

Metadados

- *Author: Raphael Lira dos Santos | RA: 223865*
- *Author: Matheus Percário Bruder | RA: 222327*
- *Created at: 15/04/2021*
- *File: "rocket.cpp"*

Exercício 01 - Rocket

Criar uma cena gráfica com a ilustração de um animal ou objeto em formato de malha triangular. A janela pode ter qualquer tamanho. Veja outros requisitos:

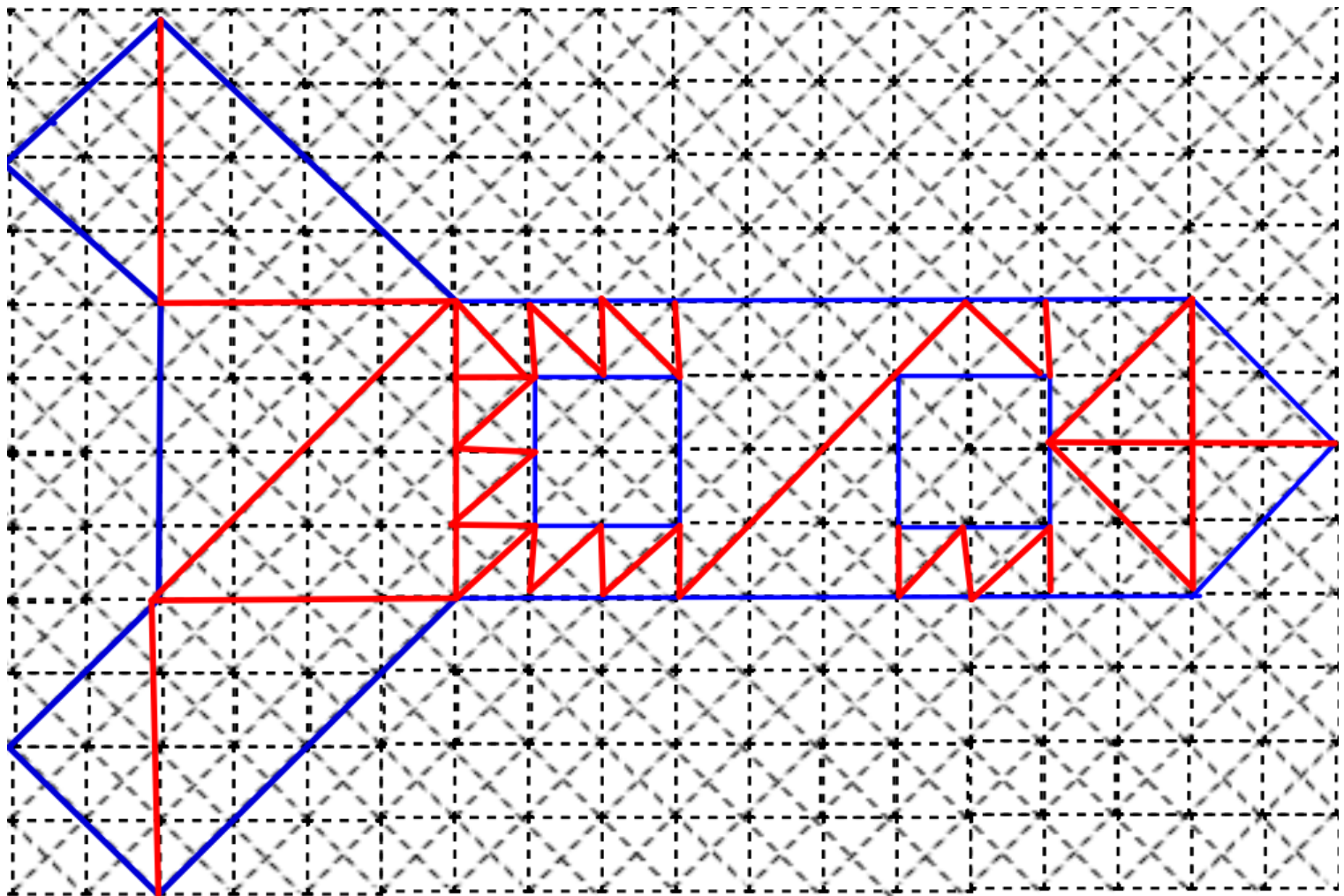
1. O animal ou objeto deve possuir um "buraco";
2. Os vértices do animal ou objeto podem estar em um arquivo txt ou no próprio programa C+OpenGL;
3. A cena deve alternar entre a apresentação do animal ou objeto por meio de seu contorno ou com preenchimento; essa ação deve ser executada via clique do mouse em qualquer parte da cena.

Explicação

Nós optamos por criar a cena gráfica de um foguete. Logo, para preencher os requisitos:

1. As janelas do foguete vão ser os "buracos necessários";
2. Os vértices dos triângulos que formam o foguete estarão no próprio programa;
3. Execute o programa e clique sobre o foguete para alternar seu estado!

Para estabelecer os vértices do foguete utilizamos a malha gráfica fornecida pelo professor, veja abaixo:

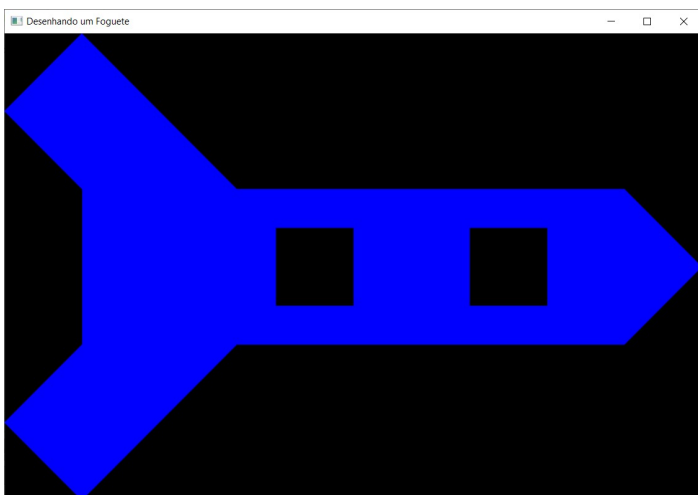


Ao todo existem 36 triângulos, os quais devem possuir vértices semelhantes, então, utilizando alguns recursos da biblioteca OpenGL serão unificados, dando origem ao desenho do foguete. Veja o código abaixo para entender como os triângulos foram representados e preenchidos.

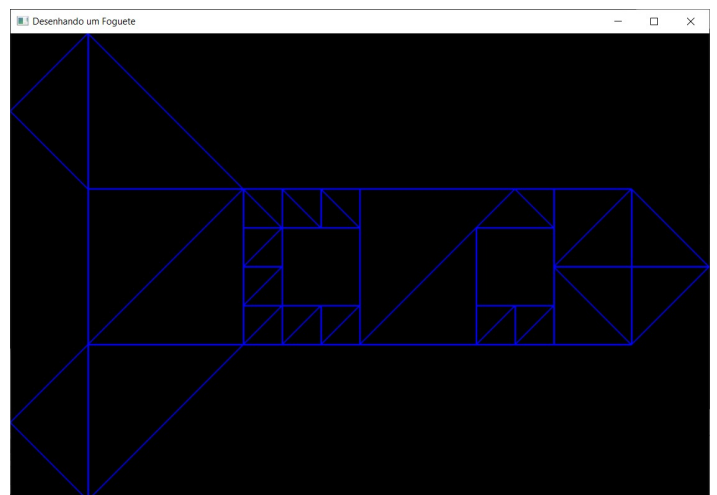
Formas de visualização

A seguir, as duas formas de visualização:

Preenchido



Contorno



Código fonte

```
#include <GL/glut.h>
#include <stdlib.h>

// prototipos das funcoes
void init(void);
void display(void);
void keyboard(unsigned char key, int x, int y);
void mouse(int botao, int state, int x, int y);

// Variaveis de controle
int is_preenchido = 0;

// funcao principal
int main(int argc, char** argv){
    glutInit(&argc, argv);                // inicializa o glut
    glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB); // especifica uso de cores e buffers
    glutInitWindowSize (900, 600);        // especifica as dimensoes da janela
    glutInitWindowPosition (100, 100);    // especifica aonde a janela aparece na tela
    glutCreateWindow ("Desenhando um Foguete"); // cria a janela
    init();
    glutDisplayFunc(display);              // funcao que sera redesenhada pelo GLUT
    glutKeyboardFunc(keyboard);            // funcoes de teclado
    glutMouseFunc(mouse);                  // funcoes de mouse
    glutMainLoop();                        // mostra todas as janelas criadas
    return 0;
}

// definicao de cada funcao

void init(void){
    glClearColor(0.0, 0.0, 0.0, 1.0);    // cor de fundo
    glOrtho(0, 900, 0, 600, -1, 1);      // modo de projecao ortogonal
}

void display(void)
{
    int i;
    glClear(GL_COLOR_BUFFER_BIT);        // limpa a janela
    glColor3f (0.0, 0.0, 1.0);          // cor da linha
    glLineWidth(2.0);
    glBegin(GL_TRIANGLE_STRIP);
    glVertex2i(800, 400);
    glVertex2i(900, 300);
    glVertex2i(800, 300);
    glVertex2i(800, 200);
    glVertex2i(700, 300);
    glVertex2i(700, 200);
    glVertex2i(700, 250);
}
```

```
glVertex2i(650, 200);
glVertex2i(650, 250);
glVertex2i(600, 200);
glVertex2i(600, 250);
glVertex2i(600, 350);
glVertex2i(600, 200);
glVertex2i(450, 200);
glVertex2i(450, 200);
glVertex2i(450, 250);
glVertex2i(400, 200);
glVertex2i(400, 250);
glVertex2i(350, 200);
glVertex2i(350, 250);
glVertex2i(300, 200);
glVertex2i(300, 250);
glVertex2i(300, 200);
glVertex2i(300, 200);
glVertex2i(100, 0);
glVertex2i(100, 200);
glVertex2i(0, 100);
glVertex2i(100, 200);
glVertex2i(100, 200);
glVertex2i(300, 400);
glVertex2i(100, 400);
glVertex2i(100, 600);
glVertex2i(0, 500);
glVertex2i(100, 400);
glVertex2i(100, 400);
glVertex2i(300, 400);
glVertex2i(100, 200);
glVertex2i(300, 200);
glVertex2i(300, 200);
glVertex2i(350, 250);
glVertex2i(300, 250);
glVertex2i(350, 300);
glVertex2i(300, 300);
glVertex2i(350, 350);
glVertex2i(300, 350);
glVertex2i(300, 400);
glVertex2i(300, 400);
glVertex2i(350, 350);
glVertex2i(350, 400);
glVertex2i(400, 350);
glVertex2i(400, 400);
glVertex2i(450, 350);
glVertex2i(450, 400);
glVertex2i(450, 400);
glVertex2i(450, 200);
glVertex2i(650, 400);
glVertex2i(600, 350);
glVertex2i(700, 350);
glVertex2i(650, 400);
glVertex2i(700, 400);
glVertex2i(700, 400);
glVertex2i(700, 300);
```

```
glVertex2i(800, 400);
glVertex2i(700, 300);
glVertex2i(800, 300);
glEnd();
glFlush();
}
```

```
void keyboard(unsigned char key, int x, int y){
    switch (key) {
        case 27: // tecla Esc (encerra o programa)
            exit(0);
            break;
    }
}
```

```
void mouse(int botao, int state, int x, int y){
    switch (botao) {
        case GLUT_LEFT_BUTTON:
            if(state == GLUT_DOWN && is_preenchido == 0){
                glPolygonMode(GL_FRONT_AND_BACK, GL_LINE);
                is_preenchido = 1;
            }else if(state == GLUT_DOWN){
                glPolygonMode(GL_FRONT_AND_BACK, GL_FILL);
                is_preenchido = 0;
            }

            display();

            break;
    }
}
```