Relatório de Desenvolvimento do Jogo 3D - Creepy Cat

Luan De Nale
E-mail: luan.nale@acad.pucrs.br

Raffaela Monteiro
E-mail: raffaela.monteiro@acad.pucrs.br

I. DESCRIÇÃO

O trabalho proposto consiste em fazer um jogo 3D utilizando a biblioteca OpenGL/GLUT e os conceitos adquiridos ao longo do semestre. O OpenGL é uma API utilizada na computação gráfica, para desenvolvimento de aplicativos gráficos, ambientes 2D/3D, jogos, entre outros.

Para desenvolvimento do jogo, foi dada continuidade na temática utilizada no projeto de rendering da disciplina de Computação Gráfica I.

Regras do jogo:

- Objetivo: empurrar o teapot para fora dos limites da mesa;
- O jogador perde a partida se o gato ultrapassar os limites da mesa ou tocar em uma esfera;
- Cada teapot derrubado é pontuado 100 pontos;
- A cada 500 pontos acumulados, o jogador sobe um level e também aumenta o nível de dificuldade da partida (velocidade do gato e obstáculos na mesa);
- Controle do gato é feito pelas teclas direcionais (UP, DOWN, LEFT, RIGHT);
- Controle da câmera é feito pelas teclas numéricas (1, 2, 3, 4, 5, 6);
- Tecla 'esc' para sair do jogo;
- Tecla 'enter' inicia o jogo e retorna a posição original de câmera.

II. MECÂNICA E MODELAGEM DO JOGO

O cenário é composto por quatro objetos: um gato, uma mesa, uma esfera e um teapot. O jogador realiza o controle do personagem pelas teclas direcionais e controle da câmera pelas teclas numéricas de 1 a 6.

O arquivo main.cpp é o arquivo que possui todo o código fonte do jogo e abaixo pode-se conferir mais detalhes de implementação.

A. Modelagem Gato

O personagem principal do jogo foi modelado através do instanciamento de primitivas, utilizando esferas e cones como primitivas principais.

```
1  // Variaveis do gato
2  GLint catX = 30;
3  GLint catZ = 30;
4  GLfloat catRotation = 90.0f;
5  GLbyte direction  = 0;
6
7  void drawCat()
8  {
9      //Push
10     glPushMatrix();
11
12     //Gato
13     rotaciona();
14     glColor3f(0.5f,0.5f,0.4f); //Cor do gato
15     glTranslatef(catX,-10.0f,catZ); //Posicao inicial do gato
16     glScalef(0.5f,0.5f,0.5f); //Escala (Cabeca do gato)
17     glutSolidSphere(8,16,16); //Esfera (Cabeca do gato)
18     glRotatef(catRotation, 0.0f, 1.0f, 0.0f); //Rotacao do
           gato
19
20     //Fucinho
21     // cor, translacao, escala e cone do fucinho do gato [...]
22
23     //Olhos
24     // cor, translacao, escala e esferas dos olhos do gato
           [...]
25
26     //Orelhas
27     // cor, translacao, escala, rotacao e cones das orelhas do
           gato [...]
28
29     //Iris dos Olhos
30     // cor, translacao, escala, rotacao e esferas das iris do
           gato [...]
31
32     //Pop
33     glPopMatrix();
34 }
```

B. Modelagem Teapot

O objeto Teapot foi modelado através do método nativo da API do OpenGL chamado *glutSolidTeapot*.

```
1  // Variaveis do teapot
2  GLint teapotX = 0;
3  GLint teapotZ = 0;
4
5  //Desenha um teapot
6  void drawTeapot()
```

```
{
    glPushMatrix();
    rotaciona();
    glTranslatef(teapotX,-10.0,teapotZ);
    glColor3f(1, 1, 0);
    glScalef(0.5f,0.5f,0.5f);
    glutSolidTeapot(5);
    glPopMatrix();
}
```

C. Modelagem da mesa

A modelagem da mesa foi feita utilizando o mesmo método de instanciamento de primitivas utilizado no gato, só que as primitivas principais são um cubo e 4 cilindros.

```
void drawTable()
{
    //Desenha a tabua da mesa
    glPushMatrix();
    rotaciona();
    glColor3f(0.5f, 0.35f, 0.05f);
    glTranslatef(75, -16.0, 75.0);
    glScalef(100,2.0,100);
    glutSolidCube(1);
    glPopMatrix();

    //Desenha as pernas da mesa
    //Perna 1
    glPushMatrix();
    rotaciona();
    glColor3f(0.5f, 0.35f, 0.05f);
    glTranslatef(30,-87,30);
    glScalef(1.0,1.0,1.0);
    glRotatef(-90,1.0,0,0);
    glutSolidCylinder(5,70,20,10);
    glPopMatrix();

  //Perna 2

  //Perna 3

  //Perna 4
}
```

D. Modelagem da esfera

O objeto Esfera foi modelado através do método nativo da API do OpenGL chamado *glutSolidSphere*.

```
// Variaveis da esfera
GLint   esferaX = -1000;
GLint   esferaZ = -1000;

//Desenha uma esfera
void drawSphere()
{
    glPushMatrix();
    rotaciona();
    glTranslatef(esferaX,-11.0,esferaZ);
    glColor3f(0.5, 0.1, 0.1);
    glScalef(0.5,0.5,0.5);
    glutSolidSphere(7,12,12);
    glPopMatrix();
}
```

E. Movimentação de câmera

A visão do jogo pode ser alterada a qualquer momento utilizando as setas numérias de 1 a 6, e esta funcionalidade está implementada no método *keyboard* com deslocamento +-10. As teclas estão definidas da seguinte forma:

- Tecla '1': para baixo (em relação ao eixo y);
- Tecla '2': para cima (em relação ao eixo y);
- Tecla '3': direita (em relação ao eixo x);
- Tecla '4': esquerda (em relação ao eixo x);
- Tecla '5': menos zoom;
- Tecla '6': mais zoom;
- Tecla 'esc': sai do jogo.
- Tecla 'enter': retorna a posição original de câmera.

```
//Variaveis do angulo da camera
GLfloat angleX = 30.0f;
GLfloat angleY = 0.0f;
GLfloat zoom  = -300.0f;

//Trata teclas numericas
void keyboard (unsigned char key, int catX, int catY)
{
    switch (key)
    {
    //Movimentacao de camera
    case '1':
        angleX -= 10;
        break;
    case '2':
        angleX += 10;
        break;
    case '3':
        angleY -= 10;
        break;
    case '4':
        angleY += 10;
        break;
    case '5':
        zoom -= 10;
        break;
    case '6':
        zoom += 10;
        break;
    // Sair do jogo
    case 27:
        exit(0);
        break;
    default:
        break;
    }
```

F. Geração objetos utilizando o random

O teapot e a esfera são gerados em cima da mesa utilizando um método chamado random, com o intuito de os objetos serem gerados em locais aleatórios do cenário, sem conhecimento prévio do jogador.

```
int random(int up, int low)
{
    return (rand() % (up-low))+low;
}

//Gera novo TeaPot baseado no calculo de randomico
void newTeaPot()
{
    time_t seconds;
    time(&seconds);
    srand((unsigned int) seconds);
    teapotX = random(95, 40);
    teapotZ = random(95, 40);
}

//Gera nova esfera baseada no calculo de randomico
void newSphere()
{
    time_t seconds;
    time(&seconds);
    srand((unsigned int) seconds*13);
    esferaX = random(95, 40);
    esferaZ = random(95, 40);
}
```

G. Interações com o objeto Teapot

O gato realiza a movimentação do teapot pelo método *pushTeapot* que empurra o teapot a cada 5 casas nas 4 direções de controle. Ao objeto ultrapassar os limites da mesa, é chamado o método *breakTeapot* que "quebra"o objeto e inicializa ele em um novo local aleatório (método randômico mencionado anteriormente).

```
//Ocorre quando um TeaPot ultrapassar a borda da mesa, quebrando ele
void breakTeapot()
{
    PlaySound("break.wav", NULL, SND_ASYNC|SND_FILENAME);
    score = score + 100;
    if(score%500 == 0)
    {
        lvl++;
        if (speed==50) speed-=10;
        else speed-=50;
        if (speed<2) speed = 1;
        PlaySound("SuperMarioBros_Mushroom.wav", NULL, SND_ASYNC|SND_FILENAME);
        newSphere();
    }
    newTeaPot();
}
//Chamado quando gato estiver empurrando TeaPot
void pushTeapot()
{
    switch(direction)
    {
    case RIGHT:
        teapotX += 5;
        if(teapotX > 125) breakTeapot();
        break;
    case LEFT:
        teapotX -= 5;
        if(teapotX < 25) breakTeapot();
        break;
    case UP:
        teapotZ += 5;
        if(teapotZ > 125) breakTeapot();
        break;
    case DOWN:
        teapotZ -= 5;
        if(teapotZ < 25) breakTeapot();
        break;
    }
}
```

H. Movimentos do personagem

O gato se movimenta através do método *moveCat* que desloca o personagem a cada 5 casas nas 4 direções de controle. Esse método chama o *catRotation* para informar qual a direção (ângulo) que ele irá girar e delimita os limites da mesa que o gato não pode ultrapassar, caso ultrapasse irá chamar o método restart e iniciar uma nova partida.

```
//Trata o movimento do gato
void moveCat(int value)
{
    int i;
    switch(direction)
    {
    case RIGHT:
        catRotation =90;
        catX += 5;
        if(catX > 125) restart();
        break;
    case LEFT:
        catRotation =-90;
        catX -= 5;
        if(catX < 25) restart();
        break;
    case UP:
        catRotation =0;
        catZ += 5;
        if(catZ > 125) restart();
        break;
    case DOWN:
        catRotation =180;
        catZ -= 5;
        if(catZ < 25) restart();
        break;
    }
    //Testa colisaentre gato e teapot, se houver entao empurra teapot
    if((catX >= teapotX - 6) && (catX <= teapotX + 6) && (catZ >= teapotZ - 6) && (catZ <= teapotZ + 6))
    {
        pushTeapot();
    }
    //Testa colisao entre gato e as esferas, se houver entao reinicia o jogo
    if((catX >= esferaX - 5) && (catX <= esferaX + 5) &
```

```
            (catZ >= esferaZ - 5) && (catZ <= esferaZ + 5))
            restart();
35      //Define a velocidade do gato (speed eh tratado em
            milisegundos, logo quanto menor mais rapido chama
            moveCat)
36      glutTimerFunc(speed, moveCat, 0);
37  }
```

III. ELEMENTOS ADICIONAIS

Na seção anterior descrevemos os elementos essenciais para entender a mecânica do jogo. Nesta seção iremos analisar elementos adicionais que complementam a experiência do jogador que jogar Creepy Cat.

A. Àudios

Temos alguns elementos de áudio que são usados durante o jogo para dar maior imersão ao jogador quando este estiver jogando *Creepy Cat*.
Estes áudios são todos disparados pelo método:

```
1  PlaySound("file_name.wav", NULL,
       SND_ASYNC|SND_FILENAME);
```

Agora vamos dar uma olhada nos áudios utilizados no jogo:

- **Psicose** - Quando o jogo inicia, a fim de dar uma atmosfera mais sombria ao jogo, é tocada a emblemática música do filme **Psicose**;
- **Break** - Este é um áudio de cerâmica/vidro quebrando e é usado quando um Teapot ultrapassa a borda da mesa, indicando que ele caiu e quebrou;
- **Meow** - Áudio de um gato miando, usado quando o *Creepy Cat* ultrapassa a borda da mesa (ou toca na esfera vermelha), indicando que ele se machucou e o jogo terminou;
- **Super Mario Mushroom** - Este é o áudio característico de quando Mario do jogo Super Mario, coleta um cogumelo. Este som é usado quando o jogador subir de level em *Creepy Cat* para sinalizar uma progressão no jogo.

B. Pontuação

Um conceito muito importante em qualquer jogo é competitividade e a noção de progresso perante os desafios encontrados no jogo. Para atender isso é utilizado comumente pontuações e níveis de progressão no jogo, que foi como utilizamos neste projeto.

O jogador iniciará no *Level* 1 e com *Score* 0. Para cada Teapot que o gato derrubar da mesa serão acrescidos 100 pontos ao *Score* e, de 500 em 500 pontos, será incrementado o *Level*.

Para acrescentar uma dificuldade a mais, a partir do *Level* 2 teremos uma esfera que aparecerá randômicamente na mesa e que o gato deverá evitar tocar.

Quando o jogo terminar, seja pelo gato tocar na esfera vermelha ou cair da mesa, o jogo será reiniciado e será exibida a pontuação recorde.

IV. CONCLUSÃO

Com este projeto foi possível aprender os conceitos e a manipulação do OpenGL durante a criação do jogo. Também foi um processo prazeroso visto que foi uma oportunidade de exercer o senso criativo na criação de jogos.

V. REFERÊNCIAS

- Biblioteca OpenGL/GLUT: OpenGL
- Biblioteca MinGW: MinGW
- Gravação GamePlay com ZD Soft: ZD Soft Screen Recorder 10.1.3
- Resolução de dúvidas: Fórum OpenGL
- OpenGL - Uma abordagem prática e objetiva; COHEN, Marcelo; MANSSOUR, Isabel Harb; Novatec Editora