

 **Posicionament de la Càmera: opció Moure Objectes**

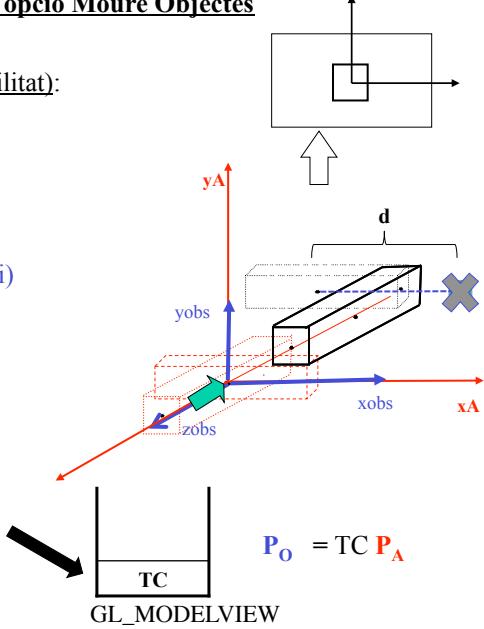
Moviment de l'objecte (una possibilitat):

- 1) Tr(-VRP)
- 2) Gy(-90)
- 3) Tr(0,0,-d)

$TC = T(0,0,-d) G_Y (-90) T(-VRP)$   
(angles positius  $\Rightarrow$  girs anti-horari)

*La podem calcular amb OpenGL*

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glTranslate3f(0,0,-d);
glRotate3f(-90,0,1,0);
glTranslate3f(-VRP.x,-VRP.y,-VRP.z);
```

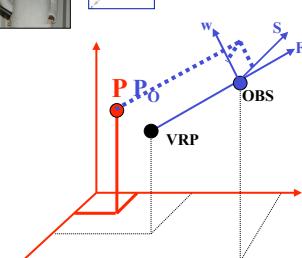


$P_O = TC \cdot P_A$

GL\_MODELVIEW

IDI 2013-2014 2Q 8

 **gluLookAt() → calcul TC**



$$TC = \begin{bmatrix} s.x & s.y & s.z & 0 \\ w.x & w.y & w.z & 0 \\ F.x & F.y & F.z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \text{Trans } (-OBS)$$

$F = OBS - VRP = (F.x, F.y, F.z)$      $F = F / \| F \|$

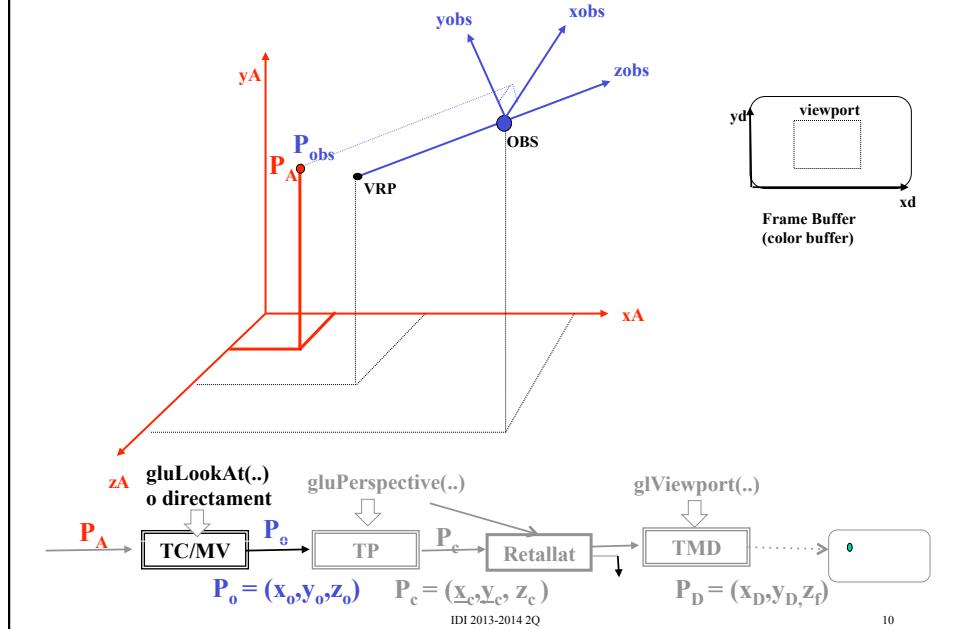
$s = up \times F$      $s = s / \| s \|$

$w = F \times s$

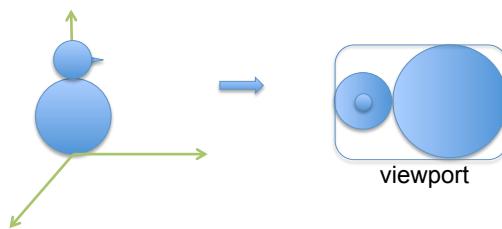
**OBS** = Observador  
**VRP** = View Reference Point  
**up** = View Up Vector  
**up** "indica" la direcció de l'eix vertical de la Càmera (inclinació)

IDI 2013-2014 2Q 9

## Procès Visualització OpenGL (1)



**Exemple 1:** Donada una funció `pinta_ninot()` que pinta l'objecte de la figura, format per: una esfera de radi 10 i centre (0,10,0), altre esfera de radi 5 i centre (0,25,0), i un con de base centrada en (2.5, 25,0), r=2 i llargada 5 orientat segons l'eix X; indicar tots els paràmetres d'una càmera ortogonal que permeti obtenir la imatge que s'indica en un viewport de 600x400.



**Cal definir la posició i orientació de la càmera i l'òptica**

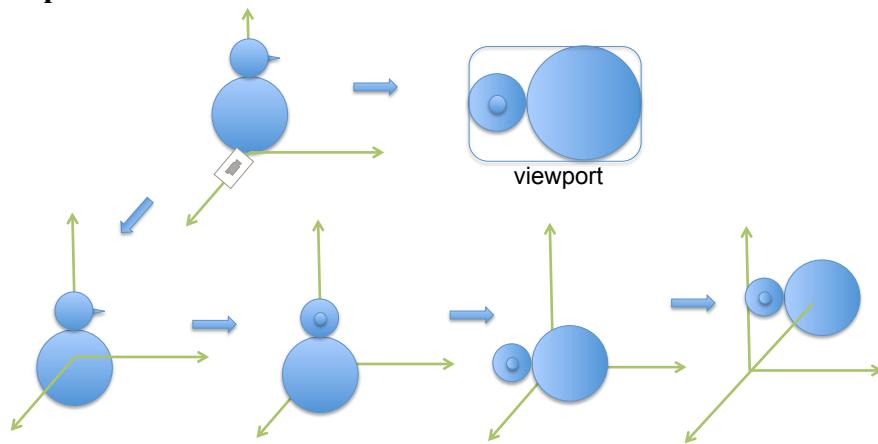
```

Pinta_Ninot()
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    // esfera 1
    glPushMatrix();
    glTranslatef (0.,10.,0.);
    glutSolidSphere(10.,20,20);
    glPopMatrix();
    //esfera 2
    glPushMatrix();
    glTranslatef (0.,25.,0.);
    glutSolidSphere(5.,20,20);
    glPopMatrix();
    //con
    glPushMatrix();
    glTranslatef(2.5,25,0.);
    glRotatef(90.,0.,1.,0.,);
    glut SolidCone(2.,5.,20,20)
    glPopMatrix();

```

IDI 2013-2014 2Q

14

**Exemple 1: Càmera amb TG**

$$TC = T(0,0,-30)G_z(90)G_y(-90)T(0,-15.0)$$

```

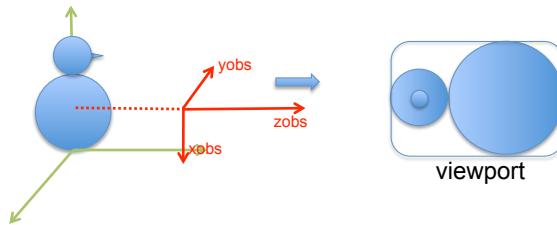
glMatrixMode (GL_MODELVIEW);
glLoadIdentity();
glTranslatef (0.0,-30.0);
glRotatef (90.0,0.1,0.0,1.0);
glRotatef (-90.0,0.1,0.0,1.0);
glTranslatef (0.0,-15.0,0.0);
Pinta_Ninot();

```

IDI 2013-2014 2Q

15

### Exemple 1: Càmera amb gluLookAt



Pensant en càmera i com ha de quedar la imatge

$\text{VRP}=(0,15,0); \text{OBS}=(30,15,0), \text{up}=(0,0,-1)$

IDI 2013-2014 2Q

16

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glTranslatef(0.,0.,-30.)
glRotatef(90.,0.,0.,1.)
glRotatef(-90.,0.,1.,0.)
glTranslatef(0.,-15.,0.)
```

Interpretació alternativa de TG per a determinar OBS, VRP i up

$\text{TC}=\text{T}(0,0,-30)\text{G}_z(90)\text{G}_y(-90)\text{T}(0,-15,0)$

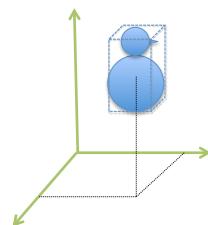
$\text{TG}_c \Rightarrow 1) \text{T}(0,15,0) \quad 2) \text{G}_y(+90) \quad 3) \text{G}_z(-90) \quad 4) \text{T}(0,0,30)$

IDI 2013-2014 2Q

17

### Exercici d'inicialització i moure càmera

- Veure escena sempre sense retallar i sense deformació (en pas a viewport)
- La imatge inicial volem que estigui centrada i ocupant raonablement el viewport i mostri escena vista des d'una posició arbitrària.
- Càmera perspectiva.
- Permetre modificació interactiva de punt de vista.

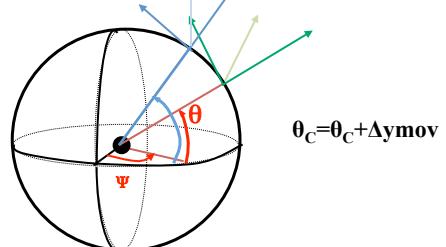
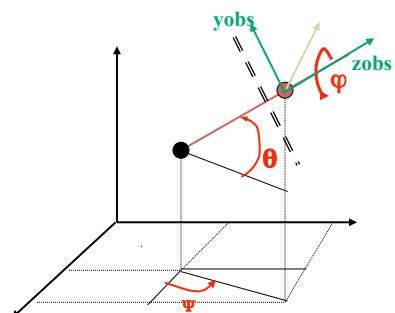
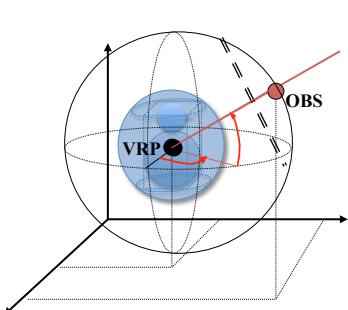


Imaginem aquesta escena

IDI 2013-2014 2Q

18

### Moure la Càmera amb angles d'Euler



- Imaginem que en la interfície d'usuari estem ubicant la Càmera movent el cursor dreta/esquerra ( $\Psi$ ) i pujar/baixar ( $\Theta$ ). És com moure OBS sobre l'esfera i els angles d'Euler determinen un punt en esfera.

- També ho podeu pensar com si girem l'esfera per a què la seva posició respecte la càmera de defecte sigui la mateixa (agafar l'esfera amb la mà i girar-la).
- Codi per OpenGL directe a partir dels angles.

$$\theta_C = \theta_C + \Delta \text{yMov}$$

IDI 2013-2014 2Q

19

### Exercici d'inicialització càmera: Posicionament

**Ull amb signes:**

- Si s'ha calculat  $\psi$  positiu quan càmera gira cap a la dreta, serà un gir anti-horari respecte eix Y de la càmera, per tant, matemàticament positiu; com girem els objectes en sentit contrari, cal posar  $-\psi$  en el codi.
- Si s'ha calculat  $\theta$  positiu quan pugem la càmera, serà un gir horari; per tant, matemàticament un gir negatiu; com objecte girarà en sentit contrari (anti-horari), ja és correcte deixar signe positiu.

IDI 2013-2014 2Q

- Imaginem movem la càmera (OBS) sobre una esfera centrada en VRP de radi d.
- VRP (punt enfoc) i d ( $>$ Radi esfera contenidora).
- Podem definir la posició amb angles Euler:  $\psi$  i  $\theta$ .  
per exemple:  $\psi=45^\circ$  i  $\theta=45^\circ$  (o altres valors)
- gir càmera sobre si mateixa:  $\varphi$   
per exemple:  $0^\circ$  per veure eix Y vertical.
- Podem indicar càmera a OpenGL amb TG a objecte (més directe) o calculant VRP, OBS i up (veure transpa següent). Recordeu que transformacions "en codi" són als objectes.

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glTranslatef(0.,0.,-d)
glRotatef(0,1,0,0)
glRotatef(-psi,0,1,0.)
glTranslatef(-VRP.x,-VRP.y,-VRP.z))
```

20

### Càlcul VRP, OBS a partir angles Euler

**VRP** = Punt d'enfoc  
**OBS** = **VRP** + d **v**  
**d** > R ; per exemple: d = 2R

$$v_y = \sin(\theta); \quad a = \cos(\theta);$$

$$v_z = \cos(\theta) \cos(\Psi);$$

$$v_x = \cos(\theta) \sin(\Psi);$$

Un possible **up**: **up** = (0,1,0) ( $\varphi = 0^\circ$ )

Noteu que "aqui" estem considerant els angles d'orientació de la càmera ==>  $\psi$ ,  $\theta$  positius quan movem la càmera cap a la dreta i quan la pugem.

IDI 2013-2014 2Q

21