Examen parcial d'IDI 2015-2016, Q1. 4/11/2015 12:30

Nom i cognoms: Temps: 1h 20'

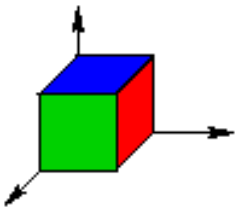
1. *(1.5 punts)* Tenim una escena amb un cub de costat 2 orientat amb els eixos i de manera que el seu vèrtex mínim està situat a l'origen de coordenades. La cara del cub que queda sobre el pla x=2 és de color vermell, la cara que queda sobre el pla z=2 és de color verd i la resta de cares són blaves.

   a) Indica TOTS els paràmetres d'una càmera axonomètrica que permeti veure a la vista un rectangle centrat amb la meitat esquerra de color verd i la meitat dreta de color vermell. La relació d'aspecte del viewport (vista) és 2.

   b) Quin efecte tindria en la imatge final modificar l'òptica a perspectiva? Pots indicar-ho utilitzant un dibuix si vols.

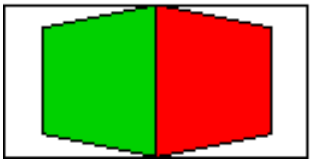
Solució:

a) El cub descrit és com es veu a la imatge, així doncs, l'única manera de veure les dues cares vermella i verda al viewport, tal i com descriu l'enunciat, és mirant-les des d'una direcció paral·lela al vector (1,0,1) i a alçada 1 per a què les cares quedin centrades. LLavors, posem el VRP per exemple a **VRP=(2,1,2)** que és el centre de l'aresta que uneix aquestes dues cares, i la càmera la posem per exemple a **OBS=(3,1,3)** (podria ser qualsevol punt (x,1,x) on x>=2. Com que a la imatge final hem de veure la meitat esquerra del rectangle de color verd i la meitat dreta de color vermell, això vol dir que la vertical de la càmera ha de ser la mateixa que la de l'escena, i per tant **up=(0,1,0)**.



Els paràmetres d'una òptica axonomètrica són el window (left, right, bottom, top), el Znear i el Zfar. El window, com que l'alçada de les cares que volem veure és de 2 unitats i la relació d'aspecte del viewport és de 2, podria ser **left=-2, right=2, bottom=-1, top=1**. El Znear ha d'estar per davant del cub, així que com que la distància de l'observador a l'aresta més propera és d'arrel(2), podem posar un **Znear=1**, per exemple, i el Zfar el podem posar darrera del tot del cub i per tant hauria de ser més gran que 3*arrel(2), per exemple **Zfar=5**.

b) Amb una càmera perspectiva, ben definida per a què es vegi el mateix window, el que observaríem és que les arestes que estan més lluny de l'observador es veuen més petites, i per tant, les arestes de la vora del que es veu del cub es veuran més petites que l'aresta que es veu al centre. L'efecte seria el mostrat a la figura.



2. *(1.5 punts)* Els mètodes `pintaPatricio()` i `pintaTaula()` envien a pintar la geometria dels models d'un Patricio i d'una taula. Les capses mínimes contenidores d'aquests models estan determinades pels punts: `(Patxmin, Patymin, Patzmin)`, `(Patxmax, Patymax, Patzmax)` i `(taulaxmin, taulaymin, taulazmin)`, `(taulaxmax, taulaymax, taulazmax)`. El Patricio mira cap a les Z+ i la taula està inicialment orientada de manera que les seves potes estan recolzades sobre el pla XZ (la taula està "dreta").

   Es vol visualitzar una escena formada per una taula amb dos Patricios al seu voltant. La taula tindra mida 4 en X, escalada uniformement, i estarà amb el centre de la seva base al punt (5, 0, 5) i els Patricios estaran situats al voltant de la taula (mirant cap a ella) de manera que el primer estarà mirant en direcció de les X- i l'altra en direcció de les X+. Els Patricios un cop ubicats han de fer mida 1 en X de l'escena. El primer Patricio haurà d'estar situat amb el centre de la seva base al punt (8, 0, 5), i el segon amb el centre de la seva base al punt (2, 0, 5).

   Suposant que la càmera està correctament inicialitzada (no cal que diguis res sobre la càmera), es demana que indiquis el pseudocodi d'una funció `pintaEscena()` que pinti l'escena tal com s'ha descrit, especificant com es troben les TGs indicades i utilitzant els mètodes `pintaTaula()` i `pintaPatricio()`. Justifica la resposta.

   Solució:

   La taula primer s'ha de portar a l'origen (portarem el centre de la base de la capsa contenidora). El centre de la base és: `CBTau=((taulaxmin+taulaxmax)/2.0, taulaymin, (taulazmin+taulazmax)/2.0)`. Després s'ha d'escalar uniformement per a què la seva mida en X sigui 4. El factor d'escala és doncs `Esctau=4.0/(taulaxmax-taulaxmin)`. Finalment hem de traslladar el centre de la base (que ara està a l'origen) al punt (5, 0, 5).

   Els dos Patricios s'han de portar també cadascun d'ells a l'origen (portarem també el centre de la base de la seva capsa contenidora). El centre de la base del Patricio és: `CBPat=((Patxmin+Patxmax)/2.0, Patymin, (Patzmin+Patzmax)/2.0)`. Tots dos també s'han d'escalar, però en aquest cas s'han d'escalar de manera que **la seva mida en X un cop ubicats en l'escena sigui 1**, això vol dir que el factor d'escala ha de ser `EscPat=1.0/(Patzmax-Patzmin)`, donat que la Z del model es correspondrà a la X del model un cop ubicat (girat).

   El primer Patricio s'ha de girar -90 graus respecte l'eix Y i portar-lo finalment a la posició (8,0,5). El segon Patricio s'ha de girar 90 graus respecte l'eix Y i portar-lo finalment a la posició (2,0,5).

   El codi de la rutina `pintaEscena()` serà el següent:

```
pintaEscena () {
   TGTau = Translate (5,0,5);
   TGTau = TGTau * Scale (Esctau,Esctau,Esctau);
   TGTau = TGTau * Translate (-CBTau);
   ModelMatrix (TGTau);
   pintaTaula();
   TGPat1 = Translate (8,0,5);
   TGPat1 = TGPat1 * Rotate (-90, 0,1,0);
   TGPat1 = TGPat1 * Scale (EscPat,EscPat,EscPat);
   TGPat1 = TGPat1 * Translate (-CBPat);
   ModelMatrix (TGPat1);
   pintaPatricio();
   TGPat2 = Translate (2,0,5);
   TGPat2 = TGPat2 * Rotate (90, 0,1,0);
   TGPat2 = TGPat2 * Scale (EscPat,EscPat,EscPat);
   TGPat2 = TGPat2 * Translate (-CBPat);
   ModelMatrix (TGPat2);
   pintaPatricio();
}
```

Examen parcial d'IDI 2015-2016, Q1. 4/11/2015 12:30

Nom i cognoms: Temps: 1h 20'

3. *(1 punt)* Ordena de forma correcta els processos del pipeline de visualització projectiu d'OpenGL, és a dir, en quin ordre afecten aquests processos a la primitiva que s'envia a pintar:

   a) 1) ProjectTransform; 2) ViewTransform; 3) ModelTransform; 4) Retallat;
   b) 1) ModelTransform; 2) ViewTransform; 3) ProjectTransform; 4) Retallat;
   c) 1) ModelTransform; 2) ViewTransform; 3) Retallat; 4) ProjectTransform;
   d) 1) ViewTransform; 2) ModelTransform; 3) ProjectTransform; 4) Retallat;

   Solució: b)

4. *(1 punt)* Tenim una càmera axonomètrica definida amb els paràmetres: OBS = (5,0,0), VRP = (0,0,0), up = (0,1,0), Window = (-2,2,-2,2), Znear = 2, Zfar = 8. Indica quins paràmetres definirien el mateix volum de visió considerant que l'observador passa a estar en OBS = (0,5,0). La visió de la imatge final no té perquè ser la mateixa i la relació d'aspecte del viewport no és rellevant.

   a) VRP = (0,1,0), up = (0,0,1), Window = (-3,3,-2,2), Znear = 2, Zfar = 8.
   b) VRP = (0,0,0), up = (1,0,0), Window = (-2,2,-2,2), Znear = 2, Zfar = 8.
   c) VRP = (0,2,0), up = (0,0,1), Window = (-3,3,-2,2), Znear = 3, Zfar = 7.
   d) VRP = (0,0,0), up = (0,0,-1), Window = (-2,2,-2,2), Znear = 3, Zfar = 7.

   Solució: c)

5. *(1 punt)* Volem ubicar un model en una posició concreta d'una escena que es visualitza amb una càmera correctament definida. Tal i com indica el codi següent, hem passat al vèrtex shader com uniforms les matrius següents: TG que permet ubicar el model, View Matrix (VM) i Project Matrix (PM). Completa la instrucció que permet calcular les coordenades d'un vèrtex de l'objecte respecte el sistema de coordenades de l'observador (SCO).

```
in vec3 vertex;
uniform mat4 TG, VM, PM;

void main(){
   vec4 vobs;
   vobs =
    ...
}
```

   a) `vobs = TG*VM*PM*vec4(vertex,1.0);`
   b) `vobs = VM*TG*vec4(vertex,1.0);`
   c) `vobs = TG*VM*vec4(vertex,1.0);`
   d) `vobs = VM*vec4(vertex,1.0);`

   Solució: b)

6. *(1 punt)* Tenim un objecte centrat a l'origen i amb capsa contenidora de mides 3 d'ample, 3 d'alçada i 3 de profunditat. Es vol modificar **només** la seva alçada per a què passi a ser 2, quina de les següents TG és la correcta?

   a) `TG = glm::scale (glm::mat4(1.f), glm::vec3(1.0, 2.0, 1.0));`
   b) `TG = glm::scale (glm::mat4(1.f), glm::vec3(3.0, 2.0, 3.0));`
   c) `TG = glm::scale (glm::mat4(1.f), glm::vec3(1.0, 2.0/3.0, 1.0));`
   d) `TG = glm::scale (glm::mat4(1.f), glm::vec3(2.0/3.0, 2.0/3.0,2.0/3.0));`

   Solució: c)

7. *(1 punt)* Tenim una càmera en primera persona correctament definida en posicionament i en òptica. El viewport és de 500x500. Quina de les següents afirmacions és correcta respecte a la relació d'aspecte (`ra`)?

   a) S'ha de modificar `ra` sempre que és modifiqui el viewport sigui quin sigui el tipus de l'òptica.
   b) S'ha de modificar `ra` només si es modifica el viewport i l'òptica és perspectiva.
   c) S'ha de modificar `ra` si es modifica la finestra gràfica encara que el viewport no es modifiqui.
   d) En la càmera en primera persona mai s'ha de modificar la relació d'aspecte de la càmera.

   Solució: a)

8. *(1 punt)* Tenim una escena en la que utilitzem el codi següent per ubicar la càmera. Quins serien els paràmetres OBS, VRP i up que permetrien definir la mateixa càmera? (no modifiquem l'òptica).

```
VM = Translació(0,0,-10);
VM = VM*Rotacio_z (90);
VM = VM*Rotació_y (-90);
VM = VM*Translació (10,-10,0)
ViewMatrix (VM);
```

   a) OBS = (0,10,0), VRP = (10,10,0), up = (0,0,1)
   b) OBS = (10,10,0), VRP = (0,10,0), up = (0,1,0)
   c) OBS = (10,10,0), VRP = (-10,10,0), up = (0,0,1)
   d) OBS = (0,10,0), VRP = (-10,10,0), up = (0,0,-1)

   Solució: d)

9. *(1 punt)* Quina de les següents afirmacions **és incorrecta**?

   a) Si tenim una càmera axonomètrica i reduïm el seu window (respectant la seva relació d'aspecte), estem fent un zoom-in.
   b) Si incrementem el FOV de la càmera perspectiva, haurem d'incrementar la relació d'aspecte, per a què el window mantigui la seva proporció.
   c) L'algorisme de retallat (clipping) és el mateix sigui la càmera perspectiva o axonomètrica.
   d) L'eix Y del sistema de coordenades de l'observador (SCO) sempre es projecta vertical (direcció Y) en el sistema de coordenades de dispositiu (SCD).

   Solució: b)