
# Relatório Técnico

Version 1.0

Gustavo Lopes Rodrigues, Lucas Santiago de Oliveira

25 de novembro de 2021

# Conteúdo

# 1 Introdução

Esse programa feito para a disciplina de Computação gráfica tem como objetivo integrar C++, Qt e OpenGL para fazer uma aplicação que seja capaz de abrir arquivos .off (Object File Format) e manipulá-los. O programa, em grande parte, consiste na implementação do artigo *"Interactive Graphics Applications with OpenGL Shading Language and Qt"* com apenas algumas melhoras a sua interface e a adição da leitura de malha mista como uma nova funcionalidade.

Utilizamos C++ como a linguagem de programação, o Qt como o framework para construir a interface visual no qual o usuário interage e o OpenGL é a API gráfica que renderiza os objetos, processa shaders e texturas. As seções a seguir serão separadas da seguinte forma: primeiro apresentaremos a comunicação entre a interface visual do Qt e o código para em seguida apresentarmos o código em si e sua estruturação.

# 2 Qt

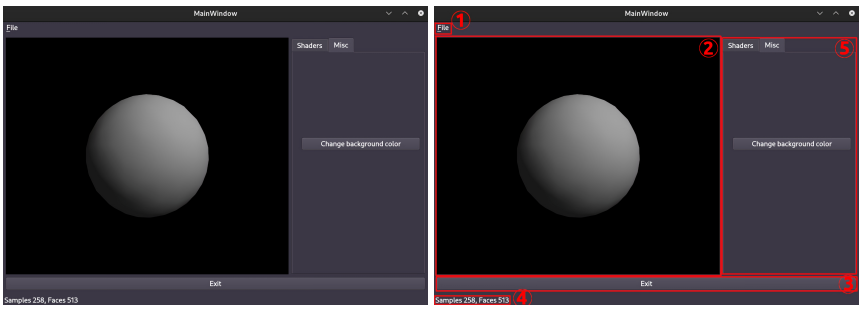


**Figura 1:** À esquerda temos a principal interface do programa e ao lado temos a mesma interface com marcações para sinalizar os principais componentes.

A área de interação do Qt é composta de Widgets, desde os botões, até o canvas do OpenGL, todos os elementos são Widgets que podem ser dimensionados na tela e mais importante, podem usar sinais para comunicar entre-si.

## 2.1 File

Primeiro temos o QMenuBar responsável em expandir e mostrar as duas opções presentes:

- **Open** - O primeiro QMenu possui um QAction, um sinal conectado com o visualizador dos objetos (QGLWidget). Quando acionado (triggered) realiza a ação de abrir o gerenciador de arquivos e abrir um arquivo .off.
- **Screenshot** - O outro QMenu possui um QAction que conecta ao QGLWidget. Quando acionado (triggered) realiza a ação de abrir o gerenciador de arquivos e permite o usuário salvar o frame atual do QGLWidget como uma imagem png.

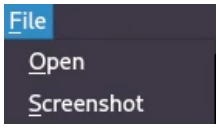


**Figura 2:** O botão **File** expandido

Além disso, o QMenuBar possui **Keybinds**, atalhos do teclado do usuário, que permitem que todas as iterações com o QMenuBar sejam mais rápidas. Eis a seguir os keybinds implementados:

- **Alt + f** $\rightarrow$ clicar no botão *File*
  - **Alt + f + o** $\rightarrow$ abrir um arquivo
  - **Alt + f + s** $\rightarrow$ salvar uma captura de tela

## 2.2 Visualizador de Objetos 3D

O principal componente da aplicação, a janela onde os objetos 3D são exibidos é um QGLWidget, em que há grande parte da implementação da API do OpenGL.

Esta janela ilustra o resultado final do processamento dos objetos, com texturas e shaders, se necessário. Além de capturar as teclas do teclado e o movimento do mouse.

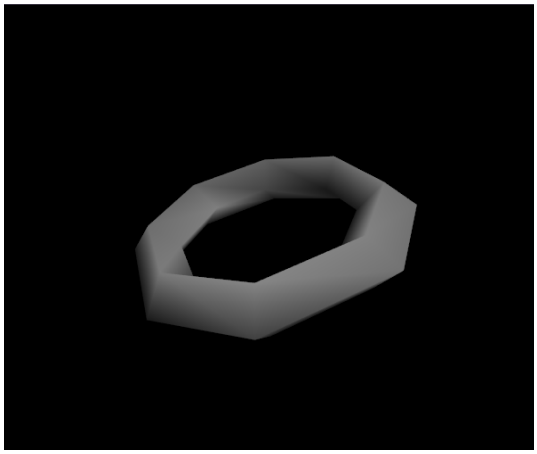

**Figura 3:** Objeto **octtorus.off** sendo exibido na janela

As possíveis entradas de teclado na aplicação são:
**Teclado:**

- **Número 1 do teclado** → aplicar o algoritmo de *Gouraud*
- **Número 2 do teclado** → aplicar o algoritmo de *Phong*
- **Número 3 do teclado** → aplicar uma textura
- **Número 4 do teclado** → aplicar uma textura com normal
- **Esc** → fechar a aplicação

**Mouse:**

- **Scroll** - Realizar ampliação/distanciamento (zoom) no objeto atualmente selecionado.
- **Botão direito + movimentar o mouse** - Rotacionar o objeto.

## 2.3 Botão de saída

QPushButton é um widget que possui um sinal ligado a janela principal do programa, ao ser clicado fechará a aplicação.


**Figura 4:** Botão para sair do programa

## 2.4 Barra de status

A barra de status fica no canto inferior da tela informando usuários de quantos *samples* e *faces* foram carregadas do objeto 3D. Esse possui um sinal ligado ao QGLWidget, que recebe uma QString com as informações mencionadas.


**Figura 5:** Barra de status com informações do **octtorus.off**

## 2.5 Abas de edição

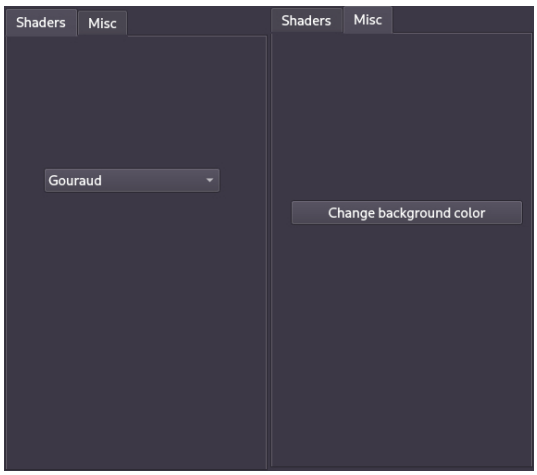


**Figura 6:** As duas abas de edição

————————- Para armazenar as possíveis, utilizamos um gerenciador de abas (QTabWidget), que armazenará duas abas(QWidget): uma responsável em guardar operações com shaders, e outra para trocar cor do fundo do QGLWidget. ————————-

- **Shaders** - Possui um QComboBox, que apresenta vários textos com nome dos possíveis shaders do programa. Após selecionar um, será acionado um sinal que será enviado para o QGLWidget pedindo para mudar para o shader especificado pelo usuário.
- **Misc** - Possui um QPushButton, que acionará a paleta de cor do Qt, permitindo que o usuário escolha uma cor para ser a nova cor do fundo (mais informações na seção do código).

## 2.6 Código

Agora que a comunicação na interface do Qt foi detalhada, podemos passar a explicar a comunicação dentro do código em si. Como dito anteriormente, o maior componente dentro do projeto que possui a grande porção do código é o QGLWidget. Ademais, temos a *main.cpp*, que inicia a aplicação e inclusive inicia a *mainwindow.cpp*, que é a janela principal, onde está o QGLWidget.

### 2.6.1 glwidget.cpp

Este é o código referente a tela do QGLWidget, encapsulando fazer todas as operações referentes ao OpenGL, além de carregar os arquivos.

# 3 Detalhando módelos matemáticos