Examen Final de Gràfics Curs 2017-18 Q1

Nom i Cognoms:

Tots els exercicis tenen el mateix pes.

Exercici 1

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

- Depth test
- Fragment Shader
- Geometry shader
- Rasterització

Geometry shader
- Rasterització
- Fragment Shader
- Depth test

Exercici 2

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

- Clipping
- Divisió de perspectiva
- Rasterització
- Vertex shader

- Vertex shader
- Clipping
- Divisió de perspectiva
- Rasterització

Exercicis 3, 4, 5 i 6

Indica quina és la matriu (o **producte de matrius**) que aconsegueix la conversió demanada**, usant la notació següent** (vigileu amb l'ordre en que multipliqueu les matrius):

M = modelMatrix V = viewingMatrix P = projectionMatrix N = normalMatrix

M^{-1} = modelMatrixInverse V^{-1} = viewingMatrixInverse P^{-1} = projectionMatrixInverse I = Identitat

a) Pas d'un vèrtex de eye space a world space V^{-1}

b) Pas d'un vèrtex de clip space a world space $V^{-1} * P^{-1}$

c) Pas d'un vèrtex de object space a clip space

$P * V * M$

d) Pas d'un vèrtex de object space a model space I

e) Pas d'un vèrtex de object space a world space M

f) Pas d'un vèrtex de world space a eye space V

g) Pas de la normal de model space a eye space N

h) Pas d'un vèrtex de eye space a clip space P

Exercici 7

Indica, per cadascuna de les següents tècniques basades en textures, si sempre requereixen (SI) o no (NO) accedir a un *height field*:

(a) Color mapping NO

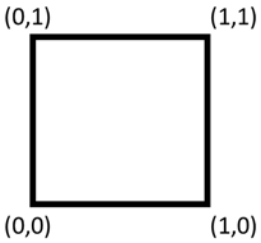
(b) Relief mapping SI

(c) Parallax mapping SI

(d) Displacement mapping SI

Exercici 8

Amb la imatge de l'esquerra, volem texturar el quad del mig, per obtenir la imatge de la dreta:



Completa el següent VS per obtenir el resultat desitjat:

```
void main() {


      vtexCoord = vec2(-1,1)*texCoord;


   glPosition = vec4(vertex, 1.0);
}
```

Exercici 9

Sigui F(u,v) un height field. Indica una tècnica vista a classe que faci servir el gradient de F(u,v).

Bump mapping / Normal mapping

Exercici 10

Indica, per cada path en la notació estudiada a classe, L(D|S*E, si és simulat (SI o no (NO per la tècnica de *Two-pass raytracing*:

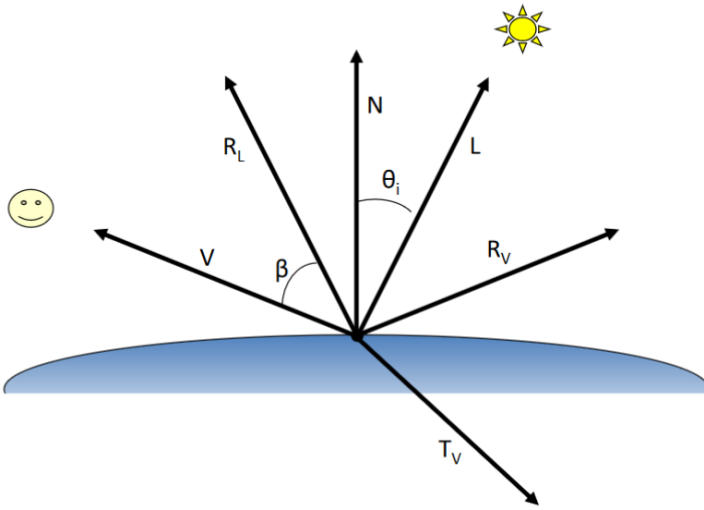
(a) LSSDSSE I

(b) LDE SI

(c) LSE SI

(d) LDDSE NO

Exercicis 11 i 12



Amb la notació de la figura, indica, en el cas de Ray-tracing

(a) Quin vector té la direcció del *shadow ray*?

L

(b) Quin vector és paral·lel al raig reflectit?

R_v

(c) Què dos vectors determinen la contribució de

Lambert?L i N

(d) Quin vector depèn de l'índex de refracció?

Tv

Exercici 13

Aquí teniu l'equació d'obscuràncies:

$$W(P,N) = \frac{1}{\pi}\int_{\Omega} \rho(d(P,\omega)) \cdot (N \cdot \omega) d\omega$$

Què representa ρ ? Una funció que decreix segons la distància

(b) Com hauria de ser ρ per obtenir oclusió ambient? Funció constant ρ = 1

Exercici 14

Tenim un cub representat amb una malla triangular formada per 8 vèrtexs i 12 triangles. Volem construir un VBO per representar aquest cub, de forma que el VS rebi com a atributs les coordenades (x,y,z del vèrtex i les components del vector normal (nx,ny,nz), **sense cap suavitzat d'aresta** (volem que el cub aparegui il·luminat correctament). Quants vèrtexs necessitem representar al VBO?

12 tri * 3 v/tri = **36 vèrtexs** (hem de repetir vèrtexs perquè no comparteixen normals)

També és possible fer-ho només amb **24 vèrtexs** (cadascú dels 8 vèrtexs del cub només ha d'aparèixer amb 3 normals diferents; els dos triangles de cada cara poden re-usar un parell de vèrtexs).

Exercici 15

Indica clarament la línia on ens podria ser útil un *environment map*:

```
funció traçar_raig(raig, escena, μ)
  si profunditat_correcta() llavors
    info:=calcula_interseccio(raig, escena)
    si info.hi_ha_interseccio() llavors
      color:=calcular_I_D(info,escena); // I_D
      si es_reflector(info.obj) llavors
         raigR:=calcula_raig_reflectit(info, raig)
         color+= K_R*traçar_raig(raigR, escena, μ) //I_R
      fsi
      si es_transparent(info.obj) llavors
         raigT:=calcula_raig_transmès(info, raig, μ)
         color+= K_T*traçar_raig(raigT, escena, info.μ) //I_T
      fsi
    sino color:=colorDeFons
    fsi
  sino color:=Color(0,0,0);  // o colorDeFons
  fsi
  retorna color
ffunció
```

Exercici 16

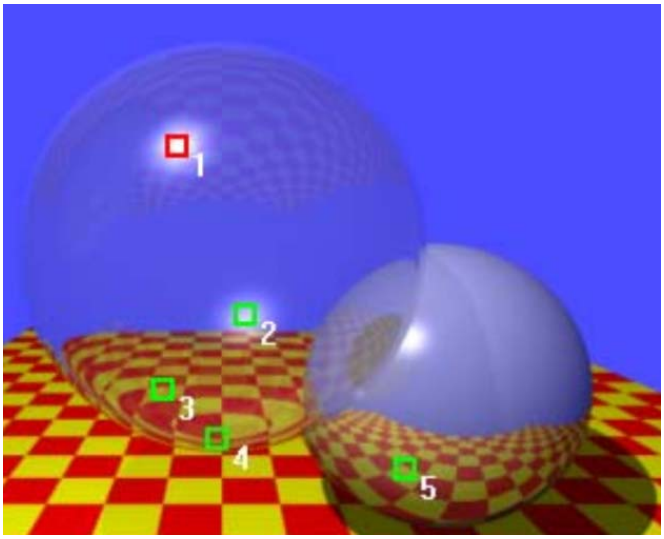
Completa aquest fragment shader que implementa la tècnica de Shadow mapping:

```
uniform sampler2D shadowMap;
uniform vec3 lightPos;
in vec3 N;
in vec3 P;
in vec4 vtexCoord; // coordenades de textura en espai
homogeni out vec4 fragColor;

void main()
{
    vec3 L =  normalize(lightPos – P);     float NdotL =
max(0.0, dot(N,L));       vec4 color = vec4(NdotL);

    vec2 st =       vtexCoord.st/vtexCoord.q;

float storedDepth = texture(shadowMap, st).r;

float trueDepth = vtexCoord.p / vtexCoord.q;

    if (trueDepth <= storedDepth) fragColor
= color;     else fragColor = vec4(0);
}
```

Exercici 17

Considerant aquesta figura generada amb ray-tracing,

(a) Quin és el *light path* dominant que explica el color del píxel numerat amb un “1”?
LSE
(b) Quin és el *light path* dominant que explica el color del píxel numerat amb un “5”? LSDE



Exercici 18

Indica quantes vegades cal pintar l’escena en les següents tècniques:

a) Shadow mapping, suposant que la llum és dinàmica

2 cops

b) Reflexió amb objectes virtuals, amb stencil (ignoreu els passos en que només es dibuixa el mirall):

2 cops