Digital Image Processing.

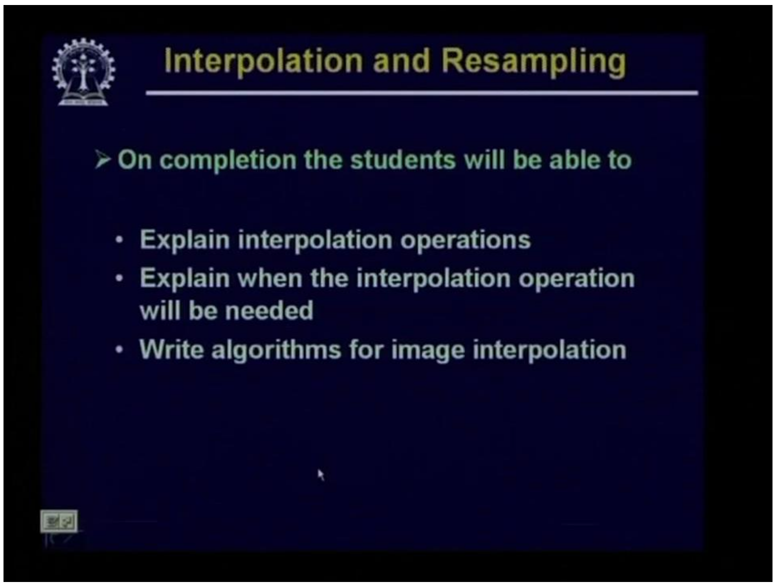
Professor P. K. Biswas.
Department of Electronics and Electrical Communication Engineering.
Indian Institute of Technology, Kharagpur.

Lecture-16.

Interpolation and 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 model and imaging a image formation process. We have also seen that how to calibrate a camera given an particular imaging set up. And, we have also seen that using two identical cameras how we can have a studio imaging setup using which the 3D coordinate of a point in the 3 dimensional scene can be obtained.

(Refer Slide Time: 1:13)

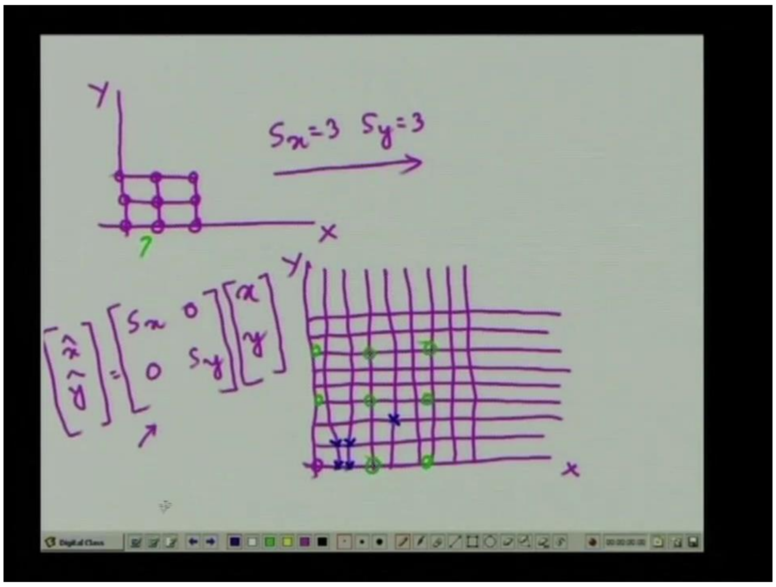


Interpolation and Resampling

- On completion the students will be able to
  - Explain interpolation operations
  - Explain when the interpolation operation will be needed
  - Write algorithms for image interpolation

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 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 desampling.

(Refer Slide Time: 1:42)



y
$S_x=3 \quad S_y=3$
x
$\begin{bmatrix}\hat{x}\\ \hat{y}\end{bmatrix}=\begin{bmatrix}S_x & 0\\ 0 & S_y\end{bmatrix}\begin{bmatrix}x\\ y\end{bmatrix}$
y
x

So, let us first introduce this problem. Say, for example if we have a 3 by 3 image like this. So, we have this XY coordinate system and this XY 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 coordinates 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 up this image by a factor of 3 in both the X dimension and Y dimension. So, if I scale us 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 Sx equal to 3 and Sy is equal to 3. That is both in the X direction and Y direction I am applying a scaling of factor 3. So, naturally this 3 by 3 image after being scaled up by factor 3 in both the directions will be converted to an image of 9 by 9. So, let us see how these pixel values will look like. Again I put this XY coordinate system, now you remember that this scaling operation is given by say x hat, y hat is equal to Sx 0, 0 Sy, into column vector xy.

Where this Sx 0 and 0 Sy, this is the transformation matrix. XY is the coordinate of the pixel in the original image and x hat, y hat, is the coordinate of the pixels in the transforming image. So, if I simply apply this scaling transformation then obviously this (0,0) point will lie at location (0,0) in the transformed image. But what will happen to the other image points, so because this will be converted to a 9 by 9 image.

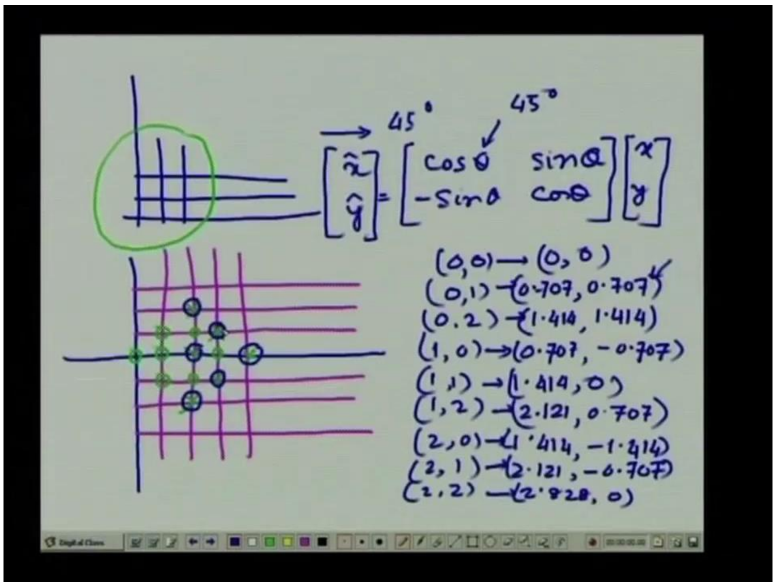
So, let us first form a 9 by 9 grid, ok. So, now you find that this (0,0) pixel even after this scaling transformation remains at location (0,0). But, the other pixels say for example this pixel (1,0) which was originally at location X coordinate equal to 1 and Y coordinate equal to 0 that will be transformed to Y coordinate will remain as 0 but now X coordinate will become equal to 3.

So, this point will be mapped to this particular location. Similarly, (2,0) will be mapped to (6,0) location, so this becomes 1,2,3,4,5,6. So, this pixel will be mapped to this particular location. Similarly, (0,1) pixel will now be mapped to (0,3) location. (0,2) pixel will now be mapped 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 pixel 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 we find that there are many pixels which are not filled up in this scaled up image. So those pixels some of them I can simply mark, say this is 1 pixel which has not been filled up. This pixel has not been filled up, this pixel has not been filled up, this pixel has not been filled up. This point has not been filled up, so likewise there are my many pixels in this particular image in this scaled up image which has not been filled up.

(Refer Slide Time: 8:03)



$\rightarrow 45^\circ \quad 45^\circ$

$$\begin{bmatrix}\hat{x}\\ \hat{y}\end{bmatrix} = \begin{bmatrix}\cos\theta & \sin\theta\\ -\sin\theta & \cos\theta\end{bmatrix}\begin{bmatrix}x\\ y\end{bmatrix}$$

$(0,0) \rightarrow (0,0)$

$(0,1) \rightarrow (0.707, 0.707)$

$(0.2) \rightarrow (1.414, 1.414)$

$(1,0) \rightarrow (0.707, -0.707)$

$(1,1) \rightarrow (1.414, 0)$

$(1,2) \rightarrow (2.121, 0.707)$

$(2,0) \rightarrow (1.414, -1.414)$

$(2,1) \rightarrow (2.121, -0.707)$

$(2,2) \rightarrow (2.828, 0)$

Let us try to take another example, say 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 a by an angel of 45 degree in the clock wise direction. So, if I rotate this image by 45 degree in the clock wise direction you know that we had a transformation matrix which takes care of rotation and that is given by cosine theta, sin theta then – 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 mage. So, in the rotated image I can represent these pixels as x hat, y hat. 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 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 -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 -1.414), (2,1) this will be transformed to location (2.121 and -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 transform locations, you find that the coordinates that we get are not always integer coordinates. In many cases, infact in this particular example in most of the cases the coordinates are real valued 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 a fractional number as a row index or a column index. So, in this case whenever I 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 had 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 mapped 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) is a transformed image. So, what I have to do is 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 mapped 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 transform location is (1.414 and 1.414) again the nearest integer of (1.414) is 1. So this point (0,2) will also be mapped to location (1,1) in the transformed image. (1,0) in the same way will be mapped to location (1,-1), so (1,0) will be mapped to this particular point in the transformed image.

(1,1) will be mapped to (1.4140) here again by integer approximation this point will be mapped to (1,0) location in the transformed image. So, I have a pixel in this particular location, (2,1) point will be mapped to (2.121 and .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.

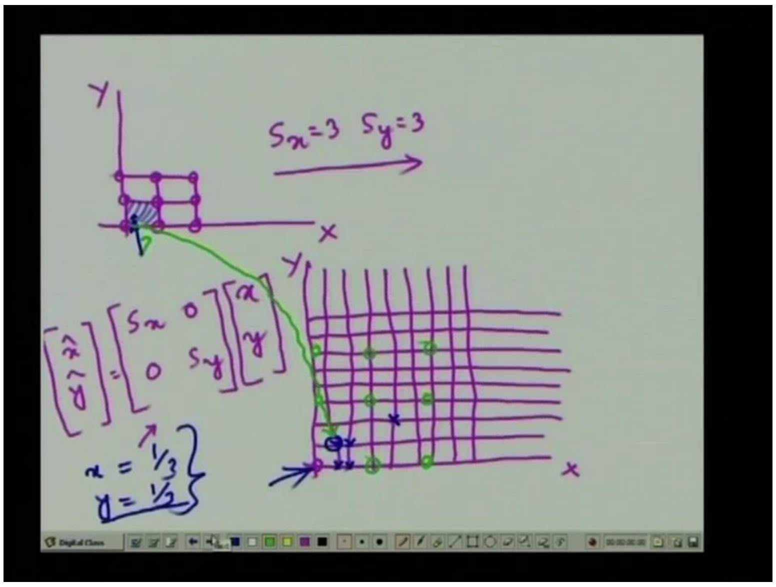
In the same manner (2,0) point will be mapped to location (1,-1) where I already have a particular point, (2,1) location will be mapped to location (2,-1). So, I will have a point here, (2,2) location will be mapped to (3,0) in the transformed image. So, I will have a pixel in this particular location. So we 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 the turns out to be fractions or real numbers in the 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 of 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, ok. 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 information.

(Refer Slide Time: 17:08)


y
x
$S_x=3 \quad S_y=3$
$\begin{bmatrix}\hat{x}\\\hat{y}\end{bmatrix}=\begin{bmatrix}S_x & 0\\0 & S_y\end{bmatrix}\begin{bmatrix}x\\y\end{bmatrix}$
$x = 1/3$
$y = 1/3$
y
x

So, to take the simple case take the previous one. Here, you find that I do not any information at location (1,1) of the transformed image or 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 the original image then to this transformed image I have to apply a scaling of one third in both X direction and Y direction. Now we find that in the transform image this particular location, this particular pixel has a coordinate of (1,1). So, if I transform this, inverse transform this using the scaling factor of one third and one 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.

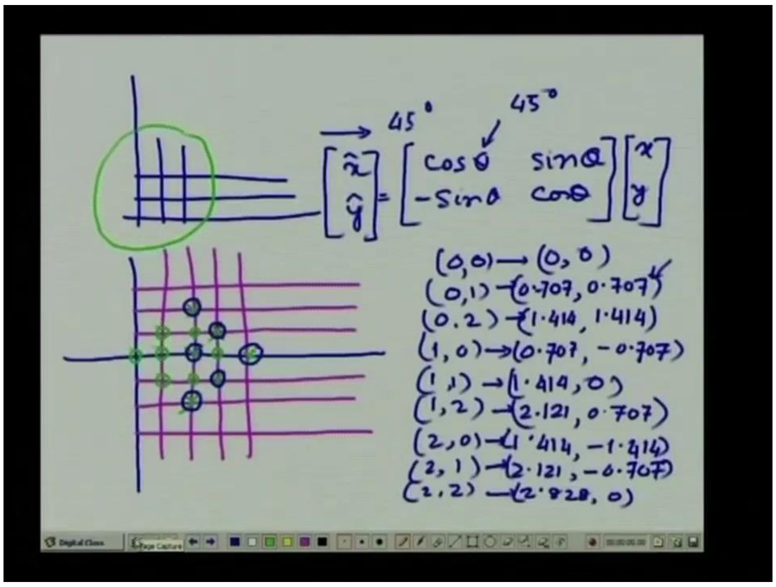
Now, here comes the problem. In the original image I have the informations at location (0,0), I have the information location information at location (0,1) I have information at location (1,0), I have information at location (1,1). But at location (1,3) and (1,3) I do not have any information. Because 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 was quantization.

Now, the moment we sample the image is, 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 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 these 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 these particular value has to be taken to fill up this location (1,1) in the transformed image.

(Refer Slide Time: 20:34)



$$\begin{bmatrix} \hat{x} \\ \hat{y} \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

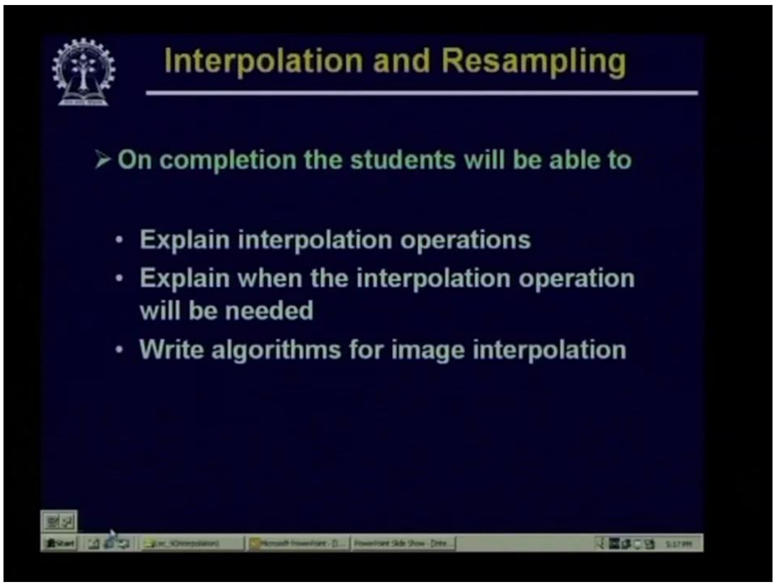
$(0,0) \rightarrow (0,0)$
$(0,1) \rightarrow (0.707, 0.707)$
$(0.2) \rightarrow (1.414, 1.414)$
$(1,0) \rightarrow (0.707, -0.707)$
$(1,1) \rightarrow (1.414, 0)$
$(1,2) \rightarrow (2.121, 0.707)$
$(2,0) \rightarrow (1.414, -1.414)$
$(2,1) \rightarrow (2.121, -0.707)$
$(2,2) \rightarrow (2.828, 0)$

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 clock wise 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 -45 degree go back to the original image point and obviously in this case in the original image these row column indices will not be integers but they will be real numbers or fractions.

And, because we have real numbers 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 is, 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.

(Refer Slide Time: 21:43)

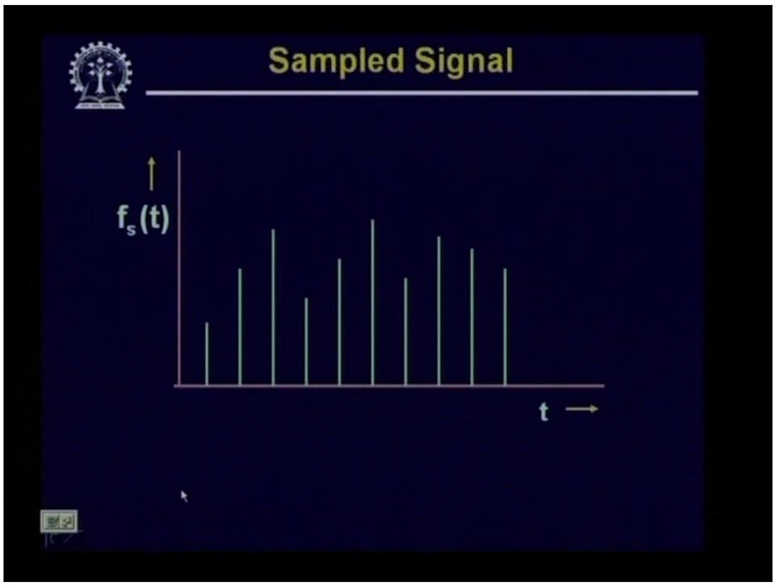


Interpolation and Resampling

- On completion the students will be able to
  - Explain interpolation operations
  - Explain when the interpolation operation will be needed
  - Write algorithms for image interpolation

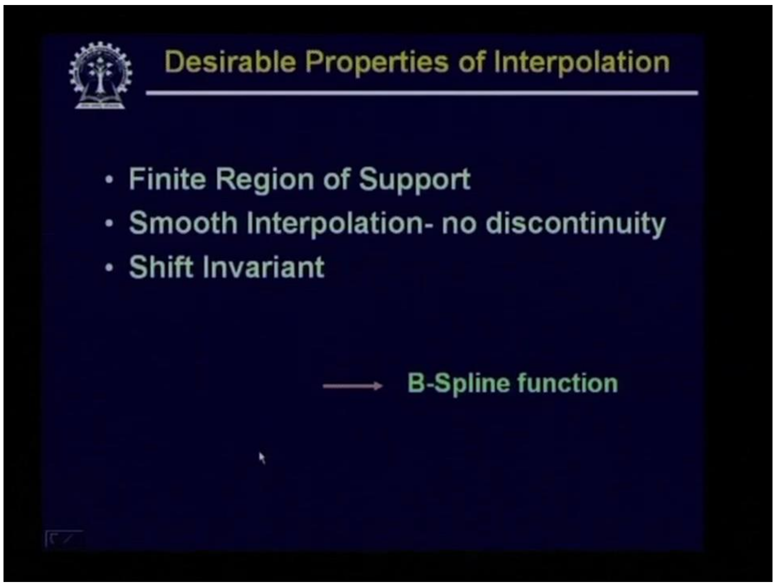
So, this is why you find that the interpolation and resampling is very very important whenever you are working in the digital domain or you are doing some sort of transformations over a digital image.

(Refer Slide Time: 22:03)



Now, whenever we go for interpolation so this gives the situation that we have a one dimensional signal f signal ft of function t and after sampling we have got the sampled signals fst.

(Refer Slide Time: 22:31)



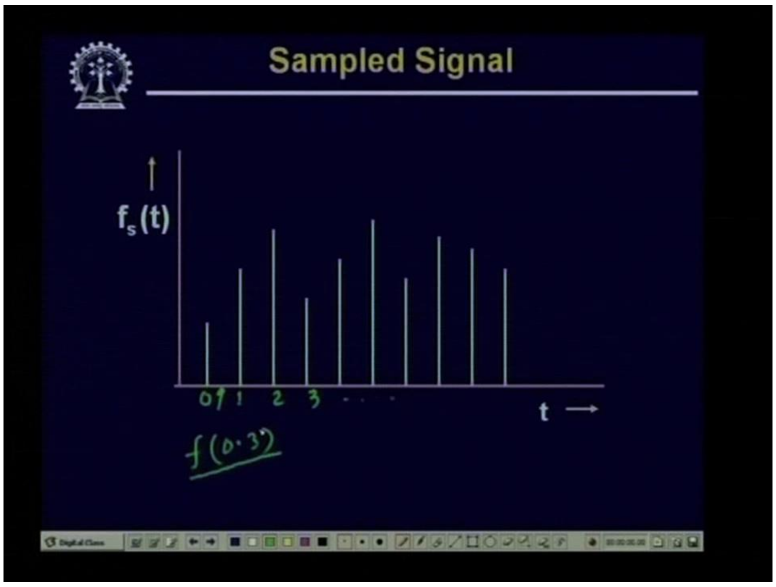
Desirable Properties of Interpolation

- Finite Region of Support
- Smooth Interpolation- no discontinuity
- Shift Invariant

$\longrightarrow$ B-Spline function

So, here we find that after sampling what we have got is the sample values which are represented by fst. Now, as we have fst 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 ft. And because we do not have these informations we have to go for interpolation for all those values of t where we do not have the samples present.

(Refer Slide Time: 23:29) 24:59



Sampled Signal

And after interpolation again we have to go for resampling to fill up those positions. So, this slide shows a sampled one dimensional signal ft of a function t. So after sampling we have represented the signal by fst, where fst 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 3 2, 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 fst 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 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 when if it is only translation does not need any kind of resampling or interpolation, thank you.