

Digital Image Processing

Prof. P.K.Biswas

Department of Electronic Electrical Communication

Indian Institute of Technology, Kharagpur

Lecture - 6

Basic Transformations

Hello, welcome to the video lectures series on digital image processing. In today's lecture we will discuss about some basic mathematical transformations and we will see that how these digital mathematical transformations help us in understanding the imaging model and image formation process by a camera.

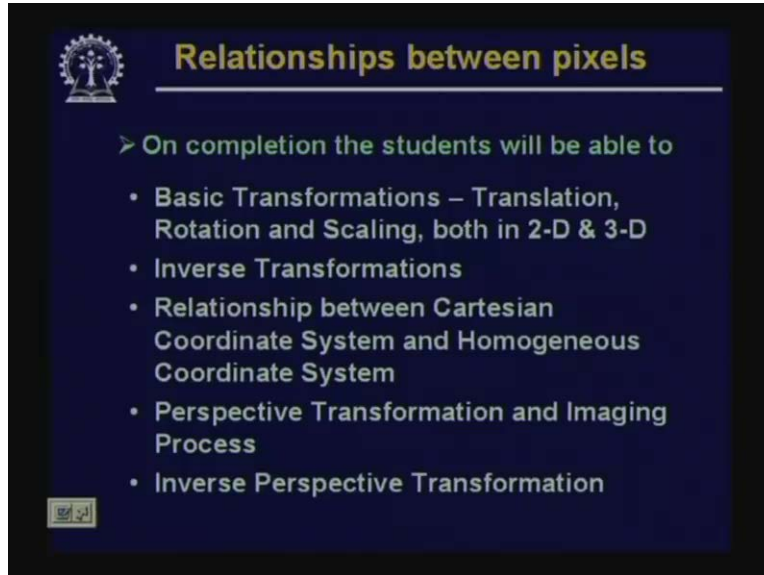
(Refer Slide Time: 1:29)



So, in our last lecture, we have learnt different distance measures, we have seen the application of the distance measure which we have said that in addition to finding out the distance between 2 different points, the distance measures are also used to find out the distance transformation of a binary image and using the distance transformation we can find out the skeleton of the image which gives a compact representation or compact description of the shape.

We have also seen different arithmetic and logical operations that can be performed on 2 images and we have also seen what are the different neighborhood operations that can be performed on a single image.

(Refer Slide Time: 2:19)



The slide features a dark blue background with a white logo in the top left corner. The title 'Relationships between pixels' is written in a bold, yellow font at the top. Below the title, a green arrow points to the text 'On completion the students will be able to'. A bulleted list follows, with each item in white text. A small white icon is located in the bottom left corner of the slide.

Relationships between pixels

➤ On completion the students will be able to

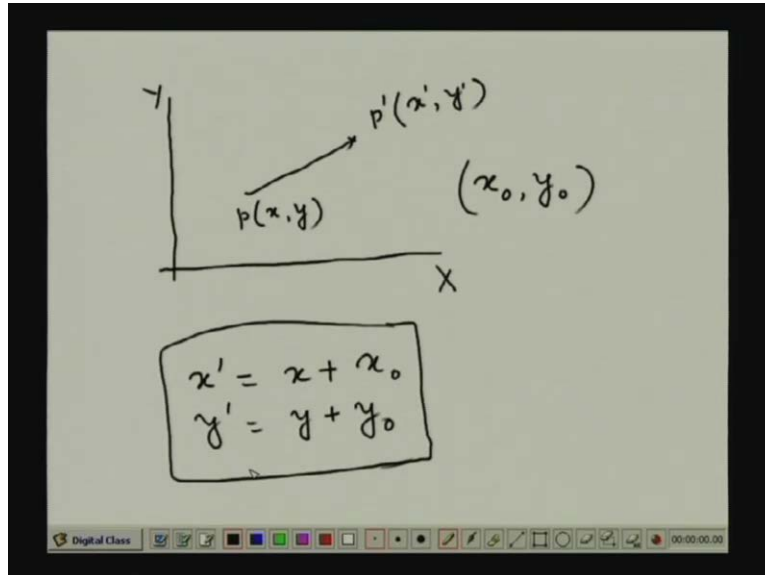
- Basic Transformations – Translation, Rotation and Scaling, both in 2-D & 3-D
- Inverse Transformations
- Relationship between Cartesian Coordinate System and Homogeneous Coordinate System
- Perspective Transformation and Imaging Process
- Inverse Perspective Transformation

So, in today's lecture 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 2 dimension as well as in 3 dimension.

We will also discuss about the inverse transformations of these different mathematical transformations. We will find out the relationship between Cartesian coordinate system and homogeneous coordinate system and we will see that this homogeneous coordinate system is very very useful while discussing about the image formations by a camera.

We will also talk about the perspective transformation and imaging process and then we will talk about the inverse perspective transformation.

(Refer Slide Time: 3:21)



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 2 dimensional. So, you know that if I have a 2 dimensional coordinate system given by the axis 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 $x_0 y_0$. So, after translating this point by the vector $x_0 y_0$, I get the translate 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 $x_0 y_0$; so you know that after translation, the new position x prime will be given by x plus x_0 and y prime will be given by y plus y_0 . **Now** so, this is the basic relation when a point at location xy is translated by a vector $x_0 y_0$. Now, let us see that how this can be represented more formally by using a matrix equation. So, if I translation this equation in the form of a matrix, the equation look like this.

(Refer Slide Time: 5:07)

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} x_0 \\ y_0 \end{bmatrix}$$
$$x' = x + x_0$$
$$y' = y + y_0$$

I have to find out the new location vector x prime y prime and we have said that this x prime is nothing but x plus x_0 and y prime is nothing but y plus y_0 . 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_0 and y prime is equal to y plus y_0 .

So, on the right hand side, you find that I have product of 2 matrices which is added to another column matrix or column vector.

(Refer Slide Time: 6:28)

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$
$$\Rightarrow \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

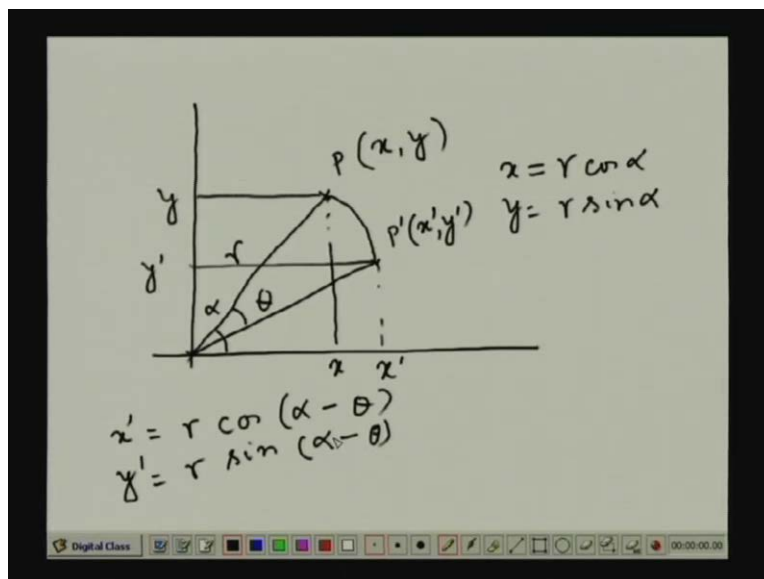
\Rightarrow Unified expression.

Now, if I want to combine all these operations in a single matrix form, then the operation will be something like this - on the left hand side I will have x' and y' which will be in the form of a matrix and on the right hand side I will have $1, 0, x_0, 0, 1, y_0$ and then I will have x, y and 1 . So, if I again do the same matrix computation, it will be x' equal to x plus 0 plus x_0 which is nothing but x plus x_0 . Similarly, y' will be 0 plus y plus y_0 which is nothing but y plus y_0 .

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', y' and I introduce one more component which I make equal to 1 , this is equal to $1, 0, x_0, 0, 1, y_0$ than $0, 0, 1$ and $x, y, 1$. So, we find that the second expression which I have just obtained from the first one is now a symmetric expression and this is what is called and unified expression.

So, we find that basically what I have is 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 $1, 0, x_0, 0, 1, y_0$ and $0, 0, 1$; then I get the translated point as $x', y', 1$ where if I just neglect the additional component which in this case is 1 , then I get the translated point P' . So, this is about the translation.

(Refer Slide Time: 9:14)

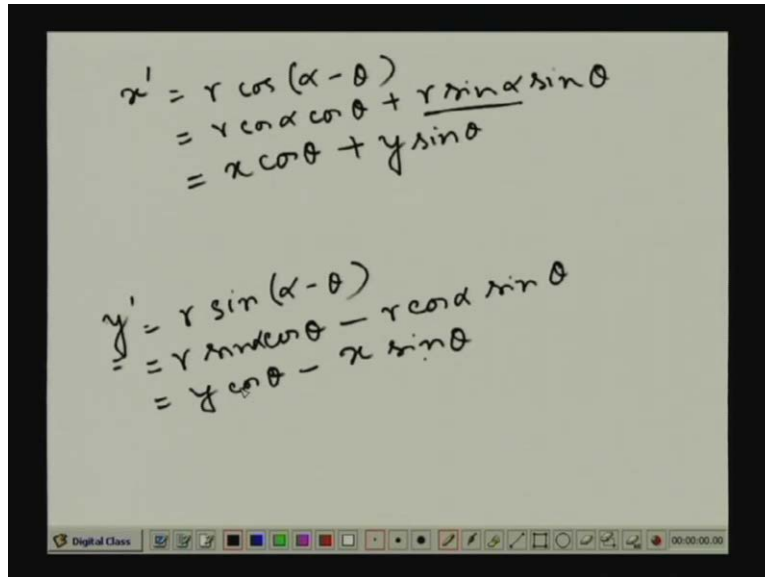


In the same manner, given a point P again in 2D; so again I have this point P which is **having** again having a coordinate (x, y) and suppose I want to rotate this point P around the origin by an angle θ . Now, one way of representing this point p is if r is the distance of point P from the origin; then these are the coordinates of the point p . This is the x coordinate, this is the y coordinate, I **can also represent** and suppose this angle is α , then I can also represent x as x equal to r cosine α and y equal to r sine α .

Now suppose, I want to rotate this point P by an angle θ in the clockwise direction; so, the new position of P will now be P' having the coordinate location x' and y' and this

rotation, angle of the 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 sin alpha minus theta.

(Refer Slide Time: 11:07)



The image shows a digital whiteboard with handwritten mathematical derivations. The top part shows the derivation for x' :

$$\begin{aligned}x' &= r \cos(\alpha - \theta) \\ &= r \cos \alpha \cos \theta + r \sin \alpha \sin \theta \\ &= x \cos \theta + y \sin \theta\end{aligned}$$

The bottom part shows the derivation for y' :

$$\begin{aligned}y' &= r \sin(\alpha - \theta) \\ &= r \sin \alpha \cos \theta - r \cos \alpha \sin \theta \\ &= y \cos \theta - x \sin \theta\end{aligned}$$

At the bottom of the whiteboard, there is a toolbar with various drawing tools and a timer showing 00:00:00:00.

So if I simply expand this, so I have x prime is equal to r cosine alpha minus theta and y prime is equal to r sin alpha minus theta. So, if I simply expand this cosine term, it will simply be r cosine alpha cosine theta plus r sin alpha sin theta. Now, we know that r cosine alpha is nothing but x, so it becomes x cosine theta plus r sin alpha is nothing but y, so it becomes y sin theta.

Similarly, in this case if I expand this, it becomes r sin alpha cosine theta minus r cosine alpha sin theta. So again, r sin alpha this is nothing but y, so this expression y cosine theta minus x sin theta. So again, so here we find that x prime given by x cosine theta plus y sin theta and y prime is given by minus x sin theta plus y cosine theta.

(Refer Slide Time: 12:38)

The image shows two handwritten equations on a digital whiteboard. The first equation is a 2D rotation matrix:
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$
 A bracket is drawn under the second row of the matrix. The second equation is a scaling matrix:
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$
 The whiteboard interface includes a toolbar at the bottom with various drawing tools and a timer showing 00:00:00:00.

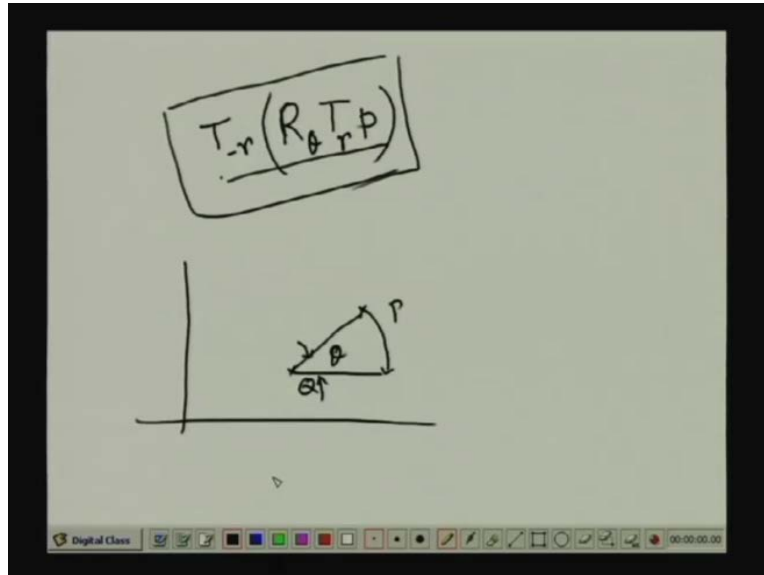
So, even now if I represent this in the form of an matrix equation, it becomes x' y' is equal to cosine theta sin theta, then minus sin theta cosine theta and then I have the original coordinates x and y . So, here you find that if I rotate the point p by an angle θ around the origin in the clockwise direction; in that case, the transformation matrix which gives the rotation transformation is given by this particular matrix which is cosine theta sin theta minus sin theta cosine theta.

Now, in the same manner if I go for scaling, say for example if I have scaling S_x , scaling vector of S_x in the x direction and I have scaling vector S_y in the y direction; in that case, the transformation matrix for scaling can also be represented as x' y' 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 2 dimensions. 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 the arbitrary point q in the 2 dimensions, then find out the expression for this rotation of point p by an angle θ 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 this transformation, the translation, if I now rotate point p by the same angle θ and now it will be rotation around the origin. So, whatever expression that we have found here that is cosine theta sin theta minus sin theta cosine theta, the same transformation matrix is applicable and after getting this rotation, now you translate back the rotated point by the same vector but in the opposite direction.

(Refer Slide Time: 15:38)



So, here the transformation 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 the transformation say R theta. So, first we are translating by a vector say r, then we are performing the 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 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 point Q by angle theta.

So, these are the different transformations, the basic mathematical transformations that we can do in 2 dimensional space. Now, let us see that what will be corresponding transformations if I move from 2 dimensional space to the 3 dimensional space.

(Refer Slide Time: 17:19)

Some Basic Transformation

Here we consider some basic image Transformation

- Translation
- Rotation
- Scaling

Coordinate system \Rightarrow 3-D cartesian coordinate (x,y,z)

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

(Refer Slide Time: 17:36)

Translation

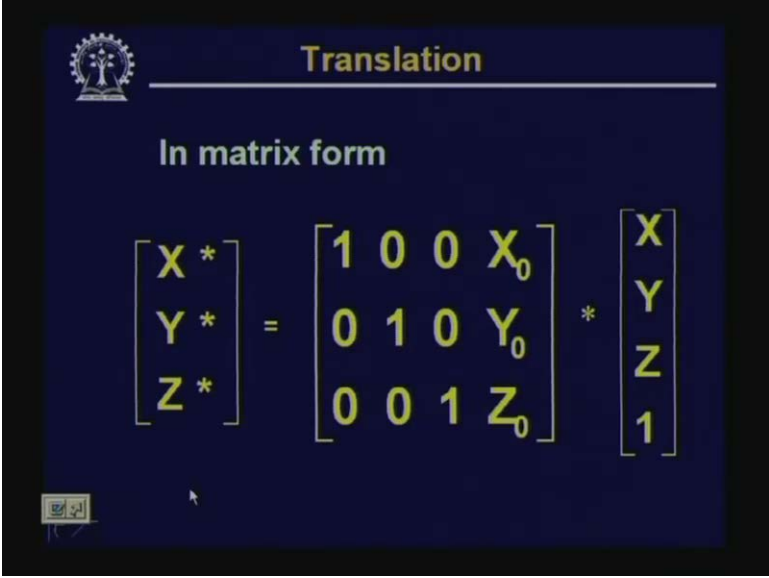
A point (x,y,z) is translated to a new Coordinate (x^*,y^*,z^*) using a displacement vector (x_0,y_0,z_0)

$$\Rightarrow \begin{aligned} x^* &= x + x_0 \\ y^* &= y + y_0 \\ z^* &= z + z_0 \end{aligned}$$

So, first let us see as we have seen in case of 2 dimension that if a point (x, y, z) is translated to new coordinate say $(x \text{ star}, y \text{ star}, z \text{ star})$ using a displacement vector $x_0 \ y_0 \ z_0$; then this translated coordinates $x \text{ star}$ will be given by x plus x_0 $y, \text{ star}$ will be given by y plus y_0 and $z \text{ star}$ will given by z plus z_0 .

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 z₀ that was absent. But now we are considering a 3 dimensional space a 3D coordinate system, so we have 3 coordinates x, y and z and all these 3D coordinates, all the 3 coordinate are to be translated by the translation vector x₀ y₀ z₀ and then new translation at the new point we get as x star, y star and z star.

(Refer Slide Time: 18:56)



Translation

In matrix form

$$\begin{bmatrix} X^* \\ Y^* \\ Z^* \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & X_0 \\ 0 & 1 & 0 & Y_0 \\ 0 & 0 & 1 & Z_0 \end{bmatrix} * \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Now, if I write these 3 equation in the form of a matrix, then the matrix equation will be like this – (X star, Y star, Z star) on the left hand side will be equal to (1 0 0 x₀) (0 1 0 y₀) (0 0 1 Z₀) into the column vector (X Y Z 1). So, this is the similar situation that we have also seen in case of 2 dementional that we have to add an additional component which is equal to 1 in our original ah position vector XYZ .

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 with the translation matrix given by (1 0 0 X₀) (0 1 0 Y₀) and (0 0 1 Z₀). 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 representations, we will represent this matrix the dimension of 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: 20:26)

Translation

Unified representation

$$\Rightarrow \begin{bmatrix} X^* \\ Y^* \\ Z^* \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & X_0 \\ 0 & 1 & 0 & Y_0 \\ 0 & 0 & 1 & Z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

So, the unified representation as we have already said is given by (X star Y Star Z star 1) is equal to the translation matrix $\begin{pmatrix} 1 & 0 & 0 & X_0 \\ 0 & 1 & 0 & Y_0 \\ 0 & 0 & 1 & Z_0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$ multiplied by the column vector $\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$.

So, this particular matrix that is $\begin{pmatrix} 1 & 0 & 0 & X_0 \\ 0 & 1 & 0 & Y_0 \\ 0 & 0 & 1 & Z_0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$ and $\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$ this represents a translation a transformation matrix used for the translation and we will represent this matrix by this uppercase letter T. So, that is about the simple translation that we can have.

(Refer Slide Time: 21:26)

Translation

Unified matrix representation is of the form
 $v^* = Av$
 $A \Rightarrow 4 \times 4$ transformation matrix

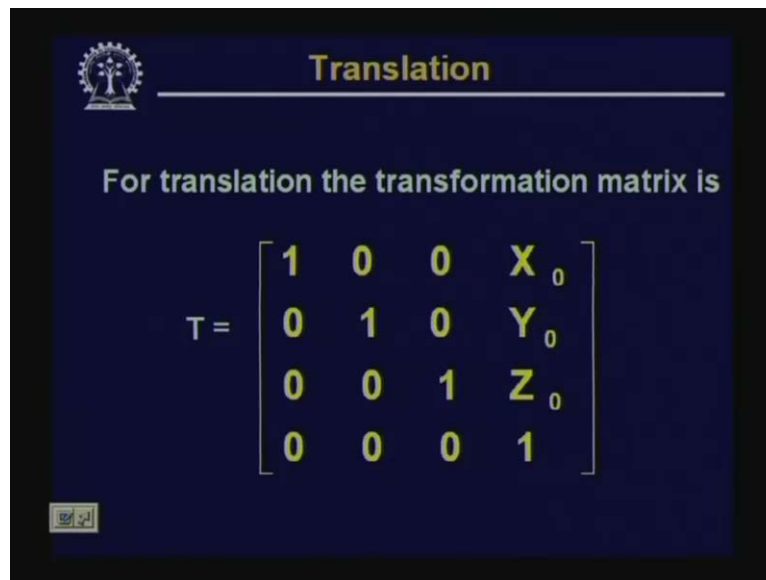
$v = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \Rightarrow$ column vector of original coordinates

$v^* = \begin{bmatrix} X^* \\ Y^* \\ Z^* \\ 1 \end{bmatrix} \Rightarrow$ column vector of transformed coordinates

So, in unified matrix representation, we have done that 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 4 dimensional vector having components X, Y, Z and 1.

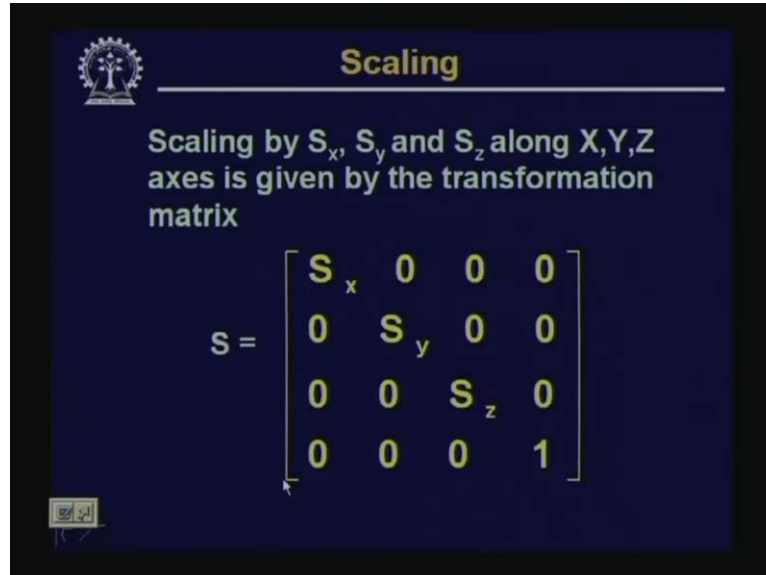
Similarly, the transformed position vector V^* is also a 4 dimensional vector which is having components X^*, Y^*, Z^* and 1. So, this is how in the unified matrix representation, we can represent the translation of a position vector or a translation of a point in 3 dimension.

(Refer Slide Time: 22:28)



Similarly, **we can have so** as I said that this is the transformation matrix which is represented, which is used for translating a point in point 3D by vector $X_1 Y X_0 Y_0 Z_0$ or the displacement vector $X_0 Y_0 Z_0$.

(Refer Slide Time: 22:49)

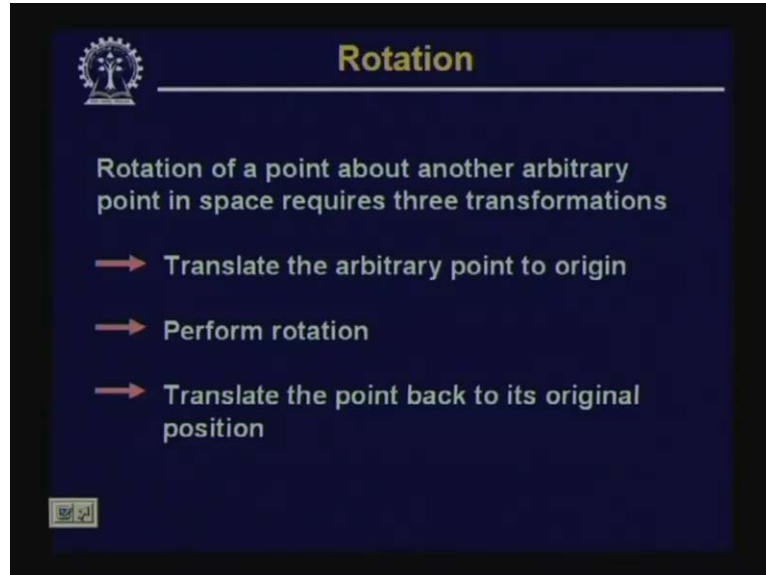


Similarly, as we have seen in case of scaling in 2 dimension that if we have the scaling factor of S_x , S_y and S_z along the directions x, y and z; so along direction x, we have the scaling factor S_x , along the direction y, we have the scaling factor S_y and along the direction z, we have the scaling factor S_z , then the transformation matrix for this scaling operation can be written by S equal to 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 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 scale form and there if we remove the last component that is equal to 1, what we get is the scaled 3D 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 is greater than 1 or they are less than 1.

(Refer Slide Time: 24:22)

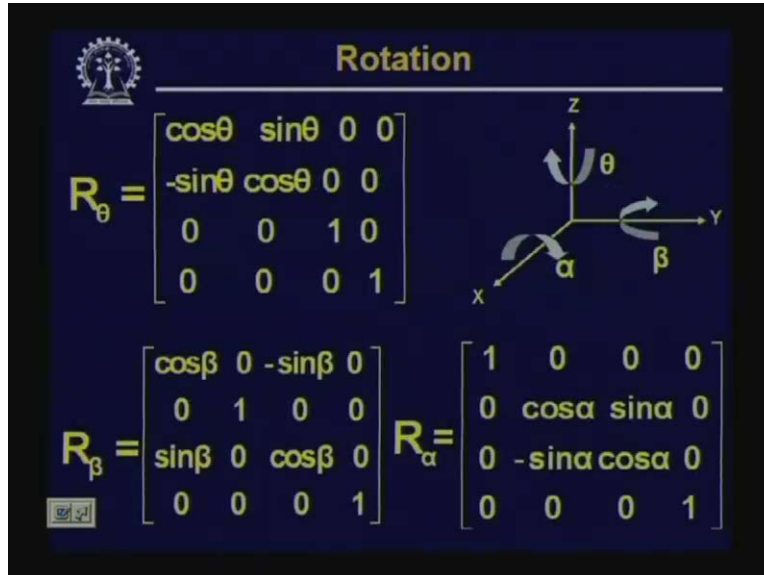


Then coming to rotation, we have seen that the translation and scaling in 3 dimensions is very simple, it is as simple as we have done in case of 2 dimensions. But the rotation in 3 dimension is a bit complicated because in 3 dimension, as we have 3 different axis - x axis, y axis and z axis; so, when I rotate a point around origin by certain angle, the rotation can be around x axis, the rotation can y axis, the rotation can be also be around z axis.

So accordingly, I can have 3 different rotation matrixes for representing rotation around a particular axis and specifically, if the rotation as to be done about an arbitrary point; then what we have to do is we had to translate the arbitrary point to the origins by using the **translation** transformation.

After translating the point 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 3D around any other arbitrary point Q also in 3D.

(Refer Slide Time: 25:46)



So, now let us see that how this rotation will look like in 3D. So, here you find that we have shown on the right hand side, this particular figure where 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 alpha. If it is rotated along z axis, the rotation is indicated by theta and if the 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 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 coordinates is remaining unchanged, so we can think that this is a rotation on a plane which is parallel to the xy pair.

So, the same transformation which we have done for rotating a point in 2 dimensions in the (x, y) coordinate, the same transformation matrix holds true for rotating this point in 3 dimension along the axis z. But know because the number of components in our position vector is more, we have to take care of the other components as well.

So using this, you find that when I rotate the point around z axis, the rotation angle is given by theta and the rotation matrix is given by cosine theta sin theta 0 0, minus sin 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 a point around z axis.

So, here you find that the first few components that is cosine theta sin theta, then minus sin 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 2 dimensions. So, this says that because the z coordinate is remaining the same, so the x coordinate and the y coordinate due to this rotation around z axis follows the same relation that we have derived in the case of 2 dimensions.

Similarly, when I translate the point around y axis where the rotate the point around y axis, there 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 around the x axis where the rotation angle is given by alpha, you find that R alpha gives you the corresponding rotation matrix along the corresponding transformation matrix for rotation along the x axis.

So as before, here you find that when you rotate the point along around the x axis, the x coordinate will remain the same. Whereas, the y coordinate and the z coordinate of the point is going to differ. Similarly, when you rotate the point around y axis, the y coordinate is going to remain the same but x coordinate and z coordinate, they are going to be different.

(Refer Slide Time: 29:45)

Concatenation

Several transformations can be represented by a single 4x4 Transformation matrix

Translation, Scaling and rotation about Z axis of a point V can be represented as

⇒

$$v^* = R_{\theta}(S(Tv))$$

$$= Av$$

where $A = R_{\theta} S T$

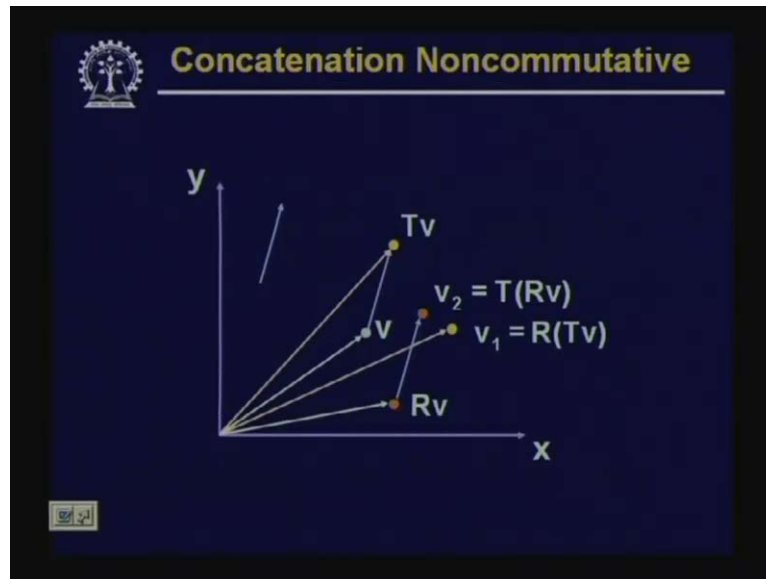
Order of application is important as these matrix operation are not commutative

Now, as we have also mentioned in case of 2 dimensions that different transformations can be concatenate. So, here we have shown that say how we can concatenate the different transformations. Here, you find that all the transformations that we have considered in 3 dimensions, 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 coordinate X Y Z so that our position vector become a 4 dimensional vector and the translated point is also a 4 dimensional vector.

So, if I want to concatenate this different transformation that is translation, scaling and rotation and if I want to rotate a point about Z axis; then this translations, scaling and rotation, this can be concatenated as first you translate the point V by the translation operation T of the translation matrix T , then you perform scaling, then you perform rotation and this rotation is R theta and all these 3 different transformation matrix that is R theta, S and T all of them being 4 dimension matrix can be combined into a single matrix say the 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 note that whenever we are going for concatenation of 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.

(Refer Slide Time: 31:51)



Now, just to illustrate that this 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 2 kind of operation. One is translation by vector, so the translations 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 2 ways in which these 2 operations can be done. The first one shows that suppose I rotate point V first by using the operation Rv . 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 .

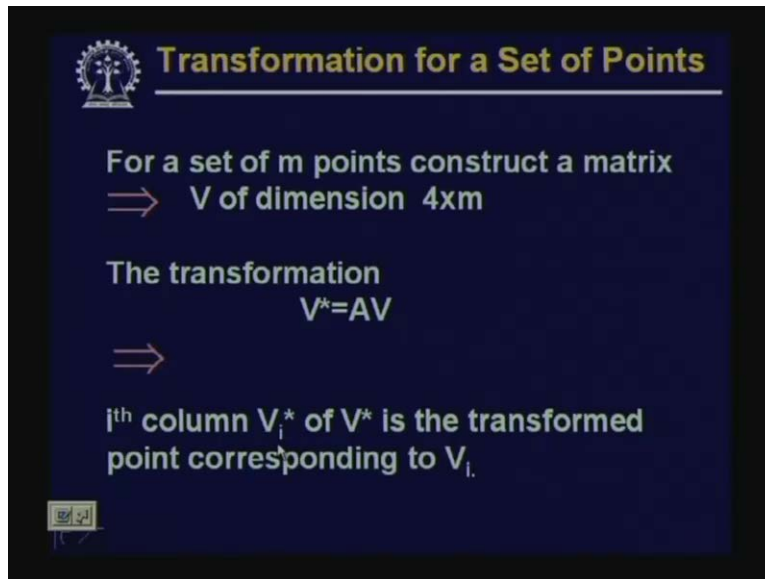
So, if I do that, you find that this V is the original position of this point. If I first rotate it by using this rotation transformation, the point V comes here in the rotated position and after this if I give the translation to this point V by this translation vector, then the translated point is coming over here which is represented by V_2 .

So, the point V_2 is obtained from 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 now Tv by the same angle θ ; so what I do is first I translate the point using the transformation T and after that this translated point is rotated by 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 in 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. So, this clearly illustrates that whenever

we go for concatenation of different transformations, we have to be very very careful about the order in which this transformations are to be specified or to 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 the concatenation is applied the transformations are applied that has to be thought of very very careful.

(Refer Slide Time: 35:23)



Transformation for a Set of Points

For a set of m points construct a matrix
 $\Rightarrow V$ of dimension $4 \times m$

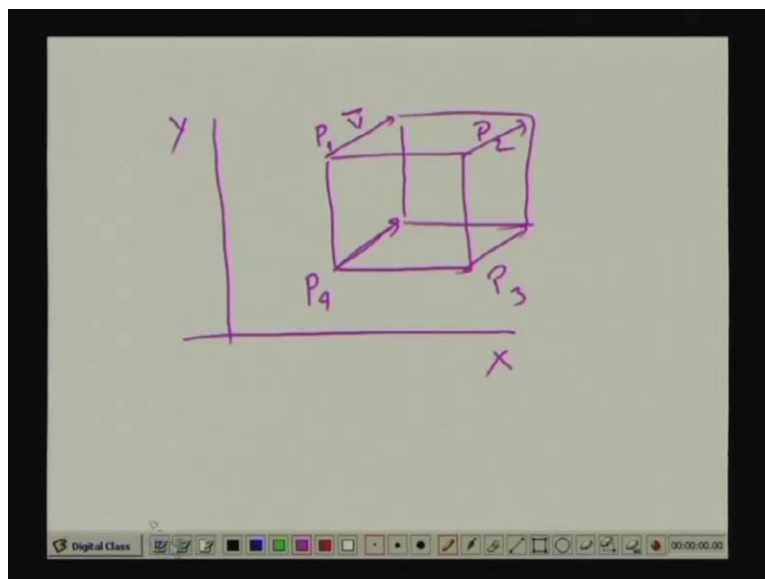
The transformation
 $V^* = AV$

\Rightarrow

i^{th} column V_i^* of V^* is the transformed point corresponding to V_i .

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 it is like this, say for example, in a 2 dimensional space $X Y$, I have the square figure like this.

(Refer Slide Time: 35:42)



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 this 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 say for example I want to translate the 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 all the points simultaneously, rather than applying transformation to individual points 1 by 1. So, for a set of m points, what we have to do is we have to construct a matrix V of a 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 this matrix which is of dimension of 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^* which is given by the transformation A multiplied by the B .

So, you have to find that any particular column, i 'th column in the matrix in the V_i V^* which is a V_i^* is the transformed point corresponding to the i 'th column of matrix B which is represented V_i . So, if I have a set of points which are to be transformed by the same transformation, then all those points can be arranged in the form of columns of a new matrix. So, if I have n 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 transform matrix, I can identify that **which point** which is the transformed point of the original points.

(Refer Slide Time: 38:59)

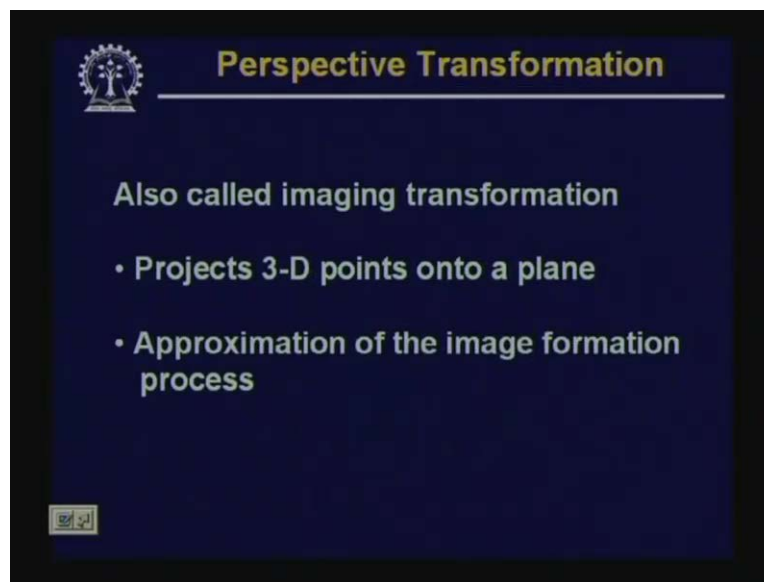
The slide is titled "Inverse Transformation" and includes a logo in the top left corner. Below the title, it states "Can be obtained by observation" with a right-pointing arrow. The slide displays two inverse transformation matrices:

$$T^{-1} = \begin{bmatrix} 1 & 0 & 0 & -X_0 \\ 0 & 1 & 0 & -Y_0 \\ 0 & 0 & 1 & -Z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad R_{\theta}^{-1} = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 & 0 \\ -\sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Now, once we get these transformations, again we can get the corresponding inverse transformation. So, the inverse transformations, in most of the cases can be obtained just by observations. Say for example, **if I apply** 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 T inverse can be obtained as $(1 \ 0 \ 0 \ \text{minus } X_0)$ $(0 \ 1 \ 0 \ \text{minus } Y_0)$ $(0 \ 0 \ 1 \ \text{minus } Z_0)$ then $(0 \ 0 \ 0 \ 1)$.

So, you remember that the corresponding transformation matrix that we said was $(1 \ 0 \ 0 \ x_0)$ **$(1 \ 0 \ 0 \ 0)$** $(0 \ 1 \ 0 \ Y_0)$ $(0 \ 0 \ 1 \ Z_0)$ and $(0 \ 0 \ 0 \ 1)$. So, we find that X_0 Y_0 and Z_0 , 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 R theta inverse where what we have to do is in the transformation matrix original rotation matrix, we have the term cosine theta sin theta minus sin theta cosine theta; now, all this thetas are to be replaced by minus theta which gives me the inverse rotation matrix around the Z axis.

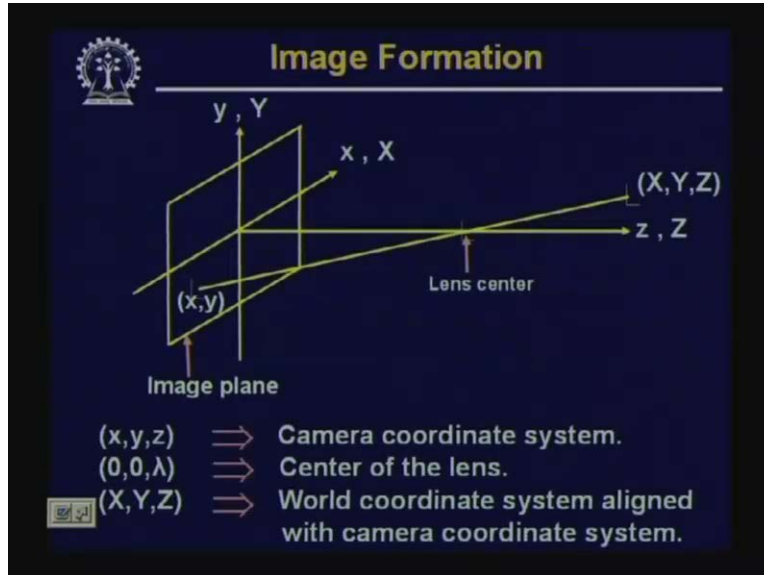
(Refer Slide Time: 40:45)



Similarly, we can also find out the inverse matrix for scaling where the S_x will be replaced by 1 upon S_x . So, these are the basic transformation, basic mathematical transformations. Now, we will see another form of transformation which is called a perspective transformation. Now, this perspective transformation is very very important to understand how a point in the 3 dimension in the 3D world is imaged by a camera.

So, this perspective transformation is also known as an imaging transformation and the purpose of this imaging transformation is to project a 3D point, a 3D world point into the image plane and this gives an approximation to the image formation process which is actually followed by a camera. Now, let us see what is this perspective transformation.

(Refer Slide Time: 41:49)



Here, we have shown a figure which is an approximation of the **image perform** image formation process. Here, you find that we have 2 coordinate systems which as a superimposed one over the other. One is the 3D world coordinate system represented by capital X, capital Y, capital Z. So, this is the 3D world coordinate system capital X capital, Y and capital Z and I also have the camera coordinate system which is given by lowercase x, lowercase y and lowercase z.

Now, here we have assumed that this camera coordinate system and the 3D world coordinate system, the perfectly aligned. That is x axis of the 3D world coordinate systems coincides with the x axis of the camera coordinate system, y axis of the world coordinate system coincides with the y axis of the camera coordinate system. Similarly, Z axis of the world coordinate system coincides the Z axis of the camera coordinate system. So, they have, both this coordinate systems have the same origin.

Now, if I have a point X, Y, Z in 3D; so this is the point XYZ in 3 dimension and I have assume that the center of the lens is at location 0 0 lambda; so obviously, the lambda which is the Z coordinate of the lens center, this also nothing but the focal length of the camera and this X, Y, Z this particular 3D point, we are assume that it is mapped to the camera coordinate given by lowercase x and lowercase y.

Now, our purpose is that if I know this 3D coordinate system - capital X, capital Y, capital Z and I know the value of lambda that is the focal length of the camera whether it is possible to find out the coordinate, the image coordinate corresponding to this 3D world coordinate X, Y, Z.

(Refer Slide Time: 44:31)

Image Formation

$(X, Y, Z) \Rightarrow$ World coordinate of any point in 3-D scene

We are interested in $(x, y) \Rightarrow$ projection of (X, Y, Z) on the image plane.

By using similar triangles

$$x/\lambda = -X/(Z - \lambda) = X/(\lambda - Z)$$

and $y/\lambda = Y/(Z - \lambda) = Y/(\lambda - Z)$

$\Rightarrow \left. \begin{array}{l} x = \lambda X / (\lambda - Z) \\ y = \lambda Y / (\lambda - Z) \end{array} \right\} \rightarrow$ Can be expressed as matrix expression in Homogeneous Coordinate

So to this, we apply the concept of similar triangles. So here, what we do is by using the similar triangles we can find out an the expression that lowercase x by lambda is equal to minus capital X by capital Z minus lambda which is nothing but capital X by lambda minus Z and y by lambda is in the same manner is given by capital Y by lambda minus capital Z.

So from this, I can find out that the image coordinates of the 3D world coordinate capital X capital Y, capital Z is given by x coordinate x inverse coordinate x is given by lambda x by lambda minus capital Z. Similarly, the inverse coordinate y is given by lambda capital Y divided by lambda minus capital Z. Now, these expressions also can be represented in the form of the matrix and here we find that if I go for homogeneous coordinate system, then this matrix expression is even simpler.

(Refer Slide Time: 45:44)

The slide features a dark blue background with a university logo in the top left corner. The title 'Homogeneous coordinates' is centered at the top in a yellow font. Below the title, two columns are defined: 'Cartesian coordinate' and 'Homogeneous coordinate'. An arrow points from the Cartesian coordinate (X, Y, Z) to the homogeneous coordinate (kX, kY, kZ, k) . A note states that k is an arbitrary nonzero constant. The text 'Homogeneous to Cartesian coordinate conversion is simple' is followed by the vector equation $w = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \Rightarrow w_h = \begin{bmatrix} kX \\ kY \\ kZ \\ k \end{bmatrix}$. The phrase 'In vector form' is followed by an arrow pointing to the vector equation.

So, let us see what is this homogeneous coordinate system. Homogeneous coordinate system is if I have the Cartesian coordinate capital X, capital Y capital Z; then we have said that in unified coordinate system, we just append a value 1 as an additional component. Homogeneous coordinate system is instead of simply adding 1, we add an arbitrary non 0 constant say k and multiply all the coordinated X, Y and Z by the same value k.

So, given the Cartesian coordinate, capital X capital Y capital Z; I can convert this to homogenous coordinate by k times capital X, k times capital Y and k times capital Z. The inverse process is also very simple that if I have a homogenous coordinate, then what have to do is I have to divide all the components of the homogenous coordinate by the fourth term.

So, in this case the fourth term is k and all the other terms where kX, kY and kZ; so if I divided all this 3 terms by the fourth component k, I get the Cartesian coordinate X Y Z. So, I can convert a 3D point, the coordinates from the Cartesian coordinate system to the homogeneous coordinate system and I can also very easily convert **from homogenous coordinate** from homogenous coordinate system to Cartesian coordinate system.

(Refer Slide Time: 47:24)

Image Formation

Define a perspective transformation matrix $P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1/\lambda & 1 \end{bmatrix}$

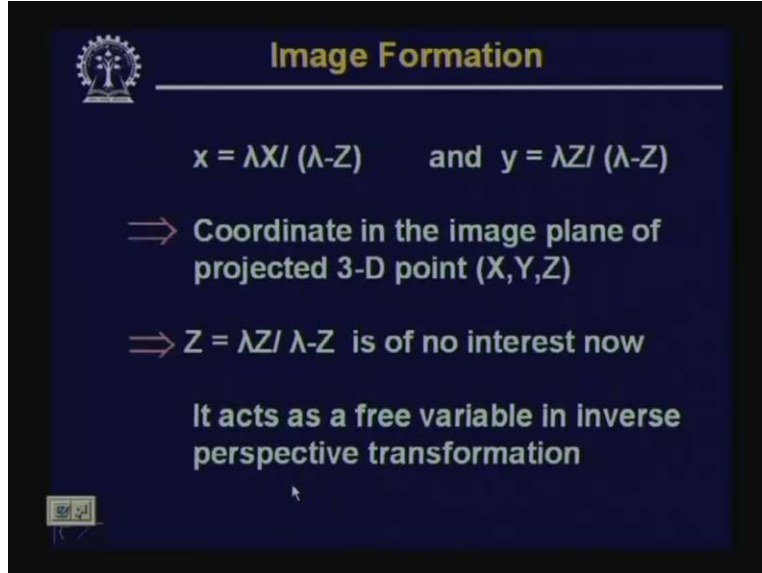
$$C_h = P W_h = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1/\lambda & 1 \end{bmatrix} * \begin{bmatrix} kX \\ kY \\ kZ \\ k \end{bmatrix} = \begin{bmatrix} kX \\ kY \\ kZ \\ -k(Z/\lambda) + k \end{bmatrix}$$

Now, to understand the imaging process, let us define a perspective transformation which is given by P equal to (1 0 0 0) (0 1 0 0) then (0 0 1 0) then (0 0 minus 1 upon lambda). You remember, this lambda is focal length of the camera and then 1 and we translate our world coordinate W to the homogeneous coordinates; so, it becomes kX, kY, kZ and k.

Now, **if I translate** if I transform this homogeneous world coordinate by this perspective transformation matrix P, then I get the homogeneous camera coordinates C_h which is given by kX, kY, kZ then minus k into Z by lambda plus k. So, this is the homogeneous camera coordinate.

Now, if just convert this homogeneous **coordinate** camera coordinate to the Cartesian camera coordinate, I find that the Cartesian camera coordinate is given by C equal to small x, small y small z which is nothing but lambda X divided by lambda minus Z, lambda Y divided by lambda minus Z and lambda Z divided by lambda minus Z. So, in the right hand side, this x, y, z they are all in uppercase indicating those are the coordinate of the world point.

(Refer Slide Time: 49:05)



The slide is titled "Image Formation" and features a logo in the top left corner. It contains the following text and equations:

$$x = \lambda X / (\lambda - Z) \quad \text{and} \quad y = \lambda Y / (\lambda - Z)$$

⇒ Coordinate in the image plane of projected 3-D point (X,Y,Z)

⇒ $Z = \lambda Z / \lambda - Z$ is of no interest now

It acts as a free variable in inverse perspective transformation


Now, if I compare this expression with the camera coordinate that we have obtained with respective our previous diagram, you find that here we get lowercase x is nothing but lambda capital X divided by lambda minus Z. Similarly, lowercase y is also nothing but lambda capital Z by lambda minus Z.

So, using our previous diagram, we have also seen that these lowercase x and lowercase y, they are the image point on the image plane of the world coordinate - capital X, capital y and capital Z. So, this shows clearly that using the perspective transformation that we have defined as the matrix p; I can find out the image coordinates of world coordinate point capital X, capital Y capital Z following this particular transformation P.

Now, here in this particular case, the third component that we have obtained that is the value of Z which is not importance in our case because in the camera coordinate, the value of Z is always equal to 0 because we are assuming that the imaging plane is the X Y plane of the world coordinate system as well as the camera coordinate system.

So, we will stop our discussion here today and the next class, we will see that as we have seen that with the perspective transformation; **we can transform a world coordinate**, we can project a world point, a 3D world point onto an imaging plane similarly using the inverse perspective transformation; whether it is possible that given a point in inverse plane whether we can find out the corresponding 3D point in the 3D world coordinate system.

(Refer Slide Time: 51:37)

 **Lecture 4 & 5 Quiz Answers**

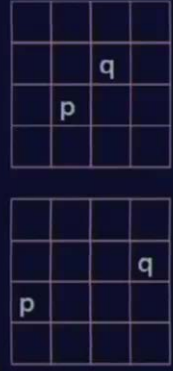
1. In the following figure which of the options are true?

- a. $q \in N_4(p)$
- b. $q \in N_8(p)$
- c. $q \in N_D(p)$

2. Find out

- a. Euclidean
- b. City block
- c. Chess board

distances between p and q in the above figure

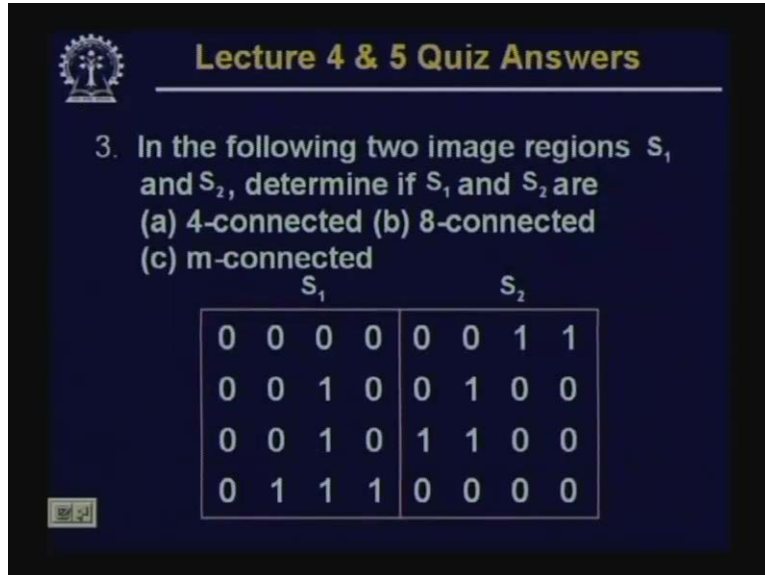


So, now let us discuss about the questions that we have given at the end of lecture 5. The first question that was asked is there was a figure showing 2 points - p and q and the question was we have to find out which of the options are true? That is whether q is a 4 neighbor of p, q is 8 neighbor of p or q is diagonal neighbor.

Now, here you find that in this particular figure, when you look at the locations of p and q, the 4 neighbors of P are the 2 points which are vertically upward and downward and the 2 points which a horizontally to the left and to the right but the point q is in the diagonally upward direction of point and we have said that all the 8 point around point p, they form the 8 neighbors of point p.

So, in this particular case, the point q which a diagonal neighbor point p and it is also one of the 8 neighbors and point p. So, the options b, that is q belonged to N_8 p and q belonging to N_D (p) both these options are true; whereas the option a, that is q belongs to N_4 (p), that is wrong. The second question quit obvious, quit simple. You have to find out the Euclidean distance, city block distance and chess board distance between the 2 points p and q.

(Refer Slide Time: 53:05)



Lecture 4 & 5 Quiz Answers

3. In the following two image regions S_1 and S_2 , determine if S_1 and S_2 are

(a) 4-connected (b) 8-connected
(c) m-connected

S_1				S_2			
0	0	0	0	0	0	1	1
0	0	1	0	0	1	0	0
0	0	1	0	1	1	0	0
0	1	1	1	0	0	0	0

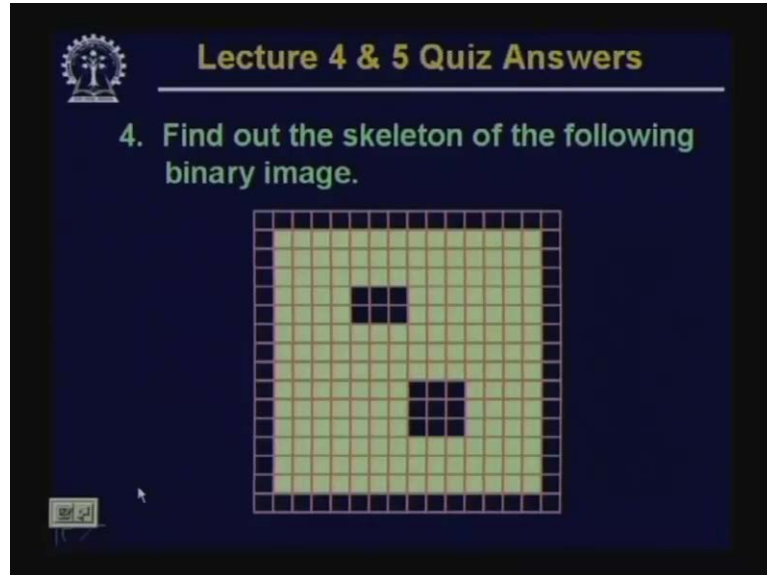
So, this is a very simple question, you can write yourself. Let us go to the third question. Here, we have given 2 sub images S_1 and S_2 and you have determine whether S_1 S_2 are 4 connected, 8 connected or m connected.

If you remember, when we discussed about the connectivity; we have said that 2 images are 4 connected, 2 sub images are 4 connected if there is any point in one of the sub image which is 4 connected to a point in the other terminals.

In this particular case, you find that there is no point in S_1 which is 4 connected with the point in S_2 and similarly, there is no point in S_2 which is 4 connected to a point in S_1 . So, S_1 and S_2 , they cannot be 4 connected. But here you find that I have a point in S_1 , say here and there is a point in S_2 , here, these 2 are actually connected but there are not 4 connected.

So, **these 2 points are connected**, we can say that these 2 points are connected in 8 connected sense because this point is an 8 neighbor of this point. Similarly, this point is also an 8 neighbor of this point and at the same time, these points are also connected in m connected sense because there is no common 4 neighbor of this point of and this point. So, we can say that these image regions S_1 and S_2 ; they are both 8 connected as well as m connected. But these two regions, S_1 and S_2 , they are not for connected.

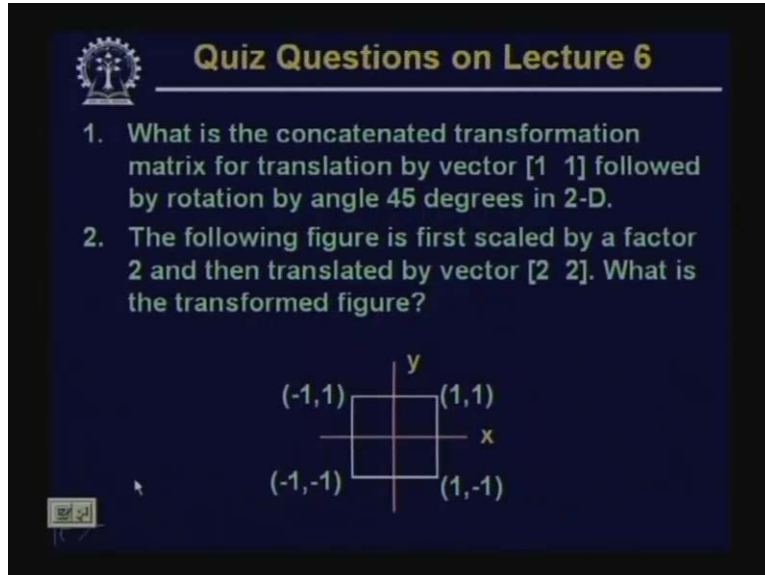
(Refer Slide Time: 54:55)



Now, coming to the next question; here, we had given a binary picture, binary image and you were asked to find out what is skeleton of this binary image. Now, you remember that when we have discussed about the skeletonization of any binary figure, the first operation that we have done was the distance transformation of that particular figure.

Now here, if I find out the distance transformation, you find that all these boundary points, they will get a distance value equal to 1. Similarly, these points will also get distance value is equal to 1, the other points, these points will get distance value is equal to 2, these points will also get distance value is equal to 2. So once, in this manner if you find out the distance values at every point, then from that distance transformation, you can find out the points where there is curvature is discontinuity and we have said that the point said curvature discontinuity, they form the skeleton of the particular image.

(Refer Slide Time: 56:10)



The slide features a logo in the top left corner and the title "Quiz Questions on Lecture 6" in yellow text. It contains two numbered questions. The second question is accompanied by a 2D Cartesian coordinate system with x and y axes. A square is drawn with vertices at (-1, 1), (1, 1), (1, -1), and (-1, -1).

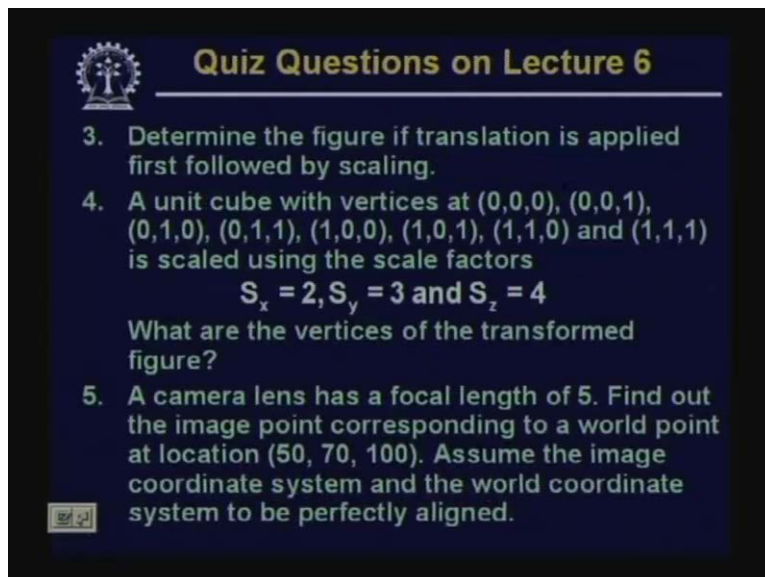
1. What is the concatenated transformation matrix for translation by vector $[1 \ 1]$ followed by rotation by angle 45 degrees in 2-D.

2. The following figure is first scaled by a factor 2 and then translated by vector $[2 \ 2]$. What is the transformed figure?

$(-1, 1)$ $(1, 1)$
 $(-1, -1)$ $(1, -1)$

So now, let us see have some quiz questions on today's lecture. The first question is what is the concatenated transformation matrix for translation by vector $[1 \ 1]$ followed by rotation by angle 45 degree in 2 dimension? The second question is here there a figure, a square with 4 corners. So, if this figure is first scaled by factor 2 and then translated by vector $[2 \ 2]$; what is the transformed figure?

(Refer Slide Time: 56:47)



The slide features a logo in the top left corner and the title "Quiz Questions on Lecture 6" in yellow text. It contains three numbered questions. The second question includes a set of scale factors $S_x = 2, S_y = 3$ and $S_z = 4$.

3. Determine the figure if translation is applied first followed by scaling.

4. A unit cube with vertices at $(0,0,0)$, $(0,0,1)$, $(0,1,0)$, $(0,1,1)$, $(1,0,0)$, $(1,0,1)$, $(1,1,0)$ and $(1,1,1)$ is scaled using the scale factors $S_x = 2, S_y = 3$ and $S_z = 4$. What are the vertices of the transformed figure?

5. A camera lens has a focal length of 5. Find out the image point corresponding to a world point at location $(50, 70, 100)$. Assume the image coordinate system and the world coordinate system to be perfectly aligned.

Third question - determine the figure if translation is applied first followed by scaling? Fourth question - a unit cube with vertices $(0 \ 0 \ 0)$ $(0 \ 0 \ 1)$ $(0 \ 1 \ 0)$ $(0 \ 1 \ 1)$ $(1 \ 0 \ 0)$ $(1 \ 0 \ 1)$ $(1 \ 1 \ 0)$ and $(1 \ 1 \ 1)$

1) is scaled using S_x equal to 2, S_y equal to 3 and S_z equal to 4; then what are the vertices of the transformed figure?

The fifth one - a camera lens as a focal length of 5, find out the image point corresponding to a world point at location (50, 70, 100). Assume the image coordinate system and the world coordinate system to be perfectly aligned.

Thank you.