

RenderGPU

Segona pràctica de GiVD 2020-21

Grup F04: Nil Ballús, Sara Bardají, David Rial i Albert Sallés

Abstract

En aquesta pràctica hem implementat una visualització projectiva d'escenes amb l'algorisme *ZBuffer* usant *shaders*. A més, hem implementat la il·luminació d'escenes amb diferents models locals *shading*: *Gouraud* i *Phong*, tenint en compte textures (tant amb mapeig directe com indirecte) en aquest últim. De forma addicional, també hem inclòs la il·luminació de toon-shading com a no realista. A més, aquests diferents *shaders* es poden activar en temps d'execució, de manera que, a través d'una interfície gràfica, es pot escollir el tipus d'il·luminació.

Per fer les visualitzacions d'escenes hem adaptat les funcionalitats de la lectura de fitxers fetes en la pràctica anterior. Concretament, la visualització de les scenes es pot realitzar a través de la lectura de fitxers VIRTUALS (*ConfigMapping* i *Data*, només amb objectes de tipus BR-Objectes) i a través de la lectura de fitxers REALDATA (*ConfigMapping* i *Data*, on hi ha dades geolocalitzades).

Features

- Fase 1
 - Adaptació a la lectura de fitxers de dades: Nil
 - [x] Objectes
 - [x] Escenes virtuals
 - [x] Escenes de dades Reals
 - Material: Albert i Sara
 - Light: David
 - [x] Puntual
 - [x] Direccional
 - [x] Spotlight
 - [x] Ambient Global
 - Shading: David i Nil

- [x] Phong
- [x] Gouraud
- Textures: Albert i Sara
 - [x] Textura com material en un objecte
 - [x] Textura al pla base
- Fase 2 (OPT)
 - [x] Toon-shading i èmfasi de siluetes: David i Nil
 - [x] Mapping indirecte de textures: Albert i Sara
 - [] Animacions amb dades temporals
 - [] Normal mapping
 - [] Entorn amb textures
 - [] Reflexions
 - [] Transparencies via objectes.
 - [] Transparencies via environmental mapping.

Extensions

- [x] Lectura de materials des del .obj i parsejat del fitxer mtl: Sara
- [x] Llums de tipus Spotlight: David
- [x] Èmfasi de siluetes: David i Nil

Memòria

- Adaptació a la lectura de fitxers de dades:

Per implementar la lectura de fitxers de dades hem partit de les classes del paquet *DataService* de la pràctica anterior i hem adaptat el codi a les noves necessitats. A diferència de la pràctica anterior, en aquest cas per visualitzar una escena només s'han d'introduir els fitxers de *ConfigMapping* i *Data*. El fitxer *ConfigVis* ja no es considera perquè els paràmetres de la càmera es poden canviar interactivament des de la interfície gràfica. A continuació es detalla el format dels fitxers txt que s'utilitzen per fer les visualitzacions:

Fitxers per a escenes virtuals:

- Fitxer *ConfigMapping*:
 - dataType, VIRTUALWORLD
 - limitsMonVirtual, xmin, xmax, ymin, ymax, zmin, zmax
 - limitsMonReal, xmin, xmax, ymin, ymax, zmin, zmax
 - globalLight, Ia.R, Ia.G, Ia.B (intensitat ambient global de l'escena)
 - light, typeLight, Ia.R, Ia.G, Ia.B, Id.R, Id.G, Id.B, Is.R, Is.G, Is.B, {a, b, c},

posx/dirx, posy/dirz, posz/dirz, {opc1, opc2, opc3, opc4}

- typeLight pot ser pointLight l directionalLight l spotLight
- En el cas de pointLight, els paràmetres {a, b, c} sí que s'hi inclouen i són els coeficients d'atenuació. Els paràmetres {opc1, opc2, opc3, opc4} no s'hi inclouen i els paràmetres posx/dirx, posy/dirz, posz/dirz fan referència a la posició de la llum.
- En el cas de directionalLight, els paràmetres {a, b, c} no s'hi inclouen. Els paràmetres {opc1, opc2, opc3, opc4} no s'hi inclouen i els paràmetres posx/dirx, posy/dirz, posz/dirz fan referència a la direcció de la llum.
- En el cas de spotLight, els paràmetres {a, b, c} sí que s'hi inclouen i són els coeficients d'atenuació. Els paràmetres {opc1, opc2, opc3, opc4} sí que s'hi inclouen. opc1, opc2, opc3 fan referència a la direcció de la llum i opc4 fa referència a l'obertura de la llum. Els paràmetres posx/dirx, posy/dirz, posz/dirz fan referència a la posició de la llum.

- toonDirection, dirx, diry, dirz
 - Direcció utilitzada per la visualització amb *toon-shading*. En el cas de no incloure's aquesta línia s'utilitza una direcció per defecte.

◦ Fitxer *Data*:

- brobject, centre.x, centre.y, centre.z, radi, path obj, path textura, direct/indirect, {Ka.R, Ka.G, Ka.B, Kd.R, Kd.G, Kd.B, Ks.R, Ks.G, Ks.B, shininess}
 - path obj és el path del fitxer .obj on es descriu l'objecte.
 - path textura és el path del fitxer de textura. També es pot posar en aquest camp NULL i llavors s'utilitza una textura per defecte (brick).
 - direct/indirect indica el tipus de mapeig en les textures. Per poder utilitzar l'opció direct cal que en el fitxer .obj hi hagi indicades les coordenades dels vèrtexs de textura.
 - {Ka.R, Ka.G, Ka.B, Kd.R, Kd.G, Kd.B, Ks.R, Ks.G, Ks.B, shininess} són paràmetres opcionals. Si s'hi inclouen descriuen el material de l'objecte. Si no s'hi inclouen, el material de l'objecte serà el que estigui definit al fitxer .obj i si no hi està definit serà un color per defecte.

Fitxers per a escenes a partir de dades reals:

◦ Fitxer *ConfigMapping*:

- El format és el mateix que per a escenes virtuals amb el següent canvi:
 - Per entrar una prop es fa de la següent manera: prop, numProp, vmin, vmax, brobject, path obj, color_map, path textura, direct/indirect
 - path obj és el path del fitxer .obj on es descriu l'objecte.
 - path textura és el path del fitxer de textura. També es pot posar en

aquest camp NULL i llavors s'utilitza una textura per defecte (brick).

- direct/indirect indica el tipus de mapeig en les textures. Per poder utilitzar l'opció direct cal que en el fitxer .obj hi hagi indicades les coordenades de textura.
- Per entrar la base (pla) del mapeig es fa de la següent manera: base, plane, nx, ny, nz, d, Ka.R, Ka.G, Ka.B, Kd.R, Kd.G, Kd.B, Ks.R, Ks.G, Ks.B, shininess, path textura, direct/indirect
 - path textura és el path del fitxer de textura. També es pot posar en aquest camp NULL i llavors s'utilitza una textura per defecte (brick).
 - direct/indirect indica el tipus de mapeig en les textures.

- Fitxer *Data*:

- data, x, z, valor

Cal destacar que a part de les llums introduïdes des del fitxer *ConfigMapping* sempre es crea una llum addicional de tipus pointLight que serà manejable des de la interfície gràfica. En cas de no voler considerar aquesta llum addicional, simplement s'han de posar els seus atributs a 0 des de la interfície gràfica.

A continuació, detallem com es fa per obrir escenes virtuals i escenes a partir de dades reals:

- Per obrir escenes virtuals, des de la interfície gràfica s'ha de seleccionar l'opció *Obre Escena (Ctrl E)*. A continuació, s'ha de seleccionar el fitxer *ConfigMapping* i un cop obert s'ha d'obrir el fitxer de dades.
- Per obrir escenes a partir de dades reals, des de la interfície gràfica s'ha de seleccionar l'opció *Obre Dades (Ctrl D)*. A continuació, s'ha de seleccionar el fitxer *ConfigMapping* i un cop obert s'ha d'obrir el fitxer de dades.

- Materials:

Per la implementació dels materials hem creat una nova classe *Material* que conté els valors necessaris per determinar un material: les components ambient, difosa i especular, i la shininess. Cada objecte té, doncs, un material associat. Per fer això, es fa de la següent manera:

Si en el fitxer de dades hi ha indicats els atributs del material, se li assigna el material amb aquests atributs a l'objecte. Si en el fitxer de dades no hi ha indicats els atributs del material i en el .obj s'especifica un fitxer .mtl, aquest fitxer es parseja per assignar aquest nou material a l'objecte en qüestió. Si el fitxer de material en conté més d'un, aleshores el material que s'assigna és l'últim material que es parseja. En canvi, si el fitxer de dades no s'especifica el material i en el fitxer .obj no s'especifica un fitxer .mtl,

s'assigna un material per defecte a l'objecte.

En el cas de visualitzar dades reals, el material de les dades s'assigna segons el valor que representa l'objecte, seguint la paleta de colors especificada en el *ConfigMapping*.

- Llums:

Per la implementació de les llums hem creat una jerarquia de classes (Light <- PointLight | DirectionalLight | SpotLight). Tal i com hem explicat en el primer apartat es poden introduir llums a les escenes a partir del fitxer *ConfigMapping* seguint el format detallat. A més, addicionalment sempre es crea una llum de tipus PointLight per defecte que és manejable des de la interfície gràfica.

- PointLight: representa una llum puntual. És a dir, una llum que està situada a una posició concreta i que emet llum en totes direccions.
- DirectionalLight: representa una llum direccional. És a dir, una llum sense posició (es considera situada a l'infinit) que emet llum en una única direcció.
- Spotlight: representa una llum de tipus *Spot Light*. És a dir, una llum que està situada a una posició concreta, que emet llum en una direcció concreta i que té un angle d'obertura associat.

Per últim, la intensitat ambient global de l'escena també es pot introduir a través del fitxer *ConfigMapping*. En cas de no introduir-la, es posa una intensitat ambient global per defecte a l'escena.

- *Shadings*:

Per tal d'implementar els dos primers tipus de *shadings* (*Gouraud* i *Phong*) hem utilitzat el model de *Blinn-Phong*. En canvi, *toon-shading* es correspon amb un *shading* discret segons el producte escalar del vector de la llum direccional amb la normal al punt.

- *Gouraud*: en aquest cas el model de *Blinn-Phong* es calcula a nivell de vèrtexs, és a dir, al *vertex shader*. Llavors, el color resultant a cada *fragment* es calcula interpolant el valor de *Blinn-Phong* obtingut als vèrtexs.
- *Phong*: en aquest cas el model de *Blinn-Phong* es calcula a nivell de *fragments*, és a dir, al *fragment shader*. En aquest cas, a cada *fragment* s'interpolava la normal dels vèrtexs i es calcula el color del *fragment* seguint el model de *Blinn-Phong* utilitzant la normal interpolada.
- *Toon-shading*: aquest tipus de *shading* es correspon amb una tècnica no realista. Per implementar aquesta tècnica, hem decidit utilitzar 4 tipus de colors. El color de cada *fragment* es determina segons el producte escalar entre el vector de la llum direccional i la normal al punt (que serà la normal interpolada dels vèrtexs). Llavors, segons el valor del producte escalar assignem una tonalitat diferent (més clara o més fosca) de la component difosa del material. A més, en aquest *shading*

també s'hi inclou l'èmfasi de siluetes, que suposa més èmfasi en els contorns de les projeccions 2D dels objectes que s'estan visualitzant.

Tal i com hem comentat al primer apartat, la direcció (de la llum direccional) utilitzada per visualitzar l'escena amb *toon-shading* es pot introduir des del fitxer *ConfigMapping*. En cas de no especificar-la, s'utilitzarà una direcció per defecte.

Per últim, cal destacar que en iniciar-se el programa s'utilitza un *shader* auxiliar, on el color es calcula segons la component difosa del material. Per activar els *shaders* explicats anteriorment s'ha de fer interactivament a través de la interfície gràfica (a l'apartat *Shaders* del menú superior).

- Textures:
 - Textura com a material en un objecte:

Per implementar l'ús de textures com a material en un objecte hem començat fent servir el mapejat directe. En aquest cas, hem hagut de llegir les coordenades dels vèrtexs de textura del fitxer *.obj* i guardar-les en l'objecte. Com que aquest mapejat requereix que el fitxer *.obj* contingui els vèrtexs de textura, no es pot fer aquest mapeig amb qualsevol objecte. Un cop llegits els vèrtexs de textura, obtenim les coordenades de textura i aquestes les passem al *fragment shader*, juntament amb la textura. Aquestes dades les fem servir pel càlcul de la component difosa de l'algorisme de *Phong*.

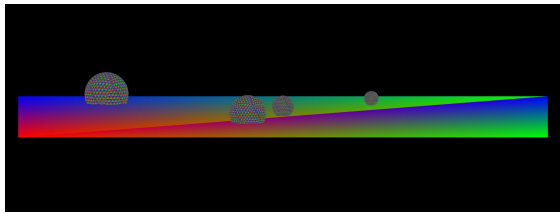
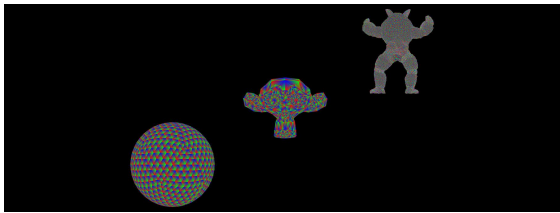
A més a més, per poder tenir textures en tots els objectes, hem implementat també l'ús de textures amb mapeig indirecte. En aquest cas, passem la textura però no necessitem les coordenades de textura, ja que aquestes les calculem en el mateix *fragment shader*. En el nostre cas, mapegem la textura sobre una esfera imaginària que embolcalla l'objecte i projectem els valors dels colors sobre aquest.

- Textura al pla base:

La textura en el pla base funciona igual que en els fitxers *.obj*. En el cas del mapejat directe, omplim nosaltres els vèrtexs de textura. En el cas del mapejat indirecte no hi ha cap diferència.

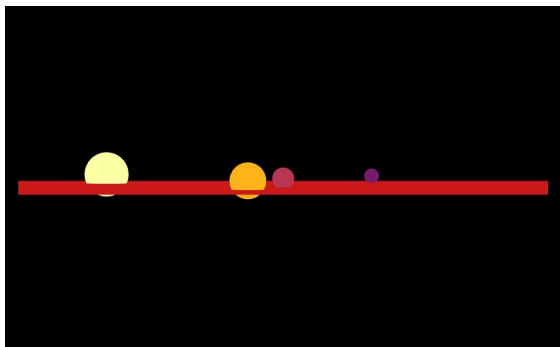
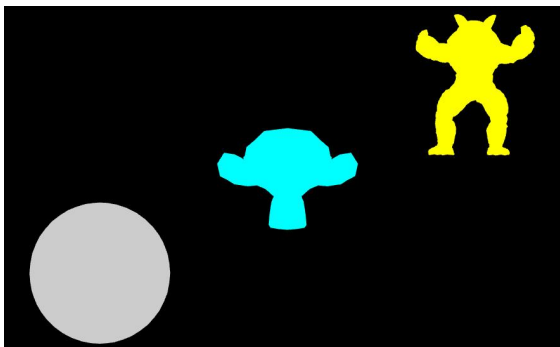
Screenshots

- **Pas 1.** Adaptació de la lectura dels fitxers de dades:

Imatge amb dades reals	Imatge amb dades virtuals
	
Fitxers escena: <i>dades/configMappingDataescena1.txt_ i</i> <i>dades/dataescena1.txt_</i>	Fitxers escena: <i>escenes/configMappingescena1.txt_ i</i> <i>escenes/dataescena1.txt_</i>

- **Pas 2.** Modificació de la classe Material i pas a la GPU:

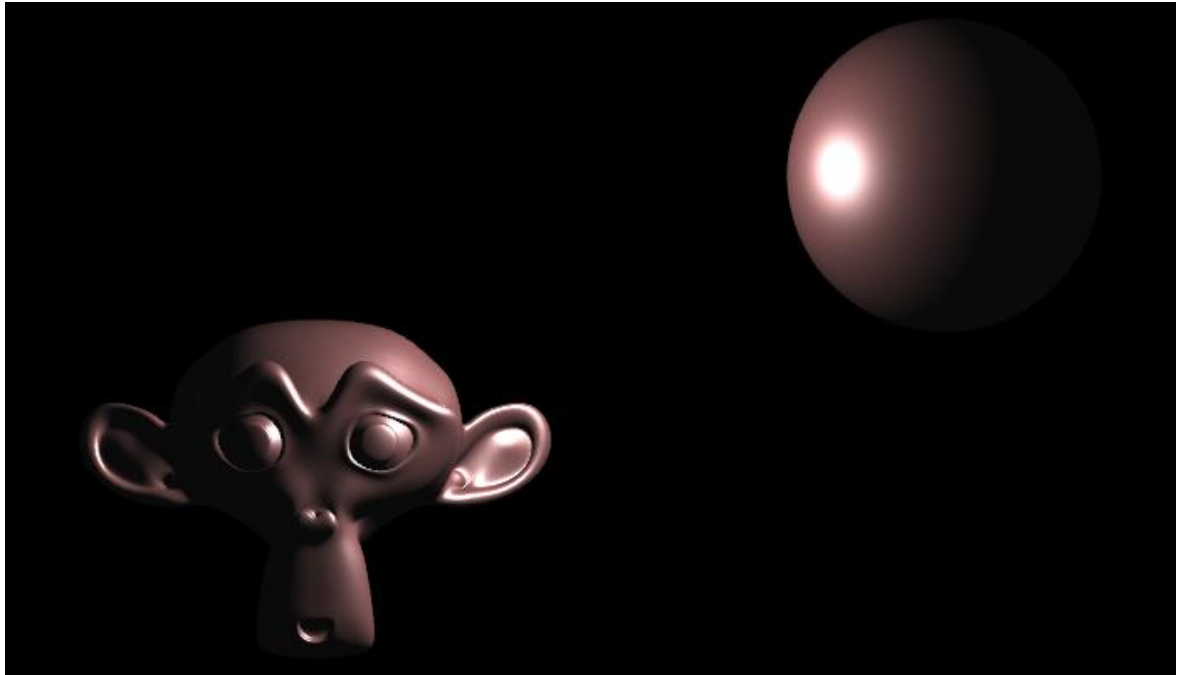
En aquest pas hem implementat la càrrega de materials per a cada objecte. En les dades virtuals aquests es poden llegir des del fitxer de dades o des del fitxer *.mtl* (com podem veure en l'esfera), mentre que a les dades reals s'assignen materials segons una paleta de colors associada al valor de la propietat que representen.

Imatge amb dades reals	Imatge amb dades virtuals
	
Fitxers escena: <i>dades/configMappingData_escena1.txt i</i> <i>dades/data_escena1.txt</i>	Fitxers escena: <i>escenes/configMapping_escena1.txt i</i> <i>escenes/data_escena1.txt</i>

- **Pas 3.** Modificació de la classe Llum i pas a la GPU:

En aquest pas hem implementat els diferents tipus de llums i el seu pas a la GPU. Cal destacar que les captures han estat creades després d'implementar el pas 4, utilitzant el tipus de *shading Phong*, ja que d'aquesta manera s'aprecia millor el comportament de cada llum.

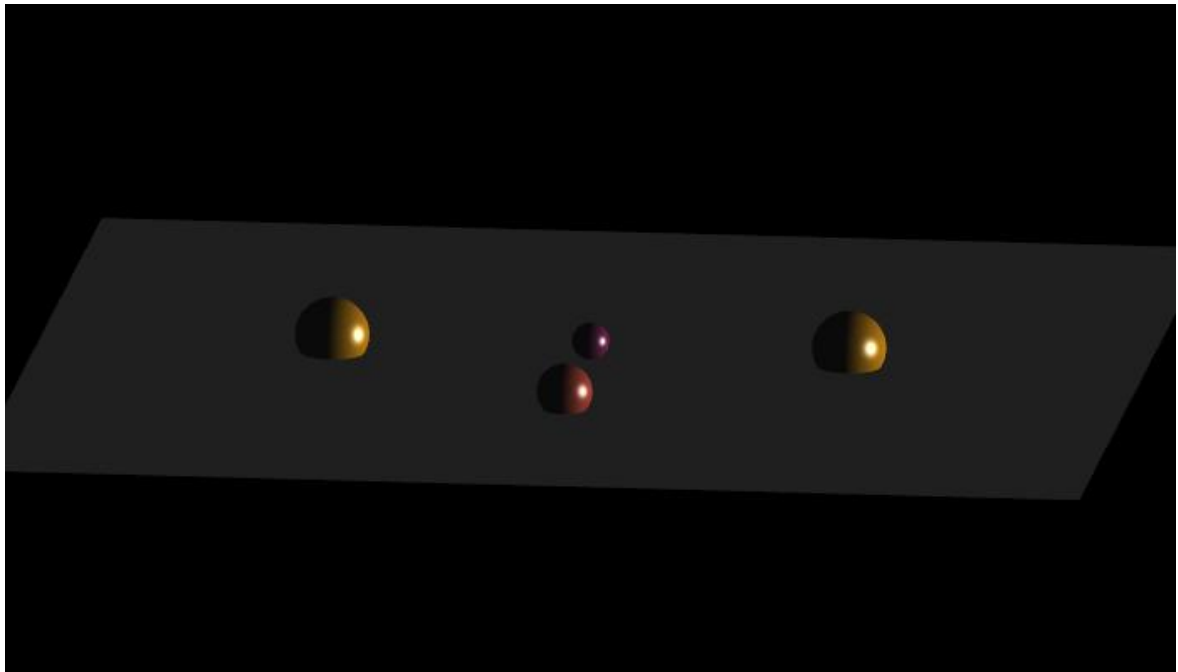
Point Light (Escena Virtual)



Fitxers escena: *escenes/configMappingData_escena3.txt* i *escenes/data_escena3.txt*.

Observació: s'ha mogut la llum puntual des de la interfície gràfica.

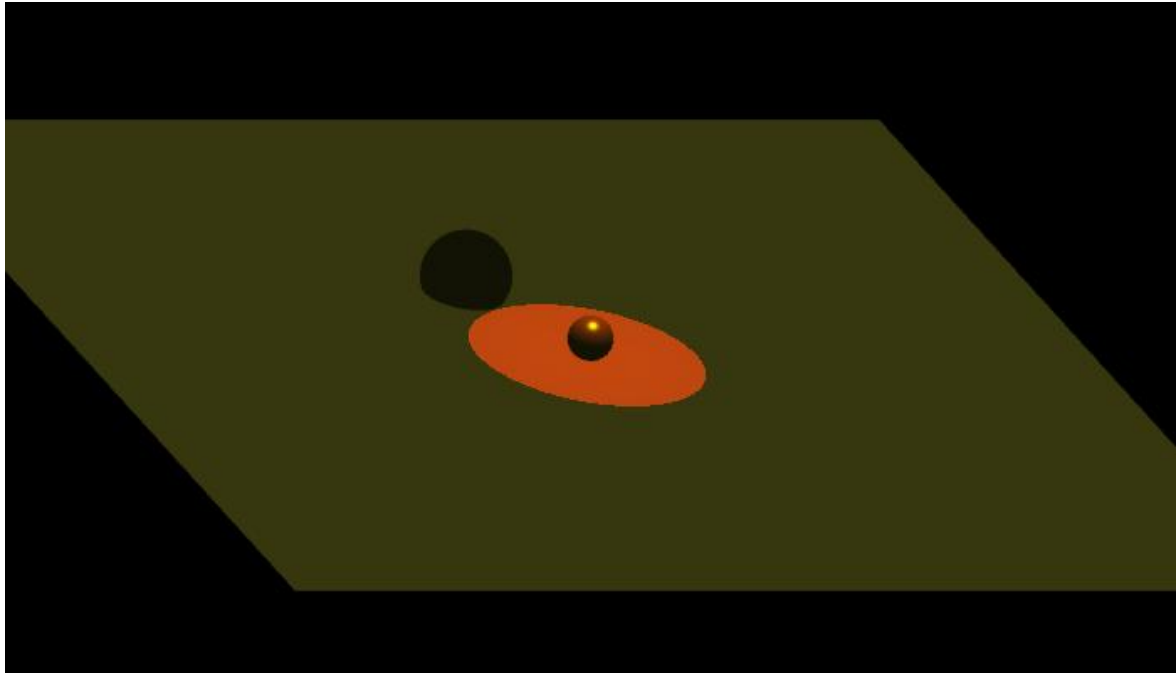
Directional Light (Escena a partir de dades reals)



Fitxers escena: *dades/configMappingData_escena4.txt* i *dades/data_escena4.txt*.

Observació: s'han posat tots els atributs de la llum puntual a 0 des de la interfície gràfica.

Spot Light (Escena a partir de dades reals)

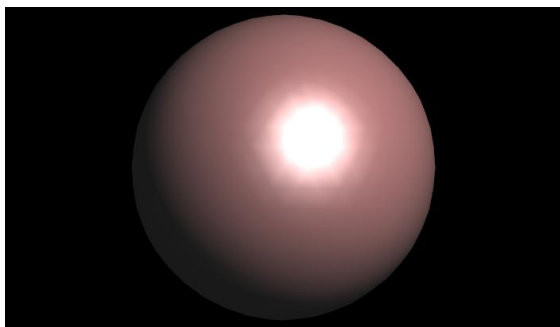
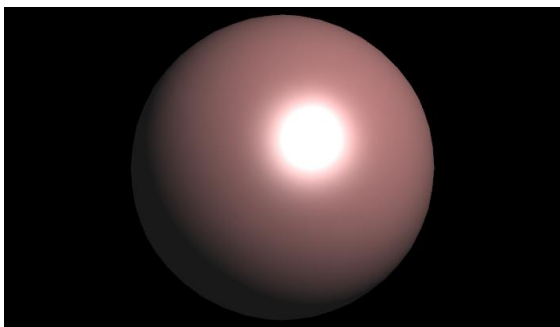


Fitxers escena: *dades/configMappingData_escena3.txt* i *dades/data_escena3.txt*.

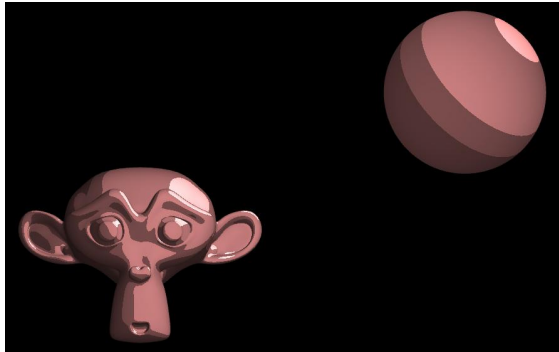
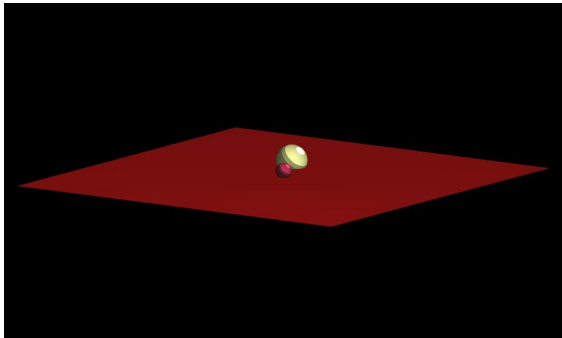
Observació: s'han posat tots els atributs de la llum puntual a 0 des de la interfície gràfica.

- **Pas 4.** Implementació dels diferents tipus de *shading* (*Gouraud* i *Phong*):

Escenes de test proposades al Campus Virtual:

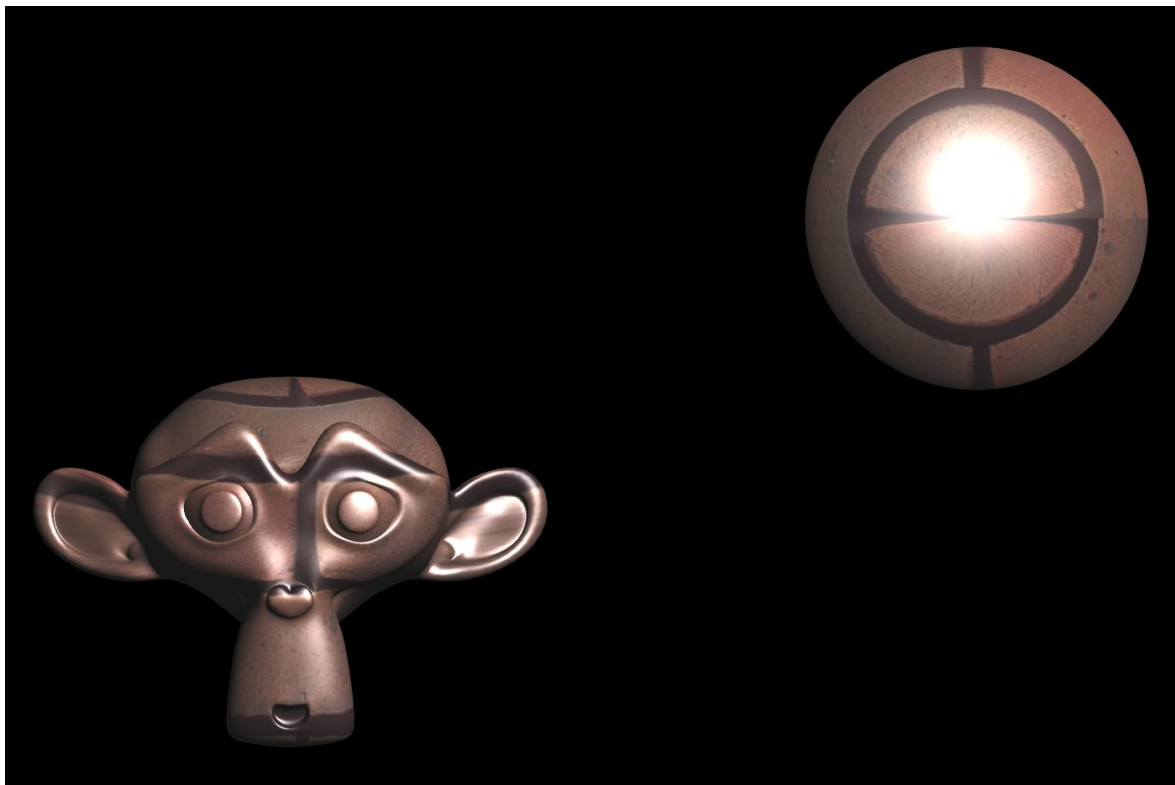
Gouraud	Phong
	
<p>Fitxers escena: <i>models/sphere1.obj</i>. Aquesta escena s'ha obtingut fent directament Obre Fitxer .obj</p>	<p>Fitxers escena: <i>models/sphere1.obj</i>. Aquesta escena s'ha obtingut fent directament Obre Fitxer .obj</p>

Toon-shading amb èmfasi de siluetes

Escena Virtual	Dades Reals
	
Fitxers escena: <i>escenes/configMapping_escena3.txt i</i> <i>escenes/data_escena3.txt</i>	Fitxers escena: <i>dades/configMapping_escena3.txt i</i> <i>dades/data_escena3.txt</i>

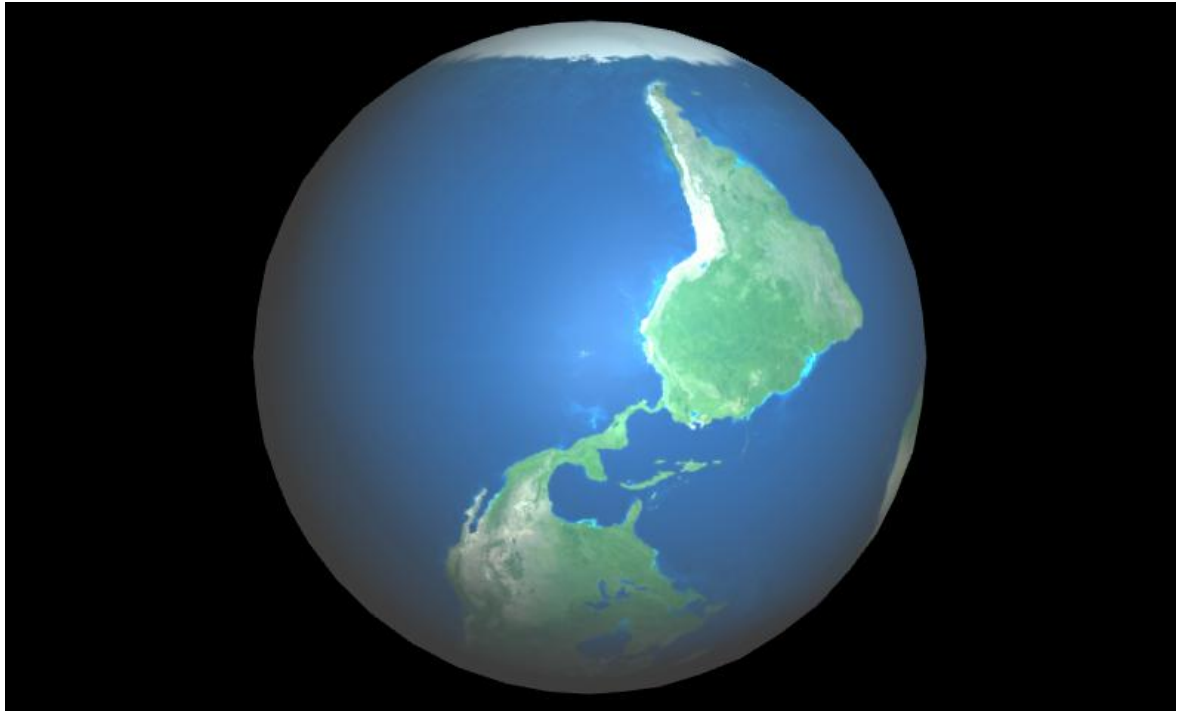
- **Pas 5. Textures:**

Mapping directe de textures: el mapejat de la textura a l'objecte es fa utilitzant els vèrtexs proporcionats en el fitxer *.obj*.

Escena virtual amb textura (mapejat directe)

Fitxers escena: <i>escenes/configMapping_escena3.txt i</i> <i>escenes/data_escena3.txt</i>

Mapping indirecte de textures: el mapejat de la textura a l'objecte es fa de forma indirecte.

Escena virtual amb textura (mapejat indirecte)



Fitxers escena: *escenes/configMapping_escena6.txt* i *escenes/data_escena6.txt*

Mapping directe vs. Mapping indirecte: seguidament, mostrem el resultat de la comparació del mapejat directe i indirecte d'un mateix objecte utilitzant la mateixa textura.

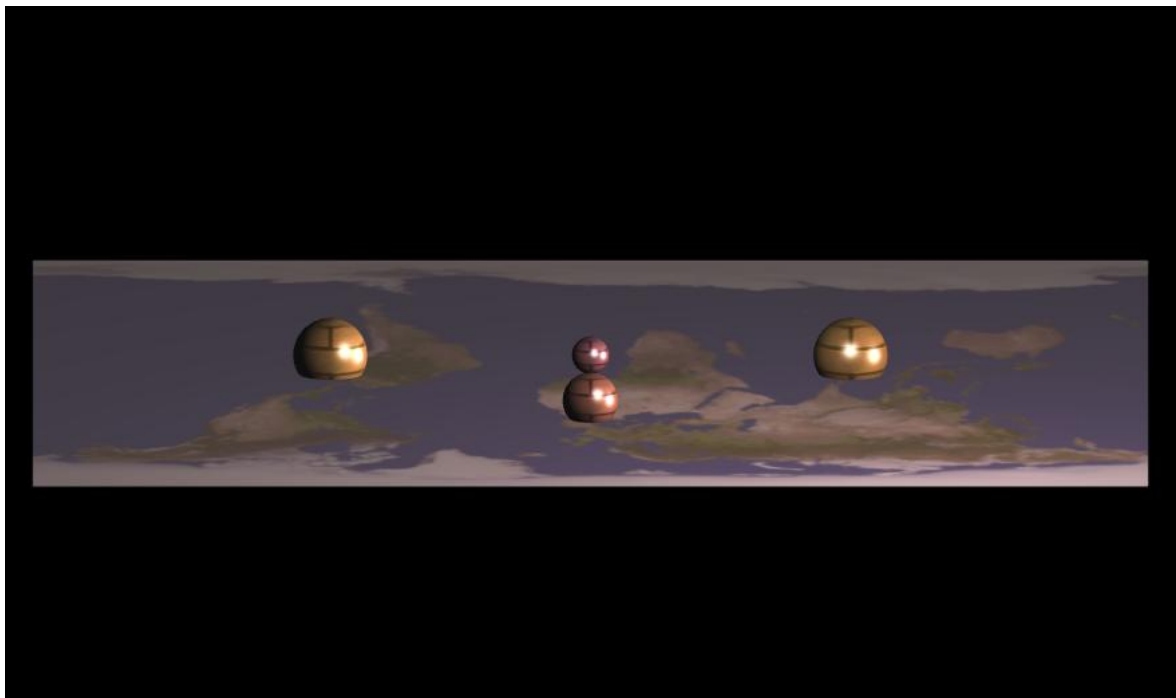
Comparació de mapejat indirecte (esquerra) i directe (dreta).



Fitxers escena: *escenes/configMapping_escena5.txt* i *escenes/data_escena5.txt*

Textura pla base

Pla base amb textura



Fitxers escena: *dades/configMapping_escena4.txt* i *dades/data_escena4.txt*

Additional Information

En la realització d'aquesta pràctica hi hem dedicat diverses hores cada setmana, ja que requeria l'assimilació de conceptes nous que no havíem vist fins ara. Per aquest motiu, a part d'assistir a classe i veure els vídeos, fer una assimilació dels conceptes era imprescindible. A més, com que no havíem treballat mai amb la *GPU*, també vam haver de fer una exploració extensa dels projectes de mostra que se'ns van donar, que van ser molt útils. Finalment, vam intentar avançar la feina cada setmana, de forma regular, treballant en grup i dividint-nos la feina tant bé com podíem.