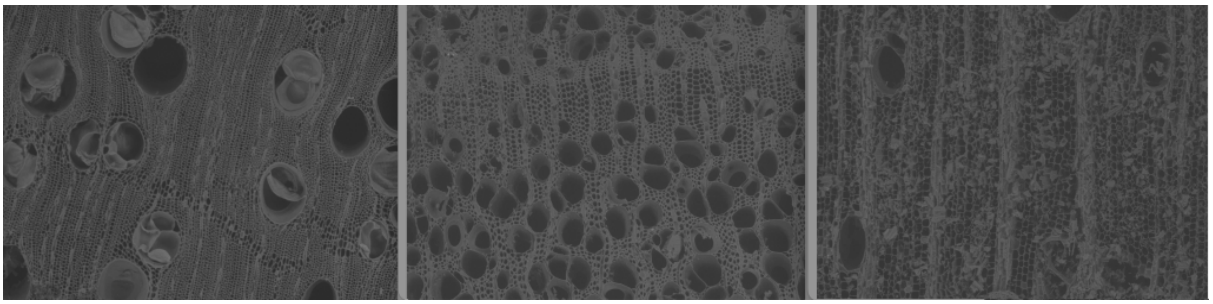
**Universidade Federal do Paraná**
**Departamento de Informática**
CI-1002 Programação II (2024/2)

**PRIMEIRO TRABALHO PRÁTICO (T1)**
Estratégias de Comparação de Imagens

Considere o seguinte problema: Você tem uma base de imagens rotuladas por especialistas e precisa desenvolver um sistema que receba uma nova imagem e encontre a mais similar nessa base de imagens.

---



Classe 1 Classe 2 Classe 3

**Figura 1**: Base de imagens rotulada por especialistas. Nesse exemplo temos três imagens de três classes diferentes.

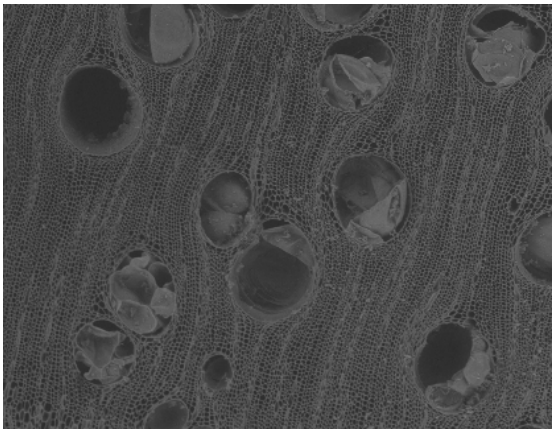


**Figura 2**: Imagem de entrada. Qual das três classes é mais similar?

---

Observando as principais características das imagens, pode-se observar que a Classe 1 é a mais similar. Mas como resolver isso computacionalmente?

Uma estratégia seria comparar as imagens pixel-a-pixel. Se você implementar essa estratégia, vai perceber logo que ela falha completamente, pois apesar das duas imagens pertencerem à mesma classe, elas são diferentes.

Dessa forma, precisamos extrair uma representação da imagem, ou seja, um vetor de características que sumarize o conteúdo da imagem. De posse desse vetor, a busca por uma imagem similar é então realizada através do cálculo de distância entre vetores.

***Local Binary Patterns***

LBP (*Local Binary Patterns*) ou Padrões Binários Locais, é um descritor muito utilizado em algoritmos de reconhecimento de imagens. O funcionamento deste filtro é simples: para cada pixel de uma imagem com 256 níveis de cinza, seleciona-se uma vizinhança de 8 pixels. O valor LBP é então calculado para o pixel central e armazenado na imagem destino, que possui as mesmas dimensões da imagem original.

Uma das formas de se calcular o valor LBP de um pixel, consiste em aplicar uma máscara com valores $2^n$, com $n = \{0, \ldots, 7\}$, ou seja: 1, 2, 4, 8, 16, 32, 64 e 128. O exemplo abaixo ilustra o cálculo de LBP para o pixel com valor 6:

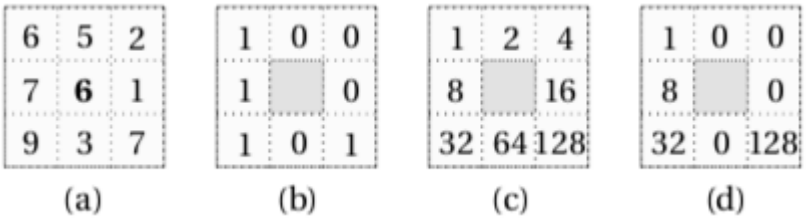



**Figura 3**: Cálculo do LPB considerando 8 vizinhos mais próximos do pixel (raio=1), o qual pode ser denotado por LBP (8,1).

Considere o pixel central com valor 6 e seus oito vizinhos na imagem (a). O valor do pixel central é utilizado como limiar e todos os pixels com valor de intensidade maior ou igual a ele recebem 1, caso contrário, 0 (b). Esses valores são então multiplicados pela máscara (c), resultando então nos valores apresentados em (d). O valor LBP do pixel 6 é dado pela soma desses valores, ou seja LPB = 1+8+32+128 = 169. Note que essa codificação garante valores LBP entre 0 e 255.

Quando todos os pixels da imagem forem processados teremos os seguintes resultados:
**a)** imagem LBP e

**b)** histograma que sumariza a ocorrência de cada um dos 256 $(2^8)$ valores LPB, como mostra a Figura 4.

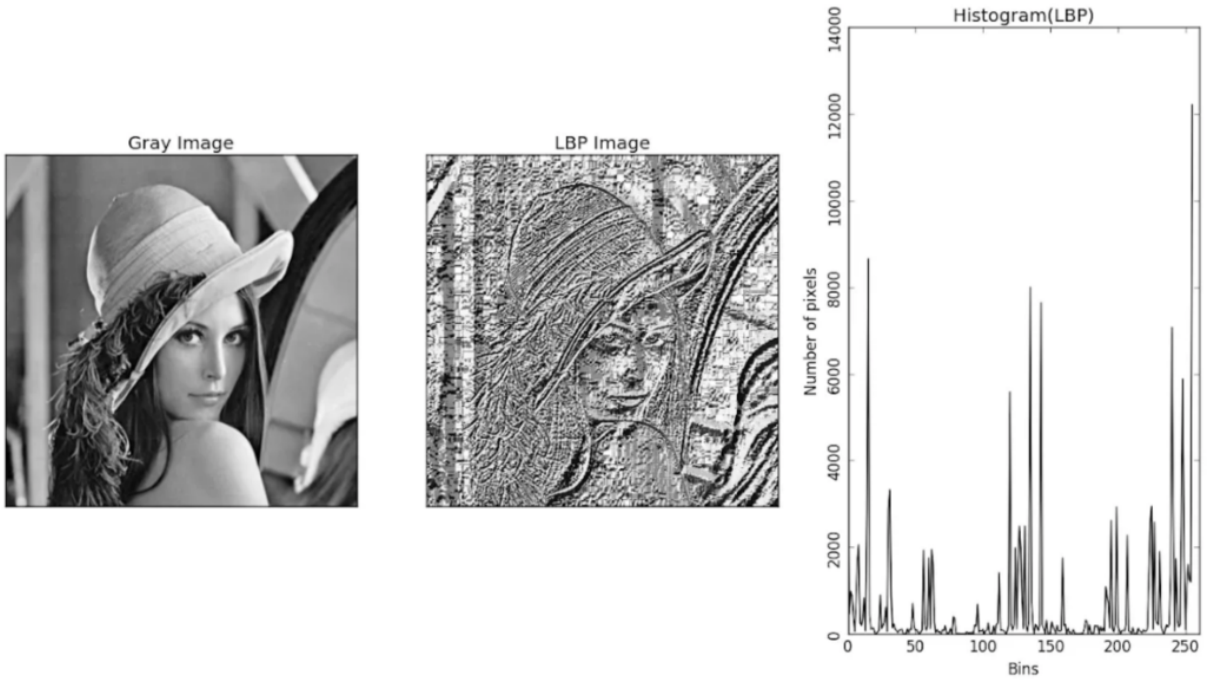



**Figura 4**: Imagem original, imagem LBP e histograma LBP.

No nosso caso, estamos interessados no histograma LPB, o qual nos permite comparar de forma eficiente duas imagens. Levando em consideração que um histograma pode ser representado por um vetor, uma forma de encontrar uma imagem similar é através do **cálculo de distância entre dois vetores**, sendo a **distância Euclidiana** uma das mais utilizadas.

**Formato *Portable Grey Map***

Para facilitar a leitura e escrita dos arquivos de imagem, neste trabalho será adotado o formato de imagem PGM (*Portable Grey Map*), um formato de imagem em níveis de cinza bem simples e fácil de ler/escrever. Boa parte dos programas de tratamento de imagens reconhece o formato PGM. A Figura 5 mostra um exemplo de uma imagem PGM. Note que existe o formato P2 (ASCII) e o formato P5 (binário). Seu programa de aceitar ambos.


(a) arquivo ASCII (b) imagem resultante

**Figura 5**: Exemplo imagem PGM.

**O que deve ser implementado:**

Você deve escrever um programa em C, chamado **lbp** que implemente as seguintes funcionalidades:

**1) Comparar uma imagem de teste com todas as imagens da base de referência**

Exemplo:
***./lbp -d ./base -i img1.tif***

A saída deve ser **EXATAMENTE** a seguinte

**Imagem mais similar: <img mais similar> <distância>**

em que <img mais similar> é a imagem com a menor distância Euclidiana encontrada no diretório ./base (**informar apenas o nome do arquivo com sua extensão**) e <distância> é o valor da distância associada a esta imagem, **um número com EXATAMENTE 6 casas decimais**. Nome e valor devem ser separados APENAS por um espaço.

Por exemplo, considerando a execução de "./lbp -d ./base -i img1.tif", uma possível saída seria **APENAS**:

**Imagem mais similar: img_comp1 0.005432**

O cálculo do LBP é a parte mais custosa do processo. Em função disso, você deverá realizar esse cálculo uma única vez para todas as imagens da base de referência. O vetor LBP deve ser armazenado em um arquivo binário que deve ter o mesmo nome da imagem, mas com extensão .lbp. Por exemplo, para a imagem img1.tif, o arquivo deverá ser img1.lbp.

Toda vez que o programa for executado, você deverá procurar pelo arquivo binário para calcular a distância com a imagem de teste. Caso ele não exista, você deve ler a imagem, calcular o LBP e gravar o vetor LBP no arquivo binário.

**2) Gerar a imagem LBP como mostrado na Figura 4**

Exemplo:

***./lbp -i img1.tif -o img_out.tif***

Nesse caso, o programa recebe como entrar imagem img1.tif e gerar a imagem LPB com o nome informado pela opção ***-o***. **NENHUMA** saída é esperada no terminal; caso algum erro ocorra, basta não criar a imagem de saída.

**Entrega no Moodle:**

Você deve entregar um **arquivo .zip** contendo:

1. Todo o código-fonte implementado;
2. Um Makefile (ao executar o comando “make”, deve ser gerado um **executável com o nome LBP**, em maiúsculas);
3. Um README que apresente, de forma geral, a estrutura do programa (por exemplo, explicando o conteúdo de cada arquivo .c).

A entrega deve ser feita pelo Moodle até o prazo máximo de **11/10/2024, às 23:59h**.