WMUG: WATERMARK UNDER GRAPH

Jérôme FOLLI, Christian NGUYEN

Laboratoire GRIM
Université de Toulon et du Var, BP 35, 83957 La Garde Cedex, France
folli@univ-tln.fr, nguyen@univ-tln.fr

Résumé – Ce travail porte sur le tatouage d'images numériques ou *watermarking* (section 2.9), c'est à dire l'insertion d'une signature "invisible" dans une image afin de retrouver son propriétaire légitime. L'originalité de l'approche présentée repose sur la caractérisation de l'image par un graphe planaire et en particulier la possibilité de déterminer si deux graphes planaires sont isomorphes en temps linéaire. Cette représentation permet de résister efficacement aux déformations géométriques et d'améliorer certaines méthodes de signature d'images.

Abstract – This study concerns *watermarking* (section 2.9), i.e. embedding of an "invisible" signature into a picture to restitute its rightful owner. The originality of our approach is based on the characterization of the picture by a planar graph, specificaly the description of the isomorphism of two planars graphs in linear time. This representation is strongly resistant to geometrics transformations and improves some watermarking's methods.

1 Introduction

Dans le cadre des problèmes liés aux droits d'auteur et au copyright, nous nous intéressons à la signature des images numériques. Leurs copies étant par définition simples et parfaites, il est difficile de retrouver le propriétaire légitime d'une image.
Le tatouage d'images ou *watermarking* permet d'identifier le propriétaire d'une image par insertion d'une marque "invisible" appelée *signature*. Il est important qu'elle soit difficile à retirer sans dégradation visible de l'image. Actuellement, l'attaque universelle et efficace est la simulation d'une impression/numérisation par rééchantillonnage de l'image. Elle combine des transformations géométriques, locales et globales, linéaires et non linéaires, invisibles à l'œil nu. A ces déformations, s'ajoute une compression qui permet de modifier les données sans dégradation visible. L'outil de référence permettant ce type d'attaques est un *benchmark* du nom de *Stirmark* [1].
Dans cet article, nous nous limitons à un système de signatures d'images numériques peu sensible aux attaques basées sur les déformations géométriques. En cela, nous améliorons la méthode par compensation de mouvement basée sur un maillage triangulaire [6] en supportant des déformations géométriques plus générales.
Nous allons dans un premier temps définir une caractérisation de l'image basée sur la notion de graphe de voisinage. Puis nous allons décrire la méthode de signature utilisée et la façon de retrouver cette marque grâce aux propriétés des graphes, plus précisément leur "insensibilité" aux déformations géométriques usuelles (rotations, permutations, ...) puisque le graphe de l'image attaquée est isomorphe au graphe de l'image d'origine.

2 Méthodologie

Notre approche se base sur la notion que tout graphe aillant subi des déformations géométriques est isomorphe à l'original. Nous caractérisons une image à l'aide d'un graphe de voisinage et nous y insérons un bit de clé dans chaque région de l'image correspondant à un nœud du graphe. Dans la pratique, la construction du graphe associé à une image s'effectue en une passe et récursivement. Dans l'étape descendante (segmentation), à partir d'une image normalisée, nous mettons en évidence les régions de même couleur à l'aide d'une structure d'arbre quaternaire ou *quadtree*. Dans l'étape ascendante (fusion), à partir du quadtree, nous construisons le graphe de voisinage à proprement parlé. Puis, dans chaque zone, nous insérons un bit de clé dans l'image d'origine à l'aide d'un registre à décalage à rétroaction linéaire permettant de générer une suite de bits de longueur quelconque.
L'extraction de la signature est basée sur un algorithme établissant en temps linéaire un isomorphisme entre deux graphes planaires [5].
Détaillons à présent les différentes étapes de la méthode.

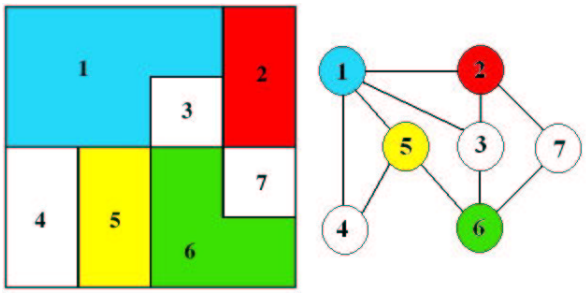

FIG. 1: *image et son graphe de voisinage associé*

3 Caractérisation de l'image

Celle-ci se fait en deux temps : normalisation puis segmentation.

3.1 Normalisation de l'image

Dans le but d'avoir des images normalisées afin de leur appliquer un traitement universel, nous utilisons une méthode de quantification. L'objectif est d'obtenir une image dont le nombre de couleurs soit fortement réduit tout en restant très proche, au sens de la perception visuelle humaine, de l'image originale. Ainsi, les structures définies par la suite (quadtree et graphe de voisinage) seront construites en un temps raisonnable et seront d'une complexité réduite tout en restant fonctionnelles. Ces structures orientant la diffusion de la signature, il est impératif qu'elles soient bâties sur une image normalisée tout en prenant en compte les propriétés du système perceptif humain. Dans une première approche, nous nous sommes limités à l'algorithme de quantification de Heckbert [3] afin d'implanter rapidement un prototype expérimental. L'algorithme utilisé par l'auteur, dit "median-cut", consistant grossièrement à subdiviser récursivement l'espace des couleurs de telle sorte que chaque représentant corresponde au même nombre de pixels, a fait depuis longtemps le preuve de son efficacité. Cependant, d'autres solutions à cette problématique, plus adaptées au contexte, sont actuellement à l'étude : caractérisation d'une image par sa matrice d'intensité, projection dans un espace des couleurs mieux adapté à la vision humaine (par exemple, L*a*b*), quantification MCEBC (*Main Color Emphasized Blob Coloring*) [2], etc.

3.2 Segmentation de l'image

Après avoir normalisé l'image, celle-ci est prête à être segmentée et décrite sous forme d'un graphe de voisinage à l'aide d'une structure intermédiaire : le quadtree. La notion de voisinage s'appuie sur une structure de type 4 voisins (nord, sud, est, ouest) pour chaque pixel : la d-connexité. La discrimination de deux nœuds se fait par la couleur : une région de l'image représentant un nœud du graphe est une zone de couleur unique. Un voisin est donc une région différente de celle considérée et adjacente par d-connexité.
Chaque nœud non terminal de l'arbre possède quatre fils : ils représentent la découpe en quatre de la portion de l'image traitée. Chaque feuille représente une zone de couleur unique de l'image normalisée.
Pour faciliter la recherche de voisins dans l'arbre, nous définissons un chemin pour chaque nœud basé sur le codage de Gray. Dans notre cas, ce chemin est codé sur un mot de 32 bits dont les 4 bits de poids fort représentent la profondeur du nœud et les 28 autres bits décrivent le chemin à proprement parlé (à raison de 2 bits par nœud).

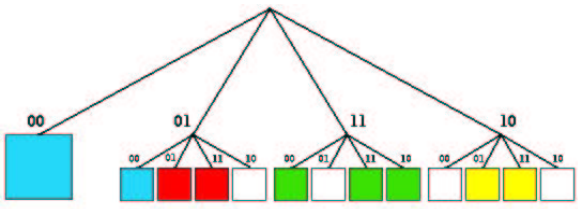

FIG. 2: *quadtree associé à l'image de la figure 1*

4 Insertion de la signature

Après avoir établi le graphe de voisinage de l'image normalisée, nous allons insérer, dans l'image d'origine, les bits de signature à l'aide d'un registre à décalage à rétroaction linéaire ou LFSR pour *Linear Feedback Shift Register* [4] permettant de générer une suite de longueur quelconque.

4.1 Principe du registre à décalage à rétroaction linéaire (LFSR)

Un LFSR binaire de longueur L est composé d'un registre à décalage contenant une suite de L bits $(s_i, ..., s_{i+L-1})$ et d'une fonction de rétroaction linéaire.
Le fonctionnement d'un LFSR binaire de longueur L est le suivant (voir la figure 3) : à chaque top d'horloge, le bit de poids faible s_i constitue la sortie du registre et les autres sont décalés vers la droite. Le nouveau bit s_{i+L} placé dans la cellule de poids fort est donné par une fonction linéaire :

$$s_{i+L} = c_1 s_{i+L-1} + c_2 s_{i+L-2} + ... + c_{L-1} s_{i+1} + c_L s_i$$

Les bits $(s_0, ..., s_{L-1})$ qui déterminent entièrement la suite

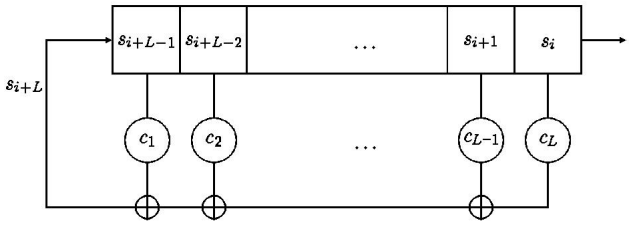

FIG. 3: *LFSR binaire de longueur L*

produite constituent l'état initial du registre.

4.2 Méthode d'insertion de la signature

Nous avons à insérer un message secret m de longueur r bits dans l'image d'origine. L'initialisation du registre $s = (s_0, ..., s_{L-1})$ détermine la clé de session de la signature (nous testerons toutes les clés de session possibles pour vérifier la présence de la signature). Nous prenons L pair et s équilibré afin que le nombre de vecteurs possible soit limité mais suffisant.
Un vecteur binaire v de longueur paire h est dit équilibré si le poids $\mathcal{W}(v)$ de v (i.e. le nombre de 1) est $h/2$, c'est à dire s'il possède autant de 0 que de 1.
A partir de ce LFSR on génère une suite de bits $K(s)$ de longueur $n = 4r$. La longueur de n est déterminée par le fait que nous insérons 4 bits dans l'image pour un bit de message.
Nous insérons la signature dans la composante bleue de tous les pixels d'une région donnée en respectant l'ordre de construction du graphe de voisinage de la façon suivante :

- si le bit m_i du message à insérer est 0 alors nous remplaçons les 4 bits de poids faible de la composante bleue de chaque pixel de la zone par $(s_{4i}, s_{4i+1}, s_{4i+2}, s_{4i+3})$,
- si le bit m_i du message à insérer est 1 alors nous remplaçons les 4 bits de poids faible de la composante bleue de chaque pixel de la zone par $(1 - s_{4i}, 1 - s_{4i+1}, 1 - s_{4i+2}, 1 - s_{4i+3})$.

L'image ainsi obtenue devient notre image signée.

5 Extraction de la signature

Pour extraire la signature d'une image éventuellement attaquée, il faut dans un premier temps établir un isomorphisme entre le graphe de voisinage de l'image attaquée et celui de l'image d'origine afin de déterminer la localisation géométrique et l'ordre de chaque bit de la signature. Il est à noter que pour retrouver la signature nous pouvons ne pas utiliser l'image d'origine, mais uniquement son graphe de voisinage.
Dans un second temps, nous établissons une corrélation entre le vecteur de bits obtenu et la signature insérée.

5.1 Etablissement d'un isomorphisme entre deux graphes planaires

Cette vérification se base sur un travail de J.E. Hopcroft et J.K. Wong [5]. Ils présentent un algorithme vérifiant si deux graphes planaires sont isomorphes en un temps linéaire sur le nombre d'arêtes.
Soient deux graphes planaires G_1 et G_2. L'algorithme associe un entier à chaque sommet et un couple d'entiers à chaque arête du graphe. Puis chaque graphe est simplifié grâce à une suite de réductions. L'ensemble des réductions s'effectue par ordre de priorité ce qui assure une forme canonique des graphes à chaque étape. Une réduction du graphe G est un remplacement de chaque sous-graphe de G d'un type donné par un sous-graphe étiqueté d'un autre type. Etiqueter un sous-graphe consiste à le réduire sans perte d'information. Il faut donc choisir des classes d'équivalences contenant assez d'informations pour reconstruire le sous-graphe d'origine à partir d'un identifiant donné. Le point important de cet algorithme réside dans le fait que la réduction de tous les sous-graphes d'un même type se fait simultanément et est étiquetée par le même identifiant : le représentant de la classe d'équivalence. Les réductions de plus haute priorité se font sur les boucles (arête ayant le même sommet comme origine et destination), les liens multiples (arêtes différentes ayant la même origine et la même destination) et les sommets de degré 1 (un sommet de degré n possède n arrêtes incidentes), puis sur les sommets de degré 2 et plus, avec des exceptions sur les sommets de degré 4. Quand l'algorithme ne peut plus appliquer de réductions, Hopcroft et Wong ont montré que les graphes ainsi obtenus correspondent à un polyèdre régulier parmis les cinqs existants (tétraèdre, hexaèdre, octoèdre, dodécaèdre et icosaèdre) ou à un sommet unique.
L'isomorphisme peut être testé en temps fini en vérifiant de façon exhaustive toutes les combinaisons possibles des deux graphes étiquetés. La recherche de l'isomorphisme se réduit donc à la détermination de la plus longue sous-séquence commune de deux séquences (à une permutation circulaire près). La longueur de cette sous-séquence en fonction de celle de la signature nous donne le taux d'isomorphisme entre les deux graphes.
Les figures 4 et 5 illustrent le comportement de l'algorithme. La figure 4 représente les graphes d'origine et la figure 5 les graphes réduits (nous utilisons ici des lettres pour noter les réprésentants des classes d'équivalences et les chiffres pour repésenter les composants non modifiés des graphes d'origine). Nous en déduisons les deux séquences suivantes : $CDA2JI12$ pour G_1 et $12CDA2JI$ pour G_2. Elles sont identiques à une permutation circulaire près, donc G_1 et G_2 sont isomorphes.

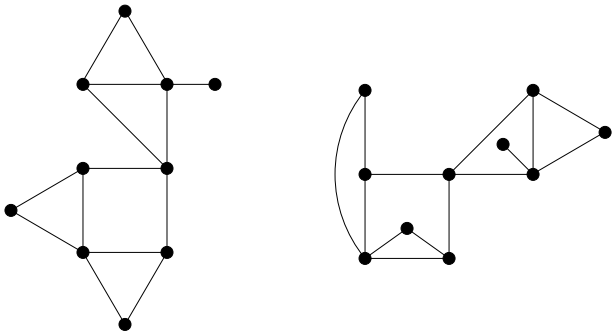

FIG. 4: *les graphes G_1 et G_2 sont-ils isomorphes ?*

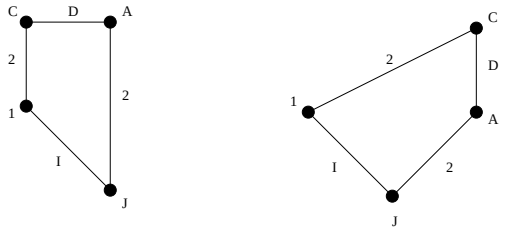

FIG. 5: *les graphes G_1 et G_2 réduits*

5.2 Vérification de la présence de la signature

Après avoir établi un isomorphisme acceptable entre les deux graphes s'il existe, nous vérifions le taux de corrélation entre un message généré et le message d'origine.
L'isomorphisme ordonne l'extraction des bits de signature à vérifier. En effet, la construction du quadtree étant déterministe et ayant établie un isomorphisme entre ces deux graphes, nous pouvons retrouver l'ordre des régions du graphe d'origine et donc l'ordre des bits du message puisqu'à chaque région on associe un bit de signature. Puis nous devons déterminer quel bit se cache dans chaque région. Pour cela nous initialisons le LFSR avec tous les vecteurs s possibles. Ils sont au nombre de $C_L^{L/2}$ car ils sont équilibrés. Ensuite nous extrayons pour chaque suite binaire générée $K(s) = (k_0, k_1, ..., k_{4r})$ le bit caché de la façon suivante : soit $B_i^j = (bj_i^0, bj_i^1, bj_i^2, bj_i^3)$ les 4 bits de poids faible du pixel j situé dans la région i. Nous cherchons le maximum entre le nombre de 0 et de 1 de la suite de bits X_i^j définie par :

$$X_i^j = (bj_i^0 \oplus k_{4i}, bj_i^1 \oplus k_{4i+1}, bj_i^2 \oplus k_{4i+2}, bj_i^3 \oplus k_{4i+3})$$

avec $\oplus$ l'opérateur XOR, telle que :

- si $\mathcal{W}(X_i^j) > 2$ alors le bit extrait est 1,
- si $\mathcal{W}(X_i^j) < 2$ alors le bit extrait est 0,
- si $\mathcal{W}(X_i^j) = 2$ alors si

$$\sum_{h=0}^{3} 2^h (bj_i^h \oplus k_{4i+h}) > 8$$

alors le bit extrait est 1, 0 sinon.

Nous procédons de même pour chaque bit de la région. Il est à noter que cette relation prends en compte le fait que les bits

de poids fort conservent mieux l'information car ils sont plus difficilement modifiables.
Enfin nous prenons le maximum entre le nombre de 0 et de 1 de tous les bits ainsi extraits. Ce résultat nous donne le i^{eme} bit du message généré.
Après avoir récupéré tous les bits du message ainsi généré, nous établissons une corrélation entre ce message et celui d'origine. Un taux de corrélation acceptable vérifie la présence de la signature.

6 Conclusion

Nous avons décrit dans cet article l'étude théorique de notre méthode de signature qui se base sur la caractérisation d'une image par un graphe de voisinage. Un prototype est en cours de développement en langage C++ qui insère une signature uniquement dans le domaine spatial. De plus, la génération de la suite de bits à insérer ne s'effectue qu'avec un seul registre.
Nous envisageons le développement d'une technique de signature hybride mêlant la caractérisation de l'image par un graphe de voisinage avec la transformée en cosinus discrète (DCT). Cette étude implique la description d'une topologie fréquentielle qui nous permettra de développer une technique de codage et de diffusion de la signature dans ce domaine. Enfin, l'utilisation de plusieurs LFSR filtrés par une fonction booléenne hautement non linéaire équilibrée permettrait une meilleure gestion de la clé de session et du message secret.

Références

[1] Fabien A.P. Petitcolas and Ross J. Anderson and Markus G. Kuhn *Attacks on Copyright Marking Systems*. Information Hiding, 1998.

[2] Sue K. Cho and Suk I. Yoo *MCEBC - A Blob Coloring Algorithm for Content-Based Image Retrieval System*. Computer Analysis of Images and Patterns, 1999.

[3] P.S. Heckbert *Color Image Quantization for Frame Buffer Display*. ACM Computer Graphics (ACM SIGGRAPH '82 Proceedings), 1982.

[4] R. Lidl and H. Niederreiter *Finite Fields*. Cambridge University Press, 1997.

[5] J.E. Hopcroft and J.K. Wong *Linear Time Algorithm for Isomorphism of Planar Graphs*. Department of Computer Science Cornell University, 1974.

[6] F. Davoine, P. Bas, P.-A. Hébert et J.-M. Chassery *Watermarking et résistance aux déformations géométriques*. CORESA, 1999.