

# GRAFICS-EXAMENS.pdf



Arnau\_FIB



Gráficos



3º Grado en Ingeniería Informática



Facultad de Informática de Barcelona (FIB)  
Universidad Politécnica de Catalunya

**Encuentra el trabajo  
de tus sueños**

Participa en retos y competiciones de programación

Ten contacto de calidad con empresas líderes en el sector tecnológico mientras vives una experiencia divertida y enriquecedora durante el proceso.

**Únete ahora**



Escanéame y  
obtén más info!!

**NUWE**



# Encuentra el trabajo de tus sueños



Escanéame y obtén más info!!



## Examen Final de Gràfics

Curs 2016-17 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.

- Alpha Blending
- Geometry shader
- Rasterització
- Stencil Test

- Geometry shader
- Rasterització
- Stencil Test
- Alpha Blending

### 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.

- Divisió de perspectiva
- Fragment shader
- Pas a Clip Space
- Rasterització

- Pas a clip space
- Divisió de perspectiva
- Rasterització
- Fragment shader

### Exercici 3

El LOD 2 d'una textura té 256 x 128 texels. Quina mida té el LOD 0 d'aquesta mateixa textura?

**LOD 0 → 1024 x 512** (LOD 1 → 512 x 256; LOD 2 → 256 x 128)

**WUOLAH**

#### Exercici 4

En quin espai de coordenades estan x, y en les crides GLSL `dFdx`, `dFdy`?

Window space.

#### Exercicis 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):

$M = \text{modelMatrix}$	$M^{-1} = \text{modelMatrixInverse}$
$V = \text{viewingMatrix}$	$V^{-1} = \text{viewingMatrixInverse}$
$P = \text{projectionMatrix}$	$P^{-1} = \text{projectionMatrixInverse}$
$N = \text{normalMatrix}$	$I = \text{Identitat}$

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

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

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

d) Pas de la normal de object space a eye space      N

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

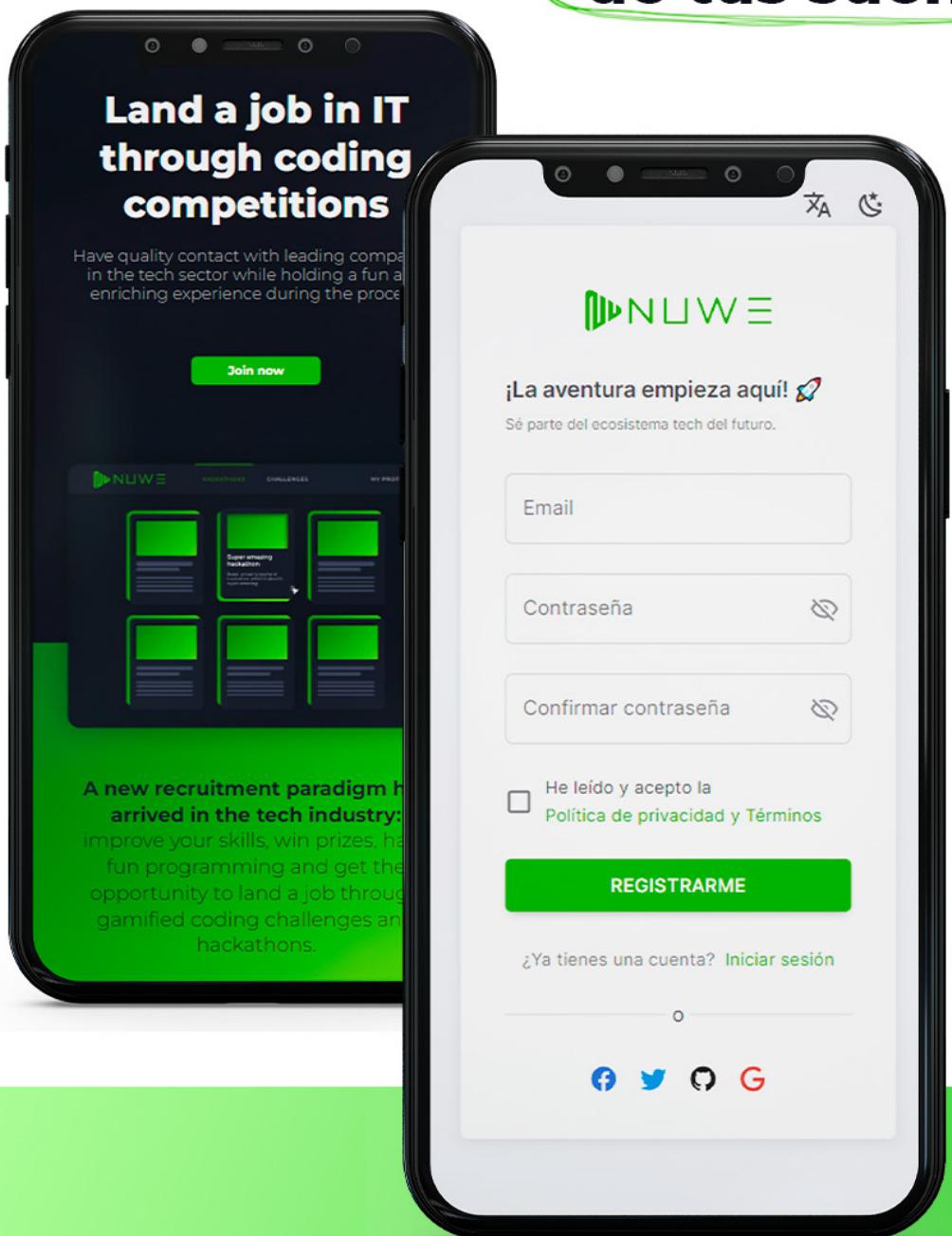
f) Pas d'un vèrtex de eye space a world space       $V^{-1}$

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

h) Pas d'un vèrtex de object space a clip space       $P * V * M$



Encuentra el trabajo  
de tus sueños



### Participa en retos y competiciones de programación

Ten contacto de calidad con empresas líderes en el sector tecnológico mientras vives una experiencia divertida y enriquecedora durante el proceso.

Únete ahora



Escanéame y obtén más info!!

### **Exercici 7**

Hem produït un fragment amb color  $\text{RGBA} = (1.0, 0.5, 0.0, 0.2)$  corresponent al pixel  $(i,j)$ . El color  $\text{RGB}$  del pixel  $(i,j)$  al buffer de color és  $(0.3, 0.1, 0.3, 0.7)$ . Indica com hem de configurar la funció de blending si volem que el resultat sigui el color  $\text{RGB} (0.5, 0.2, 0.3)$ . Justifica la resposta.

`glBlendFunc(...`

$$(1, 0.5, 0) \times \text{sfactor} + (0.3, 0.1, 0.3) * \text{dfactor} = (0.5, 0.2, 0.3)$$

$$\text{dfactor} = 1, \text{sfactor} = 0.2$$

### **Exercici 8**

Escriu l'equació general del rendering, amb la formulació vista a classe, indicant clarament el tipus de radiància als diferents termes.

### **Exercici 9**

Indica, en la notació estudiada a classe,  $L(D|S)^*E$ , quins light paths són suportats per:

(a) Raytracing clàssic

(b) Path tracing

### **Exercici 10**

Volem generar amb RayTracing una imatge 256x256 pixels d'una escena interior tancada. Quants shadow rays caldrà traçar, si tenim dos fonts de llum i els materials són difosos i opacs?

## Exercici 11

Completa el següent FS per tal que calculi correctament el terme espectral (Phong) de la il·luminació:

```
uniform vec4 matDiffuse, matSpecular, lightDiffuse, lightSpecular;
uniform float matShininess;

vec4 light(vec3 N, vec3 V, vec3 L)
{
    vec3 R = normalize( 2.0*dot(N,L)*N-L );
    float NdotL = max( 0.0, dot( N,L ) );
    float Idiff = NdotL;
    float Ispec = 0;

    if (NdotL>0)
    {
        float myDot = .....max( 0.0, dot( R,V ) )

        Ispec = .....pow( myDot, matShininess );
    }

    return
        matDiffuse * lightDiffuse * Idiff +
        matSpecular * lightSpecular * Ispec;
}
```

## Exercicis 12 i 13

Completa aquest codi, corresponent als dos primers passos de l'algorisme de simulació d'ombres per projecció, amb stencil:

```
// 1. Dibuixa el receptor al color buffer i al stencil buffer
glEnable(GL_STENCIL_TEST);
glStencilFunc(GL_ALWAYS,1,1);
glStencilOp (GL_KEEP,GL_KEEP,GL_REPLACE);
dibuixa(receptor);

// 2.Dibuixa oclusor per netejar l'stencil a les zones a l'ombra
glDisable(GL_DEPTH_TEST);
glColorMask(GL_FALSE,...GL_FALSE);
glStencilFunc(GL_EQUAL, 1, 1);
glStencilOp (GL_KEEP, GL_KEEP, GL_ZERO);
glPushMatrix();
glMultMatrixf(MatriuProjeccio);
dibuixa(oclosor);
glPopMatrix();
```

# Encuentra el trabajo de tus sueños

Participa en retos y competiciones de programación



Escanéame y  
obtén más info!!

## Exercici 14

Indica com varia l'extensió de la penombra en els següents casos:

- (a) La llum es puntual

No hi ha penombra

- (b) Apropem oclusor i receptor de l'ombra

Disminueix la penombra.

## Exercici 15

glPolygonOffset(factor,units) afegeix un offset a la z en window space d'acord a la següent fórmula:

$$Dz * \text{factor} + r * \text{units}$$

Com es calcula Dz i què representa?

$$Dz = \max(\frac{\partial z}{\partial x}, \frac{\partial z}{\partial y})$$

Representa quan tangencial és la primitiva a la direcció de visió.

## Exercici 16

Tenim una aplicació que no suporta MipMapping, però volem simular el resultat de GL\_LINEAR\_MIPMAP\_LINEAR en GLSL. Completa aquest codi, on lambda és un float amb el nivell de LOD (no necessàriament enter) més adient pel fragment:

```
vec4 sampleTexture(sampler2D sampler, vec2 texCoord, float lambda)
{
    vec4 color0 = textureLod(sampler, texCoord, floor(lambda));
    vec4 color1 = textureLod(sampler, texCoord, floor(lambda)+1);
    return mix(color0, color1, fract(lambda));
```

}

Podeu assumir que textureLod(P,sampler,lod) fa un accés bilineal a textura al punt P usant el nivell especificat a lod.



### Exercici 17

Què fa aquesta matriu?

$$\begin{bmatrix} 1-2a^2 & -2ba & -2ca & -2da \\ -2ba & 1-2b^2 & -2cb & -2db \\ -2ca & -2cb & 1-2c^2 & -2dc \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- (a) Projecció respecte una font posicional situada al punt (a,b,c)  
(b) Projecció respecte una font direccional situada al punt homogeni (a,b,c,d)  
(c) Reflexió respecte un pla (a,b,c,d)  
(d) Projecció ortogonal sobre el pla (a,b,c,d)
- (c); n'hi ha prou amb provar amb el pla (0,1,0,0)

### Exercici 18

Indica en quin interval (el més ajustat possible) poden estar les coordenades Z dels punts interiors a la piràmide de visió d'una càmera perspectiva, segons l'espai considerat. Pots fer servir  $\pm\infty$  si s'escau.

- Object space:  $(-\infty, +\infty)$
- Eye space:  $(-\infty, 0)$
- NDC:  $[-1, 1]$
- Window space:  $[0, 1]$

Per alinear (establir la correspondència espacial) els models de volum obtinguts amb les diferents modalitats de captació d'imatge mèdica (MRI, CT...)

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.

- Fragment Shader
- Geometry shader
- Rasterització
- Stencil test

- Geometry shader
- Rasterització
- Fragment Shader
- Stencil 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.

- dFdx, dFxy
- Divisió de perspectiva
- Rasterització
- Vertex shader

- Vertex shader
- Divisió de perspectiva
- Rasterització
- dFdx, dFxy

### Exercici 3

Sigui  $F(u,v)$  un *height field*. Si volem aplicar la tècnica de *bump mapping*, indica clarament què podem emmagatzemar per cada texel del bump map:

- (a) Si només disposem d'una textura amb un canal

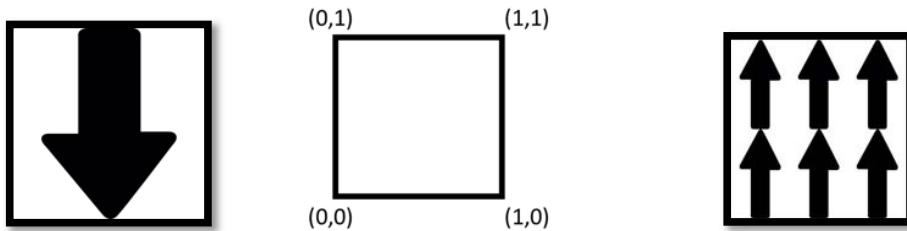
Directament  $F(u,v)$

- (b) Si disposem d'una textura amb dos canals

Gradient de  $F(u,v)$ :  $dF/du, dF/dv$

### Exercici 4

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(3,-2)*texCoord;  
  
    glPosition = vec4(vertex, 1.0);  
}
```

### Exercici 5

Tenim un FS que aplica una textura a l'objecte. Indica clarament quin efecte té incrementar el valor del uniform offset en la imatge resultant (suposa mode GL\_REPEAT):

```
uniform int offset = 0;  
  
...  
gl_FragColor = texture(sampler, vtexcoord + vec2(float(offset)))
```

Cap, ja que és un enter, i amb GL\_REPEAT només s'utilitza la part fraccionària de les coord de textura.

# Encuentra el trabajo de tus sueños

Participa en retos y competiciones de programación



Escanéame y  
obtén más info!!

## Exercicis 6, 7, 8 i 9

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

$M^{-1}$  = modelMatrixInverse

V = viewingMatrix

$V^{-1}$  = viewingMatrixInverse

P = projectionMatrix

$P^{-1}$  = projectionMatrixInverse

N = normalMatrix

I = Identitat

a) Pas de la normal de object space a eye space N

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

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

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

e) Pas d'un vèrtex de object space a clip space  $P * V * M$

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

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

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

## Exercici 10

Indica, en la notació estudiada a classe,  $L(D|S)^*E$ , quins light paths són suportats per:

(a) Raytracing clàssic

$LDS^*E, LS^*E$

(b) Path tracing

$LS^*DS^*E$



### Exercici 11 i 12

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

- (a) Quin vector és paral·lel al raig primari

V

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

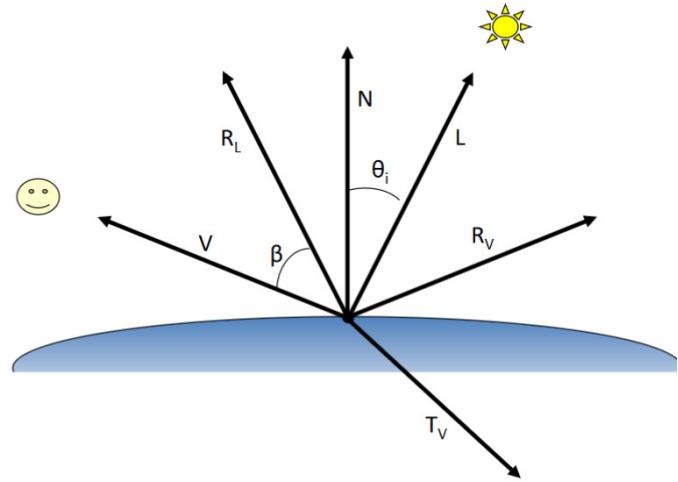
L

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

R<sub>v</sub>

- (d) Què dos vectors determinen la contribució de

Phong? R<sub>L</sub> i V

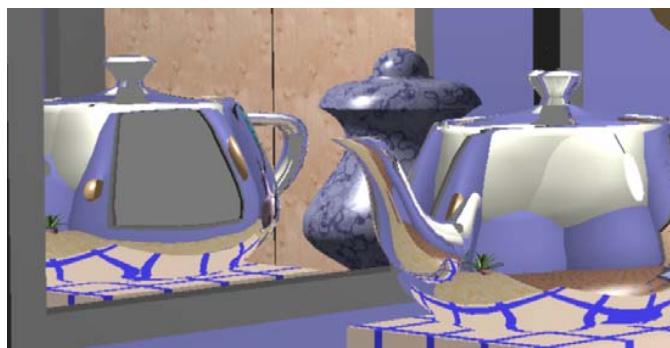


### Exercici 13

Escriu l'equació general del rendering, amb la formulació vista a classe, indicant clarament el tipus de radiància als diferents termes.

### Exercici 14

Considerant la figura:



- (a) Amb quin algorisme s'ha generat? Ray Tracing

- (b) Quin problema té clarament la imatge? Nivell de recursivitat massa baix -> manquen massa interreflexions.

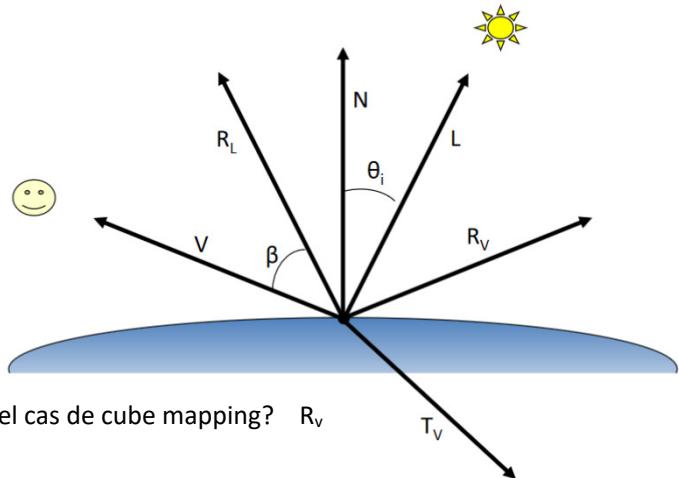
### Exercici 15

Quina unitat de radiometria, mesurada en  $\text{W/m}^2$  o en lux, es defineix com flux per unitat d'àrea?

Irradiància

### Exercici 16

Amb la notació de la figura:

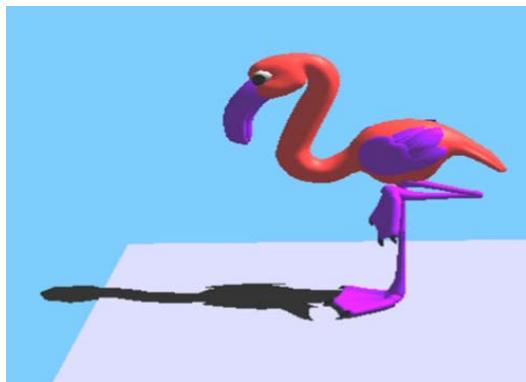


- (a) Quin vector cal usar per indexar un cube map, en el cas de cube mapping?  $R_v$

- (b) Quin dos vectors permeten calcular el terme de Lambert?  $N$  i  $L$

### Exercici 17

Indica com poder evitar aquest problema de la simulació d'ombres amb projecció:



Usant la versió amb stencil buffer per limitar el dibuix de l'ombra a la part ocupada pel receptor.

### Exercici 18

Explica en quines condicions la tècnica de mip mapping produeix una millora substancial de la qualitat de la imatge resultant.

Amb primitives texturades per les que cal un factor important de minification.

### Exercici 19

Què fa aquesta matriu?

$-d$	0	0	0
0	$-d$	0	0
0	0	$-d$	0
$a$	$b$	$c$	0

- (a) Projecta un punt sobre el pla  $(a,b,c,d)$  respecte una llum a l'origen.
  - (b) Projecció respecte una font direccional situada al punt homogeni  $(a,b,c,d)$
  - (c) Reflexió respecte un pla  $(a,b,c,d)$
  - (d) Projecció ortogonal sobre el pla  $(a,b,c,d)$
- (c); n'hi ha prou amb provar amb el pla  $(0,1,0,0)$

(a)

### Exercici 20

Indica quina és la diferència més important entre els models d'il·luminació local i els models d'il·luminació global.

Local → només llum directa

# Encuentra el trabajo de tus sueños

Participa en retos y competiciones de programación



Escanéame y  
obtén más info!!

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):

$M = \text{modelMatrix}$	$M^{-1} = \text{modelMatrixInverse}$
$V = \text{viewingMatrix}$	$V^{-1} = \text{viewingMatrixInverse}$
$P = \text{projectionMatrix}$	$P^{-1} = \text{projectionMatrixInverse}$
$N = \text{normalMatrix}$	$I = \text{Identitat}$

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

b) Pas d'un vèrtex de clip space a world space

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

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

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

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

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

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

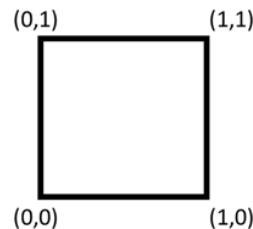
### Exercici 7

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 SI
- (b) LDE SI
- (c) LSE SI
- (d) LDDSE NO

### Exercici 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?

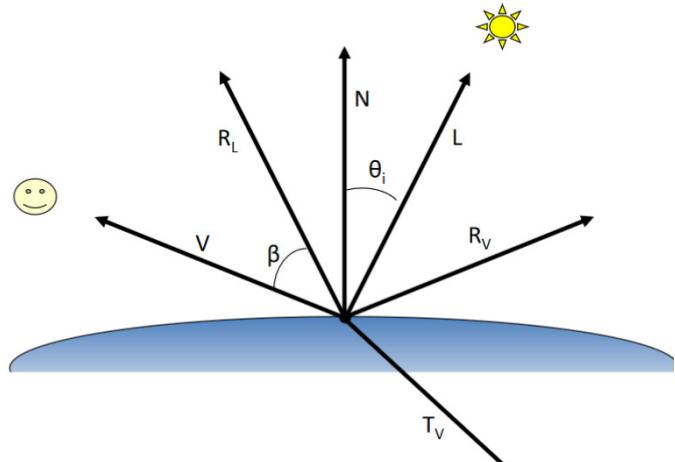
R<sub>v</sub>

(c) Què dos vectors determinen la contribució de

Lambert? L i N

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

T<sub>v</sub>



### 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  $\rho$ ? Una funció que decreix segons la distància

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

### 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).

# Encuentra el trabajo de tus sueños

Participa en retos y competiciones de programación



Escanéame y  
obtén más info!!

## 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_intersecció(raig, escena)
        si info.hi_ha_intersecció() llavors
            color:=calcular_ID(info,escena); // ID
            si es_reflector(info.obj) llavors
                raigR:=calcula_raig_reflectit(info, raig)
                color+= KR*traçar_raig(raigR, escena, μ) //IR
            fsi
            si es_transparent(info.obj) llavors
                raigT:=calcula_raig_transmès(info, raig, μ)
                color+= KT*traçar_raig(raigT, escena, info.μ) //IT
            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;

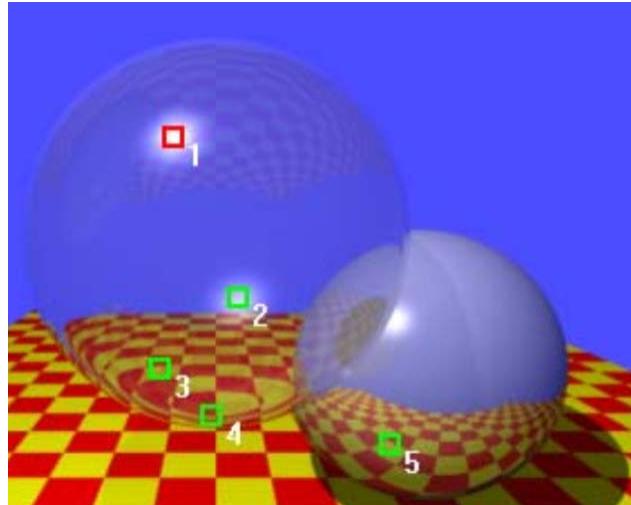
    if (trueDepth <= storedDepth) fragColor = color;
    else fragColor = vec4(0);
}
```



### 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”? LDSE



### 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

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 habitual al pipeline gràfic.

- Clipping
- Divisió de perspectiva
- Fragment shader
- Rasterització

- Clipping
- Divisió de perspectiva -
- Rasterització
- Fragment shader

**Exercici 2**

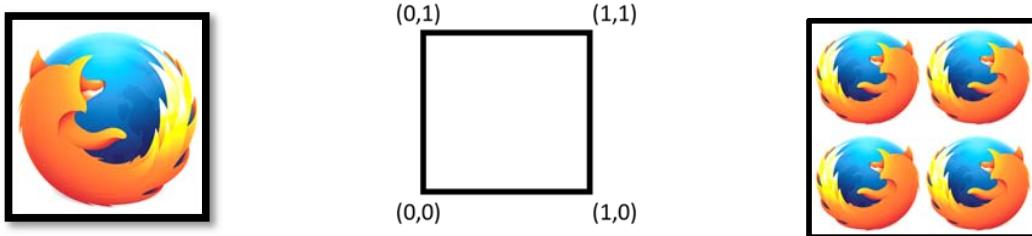
Aquí teniu una llista d'etapes/tasques, ordenades per ordre alfabètic. Torna-les a escriure a la dreta, però ordenades segons l'ordre al pipeline gràfic.

- Depth test
- Fragment Shader
- Rasterització
- Stencil test

- Rasterització
- Fragment Shader
- Stencil test
- Depth test

### Exercici 3

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(-2,2)*texCoord;  
  
    glPosition = vec4(vertex, 1.0);  
}
```

### Exercici 4

Tenim una imatge quadrada de 1Mp (1 mega pixel). Indica les resolucions (WxH) que tindran les textures de la corresponent piràmide de mipmapping.

1024 x 1024 , 512x512, 256x256, ... 1x1

### Exercici 5

Indica quin número de texels cal consultar per cada crida a la funció GLSL texture(), per a cadascú d'aquests modes de filtrat per minification:

- |                               |   |
|-------------------------------|---|
| (a) GL_LINEAR_MIPMAP_LINEAR   | 8 |
| (b) GL_LINEAR                 | 4 |
| (c) GL_NEAREST_MIPMAP_NEAREST | 1 |
| (d) GL_NEAREST                | 1 |

### Exercici 6

Indica, per a cada opció, si és una dada raonable per codificar a cada texel d'una textura per bump mapping o normal mapping (contesta SI/NO).

- |                                      |    |
|--------------------------------------|----|
| (a) Vector normal en tangent space   | SI |
| (b) Gradient del height field        | SI |
| (c) Derivades parcials de P(u,v)     | NO |
| (d) Gradient de la normal perturbada | NO |

# Encuentra el trabajo de tus sueños



Escanéame y obtén más info!!



## Exercicis 7, 8, 9 i 10

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

M = modelMatrix

$M^{-1}$  = modelMatrixInverse

V = viewingMatrix

$V^{-1}$  = viewingMatrixInverse

P = projectionMatrix

$P^{-1}$  = projectionMatrixInverse

N = normalMatrix

I = Identitat

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

b) Pas d'un vèrtex de object space a world space

c) Pas d'un vèrtex de eye space a world space

d) Pas d'un vèrtex de clip space a world space

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

g) Pas d'un vèrtex de world space a eye space

h) Pas de la normal de model space a eye space

i) Pas d'un vèrtex de eye space a clip space

## Exercici 11

Dels shaders estudiats (VS, GS, FS), indica el més adient per implementar les següents tècniques:

- (a) Relief mapping FS
- (b) Parallax mapping FS
- (c) Displacement mapping VS ó GS
- (d) Shadow mapping FS



## Exercici 12

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) LDSSDSSE NO
- (b) LDE SI
- (c) LSDE SI
- (d) LDDSE NO

## Exercicis 13 i 14

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

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

$L$

- (b) Quins dos vectors determinen la contribució de Phong?

$R_L$  i  $V$

- (c) Si el punt pertany a un mirall, quina és la direcció del raig reflectit que cal traçar?

$R_V$

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

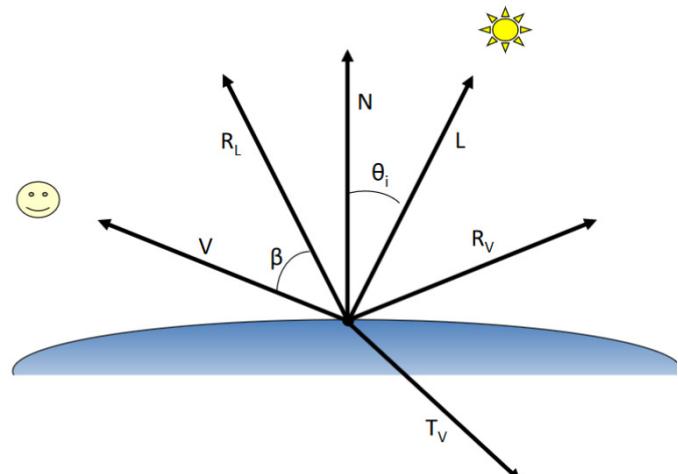
$T_V$

## Exercici 15

Què fa aquesta matriu?

$$\left[ \begin{array}{cccc} -d & 0 & 0 & 0 \\ 0 & -d & 0 & 0 \\ 0 & 0 & -d & 0 \\ a & b & c & 0 \end{array} \right]$$

- (a) Projeció respecte una font direccional situada al punt homogeni  $(a,b,c,d)$
  - (b) Reflexió respecte un pla  $(a,b,c,d)$
  - (c) Projeció ortogonal sobre el pla  $(a,b,c,d)$
  - (d) Projecta un punt sobre el pla  $(a,b,c,d)$  respecte una llum a l'origen.
- c); n'hi ha prou a (d)



## Exercici 16

Si estem generant amb ray-tracing la imatge d'una **escena interior tancada** (exemple habitació sense finestres), indica (directament al codi) què línia/es ens podem estalviar.

```
funció traçar_raig(raig, escena, μ)
    si profunditat_correcta() llavors
        info:=calcula_interseccio(raig, escena)
        si info.hi_ha_interseccio() llavors
            color:=calcular_ID(info,escena); // ID
            si es_reflector(info.obj) llavors
                raigR:=calcula_raig_reflectit(info, raig)
                color+= KR*traçar_raig(raigR, escena, μ) //IR
            fsi
            si es_transparent(info.obj) llavors
                raigT:=calcula_raig_transmès(info, raig, μ)
                color+= KT*traçar_raig(raigT, escena, info.μ) //IT
            fsi
            sino color:=colorDeFons
            fsi
            sino color:=Color(0,0,0); // o colorDeFons
            fsi
            retorna color
ffunció
```

## Exercici 17

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 18**

La tècnica de Shadow Volumes requereix identificar les arestes que pertanyen a la silueta (externa o interna) de l'objecte oclusor. Indica clarament com pots identificar si una aresta pertany a la silueta.

De les dues cares que comparteixen l'aresta, una és frontface i l'altre backface, respecte la llum.

### **Exercici 19**

Per quin càlcul estudiat a classe pot ser útil aplicar el procés d'ortogonalització de Gram-Schmidt?

Càlcul de la base ortonormal de l'espai tangent d'un triangle.

# Encuentra el trabajo de tus sueños

Participa en retos y competiciones de programación



Escanéame y  
obtén más info!!

Examen Final de Gràfics

Curs 2018-19 Q1

Nom:

## Exercici 1

Copia a la dreta aquestes quatre les tasques del pipeline gràfic, però ordenades d'acord amb l'ordre d'execució. Pots assumir VS+FS.

- Perspective Division
- glDrawElements
- Rasterization
- Stencil test

glDrawElements  
Perspective  
Rasterització  
Stencil

## Exercici 2

Copia a la dreta aquestes quatre les tasques del pipeline gràfic, però ordenades d'acord amb l'ordre d'execució. Pots assumir VS+FS.

- Alpha blending
- glBufferData
- Rasterization
- Write to gl\_Position

glBufferData Write  
to gl\_Position  
Raster  
Alpha blending

## Exercici 3

Escriu una condició necessària (per a la llum) per a que lesombres d'una escena tinguin una zona de penombra.

Area light

#### Exercici 4

Escriu, per cada tasca, si bé ABANS o DESPRÉS de la rasterització:

- |                     |        |
|---------------------|--------|
| (a) Pas a NDC       | BEFORE |
| (b) Geometry Shader | BEFORE |
| (c) discard         | AFTER  |
| (d) Depth Test      | AFTER  |

#### Exercici 5

Per a què serveix la funció **glPolygonOffset**? Indica clarament què dades del fragment modifica.

Modify depth values to modify visibility priority of primitives. E.g. give priority to projected object over receiver in *cast shadows*.

#### Exercici 6

Completa aquest codi que correspon al primer pas de l'algorisme d'ombres per projecció:

```
// Draw receiver onto stencil buffer
glEnable(GL_STENCIL_TEST);
glStencilFunc(GL_ALWAYS      , 1, 1);
glStencilOp(GL_KEEP, GL_KEEP,     GL_REPLACE);
draw(receiver);
```

#### Exercici 7

Escriu, usant la notació  $L(D|S)*E$ , els light paths que suporten aquestes tècniques:

- (a) Two-pass raytracing  $LS*DS*E, LS*E$

$LS*DS*E$  (i  $LS*E$ )

- (b) Classic Raytracing  $LDS*E, LS*E$

$LDS*E$  &  $LS*E$

#### Exercici 8

Quin concepte de radiometria/fotometria és el més adient per mesura la quantitat d'energia per unitat de temps que arriba a una superfície, per unitat d'àrea (unitats  $W/m^2$ )?

Irradiància

## Exercici 9

Aquest VS calcula coordenades de textura projectives per a un FS que implementa shadow mapping:

```
uniform mat4 lightMatrix;  
out vec4 textureCoords;  
...  
void main() {  
    ...  
    textureCoords = lightMatrix*vec4(vertex,1);  
    gl_Position = modelViewProjectionMatrix *vec4(vertex,1);  
}
```

Usant aquesta notació:

S(sx,sy,sz) -> Scale matrix

T(tx,ty,tz) -> Translate matrix

M -> model matrix (of the object)

V -> view matrix (of the light camera)

P -> projection matrix (of the light camera)

Escriu (com a producte de matrius) com l'aplicació ha de calcular la matriu pel uniform **lightMatrix**.

T(0.5)S(0.5)PVM

## Exercici 10

Escriu la matriu o producte de matrius per les conversions següents, usant la notació:

$M = \text{modelMatrix}$	$M^{-1} = \text{modelMatrixInverse}$
$V = \text{viewingMatrix}$	$V^{-1} = \text{viewingMatrixInverse}$
$P = \text{projectionMatrix}$	$P^{-1} = \text{projectionMatrixInverse}$
$N = \text{normalMatrix}$	$I = \text{Identitat}$

a) Convertir un vèrtex de world space a clip space

b) Convertir un vèrtex de eye space a clip space

c) Convertir un vèrtex de eye space a world space

d) Convertir una normal de object space a eye space

### Exercici 11

A l'equació general del rendering:

$$L_o(\mathbf{x}, \omega_o, \lambda, t) = L_e(\mathbf{x}, \omega_o, \lambda, t) + \int_{\Omega} f_r(\mathbf{x}, \omega_i, \omega_o, \lambda, t) L_i(\mathbf{x}, \omega_i, \lambda, t) (\omega_i \cdot \mathbf{n}) d\omega_i$$

(a) Què és  $\Omega$ ?

La semiesfera centrada al punt  $\mathbf{x}$  i alineada amb la normal  $\mathbf{n}$  que representa totes les possibles direccions en les que pot arribar llum a  $\mathbf{x}$ .

(b) Quin vector juga el paper del light vector  $L$  que s'utilitza al model d'il·luminació de Lambert?

Omega i

### Exercici 12

Completa aquest FS per calcular Phong lighting:

```
uniform vec4 matAmbient, matDiffuse, matSpecular;
uniform vec4 lightAmbient, lightDiffuse, lightSpecular, lightPosition;
uniform float matShininess;

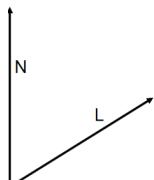
vec4 light(vec3 N, vec3 V, vec3 L)
{
    vec3 R = normalize( 2.0*dot(N,L)*N-L );
    float diff = max( 0.0, dot( N,L ) );
    float RdotV = max( 0.0, dot( R,V ) );
    float Idiff = diff;
    float Ispec = 0;

    if (diff>0) Ispec = pow( RdotV,
matShininess );

    return matAmbient * lightAmbient +
        matDiffuse * lightDiffuse * Idiff +
        matSpecular * lightSpecular * Ispec;
}
```

### Exercici 13

Dibuixa el vector calculat per l'expresió  $2(N \cdot L)N - L$



# Encuentra el trabajo de tus sueños

Participa en retos y competiciones de programación



Escanéame y  
obtén más info!!

## Exercici 14

Dóna un exemple de tècnica que requereixi dibuixar les primitives de l'escena ordenades back to front respecte la càmera.

Alpha blending.

## Exercici 15

En quin sistema de coordinades han d'estar aquestes variables per tal que el VS sigui correcte?

(a)    `vec3 L = normalize(lightPosition.xyz - P); // L is the light vector`

P ha d'estar en espai \_\_\_\_\_ eye space

(b)    `gl_Position = projectionMatrix * P;`

P ha d'estar en espai \_\_\_\_\_ eye space

## Exercici 16

Aquests fragments de codi tenen un o més errors. Re-escriu-los de forma correcta:

(a)    `vec4 P = modelMatrix * viewMatrix * projectionMatrix * vec4(vertex, 1.0);`  
`vec4 P = projectionMatrix * viewMatrix * modelMatrix * vec4(vertex, 1.0);`

(b)    `vec3 N = modelViewMatrix * vec4(normal,0);`  
`normalMatrix`



Nom:

### Exercici 1

Copia a la dreta aquestes quatre les tasques del pipeline gràfic, però ordenades d'acord amb l'ordre d'execució.

- Alpha Test
- glDrawElements
- Rasterization
- Vertex Shader

glDrawElements  
Vertex Shader  
Rasterització  
Alpha Test

### Exercici 2

Copia a la dreta aquestes quatre les tasques del pipeline gràfic, però ordenades d'acord amb l'ordre d'execució.

- Depth test
- Geometry Shader
- Rasterization
- setUniformValue

setUniformValue  
Geometry Shader  
Rasterization  
Depth Test

### Exercici 3

Escriu quin és l'espai de coordenades inicial i final de la multiplicació de la **projection matrix** per un vèrtex.

Inicial: Eye space

Final: Clip space

#### Exercici 4

Quina mena de punt o vector estem transformant amb el producte que apareix a sota?

$$\begin{bmatrix} x_a \\ y_a \\ z_a \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}^{-T} \begin{bmatrix} x_m \\ y_m \\ z_m \end{bmatrix}$$

**Vector normal**

#### Exercici 5

Escriu, per cada tasca, si s'executa ABANS o DESPRÉS del FS:

- (a) Pas a NDC
- (b) Geometry Shader
- (c) Divisió de perspectiva
- (d) Depth Test

#### Exercici 6

Indica, per cada punt, si pot ser dins (DINS) o segur que és fora (FORA) de la piràmide de visió d'una càmera perspectiva:

- (a) (0.2, -0.5, 0.8, 1) en clip space
- (b) (0, 0, 1, 1) en eye space
- (c) (200, 300, 400) en NDC
- (d) (0, 0, 0) en NDC

#### Exercici 7

Completa, en GLSL, una possible definició de la funció mix:

```
vec3 mix(vec3 a, vec3 b, float t)
{
    return ....
}
```

#### Exercici 8

Siguin R, S i T les matrius de rotació, escalat i translació a aplicar a un model com a part de la transformació de modelat. Indica en quin ordre és més habitual multiplicar aquestes matrius (escriu el producte de les matrius):

### Exercici 9

Escriu, usant la notació  $L(D|S)^*E$ , els light paths que suporten aquestes tècniques:

(a) Two-pass raytracing

(b) Classic Raytracing

### Exercici 10



Volem aplicar la textura a un quad que té coordenades de textura inicials en  $[0,1]$ .

Completa el FS per aconseguir els resultats que es mostren:



(a)

frontColor = texture(colorMap, \_\_\_\_\_ \* vtexCoord);

vec2(0, 0.5)+ vec2(0.5)\*



(b)

frontColor = texture(colorMap, \_\_\_\_\_ \* vtexCoord);

vec2(1,0.5)\*

### Exercici 11

Quin concepte de radiometria/fotometria és el més adient per mesurar la quantitat d'energia per unitat de temps que arriba a una superfície, per unitat d'àrea (unitats  $\text{W/m}^2$ )?

### Exercici 12

A l'equació general del rendering:

$$L_o(\mathbf{x}, \omega_o, \lambda, t) = L_e(\mathbf{x}, \omega_o, \lambda, t) + \int_{\Omega} f_r(\mathbf{x}, \omega_i, \omega_o, \lambda, t) L_i(\mathbf{x}, \omega_i, \lambda, t) (\omega_i \cdot \mathbf{n}) d\omega_i$$

Què representa  $\omega_o$ ?

Vector unitari en la direcció de sortida.

# Encuentra el trabajo de tus sueños

Escanéame y  
obtén más info!!



Participa en retos y competiciones de programación



## Exercici 13

Aquest VS calcula coordenades de textura per a un FS que implementa shadow mapping:

```
uniform mat4 lightMatrix;  
out vec4 textureCoords;  
const float a = .....;  
...  
void main() {  
    ...  
    textureCoords = T(a,a,a)*S(a,a,a)*lightMatrix*vec4(vertex,1);  
    gl_Position = modelViewProjectionMatrix *vec4(vertex,1);  
}
```

Usant aquesta notació:

$S(sx,sy,sz)$  -> Scale matrix  
 $M$  -> model matrix (of the object)  
 $P$  -> projection matrix (of the light camera)

$T(tx,ty,tz)$  -> Translate matrix  
 $V$  -> view matrix (of the light camera)

- Quin valor ha de tenir la constant a?
- Escriu (com a producte de matrius) com l'aplicació ha de calcular, en aquest cas, la matriu **lightMatrix**.

## Exercici 14

Escriu la matriu o producte de matrius per les conversions següents, usant la notació:

$M$ = modelMatrix	$M^{-1}$ = modelMatrixInverse
$V$ = viewingMatrix	$V^{-1}$ = viewingMatrixInverse
$P$ = projectionMatrix	$P^{-1}$ = projectionMatrixInverse
$N$ = normalMatrix	I = Identitat

- Convertir un vèrtex de world space a clip space
- Convertir un vèrtex de eye space a world space

## Exercici 15

Sigui  $P(u,v)$  la representació paramètrica d'una superfície, amb normals unitàries  $N(u,v)$ . Sigui  $F(u,v)$  un mapa d'elevacions. Escriu l'expressió que permet calcular la superfície  $P'(u,v)$  que resulta de pertorbar  $P$  segons el mapa d'elevacions, tal i com es faria servir en *displacement mapping*.



## Exercici 16

Escriu en codi GLSL com calcular la posició de la càmera en *object space*. Pot usar les matrius per defecte del viewer.

vec3 obs =

## Exercici 17

Un VS ha calculat unes coordenades de textura vtexCoord usant la tècnica de **projective texture mapping**.

(a) Indica quina ha de ser la declaració (en GLSL) de vtexCoord, al VS

(b) Completeu com caldria accedir a la textura *colormap* en aquest cas:

vec4 color = texture(colormap, .....);

## Exercici 18

En la tècnica de generació d'ombres per projecció,

(a) Quants cop cal dibuixar l'objecte que produeix l'ombra?

(b) Per què ens pot ser útil usar el stencil buffer?

## Exercici 19

En quin sistema de coordinades han d'estar aquestes variables per tal que el VS sigui correcte?

(a) vec3 L = normalize(lightPosition.xyz – modelViewMatrix \* P); // L is the light vector

P ha d'estar en espai \_\_\_\_\_

(b) gl\_Position = projectionMatrix \* P;

P ha d'estar en espai \_\_\_\_\_

## Exercici 20

Si fem servir el mode de filtrat GL\_LINEAR\_MIPMAP\_LINEAR per MINIFICATION, quants texels es fan servir per avaluar cada crida a texture()?

Nom:

### Exercici 1

Copia a la dreta aquestes quatre les tasques del *pipeline* gràfic, però ordenades d'acord amb l'ordre d'execució.

- Alpha Blending
- Fragment shader
- Geometry shader
- Rasterització

### Exercici 2

Copia a la dreta aquestes quatre les tasques del *pipeline* gràfic, però ordenades d'acord amb l'ordre d'execució.

- Depth test
- glDrawElements
- Stencil Test
- Vertex Shader

### Exercici 3

Escriu quin és l'espai de coordenades inicial i final de la multiplicació de la **modelViewProjectionMatrixInverse** per un vèrtex.

Inicial:

Final:

#### Exercici 4

Escriu, per cada tasca, en quin o quins shaders (VS, GS, FS) és possible:

- (a) discard
- (b) EndPrimitive()
- (c) Escriure gl\_Position
- (d) dFdx, dFxy

#### Exercici 5

Què coordenada del fragment modifica la funció `glPolygonOffset`?

En quin espai la modifica?

#### Exercici 6

Indica, amb la notació vista a classe, el *light path* que explica el color dominant del píxel indicat a la imatge.



#### Exercici 7

Indica, per cada punt, si pot ser dins (DINS) o segur que no (FORA) la piràmide de visió d'una càmera perspectiva. Escriu una breu explicació.

- (a) (0.2, -1.5, 1.8, 2) en clip space
- (b) (0, 0, 10, 1) en object space

# Encuentra el trabajo de tus sueños

Participa en retos y competiciones de programación



Escanéame y  
obtén más info!!

## Exercici 8

Re-escriu el codi GLSL subratllat amb una versió equivalent però més compacte:

```
float t;  
vec3 color1, color2;  
vec3 color = t*color1 + (1-t)*color2;
```

## Exercici 9

Re-escriu aquest codi GLSL amb una versió equivalent més compacte:

```
vec3 obs = (modelViewMatrixInverse * vec4(0,0,0,1)).xyz;
```

## Exercici 10

Escriu un exemple d'algorisme que suporti els *light paths* que s'indiquen:

(a) LS\*DS\*E (i LS\*E)

(b) L(D|S)\*E

## Exercici 11



Volem aplicar la textura a un quad que té coordenades de textura inicials en [0,1].



Completa el FS per aconseguir el resultat que es mostra:

```
frontColor = texture(colorMap, _____ * vtexCoord);
```



## Exercici 12

Aquest VS calcula coordenades de textura projectives per a un FS que implementa *shadow mapping*:

```
uniform mat4 lightMatrix, modelMatrix;  
out vec4 textureCoords;  
...  
void main() {  
    ...  
    textureCoords = lightMatrix*modelMatrix*vec4(vertex,1);  
    gl_Position = modelViewProjectionMatrix *vec4(vertex,1);  
}
```

Usant aquesta notació:

S(sx,sy,sz) -> Scale matrix

T(tx,ty,tz) -> Translate matrix

M -> model matrix (of the object)

V -> view matrix (of the light camera)

P -> projection matrix (of the light camera)

Escriu (com a producte de matrius) com l'aplicació ha de calcular la matriu pel uniform **lightMatrix**.

## Exercici 13

A l'equació general del rendering:

$$L_o(\mathbf{x}, \omega_o, \lambda, t) = L_e(\mathbf{x}, \omega_o, \lambda, t) + \int_{\Omega} f_r(\mathbf{x}, \omega_i, \omega_o, \lambda, t) L_i(\mathbf{x}, \omega_i, \lambda, t) (\omega_i \cdot \mathbf{n}) d\omega_i$$

Què representa  $\Omega$ ?

## Exercici 14

Escriu la matriu o producte de matrius per convertir un vèrtex de *object space* a *eye space*, usant únicament les matrius que s'indiquen (no en falta cap per aquest exercici):

modelMatrix	modelMatrixInverse
projectionMatrix	projectionMatrixInverse
modelViewProjectionMatrix	modelViewProjectionMatrixInverse

### Exercici 15

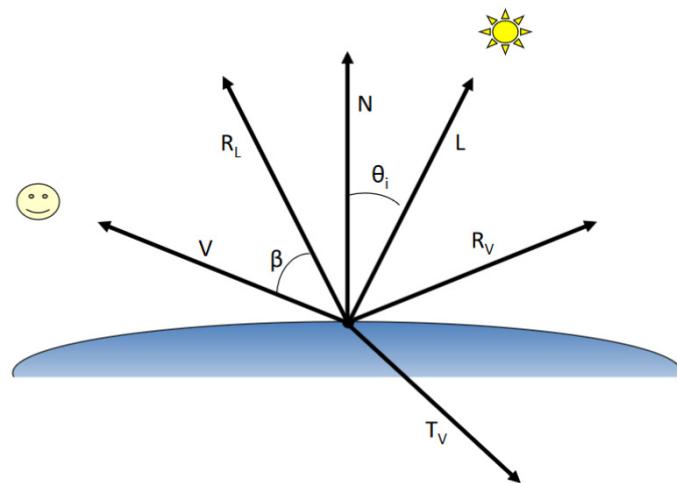
Indica quin tipus GLSL (float, vec2, vec3...) usaries per cada cas:

- (a) Segon paràmetre d'una crida a texture (colormap...)
- (b) Coordenades de textura que passa VS a FS en *shadow mapping*
- (c) Coordenades de textura que passa VS a FS en *projective texture mapping*
- (d) Vector per accedir a un *sphere map*

### Exercici 16

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

- (a) Quin vector té la direcció del *shadow ray*?
- (b) Quin vector és paral·lel al raig transmès?



### Exercici 17

Continuant amb la figura anterior...

- (a) Quin vector té la direcció del raig primari?
- (b) Quins dos vectors determinen la contribució local de Phong?

### Exercici 18

Quines són les unitats de la radiància (radiometria) en el Sistema Internacional?

## Exercici 19

Completa, a sota, el codi que falta.

```
funció traçar_raig(raig, escena, μ)
    si profunditat_correcta() llavors
        info:=calcula_interseccio(raig, escena)
        si info.hi_ha_interseccio() llavors
            color:=calcular_ID(info,escena); // ID
            si es_reflector(info.obj) llavors
                raigR:=calcula_raig_reflectit(info, raig)
                color+= KR*traçar_raig(raigR, escena, μ) //IR
            fsi
            si es_transparent(info.obj) llavors
                [REDACTED]
            fsi
        sino color:=colorDeFons
        fsi
    sino color:=Color(0,0,0); // o colorDeFons
    fsi
    retorna color
ffunció
```

## Exercici 20

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

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

void main()
{
    [REDACTED]

    float NdotL = max(0.0, dot(N,L));
    vec4 color = vec4(NdotL);
    vec2 st = vtexCoord.st / vtexCoord.q;

    float storedDepth =
    [REDACTED]

    float trueDepth = vtexCoord.p / vtexCoord.q;
    if (trueDepth <= storedDepth) fragColor = color;
    else fragColor = vec4(0);
}
```

# Encuentra el trabajo de tus sueños



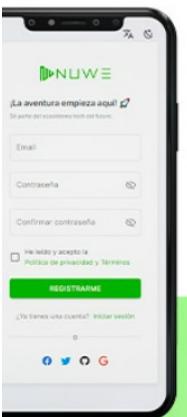
Escanéame y obtén más info!!



Tria l'espai de coordenades en què ha de portar P per tal que la transformació projectiva MatrixI verse\* tingui sentit.

Select one:

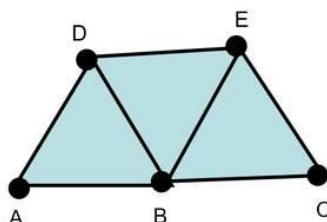
- clip space
- eye space
- object space
- world space



Assigna a cada crida/tasca l'ordre relatiu (1,2,3,4) en que s'executa en un pipeline d'OpenGL sense GS:

glGenVertexArrays	1
S'escriu gl_Position	2
Backface culling	3
Rasterització	4

Indica un ordre adient per emetre els vèrtexs dels triangles de la figura, en un GS, usant una única primitiva:



Select one:

- ADBEC
- CEBAD
- DABEC
- ABDEC

Indica el punt que segur que serà FORA de la piràmide de visió d'una càmera perspectiva:

Select one:

- (1.00, 5.00, 6.00, 2.00) en clip space
- (3.00, 4.00, 1.00, 11.00) en clip space
- (2.00, 0.00, -5.00, 1.00) en eye space
- (-2.00, -4.00, -8.00, 1.00) en eye space

Indica el valor que retorna aquesta expressió GLSL:

```
mix(9, 3, 0.7)
```

Answer:

Correct answers: [4.8]

Indica el valor que retorna aquesta expressió GLSL:

```
mod(7.9, 4)
```



Answer:

Correct answers: [3.9000000000000004]

Tenim una primitiva que ocupa tot un viewport de 2048x2048 pixels. Indica quin codi ens dona un cercle blanc de radi 295 pixels centrat al viewport:

Select one:

- fragColor = vec4(1-step(295, distance(gl\_FragCoord.xy, vec2(1024))))
- fragColor = vec4(step(1024, distance(gl\_FragCoord.xy, vec2(295))))
- fragColor = vec4(step(295, distance(gl\_FragCoord.xy, vec2(2048))))
- fragColor = vec4(step(295, distance(gl\_FragCoord.xy, vec2(1024))))

Soposa que  
P és un punt,  
N és la normal unitària en el punt,  
(a,b,c,d) és el pla perpendicular a N que conté P,  
L és un vector unitari cap a la font de llum,  
R és el vector reflectit de L,  
V és un vector unitari en direcció cap a la càmera, i  
Q és un punt arbitrari.

Quina interpretació té l'expressió **cross(dFdx(P), dFdy(P))**?

Select one:

- Vector normal
- Projecció del vector posició sobre una esfera unitària
- Vector de reflexió especular de la llum directa
- Vector tangent a la superfície en el punt

Indica, en un FS, quina transformació de la z en window space té un efecte equivalent a invertir el depth test amb glDepthFunc(GL\_GREATER):

Select one:

- gl\_FragDepth = 1 - gl\_FragCoord.z;
- gl\_FragDepth = -1 \* gl\_FragCoord.z;
- gl\_FragDepth = 0.5 \* gl\_FragCoord.z;
- gl\_FragCoord.z = 1 - gl\_FragCoord.z;

Diposem d'aquesta textura:



Indica amb quina opció el FS de sota obté aquest resultat amb l'objecte plane:



Recorda que plane.obj té coordenades de textura en [0,1].

```
fragColor = texture(colorMap, factor*vtxCoord + offset)
```

Select one:

- factor=vec2(0.1, 1.0); offset=vec2(0.1, 0.0);
- factor=vec2(1.0, 1.0); offset=vec2(0.1, 0.1);
- factor=vec2(0.1, 1.0); offset=vec2(0.0, 1.0);
- factor=vec2(0.1, 0.1); offset=vec2(0.1, 1.0);

La matriu que representa una reflexió respecte un mirall triangular definit pels vèrtexs (6.00, 0.00, 9.00), (9.00, 0.00, 1.00), (7.00, 0.00, 1.00) és...

Select one:

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

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

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

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

Assigna a cada tasca l'ordre relatiu (1,2,3,4) en què s'executa, per simular reflexions especulars utilitzant la tècnica de sphere mapping en eye space:

El VS passa P, N a eye space

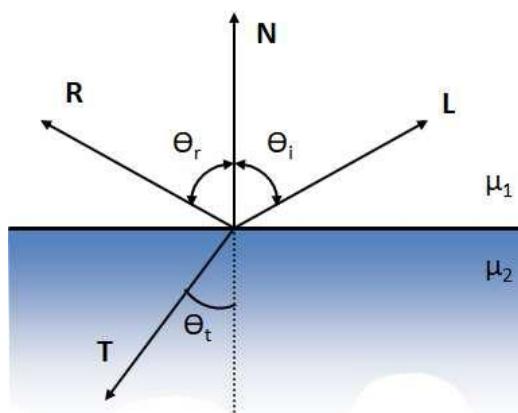
- 1
- 2
- 3
- 4

Es calcula el vector reflectit

Es calculen les coordenades (s,t) del fragment

Es pren una mostra de la textura que conté el sphere map

Tenint en compte la llei de Snell, indica quina parella de valors ( $\mu_1, \mu_2$ ) explicarien de forma aproximada la direcció del raig trasmès que s'observa a la figura:



Select one:

(1.09, 1.43)

(1.43, 1.09)

(0.09, 1.43)

(1.09, 0.43)

Tenim activat alpha blending amb la funció

```
glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
```

El color RGBA d'un fragment és (0.30, 0.20, 0.50, 0.10), i el color associat al frame buffer és (0.70, 0.50, 0.80, 0.60). Indica quin

Nom i Cognoms:

---

**Exercici 1 (1 punt)**

Completa el següent codi GLSL (vigila la sintaxi!):

- (a) Funció que donat un vèrtex (en *eye space*) i una llum posicional, retorna el vector unitari L (també en *eye space*) que cal per calcular la il·luminació:

```
vec3 lightVector(vec3 vertex, gl_LightSourceParameters light)
{
    [REDACTED]
}
```

- (b) Il·luminació bàsica a nivell de fragment:

```
// Vertex shader
varying vec3 normal;

void main() {
    normal =
    gl_Position =
    gl_FrontColor = gl_Color;
}

// Fragment shader
varying vec3 normal;

void main() {
    gl_FragColor = normal.z * gl_Color;
}
```

- (c) Completa aquest shader per tal que calculi el color del fragment com el color del texel que indiquen les seves coordenades de textura (texture unit 0):

```
uniform sampler2D sampler;

void main(void) {
    gl_FragColor =
}
```

# Encuentra el trabajo de tus sueños

Participa en retos y competiciones de programación



Escanéame y  
obtén más info!!

## Exercici 2 (1 punt)

Completa el geometry shader d'aquest programa perquè, a més a més de l'objecte 3D, dibuixi la seva reflexió respecte el pla horizontal Y=0 (veure figura).

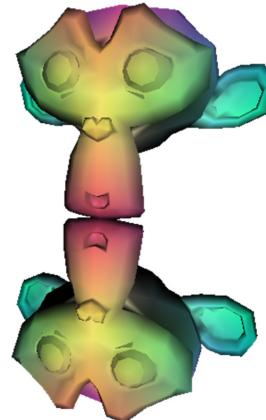


Figura 1. Resultat amb els shaders que us proporcionem (esquerra) y resultat desitjat (dreta).

**Vertex shader** (no cal modificar-lo):

```
void main()
{
    gl_Position    = gl_Vertex;
    gl_FrontColor  = gl_Color;
}
```

**Geometry shader** (a completar):

```
void main( void )
{
    for( int i = 0 ; i < 3 ; i++ )
    {
        gl_FrontColor = gl_FrontColorIn[ i ];
        gl_Position   = gl_ModelViewProjectionMatrix * gl_PositionIn[i];
        EmitVertex();
    }
    EndPrimitive();
}

EndPrimitive();
```

### Exercici 3 (1 punt)

La següent figura mostra un quadrat texturat que ocupa tot un viewport de 1024x1024 pixels. La textura utilitzada és similar a la de la figura. Indica el valor de les següents derivades parcials, on s i t són directament les coordenades de textura interpolades al fragment:

$$(a) \frac{\partial s}{\partial x} =$$

$$(b) \frac{\partial t}{\partial y} =$$



### Exercici 4 (1 punt)

Tenim una aplicació que implementa oclusió ambient i obscuràncies. L'aplicació no fa servir cap estructura de dades per accelerar els càlculs d'intersecció raig-escena.

- (a) Què creus que s'executarà més ràpid a l'aplicació, el càlcul de l'occlusió ambient o el càlcul de les obscuràncies?
  
- (b) Per què?

### Exercici 5 (1 punt)

- (a) Quan es pot produir el fenomen de reflexió total?
  
- (b) 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?
  
- (c) Explica què és l'angle crític
  
- (d) Indica quin valor de l'angle d'incidència maximitza la proporció de llum transmesa quan aquesta passa de l'aire al vidre.

$$\Theta_i =$$

### Exercici 6 (1 punt)

Aquest fragment de codi per generar reflexions és incomplet:

#### // 1. Dibuixem el mirall a l'stencil buffer

```
glEnable(GL_STENCIL_TEST);
glStencilFunc(GL_ALWAYS, 1, 1);
glStencilOp(GL_KEEP, GL_KEEP, GL_REPLACE);
glDepthMask(GL_FALSE); glColorMask(GL_FALSE...);
dibuixar(mirall);
```

#### // 2. Dibuixem els objectes en posició virtual

```
glDepthMask(GL_TRUE); glColorMask(GL_TRUE...);
glStencilFunc(GL_EQUAL, 1, 1);
glStencilOp(GL_KEEP, GL_KEEP, GL_KEEP);
glPushMatrix();
glLightfv(GL_LIGHT0, GL_POSITION, pos);
glCullFace(GL_FRONT);
dibuixar(escena);
glPopMatrix();
```

#### // 3. Dibuixem el mirall semitransparent

```
glDisable(GL_STENCIL_TEST);
glLightfv(GL_LIGHT0, GL_POSITION, pos);
glCullFace(GL_BACK);
dibuixar(mirall);
```

#### // 4. Dibuixem els objectes reals

```
dibuixar(escena);
```

- Quina crida OpenGL (relacionada amb les matrius de transformació geomètrica) manca?
- On caldria afegir-la (ho pots indicar al codi amb una fletxa)

### Exercici 7 (1 punt)

Un company vostre ha fet servir aquest codi per visualitzar l'escena amb VBO.

```
void Object::render() {  
  
    ...  
    glBindBuffer(GL_ARRAY_BUFFER, verticesID);  
    glBufferData(GL_ARRAY_BUFFER, verts);  
    glBindBuffer(GL_ELEMENT_ARRAY_BUFFER, indicesID);  
    glBufferData(GL_ELEMENT_ARRAY_BUFFER, indices);  
  
    glBindBuffer(GL_ARRAY_BUFFER, id);  
    glVertexPointer(3, GL_FLOAT, 0, (GLvoid*) 0);  
    glEnableClientState(GL_VERTEX_ARRAY);  
  
    glBindBuffer(GL_ELEMENT_ARRAY_BUFFER, id);  
    glDrawElements(GL_TRIANGLES, n, GL_UNSIGNED_INT, (Glvoid *) 0 );  
  
    glDisableClientState(GL_VERTEX_ARRAY);  
    glBindBuffer(GL_ARRAY_BUFFER, 0);  
    glBindBuffer(GL_ELEMENT_ARRAY_BUFFER, 0);  
  
}
```

Tot i que li funciona, no ha aconseguit millorar gens el rendiment respecte els VAs. Què creieu que ha fet malament?

### Exercici 8 (1 punt)

En quins d'aquests casos us poden ser útils les funcions `dFdx()` i `dFdy()` de GLSL? Indica-ho amb UTIL / NO UTIL.

- (a) Per emular de forma aproximada el funcionament de `glPolygonOffset`.
- (b) Per calcular manualment, al fragment shader, quin nivell de detall (LoD) es necessita quan es fa servir mipmapping.
- (c) Per calcular la component especular de la il·luminació.
- (d) Per calcular qualsevol derivada parcial en un vertex shader

# Encuentra el trabajo de tus sueños

Participa en retos y competiciones de programación



Escanéame y  
obtén más info!!

## Exercici 9 (1 punt)

Completa aquest codi que implementa ombres per projecció amb stencil buffer:

```
// 1. Dibuixa el receptor al color buffer i al stencil buffer
glEnable(GL_STENCIL_TEST);
glStencilFunc(          , 1, 1);
glStencilOp(GL_KEEP, GL_KEEP,           );
dibuixa(receptor);

// 2. Dibuixa oclusor per netejar stencil a les zones a l'ombra
glDisable(GL_DEPTH_TEST);
glColorMask(GL_FALSE, ... GL_FALSE);
glStencilFunc(          , 1);
glStencilOp(GL_KEEP, GL_KEEP, GL_ZERO);
glPushMatrix(); glMultMatrixf(MatriuProjeccio);
dibuixa(oclusor);
glPopMatrix();
// 3. Dibuixa la part fosca del receptor
glEnable(GL_DEPTH_TEST);
glDepthFunc(GL_LEQUAL);
glColorMask(GL_TRUE, ... , GL_TRUE);
glDisable(GL_LIGHTING);
glStencilFunc(          , 1);
Dibuixa(receptor);
// 4. Dibuixa l'oclusor
glEnable(GL_LIGHTING);
glDepthFunc(GL_LESS);
glDisable(GL_STENCIL_TEST);
Dibuixa(oclusor);
```

## Exercici 10 (1 punt)

Indica, per cadascun d'aquests mètodes de filtrat, **quants nivells de mipmap** s'han de consultar per calcular el color de la mostra.

- (a) GL\_NEAREST
- (b) GL\_LINEAR
- (c) GL\_NEAREST\_MIPMAP\_NEAREST
- (d) GL\_LINEAR\_MIPMAP\_LINEAR

# Preguntes per a l'avaluació de les competències transversals

## Pregunta 1

L'article *Principles and Applications of Computer Graphics in Medicine* menciona els tres aspectes bàsics que ha de tenir una aplicació mèdica en general, i les aplicacions per l'entrenament quirúrgic (surgery simulation) en particular. Quins són aquests tres factors?

- (a) Realista, assequible, validada
- (b) Realista, assequible, simulació acurada de la interacció de la llum amb l'escena
- (c) Realista, assequible, framerate superior a 60 fps
- (d) Realista, assequible, hàptica

## Pregunta 2

A què fa referència el terme *haptic*?

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

## Pregunta 3

La visualització de dades de volum en aplicacions mèdiques requereix un cert procés de preparació i pre-procés de les dades. Quina és la etapa d'aquest preprocés que segueix típicament al filtrat?

- (a) Compressió
- (b) Segmentació
- (c) Texturació
- (d) Estructuració

## Pregunta 4

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

---

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. Tornades 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)

**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 **x** al costat de les que no calen). Hi ha un parell de línies que ja us hem marcat com necessàries.

// 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 ocupa 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);
```



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

# Encuentra el trabajo de tus sueños

Participa en retos y competiciones de programación



Escanéame y  
obtén más info!!

## 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.

## 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);
glStencilOp(GL_KEEP, GL_KEEP, _____);
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  $4 \times 4$  cal multiplicar la posició del vertex en world space per tal d'obtenir les ombres projectades al dibuixar els objectes.



### 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;
```

```
    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();
```

### **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?

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

### **Exercici 11**

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

### **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?

Indica una unitat de mesura per la magnitud anterior

### **Exercici 13**

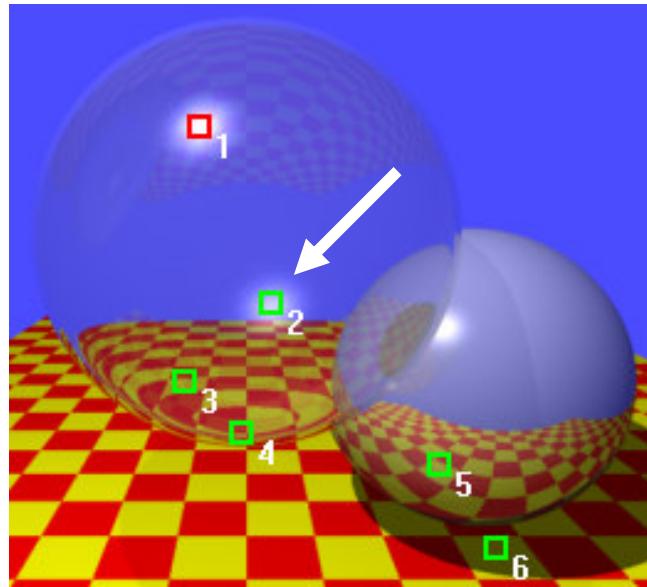
Quines equacions permeten calcular la proporció de llum reflectida i la proporció de llum transmessa, quan la llum incideix en la superfície de separació entre dos medis?

### **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).

### Exercici 15

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



La taca espeacular 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.

### 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.

# Encuentra el trabajo de tus sueños

Participa en retos y competiciones de programación



Escanéame y  
obtén más info!!

Examen Final de Gràfics

Curs 2012-13 Q2

Nom i Cognoms:

**Exercici 1 (1 punt)** Omple la següent taula comparant dues característiques diferents dels dos paradigmes principals de visualització discutits a classe.

Característica o aspecte	Paradigma projectiu	Traçat de raigs
1.		
2.		

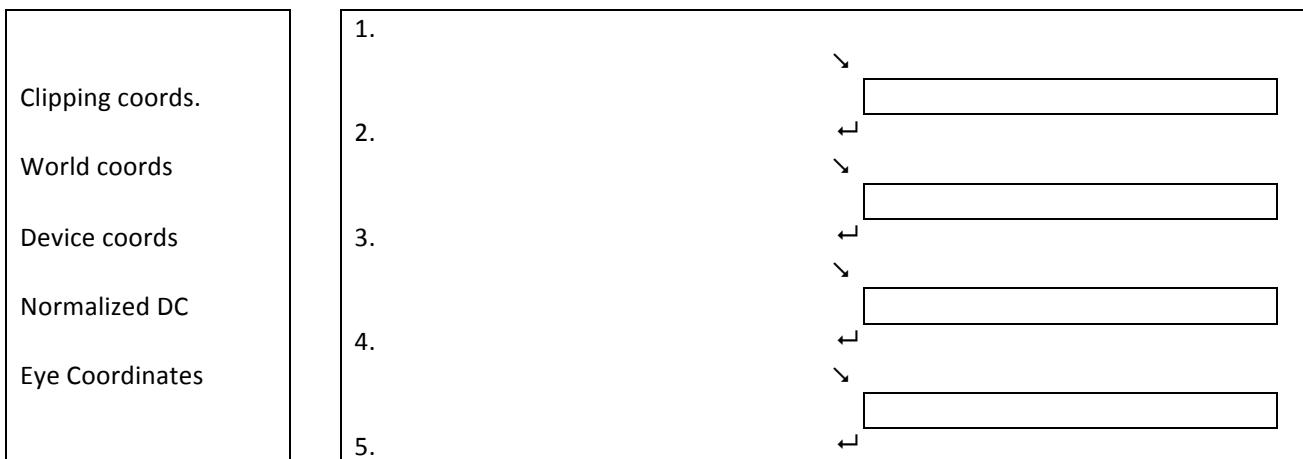
**Exercici 2 (1 punt)** Quina diferència hi ha entre *fotometria* i *radiometria*?

**Exercici 3 (1 punt)** En relació a l'acceleració del Ray Tracing clàssic, indica per a cadascuna de les següents afirmacions, si és certa o falsa:

- a) Amb un arbre BSP sempre trobem la primera intersecció de cada raig en temps logarítmic
- b) El BSP és un mètode de subdivisió dels objectes
- c) Els Octtrees són un mètode de subdivisió de l'espai
- d) Les caixes d'un arbre d'AABB són disjunes i cada objecte pertany a sols una fulla
- e) El cost de calcular la primera intersecció d'un raig auxiliant-nos d'una voxelització d' $N \cdot N \cdot N$  és  $O(N)$  independentment de la geometria de l'escena.



**Exercici 4 (1 punt)** A la llista de l'esquerra apareixen diferents sistemes de coordenades. Col·loca'ls a la caixa de la dreta ordenats de la manera que apareixen al pipeline d'OpenGL, indicant a les caixes de la dreta la matriu o operació de transformació que cal fer per passar d'un sistema al següent:



**Exercici 5 (1 punt)** Volem pintar un objecte brillant usant OpenGL, i que s'hi vegin reflectits els elements de l'entorn. Compara les tècniques de Sphere Mapping i Cube Mapping a tal efecte.

**Exercici 6 (1 punt)** Indica un avantatge i una limitació de l'algorisme de Shadow Mapping respecte del d'ombres per projecció.

**Exercici 7 (1 punt)** Per a cadascun dels següents *paths* indica si es pot simular o no en l'algorisme clàssic de Ray Tracing, i perquè:

- a) LDSDE
- b) LSSDE
- c) LDSSE
- d) LSDSE

**Exercici 8 (1 punt)** Com afecta la introducció d'un Geometry Shader al pipeline al pas d'informació entre el VS i el FS? Explica per què succeeix i és raonable que sigui així.

**Exercici 9 (1 punt)** Tenim un polígon quadrat amb vèrtexs  $(-1, 0, -1)$ ,  $(1, 0, -1)$ ,  $(1, 0, 1)$  i  $(-1, 0, 1)$ . Les coordenades de textura  $(s, t)$  assignades a aquests vèrtexs són, respectivament,  $(0, 0)$ ,  $(2, 0)$ ,  $(2, 2)$  i  $(0, 2)$ . La textura que volem projectar correspon a la de la figura:

**AB  
CD**

Contesteu, **raonant** les vostres respostes:

- Quantes lletres hi haurà a la projecció del polígon si s'utilitza el mode de wrapping GL\_REPEAT?
- Indica a cada figura com hauríem d'assignar les coordenades de textura del quadrat, i quin seria el mode de wrapping adient, per a obtenir cadascuna d'aquestes imatges:

**C**  
**ABAB**  
**CDCD**  
**ABAB**  
**CDCD**

**Exercici 10 (1 punt)** Quan fem servir la tècnica d'objectes virtuals per a simular reflexions, existeix el problema que aquests poden aparèixer directament a l'escena i no com a reflexions al mirall dels objectes reals. Expliqueu dues maneres diferents d'evitar-ho en OpenGL.

## Pregutes per a l'avaluació de les competències transversals

**Pregunta 1:** En relació a l'article sobre aplicacions dels gràfics en medicina, contesta SI/NO a les següents preguntes:

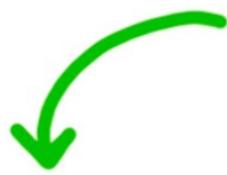
- a) Els actuals scanners 3D proporcionen dades amb un nivell de soroll negligible
- b) La principal utilitat del procés de segmentació de dades mèdiques és assignar a cada voxel una etiqueta que correspon al tipus de teixit/part de l'anatomia.
- c) L'algorisme Marching Cubes permet extreure superfícies a partir de dades MRI o CT.
- d) Les aplicacions mèdiques actuals només consideren òrgans rígids

**Pregunta 2:** Indica en què consisteix una endoscòpia virtual i quins avantatges i limitacions presenta respecte l'endoscòpia tradicional.

# Encuentra el trabajo de tus sueños



Escanéame y obtén más info!!



Examen Final de Gràfics

Curs 2013-14 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:

- Fragment Shader (FS)
- Geometry Shader (GS)
- Rasterització
- Stencil test
- Vertex Shader (VS)

## Exercici 2

Les variables varying són de sortida en un VS i d'entrada en un FS. En quina de les següents etapes **s'interpolen** els valors de les variables varying?

- (a) Fragment Shader (FS)
- (b) Rasterization
- (c) Primitive assembly
- (d) Viewport transformation

## Exercici 3

Volem aplicar un escalat uniforme 2x als vèrtexs d'un objecte. L'aplicació concreta fa servir coordenades homogènies, i ens diuen que no podem assumir que la coordenada w serà sempre 1. Indica, per cada opció, si és una forma correcta o no d'aplicar l'escalat anterior a gl\_Vertex:

- (a) `vec4 v = gl_Vertex * 2.0;`
- (b) `vec4 v = vec4(gl_Vertex.xyz * 2.0, gl_Vertex.w);`
- (c) `vec4 v = gl_Vertex.xyz * 2.0;`
- (d) `vec4 v = gl_Vertex * vec4(2.0, 2.0, 2.0, 1.0);`



## Exercici 4

Aquest fragment de codi calcula el vector V que es necessita pels càlculs d'il·luminació:

```
vec3 V = normalize( (gl_ModelViewMatrix * (Obs - Vert)).xyz );
```

(a) En quin espai estan Obs (posició de l'observador) i Vert (posició del vèrtex)?

(b) En quin espai està V?

## Exercici 5

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

```
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; // cal que el calculeu vosaltres  
  
    diff =  
  
    float spec; 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 6

Es poden modificar les coordenades (x,y) d'un fragment (gl\_FragCoord.xy) en un FS?

## Exercici 7

Si un punt és interior al frustum de visió, les seves coordenades en NDC estaran

- (a) Entre 0 i 1, amb z=0 pels punts sobre el pla de retallat anterior
- (b) Entre 0 i 1, amb z=1 pels punts sobre el pla de retallat anterior
- (c) Entre -1 i 1, amb z=-1 pels punts sobre el pla de retallat anterior
- (d) Entre -1 i 1, amb z=1 pels punts sobre el pla de retallat anterior

### **Exercici 8**

Tenim una textura de 256x256. Quina és l'amplada d'un texel en espai normalitzat de textura?

- (a)  $\Delta s = 256$
- (b)  $\Delta s = 1/256$
- (c)  $\Delta s = 1/(2*256)$
- (d)  $\Delta s = \log(256)$

### **Exercici 9**

Respete a la diferència entre calcular la il·luminació per vèrtex (al VS) o fer-ho per fragment (al FS), contesta CERT/FALS a cada pregunta:

- (a) Fer-ho per fragment té generalment més qualitat (la simulació és més acurada)
- (b) Fer-ho per fragment incrementa el número de variables varying a interpolar

### **Exercici 10**

Si fem servir un GS per dividir cada triangle en quatre triangles, generant nous vèrtexs, on té sentit que es facin els càlculs d'il·luminació?

- (a) Al VS, GS o FS, indistintament
- (b) Al VS o al FS, indistintament
- (c) Al GS o al FS, indistintament
- (d) Només es podrà fer al FS.

### **Exercici 11**

Indica, per cadascuna d'aquestes condicions relacionades amb l'estat d'OpenGL, si pot afectar negativament al funcionament correcte de la selecció d'objectes basada en lectura del buffer de color (pseudocolor). Contesta SI (podria deixar de funcionar) o NO (no afecta):

- (a) `glDisable(GL_DEPTH_TEST)`
- (b) `glEnable(GL_LIGHTING)`
- (c) `glEnable(GL_BLEND);`
- (d) `glClearColor(0,0,0,0);`

## Exercici 12

Tenim una aplicació que només treballa amb escenes de fins a 24 primitives. Volem poder seleccionar primitives amb el mouse, però volem seleccionar totes les primitives sota el mouse (no només la visible). Una forma de fer-ho és utilitzar la tècnica basada en lectura del buffer de color. La idea seria codificar el color de cada primitiva fent que cada bit del color RGB correspongui a una primitiva, de forma que el color RGB tindrà, en binari, tot 0's excepte un bit a 1 que correspon a la primitiva (en decimal, els colors RGB seran potències de 2). Quina funció de blending caldria fer servir, per tal de representar al buffer de color tots els objectes que s'ha projectat a cada píxel?

- (a) glBlendFunc(GL\_ONE, GL\_ONE);
- (b) glBlendFunc(GL\_ZERO, GL\_ONE);
- (c) glBlendFunc(GL\_SRC\_ALPHA, GL\_ONE\_MINUS\_SRC\_ALPHA)
- (d) glBlendFunc(GL\_SRC\_ALPHA, GL\_ZERO);

## Exercici 13

Volem dibuixar lesombres que projecten els objectes sobre el pla  $Z = 5$ , amb la tècnica d'ombres per projecció. Suposant una llum infinitament allunyada en direcció de l'eix Z, indica per quina matriu  $4 \times 4$  cal multiplicar la posició del vertex en world space per tal d'obtenir lesombres projectades al dibuixar els objectes.

## Exercici 14

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 object space:

```
uniform sampler2D spheremap;
uniform vec3 obs; // posició de l'observador en object space
varying vec3 P; // posició del fragment en object space
varying vec3 N; // normal en object space

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

void main() {
    vec3 R;
```

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

# Encuentra el trabajo de tus sueños

Participa en retos y competiciones de programación



Escanéame y  
obtén más info!!

## Exercici 15

Aquest fragment de codi per generar reflexions és incomplet. Quina crida OpenGL (relacionada amb les matrius de transformació geomètrica) manca i on caldria afegir-la?

### // 1. Dibuixem el mirall a l'stencil buffer

```
glEnable(GL_STENCIL_TEST);  
glStencilFunc(GL_ALWAYS, 1, 1);  
glStencilOp(GL_KEEP, GL_KEEP, GL_REPLACE);  
glDepthMask(GL_FALSE); glColorMask(GL_FALSE...);  
dibuixar(mirall);
```

### // 2. Dibuixem els objectes en posició virtual

```
glDepthMask(GL_TRUE); glColorMask(GL_TRUE...);  
glStencilFunc(GL_EQUAL, 1, 1);  
glStencilOp(GL_KEEP, GL_KEEP, GL_KEEP);  
glPushMatrix();  
glLightfv(GL_LIGHT0, GL_POSITION, pos);  
glCullFace(GL_FRONT);  
dibuixar(escena);  
glPopMatrix();
```

### // 3. Dibuixem el mirall semitransparent

```
glDisable(GL_STENCIL_TEST);  
glLightfv(GL_LIGHT0, GL_POSITION, pos);  
glCullFace(GL_BACK);  
dibuixar(mirall);
```

### // 4. Dibuixem els objectes reals

```
dibuixar(escena);
```

## Exercici 16

Contesta les següents preguntes en relació al rendiment de diferents tècniques de simulació d'ombres:

(a) Sombres del projecció amb stencil, una font de llum estàtica: quantes vegades cal pintar l'objecte oclusor cada frame?

(b) Shadow mapping, dues fonts de llum dinàmiques, amb un FS: quantes vegades cal pintar l'escena cada frame?

## Exercici 17

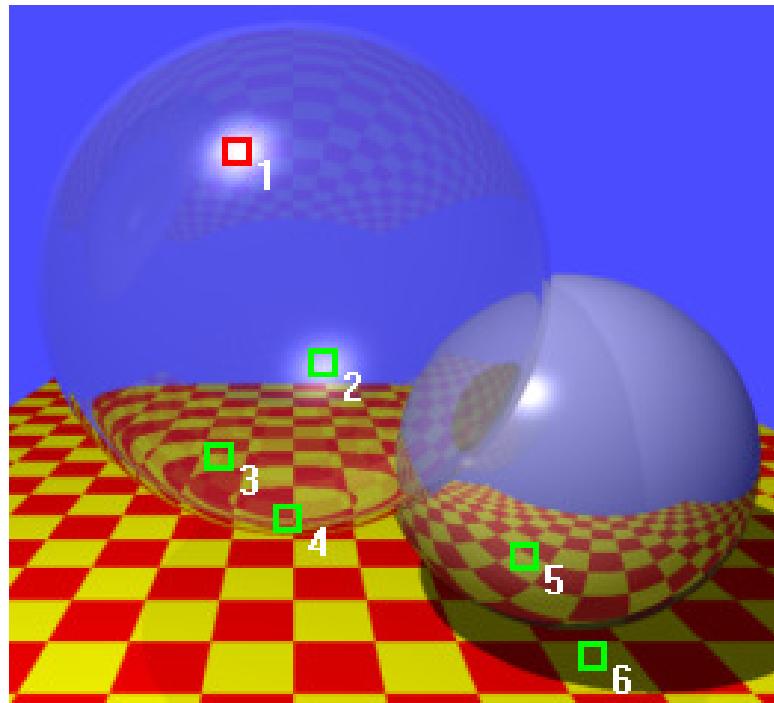
Indica quins d'aquests camins seria capaç de simular una implementació de l'algorisme clàssic de raytracing amb nivell màxim de recursitat 10:

- a) LSDE
- b) LSSSDE
- c) LSSSSSSSDE
- d) LDSSE



## Exercici 18

Aquesta figura s'ha generat amb raytracing clàssic:



Indica quins píxels (1-6) podrien veure afectat significativament el seu color final si canviem l'índex de refracció de l'esfera semi-transparent:

## 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.

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. Suposa que només hi ha un VS i un FS (no hi ha GS).

- Alpha blending
- Crides a  $dFdx$ ,  $dFdy$
- Pas de coordenades a clip space
- Rasterització

### Exercici 2

Hem produït un fragment amb color  $RGBA = (1.0, 0.5, 0.0, 0.4)$  corresponent al pixel  $(i,j)$ . El color  $RGBA$  del pixel  $(i,j)$  al buffer de color és  $(0.5, 0.5, 1.0, 1.0)$ . Si tenim activat alpha blending amb la crida

```
glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA)
```

quin serà el color resultant al buffer de color (assumint que passa tots els tests)?

### Exercici 3

Indica quina és la matriu que aconsegueix la conversió demandada (assumint que no hi ha transformació de modelat), usant aquestes abreviatures:

$$\begin{array}{ll} MV = \text{gl\_ModelViewMatrix} & MVP = \text{gl\_ModelViewProjectionMatrix} \\ MV^{-1} = \text{gl\_ModelViewMatrixInverse} & MVP^{-1} = \text{gl\_ModelViewProjectionMatrixInverse} \\ NM = \text{gl\_NormalMatrix} & \end{array}$$

- Pas d'un vèrtex de object space a eye space
- Pas d'un vèrtex de eye space a world space
- Pas d'un vèrtex de clip space a object space
- Pas de la normal de object space a eye space

#### Exercici 4

En una hipotètica versió de GLSL, no està definit el uniform **gl\_ModelViewProjectionMatrix** (però sí la resta de matrius de GLSL). Escriu, en codi GLSL vàlid, com faries la conversió de **gl\_Vertex** d'object space a clip space (assumint que no hi ha transformació de modelat).

#### Exercici 5

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

```
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; // cal que el calculeu vosaltres

    diff =

    return gl_FrontMaterial.diffuse * light.diffuse * diff;
}
```

#### Exercici 6

Per què ens pot ser útil aquesta sentència GLSL?

```
vec3 foo = gl_ModelViewMatrixInverse[3].xyz;
```

#### Exercici 7

Una textura de 1024x1024 requereix emmagatzemar en memòria, obviament, 1M texel. Quans texels cal emmagatzemar si la textura fa servir mipmapping?

# Encuentra el trabajo de tus sueños

Participa en retos y competiciones de programación



Escanéame y  
obtén más info!!

## Exercici 8

Aquest pseudocodi implementa environment mapping:

1. Calcular el vector unitari X
2. Calcular el vector R com a reflexió de X respecte la normal al punt
3. Utilitzar R per accedir al environment map i obtenir el color de l'entorn en direcció R

Què és el vector X del pas 1, i com el podeu calcular?

## Exercici 9

Quina magnitud de radiometria (o fotometria) representa millor l'energia que arriba a cada diferencial de superfície, i que es calcula al primer pass (light pass) de la tècnica two-pass raytracing?

Indica també una possible unitat de mesura d'aquesta magnitud.

## Exercici 10

Fes matching entre les magnituds/conceptes de radiometria de l'esquerra, i els conceptes de la dreta:

Flux	Energia d'un raig de llum
Radiància	Quantitat total d'energia emesa per una font de llum (per unitat de temps)
Intensitat	Propietats de reflectivitat d'un material
BRDF	Quantitat total d'energia emesa per una font de llum (per unitat de temps i angle sòlid)

## Exercici 11

Per un determinat objecte, tenim emmagatzemada la següent informació, relativa a la seva transformació de modelat: centre de l'objecte, vector de translació, matriu de rotació, paràmetres de l'escalat. Indica, **com a producte de 4 ó 5 matrius**, com calcular la transformació de modelat de l'objecte (vigileu l'ordre!). Feu servir la notació:

$T(x,y,z)$  -> matriu de translació segons el vector  $(x,y,z)$

$R$  -> matriu de rotació emmagatzemada

$S(x,y,z)$  -> matriu d'escalat segons els factors d'escalat  $x,y,z$

### **Exercici 12**

Quin és el principal avantatge d'utilitzar VBO en comptes de VA?

### **Exercici 13**

Tenim un normal map on les components RGB de cada texel codifiquen les components ( $nx', ny', nz'$ ) en espai tangent d'un vector unitari en la direcció de la normal perturbada. Donats els vectors T,B,N (tangent, bitangent i normal) d'una base ortonormal, en eye space, corresponents a un fragment, indica com calcularies la normal perturbada N' en eye space.

### **Exercici 14**

Quins són els sistemes de coordenades origen i destí de la transformació de projecció (projection transform), representada per la gl\_ProjectionMatrix en GLSL?

### **Exercici 15**

Indica quina condició han de satisfer les coordenades (x,y,z,w) d'un vèrtex en clip space, per tal de ser interior al frustum de visió d'una càmera perspectiva.

### **Exercici 16**

Aquesta és la declaració de la funció OpenGL glColorMask():

```
void glColorMask( GLboolean red, GLboolean green, GLboolean blue, GLboolean alpha )
```

- a) Quin efecte té aquesta funció?
  
- b) Menciona una tècnica concreta (dintre de les tècniques per simularombres, reflexions, transparències...) en la qual es faci servir, i perquè.

### Exercici 17

Completa (amb els uniforms apropiats) aquestes en línies en codi GLSL vàlid (assumint que només es pinten les cares frontface):

a) float shininess = // exponent de reflectivitat especular del material

b) vec4 posLum = // posició en eye space de la primera llum

### Exercici 18

De quin tipus GLSL (float, vec2, vec3...) és el resultat d'aquestes expressions?

a) dot(vec3(1,0,0), vec3(0,1,0))

a) texture2d(sampler, vec2(0.5,0.5));

b) cross (vec3(1,0,0), vec3(0, 1, 0))

c) vec3(1,0,0) \* vec3(0, 1, 0)

### Exercici 19

Què està fent aquest codi i en quina tècnica s'utilitza?

```
glLoadIdentity();
glTranslated(0.5, 0.5, 0.5);
glScaled(0.5, 0.5, 0.5);
gluPerspective(...);
gluLookAt(...)
```

### Exercici 20

En una aplicació volem reproduir l'ombra que projecten diferents globus aerostàtics sobre una pista plana i horitzontal, al vespre (sol prop de la posta), des de diferents punts de vista sobre la pista. Quin inconvenient pot tenir usar shadow mapping en aquest context?

# 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.

# Encuentra el trabajo de tus sueños

Participa en retos y competiciones de programación



Escanéame y  
obtén más info!!

Examen Final de Gràfics

Curs 2014-15 Q2

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. Suposa que només hi ha un VS i un FS (no hi ha GS).

- Escritura de gl\_Position
- Rasterització
- Retallat de primitives (clipping)
- Stencil test

### Exercici 2

Indica, per cada etapa de la llista de sota, si és ANTERIOR o POSTERIOR a la etapa de rasterització:

- (a) Primitive assembly
- (b) Processament del fragment
- (c) Ús de les funcions dFdX, dFdY
- (d) Pas de les coordenades a clip space

### Exercici 3

Indica al menys una tasca típica al pipeline de visualització programable, que es pugui fer tant abans com després de la rasterització.



WUOLAH

#### Exercici 4

Hem produït un fragment amb color  $\text{RGBA} = (1.0, 0.5, 0.0, 0.8)$  corresponent al pixel  $(i,j)$ . El color  $\text{RGBA}$  del pixel  $(i,j)$  al buffer de color és  $(1.0, 0.5, 1.0, 1.0)$ . Si tenim activat alpha blending amb la crida

```
glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA)
```

quin serà el color  $\text{RGBA}$  resultant al buffer de color (assumint que passa tots els tests)?

#### Exercici 5

Indica quina és la matriu que aconsegueix la conversió demandada (assumint que no hi ha transformació de modelat), usant aquestes abreviatures:

$\text{MV} = \text{gl\_ModelViewMatrix}$	$\text{MVP} = \text{gl\_ModelViewProjectionMatrix}$
$\text{MV}^{-1} = \text{gl\_ModelViewMatrixInverse}$	$\text{MVP}^{-1} = \text{gl\_ModelViewProjectionMatrixInverse}$
$\text{NM} = \text{gl\_NormalMatrix}$	$\text{I} = \text{Identitat}$

- a) Pas d'un vèrtex de eye space a world space
- b) Pas d'un vèrtex de clip space a object space
- c) Pas de la normal de object space a eye space
- d) Pas d'un vèrtex de object space a world space

#### Exercici 6

En una hipotètica versió de GLSL, no està definit el uniform **gl\_ModelViewProjectionMatrixInverse** (però sí la resta de matrius de GLSL). Escriu, en codi GLSL vàlid, com faries la conversió de **vec4 P** de clip space a object space (assumint que no hi ha transformació de modelat).

### Exercici 7

Completa el següent FS per tal que calculi correctament el vector L que intervé a la contribució difosa:

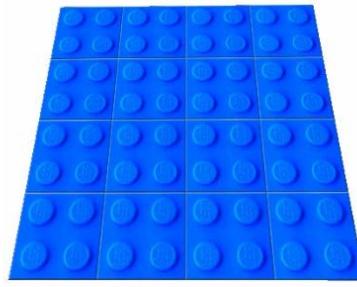
```
vec4 lightSource( vec3 N, vec3 P, gl_LightSourceParameters light ) {  
    // N és la normal en eye space  
    // P és el punt en eye space
```

**vec3 L =**

```
float diff = max( 0.0, dot( N,L ) );  
return gl_FrontMaterial.diffuse * light.diffuse * diff;  
}
```

### Exercici 8

Amb la textura de l'esquerra, volem texturar l'objecte Plane com a la dreta.



Completa la línia que necessitem al VS:

**gl\_TexCoord[0].st =**

### Exercici 9

Quants nivells de detall (nivells de LOD) formen la piràmide mipmapping completa d'una textura de 256x256 texels (tenint en compte el LOD 0)?

### Exercici 10

La següent figura mostra la textura d'un tauler d'escacs (a l'esquerra) i el Plane texturat (a la dreta):



El vèrtex inferior esquerra del polígon té coordenades de textura  $(0,0)$ , i el vèrtex superior dret  $(1,1)$ . Indica, a la imatge de la dreta, en quina regió es maximitza el valor de  $\frac{\partial s}{\partial x}$

### Exercici 11

Tenim un model amb **coordenades de textura 2D que cobreixen l'interval [-1,1]**. Ens demanen un VS que mostri el model projectat sobre aquest espai de textura (com a l'exercici UVunfold), de forma que oculti tot el viewport, amb independència de la càmera. Completa aquesta línia del VS demanat:

```
gl_Position = vec4( , , 0.0, );
```

### Exercici 12

Imagina un FS al qual li arriben les coordenades del punt (**vec4 P**) en un cert espai de coordenades.

Indica quin test hauries de fer sobre **P** per tal **d'eliminar (discard) els fragments que cauen a la meitat dreta** de la finestra OpenGL, amb mida 800x600:

a) Si **P** està en **clip space**

```
if ( ) discard;
```

b) Si **P** està en **NDC**

```
if ( ) discard;
```

c) Si **P** està en **window space**

```
if ( ) discard;
```

### Exercici 13

Escriu quina matriu 4x4 necessitem per simular la reflexió de l'escena respecte un mirall situat al pla  $Y=0$ .

# Encuentra el trabajo de tus sueños

Escanéame y  
obtén más info!!



Participa en retos y competiciones de programación

## Exercici 14

Tenim un cub format per 6 cares i 8 vèrtexs.

a) Si volem que cada corner (vèrtex associat a una única cara) tingui la normal de la cara a la que pertany, quants vèrtexs necessitem en el VBO del cub?

b) Si volem que cada vèrtex del cub tingui com a normal el promig de les normals de les cares que incideixen al vèrtex, quants vèrtexs necessitem (com a mínim) en el VBO del cub?

## Exercici 15

Tenim un normal map on les components RGB de cada texel codifiquen les components ( $nx', ny', nz'$ ) en espai tangent d'un vector unitari en la direcció de la normal perturbada. Donats els vectors T, B, N (tangent, bitangent i normal) d'una base ortonormal, en eye space, corresponents a un fragment, indica com calcularies la normal perturbada N' en eye space.

## Exercici 16

Quins són els sistemes de coordenades origen i destí de la transformació representada per la gl\_ModelViewMatrix en GLSL?

## Exercici 17

Aquesta és la declaració de la funció OpenGL glDepthMask():

```
void glDepthMask( GLboolean foo)
```

Quin efecte té aquesta funció?

## Exercici 18

De quin tipus GLSL (float, vec2, vec3...) és el resultat d'aquestes expressions?

a) dot(vec3(1,0,0), vec3(0,1,0))

b) cross (vec3(1,0,0), vec3(0, 1, 0))



### **Exercici 19**

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

- a) Shadow mapping, suposant que la llum és dinàmica
- b) Ombres per projecció, versió sense stencil
- c) Reflexió amb objectes virtuals, amb stencil (ignoreu els passos en que només es dibuixa el mirall)
- d) Projective Texture Mapping, amb una textura fixa

### **Exercici 20**

Tenim una aplicació amb una escena opaca, sense miralls, amb una única font de llum puntual situada al punt (0,0,0) en coordenades de l'observador. Quina tècnica creus apropiada per reproduir les ombres en aquest cas?

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. Suposa que només hi ha un VS i un FS (no hi ha GS).

- Divisió de perspectiva
- Escriptura de gl\_Position
- glDrawElements
- Rasterització

### Exercici 2

Indica, per cada tasca/etapa de la llista de sota, si és ANTERIOR o POSTERIOR a la etapa de rasterització:

- (a) Geometry Shader
- (b) Fragment Shader
- (c) Stencil Test
- (d) Alpha blending

### Exercici 3

Quants nivells de detall (nivells de LOD) formen la piràmide mipmapping completa d'una textura de 512x512 texels (tenint en compte el LOD 0)?

#### Exercici 4

Tenim un octaedre representat amb una malla triangular formada per 6 vèrtexs i 8 triangles. Volem construir un VBO per representar aquest octaedre, 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 l'octaedre aparegui il·luminat correctament). Cada coordenada/component estarà representada per un float (4 bytes).

(a) Quants vèrtexs necessitem representar al VBO?

(b) Quants índexs (*elements*) es necessiten a l'array d'*índexs*?

(c) Quina mida (en bytes) tindrà l'array de coordenades de vèrtexs?

#### Exercicis 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):

$M = \text{modelMatrix}$	$M^{-1} = \text{modelMatrixInverse}$
$V = \text{viewingMatrix}$	$V^{-1} = \text{viewingMatrixInverse}$
$P = \text{projectionMatrix}$	$P^{-1} = \text{projectionMatrixInverse}$
$N = \text{normalMatrix}$	$I = \text{Identitat}$

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

b) Pas d'un vèrtex de object space a eye space

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

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

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

B) Pas d'un vèrtex de world space a clip space

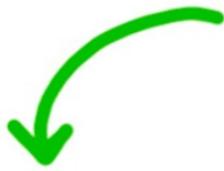
C) Pas d'un vèrtex de object space a model space

D) Pas de la normal de object space a eye space

# Encuentra el trabajo de tus sueños



Escanéame y obtén más info!!



## Exercici 7

Tenim activat alpha blending amb la crida

```
glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA)
```

Hem produït un fragment amb color RGBA = (1.0, 0.5, 0.0, **a**) corresponent al pixel (i,j). El color RGBA del pixel (i,j) al buffer de color és (1.0, 1.0, 0.0, 0.0). Quin valor ha de tenir **a** si volem que les components RGB del resultat siguin (1.0, 0.8, 0.0)?



## Exercici 8

Indica quina és la diferència més important entre els models d'il·luminació local i els models d'il·luminació global.

## Exercici 9

Completa el següent FS per tal que calculi correctament el vector L que intervé a la contribució difosa:

```
uniform vec4 matDiffuse;  
uniform vec4 lightDiffuse, lightPosition;  
  
vec4 lightSource( vec3 N, vec3 P ) {  
    // N és la normal en eye space  
    // P és el punt en eye space  
    vec3 L =
```

```
        float diff = max( 0.0, dot( N,L ) );  
        return matDiffuse * lightDiffuse * diff;  
    }
```

## Exercici 10

Indica, en la notació estudiada a classe,  $L(D|S)^*E$ , quins light paths són suportats per ray-tracing clàssic.



### Exercici 11

Volem generar amb RayTracing una imatge 1024x1024 d'una escena interior tancada. Tots els objectes són opacs i perfectament difosos, i hi ha quatre fonts de llum.

- a) Quants rajos primaris caldrà traçar?
- b) Quants shadow rays caldrà traçar, en total?
- c) Quants rajos reflectits caldrà traçar, en total?
- d) Quants rajos transmesos caldrà traçar, en total?

### Exercici 12

Relaciona cada concepte de l'esquerra amb un concepte de la dreta:

- |                |   |
|----------------|---|
| 1. Flux        | A. $\phi$ per unitat d'angle sòlid  |
| 2. Intensitat  | B. Mesurat en lúmens  |
| 3. Irradiància | C. Energia (per unitat de temps) que travessa un punt en una determinada direcció |
| 4. Radiància   | D. Mesurat en lux   |

1 →

2 →

3 →

4 →

### Exercici 13

Completa l'eqüació general del rendering:

$$L_o(x, \omega_o) = L_e(x, \omega_o) + \int f(x, \omega_i, \omega_o) L_i(x, \omega_i) \boxed{\phantom{000}} d\omega_i$$

### Exercici 14

Quin és el nom de l'equació que ens permet saber la quantitat de llum reflectida i la quantitat de llum transmesa, en funció de diferents paràmetres (angle incident, etc)?

## Exercici 15

Volem dibuixar una escena amb alguns objectes opacs i altres translúcids, amb alpha blending. Donat que no volem ordenar els objectes, fem servir aquest codi:

```
glEnable(GL_BLEND);
glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);

if (depthTestOpaque) glEnable(GL_DEPTH_TEST);
else glDisable(GL_DEPTH_TEST);
glDepthMask(maskOpaque);
drawObjects(opaqueObjects);

if (depthTestTranslucent) glEnable(GL_DEPTH_TEST);
else glDisable(GL_DEPTH_TEST);
glDepthMask(maskTranslucent);
drawObjects(translucentObjects);
```

Indica els valors més adients per les variables booleanes:

- (a) depthTestOpaque
- (b) depthTestTranslucent
- (c) maskOpaque
- (d) maskTranslucent

## Exercici 16

Tenim una escena amb diferents objectes, cadascú amb la seva transformació de modelat. Volem simular les reflexions en un mirall pla fent servir la tècnica de reflexió amb objectes virtuals. Hem calculat la matriu R de reflexió respecte el pla del mirall, usant els coeficients (a,b,c,d) del pla en world space. Fent servir la notació

$$\begin{array}{ll} M = \text{modelMatrix (original)} & M^{-1} = \text{modelMatrixInverse (original)} \\ V = \text{viewingMatrix} & V^{-1} = \text{viewingMatrixInverse} \end{array}$$

indica quin producte de matrius és necessari **en el pas en que es dibuixen els objectes en posició virtual:**

- (a) Passar un vèrtex de object space a eye space
- (b) Passar un vèrtex de object space a world space

## Exercici 17

Tenim un model amb **coordenades de textura 2D que cobreixen l'interval [-1,1]**. Ens demanen un VS que mostri el model projectat sobre aquest espai de textura (com a l'exercici UVunfold), de forma que oculti tot el viewport, amb independència de la càmera. Completa aquesta línia del VS demandat:

```
layout (location = 3) in vec2 texCoord;  
gl_Position = vec4( , , 0.0, );
```



## Exercici 18

Tenim aquest fragment de FS:

```
void main() {  
    vec3 I = normalize(Pos);  
    vec3 R = reflect(I, N);  
    fragColor = sampleTexture(R);  
}
```

(a) Quina tècnica està implementant?

(b) En quin espai de coordenades està treballant?

## Exercici 19 i 20

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

a) Shadow mapping, suposant que la llum és dinàmica

b) Ombres per projecció, versió sense stencil

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

d) Environment Mapping, amb una textura fixa

## Preguntes per a l'avaluació de les competències transversals

### Pregunta 1

Explica què és i per què es fa servir la funció de transferència (transfer function) en aplicacions mèdiques

### Pregunta 2

Explica per què serveix la tècnica de *Image Registration*, dins les aplicacions dels gràfics en medicina.

# Encuentra el trabajo de tus sueños

Participa en retos y competiciones de programación



Escanéame y  
obtén más info!!

Examen Final de Gràfics

Curs 2015-16 Q2

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. Suposa que només hi ha un VS i un FS (no hi ha GS).

- Divisió de perspectiva
- glDrawElements
- Rasterització
- Transformació a Clip Space

## Exercici 2

Indica, per cada tasca/etapa de la llista de sota, si és ANTERIOR o POSTERIOR a la etapa de rasterització:

- (a) Geometry Shader
- (b) Fragment Shader
- (c) dFdx, dFdy
- (d) Stencil Test

## Exercici 3

El LOD 0 d'una textura té 1024 x 512 texels. Quina mida té el LOD 2 d'aquesta mateixa textura?



#### Exercici 4

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).

(a) Quants vèrtexs necessitem representar al VBO?

(b) Quants índexs (*elements*) es necessiten a l'array d'*índexs*?

#### Exercicis 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):

$M = \text{modelMatrix}$	$M^{-1} = \text{modelMatrixInverse}$
$V = \text{viewingMatrix}$	$V^{-1} = \text{viewingMatrixInverse}$
$P = \text{projectionMatrix}$	$P^{-1} = \text{projectionMatrixInverse}$
$N = \text{normalMatrix}$	$I = \text{Identitat}$

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

b) Pas d'un vèrtex de eye space a world space

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

d) Pas d'un vèrtex de world space a clip space

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

f) Pas d'un vèrtex de object space a world space

g) Pas d'un vèrtex de object space a eye space

h) Pas de la normal de object space a eye space

## Exercici 7

Tenim activat alpha blending amb la crida

```
glBlendFunc(GL_ONE_MINUS_SRC_ALPHA, GL_SRC_ALPHA)
```

Hem produït un fragment amb color RGBA = (1.0, 0.5, 0.0, 0.2) corresponent al pixel (i,j). El color RGBA del pixel (i,j) al buffer de color és (1.0, 1.0, 0.5, 0.0). Indica quin serà el color RGBA resultant del blending, amb les operacions que duen a aquest resultat.

## Exercicis 8 i 9

Una forma d'expressar l'equació general del rendering és la següent:

$$L_o(\mathbf{x}, \omega_o, \lambda, t) = L_e(\mathbf{x}, \omega_o, \lambda, t) + \int_{\Omega} f_r(\mathbf{x}, \omega_i, \omega_o, \lambda, t) L_i(\mathbf{x}, \omega_i, \lambda, t) (\omega_i \cdot \mathbf{n}) d\omega_i$$

- (a) Què creus que representa  $\lambda$  ?
- (b) Què representa  $\Omega$  ?
- (c) Indica quin nom té la funció  $f_r$
- (d) Què són els tres primers paràmetres de la funció  $f_r$ ?

## Exercici 10

Completa el següent FS per tal que calculi correctament el terme de Phong de la il·luminació:

```
uniform vec4 matAmbient, matDiffuse, matSpecular;
uniform vec4 lightAmbient, lightDiffuse, lightSpecular, lightPosition;
uniform float matShininess;

vec4 light(vec3 N, vec3 V, vec3 L)
{
    vec3 R = normalize( 2.0*dot(N,L)*N-L );
    float NdotL = max( 0.0, dot( N,L ) );
    float RdotV = max( 0.0, dot( R,V ) );
    float Idiff = NdotL;
    float Ispec = 0;

    if (NdotL>0) Ispec =
        [REDACTED]

    return
        matAmbient * lightAmbient +
        matDiffuse * lightDiffuse * Idiff +
        matSpecular * lightSpecular * Ispec;
}
```

## Exercici 11

Indica, en la notació estudiada a classe,  $L(D|S)^*E$ , quins light paths són suportats per:

- (a) Raytracing clàssic
- (b) Two-pass raytracing

# Encuentra el trabajo de tus sueños

Participa en retos y competiciones de programación



Escanéame y  
obtén más info!!

## Exercicis 12 i 13

Volem generar amb RayTracing una imatge 256x256 d'una escena interior tancada. Els objectes de l'escena estan configurats de forma que la probabilitat de que qualsevol raig intersecti un mirall és de 0.5 (l'altre 0.5 correspon a un objecte difós).

- Quants rajos primaris caldrà traçar?
- Quants rajos reflectits caldrà traçar, en total, si admetem un únic nivell de recursitat (per exemple LDSE)?
- Quants rajos reflectits caldrà traçar, en total, si admetem dos nivells de recursitat (per exemple LDSSE)?
- Quants rajos primaris caldrà traçar si volem antialiàsing amb 4 mostres per píxel?

## Exercici 14

Quin concepte de radiometria/fotometria és el més adient per mesura la quantitat d'energia per unitat de temps que arriba a una superfície, per unitat d'àrea (unitats  $\text{W/m}^2$ )?

## Exercici 15

Indica, per cadascuna de les següents magnituds, si afecta (SI) o no (NO) a la direcció del raig transmès, d'acord amb la Llei de Snell (considera també efectes indirectes):

- Velocitat de propagació de la llum als medis
- Longitud d'ona de la llum
- Angle d'incidència
- Color difós de la superfície (Kd)



### Exercici 16

Tenim una aplicació que no suporta MipMapping, però volem simular el resultat de GL\_LINEAR\_MIPMAP\_LINEAR en GLSL. Completa aquest codi, on lambda és un float amb el nivell de LOD (no necessàriament enter) més adient pel fragment:

```
vec4 sampleTexture(sampler2D sampler, vec2 texCoord, float lambda)
{
    vec4 color0 = textureLod(sampler, texCoord, floor(lambda));
    vec4 color1 = textureLod(sampler, texCoord, floor(lambda)+1);
    return [REDACTED]
}
```

Podeu assumir que textureLod(P,sampler,lod) fa un accés bilineal a textura al punt P usant el nivell especificat a lod.

### Exercici 17

A classe hem estudiat un algorisme per simular reflexions especulars en miralls plans basat en objectes virtuals. Explica clarament per què és necessari, en general, fer servir la versió amb stencil buffer.

### Exercici 18

Amb la textura de l'esquerra, volem texturar l'objecte Plane com a la dreta.



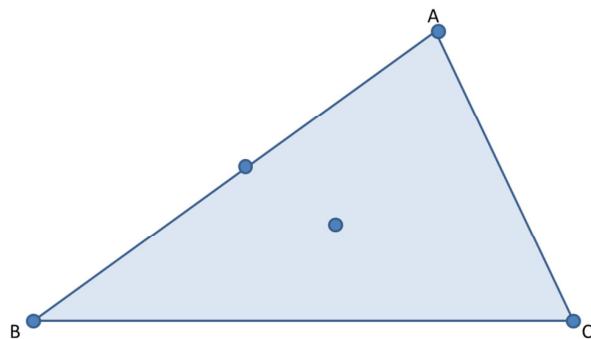
Completa la línia que necessitem al VS:

vTexCoord = [REDACTED]

### Exercici 19

Indica les coordenades baricèntriques  $(\alpha, \beta, \gamma)$  associades als vèrtexs A, B, C del triangle, pels punts que s'indiquen:

(a) Baricentre del triangle



(b) Punt mig de l'aresta AB

(c) Si un punt P té coordenades  $(\alpha, \beta, \gamma)$  amb  $\alpha = 0.2$  i  $\beta = 0.3$ , què podem dir de  $\gamma$ ?

(d) Si un punt P té coordenades  $(\alpha, \beta, \gamma)$  amb  $\alpha < 0$ , què podem dir de P en relació al triangle?

### Exercici 20

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 = [REDACTED LINE];

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

    float trueDepth = [REDACTED LINE];

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