

Classe 1: Contingut

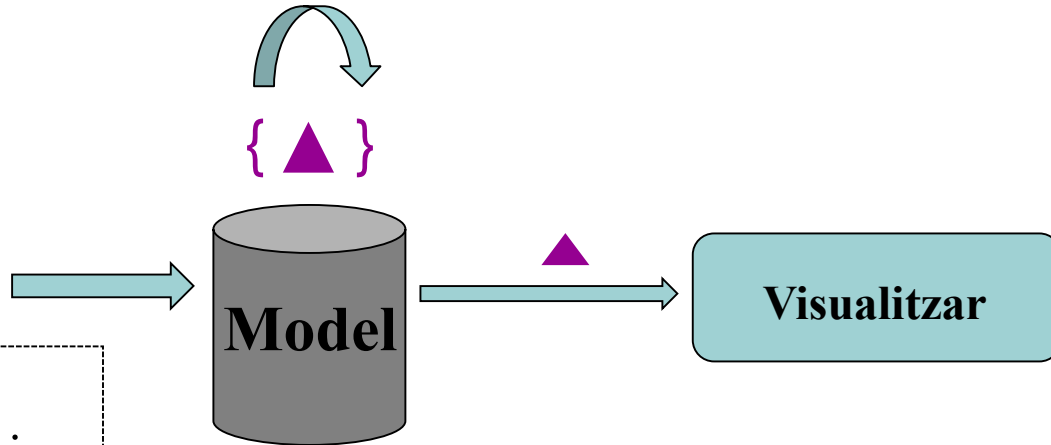
- Introducció a la Informàtica Gràfica
- **Models geomètrics**
 - **Un objecte**
 - Un conjunt d'objectes (escena)

Bibliografia del “Llibre en CD” els temes:

- Geometria2D i 3D.
- Representació d'objectes geomètrics

En el marc d'IDI...

Dades

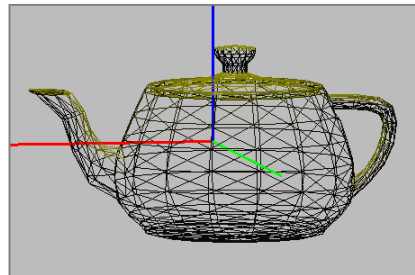


Sòlids

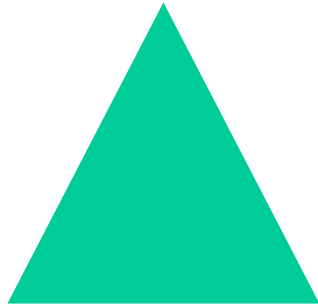
- Limitats per superfície
- Moviments no modifiquen forma

Models Geomètrics:

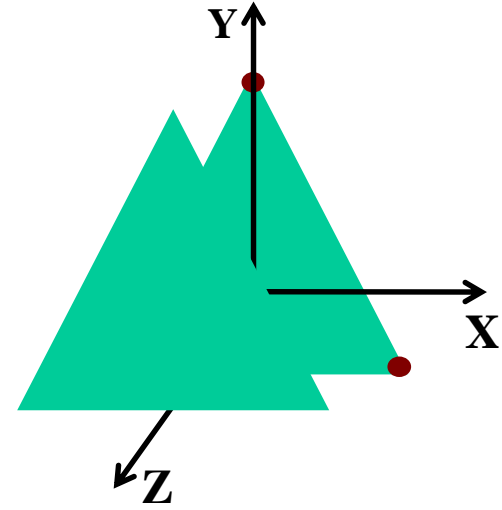
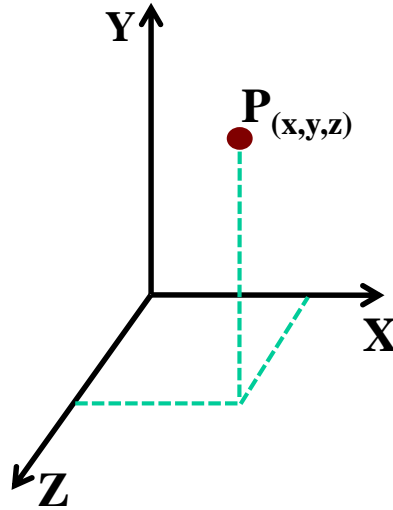
- Model de Fronteres
aproximació per cares planes => triangles
- Procedural, CSG, octrees,...



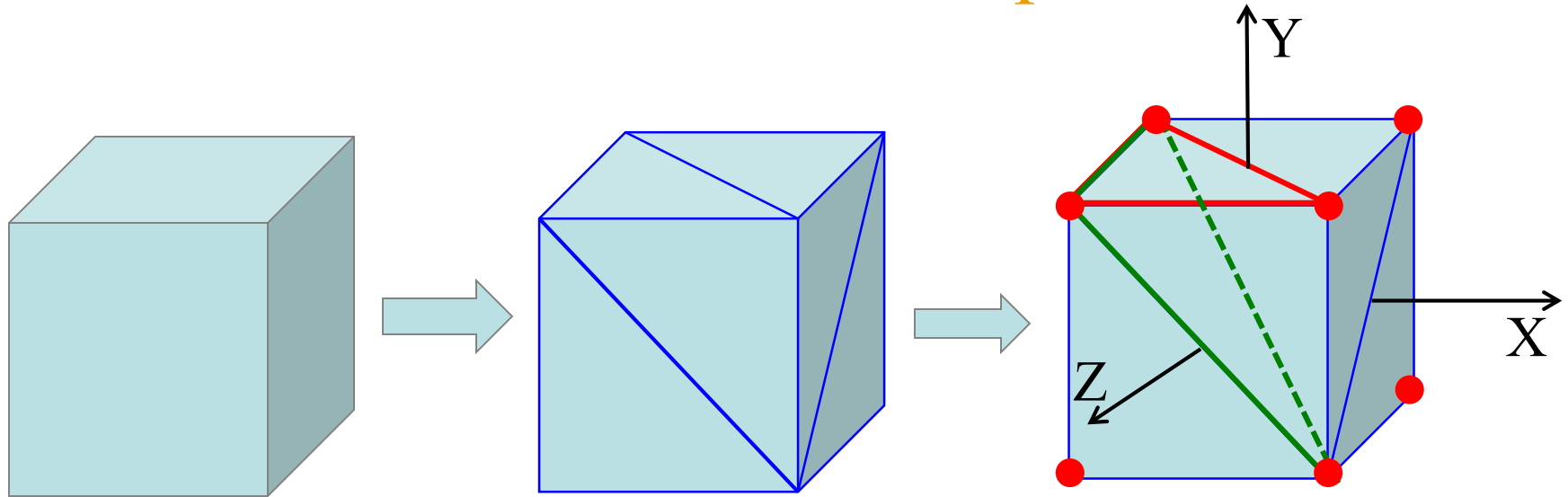
Models Geomètrics (intro)



• *Com guardar?*



Model Fronteres: Exemple Cub



Per cada triangle

- Geometria
- Topologia (implicítament)

Vèrtexs repetits ☹️

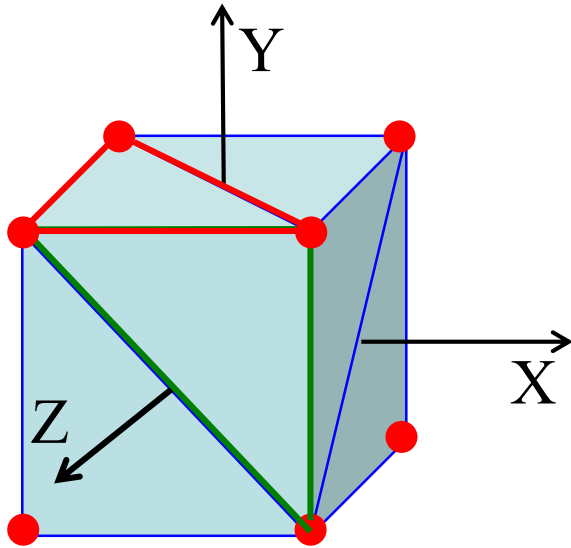
Vèrtexs

x	y	z
-1	+1	-1
-1	+1	+1
+1	+1	+1
...

Vèrtexs

x	y	z
-1	1	-1
-1	1	1
1	-1	1
1	-1	-1
1	1	-1
1	1	1
-1	-1	-1
-1	-1	1

Model Fronteres: Exemple Cub



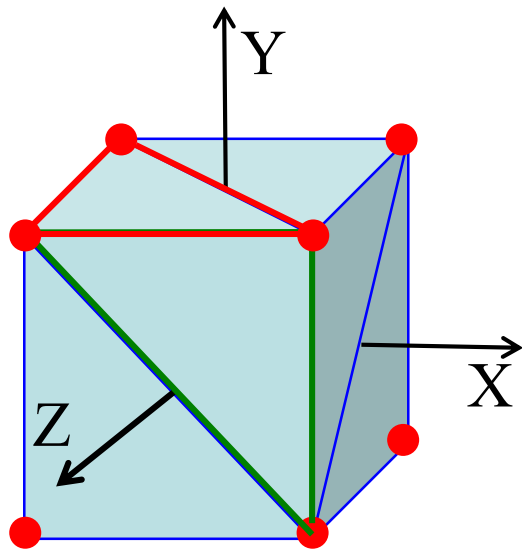
Cares

normal	<i>Id Vèrtexs</i>
a_1, b_1, c_1	1,2,3
a_2, b_2, c_2	2,4,3
...	...

Vèrtexs

x	y	z
-1	1	-1
-1	1	1
1	1	1
1	-1	1
1	-1	-1
1	1	-1
-1	-1	-1
-1	-1	1

Model Fronteres: conjunt de triangles



Vèrtexs

x	y	z
-1	1	-1
-1	1	1
1	1	1
-1	1	1
1	-1	1
1	1	1
.	.	.
.	.	.

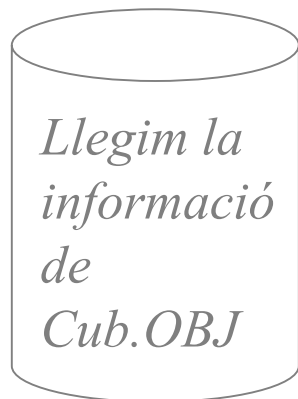
Topologia implícita

Cares

normal	Id Vèrtexs
a_1, b_1, c_1	1,2,3
a_2, b_2, c_2	2,4,3
...	...

Vèrtexs

x	y	z
-1	1	-1
-1	1	1
1	1	1
1	-1	1
1	-1	-1
1	1	-1
-1	-1	-1
-1	-1	1

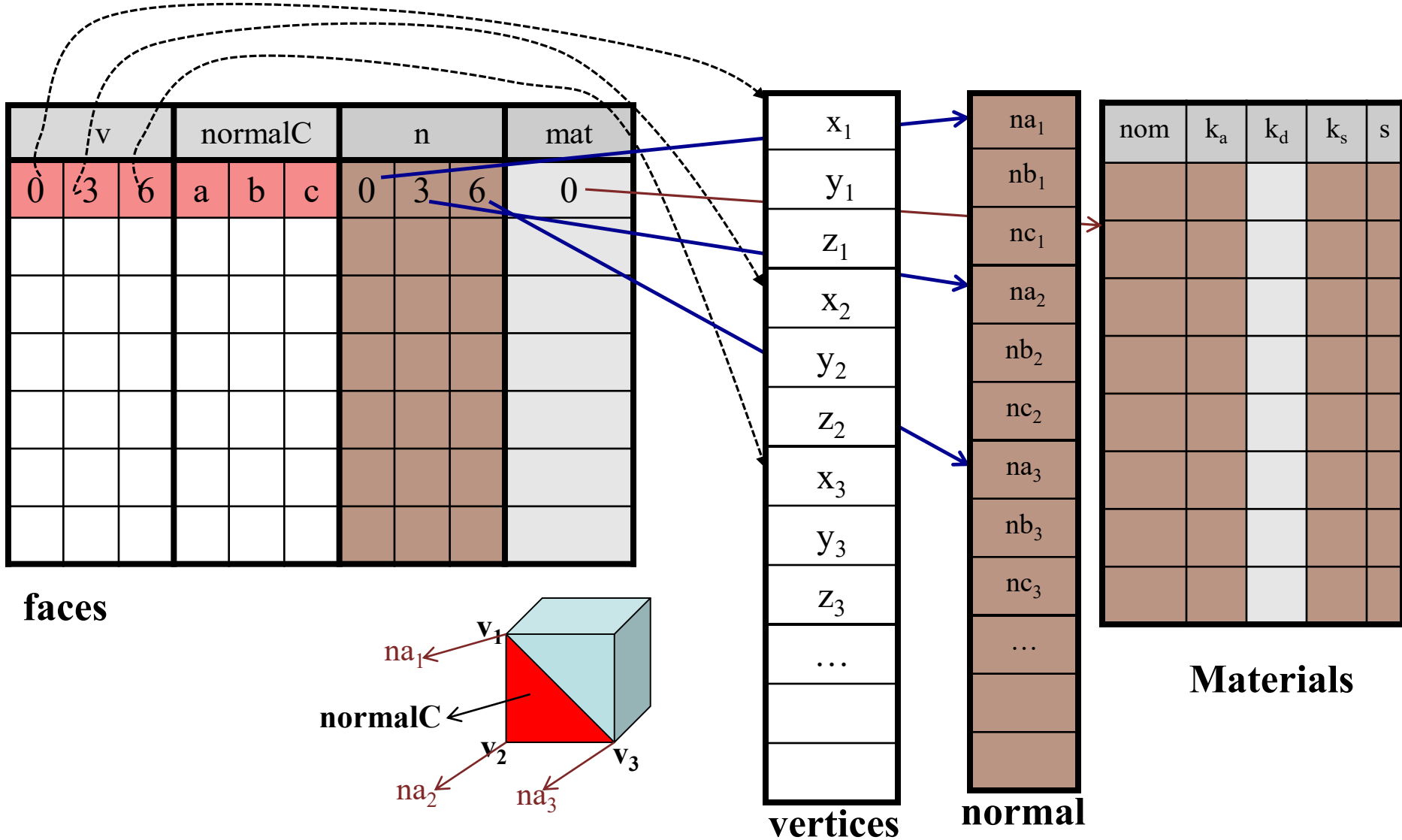


Topologia explícita

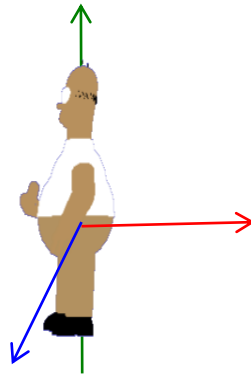
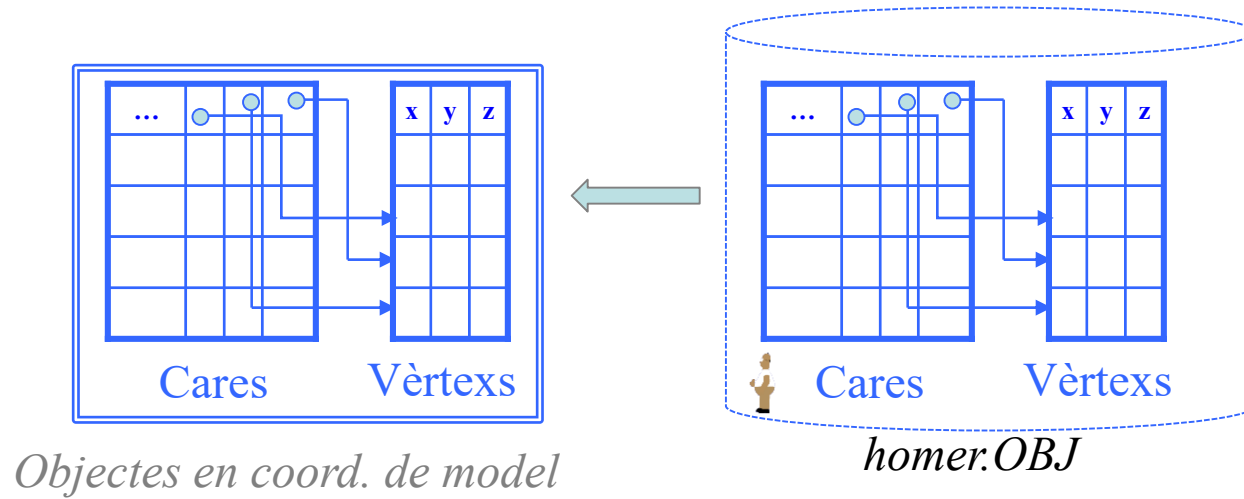


Assignem valors en el codi

Exemple: Laboratori



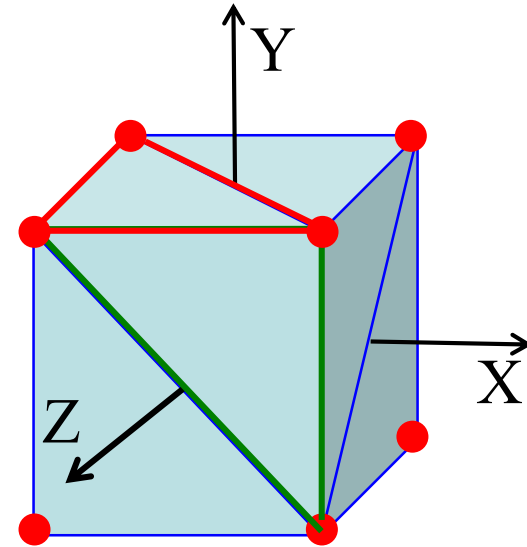
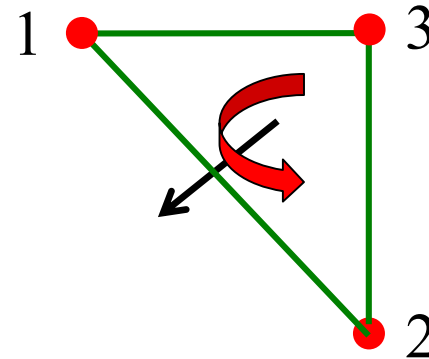
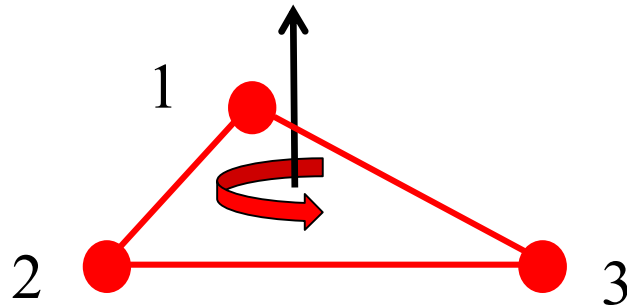
En el marc de les classes de teoria d'IDI...



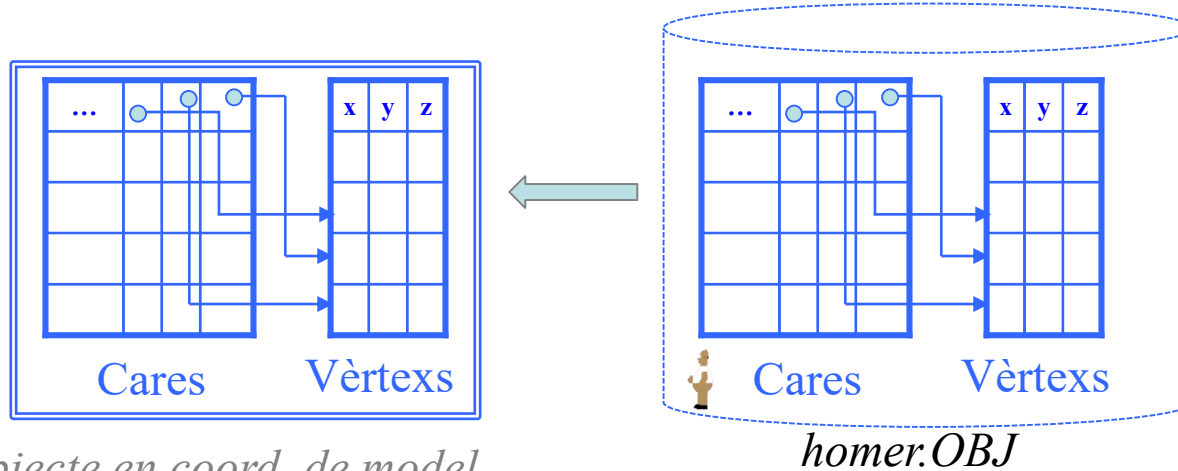
*Com podem saber si
el model és vàlid?*

Model Fronteres: propietat de ser vàlid

- Cares “orientades”.
- Ordenació vèrtexs coherent amb l'orientació de les cares.
- Cada aresta separa 2 cares.



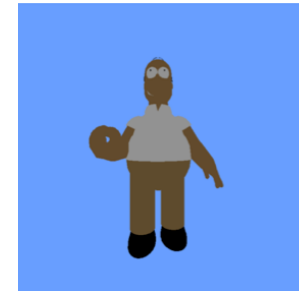
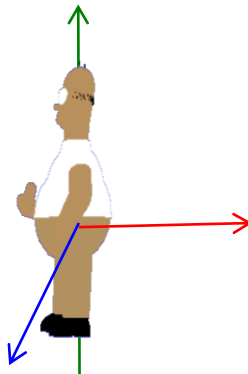
En el marc de les classes de teoria d'IDI...



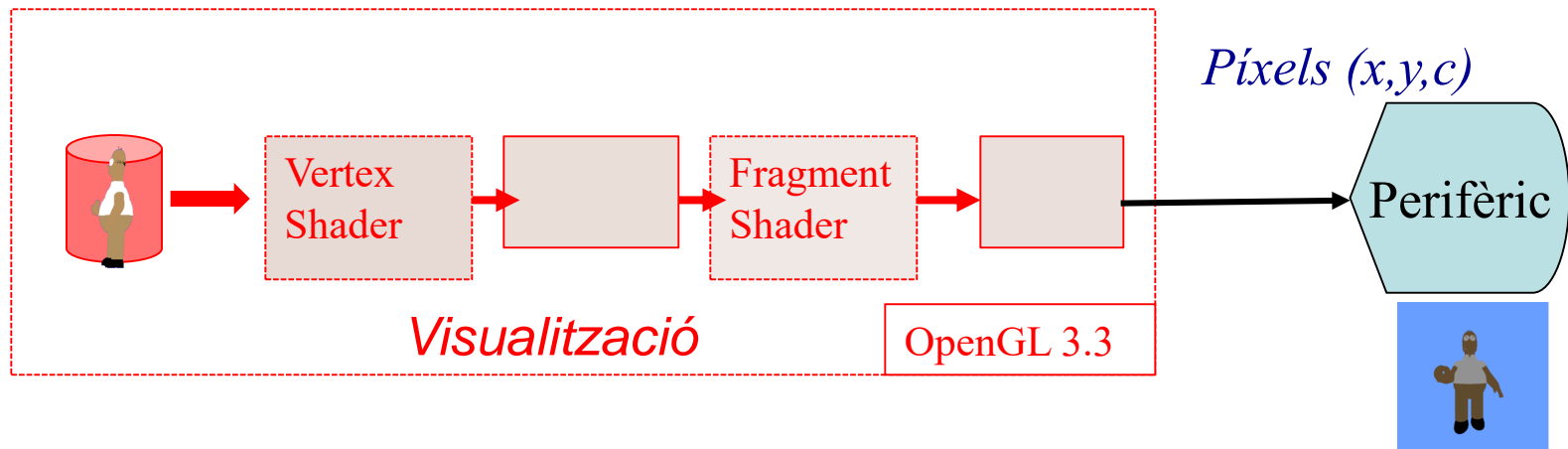
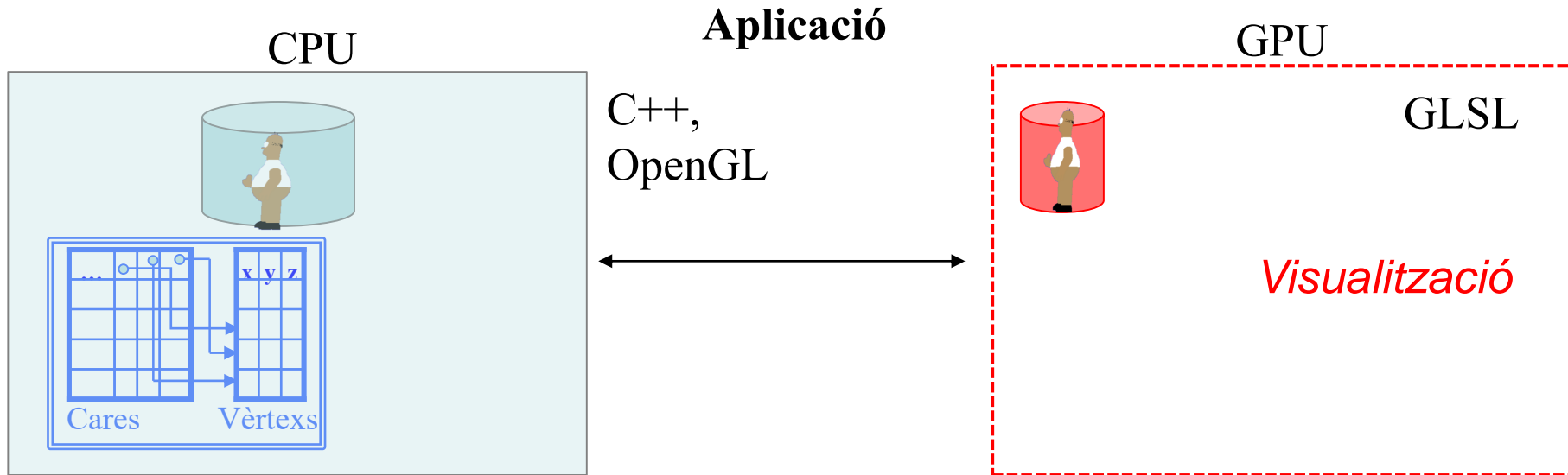
Objecte en coord. de model

homer.OBJ

*Com podem
visualitzar el model?*

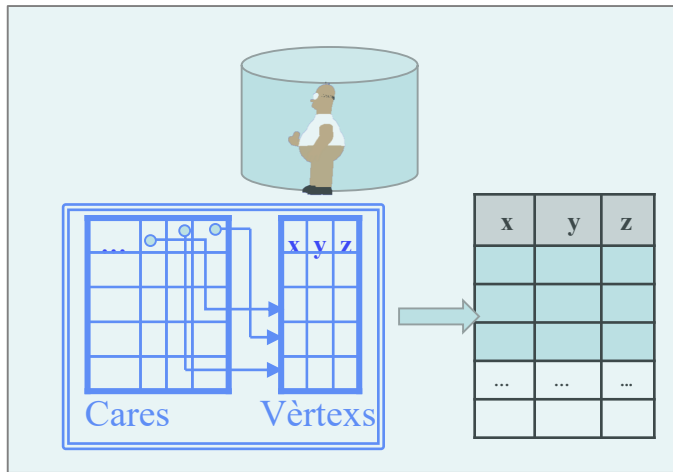


Pintar en OpenGL 3.3: “core” mode



Pintar en OpenGL 3.3: “core” mode

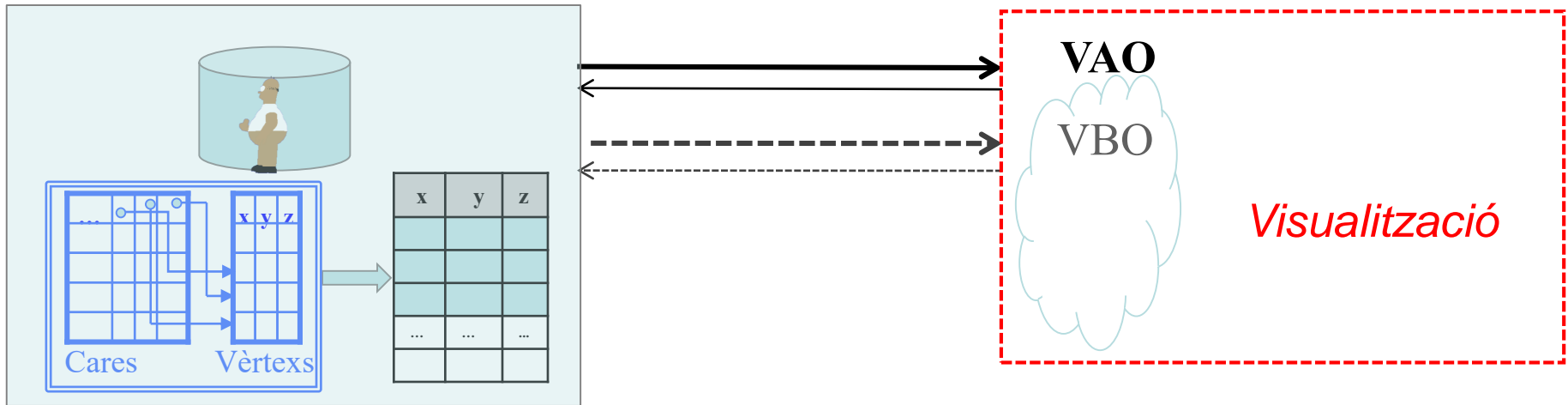
- **Un sol cop** cal enviar/passar el *model/geometria* a la GPU com una llista de triangles amb les coordenades dels vèrtexs i topologia implícita (o altres opcions com “strips”).
- Per tant, si ens cal, a partir del model haurem de crear una estructura auxiliar/temporal amb la informació en aquest format per poder enviar-la a la GPU.



Visualització

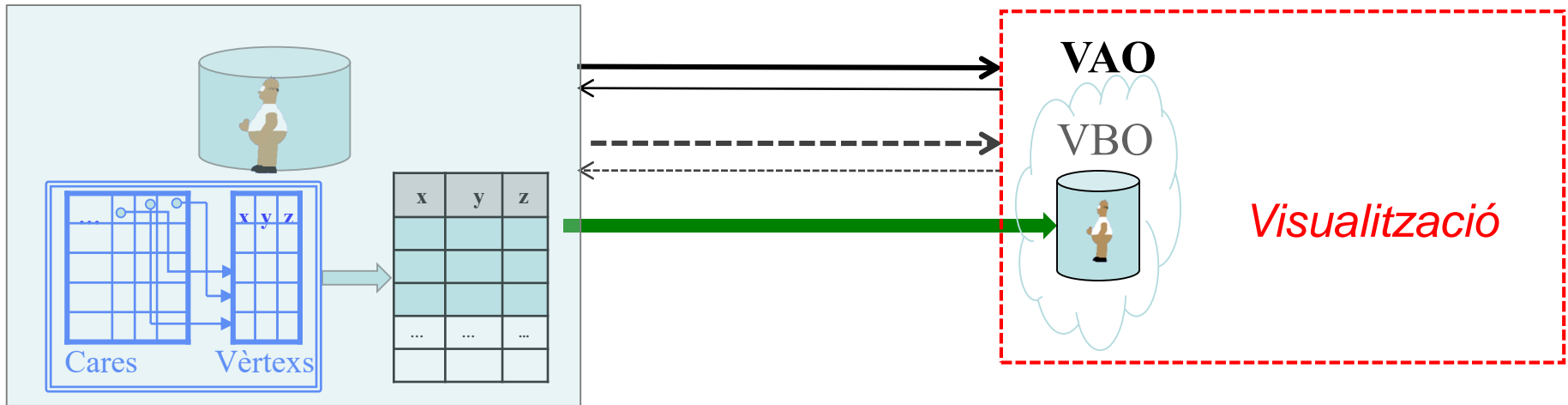
Pintar en OpenGL 3.3: “core” mode

1. Crear en GPU/OpenGL un *VAO* que encapsularà dades del model.
Crear *VBO* que guardarà les coordenades dels vèrtexs (potser cal altres per normal, color,...)



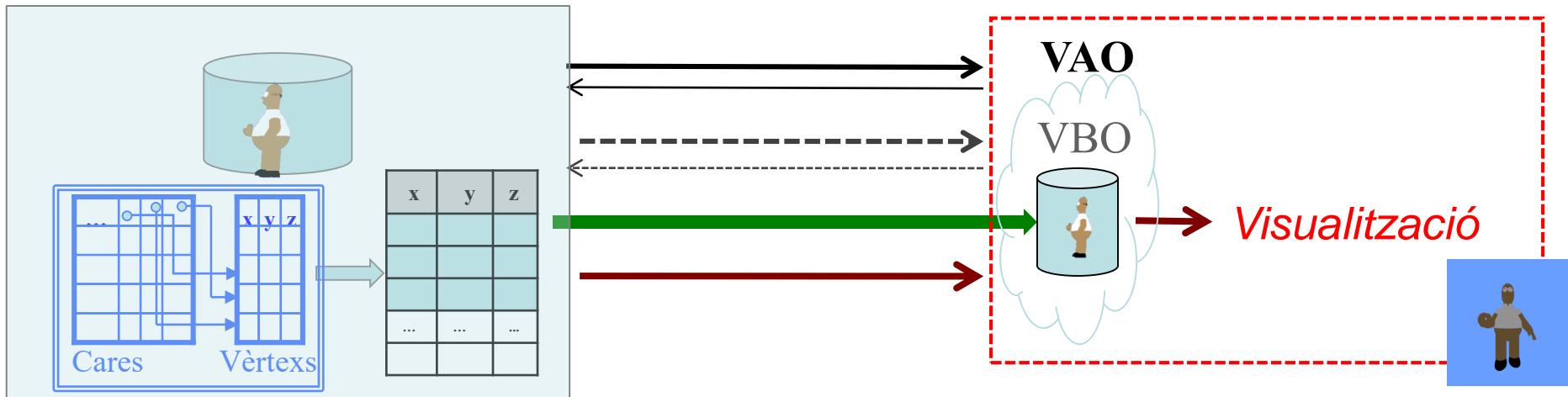
Pintar en OpenGL 3.3: “core” mode

1. Crear en GPU/OpenGL un *VAO* que encapsularà dades del model.
Crear *VBO* que guardarà les coordenades dels vèrtexs (i si cal, normal, color,...)
2. Guardar la llista de vèrtexs en el *VBO*
(i si cal, color i normal en els seus *VBO*)

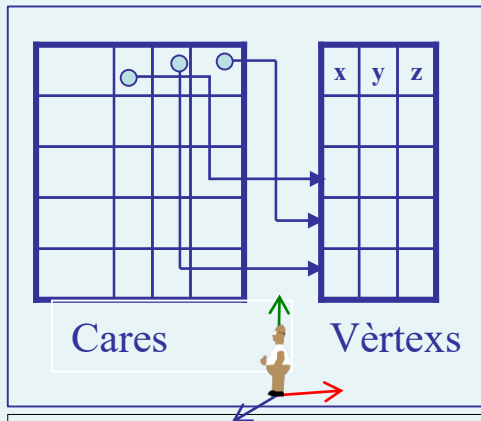


Pintar en OpenGL 3.3: “core” mode

1. Crear en GPU/OpenGL un *VAO* que encapsularà dades del model.
Crear *VBO* que guardarà les coordenades dels vèrtexs (i si cal, normal, color,...)
2. Guardar la llista de vèrtexs en el *VBO*
(i si cal, color i normal en els seus *VBO*)
3. **Cada cop** que es requereix pintar, indicar el *VAO* a pintar i dir que es pinti: *glDrawArrays(...)*. Acció **pinta_model()** a teoria.

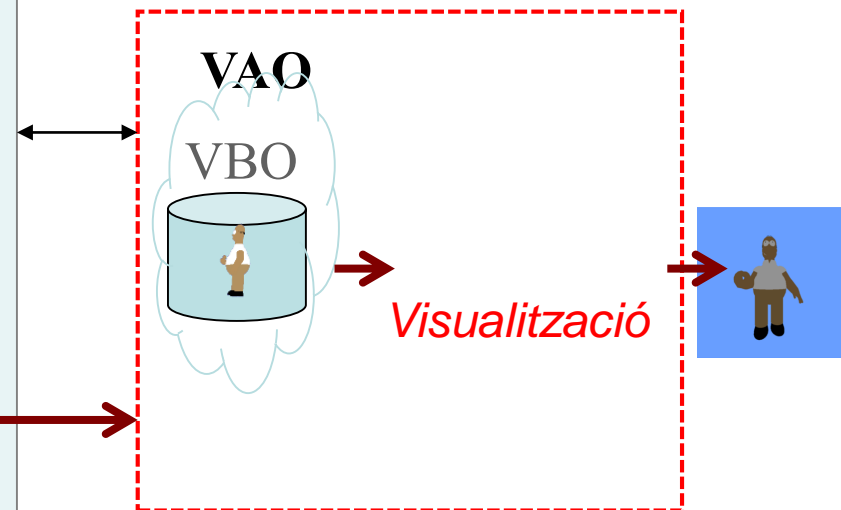


Pintar en OpenGL 3.3: “core” mode



```
// un sol cop  
“crear” i omplir un VAO i VBOi
```

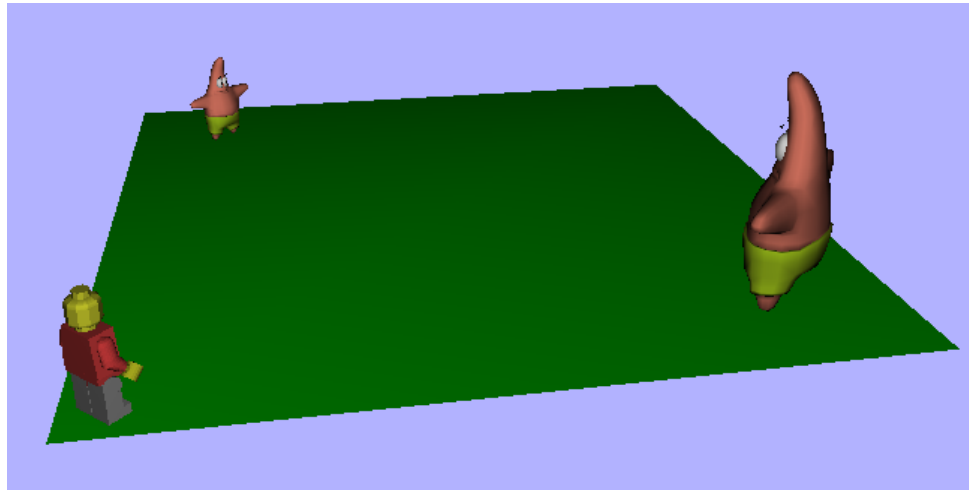
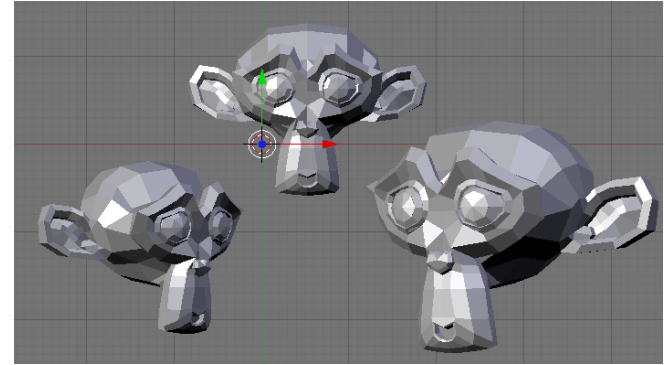
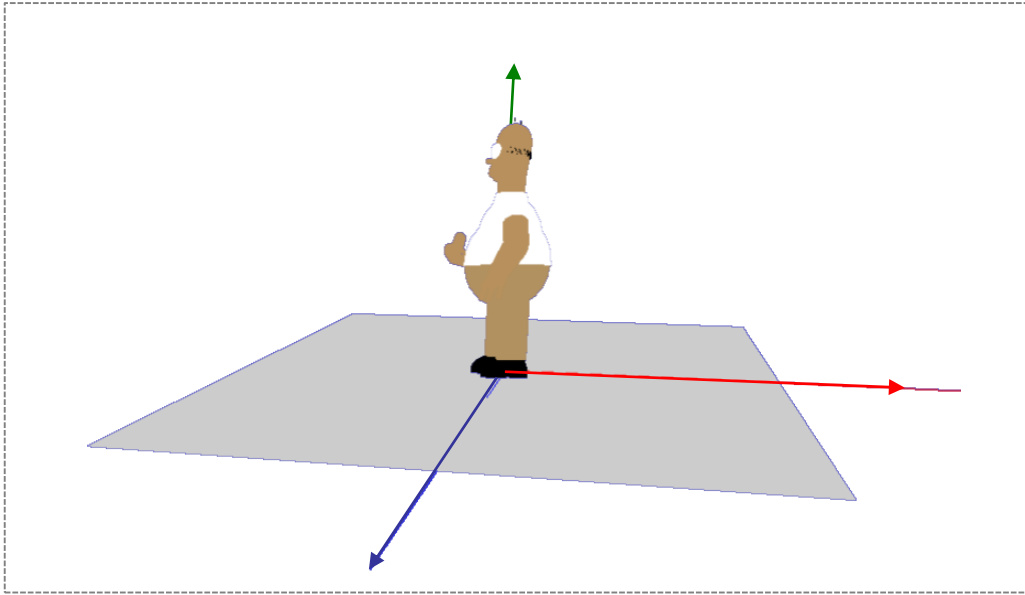
```
/*cada cop que cal refrescar finestra  
activa VAO i glDrawArrays(...)*/  
pinta_model()
```



Classe 1: Contingut

- Introducció a la Informàtica Gràfica
- **Models geomètrics**
 - Un objecte
 - **Un conjunt d'objectes (escena)**

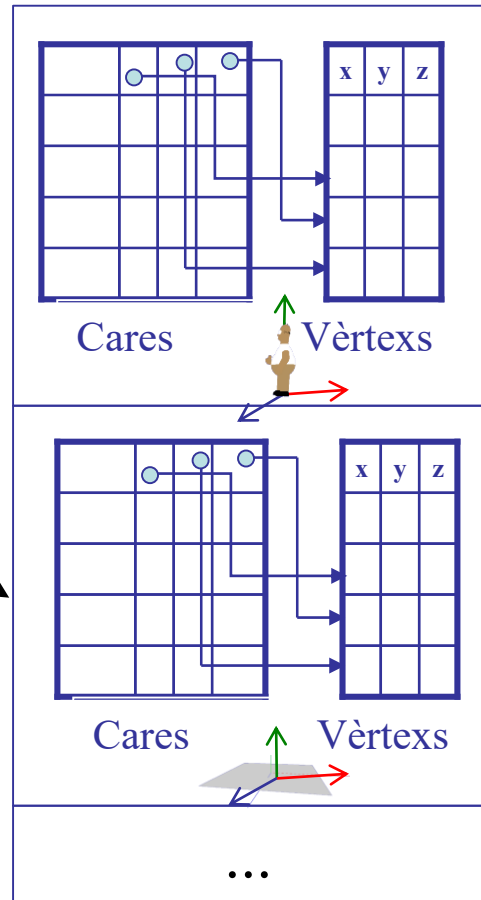
Com guardar i pintar “escenes”?



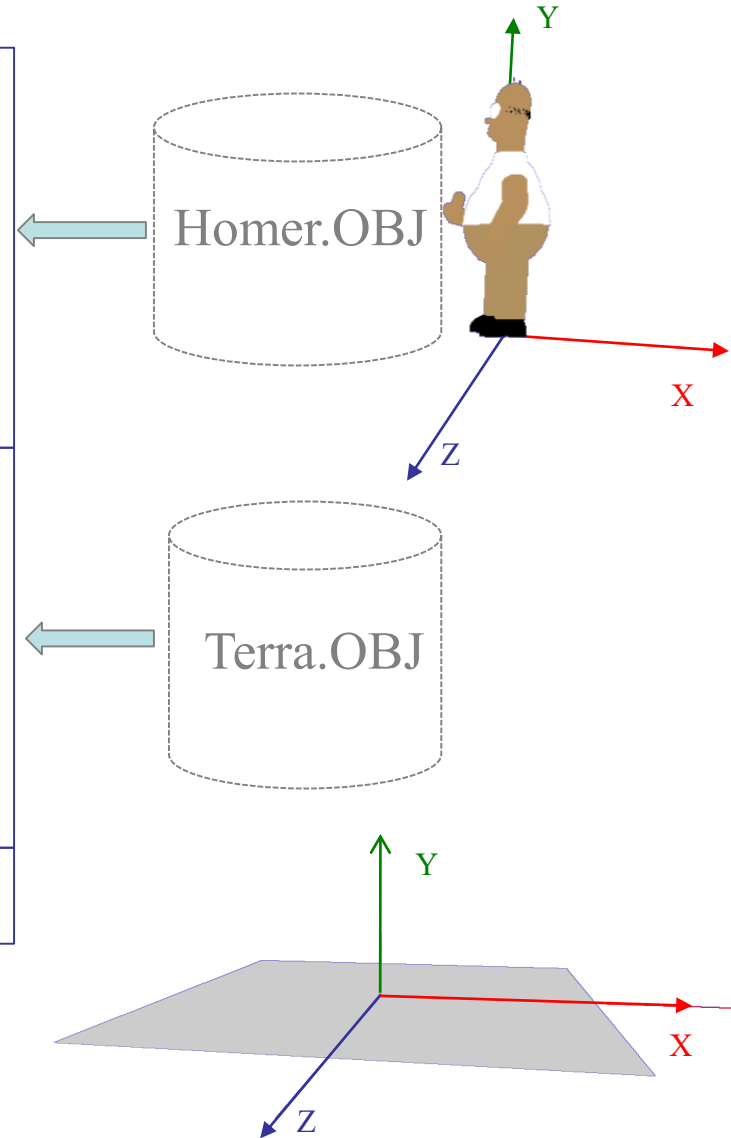
Escena : {objectes}

nom	model
Homer		
Terra		

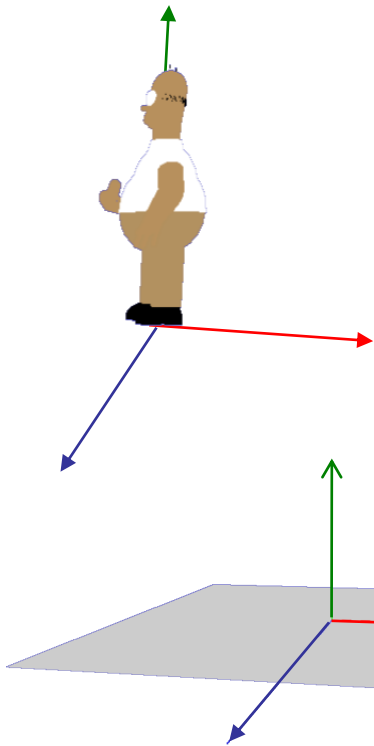
Objectes



Models



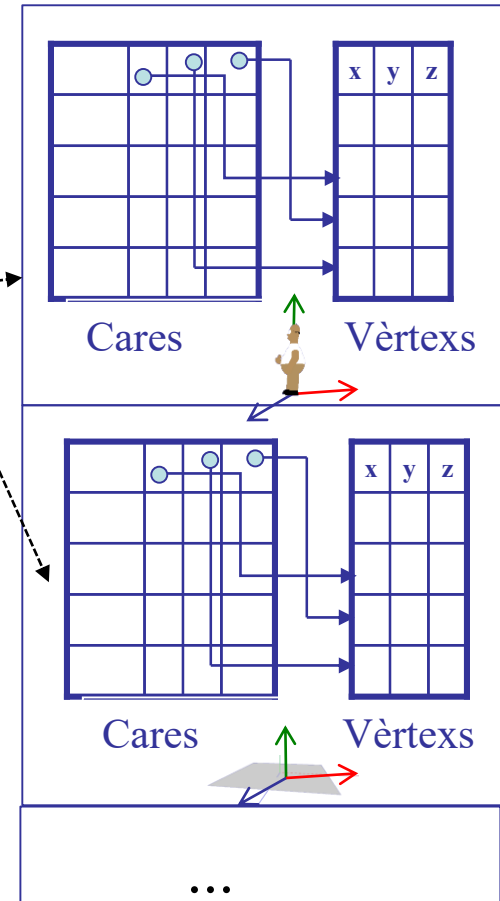
Escena : {objectes} ... com visualitzar?



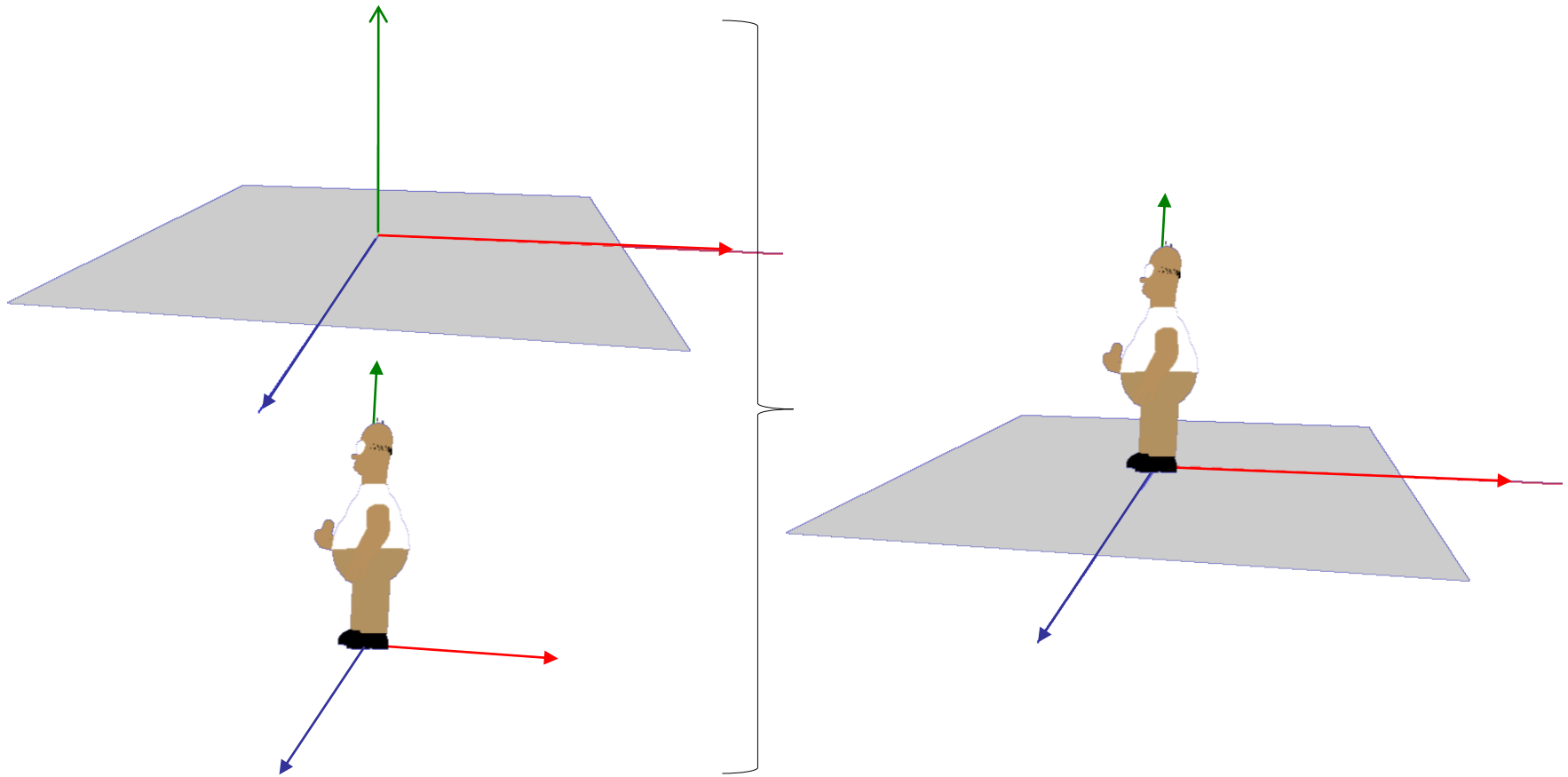
nom	model
Homer		
Terra		

```
// un sol cop  
per cada model  
crear i omplir VAOi i VBOj  
fper
```

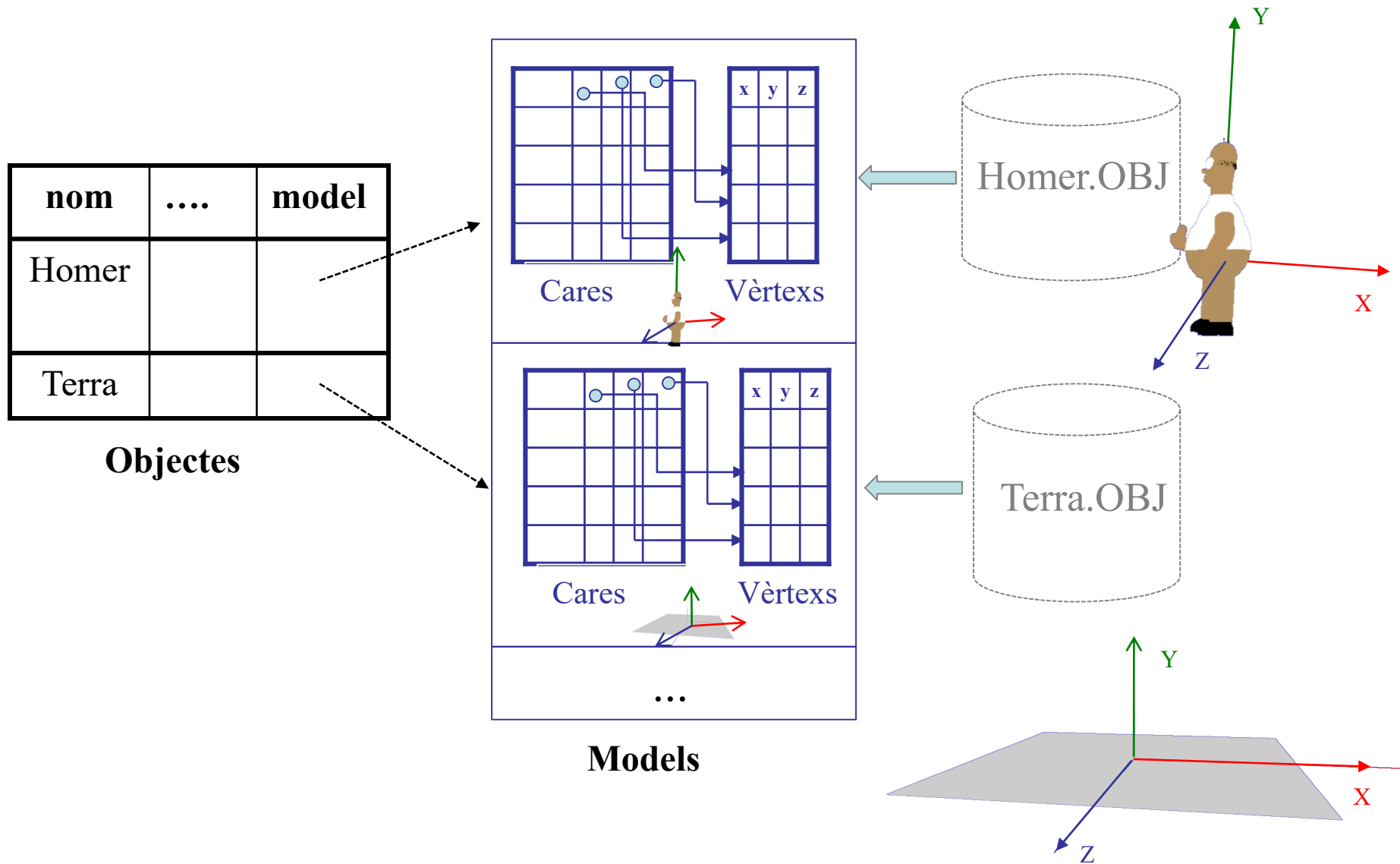
```
//cada cop que cal refrescar finestra  
per cada objecte  
/*pinta el seu model: activa VAO  
i glDrawArrays(...)*/  
pinta_modeli()  
fper
```



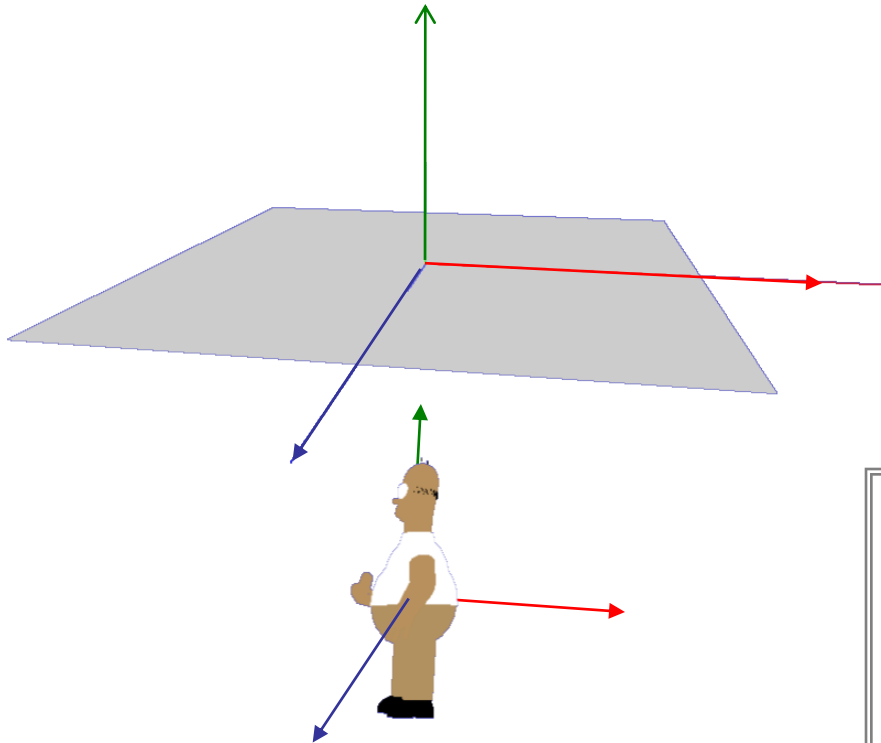
Imatge obtinguda...



Autre exemple: Homer definit en SC centrat en Homer



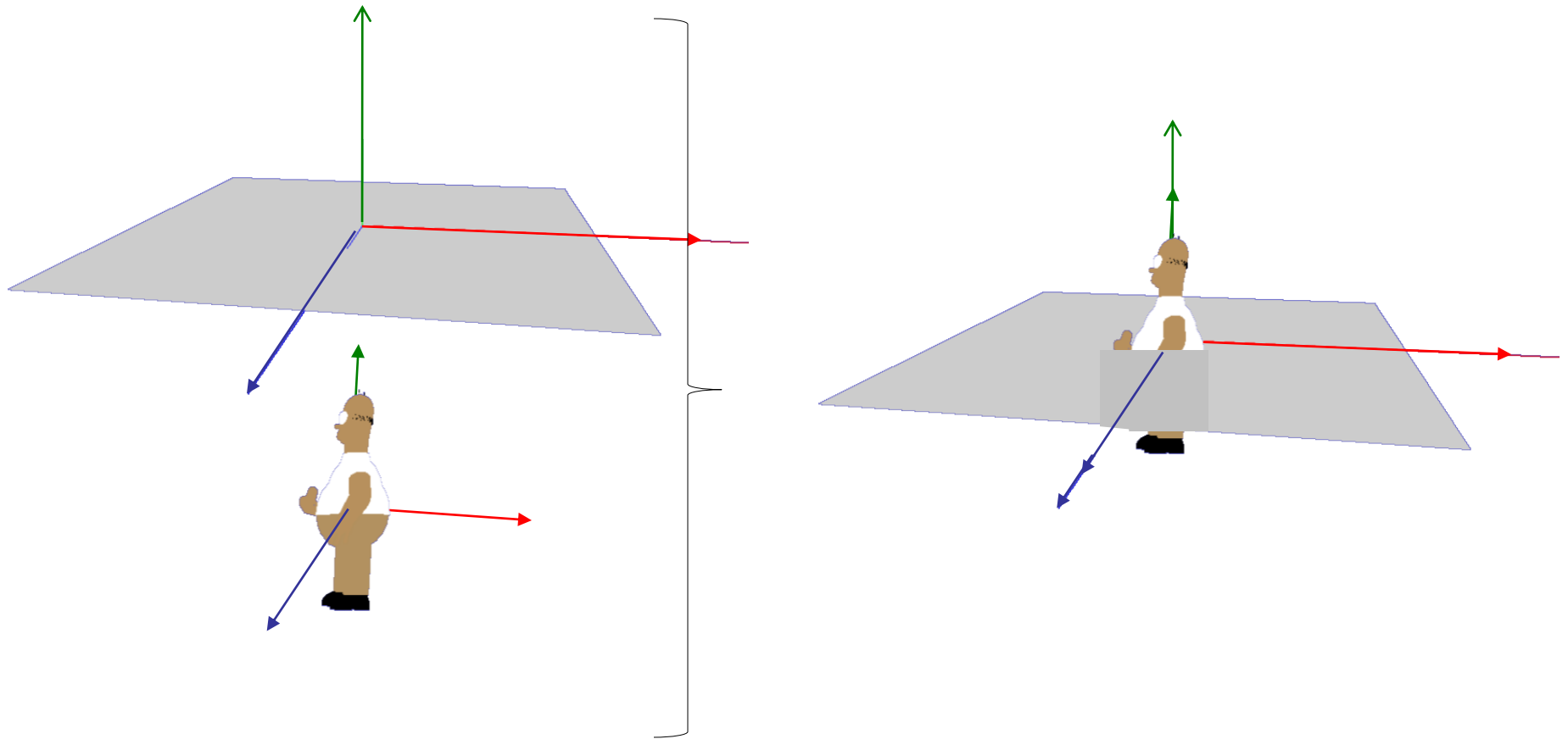
Igual que abans, per visualitzar...



```
// un sol cop  
per cada model  
    crear i omplir VAOi i VBOj  
fper
```

```
//cada cop que cal refrescar finestra  
per cada objecte  
    pinta_modeli()  
fper
```

Resultat....



Repasar Àlgebra i TG

Si volem així...

Moure/pujar
 $\text{Trans}(0,+h,0)$

- Cal aplicar TG que modifica coordenades vèrtexs
- TG queda definida per matriu 4x4:

$$\mathbf{V}_A = \text{TG } \mathbf{V}_m = T(0,+h,0) \mathbf{V}_m$$

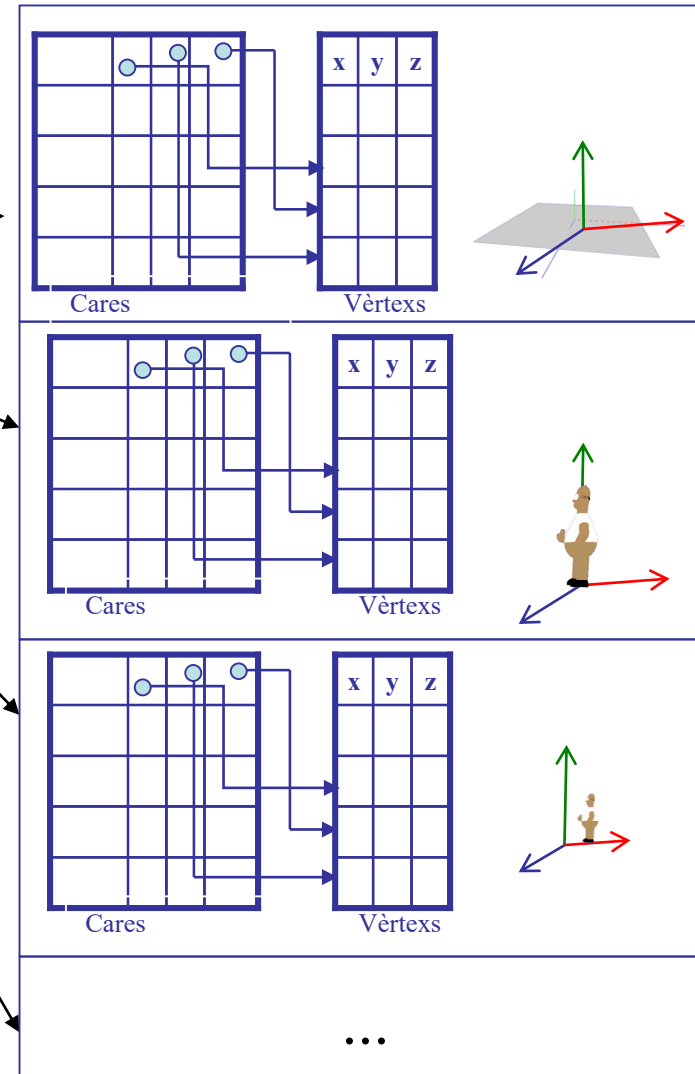
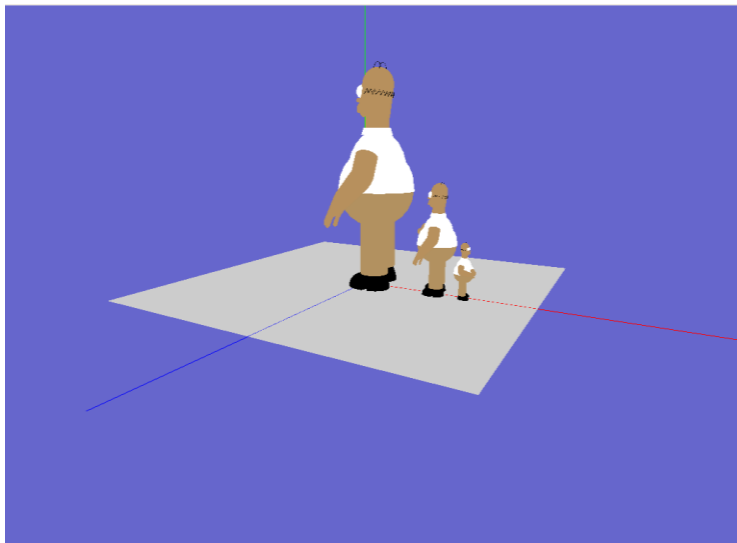
$$T(t_x, t_y, t_z) = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Quan transformem vèrtexs de SCM en SCA?

Escena: {objectes} i models en SCA

nom	model
terra		
h1		
h2		
h3		

Objectes



Models

Escenes: Objectes en SCA. Com fem per pintar?

per cada objecte

```
llegir_Model();
```

```
modelTransformi(TGi);
```

```
aplicar_TG_a_model(TGi);
```

fper

```
/* CreaBuffers que crea un VAOi i  
VBOj per cada objecte*/
```

per cada objecte

```
crear i omplir VAOi,VBOj
```

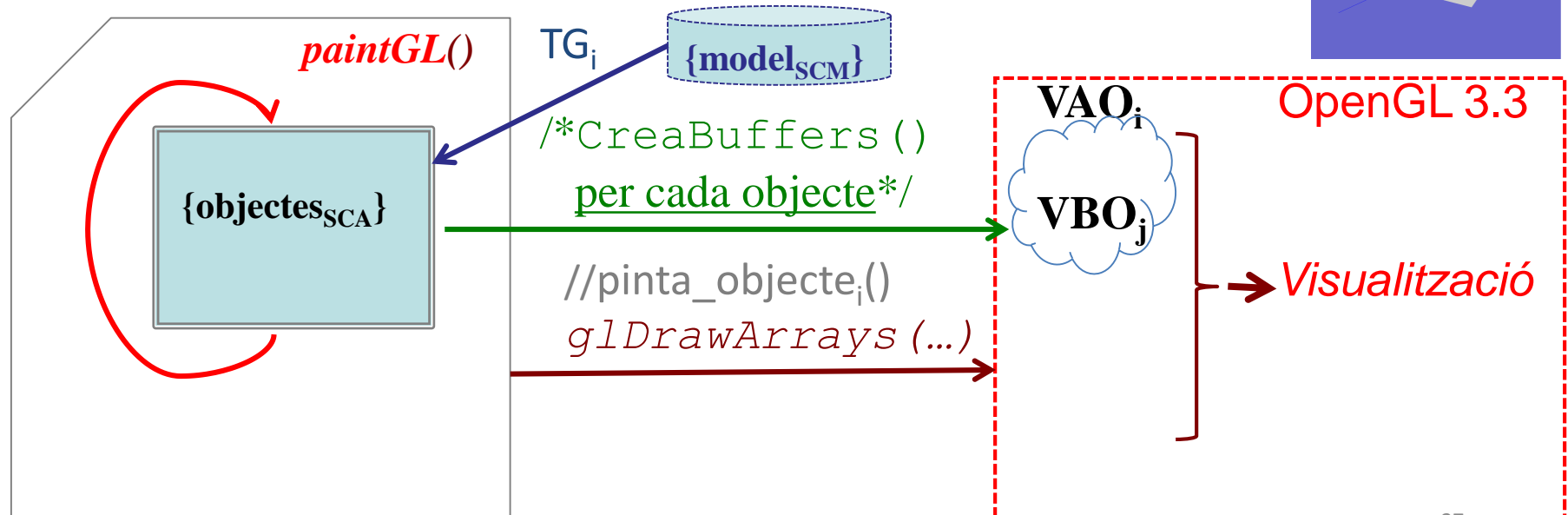
fper

```
//paintGL ();
```

```
per cada objectei
```

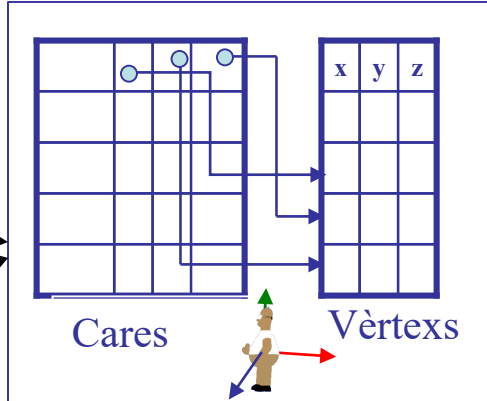
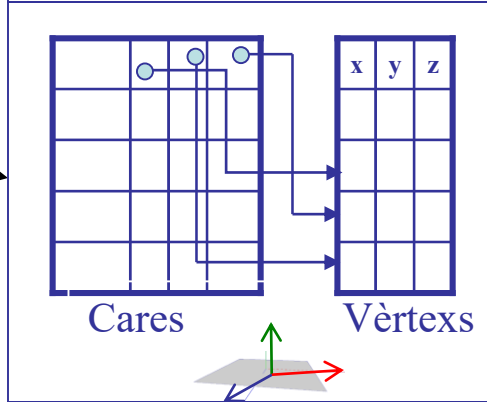
```
pinta_objectei() ; //el seu VAOi
```

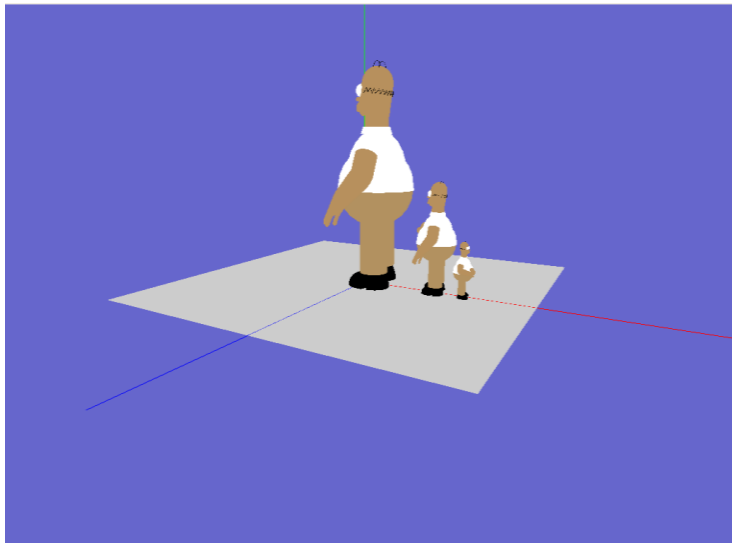
```
fper
```



Escenes: Objectes en SCM. Com fem per pintar?

Objectes

nom	...	TG/param	model
h1			
h2			
h3			
terra			

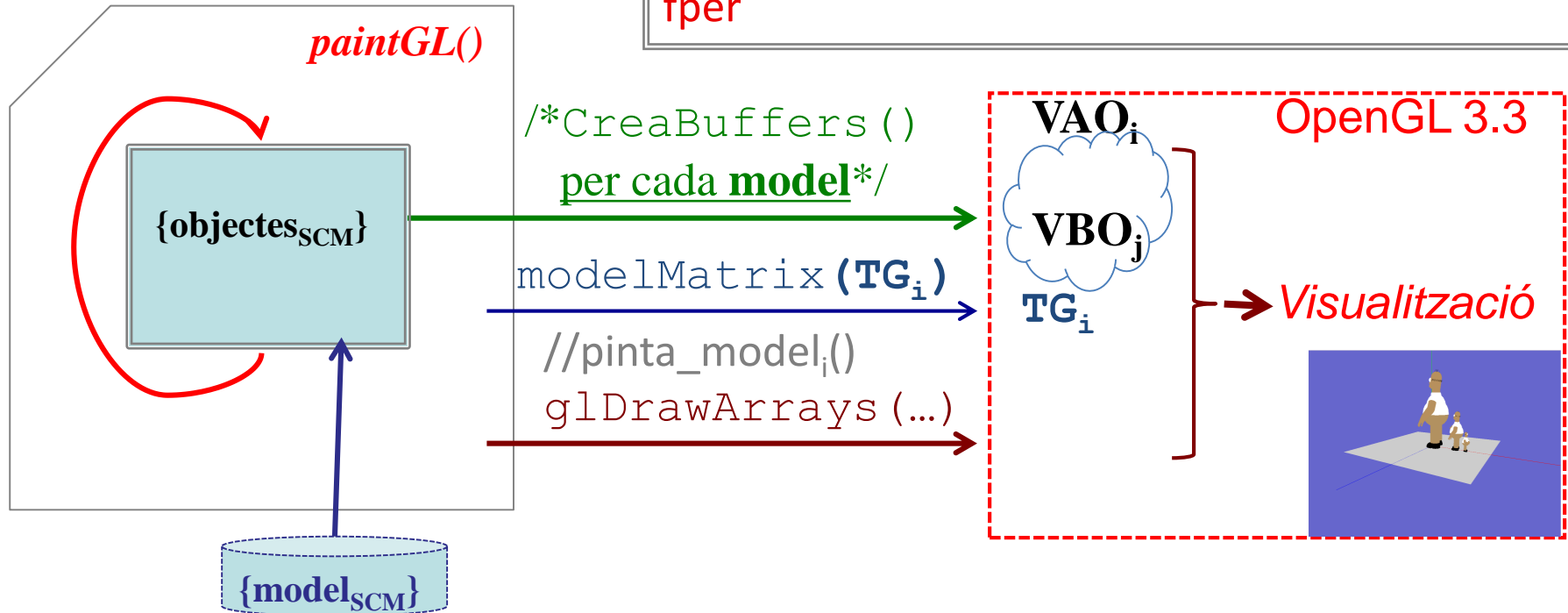


Models

Escenes: Objectes en SCM. Com fem per pintar?

```
/*crear un únic VAOi i VBOj per cada  
model, en CreaBuffers()*/  
per cada model  
    llegir_Model();  
    crear i omplir VAOi,VBOj  
fper
```

```
//paintGL ();  
per cada objectei  
    /*Calcular la TGi a aplicar a model i  
    indicar a OpenGL la TGi */  
    modelTransformi(TGi);  
    modelMatrix (TGi) ; //envia “uniform”  
    pinta_modeli() ; //el VAOi del seu model  
fper
```



Classe 1: conceptes

- Model de fronteres: com guardar un triangle.
- Topologia implícita i explícita.
- Model vàlid.
- Filosofia de visualització en OpenGL 3.3: programes en CPU i GPU, VAO, VBO, ...
- Escena = conjunt d'objectes.
- SCM i SCA.
- Possibles estructures de dades per escenes.
- Filosofia de visualització d'escenes en OpenGL 3.3.