**Bacharelado em Ciência da Computação**

**Computação Gráfica (SCC 250) - Trabalho Prático 2**

**Profª Agma (agma@icmc.usp.br)**

**PAE Eric (cabral.eric@usp.br)**

**Data de entrega: 25/06/2019**

## 1) Tonalização

Utilizando como plataforma inicial o trabalho 1 sobre preenchimento de polígonos já entregue, utilizando o algoritmo Scan-line para os objetos tridimensionais, implemente três modelos de tonalização:

- Flat
- Gouraud
- Phong (opcional, ponto extra no trabalho)

O usuário deve poder alternar entre os métodos de tonalização, mudar a posição da câmera e da fonte de luz.

## 2) Viewing

O sistema deve ser interativo, permitindo que o usuário altere o modo de visualização dos objetos como quiser, utilizando as técnicas de viewing da própria ferramenta (OpenGL). Permitir que o usuário manipule todos os parâmetros:

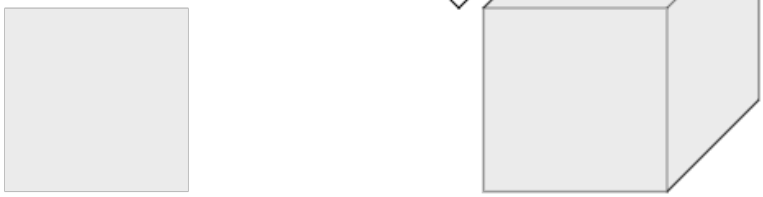
- Alternar entre projeções (ortogonal ou perspectiva)
- Manipular todos os parâmetros de cada projeção
- Inicie o programa com valores padrão que apresentem uma boa visualização da cena
- Função “reset” para voltar todos os parâmetros para os valores padrão

O usuário deve poder mudar a posição da câmera, para isso, implemente a função de câmera “LookAt”, onde o mundo é rotacionado em função da posição padrão da câmera (posicionada na origem com direção ao -z). Não utilize funções prontas de bibliotecas (p.e. gluLookAt), salvo a função LookAt do Qt.

## 3) Polígonos

O usuário deve informar os vértices do polígono a ser criado (vide Trabalho 1), demonstrando os polígono em criação por meio de um polígono convexo de linhas enquanto o usuário informa mais vértices. Deve ser utilizado o algoritmo de desenhar linhas.

Em seguida, dado um valor para a coordenada Z, o polígono 2D criado deve ser extrapolado em função da coordenada Z definida. Exemplificando:




Cada polígono deve ter uma cor vinculada a ele, de forma que o usuário possa informar a cor do polígono enquanto informa os vértices. O polígono deve ter apenas uma única cor.

## 4) Especificações

Utilize a plataforma vista nas aulas (Qt Creator). Caso decida utilizar outra plataforma, certifique-se de que a plataforma é compatível com o ambiente Linux. Informe instruções de instalação e compilação e crie um Makefile para o processo de compilação e execução.

O trabalho poderá ser feito em grupos de até 3 alunos. Fazer Upload do trabalho na pasta do Google Drive disponibilizada para cada grupo (caso seu grupo não possua uma pasta, contatar cabral.eric@usp.br). Na pasta:

- **src/:** Código fonte (C/C++)
- **Relatório.pdf:** Relatório do trabalho no formato PDF. Será descontada nota por falta de organização e legibilidade; Apresentação e discussão das considerações que cada grupo utilizou na sua implementação, com relação aos parâmetros e particularidades da apresentação;
- **README.md (Opcional):** Instruções de compilação e execução, lista de dependências caso existam, formato Markdown.

- **membros.txt:** Lista de membros. Nome e número USP separados por tabulação. Membros separados por linhas.

Assumirei que a data de entrega do trabalho será a última atualização feita nos arquivos da pasta, então qualquer mudança realizada após a data limite ( 25/06/2019 ) será considerada envio fora do prazo.

Podem ser utilizadas bibliotecas matemáticas do C/C++, tais como:

- OpenGL Mathematics
- As funções matemáticas do próprio Qt
- math.h

, nesse caso, as funcionalidades das bibliotecas externas devem ser devidamente documentadas e justificadas no relatório.

## 5) Critérios de avaliação

Será cobrado do aluno a implementação das rotinas de:

1. Desenhar linhas (1.0 pts)
2. Scan-line (7.0 pts + 2.0 pts)
    1. Preenchimento de polígonos (2.0 pts)
    2. Tonalização Flat e Gouraud (5.0 pts)
    3. Tonalização Phong (+ 2.0 pts)
3. Viewing (2.0 pts)
    1. Manipulação dos parâmetros de visualização (0.5 pts)
    2. Função de câmera (1.5 pts)

OBS.:

**1)** Implementar o algoritmo incremental de criação de retas. Caso utilize a função pronta de retas do OpenGL (GL_LINES), este critério vale metade da nota (0.5 pts).

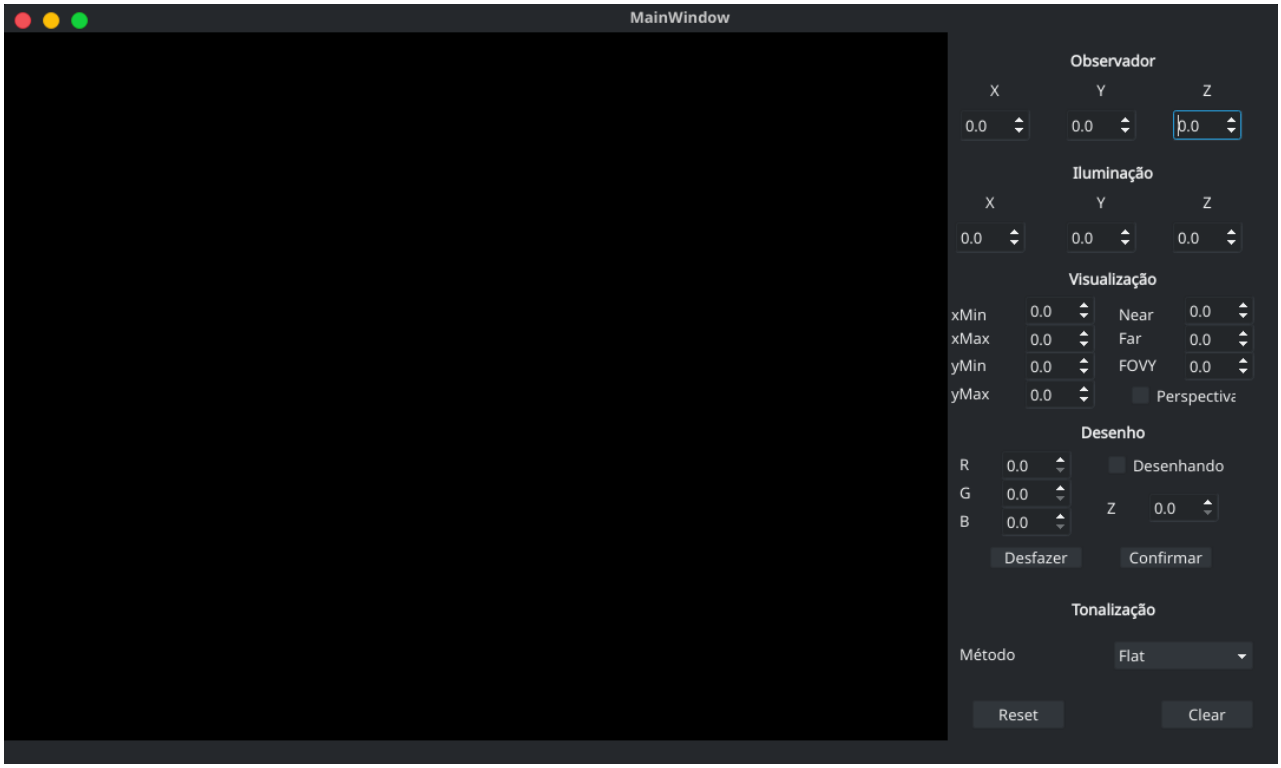
**2)** Implementar o algoritmo de scan-line para essa etapa. Caso o algoritmo não seja implementado e sejam utilizadas as funções de primitivas, iluminação e tonalização do OpenGL, este critério valerá metade da nota (3.5 pts).

3) Deve-se implementar a função de câmera LookAt realizando as transformações inversas como explicado em aula. Caso sejam utilizadas funções de câmera prontas, este critério valerá metade da nota (1.0 pts).

Espera-se que as rotinas sejam implementadas pelo grupo, possibilitando que o mesmo obtenha até 12.0 pts no trabalho. Caso o grupo não implemente as rotinas e utilize as rotinas prontas do OpenGL, a equipe pode obter até 5.0 pts.

## 6) Template

O template oferecido é apenas uma versão mínima das funcionalidades requeridas, sinta-se livre de fazer alterações ou de fazer um do zero.




### 6.1) Observações para usuários Windows

- No arquivo "`src.pro`", adicionar a linha:

```
LIBS += -lopengl32
```

- Modificar o nome das variáveis "far" e "near" para outros nomes. E modificar a definição da struct "Polygon" para outro nome.
    - Motivo: O sistema operacional tem essas palavras como reservadas.