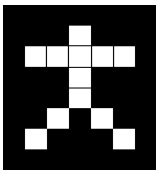
Lista #3

Processamento de Imagens

1 - Seja A o nome da imagem abaixo. Qual o resultado das operações a seguir?



a) Hit-miss de A por $B = \begin{bmatrix} X & 0 & 0 \\ 1 & 1 & 0 \\ X & 0 & 0 \end{bmatrix}$

b) Hit-miss de A por $B_{45} = \begin{bmatrix} 1 & X & 0 \\ X & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

c) Hit-miss de A por B_{90}

d) Hit-miss de A por B_{135}

e) Hit-miss de A por B_{180}

f) Hit-miss de A por B_{225}

g) Hit-miss de A por B_{270}

h) Hit-miss de A por B_{315}

i) C igual à união de todos os resultados acima

j) $A - C$

2 - Considere o elemento estruturante B abaixo. Dê exemplos de regiões conexas cujo resultado da operação de hit-miss por B seja não vazio.

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & X & X & X & X & X & 0 \\ 0 & X & 1 & 1 & 1 & X & 0 \\ 0 & X & 1 & 1 & 1 & X & 0 \\ 0 & X & 1 & 1 & 1 & X & 0 \\ 0 & X & X & X & X & X & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

3 - Seja `img` uma imagem do tipo `ndarray`, com pixels de tipo `float32` no intervalo [0.0, 1.0]. Escreva uma linha de código em Python que converta seus pixels para o tipo `uint8` no intervalo [0, 255].

4 - Seja `img` uma imagem do tipo `ndarray`, com pixels de tipo `uint8`. Escreva uma linha de código em Python que encontre sua negativa.

5 - Considere o código em Python abaixo. Insira código no lugar das reticências para fazer a saturação em 0 e 255, e converter para `uint8`.

```
def contrast(im, r, m):
    result = r * (im - m) + m

    ...

    return result
```

6 - Sejam `x` e `y` as coordenadas horizontal e vertical dos pixels de uma imagem, e `s` e `t` as coordenadas horizontal e vertical dos valores no interior de uma janela. Seja `xmask = x+s` e `ymask = y+t`. Sejam `w` e `h` a largura e altura da imagem. Escreva código que faça a operação de *clamp*, que evita o uso de pixels fora dos limites da imagem de entrada nas operações de máscara.