

# Digital Image Processing

Prof. P.K. Biswas

Department of Electronics & Electrical Communication Engineering

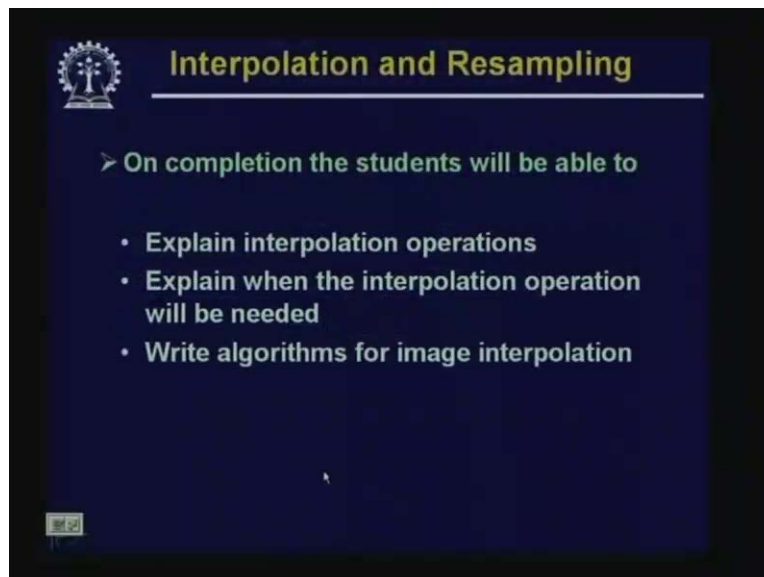
Indian Institute of Technology, Kharagpur

Lecture - 9

## Interpolation & Resampling

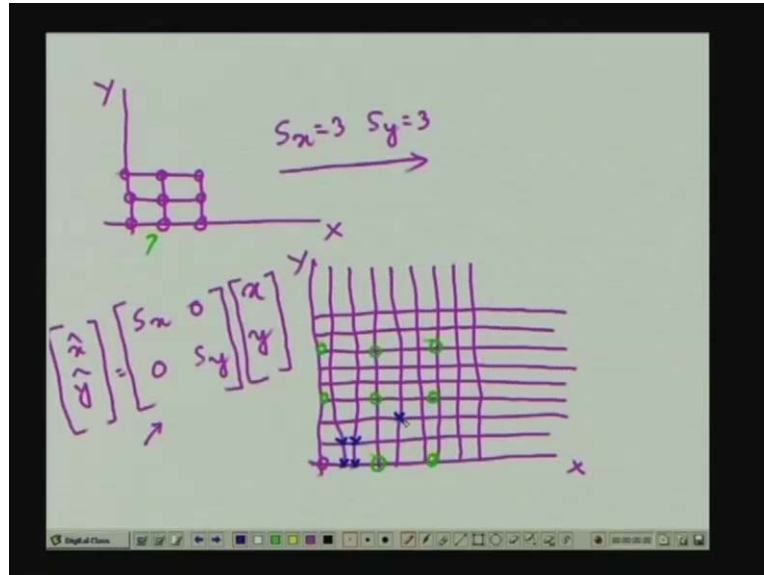
Hello, welcome to the video lecture series on digital image processing. Till the last class, we have seen various geometric transformations and we have seen how those geometric transformations can be used to the model an image formation process. We have also seen that how to calibrate a camera, given an particular imaging setup and we have also seen that using 2 identical cameras, how we can have a stereo imaging setup using which the 3D coordinate of a point in the 3 dimensional scene can be obtained.

(Refer Slide Time: 1:49)



Now, in today's lecture, we will try to explain some interpolation operations. We will explain when the interpolation operation is needed and at the end of the today's lecture, the students will be able to write algorithms for different image transformations and the needed interpolation operations. Now, let us see that why and when, do we need image interpolation and image resampling. So, let us first introduce this problem.

(Refer Slide Time: 2:25)



Say for example; if we have a 3 by 3 image like this. So, we have this X Y coordinate system and this X Y coordinate system we have a 3 by 3 image. So, I have an image pixel here, I have an image pixel here, I have an image pixel here, I have an image pixel here, here, here, here, here and here. So, you can easily identify that the coordinate of the image pixel, this particular image pixel is (0, 0), this is (1, 0), this is (2, 0) this is (0, 1), this is (1, 1), this is (2, 1), this is (0, 3) this is (1, 3) and this is (3, 3).

Now, let us try to apply some simple transformations, geometric transformations on these images. Say for example, I want to scale off this image by a factor of 3 in both the X dimension and Y dimension. So, if I scale off this image by factor 3; in that case, this 3 by 3 image will become a 9 by 9 image. And, let us see how those image points, how the pixels in the 9 by 9 image can be obtained?

So, I just apply a scaling operation by factor  $S_x$  equal to 3 and  $S_y$  is equal to 3. That is both in the X direction and Y direction and applying a scaling of factor 3. So naturally, this 3 by 3 image after being scaled off by the factor 3 in both the directions will be converted to an image of 9 by 9. So, let us see how these pixels values will look like?

Again, I put this X Y coordinate system. Now, you remember that this scaling operation is given by say  $(\hat{x}, \hat{y})$  is equal to  $(S_x, 0, 0, S_y)$  into column vector  $xy$ , where this  $(S_x, 0)$  and  $(0, S_y)$  this is the transformation matrix,  $xy$  is the coordinate of the pixel in the original image and  $(\hat{x}, \hat{y})$  is the coordinate of the pixels in the transform.

So, if I simply apply this scaling transformation, then obviously this (0, 0) point will lie at location (0, 0) in the transform. But what will happen to the other image points? So, because this will be converted to a 9 by 9 image, so let us first form a 9 by 9 grid.

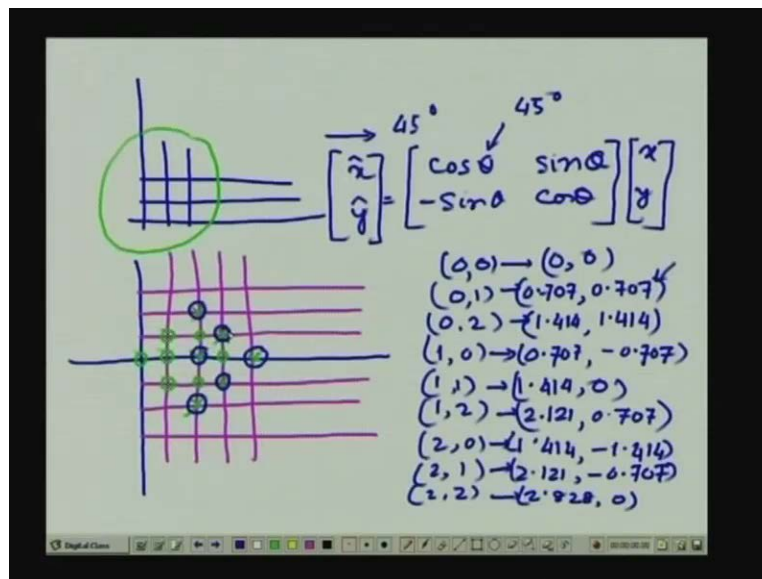
So now, you find that this (0, 0) pixel, even after this scaling transformation, remains at location (0, 0). But the other pixels, say for example, this pixel (1, 0) which was originally at location X coordinate equal to 1 and Y coordinate equal to 0, that will be transformed to Y coordinate will remain as 0 but now X coordinate will become equal to 3. So, this point will be matched to this particular location.

Similarly, (2, 0) will be matched to (6, 0) locations. So, this becomes 1, 2, 3, 4, 5, 6. So, this pixel will be matched to this particular location. Similarly, (0, 1) pixel will now be matched to (0, 3) location, (0, 2) pixel will now be matched to (0, 6) location; so 3, 4, 5, 6.

Similarly, I will have pixels in these different locations in the scaled image. But you find that because in the original image I had 9 different pixels, even in the transformed image I got 9 different pixels that is 3 pixels in the horizontal direction and 3 pixels in the vertical direction; but because I am applying a scaling of factor 3 in both X direction and Y direction, my final image size after scaling should be 9 pixels in the horizontal direction and 9 pixels in the vertical direction.

So, you find that there are many pixels which are not filled up in this scaled off image. So, those pixels, some of them I can simply mark; so this is one pixel which is not been filled up, this pixel has not been filled up, this pixel has not been filled up, this one has not been filled up. So likewise, there are many pixels in this particular image in this scaled up image which has not been filled up. Let us try to take another example.

(Refer Slide Time: 8:43)



So, I apply a instead of scaling, I apply a rotation operation to all these different pixels. So, I have this 3 by 3 pixel and suppose I rotate this image by affect by an angle of 45 degree in the clockwise direction; so if I rotate this image by 45 degree in the clockwise direction, you know that we had a transformation matrix which takes care of rotation and that is given by (cosine theta sin theta) then (minus sin theta cosine theta).

So, this is the rotation matrix which when applied to different pixels in the original image will give you the pixels in the rotated image. So, in the rotated image  $i$  can represent these pixels by  $(\hat{x} \hat{y})$  whereas, my original pixels are  $x$  and  $y$  and in this particular case, the value of  $\theta$  is simply 45 degree. So, if I apply this transformation, the rotation transformation to all the pixels in the original image; you will find that  $(0, 0)$  location will be transformed to location  $(0, 0)$  even in the transformed image,  $(0, 1)$  location will be transformed to  $0.707$  and  $0.707$  location then  $(0, 2)$  point will be transformed to location  $1.414$  and  $1.414$ .

Similarly  $(1, 0)$  location, pixel at location  $(1, 0)$  will be transformed to location  $0.707$  and minus  $0.707$ , location  $(1, 1)$  will be transformed to location  $1.414$  and  $0$  and location  $(1, 2)$  will be transformed to location  $2.121$  and  $0.707$ . Similarly the other coordinates,  $(2, 0)$  this pixel will be transformed to location  $1.414$  and minus  $1.414$ .  $(2, 1)$ , this will be transformed to location  $2.121$  and minus  $0.707$  and  $(2, 2)$  this particular image pixel will be transformed to location  $2.828$  and  $0$ . So, these are the various transformed locations of the pixels in the rotated image.

Now, if you just look at these transformed locations, you find that the coordinates that we get are not always integer coordinates. In many cases, in fact in this particular example most of the cases, the coordinates are real value coordinates. But whenever we are going to have a digital image, whether it is the original image or the transformed image, even in the transformed image, all the row index and the column index should have an integer value; I cannot represent any real number or fractional number as a row index or a column index.

So in this case, whenever I am going for this particular transformation, what I have to do is whenever I am getting a real number as a row index or a column index; I have to take its nearest integer where that particular pixel will be put. So, in this case for the original image location  $(0, 1)$  which has now been transformed to location  $0.707$  and  $0.707$ , this has to be mapped to a pixel location  $(1, 1)$  in the transformed image.

So, if I do that mapping, if you find that all the pixel locations will now be mapped like this. So, I put a new grid and the new grid will appear like this. So, you find that the  $(0, 0)$  location has been matched to  $(0, 0)$  location; so, I have a point over here, pixel over here.

$(0, 1)$  location in the original image has been transformed to  $0.707$  and  $0.707$  in the transformed image. So, what I have to do this I have to take the nearest integer of these fractional numbers where this particular pixel will be put. So,  $0.707$  in the X direction and  $0.707$  in the Y direction will be matched to  $(1, 1)$  in the transformed image. So, this  $(0, 1)$  point will now be mapped to location  $(1, 1)$  in the transformed image; so, I get a point here.

Similarly  $(0, 2)$ , you find that the corresponding transformed location is  $1.414$  and  $1.414$ . Again, the nearest integer of  $1.414$  is  $1$ ; so this point  $(0, 2)$  will also be matched to location  $(1, 1)$  in the transformed image.  $(1, 0)$  in the same way will be matched to location  $(1, \text{minus } 1)$ . So,  $(1, 0)$  will be matched to this particular point in the transformed image.  $(1, 1)$  will be matched to  $(1.414, 0)$ . Here again, by integer approximation, this point will be matched to  $(1, 0)$  location in the transformed image.

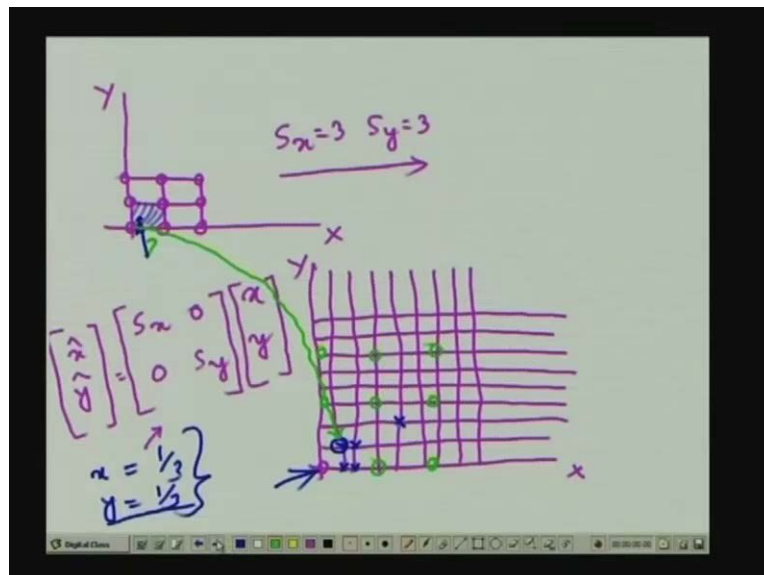
So, I have a pixel in this particular location. (2, 1) point will be matched to 2.121 and 0.707. So, again by integer approximation, I map this pixel to (2, 1). So, this is the point where the pixel (2, 1) (1, 2) of the original image will be mapped.

In the same manner, (2, 0) point will be matched to location (1, minus 1) where I already have a particular point. (2, 1) location will be mapped to location (2, minus 1), so I will have a point here. (2, 2) location will be matched to (3, 0) in the transformed image, so I will have a pixel in this particular location.

So, you find that in the original image, we had 9 pixels whereas in this rotated image, we are going to have only 7 pixels and this comes because when you are rotating any image, the integer coordinates in the original pixel, some of them turns out to be fractions or real number in this transformed image and these fractions or real numbers cannot be represented in digital form. So, we have to round it, round those numbers to the nearest integer and which gives leads to this kind problem.

And, not only this, you find that if I just rotate this original image, ideally my rotated image should have been something like this. But here, you find that there are a number of points where I do not have any information. Say for example, this point; I do not have any information. This point, I do not have any information. This point, I do not have any information. Similarly all these points, I do not have any information and the reason is because of digitization. So, to fill up these points, what I have to do is I have to identify in the transformed image, what are the locations where I do not have any transformation.

(Refer Slide Time: 17:47)



So, take the simple case, take the previous one. Here you find that I do not have any information at location (1, 1) of the transformed image, of the scaled image. So, because I do not have any information here, now I have to look in the original image to find out which value should be put at this particular location.

Now, because this image I have obtained using a scaling of 3 in both X and Y direction, if I want to go back to original image; then to this transformed image, I have to apply a scaling of 1 third in both X direction and Y direction. Now, you find that in the transformed image, this particular location, this particular pixel has a coordinate of (1, 1). So, if I transform this, inverse transform this using scaling factor of 1 third and 1 third; I get in the original image, the x coordinate should be equal to 1 upon 3, the y coordinate should also be equal to 1 upon 3.

Now, here comes the problem. In the original image, I have the informations at location (0, 0), I have the information location information at location (0, 1), I have information at location (1, 0), I have information at location (1, 1). But at location (1, 3) and (1, 3), I do not have any information because you remember from our earlier classes that whenever we have gone for image digitization, the first step we had done was sampling and the second step that we had done quantization.

Now, the moment we sample the image, what we have done is we have taken some representative value from discrete grid points. We have not considered the intensity values at all possible points in the continuous image. So, in the process of the sampling, whatever value was there at location (1 upon 3, 1 upon 3) in the continuous image, that information is lost. So, in the digital image at this particular location (1 upon 3, 1 upon 3); I do not have any information. So, what is the way out?

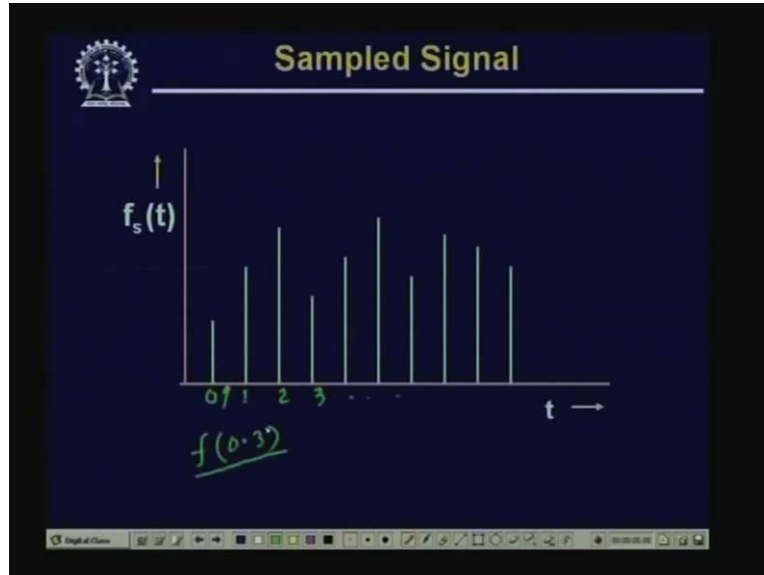
Now, the only process that I have to do is I have to go for approximation of the intensity value which should have been at this location (1 by 3, 1 by 3). Now, how to get that approximate value? That is the problem. So, only way in which this can be done is I have to interpolate the image in these points where I do not have any information.

And after interpolation, so in these locations where I do not have any information in my discrete image, I have to interpolate the values in these locations and after interpolation, I have to check what should be the interpolated value at location (1 by 3, 1 by 3). And whatever value I get at this location (1 by 3, 1 by 3), this particular value has to be taken to fill up this location (1, 1) in the transformed image. Similar is the case for the other transformation that is rotation.

In this case also, this rotated image, we have obtained by rotating the original image by 45 degree in the clockwise direction. So, whenever I find any point in the rotated image where there is no information, what I have to do is that particular coordinate, I have to inverse transform that is I have to give a rotation to that particular point by minus 45 degree; go back to the original image point and obviously in this case, in the original image, these row column will not be integers but they will be real numbers or fractions.

And because we have real number or fractions, for which we do not have any information in the original digitized image; we have to go for interpolation and after interpolation, we have to go for resampling to find out what should be the intensity value or approximate intensity value at that particular location. Then take that intensity value and put it in to the point in the transformed image where I do not have the information. So, this is why you find that the interpolation and resampling is very very important whenever you are working in the digital domain or you are doing some sort of transformations over a digital image.

(Refer Slide Time: 22:40)



Now, whenever we go for interpolation; so this gives the situation that we have a 1 dimensional signal  $f(t)$  of function  $t$  and after sampling, we have got the sampled signal  $f_s(t)$ . So here, you find that after sampling, what we have got is the sample values which are represented by  $f_s(t)$ .

Now, as we have  $f_s(t)$ , these values are present only at discrete locations. So, at any intermediate location in this particular one, we do not have any information of this function  $f(t)$  and because we do not have these informations, we go for interpolation for all those values of  $t$  where we do not have the samples present. And after interpolation, again, we have to go for resampling to fill up those positions.

So, this slide shows a sampled 1 dimensional signal  $f(t)$  of a function  $t$ . So, after resampling, we have represented the signal by  $f_s(t)$  where  $f_s(t)$  is nothing but a sequence of sample values. So, in this case you find that we have the values available for say  $t$  equal to 0. Here I can put  $t$  equal to 0 at  $t$  equal to 1,  $t$  equal to 32,  $t$  equal to 3 and so on. But I do not have any information for a value of  $t$  which is in between 0 and 1.

So, if I need to obtain a value of  $f$  at location say 0.3, then what I have to do is I have to interpolate this function  $f_s(t)$  and I have to find out after resampling that what will be the value of the function at  $t$  equal to 0.3. So, this is why the interpolation and resampling is very very important whenever you are working with a digital signal and a digital image in particular and you are going for any type of transformation, particularly the rotation and translation of the digital image.

Usually the translation operation, if it is only translation does not need any kind of resampling or interpolation. Now, whenever we interpolate an image, the interpolation operation should have certain desirable properties. Firstly, the interpolation function that you use for interpolating the discrete values that should have a finite region of support. That means interpolation should be

done based on the local information, it should not be based on the global information or it should not take into consideration all the sample values of that particular digitized signal.

The second desirable property is the interpolation should be very smooth. That is the interpolation should not introduce any discontinuity in the signal. And the third operation is the interpolation should be shift invariant. So, if the signal is shifted or given some translation, then also the same interpolation function **should be available** should be done and the B spline function is one such function which satisfies all these 3 desired properties.

(Refer Slide Time: 26:55)



Now, let us see, what is this B spline function? B spline function is a piece wise polynomial function that can be used to provide local approximation of curves using very small number of parameters and because it is useful for local approximation of curves; it can be very very useful for smoothing operation of some discrete curves, it is also very very useful for interpolation of a function from discrete number of samples. So, let us see what this B spline function is.



(Refer Slide Time: 27:54)

The image shows a whiteboard with handwritten mathematical equations. At the top, the equation is  $x(t) = \sum_{i=0}^n p_i B_{i,k}(t)$ . Below this, a bracket points to  $B_{i,k}(t)$  with the text "Normalized B-Spline of order k." and another line says " $p_i =$  control points". Below that, the first-order B-spline is defined as  $B_{i,1}(t) = \begin{cases} 1 & t_i \leq t < 1 \\ 0 & \text{otherwise} \end{cases}$ . The recursive formula for higher-order B-splines is given as  $B_{i,k}(t) = \left[ \frac{(t-t_i) B_{i,k-1}(t)}{t_{i+k-1} - t_i} \right] + \left[ \frac{(t_{i+1} - t) B_{i+1,k-1}(t)}{(t_{i+k} - t_{i+1})} \right]$ .

A B spline function is usually represented by say  $x(t)$  is equal to sum of  $p_i$  into  $B_{i,k}(t)$  where you take the summation from  $i$  equal to 0 to  $n$ , where 0 to  $n$  that is  $n$  plus 1 is the number of samples which are to be approximated.

Now, these points  $p_i$ , they are called the control points and  $B_{i,k}$  is the normalized B spline of order  $k$ . So  $B_{i,k}$ , this is the normalized B spline; B spline of order  $k$  and  $p_i$  are known as the control points. So, this control points actually decide that how the B spline functions should be guided to give you a smooth curve.

Now, this normalized B spline that is  $B_{i,k}$  can be recursively defined as  $B_{i,1}$  is equal to 1 whenever  $t_i$  is less than or equal to  $t$  less than 1. So, this is  $B_{i,1}$  of  $t$ , it is equal to 1 whenever  $t_i$  is less than or equal to  $t$  and which is less than 1. And  $B_{i,1}(t)$  is equal to 0 whenever value of  $t$  takes other values.

So, this is equal to 1 for all values of  $t$  lying between  $t_i$  and 1,  $t_i$  is inclusive and  $B_{i,1}(t)$  is equal to 0 for any other value of  $t$  and then we can find out  $B_{i,k}$  and  $k$  of  $t$  using the relation; so,  $B_{i,k}(t)$  is equal to  $(t - t_i)$  into  $B_{i,k-1}(t)$  upon  $t_{i+k-1} - t_i$  plus  $(t_{i+1} - t)$  into  $B_{i+1,k-1}(t)$  upon  $t_{i+k} - t_{i+1}$ . So once we have  $B_{i,1}$  for different values of  $t$ , then from this  $B_{i,1}$ , we can recursively compute the values of  $B_{i,k}$  using this relation.

(Refer Slide Time: 32:14)

$$B_{i,k}(t) = B_{0,k}(t-i)$$
$$B_{0,1}(t) = \begin{cases} 1 & 0 \leq t < 1 \\ 0 & \text{otherwise} \end{cases}$$
$$B_{0,2}(t) = \begin{cases} t & 0 \leq t < 1 \\ 2-t & 1 \leq t < 2 \\ 0 & \text{otherwise} \end{cases}$$

Now, you find that once we have this relation of  $B_{i,k}(t)$ , you can easily verify that once I have  $B_{0,k}(t)$ ; then  $B_{i,k}(t)$  is nothing but translates of  $B_{0,k}(t)$ . So, this  $B_{i,k}(t)$  can be written as this is nothing but  $B_{0,k}(t-i)$ . So, this can be easily verified from the way this B spline function is defined.

Now, this B spline function for various values of  $i$  and  $k$  can be obtained like this. You can easily get that  $B_{0,1}(t)$  will be equal to 1 whenever  $0 \leq t < 1$  because earlier we had said that  $B_{i,1}(t)$  is equal to 1 whenever  $t_i$  is less than or equal to  $t$  which is less than 1 and it is 0 otherwise. So, just by extending this, I can just write that  $B_{0,1}(t)$  will be equal to 1 whenever  $t$  lies between 0 and 1, 0 inclusive and it will be equal to 0 otherwise.

So, you find that  $B_{0,1}(t)$  is constant in the region 0 to 1. Similarly, we can find out  $B_{0,2}(t)$  will be equal to  $t$  for  $0 \leq t < 1$ . It will be equal to  $2-t$  for  $1 \leq t < 2$  and it will be equal to 0 otherwise.

(Refer Slide Time: 34:24)

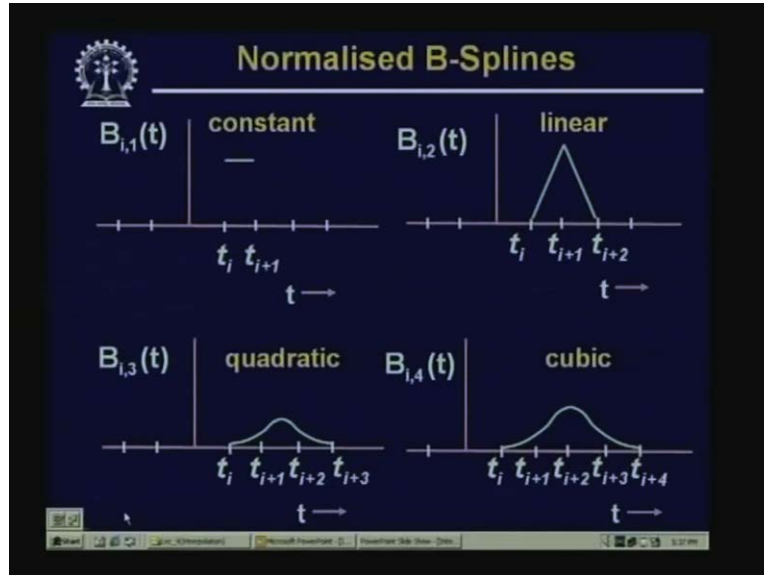
The image shows handwritten mathematical definitions for B-spline basis functions on a whiteboard. The first function is  $B_{0,3}(t)$ , defined as a piecewise function:  $\frac{t^2}{2}$  for  $0 \leq t < 1$ ,  $-t^2 + 3t - 1.5$  for  $1 \leq t < 2$ ,  $\frac{(3-t)^2}{2}$  for  $2 \leq t < 3$ , and 0 otherwise. The second function is  $B_{0,4}(t)$ , defined as a piecewise function:  $\frac{t^3}{6}$  for  $0 \leq t < 1$ ,  $\frac{-3t^3 + 12t^2 - 12t + 4}{6}$  for  $1 \leq t < 2$ ,  $\frac{3t^3 - 24t^2 + 60t - 44}{6}$  for  $2 \leq t < 3$ ,  $\frac{(4-t)^3}{6}$  for  $3 \leq t < 4$ , and 0 otherwise. The label  $B_{i,k}$  is circled next to the second function.

Similarly, you can find that  $B_{0,3}(t)$  can be written as  $t^2$  by 2 for  $0 \leq t < 1$ . This will be equal to  $-t^2 + 3t - 1.5$  for  $1 \leq t < 2$ . It will be  $\frac{3-t^2}{2}$  for  $2 \leq t < 3$  and it will be 0 otherwise.

Similarly, we can also find out  $B_{0,4}(t)$  will be equal to  $t^3$  by 6 for  $0 \leq t < 1$ . It will be equal to  $\frac{-3t^3 + 12t^2 - 12t + 4}{6}$  for  $1 \leq t < 2$ . It will be equal to  $\frac{3t^3 - 24t^2 + 60t - 44}{6}$  for  $2 \leq t < 3$ . This will be equal to  $\frac{4-t^3}{6}$  for  $3 \leq t < 4$  and it will be 0 otherwise.

So, you can obtain all these values from the definition of **the definition of** the B-spline function that we have defined earlier. Now, whenever I write this  $B_{i,k}$ , this  $k$  is called the order of the order of B-spline function.

(Refer Slide Time: 36:43)



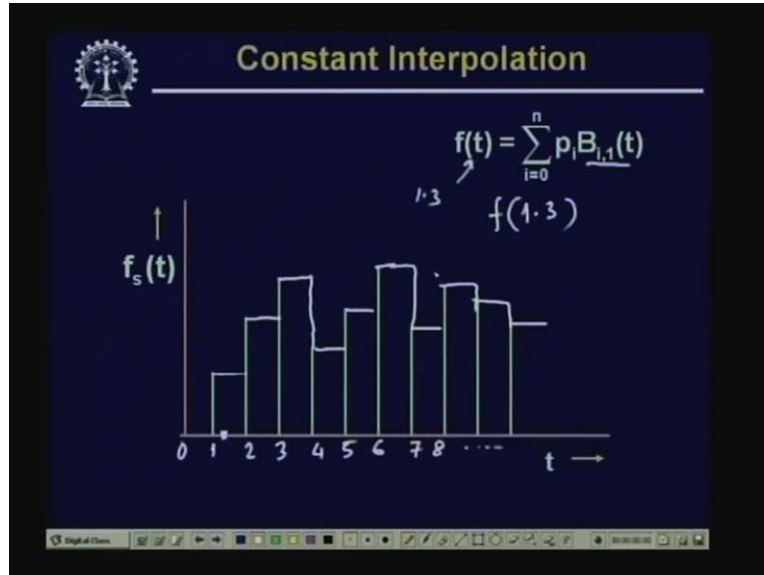
Now, after giving these equations, let us try to see that what is the nature of this B spline functions. So, you find that  $B_{i,1}(t)$  because it will be equal to 1 for  $i$  equal to 1 to 2, between  $i$  equal to 1 to 2 this  $B_{i,1}(t)$  will be equal to constant and it will be equal to 1.

So, this first figure in this particular case tells you that what is the nature of this  $B_{i,1}$ . The second figure shows what is the nature of this B spline function if it is  $B_{i,2}(t)$  and it shows that it is a linear function. So,  $B_{i,2}(t)$  will lie between, will have a support from  $i$  to  $i$  plus 2 and the points which will be supported by this  $B_{i,2}(t)$  are  $i$ ,  $i$  plus 1 and  $i$  plus 2.

Similarly, a quadratic function which is  $B_{i,3}(t)$  is given by this figure and a cubic function which is  $B_{i,4}(t)$  is given by this fourth figure and here you find that the region of support for this cubic function is 5 points, the region of support for quadratic function is 4 points, the region of support for the linear B spline is 3 points whereas, the region of support for  $B_{i,1}$  for the B spline of order 1 is only 2 points. So, in all the cases, the region of support is finite.

Now, let us see that using these B splines, how we can go for interpolation.

(Refer Slide time: 38:31)



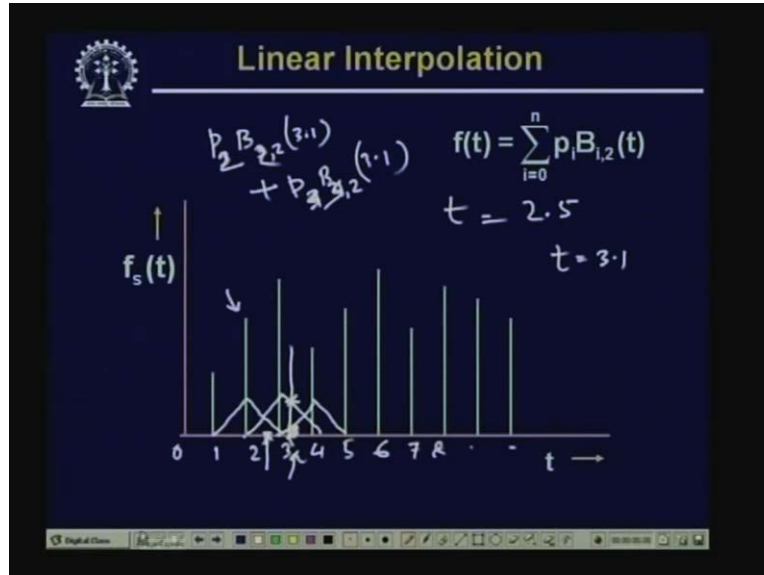
As we have said earlier that using these B splines, a function is approximated as  $f(t)$  equal to  $p_i B_{i,1}(t)$  or  $B_{i,k}(t)$  where  $i$  varies from 0 to  $n$ . So in this case, let us take the value of  $k$  to be equal to 1. That means we have a B spline of order 1 and we have shown that if we have a B spline of order 1, then the nature of the B spline function is it is constant between  $i$  and  $i$  plus 1 and which is equal to 1.

So using this, if I try to interpolate this particular function as shown in this diagram which are nothing but some sample values, I take say,  $t$  is equal to 0 here,  $t$  equal to 1 here,  $t$  equal to 2 here, 3 here, 4 here, 5, 6, 7, 8 and so on. Now, if I want to find out say  $f$  of 1.3. So,  $f$  of 1.3 should lie somewhere here. So, to find out this  $f$  of 1.3, what I have to do is I have to evaluate this function  $f(t)$  where I have to put  $t$  is equal to 1.3 and this  $f(t)$  is given by this expression that is  $p_i B_{i,1}(t)$  where  $i$  varies from 0 to  $n$ .

Now, find that this  $B_i(t)$  if I take  $i$  equal to 1, so  $B_{1,1}$  is a function like this. So, in between 1 and 2, this  $B_{1,1}$  is constant and that is equal to 1. So, if I find out the value at this particular point, it will be  $p_i$  that is  $f_s(1)$  multiplied by  $B_{1,1}$  at  $t$  equal to 1.3 and which is equal to 1 and this  $B_{1,1}$  is equal to 1 for all values of  $t$  from  $t$  equal to 1 to  $t$  equal to 2 but excluding  $t$  equal to 2.

So, if I interpolate this function using this  $B_{i,1}(t)$ , you will find that **this interpolation will be of this form sorry** this interpolation will now be of this form; so, it goes like this. So, all the values at all points between  $t$  equal to 1 and  $t$  equal to 2, the interpolated value is equal to  $f_s(1)$ . Similarly between  $t$  equal to 2 and  $t$  equal to 3, the interpolated value is equal to  $f_s(2)$  and so on.

(Refer Slide Time: 42:14)



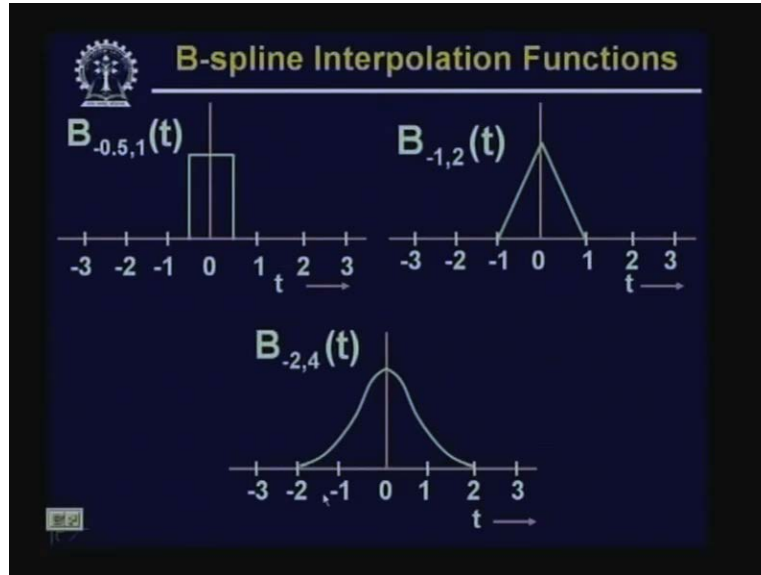
Similarly, if I go for interpolation using say linear interpolation where value of  $k$  is equal to 2; in that case, again you find that if I put say  $t$  equal to 0 here,  $t$  equal to 1,  $t$  equal to 2,  $t$  equal to 3, 4, 5, 6, 7, 8 like this; now,  $B_{1,2}$  is a linear function between 1 and 3, so  $B_{1,2}$  is something like this. Similarly,  $B_{2,2}$  is something like this,  $B_{3,2}$  is something like this.

So, if now I want to have I want to interpolate or I want to find out the value of this function at point say 2.5, at  $t$  equal to 2.5, so here. Then you will find that the sample values which take part in this interpolation are  $f_2$  and  $f_3$  and by using these 2 sample values, by linear interpolation; I have to find out what is the interpolated value at  $t$  equal to 2.5.

Now, you take a case that I want to interpolate this function value  $f(t)$  at  $t$  equal to say 3.1 that means somewhere here. So, if I want to do this, then what I have to do is this will be an interpolation of  $p_3$ . So, this will be nothing but  $p_3$  into  $B_{3,2}$  at point  $t$  equal to 3.1 plus you can easily find out it will be  $p_4$  into  $B_{4,2}$  again at point 3.1. So, I want to interpolate this value here.

Now, you find that weight of this  $p_3$  is given by only this much whereas, weight of sorry this will be  $p_2 B_{2,2}$  and  $p_3 B_{3,2}$ . So, weight of  $B_2$  is given by this value, whereas weight of  $p_3$  is given by this value. So, when I am interpolating at 3.1 and giving less weightage to  $p_3$  which is nearer to this particular point  $t$  equal to 3.1 and I am giving more weightage to this particular point  $p_2$  which is away from 3.1 which is not very logical. So in this case, we have to go for some modification of this interpolation function. So, what is the modification that we can think of?

(Refer Slide Time: 45:35)



(Refer Slide Time: 45:45)

$$f(t) = \sum_{i=0}^n p_i B_{i,k}(t)$$

↓

$$f(t) = \sum_{i=0}^n p_i B_{i-s,k}(t)$$

$s = 0.5$	$k = 1$
1	$k = 2$
2	$k = 4$

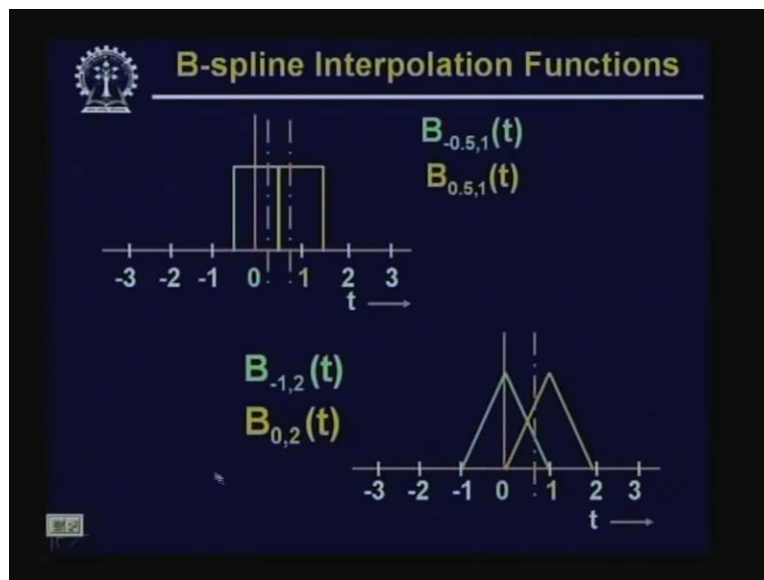
The kind of modification that we can do in this case instead of having the interpolation as say,  $f(t)$  is equal to  $p_i B_{i,k}(t)$  summation over  $i$  equal to 0 to  $n$ ; I will just modify this expression as  $f(t)$  is equal to  $i$  will put same  $B_i p_i$  but  $B$ , I will shift by some value. So, I will take it  $B_i$  minus  $s$   $k(t)$ . Again,  $i$  will vary from 0 to  $n$  and the value of this  $i$  will be, value of this shift  $s$  will depend upon what is the value of  $k$ .

So, I will take  $s$  is equal to 0.5 if I have  $k$  equal to 1 that is constant interpolation, I will take  $s$  equal to 1 if I have  $k$  equal to 2 that is linear interpolation and I will take  $s$  equal to 2 if  $k$  is equal

to 4 that is **I have by** I have cubic interpolation. Now, you find that I have not considered  $k$  equal to 3 which is quadratic interpolation because quadratic interpolation leads to asymmetric interpolation.

If I go for other interpolations like  $k$  equal to 1 with of course a shift of  $B_{i-k}$  by a value of 0.5; so  $s$  equal to 0.5 or if I take  $k$  equal to 2 which is equal to 1 or if I take  $k$  equal to 4 with  $s$  equal to 2, what I get is a symmetric interpolation. So, these are the different interpolations, the B spline interpolation functions that we can use for interpolating the function from sample values.

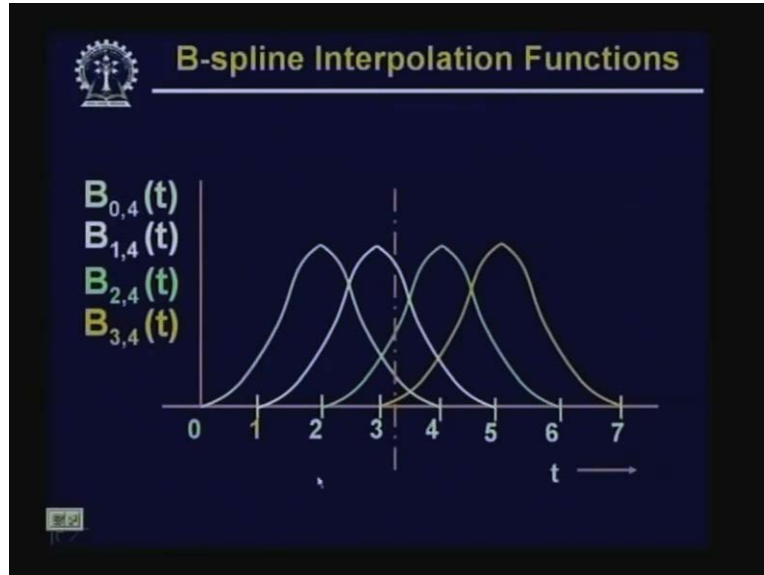
(Refer Slide Time: 48:01)



So, this shows the situation when we have shifted this  $B_{i-1}$  by a value 0.5. So here, you find now that  $B_{i-1}$  is constant from minus 0.5 to plus 0.5. Similarly 0.5 to 1.5, it will be from 1.5 to 2.5 and so on. Similarly  $B_{i,2}$ , now it is the regions of support for  $B_{0,2}$  is between minus 1 and 1, for  $B_{1,2}$  is from 0 to 2 and so on.

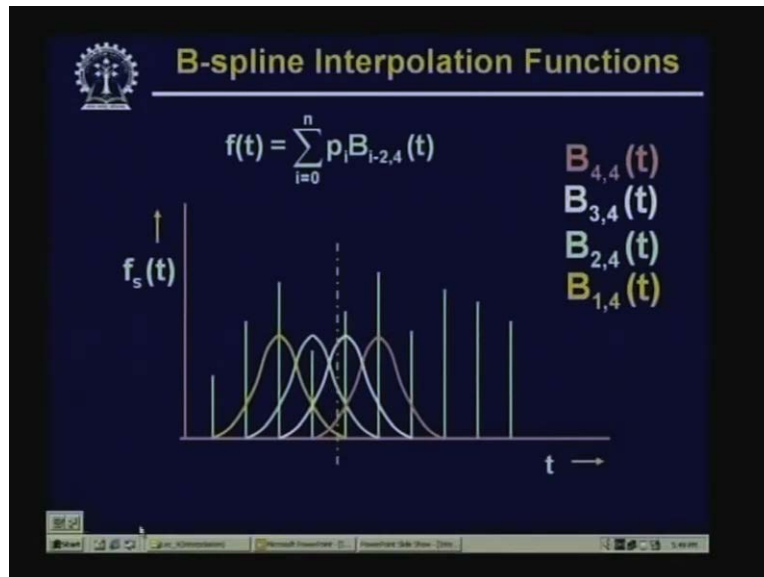


(Refer Slide Time: 48:46)



So by using this, similarly for cubic interpolation; I do the corresponding shifting.

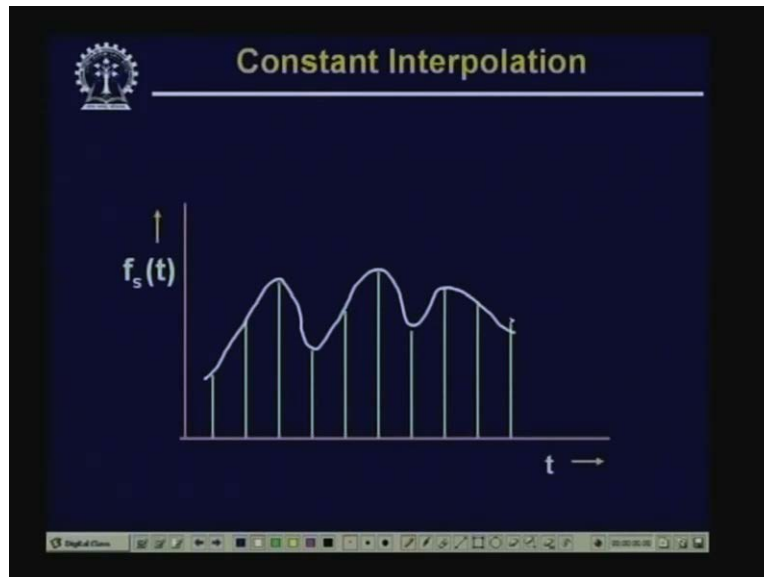
(Refer Slide Time: 48:52)



And by using this, now the interpolation can be obtained as if I go for cubic interpolation that  $f(t)$  is equal to  $p_i B_{i-2,4}(t)$  where the summation has to be taken from  $i$  equal to 0 to  $i$  equal to  $n$ . And here, it gives that if I want to interpolate this function at this particular point; the weight given by this particular sample is only this much, the weight given by this particular sample is this much, the weight given by this particular sample is this much, the weight given by this particular sample, this sample is weighted by this much.

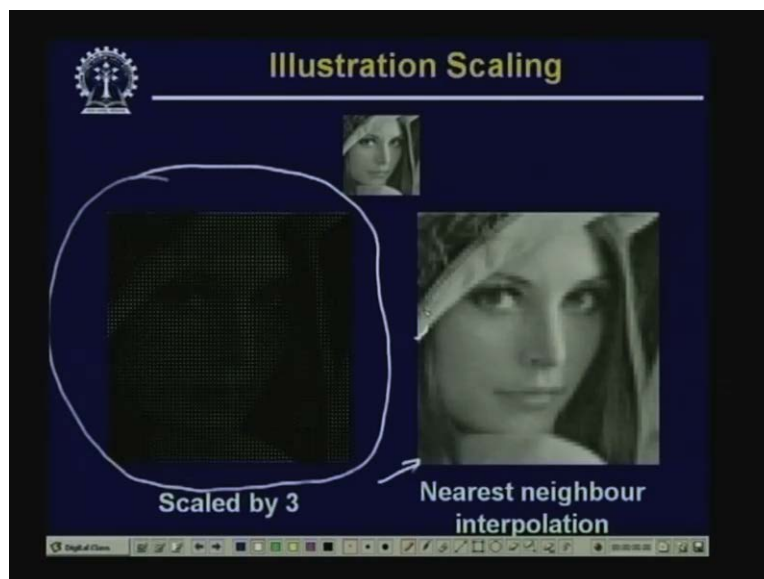
So, by taking this weighted average, the weights given by this B spline functions; I can find out what will be the interpolated value at this particular location.

(Refer Slide Time: 50:04)



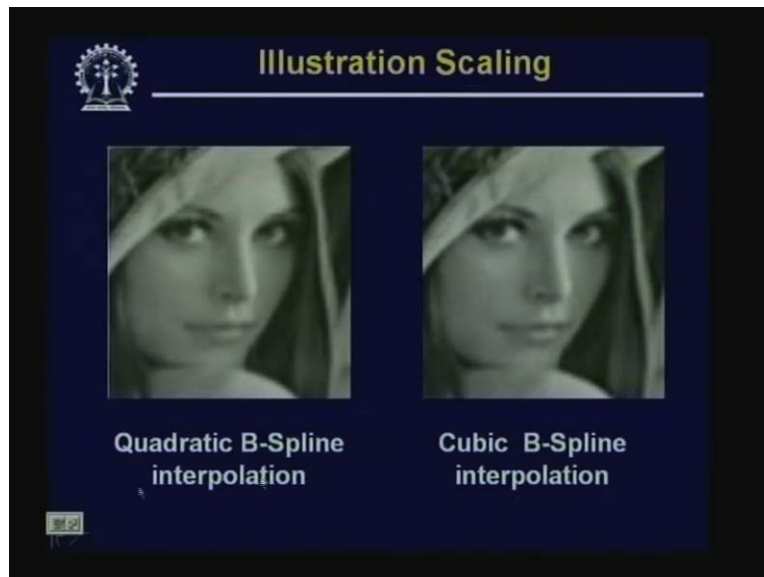
So, if I use that cubic interpolation function; possibly, I will have for this set of sample values, an interpolation a smooth interpolation like this. So maybe, this kind of smooth interpolation is possible using the cubic B spline function. Now, let us see some of the results on the images that we have obtained.

(Refer Slide Time: 50:40)



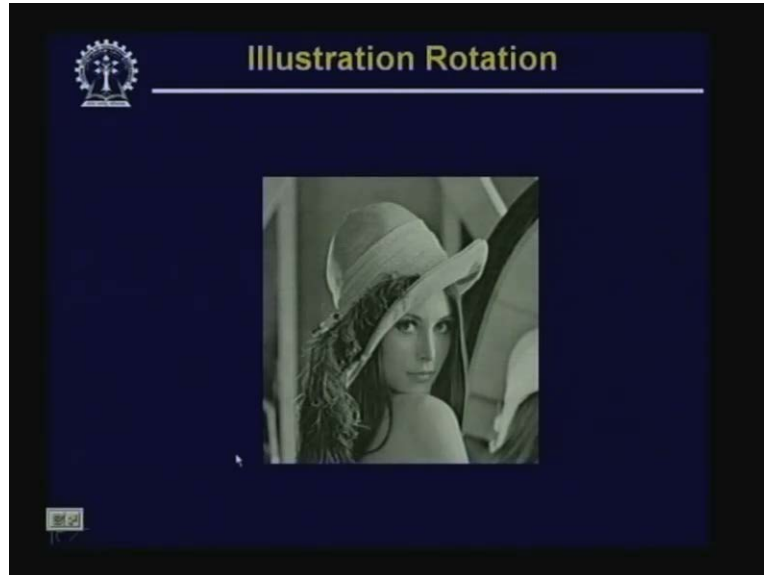
So, this is the example of a scaling operation. I have a small image which is scaled off by a factor 3 in both X direction and Y direction. You find that the image on the left side is obtained by scaling without applying any interpolation. So obviously, you will find that here the image appears to be a set of dots where many of the pixels in the image are not filled up. If I go for nearest neighbor interpolation or constant interpolation, this is the reconstructed image that I can get and you find that here, I have the blocking artifacts and the image appears to be a collection of blocks.

(Refer Slide Time: 51:34)



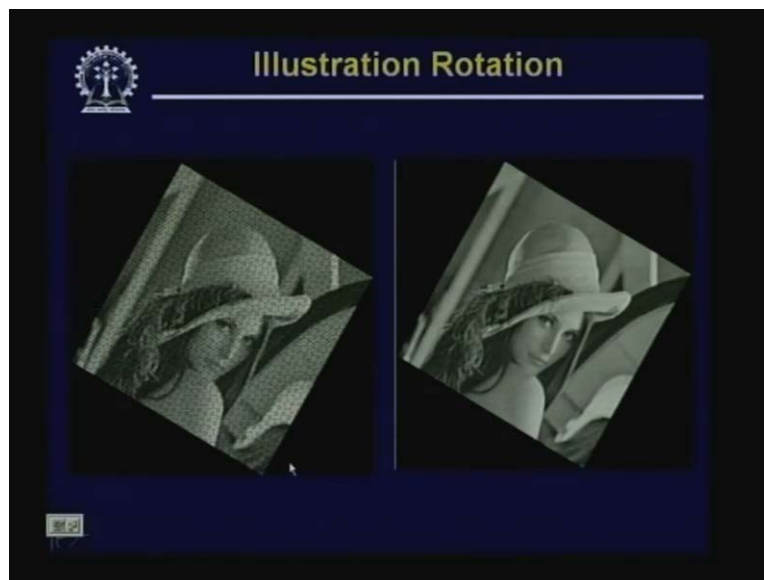
Similarly, if I go for other interpolation techniques; if I use quadratic B spline interpolation, then this is the quality of the image that we obtain. If we go for cubic B spline interpolation, then this is the quality of the image that we can obtain.

(Refer Slide Time: 51:48)



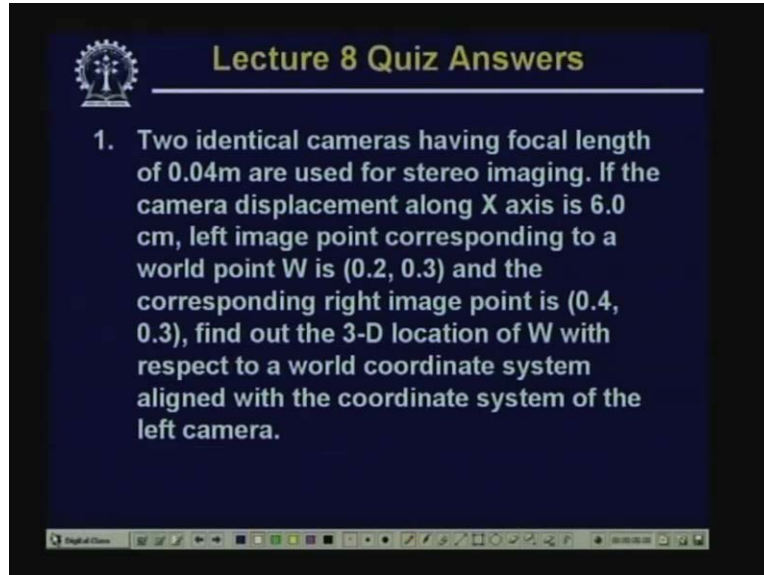
The same result is also the same experiment is also done in case of rotation.

(Refer Slide Time: 51:58)



This particular image has been rotated by 30 degrees and if I do not apply any interpolation; so without interpolation, the rotated image is shown on the left hand side and here again you find that there are a number black spots in this rotated image which cannot be filled up which could not be filled up because no interpolation was used in this case. Whereas, in the right hand side, this particular image after rotation has been interpolated using by cubic interpolation function. So, you find that all those black spots in the digital image had been filled up.

(Refer Slide Time: 52:32)

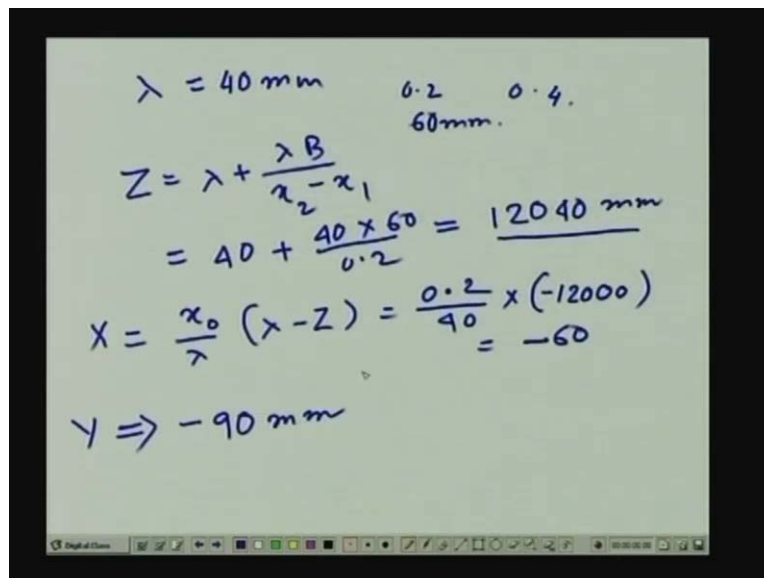


**Lecture 8 Quiz Answers**

1. Two identical cameras having focal length of 0.04m are used for stereo imaging. If the camera displacement along X axis is 6.0 cm, left image point corresponding to a world point W is (0.2, 0.3) and the corresponding right image point is (0.4, 0.3), find out the 3-D location of W with respect to a world coordinate system aligned with the coordinate system of the left camera.

Now, let us see the answers of the quiz questions that we had given in the last class. In the last class, we had given a quiz question for finding out the 3D coordinate point of a world point per its coordinate in the left camera and the coordinates in the right camera are given.

(Refer Slide Time: 52:44)



$\lambda = 40 \text{ mm}$       0.2      0.4  
60mm.

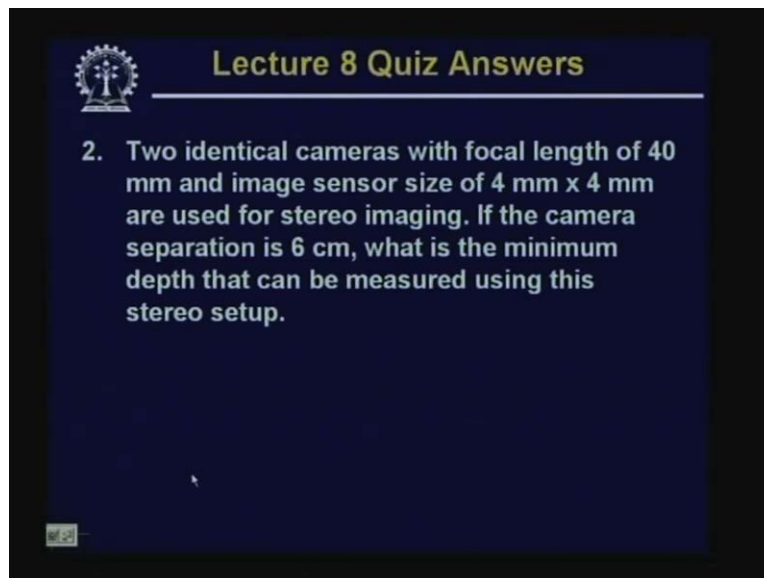
$$Z = \lambda + \frac{\lambda B}{x_2 - x_1}$$
$$= 40 + \frac{40 \times 60}{0.2} = \frac{12040 \text{ mm}}$$
$$X = \frac{x_0}{\lambda} (\lambda - Z) = \frac{0.2}{40} \times (-12000) = -60$$
$$Y \Rightarrow -90 \text{ mm}$$

The value of the focal length lambda was give as 40 millimeter and the X coordinate in one case was given as 0.2 in the left camera and in the right camera, it was given as 0.4 and the camera separation was given as 6 centimeter that is 60 millimeter.

So, if I simply use the formula; say,  $Z$  equal to  $\lambda$  plus  $\lambda B$  by  $x_2$  minus  $x_1$ , you will find that this will be equal to  $40$  plus  $40$  into  $60$  divided by  $0.2$  because  $x_2$  minus  $x_1$  in this case is  $0.2$  and which comes out to be  $12040$ , so much millimeter. So, this is the value of  $Z$  that is the depth information and once I have the value of  $Z$ , then the 3D coordinates  $X$  and  $Y$  can be computed from this value of  $Z$ .

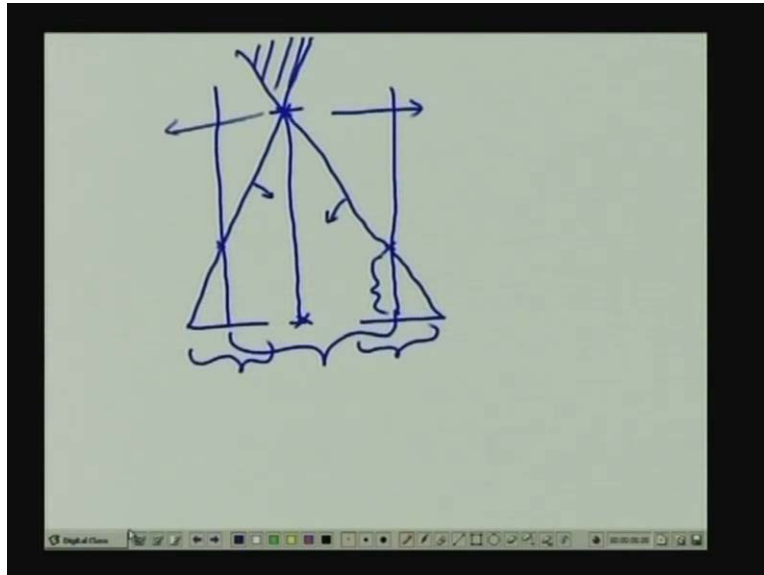
So,  $X$  is nothing but  $x_0$  by  $\lambda$  into  $\lambda$  minus  $Z$  which is nothing but  $0.2$  upon  $40$  into minus  $12000$  which is equal to minus  $60$  millimeter and by applying the same procedure, you can find out  $Y$  equal to minus  $90$  millimeter. So, this is about the first question.

(Refer Slide Time: 54:44)



Second question was to find out what is the minimum depth that can be obtained by using the stereo camera where the geometry of the stereo camera was specified. Now, this is also very simple.

(Refer Slide Time: 55:01)

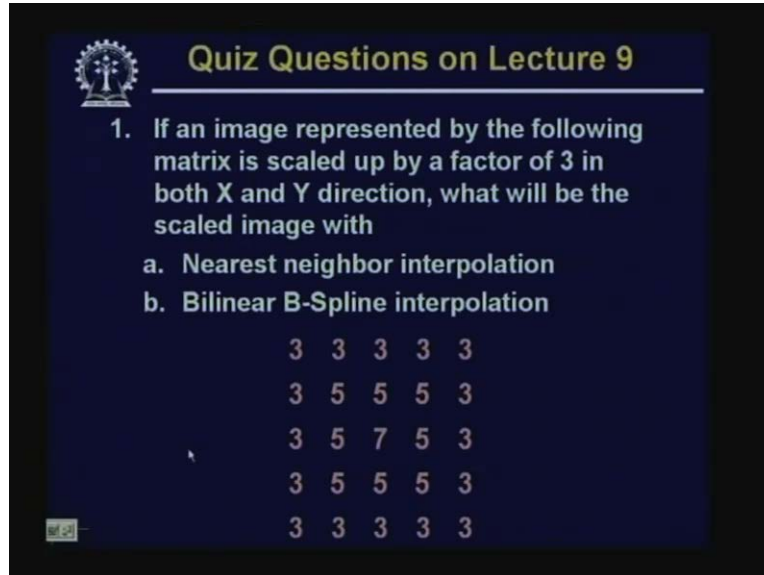


What you can do is you find that we have 2 cameras with certain dimension of the imaging plate. They have been specified certain focal length and if I just find out what is the limit of the **imaging** image points; so we find that if there is any point beyond this particular line, this point cannot be imaged by the left camera. If there is any point in this direction, that cannot be imaged by the right camera because it goes beyond the imaging plate.

And also, for finding out the depth information, it is necessary that the same point should be imaged by both the cameras. So, the points which can be imaged by both the cameras are only the points lying in this particular conical region. The points belonging to this region or the points belonging to this region cannot be imaged by both the cameras. So, all the points must be lying within this. So, the minimum depth which can be computed is this particular depth.

Now, I know what is the separation between the cameras, I know what is the dimension of the imaging planes, I also know what is the focal length. So, from these informations by using the concept of similar triangles, you can easily find out what is the minimum depth that can be computed by using this stereo setup.

(Refer Slide Time: 56:40)



**Quiz Questions on Lecture 9**

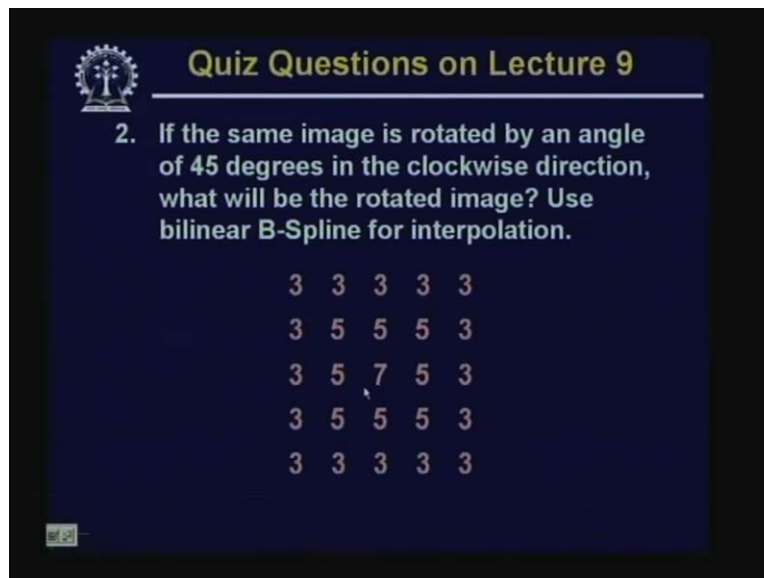
1. If an image represented by the following matrix is scaled up by a factor of 3 in both X and Y direction, what will be the scaled image with

- a. Nearest neighbor interpolation
- b. Bilinear B-Spline interpolation

3	3	3	3	3
3	5	5	5	3
3	5	7	5	3
3	5	5	5	3
3	3	3	3	3

Now, coming to today's questions. So, the question number 1 - if an image represented by the following matrix is scaled up by a factor of 3 in both X and Y directions, what will be the scaled image with nearest neighbor interpolation and bilinear B spline interpolation?

(Refer Slide Time: 57:05)



**Quiz Questions on Lecture 9**

2. If the same image is rotated by an angle of 45 degrees in the clockwise direction, what will be the rotated image? Use bilinear B-Spline for interpolation.

3	3	3	3	3
3	5	5	5	3
3	5	7	5	3
3	5	5	5	3
3	3	3	3	3

The second question - if the same image which is given by this matrix again is rotated by an angle of 45 degrees in the clockwise direction, what will be the rotated image? Use bilinear B spline for interpolation.

Thank you.