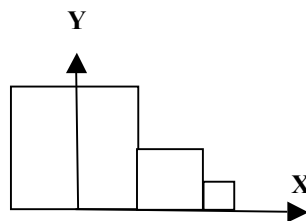


Estructura de dades i transformacions geomètriques¹

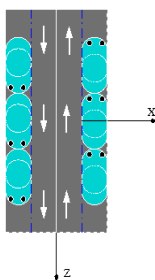
1. (2005-2006 2Q) Es disposa d'una acció `pinta_esfera()` per pintar el model d'una esfera aprofitada per triangles. Es desitja visualitzar un sistema solar molt simple: el sol, els planetes i els satèl·lits es consideren esferes, i les òrbites dels planetes i dels satèl·lits es consideren circulars essent els eixos de gir paral·lels a l'eix Y del sistema de coordenades de l'escena i passant pel centre del sol (en el cas dels planetes) o dels planetes (en el cas dels seus satèl·lits). Es demana:
 - Suposant que teniu definida una camera, dissenyeu el tros de codi que utilitzant `pinta_esfera()` i comandes OpenGL genera la imatge del sistema solar en un instant de temps determinat.
2. (2006-2007 1Q) Disposem del model d'un personatge que té com a capsa contenidora la capsa definida pels punts $\text{MinCapsa}=(10,10,10)$ i $\text{MaxCapsa}=(20,20,12)$. Ens diuen que la part del davant d'aquest personatge és la que mira cap a la part positiva de l'eix Z. Considerem que tenim una habitació dividida en 10×10 cel·les quadrades de costat $c=1$ situades en el pla XZ (la cel·la $x=1, z=1$ té coordenades mínimes $x_{\min}=0$ i $z_{\min}=0$). Quines transformacions li haurem de fer al personatge si el volem situar centrat i escalat adientment en la cel·la $x=5, z=3$ de la nostra habitació i mirant cap a la cel·la $x=5, z=4$? Escriu el tros de codi amb les transformacions en OpenGL que li aplicaries. L'alçada del terra de l'habitació (sobre el qual cal posar el personatge) és $y=0$.
3. (2006-2007 1Q) Suposa definida una funció `dibuixaCub()` que dibuixa un cub d'aresta unitat centrat a l'origen. Fent servir aquesta funció i les transformacions geomètriques d'OpenGL, has d'obtenir una successió de tres cubs similar a la indicada a la figura. Tots els cubs estan recolzats sobre el pla XZ. El primer cub té aresta unitat i el centre de la seva base està ubicat a l'origen de coordenades. El segon està a la dreta del primer, adossat a ell, la seva aresta és la meitat que la d'ell i l'eix X passa pel centre de la seva base. Anàlogament el tercer cub respecte del segon.



4. (2006-2007 2Q) Suposem que tenim el model d'un personatge que, en el seu sistema de coordenades local, té el davant en la direcció positiva de l'eix Z i la seva capsa contenidora té dimensions $x=50, y=100$ i $z=65$ i està centrada en el punt $C=(0,50,50)$. Quines transformacions de model caldrà aplicar a aquest personatge per a situar-lo, convenientment escalat, en el nostre laberint en la casella $[3,5]$? Cada casella del laberint té una dimensió de 1×1 , les caselles es numeren començant per la $[0,0]$ que té de coordenades mínimes el punt $(0,0,0)$, i el terra està a 0.25 d'alçada i és paral·lel al pla X-Z del sistema de coordenades de l'aplicació.

¹ Basats en exercicis d'examen de les assignatures VIG (pla 2013 d'Enginyeria Informàtica) i IDI del Grau en Enginyeria Informàtica (a partir del curs 2011-2012). Recordeu que trobareu solucionats la majoria dels exàmens en la web de la DAFIB i en la de la biblioteca.

5. (2007-2008 1Q) Tenim el model d'un objecte bàsic i un mètode `PintaObjecte()` que l'envia a pintar el VAO que el conté. Coneixem els punts de coordenades mínimes (`MinX, MinY, MinZ`) i màximes (`MaxX, MaxY, MaxZ`) de la seva capsa mínima contenidora (amb les seves arestes orientades segons els eixos de coordenades). Volem visualitzar una instància d'aquest objecte tal que l'amplada (mida segons l'eix X) de la seva capsa sigui `AmplaX` (escalat uniforme) i el centre de la base d'aquesta estigui ubicat en el punt (10,0,10). Indiqueu:
 - L'expressió de la composició de les transformacions geomètriques que cal efectuar a l'objecte bàsic per tal de visualitzar la instància utilitzant el mètode `PintaObjecte()`.
 - El codi OpenGL que hauríeu d'implementar.
6. (2007-2008 2Q) Es disposa del model geomètric d'una butaca. La seva capsa contenidora està centrada a l'origen de coordenades. La direcció cap amunt de la butaca és la de l'eix Y positiu i el seu davant està orientat cap a la part negativa de l'eix X. Utilitzant aquest model, es vol visualitzar la platea d'un teatre que consta de 20 files de 10 butaques cadascuna. Les files són paral·leles a l'eix Z positiu. La primera butaca de la primera fila, `butaca[1,1]`, estarà situada centrada en l'origen de coordenades (la seva capsa contenidora quedarà centrada a l'origen de coordenades). La separació entre els centres de dues butaques d'una mateixa fila serà de +5. La separació entre els centres de la butaca `[i,j]` i la butaca `[i+1,j]` de dues files consecutives serà de -10. El davant de la cadira estarà orientat cap a les X positives. Escriviu el tros de codi necessari per a pintar les 200 butaques de la platea del teatre. Suposeu que ja teniu inicialitzades les matrius `PROJECTION` i `MODELVIEW` i que disposeu d'una acció `pinta_butaca()` que envia a pintar el VAO que conté el model de la butaca.
7. (2008-2009 1Q) La trajectòria d'un cotxe es representa per una poligonal de 20 punts ubicada en el pla x-y. Es disposa de la informació de la caixa mínima contenidora del cotxe i d'una funció `pinta_cotxe()` que pinta un cotxe centrat a l'origen. Indica l'expressió de les transformacions geomètriques que cal efectuar al cotxe per a generar una animació que ubica successivament el cotxe en els vèrtexs de la trajectòria, amb un escalat uniforme decreixent proporcional a l'índex del vèrtex del polígon (el cotxe es va fent petit). Indiqueu també el codi OpenGL que, suposant una càmera ja inicialitzada, pinti el cotxe en la posició indicada.
8. (2008-2009 2Q) Un joc de computadors es basa en la recreació d'una ciutat romana. Es disposa del model de fronteres de 5 romans amb diferent indumentària, els anomenarem "romans patró". En l'escena poden haver-hi fins a 100 romans de cada tipus (és a dir, fins a 100 de cada "patró"). La ubicació de cada romà queda determinada pel centre de la seva capsa mínima contenidora, la seva orientació per un vector i la seva grandària serà la del romà patró corresponent escalat un cert %. Es demana el disseny d'una estructura de dades compacta que permeti emmagatzemar les dades requerides per a la visualització de la població dels romans de la ciutat. El model de fronteres dels romans "patró" està constituït per una malla de triangles on cada triangle pot ser d'un color diferent.
9. (2010-2011 2Q) Tenim un cilindre d'altura 1 i radi 1 amb una de les seves bases centrada al (0, 0, 0) i l'altra al (0, 0, 1). Suposant que la funció `dibuixaCilindre()` envia a pintar el model d'aquest cilindre, escriu el codi OpenGL que es necessitaria per a obtenir en la nostra escena un cilindre de radi 2 i amb les seves bases situades als punts (2,-1,-1) i (6,-1,-1).
10. (2011-2012 1Q) Disposem d'un model de cotxe que la seva part del davant mira cap a la part positiva de l'eix X i la seva capsa contenidora ve definida pels punts `Min=(3,2,0)` i `Max=(5,3,1)`. Tenim la funció `PintaCotxe()` que pinta aquest cotxe. Disposem també d'un model de carrer, d'amplada 4, centrat a l'origen, que el seu eix central coincideix amb l'eix Z i pel qual circulen els cotxes en totes dues direccions (és a dir el carrer és de doble sentit). Volem disposar d'aquests cotxes aparcats a les 2 vores del carrer de manera que estiguin aparcats orientats en la direcció correcta de circulació del carrer. Suposant que els cotxes no s'han d'escalar i que han d'estar tocant-se amb el de davant/darrera (veure esquema de la figura), indica el tros de codi OpenGL que usaries per a pintar aquests 6 cotxes suposant una càmera ja inicialitzada.



11. (2011-2012 2QP) Disposem del model geomètric d'un cub de costat 1 amb cares paral·leles als plans coordenats amb el centre de la seva base a $(0,0,0)$. Suposant que la funció `dibuixaCub()` ja fa les crides necessàries a OpenGL per a pintar aquest cub, escriu (i justifica tot indicant la transformació geomètrica requerida) el codi OpenGL "específic" que es necessitaria per a obtenir una escena formada per dos cubs: un de costat 2 amb el centre de la seva base a l'origen de coordenades i l'altre amb els vèrtexs de la seva cara inferior al centre de les arestes de la cara superior del primer cub. Considereu la càmera ja inicialitzada, només cal el codi per a pintar els dos cubs de l'escena.

12. (2011-2012 2QP) Suposem que tenim el model geomètric d'un prisma de base hexagonal amb una capsa contenidora de dimensions $a=40$, $h=100$ i $z=40$, centrada en el punt $C=(50,0,0)$. Es vol obtenir la visualització d'una escena formada per dos prismes orientats segons l'eix Z positiu; el primer ha de tenir alçada 100 i estar centrat a l'origen de coordenades, el segon d'alçada 50 ha d'estar ubicat just darrera del primer. Si disposem d'una funció `dibuixaPrisma()` que fa les crides a OpenGL per pintar el model del prisma. Escriu (i justifica) el codi OpenGL "específic" que es necessari per a generar l'escena desitjada.

13. (2012-2013 1QP) Com visualització il·lustrativa del joc de tres en ratlla, es vol mostrar una imatge formada per tres Homers de la mateixa alçada 2 tals que el seus centres estan ubicats a: $(0,1,0)$, $(-2,1,0)$, $(2,1,0)$, i els seu davant orientats, respectivament, en la direcció dels eixos $+Z$, $-X$ i $+X$. Es disposa de la funció:

void drawHomer ();

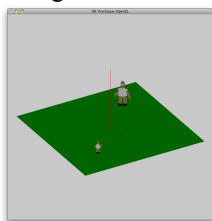
que pinta un Homer d'alçada 2 amb el seu davant orientat cap a l'eix $+Z$ i centre en $(0,1,0)$. Observació: podeu considerar que l'esfera contenidora del Homer té radi=1.

Es demana: usant `drawHomer()`, el codi necessari en OpenGL per a dibuixar els Homers de l'alçada i orientació indicades i en les posicions donades. Supposeu que la càmera i el viewport (vista) estan correctament inicialitzats.

14. (2013-2014 1Q) Disposem de la informació de la capsa contenidora d'un model $(xmin, ymin, zmin, xmax, ymax, zmax)$ que sabem està orientat amb el seu nas mirant en la direcció $(0,0,1)$. Volem situar dos instàncies d'aquest model a sobre d'un terra ubicat en el pla $Y=0$ amb centre $(0,0,0)$ i de mides 10×10 , de manera que:

- Una instància del model quedi ubicada amb el centre de la base de la seva capsa en $(-2.5, 0, 2.5)$, tingui alçada 1 i miri el seu nas cap a l'eix Y.
- L'altre instància tingui el centre de la capsa de la seva base en $(2.5, 0, -2.5)$, alçada 2 i miri el seu nas cap l'eix Y.

Disposeu d'un mètode `pinta_model()` que l'envia a pintar el model. Indiqueu: la transformació geomètrica requerida per a ubicar cada instància en l'escena, i també el codi d'OpenGL per a pintar l'escena. Podeu suposar que la càmera ja està correctament inicialitzada. La figura adjunta és una visió general de l'escena.



15. (2013-2014 2Q) Disposem del model geomètric d'un objecte i de la funció `pinta_model()` que encapsula les primitives gràfiques requerides per a pintar les seves cares. Els vèrtexs extrems de capsa mínima contenidora del model tenen coordenades $(xmin, ymin, zmin)$ i

(xmax,ymax,zmax). El “davant” del model està orientat segons les X+. Es vol visualitzar una escena formada per dos instàncies del model: una d'elles (objecte 1) amb el centre de la base de la seva caps a l'origen de coordenades i l'altra (objecte 2) amb el centre de la base de la caps al punt (5,0,5). Les capses dels dos objectes han de tenir alçada=2, amplada=1, fondària=1, i els seus davants s'han d'estar mirant.

Es demana: el codi del mètode `pinta_escena()` que utilitzant `pinta_model()` permet generar l'escena. Cal justificar les transformacions geomètriques utilitzades.

16. (2013-2014 2Q) Disposem de la funció `pinta_model()` que encapsula les crides requerides per a pintar el model d'un *legoman*. Els vèrtexs extrems de caps mínima contenidora del model tenen coordenades (xmin,ymin,zmin) i (xmax,ymax,zmax), i el seu “davant” està orientat segons les Z+.

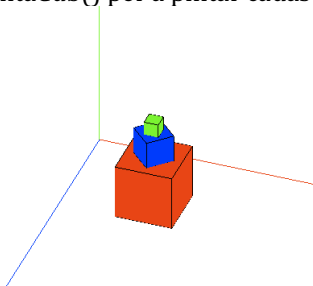
Es vol visualitzar una escena formada per dues instàncies d'aquest *legoman*: una (*legoman 1*) ha de tenir alçada=2, i l'amplada i la fondària han de ser 1; el centre de la base de la seva caps ha d'estar a l'origen de coordenades i el seu davant en la direcció de les Z+. L'altra instància (*legoman 2*) ha de tenir una caps el doble de gran que la del *legoman 1* i ubicada a sobre seu, però amb el seu “davant” mirant cap a Y+, l'eix Y (de l'escena) passant pel seu centre i essent l'eix que travessa el *legoman 2* de cap a peus paral·lel a l'eix X (de l'escena).

Es demana: el codi del mètode `pinta_escena()` que utilitzant `pinta_model()` permet generar l'escena. Cal justificar les transformacions geomètriques utilitzades.

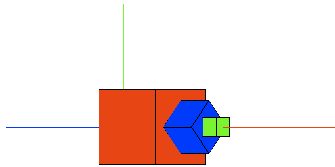
17. (2013-2014 2Q) Disposem del model geomètric d'un objecte i de la funció `pinta_model()` que encapsula les crides OpenGL per pintar-ho. Els vèrtexs extrems de caps mínima contenidora del model tenen coordenades (xmin,ymin,zmin) i (xmax,ymax,zmax). El “davant” del model està orientat segons les Z+. Es vol visualitzar una escena formada per dos instàncies del model: una d'elles (objecte 1) amb el centre de la base de la seva caps a l'origen de coordenades i escalat de manera que la seva caps sigui un cub d'aresta 2; i l'altra (objecte 2) que quedi just damunt del primer, de la meitat de la mides i cap per baix. Es demana: el codi del mètode `pinta_escena()` que utilitzant `pinta_model()` permet generar l'escena. Cal justificar les transformacions geomètriques utilitzades.

19. (2014-2015 P1Q) Disposem d'una funció `PintaCub()` que pinta un cub de costat 1 centrat a l'origen i amb les seves cares orientades segons els plans coordenats. Volem construir una escena que sigui un pilar de tres cubs de costats 2, 1 i 0.5 respectivament com es mostra a la figura (no importa el color). El cub de baix de tot té el centre de la seva base al punt (3,0, 3), i les seves cares estan orientades segons els plans coordenats. Tots tres cubs fan pila, _es a dir, les cares corresponents es toquen i comparteixen el mateix eix vertical que passa pel seu centre.

Es demana que escriguis una funció `PintaEscena()` que tingui el codi que cal per a pintar aquest pilar de cubs a partir de la funció `PintaCub()`. Considera que la càmera ja està completament inicialitzada. Justifica la resposta en base a la transformació geomètrica que cal aplicar a `PintaCub()` per a pintar cadascun dels tres cubs.



20. (2014-2015 P1Q) Què canviaries del codi de la teva resposta a la anterior (funció `PintaEscena()`) per a què l'escena inicial sigui la de la figura següent? Justifica la resposta.



21. (2014-2015P 2Q) Els mètodes `pintaVaca()` i `pintaPatricio()` envien a pintar la geometria dels models d'una vaca i d'un Patricio. Les capses mínimes contenidores d'aquests models estan determinades pels punts: (`vacaxmin`, `vacaymin`, `vacazmin`, `vacaxmax`, `vacaymax`, `vacazmax`) i (`Patxmin`, `Patymin`, `Patzmin`, `Patxmax`, `Patymax`, `Patzmax`). El cap de la vaca mira cap a les X+ i el Patricio mira cap a les Z+.

Es vol visualitzar una escena formada per una vaca amb un Patricio a sobre. Ambdós models mirant cap a les Z+; el centre de la base de la capsa de la vaca ha d'estar situat al punt (10,0,0) i les seva `al_cada` ha de ser 2; el centre de la base de la capsa del Patricio ha d'estar al punt (10,2,0) i la seva `alçada` ha de ser 1.

Suposant que la càmera està correctament inicialitzada (no cal que digueu res sobre la càmera), es demana que indiqueu, justificant les respostes:

- l'expressió de les TGs per ubicar cada model al lloc indicat de l'escena;
- el pseudocodi d'una funció `pintaEscena()` que pinti l'escena tal com s'ha descrit, calculant les TGs indicades i utilitzant `pintaVaca()` i `pintaPatricio()`.

22. (2014-2015P 2Q) Imagina que tenim l'escena de la vaca + Patricio de la pregunta 21 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> <p>AUX= Rotate(alfa,0,1,0)</p> <p>TG1= AUX*TG1</p> <p>TG2= AUX*TG2</p> <p>modelMatrix(TG1)</p> <p>pintaVaca()</p> <p>modelMatrix(TG2)</p> <p>pintaPatricio()</p>	<p>b)</p> <p>modelMatrix(TG1)</p> <p>pintaVaca()</p> <p>Rotate (alfa,0,1,0)</p> <p>modelMatrix(TG2)</p> <p>pintaPatricio()</p> <p>Rotate (alfa,0,1,0)</p>
<p>c)</p> <p>AUX= Rotate(alfa, 0,1,0)</p> <p>TG1=TG1*AUX</p> <p>modelMatrix(TG1)</p> <p>pintaVaca()</p> <p>TG2=TG1*TG2</p> <p>modelMatrix(TG2)</p> <p>pintaPatricio()</p>	<p>d)</p> <p>AUX= Rotate(alfa, 0,1,0)</p> <p>TG1=AUX*TG1</p> <p>modelMatrix(TG1)</p> <p>TG2=AUX*TG2</p> <p>modelMatrix(TG2)</p> <p>pintaVaca()</p> <p>pintaPatricio()</p>

23. (2015-2016 Q1) Els mètodes `pintaPatricio()` i `pintaTaula()` envien a pintar la geometria dels models d'un Patricio i d'una taula. Les capses mínimes contenidores d'aquests models estan determinades pels punts: (`Patxmin`, `Patymin`, `Patzmin`), (`Patxmax`, `Patymax`, `Patzmax`) i (`taulaxmin`, `taulaymin`, `taulazmin`), (`taulaxmax`, `taulaymax`, `taulazmax`). El Patricio mira cap a les Z+ i la taula està inicialment orientada de manera que les seves potes estan recolzades sobre el pla XZ (la taula està "dreta").

Es vol visualitzar una escena formada per una taula amb dos Patricios al seu voltant. La taula tindrà mida 4 en X, escalada uniformement, i estarà amb el centre de la seva

base al punt (5, 0, 5) i els Patricios estaran situats al voltant de la taula (mirant cap a ella) de manera que el primer estarà mirant en direcció de les X- i l'altra en direcció de les X+. Els Patricios un cop ubicats han de fer mida 1 en X de l'escena. El primer Patricio haurà d'estar situat amb el centre de la seva base al punt (8, 0, 5), i el segon amb el centre de la seva base al punt (2, 0, 5).

Suposant que la càmera està correctament inicialitzada (no cal que diguis res sobre la càmera), es demana que indiquis el pseudocodi d'una funció `pintaEscena()` que pinti l'escena tal com s'ha descrit, especificant com es troben les TGs indicades i utilitzant els mètodes `pintaTaula()` i `pintaPatricio()`. Justifica la resposta.

24. (2015-2016 Q1) 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_es 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));`

25. (2015-2016P 2Q) Es vol visualitzar una escena formada per dos objectes:

- Un prisma recte de base quadrada de costat 2 i alçada 5, situat amb el centre de la seva base al punt (0,0,0). Disposem del mètode `pintaCub()` que envia a pintar la geometria del model d'un cub de costat 1 centrat a l'origen de coordenades.
- Un Patricio d'alçada 2 situat al damunt de l'objecte anterior, és a dir, la base de la capsa del Patricio ha d'estar sobre la cara superior del paral·lelepípede i mirant cap a les X+.

Disposem del mètode `pintaPatricio()`, que pinta un Patricio que mira cap a les Z+ i la seva capsa contenidora està determinada pels punts: (Patxmin, Patymin, Patzmin), (Patxmax, Patymax, Patzmax). Suposant que la càmera està correctament inicialitzada, respon justificadament als següents apartats:

- a) Indica el pseudocodi d'una funció `pintaEscena()` que permeti visualitzar l'escena descrita especificant el codi (o pseudocodi) que permet trobar les TGs requerides per a cada objecte i utilitzant els mètodes `pintaCub()` i `pintaPatricio()`.
- b) Què hi afegiries i on a la funció `pintaEscena()` de l'apartat anterior per a què tota l'escena quedi finalment traslladada segons el vector (tx, 0, tz)?

26. (2015-2016P 2Q) Tenim un objecte al que se li ha aplicat la següent transformació de model (TG) per a ubicar-lo a l'escena (C és el centre de l'objecte):

```
TG = I;
TG = TG*Translacio (3,0,3);
TG = TG*Rotacio_z (-90);
TG = TG*Rotacio_x (90);
TG = TG*Rotacio_y (90);
TG = TG*Escala (1/mida,1/mida,1/mida);
TG = TG*Translacio (-C.x,-C.y,-C.z);
```

Indica amb quina de les següents transformacions aconseguir__em l'objecte centrat al mateix punt i orientat de la mateixa manera per_o essent el doble de gran.

- a) `TG = I;`**
- `TG = TG*Translacio (3,0,3);`**
- `TG = TG*Rotacio_x (90);`**
- `TG = TG*Escala (2/mida,2/mida,2/mida);`**
- `TG = TG*Translacio (-C.x,-C.y,-C.z);`**

b) $TG = I$;
 $TG = TG * \text{Translacio}(-3, 0, -3)$;
 $TG = TG * \text{Rotacio}_z(90)$;
 $TG = TG * \text{Rotacio}_x(90)$;
 $TG = TG * \text{Rotacio}_y(-90)$;
 $TG = TG * \text{Escala}(2/\text{mida}, 2/\text{mida}, 2/\text{mida})$;
 $TG = TG * \text{Translacio}(-C.x, -C.y, -C.z)$;

c) $TG = I$;
 $TG = TG * \text{Translacio}(3, 0, 3)$;
 $TG = TG * \text{Rotacio}_z(-90)$;
 $TG = TG * \text{Rotacio}_x(90)$;
 $TG = TG * \text{Rotacio}_y(90)$;
 $TG = TG * \text{Translacio}(-C.x, -C.y, -C.z)$;
 $TG = TG * \text{Escala}(2/\text{mida}, 2/\text{mida}, 2/\text{mida})$;

d) Cap de les altres és correcte.

27. (2016-2017P Q1) El mètode `pintaModel()` envia a pintar un model amb la geometria d'un Patricio, que té com a caps mínima contenidora els punts (Xmin,Ymin,Zmin) i (Xmax,Ymax,Zmax) i que inicialment té el davant mirant cap a les Z+. Es vol pintar una escena formada per dos Patricios situats formant un pilar. El primer Patricio (que li diem Pat1) ha de fer d'alçada 3 (un cop escalat uniformement) i es troba amb el centre de la base de la capsa contenidora al punt (5,0,0) i mirant cap a les Z+. I el segon Patricio (que li diem Pat2) ha de fer d'alçada 2 (també un cop escalat uniformement) i es troba amb el centre de la base de la seva capsa coincidint amb el centre de la cara del damunt de la capsa del primer Patricio (és a dir el petit "damunt" del gran) i mirant cap a les Z-.

- a) Indica quin seria el pseudocodi d'una funció `pintaEscena()` que permeti enviar a visualitzar l'escena descrita especificant el codi (o pseudocodi) que permet trobar les TGs requerides per a cada Patricio i utilitzant el mètode `pintaModel()`.
b) Indica clarament què modificaries del codi anterior per a què donat un vector (tx,ty,tz) el pilar dels dos Patricios es traslladi sencer respecte a aquest vector. Pots suposar que en altres parts del codi aquest vector es modifica adientment (per exemple, en prèmer certes tecles).

28. (2016-2017P Q1) Tenim una escena amb un terra i un homer que es troba damunt del terra i volem moure el homer sobre el terra com si caminés.

a) Caldrà declarar dos uniforms per a les dues matrius de TG en el Vertex Shader (TG1 i TG2), i mutiplicar per una o l'altra depenent del VAO que estem pintant.

b) Si tenim un mètode `modelTransform()` que calcula la matriu de translació del homer i l'envia al Vertex Shader, el pseudocodi del `pintaEscena()` seria:

```
pintaEscena() {
    modelTransform();
    pintaHomer();
    pintaTerra();
}
```

c) Si tenim un únic mètode `modelTransform()` que calcula la matriu de translació del homer i l'envia al Vertex Shader, el pseudocodi del `pintaEscena()` seria:

```
pintaEscena() {
    pintaTerra();
}
```

```

    modelTransform();
    pintaHomer();
}

```

d) Cap de les altres respostes és correcta.

29. (2018-2019P Q1) Volem muntar una escena posant un gerro sobre una taula. La taula no necessita cap transformació geomètrica per a situar-la a l'escena, però el gerro el volem uniformement escalat al doble de gran del que és originalment i situat amb la seva base centrada al punt (10, 5, 0). Tenint en compte que els punts mínim i màxim de la capsa contenidora del model del gerro són: (xmin, ymin, zmin) i (xmax, ymax, zmax), completa la funció següent que calcula i retorna la TG necessària per a pintar a que aquest gerro a la nostra escena:

```

    tg_gerro () {
        TG = I;
        ...
        ...
        return (TG); }

```

30. (2018-2019P Q1) Una escena formada per tres objectes es pinta usant el mètode següent:

```

1- pinta_Escena () {
2- modelMatrix (TG1);
3- pinta_obj1 ();
4- modelMatrix (TG2);
5- pinta_obj2 ();
6- modelMatrix (TG3);
7- pinta_obj3 ();
8- }

```

Suposant que TG1, TG2 i TG3 són variables globals que contenen les matrius de transformació que necessita cada objecte respectivament i que la crida modelMatrix(TG) només envia la matriu a la tarja gràfica, què afegiries i on en aquesta rutina pinta Escena() per a que, donat un cert vector de translació (tx,ty,tz), en coordenades de l'aplicació, tota l'escena es traslladi segons aquest vector de translació

31. (2018-2019P Q1) Un objecte està modelat per un conjunt de triangles:
- a) L'esquema de representació del model pot indicar la topologia (ordre d'unió dels vèrtexs) de manera implícita o explícita.
 - b) L'esquema de representació ha de ser sempre una única taula de vèrtexs sense repetició.
 - c) Per intercanviar la informació amb OpenGL utilitzant VBOs (com fem al laboratori) cal que no hi hagi repetició de vèrtexs i que la topologia (ordre d'unió dels vèrtexs) sigui explícita.
 - d) L'esquema de representació ha d'indicar sempre la topologia (ordre d'unió dels vèrtexs) de manera implícita.

Solucions

3. `TG= Translate(0,0.5,0);`
`ModelMatrix(TG);`
`dibuixaCub();`
`TG=Translate(0.75, 0.25,0);`
`TG=TG*Scale(0.5,0.5,0.5);`
`ModelMatrix(TG);`
`dibuixaCub();`
`TG= Translate(1.125,0.125,0);`
`TG=TG*Scale(0.25,0.25,0.25);`
`ModelMatrix(TG);`
`dibuixaCub();`
5. Per a ubicar i escalar l'objecte adequadament, traslladarem el centre de la base de la capsula contenidora a l'origen de coordenades, seguidament realitzarem un escalat uniforme respecte l'origen de valor $AmpleX/(MaxX-MinX)$ i, per últim, traslladarem l'objecte a la posició final (com que tenim el centre de la base de la capsula a l'origen i aquest és el punt que volem situar al punt donat, directament farem la translació al punt donat).
- a) Si considerem $escalat = AmpleX/(MaxX-MinX)$ tenim que la composició de transformacions és:
 $TG = T(10, 0, 10) * S(escalat, escalat, escalat) * T(-(MaxX+MinX)/2, -MinY, -(MaxZ+MinZ)/2)$
- b) Suposem que el contingut de la matriu `ModelView` en aquest punt és la matriu que permet portar els vèrtexs del sistema de coordenades de l'aplicació al sistema de coordenades de l'observador, és a dir, en aquest punt del codi la matriu `ModelView` conté la transformació de coordenades:

```
escalat = AmpleX/(MaxX-MinX);
TG=Translate (10, 0, 10);
TG=TG*Scale(escalat, escalat, escalat);
TG=TG*Translate (-(MaxX+MinX)/2, -MinY, -(MaxZ+MinZ)/2);
ModelMatrix(TG);
PintaObjecte ();
```

21. La vaca s'ha de rotar -90 graus respecte l'eix Y i s'ha d'escalar uniformement per a què la seva alçada sigui 2. El factor d'escala és doncs $2.0/(vacaymax-vacaymin)$ ($escvaca = 2.0/(vacaymax-vacaymin)$). Abans, però, s'ha de portar la vaca a l'origen, per tant portarem a l'origen el centre de la base de la capsula, que és el punt que després hem de traslladar finalment al punt (10, 0, 0).
 $cbvaca = ((vacaxmin+vacaxmax)/2.0, vacaymin, (vacazmin+vacazmax)/2.0)$.
El Patricio no s'ha de rotar, per tant apart de la translació inicial del centre de la base a l'origen ($cbpatr = ((Patxmin+Patxmax)/2.0, Patymin, (Patzmin+Patzmax)/2.0)$) i la translació final al punt (10, 2, 0), també caldrà escalar-lo per a què la seva alçada faci 1, és a dir factor d'escala $1.0/(Patymax-Patymin)$ ($escpatr = 1.0/(Patymax-Patymin)$).

Així doncs, les respostes a aquest exercici són:

- a) Expressió de les TGs de la vaca (TGV) i el Patricio (TGP):

$TGV = T(10; 0; 0) * Ry(-90) * S(escvaca; escvaca; escvaca) * T(-cbvaca)$

$TGP = T(10; 2; 0) * S(escpatr; escpatr; escpatr) * T(-cbpatr)$

- b) La rutina `pintaEscena()` en pseudocodi serà:

```
pintaEscena () {
    TGV = Translate (10,0,0);
    TGV = TGV * Rotate (-90, 0,1,0);
    TGV = TGV * Scale (escvaca,escvaca,escvaca);
    TGV = TGV * Translate (-cbvaca);
    ModelMatrix (TGV);
    pintaVaca();
    TGP = Translate (10,2,0);
    TGP = TGP * Scale (escpatr,escpatr,escpatr);
    TGP = TGP * Translate (-cbpatr);
    ModelMatrix (TGP);
    pintaPatricio();
}
```

