Examen Final de Gràfics Curs 2012-13 Q1

Nom i Cognoms:

Tots els exercicis tenen el mateix pes.

Exercici 1

Aquí teniu una llista d'etapes/tasques del pipeline gràfic, ordenades per ordre alfabètic. Torna-les a escriure a la dreta, però ordenades segons l'ordre al pipeline gràfic:

- Alpha blending
- Depth test
- Fragment Shader (FS)
- Geometry Shader (GS)
- Interpolació
- Rasterització
- Vertex Shader (VS)

Vertex Shader (VS)

Geometry Shader (GS)

Rasterització

Interpolació

Fragment Shader (FS)

Depth test

Alpha blending

Exercici 2

Elimina del següent vèrtex shader el màxim de línies possible sense que canviï la imatge resultant, tenint en compte que s'utilitzarà només amb el fragment shader de sota (escriu una marca ✓ al costat de les línies necessàries i una marca ✗ al costat de les que no calen).

```
// VS

varying vec3 vNormal;                                   ✗
void main() {     ✓
   vNormal = gl_NormalMatrix * gl_Normal;               ✗
   gl_Position = gl_ModelViewProjectionMatrix * gl_Vertex;
   gl_FrontColor  = gl_Color;                           ✗
   gl_TexCoord[0] = gl_MultiTexCoord0;                  ✗
}  ✓

// FS
void main() { gl_FragColor = vec4(1.0); }
```

Exercici 3

Hem tingut un petit accident amb el tablet i s'ha trencat la pantalla. Afortunadament, la meitat superior de la pantalla encara és utilitzable. De cara a utilitzar aplicacions OpenGL (a pantalla completa), una opció és definir un viewport més petit que ocupi només la part superior de la pantalla. La opció que us demanem, però, és completar un VS que faci que tots els vèrtexs dins el frustum de visió de la càmera es projectin a la meitat superior de la pantalla (no patiu per la relació d'aspecte).

```
void main() {
   vec4 P;
   P = gl_ModelViewProjectionMatrix * gl_Vertex;
   P= vec4(P.xyz / P.w, 1.0);
   // cal traslladar i escalar en Y
   P.y += 1.0;
   P.y /= 2.0;
   gl_Position = P;
}
```

Exercici 4

Completa el següent FS per tal que calculi correctament la contribució especular:

```
varying vec3 normal; // eye space
varying vec3 pos;  // pos en eye space

vec4 lightSource( vec3 N, vec3 V, gl_LightSourceParameters light ) {
   vec3 L = normalize( light.position.xyz - V );
   vec3 R = normalize( 2.0*dot(N,L)*N-L );
   V = normalize (-V.xyz);
   float diff = max( 0.0, dot( N,L ) );
   float spec; // cal que el calculeu vosaltres
   float RdotV = max( 0.0, dot( R,V ) );
   spec = pow( RdotV, gl_FrontMaterial.shininess );

  return gl_FrontMaterial.diffuse  * light.diffuse  * diff +
         gl_FrontMaterial.specular * light.specular * spec;
}
```

Exercici 5

Una forma de comprovar visualment les parts on una malla de triangles parametritzada *M* no és bijectiva és dibuixar *M* amb un vertex shader que escrigui gl_Position transformant a clip space les coordenades de textura de cada vèrtex (com a l'exercici *UV Unfold*). Si la parametrització té parts no bijectives, aquestes estaran representades per triangles que es veuran parcialment solapats. Indica una tècnica disponible en el pipeline d'OpenGL que ens permeti destacar visualment les parts de *M* on no es preserva la bijectivitat (és a dir, que permeti destacar les zones de la malla projectada on s'hi projecta més d'un triangle, canviant el color segons el nombre de triangles que s'hi projecten). La vostra solució només ha de requerir un pas de rendering. Indica les crides OpenGL que caldria fer servir.

Alpha blending; Exemple:

glEnable(GL_BLEND); glBlendFunc(GL_SRC_ALPHA, GL_ONE); Les primitives es dibuixen de color (1, 1, 1, 0.2). El nivell de gris del resultat indica el nombre de fragments que s'hi han projectat. Si és <=0.2 -> bijectiu; >=0.4 -> no bijectiu.

Exercici 6

Volem assignar al stencil un valor 1 als pixels corresponents als fragments visibles d'un rectangle (per exemple, pel primer pas de l'algorisme de simulació d'ombres per projecció amb stencil buffer). Completa el següent codi perquè faci aquesta tasca:

glEnable(GL_STENCIL_TEST);
glStencilFunc(_____, 1, 1); GL_ALWAYS
glStencilOp(GL_KEEP, GL_KEEP, _____); GL_REPLACE
drawRectangle();

Exercici 7

Volem dibuixar les ombres que projecten els objectes sobre el pla horitzontal Y= -2, amb la tècnica d'ombres per projecció. Suposant una llum infinitament allunyada en direcció de l'eix Y, indica per quina matriu 4x4 cal multiplicar la posició del vertex en world space per tal d'obtenir les ombres projectades al dibuixar els objectes.

Volem que (x,y,z) es projecti a (x, -2, z). Per tant la única diferència amb la matriu identitat serà la segona fila, que serà (0, 0, 0, -2):

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Exercici 8

Disposem d'una funció sampleSphereMap que, donat un vector unitari R, i una textura que representa un sphere map, ens retorna el color de l'entorn en la direcció R. Completa el següent shader per tal que implementi sphere mapping, amb els càlculs en eye space:

```
uniform sampler2D spheremap;
varying vec3 P; // posició en eye space
varying vec3 N; // normal en eye space

vec4 sampleSphereMap(sampler2D sampler, vec3 R); // funció que podeu cridar

void main()
{
   vec3 R;
```

```
   vec3 I = normalize(P);
   R = reflect(I, N);
```

```
   gl_FragColor = sampleSphereMap(spheremap, R);

}
```

Exercici 9

Suposant que drawQuad() és una funció que dibuixa un rectangle que ocupa tot el viewport, i assumint l'estat habitual d'OpenGL (sense shaders activats), indica quin serà el color RGB resultant d'aplicar aquest codi:

```
const double a = 0.0;
glClearColor(0.0, 0.0, 0.0, a);
glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
glDisable(GL_DEPTH_TEST);
glEnable(GL_BLEND);
glBlendFunc(GL_SRC_ALPHA, GL_ONE);
glColor4f(1.0, 0.0, 0.8, 0.5);
drawQuad();
glColor4f(0.5, 1.0, 0.0, 0.8);
drawQuad();
```

(0.5+0.4, 0+0.8, 0.4) = (0.9, 0.8, 0.4)

Exercici 10

Volem proporcionar a l'usuari la següent funcionalitat: l'usuari dibuixarà un rectangle en espai imatge, i l'aplicació identificarà i seleccionarà tots els triangles de l'escena la projecció dels quals sigui completament interior al rectangle especificat. Podem fer servir la tècnica de selecció basada en la lectura del buffer de color que hem vist a classe?

NO

En cas afirmatiu, descriu com ho faries. En cas negatiu, justifica la resposta.

Només permet identificar els visibles i tampoc permet distingir si la projecció cau completament dins del rectangle.

Exercici 11

Indica quins light paths permet simular el model d'il·luminació d'OpenGL.

LDE, LSE

Exercici 12

Quina magnitud de radiometria/fotometria, mesura la quantitat total d'energia per unitat de temps i àrea que arriba al sensor CCD d'una càmera?

irradiància

Indica una unitat de mesura per la magnitud anterior

W/m2 lux

Exercici 13

Quines equacions permeten calcular la proporció de llum reflectida i la proporció de llum transmesa, quan la llum incideix en la superfície de separació entre dos medis?

Les equacions de **Fresnel**

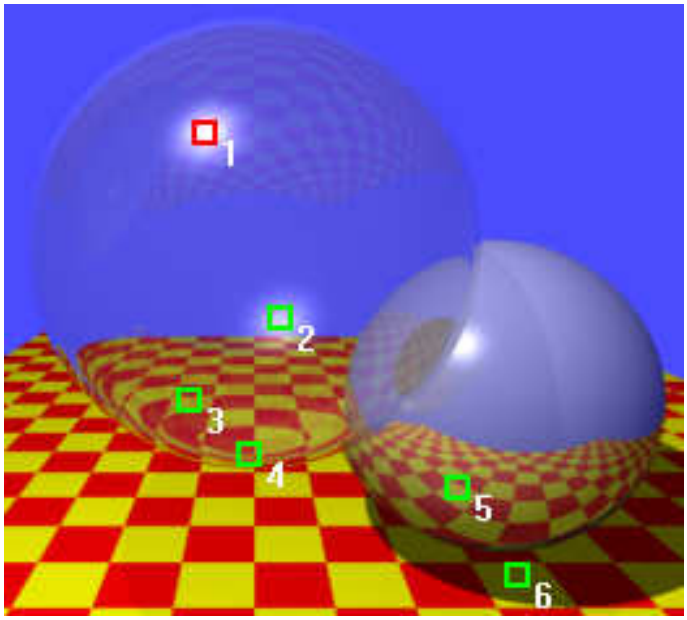
Exercici 14

Indica com detectar ràpidament que un raig amb origen al punt P i direcció **v** no intersecta un pla amb normal **n** (suposeu que P no pertany al pla).

dot(v,n) = 0 → raig paral·lel al pla → no intersecta

Exercici 15

Per què l'esfera de l'esquerra té dues taques especulars indicades amb 1 i 2?



Perquè l'esfera té una cavitat interna i per tant té una superfície interna que causa la segona taca.

La taca especular 2 no està ben bé a la posició correcta. Indica quin segment del camí LSSSE implicat s'ha calculat sense tenir en compte la refracció de la llum.

El segment LS (el shadow ray des de la cara interna a la font de llum ignora refraccions)

Preguntes per a l'avaluació de les competències transversals

Pregunta 1

A què fa referència el terme *haptic*?

(a) Interacció en aplicacions mèdiques
(b) Interacció explícita
(c) Manipulació d'objectes deformables
(d) Realimentació tàctil

Pregunta 2

Indica al menys un parell de tècniques d'adquisició de dades de volum en medicina.

CT, MR, ultrasons...