

Roteiro

Analise de imagens

Notas de linguagem Java

Número π : `double pi = Math.PI;`

Operador lógico AND: `if (a==b && b>c)`

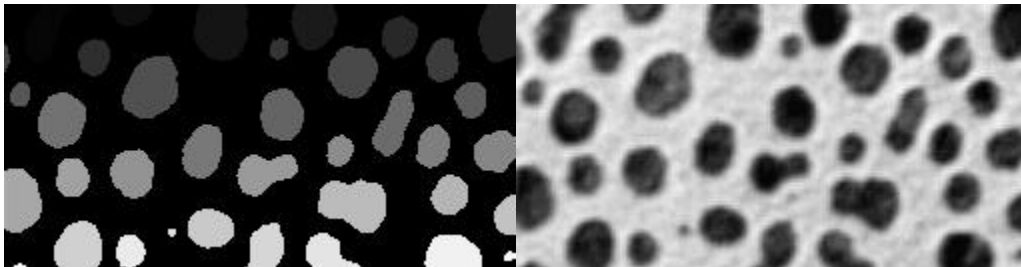
Maior valor positivo: `double max = Double.MAX_VALUE;`

Valor máximo de uma imagem: `int max = (int)im.getMaximum();`

Maior valor negativo: `double min = Double.MIN_VALUE;`

Valor mínimo de uma imagem: `int min = (int)im.getMinimum();`

1. Rotulação de objetos



Rotulação é uma função que identifica e rotula componentes conexos dentro de uma imagem. A entrada é uma imagem binária contendo objetos brancos (255) num fundo preto (0). A saída é uma imagem rotulada, onde cada objeto tem um valor de pixel correspondente ao seu rótulo (veja a figura acima) de 1 até n (n é o número de objetos) e 0 dentro do fundo.

Escreva um método `counting()` dando o número de objetos de uma imagem rotulada (use `IJ.write(""+n)` para mostrar o resultado).

Abra a imagem `blobs.tif`, ajuste o limiar ("threshold") para obter uma imagem binária (células em branco, fundo preto) e aplique um filtro de mediana (Process → Filters → Median) para suprimir o ruído. Então, rotule a imagem com o plugin **Labeling** e use **Counting** para mostrar o número de células na imagem. Preencha o relatório.

2. Cálculo de característica de objetos

2.1 Área

O método `area()` computa uma estimativa da área do objeto contando o número de pixels dentro de um objeto. Este método está disponível. Leia e entenda o código deste método em `Code.java`.

Aplique este método em `test.tif` usando o plugin **Area** preencha o relatório.

2.2 Centro de gravidade

Escreva os métodos `xGravityCenter()` E `yGravityCenter()` os quais retornam o centro de gravidade de todos os objetos em uma imagem rotulada.

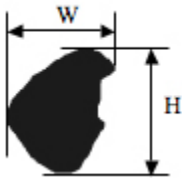
Aplique este método em `test.tif` usando o plugin **Gravity Center** e preencha o relatório.

2.3 Perímetro

Escreva o método `perimeter()` o qual computa uma estimativa do perímetro discreto de cada objeto em uma imagem rotulada. Para cada objeto, rastreia a imagem para identificar pixels pertencentes ao objeto. Então, soma o número de pixels tendo pelo menos um vizinho (componente 4-conexo) pertencente ao fundo. Assumimos não haver objetos na borda.

Aplique esta método a `test.tif` usando o plugin **Perimeter** e preencha o relatório.

2.4 Circularidade



Uma medida comum de compactação dos objetos, chamada de razão de circularidade, é a razão da área do objeto pela área do círculo (o objeto mais compacto possível) tendo o mesmo perímetro. Esta razão é expressa matematicamente como $C = 4 \cdot \pi \cdot (\text{Área}) / (\text{Perímetro})^2$. Para um círculo, esta razão é 1; para um objeto infinitesimalmente fino, ela tende a 0. Note que devido a erro na estimativa do perímetro, a razão de circularidade deve ser maior que 1.

Escreva um método `circularity()` que computa a circularidade para cada objeto em uma imagem rotulada.

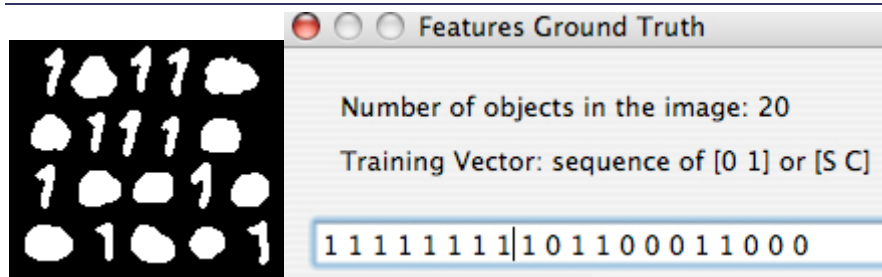
Aplique este método a `test.tif` usando o plugin **Circularity** e preencha o relatório.

2.5 Razão Largura/Altura

Escreva um método `wh()` que computa a razão largura/altura de cada objeto numa imagem rotulada.

Aplique este método a `test.tif` usando o plugin **WH** e preencha o relatório.

3. Classification based on shape features



a. Comparação de características

A proposta aqui é determinar a melhor característica para discriminar os 0s dos 1s. O plugin **Features** mostra o gráfico com o valor das características para cada objeto na imagem. A caixa de dialogo possibilita definir a seqüência previamente. Por exemplo, se tiver três objetos, um 0 e dois 1 na imagem, entre "0 1 1" como vetor de treinamento na caixa de dialogo (espaço como separador); a ordem da seqüência deve corresponder a ordem de rotulação a qual pode ser diferente de posição na imagem. O gráfico mostra os valores de características computados para 0s (vermelho) e 1s (azul). Nota: os valores estão normalizados para o valor máximo.

- Abra a imagem `numbers1.tif`.
- Determine o procedimento para obter a imagem binária baseado em comandos do ImageJ, com zeros preenchidos, como mostrado na figura. Explique seu procedimento no relatório.
- Rotule esta imagem usando **Labeling**
- Rode **Features** com o vetor de treinamento correto

Repita estes passos para `numbers2.tif` e `numbers3.tif`.

Insira os três gráficos no relatório e comente. Qual é a melhor característica para discriminar os 0s de 1s? Explique a escolha.

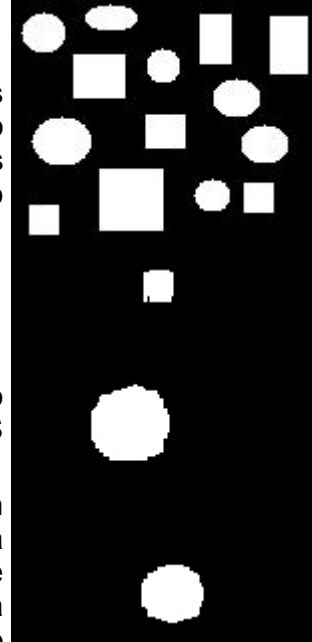
b. Classificação de objetos

A classificação é um procedimento no qual objetos individuais são colocados em diferentes classes baseado em informações quantitativas de uma ou mais características inerentes nos objetos e baseado no treinamento com itens previamente rotulados.

Treinamento

No que segue usaremos a imagem `squares-circles.tif` como um conjunto de treinamento de duas classes: quadrados S (vermelho) e círculos C (azul).

Rotule a imagem `squares-circles.tif` e rode o plugin **Features**. Comente a distribuição de amostras para a circularidade. Rode o **Circularity** para obter a circularidade para cada objeto, compute a média e desvio padrão para cada classe e preencha na tabela correspondente no relatório.



Reconhecimento

Suponha que temos três objetos na imagem `unknowns.tif`, que queremos classificar como quadrados ou círculos.

Método 1: Um modo simples de classificar os objetos desconhecidos é computar a distância mínima entre o valor do objeto e a média de cada classe ($|x - \mu_i|$). Classifique os três objetos desconhecidos no relatório.

Método 2: um modo mais eficiente de classificar os objetos é computar a distância Mahalanobis mínima, dada no nosso caso por $(|x - \mu_i|)/\sigma_i$, com σ_i o desvio padrão da classe. Classifique os três objetos desconhecidos no relatório.

4. Desafio

a. Propagação de um pixel semente

Escreva um método `prop()` que retorna uma imagem binária contendo o objeto localizado em (x,y) .

Dica: Pense recursivamente!

Defina o pixel (x,y) da saída como 1. Então, para cada um dos quatro vizinhos, se o pixel correspondente é não-zero, na imagem de entrada e zero na imagem de saída, rode o método `prop()` de um pixel específico.

Aviso: não se esqueça de testar os limites da imagem.

b. Implementação da rotulação (labeling)

O método `labeling()` rastreia os pixels da imagem de entrada, e cada vez que o valor é não zero, roda `prop()` para identificar o objeto. Então, este objeto é adicionado a imagem de saída com um rótulo único, e é suprimido da imagem de entrada.