

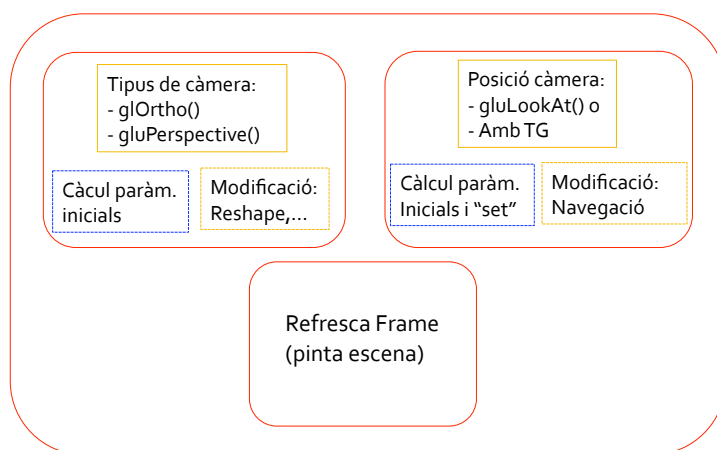
© Professors d'IDI – Curs 2013-2014

Bloc_3: Càmera. Sessió 1

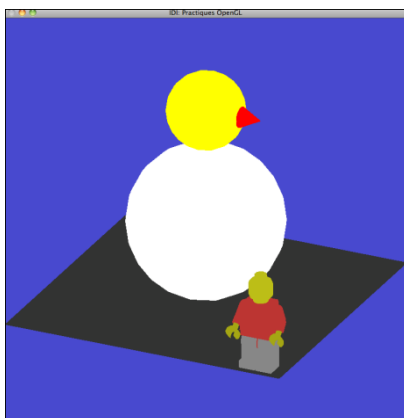
Introducció

- **3 sessions** de laboratori.
- Llegiu el guió d'aquest Bloc.
- Objectiu global:
 - Inicialitzacions de càmera per complir els requeriments d'una aplicació
 - saber utilitzar les funcions d'OpenGL
 - estructurar aplicacions, i practicar tècniques de modificació interactiva de càmera.
- Sessió1: seccions 1 a 5 del guió.
- Sessió 2: seccions 6, 7 i primera funcionalitat de la secció 8.
- Sessió 3: seccions 8 i 9.
- La secció 10 és totalment optativa amb funcionalitats més "avançades" que potser us interesen en algun moment...

Estructura recomanda per l'aplicació

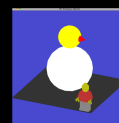


Escena de test per a la sessió 1



- Escena de l'aplicació que heu lliurat del Bloc 2.
- Netejeu el seu codi per a quedar-vos amb les funcionalitats demanades.
- Caldrà anar reorganitzant el codi amb les parts que us hem suggerit.

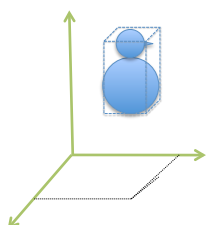
Objectiu de la sessió 1



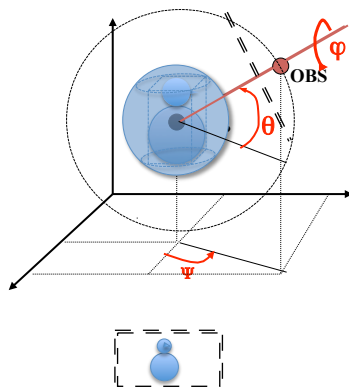
- Visualitzar l'escena:
 - L'escena s'ha de veure centrada i la posició de la càmera s'ha de declarar amb angles d'Euler (secció 3).
 - S'ha de poder inspeccionar l'escena interactivament amb el ratolí modificant els angles d'Euler (secció 3).
 - Sense retallar i sense deformació en un **viewport** que ocuparà sempre **tota la finestra gràfica** (secció 2).
 - L'òptica podrà ser axonomètrica o perspectiva (seleccionable per l'usuari) (seccions 2 i 4).

Objectiu de la sessió 1: repasseu l'exercici vist a classe de teoria

- 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 la càmera.



Recordeu l'exercici d'inicialització càmera: Posicionament



- Imaginem movem la càmera (OBS) sobre una esfera centrada en VRP de radi d .
 - VRP i d calculats com s'ha vist en exercici anterior.
 - Podem definir la posició amb angles Euler: ψ i θ .
per exemple: $\psi = 45^\circ$ i $\theta = 45^\circ$ (o altres valors)
 - gir càmera sobre si mateixa: φ
per exemple: 0° 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 (repasseu exercicis anteriors)

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

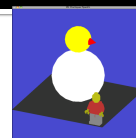
Ull amb signes:

- Si s'ha calculat ψ 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 θ 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.

7

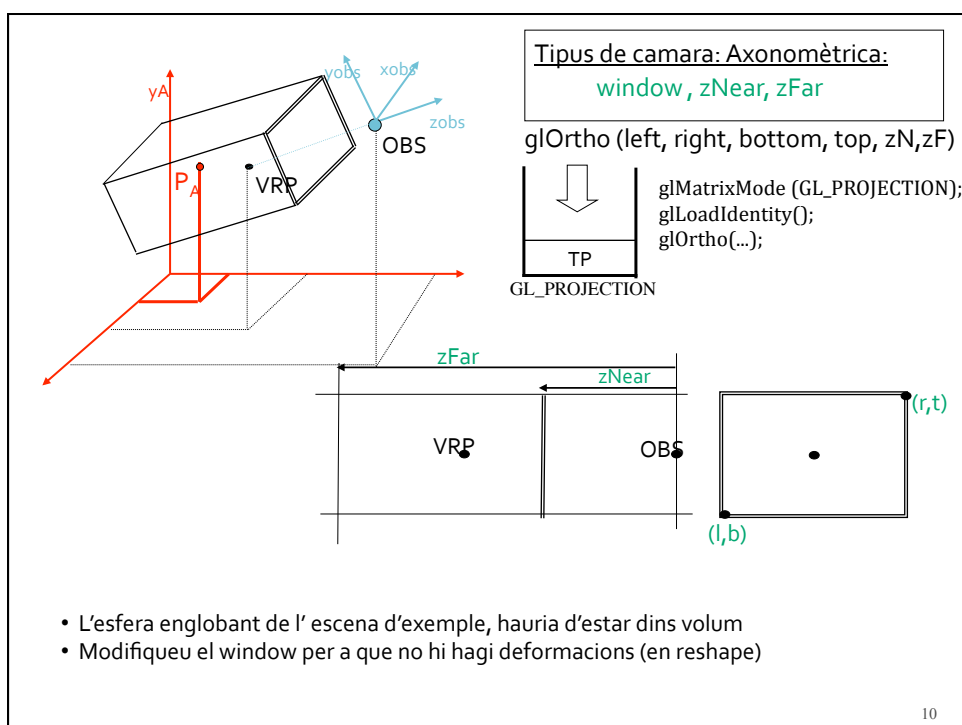
Pocionament amb angles Euler (TG). Secció 3. Pas 1

- Càlcul de paràmetres inicials de la càmera per veure l'escena, sense retallar i des d'una posició arbitrària.
 - Calculeu l'esfera mínima contenidora per a la **VOSTRA** escena.
 - Calculeu la capsula (vèrtexs extrems) i després l'esfera (Centre i Radi).
 - Centre esfera => VRP
 - Distància de *dist* l'observador a VRP: $d > \text{Radi}$ (mireu la nota de sota en vermell)
 - Angles d'Euler inicials: arbitraris (però coneguts per vosaltres ☺).
Recordeu que l'usuari ha de pensar que mou la càmera.
 - Recomanació: feu un mètode específic "ini_càmera".
- Declareu la càmera a OpenGL fora de "refresh".
 - Recordeu que les TG s'apliquen als objectes.
- **Fixeu-vos** que la òptica que teniu definida per defecte (glOrtho) comporta un volum de visió molt restrictiu. Per aquest motiu, la vostra escena es pot veure retallada –o, fins i tot, no es veurà-. **POSEU *dist*=0.2 inicialment.**

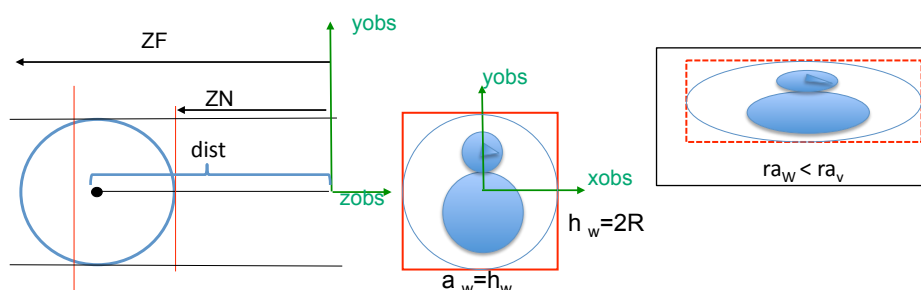


Càmera Axonomètrica. Secció 2

- `glOrtho` (left, right, bottom, top, zN, zF)
- Càlcul de la inicialització dels paràmetres per veure tota l'escena ocupant, raonablement, el volum de visió:
 - Utilitzeu les dades calculades de l'esfera contenidora.
 - Guardeu les dades del “window”, zN i zF en “ini_càmera”.
- Nou Reshape per a no tenir deformacions, maximitzant l'ocupació de la finestra gràfica.
 - Viewport que ocupi tota la finestra gràfica.
 - Modificar *window* per a què la seva ra_w sigui igual a la del ra_v del viewport. Sempre a partir dels seus valors inicials. Proveu què passa si ho feu a partir dels valors actuals. Ho enteneu?



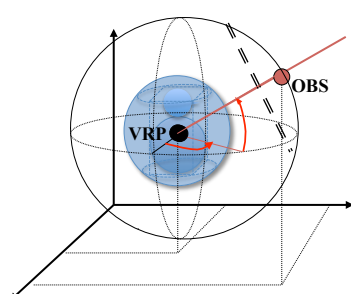
Tota escena en la vista, sense deformar i càmera axon mètrica



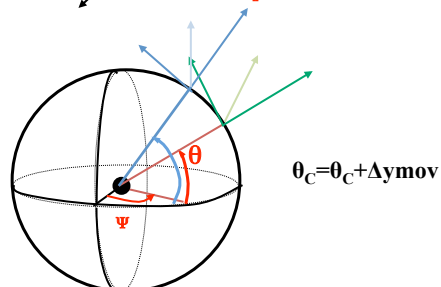
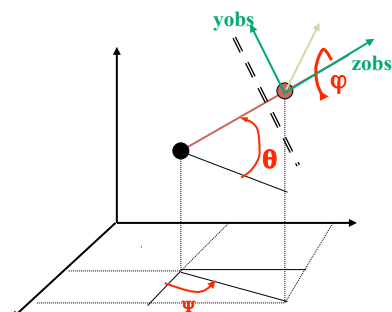
- Suposem coneguts: radi esfera R i distància $dist$ entre OBS i VRP.
- ZN i ZF ajustats a l'esfera. $ZN \leq dist - R$; $ZF \geq dist + R$
- Window mínim requerit (centrat) = $(-R, -R, R, R) \Rightarrow$ una $ra_w = 1$ (per què?)
- Si $ra_w \neq ra_v \Rightarrow$ deformació (per què?)
 - Si $ra_v > 1 \Rightarrow$ cal incrementar la $ra_w \Rightarrow$ modificar window com $ra_w = a_w/h_w \Rightarrow$ podem incrementar a_w o decrementar h_w (és retallaria esfera/escena!!)
 - Per tant: $a_w^* = ra_v * h_w = ra_v * 2R \Rightarrow inc_a = a_w^* - a_w$
 window = $(- (R + inc_a/2), R + inc_a/2, -R, R) = (-R \cdot ra_v, R \cdot ra_v, -R, R)$
 - raonament similar per recalculer window quan $ra_v < 1$

Pocionament amb angles Euler (TG). Secció 3. Pas 2

- Ara que tenim òptica, cal posar la “*dist*” que correspon a l'especificació del problema (observador sempre fora esfera).
- Experimenteu amb diferents angles i distàncies.
- **Comproveu** que, independentment del reshape, l'escena queda sempre centrada en viewport i sense deformacions.
- Modifiqueu interactivament els angles d'Euler amb el ratolí.
 - Poseu valors màxims als dos angles d'Euler: Ψ en $[0...360^\circ]$, θ en $[-90...90]$

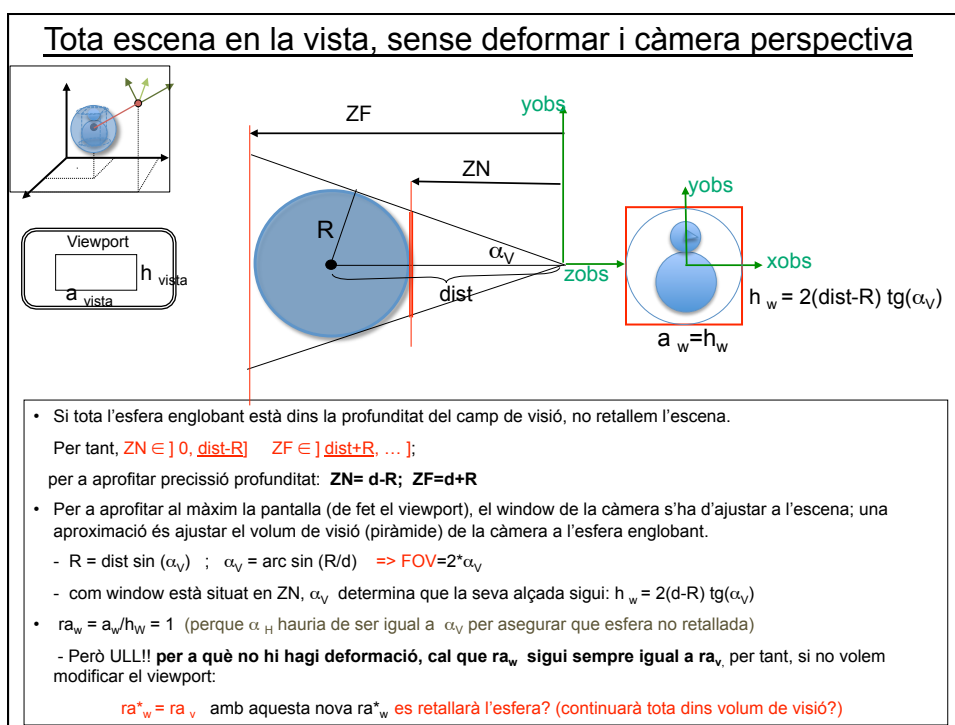
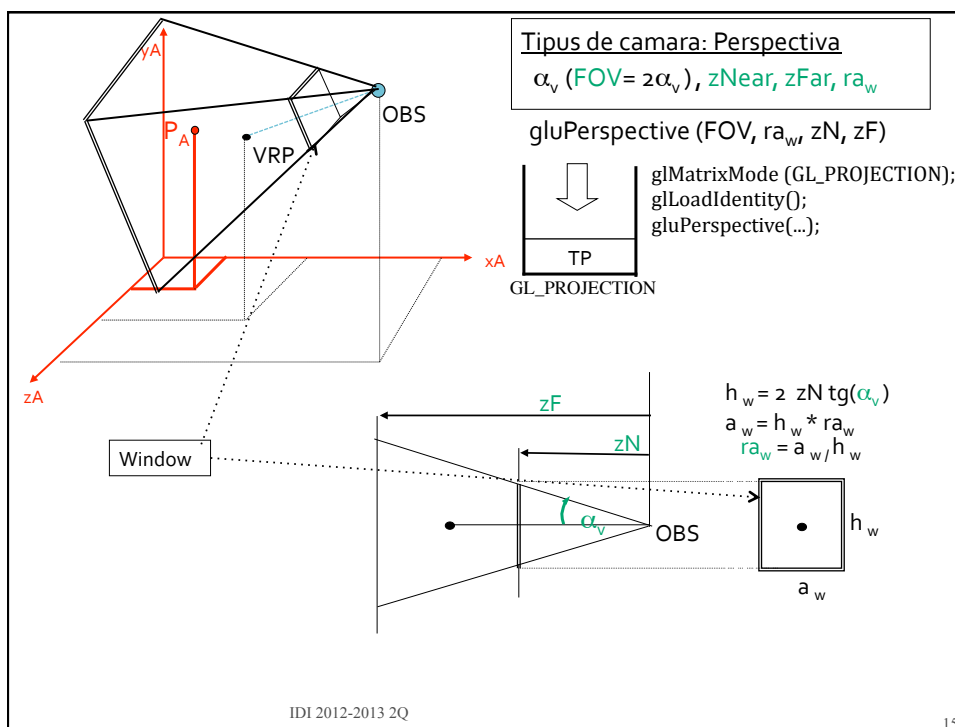
Recordeu l'exercici: Càmera amb angles d'Euler

- Imaginem que en la interfície d'usuari estem ubicant la Càmera movent el cursor dreta/esquerra (Ψ) i pujar/baixar (θ). 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 ma i girar-la).
- Codi per OpenGL directe a partir dels angles.



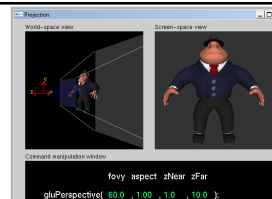
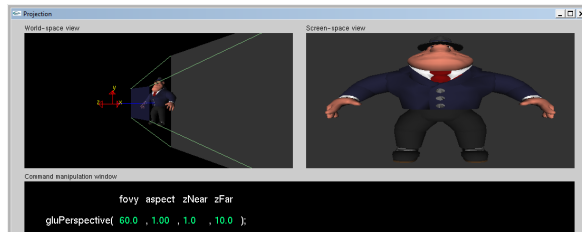
Càmera Perspectiva. Secció 3

- gluPerspective (FOV, r_{aw} , zN , zF)
- Càlcul de la inicialització dels paràmetres que permeti veure tota l'escena: sense retallar i maximitzant ocupació del volum de visió:
 - Feu els càlculs pensant en l'esfera englobant.
 - Calculeu i guardeu valors inicials de paràmetres.
- Nou Reshape per a no tenir deformacions, ocupant raonablement la finestra gràfica.
 - Viewport que ocupi tota la finestra gràfica
 - **Feu sempre els càlculs a partir del valor inicial calculat de FOV.**
- Tecles per modificar el tipus de càmera actiu: "x" axonomètric, "p" perspectiva.
- Afegiu include `GL/glu.h` i en make la llibreria -IGLU.
- Analitzeu les diferències "visuals" amb la càmera axonomètrica.

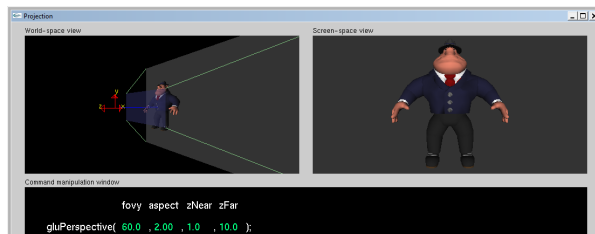


Resize incrementant ra_v

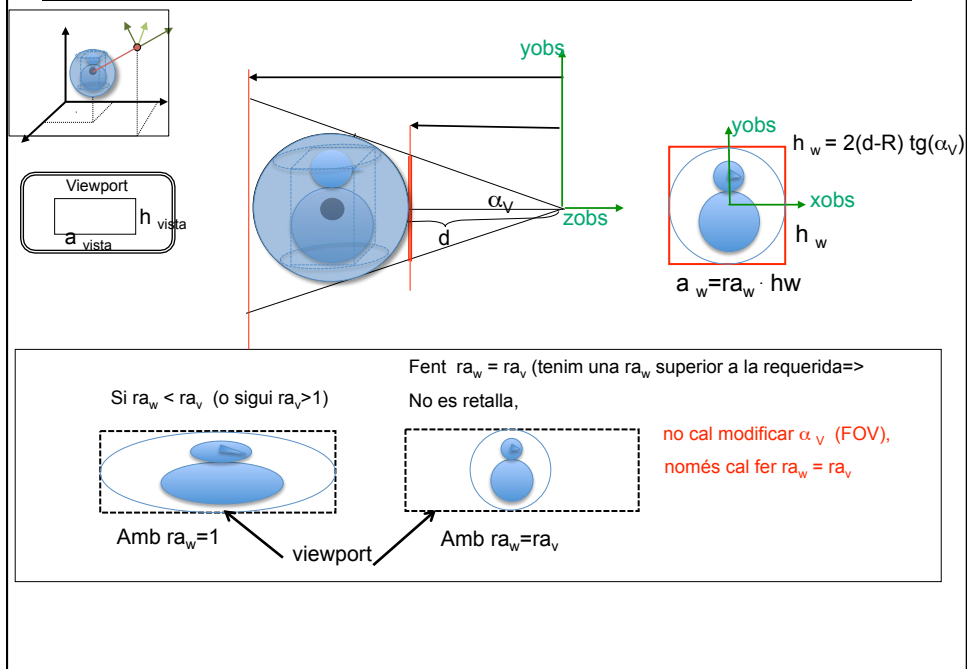
- $ra_w \neq ra_v$



- Posant $ra_w = ra_v$ **Arreglat!**



Tota escena en la vista, sense deformar i càmera perspectiva



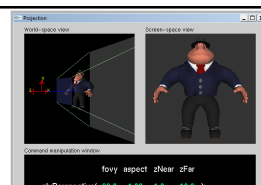
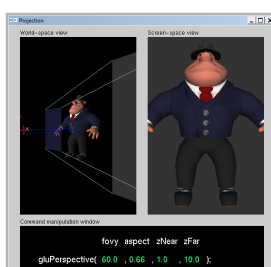
Resize Decrementant ra_v

- $ra_v \neq ra_w$

- Posant $ra_w = ra_v$

No deformació.

Retallat!



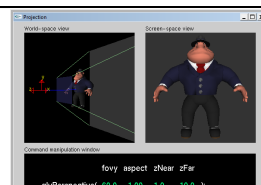
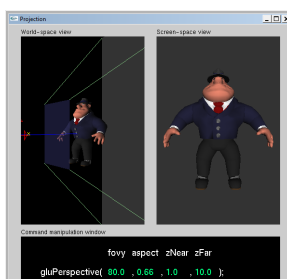
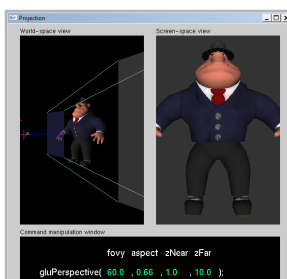
Resize Decrementant ra_v

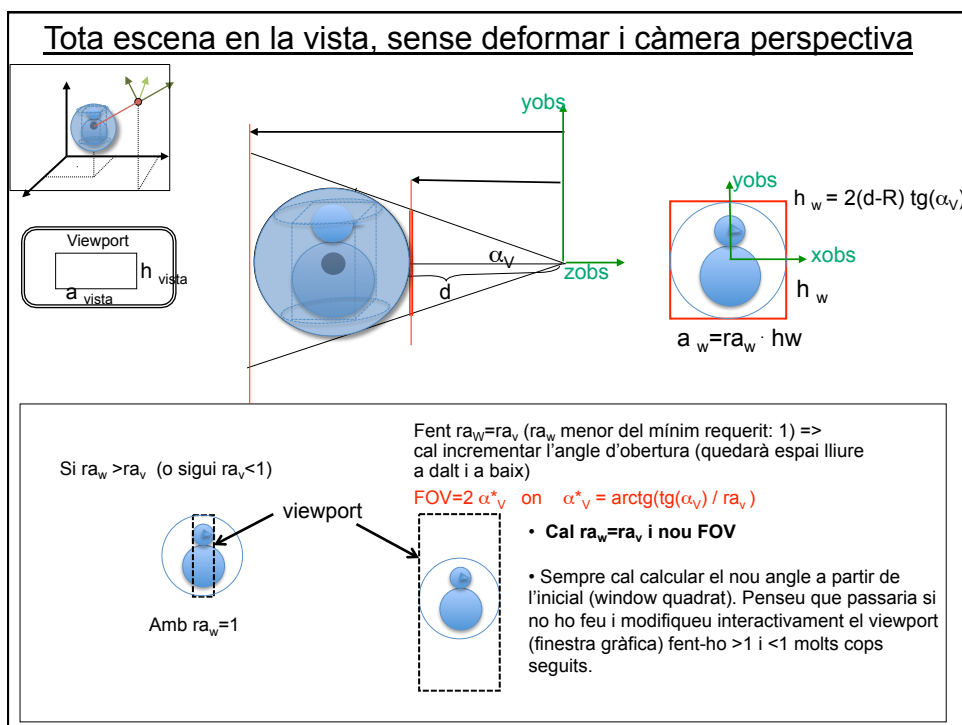
- $ra_w = ra_v$

Retallat!

- Canviant fovy

Arreglat!





© Professors d'IDI – Curs 2013-2014

Bloc_3: Càmera. Sessió 1