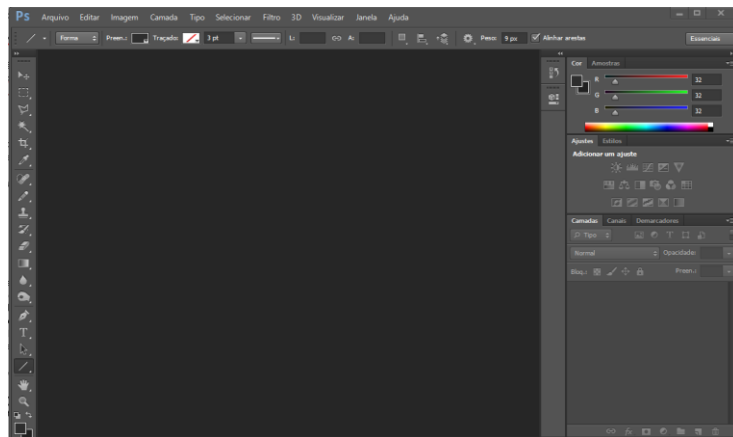


Universidade Federal de Santa Maria
Curso de Ciência da Computação
Disciplina: Computação Gráfica
Primeiro Semestre de 2025
Prof. Cesar Tadeu Pozzer
Data: 20/03/2025

Trabalho 1 – “Photoshop” caseiro



Ferramentas

Linguagem C++, utilizando a API Canvas2D (disponível no site da disciplina) e IDE Code::Blocks, compilando com MinGW 32 bits. **Não podem ser utilizadas bibliotecas auxiliares.** Desenvolva o trabalho sobre o demo 1_CV_canvasGLUT ou 2_CV_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. Deixe todos os caminhos relativos ao diretório que contém os arquivos .dll. 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.

A Canvas2D pode ser customizada, porém não é permitido o uso de funcionalidades OpenGL que não estão presentes na Canvas2D (ex: texturas, shaders). Pode-se criar sobrecargas de funções, novos métodos, classes, enums, etc.

Objetivos

Desenvolver habilidades na programação e manipulação de:

- Estruturação de código: Estruturas e/ou Orientação a objetos.
- Loop de renderização
- Componentes de interface gráfica e interação
- Manipulação de Cores
- Eventos de mouse
- Eventos de teclado
- Sistema de coordenadas

Descrição

Desenvolva um programa em C++ para fazer manipulações em imagens. O programa deve:

- Carregar imagens coloridas (RGB) em formato BMP sem compactação e 24bits. Utilize imagens retangulares pequenas (entre 200 e 300 pixels) para facilitar a exibição dentro da canvas2D. Imagens png ou jpeg podem ser convertidas para bmp usando o Paint.
- Possuir um menu lateral para controlar a aplicação.
- As imagens devem estar cada uma em uma “camada” diferente. As camadas devem ser desenhadas uma em cima da outra conforme uma ordem indicada no menu lateral. As camadas devem possuir uma transparência binária (um pixel pode ser totalmente transparente ou totalmente colorido e opaco).
- O plano de fundo do espaço destinado para a exibição das camadas deve ter um padrão reconhecível (quadriculado etc.) para identificar corretamente a transparência da camada mais inferior.
- Deve ser possível desenhar/pintar com o mouse (colorir pixels em um raio determinado ao redor do mouse ao segurar botão esquerdo) em uma camada considerada ativa. Só pode existir uma camada ativa por vez, e deve ter botões que trocam qual camada está ativa (ou seja, está recebendo modificações). Deve haver uma sinalização de qual camada é a ativa.
- Desenhos que forem em cima de uma imagem devem sobrepor a cor original (ou seja, se implementar bônus de movimento da imagem, o desenho deve acompanhar).
- Deve ter um modo para trocar a cor utilizada no desenho com o mouse.
- Deve ser possível reordenar as camadas e sua ordem de exibição durante a execução. Também deve ser possível ocultar qualquer camada
- No menu lateral, deve haver pelo menos um Botão, uma Checkbox e um Slider, todos devem ter um propósito (não podem existir controles que não servem para nada).
- Adicionar opção para aplicar os seguintes efeitos para a camada ativa:
 - Flip Horizontal e Vertical
 - Controle de Brilho

Sugere-se que seja implementado um “manager” de botões, panels, checkbox, sliders, scrolls etc. Utilize seus conhecimentos de programação. Esse manager (uma classe) poderá ser usado nos demais trabalhos da disciplina. Um slider pode ser usado, por exemplo, para ajustar o brilho de uma imagem.

Para facilitar a implementação da interface de carregamento de imagens, pode-se:

- criar botões que carregam a imagem “a.bmp”, “b.bmp” e “c.bmp”. Não é necessário carregar mais que 3 imagens, mas o programa deve ser genérico para N imagens.
- As imagens devem estar no diretório images (como está no demo da canvasGLUT). Deve-se enviar as três imagens juntamente com o trabalho, com tamanhos diferentes e resoluções pares e ímpares.

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 (obrigatórios e extras) que foram implementados. Explique em detalhes como utilizar cada funcionalidade do programa.

Faça um planejamento (numa folha de papel) de como os elementos serão distribuídos dentro da canvas2D. O programa não precisa ser bonito, mas tem que ser bem organizado e de fácil utilização. Defina se o Y cresce para cima ou para baixo.

Avaliação:

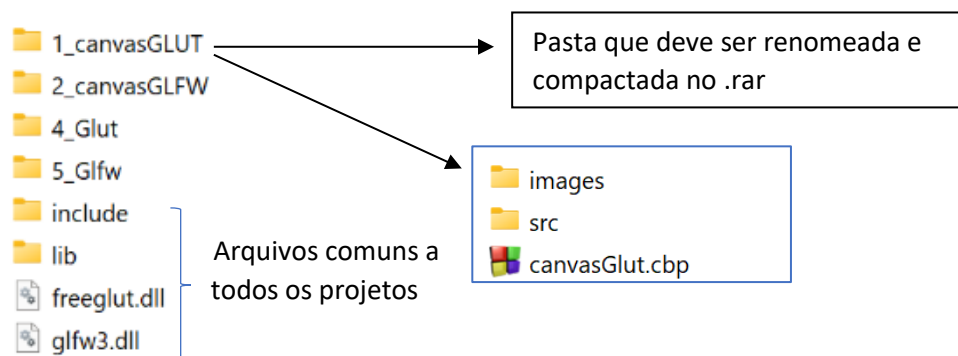
Será avaliada a estruturação do código (nomes de variáveis, funções, métodos, comentários, etc). Trabalhos com códigos replicados perderão **muita** nota (ex: um código para tratar cada botão individualmente). Não utilize números mágicos. Deve-se utilizar laços de repetição sempre que possível. Pra quem não tem muito conhecimento em C++, utilizar estruturas e arrays para organizar e generalizar o código.

Bônus (Para nota acima de 9.0):

- Adicionar efeitos extras:
 - Tons de Cinza (até 0.5 pt)
 - Correção de Gama (até 0.5 pt)
 - Contraste (até 0.5 pt)
 - Desfoque Gaussiano (até 2 pt)
- Redimensionamento e movimentação das imagens (até 2 pts)
- Aplicar uma rotação de ângulo qualquer para as imagens. A rotação deve ser feita com o uso do mouse. (até 2 pts)
- Diferentes tipos de pincel/formas para desenhar com o mouse (até 2 pts)
- Salvar e carregar estado atual do programa em arquivo: ordem e visibilidade das camadas, desenhos, posição imagens, efeitos etc. (até 3 pts)


Formato de Entrega:

- O trabalho deve ser entregue pelo Google Classroom.
- Deve-se utilizar como base o projeto 1_CV_canvasGLUT disponível nos demos da disciplina, como ilustrado na seguinte figura.



A pasta 1_CV_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, cujo nome deve ser o nome do aluno. Ex: FulanoSobrenome.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 Trab1Fulano, e não vários arquivos soltos. Os arquivos (images, src, canvasGlut.cbp, etc) devem estar imediatamente dentro desta pasta Trab1Fulano.
- Esta estrutura de pastas não pode ser modificada.

- 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 .
- Dicas: coloquem seu nome no código fonte e também no título da Janela da Canvas.
- Perderá muita nota o trabalho que não seguir essas regras.
- Conferir se o caminho das imagens está correto e está de acordo com o nome da pasta do trabalho (Trab1Fulano) antes de enviar.
- Após enviar, certificar que o arquivo foi enviado e não é inválido.

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).