Quantification de couleurs par partition dynamique.

Luc Brun et Jean Pierre Braquelaire

LaBRI - URA 1304
Université Bordeaux I
351, cours de la Libération
F-33405 Talence, France

brun@labri.u-bordeaux.fr

Résumé : *La quantification de couleurs est un problème qui bien qu'il admette d'autres applications a pris de plus en plus d'importance au fur et à mesure du développement des écrans couleurs. Nous proposons dans ce papier diverses améliorations des algorithmes de type partitionement dynamique permettant de diminuer l'erreur de quantification et de réduire l'occupation mémoire.*

1. INTRODUCTION

Etant donné une image couleur I, il sagit de choisir un petit nombre de couleurs permettant de construire l'image la plus proche possible de I. Cette notion de proximité est mesuré par une fonction d'erreur entre l'image quantifié et l'image initiale. La recherche de la meilleure solution, encore appelée quantification optimale est un problème $\mathcal{NP}$ complet dans le cas d'images couleurs [11]. Le problème peut se formaliser de la façon suivante, trouver :

- une partition $\{C_1, C_2, ..., C_N\}$ de l'ensemble des couleurs de l'image,
- une couleur représentative q_i pour chaque C_i

minimisant :

$$EQM = \sum_{i=1}^{N} EQM(C_i)$$
avec (EQM : Erreur Quadratique Moyenne)
$$EQM(C_i) = \sum_{x \in C_i} \| x - q_i \|^2$$

Les méthodes les plus efficaces sont les méthodes de partitionement de l'ensemble des couleurs dérivant de la méthode de "coupe médiane" proposée par Heckbert [3]. La démarche des méthodes de partitionement est la suivante :

1. Trouver une surface qui partitionne l'espace des couleurs C en deux sous espaces C_1 et C_2 en diminuant le plus possible l'EQM.
2. Recalculer l'EQM sur C_1 et C_2.
3. Recommencer le découpage sur le sous espace ayant la plus forte EQM.

Pour diverses raisons, et notamment la simplicité de l'implémentation, la surface de découpage choisie est généralement un plan.
Le positionnement et l'orientation de ce plan sont les deux étapes critiques de tous les algorithmes de partitionnement. Les premiers algorithmes [3] utilisaient des orientations fixes, généralement orthogonales aux axes R,G,B. Wan, Wong et Prusinkiewicz [4] ont montré qu'on obtenait de meilleurs résultats en coupant orthogonalement à la direction de plus grande variation des couleurs.Cette méthode présente deux inconvénient.

1. La direction de plus grande variation se déduit du calcul des vecteurs propres de la matrice de covariance associés au cluster considéré. Cela induit une complexité suplémentaire.
2. Le fait de couper suivant des directions à priori quelconques empèche une structuration des données qui pouraient permettre de ne traiter à chaque étape que les données concernées.

Toutefois la nature n'étant pas entièrement régie par le hasard, on peut espérer trouver des axes privilégiés dans la distribution des couleurs d'images du monde réel. Nous proposons ici d'utiliser un espace de couleurs, se déduisant linéairement de l'espace R, G ,B et dont les propriétés permettent d'approcher de façon simple un découpage selon la plus grande variance.

2. LE MODÈLE I1 I2 I3

Cet espace de couleurs introduit par Yu-Ichi Ohta, Takeo Kanade et Toshiyuki Sakai [5] se déduit des coordonnées R, G,B par la transformation suivante :

$$\begin{cases} I1 &= \frac{R+G+B}{3} \\ I2 &= R-B \\ I3 &= \frac{2G-(R+B)}{2} \end{cases}$$

Il possède plusieurs propriétés intéressantes dont deux fondamentales :

1. Les trois axes (I1,I2,I3) correspondent dans l'ordre décroissant aux axes privilégiés de répartition de la couleur.
2. Les trois axes semblent à peu près correspondre aux directions privilégiées de la vision humaine.

Le point (1) signifie qu'étant donné une image, la direction de plus grande variation des couleurs sera située “près” de l'axe I1. Une fois cet axe coupé par un plan la direction possédant le plus grand pouvoir discriminant sera “près” de l'axe I2, et une fois celui-ci coupé, la plus grande variation sera près de I3.
Bien entendu, ceci n'est qu'un résultat statistique qui comme tout résultat de ce type souffre d'exceptions. Mais la pratique semble confirmer cette distribution de couleurs avec une étonnante régularité.
Le point (2) signifie simplement que l'homme a tendance à discriminer les couleurs suivant des axes proches de I1,I2,I3. S'il est bien connu que les signaux d'entrée captés par les cônes et les bâtonnets privilégient les directions R,G,B, les travaux de Levine [6] ont montré qu'un traitement situé en arrière de l'oeil transforme ces signaux R,G,B en :

- 1 signal de luminance : H_1,
- 1 signal d'opposition Jaune-Bleu : H_2.
- 1 signal d'opposition Rouge-Vert : H_3 ,

L'axe I_1 correspond bien à la luminance et les coordonnées chromatiques I2 et I3 bien que se situant plutôt dans l'opposition violet-vert et rouge-bleu ont des directions assez proche de H_2 et H_3. Nous avons constaté que des découpes perpendiculaires aux axes I1, I2, I3 améliorent sensiblement les résultats de la quantification.

La détermination de l'axe à couper et de la position du plan de découpage sur cet axe utilisent le calculs des moments d'ordre zéros, un et deux de la partition à découper. Ces moments étant définis par :

$$\begin{array}{llll} M0(C) & = \mid C \mid & = \sum_{x \in C} f(x) & \\ M1(C) & & = \sum_{x \in C} f(x)x & = (\sum_{x \in C} f(x)x_1, \sum_{x \in C} f(x)x_2, \sum_{x \in C} f(x)x_3) \\ M2(C) & & = \sum_{x \in C} f(x)x^2 & = (\sum_{x \in C} f(x)x_1^2, \sum_{x \in C} f(x)x_2^2, \sum_{x \in C} f(x)x_3^2) \end{array}$$

Où f(x) représente le nombre de pixel de couleur x dans l'image. On voit donc qu'un algorithme de partitionnement a impérativement besoin d'accéder rapidement au nombre de pixels de couleurs i,j,k. L'ensemble des algorithmes de partitionnement résolvent ce problème en allouant une matrice cubique T où T[i][j][k] code le nombre de pixels de couleur i,j,k dans l'image. Cette méthode présente toutefois plusieurs inconvénients :

- La taille nécessaire pour stocker toutes les couleurs d'une image 24 bits plans par un tel procédé est de : $(256)^3$(taille d'un entier), soit 64 Mo. Afin de réduire l'occupation mémoire, on procède généralement à une préquantification en ne gardant que les 5 bits de poids forts sur chaque composante. La taille nécessaire n'est alors plus que de 128 Ko. Toutefois cette préquantification crée une approximation de l'image initiale (une EQM non nulle), avant même le commencement de l'algorithme de quantification proprement dit. Cette première approximation, inévitable lorsque l'on utilise un tableau, peut être sensible lorsque l'on quantifie sur plus de 256 couleurs.
- On ne stocke pas simplement l'ensemble des couleurs de l'image mais l'ensemble des couleurs possibles, d'où perte d'espace mémoire et de temps gaspillé en parcourant des couleurs non présentes dans l'image.

Enfin l'utilisation d'un tableau impose d'englober l'ensemble des couleurs de l'image dans un parallélépipède rectangle. Ceci est parfaitement adapté à l'espace R,G,B mais devient inadéquat dès qu'on change d'espace.

En effet :

- Englober l'image du cube (R,G,B) dans l'espace I_1, I_2, I_3 par un parallélépipède rectangle nécessite l'utilisation de 256 Mo, une préquantification nécessitant tout de même 512 Ko de mémoire.
- Cet englobement implique également que de nombreuses cellules du tableau ne correspondront pas à des coordonnées R,G,B et auront toujours une valeur nulle. La matrice creuse stockant les fréquences dans l'espace (R,G,B) est donc englobé par une autre matrice creuse.

Il apparaît donc que l'utilisation d'un cube englobant, bien que sous optimum est acceptable tant que l'on travaille dans l'espace (R,G,B). Elle est en revanche inadaptée pour tout traitement sur des espaces déduits du système (R,G,B) par une transformation non orthogonale.

Dans la section suivante nous proposons une mise en œuvre optimale du point de vue de l'occupation mémoire. La structure de donnée et les algorithmes proposés s'adaptant sans aucune difficulté à d'autres espaces de couleur.

3. MISE EN OEUVRE

Le calcul des moments peut se calculer simplement sur des ensemble D_α, les calculs sur les partition s'en déduisant par une simple sommation. Ces D_α sont définis par :

$$I_i = \alpha \text{ et} \left\{ \begin{array}{l} m_j \leq I_j < M_j \\ m_k \leq I_k < M_k \end{array} \right. \quad \text{avec } i, j, k \in \{1, 2, 3\} \tag{1}$$

où :

- $[m_i, M_i[\times[m_j, M_j[\times[m_k, M_k[$ définit un ensemble de couleurs C.
- $\alpha \in [m_i, M_i[$

On sent intuitivement assez bien que des données triées résoudraient le problème. Partant de cette idée nous avons utilisé une structure de données organisée de la façon suivante :

- Un tableau V contennant l'ensemble des triplets de couleurs contenus dans l'image et pour chaque triplet le nombre de pixels correspondant à ce triplet.
- Trois vecteurs tab_I1, tab_I2, tab_I3 triant les valeurs de V suivant I1,I2,I3.
- Trois vecteurs entrée_I1, entrée_I2 et entrée_I3 référençant les tableaux tab_I1, tab_I2,tab_I3 et vérifiant la propriété suivante :

  $$\forall j \in \{1, 2, 3\} \;\; \forall k \in [Min_j, Max_j] \;\; \forall\, l \geq entre\_I_j[k] \;\; V[tab\_I_j[l]].I_j \geq k$$

  Ils permettent donc d'accéder directement à l'ensemble des couleurs dont la composante choisie est supérieure à k.

La résolution du problème (1) s'effectue alors en trois temps ; posons, pour fixer les idées (i=1, j=2 et k=3)

1. Trouver l'ensemble des couleurs vérifiant $(I_1 = \alpha)$ grâce à entrée_I_1. Cet ensemble est codé de façon continue dans tab_I_1.
2. Chercher par dichotomie dans la partie de tab_I_1 définie précédemment les indices vérifiant $m_2 \leq V[j].I2 < M_2$. Ces indices sont stockés de façon continue dans tab_I_1.
3. Parcourir l'ensemble des indices trouvés dans (2) pour ne retenir que ceux vérifiant $m_3 \leq V[j].I3 < M_3$.

Cette structure de données permet donc :

- De stocker et traiter uniquement les couleurs effectivement présentes dans l'image,
- d'accéder très rapidement aux données nécessaires à l'algorithme de partitionnement,
- de supprimer l'étape de préquantification nécessaire à tout algorithme de partitionnement.

On notera que cette structure peut facilement être installée en amont des algorithmes de partitionement existant sans grande modification de ceux-ci. Ces algorithmes bénéficient alors d'un accès rapide aux seules données présentes dans l'image. Ils améliorent ainsi leurs performances en temps et en espace.

4. CONCLUSION

Nous avons testé l'algorithme sur deux images 24-bits plans en les réduisant successivement en 2 puis 4, 8 jusqu'à 512 couleurs. Nous avons constaté de très bons comportements de l'algorithme pour des taux de réduction de couleurs importants. Les traits essentiels de l'image sont conservés en 2 ou 4 couleurs. Dès 8 couleurs, les principales couleurs de l'image sont préservées et celle-ci est parfaitement reconnaissable. A partir de 256 couleurs il devient difficile de distinguer l'original de la copie. Bien sur ces résultats sont peu généralisables, aucun algorithme ne pourrait par exemple préserver les couleurs d'une mire de télévision en seulement 8 couleurs. Ils donnent cependant une idée assez juste des résultats auxquels on peut s'attendre sur des images de scènes naturelles.
Nous avons comparé notre algorithme avec celui d'Hecbert disponible sur les Sparc stations (ppmquant). Le calcul de l'EQM pour chacune de ces quantifications laisse apparaître des rapports d'EQM entre les résultats issus de la méthode d'Heckbert et la notre qui varient entre 4 et 860. Ainsi qu'une diminution nettement plus rapide de l'EQM pour notre méthode.
La méthode que nous avons présentée est un compromis entre les méthodes de partitionnement avec recherche de l'axe de plus grande variation, (qui obtiennent de bons résultats avec des temps de calculs relativement long) et les méthodes de partitionnement classiques suivant les axes R,G,B plus rapides et donnant des résultats de moins bonne qualité. L'espace I_1, I_2, I_3 semble être plus adapté aux problèmes de quantification tant du point de vue de la distribution spatiale des couleurs que du point de vue physiologique. Les problèmes liés à l'utilisation de cet espace sont résolus par l'utilisation d'une nouvelle structure de donnée. Celle-ci permet de diminuer l'espace mémoire nécessaire, de diminuer les temps de calcul et de se passer de l'étape de préquantification. On obtient ainsi des résultats bien meilleurs que les algorithmes de quantification "classiques" en des temps de traitements sensiblement équivalent et n'excédant pas deux minutes pour les deux images testées.

Références

[1] Gervautz Michael & Purgathofer Werner , *A simple method for color quantization : Octree quantization*, Andrew S., Glassner editor Graphics Gem p 287-293, Academic Press, Boston 1990.

[2] Zhigang Xiang & Gregory Joy, *Color image quantization by agglomerative clustering*, IEEE Computer Graphics and Applications 272-17-16 p 44-48, 1994.

[3] Heckbert P., *Color image quantization for frame buffer display*, Proceedings of SIG-GRAPH'82. ACM NEW York, p 297-307, 1982.

[4] Wan S, Wong, S. and Prusinkiewicz P., *An algorithm for multidimensional data clustering*, ACM Trans. Math. Softw. 14,2 p 153-162, (June 1988).

[5] Yu-Ichi Ohta, Takeo Kanade & Toshiyuki Sakai, *Color information for region segmentation*, Computer Graphics and Image Processing **13** 222-241, 1980.

[6] Levine M.D., *Vision in man and machine*, Mc Graw-Hill Book Compagny 1985.

[7] Celenk Mehmet, *A color clustering technique for image segmentation*, Computer vision, graphics, and image processing **52** p 145-170,1990.

[8] Spencer W. Thomas, *Efficient inverse color map computation*, Andrew S., Glassner editor Graphics Gem p 116-125 & 528-535, Academic Press, Boston 1991.

[9] MacAdam D.L, *Color measurement and tolerances*, Official Digest, pp 1487-1531,Dec 1965.

[10] Trémeau Alain, *Contribution des modèles de la perception visuelle à l'analyse d'image couleur*, Thèse de l'université de Saint Etienne, 1993.

[11] Xiaoling Wu, *Color quantization by dynamic programming and principal analysis*, ACM Transactions on Graphics Vol. 11,No 4, october 1992, Pages 348-372.