

Sistemas Multimídia e Hipermídia

Marcelo Hashimoto

Última Atualização: 13 de maio de 2015

Nesta disciplina, cada grupo deve desenvolver um **projeto** ao longo de todo o semestre e um **mini-projeto** apenas no final.

1 Conceitos Básicos

As definições apresentadas nesta seção são utilizadas tanto no projeto quanto no mini-projeto.

1.1 Imagens Digitais

Sejam w e h inteiros positivos. Uma **imagem digital** de largura w e altura h é uma função

$$I: \{0, \dots, h-1\} \times \{0, \dots, w-1\} \rightarrow \{0, \dots, 255\}.$$

Os elementos do domínio são chamados de **pixels** e os elementos da imagem são chamados de **níveis de cinza** ou **valores de cinza**. Quando necessário, vamos supor que $I(y, x) = 0$ quando (y, x) não pertence ao domínio.

1.2 PGM Simple

Vamos considerar uma versão restrita do formato **PGM** [?], chamada neste documento de **PGM Simple**. Um arquivo nesse formato contém uma imagem digital definida pela seguinte sequência:

1. o caractere 'P' em ASCII;
2. o caractere '5' em ASCII;
3. uma quebra de linha, ou seja, o caractere '\n' em ASCII;
4. um inteiro positivo w representando a largura, composto por dígitos decimais em ASCII;
5. um espaço, ou seja, o caractere ' ' em ASCII;
6. um inteiro positivo h representando a altura, composto por dígitos decimais em ASCII;
7. uma quebra de linha;
8. o inteiro positivo 255, composto por dígitos decimais em ASCII;
9. uma quebra de linha;
10. $h \cdot w$ bytes representando os níveis de cinza dos pixels, ordenados primeiro por linha e depois por coluna.

2 Projeto

O projeto consiste em um *processador de imagens com interface gráfica*. O desenvolvimento será dividido em *módulos*.

2.1 Entrega

Cada módulo deve ser entregue pelo *Blackboard* até 19:00 de sua respectiva data.

2.2 Avaliação

Cada módulo será avaliado de acordo com cinco critérios: **adequação, eficiência, organização, clareza e documentação**. Cabe observar que **o código deve ser original**. Trechos de outros autores *podem ser estudados e adaptados se necessário mas não podem ser simplesmente copiados e colados*.

A avaliação consiste em uma breve apresentação do grupo, seguida por perguntas do professor. Cabe observar que **tudo conhecimento adquirido individualmente deve ser compartilhado**, pois as perguntas podem ser sobre *qualquer parte* do módulo para *qualquer membro* do grupo. Uma resposta individual insatisfatória será considerada na avaliação *mesmo que o restante do grupo seja capaz de produzir respostas satisfatórias*.

2.3 Módulo 1 (25/02): Leitura, Exibição e Escrita

O processador deve **carregar, mostrar e salvar** um arquivo no formato PGM Simples. A imagem digital contida nesse arquivo deve ser armazenada em uma matriz de bytes. Por enquanto, a localização do arquivo pode ser *hardcoded*.

As funções da plataforma para leitura e escrita de imagens não podem ser utilizadas na implementação dessas funcionalidades, mas podem ser utilizadas para a renderização de botões e outros elementos da interface.

2.4 Módulo 2 (04/03): Interface Gráfica e Rotação

Primeiramente, a interface gráfica das funcionalidades do Módulo 1 deve ser implementada: o processador deve apresentar um **botão para carregar** e um **botão para salvar**, ambos funcionais. A localização do arquivo deve ser obtida através do *native dialogs addon*.

Além disso, o processador deve apresentar um **botão para rotacionar 90° à esquerda** e um **botão para rotacionar 90° à direita**, ambos funcionais.

Por fim, o processador deve **redimensionar a janela automaticamente** ao carregar ou rotacionar uma imagem. Espera-se que o tamanho da janela sempre seja o mínimo necessário para exibir a imagem e os botões.

2.5 Módulo 3 (11/03): Histórico e Refinamentos

O processador deve apresentar um **botão para desfazer operações** e um **botão para refazer operações**, ambos funcionais. A implementação dessas funcionalidades deve utilizar um *histórico* de operações, sem limite de tamanho. Cabe observar que as operações que serão implementadas em módulos posteriores são *destrutivas*, portanto a funcionalidade de desfazer não pode depender de reversibilidade.

Além disso, o processador deve **desabilitar** um botão quando a operação associada não for possível. Salvar, por exemplo, não é possível se nenhuma imagem foi carregada. A desabilitação deve ter algum indicador visual.

2.6 Módulo 4 (18/03): Redução de Cores

Seja n um inteiro entre 2 e 255 e considere o conjunto

$$\mathcal{R} = \left\{ i \cdot \left(\frac{255}{n-1} \right) : i = 0, \dots, n-1 \right\} = \left\{ 0 \cdot \left(\frac{255}{n-1} \right), 1 \cdot \left(\frac{255}{n-1} \right), \dots, (n-1) \cdot \left(\frac{255}{n-1} \right) \right\}.$$

O processador deve apresentar um **botão para reduzir cores**. A implementação dessa funcionalidade deve requisitar n do usuário e, para cada pixel da imagem, substituir o nível de cinza pelo valor mais próximo em \mathcal{R} .

Cabe observar que um valor em \mathcal{R} pode não ser inteiro. A escolha do procedimento de aproximação é livre, mas deve ser justificada.

2.7 Módulo 5 (25/03): Algoritmo de Floyd-Steinberg

Considere uma função $\rho: \{0, \dots, 255\} \rightarrow \mathcal{R}$ tal que $\rho(i)$ é o valor mais próximo de i em \mathcal{R} , ou seja, ρ representa a funcionalidade implementada no Módulo 4.

Considere também uma função $\nu: \mathbb{Z} \rightarrow \{0, \dots, 255\}$ tal que $\nu(i) = \min\{\max\{0, i\}, 255\}$, ou seja, ν representa a normalização de um inteiro para o intervalo entre 0 e 255.

O processador deve apresentar um **botão para executar o Algoritmo 1**. Assim como no Módulo 4, a implementação deve requisitar n do usuário.

Algoritmo 1: Floyd-Steinberg

```
1 para todo  $y$  de 0 a  $h - 1$ 
2   para todo  $x$  de 0 a  $w - 1$ 
3     antigo  $\leftarrow I(y, x)$ 
4      $I(y, x) \leftarrow \rho(I(y, x))$ 
5     erro  $\leftarrow$  antigo  $- I(y, x)$ 
6      $I(y + 0, x + 1) \leftarrow \nu(I(y + 0, x + 1) + (7/16) \cdot \text{erro})$ 
7      $I(y + 1, x - 1) \leftarrow \nu(I(y + 1, x - 1) + (3/16) \cdot \text{erro})$ 
8      $I(y + 1, x + 0) \leftarrow \nu(I(y + 1, x + 0) + (5/16) \cdot \text{erro})$ 
9      $I(y + 1, x + 1) \leftarrow \nu(I(y + 1, x + 1) + (1/16) \cdot \text{erro})$ 
```

2.8 Módulo 6 (01/04): Equalização de Histograma

O *histograma* de I é uma função $\delta: \{0, \dots, 255\} \rightarrow \{0, \dots, h \cdot w\}$ tal que $\delta(i)$ é a quantidade de pixels em I cujo valor é i . O *histograma cumulativo* de I é uma função $\Delta: \{0, \dots, 255\} \rightarrow \{0, \dots, h \cdot w\}$ tal que

$$\Delta(i) = \sum_{j=0}^i \delta(j).$$

Seja Δ_{\min} o menor valor positivo do histograma cumulativo e seja Δ_{\max} o maior valor positivo. *Equalizar* a imagem I significa substituir cada valor i por

$$\frac{\Delta(i) - \Delta_{\min}}{\Delta_{\max} - \Delta_{\min}} \cdot 255.$$

O processador deve apresentar um **botão para equalizar**. O procedimento de aproximação é livre.

2.9 Módulo 7 (08/04): Filtro da Média

Seja n um inteiro positivo ímpar maior ou igual a 3. A *vizinhança* de tamanho n do pixel (y, x) é o conjunto

$$\mathcal{N}(y, x) = \{(y', x') : y - r \leq y' \leq y + r \text{ e } x - r \leq x' \leq x + r\}, \text{ onde } r = \frac{n-1}{2}.$$

O processador deve apresentar um **botão de filtro da média**. A implementação dessa funcionalidade deve requisitar n do usuário e substituir a imagem original I por uma imagem filtrada \bar{I} tal que, para todo (y, x) , temos que $\bar{I}(y, x)$ é a *média* dos valores dos pixels em $\mathcal{N}(y, x)$. O procedimento de aproximação é livre.

2.10 Módulo 8 (15/04): Filtro da Mediana

O processador deve apresentar um **botão de filtro da mediana**. A implementação dessa funcionalidade deve requisitar n do usuário e substituir a imagem original I por uma imagem filtrada \bar{I} tal que, para todo (y, x) , temos que $\bar{I}(y, x)$ é a *mediana* dos valores dos pixels em $\mathcal{N}(y, x)$.

2.11 Módulo 9 (22/04): Operador Gaussiano

Seja n um inteiro positivo ímpar maior ou igual a 3 e seja σ um número positivo. Um **filtro gaussiano** de tamanho n e desvio padrão σ é uma função $G: \{0, \dots, n-1\} \times \{0, \dots, n-1\} \rightarrow \mathbb{R}$ tal que

$$G(r+a, r+b) = \frac{1}{2\pi\sigma^2} \cdot e^{-\frac{a^2+b^2}{2\sigma^2}}.$$

A **convolução** de I e G é uma função $I * G$ tal que

$$(I * G)(y, x) = \sum_{-r \leq a, b \leq r} I(y+a, x+b) \cdot G(r+a, r+b).$$

O processador deve apresentar um **botão de suavização**. A implementação dessa funcionalidade deve requisitar n e σ do usuário e substituir a imagem original I por $I * G$. O procedimento de aproximação é livre.

2.12 Módulo 10 (29/04): Operador Laplaciano

Seja n um inteiro positivo ímpar maior ou igual a 3 e seja σ um número positivo. Um **filtro laplaciano** de tamanho n e desvio padrão σ é uma função $L: \{0, \dots, n-1\} \times \{0, \dots, n-1\} \rightarrow \mathbb{R}$ tal que

$$L(r+a, r+b) = -\frac{1}{\pi\sigma^4} \left(1 - \frac{a^2 + b^2}{2\sigma^2}\right) \cdot e^{-\frac{a^2 + b^2}{2\sigma^2}}.$$

O processador deve apresentar um **botão de realce**. A implementação dessa funcionalidade deve requisitar n e σ do usuário e substituir a imagem original I por uma imagem I' tal que

$$I'(y, x) = \nu(I(y, x) - (I * L)(y, x)).$$

O procedimento de aproximação é livre.

2.13 Módulo 11 (06/05): Erosão Morfológica

Seja n um inteiro positivo ímpar maior ou igual a 3. Um **elemento estruturante** de tamanho n é uma função $K: \{0, \dots, n-1\} \times \{0, \dots, n-1\} \rightarrow \{0, \dots, 255\} \cup \{-\infty\}$.

O processador deve apresentar um **botão de erosão**. A implementação dessa funcionalidade deve requisitar K do usuário e substituir a imagem original I por uma imagem $I \ominus K$ tal que

$$(I \ominus K)(y, x) = \nu(\max\{i: K(r+a, r+b) + i \leq I(y+a, x+b) \text{ para todo } -r \leq a, b \leq r\}).$$

O procedimento para representar $-\infty$ é livre.

2.14 Módulo 12 (13/05): Dilatação Morfológica

Seja n um inteiro positivo ímpar maior ou igual a 3. O **reflexo** de um elemento estruturante K de tamanho n é uma função \check{K} tal que

$$\check{K}(y, x) = K(n-y-1, n-x-1).$$

O processador deve apresentar um **botão de dilatação**. A implementação dessa funcionalidade deve requisitar K do usuário e substituir a imagem original I por uma imagem $I \oplus K$ tal que

$$(I \oplus K)(y, x) = \nu(\min\{i: -\check{K}(r+a, r+b) + i \geq I(y+a, x+b) \text{ para todo } -r \leq a, b \leq r\}).$$

O procedimento para representar $-\infty$ é livre.

3 Mini-Projeto

O mini-projeto consiste em implementar e apresentar um estudo comparativo do desempenho de versões sequenciais e paralelas da *Transformada de Fourier*. O código-fonte e um conjunto de slides devem ser entregues pelo *Blackboard* até **23:50 de 2 de junho**. Cada grupo deve fazer uma apresentação de **25 minutos** na aula de **3 de junho**, obrigatoriamente baseada nessa versão entregue no dia anterior.

3.1 Transformada de Fourier

Dadas duas funções, $f, g: \{0, \dots, n-1\} \rightarrow \mathbb{C}$, dizemos que g é a *Transformada Discreta de Fourier* de f se:

$$g(y) = \sum_{x=0}^n f(x) \cdot e^{-\frac{i2\pi xy}{n}},$$

onde i é a constante imaginária. Também dizemos que f é a *Transformada Inversa de Fourier* de g se:

$$f(x) = \frac{1}{n} \cdot \sum_{y=0}^n g(y) \cdot e^{\frac{i2\pi xy}{n}}.$$

Cabe mencionar que a constante imaginária pode ser eliminada dos expoentes através da *fórmula de Euler*:

$$e^{iz} = \cos(z) + i \cdot \sin(z)$$

3.2 Espectro de Magnitude

Dadas duas funções, $T: \{0, \dots, h-1\} \times \{0, \dots, w-1\} \rightarrow \mathbb{C}$ e $E: \{0, \dots, h-1\} \times \{0, \dots, w-1\} \rightarrow \mathbb{R}$, dizemos que E é o *espectro de magnitude* de T se:

$$E(y, x) = \log \left(1 + \sqrt{a^2 + b^2} \right), \text{ onde } T(y, x) = a + ib.$$

Seja m o maior valor de $E(y, x)$. Dizemos que uma imagem V é a *visualização* do espectro E se:

$$V(y, x) = \frac{E(y, x)}{m} \cdot 255.$$

3.3 Implementações

Cada implementação deve realizar as seguintes tarefas:

1. recebe três caminhos de arquivo, ENTRADA, ESPECTRO e SAIDA;
2. carrega de ENTRADA uma imagem I no formato PGM Simples;
3. calcula M_1 tal que a i -ésima linha de M_1 é a Transformada Discreta de Fourier da i -ésima linha de I ;
4. calcula T tal que a i -ésima coluna de T é a Transformada Discreta de Fourier da i -ésima coluna de M_1 ;
5. salva em ESPECTRO a visualização do espectro de magnitude de T no formato PGM Simples;
6. calcula M_2 tal que a i -ésima linha de M_2 é a Transformada Inversa de Fourier da i -ésima linha de T ;
7. calcula J tal que a i -ésima coluna de J é a Transformada Inversa de Fourier da i -ésima coluna de M_2 ;
8. salva em SAIDA a função J no formato PGM Simples. Considere o coeficiente real normalizado, ou seja, se $J(y, x) = a + ib$, considere $\nu(a)$.

Duas versões dos passos 3, 4, 6 e 7 devem ser feitas: uma *sequencial* e outra *paralela*, baseada em *OpenMP*.

3.4 Apresentação

A apresentação deve conter no mínimo as seguintes informações:

- descrição das implementações;
- descrição dos testes;
- comparação de performance das duas versões;
- percentual de ganho com paralelismo.