Nomes: Gustavo Willian Martins da Silva e Vítor Aguirre Caús

**Computação Gráfica**

**Trabalho 2 - Robô Articulado Atirador**

Neste trabalho, buscou-se implementar um cenário 3d onde o usuário é capaz de controlar um veículo atirador articulado, capaz de disparar projéteis para destruir um paredão posicionado no centro do espaço onde ele se localiza. As articulações desenvolvidas permitem a rotação do tanque em torno do eixo Y e a subida/descida de seu canhão em relação ao eixo Z, de modo a posicionar a trajetória do disparo conforme desejado, também sendo possível controlar a força do tiro do canhão, de modo a alcançar alvos mais distantes. Abaixo, são descritas as implementações realizadas em relação ao código

* **Desenho do Paredão, Piso e Texturas:**

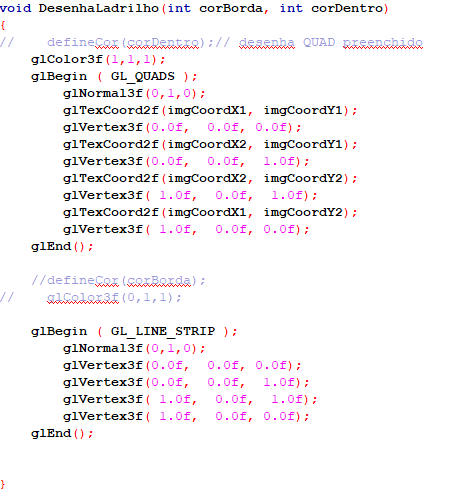
Inicialmente, é desenhado um piso de tamanho 25X50 no ponto (-20,-1,-10), a partir do qual, em seguida, é desenhada a parede de tamanho 25X15, que sofre transformações de translação e de rotação no eixo Z para que fique posicionada no centro do piso. No momento de criação desses dois planos, são aplicadas suas texturas (grama e tijolos), mediante auxílio de variáveis que guardam os valores X e Y de cada imagem para mapear corretamente um trecho das texturas a cada polígono do piso e da parede.

**Texto

Descrição gerada automaticamente**

**Texto

Descrição gerada automaticamente**

****

* **Modelagem de veículo, articulações e movimento:**

O veículo atirador é modelado com dois cubos: o primeiro é transformado com uma escala de (3,1,2), enquanto o segundo é transladado para o ponto central do cubo da base (com incremento de 0.75 em Y) e então é desenhado com escala (2.0, 0.5, 0.5).

Com as teclas “a” e “d” é realizado, respectivamente, o incremento e o decremente do ângulo do veículo em 5°, que é utilizado, durante seu desenho, para rotacionar a base do tanque e o canhão em torno do eixo Y. Já com as teclas “c” e “C”, o ângulo do canhão em torno do eixo Z é, respectivamente, aumentado e reduzido em 1°. A movimentação do canhão só é realizada quando o tanque tem ângulo de rotação igual a 0°, ou seja, quando ele está direcionado de frente para o paredão.

Para o movimento do tanque, as teclas “w” e “s” foram definidas para movê-lo para frente e para trás, de modo a alterar as posições da base do tanque e de seu canhão, além dos pontos do observador e alvo da visão de primeira pessoa.

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

* **Lançamento do projétil:**

Com base na formulação sugerida para cálculo dos pontos da trajetória, foram definidos dois pontos, “alcanceAux” e “alcanceFinal”, para tratar o deslocamento do tiro do canhão. Esses dois pontos também são rotacionados conforme o ângulo do veículo, a fim de alinhar corretamente a mira do canhão com sua estrutura quando o tanque gira para um dos lados. Dados os 2 pontos mencionados e a posição do canhão, é calculada uma curva Bèzier para definir o trajeto do objeto disparado (ativado pela tecla “m”), cuja força é incrementada e decrementada com as teclas “f” e “F”, sendo limitada de 1 a 50. Só é possível disparar um único projétil por vez, até que este atinja a parede ou saia dos limites definidos para seu cálculo e renderização.

Texto

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

* **Detecção de colisão com a parede e reconfiguração:**

Para o mapeamento dos polígonos da parede, foi definida uma matriz 25 X 15 com valores booleanos inicializada como ***true***em todas as posições, a fim de permitir o desenho dos polígonos do paredão apenas se suas respectivas posições na matriz indicarem que eles existem. A partir disso, durante o cálculo da trajetória do projétil, é verificado se ele alcançou ou atravessou o paredão (caso sua coordenada X seja maior ou igual à do paredão) e, em caso afirmativo, a posição do projétil é mapeada para sua respectiva posição na matriz de booleanos da parede, de modo a adquirir o valor que define ou não se há um polígono no ponto atingido.

Caso se confirme que o projétil acertou um ponto onde há parte do paredão, a posição desse polígono é convertida para ***false*** na matriz, e o mesmo ocorre com os 8 quadrados ao seu redor, se existirem. Dessa forma, os polígonos que passam a ser definidos como inexistentes deixam de ser renderizados. A colisão e destruição do paredão garante 10 pontos ao usuário.

Texto

Descrição gerada automaticamente com confiança média

* **Exibição de objetos TRI:**

Com base nos códigos de exemplo para leitura, armazenamento e exibição de objetos TRI, modificou-se parte da leitura para incluir o cálculo dos vetores normais de cada face triangular, além de sua adição na função responsável por exibir os objetos. Entretanto, é possível observar que a qualidade da renderização depende da posição do observador em relação a cada objeto (identificado no vídeo de demonstração).

Texto

Descrição gerada automaticamente

* **Teclas auxiliares:**
  + t = altera a visão entre primeira/terceira pessoa;
  + u/U = na terceira pessoa, rotacionam o alvo em torno do eixo Z, respectivamente, para cima e para baixo.
* **Critérios não alcançados durante a execução do trabalho:**
  + Colisão do veículo com as bordas do piso e com trechos desenhados da parede;
  + Colisão do projétil e do veículo com objetos TRI (e consequentemente sua destruição, no caso do projétil);
  + Parte do mapeamento de colisão do paredão exclui trechos errados ou não exclui nada mesmo com o projétil atingindo-o.

**Link para o vídeo:**

<https://www.youtube.com/watch?v=dKGcaakYSZs>