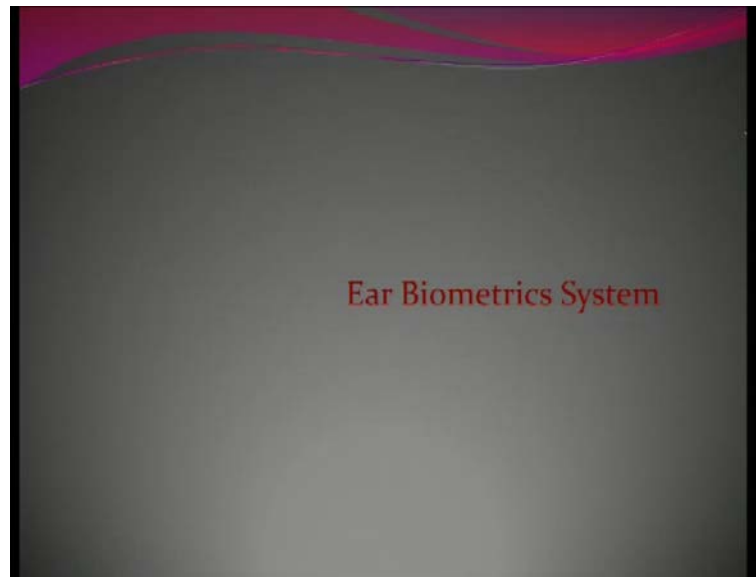


**Biometrics**  
**Prof. Phalguni Gupta**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kanpur**

**Lecture No. # 20**  
**Ear Biometrics System**

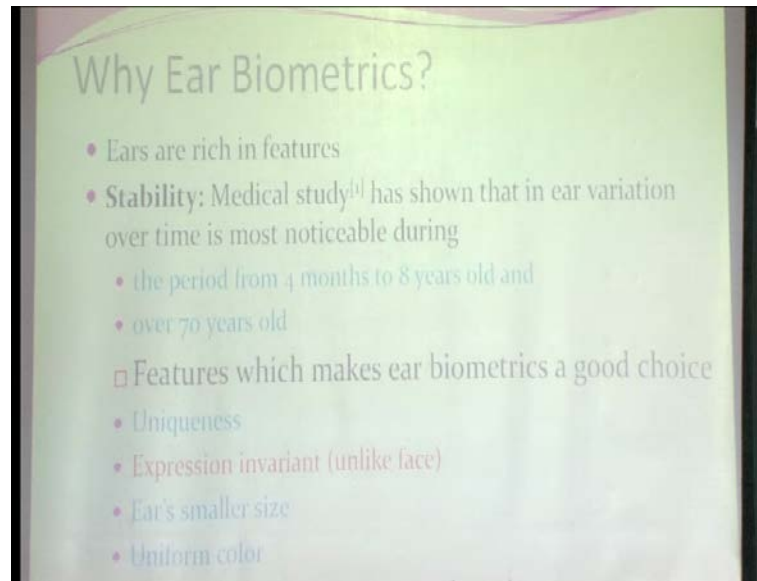
We will start with Ear Biometrics.

(Refer Slide Time: 00:18)



Then, we will go to iris and then indexing. So, face has been covered here **right yes**.

(Refer Slide Time: 00:28)



So, the techniques that we are using for handling the ear images for recognitions are almost same as the face, but just **just** I want to highlight and there are some advantages with the ear as compared to the face.

So, for example, ears are rich in features. So, I mean that is true for any biometrics; the features are not there in any biometrics, you cannot use it **right**. So, there should be some features in any biometric, so that you can distinguish one person from another. So, that is the primary requirement for any biometric, so ears are also having some features that is why we are using them for the recognition.

Another thing is very important thing is stability, so stability like for example, in case of face; we see, as we grow our face changes. So, I mean may be after 2, 3 years we all have different faces, but that is not to be the ear.

So, what medical study has shown that, mostly the changes happen from the years of 4 to 8 years after that, there is no change and then, beyond 70 years; there are some changes in a **(( ))**. Otherwise, in rest of the life we have very similar kind of ear. So, only scale changes, but the features remain the same. So, that is a very good thing with the ear; which is not there with the face, because face changes very rapidly.

Second thing is uniqueness, so everybody has a unique ear that is why we are able to distinguish the people using the ears features. Ears do not have expressions like we have

in the face; when we are laughing, when we are sad, when we are angry we have different appearance on the face, but ear does not have any such expression. So, that is why its better compare to face. Size of the ear is also smaller than the face, so we need to process less data.

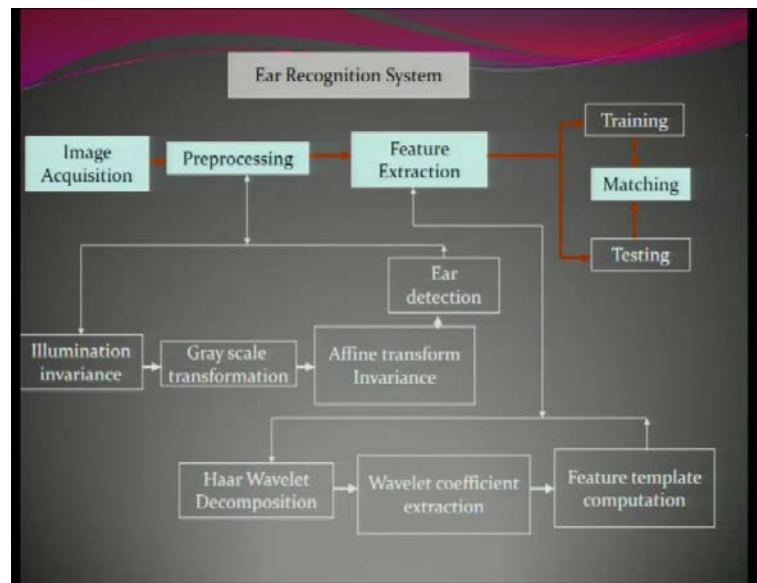
Then, uniform color and the color of the ear is also uniform, which is not the case with the face. So, if we have the beard maybe the person is black or white. Similarly, the eyebrows and so, there are some mixed colors in the face, but that is not true with the ears; ear has the uniform color. It is a passive biometric; passive biometrics means that, the biometrics where we do not need any user cooperation.

So, from the far distance also you can capture the data, same is the case with the face; face also you can capture from the far distance; ear also you can capture from the far distance. So, it is a passive biometrics. And so all this comparisons I am making with respect to face, because usually the **the** way you capture the data for the ear, the way you process the data for the ear is very similar to the face. And so normally like we use CCD camera for capturing the face, we use like ear also for the same.

So, but when we compare this all biometrics it is better than the face. In many studies, it has I mean the performance of the ear has been found better than the face, but it is not the best. So, if you compare with the iris I mean you cannot compare it with the iris; iris has much better performance compared to the ear. But, there are some other problems with the iris like for example, you cannot capture the iris data in a passive way, like you need a very **very** much user cooperative user, and then only you can collect the data.

So, if the situation is situation demands where the **the** user may not be cooperative or the subject may not be the cooperative, then you can use **you use** the biometrics like ear or face something like that. So, if you have the choice you can use ear.

(Refer Slide Time: 04:14)



So, like any other biometric system, there are some features for the ear recognition system also. You have first you have to acquire the image. Then, you have to process the image. Then, you have to extract some features. Then, you have to use in training or in testing.


Training means the enrolment and testing means the comparison or the verification or identification. So, in preprocessing you have some these are the some things we which you need to perform like we have to do the illumination and normalization, then gray scale transformation; If the images in RGB, you have to convert it to grey, if you are working in gray scale. Then, affine transformation; so, if there is any rotation or a scale problem, that you have to remove.

And then, so normally all these things are done on the old profile face, so if you have to capture the whole profile face, then you have to detect the ear and then, you have to use it for the further processing. So, after all these performance steps you have to detect the ears and then, you have to extract the features and then, you have to use for the training or testing. There are some techniques, which are normally used for feature extraction like Haar wavelet is the one technique.

(Refer Slide Time: 05:27)

### Image Acquisition

- The CCD camera is placed on a motorized stand.
- Halogen Light focused on the ear.
- Image captured on CCD camera.



How do we require the ear image? So, in the in the same way as we do for the face, there is a CCD camera and you will illuminate the ear with some Halogen or some other a light source. And then, with the help of CCD camera you capture the data.

So, while capturing the data **you only do not** only collect the ear, you actually collect the whole side face or sometimes you say the profile face. So, many times after capturing the data you have to first locate the your **(( ))** of interest that is the ear.

(Refer Slide Time: 05:33)

### Issues on Image Acquisition

- First:  
Image may be tilted in vertical direction.
  - Caution must be taken to place the head parallel to focal plane.

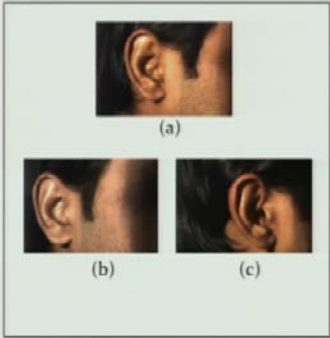
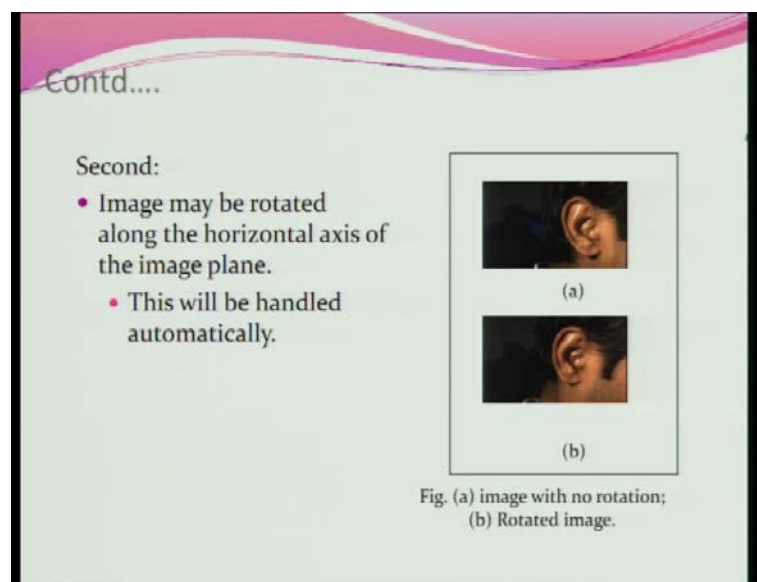


Fig. (a) image in right alignment ;  
(b) (c) titled image

So, what are the issues with the ear? So, very normal thing to notice they maybe rotational problem, so in an any ideal case when you are capturing the data, **your ear** I mean your face should be straight and your camera should be focal plane of the camera should be parallel to use side face **right**.

But, if your face is tilted with the focal plane, then there may be some rotational problems. So, rotation can be in two ways; one with respect to the vertical axis. So, when you are capturing the data, your face may be rotated in this plane or could be like this. So, **this is the problem** I mean these are the images showing the problem, when the side faces rotated with respect to vertical axis.

(Refer Slide Time: 06:36)



Anyways these are the images, when images are rotated with respect to the horizontal axis. So, this is not very common, but the previous one is very common.

(Refer Slide Time: 06:42)

Contd....

Third:

- Image may be occluded by hair.
- Hair should be removed from ear before acquisition or occlusion should be removed automatically in preprocessing.

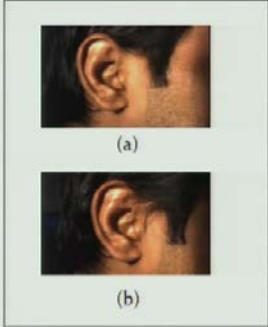
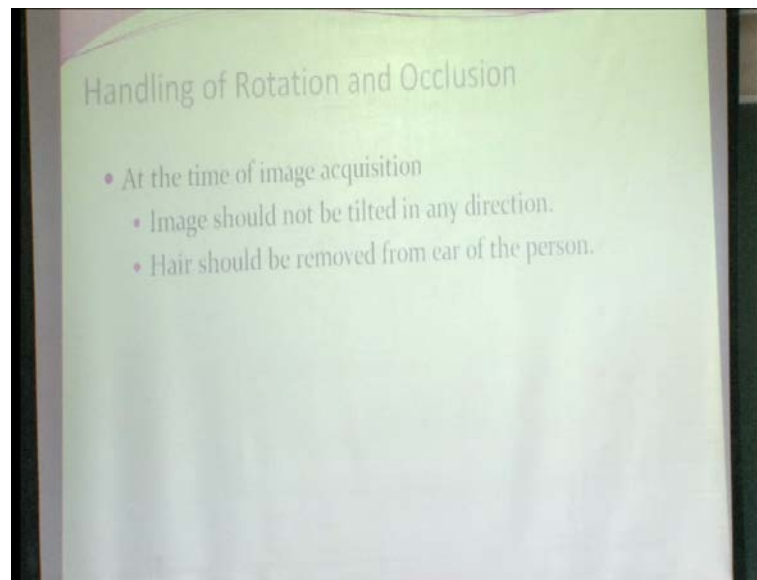


Fig. (a) Clear Image; (b) Occluded image.

There might be some occlusion also, when you are capturing the data. So, the normal occlusion is by the hair. So, the person's ear might be covered with the hair. So, if the environment it is cooperative like for example, if you are user is cooperative then actually you can ask the user to (( )) the ear and then, you can collect the data.

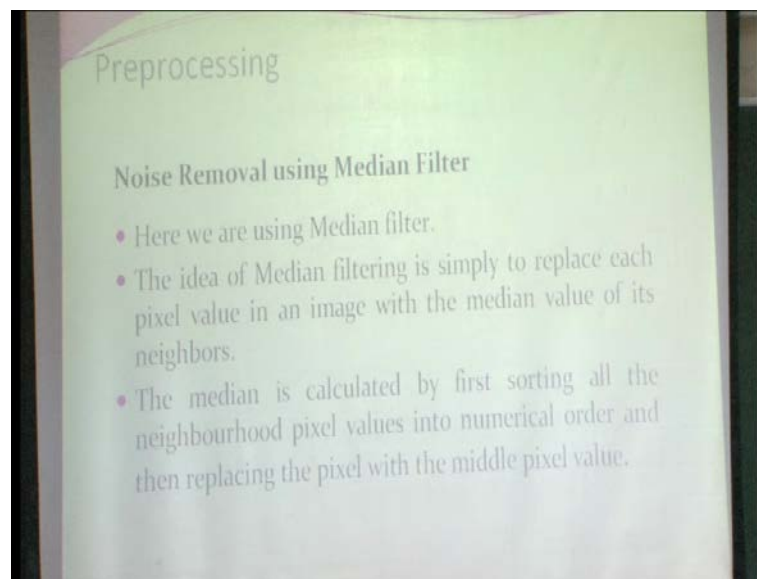
If that is not possible like in the case of passive case, then you have to process whatever you have. **So the user**, so if the user is cooperative actually you can remove the occlusion by requesting the user. Second thing is, if the user is not cooperative, then you should have some way to either exclude the area of a occlusion or some technique to remove the occlusion.

(Refer Slide Time: 07:28)



So, these are the things I just explained.

(Refer Slide Time: 07:30)



Now, what are **these** the things that you need to do for in preprocessing? So, first thing is that, you have to remove the noise. So, there are many techniques to remove the noise like Gaussian filtering, Median filtering is one of the techniques. So, there are many filtering. So, depends in what kind of noise is there in your images, depending upon then you choose the technique **for the** for a noise removal. So, Median filtering is generally used, when the noise is of **(( ))** you know that?



(Refer Slide Time: 08:04)

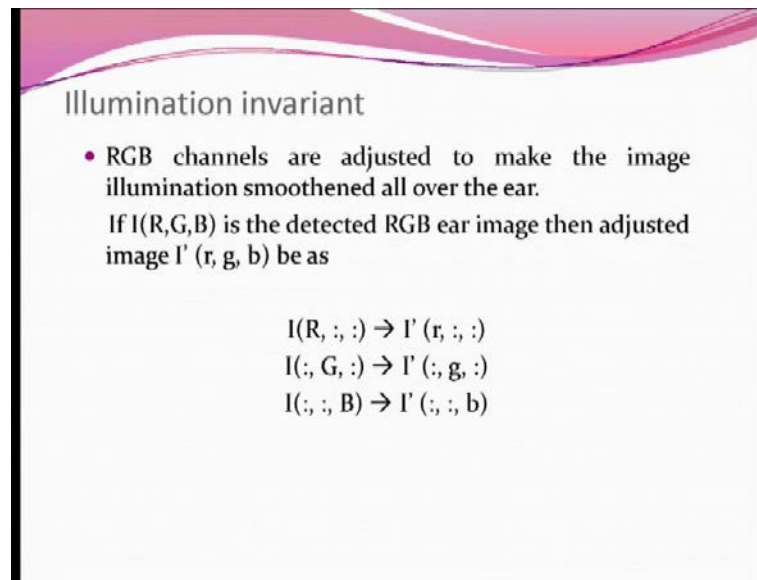


So, in the image if the noise exists like in the form of dots, some spikes. So, for example, when you are capturing the data let us say, the skin is oily or there is some dust or something like that because of that, you may get some spikes in the image. So, that kind of noise you can remove using the Median filtering.

So, in Median filtering, what you do? You around every pixel you calculate a I mean you take a (( )) and then, you take the for the center pixel; you take the value of the median. So, since it is like some random dots; it is assume that, all of the neighbors may not be affected by this noise. So, if you taking the median you may get you may be able to remove the noise.

But if it is suppose the noise is spread all over the regions, then there is not a good technique to use, the use for noise removal then, we use Gaussian (( )) something like that. So, they may be I am just explaining few techniques, there might be some other techniques also for handling these things.

(Refer Slide Time: 09:12)



Illumination invariant

- RGB channels are adjusted to make the image illumination smoothed all over the ear.

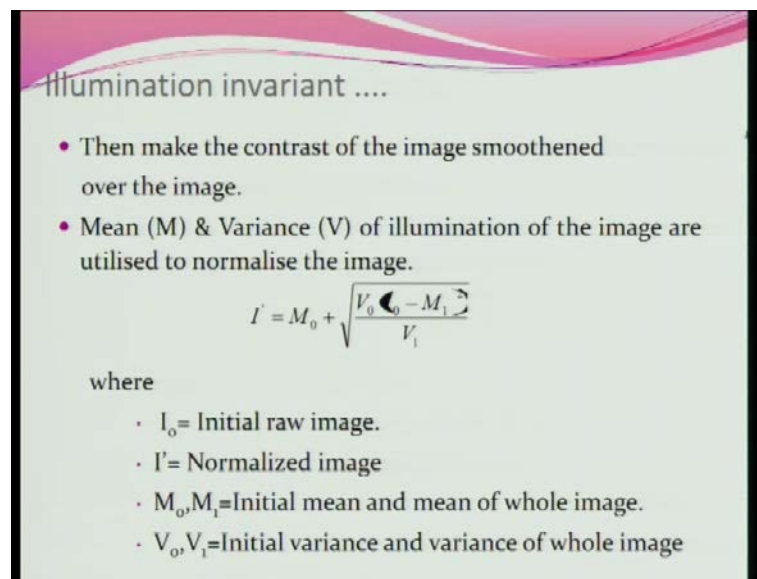
If  $I(R,G,B)$  is the detected RGB ear image then adjusted image  $I'(r, g, b)$  be as

$$I(R, :, :) \rightarrow I'(r, :, :)$$
$$I(:, G, :) \rightarrow I'(:, g, :)$$
$$I(:, :, B) \rightarrow I'(:, :, b)$$

So, illumination is also very common problem because, when you are capturing the data; your image your ear might be illuminated in different way in the different environment, the illumination might be different. So, because of that, the two images might be may look very different, because of the illumination problem. So, you have to remove the illumination problem.

So, if  $I$  is the input image, you actually transforming it to some other domain  $I'$ , where the illumination is removed.

(Refer Slide Time: 09:45)



Illumination invariant ....

- Then make the contrast of the image smoothed over the image.
- Mean ( $M$ ) & Variance ( $V$ ) of illumination of the image are utilised to normalise the image.

$$I' = M_0 + \sqrt{\frac{V_0 (I_0 - M_1)^2}{V_1}}$$

where

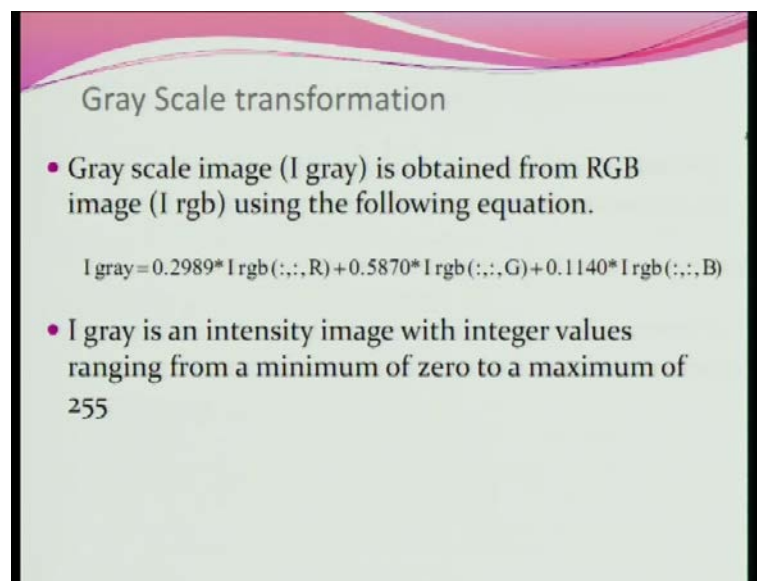
- $I_0$  = Initial raw image.
- $I'$  = Normalized image
- $M_0, M_1$  = Initial mean and mean of whole image.
- $V_0, V_1$  = Initial variance and variance of whole image

So, there are few ways to remove the illumination like for example, this is one way; where actually you have any image you want to have a certain mean and certain variance. Let say, for all the images you want  $M$  should be the mean and  $V$  should be the variance, that is the  $I$  mean let us say that is fixed for all the images than more or less the images will look same.

So, this is one technique, where which converts the given images to an image where the image has  $M$  as the mean and  $V$  as the variance. So, if you can see  $I$  mean this part  $I$  minus  $M$  divided by  $V$ ; actually this makes the variance unity **right** and **you are** you are multiplying with  $V$ . So, basically it will vary within that  $V$  limit.

And depending among the pixel value whether it is greater than  $M$  or less than  $M$  you choose whether you should add that value to  $M$  or you should subtract it to  $M$ . This you might have seen in the fingerprints also **right**. So, this is quite similar.

(Refer Slide Time: 10:49)



Gray Scale transformation

- Gray scale image ( $I_{gray}$ ) is obtained from RGB image ( $I_{rgb}$ ) using the following equation.  
$$I_{gray} = 0.2989 * I_{rgb}(:,:,R) + 0.5870 * I_{rgb}(:,:,G) + 0.1140 * I_{rgb}(:,:,B)$$
- $I_{gray}$  is an intensity image with integer values ranging from a minimum of zero to a maximum of 255

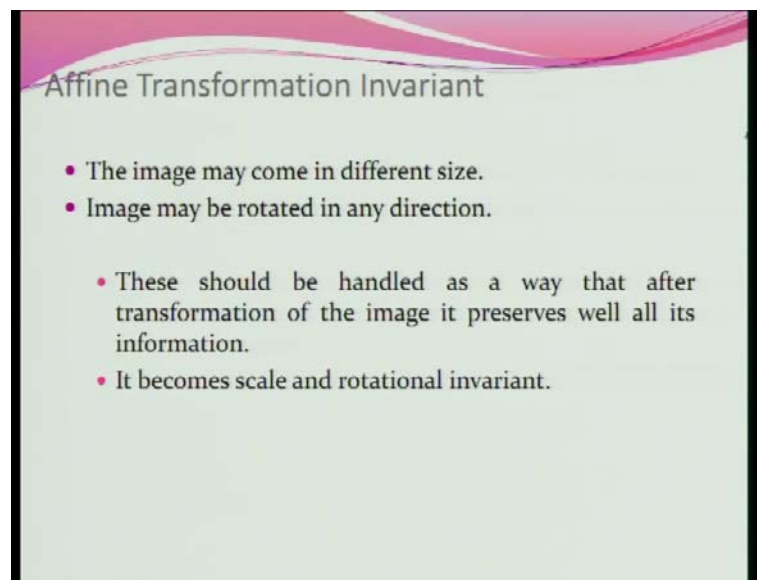
So, in gray scale transformation; given the RGB image you have to transform it to gray scale. So, there are some external free scale **that in** that in R G B; R, G and B channels the relative importance of the channels. So, you can just give some weightage to the channels and then, multiply the pixel value and then just sum it up.

So, many times we give the equal weightage and then so it is like taking the mean; mean of the RGB values. But, this is some NTSC **channel** standard, which says that based on

some experiments **we have** they have formulated that, this R should be given this much weightage, G should be given this much weightage and the B should be given this much weightage.

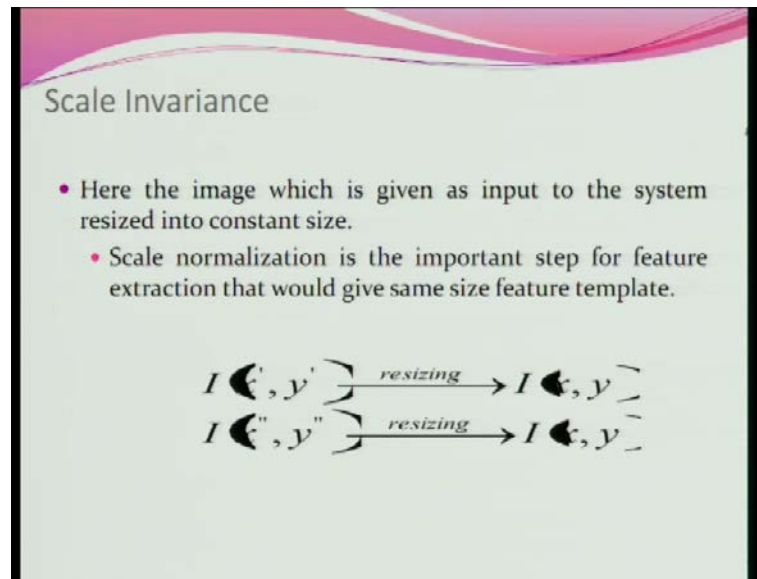
So, you consider these weightage and then, calculate the grey value for **the** that pixel. So, usually this R G B value I mean the pixel value ranges from 0 to 255 for any channel **right**. So, this I gray will be also between 0 **to** 255, because this all the weights are total sum is 1. So, the gray scale image that you get there also the value will range from 0 to 255. So, if there is any scalar rotational problem that you have to remove before using it for the recognition.

(Refer Slide Time: 12:04)



So, normally scale is removed by the resizing it.

(Refer Slide Time: 12:07)



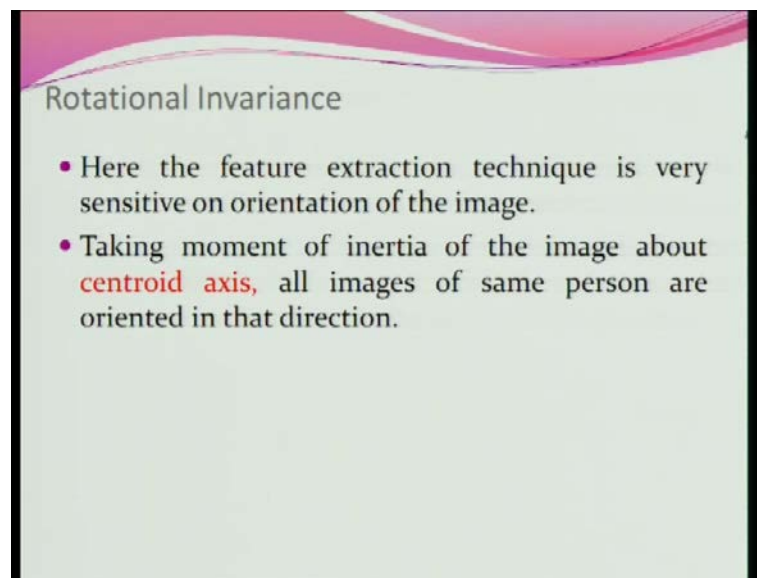
Scale Invariance

- Here the image which is given as input to the system resized into constant size.
- Scale normalization is the important step for feature extraction that would give same size feature template.

$$\begin{array}{l} I(x', y') \xrightarrow{\text{resizing}} I(x, y) \\ I(x'', y'') \xrightarrow{\text{resizing}} I(x, y) \end{array}$$

So, just fixed **the fixed** some size that you are considering let us say, 50 by 50 image. So, for any image you just resize it to 50 by 50. So, usually we do the scale normalization like this.

(Refer Slide Time: 12:23)



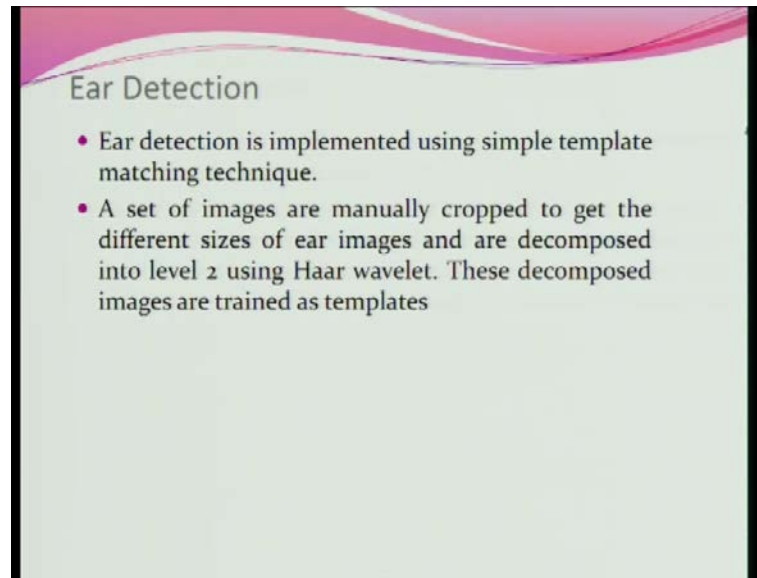
Rotational Invariance

- Here the feature extraction technique is very sensitive on orientation of the image.
- Taking moment of inertia of the image about **centroid axis**, all images of same person are oriented in that direction.

Similarly, for making them rotational invariant; you have to consider some unique axis. For example, what usually we do so suppose something this is a ear. So, we take bottom most part here point here and then, the top most point and then, it join these two point by

the line and then, make to **to** have the rotation invariance just make this as a straight or the I mean rotate it to make this perpendicular line or vertical line. So, that way you can remove the rotational problem.

(Refer Slide Time: 13:01)



Actually, you have captures the whole side face. Now, we have preprocess you have done this preprocessing things. Now, you have to detect the ROI, the Region of Interest or the ear. So, actually you have the, this side face image and here you have to just locate the ear portion.

Once ear is located **yes**

**(( ))**

So, this is actually to handle in plane rotations and that is very and that is mostly I mean there in the images, this kind of tilde I mean you have to ask to the user to give the data properly, otherwise you cannot. Because in that case, you do not have the information **right** like for this kind of tilde; you can just rotate the image, but for this kind of tilde; maybe some portions are hidden I mean you cannot produce them **right**.

So, I mean to handle that kind of problem actually then we move to 3D, because the 3D you have the complete geometry; the 3D x y z data, then you can just rotate it. In 3D you can have rotation with respect to x or y or z any axis. So, that kind of post problems you cannot handle actually in this case.

So, for **for** detecting the ear; there are many techniques, so the template matching is the easiest one and the **(( ))** initially, so what we do here is that, there is a ear template and there is a given image I, there is a template. Now, you move this template over this image and you find out the places where this the underlying images similar to the template and then that way you detect the ear.

But, there are some issues with this kind of technique. For example, it cannot handle the rotational problem, if the image is rotated with respect to template; you may not be able to detect. Similarly, to handle this scale also, you should have the template of the variance scales; which may not be always feasible because, it may exist in any size. So, you cannot **you cannot** have infinite templates.

So, there are some issues, but this is the common technique **which is** which was proposed on a very early stage. Now, there are other very efficient techniques even among our group also. So, in this technique **what we** one thing is that, we do the template matching and the second thing is that, in place of doing the template matching on the input image; We decompose it to some certain level using wavelet that you know right, using half wavelet you can decompose the image **right**.

So, if the input is I, so you get out **out** of I; if you do one level of decomposition you get four images **right**. So, among them; there is one approximation, horizontal details, vertical details and then diagonal details. So, then approximation can be again decomposed. So, if you decomposed to the two levels actually what we get is, the image; which has the more significant information.

So that is why in place of performing the template matching at this level; you decompose it to certain level and then, you perform the detection at that level. So, if let say if you are performing this ear detection at level two, then template also you have to decompose up to level two and then, you have to use it for matching.

(Refer Slide Time: 16:44)

Contd..

- Input raw image is also decomposed into level 2 using same technique.
- Each template is retrieved from the trained set and matched with the same sized overlapping block of the decomposed input image.
- Thus for each trained template the best matched block in input image is traced.
- Among them the best matched block is chosen and the corresponding region from the original image is extracted

So, this is all I have explained.

(Refer Slide Time: 16:46)

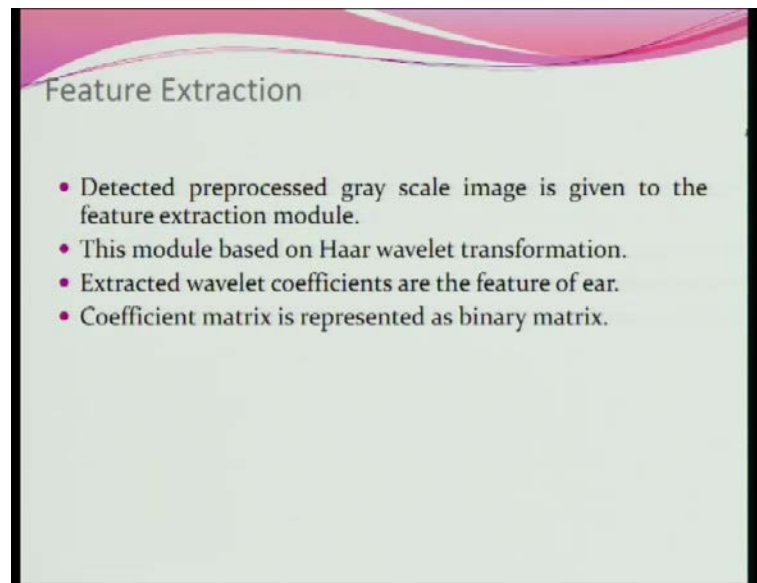
Contd..

Fig (a) Raw image, (c) Cropped image

So, this is a input image based on the template you have detected the ear and then, you have located the ear and cropped it. Now, for the further processing you have to use this part.



(Refer Slide Time: 17:00)



For feature extractions; there are numerous techniques. So, I am just highlighting Haar wavelet, because I think you have used it for **for** face also **right**. So, Haar is the one technique using the Haar wavelets you get the decomposed images and then you **yes**.

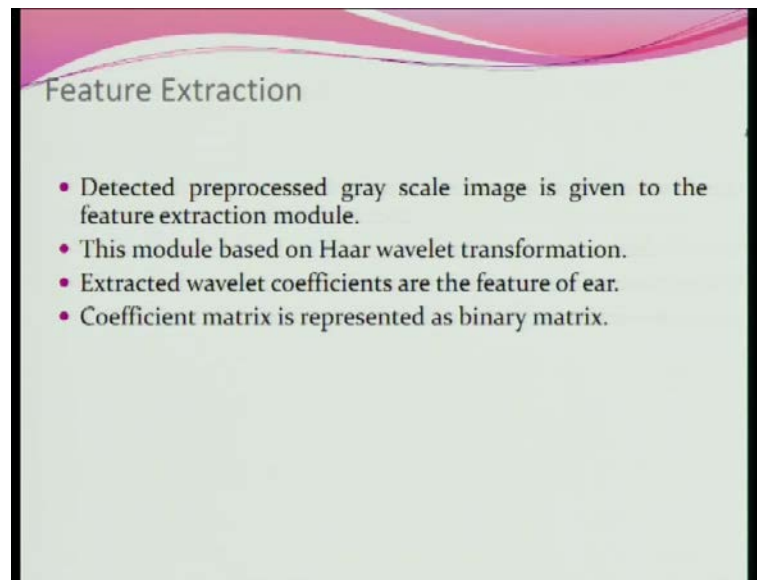
**(( ))**

This template matching point ear so you have the ear template. Now, you are moving that ear template over this image. And let say, using a correlation you find out which portion is similar how much, you have template you are moving the template and at every place you are computing the similarity; the place where the template I mean the underlying images is more similar to the template; you say that, that is a ear got it, because your template is a ear **right**.

**(( ))**

So, some training images some samples of the ear we have. For some training images we have cropped the ear and **that** that those ears we are using as a template. So, those ears we have let us say, from ten samples we have cropped the ear, those ears we are using as a other template.

(Refer Slide Time: 18:20)

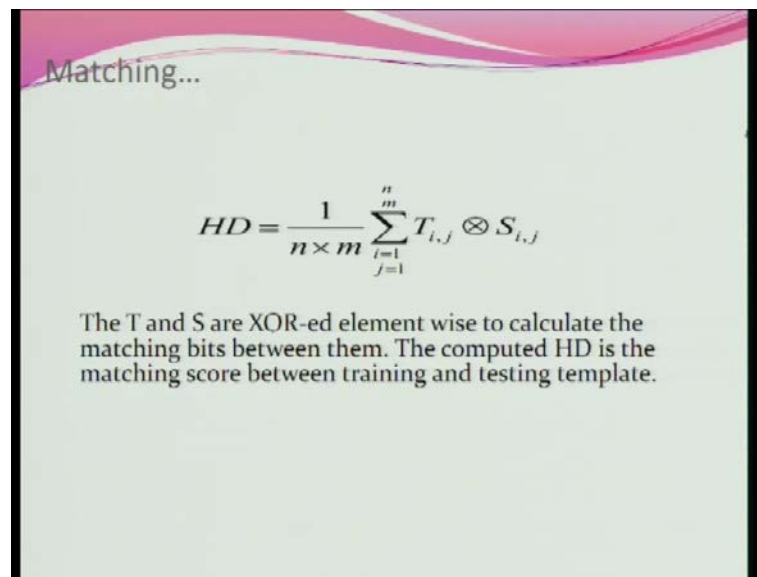


Feature Extraction

- Detected preprocessed gray scale image is given to the feature extraction module.
- This module based on Haar wavelet transformation.
- Extracted wavelet coefficients are the feature of ear.
- Coefficient matrix is represented as binary matrix.

So, using the Haar wavelet you can extract the features. So, basically when you do the decomposition; you get some gray scale values. And based on some thresholding you can convert those gray scale images to binary images, I think you have done it for the face.

(Refer Slide Time: 18:39)



Matching...

$$HD = \frac{1}{n \times m} \sum_{j=1}^m T_{i,j} \otimes S_{i,j}$$

The T and S are XOR-ed element wise to calculate the matching bits between them. The computed HD is the matching score between training and testing template.

So, once you have the extracted features in the form of binary vectors then, you can use the hamming distance for comparing them **right**. And just to normalize the value you are just dividing it by the number of pixels in the image, that gives you the similarity value

or what should I say, (( )) value right and based on that, you can take the decision. So, there is some threshold when you are matching to the images, there is one threshold based on that, you you decide whether to take or not to take I mean whether the two images which you are matching are matched or not matched. So, this is all about ear. So, it does not cover actually the everything in the ear, but it gives you some idea right.

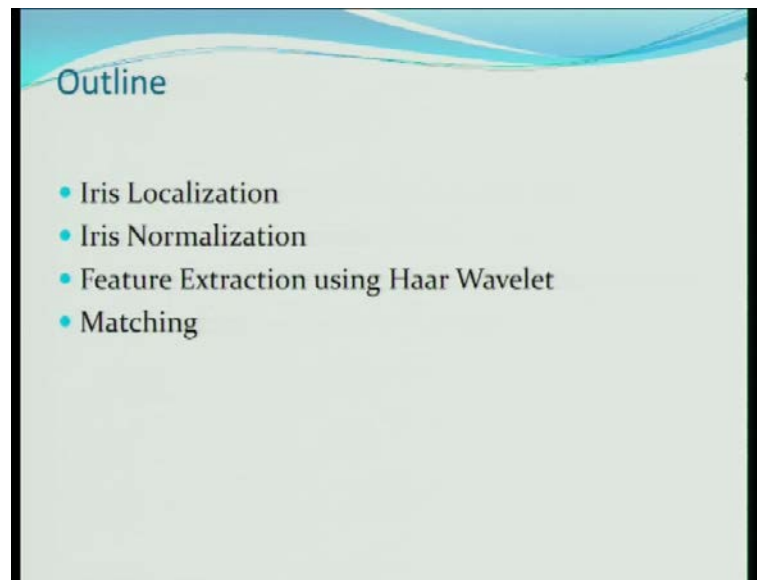
So, for example, for feature extraction also, there are very advanced techniques, for ear detection; there are some advanced techniques. So, just the idea to in this class is to cover the basics. So, that is why we have seen some basic techniques, but if you are interested you can actually go to the go through either to our website; we have some literature available for all of the (( )), you can find some literature for ear also and you can actually Google out also.

(Refer Slide Time: 19:49)



Now, we will see the iris.

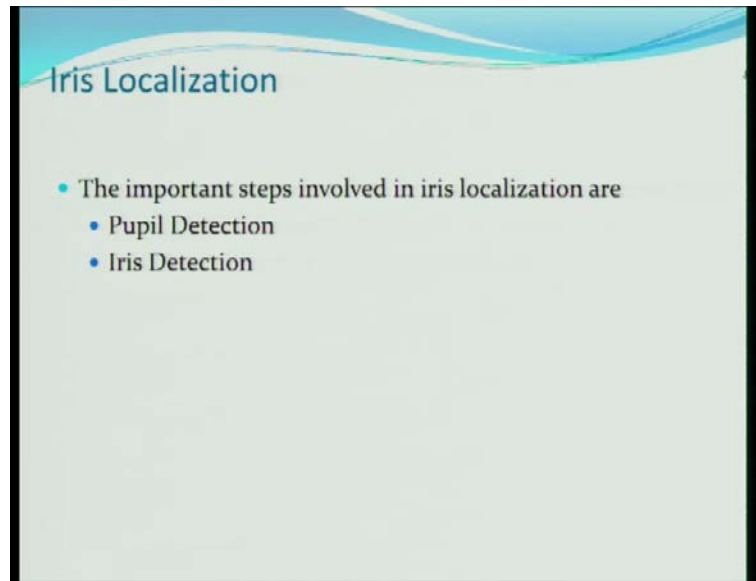
(Refer Slide Time: 19:53)



So, in iris also we have some similar steps to follow like given the iris image; you have to first locate the iris, because when you are capturing the iris image, you may not be only capturing the iris; **you will capture** you will capture some surrounding area also. So, in that image you have to first locate the iris.

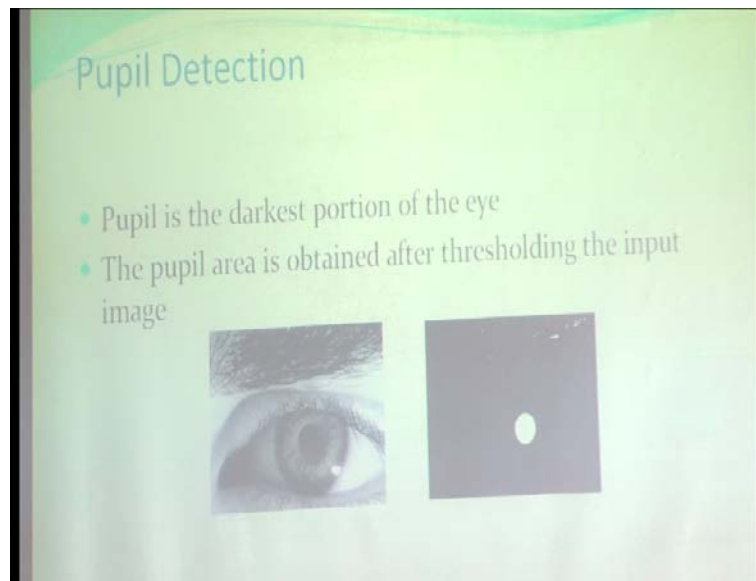
Then, you have to do some normalization to handle the scale and rotation problems. Then, you have to extract features again you can use a Haar wavelet or some other techniques and then, you perform the matching. So, these are the steps to detect the iris in the given image.

(Refer Slide Time: 20:27)



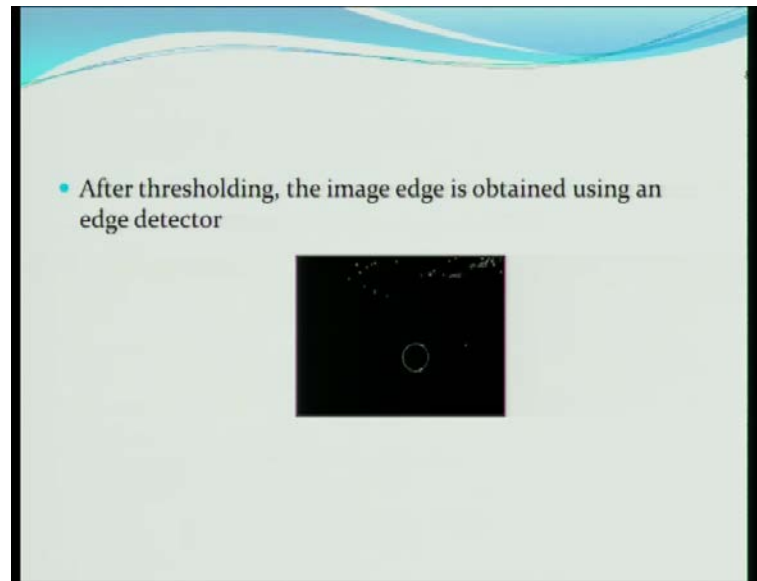
So, first we detect the pupil and then, the iris.

(Refer Slide Time: 20:34)



So, in this image you know the this is the pupil right and this surrounding area is iris, this area (Refer Slide Time: 20:35). And in the eye image actually this the pupil part is the most dark part right. If you see the gray values; you will find that, the lowest grey values are at this place. So, you can do the simple thresholding and you can find out the pupil. So, for example, if you threshold this image; you will get this image got it. Now, it has some noise also here.

(Refer Slide Time: 21:10)



So, you do the edge detection; you get the pupil boundary, but with that boundary actually you get some noise also. So, you have to detect the place where this that pupil boundary exist, because if suppose nothing would have been here, then you can say the all the white pixels are the pupil boundary, but you have some noise. So, you have to detect the place, where this boundary is there or in other way; you can say, you have to detect the circle, because all other noises will be of some other (( )), they will not be circular.

But, the iris the pupil boundary will be circular. So, what you can do you in this in this given image; you can perform the circle detection, you can detect where the circle exist and then that will be the pupil boundary.

So, do you know about the half transform? So, half transform is to detect I mean there is a half transform to detect lines, there is a half transform to detect circles. So, if you use the half transform in this image; you can find out the places where circle exist or to define any circle you need radius and centre. So, you can find out the radius and the center of the existing circle.

(Refer Slide Time: 22:27)



So, for example in case of, for line you write this equation as  $y$  is equal to  $m x$  plus  $c$  right. So, if  $m$  and  $c$  is given, you can draw the line right, you can actually it in a other way also. You can define the line in terms of take the  $x$   $y$  as the fixed value and  $m$   $c$  as the variant.

So, because in case of, when you are detecting the lines for example, you have  $x$   $y$  values and you want to detect the  $m$  and  $c$ , so what we do? But we are given usually I mean  $m$  and  $c$  are fixed,  $x$  and  $y$  are the variable right. Now, here we take  $x$  and  $y$  as the fixed value and  $m$  and  $c$  as the variable. So, for any given  $x$  and  $y$ ; you can plot a line.

So, if you do that, you will find that there is one line like this and let us say this is a this is  $x$ , this is  $y$  and in  $m$  and  $c$  plane; your  $x$   $y$  are the fixed right (Refer Slide Time: 23:43). So, for any given  $x$   $y$ , you can draw the line  $m$   $c$ . So, you will find that if you draw these lines here, you will find that all these lines will have cut at one place. And that is actually the  $m$  and  $c$  value for this, this line. So, if.

So, what we do for we have this  $m x$   $y$  is equal to  $m x$  plus  $c$  equation. For every value of  $x$  comma  $y$ ; we draw a line here and this is basically a  $m$   $c$  grid (No audio from 24:33 to 24:42). So, it is a grid like  $x$   $y$  plane,  $m$   $c$  plane. Now, from wherever this line is passing we vote that location, every location from where this  $m$   $c$  is passing we give a vote, because that shows the probable value of  $m$  and  $c$  for which that particular point will lie on that line.

So, let us first understand the procedure. So, for given any this is the line take 1 x y put it here and then, draw a line there and from wherever this line is passing to that cell you add plus 1. Now, you do for all these points, you plot the line here, you will find that the place where all this lines are cutting that particular cell will have the more votes.

If you consider this all plus 1 has the votes that location will have the most number of votes correct. So, actually that will tell you, so given this x y locations you plot this here and then, you see which location has the most votes that particular m and c will tell you the existence of the line for that m c value. If two lines are existing, you will find that in this m c grid, there will two locations which have the significant number of votes. So, that way we detect the lines, this is the half transform of the lines.

(Refer Slide Time: 26:17)

• For every edge pixel (p) find the candidate center point using

$$x_t = x_p - r \times \cos(\theta)$$

$$y_t = y_p - r \times \sin(\theta)$$

where  $x_p$  and  $y_p$  is the location of edge point p  
 $r \in [r_{\min} r_{\max}]$   
 $x_t$  and  $y_t$  is the determined circle center

Similar this is a half transform **from the** for the circle. So, circle equations we can write in this way in the parametric form, where  $x_t$  and  $y_t$  you have to determine; because to determine any circle you need  $x$  center and the radius. So,  $x_t$ ,  $y_t$  these are the center points and  $r$  is the radius.

So, this is the parametric equation for the **(( ))** for the circle **right**. So, for any given point  $x_p$ ,  $y_p$ ;  $x_p$ ,  $y_p$  is the pixel location. For any given pixel location you have to find out all the probable circles. For examples here for given any  $x$ ,  $y$ , you try to find out all possible  $m$  and  $c$  values for **(( ))** this point will lie on the line.



Similarly, for the circle also for given point  $p$ . So, for example, this is one point here; from this point they may be many circles which will pass. For example, like one circle like this, another circle may be like this, another circle maybe like this, there are many circles which will pass from one point. And they might be various  $((r))$  for multiple set of radius values, circle will pass from this location and from multiple set off center value also the circle will pass from that location.

So, how many variables you have here, three variable;  $x$ ,  $y$ ,  $r$  and  $t$  are the values that you need to find out. Like in this case of 2D, you have the grid  $m$  and  $c$  similarly; in this case of circle we have three values;  $x$ ,  $y$ ,  $r$  and  $t$ . So, in place of having this 2D grid we have 3D **3D** grid, where the every cell represent this  $1 \times y$  and  $x$ ,  $y$ ,  $t$  and  $1$   $r$  value. Now, one thing you have to notice like here we have limited size of grid.

You may say what should be the size of the  $m$  and this grid, because if you say if you take a very narrow grid, maybe you are not being able to include this line in the grid. So, based on some experiment actually you have to find out what should be the sufficient value.

So, similarly here also I mean experimentally find out, what should be the size of the  $r$ ,  $x$ ,  $y$ ,  $t$ ,  $r$  that space and then you use it for this procedure. Now, any given  $x$ ,  $y$ ; what are the thing you are given;  $x$ ,  $y$ ?  $x$ ,  $y$  are the image pixels.

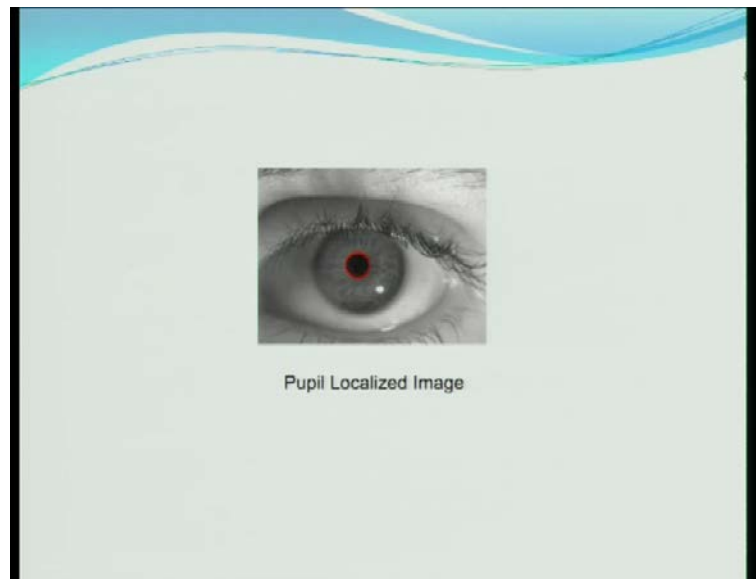
Let us say, this is the circle exist let say some other point are here. So, this is one pixel, there is another pixel, these pixel are given to you, so **these are all your** these are all your  $x$ ,  $y$ . For given any  $x$ ,  $y$  and range of radius from this space where you are detecting your circle you can find out, what should be the maximum range of the radius. So, there is range of radius. So, for all those possible radius values, you can plot circles.

So, let us say for a given radius  $r$  and given point  $x$ ,  $y$ ; you can plot a circle. And how do you I mean that will give you one point and if you vary your theta from  $0$  to  $2\pi$ . So, it will put a complete circle there getting. So, that way you have to put circles for all possible radius for a given  $x$ ,  $y$ ; you take all range all possible values of the radius and then put circle.

And so you will get the value  $x$ ,  $t$ ,  $y$ ,  $t$  and  $r$  right whatever the pixels you are putting that will be for for that or for a given  $r$ ; you will get  $(( ))$ . So, if you have the grid  $x$ ,  $t$ ,  $y$ ,  $t$ ,  $r$  are sometimes we say the accumulator (Refer Slide Time: 30:51).

So, as you are voting for  $m$  and  $c$ ; you can vote for  $x$ ,  $t$ ,  $y$ ,  $t$  and  $r$ . So, in the grid in the 2D grid wherever that circle or line has existed; the value of  $m$  and  $c$  for which the line has existed, you got the most number of votes. Similarly, if you do this procedure; you will get in that accumulator the place where the circle exist for that particular  $(( ))$  center and radius value; you will get the maximum number of votes. So, based on some thresholding you can detect the place of the circle is it clear.

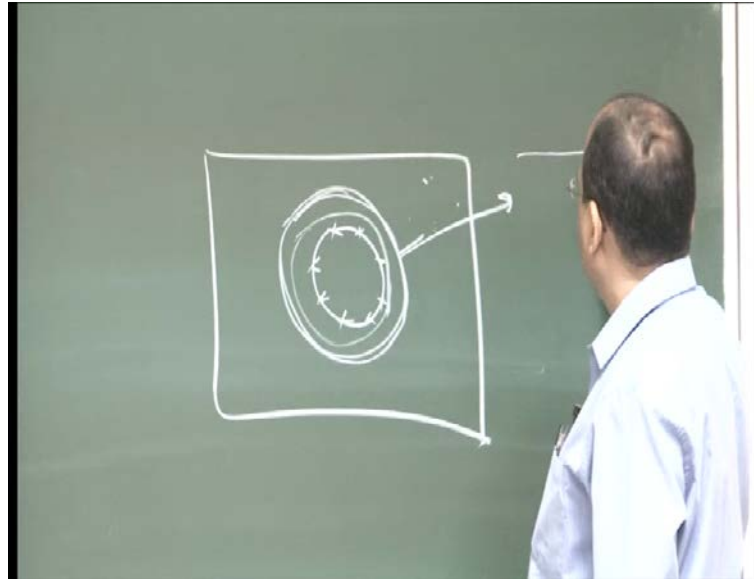
(Refer Slide Time: 31:26)



$(( ))$

$x$ ,  $p$ ,  $y$ ,  $p$  are given here; these are the value where you want to detect the circle. So,  $x$ ,  $p$  and  $y$ ,  $p$  are all your  $(( ))$  and radius you have to I mean find out from your size of the image, some range you have to take. So, once you have detected the pupil boundary, now if you move from this pupil boundary outside; you will find that if you take the concentric circles; you will find that when you are moving from this place in this direction, you will find the that on these circles on the circumference of those circles some  $(( ))$  exist. So, let us say this is your pupil.

(Refer Slide Time: 32:15)



Now, you take another circle here. So, this whole circle is lying on a textural plain **right** similarly, you take another circle; this is also lying on a textural surface. Now, if you keep on **(( ))** the concentric circles like this, there will be one place where the flat region exist, there is no texture.

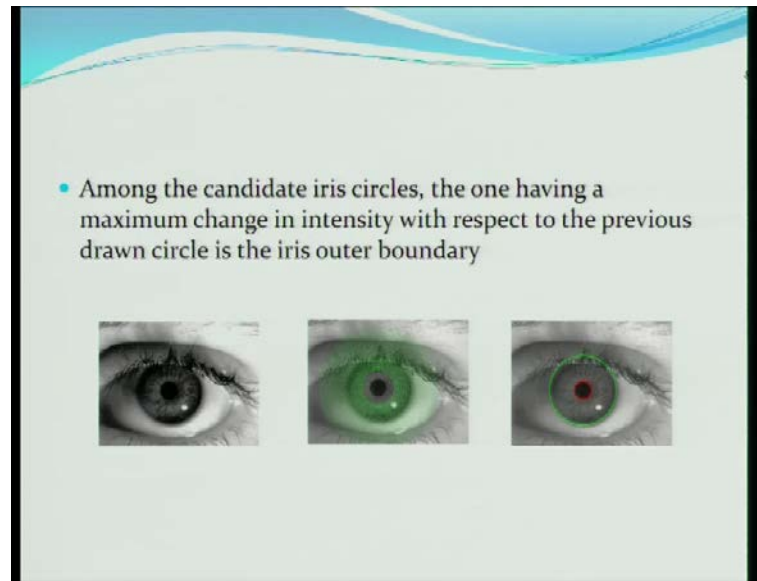
So, when you are moving outside till you are on the iris **you will** you will get the textural **(( ))** when you move out of the iris region; you will get the flat **(( ))**. So, that way you detect the outer boundary. So, keep on drying the concentric circles like this and you sum the pixel values for all the pixel values lying under the circumference.

And when you go to the flat region here you will get some **some** value of the I mean some good amount of some value, but when you go to the flat region; there all values are flat **right**. So, there is the value will suddenly go down.

So, for example, when you are moving in the texture you maybe **you will be** getting the total sum as like this, but when you go to the flat region; your value will drop. So, that place will tell you the outer boundary.

**(( ))**

(Refer Slide Time: 33:36)



Centre of the pupil is just **center of the pupil is the** center of the iris I mean your iris exist around the pupil **right**. Considering pupil as a circular is sometimes debated where it is elliptical or something like that, but the center of the iris is the pupil I mean that is true, but sometimes it may not be perfectly circular.

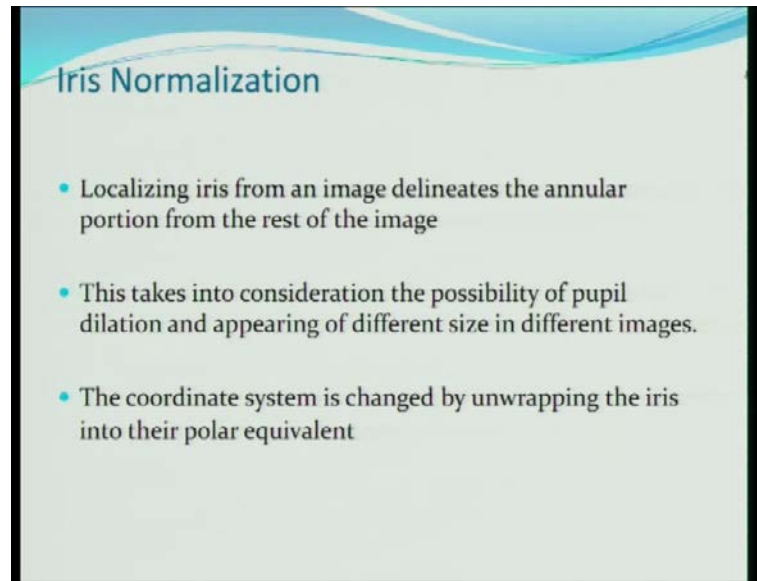
**(( ))**

It is elliptical, but it is the iris is exit around it **right** I mean like even though let say the case, but what you were saying is that, let say this is the pupil some part is visible and then iris is something somewhere here, you do not have this part. But in this case also iris exist around it **right**.

So, you should have some ways like for example, if you are detecting with the help of this technique may be you have to exclude this part using some technique. So, that is the I mean, that is there you have to use the technique for **(( ))** removal. So, this is the demonstration of this technique.

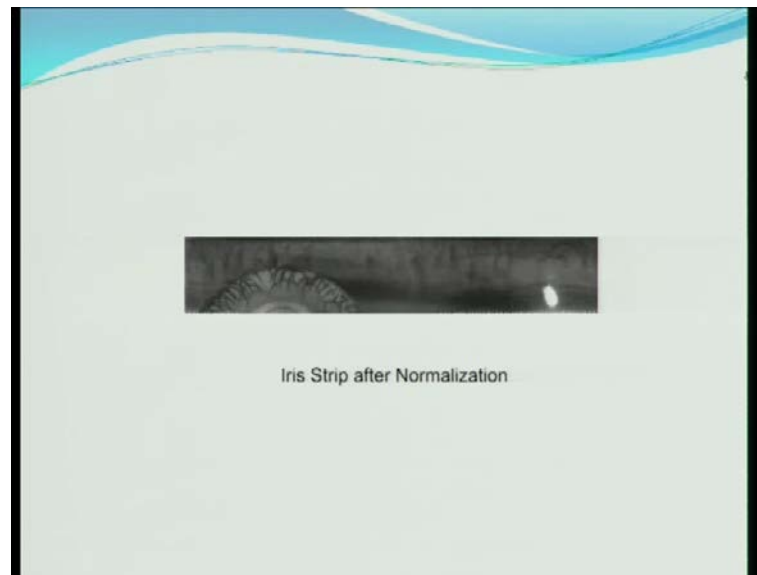
So, you are actually plotting the concentric circles and then summing the values of all the all the gray scale information and then, wherever you find it drop; that you consider the outer boundary is it clear.

(Refer Slide Time: 35:12)



Now, you should have some technique for the size normalization. So, that may be based on the distance or the way you have opened your eyes maybe, if you have opened your eyes very clearly; maybe the **size is** size of the iris is big. If you have not open the fully your eye, size may be less. So, you should have some technique to normalize the size.

(Refer Slide Time: 35:33)



And once you have done this, so there is two ways to process this image; one is that you extract the circular portion and you use it as it is the circular form or many times it is unwrapped and this strip is made circular like this. So, you just cut it, this is like this you

cut it at one place and then open it. So, maybe let us say we can say that, from the polar coordinate you have to transform it to Cartesian coordinate. Now, you see this is what is this part, this part is your, so this is actually not actually the iris part, some occluded portion. So, that we have to remove.

So, in mostly when we process this image we exclude this part or some times for example, this portion is not this iris is open from this side, it is only occluded from topside, maybe sometimes you have some occlusion from the bottom side also. So, mostly when we process the iris image; we only consider this part and this part, because that is mostly available.

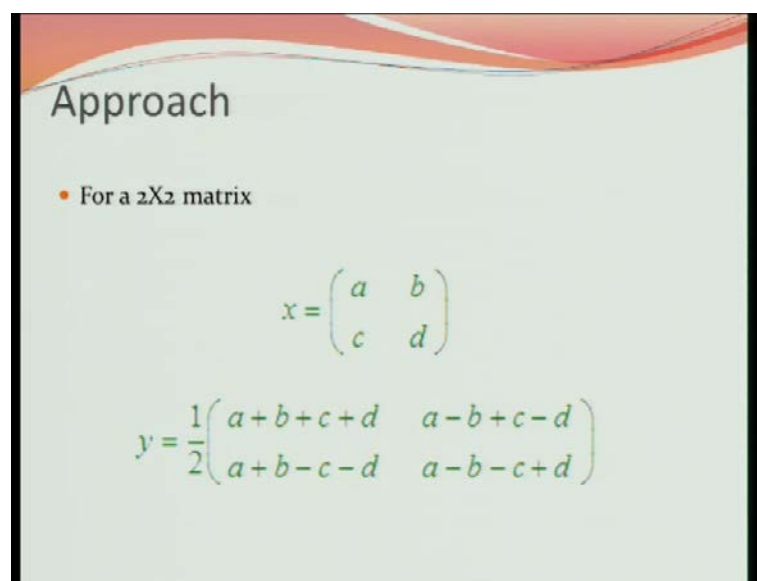
(( ))

Yes

So, actually I mean you have to create some pixels, because the length of the inner circle will be less than the outer **right**. So, you have extrapolate and then, you have to create you have to **fill the** fill the pixels.

So, in this image usually we will see that, when you are extrapolating it will be like smooth image **right**. So, when you open it you find that, this inner part is little smoother than the outer part. So, for that you have to extrapolate and then or interpolate and then you have to find out the values approximate values.

(Refer Slide Time: 37:58)



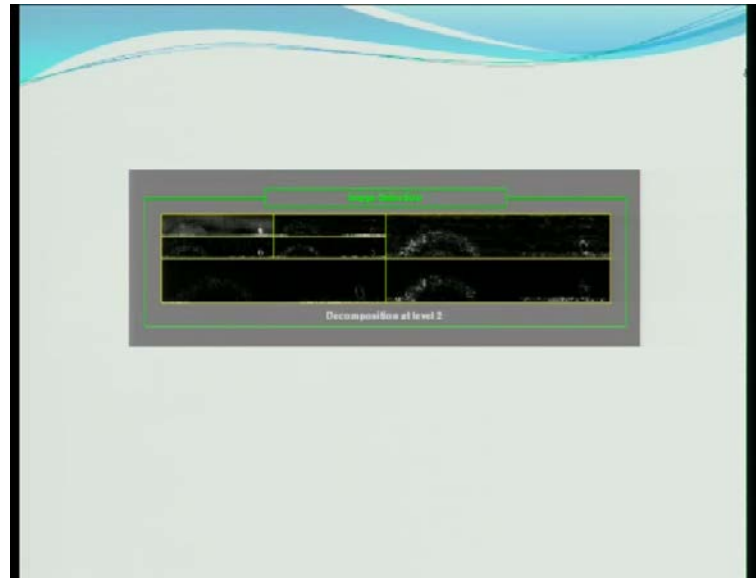
Approach

- For a 2X2 matrix

$$x = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$
$$y = \frac{1}{2} \begin{pmatrix} a+b+c+d & a-b+c-d \\ a+b-c-d & a-b-c+d \end{pmatrix}$$

So, once you have this portion, again you can use some wavelet, Haar wavelet or any other wavelet and you can decompose it and then, you can get the features, this is all you know **right** Haar wavelet **you know** correct.

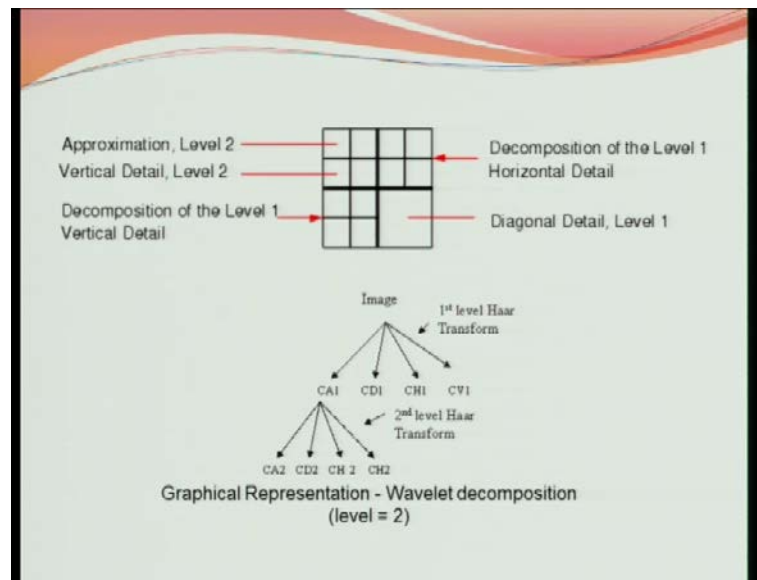
(Refer Slide Time: 38:16)



So, when you decompose you get image something like this. So, these are the gray scale value. Now, you can have some thresholding and convert this gray scale values to binary vectors **right**. Those things you have to experimentally find out all right, there is no fixed value for threshold that you can use always.

So for given set of images, for a given environment, that value changes. So, what this image is showing that, for the given image you have decompose **(( ))**, so this was let us say **horizontal detail and** horizontal detail, vertical detail and diagonal detail. And this was a approximation part that you have further decomposed.

(Refer Slide Time: 39:10)



And now what are the (( )) that you want to use for your features that depends. Sometime let us say, what are the available things let us say that you have decompose up to two levels then, you have this, this, this values and four values here correct (Refer Slide Time: 39:19). How many decompose image you have; three here and four here right.

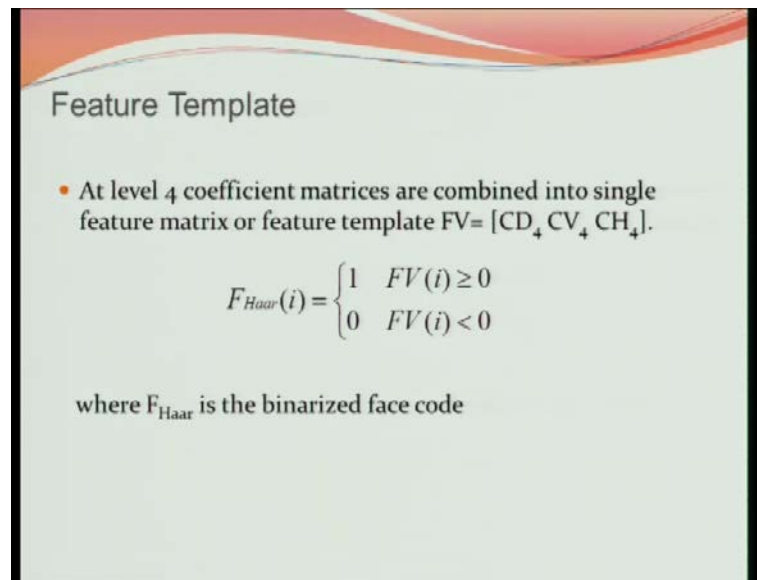
If you go to the image here, so what are the thing you have left with you, this three regions and this four regions (Refer Slide Time: 39:42). So, mostly the features are the detail exits in the detailed images, because they are basically the differences right; it is like something like s detection kind of thing, because there you take the difference. So, usually we do not consider the approximation part for recognition.

So, usually we will see that, we only consider the detail coefficients for matching (Refer Slide Time: 40:10). And another problem is that, it makes size of the vector; size of the feature vector very large right, because if let say you your image is 100 cross 100 plus 100, then in the next level you get 50 cross 50 image and if you convert it to binary and then open it 2500 is a long vector right.

So, in normally what what is done is we do not consider the level 1 coefficient; we go to the level 2, because level 2 has again done the size half. So, when will we consider the level 2 coefficient or may be if image size is very large; you can go to level 3 and then, consider the level 3 coefficients.



(Refer Slide Time: 40:56)



Feature Template

- At level 4 coefficient matrices are combined into single feature matrix or feature template  $FV = [CD_4, CV_4, CH_4]$ .

$$F_{Haar}(i) = \begin{cases} 1 & FV(i) \geq 0 \\ 0 & FV(i) < 0 \end{cases}$$

where  $F_{Haar}$  is the binarized face code

So, you do some thresholding and then based on that **there is done that** all those gray scale value are converted to 0 and 1. And then, again you use the hamming distance to match them, if the vectors are binary form; you use the hamming distance for matching this.

**(( ))**

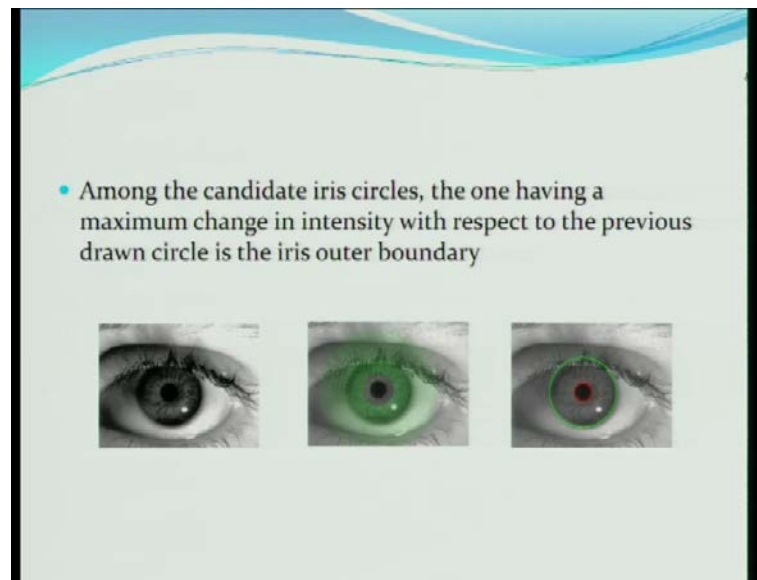
So, that is the kind of occlusion actually I mean like

**(( ))**

**Yes**

So, that does not happen **I mean** I mean all the pixels will not be bright, there may be one **(( ))** which will be very bright compared to the other portion, normally this that kind of occlusion exist.

(Refer Slide Time: 41:38)



So, for example here, you will find the few very bright locations and basically these are the locations, where your (( )) has got focused. So, I mean everything is illuminated then maybe you can use a iris for recognition, but based on some thresholding you can actually remove this kind of illuminated areas. And they are considering as a kind of a occlusion, because you do not get the iris data at that place.