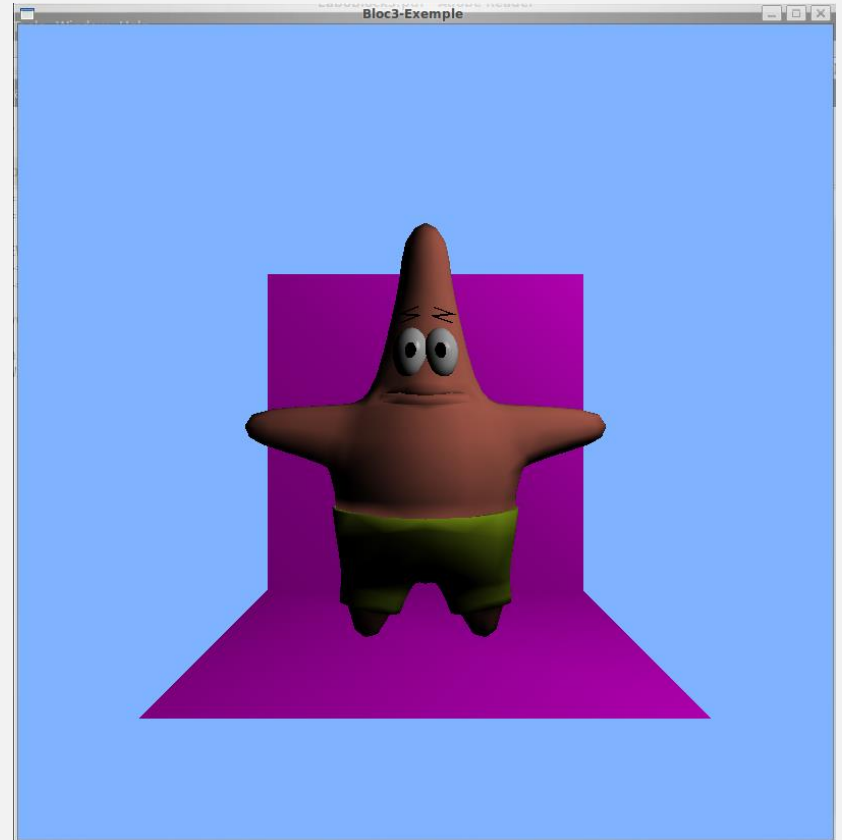
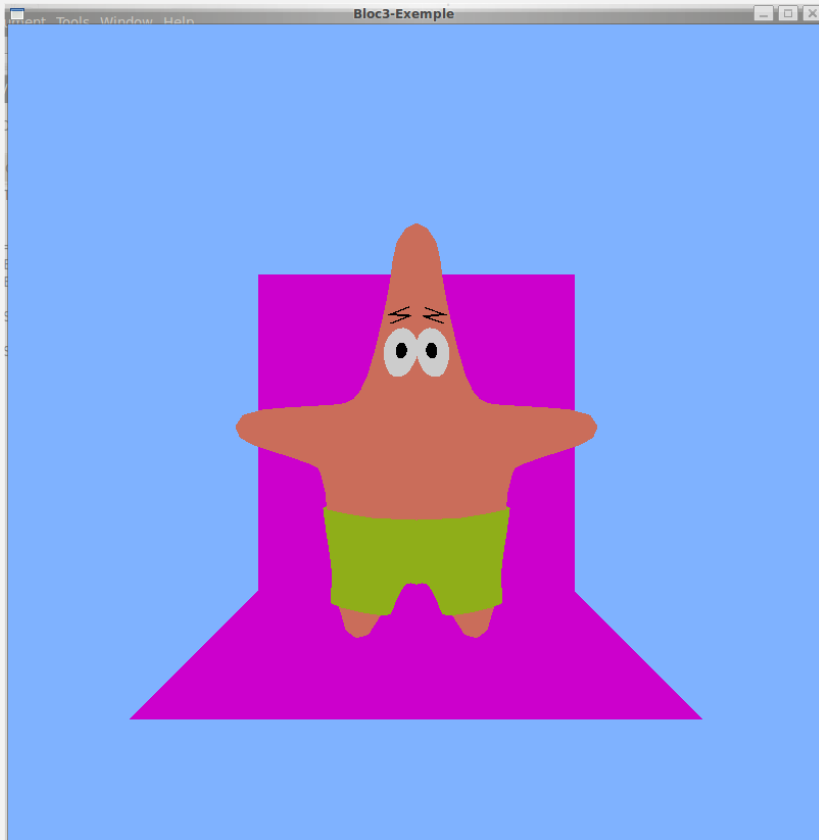


# Laboratori OpenGL – Sessió 3.1

## Bloc 3

### Realisme - Il·luminació:



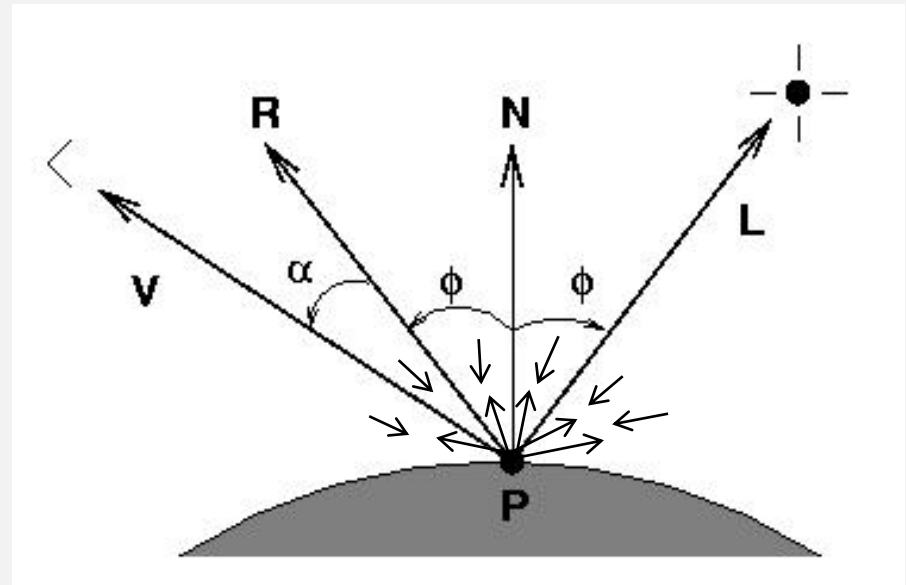
# Càlcul color en un punt: models empírics

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

$\cos(\Phi) \Rightarrow \text{dot}(L, N)$

$\cos(\alpha) \Rightarrow \text{dot}(R, v)$

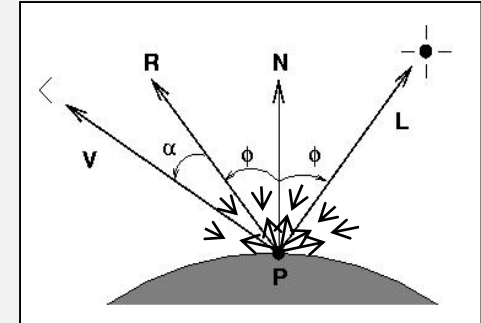
$L, N, R$  i  $v$  normalitzats



# Càlcul color en un punt: models empírics

## Què necessitem?

- Propietats del material
  - Vector normal
  - Color de llum ambient
  - Posició del focus de llum
  - Color del focus de llum
  - Posició observador – en SCO sabem que és (0,0,0) -
- Per cada vertex (punt)
- Per cada focus de llum



# Càlcul color en un punt: models empírics

Primer farem el càlcul per cada vèrtex (al Vertex Shader)

I **el farem en SCO**, per tant:

- Cal passar la posició del vèrtex a SCO
  - multiplicat per (**view \* TG**)
- Cal passar el vector normal a SCO
  - multiplicat per la matriu **inversa de la transposada de (view \* TG)**  
- li direm **NormalMatrix** -
- La posició del focus de llum també ha d'estar en SCO
  - Multiplicat per **view** (si no la tenim ja directament en SCO)

# Càlcul color en un punt: models empírics

Calcular matriu inversa de la trasposada de  $\text{view} * \text{TG}$

- Al vertex shader (en GLSL):

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

➤ es fa el càlcul de la matriu per a cada vèrtex

- Al programa (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

# Anàlisi del codi de l'esquelet

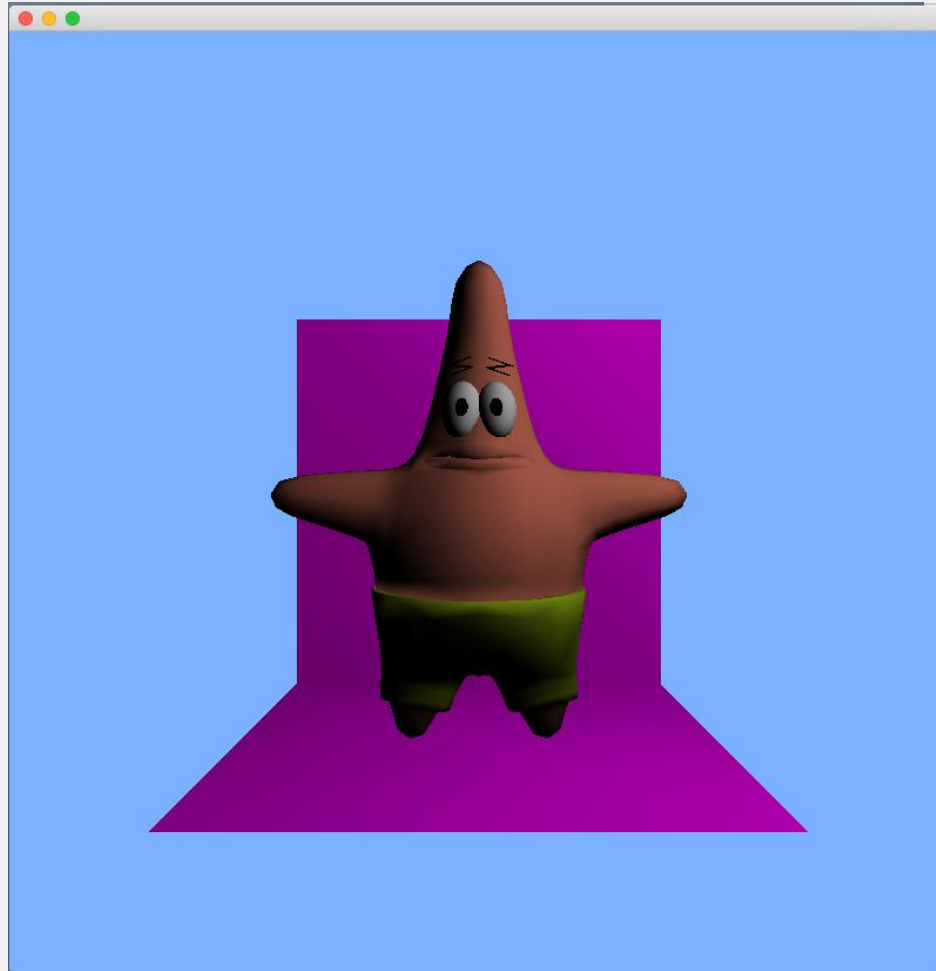
- Analitzar quins mètodes implementats.
- Analitzar implementació dels mètodes.
  - Quina càmera tenim?
  - Quina escena?
  - Quina interacció?
- Analitzar els shaders
  - Atributs, uniforms, funcions

# Exercici 1

Càlcul color usant model Lambert:

```
vec3 Ambient () {  
    return llumAmbient * matamb;  
}  
  
vec3 Difus (vec3 NormSCO, vec3 L, vec3 colFocus) {  
    // Sols retorna el terme de Lambert (sense ambient)  
    // s'assumeix que els vectors que rep com a paràmetres estan normalitzats  
    vec3 colRes = vec3(0); // Inicialitzem color a component ambient  
    // Càlcul component difusa, si n'hi ha  
    if (dot (L, NormSCO) > 0)  
        colRes = colFocus * matdiff * dot (L, NormSCO);  
    return (colRes);  
}
```

Cal calcular en *main*: L en SCO, Normal en SCO,  
normalitzar vectors i calcular Lambert (ambient + difús)



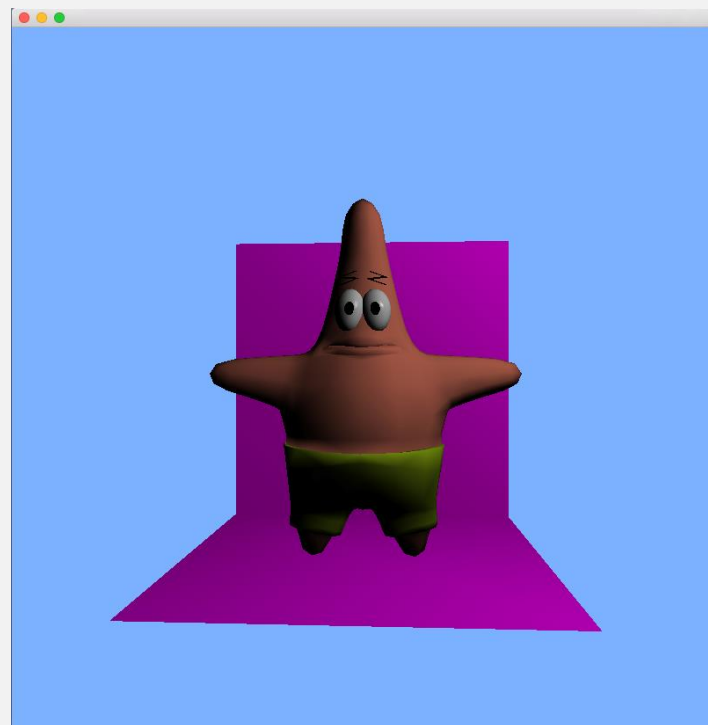
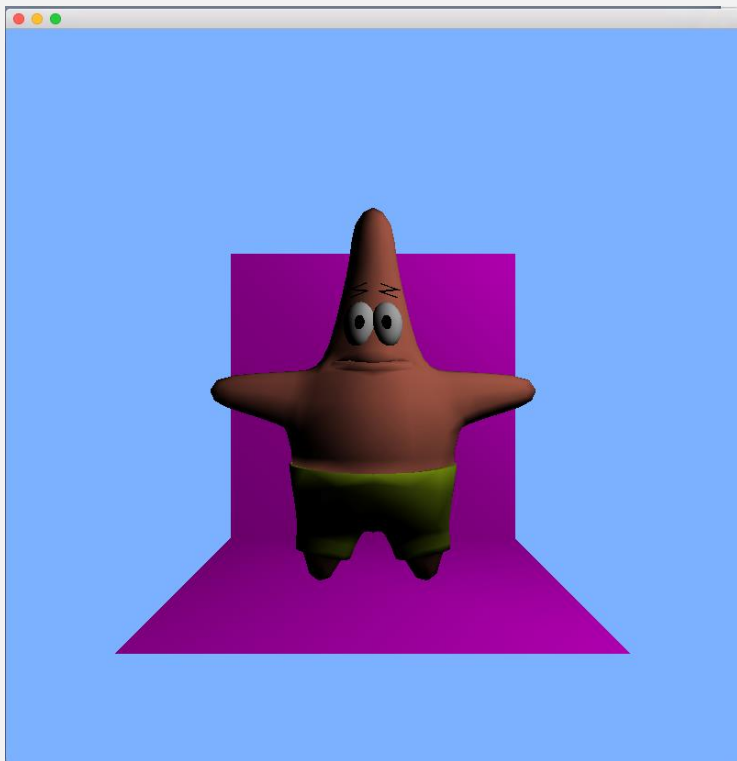
Proveu a moure càmera (si voleu poseu també rotació en X) i veureu que cares il·luminades no varien. **Llum d'escena.**



# Exercici 2

## Càlcul color usant model Phong:

```
vec3 Especular (vec3 NormSCO, vec3 L, vec4 vertSCO, vec3 colFocus) {  
    // Sols retorna el terme especular  
    // Els vectors rebuts com a paràmetres (NormSCO i L) estan normalitzats  
    vec3 colRes = vec3(0);  
    // Si la llum ve de darrera o el material és mate no fem res  
    if ((dot (NormSCO, L) < 0) || (matshin == 0))  
        return colRes; // no hi ha component especular  
    // Calculem R i V  
    vec3 R = reflect (-L, NormSCO); // equival a: 2.0 * dot (NormSCO, L) * NormSCO - L;  
    vec3 V = normalize (-vertSCO.xyz); // perquè la càmera està a (0,0,0) en SCO  
    if (dot (R, V) < 0)  
        return colRes; // no hi ha component especular  
    // Afegim la component especular  
    float shine = pow (dot (R, V), matshin);  
    return (matspec * colFocus * shine);  
}
```



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

**Llum d'escena.**

El terra pot tenir taca especular?

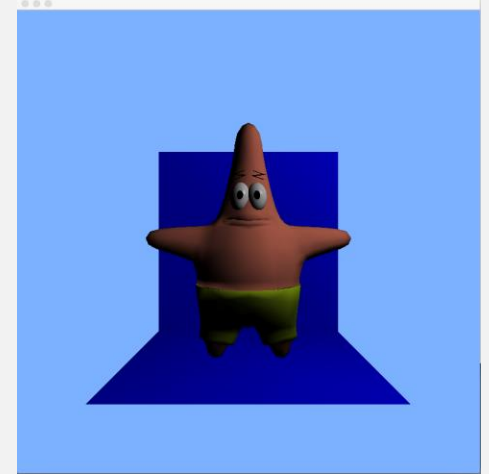
# Exercicis 3 i 4

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



# Exercici 5

Pas 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

# Exercici 6

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

- K  $\rightarrow$  mou el focus cap a les X-
- L  $\rightarrow$  mou el focus cap a les X+