

grafics.pdf



ipdleon



Gráficos



3º Grado en Ingeniería Informática



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

EAE Business School
Barcelona

MÁSTER EN PROJECT MANAGEMENT

Convocatoria Abril 2023

eaebarcelona.com

Work
to change
your life

Elige tu propio camino
y empieza a cambiar lo
que tú quieras cambiar.



We make
it happen

Te regalamos



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

BBVA está adherido al Fondo de Garantía de Depósitos de Entidades de Crédito de España. La cantidad máxima garantizada es de 100.000 euros por la totalidad de los depósitos constituidos en BBVA por persona.

1

Abre tu Cuenta Online sin comisiones ni condiciones

2

Haz una compra igual o superior a 15€ con tu nueva tarjeta

3

BBVA te devuelve un máximo de 15€

Apunts de Gràfics

Pol Casacuberta

Contents

1 Sistemes de coordenades	4
1.1 Punts	4
1.2 Vector	6
1.3 Vectors normals	8
2 Objectes translúcids	9
2.1 Equacions de Fresnel a.k.a. Snell	9
3 Equació general de rendering	11
4 Light paths	12
5 Ray casting	12
6 Ray tracing Types	13
6.1 Local	13
6.2 Ray tracing clàssic	13
6.3 Path tracing	14
6.4 Distrib RT	15
6.5 Two-pass RT	16
7 RayTracing clàssic	17
8 Phong	20
9 Sombras	20
9.1 Sombras por proyección	21
9.1.1 Sin stencil	21
9.1.2 Con stencil	21
9.1.3 Matrices de proyección	23
9.2 Shadow Volumes	24
9.3 Shadow Mapping	26
9.4 RayTracing	30
9.5 Sombras pre-calculadas (light maps)	30
10 Reflections	30
10.1 Ray-tracing	31
10.2 Reflexions basada en objectes virtuels	31
10.2.1 Modelats	31
10.2.2 Reflectits sense stencil test	31
10.2.3 Reflectits amb stencil test	32
10.2.4 Textures dinàmiques	32
10.3 Environment mapping	33
10.3.1 Sphere mapping	36
10.3.2 Cube mapping	36
11 Alpha blending	36
12 Iluminacion local/global	38
12.1 Local	38
12.2 Global	39
13 Radiometria	39



WUOLAH + BBVA

1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

BBVA está adherido al Fondo de Garantía de Depósitos de Entidades de Crédito de España. La cantidad máxima garantizada es de 100.000 euros por la totalidad de los depósitos constituidos en BBVA por persona.

Te regalamos

15€

1
Abre tu Cuenta Online sin comisiones ni condiciones

2
Haz una compra igual o superior a 15€ con tu nueva tarjeta

3
BBVA te devuelve un máximo de 15€

¿Cómo? →

Cuéntame más



WUOLAH
+ BBVA

14 BRDFs, BTDF and BSDF	39
15 Textures	40
15.1 Mapping	40
15.2 Generació de coordenades de textura	41
15.3 Filtrat	43
15.4 Magnification filters	44
15.5 Minification filters	45
15.6 Wrapping	46
15.7 Combinació de color de la textura	46
15.8 Perspective correct interpolation (textures)	47
15.9 Projective texture mapping	47
15.10 Mappings	48

1 Sistemes de coordenades

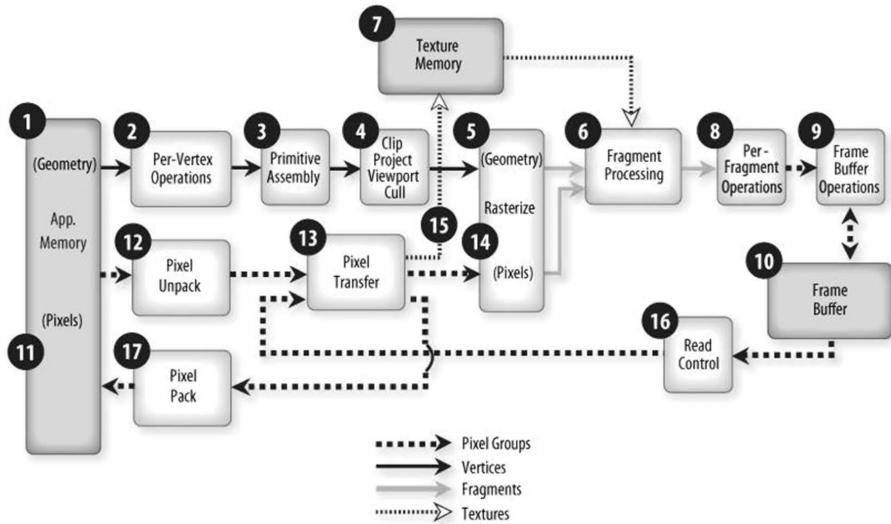


Figure 1:

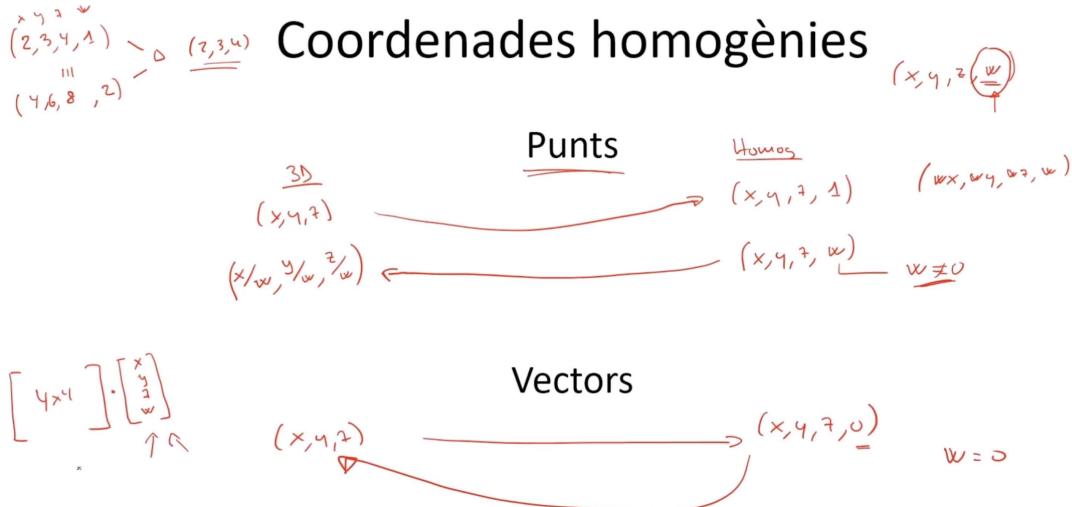


Figure 2:

Indica quins tres vectors hauria de passar el VS al FS per que aquest últim pugui passar un vector qualsevol de tangent space a object space. T, B, N (tangent, bitangent, normal).

1.1 Punts

Object space (x_m, y_m, z_m, w_m):

w_m normalment 1.0

Modeling transform ↓

$T^* R^* S$ (THIS ORDER) Les Z en una cam perspectiva poden estar entre $(-\infty, +\infty)$

Te regalamos

15€

WUOLAH
+ BBVA

1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

BBVA está adherido al Fondo de Garantía de Depósitos de Entidades de Crédito de España. La cantidad máxima garantizada es de 100.000 euros por la totalidad de los depósitos constituidos en BBVA por persona.

1

Abre tu Cuenta Online sin comisiones ni condiciones

2

Haz una compra igual o superior a 15€ con tu nueva tarjeta

3

BBVA te devuelve un máximo de 15€

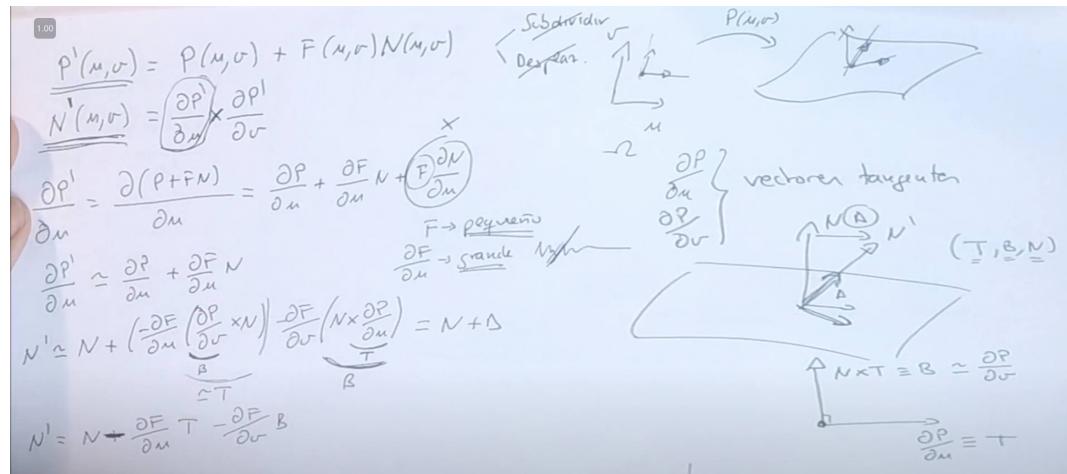


Figure 3:

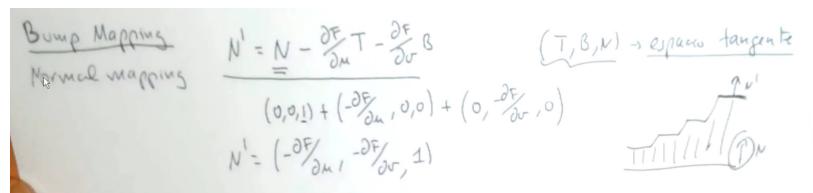


Figure 4:

World space

Viewing transform ↓

w = 1 normalment

Perqué el punt sigui visible Z < 0.0 and Z < -Znear

Fa un canvi al sistema de referència:

- lookAt(eye, target, up)
- $T(0, 0, -d) * Rz(-\phi) * Rx(\theta) * Ry(-\psi) * T(-VRP)$

Eye space

Projection transform ↓

Examples: perspective, frustum, ortho

perspective(fovy, aspect, near, far)

Apertura camara width/height nearZ zfar

Perspective camera Z :(-∞, 0)

Clip space

Perspective division ↓

Punt interior al frustum:

$$-wc \leq xc \leq wc$$

$$-wc \leq yc \leq wc$$

$$-wc \leq zc \leq wc$$

Si tenim càmera perspectiva, wc = -ze

(xc, yc, zc, wc) → (xc/wc, yc/wc, zc/wc) div perspectiva

Normalized device space

Viewport transform & depth transform ↓

Range[-1,1] Punt interior al frustum:

$$-1 \leq xc \leq 1$$

$$-1 \leq yc \leq 1$$

$$-1 \leq zc \leq 1$$

Punts sobre znear: zn = -1

punts sobre zfar: zf = +1

Passem a pixels. (Veure imatges al final de capítol)

Window space

Punt interior al frustum:

$$0 \leq xc \leq w$$

$$0 \leq yc \leq h$$

$$0 \leq zc \leq 1$$

Punts sobre zd: zd = 0

punts sobre zfar: zd = 1

$$\text{gl FragCoord.w} = 1/wc = -1/ze$$

Modeling transform

Transformació de modelat

$$\text{Translate}(t_x, t_y, t_z) \quad T = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\text{Scale}(s_x, s_y, s_z) \quad T = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\text{Rotate}(a, x, y, z) \quad T = \begin{bmatrix} x^2d + c & xyd - zs & xzd + ys & 0 \\ yxd + zs & y^2d + c & yzd - xs & 0 \\ xzd - ys & yzd + xs & z^2d + c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

c=cos(a), s=sin(a), d=1-cos(a)

Figure 5:

$$\text{glRotate*}(a, 1, 0, 0): \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos a & -\sin a & 0 \\ 0 & \sin a & \cos a & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\text{glRotate*}(a, 0, 1, 0): \begin{bmatrix} \cos a & 0 & \sin a & 0 \\ 0 & 1 & 0 & 0 \\ -\sin a & 0 & \cos a & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\text{glRotate*}(a, 0, 0, 1): \begin{bmatrix} \cos a & -\sin a & 0 & 0 \\ \sin a & \cos a & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Figure 6:

1.2 Vector

La w = 0 son "inmunes a las translaciones, de esta manera"

La componente homogénea a diferencia de los puntos donde es 1 aquí es 0. Entre object space,

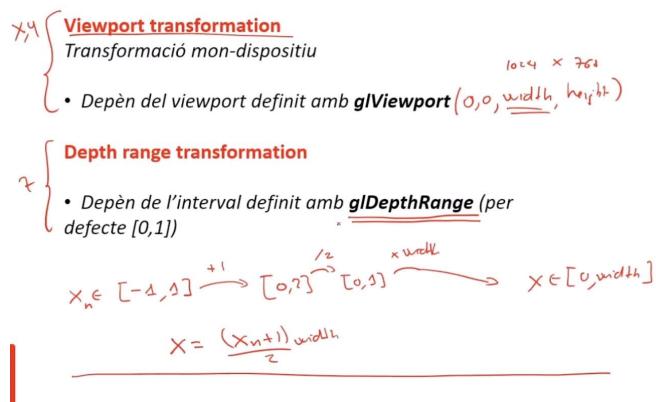


Figure 7:

$$z \in [-1,1] \xrightarrow{+1} [0,2] \xrightarrow{1/z} [0,1]$$

$$z = \frac{(z_{\text{out}} + 1)}{2}$$

Figure 8:

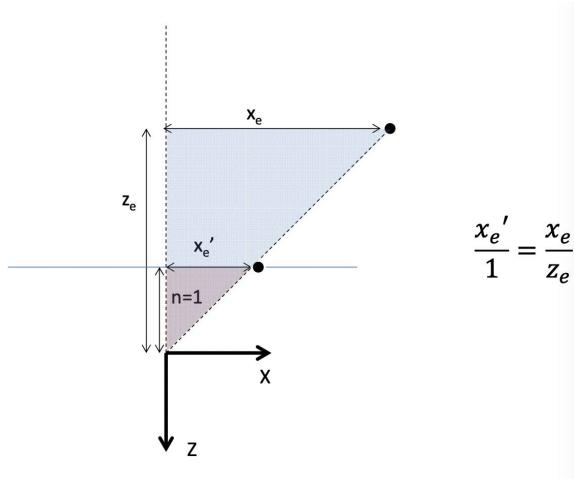


Figure 9:

world space i eye space.

```
perspective(90, 1, 1, 11);
```

$$\left[\begin{array}{cccc} \frac{\cot fovy}{2} & 0 & 0 & 0 \\ aspect & & & \\ 0 & \cot \frac{fovy}{2} & 0 & 0 \\ 0 & 0 & \frac{n+f}{n-f} & \frac{2*n*f}{n-f} \\ 0 & 0 & -1 & 0 \end{array} \right] \xrightarrow[n=1, f=11]{\text{fovy}=90, \text{aspect}=1} \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1.2 & -2.2 \\ 0 & 0 & -1 & 0 \end{array} \right]$$

$$\text{Pas eye space} \rightarrow \text{clip space}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1.2 & -2.2 \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} x_e \\ y_e \\ z_e \\ 1 \end{bmatrix} = \begin{bmatrix} x_e \\ y_e \\ -1.2z_e - 2.2 \\ -z_e \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1.2 & -2.2 \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} x_e \\ y_e \\ -1 \\ 1 \end{bmatrix} = \begin{bmatrix} x_e \\ y_e \\ -1 \\ 1 \end{bmatrix}$$

Punt sobre el pla znear

Punt sobre el pla zfar

Figure 10:

1.3 Vectors normals

Entre object space, world space i eye space.

`normalMatrix` = from object space to eye space.

Where M = `modelMatrix`, V = `viewMatrix`

$$\text{normalMatrix} = (M * V)^{-T}$$

2 Objectes translúcids

Dispersió de la llum transmesa, si la llum transmesa va tota en la mateixa direcció ho veurem com un objecte transparent, si l'objecte reparteix la llum amb la mateixa probabilitat a totes direccions ho veurem com un objecte translúcid.

Si passem d'un medi menys dens μ_1 a un més dens μ_2 llavors $\mu_2 > \mu_1$. Llavors anem d'un angle més separat de N a un angle més proper a N (passem a un angle menor).

En cas que passem d'un medi més dens a un menys dens passa el contrari.

2.1 Equacions de Fresnel a.k.a. Snell

Suposarem varies coses:

- Comportament espectral pur (no difós és a dir sabem a on va la reflexió, i la transmissió).
 - No hi ha absorció. $R+T = 100$ o $R + T = 1$ o $T = 1-R$.
 - El material no és conductor, dielèctric.

Quantitat de llum reflectada i transmesa:

$$R_s = \left(\frac{\sin(\theta_t - \theta_i)}{\sin(\theta_t + \theta_i)} \right)^2$$

Te regalamos



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

BBVA está adherido al Fondo de Garantía de Depósitos de Entidades de Crédito de España. La cantidad máxima garantizada es de 100.000 euros por la totalidad de los depósitos constituidos en BBVA por persona.

$$R_p = \left(\frac{\tan(\theta_t - \theta_i)}{\tan(\theta_t + \theta_i)} \right)^2$$

Schlick approximation:

$$f = \frac{(1-\mu)^2}{(1+\mu)^2}$$

$$R = f + (1-f)(1 - L \cdot N)^5$$

Ángulo crítico:

$$\sin \theta_T = \mu \cdot \sin \theta_i$$

$$1 = \mu \cdot \underline{\underline{\sin \theta_c}}$$

$$\sin \theta_c = \frac{1}{\mu} = \frac{N_2}{N_1}$$

$$\theta_c \approx \arcsin \left(\frac{N_2}{N_1} \right)$$

1

Abre tu Cuenta Online sin comisiones ni condiciones

2

Haz una compra igual o superior a 15€ con tu nueva tarjeta

3

BBVA te devuelve un máximo de 15€

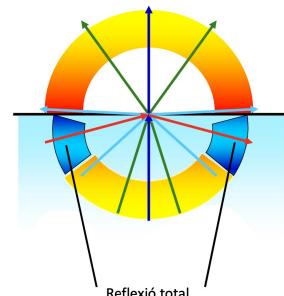


Figure 11:

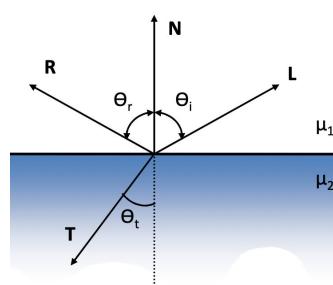


Figure 12: Raig de llum i el seu reflectit i transmès

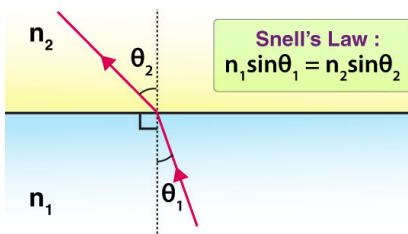


Figure 13: Llei de Snell

3 Equació general de rendering

- **Outgoing light** "Radiancia saliente":
El color de la llum que surt des del punt X (on llancem el raig") cap a l'observador, on $\hat{\omega}_o$ és el vector des del punt X al observador.
- **Emitted light** "Radiancia emitida":
Radiancia emitida por un objeto, una fuente de luz.
- S^2 : Todas las posibles direcciones unitarias de una esfera.
- **Incoming light:** Todas las posibles direcciones de entrada en el punto que estamos mirando, X. Donde todas estas direcciones son representadas por $\hat{\omega}_i$. Incoming Light * Lambert = Irradiancia "Total de energia que llega al punto X".
- **Material:** Parte de esta energia se puede transmitir a $\hat{\omega}_o$ pero no toda, por eso teniendo en cuenta la radiancia que llega por cada $\hat{\omega}_i$ hay que ver qué proporción de esa radiancia se transmite a $\hat{\omega}_o$, recordemos, en el punto X. (La radiancia que se refleje o se transmita hacia otros sitios que no sean $\hat{\omega}_o$ no nos interesan).
- **Lambert:** Tiene en cuenta con que ángulo impacta en la superficie el rayo de luz, intuitivamente si la luz impacta de frente, se vera más luminoso que no si la luz impacta de forma mas lateral.
Lambert: $\cos \theta_i$
- $d\hat{\omega}_i$: Es la variable de integración, para cualquier posible dirección de entrada.

$$L_o(X, \hat{\omega}_o) = L_e(X, \hat{\omega}_o) + \int_{S^2} L_i(X, \hat{\omega}_i) f_X(\hat{\omega}_i, \hat{\omega}_o) |\hat{\omega}_i \cdot \hat{n}| d\hat{\omega}_i$$

Outgoing light	Emitted light	Incoming light	Material	Lambert
----------------	---------------	----------------	----------	---------

Figure 14: Equació general de rendering

4 Light paths

Los light paths se pueden describir de la forma:

$$L(D|S)^*E \quad (1)$$

Donde:

- L: Luz
- D: Difuso
- S: Especular
- E: Eye

Por ejemplo: LDDE, LLDSSDSE, LSDE...

Clear: Specular.

Opaque: Diffuse (if it has specular highlight then both), you would consider both paths LDE + LSE.

Mirror: Specular.

Phong: (modelo de iluminacion local) LDE + LSE

LE no se soporta en Phong.

5 Ray casting

No es recursivo. En vez de utilizar rasterización, lanzar rayos desde la cámara hacia la escena.

1. **Recorrer todos los píxeles** de la escena.
2. **Calculamos el rayo que pasa por ese pixel.** (Podemos calcular la posición del fragmento ademas de en window space, en cualquier otro espacio).
3. **Con ese rayo podemos calcular todas las intersecciones del rayo con la escena,** en particular queremos obtener la intersección más cercana con la escena.
4. **Calcular cual es el color en ese punto** (en la intersección). O lo que es lo mismo, la radiancia saliente: $L_o(X, \hat{\omega}_o)$. En el caso de ray-casting no se hace de forma recursiva, **se hace con un modelo de iluminación local**, ray-casting, sí hará este cálculo recursivo.

Te regalamos



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

BBVA está adherido al Fondo de Garantía de Depósitos de Entidades de Crédito de España. La cantidad máxima garantizada es de 100.000 euros por la totalidad de los depósitos constituidos en BBVA por persona.

1

Abre tu Cuenta Online sin comisiones ni condiciones

2

Haz una compra igual o superior a 15€ con tu nueva tarjeta

3

BBVA te devuelve un máximo de 15€

6 Ray tracing Types

6.1 Local

Ray tracing local nos permite modelar: LDE, LSE.

6.2 Ray tracing clásico

Este modelo soporta path lights tal que:

$$LDS^*E \quad (2)$$

Parecido a ray-casting, lanza rayos en sentido contrario al de la propagación de la luz, desde el observador. Lanzamos un rayo primario (por pixel), si la interficie choca contra un objeto, lanzamos un shadow ray (o light rays) hacia cada luz, para saber si la luz directa llega hasta el punto donde choca el rayo. Si el rayo choca contra una superficie especular entonces desde esa superficie debo lanzar el rayo reflejado y si además es translúcida otro rayo transmitido, cuando llega a superficie difusa utilizo un modelo de iluminación local, por cada rayo en chocar contra una superficie también lanzamos los shadow rays que sean necesarios.



Figure 15: Render usando ray tracing clásico

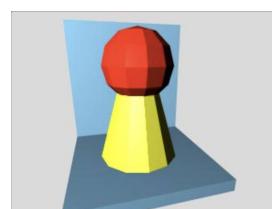


Figure 16: Render usando ray casting

6.3 Path tracing

Este modelo soporta path lights tal que:

$$L(D|S)^* E \quad (3)$$

Variante de ray tracing, por cada pixel no generamos un rayo, lanzamos **r rayos** (distribuidos con gittering dentro de ese pixel, es decir, no pasan necesariamente por el centro del pixel). Lanzamos un rayo primario (desde el observador), i cuando intersecciona con una superficie a partir de ese punto, **genera otro rayo tanto si es difusa como especular**, por lo tanto generamos un único camino por rayo (de manera aleatoria), a diferencia de ray-tracing clásico. Deberíamos también limitar el número de rebotes ya que no paramos al llegar a superficie difusa, (por ejemplo 10). Serà un rayo reflejado o transmitido? Depende, **si el objeto es translúcido elegiremos al azar** (no necesariamente con probabilidad uniforme) **si el rayo que lanzamos es reflejado o transmitido**.

Si la superficie es especular, la dirección del rayo especular está claro como calcularla, si la superficie es difusa utilitzaremos una probabilidad (preferentemente con un BRDF que nos indica la probabilidad de que se refleje en cada dirección distinta). Con path tracing, debido a soportar superficies difusas hay mucha variancia, (variabilidad) se representa con mucho ruido en la imagen, el rayo primario impacta en una superficie y los rayos reflejados impactan por casualidad por ejemplo en zonas muy luminosas i eso provoca ruido en la imagen.

También lanza shadow rays.



Figure 17: Render con path tracing

6.4 Distrib RT

Este modelo soporta path lights tal que:

$$LDS^*E \quad (4)$$

De forma análoga a ray-tracing clásico, cuando topamos con una superficie difusa, paramos la recursividad. Cuando alcanzamos una superficie especular en lugar de lanzar un rayo como en ray-tracing clásico, lanzamos varios rayos reflejados y si fuera translúcida lanzaríamos varios rayos transmitidos.

Por qué lanzar varias reflexiones? Si la reflexión que esperamos de un objeto es un poco difuminada (la reflexión no es solo en una dirección sino en un cono por ejemplo)

Ray-tracing clásico + nos permite simular reflexiones imperfectas.

También lanza shadow rays.

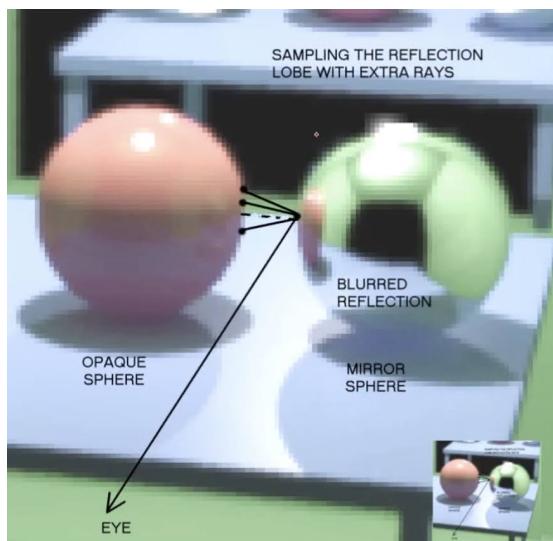


Figure 18: Render con path tracing

6.5 Two-pass RT

Este modelo soporta path lights tal que:

$$LS^* DS^* E \quad (5)$$

Light pass: LS*DE

Eye pass: LDS*E

Dos pasos:

1. **Light pass:** Lanzamos rayos desde la fuente de luz, vamos impactando y rebotando rayos con superficies especulares y si son translúcidas pue transmitidos también, hasta que llegamos a una superficie difusa, donde se almacena cierta energía (la que tiene el fotón). Almacenamos la radiancia de RGB que ha llegado en ese punto en una estructura de datos.
2. **Eye pass:** Como en ray-tracing clásico, para cada pixel lanzamos un rayo desde la cámara, hasta llegar finalmente a una superficie difusa, aparte de lanzar los shadow-rays para saber si llega luz directa, también consulto la información almacenada en puntos cercanos del paso anterior(light pass), miro cual es la irradiancia que ha llegado.

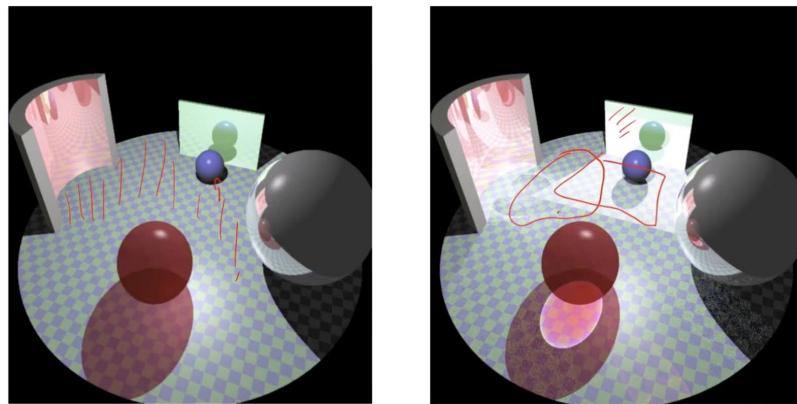


Figure 19: Render comparando ray tracing clásico con two-pass ray tracing

Te regalamos

15€



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

BBVA está adherido al Fondo de Garantía de Depósitos de Entidades de Crédito de España. La cantidad máxima garantizada es de 100.000 euros por la totalidad de los depósitos constituidos en BBVA por persona.

1

Abre tu Cuenta Online sin comisiones ni condiciones

2

Haz una compra igual o superior a 15€ con tu nueva tarjeta

3

BBVA te devuelve un máximo de 15€

7 RayTracing clàssic

$$I(P) = I_D(P) + I_R(P) + I_T(P)$$

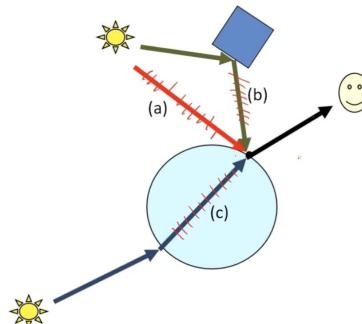


Figure 20:

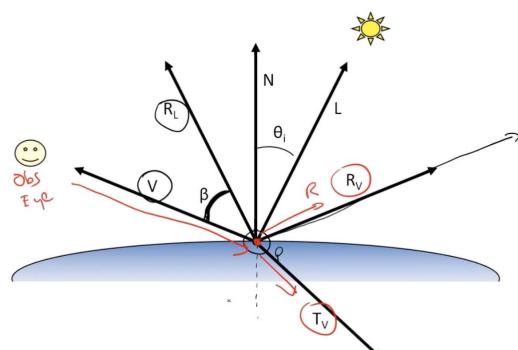


Figure 21:

$$I(P) = I_D(P) + I_R(P) + I_T(P)$$

$$I_D(P) = K_a I_a + K_d \sum_{N \cdot L} I_L \cos(\theta_i) + K_s \sum_{(R+V)^n} I_L \cos^n(\beta)$$

- $\cos(\theta_i) = N \cdot L$
- $\cos(\beta) = R \cdot V$
- El sumatori només considera les fonts de llum no ocluides (ombres)

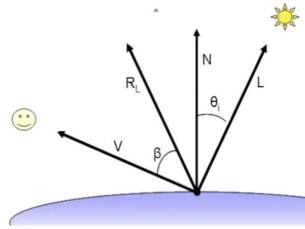


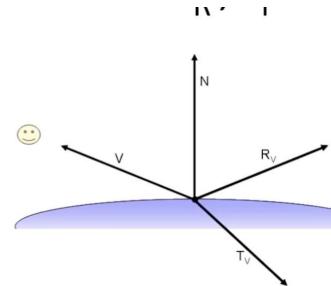
Figure 22:

Mat
Local $\rightarrow K_d \leftarrow$
 $K_a \leftarrow$
Ind $\rightarrow K_R \leftarrow$
 $K_T \leftarrow$

$$I(P) = I_D(P) + I_R(P) + I_T(P)$$

$$I_R(P) = K_R L_R$$

$$I_T(P) = K_T L_T$$



- K_R, K_T coeficients empírics de reflexió/transmissió espelular
- L_R = llum que incideix en P en la direcció R_V
- L_T = llum que incideix en P en la direcció T_V

Es calculen recursivament, traçant un nou raig reflectit i un altre transmès

Figure 23:

```
acció rayTracing
    per i en [0..w-1] fer
        per j en [0..h-1] fer
            ||raig:=raigPrimari(i, j, camera);
            color:=traçarRaig(raig, escena, mu); ~ 1.*
            setPixel(i, j, color);
    fper
    fper
faccio
```

Figure 24:

```

funció traçar_raig(raig, escena, μ)
    si profunditat_correcta() llavors
        info:=calcula_interseccio(raig, escena)
        si info.hi_ha_interseccio() llavors
            color:=calcular_I0(info, escena) // I0
            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

```

Figure 25:

8 Phong

Lambert: $N \cdot L$ (difusa)
Phong: $R \cdot V$ (specular)

$$K_e + K_a(M_a + I_a) + K_d I_d(N \cdot L) + K_s I_s(R \cdot V)^s$$

- K_* = material $\underbrace{\hspace{10em}}$ Només si $N \cdot L > 0$
- I_* = Illum $\underbrace{\hspace{10em}}$ Només si $N \cdot L > 0$

Figure 26: Fórmula que calcula el model de phong

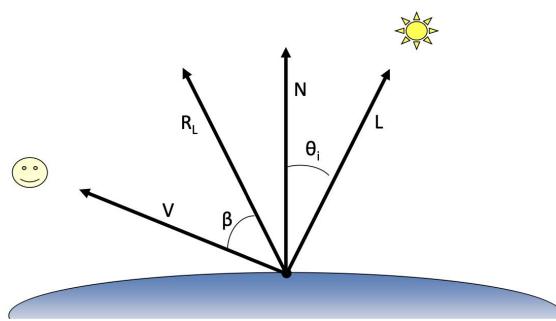


Figure 27: Notació de la fórmula

9 Sombras

- Oclusor: Objeto que provoca una sombra en el objeto receptor.
- Receptor: Superficie que recibe parte de la luz.

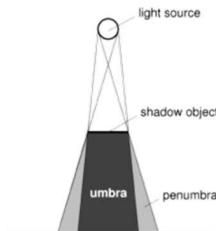


Figure 28: Sombras provocadas por una fuente de luz con un cierto volumen (no es puntual)

Te regalamos



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

BBVA está adherido al Fondo de Garantía de Depósitos de Entidades de Crédito de España. La cantidad máxima garantizada es de 100.000 euros por la totalidad de los depósitos constituidos en BBVA por persona.

1

Abre tu Cuenta Online sin comisiones ni condiciones

2

Haz una compra igual o superior a 15€ con tu nueva tarjeta

3

BBVA te devuelve un máximo de 15€



9.1 Sombras por proyección

9.1.1 Sin stencil

Precondiciones:

- Receptor plano
 - Luz puntual
1. Dibujar el receptor.
 2. Dibujar la sombra = dibujar el oclusor proyectado (se puede representar como una matriz 4x4). Las z's de la sombra deben de tener prioridad respecto al receptor, ya que si no se produce z-fighting.
 3. Dibujar el oclusor.

glPolygonOffset(factor,units), abans del depth test, modifica el valor de la z del fragment amb

$$z' = z + \partial z * factor + r * units \quad (6)$$

on r es el valor més petit que garantitza un offset > 0 on units = offset.
Representa quan tangencial és la primitiva a la direcció de visió

$$\partial z = \max\left(\frac{\partial z}{\partial x}, \frac{\partial z}{\partial y}\right) \quad (7)$$

Factor: permet introduir un offset variable (en funció de la inclinació del polígon). glPolygonOffset(-1,-1) si ho volem veure més a prop.

Problema del método, no tiene en cuenta la extensión del receptor.

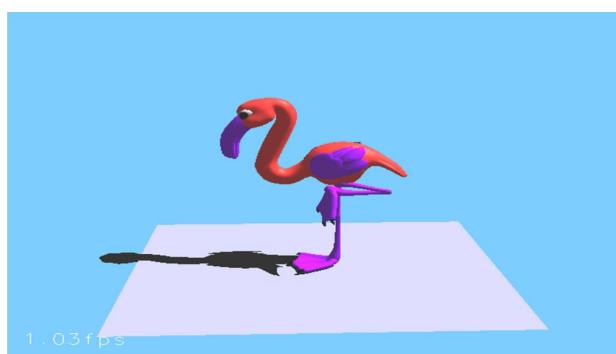


Figure 29: Sombra por proyección sin usar Stencil buffer

9.1.2 Con stencil

Stencil buffer:

Stencil buffer funciona como una máscara. El stencil test se hace antes del depth test. El stencil buffer guarda, para cada pixel un entero entre 0.. $2^n - 1$. Como es un buffer podemos borrarlo, glClearStencil(0), glClear(GL_STENCIL_BUFFER_BIT);
Establecer el test de comparación:

1. glEnable(GL_STENCIL_TEST)

2. glStencilFunc(comparació, valorRef, mask)

comparacio: GL NEVER, GL ALWAYS, GL LESS

Operacions a fer a stencil buffer segons el resultat:

1. glStencilOp(fail,zfail, zpass)

fail: operacio a fer quan el fragment no passa el test de stencil.

Zfail: operació a fer quan passa stencil, pero no passa z-buffer.

Zpass: operació a fer quan passa stencil i passa z-buffer.

2. Els valors poden ser: GL KEEP, GL ZERO, GL INCR, GL DECR, GL INVERT, GL REPLACE (valorRef).

Sombras por proyección:

Targets	Passos
Color, Z, Stencil	1. Dibuixa el receptor al color buffer y al stencil buffer
Stencil	2. Dibuixa l'oclosor per neterejar l'stencil a les zones a l'ombra
Color, Z	3. Dibuixa la part fosca del receptor
Color, Z	4. Dibuixa l'oclosor

Pas 3: Activa depth test, activa color, desactiva iluminació (per a que es vegi negre) i amb glStencilFunc(GL equal, 0, 1) quan en l'stencil tenim un 0, pintarem, para solucionar z fighting se usa glDepthFunc(GL LEQUAL).

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

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

// 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'oclosor
glEnable(GL_LIGHTING);
glDepthFunc(GL_LESS);
glDisable(GL_STENCIL_TEST);
Dibuixa(oclosor);
```

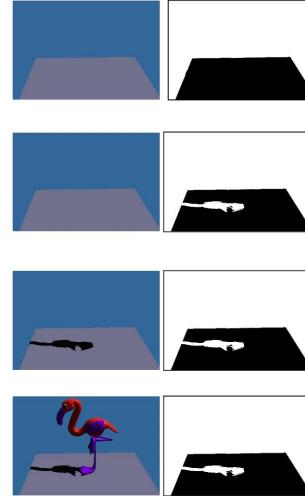


Figure 30: Render de una proyección de sombra usando stencil buffer

9.1.3 Matrices de proyección

Luz	Dir
Puntual en el origen	\vec{OP}
Puntual en F	\vec{FP}
Direccional L	$-\vec{L}$

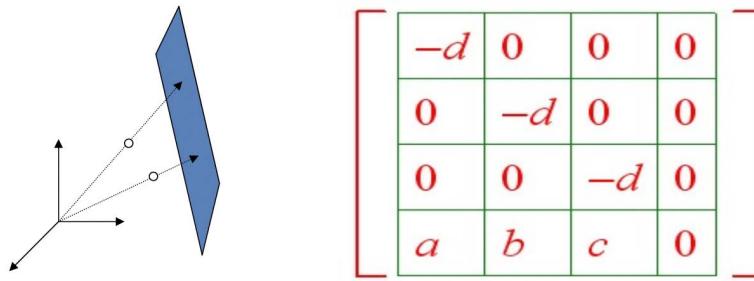


Figure 31: Matriz de proyección con luz puntual en el origen

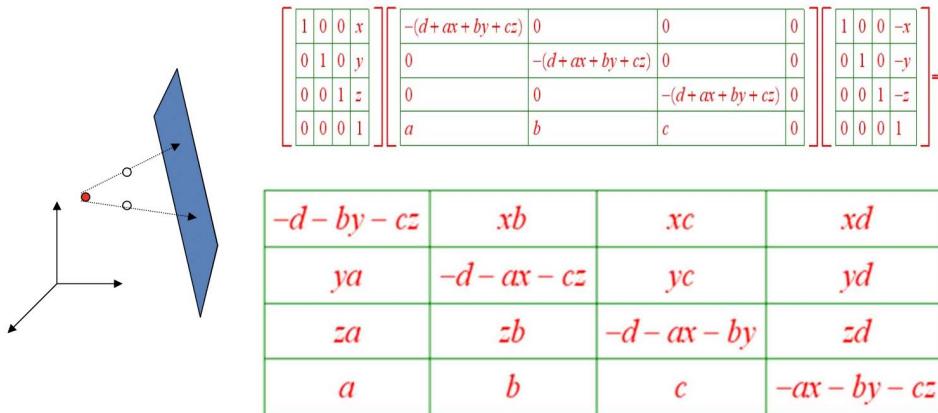


Figure 32: Matriz de proyección con luz puntual

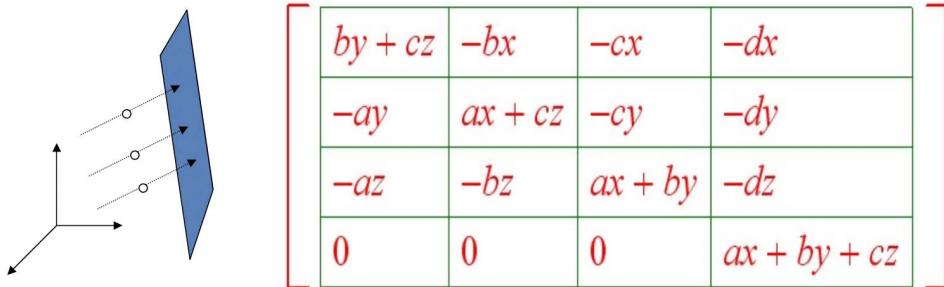


Figure 33: Matriz de proyección con luz direccional

9.2 Shadow Volumes

Precondiciones:

1. Receptor arbitrario (antes tenia q ser plano)
2. Oclusor sencillo
3. Luz puntual

A partir de las aristas contorno del oclusor generamos un shadow volume.

Arista es contorno \equiv las caras que comparten la arista: 1 es backface, la otra es frontface.

Número de intersecciones con un polígono o un volumen:

1. Par: el punto és exterior
2. Impar: el punto és interior

Anàlogament:

Interseccions frontface - interseccions backface > 0

P in SV, else P not in SV

Primer pintem l'escena al Z buffer per a saber les distàncies de l'escena.

Després dibuixem a l'estencil les cares frontals del volum amb el backface culling eliminem les cares backface.

Volums d'ombra (1/2)

```

    ① // 1. Dibuixa l'escena al z-buffer
    glColorMask(GL_FALSE, ..., GL_FALSE);
    dibuixa(escena);

    // 2. Dibuixa al stencil les cares frontals del volum SV
    glEnable(GL_STENCIL_TEST);
    glDepthMask(GL_FALSE);
    glStencilFunc(GL_ALWAYS, 0, 0);
    glEnable(GL_CULL_FACE);
    glStencilOp(GL_KEEP, GL_KEEP, GL_INCR);
    glCullFace(GL_BACK);
    dibuixa(volum_ombra);

    // 3. Dibuixa al stencil les cares posteriors del volum SV
    glStencilOp(GL_KEEP, GL_KEEP, GL_DECR);
    glCullFace(GL_FRONT);
    dibuixa(volum_ombra);
  
```

43

Figure 34:

Volums d'ombra (2/2)

```

    // 4. Dibuixa al color buffer la part fosca de l'escena
    glDepthMask(GL_TRUE);
    glColorMask(GL_TRUE, ..., GL_TRUE);
    glCullFace(GL_BACK);
    glDepthFunc(GL_EQUAL);
    glStencilOp(GL_KEEP, GL_KEEP, GL_KEEP);
    glStencilFunc(GL_EQUAL, 1, 1);
    glDisable(GL_LIGHTING);
    dibuixa(escena);

    // 5. Dibuixem al color buffer la part clara de l'escena
    glStencilFunc(GL_EQUAL, 0, 1);
    glEnable(GL_LIGHTING);
    dibuixa(escena);

    // 6. Restaura l'estat inicial
    glDepthFunc(GL_LESS);
    glDisable(GL_STENCIL_TEST);
  
```

Figure 35:

Te regalamos



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

BBVA está adherido al Fondo de Garantía de Depósitos de Entidades de Crédito de España. La cantidad máxima garantizada es de 100.000 euros por la totalidad de los depósitos constituidos en BBVA por persona.

1

Abre tu Cuenta
Online
sin comisiones
ni condiciones

2

Haz una compra
igual o superior
a 15€ con tu
nueva tarjeta

3

BBVA
te devuelve
un máximo de
15€

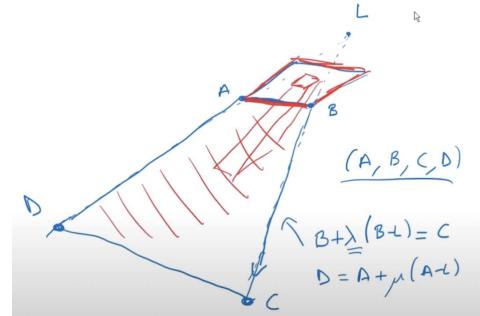


Figure 36: Calcular SV usando las aristas contorno

9.3 Shadow Mapping

Precondiciones:

- No hace falta que haya receptor y oclusor, de hecho cada objeto puede actuar como ambas
- Luz puntual → spotlight

Depth map → shadow map.

1. Definir cámara en la fuente de luz, dibuja la escena y define la z con el punto mas cercano a la fuente de luz. Y la ponemos en una textura.
2. Definir cámara del observador, dibujamos la escena.

$$T(0.5) * S(0.5) * P * V * M \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} s \\ t \\ p \\ q \end{bmatrix} \quad (8)$$

$P * V$ = lightCamera

M = Object

(x, y, z) = vertex (object space).

p = z in window space.

If $\text{shadowmap}\left(\frac{s}{q}, \frac{t}{q}\right) \simeq \frac{p}{q}$ then el vertice està iluminado,
sino si $\text{shadowmap}\left(\frac{s}{q}, \frac{t}{q}\right) < \frac{p}{q}$, el vertice està en la sombra.

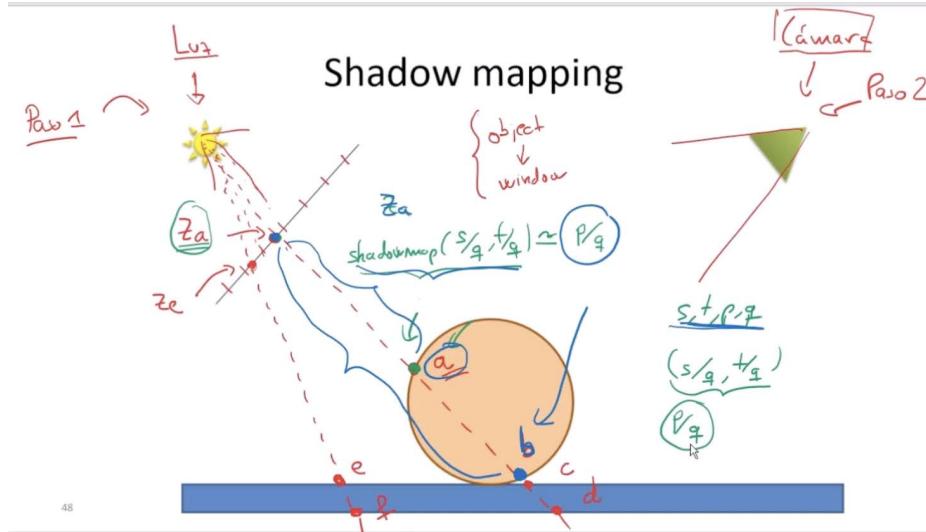


Figure 37:

```
// Pas 1. Actualització del shadow map
// 1. Definir càmera situada a la font de llum
glViewport( 0, 0, SHADOW_MAP_WIDTH, SHADOW_MAP_HEIGHT );
glMatrixMode( GL_PROJECTION );
glLoadIdentity();
gluPerspective( fov, ar, near, far); // de la càmera situada a la llum!
glMatrixMode( GL_MODELVIEW );
glLoadIdentity();
gluLookAt( lightPos, ..., lightTarget, ...., up,...);

// 2. Dibuixar l'escena
glClear( GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT );
glPolygonOffset(1,1); glEnable(GL_POLYGON_OFFSET_FILL);
drawScene();
glDisable(GL_POLYGON_OFFSET_FILL);

// 3. Guardar el z-buffer en una textura
glBindTexture(GL_TEXTURE_2D, textureId);
glCopyTexSubImage2D(GL_TEXTURE_2D, 0, 0, 0, 0, 0, SHADOW_MAP_WIDTH,
SHADOW_MAP_HEIGHT);

// Restaurar càmera i viewport
```

Figure 38:

```

// Generació de coords de textura pel shadow map
// La generació és similar a projective texture mapping
glLoadIdentity();
glTranslated( 0.5, 0.5, 0.5 );
glScaled( 0.5, 0.5, 0.5 );
gluPerspective( fov, ar, near, far);
gluLookAt( lightPos, ... lightTarget, ... up...);

→ La matriu resultant és la que passa les coordenades del vertex (x,y,z,1) de world space a homogeneous texture space (s,t,p,q)

```

Figure 39:

```

// VS
uniform mat4 lightMatrix;
out vec4 texCoord;

void main()
{
    ...
texCoord = lightMatrix*vec4(vertex,1);
gl_Position = modelViewProjectionMatrix * vec4(vertex,1);
}

```

```

// FS
...
vec2 st = texCoord.st / texCoord.q;
float trueDepth = texCoord.p / texCoord.q;
float storedDepth = texture(shadowMap, st).r;
float bias = 0.01; // només si no hem usar glPolygonOffset
if (trueDepth - bias <= storedDepth)
    fragColor = ... // iluminat
else
    fragColor = ... // a l'ombra

```

Figure 40:

Te regalamos



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

BBVA está adherido al Fondo de Garantía de Depósitos de Entidades de Crédito de España. La cantidad máxima garantizada es de 100.000 euros por la totalidad de los depósitos constituidos en BBVA por persona.

1

Abre tu Cuenta Online sin comisiones ni condiciones

2

Haz una compra igual o superior a 15€ con tu nueva tarjeta

3

BBVA te devuelve un máximo de 15€

9.4 RayTracing

9.5 Sombras pre-calculadas (light maps)

10 Reflections

- Reflexión difusa: la luz se refleja con la misma intensidad en todas direcciones.
- Reflexión specular: la luz se refleja siguiendo la ley de snell.

Vector reflectit

R, N, L copланарен

$$\begin{aligned} R &= \underline{R_1} + \underline{R_2} \rightarrow -L + 2R_2 \\ \underline{R_1} &= -L + R_2 \quad | -L + 2N(N \cdot L) \\ \underline{R_2} &= N(N \cdot L) \end{aligned}$$

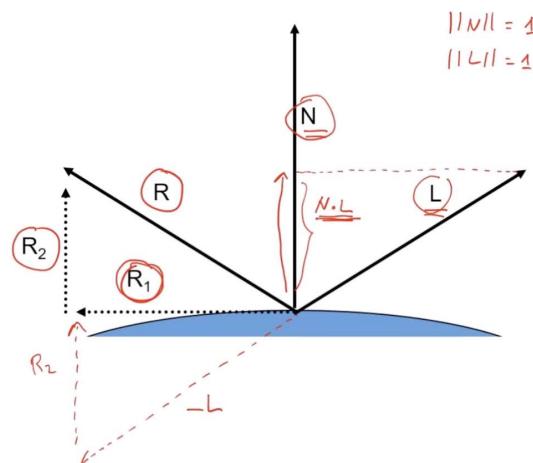


Figure 41:

$$\text{Matriu de reflexió respecte un pla } (a,b,c,d): \begin{bmatrix} 1 - 2a^2 & -2ba & -2ca & -2da \\ -2ba & 1 - 2b^2 & -2cb & -2db \\ -2ca & -2cb & 1 - 2c^2 & -2dc \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$x=0 \rightarrow S(-1, 1, 1)$$

Matriu de reflexió

$$d = \text{dist}(P, P_{\text{plane}}) = \underbrace{ap_x + bp_y + cp_z + d}_{(a, b, c, d)} \rightarrow \underbrace{ax + by + cz + d = 0}$$

$$\begin{aligned} P' &= P - 2d \vec{n} \\ P'_x &= \underline{p_x} - \underline{2(a p_x + b p_y + c p_z + d) a} \\ &= (1 - 2a^2)p_x - 2ab p_y - 2ac p_z - 2ad \end{aligned}$$

$$\begin{bmatrix} 4 \times 4 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} = \begin{bmatrix} p'_x \\ p'_y \\ p'_z \\ 1 \end{bmatrix}$$

Figure 42:

10.1 Ray-tracing

10.2 Reflexions basada en objectes virtuals

Precondición: Espejo plano.

10.2.1 Modelats

Duplicar los objetos y reflejar-los.

10.2.2 Reflectits sense stencil test

En vez de duplicar los objetos, pintar-los dos veces. Para que la reflexion sea correcta hay que hacer reflexion de la fuente de luz también.

Algorisme (versió 1)

```

// 1. Dibujar els objectes en posició virtual
glPushMatrix();
glMultMatrix(matriu_simetria);
glLightfv(GL_LIGHT0, GL_POSITION, pos);
glCullFace(GL_FRONT);
dibuixar(escena);
glPopMatrix();

// 2. Dibujar el mirall semi-transparent
glLightfv(GL_LIGHT0, GL_POSITION, pos);
glCullFace(GL_BACK);
dibuixar(mirall);

// 3. Dibujar els objects en posició real
dibuixar(escena);

```

Figure 43:

10.2.3 Reflectits amb stencil test

Stencil

<u>Op</u>	<u>Test</u>	<u>True</u>
0	1	

Algorisme (versió 2)

```

// 1. Dibuxem 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. Dibuxem els objectes virtuals
 glDepthMask(GL_TRUE); glColorMask(GL_TRUE...);
 glStencilFunc(GL_EQUAL, 1, 1);
 glStencilOp(GL_KEEP, GL_KEEP, GL_KEEP);
 glPushMatrix(); glMultMatrix(matriu simetria)
 glLightfv(GL_LIGHT0, GL_POSITION, pos);
 glCullFace(GL_FRONT);
 dibuixar(escena);
 glPopMatrix();

// 3. Dibuxem el mirall semitransparent
 glDisable(GL_STENCIL_TEST);
 glLightfv(GL_LIGHT0, GL_POSITION, pos);
 glCullFace(GL_BACK);
 dibuixar(mirall);

// 4. Dibuxem els objectes reals
 dibuixar(escena);

```

Color buf Stencil Buf Test Final

X	X	0	X
			X
		✓	✓
		✓	✓
		✓	✓

23

Figure 44:

10.2.4 Textures dinàmiques

1. Dibujar objeto en posición virtual (la escena reflejada)
2. Crear una textura con la posición virtual creada.
3. Dibujar el espejo usando la textura.
4. Dibujar objeto en posición real.

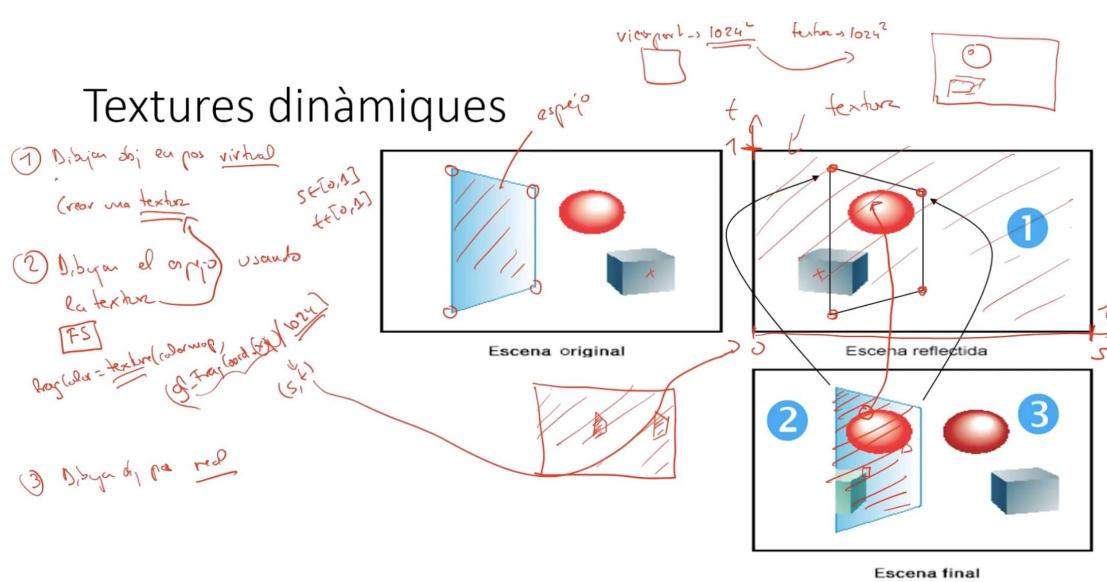


Figure 45:

10.3 Environment mapping

Donada una direcció arbitrària R , ens retorna el color de l'entorn en direcció R .

Suponem que el entorn està infinitament alejado (no cambia cuando nos movemos). Se puede codificar de muchas maneras.

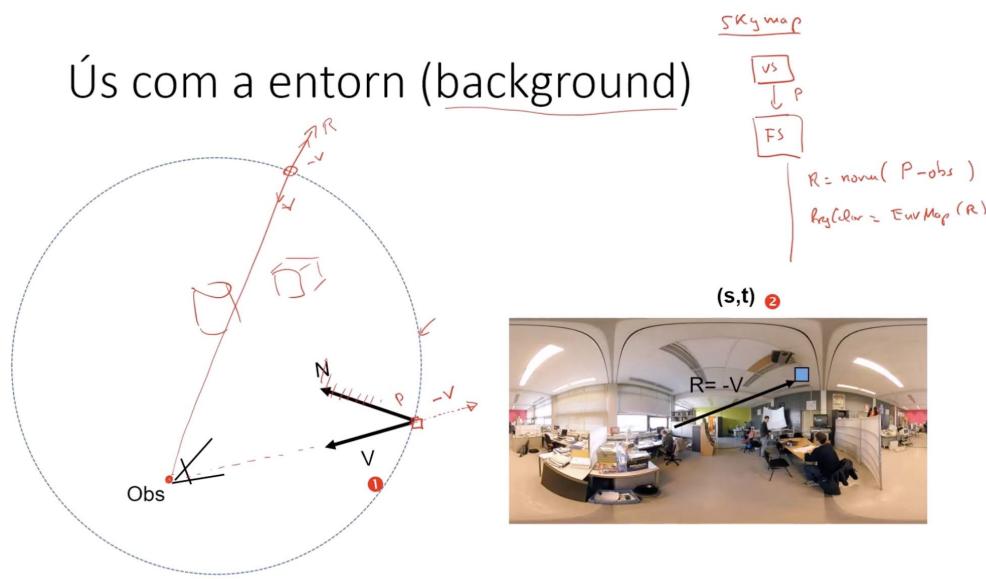


Figure 46:

Te regalamos

15€



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

BBVA está adherido al Fondo de Garantía de Depósitos de Entidades de Crédito de España. La cantidad máxima garantizada es de 100.000 euros por la totalidad de los depósitos constituidos en BBVA por persona.

1

Abre tu Cuenta Online sin comisiones ni condiciones

2

Haz una compra igual o superior a 15€ con tu nueva tarjeta

3

BBVA te devuelve un máximo de 15€

Us per reflexions especulars

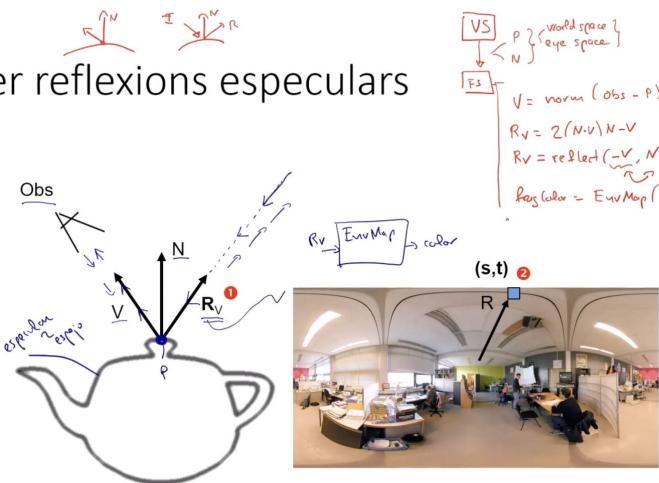


Figure 47:

Relació entre vector R i (u,v)

$$\begin{aligned}
 & V = (0, 0, 1) \\
 & R_v = 2(N \cdot V)N - V \\
 & R_v = 2n_x(n_x, n_y, n_z) - (0, 0, 1) \\
 & R_v = \left(\frac{2n_x n_x}{r_x}, \frac{2n_x n_y}{r_y}, \frac{2n_x n_z}{r_z} - 1 \right)
 \end{aligned}$$

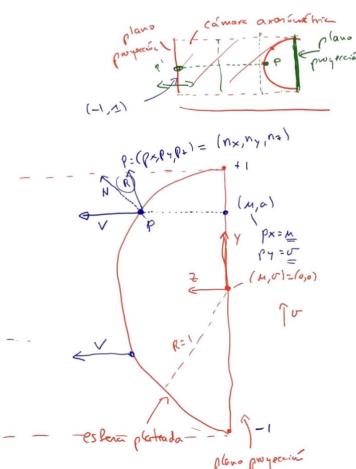


Figure 48:

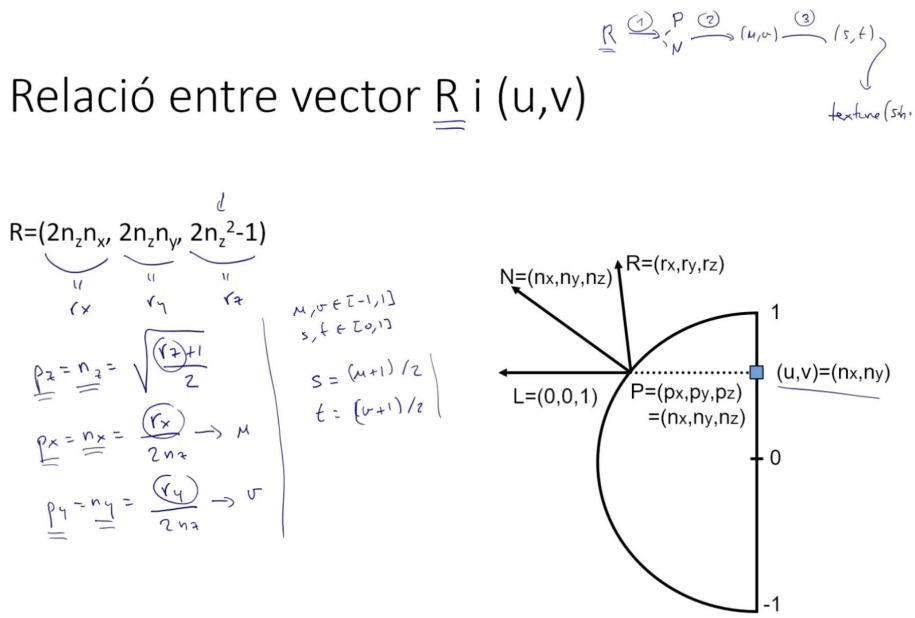


Figure 49:

```
vec4 sampleSphereMap(sampler2D sampler, vec3 R)
{
    float z = sqrt((R.z+1.0)/2.0);
    vec2 st=vec2((R.x/(2.0*z)+1.0)/2.0,(R.y/(2.0*z)+1.0)/2.0);
    return texture(sampler, st);
}
```

Figure 50:

10.3.1 Sphere mapping

- Només s'aprofita el cercle inscrit de la textura
- Conté informació de aproximadament tot l'entorn (totes direccions).
- Distorsió considerable a prop de la vora del cercle

10.3.2 Cube mapping

```

uniform sampler2D sampler;
...
fragColor = texture(sampler, vtexCoord);
    ↓
    s, t
|| FS

```



```

uniform samplerCube samplerC;
...
vec3 R;
...
fragColor = textureCube(samplerC, R);
    ↗ GLSL
    ↘ vec3

```

Figure 51:

11 Alpha blending

`glBlendFunc(a1,a2): srcColor * a1 + dstColor * a2`

✓ = must have ON.

✗ = Should NOT be on

1. Dibujar todo ordenado back-to-front

depth-test: indiferente

2. Dibujar opacos sin ordenar:

depth test: ✓ (glEnable) (sino, pondriamos objetos opacos que estan mas lejos, delante de objetos opacos que estan mas cerca)

depth write: ✓ (glDepthMask)

Dibujar transparentes ordenados:

depth test: ✓ (sino la transparencia apareceria en frente de un objeto opaco)

depth write: indiferente (da igual porque estan ordenados)

3. Opacos sin ordenar:

test: ✓

write: ✓

Transparentes sin ordenar:

test: ✓

write: ✗ (Si está activo, si se dibuja primero un objeto delante de otro, cuando dibujes el segundo, el segundo no se verá, pero debería, ya que los dos son translúcidos)

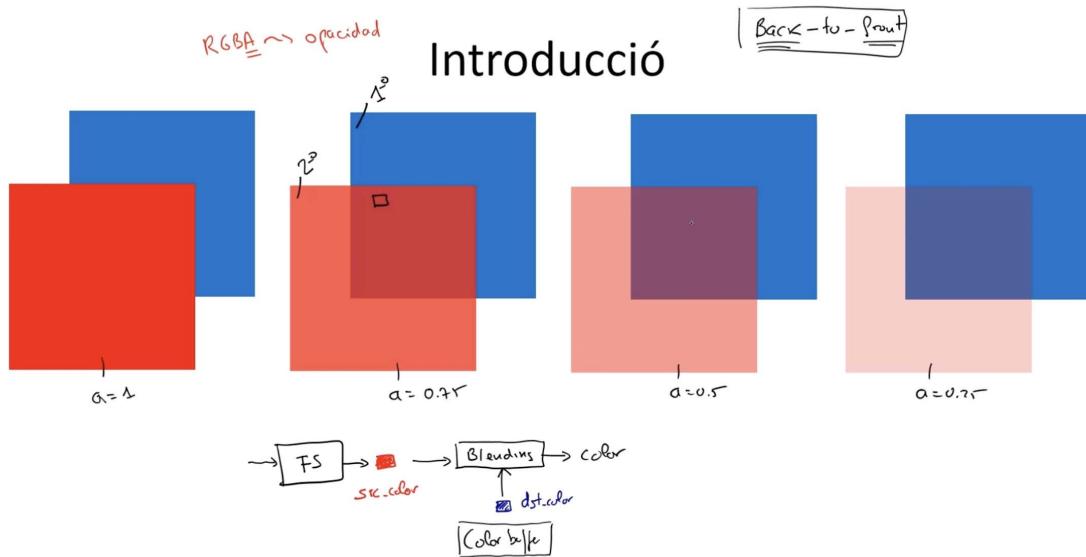


Figure 52:

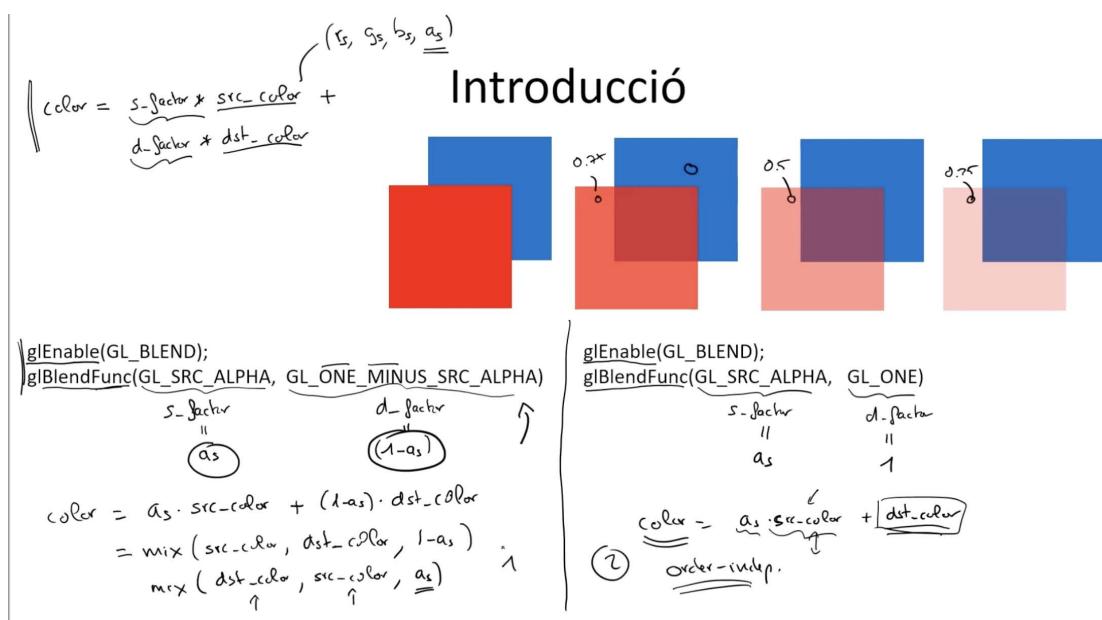


Figure 53:

Te regalamos

15€



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

BBVA está adherido al Fondo de Garantía de Depósitos de Entidades de Crédito de España. La cantidad máxima garantizada es de 100.000 euros por la totalidad de los depósitos constituidos en BBVA por persona.

1

Abre tu Cuenta Online sin comisiones ni condiciones

2

Haz una compra igual o superior a 15€ con tu nueva tarjeta

3

BBVA te devuelve un máximo de 15€

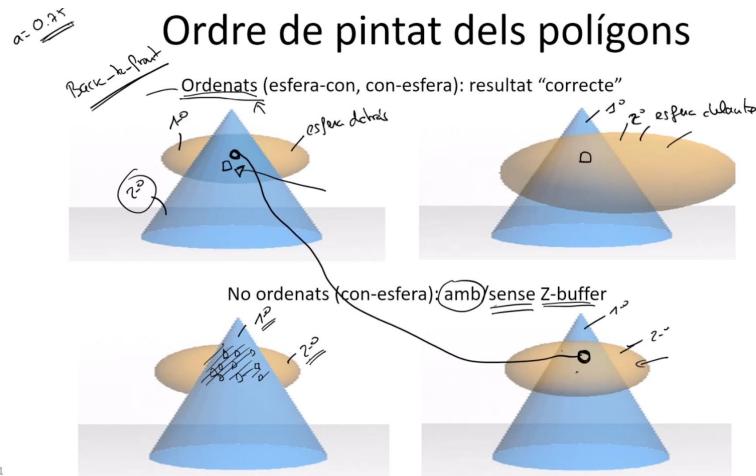


Figure 54:

12 Iluminacion local/global

12.1 Local

Considera la luz directa

- Emisor: luz o environment map
- superficie

12.2 Global

Considera luz directa + indirecta

- emisor: luz o cualquier superficie
- receptor

13 Radiometria

Sím.	Radiomet.	Fotometria	Definició	Ús
Φ	Fluxe (W)	Fluxe (lm)	Energia que travessa una superficie per unitat de temps	Energia total que emet una font de llum
E	Irradiancia (W/m^2)	Iluminància ($\text{lux} = \text{lm}/\text{m}^2$)	Fluxe per unitat d'àrea	Llum que incideix en un punt, des de qualsevol direcció
I	Intensitat (W/sr)	Intensitat ($\text{cd} = \text{lm}/\text{sr}$)	Fluxe per unitat d'angle sòlid	Distribució direccional d'una llum puntual
L	Radiància $\text{W}/(\text{sr} \cdot \text{m}^2)$	Luminància (cd/m^2)	Fluxe per unitat d'àrea i unitat d'angle sòlid	Energia que travessa un punt en una determinada direcció

Figure 55:

Radiancia emitida: $Le(p,w)$ radiancia que emite p hacia la dirección w.

Radiancia incidente: $Li(p,w)$ radiancia que incide en p desde la dirección de la fuente de luz.

Radiancia saliente: $Lo(p,w)$ radiancia que rebota de p hacia w.

14 BRDFs, BTDF and BSDF

$$f(p, \omega_o, \omega_i)$$

BRDF: para cualquier punto i para cualquier par de vectores de entrada (no necesariamente coplanares), de la radiancia que incide en P que radiancia sale en dirección ω_o

R de reflectividad

$$l_i(p, \underline{\omega}_o, \underline{\omega}_i) = \frac{L_o(p, \omega_o)}{L_i(p, \omega_i) \cos \theta_i}$$

Figure 56: Possible Funcion de Reflexion

- $l_i(p, \omega_o, \omega_i) \geq 0$
- $\frac{l_i(p, \omega_o, \omega_i)}{l_i(p, \omega_i, \omega_o)} =$
- $\int_{\Omega} l_i(p, \omega_o, \omega) \cdot \cos \theta \cdot d\omega \leq 1$

Figure 57: Propiedades de la física

BTDF
Transmission

BSDF
Scattering = Reflexion + transmission

15 Textures

Kd Color map

Ks gloss map

Opacitat: opacity map, alpha mask

Normal: normal map (3 canals)

Desplaçament: Bump map (1 canal), displacement mapping (creem geometria)

Mida textura: # texels en cada dimensió, normalment en potencies de 2.

15.1 Mapping

Forward mapping: donada una coordenada de textura, a on hem de posar-la en la imatge 2D.
(Serveix per deformar la imatge)

Inverse mapping: donada una coordenada de la imatge 2D, saber amb quin color de la textura hem de pintar-la.

Mapping esfèric:

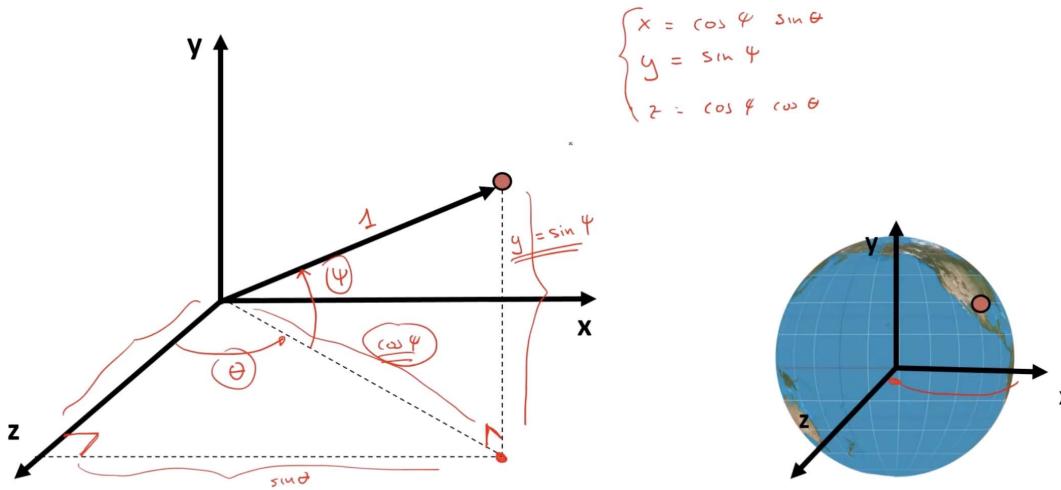


Figure 58:

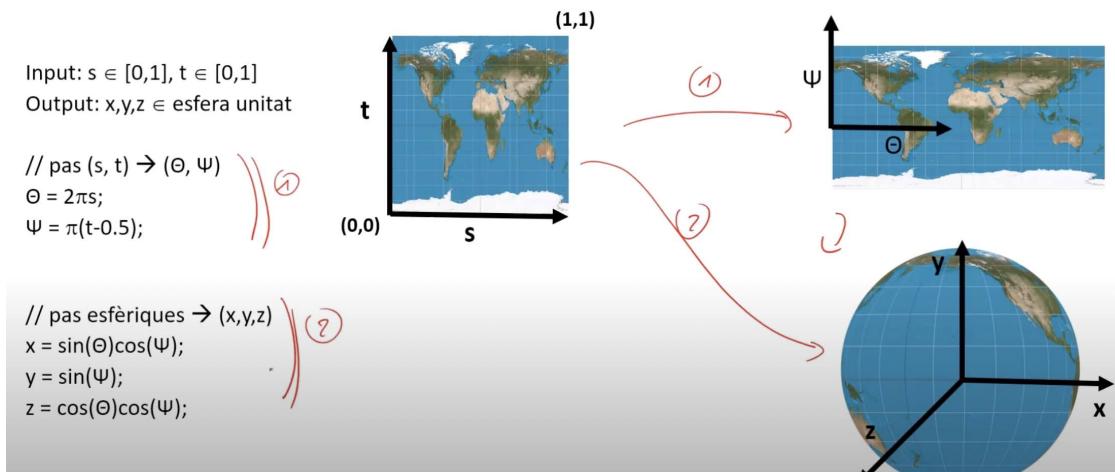


Figure 59:

Es pot modelar en temps de modelat o en CPU (application) en el VBO se crean (s,t) ,o al VS,o al FS.

15.2 Generació de coordenades de textura

Amb plans: V de entrada $(x,y,z,1)$ con dos planos, un plano para S y otro para T.

$S = ax + by + cz + dw$ (con plano S)

$T = ax + by + cz + dw$ (con plano T)

VS:

Te regalamos

15€



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

BBVA está adherido al Fondo de Garantía de Depósitos de Entidades de Crédito de España. La cantidad máxima garantizada es de 100.000 euros por la totalidad de los depósitos constituidos en BBVA por persona.

```
vec4 S = vec4(1, 0, 0, 0);
vec4 T = vec4(0, 1, 0, 0);
vtxCoord.s = dot(S, vec4(vertex, 1.0));
vtxCoord.t = dot(T, vec4(vertex, 1.0));
gl_Position = modelViewProjectionMatrix * vec4(vertex, 1.0);
```

Figure 60:

Amb Superfície auxiliar:

O-Mappings: d'un model passem a un punt en una esfera

Reflected view ray: amb environment mapping

Intermediate surface normal: la normal de la superficie de l'esfera, la projectem cap al model.

Object normal: utilitzem la normal del vertex per saber a on impacta en l'esfera.

Object centroid: Des del centre del model cap a les coordenades del model on volem saber quina textura te, i seguim la direcció del vector per saber on impacta amb l'esfera.

1
Abre tu Cuenta
Online
sin comisiones
ni condiciones

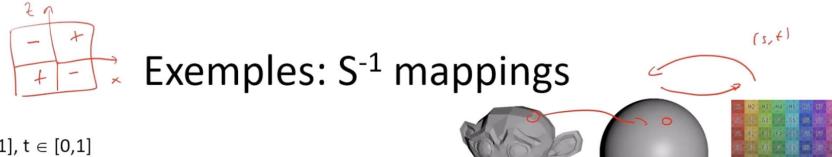
2
Haz una compra
igual o superior
a 15€ con tu
nueva tarjeta

3
BBVA
te devuelve
un máximo de
15€

Input: $s \in [0,1]$, $t \in [0,1]$
Output: $x, y, z \in$ esfera unitat

// pas $(s, t) \rightarrow (\theta, \psi)$
 $\theta = 2\pi s$
 $\psi = \pi(t-0.5)$

// pas esfèriques $\rightarrow (x, y, z)$
 $x = \sin(\theta)\cos(\psi)$
 $y = \sin(\psi)$
 $z = \cos(\theta)\cos(\psi)$



Exemples: S^{-1} mappings

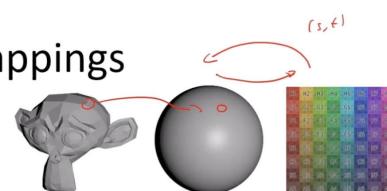


Figure 61:

15.3 Filtrat

Accés a textura: Preimatge d'un fragment: color d'un pixel -> color de la seva preimatge a la textura. Donat un pixel, qué li correspon en la textura?

Magnification: la preimatge és < texel Minification: la preimatge és > texel

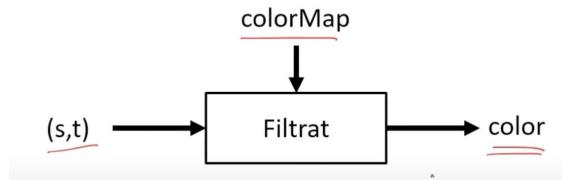


Figure 62:

- Siguin $s(x,y)$, $t(x,y)$ les coordenades s,t del fragment (x,y)
- Derivades parcials de $s(x,y)$:

$$\frac{\partial s}{\partial x} \approx s(x+1, y) - s(x, y)$$

$$\frac{\partial s}{\partial y} \approx s(x, y+1) - s(x, y)$$
- En GLSL es poden calcular amb $dFdx$, $dFdy$:

$$\frac{\partial s}{\partial x} \approx dFdx(\text{texCoord}.s)$$

$$\frac{\partial s}{\partial y} \approx dFdy(\text{texCoord}.s)$$

(anàlogament per t)

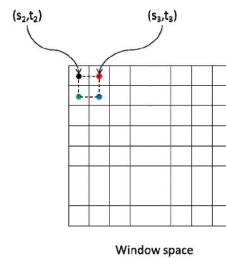
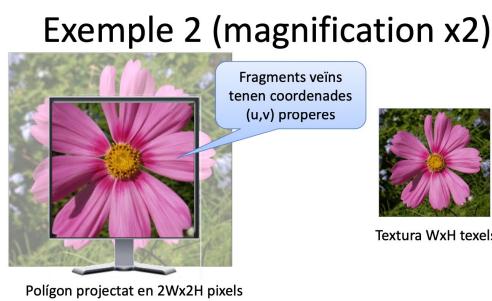


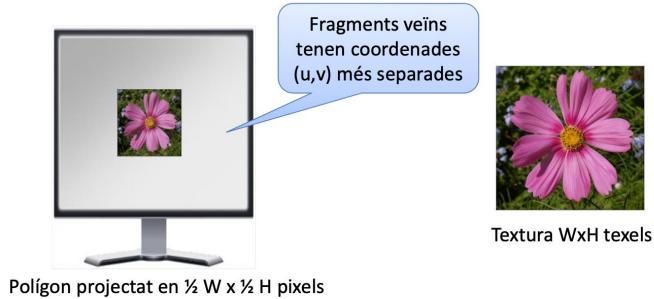
Figure 63:



En aquest cas un pixel correspon a mig texel:
 $\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y} = \frac{1}{2}$ $\frac{\partial v}{\partial x} = \frac{\partial u}{\partial y} = 0$

Figure 64:

Exemple 3 (minification x2)



En aquest cas un pixel correspon a dos texels:

$$\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y} = 2 \quad \frac{\partial v}{\partial x} = \frac{\partial u}{\partial y} = 0$$

Figure 65:

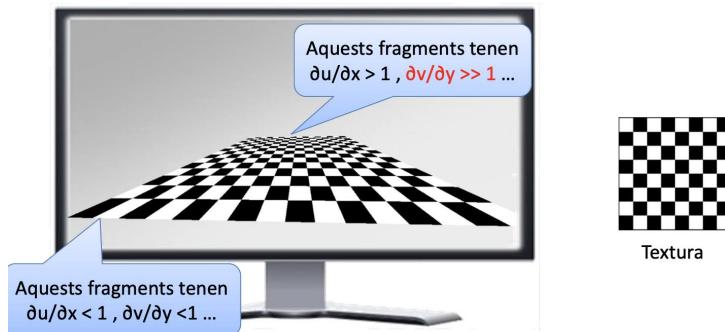


Figure 66:

15.4 Magnification filters

GL NEAREST: nearest neighbor sampling. Pren el color que estigui més a prop de la textura

GL LINEAR: bilinear interpolation. Pren la interpolació entre els 4 veïns més propers.

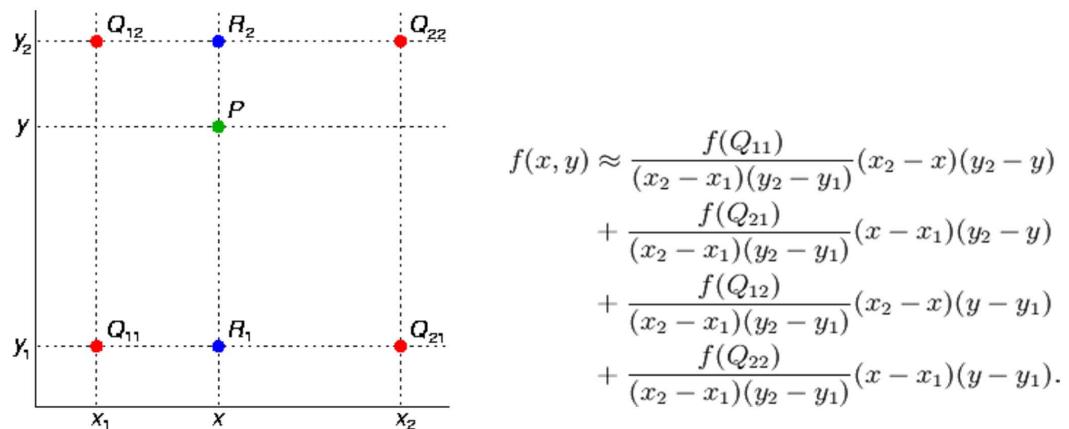


Figure 67: La contribució d'un punt al color depén del tamany del rectangle oposat quan més gran, major serà la contribució

15.5 Minification filters

Mipmapping: textura està representada amb diferents resolucions (LODs: level-of-details).

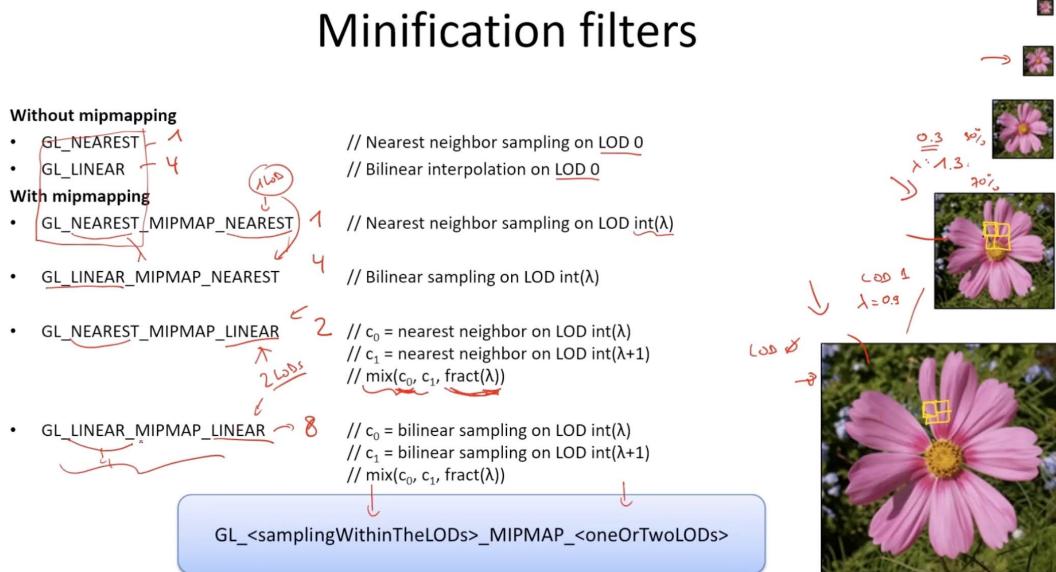


Figure 68:

Te regalamos



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

BBVA está adherido al Fondo de Garantía de Depósitos de Entidades de Crédito de España. La cantidad máxima garantizada es de 100.000 euros por la totalidad de los depósitos constituidos en BBVA por persona.

1

Abre tu Cuenta Online sin comisiones ni condiciones

2

Haz una compra igual o superior a 15€ con tu nueva tarjeta

3

BBVA te devuelve un máximo de 15€

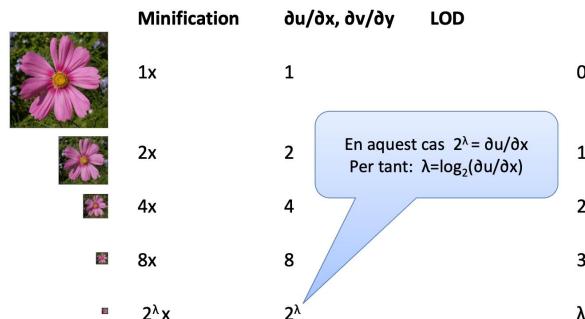


Figure 69:

En general:

- Es calcula un valor $\rho = f(\partial u/\partial x, \partial v/\partial x, \partial u/\partial y, \partial v/\partial y)$
- Es calcula el $\lambda = \log_2(\rho)$

Figure 70:

```
float computeLambda()
{
    const float SIZE = 512.0; // textures 512x512
    vec2 uv = SIZE * vtexCoord;
    float rho = max(length(dFdx(uv)), length(dFdy(uv)));
    return log(rho)/log(2.0);
}
```

Figure 71:

15.6 Wrapping

Modo por defecto: GL REPEAT $\rightarrow s' = \text{fract}(s)$ si accedemos a textura entre 0 i 1 entonces la textura se repetirá 3 veces.

GL CLAMP TO EDGE cuando nos pasamos de 0 a 1 si accedemos más allá de la textura, accederemos a las posiciones de textura del borde.

15.7 Combinació de color de la textura

Replace: fragColor = texColor;

Modulate: fragColor = texColor * frontColor (frontcolor puede incorporar phong, iluminacion).

Decal: fragColor = mix(frontColor, texColor, texColor.a) (depen de la component alfa)
if (texcolor.a < alphaThreshold) discard;

15.8 Perspective correct interpolation (textures)

15.9 Projective texture mapping

15.10 Mappings

```

program->setUniformValue("tiles", int(1));
// bind textures
g.glActiveTexture(GL_TEXTURE0);
g glBindTexture(GL_TEXTURE_2D, textureId0);
// setup texture parameters
GLenum wrapMode = GL_REPEAT;
g glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, wrapMode);
g glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, wrapMode);
g glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR );

```

Figure 72:

Solucions:

- a) Interpolem (s, t) en **object space** (també valdria en **world, eye i clip space**, perquè són abans de la div. de perspectiva)
- b) Interpolem (sw, tw, w) en **window space**, obtenint un texel (s, t, q); accedirem a la textura amb $(s/q, t/q)$

Recordeu que $w = 1/w_c = -1/z_e$

Perspecti·

Figure 73:

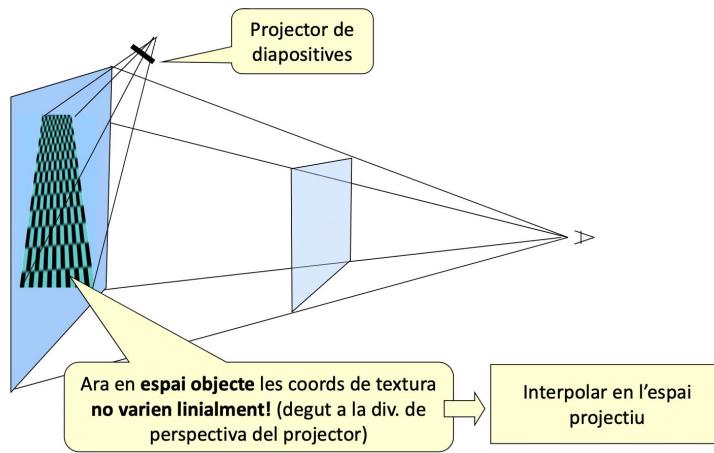


Figure 74:

Opció 1 (incorrecta) - 1..1

VS – generació coords de textura

Passa el vèrtex de object space a window space (viewport 1x1)

Calcula (s,t) com (x,y) del resultat

$$\left\{ \begin{array}{l} \text{Diagram showing vertex transformation from object space to window space.} \\ \text{Object space vertex } \begin{bmatrix} s \\ t \\ z \\ 1 \end{bmatrix} \text{ is transformed by matrix } T(0.5) \cdot S(0.5) \text{ to intermediate coordinates } \begin{bmatrix} x_c/v_w \\ y_c/v_w \\ z_c/v_w \\ v_w \end{bmatrix}. \\ \text{These are then transformed by matrix } P_p \cdot V_p \cdot M \text{ to final coordinates } \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \text{ in view space.} \end{array} \right.$$

FS – accés a textura

Usa (s,t) per accedir a la textura

Figure 75:

Opció 2 (correcta)

VS – generació coords de textura

Passa el vèrtex de object space a window space (viewport 1x1)

però sense aplicar la div de perspectiva.

Calcula (s,t,p,q) com (x,y,z,w) del resultat

$$\left\{ \begin{array}{l} \text{Diagram showing vertex transformation from object space to window space.} \\ \text{Object space vertex } \begin{bmatrix} s \\ t \\ z \\ 1 \end{bmatrix} \text{ is transformed by matrix } T(0.5) \cdot S(0.5) \text{ to intermediate coordinates } \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix}. \\ \text{These are then transformed by matrix } P_p \cdot V_p \cdot M \text{ to final coordinates } \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \text{ in view space.} \\ \text{A separate row shows the resulting coordinates } \begin{bmatrix} 1 & 1 & 1 & \frac{0.5}{0.5} \\ 0.5 & 0.5 & 0.5 & 1 \end{bmatrix} \text{ which are used for texture access.} \end{array} \right.$$

FS – accés a textura

Usa (s/q, t/q) per a accedir a la textura

Figure 76:

Tècnica	Info a la textura	Ús de la textura	Coords de textura	View parallax	Self-occlusion	Detailed silhouette	On s'aplica
Color mapping	RGB 3	Kd del material	(s,t)	-	-	-	FS
Bump mapping	D 1	Modificar la normal	(s,t)	N	N	N	FS
Normal mapping	Normal 3	Modificar la normal	(s,t)	N	N	N	FS
Parallax mapping	Normal + D 3+1 o 4	Modificar la normal	(s+d _s , t+d _t)	S	N	N	FS
Relief mapping	Normal + D 3+1 o 4	Modificar la normal; descartar fragments	(s+d _s , t+d _t)	S	S	S	FS!!!
Displacement mapping	D 1	Desplaçar els vèrtexs un cop subdividits els polígons.	(s,t)	S	S	S	CPU GS TCS+TES

Figure 77:

Bump mapping/Normal mapping

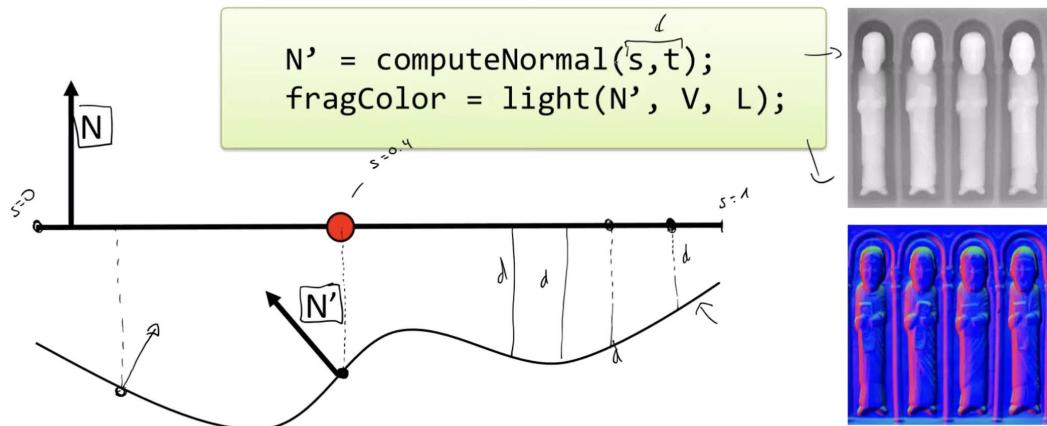


Figure 78:

Te regalamos

15€

WUOLAH
+ BBVA



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

BBVA está adherido al Fondo de Garantía de Depósitos de Entidades de Crédito de España. La cantidad máxima garantizada es de 100.000 euros por la totalidad de los depósitos constituidos en BBVA por persona.

1

Abre tu Cuenta Online sin comisiones ni condiciones

2

Haz una compra igual o superior a 15€ con tu nueva tarjeta

3

BBVA te devuelve un máximo de 15€

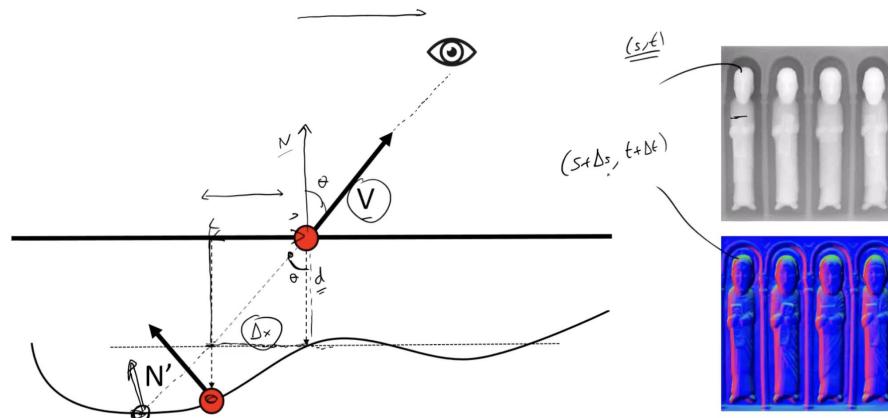


Figure 79:

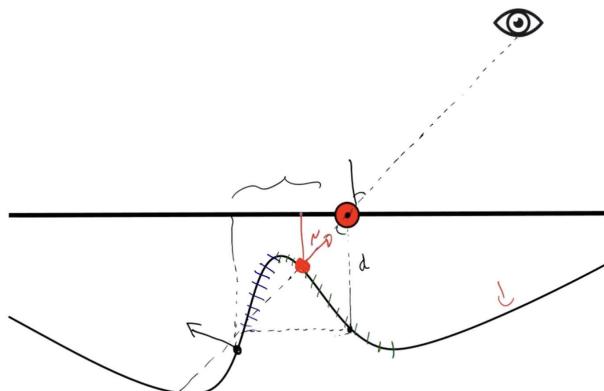


Figure 80:

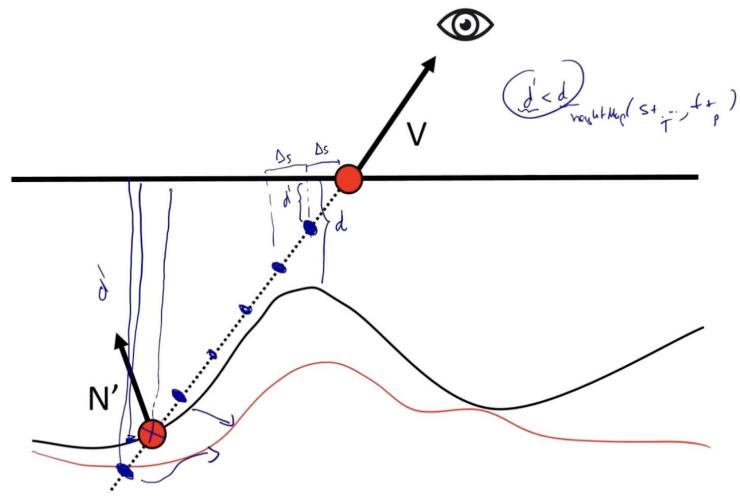


Figure 81: ReliefMap use dichotomic search to find the spot where $d' = d$

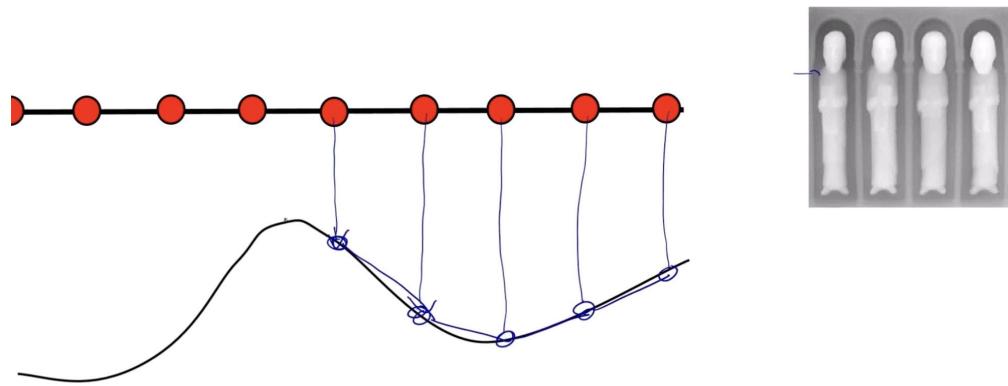


Figure 82: DisplacementMap, afegeix molts vèrtexs i després els desplaça segons el map

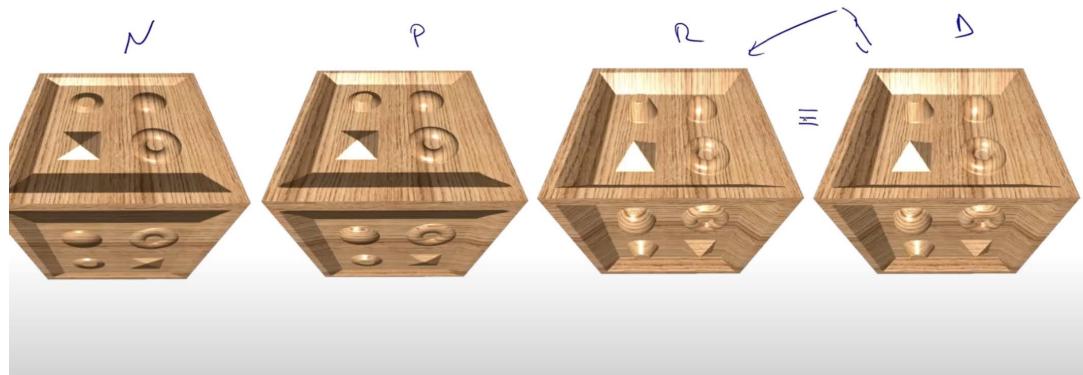


Figure 83: