Trabalho - Unidade I

Curso: Engenharia da Computação Orientador: Agostinho Júnior Data: 2025-10-22 :toc: macro :toclevels: 3 :imagesdir: ./img

Introdução

Este relatório apresenta a resolução dos desafios propostos pelo orientador ao final de cada capítulo da Unidade I da disciplina de Processamento Digital de Imagens. Os objetivos incluem:

- Resolver os problemas apresentados;
- Realizar discussões sobre os temas relacionados a cada questão;
- Analisar os resultados obtidos;
- Explorar novas ferramentas que contribuam para o conhecimento e para o desenvolvimento do currículo.

Ferramentas Utilizadas

Para a execução das atividades, foram empregadas as seguintes ferramentas:

- **OpenCV** — para manipulação de imagens e vídeos;
- **asciidoc** — utilizado para a criação deste documento de registro das atividades;
- **makefile** — para a compilação e execução dos scripts;
- **C** — linguagem predominante para a implementação das soluções.

Metodologia

A resolução dos desafios foi realizada seguindo os seguintes passos:

1. Leitura e interpretação do enunciado de cada desafio;
2. Planejamento da solução utilizando **C** e bibliotecas auxiliares;
3. Implementação do código e teste de resultados;
4. Análise crítica dos resultados obtidos;
5. Registro das conclusões no documento utilizando **asciidoc**.

Manipulação de pixels em uma imagem

Enunciado

Utilizando o programa da Listagem 4, **pixels.cpp**, como referência, implemente um programa

regions.cpp. Esse programa deverá solicitar ao usuário as coordenadas de dois pontos P1 e P2 localizados dentro dos limites do tamanho da imagem fornecida. A região definida pelo retângulo de vértices opostos definidos pelos pontos P1 e P2 será exibida com o negativo da imagem na região correspondente. O efeito é ilustrado na Figura 4, "Exemplo de saída do programa **regions.cpp**".

Resumo do Programa

O programa realiza as seguintes etapas principais:

1. **Carregamento da imagem** A imagem é carregada usando **OpenCV** (`cv::imread`). Caso a imagem não seja encontrada, o programa lança uma exceção.
2. **Entrada de coordenadas pelo usuário** O usuário insere os pontos P1 e P2, que definem a região do retângulo a ser processado.
3. **Validação das coordenadas** A função `getPixelSafe` verifica se as coordenadas fornecidas estão dentro dos limites da imagem. Se forem inválidas, é lançada a exceção `ImageException`.
4. **Aplicação do efeito negativo na região** A função `negative_image` percorre cada pixel da região definida pelos pontos P1 e P2 e aplica a transformação negativa:

```
for (int i = p1.getX(); i <= p2.getX(); i++) {
    for (int j = p1.getY(); j <= p2.getY(); j++) {
        pixel = image.at<cv::Vec3b>(i, j);
        pixel[0] = 255 - pixel[0];
        pixel[1] = 255 - pixel[1];
        pixel[2] = 255 - pixel[2];
        image.at<cv::Vec3b>(i, j) = pixel;
    }
}
```

1. **Exibição da imagem** A imagem resultante é exibida em uma janela usando `cv::imshow`. O programa aguarda a tecla do usuário antes de encerrar.

Conceito do Efeito Negativo

O **efeito negativo** de uma imagem consiste em inverter as cores de cada pixel. Para cada canal de cor \(C\) (B, G, R), a transformação é dada pela equação:

C_{negativo} = 255 - C_{original}

Onde:

- \(C_{original}\) é o valor do pixel no intervalo [0, 255]
- \(C_{negativo}\) é o valor do pixel após a inversão

O efeito faz com que cores claras se tornem escuras, cores escuras se tornem claras, e cada tonalidade seja invertida em relação ao valor máximo.

Instruções de Execução

1. Certifique-se de ter **OpenCV** instalado e configurado para compilação em **C++**.
2. Compile o programa com o cmake:
3. Insira as coordenadas dos pontos P1 e P2 quando solicitado.

Figura abaixo mostra o efeito negativo aplicado na região da imagem definida pelos pontos P1 e P2:

Visualização de imagens

Enunciado

Utilizando o programa da Listagem 10, “visualizacao.cpp” como referência, prepare os arquivos matriz.txt e line.txt para as imagens sementes1.png e sementes2.png mostrada nas Figura 9, “Sementes 1” e Figura 10, “Sementes 2”. Utilize o programa gnuplot para realizar as visualizações mostradas nessa lição para ambas as imagens. Compare os gráficos gerados e discuta o que pode ser observado acerca da iluminação das cenas nas duas imagens.

O que é um Histograma de Imagem

Um histograma de imagem é uma representação gráfica da distribuição de intensidades de pixel em uma imagem digital. Ele mostra, em forma de barras, a quantidade de pixels para cada nível de brilho, variando geralmente de 0 (preto) a 255 (branco) em imagens em escala de cinza.

Essa ferramenta é fundamental no processamento digital de imagens, pois permite:

- Avaliar o contraste da imagem.
- Identificar áreas superexpostas ou subexpostas.
- Guiar técnicas de realce, como equalização de histograma.
- Auxiliar na segmentação e na detecção de bordas.

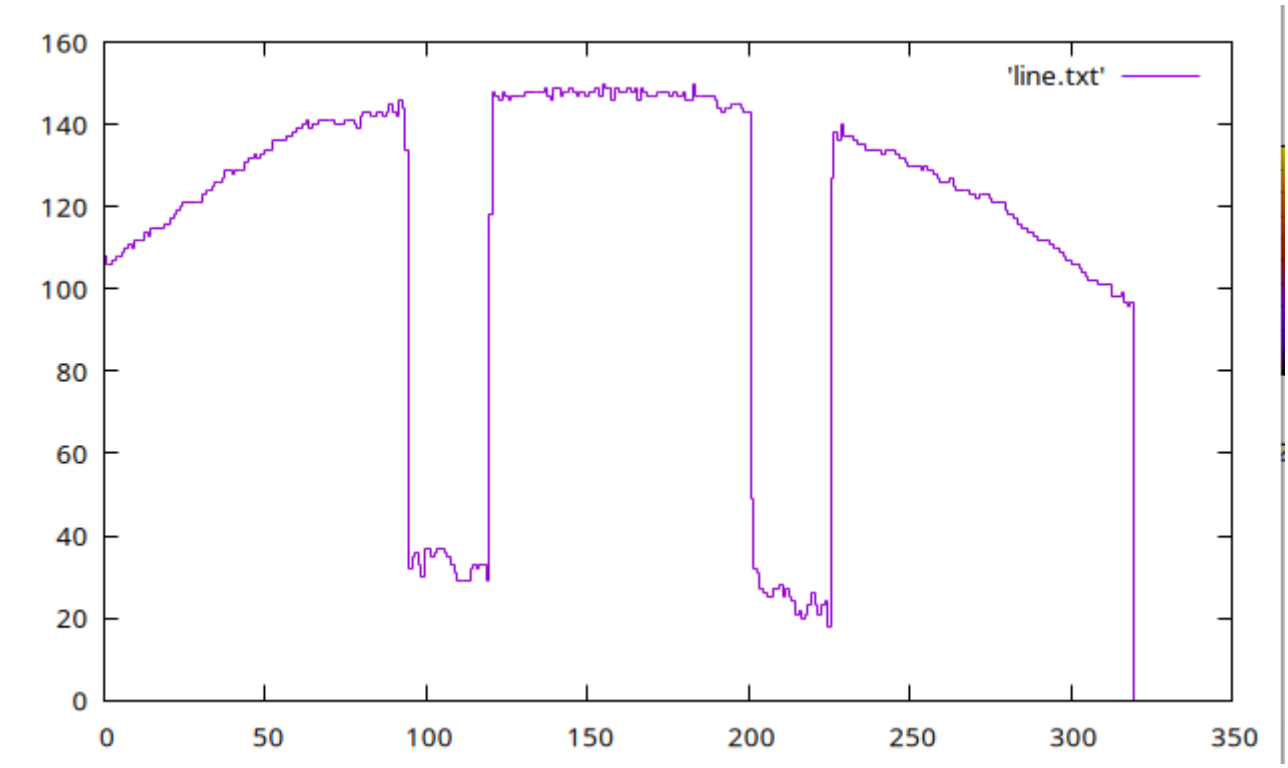
Em imagens com boa distribuição de intensidade, o histograma tende a ocupar uma faixa ampla de valores. Já em imagens com iluminação desigual ou baixo contraste, o histograma pode se concentrar em uma faixa estreita, indicando perda de detalhes visuais.

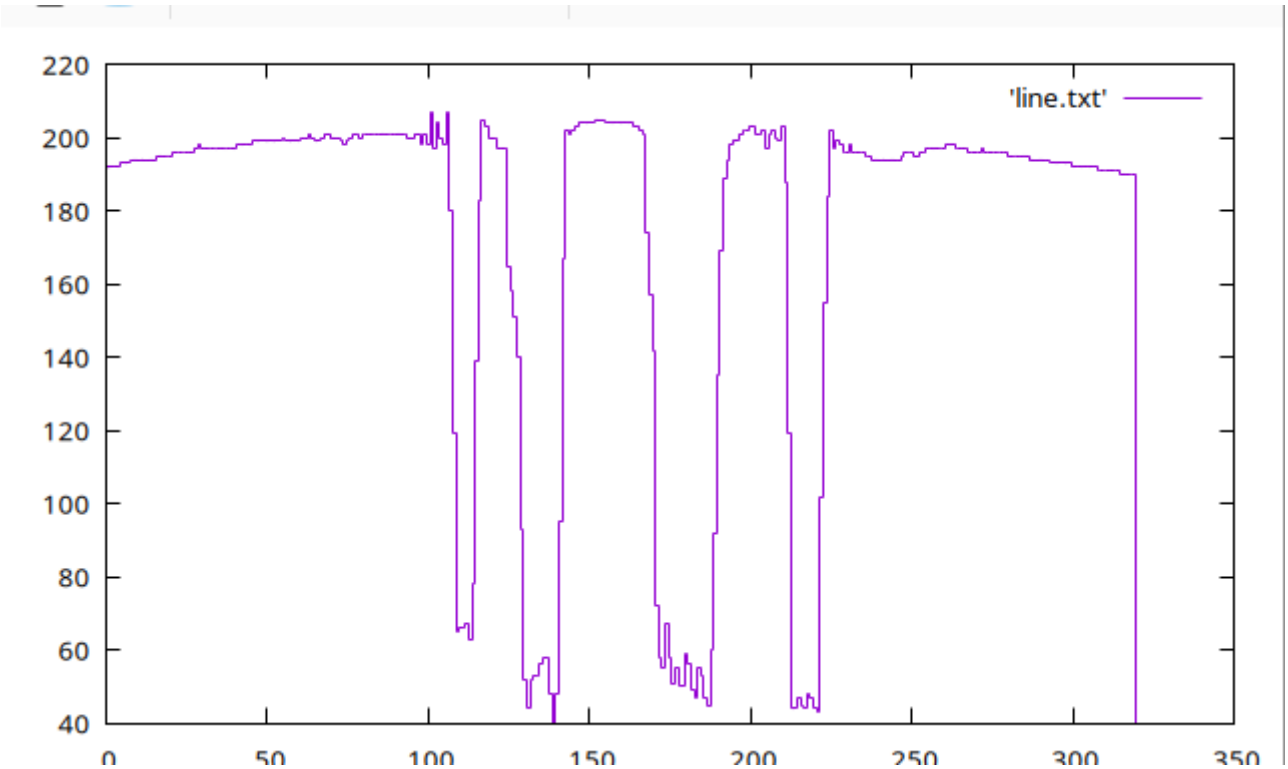
Histogramas de Intensidade

Foram gerados os histogramas para duas imagens com pouca iluminação (sem flash) e com a iluminação melhorada (com flash), sendo essa última com objetivo de realçar os contornos dos objetos inseridos no espaço em branco. Os histogramas revelam diferenças significativas entre as duas imagens:

- **Imagem com flash**: apresenta distribuição concentrada em valores altos, indicando predominância de tons claros. O contraste é reduzido, o que suaviza os contornos e pode ocultar detalhes importantes.

- **Imagem sem flash**: exibe uma distribuição mais ampla de intensidades, com presença marcante de tons escuros e médios. O contraste natural favorece a definição de bordas e texturas.



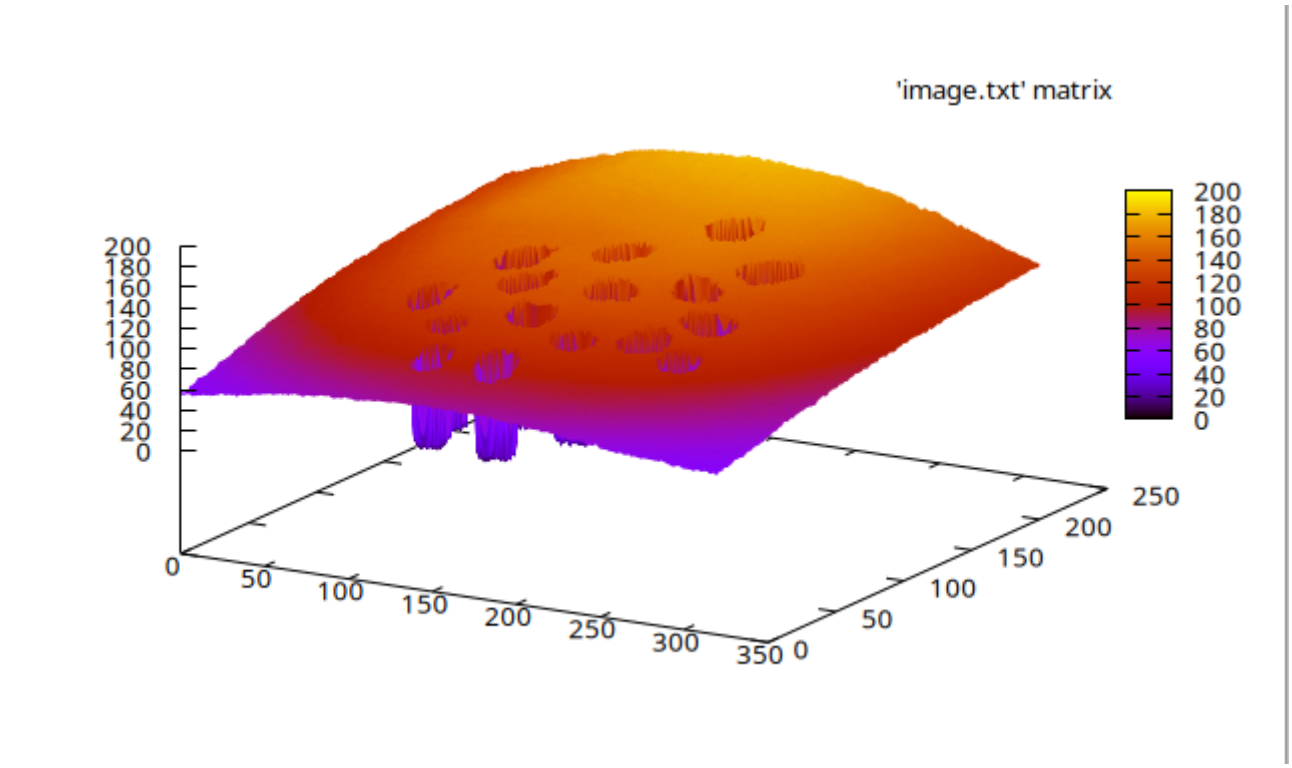


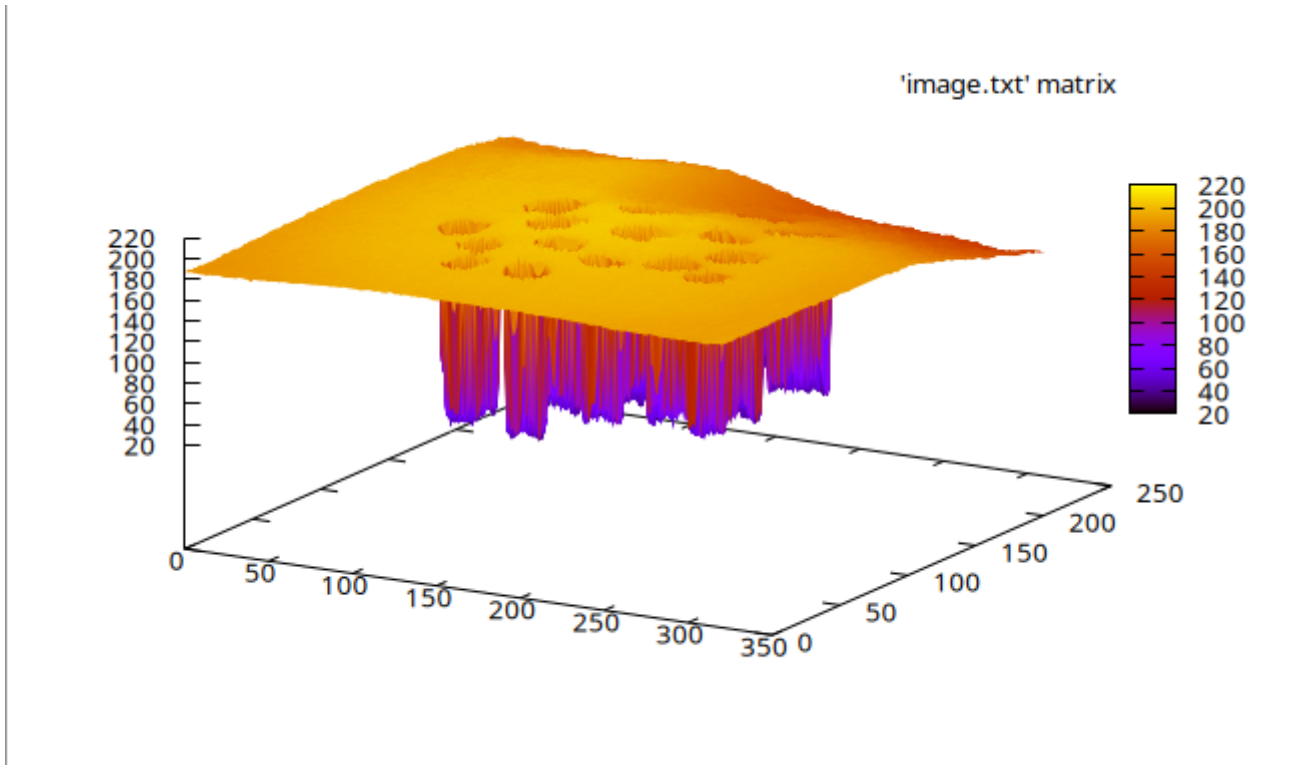
Visualizações 3D das Matrizes de Intensidade

As visualizações tridimensionais permitem observar a topografia da imagem em termos de brilho:

- **Imagem com flash**: a superfície é mais plana e elevada, com poucos vales. Isso indica uma iluminação uniforme que reduz variações locais.

- **Imagem sem flash**: a superfície apresenta depressões e picos irregulares, evidenciando áreas de sombra e contorno. Essa variação é ideal para aplicações de segmentação e detecção de bordas.





A imagem com flash é útil para reconhecimento geral de formas em ambientes escuros, mas pode exigir técnicas de normalização para compensar saturações. Já a imagem sem flash é mais adequada para algoritmos de realce de bordas, como Sobel ou Canny, e para reconstrução de formas baseadas em variações de intensidade.