La cámara. Eventos de ratón

A. Gavilanes
Departamento de Sistemas Informáticos y Computación
Facultad de Informática
Universidad Complutense de Madrid

Objetivo

- Objetivo: realizar movimientos de cámara con el ratón
- Movimientos buscados (ver una demo en la clase grabada)
  - Con el botón derecho: trasladar la escena desde que se pulsa el botón hasta que se suelta, a izquierda/derecha y arriba/abajo
  - Con el botón izquierdo: hacer girar la cámara alrededor de la escena. El efecto es que la escena rota, aunque lo que se mueve es la cámara
  - Con la rueda del ratón:
    - Con la tecla de Control pulsada: escalar la escena
    - Sin la tecla de Control pulsada: aproximar o alejar la escena, según se gire en uno u otro sentido

La clase Camera. Recordatorio de atributos

- Para implementar algunos movimientos se usan los siguientes atributos de la clase **`Camera`**:
  - **`mRadio`**: es el radio de una esfera imaginaria sobre cuya superficie se mueve la cámara
  - **`mAng`**: es la longitud (este-oeste, es decir, distancia angular) a la que se encuentra la cámara sobre esa esfera imaginaria, a contar desde el eje **`X`** positivo, que tiene longitud 0°, medida en sentido antihorario
- Para procurar que el enlace de unos movimientos de la cámara con otros sea coherente (no se produzcan saltos) es preciso mantener actualizados estos dos atributos, después de realizar cualquier movimiento que les afecte
- Estos atributos se suponen inicializados así:

```
GLdouble mAng = -45;          // Longitud 45° oeste
GLdouble mRadio = 1000.0;     // Esfera virtual de radio 1000
```

La clase Camera. Recordatorio de movimientos

- La clase **`Camera`** dispone de los siguientes métodos públicos:
  - **`void moveLR(GLdouble cs)`**: movimiento de la cámara a izquierda/derecha, sobre el eje **`U`**, una distancia **`cs`**, sin cambiar la dirección de vista
  - **`void moveFB(GLdouble cs)`**, **`void moveUD(GLdouble cs)`**: movimientos análogos sobre los ejes **`N`** y **`V`**, respectivamente
  - **`void lookLR(GLdouble cs)`**, **`void lookUD(GLdouble cs)`**: movimientos análogos sobre los ejes **`U`** y **`V`**, respectivamente, pero sin cambiar **`eye`**, solo **`look`**
  - **`void orbit(GLdouble ax)`**: orbita la cámara alrededor de **`look`**, en sentido antihorario, haciendo que **`eye`** describa el paralelo de la esfera de radio **`mRadio`**, que se encuentra a altura **`eye.y`**, incrementando **`mAng`** en la cantidad **`ax`**
  - **`void orbit(GLdouble ax, Gldouble ay)`**: método similar al anterior, pero se deja que la cámara cambie su altura **`eye.y`**, una cantidad **`ay`**. Observa que **`ax`** es un double que incrementa un ángulo mientras que **`ay`** es un double que incrementa una distancia

Programación de los eventos de ratón

- Para programar los eventos de ratón sigue estos pasos
  - Registrar los respectivos callbacks en **`IG1App`**

    `glutMouseFunc(s_mouse)`: para implementar lo que ocurre cuando se presiona o se suelta un botón del ratón

    `glutMotionFunc(s_motion)`: para implementar lo que ocurre cuando se mueve el ratón con un botón presionado

    `glutMouseWheelFunc(s_mouseWheel)`: para implementar lo que ocurre cuando se gira la rueda del ratón

  - Implementar los callbacks estáticos anteriores **`s_...`**

```
static void s_mouse(int button, int state, int x, int y);

static void s_motion(int x, int y);

static void s_mouseWheel(int n, int d, int x, int y);
```

invocando sus respectivos métodos con una instancia estática **`s_ig1app`** de **`IG1App`**

Programación de los eventos de ratón

- Para programar los eventos de ratón sigue estos pasos (continuación)
  - Añadir los siguientes callbacks a **`IG1App`**
    - **`void mouse(int button, int state, int x, int y);`**
      - Se genera cuando se presiona o se suelta un botón del ratón (**`button`**), y recoge en coordenadas de la ventana **`(x, y)`** el momento en que el estado del botón (**`state`**) cambió y pasó a estar pulsado o a estar soltado
    - **`void motion(int x, int y);`**
      - Se genera cuando un botón del ratón se encuentra pulsado y recoge, en coordenadas de la ventana **`(x, y)`**, el lugar donde se soltó
    - **`void mouseWheel(int wheelButtonNumber, int direction, int x, int y);`**
      - Se genera cuando se mueve la rueda del ratón en la **`direction(=+1/-1),`** y recoge en coordenadas de la ventana **`(x, y)`** el momento en que se movió la rueda

Programación de los eventos de ratón

- Declarar dos variables en **IG1App** para guardar las coordenadas del ratón, la última vez que se pulso, y el botón pulsado, la última vez que se hizo:

```
glm::dvec2 mCoord;

int mBot=0;
```

- Las constantes para referirse a los botones del ratón son:

```
GLUT_LEFT_BUTTON/GLUT_RIGHT_BUTTON
```

- Las constantes para referirse al estado del botón, si soltado o pulsado, son, respectivamente:

```
GLUT_UP/GLUT_DOWN
```

- Las constantes para averiguar si ciertas teclas están pulsadas son:

```
GLUT_ACTIVE_CTRL / ..._ALT / ..._SHIFT
```

Para saber cuántas de estas teclas están pulsadas se puede usar:

```
int m = glutGetModifiers();
```

Programación de los eventos de ratón

- Programar **`mouse()`** de forma que se registren los valores lanzados en los atributos mencionados de **`IG1App:`**

```
void mouse(int button, int state, int x, int y) {
```

1. Guarda en **`mBot`** el valor de **`button`**
2. Guarda en **`mCoord`** la posición **`(x, y)`** del ratón

```
}
```

- Recuerda que la variable **`y`** se refiere a coordenadas de ventana y esta tiene su origen en la esquina superior izquierda, mientras que en el puerto de vista el origen está en la esquina inferior izquierda. Puede ser conveniente el paso de una a otra:

```
y(viewport) = glutGet(GLUT_WINDOW_HEIGHT)-y;
```

Programación de los eventos de ratón

- Para programar **`motion()`** recuerda que se quiere implementar el proceso de pulsar un botón del ratón, arrastrar el cursor y soltarlo

`void motion(int x, int y) {`

1. Guarda en una variable auxiliar **`mp`** la diferencia entre **`mCoord`** y **`(x, y)`**
2. Guarda en **`mCoord`** la posición **`(x, y)`** del ratón
3. Si **`mBot`** es el botón izquierdo, la cámara orbita **`(mp.x*0.05, mp.y)`**
4. Si **`mBot`** es el botón derecho, la cámara se desplaza **`moveUD()`** y **`moveLR()`** según indique **`mp`**
5. **`glutPostRedisplay();`**

`}`

- Recuerda que, cuando la cámara orbita, **`mp.x`** hace referencia a una distancia en radianes y, por tanto, es preciso reducirla

Evento de la rueda del ratón

- Programar **`mouse()`** teniendo en cuenta si se tiene pulsada la tecla de Control

`void mouseWheel(int n, int d, int x, int y) {`

1. Averigua el número de teclas pulsadas con **`glutGetModifiers()`**
2. Si no hay ninguna, la cámara se mueve con **`moveFB()`**, según el valor de **`d`**
3. Si está pulsada la tecla Ctrl **`(GLUT_ACTIVE_CTRL)`**, la cámara cambia la escala con **`setScale()`**, según el valor de **`d`**
4. **`glutPostRedisplay();`**

`}`