Computação Gráfica

Prof. Bruno Augusto Dorta Marques

2021.3

Atividade 1

Jogo da Memória

Guilherme Assencio RA:11201810653

Henrique Queiroz Reuter RA:11201812261

Implementação

- M_program criado com o VertexShader e FragmenteShader fornecidos em aula, na sessão dos polígonos regulares.
- Foi utilizado um vector de tuplas “inicialPos” para armazenar os pontos onde as cartas do jogo da memória seriam estanciadas.
```
std::vector<std::tuple<float, float>> inicialPos;
```
- Foi utilizado um map “nLados” para associar cada ponto do “inicialPos” ao número de lados do polígono que será gerado ao virar a carta.
```
std::map<std::tuple<float, float>, int> nLados;
```
- Foi utilizado um map “cores” para associar cada número de lados do “nLados” à uma cor RGB.
```
std::map<int, glm::vec3> cores;
```
- Ao chamar a função “iniciar()”, “inicialPos”, “nLados” e “cores” são preenchidos, e variáveis auxiliares para realizar a lógica do programa tem valores atribuídos. Além de embaralhar o “inicialPos” com o auxilio da função std::shuffle.
```
m_randomEngine.seed(
    std::chrono::steady_clock::now().time_since_epoch().count());

std::array<int, 12> shuf{4, 4, 3, 3, 5, 5, 6, 6, 7, 7, 8, 8};
unsigned num = std::chrono::system_clock::now().time_since_epoch().count();
std::shuffle(shuf.begin(), shuf.end(), std::default_random_engine(num));
```
- Ainda na função iniciar, o game state é setado como Playing e o game input é setado como Not Clicking.
- A cada execução do paintGl é desenhado um quadrado azul em cada ponto do “incialPos” com o auxílio da função setupModel, fornecida em aula.
- A função setupModel foi modificada para aceitar o número de lados e a cor de cada polígono a ser desenhado
```
void setupModel(int sides, const glm::vec3 color);
```
- Foi utilizado a função handleEvent para identificar o click do usuário e calcular a posição X e Y do click.

```
void OpenGLWindow::handleEvent(SDL_Event &event) {
  // Mouse events
  if (event.type == SDL_MOUSEBUTTONDOWN) {
    if (event.button.button == SDL_BUTTON_LEFT) {
      SDL_GetMouseState(&mousePosition.x, &mousePosition.y);
      x = ((mousePosition.x / (double)600) * 2) - 1;
      y = -(((mousePosition.y / (double)600) * 2) - 1);
      m_gameData.m_input.set(static_cast<size_t>(Input::Click));
    }
  }
  if (event.type == SDL_MOUSEBUTTONUP) {
    if (event.button.button == SDL_BUTTON_LEFT)
      m_gameData.m_input.reset(static_cast<size_t>(Input::Click));
  }
}
```

- Ao clicar em uma carta virada para trás, a função paintGl desenha um quadrado branco na carta, e um polígono de acordo com a cor e o número de lados associados ao ponto.

```
if (m_gameData.m_input[static_cast<size_t>(Input::Click)]) {
  m_gameData.m_input.reset(static_cast<size_t>(Input::Click));
  for (pos = inicialPos.begin(); pos != inicialPos.end(); pos++) {
    if (x >= std::get<0>(*pos) - 0.176776695f &&
        x <= std::get<0>(*pos) + 0.176776695f &&
        y >= std::get<1>(*pos) - 0.176776695f &&
        y <= std::get<1>(*pos) + 0.176776695f) {
```

(Condicional utilizada para identificar o click em uma carta)

- A carta clicada é temporariamente removida do “inicialPos” para que não seja desenhado um quadrado azul naquele ponto.
- Ao clicar em uma segunda carta, a carta é “virada” da mesma forma da primeira e é verificado:
    - Se as figuras forem iguais, os dois pontos são removidos do “inicialPos”, desse modo, não será desenhado mais nada naquele ponto
    - Se as figuras forem diferentes, o programa entra em “sleep” por 1 segundo para o usuário conseguir ver a segundo carta virada, então os dois pontos são colocados de volta no “inicialPos”
- Ao encontrar todos os pares de carta, o state do jogo é alterado para *Win*, é exibido uma mensagem “You Win!!!!” na função paintUI com o auxílio do ImGUI::Text.
- Após 3 segundos o jogo é colocado no estado inicial chamando a função “iniciar” novamente.
- Os VBOs e VAOs foram manipulados nas funções setupModel e paintGl.