الجلسة السادسة

تحويل الاسقاط Projection

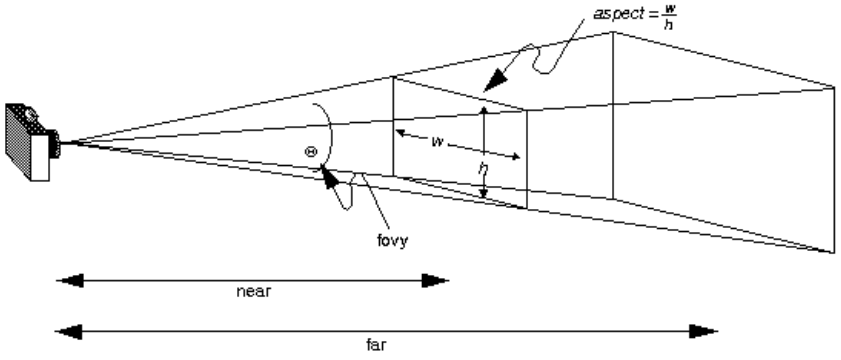
- تستخدم تحويلات الاسقاط لتحويل الرؤوس في المشهد. قبل إصدار أي أمر تحويل اسقاط يجب تنفيذ التالي:

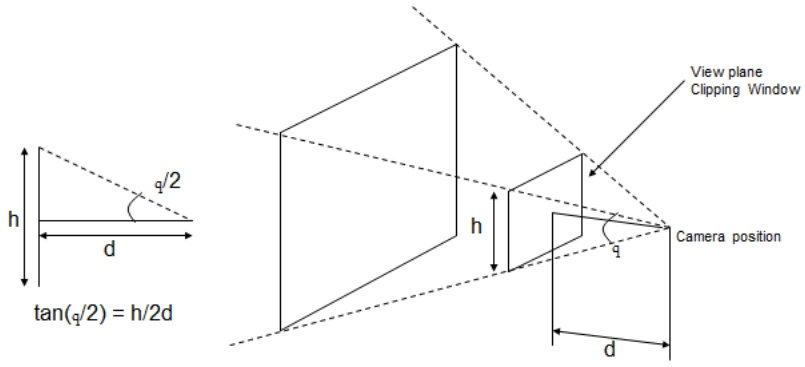
```
glMatrixMode( GL_PROJECTION);
glLoadIdentity();
```

- لتأثير التعليمات الحالية على مصفوفة الاسقاط وليس مصفوفة ModelView
- يستخدم تحويل الاسقاط لتعريف فضاء الرؤية الذي يستخدم بطريقتين:
- الاسقاط المنظور perspective projection والاسقاط المتعامد orthographic projection

- **الاسقاط المنظوري perspective projection**:

- كلما كان الجسم أبعد عن الكاميرا، كلما ظهر أصغر. يحدث ذلك لأن حجم الإظهار للإسقاط المنظوري عبارة عن جزء من هرم (جذع هرم) بدون رأس. الأجسام الأقرب من عين الناظر (الكاميرا) تبدو أكبر لأنها تحتل مساحة أكبر من حجم الإظهار. تستخدم هذه الطريقة من الاسقاط في الحركة وفي المحاكاة البصرية والتطبيقات الواقعية.
- gluPerspective() : يحدد زاوية حقل الرؤية في المستوي xz ونسبة الطول للعرض x/y كما في الشكل :





- الصيغة العامة للأمر gluPerspective():

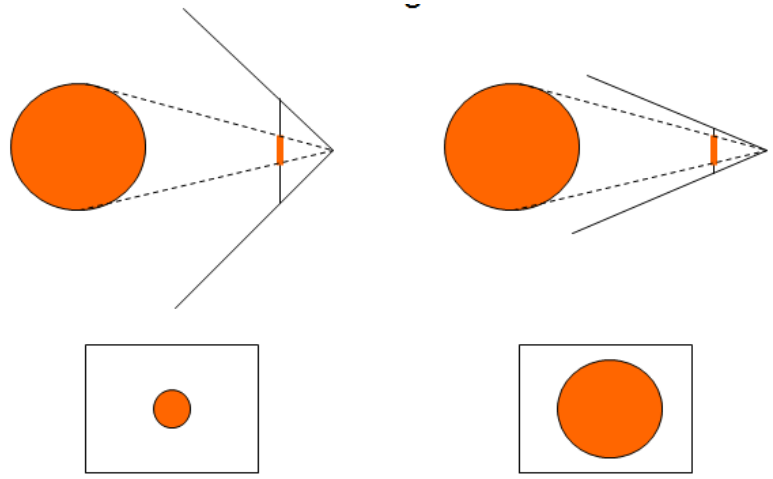
void **gluPerspective**(GLdouble *fovy*, GLdouble *aspect*, GLdouble *zNear*, GLdouble *zFar*);
gluPerspective(45,1.0,10.0,200.0);

حيث إن:

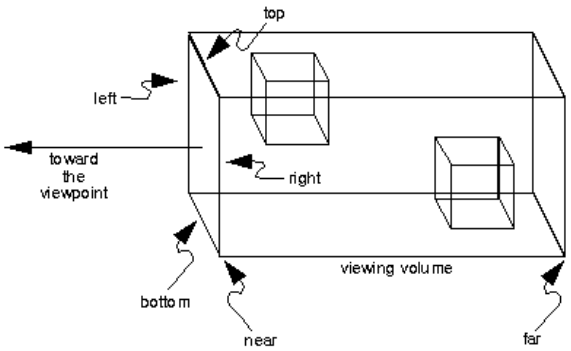
- Fovy: الزاوية بالدرجات بين مستوي القطع العلوي والسفلي. وهناك علاقة بين الزاوية وبين أبعاد اطار القطع كما يلي:

$$\tan(q/2) = h/2d$$

وكلما كانت الزاوية أكبر، كلما كان المشهد أصغر:



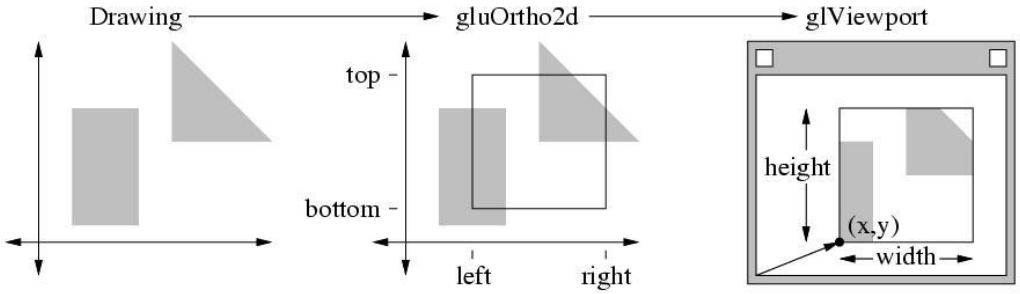
- **الإسقاط العمودي (على جملة إحداثيات) orthographic projection**:
  - فراغ الإظهار هنا عبارة عن صندوق



  - لا تؤثر بعد المسافة عن الكاميرا على حجم الجسم. يستخدم هذا النوع من الاسقاط لإنشاء الخرائط المعمارية والتصاميم باستخدام الحاسب .
  - الشكل العام لأمر glOrtho() (الذي ينشئ فضاء إظهار صندوق)

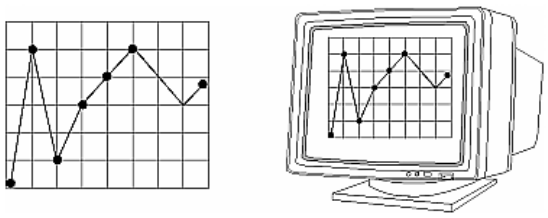
void **glOrtho**(GLdouble *left*, GLdouble *right*, GLdouble *bottom*, GLdouble *top*, GLdouble *near*, GLdouble *far*);

```
glOrtho(0.0, 400.0, 0.0, 400.0,10,200);
gluOrtho2D(0.0, 400.0, 0.0, 400.0);
```



تحويل viewport

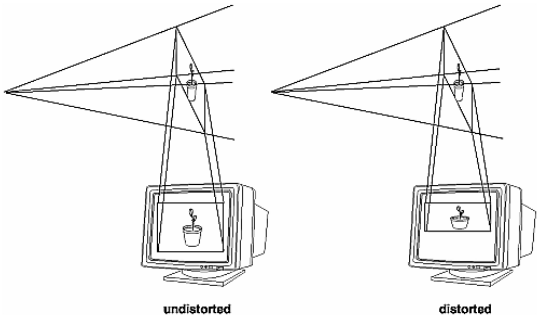
- viewport عبارة عن المنطقة المستطيلة من النافذة التي يتم فيها رسم الصورة



- **تعريف viewport**:
  - افتراضياً تعتبر viewport كامل بكسلات نافذة الرسم. سنستخدم الأمر glViewPort() لاختيار منطقة رسم أصغر. وبذلك نستفيد من النافذة الواحدة في الحصول على عدة مشاهد.
  - الشكل العام لأمر glViewPort():

void **glViewport**(GLint *x*, GLint *y*, GLsizei *width*, GLsizei *height*);

  - نسبة الطول للعرض لـ viewport يجب أن تكون متساوية مع نسبة الطول للعرض بالنسبة لفضاء الإظهار والا ستشوه الصورة .



  - مثال: الوضع الافتراضي

```
gluPerspective(myFovy, 1.0, myNear, myFar);
glViewport(0, 0, 400, 400);
```

  - الصورة مشوهة لأنها مضغوطة وفق المحور x

```
gluPerspective(myFovy, 2.0, myNear, myFar);
glViewport (0, 0, 400, 400);
```

- لتجنب التشوه

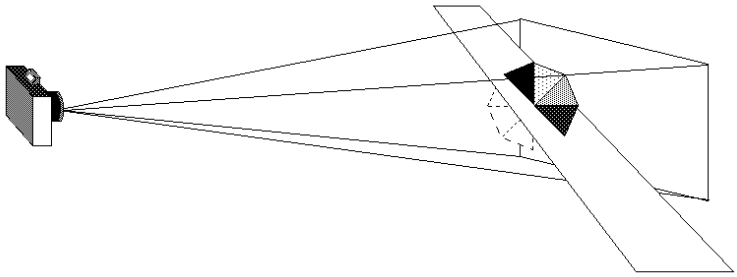
```
glViewport(0, 0, 400, 200);
```

- لإنشاء viewport عدد 2:

```
glViewport (0, 0, sizex/2, sizey);
                .
                .
glViewport (sizex/2, 0, sizex/2, sizey);
```

سطوح قطع إضافية

- بالإضافة إلى سطوح القطع الستة لفضاء الرؤية (يسار، يمين، أسفل، أعلى، قريب، بعيد) يمكنك تعريف 6 سطوح قطع إضافية كما في الشكل التالي:



- وهذا يفيد في إزالة عناصر غير مرغوبة من المشهد.
- الشكل العام لأمر تحديد سطح القطع :

```
void glClipPlane(GLenum plane, const GLdouble *equation);
```

- يفعل الأمر السابق بـ

```
glEnable(GL_CLIP_PLANEi);
```

- ويلغى تفعيله بـ

```
GlDisable(GL_CLIP_PLANEi);
```

النواظم (الأشعة الاعتيادية) Normal Vectors

- الناظم هو شعاع عمودي على السطح. تستطيع في OpenGL تحديد الناظم لكل رأس. يستخدم الأمر glNormal*() لتحديد الناظم الحالي، غالباً ما يكون لكل رأس ناظم مختلف:

```
glBegin (GL_POLYGON);
  glNormal3fv(n0);
  glVertex3fv(v0);
  glNormal3fv(n1);
  glVertex3fv(v1);
  glNormal3fv(n2);
  glVertex3fv(v2);
  glNormal3fv(n3);
  glVertex3fv(v3);
glEnd();
```

- الشكل العام لأمر الناظم:

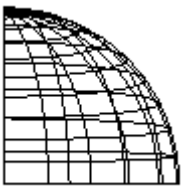
```
void glNormal 3 {bsidf} (TYPE nx, TYPE ny, TYPE nz);
void glNormal 3 {bsidf}v (Const TYPE *v);
```

- للناظم اتجاهان متعاكسان، للخارج والآخر للداخل ولكي ترى السطح يجب أن يكون اتجاه الناظم من السطح باتجاه الناظر , تستطيع قلب الناظم من (x,y,z) إلى (-x,-y,-z).تبنى السطوح اعتماداً من مضلعات صغيرة (مثلثات أو مربعات...) .
- تقليل المضلعات المشكلة للسطح يسرع التصيير ولكن مظهر السطح يكون مشوهاً، وزيادتها بشكل كبير تؤدي إلى مظهر جيد ولكن بزمن تصيير كبير. عادة يمكنك تزويد روتين التقسيم ببارامتر يشير إلى عدد التقسيمات المطلوبة.
- تأكد عند رسم سطح مغلق من استخدام نفس أرقام الإحداثيات لبداية ونهاية الحلقة المغلقة وإلا ستحصل على ثغرات.

القسم العملي

التطبيق 1 :

مثال عن سطح قطع (تصيير كرة سلكية مع سطحي قطع يستخدمان لقطع ¾ الكرة الأصلية



```
#include <GL/glut.h>
#include <stdlib.h>
#include<math.h>
static void redraw(void)
{
      GLdouble eqn[4] = {0.0, 1.0, 0.0, 0.0};    /* y < 0 */
            GLdouble eqn2[4] = {1.0, 0.0, 0.0, 0.0};   /* x < 0 */
            glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
            glLoadIdentity();
            glTranslatef(0.0f,0.0f,-100.0f);
            glColor3f(1.0, 1.0, 1.0);
            glClipPlane (GL_CLIP_PLANE0, eqn);
            glEnable (GL_CLIP_PLANE0);
            glClipPlane (GL_CLIP_PLANE1, eqn2);
            glEnable (GL_CLIP_PLANE1);
            glRotatef (90.0, 1.0, 0.0, 0.0);
            glutWireSphere(30.0,10,20);
            glutSwapBuffers();

}
 int main(int argc, char**argv)
{
   glutInit(&argc,argv);
            glutInitDisplayMode(GLUT_RGB | GLUT_DOUBLE | GLUT_DEPTH);
            glutInitWindowPosition(100,100);
            glutInitWindowSize(400,400);
            glutCreateWindow("Application11");
            glutDisplayFunc(redraw);
            glMatrixMode(GL_PROJECTION);
            gluPerspective(45,1.0,10.0,200.0);
            glMatrixMode(GL_MODELVIEW);
            glutMainLoop();
            return 0;
}
```

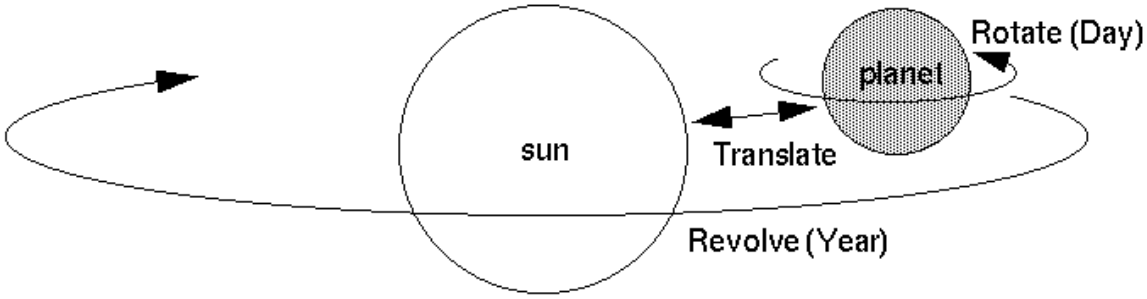
تعديل1: يطلب تغيير مستوى القطع لاظهار الربع الثالث

تعديل2: يطلب تغيير مستوى القطع لاظهار النصف السفلي للكرة

تعديل3: يطلب استخدام glortho ومقارنة النتائج

التطبيق 2 :

النظام الشمسي (كوكب وشمس) تدور فيه العناصر حول محورها وفي المدار حول بعضها.



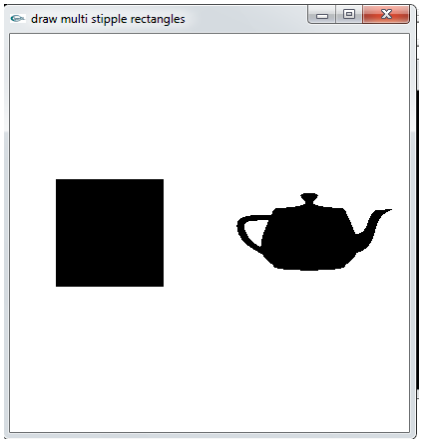

```
static int year = 0, day = 0;
static void redraw(void);

static void redraw(void)
{
    glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
    glLoadIdentity();
    glTranslatef(0.0f,0.0f,-100.0f);
    glColor3f(1.0, 1.0, 1.0);
    glutSolidSphere(10.0,20,20);            /* رسم الشمس */
    glRotatef ((GLfloat) year, 0.0, 1.0, 0.0);
    glTranslatef (20.0, 0.0, 0.0);
    glRotatef ((GLfloat) day, 0.0, 1.0, 0.0);
    glColor3f (1.0, 0.0, 0.0);
    glutWireSphere(3.0,10,20);              /* رسم كوكب صغير */
    day = (day + 15) % 360;/* دوران الكوكب حول نفسه */
    year = (year - 1) % 360;/* دوران الكوكب حول الشمس */
    glutPostRedisplay();/* من أجل إعادة رسم الإطار لتطبيق تأثير الحركة */
    glutSwapBuffers();
}
```

تعديل1: يطلب اظهار الكرة خلف الشمس بشكل صحيح

تعديل2: يطلب زيادة سرعة الدوران وتغيير الاتجاه

تمرين: يطلب رسم الشكلين التاليين، كل شكل في viewport الخاصة به:



تعديل: النظر إلى ابريق الشاي من الأعلى

****