Une image couleur cachée dans une image en niveaux de gris

Concours Jeune Chercheur : Non

Résumé

Dans ce papier, nous proposons une méthode originale d'insertion des informations couleur d'une image dans l'image en niveaux de gris correspondante. L'objectif de ce travail est de mettre en place une base de données images dont les images en niveaux de gris comprimées sont accessibles librement et dont la reconstruction de l'image couleur n'est possible qu'avec l'utilisation d'une clé secrète. Cette méthode est composée de trois étapes importantes : la quantification couleur, l'ordonnancement des couleurs et l'insertion des données cachées basée DCT. La nouveauté de ce papier concerne la construction d'une image d'index *associée à une palette couleur qui est également une image en niveaux de gris sémantiquement intelligible. Pour obtenir cette* image d'index *particulière, qui doit être robuste à l'insertion de données cachées, nous proposons un algorithme original d'ordonnancement des K couleurs :* l'algorithme de parcours en couche. *Enfin la méthode d'insertion est intelligemment choisie puisqu'elle repose sur une approche d'aqua-compression via l'utilisation d'un codeur JPEG hybride qui permet de comprimer les images avec un format standard du World Wide Web et dans le même temps propose une fonctionnalité d'insertion de données cachées.*

Mots clefs

Insertion de données cachées, aqua-compression, quantification couleur.

1 Introduction

Actuellement peu de solutions sécurisées sont proposées pour donner à la fois un accès gratuit à des images de basse qualité et à la fois un accès sécurisé aux même images de qualités supérieures. Nous proposons ici une solution à ce problème de sécurisation des bases de données images par l'intermédiaire d'une méthode d'insertion de données cachées. L'image peut être obtenue librement mais sa visualisation à haute qualité exige l'utilisation d'une clé secrète. Plus précisément, dans notre solution l'image en niveaux gris comprimée en JPEG est librement accessible mais seulement les possesseurs d'une clé secrète peuvent reconstruire l'image en couleur. Notre objectif est donc de protéger les informations couleur en incorporant ces informations dans l'image en niveaux de gris[1].

La méthode proposée est composée de trois grandes étapes : la quantification couleur (section 2.1), l'ordonnancement des couleurs (section 2.2) et l'insertion de données basée DCT (section 3). Dans l'étape de quantification couleur, le but est de trouver K couleurs et d'attribuer à chacun des pixels une de ces K couleurs. Dans l'étape d'ordonnancement, l'objectif est d'organiser ces K couleurs pour construire une palette de couleurs régulière chromatiquement et une *image d'index* sémantiquement intelligible. Dans l'étape d'insertion de données cachées basée DCT, le but est d'incorporer la palette de couleurs dans l'*image d'index* et être robuste à la compression JPEG.
Aucuns travaux similaires n'ont été portés à la connaissance des auteurs. On peut citer par exemple Wu *et al.* qui proposent de construire une nouvelle palette pour incorporer un bit de message dans chaque couleur de la palette [1], mais ils n'incorporent pas la palette dans l'*image d'index*.

2 Quantification couleur et algorithme de parcours en couche

Dans cette section nous présentons la quantification couleur et les étapes d'ordonnancement des couleurs.

2.1 Quantification couleur

La réduction du nombre de couleurs d'une image couleur est un problème de quantification classique. La solution optimale, pour extraire les K couleurs, est obtenue en résolvant :

$$\{P_{\imath,k}, C(k)\} = \arg \min_{P_{\imath,k}, C(k)} \sum_{\imath=1}^{N} \sum_{k=1}^{K} P_{\imath,k}.dist^2(I(\imath), C(k)), \tag{1}$$

où I est une image couleur de dimension N pixels, $C(k)$ est la k-ième couleur parmi les K couleurs recherchées, $dist$ est une fonction de distance dans l'espace couleur (L2 dans l'espace couleur RGB) et $P_{\imath,k} \subset \{0, 1\}$ est la valeur d'appartenance du pixel $\imath$ à la couleur k.
Une solution bien connue pour minimiser l'équation (1) et ensuite obtenir les K couleurs, est d'utiliser l'algorithme ISODATA des k-moyens [2]. $P_{i,k}$ est définie telle que :

$$\forall i, \forall k, P_{i,k} = \begin{cases} 1 \text{ si } k = \underset{\{k'\}}{\arg}\{\min_{k' \in [1,K]} dist(I(i), C(k'))\}, \\ 0 \text{ sinon,} \end{cases} \tag{2}$$

avec $C(k) = \frac{\sum_{i=1}^{N} P_{i,k} \times I(i)}{\sum_{i=1}^{N} P_{i,k}}$.

[1] Ce travail s'inscrit dans le cadre du projet TSAR 2005-2008 (Transfert Sécurisé d'images d'Art haute Résolution) retenu par l'ANR (Agence Nationale de la Recherche) dont l'un des objectifs est de donner un accès limité à la base de données de peintures numériques du Musée du Louvre à Paris.

Dans notre approche le nombre K est significatif en comparaison du nombre original de couleurs. Si nous utilisons l'algorithme classique des k-moyens, le nombre de couleurs extrait sera souvent en dessous de K. C'est le problème bien connu "de classes mortes". Pour surmonter ce problème, on initialise les valeurs $P_{\imath,k}$ en résolvant l'équation floue des k-moyens ci-dessous :

$$\{P_{\imath,k}, C(k)\} = \arg \min_{P_{\imath,k}, C(k)} \sum_{\imath=1}^{N} \sum_{k=1}^{K} P_{\imath,k}^{m}.dist^2(I(\imath), C(k)), \tag{3}$$

où m est le coefficient flou (m est positionné à 1.6 comme proposé dans [3]) et les $P_{\imath,k} \in [0,1]$ sont les coefficients d'appartenance flous. Cette équation est résolue par un algorithme k-moyens flou [4].

2.2 Algorithme de parcours en couche

Une fois que la quantification couleur a été traitée, l'image à K couleurs obtenue peut être représentée par une *image d'index* (grâce au valeurs des $P_{\imath,k}$) et une palette de couleurs (grâce aux valeurs $C(k)$). Notre but est alors de résoudre deux contraintes ; la première contrainte est d'obtenir une *image d'index* où chaque niveaux de gris est proche de la luminance de l'image couleur originale ; la deuxième contrainte consiste à obtenir une palette de couleurs dont les couleurs consécutives sont peu éloignées. Grâce à la quantification couleur, nous possédons déjà une *image d'index* et une palette de couleurs. Notre problème est alors de trouver une fonction de permutation qui permutent dans le même temps les valeurs de l'*image d'index* et les valeurs de la palette de couleurs. La fonction de permutation Φ est trouvée en résolvant l'équation :

$$\Phi = \arg \min_{\Phi} \left[\sum_{\imath=1}^{N} E_i^{ind} + \lambda \sum_{k=1}^{K-1} E_k^{palette} \right], \tag{4}$$

$$E_i^{ind} = dist^2(Y(\imath), \Phi(Index(\imath))), \tag{5}$$

$$E_k^{pal} = dist^2(Palette(\Phi^{-1}(k)), Palette(\Phi^{-1}(k+1)))], \tag{6}$$

où Y est la luminance de l'image couleur originale et λ est la valeur du Lagrangien. La fonction de permutation Φ est une fonction bijective dans $\mathbb{N}$ et défini tel que $\Phi : [1..K] \rightarrow [1..K]$.

Nous approchons l'optimum de l'équation (4) en utilisant un algorithme heuristique. Le but de cet algorithme est de trouver un ordonnancement pour les K couleurs tel que les couleurs consécutives soient peu éloignées et tel que la luminance des couleurs soient ordonnées des plus sombres aux plus claires. Cette ordonnancement définit pour chaque k-ième couleur une position k' qui nous donne la fonction Φ tel que $\Phi(k) = k'$.

Pour trouver un ordonnancement des K couleurs, l'algorithme parcours l'espace des couleurs pour construire la suite ordonnée de couleurs. La figure 1 illustre le chemin obtenu après un parcours dans le cube RGB. Ce parcours est effectué en "sautant" de couleur en couleur, dans l'espace couleur, en choisissant la couleur plus proche de la couleur courante. La première couleur de cette suite est choisie comme étant la couleur la plus sombre parmi les K couleurs. Une contrainte supplémentaire à ce parcours consiste à limiter la recherche de couleur aux couleurs peu éloignées en luminance. Cela signifie que le parcours dans l'espace des couleurs est limitée à une fenêtre définie sur les informations de luminance. Cette *algorithme de parcours en couche* peut être vu comme une sorte "de parcours 3D en spirale" dans l'espace des couleurs.

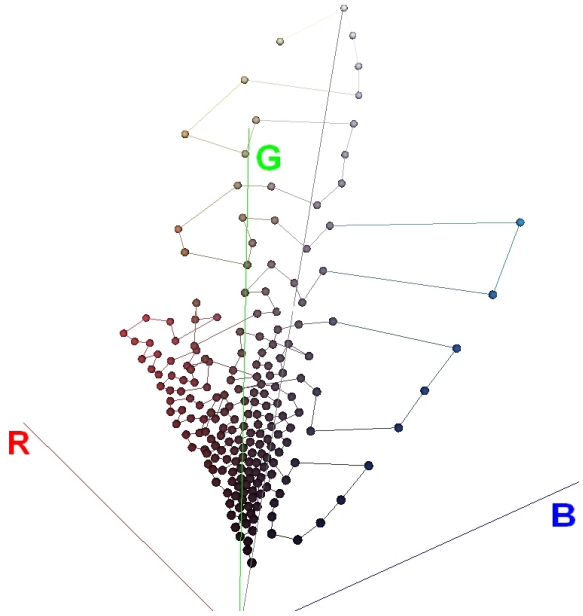


Figure 1 – *Vue du parcours en couche dans le cube RGB.*

3 Insertion de la palette de couleurs

3.1 Choix du nombre de couleurs

Dans la section précédente nous donnions la méthode utilisée pour construire une *image d'index* (peu éloigné de la luminance de l'image couleur originale) et une palette de couleurs (dont les couleurs consécutives sont proches). Le nombre de couleurs K était supposé connu. Dans cette section nous présentons une solution empirique pour choisir le nombre K de couleurs. On pourrait choisir un nombre de couleurs égal à 256 mais ce nombre n'est pas adapté pour construire une *image d'index* semblable à l'image de luminance. En effet, les 256 valeurs d'index sont couvertes dans l'*image d'index* alors qu'il y a souvent beaucoup moins de niveaux de gris dans l'image de luminance. Une solution plus intelligente est de choisir un nombre de couleurs égal à l'intervalle de niveaux de gris *significatif* de l'image de luminance. L'énergie de l'équation (4) doit être modifiée afin d'exprimer la réduction du nombre de couleurs. Seul le premier terme est changé :

$$E_i^{ind} = dist^2(Y(\imath), t + \Phi(Index(\imath))), \tag{7}$$

où t est une valeur de translation.

Pour choisir le nombre de couleurs à partir de l'histogramme de luminance nous définissons un *seuil significatif* correspondant à 1% de la valeur maximale de l'histogramme. Les valeurs de niveaux de gris inférieures à ce seuil sont considérées comme *non significatives*. On définit alors l'*intervalle significatif* de niveaux de gris tel que : la borne inférieure de cet intervalle est le premier niveau de gris *significatif*, et la borne supérieure de cet intervalle est le dernier niveau gris *significatif*. La largeur de l'*intervalle significatif* correspond au nombre K de couleurs et la borne inférieure est la valeur t de translation.

Remarquons que choisir un nombre de couleurs égal à la largeur de l'*intervalle significatif* réduit l'intervalle des index. Il en résulte une *image d'index* moins contrastée comparée à celle obtenue avec $K = 256$. L'*image d'index* est alors visuellement meilleure ; son PSNR est également amélioré par rapport à l'image de luminance. Remarquons également qu'attribuer à t la valeur de la borne inférieure de l'intervalle n'est pas nécessairement la meilleure solution. Cependant, en considérant que l'histogramme de l'*image d'index* est pratiquement plat, la valeur de t qui minimise au mieux le premier terme d'énergie de l'équation (7), est nécessairement proche de la valeur de la borne inférieure de l'*intervalle significatif*.

3.2 Méthode utilisée pour l'insertion de données cachées basée DCT

Les méthodes d'insertion de données cachées peuvent être utilisées pour faire de la transmission d'image sécurisée. Pour les applications traitant des images, l'objectif est d'insérer de manière invisible un message ou une marque à l'intérieur de l'image. L'insertion de données cachées est alors effectuée de manière différente selon la longueur du message et la robustesse désirée [6, 7, 8]. On définit généralement deux groupes de méthodes d'insertion de données cachées relativement au domaine d'insertion : les méthodes d'insertion dans le domaine spatial [9, 10, 11] et les méthodes d'insertion dans le domaine de fréquentiel [12, 13, 14]. Nous nous intéressons dans ce papier à une méthode d'insertion similaire à celle de [15] et qui combine les deux domaines spatial et fréquentiel, pour effectuer l'insertion. Nous proposons de plus une solution d'**aqua-compression** c'est-à-dire une solution permettant de faire de manière conjointe une insertion de données cachées et une compression.

Dans cette sous-section, nous décrivons dans un premier temps un codeur JPEG hybride avec une méthode d'insertion de données cachées dans le domaine fréquentiel. La méthode d'insertion de données cachées est effectuée après une transformation DCT. Chaque bit b_i, d'un message $M = b_1 b_2 ... b_m$ composé de m bits, est inséré dans le coefficient **DC** d'un bloc DCT [16]. Le processus d'insertion s'effectue en substituant le bit de poids faible (*Least Significant Bit*) du coefficient de DC par le bit b_i à insérer.

Avant insertion du message, nous calculons un facteur d'insertion (fonction de la longueur du message et de la taille de l'image) indiquant le nombre de bits à insérer par pixels de l'image. Le facteur d'insertion, en $bits/pixel$ est :

$$E_f = m/N. \tag{8}$$

L'*image d'index* est alors divisée en régions de taille $\lfloor 1/E_f \rfloor$ pixels. Comme nous utilisons la composante DC pour insérer un bit du message, notons que la taille minimale de l'image (en pixel) doit être au minimum égale à 64 fois le nombre de bits du message M à insérer ($N > 64m$). Chaque région est alors utilisée pour cacher **un seul** bit b_i du message. Ce bit est caché dans la composante DC de l'**un** des blocs appartenant à la région. Cette procédure de partitionnement garantie une répartition homogène du message sur toute l'image.

L'objectif est donc d'insérer le message M représentant la palette de couleurs. Pour cacher la palette de couleurs dans l'image nous devons donc insérer dans l'*image d'index* $m = 3 \times K \times 8$ bits plus une entête précisant les valeurs de K et de t. Par conséquent, le facteur d'insertion E_f, équation (8) est égal à :

$$E_f = (3 \times 8 \times K + 2 \times 8)/N. \tag{9}$$

Soit un bloc carré composé de n^2 pixels d'une image I, à partir de la DCT, le coefficient continu $F(0,0)$ du bloc est :

$$F(0,0) = \frac{1}{n} \sum_{i=0}^{n^2-1} I(i). \tag{10}$$

Dans la compression JPEG avec perte, le coefficient DC est quantifié et donne un coefficient quantifié $F'(0,0)$:

$$F'(0,0) = [F(0,0)/Q(0,0)], \tag{11}$$

où le $[.]$ est la fonction retournant le nombre entier le plus proche et $Q(0,0)$ est le coefficient de quantification.

Une solution classique pour insérer le message est de remplacer $F(0,0)$ par $F_w(0,0)$. Cette substitution prend donc en compte l'étape de quantification du codeur JPEG tel que :

$$F_w(0,0) = \begin{cases} \lfloor \frac{F(0,0)}{Q(0,0)} \rfloor \times Q(0,0) & \text{si } \lfloor \frac{F(0,0)}{Q(0,0)} \rfloor mod\ 2 = b_i, \\ \lceil \frac{F(0,0)}{Q(0,0)} \rceil \times Q(0,0) & \text{si } \lceil \frac{F(0,0)}{Q(0,0)} \rceil mod\ 2 = b_i, \end{cases} \tag{12}$$

où $\lfloor . \rfloor$ est la partie entière d'un nombre et $\lceil . \rceil$ est la fonction retournant le nombre entier supérieur ou égale le plus proche. Notons que la substitution de $F(0,0)$ par $F_w(0,0)$ est effectué avant l'étape de quantification. Remarquons également que $F_w(0,0)$ est un nombre entier.

Nous proposons maintenant d'améliorer la méthode d'insertion précédente. En effet, la modification du coefficient de DC ne prend pas en compte les informations spatiales du bloc correspondant. La modification du coefficient DC peut entraîner une gêne visuelle. Pour améliorer le résultat visuel, la modification du coefficient DC est obtenue par modification des niveaux de gris d'un certain nombre de

pixels appartenant au bloc correspondant. Les pixels modifiés sont les pixels possédant la plus forte variance. Ainsi, lors de l'insertion, au lieu de modifier le coefficient $F(0,0)$ en effectuant une insertion dans le domaine fréquentiel, nous modifions n_w pixels du bloc correspondant de sorte qu'après la DCT, on obtienne la valeur adéquate pour $F_w(0,0)$. L'insertion est alors faite dans le domaine spatial avec prise en compte de l'impact fréquentiel et de quantification JPEG. Les n_w pixels $I(\imath)$ sont modifiés pour obtenir de nouveaux pixels $I_w(\imath)$ tel que :

$$I_w(\imath) = I(\imath) - sign(F(0,0) - F_w(0,0)), \qquad (13)$$

où $sign(x) = -1$ si $x < 0$ et $sign(x) = 1$ si $x \geq 0$. Notons que le nombre de pixels à modifier n_w vaut :

$$n_w = [|F(0,0) - F_w(0,0)| \times n]. \qquad (14)$$

Remarquons également que lorsque $n_w > n^2$, nous appliquons une première fois l'équation (13) sur tous les pixels du bloc et nous répétons de nouveau cette opération sur $n_w \ mod \ n^2$ pixels.
Pour résumer notre méthode d'insertion (qui ajoute la fonctionnalité d'insertion de données cachées au codeur JPEG), l'équation d'un coefficient DC quantifié marqué est :

$$F'_w(0,0) = \frac{1}{n \times Q(0,0)} \Big(\sum_{i \in \Omega_w} I_w(\imath) + \sum_{i \in \overline{\Omega_w}} I(\imath) \Big), \qquad (15)$$

où Ω_w est l'ensemble des n_w pixels modifiés d'un bloc.

4 Résultats

Nous avons appliqué notre méthode sur une peinture numérique illustrée figure 2.a, de taille 523×778 pixels, donnée par le Musée de Louvre. L'image de luminance est illustrée figure 2.b et son histogramme est représenté figure 2.c.
Nous pouvons observer à partir de l'histogramme de luminance qu'un grand nombre de niveaux de gris sont non *significatifs*. Pour obtenir le nombre de couleurs K et la valeur de translation t, un seuillage automatique est réalisé et donne un intervalle de niveaux de gris *significatif* de $[20, 222]$ comme présenté section 3.1. Le nombre de couleurs et la translation automatiquement déduits sont : $K = 203$ et $t = 20$. Choisir un nombre de couleurs égal à la taille de l'intervalle de niveaux de gris garantit une forte réduction du premier terme de l'énergie de l'équation (7) sans forte augmentation du deuxième terme d'énergie de l'équation (6) et donc sans forte augmentation de la distorsion sur l'image couleur après insertion de données cachées.
En appliquant l'algorithme de quantification couleur présenté section 2.1 nous obtenons les images quantifiées figure 3.c pour $K = 254$ et figure 3.d pour $K = 203$. Pour ces deux images quantifiées, aucune différence visuelle ne peut être remarquées entre elles. En comparant les images d'index correspondantes (figure 3.a pour $K = 254$ et figure 3.b pour $K = 203$) nous pouvons constater visuellement que la réduction du nombre d'index permet d'obtenir une image en niveaux de gris plus plaisante visuellement (moins contrastée) pour l'image avec $K = 203$ couleurs. Nous détaillons maintenant l'algorithme proposé en utilisant $K = 203$. Une fois que la quantification avec $K = 203$ couleurs a été réalisée, une palette de couleurs et son *image d'index* sont obtenus. Les figures 4.a et 4.c. illustrent le résultat classique obtenu pour la phase de quantification couleur. On peut remarquer que l'*image d'index* ne permet pas de comprendre ni même d'identifier son contenu. Avec l'application de l'*algorithme de parcours en couche*, nous obtenons une palette de couleurs ordonnée présentée figure 4.b) et une *image d'index* (figure 4.d) sémantiquement intelligible. Notons que *l'algorithme de parcours en couche* ne change pas le contenu informationnel i.e la palette de couleurs et l'*image d'index* permettent de reconstruire la même image couleur avant et après déroulement de *l'algorithme de parcours en couche*. Nous détaillons maintenant la phase d'insertion des données. Avant l'insertion, le message à insérer (la palette de couleurs) est codé prédictivement puis arithmétiquement. La longueur du message diminue de 4888 bits à 3183 bits. Le facteur d'insertion est alors $E_f = 0.0078$ bits/pixel. L'*image d'index* est donc par-

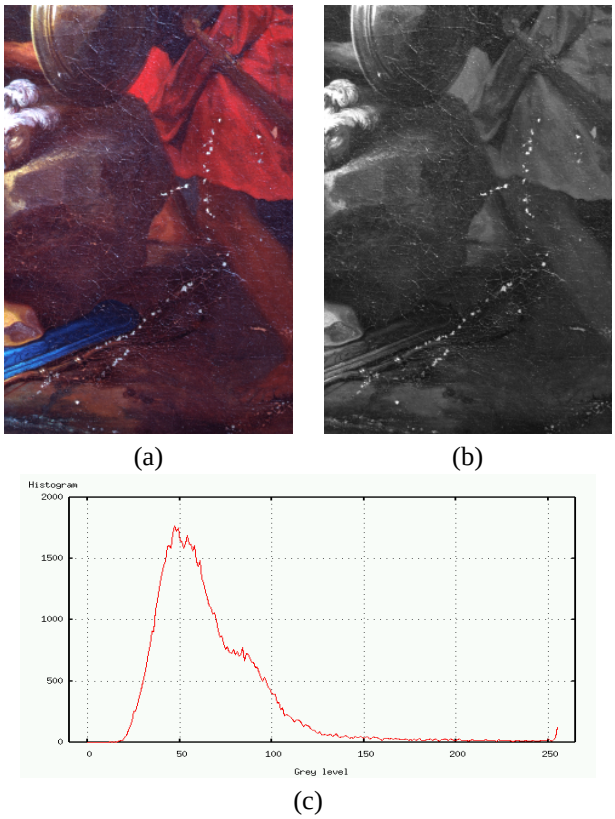
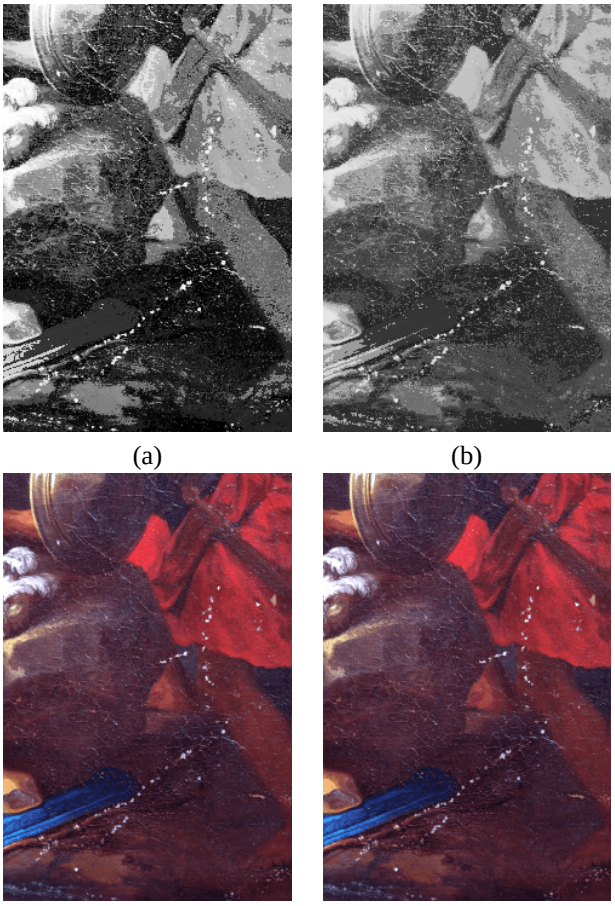


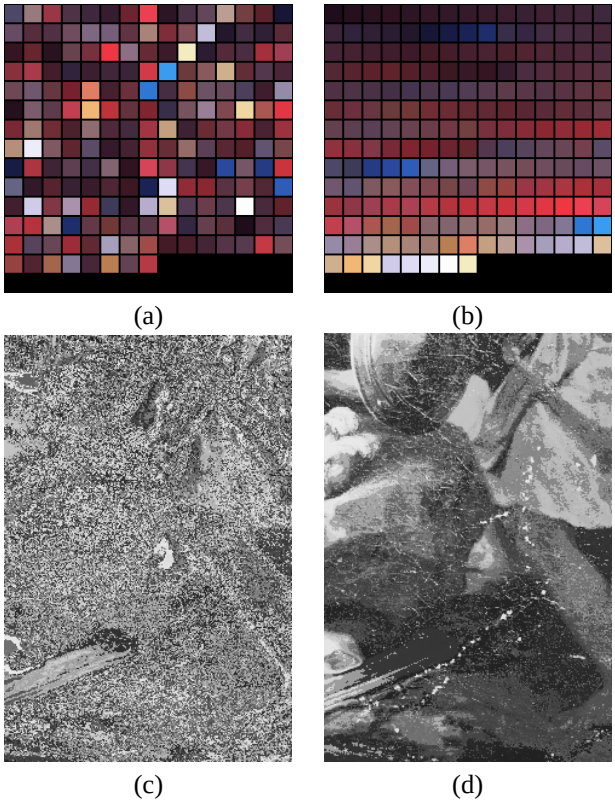
Figure 2 – *a) Image couleur originale, b) Luminance de l'image couleur originale, c) Histogramme de la luminance.*



(a) (b)

(c) (d)

Figure 3 – *Comparaison entre* $K = 254$ *couleurs et* $K = 203$ *couleurs. a) L'*image d'index *après l'ordonnancement des couleurs avec* $K = 254$*, b) L'*image d'index *après l'ordonnancement des couleurs avec* $K = 203$*, c) Image quantifiée avec* $K = 254$ *couleurs, d) Image quantifiée avec* $K = 203$ *couleurs.*

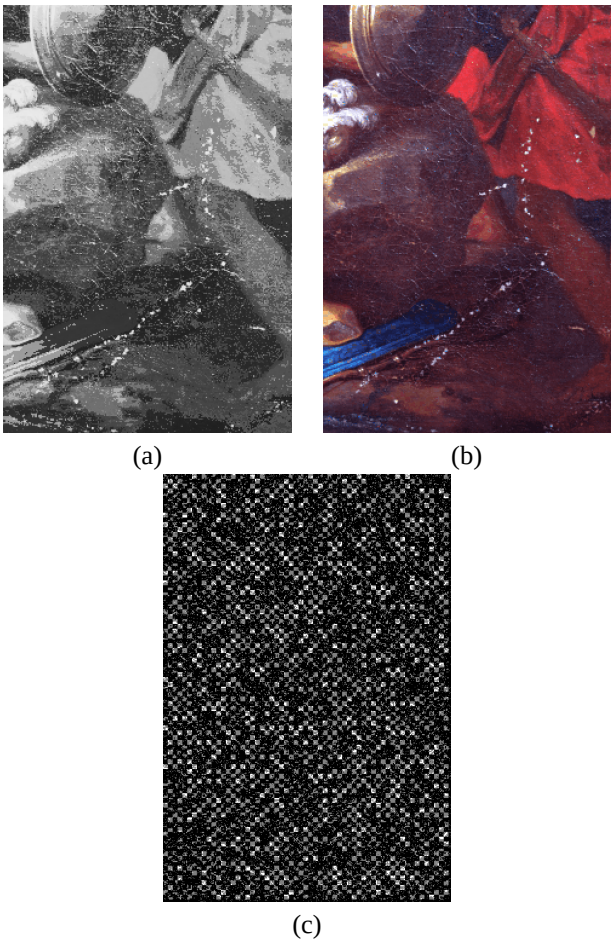


(a) (b)

(c) (d)

Figure 4 – *Application de* l'algorithme de parcours en couche. *a) La palette de couleurs de l'image originale quantifiée (*$K = 203$ *couleurs), b) La palette de couleurs après l'ordonnancement des couleurs, c) L'*image d'index *de l'image quantifiée, d) L'*image d'index *après l'ordonnancement des couleurs.*

titionnée en régions de $1/E_f = 128$ pixels et un bit du message est inséré dans un bloc appartenant à une région. Pour répartir le message sur l'image avec une distribution dépendant de la clé, nous utilisons une clé secrète de 128 bits comme "germe" pour le générateur de nombres pseudo aléatoires. Cette clé secrète est également utilisée pour crypter la palette de couleurs avant l'insertion. L'image est alors comprimée avec notre codeur hybride JPEG (compression + insertion de données cachées) avec un facteur de qualité de 100%.



(a) (b)

(c)

Figure 5 – *Insertion de données cachées basée DCT. a)* Image d'index *marquée avec* $K = 203$ *et* $m = 3283$ *bits, b) Image couleur reconstruite à partir de l'*image d'index *marquée, c) Image de différence entre l'*image d'index *et l'*image d'index *marquée.*

Une fois que notre codeur JPEG hybride (compression + insertion de la palette couleur) a encodé l'*image d'index*, il est possible avec un décodeur JPEG classique et sans clé d'accéder librement à l'image JPEG en niveaux de gris représentée figure 5.a). Avec la clé secrète, la palette de couleurs est extraite et l'image couleur est reconstruite. La figure 5.b montre l'image couleur reconstruite à partir de l'*image d'index* marquée. On peut observer que la qualité de l'image est très bonne. La valeur du PSNR entre l'image quantifiée en $K = 203$ couleurs et l'image couleur reconstruite à partir de l'*image d'index*-marquée de $41.2dB$, confirme cette évaluation subjective. La méthode d'insertion de données cachées utilisée et la proximité des couleurs consécutives dans la palette de couleurs sont à l'origine de ce bon résultat. La figure 5.c montre l'image de différence calculée entre l'*image d'index* et l'image reconstruite à partir de l'*image d'index-marquée-comprimée*. On peut remarquer que la modification de pixels se produit sur l'ensemble de l'image. On peut également remarquer que même si la modification de l'*image d'index* est dense, l'image reconstruite est toujours satisfaisante visuellement.

5 Conclusion

Dans ce papier, nous avons proposé une méthode pour cacher les informations couleur dans une image en niveaux de gris comprimée. Cette méthode est composée de trois étapes importantes qui sont la quantification couleur, l'ordonnancement des couleurs et l'insertion de données cachées. L'originalité de ce papier est de construire une *image d'index* qui est une image en niveaux de gris intelligible sémantiquement. Pour obtenir cette *image d'index* particulière, un algorithme original d'ordonnancement en K couleurs est proposé : *l'algorithme de parcours en couche*. Un codeur JPEG hybride permet d'insérer la palette de couleurs au sein de l'*image d'index*. Ce processus d'insertion de données cachées permet de comprimer les images avec un format standard du Word Wide Web et donne une solution prête-à-l'emploi pour publier de manière sécurisée les peintures numériques du Musée de Louvre.

Remerciements

Nous remercions M. Lahanier Christian du C2RMF (Centre de Recherche et de Restauration des Musées de France) et le Musée du Louvre pour nous avoir donné un accès aux peintures numériques privées du Louvre et également pour sa participation aux discussions du groupe de travail.

Références

[1] M.-Y. Wu, Y.-K. Ho, et J.-H. Lee. An Iterative Method of Palette-Based Image Steganography. *Pattern Recognition Letters*, 25 :301–309, 2003.

[2] G. H. Ball et D. J. Hall. ISODATA, a novel method of data analysis and pattern classification. Dans *In Proceedings of the International Communication Conference*, Juin 1966.

[3] R. Castagno et A. Sodomaco. Estimation of image feature reliability for an interactive video segmentation scheme. Dans *International Conference on Image Processing, ICIP'1998*, volume 1, pages 938–942, Chicago, USA, Octobre 1998.

[4] J. C. Dunn. A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, 3 :32–57, 1974.

[5] M. Chaumont et W. Puech. A Color Image in a Grey-Level Image. Dans *CGIV'06, Leeds, UK*, 2006.

[6] J.F. Delaigle, C. De Vleeschouwer, et B. Macq. Watermarking algorithm based on a human visual model. *Special Issue on Watermarking, Signal Processing*, 66(3) :319–336, 1998.

[7] Z. Duric. *Information Hiding, Steganography and Watermarking - Attacks and Countermeasures*. Kluwer Academic Publishers, Boston, 2001.

[8] F.A.P. Petitcolas, R. J. Anderson, et M.G. Kuhn. Information Hiding-A Survey. *IEEE, special issue on protection of multimedia content*, 87(7) :1062–1078, July 1999.

[9] W. Bender, D. Gruhl, N. Morimoto, et A. Lu. Techniques for Data Hiding. *I.B.M. Systems Journal*, 35(3-4) :313–336, 1996.

[10] N. Nikolaidis et I. Pitas. Robust Image Watermarking in the Spatial Domain. *Signal Processing*, 66(3) :385–403, 1998.

[11] G. Jagpal. Steganography in Digital Images. Dans *Dissertation, University of Cambridge, Selwyn College*.

[12] A.G. Bors et I. Pitas. Image watermarking using block site selection and DCT domain constraints. *Optics Express*, 3(12) :512–522, 1998.

[13] C.-C. Chang, T.-S. Chen, et L.-Z. Chung. A Steganographic Method Based Upon JPEG and Quantization Table Modification. *Information Sciences, Elsevier*, 141 :123–138, 2002.

[14] H.-W. Tseng et C.-C. Chang. High Capacity Data Hiding in JPEG-Compressed Images. *Informatic, Institute of Mathematics and Informatic, Vilnius*, 151(1) :127–142, 2004.

[15] F. Y. Shih et S. Y.T. Wu. Combinational image watermarking in the spatial and frequency domains. *Pattern Recognition*, 36 :969–975, 2003.

[16] D. Upham. Jpeg-jsteg, Modification of the Independent Jpeg Group's Jpeg Software for 1-bit Steganography in Jfif Output Files. Dans *ftp ://ftp.funet.fi/pub/crypt/steganography*, 1997.