DjVu: UN SYSTEME DE COMPRESSION D'IMAGES POUR LA DISTRIBUTION RETICULAIRE DE DOCUMENTS NUMERISES.

Léon Bottou, Patrick Haffner, Yann LeCun,
Paul Howard, Pascal Vincent, Bill Riemers
AT&T Labs - Research
100 Schulz Drive, Red Bank, NJ 07701, USA
{leonb,haffner,yann,pgh,vincent,bcr}@research.att.com

Résumé

Nous présentons une technique nouvelle de compression d'images appelée "DjVu". Cette technique est spécialement conçue pour la compression de documents en couleurs numérisés à haute résolution. Un fichier DjVu représentant une page typique d'un magazine en couleurs, numérisée à 300 points par pouce (dpi), requiert entre 40 et 80 KB, ce qui est est 5 à 10 fois meilleur qu'un fichier JPEG offrant une lisibilité similaire. Le compresseur DjVu commence par classer chaque pixel de l'image numérisée comme pixel d'avant-plan (texte, dessins au trait) ou pixel d'arrière-plan (images, photos, texture du papier). Cette classification forme une image bitonale qui est compressée grâce à une technique qui tire parti des similitudes de forme entre les divers caractères composant l'avant-plan. Les images d'avant-plan et d'arrière-plan sont ensuite compressées à l'aide d'un algorithme a base d'ondelettes à résolution reduite. Un algorithme de masquage minimise le nombre de bits utilisés pour coder les pixels d'avant-plan ou d'arrière-plan qui ne sont pas visibles dans l'image finale. Un plug-in pour butineur permet de visualiser très efficacement les images DjVu sur le Web.

1 Introduction

Avec l'utilisation généralisée de l'Internet, avec les coûts décroissants des numériseurs et des disques, l'archivage, la transmission et la manipulation des documents se fait de plus en plus sur ordinateur et de moins en moins sur papier. L'écran de nos ordinateurs est en train de devenir le moyen privilégié de consultation de documents parce qu'il permet un accès immédiat à l'information.

Les technologies de compression d'images bitonales (noir et blanc) de documents ont une longue histoire (cf. [14] et références). Une industrie florissante utilise des techniques standardisées très bien acceptées (Group 3, MMR/Group 4), parfois moins connues (JBIG), ou même franchement obscures (JBIG2).

Curieusement, jusqu'à présent, il n'existait pas de standard permettant de traiter efficacement les documents en *couleur.* Le besoin d'une telle technologie se sont fait plus pressant ces dernières années avec la généralisation de l'Internet. Associée à l'apparition de numériseurs couleur performants, une telle technologie autorise l'existence de bibliothèques digitales offrant en ligne des images de documents anciens ou historiques, des catalogues de vente par correspondance, des gazettes électroniques, des formulaires, des archives gouvernementales ou commerciales, des publications scientifiques, techniques, légales, ou tout autre document actuellement prisonnier du support papier.

Les méthodes usuelles de compression d'images ne permettent pas de telles applications parce qu'elles produisent soit des fichiers trop gros, soit des images difficilement lisibles. L'image couleur d'une page de magazine numérisée à 100 points par pouce (dpi), ou 4 points par mm (p/mm), requiert 100 KB à 200 KB tout en étant difficile à lire. La même image numérisée à 300 dpi (12 p/mm) possède une qualité acceptable mais une taille variant entre 300 KB et 600 KB. Même avec une connexion à large bande passante, ces tailles se traduisent sur le Web par des temps de chargement insupportables pour l'utilisateur final.

D'une part, une résolution élevée (au moins 300 dpi) est nécessaire pour coder les contours de fort contraste afin de préserver la lisibilité du texte et la définition des dessins au trait. D'autre part, une résolution inférieure (de l'ordre de 100 dpi) est suffisante pour préserver l'apparence des images photographiques ou la texture du papier. Il semble donc naturel de séparer l'image initiale en deux plans: l'avant-plan contenant le texte et les dessins au trait, l'arrière-plan contenant les images photographiques et la texture du papier. Cette séparation apporte un second avantage de taille: il est possible de placer l'avant-plan en tête du fichier et de permettre à l'utilisateur final de voir le texte dès que celui ci est disponible, sans attendre le chargement du fichier complet.

Nous pensons que le cahier des charges suivant permet d'offrir à l'utilisateur final une expérience acceptable. Le texte doit apparaître à l'écran après un délai d'au plus quelques secondes. Cela signifie que l'avant-plan doit tenir en moins de 40 KB (en supposant une vitesse de transmission de 56Kb/s). Le reste de l'image doit ensuite être affiché progressivement, en améliorant la qualité au fur et à mesure de la transmission. La taille totale du fichier ne doit pas excéder 100 Kb afin de limiter le temps de transmission total et les contraintes de stockage de données.

Les images de grande tailles posent également problème pendant la décompression. Une page de magazine fait environ 3300x2500 pixels et occupe 25 MB de mémoire sous forme non compressée. De telles images peuvent être difficiles à manipuler sur l'ordinateur personnel de l'utilisateur moyen. Le logiciel de visualisation doit donc maintenir en mémoire l'image complète sous forme compressée, et décompresser à la demande les pixels visibles sur l'écran à un instant donné.

Cet article décrit un technique de compression d'images de documents appelée DjVu (qui se prononce "déjà vu" en Français comme en Anglais) qui apporte une solution à tous les problèmes cités ci-dessus. Avec DjVu, une pages numérisée à 300 dpi en couleur (25 MB sous forme non compressée) peut être compressée en un fichier de 30 KB à 80 KB avec une excellente qualité. Cette taille est comparable à la taille moyenne d'une page Web (HTML et images) qui est d'environ 50 à 100 KB. Un plug-in pour butineur ("browser plug-in") permet la visualisation progressive des pages DjVu au fur et à mesure de la transmission. L'utilisateur peut sélectionner le grossissement et déplacer librement l'image dans la fenêtre de visualisation. En interne, le logiciel de visualisation ne génère jamais l'image complète de 25 MB, mais conserve l'image en mémoire sous une forme partiellement décodée qui occupe environ 1.5 MB, et reconstruit à la demande les pixels qui sont affichés à l'écran.

La première section explique les idées générales de compression et décompression DjVu. La sections 3 décrivent le comportement de l'algorithme de segmentation avant-plan/arrière-plan. Les sections suivantes donnent quelques résultats, et la dernière section détaille certaines caractéristiques importantes qui contribuent à la performance de DjVu.

2 Le Système DjVu

Une des idées essentielles de DjVu consiste à décomposer l'image en un avant-plan contenant les objets délimités par des contours fortement contrastés (comme le texte et les dessins au trait), et un arrière-plan contenant le reste de l'image (comme les photographies ou la texture du papier). Différentes techniques peuvent ensuite être utilisées pour compresser chaque plan. Les méthodes usuelles de compression d'images sont en effet conçues, soit pour compresser des images naturelles contenant peu de contours (JPEG), soit des images de documents noir et blancs, entièrement composées de contours à fort contraste (Group 3, MMR/Group 4, et JBIG). Le système DjVu intègre deux nouvelles techniques appelées JB2 et IW44 pour coder les divers plans.

L'algorithme de séparation avant-plan/arrière-plan produit trois sorties: un *masque* bitonal de haute résolution (en général 300 dpi), une image en couleur représentant l'arrière-plan avec une résolution moindre (en général 100 dpi), et une structure qui code la couleur des objets d'avant-plan. Si un pixel du masque est zéro, le pixel correspondant de l'image reconstruite prend la couleur extraite de l'image d'arriere-plan à l'endroit correspondant, sinon il prend une couleur définie par l'avant-plan.

Le masque est codé avec un nouvel algorithme de compression d'image bitonale, nommé JB2, qui est en fait un descendant de la proposition originale d'AT&T pour le standard JBIG-2. L'algorithme consiste à identifier les formes individuelles qui composent l'image (généralement des composantes connexes, comme des caractères) et à les grouper en catégories de formes similaires [2]. Les formes représentatives de chaque classe sont codées grâce à une méthode similaire au standard JBIG. Chaque pixel de la forme est codé à l'aide d'une technique de codage arithmétique adaptatif appelé le ZP-Coder [4]. Le codeur arithmétique utilise un *contexte* formé par les pixels voisins déjà transmis pour prédire la valeur du pixel à coder, et pour transmettre sa valeur en utilisant un nombre de bits optimal (à quelques pourcent de la limite de Shannon). Les formes autres que le prototype appartenant à une classe sont codées en utilisant un contexte augmenté qui incorpore la valeur des pixels de la forme prototype [6]. Cette stratégie maximise la réduction du nombre de bits utilisés puisque la majorité des pixels sont similaires aux pixels de la forme prototype. Comme il est rarement nécessaire de reconstruire exactement l'image originale, les taux de compression peuvent être significativement augmentés en remplaçant ces formes par la forme prototype dès que les différences sont suffisamment faibles pour être imperceptibles. Les positions auxquelles les formes doivent être affichée dans l'image sont également codées arithmétiquement. Le codeur arithmétique adaptatif ZP-Coder est très rapide, et fournit des taux de compression moyens situés à environ 5% de la limite théorique de Shannon [4]. Dans le cas où le document comporte plusieurs pages, il est avantageux de construire un dictionnaire de formes prototypes partagé entre les pages, ainsi qu'un dictionnaire propre à chaque page qui contient les formes n'apparaissant que dans la page considérée. JB2 utilise une méthode très rapide et incrémentale pour ce faire. Des comparaisons avec les méthodes standard sont données dans la section 5

Pour l'image d'arrière-plan, DjVu utilise un algorithme de compression par ondelettes, nommé IW44, qui présente plusieurs avantages importants vis-à-vis des autres méthodes de compression d'images naturelles. Premièrement, IW44 utilise une transformée d'ondelettes basée sur sur la méthode du "lifting" qui est très rapide [12]. Deuxièmement cette transformée est implémentée de manière à ne pas requérir de multiplication, en s'appuyant exclusivement sur des additions et des décalages. Cela réduit beaucoup le temps de calcul. Troisièmement, la structure de données interne des images IW44 permet d'améliorer progressivement les coefficients d'ondelettes au fur et à mesure de leur réception, tout en utilisant une quantité de mémoire proportionelle au nombre de coefficients non nuls (et non au nombre de pixels). Troisièmement, cette structure de données permet à tout moment, pendant et après la transmission, de reconstruire un segment quelconque de l'image, avec une résolution quelconque. Cela permet de ne reconstruire que les parties de l'image affichées à l'écran. Finalement, la technique de *masquage* par projections successives [5], permet d'éviter de dépenser des bits pour coder les régions de l'arrière-plan qui sont invisibles car recouvertes par des objets situés à l'avant-plan. L'algorithme IW44 utilise également le ZP-Coder pour le codage final des données.

La couleur des objets d'avant-plan peut être codée de deux méthodes différentes. La première méthode consiste à identifier une couleur unique pour chaque forme qui apparaît dans le masque, et à le coder l'aide du ZP-Coder. Ceci fournit un codage très compact mais exige une segmentation presque parfaite, et ne fonctionne que si les composantes d'avant-plan sont de couleur uniforme. La seconde méthode consiste à utiliser IW44 pour coder une image de très basse résolution (en général 25 dpi ou 1 p/mm). L'interprétation de cette image d'avant-plan est exactement symétrique à celle de l'image d'arrière-plan: le masque sert essentiellement "d'Alpha channel" pour mixer les images d'arrière-plan et d'avant-plan. Cette seconde méthode est similaire aux suggestions du standard MRC/T.44 [9]. La section 6 quantifie les gains résultant de l'usage des algorithmes JB2 et IW44 au lieu des algorithmes traditionnels MMR/Group 4 et JPEG préconisés par le standard.

3 L'Algorithme de Segmentation

La première phase de la segmentation avant-plan/arrière-plan repose sur un Champ de Markov causal bidimensionnel, dont les états représentent l'avant-plan et l'arrière-plan. Chaque état décrit localement la distribution des pixels d'avant-plan ou d'arrière-plan à l'aide d'une gaussienne unique paramétrée par la couleur moyenne des pixels et leur variances. Les probabilités de transition du Champ de Markov sont des constantes dont le choix est discuté dans la dernière section.

Cette phase initiale effectue en fait une *sur-segmentation*, afin de collecter tous les objets qui pourraient mériter d'être codés en avant-plan. En conséquence, certaines parties très contrastées de photographies, ou certains tramages d'imprimerie peuvent très bien être catégorisés incorrectement comme objets d'avant-plan. Une succession de filtres heuristiques sont ensuite appliqués pour rejeter les candidats catégorisés par erreur comme objets d'avant-plan.

Les critères de décision implémentés par le filtre principal repose sur le principe de Longueur Minimale de Description, ou "Minimum Description Length (MDL)" [8]. Ce principe permet de minimiser le nombre de paramètres heuristiques à optimiser manuellement en comparant des centaines d'images de type divers. Il suffit en effet de savoir s'il est préférable de coder chaque candidat comme objet d'avant-plan ou d'arrière-plan. Le coût de codage associé à chaque possibilité est obtenue en additionnant le coût de codage des paramètres d'un modèle génératif de l'image compressée et le coût de codage des résidus permettant de reconstruire l'image initiale. La possibilité qui présente le coût de codage minimum est retenue. Coder un candidat dans l'arrière-plan requiert seulement un modèle de l'image d'arrière plan. Coder un candidat comme objet d'avant-plan requiert un modèle de l'image d'avant-plan, un modèle de l'image d'arrière-plan située sous l'objet d'avant-plan, et un modèle de l'image masque.

Le modèle retenu pour l'image d'arrière-plan suppose simplement que la couleur de chaque pixel est une moyenne des pixels d'arrière-plan les plus proches trouvés au dessus et à gauche du pixel. Le modèle retenu pour l'image d'avant plan suppose une couleur uniforme pour l'objet. Le coût de codage associé au modèle du masque est évalué de manière heuristique à partir de la longueur du périmètre de l'objet candidat. Le modèle de l'arrière-plan autorise les variations continues caractéristiques des zones que l'on souhaite conserver en arrière-plan. Le modèle d'avant-plan favorise les objets bien délimités de couleur uniforme.

Bien que l'on puisse souhaiter des modèles reproduisant plus finement les caractéristiques des algorithmes de codage de DjVu, ces trois modèles simples offrent un bon compromis entre temps d'exécution et qualité de la segmentation. La segmentation complète d'une image couleur à 300 dpi (une page de magazine par exemple) s'effectue en 3 secondes environ avec un processeur Pentium 400Mhz.

Puisqu'il n'est pas réellement possible d'estimer automatiquement les probabilités de transition entre les états avant-plan et arrière-plan du Modèle de Markov utilisé dans la première phase de l'algorithme de segmentation, il est nécessaire de choisir une valeur de façon heuristique. La proportion de pixels catégorisés en avant-plan diminue avec ce paramètre. Lorsqu'il devient nul, tous les pixels sont catégorisés en arrière-plan.

Une valeur trop élevée du paramètre sur-segmente le document. La taille du fichier DjVu est alors dominée par le masque qui contient des éléments de photographies, des motifs de tramage, ou bien du bruit. Une valeur trop faible sous-segmente le document. Le texte est alors codé comme élément d'arrière-plan. Des expériences systématiques indiquent que la valeur du paramètre qui produit la meilleure segmentation pour un obervateur humain est aussi celle qui produit le meilleur taux de compression. Cette propriété intéressante et facilite grandement la mise au point du segmenteur.

4 Résultats: IW44 pour les images naturelles

La performance de IW44 est typique des algorithmes de compression d'images par ondelettes, mais a été optimisée pour accélérer la vitesse de décodage et minimiser l'utilisation mémoire, plutôt que pour maximiser la compression. L'espace occupé par une image IW44 est en général de 30% à 50% inférieur à celui d'une image JPEG de même rapport signal/bruit. Les gains les plus spectaculaires sont obtenus pour les taux de compression élevés. Un avantage significatif de IW44 par rapport à JPEG est la progressivité.

La table 1 présente les rapports Signal/Bruit obtenus avec IW44 sur deux images de test standard (Lena et Goldhill), et les compare à JPEG (commande Unix "cjpeg" du Independent JPEG Group), et à plusieurs méthodes à base d'ondelettes décrites dans la littérature: EZW [11], SPIHT [10], et EBCOT [13].

Sur l'image Lena, IW44 surpasse JPEG (avec tables optimisées) d'un peu moins de 2 dB. IW44 surpasse également EZW, mais est légèrement moins bon que SPIHT et EBCOT. Ceci vient du fait que l'architecture d'IW44 et les filtres d'ondelette utilisés sont optimisés de manière à accélérer la vitesse de decodage. SPIHT et EBCOT utilisent le filtre 9-7 d'Antonini et coll. [1] calculé en virgule flottante. IW44 utilise le filtre 4/4 de Deslauriers-Dubuc calculé en virgule fixe sur 16 bits, et sans utiliser de multiplications (uniquement en combinant additions et décalages). L'utilisation d'un filtre 9-7 dans IW44 améliorerait sensiblement les résultats d'IW44, mais rendrait le décodage beaucoup trop lent pour les

Image	bpp	IW44	JPEG	EZW	SPIHT	EBCOT
Lena	0.25	33.62	31.67	33.17	34.11	34.28
Lena	0.50	36.61	34.84	36.28	37.21	37.43
Lena	1.00	39.67	37.94	39.55	40.46	40.61
Goldhill	0.25	30.25	29.23	n/a	30.56	n/a
Goldhill	0.50	32.56	31.03	n/a	33.12	n/a

Table 1: *Rapport signal/bruit (en dB) obtenu sur des images test usuelles avec IW44, JPEG, et plusieurs méthodes à base d'ondelettes: EZW, SPIHT, et EBCOT. Le filtre utilisé par IW44 est très rapide, mais sous optimal pour le taux de compression. Les ondelettes utilisées sont les suivantes: IW44: 5 niveaux, avec filtre 4/4 de Deslauriers-Dubuc; JPEG: Independent JPEG Group avec tables de Huffman optimisées; EZW: 6 niveaux, ordre 9 QMF, codage "Embedded Zero Tree"; SPIHT: 5 niveaux, filtre 9-7, avec codeur arithmétique; EBCOT: 5 niveaux, filtre 9-7*

applications considérées. En outre, IW44 est le résultat de compromis qui limitent la complexité des contexte du codage arithmétique de manière à maximiser la vitesse du decodage progressif, et à minimiser l'utilisation mémoire. à titre de comparaison, le futur standard JPEG-2000 est basé sur une version simplifiée d'EBCOT.

5 Résultats: JB2 pour les images bitonales

Un test indépendant de compression "sans pertes" d'images bitonales [7] effectué sur une large collection de documents a mesuré un taux de compression moyen de 26.5 pour JB2 v2.0. Cela peut être comparé au taux de 13.5 obtenu par MMR/Group 4 et 19.4 obtenu par JBIG. Les gains sont plus importants pour les images contenant beaucoup de texte, et moindres pour les images contenant plutôt des dessins, ou des photographies tramées.

En mode "avec pertes", les résultats sont beaucoup plus intéressants. La notion de "perte" est ici relative, car l'effet des pertes est principalement d'uniformiser et de nettoyer la forme des caractères.

La table 3 présente les résultats de DjVu bitonal et les compare à MMR/CCITT–GroupIV. Les taux de compression obtenus avec DjVu v3.0 en mode multipage avec dictionnaire partagé sont de 4.5 à 10 fois supérieurs à MMR/G4. La taille moyenne d'une page est de 5 à 11 kilo-octets, en fonction du document.

Une comparaison avec avec PDF s'impose. Le format PDF lorsqu'il est utilisé pour des documents bitonaux numérisés donne des tailles similaires à MMR/G4 car PDF ne fait qu'encapsuler le document MMR/G4 dans un format PDF.

Le document Nips10 est particulièrement intéressant car représentative d'une publication scientifique typique. Avec ses 1090 pages, le fichier TIFF/G4 (ou l'équivalent PDF) pèse environ 76MB. En DjVu, il est réduit à environ 13MB. La taille moyenne par page est d'environ 10KB, ce qui permet d'envisager le stockage d'environ 60000 pages sur un CD-ROM (environ 50 volumes de ce type).

Le document Snowbird est différent des autres en ce qu'il n'a pas été numérisé à partir de papier, mais directement converti à partir de fichiers PostScript créés à l'aide de logiciels de traitement de texte tels que LaTeX ou MS-Word. Le fichier PostScript original occupe environ 10.7 MB. Compressé a l'aide de "gzip", il occupe encore 3.1 MB. Converti en TIFF/G4 à 300 dpi a l'aide de "ghostscript", il occupe 7.4 MB. Compressé avec DjVu avec

Compression	Aucune	GIF	JPEG	DjVu
Hobby p15	24715	1562	469	58
dict. médical	16411	1395	536	110
time zone	9174	576	249	36
cookbook	12128	1000	280	52
Hobby p17	23923	1595	482	52
U.S. Constit.	31288	2538	604	134
Hobby p2	23923	1213	383	68
ATT Olympic	23946	955	285	41

Table 2: *Taille des fichiers (en KB) obtenus à partir de huit documents numérisés à 300 dpi en utilisant les algorithmes suivants: pas de compression, GIF à résolution 150 dpi, JPEG à résolution 300 dpi (IJG-JPEG, qualité 20%), et DjVu avec un masque à 300 dpi et un arrière-plan à 100 dpi. La qualité visuelle de ces images peut être comparée dans la section exemple du site Internet* `http://www.djvu.com`

dictionnaire partagé, il se réduit à 728KB. C'est un gain de 10 par rapport a TIFF/G4, de 14 par rapport au PostScript, et de 4.25 par rapport au PostScript gzippé.

6 Résultats: DjVu pour les documents en couleur

Sur un document en couleur, le système DjVu avec séparation avant-plan/arrière-plan atteint des taux de compression entre 300:1 et 1000:1. La table 2 montre les tailles de fichiers DjVu correspondant à quelques documents numérisés en couleurs à 300 dpi. Les documents ordinaires (magazines, catalogues) occupent entre 30 KB et 80 KB. Certains documents occupent de 80 KB à 140 KB parce que leur taille physique est supérieure ou parce qu'ils contiennent des photographies détaillées. Ces tailles de fichiers sont typiquement de 5 à 10 fois inférieures à celles obtenues avec JPEG à lisibilité équivalente.

A partir d'une image de document à 300 dpi et 24 bits/pixel, le segmenteur produit trois sous-images: le masque bitonal (1 bit/pixel) est 24 fois plus petit que l'image originale; l'image d'arrière-plan (100 dpi) est $3 \times 3 = 9$ fois plus petite; l'image d'avant-plan (25 dpi) est $12 \times 12 = 144$ fois plus petite. En additionnant les tailles non compressées de ces sous-images, on obtient déjà un taux de compression de 6.25:1 par rapport à l'image originale. Nous proposons de comparer les résultats obtenus avec DjVu (qui utilise JB2 pour le masque et IW44 pour les autres plans), et avec les techniques standards qui utilisent MMR/Group 4 pour le masque et JPEG pour l'avant-plan et l'arrière-plan. La figure 1 montre les gains apportés par JB2 et IW44 mesurés sur un corpus composé de 70 images variées. Ces images contiennent des textes imprimés, des photographies, de fort tramages, de l'écriture manuscrite, des symboles mathématiques, des dessins à main levée, et des partitions musicales.

Les compressions JPEG et IW44 ont été ajustées de sorte à produire des images compressées avec une même rapport signal/bruit mesuré sur les *pixels visibles* de l'image. Le taux de compression pour les images d'arrière-plan passe de 59:1 pour JPEG à 103:1 pour IW44. Globalement, le taux de compression moyen obtenu avec DjVu sur ces 70 images atteint 399:1. C'est à peu près deux fois meilleur que les taux de compression que l'on aurait obtenu si l'on avait utilisé les algorithmes usuels JPEG et MMR/Group 4 dans un contexte MRC/T.44.

De grandes quantités d'exemples peuvent être consultés en ligne sur la bibliothèque dig-

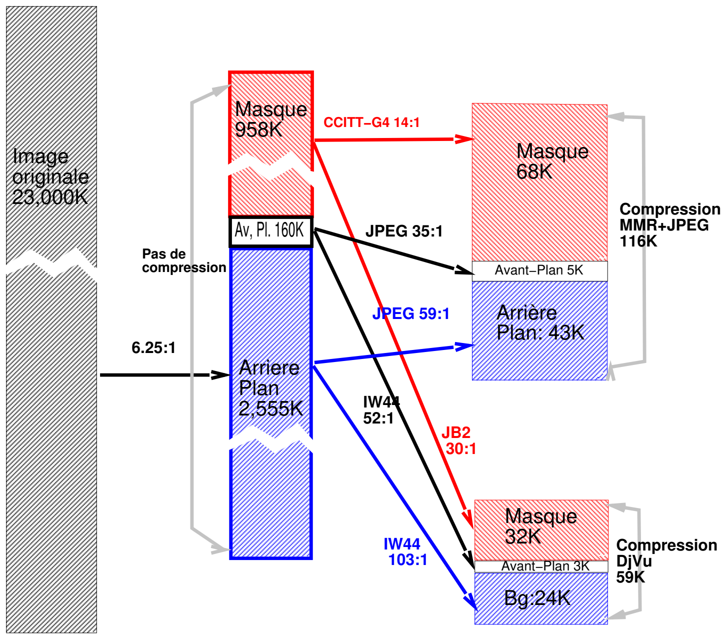

Figure 1: *Comparaison, pour une segmentation donnée, de la taille moyenne des trois sous-images (masque, arrière-plan, avant-plan) compressées en utilisant les techniques suivantes: (i) pas de compression, (ii) compression standard avec MMR/Group 4 et JPEG, (iii) compression DjVu avec JB2 et IW44. En supposant une image initiale de 23 MB, chaque bloc indique la taille moyenne du plan correspondant.*

itale DjVu à l'adresse `http://www.djvu.com`. D'autres exemples sont disponibles auprès de nombreux utilisateurs commerciaux et non-commerciaux de DjVu sur l'Internet.

7 Conclusion

DjVu est une technique de compression d'images qui établit une passerelle entre le monde du papier et le monde des bits. Elle permet de publier sur l'Internet des documents numérisés à haute résolution sans pour autant requérir des temps de chargement qui éprouveraient la patience de l'utilisateur, et sans requérir des ressources informatiques qui éprouveraient son portefeuille.

Des logiciels de compression, de décompression, et de visualisation DjVu sont disponibles sur le site `http://www.djvu.com` et sont gratuits pour tout usage non commercial. Le code source de la bibliothèque de référence est disponibles à la même adresse avec une licence de type "Logiciel Libre". La spécification du format de fichier et également disponible. Le plug-in DjVu pour butineur Web est disponible pour Linux, Windows, et Mac ainsi que pour plusieurs versions d'Unix. Le site contient aussi une bibliothèque virtuelle offrant quelques milliers de pages numérisées d'origines diverses.

DjVu est d'ores et déjà utilisé par un grand nombre d'utilisateurs commerciaux et non-commerciaux sur l'Internet. Entre autres, Bell and Howell (anciennement University Microfilm Inc.) propose un service basé sur DjVu nommé "Early English Books Online" qui regroupe la totalité des 96000 livres publiés en Anglais entre l'invention de l'imprimerie et l'année 1700 (22 millions de pages) numérisés à partir de microfilms. Certains groupements, tels l'Acoustic Society of America, offrent les archives de leurs publications scientifiques au format DjVu sur CD-ROM ou sur l'Internet. En outre, de nombreuses universités et bibliothèques commencent à proposer leur collections au format DjVu.

DjVu est non seulement le plus performant des systèmes disponibles aujourd'hui pour la compression et la distribution de documents numérisés bitonaux et couleurs, mais c'est aussi le seul qui soit un standard ouvert, implémenté, et qui ait fait ses preuves dans des applications à grande échelle.

References

[1] M. Antonini, M. Barlaud, P. Mathieu, and Daubechies I. Image coding using wavelet transform. *IEEE Transactions on Image Processing*, 1:205–220, 1992.

[2] R. N. Ascher and G. Nagy. A means for achieving a high degree of compaction on scan-digitized printed text. *IEEE Trans. Comput.*, C-23:1174–1179, November 1974.

[3] L. Bottou, P. Haffner, P. G. Howard, P. Simard, Y. Bengio, and Y. LeCun. High quality document image compression with djvu. *Journal of Electronic Imaging*, 7(3):410–428, 1998.

[4] L. Bottou, P. G. Howard, and Y. Bengio. The Z-coder adaptive binary coder. In *Proceedings of IEEE Data Compression Conference*, pages 13–22, Snowbird, UT, 1998.

[5] L. Bottou and S. Pigeon. Lossy compression of partially masked still images. In *Proceedings of IEEE Data Compression Conference*, Snowbird, UT, March-April 1998.

[6] P. G. Howard. Text image compression using soft pattern matching. *Computer Journal*, 40(2/3):146–156, 1997.

[7] Stuart Inglis. *Lossless Document Image Compression.* PhD thesis, University of Waikato, March 1999.

[8] W. Niblack J. Sheinvald, B. Dom and D. Steele. Unsupervised image segmentation using the minimum description length principle. In *Proceedings of ICPR 92*, 1992.

[9] MRC. Mixed rater content (MRC) mode. ITU Recommendation T.44, 1997.

[10] A. Said and W. A. Pearlman. A new, fast, and efficient image codec based on set partitioning in hierarchical trees. *IEEE Transactions on Circuits and Systems for Video Technology*, 6(3):243–250, June 1996.

[11] J. M. Shapiro. Embedded image coding using zerotrees of wavelets coefficients. *IEEE Transactions on Signal Processing*, 41:3445–3462, December 1993.

[12] W. Sweldens. The lifting scheme: A custom-design construction of biorthogonal wavelets. *Journal of Applied Computing and Harmonic Analysis*, 3:186–200, 1996.

[13] D. Taubman. High performance scalable image compression with ebcot. *IEEE Transactions on Image Processing*, 1999. Preprint March 1999. To appear.

[14] I. H. Witten, A. Moffat, and T. C. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images.* Van Nostrand Reinhold, New York, 1994.

Tailles totales des fichiers
(et tailles moyennes par page)

Document	pages	brut (KB)	G4 (octets)	JB2 v2.0 (octets)	JB2-3(1) (octets)	JB2-3(10) (octets)	JB2-3(inf) (octets)
p5960411	19	18338 (965)	880372 (46335)	154923 (8153)	129912 (6837)	114610 (6032)	110148 (5797)
p5960412	167	161181 (965)	3819436 (22870)	1129964 (6766)	1056946 (6329)	842448 (5044)	730805 (4376)
p5960414	12	11582 (965)	769390 (64115)	109700 (9141)	91278 (7606)	79581 (6631)	77855 (6487)
p5960413	17	16408 (965)	1585926 (93289)	238924 (14054)	201166 (11833)	182141 (10714)	180435 (10613)
p5960415	18	17373 (965)	955482 (53082)	147030 (8168)	112629 (6257)	92793 (5155)	90028 (5001)
p5960416	53	51153 (965)	2750910 (51903)	559900 (10564)	469527 (8859)	410680 (7748)	393792 (7430)
Snowbird	133	137067 (1030)	7406482 (55687)	1099934 (8270)	988321 (7430)	869625 (6538)	728196 (5475)
Nips10	1090	1276971 (1171)	76443591 (70131)	n/a (n(a)	15838908 (14531)	12692602 (11644)	11013261 (10103)

Taux de Compression (et taux relatifs a CCITT-GroupIV)

Document	pages	brut (KB)	G4 (ratio)	JB2 v2.0 (ratio)	JB2-3(1) (ratio)	JB2-3(10) (ratio)	JB2-3(inf) (ratio)
p5960411	19	18338	20.8	118.4 (5.7)	141.2 (6.8)	160.0 (7.7)	166.5 (8.0)
p5960412	167	161181	42.2	142.6 (3.4)	152.5 (3.6)	191.3 (4.5)	220.6 (5.2)
p5960414	12	11582	15.1	105.6 (7.0)	126.9 (8.4)	145.5 (9.7)	148.8 (9.9)
p5960413	17	16408	10.3	68.7 (6.6)	81.6 (7.9)	90.1 (8.7)	90.9 (8.8)
p5960415	18	17373	18.2	118.2 (6.5)	154.2 (8.5)	187.2 (10.3)	193.0 (10.6)
p5960416	53	51153	18.6	91.4 (4.9)	108.9 (5.9)	124.6 (6.7)	129.9 (7.0)
Snowbird	133	137067	18.5	124.6 (6.7)	138.7 (7.5)	157.6 (8.5)	188.2 (10.2)
Nips10	1090	1276971	16.7	n/a (n/a)	80.6 (4.8)	100.6 (6.0)	115.9 (6.9)

Table 3: *comparaison entre MMR/G4, et JB2 (DjVu bitonal). Les six premiers documents sont des brevets numérisés à 300 dpi directement obtenus du site Internet du Bureau Américain des Brevets, le document Nips10 est le volume 10 des actes de la conférence NIPS numérisé à 400 dpi, le document Snowbird est en ensemble de résumés soumis à une conférence en format PostScript (donc pas de numérisation). Les colonnes donnent les informations suivantes. "pages": nombre de pages du document; "brut": taille de l'original à 1 bit par pixel (en kilo-octets); "MMR/G4": compression en MMR/CCITT-GroupIV; "JB2 v2.0": DjVu bitonal de précédente génération; "JB2 v3.0(1)": DjVu bitonal de nouvelle génération en compressant chaque page isolément; "JB2-3(10)": nouveau DjVu bitonal avec un dictionnaire de formes partagé toutes les 10 pages; "JB2-3(inf)": nouveau DjVu bitonal avec un seul dictionnaire de formes partagé par toutes les pages du document.*