

Lecture d'un dé

Le but de ce mini problème est de lire une image représentant une face d'un dé et de trouver le nombre de points présents sur cette face. Vous trouverez sur le site une image de dé (mais vous pouvez tester avec n'importe quelle image) ainsi qu'une fonction permettant de la charger dans une matrice et de la convertir en noir et blanc. La matrice obtenue contient un 1 pour chaque pixel noir et un 0 pour chaque pixel blanc.

Nous supposons l'image très propre : aucun pixel noir parasite ailleurs que sur les points du dé, et pas de pixel blanc au milieu des points noirs (en réalité, il suffit que tous les pixels noirs d'un point du dé soient reliés entre eux pour que l'algo fonctionne).

La méthode employée est simple : nous commençons par créer un graphe dont les sommets sont les pixels noirs de l'image, et les arêtes les couples de deux pixels noirs adjacents dans l'image (un pixel de coordonnées (i, j) est dit adjacents aux pixels $(i + 1, j)$, $(i - 1, j)$, $(i, j + 1)$ et $(i, j - 1)$ à condition que ceux-ci soient dans l'image).

Il ne restera plus qu'à compter les composantes connexes de ce graphe.

1 Création du graphe

1. Écrire une fonction **estDedans** prenant en entrée un couple c d'entiers et deux entiers n et p et indiquant si c est les coordonnées¹ d'un pixel dans un image de format (n, p) .
2. Écrire une fonction **voisins** prenant les mêmes entrées que la fonction précédente et renvoyant la liste des voisins de c qui sont dans l'image. La liste renvoyée aura donc entre deux et quatre éléments.
3. Écrire alors une fonction prenant en entrée une matrice de 0 et de 1 représentant une image de dé et renvoyant le graphe correspondant. Le graphe sera un dictionnaire de listes d'adjacence. Les clefs seront des couples : les coordonnées des pixels noirs de l'image.

2 Nombre de composantes connexes

1. (*Cours*) Écrire une fonction **une_cc** prenant en entrée un graphe g et un sommet s_0 de celui-ci et renvoyant la composante connexe de s_0 dans g .
2. Écrire une fonction comptant le nombre de composantes connexes d'un graphe. On pourra utiliser une version légèrement modifiée de la fonction précédente comme sous-fonction.
L'idée est de lancer **une_cc** sur chaque sommet de g mais en gardant le *même* dictionnaire pour garder en mémoire les sommets déjà vus (noirs) au fil des différents appels à **une_cc**.
3. En déduire le nombre de points présents sur l'image de dé fournie. Le résultat obtenu n'est pas forcément celui attendu, voyez-vous pourquoi ?

1. J'ignore comment la grammaire française préconise d'accorder cette phrase.