

## Trabalho 4 - Geração Objeto 3D com Sweep e Curvas

### Ferramentas:

Linguagem C++, utilizando a API Canvas2D (disponível no [site da disciplina](#)) e IDE Code::Blocks, compilando com MinGW de 32 bits. **Não podem ser utilizadas bibliotecas auxiliares.** Desenvolva o trabalho sobre o demo 1\_CanvasGlut ou 2\_CanvasGlfw. Antes de enviar, retire todas as funções e arquivos não utilizados. Se alguém fizer em Linux, deve ajustar todos os paths para execução em Windows. Teste uma máquina Windows antes de enviar. O melhor neste caso é rodar o Windows em uma máquina virtual, como o Virtual Box ou Parallels.

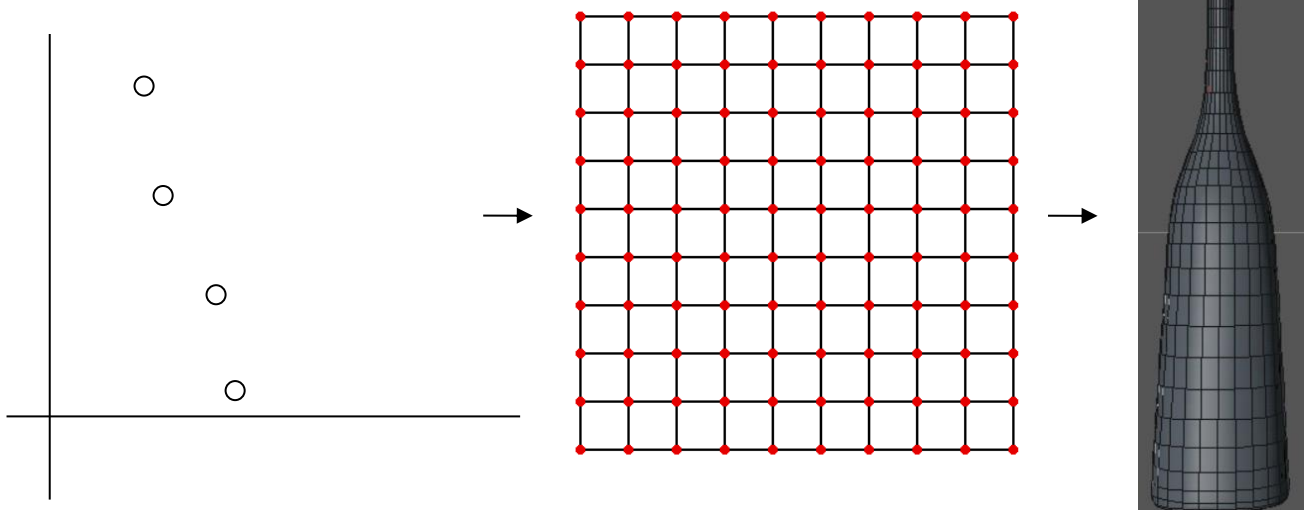
### Objetivos:

Explorar transformações geométricas, curvas, modelagem 3D, projeção em perspectiva, movimentação de câmera, visualização 3D.

### Descrição:

Implemente um programa para fazer modelagem e visualização 3D de objetos gerados sweep rotacional e curvas de Bezier.

O usuário deve informar uma sequência de 4 pontos de controle (ou mais). A partir desses pontos, deve-se gerar uma curva de Bezier (com N pontos amostrados) que deve ser rotacionada (sweep rotacional) para geração do objeto tridimensional.



Os pontos de controle podem ser editados a qualquer momento, e deve refletir **instantaneamente** na forma do objeto gerado. **Nada de clicar em botão para atualizar o modelo.** Crie interfaces gráficas apropriadas. O objeto também pode ir sendo gerado à medida que o usuário insere os pontos.

Requisitos básicos (Max: 9.0)

- Modelagem do objeto com opção de rotação e translação, para geração de objetos como garrafas, donuts e molas
- Visualização ortográfica e perspectiva (sob vários ângulos) em wireframe, com uso de **triângulos**
- Parametrização do número de pontos e faces do objeto gerado.

Avançados:

- Exibir vetores normais em cada face (até 1 ponto)
- Adição de mais de 4 pontos de controle (até 1 ponto)
- Adição de mais de um patch (até 1 ponto)
- Remoção de faces ocultas – sem preenchimento (até 2 pontos)
- Preenchimento de polígonos com z-buffer e scanline (uma das duas abaixo)
  - com Iluminação por vértice (até 4 pontos)
  - com Iluminação por pixel (até 5 pontos)
- textura, transparência, deformações, refração etc.

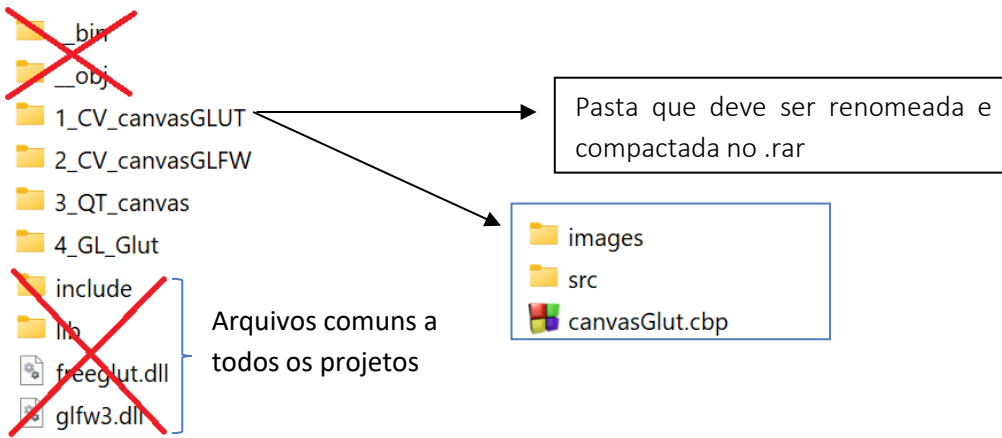
Dicas:



- Inicialmente implemente um programa para fazer a visualização de uma curva de Bezier em 2D com pontos estáticos definidos a mão.
- Quando todos os conceitos envolvidos estirem dominados, faça a implementação do trabalho.

O trabalho deve apresentar uma lista de instruções, explicando de forma como o usuário deve interagir com o programa. Enumere no início do código fonte (arquivo main.cpp) os quesitos que foram implementados.

### Formato de Entrega:

- O trabalho deve ser entregue pelo Google Classroom. Veja no arquivo cg\_1\_apresentacao.pdf instruções de envio.
- Deve-se utilizar como base o projeto 1\_CV\_canvasGLUT disponível nos demos da disciplina, como ilustrado na seguinte figura.



- A pasta `gl_1_canvasGlut` tem todos os códigos fonte e recursos (`images`, `src` e projeto). Esta pasta **deve ser renomeada** com o nome do aluno. Ex: **Trab1Maria**, **Trab2Paulo**, **Trab3Pedro**, **Trab4JoaoPedro** etc. Esta estrutura vai facilitar a execução e correção dos trabalhos. Todos os arquivos do trabalho devem estar dentro desta pasta, que deve ser a única pasta enviada, compactada em formato `.rar` (ex: **Trab1Maria.rar**). **Os caminhos relativos para as pastas `include`, `lib` e para as dlls devem ser mantidos, e no padrão Windows.**
- Ao utilizar a opção “Extrair Aqui” do WinRAR, deve ser criada apenas a pasta **Trab1Maria**, e não vários arquivos soltos. Os arquivos (`images`, `src`, `canvasGlut.cbp`, etc) devem estar imediatamente dentro desta pasta. Esta estrutura de pastas não pode ser modificada.
- Após enviar, baixe o arquivo enviado do Classroom, descompacte na pasta conforme o nome do aluno, compile (com opção Build/Rebuild ) e teste, para ter certeza que está correto, que não faltaram arquivos, e que os caminhos, principalmente para imagens, estejam corretos.
- Não devem ser enviadas as pastas `lib`, `include`, `bin`, `obj` pois são comuns a todos os projetos.
- Não devem ser enviados arquivos `lib`, `exe`, `obj`, `dll`, `pdf`, `doc`, `docx`.
- Retire todo código não utilizado no trabalho (arquivos, métodos, variáveis etc), bem como `printf`s de depuração, além do manual de uso da `canvas`.
- O trabalho será compilado em Windows .
- Coloquem seu nome no código fonte e também no título da Janela da `Canvas`.
- Perderá muita nota o trabalho que não seguir essas regras.

### Critérios de avaliação:

- Documentação: descrever no cabeçalho de cada arquivo a ideia geral do código e comentar o que cada método e classe faz.
- Clean code: estrutura do código e nomeação de métodos, classes e variáveis devem ser fáceis de ler e entender. Procurar manter o código o mais simples e organizado possível. Utilizar diferentes arquivos para diferentes classes.

- Pontualidade: Trabalhos não entregues na data não serão avaliados e receberão nota zero.
- Funcionalidade: o programa deve satisfazer todos os requisitos. Programas que não compilarem ou que não atenderem nenhum requisito receberão nota 0 (zero).
- Você pode discutir estratégias e ajudar o colega na implementação, porém evite passar código fonte. Programas semelhantes terão a nota 0 (zero).