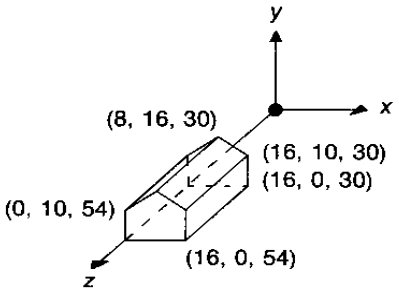
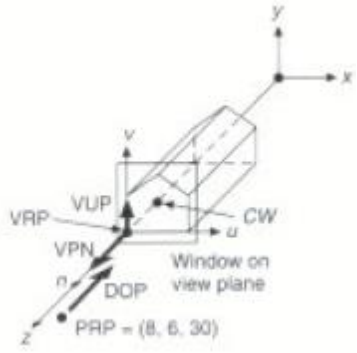
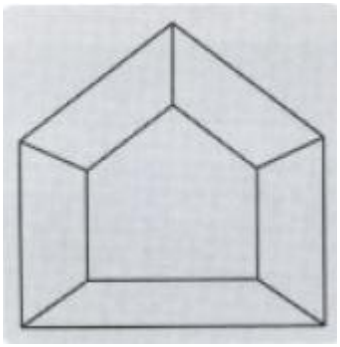
بسمه تعالی

مشخص کردن یک دید سه بعدی دلخواه

مثال: تصویر پرسپکتیو یک نقطه ای







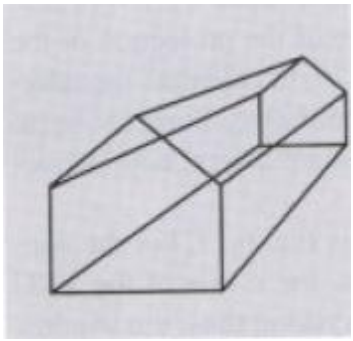
پارامتر دید	مقدار	توضیح
VRC (WC)	(۵۴و ۰ و ۰)	(x,y,z)
VPN (WC)	(۱و ۰و ۰)	محور z
VUP (WC)	(۰و ۱و ۰)	محور y
PRP (VRC)	(۳۰و ۶و ۸)	(u,v,n)
پنجره (VRC)	(۱۷و ۱– و ۱۷و ۱–)	$(u_{min}, u_{max}, v_{min}, v_{max})$
نوع تصویر	پرسپکتیو	

مثال: بصورتی دیگری هم می توان همان شکل را بدست آورد.

پارامتر دید	مقدار	توضیح

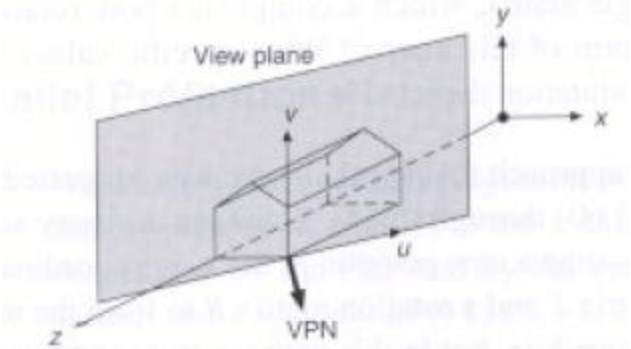
VRC (WC)	(۵۴ و۶ و۸)	(x,y,z)
VPN (WC)	(۱و ۰و ۰)	محور z
VUP (WC)	(۰و ۱و ۰)	محور y
PRP (VRC)	(۳۰و ۶و ۸)	(u,v,n)
پنجره (VRC)	(۱۱و ۷– و ۹و ۹–)	$(u_{min}, u_{max}, v_{min}, v_{max})$
نوع تصویر	پرسپکتیو	

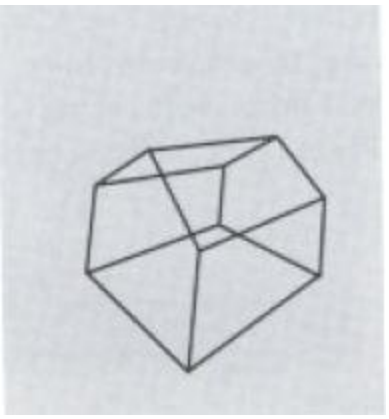
مثال: تصویر پرسپکتیو یک نقطه ای



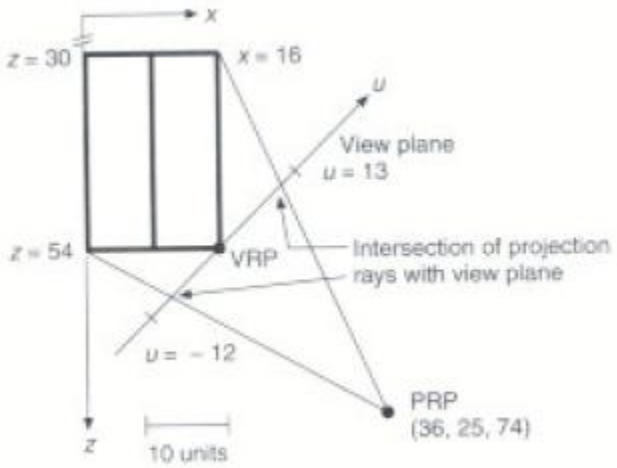
پارامتر دید	مقدار	توضیح
VRC (WC)	(۵۴ و۰ و۱۶)	(x,y,z)
VPN (WC)	(۱و ۰و ۰)	محور z
VUP (WC)	(۰و ۱و ۰)	محور y
PRP (VRC)	(۲۰و ۲۵و ۲۰)	(u,v,n)
پنجره (VRC)	(۳۵و ۵– و ۲۰و ۲۰–)	$(u_{min}, u_{max}, v_{min}, v_{max})$
نوع تصویر	پرسپکتیو	

مثال: تصویر پرسپکتیو دو نقطه ای



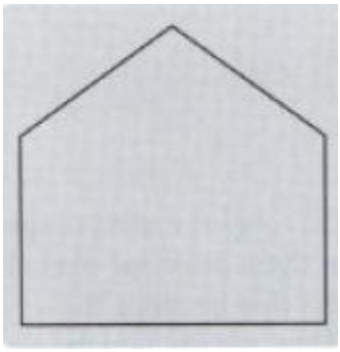


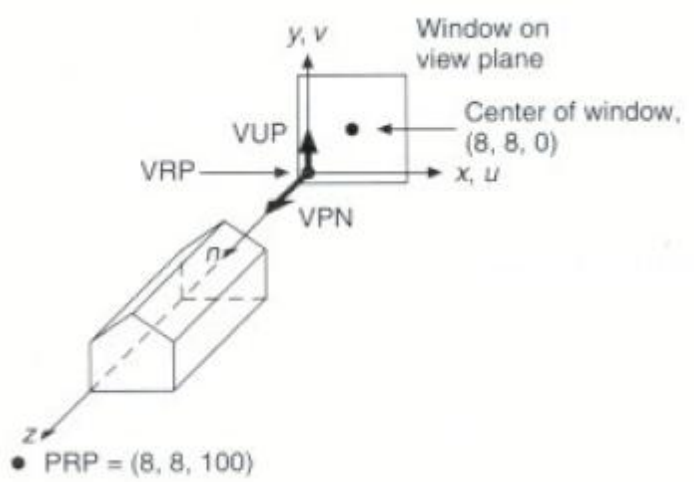
پارامتر دید	مقدار	توضیح
VRC (WC)	(۵۴ و ۰ و ۱۶)	(x,y,z)
VPN (WC)	(۱و ۰و ۱)	در صفحهٔ zx
VUP (WC)	(۰و ۱و ۰)	محور y
PRP (VRC)	(۰ ۲۰و ۲۵و ۰)	(u,v,n)
پنجره (VRC)	(۳۵و ۵– و ۲۰و ۲۰–)	(u_{min}, u_{max}, v_{min}, v_{max})
نوع تصویر	پرسپکتیو	



شکل ۱ نمای بالای خانه برای مشخص کردن اندازهٔ پنجرهٔ مناسب.

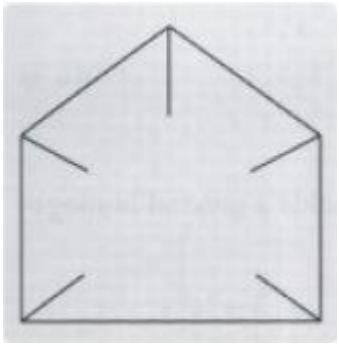
مثال: تصویر موازی- تصویر جلو





پارامتر دید	مقدار	توضیح
VRC (WC)	(۰ و ۰ و ۰)	(x,y,z)
VPN (WC)	(۱و ۰و ۰)	محور z
VUP (WC)	(۰و ۱و ۰)	محور y
PRP (VRC)	(۱۰۰و ۸و ۸)	(u,v,n)
پنجره (VRC)	(۱۷و ۱– و ۱۷و ۱–)	(u_{min}, u_{max}, v_{min}, v_{max})
نوع تصویر	موازی	

مثال: اضافه کردن صفحهٔ برش جلو و صفحهٔ برش عقب

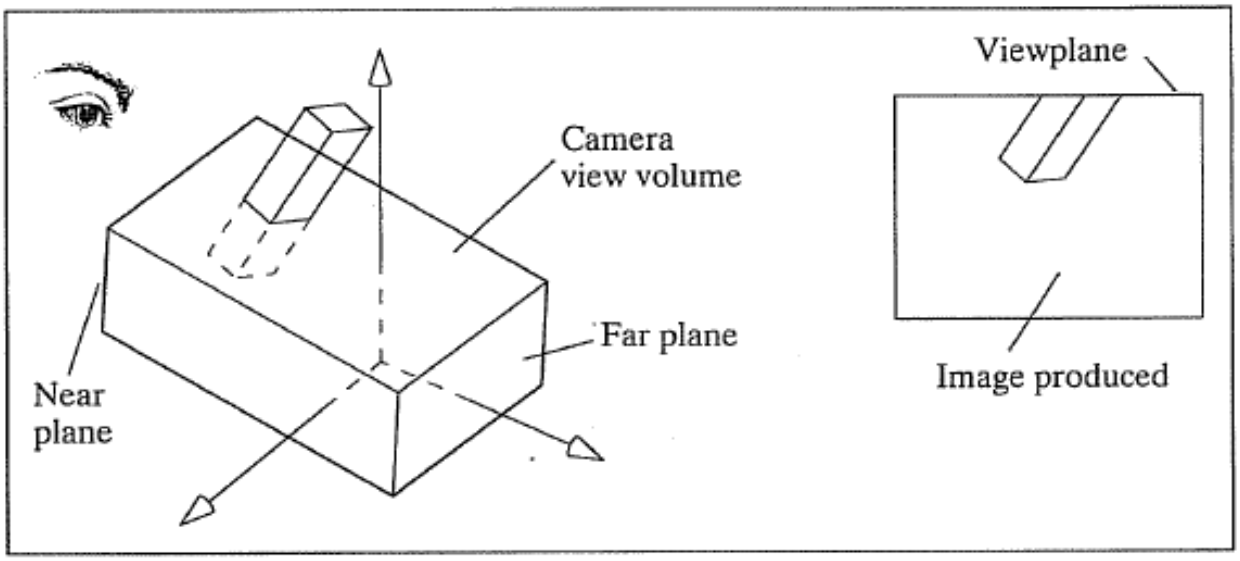


پارامتر دید	مقدار	توضیح
VRC (WC)	(۵۴و ۰ و ۰)	(x,y,z)
VPN (WC)	(۱و ۰و ۰)	محور z
VUP (WC)	(۰و ۱و ۰)	محور y
PRP (VRC)	(۳۰و ۶و ۸)	(u,v,n)
پنجره (VRC)	(۱۷و ۱– و ۱۷و ۱–)	(u_{min}, u_{max}, v_{min}, v_{max})
نوع تصویر	پرسپکتیو	

F (VRC)	+۱	n
B (VRC)	-۲۳	n

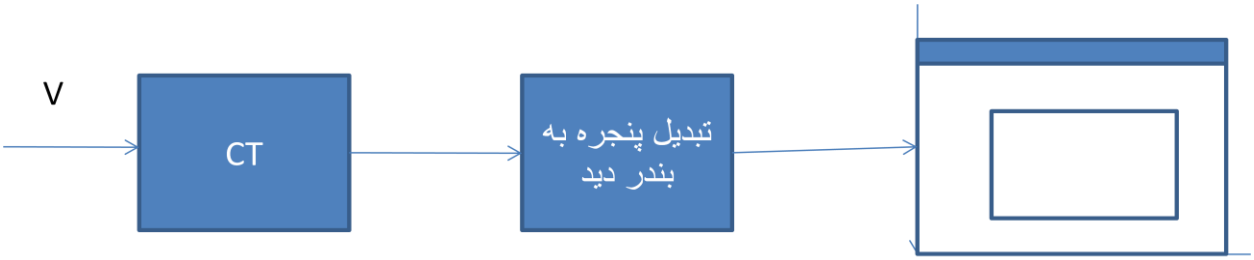
نگرش سه بعدی در OpenGL

تصور می کنیم که یک دوربین به منظره نگاه می کند. مکان دوربین همانند مکان PRP می باشد. دوربین به نقطه ای توجه دارد (look). باید مشخص کرد بالا چه سمتی است (VUP)

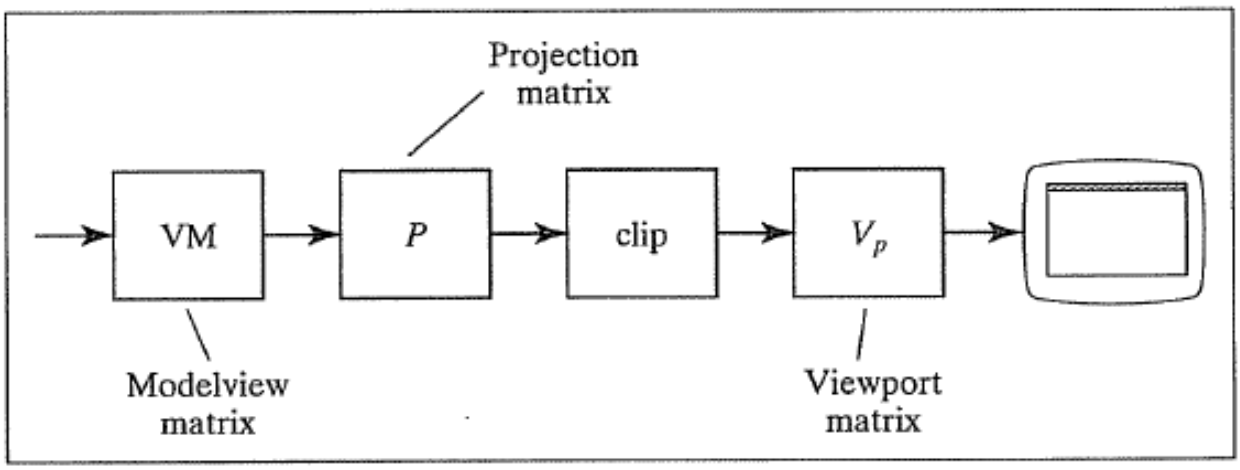


شکل ۲

خط لولهٔ OpenGL که تاکنون آشنا شدیم:



تصور جدید:



شکل ۳

هر رأس با ماتریسهای متفاوتی مواجه می شود:

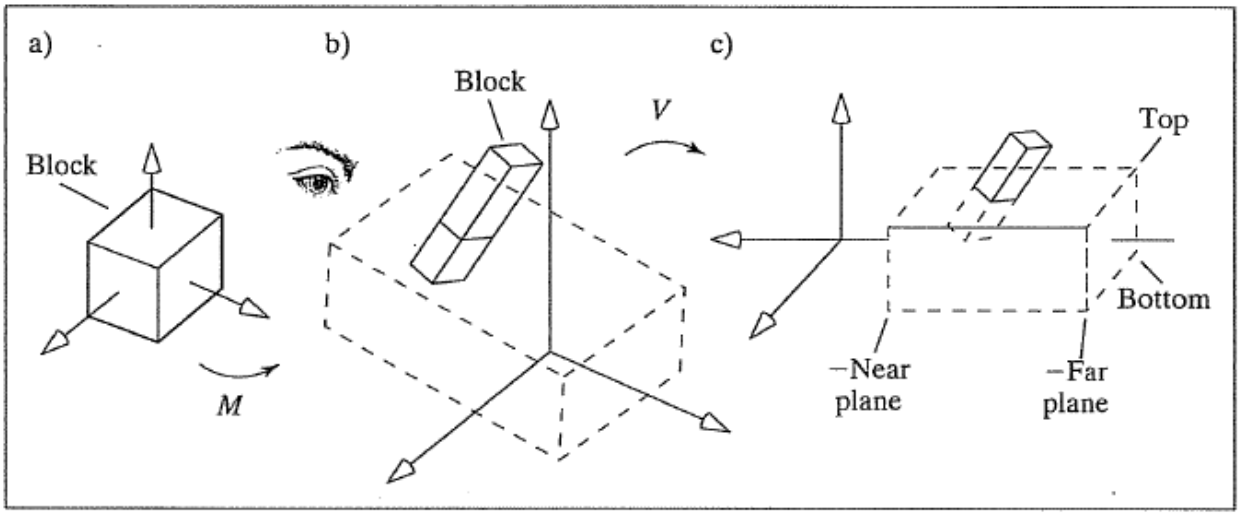
- ماتریس modelview
- ماتریس projection
- ماتریس viewport

ماتریس modelview همان ماتریسی است که CT می نامیدیم. شامل دو قسمت:

- تبدیلهای مدلسازی که به اشیاء اعمال می شوند
- تبدیلی که دوربین را در نقطه ای از فضا مکاندهی و جهتدهی می کند.

هر چند یک ماتریس است می تواند بصورت ضرب دو ماتریس تصور شود: ماتریس مدلسازی M، ماتریس نگرش V

در واقع VM

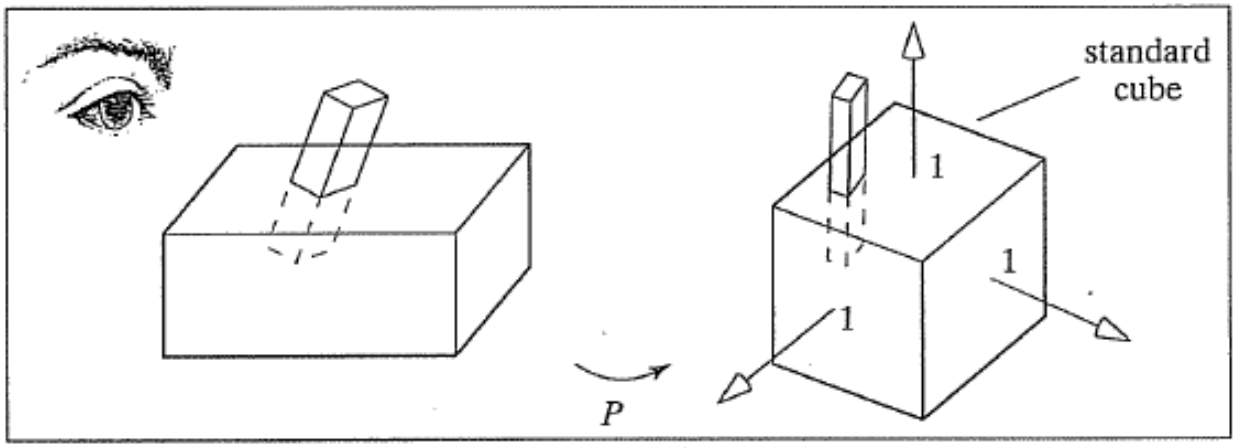


شکل ٤

M بلوک واحد را تغییر اندازه، دوران، و انتقال به مکان جدید می دهد. در این حال مکان شئ در دستگاه مختصات جهان بدست می آید. V دوربین را از مکان خود به مکان بنیادی خود می برد. در واقع حال نقاط را دستگاه مختصات دوربین خواهیم داشت. در مکان بنیادی دوربین در مبدأ بوده و حجم دید بر روی z تراز شده است

برای اطلاع OpenGL که بر روی ماتریس modelview می خواهیم کار کنیم از دستور glMatrixMode(GL_MODELVIEW) استفاده می کنیم. حجم دید از چپ به راست در راستای x ، از پائین به بالا در راستای y، و از نزدیک به دور در راستای z بسط دارد.

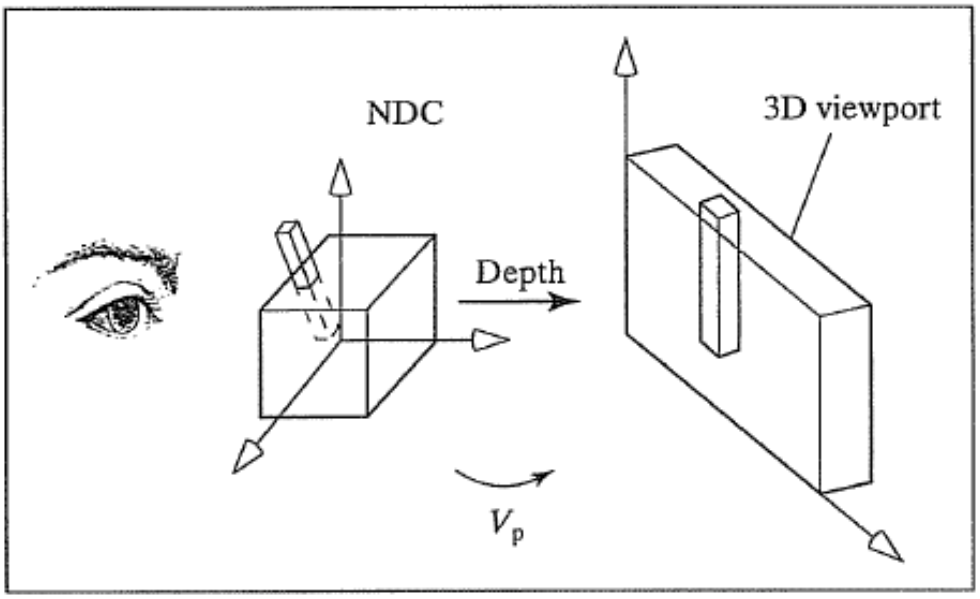
ماتریس projection تبدیلی انجام می دهد که حجم دید به حجم دید قانونی که مکعبی است که در هر بعد از ۱- تا ۱ بسط دارد تبدیل شود. برش در مقابل این حجم دید ساده تر است.



شکل ٥

- دستور glMatrixMode(GL_PROJECTION) برای کار با ماتریس projection استفاده می شود.

پس از بدست آوردن حجم دید قانونی عمل برش انجام می شود. نهایتاً ماتریس viewport قسمت باقیماندهٔ اشیاء را به یک بندردید سه بعدی تبدیل می کند. در بندر دید سه بعدی x و y در راستای بندردید دوبعدی (در مختصات دستگاه) تغییر می کنند. عمق از ۰ تا ۱ تغییر می کند.

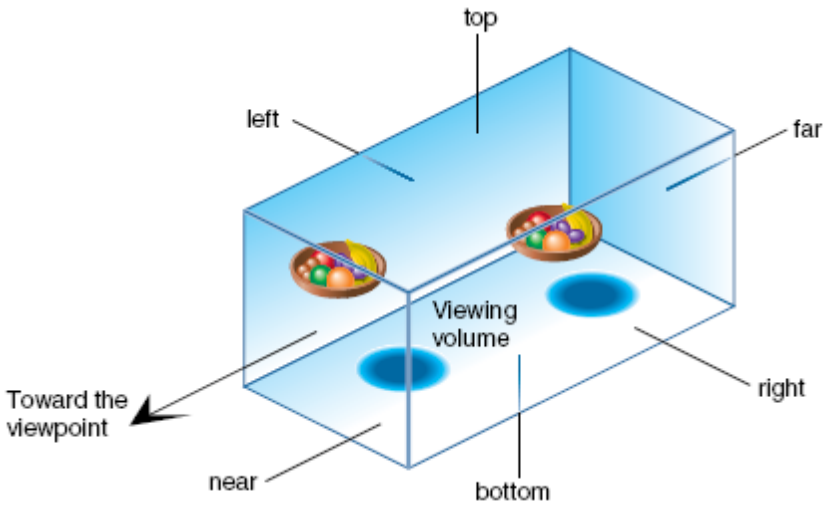


شکل ٦ تبدیل حجم دید قانوني به بندردید سه بعدي.

دستور glOrtho(left,right,bottom, top, near, far) یک منشور در دستگاه مختصات دوربین (یا چشم) مشخص می نماید که همان حجم دید مورد نظر ما است. در این دستگاه دوربین در مبدأ بوده و در جهت z– نگاه می کند. این دستور یک ماتریس ساخته و ماتریس فعلی را در آن پس ضرب می کند.

```
glMatrixMode(GL_PROJECTION); // make the projection matrix
                                current
glLoadIdentity();            // set it to the identity matrix
glOrtho(left, right, bottom, top, near, far); // multiply it by
                                                the new matrix
```

مقادیر near و far را مثبت می دهیم. مثلاً ۵ و ۱۵ یعنی به این مقدار در جلوی دوربین قرار دارند.

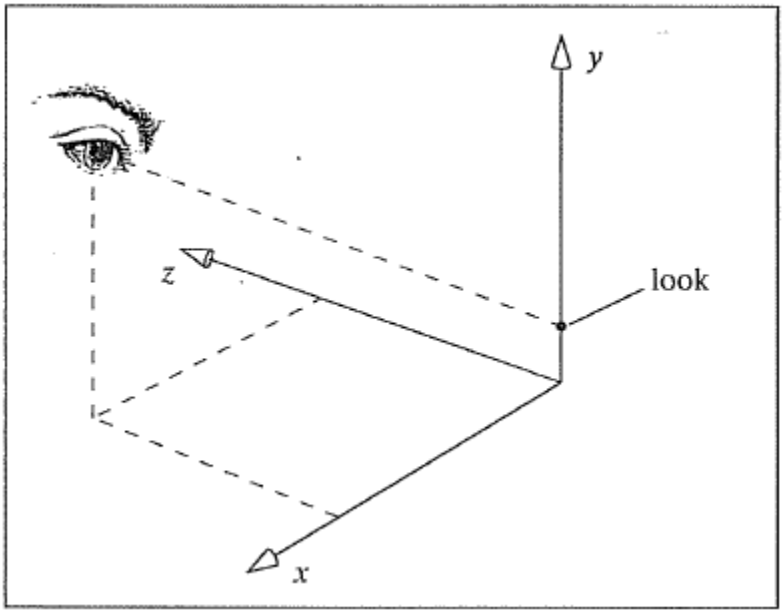


شکل ۷ [Shriener]

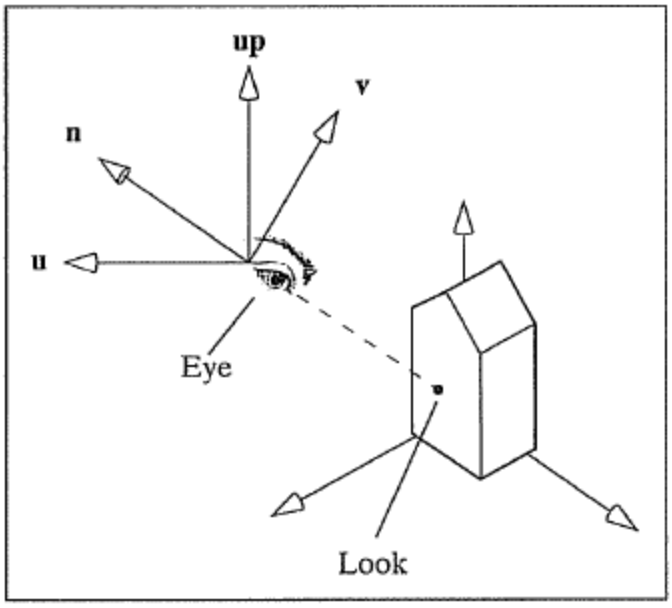
مکان و جهت دهی دوربین

- gluLookAt(eye.x,eye.y,eye.z,look.x, look.y, look.z, up.x, up.y, up.z)

در واقع دستگاه u، v،n بوجود آمده و ماتریس V ساخته می شود و نقاط را به دستگاه مختصات دوربین تبدیل می کند. به همین علت باید قبل از آنکه هر تبدیل مدلسازی صدا زده شود اعمال شود.



شکل ۸



شکل ۹ دستگاه مختصات جهان به دستگاه مختصات دوربین تبدیل می شود.

```
glMatrixMode(GL_PROJECTION);  // set the view volume
glLoadIdentity();
glOrtho(-3.2, 3.2, -2.4, 2.4, 1, 50);
glMatrixMode(GL_MODELVIEW);   // place and aim the camera
glLoadIdentity();
gluLookAt(4, 4, 4, 0, 1, 0, 0, 1, 0);
```

رسم برخی از اشیاء مهیا شده توسط OpenGL

کتابخانهٔ GLUT تعدادی از اشیاء را بصورت آماده مهیا ساخته است.

بصورت مدل قاب سیمی (wireframe)

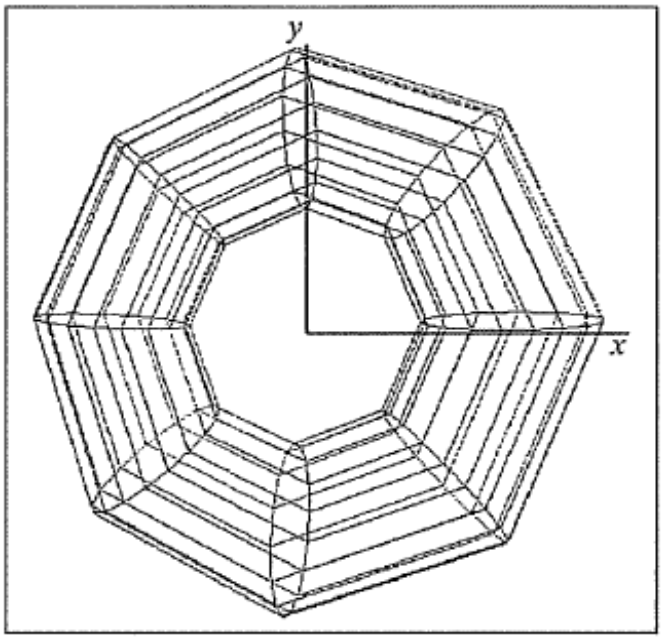
- **cube:** `glutWireCube(GLdouble size)` **Each side is of length** `size`
- **sphere:** `glutWireSphere(GLdouble radius, GLint nSlices, GLint nStacks)`
- **torus:** `glutWireTorus(GLdouble inRad, GLdouble outRad, GLint nSlices, GLint nStacks)`
- **teapot:** `glutWireTeapot(GLdouble size)`

- **cone:** glutWireCone(GLdouble baseRad, GLdouble height, GLint nSlices, GLint nStacks)
- **tapered cylinder:** gluCylinder(GLUquadricObj * qobj, GLdouble baseRad, GLdouble topRad, GLdouble height, GLint nSlices, GLint nStacks)

تعریف استوانه مقداری با دیگر اشیاء متفاوت است.

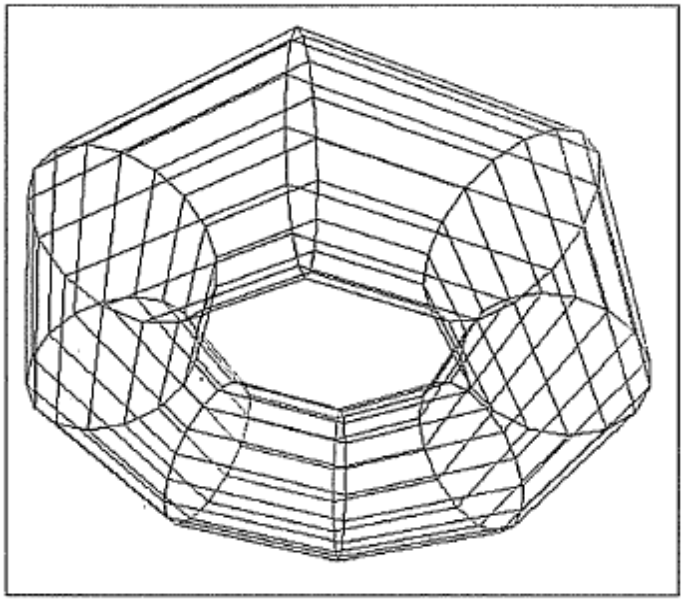
```
GLUquadricObj * qobj = gluNewQuadric(); // make a quadric object
gluQuadricDrawStyle(qobj,GLU_LINE);     // set style to wireframe
gluCylinder(qobj, baseRad, topRad, nSlices, nStacks); // draw the cylinder
```

کره و تورس با وجوه چندضلعی مدل می شوند. برای مشخص کردن تعداد وجوه از nSlices و nStacks استفاده می کنیم. اگر به تورس در راستای محور z نگاه کنیمnSlices تعداد قاچهائی که حول محور z وجود دارد را نشان می دهد (همانند برشهای یک پیتزا). برای شکل زیر تعداد nSlices ۸ است.



شکل ۱۰

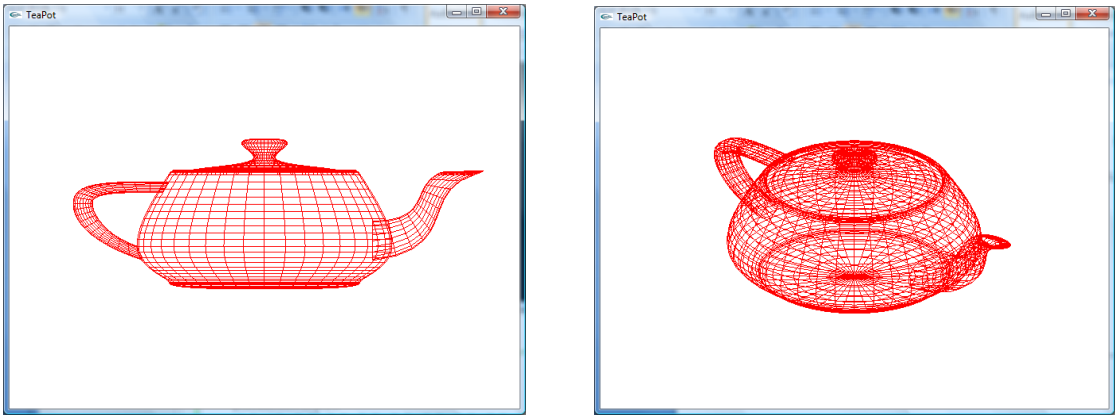
nStackes تعداد چندضلعیهائی است که در هر برش وجود دارند. برای شکل زیر تعداد nStackes ۱۶ است.



شکل ۱۱

مثال:

برنامهٔ زیر یک قوری که مرکز آن در مبدأ مختصات قرار دارد را ترسیم می کند. دوربین در دو وضعیت قرار دارد.



الف ب

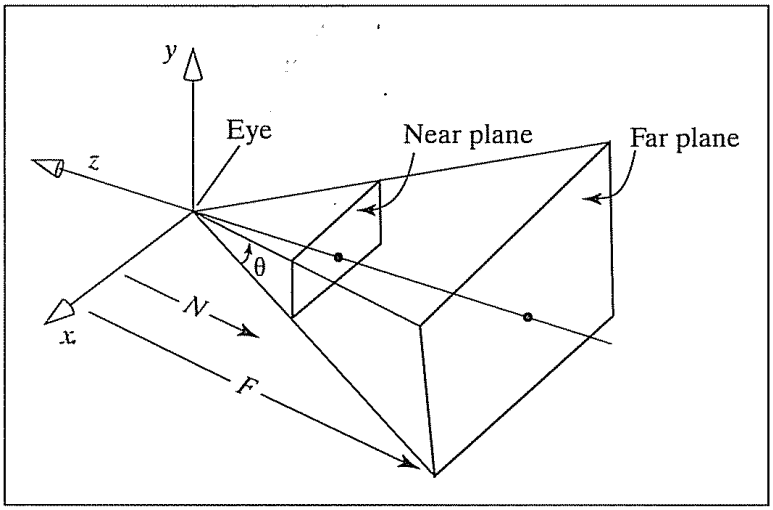
شکل ۱۲ الف) شکل قوري در صورتي که دوربين در (1,1,1) قرار داشته باشد. ب) شکل قوري در صورتي که دوربين در (0,0,1) قرار داشته باشد.

```
// In His Name the Most High
// This program draws a teapot.
// Programmer : Maziar Palhang

#include <windows.h>
#include <gl/gl.h>
#include <gl/glu.h>
#include <gl/glut.h>

void myInit(){
	glClearColor(1.0f, 1.0f, 1.0f, 0.0f);
	glViewport(0, 0, 640, 480);
}

void myDisplay(){
	glMatrixMode(GL_PROJECTION);
	glLoadIdentity();
	glOrtho(-2, 2, -2, 2, 0.1, 20);		//تعیین حجم دید

	//قرار دادن دوربین
	glMatrixMode(GL_MODELVIEW);
	glLoadIdentity();
	gluLookAt(1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);

	glClear(GL_COLOR_BUFFER_BIT);
	glColor3d(1.0, 0.0, 0.0);			//رسم به رنگ قرمز
	glutWireTeapot(1.0);
	glFlush();
}

int main(int argc, char** argv){
	glutInit(&argc, argv);
	glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
	glutInitWindowSize(640, 480);
	glutInitWindowPosition(100, 100);
	glutCreateWindow("TeaPot");
	glutDisplayFunc(myDisplay);
	myInit();
	glutMainLoop();

}
```

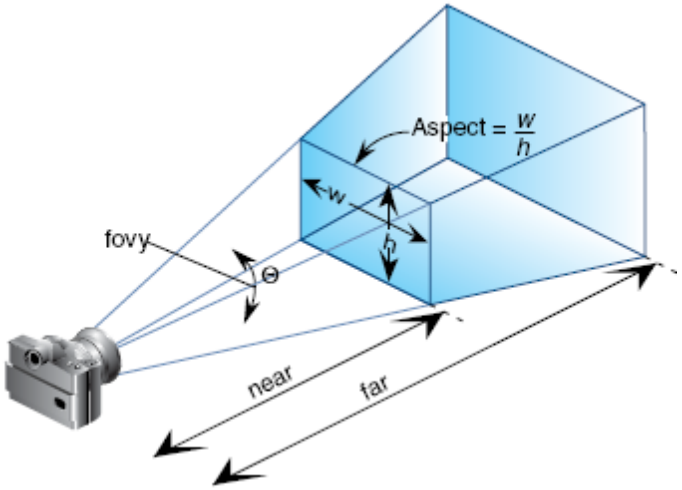
شکل ۱۳ یک برنامهٔ ساده براي رسم یک قوري

مشخص کردن حجم دید پرسپکتیو

حجم دید پرسپکتیو یک هرم می باشد که سر آن توسط صفحهٔ برش جلو بریده شده است. به چنین شکلی یک فراستام (frustum) گفته می شود. شکل ۱٤چشم (یا دوربین) را در مکان پیش فرض خود نشان می دهد. محور هرم در این شکل بر روی محور z تراز است. در این حالت حجم دید نسبت به محور z متقارن است.



شکل ۱٤



شکل ۱۵

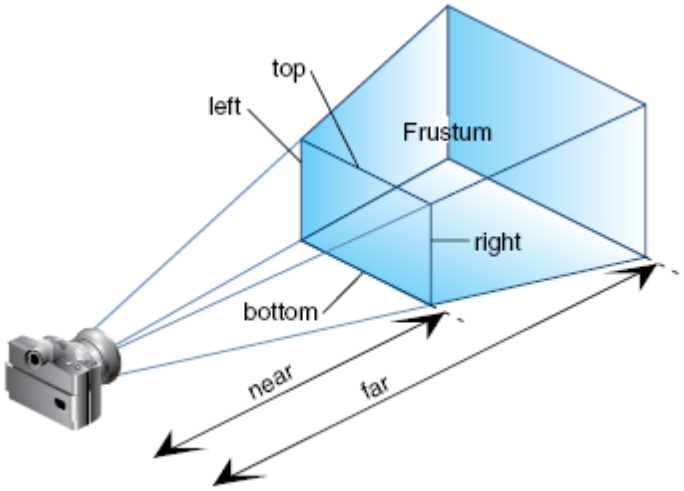
یک چنین حجم دیدی را می توان توسط دستور

gluPerspective(viewAngle, aspectRatio, N, F)

بوجود آورد. viewAngle زاویهٔ دید دوربین را نشان می دهد که در صفحهٔ yz اندازه گیری می شود. aspectRatio نسبت پهنا به ارتفاع صفحهٔ دید جلو یا عقب را نشان می دهد. N و F به ترتیب فاصلهٔ صفحهٔ جلو (نزدیک) و دور را نسبت به دوربین نشان می دهند.

در حالت کلی لازم نیست که حجم دید پرسپکتیو متقارن باشد. در این حالت می توان از دستور

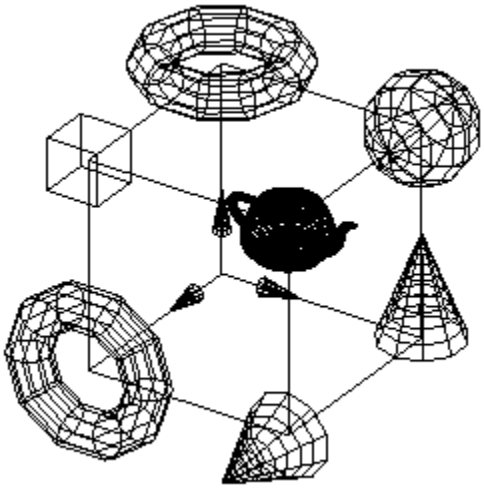
glFrustum(left,right,bottom,top,near,far) استفاده کرد. پنجرهٔ صفحهٔ جلو از left تا right در راستای x، و از bottom تا top در راستای y بسط دارد. صفحات جلو و دور در فاصلهٔ near و far به ترتیب نسبت به دوربین در راستای z– قرار دارند. این فواصل در دستگاه مختصات دوربین سنجیده می شوند.



شکل ۱۶

مثال:

رسم شکلي با چند شئ

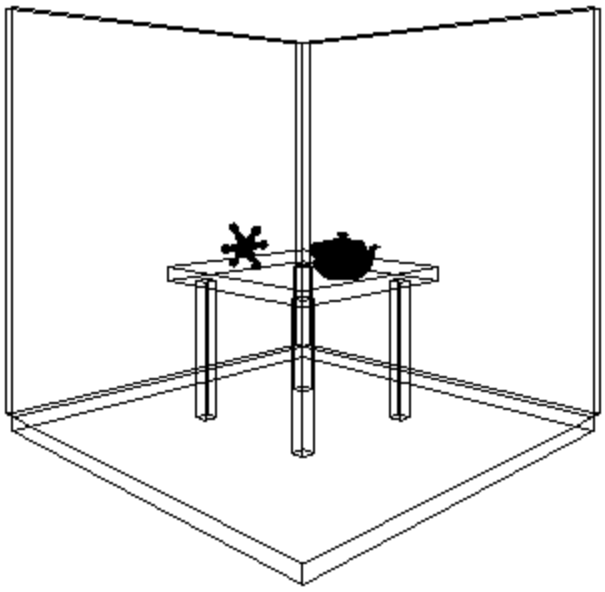


شکل ۱۷

```
// In His Name, the Most High

#include <windows.h>
#include <gl/gl.h>
#include <gl/glu.h>
#include <gl/glut.h>

void axis(){
	glPushMatrix();
	glTranslated(0.0, 0.0, 0.2);
	glutWireCone(0.04,0.2,10,8);
	glPopMatrix();
}

void myDisplay(){
	glMatrixMode(GL_PROJECTION);
	glLoadIdentity();
	glOrtho(-2.0*64/48.0, 2.0*64/48.0, -2.0, 2.0, 0.1, 100);

	glMatrixMode(GL_MODELVIEW);
	glLoadIdentity();
	gluLookAt(1.0,1.0,1.5,0.0,0.0,0.0,0.0,1.0,0.0);

	glClear(GL_COLOR_BUFFER_BIT);
	glColor3d(0.0, 0.0, 0.0);

	glPushMatrix();
	glTranslated(0.5,0.5,0.5);
	glutWireCube(1.0);
	glPopMatrix();

	glPushMatrix();
	glTranslated(1.0,1.0,0.0);
	glutWireSphere(0.25,10,8);
	glPopMatrix();

	glPushMatrix();
	glTranslated(1.0,0.0,1.0);
	glutWireCone(0.2,0.5,10,8);
	glPopMatrix();

	glPushMatrix();
	glTranslated(1.0,1.0,1.0);
	glutWireTeapot(0.2);
	glPopMatrix();

	glPushMatrix();
	glTranslated(0.0,1.0,0.0);
	glRotated(90.0, 1.0, 0.0, 0.0);
	glutWireTorus(0.1,0.3,10,10);
	glPopMatrix();
```

```
	glPushMatrix();
	glTranslated(0.0,1.0,1.0);
	glutWireCube(0.25);
	glPopMatrix();

	glPushMatrix();
	glTranslated(0.0,0.0,1.0);
	glutWireTorus(0.1,0.3,10,10);
	glPopMatrix();


	glPushMatrix();
	glTranslated(1.0,0.0,0.0);
	glRotated(-90.0, 1.0, 0.0, 0.0);
	glutWireCone(0.2,0.5,10,8);
	glPopMatrix();

	glPushMatrix();				// z axis
	axis();
	glPopMatrix();

	glPushMatrix();				// x axis
	glRotated(90.0, 0.0, 1.0, 0.0);
	axis();
	glPopMatrix();

	glPushMatrix();				// y axis
	glRotated(-90.0, 1.0, 0.0, 0.0);
	axis();
	glPopMatrix();

	glFlush();
}

int main(int argc, char** argv)
{
	glutInit(&argc, argv);
	glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
	glutInitWindowSize(640,480);
	glutInitWindowPosition(100,100);
	glutCreateWindow("scene");
	glutDisplayFunc(myDisplay);
	glClearColor(1.0f, 1.0f, 1.0f, 0.0f);
	glViewport(0,0,640,480);
	glutMainLoop();
}
```

مثال: رسم یک منظره از اتاق



شکل ۱۸

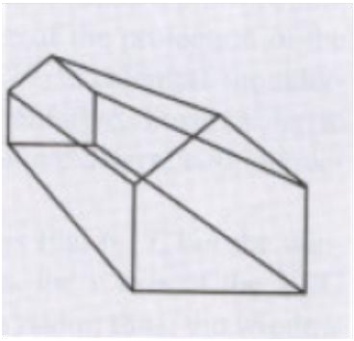
```
// In His Name, the Most High
// Programmer : Maziar Palhang

#include <windows.h>
#include <gl/gl.h>
#include <gl/glu.h>
#include <gl/glut.h>

void wall(){
      glPushMatrix();
      glScaled(3.0,0.1,3.0);
      glutWireCube(1.0);
      glPopMatrix();
}
void tableLeg(double thick, double len){
      glPushMatrix();
      glTranslated(0.0,len/2,0.0);
      glScaled(thick,len,thick);
      glutWireCube(1.0);
      glPopMatrix();
}

void drawTable(){
      glPushMatrix();                        //رسم پـايـه هـاي مـيـز
      tableLeg(0.1,1.0);
      glTranslated(1.0,0.0,0.0);
      tableLeg(0.1,1.0);
      glTranslated(0.0,0.0,1.0);
      tableLeg(0.1,1.0);
      glTranslated(-1.0,0.0,0.0);
      tableLeg(0.1,1.0);
      glPopMatrix();

      glPushMatrix();                        //رسم روي مـيـز
      glTranslated(0.5,1.05,0.5);
      glScaled(1.4,0.1,1.4);
      glutWireCube(1.0);
      glPopMatrix();
}
void jackPart(){
      glPushMatrix();
      glScaled(0.2,0.2,1.0);
      glutWireSphere(1,15,15);
      glPopMatrix();
      glPushMatrix();
      glTranslated(0.0,0.0,1.2);
      glutWireSphere(0.2,15,15);
      glTranslated(0.0,0.0,-2.4);
      glutWireSphere(0.2,15,15);
      glPopMatrix();
}
void jack(){
      glPushMatrix();
      jackPart();
      glRotated(90.0,0.0,1.0,0.0);
      jackPart();
      glRotated(90.0,1.0,0.0,0.0);
      jackPart();
      glPopMatrix();
}
```

```
void myDisplay(){
      glMatrixMode(GL_PROJECTION);
      glLoadIdentity();
      gluPerspective(60.0,64.0/48.0,0.1,100);

      glMatrixMode(GL_MODELVIEW);
      glLoadIdentity();
      gluLookAt(4.8,2.0,4.8,0.0,2.0,0.0,0.0,1.0,0.0);

      glClear(GL_COLOR_BUFFER_BIT);
      glColor3d(0.0, 0.0, 0.0);

      glPushMatrix();          //کف زمین
      glTranslated(0.5,-0.05,0.5);
      wall();
      glPopMatrix();

      glPushMatrix();          //دیوار سمت چپ
      glTranslated(-1.05,1.5,0.5);
      glRotated(90.0,0.0,0.0,1.0);
      wall();
      glPopMatrix();

      glPushMatrix();          //دیوار سمت راست
      glTranslated(0.5,1.5,-1.05);
      glRotated(90.0,1.0,0.0,0.0);
      wall();
      glPopMatrix();

      glPushMatrix();          //رسم میز
      drawTable();
      glPopMatrix();

      glPushMatrix();          //رسم جک
      glTranslated(0.2,1.25,0.8);
      glRotated(45.0,0.0,0.0,1.0);
      glScaled(0.15,0.15,0.15);
      jack();
      glPopMatrix();

      glPushMatrix();          //رسم قوري
      glTranslated(0.8,1.2,0.4);
      glutWireTeapot(0.2);
      glPopMatrix();
      glFlush();
}
```

```
int main(int argc, char** argv)
{
	glutInit(&argc, argv);
	glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
	glutInitWindowSize(640,480);
	glutInitWindowPosition(100,100);
	glutCreateWindow("scene");
	glutDisplayFunc(myDisplay);
	glClearColor(1.0f, 1.0f, 1.0f, 0.0f);
	glViewport(0,0,640,480);
	glutMainLoop();
	return 0;
}
```

تمرین

۱- پارامترهای دید برای آنکه یک تصویر پرسپکتیو یک نقطه ای از خانهٔ نشان داده شده در درس داشته باشیم مشابه شکل ۱۹ را بدست آورید.



شکل ۱۹

۲- پارمترهای دید برای آنکه یک تصویر آیزومتریک از خانه داشته باشیم را بدست آورید.

۳- برنامهٔ ترسیم قوری را نوشته و با تغییر حجم دید اشکال مختلفی را که بدست می آیند را بررسی نمائید. چه اتفاقی می افتد اگر صفحهٔ جلو از درون شکل عبور کند؟ در صورتی که دوربین در راستای محور y قرار داشته و به طرف مبدأ نگاه کند، بردار VUP را در چه سمتی باید در نظر گرفت؟

۴- توسط اشکال از پیش مهیا شده در OpenGL یک بوستان بازی شامل سرسره، تاب، و درخت ترسیم نمائید.

۵- برنامه ای بنویسید که توسط آن بتوان دوربین را در محیط به پرواز در آورد، بدین صورت که با فشردن هر یک از کلیدهای جهتی دوربین به جهت متناظر با آن حرکت نماید. دقت کنید که حرکت بهتر است نسبت به دستگاه دوربین انجام گیرد.