

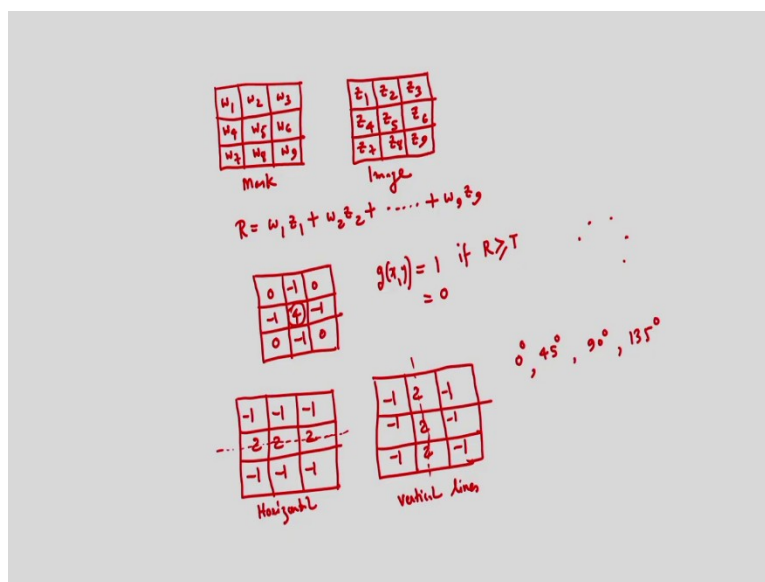
Computer Vision and Image Processing - Fundamentals and Applications
Professor Doctor M. K. Bhuyan
Department of Electronics and Electrical Engineering
Indian Institute of Technology, Guwahati
Lecture 23
Hough Transform

Welcome to NPTEL MOOCs course on Computer Vision and Image Processing - Fundamentals and Applications. In my last classes I discussed about the concept of edge detection. So, how to detect the edges in an image, many edges in an image can be approximated by lines. So, I can detect lines present in an image. So, today I am going to discuss how to detect the lines or maybe the circles in an image.

Line detection by one transformation, that is the directional transformation and that is called Hough Transform because these lines or the edges or the points are the features of the image. So, line detection is quite important that is nothing but the extraction of the line, the detection of the line and that can be considered as an image feature.

This Hough transform is quite useful to detect the lines present in an image, also I can detect circles present in an image. So, today I am going to discuss about this Hough transform. So, before going to the Hough transform how to detect the lines by using the mask, the masking operations.

(Refer Slide Time: 1:47)



So, I can explain this concept suppose, I have a mask and this is my mask, the weights are w_1 w_2 w_3 these are the weights of the masks and I am considering suppose one image, the mask I am considering it is a 3 by 3 mask I am considering and suppose I am considering the image the pixels are suppose z_1, z_2 like this, these are the pixels of the image and I am only considering the 3 by 3 image.

Now, how to detect a particular line or maybe a particular point. So, in this case, what is the masking operation? So, I have to place the mask over the image. So, this is my mask and this is my image. So, how to do the masking operation? So, I have to place the mask over the image and corresponding to this I have to determine the response of the mask. So, response of the mask is w_1z_1, w_2z_2 , up to w_9z_9 . So, this is the response of the mask.

Suppose, if I consider this mask, this is nothing but the Laplacian mask. So, you probably considered this mask and if I convolve the image with this mask, this Laplacian mask then you can see in the center pixel the value is 4. So, that means, corresponding to any isolated point the response of the mask will be maximum. So, in this case $g(x, y)$ that is the output will be 1 if my response of the mask is greater than a particular threshold otherwise 0.

So, that means corresponding to isolated points I will be getting $g(x, y)$ is equal to 1 and otherwise it will be 0. So, for all these points isolated points and my response will be 1, otherwise it will be 0. So, that means, I can detect points in an image by using this mask, the mask is the Laplacian mask and already I have discussed about the compass operators.

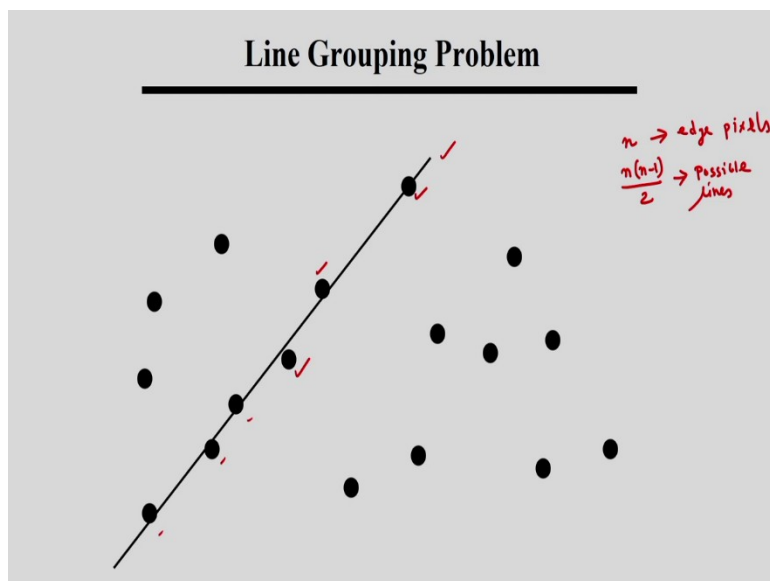
So, suppose if I consider the mask, suppose if I consider this mask or if I consider another mask suppose, so, if you see the first mask, this mask you can see, you can see the response, response will be maximum along this line. That means, this mask can detect horizontal lines and this mask if you see, this mask can detect the vertical lines.

If I use the compass operators, I can detect the lines, suppose it is in 0-degree direction, 45-degree directions, 90-degree directions and also 135-degree directions. So, these lines I can detect, the lines oriented in this particular direction that is 0-degree direction, 45-degree direction, 90-degree direction and 135-degree directions.

So, for this I can use the compass operator, but for other lines, I cannot use the mask, it is not possible to detect the all the lines which are oriented in different directions not the 0 degree, not

the 45 degree in different directions, then in this case it is very difficult to detect the lines present in an image. So, that is why I can consider the transformation that directional transformation and that is called the Hough transformation.

(Refer Slide Time: 5:49)



So, before going to the Hough transformation, suppose I have the edge pixels and already I have explained that is the many edges can be approximated by a straight line. So, mainly in this case I am considering how to detect the straight lines, suppose, if I consider n number of edge pixels, the edge pixels then corresponding to this, how many possible lines will be there, because it is I have to group the points to make lines. So, how many possible lines? This $\frac{n(n-1)}{2}$ possible lines.

So, I have n number of edge pixels and corresponding to n number of edge pixels I have to find how many lines I can produce from these edge pixels. So, that means, I can get $\frac{n(n-1)}{2}$ possible lines. So, this is nothing but the line grouping problem. So, in this case you can see I am considering one line considering the edge pixels. So, this edge pixel I am considering, this edge pixel I can consider like this, I can consider these pixels.

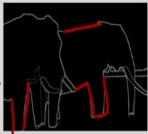
Whether this pixel is close to a particular line that I am considering and that is mainly the line grouping technique and by this I can consider or I can detect a line, but the main problem is if I

consider n number of edge pixels, then number of possible lines by this grouping process it will be $\frac{n(n-1)}{2}$. So, that is very difficult to do that is computationally complex. So, that is why I have to consider the transformation, the transformation is the Hough transform.

(Refer Slide Time: 7:48)

Line Detection and Hough transform

- Many edges can be approximated by straight lines.
- For n edge pixels, there are $\frac{n(n-1)}{2}$ possible lines. ✓
- To find whether a point is closer to a line we have to perform $n \times \frac{n(n-1)}{2}$ comparisons. Thus, a total of $O(n^3)$ comparisons.



So, this concept I have explained. So, many edges can be approximated by straight lines. So, here you can see these many edges I can approximate as straight lines. So, these are my straight lines. So, many edges can be approximated by straight lines. The second point is if I consider n number of edge pixels, then in this case how many possible lines? The number of possible lines will be $\frac{n(n-1)}{2}$ possible lines.

And also to find whether a point is closer to a line, we have to perform some comparison, how many comparisons we have to do? $\frac{n(n-1)}{2}$ number of comparisons I have to do. So, then in this case, you can see the order of the complexity. We have to do many comparisons to find a particular line. So, that is the problem of line grouping. So, that is why we will consider the transformation is the Hough transform.

(Refer Slide Time: 8:57)

Hough Transform

- Elegant method for direct object recognition
 - Edges need not be connected ✓
 - Complete object need not be visible ✓
 - Key Idea: Edges VOTE for the possible model

So, in this case very important technique to detect the lines or I can detect the circles present in an image and in this case, edges need not be connected, in case of the Hough transform. And one thing is that the complete object need not be visible. And this is also another point for the Hough transform and the key idea is, so, we have to do the voting, we have to go for the voting to find possible models. So, that is the case.

(Refer Slide Time: 9:31)

Image and Parameter Spaces

Equation of Line: $y = mx + c$

Find: (m, c)

Consider point: (x_i, y_i)

$y_i = mx_i + c$ or $c = -x_i m + y_i$

Parameter space also called Hough Space

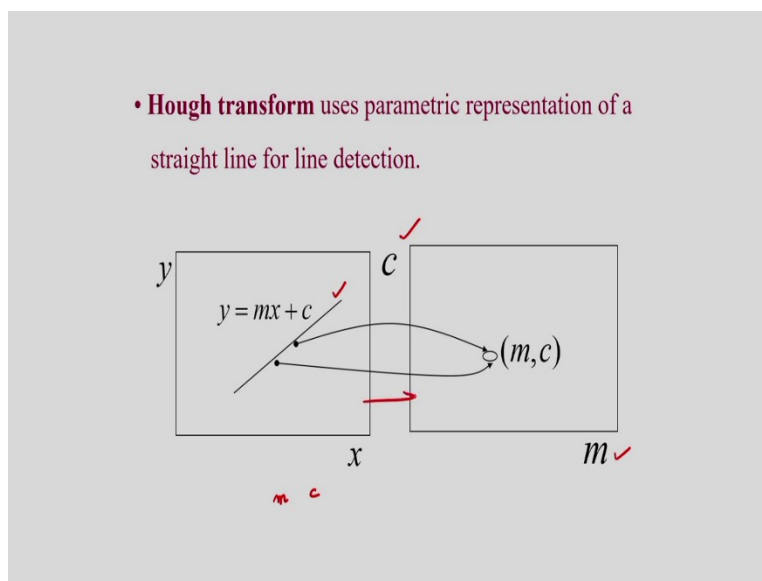
So, in this case I have shown here you can see I am considering one line, the line equation is $y = mx + c$. So, that is the equation of the line and I am considering the parameters, these are the

parameters of the line one is the gradient, the gradient is m and the intercept is c . So, I have these parameters one is the m , another one is c , so line is $y = mx + c$. In these two figures I have shown a two spaces, one is the image space that is the x and y space, I am showing the x and y coordinates and another one is the parametric space, I have shown the m and c values, the m is the gradient and c is the intercept.

Suppose, if I consider one point, the point is suppose x_i, y_i , I am considering them, this point I am considering. Corresponding to this point what I will be getting in the m, c space, the m, c space is called the parametric space. In the parametric space I will be getting a line. So, you can see suppose anyone on the line you can see. Suppose, if I consider another point that is collinear with the previous point, suppose, if I consider this point, the second point that is collinear with the previous point and corresponding to this point also I will be getting another line, suppose, this line I am getting.

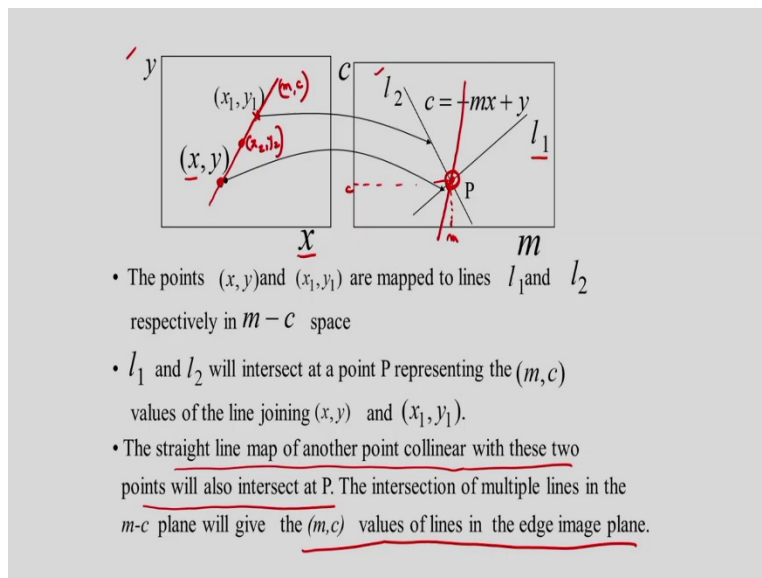
These lines will intersect at a particular point, the point is this point and that is the corresponding point, the point value is basically the m, c value of the line I have shown here in the figure 1. For corresponding to the line, I have shown in the figure, I have the m, c value that is a gradient and the intercept value. And in the second figure, if I consider the parametric space, corresponding to all the points which are collinear I will be getting lines and they will intersect at a particular point and that point is nothing but the m, c value of the line.

(Refer Slide Time: 11:33)



So, in this case you can see, so, how to consider the parametric representation of a straight line for line detection. So, here I have shown the x, y space and the parametric space and you can see corresponding to this line, particular line in the x, y space, I am getting the m, c value; m, c value in the parametric space, the parametric space is m and c . Because corresponding to this line, I have the fix value, the fix value is m is the fix and c is fix corresponding to that particular line. So, you can see the mapping between this x, y space and the m, c space. So, just you see the mapping of this.

(Refer Slide Time: 12:20)



Now, in this case you can see I am considering two points one is (x_1, y_1) another one is suppose x, y that is in the x, y space. So, two points I am considering one is the x, y point, this point and another point is suppose (x_1, y_1) . So, corresponding to this (x, y) point the first point suppose, corresponding to this first point, I will be getting a line in the parametric space. So, parameter space is m, c already I have explained. So, corresponding to the first point, the point is (x, y) , I will be getting a line that is l_1 in the parametric space.

And suppose the (x_1, y_1) that is a collinear point with respect to the first point, that is a collinear point, then in this case corresponding to the second point, the second point is (x_1, y_1) , I will be getting another line the line is l_2 in the parametric space, these two lines will intersect at the point, the point is P , that gives the m, c value corresponding to the line, that line I can join between these two points corresponding to this line I have the m and c value.

So, in this case, if you see here, these two lines l_1 and l_2 , they will intersect at the point P and the value of the point P is the m and c value of the line joining these two points (x, y) and (x_1, y_1) . So, that is already I have explained the straight-line map of another point collinear with these two points will also intersect at point P. So, if I consider another point that is collinear with the points (x, y) and (x_1, y_1) that is another point I am considering, (x_2, y_2) .

So, corresponding to this point also, I will be getting one line in the parametric space and in this case, it will intersect at the point, the point is P. P is nothing but the m, c value of the lines in the edge image plane. So, this is the concept of the Hough transformation. So, that means, in this case what I have to do I have to count how many times it is intersecting. So, in suppose I have considered three points, then in this case, three lines I can consider, that means how many times it is intersecting at a particular point? Three times it is intersecting.

So, that means I have to count that is the voting, this is called the voting. So, how many times I am getting the vote corresponding to these points, which are collinear, so, that means, in this case, I will be getting three votes, if I consider three points, that is which are collinear if I consider four points, then in this case I will be getting four votes. So, that means, I have to count how many intersections in the (m, c) space that I have to consider.

(Refer Slide Time: 15:28)

Line Detection by Hough Transform

Algorithm:

- Quantize Parameter Space (m, c)
- Create Accumulator Array $A(m, c)$
- Set $A(m, c) = 0 \quad \forall m, c$
- For each image edge (x_i, y_i) increment:

$$A(m, c) = A(m, c) + 1$$
- If (m, c) lies on the line:

$$c = -x_i m + y_i$$
- Find local maxima in $A(m, c)$

Group the edges that belong to each line by traversing each line.

So, for this Hough transformation, I have to consider how many times I will get the intersections in the (m, c) space. So, that is why to count these I am considering one accumulator the

accumulator is $A(m, c)$, because in this case my parameter space is (m, c) that is also quantized and I have to initialize the accumulator. So, initial value is 0 and corresponding to each image edge point, suppose, if I consider one point, the edge point is this (x_i, y_i) , I have to increment the accumulator, that means I am observing how many times I am getting the intersections, that means I am counting the voting.

And if m, c lies on the line, then in this case, I can get this $c = -x_i m + y_i$ and like this I have to count for all the pixels, the pixels means the edge pixels and I can find the local maximum the local maximum is I can find from the accumulator. So, suppose in this case if I consider these two votes that means I have the intersections corresponding to two points.

So, like this I can find the maximum in the accumulator, accumulator array. After this, the group the edge that belongs to each line by traversing each line. So, that in this case I can do the grouping, group the edges that belong to each line by traversing each line, so, that I can find a line. So, this is nothing but the voting.

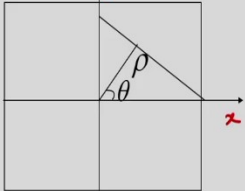
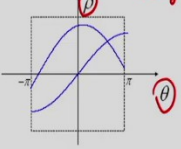
(Refer Slide Time: 17:11)

Better Parameterization

Note: m and c are, in principle, unbounded: cannot handle all situations.

$$-\infty \leq m \leq \infty$$
 Large Accumulator
 Improvement: (Finite Accumulator Array Size)

- Instead of (m, c) , we can consider (ρ, θ) as the parameters with θ varying between -90° and 90° and ρ varying from 0 to $\sqrt{M^2 + N^2}$ for an $M \times N$ image.

$x \cos \theta + y \sin \theta = \rho$
 $y = mx + c$

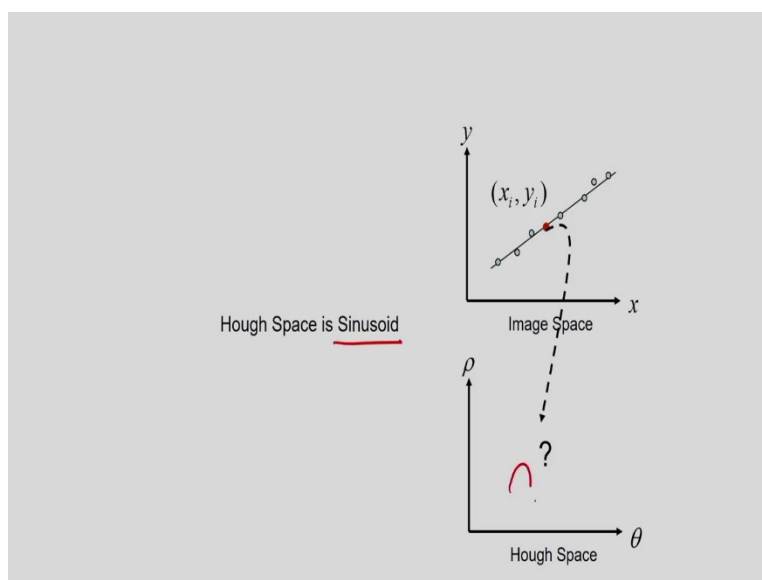
So, we have considered the parametric space, the parametric space is the m and c space. So, we consider the gradient of a line and see the intercept we have considered. But in this case, it is in principle they are unbounded. So, they cannot consider all the situations, suppose, if I consider a vertical line, for vertical line the gradient will be infinite, then it is very difficult to apply this algorithm because m and c will be unbounded for some of the cases.

So, then in this case, instead of considering this m and c space, that is the, the gradient and the intercept, what I can consider, I can consider two parameters that is the parameters of the straight line ρ and the θ , because equation of a straight line is I can write like this $x \cos \theta + y \sin \theta = \rho$. So, this is the equation of a straight line. So, instead of considering $y = m x + c$, I can consider this equation, the equation of the straight-line $x \cos \theta + y \sin \theta = \rho$ that I can consider.

And corresponding to this equation, I have two parameters one is the ρ , another one is θ . Then in this case the θ varies between minus 90 degree to plus 90 degree and this ρ varies from 0 to $\sqrt{M^2 + N^2}$. So, in this case the size of the image is $M \times N$. So, that means, because m and c are unbounded for some of the cases, so, we are considering the parametric space, the parametric space is ρ and θ now, instead of considering m and c .

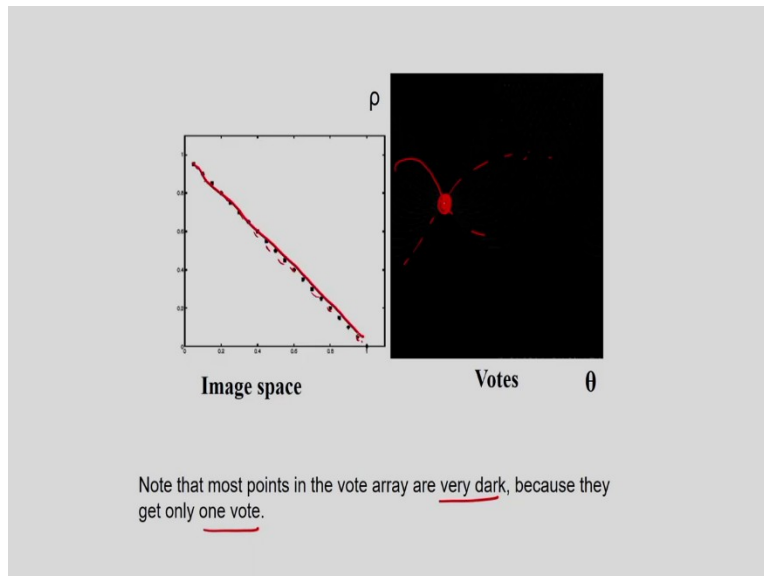
I have shown it is in the x, y space, image space and this you can see this is θ and this is ρ , this is the parametric space. In the parametric space I will be getting sinusoid because, you can see the equation it is equation is $x \cos \theta + y \sin \theta = \rho$. So, I will be getting sinusoid in the parametric space.

(Refer Slide Time: 19:23)



So, here you can see corresponding to this point what I will be getting in the parametric space. So, that means, the Hough space is the sinusoid. So, I will be getting a sinusoid here in the parametric space. So, Hough space is sinusoid.

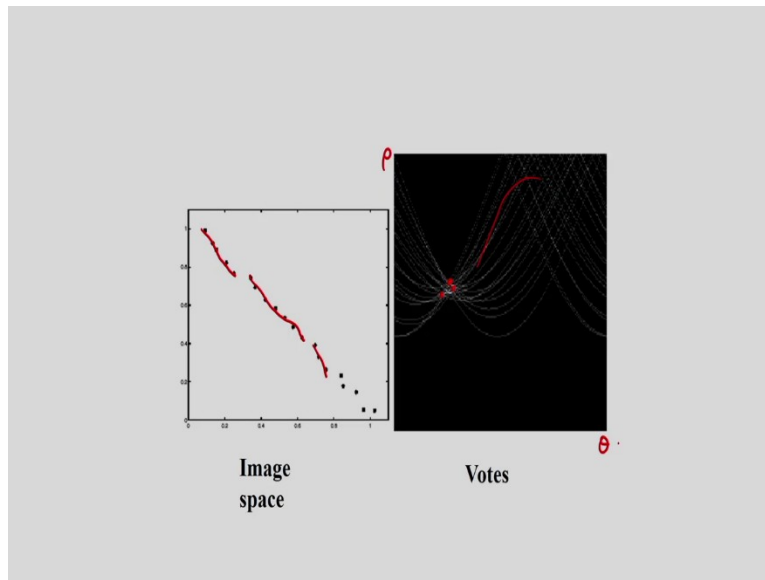
(Refer Slide Time: 19:40)



So, in this case you can see I am considering these points, these points actually, I can form a line, this is the collinear points, all the points are collinear corresponding to a particular line. So, in this case all these points will give the maximum vote at this point, but rest of the point you can see it is almost dark. Why it is dark? You can see the most point in the vote array are very dark, because they get only one vote.

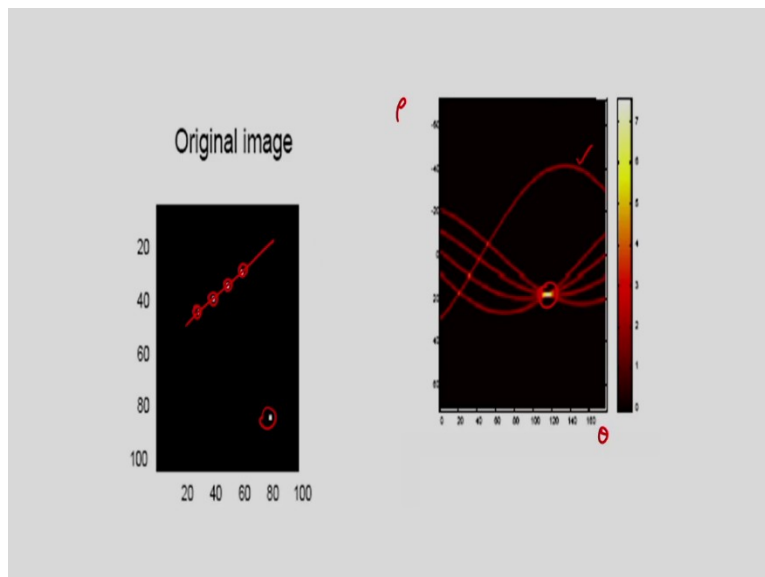
Because the maximum vote I am getting because these points are collinear points. So, other points if you see because I am getting some sinusoid like this, it is not visible here, but the maximum the vote I am getting corresponding to this point, that is the rho theta value corresponding to this line if I get a line by joining these points.

(Refer Slide Time: 20:35)



In this case, you can see I am considering the image space, you can see the parametric space, the parametric space is the rho and theta space. And you can see the votes corresponding to suppose if I consider this line, I will be getting one vote here, corresponding to these points, I will be getting one line, so, I will be getting another vote. Corresponding to this line, I will be getting another vote. So, like this, I am getting the votes and you can see the sinusoids here, you can these are the sinusoids in the rho theta space, this is the parametric space.

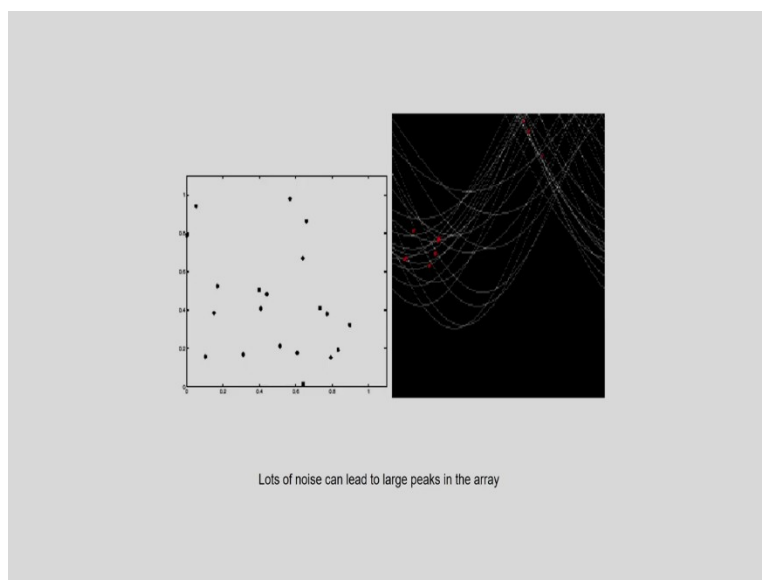
(Refer Slide Time: 21:09)



Here I have shown one example. So, what is this example you can see, I am considering 4 points, which are collinear and also, I am considering one isolated point, this is the isolated point. Corresponding to the isolated point I am getting one sinusoid and corresponding to all 4 collinear points, I am getting 4 sinusoids and they will intersect at this point, they will intersect at this point, but that corresponds to rho theta value of the line joining these points.

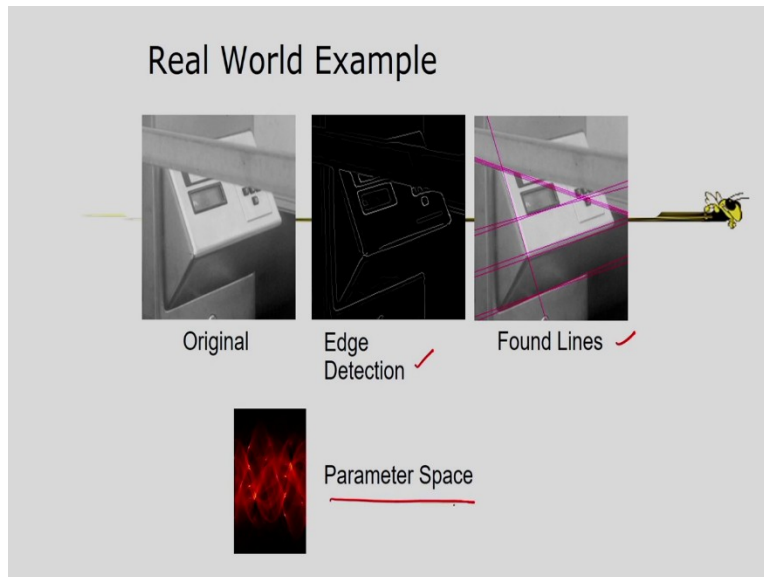
And for the isolated points there is no intersections, that is obvious. That means I am getting maximum vote corresponding to these 4 points, corresponding to these 4 points I am getting 4 votes. And corresponding to this isolated one I am not getting any votes. So, based on this I can consider that means these 4 points are the collinear points and I can join a line, I can get a line from these points.

(Refer Slide Time: 22:11)



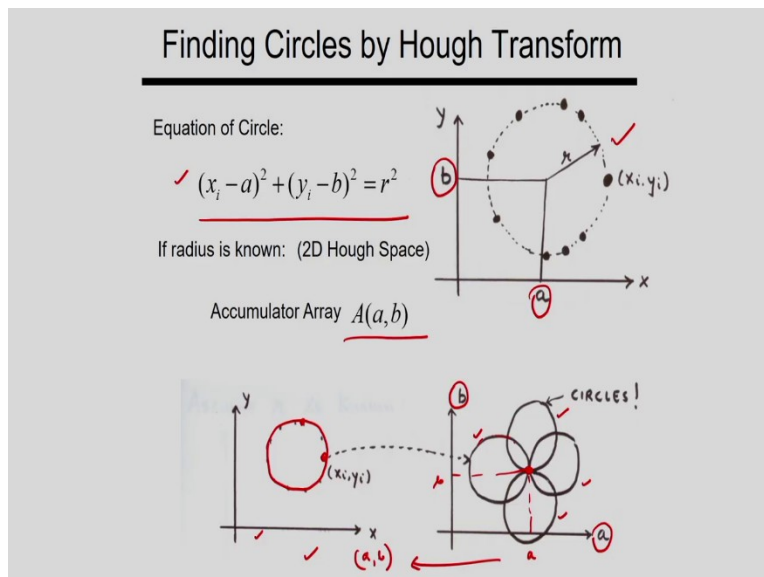
In this case what I am considering the lot of noise we can see, the noisy points and in this case, it is very difficult to find the lines because you can see I am getting votes for different noisy points, the votes are like this. So, in this case noisy points are available. And in this case, it is very difficult to find a maximum value in the accumulator, the accumulator array.

(Refer Slide Time: 22:39)



And in this case, you can see some real world examples, I have shown the original image and you can see that the edge detection technique I am applying like I can apply the Sobel operator or maybe the Prewitt operator or maybe I can apply the log operation that Laplacian of Gaussian. So, this is the edge detection technique I have applied and in the second case, I am applying the Hough transform to find the lines. So, you can see I can determine the lines present in an image and in this case, I have shown that parametric space.

(Refer Slide Time: 23:18)



Now, this Hough transform can be extended for detection of circles present in an image. So, in this case, I can consider the equation of the circle, the equation of the circle is $(x - a)^2 + (y - b)^2 = r^2$. And in this case, if the radius is known, then in this case I can consider the two-dimensional Hough space. So, I have shown the circle here this circle and corresponding to the circle I have the radius, the radius is known suppose.

And the parameters, I have two parameters one is a and another one is b. So, what is the equation of the line? The equation of the line is this is the equation of the line $(x - a)^2 + (y - b)^2 = r^2$. And corresponding to this if the radius is known in my accumulator, I have only two parameters one is a and another one is b. So, in this case you can see I am considering a circle in the second figure, this figure and I have shown the points of the circle.

So, corresponding to suppose this point what I will be getting in the parametric space, the parametric space is now a and b, because I have two parameters of the circles. So, corresponding to this x_i, y_i that point in the x, y space, so, what I will be getting in the parametric space? In the parametric space I will be getting the circle, this circle I am getting. So, this circle I will be getting in the parametric space.

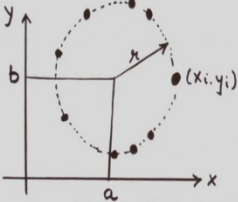
Similarly, corresponding to another point if I consider the point is actually the circle points. Corresponding to the second point suppose I will be getting another circle, like this corresponding to all these points I will be getting the circles like this and these circles will intersect at a particular point, this point, at this point it will be intersecting. That corresponds to a and b value of the circle that I have shown in the figure, this figure that is the corresponding to this circle, I have the a and b value, I have the a and b values.

So, in this case in the parametric space, so, all these circles will be intersecting at this point and that will give the value of this a and b, that is the a and b is this. So, in this case also what I have to consider I have to again consider the number of intersections. So, for this again I have to consider accumulator, so that I can count voting. So, how many times it is intersecting.

(Refer Slide Time: 26:03)

Finding Circles by Hough Transform

Equation of Circle:

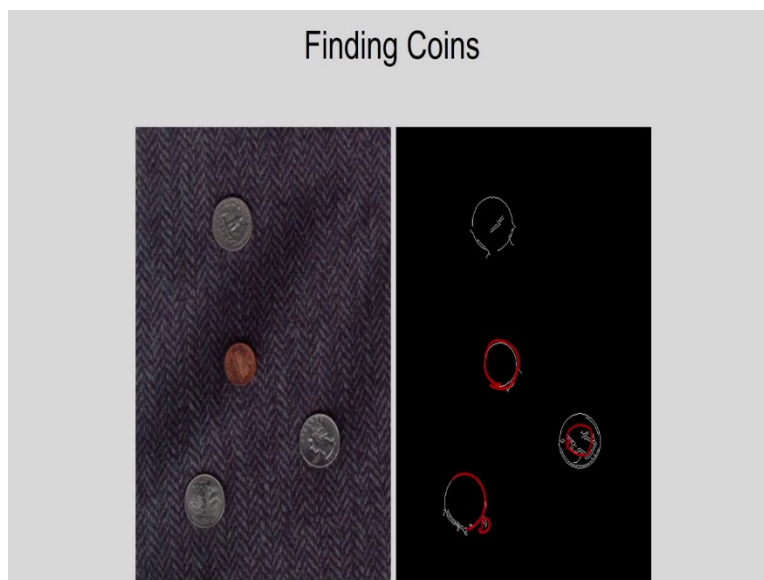
$$(x_i - a)^2 + (y_i - b)^2 = r^2$$


If radius is not known: 3D Hough Space!
Use Accumulator array $A(a, b, r)$

$A(a, b, r)$

So, in this case you can see equation of the circle, suppose the radius is not known, then in this case my accumulator array will be, these parameters one is a, b and r, if the radius is not known. If the radius is known, then that accumulator will be only A is the accumulator and I have the parameters a and b if the radius is known. If radius is not known, then the accumulator will be a, b, r.

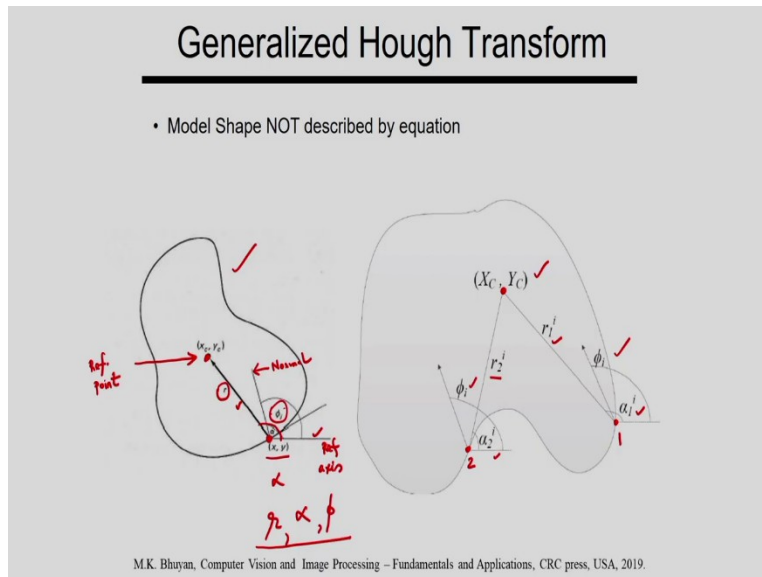
(Refer Slide Time: 26:30)



And in this case, you can see the detection of the coins by using the Hough transform and also I have shown the results here also the edge detection principle also I am applying and in the edge

detection principle, you can see the small, small edges, I can also get by edge detection. But if I apply the Hough transformation, then only I will be getting the circles, I will be getting the circles corresponding to the coins all the coins.

(Refer Slide Time: 26:59)



Now, the next point is I want to discuss about does generalized Hough transform. So, what is the generalized Hough transform? The model shape not described by equation because already I have discussed about how to detect the straight line, I am considering the equation is $y = m x + b$ or may be $x \cos \theta + y \sin \theta = \rho$. Also, I discussed how to detect the circles present in an image. For circle also I have the equation, $(x - a)^2 + (y - b)^2 = r^2$.

But suppose if I consider this object, then in this case I cannot represent by equation, that is the model shape not described by equation. So, for this I can apply the generalized Hough transform. So, what is the method here? So, suppose corresponding to this pixel, that is the edge pixel, the x, y is an edge pixel and I have shown the edge normal, the edge normal you can see, this is the edge normal, that is the normal to the edge.

And I am considering one reference point in the object. So, this is my reference point, this is the reference point X_c, Y_c is the reference point, reference point I am considering, reference point in the object. And corresponding to this reference point you can see one line that is a one vector I am considering from this point that from the edge point to the reference point. And in this case, I am considering this length of this vector is suppose r .

And if you see the angle, what is the angle? The angle in between this line, line is the, this line the line joining between the edge pixel and the reference point and the reference axis I am considering, so, this is my reference axis, reference axis I am considering. So, corresponding to this reference axis, the angle between the reference axis and the line joining the edge point and the reference point is alpha. So, this alpha angle I am considering, so alpha angle I am considering, so, this is my alpha angle.

And I am considering and phi angle, what is the phi angle here? This is my phi angle; the phi angle is the direction of the normal to the edge with respect to the reference axis. So, what are the parameters I am considering here, one is the parameter is r, what is r? r is the distance between the point, the point is the edge point, edge point is the X and Y point. And the reference points the reference point is X_c, Y_c , that is a centroid point that I am considering.

So, the first parameter is r, the second parameter is alpha, what is alpha? alpha is the angle between the reference axis and the line joining the edge pixel and a reference point. So, alpha I am considering, so this is my alpha. And also, another parameter I am considering the phi, what is phi? The phi is the direction of the normal to the edge. So, these parameters I am considering, one is r, α, φ_i . So, these parameters I am considering to represent a particular edge pixel.

In the second picture what I am showing, the same thing I am showing, I am considering two edge pixels here and corresponding to these two edge pixels I am considering suppose, my reference point is this, the reference centroid point of the object is X_c, Y_c and you can see the parameters, what are the parameters; one is the r_1 that is the r I am considering r is the parameter, that is the distance between the point and the reference point, the reference point is X_c, Y_c .

So, I have r_1^i and similarly, corresponding the second point, the second point is supposed 2, this is the first point. So, corresponding the second point my distance is r_2 , that is the distance between the point 2 and the reference point that is r_2 , that is the vector and also I have the angle, the angles already I have defined, one is α^1 , you can see α_1^i and α_2^i and also I have the angles that is the direction of the edge normal that is φ_i and φ_i , that I am considering their phi angle. So, you can understand what are the parameters one is the r, α, φ these are the parameters.

(Refer Slide Time: 31:43)

Generalized Hough Transform

- Model Shape NOT described by equation

ϕ - Table for Generalized Hough Transform.

Edge Direction	$\vec{r} = (r, \alpha)$
ϕ_1	$\vec{r}_1^1, \vec{r}_2^1, \vec{r}_3^1$
ϕ_2	\vec{r}_1^2, \vec{r}_2^2
:	:
ϕ_i	\vec{r}_1^i, \vec{r}_2^i
ϕ_n	\vec{r}_1^n, \vec{r}_2^n

ϕ_1 \vec{r}_1 \vec{r}_2 \vec{r}_3

So, for this model, this technique the generalized Hough transform that is the model shape not described by equation. So, for this what I have to consider I have to make one table that table is called phi table, the phi table for generalized Hough transform. So, in this table you can see I am considering different edge directions like this ϕ_1, ϕ_2 all these directions I am considering. And corresponding to these edge directions you can see the vector, the vector is r , the vector r I am considering and it is two components one is r , r is the distance I have defined what is r and another one is the angle is alpha.

So, in this case you can see corresponding to ϕ_1 I may have r_1, r_2, r_3 because I can get the value r, α corresponding to the angle, the angle is ϕ_1 . For this angle I can have r_1 or maybe r_2 or maybe r_3 . For a particular this angle condition, I can get this vector r_1, r_2, r_3 like this I can get.

And in the vector r what are the components? The component is r , r is the distance and α is the angle. Similarly, corresponding to this angle, the direction of the edge normal I can get r_1 and r_2 like this. So, like this I am showing for edge direction, for different edge directions I can get the vector r this is the phi table for generalized Hough transform.

(Refer Slide Time: 33:16)

Generalized Hough Transform

Find Object Center (x_c, y_c) given edges (x_i, y_i, ϕ_i)

Create Accumulator Array $A(x_c, y_c)$ ✓

Initialize: $A(x_c, y_c) = 0 \quad \forall (x_c, y_c)$

For each edge point (x_i, y_i, ϕ_i) ✓

For each entry \vec{r}_k^i in table, compute:

$$\begin{cases} x_c = x_i + r_k^i \cos \alpha_k^i \\ y_c = y_i + r_k^i \sin \alpha_k^i \end{cases}$$

Increment Accumulator: $A(x_c, y_c) = A(x_c, y_c) + 1$ ✓

Find Local Maxima in $A(x_c, y_c)$ ✓

Generalized Hough Transform

- Model Shape NOT described by equation

ϕ - Table for Generalized Hough Transform.

Edge Direction	$\vec{r} = (r, \alpha)$
$\phi_1 \rightarrow$	$\vec{r}_1^1, \vec{r}_2^1, \vec{r}_3^1$
$\phi_2 \rightarrow$	\vec{r}_1^2, \vec{r}_2^2
\vdots	\vdots
$\phi_i \rightarrow$	\vec{r}_1^i, \vec{r}_2^i
ϕ_n	\vec{r}_1^n, \vec{r}_2^n

$\vec{r}_1 \quad \vec{r}_2 \quad \vec{r}_3$

So, in this case, so what will be the algorithm to detect the object, that is not described by the model equation. I can find the object center based on the centroid. So, suppose the centroid is given. So, this is the object center and also, I have the given is the edges, the edges are x_i, y_i, θ_i . And for this again I have to do the voting. So, for this I can create the accumulator, the accumulator is x_c, y_c these values and I have to initialize the accumulator, so, this is the initialization of the accumulator.

After this for each edge point, so, I can consider edge points are like this x_i, y_i, ϕ_i . So, from the ϕ table, corresponding to ϕ_i , I can find this one, this vector. So, corresponding to this r_k , I can

find x_c and y_c by these equations and after this I can increment the accumulator. And finally, like I described earlier I can find a local maximum of the accumulator, the local maxima in the accumulator I can determine.

So, that means, corresponding to this angle, corresponding to the direction of the edge normal, I can determine this vector r_k from the phi table and after this I have to find a value of x_c and y_c I can find and based on this, I can increment accumulator. So, the process is very simple. So, one thing is only important you can see in my previous slide, so, I have to make the phi table corresponding to these angles.

The angle is basically that ϕ_1, ϕ_2 , these are the angles that means the direction of the, direction of the edge normal. And corresponding to a particular angle there may be two or three vectors, maybe r_1 , maybe r_2 , because for the same angle I can have r_1 or I can have r_2 , because r has the two components, one is the r and other one is α .

(Refer Slide Time: 35:18)

Scale and Rotation

Use accumulator array $A(x_c, y_c, S, \theta)$

$$x_c = x_i + r_k^i S \cos(\alpha_k^i + \theta)$$

$$y_c = y_i + r_k^i S \sin(\alpha_k^i + \theta)$$


$A(x_c, y_c, S, \theta) = A(x_c, y_c, S, \theta) + 1$ ✓

And suppose, if I consider the scaling and the rotation of the object, then in this case what I have to consider, I have to consider the accumulator, the accumulator is already I have considered x_c and y_c parameters I have considered. And if I consider the scaling by a factor of S and a rotation by an angle θ . So, this is the object, it is scaled and it is rotated suppose, so, in this case how to detect this object.

So, corresponding to these I can use these equations, that because I have another two parameters, one is the S , other one is θ . S is just scaling parameter and theta is the rotation parameter and I can find x_c and y_c by these equations and accordingly I can increment the accumulator, I can get the value in the accumulator and I can find a maximum in the accumulator. So, this is for any arbitrary shape.

(Refer Slide Time: 36:12)

Hough Transform: Comments

- Works on Disconnected Edges ✓
- Relatively insensitive to occlusion
- Effective for simple shapes (lines, circles, etc)
- Trade-off between work in Image Space and Parameter Space
- Handling inaccurate edge locations:


And the Hough transform comments. So, this Hough transform works on disconnected edges you can see this example, already I have given some examples like in the coins also it works on disconnected edges. It is relatively insensitive to occlusions that you can see and it is effective for simple shaped like I can detect the lines, I can detect the circles like this and in this case in the in case of the Hough transformation, we can get the image space and we can get the parametric space.

So, I can work in the image space or I can work in the parameter space. So, you can see the mapping from image space to the parametric space. And also, the handling inaccurate edge locations. So, suppose if I apply the edge detection techniques, then in this case I will be getting edge pixels that may not be correct, but I can apply the Hough transform techniques, so that I can get the lines or I can get the circles. So, this, the noisy edge pixels I can neglect. So, these noisy edge pixels I can neglect and I can find a particular line or I can find a particular circle like this.

So, these are the advantages of the Hough transform. So, today I have discussed the concept of line detection and the circle detection by considering the Hough transformation. And also suppose for some objects, which cannot be described by model equations, then in this case, I have to apply generalized Hough transformation. So, this is about the Hough transformation. So, let me stop here today. Thank you.