Un système pour l'indexation rapide d'image de lettrine

Mathieu Delalandre, Jean-Marc Ogier

Un système pour l'indexation rapide d'image de lettrine

Mathieu Delalandre – Jean-Marc Ogier

Laboratoire L3i
Pôle Sciences et Technologie, Avenue Michel Crépeau, 17042 La Rochelle, France
{mathieu.delalandre;jean-marc.ogier}@univ-lr.fr

Résumé : *Cet article traite de l'indexation d'image de document appliquée aux livres anciens imprimés. Nous y abordons le problème de l'indexation des parties graphiques au sein des ces documents et plus particulièrement des lettrines. Le but de notre approche est le traitement de larges bases d'image. Pour ce faire, nous proposons un système d'indexation rapide exploitant une représentation à base de plages. Cette dernière permet en effet de réduire la complexité de nos algorithmes. Nous l'exploitons plus particulièrement au sein d'un algorithme procédant en deux étapes : une de comparaison des images et une autre de recalage. Cette dernière étape permet de palier aux problèmes de translation rencontrés entre les images de lettrine. Nous présentons quelques résultats et expérimentations de notre système en termes de temps de calcul et de précision de recherche.*

Mots-clés : indexation, complexité, RLE, comparaison d'images, recalage

1 Introduction

Cet article traite de l'indexation d'image et plus particulièrement d'image de document [DOE 98]. Il existe différentes catégories d'image de document : journaux, formulaires, cartes géographiques, dessins techniques, partitions musicales …. Dans cet article nous traitons des livres anciens imprimés[1]. En effet, depuis l'émergence des librairies numériques dans les années 1990 [BAI 03] de nombreux travaux de numérisation des collections historiques ont été entrepris. De grandes bases d'image de livres anciens imprimés sont désormais disponibles sur Internet et sont de plus amenées à se développer d'avantage dans le futur comme en témoigne l'ampleur des projets en cours (Google Print, Million Book Project, MSN ….)[2]. Des solutions d'indexation et de recherche sont donc maintenant attendues.

Différents travaux ont été proposés ces dernières années sur le traitement d'image de livre ancien imprimé [BAI 03] : compression, prétraitement (correction de luminance, filtrage, binarisation …), reconnaissance des caractères, séparation texte/graphique …. Le sujet qui nous intéresse ici est l'indexation des parties graphiques. Cette dernière a pour but d'appliquer des traitements sur les parties graphiques composant les images en vue de les comparer. En effet, les livres anciens imprimés sont majoritairement composés de caractères typographiques mais aussi de nombreuses parties graphiques comme les bandeaux, les illustrations, les lettrines et les cul-de-lampes. La Figure 1 en donne des exemples.

FIG. 1 – Parties graphiques dans les livres anciens imprimés

Il existe très peu de travaux traitant du problème de l'indexation des parties graphiques dans les images de livres anciens imprimés. À notre connaissance seuls quatre systèmes ont été proposés ces dernières années : [BIG 96], [BAU 05], [PAR 05] et [UTT 05]. Chacun de ces systèmes propose une approche permettant la mise en oeuvre d'un type spécifique de requête : même images [BIG 96], styles équivalents [PAR 05], régions communes [BAU 05], structures similaires [UTT 05]. Pour ce faire, ils exploitent tous des méthodes à base de fenêtre pour l'extraction de descripteurs des images (histogrammes, patterns, régions, textures …). Ceci les rend particulièrement complexes et peu adaptés au traitement de larges bases d'image. En effet, pour chaque pixel différents calculs peuvent être effectués : vecteurs de symétrie [BIG 96], distances locales de Haudorff [BAU 05], extraction de patterns [PAR 05], matrices de co-occurrence et de longueur de plages [UTT 05]. Certains auteurs ([BIG 96] [PAR 05], [UTT 05]) évoquent d'ailleurs ces problèmes de complexité au sein de leurs articles. Nous présentons ici un nouveau système d'indexation des parties graphiques pour les livres anciens imprimés. Le but de notre approche va à l'encontre des systèmes existants et vise au traitement de larges bases d'image. Aussi, avons nous centré nos travaux d'avantage sur les aspects complexité que sur la construction d'index hautement discriminant.

[1] Livres majoritairement édités entre le XV° et le XIX° siècles.

[2] http ://en.wikipedia.org/wiki/Digital_libraries

Dans la suite de cet article nous présentons tout d'abord dans la section (2) notre système. Dans la section (3) nous donnons quelques résultats et expérimentations de celui-ci en termes de temps de calcul et de précision de recherche. Finalement, dans la section (4) nous concluons et donnons nos perspectives sur ce travail.

2 Système d'indexation rapide

2.1 Introduction

Dans cette section nous présentons notre système d'indexation des parties graphiques. Plus particulièrement, nous avons choisi d'appliquer ce système à l'indexation des images de lettrine. Comme présenté précédemment, le but de notre approche est le traitement de larges bases d'image. Nous avons donc centré nos travaux d'avantage sur les aspects complexité que sur la construction d'index hautement discriminant. L'objectif est alors d'indexer "à la louche" les images de lettrine appartenant à de larges bases. Pour ce faire, nous avons choisi comme cas d'usage d'indexation la recherche d'image de lettrine de même classe de tampon. La Figure 2 en donne quelques exemples. Chacune de ces images correspond à la numérisation d'une impression papier produite par un tampon[3] [BLA 97]. On qualifie usuellement les impressions papier de ces images d'estampes ou d'empreintes. Les tampons étaient en effet utilisés pour produire différentes estampes sur un même ouvrage ou entre ouvrages différents. Ils pouvaient également être dupliqués en cas d'usure ou afin d'accélérer les processus d'impression.

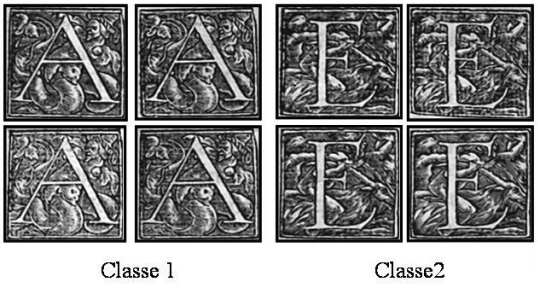


FIG. 2 – Exemples de classe de tampon

Ce cas d'usage d'indexation semble à première vue se ramener à un problème de comparaison d'images [GES 99]. Cependant il soulève des problèmes de translation des images. En effet, les images de lettrine que nous utilisons proviennent de bases en ligne sur Internet ou locales. Elles ont déjà été pré-traitées par différentes plate-formes pour leur filtrage, redressement, restauration …. Elles ont été par la suite segmentées de différentes façons [RAM 05] : automatiquement, semi-automatiquement ou par un utilisateur (par crop). Les images de même classe de tampon présentent donc des dimensions variables et sont surtout centrées différemment.

[3]Pièce de pierre, de cuivre ou de bois gravée pour la reproduction de parties graphiques ou de caractères.

De façon à répondre à nos problématiques (complexité et translation) nous avons mis en place un système se décomposant en deux étapes principales : une d'encodage des images en longueur de plages et une de comparaison par recalage. La Figure 3 suivante détaille l'architecture générale de notre système. Nous abordons chacune de ces deux étapes dans les sous-sections suivantes.

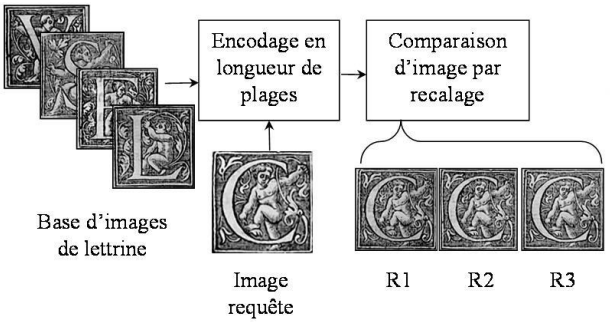


FIG. 3 – Architecture de notre système

2.2 Encodage en longueur de plages

L'objectif de notre approche est le traitement de larges bases d'image. De façon à satisfaire cette contrainte il est nécessaire dans notre système d'accélérer les temps de traitement de nos algorithmes. Pour ce faire il existe deux approches. La première consiste à utiliser des architectures matérielles dédiées [KUM 91] comme les processeurs pipeline, les ordinateurs massivement parallèle …. Cependant, la montée en puissance des machines et le développement des réseaux et d'Internet dans les années 90 ont quelque peu fait tomber en désuétude cette approche. L'autre approche consiste à employer des représentations adaptées aux algorithmes utilisés dans des buts de réduction de complexité. Il existe peu de travaux traitant de cette thématique. [VLI 98] par exemple utilise une représentation basée sur les contours pour exécuter rapidement des algorithmes exploitant des masques de voisinage comme l'érosion, la squelettisation …. [KIM 88] propose un système pour la détection rapide de contour basé sur une représentation à base de plages. Enfin, les auteurs dans [BIA 96] utilisent une représentation à base de composantes connexes pour la recherche rapide de sous-structures au sein d'images de document.

De cette manière, tout ces systèmes exploitent des représentations adaptées aux algorithmes qu'ils mettent en oeuvre. Dans le cadre de notre système nous nous sommes intéressés à la représentation à base de plages [WEN 98]. La plage permet en effet de représenter de façon compacte des successions de pixel comme l'illustre la Définition 1, elle semble donc adaptée pour faire de la comparaison d'images. La conversion d'une représentation raster vers plages est qualifiée d'Encodage en Longueur de Plages (RLE)[4] dans la littérature. Celle-ci est généralement appliquée aux images binaires, la Figure 4 (a) en donne un exemple.

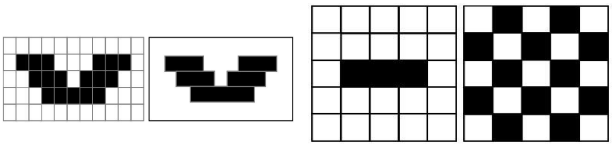
[4]Run Lenght Encoding : l'acronyme français est peu usité.

Définition 1 *Une plage est une séquence maximale de pixel définie par un triplet {o,(x, y),l} où :*

o *est l'orientation de la plage (verticale ou horizontale)*

(x,y) *est le point d'origine de la plage*

l *la longueur de la plage en pixel*



(a) raster/RLE (b) compacité forte/faible

FIG. 4 – Exemples d'encodage en longueur de plages

$$t_c = \frac{n_r}{n_p} \qquad n_r \in [0, n_p] \qquad t_c \in [0, 1] \tag{1}$$

$$c = ln\left(\frac{1}{t_c}\right) \qquad c \in]+\infty, 0[\tag{2}$$

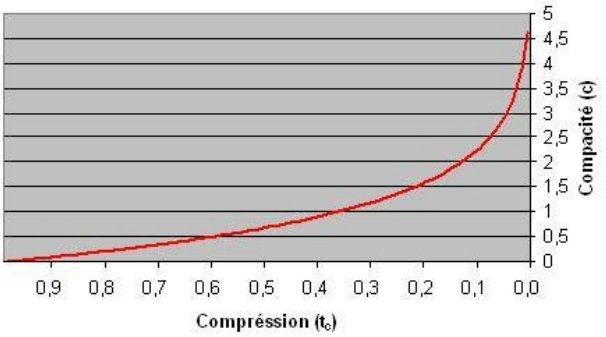


FIG. 5 – Courbe de compacité des plages

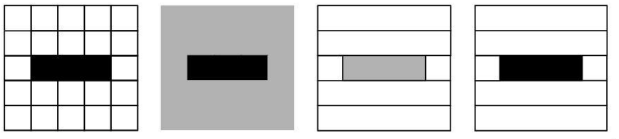
Le premier système à base de plages a été proposé par [PAV 78]. Aujourd'hui, les plages sont communément utilisées pour différentes applications [WEN 98] : reconnaissance d'écriture manuscrite, reconnaissance de symboles, segmentation …. Cependant, malgré le nombre de travaux existants sur les plages peu d'entre eux traitent de leur utilisation dans des buts de réduction de complexité. En effet, les plages ont surtout été utilisées pour la définition de graphes de plages (LAG[5], VSG[6] …) dans des buts de reconnaissance de formes. Selon notre point de vue, l'utilisation de la représentation à base de plages dans des buts de réduction de complexité peut s'évaluer selon deux critères : taux de compression et type de RLE. Nous présentons chacun d'entre eux par la suite.

Le taux de compression (t_c) s'exprime naturellement par le rapport du nombre de plage n_r (ie. run) sur le nombre de pixel n_p au sein d'une image (Équation 1). Ce taux peut également être utilisé pour définir une mesure de compacité des plages. Nous entendons ici par compacité des plages la façon dont celles-ci se concentrent au sein d'une image. La Figure 4 (b) donne des exemples d'images de forte et faible compacité. La compacité correspond donc à la dispersion des plages sur l'image et s'exprime naturellement comme l'inverse du taux de compression ($\frac{1}{t_c}$). Néanmoins, cette expression étant de nature exponentielle nous proposons de définir la compacité sous forme logarithmique (Équation 2). La Figure 5 donne la courbe de compacité (c) en fonction du taux de compression (t_c). Sur cette courbe deux zones se distinguent visuellement : une zone large de faible croissance suivie d'une zone courte de forte croissance. La représentation à base de plages permet donc une compression "aisée" des images pour des valeurs faibles et moyennes de compacité.

[5]Line Adjacency Graph

[6]Vertical Simple Graph

Le second critère d'évaluation concerne le type d'encodage en longueur de plages. En effet, les travaux de la littérature [WEN 98] appliquent naturellement, pour des problèmes de reconnaissance de formes, l'encodage aux pixels formes de l'image. Cependant, cet encodage peut être appliqué sur la forme et/ou sur le fond de l'image. De cette manière, trois encodages peuvent être effectués (Figure 6) : RLE forme (b) RLE fond (c) et RLE forme/fond (d). Ceux-ci peuvent encore se décliner en sous types si on considère[7] les encodages verticaux et/ou horizontaux des plages. Nous parlerons ici globalement d'encodage simple (b et c) et mixte (d) pour les désigner. Chacun de ces encodages possèdent des inconvénients et avantages. L'encodage simple compresse d'avantage le raster car une seule partie de ses pixels sont encodés. En contre partie, il polarise la représentation de l'image discrétisant alors le raster. L'encodage mixte permet une représentation entière du raster. Évidemment, en contre partie le taux de compression s'en trouve réduit.



(a) raster (b) RLE forme (c) RLE fond (d) RLE forme/fond

FIG. 6 – Types d'encodage en longueur de plages

Dans le cadre de notre système nous avons choisi d'exploiter un encodage mixte. Ce dernier nous semble en effet plus approprié dans des buts de comparaison d'images de lettrine. Le fond de ces images représente des objets (lettre,

[7]Nous reportons le lecteur à [BUR 98] sur ces aspects.

personnage ...) au même titre que leur forme (décor, ornement). Nous effectuons cet encodage à partir des images de lettrine en niveaux de gris ou binaires. Dans le premier cas une binarisation est appliquée utilisant un seuil par défaut de 128. Enfin, dans des buts liées à notre algorithme de comparaison nous effectuons deux encodages de nos images à la fois vertical et horizontal. Nous présentons cet algorithme de comparaison dans la sous-section suivante.

2.3 Comparaison d'images par recalage

Dans cette sous-section nous présentons notre algorithme pour la comparaison des images de lettrine exploitant notre RLE mixte vertical/horizontal. Nous avons présenté dans la sous-section (2.1) que ces images présentent des problèmes de translation. Pour palier à ce problème notre algorithme de comparaison procède en deux étapes : une étape de recalage suivie d'une étape de calcul de distance. Nous présentons chacune de ces étapes par la suite.

Le recalage de nos images est basée sur l'analyse de leurs histogrammes de projection verticale et horizontale des pixels noirs. La Figure 7 en donne des exemples. En effet, la segmentation des images de page de livre ancien imprimé induit l'apparition de bords (de couleur du fond) de dimensions variables sur les images de lettrine. Les paramètres de recalage se déduisent donc intuitivement à partir de l'analyse des pixels de couleur de la forme. Nos histogrammes sont construits à partir des parcours des plages verticales et horizontales de nos images. Nous comparons alors deux à deux chaque couple histogrammes (soit horizontaux soit verticaux). Pour ce faire, nous calculons une distance pondérée entre couple d'histogrammes $\{g, h\}$ (Équation 3 et 4). La pondération rend en effet la comparaison des histogrammes moins sensible aux fortes variations d'amplitudes [BRU 99]. Un calcul de distance par pondération semble donc plus adapté aux histogrammes des images de lettrine qui présentent ces fortes variations. Nos images pouvant être de dimensions différentes, une distance pondérée est calculée par offset $\{0, ., l-k\}$. Le delta (en x ou y) à appliquer correspond alors à l'offset de distance minimum. La Figure 7 donne deux images présentant un recalage de $d_x = 1$ et $d_y = 4$.

$$g_{1,2,..,k} \qquad h_{1,2,..,l} \qquad k \leq l \tag{3}$$

$$delta = \min \left(\bigcup_{j=0}^{l-k} \quad \sum_{i=1}^{k} \frac{|(h_i - g_{i+j})|}{h_i} \right) \tag{4}$$

Dans une deuxième étape nous procédons à la comparaison de nos images [GES 99] par calcul d'une (simple) distance pixel à pixel. Cependant, cette distance est évidemment obtenue à partir des représentations en plages ce qui nécessite la mise en oeuvre d'un algorithme adapté. Nous présentons[8] ci-dessous l'algorithme que nous avons défini. Nous le détaillons dans par la suite.

[8] Via le package LaTeX Pseudocode [KRE 05].

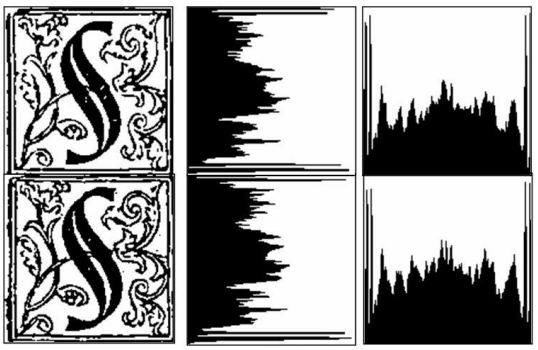


FIG. 7 – Comparaison d'histogrammes de projection

Pseudo-algorithme
2.1: DISTANCE(i_1, i_2, d_x, d_y)

$s \leftarrow 0 \qquad x_1 \leftarrow x_2 \leftarrow 0 \qquad a_1 \leftarrow a_2 \leftarrow 0$
pour chaque ligne $L1$ d'indice y de i_1
et $L2$ d'indice $y + d_y$ de i_2

faire
- $p_1 \leftarrow$ SUIVANT$(L1) \qquad x_1 += p_1.long$
- $p_2 \leftarrow$ SUIVANT$(L2) \qquad x_2 += p_2.long$
- **tant que** $(p_1 \neq fin) \vee (p_2 \neq fin)$
  - **faire**
    - **tant que** $x_1 \geq (x_2 + d_x)$
      - **faire**
        - **si** $p_2.couleur = p_1.couleur$ **alors** $s += p_2.long - a_2$
        - $p_2 \leftarrow$ SUIVANT$(L2)$
        - $x_2 += p_2.long$
        - $a_1 += p_2.long - a_2$
        - $a_2 = 0$
    - **tant que** $(x_2 + d_x) \geq x_1$
      - **faire**
        - **si** $p_1.couleur = p_2.couleur$ **alors** $s += p_1.long - a_1$
        - $p_1 \leftarrow$ SUIVANT$(L1)$
        - $x_1 += p_1.long$
        - $a_2 += p_1.long - a_1$
        - $a_1 = 0$

$s \leftarrow s/(\min(i_1.largeur, i_2.largeur) \times i_1.hauteur)$

Notre algorithme utilise par défaut l'encodage en plages horizontales des images i_1 et i_2. Il parcourt tout d'abord toutes les lignes $L1$ et $L2$ de ces images d'indices respectifs y et $y + d_y$. Il parcourt ensuite alternativement les plages de chacun des couples de lignes $\{L1, L2\}$ à l'aide de deux variables $\{p_1, p_2\}$. La Figure 8 en illustre le principe. Deux pointeurs $\{x_1, x_2\}$ sont utilisés pour indiquer les positions en cours de ces parcours. La ligne parcourue est alors toujours celle de position inférieure (tenant compte du d_x de recalage). La plage dernièrement lue de ligne de position supérieure sert alors de plage de référence. Les plages de la ligne parcourue sont comptabilisées par une variable s si elles sont de couleur identique à la plage de référence. Durant ce parcours alternatif deux accumulateurs $\{a_1, a_2\}$ sont utilisés. Ces derniers permettent de gérer les transitions de parcours ($L1 \leftrightarrows L2$) en comptabilisant les distances déjà parcourues sur les plages de référence.

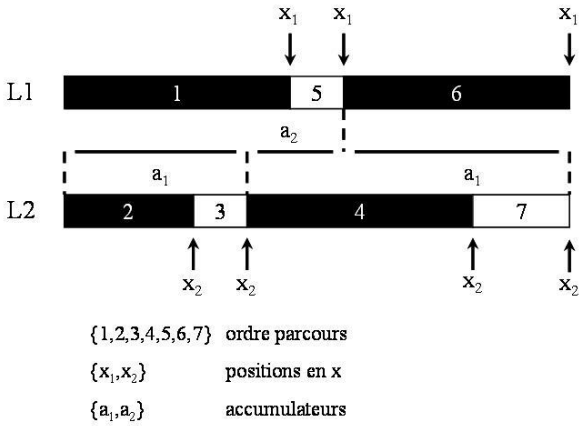


FIG. 8 – Parcours des plages pour la comparaison d'images

Notre algorithme de comparaison d'images procède donc en deux étapes successives de recalage puis de calcul de distance. Il utilise pour cela les images encodées et préalablement stockées. Il parcourt ces images encodées à différentes reprises afin de les comparer. Tout d'abord, deux parcours sont nécessaires par image pour l'extraction des histogrammes verticaux et horizontaux. Cependant, ces histogrammes étant des méta-données naturelles de la représentation à base de plages nous les avons calculés et stockés préalablement à l'encodage. Nous utilisons donc seulement par la suite deux autres parcours pour le calcul de la distance. Ces parcours sont effectués à la fois sur les images cible et requête. Nous présentons dans la section suivante quelques résultats en temps de calcul et précision de recherche de cet algorithme dans notre section expérimentations et résultats.

3 Expérimentations et résultats

Notre système est en cours de développement, dans cette section nous présentons nos premières expérimentations et résultats. Pour ce faire, nous avons testé notre système sur une base d'image de lettrine des BVH[9]. Cette base se compose de 714 images numérisées à 300 ppi. Ces images sont en niveaux de gris non compressé au format Tiff. La taille totale[10] de notre base est de 489.6 Mo. Nous présentons nos résultats sur cette base en ce qui concerne trois aspects : calcul de compacité, temps de traitement et cas d'usage de requête.

Dans une première étape nous avons procédé à l'encodage des images de notre base en vue d'extraire des statistiques sur leur compression et compacité. Nous avons obtenu des taux moyens de $\overline{t_c} = 0.143$ et $\overline{c} = 1.94$. Ces taux indiquent une réduction moyenne d'environ 85 % des pixels en plages soit 7 fois moins de données à traiter. Ils ont été calculés qu'à partir des encodages en plages horizontales des images utilisés par notre étape de comparaison. La Figure 9 donne quelques exemples d'image de lettrine avec leur taux de compression (t_c) associés.

[9]Bibliothèques Virtuelles Humanistes : http ://www.bvh.univ-tours.fr/

[10]Taille des rasters sans prise en compte des entêtes de fichiers.

FIG. 9 – Lettrine et taux de compression (t_c) associés

Nous avons évalué dans une seconde étape les temps de traitement de notre système. Nos résultats sont présentés dans le Tableau 1. Notre système a été implémenté en C++ et est piloté par une application de contrôle écrite en Java. Il a été testé sur un ordinateur portable (Pentium 2GHz) utilisant un système Windows XP.

	Temps total (secondes)	**Vitesse** (Mega octets / minute)
Lecture	108.93 s	269.4 Mo/mn
Requête (petite)	255.6 s	115.8 Mo/mn
Requête (grande)	384.78 s	76.2 Mo/mn

TAB. 1 – Temps de traitement

Dans un premier temps nous avons évalué le temps de lecture de nos images : ç-à-d le parcours de l'ensemble des encodages en plages de notre base. Nous avons obtenu un temps de 108.93 s pour la lecture de nos 714 images, soit un temps de lecture d'environ 270 Mo/mn. Dans un deuxième temps nous avons évalué les temps d'exécution de nos requêtes. Ces temps sont spécifiques à chacune des requêtes de par les comparaisons d'histogrammes effectuées. En effet, le nombre de comparaison d'histogrammes dépend des différences de tailles entre images. Nous avons effectué une trentaine de requêtes tests et obtenu des temps de traitement s'étalant de 255.06 s à 384.78 s, soit environ une centaine Mo/mn. Ces temps correspondent aux deux parcours de plages nécessaires par notre algorithme plus les temps liés au processus de recalage.

Enfin nous avons testé à la volée la précision de recherche de notre système. La Figure 10 suivante un exemple de résultat de requête. Au regard de ce type de résultat notre système permet, à première vue[11], une recherche efficace des images de lettrine de même classe de tampon. En effet, comme nous l'avons expliqué dans cet article cette application d'indexation semble élémentaire. Sa difficulté majeure réside dans les problèmes de translation rencontrés entre images de même classe de tampon. Ceux-ci sont cependant, en large partie, paliés par notre processus de recalage. Les difficultés principales rencontrées par notre système concernent le cas des images fortement dégradées et présentant de grandes différences avec leur classe d'appartenance. Ces fortes dégradations peuvent avoir différentes causes : différence de contraste, image mal redressée, lettrine effacée à l'impression ….

[11]Un fichier de vérité terrain est en cours de constitution et permettra d'évaluer plus précisément nos résultats d'indexation.

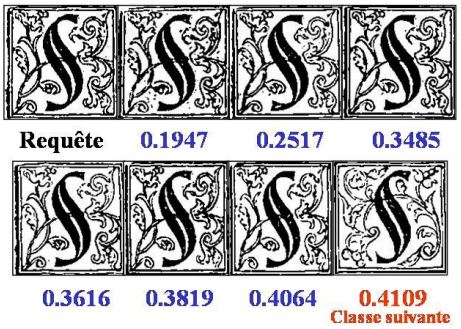


FIG. 10 – Exemple de résultat de requête

4 Conclusion et perspectives

Dans cet article nous avons présenté un système d'indexation d'image de document appliqué aux livres anciens imprimés et plus particulièrement aux lettrines. Le but de ce système est le traitement de larges bases d'image. Pour ce faire, il exploite une représentation à base de plages permettant de réduire la complexité des algorithmes utilisés et d'accélérer les temps de traitement. Le coeur de ce système est un algorithme de comparaison d'images. Ce dernier procède à une étape de recalage avant de calculer une distance entre images. De cette manière, il palie aux problèmes de translation fréquemment rencontrés sur les images de lettrine. Nous avons présenté différents résultats et expérimentations de ce système. Au travers d'eux, nous avons montré que notre système permet un traitement en moyenne 7 fois plus rapide des images de lettrine pour des temps de traitement d'environ une centaine Mo par minute. Nous avons illustré également l'efficacité de notre système au travers d'un exemple de requête.

Les perspectives à ce travail sont de deux natures. Dans un premier temps nous souhaitons utiliser notre système pour l'acquisition de vérité terrain. Le but est alors d'assister automatiquement un utilisateur dans le classement des lettrines et la définition de méta-données. Dans un deuxième temps nous souhaitons étendre notre système pour la recherche locale d'objets graphiques au sein des lettrines. Pour cela, nous travaillons actuellement sur une signature à base de plages inspirée des travaux de [ZHE 01].

Références

[BAI 03] BAIRD H., Digital Libraries and Document Image Analysis, *International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1, 2003, pp. 2-14.

[BAU 05] BAUDRIER E., Comparaison d'images binaires reposant sur une mesure locale des dissimilarités Application à la classification, PhD thesis, Université de la Rochelle, 2005.

[BIA 96] BIANCARDI A., MÉRIGOT A., Connected Component Support for Image Analysis Programs, *International Conference on Pattern Recognition (ICPR)*, vol. 4, 1996, pp. 620-624.

[BIG 96] BIGUN J., BHATTACHARJEE S., MICHEL S., Orientation Radiograms for Image Retrieval : An Alternative to Segmentation, *International Conference on Pattern Recognition (ICPR)*, vol. 3, 1996, pp. 346-350.

[BLA 97] BLASSELLE B., *Histoire du Livre*, Editions Gallimard, 1997.

[BRU 99] BRUNELLI R., MICH O., On the Use of Histograms for Image Retrieval, *International Conference on Multimedia Computing and Systems (ICMC)*, 1999, pp. 143-147.

[BUR 98] BURGE M., KROPATSH W., A Minimal Line Property Preserving Representation of Line Images, *Conference on Structural and Syntactical Pattern Recognition (SSPR)*, vol. 1451 de *Lecture Notes in Computer Science (LNCS)*, 1998, pp. 355-368.

[DOE 98] DOERMANN D., The Indexing and Retrieval of Document Images : a Survey, rapport n° CS-TR-3876, 1998, University of Maryland, USA.

[GES 99] GESU V. D., STAROVOITOV V., Distance Based Function for Image Comparison, *Pattern Recognition Letters (PRL)*, vol. 20, n° 2, 1999, pp. 207-214.

[KIM 88] KIM S., LEE J., KIM J., A New Chain-Coding Algorithm for Binary Images Using Run-Length Codes, *Computer Graphics and Image Processing (CGIP)*, vol. 41, 1988, pp. 114-128.

[KRE 05] KREHER D., STINSON D., Pseudocode : A LATEX Style File for Displaying Algorithms, Department of Mathematical Sciences, Michigan Technological University, Houghton, USA, 2005.

[KUM 91] KUMAR V., *Parallel Architectures and Algorithms for Image Understanding*, Academic Press, 1991.

[PAR 05] PARETI R., VINCENT N., Global Discrimination of Graphics Styles, *Workshop on Graphics Recognition (GREC)*, 2005, pp. 120-128.

[PAV 78] PAVLIDIS T., A Minimum Storage Boundary Tracing Algorithm and Its Application to Automatic Inspection, *Transactions on Systems, Man and Cybernetics (TSMC)*, vol. 8, n° 1, 1978, pp. 66-69.

[RAM 05] RAMEL J., LERICHE S., Segmentation et analyse interactives documents anciens imprimés, *Traitement du Signal (TS)*, vol. 22, n° 3, 2005, pp. 209-222.

[UTT 05] UTTAMA S., HAMMOUD M., GARRIDO C., FRANCO P., OGIER J., Ancient Graphic Documents Characterization, *Workshop on Graphics Recognition (GREC)*, 2005, pp. 97-105.

[VLI 98] VAN VLIET L., VERWER B., A Contour Processing Method for Fast Binary Neighbourhood Operations, *Pattern Recognition Letters (PRL)*, vol. 7, n° 1, 1998, pp. 27-36.

[WEN 98] WENYIN L., DORI D., Performance Evaluation of Graphics Recognition Algorithms : Principles and Applications, *Pattern Recognition (PR)*, vol. 2, 1998, pp. 1180-1182.

[ZHE 01] ZHENG Y., LIU C., DING X., Single Character Type Identification, *Document Recognition and Retrieval*, vol. 4670 de *SPIE Proceedings*, 2001, pp. 49-56.