

Mirror (mirror.*)

2.5 punts

Escriu VS+FS shaders per aplicar una textura al model, de forma que la meitat superior/inferior de la textura aparegui reflectida respecte la línia horitzontal que passa pel centre de la textura:



textura



time = 0.25



time=3.75

L'exercici està pensat per l'objecte **plane.obj**, amb coordenades de textura entre 0 i 1.

El **VS** farà les tasques habituals que resultin imprescindibles per a aquest exercici.

El **FS** serà l'encarregat d'aplicar la reflexió de la textura, modificant les coordenades de textura del fragment de forma adient per aconseguir l'efecte demanat. El color resultant serà directament el color de la textura, usant les coordenades de textura modificades. Podeu aprofitar que el viewer usa el mode de wrapping `GL_REPEAT`.

La reflexió a aplicar dependrà de la variable **time**.

Quan la part fraccionària de time sigui **menor o igual a 0.5**, només es farà servir la **meitat inferior** de la textura, que apareixerà al model tal qual o reflectida, depenent de la coordenada t (vtexCoord.t).

Altrament, només es farà servir la **meitat superior** de la textura, que apareixerà al model tal qual o reflectida, depenent de la coordenada t, com al cas anterior.



Identificadors (ús obligatori):

```
mirror.vert, mirror.frag  
uniform sampler2D colorMap;  
uniform float time;
```

Crèdit imatges:

Bill Gates: <http://www.thefamouspeople.com/profiles/images/bill-gates-1.jpg>

Starfish: https://pixabay.com/p-732391/?no_redirect

Zoom (zoom.*)

2.5 punts

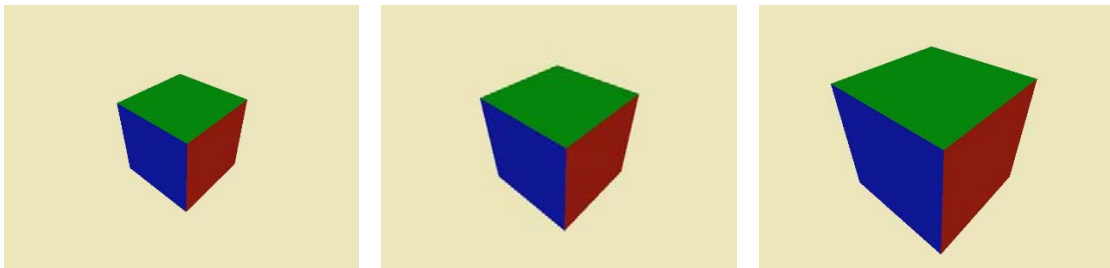
Escriu VS+FS per simular un zoom automàtic que ampliï i redueixi l'objecte a mesura que passi el temps.

El VS serà l'encarregat d'aplicar aquest zoom, que implementarem **desplaçant el vèrtex en eye space** per apropar-lo o allunyar-lo de la càmera.

Sigui **r** la **meitat de la diagonal** de la capsula englobant de l'escena. La distància a apropar o allunyar l'objecte respecte la càmera ha de variar segons una sinusoidal **sin(time)** amb **amplitud 0.5r**. **Recordeu que us demanem que apliqueu la translació en eye space**. El color del vèrtex serà simplement el color original multiplicat per la N.z en eye space.

El FS farà les tasques per defecte (que siguin imprescindibles per al resultat demanat).

Aquí teniu un exemple (vegeu també el vídeo):



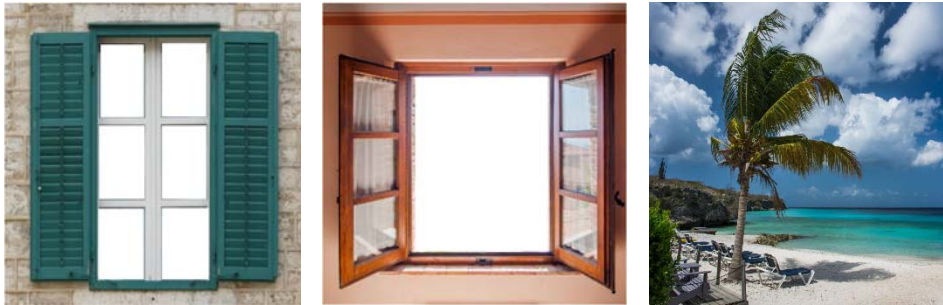
Identificadors (ús obligatori):

```
zoom.vert, zoom.frag  
uniform float time;  
uniform vec3 boundingBoxMin;  
uniform vec3 boundingBoxMax;
```

Magic window (magic.*)

2.5 punts

Escriu un VS i un FS per tal de simular una finestra a través de la qual es pot veure l'interior d'una habitació, i un altre cop l'exterior. A /assig/grau-g/Textures trobareu les textures **window.png**, **interior.png** i **exterior.png**:



El VS, a banda de les tasques habituals, li passarà al FS la **normal N en eye space**.

El FS accedirà a la textura **window.png** (amb les coordenades de textura habituals) per obtenir un color que li direm **C**.

Si la component alfa de **C** és 1.0 (part opaca de la primera finestra), el color del fragment serà **C**.

Si la component alfa de **C** és inferior a 1.0, per calcular el color del fragment s'accedirà a la textura **interior.png** (amb coordenades de textura **vtexCoord+0.5*N.xy**) per obtenir un color que li direm **D**.

Si la component alfa de **D** és 1.0 (part opaca de la finestra interior), el color del fragment serà **D**. Altrament, el color del fragment serà el color de la textura **exterior.png** al punt de coordenades **vtexCoord+0.7*N.xy**.

Observeu que estem usant un offset en les coordenades de textura que depèn de les components de la normal en eye space. Degut a aquest offset, la part visible de l'interior i de l'exterior dependrà de l'orientació del model. Aquí teniu el resultat esperat (**plane.obj**), des de diferents punts de vista:



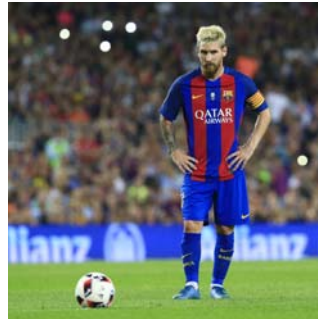
Identificadors (ús obligatori):

```
magic.vert, magic.frag
uniform sampler2D window;
uniform sampler2D interior1; // observeu el digit 1 al final
uniform sampler2D exterior2; // observeu el digit 2 al final
```

Messi (messi.*)

2.5 punts

Escriu un VS+FS per aplicar una textura (amb una foto de Messi) a l'objecte **plane.obj**:



La pilota està centrada al punt amb coordenades de textura $C=(0.272, 0.09)$ i té un radi (també en espai normalitzat de textura) $r=0.065$.

El VS farà les tasques habituals que resultin imprescindibles per a aquest exercici.

El FS serà l'encarregat d'aplicar la textura, amb la peculiaritat de que el baló girarà respecte el seu centre C (sense desplaçar-se). El color del fragment es calcularà per tant de manera diferent, segons si les coordenades de textura (s,t) del fragment (vtxCoord) estan o no dins el cercle centrat a C i radi r.

Quan la distància de (s,t) al centre C sigui inferior al radi del baló, haureu de rotar les coordenades (s,t) respecte C. Per tant, haureu de (1) traslladar (s,t) de forma adient; (2) aplicar una rotació 2D, i (3) desfer la translació que heu aplicat a (1). Per la rotació 2D podeu fer servir la matriu de rotació 2D,

$$\begin{bmatrix} \cos(\text{time}) & \sin(\text{time}) \\ -\sin(\text{time}) & \cos(\text{time}) \end{bmatrix}$$

Quan la distància de (s,t) al centre C sigui superior o igual al radi, usareu les (s,t) originals.

El color final del fragment serà el color de la textura al punt (s,t).

Aquí teniu alguns exemples (observeu la rotació del baló, en sentit anti-horari). Vegeu també el vídeo.



Identificadors (ús obligatori):

```
messi.vert, messi.frag  
uniform sampler2D colorMap;  
uniform float time;
```