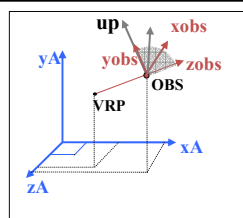


Classe 4: Contingut

- **Breu repàs càmera**
- **Zoom**
- Primers exercicis d'especificació de càmera
- Breu repàs del Procés de Visualització d'OpenGL i ubicar models en escena en el Vèrtex Shader
- Càmera en tercera persona

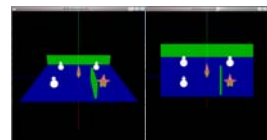
IDI Q2 2019-2020



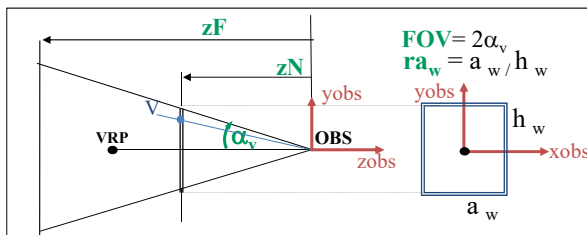
OBS = Observador
VRP = View Reference Point
up = View Up Vector
up “indica” la direcció de l'eix vertical de la Càmera (inclinació)

$F = \text{OBS} - \text{VRP} = (F.x, F.y, F.z)$
 $s = \text{up} \times F$
 $w = F \times s$

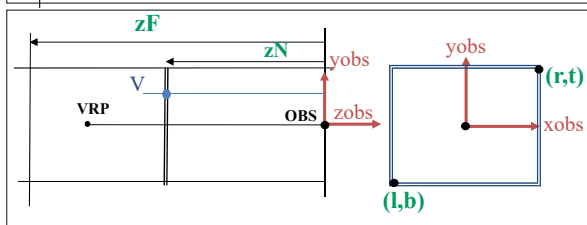
```
VM=lookAt (OBS,VRP,up);
viewMatrix (VM);
```



```
PM=perspective (FOV,ra,
                zN,ZF);
projectMatrix (PM);
```

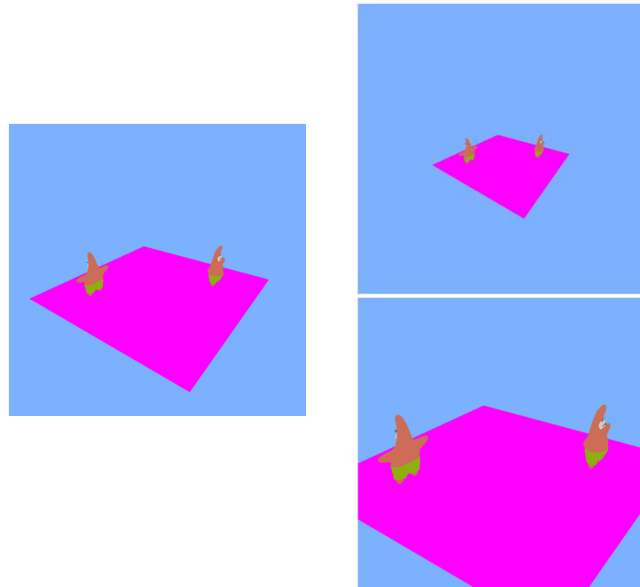


```
PM=ortho (l,r,b,t,zN,ZF);
projectMatrix (PM);
```



IDI Q2 2019-2020

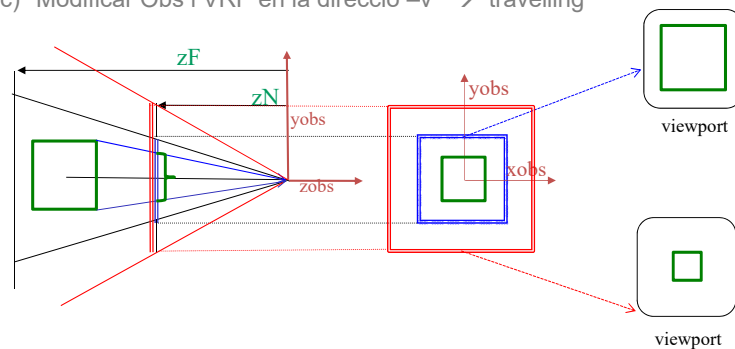
ZOOM



IDI Q2 2019-2020

L'òptica i el ZOOM

- Modificar el window de la càmera:
 - L'angle d'obertura (tot mantenint la ra)
 - Modificar window en ortogonal (tot mantenint la ra)
- Modificar la distància de l'OBS al VRP (modificant ZN i ZF adequadament)
- Modificar Obs i VRP en la direcció $-v \rightarrow$ travelling



IDI Q2 2019-2020

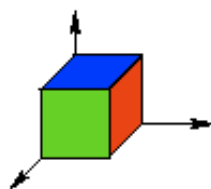
Classe 4: Contingut

- Breu repàs càmera
- Zoom
- **Primers exercicis d'especificació de càmera**
- Breu repàs del Procés de Visualització d'OpenGL i ubicar models en escena en el Vèrtex Shader
- Càmera en tercera persona

IDI Q2 2019-2020

Exemple 2. Tenim una escena amb un cub de costat 2 orientat amb els eixos i de manera que el seu vèrtex mínim està situat a l'origen de coordenades. La cara del cub que queda sobre el pla $x=2$ és de color vermell, la cara que queda sobre el pla $z=2$ és de color verd i la resta de cares són blaves.

a) Indica TOTS els paràmetres **d'una càmera perspectiva** que permeti veure completes a la vista només les cares vermella i verda. La relació d'aspecte del viewport (vista) és 2. Fes un dibuix indicant la imatge final que s'obtingria.



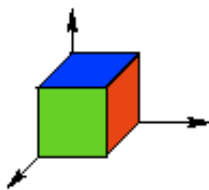
VRP= (2,1,2)
OBS= (3,1,3)
Up = (0,1,0)

$0 < Z_{near} \leq \sqrt{2}$
 $Z_{far} > 3 \cdot \sqrt{2}$
 $FOV = 2 \cdot \arctg(1/\sqrt{2})$
 $ra=2$

IDI Q2 2019-2020

Exemple 2.bis. Tenim una escena amb un cub de costat 2 orientat amb els eixos i de manera que el seu vèrtex mínim està situat a l'origen de coordenades. La cara del cub que queda sobre el pla $x=2$ és de color vermell, la cara que queda sobre el pla $z=2$ és de color verd i la resta de cares són blaves.

- Quin efecte tindria en la imatge final modificar a una òptica ortogonal?
- Defineix l'òptica d'una càmera ortogonal.

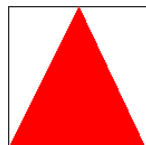


VRP= (2,1,2)
OBS= (3,1,3)
 $U_p = (0,1,0)$

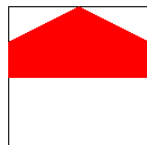
IDI Q2 2019-2020

Exemple 3

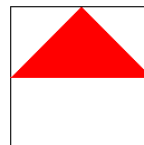
Tenim una escena amb un triangle vermell amb vèrtexs $V1=(-2,0,0)$, $V2 = (2, 0, 0)$ i $V3=(0, 1, 0)$. Suposant que tenim un viewport quadrat de 600x600 píxels, i que hem inicialitzat les matrius de càmera (view) i projecció (proj) a la matriu identitat, indica quina de les següents imatges és la que sortirà en un viewport de 600x600 (sabem que el Vertex Shader i el Fragment Shader estan correctament implementats):



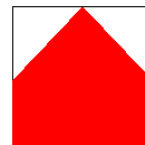
a)



b)



c)



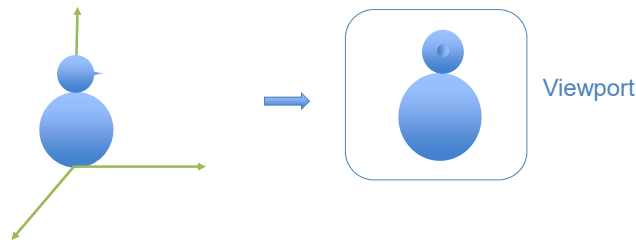
d)

IDI Q2 2019-2020

Recordem...

Exemple 1: Donada una funció `pinta_ninot()` que envia a visualitzar el VAO d'un objecte com el de la figura que està format per: una esfera de radi 10 amb centre (0,10,0), una altra esfera de radi 5 amb centre (0,25,0), i un con de base centrada en (2.5, 25,0), radi 2 i llargada 5 orientat segons l'eix X+

Indica tots els paràmetres d'una càmera perspectiva que permeti obtenir una imatge amb *escena centrada, optimitzant viewport i vista similar a l'esquema de la figura*. El viewport de 600x600 ocupa tota la finestra gràfica.



VRP=(0,15,0); OBS=(30,15,0), up=(0,1,0)

$\alpha = \arctg(15/20) \rightarrow \alpha = 36,8^\circ \rightarrow FOV=2*\alpha$

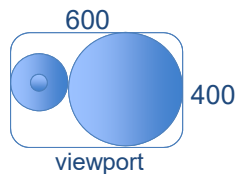
raw = 1 (per no deformar)

ZN=20, ZF=40

IDI Q2 2019-2020

Per pensar...

- Quins paràmetres requiria una òptica ortogonal? Quin efecte tindria en la imatge generada?
- Quins **paràmetres de posicionament** de la càmera per a obtenir (amb òptica ortogonal):



- Com fer un zoom? Quins paràmetres de la càmera modificaries?

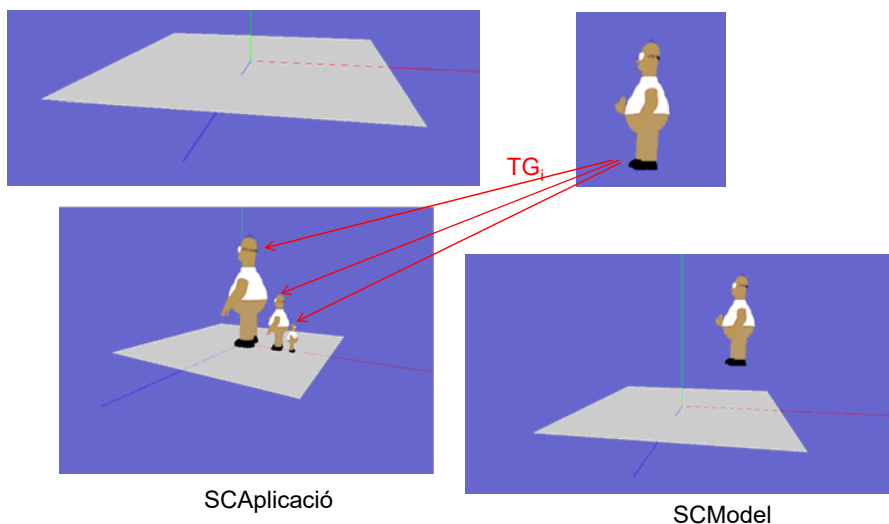
IDI Q2 2019-2020

Classe 4: Contingut

- Breu repàs càmera
- Zoom
- Primers exercicis d'especificació de càmera
- **Breu repàs del Procés de Visualització d'OpenGL i ubicar models en escena en el Vèrtex Shader**
- Càmera en tercera persona

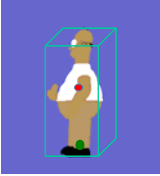
IDI Q2 2019-2020

Recordem: Objectes en coordenades de model i ubicació en escena



$$V_A = TG_i * V_m$$

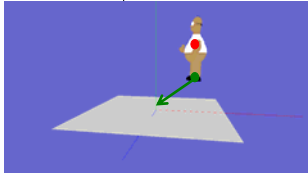
IDI Q2 2019-2020



$CapsaMinCont = (xmin, ymin, zmin, xmax, ymax, zmax)$

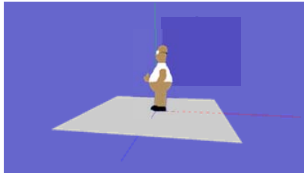
Mides => $a = (xmax - xmin)$, $h = (ymax - ymin)$, $f = (zmax - zmin)$

$CentBaseCapsa = (cbx, cby, cbz) = (xmin + xmax)/2, ymin, (zmin + zmax)/2)$



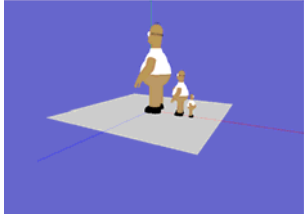
$TG_{H1} = Trans(t)$
 $t = (-cbx, -cby, -cbz)$

→



$TG_{H2} = Trans(3a/4, 0, 0) S(1/2, 1/2, 1/2) Trans(t)$

$TG_{H3} = Trans(9a/8, 0, 0) S(1/4, 1/4, 1/4) R_y(-180) Trans(t)$



IDI Q2 2019-2020

Recordem: com visualitzar l'escena?

```
per cada model
    llegir_Model();
    crear_buffer_model();
fper
```

```
//paintGL ();
per cada objectei
    modelTransformi(TGi); //calcular TG
    modelMatrix(TGi); //envia "uniform"
    pinta_modeli(); //visualitza model
fper
```

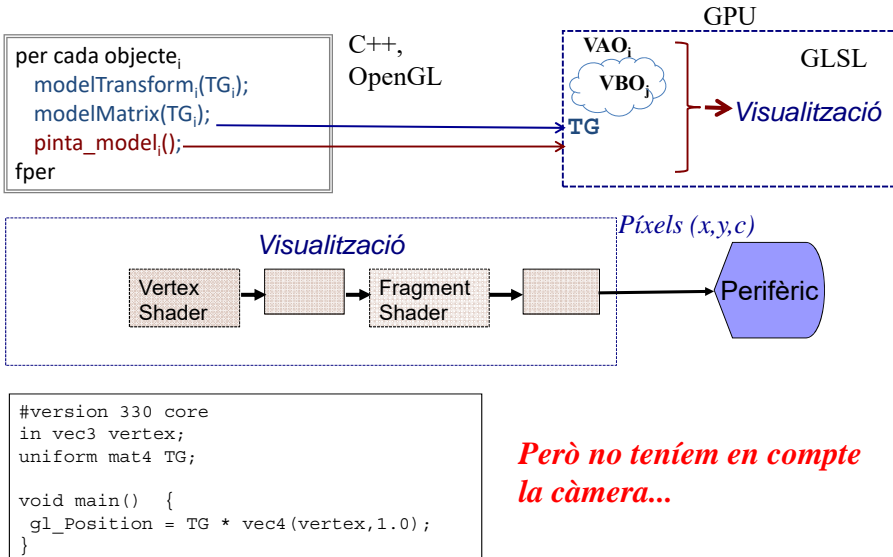
$TG_{H3} = Trans(9a/8, 0, 0) S(1/4, 1/4, 1/4) R_y(-180^\circ) Trans(t)$

4
3
2
1
→

```
modelTransformHomer3()
//tercer homer
{
    TG=I;
    TG= TG*Translate(posx, posy, posz);
    TG= TG*Scale(s, s, s);
    TG= TG*Rotate(-180, (0, 1, 0));
    TG= TG*Translate(-cb.x, -cb.y, -cb.z);
    modelMatrix(TG); //enviar uniform
}
```

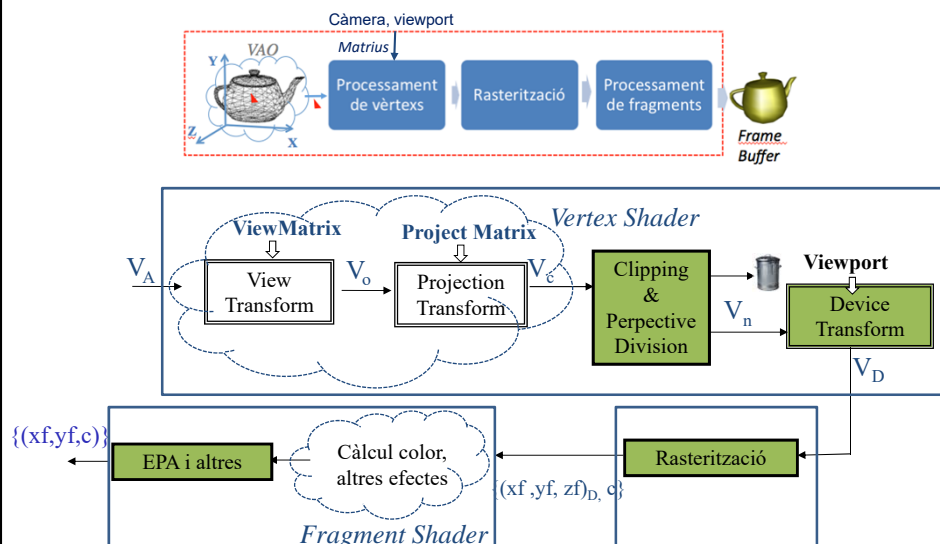
IDI Q2 2019-2020

Recordem: com visualitzar l'escena?



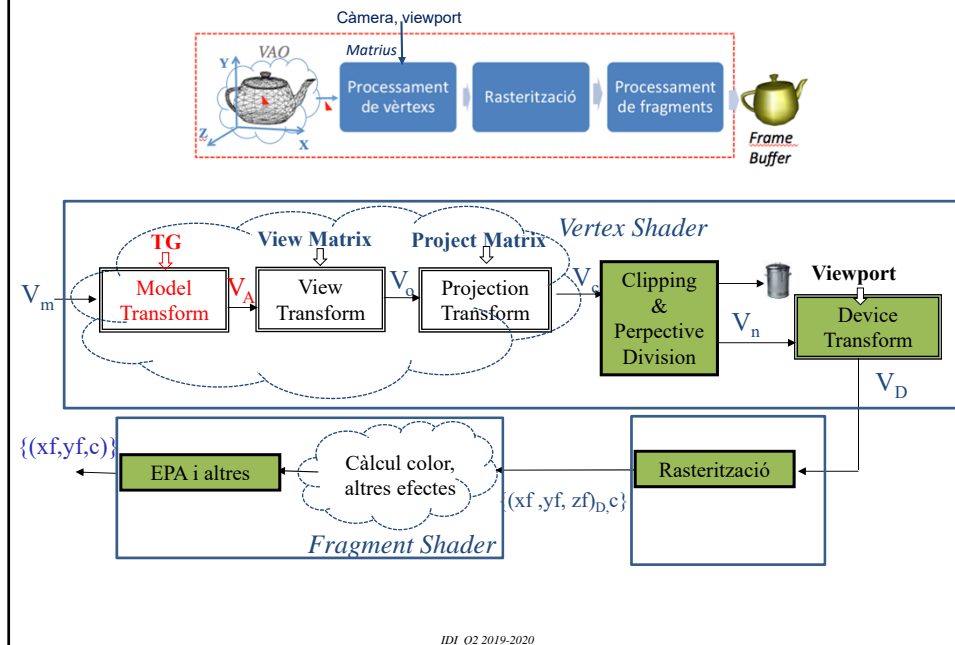
IDI Q2 2019-2020

Pintar/visualitzar en OpenGL 3.3



IDI Q2 2019-2020

Pintar/visualitzar en OpenGL 3.3: VAOs en coordenades de model



Programació mínima dels shaders

```
#version 330 core
```

```
in vec3 vertex;
uniform mat4 PM;
uniform mat4 VM;
uniform mat4 TG;
```

```
void main() {
    gl_Position = PM*VM*TG*vec4 (vertex, 1.0);
}
```

Vertex Shader

Fragment Shader

```
#version 330 core
```

```
out vec4 FragColor;
```

```
void main() {
    FragColor = vec4(0, 0, 0, 1);
}
```

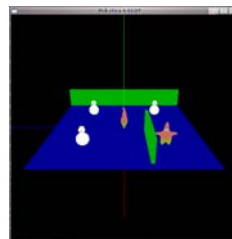
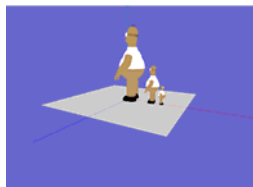
IDI Q2 2019-2020

Classe 4: Contingut

- Breu repàs càmera i exercicis d'especificació de càmera
- Zoom
- Breu repàs del Procés de Visualització d'OpenGL i ubicar models en escena en el Vèrtex Shader
- **Càmera en tercera persona**

IDI Q2 2019-2020

Càmera en 3ra persona

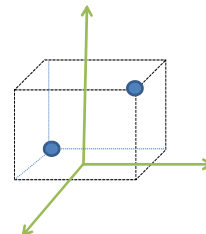


Visualització inicial de l'escena tal que:

- inclogui tota l'escena (no retalli cap objecte)
- posició arbitrària de l'observador
- centrada en viewport
- optimitzant ocupació del viewport/vista
- sense deformació

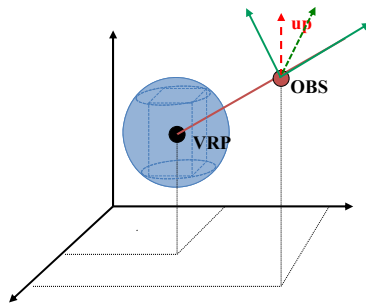
Dada: capsa mínima contenidora de l'escena

$cmin=(xmin, ymin, zmin)$ i $cmax=(xmax, ymax, zmax)$



IDI Q2 2019-2020

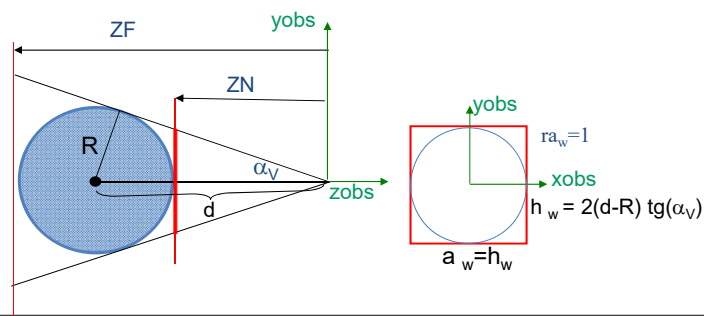
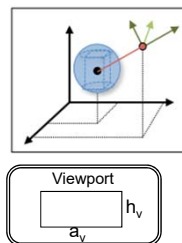
Càmera tercera persona (1): Inicialització posicionament amb OBS, VRP, up



- Centrat => **VRP=CentreEscena**
- Per assegurar que l'escena es veu sense retallar des d'una posició arbitrària CAL que **OBS** sempre fora capsa mínima contenidora; per assegurar-ho CAL que **OBS** fora de l'esfera englobant de la capsa => distància "d" de l'**OBS** a **VRP** superior a R esfera.
 - CapsaMinCont=(xmin,ymin,zmin,xmax,ymax,zmax)
 - CentreEscena=Centre(CapsaMinCont) =
 $((xmax+xmin)/2, (ymax+ymin)/2, (zmax+zmin)/2)$
 - $R = \text{dist}((xmin,ymin,zmin), (xmax,ymax,zmax))/2$
 - $d > R$; per exemple $d=2R$
 - **OBS=VRP+ d*v**; **v** normalitzat en qualsevol direcció;
per exemple $\mathbf{v} = (1,1,1) / \|(1,1,1)\|$
- **up** qualsevol que no sigui paral·lel a **v**; si volem escena vertical (eix Y es vegi vertical) **up**=(0,1,0)

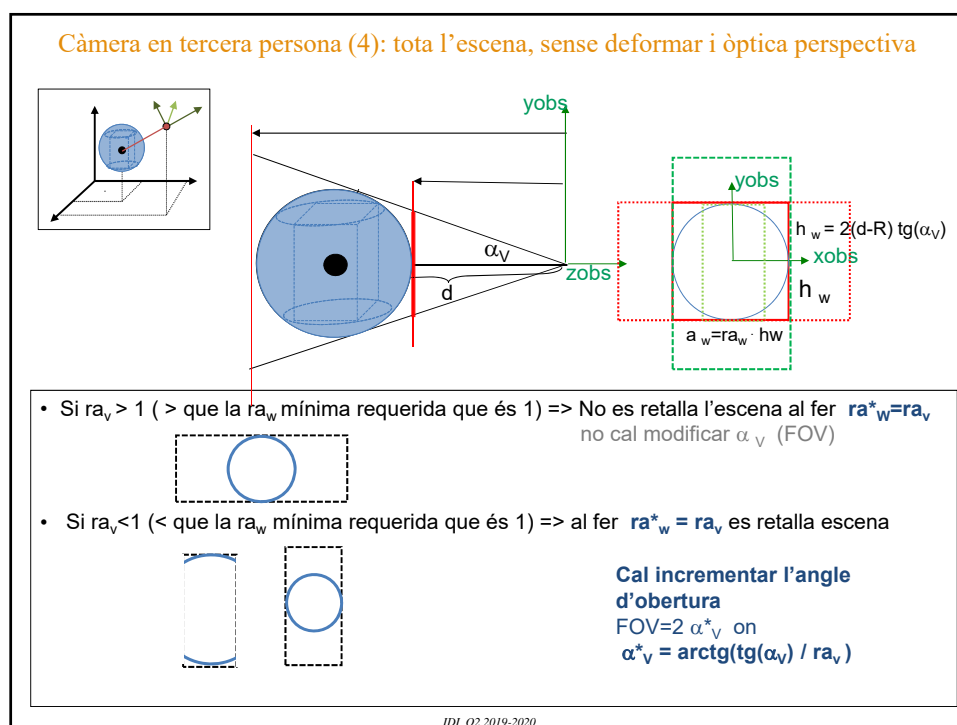
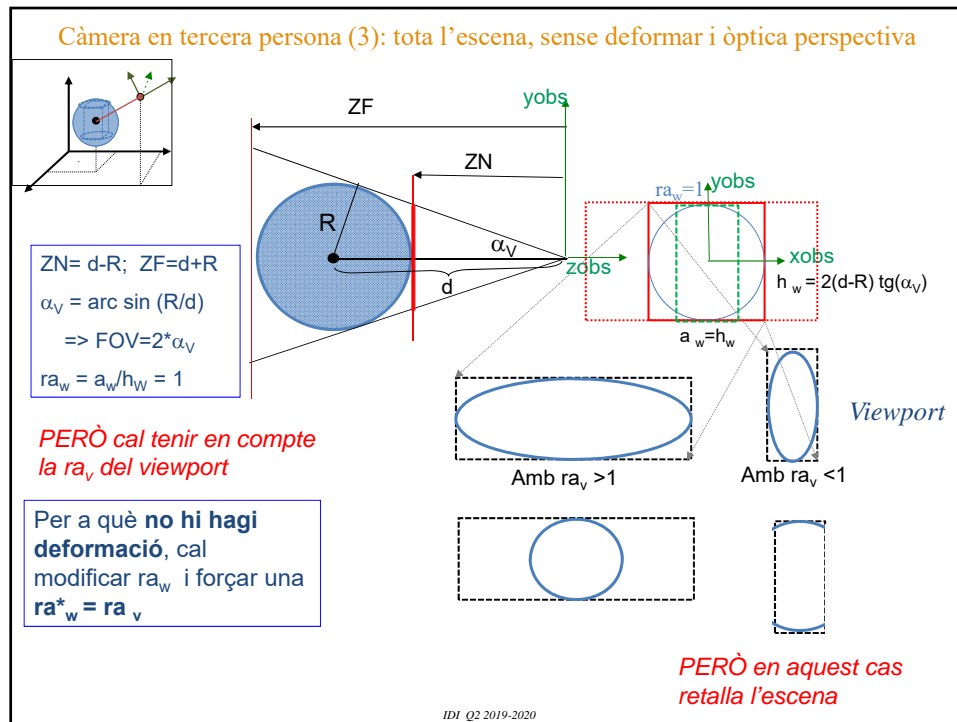
IDI Q2 2019-2020

Càmera en tercera persona (2): tota l'escena, sense deformar i òptica perspectiva

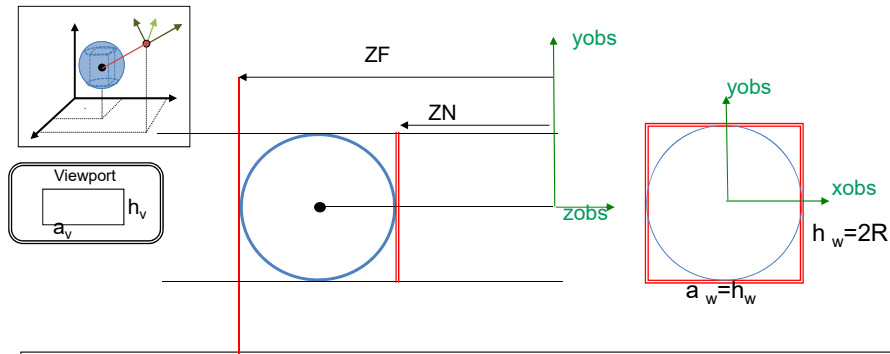


- Si tota l'esfera englobant està dins la profunditat del camp de visió, no retallem l'escena.
 Per tant, $ZN \in]0, d-R]$ $ZF \in [d+R, \dots]$;
 $ZN = d-R$; $ZF = d+R$ per aprofitar la precisió en profunditat
- Per a aprofitar al màxim la pantalla, el viewport, el window de la càmera s'ha d'ajustar per veure tota l'escena; una aproximació és ajustar el window per veure l'esfera englobant.
 - $R = d \sin(\alpha_v)$; $\alpha_v = \arcsin(R/d)$ $\rightarrow FOV = 2 * \alpha_v$
 - com el window està situat en ZN , α_v determina que la seva alçada sigui:
 $h_w = 2(d-R) \tan(\alpha_v)$
- $ra_w = a_w/h_w = 1$ (α_H hauria de ser igual a α_v per assegurar que esfera no retallada)

IDI Q2 2019-2020



Càmera en tercera persona (5): tota l'escena, sense deformar i òptica ortogonal



- **ZN i ZF** mateix raonament que en càmera perspectiva.
- **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!!)
 Per tant, modifiquem a_w :
 $a_w^* = ra_v * h_w = ra_v * 2 * R$
 $window = (-R * ra_v, R * ra_v, -R, R)$
- Raonament similar per recalculer window quan $ra_v < 1$

IDI Q2 2019-2020

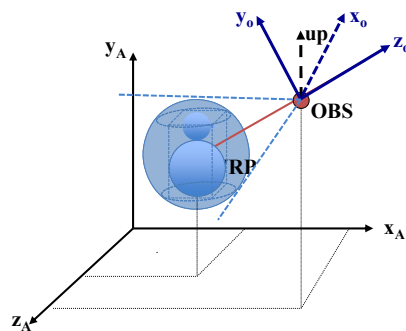
Classe 4: Contingut

- Breu repàs càmera i exercicis d'especificació de càmera
- Zoom
- Breu repàs del Procés de Visualització d'OpenGL i ubicar models en escena en el Vèrtex Shader
- Càmera en tercera persona
- **Per anar pensant...**
 - Com moure la càmera per inspeccionar escena?
 - Alguns exercicis

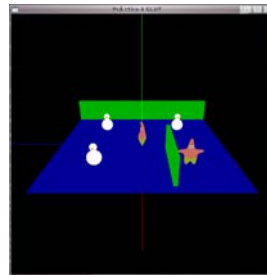
IDI Q2 2019-2020

Vist...

- Posicionament: OBS, VRP, up \rightarrow viewMatrix
- Òptica perspectiva: zN, zF, FOV, ra \rightarrow projectMatrix
- Càmera en 3ra persona: posició inicial

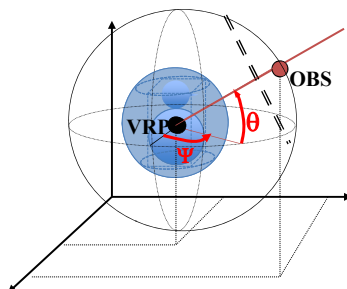


Com Moure la Càmera
per inspeccionar escena?



IDI Q2 2019-2020

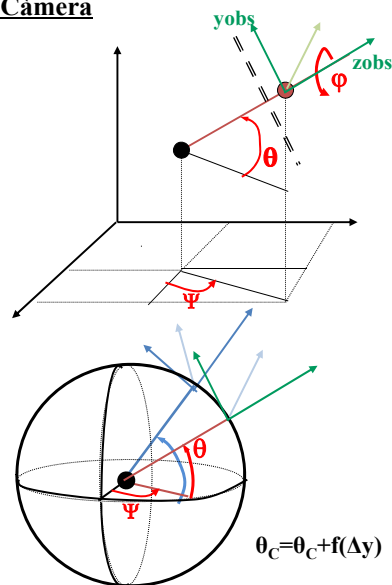
Moure la Càmera



- Els angles (d'Euler) determinen la posició d'un punt en l'esfera
- Des de la interfície d'usuari desplacem el cursor dreta/esquerra (Ψ) i pujar/baixar (θ); per moure **OBS** sobre l'esfera

Com calculem **OBS, VRP, up?**

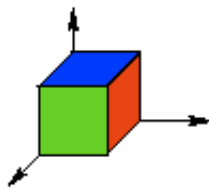
```
VM = lookAt (OBS, VRP, up);  
viewMatrix (VM);
```



IDI Q2 2019-2020

Exemple 4: Tenim una escena amb un cub de costat 2 orientat amb els eixos i de manera que el seu vèrtex mínim està situat a l'origen de coordenades. La cara del cub que queda sobre el pla $x=2$ és de color vermell, la cara que queda sobre el pla $z=2$ és de color verd i la resta de cares són blaves.

Indica TOTS els paràmetres d'una càmera que permeti veure en pantalla una polígon hexàgon regular amb l'interior de 3 colors. La relació d'aspecte del viewport (vista) és 2.



IDI Q2 2019-2020

Exemple 5: Una esfera de radi 1 es visualitza en un viewport quadrat de 400 per 400, amb una càmera posicionada correctament per poder veure tota l'esfera, i on el mètode per a definir la projecció de la càmera utilitza la següent crida:

```
TP = Perspective (60.0, 1.0, 1.0, 10.0);
projectMatrix (TP);
```

L'usuari ha redimensionat la finestra a 500 d'amplada per 400 d'alçada. Digues què cal canviar de la càmera per tal que es vegi l'esfera correctament (sense retallar-la ni deformar-la).

- Incrementar l'angle d'obertura vertical (FOV) i la relació d'aspecte del window.
- Augmentar la relació d'aspecte del window i la distància al ZNear.
- Només augmentar la relació d'aspecte del window.
- Només canviar l'angle d'obertura vertical (FOV).

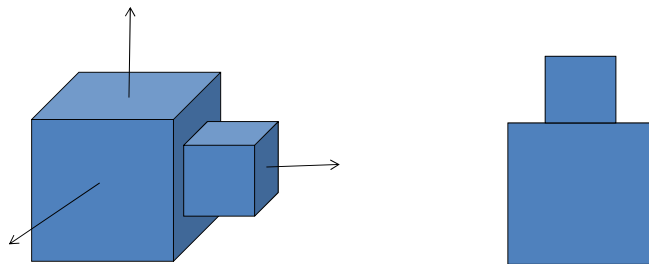
IDI Q2 2019-2020

Exemple 6: Quan s'inicialitza la càmera, en quin ordre cal indicar les transformacions de càmera i el viewport a OpenGL?

- a) No importa l'ordre en què s'indiquen.
- b) Transformació de posició + orientació, transformació de projecció, *viewport*.
- c) La transformació de projecció, transformació de posició + orientació, *viewport*.
- d) *Viewport*, transformació de projecció, transformació de posició + orientació.

IDI Q2 2019-2020

Exemple 7: Una escena està formada per dos cubs, un de costat 20 centrat al punt (0,0,0), i l'altre de costat 10 centrat al punt (15,0,0). Indiqueu TOTS els paràmetres d'una càmera **ortogonal/perspectiva** que permeti veure a la vista dos quadrats, un damunt de l'altre (el més gran a sota), de manera que ocupin el màxim de la vista (*viewport*). Cal que indiqueu la posició i orientació de la càmera especificant: **VRP**, **OBS** i **up**



IDI Q2 2019-2020

Classe 4: Conceptes i preguntes

- Zoom i modificació de paràmetres que comporta.
- Càlcul dels paràmetres de càmera per crear una imatge determinada.
- Valor prohibit pel vector up. Ha d'estar normalitzat? Ha de coincidir amb l'eix Y del sistema de coordenades de l'observador?.
- Com passar de coordenades de model a coordenades d'escena en el Vertex Shader?.
- Si hi ha un objecte que ja està correctament posicionat en el seu VAO, cal passar una TG al shader? Suposa que l'escena té més d'un objecte.
- Propietats de la càmera en tercera persona.
- Capsa i esfera contenidores de l'escena. Són les d'un model?
- Càlcul dels paràmetres d'una càmera en tercera persona.
- Què cap complir per a què no hi hagi deformació? I per a poder veure sempre tota l'escena.
- Re-càlcul dels paràmetres de l'òptica en fer un "resize". En quina part del codi els re-calcularies?

IDI Q2 2019-2020