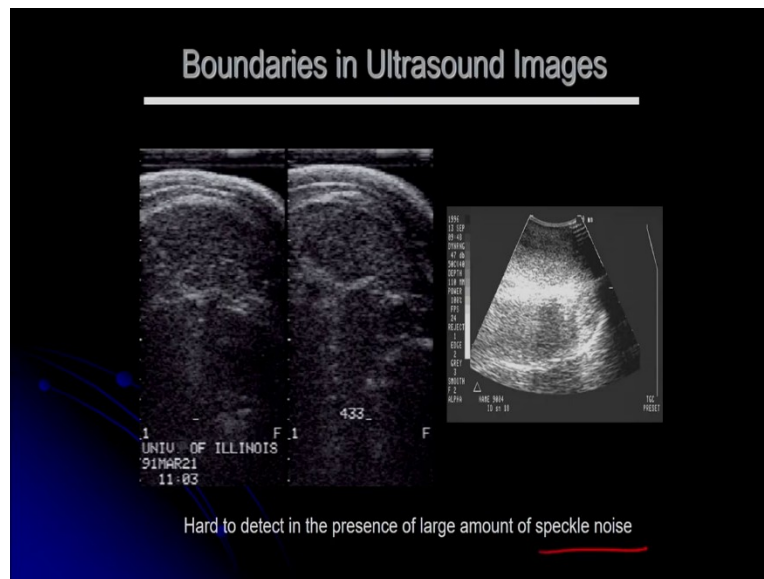
because of this, the selection of the gradient, how to select the gradient and what are the thresholds, because in this gradient methods I have to select the threshold. So, based on this I may get many unwanted edges, the edge pixels.

(Refer Slide Time: 13:14)



And here I am showing one example that is the boundary of the objects from edges, then in this case I am using the multi-scale brightness gradient, that I am going to discuss, but in this case, you can see low strength edges may be very important. So, in this example I have shown the boundaries of the objects from the edges and for this I am considering the multi-scale brightness gradient. But in this case you can see minor edges or may be the low magnitude edges are missing, because of the selection of the threshold. So, that may be also important, that is the low magnitude edges, small-small edges that are missing in this output.

(Refer Slide Time: 14:02)



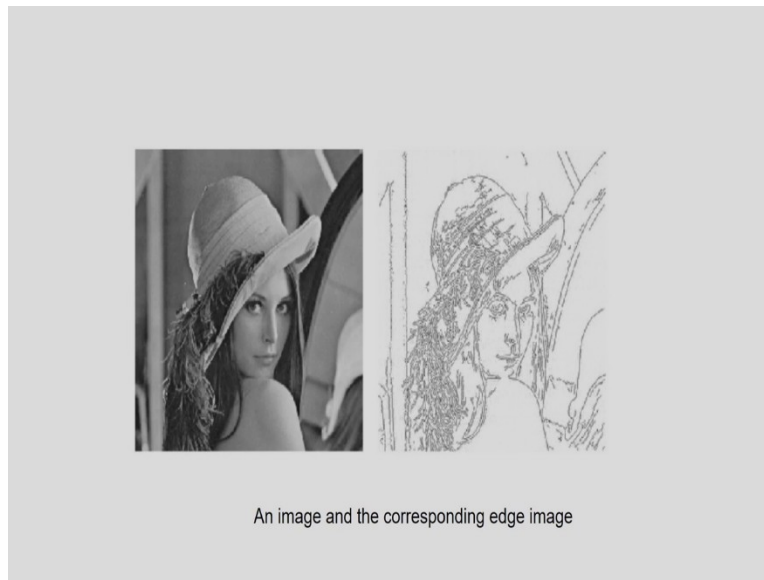
And sometimes it is very difficult to determine the boundaries. So, here you can see, I am considering one ultra sound image and corresponding to this ultra sound image you can see it is effected by speckle noise. It is effected by speckle noise, in this case it is very difficult to find the edges and the boundaries.

(Refer Slide Time: 14:21)



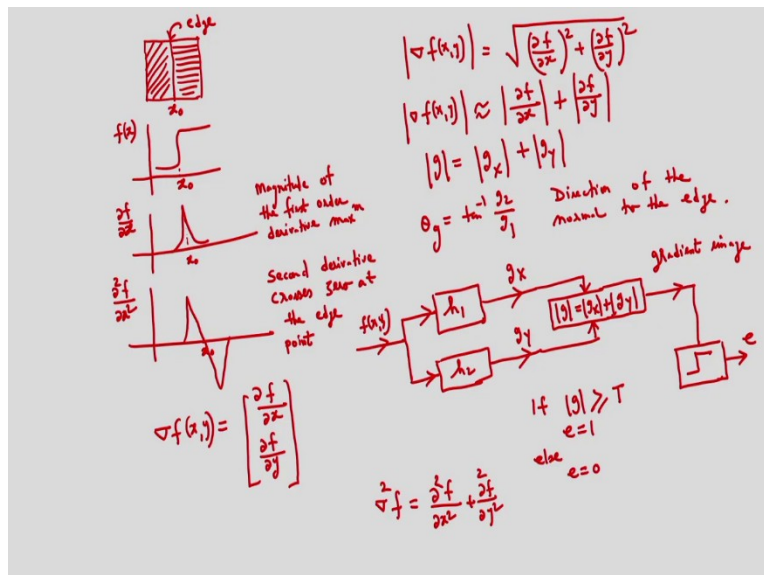
And in this case for these images, you can see this type of images, it is really very difficult to find the edges and the boundaries, even for humans. Now, I will discuss the concept of the edge detection, so how to determine the edges in an image. So, let us see the principle.

(Refer Slide Time: 14:42)



And here in this example I can show you, corresponding to this input image, you can see the edges of the image. So, for this I am applying the gradient information, so how to determine the gradient information and based on this gradient information how to determine the location of the edge pixels that concept I am going to explain now.

(Refer Slide Time: 15:08)



Suppose I am considering one image, so this side is one side and that is another grayscale value. So, this is the edge, edge means the abrupt change of intensity value. So, you can see the location of the edge corresponding to the point x naught. So, if I draw the 1D profile of this one, the 1D

profiles will be something like this, so corresponding to the location x naught and I am considering the 1D profile, so it is f_x suppose, so in the left side of the image you can see one intensity the brightness, in the left side of the image you can see one the grayscale value and after the, if you see the right side this is one grayscale value.

So, edge means abrupt change of the intensity value. So, if you see this 1D profile, so how to determine the location of the edge? So, if I take the differentiation of this, the first order differentiation, that means I am determining the gradient, so that means corresponding to x naught this point, you will be getting the maximum corresponding to this point x naught. If I consider a second order derivative, so corresponding to this x naught, you will be getting a zero crossing.

So, you will be getting a zero crossing, so that means, so how to determine the location of the edge, the principle is you can see the magnitude of the first derivative, the first order derivative, is maximum. So, based on this information the magnitude of the first order derivative is maximum, based on this information you can determine the location of the edge pixel. The next one is, if I consider a second order derivative, the second derivative crosses 0 at the edge point.

So, you can see by determining the first order derivative, you can determine the location of the edge pixel, the location of the edge and also by determining the second order derivative, you can determine the location of the edge pixel. In case of the first order derivative, we have to see the magnitude, because maximum magnitude I have to see and for the second order derivative, I have to see the zero crossing. So, that means by using, by determining the gradient, we can determine the location of the edge pixel. So, that means I have to determine the gradient, the gradient of the image.

So, suppose the image is $f(x, y)$ and I have to determine the gradient, so one is gradient along the x direction and another one is the gradient along the y direction I can determine. And from this I can determine the gradient magnitude. So, this is the gradient magnitude and approximately it is gradient magnitude will be, so this is the approximate value of the gradient magnitude. So, that means the gradient magnitude is equals to gradient along the x direction plus gradient along the y direction. That we can determine.

And also, I can determine one angle, the angle is suppose, θ_g that is $\tan^{-1} g_2$ divided by g_1 , that I can determine. That is nothing but the direction of the normal to the edge, that I can determine. A direction of the normal to the edge, I can determine. So, you can see from the gradient I can determine the gradient magnitude, so for this I have to determine the gradient along the x direction, gradient along the y direction and also I can determine the angle, angle is nothing but the direction of the normal to the edge.

So, how to determine the edge? So, suppose I have image $f(x, y)$ and suppose I am considering one operator, operator is suppose h_1 and another operator I am considering, operator is h_2 . This h_1 operator I am considering for determining the gradient along the x direction. So by using h_1 , I can determine the gradient along the x direction. And by using h_2 , I am determining the gradient along the y direction. And after this, from this two information the gradient along the x direction and the gradient along the y direction, I can determine gradient magnitude.

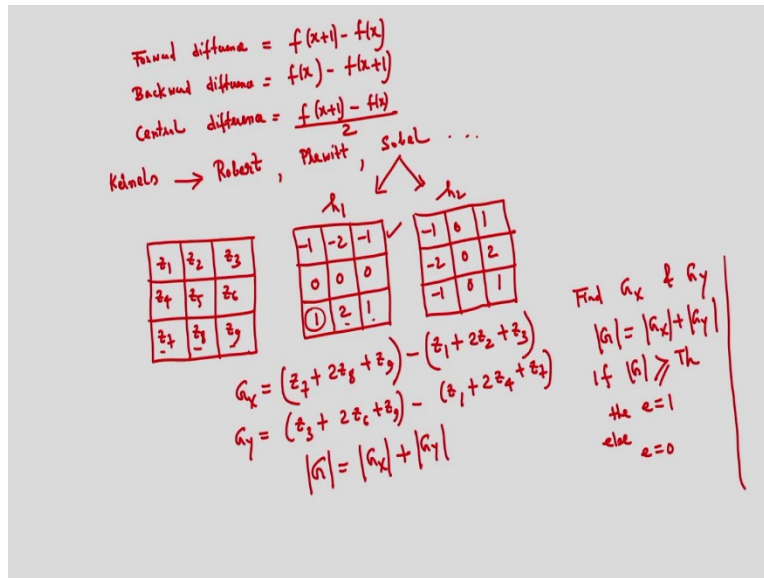
The gradient magnitude I can determine. And after determining the gradient magnitude, what I can consider, I am getting the gradient image, I will be getting gradient image I will be getting. And I can compare this one with a threshold and after thresholding we can determine the edge pixel. If the gradient magnitude is greater than a particular threshold, then that will be the edge, otherwise it is not the edge pixel. That means if the gradient magnitude is greater than equal to particular threshold, then edge will be 1, else edge will be 0.

So, that means after edge detection, I will be getting a binary image. Edge means 1 and no edge means 0, e is equals to 1 and e is equals to 0. How to select the threshold? I can give one example how to select the threshold. So, there are different methods but I can give one example like this, suppose the histogram of the gradient image has a valley, so we can select the threshold at this valley, so based on this we can select the threshold.

And in the case the procedure you can see, so first I have to determine the gradient along the x direction, gradient along the y direction. And after this, I can determine the gradient magnitude and after this the gradient magnitude is compared with a threshold and after this based on this we can determine whether particular pixel is the edge pixel or not the edge pixel. This is about the gradient based method, that is the first order gradient. A second order derivative I can write like this, that is nothing but a Laplacian, the second order derivative I can determine like this, this is the Laplacian.

And in this case, I have to see zero crossing to determine the location of the edge pixel. In case of the gradient magnitude, I have to see the maximum value, but in case of the second order derivative I have to see the 0 crossing. But one problem is here because of the second derivative, the noise will increase and that concept I can show you later on. This differentiation can be implemented by finite difference operations.

(Refer Slide Time: 24:50)



So, this what are the finite difference operations, so I can give one example, that is the forward difference. What is forward difference? The forward difference is $f(x + 1) - f(x)$, so this is one finite difference operation. The backward difference I can also calculate, that is $f(x) - f(x + 1)$, that is the backward difference also I can determine. I can also determine central difference, that

is $\frac{f(x+1) - f(x)}{2}$, that is the central difference. So, these are the finite difference operations.

So, by using these operations I can determine gradient, a gradient along the x direction, gradient along the y direction. And for this implementation, I can consider some kernels, some common kernels I can consider or maybe the mask I can consider, are something like the Roberts, another is the Prewitt. Roberts, Prewitt, Sobel, so these are some common operators, the common kernels. So, by using these kernels we can determine the gradient of an image. So, I can give one example how to determine the gradient of an image.

Suppose, I am considering one image, the pixels I am considering suppose $z_1, z_2, z_3, z_4, z_5, z_6, z_7, z_8, z_9$. So, these are the pixels and for determining gradient image, I am considering two mask, one is for the horizontal gradient, another one is for a vertical gradient. So, the first mask I am considering that is suppose h_1 , so the mask is suppose $-1, -2, -1, 0, 0, 0, 1, 2, 1$ and another mask I am considering that is for determining the vertical gradient, I can consider. So, these two masks I am considering, actually these are Sobel mask. So, one is for the horizontal gradient, another one is for the vertical gradient.

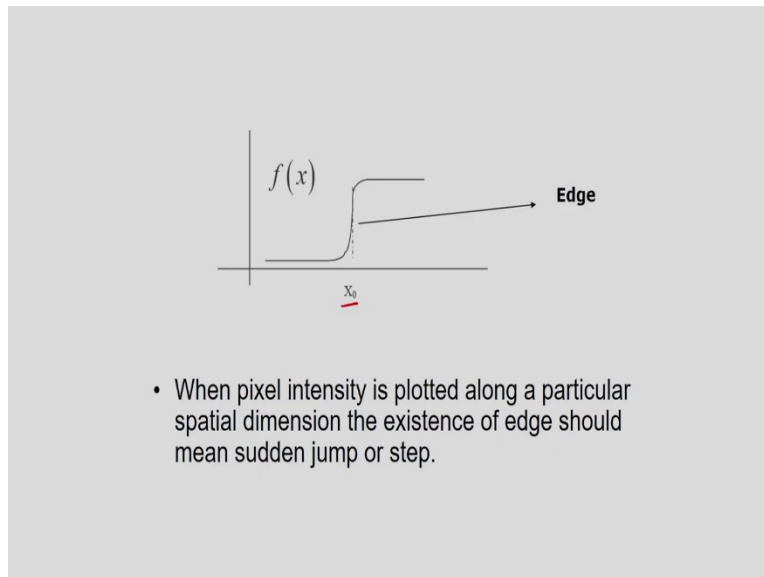
So, by using this mask I can determine the horizontal gradient and the vertical gradient. So, if I first consider this mask, the h_1 , I can determine the horizontal gradient G_x I can determine, so I have to overlap the mask over the image and I have to find the values, the gradient values. So, gradient is $z_7 + 2z_8 + z_9 - z_1 + 2z_2 + z_3$. This is the horizontal gradient, so you can see you can determine the gradient. So, here you see 1 is multiplied with z_7 , 2 is multiplied with z_8 , 1 is multiplied with z_9 , minus and after this I have to consider these values, so z_1 is multiplied with minus 1, z_2 is multiplied with -2 and z_3 is multiplied with -1.

So, like this I can determine z_x , a gradient along the x direction and similarly I can determine the gradient along the y direction also I can determine. $z_3 + 2z_6 + z_9 - z_1 + 2z_4 + z_7$. So, you can see, I can determine the gradient along the x direction and the gradient along the y direction. And finally, you can determine, the gradient magnitude also you can determine, the gradient magnitude is suppose z , if the gradient magnitude is z then it is mainly the $z_x + z_y$. The gradient along the x direction, gradient along the y direction I can determine.

And based on this gradient, what I can do? I can determine the location of the edge pixel, that means what is the algorithm, find, so algorithm will be like this, so find G_x and G_y , that is the gradient along the x direction and the gradient along the y direction. And after this I can calculate the gradient magnitude. I can determine the gradient magnitude and if the gradient magnitude is greater than a particular threshold, threshold is this, greater than equal to particular threshold, then edge will be 1, else edge will be 0.

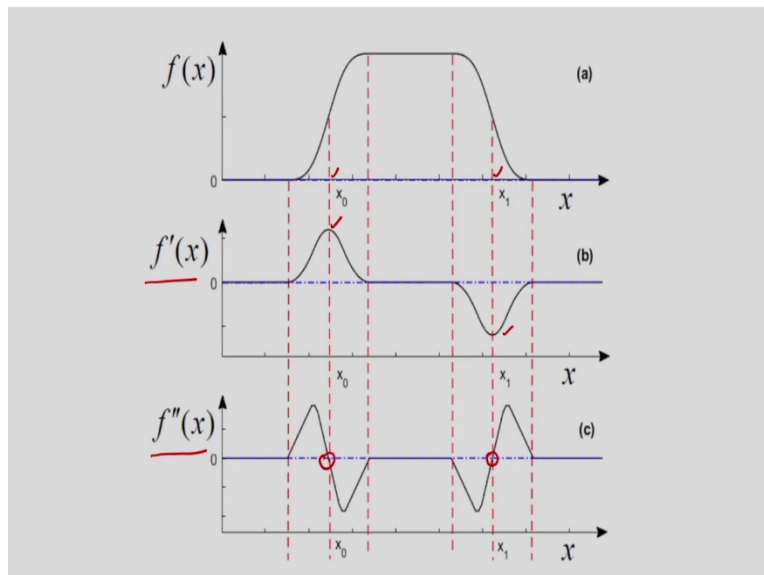
So, by using this algorithm I can determine the edge. So, that means, so by using these two masks I can determine the gradient along the x direction and the gradient along the y direction and after this determine the gradient magnitude and after this I can determine the edge after the comparison with a threshold.

(Refer Slide Time: 31:37)



So, this concept I have shown here, the what I have explained already, so it is the 1D profile and here I have shown the edge, the location here I have shown corresponding with the point x_0 . So, that means corresponding to the edge point, abrupt jump or the step, that is the 1D profile I am considering.

(Refer Slide Time: 32:02)

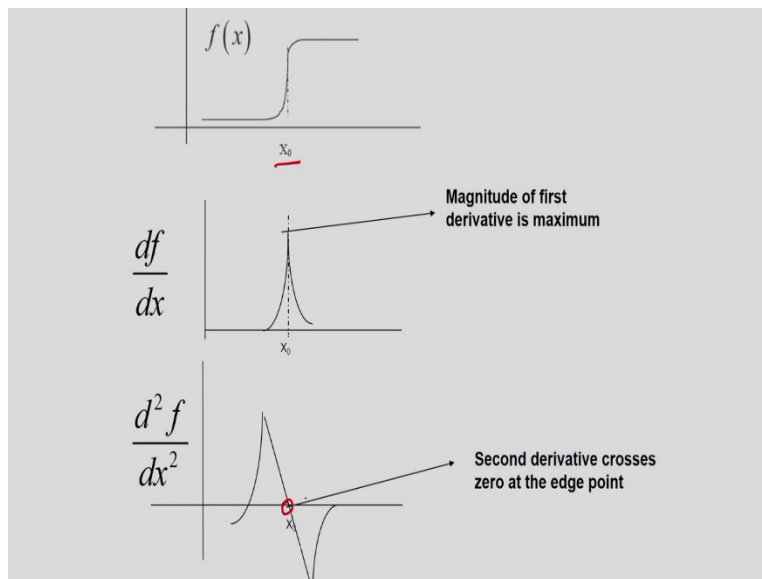


You can see here already this concept I have explained that $f(x)$ is the 1D profile of this and you can see location of the edge pixels corresponding to x_0 and corresponding to x_1 . And in this case,

I can determine the first order derivative, so it is $f'(x)$ I can determine and you can see the maximum value corresponding to the edge pixels, the location of the edge pixels.

And also, I can determine the second order derivative, that is $f''(x)$ I can determine and you can see the zero crossing you can see here corresponding to the location of the edge pixels, so that is x_0 and x_1 . So, that means by determining the first order derivative and the second order derivative, I can determine the location of the edge pixels.

(Refer Slide Time: 32:55)



A same thing I am showing here again, I am showing the 1D profile corresponding to the location of the edge at the point x_0 . And if I take the first order derivative, you can see the maximum value will give the location of the edge pixels. And if I consider the second order derivative, then in this case I have to see the zero crossing. For the first order derivative, I have to see the maximum value.

(Refer Slide Time: 33:24)

- All edge detection methods are based on the above two principles.
- In two dimensional spatial coordinates the intensity function is a two dimensional surface. We have to consider the maximum of the magnitude of the gradient.

$$\nabla f(x, y) = \begin{bmatrix} \frac{\delta f}{\delta x} \\ \frac{\delta f}{\delta y} \end{bmatrix}$$

And after this I have explained, so how to calculate the gradient along the x direction and the gradient along the y direction. So, here you can see the $\frac{\delta f}{\delta x}$ that is the gradient along the x direction. And $\frac{\delta f}{\delta y}$ that is the gradient along the y direction. So, this is the gradient operation I am considering, the gradient of an image I am determining.

(Refer Slide Time: 33:48)

- The gradient magnitude gives the edge location.

$$|\nabla f| = \sqrt{\left(\frac{\delta f}{\delta x}\right)^2 + \left(\frac{\delta f}{\delta y}\right)^2}$$

- For simplicity of implementation, the gradient magnitude is approximated by

$$|\nabla f| = \left| \frac{\delta f}{\delta x} \right| + \left| \frac{\delta f}{\delta y} \right|$$

- The direction of the normal to the edge is obtained from

$$\tan^{-1} \left(\frac{\frac{\delta f}{\delta y}}{\frac{\delta f}{\delta x}} \right)$$

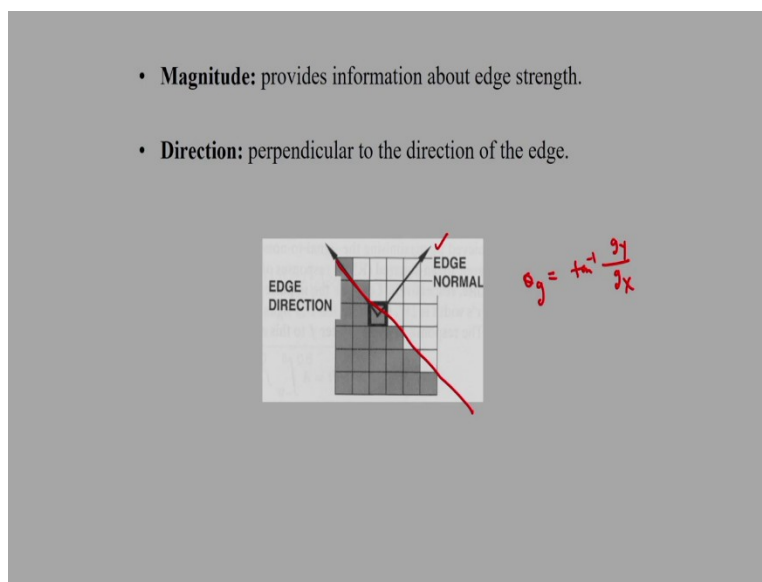
- Second derivative is implemented as a Laplacian given by

$$\frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} = \nabla^2 f$$

And after this I can determine, the gradient magnitude I can determine, so gradient along the x direction square plus the gradient along the y direction square, so from this I can determine the gradient magnitude. The approximately I can represent like this, the gradient magnitude, that is the gradient magnitude is equal to gradient along the x direction + gradient along the y direction, that is the approximately I can represent like this.

And also, I can determine the direction of the edge normal, direction of the normal to the edge, that also I can determine. So, this tan inverse g_y divided by g_x , that also I can determine. And already I have defined the Laplacian, the Laplacian is nothing but the second order derivative, so based on the second order derivative, I have to see the zero crossing.

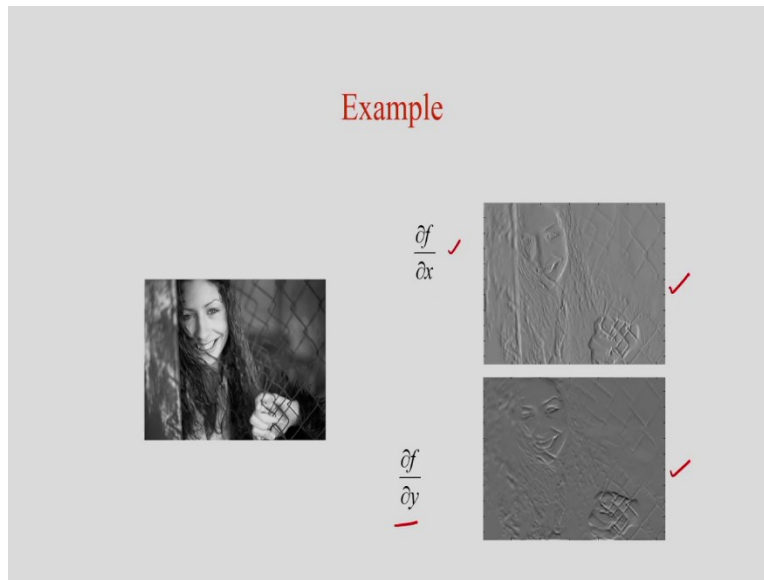
(Refer Slide Time: 34:41)



Here I have shown the concept of the edge normal, here you can see I have the edge, suppose this is the edge and this is the edge normal, so this angle already I have determined, that is the θ_g

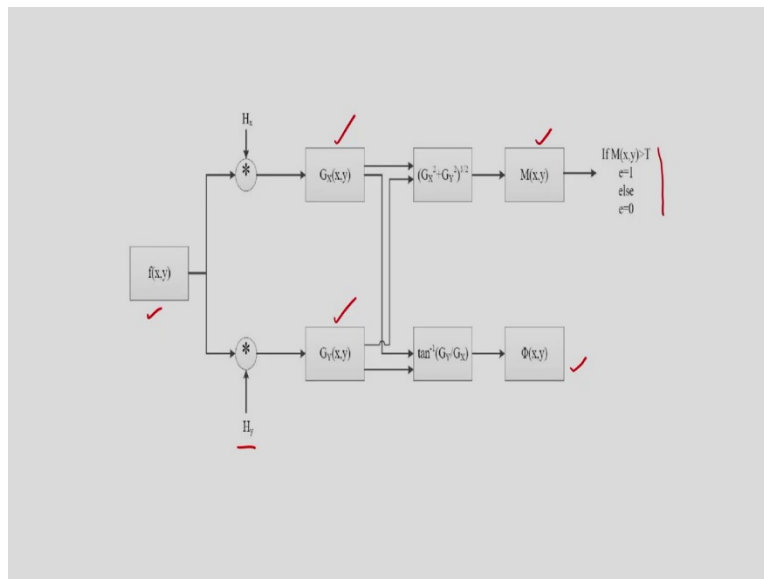
is nothing but $\tan^{-1} \frac{g_y}{g_x}$, that I can determine. So, that is nothing but the direction of the edge normal, direction of the normal to the edge, that also I can determine.

(Refer Slide Time: 35:10)



In this example I have shown the input image and I am determining the gradient along the x direction, so corresponding to this, this is the image and I can also determine the gradient along the y direction and corresponding to this, this is my output image.

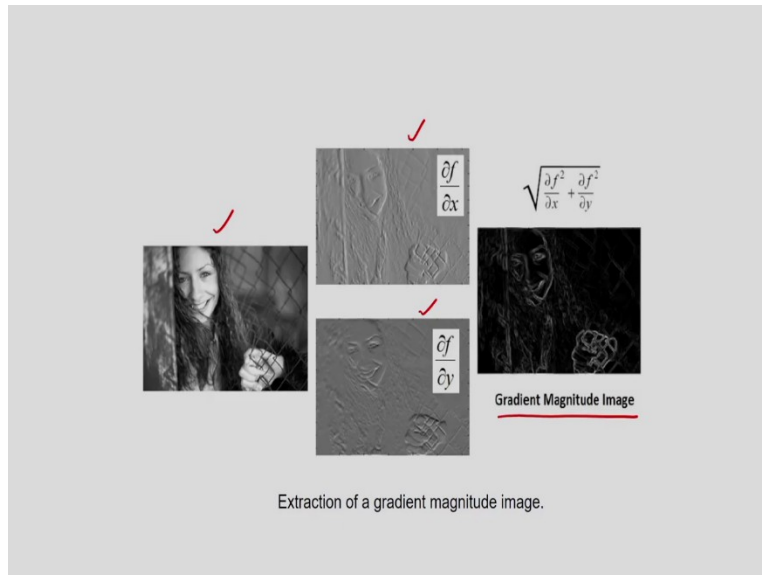
(Refer Slide Time: 35:28)



And this block diagram already I have explained, so my input image is $f(x, y)$ and I am considering two operators, one is H_x , another one is H_y , H_x is mainly to determine the gradient along the x direction and H_y to determine the gradient along the y direction. So, I can determine the gradient along the x direction and the gradient along the y direction I can determine.

And from this, I can determine the gradient magnitude I can determine and also, I can determine the angle, that is the direction of the normal to the edge, that angle also I can determine. And based on the gradient magnitude I can determine the location of the edge pixel. If the gradient magnitude is greater than a particular threshold, the edge will be 1, else edge will be 0. So, after edge detection I will be getting a binary image.

(Refer Slide Time: 36:28)



This concept I am showing here, so my input image is this, I am determining the gradient along the x direction, gradient along the y direction and after this I am determining the gradient magnitude image I am determining. And you can see the location of the edge pixels, you can see the edges of the image.

(Refer Slide Time: 36:48)

- Differentiation is highly prone to high frequency noise.
- An ideal differentiation corresponds to the function being changed in the frequency domain by the addition of a zero at origin. Thus there is an increase of 20dB per decade. This will lead to high frequency noise being amplified.
- To circumvent this problem, low pass filtering has to be performed.
- Differentiation is implemented as finite difference operation.

$$\begin{aligned}f(x) &= A \sin \omega_0 x \\f'(x) &= A \omega_0 \cos \omega_0 x \\f''(x) &= -A \omega_0^2 \sin \omega_0 x\end{aligned}$$

Now, one this that differentiation is highly prone to high frequency noise, so because if I do the differentiation, the noise will be increase. Even for a second order derivative, if I consider a second order derivative, noise will be increase. So, that is why we have to do low pass filtering before edge detection and in this case the differentiation can be implemented as finite difference operation.

So, already I have discussed about the finite difference operation, one is the forward difference, one is the backward difference, one is the central difference. So, this point is important that is because of the differentiation, the noise will be increase. So, suppose I can give one example, suppose my function is f_x , I am considering only 1D function, suppose $A \sin \omega x$. If I determine the first order derivative, that is $f'(x)$ that is $A \omega_0 \cos \omega_0 x$. So, here you see the magnitude of the noise, it is increasing.

And similarly, if I determine the double order derivative, the second order derivative, so for second order derivative, it will be $A \omega_0^2$, so you see the noise is increasing. So, you see the noise, that is the magnitude of the noise, it is increasing. Previously it is A and after this, after differentiation it will be $A \omega_0$ and after the second order derivative, it is $-A \omega_0^2$, that means the magnitude of the noise, it is increasing because of the differentiation. So, that is why we have to do low pass filtering before edge detection.

(Refer Slide Time: 38:40)

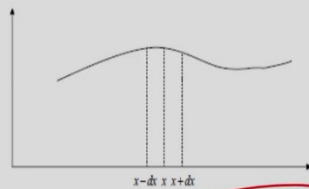
- Three types of differences generally done are:
 - forward difference = $f(x+1) - f(x)$ ✓
 - backward difference = $f(x) - f(x+1)$ ✓
 - centre difference = $\{ f(x+1) - f(x) \} / 2$ ✓
- The most common kernels used for the gradient edge detector are the Roberts, Sobel and Prewitt edge operators.

And this the operations, the finite difference operations, I have mentioned, one is the forward difference operation, the backward difference operation, another one is the central difference operation. And some common kernels are Roberts, the Sobel and the Prewitt, these are the operators. So, by using these operators, I can determine the horizontal gradient and the vertical gradient I can determine.

(Refer Slide Time: 39:09)

FINITE DIFFERENCES (1-D)

In one dimension:



$$\checkmark \frac{df}{dx} \approx \frac{f(x+dx) - f(x)}{dx} \approx \frac{f(x+dx) - f(x-dx)}{2dx}$$

$$\frac{d^2f}{dx^2} \approx \frac{f(x+dx) - 2f(x) + f(x-dx)}{dx^2} \checkmark$$

So, what is the finite difference in 1D? You can see I am determining, $\frac{df}{dx}$ I am determining, that


is nothing but $\frac{f(x+dx)-f(x)}{dx}$, that is approximately equal to, if I sends this point here to $x - dx$,

then this dx will be sends to $2dx$. So, $\frac{f(x+dx)-2f(x)+f(x-dx)}{dx^2}$. And corresponding to the

second order differentiation, you can see I can represent in the form of finite difference operations.

(Refer Slide Time: 39:50)

**FINITE DIFFERENCES IN C
(1-D)**

p → 

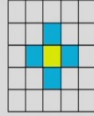
Line stored as an array:

```
{
  int i;
  for(i=0;i<n;i++){
    q[i]=p[i+1]-p[i];
  }
}
```

I can show one program segment in C, so how to determine the forward difference and the backward difference. Here I am showing the forward difference, here you can I am considering this operation you can see, this is the difference equation, the $p[i+1] - p[i]$, that I can compute to determine the finite difference.

(Refer Slide Time: 40:15)

FINITE DIFFERENCES (2-D)



$$\begin{aligned} \checkmark \frac{\partial f}{\partial x} &\approx \frac{f(x+dx, y) - f(x, y)}{dx} \approx \frac{f(x+dx, y) - f(x-dx, y)}{2dx} \checkmark \\ \checkmark \frac{\partial f}{\partial y} &\approx \frac{f(x, y+dy) - f(x, y)}{dy} \approx \frac{f(x, y+dy) - f(x, y-dy)}{2dy} \end{aligned}$$

And if I consider a 2D finite difference, you can see I am determining finite difference for x component and for y component. So, first I am determining $\frac{\delta f}{\delta x}$ that is the gradient along the x

direction, $\frac{\delta f}{\delta y}$ that is the gradient along the y direction. So, that is equal to if I consider a gradient

along the x direction, so it is equal to $\frac{f(x+dx, y) - f(x-dx, y)}{2dx}$. Because here you can see here

this point I am changing here, the point is $f(x, y)$ but now I am considering $f(x-dx, y)$. And similarly, I can determine the gradient along the y direction.

(Refer Slide Time: 41:12)

FINITE DIFFERENCES IN C (2-D)

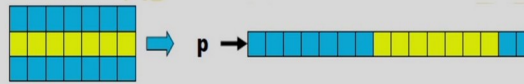
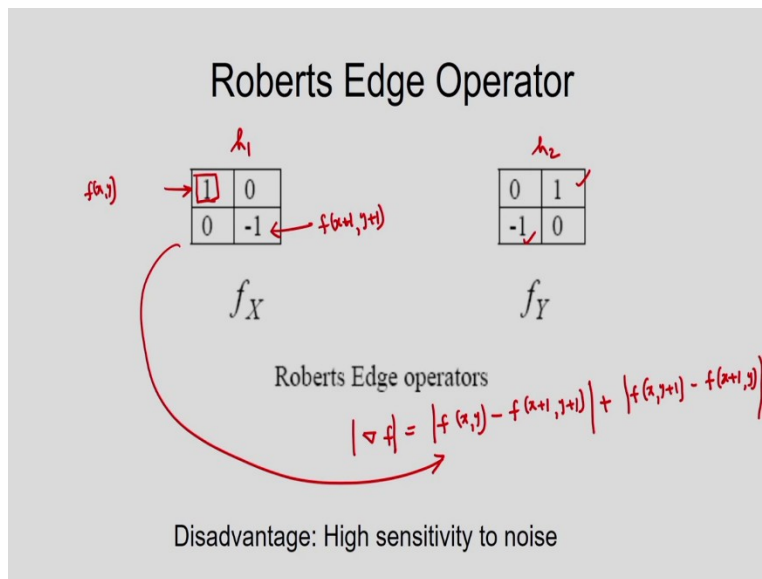


Image stored in raster format:

```
{  
  int i;  
  for(i=0;i<xdim;i++){  
    dx[i] = p[i+1]-p[i]; ✓  
    dy[i] = p[i+xdim]-p[i]; ✓  
  }  
}
```

And corresponding to this, you can see the finite difference in C program segment I am considering, you can determine the finite difference for the x component and finite difference for the y component.

(Refer Slide Time: 41:26)



So, first operator, the edge operator I am considering, the Roberts edge operator. So, what is the Roberts edge operator? So, first operator, first mask I am showing here and that is the h_1 , that is to compute the horizontal gradient and another mask I am considering, that is to compute the vertical gradient. And if I consider this is the central pixel suppose, so this is the central pixel, so I can consider, I can determine the gradient, the gradient magnitude I can determine.

So, first one $f(x, y) - f(x + 1, y + 1)$. That is the gradient along the x direction, that means I am considering this one. So, this pixel, what is this pixel? Because I am considering this pixel, this is nothing but $f(x, y)$. What is this pixel? I am considering this pixel position is $f(x + 1, y + 1)$, I am considering. So, similarly I can determine the gradient along the y direction, so $f(x, y + 1) - f(x + 1, y)$, so I can determine the gradient along the y direction.

So, that means I am considering this pixel and this pixel I am considering. So, like this I can determine the gradient magnitude, but one disadvantage of the Roberts operator is, it is highly sensitive to noise. So, that is why I can consider another operator that is the Prewitt operator.

(Refer Slide Time: 43:20)

Prewitt Edge Operator

h_1

| | | |
|----|----|----|
| -1 | -1 | -1 |
| 0 | 0 | 0 |
| 1 | 1 | 1 |

 g_x
 f_X

h_2

| | | |
|----|---|---|
| -1 | 0 | 1 |
| -1 | 0 | 1 |
| -1 | 0 | 1 |

 g_y
 f_Y
 $|g| = |g_x| + |g_y|$

Prewitt Edge operators

- Does some averaging operation to reduce the effect of noise.
- May be considered as the forward difference operations in all 2-pixel blocks in a 3 x 3 window.

So, you can see the Prewitt operator, and the first mask I am considering, that is the h_1 to determine the horizontal gradient and h_2 I am considering to determine the vertical gradient. And in this case, if you see this is the 3 by 3 mask, so that means, it does some averaging operation to reduce the effect of noise. And in this case, it may be considered as the forward difference operations in all 2-pixel blocks in a 3 by 3 window.

So, by using the Prewitt edge operators you can determine the horizontal gradient and the vertical gradient, that is h_1 and h_2 I am considering and based on this I can determine the g_x for this and also, I can determine g_y and after this I can determine the gradient magnitude I can determine. So, one advantage of this operator, the mask is, because the averaging operation is done, so that means it reduces noise.

(Refer Slide Time: 44:26)

Sobel Edge Operator

| | | |
|-------------|----|----|
| λ_1 | | |
| -1 | -2 | -1 |
| 0 | 0 | 0 |
| 1 | 2 | 1 |

f_X

| | | |
|-------------|---|---|
| λ_2 | | |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

f_Y

Sobel Edge operators

- Does some averaging operation to reduce the effect of noise, like the Prewitt operator.
- May be considered as the forward difference operations in all 2 x 2 blocks in a 3 x 3 window.

The next one is the, the Sobel edge operator. So, already I have explained about the Sobel edge operator and again I am considering two mask, one is the horizontal mask, another one is the vertical mask. So, by using these mask we can determine horizontal gradient and the vertical gradient. And in this case it is similar to Prewitt operator, because in this case also averaging operation is done to reduce the effect of noise. And in this case, we may consider the forward difference operations in all 2 by 2 blocks in a 3 by 3 window. That we can consider, because this mask is the 3 by 3 mask.

(Refer Slide Time: 45:09)

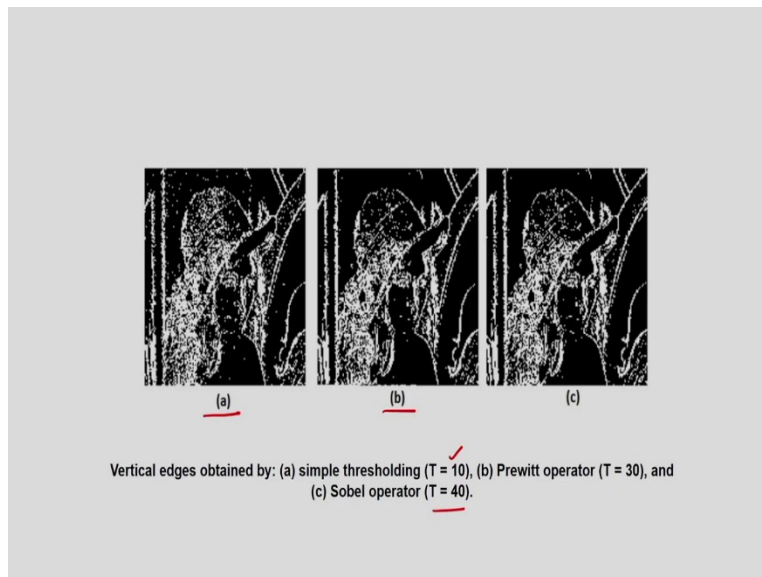
(a) (b) (c)

(a) Simple DFT, (b) Prewitt DFT, and (c) Sobel DFT

$[-1 \ 0 \ 1]$

So, here I am showing the DFT, the simple DFT; simple DFT means I am determining the DFT of $[-1 \ 0 \ 1]$, I am finding the DFT of this $[-1 \ 0 \ 1]$. And if I consider a Prewitt mask, corresponding to the Prewitt mask this is the DFT and corresponding to the Sobel, this is the DFT. You can determine the DFT considering the mask. For a simple DFT is I am considering $[-1 \ 0 \ 1]$, so you can see the response of the mask corresponding to this portion, corresponding to this portion like this you can see the response of the mask.

(Refer Slide Time: 45:47)



And in this case I am considering, vertical edges obtained by simple thresholding, so I can apply the thresholding operation, so T is equal to 10, the threshold is 10, so corresponding to this the output image is a. And if I consider b that is the Prewitt operator, I am considering, so threshold is 30 I am considering. And the third one is I am considering the Sobel operator, so for this I am considering the threshold value is equal to 40. So, you can see, so how to detect the vertical edges.

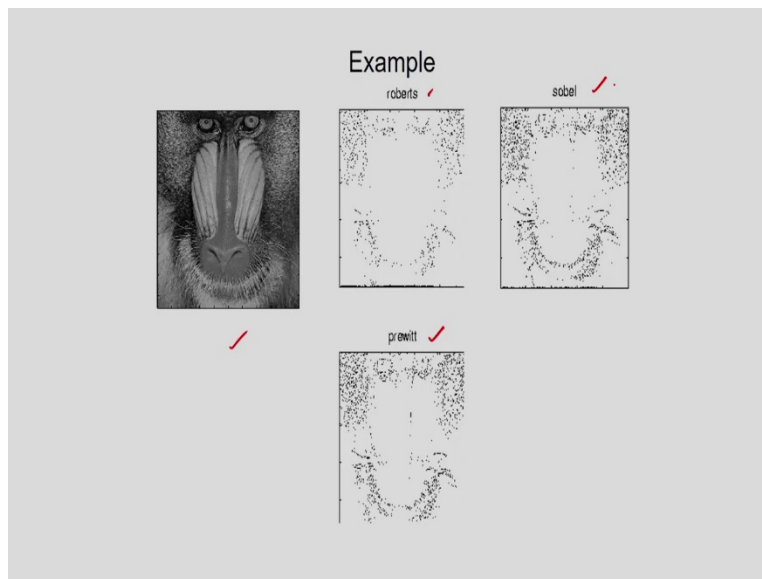
(Refer Slide Time: 46:24)

Gradient Based Edge detection

- Find f_x and f_y using a suitable operator. $\partial_x \quad \partial_y$
- Compute gradient $|\nabla f| = |f_x| + |f_y|$ $|\nabla f| = |g_x| + |g_y|$
- Edge pixels are those for which $|\nabla f| > T$ where T is a suitable threshold

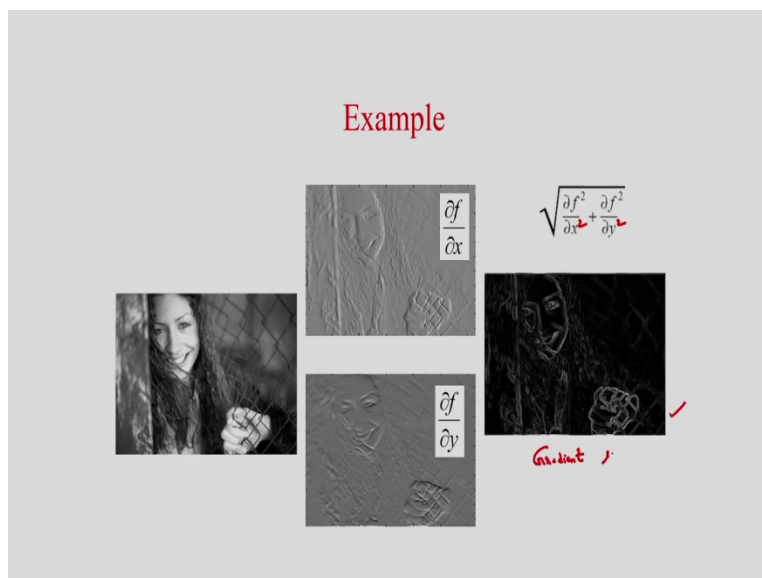
So, the procedure already I have explained, so how to determine the edge, so for this we have to determine the f_x , here I am showing f_x , actually it is the g_x , already I have explained g_x means the gradient along the x direction, gradient along the y direction I have to determine. And after this I have to determine the gradient magnitude, that is nothing but the gradient magnitude is equal is g_x gradient along the x direction and the gradient along the y direction I can determine. And after this I have to compare the gradient magnitude with the threshold and based on this I can determine the edge pixel, whether particular pixel is the edge pixel or not the edge pixel. And after edge detection I will be getting a binary image.

(Refer Slide Time: 47:10)



Here I am showing one example, input image I am considering and you can see the outputs corresponding to the mask, first I am considering the Roberts mask, the next I am considering the Prewitt mask and the third one is the Sobel mask. You can see the comparisons between these three masks.

(Refer Slide Time: 47:31)



Now, I am considering the case of the Laplacian that is the second order derivative I am considering. In this example I am showing mainly the gradient operation that is I am finding the

gradient along the x direction, gradient along the y direction. And after this I can determine the gradient magnitude, I can determine like this. So, this is the gradient image.

(Refer Slide Time: 47:56)

Second derivative Based

- For the two-dimensional image, we can consider the orientation-free Laplacian operator as the second derivative. The Laplacian of the image f is given by

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

$\frac{\partial f}{\partial x} = f(x+1, y) - f(x, y)$
 $\frac{\partial^2 f}{\partial x^2} = \frac{\partial}{\partial x} \{ f(x+1, y) - f(x, y) \}$
 $= f(x+2, y) - f(x+1, y) - f(x, y) + f(x, y)$
 $\frac{\partial^2 f}{\partial x^2} = f(x+1, y) - f(x, y) - f(x, y) + f(x-1, y)$
 $= f(x+1, y) + f(x-1, y) - 2f(x, y)$

$\frac{\partial f}{\partial y} = f(x, y+1) - f(x, y)$
 $\frac{\partial^2 f}{\partial y^2} = \frac{\partial}{\partial y} \{ f(x, y+1) - f(x, y) \}$
 $= f(x, y+2) - f(x, y+1) - f(x, y) + f(x, y)$
 $\frac{\partial^2 f}{\partial y^2} = f(x, y+1) + f(x, y-1) - 2f(x, y)$

$\nabla^2 f = f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)$

Second derivative Based

- For the two-dimensional image, we can consider the orientation-free Laplacian operator as the second derivative. The Laplacian of the image f is given by

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

$\frac{\partial f}{\partial x} = f(x+1, y) - f(x, y)$
 $\frac{\partial^2 f}{\partial x^2} = \frac{\partial}{\partial x} \{ f(x+1, y) - f(x, y) \}$
 $= f(x+2, y) - f(x+1, y) - f(x, y) + f(x, y)$
 $\frac{\partial^2 f}{\partial x^2} = f(x+1, y) - f(x, y) - f(x, y) + f(x-1, y)$
 $= f(x+1, y) + f(x-1, y) - 2f(x, y)$

$\frac{\partial f}{\partial y} = f(x, y+1) - f(x, y)$
 $\frac{\partial^2 f}{\partial y^2} = \frac{\partial}{\partial y} \{ f(x, y+1) - f(x, y) \}$
 $= f(x, y+2) - f(x, y+1) - f(x, y) + f(x, y)$
 $\frac{\partial^2 f}{\partial y^2} = f(x, y+1) + f(x, y-1) - 2f(x, y)$

$\nabla^2 f = f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)$

Now, I am considering the second order derivative, that is the Laplacian. So, Laplacian I am considering, so for Laplacian we have to find a 0 crossing. So, how to determine the Laplacian by considering the finite difference operation, that I want to explain. So, we have to determine

the gradient along the x direction, gradient along the y direction; $\frac{\delta f}{\delta x}$ and after this I can again differentiate to get the second order derivative.

So, I can show how to determine the second order derivative. So, suppose I am applying the forward difference equation, so $\frac{\delta f}{\delta x}$ is nothing but $f(x + 1, y) - f(x, y)$. And this

is first order differentiation, that is the forward difference I am considering. And after this $\frac{\delta f^2}{\delta x^2}$ I

am considering, this is the second order derivative, $\frac{\delta}{\delta x} f(x, y) - f(x, y)$ I am determining. So, it is $f(x + 2, y) - f(x + 1, y) - f(x + 1, y) + f(x, y)$. This is the differentiation with respect to the point x plus 1.

So, if I want to determine the differentiation with respect to x , then in this case I have to subtract 1. So, I am repeating this, this is the differentiation with respect to the point $x + 1$. And in this case if I want to determine the differentiation with respect to x , I have to subtract 1. So, that

means what I am getting this $\frac{\delta f^2}{\delta x^2}$ will be $f(x + 1, y) - f(x, y) - f(x, y) + f(x - 1, y)$. So, finally I

will be getting $f(x + 1, y) + f(x - 1, y) - 2 f(x, y)$. So, this is the $\frac{\delta f^2}{\delta x^2}$.

Similarly, I can also determine $\frac{\delta f^2}{\delta y^2}$ also I can determine, that is the second order derivative also I

can determine, so it will be $f(x, y + 1) + f(x, y - 1) - 2 f(x, y)$. So, this $\frac{\delta f^2}{\delta x^2}$ and $\frac{\delta f^2}{\delta y^2}$ I can

determine and finally I can determine the Laplacian I can determine. So, Laplacian will be $f(x + 1, y) + f(x - 1, y) + f(x, y + 1) + f(x, y - 1) - 4 f(x, y)$. So, that is the second order derivative.

So, corresponding to this, what will be the mask? My mask will be you can see here, so if I consider coefficients here, so corresponding to the central pixel, the central pixel is nothing but x, y this is central pixel, x, y . So, corresponding to the central pixel my coefficient will be, the coefficient of the mask will be - 4 because it is - 4, you can see here and corresponding to this point $f(x + 1, y)$ so that means corresponding to this point $f(x + 1, y)$ then this value will be 1, $f(x - 1, y)$ that means $x - 1, y$ corresponding to this point that value will be 1, the coefficient.

Corresponding to $f(x, y + 1)$, so $(x, y + 1)$ is this point, so this will be 1, and the finally $f(x, y - 1)$, so corresponding to the point it will be 1. So, my mask will be, so my mask for this Laplacian will be now - 4, 1, 1, 1, 1. So, this is the mask and the remaining it will be 0, 0, 0, 0. So, this is the mask corresponding to the Laplacian and if I consider a diagonal pixel for this computations, then in this case the mask, my mask will be, so this will be the (Laplacian), this is the mask corresponding to the Laplacian if I consider the diagonal pixels. So, I have defined how to determine the mask corresponding to the operation that is the Laplacian operation.

(Refer Slide Time: 54:57)

$$\begin{aligned}\frac{\partial f}{\partial x} &= f[x+1, y] - f[x, y] \\ \frac{\partial^2 f}{\partial x^2} &= \frac{\partial}{\partial x} \frac{\partial f}{\partial x} \\ &= \frac{\partial}{\partial x} (f[x+1, y] - f[x, y]) \\ &= f[x+1, y] - f[x, y] - (f[x, y] - f[x-1, y]) \\ &= f[x+1, y] - 2f[x, y] + f[x-1, y]\end{aligned}$$

Similarly

$$\begin{aligned}\frac{\partial^2 f}{\partial y^2} &= f[x, y+1] - 2f[x, y] + f[x, y-1] \\ \nabla^2 f &= f[x+1, y] - 2f[x, y] + f[x-1, y] + f[x, y+1] - 2f[x, y] + f[x, y-1] \\ &= f[x+1, y] + f[x-1, y] + f[x, y+1] + f[x, y-1] - 4f[x, y]\end{aligned}$$

So, this derivation already I have shown, so how to determine the Laplacian, so you can determine the Laplacian here.

(Refer Slide Time: 55:08)

Laplacian Operator

| | | |
|---|----|---|
| 0 | 1 | 0 |
| 1 | -4 | 1 |
| 0 | 1 | 0 |

Laplacian mask ✓

| | | |
|---|----|---|
| 1 | 1 | 1 |
| 1 | -8 | 1 |
| 1 | 1 | 1 |

Laplacian mask including the diagonal pixels ✓

- Apply Laplacian mask on the image.
- Detect the zero crossings, as the zero crossing corresponds to the situation where pixels in a neighbourhood differ from each other in an image, *i.e.*, $|\nabla^2 f(p)| \leq |\nabla^2 f(q)|$, where p and q are two pixels.
 - Advantages:
 - No thresholding ✓
 - symmetric operation
 - Disadvantages:
 - Noise is more amplified ✓
 - It does not give information about edge orientation

And based on this Laplacian, you can determine the mask, that is the Laplacian mask and if I consider a diagonal pixels that is the Laplacian mask. So, for this how to determine the edge pixels? So, the method is apply Laplacian mask on the image, so this is the first step and what is the method? We have to determine the zero crossings, so detect the zero crossings, as the zero crossing corresponds to the situation where the pixels in a neighborhood differ from each other in an image, so that is the condition.

And if I consider p and q are two pixels, so that means I have to determine the zero crossings, and based on the zero crossings, I can determine the pixels, the particular pixel is an edge pixel or the not edge pixel, that condition I can determine. So, one advantage of the Laplacian operator is no thresholding is required because in case of the gradient operation we considered thresholding operation, but in this case only we have to consider zero crossings, we have to find a zero crossing, so that is why no thresholding is required.

And it is symmetric, if you see the mask it is a symmetric operation. But disadvantages, because of the second order derivative, the noise is amplified. And it does not give the information about the edge orientation. In case of the gradient operation, we can determine θ_g that is the direction of the edge normal, the normal to the edge I can determine. But in this case, it is not possible to get that information, the information about the edge orientation.

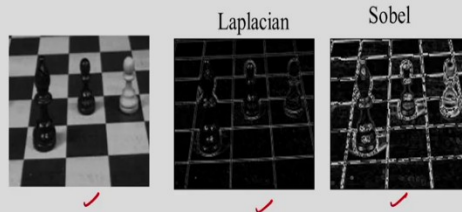
(Refer Slide Time: 56:53)

Edge sharpening with the Laplacian filter:

The Laplacian filter is first applied to an image $f(x,y)$ and then a fraction is subtracted (determined by the weight w) of the result from the original image as follows:

$$\hat{f}(x,y) \leftarrow f(x,y) - w(H * f(x,y))$$

↑ ↑ ↑
Edge sharpening Original Laplacian
image image filtering



After this I am considering edge sharpening with a Laplacian filter. So, here you can see I am considering the original image $f(x,y)$ and I am considering $f(x,y)$, the $f(x,y)$ is a convolved with H , H is nothing but the Laplacian I am considering. That means I am applying the Laplacian filtering and I am considering the weight, some weight w is considered. So, for this what I have to do, the Laplacian filter is first applied to an image, the image is $f(x,y)$. Then a fraction is subtracted, the fraction is determined by the weight, the weight is w , from the original image, the original image is $f(x,y)$. So, $f(x,y)$ is my original image.

So, if I apply this operation, you can see the edges. So, I am showing one example here, so this is my input image, I am applying the Laplacian for edge sharpening. And the second case I am showing the edge detection by using the Sobel operator, you can see the difference between these, one is the edge sharpening by using the Laplacian and another one is the edge detection by Sobel operator. So, this operation is very simple, so for this what I have to consider, first I have to apply the Laplacian filter to an image and after this I have consider a fraction of this, so for this I am considering the weight w , the result of this is subtracted from the original image. The original image is $f(x,y)$.

(Refer Slide Time: 58:28)

Unsharp masking

- Subtract a smoothed-version of an image from the original image. This step enhances the edges.
- Subsequently, the unsharp version of the image is obtained by again adding a fraction (determined by the weight w) of the result/mask M to the original image.
- The edges in the image are sharpened.

$$\begin{aligned} \checkmark \quad M &\leftarrow f(x, y) - (f(x, y) * H) = f(x, y) - \hat{f}(x, y) \\ \hat{f}(x, y) &\leftarrow f(x, y) + wM = f(x, y) + w(f(x, y) - \hat{f}(x, y)) \\ \therefore \checkmark f(x, y) &\leftarrow f(x, y) + wM = (1 + w)f(x, y) - w\hat{f}(x, y) \end{aligned}$$

And after this, I am considering another masking that is the unsharp masking. So, what I am considering, so first the subtract a smoothed version of an image from the original image. So, if I do this operation, this enhance the edges. So, first what I am considering, this is my original image $f(x, y)$ and I am considering the Laplacian operator, the Laplacian operator is H . So, image is convolved with H , that means I am considering the Laplacian operator, and subtract a smoothed version of the image from the original image, that I am considering.

And subsequently, the unsharp version of the image is obtained by adding a fraction of the resultant mask because I am determining the mask, that mask I am obtaining from this operation, this operation is this, so $f(x, y)$ is the original image and this mask I am adding and for this I am considering the fraction of the mask, so that means I am considering the weight, the weight is w , that I am considering.

So, this is the concept of the unsharp masking. So, subtract a smoothed version of an image from the original image, this step enhances the edges. So, first step you can see, I am determining the mask, the mask is M , so it is obtained by $f(x, y)$ that is f approximate x, y ; that I am considering this one. And after this the unsharp version of the image is obtained by adding the fraction of the resultant mask, the mask is M to the original image, the original image is $f(x, y)$. So, I will be getting this one, so this is called the unsharp masking.

(Refer Slide Time: 60:23)

Kirsch Compass Masks

- Taking a single mask and rotating it to 8 major compass orientations: N, NW, W, SW, S, SE, E, and NE.
- The edge magnitude = The maximum value found by the convolution of each mask with the image.
- The edge direction is defined by the mask that produces the maximum magnitude.

0°, 45°, 90°, 135°

And after this I am considering some operations, these are called the compass operations. So, by using these operations we can determine different lines present in an image. So, suppose some lines oriented in the direction, the directions are suppose; North directions, North West direction, West direction, South West direction, South direction like this, so all the direction suppose the lines, some lines are available. So, these lines can be determined by using these compass masks.

So, what is the compass masks, I can explain now this. So, by using these compass masks, you can determine the lines oriented in different directions. Directions means, I can consider the directions like 0-degree directions, I can consider may be 45-degree direction I can consider, 0-degree directions I can consider, 45-degree directions I can consider, 90-degree directions I can consider, or maybe 135-degree directions I can consider. So, I can determine or I can detect a line oriented in these directions. So, for this I can apply the compass masks.

(Refer Slide Time: 61:39)

Kirsch Compass Masks

- Similar to the Kirsch masks, with mask coefficients of 0, 1, and 2:

$$\begin{array}{cccc}
 N = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} & W = \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix} & S = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} & E = \begin{bmatrix} 2 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -2 \end{bmatrix} \\
 NW = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} & SW = \begin{bmatrix} 0 & -1 & -2 \\ 1 & 0 & -1 \\ 2 & 1 & 0 \end{bmatrix} & SE = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} & NE = \begin{bmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix}
 \end{array}$$

So, you can see these masks, so by using these masks I can determine a particular line oriented in a particular direction, so first one is the North, next one is the West, next one is the South, next one is the East, North West, South West, like this I can determine the lines in these directions.

(Refer Slide Time: 62:02)

$H_0^K = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$ $H_4^K = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$ $H_1^K = \begin{bmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix}$ $H_5^K = \begin{bmatrix} 2 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -2 \end{bmatrix}$ $H_2^K = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$ $H_6^K = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$ $H_3^K = \begin{bmatrix} 0 & -1 & -2 \\ 1 & 0 & -1 \\ 2 & 1 & 0 \end{bmatrix}$ $H_7^K = \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix}$

$H_4^K = -H_0^K$
 $0^\circ, 45^\circ, 90^\circ, 135^\circ$

$f(x,y) + H_4^K$
 $= f(x,y) + (-H_0^K)$
 $= -(f(x,y) + H_0^K)$

$D_0 \leftarrow f(x,y) * H_0^K$
 $D_1 \leftarrow f(x,y) * H_1^K$
 $D_2 \leftarrow f(x,y) * H_2^K$
 $D_3 \leftarrow f(x,y) * H_3^K$
 $D_4 \leftarrow -D_0$
 $D_5 \leftarrow -D_1$
 $D_6 \leftarrow -D_2$
 $D_7 \leftarrow -D_3$

$M(x,y) \triangleq \max\{|D_0(x,y)|, |D_1(x,y)|, |D_2(x,y)|, |D_3(x,y)|\}$

So, for this I am considering the masks like these, these are the masks, one is $H_0, H_1, H_2, H_3, H_4, H_5, H_6, H_7$. So, these masks I am considering and in this case you can see, if I compare these masks H_0 and H_4 , so if you just compare this one, you can see H_0 and the H_4 , you can see these masks. So, what is the difference between these two masks? Only the sign is reverse, the H_0 and H_4 , it is very similar, only the sign is reverse.

Similarly, H_1 and H_5 , it is almost similar, only the sign is reverse, because this is + 2, 1, 1; - 1, - 1, - 2; you can see. So, H_1 and H_5 they are very similar, except the sign. Similarly, H_2 and the H_6 these are very similar, the sign is reverse, and H_3 and H_7 that is also similar, because only the sign is reversed. So, that means in this case I can write like this, H_4^k is equal to $-H_0^k$, I can write like this.

And in this case I can apply the convolution operations to determine the direction of the line, so I can determine a particular line oriented in different directions, the directions already I have mentioned, the 0 degree, 45 degree, 90 degree or maybe 135 degree, these directions I am considering. So, orientation of a particular line along these directions, so suppose the line is present in this direction, suppose this is the image I am considering. So, one line is in this direction, I can consider or maybe one line maybe in this direction, so one line maybe in this direction, 0-degree, 90-degree, 45 degree like these lines I can consider.

So, maybe like this, this line I can also consider. So, I can determine all these lines. So, for this I have to do the convolution of the image with the mask. So, suppose if I do the convolution of the image with mask $f(x, y)$ that is convolved with, suppose the mask is H_4^k , because H_4^k is very similar to H_0^k . So, that is equal to $f(x, y)$ convolved with $-H_0^k$. Because, H_0^k and the H_4^k they are very similar, only the sign is reversed, so I can consider this one.

That means it is equal to minus $f(x, y)$; I can consider this one, only the sign is reverse between the mask H_0 and the H_4 . Similarly, for the H_1 and the H_5 ; similarly, H_2 and H_6 ; H_3 and the H_7 . So, I have to do the convolution between the mask and the image and after this what I have to consider like, convolution I can consider like this, suppose I can get the value D_0 , suppose after the convolution. So, $f(x, y)$ it is convolved with H_0^k , I am getting value D_0 . I am getting another value suppose D_1 , the image is convolved with H_1^k , the another mask is H_1 .

I am getting another value that is D_2 , the image is convolved with H_2^k and I can get another value that is D_3 , the image is convolved with H_3^k . So, only I have to do the convolution for these masks; H_0 , H_1 , H_2 , H_3 . For remaining what I can do? Suppose, for D_4 , how to get D_4 ? D_4 is nothing but it is $-D_0$. What is D_5 ? It is nothing but $-D_1$. What is D_6 ? It is nothing but $-D_2$. And

what is D_7 ? D_7 is nothing but simply minus D_3 . So, only for these values, I have to compute the convolutions, that I can determine D_0, D_1, D_2, D_3 I can determine.

And from D_0, D_1, D_2, D_3 I can determine D_4, D_5, D_6 and the D_7 . Then in this case I can determine, this value the D_0, D_1, D_2, D_7 I can determine and I am considering the absolute value I am considering. And I am determining the maximum of this, the maximum of this will give the direction of a particular line. So, suppose that this is maximum, so corresponding to this, what I will be getting, then in this case I will be getting a line in the vertical direction, in the 90-degree directions.

Because if I consider D_0 from this mask, so you can see the response of the mask, the response of the mask you can see here. So, by using this information that is the maximum I can determine from D_0, D_1, D_2, D_7 , I can determine a particular line. So, this line that is suppose the 90 degree direction, 0 degree directions, 45 degree directions, 135 degree directions, so detection of a particular line in these directions, I can determine by using this compass operator.

Up till now I have discussed the concept of the edge detection. So, I have discussed how to determine the edge by using the gradient operations, I can determine the first order derivative or I can determine the second order derivatives. The maximum of the first order derivative, it will give the location of the edge pixels and for the second order derivative, I have to see the zero crossings. And after this I defined some masks, the masks very important, the Roberts, Sobel and the Prewitt, I have defined.

And after this I discussed about the Laplacian operators, so that means by using the Laplacian operator, we can determine the second order derivative and we have to see the zero crossings. And after this I discussed the concept of the compass operators, so we can determine a particular line oriented in different directions, directions maybe 0-degree, 45-degree, 90-degree, 135 degree. So, these lines I can determine by using the compass operators.

After this I will discuss another technique of edge detection, that is the model-based technique, which is based on the mammalian visual system that is the human visual system. So, next class I am going to discuss about the model-based edge detection techniques, so let me stop here today. Thank you.