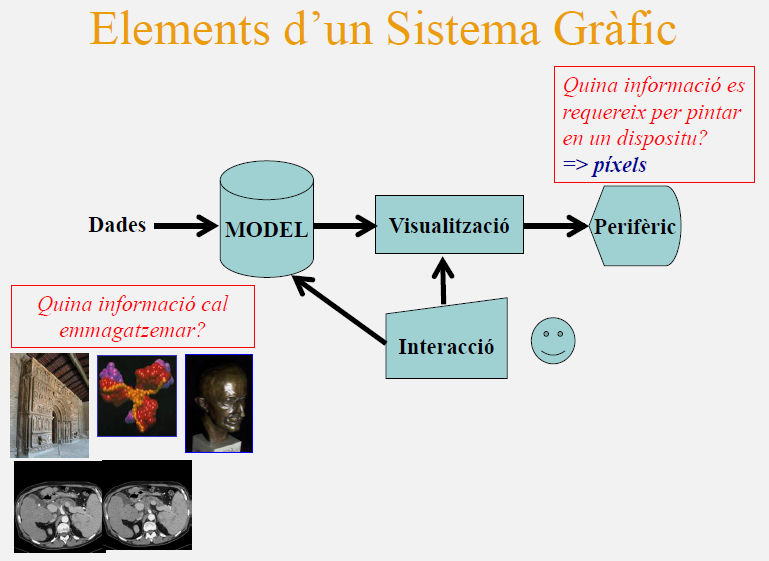
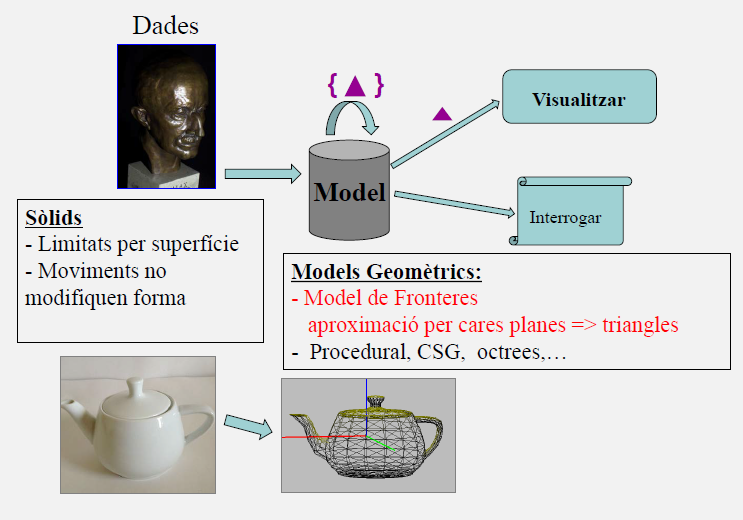
**Introducció a la informàtica gràfica i models geomètrics**

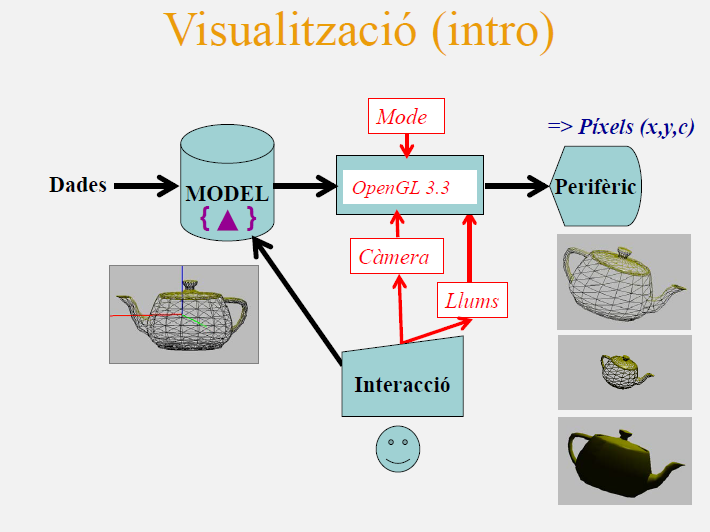
Informàtica gràfica: l’art i la tècnica de comunicar informació utilitzant imatges generades per un computador a partir de models de dades i interactuar amb les dades i la seva visualització.



El sistema gràfic està composta per unes dades que s’emmagatzemen en un model determinat, del qual es pot visualitzar (amb imatges constituïdes amb píxels) en un perifèric (pantalla), i els usuaris poden interactuar amb aquesta imatge/dades.

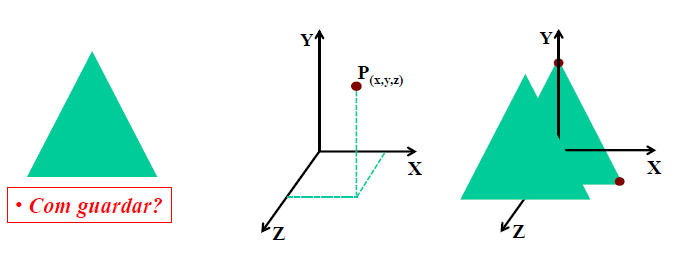


El model usat a l’assignatura es geomètric, es a dir, són sòlids limitats per superfície i els moviments no modifiquen la forma, es a dir, no son deformables, mantenen la mateixa forma sempre. Dintre del model geomètric, s’usa el model de fronteres, que fa una aproximació per cares planes amb triangles (totes les figures seran constituïts per un conjunt triangles).

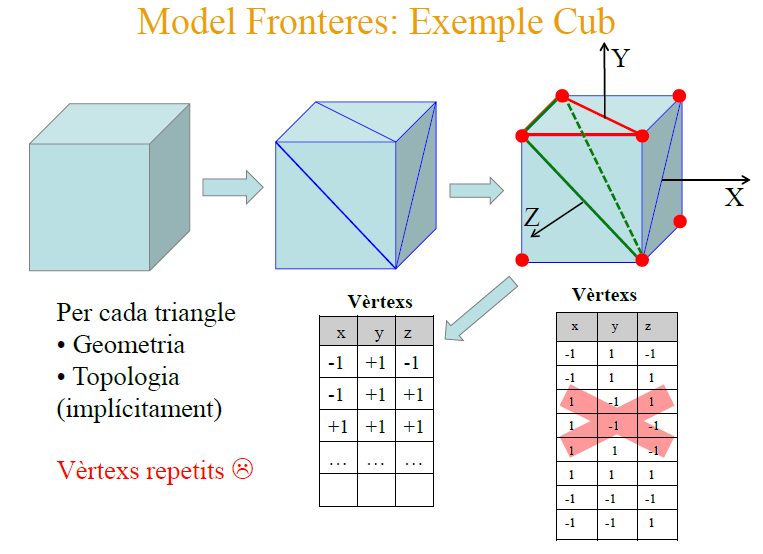


Per visualitzar aquesta imatge necessito una càmera per “fer una foto” a l’objecte real i obtindré la imatge (passar de 3D a 2D). Es important l’angle en què es “fa la foto” i la llum (il·luminació), com també el mode (colors). Tots aquests paràmetres estarà controlat per l’usuari, que es qui ho decideix (part interactiva de l’usuari). Per modificar aquests paràmetres, s’utilitzarà la llibreria OpenGL 3.3.

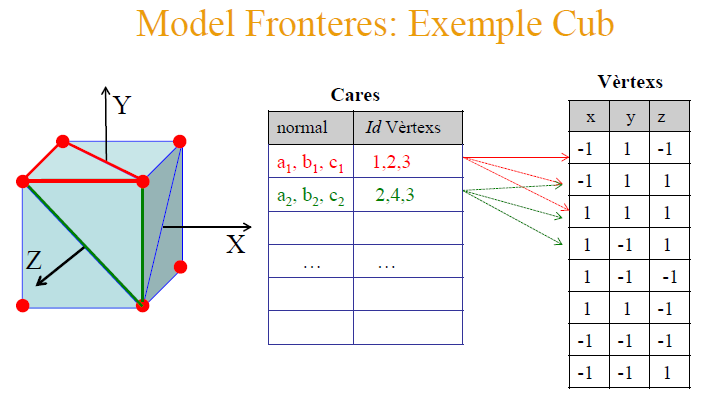
**Models geomètrics per a un objecte**



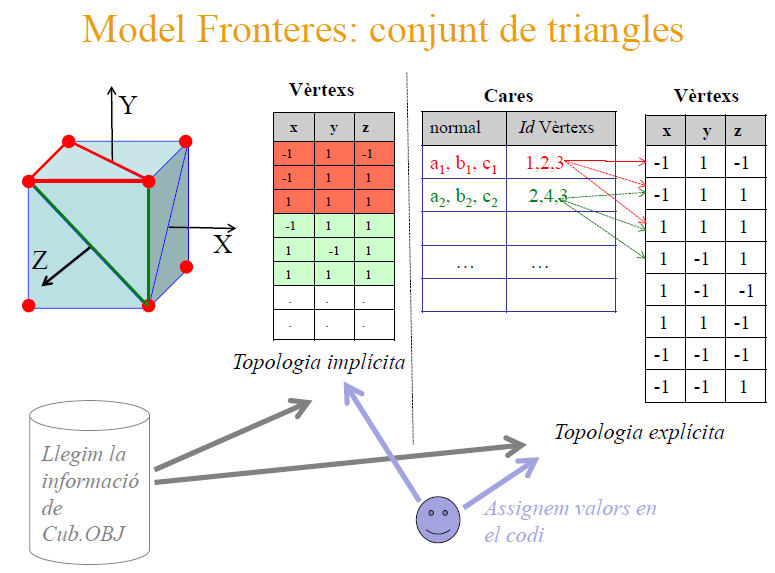
Els triangles els guardarem mitjançant un sistema de coordenades de referencia. Com es en 3D, es té 3 coordenades (X, Y, Z) que representen els tres vèrtexs del triangle.



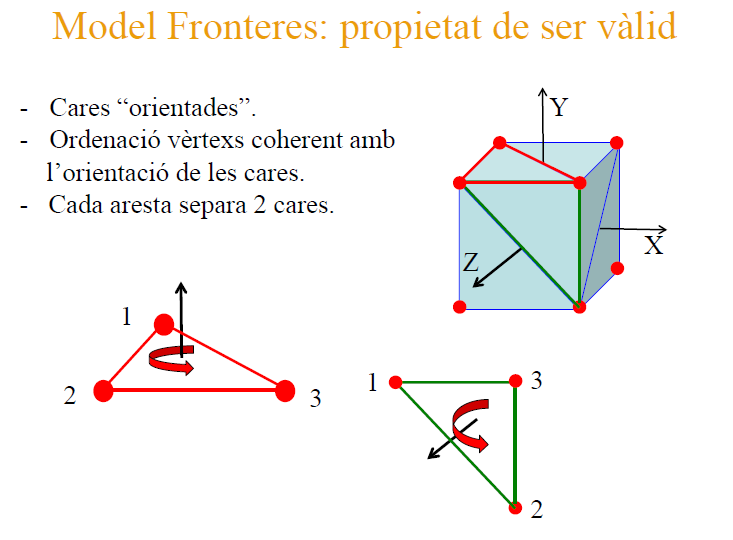
Primer de tot, dividim el cub en un conjunt de triangles. Com que no n’hi ha prou d’obtenir només les coordenades de tots els vèrtex, ja que necessitem saber la connectivitat dels triangles (com estan formats els triangles, informació topològica), per saber quins tres vèrtexs formen el triangle que els correspon, formem una taula tal que cada tres vèrtexs, amb els seus respectius coordenades, en ordre descendent, formen el triangle que els correspon. A part de la informació geomètrica (coordenades), també estem donant la informació topològica, però aquest mètode també comporta la repetició de vèrtexs, ja que un vèrtex pot formar part de diversos triangles i per això pot aparèixer a diversos tripletes de coordenades.



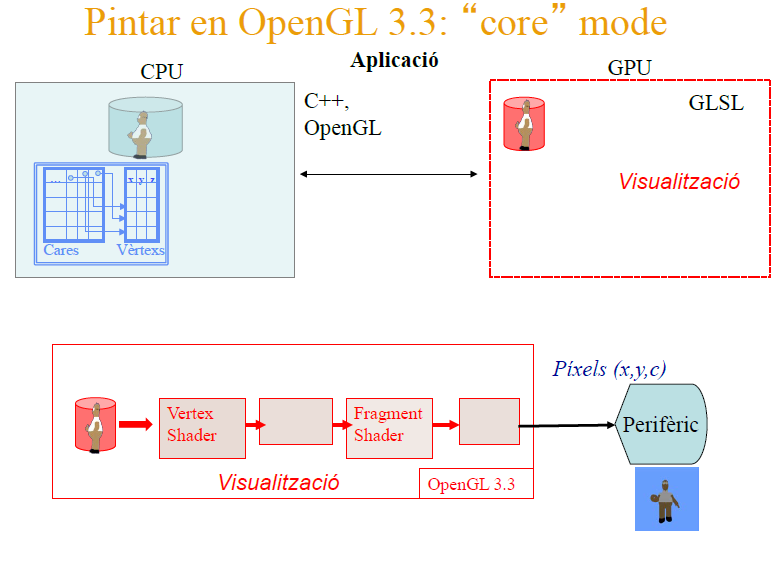
També es pot utilitzar un model que doni la informació topològica de manera explicita, i així estalviar-se que es repeteixin vèrtexs. S’hauria d’afegir una taula de cares, que indiqui la tripleta de coordenades de vèrtexs que formen el triangle.



Per tant, tenim dues maneres d’emmagatzemar la informació topològica, ja sigui de manera implícita o de manera explicita, totes dues maneres son vàlides.

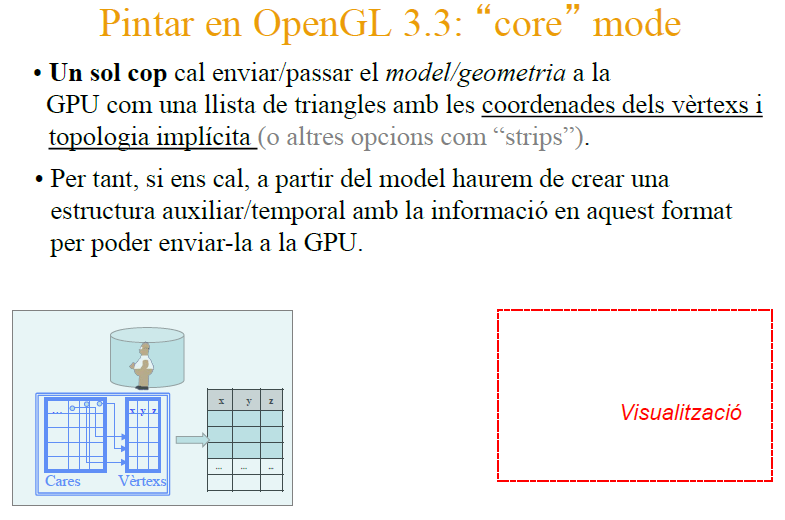


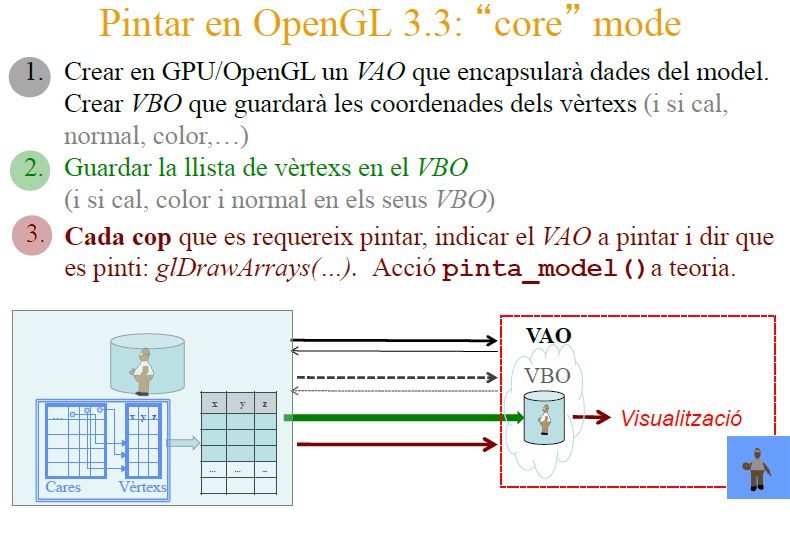
En sentit antihorari vist des de fora de l’objecte amb la normal apuntant cap a fora de l’objecte.



2 unitats de procés: unitat de procés de l’aplicació (CPU, on implementem el codi) i la unitat de procés gràfica (GPU, on es fa la visualització). Memòria GPU es diferent de la memòria de la CPU, per tant, ha d’haver una forma de passar-li informació des de la CPU a la GPU.

Com passar-li aquesta informació?

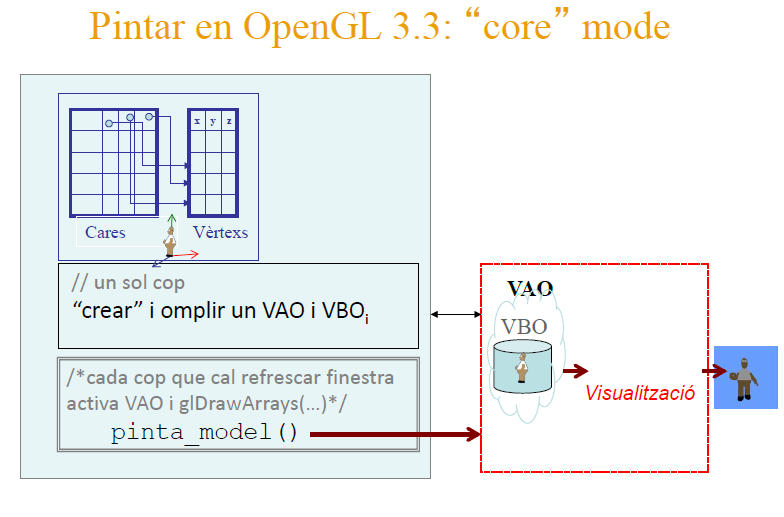




VAO = vertex array object, identificador/encapsulador de l’objecte. Nom de l’objecte i engloba les dades (una o més d’una).

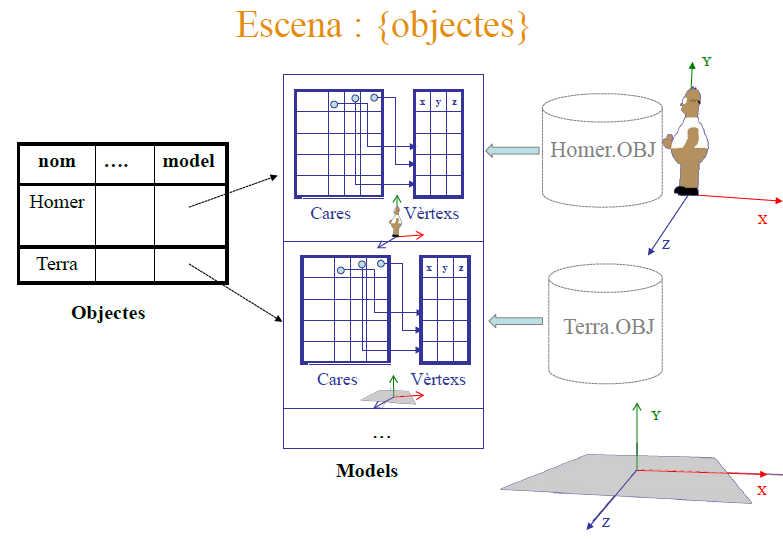
VBO = vertex buffer object, contenen les dades.

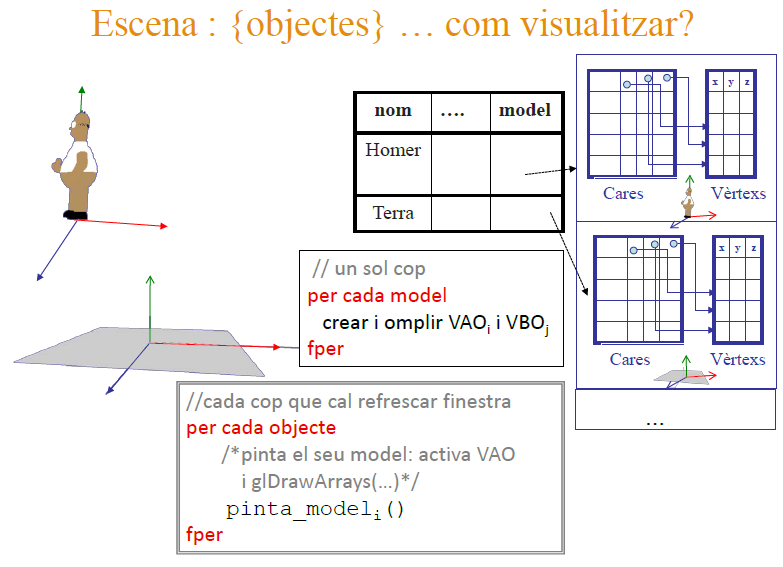
Aquestes dades, que es passen de la CPU a la GPU només es fa un sol cop. Però la GPU pot pintar aquestes dades més d’un cop donant una instrucció de pinta al VAO.

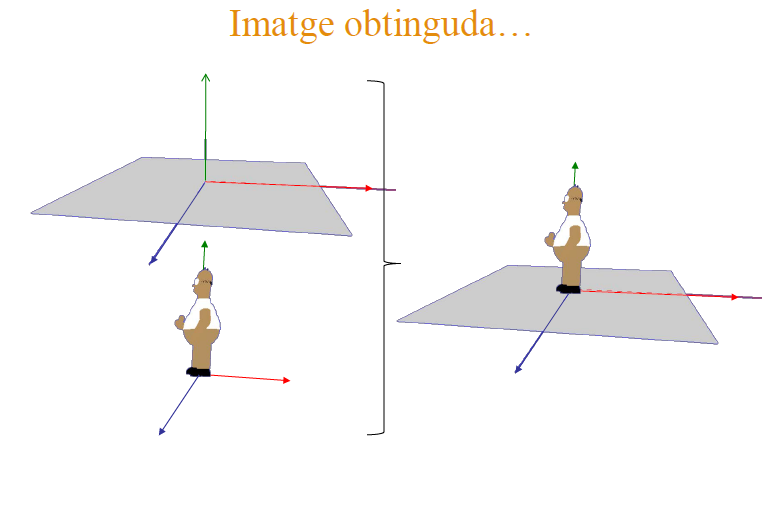


**Models geomètrics per a un conjunt d’objectes**

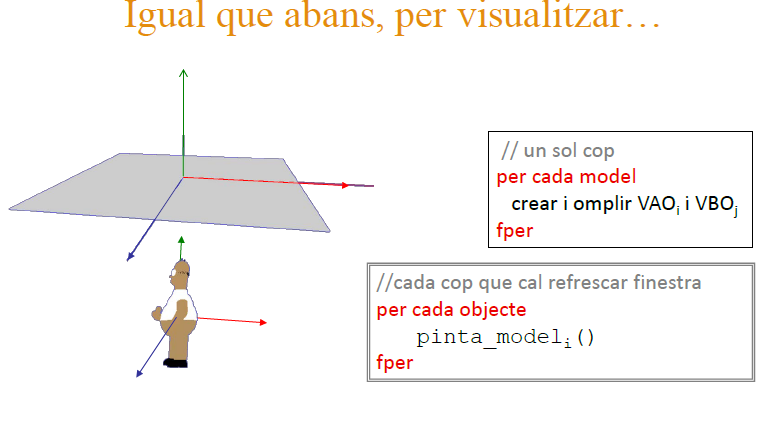
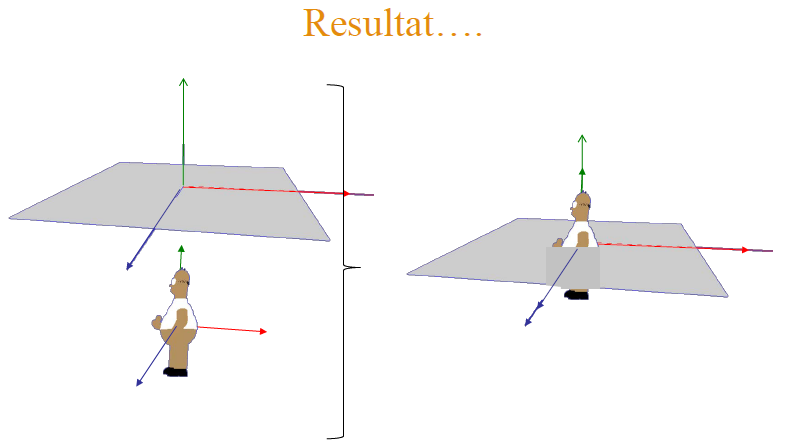
Escena = conjunt d’objectes.



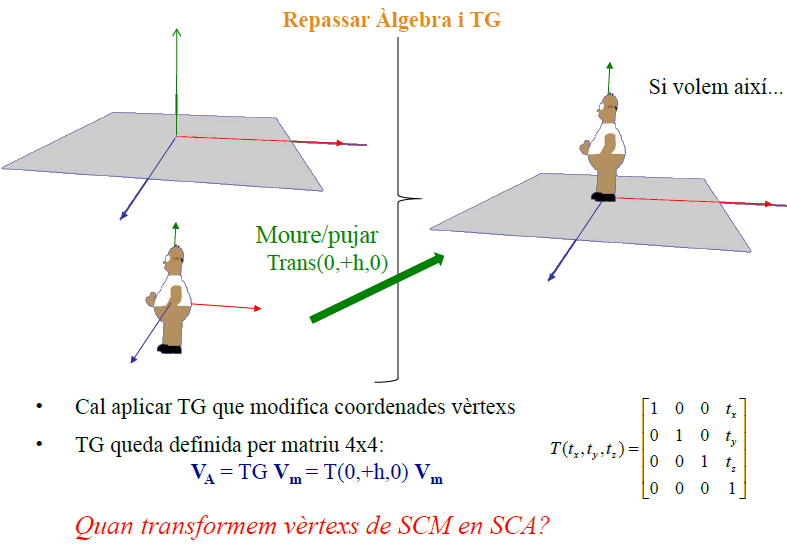
Afegir una taula d’objectes que engloba tots els objectes que tenim.



Surt bé so tenen l’origen de coordenades sincronitzats, però no sempre es així.

Aquí un exemple que no surt bé.

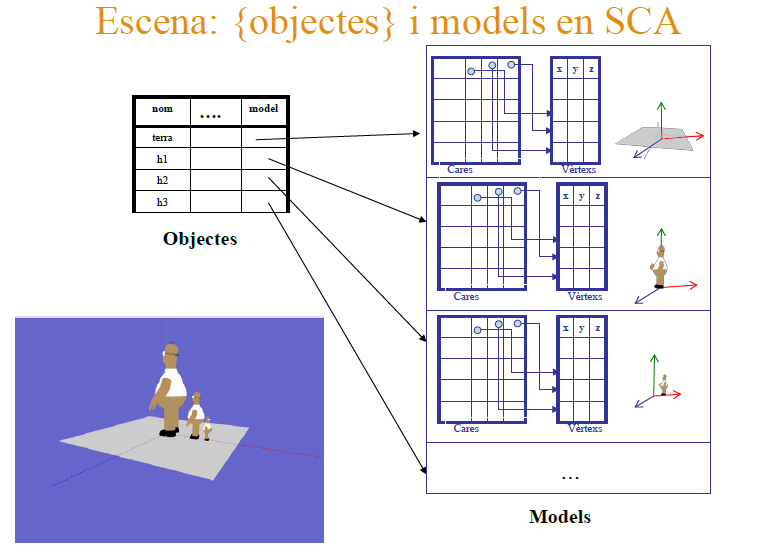
Per solucionar-ho, hem de “pujar” tots els vèrtexs del Homer mitjançant una transformació geomètrica (TG).



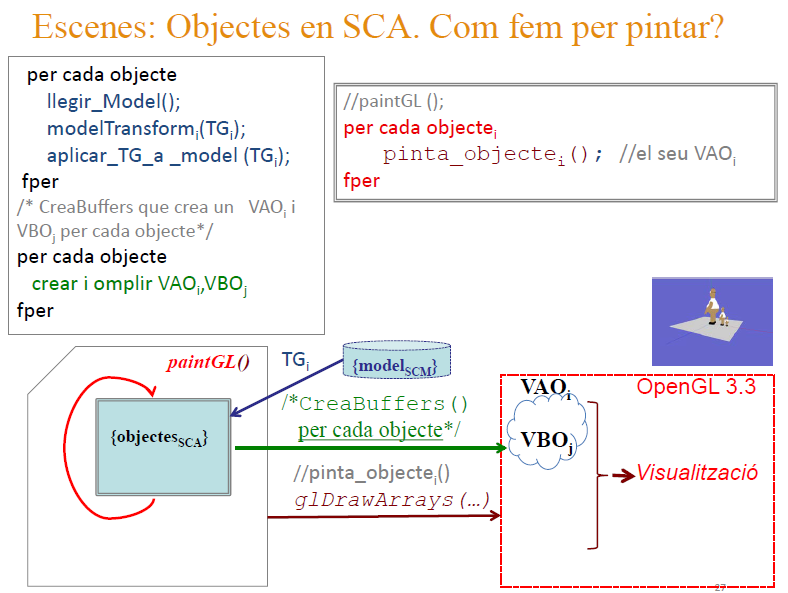
SCM = sistema de coordenades de model (objectes independents).

SCA = sistema de coordenades d’aplicació (escena, els objectes posats conjuntament).

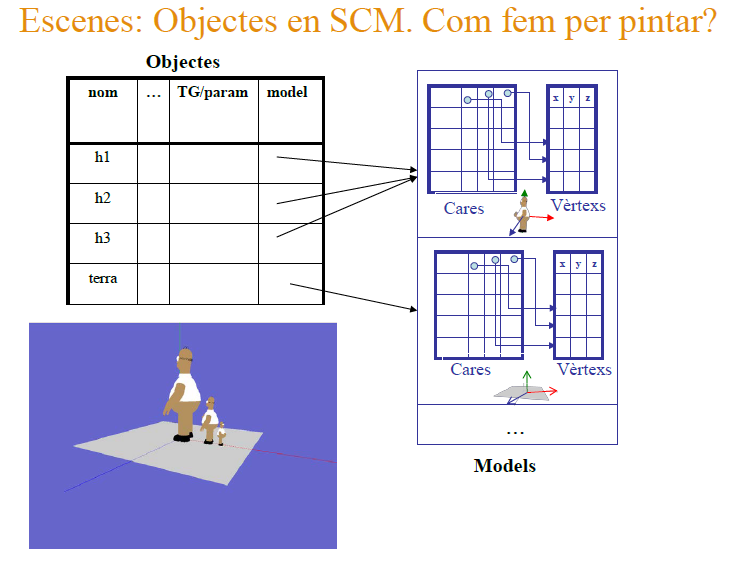
SCA

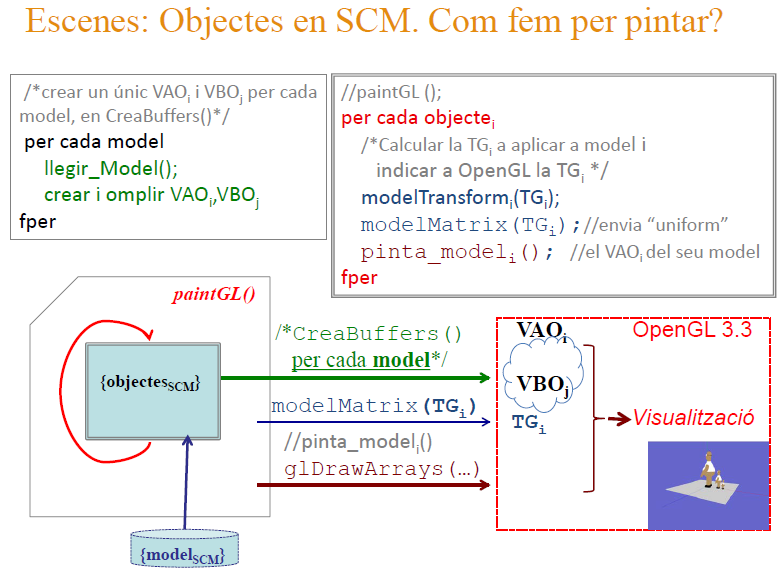


Els objectes guardats directament en SCA (per cada objecte té un model diferent determinat, per x objectes tindríem x models diferents). Es repeteixen models, si hi ha objectes diferents (Homers iguals). Una mica ineficient.



SCM





Guardar els mateixos objectes en un mateix model (els Homers en un mateix model) i la terra en un altre model. (molt més eficient). Però a més li guardo la transformació geomètrica que necessiti per tenir les característiques determinades que vulgui.