Computer Graphics

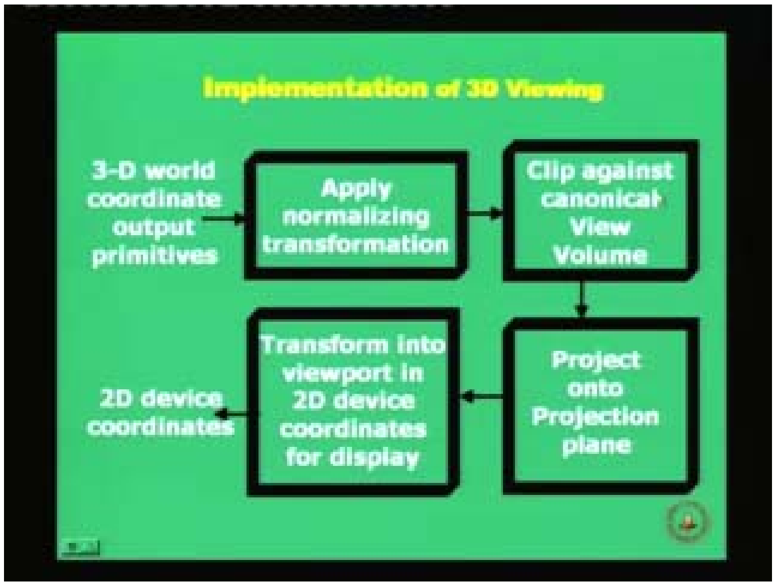
Prof. Sukhendu Das
Dept. of Computer Science and Engineering
Indian Institute of Technology, Madras

Lecture - 11

Projection Transformations and Viewing Pipeline

Hello and welcome everybody to the lecture series on computer graphics. Today we are going to discuss about projective transformations and 3D viewing pipeline. So essentially we discussed about 3D viewing and in the last few lectures we had basically discussed 3D transformations and within transformations the last few part of the material on 3D transformation was basically on projective transformation. So as continuity we continue with the 3D viewing pipeline and projection transformations and that is the main subject of the lecture today which is essentially the fifth part of the lecture series.

(Refer Slide Time: 2:02)



Implementation of 3D Viewing

3-D world coordinate output primitives
Apply normalizing transformation
Clip against canonical View Volume
Project onto Projection plane
Transform into viewport in 2D device coordinates for display
2D device coordinates

So if we look into the flow chart of the implementation of 3D viewing you essentially give to the 3D viewing pipeline the set of 3D world co-ordinate output primitives on left hand side using the input here, 3D world co-ordinates output primitives goes through a set of normalizing transformations. This is the essential part of the discussion today which we will see. And the main focus of the discussion today will be on the meaning of normalizing transformations and why it is essential before we apply what is called as clipping that is the second stage, it is clipping against the canonical view volume.

Now these are, I will say larger groups of task that is divided into various categories. First is applying a normalizing transformation and the second is clip against canonical view volume. So this is the new terminology which we will come across today. So we will define immediately what is a canonical view volume and how it helps in 3D viewing and where does it occur in the 3D viewing pipeline so we need to clip. Of course with this there is a separate section and discussion on clipping on both 2D and 3D clipping algorithms.

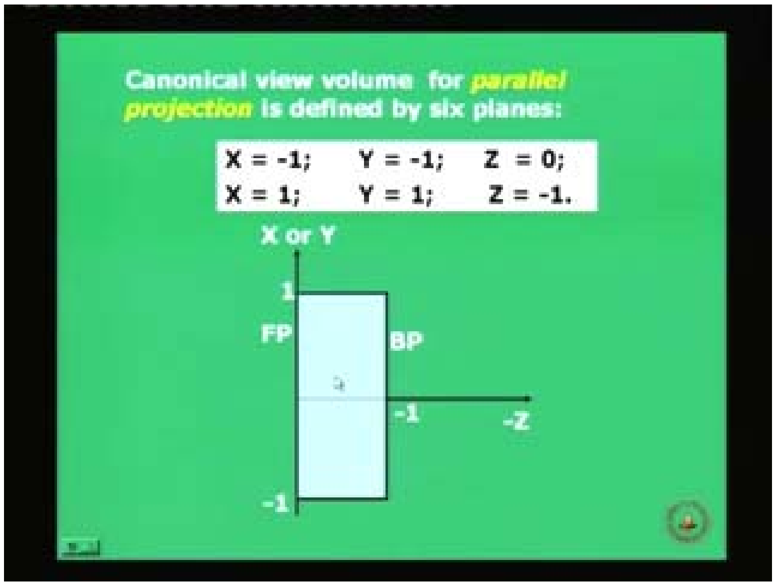
As we discuss that it will be clear how to clip against 2D and closed shape or 3D volume. So for the time being we assume that there is a clipping algorithm which clips against a canonical view volume and after that we project on to the projection plane. Essentially we have discussed this earlier and the part of 3D transformations and I have also discussed about projective transformations in detail. And we also know the general formulation about projective transformations only. So the third task is known. And the last part is the simple transformation into view port, we will also define what a view port is.

We have in fact seen this and will again see what a view port is in 2D device coordinates for display and how the 2D device coordinates are generated. So the input is 3D, world coordinate, output primitives in terms of lines, polygons and circle, text in 3D. And essentially those 3D coordinates are transformed in 2D device coordinates which must be given to the electron gun to fire the corresponding pixels on the screen.

But in between that there are four sub categories and essentially in these four sub categories, number one is, as you see again on the screen, apply normalize transformation, second is clip against canonical view volume, third is projection on to the plane and the fourth off course is transform in to view port. Out of them we have discussed the third category that is projection on to plane. Of course we will see the formulation once again, the formula of the general projection transformation matrix but before that we need to do clipping. So we will understand what a canonical view volume is and before that the major part of the discussion in this particular lecture will be about applying normalizing transformations.

So first we will see what is a canonical view volume. Well there are two types of canonical view volumes which are used in projection geometry in terms of 3D viewing. One of them is the parallel projection canonical view volume and other should be perspective projection canonical view volume. So the canonical view volume for parallel projection is defined by the set of six planes.

(Refer Slide Time: 4:51)


Canonical view volume for *parallel projection* is defined by six planes:

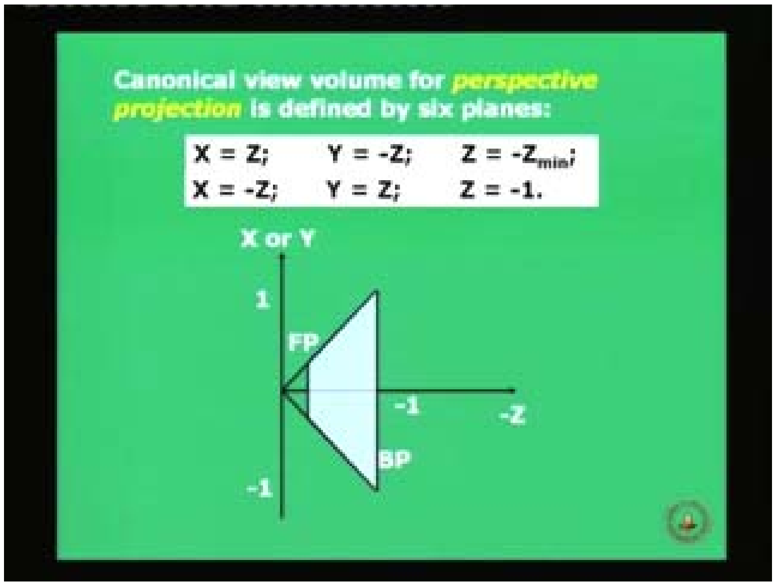
X = -1;	Y = -1;	Z = 0;
X = 1;	Y = 1;	Z = -1.

Essentially you can visualize these to be a rectangular parallel pipette, almost a square not exactly a rectangular parallel pipette where along the x and y dimensions or the x and y axis the difference between the two binding planes are at plus and minus 1 respectively.

So along on x and y the size is 2 and along the z axis the size is 1 because its starts from z equal to 0 and ends at z equal to minus 1. So along this viewing axis it is minus z axis, the canonical view volume is bound by an FP or BP which basically means the front plane and the back plane which is also called as the front clipping plane and the back clipping plane respectively. FP and BP is what we use and FP is nothing but the x y plane z equal to 0 and back plane is nothing but z equal to minus 1. So both planes are parallel to the x y plane and that is the front plane and back plane defined by z equal to 0 and z equal to minus 1 as given on the right hand side of the equation.

And the rest is x equal to minus 1, y equal to minus 1, x equal to 1, y equal to 1. So basically if you think of a square or a rectangle parallel pipette you need six surfaces to bind a volume and the corresponding equations of the six planes are given here by these six simple equations as given on the top. And this is how the side view of the canonical view volume for parallel projection is given. Let us now look in to how this canonical volume takes a shape for a perspective projection.

(Refer Slide Time: 6:43)


Canonical view volume for *perspective projection* is defined by six planes:

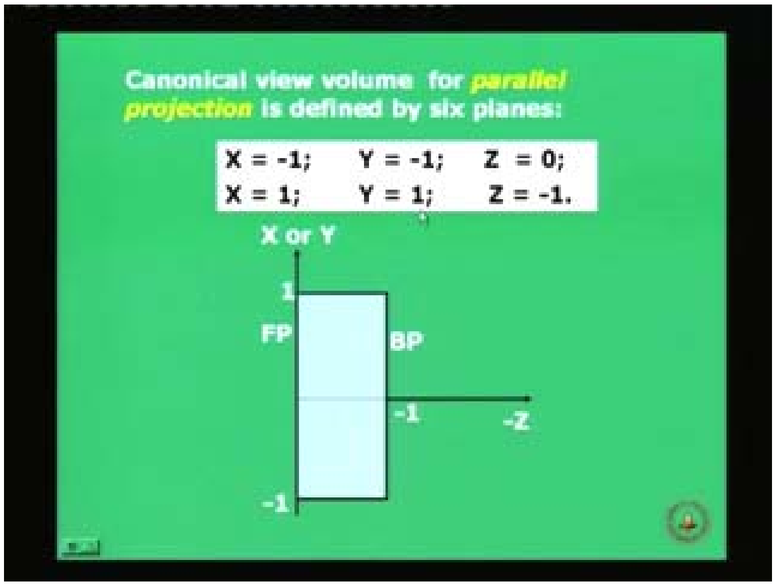
$X = Z$;	$Y = -Z$;	$Z = -Z_{min}$;
$X = -Z$;	$Y = Z$;	$Z = -1$.

Now you can see there is a relation between perspective geometric and perspective projection canonical view volume as well as the parallel projection and the parallel projection canonical view volume. Remember, in parallel projection we have the rays which are parallel and hitting the corresponding projection plane. And in the case of perspective the rays will focus or meet at a particular point so that is why you have a tapering effect of the canonical view volume for perspective projections.

We still have six planes which define them as x equal to z, x equals minus z. So they are inclined plane symmetrical located with respect to the origin or the minus z axis and y equals z and y equals minus z and z. The front plane and back plane is defined by z equals minus 1 which is the back plane. As given in the figure here you see a minus 1 which is this point basically. The z equal to minus 1 equation gives this plane and z equal to minus z mean gives the front plane.

You can actually put the z mean equal to 0 if you like but typically it is a non 0 value which is less than unit. So, that is the front plane and the back plane which we are discussing about for perspective projection. To look into the comparison I will roll back this slide and you can have a look into the comparative figures, this is the canonical view volume for parallel projection and this is the canonical view volume for perspective projection.

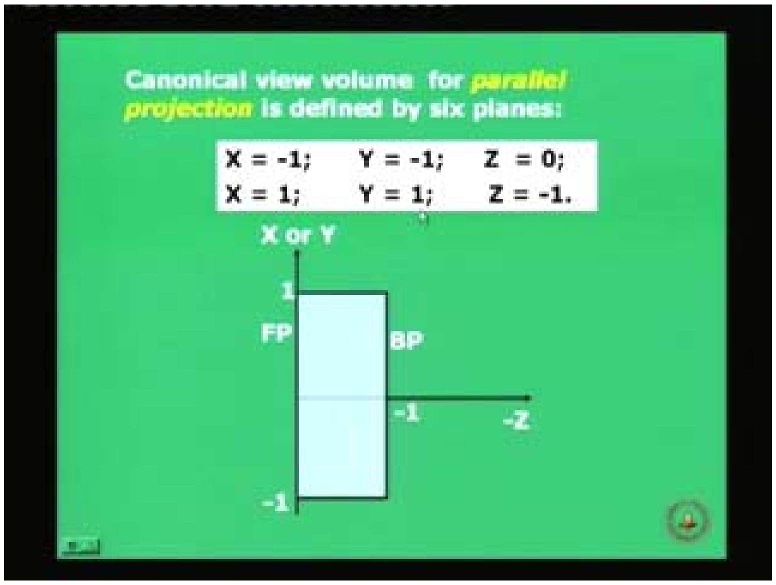
(Refer Slide Time: 7:56)



Canonical view volume for *parallel projection* is defined by six planes:

X = -1;	Y = -1;	Z = 0;
X = 1;	Y = 1;	Z = -1.

(Refer Slide Time: 8:02)

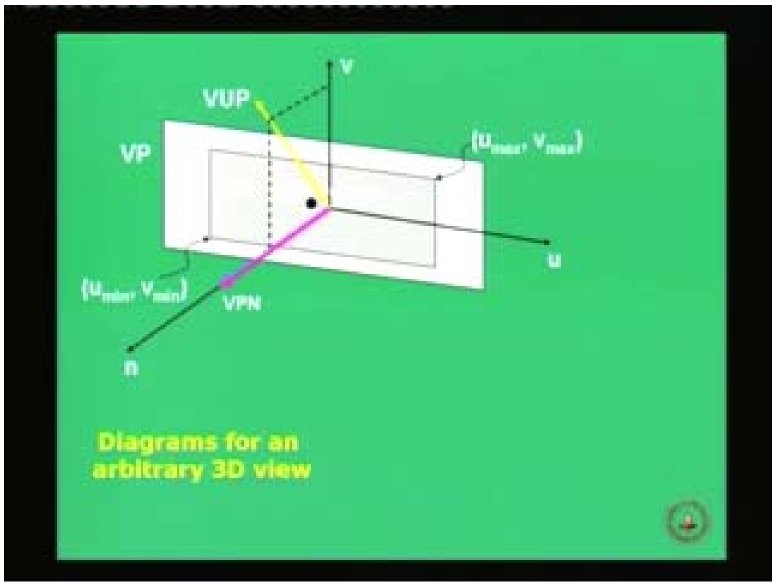


Canonical view volume for *parallel projection* is defined by six planes:

X = -1;	Y = -1;	Z = 0;
X = 1;	Y = 1;	Z = -1.

You see a sort of rectangular parallelepiped structure because the rays are parallel here and the rays are tapered off and they meet at a focal point. So for the x y you have a sort of a pyramidal structure but for a canonical view volume for perspective projection we only see the side view. You have to visualize the six planes which bind this particular view volume, so that is the idea. And we will see or discuss later on about clipping in different sections and how to clip a certain line segment, basically line segments or even planes with respect to this canonical volume when we discuss about clipping but not in this lecture and we move on because now we actually have to capture this entire world in to a canonical view volume before they project it into the projection plane in the third step. So we will look back into this flow chart.

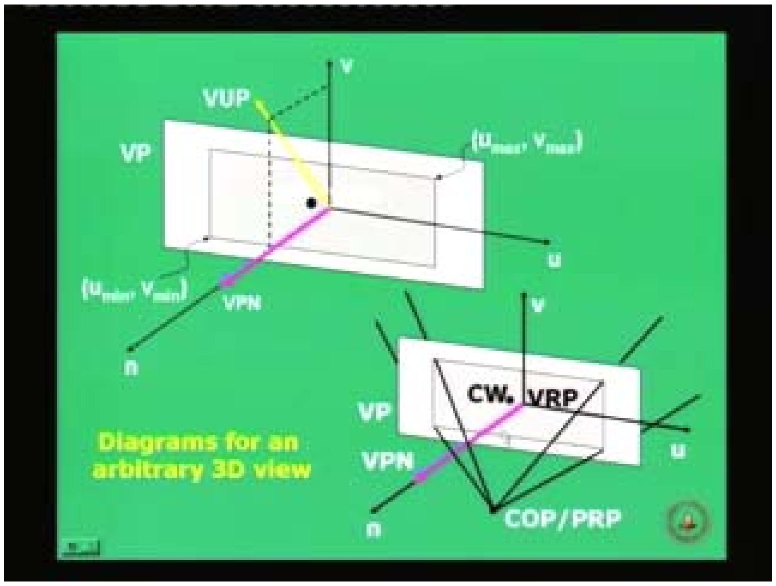
(Refer Slide Time: 8:50)



And typically this is the diagram for an arbitrary 3D view and I did cover this chapter under projective 3D transformations and a projective geometry and I hope you remember all the terminologies which I will repeat once again.

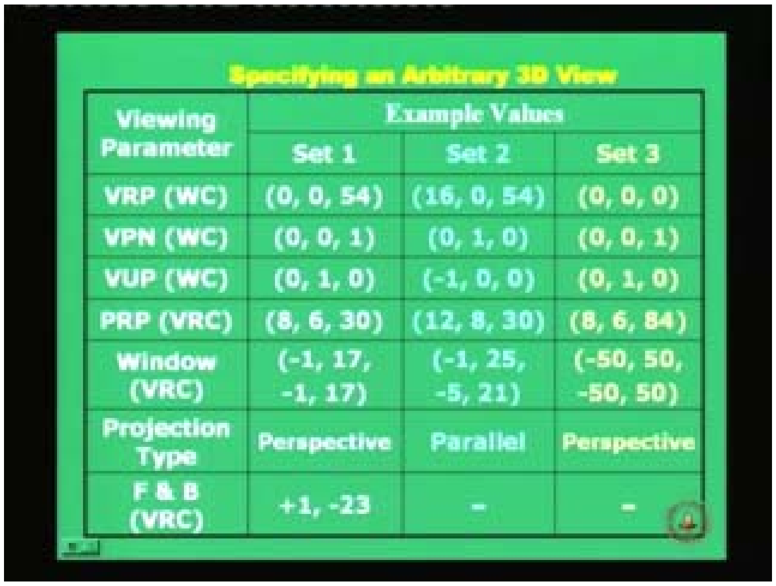
The term VP means the view plane, VUP means by view up vector which is given by the yellow line or arrow representing the view up vector, CW the center of the window of the view port which is defined by the coordinates U min V min to U max and V max respectively. The grey color, the light blue color outside is the view plane, VRP is the view reference point which may not be the center of the window but VRP is the reference coordinate on the projection plane with respect to which you have the U and V axis, the 2D coordinates axis on the view port or the projection plane defined and of course the n of the VPN which is nothing but the surface normal. The surface normal to the projection plane we have an VPN along the normal n to the surface and so I hope you remember and collect the terminologies based on which we also defined the projection geometry and the projective equations for both perspective that is on orthographic or parallel projection. We look in to the next diagram, this is also a diagram for perspective projection geometry and the VP and VPN are same that is a view plane and the view plane normal.

(Refer Slide Time: 9:40)



We have the center of the window and we have a view reference point and in addition we have a center of projection or perspective reference projection respectively or center of projection basically or PRP as it is called that is the projective reference point. PRP is projective reference point or COP as discussed about where the rays meet. And basically for the parallel projection the COP goes to infinity and we can have a PRP defined even for perspective geometry but COP basically goes to infinity because the rays have to be parallel to the projection plane. In this case, for the perspective case they meet at the center. So I hope you remember these two diagrams which we also covered in projective geometry earlier.

(Refer Slide Time: 11:03)



Specifying an Arbitrary 3D View

Viewing Parameter	Example Values		
	Set 1	Set 2	Set 3
VRP (WC)	(0, 0, 54)	(16, 0, 54)	(0, 0, 0)
VPN (WC)	(0, 0, 1)	(0, 1, 0)	(0, 0, 1)
VUP (WC)	(0, 1, 0)	(-1, 0, 0)	(0, 1, 0)
PRP (VRC)	(8, 6, 30)	(12, 8, 30)	(8, 6, 84)
Window (VRC)	(-1, 17, -1, 17)	(-1, 25, -5, 21)	(-50, 50, -50, 50)
Projection Type	Perspective	Parallel	Perspective
F & B (VRC)	+1, -23	-	-

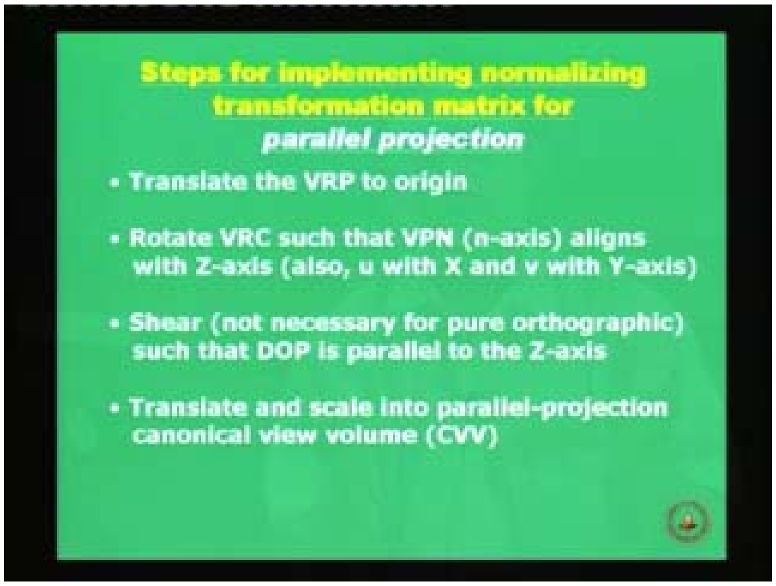
Typically in a general framework of computer graphics you need to specify an arbitrary 3D view. Of course you must ensure that when we specify an arbitrary 3D view the camera is looking towards an object and the object is within the view zone of the camera and things like that.

But let us take a typical example or a few examples of the viewing parameters, just a set of few example values to give you a rough idea about how to set these parameters. And then of course we will see how these parameters are used for normalizing transformations and also for projection geometry. The viewing parameters used are VRP which is view reference point, VPN view plane normal, view up vector or projective reference point or COP or PRP and the corresponding window coordinates of the view port.

The view reference coordinates in bracket means VRP VPN VUP PRP, the first three are in world coordinates, WC VRC is view reference coordinates, projection types could be perspective or parallel and you can also have the front plane and the back plane coordinates defined. These are optional with respect to VRC and the values are given for the set one, you may not defined mentioned job to set one. And set two does not contain the values for the front clipping plane and back clipping plane with respect to the view reference coordinates systems.

So if we look into set one these are typical example values which you can specify for the arbitrary 3D view and you can make a function called with this set of parameters and we will set up a specify arbitrary 3D view for the object for you. So that is just a typical example when you use the graphical software and for the corresponding fixed standards you have to use this sort of conventions and these are just the example values which you can use. Well we go into the steps for implementing normalizing transformations matrix in the case of parallel projection first and then we look in to the perspective projection and the set of steps are defined in the slide.

(Refer Slide Time: 14:40)



Steps for implementing normalizing transformation matrix for *parallel projection*

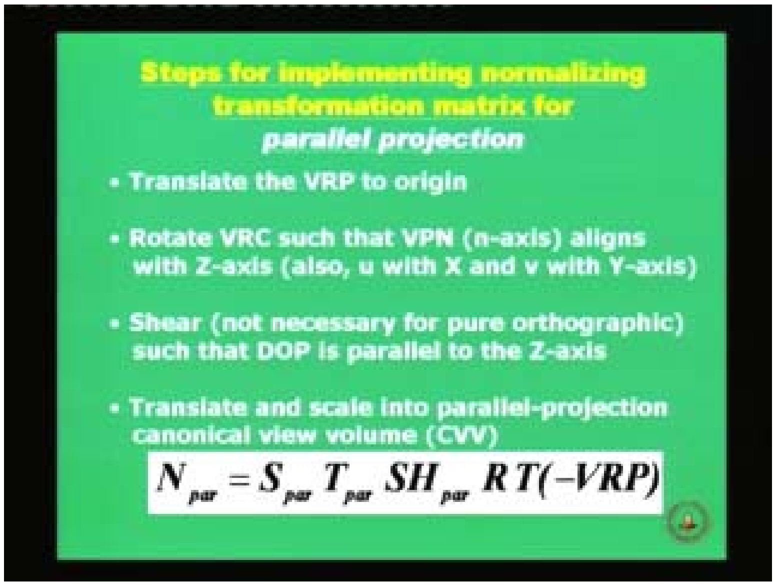
- Translate the VRP to origin
- Rotate VRC such that VPN (n-axis) aligns with Z-axis (also, u with X and v with Y-axis)
- Shear (not necessary for pure orthographic) such that DOP is parallel to the Z-axis
- Translate and scale into parallel-projection canonical view volume (CVV)

The first step is you need to transform the VRP or the view reference point to the origin. Translate the VRP to the origin so you need to have a translation matrix first and then you have a rotation matrix which helps you to rotate the view reference coordinates such that the n axis of the VPN of the view plane normal axis here aligns with z-axis of the world coordinate system and in turn also the u with the x and v with the y. So u aligns with x and v with y and the VPN aligns with z. I hope you remember the VPN we just discussed about what it means of the figure to be will describe as to what is meant by translating or at least rotating this, because the second step of rotation is a most crucial step in the process of implementation and the rest are easy steps but I will complete the sequence of steps.

The third step is basically sheared which is not necessary for pure orthographic projection general necessary for parallel. We know orthographic is the special case of parallel projection and in the case of non-orthographic parallel where the direction of projection or DOP is not parallel to the z-axis, we need to make it parallel to z-axis before we actually project it. So you may need a shear in case of parallel projection which are non-orthographic images and that is the third step. But remember it is non-optional but will have a sheared matrix in the third step.

In the fourth step you have translate and scale into parallel projection canonical view volume. We know the planes which define the canonical view volume and we need to provide a translation and scale to adjust your view volume to the canonical view volume. This is also necessary in the view pipeline before you actually use the projection matrix and in fact before you use the clipping itself and this is the form. So these are the sequence of steps and you see that if there are five different steps it results in a composite matrix with five different matrix multiplication in the particular order and since the first step is translate the VRP to origin that is the T with a minus VRP parameter, then you have the R which we are going to discuss in detail today.

(Refer Slide Time: 15:35)



Steps for implementing normalizing transformation matrix for *parallel projection*

- Translate the VRP to origin
- Rotate VRC such that VPN (n-axis) aligns with Z-axis (also, u with X and v with Y-axis)
- Shear (not necessary for pure orthographic) such that DOP is parallel to the Z-axis
- Translate and scale into parallel-projection canonical view volume (CVV)

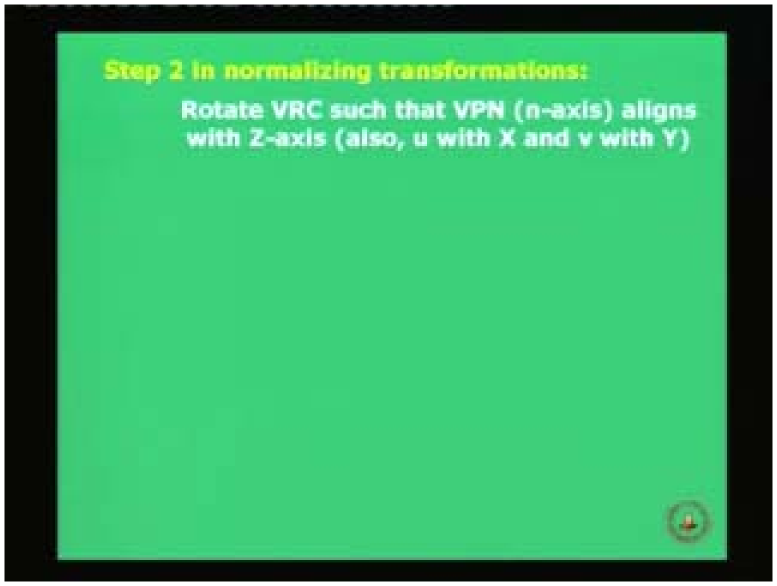
$$N_{par} = S_{par}\, T_{par}\, SH_{par}\, R\, T(-VRP)$$

And then SH matrix parallel means the shear in case of parallel projection and then T and S successibly for parallel projection respectively. This S and T correspond to the last step fourth and fifth respectively which is translate and scale into parallel projection canonical view volume. So this is the sequence of operations one is to do of the first stage itself of the 3D viewing pipeline. If you remember the flow diagram which we discussed about in the beginning of the lecture today, there were four stages; the first stage was normalizing transformation, second was clipping, third was projection and fourth was simple 2D in view port.

So the first one is what we are discussing now and these are the five steps of the first stage itself, the normalizing transformation. So you need to normalize as that the entire world, what this stage does is, the entire world is brought in to the canonical view volume so that you can apply clipping easily and you can also apply the perspective projection transformation matrix. So, perspective projection transformation matrix, although we know the generalize form before we can apply two sub stages prior to that, one is clipping which we will of course discuss later on. So do not worry about it for the time being, I will keep repeating it, do not worry about clipping, we will do that with respect to the canonical view volume when we exclusively discuss clipping in 2D and 3D.

We will mainly discuss the first stage which is the normalizing transformation. And that has five stages as given here and the composition matrix N par where N par is nothing but the overall normalizing transformation matrix for parallel projections. I repeat it is composed of sequence of five different operations of five transformations. First translate T then rotate R shear SH then a translation T and again as scaling S par. Out of these the most significant translations will be very easy as you can visualize. We will first discuss R and then of course spend a little bit time about the shear. So rotation is the one which is very crucial and we will see why. This is the step two of normalizing transformation,

(Refer Slide Time: 17:14)

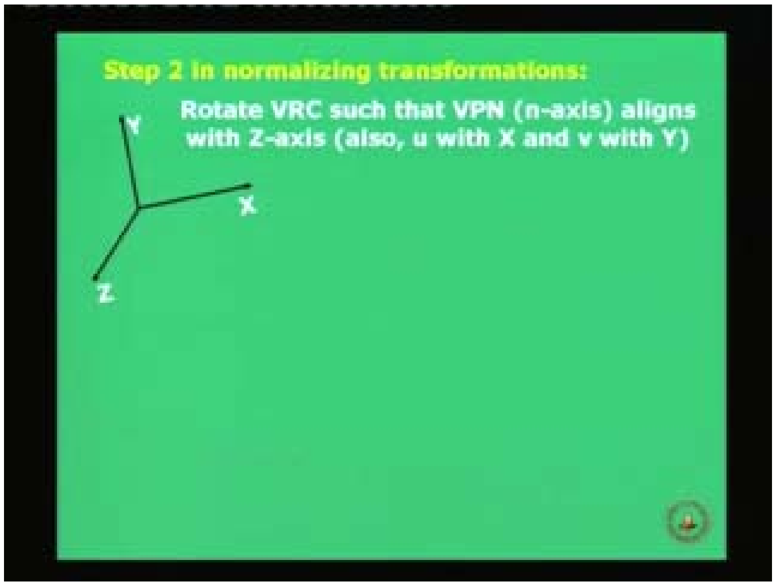


Step 2 in normalizing transformations:

Rotate VRC such that VPN (n-axis) aligns with Z-axis (also, u with X and v with Y)

I repeat that state again, is you have to rotate the view reference coordinate system, the view reference coordinate system which is defined with respect to view port and camera and things like that. So the camera will have its own u v and the n axis, you have to rotate that, of course translate the VRC to the origin and then rotate in such a manner such that it gets aligned with the world coordinates just as the VPN or the n axis or the view plane normal aligns with the z axis of the world system and the u and the v also aligns with the x and y axis respectively. So this is the most difficult task in normalized transformations.

(Refer Slide Time: 17:56)

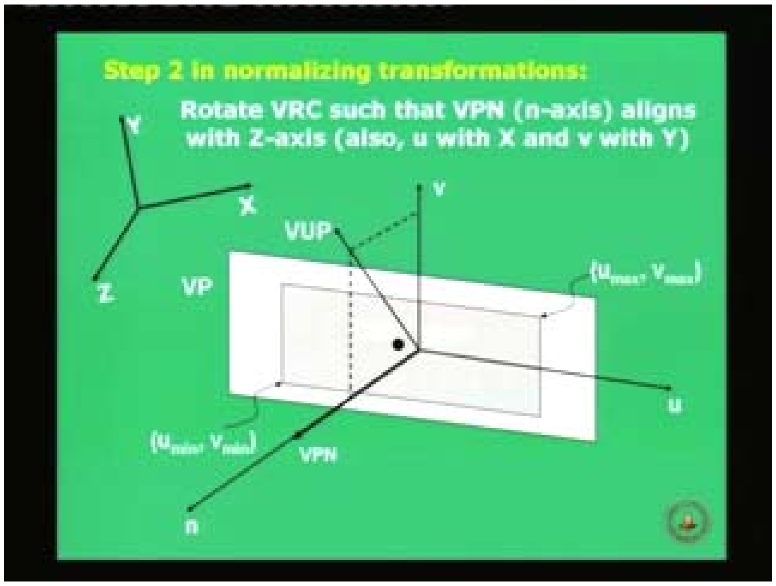


Step 2 in normalizing transformations:

Rotate VRC such that VPN (n-axis) aligns with Z-axis (also, u with X and v with Y)

And just to give an example what I mean by this, let us see this xyz or xyz is the world coordinate system. That means originally you have defined the world coordinate system with respect to this xyz as origin and however the camera is in such a position that it may not be at the origin of the system it could be at some arbitrary position looking down at some other direction and there is where you have the n and u and the v axis which defined the 3D coordinate system for the camera let us say because basically through the camera you are looking into the virtual world which gives the visualization.

(Refer Slide Time: 18:12)



Step 2 in normalizing transformations:

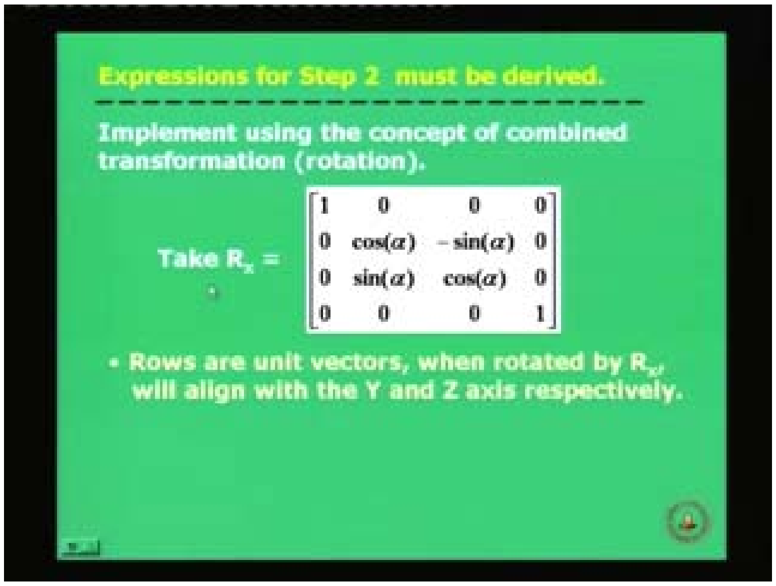
Rotate VRC such that VPN (n-axis) aligns with Z-axis (also, u with X and v with Y)

The camera will have its own view plane normal of the n axis and it will have its own x and y which we call it as u and v and this diagram as we already know where is the view plane, what is the view up vector, what is the VPN, center of the window CW, view reference point and all that. So the first stage is to basically take this diagram or with respect to the camera or the view system and first give a translation that the VRP goes to the origin and then give a composite rotation such that the VPN has to align with z, u with x and v with y. I repeat again, first the VRP has to be taken to the origin, here it will go, I repeat VRP is here, it has to go to the origin at this point then the VPN of the n axis should align with the z axis here.

I hope you can see the cursor, then the u axis should align with x and the v with y. So these three must simultaneously happen together with the composite rotation on three dimensional rotation matrix. Three dimensional rotation must takes place to have this alignment done. So this is the most crucial stage or step in normalizing transformations and this requires a sequence of operations.

Now, how to get this R, how to get this R? You will probably think of the divided steps of 3D rotation and transformations which take place but we dig into the depth of a little bit of mathematics here to easily understand about 3D rotations, get some principles out of the matrix property of rotation and see how easily we can formulate this R. And then we go to the next slide. So we are now essentially under the first category or the first stage of normalizing transformations. Within the five steps of the first stage we are in the second step. The first stage of the four stages is normalizing transformations, in that first stage we have five steps and in that five steps first is translation and second is rotation. We are talking of step two and that is what is given in the slide.

(Refer Slide Time: 20:24)



Expressions for Step 2 must be derived.

Implement using the concept of combined transformation (rotation).

$$\text{Take } R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) & 0 \\ 0 & \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Rows are unit vectors, when rotated by R_x, will align with the Y and Z axis respectively.

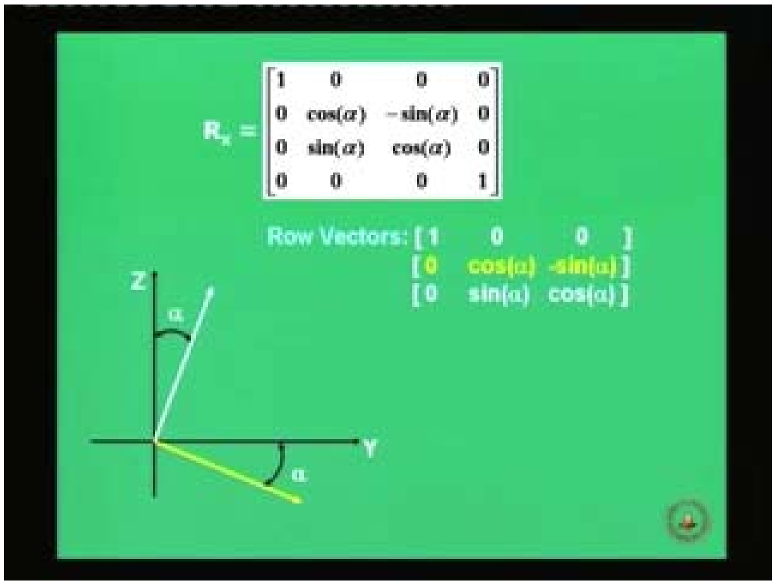
So we are looking for expressions for step two which must be derived and that is what we are looking at and then we implement that using the concept of what we call as combined transformation or rotation. Let us take a rotation matrix or x, this rotation matrix you know what it does, it takes the point in the view space and rotates about the x axis. So the x axis coordinates of that point will not be disturbed, the x axis pointing towards you; you rotate it about the x axis. So the x axis coordinates of the point does not get disturbed but what are the two other coordinates we change? y and z, y and z are the two coordinates we change and that is why you see the parameters cos alpha sin alpha, of course we were talking about looking into the axis and counter clockwise rotation, positive alpha, the second and third axis cosine alpha and sin alpha terms causing the change in y and z coordinates but the x axis will not change. So let us try to visualize this rotation from a slightly different angle.

I make a statement here and we will validate that with a figure and illustration and later on that if we take the rows of this matrix Rx as unit vector, that is what is written. If rows are unit vectors, you remember it is very easy to see here that due to the orthogonal property of this Rx matrix the rows and the columns are unit vectors. So take the rows as unit vectors and when these unit vectors are rotated by this rotation matrix Rx they will align with the y and z axis respectively. Well straight away it is difficult to visualize the statement but I will explain that soon with the help of a figure but just note down for now that the rows are unit vectors. If we take the rows, I will take these rows of the matrix, each row is a unit vector, we can verify that, take the first three terms of each row, you do not have take the fourth term because the fourth term or the 4/4 matrix is coming due to the homogenous coordinate system. So just take the top left 3/3 sub matrix and take the three elements of each row, the three dimensional vector we are talking about. So each of these rows are unit vectors and these unit vectors I say that when they rotate it by Rx the x coordinate does not change.

So 1 0 0 the first row vector remains 1 0 0 itself but the next two row vectors which are again unit vectors they will get aligned with y and z axis respectively. So it is a very nice phenomenon which we will discover now with an example. But before that there is another complimentary or similar concept where I say, what about the column vectors? If I say that the rows are unit vectors and if they are rotated they will align with y and z respectively. Then the second point says that if I take the unit vectors along the principle axis of the coordinate system and those unit vectors are rotated by Rx they form the corresponding column vectors as given here in this matrix.

So if take the column vectors 1 0 0 0 0 cos alpha sin alpha 0 minus sin alpha cos alpha you can form these vectors by taking unit vectors along the principle axis and then rotating by Rx. Take the principle axis vectors, unit vectors, rotate them and you get the column vectors. And the other case, if you take the row vectors and then rotate by Rx they will get aligned with the principle axis y and z. Very interesting phenomenon here and we will see with an example figure as to how this happens. Again I repeat, this is again the row matrix Rx as given the top and what are those row vectors, that I have given here and the row vectors of the first row I get 1 0 0, that is the first row vector 1 0 0 as given here, the second row vector or take the first three terms upon left sub matrix 0 cos alpha minus sin alpha is the second row. The third row is 0 sin alpha cos alpha.

(Refer Slide Time: 23:55)



$$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) & 0 \\ 0 & \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

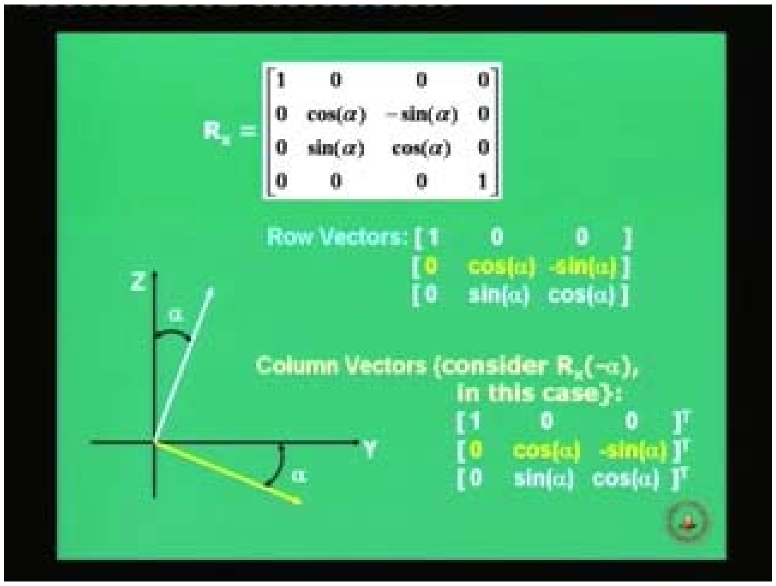
Row Vectors: $[1 \quad 0 \quad 0]$
$[0 \quad \cos(\alpha) \quad -\sin(\alpha)]$
$[0 \quad \sin(\alpha) \quad \cos(\alpha)]$

These are the three row vectors in 3D. How will these row vectors look like? Let us take a three dimensional coordinate system as given in this figure on the left hand side of the slide with you and the x axis is not shown because it is coming out from the slide and pointing towards you. You can take a right hand coordinate system and assume x to be pointing towards you. So I have not drawn that because they are essentially rotating around the x axis, so it will rotate about the origin in 2D and x axis pointing towards you, to the right you have y and the vertical one is z. So look back in to the figure and you will find x axis pointing towards you, y to the right and z axis vertically upwards.

So now look back in to the row vectors which I have just written. Look at to the row vectors and try to map it on to the system. You will find that the 1 0 0 vector is nothing but it is along the x axis pointing towards you. Anyway since it is rotating by Rx it does not matter, 1 0 0 nothing will happen. What about the second vector 0 cos alpha minus sin alpha, you look at this, I have labelled with the same color as the row vector in text and this is the corresponding yellow vector which is giving the second vector because if you look this the angle alpha then the X coordinate and if it is a unit vector then the y coordinate will be cos alpha and the z coordinate of the vector will be sin alpha in the negative direction hence we have minus sin alpha so that is the second vector.

For the third vector also I have used the corresponding color; it is sin alpha along the y and z alpha along the z axis. So these are the two vectors which are the second and third row vectors of the transformation matrix Rx. Now what will happen if I take these two vectors and give it a counter clockwise positive rotation alpha. You can visualize what will happen if I give it a rotation by an amount of alpha, what will happen? These corresponding row vectors will automatically get aligned with the y and z axis. You have them bent, they will turn and they will get aligned with the y and z axis respectively. I discussed this as a point in the previous slide. I will show that point once again where you take the row vectors as given in the transformation matrix and those row vectors when given a corresponding rotation by the same matrix they will get aligned with y and z. Now what with the column vectors? I can view the same diagram and visualize the column vectors.

(Refer Slide Time: 26:48)



$$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) & 0 \\ 0 & \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

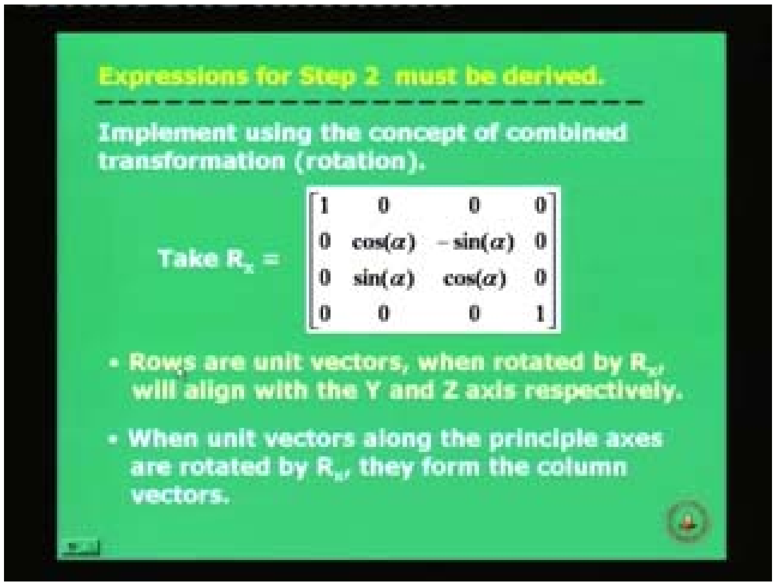
Row Vectors: $[1 \quad 0 \quad 0]$
$[0 \quad \cos(\alpha) \quad -\sin(\alpha)]$
$[0 \quad \sin(\alpha) \quad \cos(\alpha)]$

Column Vectors (consider $R_x(-\alpha)$, in this case):
$[1 \quad 0 \quad 0]^T$
$[0 \quad \cos(\alpha) \quad -\sin(\alpha)]^T$
$[0 \quad \sin(\alpha) \quad \cos(\alpha)]^T$

Now for the column vectors what you assume is let us take a column vector, you take unit vector along y and z respectively and give it a negative rotation that is now the clockwise looking into this figure negative rotation of R_x of minus alpha. In this case what will typically happen is you will get the corresponding column vectors as 1 0 0 0 cos alpha minus sin alpha. Remember R_x of minus alpha, so what will happen is in this matrix the sin alpha here will change sign to minus and in the third column the minus will become plus. So that is what is happening here, the reverse could also take place, you have to given a positive alpha and seen what could have been the result.

But in this case I can show you that with the negative alpha and the same figure if you take unit vectors along y and z and give it a clockwise rotation you will get the column vectors exactly as given by the transformation matrix. So this is an interesting phenomenon which we will use to derive the general transformation matrix in which you have an arbitrary coordinate system and you need to give a composite direction to get it all aligned with another coordinate system respectively. I will roll back to those two points once again.

(Refer Slide Time: 28:00)



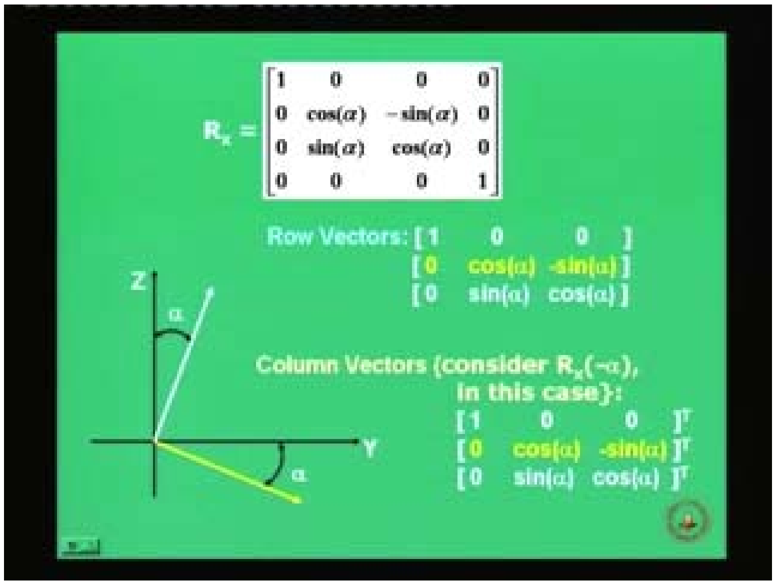
Expressions for Step 2 must be derived.

Implement using the concept of combined transformation (rotation).

Take $R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) & 0 \\ 0 & \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

- Rows are unit vectors, when rotated by R_x, will align with the Y and Z axis respectively.
- When unit vectors along the principle axes are rotated by R_x, they form the column vectors.

(Refer Slide Time: 28:27)



$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) & 0 \\ 0 & \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

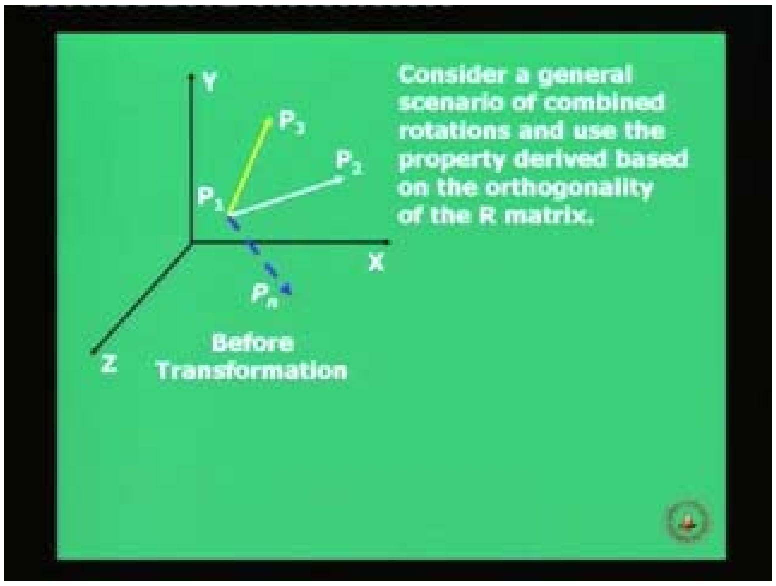
Row Vectors: $[1 \quad 0 \quad 0]$
$[0 \quad \cos(\alpha) \quad -\sin(\alpha)]$
$[0 \quad \sin(\alpha) \quad \cos(\alpha)]$

Column Vectors (consider $R_x(-\alpha)$, in this case):
$[1 \quad 0 \quad 0]^T$
$[0 \quad \cos(\alpha) \quad -\sin(\alpha)]^T$
$[0 \quad \sin(\alpha) \quad \cos(\alpha)]^T$

You see here these are the two points where I say, I repeat again, if rows are unit vectors and then they are rotated by a rotation matrix R_x, remember then they will get aligned with the y and z axis respectively and we have seen that. These rows of unit vectors obtain itself from the R_x. And the next point is when the unit vectors along the principle axis are rotated by a transformation matrix R_x they form the corresponding column vectors of R_x. So that is a very interesting phenomenon and we have seen that with the help of this example.

You can copy this figure and the corresponding row vectors, column vectors and keep checking it yourself with both positive alpha and minus alpha. Rotation of course you can use R_x R_y or R_z you will also get the same effect. So keeping this concept in mind that the row vectors can be rotated and formed and they form the vectors as given in the rotation matrix and the column vectors you take, in fact the row vectors get aligned with the y and z and in the case of columns you take unit vectors along y and z and what happens is they form the vectors as given by the R_x. So, if that is true and if it is possible to use this concept and this idea because of the property of orthogonality of the matrix, because unless the matrix would have been orthogonal you would not have had the rows and columns giving you unit vectors, that is number one.

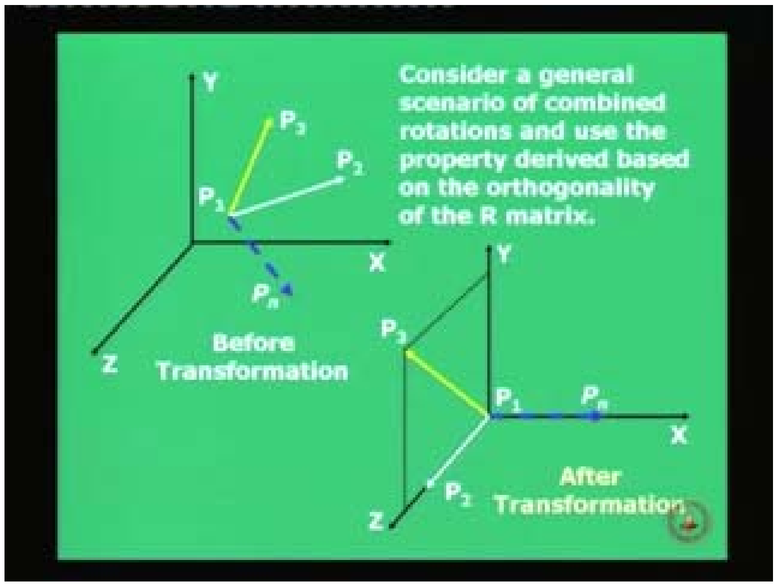
(Refer Slide Time: 29:36)


Consider a general scenario of combined rotations and use the property derived based on the orthogonality of the R matrix.

And so if that is the case I make a step jump and come to a general scenario of combined rotations where if I use this property based on the orthogonality of the R matrix where I say that if this is a scenario of three different vectors on a system xyz, I have the yellow vector P_1P_3, the white is vector given by the white arrow P_1P_2 and the dashed vector in blue color P_1 P_n. So these are the three vectors and in these three vectors I must tell you that the P_1 and P_n is nothing but a vector which is normal to the surface obtained by P_1 P_2 P_3. So actually what it means you can cross product of the two vectors P_1P_2 and P1 P_3 that will give the normal vector P_n that is the relation.

So basically P_1P_n is normal to both P_1P_2 as well as P_1P_3 whereas P_1P_3 and P_1P_2 may not be orthogonal to each other or perpendicular to each other. So P_n can be visualized again as normal to a surface formed by P_1P_3 and P_1P_2 vectors. But what I essentially want to do is I want to get this a set of vectors aligned with, at least two of them I can get them aligned with two of the other axis in xyz axis because I have perpendicularity with respect to P_n and P_2. P_1P_n is perpendicular to P_1P_2. So before transformation this was the scenario, I would like to give this set of three vectors a combined transformation where these three get aligned with the homogeneous. If this the xyz and I have this P_1P_2 P_3P_n vectors and I will give it a combined transformation to get itself aligned with xyz.

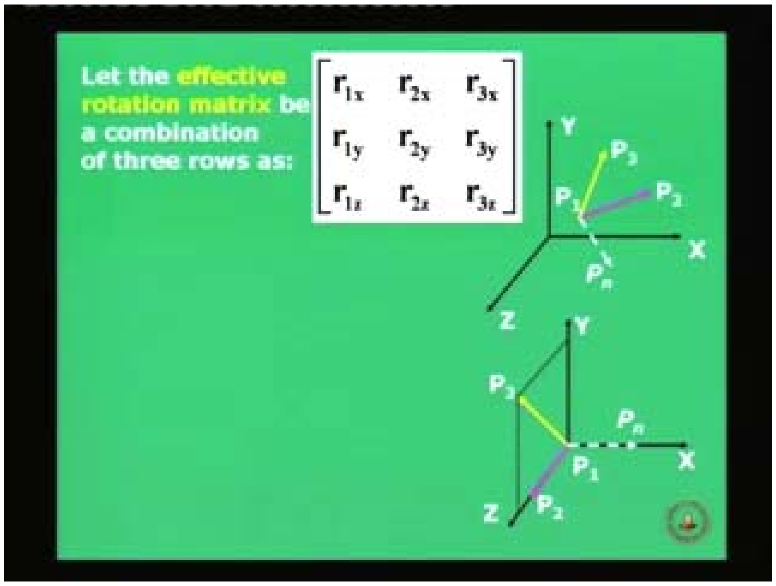
(Refer Slide Time: 31:23)


Consider a general scenario of combined rotations and use the property derived based on the orthogonality of the R matrix.

And now we go back into the slide and I want to give it a combined transformation and after the transformation I want the following to takes place. The P_1P_2 vector before transformation was given in the previous figure here in the same slide and it gets aligned with the z axis and corresponding normal to that is the P_1P_n the blue colored dashed one gets aligned with the x axis and of course P_1P_3 also takes its own place somewhere because it has to be perpendicular also to the P_1P_n. So basically these P_1 P_2 P_3 vectors, these two vectors will lie in the YZ plane and that is bound to happen with this set of transformation.

So, if you remember the previous two concepts in which we talked about the orthogonally matrix in terms of rows and columns, which one was the property when given a rotation arbitrary vectors or some vectors gets aligned with the coordinate axis xyz? We have to deal with rows because if we take the rows as unit vectors of the transformation matrix and give it a transformation they in fact get aligned with the corresponding principle axis. So that relation will keep in mind and the corresponding transformation rows or columns of the transformation matrix in fact can now be defined or obtained directly from these vectors. If we just expand this concept from 2D which was a rotation on x axis to a general scenario, just a step more, of course we can derive an analytical expression and check this also but if you just draw an analogy to a general three dimensional scenario what will happen is now that you can derive this transformation matrix from the vectors directly. That is the advantage, remember the previous examples we were taking these rows and columns with the transformation matrix R_x. Keep that in mind and we will derive now.

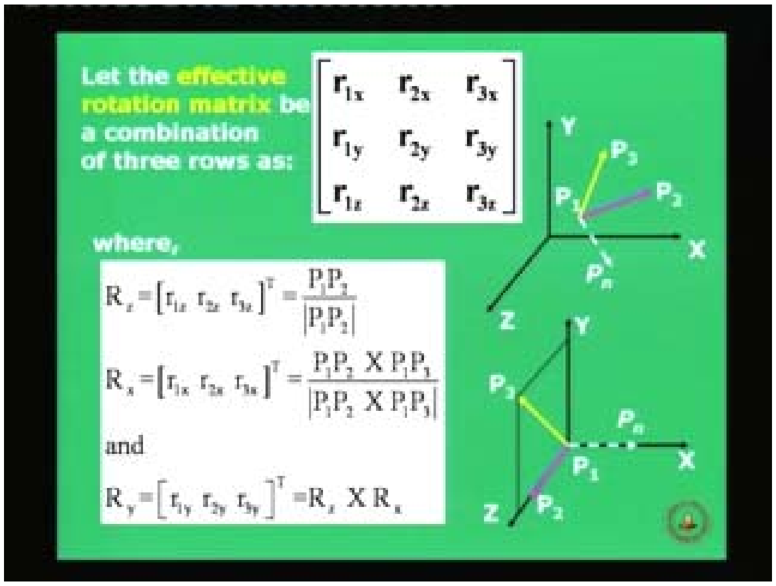
(Refer Slide Time: 33:21)


Let the effective rotation matrix be a combination of three rows as:

$$\begin{bmatrix} r_{1x} & r_{2x} & r_{3x} \\ r_{1y} & r_{2y} & r_{3y} \\ r_{1z} & r_{2z} & r_{3z} \end{bmatrix}$$

Let the effective rotation matrix be a combination of three rows as given by these where I will the first row corresponds to something in x such as r_{1x} r_{2x} r_{3x} respectively, second with respect to similarly with y and similarly the third row with respect to z. And this is the effective rotation matrix which will cause the transformation to take place and if that is so this rotation matrix, what were you supposed to do before transformation? This P_1 P_3 P_1 P_2 P_1 $_{Pn}$ the three vectors will be transformed in such a manner that the P_1 P_2, the purple color vector will get aligned with z axis the whitish dashed vector P_1 P_n will align with the x axis and of course the P_3 vector will also align somewhere in the YZ plane.

(Refer Slide Time: 34:11)


Let the effective rotation matrix be a combination of three rows as:

$$\begin{bmatrix} r_{1x} & r_{2x} & r_{3x} \\ r_{1y} & r_{2y} & r_{3y} \\ r_{1z} & r_{2z} & r_{3z} \end{bmatrix}$$

where,

$$R_z = \left[r_{1z}\ r_{2z}\ r_{3z}\right]^T = \frac{P_1P_2}{|P_1P_2|}$$

$$R_x = \left[r_{1x}\ r_{2x}\ r_{3x}\right]^T = \frac{P_1P_2\ X\ P_1P_3}{|P_1P_2\ X\ P_1P_3|}$$

and

$$R_y = \left[r_{1y}\ r_{2y}\ r_{3y}\right]^T = R_z\ X\ R_x$$

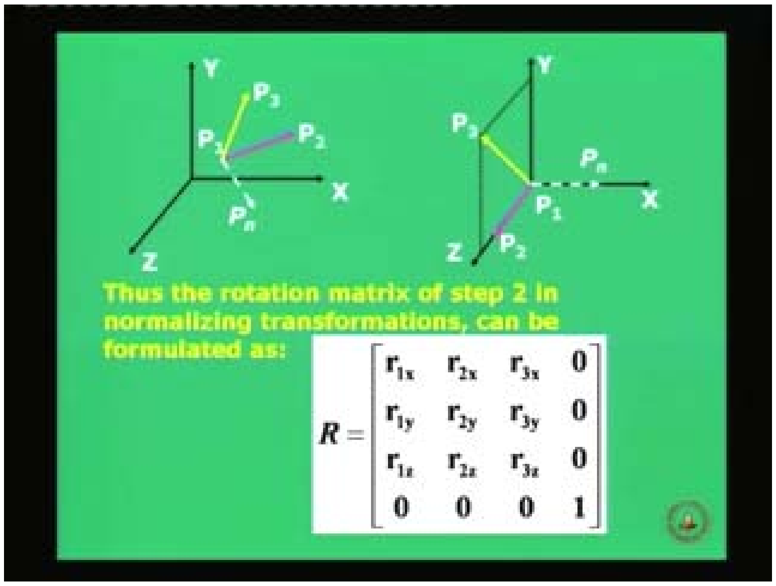
To implement this concept we look back into the equations and here the corresponding form of the coefficients of these combined or effective rotation matrix is given. And we start with the bottom row, the last row of that matrix where I defined R_z to be vector of the last three coefficients of the last row elements of the matrix given by what it is nothing but the unit vector along P_1P_2. So if I put the elements to the last row equal to the unit vector along P_1P_2 that will help me to get this vector align with the z axis after transformations. Similar to that what I have to do next? I have to take this P_1P_n vector and align with the x axis. So my first row which consists of elements r_{1x} r_{2x} and r_{3x} respectively which is given by the vector R_x is nothing but the vector P_1P_n.

And what is P_1P_n vector? It is nothing but a cross product of P_1P_2 and P_1P_3. Unit vector along P_1P_n can be obtained by the cross product of P_1P_2 and P_1P_3. So that is the unit vector which is given by P_1P_2 cross P_1P_3 cross product divided by the norm correspondingly will give the unit vector along P_1P_n and that gives the elements r_{1x} r_{2x} r_{3x} respectively. So the top row and the last row elements are coefficients of the matrix are already obtained, you need the middle row which is nothing but the r_{1y} r_{2y} r_{3y} respectively.

Easily you can visualize that they have to be nothing but a cross product of the two vectors P_1P_2 and P_1P_n, and nothing but basically R_z into R_x that will give you the vector which will align automatically with the y axis, because after you have aligned P_1P_2 with z, P_1P_n with x automatically the cross product of these two will get aligned with the Y axis respectively. So the cross product of R_z and R_x will be elements of the Y axis. So this is the philosophy or concept by which you come up with an effective rotation matrix by which any arbitrary three dimensional coordinate systems can get aligned with another coordinate system respectively.

Of course we are talking of Cartesian coordinate system XYZ and nuv let us say, and you want to put this here or you want to put that here sort of the thing, you can do that simply if you know the corresponding vectors, unit vectors of the other axis with respect to the one which you are mapping on to or aligning it with. And if you know the source and the target of these vectors with respect to this coordinate system, then you just form the transformation matrix with the help of these vectors, you put the coefficients as these unit vectors apply that and the system gets aligned. So that is the basic philosophy of the rotation matrix which is essential for implementing step two of the transformation of normalized transformation. So understand the theory now and I repeat this figure once again.

(Refer Slide Time: 36:03)

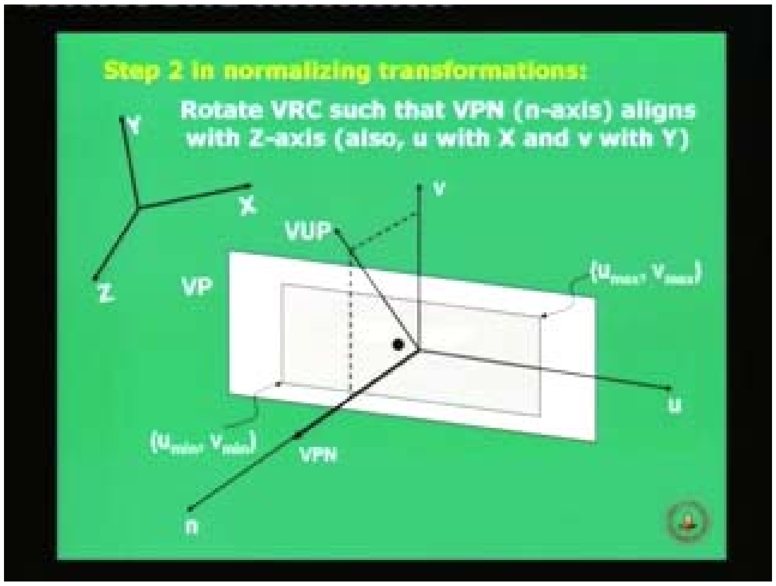


Thus the rotation matrix of step 2 in normalizing transformations, can be formulated as:

$$R = \begin{bmatrix} r_{1x} & r_{2x} & r_{3x} & 0 \\ r_{1y} & r_{2y} & r_{3y} & 0 \\ r_{1z} & r_{2z} & r_{3z} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

This is essentially showing figure in which the rotation matrix of step two has to be formulated in a manner in which we illustrated with respect to this figure where I made P_1P_2 align with z axis and P_1P_n align with the x axis respectively. And if I can do that similarly I can do it for my normalizing transformation where I need to take the VRC at the view reference coordinates system and align it with the world coordinate system. And I will have the similar form of the rotation matrix and I remember I repeat the step two and this is what we have been discussing about how to implement step two in normalizing transformation and it says that you rotate the VRC such the view plane normal or the n axis aligns with the Z axis and also you make u align with x and v align with y axis respectively. That was your world coordinate system xyz and you have to align this, this figure comes again.

(Refer Slide Time: 38:07)

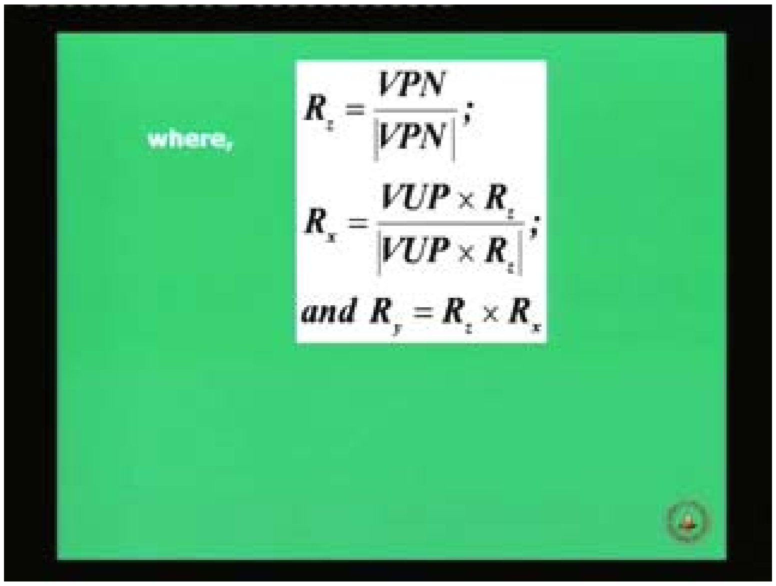


Step 2 in normalizing transformations:

Rotate VRC such that VPN (n-axis) aligns with Z-axis (also, u with X and v with Y)

I repeat this because this is essentially what we were planning to do and we have to do and we have understood the basis theory of how to do that. How do you do that? Now you need to just align VPN with Z axis, u with the X axis, v with Y axis, this was our original problem and we know to solve it. We know to solve it because if you know the vectors VPN u and v respect to the world coordinate system xyz you can form a rotation matrix with the help of these vectors and that is the rotation matrix which will help you to get the system aligned with the world coordinate system. So let us look at this form.

(Refer Slide Time: 38:41)



where,

$$R_z = \frac{VPN}{|VPN|};$$

$$R_x = \frac{VUP \times R_z}{|VUP \times R_z|};$$

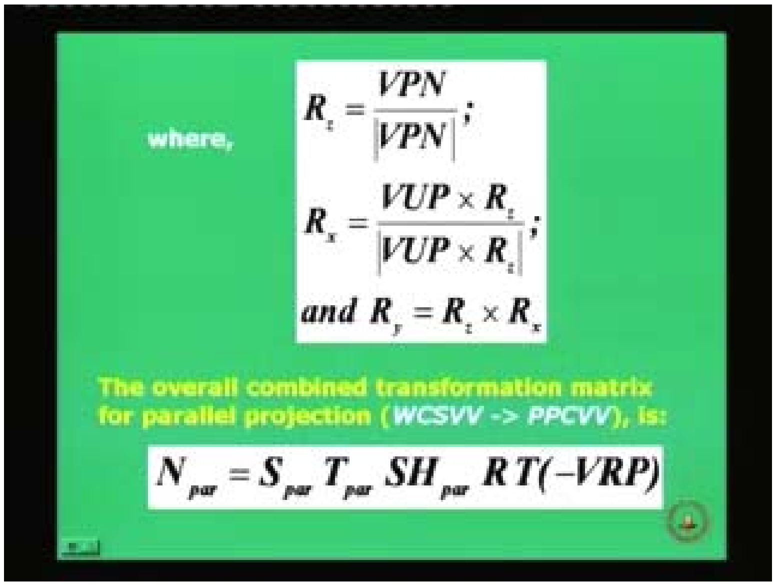
$$and\ R_y = R_z \times R_x$$

That is simply like the one we did, take the z axis, the last row coefficients R_z which is nothing but the view plane normal. So take the unit vector along the view plane normal that will give the last row of the transformation matrix for rotation. The x will also be obtained by cross product of the view up vector and R_z. Let us go back to the figure to understand the significance of R_x here, it is a cross product of view up vector and the view plane normal. If you take the cross product of the view up vector, view up vector is nothing but point which is the vector lying in the nv axis and the cross product of the view up vector with VPN will actually give you nothing but the u axis.

So this specification of view up vector and VPN itself, these two vectors are sufficient to tell you where your u axis is and a combination of u and view up vector will tell you or the VPN and u will tell you where is your v but to get u, given VUP and given VPN, I repeat the given VUP or view up vector or given view plane normal or the VPN the cross product of these two will give you the u axis and that is the concept of taking the cross product of view plane normal which is nothing but R_z and the VUP gives elements of x and of course the elements of y which is the second row can be obtained easily by the cross product of R_z and R_x. So these are the equations which will help you to get the equations for the combined transformation of the rotation matrix. That is step two.

The step two coefficients can be obtained by Rx R_y R_z respectively, individual rows and they are computed. Nothing is required except you require the view plane normal and view up vector. Once VPN and VUP are given you can actually obtain the R_x R_y R_z elements required for the rotational matrix, that is the step two.

(Refer Slide Time: 39:48)



where,

$$R_z = \frac{VPN}{|VPN|};$$

$$R_x = \frac{VUP \times R_z}{|VUP \times R_z|};$$

$$\text{and } R_y = R_z \times R_x$$

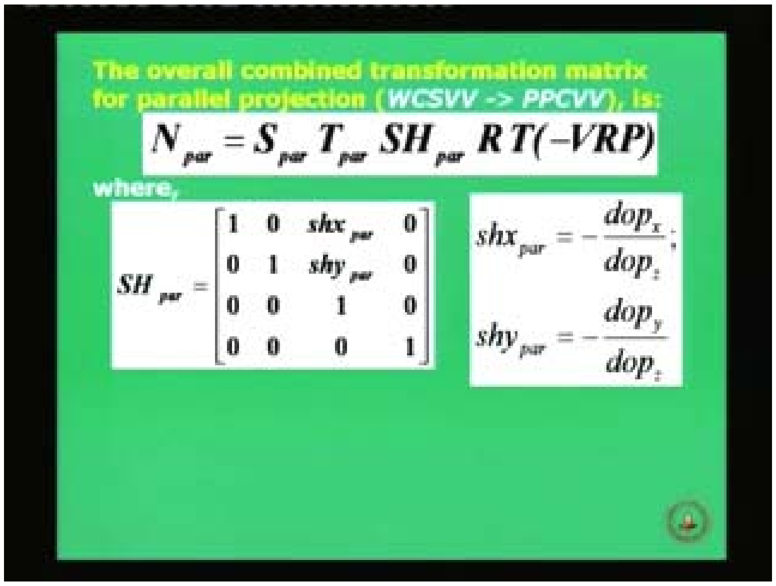
The overall combined transformation matrix for parallel projection (*WCSVV -> PPCVV*), is:

$$N_{par} = S_{par}\,T_{par}\,SH_{par}\,R\,T(-VRP)$$

In step two, you remember, there were five different transformation matrix required for the overall combined normalize transformation matrix for projection. I have put in brackets WCSVV to be projected to PPCVV. I will explain what these terminologies mean. Well WCSVV expand to world coordinate system view volume has to be transforms to a parallel projection canonical view volume PPCVV. World coordinate system view volume to a parallel projection canonical view volume that is what we have been discussing as the overall combined transformation matrix and it consists of five different transformation matrices.

First you remember the steps; translation, rotation, shear and translation, scale again. And after the translation with respect to minus VRP so far in most part of the talk in the last an hour so I have to discussed how to get this R and the equations to obtain the elements of R are given in the same slide in the top. The third, the last row of R is given by the VPN VUP multiplied by VPN cross product gives you the first row or the elements of x and the R_x and R_y can be obtained by the cross product. So once you get R the first two stages are over, you need to only find out how to get the shear transformation matrix and what should be translation and scale. So in the remaining time we will discuss these three transformation matrix and complete the first stage of the viewing pipeline of the normalizing transform.

(Refer Slide Time: 42:17)


The overall combined transformation matrix for parallel projection (*WCSVV -> PPCVV*), is:

$$N_{par} = S_{par}\, T_{par}\, SH_{par}\, R\, T(-VRP)$$

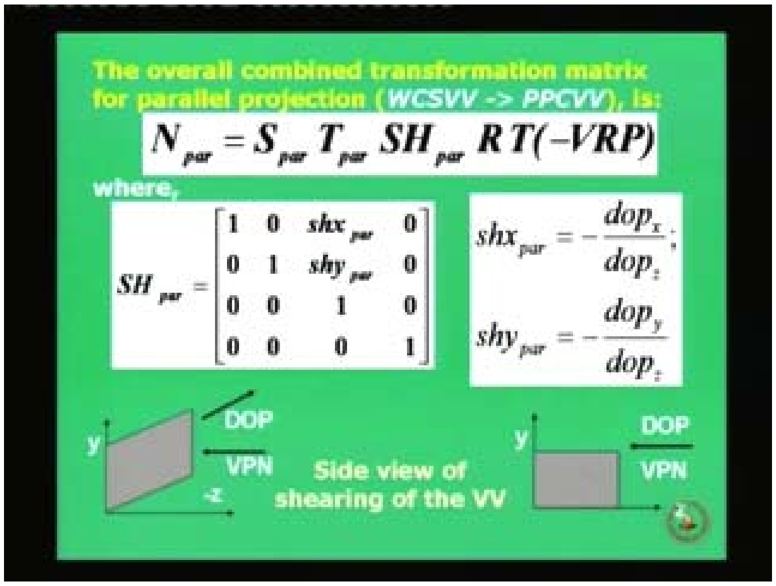
where,

$$SH_{par} = \begin{bmatrix} 1 & 0 & shx_{par} & 0 \\ 0 & 1 & shy_{par} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$shx_{par} = -\frac{dop_x}{dop_z}; \qquad shy_{par} = -\frac{dop_y}{dop_z}$$

This was the equation, the combined transformation matrix for parallel projection which transforms a world coordinate system view volume to a parallel projection canonical view volume WCSVV to a PPCVV is given by these five transformation matrices and we know that how to get R and T. What is a shear matrix? The shear is necessary if you remember carefully we were discussing these steps, the shear matrix is necessary for parallel projection which is non-orthographic. Parallel projection could be of two types; orthographic and non-orthographic as well as symmetric and asymmetric and all that. What is the difference between orthographic and non-orthographic among the class of parallel projections. Well in case of parallel projection there is no doubt that all the rays will be parallel and they will be going towards the direction of projection, somewhere towards infinity. COP is inter projection and it goes to infinity, that is why the rays are all parallel. But there is a direction of projection, the direction of projection may not be normal to the view plane. If it is normal and gives orthographic, if it is not normal to the view plane if this is the view plane then they are all perpendicular to the view plane, you get perfect orthographic parallel projection. But you can have non-orthographic parallel projection where the parallel rays are now not normal to the surface, they just come and hit this in an inclined manner. So that is the case where you need to provide a shear to the view volume now, otherwise if it is pure orthographic rays are anyway normal to the projection plane, then you do not need to provide this third step or the middle step of shear and to explain this shear step I have two figures where before this is the side view of the shearing of the view volume.

(Refer Slide Time: 43:55)



The overall combined transformation matrix for parallel projection (*WCSVV -> PPCVV*), is:

$$N_{par} = S_{par} T_{par} SH_{par} R T(-VRP)$$

where,

$$SH_{par} = \begin{bmatrix} 1 & 0 & shx_{par} & 0 \\ 0 & 1 & shy_{par} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$shx_{par} = -\frac{dop_x}{dop_z}; \quad shy_{par} = -\frac{dop_y}{dop_z}$$

Side view of shearing of the VV

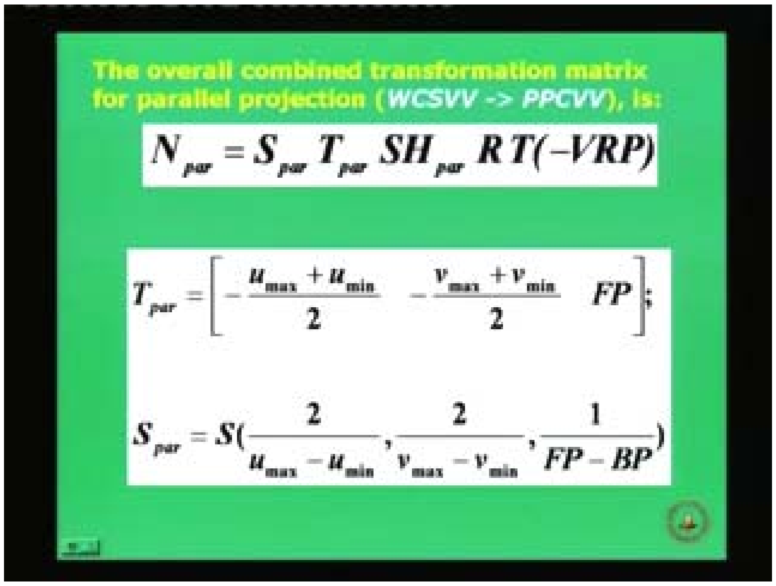
y
DOP
VPN
-z
y
DOP
VPN

What actually may happening is as you can see the left hand side at the bottom of the frame you have the view plane normal and the direction of projection is somewhere different, it is not parallel to the view plane normal. This can happen in a parallel projection which is non-orthographic in nature.

In case of an orthographic automatically direction of projection and view plane normal will actually be parallel, you do not have to provide the shear. But here the direction of projection is not aligned with the view plane normal and we need to provide a shear to this view volume to make the direction of projection align with the view plane normal. That is given in the side, with the side view of the shearing of the view volume you can provide it shear, the right hand side figure; the rhombus type of figure basically becomes a perfect rectangle or a square and that helps to make a direction of projection and the view plane normal and now they are aligned. That can be provided by two types of shear along with the x and y axis you need to give the shear. And the matrix is given here SH par and just two elements here SHx par and SHy par, I eave it as an exercise for the you, the first exercise in today’s lecture is to obtain this Shx par and Shy par elements of the shear, you have discussed about shear matrix in the 3D transformation.

You should be able to derive that but the answer is given to you here where the elements are nothing but the ratio of x to z component of direction of projection vector.DOP vector is direction of projection vector so the x component of that divided by z gives you the SHx par and the ratio of y to z gives you the SHy par. The SHy par is given by DOPy by DOPz respectively.

(Refer Slide Time: 45:50)



The overall combined transformation matrix for parallel projection (*WCSVV -> PPCVV*), is:

$$N_{par} = S_{par} T_{par} SH_{par} R T(-VRP)$$

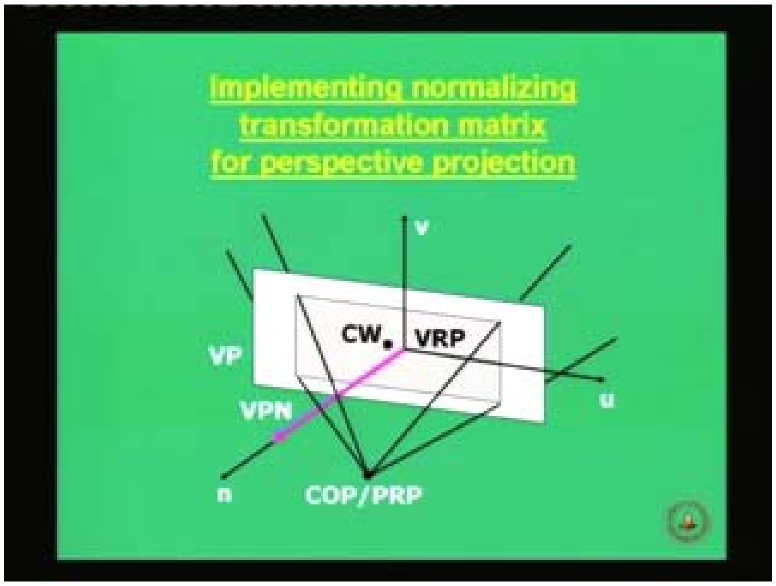
$$T_{par} = \left[-\frac{u_{max} + u_{min}}{2} \quad -\frac{v_{max} + v_{min}}{2} \quad FP \right];$$

$$S_{par} = S(\frac{2}{u_{max} - u_{min}}, \frac{2}{v_{max} - v_{min}}, \frac{1}{FP - BP})$$

We are discussing this stage of overall combined transformation and now there are two stages left; one is scale and the other is translation. I leave it as an exercise for you, these last two stages of translation and scale are required because the view volume has been transformed after rotation, translation, shear but it has to be shrunk to the canonical view volume defined by the six planes in parallel projection. And it would have scaling before itself, you may need to translate it such that the front plane gets aligned with the corresponding XY plane and the corresponding U_{max} U_{min} V_{max} V_{min} have to be aligned.

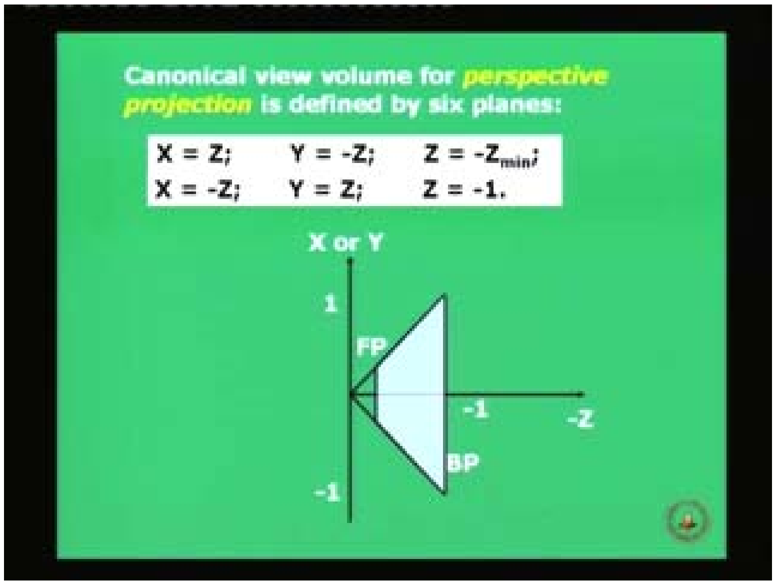
So you need a little bit of translation in 3D to align it perfectly and then give it an overall scale factor. You see that the T tree matrix is a vector whereas the x is a functional form. These are the three diagonal elements of the T matrix which are given in the last equation and you provide that scale, you provide that scale to actually fit it and form the canonical view volume in parallel. In the remaining time we will quickly go through the perspective transformation matrix for normalizing transformation and the steps are almost similar to the parallel projection, only now the canonical view volume have a different shape is like a pyramid unless in the case of a rectangular parallelepiped in the case of a parallel projection.

(Refer Slide Time: 47:09)



Implementing normalizing transformation matrix for perspective projection

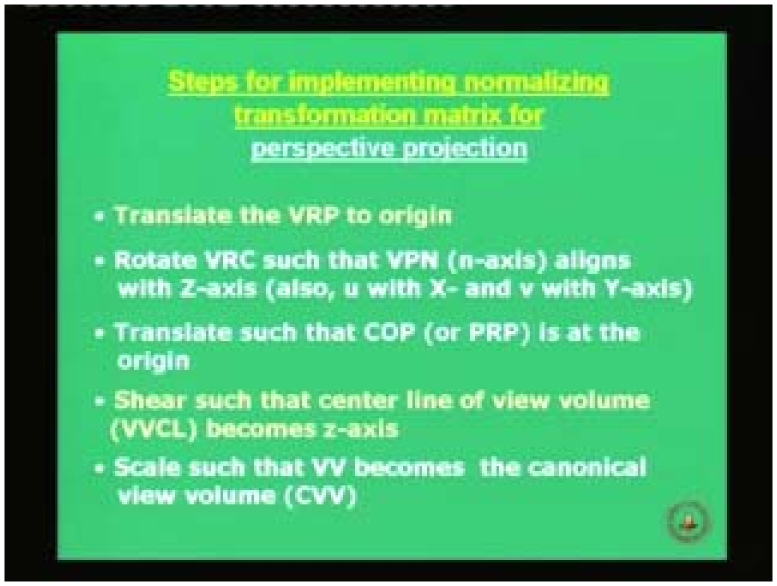
(Refer Slide Time: 47:15)



Canonical view volume for *perspective projection* is defined by six planes:

$X = Z;$	$Y = -Z;$	$Z = -Z_{min};$
$X = -Z;$	$Y = Z;$	$Z = -1.$

(Refer Slide Time: 47:55)



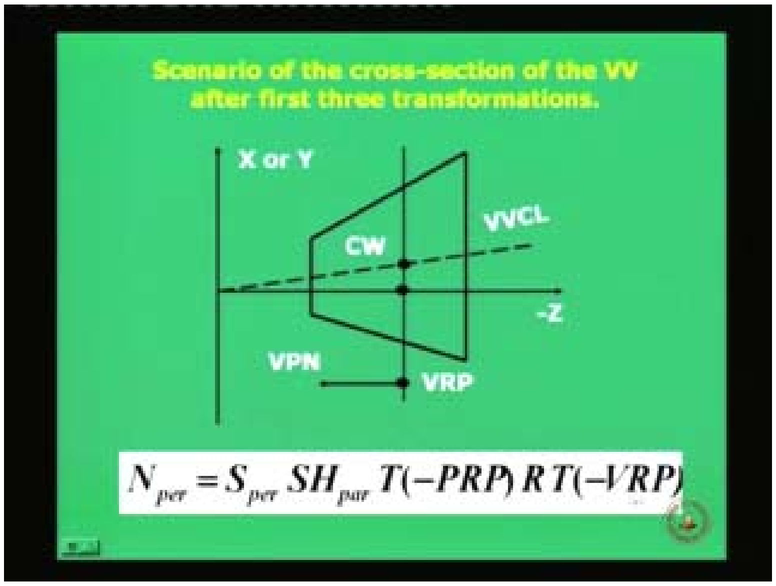
Steps for implementing normalizing transformation matrix for perspective projection

- Translate the VRP to origin
- Rotate VRC such that VPN (n-axis) aligns with Z-axis (also, u with X- and v with Y-axis)
- Translate such that COP (or PRP) is at the origin
- Shear such that center line of view volume (VVCL) becomes z-axis
- Scale such that VV becomes the canonical view volume (CVV)

These are the equations, six planes we discussed about in case of perspective projection. So it forming a pyramid we are having a side view in XZ plane or YZ plane only we are looking if we look into either XZ or YZ you will have this figure, it is basically like a pyramidal structure. I will quickly go through the steps.

It is almost the same, translate the VRP to the origin, rotate the VRC, the second step is also the same we do not need a shear now, prior to that we need a translation because there is a center of projection or a perspective reference point or a PRP or COP which comes. You need to provide a translation first, then you need to provide a shear such that the center line of the view volume becomes the z axis like the direction of projection which you had in the case of parallel projection and then of course you do a scale. So a sequence of steps are almost similar especially the first two and then the last three are little bit jumbled up in the sense that translation first, then the shear and the then the scale. This is the scenario of the cross section of the view volume after first three transformations.

(Refer Slide Time: 48:04)

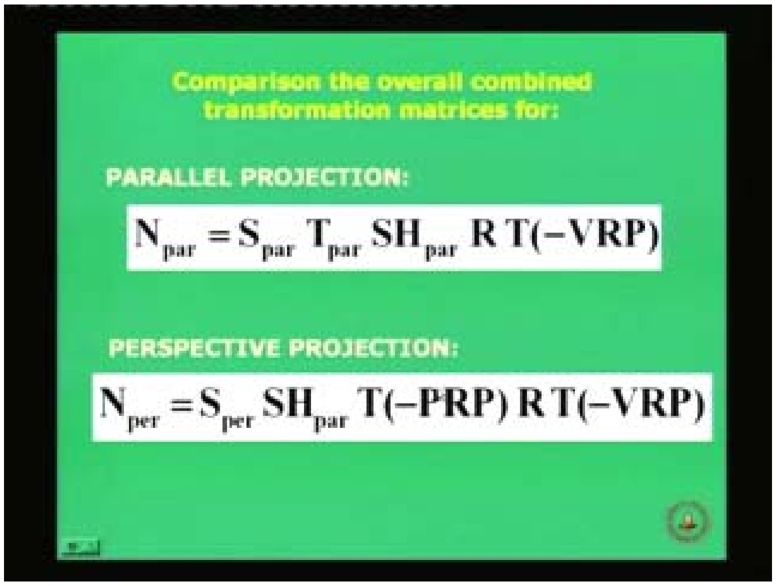


Scenario of the cross-section of the VV after first three transformations.

$$N_{per} = S_{per}\, SH_{par}\, T(-PRP)\, R\, T(-VRP)$$

What are the first three transformations? Translate with respect to minus VRP, rotate which we have discussed first, the rotation of one system to the other and then of course you translate it to the minus with respect to minus COP or PRP, then you need to give it a shear because what may happen, after the first three stages of translation, rotation, translation is the view volume, before giving it a scaling the view volume might appear to be something like this, it may not be perfectly aligned with z axis. You need to do this otherwise the canonical view volume will not come in shape and it cannot apply the projectory transformation later on. So the view volume center align VVCL as given here passes the center of the window from the origin and these has to be aligned with respect to the minus z axis and that is what is required and that will be divided to the SHpar matrix which should give you that shear.

(Refer Slide Time: 50:10)



Comparison the overall combined transformation matrices for:

PARALLEL PROJECTION:

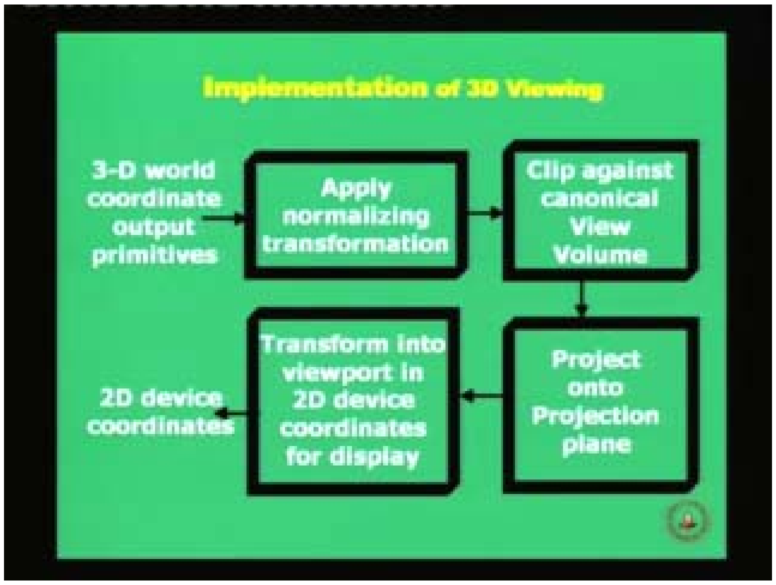
$$N_{par} = S_{par}\, T_{par}\, SH_{par}\, R\, T(-VRP)$$

PERSPECTIVE PROJECTION:

$$N_{per} = S_{per}\, SH_{par}\, T(-PRP)\, R\, T(-VRP)$$

So this is the overall picture and just as a comparative study of the overall combined transformation matrices for parallel and perspective. As you can see here the nature is almost same. Specifically the first two transformations T of minus VRP and R are same. Then you have a translation and shear in the case of perspective and in the case of a parallel on the top you first have the shear and then translation. So that is the reverse and the end but anyway you have to scale it to the corresponding canonical view volume. So that is the comparison between the two projection geometries of transformation matrices. And now if you go and look back, we will stop with this today and we come back to the original viewing pipeline.

(Refer Slide Time: 49:44)



Implementation of 3D Viewing

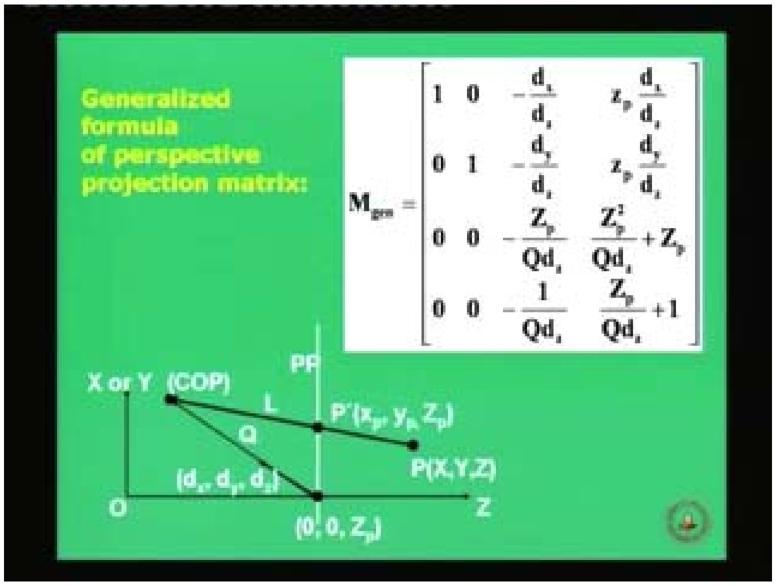
3-D world coordinate output primitives
Apply normalizing transformation
Clip against canonical View Volume
Project onto Projection plane
Transform into viewport in 2D device coordinates for display
2D device coordinates

The implementation of 3D viewing pipeline, we started to talk about today and you remember, to repeat again we started with 3D world coordinate output primitives, apply a normalizing transformation, then clip then project on projection plane and ultimately the 2D transformation to get the 2D device coordinates. So far what we have discussed in the entire lecture today is the first stage only. The applying normalizing transformation and that to two parts of it; one is the parallel geometry, parallel projection and another is the perspective geometry. And among those there are five steps of transformations in the first stage itself to applying the normalizing transformation and among those five steps the second step rotation was probably the most crucial one. And of course we have to apply the shear and the scale as well.

So, applying the normalize transformation and consists of five steps, both in the case of perspective and parallel projection geometries. Then you of course need to provide a clip in the canonical view volume and then we project into projection plane. Now the third stage is the projection plane, this we have studied earlier in the case of 3D transformation itself, I will just flash this slide so you can recollect the general transformation matrix for projection and once that is done you just need a 2D scaling in terms of the transform into view port in 2D coordinate for display. And the next slide is the third stage, it is because the second stage of clipping we will discuss when we talk of clipping entirely in 2D and 3D.

So now we know the first stage, the second stage of clipping will come in due course of time of clipping, third stage in fact we have already discussed under 3D dimensional transformations and this was the generalized formula. If you remember the generalized formula for perspective projection matrix was given as this. This we derived in the pervious lecture under three dimensional transformations.

(Refer Slide Time: 51:23)



Generalized formula of perspective projection matrix:

$$M_{gen} = \begin{bmatrix} 1 & 0 & -\frac{d_x}{d_z} & z_p\frac{d_x}{d_z} \\ 0 & 1 & -\frac{d_y}{d_z} & z_p\frac{d_y}{d_z} \\ 0 & 0 & -\frac{Z_p}{Qd_z} & \frac{Z_p^2}{Qd_z}+Z_p \\ 0 & 0 & -\frac{1}{Qd_z} & \frac{Z_p}{Qd_z}+1 \end{bmatrix}$$

Under three dimensional transformations we discussed a lot about projection geometry, parallel axonometric, trignometric one point, two points, three point perspective geometric and also transformation matrix in the case of perspective geometry. And also this generalized projection matrix is applicable for both perspective and parallel and if you recollect back into the notes which you have taken from the slide, now you can put special cases of the value of (d_x d_y d_z) direction of projection and q which will take the COP either infinity or in a finite distance will give you projections of perspective or perspective or perspective or parallel or orthographic. So with this we stop today and we continue with the 3D viewing pipeline in the next lecture. This is the perspective projection matrix which we use as the third stage of the viewing pipeline. I go back and we stop this same slide which we were taking and we started with this slide where we have discussed extensively about applying normalized transformation and we have also seen the expression of the third stage which is projected onto the projection plane.

We will probably not discuss canonical view volume clipping now. A little bit of transform into view port we will discuss. But in the next lecture we will start on this and we will see the viewing pipeline broken up into several stages where we start from object model coordinate system pass through world coordinate system go into the eye space or image coordinates and then to device with it. So it is entirely a sequence of transformations which takes your object somewhere in space, the camera is somewhere here, the model is somewhere here and the sequence of transformations which will slowly take all these into the device coordinates in the screen on the TV or on the projection system which you see. So we stop here with the first lecture on 3D viewing and we will continue this with the 3D viewing pipeline with different matrices revisited again at the different stages and how to get the 2D devise coordinates and we will put it. Thank you.