

# Roteiro

## Operadores Morfológicos

### 1. Erosão e dilatação com elementos estruturantes 3\*3

#### 1.1 Entendendo o operador dilatação

Entenda o código do operador de dilatação que está disponível no método `dilate3()` do arquivo `Code.java`. O nome do plugin é `Dilate3`.

#### 1.2 Codifique o operador Erosion

Escreva um método `erode3()` que implementa o operador erosão no arquivo `Code.java`. O nome do plugin é **Erode3**.

Teste o seu código nas imagens `test.tif`, `lena.tif`, `keys.tif`.

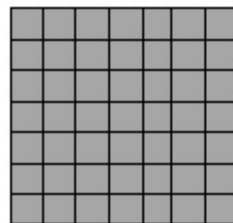
	Notas de programação	
Duplica uma imagem	<code>Image2 = image1.clone();</code>	<code>image1 → image2</code>
Adição de imagens	<code>out.add(im1, im2);</code>	<code>im1 + im2 → out</code>
Subtração de imagens	<code>out.subtract(im1, im2);</code>	<code>im1 - im2 → out</code>
Define o maior valor double	<code>Double max = Double.MAX_VALUE;</code>	
Define o menor valor Double	<code>Double min = -Double.MAX_VALUE;</code>	

### 2. Operadores morfológicos com elementos estruturantes definidos pelo usuário

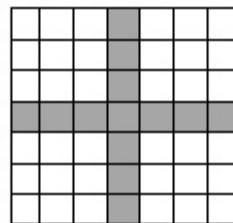
Propomos aqui, implementar os operadores morfológicos para três elementos estruturantes: SQUARE, CROSS e DISC. O DISC é uma aproximação do disco numa grade quadrada.

#### 2.1 Criação de elementos estruturantes

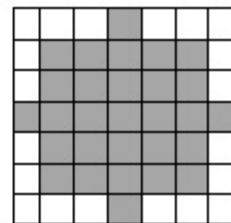
Um elemento estruturante é um arranjo booleano 2D, contendo `true` nas células cinzas e `false` nas células brancas. O método `boolean[][] cross(int n)` constrói um elemento estruturante CROSS de tamanho `n`. Codifique o método `boolean[][] square(int n)` e `boolean[][] disc(int size)` para construir o SQUARE e o DISC, respectivamente. O tamanho do elemento estruturante é sempre ímpar.



Square



Cross



Disc

#### 2.2 Implementação dos operadores morfológicos

A tarefa é implementar os seguintes operadores. O usuário pode escolher a forma e o tamanho (ímpar) do elemento estruturante através de uma caixa de diálogo.

Operador	Metodo no Code.java	Nome do plugin	Plugin solução do professor
Dilatação	dilate()	Dilate	
Erosão	erode()	Erode	Teacher Erode
Abertura	open()	Open	Teacher Open
Fechamento	close()	Close	Teacher Close
Gradiente	gradient()	Gradient	Teacher Gradient
Top Hat Dark	topHatDark()	Top Hat Dark	
Top Hat Bright	topHatBright()	Top Hat Bright	

Teste o seu código nas imagens test.tif, lena.tif, and keys.tif.

### 2.3 Testando

Teste o seu código em cat\_impulse.tif. insira os resultados das seguintes operações em relatorio.doc:

- Dilatação SQUARE, tamanho 5
- Dilatação, CROSS, tamanho 5
- Dilatação, DISC, tamanho 5
- Top Hat Bright, SQUARE, tamanho 3
- Top Hat Dark, SQUARE, tamanho 5

### 3. Aplicação: Contando objetos (2)

Para resolver este problema, você pode usar operadores morfológicos, e commands básicos do ImageJ: "Threshold", "Inverse", "Arithmetic operations" e "Analyse Particles" para contar o número de objetos na imagem binária. Selecione "Show Outlines" e verifique "Display Results" para visualizar os resultados do comando "Analyse Particles".

#### 3.1 Contando objetos redondos na image1

Proponha um algoritmo para contar objetos redondos em image1.tif. Salve os resultados e descreva o algoritmo em "relatorio.doc".

#### 3.2 Contando todas as células em image2

Proponha um algoritmo para contar as células na image2.tif. Salve os resultados e descreva o algoritmo em "relatorio.doc".

#### 3.3 Contando células cheias (filled) na image2

Proponha um algoritmo para contar as células cheias na image2.tif. Salve os resultados e descreva o algoritmo em "relatorio.doc".

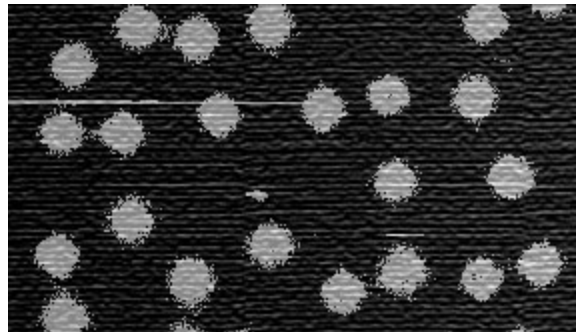


image1.tif

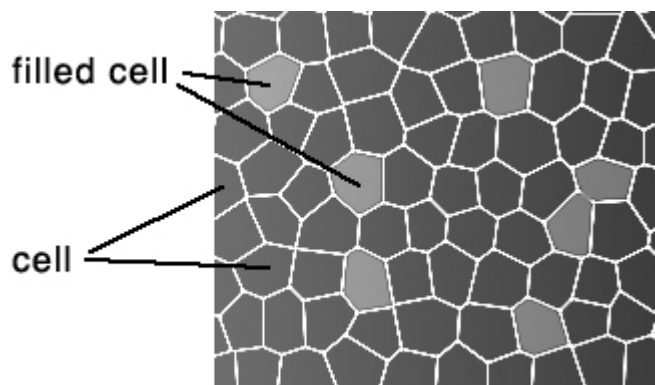


image2.tif