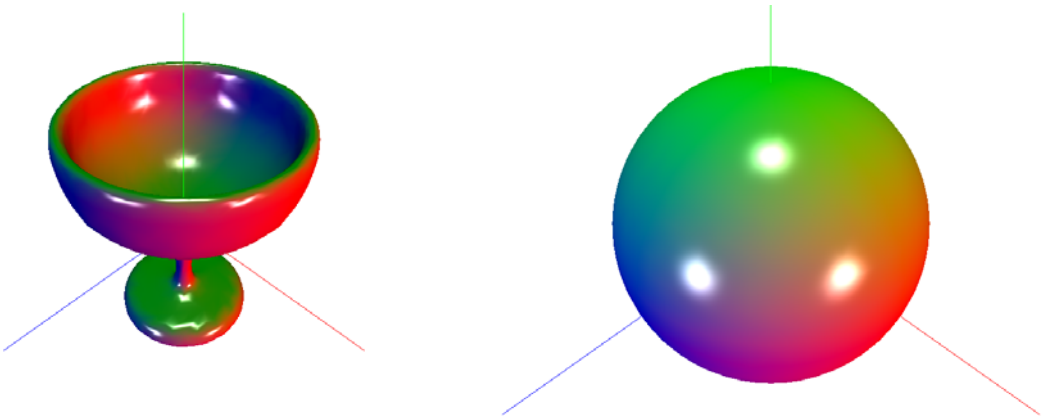
3lights (3lights.*)

Escriu **VS+FS** per aplicar il·luminació de Phong **per fragment**, amb 3 llums direccionals fixes:



El VS farà les tasques habituals i passarà al FS les dades necessàries pel càlcul d'il·luminació.

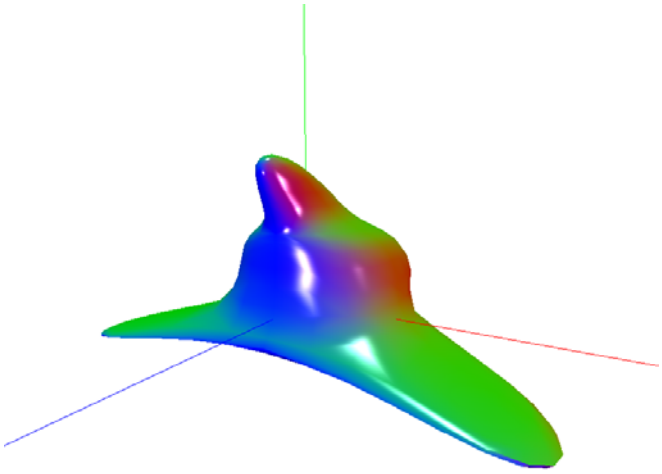
El FS calcularà el color del fragment acumulant la contribució de tres llums direccionals fixes (per tant, ignoreu *lightPosition*):

- Llum infinitament allunyada en la direcció X+ (model space), amb *lightDiffuse* **vermell**.
- Llum infinitament allunyada en la direcció Y+ (model space), amb *lightDiffuse* **verd**.
- Llum infinitament allunyada en la direcció Z+ (model space), amb *lightDiffuse* **blau**.

Per calcular la contribució de cada llum, ignoreu *matAmbient* i *lightAmbient,* i useu l'expresió:

$$K_d\, I_d\, (N \cdot L) + K_s\, I_s\, (R \cdot V)^S$$

on I_d (*lightDiffuse*) és diferent per cada llum. Per la resta de paràmetres (K_d, K_s, I_s, s) feu servir els uniforms habituals.

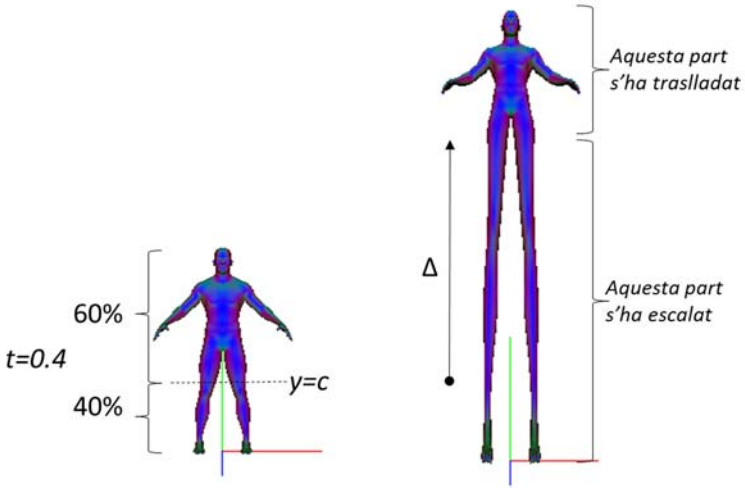


Dalify (dalify.*)

Escriu **VS+FS** per deformar el model en direcció vertical (eix Y en *model space*), per obtenir una aparença similar a la d'alguns animals en quadres de Salvador Dalí:

Les temptacions de Sant Antoni (Salvador Dalí, 1946)

El VS deformarà el model modificant únicament la coordenada Y en *model space*:

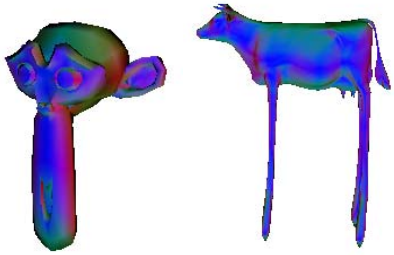


Sigui c el resultat d'interpolar linealment boundingBoxMin.y i boundingBoxMax.y, segons un paràmetre d'interpolació t, **uniform float t = 0.4**.

Si la coordenada Y és inferior a c, el VS li aplicarà l'escalat donat per **uniform float scale = 4.0** per tal d'allargar les potes del model. Altrament, no li aplicarà cap escalat, però sí una translació Δ en Y. Per calcular Δ, observeu que per tenir continuïtat a y=c, llavors **c*scale = c + Δ** (aïlleu Δ).

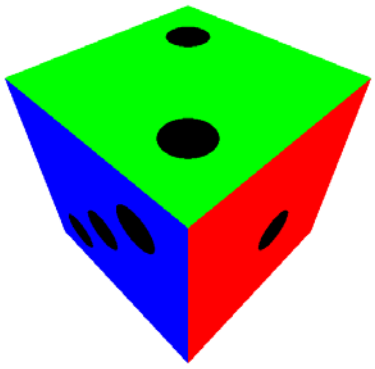
Degut a que no estem recalculant els plans de *clipping*, és possible que el model surti retallat.

El FS farà les tasques habituals.



Dice (dice.*)

Escriu **VS+FS** per *texturar* procedimentalment l'objecte **cube.obj** de forma que aparegui com un dau amb cares d'un, dos o tres punts:



El VS farà les tasques habituals, passant-li al FS les dades que requereix, amb el color sense il·luminació.

El FS no farà servir cap textura, sinó que calcularà el color final del fragment en funció de les coordenades XYZ i la normal, que li enviarà el VS en *object space*.

El número de punts de cada cara del cub dependrà de la normal de la cara en *object space*:

- Les cares amb normal paral·lela a l'eix **X**, mostraran **un** punt.
- Les cares amb normal paral·lela a l'eix **Y**, mostraran **dos** punts.
- Les cares amb normal paral·lela a l'eix **Z**, mostraran **tres** punts.

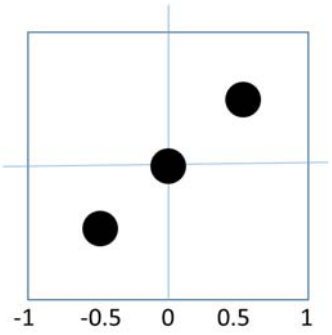
Useu un test robust per determinar quina cara esteu tractant (eviteu comparacions d'igualtat amb float).

No farem servir les coordenades de textura del cub, sinó directament les coordenades XYZ que el VS passarà al FS en *object space*. En el cas del cub, les coordenades estan en [-1, 1].

Exemple per a una cara amb normal paral·lela a l'eix X:

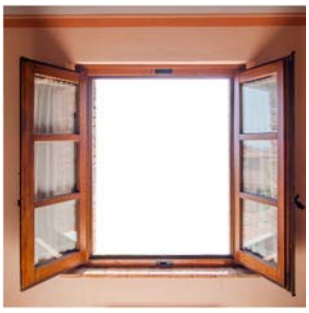
- Sigui Q el punt 2D definit per les coordenades YZ del punt (ignorem la coordenada X).
- Si la distància entre Q i el punt central de la cara (0, 0) és inferior a 0.2, el color del fragment serà negre (el fragment correspon al punt negre); altrament el color serà el color original sense il·luminació.

El tractament per les altres cares és similar, prenent XZ ó XY per definir el punt Q, i calculant el color del fragment com a negre si la distància de Q a qualsevol dels centres (0,0), (0.5, 0.5), (-0.5, -0.5) és inferior al radi 0.2:



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
```