VIG - EXAMEN FINAL – Quadrimestre de tardor – 2007-2008 – **Temps: 2h**

Cognoms: Nom:

1. (1 punt) Dos estudiants volen obtenir una visualització en planta d'una escena que té el seu centre en l'origen de coordenades. Un d'ells utilitza el tros de codi *codi-1* i l'altre el *codi-2* per a ubicar la càmera. Creus que obtindran la mateixa imatge? Per què? Justifiqueu la resposta.

```
glMatrixMode (GL_PROJECTION);
glLoadIdentity();
glOrtho (-20,20,-20,20,5,15);
glMatrixMode (GL_MODELVIEW);
glLoadIdentity();
gluLookAt(0,10,0,0,0,0,0,0,-1);
```

codi-1 (càmera_1)

```
glMatrixMode (GL_PROJECTION);
glLoadIdentity();
glOrtho (-20,20,-20,20,5,15);
glMatrixMode (GL_MODELVIEW);
glLoadIdentity();
glTranslatef (0,0,-10);
glRotatef(90,1,0,0);
```

codi-2 (càmera_2)

La matriu de projecció (TP) es construeix de la mateixa manera en tots dos codis i per tant és la mateixa.

El posicionament de la càmera (matriu TC) es construeix de forma diferent en els dos codis:

a) En el codi-1, l'observador es situa a la posició (0,10,0), el VRP a l'origen de coordenades i el vector up en la direcció negativa de l'eix Z, per tant estem definint un sistema de coordenades d'observador amb l'origen en el punt (0,10,0) i eixos amb la següent orientació: X_O igual que X_A ; Y_O la de $-Z_A$; Z_O la de Y_A .

b) En el codi-2, les transformacions a fer al sistema de coordenades (a la càmera de referència) són: un gir sobre l'eix X de -90 graus seguit d'una translació de 10 unitats sobre l'eix Z resultant. El gir ens deixa el sistema de coordenades orientat tal que: $X' = X_A$; $Y' = -Z_A$; $Z' = Y_A$. Després la translació, símplement modifica la posició de la càmera sobre l'eix Z' (Y_A) aplicant la translació (0,0,10); de manera que l' origen del sistema de coordenades resultant estarà en (0,10,0). El resultat final de les dues transformacions ens defineix el mateix sistema de coordenades d'observador que el codi-1.

Com que els dos codis ens situen la càmera al mateix lloc i orientada de la mateixa manera i amb la mateixa projecció, és evident que tots dos codis obtindran la mateixa imatge.

2. (1 punt) Tenim el punt (10,0,0) en coordenades de l'aplicació. Indiqueu les seves coordenades d'observador, normalitzades i de dispositiu en relació amb la *càmera_1* definida en l'exercici anterior. El viewport té una resolució de 1024x1024 píxels. Justifiqueu la resposta.

Seguint l'explicació que hem vist a l'exercici anterior de com queda el SCO, tindrem que el punt (10,0,0) serà el punt (10,0,-10) en el sistema de coordnades d'observador.

En el sistema de coordenades normalitzat, sabem que el volum de visió es normalitza a valors entre -1 i 1 en les tres components. El volum de visió definit (segons la matriu de projecció definida) va de -20 a 20 en l'eix X_O, de -20 a 20 en l'eix Y_O i de -5 a -15 en l'eix Z_O. I per tant, el punt en el sistema de coordenades normalitzat és el (0.5, 0, 0).

Finalment, com que sabem que la vista està definida de 1024 x 1024 píxels, i el punt (0,0,0) del sistema normalitzat és al centre de la vista, tindrem que el punt en coordenades de dispositiu és el (768,512).

3. (1.5 punts) Disposem del model geomètric de les cares i vèrtexs d'un cub amb longitud d'aresta l i ubicat en una posició i orientació arbitrària en l'espai. L'usuari selecciona una de les seves cares (`cara_i`). Descriviu el procediment per a calcular totes les inicialitzacions d'una càmera (posició, orientació, tipus,...) que permeti visualitzar la `cara_i` en pantalla en *verdadera magnitud*; és a dir, que multiplicant per un mateix factor d'escala les seves mides en píxels és corresponguin a les reals. El viewport és de 1024x1024. Suposeu que teniu accés a tota la informació geomètrica de la `cara_i`.

   Usarem una càmera axonomètrica situada en la recta que passa pel punt central de la cara_i en direcció de la normal de la cara. La definició del window (paràmetres de la càmera axonomètrica) haurà de permetre que la cara quedi dins del window. Si es vol veure la cara en verdadera magnitud no pot haver deformació, per tant la relació d'aspecte ha de ser la mateixa que la del viewport, és a dir 1.

   Podem definir una càmera amb els paràmetres següents:

   OBS = C + dist * N,

   on C és el centre de la cara_i que es pot calcular com el punt mig de la seva diagonal i N és el vector normal a la cara_i.

   VRP = C,

   VUV (vector up) = qualsevol vector contingut en el pla de la cara_i (per exemple en la direcció d'una aresta de la cara),

   Left = -l/2, Right = l/2, Bottom = -l/2, Top = l/2

   0 < Znear < dist, Zfar > dist (perquè només cal que garantim que es vegi la cara).

4. (1 punt) En la pràctica seleccioneu (enceneu) els 4 fanals més propers a la posició del vehicle que estan dins del seu volum de visió (càmera en primera persona). Què modificarieu del procés de selecció de fanals per a que <u>només</u> es seleccionin aquell/s fanal/s que, quan es visualitza/en (<u>amb la càmera que estigui activada</u>), es projecten en el píxel central de la vista? Suposeu un viewport de 600x800 píxels.
   Observació: No cal que expliqueu els canvis en el parsejat del buffer, simplement cal que indiqueu els canvis a introduir en el codi per a que quan envieu a pintar els fanals, el procés de selecció d'OpenGL emmagatzemi en el buffer la informació del/s fanal/s que es volen identificar.

   Com que el que es vol és seleccionar els fanals que es projecten al centre del viewport usant la càmera que estigui activa en cada moment, la primera diferència amb el que s'ha fet a la pràctica és que no cal canviar a la càmera en primera persona per a fer la selecció.

   El que caldrà fer és multiplicar la matriu de projecció activa per la matriu produïda per la crida a `gluPickMatrix`*, que ens reduirà el volum de selecció a la zona del viewport que li definim amb els paràmetres que li passem. Així, doncs, caldrà passar-li a aquesta crida el píxel del centre del volum de selecció (que serà el centre del nostre viewport – (300, 400) –), la mida en píxels que volem per a aquest volum (per exemple 1 x 1) i la matriu de transformació món-dispositiu activa. Així, el codi per a posar la matriu de projecció un cop modificat quedarà de la següent manera:*

```
int vp[4];
glGetIntegerv (G_VIEWPORT, vp);
float proj[16];
glGetFloatv (GL_PROJECTION_MATRIX, proj);
glMatrixMode (GL_PROJECTION);
glLoadIdentity ();
gluPickMatrix (300, 400, 1, 1, vp);
glMultMatrix (proj);
```

5. (1 punt) Una escena està formada per un conjunt d'objectes. Hi ha dos focus de llum encesos. Un està ubicat en la posició (`F1.x,F1.y,F1.z`) en coordenades de l'aplicació. L'altre es mou per l'escena i es disposa d'una funció `PosicioFocus(t,F2.x,F2.y,F2.z)` que retorna la seva posició en coordenades de l'aplicació en un instant determinat `t`. La càmera està ubicada i orientada segons `(OBS1,VRP1,up1)`. Podeu suposar que existeix una acció `pinta_escena()` encarregada de pintar l'escena. Escriviu i justifiqueu el tros de codi OpenGL requerit per a la ubicació de les posicions de les llums i de la càmara, així com la crida a l'acció de pintat. No cal que definiu el tipus de càmera.

Per a tots dos focus disposem de la seva posició fixa respecte del sistema de coordenades de l'aplicació, per tant, les crides a `glLight` que s'encarreguen de posicionar-los han d'estar situades en el codi de manera que la matriu MODELVIEW activa sigui la mateixa que afecta als vèrtexs de l'aplicació (és a dir, la matriu que fa la transformació de coordenades). Per tant, el tros de codi requerit per a posicionar la càmera i els focus de llum serà el següent:

```
// definim el posicionament de la càmera
glMatrixMode (GL_MODELVIEW);
glLoadIdentity ();
gluLookAt (OBS1.x, OBS1.y, OBS1.z, VRP1.x, VRP1.y, VRP1.z,
           up1.x, up1.y, up1.z);
// ara posicionem els focus de llum
float posf1[4] = {F1.x, F1.y, F1.z, 1};
// suposem el focus 1 en el GL_LIGHT0
glLightfv (GL_LIGHT0, GL_POSITION, posf1);
PosicioFocus (t, F2.x, F2.y, F2.z);  // suposem t coneguda
float posf2[4] = {F2.x, F2.y, F2.z, 1};
// suposem el focus 2 en el GL_LIGHT1
glLightfv (GL_LIGHT1, GL_POSITION, posf2);
// pintem l'escena
pinta_escena () ;
```

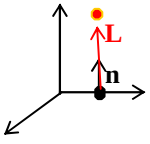
6. (1.5 punts) Suposeu que tenim un triangle de vèrtexs V1=(2,0,0), V2=(-2,0,-2) i V3=(-2,0,2) i volem calcular la il·luminació que fa en ell un focus de llum puntual de color blanc situat a la posició (2,4,0). Suposant que usem el model d'il·luminació de Phong, que l'observador es troba a la posició (2,4,0) i que el material del triangle té propietats ka=(0,0,0), kd=(0,0,0.8), ks=(0.8,0.8,0.8) i n=100, indiqueu, justificant la resposta, com es veurà pintat el triangle a la vista en els següents casos (no es necessari que realitzeu càlculs explícits, simplement raoneu el resultat):

a) La normal als tres vèrtexs és la normal del triangle. La cara mira cap al semiespai on es troba el focus de llum i s'usa colorat (shading) constant.
b) La normal és diferent per a cada vèrtex. La normal al vèrtex V1 és (0,1,0).
La normal als vèrtexs V2 i V3 ès (-1,1,0). S'usa colorat de Gouraud.

Si tenim shading constant tot el triangle és veura del mateix color. Aquest color serà el d'un dels seus vèrtexs. Per exemple, si triem el vèrtex V1, fàcilment es pot comprobar que l'angle "fita" entre la direcció L (vector director de la recta que uneix V1 amb el focus) de la llum incident en V1 i la normal del triangle en V1 és de 0º, per tant, pel càlcul del color (segons model de Phong) cal tenir en consideració tant la reflecció difusa com l'especular. L'angle "fi" entre la llum reflectida de manera especular i la direcció de visió del punt és també de 0º. No caldrà considerar la reflexió de la llum ambient perquè la constant ka és (0,0,0). Per tant, el color del vèrtex V1 (i de tot el triangle) serà:

$I_\lambda(V1) = (0,0,0.8)(1,1,1).\cos(0) + (0.8,0.8,0.8)(1,1,1)\cos(0)^{100} => (0,8, 0.8, 1)$

que és un color, pràcticament, blanc.



Si tenim shading de Gouraud, el color dels píxels en que es projecta el triangle es calcula interpolant el color dels vèrtexs del triangle en funció de la seva distància als píxels en que es projecten els vèrtexs. El color del V1 és el mateix que hem calculat anteriorment (pràcticament blanc) perquè no s'han modificat ni les seves condicions d'il·luminació ni la seva normal. Les normals dels vèrtexs V2 i V3 forman ara una angle"fita" de 90º amb la direcció d'incidència de la llum; per tant, cos(fita)=0 i el punt no reflecteix ni llum difusa ni llum especular (la llum no afecta als vèrtexs). Com ka=(0,0,0), tampoc tindran reflexió de llum ambient; per tant, el color de V2 i V3 serà (0,0,0), és a dir, negre. En conseqüència el triangle es veurà de tonts grisos: negre en els píxels propers a les projeccions de V2 i V3 i quasi blanc en els propers a V1.

7. (1 punt) Un estudiant ha implementat la il·luminació de la pràctica 2. Està activat <u>només</u> un llum d'escena però per error l'ha col·locat DINS d'un dels edificis. Els edificis són objectes convexos tancats. En visualitzar l'escena des d'una càmera exterior a l'edifici, de quin color es veurà aquest edifici? De quin color es veuran la resta d'edificis? Raoneu la resposta.

   Com que els edificis són convexos i tancats, l'edifici que té el focus dins tindrà il·luminades la part interior de les seves cares.Per tant, la seva part exterior –visible per l'observador- no estarà il·luminada i aquest edifici es veurà negre (si considerem que no hi ha llum ambient). Més formalment, l'angle "fita" entre la normal de les cares (part externa) i la direcció d'incidència de la llum en elles serà superior a 90º i, per tant, la reflexió de llum difusa i especular en els vèrtexs de la cara és nul·la.

   Les cares de la resta d'edificis és veuran del color que els hi correspongui segons estiguin il·luminades o no. Aquesta condició es compleix si l'angle fita entre la llum incident al vèrtexs de les cares i la seva normal sigui $|fita| \leq 90º$.

8. (1 punt) Els models empírics d'il·luminació d'OpenGL tenen certes limitacions quan calculen la il·luminació d'una escena. Per exemple, no es contemplen ombres ni miralls. A què és degut? Tanmateix, en les visualitzacions podem observar algunes cares més fosques, és a dir, no els arriba la llum del focus. Com és possible tenir aquest efecte si s'utilitzen models empírics? Justifiqueu la resposta.

   Els models d'il·luminació empírics per al càlcul del color en un punt tant sols tenen en consideració la llum incident del focus en el punt. Com no tenen en compte que pugui arribar llum reflectida d'altres objectes, no poden simular miralls. Com que tampoc consideren que entre el focus de llum i el punt de càlcul de color pugui existir altre objecte, tampoc poden considerar ombres.

   Tanmateix, en els models empírics es té en consideració l'angle entre la direcció de la llum incident a un cert punt i la normal de la superfície en ell. Si aquest angle és superior a 90º, vol dir que la llum no il·lumina la part externa de la cara i que és nul·la la contribució al color del punt tant per a la reflexió difusa com per a l'especular. Això provoca que algunes cares només es vegin il·luminades per la llum ambient i es vegin més fosques que d'altres i pugui semblar que estan a l'ombra. A aquestes ombres se les denomina "ombres pròpies".

9. (1 punt) Indiqueu la codificació en RGB, CMYK i HSB d'un mateix color pur (que no sigui el negre). Descriu el color que estas codificant. *Observació*: un color pur és aquell que no conté "gris".

   En RBG si una llum no conté gris vol dir que al menys una de les seves tres components és nul·la. Un possible color seria el vermell (1,0,0).

   En CMY aquest color seria (1-r, 1-g, 1-b) = (0,1,1) i en CMYK (0,1,1,0).

   En HSB, el Hue seria 0º per ser vermell, la saturació S=1 per ser un color pur i la brillantor B=1 perquè tenim el màxim de llum; per tant, en HSB el vermell és (0,1,1).