Κεφάλαιο 4ο
Προβολές

Εισαγωγή

Στο Κεφάλαιο 2 αναλύσαμε τις τεχνικές σχεδίασης στις δύο διαστάσεις. Σε αυτό το κεφάλαιο θα επεκταθούμε σε τεχνικές αναπαράστασης τρισδιάστατων σκηνών στο επίπεδο του παρατηρητή. Η απεικόνιση τρισδιάστατων σκηνών καθορίζεται, στη γενικότερη περίπτωση, από την οπτική γωνία από την οποία ο θεατής παρατηρεί τη σκηνή, καθώς και από τον τύπο προβολής που επιλέγουμε για την αναπαράσταση της σκηνής.

Με αλλαγή της θέσης και του προσανατολισμού της κάμερας, προκύπτει διαφορετική αναπαράσταση της σκηνής στο διδιάστατο επίπεδο της συσκευής εξόδου. Η δυνατότητα αλλαγής της οπτικής γωνίας προσφέρεται με το μετασχηματισμό οπτικής γωνίας και μας επιτρέπει την επισκόπηση της σκηνής από πολλαπλές θέσεις. Επιπλέον, υπάρχει η δυνατότητα χρήσης διαφορετικών τύπων προβολής της τρισδιάστατης σκηνής, ανάλογα με την επιθυμητή απόδοση της σκηνής. Η παράλληλη προβολή διατηρεί τις αναλογίες των σχημάτων και είναι χρήσιμη για την αναπαράσταση σχεδίων και γενικά για εφαρμογές που η διατήρηση της κλίμακας έχει σημασία. Εάν ο προγραμματιστής αποσκοπεί στην απόδοση ρεαλιστικών σκηνών, χρησιμοποιεί την προοπτική προβολή, η οποία αποδίδει το σκηνικό, ακολουθώντας τους κανόνες Οπτικής που ακολουθούν οι κάμερες και το ανθρώπινο μάτι. Στο κεφάλαιο αυτό, αναλύουμε τις αρχές και τον τρόπο παραγωγής παράλληλης και της προοπτικής προβολής.

4.1 Μετασχηματισμός οπτικής γωνίας

Στο Κεφάλαιο 2, κατά την εξέταση των αλγορίθμων αποκοπής και μετασχηματισμού παρατήρησης στις δύο διαστάσεις, το επίπεδο παρατήρησης ταυτιζόταν με το επίπεδο XY. Ωστόσο, σε τρισδιάστατες σκηνές, η ταύτιση αυτή δεν είναι υποχρεωτική. Απεναντίας, υπάρχει η δυνατότητα αλλαγής της οπτικής γωνίας από την οποία παρατηρούμε τη σκηνή. Αυτή η δυνατότητα μας επιτρέπει την επισκόπηση της σκηνής από πολλαπλές θέσεις παρέχοντας λ.χ. την ικανότητα δημιουργίας κατόψεων και πλαγίων όψεων σε εφαρμογές CAD.

Στο μετασχηματισμό οπτικής γωνίας θεωρούμε το σύστημα **συντεταγμένων παρατηρητή**. Η θέση αυτού του συστήματος συντεταγμένων καθορίζεται από δύο παράγοντες:

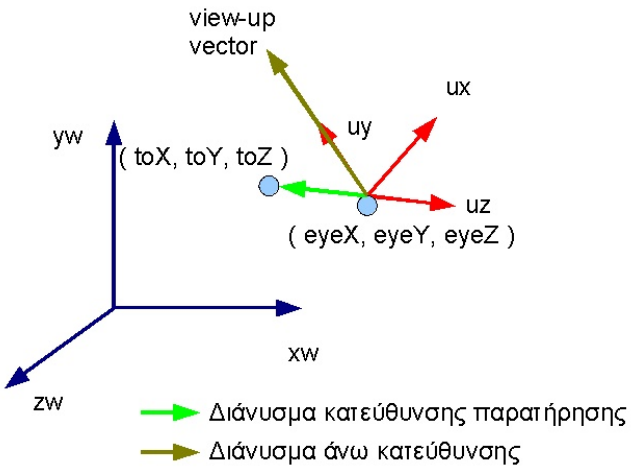
- Από τη θέση του παρατηρητή: Η αρχή του συστήματος συντεταγμένων παρατηρητή ταυτίζεται με το σημείο παρατήρησης.

- Από την κατεύθυνση παρατήρησης: Η κατεύθυνση παρατήρησης ταυτίζεται με τον αρνητικό ημιάξονα z του συστήματος συντεταγμένων παρατηρητή.

Η θέση της κάμερας στη σκηνή προσδιορίζεται με τις συντεταγμένες ενός σημείου, έστω $eye = (eyeX, eyeY, eyeZ)$. Ο προσανατολισμός της κάμερας προσδιορίζεται με δύο διανύσματα: το **διάνυσμα κατεύθυνσης παρατήρησης** και το *διάνυσμα* **άνω κατεύθυνσης** (**view-up vector**). Το πρώτο διάνυσμα δηλώνει την κατεύθυνση προς την οποία είναι προσανατολισμένος ο παρατηρητής. Ταυτίζεται με τον αρνητικό ημιάξονα των z του **συστήματος συντεταγμένων του παρατηρητή** και η αρχή του βρίσκεται τη θέση του παρατηρητή. Το διάνυσμα άνω κατεύθυνσης δηλώνει την προς τα πάνω κατεύθυνση του επιπέδου προβολής.

Προφανώς, εφόσον το διάνυσμα κατεύθυνσης παρατήρησης προσδιορίζει τον αρνητικό ημιάξονα z του συστήματος συντεταγμένων παρατηρητή, το διάνυσμα άνω κατεύθυνσης που θα ορίσουμε αρκεί να είναι κάθετο στο διάνυσμα κατεύθυνσης παρατήρησης. Ωστόσο, σε περίπτωση του το δοθέν διάνυσμα άνω κατεύθυνσης δεν είναι κάθετο στο διάνυσμα διεύθυνσης παρατήρησης λαμβάνεται υπόψη μόνο η κάθετη συνιστώσα του.

Ουσιαστικά, με το μετασχηματισμό οπτικής γωνίας, καλούμαστε να αναπαραστήσουμε τη σκηνή ως προς το σύστημα συντεταγμένων παρατηρητή. Αρκεί λοιπόν να αναγάγουμε την περιγραφή της σκηνής σε ένα σύστημα συντεταγμένων που η αρχή των αξόνων του ταυτίζεται με τη θέση της κάμερας, ο αρνητικός ημιάξονάς του z δηλώνει την κατεύθυνση παρατήρησης και ο θετικός ημιάξονας Oy έχει τον ίδιο προσανατολισμό με το διάνυσμα άνω κατεύθυνσης.



Σχ 4.1 Μετασχηματισμός οπτικής γωνίας. Τα διανύσματα κατεύθυνσης παρατήρησης και άνω κατεύθυνσης επιπέδου παρατήρησης καθορίζουν τον προσανατολισμό του συστήματος συντεταγμένων παρατηρητή

Αρχικά λοιπόν μετατοπίζουμε την αρχή των αξόνων στη θέση παρατήρησης με το μητρώο μετατόπισης:

$$T = \begin{bmatrix} 1 & 0 & 0 & -eyeX \\ 0 & 1 & 0 & -eyeY \\ 0 & 0 & 1 & -eyeZ \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Μετά την μετατόπιση του συστήματος συντεταγμένων στη θέση $(eyeX, eyeY, eyeZ)$ έπεται η περιστροφή του, ούτως ώστε ο αρνητικός ημιάξονας z να ταυτιστεί με τη φορά παρατήρησης και το διάνυσμα άνω κατεύθυνσης να ταυτιστεί με το θετικό ημιάξονα Oy. Εάν οι συνιστώσες κατεύθυνσης των αξόνων του περιστραμμένου συστήματος συντεταγμένων είναι γνωστές, η περιγραφή της σκηνής ως προς αυτό το σύστημα συντεταγμένων, προκύπτει με το μητρώο μετασχηματισμού:

$$R = \begin{bmatrix} u_{x1} & u_{x2} & u_{x3} & 0 \\ u_{y1} & u_{y2} & u_{y3} & 0 \\ u_{z1} & u_{z2} & u_{z3} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Επομένως, ο μετασχηματισμός οπτικής γωνίας εκτελείται με την εξής αλληλουχία πολλαπλασιασμών μητρώων:

$$\begin{bmatrix} x_{eye} \\ y_{eye} \\ z_{eye} \\ 1 \end{bmatrix} = R \cdot T \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Στην OpenGL, ο μετασχηματισμός οπτικής γωνίας εκτελείται με απλό τρόπο χρησιμοποιώντας την εντολή ***gluLookAt***:

gluLookAt(GLdouble eyeX, GLdouble eyeY, GLdouble eyeZ, GLdouble centerX, GLdouble centerY, GLdouble centerZ, GLdouble upX, GLdouble upY, GLdouble upZ);

Οι συντεταγμένες $eyeX$, $eyeY$, $eyeZ$ καθορίζουν τη θέση του παρατηρητή ως προς το σύστημα συντεταγμένων σκηνής. Ο προσανατολισμός της κάμερας καθορίζεται από το διάνυσμα με αρχή το σημείο $(eyeX, eyeY, eyeZ)$ και πέρας το σημείο $(centerX, centerY, centerZ)$. Οι τιμές (upX, upZ, upZ) καθορίζουν τον προσανατολισμό του διανύσματος άνω κατεύθυνσης.

Με τη χρήση της εντολής ***gluLookAt***, ο προγραμματιστής μπορεί να καθορίσει με ευκολία τη θέση και τον προσανατολισμό παρατήρησης. Με αυτό τον τρόπο, η OpenGL απαλλάσσει τον προγραμματιστή από τον ορισμό περίπλοκων μετασχηματισμών αλλαγής συστήματος συντεταγμένων.

Δεδομένου ότι η ***gluLookAt*** επενεργεί στο μητρώο μετασχηματισμού μοντέλου, πρέπει να μεταβούμε στην κατάσταση τροποποίησής του πριν την εκτέλεση της εντολής, δίνοντας στην ***glMatrixMode*** την παράμετρο *GL_MODELVIEW*:

glMatrixMode(GL_MODELVIEW);

Η ***gluLookAt*** ουσιαστικά πολαπλασιάζει το μητρώο μετασχηματισμού οπτικής γωνίας που ορίστηκε παραπάνω με το τρέχον μητρώο μετασχηματισμού μοντέλου από δεξιά. Επομένως είναι σημαντικό η εντολή ***gluLookAt*** και οι μετέπειτα εντολές μετασχηματισμού μοντέλου να εκτελεστούν με τη σωστή διαδοχή. Δεδομένου ότι ο μετασχηματισμός αλλαγής οπτικής γωνίας εφαρμόζεται στα σημεία της σκηνής αφού η τελευταία συντεθεί, **η εντολή *gluLookAt* θα πρέπει να δηλωθεί μετά την εντολή αρχικοποίησης του μητρώου μετασχηματισμού μοντέλου και πριν από οποιοδήποτε μετασχηματισμό μοντέλου**.

4.2 Είδη προβολών - Μητρώο προβολής

Η διαδικασία της προβολής ουσιαστικά είναι μια διεργασία, κατά την οποία, οι συντεταγμένες της σκηνής αντιστοιχίζονται, βάσει ενός καθορισμένου κανόνα, σε συντεταγμένες πάνω στο επίπεδο προβολής. Ο μαθηματικός κανόνας στον οποίο βασίζεται η αντιστοίχιση αυτή, αποθηκεύεται στο *μητρώο* **προβολής**. Το μητρώο προβολής πολλαπλασιάζεται με τις συντεταγμένες των σημείων της σκηνής (τις ανηγμένες στο σύστημα συντεταγμένων του παρατηρητή) και παράγει (σε ενδιάμεσο στάδιο) τις συντεταγμένες στις οποίες προβάλλονται στο επίπεδο του παρατηρητή, το λεγόμενο ***επίπεδο προβολής***. Προφανώς, ανάλογα με τον τύπο της προβολής μεταβάλλεται και η δομή του μητρώου προβολής

Οι ευρέως χρησιμοποιούμενοι τύποι προβολής είναι η παράλληλη και η προοπτική. Το είδος προβολής που θα επιλέξουμε για την απόδοση της σκηνής στο παράθυρο παρατήρησης εξαρτάται από το είδος της εφαρμογής που υλοποιούμε. Λ.χ., σε εφαρμογές CAD, ενδιαφερόμαστε για την απόδοση των αντικειμένων, διατηρώντας τις αναλογίες στις διαστάσεις τους, χωρίς αλλοιώσεις. Αντίθετα, σε εφαρμογές σύνθεσης ρεαλιστικών σκηνών, μας ενδιαφέρει η απόδοση του σκηνικού με τον ίδιο τρόπο που θα τον αντιλαμβανόταν ένας θεατής, βάσει κανόνων της Οπτικής.

Όπως αναφέραμε στο Κεφάλαιο 3, σε κάθε χρονική στιγμή, η μηχανή καταστάσεων της OpenGL επιτρέπει την επεξεργασία ενός μόνο από τα δύο μητρώα μετασχηματισμού (του μητρώου μετασχηματισμού μοντέλου ή του μητρώου προβολής). Η επιλογή της κατάστασης γίνεται, όπως προαναφέρθηκε, με τη χρήση της εντολής ***glMatrixMode( )***. Προκειμένου λοιπόν να μεταβούμε στην κατάσταση επεξεργασίας του μητρώου προβολής, δίνουμε στην ***glMatrixMode*** το όρισμα *GL_PROJECTION*.

glMatrixMode(GL_PROJECTION);

4.3 Παράλληλη προβολή

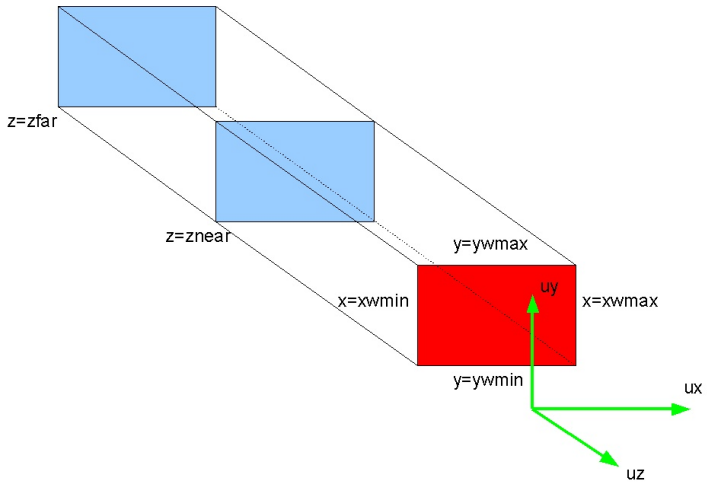
Την απλούστερη μορφή προβολής αποτελεί η **παράλληλη** ή **ορθογραφική προβολή**. Στην προβολή αυτή, τα σημεία της σκηνής προβάλλονται στο επίπεδο προβολής ακολουθώντας δέσμες κάθετες προς το επίπεδο προβολής . Κάθε σημείο δηλαδή προβάλλεται στην τομή της δέσμης του και του επιπέδου προβολής:

$$x_p = x$$
$$y_p = y$$

Συνεπώς, το μητρώο της παράλληλης προβολής ταυτίζεται με το μητρώο I_4

$$M_{ortho} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Στην OpenGL, οι διαδικασίες της παράλληλης προβολής και της αποκοπής ενοποιούνται. Δηλαδή, ταυτόχρονα με τον ορισμό της παράλληλης προβολής, δίνονται και πληροφορίες που καθορίζουν τη διαδικασία της αποκοπής, με τρόπο παρόμοιο με αυτόν της αποκοπής στις δύο διαστάσεις. Στις τρεις διαστάσεις βέβαια, αντί για ευθείες, ορίζουμε **επίπεδα αποκοπής**. Επομένως, στον άξονα x ορίζουμε το αριστερό και δεξιό επίπεδο αποκοπής $x = xw_{\min}$ και $x = xw_{\max}$, στον άξονα y ορίζουμε το κάτω και άνω επίπεδο αποκοπής $y = yw_{\min}$ και $y = yw_{\max}$ και στον άξονα z ορίζουμε το εγγύς και μακρινό επίπεδο αποκοπής $z = z_{near}$ και $z = z_{far}$. Με τον προσδιαρισμό των επιπέδων αυτών, ορίζουμε στο χώρο ένα παραλληλεπίπεδο, τα περιεχόμενα του οποίου επιλέγονται για απεικόνιση στο επίπεδο προβολής όπως φαίνεται στο Σχ. 4.2



Σχ 4.2: Επίπεδα αποκοπής στην παράλληλη προβολή

Επισημαίνουμε ότι **τα επίπεδα αποκοπής ορίζονται ως προς το σύστημα συντεταγμένων του παρατηρητή**. Επομένως, το εγγύς και μακρινό επίπεδο αποκοπής, δεδομένης της θέσης τους ως προς την αρχή των αξόνων, έχουν αρνητικές τιμές βάθους $\left(z_{far} < z_{near} < 0\right)$.

Κατόπιν της αποκοπής, έπεται ο μετασχηματισμός κανονικοποίησης, που εκτελείται με τρόπο επίσης παρόμοιο με αυτόν που εφαρμόζεται και στις δύο διαστάσεις. Στην περίπτωση της κανονικοποίησης στις τρεις διαστάσεις, το εύρος των συντεταγμένων x, y, z του αποκοπτόμενου παραλληλεπιπέδου κανονικοποιείται στο εύρος $[-1,1]$. Δηλαδή η περιγραφή του τμήματος της σκηνής που περικλείεται από τα επίπεδα αποκοπής μετασχηματίζεται ούτως ώστε να περικλείεται εντός του κύβου που ορίζεται από τα επίπεδα $x = \pm 1$, $y = \pm 1$ και $z = \pm 1$. Στην περίπτωση αυτή, το μητρώο κανονικοποίησης έχει τη μορφή:

$$M_{ortho,norm} = \begin{bmatrix} \frac{2}{xw_{\max} - xw_{\min}} & 0 & 0 & -\frac{xw_{\max} + xw_{\min}}{xw_{\max} - xw_{\min}} \\ 0 & \frac{2}{yw_{\max} - yw_{\min}} & 0 & -\frac{yw_{\max} + yw_{\min}}{yw_{\max} - yw_{\min}} \\ 0 & 0 & \frac{-2}{z_{near} - z_{far}} & \frac{z_{near} + z_{far}}{z_{near} - z_{far}} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Στην OpenGL, ο ορισμός της παράλληλης προβολής στις τρεις διαστάσεις συνδυάζεται με την δήλωση των επιπέδων αποκοπής χρησιμοποιώντας την εντολή ***glOrtho***:

void glOrtho(GLdouble xwmin, GLdouble xwmax, GLdouble ywmin, GLdouble ywmax, GLdouble dnear, GLdouble dfar);

όπου *xwmin*, *xwmax*, *ywmin* και *ywmax* το αριστερό, το δεξιό, το κάτω και το άνω επίπεδο αποκοπής αντίστοιχα. Οι συντεταγμένες του εγγύς και μακρινού επιπέδου αποκοπής δίνονται ως θετικοί αριθμοί *dnear* και *dfar* αντίστοιχα και ισχύει:

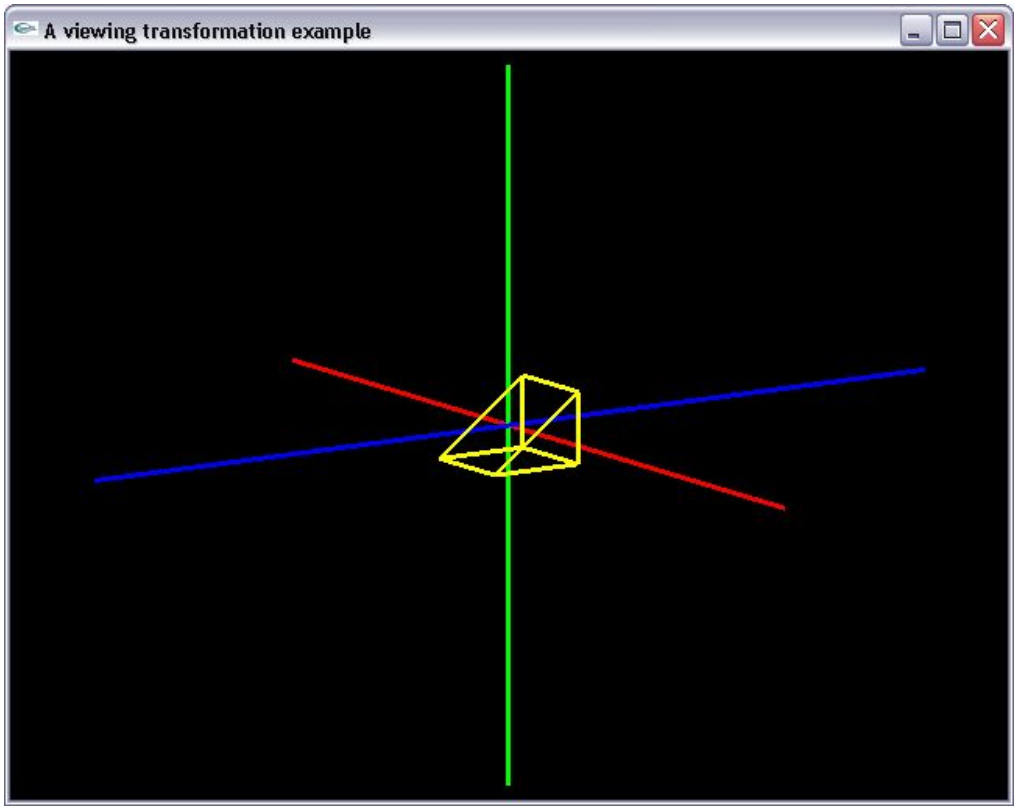
$$z_{near} = -d_{near}$$

$$z_{far} = -d_{far}$$

Παράδειγμα: Μετασχηματισμός οπτικής γωνίας και παράλληλη προβολή

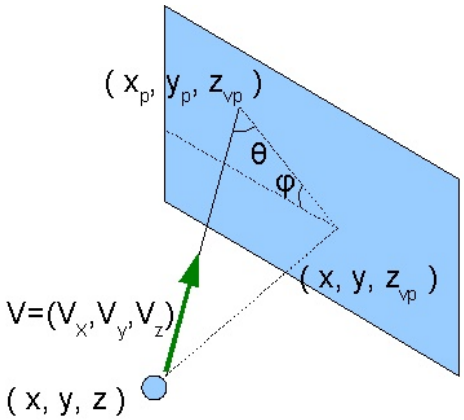
```
#include <glut.h>

GLfloat eyeX,eyeY,eyeZ;
GLfloat toX,toY,toZ;
GLfloat viewUpX, viewUpY, viewUpZ;

void display()
{
	glClearColor(0,0,0,0);
	glClear(GL_COLOR_BUFFER_BIT);

	//Drawing the x axis in red
	glColor3f(1,0,0);
	glBegin(GL_LINES);
	glVertex3f(-100,0,0);
	glVertex3f(100,0,0);
	glEnd();

	//Drawing the y axis in green
	glColor3f(0,1,0);
	glBegin(GL_LINES);
	glVertex3f(0,-100,0);
	glVertex3f(0,100,0);
	glEnd();

	//Drawing the z axis in blue
	glColor3f(0,0,1);
	glBegin(GL_LINES);
	glVertex3f(0,0,-100);
	glVertex3f(0,0,100);
	glEnd();


	//Drawing a yellow shape

	glColor3f(1,1,0);

	glBegin(GL_LINE_LOOP);
	glVertex3f(-10,-10,10);
	glVertex3f(10,-10,10);
	glVertex3f(10,-10,-10);
	glVertex3f(-10,-10,-10);
	glEnd();
```

```
	glBegin(GL_LINE_LOOP);
	glVertex3f(-10,-10,10);
	glVertex3f(10,-10,10);
	glVertex3f(10,10,-10);
	glVertex3f(-10,10,-10);
	glEnd();

	glBegin(GL_LINE_LOOP);
	glVertex3f(-10,-10,-10);
	glVertex3f(10,-10,-10);
	glVertex3f(10,10,-10);
	glVertex3f(-10,10,-10);
	glEnd();

	glFlush();
}

int main(int argc, char **argv)
{
	glutInit(&argc,argv);
	glutInitWindowPosition(50,50);
	glutInitWindowSize(640,480);
	glutCreateWindow("A viewing transformation example");

	glMatrixMode(GL_PROJECTION);
	glOrtho(-100,100,-100,100,-100,100);
	glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);


	glMatrixMode(GL_MODELVIEW);

	eyeX=30;
	eyeY=10;
	eyeZ=20;
	toX=0;
	toY=0;
	toZ=0;
	viewUpX=0;
	viewUpY=1;
	viewUpZ=0;

	gluLookAt(eyeX,eyeY,eyeZ,toX,toY,toZ,viewUpX,viewUpY,viewUpZ);

	glutDisplayFunc(display);
	glutMainLoop();
	return 0;
}
```



4.4 Πλάγια παράλληλη προβολή

Στην παράλληλη προβολή θεωρήσαμε ότι τα σημεία της σκηνής ακολουθούν δέσμες κάθετες προς το επίπεδο προβολής. Ωστόσο, είναι δυνατόν να ορίσουμε προβολές στις οποίες τα σημεία ακολουθούν ακτίνες που σχηματίζουν γωνία διαφορετική των 90 μοιρών με το επίπεδο προβολής όπως φαίνεται στο Σχ. 4.3. Στην περίπτωση αυτή ορίζεται ή **πλάγια παράλληλη προβολή** (**oblique parallel projection**).



Σχ. 4.3 Αρχή δημιουργίας πλάγιας παράλληλης προβολής

Στην πλάγια παράλληλη προβολή θεωρούμε ένα διάνυσμα $V = (V_x, V_y, V_z)$ που η διεύθυνσή του ταυτίζεται με τη διεύθυνση των ακτίνων που προβάλλουν το κάθε σημείο της σκηνής στο επιπεδο προβολής.

.

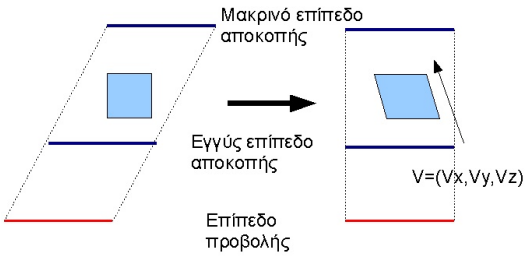
Βάσει σχέσεων αναλογίας προκύπτουν:

$$\frac{x_p - x}{z_{vp} - z} = \frac{V_x}{V_z}$$
$$\frac{y_p - y}{z_{vp} - z} = \frac{V_y}{V_z}$$

και σε μορφή μητρώου:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\frac{V_x}{V_z} & z_{vp} \cdot \frac{V_x}{V_z} \\ 0 & 1 & -\frac{V_y}{V_z} & z_{vp} \cdot \frac{V_y}{V_z} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Παρατηρούμε λοιπόν ότι **η πλάγια παράλληλη προβολή μπορεί να εκφραστεί ισοδύναμα ως μια παράλληλη προβολή η οποία εφαρμόζεται αφού προηγηθεί στη σκηνή μετασχηματισμός κλίσης**. Ουσιαστικά, στην πλάγια παράλληλη προβολή, απομονώνουμε το περιεχόμενο της σκηνής που περικλείεται εντός ενός πλαγίου παραλληλεπιπέδου (Σχ. 4.4). Με το μετασχηματισμό κλίσης μεταφέρουμε αυτό το τμήμα της σκηνής εντός των ορίων ενός ορθογωνίου παραλληλεπιπέδου και κατόπιν εφαρμόζουμε τρισδιάστατη αποκοπή.



Σχ 4.4: Ένας μετασχηματισμός κλίσης της σκηνής σε συνδυασμό με το μητρώο παράλληλης προβολής I_4 παράγει μια πλάγια παράλληλη προβολή

Μετά την αποκοπή, ακολουθεί ο μετασχηματισμός κανονικοποίησης των συντεταγμένων στο εύρος τιμών $[-1,1]$ Επομένως το μητρώο $M_{oblique,norm}$ που εκφράζει την αλληλουχία μετασχηματισμών πλάγιας παράλληλης προβολής και κανονικοποίησης έχει τη μορφή:

$$M_{oblique,norm} = M_{ortho,norm} \cdot M_{oblique} =$$

$$= \begin{bmatrix} \frac{2}{xw_{\max} - xw_{\min}} & 0 & 0 & -\frac{xw_{\max} + xw_{\min}}{xw_{\max} - xw_{\min}} \\ 0 & \frac{2}{yw_{\max} - yw_{\min}} & 0 & -\frac{yw_{\max} + yw_{\min}}{yw_{\max} - yw_{\min}} \\ 0 & 0 & \frac{-2}{z_{near} - z_{far}} & \frac{z_{near} + z_{far}}{z_{near} - z_{far}} \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -\frac{V_x}{V_z} & z_{vp} \cdot \frac{V_x}{V_z} \\ 0 & 1 & -\frac{V_y}{V_z} & z_{vp} \cdot \frac{V_y}{V_z} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

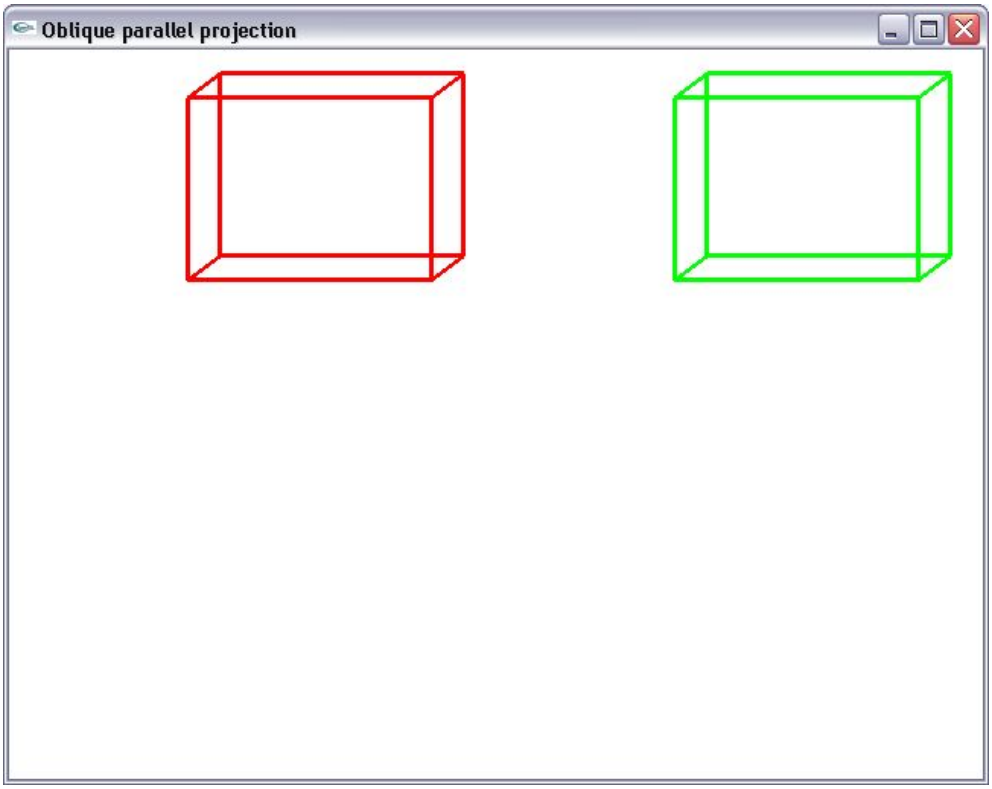
Στην OpenGL, ο ορισμός ενός μητρώου πλάγιας παράλληλης προβολής είναι εφικτός με την άμεση ανάθεση τιμών στο μητρώο προβολής χρησιμοποιώντας τις εντολές ***glLoadMatrix**** και ***glMultMatrix****.

Παράδειγμα: Πλάγια παράλληλη προβολή

```
#include <glut.h>

GLfloat *X;
GLfloat *Y;
GLfloat *Z;

//Shearing vector components
GLfloat vx,vy,vz;

GLdouble xwmin, xwmax, ywmin, ywmax, znear, zfar;

GLfloat *shearingMatrix;

void DrawCube(GLfloat *X, GLfloat *Y, GLfloat *Z)
{


      //     1--------------2
      //     |\              |\
      //     |  0--------------3
      //     |  |            | |
      //     |  |            | |
      //     |  |            | |
      //     5--|-----------6 |
      //      \ |             \|
      //        4--------------7

      glBegin(GL_LINE_LOOP);
      glVertex3f(X[0],Y[0],Z[0]);
      glVertex3f(X[1],Y[1],Z[1]);
```

```
	glVertex3f(X[2],Y[2],Z[2]);
	glVertex3f(X[3],Y[3],Z[3]);
	glEnd();

	glBegin(GL_LINE_LOOP);
	glVertex3f(X[4],Y[4],Z[4]);
	glVertex3f(X[5],Y[5],Z[5]);
	glVertex3f(X[6],Y[6],Z[6]);
	glVertex3f(X[7],Y[7],Z[7]);
	glEnd();

	glBegin(GL_LINES);
	glVertex3f(X[0],Y[0],Z[0]);
	glVertex3f(X[4],Y[4],Z[4]);
	glVertex3f(X[1],Y[1],Z[1]);
	glVertex3f(X[5],Y[5],Z[5]);
	glVertex3f(X[2],Y[2],Z[2]);
	glVertex3f(X[6],Y[6],Z[6]);
	glVertex3f(X[3],Y[3],Z[3]);
	glVertex3f(X[7],Y[7],Z[7]);
	glEnd();
}

void display()
{
	glClear(GL_COLOR_BUFFER_BIT);

	X[0]=-5; Y[0]=25; Z[0]=-10;
	X[1]=-5; Y[1]=25; Z[1]=-20;
	X[2]=-20; Y[2]=25; Z[2]=-20;
	X[3]=-20; Y[3]=25; Z[3]=-10;
	X[4]=-5; Y[4]=10; Z[4]=-10;
	X[5]=-5; Y[5]=10; Z[5]=-20;
	X[6]=-20; Y[6]=10; Z[6]=-20;
	X[7]=-20; Y[7]=10; Z[7]=-10;

	glColor3f(1,0,0);
	DrawCube(X,Y,Z);

	X[0]=10; Y[0]=25; Z[0]=-10;
	X[1]=10; Y[1]=25; Z[1]=-20;
	X[2]=25; Y[2]=25; Z[2]=-20;
	X[3]=25; Y[3]=25; Z[3]=-10;
	X[4]=10; Y[4]=10; Z[4]=-10;
	X[5]=10; Y[5]=10; Z[5]=-20;
	X[6]=25; Y[6]=10; Z[6]=-20;
	X[7]=25; Y[7]=10; Z[7]=-10;

	glColor3f(0,1,0);

	DrawCube(X,Y,Z);


	glFlush();
}
int main(int argc,char **argv)
{
	X=new GLfloat[8];
	Y=new GLfloat[8];
	Z=new GLfloat[8];
```

```
        vx=0.2;
        vy=0.2;
        vz=1;

        xwmin=-30;
        xwmax=30;
        ywmin=-30;
        ywmax=30;
        znear=-5;
        zfar=-30;

        glutInit(&argc,argv);
        glutInitWindowPosition(50,50);
        glutInitWindowSize(640,480);
        glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
        glutCreateWindow("Oblique parallel projection");

        glClearColor(1,1,1,0);

        glMatrixMode(GL_MODELVIEW);
        gluLookAt(0,0,0,0,0,-1,0,1,0);

        glMatrixMode(GL_PROJECTION);

        glOrtho(xwmin,xwmax,ywmin,ywmax,-znear,-zfar);

        shearingMatrix=new GLfloat[16];

        shearingMatrix[0]=1;
        shearingMatrix[1]=0;
        shearingMatrix[2]=0;
        shearingMatrix[3]=0;
        shearingMatrix[4]=0;
        shearingMatrix[5]=1;
        shearingMatrix[6]=0;
        shearingMatrix[7]=0;
        shearingMatrix[8]=-vx/vz;
        shearingMatrix[9]=-vy/vz;
        shearingMatrix[10]=1;
        shearingMatrix[11]=0;
        shearingMatrix[12]=znear*vx/vz;  shearingMatrix[13]=znear*vy/vz;
        shearingMatrix[14]=0;
        shearingMatrix[15]=1;

        glMultMatrixf(shearingMatrix);

        glutDisplayFunc(display);
        glutMainLoop();
        return 0;
}
```

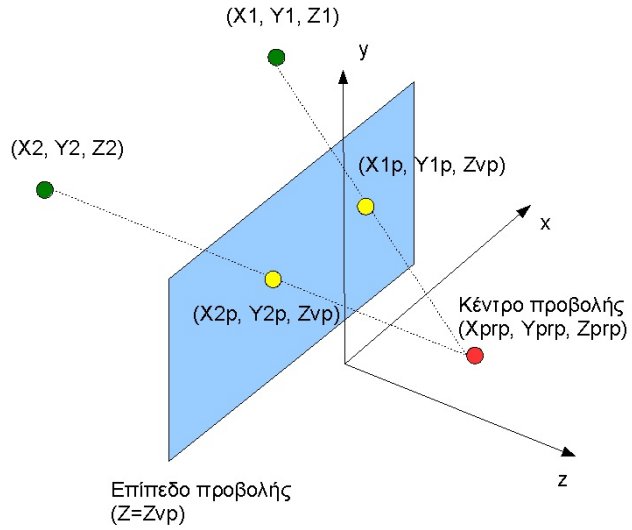


4.5 Προοπτική προβολή

Υπάρχουν περιπτώσεις στις οποίες δεν επιθυμούμε τη διατήρηση των αναλογιών στις διαστάσεις των αντικειμένων στη σκηνή κατά την προβολής τους και μας ενδιαφέρει η απόδοση ρεαλιστικών σκηνών. Με τον όρο «απόδοση ρεαλιστικών σκηνών», αναφερόμαστε στην εφαρμογή κανόνων της Οπτικής για το σχηματισμό του τελικού αποτελέσματος. Αυτόν ακριβώς το ρόλο εκτελεί η προοπτική προβολή.

Στη προοπτική προβολή, τα σημεία της σκηνής, αντί να προβάλλονται στο επίπεδο προβολής ακολουθώντας παράλληλες δέσμες , ακολουθούν δέσμες οι οποίες συγκλίνουν σε ένα κοινό σημείο, το **κέντρο προβολής**, όπως φαίνεται και στο Σχ. 4.5. Συνεπώς, για κάθε σημείο ορίζεται και μια δέσμη με διαφορετική κλίση. Η τομή κάθε δέσμης με το επίπεδο προβολής καθορίζει τη θέση που αντιστοιχεί για το εκάστοτε σημείο πάνω στο επίπεδο.

Η προοπτική προβολή προσωμοιώνει τις αρχές στις οποίες βασίζεται ο σχηματισμός του ειδώλου μιας σκηνής σε κάμερες με συγκλίνοντες φακούς, καθώς και τον τρόπο με τον οποίο σχηματίζεται το είδωλο στο ανθρώπινο μάτι. Αποτέλεσμα του κανόνα αντιστοίχισης της προοπτικής προβολής είναι το ότι, αντικείμενα που βρίσκονται μακρινότερα από τον παρατηρητή σχηματίζουν μικρότερο είδωλο στο επίπεδο προβολής σε σχέση με αντικείμενα του ίδιου μεγέθους που βρίσκονται κοντύτερα στον θεατή.

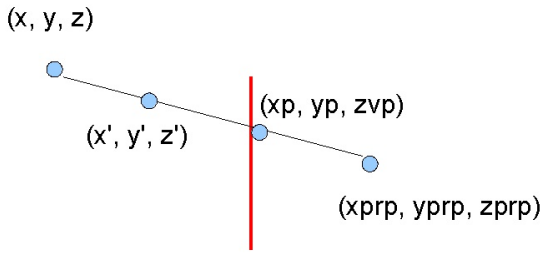


Σχ. 4.5: Αρχή λειτουργίας της προοπτικής προβολής

Προφανώς η απόσταση του κέντρου προβολής από το επίπεδο προβολής (**εστιακή απόσταση**) παίζει ρόλο ως προς το τελικό αποτέλεσμα. Συγκεκριμένα, μικρή εστιακή απόσταση συνεπάγεται μεγάλη γωνία κάλυψης της σκηνής και προσωμοιώνει ευρυγώνιους φακούς. Στην περίπτωση αυτή το προοπτικό φαινόμενο ενισχύεται και κοντινά αντικείμενα προβάλλονται με αρκετά μεγάλες διαστάσεις σε σχέση με μακρινά αντικείμενα του ίδιου μεγέθους. Μεγάλη εστιακή απόσταση συνεπάγεται μικρή γωνία κάλυψης και προσομοιώνει τους οξυγώνιους φακούς. Στην περίπτωση αυτή, το προοπτικό φαινόμενο παρουσιάζει μικρότερη βαρύτητα. Στην περίπτωση που η εστιακή απόσταση απειρίζεται, προκύπτει η παράλληλη προβολή.

Αναζητούμε το σημείο (x_p, y_p, z_{vp}) του επιπέδου προβολής $z = z_{vp}$, στο οποίο αντιστοιχίζεται ένα σημείο της σκηνής με συντεταγμένες σκηνής (x, y, z).

Έστω το σύνολο όλων των σημείων (x', y', z') που περιέχονται στη δέσμη που ορίζει το σημείο της σκηνής (x, y, z) και το κέντρο προβολής $(x_{prp}, y_{prp}, z_{prp})$ (Σχ.).



Σχ. 4.6: Σημεία που βρίσκονται επί της ίδιας δέσμης στην προοπτική προβολή

Το σύνολο των συντεταγμένων που ανήκουν στη δέσμη μπορεί να περιγραφεί με τις παρακάτω παραμετρικές εξισώσεις:

$$\begin{aligned} x' &= x - (x - x_{prp}) \cdot u \\ y' &= y - (y - y_{prp}) \cdot u \quad 0 \le u \le 1 \\ z' &= z - (z - x_{prp}) \cdot u \end{aligned}$$

Οι συντεταγμένες x_p, y_p στις οποίες προβάλλεται το σημείο (x, y, z) προκύπτουν, βρίσκοντας την τιμή της παραμέτρου u για την οποία $z' = z_{vp}$:

$$u_p = \frac{z_{vp} - z}{z_{prp} - z}$$

Επομένως οι συντεταγμένες της προβολής του σημείου προκύπτουν από τις σχέσεις:

$$x_p = x \cdot \left(\frac{z_{prp} - z_{vp}}{z_{prp} - z} \right) + x_{prp} \cdot \left(\frac{z_{vp} - z}{z_{prp} - z} \right)$$

$$y_p = y \cdot \left(\frac{z_{prp} - z_{vp}}{z_{prp} - z} \right) + y_{prp} \cdot \left(\frac{z_{vp} - z}{z_{prp} - z} \right)$$

Η περιγραφή των παραπάνω σχέσεων με τη μορφή μητρώου μετασχηματισμού είναι αδύνατη, καθώς η συντεταγμένη z του σημείου της σκηνής εμφανίζεται στον παρονομαστή των συντελεστών. Ωστόσο, μπορούμε να καταφύγουμε σε μια εναλλακτική περιγραφή, εκμεταλλευόμενοι την αναπαράσταση σε μορφή ομογενών συντεταγμένων. Στην περίπτωση αυτή για κάθε σημείο της σκηνής (x, y, z), αντί για τις συντεταγμένες προβολής x_p, y_p, εξάγουμε τις συντεταγμένες x_h, y_h που σχετίζονται με τις συντεταγμένες προβολής βάσει των εξισώσεων.

$$x_{vp} = \frac{x_h}{h}$$

$$y_{vp} = \frac{y_h}{h}$$

όπου h η ομογενής παράμετρος, η οποία, στην περίπτωσή μας, παίρνει την τιμή του παρονομαστή

$$h = z_{prp} - z$$

και έχει διαφορετική τιμή για κάθε διαφορετικό σημείο της σκηνής (x, y, z).

Συνεπώς οι συντεταγμένες x_h, y_h προκύπτουν από τις σχέσεις

$$x_h = x \cdot (z_{prp} - z_{vp}) + x_{prp} \cdot (z_{vp} - z)$$
$$y_h = y \cdot (z_{prp} - z_{vp}) + y_{prp} \cdot (z_{vp} - z)$$

που αναπαρίστανται σε μορφή μητρώου ως εξής

$$\begin{bmatrix} x_h \\ y_h \\ z_h \\ h \end{bmatrix} = M_{pers} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

όπου:

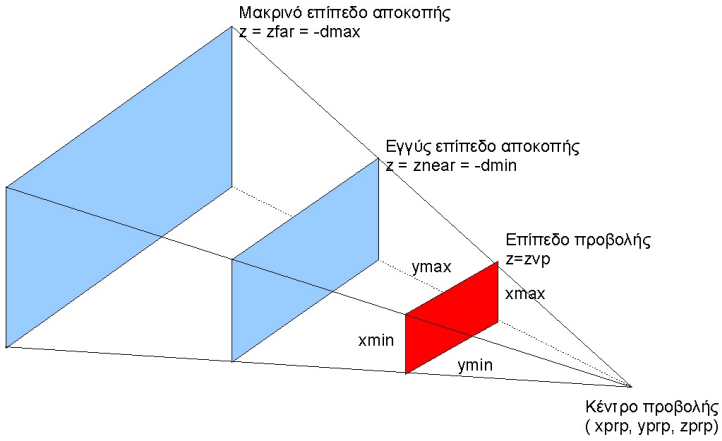
$$Mpers = \begin{bmatrix} z_{prp} - z_{vp} & 0 & -x_{prp} & x_{prp} \cdot z_{vp} \\ 0 & z_{prp} - z_{vp} & -y_{prp} & y_{prp} \cdot z_{vp} \\ 0 & 0 & s_z & t_z \\ 0 & 0 & -1 & z_{prp} \end{bmatrix}$$

Οι τιμές s_z και t_z λειτουργούν ως συντελεστές κλιμάκωσης και μετατόπισης των συντεταγμένων z. Εισάγονται στο μητρώο προοπτικής προβολής ούτως ώστε να αποφευχθεί η αλλοίωση στις τιμές βάθους των σημείων της σκηνής. Οι ακριβείς τιμές των συντελεστών s_z και t_z εξαρτώνται από τα όρια κανονικοποίησης που επιλέγουμε για τις συντεταγμένες βάθους. Ο προσδιορισμός τους παρουσιάζεται στο Παράρτημα ΙΙ.

Συνεπώς, στην περίπτωση της προοπτικής προβολής, ορίζουμε ένα μητρώο που για κάθε σημείο (x, y, z) εξάγει τις αντίστοιχες τιμές (x_h, y_h, z_h). Κατόπιν, διαιρώντας τις τελευταίες παραμέτρους με h, εξάγουμε τις συντεταγμένες προβολής x_p, y_p.

Όπως και στην περίπτωση της παράλληλης προβολής, έτσι και στην προοπτική προβολή, ο ορισμός της στην OpenGL είναι συνδεδεμένος με τη διαδικασία αποκοπής. Συγκεκριμένα, με τα επίπεδα αποκοπής που επιλέγουμε, καθορίζουμε το άνω, το κάτω, το αριστερό και το δεξιό όριο του επιπέδου προβολής, όπως φαίνεται στο Σχ. 4.7.

Δεδομένου ότι στην προοπτική προβολή όλες οι ακτίνες αναχωρούν από το κέντρο προβολής με διαφορετικές κλίσεις, απομονώνουμε ένα τμήμα που δεν έχει όρια παραλληλεπιπέδου. Αντίθετα, ο χώρος αποκοπής προσδιορίζεται από τα όρια μιας **κόλουρου πυραμίδας (frustum)**. Η νοητή κορυφή της πυραμίδας ταυτίζεται με το κέντρο προβολής και οι έδρες της καθορίζονται από το εγγύς και το μακρινό επίπεδο αποκοπής.

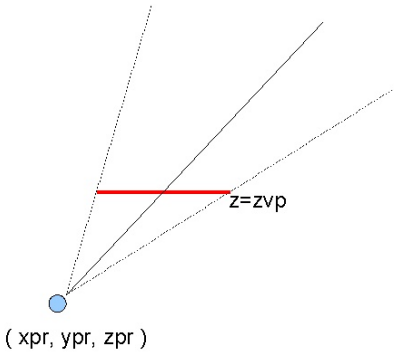


Σχ 4.7: Τα επίπεδα αποκοπής στην προοπτική προβολή ορίζουν μια κόλουρο πυραμίδα η οποία περικλείει το παρατηρούμενο τμήμα της σκηνής

4.6 Συμμετρική και πλάγια προοπτική προβολή

Στην προοπτική προβολή ορίζουμε ως **οπτικό άξονα** την ευθεία που συνδέει το κέντρο προβολής $\left(x_{prp}, y_{prp}, z_{prp}\right)$ με το κέντρο του επιπέδου προβολής $\left(\frac{xw_{\min}+xw_{\max}}{2}, \frac{yw_{\min}+yw_{\max}}{2}, z_{vp}\right)$

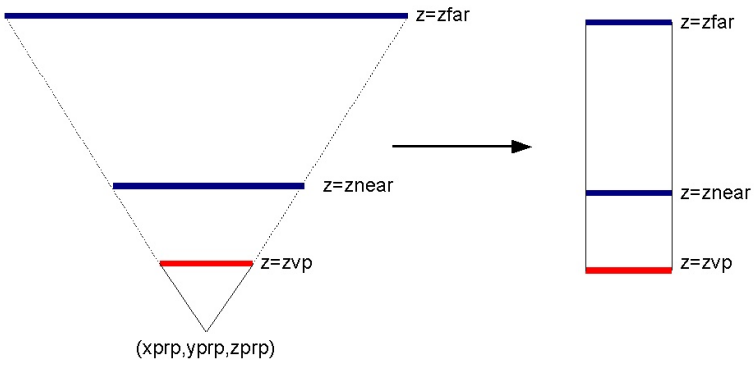
Στην περίπτωση που ο οπτικός άξονας της προοπτικής προβολής είναι κάθετος στο επίπεδο προβολής έχουμε μια **συμμετρική προοπτική προβολή**. Ωστόσο αυτή η συνθήκη δεν είναι απαραίτητη. Στην περίπτωση που ο άξονας της προβολής δεν είναι κάθετος στο επίπεδο προβολής παράγουμε μια **πλάγια προοπτική προβολή** όπως φαίνεται στο Σχ. 4.8 .



Σχ. 4.8:Πλάγια προοπτική προβολή

Προφανώς, το αν παράγουμε μια συμμετρική ή πλάγια προοπτική προβολή εξαρτάται από τη σχετική θέση του κέντρου προβολής και του επιπέδου προβολής. Συγκεκριμένα, η έκταση και τα όρια του επιπέδου προβολής εξαρτώνται από τη θέση των επιπέδων αποκοπής. Επομένως, το αν θα σχηματιστεί μια προοπτική ή πλάγια προοπτική προβολή καθορίζεται από τις τιμές που θα αναθέσουμε στο άνω, κάτω, αριστερό και δεξιό επίπεδο αποκοπής.

Χρησιμοποιώντας το μετασχηματισμό προοπτικής προβολής εξάγουμε τις ομογενείς συντεταγμένες (x_h, y_h, z_h, h), οι οποίες, διαιρούμενες με h, εξάγουν τις συντεταγμένες προβολής. Οι τιμές των συντεταγμένων προβολής εμπεριέχονται σε ένα ορθογώνιο παραλληλεπίπεδο που τα όριά του καθορίζονται από τα επίπεδα αποκοπής, όπως φαίνεται στο Σχ. 4.10. Ουσιαστικά, η προοπτική προβολή είναι ένας μετασχηματισμός που αντιστοιχίζει τα περιεχόμενα μιας συμμετρικής ή ασύμμετρης πυραμίδας στο εσωτερικό ενός ορθογωνίου παραλληλεπιπέδου.



Σχ 4.10: Αποτέλεσμα του μετασχηματισμού προοπτικής προβολής. Η σκηνή που περιπλείεται από τα όρια μιας συμμετρικής ή ασύμμετρης πυραμίδας μετασχηματίζεται σε ένα παραλληλεπίπεδο. Το παραλληλεπίπεδο αυτό περικλείεται από τα επίπεδα αποκοπής που ορίζουν την έκταση του επιπέδου προβολής.

Το μητρώο που εκτελεί μετασχηματισμό **συμμετρικής ή πλάγιας προοπτικής προβολής** και μετασχηματισμό κανονικοποίησης στο εύρος τιμών $[-1,1]$ έχει τη μορφή

$$M_{norm,pers} = \begin{bmatrix} \frac{-2 \cdot z_{near}}{xw_{\max} - xw_{\min}} & 0 & \frac{xw_{\max} + xw_{\min}}{xw_{\max} - xw_{\min}} & 0 \\ 0 & -\frac{2 \cdot z_{near}}{yw_{\max} - yw_{\min}} & \frac{yw_{\max} + yw_{\min}}{yw_{\max} - yw_{\min}} & 0 \\ 0 & 0 & \frac{z_{near} + z_{far}}{z_{near} - z_{far}} & \frac{2 \cdot z_{near} \cdot z_{far}}{z_{near} - z_{far}} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

Η διαδικασία προσδιορισμού του παραπάνω μητρώου παρουσιάζεται στο Παράρτημα.

Σε ό,τι αφορά την κανονικοποίηση των τιμών βάθους, η τιμή -1 αντιστοιχεί σε σημεία που βρίσκονται στο εγγύς επίπεδο αποκοπής και η κανονικοποιημένη τιμή βάθους 1 στα σημεία που βρίσκονται στο μακρινό επίπεδο αποκοπής. Με την εφαρμογή του παραπάνω μητρώου προκύπτουν οι συντεταγμένες (x_h, y_h, z_h, h) και κατόπιν οι κανονικοποιημένες συντεταγμένες προβολής με τη διαίρεση με $h = -z$.

Στην περίπτωση **συμμετρικής προοπτικής προβολής**, μπορούμε να αναπαραστήσουμε τους συντελεστές του μητρώου προβολής/κανονικοποίησης συναρτήσει της κατακόρυφης γωνίας κάλυψης θ και της αναλογίας πλάτους προς ύψος *aspectRatio* του επιπέδου προβολής ως εξής:

$$M_{norm,symm,pers} = \begin{bmatrix} \frac{\cot\left(\frac{\theta}{2}\right)}{aspectRatio} & 0 & 0 & 0 \\ 0 & \cot\left(\frac{\theta}{2}\right) & 0 & 0 \\ 0 & 0 & \frac{z_{near} + z_{far}}{z_{near} - z_{far}} & \frac{2 \cdot z_{near} \cdot z_{far}}{z_{near} - z_{far}} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

Η OpenGL, για τον ορισμό μιας προοπτικής προβολής, διαθέτει δύο εντολές: την ***glFrustum*** και την ***gluPerspective***. Και οι δύο εντολές ακολουθούν τις παραδοχές που αναφέραμε προηγουμένως:
α) Θέτουν το κέντρο προβολής στην αρχή του συστήματος συντεταγμένων παρατηρητή.
β) Ταυτίζουν το επίπεδο προβολής με το εγγύς επίπεδο αποκοπής

Με την εντολή ***glFrustum***, ο προγραμματιστής μπορεί να ορίσει μια **συμμετρική ή πλάγια προοπτική προβολή**:

glFrustum(GLdouble xwmin, GLdouble xwmax, GLdouble ywmin, GLdouble ywmax, GLdouble dnear, GLdouble dfar);

όπου *xwmin*, *xwmax*, *ywmin* και *ywmax* το αριστερό, δεξιό, κάτω και άνω επίπεδο αποκοπής αντίστοιχα. Οι θετικοί αριθμοί *dnear* και *dfar* καθορίζουν το εγγύς και μακρινό επίπεδο αποκοπής. $\left(z_{near} = -d_{near}, z_{far} = -d_{far}\right)$

Με τα παραπάνω δεδομένα, η εντολή ***glFrustum*** δημιουργεί το μητρώο συμμετρικής/πλάγιας προοπτικής προβολής και μετασχηματισμού κανονικοποίησης. Η προβολή είναι συμμετρική προοπτική ή πλάγια προοπτική, ανάλογα με τις τιμές που αναθέτουμε στα επίπεδα αποκοπής.

Με την εντολή ***gluPerspective*** ορίζουμε μια **συμμετρική προοπτική προβολή**:

gluPerspective(GLdouble angle, GLdouble aspectRatio, GLdouble dnear, GLdouble dfar);

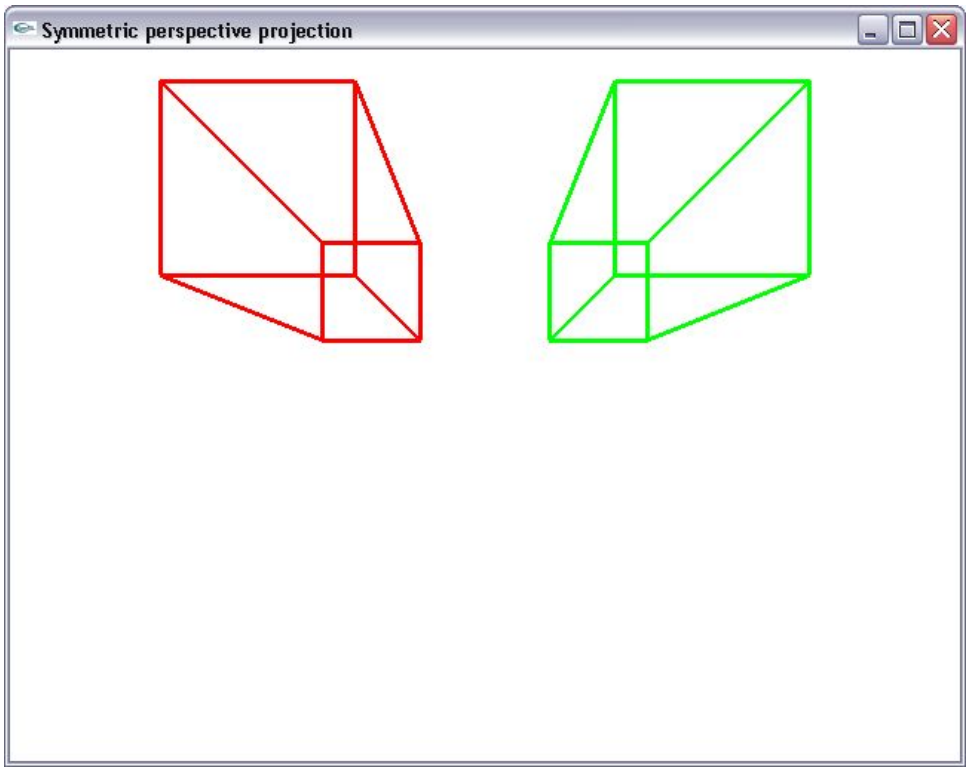
όπου *angle* η κατακόρυφη γωνία παρατήρησης σε μοίρες, *aspectRatio* ο λόγος πλάτους προς ύψος του επιπέδου προβολής και *dnear* , *dfar* θετικές τιμές που καθορίζουν το εγγύς και μακρινό επίπεδο αποκοπής $\left(z_{near} = -d_{near}, z_{far} = -d_{far}\right)$. Με τα δεδομένα αυτά, η εντολή κατασκευάζει το μητρώο συμμετρικής προοπτικής προβολής.

Παράδειγμα: Συμμετρική προοπτική προβολή

```
#include <glut.h>

GLfloat *X;
GLfloat *Y;
GLfloat *Z;
GLdouble angle=140;
GLdouble aspectRatio=1.33333333;
GLdouble dnear=8;
GLdouble dfar=30;
void DrawCube(GLfloat *X, GLfloat *Y, GLfloat *Z)
{

      //    1--------------2
      //    |\              |\
      //    |  0--------------3
      //    |  |            | |
      //    |  |            | |
      //    |  |            | |
      //    5--|------------6 |
      //     \ |             \|
      //       4--------------7
```

```
	glBegin(GL_LINE_LOOP);
	glVertex3f(X[0],Y[0],Z[0]);
	glVertex3f(X[1],Y[1],Z[1]);
	glVertex3f(X[2],Y[2],Z[2]);
	glVertex3f(X[3],Y[3],Z[3]);
	glEnd();

	glBegin(GL_LINE_LOOP);
	glVertex3f(X[4],Y[4],Z[4]);
	glVertex3f(X[5],Y[5],Z[5]);
	glVertex3f(X[6],Y[6],Z[6]);
	glVertex3f(X[7],Y[7],Z[7]);
	glEnd();

	glBegin(GL_LINES);
	glVertex3f(X[0],Y[0],Z[0]);
	glVertex3f(X[4],Y[4],Z[4]);
	glVertex3f(X[1],Y[1],Z[1]);
	glVertex3f(X[5],Y[5],Z[5]);
	glVertex3f(X[2],Y[2],Z[2]);
	glVertex3f(X[6],Y[6],Z[6]);
	glVertex3f(X[3],Y[3],Z[3]);
	glVertex3f(X[7],Y[7],Z[7]);
	glEnd();
}

void display()
{
	glClear(GL_COLOR_BUFFER_BIT);

	X[0]=-10; Y[0]=25; Z[0]=-10;
	X[1]=-10; Y[1]=25; Z[1]=-20;
	X[2]=-25; Y[2]=25; Z[2]=-20;
	X[3]=-25; Y[3]=25; Z[3]=-10;
	X[4]=-10; Y[4]=10; Z[4]=-10;
	X[5]=-10; Y[5]=10; Z[5]=-20;
	X[6]=-25; Y[6]=10; Z[6]=-20;
	X[7]=-25; Y[7]=10; Z[7]=-10;

	glColor3f(1,0,0);
	DrawCube(X,Y,Z);

	X[0]=10; Y[0]=25; Z[0]=-10;
	X[1]=10; Y[1]=25; Z[1]=-20;
	X[2]=25; Y[2]=25; Z[2]=-20;
	X[3]=25; Y[3]=25; Z[3]=-10;
	X[4]=10; Y[4]=10; Z[4]=-10;
	X[5]=10; Y[5]=10; Z[5]=-20;
	X[6]=25; Y[6]=10; Z[6]=-20;
	X[7]=25; Y[7]=10; Z[7]=-10;

	glColor3f(0,1,0);

	DrawCube(X,Y,Z);


	glFlush();
}
```

```
int main(int argc,char **argv)
{
	X=new GLfloat[8];
	Y=new GLfloat[8];
	Z=new GLfloat[8];

	glutInit(&argc,argv);
	glutInitWindowPosition(50,50);
	glutInitWindowSize(640,480);
	glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
	glutCreateWindow("Symmetric perspective projection");

	glClearColor(1,1,1,0);

	glMatrixMode(GL_MODELVIEW);
	gluLookAt(0,0,0,0,0,-1,0,1,0);

	glMatrixMode(GL_PROJECTION);

	gluPerspective(angle,aspectRatio,dnear,dfar);

	glutDisplayFunc(display);
	glutMainLoop();
	return 0;
}
```

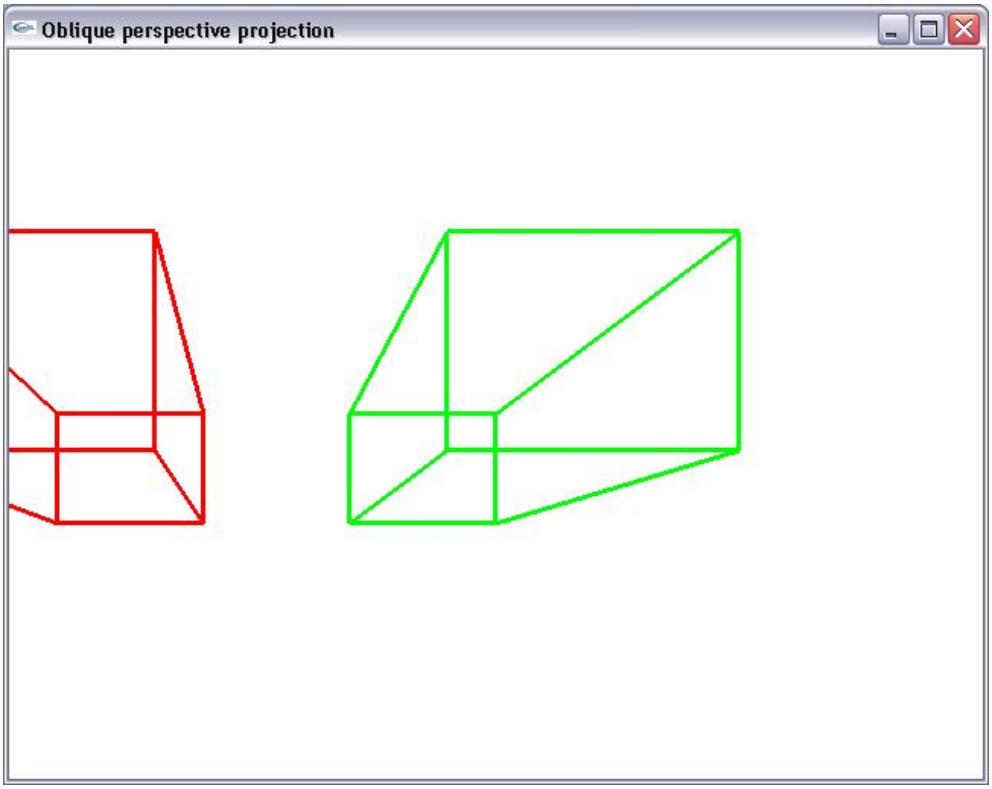


Παράδειγμα: Πλάγια προοπτική προβολή

```
#include <glut.h>
```

```
GLfloat *X;
GLfloat *Y;
GLfloat *Z;

GLdouble xwmin,xwmax,ywmin,ywmax,dnear,dfar;

void DrawCube(GLfloat *X, GLfloat *Y, GLfloat *Z)
{


      //    1--------------2
      //    |\              |\
      //    |  0--------------3
      //    |  |            | |
      //    |  |            | |
      //    |  |            | |
      //    5--|-----------6 |
      //     \ |             \|
      //       4--------------7

      glBegin(GL_LINE_LOOP);
      glVertex3f(X[0],Y[0],Z[0]);
      glVertex3f(X[1],Y[1],Z[1]);
      glVertex3f(X[2],Y[2],Z[2]);
      glVertex3f(X[3],Y[3],Z[3]);
      glEnd();

      glBegin(GL_LINE_LOOP);
      glVertex3f(X[4],Y[4],Z[4]);
      glVertex3f(X[5],Y[5],Z[5]);
      glVertex3f(X[6],Y[6],Z[6]);
      glVertex3f(X[7],Y[7],Z[7]);
      glEnd();

      glBegin(GL_LINES);
      glVertex3f(X[0],Y[0],Z[0]);
      glVertex3f(X[4],Y[4],Z[4]);
      glVertex3f(X[1],Y[1],Z[1]);
      glVertex3f(X[5],Y[5],Z[5]);
      glVertex3f(X[2],Y[2],Z[2]);
      glVertex3f(X[6],Y[6],Z[6]);
      glVertex3f(X[3],Y[3],Z[3]);
      glVertex3f(X[7],Y[7],Z[7]);
      glEnd();
}

void display()
{
      glClear(GL_COLOR_BUFFER_BIT);

      X[0]=-5; Y[0]=25; Z[0]=-10;
      X[1]=-5; Y[1]=25; Z[1]=-20;
      X[2]=-20; Y[2]=25; Z[2]=-20;
      X[3]=-20; Y[3]=25; Z[3]=-10;
      X[4]=-5; Y[4]=10; Z[4]=-10;
      X[5]=-5; Y[5]=10; Z[5]=-20;
      X[6]=-20; Y[6]=10; Z[6]=-20;
      X[7]=-20; Y[7]=10; Z[7]=-10;

      glColor3f(1,0,0);
      DrawCube(X,Y,Z);
```

```
	X[0]=10; Y[0]=25; Z[0]=-10;
	X[1]=10; Y[1]=25; Z[1]=-20;
	X[2]=25; Y[2]=25; Z[2]=-20;
	X[3]=25; Y[3]=25; Z[3]=-10;
	X[4]=10; Y[4]=10; Z[4]=-10;
	X[5]=10; Y[5]=10; Z[5]=-20;
	X[6]=25; Y[6]=10; Z[6]=-20;
	X[7]=25; Y[7]=10; Z[7]=-10;

	glColor3f(0,1,0);

	DrawCube(X,Y,Z);


	glFlush();
}
int main(int argc,char **argv)
{
	X=new GLfloat[8];
	Y=new GLfloat[8];
	Z=new GLfloat[8];

	glutInit(&argc,argv);
	glutInitWindowPosition(50,50);
	glutInitWindowSize(640,480);
	glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
	glutCreateWindow("Oblique perspective projection");

	glClearColor(1,1,1,0);

	glMatrixMode(GL_MODELVIEW);
	gluLookAt(0,0,0,0,0,-1,0,1,0);

	glMatrixMode(GL_PROJECTION);


	//The clipping planes define an oblique perspective projection
	xwmin=-10;
	xwmax=30;
	ywmin=-10;
	ywmax=30;
	dnear=8;
	dfar=30;

	glFrustum(xwmin,xwmax,ywmin,ywmax,dnear,dfar);

	glutDisplayFunc(display);
	glutMainLoop();
	return 0;
}
```



4.7 Μετασχηματισμός παρατήρησης στις τρεις διαστάσεις

Σε τρισδιάστατες σκηνές, μετά το μετασχηματισμό παράλληλης ή προοπτικής προβολής και τη διαδικασία κανονικοποίησης, έπεται η διαδικασία μετασχηματισμού παρατήρησης, ούτως ώστε οι κανονικοποιημένες συντεταγμένες να αντιστοιχιστούν σε συντεταγμένες σκηνής. Ο τρισδιάστατος μετασχηματισμός παρατήρησης εκτελείται με τρόπο παρόμοιο με αυτόν στις δύο διαστάσεις. Ωστόσο, στον τρισδιάστατο μετασχηματισμό παρατήρησης, πέραν των συντεταγμένων x και y, μετασχηματίζονται και οι τιμές βάθους. Η OpenGL μετασχηματίζει τις κανονικοποιημένες τιμές βάθους από το εύρος τιμών $[-1,1]$ σε ένα σταθερό εύρος τιμών $[0,1]$. Συνεπώς, με την εφαρμογή των κανόνων αναλογίας που αναπτύχθηκαν στο Κεφάλαιο 2, το μητρώο τρισδιάστατου μετασχηματισμού παρατήρησης αποκτά τη μορφή:

$$M_{norm,view} = \begin{bmatrix} \frac{xu_{\max} - xu_{\min}}{2} & 0 & 0 & \frac{xu_{\max} + xu_{\min}}{2} \\ 0 & \frac{yu_{\max} - yu_{\min}}{2} & 0 & \frac{yu_{\max} + yu_{\min}}{2} \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Οι τιμές βάθους αξιοποιούνται στην καταστολή κρυμμένων επιφανειών, όπως θα δούμε στο επόμενο κεφάλαιο.

4.8 Το ενοποιημένο διάγραμμα μετασχηματισμού συντεταγμένων

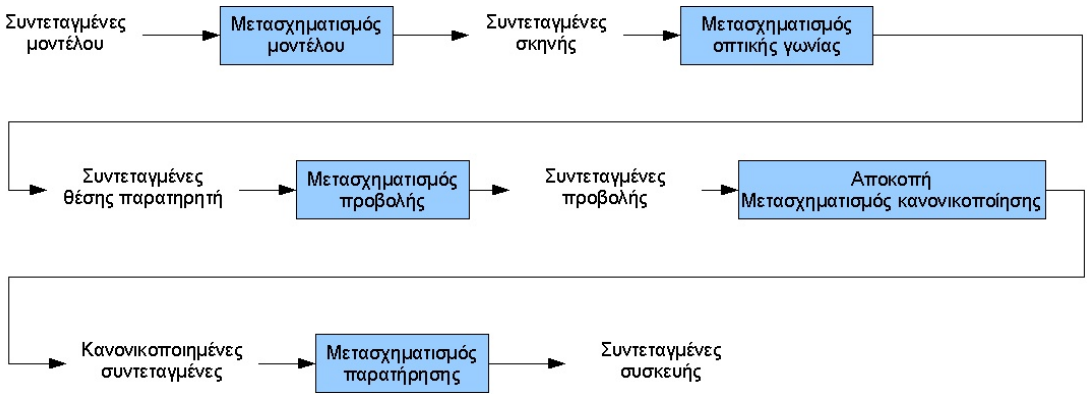
Όλοι οι μετασχηματισμοί που παρουσιάστηκαν στα προηγούμενα κεφάλαια συμμετέχουν σε μια ενοποιημένη διαδικασία μετασχηματισμού συντεταγμένων που ακολουθεί η OpenGL και που παρουσιάζεται στο Σχ. 4.11.

Από τη φάση της δήλωσης ενός γεωμετρικού μοντέλου έως τη φάση απεικόνισής του στη συσκευή εξόδου, οι συντεταγμένες του υφίστανται διαδοχικές επεξεργασίες, που καθορίζονται από τα μητρώα μετασχηματισμού που προαναφέραμε. Όλες οι ενδιάμεσες διαδικασίες ενσωματώνονται στα μητρώα μετασχηματισμού μοντέλου και προβολής (μετασχηματισμός μοντέλου, μετασχηματισμός οπτικής γωνίας, μετασχηματισμός προβολής, αποκοπή, κανονικοποίηση και μετασχηματισμός παρατήρησης).

Στο πρώτο στάδιο, το μητρώο μετασχηματισμού μοντέλου δέχεται ως είσοδο τις συντεταγμένες των γεωμετρικών μοντέλων, επιτελεί το μετασχηματισμό μοντέλου και παράγει τις συντεταγμένες σκηνής. Κατόπιν, εκτελεί το μετασχηματισμό οπτικής γωνίας και ανάγει την περιγραφή της σκηνής στο σύστημα συντεταγμένων του παρατηρητή.

Στο επόμενο στάδιο, οι συντεταγμένες θέσης παρατηρητή υφίστανται επεξεργασία από το μητρώο προβολής. Αρχικά υφίστανται το μετασχηματισμό προβολής, εξάγοντας έτσι τις συντεταγμένες προβολής. Έπονται οι διαδικασίες αποκοπής και κανονικοποίησης που εξάγουν τις κανονικοποιημένες συντεταγμένες. Ακολουθεί το τελικό στάδιο του μετασχηματισμού παρατήρησης κατά το οποίο εξάγονται οι συντεταγμένες συσκευής.

Αυτή η αλληλουχία μετασχηματισμών εφαρμόζεται πάντοτε. Σε κάθε χρονική στιγμή, η μηχανή της OpenGL επιβάλλει την διαδοχή των παραπάνω μετασχηματισμών **σε κάθε σημείο ή γεωμετρικό σχήμα** που ορίζει ο προγραμματιστής. Με τη συνδυασμένη επίδραση των τρεχόντων μητρώων μοντέλου και προβολής, μετρατρέπει τις συντεταγμένες μοντέλου σε συντεταγμένες συσκευής.



Σχ. 4.11 : Ενοποιημένο διάγραμμα μετασχηματισμού συντεταγμένων της OpenGL