VIG. Examen parcial. Curs 06-07. Quadrimestre de Tardor Temps: 1h 30m.

Només es valoraran les respostes adequadament justificades

---

1. (1 punt) Suposem que tenim dos objectes de Qt: A i B que corresponen a classes definides per nosaltres i amb els seus signals i slots ben definits. Indica per a cada una de les següents connexions si són correctes o no i per què:
   a) connect (&A, SIGNAL(valueChanged(int)), &B, SLOT(close()));
   b) connect (&A, SIGNAL(pressed()), &B, SLOT(setTracking(bool)));
   c) connect (&A, SIGNAL(valueChanged(int)), &A, SLOT(setValue(int)));
   Nota: suposeu que els noms dels signals i slots i els tipus dels seus paràmetres són correctes.

   Sabem que la connexió entre signals i slots només es pot fer si els paràmetres d'aquests són compatibles, és a dir, que el/s tipus del/s valor/s que vol rebre un slot com a paràmetres ha/n de ser el/s mateix/os que el/s que emet el signal, i en el mateix ordre, però la signatura del slot pot ser més curta que la del signal.

   a) Correcte. El signal emet un valor de tipus int que el slot no necessita. No hi ha problema, el valor no s'utilitza.
   b) Incorrecte. El slot necessita rebre un paràmetre de tipus bool que el signal no emet.
   c) Correcte. El slot necessita un paràmetre de tipus int que és emès pel signal.

2. (1 punt) Suposem que tenim una escena en què la seva esfera contenidora està centrada a l'origen i té com radi 50. Inicialment tenim una definició de càmera donada per les sentències següents d'OpenGL:

```
glMatrixMode (GL_PROJECTION);
glLoadIdentity ();
gluPerspective (60, 1, 50, 150);
glMatrixMode (GL_MODELVIEW);
glLoadIdentity ();
glTranslatef (0, 0, -100);
```

   Quins paràmetres ens caldria canviar d'aquesta definició de càmera si volem realitzar un *zoom* apropant l'observador cap el centre de l'esfera?

   Els paràmetres que cal modificar per a apropar l'observador al centre de l'escena són:

   - *la distància de l'observador al centre de l'esfera contenidora, de manera que aquesta disminueixi segons el zoom que es vol fer.*
   - *els plans de retallat anterior i posterior per a que no retallin sinó que s'ajustin al que cal visualitzar.*

   Suposant que 'zoom' és la distància que volem apropar l'observador al centre de l'esfera, i presuposant que aquest 'zoom' no supera la distància actual de l'observador al VRP, els canvis en el codi serien:

```
glMatrixMode (GL_PROJECTION);
glLoadIdentity ();
gluPerspective (60, 1, max(0.001,50-zoom), 150-zoom);
glMatrixMode (GL_MODELVIEW);
glLoadIdentity ();
glTranslatef (0,0,-(100-zoom));
```

3. (2 punts) Disposem del model d'un personatge que té com a capsa contenidora la capsa definida pels punts MinCapsa=(10,10,10) i MaxCapsa=(20,20,12). Ens diuen que la part del davant d'aquest personatge és la que mira cap a la part positiva de l'eix Z.

   Considerem que tenim una habitació dividida en 10x10 cel·les quadrades de costat c=1 situades en el pla XZ (la cel·la x=1, z=1 té coordenades mínimes xmin=0 i zmin=0). Quines transformacions li haurem de fer al personatge si el volem situar centrat i escalat adientment en la cel·la x=5, z=3 de la nostra habitació i mirant cap a la cel·la x=5, z=4? Escriu el tros de codi amb les transformacions en OpenGL que li aplicaries. L'alçada del terra de l'habitació (sobre el qual cal posar el personatge) és Y=0.

   - *Donat que glScale realitza l'escalat respecte de l'origen de coordenades, primer de tot caldrà moure el personatge a l'origen de coordenades. Una possibilitat es moure el centre de la seva capsa, que és el (15,15,11, i, per tant, caldrà aplicar-li una translació de (-15,-15,-11) (una altra solució seria moure el centre de la seva base).*

   - *La capsa del personatge té dimensions 10x10x2, per tant li caldrà un escalat uniforme –per a no deformar- de 1/10 per a que hi càpiga a la cel·la de 1x1. 1/10 és l'escalat màxim a realitzar considerant les dimensions en x i en z.*

   - *Com que el personatge "mira" cap a les Z positives i el volem posar també "mirant" cap a les Z positives (situat a la cel·la x=5, z=3 i "mirant" cap a la cel·la x=5, z=4), no li caldrà cap rotació.*

   - *Finalment caldrà moure el personatge des de l'origen de coordenades on està centrat a la posició final que es vol. Cal fixar-se que per a situar el personatge sobre l'alçada del terra de l'habitació (Y=0) caldrà "pujar-lo" la meitat de l'altura de la seva capsa un cop escalada (5*1/10 = 0.5). Així doncs, la posició on cal moure el personatge en aquest punt és (4.5, 0.5, 2.5) que correspon al punt mig de la cel·la x=5, z=3 i a pujar l' altura de 0.5.*

   El tros de codi d'OpenGL que cal per a fer aquestes transformacions serà:

```
glTranslatef (4.5, 0.5, 2.5);
glScalef (0.1, 0.1, 0.1);
glTranslatef (-15.0, -15.0, -11.0);
```

4. (1 punt) Una llum de color rgb:(1.0,0.5,0.0) brilla a través d'un filtre de color cyan (que sols deixa passar la llum de color cyan), i il·lumina un paper blanc. De quin color es veurà el paper sota aquesta llum?

   *Com que el filtre és de color cyan, d'una llum (r,g,b) incident, passarà (0, x*g, x*b), on x és un valor entre 0 i 1 que indica el grau de transparència del filtre. De la llum indicada a l'enunciat - (1,0.5,0), o sigui un color taronja- per tant, passarà (0, 0.5, 0). Per tant, el paper es veurà d'aquest color (perquè essent blanc, es veurà del mateix color que la llum que l'il·lumina): verd fosc.*

5. (2 punts) Estem inspeccionant un objecte modelat amb molts triangles i ens volem fixar en una zona quasi plana (els triangles d'aquesta zona estan tots quasi en el mateix pla). La zona té unes dimensions de 4 x 7 mm. La interfície ens dóna el punt mig de la zona **P**, i el vector normal del seu pla aproximat, **n**. La distància màxima entre aquest pla aproximat i els vèrtexs dels triangles de la zona que volem inspeccionar és d'1 mm. Especifica una càmera que ens permeti veure bé la zona que volem inspeccionar.

   - *El VRP el situarem al centre de la zona per a que aquesta quedi centrada en el viewport: **VRP** = **P.***
   - *Per a veure la zona més acuradament, situem la direcció de visió perpendicular a ella:*
     OBS**:= **VRP**+dn**; podem posar una d=8 que estarà fora de la zona a inspeccionar (de gruix 2mm)*
   - *Com a **VUV** podem agafar un vector que tingui la direccio d'una de les 4 fronteres de la zona. Aquella que volguem que surti vertical en pantalla. De fet serveix qualsevol vector que indiqui una direcció diferent a **n**.*

Suposant una càmera perspectiva:

- *Zprop = d – 1; Zlluny = d + 1; per a no tallar la zona i que el frustum estigui ajustat a ella.*
- *Relació d'aspecte: ar= arv per a no tenir deformació (arv és la relació d'aspecte de la vista)*
- *Si volem veure vertical el costat de 4mm, requeriríem una window amb relació d'aspecte arz=7. /4. i un angle mínim d'obertura de la càmera de valor alfa=arc tg (2./d) (FOV=2*alfa).*

Com que la relació d'aspecte ar ha de ser la de la vista, hem de modificar, si s'escau, alfa, per a poder continuar veient tota la zona – sense deformació-,

```
Si ar > arz llavors
    alfa := arc tg (2./d)
sino
    altura := 7./(2.*ar)
    alfa := arc tg (altura/d)
fiSi
```

6. (1 punt) Estem inspeccionant una escena i definim els paràmetres de la càmera a partir de la seva esfera contenidora. Quin inconvenient pot tenir fixar el valor de la relació d'aspecte de la càmera a 1?

Es produeix una deformacio de la imatge sempre que la relació d'aspecte de la vista no sigui 1. Això es degut a que la transformació Mon-Dispositiu transforma tot el que es projecta a la finestra del pla de projecció, sobre tota la vista. Si fixem la relació d'aspecte de la càmera a 1, la imatge es projecta sobre una finestra quadrada al pla de projecció i després es deforma durant la TMD per tal d'adaptar-se a la forma de la vista.

Només podem evitar la deformació si la relació d'aspecte és la de la vista, que l'usuari pot modificar interactivament.

7. (1 punt) Quina informació d'entrada requereix i quina informació de sortida genera l'algorisme de rasterització de polígons?

Entrada: Vèrtexs (en Sistema de Coord. de Dispositiu) del polígon i arestes del mateix. O bé, només una llista ordenada de vèrtexs que defineixi les arestes de forma implícita.

Sortida: Els fragments que corresponen a tots els píxels interiors al polígon.

8. (1 punt) Tenim una càmera perspectiva definida per OBS=(0., 50., 0.), VRP=(0., 0., 0.), VUV=(0., 0., 1.) i tal que l'angle alfa d'obertura de la càmera és de 45 graus. Indica (i justifica) quines podrien ser les coordenades (en el sistema de coordenades de l'aplicació) d'un punt que veiem al centre de la vora superior de la vista. (Per si ho necessites: sin(45) ≅ 0.707; cos(45) ≅ 0.707; tangent(45) = 1).

El sistema de coordenades de l'observador (SCO) té els eixos:
- Eix x: en direcció negativa de l'eix x del SCA
- Eix y: en direcció positiva de l'eix z del SCA
- Eix z: en direcció positiva de l'eix y del SCA

*El punt que veiem al centre de la vora superior de la vista sempre té coordenada x=0., respecte del SCO. Per tant, es troba al pla y-z en el SCO, que en el nostre cas (veure definicio del SCO) també és el pla y-z en el SCA. Com que es troba a la cara superior del frustum, s'acompleix (en coordenades del SCA) que z=(50-y)*tangent(alfa)=50-y ja que la tangent de alfa és 1.*

El punt pot ser el (0., 0., 50.), o qualsevol altre punt de la recta que va de l'observador a aquest punt: (0., y, 50.-y). Cal només tenir en compte que 50-y ha de ser un valor comprès entre els paràmetres Zprop i Zlluny de la càmera.