

## Componentes Conexos em Imagens

A rotulação (ou extração) de componentes conexos é de extrema importância em diversas aplicações de visão computacional e reconhecimento de padrões, tais como reconhecimento de texto em documentos e classificação de imagens. Basicamente, consiste no procedimento de atribuição de um único rótulo a cada objeto (ou componente conexo) da imagem. A partir dessa divisão, é possível extrair diversas informações de interesse, tais como estrutura espacial, tamanho médio ou tamanho máximo do componente conexo.

Dois pixels  $p$  e  $q$  estão conectados de 4 se obedecem ao critério de similaridade e se são vizinhos horizontais ou verticais. Da mesma forma, são ditos conectados de 8 se obedecem ao critério de similaridade e são vizinhos horizontais, verticais ou diagonais (veja conceito de vizinhança no projeto computacional - imagens). Um componente conexo de uma imagem é dado por um conjunto de pixels que estão conectados entre si, ou seja, dado um pixel  $p$  do componente conexo, existe pelo menos um caminho de  $p$  a todos os outros pixels deste componente.

São diversos os algoritmos para executar esta tarefa, sendo classificados em *multi-pass*, *two-pass* e *one-pass*. De forma geral, tais algoritmos atribuem rótulos provisórios para cada pixel. Se a varredura for feita da esquerda para a direita e de cima para baixo, ela é denominada *forward scan*. Se for feita da direita para a esquerda e de baixo para cima, é denominada *backward scan*.

### Algoritmo convencional - *multi-pass*

A matriz que representa a imagem de entrada será representada por  $I$ . Como neste trabalho a rotulação será feita em imagens binárias, assuma que o pixel na coordenada  $(i, j)$  pertence ao fundo (*background*) se  $I(i, j) = 0$  e ao objeto (*foreground*) se  $I(i, j) = 1$ . A imagem rotulada será armazenada na matriz  $L$  (que possui a mesma dimensão que  $I$ ).

Seja  $r$  um valor inteiro inicializado em 1. A primeira etapa deste algoritmo consiste na atribuição dos rótulos provisórios da seguinte forma:

$$L(i, j) = \begin{cases} 0, & \text{se } I(i, j) = 0; \\ r, (r \leftarrow r + 1), & I(k, l) = 0, \text{ para } (k, l) \in V; \\ \min(L(k, l)), & \text{para } (k, l) \in V, \text{ caso contrário} \end{cases} \quad (1)$$

Segundo esta regra, um pixel pertencente ao fundo não tem seu valor alterado (primeira consideração). Caso todos os vizinhos de uma posição  $(i, j)$  pertençam ao fundo, é atribuído o rótulo  $r$  a esta posição na imagem resultado  $L$  e o valor de  $r$  é incrementado (segunda consideração). Caso um ou mais vizinhos de  $I(i, j)$  já tenham um rótulo atribuído, o menor destes rótulos é atribuído à  $L(i, j)$  (terceira consideração).

A vizinhança depende do tipo de varredura sendo realizada. As Tabelas 1 e 2 ilustram a vizinhança considerada para *forward scan* e *backward scan*, respectivamente. Os vizinhos são denotados pelas letras  $a, b, c$  e  $d$ .

a	b	c
d	(i,j)	

Tabela 1: Máscara *forward scan*

	(i,j)	d
c	b	a

Tabela 2: Máscara *backward scan*

Após essa primeira etapa, todos os pixels pertencentes a um objeto terão um rótulo temporário atribuído. O problema é que pixels pertencentes a um mesmo objeto conexo poderão ter rótulos diferentes. Suponha que a imagem de entrada,  $I$ , é representada pela matriz da Tabela 3. Após uma aplicação do procedimento acima considerando a varredura *forward scan*, a rotulação resultante seria aquela ilustrada na Tabela 4, a qual resultou na atribuição dos rótulos 2 e 3 para um mesmo objeto conexo.

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	<b>1</b>	<b>1</b>	0	0	0	0	0	0
0	0	<b>1</b>	<b>1</b>	0	0	0	0	0	0
0	0	0	0	0	0	0	<b>1</b>	0	0
0	0	0	0	<b>1</b>	<b>1</b>	<b>1</b>	<b>1</b>	0	0
0	0	0	0	0	0	<b>1</b>	<b>1</b>	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Tabela 3: Resultado - *forward scan*

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	<b>1</b>	<b>1</b>	0	0	0	0	0	0
0	0	<b>1</b>	<b>1</b>	0	0	0	0	0	0
0	0	0	0	0	0	0	<b>2</b>	0	0
0	0	0	0	<b>3</b>	<b>3</b>	<b>2</b>	<b>2</b>	0	0
0	0	0	0	0	0	<b>2</b>	<b>2</b>	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Tabela 4: Resultado - *backward scan*

Para resolver este problema, o algoritmo *multi-pass* mais simples irá aplicar repetidamente as varreduras *forward* e *backward scan* segundo o procedimento acima até que nenhuma alteração seja feita na imagem resultante  $L$  (observe que, como todos os pixels pertencentes a um objeto já receberam um rótulo inicial, apenas a terceira consideração será de fato utilizada).