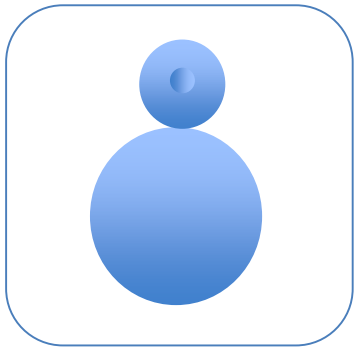
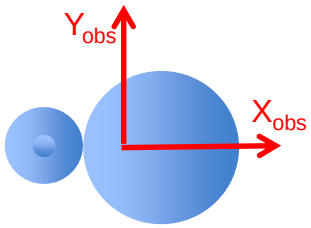
Alguns exercicis

Exemple 1 bis:

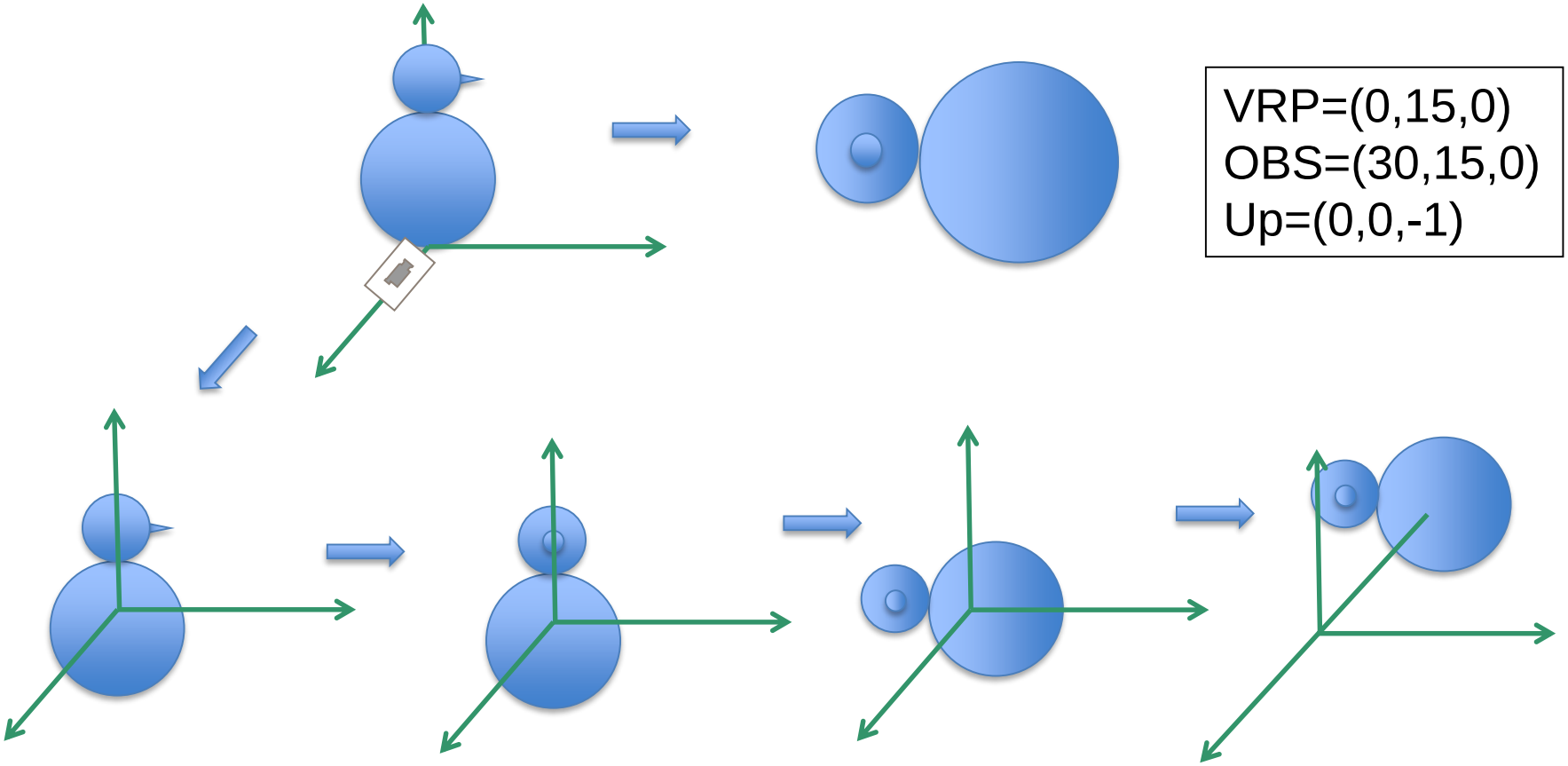


VRP=(0,15,0); OBS=(30,15,0), up=(0,1,0)



Quins paràmetres si volem que quedi així?
amb lookAt() i ***amb Euler***

Solució Exemple 1 bis.

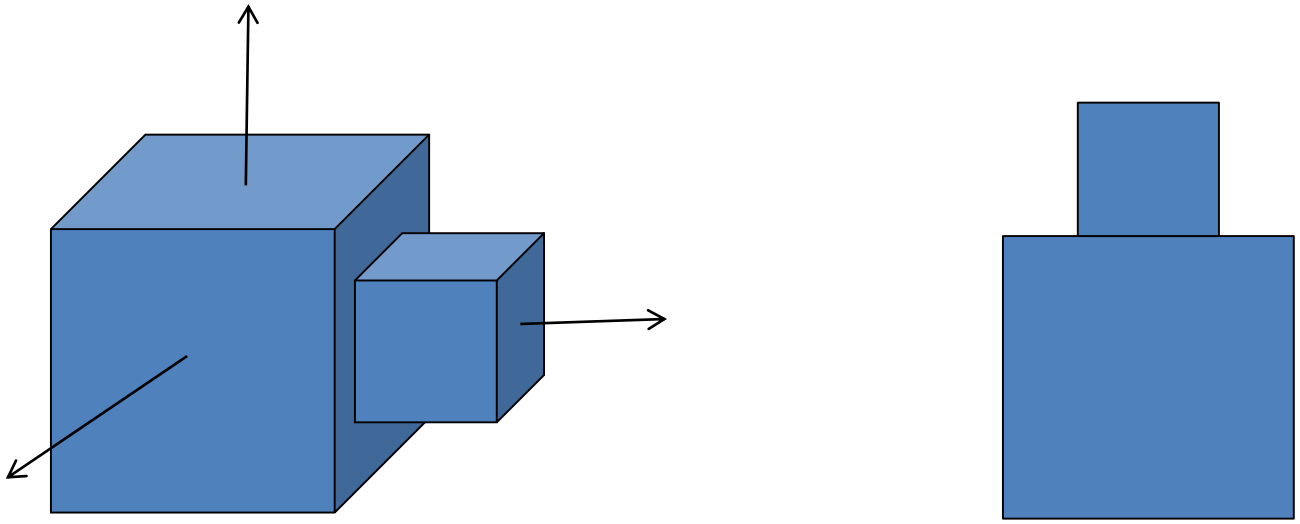


VRP=(0,15,0)
OBS=(30,15,0)
Up=(0,0,-1)

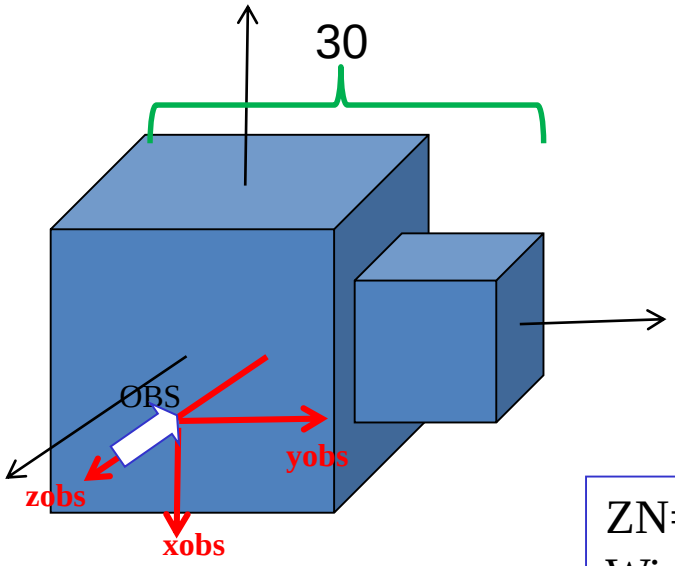
$TC=T(0,0,-30)G_z(90)G_Y(-90)T(0,-15.0)$

```
VM= Translatef(0.,0.,-30.);
VM= VM*Rotate (90.,0.,0.,1.);
VM= VM*Rotate (-90.,0.,1.,0.);
VM= VM*Translate (0.,-15,0.);
viewMatrix(VM);
Pinta_Ninot();
```

Exemple 6: 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; a) VRP, OBS i up b) **angles Euler**. El viewport és quadrat.



Solució 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; a) VRP, OBS i up b) **angles Euler**. El viewport és quadrat.



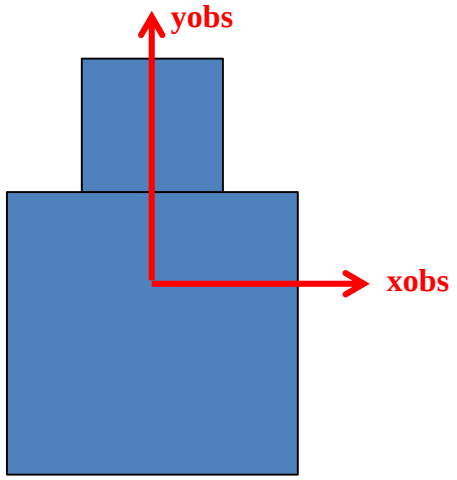
cmin= (-10, -10,-10)
cmax=(20,10,10)

OBS=(5,0,25)
VRP=(5,0,0)
Up=(1,0,0)

VRP=(5,0,0)
d= 25
Psi,fita=0
Fi=90º

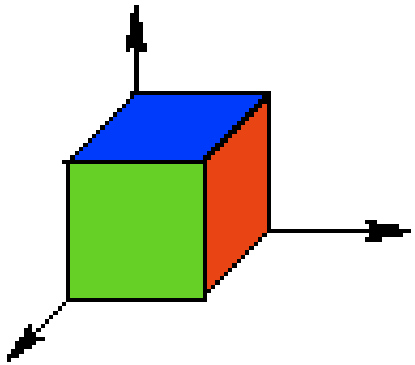
ZN=5 ZF=35
~~Window= (-10,10,-15,15)~~

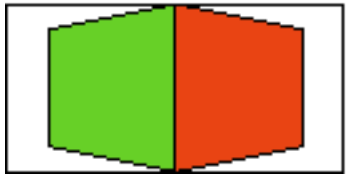
Window= (-15,15,-15,15)



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.

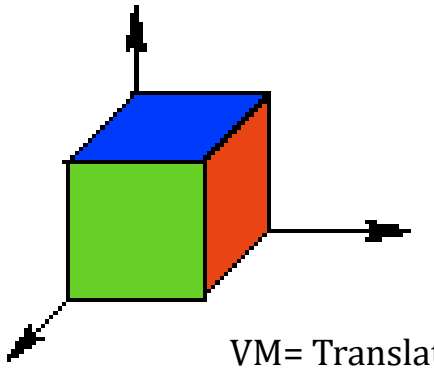
Indica TOTS els paràmetres d'una càmera perspectiva que permeti veure completes a la vista només les cares vermella i verda. La relació d'aspecte del viewport (vista) és 2. Fes un dibuix indicant la imatge final que s'obtindria. **Posiciona la càmera amb angles d'Euler**.

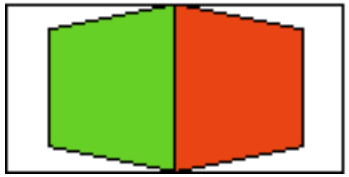




Solució exemple 2: Tenim una escena amb un cub de costat 2 orientat amb els eixos i de manera que el seu vèrtex mínim està situat a l'origen de coordenades. La cara del cub que queda sobre el pla x=2 és de color vermell, la cara que queda sobre el pla z=2 és de color verd i la resta de cares són blaves.

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'obtindria. **Posiciona la càmera amb angles d'Euler**.





VM= Translatef(0.,0.,- √2)
VM= VM*Rotate (-45.,0.,1.,0.);
VM= VM*Translate (-2.,-1,-2.);
viewMatrix(VM);
Pinta_cub();

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

VRP= (2,1,2)
d= √2
Psi=-45, fita=0,
fi=0

Exercici 73 (de la llista de càmera): Es vol realitzar una vista en planta (visió des de dalt) d'una escena/objecte que està centrat a l'origen amb una capsa contenidora de mides 10x10x10. Quina de les següents definicions et sembla correcta per definir la posició + orientació de la càmera (per calcular la viewMatrix)? Sabem que la càmera és perspectiva i els angles de les rotacions estan en graus.

a) OBS = (0,10,0); VRP = (0,0,0); up = (0,1,0);
VM = lookAt (OBS, VRP, up);
viewMatrix(VM);

b) OBS = (0,0,0); VRP = (0,10,0); up = (0,0,-1);
VM = lookAt (OBS, VRP, up);
viewMatrix(VM);

c) VM = translate (0,0,-10);
VM = VM * rotate (90, 1,0,0);
viewMatrix(VM);

d) VM = translate (0,0,-10);
VM = VM * rotate (-90, 0,1,0);
viewMatrix(VM);

Exemple 8: 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 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).

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

Exercici 22 (de la llista de TG): Imagina que tenim una escena amb una vaca i un Patricio i els volem fer girar entorn l'eix Y, com si es tractés d'una peça d'uns cavallets ("tiovivo"). Suposant que TG1 és la matriu de TG per ubicar la vaca i TG2 és la matriu de TG per ubicar el Patricio quin dels següents codis és correcte?.

a) AUX= Rotate(alfa,0,1,0) TG1= AUX*TG1 TG2= AUX*TG2 modelMatrix(TG1) pintaVaca() modelMatrix(TG2) pintaPatricio()	b) modelMatrix(TG1) pintaVaca() Rotate (alfa,0,1,0) modelMatrix(TG2) pintaPatricio() Rotate (alfa,0,1,0)
c) AUX= Rotate(alfa, 0,1,0) TG1=TG1*AUX modelMatrix(TG1) pintaVaca() TG2=TG1*TG2 modelMatrix(TG2) pintaPatricio()	d) AUX= Rotate(alfa, 0,1,0) TG1=AUX*TG1 modelMatrix(TG1) TG2=AUX*TG2 modelMatrix(TG2) pintaVaca() pintaPatricio()

Exercici 24 (de la llista de TG): Tenim un objecte centrat a l'origen i amb capsa contenidora de mides 3 d'ample, 3 d'alçada i 3 de profunditat. Es vol modificar només la seva alçada per a què passi a ser 2, quina de les següents TG és la correcta?

a) TG = glm::scale (glm::mat4(1.f), glm::vec3(1.0, 2.0, 1.0));
b) TG = glm::scale (glm::mat4(1.f), glm::vec3(3.0, 2.0, 3.0));
c) TG = glm::scale (glm::mat4(1.f), glm::vec3(1.0, 2.0/3.0, 1.0));
d) TG = glm::scale (glm::mat4(1.f), glm::vec3(2.0/3.0, 2.0/3.0,2.0/3.0));

Exercici 43 (de la llista de càmera). Indica quina de les inicialitzacions de l'òptica perspectiva és més apropiada per a una càmera que porta un observador que camina per una escena fent fotos amb una òptica constant. Esfera englobant d'escena té radi R, d és la distància entre OBS i VRP. Observació: ra_v és la relació d'aspecte del *viewport*

a) FOV = 60º, ra = ra_v, zNear = 0.1, zFar = 20
b) FOV = 60º, ra = ra_v, zNear = R, zFar = 3R;
   essent R el radi de l'esfera contenidora de l'escena.
c) FOV = 2*(arcsin(R/d)*180/PI), ra = ra_v, zNear = R, zFar = 3R;
   essent R el radi de l'esfera contenidora de l'escena i d la distància d'OBS a VRP.
d) FOV = 2*(arcsin(R/d)*180/PI), ra = ra_v, zNear = 0, zFar = 20;
   essent R el radi de l'esfera contenidora de l'escena i d la distància d'OBS a VRP

Exercici 71 (de la llista de càmera): Cal definir una càmera a OpenGL; quin dels seg üents pseudocodis és correcte? Noteu que tant sols canvia l'ordre en què es fan les crides.

1) VM=lookAt(OBS, VRP, up)
viewMatrix (VM)
PM=perspective (FOV, ra, zn,zf)
projectMatrix(PM)
glViewport(...)
modelMatrix(TG)
pintaescena()

2) modelMatrix(TG)
PM=perspective (FOV, ra, zn,zf)
projectMatrix(PM)
VM=lookAt(OBS, VRP, up)
viewMatrix (VM)
glViewport(...)
pintaescena()

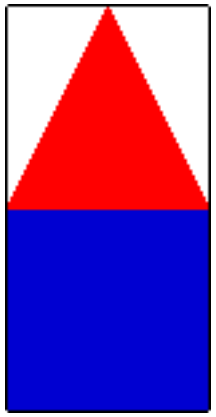
3) VM=lookAt(OBS, VRP, up)
viewMatrix (VM)
PM=perspective (FOV, ra, zn,zf)
projectMatrix(PM)
modelMatrix(TG)
glViewport(...)
pintaescena()

4) glViewport(...)
VM=lookAt(OBS, VRP, up)
viewMatrix (VM)
PM=perspective (FOV, ra, zn,zf)
projectMatrix(PM)
modelMatrix(TG)
pintaescena()

---

a) només 1) i 4) són correctes
b) només 4) és correcte
c) tots són correctes
d) tots són correctes menys 2)

Exercici 72 (de la llista de càmera): Tenim una piràmide de base quadrada de costat 5, amb la base centrada al punt (0,0,2.5) i alçada de la piràmide 5 amb l'eix en direcció Z+. A l'escena tenim també un cub de costat 5 centrat a l'origen. El viewport esta definit amb glViewport (0,0,400,800). Si a la vista es veu la imatge que teniu al dibuix (caseta), quines inicialitzacions d'una càmera axonomètrica (posició+orientació i òptica) permetrien veure aquesta imatge? Tots els angles estan en graus.



```
PM=perspective(90,1,5,10);
projectionMatrix (PM)
VM=translate (0,0,-10);
VM=VM*rotate (90,1,0,0);
VM=VM*translate (0,0,-2.5);
viewMatrix (VM);
pinta_escena ();
```

```
PM=ortho(-2.5,2.5,-5,5,5,10);
projectionMatrix (PM)
VM=translate (0,0,-7.5);
VM=VM*rotate (-90,0,0,1);
VM=VM*rotate (90,0,1,0);
VM=VM*translate (0,0,-2.5);
viewMatrix (VM);
pinta_escena ();
```

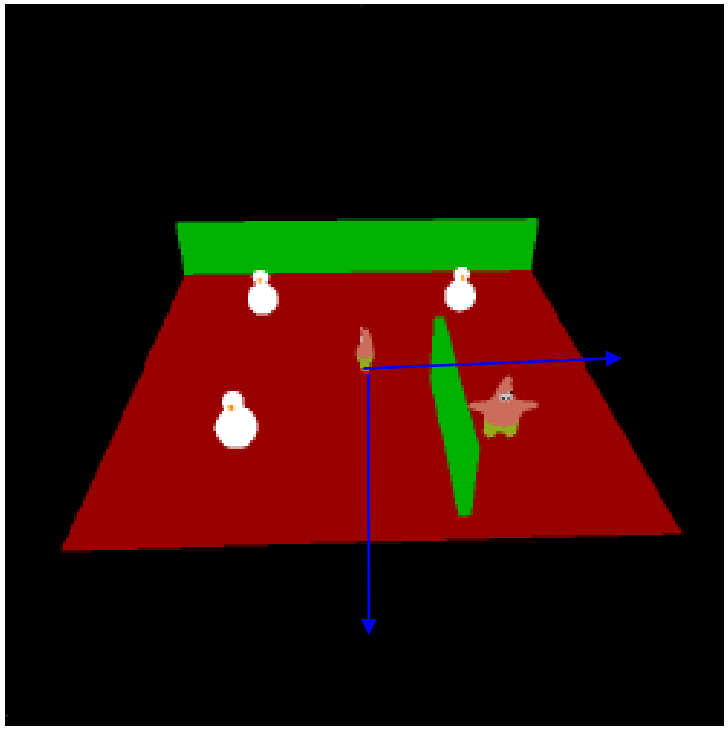
```
PM=ortho(-2.5,2.5,-5,5,5,10);
projectionMatrix (PM)
VM=translate (0,0,-7.5);
VM=VM*rotate (90,0,0,1);
VM=VM*rotate (90,0,1,0);
VM=VM*translate (0,0,-2.5);
viewMatrix (VM);
pinta_escena ();
```

```
PM=ortho(-5,5,-5,5,5,10);
projectionMatrix (PM)
VM=translate (0,0,-7.5);
VM=VM*rotate (90,0,0,1);
VM=VM*rotate (90,0,1,0);
VM=VM*translate (0,0,-2.5);
viewMatrix (VM);
pinta_escena ();
```

Per pensar: Càmera en primera persona

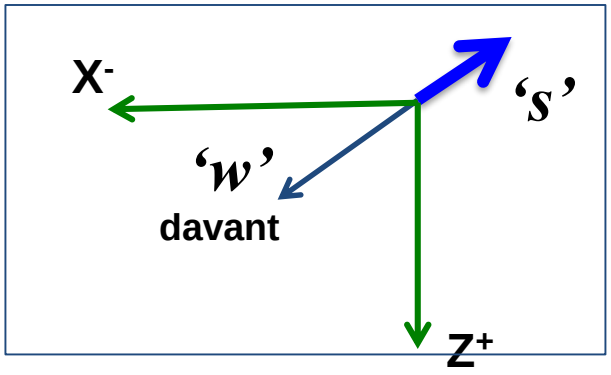
Altres exercicis recomanats de la llista de exercicis de càmera:

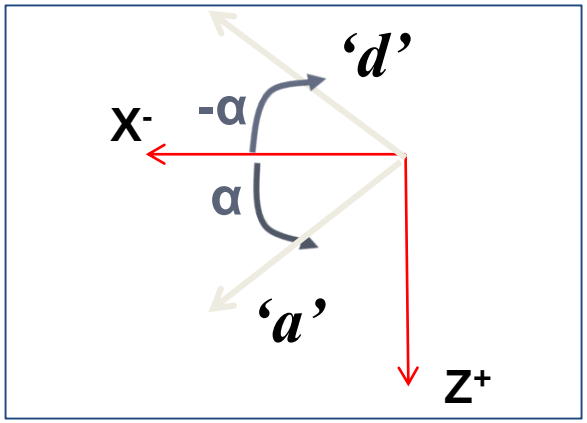
22, 24, 66, 67, 73, 82, 103, 104, 105 i 114



Moure el Patricio central

- Avançar/retrocedir (tecles ‘w’ i ‘s’):
  - modificar posició en la direcció del moviment “davant”
- Girar a la dreta/esquerra (tecles ‘d’ i ‘a’)
  - modificar “davant” (gir respecte eix Y).





OBS = (0,1,0)
VRP = (-10,1,0)
Up = (0,1,0)

zN = 0.1
zF = 15
FOV= 60º
ra = ra_v