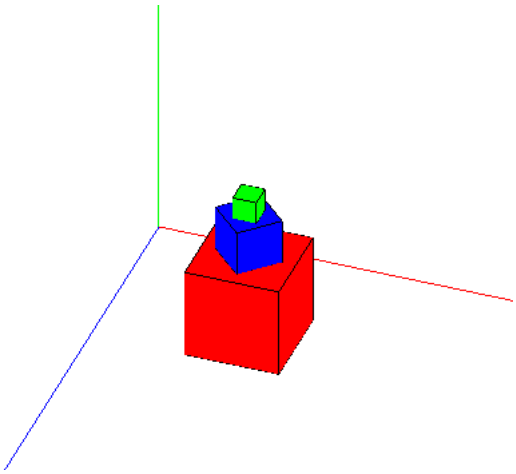
Examen parcial d'IDI 2014-2015, Q1. 29/10/2014 12:30

Nom i cognoms: Temps: 1h 30'

1. *(2 punts)* Disposem d'una funció `pinta_cub()` que pinta un cub de costat 1 centrat a l'origen i amb les seves cares orientades segons els plans coordenats.

   Volem construir una escena que sigui un pilar de tres cubs de costats 2, 1 i 0.5 respectivament com es mostra a la figura (no importa el color). El cub de baix de tot té el centre de la seva base al punt (3, 0, 3), i les seves cares estan orientades segons els plans coordenats. Tots tres cubs fan pila, és a dir, les cares corresponents es toquen i comparteixen el mateix eix vertical que passa pel seu centre.

   Es demana que escriguis una funció `pinta_escena()` que tingui el codi OpenGL que cal per a pintar aquest pilar de cubs a partir de la funció `pinta_cub()`. Considera que la càmera ja està completament inicialitzada. Justifica la resposta en base a la transformació geomètrica que cal aplicar a `pinta_cub()` per a pintar cadascun dels tres cubs.



Solució:
D'acord amb les condicions de l'enunciat, les transformacions a efectuar a cada cub són:

TG_CubBase = Trans (3, 1, 3) * Escalat (2, 2, 2)
TG_CubMig = Trans (3, 2.5, 3) * Gir$_y$ (45)
TG_CubDalt = Trans (3, 3.25, 3) * Escalat (0.5, 0.5, 0.5)

El CubBase requereix d'un escalat per a què el seu costat valgui 2, i la translació puja la seva base la meitat del costat en Y i posiciona el centre de la base en (3, 0, 3).

El CubMig no necessita escalat perquè ja és de costat 1 però sí requereix una rotació respecte l'eix Y de 45 graus. I la seva translació final és al punt (3, 2.5, 3) perquè és la mateixa posició que el CubBase però a una altura (Y) de 2.5.

Finalment el CubDalt també ha d'estar escalat, aquest cop per 0.5, i la translació és al punt (3, 3.25, 3) perquè és just al damunt del CubMig.

El codi final de la funció `pinta_escena()` podria ser el següent:

```
void pinta_escena ()
{
  glMatrixMode (GL_MODELVIEW);
  // Pintem el cub Base
  glPushMatrix ();
  glTranslated (3, 1, 3);
  glScaled (2, 2, 2);
  pinta_cub ();
  glPopMatrix();
  // Pintem el cub Mig
  glPushMatrix();
  glTranslated (3, 2.5, 3);
  glRotated (45, 0, 1, 0);
  pinta_cub ();
  glPopMatrix();
  // segueix al costat...
  // Pintem el cub Dalt
  glPushMatrix();
  glTranslated (3, 3.25, 3);
  glScaled (0.5, 0.5, 0.5);
  pinta_cub ();
  glPopMatrix();
}
```

2. *(2 punts)* Indica TOTS els paràmetres d'una càmera axonomètrica que en pintar l'escena de l'exercici 1 cridant a la funció `pinta_escena()` en un *viewport* (vista) de 800x400, es vegin a la vista només les cares superiors de tots tres cubs, centrades, ocupant el màxim del *viewport* i sense deformació. Utilitza com a paràmetres de posició i orientació OBS, VRP i Vup. Dibuixa la imatge que et quedarà a la vista i justifica les respostes.

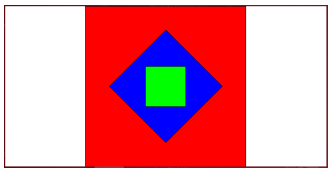
Solució:

Si s'han de veure les cares superiors dels tres cubs, s'ha de fer una vista en planta, per tant s'han de veure els cubs des de dalt i amb un vector de visió perpendicular a les cares dels cubs.

Si a més s'han de veure aquestes cares centrades a la vista, això vol dir que l'observador ha d'estar per damunt dels tres cubs i mirant cap al seu centre, per exemple: **VRP = (3, 0, 3)** i **OBS = (3, 5, 3)**. El vector Vup pot ser qualsevol que no sigui paral·lel a la direcció de visió, i en aquest cas, per a què les cares ocupin la màxima alçada del *viewport* podem fer que sigui paral·lel a un dels costats del cub més gran, per exemple: **Vup = (0, 0, -1)**.

Com que la relació d'aspecte del *viewport* és 2 (800/400), la del *window* també ha de ser 2 per a què no hi hagi deformació, i per tant l'amplada del *window* ha de ser el doble que l'alçada. Si per a l'alçada necessitem que es vegi sencer el nostre cub gran, hem de posar: **bottom = -1** i **top = 1**. I per a complir amb la relació d'aspecte tindrem que: **left = -2** i **right = 2**.

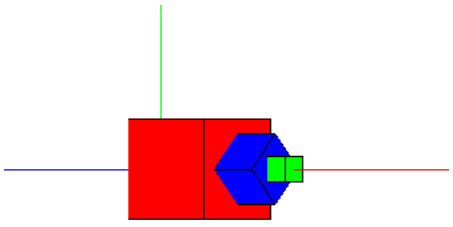
Els plans de retallat anterior i posterior hauran d'estar d'acord amb la posició de l'observador que hem posat, i per tant tindrem: **ZNear = 1.5** i **ZFar = 5**.



En resum, els paràmetres de la càmera ortogonal que em permeten veure aquesta imatge són:

OBS = (3, 5, 3); VRP = (3, 0, 3); Vup = (0, 0, -1);
left = -2; right = 2; bottom = -1; top = 1;
ZNear = 1.5; ZFar = 5.

3. *(1 punt)* Què canviaries del codi de la teva resposta a la pregunta 1 (funció `pinta_escena()`) per a què l'escena inicial sigui la de la figura següent? Justifica la resposta.



Solució:

Cal portar tota l'escena anterior a l'origen (translació (-3,0,-3)) i després rotar-la respecte l'eix Z (rotació de -90º). Per tant, caldria afegir a la funció, després del `glMatrixMode(GL_MODELVIEW);`, el codi següent:

```
glRotatef (-90, 0, 0, 1);
glTranslatef (-3, 0, -3);
```

4. *(1 punt)* Tenim una escena com la de la pregunta 1 però amb la base (de la pila de cubs) centrada en el punt (X, 0, Z). Volem una càmera que miri els cubs en una vista en planta (des de dalt) i els vegi centrats a la vista. Suposant l'òptica de la càmera ben definida, indica quin dels següents trossos de codi et permetria definir amb transformacions geomètriques la posició i orientació de la càmera.

a)
```
glMatrixMode (GL_PROJECTION);
glLoadIdentity ();
glTranslatef (0, 0, -5);
glRotatef (-90, 0, 0, 1);
glRotatef (90, 1, 0, 0);
glRotatef (-90, 0, 1, 0);
glTranslatef (X, 0, Z);
```

b)
```
glMatrixMode (GL_MODELVIEW);
glLoadIdentity ():
glTranslatef (0, 0, -5);
glRotatef (90, 1, 0, 0);
glRotatef (-90, 0, 1, 0);
```

c)
```
glMatrixMode (GL_MODELVIEW);
glLoadIdentity ():
glTranslatef (0, 0, -5);
glRotatef (90, 0, 0, 1);
glRotatef (-90, 1, 0, 0);
glRotatef (-90, 0, 1, 0);
glTranslatef (-X, 0, -Z);
```

d)
```
glMatrixMode (GL_MODELVIEW);
glLoadIdentity ():
glTranslatef (0, 0, -3);
glRotatef (90, 0, 0, 1);
glRotatef (90, 1, 0, 0);
glTranslatef (-X, -2, -Z);
```

5. *(1 punt)* Tenim definida una càmera axonomètrica amb paràmetres: OBS = (0, -1, 0), VRP = (0, 0, 0); Vup = (0, 0, 1), Window = (-2, 2, -2, 2), ZNear = 1, ZFar = 4, i una relació d'aspecte de la vista de 1 (rav = 1). Indica quina de les següents càmeres perspectiva seria adient per a definir un volum de visió **que inclogui completament** l'anterior:

a) OBS=(0, -2, 0), VRP=(0, 2, 0), Vup=(0, 0, 1), FOV=90, ra=1, ZNear=2, ZFar=5.
b) OBS=(0, 2, 0), VRP=(0, 0, 0), Vup=(0, 1, 0), FOV=90, ra=1, ZNear=1, ZFar=4.
c) OBS=(0, -1, 0), VRP=(0, 0, 0), Vup=(0, 0, 1), FOV=90, ra=1, ZNear=1, ZFar=4.
d) OBS=(0, -2, 0), VRP=(0, 0, 0), Vup=(0, 0, 1), FOV=60, ra=1, ZNear=1, ZFar=4.

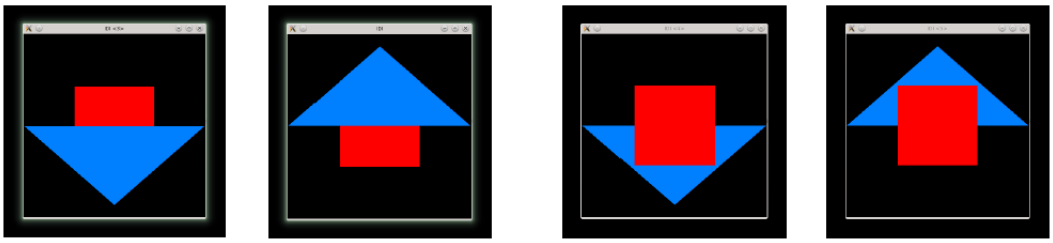
6. *(1 punt)* Tenim definida correctament una càmera perspectiva que ens permet veure una escena completa sense retallar i sense deformació. Si a aquesta càmera li modifiquem el FOV, indica què caldrà modificar també si no volem que hi hagi deformació:

a) La relació d'aspecte de la vista (rav).
b) No cal modificar res més.
c) La relació d'aspecte del window (raw).
d) Depèn de si incrementem o decrementem el FOV.

7. *(1 punt)* Pintem una escena amb el següent codi:

```
glViewport (0, 0, 800, 800);
glEnable (GL_DEPTH_TEST);
glClearColor (0, 0, 0, 1);
glClear (GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
glMatrixMode (GL_PROJECTION); // Inici definició de càmera
glLoadIdentity ();
glOrtho (-5, 5, -5, 5, 5, 15);
glMatrixMode (GL_MODELVIEW);
glLoadIdentity ();
glTranslatef (0, 0, -10);
glRotatef (180, 0, 0, 1);
glRotatef (90, 1, 0, 0);  // Fí definició de càmera
glColor3f (0, 0.5, 1);
glutSolidCone (5,5,20,20); // radi, alçada, orientació Z+, base centrada en (0,0,0)
glColor3f (1, 0, 0);
glPushMatrix ();
glTranslatef (0, -2, 0);
glutSolidCube (4); // costat 4, centrat a l'origen
glPopMatrix ();
```

Digues quina de les imatges següents es veu:



a) La primera imatge

b) La segona imatge

c) La tercera imatge

d) La quarta imatge

8. *(1 punt)* Tenim una esfera de radi 3 centrada a l'origen de coordenades i un focus de llum situat a la posició (0, 3, 5) de color (1, 1, 0). No hi ha llum ambient. Un observador es mira aquesta escena des de la posició (0, 0, 5) i mirant cap al centre de l'esfera, i el que observa és una esfera que té una part propera a la silueta per la part de baix de l'esfera de color negre, un degradat de colors verds que són més clars per la part de dalt de l'esfera i més foscos per la part de baix i una taca de color groc cap a la part del mig de la semiesfera superior. Quines constants de material de l'esfera permeten que es pugui veure aquesta escena de la forma descrita?

a) Ka = (0, 0.2, 0), Kd = (0, 0.8, 0), Ks = (0, 0, 0) i N = 100

b) Ka = (0.2, 0.2, 0.2), Kd = (0.8, 0.8, 0.8), Ks = (1, 1, 1) i N = 100

c) Ka = (0, 0.2, 0.2), Kd = (0, 0,8, 0.8), Ks = (1, 1, 1) i N = 100

d) Ka = (0, 0.2, 0.2), Kd = (0, 0.8, 0.8), Ks = (0, 1, 1) i N = 100