Gràfics

Practica 0

Xavi Cano
Grup A

PAS 2: Visualització d'un model poligonal

- Des d'on es crida el metode draw()? Per què?

Es crida desde el mètode paintGL() de l'arxiu glwidget.cpp.

Es crida desde paintGL() perque aquest mètode es el que s'encarrega de refrescar la finestra, ja sigui per que una altra finestra la tapa, o per qué hi ha un canvi de refresc, per exemple per que es canvia un angle de visió i per aquest motiu el cub sempre ha de estar pintat.

- Dins de quina classe està definit?

Esta definit dins de la classe cub com a mètode public void(), es a dir, no retorna res.

- Quina diferència hi ha entre el constructor i el mètode draw()? On es defineix la geometria? On es dibuixa? Qui ho envia a la GPU?

La diferencia que hi ha es que el constructor s'encarrega de definir les coordenades del cub i la funció draw() s'encarrega de posar el color a les cares del cub.

- La geometría es defineix amb glBegin().
- Qui ho envia a la GPU es el metode "glEnd() ".

- Quina geometria defineix el cub? Les seves cares estan representades per triangles o per quadrilàters?

La geometría que defineix el cub son quadrats, les seves cares estan representades per quadrilàters.

- Com estan ordenats els punts? Per què?

Els punts estan ordenats per tres eixos(x,y,z) on cada cuatre glVertex3f es una de les cares del cub.

- Com s'aconsegueix que roti el cub?

El cub aconsegueix rotar mitjançant el metode glTranslatef().

PAS 3: Visualització del model reprogramant la GPU

- Que fa el mètode draw()? Des d'un es crida? Per què?

Es crida desde dos metodes, que es troben en glwidget.cpp.

La funció del metode draw() es pintar el cub.

Primer es crida desde *initializeGL*() per definir la seva estructura i després desde *paintGL*() perquè al calcular l'interpolació de colors si no es crida a draw() un altre cop no es veurien els resultats.

- Dins de quina classe està definit?

Esta definit dins de la classe cub com a mètode public void(), es a dir, no retorna res.

- Quina diferència hi ha entre el constructor i el mètode draw()?

La diferencia es que el constructor s'encarrega de centrar el cub, definir el tamany de les arestes, definir els vertex i definir els colors.

En canvi el metode draw() amb l'ajuda del metode glDrawArrays() el que fa es agafar tota aquesta estructura definida abans al constructor per poder dibuixar triangles mitjançant el parametre GL_TRIANGLES.

- On es defineix la geometria? Cada cara del cub està formada per triangles o per quadrilàters? Quin ordre segueixen els punts?

La geometría es defineix al metode make().
Cada cara del cub esta formada per triangles, concretament dos.

L'ordre que segueixen els punts es començant des de la posició zero de l'array fins a arribar a la mida total del vector, que en el nostre cas es 36(6 cares, 2 triangles per cara i 3 vertex per triangle).

L'ordre que segueixen els punts es antihorari.

- On es dibuixa?

Al metode draw().

- Qui ho envia a la GPU?

El metode toGPU().

- Què s'envia a la GPU?

S'envien dades de les posicions dels vèrtex i atributs i en forma de matrius o arrays.

- Quin shader processa els vèrtexs?

El vertexShader.

- Quin shader s'encarrega dels colors?

El fragmentShader.

- Per què surten els colors purs en els vèrtexs i en les arestes i cares surten degradats?

Perque en els vèrtexts s'han definit els colors purs , llavors el procés d'interpolació de sempre va d'un color pur a un altre i visualment aquest canvi de color progresiu es nota a les arestes, que fa un efecte de degradat.

- Com s'aconsegueix que es roti el cub?

S'aconsegueix que roti gracies als metodes *mousePressEvent()* que detecta el clic del mouse i *mouseMoveEvent()* que detecta quan mantenim pulsat el mouse i el movem.

Aquest segon metode a la vegada truca als metodes *setXRotation(),setYRotation(),setZRotation()* que realment son els que s'encarreguen de rotar del cub i després actualitzar-ho amb *update()*.

PAS 4: Visualització del model utilitzant textures a la GPU

- Ha canviat el mètode draw()?

Si.

- On es defineixen les coordenades de textura? Quin mètode les calcula?

Es defineixen en la clase cub com a atribut privat.

El metode que les calcula es el metode make(), que dins seu te altres dos metodes que son *quad()* i *initTextura()* que son els que fan la feina realment.

- Qui les envia a la GPU?

Qui les envia es el metode *toGPU()*.

- Què s'envia a la GPU?

S'envia un VBO (Vertex Buffer Object), es a dir, els punts, colors i les coordenades de la textura s'encapsulen en aquest VBO.

- Quin shader té com a entrada els vèrtexs?

El vertexShader.

- Quin shader té en compte les coordenades de textura?

El fragmentShader.

- Quin shader utilitza la textura?

El fragmentShader.

- Com canviaries la imatge de la textura mapejada al cub?

Doncs aniría al metode *initTextura()* i a l'hora de fer la instancia de la textura canviaria la ruta amb una altre imatge, concretament en aquesta linea de codi:
texture = new QopenGLTexture(QImage("://resources/ImatgeNova.png"));