

- 1) 24-25 Q4
- 2) 23-24 Q2
- 3) 22-23 Q2
- 4) 22-23 Q1
- 5) 21-22 Q2
- 6) Qüestionari 18 Març
- 7) Reflexió AteneaLabs
- 8) Extres

1) 24-25 Q1

## Pregunta 1

Indica el valor que ha de tenir la variable entera `E` en aquest fragment de codi, si estem dibuixant una malla de triangles amb 33 cares, 41 arestes i 13 vèrtexs:

`g glBindVertexArray(VAO);`  
`g glDrawElements(GL_TRIANGLES, E, GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, &index);`

Algunas otras      GL\_TRIANGLES      E, GL\_UNSIGNED\_INT, (GLvoid\*)0;  
                            GL\_POINTS      GL\_UNSIGNED\_BYTE      GL\_UNSIGNED\_SHORT  
                            GL\_LINES      GL\_UNSIGNED\_BYTE  
                            GL\_LINE\_STRIP      GL\_UNSIGNED\_BYTE

33 cares  $\Rightarrow$  33 triangles (Backface culling)  
3 vertices per triangle  
↓

$E = 33 \times 3 = 99$

## Pregunta 2

Tria l'espai de coordenades en què ha d'estar `P` per tal que la transformació `projectionMatrixInverse * P` tingui sentit:

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

Projection ( $CS \rightarrow CS'$ )

Inverse ( $CS' \rightarrow CS$ )

### Pregunta 3

Indica el valor que ha de tenir la constant **S** en aquest fragment de codi:

```
const unsigned int S = 36;  
GLubyte read[S];  
glReadPixels(x, y, 4, 3, GL_RGB, GL_UNSIGNED_BYTE, read);
```

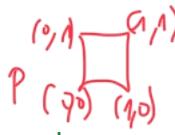
llegeix ( $x, y$ )        
 $\rightarrow$  pixels

$$S = 3 \times 4 \text{ pixels} \cdot 3 \text{ components} = 12 \cdot 3 = 36$$

### Pregunta 4

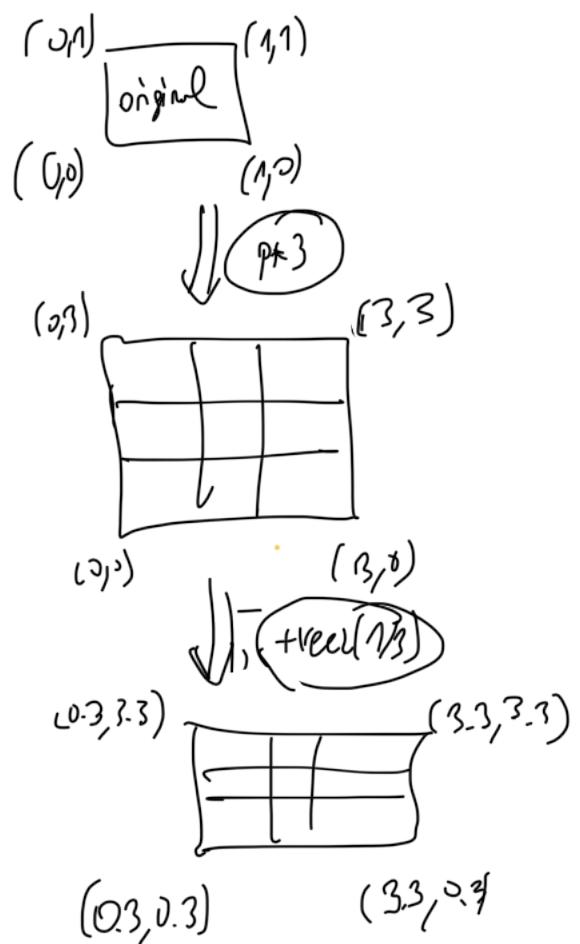
Indica el valor de **X** per tal que el fragment shader generi aquesta sortida amb l'objecte **plane** i la textura **smile**:

```
uniform sampler2D smile;  
const float X = ?;  
void main()  
{  
    vec2 p = vtexCoord;  
    vec2 st = p * X + vec2(1/X);  
    fragColor = texture(smile, st);  
}
```

$P$  

Multiplicar  $\rightarrow$  zoom out (veus més espai de controles)

Dividir  $\rightarrow$  zoom in (el veus)



## Pregunta 5

S'assumeix que tots els roigs primaris interseguen a objecte d'imatge

Tenim una escena tancada que conté 73 objectes difosos i 4 llums puntuals. Volem generar una imatge de 1024 x 768 píxels amb Ray Tracing clàssic. Quants shadow rays cal llançar?

- 1 ray per pixel
- Des de cada objecte es llancen més roigs per cercar si hi ha interseccions amb parts de llum o altres objectes de l'escena (per veure si estan a l'antre)

$$\begin{aligned} \text{Nombre de pixels} &= 1024 \times 768 \\ \text{Nombre de llums} &= 4 \end{aligned} \quad \left. \begin{array}{l} \text{Nombre de pixels} = 1024 \times 768 \\ \text{Nombre de llums} = 4 \end{array} \right\} 1024 \times 768 \times 4 = \boxed{3145728 \text{ shadow rays}} \quad \boxed{\text{3145728 shadow rays}}$$

## Pregunta 6

Per un determinat píxel ( $x, y$ ), els valors al frame buffer són: `depthBuffer[x, y]=0.5`, `stencilBuffer[x, y]=4`. El test s'ha configurat amb `glStencilFunc(GL_ALWAYS, 6, 255)` i `glStencilOp(GL_ZERO, GL_INCR, GL_REPLACE)`. Si es genera un fragment per aquest píxel, amb `gl_FragCoord.z = 0.6`, indica quin serà el resultat final al `stencilBuffer`:

- 4
- 5
- 6
- 3

Stencil Buffer      (cf. mal<sup>2</sup>)  
glStencilFunc(GL\_ALWAYS, 6, 255)  
(6 & 255) GL\_ALWAYS (4 & 255)  
↓      passen stencil

Depth Test  
per defecte  $GL_LESS$  ( $\text{fragz} < \text{depth buffer}$ )  
però tenim  $0.6 \cancel{<} 0.5$       no passen depth

`glStencilOp(GL_ZERO, GL_INCR, GL_REPLACE)`

stencil	depth	depth	$(4+1 = 5)$
fail	fail	pass	

## Pregunta 7

Indica quina és l'opció més adient per a completar aquest codi a l'espai o espais indicats per "":

```
// draw a scene containing opaque and semitransparent objects
void X::paintGL()
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glEnable(GL_DEPTH_TEST); → opacs require depth test
    glDepthMask(GL_TRUE);
    opaque_objects.draw(); // unsorted
    glEnable(GL_BLEND); → alpha blending, we use semi transparents
    glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
    -----
    semitransparent_objects.draw(); // unsorted
    glDisable(GL_BLEND);
}
```

- el conten
- glDepthMask(GL\_FALSE)
  - glDepthMask(GL\_TRUE)
  - glColorMask(GL\_FALSE) *No vullien res*
  - glDisable(GL\_DEPTH\_TEST)

→ Valen que els elements de vedere tots els en la pàgina, ja que dibuixen uns objectes semi transparents.

(  
clavar per completar errors i tiquets  
un objecte opac sobre un transparent.

stencil func / depth test  
 no stencil  
 gl depth buffer

Si es dibuixa mirell i per depth test l'estencil  
 & sobrecapa amb

**Pregunta 8:** "Indica quina substitució de la línia 04 no alteraria la imatge resultant:"

```

01 // 1. Draw mirror onto stencil buffer
02 glEnable(GL_STENCIL_TEST);
03 glStencilFunc(GL_ALWAYS, 1, 1); ref
04 glStencilOp(GL_KEEP, GL_KEEP, GL_REPLACE); (amb ref)
05 glDepthMask(GL_FALSE); glColorMask(GL_FALSE...);
06 draw(mirror);

07 // 2. Draw virtual objects

08 glDepthMask(GL_TRUE); glColorMask(GL_TRUE...);

09 glStencilFunc(GL_EQUAL, 1, 1);

10 glStencilOp(GL_KEEP, GL_KEEP, GL_KEEP);

11 ...

13 glCullFace(GL_FRONT);

14 draw(scene);

15 ...

16 // 3. Draw mirror

17 glDisable(GL_STENCIL_TEST);

18 ...

19 glCullFace(GL_BACK);

20 draw(mirror);

21 // 4. Draw real objects
22 draw(scene);



- glStencilOp(GL_KEEP, GL_INCR, GL_KEEP)
- glStencilOp(GL_KEEP, GL_KEEP, GL_ZERO)
- glStencilOp(GL_KEEP, GL_REPLACE, GL_KEEP)
- glStencilOp(GL_ZERO, GL_KEEP, GL_REPLACE)

```

Al 2 no s'apartava objecte virtual pel GL-KEEP  
 Totore el replace amb l'1(ref).

*matrix problem*

↗ No volem pre a t l'actual dep  
 del depth fent!

*me lo pels, stencil hi GL ALWAYS*

## Pregunta 9

(cross)

Indica quina ha de ser la funció **G** perquè aquest FS tingui sentit. escriu únicament l'identificador de la funció GLSL:

```
#version 330 core

uniform mat3 normalMatrix;

in vec2 vtexCoord;
in vec3 pos; // posicio en object space

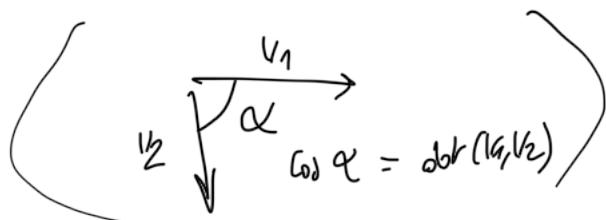
out vec4 fragColor;

void main()
{
    vec3 q = pos;
    vec3 dx = dFdX(q); } on conria el posició en OS que sumo en
    vec3 dy = dFdY(q); direcció X o Y

    vec3 n = G(dx,dy);
    n = normalize(normalMatrix * n);
    fragColor = vec4(n.z);
}
```



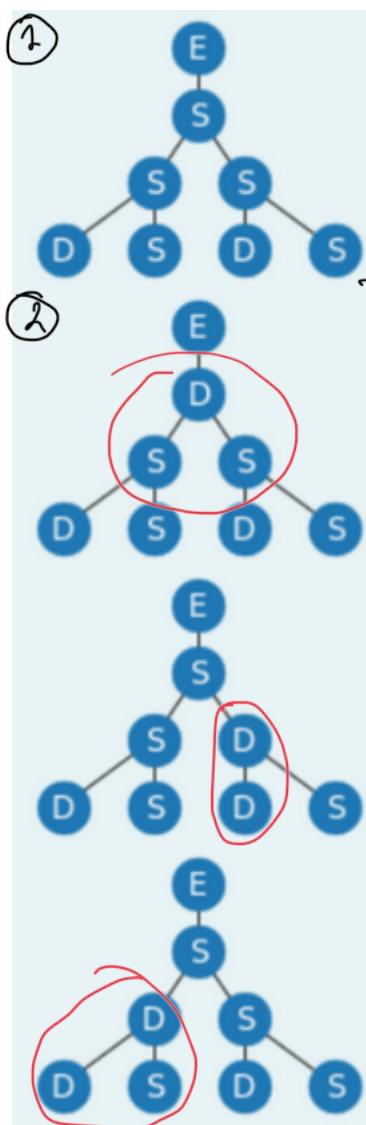
$$N = \cos(V_1, V_2)$$



Pregunta 10

glossy ← amb  
 S espejuelo  
 D difus

Indica quin arbre de rajos pot ser generat per Ray Tracing clàssic:



Cada S genera un nou S i D.  
 Un conjunt de {Reflexió} + els shadow rays

OK

D no pot tenir fills  
 a RT clàssic

A RT, quan un ray colpeja una superfície, tenim com a més 2 rajos:

- 1 Reflexió ( $R \circ S$ )
- 1 Refracció
- 1 o més rajos d'ombra

## Pregunta 11

Indica el valor positiu més petit de la constant t per tal que el FS generi aquesta sortida amb l'objecte plane i la textura smile.

```
uniform sampler2D smile;
uniform float time;

const float PI = 3.141592653589793;
const float t = 0.25

void main()
{
    vec2 p = vtexCoord;
    float a = 2 * PI * t;
    vec2 st = vec2( cos(a)*p.x - sin(a)*p.y, sin(a)*p.x +
cos(a)*p.y);
    fragColor = texture(smile, st);
}
```

## Pregunta 12

Un estimador de la il·luminació global amb molta variància es distingeix per...

- Té problemes per simular camins de llum especulars
- Suporta una gran varietat de camins de llum
- Suporta una gran varietat de materials
- Genera imatges que requereixen noise filtering

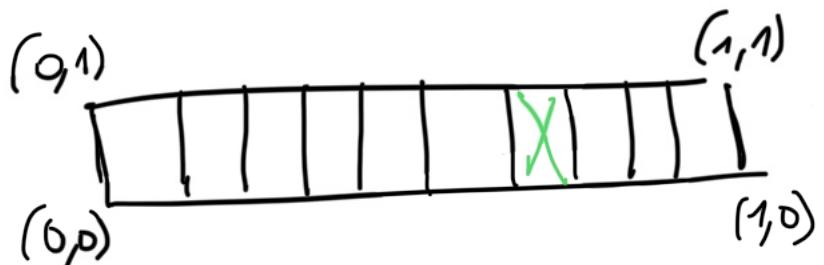
### Pregunta 13

Assigna a cada crida el seu ordre relativ al pipeline d'OpenGL (sense Geometry Shader):

- 1 glGenVertexArrays (crear contenedor vértices)
- 2 glBindBufferData (Ponle datos al VBO) (después crearán el VAO)
- 3 VS execution finishes (Falls o clipping)
- 4 Viewport transformation (NDC → WS)

### Pregunta 14

Tenim aquest fragment shader:



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

Indica amb quina opció (**factor**, **offset**) es genera la textura correcta:

- **factor=vec2(1.0, 1.0); offset=vec2(6.0, 6.0);**
- **factor=vec2(0.1, 0.6); offset=vec2(6.0, 6.0);**
- **factor=vec2(0.1, 1.0); offset=vec2(0.6, 0.0);**
- **factor=vec2(0.6, 0.6); offset=vec2(0.0, 0.0);**

## Pregunta 15

En quin espai **no és correcte** interpolar simplemet ( $s$ ,  $t$ ) quan usem una càmera perspectiva?

- object space
- clip space
- world space

• NDC

Perquè ja s'ha fet la divisió de perspective, el color no necessita interpolar  $(\frac{s}{w}, \frac{t}{w}, \frac{1}{w})$

## Pregunta 16

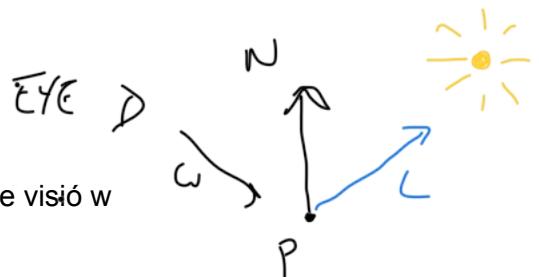
Siguin

- $P$  = punt visible de l'escena en una certa direcció de visió  $w$
- $N$  = normal de la superfície en el punt  $P$
- $L$  = vector unitari del punt  $P$  cap a la llum

En última instància, resoldre el problema de la il·luminació global, a nivell de cada punt de la imatge, equival a estimar...

- La irradiància de  $P$  en direcció  $w$
- $N \cdot L$
- La irradiància que arriba a  $P$

• La radiància de sortida al punt  $P$  en direcció  $-w$



## Pregunta 17

Siguin:

- M: submatriu 3x3 de la modelMatrix
- V: submatriu 3x3 de la viewMatrix,

} V.M es multiplicació de dreta a esq

la normalMatrix es pot calcular com...

- $(MV)^{-1}$  → object → eye

- $M^{-1}$

- $(VM)^{-T}$

- $(VM)^{-1}$

→ assign que les normals seguiran  
sent +, 2º es deformar vectors

## Pregunta 18

En aquest fragment de codi, la matriu M ha de ser...

```
// Pass 2. Draw the scene using the shadowmap
glClear(GL_DEPTH_BUFFER_BIT | GL_COLOR_BUFFER_BIT);
program->bind();
program->setUniformValue("lightViewMatrix",
lightCamera.viewMatrix());
program->setUniformValue("lightProjectionMatrix",
lightCamera.projectionMatrix());
QMatrix4x4 M = _____;
program->setUniformValue("biasMatrix", M);
program->setUniformValue("lightPos", lightCamera.getObs());
...
drawPlugin()->drawScene();
```

*NDC [ -1, 1 ] → texture space [ 0, 1 ]*

VS:

```
...
out vec4 vtexCoord;
void main()
{
    vtexCoord =
biasMatrix*lightProjectionMatrix*lightViewMatrix*vec4(vertex, 1.0);
    ...
}
[1.0 0.0 0.0 0.0]
|0.0 1.0 0.0 0.0|
|0.0 0.0 1.0 0.0|
|0.5 0.5 0.5 1.0|
[0.5 0.0 0.0 0.0]
|0.0 0.5 0.0 0.0|
|0.0 0.0 0.5 0.0|
[-0.25 -0.25 -0.25 1.0]
[0.5 0.0 0.0 0.0]
|0.0 0.5 0.0 0.0|
|0.0 0.0 0.5 0.0|
|0.5 0.5 0.5 1.0]
[0.5 0.0 0.0 0.0]
|0.0 0.5 0.0 0.0|
|0.0 0.0 0.5 0.0|
|0.25 0.25 0.25 1.0]
```

*[ -1, 1 ] → [ 0, 1 ]  
or [ 0, 5 + 0, 5 ]*

## Pregunta 19

El punt 3D que resulta d'aplicar la transformació representada per la matriu

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 15 \\ 25 \\ 15 \\ 5 \end{pmatrix} = \begin{pmatrix} 15 \\ 4 \cdot 25 \\ 15 \\ 5 \end{pmatrix} = \begin{pmatrix} 15 \\ 100 \\ 15 \\ 5 \end{pmatrix} = \begin{pmatrix} 3 \\ 20 \\ 3 \end{pmatrix}$$

al punt  $(15.00, 25.00, 15.00, 5.00)$  és:

Trieu-ne una:

- $(15.00, 25.00, 15.00)$
- $\underline{\underline{(3.00, 20.00, 3.00)}}$
- $(100.00, 15.00, 15.00)$
- $(15.00, 100.00, 15.00)$

## Pregunta 20

Considera un sistema de cinema 360° basat en una pantalla gegant de forma esfèrica, de radi  $R = 22$  m, la qual rep imatges projectades per un projector 360° situat al centre. Si el projector emet 7500 lumens, distribuïts de manera uniforme en totes direccions, indica la il·luminància resultant en la pantalla.

Trieu-ne una:

$$\text{Superficie } A = 4\pi(22)^2$$

- 0.785029832
- 0,432324122
- 1,2331219764351
- 2,004122001122

$$E = \frac{P}{A} = \frac{7500}{4\pi(22)^2} = 1,23312..$$

2) 23-24 Q2

## Pregunta 1

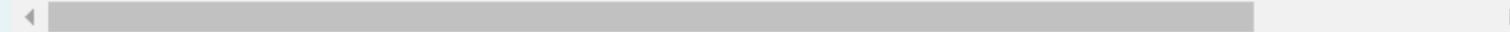
Incorrecte

Puntuació 0,00  
sobre 1,00

Marca la pregunta

Indica el valor de u (enter) que fa correcte aquest codi:

```
QImage img5("file.png");
QImage T6 = img5.convertToFormat(QImage::Format_ARGB32);
glGenTextures(1, &textureId7);
glBindTexture(GL_TEXTURE_2D, textureId7);
glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, T6.width(), T6.height(), 0, GL_RGBA, GL_UNSIGNED
g.glActiveTexture(GL_TEXTURE4);
g glBindTexture(GL_TEXTURE_2D, textureId7);
program->bind();
program->setUniformValue("textureMap", u); // sampler2D
```



[Cast]

Resposta:

0 4



La resposta correcta és: 4

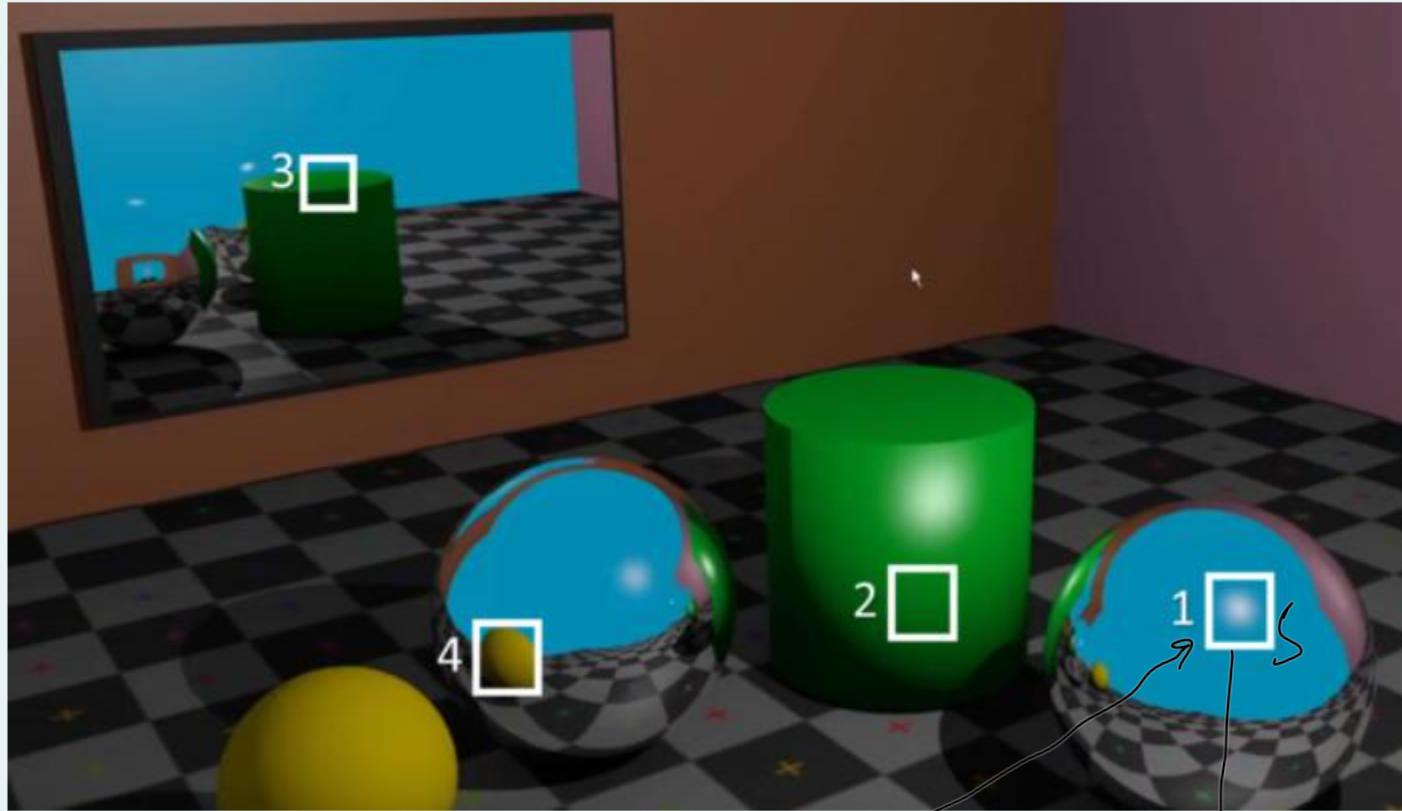
## Pregunta 2

Correcte

Puntuació 1,00  
sobre 1,00

◀ Marca la  
pregunta

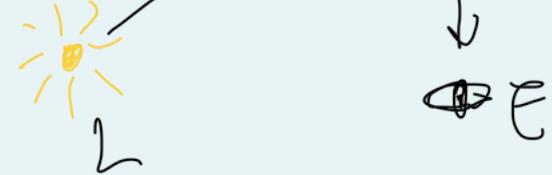
El light path que explica el color dominant al pixel central del quadrat 1 és...



[Cast]

Trieu-ne una:

- LSDSE
- LSDE
- LDDE
- LSE ✓



La resposta correcta és: LSE

### Pregunta 3

Correcte

Puntuació 1,00  
sobre 1,00

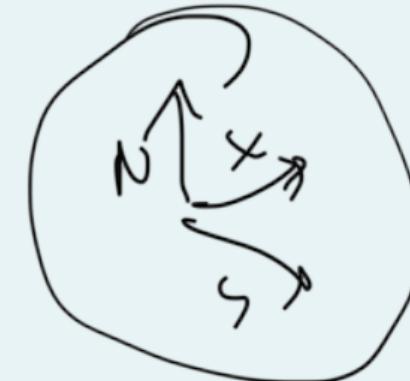
Marca la pregunta

Soposa que P és un punt en eye space. Una forma d'obtenir en un FS la direcció del vector normal en eye space és...

[Cast]

Trieu-ne una:

- normalize(cross(dFdx(P), dFdy(P))) ✓
- normalMatrix \* P
- normalize(normalMatrix \* P)
- normalize(dot(dFdx(P), dFdy(P)))



dot fa el conjunt...

La resposta correcta és: normalize(cross(dFdx(P), dFdy(P)))

**Pregunta 4**

No s'ha respuest

Puntuat sobre  
1,00Marca la  
pregunta

Amb les matrius indicades, calcula la distància al pla de retallat posterior (zfar):

$$\text{projectionMatrix} = \begin{bmatrix} 0.5 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.5 & 0.0 & 0.0 \\ 0.0 & 0.0 & \underbrace{\begin{pmatrix} -1.25 & 2.25 \end{pmatrix}}_{(x,y)} & \\ 0.0 & 0.0 & -1.0 & 0.0 \end{bmatrix}$$

$$x = \frac{-(zF + zN)}{(zF - zN)} \quad , \quad y = \frac{-(zF + zN)}{zF - zN}$$

$$zN = \frac{y}{x+1} = \frac{2.25}{-1.25-1} = -1.0$$

$$zF = \frac{y}{x+1} = \frac{2.25}{-1.25+1} = -9.0$$

$$\text{projectionMatrixInv} = \begin{bmatrix} 2.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 2.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & -1.0 \\ 0.0 & 0.0 & 0.4444 & -0.5556 \end{bmatrix}$$

*cal fer-se a veure absolut*

[Cast]

Resposta:  ×

La resposta correcta és: 9

## Pregunta 5

### Corrected

Puntuació 1,00  
sobre 1,00

 Marca la  
pregunta

Indica quin arbre de rajos pot ser generat per Ray Tracing clàssic: **LDS\***

## [Cast]

## Trieu-ne una:

- The diagram shows a search tree with the following structure:

  - Level 0: Root node E
  - Level 1: Two nodes S
  - Level 2: Four nodes (D, S, S, D)
  - Level 3: Eight nodes (D, S, S, D, S, D, S, S)

Red ovals highlight specific regions of the tree:

  - A red oval encloses the two S nodes at Level 1.
  - A red oval encloses the four nodes at Level 2: D, S, S, and D.
  - A red oval encloses the eight nodes at Level 3: D, S, S, D, S, D, S, and S.
  - A red oval encloses the two D nodes at Level 2.

S gener { S  
D no gener

## Pregunta 6

Correcte

Puntuació 1,00

sobre 1,00

Marca la pregunta

Indica quina tasca/opció pot fer que alguns fragments no segueixin processant-se:

[Cast]

Trieu-ne una:

- glClear *→ Neteja buffer, no desce la res*
- glColorMask *→ fragment es guarda però colors 10*
- depth test *✓ → si altres fragments el tapen*
- alpha Blending *→ modifica colors, no desce els fragments*

La resposta correcta és: depth test

## Pregunta 7

Incorrecte

Puntuació -0,33  
sobre 1,00

Desmarca  
aquesta  
pregunta

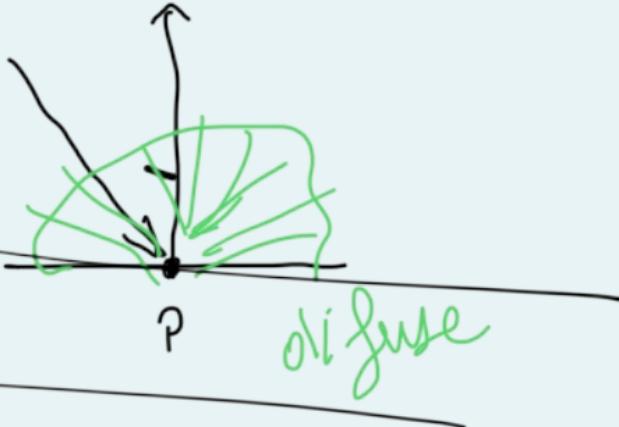
*igual cop a l'anell*  
Una superfície plana perfectament difosa rep una irradiància de  $8 \text{ W/m}^2$ . Si  $L$  és la radiància reflectida de sortida en direcció perpendicular a la superfície, i  $L'$  és la radiància reflectida de sortida a  $70$  graus de la normal, llavors...

[Cast]

Trieu-ne una:

- $L' = L / \cos(70)$  ✗
- $L = L'$
- $L' = L / \cos(20)$
- $L' = L * \cos(70)$

$$E = 8 \text{ W/m}^2$$



La resposta correcta és:  $L = L'$

## Pregunta 8

No s'ha respot

Puntuat sobre  
1,00

Marca la  
pregunta

Siguin

$P$  = punt visible de l'escena en una certa direcció de visió  $w$

$N$  = normal de la superfície en el punt  $P$

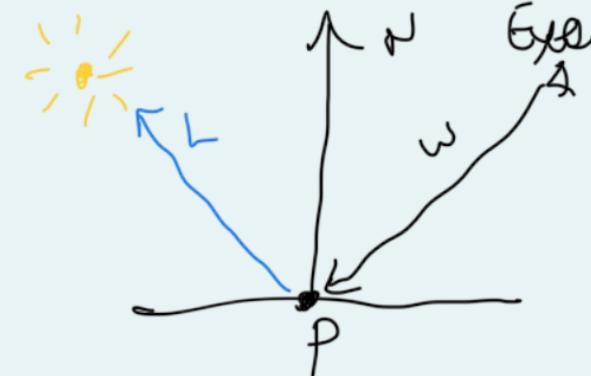
$L$  = vector unitari del punt  $P$  cap a la llum

En última instància, resoldre el problema de la il·luminació global, a nivell de cada punt de la imatge, equival a estimar...

[Cast]

Trieu-ne una:

- La irradiància de  $P$  en direcció  $w$
- $N \cdot L$
- La irradiància que arriba a  $P$
- La radiància de sortida al punt  $P$  en direcció  $-w$



La resposta correcta és: La radiància de sortida al punt  $P$  en direcció  $-w$

## Pregunta 9

Correcte

Puntuació 1,00  
sobre 1,00

Marca la pregunta

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

fragColor is written



FS

Rasterization



VS → FS

Call to glDrawArrays



antes de la VS

Stencil Test



després del fragment shader

La resposta correcta és: fragColor is written → 3, Rasterization → 2, Call to glDrawArrays → 1, Stencil Test → 4.

## Pregunta 10

Correcte

Puntuació 1,00  
sobre 1,00

Marka la  
pregunta

Diposem d'aquesta textura:



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

G

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

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

[Cast]

facal

Trieu-ne una:

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

La resposta correcta és: factor=vec2(0.1, 1.0); offset=vec2(0.0, 0.0);

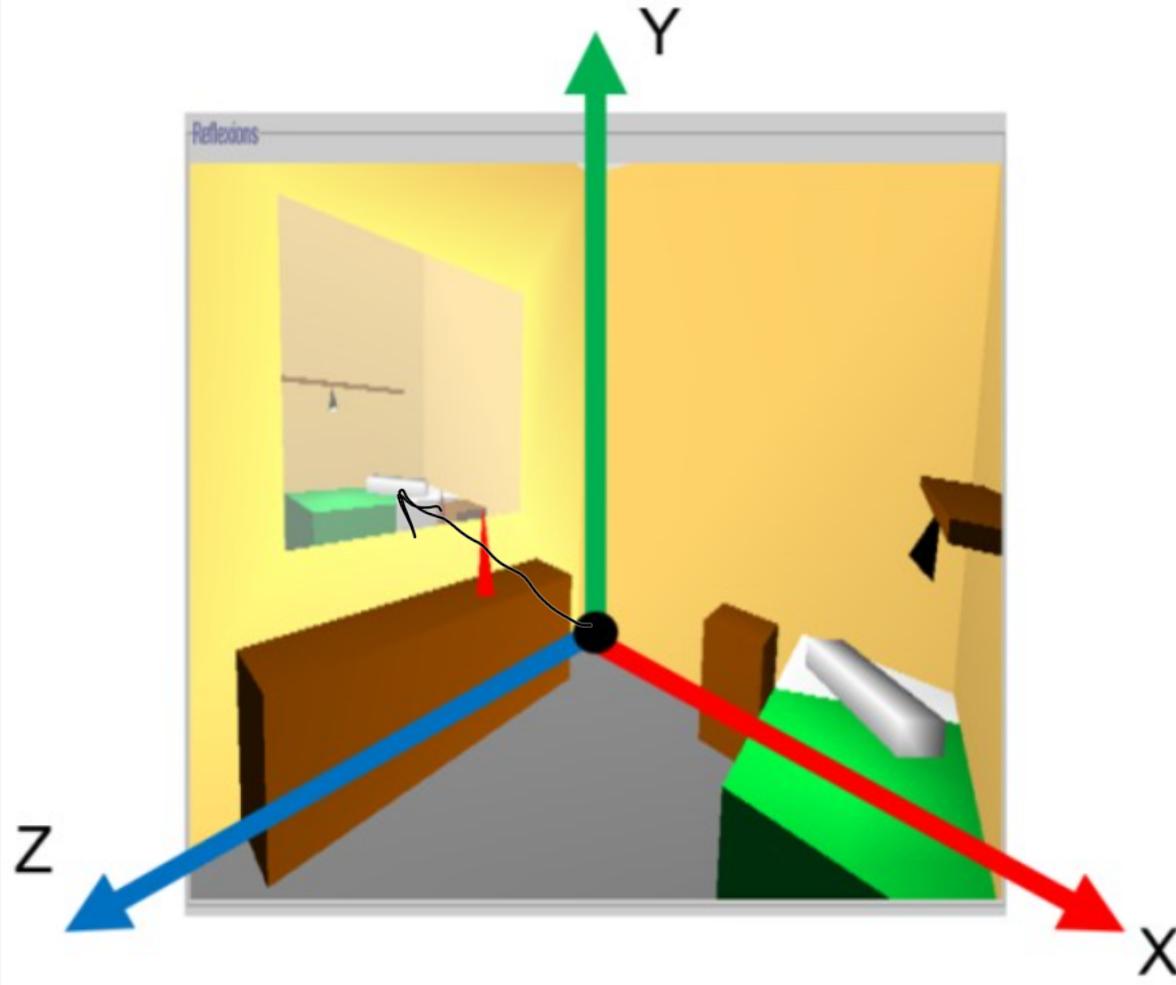
## Pregunta 11

Correcte

Puntuació 1,00  
sobre 1,00

Marca la pregunta

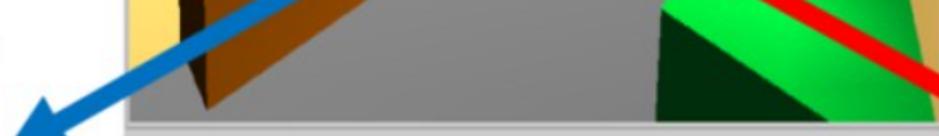
Considerant la figura



la matriu de reflexió per dibuixar l'escena reflectida al mirall és...

[Cast]

Z



X

la matriu de reflexió per dibuixar l'escena reflectida al mirall és...

[Cast]

Trieu-ne una:

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

~~X negatiu~~

La resposta correcta és:

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

## Pregunta 12

Correcte

Puntuació 1,00  
sobre 1,00

Marca la pregunta

Indica la matriu que s'utilitza a la tècnica de shadow mapping per obtenir les coordenades de textura (s,t,p,q) d'un vèrtex, **si el vèrtex es troba en object space.**

[Cast]

Trieu-ne una:

- T(0.5)\*S(0.5)
- T(0.5)\*S(0.5)\*P
- T(0.5)\*S(0.5)\*P\*V\*M
- S(0.5)\*T(0.5)\*P\*V\*M

X 03

*(Escalar, Rotar i Traslladar*

*(; el fe de  
metre a  
esquerre).*

La resposta correcta és: T(0.5)\*S(0.5)\*P\*V\*M

**Pregunta 13**

No s'ha respuest

Puntuat sobre  
1,00

▼ Desmarca  
aquesta  
pregunta

Considera un sistema de cinema 360° basat en una pantalla gegant de forma esfèrica, de radi  $R=18$ , les quals reben imatges projectades per un projector 360° situat al centre. Si el projector emet 3000 lumens, distribuïts de manera uniforme en totes direccions, indica la il·luminància resultant en la pantalla.

[Cast]

Resposta: 0,7368284902

$$A = 4\pi R^2$$

$$E = \frac{Q}{A} = \frac{3000}{4\pi(18)^2}$$

La resposta correcta és: 0,73682844024026

## Pregunta 14

Correcte

Puntuació 1,00  
sobre 1,00

◀ Marca la  
pregunta

Indica quina és l'opció més adient per a completar aquest codi a l'espai o espais indicats per '\_\_\_':

```
// draw a scene containing opaque and semitransparent objects
void X::paintGL()
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glEnable(GL_DEPTH_TEST);
    glDepthMask(GL_TRUE);
    opaque_objects.draw(); // unsorted
    glEnable(GL_BLEND);
    glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
    _____;
    semitransparent_objects.draw(); // unsorted
    glDisable(GL_BLEND);
}
```

[Cast]

Trieu-ne una:

- glDisable(GL\_BLEND)
- glDepthMask(GL\_FALSE) ✓
- glColorMask(GL\_FALSE)
- glDepthMask(GL\_TRUE)

*pels xin franyoreus*

La resposta correcta és: glDepthMask(GL\_FALSE)

## Pregunta 15

Incorrecte

Puntuació -0,33  
sobre 1,00

Marca la pregunta

Per un determinat pixel  $(x,y)$ , els valors al frame buffer són:  $\text{depthBuffer}[x,y]=0.5$ ,  $\text{stencilBuffer}[x,y]=4$ . El test s'ha configurat amb  $\text{glStencilTest(GL_ALWAYS, 6, 255)}$ . Si es genera un fragment per aquest pixel, amb  $\text{gl_FragCoord.xyz} = (x, y, 0.6)$ , indica quin serà el resultat final al stencilBuffer, si l'operació està configurada amb:  $\text{glStencilOp(GL_ZERO, GL_INCR, GL_REPLACE)}$ ;

[Cast]

- stencil pose (always)

- Buffer no ( $0.6 \neq 0.5$ )

$++h = 5$

Trieu-ne una:

- 6 X
- 5
- 4
- 0

La resposta correcta és: 5

## Pregunta 16

Incorrecte

Puntuació -0,33  
sobre 1,00

Marca la pregunta

El punt 3D que resulta d'aplicar la transformació representada per la matriu

(24.00, 8.00, 16.00, 4.00) és...

[Cast]

Trieu-ne una:

- (32.00, 24.00, 16.00)
- (16.00, 8.00, 24.00)
- (6.00, 8.00, 4.00)
- (24.00, 32.00, 16.00) ✖

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 24 \\ 8 \\ 16 \\ 4 \end{pmatrix} = \begin{pmatrix} 24 \\ 32 \\ 16 \\ 4 \end{pmatrix}$$

$$\left( \frac{24}{4}, \frac{32}{4}, \frac{16}{4} \right)$$

$$(6, 8, 4)$$

La resposta correcta és: (6.00, 8.00, 4.00)

**Pregunta 17**

No s'ha respot

Puntuat sobre

1,00

 Desmarca  
aquesta  
pregunta

Selecciona la única matriu de projecció (projectionMatrix) plausible per a una càmera perspectiva:

[Cast]

Trieu-ne una:

- $$\begin{bmatrix} 2.0 & 0 & 0 & 6.0 \\ 0 & 1.0 & 0 & 3.0 \\ 0 & 0 & 1.0 & 2.0 \\ 0 & 0 & 0 & 1.0 \end{bmatrix}$$
- $$\begin{bmatrix} 1.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.667 & 0.0 & 0.0 \\ 0.0 & 0.0 & -3.0 & -8.0 \\ 0.0 & 0.0 & -1.0 & 0.0 \end{bmatrix}$$
- $$\begin{bmatrix} 1.0 & 0 & 0 & 3.0 \\ 0 & 1.0 & 0 & 3.0 \\ 0 & 0 & 4.0 & 2.0 \\ 0 & 0 & 0 & 1.0 \end{bmatrix}$$
- $$\begin{bmatrix} 1.0 & 0 & 0 & 4.0 \\ 0 & 4.0 & 0 & 5.0 \\ 0 & 0 & 1.0 & 5.0 \\ 0 & 0 & 0 & 1.0 \end{bmatrix}$$

Perspectiva  $\begin{pmatrix} 0 & 0 & -1 & 0 \end{pmatrix}$   
Ortogonal  $\begin{pmatrix} 0 & 0 & 0 & 2 \end{pmatrix}$

La resposta correcta és:

$$\begin{bmatrix} 1.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.667 & 0.0 & 0.0 \\ 0.0 & 0.0 & -3.0 & -8.0 \\ 0.0 & 0.0 & -1.0 & 0.0 \end{bmatrix}$$

## Pregunta 18

Correcte

Puntuació 1,00  
sobre 1,00

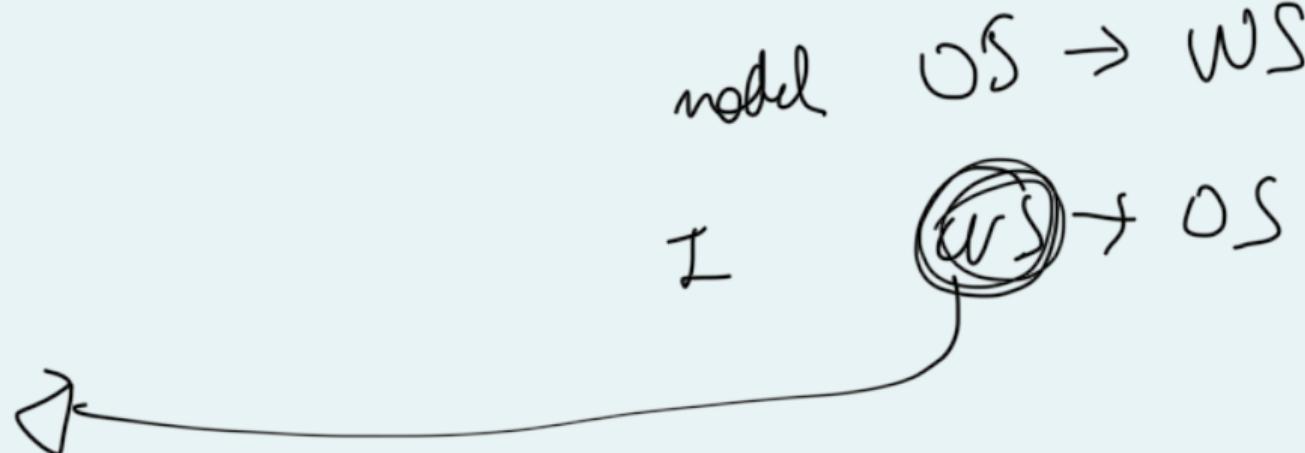
Marca la pregunta

Tria l'espai de coordenades en que ha d'estar P per tal que la transformació **modelMatrixInverse\*P** tingui sentit

[Cast]

Trieu-ne una:

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



La resposta correcta és: world space

## Pregunta 19

Correcte

Puntuació 1,00  
sobre 1,00

◀ Marca la pregunta

Donat aquest codi per generar reflexions en miralls,

```
// 1. Dibuixem el mirall a l'stencil buffer
glEnable(GL_STENCIL_TEST);
glStencilFunc(GL_ALWAYS, 1, mask);
glStencilOp(GL_KEEP, GL_KEEP, GL_REPLACE);
glDepthMask(GL_FALSE); glColorMask(GL_FALSE...);
dibuixar(mirall);

// 2. Dibuixem els objectes virtuals
glDepthMask(GL_TRUE); glColorMask(GL_TRUE...);
glStencilFunc(GL_EQUAL, 1, mask); 
glStencilOp(GL_KEEP, GL_KEEP, GL_KEEP);
glPushMatrix(); glMultMatrix(matriu simetria)
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);
```

un valor de la variable mask que preserva la correctesa del resultat és...

[Cast] 0 fa mal (reflexió) i 1 (síntesi de mirall)

Trieu-ne una:

- 0
- 3 ✓
- 256
- 2

La resposta correcta és: 3

Handwritten notes below the list:  
0 → false  
1 → true  
2 → false  
3 → true  
256 → false

## Pregunta 20

No s'ha respost

Puntuat sobre  
1,00

Marca la  
pregunta

Un estimador de la il·luminació global amb molta variància es distingeix per...

[Cast]

Trieu-ne una:

- Genera imatges que requereixen noise filtering
- Suporta una gran varietat de camins de llum, especialment especulars.
- Té problemes per simular camins de llum especulars
- Suporta una gran varietat de materials (difosos i especulars)

La resposta correcta és: Genera imatges que requereixen noise filtering

3) 22-23 Q2

CAMPUS VIRTUAL UPC / Les meves assignatures / G (CUTotal) - 2022/23-02:FIB - 270022 / General / Examen final 12 de juny 2023



**Començat el** dilluns, 12 de juny 2023, 11:32**Estat** Acabat**Completat el** dilluns, 12 de juny 2023, 12:32**Temps emprat** 59 minuts 56 segons**Punts** 12,00/20,00**Qualificació** 6,00 sobre 10,00 (60%)**Pregunta 1**

Correcte

Puntuació 1,00 sobre 1,00

Tria l'espai de coordenades en que ha d'estar P per tal que la transformació **modelViewMatrix**\*P tingui sentit

[Cast]

$$P(OBJ) \rightarrow CYE$$

Trieu-ne una:

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



La resposta correcta és: object space

**Pregunta 2**

Incorrecte

Puntuació -0,33 sobre 1,00

Indica quina expressió GLSL permet calcular el cosinus de l'angle incident (angle entre la normal i el light vector):

[Cast]

Trieu-ne una:

- dot(N, L)
- acos(N, L)
- cross(N, L)
- N\*L

$$\text{dot}(N, h)$$

↙ ↘



La resposta correcta és: dot(N, L)

**Pregunta 3**

Correcte

Puntuació 1,00 sobre 1,00

Indica la matriu que s'utilitza a la tècnica de shadow mapping per obtenir les coordenades de textura (s,t,p,q) d'un vèrtex, **si el vèrtex es troba en object space**.

[Cast]

Trieu-ne una:

- T(0.5)\*S(0.5)\*P
- T(0.5)\*S(0.5)\*P\*V\*M
- M\*P\*V
- T(0.5)\*S(0.5)

← clipping + scale + translate



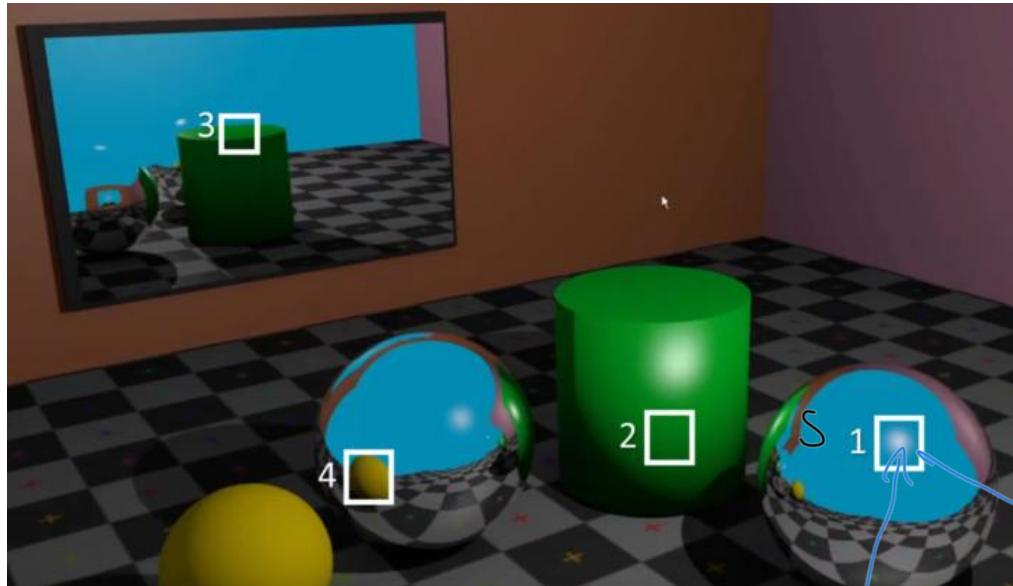
La resposta correcta és: T(0.5)\*S(0.5)\*P\*V\*M

## Pregunta 4

Correcte

Puntuació 1,00 sobre 1,00

El light path que explica el color dominant al pixel central del quadrat 1 és...



[Cast]

Trieu-ne una:

- LSE
- LDSE
- LDDE
- LDE



LSE

La resposta correcta és: LSE

## Pregunta 5

Correcte

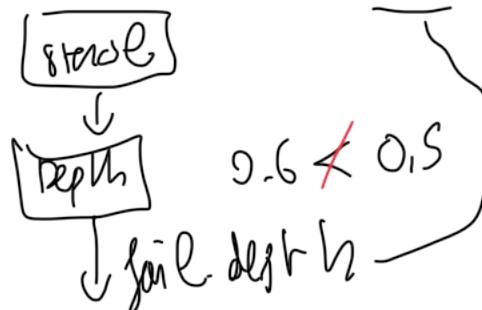
Puntuació 1,00 sobre 1,00

Per un determinat pixel  $(x,y)$ , els valors al frame buffer són:  $\text{depthBuffer}[x,y]=0.5$ ,  $\text{stencilBuffer}[x,y]=4$ . El test s'ha configurat amb  $\text{glStencilTest(GL_ALWAYS, 6, 255)}$ . Si es genera un fragment per aquest pixel, amb  $\text{gl_FragCoord.xyz} = (x, y, 0.6)$ , indica quin serà el resultat final al stencilBuffer, si l'operació està configurada amb:  $\text{glStencilOp(GL_DECR, GL_ZERO, GL_REPLACE)}$ :

[Cast]

Trieu-ne una:

- 6
- 0
- 3
- 4



La resposta correcta és: 0

## Pregunta 6

Correcte

Puntuació 1,00 sobre 1,00

Quin terme o factor que apareix a l'equació general del rendering és el que retorna, de forma molt aproximada, un shader que implementa il·luminació de Phong?

[Cast]

Trieu-ne una:

- $Le(x, wo, t)$
- $wi \cdot n$
- $Lo(x, wo, t)$
- $wi$



La resposta correcta és:  $Lo(x, wo, t)$



Pregunta 7

Incorrecte

Puntuació -0,33 sobre 1,00

Un FS rep una variable `in vec3 P` amb la posició del fragment en eye space. Per calcular el *light vector* L cal usar...

[Cast]

Trieu-ne una:

- `vec3 L = normalize((modelViewMatrix * lightPosition).xyz - P);`
- `vec3 L = normalize(P - lightPosition.xyz);`
- `vec3 L = lightPosition.xyz - P; L = normalize(L);`
- `vec3 L = lightPosition.xyz - P;`

Cal Normalitzar!



La resposta correcta és: `vec3 L = lightPosition.xyz - P; L = normalize(L);`

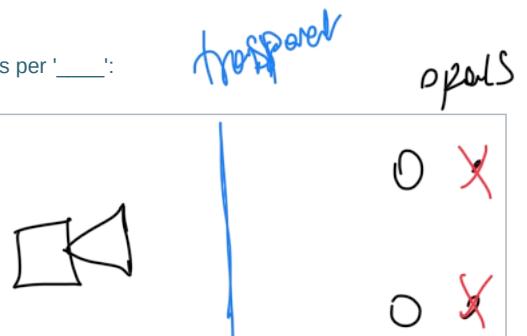
Pregunta 8

Incorrecte

Puntuació -0,33 sobre 1,00

Indica quina és l'opció més adient per a completar aquest codi a l'espai o espais indicats per '\_\_\_\_\_':

```
// draw a scene containing opaque and semitransparent objects
void X::paintGL()
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glEnable(GL_DEPTH_TEST);
    glDepthMask(GL_TRUE);
    opaque_objects.draw(); // unsorted
    glEnable(GL_BLEND);
    glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
    _____;
    semitransparent_objects.draw(); // unsorted
    glDisable(GL_BLEND);
}
```



- X
- X

[Cast]

Trieu-ne una:

- `glColorMask(GL_FALSE)`
- `glDisable(GL_BLEND)`
- `glDepthMask(GL_FALSE)`
- `glDepthMask(GL_TRUE)`

→ Volen evitar que el vídeo tapi els de

La resposta correcta és: glDepthMask(GL\_FALSE)

Pregunta 9

Correcte

Puntuació 1,00 sobre 1,00

Color non

Indica quina tasca/opció pot fer que alguns fragments no segueixin processant-se:

[Cast]

Trieu-ne una:

- glClearColor
- alpha Blending
- alpha test
- glClear

*→ desabilitar els fragments amb alpha < 0.5, per exemple*



La resposta correcta és: alpha test

Pregunta 10

No s'ha respost

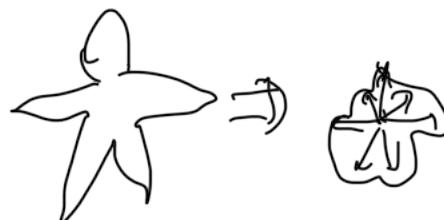
Puntuat sobre 1,00

Com a part d'un procés de generació de coordenades de textura, volem projectar un model centrat en el punt C sobre una esfera unitària. El mètode més adient és...

[Cast]

Trieu-ne una:

- vec3 P = (modelViewProjectionMatrix \* vertex).xyz - C;
- vec3 P = (modelViewProjectionMatrix \* vertex).xyz;
- vec3 P = vertex.xyz - C;
- vec3 P = vertex.xyz - C; P = normalize(P);



La resposta correcta és: vec3 P = vertex.xyz - C; P = normalize(P);

Pregunta 11

No s'ha respost

Puntuat sobre 1,00

Amb les matrius indicades, calcula la distància al pla de retallat posterior (zfar):

$$\text{projectionMatrix} = \begin{bmatrix} 0.3333 & 0 & 0 & 0 \\ 0 & 0.3333 & 0 & 0 \\ 0 & 0 & -1.2222 & 2.2222 \\ 0 & 0 & -1.0 & 0 \end{bmatrix} \rightarrow z^F = \frac{y}{x+1} = \frac{2.2222}{0.2222} = 10$$

$$\text{projectionMatrixInv} = \begin{bmatrix} 3.0 & 0 & 0 & 0 \\ 0 & 3.0 & 0 & 0 \\ 0 & 0 & 0 & -1.0 \\ 0 & 0 & 0.45 & -0.55 \end{bmatrix}$$

[Cast]

Resposta: 10



La resposta correcta és: 10

Pregunta 12

Correcte

Puntuació 1,00 sobre 1,00

Indica el valor que retorna aquesta expressió GLSL:

`dot(vec2(9, 7), vec2(3,8))`



$$9 \cdot 3 + 7 \cdot 8 = 27 + 56 = 83$$

[Cast]

Resposta: 83



La resposta correcta és: 83

Pregunta 13

Correcte

Puntuació 1,00 sobre 1,00

Tenim una escena tancada que conté 81 objectes difosos i 2 llums puntuals. Volem generar una imatge de 1024 x 480 pixels amb Ray Tracing clàssic. Quants **shadow rays** cal llançar? (indica la resposta amb un enter)

[Cast]

Resposta: 983040



$$1024 \cdot 480 \cdot 2 = 983040$$

La resposta correcta és: 983040

Pregunta 14

Correcte

Puntuació 1,00 sobre 1,00

El punt amb coordenades homogènies (24.00, 12.00, 20.00, 4.00) correspon al punt 3D...

[Cast]

Trieu-ne una:

- (6.00, 3.00, 5.00)
- (-24.00, -12.00, -20.00)
- (3.00, 6.00, 5.00)
- (24.00, 12.00, 20.00)

$$\left( \frac{24}{4}, \frac{12}{4}, \frac{20}{4} \right) = (6, 3, 5)$$

La resposta correcta és: (6.00, 3.00, 5.00)

Pregunta 15

Correcte

Puntuació 1,00 sobre 1,00



Assigna a cada ciutat les seves tarefas relativa (1,2,3,4) en què s'executa en un pipeline d'OpenGL sense GS.

[Cast]

Depth test	3	✓
Primitive assembly	1	✓
Backface culling	2	✓
Alpha blending	4	✓

apunts es seu

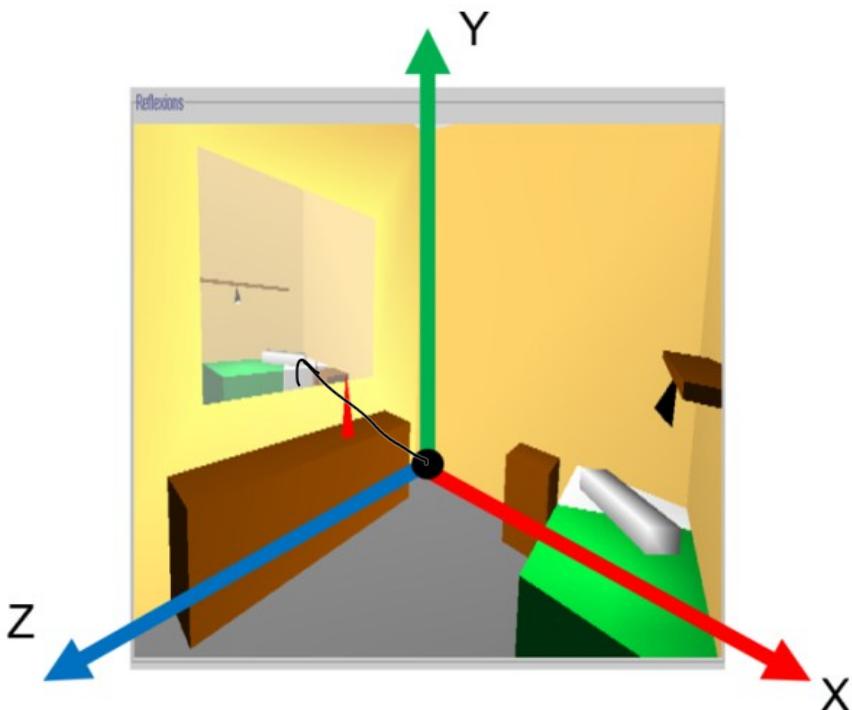
La resposta correcta és: Depth test → 3, Primitive assembly → 1, Backface culling → 2, Alpha blending → 4.

## Pregunta 16

Correcte

Puntuació 1,00 sobre 1,00

Considerant la figura



la matriu de reflexió per dibuixar l'escena reflectida al mirall és...

[Cast]

Trieu-ne una:

- $\begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$  }(-x)
- $\begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
- $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
- $\begin{bmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$

✓



[0 0 0 1]

La resposta correcta és:

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

**Pregunta 17**

No s'ha respot

Puntuació sobre 1,00

Amb les matrius indicades, la posició de l'observador en model space és:

$$\text{modelMatrixInv} = \begin{bmatrix} 1.0 & 0 & 0 & -1.0 \\ 0 & 1.0 & 0 & -2.0 \\ 0 & 0 & 1.0 & -3.0 \\ 0 & 0 & 0 & 1.0 \end{bmatrix}$$

$$\text{viewMatrixInv} = \begin{bmatrix} -0.95 & -0.17 & 0.27 & 1.0 \\ 0 & 0.85 & 0.53 & 2.0 \\ -0.32 & 0.51 & -0.8 & -3.0 \\ 0 & 0 & 0 & 1.0 \end{bmatrix}$$

$$\text{modelViewMatrixInv} = \begin{bmatrix} -0.95 & -0.17 & 0.27 & 0 \\ 0 & 0.85 & 0.53 & 0 \\ -0.32 & 0.51 & -0.8 & -6.0 \\ 0 & 0 & 0 & 1.0 \end{bmatrix}$$

*(sota la correa a eye space a)*

$$\begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ -6 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ -6 \\ 1 \end{pmatrix}$$

[Cast]

Trieu-ne una:

- (0.00, 0.00, 0.00)
- (1.00, 2.00, 3.00)
- (-1.00, -2.00, -3.00)
- (0.00, 0.00, -6.00)

La resposta correcta és: (0.00, 0.00, -6.00)

**Pregunta 18**

Correcte

Puntuació 1,00 sobre 1,00

Diposem d'aquesta textura:

**G H I J K L M N O P**

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

**J**

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

$$\times (0.1) + 0.3$$

*(página següent)*

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

[Cast]

Trieu-ne una:

- factor=vec2(0.0, 0.3); offset=vec2(0.3, 0.1);
- factor=vec2(0.0, 1.0); offset=vec2(1.0, 0.1);
- factor=vec2(1.0, 1.0); offset=vec2(3.0, 3.0);



`factor=vec2(0.1, 1.0); offset=vec2(0.3, 0.0);`



La resposta correcta és: `factor=vec2(0.1, 1.0); offset=vec2(0.3, 0.0);`

### Pregunta 19

Correcte

Puntuació 1,00 sobre 1,00

L'expressió GLSL que representa l'expressió matemàtica  $K_d I_d(N \cdot L)$  és:

[Cast]

Trieu-ne una:

- `matDiffuse * lightDiffuse * cross(N,L)`
- `matDiffuse * lightDiffuse * normalize(N) * normalize(L)`
- `matDiffuse * lightDiffuse * normalize(N) * L`
- `matDiffuse * lightDiffuse * dot(N,L)`

*(- ja n'hi ha)*



La resposta correcta és: `matDiffuse * lightDiffuse * dot(N,L)`

### Pregunta 20

No s'ha respot

Puntuat sobre 1,00

El punt 3D que resulta d'aplicar la transformació representada per la matriu

$$\begin{bmatrix} 1 & 0 & 0 & 5 \\ 0 & 1 & 0 & 5 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

al punt (12.00, 12.00, 12.00, 4.00) és...

[Cast]

Trieu-ne una:

- (3.00, 3.00, 12.00)
- (8.00, 8.00, 5.00)
- (12.00, 12.00, 12.00)
- (32.00, 32.00, 20.00)

La resposta correcta és: (8.00, 8.00, 5.00)

◀ Missatges del professorat

Salta a...

► Qüestionari competències transversals ►



4) 22-23 Q1

CAMPUS VIRTUAL UPC / Les meves assignatures / G (CUTotal) - 2022/23-01:FIB-270022 / General / Examen final G (20 de gener 2023)

**Començat el** divendres, 20 de gener 2023, 11:35**Estat** Acabat**Completat el** divendres, 20 de gener 2023, 12:27**Temps emprat** 52 minuts 2 segons**Punts** 17,67/20,00**Qualificació** 8,83 sobre 10,00 (88%)

## Pregunta 1

Correcte

Puntuació 1,00 sobre 1,00

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

[Cast]

Backface culling	4	✓
gl_Position is written	2	✓
VS execution starts	1	✓
Clipping to viewing frustum	3	✓

fàcil

La resposta correcta és: Backface culling → 4, gl\_Position is written → 2, VS execution starts → 1, Clipping to viewing frustum → 3.

## Pregunta 2

Correcte

Puntuació 1,00 sobre 1,00

Quin terme o factor que apareix a l'equació general del rendering és el que retorna, de forma molt aproximada, un shader que implementa il·luminació de Phong?

[Cast]

Trieu-ne una:

- $L_i(x, w_i, t)$
- $w_i \cdot n$
- $w_i$
- $L_o(x, w_o, t)$

( ja visto )

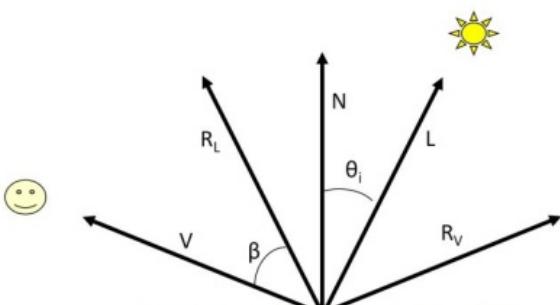
La resposta correcta és:  $L_o(x, w_o, t)$ 

## Pregunta 3

Correcte

Puntuació 1,00 sobre 1,00

La figura mostra diferents vectors unitaris que intervenen en els càlculs d'il·luminació.





Quins vectors corresponen a direccions de rajos que traçaria two-pass raytracing, durant el *light pass*? (tria la més completa de les correctes) [Cast]

Trieu-ne una:

- $-L, R_V, T_V$
- $-L, R_V$
- $-L, R_L$
- $R_L, R_V$



La resposta correcta és:  $-L, R_L$

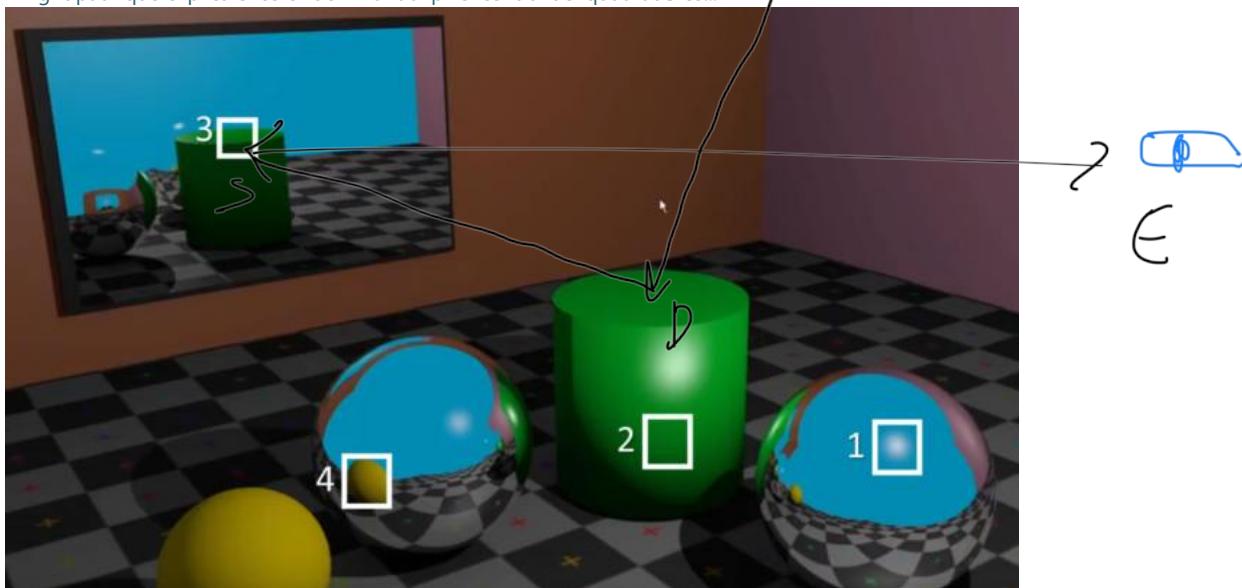
#### Pregunta 4

Correcte

Puntuació 1,00 sobre 1,00



El light path que explica el color dominant al pixel central del quadrat 3 és...



[Cast]

Trieu-ne una:

- LSDE
- LDSE
- LSDSE
- LSE



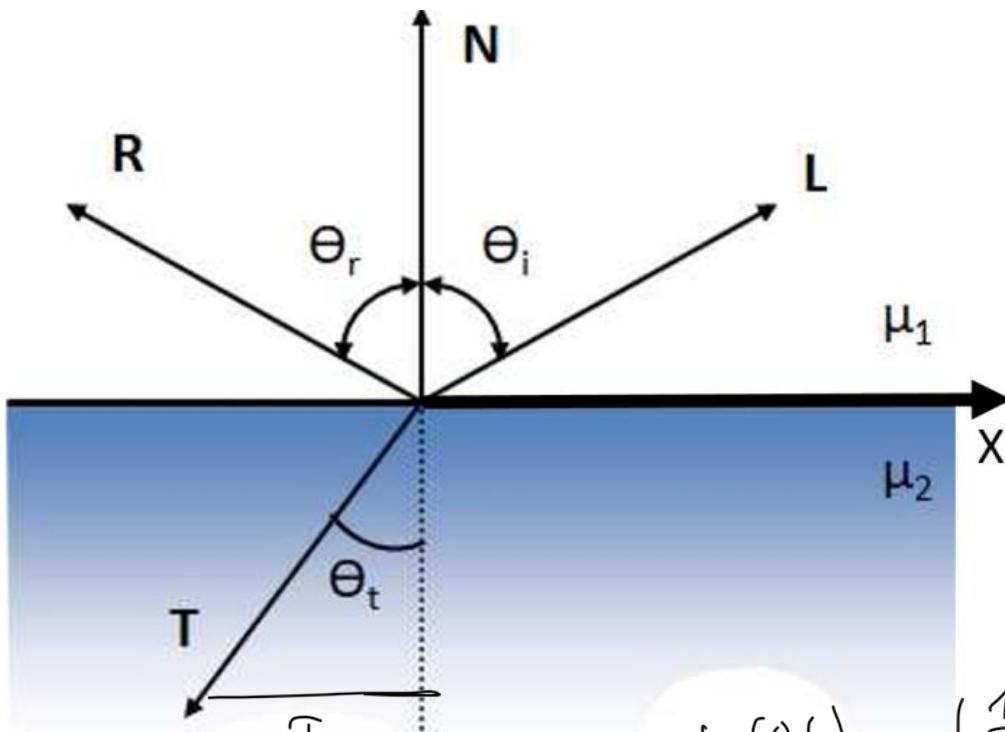
La resposta correcta és: LDSE

#### Pregunta 5

Incorrecte

Puntuació 0,00 sobre 1,00

Assuming que  $N = (0, 1, 0)$  i  $L = (0.44, 0.90, 0.00)$ , i que els dos medis tenen índexs de refracció 1.30 i 2.50, calcula (amb el signe correcte) la component X del vector unitari trasmès T.



[Cast]

Resposta: 0,2288

$$T_x = -\sin(\theta_t) = -\left(\frac{1.3}{2.5} \cdot \sin(60^\circ)\right)$$

$$= -0,2266627451$$

La resposta correcta és: -0,22666274506411

## Pregunta 6

Correcte

Puntuació 1,00 sobre 1,00

Indica el valor de u (enter) que fa correcte aquest codi:

```
QImage img4("file.png");
QImage T5 = img4.convertToFormat(QImage::Format_ARGB32);
 glGenTextures(1, &textureId6);
 glBindTexture(GL_TEXTURE_2D, textureId6);
 glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, T5.width(), T5.height(), 0, GL_RGBA, GL_UNSIGNED_BYTE, T5.bits());
 glActiveTexture(GL_TEXTURE0);
 glBindTexture(GL_TEXTURE_2D, textureId6);
 program->bind();
 program->setUniformValue("textureMap", lu); // sampler2D
```

[Cast]

Resposta: 8

La resposta correcta és: 8

## Pregunta 7

Correcte

Puntuació 1,00 sobre 1,00

El punt 3D que resulta d'aplicar la transformació representada per la matriu

$$\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

al punt (18.00, 6.00, 18.00, 3.00) és...

$$\begin{bmatrix} 0 & 0 & 1 & 4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

[Cast]

Trieu-ne una:

- (6.00, 2.00, 18.00)
- (7.00, 5.00, 10.00)
- (15.00, 21.00, 30.00)
- (21.00, 15.00, 30.00)

multiplicació de vector-matris  
ja ~~est~~ ...



La resposta correcta és: (7.00, 5.00, 10.00)

Pregunta 8

Correcte

Puntuació 1,00 sobre 1,00

Indica quin arbre de rajos pot ser generat per Ray Tracing clàssic:

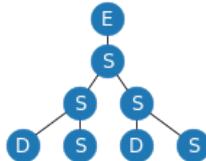
[Cast]

Trieu-ne una:

- 
- 
- 
- 



La resposta correcta és:



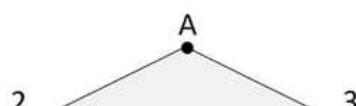
.

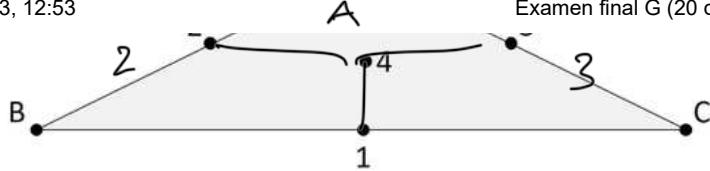
Pregunta 9

Incorrecte

Puntuació -0,33 sobre 1,00

La figura mostra un triangle definit pels punts A, B, C.





Indica les coordenades baricèntriques (alfa, beta, gamma) del punt indicat amb un 4.

[Cast]

Trieu-ne una:

- (1/2, 0, 1/2)
- (1/2, 1/2, 1/2)
- (1/3, 1/3, 1/3)
- (1/2, 1/2, 0)

És un triangle "simètric", tots els costats tenen el mateix pes

✗

La resposta correcta és: (1/3, 1/3, 1/3)

Pregunta 10

Correcte

Puntuació 1,00 sobre 1,00

En quin espai no és correcte interpolar simplement ( $s, t$ ) quan usen una càmera perspectiva?

[Cast]

Trieu-ne una:

- object space
- world space
- clip space
- NDC

[j]a que divideix per  $w$  no té sentit

✓

La resposta correcta és: NDC

Pregunta 11

Correcte

Puntuació 1,00 sobre 1,00

Indica quina és l'opció més adient per a completar aquest codi a l'espai o espais indicats per '\_\_\_':

```
// draw a scene containing opaque and semitransparent objects
void X::paintGL()
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glEnable(GL_DEPTH_TEST);
    glDepthMask(GL_TRUE);
    opaque_objects.draw(); // unsorted
    glEnable(GL_BLEND);
    glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
    _____;
    semitransparent_objects.draw(); // unsorted
    glDisable(GL_BLEND);
}
```

la de sempre.

[Cast]

Trieu-ne una:

- glDisable(GL\_DEPTH\_TEST)
- glDepthMask(GL\_FALSE)

✓

- glDisable(GL\_BLEND)
- glEnable(GL\_SUB)

La resposta correcta és: glEnable(GL\_DEPTH)

### Pregunta 12

Correcte

Puntuació 1,00 sobre 1,00

Tenim una escena tancada que conté 49 objectes difosos i 3 llums puntuals. Volem generar una imatge de 640 x 1080 pixels amb Ray Tracing clàssic. Quants **shadow rays** cal llançar? (indica la resposta amb un enter)

[Cast]

$$3 \cdot 640 \cdot 1080 = 2073600$$

Resposta: 2073600 ✓

La resposta correcta és: 2073600

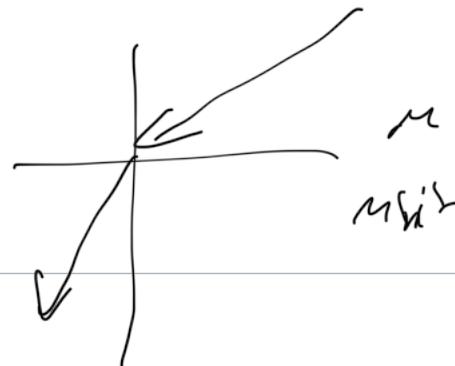
### Pregunta 13

Correcte

Puntuació 1,00 sobre 1,00

Indica quina és l'opció més adient per a completar aquest codi a l'espai o espais indicats per '\_\_\_':

```
vec4 trace(Ray ray, float mu) {
    if ( intersectScene(ray, Phit, Nhit, muHit, Kd, Kr, Kt) ) {
        Ray shadowRay = Ray(Phit, normalize(lightPos-Phit));
        bool inShadow = intersectScene(shadowRay, aux);
        if (inShadow) shadow = 0.2; else shadow = 1.0;
        color+= shadow*light(Nhit, -ray.dir, lightPos-Phit, Kd, vec4(1.0));
        vec3 R = reflect(ray.dir, Nhit);
        color += Kr*trace(Ray(Phit, R), mu);
        vec3 T = refract(ray.dir, Nhit, _____);
        if (length(T)>0.0) color+=Kt*trace(Ray(Phit, T), muHit);
    }
    else color+=samplePanorama(ray.dir);
    return color;
}
```



[Cast]

Trieu-ne una:

- mu
- mu\*muHit
- mu/muHit
- muHit/mu



La resposta correcta és: mu/muHit

### Pregunta 14

Correcte

Puntuació 1,00 sobre 1,00

Per un determinat pixel (x,y), els valors al frame buffer són: depthBuffer[x,y]=0.5, stencilBuffer[x,y]=4. El test s'ha configurat amb

<https://atenea.upc.edu/mod/quiz/review.php?attempt=4921502&cmid=3802049>

glStencilTest(GL\_ALWAYS, 6, 255). Si es genera un fragment per aquest pixel, amb gl\_FragCoord.xyz = (x, y, 0.6), indica quin serà el resultat final al stencilBuffer, si l'operació està configurada amb: glStencilOp(GL\_DECR, GL\_ZERO, GL\_REPLACE);

[Cast]

Trieu-ne una:

- 0
- 6
- 4
- 3



La resposta correcta és: 0

Pregunta 15

Correcte

Puntuació 1,00 sobre 1,00

Siguin:

M: submatriu 3x3 de la modelMatrix

V: submatriu 3x3 de la viewMatrix,

P: submatriu 3x3 de la projectionMatrix,

la matriu que ens permet transformar **normals** de **object space** a **world space** es pot calcular com...

[Cast]

Trieu-ne una:

- $(MV)^{-1}$
- $M^{-T}$
- $M^T$
- $(MV)^{-T}$



La resposta correcta és:  $M^{-T}$

Pregunta 16

Correcte

Puntuació 1,00 sobre 1,00

L'expressió GLSL que representa l'expressió matemàtica  $K_d I_d(N \cdot L)$  és:

[Cast]

Trieu-ne una:

- matDiffuse \* lightDiffuse \* normalize(N) \* normalize(L)
- matDiffuse \* lightDiffuse \* dot(N,L)
- matDiffuse \* lightDiffuse \* cross(N,L)
- matDiffuse \* lightDiffuse \* N \* L



La resposta correcta és: matDiffuse \* lightDiffuse \* dot(N,L)

Pregunta 17

Correcte

Puntuació 1,00 sobre 1,00

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)
```

[Cast]

Trieu-ne una:

- `factor=vec2(0.1, 0.0); offset=vec2(1.0, 1.0);`
- `factor=vec2(0.1, 0.0); offset=vec2(0.0, 1.0);`
- `factor=vec2(0.1, 1.0); offset=vec2(0.4, 0.0);` → *l'objecte*
- `factor=vec2(1.0, 0.4); offset=vec2(0.4, 4.0);`



La resposta correcta és: `factor=vec2(0.1, 1.0); offset=vec2(0.4, 0.0);`

Pregunta 18

Correcte

Puntuació 1,00 sobre 1,00

Tria l'espai de coordenades en que ha d'estar  $P$  per tal que la transformació `modelViewMatrix * P` tingui sentit

[Cast]

Trieu-ne una:

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

*P (està en ES)*



La resposta correcta és: object space

Pregunta 19

Correcte

Puntuació 1,00 sobre 1,00

Soposa que  $P$  és un punt en eye space. Una forma d'obtenir en un FS la direcció del vector normal en eye space és...

[Cast]

Trieu-ne una:

- `normalize(cross(dFdx(P), dFdy(P)))`
- `dot(dFdx(P), dFdy(P))`
- `normalMatrix * P`
- `normalize(dot(dFdx(P), dFdy(P)))`



La resposta correcta és: `normalize(cross(dFdx(P), dFdy(P)))`

Pregunta 20

<https://atenea.upc.edu/mod/quiz/review.php?attempt=4921502&cmid=3802049>

Correcte

Puntuació 1,00 sobre 1,00

Indica la matriu que s'utilitza a la tècnica de shadow mapping per obtenir les coordenades de textura (s,t,p,q) d'un vèrtex, **si el vèrtex es troba en object space.**

[Cast]

Trieu-ne una:

- S(0.5)\*T(0.5)\*P
- T(0.5)\*S(0.5)\*P\*V\*M
- T(0.5)\*S(0.5)\*P\*V
- M\*P\*V

obj → clipping + scale + translate



La resposta correcta és: T(0.5)\*S(0.5)\*P\*V\*M

◀ Qüestionari Gràfics 11 octubre 2022

Salta a...

Qüestionari competències transversals Gener 2023 ►

5) 21-22 Q2

**Començat el** dimecres, 15 de juny 2022, 15:05

**Estat** Acabat

**Completat el** dimecres, 15 de juny 2022, 16:05

**Temps emprat** 59 minuts 53 segons

**Punts** 4,25/20,00

**Qualificació** 2,13 sobre 10,00 (21%)

Pregunta 1

Incorrecte

Puntuació -0,33 sobre 1,00

Per un determinat pixel (x,y), els valors al frame buffer són: depthBuffer[x,y]=0.5, stencilBuffer[x,y]=4. El test s'ha configurat amb glStencilTest(GL\_ALWAYS, 6, 255). Si es genera un fragment per aquest pixel, amb glFragCoord.xyz = (x, y, 0.6), indica quin serà el resultat final al stencilBuffer, si l'operació està configurada amb: glStencilOp(GL\_ZERO, GL\_INCR, GL\_REPLACE);

[Cast]

Trieu-ne una:

- 3
- 0
- 5
- 4
- No vull contestar la pregunta

① stencil  
② depth 0.6 < 0.5 → 4 + 1 = 5

La resposta correcta és: 5

Pregunta 2

Incorrecte

Puntuació 0,00 sobre 1,00

Tenim una escena tancada que conté 66 objectes difosos i 5 llums puntuals. Volem generar una imatge de 1024 x 768 pixels amb Ray Tracing clàssic. Quants **shadow rays** cal llançar? (indica la resposta amb un enter)

[Cast]

Resposta: 789432

✗

pixels · llum

$$1024 \cdot 768 \cdot 5 = \underline{\underline{3932160}}$$

La resposta correcta és: 3932160

Pregunta 3

Parcialment correcte

Puntuació 0,25 sobre 1,00

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

[Cast]

VS execution starts	1	x
Clipping to viewing frustum	2	x
glBufferData	3	x 2
Depth test	4	✓ 3

1

La resposta correcta és: VS execution starts → 1, Clipping to viewing frustum → 2, glBufferData → 3, Depth test → 4.

Pregunta 4

Correcte

Puntuació 1,00 sobre 1,00

Diposem d'aquesta textura:



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



10

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

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

✓ 10 + 0,5

[Cast]

Trieu-ne una:

- factor=vec2(1.0, 5.0); offset=vec2(0.1, 5.0);
- factor=vec2(0.5, 0.1); offset=vec2(5.0, 0.5);
- factor=vec2(0.1, 1.0); offset=vec2(0.5, 0.0); ✓
- factor=vec2(0.5, 0.5); offset=vec2(5.0, 1.0);
- No vull contestar la pregunta

(0.1 .. 0.10) + 0.5

La resposta correcta és: factor=vec2(0.1, 1.0); offset=vec2(0.5, 0.0);

Pregunta 5

Correcte

Puntuació 1,00 sobre 1,00

Diposem d'aquesta textura:

DISPOSEM d'aquesta textura.



(6,3)

Volem texturar un polígon rectangular situat sobre el pla Z = 0. Sabem que el seu vèrtex mínim té coordenades (0,0,0), i el vèrtex màxim té coordenades (4, 3, 0). Si usem dos plans (S,T) per a generar les coordenades de textura, indica l'opció que permet texturar el polígon així (ignora la relació d'aspecte):



[Cast]

Trieu-ne una:

- S=vec4(0.00, 0.75, 1.00, 0.00); T=vec4(3.00, 1.00, 0.25, 0.00); 3
- S=vec4(4.00, 0.75, 1.33, 0.00); T=vec4(1.00, 0.33, 1.00, 0.00);
- S=vec4(0.75, 0.00, 0.00, 0.00); T=vec4(0.00, 1.33, 0.00, 0.00); ✓
- S=vec4(0.75, 3.00, 0.00, 0.00); T=vec4(4.00, 1.00, 0.25, 0.00);
- No vull contestar la pregunta

La resposta correcta és: S=vec4(0.75, 0.00, 0.00, 0.00); T=vec4(0.00, 1.33, 0.00, 0.00);

Pregunta 6

Incorrecte

Puntuació -0,33 sobre 1,00

vfdn

0 ✗ no es veuen



Indica quina és l'opció més adient per a completar aquest codi a l'espai o espais indicats per '\_\_\_\_\_':

```
// draw a scene containing opaque and semitransparent objects
void X::paintGL()
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glEnable(GL_DEPTH_TEST);
    glDepthMask(GL_TRUE);
    opaque_objects.draw(); // unsorted
    glEnable(GL_BLEND);
    glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
    _____;
    semitransparent_objects.draw(); // unsorted
    glDisable(GL_BLEND);
}
```

glDepthMask(GL\_FALSE);

→ Volen evitar l'objecte transparent  
tapi els altres objectes, manten  
el depth test (més fàcil  
test anterior)

[Cast]

Trieu-ne una:

- glBlendComb(GL\_SUB)
- glDepthMask(GL\_TRUE)
- glColorMask(GL\_FALSE)
- glDepthMask(GL\_FALSE)
- No vull contestar la pregunta



La resposta correcta és: glDepthMask(GL\_FALSE)



Pregunta 7

Incorrecte

Puntuació -0,33 sobre 1,00

El punt 3D que resulta d'aplicar la transformació representada per la matriu

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

al punt (30.00, 25.00, 25.00, 5.00) és...

[Cast]

Trieu-ne una:

- (6.00, 5.00, 10.00)
- (6.00, 5.00, 25.00)
- (30.00, 25.00, 25.00)
- No vull contestar la pregunta
- (30.00, 25.00, 50.00)



$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 30 \\ 25 \\ 25 \\ 5 \end{pmatrix} = \begin{pmatrix} 30 \\ 25 \\ 50 \\ 5 \end{pmatrix} = \begin{pmatrix} 6 \\ 5 \\ 10 \\ 1 \end{pmatrix}$$



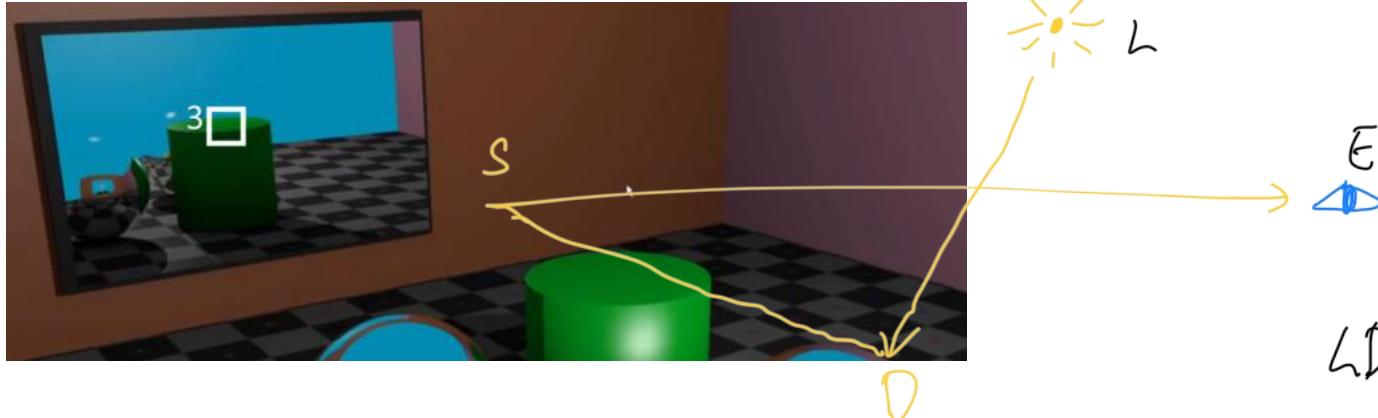
La resposta correcta és: (6.00, 5.00, 10.00)

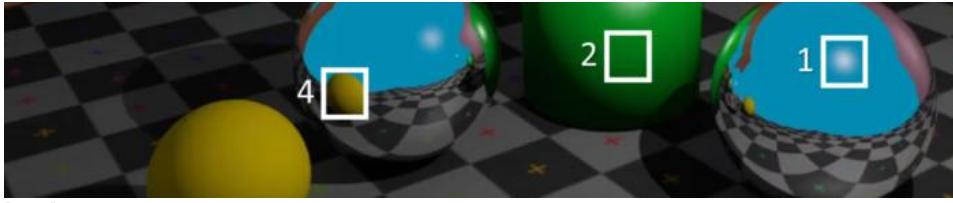
Pregunta 8

No s'ha respondt

Puntuat sobre 1,00

El light path que explica el color dominant al pixel central del quadrat 3 és...





[Cast]

Trieu-ne una:

- LDSE
- LSDE
- LSSE
- No vull contestar la pregunta
- LSDSE

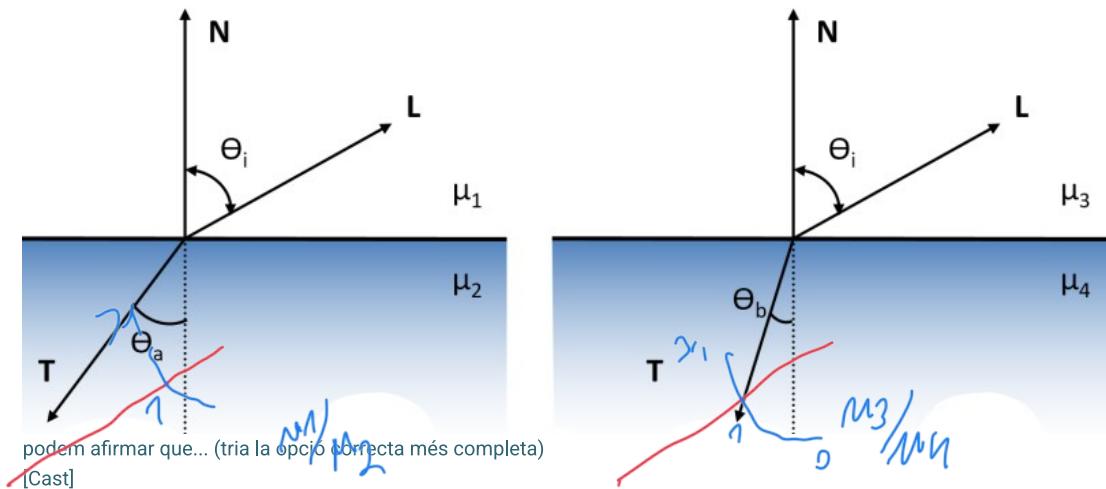
La resposta correcta és: LDSE

Pregunta 9

No s'ha respost

Puntuat sobre 1,00

Considerant la figura



[Cast]

Trieu-ne una:

- $(\mu_1 < \mu_2) \wedge (\mu_3 < \mu_4) \wedge (\mu_3/\mu_4 > \mu_1/\mu_2)$
- No vull contestar la pregunta
- $(\mu_1 < \mu_2) \wedge (\mu_3 < \mu_4) \wedge (\mu_2 > \mu_4)$
- $(\mu_1 < \mu_2) \wedge (\mu_3 < \mu_4) \wedge (\mu_1 > \mu_3)$
- $(\mu_2 > 0) \wedge (\mu_4 > 0) \wedge (\mu_1/\mu_2 > \mu_3/\mu_4)$

La resposta correcta és:  $(\mu_2 > 0) \wedge (\mu_4 > 0) \wedge (\mu_1/\mu_2 > \mu_3/\mu_4)$

Pregunta 10

Incorrecte

Puntuació 0,00 sobre 1,00

Indica el valor que retorna aquesta expressió GLSL:

`cross(vec3(3, 4, 2), vec3(2,2,4)).x`

$$\begin{vmatrix} i & j & k \\ a & b & c \\ d & e & f \end{vmatrix}$$

$$\begin{cases} x = bf - ce \\ y = af - cd \\ z = ae - bd \end{cases}$$

$$\rightarrow \text{producte vectorial} = \begin{vmatrix} i & j & k \\ 3 & 4 & 2 \\ 2 & 2 & 4 \end{vmatrix} \rightarrow x = 4 \cdot 2 - 2 \cdot 2 = 16 - 4 = 12$$

[Cast]

Resposta:  X

La resposta correcta és: 12

Pregunta 11

Incorrecte

Puntuació -0,33 sobre 1,00

Quin terme o factor que apareix a l'equació general del rendering és el que retorna, de forma molt aproximada, un shader que implementa il·luminació de Phong?

[Cast]

Trieu-ne una:

- $L_i(x, w_i, t)$
- $L_o(x, w_o, t)$
- $L_e(x, w_o, t)$
- No vull contestar la pregunta
- $\bullet r(x, w_i, w_o, t)$

$$\underline{d_o(\rho, w_o)} = d_e(\rho, w_o) + \sum_{\rho} S(\rho, w_o, w_o) \cdot L_i(\rho, w_i, w_o, t)$$

X

La resposta correcta és: Lo(x,wo,t)

Pregunta 12

Correcte

Puntuació 1,00 sobre 1,00

Indica el tipus de la següent expressió (en el context dels shaders del laboratori): `mix(texCoord.s, texCoord.t, 0.5)`

[Cast]

Trieu-ne una:

- $\text{vec2}$
- $\text{float}$
- $\text{mat3}$
- No vull contestar la pregunta
- $\text{vec4}$

✓

La resposta correcta és: float

Pregunta 13

No s'ha respot

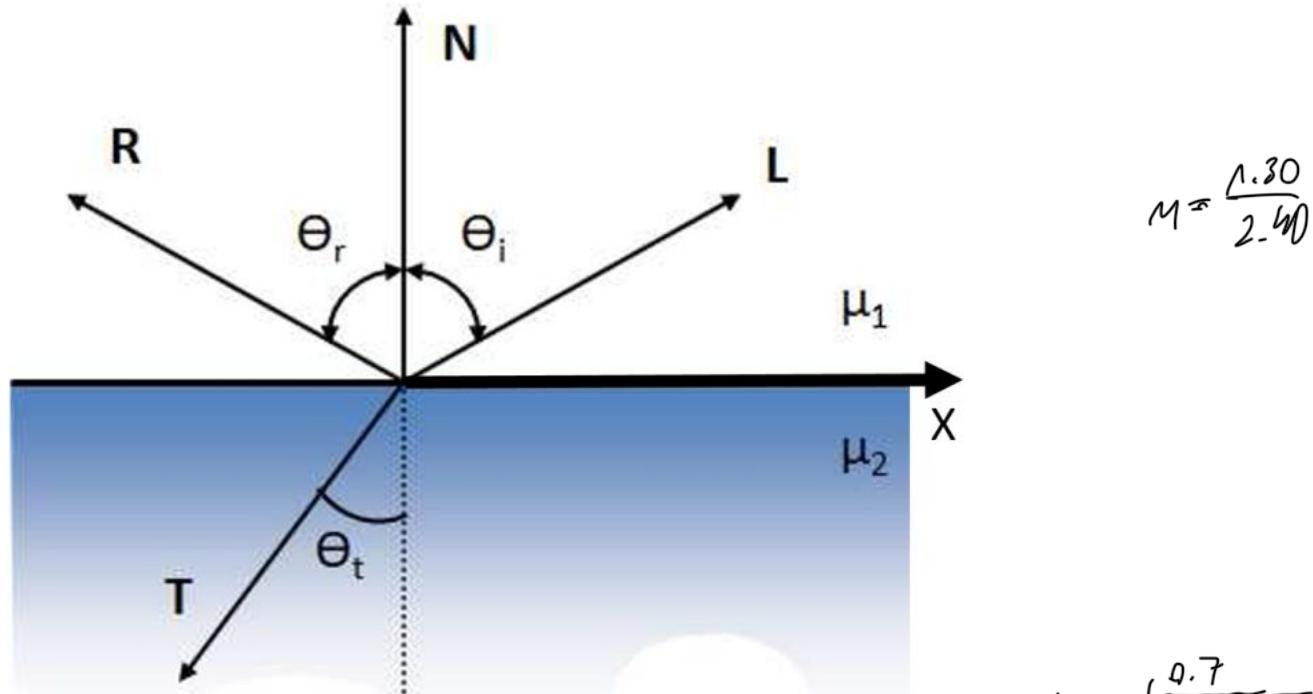
Puntuat sobre 1,00

$$\cos(\theta_i) = \frac{\text{dot}(N, L)}{|N| \cdot |L|}$$

$$\sin(\theta_i) = M \cdot \sin(\theta_i)$$

$$M = \frac{\mu_1}{\mu_2}$$

Assumint que  $N = (0, 1, 0)$  i  $L = (0.71, 0.70, 0.00)$ , i que els dos medi tenen índex de refracció 1.30 i 2.40, calcula (amb el signe correcte) la component X del vector unitari trasmès T.



[Cast]

Resposta:

$$T_x = -\sin(\theta_t) = -\left(\frac{1.3}{2.4} \cdot \sin(\arccos\left(\frac{0.7}{\sqrt{1 + 0.7^2}}\right))\right) \\ = -0.38682737321274$$

La resposta correcta és: -0,38682737321274

- sense el mòdul amb el

Pregunta 14

Correcte

Puntuació 1,00 sobre 1,00

Siguin:

M: submatriu 3x3 de la modelMatrix

V: submatriu 3x3 de la viewMatrix,  
la normalMatrix es pot calcular com...

[Cast]

Trieu-ne una:

- $(VM)^{-T}$
- $V^{-T}$
- $(MV)^{-1}$

← per mantenir vectors perpendiculars

- $(MV)^{-T}$
- No vull contestar la pregunta

La resposta correcta és:  $(VM)^{-T}$

Pregunta 15

Incorrecte

Puntuació 0,00 sobre 1,00

Indica el valor de u (enter) que fa correcte aquest codi:

```
QImage img4("file.png");
QImage T8 = img4.convertToFormat(QImage::Format_ARGB32);
glGenTextures(1, &textureId6);
 glBindTexture(GL_TEXTURE_2D, textureId6);
 glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, T8.width(), T8.height(), 0, GL_RGBA, GL_UNSIGNED_BYTE, T8.bits());
 g.glActiveTexture(GL_TEXTURE5);
 g glBindTexture(GL_TEXTURE_2D, textureId6);
 program->bind();
 program->setUniformValue("textureMap", u); // sampler2D
```

0

[Cast]

Resposta: 0

0

✗

La resposta correcta és: 5

5

Pregunta 16

Incorrecte

Puntuació -0,33 sobre 1,00

Indica la matriu que s'utilitza a la tècnica de shadow mapping per obtenir les coordenades de textura (s,t,p,q) d'un vèrtex, **si el vèrtex ja es troba en 'clip space of the light camera'**.

[Cast]

Trieu-ne una:

- $T(0.5)*S(0.5)$
- $S(0.5)*T(0.5)*P*V*M$
- No vull contestar la pregunta
- $T(0.5)*S(0.5)*P$
- $M*P*V$

A

CS → OS → CS

✗

La resposta correcta és:  $T(0.5)*S(0.5)$

CS → OS

✗

Pregunta 17

Incorrecte

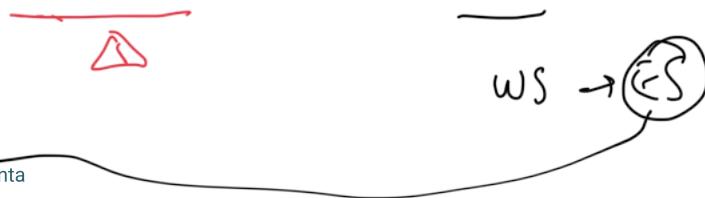
Puntuació -0,33 sobre 1,00

Tria l'espai de coordenades en que queda el resultat d'aplicar la transformació `viewMatrix*P`, suposant que P està en l'espai adient:

[Cast]

Trieu-ne una:

- clip space
- eye space
- object space
- world space
- No vull contestar la pregunta



La resposta correcta és: eye space

Pregunta 18

Correcte

Puntuació 1,00 sobre 1,00

L'expressió GLSL que representa l'expressió matemàtica  $K_d \cdot N \cdot L$  és:

[Cast]

$$matDiffuse * lightDiffuse * normalize(N) * normalize(L)$$

Trieu-ne una:

- matDiffuse \* lightDiffuse \* normalize(N) \* normalize(L)
- matDiffuse \* lightDiffuse \* N \* L
- matDiffuse \* lightDiffuse \* dot(N,L)
- No vull contestar la pregunta
- matDiffuse \* lightDiffuse \* normalize(N) \* L



La resposta correcta és: matDiffuse \* lightDiffuse \* dot(N,L)

Pregunta 19

Correcte

Puntuació 1,00 sobre 1,00

Tria l'espai de coordenades en que ha d'estar P per tal que la transformació `modelViewMatrix*P` tingui sentit

[Cast]

Trieu-ne una:

- No vull contestar la pregunta
- world space
- eye space
- clip space
- object space

$$P(OS \rightarrow WS + ES)$$



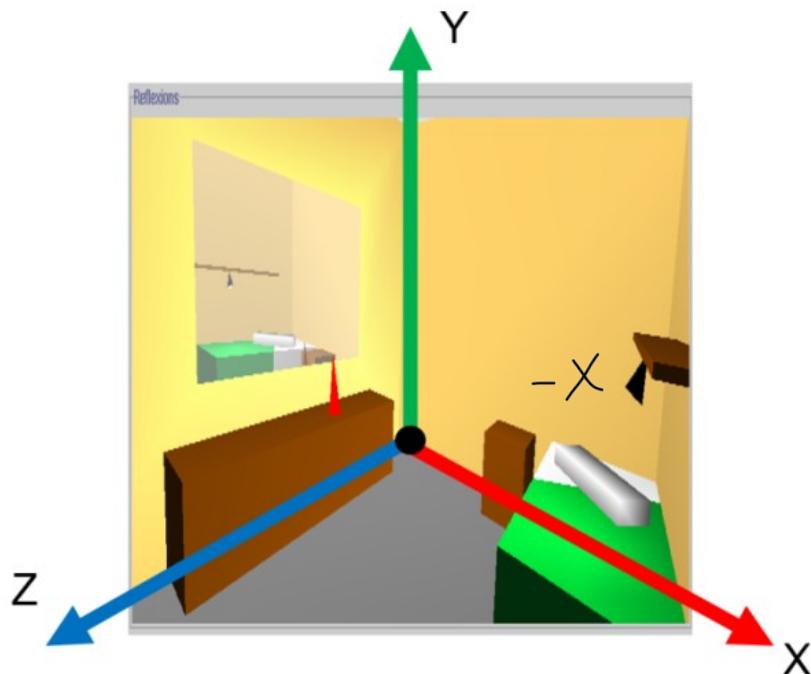
La resposta correcta és: object space

Pregunta 20

No s'ha respot

Puntuat sobre 1,00

Considerant la figura



la matriu de reflexió per dibuixar l'escena reflectida al mirall és...

[Cast]

Trieu-ne una:

No vull contestar la pregunta

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

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

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

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

(-X)

La resposta correcta és:  $\begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

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

[← Qüestionari 21 març 2022](#)

Salta a...

[Qüestionari competències transversals Juny 2022 →](#)

6) Qüestionari 18 Març

**Començat el** dimarts, 18 de març 2025, 11:57**Estat** Acabat**Completat el** dimarts, 18 de març 2025, 12:30**Temps emprat** 32 minuts 31 segons**Notes** 15,67/18,00**Qualificació** 8,70 sobre 10,00 (87,04%)**Pregunta 1**

Correcte

Puntuació 1,00 sobre 1,00

Disposem d'aquesta textura:



$$(0,0) \xrightarrow{\text{ }} \left( \begin{matrix} 5 \\ 6 \end{matrix} \right) \quad \left\{ \begin{matrix} u \\ v \end{matrix} \right\} = \left( \begin{matrix} 5 \\ 6 \end{matrix} \right)$$

(0,0)

Volem texturar un polígon rectangular situat sobre el pla  $Z = 0$ . Sabem que el seu vèrtex mínim té coordenades  $(0,0,0)$ , i el vèrtex màxim té coordenades  $(5, 6, 0)$ . Si usem dos plans  $(S,T)$  per a generar les coordenades de textura, indica l'opció que permet texturar el polígon així (ignora la relació d'aspecte):



- més instantàni  $\Rightarrow$  zoom-out  $\Rightarrow$  dividir coord
- més instantàni  $\Rightarrow$  zoom-in  $\Rightarrow$  multiplicar coord

$$4 \quad (u, v) = (5, 6) \rightarrow (s, t) = \left( \frac{u}{5}, \frac{v}{6} \right) = (0.6, 0.67)$$

[Cast]

3

Trieu-ne una:

- S=vec4(0.60, 0.00, 0.00, 0.00); T=vec4(0.00, 0.67, 0.00, 0.00); ✓
- S=vec4(0.25, 1.00, 0.00, 0.00); T=vec4(0.67, 0.25, 3.00, 0.00);
- S=vec4(3.00, 1.00, 0.25, 0.00); T=vec4(4.00, 1.00, 4.00, 0.00);
- S=vec4(1.00, 4.00, 1.00, 0.00); T=vec4(0.67, 0.67, 0.25, 0.00);

La resposta correcta és: S=vec4(0.60, 0.00, 0.00, 0.00); T=vec4(0.00, 0.67, 0.00, 0.00);

**Pregunta 2**

Correcte

Puntuació 1,00 sobre 1,00

Les diferents etapes del pipeline d'OpenGL (VS, etc) comencen a executar-se quan s'invoca la funció...

[Cast]

Trieu-ne una: *Envia dades GPU (i espera)*

- glFinish()
  - glStart() *NO EXISTIX*
  - glDrawElements() ✓ *glDrawElements()*
  - glFlush()
- envia dades GPU No espera*

La resposta correcta és: **glDrawElements()**

- Elements (*indexes*)
- Arrays (*vèrtexs d'un format*)

**Pregunta 3**

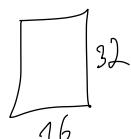
Incorrecte

Puntuació 0,00 sobre 1,00

Indica el valor més gran que pot tenir la coordenada de textura s després del wrapping, si hem activat el mode **GL\_CLAMP\_TO\_EDGE**, la textura té una mida de 16x32 texels:

[Cast]

Resposta: **0,96875**



$$S = \frac{(16 - 0.5)}{16} = \frac{15.5}{16} = 0.96875$$

La resposta correcta és: 0,96875

**Pregunta 4**

Correcte

Puntuació 1,00 sobre 1,00

**(2,5) → (12,18)**

En un determinat objecte, la coordenada s d'un fragment varia en (2, 5). Per convertir linealment aquesta coordenada per tal d'omplir l'interval (12, 18), la transformació correcta és...

[Cast]

Trieu-ne una:

- $2 + (s-2)/(18-12) * 18$
- $(s-2)/(5-2)*(18-12) + 12$
- $(s-2)/(5-2) * 18$
- $12 + (s-2)/(5-2) * (18-2)$

- Desplaça origen  $s-2$   $(0,3)$
- Normalitza  $(5-2)$   $(0,1)$
- Escala  $* (18-12)$   $(0,6)$

La resposta correcta és:  $(s-2)/(5-2)*(18-12) + 12$

- Desplaça on a destí  $+ 12$   $(12,18)$

$$S = \frac{(s-2)}{(5-2)} * (18-12) + 12$$

**Pregunta 5**

Correcte

Puntuació 1,00 sobre 1,00

Què fa aquest codi?

```
vec4 aux = modelViewProjectionMatrix * vec4(vertex, 1.0); aux (clip space)
vec3 foo = aux.xyz / aux.w; NDC (perspective division)
```

[Cast]

Trieu-ne una:

- Calcula la posició del vertex en eye space
- Calcula la posició del vertex en object space
- Projecta el vèrtex sobre una esfera
- Passa vertex a clip space, i a continuació fa la divisió de perspectiva ✓

OBJECT SPACE

↓ model matrix

WORLD SPACE

↓ view matrix

EYE SPACE

↓ projection matrix

CLIP SPACE

 $(\frac{x}{w}, \frac{y}{w}, \frac{z}{w})$  Divisió de perspectiva

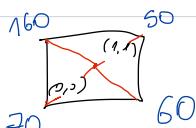
NDC (NORMALIZED DEVICE COORDINATES)  
 $\text{viewport}(w, h) \downarrow (x = (\text{fwd}) \frac{w}{2}, y = (\text{fwd}) \frac{h}{2}, z = \frac{z^m}{2})$   
 wS (WINDOW SPACE)

La resposta correcta és: Passa vertex a clip space, i a continuació fa la divisió de perspectiva

**Pregunta 6**

Correcte

Puntuació 1,00 sobre 1,00

Donat el codi següent, indica el resultat de calcular una mostra de la textura al punt amb coordenades (s,t) = (1.5, 2.5).  $\equiv [0.5, 0.5]$ 

```
// Dades de la textura (4 texels monocromàtics)
GLubyte textureData[4] = {
    70, // Texel (0,0) - lower left
    60, // Texel (1,0) - lower right
    160, // Texel (0,1) - upper left
    50 // Texel (1,1) - upper right
};

glEnable(GL_TEXTURE_2D);
GLuint textureID;
 glGenTextures(1, &textureID);
 glBindTexture(GL_TEXTURE_2D, textureID);
 glTexImage2D(
    GL_TEXTURE_2D,
    0, // mipmap level
    GL_LUMINANCE, // internal format
    2, // width
    2, // height
    0, // border
    GL_LUMINANCE, // buffer format
    GL_UNSIGNED_BYTE, // data type
    textureData // pointer to data
);

glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
```

S'interpolen els texels

per a la matrícula divisionar  $\Rightarrow (0.25, 0.25)$ 

$$V = 0.25 (70 + 60 + 160 + 50) = 85$$

es repeteixen les word però diàument

[Cast]

Resposta: 85 ✓

La resposta correcta és: 85

**Pregunta 7**

Correcte

Puntuació 1,00 sobre 1,00

permilte  
 $\epsilon [0,1]$

Com a part d'un procés de generació de coordenades de textura, volem projectar un model centrat en el punt C sobre una esfera unitària. El mètode més adient és...

[Cast]

Trieu-ne una:

- vec3 P = vertex.xyz - C;
- vec3 P = distance(vertex.xyz, C);
- vec3 P = (modelViewProjectionMatrix \* vertex).xyz - C;
- vec3 P = normalize(vertex.xyz - C); ✓

La resposta correcta és: vec3 P = normalize(vertex.xyz - C);

**Pregunta 8**

Correcte

Puntuació 1,00 sobre 1,00

*divisio per w*

$$(x, y, z, w) \rightarrow \left( \frac{x}{w}, \frac{y}{w}, \frac{z}{w} \right)$$

El punt amb coordenades homogènies (36.00, 24.00, 36.00, 4.00) correspon al punt 3D...

[Cast]

Trieu-ne una:

- (-36.00, -24.00, -36.00)
- (6.00, 9.00, 9.00)
- (36.00, 24.00, 36.00)
- (9.00, 6.00, 9.00) ✓

La resposta correcta és: (9.00, 6.00, 9.00)

**Pregunta 9**

Correcte

Puntuació 1,00 sobre 1,00

Les dades que li arriben interpolades al FS per cada fragment corresponen a...

[Cast]

Trieu-ne una:

- Variables uniform - *com常数*
- Dades del VAO } *entre VS i FN, FS no hi accedeix*
- Dades dels VBOs
- Variables out del VS ✓ *Salida*

La resposta correcta és: Variables out del VS

**Pregunta 10**

Incorrecte

Puntuació -0,33 sobre 1,00

Indica com calcular la coordenada s per texturar una esfera unitària amb una projecció equirectangular, usant únicament el mapa de la dreta.

Suposa que  $\theta = \text{atan}(\text{vertex.x}, \text{vertex.z}) + \pi$ , amb  $\theta$  en  $[0, 2\pi]$ .



Inicialment tenim:

$$\theta \in [0, 2\pi]$$

$$s = \frac{\theta}{2\pi} \in [0, 1]$$

Ara més aviat venen els dos mapes  $\theta \in [0^\circ, 360^\circ]$

[Cast]

Per veure le mitjà de le ferre normativament a  $[0, 0.5]$  dividir per 2 s

Trieu-ne una:

- $s = \theta / \pi + 0.5; s = s/2 + 0.5$
- $s = \theta / (2\pi) + 0.5 \times$
- $s = \theta / (4\pi) + 0.5 \checkmark$
- $s = \theta / (2\pi)$

La resposta correcta és:  $s = \theta / (4\pi) + 0.5$

$$s = \frac{(\theta/2\pi)}{2} \in [0, 0.5]$$

Amb ençò venen el mapa do equirectangular de la dreta i  
a un offset de 0.5:

$$s = \frac{\theta}{2(2\pi)} + 0.5 = \boxed{\frac{\theta}{4\pi} + 0.5}$$

**Pregunta 11**

Correcte

Puntuació 1,00 sobre 1,00

Selecciona la única matriu de projecció (projectionMatrix) plausible per a una càmera perspectiva:

[Cast]

Trieu-ne una:

- |     |     |     |     |
|-----|-----|-----|-----|
| 1.0 | 0   | 0   | 5.0 |
| 0   | 1.0 | 0   | 1.0 |
| 0   | 0   | 3.0 | 3.0 |
| 0   | 0   | 0   | 1.0 |
- |     |     |      |       |
|-----|-----|------|-------|
| 3.0 | 0.0 | 0.0  | 0.0   |
| 0.0 | 1.5 | 0.0  | 0.0   |
| 0.0 | 0.0 | -3.0 | -12.0 |
| 0.0 | 0.0 | -1.0 | 0.0   |

Perspectiva
 $(0 \ 0 \ -1 \ 0)$
- |     |     |     |     |
|-----|-----|-----|-----|
| 1.0 | 0   | 0   | 5.0 |
| 0   | 1.0 | 0   | 1.0 |
| 0   | 0   | 1.0 | 1.0 |
| 0   | 0   | 0   | 1.0 |

Ortogonal
 $(0 \ 0 \ 0 \ 1)$
- |     |     |     |     |
|-----|-----|-----|-----|
| 1.0 | 0   | 0   | 5.0 |
| 0   | 1.0 | 0   | 1.0 |
| 0   | 0   | 2.0 | 2.0 |
| 0   | 0   | 0   | 1.0 |

La resposta correcta és:

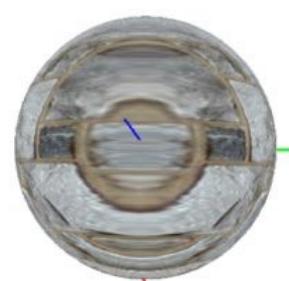
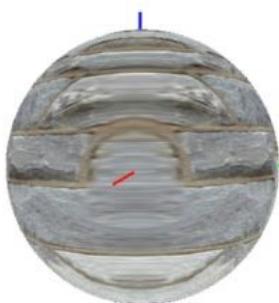
$$\begin{bmatrix} 3.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 1.5 & 0.0 & 0.0 \\ 0.0 & 0.0 & -3.0 & -12.0 \\ 0.0 & 0.0 & -1.0 & 0.0 \end{bmatrix}$$

**Pregunta 12**

Correcte

Puntuació 1,00 sobre 1,00

Indica els plans S, T que calen per texturar una esfera de radi  $R=1$  centrada a l'origen, com indica la figura (es mostra la textura, i el resultat dels eixos X, -Y i Z).



[Cast]  $\text{d'aproximació}$   $\left\{ \begin{array}{l} x \in [-1,1] \\ z \in [-1,1] \end{array} \right.$

Trieu-ne una:

- vec4 S = vec4(1.0, 0, 0, 0); vec4 T = vec4(0,0,1.0, 0);
- vec4 S = vec4(1, 0, 0, 0); vec4 T = vec4(0,0,1, 0);
- vec4 S = vec4(0.5, 0, 0, 0.5); vec4 T = vec4(0,0,0.5, 0.5); ✓
- vec4 S = vec4(1.0, 0, 0, 0.5); vec4 T = vec4(0,0,1.0, 0.5);

Volent  $\left\{ \begin{array}{l} x \in [-1,1] \rightarrow x \in [0,2] \\ z \in [-1,1] \rightarrow z \in [0,1] \end{array} \right.$

$[-1,1] \rightarrow [0,1]$  és dividir entre 2 i sumar

$$0,5. \text{ S'aconsegueix} \left\{ \begin{array}{l} 0,5x + 0,5 \\ 0,5z + 0,5 \end{array} \right.$$

La resposta correcta és: vec4 S = vec4(0.5, 0, 0, 0.5); vec4 T = vec4(0,0,0.5, 0.5);

**Pregunta 13**

Correcte

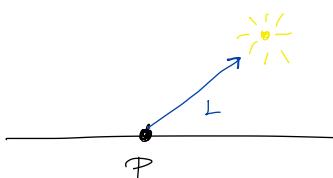
Puntuació 1,00 sobre 1,00

Un FS rep una variable **in vec3 P** amb la posició del fragment en *eye space*. Per calcular el *light vector* L cal usar...

[Cast]

Trieu-ne una:

- vec3 L = normalize(P - lightPosition.xyz);
- vec3 L = normalize(lightPosition.xyz);
- vec3 L = normalize((modelViewMatrix \* lightPosition).xyz - P);
- vec3 L = normalize(lightPosition.xyz - P); ✓



La resposta correcta és: vec3 L = normalize(lightPosition.xyz - P);

**Pregunta 14**

Correcte

Puntuació 1,00 sobre 1,00

El punt 3D que resulta d'aplicar la transformació representada per la matriu al punt (30.00, 36.00, 36.00, 6.00) és...

[Cast]

Trieu-ne una:

- (36.00, 36.00, 30.00)
- (6.00, 9.00, 9.00) ✓
- (5.00, 6.00, 36.00)
- (36.00, 54.00, 54.00)

$$\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 30 \\ 36 \\ 36 \\ 6 \end{bmatrix} = \begin{bmatrix} 30+6 \\ 36+18 \\ 36+18 \\ 6 \end{bmatrix} = \begin{bmatrix} 36 \\ 54 \\ 54 \\ 6 \end{bmatrix} = \begin{bmatrix} 36/6 \\ 54/6 \\ 54/6 \\ 6 \end{bmatrix} = (6, 9, 9)$$

La resposta correcta és: (6.00, 9.00, 9.00)

**Pregunta 15**

Correcte

Puntuació 1,00 sobre 1,00

Indica la transformació geomètrica que **no** es pot aplicar com el producte d'una matriu **3x3** per un punt (x,y,z):

[Cast]

Trieu-ne una:

- escalat uniforme
- escalat no uniforme
- rotació
- projecció ✓

$\left. \begin{array}{l} 3 \times 3 \\ 4 \times 4 \end{array} \right\}$  Col·les coordenades homogenies per fer les divisió de perspectiva

La resposta correcta és: projecció

**Pregunta 16**

Correcte

Puntuació 1,00 sobre 1,00

En el mipmap d'una textura de 512x512 texels, un texel del LOD 8 correspon a un grup de NxN texels del LOD 0. Indica el valor de N (enter):

[Cast]

Resposta: 256 ✓

NxN	Relació amb LOD 0
1x1	
2x2	
4x4	
8x8	
16x16	
32x32	
64x64	
128x128	
256x256	
512x512	
1x1	
2x2	
4x4	
8x8	
16x16	
32x32	
64x64	
128x128	
256x256	
512x512	

La resposta correcta és: 256

**Pregunta 17**

Correcte

Puntuació 1,00 sobre 1,00

Indica en quina d'aquestes etapes del pipeline cal interpolar les sortides (variables **out**) del VS:

[Cast]

Trieu-ne una:

- triangle* *face* *normal* *position* *texCoord*
- Clipping ✓ Si el taller un triangle s'interpolen les cot variables dels més nuls
  - Viewport transformation ( $NDC \rightarrow WS$ ) No interpole
  - Perspective division ( $xyz/w$ ) No interpole
  - Back face culling Depen de la normal del triangle, no interpole

La resposta correcta és: Clipping

**Pregunta 18**

Correcte

Puntuació 1,00 sobre 1,00

Tria l'espai de coordenades en que ha d'estar P per tal que la transformació **projectionMatrixInverse\*P** tingui sentit

[Cast]  $\text{projectionMatrix} \quad (\text{eyeSpace} \rightarrow \text{Clip Space})$ 

Trieu-ne una:



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

 $\text{projectionMatrixInverse} \quad (\text{clipSpace} \rightarrow \text{EyeSpace})$ 

La resposta correcta és: clip space

## 7) Reflexió AteneaLabs

**Començat el** dissabte, 14 de juny 2025, 10:20

**Estat** Acabat

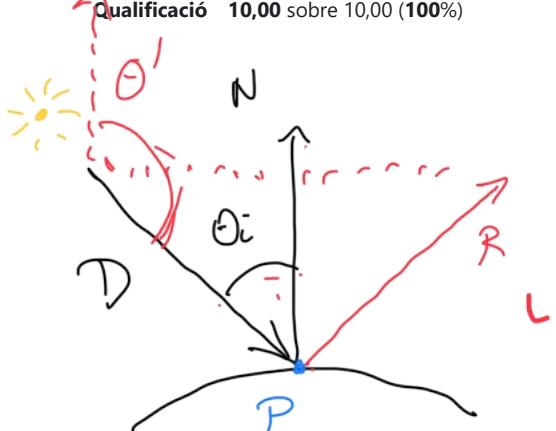
**Completat el** dissabte, 14 de juny 2025, 10:22

**Temps emprat** 1 minut 43 segons

**Notes** 1,00/1,00

**Qualificació** 10,00 sobre 10,00 (100%)

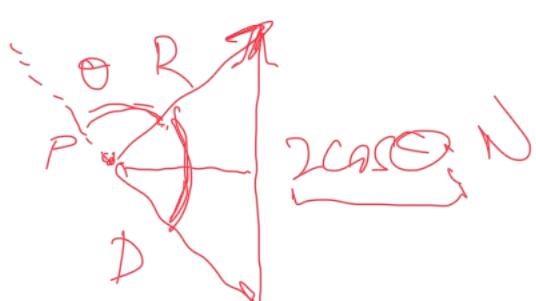
$$D = 2 + \cos(\theta_i) \cdot N$$



$$\theta' = 180^\circ - \theta_i$$

$$\cos(\theta_i) = -\cos(\theta')$$

↳ d'aquí surt el resultat



**Pregunta 1**

Correcte

Puntuació 1,00 sobre 1,00

Un raig el podem representar d'aquesta forma:

```
struct Ray
{
    vec3 origin;
    vec3 direction;
};
```

Completa la funció següent de forma que donats

- **Ray R** amb un raig incident,
- **vec3 P** amb la posició de la intersecció del raig amb la superfície, i
- **vec3 N** amb la normal al punt d'intersecció,

retorni un **Ray** amb el **raig reflectit** especularment en el punt P.

**Resposta:** (règim de penalització: 10, 20, ... %)

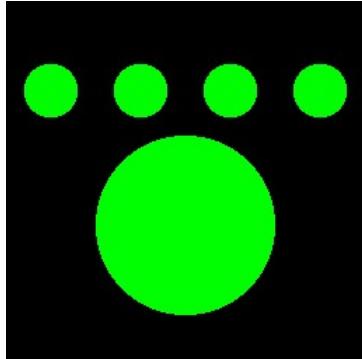
[Reinicia la resposta](#)

```
1 Ray reflectRay(Ray R, vec3 P, vec3 N) {
2     vec3 D = normalize(R.direction);
3     Ray reflectedRay;
4     reflectedRay.origin = P;
5     reflectedRay.direction = D - 2.0 * dot(N,D)*N;
6     return reflectedRay;
7 }
8
```

...

**Output:**

El codi compila. El color de cada cercle indica si ha passat o no el test. El color del cercle gran és el resultat global.

**output.png**

S'han passat totes les proves! ✓

[Correcte](#)

Puntuacions per a aquesta tramesa: 1,00 / 1,00.

## 8) Extres

## Examens

2021-01-13:

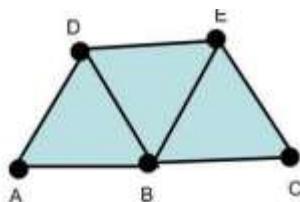
Tria l'espai de coordenades en que ha d'estar P per tal que la transformació `projectionMatrixInverse * P` tingui sentit.

- **Correcta (◎):** Clip space
  - **Incorrectes (0):** 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:

- **Correcta (◎):**
    1. `glGenVertexArrays`
    2. S'escriu `gl_Position`
    3. Backface culling
    4. Rasterització
- 

Indica un ordre adient per emetre els vèrtexs dels triangles de la figura, en un GS, usant una única primitiva (D E, A B C):



- **Correcta(◎):** ADBEC
- 

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

- **Correcta (◎):** (1.00, 5.00, 6.00, 2.00) en clip space
  - **Incorrectes (0):**
    - (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)`

- **Correcta (C):** [4.8]
- 

Indica el valor que retorna aquesta expressió GLSL: `mod(7.9, 4)`

- **Correcta (C):** [3.9000000000000004]
- 

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

- **Correcte(C):** `fragColor = vec4(1-step(295, distance(gl_FragCoord.xy, vec2(1024)))`
- 

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

- **Correcta (C):** Vector normal
  - **Incorrectes (0):**
    - Projecció del vector posició sobre una esfera unitària
    - Vector de reflexió especular de la llum directa
    - Vector tangent a la superfície en el punt
- 

Quina transformació de la z en window space té un efecte equivalent a invertir el depth test amb `glDepthFunc(GL_GREATER)`:

- **Correcta (C):** `gl_FragDepth = 1 - gl_FragCoord.z;`
  - **Incorrectes (0):**
    - `gl_FragDepth = -1 * gl_FragCoord.z;`
    - `gl_FragDepth = 0.5 * gl_FragCoord.z;`
    - `gl_FragCoord.z = 1 - gl_FragCoord.z;`
- 

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)`

- **Correcte(C):** `factor=vec2(0.1, 1.0); offset=vec2(0.1, 0.0);`
- 

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...

- **Correcta (©):**
  - $[1 \ 0 \ 0 \ 0]$
  - $[0 \ -1 \ 0 \ 0]$
  - $[0 \ 0 \ 1 \ 0]$
  - $[0 \ 0 \ 0 \ 1]$
- 

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

- **Correcta (©):**
    1. El VS passa P, N a eye space
    2. Es calcula el vector reflectit
    3. Es calculen les coordenades (s, t) del fragment
    4. Es pren una mostra de la textura que conté el sphere map
- 

**Tenint en compte la llei de Snell, indica quina parella de valors (1, 2) explicarien de forma aproximada la direcció del raig transmès observat:**

- **Correcte(©):** (1.09, 1.43)

2020-01-14:

Exercici 1

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

Correcte:

Geometry shader

Rasterització

Fragment shader

Alpha Blending

Exercici 2

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

glDrawElements

Vertex Shader

Stencil Test

Depth Test

Exercici 3

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

Inicial: Clip space

Final: Object space

Exercici 4

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

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

Exercici 5

Què coordenada del fragment modifica la funció glPolygonOffset?

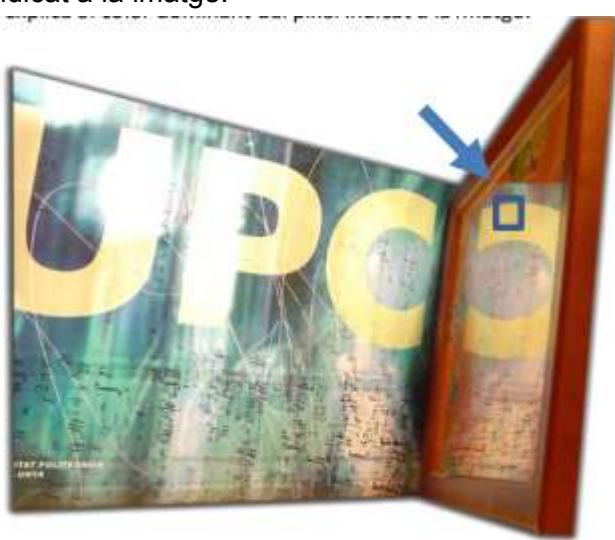
Z

En quin espai la modifica?

Window space

Exercici 6

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



Camí possible:

L → D → M → E

Aquest camí indica:

La llum surt de la font (L).

Es reflecteix difusament en alguna superfície (D).

És reflectida pel mirall (M).

Arriba a l'observador o càmera (E).

Exercici 7

Indica, per cada punt, si potser 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

DINS (tot entre +/- w)

(b) (0, 0, 10, 1) en object space DINS (pot ser...)

### 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;
```

```
vec3 color = mix(color2, color1, t);
```

### Exercici 9

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

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

```
vec3 obs = modelViewMatrixInverse[3].xyz;
```

### Exercici 10

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

- (a) LS\*DS\*E (i LS\*E)      Two-pass raytracing  
(b) L(D|S)\*E      Path tracing

### 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);  
vec2(1,0.5)
```

### 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.

$T(0.5)S(0.5)PV$

### 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  $\Omega$ ?

Totes les direccions en una semi-esfera centrada al punt.

### 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

projectionMatrixInverse \* modelViewProjectionMatrix

### Exercici 15

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

(a) Segon paràmetre d'una crida a texture(colormap...)    vec2

(b) Coordenades de textura que passa VS a FS en shadow mapping    vec4

(c) Coordenades de textura que passa VS a FS en projective texture mapping vec4

(d) Vector per accedir a un sphere map    vec3

### Exercici 16

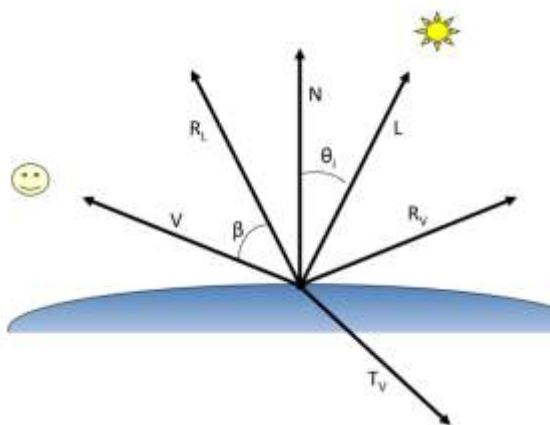
Amb la notació de la figura, indica, en el cas de Ray-tracing

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

L

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

T



### Exercici 17

Continuant amb la figura anterior...

- (a) Quin vector té la direcció del raig primari?

V

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

RL i V

### Exercici 18

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

$\text{W}/(\text{m}^2 * \text{sr})$

## 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
```

```
si es_transparent(info.obj) llavors
    raigT = calcula_raig_refractat(info, raig, μ)
    color += kt * traçar_raig(raigT, escena, μ) // Suma el color transmès
    fsi
```

## 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()
{
    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);  
  
}
```

2019-06-05

### Exercici 1

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

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.

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      ABANS
- (b) Geometry Shader      ABANS
- (c) Divisió de perspectiva      ABANS
- (d) Depth Test      DESPRES

#### 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      DINS (tot ent re +/- w)
- (b) (0, 0, 1, 1) en eye space      FORA (z no negativa)
- (c) (200, 300, 400) en NDC      FORA (no en -1..1)
- (d) (0, 0, 0) en NDC      DINS

### Exercici 7

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

```
vec3 mix(vec3 a, vec3 b, float t)  
{  
    return (1-t)*a + t*b;  
}
```

### 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):

$T^*R^*S$

### Exercici 9

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

(a) Two-pass raytracing

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

(b) Classic Raytracing

$LDS^*E$  &  $LS^*E$

### 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:



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

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

(b)



```
frontColor = texture(colorMap, _____) * vtxCoord);  
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 W/m<sup>2</sup>)?

Irradiància

Exercici 12

A l'equació general del rendering:

Què representa  $\omega_o$ ?

Vector unitari en la direcció de sortida.

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

$M$  -> model matrix (of the object)     $V$  -> view matrix (of the light camera)

$P$  -> projection matrix (of the light camera)

(a) Quin valor ha de tenir la constant  $a$ ? 0.5

(b) Escriu (com a producte de matrius) com l'aplicació ha de calcular, en aquest cas, la matriu  $\text{lightMatrix}$ .

$P^*V^*M$

#### Exercici 14

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     $P * V$

b) Convertir un vèrtex de eye space a world space     $V^{-1}$

#### 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.

$$P'(u,v) = P(u,v) + F(u,v)*N(u,v)$$

#### 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 = modelViewMatrixInverse[3].xyz;
```

```
Més eficient que (modelViewMatrixInverse*vec4(0,0,0,1)).xyz;
```

### 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

```
out vec4 vtexCoord;
```

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

```
vec4 color = texture(colormap, .....);    vtexCoord.st /  
vtexCoord.q
```

### Exercici 18

En la tècnica de generació d'ombres per projecció,

- (a) Quants cop cal dibuixar l'objecte que produceix l'ombra? 2

- (b) Per què ens pot ser útil usar el stencil buffer? Limitar l'ombra al receptor.

### Exercici 19

En quin sistema de coordenades 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 \_\_\_\_\_ model space

- (b) `gl_Position = projectionMatrix * P;`

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

### 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()? 8 texels

2019-01-15:

# Examen Final de Gràfics Curs 2014-15 Q1

## Exercici 1

Ordena las siguientes etapas del pipeline gráfico según su orden correcto. Actualmente están en orden alfabético:

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

## Exercici 2

Dado un color RGBA de fragmento  $(1.0, 0.5, 0.0, 0.4)(1.0, 0.5, 0.0, 0.4)$  y un color de pixel RGBA en el buffer de color  $(0.5, 0.5, 1.0, 1.0)(0.5, 0.5, 1.0, 1.0)$ , calcula el color resultante del buffer utilizando glBlendFunc(GL\_SRC\_ALPHA, GL\_ONE\_MINUS\_SRC\_ALPHA):

$$\begin{aligned}0.4*1.0 + 0.6*0.5 &= 0.7 \\0.4*0.5 + 0.6*0.5 &= 0.5 \\0.4*0.0 + 0.6*1.0 &= 0.6 \\0.4*0.4 + 0.6*1.0 &= 0.76 \quad (0.7, 0.5, 0.6, 0.76)\end{aligned}$$

## Exercici 3

Indica las matrices requeridas para las siguientes transformaciones:

- a) De object space a eye space: MV
- b) De eye space a world space: MV<sup>-1</sup>
- c) De clip space a object space: MVP<sup>-1</sup>
- d) De normal de object space a eye space: NM

## Exercici 4

En GLSL, si no existe la variable gl\_ModelViewProjectionMatrix, completa la conversión de gl\_Vertex de object space a clip space:

```
gl_Position = gl_ModelViewProjectionMatrix * gl_Vertex;  
->  
gl_Position = gl_ProjectionMatrix * gl_ModelViewMatrix * gl_Vertex;
```

## Exercici 5

Completa este Fragment Shader para calcular la contribución difusa:

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

```

## Exercici 6

¿Por qué puede ser útil esta instrucción GLSL?

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

Respuesta: Para obtener la posición del observador en object space o world space.

## Exercici 7

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

$1024 \times 1024 + 512 \times 512 + 256 \times 256 + \dots 1 \times 1 = \lVert (2^{10}, 2^9, \dots, 2^0) \rVert^2 = 1.33 \text{ M texel}$

## 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?

Vector L: vector unitari del punt cap a l'observador

## 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?

Irradiancia

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

(W/m<sup>2</sup>,lux)

## Exercici 10

Asocia conceptos de radiometría con sus definiciones:

Flux: Cantidad total de energía emitida por una fuente de luz (por unidad de tiempo).

Radiància: Energía de un rayo de luz.

Intensitat: Energía emitida por unidad de tiempo y ángulo sólido.

BRDF: Propiedades de reflectividad de un material.

## 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 (vigieu l'ordre!).

$T(\text{centre}) * T(\text{trans}) * R * S * T(-\text{centre})$ .

## Exercici 12

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

Bàsicament un VBO és un VA emmagatzemat en memòria del servidor OpenGL (ex. GPU). Per tant, estalvi en les dades que han de passar-se cada frame de RAM a memòria de la GPU.

## 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.

Simplement hem de passar la normal ( $nx', ny', nz'$ ) de tangent space a eye space:  $[T \ B \ N] * (nx', ny', nz')$  on  $[T \ B \ N]$  és una matriu  $3 \times 3$  que té per columnes els tres vectors T, B, N.

## 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?

Eye Space --> Clip Space

## 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.

$-w \leq x \leq w$  (el mateix per y,z)

## 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?

Activar (true) / desactivar (false) l'escriptura en els buffers de color.

b) Menciona una tècnica concreta (dintre de les tècniques per simular ombres, reflexions, transparències...) en la qual es faci servir, i perquè.

Ombres per projecció amb stencil, als passos on només volem modificar el depth buffer o el stencil buffer, però no pas el color buffer.

## 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 espectral del material
    float shininess = gl_FrontMaterial.shininess
b) vec4 posLlum =             // posició en eye space de la primera llum
    vec4 posLlum = gl_LightSource[0].position

```

## 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))
   
float
- a) texture2d(sampler, vec2(0.5,0.5));
   
vec4
- b) cross (vec3(1,0,0), vec3(0, 1, 0))
   
vec3
- c) vec3(1,0,0) \* vec3(0, 1, 0)
   
vec3

## 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(...)

```

Està definint la matriu de conversió world space -> texture space; es fa servir a proj texture mapping.

## 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?

Aliasing greu a l'ombra degut al problema "duelling frustra".

## Examen Final de Gràfics Curs 2013-14 Q1

### 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) | Vertex Shader (VS)   |
| - Geometry Shader (GS) | Geometry Shader (GS) |
| - Rasteritzacio        | Rasteritzacio        |
| - Stencil test         | Fragment Shader (FS) |
| - Vertex Shader (VS)   | Stencil test         |

### 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)?  
Object space
- (b) En quin espai està V?  
Eye space

### 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 = max( 0.0, dot( N,L ) );
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?

NO

## 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)  $\Delta s = 1/(2 \cdot 256)$
- (d)  $\Delta s = \log(256)$

## Exercici 9

Respecte 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

càculs 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.

Volem que  $(x,y,z)$  es projecti a  $(x, y, 5)$ . Per tant la única diferència amb la matriu identitat serà la tercera fila, que serà  $(0, 0, 0, 5)$ :

## 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;
    vec3 I = normalize(P - obs);
    R = reflect(I, N);
    gl_FragColor = sampleSphereMap(spheremap, R);
}

```

## 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);

```

Cal afegir una crida a `glMultMatrix(matriu simetria)` immediatament després de `glPushMatrix`.

## 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? 2 vegades

(b) Shadow mapping, dues fonts de llum dinàmiques, amb un FS: quantes vegades cal pintar l'escena cada frame? 3

### Exercici 17

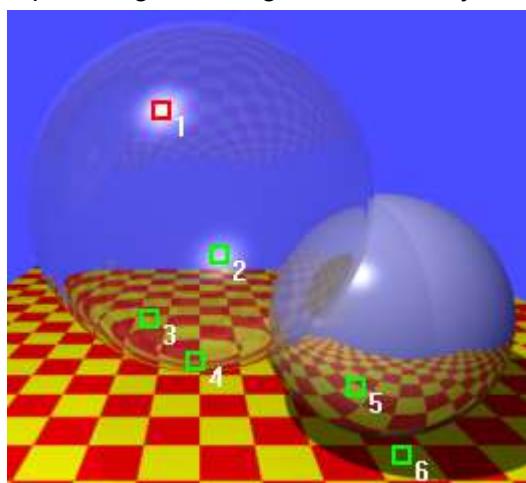
Indica quins d'aquests camins seria capaç de simular una implementació de l'algorisme clàssic de raytracing amb nivell màxim de recursitat 10:

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

Només el camí (d); als altres la recursitat s'atura quan es troba l'objecte difós

### 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:

- 1: No (domina highlight espectral de la reflexió del raig primari)
- 2,3,4: Sí (els camins dominants tenen refracció)
- 5-6: No (només hi ha reflexions)

## Examen Final de Gràfics Curs 2012-13 Q2

### 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:

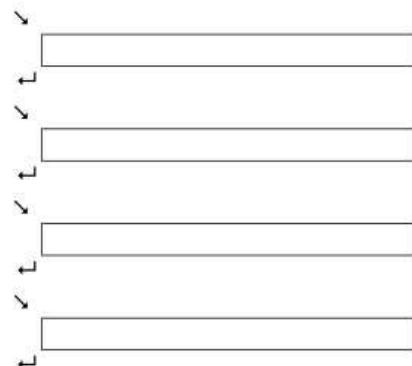
- a) Amb un arbre BSP sempre trobem la primera intersecció de cada raig en temps logarítmic.
  - b) El BSP és un mètode de subdivisió dels objectes.
  - c) Els Octrees són un mètode de subdivisió de l'espai.
  - d) Les caixes d'un arbre d'AABB són disjunes i cada objecte pertany a sols una fulla.
  - 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)

Col·loca els diferents sistemes de coordenades a la caixa de la dreta ordenats de la manera que apareixen al pipeline d'OpenGL, indicant les matrius o operacions de transformació necessàries per passar d'un sistema al següent.

Clipping coords.  
World coords  
Device coords  
Normalized DC  
Eye Coordinates

1.  
2.  
3.  
4.  
5.



---

### Exercici 5 (1 punt)

Compara les tècniques de Sphere Mapping i Cube Mapping per pintar un objecte brillant reflectint l'entorn.

---

### 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 per què:

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

### Exercici 8 (1 punt)

Com afecta la introducció d'un Geometry Shader al pipeline en el pas d'informació entre el VS i el FS? Explica per què succeeix i si és raonable.

---

### Exercici 9 (1 punt)

Tenim un polígon quadrat amb vèrtexs  $(-1, 0, -1)$ ,  $(1, 0, -1)$ ,  $(1, 0, 1)$  i  $(-1, 0, 1)$  té coordenades de textura  $(s, t)$  respectivament:  $(0,0)$ ,  $(2,0)$ ,  $(2,2)$  i  $(0,2)$ .

**AB**  
**CD**

Respon:

- a) Quantes lletres hi haurà a la projecció del polígon si s'utilitza el mode de wrapping GL\_REPEAT?
- b) Com hauríem d'assignar les coordenades de textura i quin mode de wrapping seria adient per obtenir diferents resultats?

**ABAB**  
**CDCD**  
**ABAB**  
**CDCD**

**C**

---

### **Exercici 10 (1 punt)**

Quan fem servir la tècnica d'objectes virtuals per a simular reflexions, aquests poden aparèixer directament a l'escena. Explica dues maneres d'evitar-ho en OpenGL.

---

#### **Preguntes per a l'avaluació de competències transversals:**

**Pregunta 1:** Relaciona les afirmacions següents sobre gràfics en medicina amb SI/NO:

- a) Els actuals scanners 3D proporcionen dades amb un nivell de soroll negligible.
- b) La segmentació de dades mèdiques assigna etiquetes a cada voxel segons el tipus de teixit/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:** Explica en què consisteix una endoscòpia virtual, els seus avantatges i limitacions respecte l'endoscòpia tradicional.



## Examen Final de Gràfics Curs 2012-13 Q1

### Exercici 1

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

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

### Exercici 2

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

```
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;  
}
```

#### 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

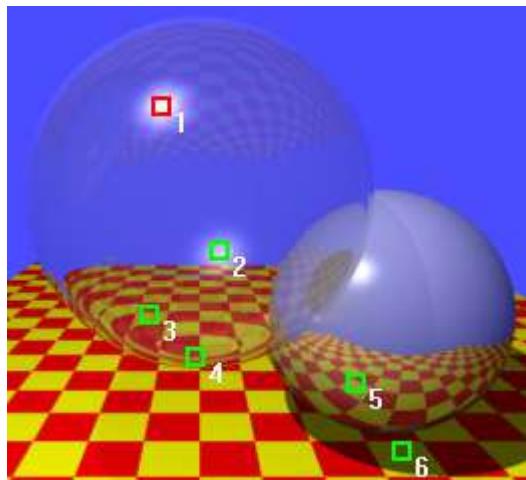
Quines equacions permeten calcular la proporció de llum reflectida i la proporció de llum transmesa, quan la llum incideix en la superfície de separació entre dos medis?

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

## Examen Final de Gràfics Curs 2011-12 Q2

### 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) {
    return normalize( light.position.xyz - vertex );
}
```

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

Vertex shader:

```
varying vec3 normal;
```

```
void main() {
    normal = gl_NormalMatrix * gl_Normal;
    gl_Position = gl_ModelViewProjectionMatrix * gl_Vertex;
    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 = texture2D(textureImage, gl_TexCoord[0].st);
}
```

### 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).

**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();

for (int i = 2; i >= 0; i--) {
    gl_FrontColor = gl_FrontColorIn[i];
    vec4 p = gl_PositionIn[i];
    p.y = -p.y;
    gl_Position = gl_ModelViewProjectionMatrix * p;
    EmitVertex();
}
EndPrimitive();
}

```

### Exercici 3 (1 punt)

La següent figura mostra un quadrat texturat que ocupa tot un viewport de 1024x1024 pixels.

- (a)  $\partial s / \partial x = 1/1024$
- (b)  $\partial t / \partial y = 1/1024$

### 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?

Ambient occlusion.

- (b) Per què?

Donat que no hi ha cap estructura de dades auxiliar, el calcul d'intersecció raig-escena ha de recórrer els objectes fins trobar una intersecció (ambient occlusion) o recórrer tots els objectes per calcular la intersecció més propera al punt (obscuràncies).

### Exercici 5 (1 punt)

- (a) Quan es pot produir el fenomen de reflexió total?

Quan la llum travessa la superfície que separa un medi dens (ex. vidre) d'un menys dens

(ex. aire)

(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?

Les equacions de Fresnel

(c) Explica què és l'angle crític

Angle d'incidència a partir del qual no hi ha transmissió, només reflexió.

(d) Indica quin valor de l'angle d'incidència maximitza la proporció de llum transmesa quan aquesta passa de l'aire al vidre.

$$\Theta_i = 0$$

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);
```

(a) Quina crida OpenGL (relacionada amb les matrius de transformació geomètrica) manca?

Cal afegir una crida a glMultMatrix(matriu simetria)

(b) On caldria afegir-la (ho pots indicar al codi amb una fletxa)

Immediatament després de glPushMatrix.

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?

La crida glBufferData s'ha de fer un cop, no cada frame!

Exercici 8 (1 punt)

Casos d'ús de dFdx() i dFdy():

- (a) Emular glPolygonOffset: **UTIL**
- (b) Càcul manual del LoD en mipmapping: **UTIL**
- (c) Component espectral d'il·luminació: **NO UTIL**
- (d) Derivades parcials en vertex shader: **NO UTIL**

Exercici 9 (1 punt)

Completa aquest codi que implementaombres per projecció amb stencil buffer:

```
// 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 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(occlusor);  
glPopMatrix();
```

```

// 3. Dibuixa la part fosca del receptor
glEnable(GL_DEPTH_TEST);
glDepthFunc(GL_LESS);
glColorMask(GL_TRUE, ... , GL_TRUE);
glDisable(GL_LIGHTING);
glStencilFunc(GL_EQUAL, 0, 1);
Dibuixa(receptor);

// 4. Dibuixa l'occlusor
glEnable(GL_LIGHTING);
glDepthFunc(GL_LESS);
glDisable(GL_STENCIL_TEST);
Dibuixa(occlusor);

```

**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                | 1 |
| (b) GL_LINEAR                 | 1 |
| (c) GL_NEAREST_MIPMAP_NEAREST | 1 |
| (d) GL_LINEAR_MIPMAP_LINEAR   | 2 |

Tria l'espai de coordenades en que ha d'estar P per tal que la transformació  $\text{projectionMatrixInverse} * \mathbf{P}$  tingui sentit

Clipping

Select one:

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

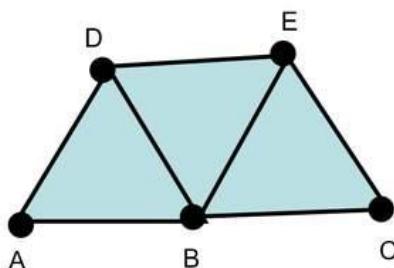


Assigna a cada crida/tasca l'ordre relatiu (1,2,3,4) en que s'executa en un pipeline d'OpenGL sense GS:

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:



No col posor backface

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

y & [-2, 2]  
x & [-11, 11]

} >> (davant càmera)

Indica el valor que retorna aquesta expressió GLSL:

mix(9, 3, 0.7)

Answer:

Correct answers: [4.8]

$$\begin{aligned} & \left[ \begin{array}{l} 3/9 \\ 0/3 \end{array} \right] \rightarrow 3 + 0,3 \cdot (9-3) = \\ & = 3 + 1,8 = 4,8 \end{aligned}$$

Indica el valor que retorna aquesta expressió GLSL:

mod(7.9, 4)

$$7,9 - 4 = 3,9$$

Answer:

Correct answers: [3.900000000000004]

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))))



} centre (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):

depth Test (0,1)

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:

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

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

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

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

No veïm  $\gamma \rightarrow$  reflection  $\gamma$

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

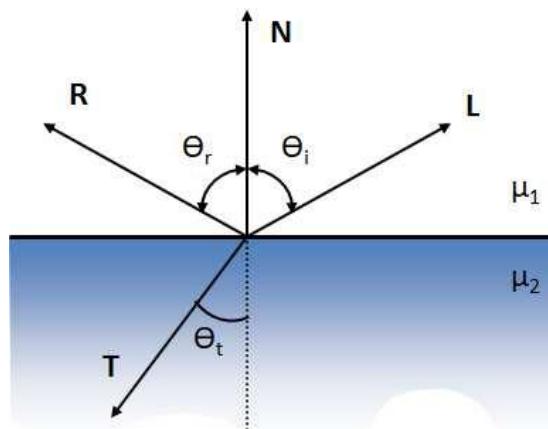
- |   |
|---|
| 1 |
| 2 |
| 3 |
| 4 |

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:



$$\frac{\mu_1}{\mu_2} < 1 \Rightarrow \mu_1 < \mu_2$$

Select one:

(1.09, 1.43)

(1.43, 1.09)

(0.09, 1.43)

(1.09, 0.43)

} de tercere matriu més pesada

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:

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

et

**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

document

### Exercici 3, 4, 5 i 6

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

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

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

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

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

$$P * V * M$$

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

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

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

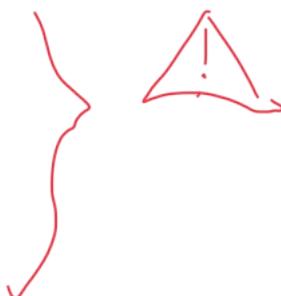
g) Pas de la normal de model space a eye space  $N$

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

### Exercici 7

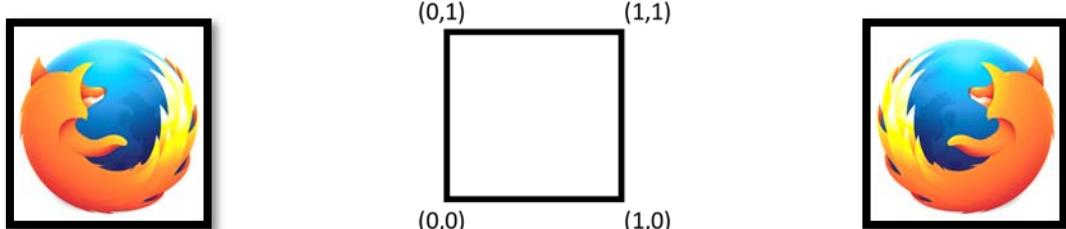
Indica, per cadascuna de les següents tècniques basades en textures, si sempre requereixen (SI) o no (NO) accedir a un *height field*:

- |                          |    |
|--------------------------|----|
| (a) Color mapping        | NO |
| (b) Relief mapping       | SI |
| (c) Parallax mapping     | SI |
| (d) Displacement mapping | SI |



### Exercici 8

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



Completa el següent VS per obtenir el resultat desitjat:

```
void main() {  
  
    vtexCoord = vec2(-1,1)*texCoord;  
  
    glPosition = vec4(vertex, 1.0);  
}
```

/

### Exercici 9

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

Bump mapping / Normal mapping

### Exercici 10

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

(a) LSSDSSE SI

(b) LDE SI

(c) LSE SI

(d) LDDSE NO

$L(S^*D^*E)$   
llum cara (two-pass)

?

### 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?

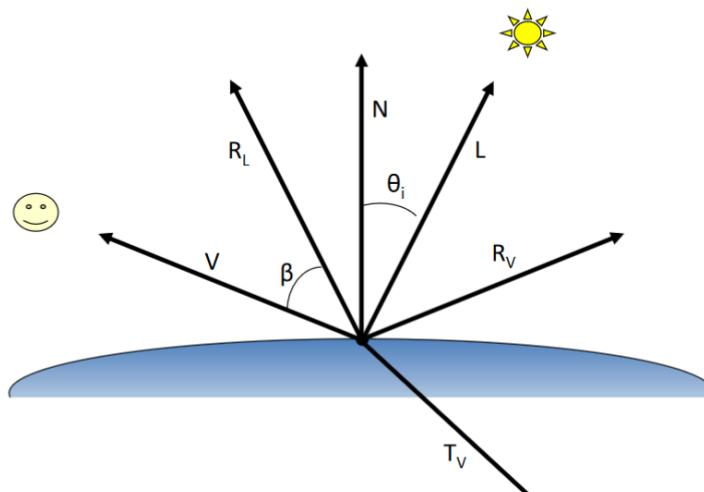
R<sub>v</sub>

- (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).

### Exercici 15

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

```
funció traçar_raig(raig, escena, μ)
    si profunditat_correcta() llavors
        info:=calcula_interseccio(raig, escena)
        si info.hi_ha_interseccio() llavors
            color:=calcular_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];
[red box]

    float storedDepth = texture(shadowMap, st).r;
[red box]

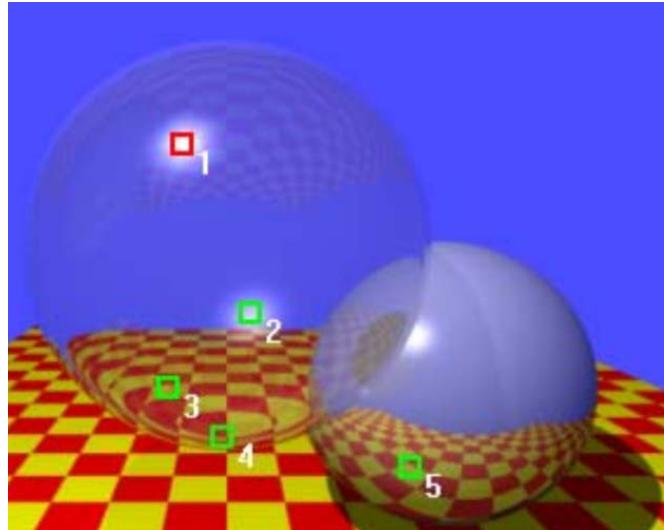
    float trueDepth = vtexCoord.p / vtexCoord.q;
[red box]

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

### Exercici 17

Considerant aquesta figura generada amb ray-tracing,

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



### Exercici 18

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

- a) Shadow mapping, suposant que la llum és dinàmica

2 cops

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

2 cops

**Gràfics 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.

- 3 - Fragment Shader
- 1 - Geometry shader
- 2 - Rasterització
- 4 - 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

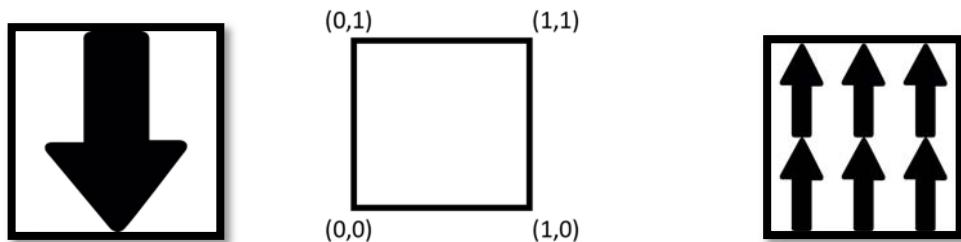
Directament  $F(u,v)$

- (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.

### Exercicis 6, 7, 8 i 9

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

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

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

$$\text{LDS}*E, \text{LS}*E$$

(b) Path tracing

$$\text{LS}*DS*E$$

### 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*?

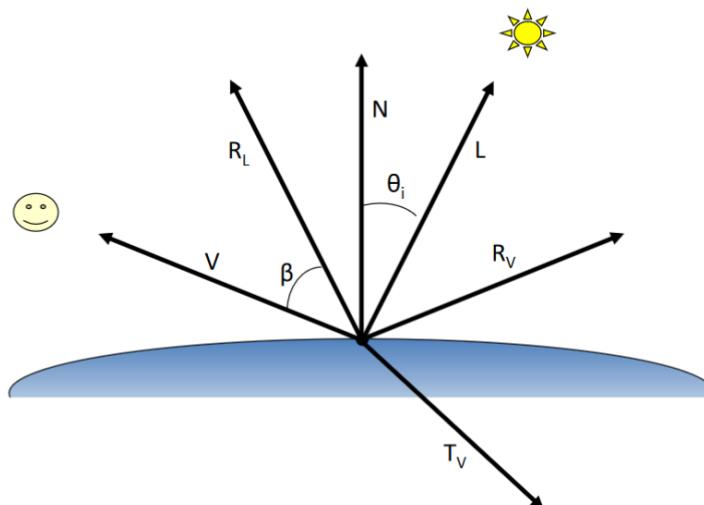
L

- (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

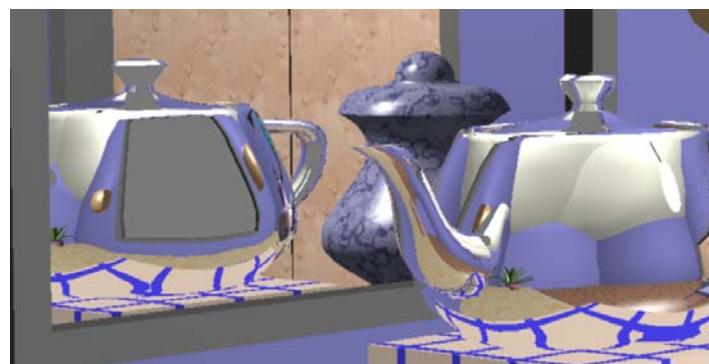


### Exercici 13

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.

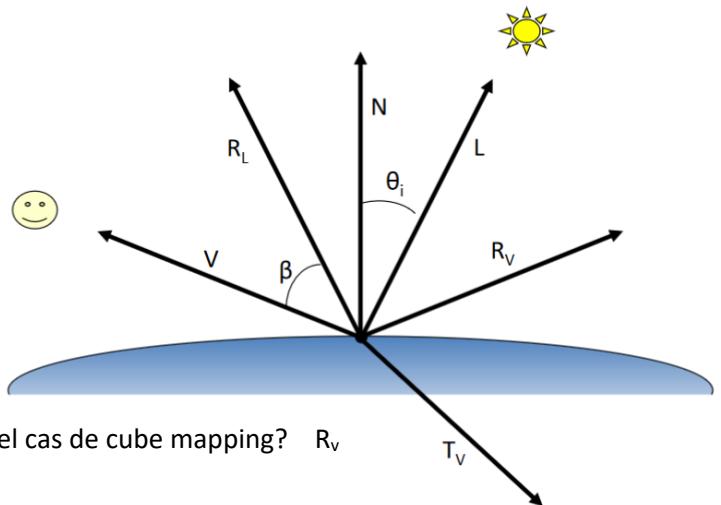
### Exercici 15

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:

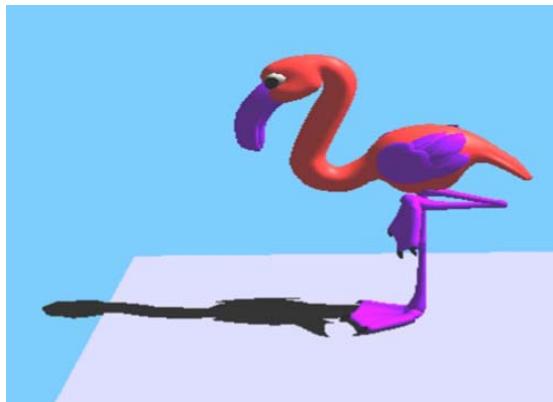


(a) Quin vector cal usar per indexar un cube map, en el cas de cube mapping?  $R_v$

(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?

$$\begin{bmatrix} -d & 0 & 0 & 0 \\ 0 & -d & 0 & 0 \\ 0 & 0 & -d & 0 \\ a & b & c & 0 \end{bmatrix}$$

- (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

## Gràfics 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)**

#### 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 demandada, **usant la notació següent** (vigileu amb l'ordre en que multipliqueu les matrius):

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

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

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       $P * V * M$

### **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  $\text{RGBA}$  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 RGB  $(0.5, 0.2, 0.3)$ . Justifica la resposta.

**glBlendFunc(...**

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

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

→ **glBlendFunc(GL\_SRC\_ALPHA, GL\_ONE)**

### **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(occlusor);
glPopMatrix();
```

## 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...)

## Gràfics 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

glDrawElements  
Transformació a   
Space Divisió de  
perspectiva 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

ANTERIOR

(b)	Fragment	 der	
POSTERIOR	(c)	dFdx,	dFdy
POSTERIOR	(d)	Stencil	Test
POSTERIOR			

### Exercici 3

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



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

## 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?

$12 \text{ tri} * 3 \text{ v/tri} = 36$   vèrtexs (hem de repetir vèrtexs perquè no comparteixen normals)

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

12 tri \* 3 index = 36 indexes

## Exercicis 5 i 6

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

M = modelMatrix V

$M^{-1} = \text{modelMatrixInverse}$

= viewingMatrix P =

$V^{-1} = \text{viewingMatrixInverse}$

projectionMatrix N

$$P^{-1} =$$

= normalMatrix

## projection

Ident

P

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

 V<sup>-1</sup>

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

V<sup>-1</sup> \* P<sup>-1</sup>

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

P \* V

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

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

M

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

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

N

## Exercici 7

Tenim activat alpha blending amb la

crida `glBlendFunc(GL_ONE_MINUS_SRC_ALPHA, GL_SRC_ALPHA)`

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

$$(1-0.2) (1.0, 0.5, 0.0, 0.2) + 0.2 (1.0, 1.0, 0.5, 0.0) =$$

$$(0.8, 0.4, 0.0, 0.16) + (0.2, 0.2, 0.1, 0) = \boxed{(1.0, 0.6, 0.1, 0.16)}$$

## 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  $\lambda$  ?

Longitud d'ona de la llum

(b) Què representa  $\Omega$  ?



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  $\mathbf{x}$ .

(c) Indica quin nom té la funció  $f_r$

És el BRDF.

(d) Què són els tres primers paràmetres de la funció  $f_r$ ?

Els paràmetres són: posició, direcció d'entrada, direcció de sortida.

### 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 = .......pow( RdotV, matShininess );
}

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

$LDS^*E$  i  $LS^*E$



(b) Two-pass raytracing

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

## 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).

a) Quants rajos primaris caldrà traçar?

$$256 \times 256 = 2^{16} \text{ rafas.}$$

b) Quants rajos reflectits caldrà traçar, en total, si admetem un únic nivell de recursivitat (per exemple LDSE)?

$$= 2^{15} = 32768$$



c) Quants rajos reflectits caldrà traçar, en total, si admetem dos nivells de recursivitat (per exemple LDSSE)?

$$2^{15} + 2^{14} = 49152$$

d) Quants rajos primaris caldrà traçar si volem antialiàsing amb 4 mostres per píxel?

$$4 \times 256 \times 256 = 2^{18} \text{ rafas.}$$

## 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  $\text{W/m}^2$ )?

### Irradiância



## 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):

- (a) Velocitat de propagació de la llum als medis SI

(b) Longitud d'ona de la llum SI

(c) Angle d'incidència SI

(d) Color difós de la superfície (Kd) NO

### 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

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.



Per tal que els objectes virtuals (reflectits) apareguin només a la porció del pla ocupada pel mirall.

### Exercici 18

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



Completa la línia que necessitem al VS:

```
vtxCoord = 4.0 * texCoord
```

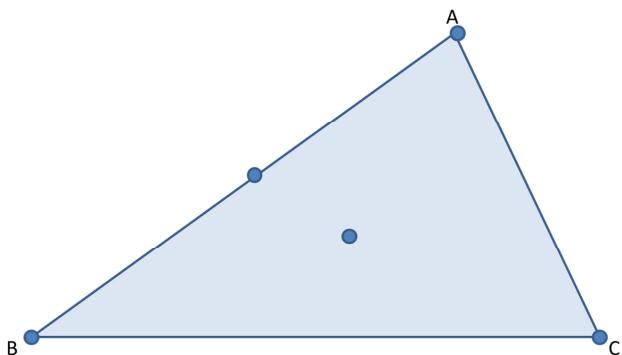


### Exercici 19

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

(a) Bicentre del triangle

(1/3, 1/3, 1/3)



(b) Punt mig de l'aresta AB

(0.5, 0.5, 0)



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

$\gamma = 0.5$  (han de sumar 1)

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

P és exterior 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 = 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);
}
```

**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

- glDrawElements
- Escriptura de gl\_Position
- Divisió de perspectiva
- 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 | ANTERIOR  |
| (b) Fragment Shader | POSTERIOR |
| (c) Stencil Test    | POSTERIOR |
| (d) Alpha blending  | POSTERIOR |

**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)?

$$1 + \log_2(512) = 10 \quad (512 \times 512, 256 \times 256, 128 \times 128, 64 \times 64, 32 \times 32, 16 \times 16, 8 \times 8, 4 \times 4, 2 \times 2, 1 \times 1)$$

#### 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?

$8 \text{ tri} * 3 \text{ v/tri} = 24 \text{ vèrtexs}$  (hem de repetir vèrtexs perquè no comparteixen normals)

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

$8 \text{ tri} * 3 \text{ indexs/tri} = 24 \text{ indexs}$

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

$24 \text{ v} * 3 \text{ coord/v} * 4 \text{ bytes/coord} = 288 \text{ bytes}$

#### Exercicis 5 i 6

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

$$\begin{array}{ll} M = \text{modelMatrix} & M^{-1} = \text{modelMatrixInverse} \\ = \text{viewingMatrix} P = & = \text{viewingMatrixInverse} P^{-1} = \\ \text{projectionMatrix} N & \text{projectionMatrixInverse} I = \\ = \text{normalMatrix} & \text{Identitat} \end{array}$$

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

b) Pas d'un vèrtex de object space a eye space  $V * M$

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

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

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

B) Pas d'un vèrtex de world space a clip space  $P * M$

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

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

### 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)?

D'acord amb la component G:  $a \cdot 0.5 + (1-a) \cdot 1.0 = 0.8 \rightarrow a=0.4$

### Exercici 8

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

Global → llum directa + llum indirecta

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

**vec3 L = normalize( lightPosition.xyz - P );**

```
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. LDS\*E i LS\*E

### 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?  $1024 \times 1024 = 2^{20}$  rajos.
- b) Quants shadow rays caldrà traçar, en total?  $2^{20}$  hits \* 4 shadow rays/hit =  $2^{22}$  shadow rays
- c) Quants rajos reflectits caldrà traçar, en total? Cap
- d) Quants rajos transmesos caldrà traçar, en total? Cap

### 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 → B

2 → A

3 → D

4 → C

### Exercici 13

Completa l'equació general del rendering:

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

### 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)?

Equacions de Fresnel

### 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      true
- (b) depthTestTranslucent      true
- (c) maskOpaque      true
- (d) maskTranslucent      false

### 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

$$M = \text{modelMatrix} \text{ (original)} \quad M^{-1} = \text{modelMatrixInverse} \text{ (original)}$$

$$V = \text{viewingMatrix} \quad V^{-1} = \text{viewingMatrixInverse}$$

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

$$V^*R^*M$$

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

$$R^*M$$

### 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 ocipi tot el viewport, amb independència de la càmera. Completa aquesta línia del VS demandat:

```
layout(location = 3) in vec2 texCoord;
```

```
gl_Position = vec4(  texCoord.s,      texCoord.t,      0.0,      1.0);
```

## **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?

Environment mapping

(b) En quin espai de coordenades està treballant?

Eye space; en cas contrari  $I = \text{normalize}(\text{Pos}-\text{Obs})$

## **Exercici 19 i 20**

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

a) Shadow mapping, suposant que la llum és dinàmica

2 cops

b) Ombres per projecció, versió sense stencil

2 cops

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

2 cops

d) Environment Mapping, amb una textura fixa

1 cop

## 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

Proporciona el color i la opacitat a partir del valor del voxel.

### **Pregunta 2**

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

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 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

- Escritura de gl\_Position
  - Retallat de primitives (clipping)
  - Rasterització
  - 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                  | ANTERIOR  |
| (b) Processament del fragment           | POSTERIOR |
| (c) Ús de les funcions dFdX, dFdY       | POSTERIOR |
| (d) Pas de les coordenades a clip space | ANTERIOR  |

**Exercici 3**

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

Càlculs d'il·luminació; generació de coordenades de textura (sphere mapping...), ...

#### 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)?

$$0.8 \cdot 1.0 + 0.2 \cdot 1.0 = 1.0$$

$$0.8 \cdot 0.5 + 0.2 \cdot 0.5 = 0.5$$

$$0.8 \cdot 0.0 + 0.2 \cdot 1.0 = 0.2$$

$$0.8 \cdot 0.8 + 0.2 \cdot 1.0 = 0.84 \rightarrow (1.0, 0.5, 0.2, 0.84)$$

#### Exercici 5

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

$$\text{MV} = \text{gl_ModelViewMatrix} \quad \text{MVP} = \text{gl_ModelViewProjectionMatrix}$$

$$\text{MV}^{-1} = \text{gl_ModelViewMatrixInverse} \quad \text{MVP}^{-1} = \text{gl_ModelViewProjectionMatrixInverse}$$

$$\text{NM} = \text{gl_NormalMatrix} \quad \text{I} = \text{Identitat}$$

- a) Pas d'un vèrtex de eye space a world space  $\text{MV}^{-1}$
- b) Pas d'un vèrtex de clip space a object space  $\text{MVP}^{-1}$
- c) Pas de la normal de object space a eye space  $\text{NM}$
- d) Pas d'un vèrtex de object space a world space  $\text{I}$

#### 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).

```
gl_ModelViewProjectionMatrixInverse * P;
```

->

```
gl_ModelViewMatrixInverse * gl_ProjectionMatrixInverse * P;
```

### 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
```

```
    vec3 L = normalize( light.position.xyz - P );
```

```
    float diff = max( 0.0, dot( N,L ) );
```

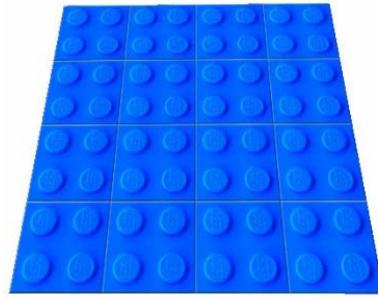
```
    return gl_FrontMaterial.diffuse * light.diffuse *
```

```
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 = 4.0 * gl_MultiTexCoord0.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)?

$$1 + \log_2(256) = 9 \quad (256 \times 256, 128 \times 128, 64 \times 64, 32 \times 32, 16 \times 16, 8 \times 8, 4 \times 4, 2 \times 2, 1 \times 1)$$

### 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  $\frac{\partial s}{\partial x}$

Es maximitza a l'aresta superior del polígon.

### 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 demandat:

```
gl_Position = vec4( gl_TexCoord[0].s, gl_TexCoord[0].t, 0.0, 1.0);
```

### 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 ( P.x/P.w > 0 ) discard;
```

b) Si **P** està en **NDC**

```
if ( P.x > 0 ) discard;
```

c) Si **P** està en **window space**

```
if ( P.x > 400 ) discard;
```

### Exercici 13

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

```
Y=0.1  0  0  0  
 0  -1 0  0  
 0  0  1  0  
 0  0  0  1
```

### **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?

6 cares \* 4 vèrtexs per cada cara = **24 vèrtexs**

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?

**8 vèrtexs (mínim)**

### **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.

Simplement hem de passar la normal ( $nx', ny', nz'$ ) de tangent space a eye space:  $[T \ B \ N] * (nx', ny', nz')$  on  $[T \ B \ N]$  és una matriu  $3 \times 3$  que té per columnes els tres vectors T, B, N.

### **Exercici 16**

Quins són els sistemes de coordenades origen i destí de la transformació representada per la `gl_ModelViewMatrix` en GLSL?

Object Space --> Eye Space

### **Exercici 17**

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

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

Quin efecte té aquesta funció?

Activar (true) / desactivar (false) l'escriptura en el depth buffer.

### **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))`

float

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

vec3

### **Exercici 19**

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

- a) Shadow mapping, suposant que la llum és dinàmica                    2 cops
- b) Ombres per projecció, versió sense stencil                    2 cops
- c) Reflexió amb objectes virtuals, amb stencil (ignoreu els passos en que només es dibuixa el mirall): 2 cops
- d) Projective Texture Mapping, amb una textura fixa                    1 cop

### **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?

Cap; en aquest cas, la càmera coincideix amb la llum, i per tant les ombres no seran visibles.

CT, MR, ultrasons...

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

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

### Exercici 2

Hem produït un fragment amb color  $\text{RGBA} = (1.0, 0.5, 0.0, 0.4)$  corresponent al pixel  $(i,j)$ . El color  $\text{RGBA}$  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)?

$$0.4 \cdot 1.0 + 0.6 \cdot 0.5 = 0.7$$

$$0.4 \cdot 0.5 + 0.6 \cdot 0.5 = 0.5$$

$$0.4 \cdot 0.0 + 0.6 \cdot 1.0 = 0.6$$

$$0.4 \cdot 0.4 + 0.6 \cdot 1.0 = 0.76 \rightarrow (0.7, 0.5, 0.6, 0.76)$$

### Exercici 3

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

$$\text{MV} = \text{gl_ModelViewMatrix}$$

$$\text{MVP} = \text{gl_ModelViewProjectionMatrix}$$

$$\text{MV}^{-1} = \text{gl_ModelViewMatrixInverse}$$

$$\text{MVP}^{-1} = \text{gl_ModelViewProjectionMatrixInverse}$$

$$\text{NM} = \text{gl_NormalMatrix}$$

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

b) Pas d'un vèrtex de eye space a world space     $\text{MV}^{-1}$

c) Pas d'un vèrtex de clip space a object space     $\text{MVP}^{-1}$

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

#### 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).

```
gl_Position = gl_ModelViewProjectionMatrix * gl_Vertex;  
->  
gl_Position = gl_ProjectionMatrix * gl_ModelViewMatrix * gl_Vertex;
```

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

#### Exercici 6

Per què ens pot ser útil aquesta sentència GLSL?

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

--> Per obtenir la posició del observador, en object space (o world space, si no hi ha transf de modelat).

#### Exercici 7

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

$$1024 \times 1024 + 512 \times 512 + 256 \times 256 + \dots 1 \times 1 = | | (2^{10}, 2^9, \dots, 2^0) | |^2 = 1.33 \text{ M texel}$$

### 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?

Vector L: vector unitari del punt cap a l'observador

### 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?

Irradiància

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

(W/m<sup>2</sup>,lux)

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

### 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

T(centre)\*T(trans)\*R\*S\*T(-centre)

### **Exercici 12**

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

Bàsicament un VBO és un VA emmagatzemat en memòria del servidor OpenGL (ex. GPU).

Per tant, estalvi en les dades que han de passar-se cada frame de RAM a memòria de la GPU.

### **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.

Simplement hem de passar la normal ( $nx', ny', nz'$ ) de tangent space a eye space:  $[T \ B \ N] * (nx', ny', nz')$  on  $[T \ B \ N]$  és una matriu 3x3 que té per columnes els tres vectors T, B, N.

### **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?

Eye Space --> Clip Space

### **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.

$-w \leq x \leq w$  (el mateix per y,z)

### **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?

Activar (true) / desactivar (false) l'escriptura en els buffers de color.

b) Menciona una tècnica concreta (dintre de les tècniques per simular ombres, reflexions, transparències...) en la qual es faci servir, i perquè.

Ombres per projecció amb stencil, als passos on només volem modificar el depth buffer o el stencil buffer, però no pas el color buffer.

### **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 espectral del material

float shininess = gl\_FrontMaterial.shininess

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

vec4 posLlum = gl\_LightSource[0].position

### **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))

float

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

vec4

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

vec3

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

vec3

### **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(...)
```

Està definint la matriu de conversió world space -> texture space; es fa servir a proj texture mapping.

### **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?

Aliasing greu a l'ombra degut al problema "duelling frustra".

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

### **Pregunta 1**

A què fa referència el terme *haptic*?

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

### **Pregunta 2**

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

CT, MR, ultrasons...

Nom i Cognoms:

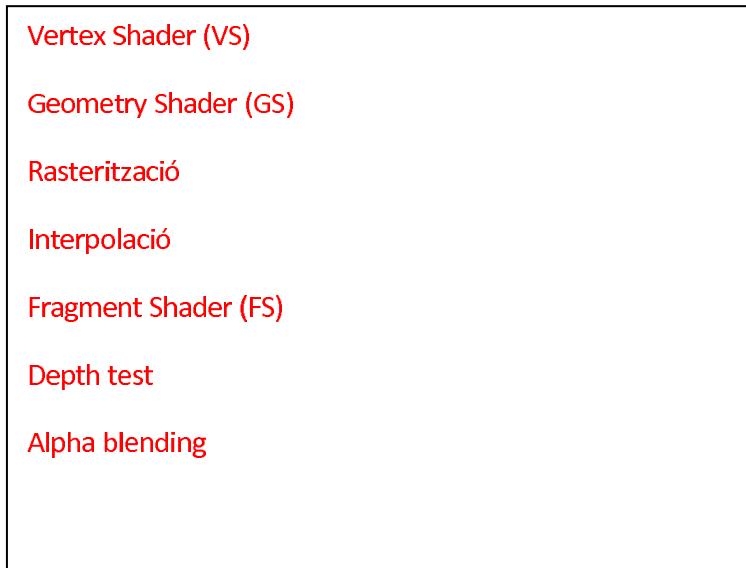
---

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



### Exercici 2

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

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

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

### Exercici 3

Hem tingut un petit accident amb el tablet i s'ha trencat la pantalla. Afortunadament, la meitat superior de la pantalla encara és utilitzable. De cara a utilitzar aplicacions OpenGL (a pantalla completa), una opció és definir un viewport més petit que 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);  
  
    // cal traslladar i escalar en Y  
  
    P.y += 1.0;  
  
    P.y /= 2.0;  
  
    gl_Position = P;  
}
```



### Exercici 4

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

```
varying vec3 normal; // eye space  
varying vec3 pos; // pos en eye space  
  
vec4 lightSource( vec3 N, vec3 V, gl_LightSourceParameters light ) {  
    vec3 L = normalize( light.position.xyz - V );  
    vec3 R = normalize( 2.0*dot(N,L)*N-L );  
    V = normalize ( -V.xyz );  
    float diff = max( 0.0, dot( N,L ) );  
    float spec; // cal que el calculeu vosaltres  
  
    float RdotV = max( 0.0, dot( R,V ) );  
  
    spec = pow( RdotV, gl_FrontMaterial.shininess );  
  
  
    return gl_FrontMaterial.diffuse * light.diffuse * diff +  
           gl_FrontMaterial.specular * light.specular * spec;  
}
```

## Exercici 5

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

Alpha blending; Exemple:

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

## Exercici 6

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

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

## Exercici 7

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

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

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

### Exercici 8

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

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

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

void main()
{
    vec3 R;

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

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

### Exercici 9

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

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

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

### **Exercici 10**

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

NO

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

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

### **Exercici 11**

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

LDE, LSE

### **Exercici 12**

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

irradiància

Indica una unitat de mesura per la magnitud anterior

W/m<sup>2</sup> lux

### **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?

Les equacions de Fresnel

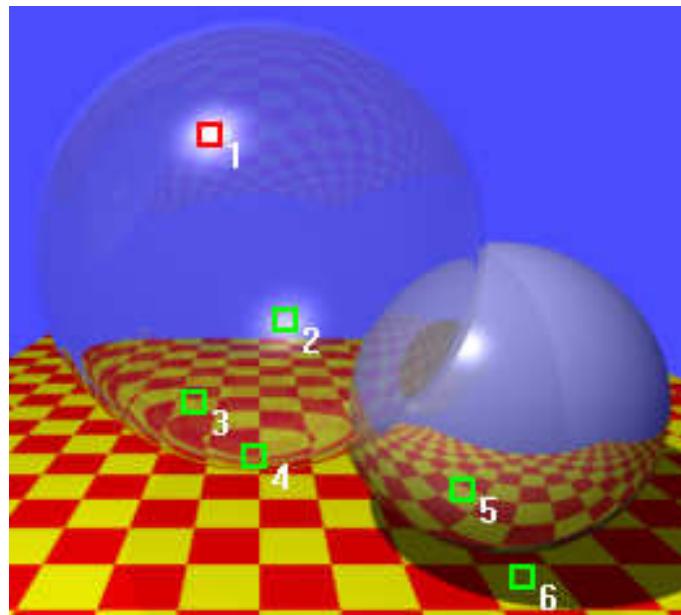
### **Exercici 14**

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

$\text{dot}(\mathbf{v}, \mathbf{n}) = 0 \rightarrow \text{raig paral·lel al pla} \rightarrow \text{no intersecta}$

### Exercici 15

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



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

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

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

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

### Pregunta 1

A què fa referència el terme *haptic*?

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

### Pregunta 2

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

CT, MR, ultrasons...

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)
{
    return normalize( light.position.xyz - vertex );
}
```

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

```
// Vertex shader
varying vec3 normal;

void main() {
    normal = gl_NormalMatrix * gl_Normal;
    gl_Position = gl_ModelViewProjectionMatrix * gl_Vertex;
    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 = texture2D( sampler, gl_TexCoord[0].st);
}
```

## 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).

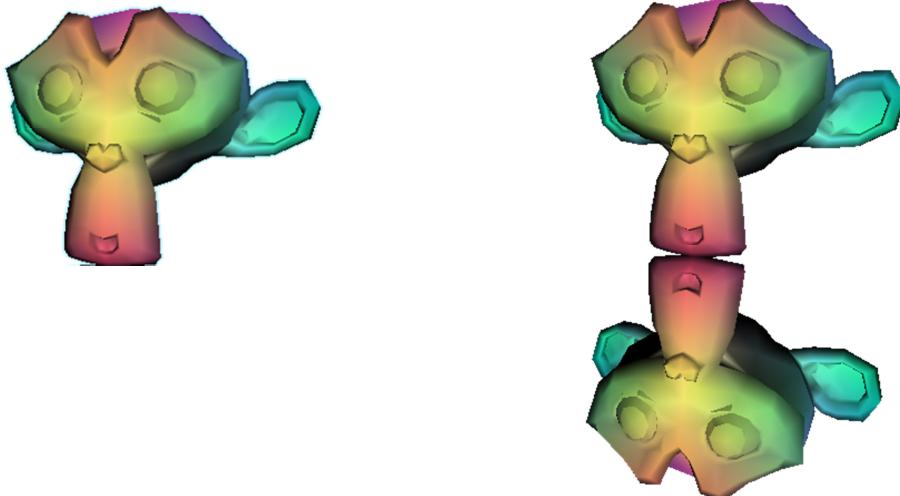


Figura 1. Resultat amb els shaders que us proporcionem (esquerra) y resultat desitjat (dreta).

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

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

**Geometry shader** (a completar):

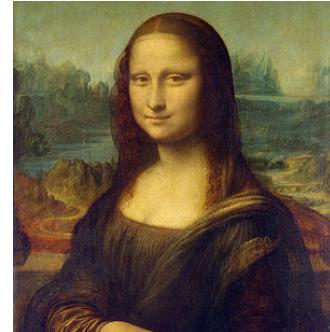
```
void main( void )
{
    for( int i = 0 ; i < 3 ; i++ )
    {
        gl_FrontColor = gl_FrontColorIn[ i ];
        gl_Position   = gl_ModelViewProjectionMatrix * gl_PositionIn[i];
        EmitVertex();
    }
    EndPrimitive();
    for( int i = 2 ; i >=0 ; i-- )
    {
        gl_FrontColor = gl_FrontColorIn[ i ];
        vec4 p = gl_PositionIn[i];
        p.y = -p.y;
        gl_Position = gl_ModelViewProjectionMatrix * p;
        EmitVertex();
    }
    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)  $\frac{\partial s}{\partial x} = \textcolor{red}{1/1024}$

(b)  $\frac{\partial t}{\partial y} = \textcolor{red}{1/1024}$



### 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?

**Ambient occlusion.**

- (b) Per què?

Donat que no hi ha cap estructura de dades auxiliar, el calcul d'intersecció raig-escena ha de recórrer els objectes **fins trobar una intersecció** (ambient occlusion) o recórrer **tots els objectes** per calcular la intersecció més propera al punt (obscuràncies).

### Exercici 5 (1 punt)

- (a) Quan es pot produir el fenomen de reflexió total?

**Quan la llum travessa la superfície que separa un medi dens (ex. vidre) d'un menys dens (ex. aire)**

- (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?

**Les equacions de Fresnel**

- (c) Explica què és l'angle crític

**Angle d'incidència a partir del qual no hi ha transmissió, només reflexió.**

- (d) Indica quin valor de l'angle d'incidència maximitza la proporció de llum transmesa quan aquesta passa de l'aire al vidre.

$$\Theta_i = 0$$

### **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);
```

- (a) Quina crida OpenGL (relacionada amb les matrius de transformació geomètrica) manca?

Cal afegir una crida a **glMultMatrix(matriu simetria)**

- (b) On caldria afegir-la (ho pots indicar al codi amb una fletxa)

Immediatament després de **glPushMatrix**.

### **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?

La crida `glBufferData` s'ha de fer un cop, no cada frame!

### **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 espectral de la il·luminació.
- (d) Per calcular qualsevol derivada parcial en un vertex shader

### 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(GL_ALWAYS, 1, 1);  
glStencilOp(GL_KEEP, GL_KEEP, GL_REPLACE);  
dibuixa(receptor);  
  
// 2. Dibuixa oclusor per netejar 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(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(GL_EQUAL, 0, 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                | 1 |
| (b) GL_LINEAR                 | 1 |
| (c) GL_NEAREST_MIPMAP_NEAREST | 1 |
| (d) GL_LINEAR_MIPMAP_LINEAR   | 2 |