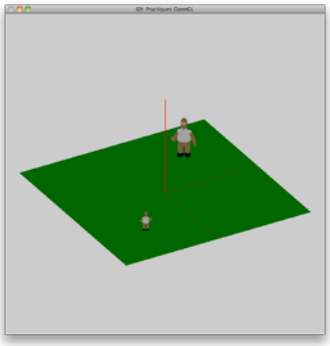
IDI Examen parcial - Grup 10 - Curs 2013-2014 1Q - Temps 1h 15m

1. Disposem de la informació de la capsa contenidora d'un model (xmin,ymin,zmin,xmax,ymax,zmax) que sabem està orientat amb el seu nas mirant en la direcció (0,0,1). Volem situar dos instàncies d'aquest model a sobre d'un terra ubicat en el pla Y=0 amb centre (0,0,0) i de mides 10x10, de manera que:
   - Una instància del model quedi ubicada amb el centre de la base de la seva capsa en (-2.5,0,2.5), tingui alçada 1 i miri el seu nas cap a l'eix Y.
   - L' altre instància tingui el centre de la capsa de la seva base en (2.5,0.,-2.5), alçada 2 i miri el seu nas cap l'eix Y.

   Disposeu d'un mètode `pinta_model()` que fa el recorregut de la geometria del model i l'envia a pintar.
   Indiqueu: la transformació geomètrica requerida per a ubicar cada instància en l' escena, i també el codi d'OpenGL per a pintar l'escena. Podeu suposar que la càmera ja està correctament inicialitzada.
   La figura adjunta és una visió general de l' escena.



Per a poder ubicar les instàncies s'agafa com a referència la capsa mínima contenidora del model:

- *CentreBaseCapsa= ((xmin+xmax)/2, ymin, (zmin+zmax)/2)*
- *alçadaCapsa=(ymax-ymin); escH1= 1/alçadaCapsa; escH2=2/alçadaCapsa*
- *Al cada homer cal aplicar: una translació per a portar la seva capsa a l'origen, un escalat, un gir respecte l'eix Y per a que miri en la direcció desitjada i, per últim, una translació a la seva posició final*

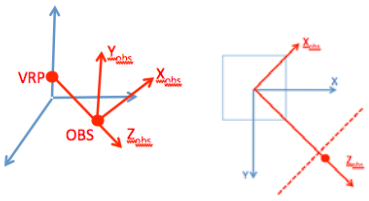
Transformacions a aplicar a les instàncies:

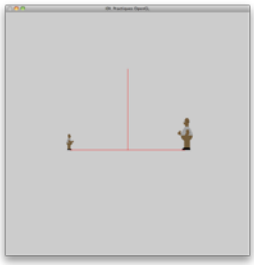
- *TG1= Translació(-2.5,0,2.5)Gir$_Y$ (135º)*Escalat(escH1)*Translació(-CentreBaseCapsa)*
- *TG2= Translació(2.5,0,-2.5)Gir$_Y$ (-45º)*(Escalat(escH2)*Translació(-CentreBaseCapsa)*

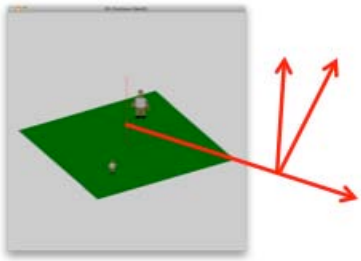
```
glMatrixMode(GL_MODELVIEW);
glPushMatrix();
glTranlatef(-2.5,0,2.5);
glRotatef(-135,0,1,0);
glScalef(escH1,escH1,escH1);
glTranslatef (-CentreBaseCapsa.x, -CentreBaseCapsa.y, -CentreBaseCapsa.z);
pinta_model();
glPopMatrix();
glPushMatrix();
glTranlatef(2.5,0,-2.5);
glRotatef(-45,0,1,0);
glScalef(escH2,escH2,escH2);
glTranslatef (-CentreBaseCapsa.x, -CentreBaseCapsa.y, -CentreBaseCapsa.z);
pinta_model();
glPopMatrix();
```

2. L'escena anterior, s'observa amb una càmera posicionada amb VRP=(0,1,0), OBS=(20,1,20) i up=(0,1,0), i una òptica axonomètrica amb un *window*=(-7,5,7.5,-7.5,7.5), Znear=1 i Zfar=50. Contesta i raona les respostes següents:
   a) Si el *viewport* és de 600x600, dibuixa la imatge que es veuria (pots aproximar els "ninots" per cilindres i indicar l'orientació del seu nas amb una fletxa).
   b) Si en comptes d'una òptica axonomètrica, utilitzes una de perspectiva amb ra=1, els mateixos Znear i Zfar, i un angle FOV que permet veure el mateix tros d'escena, observaries alguna diferència en la image final?

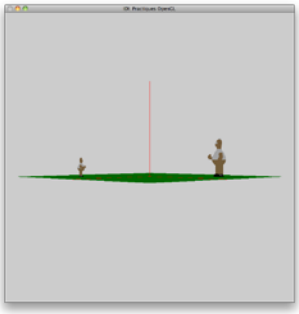
a) Donada la ubicació de la càmera, mostrada esquemàticament la figura d'abaix a l'esquerra; la direcció de visió és perpendicular a la diagonal del terra sobre la que estan els homers, i el punt (0,1,0) -VRP- quedarà enmig del viewport. Com up=(0,1,0) és perpendicular a Z_{OBS} , l'eix vertical de la càmera tindrà la direcció de l'eix Y i l'eix horitzontal (X_{OBS}) la direcció de la diagonal que conté el centre de la base dels homers; per tant, aquests estaran posicionats -respecte la càmera- un davant de l'altre (el homer més alt a la dreta) en una recta paral·lela a la base del viewport. L'òptica de la càmera permet veure tota l'escena. Noteu que l'amplada del window val 15 i la diagonal del terra 10sqrt(2)=14,1 i znear i zfar permeten abarcar tota l'escena. Al ser el viewport també quadrat no hi haurà deformacions. La imatge que s'obtindrà serà la de la figura d'abaix a la dreta. Noteu que el terra no es veu perquè, en ser paral·lel al pla $X_{OBS}Z_{OBS}$ i ser una projecció axonomètrica es projecta en una línia.







b) En aquest cas, la posició relativa dels homers respecta la càmera no es modifica (només és funció de la posició de la càmera que no s'ha modificat). Tanmateix, degut a la projecció perspectiva el terra ja no es projecta en una línea, sinó que queda "deformat" com mostra la figura.



3. Una esfera de radi 1 es visualitza en un *viewport* quadrat de 400 per 400, amb una càmera posicionada correctament amb angles d'Euler, i on el mètode per a definir la projecció de la càmera utilitza la següent crida:

```
gluPerspective(60.0, 1.0, 1.0, 10.0);
```

L'usuari ha redimensionat la finestra a 500 d'amplada per 400 d'alçada. Digues què cal canviar de la càmera per tal que es vegi l'esfera correctament (sense retallar-la ni deformar-la).

a. Incrementar l'angle d'obertura vertical (FOV) i la relació d'aspecte del *window*.
b. Augmentar la relació d'aspecte del *window* i la distància al ZNear.
c. Només augmentar la relació d'aspecte del *window*.
d. Només canviar l'angle d'obertura vertical (FOV).

4. Tenim una esfera de radi 1 centrada al punt (0.0, 0.0, 0.0) i una càmera amb una òptica definida com

```
glMatrixMode/(GL_PROJECTION);
glLoadIdentity();
glOrtho(0.0, 1.0, 0.0, 1.0, 0.0, 3.0);  // left,right,bottom,up,znear,zfar
```

Si el mètode de pintat és:

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glColor3f(1.0, 0.0, 0.2);
glTranslatef(0.0, 0.0, -2.0);
glutSolidSphere(1.0, 20., 20.);
```

Què es veurà?

a. La part superior dreta de l'esfera.
b. La part superior de l'esfera.
c. La part inferior esquerra de l'esfera.
d. La part dreta de l'esfera.

5. Quin dels següents codis creus que permetria definir la transformació de la càmera VRP=(0,1,0), OBS=(20,1,20) i up=(0,1,0) amb transformacions geomètriques?

a.

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glTranslatef(0, 0, -20*sqrt(2.));
glRotated(45, 0, 0, 1);
glTranslatef(0, -1, 0);
```

b.

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glTranslatef(0, 0, 20);
glRotated(-45, 0, 1, 0);
glTranslatef(0, -1, 0);
```

c.

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glTranslatef(0, 0, -20);
glRotated(45, 0, 1, 0);
glTranslatef(0, -1, 0);
```

d.

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glTranslatef(0, 0, -20*sqrt(2.));
glRotated(-45, 0, 1, 0);
glTranslatef(0, -1, 0);
```

6. Quan s'inicialitza la càmera, en quin ordre cal indicar les transformacions de càmera i el *viewport* a OpenGL?

a. No importa l'ordre en què s'indiquen.
b. Transformació de posició+orientació, transformació de projecció, *viewport*.
c. La transformació de projecció, transformació de posició+orientació, *viewport*.
d. *Viewport*, tansformació de projecció, transformació de posició+orientació.