Composantes connexes

Introduction

Dans une image binaire (e.g. voir tableau 1 et figure 2), il est courant d'étiqueter les composantes connexes afin de pouvoir distinguer et extraire les différentes structures "déconnectées". Ceci se traduit par l'affectation d'une valeur ("intensité") distincte à chaque composante connexe.

L'étiquettage dépend directement la connexité considérée, cette connexité définissant la notion de voisinage.

Deux types de connexité sont généralement considérées: la 8-connexité (tableau 3) et la 4-connexité (tableau 4). En pratique, cette représentation de la connexité est faite à l'aide d'un masque binaire spécifiant, par rapport à l'élément central, la liste des voisins (points à 1). En plaçant ce masque en chaque point d'une image binaire, on ainsi identifier les points voisins (connexes), appartenant donc à la même composante connexe.

L'étiquetage des composantes permet, par exemple, d'extraire chaque composante connexe par simple seuillage de l'image étiquetée. Cet algorithme peut permettre d'éliminer les petites composantes connexes, associées

0	0	0	0	0	0	0
0	0	1	1	1	1	0
0	1	1	1	0	1	0
0	0	1	1	1	1	0
0	1	0	0	0	0	0
0	1	1	0	0	1	0
0	0	0	0	0	0	0

Table 1: Image binaire

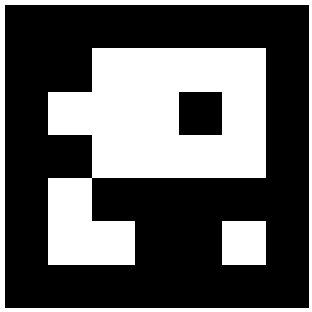

Table 2: Image binaire

1	**1**	**1**
1	**1**	**1**
1	**1**	**1**

Table 3: 8-connexité.

0	**1**	0
1	**1**	**1**
0	**1**	0

Table 4: 4-connexité.

à du bruit, ou bien encore de trouver le plus grand objet (composante connexe) présent dans une image binaire.

English:
In a binary image (e.g. see table 1 and figure 2), it is usefull to label connected components in order to distinguish and extract different objects ("disconnected" with respect to each other). Each connected component is usually labelled by a specific value (a "label"). Labelling directly depends on the considered connexity, based on the notion of neighborhood. Two kinds of connexity are usually considered: 8-connexity (table 3) and 4-connexity (table 4). In practice, connexity is represented using a mask specifying, with respect a central pixel, the set of neighbors. By placing this mask in any image point, one can identify neighbor of this point, therefore belonging to the same connected component. The labelling of connected components allows, for instance, to extract each connected component by thresholding the labelled image. This algorithm can enable to remove some connected components (e.g. of small size, related to artefacts/noise), or to select specific ones (e.g. largest object of a binary image).

1 La plus grande composante connexe - Largest connected component

Dans le cas de l'image de la table 1, quelles sont les composantes connexes (affecter une intensité spécifique à chacune d'elle), en 4 et 8 connexité ?

Implémenter l'algorithme permettant d'extraire la plus grande composante connexe. On utilisera la fonction `scipy.ndimage.measurements.label` pour étiqueter des composantes connexes. On analysera ensuite l'histogramme de l'image étiquetée afin d'appliquer le seuillage approprié, permettant d'extraire la plus grande composante connexe.

Les étapes sont:

1. Générer l'image `Ic` des composantes connexes étiquettées, en utilisant `scipy.ndimage.measurements.label`.
2. Calcul l'histogramme de `Ic`.
3. Cherche l'intensité `T` (abscisse de l'histogramme) correspondant à la plus grande valeur de l'histogramme (ordonnée de l'histogramme): cette intensité sera donc celle de la plus grande composante connexe (le plus grande nombre de pixels associés à une même intensité). Attention à ignorer l'intensité du fond lors de la recherche.
4. Générer l'image résultat `R` par seuillage de l'image `Ic`, de la manière suivante: si l'intensité dans `Ic` vaut `T` (i.e. $Ic == T$), alors l'intensité dans `R` vaut 255, sinon 0. On utilisera la fonction `numpy.where`.

L'image `R` ne doit contenir que la plus grande composante connexe.

English:

In the case of the image of table 1, what are the connected components, for a 4-connexity and a 8-connexity ?

Implement the algorithm allowing to extract the largest connected component. Use the function `scipy.ndimage.measurements.label` for labelling. Analyze then the histogram of the labelled image to automatically find the label ("intensity") corresponding to the largest connected component : the component can be finally extracted by thresholding the labelling image using, for the threshold, the previous found label.

The step of the algorithm are :

1. Generate the labelled image `Ic` using `scipy.ndimage.measurements.label`.
2. Compute the histogram of `Ic`.
3. Find the intensity `T` (histogram abscissa, x-axis) corresponding to the largest value of the histogram (y-axis): this intensity will be the one corresponding to the largest connected component (i.e. the largest number of pixels associated to a same intensity). Note : intensities related to the background must be ignored.
4. Generate the result image `R` by thresholding the `Ic` image : if the intensity is `Ic` egals `T` (i.e. $Ic == T$), then the intensity in `R` becomes 255, and 0 otherwise. For this, think about using the `numpy.where` function. This resulting image `R` should finally only contain the largest connected component.

2 Distribution de composantes connexes

Déterminer automatiquement le nombre de biscuits présents dans l'image 1 ainsi que leur taille moyenne. On considérera qu'un biscuit est associé à une composante connexe de taille significative, d'au moins 100 pixels dans notre cas. L'algorithme à coder est le suivant:

1. Seuiller l'image avec un seuil déterminé automatiquement par la méthode d'otsu (voir TD1).
2. Etiqueter les composantes connexes de l'image binaire (image précédemment seuillée).
3. Calculer l'histogramme de l'image étiquetée (chaque intensité est associée à une étiquette).
4. Calculer le nombre de biscuits et leur taille moyenne en analysant l'histogramme (nombre de pixels par composante).

En fin de traitement, le programme doit afficher: "Il y a 12 biscuits, de taille moyenne 2276.91666667 pixels"

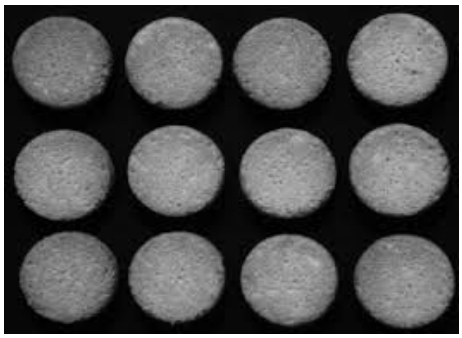

Figure 1: Image de biscuits - Image of biscuits

English:
The purpose is to automatically determine the number of biscuits in an image (see Figure 1) as well as their mean size. One considers that a biscuit is associated to a connected component of, at least, 100 pixels. The differents of the algorithm to code are :

1. Threshold the image (using the Otsu method for threshold determination).
2. Label connected components of the binary thresholded image.
3. Compute the histogram of the labelled image (each intensity being associated to a specific label).
4. Determine the number of biscuits and their mean size by analyzing this histogram.

Note : expected results are "12 biscuits" and mean size of "2276.91666667 pixels"