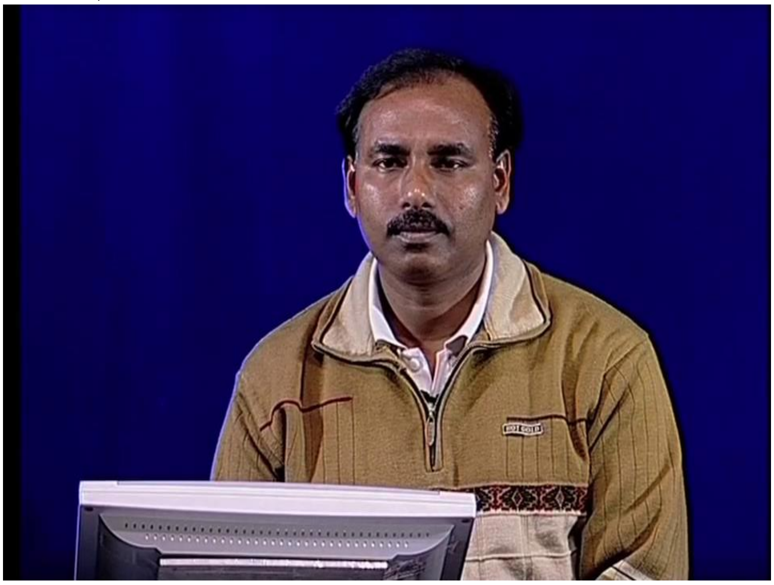
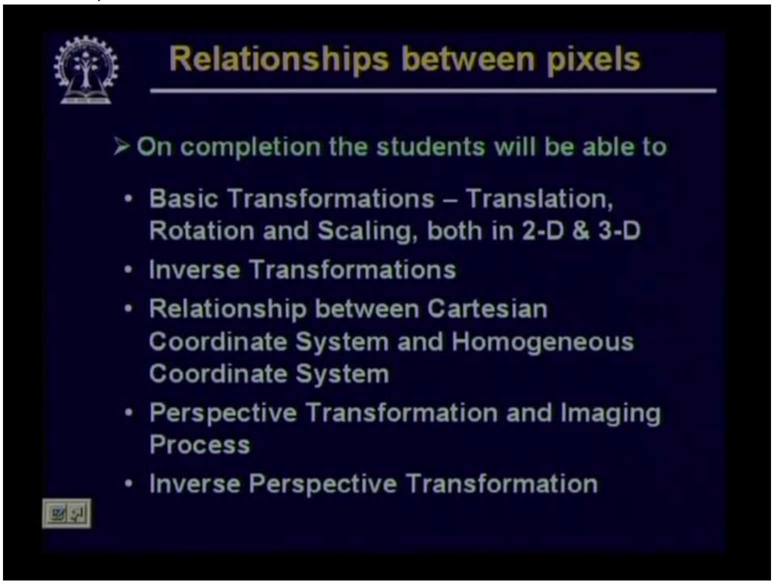
Digital Image Processing
Prof. P. K. Biswas
Department of Electronics and Electrical Communications Engineering
Indian Institute of Technology, Kharagpur
Module 02 Lecture Number 10
Basic Transform

(Refer Slide Time 00:19)



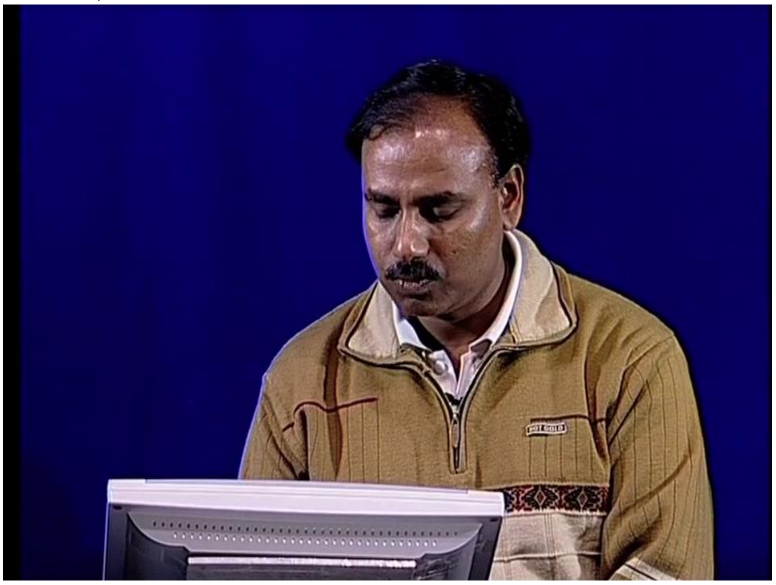
Hello, welcome to the video lecture series on Digital Image Processing. In today's lecture

(Refer Slide Time 00:28)



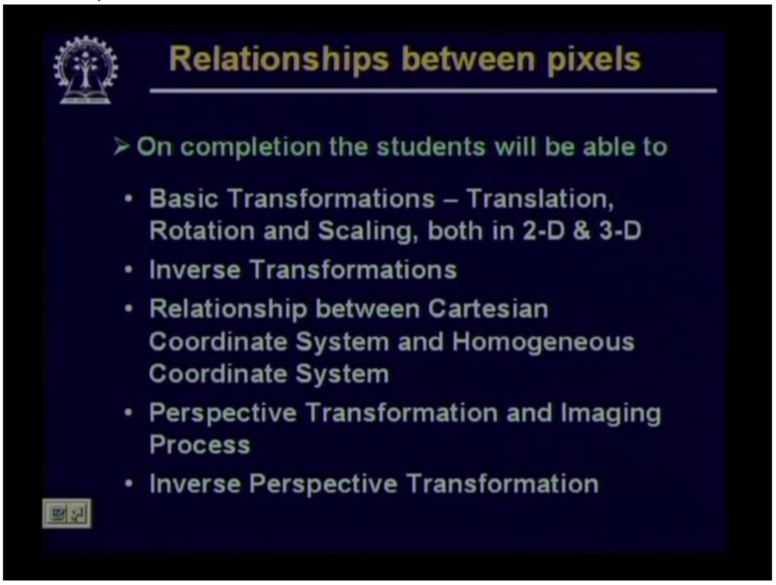
as we said that we will discuss about some basic mathematical transformations which will include translation, rotation and scaling and this we will discuss both in two-dimension as well as in three-dimension. We will also discuss about the inverse transformations of these different mathematical transformations. We will find out the relationship between the cartesian and an homogenous coordinate system and we will see that this homogenous coordinate system is very, very useful while discussing about the image formation

(Refer Slide Time 01:09)



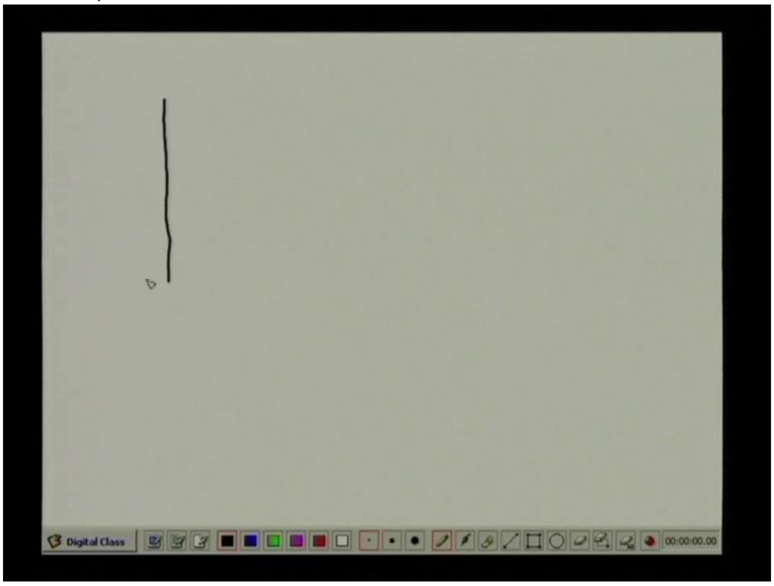
by a camera We will also talk about the perspective transformation and the imaging

(Refer Slide Time 01:15)



process and then we will talk about the inverse perspective transformation. Now coming to the basic mathematical transformations let us first talk about that what is the translation operation and we will start our discussion with a point in

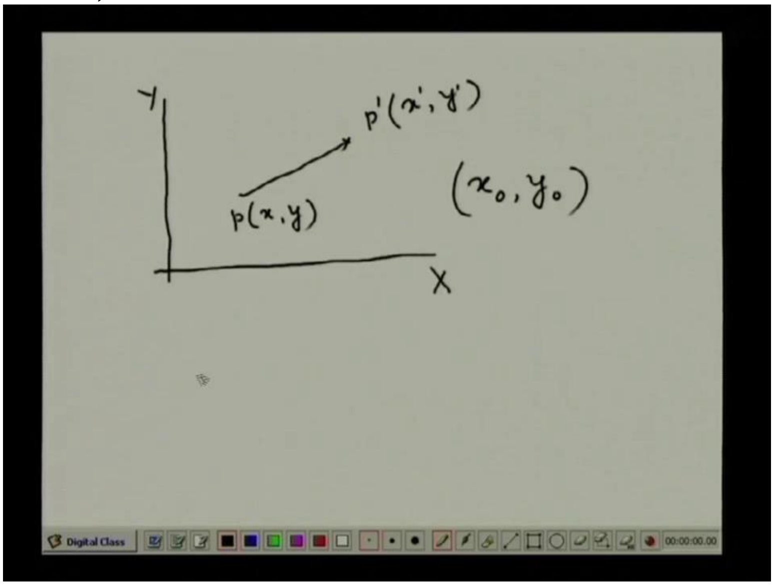
(Refer Slide Time 01:40)



two-dimension

So you know that if I have a two-dimensional coordinate system given by the axes x and y and if I have a point p which is having a coordinate given by say, x y. And I want to translate this point p x y by a vector x naught y naught so after translating this point by the vector x naught y naught I get the translated point say at point p prime whose coordinates are x prime and y prime. And because the translation vector in this case we have assumed as x naught y naught so you know that after translation the new position

(Refer Slide Time 02:36)



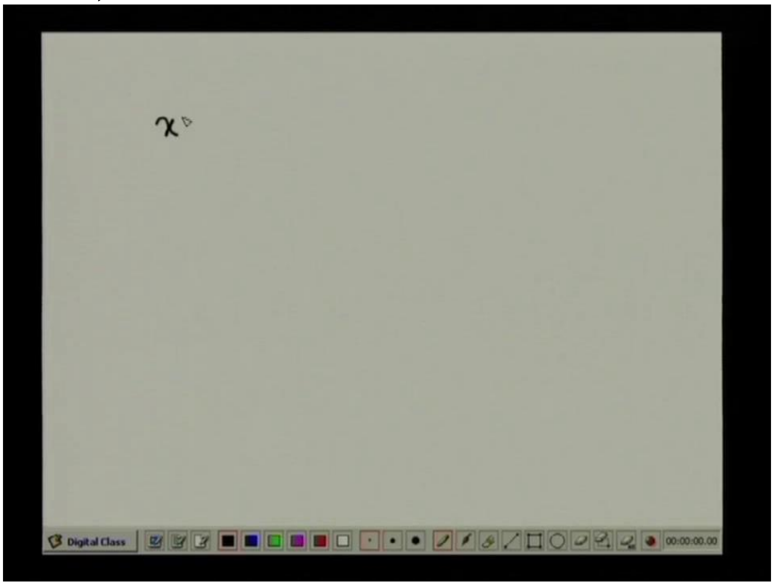
x prime will be given by x plus x naught and y prime will be given by y plus y naught. Now this is the basic relation when a point at location x y is translated by a vector

(Refer Slide Time 02:58)

x naught y naught

Now let us see that how this can be represented more formally by using a matrix equation. So if I translate this equation in the form of a matrix, the equation looks like this. I have to find out the new

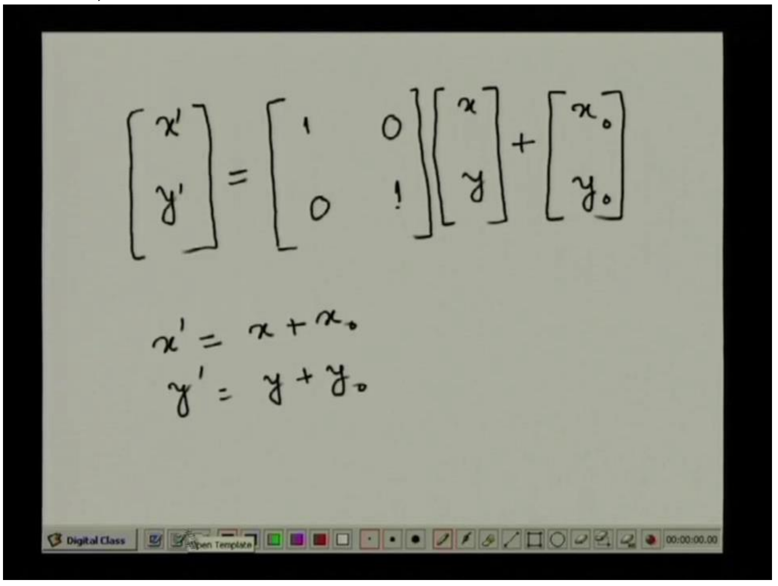
(Refer Slide Time 03:18)



location vector x prime y prime and we have said that this x prime is nothing but x plus x naught and y prime is nothing but y plus y naught. So this particular relation, if I represent in the form of a matrix, it will simply look like this.

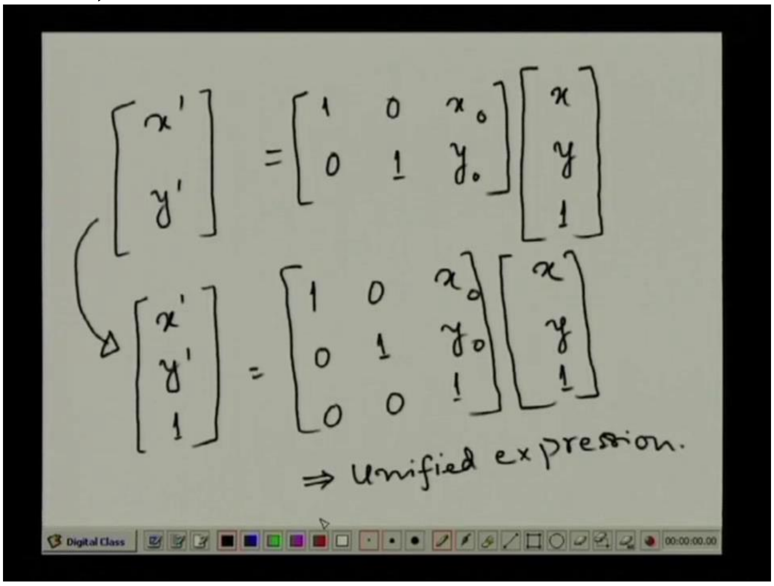
So you find that if you solve this particular matrix expression it gives you the same expression x prime equal to x plus x naught and y prime is equal to y plus y naught. So on the right hand side you find that I have product of two matrices which is added to another column matrix or column vector. Now if I want to combine all these operations in a single matrix form

(Refer Slide Time 04:32)



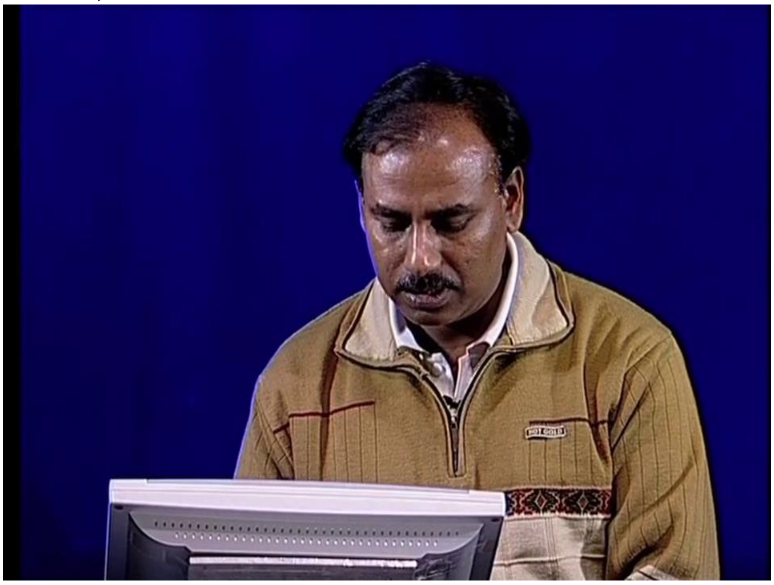
then the operation will be something like this. On the left hand side I will have x prime and y prime which will be in the form of a matrix and on the right hand side, I will have 1 0 then x naught 0 1 y naught and then I will have x y and 1. So if I again do this same matrix computation it will be x prime equal to x plus 0 plus x naught so which is nothing but x plus x naught similarly y prime will be 0 plus y plus y naught which is nothing but y plus y naught. But you find that in this particular case, there is some asymmetry in this particular expression. So if I want to make this expression symmetric, then I can write it in this form, x prime y prime and I introduce one more component which I make equal to 1 this is equal to 1 0 x naught 0 1 y naught then 0 0 1 and x y 1. So find that this second expression which I have just obtained from the first one is now a symmetric expression and this is what is called an unified expression. So find that basically what I have is

(Refer Slide Time 06:34)



I had the original coordinate x y of the point p which is appended with one more component that is given as 1 and if this modified coordinate is now transformed by a transformation matrix which is given as 1 0 x naught 0 1 y naught and 0 0 1, then I get the translated point as x prime y prime 1 where if I just neglect the additional component which in this case is 1, then I get the translated point

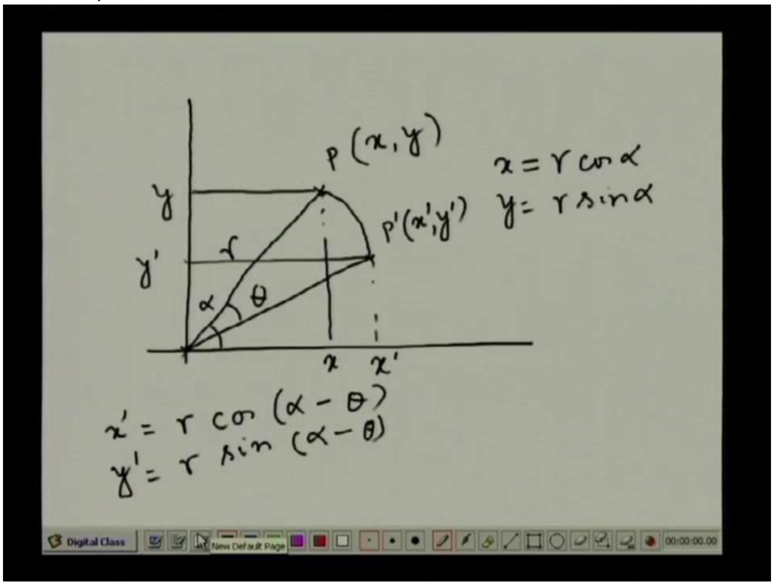
(Refer Slide Time 07:10)



p prime so this is about the translation

In the same manner, given a point p again in 2 D so again I have this point p which is having again having a coordinate x y and suppose I want to rotate this point p around the origin by an angle theta now one way of representing this point p is if r is the distance of point p from the origin then these are the coordinates of the point p, this is the x coordinate, this is the y coordinate so I can also represent and suppose this angle is alpha, then I can also represent x as x equal to r cosine alpha and y equal to r sine alpha. Now suppose I want to rotate this point p by angle theta in the clockwise direction, so the new position of p will now be p prime having the coordinate location x prime and y prime and this rotation angle angle of rotation is now angle theta. So our job is that what will be these points the coordinate points x prime and y prime. So here you find that I can write this x prime as r cosine alpha minus theta and I can write y prime as r sine alpha minus theta. Ok, so if I simply

(Refer Slide Time 09:15)



expand this so I have x prime is equal to r cosine alpha minus theta and y prime is equal to r sine alpha minus theta. So if I simply expand this cosine term it will simply be r cosine alpha cosine theta plus r sine alpha sine theta. Now we know that r cosine alpha is nothing but x so it becomes x cosine theta plus r sine alpha is nothing but y so it becomes y sine theta. Similarly in this case, if I expand this, it becomes r sine alpha cosine theta minus r cosine alpha sine theta. So again r sine alpha this is nothing but y so this takes the expression y cosine theta minus x sine theta. So again so here you find that x prime is given x cosine theta plus y sine theta and y prime is given by minus x sine theta plus y cosine theta. So even now if I represent this in a form of a matrix equation, it becomes x prime y prime is equal to cosine theta sine theta then minus sine theta cosine theta and then i have the original coordinates x and y. So here you will find that if I rotate the point p by

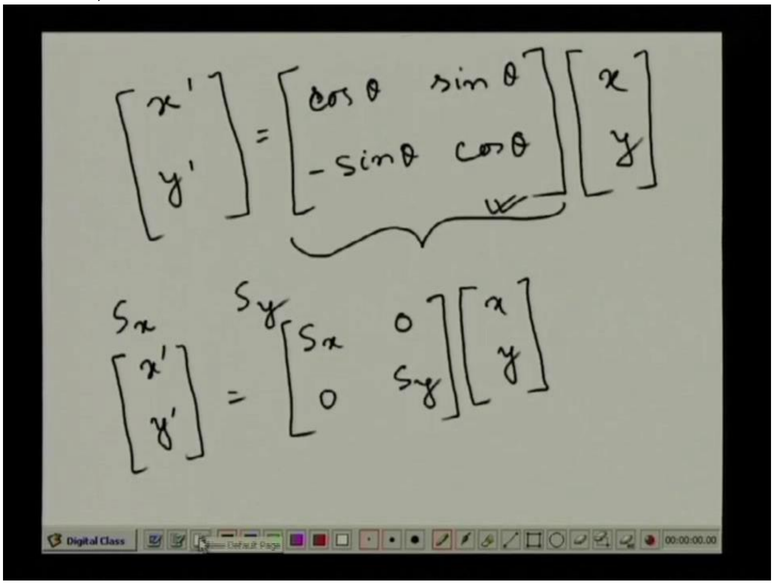
(Refer Slide Time 11:21)

an angle theta around the origin in the clockwise direction, in that case the transformation matrix which gives you the rotation transformation is given by this particular matrix which is cosine theta sine theta minus sine theta cosine theta.

Now in the same manner if I go for scaling say for example if I have a scaling s x scaling factor of s x in the x direction and I have scaling factor s y in the y direction, in that case the transformation matrix for scaling can also be represented as x prime y prime is equal to s x 0 0 s y and which is multiplied by the original coordinate x y. So here you find that the transformation matrix for performing the scaling operation is nothing but the matrix s x 0 and 0 s y. So these are the simple transformations that I can have in two-dimension. Now it is also possible to concatenate the transformations.

For example, here I have considered the rotation of a point around the origin. Now if my application demands that I have to rotate the point p around an arbitrary q in the two-dimension. Then finding out the expression for rotation of this point p by an angle theta around another point q is not an easy job, I mean that expression will be quite complicated. So I can simplify this operation just by translating the point q to the origin and the point p also has to be translated by the same vector and after performing the transformation, the translation, if I now rotate point p by the same angle theta and now it will be the rotation around the origin so whatever expression that we have found here that is cosine theta sine theta minus sine theta cosine theta, the same transformation matrix will be applicable and after getting this rotation, now you translate back the rotated point by the same vector but in the opposite direction. So here

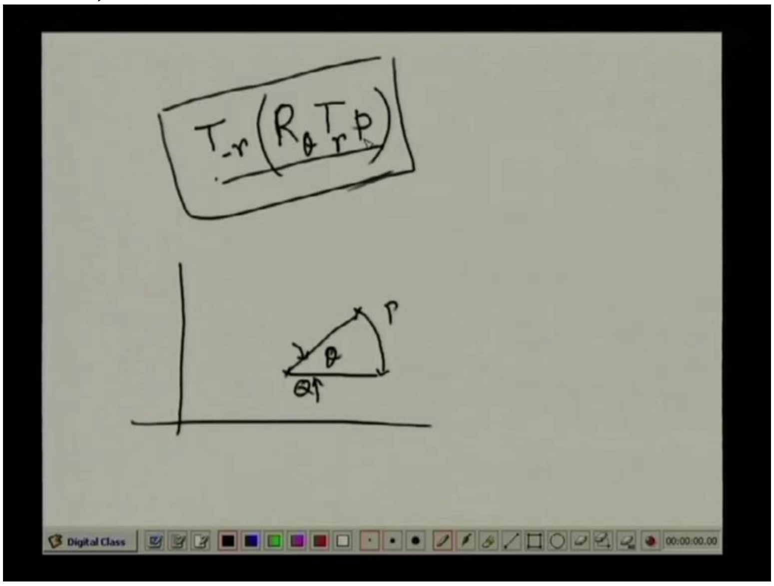
(Refer Slide Time 13:45)



the transformations that we are applying is first we are performing a translation of point p by the vector and after performing this translation we are performing the rotation so this is your transformation say R theta so first we are translating by a vector say R then we are performing rotation by vector R theta and after doing this, whatever point I get, that has to be translated back by minus R, so I will put it as translation by the vector minus r.

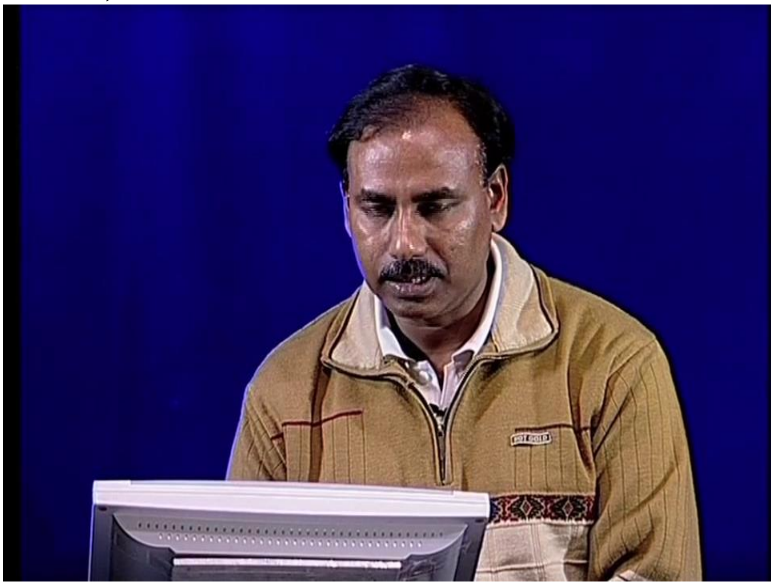
So this entire operation will give you the rotation of a point p, suppose this is point p and I want to rotate this around the point q. So if I want to rotate p around q by an angle theta then this operation can be performed by concatenation of this translation, rotation then followed by inverse translation which puts back the point to its original point where it should have been after rotating around

(Refer Slide Time 15:04)



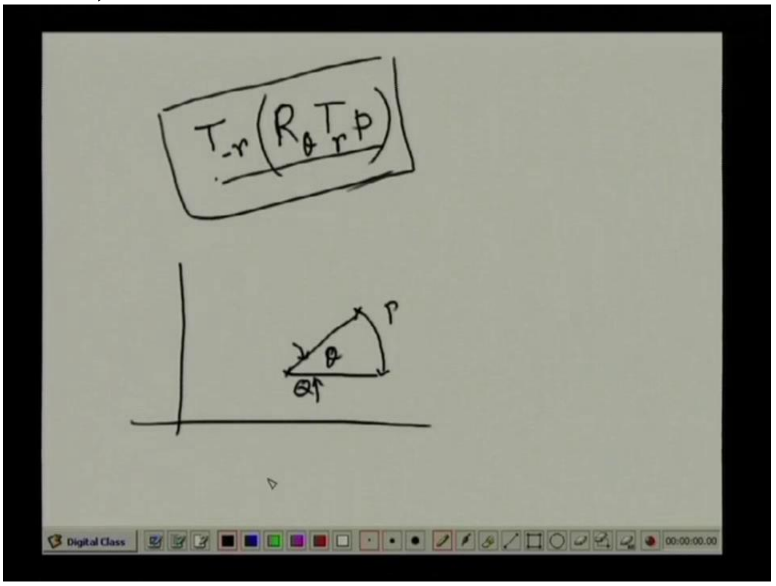
point q by angle theta

(Refer Slide Time 15:07)



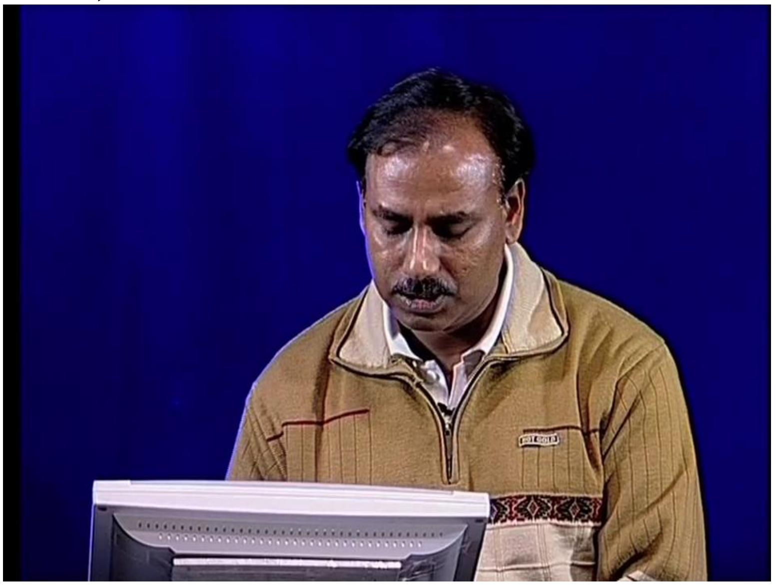
So these are the different transformations, the basic mathematical transformations that we can do in a two-dimensional space. Now let us what will be the corresponding

(Refer Slide Time 15:19)



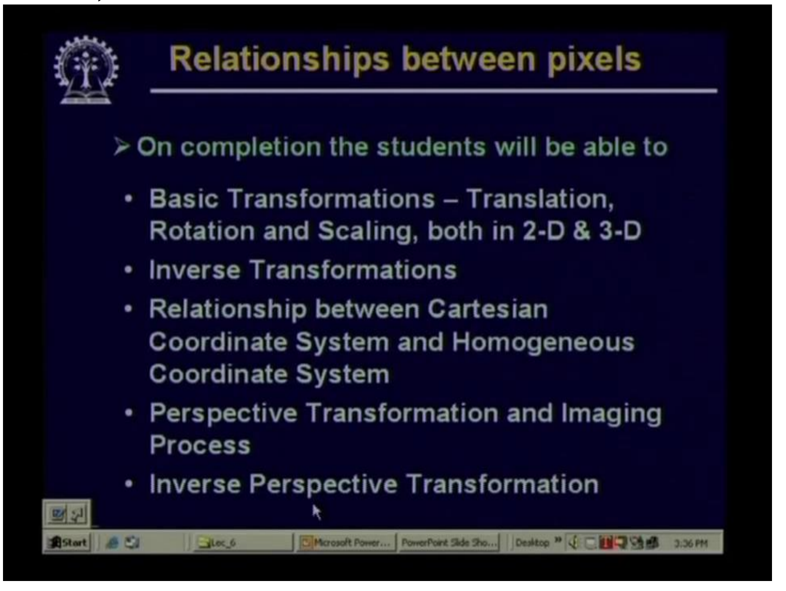
transformations if I move from two-dimensional space

(Refer Slide Time 15:23)

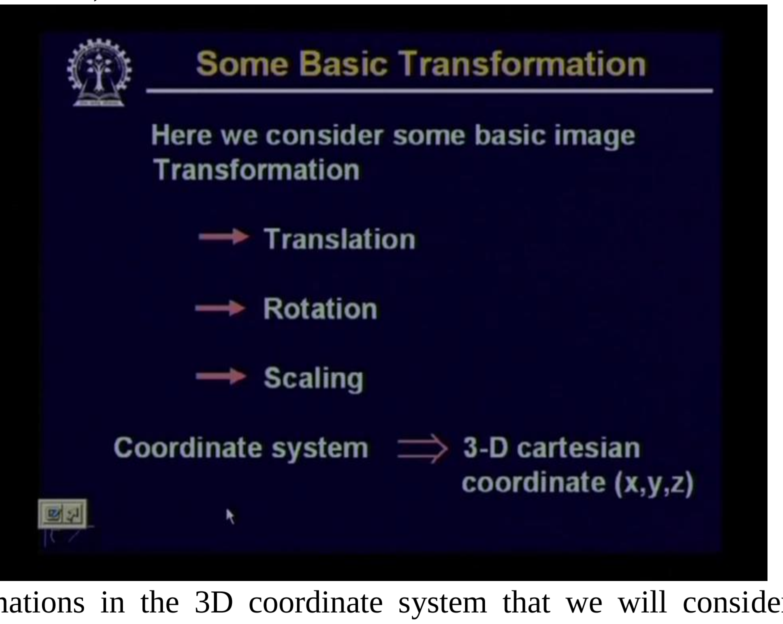


to the three-dimensional space

(Refer Slide Time 15:26)

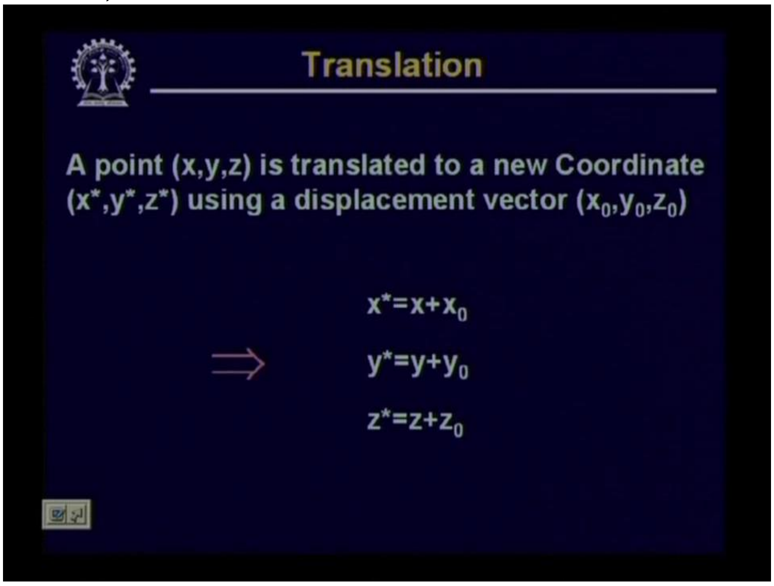


(Refer Slide Time 15:28)



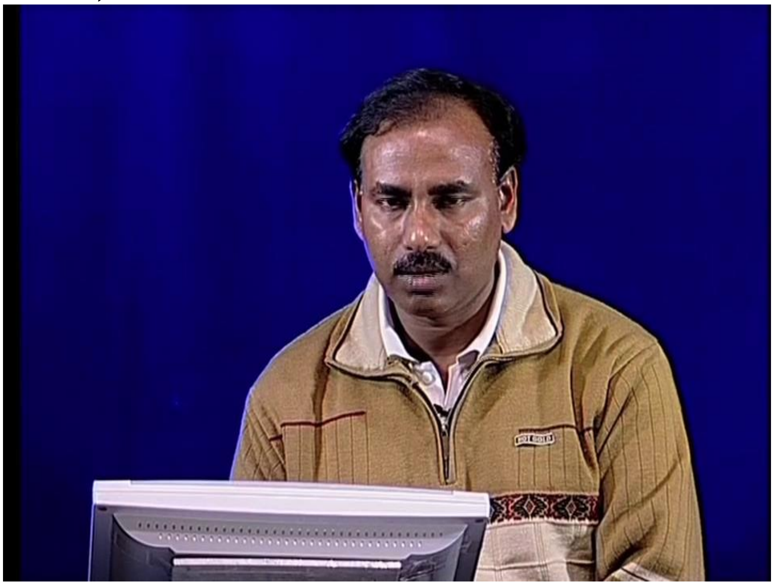
So the transformations in the 3D coordinate system that we will consider is translation, rotation and scaling and the coordinate system that we will consider in this case is three-dimensional coordinate system.

(Refer Slide Time 15:45)



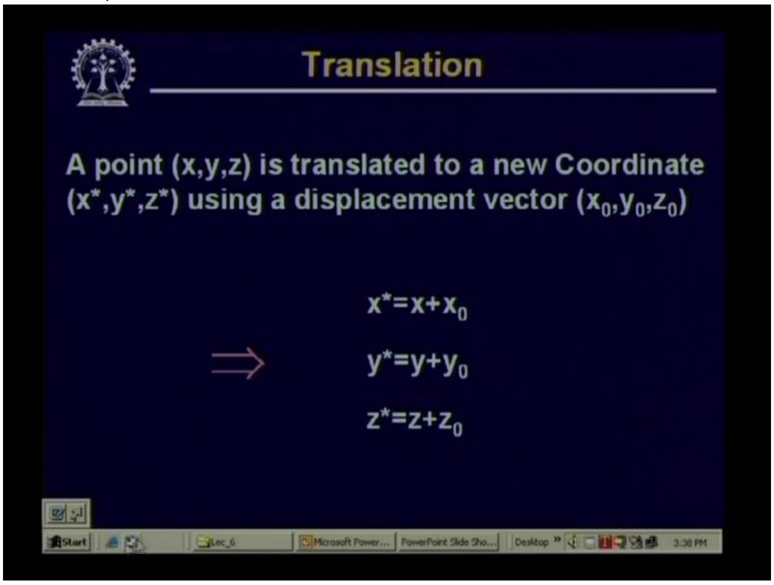
So first, let us see as you have seen in case of two-dimension, that if a point x, y, z is, is translated to a new coordinate say x star y star z star using a displacement x naught y naught z naught, then this translated coordinates x star will be given by x plus x naught, y star will be given by y plus y naught and z star will be given by z plus z naught. So you see that in our previous case we have said that, because we had only the coordinates x and y so this third expression z star is equal to z plus z naught that was absent. But now we are considering the three-dimensional space, a 3 D coordinate system, so we have 3 coordinates x, y and z and all these 3 D coordinates, all these 3 coordinates are to be translated by the translation vector x naught y naught z naught and the new translation vector, and the new point we get as x star y star and z star.

(Refer Slide Time 16:52)



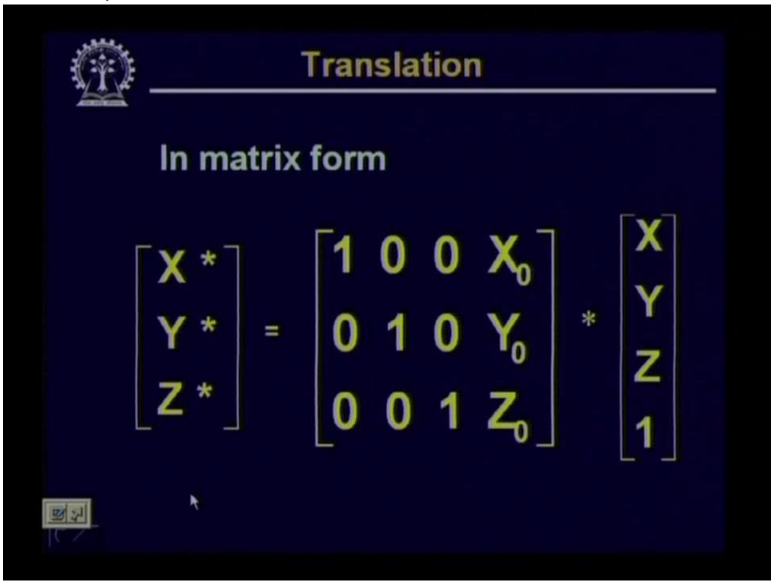
Now if I write these 3 equations in the form of a matrix then the matrix equation will be

(Refer Slide Time 17:03)



like this.

(Refer Slide Time 17:05)

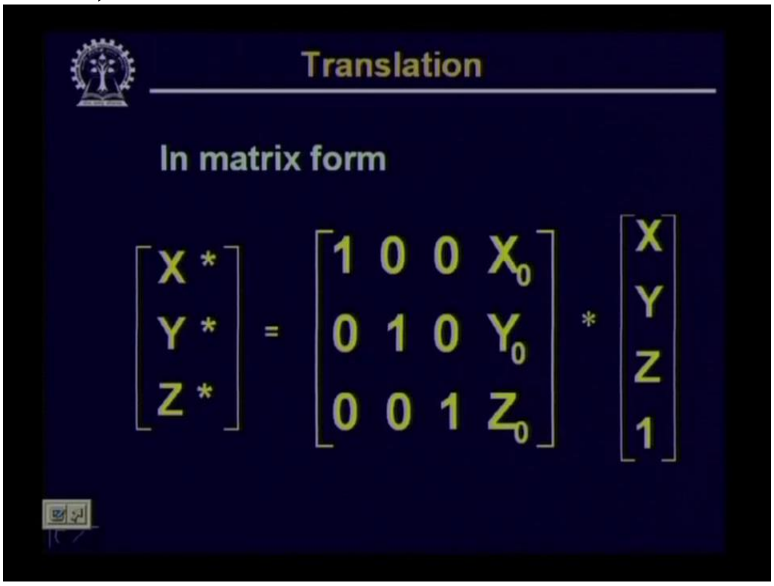


x star y star z star on the left hand side will be equal to 1 0 0 x naught 0 1 0 y naught 0 0 1 z naught into the column vector x y z 1. So this is the similar situation that we have also seen in case of two-dimension

(Refer Slide Time 17:27)

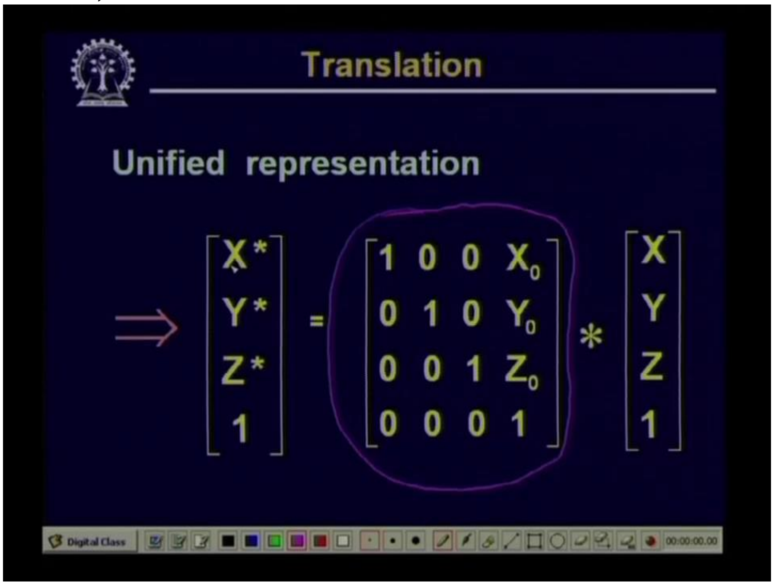
that we have to add an additional component which is equal to 1 in our original position vector x y z so in this case again we have added the additional component which is equal to 1, so our next position, our new position vector becomes x y z 1 which has to be multiplied by the translational matrix given by 1 0 0 x naught 0 1 0 y naught and 0 0 1 z naught. So again as before we can go for an unified expression where this translation matrix which at this moment is having a dimension 3 by 4 that is, it is having 3 rows and 4 columns, in unified representation we represent this matrix, the dimension of the matrix will be 4 by 4 which will be a square matrix and the left hand side also will have the same unified coordinate, that is x star y star z star 1.

(Refer Slide Time 18:32)



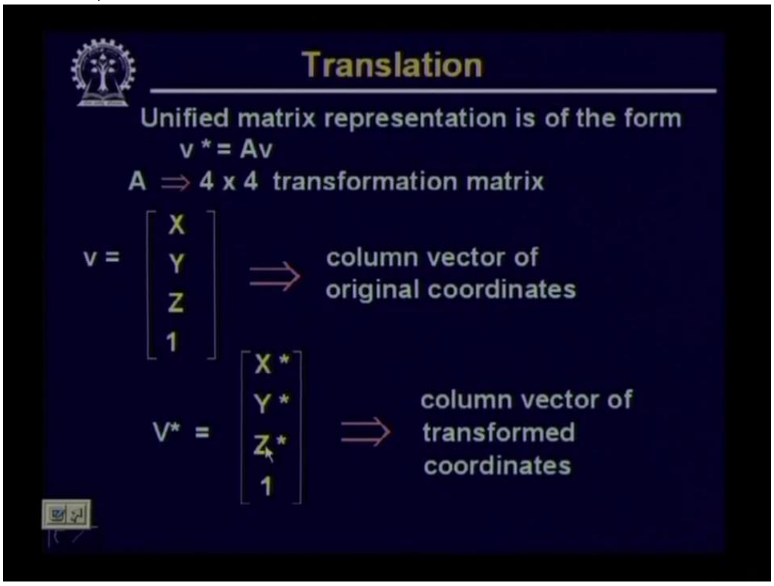
So the unified representation of as we have already said is given by x star y star z star 1 is equal to the translation matrix 1 0 0 x naught, 0 1 0 y naught, 0 0 1 z naught, 0 0 0 1 multiplied by the column vector x y z 1. So this particular matrix that is 1 0 0 x naught, 0 1 0 y naught,

(Refer Slide Time 19:07)



0 0 1 z naught and 0 0 0 1 this represents a translation, a transformation matrix used for the translation and we will represent this matrix by this upper case letter T. So that is the simple translation that we can have

(Refer Slide Time 19:31)

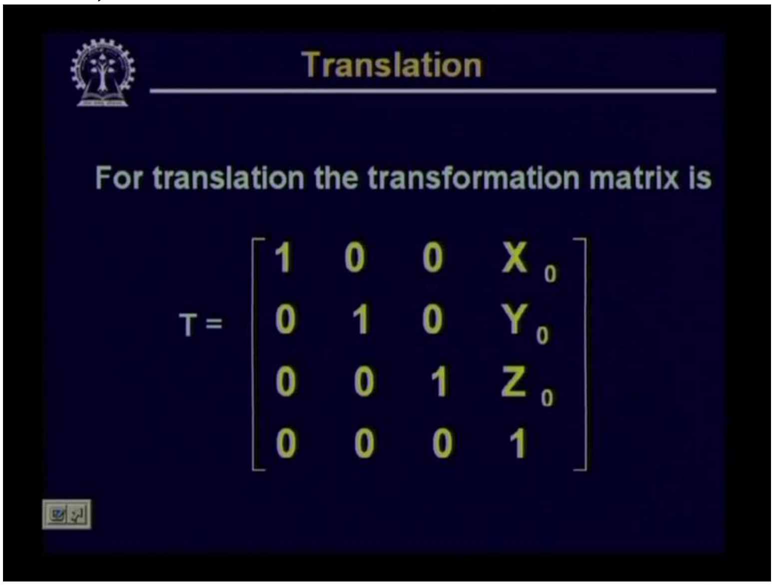


So in our unified matrix representation, we have done if you have a vector v, a position vector v which is translated by the transformation matrix A, the transformation matrix A is a 4 by 4 transformation matrix, the v, if the original position vector was x y z we have added an additional component 1 to it in our unified matrix representation, so v now becomes a four-dimensional vector having components x y z and 1. Similarly the transformed position vector v star is also a four-dimensional vector which is having components x star y star z star and 1. So this how in the unified matrix representation, we can represent the translation of a position vector or

(Refer Slide Time 20:29)

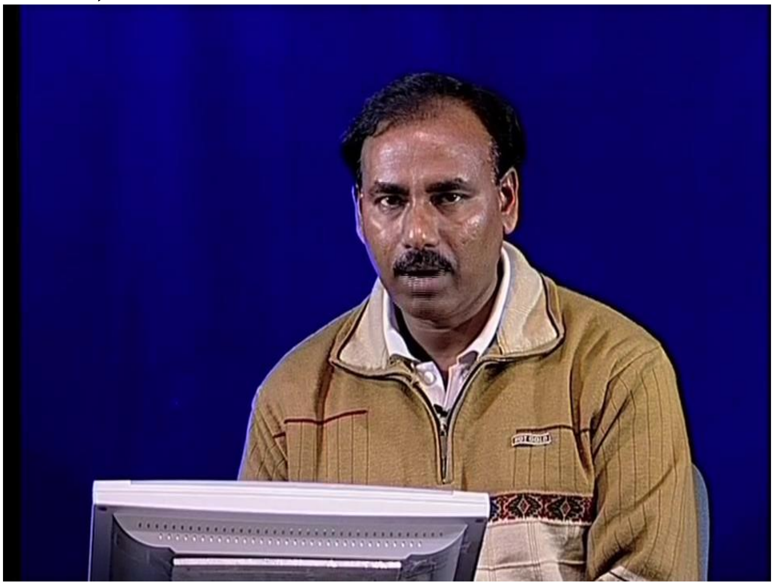
a translation of a point in three-dimension Similarly we can have so as I said

(Refer Slide Time 20:36)



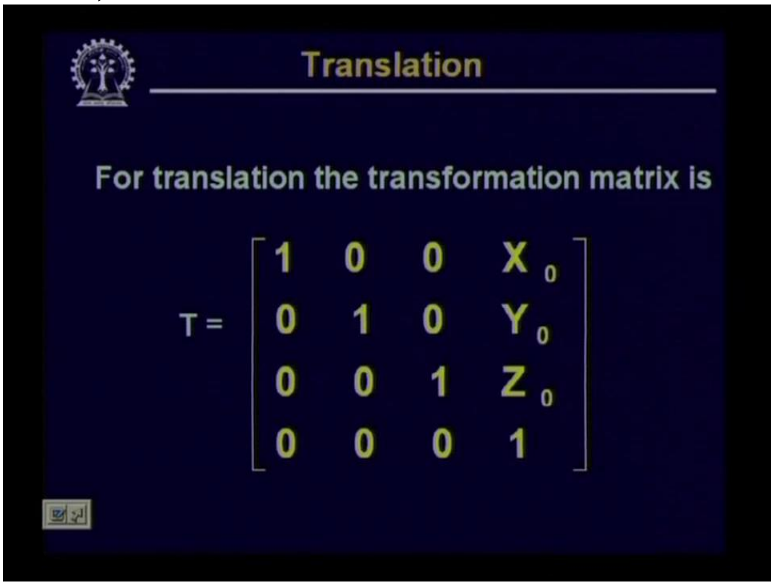
that this is the transformation matrix which is represented, which is used for translating a point in 3 D by

(Refer Slide Time 20:45)



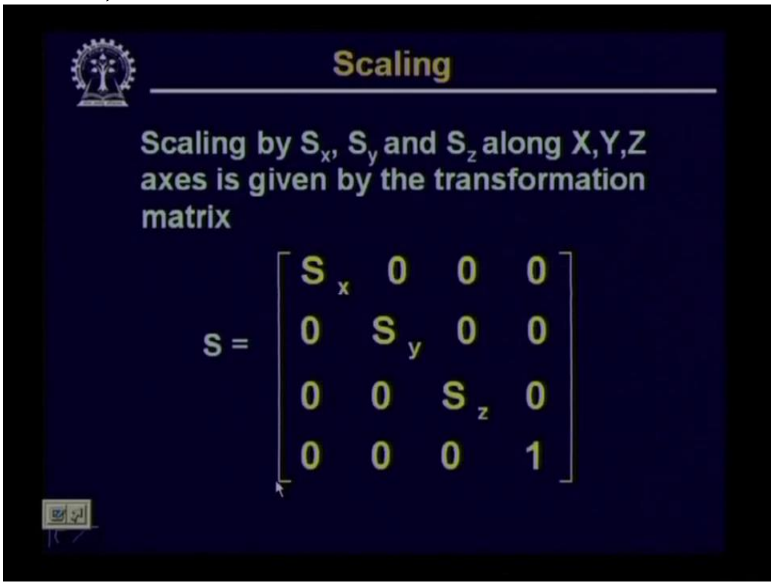
vector x 1, x naught y naught z naught

(Refer Slide Time 20:50)



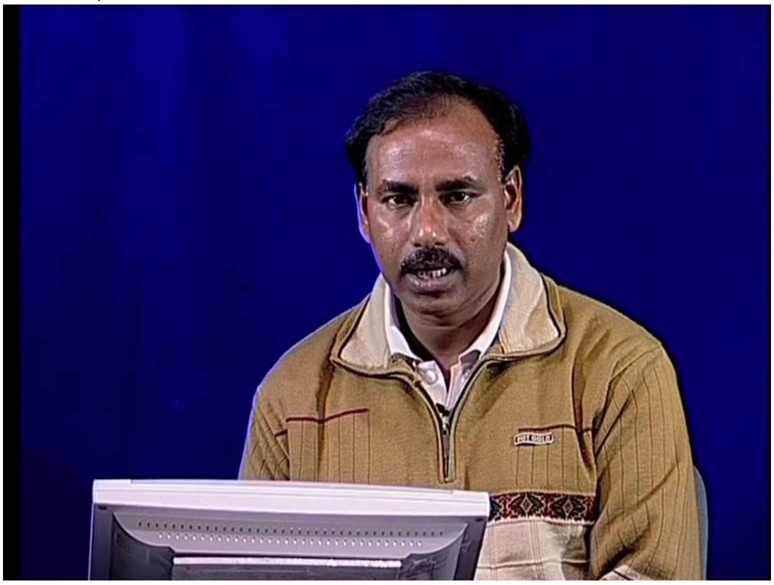
or the displacement vector x naught y naught z naught

(Refer Slide Time 20:56)



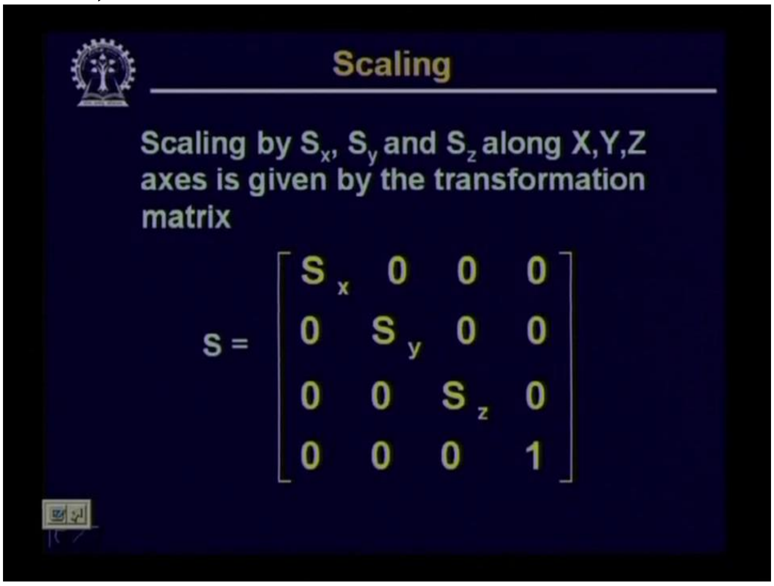
Similarly as we have seen in case of scaling in two-dimension that if we have the scaling factor of s x s y and s z along the directions x y and z .so along direction x, we have the scaling factor s x. Along direction y, we have the scaling factor

(Refer Slide Time 21:17)



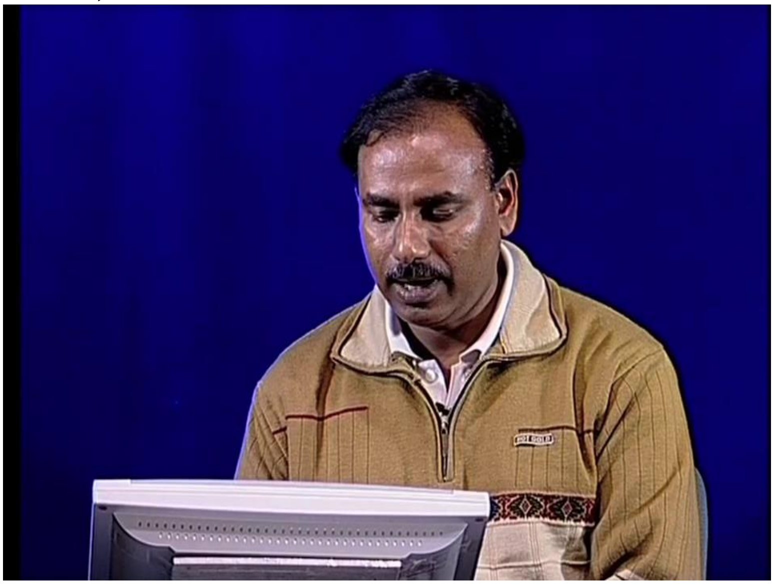
s y And

(Refer Slide Time 21:29)



along the direction z we have the scaling factor s z. Then the transformation matrix for this scaling operation can be written by S equal to x nau, s x 0 0 0 then 0 s y 0 0 then 0 0 s z 0 then 0 0 0 1. So here again if you find, you find

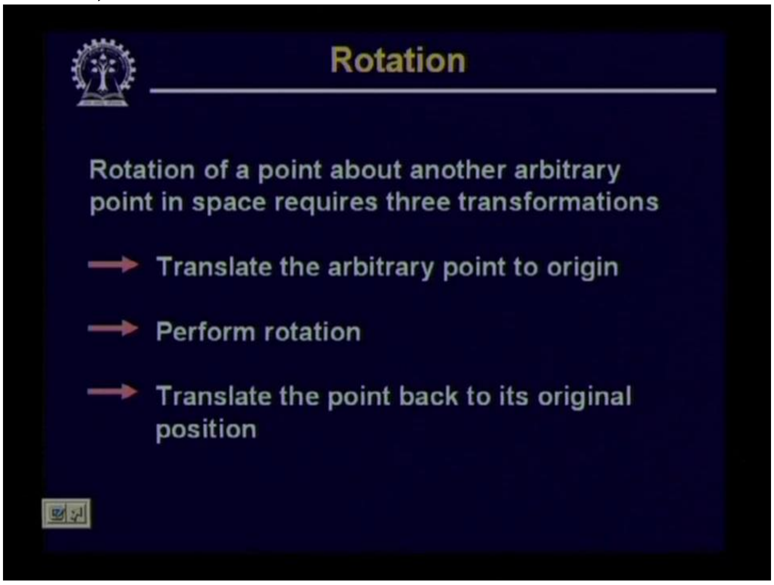
(Refer Slide Time 21:45)



that a position vector x y z in unified form it will be x y z 1. If that position vector is translated by this scaling matrix, then what we get is the new position vector corresponding to point x y z 1 in the scaled form and there if we remove the last component that is equal to 1, what we get is the scaled 3 D coordinate of the point which has been scaled up or scaled down. So it will be scaling up or scaling down depending upon whether the value of the scale factors are greater than 1 or they are less than 1.

Then coming to rotation

(Refer Slide Time 22:30)



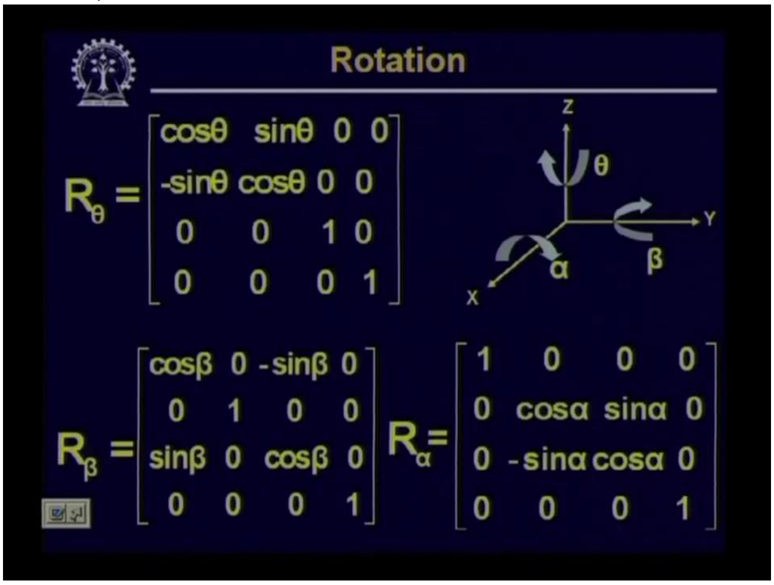
We have seen that the translation and scaling

(Refer Slide Time 22:35)

in three-dimension is very simple. It is as simple as we have done in case of two-dimension. But the rotation in three-dimension is a bit complicated. Because in three-dimension as we have 3 different axis, x axis, y axis and z axis so when I rotate a point around origin by certain angle, the rotation can be around x axis, the rotation can be around y axis, the rotation can also be around z axis. So accordingly I can have 3 different rotation matrices for representing rotation around a particular axis. And specifically if the rotation has to be done about an arbitrary point then what we have to do is we have to translate the arbitrary point to the origin by using the translation transformation. After translating the points to the origin, then we have to perform rotation around the origin, then we have to translate back the point to its original position, so which gives us the desired rotation of any point p in 3 D around any arbitrary point q also in 3 D.

So now let us see that how this rotation

(Refer Slide Time 23:54)

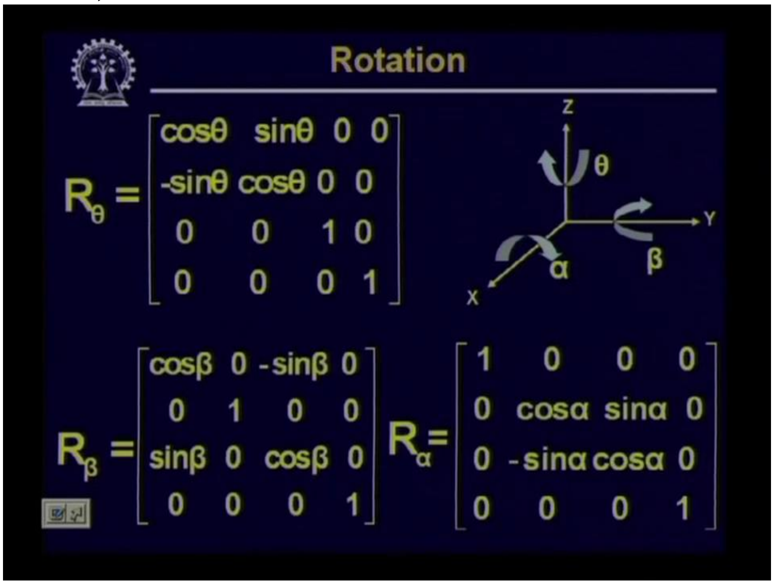


will look like in 3 D So here you will find that we have shown, on the right hand side this particular figure where this this figure shows the rotation of the point along x axis. So if the point is rotated along x axis, the rotation is given by, is indicated by alpha, if it is rotated along z axis, the rotation is indicated by theta and if the rotated, rotation is done along y axis, the rotation angle is indicated by beta. So if I rotate the point along z axis, so when I am rotating a point along z axis, then

(Refer Slide Time 24:46)

obviously the z coordinate of the point will remain unchanged even in the rotated position. But what will change is the x coordinate and y coordinate of the point in its new rotated position. And because the z coordinate is remaining unchanged so we can think that this is a rotation on a plane which is parallel to the x y plane. So the same transformation which we had done for rotating a point in two-dimension in the x y coordinate, the same transformation matrix holds true for rotating this point in three-dimension along the axis z. But now because the number of components in our position vector is more, so we have to take care of the other components as well. So using this you find that

(Refer Slide Time 25:37)

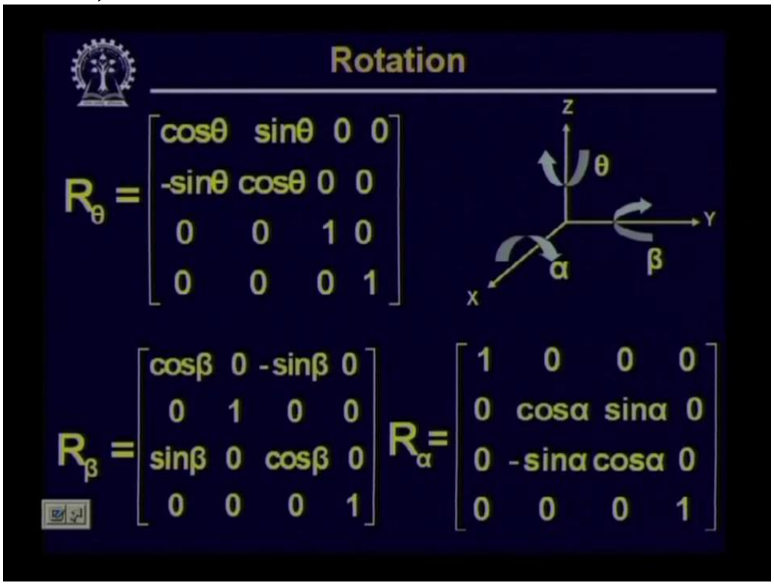


when I rotate the point around z axis, the rotation angle is given by theta and the rotation matrix is given by cosine theta sine theta 0 0 minus sine theta cosine theta 0 0 then 0 0 1 0 and 0 0 0 1. So this is the transformation matrix or rotation matrix for rotating

(Refer Slide Time 26:03)

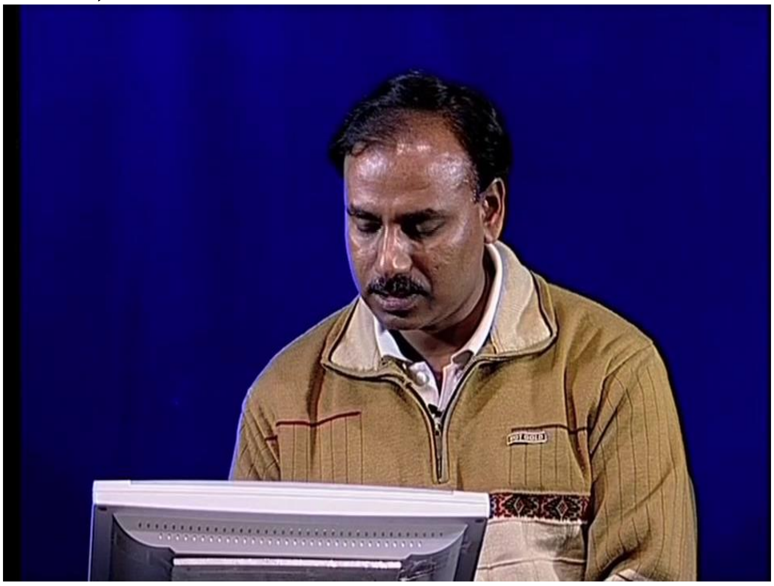
a point around z axis So here you find that the first few components that is the

(Refer Slide Time 26:11)



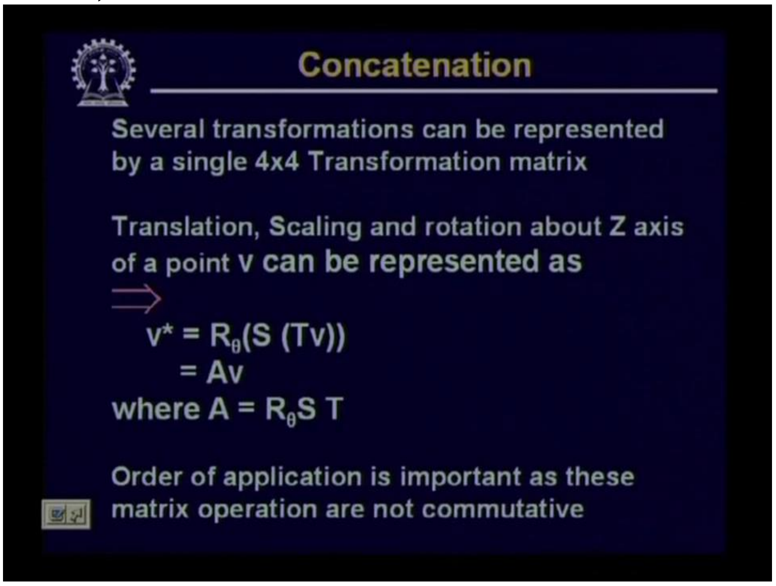
cosine theta sine theta then minus sine theta and cosine theta, so this 2 by 2 matrix is identical with the transformation matrix or rotation matrix that we have obtained in case of two-dimension. So this says, because the z th coordinate is remaining the same, so the x coordinate and y coordinate due to this rotation around z axis follows the same relation that we had derived in case of two dimension similarly when i translate the point around y axis where the rotate the point around y axis the angle of rotation is given by beta, so r beta as given in this particular case gives you the rotation matrix and if you rotate the angle along x axis where the rotation angle is given by alpha you will find that r alpha gives you the corresponding rotation matrix along the corresponding transformation matrix for rotation along the x axis. So as before, you find that when you rotate the point along around x axis the x coordinate will remain the same where as the y coordinate and the z coordinate of the point is going to differ, similarly when you rotate the point around y axis,

(Refer Slide Time 27:37)



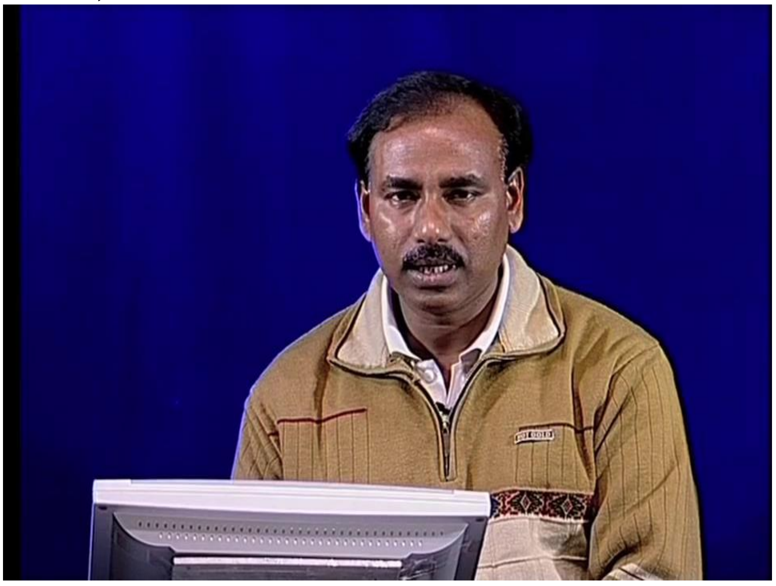
the y coordinate is going to remain the same but x coordinate and z coordinate they are going to be different. now as we also mentioned in case of two-dimension that

(Refer Slide Time 27:54)



different transformations can be concatenated. So here we have shown that how we can concatenate the different transformations. Here you find that all the transformations that you have considered in three-dimension all of them are in the unified form, that is, every transformation matrix is a 4 by 4 matrix and all the coordinates that we consider, we add 1 to the

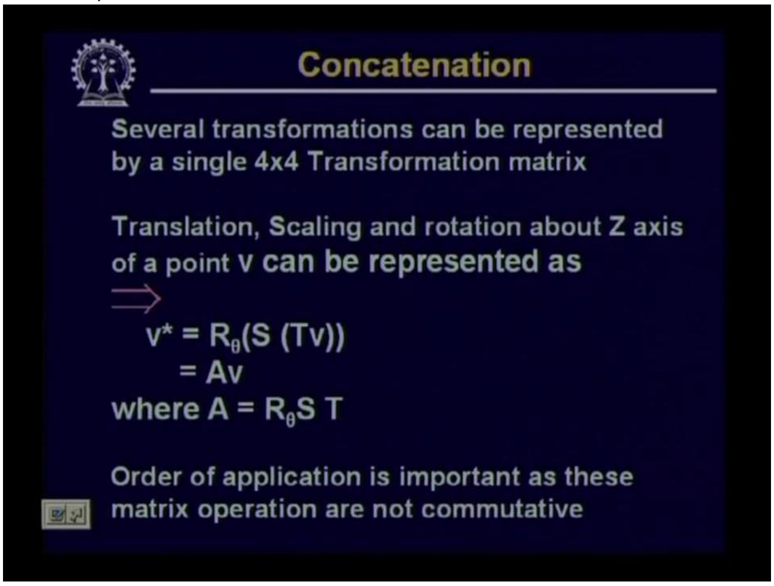
(Refer Slide Time 28:25)



coordinate x y z so that our position vector becomes a four-dimensional vector and the translated point is also a four-dimensional vector.

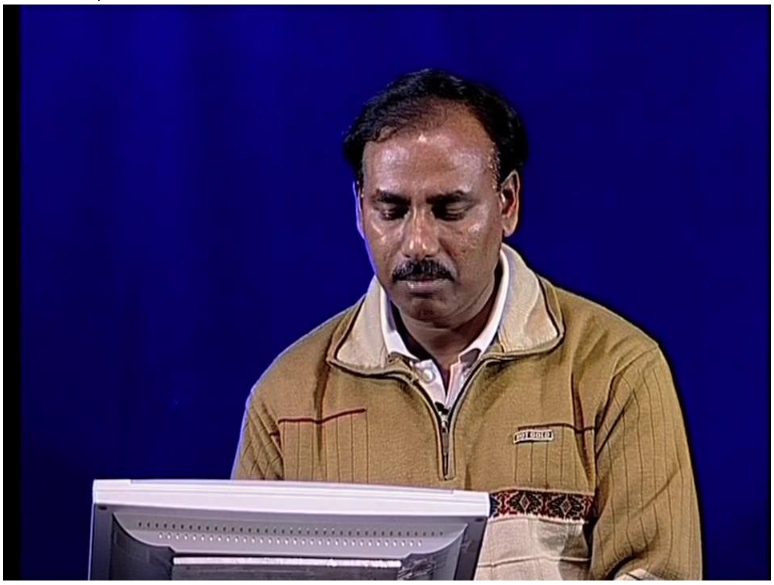
So if I want to concatenate these different transformations that is translation, scaling and rotation and if I want to trans rotate a point about z axis

(Refer Slide Time 28:48)



then this translation, scaling and rotation, these can be concatenated as, first you translate the point v by the translation operation t or the translation matrix t, then you perform scaling, then you perform rotation and this rotation is r theta and all these 3 different transformation matrices that is r theta, s and t all of them being four-dimensional matrices can be combined into a single matrix,

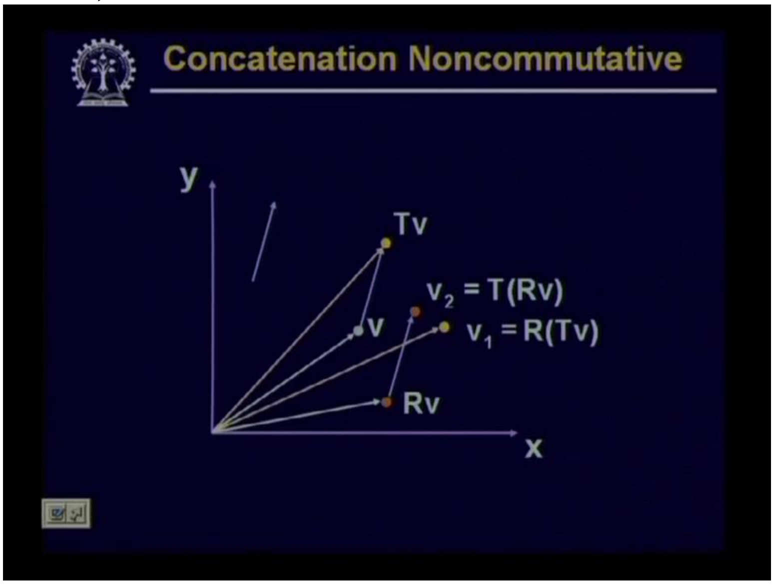
(Refer Slide Time 29:21)



say this single transformation matrix in this case a which is nothing but the product of r theta s and t and this A again will be a matrix of dimension 4 by 4. Now you know that whenever we are going for the concatenation of the transformations, the order in which these transformations are to be applied, that is very, very important. Because these matrix operations are in general, not commutative

Now just to illustrate

(Refer Slide Time 29:58)



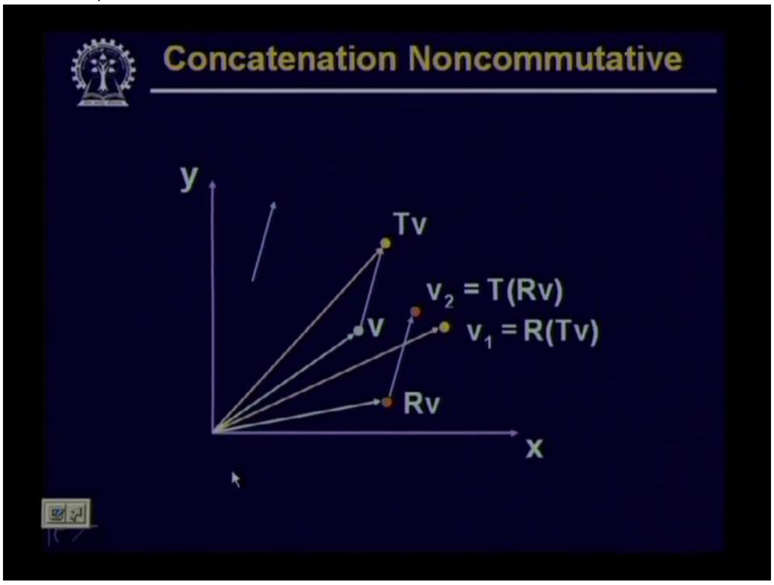
that these matrix operations are not commutative, let us take this example. Suppose I have this particular point v and to this point v I want to perform two kinds of operations. One is translation by vector so the translation vector, we have represented by this arrow by which this point has to be translated and the point v is also to be rotated by certain angle. Now there are two ways in which these two operations can be done. The first one shows that suppose I rotate point v first by using the operation r v so here the transformation matrix is a r for the rotation operation and after rotating this point v by using this transformation r, I translate the rotated point by using the transformation matrix t. So if I do that you find that this v is the original position of this point. If I first rotate it by using this rotation transformation, the point v comes here in the rotated position. And after this, if I give translation to this point v by this translation vector then the translated point is coming over here which is represented by v 2. So the point v 2 is obtained by v first by applying the rotation by transformation r followed by applying the translation by the transformation t.

Now if I do it reverse, that is first I translate the point v using this translation transformation t and after this translation I rotate this translated point which is now t v by the same angle theta. So what I do is, first I translate the point using the transformation t and after that this translated point is rotated using the rotation transformation r and this gives me the rotated point or the new point that is equal to v 1. Now from here you find that in the earlier case where I got the point v 2 and now I get the point v 1, this v 1 and v 2 they are not the same point.

(Refer Slide Time 32:51)

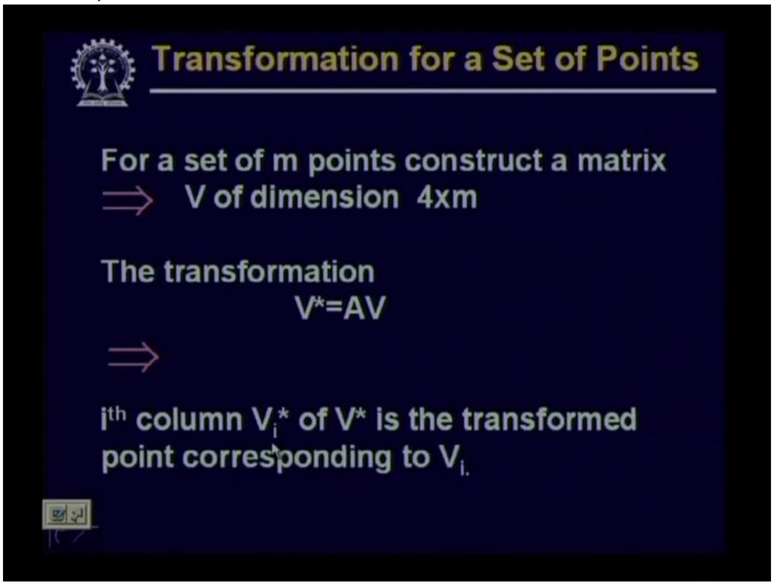
So this really illustrates that whenever we go for concatenation of different transformations we have to be very, very careful about the order in which these transformations are to be specified or the transformations are to be applied because if the order in which the transformations are applied vary, then we are not going to get the same end result. So for any such concatenation the order in which

(Refer Slide Time 33:22)



the concatenation is applied, the transformations has to be applied, that has to be thought of very, very carefully

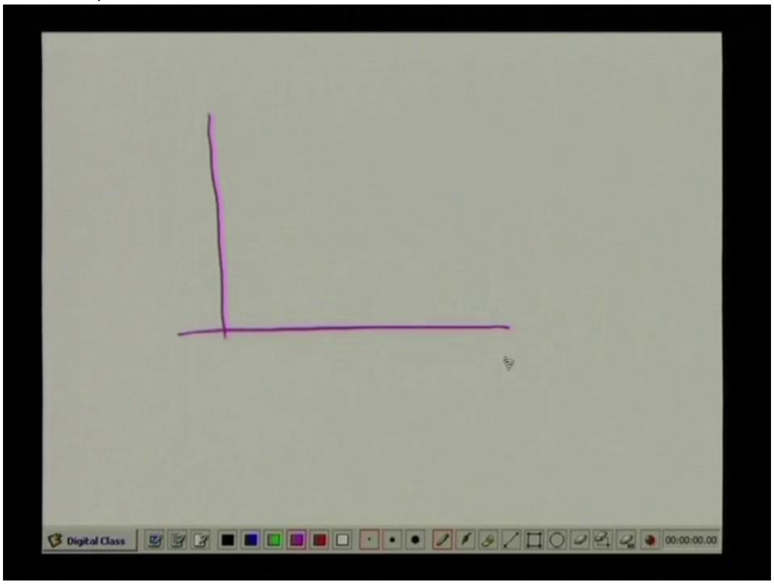
(Refer Slide Time 33:31)



So that is about translation of the transformation of a single point.

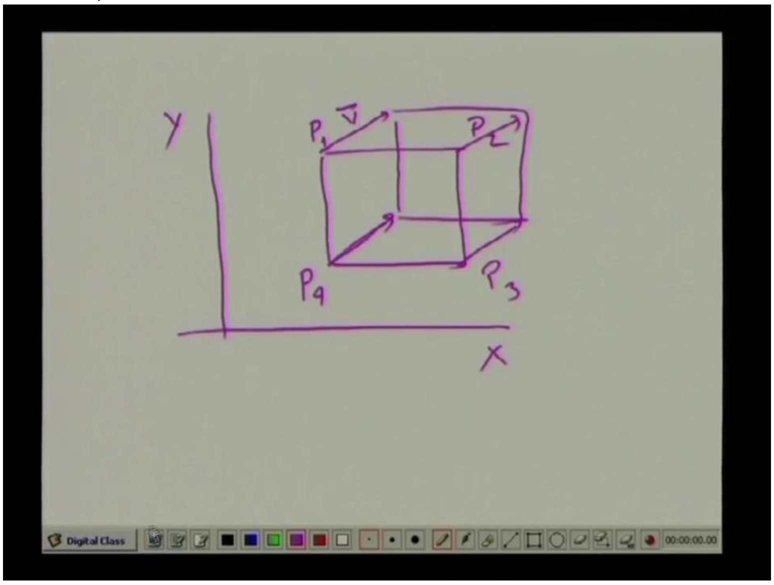
Now if I have to transform a set of points say for example, I can have a square figure which, say like this, say for example in a two-dimensional space

(Refer Slide Time 33:53)



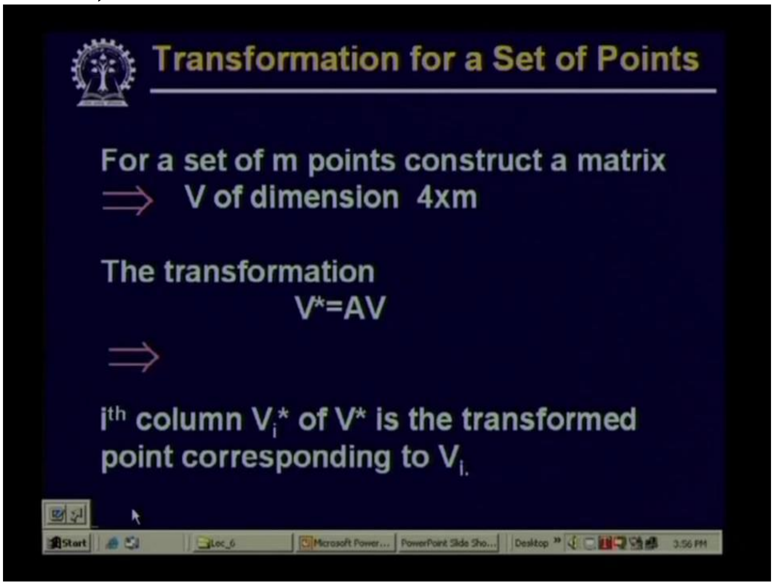
x y I have a square figure like this So this will have 4 vertices, the vertices I can represent as point p 1, point p 2, point p 3 and point p 4. Now so far the transformations that we have discussed, that is the transformation of a single point around origin or the transformation of a single point around another arbitrary point in the same space. Now here, if I have to transform the entire figure, that is for example, I want to rotate this entire figure about the origin or I want to translate this entire figure by certain vector say v. So for example I want to translate this entire figure to this particular position, so you find that here, all these points p 1, p 2, p 3 and p 4 all these points are going to be translated by the same displacement vector v. So it is also possible that we can apply transformation to

(Refer Slide Time 35:09)



all the points

(Refer Slide Time 35:10)



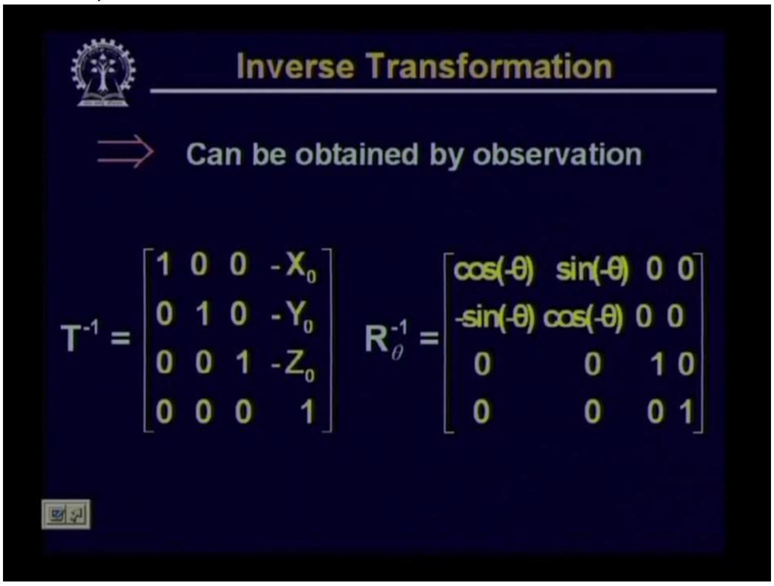
simultaneously rather than applying transformation to individual points one by one So for a set of n points what we have to do is, we have to construct a matrix v of dimension 4 by m, that is every individual point will now be considered of course in the unified form will now be considered as a column vector of a matrix which is of dimension 4 by m and then we have to apply the transformation a to this entire matrix and the transformation, after this transformation we get the new matrix v star which is given by the transformation A multiplied by the matrix v. So here we find that any particular column, the i th column in the matrix v star which is a v i star is the transformed point corresponding to the i th column of matrix v which is represented by v i. So if I have a set

(Refer Slide Time 36:17)

of points which are to be transformed by the same transformation then all those points can be added in the form of columns of a new matrix. So if I have n number m number of points I will have a matrix having m number of columns. The matrix will also obviously have 4 rows and this new 4 by m matrix that I get, this entire matrix has to be transformed using the same transformation operation and I get the transformed points again in the form of a matrix. And from that transformed matrix I can identify that which point which is the transformed point of the original point.

Now once we get these transformations,

(Refer Slide Time 37:08)

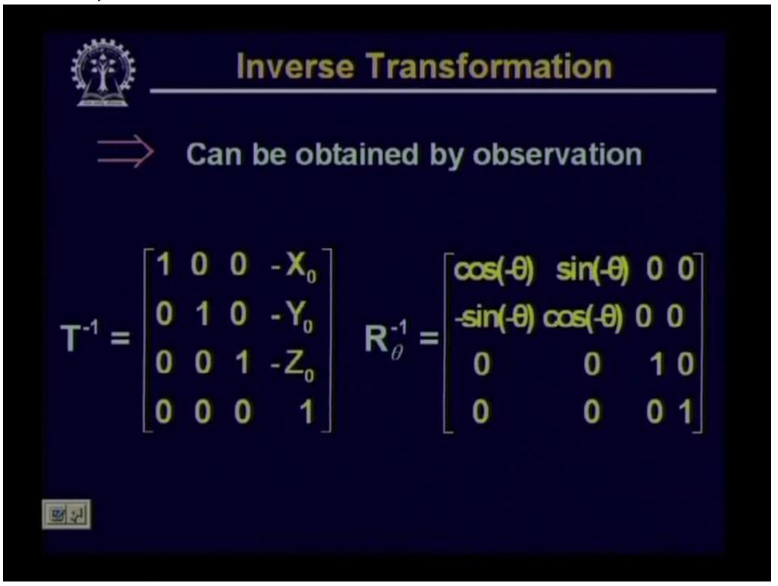


again we can get the corresponding inverse transformations. So the inverse transformations in most of the cases can be obtained just by observation. Say for example, if I apply,

(Refer Slide Time 37:23)

if I translate a point by a displacement vector v, then inverse transformation should bring back that translated point, that transformed point to its original position. So if my translation is by a vector v, the inverse transformation or the inverse translation should be by a vector minus v. So the inverse transformation matrix

(Refer Slide Time 37:47)

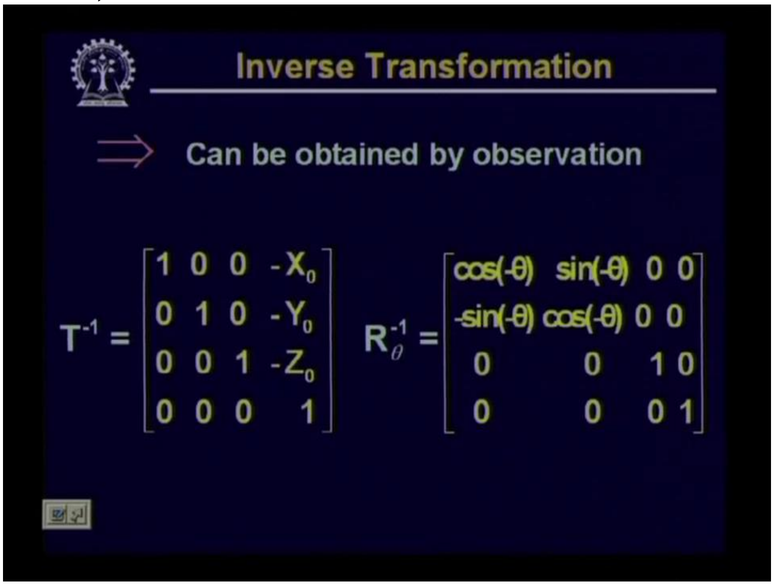


t inverse can be obtained as 1 0 0 minus x naught 0 1 0 minus y naught 0 0 1 minus z naught then 0 0 0 1. So you remember that the corresponding transformation matrix that we said was 1 0 0 x naught

(Refer Slide Time 38:07)

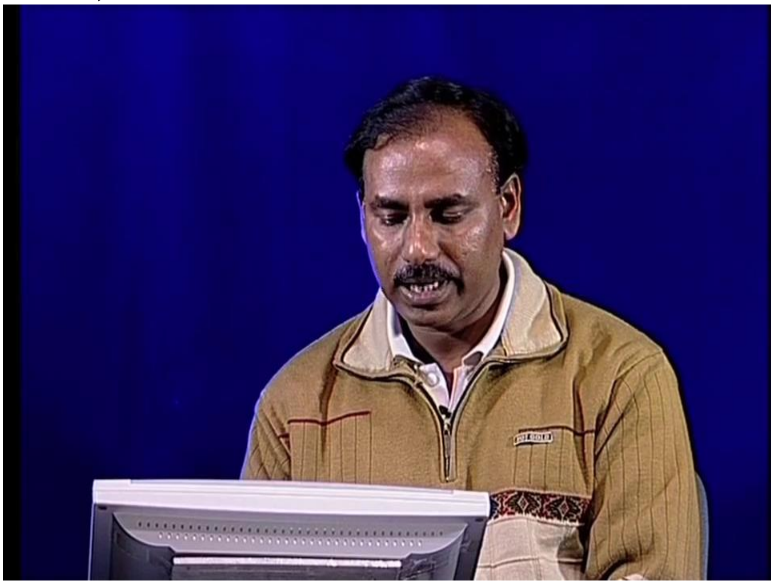
1 0 0, 0 1 0 y naught,

(Refer Slide Time 38:11)



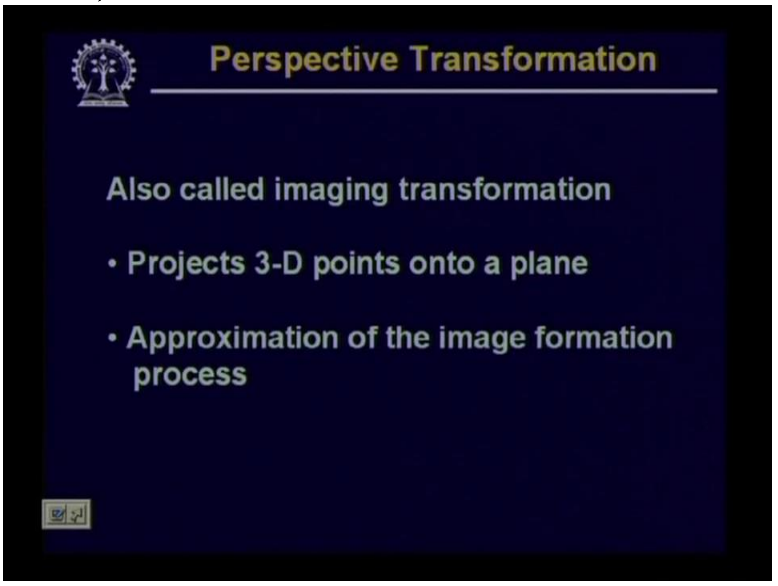
0 0 1 z naught and 0 0 0 1 So you will find that x naught y naught and z naught, they have just been negated to give you the inverse translation matrix t inverse. So similarly by the same observation we can get inverse rotations

(Refer Slide Time 38:30)



r theta inverse where what we have to do is in the transformation matrix original rotation matrix we had the term cosine theta sine theta minus sine theta cosine theta, now all these thetas are to be replaced by minus theta which gives me the inverse rotation matrix around the z axis. Similarly we can

(Refer Slide Time 38:54)



also find out the inverse matrix for scaling where the s x will be replaced by 1 upon s x. Thank you.