COLORIAGE D'UNE MOSAIQUE

Serge BEUCHER

Juin 1993

Introduction

L'objet de cet algorithme est de colorier une mosaïque de telle sorte que deux cellules adjacentes n'aient jamais la même couleur. Cet algorithme avait primitivement été écrit à des fins de visualisation uniquement. Il s'avère cependant utile pour partitionner une mosaïque dans laquelle on veut effectuer des traitements différents dans toutes les cellules adjacentes. Cet étiquetage permet alors de générer simplement les masques de travail.

On sait (théorème des quatre couleurs, en supposant que sa démonstration est bien exacte) que, théoriquement quatre couleurs suffisent. Cependant, même démontré, ce théorème ne donne pas de moyen opératoire pour arriver à une bonne répartition des couleurs. C'est pourquoi, et en attendant une solution géniale à ce problème, on se contente d'étiqueter les cellules de façon à faire intervenir le minimum de couleurs dans un temps raisonnable.

L'algorithme est décrit sous forme d'ordinogramme. Il a été écrit à l'origine pour le MorphoPéricolor. Le programme commenté correspondant en TIM est donné en annexe (pour les amateurs!).

Principe et réalisation

1) Principe

L'algorithme part d'une image binaire telle celle représentée à la figure 1. C'est une mosaïque X de cellules, chacune d'elles étant séparée de ces voisines par une frontière d'épaisseur un pixel. La mosaïque n'a pas besoin d'être connexe. X peut être formé de plusieurs grappes de cellules sans que la généralité de l'algorithme ait à en souffrir.

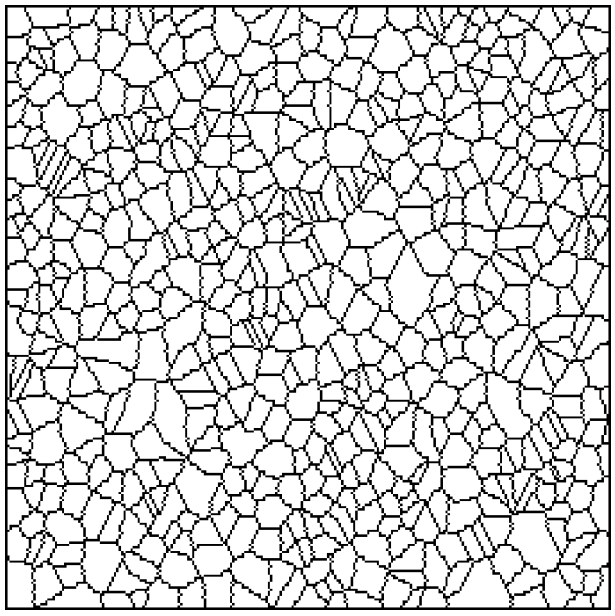

Figure 1 - La mosaïque initiale

L'algorithme travaille en plusieurs passes. Dans la première passe, on essaie de colorier la mosaïque en utilisant quatre couleurs seulement. Puis, et s'il reste des cellules qui n'ont pu être coloriées, on réitère la procédure sur les cellules restantes, en choisissant de nouvelles

couleurs. En pratique, on n'a jamais rencontré de configurations nécessitant plus de deux passages (en fait même, six couleurs suffisent dans la plupart des cas).

2) Réalisation

On se reportera également aux commentaires du programme. L'ordinogramme général est donné à la figure 2.

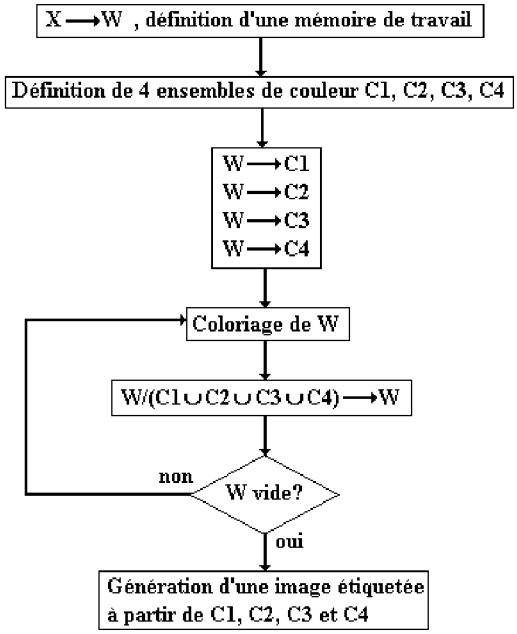

Figure 2 - Ordinogramme général

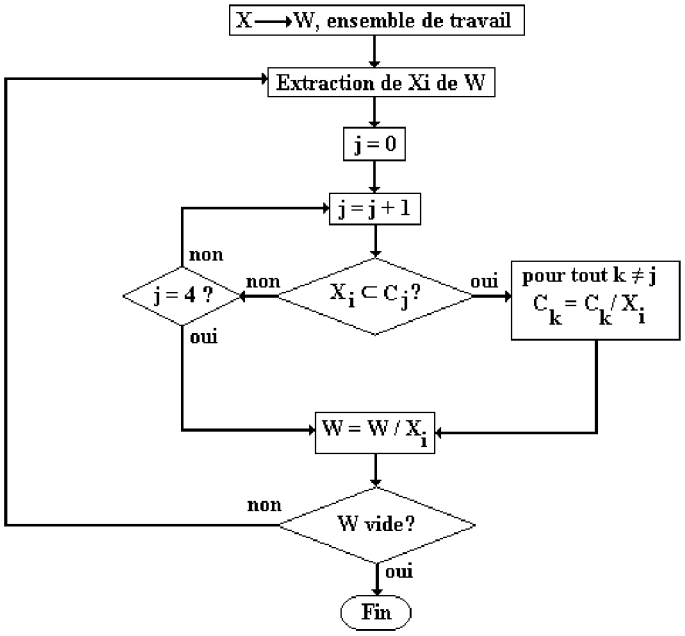

Figure 3 - Algorithme de coloriage

Le coloriage de la mosaïque à l'aide de quatre couleurs s'effectue de la manière suivante:

C1, C2, C3 et C4 sont quatre ensembles binaires contenant à chaque instant les cellules qui peuvent encore être coloriées dans la couleur correspondante. Si une cellule X_i appartient par exemple aux ensembles C1 et C3, cela signifie que cette cellule peut prendre la couleur 1 ou la couleur 3. Chaque cellule X_i est alors traitée individuellement. On test d'abord que la cellule appartient à C1. Si ce n'est pas le cas on passe à l'ensemble Ci suivant. Si X_i n'appartient à aucun Ci, alors cela signifie qu'on ne peut pas la colorier 'ses couleurs possibles sont épuisées). Par contre si X_i appartient à C1, X_i prend la couleur 1 et dans le même temps, les cellules adjacentes à X_i sont supprimées des autres ensembles Ci, $i \neq 1$. L'ordinogramme est fourni à la figure 3.

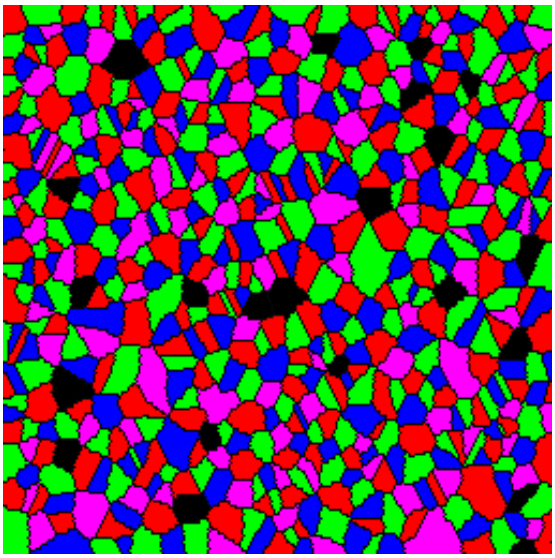

Figure 4 - Coloriage après la première étape

La figure 4 représente les cellules de la figure 1 coloriées après la première étape. La deuxième étape colorie alors les cellules restantes (il faut deux couleurs supplémentaires) pour donner finalement le coloriage de la figure 5.

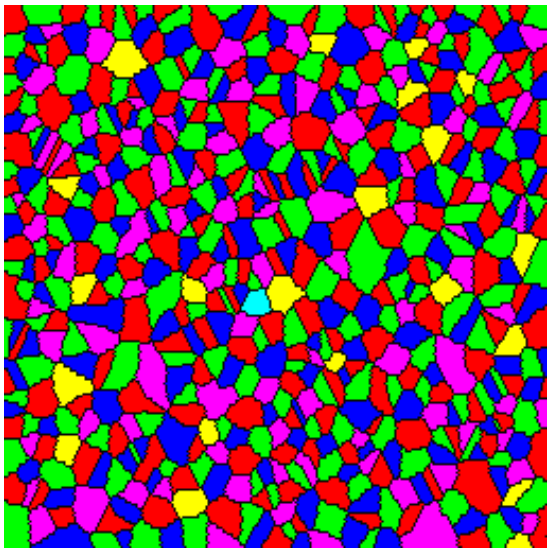

Figure 5 - Coloriage final de la mosaïque

Peut-on améliorer le coloriage?

En guise de conclusion, on peut se demander s'il n'existe pas un moyen d'améliorer le coloriage en réduisant le nombre de couleurs grâce au premier coloriage et à une réitération de la procédure mais en coloriant d'abord les cellules qui n'ont pu être coloriées dans la première phase, puis dans un deuxième temps, les cellules adjacentes à ces premières cellules, puis toutes les autres de proche en proche. Cependant cette approche pour séduisante qu'elle puisse paraître, ne marche pas. Elle ne réduit même pas le nombre de couleurs.

Programme TIM commenté

Rappelons pour faciliter la lecture que le langage TIM utilise une notation préfixée. Dans ce listing, les transformations ont été écrites en gras et les images en italique. Une routine commence toujours par le symbole deux-points (:) et se termine par un point-virgule (;). INPUT et OUTPUT désignent respectivement les images binaires d'entrée et de sortie. NOUTPUT représente une image numérique de sortie. WORK désigne des images de travail.

```
/* Routine VOISIN: Extraction des cellules voisines de ZD dans la mosaïque ID et stockage
dans OD */
: VOISIN
  OUTPUT OD
  INPUT ID ZD
  2 ZD OD XDILATE
  OD ZD %DIFF OD MOVE /* Différence ensembliste */
  OD ID %AND OD MOVE
  ID OD BUILD  /* Reconstruction binaire de ID, avec OD comme marqueur */
;

/* Routine COLOR_CHOICE: Test d'affectation de la couleur 1. La cellule ID peut prendre la
couleur 1 si d'abord cette couleur n'a pas déjà été interdite par un test précédent
(ID ∩ C1 ≠ Ø) */
: COLOR_CHOICE
  OUTPUT C4 C3 C2 C1
  INPUT SD ID
  WORK WK WL
  ID C1 %AND WK MOVE                  /* calcul ID ∩ C1 et stockage dans WK */
  WK AREA OR IF                       /* Si WK non vide, alors... */
                WK SD WL VOISIN       /* ..on extrait les voisins de WK dans WL */
                C1 WL %DIFF C1 MOVE   /* puis on interdit la couleur 1 à ses voisins */
                C2 WK %DIFF C2 MOVE   /* en les supprimant de C1. De la même */
                C3 WK %DIFF C3 MOVE   /* façon, on interdit les autres couleurs à */
                C4 WK %DIFF C4 MOVE   /* la cellule WK (égale à ID). */
             THEN
;

/* Routine FOUR_COLOR: Répartition des cellules de l'image ID dans 4 plans binaires
C1..C4, chaque plan correspondant à une couleur */
: FOUR_COLOR
  OUTPUT C4 C3 C2 C1
```

```
  INPUT ID
  WORK WK WL    /* deux mémoires de travail sont utilisées */
  ID C1 MOVE      /* à l'initialisation, ID est transférée dans la mémoire WK ainsi que */
  ID C2 MOVE      /* dans les 4 plans de couleur. En effet, toutes les couleurs sont     */
  ID C3 MOVE      /* possibles pour l'ensemble des cellules, l'algorithme procédant      */
  ID C4 MOVE      /* par élimination.    */
  ID WK MOVE
  BEGIN           /* traitement individuel des cellules */
     WK WL BUILD_FIRST  /* extraction de la première cellule de WK  dans WL */
     WL WK C1 C2 C3 C4 COLOR_CHOICE   /* test d'affectation de WL à C1 */
     WL WK C2 C1 C3 C4 COLOR_CHOICE   /* idem pour C2, etc...  */
     WL WK C3 C1 C2 C4 COLOR_CHOICE
     WL WK C4 C1 C2 C3 COLOR_CHOICE
     WK WL %DIFF WK MOVE    /* la cellule WL est éliminée de WK */
     WK AREA          /* et on passe à la cellule suivante si WK n'est pas vide */
   OR 0= UNTIL
;
```

/* Routine GREYSET: Affectation de la valeur VAL aux pixels de l'image numérique ODN marqués par le masque binaire ID. */

```
: GREYSET
  OUTPUT ODN
  INPUT ID
  PARAM VAL
  ID LOAD_TEMPLATE
  VAL ODN NSET
;
```

/* Routine COLOR_IMAGE: Programme principal. En entrée, la mosaïque initiale ID. En sortie, une image à teints de gris ODN où chaque cellule est étiquetée par une couleur (niveau de gris). 5 mémoires de travail, 4 d'entre elles contiennent les cellules d'une couleur donnée (W1 à W4), la cinquième est une image de travail (WK). GREYVAL est une variable entière tenant la couleur (niveau de gris) courante. */

```
: COLOR_IMAGE
  NOUTPUT ODN
  INPUT ID
  WORK W4 W3 W2 W1 WK
  INTEGER GREYVAL
  ODN NCLEAR    /* ODN est mis à zéro ainsi que GREYVAL  */
  0 -> GREYVAL
  ID WK MOVE    /* la mosaïque ID est transférée dans la mémoire de travail WK */
  BEGIN
    WK W1 W2 W3 W4 FOUR_COLOR    /*  les cellules sont réparties dans W1..W4   */
    GREYVAL 1+ -> GREYVAL        /*  en fonction de la couleur qui leur a été  */
    GREYVAL W1 ODN GREYSET       /*  assignée par la routine FOUR_COLOR */
    GREYVAL 1+ -> GREYVAL        /*  les 4 mémoires W1..W4 sont utilisées      */
    GREYVAL W2 ODN GREYSET       /*  comme masque pour générer l'image         */
    GREYVAL 1+ -> GREYVAL        /*  numérique ODN (routine GREYSET)    */
```

```
    GREYVAL W3 ODN GREYSET
    GREYVAL 1+ -> GREYVAL
    GREYVAL W4 ODN GREYSET
    1 63 ODN W1 THRESH      /* W1 contient toutes les cellules traitées (ODN positif) */
    WK W1 %DIFF WK MOVE     /* WK contient les cellules pas encore coloriées    */
    WK AREA OR   /*  si WK est non vide, on recommence la procédure de coloriage de  */
  0= UNTIL       /* WK avec une valeur initiale de GREYVAL égale à 4  */
;

/* Routine PALETTE: Palette de couleur pour visualisation  */
: PALETTE
  0 0 0 0 DEFINE_COLOR
  0 0 63 1 DEFINE_COLOR
  0 63 0 2 DEFINE_COLOR
  63 0 0 3 DEFINE_COLOR
  0 63 63 4 DEFINE_COLOR
  63 63 0 5 DEFINE_COLOR
  63 0 63 6 DEFINE_COLOR
  0 32 63 7 DEFINE_COLOR
  0 63 32 8 DEFINE_COLOR
  32 63 0 9 DEFINE_COLOR
  63 32 0 10 DEFINE_COLOR
  63 0 32 11 DEFINE_COLOR
  32 0 63 12 DEFINE_COLOR
  64 13 DO
        0 0 0 I DEFINE_COLOR
      LOOP
  128 64 DO
        24 24 24 I DEFINE_COLOR
      LOOP
  256 128 DO
        60 60 60 I DEFINE_COLOR
      LOOP
;
```