UNIVERSIDADE FEDERAL DE SÃO CARLOS - UFSCAR
CIÊNCIA DA COMPUTAÇÃO
PROCESSAMENTO DIGITAL DE IMAGENS

RELATÓRIO: MORFOLOGIA EM NÍVEL DE CINZA, COM APLICAÇÕES

São Carlos, Outubro de 2020

Morfologia em nível de cinza

Implementar dilatação, erosão, abertura, fechamento e transformada top-hatem nível de cinza.

Discentes:

Agustin Gabriel Amaral Castillo, 592862

Jorge Vinicius Gonçalves, 758594

Thiago de Moraes Teixeira, 760667

Victoria de Martini de Souza, 75937

Docente:
Cesar Henrique Comin, Prof.
Universidade Federal de São Carlos – UFSCar

Dilatação

- **Explicação do método implementado**

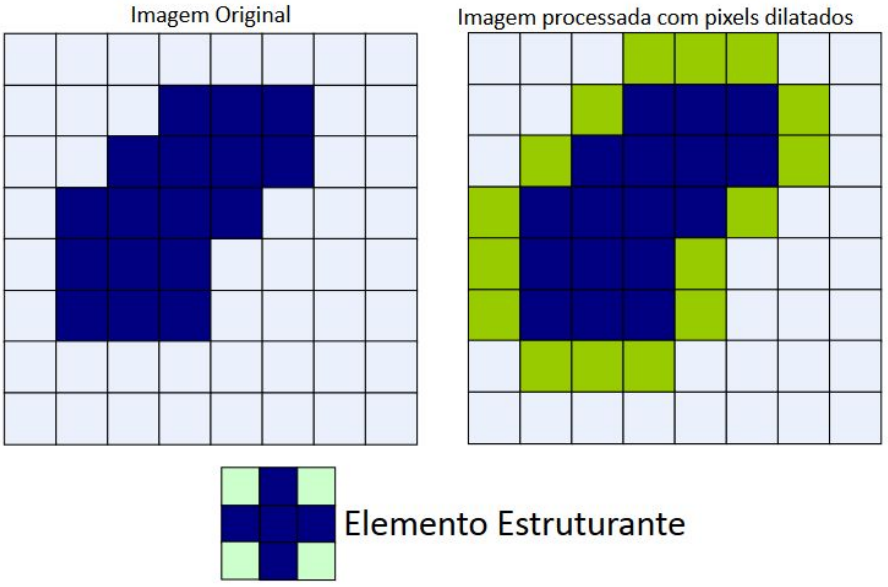
  Sendo A e B conjuntos de Z^2 e ∅o conjunto vazio, define-se a dilatação de A por B, denotada por A⊕B, como:

$$A \oplus B = \{z \mid (\hat{B})_x \cap A \neq \varnothing\}$$

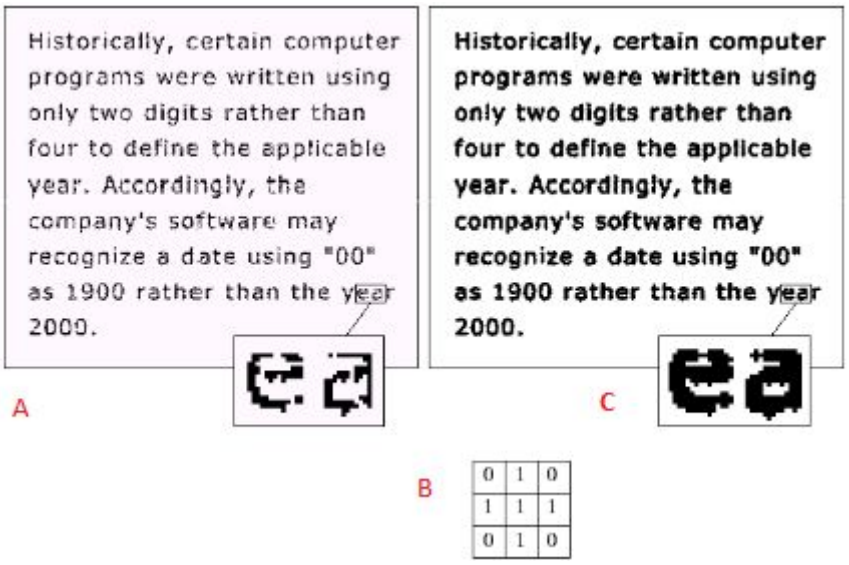
  A dilatação de A por B é então o conjunto de todos os deslocamentos x tais que A sobreponham-se em pelo menos um elemento não nulo.

  - O conjunto B é normalmente chamado de elemento estruturante da dilatação

  Note que a dilatação de A por B é determinada da seguinte forma: Primeiro, B é rotacionado em torno da origem, para obter B'. Depois, A' é erodido por B' e, finalmente, a dilatação é obtida determinando o complementar do resultado da erosão. Intuitivamente, a dilatação incha a imagem utilizando o elemento estruturante.



- **Motivação do uso do método**
    - Aplicação:
        - Preenchimento de espaço



A) Exemplo de texto de baixa resolução com caracteres quebrados (visualização ampliada)

B) Elemento estruturante

C) Dilatação de a por b, os segmentos quebrados foram unidos

Visualmente, a dilatação carimba o elemento estruturante na imagem, já a erosão diminui a estrutura da imagem.

- **Explicação da parte mais importante do código**

Implementação da fórmula de dilatação a partir do conceito de filtro de máximo com a seguinte fórmula:

$$\left[f \oplus b\right](x, y) = \max_{(s,t)\in b} \left\{f(x - s, y - t)\right\}$$

A fórmula pode ser escrita como um filtro de máximo entre os pixels somente porque foi utilizado o elemento estruturante uniforme “flat”, onde todos os elementos são 1.

```
# Elementro estruturante uniforme
elemento_estruturante = np.array([[1, 1, 1],
                                  [1, 1, 1],
                                  [1, 1, 1]])
```

Varre a imagem com borda (preenchida com zeros anteriormente) e aplica o maior valor da vizinhança ao pixel central do elemento estrutural na imagem_res que será a saída. Valor de máximo inicia com -1 para qualquer pixel ser maior do que ele, assim comparando com todos pois o valor mínimo de um pixel é 0.

```
for row in range(1, num_rows+1):
    for col in range(1, num_cols+1):
        maximum = -1
        for s in range(num_rows_ee):
            for t in range(num_cols_ee):
                current = img_padded[row-1+s, col-1+t] * elem_est[s, t]
                if current > maximum:
                    maximum = current
        img_res[row-1, col-1] = maximum

return img_res
```

Erosão

- **Explicação do método implementado**

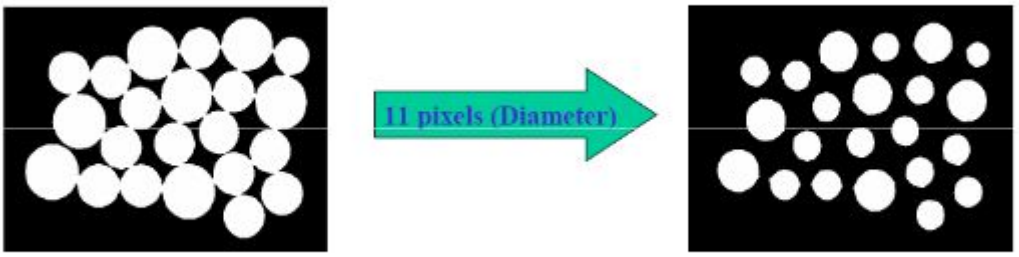
  Sendo A e B em Z^2, a erosão de A por B, denotada por $A \ominus B$, é denotada por:

$$A \ominus B = \{x \mid (B)_x \subseteq A\}$$

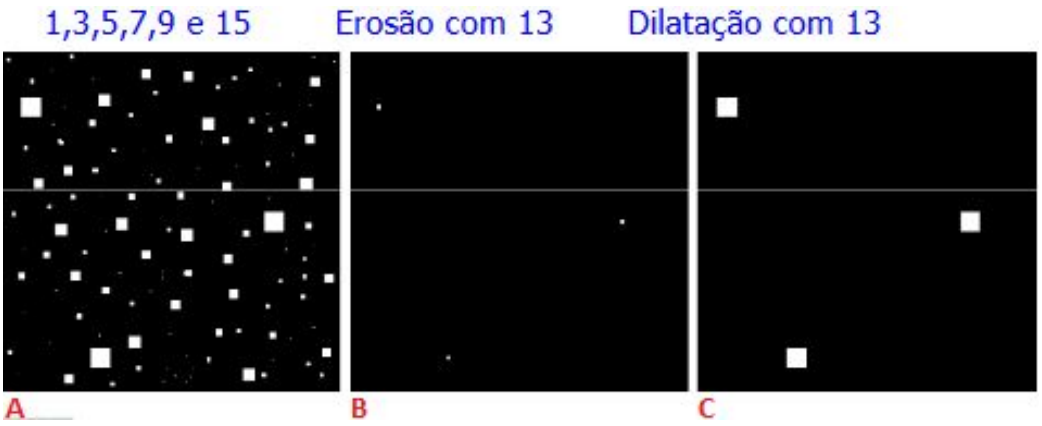
A erosão de A por B é o conjunto de todos os pontos x tais que B,quando transladado pro x fique contido em A.

Em palavras, A º B consiste de todos os pontos em que B transladado x cabe em A. Acerca da erosão, A é a imagem de entrada e o B o elemento estruturante.

A erosão é escrita usando uma notação de operador da seguinte forma: εB(A) :=AB. Se a origem do elemento estruturante assume valor 1, então a erosão diminui o tamanho da imagem.



- **Motivação do uso do método**
  - Aplicação:
    - Remoção de componentes



A) Imagem de quadrados de tamanho 1,3,5,7 e 9 e 15 pixels na lateral.

B)Erosão de a com elementos estruturantes quadrados de 1's (escala) e 13 pixels na lateral

C)Dilatação de b com o mesmo elemento estruturante

- **Explicação da parte mais importante do código**

Implementação da fórmula de erosão a partir do conceito de filtro de mínimo com a seguinte fórmula:

$$[f \ominus b](x, y) = \min_{(s,t) \in b} \{f(x + s, y + t)\}$$

A fórmula pode ser escrita como um filtro de mínimo entre os pixels somente porque foi utilizado o elemento estruturante uniforme “flat”, onde todos os elementos são 1.

```
# Elementro estruturante uniforme
elemento_estruturante = np.array([[1, 1, 1],
                                  [1, 1, 1],
                                  [1, 1, 1]])
```

Varre a imagem com borda (preenchida com zeros anteriormente) e aplica o menor valor da vizinhança ao pixel central do elemento estrutural na imagem_res que será a saída. Valor de minimum inicia com 300 para qualquer pixel ser menor do que ele quando comparado com todos, logo que o valor máximo de um pixel é 255 e utilizamos a verificação se o novo mínimo é maior do que zero (considerando zeros como os elementos de borda) para não interferirem nos pixels das extremidades da imagem.

```
for row in range(1, num_rows+1):
    for col in range(1, num_cols+1):
        minimum = 300
        for s in range(num_rows_ee):
            for t in range(num_cols_ee):
                current = img_padded[row-1+s, col-1+t] * elem_est[s, t]
                if current < minimum and current>0:
                    minimum = current
        img_res[row-1, col-1] = minimum

return img_res
```

Dessa forma, podemos concluir que a dilatação expande, enquanto a erosão reduz uma imagem.

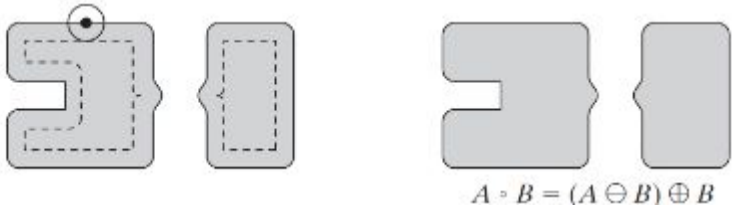
Abertura

- **Explicação do método implementado**

A abertura da imagem A por um elemento estruturante B, denotado A◦B, é definida como a erosão de A por B seguida da dilatação por B. Precisamente, tem-se

$$A \circ B = (A \ominus B) \oplus B$$

Portanto, a abertura é a união de todas as translações do elemento estruturante que cabem na imagem. Intuitivamente, a abertura carimba o elemento estruturante em cada ponto onde esse cabe na imagem. A abertura pode ser aplicada como filtro para suavizar bordas ou cantos em imagens.



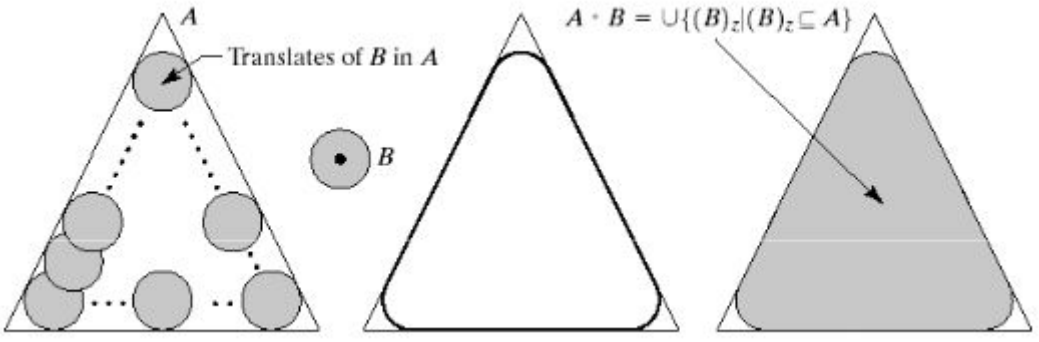
Propriedades da operação abertura:

– A o B é um subconjunto (sub-imagem) de A.

– Se C for um subconjunto de D, então C o D será um subconjunto de D o B.

– (A o B) o B = A o B.

- **Motivação do uso do método**
  - Suaviza o contorno da imagem, quebra partes estreitos, e elimina partes finas.



Elemento estruturante B rolando ao longo do limite interno de A (o ponto indica a origem de B). A linha grossa é o limite externo da abertura e o sombreado é a abertura completa.

- **Explicação da parte mais importante do código**

  Aplica-se erosão sobre a img_res e na sequência aplica-se a dilatação, funções já feitas anteriormente.

```
# Função de abertura
def abertura(img, elem_est):

    # Imagem resultante
    img_res = np.zeros_like(img)

    # Abertura - aplica-se a erosão e a dilatação logo após
    img_res = erosion(img, elem_est)
    img_res = dilation(img_res, elem_est)

    return img_res
```

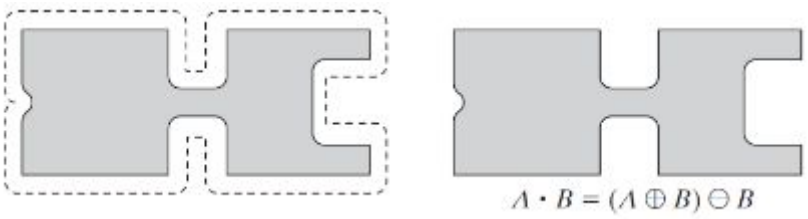
Fechamento

- **Explicação do método implementado**

  O fechamento é definido por uma dilatação seguido da erosão pelo mesmo elemento estruturante. O fechamento de A por B, denotado por A•B, é definido como segue:

$$A \bullet B = (A \oplus B) \ominus B$$

Visualmente, ao contrário da abertura que suaviza as bicos das imagens, o fechamento de uma imagem suaviza bicos dentro do plano de fundo.



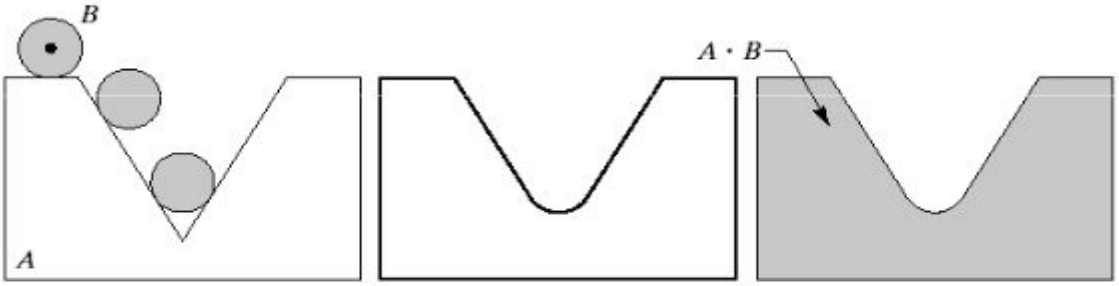
Propriedades da operação fechamento:

– A é um subconjunto (sub-imagem) de A • B.

– Se C for um subconjunto de D, então C • B será um subconjunto de D • B.

– (A • B) • B = A • B.

- **Motivação do uso do método**
    - Suaviza o contorno da imagem, elimina pequenos buracos e preenche fendas em um contorno.



Elemento de estruturação B rolando no limite externo do conjunto A. A linha grossa é o limite externo do fechamento. Por último o fechamento completo sombreado.

- **Explicação da parte mais importante do código**

Aplica-se a dilatação sobre a img_res e na sequência aplica-se a erosão, funções já feitas anteriormente.

```python
# Função de fechamento
def fechamento(img, elem_est):

    # Imagem resultante
    img_res = np.zeros_like(img)

    # Fechamento - aplica-se a dilatação e a erosão logo após
    img_res = dilation(img, elem_est)
    img_res = erosion(img_res, elem_est)

    return img_res
```

Transformada top-hat

- **Explicação do método implementado**

  A diferença entre uma imagem e o resultado de sua abertura é chamada de transformada top-hat, dada por:

$$h = f - (f \circ b)$$

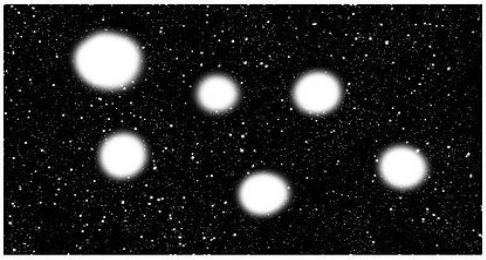
  Top-hat é responsável extrair pequenos elementos e detalhes de imagens fornecidas, é definida como a diferença entre a imagem de entrada e sua abertura por algum elemento estruturante.

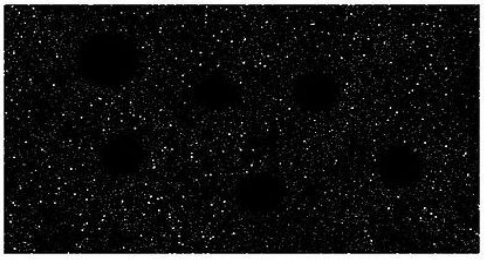
- **Motivação do uso do método**:
  - A transformação top-hat é usada para extração de recurso, equalização de fundo, aprimoramento de imagem e outros.

  Suponha que estamos interessados apenas em pequenas manchas na imagem e queremos remover os objetos brilhantes maiores. Nesse caso, a transformação top-hat pode remover objetos brilhantes maiores e reter pequenas manchas selecionando o tamanho do elemento de estruturação que está entre os objetos removidos e os objetos de interesse.

  O raio dos seis maiores objetos brilhantes é de aproximadamente 50 a 100 pixels, enquanto o raio dos objetos de interesse é de cerca de 2 a 4 pixels.

Além disso, os objetos de interesse são formas circulares, por isso escolhemos um elemento estruturante em forma de disco com raio 5.





- **Explicação da parte mais importante do código**

  Aplica-se a abertura na img_res e na sequência subtrai a imagem que foi aplicado a abertura da imagem original, função já feita anteriormente(abertura).

```python
# Função da transformada top-hat
def top_hat(img, elem_est):

    img_opened = np.zeros_like(img)

    # Transformada Top-Hat - aplica-se a abertura e logo após subtrai a imagem original da 'imagem aberta'
    img_opened = abertura(img, elem_est)
    img_top_hat = img - img_opened

    return img_top_hat
```

Referências:

https://www.cin.ufpe.br/~tir/ComputacaoCientifica/7.Morfologia%20Matematica.pdf

http://www.ime.unicamp.br/~valle/PDFfiles/valente10.pdf

https://www.ic.unicamp.br/~ffaria/pi2s2015/class13/aula_morfologia2.pdf