Laboratori:
Il·luminació (I) – Sessió 3.1

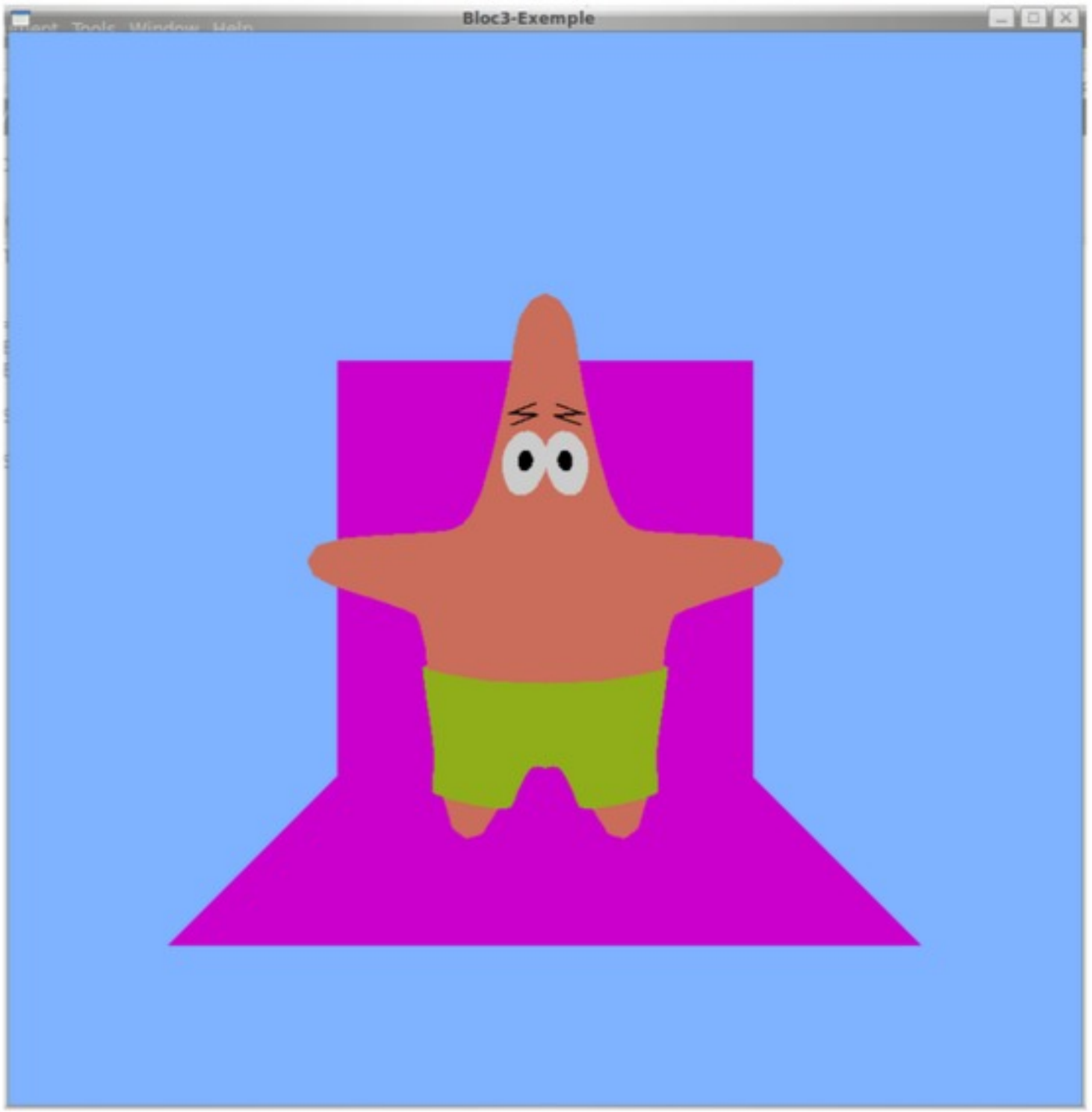
Professors de IDI

Index

- Introducció
- Càlcul del color en un punt amb models empírics
- Nou esquelet
- Feina per avui
- Consideracions importants

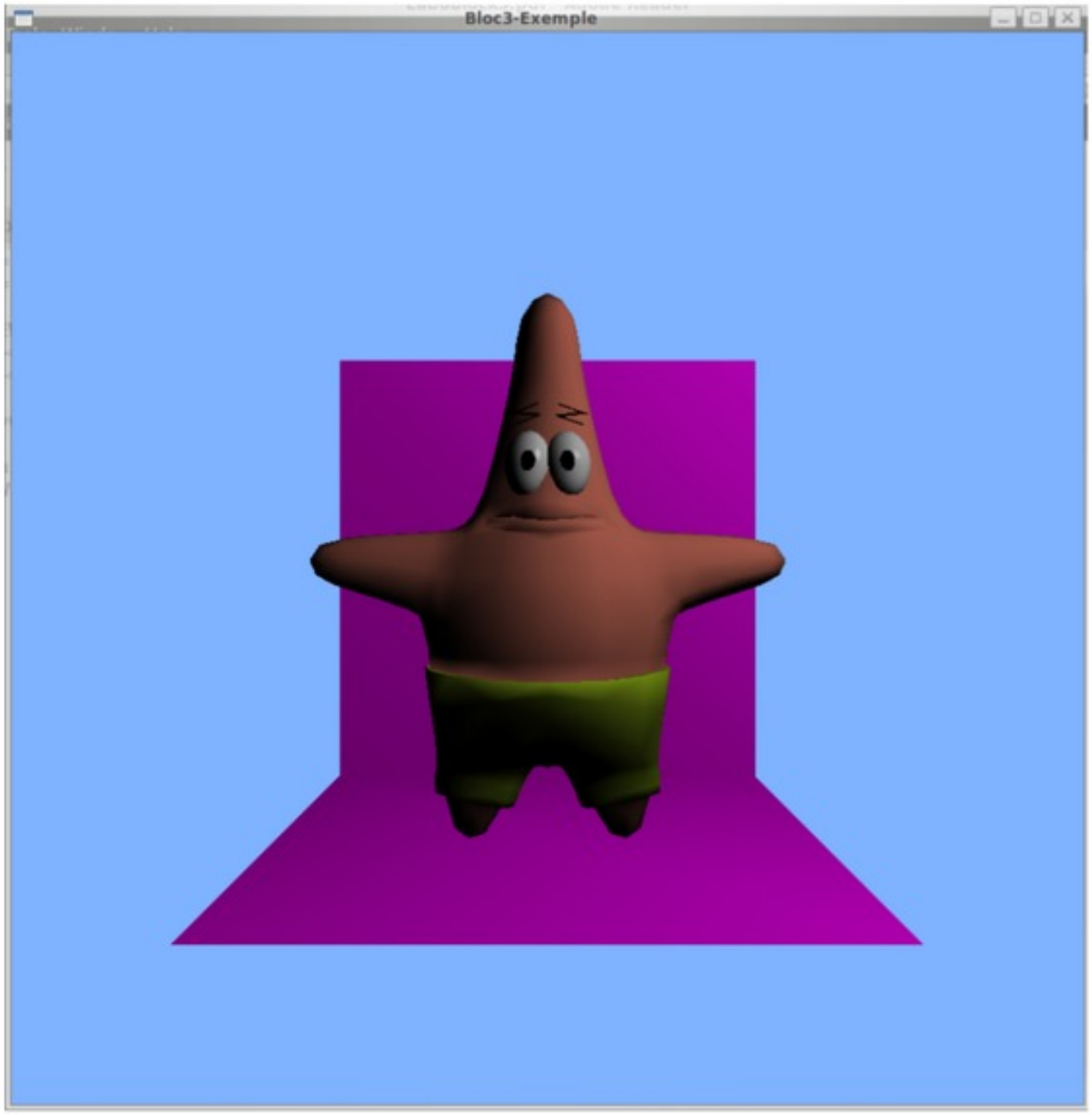
Introducció

Afegir més realisme a l'escena
Evitar colors plans
Donar volum
Il·luminar

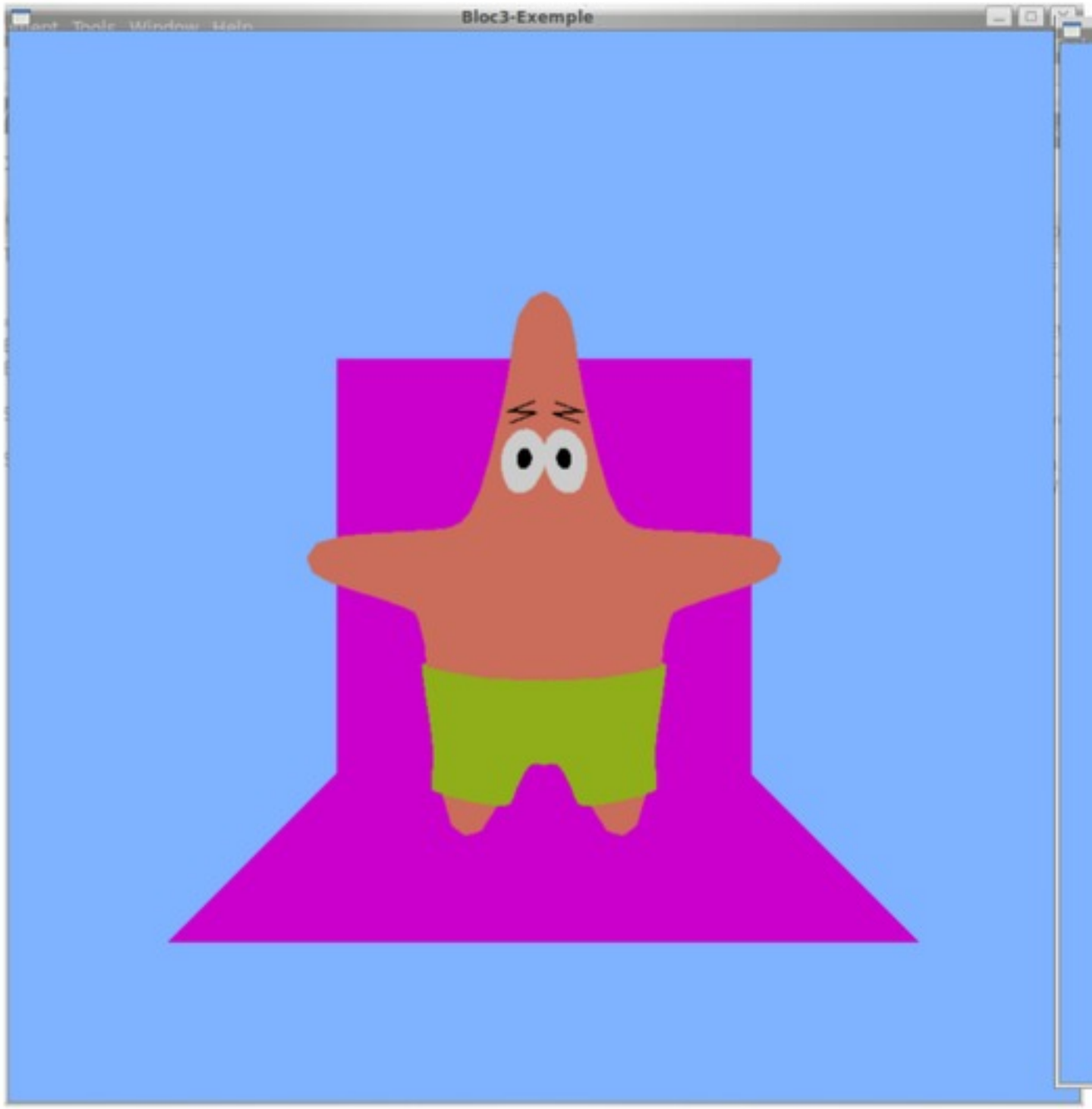


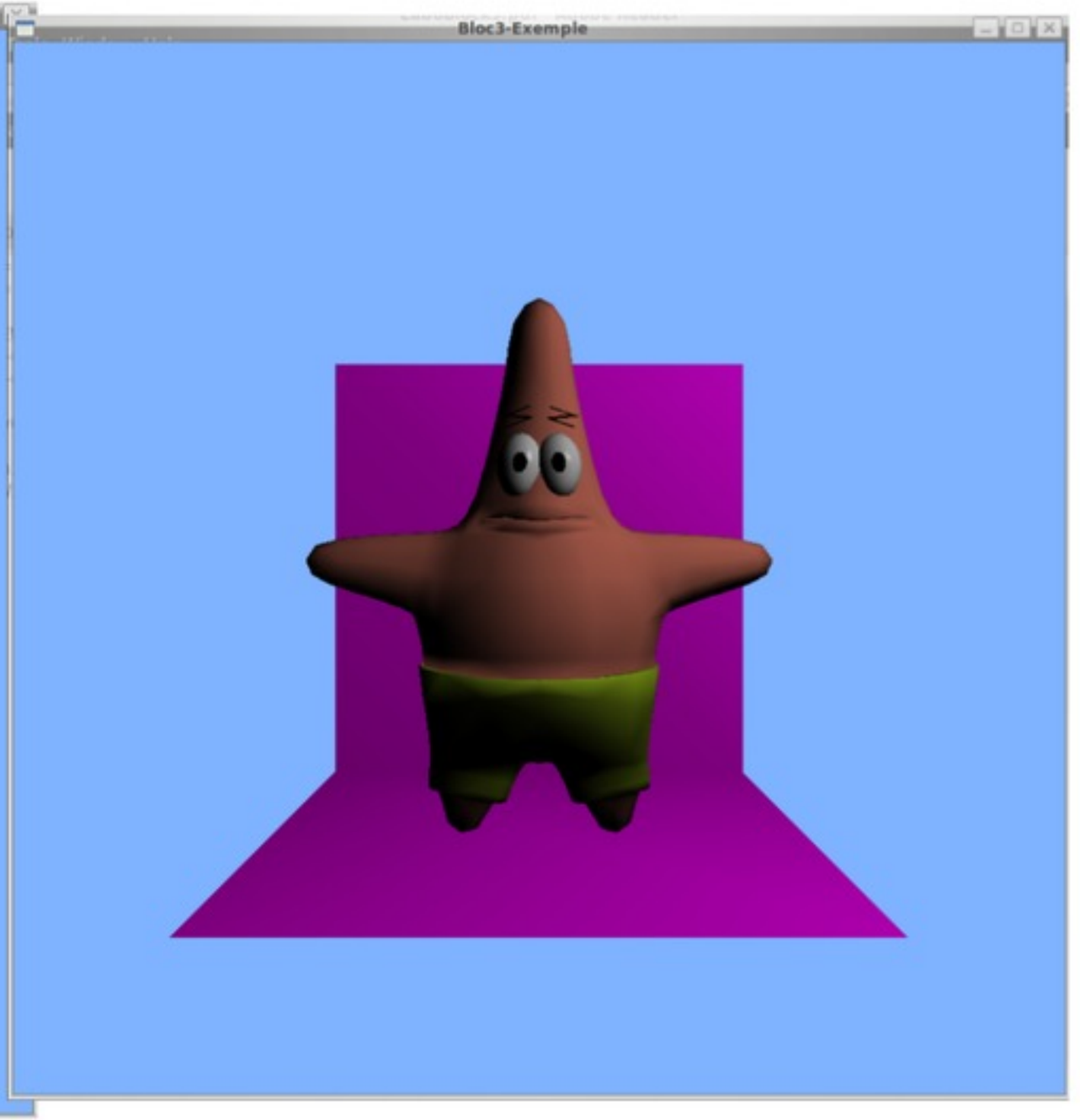
Introducció

Afegir més realisme a l'escena
Evitar colors plans
Donar volum
Il·luminar



Introducció





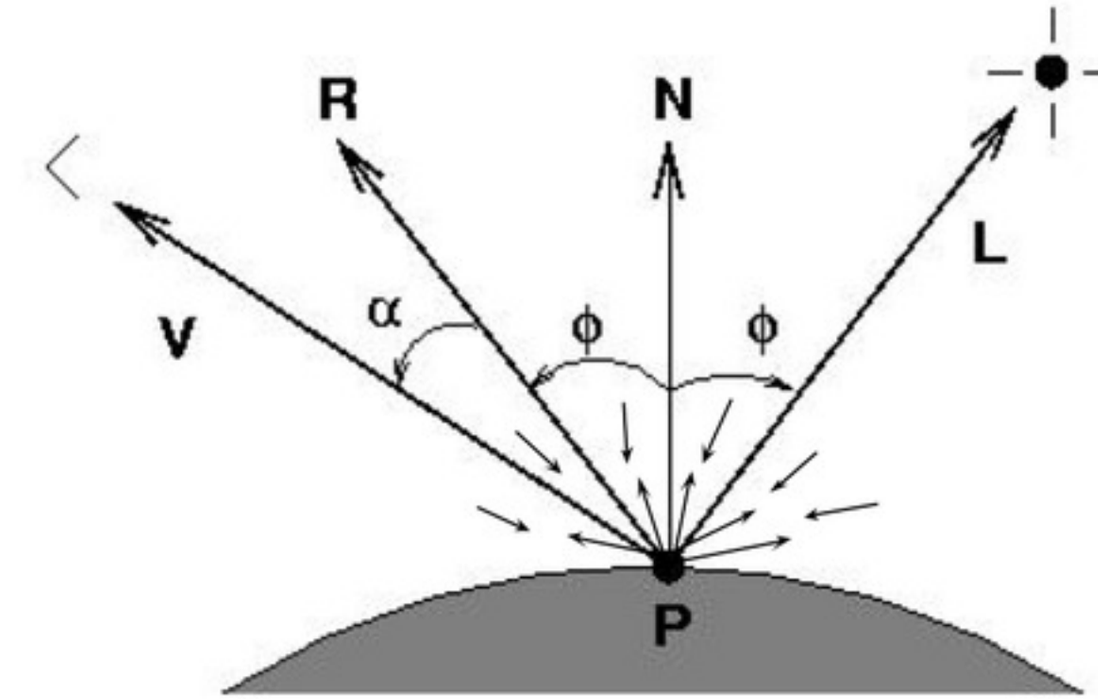
Càlcul amb models empírics

$$I_\lambda(P) = I_{a\lambda}k_{a\lambda} + \Sigma_i (I_{fi\lambda}\, k_{d\lambda}\, cos(\Phi_i)) + \Sigma_i (I_{fi\lambda}\, k_{s\lambda}\, cos^n(\alpha_i))$$

$cos(\Phi) \Rightarrow dot\ (L,N)$

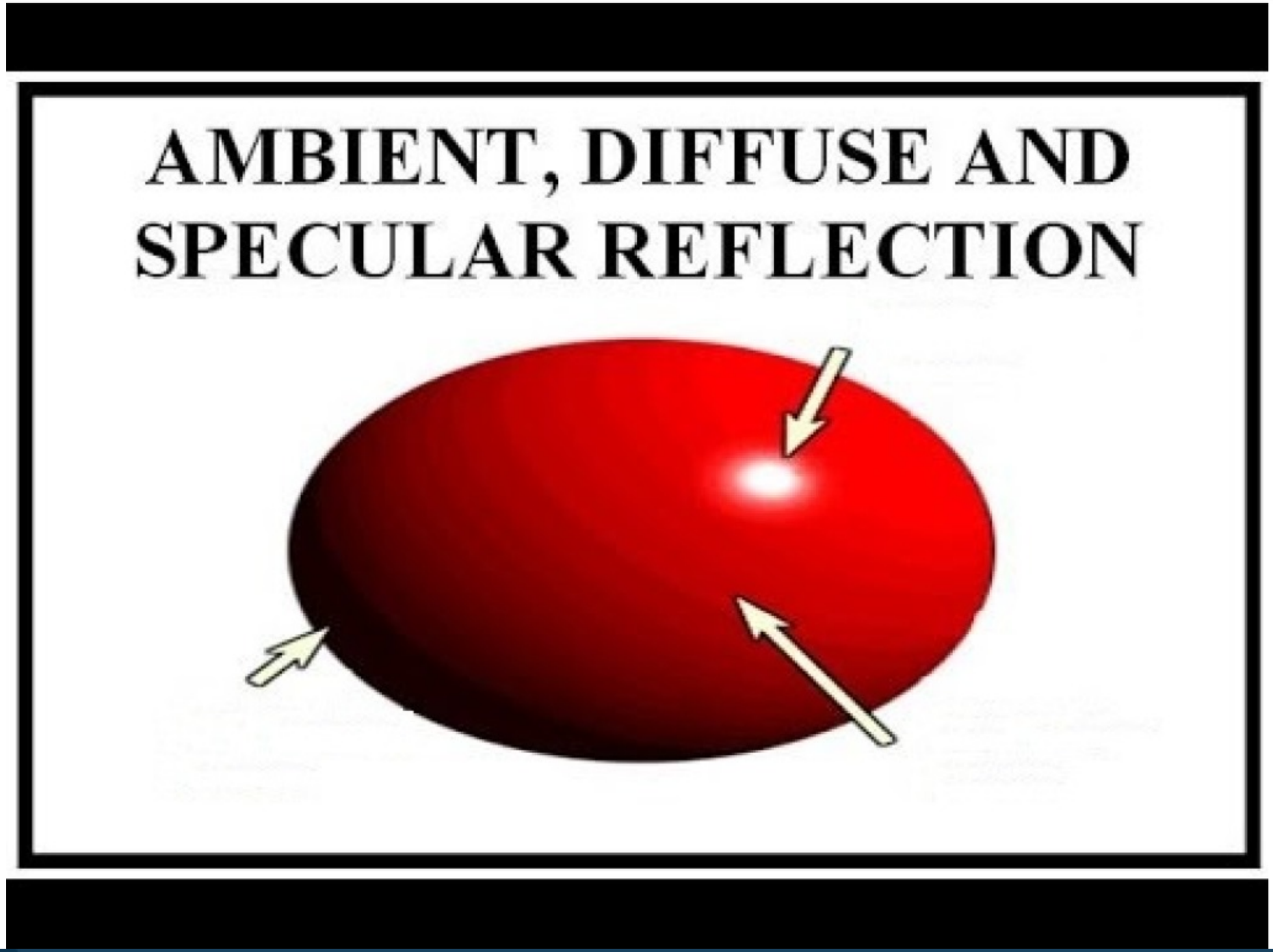
$cos(\alpha) \Rightarrow dot\ (R, v)$

L, N, R i v normalitzats

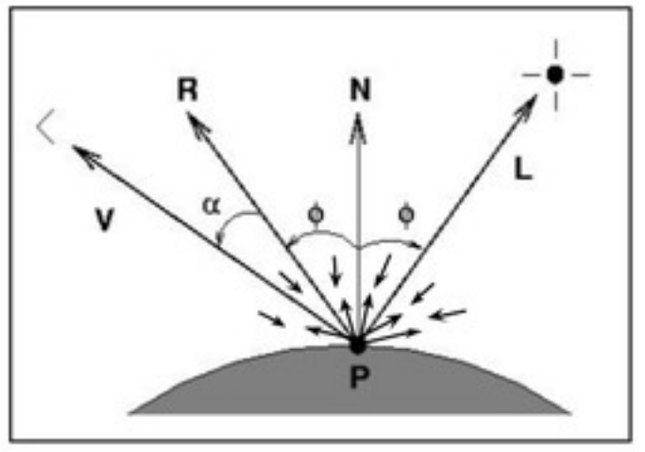


Draw It

On es troben les tres components de la llum ?



Càlcul amb models empírics



Què necessitem?

- Propietats del material
- Vector normal

} Per cada vertex (punt)

- Color de llum ambient
- Posició del focus de llum
- Color del focus de llum

} Per cada focus de llum

- Posició observador – en SCO sabem que és (0,0,0)

Farem tots els càlculs en SCO perquè resulten més simples

Càlcul amb models empírics

Càlcul a cada vertex al Vertex Shader EN SCO. Per a fer-ho cal

ORIGINAL	COM CONVERTIR A SCO
Vèrtex (v)	View * TG * v
Normal (n)	NormalMatrix * n
Focus de llum (posFocus)	View * posFocus

NormalMatrix = inversa(transposada(view*TG))

El focus de llum només es passa a SCO si no està ja en SCO (ens ho diran a l'enunciat)

El càlcul de la il·luminació la farem al VS.

Passar normal a SCO

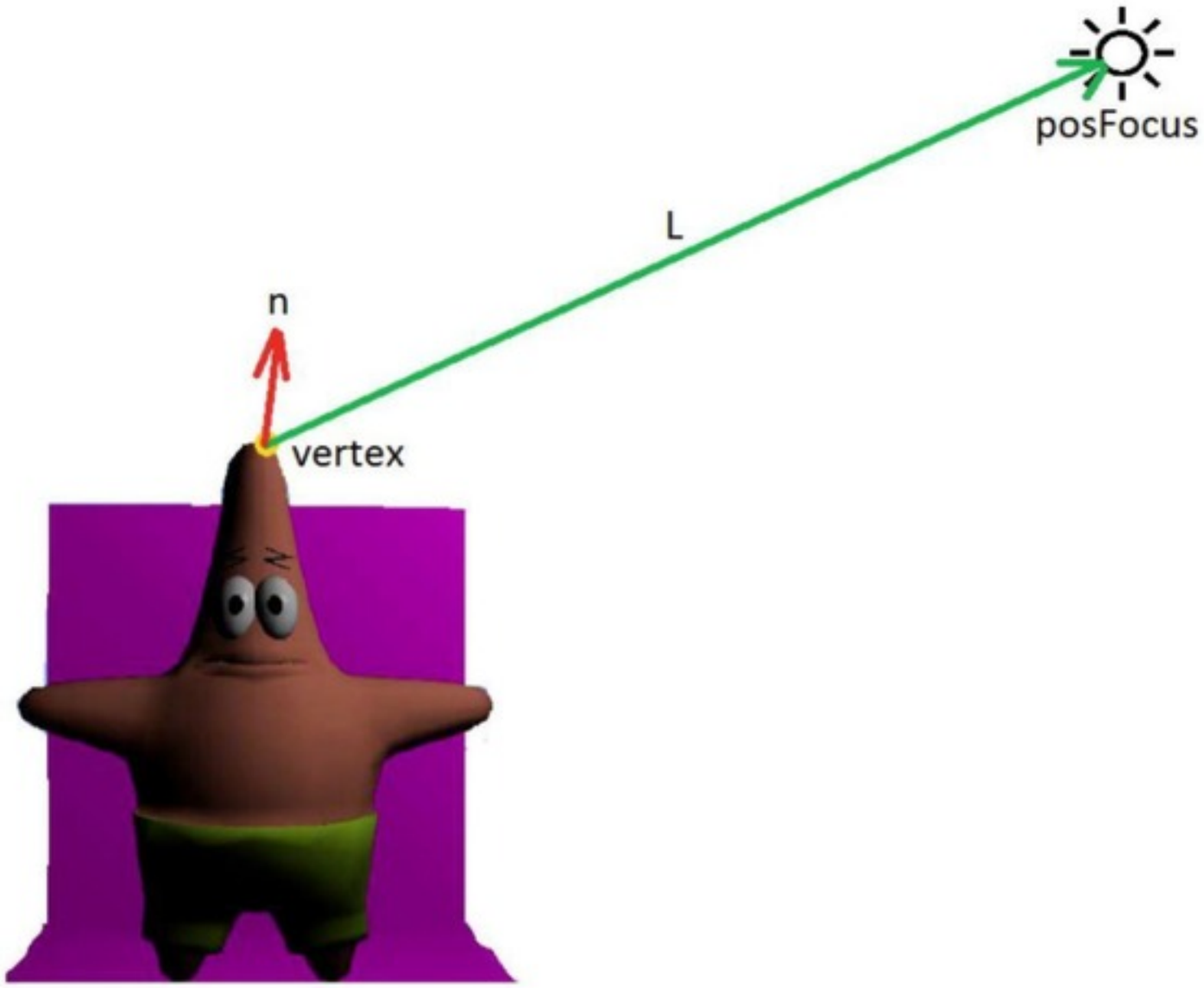
atrix * no al

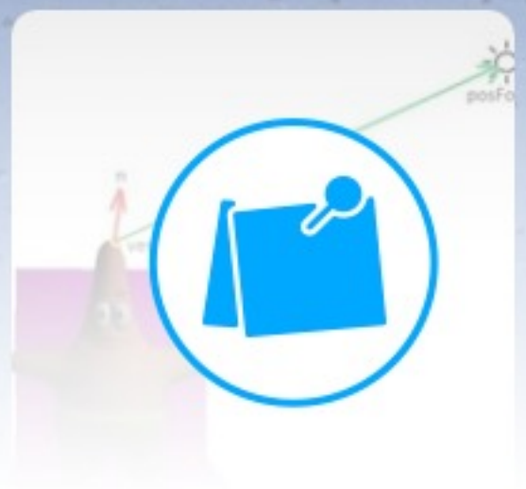
Passar focus de llum a SCO

Matching Pairs

Càlcul amb models empírics

També necessitarem que la direcció de la llum (L) estigui en coordenades d'observador





Collaborate!

Com podem calcular L en SCO ?

Càlcul amb models empírics

Calcular matriu inversa de la trasposada de view * TG

- Al vertex shader (en GLSL):

```
mat3 NormalMatrix = inverse (transpose (mat3 (view * TG)));
```

es fa el càlcul de la matriu per a cada vèrtex

- Al MyGLWidget (amb glm):

```
#include “glm/gtc/matrix_inverse.hpp”
glm::mat3 NormalMatrix = glm::inverseTranspose(glm::mat3(View*TG));
```

cal tenir les matrius View i TG com a atributs de la classe

i cal passar la NormalMatrix com a uniform al VS per cada objecte

Quiz

Com es calcula la direcció de la llum L ? És el vector que uneix el vèrtex amb el focus de llum

- L=posfocus-vertex (els dos en SCO)
- L=vertex-posfocus (els dos en SCO)
- L=n

Per a què vertex estigui en SCO cal

- multiplicar view * TG * vertex
- multiplicar view * vertex
- multiplicar vertex * TG * view

Per a què posfocus estigui en SCO cal

- multiplicar view * TG * posfocus
- multiplicar view * posfocus
- multiplicar posfocus * view

Quina d'aquestes afirmacions sobre posfocus és certa ?

- posfocus no s'ha de multiplicar per TG per passar-lo a SCO perquè no forma part de cap model
- posfocus no s'ha de passar a SCO
- **per poder calcular la L no cal passar posfocus a SCO**
- **cal multiplicar posfocus per TG i per View**

Per quina matriu s'ha de multiplicar la normal ?

- transposada(inversa(view * TG)) * normal
- inversa(transposada(view * TG) * normal
- inversa(transposada(TG * view)) * normal
- normal * inversa(transposada(view * TG)

Nou esquelet

- Analitzar quins mètodes implementats.
- Analitzar implementació dels mètodes.
  - Quina càmera tenim?
  - Quina escena?
  - Quina interacció?
- Analitzar els shaders
  - Atributs, uniforms, funcions

Feina per avui

Càlcul color usant model Lambert:

```
vec3 Lambert (vec3 NormSCO, vec3 L)
{
   // Aquesta funció calcula la il·luminació amb Lambert assumint que els vectors
   // que rep com a paràmetres estan normalitzats

   vec3 colRes = llumAmbient * matamb; // Inicialitzem color a component ambient
   // Afegim component difusa, si n'hi ha
   if (dot (L, NormSCO) > 0)
      colRes = colRes + colFocus * matdiff * dot (L, NormSCO);
   return (colRes);
}
```

Cal calcular en *main*:

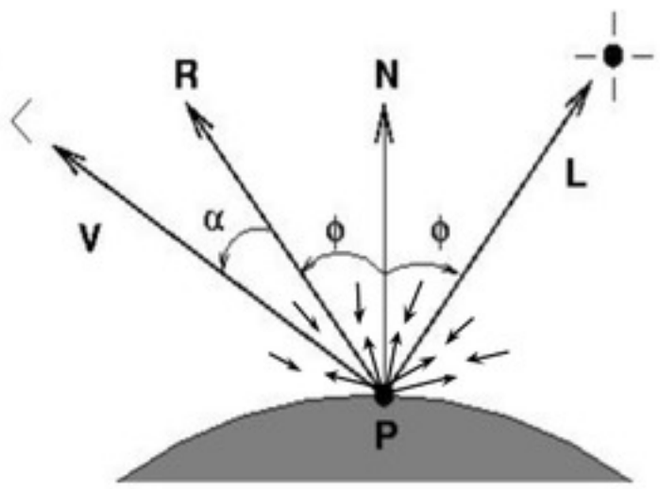
- L en SCO
- Normal en SCO
- normalitzar vectors
- cridar a Lambert

Feina per avui

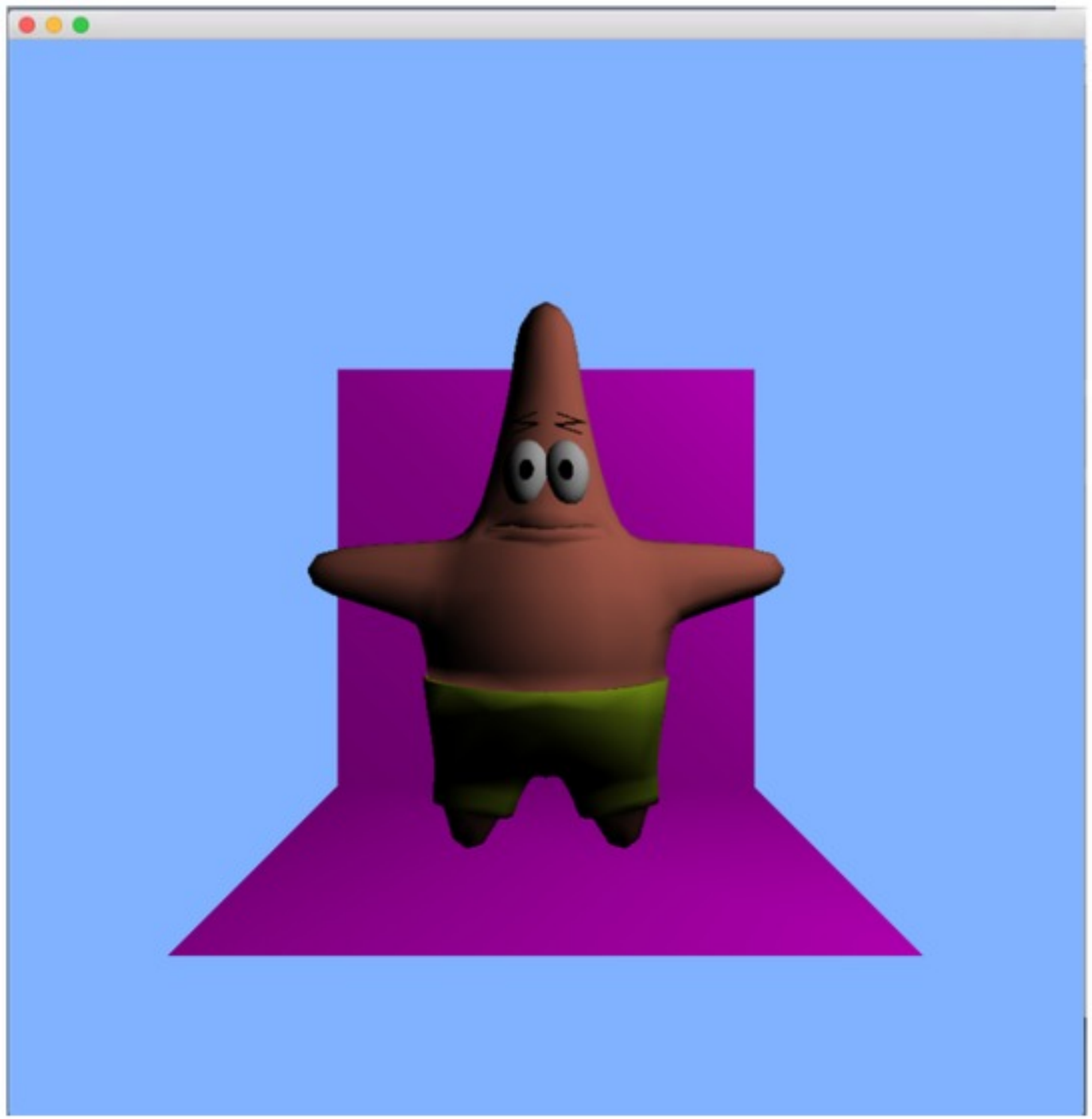
Pseudocodi de Vertex Shader per calcular color en SCO:

$$c_v = I_{a\lambda} k_{a\lambda} + \Sigma_i (I_{fi\lambda} k_{d\lambda} dot(N_o, L_o)) + \Sigma_i (I_{fi\lambda} k_{s\lambda} dot(R_o, v_o)^n)$$

```
in vec3 vertex, N;
in vec3 matamb, matdiff, matspec;
in float matshin;
uniform mat4 proj, view, TG;
uniform vec3 posFocus, Ia, If;
out vec3 fcolor;
...
void main()
{
   mat3 NormalMatrix= inverse (transpose (mat3 (view*TG)))
   vec3 NormSCO= normalize(NormalMatrix*N);
   vec4 vertexSCO= view*TG* vec4(vertex,1.0);
   vec4 focusSCO = view* vec4 (posFocus, 1.0);
   vec3  LSCO=normalize (focusSCO.xyz-vertexSCO.xyz);
   fcolor= color_vertex(NormSCO, LSCO, vertexSCO);
   gl_Position = proj * view * TG * vec4 (vertex, 1.0);
}
```

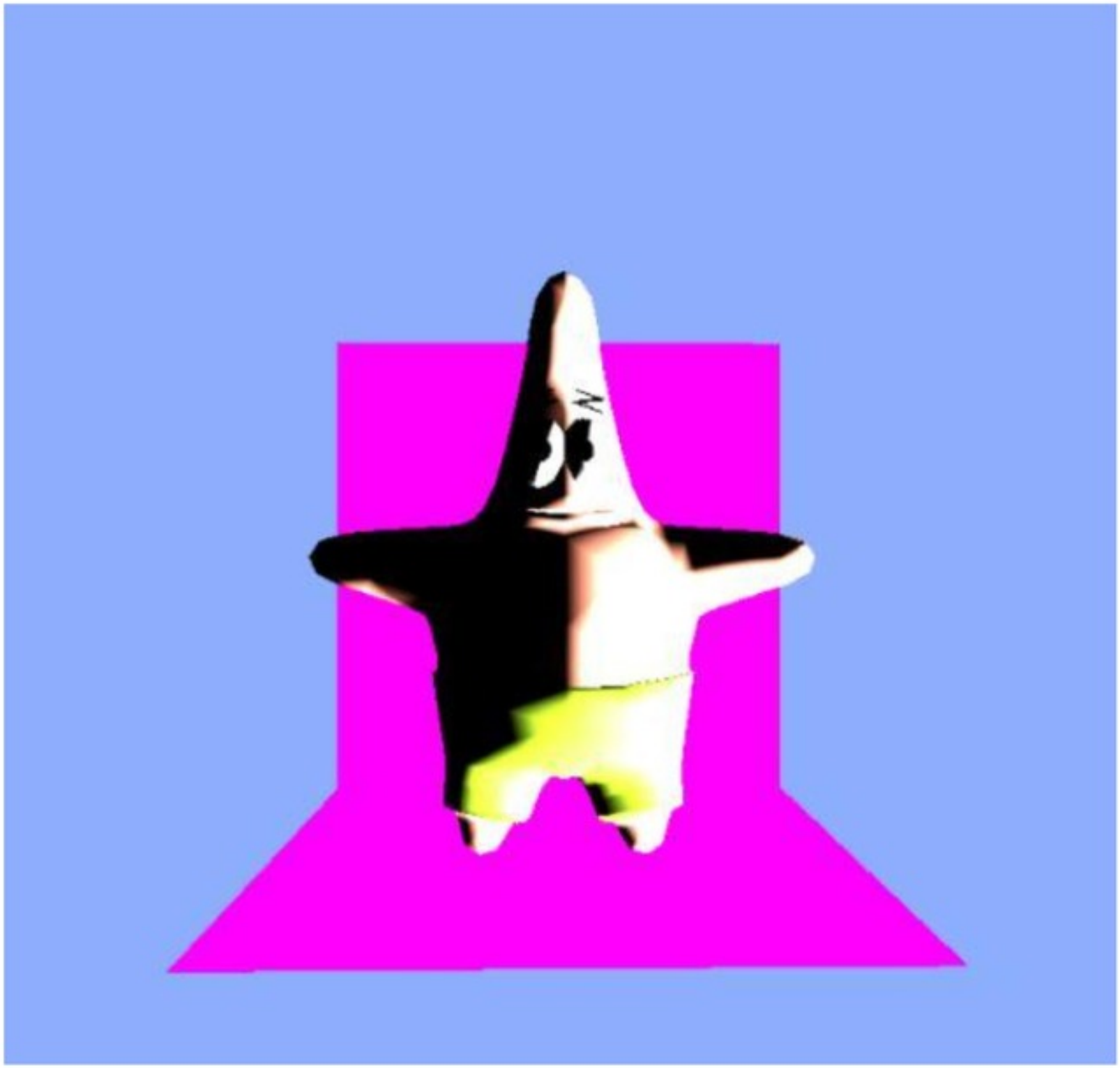


Feina per avui



Proveu a moure càmera (si voleu poseu tb rotació en X) i veureu que cares il·luminades no varien. **Llum d'escena**.

Feina per avui

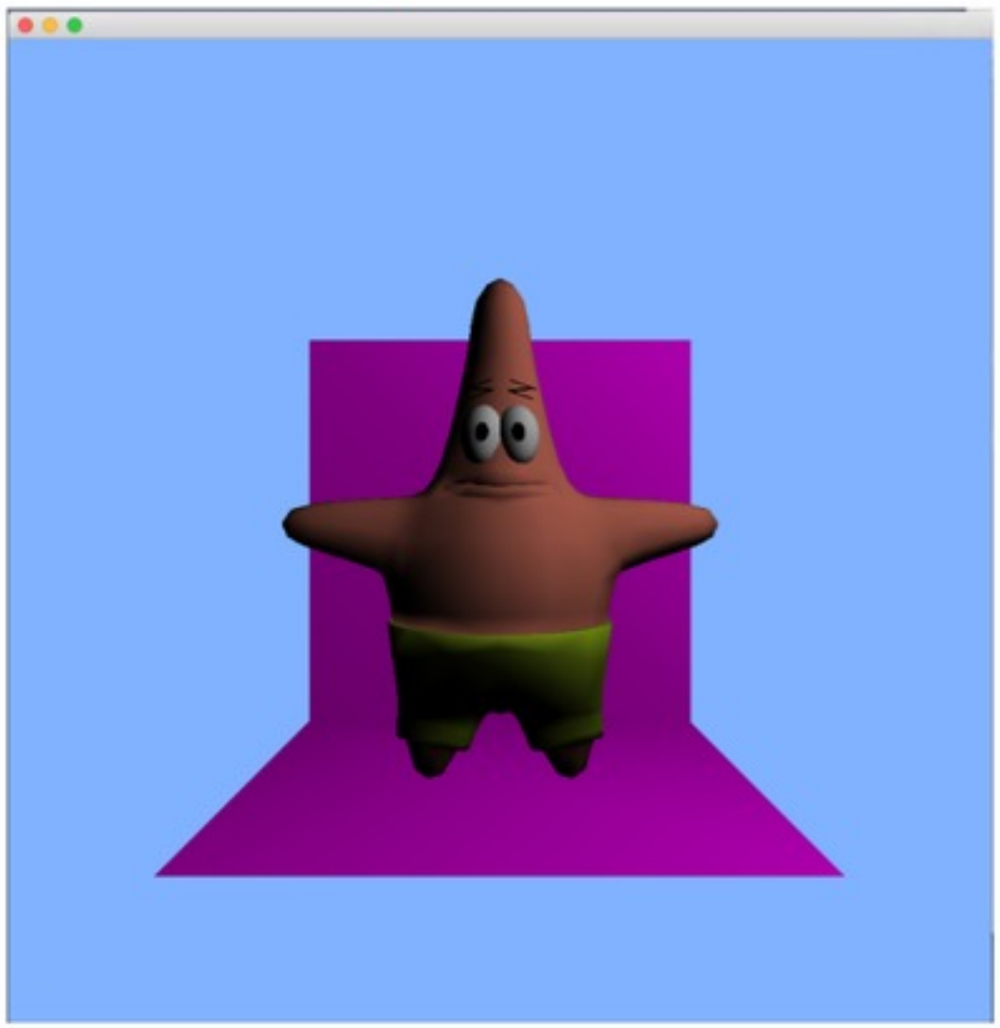


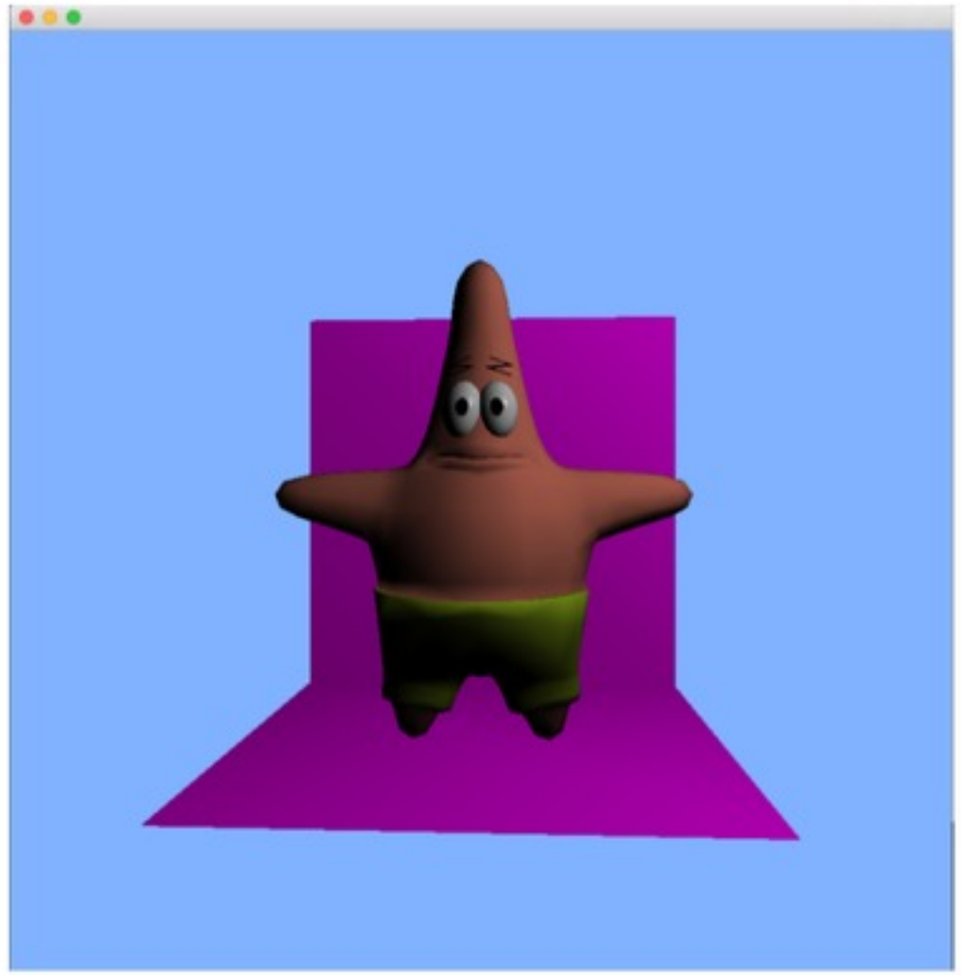
Feina per avui

Càlcul color usant model Phong:

```
vec3 Phong (vec3 NormSCO, vec3 L, vec4 vertSCO)
{
    // Els vectors rebuts com a paràmetres (NormSCO i L) estan normalitzats
    vec3 colRes = Lambert (NormSCO, L); // Inicialitzem color a Lambert
    // Calculem R i V
    if (dot (NormSCO, L) < 0)
        return colRes;  // no afecta la component especular
    vec3 R = reflect (-L, NormSCO); // equival a:: 2.0 * dot (NormSCO, L) * NormSCO - L;
    vec3 V = normalize (-vertSCO.xyz);
    if ((dot (R, V) < 0) || (matshin == 0))
        return colRes; // no afecta la component especular
    // Afegim la component especular
    float shine = pow (dot (R, V), matshin);
    return colRes + matspec * colFocus * shine;
}
```

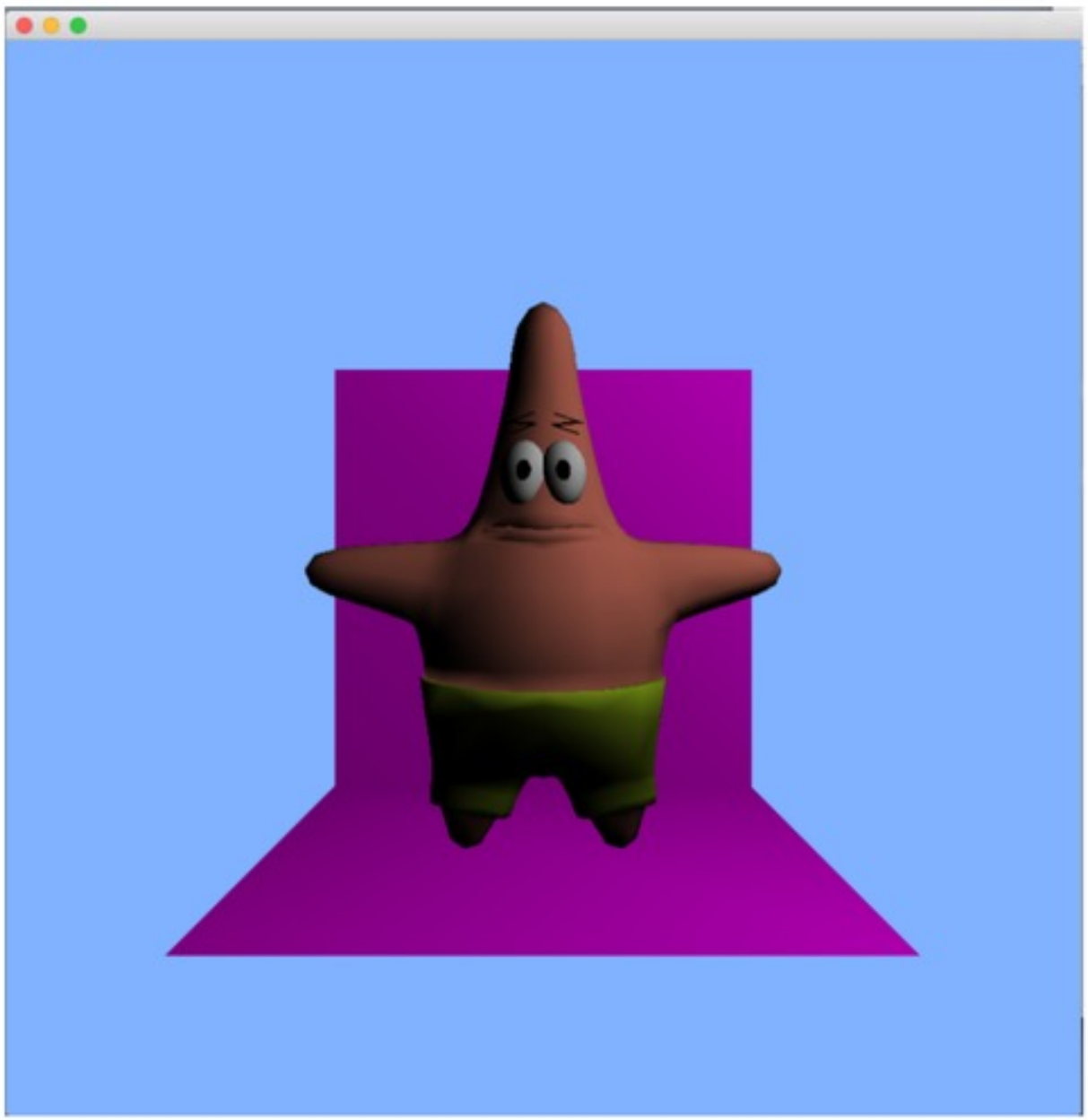
Feina per avui

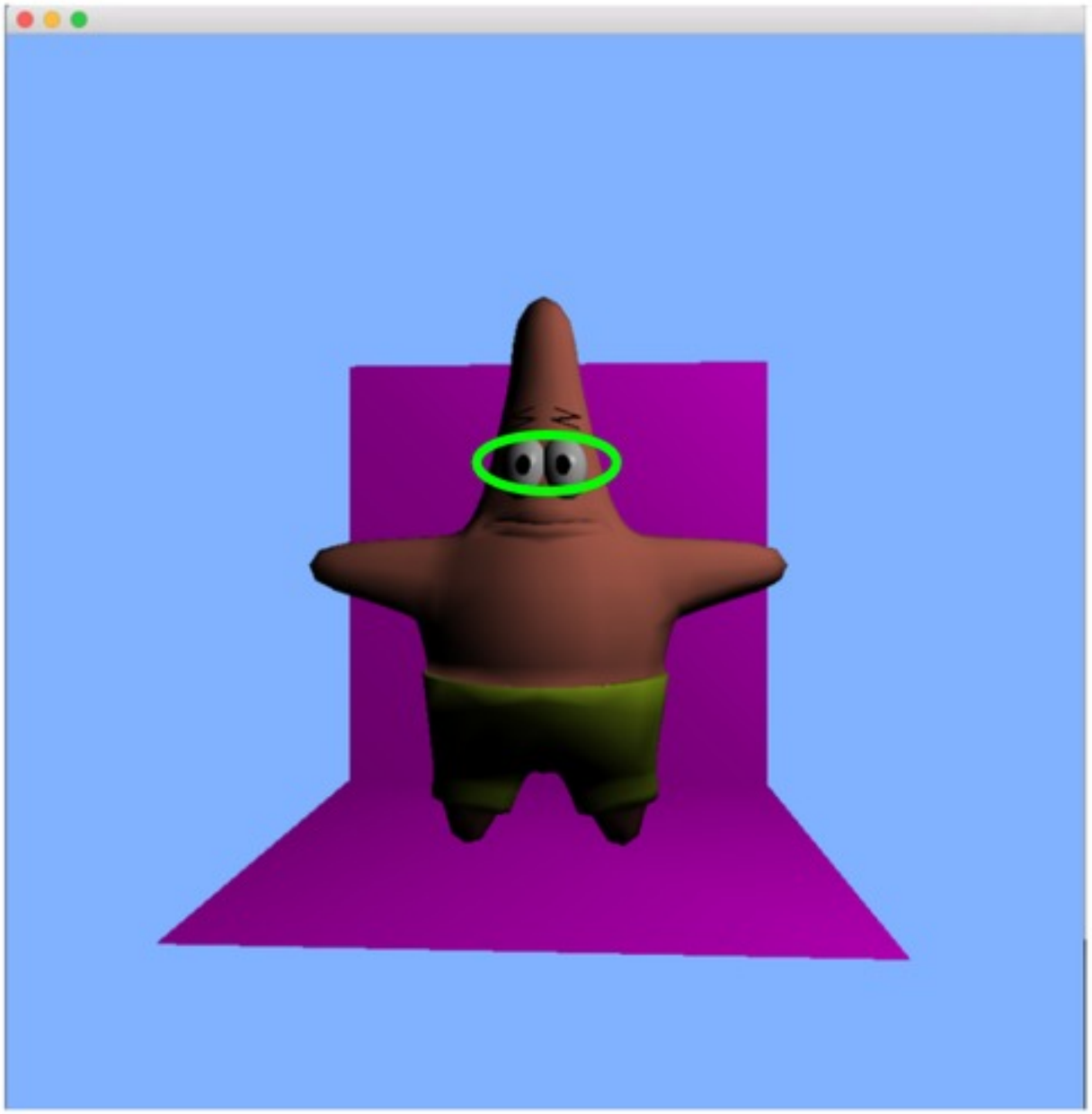




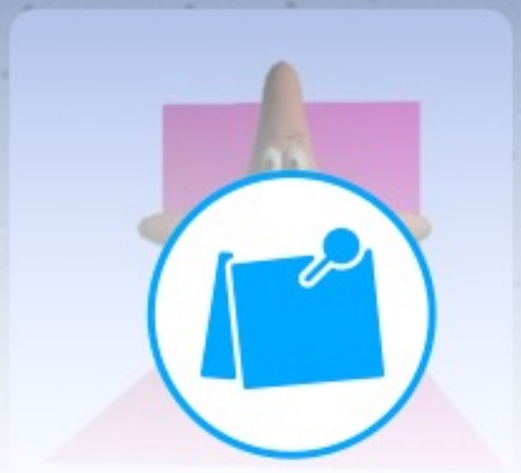
Proveu a moure càmera (si voleu poseu tb rotació en X) i veureu que cares il·luminades no varien; però taca especular sí (en ulls).
Llum d'escena.

Feina per avui





Als ulls podreu observar reflexions especulars. Canvien si es modifica la vista.



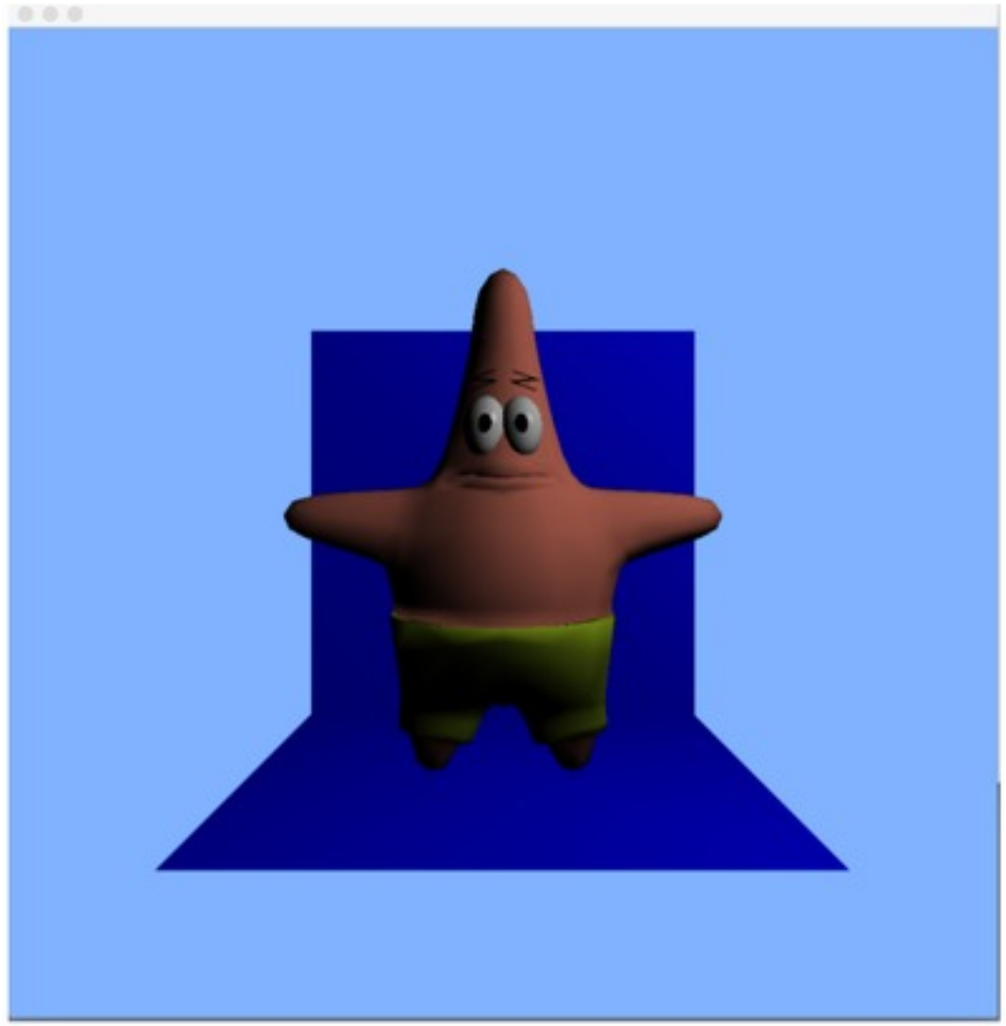
Collaborate!

Per què el terra no té reflexions especulars en la imatge ?

Feina per avui

3) Canvi material terra+paret

- Ha de ser de plàstic blau



4) Canvi posició focus de llum

- Ha de ser la posició (1, 0, 1) en SCA

Feina per avui

Pasar a uniforms de la posició i el color del focus de llum:

- Convertir la posició i el color en uniforms en el VS
- Inicialitzar aquests uniforms al MyGLWidget
- Fixem-nos que ara podríem passar el uniform de posició directament ja en SCO

Podem també passar a uniform el color de la llum ambient

Feina per avui

Fer que la posició del focus de llum es mogui amb les tecles K i L:

K: mou el focus cap a les X-

L: mou el focus cap a les X+

Consideracions importants

- No es pot multiplicar vec3 * mat4
- Els vèrtexs s'han de passar a coordenades homogènies
- Els vectors s'han de normalitzar: L, n
- No tot el que és vec3 és un vector.
- La shininess va de 0 a 128
- Les crides a Lambert i Phong retornen un valor
- Les components r,g,b d'un color van entre 0 i 1
- L=pos_Focus - vertex ni normalitzar