Departamento de Ciência da Computação

Profª Ana Paula Piovesan Melchiori

DOCUMENTAÇÃO: escavadeira.cpp v2.0

Grupo:
- Sérgio Henrique Menta Garcia
Matrícula 201810013;
- Mateus Carvalho Gonçalves
Matrícula 201810245.

Turma 10A

Documentação entregue como parte das exigências do Projeto Prático da disciplina GCC124 - Computação Gráfica

LAVRAS - MARÇO/2021

SUMÁRIO

1 INTRODUÇÃO

Este documento descreve o projeto da aplicação **escavadeira.cpp**, na sua versão 2.0, um programa de computação gráfica em OpenGL que deve desenhar uma escavadeira simplificada que aplica interações e conceitos estudados previamente.

Pode-se discutir a importância do estudo de Computação Gráfica na área de Tecnologia da Informação a partir da sua definição. Gomes e Velho (1998) definem-na como a ciência que “estuda os métodos que permitem a visualização de informações armazenadas na memória do computador”.

Molina (2014) data as origens da Computação Gráfica nos anos 1960, e que documentos da década de 1950 já apontavam a melhoria da velocidade e facilidade de uso nos chamados sistemas gráficos. Podemos perceber, então, que a Computação Gráfica surgiu da necessidade de melhorar a visualização dos dados e a eficiência do trabalho, além de objetivos militares também, como simuladores de voo, por exemplo (Wikipedia, The Free Encyclopedia).

Diante dessas informações, podemos perceber que a Computação Gráfica está em inúmeros âmbitos da vida cotidiana, como em interfaces gráficas de usuário em computadores e smartphones, vídeo games, filmes, animações, etc. Logo, a importância dessa área na atualidade já está em âmbitos de qualidade de vida e entretenimento.

Para solidificar esses conceitos tão importantes da disciplina obrigatória GCC124 - Computação Gráfica do curso de Ciência da Computação, ofertada pelo Departamento de Ciência da Computação da Universidade Federal de Lavras, o Projeto Prático tratado neste documento tem os seguintes requisitos básicos:
(1) Desenvolvimento de ambiente 3D em OpenGL; (2) Ambiente colorido - cores diferentes para objetos distintos; (3) Interação com teclado e mouse; (4) Operações de transformações geométricas; (5) Iluminação - permitindo ligar/desligar luz; (6) Animação de pelo menos 1 objeto; e (7) Textura em pelo menos 1 objeto.

O restante desse documento se organiza da seguinte forma. Seção 2 lista e apresenta os requisitos do sistema discutido no presente trabalho. Seção 3 apresenta as estratégias de codificação, incluindo as tecnologias utilizadas, ambiente de desenvolvimento, compilação e execução, entre outras informações

inerentes ao código. Seção 4 reporta os resultados obtidos e discute-os, também apontando as limitações do trabalho. E por fim, a Seção 5 apresenta as conclusões.

2 REQUISITOS DO SISTEMA

Esta seção apresenta os requisitos da aplicação proposta, separados em requisitos funcionais e requisitos não-funcionais.

Os requisitos funcionais foram baseados nas interações desenvolvidas correspondentes a resposta do programa ao pressionar uma tecla do teclado ou botão do mouse, que resultam numa animação do objeto. Esses são apresentados na Subseção 2.1.

Os requisitos não-funcionais foram baseados nos comportamentos e estética de todo o programa, desde o objeto desenhado até a janela de visualização. Esses são apresentados na Subseção 2.2.

Cada requisito é descrito numa tabela com 2 linhas e 2 colunas. Possui a anotação RF e RNF para diferenciar os requisitos funcionais dos não-funcionais (célula 1x1), seguida de uma numeração sequencial, e o título do requisito (célula 1x2). Também, o título da descrição (célula 2x1) e o conteúdo relacionado a descrição do requisito (célula 2x2).

2.1 Requisitos Funcionais (RF)

[RF-001]	**Fechar janela**
Descrição	Ao pressionar a tecla "Esc", a janela será fechada, encerrando a aplicação.

[RF-002]	**Ligar a luz**
Descrição	Ao clicar no botão esquerdo do mouse, e com a luz desligada, ela deverá ser ligada.

[RF-003]	**Desligar a luz**
Descrição	Ao clicar no botão esquerdo do mouse, e com a luz ligada, ela deverá ser desligada.

[RF-004]	**Girar as rodas no sentido horário**
Descrição	Ao pressionar as teclas “Shift + A”, as quatro rodas da escavadeira deverão girar no sentido horário.

[RF-005]	**Girar as rodas no sentido anti-horário**
Descrição	Ao pressionar a tecla “A”, as quatro rodas da escavadeira deverão girar no sentido anti-horário.

[RF-006]	**Trocar a cor da luz da cabine para amarelo**
Descrição	Ao clicar no botão esquerdo, e com a cor da luz da cabine estiver branca, essa deverá ser trocada para amarelo.

[RF-007]	**Trocar a cor da luz da cabine para branco**
Descrição	Ao clicar no botão esquerdo, e com a cor da luz da cabine estiver amarela, essa deverá ser trocada para branco.

[RF-008]	**Girar a cintura da escavadeira no sentido horário**
Descrição	Ao pressionar as teclas “Shift + Q”, a cintura da escavadeira deverá girar no sentido horário.

[RF-009]	**Girar a cintura da escavadeira no sentido anti-horário**
Descrição	Ao pressionar a tecla “Q”, a cintura da escavadeira deverá girar no sentido anti-horário.

[RF-010]	**Levantar o braço da escavadeira**
Descrição	Ao pressionar as teclas “Shift + W”, o braço da escavadeira deverá levantar-se.

[RF-011]	**Abaixar o braço da escavadeira**
Descrição	Ao pressionar a tecla “W”, o braço da escavadeira deverá abaixar-se.

[RF-012]	**Levantar o antebraço da escavadeira**
Descrição	Ao pressionar as teclas “Shift + E”, o antebraço da escavadeira deverá levantar-se.

[RF-013]	**Abaixar o antebraço da escavadeira**
Descrição	Ao pressionar a tecla “E”, o antebraço da escavadeira deverá abaixar-se.

[RF-014]	**Levantar a pá da escavadeira**
Descrição	Ao pressionar as teclas “Shift + R”, a pá da escavadeira deverá levantar-se.

[RF-015]	**Abaixar a pá da escavadeira**
Descrição	Ao pressionar a tecla “R”, a pá da escavadeira deverá abaixar-se.

[RF-016]	**Movimentar a escavadeira para frente**
Descrição	Ao pressionar a tecla “Space”, a escavadeira deverá movimentar-se para frente.

[RF-017]	**Movimentar a escavadeira para trás**
Descrição	Ao pressionar a tecla “Backspace”, a escavadeira deverá movimentar-se para trás.

[RF-018]	**Aproximar a escavadeira e o cenário do campo de visualização (Zoom In)**
Descrição	Ao pressionar a tecla “+”, a escavadeira e o cenário deverão aproximar-se do campo de visão.

[RF-019]	**Afastar a escavadeira e o cenário do campo de visualização (Zoom Out)**
Descrição	Ao pressionar a tecla “-”, a escavadeira e o cenário deverão afastar-se do campo de visão.

[RF-020]	**Girar, na horizontal, a escavadeira e o cenário no sentido horário**
Descrição	Ao pressionar a tecla “seta para direita”, a escavadeira e o cenário deverão girar na horizontal e no sentido horário.

[RF-021]	**Girar, na horizontal, a escavadeira e o cenário no sentido anti-horário**
Descrição	Ao pressionar a tecla “seta para esquerda”, a escavadeira e o cenário deverão girar na horizontal e no sentido anti-horário.

[RF-022]	**Girar, na vertical, a escavadeira e o cenário no sentido horário**
Descrição	Ao pressionar a tecla “seta para baixo”, a escavadeira e o cenário deverão girar na vertical e no sentido horário.

[RF-023]	**Girar, na vertical, a escavadeira e o cenário no sentido anti-horário**
Descrição	Ao pressionar a tecla “seta para cima”, a escavadeira e o cenário deverão girar na vertical e no sentido anti-horário.

2.2 Requisitos Não-Funcionais (RNF)

[RNF-001]	**Objetos em 3D**
Descrição	Todos os objetos desenhados deverão ser em 3D.

[RNF-002]	**Aplicação de iluminação**
Descrição	A iluminação será feita através de uma fonte padrão do OpenGL e utilizará os tipos ambiente, difusa e especular.

[RNF-003]	**Tamanho inicial da janela de visualização**
Descrição	A janela terá o tamanho inicial de 700px por largura e 500px por altura.

[RNF-004]	**Expansão da janela de visualização**
Descrição	A janela poderá se expandir para ocupar 100% da tela do dispositivo.

[RNF-005]	**Responsividade da janela de visualização**
Descrição	A janela poderá ser responsiva, mas não necessariamente atender a todos os tamanhos.

[RNF-006]	**Limitar movimento do braço da escavadeira**
Descrição	O movimento do braço da escavadeira deverá ser limitado entre 0° e 75°.

[RNF-007]	**Limitar o movimento do antebraço da escavadeira**
Descrição	O movimento do antebraço da escavadeira deverá ser limitado entre -75° e 0°.

[RNF-008]	**Limitar o movimento da pá da escavadeira**
Descrição	O movimento da pá da escavadeira deverá ser limitado entre -5° e 55°.

[RNF-009]	**Limitar o movimento da escavadeira e do cenário na vertical**
Descrição	O movimento da escavadeira e do cenário deverá ser limitado entre -10° e 90°.

3 ESTRATÉGIAS DE CODIFICAÇÃO

Esta seção lista as tecnologias utilizadas no desenvolvimento do projeto e o ambiente de programação, bem como explica a instalação das dependências, compilação e execução em ambiente Linux. Também são discutidas as técnicas e estratégias utilizadas para cumprir os requisitos apresentados na seção anterior.

3.1 Ambiente, tecnologias, compilação e execução

O projeto foi desenvolvido no sistema operacional Ubuntu 20.04.1 LTS. A linguagem de programação utilizada foi o C++17, por ser a linguagem adotada como padrão na disciplina, e compilador g++ v9.3.0.

Juntamente, a API de computação gráfica OpenGL foi utilizada na versão 4.6.0 com as bibliotecas GLU (OpenGL Utility) e GLUT (OpenGL Utility Toolkit - uma API de janela portátil e não faz parte oficialmente do OpenGL). A instalação pode ser feita pelo comando:

```
$ sudo apt-get install libglu1-mesa-dev freeglut3-dev mesa-common-dev
```

E para compilação e execução do programa em ambiente Linux:

```
$ g++ escavadeira.cpp -o escavadeira -lglut -lGLU -lGL
$ ./escavadeira
```

A IDE escolhida para codificação do projeto foi o VSCode, que suporta inúmeras linguagens de programação e fornece extensões para melhorar o desempenho e produtividade do programador.

3.2 Estratégias e técnicas de codificação

A API OpenGL Utility Toolkit (GLUT) fornece métodos para criação e manuseio de janelas portáteis para os programas em OpenGL, Esses métodos são chamados na rotina principal e alguns possuem como parâmetro métodos definidos no código para definir as configurações e interações, como mostra a Tabela 1. Inclusive, o requisito básico (3) citado na Seção 1, por exemplo, é satisfeito pelos eventos definidos nas funções passadas por parâmetro nas chamadas **glutMouseFunc(mouseEvents)**, **glutKeyboardFunc(keyboardEvents)** e **glutSpecialFunc(specialKeysEvents)**.

Duas funções do OpenGL são vitais para o método encarregado de desenhar a cena, no presente projeto intitulado *renderScene()*, são elas **glPushMatrix()** e **glPopMatrix**. Essas chamadas empilham e desempilham matrizes de desenho, respectivamente, isso permite isolar transformações geométricas, definições de cores, etc.

Tabela 1. Rotina principal definindo métodos de janela com API GLUT

```
487 int main(int argc, char** argv) {
488     glutInit(&argc, argv);
489     glutInitDisplayMode (GLUT_DOUBLE | GLUT_DEPTH | GLUT_RGBA);
490     glutInitWindowSize (700, 500);
491     glutInitWindowPosition (100, 100);
492     glutCreateWindow ("Escavadeira v2.0");
493     init();
494     glutReshapeFunc(reshape);
495     glutDisplayFunc(renderScene);
496     glutMouseFunc(mouseEvents);
497     glutKeyboardFunc(keyboardEvents);
498     glutSpecialFunc(specialKeysEvents);
499     glutMainLoop();
500
501     return 0;
502 }
```

O objeto principal deste projeto (escavadeira), foi desenvolvido a partir de várias formas 3D. Paralelepípedos foram desenhados a partir de chamadas **glutSolidCube()** precedidas de transformações de escalas com **glScalef()**. Já os cilindros, utilizados nas rodas e na “cintura”, são feitos pelo método **gluCylinder()**. No caso das rodas, que devem ser sólidas (fechadas), foi necessário também complementar com 2 chamadas **gluDisk()**, uma para cada base. É interessante apontar que para utilização da função de desenhar cilindros é preciso definir uma quádrica e passá-la como parâmetro para a chamada do método.

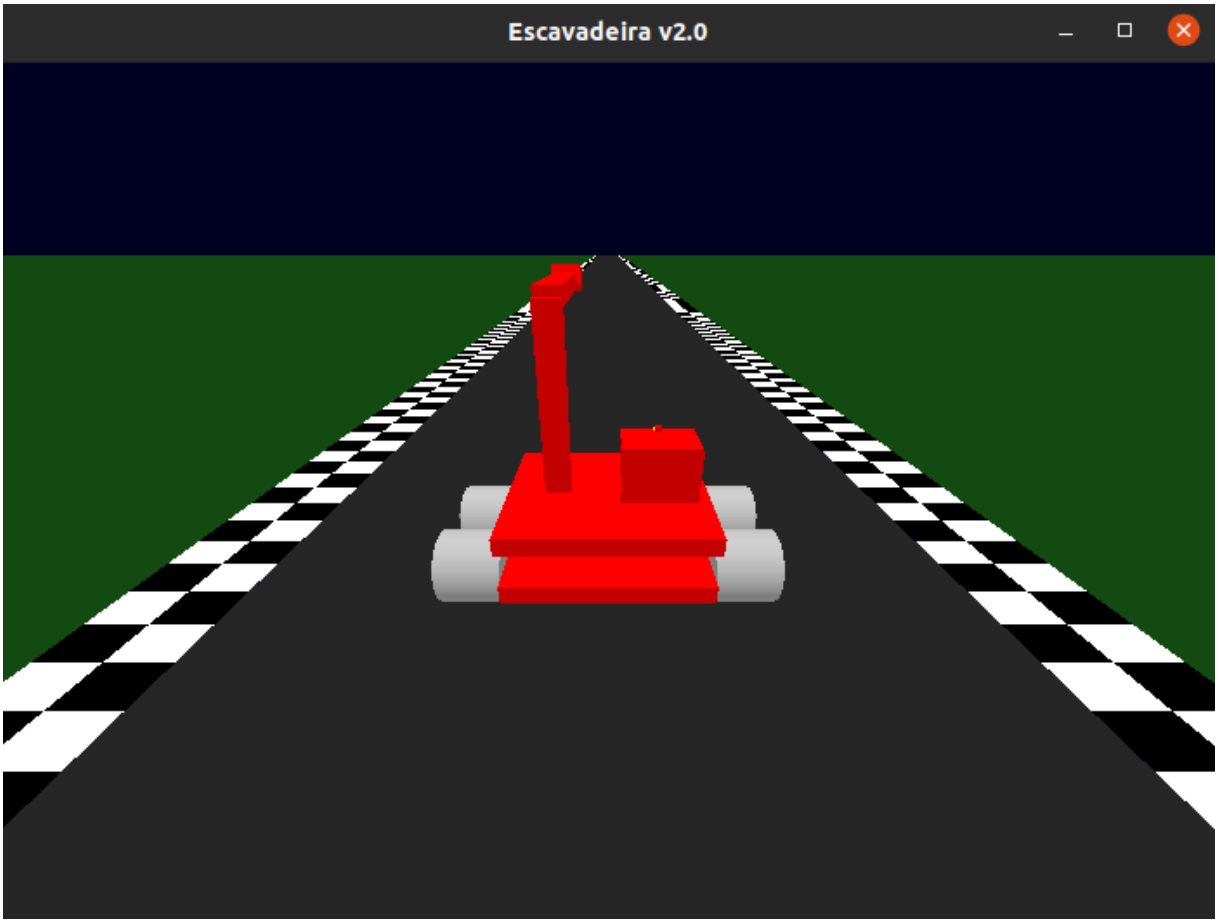
Já os elementos que compõem o chão - asfalto, bordas e planos laterais - foram desenhados utilizando a diretiva **GL_QUADS**.

As transformações geométricas são essenciais para a construção de um programa interativo e animado em OpenGL. Por meio dos métodos **glRotatef()**, **glTranslatef()** e o anteriormente citado **glScalef()** é possível aplicar rotações, translações e operações de escala. Apoiados por variáveis de controle que são atualizadas nas funções de interação por teclado e mouse, estes métodos permitiram as animações e movimentos de câmera implementados no projeto, além do suporte às operações para renderizar a cena.

As cores dos elementos foram implementadas por meio do método de definição de cor **glColor3f()**, que recebe três parâmetros do tipo float que integram a cor no formato RGB. O “céu” azul foi definido pela cor de fundo com **glClearColor()**.

A projeção perspectiva foi escolhida para transmitir sensação de profundidade, ainda mais se tratando de um ambiente 3D. Essa definição pode ser encontrada na função *reshape()* por meio da chamada **gluPerspective()**. A sensação de profundidade é facilmente identificada na Figura 1 uma vez que as bordas paralelas do asfalto convergem à medida que a distância aumenta.

Figura 1. Programa executando, visão traseira da escavadeira



Para iluminação, o grupo optou pela utilização de uma das fontes disponibilizadas pelo OpenGL (**GL_LIGHT0**), também dos tipos de iluminação: ambiente, difusa e especular; ambas com seus valores de RGB próximos ou iguais a cor branca, inicialmente, e uso das próprias cores configuradas dos objetos como material. Para permitir que a luz fosse desligada ou apagada, foi utilizado uma variável com coloração mais escura, sendo atribuída ao clicar no botão esquerdo do mouse.

Para texturização, o grupo tentou implementar três técnicas diferentes, que serão discutidas na Seção 4. Por fim, o caminho escolhido foi por mapeamento de

matriz em código para desenhar a imagem, o método utilizado está presente na Tabela 2. A matriz desenha uma textura em formato de tabuleiro de xadrez que foi utilizado como as bordas da pista. Na função de desenhar as bordas é necessário mapear os pontos da imagem com os vértices da figura geométrica.

Tabela 2. Método que mapeia matriz para formar imagem usada como textura

```
37  void makecheckeredBorder(void) {
38     int i, j, c;
39
40     for(i = 0; i < checkeredBorderHeight; i++) {
41          for (j = 0; j < checkeredBorderWidth; j++) {
42              c = ((((i&0x8)==0)^((j&0x8))==0))*255;
43              checkeredBorder[i][j][0] = (GLubyte) c;
44              checkeredBorder[i][j][1] = (GLubyte) c;
45              checkeredBorder[i][j][2] = (GLubyte) c;
46              checkeredBorder[i][j][3] = (GLubyte) 255;
47          }
48     }
49  }
```

Por se tratar de um projeto relativamente pequeno foi decidido que o programa seria estruturado em apenas um arquivo. Da mesma forma, é de conhecimento dos desenvolvedores que o uso de variáveis de escopo global pode ser prejudicial a um projeto de software em aspectos de legibilidade e proteção de dados, porém o baixo número de linhas de código possibilita a manutenção dessas variáveis.

4 RESULTADOS E DISCUSSÃO

Considerando o projeto desenvolvido e os requisitos básicos descritos na Seção Introdução, os resultados alcançados foram satisfatórios, já que atendeu a esses requisitos.

Com os conceitos de desenho 3D colorido, uso das diretivas do OpenGL e das funções de desenhos das API GLUT e GLU apresentados no decorrer desta disciplina, foi possível desenhar o projeto proposto. Começando pelo cenário de fundo, que é composto pelo asfalto, canteiro listrado e planície, cada um em formato de retângulo sólido.

Em relação à escavadeira, resumidamente, pode ser dividida em parte inferior e superior, ambas tendo como base um paralelepípedo sólido, unidas por uma

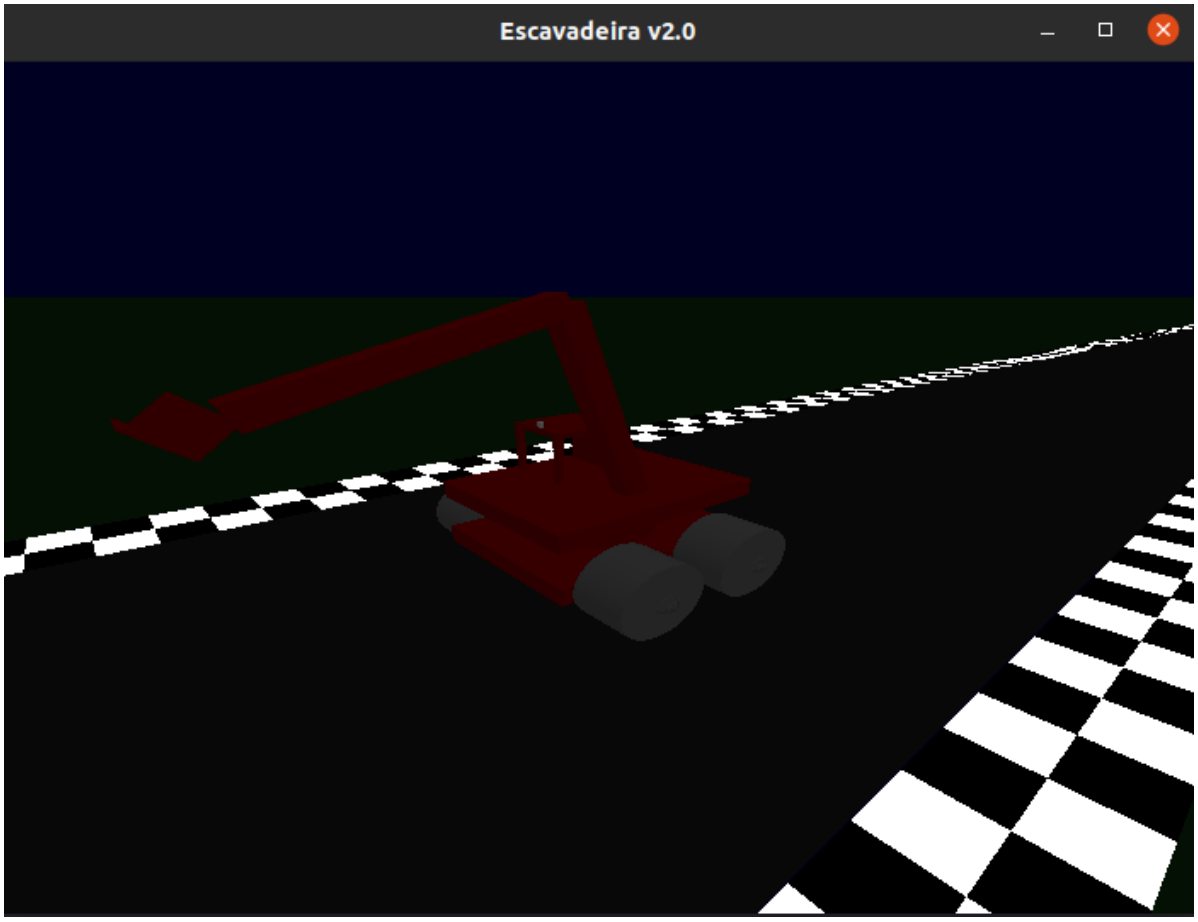
“cintura” em formato de cilindro sólido. Em seguida, mais detalhes sobre as partes inferior e superior.

A parte inferior é composta por quatro rodas em formato cilíndrico sólido e unidas pela base. Nas pontas desses cilindros, para não ver a parte interior, foram inseridos discos sólidos, representando as calotas. Para apresentar a animação do giro da roda, foram inseridas esferas de arames nas calotas de “fora”. Isso conclui a parte inferior.

A parte superior é composta por uma cabine e o braço hidráulico da escavadeira. Essa cabine é composta por quatro cubos sólidos: as costas (retângulo largo), as colunas (retângulos compridos, menos largo) e o telhado (retângulo largo). Além de um farol no topo, composto por dois cubos: um para a luz e o outro para a “capa” do farol.

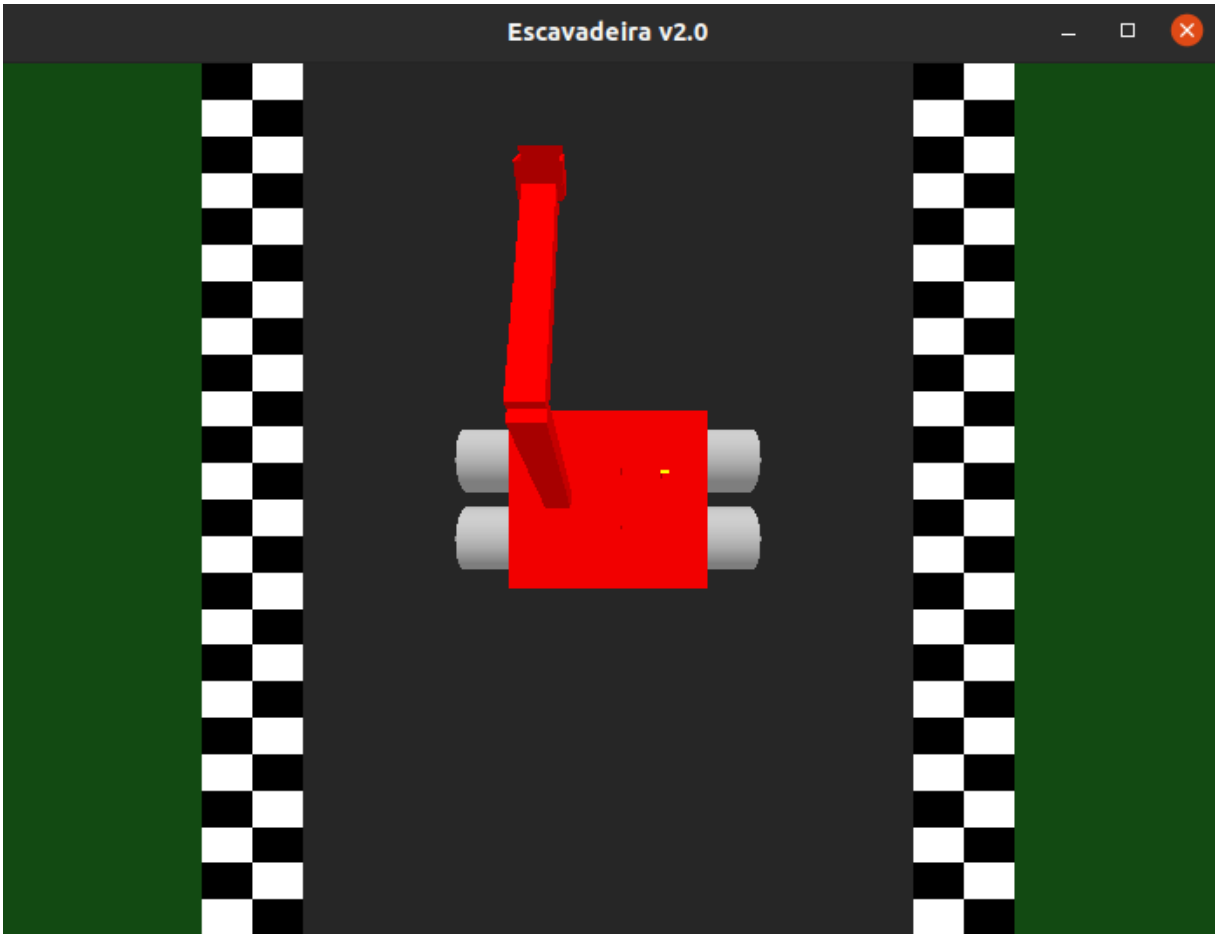
Quanto ao braço hidráulico, é composto por: um braço retangular, um antebraço retangular e uma pá representada pela união de cinco retângulos, sendo todos os desenhos geométricos sólidos. Isso conclui a parte superior.

Figura 2. Programa executando com luz “desligada”



A iluminação foi aplicada através do uso de uma fonte de luz nativa do OpenGL e dos tipos ambiente, difusa e especular; todos apresentados no decorrer da disciplina também. Inicialmente, a luz inicia com uma valor de cor branca ou quase branca, definida pela notação RGB, representando a luz ligada. Já a luz desligada é representada por uma cor quase preta, definida pela notação RGB. O resultado da animação de luz “desligada” e “ligada” é apresentado na Figura 2 e Figura 3, respectivamente.

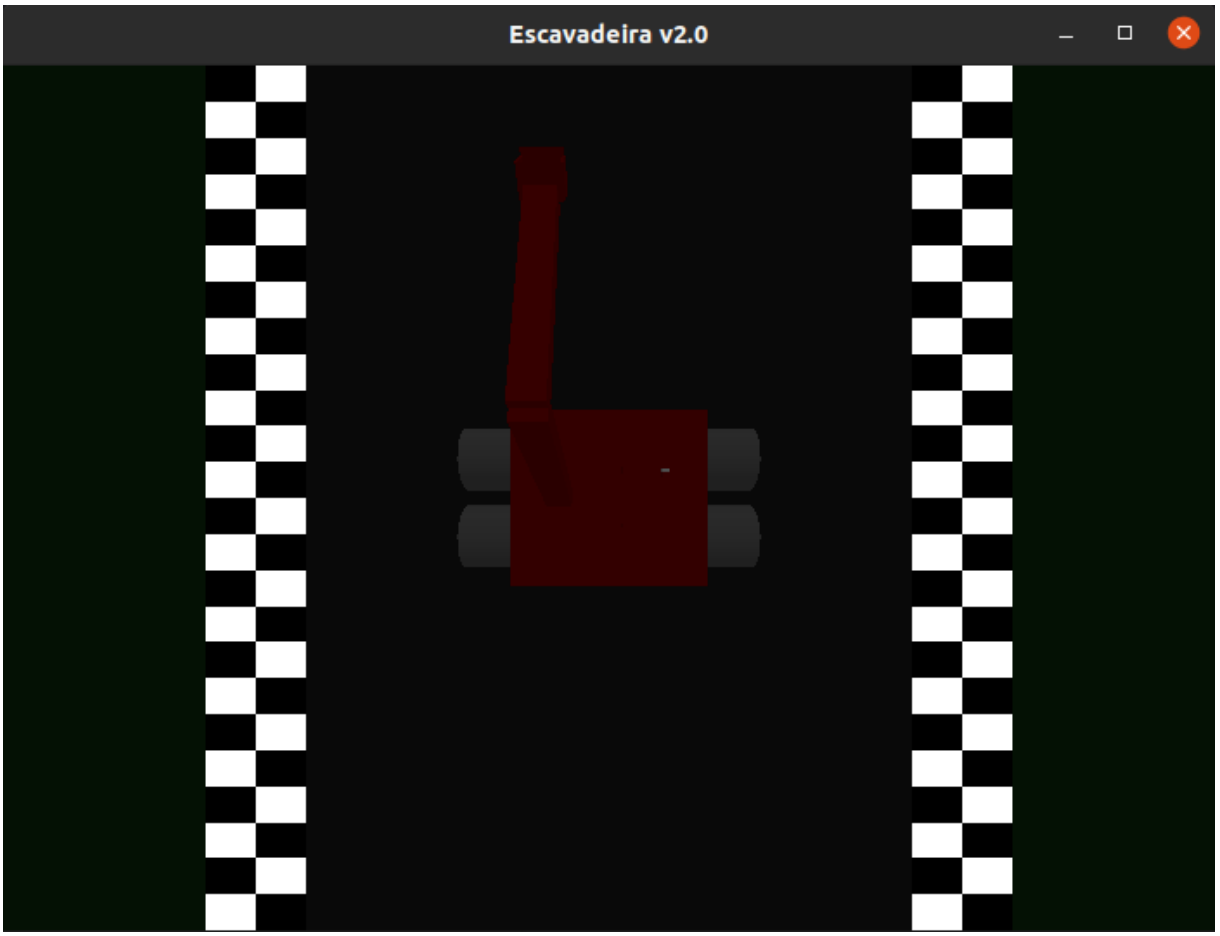
Figura 3. Objeto texturizado visto de cima com luz “ligada”



A textura aplicada no programa é similar a um tabuleiro de xadrez. Para aplicar uma textura, é necessário ligar os pontos do objeto geométrico aos pontos da imagem. A imagem distorceria caso fosse ligada diretamente a uma única e grande forma geométrica, uma vez que as bordas possuem um comprimento muito grande. Para evitar esse problema, as bordas foram desenhadas em parcelas inúmeras

vezes. As Figuras 3 e 4 mostram as bordas texturizadas vistas de cima ao ligar a luz e ao desligá-la, respectivamente, como se fossem calçadas iluminadas ao anoitecer.

Figura 4. Objeto texturizado visto de cima com luz “desligada”



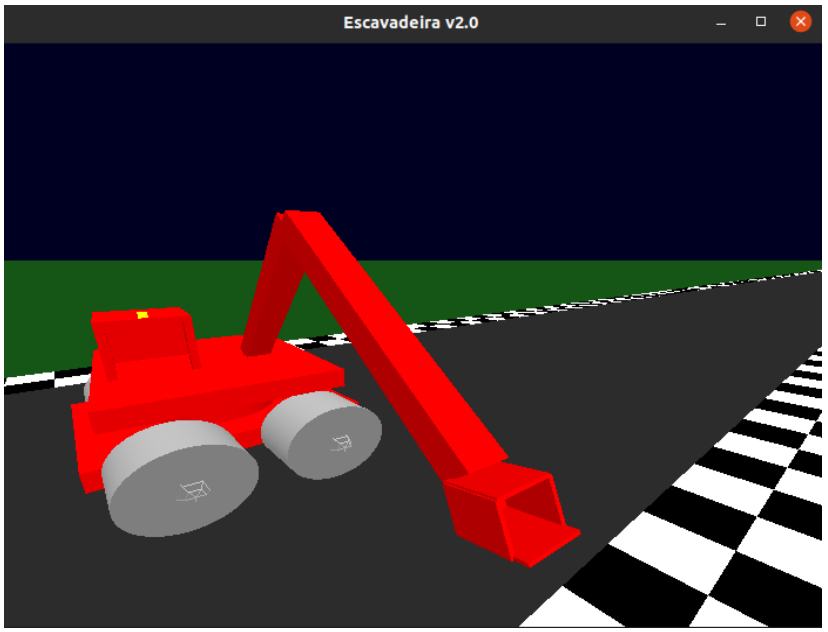
O conhecimento das transformações geométricas rotação e translação combinados com o mapeamento do teclado e do mouse, através do OpenGL e da API GLUT, permitiram desenvolver as animações do projeto, como o giro das rodas e os movimentos do braço da escavadeira.

A Tabela 3 enumera e descreve todas as interações que o programa aceita, enquanto na Figura 5 mostra o programa em um estado após computar vários desses comandos.

Tabela 3. Comandos disponíveis no programa escavadeira.cpp

COMANDO	DESCRIÇÃO
‘Q’ / ‘q’	Gira toda parte superior da escavadeira, nos dois sentidos, por meio da rotação da ‘cintura’
‘W’ / ‘w’	Movimenta, nos dois sentidos, o ‘braço’ da escavadeira por meio da rotação do ‘ombro’
‘E’ / ‘e’	Movimento, nos dois sentidos, o ‘antebraço’ da escavadeira por meio da rotação do ‘cotovelo’
‘R’ / ‘r’	Movimenta, nos dois sentidos, a pá da escavadeira por meio da rotação do ponto que liga a pá ao antebraço
‘A’ / ‘a’	Gira as 4 rodas simultaneamente, nos dois sentidos de rotação
‘+’ / ‘-’	Zoom in e zoom out
<space> / <backspace>	Movimenta a escavadeira para frente e para trás, e gira as 4 rodas simultaneamente, nos dois sentidos de rotação
‘↑’ / ‘↓’	Rotaciona o plano em torno do eixo x, nos dois sentidos
‘←’ / ‘→’	Rotaciona o plano em torno do eixo y, nos dois sentidos
<mouse left button>	‘Liga’ / ’Desliga’ luz e troca cor do farol

Figura 5. Escavadeira após aplicação de várias transformações



4.1 Limitações

O programa possui várias limitações devido aos prazos e abordagens da disciplina. A movimentação da escavadeira ao longa do plano é bastante restrita, aceitando apenas movimentações para frente e para trás, além de que não há fatores de aceleração. Também, a pista é uma linha reta. Para aumentar a diversão, curvas poderiam ser feitas.

O terreno é plano e apenas as bordas da pista apresentam texturas, melhorias nesses quesitos seriam interessantes. No terreno, também poderiam ser adicionados fatores diferentes de facilidade de locomoção entre “grama” e asfalto.

Outro fator é que, apesar de existirem as animações da escavadeira, a pá não escava de fato o terreno e extrai componentes. Isso leva a outro ponto, não há tratamento de colisão.

Antes da implementação da técnica de texturização discorrido no presente documento, tentou-se outras duas: pela biblioteca SOIL e stb_image. A primeira é uma biblioteca mais atual e foram encontrados poucos materiais na internet para apoiar a implementação. A segunda técnica já é mais consolidada, porém em conversas com colegas de turmas e em tutoriais descobriu-se que a implementação de texturização por esses métodos é dificultada pelo uso da biblioteca GLUT, sendo mais recomendado o uso da GLEW (OpenGL Extension Wrangler Library) e GLFW (Graphics Library Framework).

É importante comentar que para tornar o projeto escalável para melhorias e expansões futuras, seria necessário mudanças estruturais no código, como separação de arquivos de acordo com função ou classe de um componente, modularização do desenho dos componentes da escavadeira, reestruturação das variáveis de escopo global, entre outras.

5 CONCLUSÃO

Através do desenvolvimento deste projeto, foi possível abordar os seguintes temas e, também, apresentados no decorrer da disciplina: transformações geométricas, desenhos em 3D, interação com teclado e mouse, coloração, iluminação e textura.

É possível concluir que o presente projeto visou ensinar a utilização da API de computação gráfica OpenGL em C++, em conjunto com as bibliotecas GLUT e GLU,

o que de fato ocorreu. Porém, na visão dos integrantes do grupo, a disciplina poderia ter um viés mais teórico para expor as transformações e teoremas geométricos que alicercem a Computação Gráfica.

Como trabalhos futuros relativos ao presente projeto, pode-se focar nas implementações expostas na Seção 4.1 deste documento. O projeto pode ter como objetivo futuro um jogo de corrida ou um simulador de mineração, por exemplo.

REFERÊNCIAS BIBLIOGRÁFICAS

GOMES, J. & VELHO, L. *Computação Gráfica*. Vol. 1, Rio de Janeiro, IMPA, 1998.

MOLINA, Eduardo Perez. The Technological Roots of Computer Graphics. **Ieee Annals Of The History Of Computing**, [S.L.], v. 36, n. 3, p. 30-41, jul. 2014. Institute of Electrical and Electronics Engineers (IEEE). http://dx.doi.org/10.1109/mahc.2014.47.

Wikipedia, The Free Encyclopedia. *Computer Graphics*. Disponível em https://en.wikipedia.org/wiki/Computer_graphics_(computer_science)#:~:text=Computer%20graphics%20is%20a%20sub,dimensional%20graphics%20and%20image%20processing. Acesso em 13/03/2021.