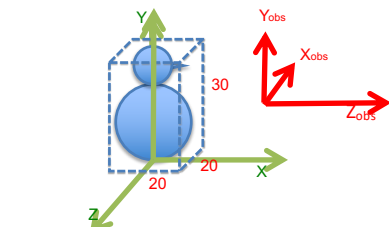


Alguns exercicis

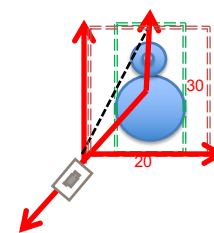
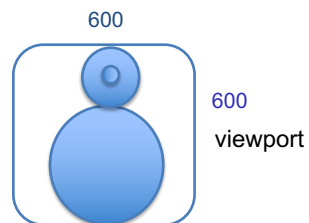
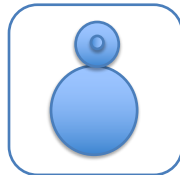
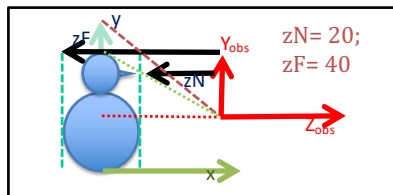
IDI 2018-2019 1Q

1

Exemple 1: Òptica perspectiva



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




$\alpha = \arctg(15/20) \rightarrow \alpha = 36,8^\circ$
 $ra_W = 20/30 = 0,66$

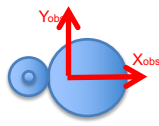
Com $ra_v = 1$ i $ra_W = 20/30 \rightarrow$ deformació

Solució $ra_W = 1$

IDI 2018-2019 1Q



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

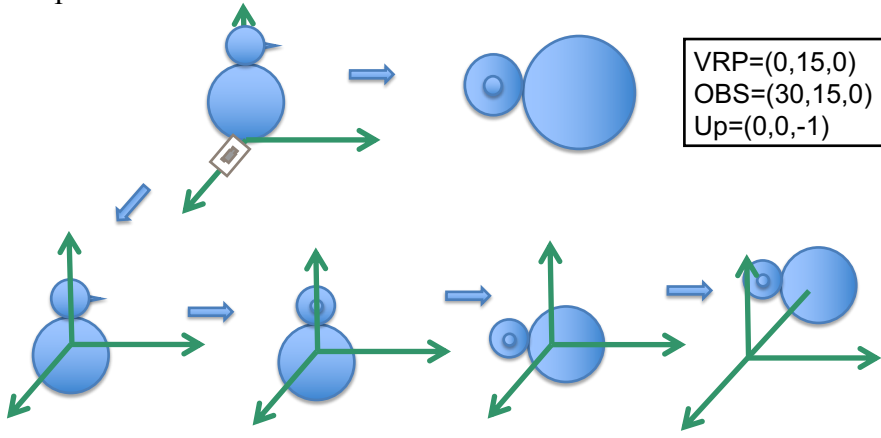


Quins paràmetres si volem que quedi així? amb lookAT() i amb Euler

IDI 2018-2019 1Q

3

Exemple Ninot: càlcul de VM amb lookAT i Euler



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

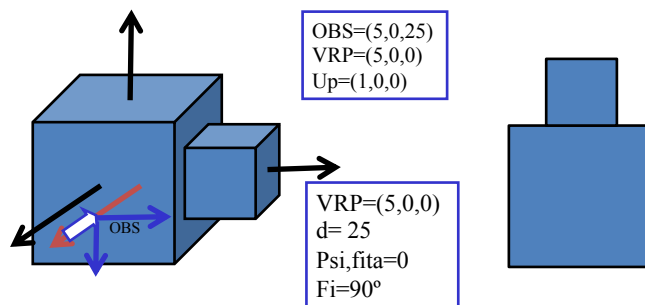
$TC=T(0,0,-30)G_z(90)G_y(-90)T(0,-15.0)$

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

IDI 2018-2019 1Q

4

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

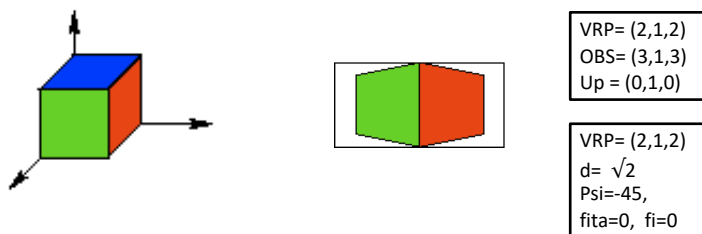


5

IDI 2018-2019 1Q

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. Posiciona la càmera també amb Euler.



6

IDI 2018-2019 1Q

Es vol realitzar una vista en planta (visió des de dalt) d'una escena/objecte que està centrat a l'origen amb una capsula contenidora de mides 10x10x10. Quina de les següents definicions et sembla correcta per definir la posició + orientació de la càmera (per a 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);`

IDI 2018/2019 1Q

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).

- 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).

IDI 2018-2019 1Q

8

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 2018-2019 1Q

9

Imagina que tenim una escena amb una vaca i un patricio i els volem girar entorn l'eix Y (com si es tractés d'una peça d'uns cavallets ("tio vivo")). 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?

<p>a)</p> <pre>AUX= Rotate(alfa,0,1,0) TG1= AUX*TG1 TG2= AUX*TG2 modelMatrix(TG1) pintaVaca() modelMatrix(TG2) pintaPatricio()</pre>	<p>b)</p> <pre>modelMatrix(TG1) pintaVaca() Rotate (alfa,0,1,0) modelMatrix(TG2) pintaPatricio() Rotate (alfa,0,1,0)</pre>
<p>c)</p> <pre>AUX= Rotate(alfa, 0,1,0) TG1=TG1*AUX modelMatrix(TG1) pintaVaca() TG2=TG1*TG2 modelMatrix(TG2) pintaPatricio()</pre>	<p>d)</p> <pre>AUX= Rotate(alfa, 0,1,0) TG1=AUX*TG1 modelMatrix(TG1) TG2=AUX*TG2 modelMatrix(TG2) pintaVaca() pintaPatricio()</pre>

IDI 2018-2019 1Q

10

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));`

IDI 2018-2019 1Q

11

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) $FOV = 60^\circ$, $ra = ra_v$, $zNear = 0.1$, $zFar = 20$
- b) $FOV = 60^\circ$, $ra = ra_v$, $zNear = R$, $zFar = 3R$;
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

IDI 2018-2019 1Q

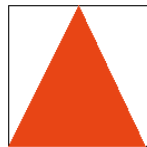
Cal definir una càmera a OpenGL; quin dels següents pseudocodis és correcte? Noteu que tant sols canvia l'ordre en què es fan les crides.

- | | |
|---|---|
| <p>1) VM=lookAt(OBS, VRP, up)
viewMatrix (VM)
PM=perspective (FOV, ra, zn,zf)
projectMatrix(PM)
glViewport(...)
modelMatrix(TG)
pintaescena()</p> | <p>2) modelMatrix(TG)
PM=perspective (FOV, ra, zn,zf)
projectMatrix(PM)
VM=lookAt(OBS, VRP, up)
viewMatrix (VM)
glViewport(...)
pintaescena()</p> |
| <p>3) VM=lookAt(OBS, VRP, up)
viewMatrix (VM)
PM=perspective (FOV, ra, zn,zf)
projectMatrix(PM)
modelMatrix(TG)
glViewport(...)
pintaescena()</p> | <p>4) glViewport(...)
VM=lookAt(OBS, VRP, up)
viewMatrix (VM)
PM=perspective (FOV, ra, zn,zf)
projectMatrix(PM)
modelMatrix(TG)
pintaescena()</p> |

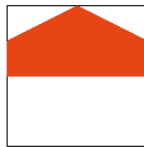
- a) nom_es 1) i 4) són correctes
b) només 4) és correcte
c) tots són correctes
d) tots són correctes menys 2)

IDI 2018-2019 1Q

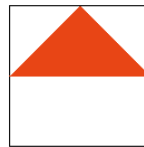
Tenim una escena amb un triangle vermell amb vèrtexs $V1=(-2,0,0)$, $V2 = (2, 0, 0)$ i $V3=(0, 1, 0)$. Suposant que tenim un viewport quadrat de 600x600 píxels, i que hem inicialitzat les matrius de càmera (view) i projecció (proj) a la matriu identitat, indica quina de les següents imatges és la que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



a)



b)



c)



d)

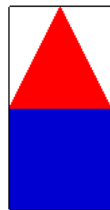
IDI 2018-2019 1Q

Dos estudiants discuteixen respecte a la implementació del zoom amb òptica axonomètrica (ortogonal) i perspectiva. Quina de les seves afirmacions és certa?

- a) En òptica ortogonal només es pot obtenir un efecte de zoom modificant OBS i VRP en la direcció de visió.
- b) En òptica perspectiva cal modificar FOV, Znear i Zfar.
- c) En les dues òptiques es pot fer zoom modificant el window de la càmera.
- d) En òptica perspectiva si avancem OBS i VRP en la direcció de visió cal anar amb compte amb la ra.

IDI 2018-2019 1Q

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 està 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.



<pre>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 ();</pre>	<pre>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 ();</pre>
<pre>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 ();</pre>	<pre>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 ();</pre>

Disposem d'una càmera ortogonal amb els següents paràmetres:

OBS=(0.,0.,0.), VRP=(-1.,0.,0.), up=(0.,1.,0.), window de (-5,-5) a (5,5), ra=1, zn=5, zf=10.

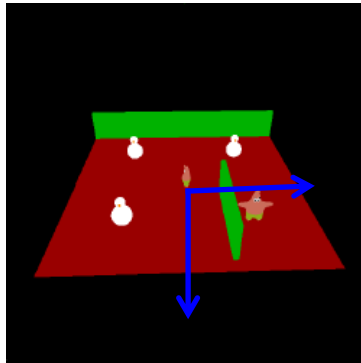
Indiqueu quin conjunt de paràmetres d'una càmera perspectiva defineix un volum de visió que conté l'anterior (és a dir, garanteix que es veurà, com a mínim, el mateix que amb la càmera axonomètrica):

- a) FOV= 90, ra=1, zn= 5, zf=10
- b) FOV= 60, ra=1, zn=5, zf=10
- c) FOV= 60, ra= 2, zn=6, zf=11
- d) FOV= 90, ra= 0.5, zn=5, zf=10

Per pensar: Càmera en primera persona

Exercicis de la llista a fer (mínims):

- 33
- 45
- 22
- 48
- 52
- 63
- 70
- 86
- 89



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

zN = 0.1
zF = 15
FOV = 60°
ra = ra_v

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).

