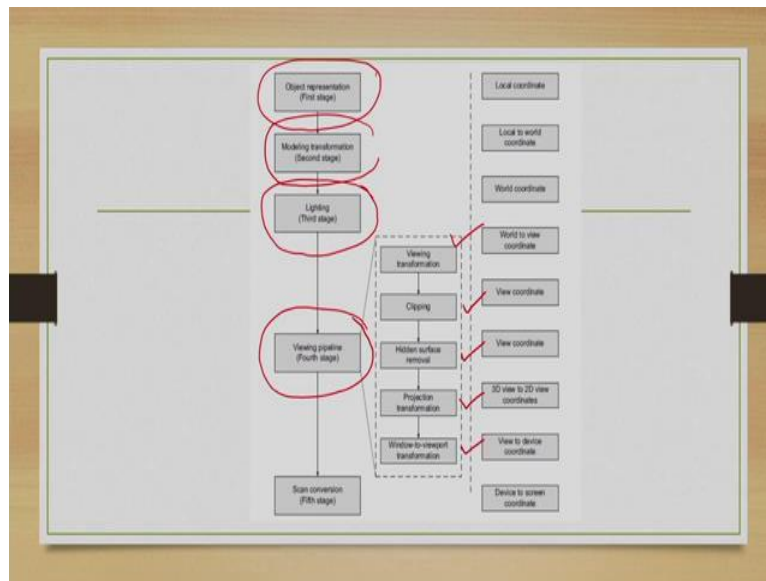


Computer Graphics
Professor. Samit Bhattacharya
Department of Computer Science and Engineering
Indian Institute of Technology, Guwahati
Lecture No. 12
Transformations in 3D

Hello and welcome to lecture number 12 in the course Computer graphics. So, we are, as you may recollect, discussing the graphics pipeline, and as we are doing for last few lectures, we will start with having a relook at the pipeline stages so that we are able to remember it better.

(Refer Slide Time: 0:58)



So, there are 5 stages in the graphics pipeline, the first stage is object representation, second stage is modelling or geometric transformation third stage is lighting or assigning colour to points on the objects, fourth stage is viewing pipeline where we transfer a 3D object to a 2D viewing plane. The transformation takes place through 5 sub stages viewing transformation, clipping, hidden surface removal, projection transformation and window to viewport transformers and the fifth and final stage is scan conversion. Here, we actually map the view plane object to the pixel grid on the screen.

And as we have mentioned earlier, each of these stages take place in specific coordinate systems, object representation is done in local or object coordinate system, modelling transformation here we actually transfer from local to world coordinate system, lighting takes place in world coordinate, then viewing pipeline takes place in 3 coordinate systems; world coordinate, view coordinate and then device coordinate. And finally, scan conversion takes

place in screen coordinate system. So, the different coordinates are involved in different stages of the pipeline.

(Refer Slide Time: 3:00)



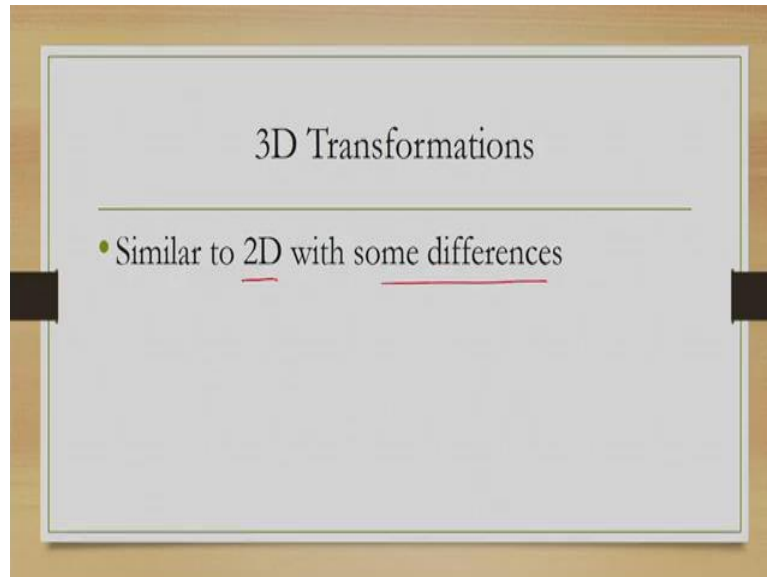
Among these stages, so far we have discussed the first stage object representation. Currently we are discussing the second stage that is modelling or geometric transformation. And in the last couple of lectures, we have discussed the basic transformation idea including how to perform complicated transformations in terms of sequence of basic transformation, but all our discussion were based on 2D transformations.

(Refer Slide Time: 3:32)



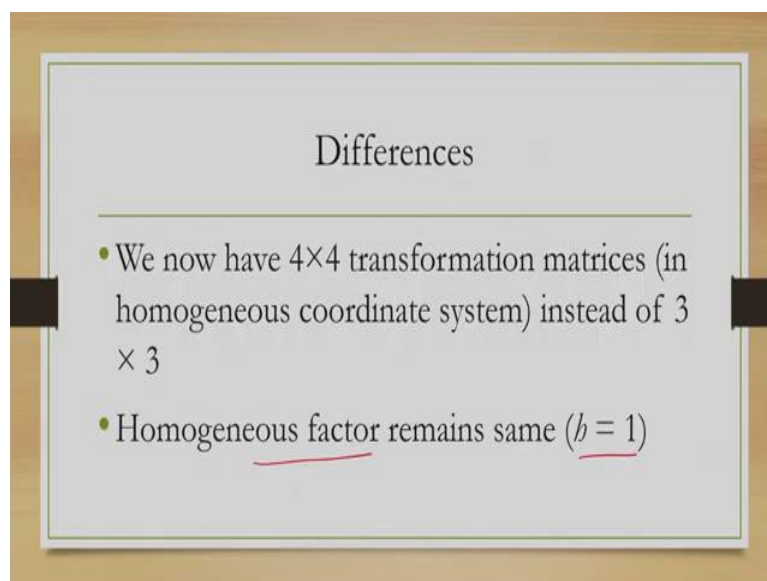
In other words, we were performing transformations in 2 dimensional reference frame. Now, let us have a look at 3D transformation. So, 3D transformation will be the topic of discussion for our lecture today.

(Refer Slide Time: 4:12)



So, when we talk of 3D transformation, essentially we refer to all the basic transformations that we have already discussed in 2D but in a modified form. And the transformations are actually same as in 2D, but their representation is different. In 2D, we discussed 4 basic transformations, namely translation, rotation, scaling and shearing. Now, these 4 remainders basic transformations in 3D word also. However, their representation is different.

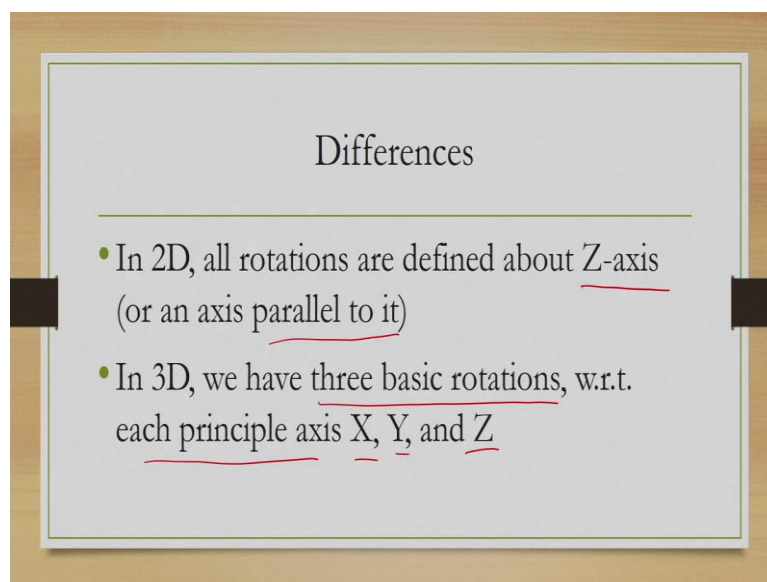
(Refer Slide Time: 04:47)



So, earlier we had used homogeneous coordinate system to represent the transformation. We will use the same coordinate system here to represent the 3D transformation as well, but with the difference. Now, earlier in the matrix representation, we used 3×3 matrices in the homogeneous coordinate system to represent each of the transformations. In 3D, we use 4×4 matrices to represent each transformation. However, the homogeneous factor h remains the same that is $h=1$.

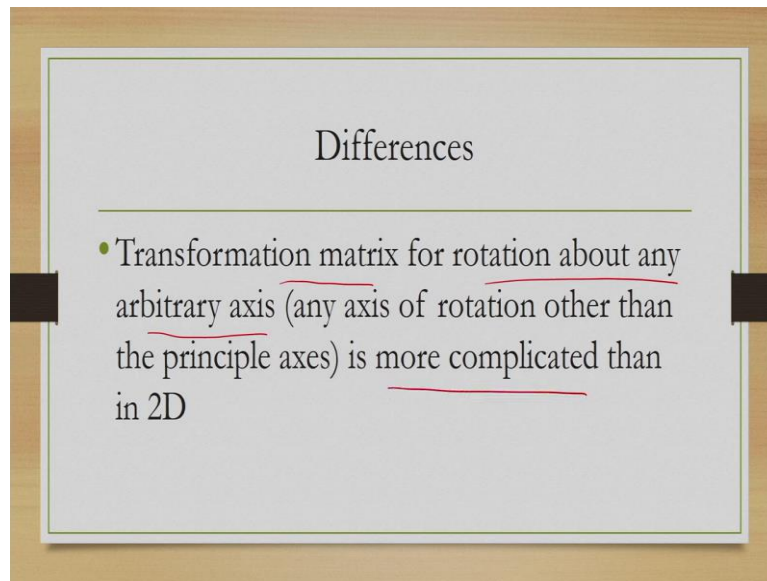
So, essentially we are using instead of 3×3 , we are using 4×4 transformation matrices to represent a transformation in 3D, and the homogeneous factor h remains equal to 1 because we are dealing with modelling transformation. But there are certain differences and we should keep in mind these differences, the differences are primarily with respect to the 2 transformations; rotation and shearing.

(Refer Slide Time: 6:10)



In rotation earlier, we assumed that the rotations are taking place with respect to the z axis or some axis that is parallel to it. That was our basic assumption in 2D rotations. In 3D this assumption is no longer valid, here we have 3 basic rotations with respect to each principle axis x , y and z . Earlier we had defined only one rotation with respect to z axis. Now, in 3D we are defining 3 basic rotations with respect to the 3 principle axis x , y and z , so number of basic transformations changed. So, earlier we had one for rotation, now we have 3 for rotation also.

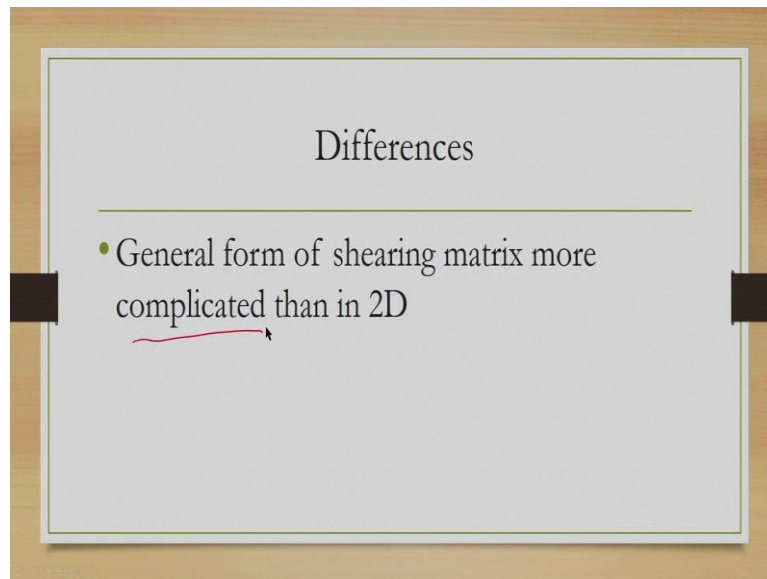
(Refer Slide Time: 7:05)



Also previously, we did not face this situation when we defined rotation with respect to z axis. Now here, transformation matrix that we should use to represent rotation about any arbitrary axis that means, any axis that is not the principle axis, is more complicated than in 2D. So, in 2D we can have only z as principle axis, in 3D we have 3 principle axis, we have to take into account all 3.

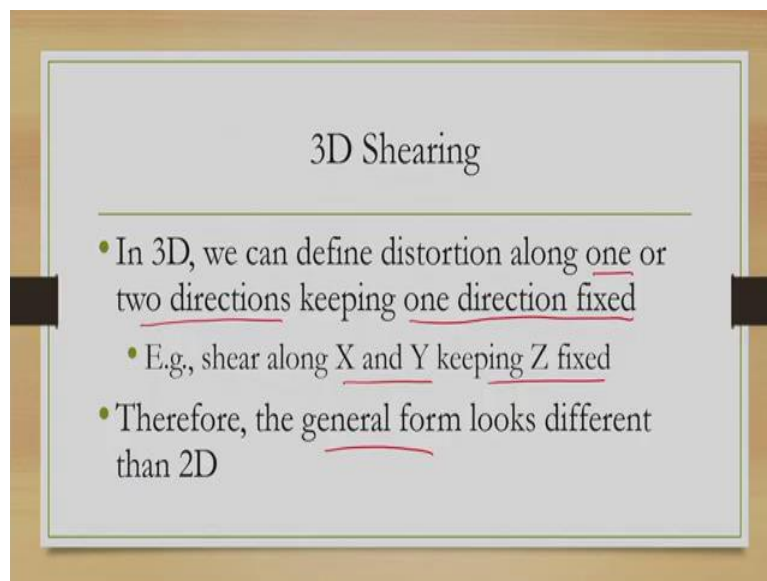
So, when we are trying to define an arbitrary rotation with respect to any arbitrary axis then deriving the transformation matrix becomes more complicated. And the form of the matrix also is more complicated than what we have encountered in 2D. We will have a look at this derivation of rotation matrix with respect to any arbitrary axis later in the lecture that is about rotation.

(Refer Slide Time: 8:33)



Now as I said, shearing is also having some difference with respect to its 2D counterpart. It is in fact more complicated compared to what we have seen in 2D.

(Refer Slide Time: 8:50)



So, let us start our discussion with shearing in 3D then we will talk about the differences in rotation and then we will see how to derive a composite transformation matrix for rotation about any arbitrary axis. Now, when we are talking of shearing, as we have seen earlier we are trying to basically change the shape of the object. So, essentially to introduce some deformity in the object shape. Now this distortion or deformation can be defined along 1 or 2 directions at a time while keeping 1 direction fixed that is 1 constrain that we follow for defining shearing in 3D.

For example, if we are trying to shear along x and y direction then we have to keep z direction shearing fixed as a result, the general form is different than in 2D shearing.

(Refer Slide Time: 10:11)

Shearing Matrix

- There are six shearing factors
- Each can take any real value or zero (if no shear along that particular direction)

$$\begin{bmatrix} 1 & sh_{xy} & sh_{xz} & 0 \\ sh_{yx} & 1 & sh_{yz} & 0 \\ sh_{zx} & sh_{zy} & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

In fact, we can define 6 shearing factors. Recollect that shearing factor refers to the amount of distortion or deformation we want to introduce along a particular axis. So, in this case in case of 3D shearing we can define 6 shearing factors and each factor can take any real value or zero if no shear along that particular direction takes place, so when the shearing factor is 0 that means, along that direction there is no shearing. And with respect to the six factors, the shearing matrix looks something like this, where sh_{xy} , sh_{xz} , sh_{yx} , sh_{yz} , sh_{zx} and sh_{zy} are the six shearing factors.

(Refer Slide Time: 11:25)

Shearing Matrix

- sh_{xy} and sh_{xz} are used to shear along Y and Z directions, respectively, leaving X coordinate value unchanged

$$\begin{bmatrix} 1 & sh_{xy} & sh_{xz} & 0 \\ sh_{yx} & 1 & sh_{yz} & 0 \\ sh_{zx} & sh_{zy} & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Among these factors sh_{xy} and sh_{xz} are used to shear along y and z directions respectively leaving the x coordinate value unchanged. We earlier mentioned that while performing shearing one direction has to be left unchanged. So, in this case, we are performing shearing along y and z directions whereas, shearing along x direction remains 0.

(Refer Slide Time: 12:09)

Shearing Matrix

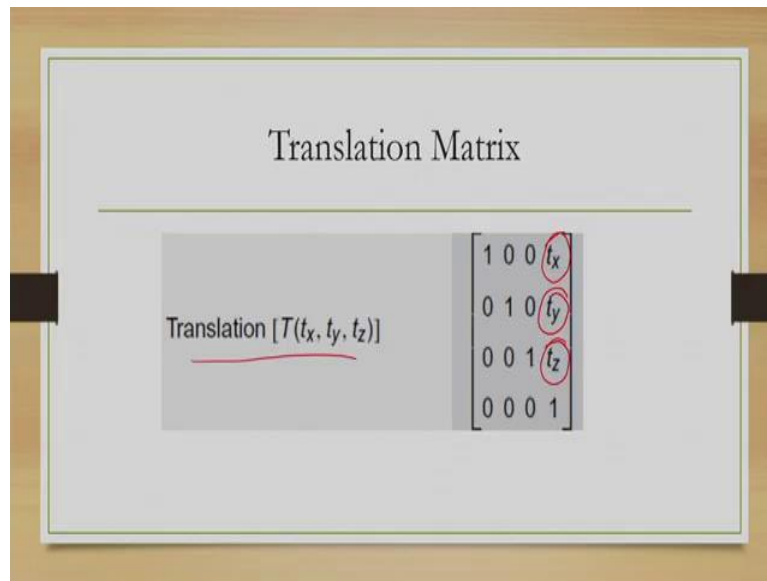
- sh_{zx} and sh_{zy} used to shear along X and Y directions, respectively, leaving z coordinate value unchanged

$$\begin{bmatrix} 1 & sh_{xy} & sh_{xz} & 0 \\ sh_{yx} & 1 & sh_{yz} & 0 \\ sh_{zx} & sh_{zy} & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Similarly, sh_{yx} and sh_{yz} refers to the shearing factors along x and z direction when y coordinate value remains unchanged. And likewise, the other 2 shearing factors can be defined that is sh_{zx} and sh_{zy} , these 2 refer to shearing along x and y direction leaving z value unchanged. So, each pair actually refers to shearing along 2 directions while the third directions remain unchanged that means shearing along that third direction does not take place.

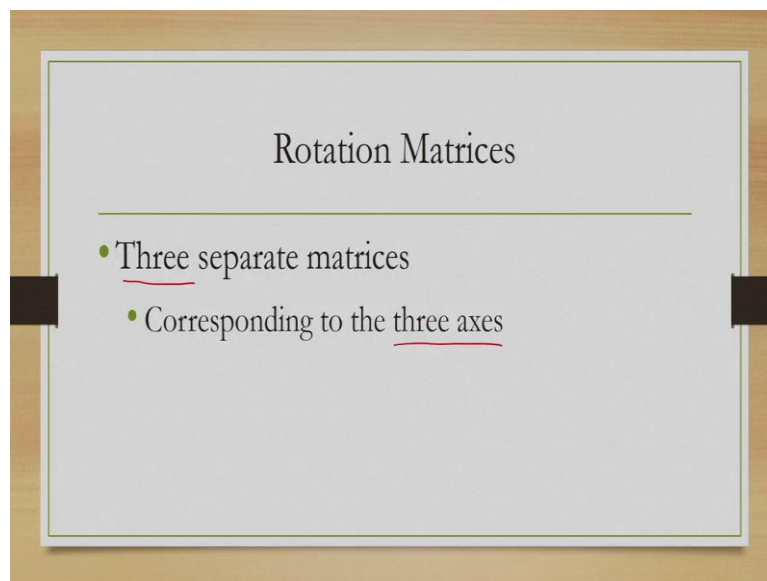
So, that is about shearing as you can see it is more complicated compared to the shearing matrix that we have seen for 2D transformation that is because we now have 6 shearing factors. Now, let us have a look at other transformation matrices basic transformation matrices.

(Refer Slide Time: 13:31)



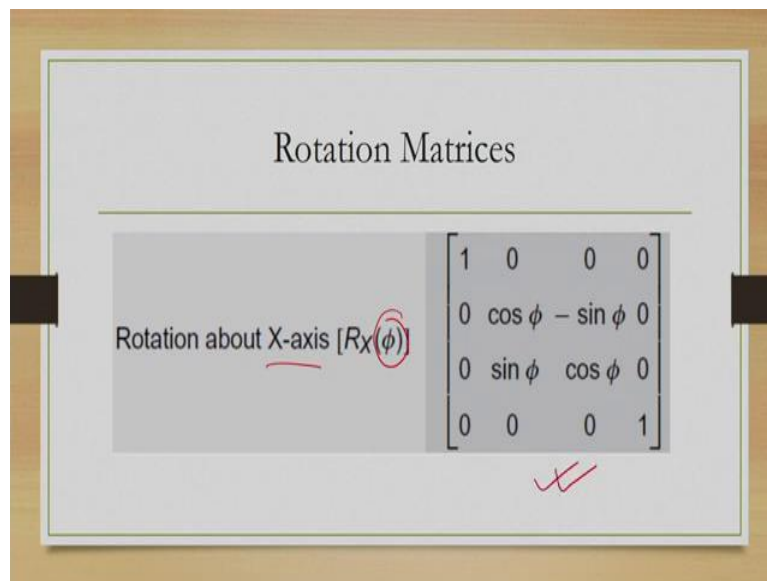
Translation is the simplest and the form remains almost the same with the addition of one more dimension. So, we have t_x referring to translation along x direction, t_y referring to translation along y direction and t_z referring to translation along z direction.

(Refer Slide Time: 14:01)



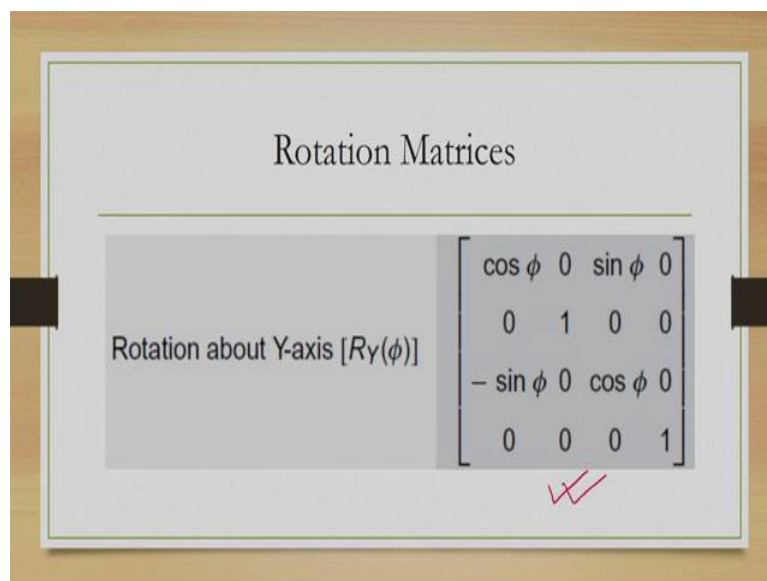
As I said before, for rotation, we do not have a single matrix. Instead, we have 3 separate matrices, each matrix corresponding to the rotation along a particular principle axis. So, therefore, since there are 3 axis, so we have 3 rotation matrices.

(Refer Slide Time: 14:31)



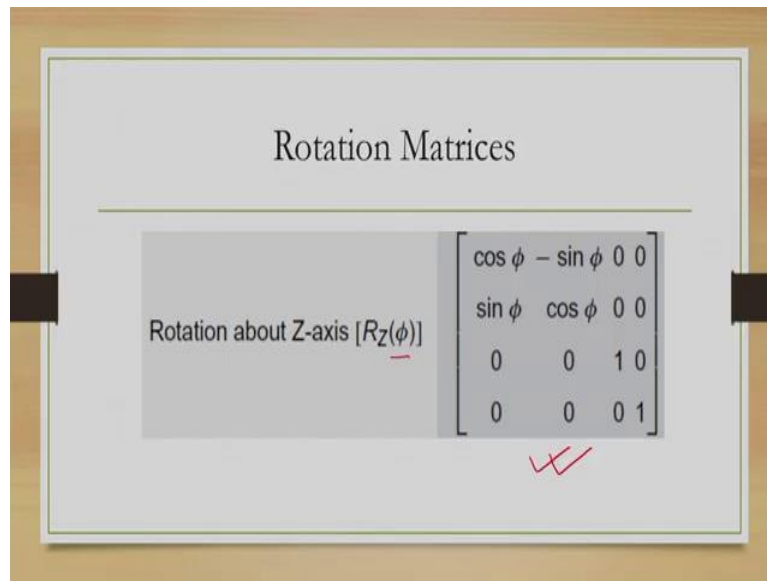
Rotation about x axis, when the angle of rotation is ϕ looks something like this matrix.

(Refer Slide Time: 14:42)



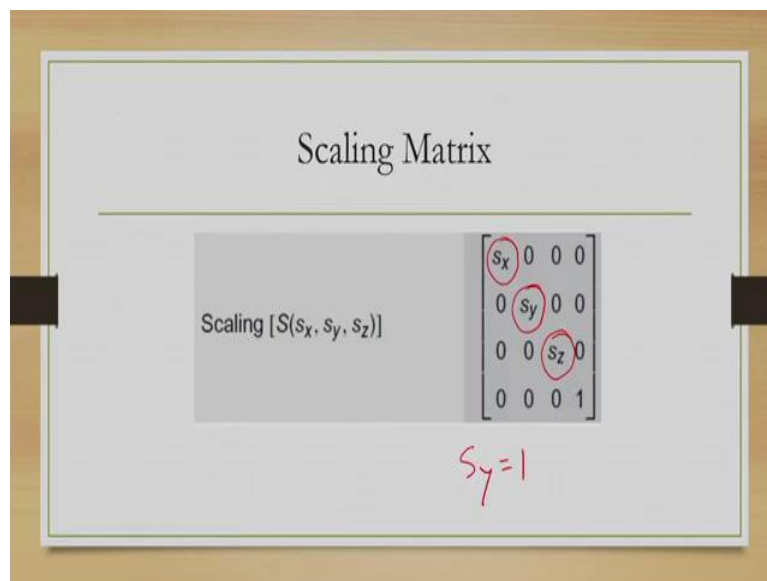
Rotation along the y axis again, assuming the rotation angle to be ϕ is shown here.

(Refer Slide Time: 15:06)



And finally, rotation about z axis by an angle ϕ is shown here in this matrix. So, we have 3 matrices representing 3 basic rotations; one about x axis, one about y axis and one about z axis.

(Refer Slide Time: 15:31)

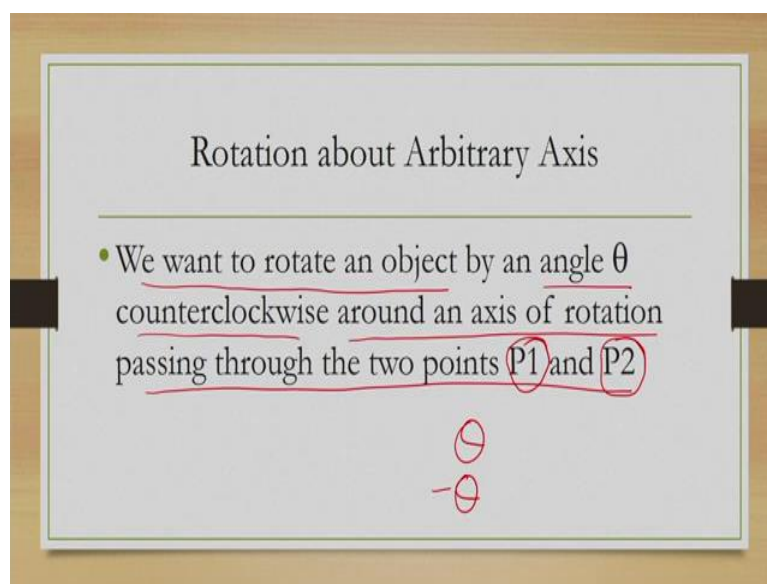


Scaling is also similar to the 2D counterpart, S_x is the scaling factor along x direction, S_y is the scaling factor along y direction, S_z is the scaling factor along z direction. So, if we do not want to perform any scaling along a particular direction, we simply set that particular scaling factor as 1. So, if we do not want scaling along say y direction, then we will set $S_y=1$. And if you may recollect scaling factor less than 1 means, in that particular direction we want to

reduce the size and scaling factor greater than 1 means in that particular direction we want to increase the size.

So, scaling is related to size, shearing is related to shape, translation and rotation is related to position. So, then we have in 3D more than 3 basic matrices, we have one for translation, one for scaling, one for shearing, and three for rotation so total 6 basic matrices representing 6 basic transformations in 3D. The other difference that I mentioned with respect to 2D transformation is the rotation of an object with respect to any arbitrary axis that means, any axis that is not one of the principle axis x , y and z .

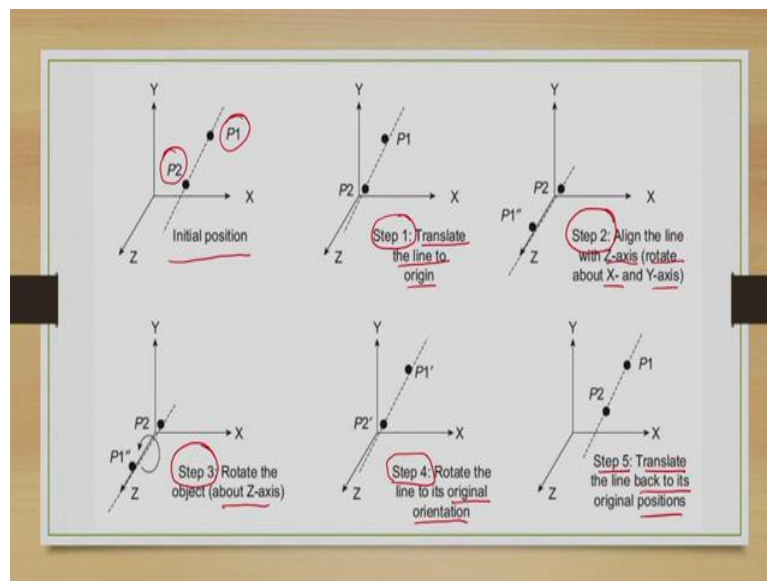
(Refer Slide Time: 17:17)



So, what is the idea that we want to rotate an object by an angle θ counter clockwise around an axis of rotation passing through 2 points P_1 and P_2 . So, here we are defining these two points because with these two points, we can define a line or line segment that represents the axis of rotation. So, unless we mentioned the points, it will be difficult to represent the axis. So, then we have an axis defined by the 2 points, we have an angle of rotation theta, which is counter clockwise.

Remember that we are using a convention that if angle of rotation is counter clockwise then it is positive angle, if angle of rotation is clockwise, then we consider it to be negative angle. So, if we are rotating the object by an angle θ counter clockwise, then it will be simply θ , but if we are rotating the same object by an angle θ clockwise, then we will replace θ with $-\theta$. Now, let us see what happens when we are trying to perform this rotation with respect to any arbitrary axis, how we can derive a composite matrix representing this rotation.

(Refer Slide Time: 18:59)



The idea is illustrated in the series of steps. So, this one top left figure shows the initial situation where P_1 and P_2 define the axis of rotation represented with the dotted line with respect to the 3D reference frame or coordinate frame. Now then, in step 1, what we do? We translate the line to the origin. Remember, earlier in our discussion on composition of transformation, we discussed how to combine multiple transformations.

So, there what we said that if we are trying to perform some basic operation with respect to any arbitrary fixed point other than origin, then what we follow? We first translate the point to origin, perform the basic transformation and then translate it back to its original location. So, the same basic principle we are following here, we are given the arbitrary axis or arbitrary fixed line. In the first step, we translate it to the origin that means the axis passes through the origin.

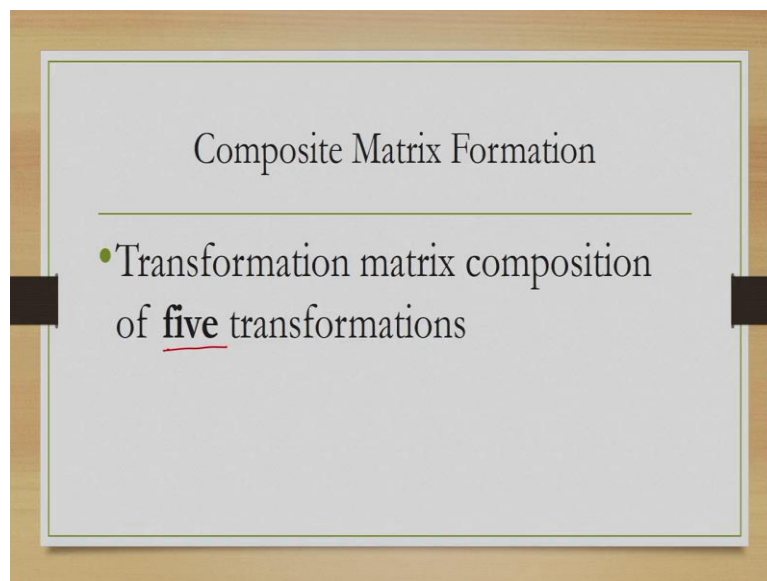
In step 2, what we do? Now the axis passes through the origin, but there is no guarantee that it aligns with any of the principle axis. So, in this step 2, we align the line with z axis in our particular explanation, but it is not necessary to always align with z axis, instead you can always align it with either x or y axis as well. But let us assume that we are aligning it with the z axis. So, then that involves rotation about x and y axis.

So, now our arbitrary axis is aligned with the z axis. So, rotation will take place around or about z axis that we do in Step 3, we apply the rotation about the z axis. After the rotation is done in step 4, what we do is, we rotate the line back to its original orientation. So, when we brought it or translated it in the step 1 to pass it through origin, it had one orientation. So in

step 4, we return it to that orientation and in step 5 or the final step, we translate it back to its original position.

So in step 4, we are returning it to its original orientation and in step 5 we are translating it back to its original position. So, these 5 steps are needed to construct the composite matrix representing rotation of an object with respect to any arbitrary axis. So, let us try to derive it then.

(Refer Slide Time: 22:39)

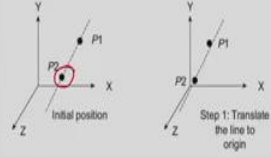


As we have seen in the figure, so there are 5 transformations. So, the composite matrix or the ultimate final, the final transformation matrix would be a composition of these 5 basic transformations.

(Refer Slide Time: 23:03)

Sequence

- 1. Translation to origin ($T[-x, -y, -z]$: (x, y, z) is the coordinate of P_2)

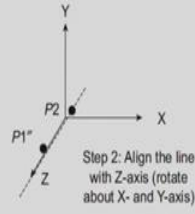


So, the first transformation is translation. Translating the line so that it passes through origin. Now, the translation amount would be minus x, minus y, minus z since we are moving along the negative z direction where x, y, z is the coordinate of P_2 , one of the endpoints.

(Refer Slide Time: 23:30)

Sequence

- 2. Alignment of axis of rotation with Z (can be X or Y also)
 - 2.1 Rotation about X, to put axis on XZ plane [$R_x(\alpha)$: α is angle of rotation about X]
 - 2.2 Rotation about Y, to align axis with Z [$R_y(\beta)$: β is angle of rotation about Y]



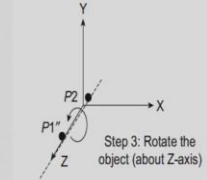
Then in step 2, we align the line to the z axis, but as I said it need not be always z axis, it can be x or y axis also. So, in order to do that what we need to do? We need to perform some rotations about x and y axis. So first, let us assume that first we are rotating the line about x axis to put the axis on the x-z plane and the angle of rotation is α . Then, we are rotating it about the y axis to align the axis with the z axis.

So, first we rotate it about x axis to put it on the x-z plane and then we rotate it about y axis to align it with the z axis. So, in the first case the angle of rotation let us denote it by α and in the second case, let us denote it by β , both are anticlockwise rotation so both are positive at this stage.

(Refer Slide Time: 24:59)

Sequence

- 3. Rotation of the object about Z [$R_z(\theta)$: θ is angle of rotation of the object about the axis of rotation]



Then in stage 3, what we do? Now we have aligned the axis with z axis and then we perform the rotation about z axis which is our original objective. So then, we use the rotation matrix with respect to z axis, so here θ is the angle of rotation of the object. Remember that this θ angle of rotation is with respect to arbitrary axis, now we are using it to rotate about z axis because we have aligned arbitrary axis with the z axis.

(Refer Slide Time: 25:46)

Sequence

- 4. Reverse rotation about Y and X to bring axis of rotation back to its original orientation

$$\begin{matrix} X & Y \\ Y & X \end{matrix}$$

Step 4: Rotate the line to its original orientation

Then, in step 4 and 5, we reverse the operations we performed in step 1 and 2. So first, we take the line to its original alignment, which involves reverse rotation about y and x axis to bring the axis of rotation back to its original orientation. While aligning, we rotated with respect to x first and then y. Since now, we are reversing the operation, so we rotate it with respect to y first and then x.

(Refer Slide Time: 26:28)

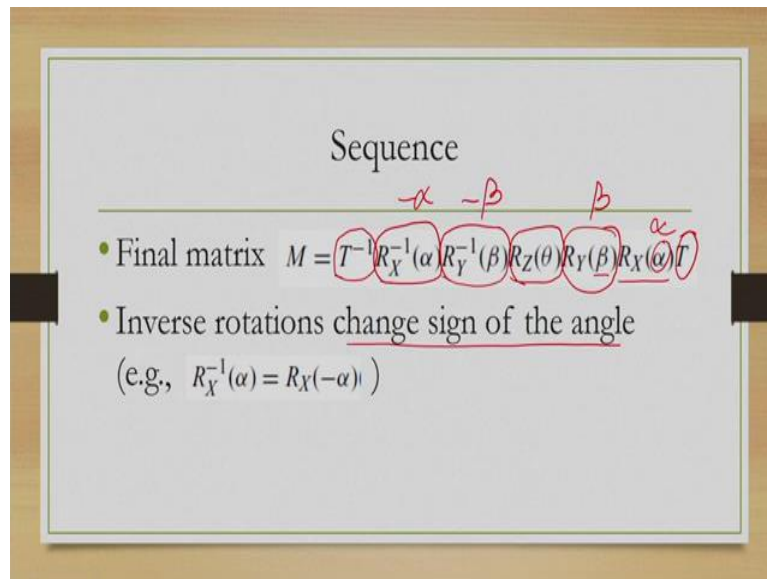
Sequence

- 5. Translation of axis of rotation back to its original position

Step 5: Translate the line back to its original positions

And in Step 5, what do we do? We then translate it back to its original position, which is the last step. So, then what would be the composite matrix?

(Refer Slide Time: 26:43)



We can get it by matrix multiplication and we will follow the right to left rule. So, first we perform the translation to take the line passing through origin, then we performed a rotation about x axis by an angle α to bring the line on the x-z plane then we perform a rotation by an angle β around the y axis to align it with z axis, then we performed the actual rotation by an angle θ with respect to the z axis. Then we reverse the earlier steps that is first we perform rotation with respect to y, then rotation with respect to x by the same angle amount as in the earlier cases and then reverse translation.

Now, since we are rotating in inverse of what we did in step 2, now these inverse rotations can simply be represented by a change of sign of the angle. So, earlier if the angle was β then it will be $-\beta$ here and if the angle was α , then it will be $-\alpha$ here. So, when we rotate it about x axis with α , in case of reverse rotation we will rotate about x axis by $-\alpha$. Similarly, we rotated here with β , here we will rotate by $-\beta$. So, the reverse rotation means changing the angle of rotation because from counter clockwise we are now rotating clockwise.

So, these matrices, multiplied in the particular sequence shown here will give us the composite matrix for rotating an object by an angle θ about any arbitrary axis of rotation. So that is in summary, what are there in 3D transformation. So, it is mostly the same with 2D transformation with some differences. First difference is that in homogeneous coordinate system, now we require 4×4 matrices instead of 3×3 matrices to represent each transformation.

Then earlier we defined 4 basic transformations namely, translation, rotations, scaling and shearing in the context of 2D transformation. Now we have 6 basic transformations; translation, rotation about x axis, rotation about y axis, rotation about z axis, scaling and shearing. Earlier we had defined 2 shearing factors, now there are 6 shearing factors, it is a bit more complicated than the earlier case.

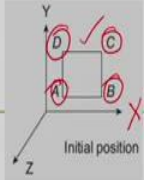
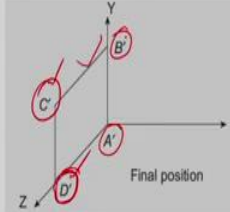
Now, in shearing, when he perform shearing along 2 axis, 2 principle axis, there is no shearing along the third principle axis that we follow in 3D shearing. Apart from these differences, there is another major difference in the way we derive composite transformation matrix for rotation about any arbitrary axis.

So, in order to do that, we follow 5 step process, first we translate the line to pass through origin, then we align it with one of the principle axis, then we perform the rotation by the desired angle about that axis, then we place the line back to its original orientation by performing reverse rotations, and then we translate it back to its original position. And we put the individual basic matrices in right to left manner to get the final composite matrix as we have shown in the discussion. Now, let us try to understand the 3D transformation with respect to 1 illustrative example.

(Refer Slide Time: 32:01)

Example

- An object ABCD defined as $A(1,1,0)$, $B(3,1,0)$, $C(3,3,0)$, and $D(1,3,0)$. It is required to construct a **partition wall** $A'B'C'D'$ in a scene (A' corresponds to A and so on). New vertices are $A'(0,0,0)$, $B'(0,4,0)$, $C'(0,4,4)$, and $D'(0,0,4)$. Calculate composite transformation matrix.

Let us consider a situation, there is an object defined with the vertices defined with the vertices A B C D here in this figure top figure, as you can see it is on the x-y plane This is the initial situation, now we want to use this particular object to construct a partition wall defined

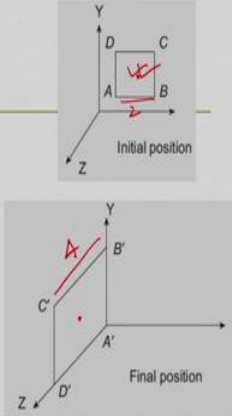
by the vertices A' , B' , C' and D' in a scene where A' corresponds to A , B' corresponds to B , C' corresponds to C and D' corresponds to the vertex D .

So, here as we can clearly see some transformation took place, the question is try to calculate the composite transformation matrix that enables this object to be positioned as a partition wall in this scene. Let us see how we can do this.

(Refer Slide Time: 33:34)

Example

- Initially the square is in the XY plane with each side equal to 2 units and center at $(2, 2, 0)$
- The final square is on the YZ plane with side equal to 4 units and the center at $(0, 2, 2)$



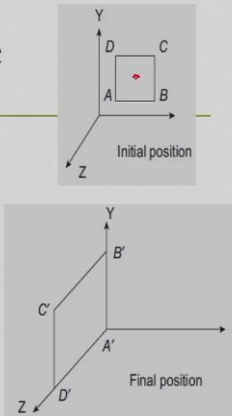
The diagram consists of two 3D coordinate systems. The top one, labeled 'Initial position', shows a square in the XY plane with vertices A, B, C, and D. The bottom one, labeled 'Final position', shows a larger square in the YZ plane with vertices A', B', C', and D'. Red arrows and markings indicate the transformation from the initial to the final state.

So, initially the square is in the x-y plane and each side had 2 units of length and the centre is given as $(2, 2, 0)$. The final square is on the y-z plane with each side equal to 4 units and the centre is now at $(0, 2, 2)$. Now, these lengths and centres can be found out by the coordinates of the vertices.

(Refer Slide Time: 34:28)

Transformation Sequence

- Translate center to origin $\rightarrow T(-2, -2, 0)$



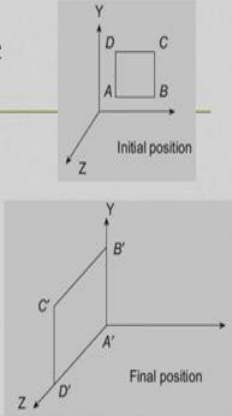
The diagram is identical to the one in the previous slide, showing the initial square in the XY plane and the final square in the YZ plane.

So, then what we need to do? So, in this case, we need a rotation from x-y plane to y-z plane, but the axis of rotation is not z axis, it is parallel to z axis so we will follow this composite matrix transformation creation approach. So, first we translate the centre to origin, centre of this original object so then the translation amount will be -2, -2 and 0.

(Refer Slide Time: 35:13)

Transformation Sequence

- Rotate by 90 degrees anticlockwise around z-axis → $R_z(90)$



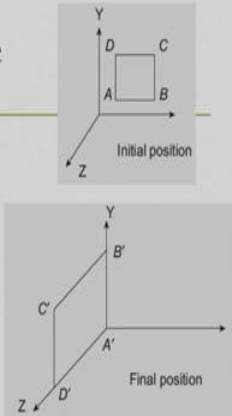
The diagram illustrates a 3D coordinate system with X, Y, and Z axes. In the 'Initial position', a square ABCD is shown in the XY-plane, with vertices A, B, C, and D. In the 'Final position', the square has been rotated 90 degrees anticlockwise around the Z-axis, resulting in a new square A'B'C'D' where the vertices are now aligned with the Z-axis and the Y-axis.

So, if we are translating the centre to origin then the axis of rotation which was parallel to z axis now will be automatically aligned with the z axis. So, then we perform the rotation by 90 degrees anti clockwise around the z axis. So, we will use the rotation matrix defined for rotation about z axis with the angle of rotation 90. Since the rotation is anti-clockwise, so it will be positive angle.

(Refer Slide Time: 35:58)

Transformation Sequence

- Rotate by 90 degrees anticlockwise around y-axis → $R_y(90)$



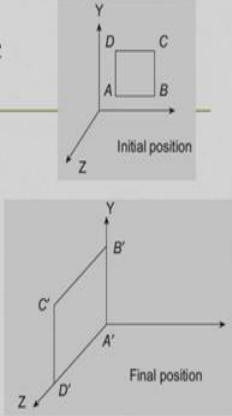
The diagram illustrates a 3D coordinate system with X, Y, and Z axes. In the 'Initial position', a square ABCD is shown in the XY-plane, with vertices A, B, C, and D. In the 'Final position', the square has been rotated 90 degrees anticlockwise around the Y-axis, resulting in a new square A'B'C'D' where the vertices are now aligned with the Z-axis and the X-axis.

Then we rotate by 90 degrees anti clockwise around y axis. So, again we will use $R_y(90)$ where $R_y \theta$ is the basic rotation matrix about y axis.

(Refer Slide Time: 36:28)

Transformation Sequence

- Scale up by 2 in Y and Z direction $\rightarrow S(1, 2, 2)$



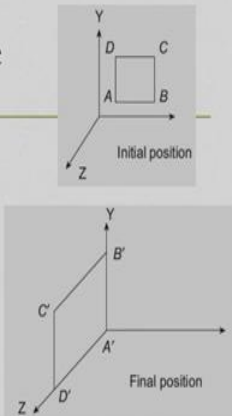
The diagram illustrates a 3D coordinate system with X, Y, and Z axes. In the 'Initial position', a square is drawn in the XY plane with vertices labeled A, B, C, and D. In the 'Final position', the square is scaled up by a factor of 2 in the Y and Z directions, resulting in a larger square with vertices labeled A', B', C', and D'.

Then we perform scaling because the size increased scale up by 2 in y and z direction, so x direction will have a scaling factor 1, there is no change and z and y direction will have scaling factor 2, the size will double.

(Refer Slide Time: 36:54)

Transformation Sequence

- Translate center to (0,2,2) $\rightarrow T(0, 2, 2)$



The diagram illustrates a 3D coordinate system with X, Y, and Z axes. In the 'Initial position', a square is drawn in the XY plane with vertices labeled A, B, C, and D. In the 'Final position', the square is translated to a new center at (0, 2, 2) and scaled up by a factor of 2 in the Y and Z directions, resulting in a larger square with vertices labeled A', B', C', and D'.

And then we translate the centre to the new object centre using the translation matrix.

(Refer Slide Time: 37:14)

Final Matrix

$$M = T(0, 2, 2)S(1, 2, 2)R_Y(90^\circ)R_Z(90^\circ)T(-2, -2, 0).$$

Initial position *Final position*

P' = M.P
h=1

So, then the composite transformation matrix can be obtained by multiplying these individual basic transformation matrices together where we followed the right to left rule that is, first is the translation to origin, then rotation about z axis, then rotation about y axis, then scaling up by 2 along y and z direction, then translation to the new origin. So if we multiply, we will get the new composite transformation matrix.

And after we get this matrix, just to recap the procedure, what we need to do? We need to multiply each vertex with this composite transformation matrix. So, if vertex is represented by column vector P and this composite transformation matrix is M, then we perform this (M.P) for each vertex to get the new vertex position in homogeneous coordinate. So, eventually to get the physical coordinate, we perform this operation, we divide the x coordinate by homogeneous factor, y coordinate by homogeneous factor and z coordinate by the homogeneous factor.

So in our case of course, $h=1$. So, it really does not matter, the x, x and z coordinate will remain the same. But later on, as I mentioned earlier, we will see that there are situations where $h \neq 1$. So, in that case, this division is very important that we will see in subsequent lectures.

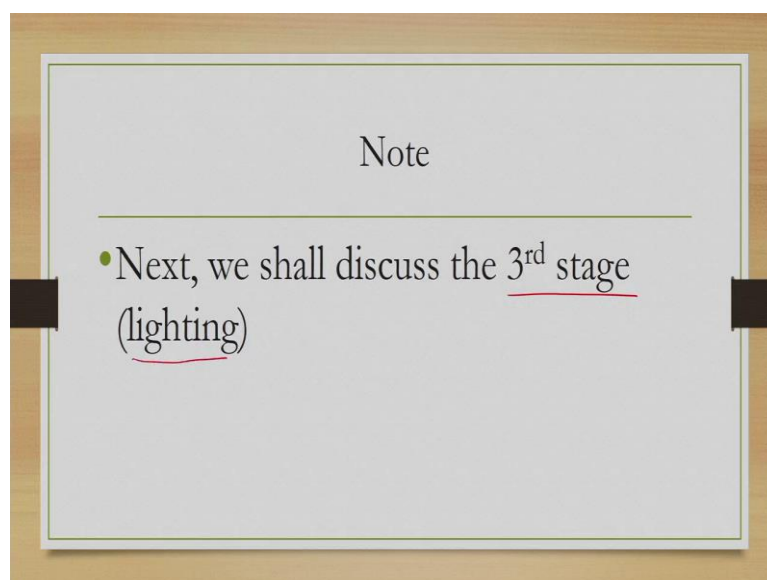
So, with that, we come to a conclusion to our discussion on 3D transformation. And also, we have come to a conclusion to our discussion on the second stage that is modelling transformation. So, we started our discussion with 2D transformations, there we introduced the basic idea of modelling transformation that is to assemble objects that are defined in their

own or local coordinate systems into a world coordinate scene. In order to do that, we perform geometric transformations, any transformation can be considered to be a sequence of basic transformations in 2D transformations.

We have discussed 4 basic transformations, those are translation, rotation, scaling and shearing. We also discussed why it is important to represent transformations in terms of matrices, because of modularity and compatibility with subsequent stages when we are implementing a package in the form of library functions or APIs or standard functions. Now, for matrix representation we discussed the importance and significance of homogeneous coordinate system and we have seen how to use the homogeneous coordinate system to represent basic transformations or any composite transformation.

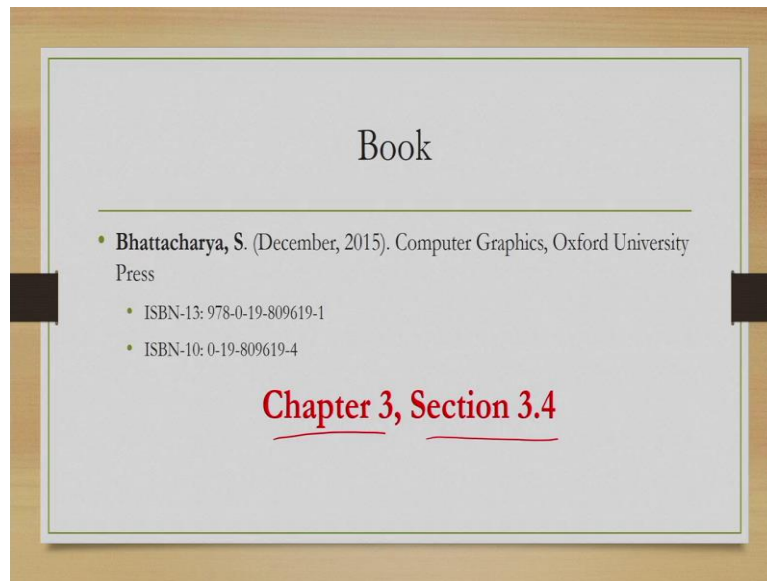
So, in summary, in modelling transformation, we perform transformation by considering individually or in sequence basic transformations, these transformations are represented in the form of matrices, where the matrices are themselves representations in homogeneous coordinate system. And in 2D transformation, we have 4 basic transformations. In 3D modelling transformations, we have 6 basic transformers. And any transformation with respect to any arbitrary point or axis of rotation can be derived by using a sequence of basic transformations, the way we derive composite transformation.

(Refer Slide Time: 42:13)



So in the next lecture, we shall start our discussion on the third stage of the graphics pipeline that is assigning colour or the lighting.

(Refer Slide Time: 42:31)



Whatever I have discussed today can be found in this book. And you may refer to chapter 3, section 3.4 to know more about the topics that I have covered today. So, we will meet you in the next lecture. Till then, thank you and goodbye.