Adaptation de contenu MPEG-4 BIFS suivant la norme MPEG-21

Cyril Concolato, Jean-Claude Dufourd

Ecole Nationale Supérieure des Télécommunications
46 rue Barrault 75013 Paris
{concolato, dufourd}@enst.fr

Résumé. Le groupe MPEG a récemment standardisé la norme MPEG-21 qui a pour but, entre autres, de fournir des outils inter-opérables pour l'adaptation de contenu multimédia. Dans cet article, nous nous intéressons à l'adaptation de contenu vectoriel 2D encodé au format MPEG-4 BIFS. Nous proposons de nouvelles méthodes d'encodage pour produire des flux BIFS scalables. Nous présentons également une méthode d'intégration de ces flux dans le cadre de MPEG-21 à l'aide du langage gBSD. Enfin, nous présentons les résultats d'une méthode d'encodage scalable sur un contenu de type dessin animé.

1 Introduction

Avec la croissance des réseaux de téléphonie mobile et du réseau Internet, le processus de convergence des technologies de l'information et du multimédia affronte de nouveaux problèmes. D'une part, un contenu multimédia doit être diffusable sur différents types de réseaux pour lesquels la bande passante disponible est très différente (e.g. GSM, ADSL, LAN). D'autre part, ce même contenu doit être visualisable sur une large gamme de terminaux multimédia variant du téléphone mobile à l'ordinateur en passant par les décodeurs de télévision numérique. Or, il apparaît difficile voire impossible de prévoir tous les scénarios d'utilisation d'un contenu multimédia. C'est pourquoi, la solution qui consiste à encoder plusieurs versions du même contenu pour cibler des scénarios différents tend à disparaître au profit de techniques d'encodage qui permettent une adaptation à la volée du contenu. C'est dans ce contexte que le groupe de normalisation MPEG développe la norme MPEG-21 [1]. Cette norme définit un cadre général d'adaptation de contenus ainsi qu'un certain nombre d'outils pour décrire ces contenus et les contraintes à respecter pour effectuer l'adaptation.

Dans ce document, nous présentons les travaux qui ont été réalisés dans le cadre du projet européen ISIS [2]. Nous nous intéressons à un type de média particulier que constituent les flux de contenu graphique vectoriel 2D. Le reste du document se décompose comme suit. Dans une première partie, nous décrivons l'architecture d'un système multimédia de type MPEG-21 permettant l'adaptation de contenu. Dans une seconde partie, nous présentons le type de média auquel nous nous intéressons ainsi que les techniques d'adaptation spécifiques. Dans une troisième partie, nous

décrivons comment nous avons intégré ce type de média dans le schéma général MPEG-21. Dans la quatrième partie, nous présentons les résultats de nos travaux. Enfin, nous concluons ce document en présentant les perspectives de travaux futurs.

2 Description du modèle d'adaptation MPEG-21

La notion fondamentale décrite par la norme MPEG-21 est la notion de « Digital Item » ou DI. Un DI est une œuvre à laquelle on peut attacher un certain nombre de descriptions. Chaque type de description fait l'objet d'une partie de la norme. Parmi ces parties, on distingue celles qui spécifient un format de description de déclaration de l'environnement d'utilisation de ce DI (« Digital Item Declaration » ou DID), un format de description des techniques d'adaptation associées (« Digital Item Adaptation » ou DIA), un format de description des droits et licences associés (« Rights Expression Language » ou REL). Dans cette section, nous présentons le schéma d'adaptation de contenu décrit par la partie MPEG-21 DIA.

Le modèle d'adaptation décrit par MPEG-21 DIA est composé d'une unité d'adaptation qui détermine l'adaptation optimale d'un DI en fonction de ses descriptions DID et DIA. Il s'applique à un type de média « scalable », c'est-à-dire un media organisé de sorte qu'il soit possible de visualiser d'abord une partie de faible qualité, et ensuite de progressivement améliorer la qualité en traitant des données additionnelles.

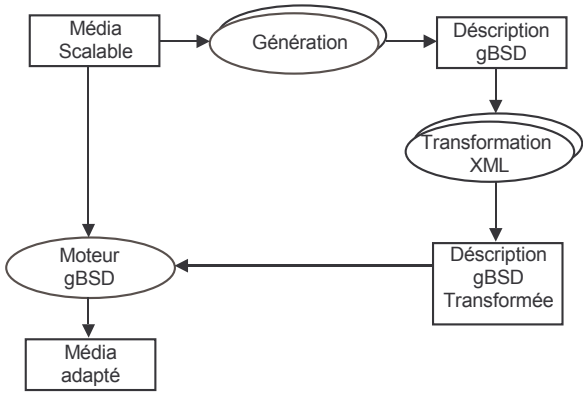

Figure 1. Adaptation d'un flux média utilisant gBSD

Le procédé d'adaptation décrit par la norme MPEG-21 DIA peut-être media-agnostique grâce à un langage de description des médias appelé gBSD. gBSD est un langage XML [8] qui permet de décrire de manière textuelle la structure d'un flux média binaire. Cette description peut-être utilisée pour produire une version modifiée, plus compacte du flux. Dans le modèle DIA, représenté dans la figure 1, le média à adapter est décrit par un document gBSD. Ce document est transformé à l'aide d'un mécanisme de transformation XML, par exemple XSLT [9]. Enfin, un moteur d'interprétation gBSD crée le média adapté à partir du média initial et de la

description gBSD transformée. Ce moteur est totalement indépendant du type de média à transformer puisqu'il s'appuie sur la description gBSD.

3 Description des contenus et techniques d'encodage

3.1 Description des contenus et du format de codage

Les contenus auxquels nous nous intéressons sont des contenus représentant des dessins animés 2D de type classique, tels qu'on peut en voir sur Internet [3] [4]. Ces dessins sont généralement composés d'un fond uniforme sur lequel se déplacent des personnages ou objets. Ces objets sont représentés par des zones de couleurs uniformes ou en dégradé, bordées par un contour d'une certaine épaisseur et d'une certaine couleur [5][6]. Ces dessins animés sont encodés au format MPEG-4 BIFS [7].

BIFS est un format binaire de description de scènes multimédia qui permet également de décrire des objets vectoriels 2D et/ou 3D. Un flux BIFS se décompose en commandes d'ajout, d'insertion ou de suppression d'éléments dans la scène. Nous avons réalisé un encodeur BIFS qui permet d'encoder un contenu vectoriel 2D de manière « scalable ». Le principe est le suivant. Le flux BIFS représentant le dessin animé est encodé en couches. La couche de base du flux contient les commandes BIFS permettant de visualiser la version du contenu de faible qualité. Les couches d'améliorations contiennent des commandes BIFS d'ajout ou d'insertion d'éléments graphiques permettant d'améliorer la qualité. Ces dernières commandes sont des commandes supplémentaires par rapport au contenu initial. Elles apportent la possibilité d'adapter le contenu mais en revanche nécessitent des débits plus importants.

3.2 Techniques d'encodage

Pour créer ces commandes d'amélioration, l'encodeur doit lui-même isoler les éléments graphiques destinés au flux de base de ceux destinés aux flux d'amélioration. Nous avons implémenté deux types de techniques.

3.2.1 Encodage scalable par répartition de points

Le premier type de technique consiste à réduire le nombre de points des éléments graphiques du flux de base et d'insérer par la suite ces éléments dans les flux d'amélioration. Deux techniques séparées de ce type ont été expérimentées : la réduction de points à partir d'un calcul de courbure ou d'un calcul d'aire.

La première consiste à considérer que les points à forte courbure sont plus importants que les points à faible courbure. Ainsi, les premiers sont conservés dans le flux de base alors que les seconds sont affectés aux flux d'amélioration. Le processus se répète ensuite sur les flux d'amélioration. La figure 2 illustre ce procédé. On

remarque que tous les points des différents contours de base ou d'amélioration appartiennent au contour initial. Ce procédé ne crée pas de points supplémentaires ni de points intermédiaires.

La seconde technique de réduction du nombre de points utilise un calcul d'aire. Cette technique traite un contour fermé. Les contours ouverts sont fermés artificiellement pour le traitement. Ensuite, à chaque point P_n du contour on associe l'aire du triangle $P_{n-1}P_nP_{n+1}$. Les points P_n pour lesquels l'aire associée est inférieure à un certain seuil sont affectés au niveau d'amélioration supérieur.

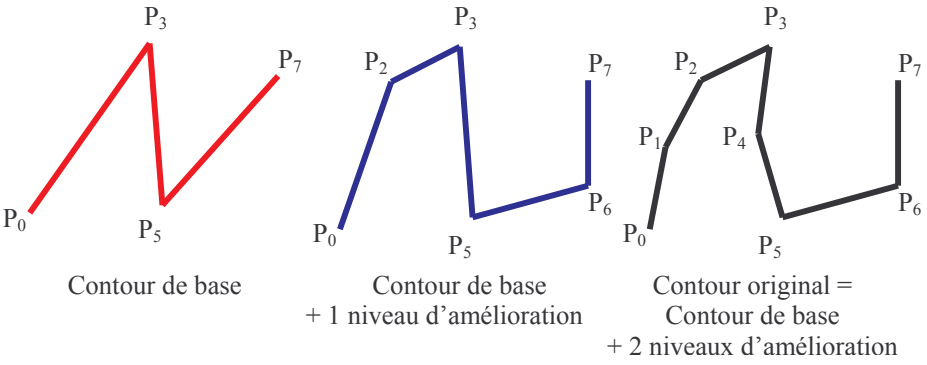

Figure 2. Décomposition d'un contour en flux de base et flux d'amélioration

3.2.2 Encodage scalable par répartition des contours

L'autre type de technique que nous avons implémenté se fonde sur le résultat expérimental suivant. Il existe deux types de lecteurs de contenu vectoriel 2D pour ce qui concerne la manière de tracer les contours. Le premier type utilise une brosse sous forme Bitmap qui est déplacée qui laisse sont empreinte le long du contour. Cette technique est rapide mais produit des résultats peu précis. L'autre type de lecteur calcule à partir de l'épaisseur du contour la liste des polygones remplis formant le contour. Cette technique fournit un résultat entièrement vectoriel mais est plus coûteuse. Nous avons remarqué que la vitesse de rendu des contours varie d'un facteur 4 d'un type à l'autre. C'est pourquoi il nous est apparu intéressant comme technique d'adaptation de séparer les contours de l'intérieur des polygones pour que les terminaux peu puissants puissent jouer la version dégradée du contenu, c'est-à-dire sans les contours alors qu'un terminal suffisamment puissant jouerait le contenu original.

Deux techniques de ce type ont été mises en place. La première technique est très simple, elle consiste à répartir les primitives BIFS sur 2 couches : celles qui définissent un contour sur le flux d'amélioration et les autres sur le flux de base. La seconde technique consiste à affecter une épaisseur nulle au contour sur le flux de base et à créer des commandes BIFS de remplacement qui corrige l'épaisseur et qui sont placées dans le flux d'amélioration.

4 Utilisation de MPEG-21 DIA pour adapter un flux BIFS

Le processus d'adaptation défini par MPEG-21 nécessite une description gBSD du média. Pour certains types de média, il existe une description gBSD normative, c'est le cas de média encodés au format MPEG-4 Vidéo. Cependant, dans le cas de BIFS, nous avons dû définir une telle description. Cette section décrit la manière dont nous avons procédé.

Un document gBSD est une description XML composée d'éléments *gBSDUnit* qui peuvent contenir eux-même d'autres *gBSDUnit*. A chaque *gBSDUnit* correspond une partie du flux média associé à cette description. Cette partie est délimitée par les attributs *start* et *length*. Ces attributs peuvent être exprimés en bits ou en octets, ce qui permet de décrire un flux binaire de manière très fine. De plus, l'attribut *start* peut être exprimé de manière absolue par rapport à l'élément *gBSDUnit* dans lequel il est contenu ou de manière relative par rapport à l'élément *gBSDUnit* précédent. La valeur des bits ou octets décrits par une *gBSDUnit* est inconnue. Il existe un élément *Parameter* qui peut être placé au même niveau qu'un élément *gBSDUnit*. Cet élément signale une valeur présente dans le flux média binaire ainsi que le nombre de bits sur lequel cette valeur est codée.

Exemple de description gBSD

```
<?xml version="1.0" encoding="UTF-8"?>
<dia:DIA xmlns="urn:mpeg:mpeg21:2003:01-DIA-gBSD-NS"
xmlns:dia="urn:mpeg:mpeg21:2003:01-DIA-NS">
<dia:Description xsi:type="gBSDType">
<Header>
<ClassificationAlias alias="M4B"
href="urn:mpeg:mpeg4:bifs:cs:syntacticalLabels"/>
<DefaultValues addressUnit="byte" addressMode="Absolute"
globalAddressInfo="file.bifs"/>
</Header>
<gBSDUnit start="0" length="6253"
syntacticalLabel=":M4B:AccessUnit">
<Header>
<DefaultValues addressUnit="bit" addressMode="Absolute"/>
</Header>
<gBSDUnit start="0" length="1012"
syntacticalLabel=":M4B:Command"/>
<gBSDUnit start="1012" length="12685"
syntacticalLabel=":M4B:Command"/>
<gBSDUnit start="13697" length="35"
syntacticalLabel=":M4B:Command"/>
<gBSDUnit start="13732" length="84"
syntacticalLabel=":M4B:Command"/>
<gBSDUnit start="13816" length="84"
syntacticalLabel=":M4B:Command"/>
<gBSDUnit start="13900" length="99"
syntacticalLabel=":M4B:Command" marker="LAYER-1"/>
<gBSDUnit start="13999" length="99"
syntacticalLabel=":M4B:Command" marker="LAYER-4"/>
```

```
<gBSDUnit start="14098" length="99"
syntacticalLabel=":M4B:Command" marker="LAYER-5"/>
<gBSDUnit start="14197" length="99"
syntacticalLabel=":M4B:Command" marker="LAYER-2"/>
<!-- ... -->
<gBSDUnit start="49922" length="99"
syntacticalLabel=":M4B:Command" marker="LAYER-3"/>
<Parameter start="50021" length="1" name=":M4B:MoreCommand">
<Value>0</Value></Parameter>
<Parameter start="50022" length="2" name=":M4B:AUPad">
<Value>0</Value></Parameter>
</gBSDUnit>
<!-- ... -->
</dia:Description>
```

L'exemple gBSD ci-dessus montre une partie d'un document gBSD décrivant un flux BIFS « scalable ». Un premier niveau d'élément *gBSDUnit* décrit les trames BIFS (*Access Units*) avec un adressage en octets. Un autre niveau décrit les commandes BIFS avec un adressage en bits. Certaines commandes possèdent un attribut *marker* pour indiquer à quel niveau d'amélioration elles appartiennent. Enfin, deux éléments *Parameter* sont présents. Le premier indique la fin de la trame BIFS, l'autre indique la longueur du bourrage car les trames BIFS sont alignées à l'octet.

La transformation XSLT associée à une telle description implémente un algorithme de transformation composé de règles élémentaires de transformation. Un paramètre d'entrée décrit le nombre de couches d'améliorations souhaitées dans la description gBSD après la transformation. Ensuite, il existe quatre règles de transformations. La première supprime tous les éléments *gBSDUnit* dont l'attribut *marker* indique une couche d'amélioration d'indice supérieur au paramètre d'entrée. La seconde supprime la description du bourrage (*AUPad*) car celui-ci doit être recalculé du fait que certaines commandes BIFS ont été enlevées du flux original. La troisième règle XSLT effectue ce calcul. Enfin, la dernière règle recopie tout reste du document gBSD en entrée.

5 Résultats

Les techniques et algorithmes présentés dans les sections précédentes ont été implémentés dans un encodeur BIFS dans le cadre du projet européen ISIS. Les fichiers BIFS générés ont été adaptés suivant le modèle MPEG-21 grâce au processeur gBSD du projet. La figure 3 montre un exemple d'adaptation d'un personnage. Les trois images montrent, de gauche à droite, la couche de base, puis la couche de base et successivement une et quatre couches d'amélioration en utilisant la technique de répartition des points en couches à partir de la courbure des contours. Il apparaît sur ces images que la qualité du flux de base peut être grandement dégradée si les seuils de filtrage sont mal choisis. Par exemple le contenu correspondant à l'image 1 est inutilisable d'un point de vue applicatif. En revanche, le contenu correspondant à l'image 2 l'est avec un gain de 25% sur le débit par rapport au contenu original représenté sur l'image 3.

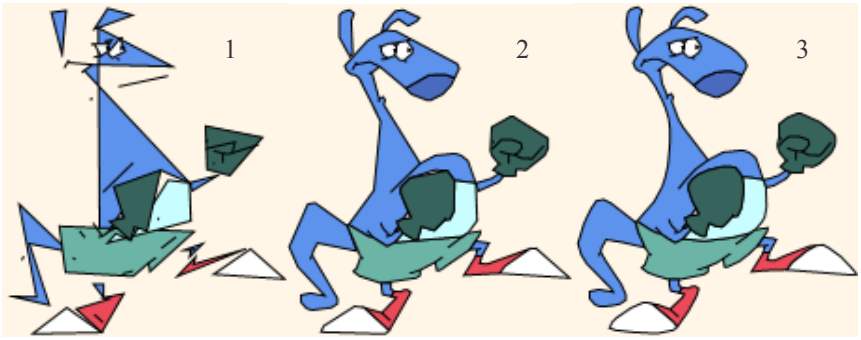

Figure 3. Exemple d'adaptation d'un personnage (seuils 0,3 ; 0,7 ; 1)

De plus, l'encodage « scalable » induit un surcoût en terme de codage. Il faut donc trouver un compromis entre le nombre de couches nécessaires et la taille du flux complet, i.e. couche de base et toutes les couches d'amélioration. La figure 4 illustre ce problème.

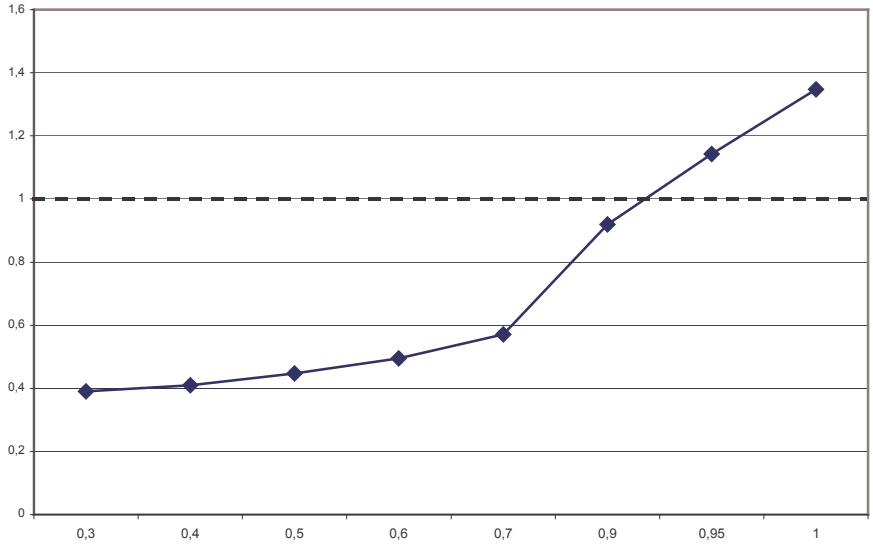

Figure 4. Taille du flux adapté divisé par la taille du flux d'origine en fonction du seuil de filtrage des points

Sur cet exemple, le contenu a été encodé avec 7 flux d'améliorations. On remarque que la taille du flux scalable complet, paramètre d'encodage égal à 1, est supérieure de près de 40 % à la taille du flux non scalable représenté par la droite horizontale pointillée. De même la taille du flux de base et de six flux d'améliorations lui est supérieure de près de 20%.

6 Conclusion et Perspectives

Nous avons montré dans ce document qu'il était possible d'appliquer le modèle d'adaptation MPEG-21 à des contenus de type dessin animé 2D encodés en BIFS. Cependant, les premiers résultats démontrent qu'il existe des compromis à trouver entre le nombre de couches d'amélioration à encoder et la taille du flux complet résultant. De même, un travail plus approfondi sur les techniques d'encodage BIFS permettant un encodage scalable doit être fait pour évaluer si d'autres techniques d'encodage existent et pour améliorer l'efficacité de codage des techniques proposées ici. Par exemple, la détermination des paramètres optimaux de réduction de points dans les couches de bases. Enfin, une perspective de travail réside dans l'adaptation dynamique de contenu, c'est à dire en cours de streaming, en fonction par exemple de la bande passante disponible. Ce sera le cadre de nos travaux dans le projet de recherche européen DANAE [10].

7 Remerciements

Ce travail a été partiellement financé par la commission européenne dans le cadre du projet IST ISIS. Nous tenons à remercier les membres de ce projet pour les outils qu'ils ont mis à notre disposition pour valider notre travail dans une chaîne complète d'adaptation MPEG-21. Enfin, nos remerciements vont également à la société MediaPegs pour nous avoir autorisé à utiliser leurs contenus dans le cadre de nos recherches.

Références

1. Jan Bormans, Keith Hill, "MPEG-21 Overview", http://www.chiariglione.org/mpeg/standards/mpeg-21/mpeg-21.htm
2. Projet IST-2001-37253 [site internet], "Intelligent Scalability for Interoperable Services", http://isis.rd.francetelecom.com
3. MediaPegs [site internet], http://www.mediapegs.com
4. Cartoon Network [site internet], http://www.cartoonnetwork.com
5. Concolato et al., "Codage MPEG-4 de dessins animés", Compression et Représentation des Signaux Audiovisuels 2001,
6. Concolato et al., "Creating and Encoding of Cartoons Using MPEG-4 BIFS : Methods and Results", IEEE Trans. Circuits and Systems for Video Technology, vol. 13, num. 11, pp. 1129-1135
7. Rob Koenen, "Overview of the MPEG-4 Standard", http://www.chiariglione.org/mpeg/standards/mpeg-4/mpeg-4.htm
8. Extensible Markup Language (XML) 1.0 (Second Edition), http://www.w3.org/TR/REC-xml
9. XSL Transformations (XSLT) Version 1.0, http://www.w3.org/TR/xslt
10. Projet IST FP6 DANAE, "Dynamic and Distributed Adaptation of scalable multimedia conteNt in a context-Aware Environment", http://danae.rd.francetelecom.com