

**Algorithmus 11.3**

Kombinierte Konturfindung und Regionenmarkierung. Die Prozedur COMBINEDCONTOURLABELING erzeugt aus dem Binärbild  $I$  eine Menge von Konturen sowie ein Array mit der Regionenmarkierung aller Bildpunkte. Wird ein neuer Konturpunkt (äußere oder innere Kontur) gefunden, dann wird die eigentliche Kontur als Folge von Konturpunkten durch den Aufruf von TRACECONTOUR (Zeile 20 bzw. Zeile 27) ermittelt. TRACECONTOUR selbst ist in Alg. 11.4 beschrieben.

```

1: COMBINEDCONTOURLABELING ( $I$ )
    $I$ : binary image
2:   Create an empty set of contours:  $\mathcal{C} \leftarrow \{\}$ 
3:   Create a label map  $LM$  of the same size as  $I$  and initialize:
4:   for all  $(u, v)$  do
5:      $LM(u, v) \leftarrow 0$                                      ▷ label map  $LM$ 
6:    $R \leftarrow 0$                                            ▷ region counter  $R$ 

7:   Scan the image from left to right, top to bottom:
8:   for  $v \leftarrow 0 \dots N-1$  do
9:      $L_c \leftarrow 0$                                        ▷ current label  $L_c$ 
10:    for  $u \leftarrow 0 \dots M-1$  do
11:      if  $I(u, v)$  is a foreground pixel then
12:        if  $(L_c \neq 0)$  then                                ▷ continue existing region
13:           $LM(u, v) \leftarrow L_c$ 
14:        else
15:           $L_c \leftarrow LM(u, v)$ 
16:          if  $(L_c = 0)$  then                                ▷ hit new outer contour
17:             $R \leftarrow R + 1$ 
18:             $L_c \leftarrow R$ 
19:             $\mathbf{x}_S \leftarrow (u, v)$ 
20:             $C_{\text{outer}} \leftarrow \text{TRACECONTOUR}(\mathbf{x}_S, 0, L_c, I, LM)$ 
21:             $\mathcal{C} \leftarrow \mathcal{C} \cup \{C_{\text{outer}}\}$               ▷ collect new contour
22:             $LM(u, v) \leftarrow L_c$ 
23:          else                                             ▷  $I(u, v)$  is a background pixel
24:            if  $(L_c \neq 0)$  then
25:              if  $(LM(u, v) = 0)$  then                    ▷ hit new inner contour
26:                 $\mathbf{x}_S \leftarrow (u-1, v)$ 
27:                 $C_{\text{inner}} \leftarrow \text{TRACECONTOUR}(\mathbf{x}_S, 1, L_c, I, LM)$ 
28:                 $\mathcal{C} \leftarrow \mathcal{C} \cup \{C_{\text{inner}}\}$       ▷ collect new contour
29:                 $L_c \leftarrow 0$ 
30:   return  $(\mathcal{C}, LM)$ .                                     ▷ return the set of contours and the label map

```

Fortsetzung in Alg. 11.4 ▷▷

- Zunächst wird das Bild  $I$  (pixelMap) und das zugehörige label map  $LM$  (labelMap) an allen Rändern um ein zusätzliches Pixel vergrößert, wobei im Bild  $I$  diese Pixel als Hintergrund markiert werden. Dies vereinfacht die Verfolgung der Konturen, da bei der Behandlung der Ränder nun keine besonderen Vorkehrungen mehr notwendig sind.
- Die gefundenen Konturen werden in einem Objekt der Klasse ContourSet zusammengefasst, und zwar getrennt in äußere und innere Konturen. Diese sind wiederum Objekte der Klassen OuterContour bzw. InnerContour mit der gemeinsamen Überklasse Contour. Jede Kontur besteht aus einer geordneten Folge von Koordinatenpunkten der Klasse Node (Definition auf S. 199). Als dynamische Datenstruktur für die Koordinatenpunkte sowie für die Menge der äußeren und