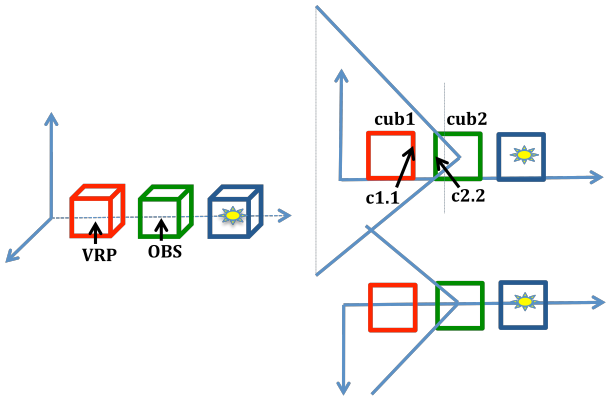
IDI – EXAMEN FINAL. Quadrimestre tardor 2011-2012. **Temps: 2h**

Cognoms: Nom:

1. (2 Punts) Tenim una escena formada per 3 cubs sòlids de longitud d'aresta 2, centres de les seves bases en els punts (2,0,0), (5,0,0) i (8,0,0), cares paral·leles als plans de coordenades i d'un material mate de color vermell, verd i blau respectivament. Situem un focus puntual blanc en el punt (8,1,0), l'observador en (5,1,0), el VRP en (0,1,0), up (0,1,0) i una càmera perspectiva amb relació d'aspecte 1, zN= 0.5, zF=6 i FOV=90º. Quines cares dels cubs seran visibles en el *viewport* i de quin color es veuran si es pinta l'escena utilitzant OpenGL si per eliminar parts amagades s'utilitza:
a) Només el *back-face culling*
b) Activat només el *zbuffer*



La posició dels cubs, llum i càmera es mostra en la figura adjunta. Noteu que l'observador es troba al centre del cub c2 mirant cap el centre del cub c1 i que tant el cub1 com la cara c2_2 es troben dins del frustum de la càmera i, per tant, són potencialment visibles. Donada la posició de la llum (centre del cub c3), i com OpenGL utilitza models empírics locals, d'aquestes cares només estarà il·luminada la cara del c1 amb normal segons l'eix x positiu, a la resta els arriba la llum pel darrera i, per tant, no serà considerada en el càlcul d'il·luminació.

a) Aquelles cares respecte a les quals l'observador es troba en el semi-espai interior seran eliminades per back-face culling. Per tant, només quedarà visible la cara c1.1 que es pintarà de color vermell donat que la llum és blanca. A més el seu color serà constant perquè en tots els vèrtexs l'angle entre la direcció incidència de la llum i la seva normal és el mateix (no hi ha reflexió especular perquè ens diuen que és un material mate).

b) En aquest cas, la cara visible serà la més propera a l'observador: c2.2. Com la llum li arriba per la banda interior no es considerarà en el càlcul del color; i, per tant, aquesta cara es veurà negra.

2. (2 Punts) Un joc per computador vol simular un helicòpter amb un llum (centrat en ell) que segueix un cotxe que es mou per una ciutat (el focus de l'helicòpter il·lumina sempre al cotxe). Suposant que tenim les crides següents: `posicio_cotxe(float& x, float& y, float& z)` que retorna en coordenades d'aplicació la posició del cotxe en un instant; `posicio_helicopter(float& xh, float& yh, float& zh)` que retorna la posició de l'helicòpter; `pinta_cotxe()` que pinta el cotxe centrat a l'origen de

coordenades; `pinta_helicopter()` que pinta l'helicòpter centrat a l'origen de coordenades; i `set_camera()` que contínuament actualitza la càmera per veure correctament en tercera persona l'escena. Es demana que doneu el tros de codi que pinta aquesta escena suposant que la llum de l'helicòpter és puntual i de tipus SPOT. A l'hora de pintar, no us preocupeu per les rotacions.

```
// els altres paràmetres de la llum no es modifiquen en cada frame
float x, y, z, xh, yh, zh;
set_camera();
glMatrixModel (GL_MODELVIEW);
posicio_cotxe(x,y,x);
posicio_helicopter(xh,yh,zh);
glPushMatrix();
glLight (GL_LIGHT0, GL_POSITION,  (xh,yh,zh,1));
glLight (GL_LIGHT0, GL_SPOT_DIRECTION, (x-xh,y-yh,z-zh));
glTranslate (x,y,z);
pinta_cotxe();
glPopMatrix();
glPushMatrix();
glTranslate(xh,yh,zh);
pinta_helicopter();
glPopMatrix();
```

3. (2 Punts) Tenim el model geomètric d'un cub de costat 4, centrat en el punt (2,2,2) amb les cares paral·leles als plans de coordenades. Tenint en compte que no usem il·luminació i els objectes es pinten totes amb el mateix color (*glColor(...)*), indica TOTS els paràmetres d'una càmera axonomètrica que genera com imatge del cub un hexàgon regular. La vista és de 512x512 píxels. Justifica la resposta. Doneu dues inicialitzacions alternatives de la *ModelView*: una amb transformacions geomètriques i altre usant *gluLookAt(..)*.

Per a obtenir aquesta visualització, les úniques restriccions són que la direcció de visió coincideixi amb una de les diagonals principals del cub i que el window permeti veure el cub des d'aquesta posició.

Posició càmera:
(a) **VRP**= (0,0,0); **OBS** en la direcció de la recta que uneix el **VRP** i **OBS**; **OBS**=(5,5,5) i **up** potser qualsevol vector diferent de (**OBS**-**VRP**); per exemple (0,1,0).
Per tant, podríem inicialitzar la càmera amb: gluLookAt(5,5,5, 0,0,0, 0,1,0)

(b) Una solució anàloga a l'anterior:

```
glMatrixMode (GL_ModelView);
glLoadIdentity();
//noteu que la distància entre OBS i el vèrtex més allunyat és 5sqrt(3);
glTranslate(0,0,-5sqrt(3));
float angle=arctg(1/sqrt(2));  // aproximadament 35.26
glRotate(angle, 1, 0, 0);
glRotate(-45,0,1,0);
```

Cal fer les transformacions necessàries per a ubicar la direcció de visió (diagonal escollida del cub) amb l'eix z de l'aplicació i, posteriorment, desplaçar el cub (que tindrà un vèrtex en l'origen de coordenades) davant de l'observador (en origen de coordenades, mirant en z negatives). Una possible solució:

1. Gir respecte l'eix y de -45º (horari) per ubicar la direcció de visió (diagonal del cub) en el pla y-z

2. Per a fer-la coincidir amb l'eix z cal fer un gir respecte l'eix x en sentit anti-horari. Per a calcular l'angle, noteu que en el pla yz tenim un triangle rectangle resultat del gir d'un triangle format per la diagonal principal del cub inicial (longitud 4sqrt(3)), la seva projecció en el pla xz (longitud 4sqrt(2)) i una aresta del cub (longitud 4); fent trigonometria: angle=artg(4/(4*sqrt(2)).
3. La translació, ha de ser (0,0,-5sqrt(3)

Respecte a l'òptica pel posionament anterior: zN= 0.5 (menor que la distància de l'observador al primer vèrtex); zF= 10 (superior a la distància de l'observador al vèrtex més llunyà 5sqrt(3))
La relació d'aspecte del window ha de ser la del viewport ==> 1
El window ha de permetre veure tot el cub, el radi de l'esfera contenidora del cub és 4sqrt(3)/2; per tant, per asegurar veure el cub, aquesta ha de ser la longitud de les seves arestes: window (-2sqrt(3), 2sqrt(3), -2sqrt(3), 2sqrt(3))

4. (1 Punt) Quines constants de material definiries si es vol que un objecte sigui de plàstic polit de color vermell? Raona la resposta.

Per ser polit tindrà reflexió especular alta; per tant, uns possibles valors de shininess i ks_λ són: N=100 i ks_λ =(0.8,0.8,0.8) (color de la taca especular serà el de la llum).
El color difús ha de ser vermell; per tant, per exemple, kd_λ =(0.5,0,0) i constant de reflexió ambient també vermella però d'un valor inferior ka_λ = (0.2,0,0).

5. (1 Punt) Per què diem que utilitzem models d'il·luminació locals en OpenGL? Quines limitacions tenen en quant al realisme de l'escena?

Aquests models, en el càlcul del color de cada vèrtex, només tenen en compte la **posició de l'observador**, la **posició del focus**, la **posició** i la **normal del vèrtex** i les **propietats del material**; no es consideren ni la resta d'objectes de l'escena, ni les interaccions de la llum entre els objectes. Això es fa per qüestió d'eficiència i facilitat de càlcul. Les limitacions que ens podem trobar, per exemple, és que **no produeïxen ombres ni miralls**.

6. (2 punts)
Un estudiant fa la següent afirmació en un examen:
"La llei de Fitts no es pot emprar per ajudar al disseny d'interfícies en dispositius tàctils perquè en aquest tipus de dispositius no tenim cursor."
Raona si és certa o falsa.

La llei de Fitts mesura l'esforç que s'ha de fer per tal de seleccionar o assenyalar un element en la pantalla com a funció de la distància a recórrer pel cursor i la mida de l'objectiu. A més, la llei de Fitts utilitza dues constants que depenen del dispositiu per assenyalar i que tenen a veure amb la velocitat de desplaçament i la de posada en marxa.

L'estudiant té raó en què en els dispositius tàctils no hi ha cursor, per aquesta raó, no podem calcular la distància que s'ha de moure el cursor. Però l'element que emprem per a assenyalar és el nostre dit, o un apuntador en forma de llapis, de forma que sí que podem mesurar la distància que cal recórrer per a assenyalar un objectiu en la pantalla i d'aquesta forma podríem utilitzar la llei de Fitts per a ajudar-nos a dissenyar els elements de la interfície (per exemple posant més a prop els elements més freqüents i utilitzant mides raonables per facilitar la selecció), sobretot en el cas de la interacció que requereix posar el dit sobre la pantalla i després arrossegar per a seleccionar un objecte o posició ("lift-off"). Un exemple clar són els menús semi-circulars del navegador d'Android per a *tablets*.

Una discussió més completa de la resposta podria afegir els següents detalls:

Per a ser més concrets, hauríem de distingir dos casos: 1) Interacció que requereix un primer "clic" o posar el dit sobre la pantalla i després arrossegar, com en la tècnica del "lift-off". 2) Interacció que realitzem amb el dit (o *stylus*) sense estar reposant sobre la pantalla.
El primer cas és un cas típic on sembla que la llei de Fitts es pot emprar directament, calculant, això sí, les constants *a* i *b* del nostre dispositiu.
El segon cas sembla que també pot beneficiar-se de l'ús de la llei de Fitts, però igual que en el cas del desplaçament en diagonal, caldria adaptar-la per a moviments que no necessàriament seran rectes, sinó que possiblement realitzaran una corba a l'aire. Podem veure que el temps de selecció continua tenint relació amb la disposició dels elements en pantalla i de la seva mida agafant un mòbil amb la nostra mà dominant i mirant de seleccionar un botó molt proper al nostre dit polze, i un punt molt llunyà. En qualsevol cas, si es volen prendre mides exactes de temps de moviment o valorar l'índex de dificultat, en aquest escenari el que s'hauria d'investigar és si la relació exacta continua essent logarítmica o té alguna altra forma (lineal, per exemple).

A banda d'això, la llei de Fitts també té en compte la mida dels objectius. En el cas dels dispositius tàctils també és senzill veure que els objectius petits són més difícils de prémer. També és fàcil fer un experiment comparant el temps que es triga a seleccionar amb un dit comparat amb la selecció amb un *stylus* (utilitzant la mateixa mà, òbviament, així com la mateixa distància a recórrer, etc.), ja que la mida efectiva dels controls de la pantalla es redueixen de forma efectiva quan el nostre dispositiu de selecció és el dit en comptes de quelcom més precís com el llapis o *stylus*.