OPENGL – Interação

Profa. Dra. Regina Célia Coelho

rccoelho@unifesp.br

UNIFESP
UNIVERSIDADE FEDERAL DE SÃO PAULO

Interação com o teclado

- Para interagir usando o teclado temos que incluir a chamada da função **glutKeyboardFunc(func)** na função *main* do programa.
- O parâmetro **func** desta função é o nome da função que irá definir a interação com o teclado.
- A função **func** terá como parâmetro a constante que identifica a tecla pressionada e a posição corrente (x,y) do *mouse* quando a tecla foi pressionada.

Interação com o teclado

- O exemplo abaixo ilustra a estrutura de uma função chamada keyboard que pode mudar entre sombreamento *flat* e *smooth* usando as teclas 'f' ou 'F' (*flat*) e 's' ou 'S' (*smooth*).

```
void keyboard (unsigned char key, int x, int y)
{
  switch (key)
  {
    case 's' :
    case 'S' :
      glShadeModel(GL_SMOOTH) ;
      break ;
    case 'f' :
    case 'F' :
      glShadeModel(GL_FLAT) ;
      break ;
    default :
      break ;
  }
  glutPostRedisplay() ;
}
```

Interação com o teclado

- No exemplo anterior temos a chamada da função glutPostRedisplay().
- Esta função marca a janela corrente como “precisando ser redesenhada”.
- Se a chamada para esta função não for incluída neste ponto, o usuário poderia pressionar teclas sem ver qualquer efeito.
- Isso ocorre porque OpenGL não sabe que o conteúdo da janela muda se o usuário pressionar uma tecla.

Interação com o teclado

- Funções especiais podem ser usadas incluindo a chamada à função ***glutSpecialFunc(func)*** na função *main* do programa.

- Novamente, o parâmetro **func** desta função é o nome da função que irá definir a interação com o teclado.

- Usando teclas especiais no exemplo anterior, teríamos:

Interação com o teclado

```
void SpecialKeys (int key, int x, int y)
{
  switch (key)
  {
    case GLUT_KEY_LEFT :
      glShadeModel(GL_SMOOTH) ;
      break ;
    case GLUT_KEY_RIGHT :
      glShadeModel(GL_FLAT) ;
      break ;
    default :
      break ;
  }
  glutPostRedisplay() ;
}
```

Interação com o *mouse*

- A interação com o *mouse* pode ser feita incluindo a chamada da função **glutMouseFunc(func)** na função *main* do programa.
- Como nas funções anteriores, o parâmetro **func** desta função corresponde ao nome da função que irá definir a interação com o *mouse*.
- A função **func** terá como parâmetro uma constante que identifica qual botão foi pressionado (*left,middle,right*), uma constante que identifica a ação executada sobre o botão (*up* ou *down*) e a posição corrente (x,y) do *mouse* quando o botão foi pressionado.

Interação com o *mouse*

- Utilizando o mesmo exemplo da interação com o teclado, teremos:

```
void MouseInt (int botao, int estado, int x, int y)
{
  switch (botao)
  {
    case GLUT_LEFT_BUTTON :
       if (estado == GLUT_DOWN)
           glShadeModel(GL_SMOOTH) ;
      break ;
    case GLUT_RIGHT_BUTTON :
       if (estado == GLUT_DOWN)
            glShadeModel(GL_FLAT) ;
       break ;
    default :
      break ;
  }
  glutPostRedisplay() ;
}
```

```
#include <stdio.h>
#include <stdlib.h>
#include <GL/glut.h>
void triangulo(){
   glClearColor(0.0, 0.0, 0.0, 0.0);
   glClear(GL_COLOR_BUFFER_BIT);
  glBegin(GL_TRIANGLES);
     glColor3f(1.0, 0.0, 0.0);  glVertex2f(-0.8,-0.8);
     glColor3f(0.0, 1.0, 0.0);  glVertex2f(0.8, -0.8);
     glColor3f(0.0, 0.0, 1.0);  glVertex2f(0.0, 0.8);
    glEnd();
   glFlush();}

void Keyboard (unsigned char key, int x, int y) {
   switch(key) {
      case 's':
      case 'S':  glShadeModel(GL_SMOOTH);
          break;
      case 'f':
      case 'F': glShadeModel(GL_FLAT);
          break; }
   glutPostRedisplay();  }

void SpecialKeys (int key, int x, int y) {
   switch(key) {
      case GLUT_KEY_LEFT:
      case GLUT_KEY_UP: glShadeModel(GL_SMOOTH);
          break;
      case GLUT_KEY_RIGHT:
      case GLUT_KEY_DOWN:  glShadeModel(GL_FLAT);
          break; }
   glutPostRedisplay(); }
```

```
void MouseInt (int botao, int estado, int x, int y) {
   switch(botao) {
      case GLUT_LEFT_BUTTON:
         if(estado == GLUT_DOWN)
            glShadeModel(GL_SMOOTH);
          break;
      case GLUT_RIGHT_BUTTON:
         if(estado == GLUT_DOWN)
             glShadeModel(GL_FLAT);
          break;  }
   glutPostRedisplay(); }

int main(int argc, char *argv[ ] ) {
  glutInit(&argc, argv);
  glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
  glutInitWindowPosition(50,100);
  glutInitWindowSize(600,600);
  glutCreateWindow("Cria um triângulo");
  glutDisplayFunc(triangulo);
  glutKeyboardFunc(Keyboard);
  glutSpecialFunc(SpecialKeys);
  glutMouseFunc(MouseInt);
  glutMainLoop ();
  return 0;
}
```

Interação com o *mouse*

- Capturando os movimentos do mouse sem depender dos botões do mouse:
  - void glutPassiveMotionFunc(void (*func)(int x,int y));
- Uso na main():
  - glutPassiveMotionFunc(func);
- Funciona no momento em que o mouse for movimentado
- Não é necessário estar com nenhum botão clicado.

Exercício

1. Escreva um programa que imprima um ponto azul de tamanho 5.0 na posição em que houver um clique de botão esquerdo do mouse. Inicialmente, o ponto deve ficar nas coordenadas (0,0). (Este exercício será fundamental para aprender: (a) a ajustar as coordenadas do mouse e da janela; (b) a desenhar o ponto na função correta; (c) a guardar as coordenadas lidas). O ponto anterior deverá ser apagado sempre que um novo ponto for traçado.

2. Escreva um programa que realize duas funções: a de imprimir o ponto nas coordenadas do mouse do exercício 1 e de alterar a cor do ponto. Novamente, o ponto inicial é azul e está nas coordenadas (0,0). Para mudar a cor, o usuário digitará as teclas de 0 a 9. Cada tecla deverá ter cores indexadas previamente escolhidas. Escolha as cores que desejar.

Exercício

3. Crie um programa com duas funções: a de imprimir uma linha entre dois pontos quaisquer e a de alterar a cor da linha. Uma linha inicial entre os pontos (0,0) e (0,0.9) azul deve ser traçada. A linha muda de posição de acordo com as coordenadas dadas por um clique inicial e um clique final com o botão esquerdo do mouse. As cores são selecionadas da mesma maneira que no exercício 2.

4. Crie um programa que contenha as mesmas duas funções do exercício 3 para mudar a cor e traçar retas. Desta vez, não existe uma linha inicial, e a cor inicial é a preta (considere fundo branco). A cada dois cliques do botão esquerdo do mouse uma nova linha é traçada, sem perder as anteriores.

Exercício

5. Escreva um programa que contenha três funções: as funções de traçar uma linha e de mudar a cor do exercício 4 e uma função de traçar circunferências. Inicialmente, deve ser impressa uma linha azul de coordenadas (0,0) – (0,0.9). Apenas uma figura deve ser apresentada de cada vez na tela. As figuras anteriores são apagadas. Caso o usuário clique a tecla 'r' ou 'R', a função de traçar retas é ativada. Caso o usuário clique a tecla 'c' ou 'C' a função de traçar circunferências é ativada. O traçado da reta continua da mesma maneira do exercício 3. No traçado de circunferências, o centro da circunferêcia é determinado pelo primeiro clique com o botão esquerdo do mouse e o raio é calculado pela distância do centro à coordenada do segundo clique do botão esquerdo do mouse.

Exercício

- Entregar no Moodle até dia 04/09): Implementar o jogo Pong, em que a "raquete"(reta que defende a bola) deverá se mover de acordo com o mouse. A bolinha deve se mover aleatoriamente na janela (nunca sair dela, a menos pela direita ou esquerda quando não houver defesa). Para selecionar qual raquete deve mexer, analise a direção da bolinha.
- As teclas numéricas (0 a 9) devem mudar a cor da bolinha.
- O jogo termina quando algum jogador não conseguir defender a bolinha. Quando isso acontecer, todo o fundo deverá mudar de cor indicando que o jogo acabou.