## Curs 2019-20 Q1

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

1. Geometry shader

2. Rasterització

3. Fragment shader

4. Alpha Blending

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

1. setUniformValue

2. Vertex Shader

3. Depth Test

4. Stencil Test

**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

**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

**5.- Què coordenada del fragment modifica la funció glPolygonOffset? Z**

En quin espai la modifica? Window space

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

NOSE

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

(a) (0.2, ‐1.5, 1.8, 2) en clip space

DINS (tot entre +/‐ w)

(b) (0, 0, 10, 1) en object space

DINS (pot ser...)

**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);

**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;

**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

**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, \_\_\_\_\_\_\_\_vec2(1,0.5)\_\_\_\_\_\_\_\_\_\_ \* vtexCoord);

**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

M ‐> model matrix (of the object)

P ‐> projection matrix (of the light camera)

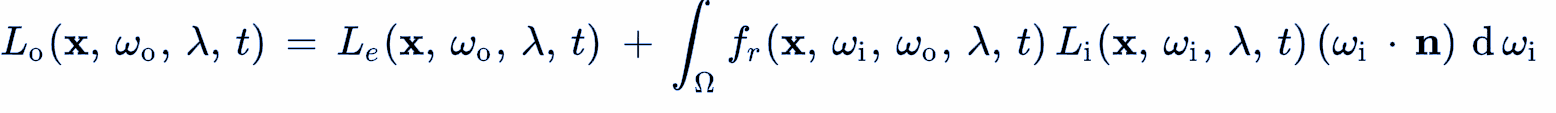
T(tx,ty,tz) ‐> Translate matrix

V ‐> view matrix (of the light camera)

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

T(0.5)S(0.5)PV

**13.- A l’equació general del rendering:**

****

**Què representa Ω?**

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

**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

projectionMatrix

modelViewProjectionMatrix

modelMatrixInverse

projectionMatrixInverse

modelViewProjectionMatrixInverse

projectionMatrixInverse \* modelViewProjectionMatrix

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

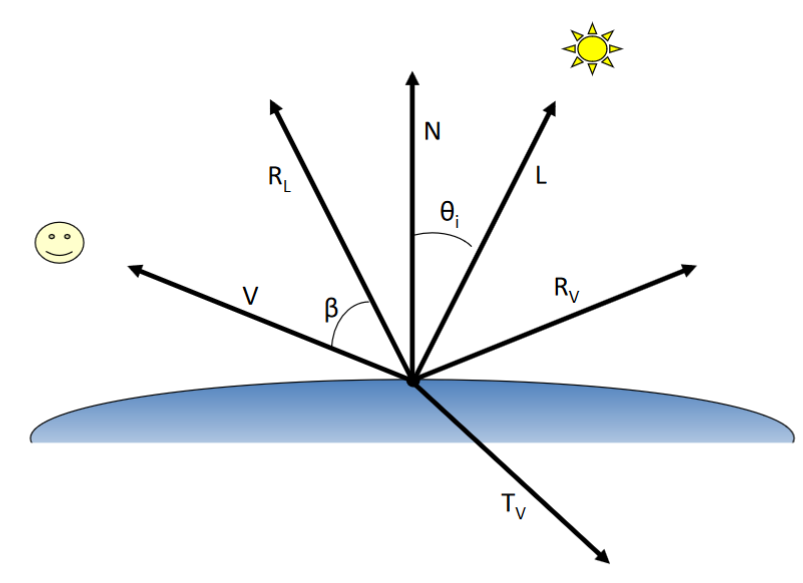
(a) Segon paràmetre d’una crida a texture(colormap...) vec2

(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

**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

**17.- Continuant amb la figura anterior…**

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

V

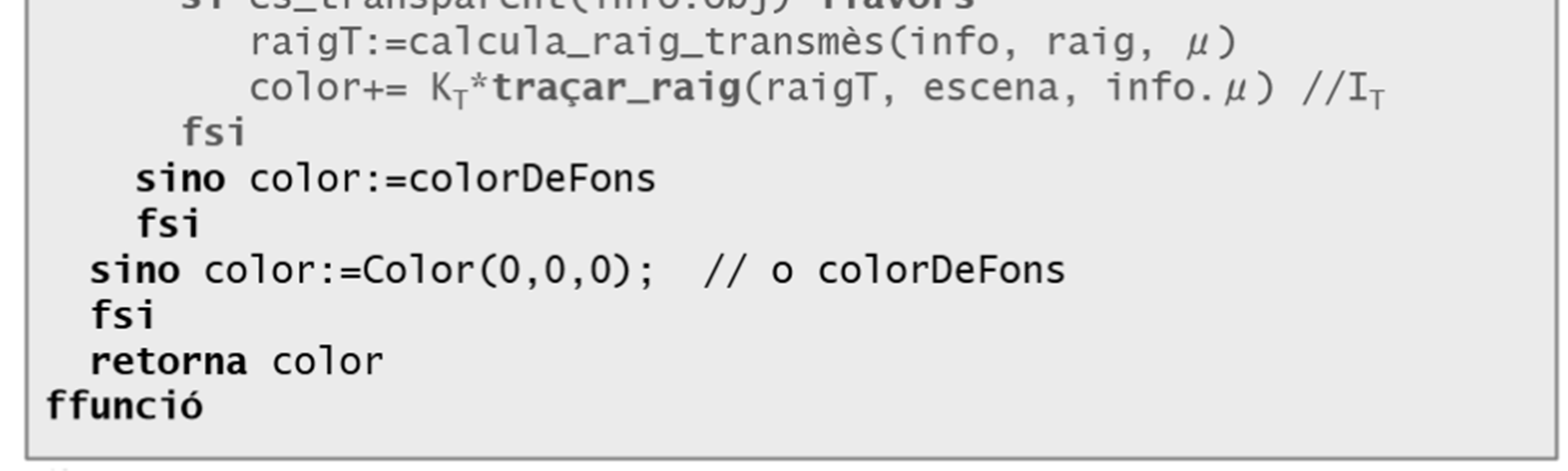
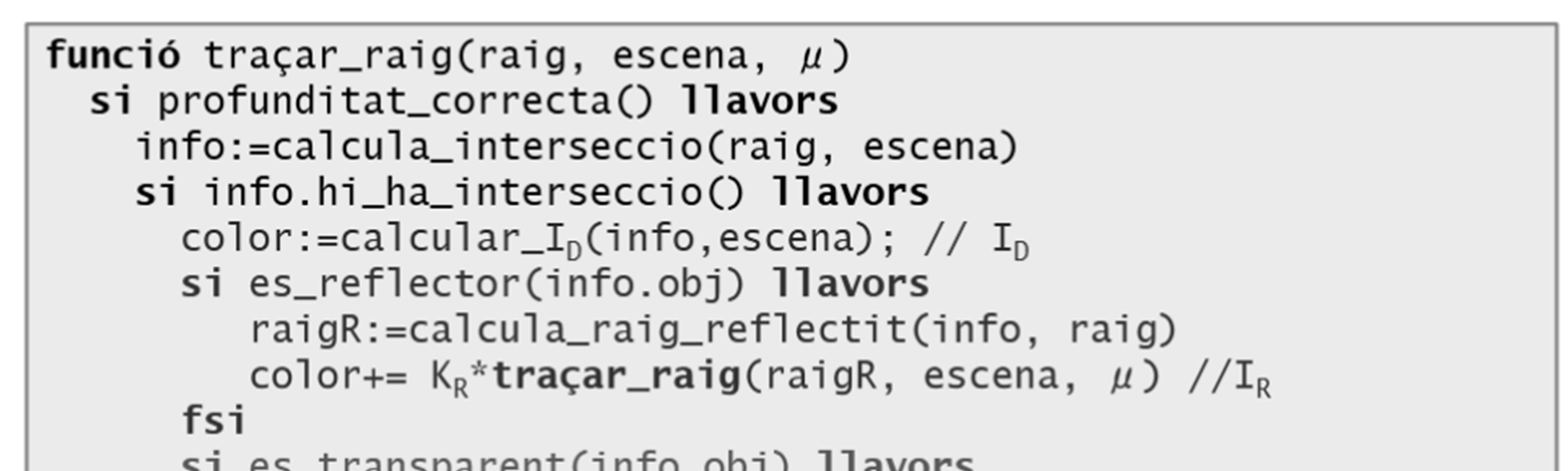
(b) Quins dos vectors determinen la contribució local de Phong?

RL i V

**18.- Quines són les unitats de la radiància (radiometria) en el Sistema Internacional?**

W/(m^2 \* sr)

**19.- Completa, a sota, el codi que falta.**



**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);

}

## Curs 2018-19 Q2

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

1. glDrawElements

2. Vertex Shader

3. Rasterization

4. Alpha Test

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

1. setUniformValue
2. Geometry Shader
3. Rasterization
4. Depth Test

**Exercici 3**

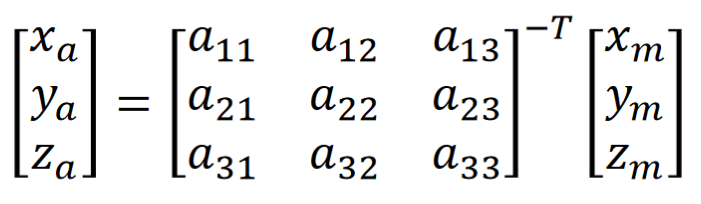
**Escriu quin és l’espai de coordenades inicial i final de la multiplicació de la projection matrix per un vèrtex.**

Inicial: Eye space

Final: Clip space

**Exercici 4**

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

****

vector normal

**Exercici 5**

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

1. Pas a NDC ABANS
2. Geometry Shader ABANS
3. Divisió de perspectiva ABANS
4. 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:**