ANNÉE 2016

THÈSE / UNIVERSITÉ DE RENNES 1
sous le sceau de l'Université Bretagne Loire

pour le grade de

DOCTEUR DE L'UNIVERSITÉ DE RENNES 1

Mention : Traitement du Signal et Télécommunications

Ecole doctorale Matisse

présentée par

Philippe Bordes

Préparée à l'unité de recherche UMR 6074 - INRIA
Institut National Recherche en Informatique et Automatique

Adapting Video Compression to new formats

Thèse soutenue à Rennes le 18 Janvier 2016

devant le jury composé de :

Thomas SIKORA
Professeur [Tech. Universität Berlin] / rapporteur

François-Xavier COUDOUX
Professeur [Université de Valenciennes] / rapporteur

Olivier DEFORGES
Professeur [Institut d'Electronique et de Télécommunications de Rennes] / examinateur

Marco CAGNAZZO
Professeur [Telecom Paris Tech] / examinateur

Philippe GUILLOTEL
Distinguished Scientist [Technicolor] / examinateur

Christine GUILLEMOT
Directeur de Recherche [INRIA Bretagne et Atlantique] / directeur de thèse

Résumé en français

Adaptation de la Compression Vidéo aux nouveaux formats

Introduction

Le domaine de la compression vidéo se trouve au carrefour de l'avènement des nouvelles technologies d'écrans, l'arrivée de nouveaux appareils connectés et le déploiement de nouveaux services et applications (vidéo à la demande, jeux en réseau, YouTube et vidéo amateurs…). Certaines de ces technologies ne sont pas vraiment nouvelles, mais elles ont progressé récemment pour atteindre un niveau de maturité tel qu'elles représentent maintenant un marché considérable, tout en changeant petit à petit nos habitudes et notre manière de consommer la vidéo en général.

Les technologies d'écrans ont rapidement évolués du plasma, LED, LCD, LCD avec rétro-éclairage LED, et maintenant OLED et Quantum Dots. Cette évolution a permis une augmentation de la luminosité des écrans et d'élargir le spectre des couleurs affichables. Le dynamisme des smart phones et des tablettes a alimenté cette tendance tout en popularisant le format « Haute Définition » même pour des petits appareils. Et le format 4K est naturellement devenu incontournable pour la prochaine génération de téléviseurs et dans les salles de cinéma. Inspiré des études menées par NHK sur la Super Hi-Vision, la recommandation ITU-R rec.2020 définit un format de signal vidéo qui pourrait remplir les besoins à la fois en termes de taille d'images (4K et 8K) et de gamut couleur (au-delà des technologies déployées actuellement) pour la décennie avenir.

Dans le même temps, la révolution des smart phones et autres nouveaux appareils interconnectés, en coïncidence avec le développement grandissant des services utilisant la vidéo, requiert des technologies capables d'adapter dynamiquement la qualité du service aux capacités du réseau ou du lien de distribution à tout instant. Dans ce contexte d'environnement en perpétuelle évolution, la compatibilité des nouveaux contenus et formats vidéo avec les systèmes existants est un véritable défi. C'est pourtant un des éléments clé du succès lors du déploiement d'un nouveau standard. Car cette compatibilité permet d'opérer les transitions technologiques en douceur et de développer des relations de confiance dans la durée avec les clients.

Aujourd'hui, les nouvelles techniques de compression vidéo et les nouveaux formats vidéo doivent être conçus tous les deux avec ce haut niveau d'adaptabilité intrinsèque, à la fois en terme de bande passante réseau, de scalabilité des formats (taille d'images, espace de couleur…) et de compatibilité avec l'existant. C'est pourquoi les comités de standardisation tels que MPEG, DVB, BDA et ATSC ont prévu plusieurs phases de déploiement et recommandent d'examiner les différents outils d'extension scalable disponibles, avant de définir les nouveaux systèmes de distribution de la vidéo.

Contributions

Cette thèse s'inscrit dans ce contexte de formats vidéo en constante évolution, avec des couleurs de plus en plus étendues et une luminosité croissante, qui sont autant d'opportunités pour le développement de nouvelles applications et de services. Si de nouveaux services apparaissent pour un parc d'appareils de plus en plus hétérogène en termes de caractéristiques et de capacités, le besoin en une technique de compression du contenu vidéo efficace et adaptable est devenu de plus en plus nécessaire.

Ce document regroupe plusieurs études en lien avec l'évolution de la compression vidéo dans ce contexte. La plupart de ces études ont été menées durant les cinq dernières années, à l'exception de quelques chapitres liés à des travaux plus anciens mais dont les résultats sont toujours d'actualité. Ces études sont donc principalement basées sur des extensions ou des améliorations du standard de compression vidéo HEVC, mais aussi AVC son prédécesseur largement déployé. Elles ont donné lieu à de nombreuses contributions à MPEG.

Considérant les différentes étapes d'une chaîne de distribution vidéo, deux types d'adaptation du codage de la vidéo sont distingués :

- Les adaptations qui exploitent les propriétés du signal et qui sont mises en place lors de la création du bit-stream.
- Les adaptations réalisées sur des flux vidéo compressés existants et qui s'appuient sur des propriétés de flexibilité intrinsèque de certains bit-streams.

Codage vidéo adapté aux propriétés du signal

Le développement du standard HEVC a constitué une opportunité pour étudier et proposer des outils de compression permettant un niveau d'adaptation plus important aux propriétés du signal vidéo. Mais l'objectif principal était l'obtention d'un nouveau codec qui soit considérablement plus efficace que AVC.

Dans cette perspective, l'étude d'un nouveau partitionnement des images pour améliorer la prédiction inter était une idée prometteuse. Son principe est de mieux s'ajuster aux frontières réelles du mouvement et par conséquent de réduire l'énergie des résidus de prédiction par compensation de mouvement. Des gains significatifs peuvent être obtenus en comparaison d'une segmentation classique construite à partir de blocs rectangulaires, même si la complexité de l'encodeur est augmentée. Nos travaux présentent des adaptations de l'algorithme permettant de réduire cette complexité tout en préservant un haut niveau de performance.

Ce principe de segmentation du mouvement est étendu à la modélisation long-terme du mouvement à l'aide de trajectoires. Cependant, les résultats encourageant de cette approche ne peuvent pas encore rivaliser avec les architectures de codec hybride traditionnelles. Probablement parce que cette technique n'est pas encore mature et ce formalisme n'est pas bien adapté aux mesures de distorsions du signal basées sur des métriques de haute-fidélité comme le psnr.

Par ailleurs, l'analyse des réponses faites au « Call for Evidence » (CfE) de MPEG confirma que ce nouveau standard n'intègrerait pas de rupture technologique en ce qui concerne les principes généraux de de compression vidéo. Cependant, au regard de la variété des nouveaux formats

vidéo, il se pourrait qu'ils réagissent de manière différente aux algorithmes de compression et par conséquent nécessitent des techniques d'adaptation dépendantes des contenus.

En effet, en observant des panels de séquences vidéo de test communément utilisées pour la compression vidéo, il s'avère que les composantes couleurs du signal peuvent présenter des niveaux de corrélation importants et une excursion limitée. Nous avons étudié comment exploiter ces propriétés et nous proposons quelques adaptations simples pour augmenter l'efficacité du codec HEVC.

Dans le cas où les caractéristiques du signal vidéo varient avec le temps, la prédiction temporelle (inter-images) peut être impactée et les performances du codec vidéo peuvent être dégradées. Nous avons investigué l'efficacité de la prédiction pondérée (« weighted prediction »), ce qui a concouru à son adoption dans HEVC, et nous proposons une nouvelle méthode d'estimation des paramètres explicites. De plus, nous expliquons pourquoi l'ancienne prédiction pondérée implicite (« implicit weighted prediction ») n'est plus efficace dans le cadre de HEVC et nous en déduisons une version modifiée combinant prédiction implicite et explicite.

Comme pour toutes les architectures de codec hybrides, la quantification des résidus transformés est l'outil le plus efficace pour ajuster temporellement et localement le niveau de compression. Nous proposons une technique de prédiction du pas de quantification dérivée des similarités des vecteurs de mouvements appartenant aux groupes de quantification voisins, qui obtient des résultats comparables avec la méthode finalement retenue pour HEVC. Le bénéfice de la quantification locale est illustré avec un encodeur intégrant un modèle de perception psycho-visuelle permettant d'obtenir une qualité globalement homogène sur toute l'image.

Méthodes de codage vidéo permettant l'adaptation des flux après leur création

Toutes ces techniques opèrent lors de l'étape de la création du bit-stream. Pourtant, un certain niveau d'adaptation peut-être nécessaire après que le flux ait été généré. C'est par exemple le cas pour les applications de type vidéo à la demande. Si le transcodage est une technique d'adaptation traditionnellement utilisée par les professionnels, elle demande des capacités de calcul importantes et n'est donc pas appropriée pour des applications grand public impliquant un grand nombre de clients simultanément.

Dans le cas du codage « off-line » (non temps réel), les systèmes à flux multiples (« stream-switching ») ont largement démontré leur efficacité, mais ils souffrent d'un manque de flexibilité, comme le caractère imposé des points de saut inter-flux (« switching points ») et la moindre efficacité de codage en corollaire. Comme réponse possible à cette limitation, nous explorons le codage à résolution variable, et nous proposons une utilisation particulière de l'extension scalable de HEVC (SHVC) contraint à se comporter comme un codec mono-layer dans un schéma de « stream-switching ».

En comparaison, SHVC est probablement la solution la plus flexible et la plus performante d'un point de vue technique. C'est pourquoi nous avons aussi étudié les extensions nécessaires pour que SHVC supporte efficacement l'adaptation de format (d'espace couleur) de la vidéo comme une nouvelle dimension de scalabilité à part entière (« colour gamut scalability »). Cette technique a été finalement incorporée au standard SHVC et en constitue probablement l'un des outils les plus innovants. De ces travaux, nous avons dérivé des metadata et un modèle associé pour opérer une transformation couleur (« colour re-mapping ») des images reconstruites,

fournissant potentiellement une rétro compatibilité (couleur) avec tous les types d'écrans existants.

Conclusion et perspectives

Dans cette thèse, plusieurs des techniques étudiées ont été proposées à MPEG. Certaines ont été adoptées dans le standard HEVC et ses extensions (seconde édition), et aussi dans la nouvelle norme UHD Blu-ray Disc. Elles pourraient être prochainement poussées dans des organismes de recommandation (tels que DVB, ATSC, HDMI…) pour la distribution de vidéo et le déploiement de services associés.

Nous avons investigué des méthodes variées pour adapter le codage de la vidéo aux différentes conditions de distribution et aux spécificités de certains contenus. Suivant les scénarios, nous montrons que l'on peut sélectionner et combiner plusieurs d'entre elles pour répondre au mieux aux besoins des applications.

En comparaison avec les standards précédents, HEVC et ses extensions intègrent un niveau d'adaptabilité sans précédent. Nous prédisons que cette tendance se poursuivra dans les années à venir et que le futur standard de compression vidéo devra intégrer un niveau minimal d'adaptation potentiel.

Pour les applications bas débit, le choix d'une métrique de distorsion visuelle qui soit mieux corrélée avec la qualité subjective perçue que le psnr, facilitera l'avènement de nouvelles techniques de codage.

Les métadonnées futures intègreront des modèles d'adaptation locale plus évolués pour mieux prendre en compte les spécificités et les traitements locaux dans les images. Etant donné la toujours plus grande variabilité des écrans, un modèle qui pourrait s'adapter aux caractéristiques particulières de l'écran de visualisation permettrait une qualité plus maîtrisée du service de bout-en-bout.

Table of Contents

Acknowledgements

First I would like to express my gratitude to my PhD advisor Mrs. Christine Guillemot who has guided me into the whole thesis process.

Besides my advisor, I would like to thank the rest of my thesis committee, Thomas Sikora, François-Xavier Coudoux, Olivier Deforges and Marco Cagnazzo who have accepted to act as rapporteur for thesis.

My sincere thanks also goes to my colleagues, in particular Pierre Andrivon, Edouard François, Patrick Lopez, Franck Hiron and Philippe Guillotel that worked with me during these last years and without whom nothing would have been possible.

At last, I would like to express my appreciation to my company Technicolor and training staff that offered me the opportunity to realize this project.

General introduction

The video compression domain is at the confluence of new display technologies advent (OLED, quantum-dot…), the arrival of new screen devices (smart phones, tablets…) and the deployment of new services and applications (VOD streaming, gaming, YouTube and personal generated video…). Some of these technologies were existing for a while, but they have recently grown up to the age of maturity, representing hundreds of billion dollars markets and contributing to change our way of life and consuming the video content in general.

Display technology has rapidly evolved from plasma, LED, LCD, LCD with LED (local dimming) backlighting, and now OLED and Quantum Dots. This evolution has allowed increasing the brightness of the screens and widering the rendering capability of visible color spectrum. The smart phones and tablets dynamism has fueled this trend while popularizing the high definition format even for small devices. And the 4K format is become a matter of course for the next generation of TVs and movie theaters.

Inspired from the Super Hi-Vision studies of NHK, the ITU-R Rec.2020 defines a video signal format that could fulfil the future requirements both in terms of frame resolution (4K and 8K) and color gamut (beyond all currently deployed display technologies) for the next decade.

At the same time, the revolution of the smart phones and other new inter-connected devices, coinciding with the growing developments of video based services (VOD, network gaming, video sharing), requires technologies enabling dynamic adaptation of the (quality of) service to the network or distribution link instant capability.

In this ever-evolving environment, backward compatibility of the video content or video format with the legacy is becoming more and more challenging. However it is of key importance for the success of new standards market adoption, because it allows smoothing the technological transitions and developing trustee relationship with the customers.

Today, both the new video compression techniques and the new video formats should be designed with all this high level of adaptability in terms of network bandwidth, format scalability (size, color space…) and backward compatibility. That is why the MPEG, DVB, BDA and ATSC standard committees made no mistake when they recommended video scalable extensions tools, multiple phase deployments and defined special functions for legacy compatibility.

Contributions

This thesis takes place in this context of continuously evolving video formats, with wider color and brightness, opening opportunities to new applications and services. At the same time, because such new services on heterogeneous devices are appearing, the need for higher compression efficiency and adaptability is becoming more pressing.

This document groups several studies related to the evolution of the video compression techniques in this context. Most of these studies were conducted during the last five years, except for a few sections which are related to older results which are nevertheless still valid. They are mainly based on extensions or improvements of HEVC, the recently released MPEG video compression standard, but also its predecessor AVC that is widely deployed.

Following the different processing steps in the video distribution chain, two main video coding adaptations stages are distinguished:

- Video Coding adaptations that exploit the signal properties and which take place at the bit-stream creation typically,
- Post-creation stream adaptations relying on intrinsic stream flexibility which occur at the distribution (ex: network) stage.

Video Coding adapted to the Signal properties

The development of the HEVC standard [30] has been the opportunity to study and propose tools dedicated to higher flexibility and adaptation, but the primary objective was to achieve better encoding efficiency compared to AVC.

For this primary purpose, improved frame partitioning for inter prediction was a promising idea. Its principle is to better fit the actual motion frontiers and therefore to reduce the energy of the motion compensated prediction residuals. The experimental results show significant gains [52] compared to classical block-based segmentation, obtained at the expense of encoding cost though. Our work presents some adaptations to reduce the encoder complexity while still preserving high level of performance [58]. This principle of motion partitioning is further extended to the study of long-term motion modeling with trajectories. Nonetheless, the encouraging results of this approach cannot compete with hybrid codec architecture yet. Probably because it is still in its infancy and this scheme is not well adapted to the high fidelity signal measures such as PSNR used in standardization organizations.

Moreover, the analysis of the responses to the MPEG Call for Evidence (CfE) confirmed that no technological breakthrough regarding the next video codec principles would occur. However, given the variety of new video formats, the compression algorithms may react differently and hence may require content adapted techniques at least. The observation of a set of video test sequences commonly used for video compression shows the color signal may have very particular cross-component correlation statistics as well as color samples range limitations. We study how to exploit these properties and we propose simple codec adaptations to increase the HEVC coding efficiency [135].

In case the video signal characteristics vary with time, it may impact the inter-frame prediction and degrade the video codec performance. We have investigated the efficiency of the weighted prediction that legitimated its adoption in HEVC [71][77] and we propose a new method to estimate explicit parameters [66]. We show the former implicit weighted prediction is no more working with HEVC and we come up with a modified scheme combined with explicit that could replace it [79].

All the same, the coding artefacts are not generally homogenously distributed inside the pictures and our perception of these errors depends on the surroundings textures. That is why we also consider the case of

the adaptation to the local features in the frame, exploring techniques for adjusting the quantization step locally in the video encoders [91] and exploiting this feature to perceptually adapt the video coding [102].

Video Coding methods enabling post-creation streams adaptation

All these techniques operate at the bit-stream creation stage. However, in some cases such as one-to-many applications or for systems constrained by off-line encoding limitations, a certain level of adaptability may be still required after the video stream has been produced.

Historically, as corporate research of a company specialized on video coding products, we first faced the problematic of adapting the compressed video streams to the broadband distribution constraints. The transcoding [126] [127] is a natural technique used for professional use cases, but it requires dedicated computing ressources and hence is not adapted to consumer one-to-many scenarios such as VOD.

For these off-line encoding cases, the stream-switching systems has proven its efficiency but suffer from some lack of flexibility, such as imposed switching point instants and loss of coding efficiency as a result. We explore the adaptive resolution coding as a way to alleviate this limitation [107] as well as to constraint the SHVC to behave as single decoding layer in a stream-switching based scheme [134].

In comparison, SHVC is probably the most flexible and performant solution from a technical point of view. Then, we have studied the necessary additions so that SHVC can efficiently address the (color space) video format adaptation as a new scalability dimension (a.k.a. color gamut scalability) [18][19][20] [24][25][37][38][39]. It was incorporated to the SHVC standard and it is probably the most innovative tool compared to previous scalable standards. From this work, we have derived color remapping metadata and an associated model to provide low complexity (color space) backward compatibility to any type of video decoder [121][122][124].

Document organization

This thesis is structured in three main parts. In the first part, a state-of the art recalls some basics of the digital video signal, starting with the color perception and representation, followed by a short overview of common techniques used to evaluate the video signal fidelity. Finally this part introduces the video compression basics and presents the outlines of the recently finalized MPEG HEVC video compression standard, including its scalable extension.

The second part describes the studies carried out by the author in the context of video coding improvement in general, that exploit the knowledge of some particular properties of the video signal to be encoded. Most of the work is related to HEVC, but one to AVC. First, different approaches for structuring the frames for coding and based on motion representation are depicted, and the performance is compared to existing codecs. Next, a set of video coding tools globally or locally adapted to the spatial signal characteristics are proposed.

The third part focuses on the adaptability of the created video bit-streams. The addressed technologies are the stream-switching and the scalable video coding. Several coding tools are considered that allow adapting the bit-streams and the reconstructed frames to the heterogeneity of the actual decoding devices capability. These levels of adaptation include the color rendering and the bandwidth capacity.

Many of these works have led to technical proposals to MPEG, and several of them have been adopted in HEVC and SHVC standards.

Part I - Video Signal and Video Coding: State of the Art

1. Video Signal/Formats

1.1 Elements of human visual perception

1.1.1 Color perception: the physics

It all starts with our ability to see colors. Experts estimate that we can distinguish 10 million colors…

The retina of the human eye is composed of two types of photoreceptor cells. The cone cells are responsible for color vision as well as eye color sensitivity. They function best in relatively bright light, as opposed to rod cells that work better in dim light (night vision).

There are three types of cones, each with a different pigment, namely: L-cones, M-cones and S-cones (LMS). Each cone is therefore sensitive to visible wavelengths of light that correspond to Long-wavelength (red), Medium-wavelength (green) and Short-wavelength (blue) light respectively (Figure 1). However, different individuals may have cones with (slightly) different color sensitivity.

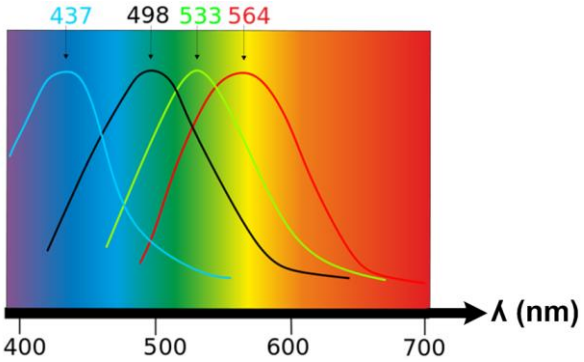


Figure 1: Normalized response spectra of cones (blue, green, red) and rods (black).

From the observation of the cones responses (Figure 1), one can make two important remarks:

1) Two light sources made up of different mixtures of various wavelengths may appear to be the same apparent color to an observer if they produce the same tri-stimulus values. This effect is called *metamerism*.
2) Since the cones responses curves overlap, certain tri-stimulus values are thus physically impossible (for instance LMS tri-stimulus values that are non-zero for one component, and zeros for the others).

1.1.2 XYZ Color space

Besides the previous remarks, a tri-chromatic color space based on LMS has another limitation: when judging the relative luminance (brightness) of different colors, humans tend to perceive light within the green parts of the spectrum as brighter than red or blue light of equal power. That is why the CIE[1] RGB associates each components with weights (1., 4.59, 0.06), to ensure that identical r=g=b color coordinates correspond to perceived white color.

The CIE defines the color functions $\bar{x}(\lambda)$, $\bar{y}(\lambda)$, $\bar{z}(\lambda)$ of wavelength λ (Figure 2) that allows deriving the color tristimulus (X,Y,Z) corresponding to a light wave signal $S(\lambda)$ as follows (1):

[1] CIE: Internal Commission on Illumination standard.

$$\begin{cases} X = k.\int \bar{x}(\lambda).S(\lambda).d\lambda \\ Y = k.\int \bar{y}(\lambda).S(\lambda).d\lambda \\ Z = k.\int \bar{z}(\lambda).S(\lambda).d\lambda \end{cases} \quad (1)$$

The (X,Y,Z) are converted to (x,y,z) normalized to 1 (x+y+z=1). One rather uses CIE-xyY, with Y representing the Luminance, the CIE-xyY being defined as follows (2):

$$\begin{cases} X = \frac{Y}{y}x \\ Z = \frac{Y}{y}(1-x-y) \end{cases} \quad (2)$$

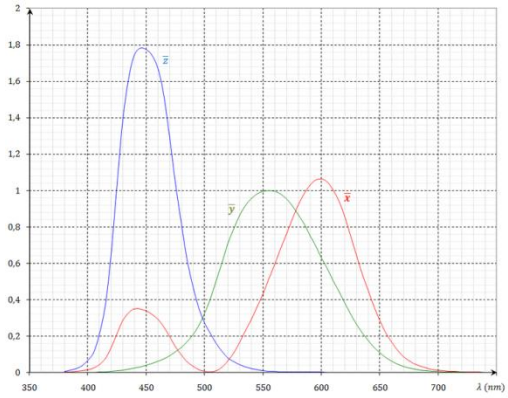


Figure 2: CIE-XYZ color functions.

All the human visible colors can be represented in the xy chromaticity plane. But our perception of the visual signal depends upon luminance too, a dimension that is missing from a chromaticity diagram.

Another limitation is that it is not perceptually uniform. An increment in x of 0.01 is far more perceptible near blue than a 0.01 increment near green. CIE-LAB (*L*a*b**) coordinates are much more perceptually uniform than xyY. This is illustrated with the MacAdam ellipses that figure out the local colors which are indistinguishable to the average human eye, from the color at the center of the ellipse (Figure 3).

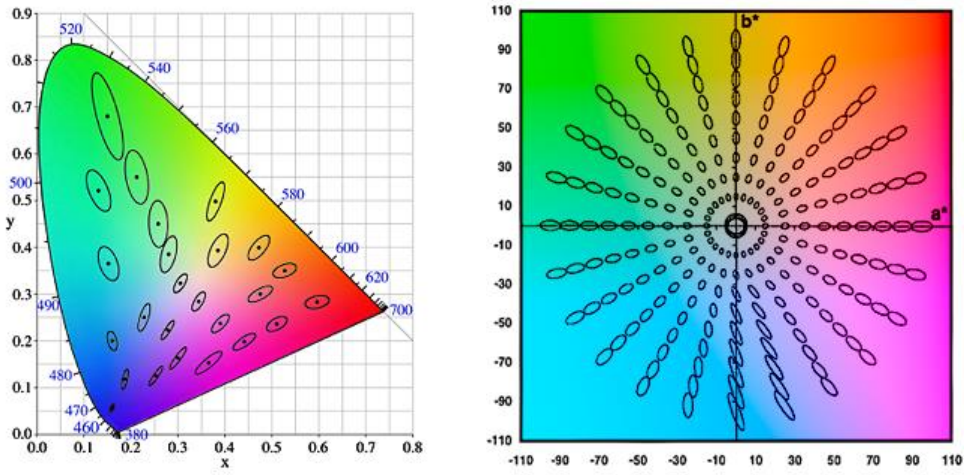


Figure 3: The neighboring Mac Adam ellipses have uniform size in L*a*b* (right) but not in *xyY* (left).

1.1.3 Luminance contrast sensitivity/perception

The Human Vision System (HVS) is based on extremely complex mechanisms involving not only the eye (from the crystalline lens to the optical nerve) but also the brain for a major part. In the following, we list some simple psycho-visual properties that can be exploited to build an HVS model with reasonable complexity.

A first set of HVS properties is the visual thresholds that can be modeled as a non-linear relation between a luminance stimulus L (presented in the previous section as Y) and its visibility. The most famous one is the Weber law for luminance contrast sensitivity that states that the just-noticeable difference between two stimuli (ΔL) is proportional to the magnitude of the stimuli (L): $\Delta L \propto L$. For low luminance values, the DeVries-Rose law states $\Delta L \propto L^{½}$.

Derived from this physics, the luminance signal L is quantized into numerical values C using the following non-linear equation (3) that allocates more precision to low luminance than to high luminance:

$$C = K.L^{1/\gamma} \qquad (3)$$

where L is expressed in cd.m^{-2}, K is a constant set according to the desired numerical range values and γ is around 2 typically. This Gamma γ model is the basis of most of Optical to Electronic Transfer Functions (OETF), the rendering display implementing an inverse EOTF.

Traditionally, the TV systems uses $\gamma=2.2$ (BT-1886 [108]) while the Cinema uses $\gamma=2.4$. The difference originates in the nature of the viewing conditions mainly, the movie theaters being controlled dark environment with low luminance rendered signal (~100 cd/m^2), whereas TV viewers are watching, in day ambient light, screens with higher luminous intensity (~300 cd/m^2).

Besides, for High Dynamic Range (HDR) video signals with potentially significantly higher luminance peak (1000, 4000 or even 10000 cd/m^2), these former OETFs are no more valid because they would use unnecessary precision for coding brighter values to the detriment of darker ones. The SMPTE recently proposed a new EOTF for HDR [109] and several other candidates have been proposed in MPEG-XYZ group [110] and in ITU-R (ITU-R SG6/W6-C group) [131] .

Contrast sensitivity functions

One defines the contrast of a difference of luminance values ΔL as:

$$Contrast = \frac{\Delta L}{L} \qquad (4)$$

where L is the average luminance in cd.m^{-2}. The level of contrast necessary to elicit a perceived response by the human visual system is known as the contrast threshold. The inverse of the threshold is known as the contrast sensitivity. Then the Weber law states this threshold is a constant independent of the luminance value.

Several studies have shown the contrast sensitivity varies with the background spatial frequency v (in cycles.deg^{-1}). The relationship between the contrast sensitivity and spatial frequency is known as the contrast sensitivity function (CSF) of the human visual system. Different contrast sensitivity models can be found [95][96][97][98] (Figure 4).

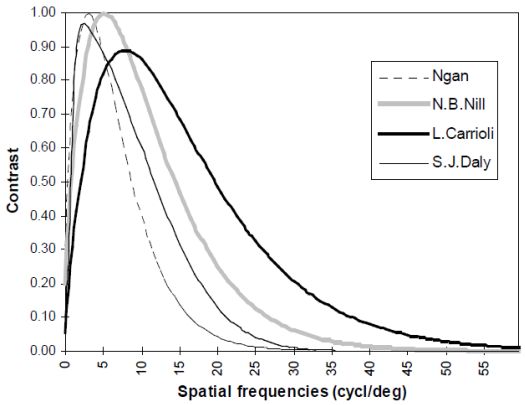


Figure 4: Several HVS Contrast sensitivity models.

D.H.Kelly [99] and S.Daly [100] measured the HVS contrast sensitivity for several spatial frequencies (cycles.deg^{-1}) by using various resolution gratings patterns with different velocities (deg.s^{-1}), making the assumption that the eye movements have been stabilized (Figure 5). From these results, it turns out the sensitivity for small details decreases with motion, and for intermediate velocities (3 deg/s) low spatial frequencies are more visible than high spatial frequencies ones.

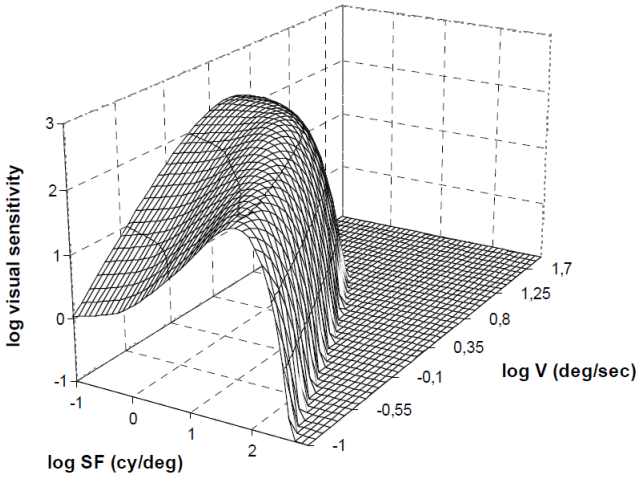


Figure 5: HVS Contrast sensitivity for moving patterns.

1.2 Color Video signal formats

sRGB

We have seen that the CIE has defined a set of tri-chromatic color spaces that allow representing visible colors. They de-correlate the Luminance signal, i.e. the intensity in cd.m^{-2} to which the human eye is sensitive, from the chromaticity, i.e. the color (hue, saturation).

However, given the rendering technology limitations and considering the requirements of traditional applications, it is sometimes not necessary to cover such a wide color range. Then it is wasteful to use codewords for representing colors that cannot be displayed or which turn out to be useless eventually.

For example, the sRGB [128] uses a tri-chromatic color basis defined with three color primaries corresponding to the standard TV colors [85]. All the colors represented with three coordinates (r,g,b) ϵ [0;255]^3 are situated in a triangle in the xy chromaticity diagram.

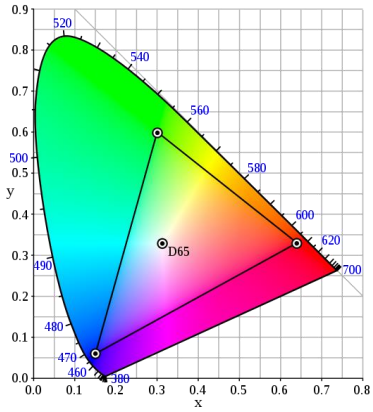


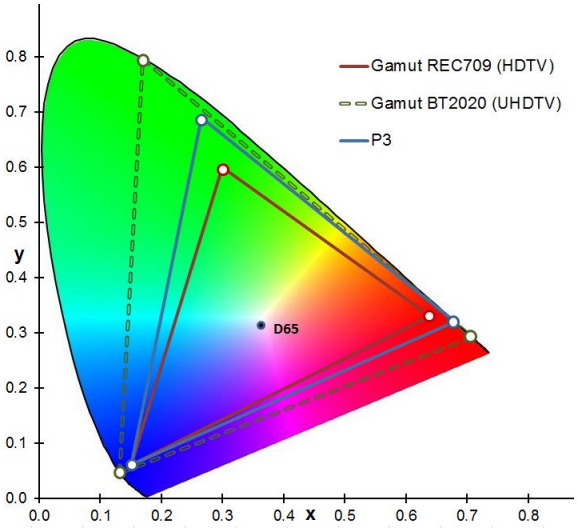
Figure 6: sRGB gamut in the xy chromaticity diagram.

Then the RGB to XYZ matrix conversion can be simply derived from the three color primaries $(x_C,y_C)_{C=R,G,B}$ coordinates and normalized white coordinates (x_W,y_W), in order that the white value corresponds to R=G=B=1 (5) (SMPTE-RP-177 [10]):

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} x_R & x_G & x_B \\ y_R & y_G & y_B \\ 1-x_R-y_R & 1-x_G-y_G & 1-x_B-y_B \end{pmatrix} \begin{pmatrix} C_R & 0 & 0 \\ 0 & C_G & 0 \\ 0 & 0 & C_B \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} = [P].[C].\begin{pmatrix} R \\ G \\ B \end{pmatrix} \quad (5)$$

$$\text{with: } \begin{pmatrix} C_R \\ C_G \\ C_B \end{pmatrix} = [P]^{-1}.\begin{pmatrix} x_W/y_W \\ 1 \\ z_W/y_W \end{pmatrix}$$

The most used color primaries systems are the Rec.709 for High Definition TV [85] and the DCI-P3 [12] for Digital Cinema. The ITU-R has recently released a new format for Ultra High Definition TV (UHDTV) [5] supporting wider color space as depicted in Figure 7.



Rec.709	x	y
Blue	0.15	0.06
Green	0.30	0.60
Red	0.64	0.33
White	0.31	0.33

DCI-P3	x	y
Blue	0.15	0.06
Green	0.27	0.69
Red	0.68	0.32
White	0.31	0.35

Rec.2020	x	y
Blue	0.13	0.05
Green	0.17	0.80
Red	0.71	0.29
White	0.31	0.33

Figure 7: typical color primaries used in Digital TV and Digital Cinema.

Application of OETF : Constant Luminance (CL) and Non-Constant Luminance (NCL)

There are two ways of applying the OETF: the Constant Luminance (CL) and the Non-Constant Luminance (NCL).

Constant Luminance (CL)

The coding of RGB may be done after quantization (to obtain numerical integer values) of the trichromatic color samples values. However, since the human eye is mainly sensitive to Luminance, one prefers to encode the Y component plus two secondary chrominance parameters derived from R and B that may be coarsely encoded, Y being re-scaled into Y' with a transfer function similar to the lightness sensitivity of human vision (see OETF in 1.1.3). This is the Constant Luminance (CL) scheme depicted in Figure 8.

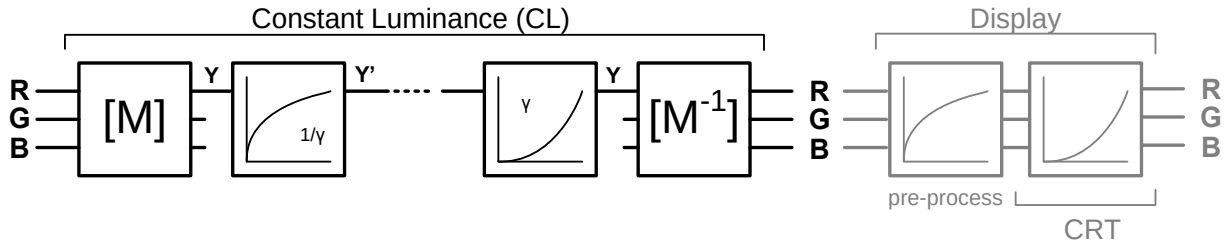


Figure 8: Principle of Constant Luminance (CL). The displays need to implement pre-processing correction to compensate for non-linear power function in case of CRT.

Non-Constant Luminance (NCL)

At the display side, to compensate for the non-linear gamma (γ) power function of the CRT, one has to pre-process the RGB signal before rendering. Because this CRT power function is almost same as the inverse of the Y component re-scaling, an alternative Non-Constant Luminance (NCL) scheme has been proposed, where the re-scaling is directly applied on the RGB components (Figure 9). In that way, the inverse re-scaling and the CRT pre-processing can be removed. However, this simplification is made at the expense of the value of the re-scaled Y that is not the same for NCL as for CL. The Y_{CL} corresponds to the true luminance but not Y_{NCL} (that is called Luma). The issue arises when the two other secondary chrominance (called Chroma for NCL) components are coarsely encoded, which may result in a reconstructed RGB signal with bad luminance and more visible artefacts than with CL, as reported by C.Poynton [137].

Besides, the CRT displays are progressively replaced with other technology that no more justify using NCL, except for compatibility with legacy system.

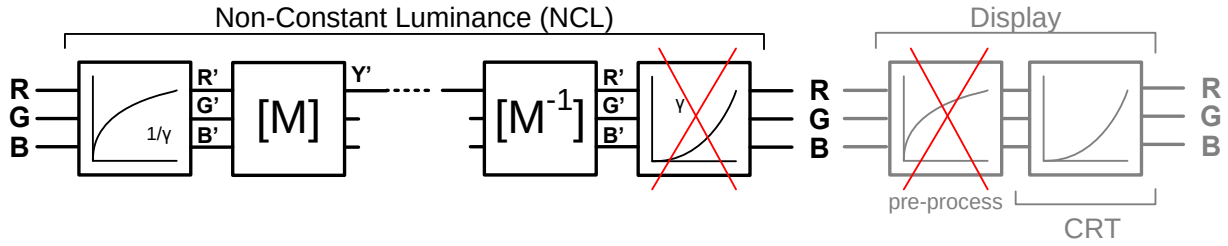


Figure 9: Principle of Non-Constant Luminance (NCL). The inverse re-scaling and the CRT displays pre-processing can be removed.

Y' Cb Cr

The matrix [M] allows deriving the Y'CbCr (a.k.a. YUV) digital signals from the R'G'B' re-scaled color samples (NCL). The Rec.601 [84] specifies that the components (Y',Cb,Cr) are computed as:

$$\begin{cases} Y' = m_{10}.R' + m_{11}.G' + m_{12}.B' \\ C_b = \frac{0.5\times(B'-Y\prime)}{(1-m_{12})} \\ C_r = \frac{0.5\times(R'-Y\prime)}{(1-m_{10})} \end{cases} \quad (6)$$

where m_{ij} are given by (5).

Then the Cb,Cr components are defined as the complementary to B and R respectively and the value of [M] depends on the color primaries and reference white.

In lossy coding systems, since HVS is more sensitive to luma (Y) than to chroma (Cb,Cr), higher resources are dedicated to the coding of luma than for chroma typically. For example, the chroma components are sub-sampled by two horizontally and vertically in 4:2:0 format (see also 8.2.1).

However, it is worth noting that only a portion of the Y'CbCr 3D range values is meaningful (Figure 10).

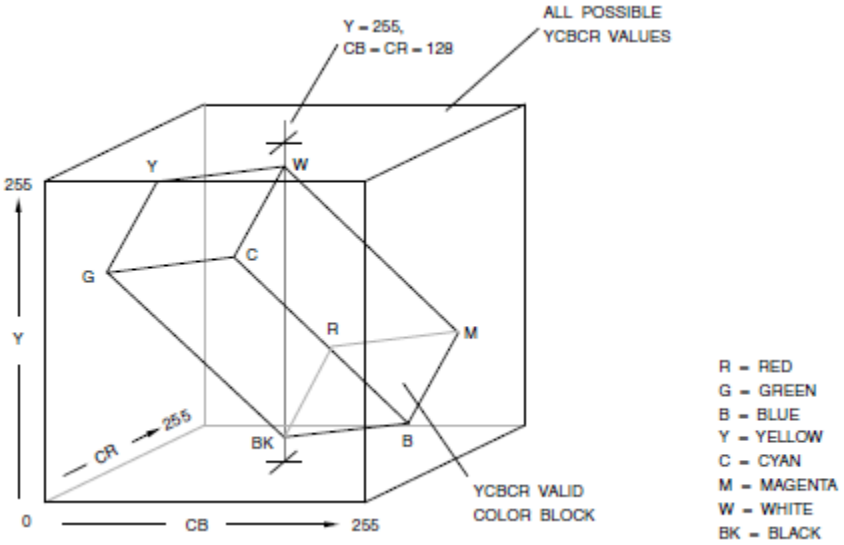


Figure 10: R'G'B' limits transformed into Y'CbCr.

Headroom/Footroom

In practice [84], a reduced range of the possible numerical values are used to code the RGB or YUV samples. For example, for 8-bits coding, only numerical values between 16 and 235 may be used while the full range is 2^8=256. This is for some historical legacy and particular use cases reasons:

- To support over-shoot due to filtering (inheritance of analogic world), where the filter process may exceed the normal sample range (ex: [0;255] for 8-bits data). In this case, the headroom/footroom allows provisioning the necessary margin.
- Some code words below/upper the headroom/footroom can be reserved for signal synchronization (ex: SDI, HDMI).
- Some specifications may use these extra range codes to represent values beyond normal range (ex: specular values exceeding the normal range by 800% in the case of Rec.709-800% specification).

1.3 Video Signal fidelity measurement

Finding the adequate video quality metric is of key importance in video coding because it allows selecting the best compression technology that will provide the better trade-off between compression (bit rate) and the subjective quality of the reconstructed frames. A fine quality metric allows performing the necessary tuning of the coding algorithms, without setting up costly subjective tests that require many resources (test viewers, controlled viewing conditions, etc…) and that have difficulty to distinguish the small differences.

One commonly and widely used video compression distortion metric is the Peak Signal to Noise Ratio (PSNR). It is computed as:

$$MSE = \frac{1}{m \times n} \sum_{i=0}^{m} \sum_{j=0}^{n} \left[ima_{test}(i,j) - ima_{ref}(i,j) \right]^2, \quad PSNR = 10.log_{10}\left(\frac{MAX^2}{MSE}\right) \qquad (7)$$

where: ima_{test} is the reconstructed picture,
ima_{ref} is the reference picture,
(m,n) is the picture size,
MAX is the maximal picture sample value.

The PSNR is useful to measure the reconstructed signal (ex: image) fidelity with the reference (or original) one. It has the advantage to be simple to implement. However, PSNR does not perfectly correlates with a perceived visual quality due to the non-linear behavior of the Human Visual System (HVS, see 1.1.3) and because HVS is particularly sensitive to local contrast variations.

That is why one may prefer the Structural Similarity Index Metric (SSIM) that better catch the structural information degradation through the calculation of local means and variances in Luma. The SSIM formula is given by:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2\mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \qquad (8)$$

where x and y are the images to be compared, μ_x, σ_x, μ_y, σ_y are the local average and variance of x and y respectively, and c_1, c_2 are constant equal to $(0.01 \times 2^{bitdepth})^2$ and $(0.03 \times 2^{bitdepth})^2$. SSIM range value is [0;1], 1 meaning x and y are identical.

Several variants exist that compute the PSNR or SSIM at multiple image scales (resolutions) for instance, also known as m-psnr and m-ssim, to better mimic the HVS quality assessment.

If the PSNR and SSIM are well correlated with subjective tests in case of luminance, they are very much less efficient to measure the perceived color degradation/difference. For this purpose, the CIE has defined the DeltaE-76 that computes the signal difference in the $L^*a^*b^*$ domain (see 1.1.2)(9), revisited in 1994 and 2000 for improved perceptual uniformity assessment. It is computed as:

$$\Delta E_{76} = \sqrt{\left(L_x^* - L_y^*\right)^2 + \left(a_x^* - a_y^*\right)^2 + \left(b_x^* - b_y^*\right)^2}, \qquad \Delta E_{dB} = 10.log_{10}\left(\frac{MAX^2}{\overline{\Delta E_{76}}}\right) \qquad (9)$$

where: (L_x^*, a_x^*, b_x^*) is the CIE-LAB value of one color sample of the reconstructed picture,
(L_y^*, a_y^*, b_y^*) is the CIE-LAB value of one color sample of the reference picture,
$\overline{\Delta E_{76}}$ is the average of ΔE_{76} for all the color samples in the pictures.

The finding of the best video quality metric is still on-going, carried out by some technical experts groups such as the Video Quality Expert Group (VQEG). The arrival of new video formats with Wide Color Gamut and High Dynamic Range may challenge the validity of the existing models.

2. Video Compression

The video coding algorithms take advantage of the intrinsic redundancy existing in the video sequences to reduce the amount of data to encode:

- The temporal redundancy between consecutive video frames is very important in general. Ideally, one may encode the first frame, and next only the moving and changing objects in the subsequent frames. This is the most straightforward and powerful property used in video coding.
- Inside an image, there may exist spatial redundancy too. Some objects or patterns may be repeated at different locations in the same frame. In addition, the signal may have local periodicities that can be modeled through periodic signal decompositions. In particular, contiguous pixels have many chances to share same color values.
- At last, the statistical analysis (probability) of the occurrence of the various symbol syntax elements that compose the video stream may be exploited to generate minimal length codewords.

Figure 11 depicts a typical hybrid (block based) video encoder architecture that exploits these three types of redundancies. It is compliant with most of the currently deployed video codec standards (MPEG-2, AVC) and the recently released HEVC standard.

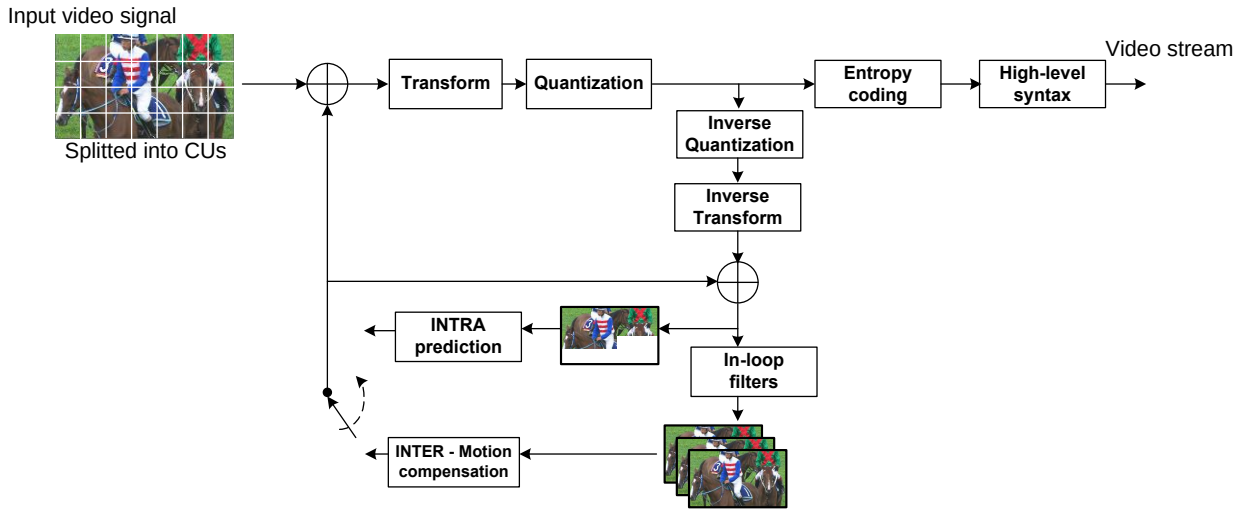


Figure 11: typical hybrid video encoder architecture.

Block based coding

The hybrid video encoders partition the frames into slices and blocks (a.k.a. macro-blocks or coding units) that are encoded consecutively. The color samples of the blocks are not coded directly but are first predicted from surrounding samples or previously decoded pictures, and the prediction residuals are then encoded. These residuals are first transformed using orthogonal basis functions which allow decorrelating the residuals samples values.

Signal prediction: Intra/Inter

Video codecs feature two prediction types (Intra and Inter) depending on whether the block prediction samples are built from previously reconstructed samples of the current frame or of other frames (previously reconstructed and stored in a Decoded/reference Picture Buffer, DPB). In Intra, the neighboring reconstructed samples to the current block are used to build the block prediction, while in Inter, they are copied from a possibly translated (motion vector) and filtered block of same size (except in

case of Adaptive Resolution Coding (ARC) or scalable coding, see also 7.1 and 3 respectively) in the reference picture.

Entropy coding

All the syntax elements (encoding modes, transformed residuals, motion vectors…) are possibly predicted and entropy coded. Commonly used entropy coders are Huffman, Context-Adaptive Variable Length Code (CAVLC) and Context-Adaptive Binary Arithmetic Code (CABAC). In this last case, the syntax element is associated with a context model that provides estimates of conditional probabilities of the coding symbol.

High-Level Syntax

The High Level Syntax (HLS) includes all the pictures headers and above. Generally, it is not coded with CABAC but with fixed or predictable code-words length for better error resiliency. HEVC re-uses the same concept as AVC of Network Abstraction Layer Unit (nalu) that allows partitioning the bit-stream into easily identifiable chunks to facilitate the transport layer multiplexing.

2.1 Presentation of HEVC

In July 2012, we published an early presentation of HEVC (a.k.a. H.265) [30]. After finalization of HEVC, a more detailed description was published by the ISO/IEC MPEG and ITU-T VCEG chairs [31]. The following short overview is inspired from these 2 documents, stressing the new features of HEVC compared to the previous AVC (a.k.a. H.264) standard.

The High Efficiency Video Coding (HEVC) has been developed by a Joint Collaborative Team of ISO/IEC MPEG and ITU-T VCEG (JCT-VC). One main requirement of HEVC was to improve the compression efficiency by a factor of at least two compared to the AVC compression standard, AVC having the best coding efficiency performance among the current generation of video standards. Another particular focus in the design of HEVC was its capacity to address increased video resolution and increased use of parallel processing architectures.

HEVC is a hybrid video codec. Some existing tools from AVC have only been revisited in this standard.

Test conditions

JCT-VC selected a set of 20 progressive sequences of various size and different frame rates, about 10 sec. duration each: 4K (class-A), 1080p (class-B), WVGA (class-C), WQVGA (class-D), 720p (class-E). Three encoding parameters configurations have been considered [55], using 4 reference frames:

- Random Access (RA): hierarchical GOP with B pictures (bi-prediction). This is the typical configuration used for broadcast applications.
- Low-Delay (LD): the last 4 reconstructed frames are used as reference. The Low-Delay-P and Low-Delay-B uses uni-directional and bi-prediction respectively. The low-delay feature is required for applications where the end-to-end delay should be minimal, such as video conferencing for example.
- All-Intra (AI): all the frames are encoded in Intra. This encoding mode can be used by professional encoding applications to facilitate video editing, when the BD-rate performance is not critical.

BD-rate performance

Both objective (PSNR-based) and subjective quality assessments have been performed within the JCT-VC and test results confirmed that the initial goal has been reached:

- In Random Access (RA) configuration, HEVC can provide a bit rate savings around 40% for equal PSNR for the 1080p test sequences,
- The bitrate savings when considering equal subjective quality is even greater, more than 50% for all the test sequences [32].

Concerning UHD-1 test material (3840x2160), HEVC Main and Main 10 Profiles (HM10.0) outperforms equivalent AVC High and High 10 Profiles (JM18.4 with HM-like configuration files) in objective measure (PSNR BD-rate) of around 25% for All Intra (AI), 45 % for Random Access (RA), 45 % for Low Delay B (LB) on average. Besides Main 10 profile outperforms Main profile for 10-bit content coding of around 4% luma for AI, 5% luma for RA, 5% luma for LB on average [33].

Complexity considerations

Despite the intrinsic higher complexity of HEVC, the encoding and decoding computation costs increase seem to stay in reasonable limits given the decade separating HEVC from the previous AVC standard:

- The HEVC reference SW encoding time is increased only by 10% compared to AVC reference SW model [34] and real-time software decoding implementation of HEVC 3840x2048@60Hz have been published [136].

Even if one can argue that the HEVC reference software model is written in a more efficient way than the AVC reference software model, this limited extra computation time is good news to allow real-time

coding and decoding of bigger video formats (bigger picture resolution, higher dynamic ranges and/or frame rates) and rapid deployment in the next few years.

2.1.1 Image partitioning and residuals coding

HEVC has introduced a new partitioning image scheme based on a quad-tree structure (Coding Tree-Unit, CTU) with larger block size – a 64x64 Coding Unit (CU) typically. A Coding Unit can be recursively divided into 4 CUs (Quad-tree). A CU is composed of one CB (Coding Block) of luma samples and 2 CB of chroma samples (size divided by two in both directions in case of 4:2:0).

For transform coding of the prediction residuals, a CU can be further split into smaller square transform units (TU). The splitting is signaled using a second quad-tree, the residual quad-tree (RQT) [35]. RQT allows adapting the transform to the frequency characteristics of the residual signal.

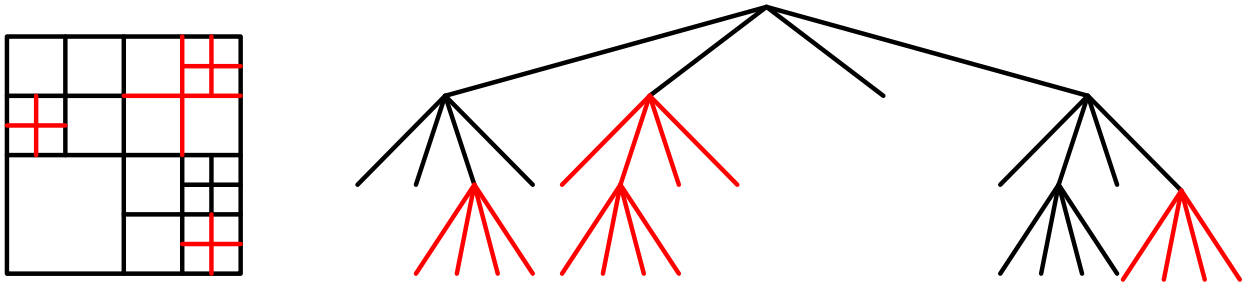


Figure 12: Subdivision of a CTU into CUs [and transform block (TBs)]. Solid lines indicate CB boundaries and red lines indicate TB boundaries. Left: CTB with its partitioning, right: corresponding quadtree.

The Transforms are DCT 4x4, 8x8, 16x16 or 32x32, except for Intra 4x4 that can be DST (Mode Dependent Decoding Transform).

The Coding Unit Quantization Group (CUQG) is specified as a superset of CU for conveying Quantization Parameter (QP) values. Each CUQG is composed of single or multiple CUs, all of which have same QP values. Average QP of left and top CUQG (if available) are used as predictor for current CU quantization parameter (see also chapter 6.1.1).

The significance map (SM) of a TU indicates the positions of non-zero coefficients in the TU. For the largest TU size, a two level structure allows splitting the SM into 4x4 SMs for coding.

In Intra, the scanning order of the coefficients (zig-zag, horizontal or vertical) is selected depending on the mode (Mode Dependent Coefficient Scanning, MDCS).

The first coefficients (levels) are coded using CABAC, next Golomb Rice and the last levels with Exp. Golomb. For each 4x4 coefficients set that satisfies certain condition, the sign bit of the first nonzero coefficient is not coded but later inferred from the parity of the sum of all nonzero coefficients in that set.

2.1.2 Intra coding tools

The number of Intra prediction directions depends on the CU size as depicted in Figure 13: 33 angular modes for 8x8, 16x16 and 32x32, 16 for 4x4 and 4 for 64x64 (CTU). Depending on the direction, the reference sample used for the prediction may be filtered by a 3 taps smoothing filter, and/or some of the reconstructed boundary samples by 2-taps filter.

As for AVC, the DC mode corresponds to the mean value of the samples from both top row and left column and I_PCM allows coding sample values directly. Additionally, a new intra mode has been introduced: the planar mode (depicted in Figure 14).

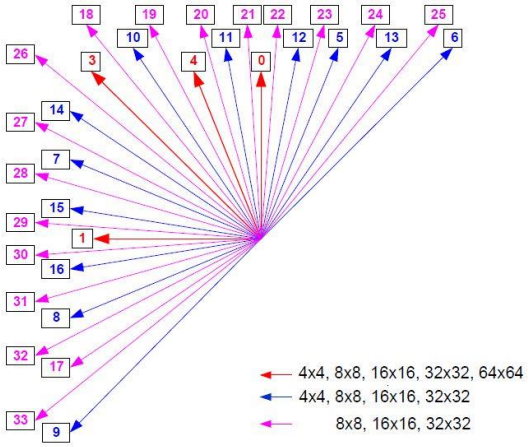


Figure 13: Intra prediction directions, depending on the CB size.

The top and left edges of the Intra DC and some angular predictions are filtered. The number of filter taps depends both on the intra direction and the CB size (Mode Dependent Intra Smoothing).

The decoding of the Intra mode is made in 2 steps. First the 3 Most Probable Modes (MPM) are derived from the neighboring coded CUs. Next, a decoding mode tree allows deriving the intra mode from one binary flag, and 1 or 2 more syntax elements.

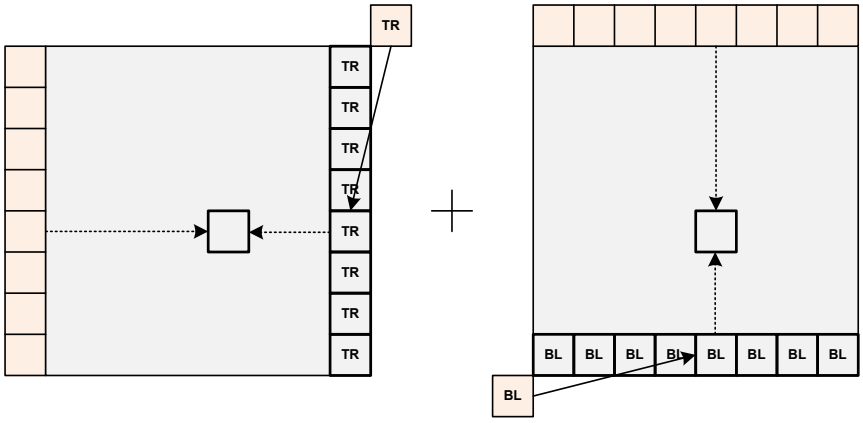


Figure 14: Planar mode: Interpolation indicated by dashed arrow, replication indicated by dotted arrow.

2.1.3 Inter coding tools

A non-split CU is a Prediction Unit (PU) which can be sub-partitioned into 4 square or 2 rectangular partitions (Figure 15). In the same way as AVC, each PU partition is built with uni-directional or bi-prediction motion compensation, using ¼ (luma) or 1/8 (chroma) pel precision motion vectors (*mv*), using filters composed of 7 or 8 taps and 4 taps respectively). But the *mv* values are predicted using motion vector competition: an index corresponding to a list of (spatial and temporal co-located TMVP) *mv* predictors is coded (AMVP).

The modes Merge and Skip (no residuals) enable deriving *mv* and reference index from 1 or 2 neighboring CUs.

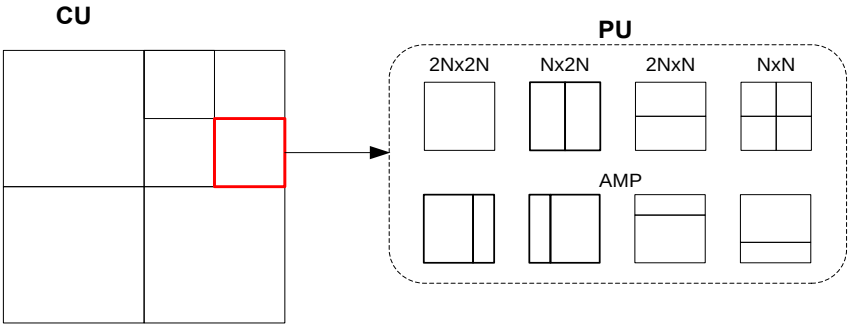


Figure 15: CU and PU partitioning.

2.1.4 In-loop filters

In order to restore the degraded frame caused by compression, two kinds of filtering are applied successively. First a de-blocking filter reduces the blocking artifacts in the same way as AVC at CU boundaries aligned on 8x8 grid basis, except at tile or slice frontiers possibly. The smoothing strength depends on the QP value, on the comparison of some encoding parameters of the current and adjacent block and on the reconstructed sample values difference at the CU boundaries.

Second, the Sample Adaptive Offset (SAO) filter is operated. It allows reducing the distortion by adding an offset to some selected reconstructed samples. To this end, SAO classifies each CTB into 3 categories: no-filtering, band offset (BO) or edge offset (EO). For the 2 last ones, 4 offsets are transmitted. One main purpose of BO case is to reduce the "banding" artefacts, while EO rather reduce ringing coding noise [80].

In case of BO, the pixels are classified into multiple bands where each band contains pixels in the same intensity interval. The intensity range is equally divided into 32 intervals from zero to the maximum intensity value (e.g. 255 for 8-bit pixels). The sample values belonging to four of these bands (which are consecutive within the 32 bands) are modified by adding transmitted offsets, which can be positive or negative, at the transmitted position (Figure 16).

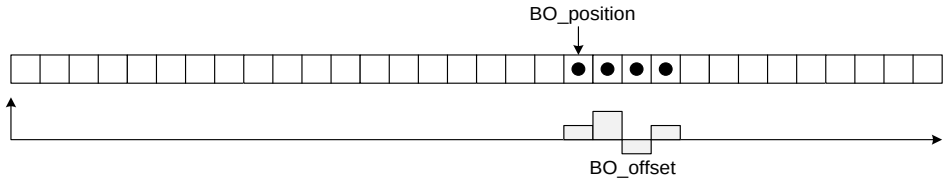
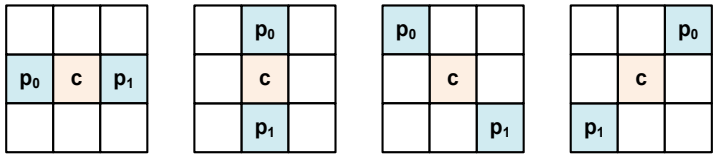


Figure 16: SAO Band-Offset.

Edge offset (EO) uses four 1-D 3-pixel patterns for pixel classification with consideration of edge directional information, as shown in Figure 17.



	Condition
1	$c < p_0$ and $c < p_1$
2	$c < p_0$ and $c = p_1$ or $c < p_1$ and $c = p_0$
3	$c > p_0$ and $c = p_1$ or $c > p_1$ and $c = p_0$
4	$c > p_0$ and $c > p_1$

Figure 17: SAO Edge-Offset patterns and classification conditions.

2.1.5 High-Level Syntax

Even if the High-Level Syntax (HLS) is highly inspired from AVC, it has anticipated the future scalability extension right from the start. Then some tricky patches of the SVC HLS have been avoided in SHVC and the scalability signaling is treated in a more efficient and elegant way (ex: see VPS below).

As for AVC, the video bit-stream is composed of Network Abstraction Units (NALUs) that provides the ability to map the video coding layer (VCL) data onto various transport layers (ex: RTP/IP, ISO MP4, MPEG-2 Systems) and provides a framework for packet loss resilience.

Random Access Points

HEVC defines 3 types of Random Access Points (RAP) corresponding to locations in a bit-stream at which a decoder can begin successfully decoding pictures, without needing to decode any pictures that appeared earlier in the bit-stream. They are useful for channel switching, seek operations, and dynamic streaming services (stream switching) typically (see also stream-switching chapter 7).

As in AVC, the Instantaneous Decoder Refresh (IDR) insures that no subsequent picture in the bit-stream will require reference to pictures prior to the IDR, contrarily to the Clean Random Access (CRA) picture and the Broken-Link Access (BLA) picture that indicates splicing points.

When starting decoding just after a CRA or BLA, one may have pictures that are not decodable because they contain references to pictures that are not actually present in the bit-stream. This occurs typically when the decoding order number (DON) of the reference pictures is less than the decoding order number of the RAP. These pictures are signaled as Random Access Skipped Leading (RASL) (Figure 18, left).

One may have also the case of decodable pictures but their display order (a.k.a. Picture Order Count, POC) is less than the display order of the RAP. These pictures are signaled as Random Access Decodable Leading (RADL) (Figure 18, right).

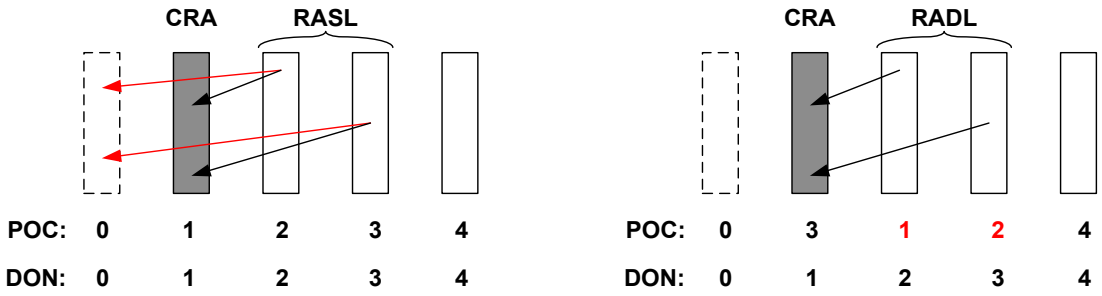


Figure 18: Example of RASL (left) and RADL (right) cases.

Some NAL unit types

In addition to coded slices, Sequence Parameter Sets (SPS) are used to carry data valid to the whole video sequence, whereas Picture Parameter Sets (PPS) carry information valid on a picture-by-picture base. A new NALU has been added in HEVC compared to AVC. The Video Parameter Sets (VPS) contains metadata such as number of layers and dependencies, timing information.

Unlike its predecessors, HEVC focuses on progressive content coding only. However, legacy interlaced content representing still a very large part of the current video traffic, it is provided with a free support (without core-design change) of interlace through the Field Indication SEI message which indicates the source type and some relative information for field applications (the field pictures being encoded as two separate frames).

2.1.6 High-Level Parallelism tools

Slices

In HEVC, it is possible to divide a frame into slices, as in AVC. Slices are groups of CTUs in raster scan order. Slices can be used both for network packetization and for parallel processing. However, a severe penalty on rate distortion performance is incurred when using slices, due to the breaking of all dependencies at their boundaries and to the slice header size, a set of parameters that has to be transmitted at the beginning of each slice. Because of this, new approaches aiming at facilitating parallel processing have been adopted in HEVC, described in the following sections: the independent and the dependent slices segments, the tiles and the wavefront parallel processing.

Slices segments

The slices may be partitioned into slice segments that are a sequence of CTUs, starting with an independent slice segment and subsequent dependent slice segments (if any) that precede the next independent slice segment (if any) within the same access unit.

Dependent slice segments have header syntax elements which can be inferred from the precedent independent slice segment (Figure 19).

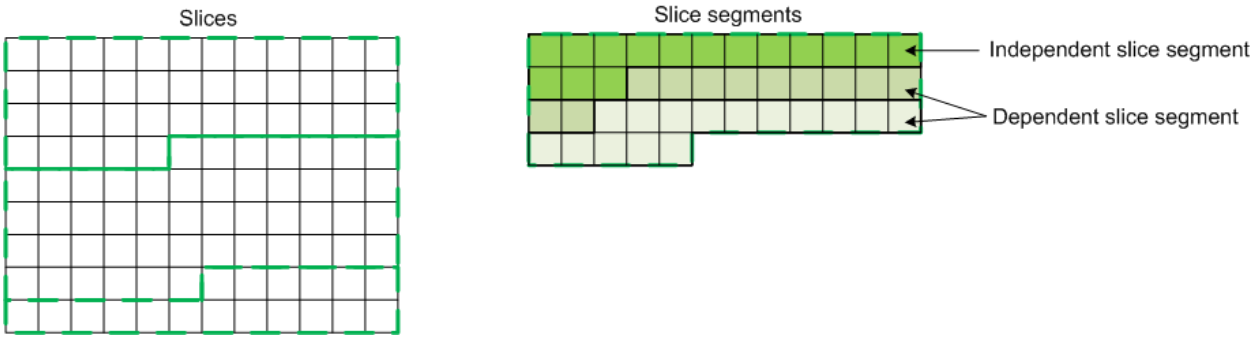


Figure 19: A slice may be composed of (several) independent and dependent slice segments.

Tiles

A frame can be partitioned into a number of independent tiles. Tiles are rectangular groups of CTUs. Tile boundaries are vertical and horizontal and extend across the whole picture. Tiles are processed in raster scan order, as well as the CTUs inside each tile. All dependencies are broken at tile boundaries, so there can be no pixel, motion vector or context prediction across them. The entropy coding engine is reset at the start of each tile. Only the de-blocking filter is applied across tiles, in order to limit visual artifacts. Consequently, tiles can be encoded and decoded by independent cores working in parallel, and only the de-blocking stage requires cross-tile communications. This comes at the expense of rate-distortion loss (about 2.7% loss compared to a non-parallel bit-stream in random access configuration [36]).

Within the same picture, there may be both slices that contain multiple tiles and tiles that contain multiple slices, as depicted in example in Figure 20.

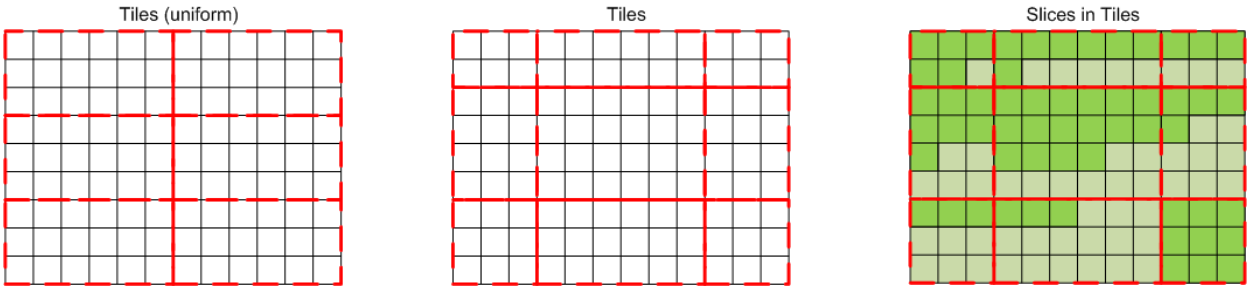


Figure 20: Example of a frame partitioned into tiles (left and middle), with slice segments included in tiles (right).

Wavefront Parallel Processing

Wavefront parallel processing (WPP) is another tool enabling parallel encoding and decoding. WPP consists of resetting the CABAC probabilities of the first CTU in each line with the probabilities obtained after processing the second CTU of the line above (Figure 21). Otherwise, all inter-block dependencies are maintained. Thus, parallel encoding and decoding is possible with moderate BD-rate degradation (around 1.0% compared to a non-parallel friendly bit-stream in random access configuration [36]). Because dependencies are not broken, it is possible to convert a non-parallel compressed video to and from a parallel-friendly one (this is simply an entropy level operation) and conversely. Furthermore, CABAC encoding is flushed after the last CTU of each row, making the bit-stream representing each row of CTU accessible using entry point defined in the slice header. Thus, it is possible to use, in the decoder or in the encoder, any number of cores between one and the number of CTU rows in the frame.

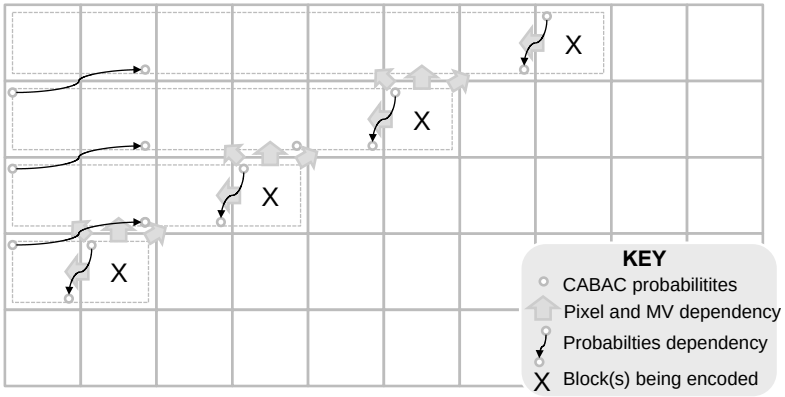


Figure 21: Wavefront Parallel Processing.

2.1.7 HEVC Extensions

From its early starting, the JCT-VC scheduled a set of key steps in the process of defining a new video standard which would target several types of applications and market needs. Each of these steps leaded to the definition of particular HEVC extension tools specially designed to support specific requirements. The list of the HEVC extension activities is listed below with the corresponding intended use-cases and the technical characteristics.

Still pictures

Since the advent of Internet and the success of the numerical cameras (possibly embedded in mobile phones), the still pictures are undoubtedly the most popular and most common multimedia format. Right from 2012, it appeared the HEVC intra coding tools were competing very efficiently with the widely deployed JPEG and the JPEG-2000 standards, in particular for large pictures [1]. This trend was confirmed in many studies afterwards, showing subjective improvement from 15% to 40% compared to JPEG2000 or JPEG [2].

Range Extension

This activity covers the other than 4:2:0 chromaticity formats (4:2:2 and 4:4:4) and higher bit-depth (12, 14 or even 16 bits) basically. The HEVC Range Extensions (Rext) can be seen as the natural evolution of the HEVC Main and Main10 profiles, targeting traditional professional applications (content production/mastering, TV contribution) where higher signal accuracy is required to support various frame processing. But Rext may also be used by new applications such as medical imaging (up to 16-bits) and screen content coding which might find application in remote desktop services and could be a response to the increasing end-user display resolution (e.g. 4K screens) needing high-speed and high-bandwidth wire/wireless connectivity.

Scalable video coding

The encoded video bit-streams, as such, are something quite static and not adaptable basically. The encoding parameters choices such as bit-rate, frame size, quality… are set once and for all. One can dedicate one real-time encoder for every end-user, but this approach is not feasible in case of very large number of clients such as TV broadcast or broadband services. An alternative approach is to distribute several instances of the same content, each instance being encoded with different parameters. However, the storage amount and/or the required bandwidth are increased proportionally to the number of the provided streams instances.

The scalable video coding technique avoids duplicating the data in as many video streams as video formats or encoding parameters, by exploiting the levels of redundancy that may exist in between different reconstructed video frames. The video coding process is split in several layers, with hierarchical dependencies. Each layer corresponds to a video stream, targeting one particular video representation or characteristics. At the decoding side, the reconstructed information from some lower (dependent) layers is used to improve the coding of the upper layers in the hierarchy/dependency tree.

Multi-view video coding

The main purpose of the Multi-view coding (MV-HEVC) is to support 3DTV applications. It extends the video scalable tools to predict one view from another one, generating as many dependent streams as views. Additionally, depth maps may be encoded using monochrome auxiliary pictures framework (inter-layer prediction disabled). Alternatively, one may use Frame Packing Arrangement Supplemental Enhancement Information message to transport multiple views in one single frame.

Conclusion

The edition 2 of HEVC (Final Draft International Standard, a.k.a. FDIS) [133] has been finalized in august 2014. It includes REext, SHVC and MV-HEVC extensions as annexes to the HEVC first edition. Further editions may be released to support Screen Content Coding and additional profiles if required.

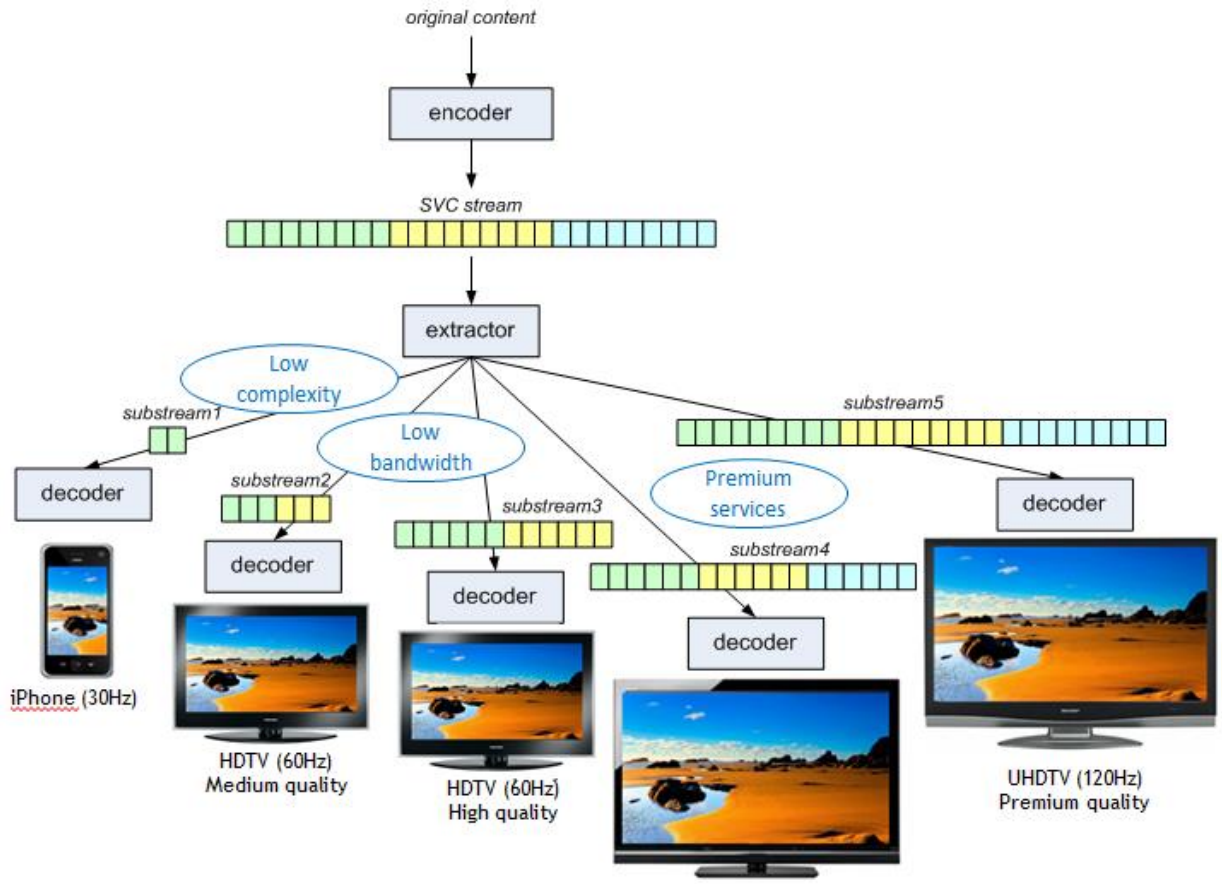


Figure 22: video coding scalability.

2.1.8 Conclusion

Overall, the complexity of HEVC decoders does not appear to be significantly different from that of AVC decoders. Because if the complexity of some key modules such as transforms, intra picture prediction, and motion compensation is likely higher in HEVC than in AVC, complexity was reduced in others such as entropy coding and de-blocking. This makes HEVC decoding in software very practical on current hardware [136].

Software-based HEVC decoder capabilities (published by NTT Docomo):

- Single-threaded: 1080p@30 on ARMv7 (1.3GHz),1080p@60 decoding on i5 (2.53GHz)
- Multi-threaded: 4Kx2K@60 on i7 (2.7GHz), 12Mbps, decoding speed up to 100fps

However, HEVC encoders are expected to be several times more complex than AVC ones. Indeed, HEVC incorporates many more mode combinations as a result of the added flexibility from the quadtree structures and the increase of intra picture prediction modes. This will be challenging for the real-time implementations, in particular for UHD (4K) or High Frame Rate (HFR) video coding.

3. Presentation of HEVC Scalability: SHVC

Basically, the co-existence of multiple formats for video distribution, such as UHDTV 3840x2160 and HDTV 1920x1080, coupled with an increased level of network heterogeneity, makes scalable video coding a suitable choice for delivery because it offers storage and bandwidth efficiency.

The developments of HEVC extensions for scalable video coding (SHVC) started in July 2012 with a joint ISO/IEC MPEG and the ITU-T VCEG (JCT-VC) call for proposal. The Final Draft International Standard (FDIS) was finalized in July 2014.

SHVC architecture overview

SHVC supports up to 64 layers. The Figure 23 depicts the SHVC decoder architecture for two layers: a Base Layer (BL) and an Enhancement Layer (EL). When necessary, appropriate interlayer processing is applied to the reconstructed BL picture to obtain the Inter-Layer Reference (ILR) picture. The ILR picture is then put into the EL Decoded Pictures Buffer (DPB) as a reference picture and used with the EL temporal reference pictures for enhancement layer coding. However, since the ILR is temporally co-located with the current EL frame, SHVC imposes a zero-motion constraint when computing the Inter-Layer Prediction (ILP) with the ILR picture.

All block-level logic of the EL codec remains the same as a single-layer HEVC codec. Any necessary changes to the EL decoder, indicated as HEVC* in Figure 23, are allowed only at the High-Level Syntax that is slice headers and above. This has the advantage to reduce the implementation costs of SHVC codecs potentially, because existing design of HEVC codecs can be maximally reused.

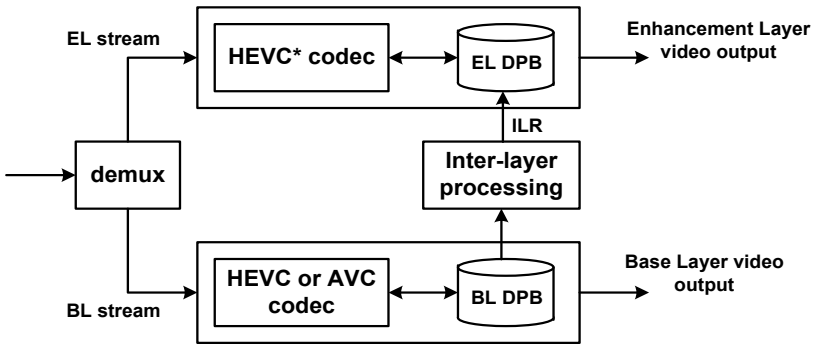


Figure 23: SHVC decoder architecture.

In the hybrid scalability case, the base layer in Figure 23 can operate as a black box, the scalable coding of the EL needing only the reconstructed BL pictures. In this case, the BL stream can be AVC for example.

Inter-layer processing in SHVC

In the case the BL and EL frames have different size, up-sampling filters (8-tap for luma, 4-tap for chroma) are applied on the reconstructed BL. They support arbitrary spatial ratio as well as scaled offsets to adjust the spatial relationship between the scaled BL and the EL picture.

Temporal Motion Vector Prediction (TMVP) is a new coding tool in HEVC used to improve motion vector coding (see 2.1.3). To enable TMVP, the motion field of the temporal reference pictures is stored in the DPB at 16x16 block based resolution. In SHVC, the BL picture's motion field is spatially re-sampled to produce the "mapped" motion field for the ILR picture.

If the colorimetry between EL and BL frames is different, SHVC uses a special color-mapping process to derive ILR from the reconstructed BL frame: the Color Gamut Scalability (CGS). The genesis of the CGS

is presented in the following section 0 and the work we carried out in this domain are detailed in the section 8.

SHVC performance

The performance of SHVC has been evaluated for a set of selected reference Low Quality (LQ) and High Quality (HQ) test sequences. The HQ frames can have same size (SNR), 1.5x or 2x the LQ size horizontally and vertically. In case of SNR, the quantization steps parameters (QP) used for the HQ streams (EL) are lower than for the BL. The BD-rate values are estimated using Bjontegaard [23] polynomial interpolation through (PSNR, rate) points obtained with several QPs. These QPs have been selected by JCT-VC to be in the "knee" of the BD-rate curves.

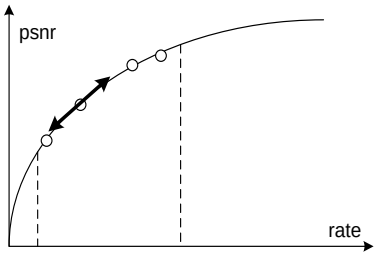


Figure 24: BD-rate curve with "knee" part in-between the dashed lines.

The Table 1 and Table 2 give the relative BD-rate gain obtained with SHM1.0 in two cases: all frames are encoded in Intra (All-Intra, AI) and using hierarchical GOP (Random-Access, RA) configurations respectively.

In the first row, the SHVC streams (BL stream + EL stream) are compared to coding the HQ frames only with HEVC. The positive values mean that the sum of the BL and EL bit rates is superior to the HQ single layer stream for same HQ frames quality. It figures out the supplemental cost for offering the scalability feature (HQ and LQ services) compared to providing HQ service only.

In the second row, the SHVC streams are compared to the simulcast case where two independent HEVC streams are used for the LQ and the HQ. The negative values means the sum of the BL and EL streams is inferior to coding LQ and HQ stream separately, for same LQ and HQ quality. It figures out the relative gain of SHVC when distributing both HQ and LQ video compared to simulcast.

Table 1: SHVC RD performance in AI configuration compared to HEVC single layer HQ and simulcast (BL + single layer HQ).

SHM1.0	AI HEVC 2x			AI HEVC 1.5x		
	Y	U	V	Y	U	V
(BL+EL) vs single layer HQ	12.90	15.25	14.70	10.50	9.58	8.94
(BL+EL) vs simulcast	-20.91	-19.29	-19.30	-32.56	-32.97	-33.36

Table 2: SHVC RD performance in RA configuration compared to HEVC single layer HQ and simulcast (BL + single layer HQ).

SHM1.0	RA HEVC 2x			RA HEVC 1.5x			RA HEVC SNR		
	Y	U	V	Y	U	V	Y	U	V
(BL+EL) vs single layer HQ	20.17	34.55	35.58	17.39	29.13	31.67	16.17	29.23	34.01
(BL+EL) vs simulcast	-15.82	-7.80	-6.10	-27.11	-19.85	-18.32	-20.68	-11.19	-6.50

3.1 Requirements in Color Gamut and Bit-Depth Scalability

The Color Gamut Scalability (CGS) is an innovative new feature of SHVC that did not exist in previous scalable video coding MPEG standards. In this section, we recall the new market and technical requirements that have justified the study of the CGS technology.

3.1.1 MPEG Requirements

MPEG has identified typical use cases where changes in resolution in digital video distribution often result in a change in color space [3][4]. For example, in a digital TV application that includes the support of standard definition (SDTV) and high definition (HDTV) content, SDTV may be represented with color primaries specified in Rec.601 [2], whereas HDTV may be represented with color primaries specified in Rec.709 [3]. Considering these applications, it is desirable to have a scalable capability with multiple coding layers in different color spaces to support standard definition and high definition services simultaneously and efficiently.

Another relevant digital TV application is that of extending the resolution beyond current HD resolutions in a backwards compatible way. For example, in some scenarios, it is desirable to keep picture aspect ratio by extending resolution from 1920x1080 to 3840x2160 such that the Base Layer (BL) has a 1920x1080 sampling resolution and Enhancement Layer (EL) has a 3840x2160 sampling resolution. In these cases, the HD resolution version is represented with Rec.709 color primaries to provide backward compatibility. However, enhancement layer may be represented with color primaries that provide a larger color gamut such as Rec.2020.

Consequently, the following three basic requirements with regards to color space were established during the development of Scalable High-efficiency Video Coding standard extensions (SHVC):

- Support base and enhancement layers with different color spaces
- Support a combination of color space scalability, spatial scalability and bit-depth scalability
- Support tri-stimulus values in the enhancement layer that cannot be represented in the base layer color space

In JCT-VC, an Ad Hoc Group dedicated to CGS [19] was created to study coding tools that may satisfy these requirements.

3.1.2 DVB Requirements

With the arrival of the new UHD video signal format Rec.2020 [5] for television broadcasting with color gamut and resolution significantly wider than existing formats (SDTV, HDTV), DVB-UHDTV Commercial Module group has foreseen 3 steps for the deployment of UHDTV services as depicted in Figure 25 [139].

1. UHD-1 Phase-1: The devices should support the decoding of HEVC Main-10 profile up to 3840x2160p 50/60 Hz, with color space signaling of both Rec.709 and Rec.2020 specifications. However, the support for rendering of Rec. 2020 is optional: a device may decode the frames but display them in a best effort mode, i.e. potentially with wrong colors.
2. UHD-1 Phase-2: The devices should support the decoding of HEVC Main-10 profile up to 3840x2160p 100/120 Hz. In case of Rec. 2020 color space, the end device should support the rendering of the full gamut or perform appropriate color mapping corresponding to the display characteristics. This may be achieved by mean of color metadata.

[2] ITU-R Rec. BT.601 .
[3] ITU-R Rec. BT.709 .

3. UHD-2: The support of 7680x4320p at 100/120 Hz is considered for specific applications (e.g. wall screens, digital signage).

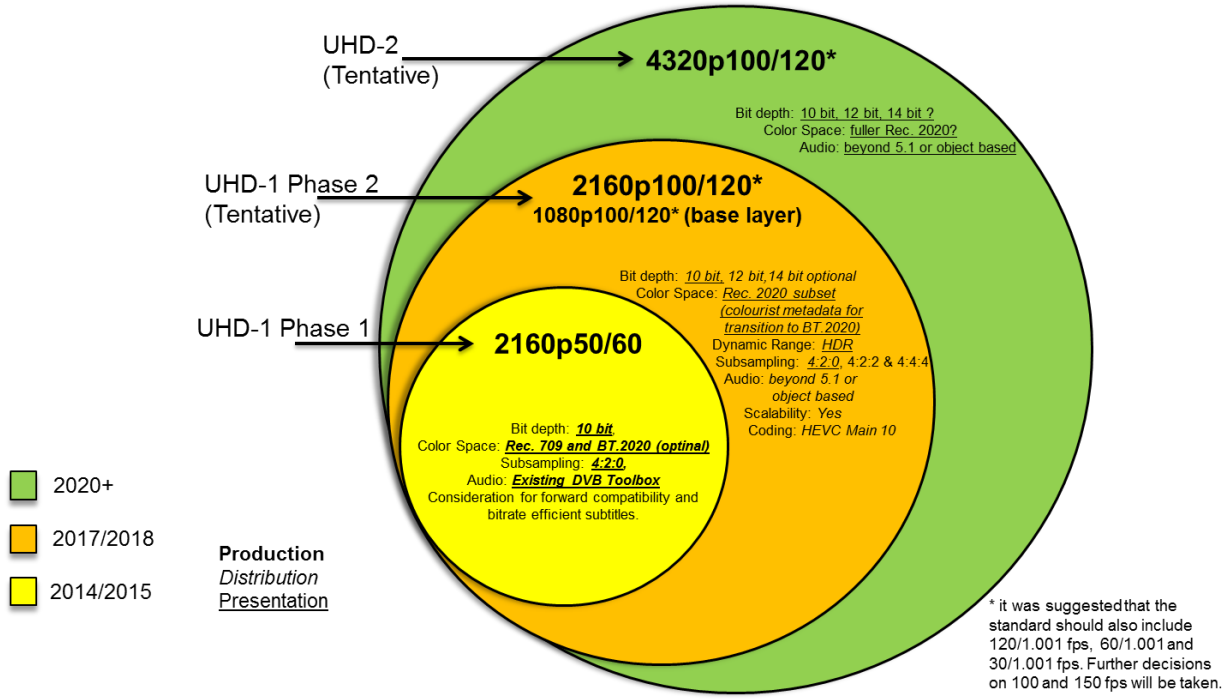


Figure 25: DVB proposed a 3-steps approach for the deployment of UHDTV.

Scalable video coding is a natural candidate for UHD-1 Phase-2 since it provides de facto backward compatibility with UHD-1 Phase-1 and former DVB services (e.g., SHVC streams may contain HDTV AVC-coded base layer stream, as SHVC supports hybrid scalability which allows non-HEVC coded base layer stream).

In this context, the CGS use case presented to the JCT-VC group for evaluating and defining appropriate SHVC tools [6] is a 2160p Ultra HDTV service deployed over 1080p HDTV service. In this case, HDTV video is represented with Rec.709 primaries whereas UHD-1 video is represented with Rec. 2020 color encoding system, and color grading of the UHD-1 video is performed with a reference monitor whose gamut is a subset of Rec.2020 gamut but beyond Rec.709 gamut. This use case was chosen as the SHVC color gamut scalability test conditions.

Part II – How to better adapt Video Coding to the Signal properties?

4. Structuring frames for coding

We have seen in section 2 that all the standardized solutions first partition each picture in slices, next in fixed square blocks (macro-blocks in AVC and Coding Tree Units for HEVC) and further in coding units following a quadtree structure (in HEVC).

This partitioning is a very important feature in video coding since it structures the whole encoding process. The rationale behind the idea of partitioning is to group areas sharing the same characteristics, such as texture or motion. One benefit is to factorize the coding of homogeneous regions: same prediction (ex: Intra direction, motion, references), same parameters (ex: quantizer):

- One can group the prediction process all in one single process for a whole region. This may reduce the memory bandwidth access and homogenize the coding artefacts.
- One can signal the common parameters only once, then reducing the overhead.
- Another advantage is to reduce boundary artefacts since adequate partitioning can better fit with the actual motion boundary.

When partitioning is combined with a “Merge mode” function as in HEVC, it leverages its efficiency because it allows propagating parameters and to describe more complex region shapes rather than simple geometrical ones (such as squares or rectangles).

At last, when combined with other modes selection/adaptation such as transform type or size, or coefficients scanning, the R-D gains can be significant.

The first chapter of this section considers a partitioning based on parameterized shapes that was proposed in JCT-VC and that allows increasing the coding efficiency significantly compared to the existing partitions, at the expense however of an increased encoding complexity. Then we propose two methods to reduce this encoder extra complexity.

In the second chapter of this section, the spatial partitioning paradigm is further extended to the temporal dimension. Such a spatio-temporal approach is exploited via the use of motion trajectories and meshes (see 4.2). We show how the resulting modeling of the optical flow in the scene can provide efficient (long-term) motion segmentation. Its application to video coding provides encouraging results for low bit rates long term prediction cases.

4.1 Geometry-Adaptive block partitioning

Traditional state-of-art partitioning techniques are based on Tree-structured-Variable Block Size (TVBS) such as Quad-tree with square or rectangular basic shapes, using a split flag signaling recursively the iterative tree division process (ex: HEVC).

At low bit-rate, the Pattern Video Coding (PVC) can compete with TVBS [42]. In HEVC, a study group was in charge of looking at several Flexible Motion Partitioning techniques. One of the techniques explored was the Geometry adaptive block partitioning (GEO). In this section, we present and discuss some fast encoding algorithms for GEO [58].

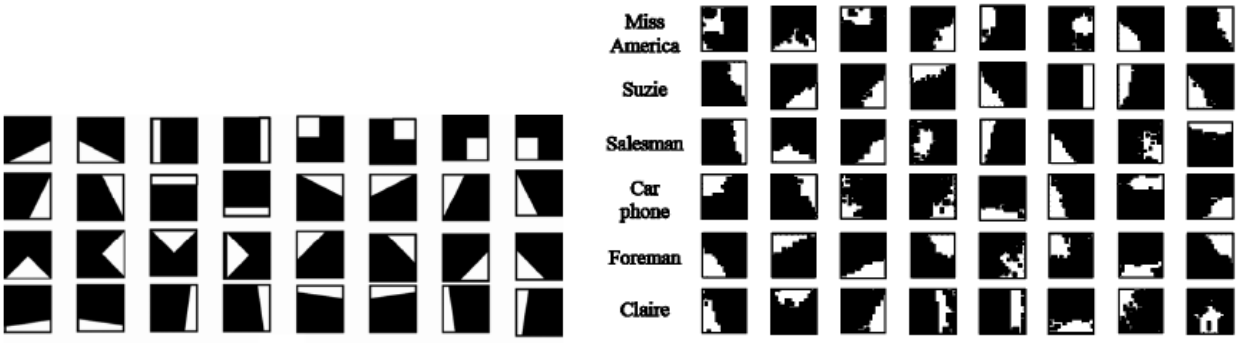


Figure 26: examples of patterns used for PVC. Left: pre-defined pattern, right: Content based Pattern.

4.1.1 Introduction/context

The general concept of Geometry-adaptive block partitioning (GEO mode) has been explored for several years to improve the motion segmentation in video coding [43][44][45][46]. In addition to the classical horizontal and vertical motion partitions used in state-of-the-art video coding schemes, GEO mode consists of another kind of motion partition. This motion partition divides the block into 2 regions potentially not rectangular. The edge separating the 2 regions is defined by a parameterized straight line defined in polar coordinates (see Figure 27). One set of motion data (inter prediction type, reference picture indices, motion vectors) is then associated to each region. Thanks to this more accurate partitioning, it is possible to better fit the actual motion frontiers and therefore to reduce the energy of the motion compensated prediction residuals in particular near these frontiers.

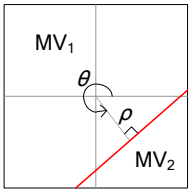


Figure 27: Parameters defining a geometry motion partition.

While offering improved visual quality and increased rate-distortion (RD) performance, this tool potentially involves a significant increase of encoding complexity due to the important number of new partitioning modes to be tested. For instance, when tested in the HEVC test model software and with the tests conditions used in JCT-VC [47][55], 5% average bit-rate reduction is reached with an encoding time multiplied by 20 [52]. Let's nevertheless point out that the decoder complexity is slightly impacted.

This huge encoding complexity cannot be practically handled in most of the video coding applications and some efforts are required to develop simplified algorithms. A number of works have been focused on this goal. In [48][49], the GEO partitioning line is predicted based on spatial edge detection. A refinement step is then applied around the detected spatial edges. In [50][51], based on the relative statistical RD importance of the various adaptive-shape modes, a limitation of the set of considered modes is proposed.

In the next chapters, these solutions are further investigated. First, we describe the simplified algorithm implemented in the reference software and that is the basics of our developments. Next, we propose two alternative methods that are compared: the first approach is based on early termination strategies and reduction of the number of considered modes, the second one is based on spatio-temporal edge detection. In both cases, the goal is to limit the number of GEO modes to be tested at the encoder, and to avoid such tests when GEO is of no use. The algorithms are tested on top of the HEVC test model software (called TMuC) [47].

4.1.2 GEO implementation in HEVC

As in [46], the implementation of GEO in the TMuC uses at each block size 32 different values of θ (from 0 to 360° in steps of 11.25°, i.e., $\Delta\theta = 11.25°$). When considering $\Delta\rho = 1$, this leads to 256, 512, and 1024 possible geometry partitions, for 8x8 16x16 and 32x32 block sizes, respectively. The number of supported partitions can be relatively big.

In order to reduce the encoding complexity, a simplified search algorithm exploits the motion vectors previously computed when testing the square, rectangular and further splitting partitioning modes [46]. In TMuC, a coding unit can be recursively split into 4 sub-units until a given block size (practically 4x4), as illustrated in the split example of Figure 28 (the AMP partitioning was not part of TMuC but added later on HM, see 2.1.3). Prior to testing the GEO modes, the square 2Nx2N, NxN, rectangular 2NxN and Nx2N, and split CU partitioning modes are first tested.

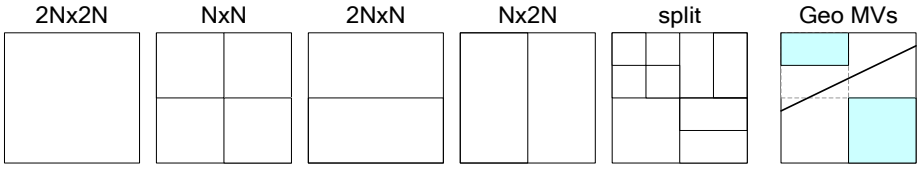


Figure 28: square and rectangular partitioning modes.

Then GEO modes are evaluated as follows. First, fast geometry mode decision pre-selects the K best modes. For each partition region, the largest rectangular block previously tested and entirely inside the GEO region (Figure 28, right) is used to compute the Sum of Absolute prediction error Difference (SAD) over this partition. The K partitioning modes with the minimum SAD are selected with their associated motion data. Then the best mode among these K candidates is selected based on full RD computation. Practically K is set to 8. This algorithm is the basis of the simplified versions proposed in the sequel.

4.1.3 Early termination strategies

In [49][50][51], a strategy based on a statistical analysis of the actually used GEO modes was proposed. The most valuable partitions are identified, which results in a significant reduction of the number of GEO modes to be considered at the encoder and at the decoder. A first implementation of this strategy has been made in the TMuC [50], showing an encoding complexity reduction of around 25%, but with a non-negligible decrease of the RD performance. Possibly, a less harsh selection is required to limit the coding efficiency loss. Finest strategies, described in the following subsections, have therefore been evaluated.

4.1.3.1 Early termination strategy

As conventional rectangular modes are tested first, their corresponding RD cost can be used to select the cases when GEO has to be further tested or not. If the RD cost (Lagrangian Rate-distortion [54]) of the 2Nx2N partitioning (including the Skip mode) is low compared to the other partitioning modes, it can be expected that further block splitting is most often not necessary.

Practically GEO is tested when the following condition is true:

$$R = \min(C_{NxN}, C_{2NxN}, C_{Nx2N}, C_{split}) / C_{2Nx2N} > \mu \qquad (10)$$

with C_P corresponding to the RD cost of partitioning mode P (*NxN, 2NxN, Nx2N, 2Nx2N, split*). To derive the ratio parameter μ, an analysis over different sequences and QPs has been performed, using a systematic test of GEO (i.e. test (10) disabled). For each picture, the maximum value μ_{max} of the ratio R is computed over all the blocks for which GEO is the best mode. An optimal value of μ is then computed as the average of μ_{max} over the pictures of a same sequence class (as define in HEVC test conditions) and same QP. Figure 29 depicts the parameter μ per QP value and sequence class, for block size of 32x32. It can be first observed that this parameter should depend on QP. Then it appears that with a μ value corresponding to the maximum trend line depicted in Figure 29, test (10) would almost never result in not testing GEO though this mode would have been the best one. Similar trends are observed for other block sizes. Therefore μ is defined as μ = a+b.QP with a and b being empirically defined per block size based on the observed data similar to those depicted in Figure 29. Practically, a is set to 0.98 and b to 0.004.

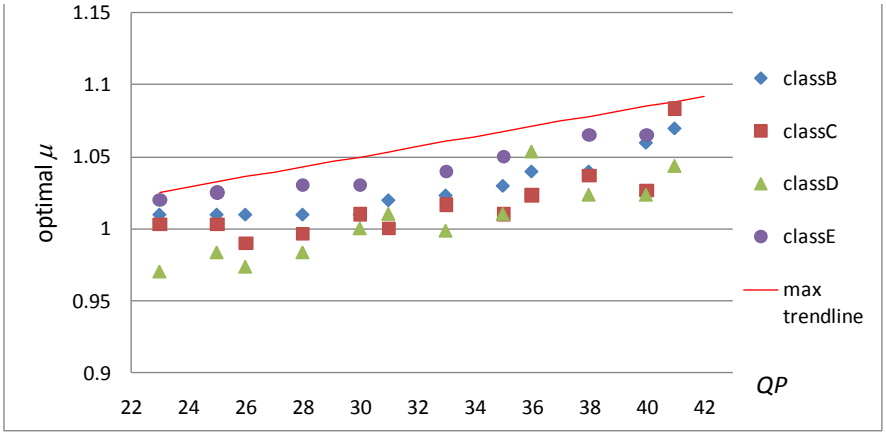


Figure 29: optimal ratio parameter μ per QP value.

4.1.3.2 Number of pre-selected candidates

The other demanding task of GEO evaluation is the final RD optimization applied to the K pre-selected candidates. Figure 30 depicts the relative total RD cost increase when considering only M candidates (M=1 to 7) over 8. For instance, keeping only 7 candidates generates an increase of the total RD cost of less than 2%. It can be first observed that the trend is independent on the QP value. Similarly, it appears that the trend is consistent over the different considered sequences. Practically, keeping 6 or 7 candidates allows saving most of the GEO coding efficiency.

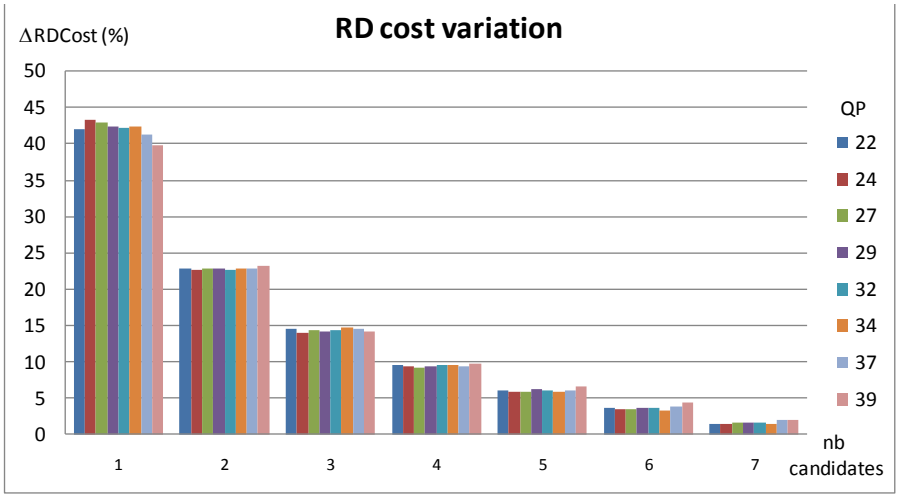


Figure 30: RD cost impact of the number of GEO candidates.

4.1.4 Spatio-Temporal Edge Detection

GEO modes pre-selection can also be based on texture analysis. This solution can be complementary to the early termination strategies previously described. In [48][49], the pre-selection starts with spatial edge detection. Next, in [48], a linear regression is applied to find the most probable line edge. Alternatively, [49] applies a Hough transform. However, these methods are not very accurate or computationally costly. In addition, working on spatial edges may generate an over-detection of potential candidates. Indeed spatial edges are most often inside homogeneous motion regions and do not require splitting a block into two partitions. In order to reduce this over-detection, the pre-selection approach proposed in section consists of a spatio-temporal edge detection. Only non-static edges are considered as potential candidates for GEO.

The process works in the following steps. First the temporal frame difference is computed as follows:

$$S(x,y) = 2 \times I_{cur}(x,y) - I_{prev}(x,y) - I_{next}(x,y) \qquad (11a)$$

with S being the temporal difference signal, I_{cur}, I_{prev}, and I_{next} being respectively the current, the previous and the next frames.

Next, contrarily to [48][49], no explicit edge detection is applied. For each θ, the histogram H(θ) of accumulated projected gradients obtained with Sobel operators is built (see in Figure 31). The sum of gradients values is weighted by their standard deviation $\sigma_{\rho,\theta}$ along the line (ρ,θ), as follows:

$$H(\theta,\rho) = \left(1 - \sigma_{\theta,\rho}\right) \times \sum_{(x,y)\in\rho(\theta)} G(x,y) \qquad (11b)$$

where: $H(\theta, \rho)$ is the histogram value for bin ρ,
$G(x,y)$ is the gradient of $S(x,y)$ of (11a),
$\sigma_{\rho,\theta}$ is the standard deviation of G for the points (x,y) belonging to the line (ρ,θ).

This weighting intends to favor continuous edges and to reduce false detections due to textured regions. The maxima of the full histograms correspond to the most probable GEO candidates. Possibly a global motion compensation could be applied prior to the edge detection in order to detect only moving edges related to the scene background. In the current version this option is not used.

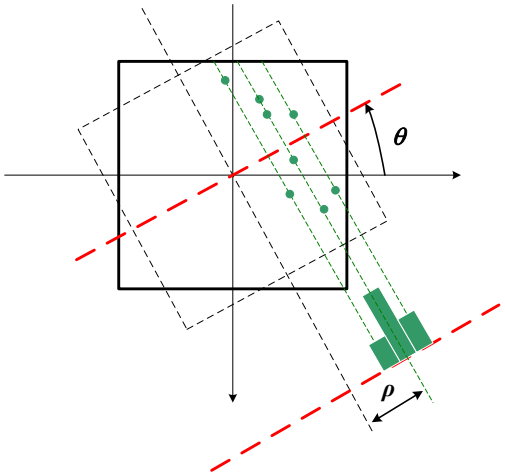


Figure 31: directional projection histograms.

4.1.5 Simulation results

Different algorithmic versions have been evaluated based on the HEVC reference software TMuC (version 0.9) using the test conditions specified in [55]. 15 sequences among 4 sequence classes are

considered (A: cropped areas of size 2560x1600, B: 1080p, C: WVGA, D: WQVGA). Encoding structure is hierarchical B pictures with GOP size of 8. GEO applies to 16x16 and 32x32 blocks. For larger block sizes, the gain appears to be minor considering the increased encoding complexity. The reference anchors are generated by TMuC, which supports symmetric rectangular partitioning (a square block is divided into two rectangles of equal size).

First, the full complexity GEO scheme (denoted FGEO), consisting in an exhaustive RD evaluation of all the available GEO modes, has been tested. FGEO gives 5% (with peaks of 10%) BD-rate reduction [56] (Table 3) but the encoding time is multiplied by 20 compared to the TMuC. This shows the high potential of this coding tool, but also demonstrates that encoding complexity is definitely a key issue.

Table 3: FGEO BD-rate gains (figures above 5% are highlighted in red).

Class	sequence	Random Access			Low Delay		
		Y(%)	U(%)	V(%)	Y(%)	U(%)	V(%)
A	PeopleOnStreet	-6.2	-5,3	-6.2			
	Traffic	-7,5	-10.0	-8,6			
B	Kimono	-1,1	-1,8	-1,8	-1,3	-1,5	-1,5
	ParkSene	-5,1	-5,7	-5,8	-3,7	-5,7	-5,9
	BasketballDrive	-2,9	-3,7	-3,8	-3,5	-4,0	-4,6
	BQTerrace	-2,9	-4,4	-5,0	-2,1	-3,2	-3,2
	Cactus	-3,1	-5,2	-6,2	-2,4	-4,6	-5,1
C	RaceHorses	-5,0	-7,3	-7,8	-5,0	-6,9	-8,7
	PartyScene	-8,7	-9,8	-10,5	-6,4	-9,0	-9,3
	BQMall	-3,8	-4,4	-5,1	-2,9	-4,2	-3,9
	BasketballDrill	-8,4	-10,8	-11,3	-5,8	-6,9	-7,4
D	BasketballPass	-3,7	-4,0	-4,5	-4,6	-5,1	-5,7
	BQSquare	-2,8	-3,0	-3,8	-3,3	-3,4	-5,9
	BlowingBubbles	-4,7	-5,7	-4,8	-4,5	-5,3	-5,3
	RaceHorses	-9,0	-9,0	-9,8	-6,5	-8,1	-7,1
E	Vidyo1				-5,3	-5,8	-5,3
	Vidyo3				-7,7	-9,0	-6,1
	Vidyo4				-4,4	-5,4	-4,9
	Total average	**-5.0**	**-6.0**	**-6.3**	**-4,3**	**-5,5**	**-5,6**

Four simplified versions have been compared (Figure 32):

- AMP (Asymmetric Motion Partitioning) [57], which consists of splitting square blocks into non-symmetric rectangular partitions.
- SGEO1, corresponding to the initial simplified GEO algorithm described in chapter 4.1.2.
- SGEO2, corresponding to the early termination algorithms of chapter 4.1.3.
- EDGE, based on the spatio-temporal edge detection as described in chapter 4.1.4.

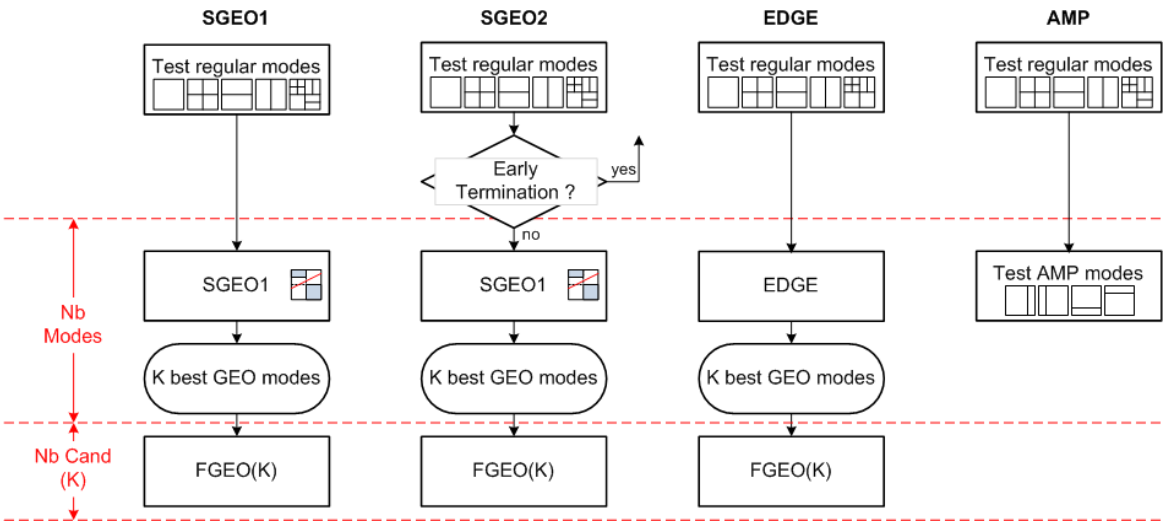


Figure 32: synoptic of the four simplified algorithms.

Table 4: characteristics of the different tested algorithms.

Algo	Nb Modes 16x16	Nb Modes 32x32	Nb cand.	Early termination	Edge Detection
AMP	4	4	4	No	No
SGEO1	292	494	8	No	No
SGEO2	122	127	7	Yes	No
EDGE	292	494	3	No	Yes

Table 4 provides the main parameters of these different solutions. The trade-off encoding gain / complexity is highly dependent on the number of RD optimization candidates.

Figure 33 depicts the encoding time as a function of the average BD-rate gain. The most efficient solution is SGEO1, with a coding gain close to 3%, but the encoding time is increased by 90%. SGEO2 performs better than AMP and EDGE with a lower complexity. It can also be observed that using a strategy only based on edge detection is not optimal. One reason is that actual GEO partitions can be selected in blocks where no strong texture frontier is present (see Figure 34). Other parameterizations of the EDGE algorithm lead to similar trends.

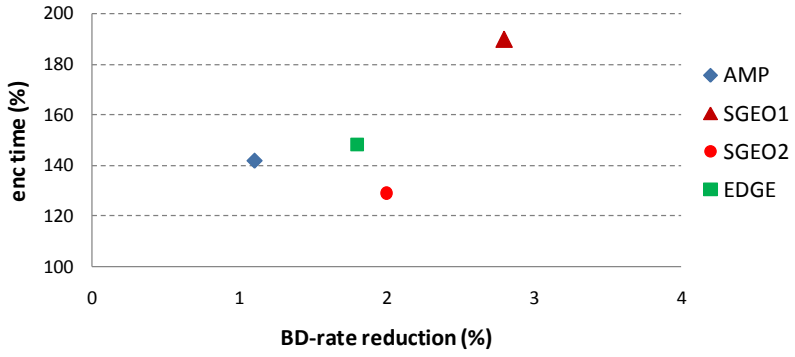


Figure 33: coding gain vs efficiency.

In addition, GEO mode is mainly selected in presence of slow motion and rich textured areas, while conventional rectangular modes are most often optimal in terms of RD optimization in low textured areas.

Finally SGEO2 appears to be a good compromise with a gain of 2% for an encoding time increased by 30%.

Figure 34: example of usage of GEO modes.

4.1.6 Conclusion

In this section, two different low-complexity geometry-adaptive block partitioning (GEO) encoding algorithms have been studied and compared with other existing techniques. They correspond to two approaches: early termination strategies, coupled with controlling the number of tested GEO modes, and fast edge detection algorithms based on spatio-temporal texture analysis. The obtained results show that an edge detection based approach is not necessarily the optimal one, and that more texture agnostic algorithms may be more appropriate to catch motion boundaries.

Among all the different considered versions, the algorithm limiting the number of GEO candidates based on early termination strategies offers a good trade-off coding gain/complexity.

4.2 Long-term motion modeling with trajectories

Efficient representation of motion between frames is useful in many video processing applications such as video analysis, object tracking, image pre- and post-processing or video compression. Motion modeling using point trajectories concept has been proposed recently. This section proposes a low-complexity point trajectories building method based on variable size block tunneling. A grouping step is further applied to merge similar trajectories, next modeled using DCT basis functions. The potential of the proposed approach is illustrated in two application domains: motion segmentation and video coding.

4.2.1 Introduction/context

One of the important exploration work on motion analysis in video is focusing on trajectories estimation, segmentation and modeling. These processes are often performed using feature tracking or optical flow estimation. Feature tracking follows a sparse set of salient image points over many frames, whereas optical flow estimates a dense motion field from one frame to the next. Sand et al proposed to combine the two approaches to produce motion estimates (particles video) that are both spatially dense and temporally long-range [59]. This concept of point trajectories associates for any image point, a corresponding scene point appearing in all other video frames (until the point leaves the field of view or becomes occluded).

In this section, we present a robust and low-complexity algorithm for building dense point trajectories from optical flow data and modeling those trajectories using generic DCT basis functions. Advantages of our approach are its scalable properties (in terms of point trajectories spatial density) and its genericity (independent from the motion estimation algorithm). The interest of the proposed modeling is illustrated in a motion clustering application and in a block-based video coding framework.

More generally, this kind of motion representation can be useful for many applications of video analysis such as motion segmentation, video post-processing (manipulation, spatio-temporal filtering) or object tracking for video-surveillance.

4.2.2 Point trajectories estimation

There are different ways for estimating dense motion trajectories. The motion threading technique proposed in [60] was originally dedicated to 3D wavelet coding. It performs the motion estimation and the motion paths derivation together, at the macro-block level, reducing the number of overlapping trajectories during a second refinement stage (TOMTH).

Another approach is to perform the motion trajectories derivation independently from the motion estimation process. The advantage is to treat flow estimation as a black box that can be replaced with an alternate flow algorithm, depending on the targeted application. For instance, for motion clustering (4.2.4), a pel recursive algorithm is preferably chosen to finely track objects contours, while for motion coding (4.2.5) a block based motion estimator is favored for its simplicity, its ability to produce more homogeneous vectors and its higher temporal stability.

First, a best in class optical flow algorithm estimates the frame-to-frame optical flow in forward and backward directions. Next, the trajectories are built and the particles positions are refined. Our technique to build point trajectories is widely inspired from the method proposed in [59] but it differs in several important points. One of our goals is to limit the computational complexity by suppressing the costly global trajectory models optimization step that we replace by a particles position refinement integrated into the propagation step.

Another major difference concerns the linking stage replaced by a trajectory grouping optionally performed to reduce the dataset size and which is better tiled to coding constraints. A section parameter S representing the particle size is associated to each trajectory. In that way, the motion trajectories can be

seen as a 3-D volume "carved out" by a moving planar object of shape *S*, like the object tunnels described in [61].

The following steps are performed in forward direction for each frame, from the first frame up to the last frame of a group of pictures, then in backward direction:

- **Propagation**: particles terminating in the previous (or next) frame are extended into the current frame according to the forward (resp. reverse) flow fields.
- **Pruning**: trajectory portions with locally highest matching costs (see below) are removed.
- **Grouping**: neighboring trajectories with similar motion are grouped.

We chose to consider relatively short length sequences only, in which objects disappearances or appearances may occur, but are visible at least in the first or the last frame. Therefore no additional stage to add new particles in gaps between existing particles is required, contrarily to [59].

4.2.2.1 Propagation

In the propagation step, all particles defined in adjacent frames, are projected into the current frame according to the flow fields between the frames. Particle *i* is propagated from frame *t-1* to *t* using the *forward* motion vector $\overline{vf}_{med}(xc_{i,t-1}, S_i)$ computed as:

$$\overline{vf}_{med}(xc_{i,t-1}, S_i) = median\{\overline{v_f}(x, t-1), x \in S_i\} \quad (12)$$

where: S_i is the section of the particle *i*,
$xc_{i,t-1}$ is the center of S_i,
$\overline{v_f}(x, t-1)$ is the estimated *forward* motion vector from frame *t-1* to *t* for sample location *x* in frame *t-1*.

Similarly, one computes the *backward* motion vector $\overline{vb}_{med}(xc_{i,t}, S_i)$ with $\overline{v_b}(x, t)$.

Next, the corresponding particle center position $xc_{i,t} = xc_{i,t-1} + \overline{vf}_{med}(x_{i,t-1}, S_i)$ in the current frame is refined in order to take into account both the signal distortion and the bi-directional flow field coherency (optimization stage). The best refinement value *dx* is chosen among a set of small displacements around $xc_{i,t}$ that minimizes the matching cost locally:

$$cost_i(dx) = SAD(dx) + \lambda . \bar{v}_{diff}(dx) \quad (13)$$

where:

$$SAD(dx) = \frac{1}{n(S_i)} \sum_{x \in S_i} abs\left(I_t(x_{i,t} + dx) - I_{t-1}(x_{i,t-1})\right)$$

$$\bar{v}_{diff}(dx) = \left\| \overline{vf}_{med}(x_{i,t-1}, S_i) - \overline{vb}_{med}(x_{i,t} + dx, S_i) \right\| \quad (14)$$

$n(S_i)$ being the number of pixels contained in S_i,
λ a Lagrangian parameter set to 1 in our experiments,
$I_t(x)$ is the sample value of frame *t* at location *x*.

4.2.2.2 Pruning

In the proposed modeling, crossing trajectories are not considered nor authorized. In case of areas occlusions or disappearing, corresponding trajectories are enforced to end. After the propagation stage, if two different paths propagate at the same location in the current frame, the connection with higher matching cost is removed.

For a given trajectory *i*, the crossing condition with another trajectory *j* corresponds to 3D tunnels intersection (Figure 35) and is computed as:

$$crCond_i = \frac{n(S_i \cap S_j)}{n(S_i)} > Th_c \qquad (15)$$

where: Th_c is a threshold value chosen equal to 0.5,
n(S) is the number of pixels contained in section *S*.

When $crCond_i$ is verified and $cost_i > cost_j$, then the trajectory *i* is ended. In the final step of the pruning stage, we classify the trajectories in 3 classes (some trajectories may belong both to the 1st and 2nd class):

- **Left trajectories**: trajectories starting from the first frame.
- **Right trajectories**: trajectories ending in the last frame.
- **Incomplete trajectories**: trajectories starting and ending in between the first and last frames.

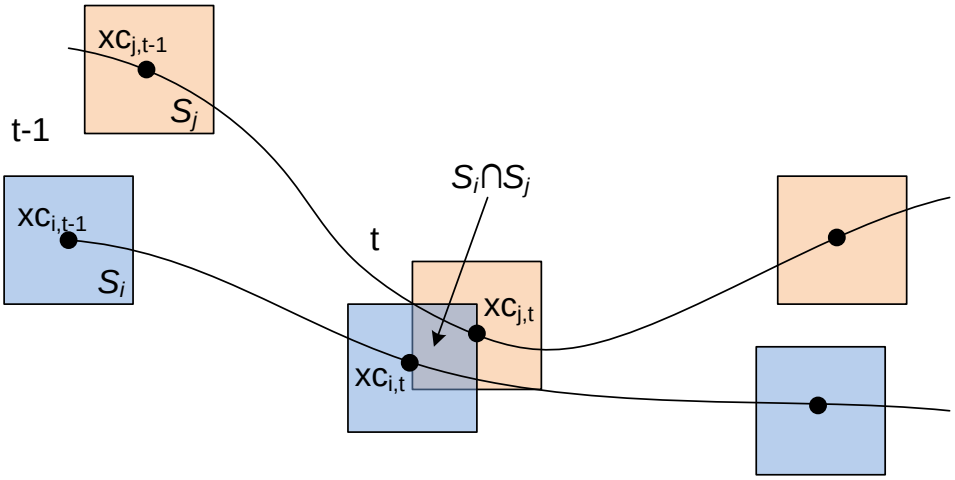


Figure 35: Intersection in frame *t* of two trajectories *i* and *j*, with section *Si* and *Sj* respectively.

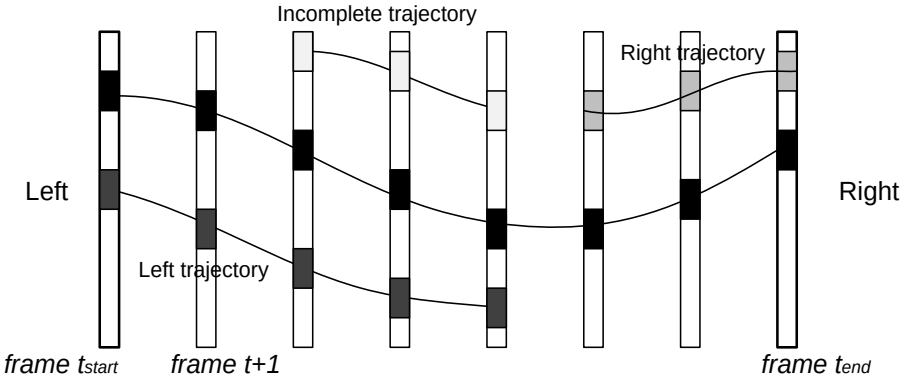


Figure 36: There are 3 classes of trajectories: Left trajectories, Right trajectories and Incomplete trajectories.

4.2.2.3 Grouping

In order to homogenize the trajectories and to reduce the amount of motion data, a grouping process is achieved to cluster similar trajectories. Left (or right) trajectories of same section size are grouped if they are neighboring in the first (resp. last) frame and if their maximum mutual distance $grCond_{i,j}$ is inferior to a threshold Th_g:

$$grCond_{i,j} = max\left\{\left\|xc_{i,t} - xc_{j,t}\right\|, t \in \{t_s, .. t_e\}\right\} < Th_g \qquad (16)$$

where: $xc_{i,t}$ is the position of the center of section S_i of particle i at frame t,
$xc_{j,t}$ is the position of the center of section S_j of particle j at frame t.

Figure 37 shows an example of trajectories obtained after the grouping stage with the CIF sequence Flower, using square sections and Th_g=0.5.

Figure 37: Trajectories grouping for Flower CIF sequence, for frames 200 and 218, using original square sections of size 8.

4.2.3 Trajectories modeling

Modeling trajectories has several advantages. First, it is a good compromise when consistency is more important than correctness. Indeed, if the scene includes arbitrary deforming objects with inadequate texture, finding the true motion may be hopeless. Modeling allows smoothing trajectories and reducing motion heterogeneity due to features mismatches. Another motivation for trajectory modeling is to reduce the amount of data in the perspective of motion compression. At last, it facilitates the clustering because trajectories belonging to same group will share common model features.

However, the modeling of trajectories has some counterpart. It modifies slightly the position of the key points and may distort the objects. Additionally, the property "*the paths do not cross each other*" may not be verified anymore, which can be an issue in some applications like temporal frames interpolation.

In [62] Bregler et al. expressed the idea that observed objects can be represented as a linear combination of a compact set of basis shapes. However, the space of spatial deformations is highly object specific and the shape basis needs to be estimated for each video sequence. To overcome this limitation, Akhter proposed in [63] to use a generic trajectory basis: the Discrete Cosine Transform (DCT) basis. This approach is independent of the scene and another useful advantage is that the structure is automatically compressed without the need for transmitting a sequence specific basis. Then the trajectory can be modeled as:

$$x_{i,t} - x_{i,t-1} = \sum_{k=0}^{K} c_{i,k}.b_k(t) \qquad (17)$$

where: $x_{i,t}$ is the position of the center of section S_i of particle i at frame t,
$b_k(t)$ is the k^{th} basis function,
$c_{i,k}$ are the coefficients associated with trajectory i.

If the length of the ith trajectory is higher than the number of basis vectors (K), the coefficients $c_{i,k}$ can be estimated by solving an over constrained linear system of equations. However the stability of this method tends to decrease when K increases. Higher robustness is obtained if the coefficients $c_{i,k}$ are determined iteratively using a least square method, from the low to high DCT basis frequencies. This algorithm has several advantages. First, one can choose to stop the iteration as soon as a predefined precision has been

reached, because the values of the coefficients $c_{i,k}$ are independent of K. Second, one can identify the coefficients even for incomplete or short trajectories (length < K).

4.2.4 Clustering point trajectories

One straightforward application based on this trajectory modeling is motion segmentation. The projection into basis functions can be considered as a Generalized Principal Component Analysis (GPCA). Then, using a K-mean algorithm we obtain the motion clusters directly. In [64], a GPCA is also used but with polynomial fitting functions, and feature clustering is then made via polynomial differentiation and spectral clustering.

The ability to handle moving objects without any complete trajectory is illustrated by Figure 38. Here, in the *Carmap* sequence, the car is partially occluded by a map. The back and the front parts of the car are classified as left and right distinct trajectories. But since they share common motion features, they are correctly grouped together.

The *Hand* sequence contains more complex motion and a non-rigid object. In *Flower*, the tree is correctly tracked despite the large displacement amplitude.

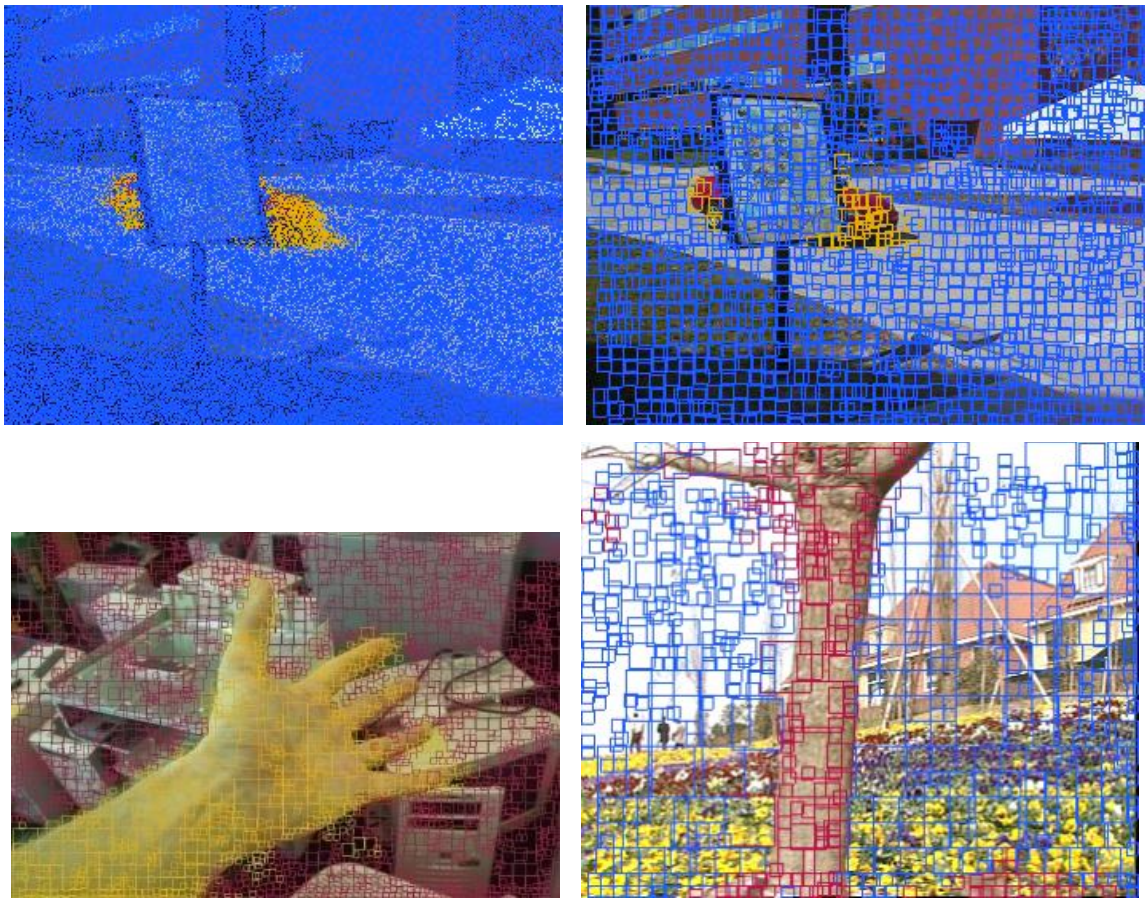


Figure 38: Top: *Carmap* sequence, motion traj. using square sections of size 1 (left), and 8 (right). Bottom: *Hand* and *Flower* sequences with motion trajectories grouping and square sections of size 4,8,16,32.

4.2.4.1 Results on Hopkins155 database

We tested our algorithm on the sequences of the Hopkins155 benchmark presented in [64]. It contains video sequences along with some features have been extracted and tracked in all the frames using KLT

algorithm [138] to generate trajectories. The ground-truth segmentation (2 or 3 motions) is also provided for comparison purposes.

In Table 5 we compare the average classification errors and computation times of our method with four state-of-the-art algorithms reported in [64]. The LSA (Local Subspace Affinity) fits a subspace locally around each projected point, while GPCA uses the gradients of a polynomial that is globally fit to the projected data. The Multi-Stage Learning (MSL) and RANSAC are statistical algorithms.

The misclassification rates are computed as the ratio between the number of misclassified trajectories and the total number of trajectories. Our method offers a good compromise between rapidity, clustering efficiency and robustness.

Table 5: Average misclassification rates and computation times on the Hopkins155 database.

	GPCA	LSA 4n	MSL	RANSAC	Our method
Checkerboard (90 sequences)					
2 motions	6.09%	2.57%	4.46%	6.52%	8.12%
3 motions	31.95%	5.80%	10.38%	25.78%	13.8%
Traffic (37 sequences)					
2 motions	1.41%	5.43%	2.23%	2.55%	0.45%
3 motions	19.83%	25.07%	1.80%	12.83%	4.72%
Articulated (11 sequences)					
2 motions	2.88%	4.10%	7.23%	7.25%	7.84%
3 motions	16.85%	7.25%	2.71%	21.38%	6.00%
All (138 sequences)					
2 motions	4.59%	3.45%	4.14%	5.56%	5.73%
3 motions	28.66%	9.73%	8.23%	22.94%	11.37%
CPU	324ms	7.584s	11h 4m	175ms	179ms

4.2.5 Motion modeling for video coding

This section deals with application of our trajectories model to video coding. Indeed, one design principle of our motion model is that objects can be tracked over time periods and they draw a continuous trajectory in the image plane. This property is naturally well suited for temporal frame interpolation and images sequence coding, in particular for low bit rate applications targeting signal similarity rather than signal fidelity.

4.2.5.1 Motion-compensated temporal interpolation

Trajectories model

We slightly modify our trajectory modeling algorithm as follows. We use square sections (blocks). Since only the first and last images of the sequence are used to interpolate the in-between frames, we modify the computation of the SAD in the propagation stage ($xc_{i,t}$ center positions refinement stage) in order to use the first or the last reference frame (t_{ref}= t_{start} or t_{ref}=t_{end} respectively) only, with ¼ pixel accuracy for dx, as follows:

$$SAD(dx) = \frac{1}{W_i}\sum_{x\in S_i} w(x_{i,t}).\,abs\left(I_t(x_{i,t}+dx) - I_{tref}(x_{i,tref})\right) \qquad (18)$$

$$\text{with} \quad W_i = \sum_{x\in S_i} w(x_{i,t}) \text{ and } w(x) = 1 - 2.\frac{(x-xc_{i,t})^2}{sw_i^2}$$

where: sw_i is the square block width,

$xc_{i,t}$ is the center position of the block in picture t for the trajectory i,
$I_t(x)$ is the sample value of frame t at location x,
t_{ref} is equal to t_{start} or t_{end}.

The incomplete trajectories are removed systematically. Additionally, we introduce a weighting factor $w(x)$ to reduce the influence of the pixels far from the block center $xc_{i,t}$. At the reconstruction stage, these pixels have more chance to overlap with other neighboring blocks and their contribution will be smoothed consequently. The intermediate frames are interpolated by projecting along left and right trajectories, and redundant predictions are merged using Weighted Adaptive Motion Compensation Interpolation (WAMCI) principle [65].

Holes filling

To reconstruct the remaining unpredicted areas we locally expand the block size in the direction of the hole to fill, horizontally or vertically, as depicted in Figure 39. The expanded pixels are copied from the image reference ensuring texture continuity with the neighboring block. They are weighted with a value $w(x)$ decreasing as the distance with the block outline increases. Since square blocks are used, at most 4 different trajectories may contribute to compute one sample in a hole, corresponding to the expansion of blocks at left, right, top and bottom of the hole sample (as depicted in example in Figure 39). Then the sample values in the holes to fill are computed as follows:

$$value(x) = \frac{\sum_i^{ntraj} w\left(x - xc_i(t) - \frac{sw_i}{2}\right) \times I_{tref}\left(xc_i(t) - xc_i(tref_i) + x\right)}{\sum_i^{ntraj} w\left(x - xc_i(t) - \frac{sw_i}{2}\right)} \quad (19)$$

where: t is the frame instant of the frame to interpolate,
$ntraj$ is the number of expanded trajectory sections (at most 4),
$tref_i$ is equal to t_{start} if trajectory i is classified as left, t_{end} if trajectory i is classified as right,
$Iref(x)$ is the reference frame sample value at position x,
$xc_i(t)$ is the center position of the block in picture at instant t for the trajectory i,
sw_i is the square block width,
x is the hole sample position to fill-in,
$w(x)$ is a linear decreasing function, with $w(0)=1$.

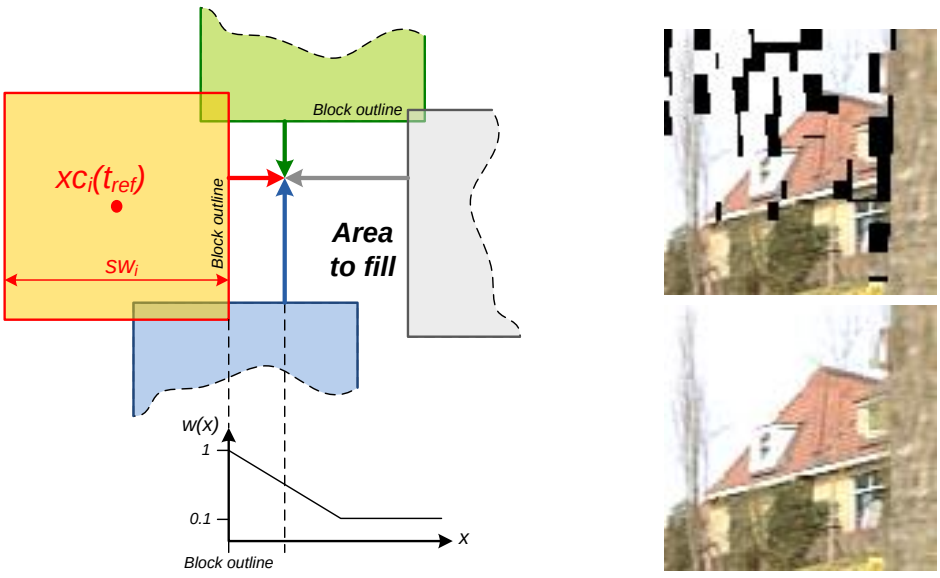


Figure 39: holes filling principle and result on part of *Flower* sequence.

Table 6 presents the PSNRs and reconstruction percentages obtained on the regions reconstructed directly and on the whole frames using our filling algorithm, for different sequences (GOP=16 images). Comparison is done with results obtained by Urvoy in [67] using motion tubes technique without holes filling.

Table 6: Efficiency of intermediate frames reconstruction on different sequences: PSNR and reconstruction percentage.

Algorithm	Mobile		Foreman		Crew	
	psnr (dB)	Rec. (%)	psnr (dB)	Rec. (%)	psnr (dB)	Rec. (%)
Urvoy [67]	28.4	99.4	32.1	93.1	34.7	93.1
Our algo.	28.8	97.7	34.0	97.2	35.0	92.7
Our fill. algo.	28.7	100	31.6	100	34.2	100

4.2.5.2 Video coding based on point trajectories

The temporal interpolation algorithm based on trajectories described in (4.2.5.1) can be used as a prediction stage for coding the intermediate frames of a GOP. If the grouping process uses dyadic ratio, the trajectory sections can be coded with quad-trees as traditional block partitioning. Each block in the first and last frame of the GOP corresponds to a starting (resp. ending) motion trajectory. For each trajectory, the trajectory length and the coefficients $c_{i,k}$ are coded using spatial causal predictors. Advantageously, we exploit the motion segmentation described in (4.2.4) and code a label corresponding to the motion cluster the trajectory belongs to. The spatial predictors and entropy coding contexts are managed separately for each class.

Our trajectory-based video coding scheme is compared to an AVC encoding using the following approach. In both cases, key frames are coded using AVC (JM14.2 software, QP=33). In the AVC coding, intermediate frames are then encoded with a high QP (QP=51, hierarchical B, GOP length of 16 images, CAVLC). In our scheme, intermediate frames are interpolated without residue coding. We re-use the first and last decoded key frames to compute motion trajectories and interpolate the intermediate frames. We quantify and code $c_{i,k}$ coefficients, as well as trajectory length and label with Huffman coder. We choose the quantization step in order to have same average SSIM as for the B frames encoded with AVC. In Table 7 we compare the corresponding intermediate frames bit rates and PSNRs for *Flower* and *Bus*.

Table 7: Performance of our intermediate frames prediction method compared with AVC on different sequences.

	H.264/AVC			Traj. based temp. inter.		
	Rate (bits/s)	PSNR (dB)	SSIM	Rate (bits/s)	PSNR (dB)	SSIM
Flower	48768	22.58	0.74	51450	20.54	0.73
Bus	47152	23.10	0.61	48840	19.75	0.61

We obtain results subjectively better than AVC, with equivalent encoding cost. As expected, the textures are temporally more stable and less smoothed. However, our simple coding scheme suffers from lack of alternative modes such as Intra, in particular for the sequence *Bus* where some regions appearing and disappearing are not visible neither in the first and the last frames. In these cases AVC can rely on Intra modes successfully.

4.2.6 Conclusion

In this section, we have presented a motion-modeling framework based on point trajectories and DCT basis functions. This generic representation supports various section shapes, including block partitioning and is independent of the motion estimator process. From this motion model, we have derived a simple algorithm for motion segmentation offering a good compromise between rapidity, clustering efficiency and robustness. For video coding, we propose a simple bi-directional temporal frame interpolation scheme giving comparable performance in terms of SSIM to AVC hierarchical B prediction at very low bit rates.

Further work could be focused first on improving the motion modeling by considering incomplete trajectories. Regarding our coding solution, several tracks are foreseen: new prediction modes, insertion of anchor blocks in the intermediate frames, holes filling algorithm improvements.

5. Video Coding adapted to the global frames properties

The previous chapters have been dealing with the improvement of the video coding in general, without a-priori about the spatio-temporal statistics of the pictures, all the video content being considered as a globally homogeneous material. The following sections will show how the video coding can take advantage of the knowledge of some particular signal properties. In the proposed two methods, a pre-analysis stage allows determining global frames characteristics information at the encoding stage. This information is coded in the bit-stream and exploited to improve the coding efficiency.

In the first section, we will show that even the video test material used for standardization and supposed to be representative of the variety of video content, contains pronounced signal properties [135]. We propose to transmit few additional data in the bit-stream computed by the encoder with a frame pre-analysis and a pre-processing possibly. At the decoder, slight modifications in the video decoding process are required only.

In the second section, the weighted prediction tool is revisited in the context of HEVC and its performance is analyzed with fading test sequences. A new pre-analysis method for calculating the explicit parameters is proposed and compared with several other state-of-art algorithms. We show the limitations of the former implicit weighted prediction and an alternative model is proposed.

5.1 Content-Adaptive Colour Transform

Two profiles have been defined for the first version of the HEVC Standard. The Main 10 Profile has the same constraints as the Main Profile except the input pictures can be provided with a bit-depth up to 10 bits rather than strictly 8 bits. A 10-bit consumer profile offers several advantages compared to traditional 8-bit coding such as less banding and contouring artifacts as well as an increase in coding accuracy in general [80].

This new 10 bits consumer-oriented profile aims mainly at easing UHDTV (i.e. with Rec.2020 [5] parameters) advent and deployment. Support for 10-bit bit-depth is becoming available on consumer display systems today. And the larger physical size of UHDTV displays, coupled with the wider colour gamut and higher dynamic ranges supported by them, may more readily expose the visual artifacts of 8-bit video content.

In another hand, the potential usage of 10-bit precision in internal data paths (a.k.a Internal Bit-Depth Increase or IBDI) results in better prediction, smaller residuals and better overall visual experience. Experimental results conducted with the HM software which codes 8 bits video contents using internal 10 bits precision for reference frames showed significant BD-rate improvement [81].

5.1.1 Content-adaptive colour space transform

The general principle of our content-adaptive colour transform scheme for video compression is depicted in Figure 40. Basically, the input video samples are transformed before coding and the inverse colour transform is used by the decoder to output the reconstructed pictures in a standard display format. Then additional data should be conveyed in the bit-stream to allow the decoder building the inverse transform. One advantage of this approach is that it does not need to modify the existing core codec except with pre/post processing steps. Other methods such as a block adaptive transform may take advantage of the local colour features of the images but require codec modifications [82].

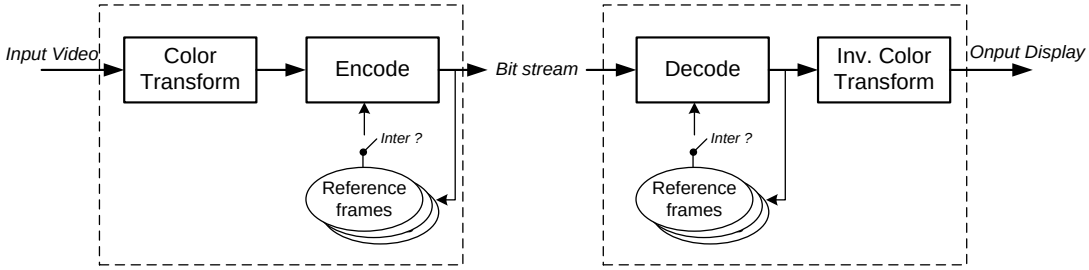


Figure 40: Colour transform principle for video coding.

For the pictures with all the slices Intra encoded, one can set a colour transform per picture. However, in case of Inter pictures coding, one has to define separate groups of pictures with their own transform. Because Inter prediction should use reconstructed reference pictures with same colour space as the current picture to keep coding efficiency.

Several colour models exist for transforming RGB raw data into a format better suited for coding. Their characteristics differ depending on the application requirements. The YCoCg transform and its derivations are good candidates when exact reversibility is required [83]. However, the commonly used format in distribution video codecs originally designed for TV broadcasting is 4:2:0 Y'CbCr (a.k.a. YUV), with a chroma resolution twice less than luma one, both horizontally and vertically. This different sampling used for luma and chroma samples can be an issue when designing a colour transform and the corresponding inverse transform. Hence, in this paper, we consider colour transforms modifying only the chroma components. Note that in commonly used Non-Constant Luminance scheme (NCL, see also chapter 1.2) the luma component (Y') is built as a linear combination of R'G'B' primaries based on human psychovisual considerations, the chroma components Cr and Cb are simply derived as the weighted difference of R' and B' with Y', respectively [85].

The basic idea behind using a colour transform before coding is to re-align the colour basis axes with the main colour characteristics of the content, so that the video codec quantization artifacts are relatively lower on the reconstructed signal as depicted in a 2D example in Figure 41.

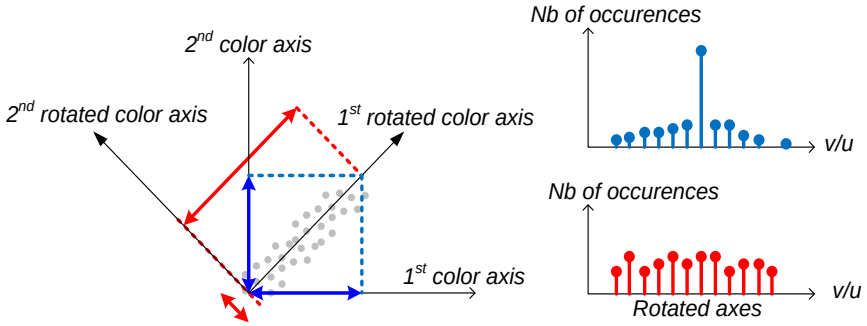


Figure 41: The rotated colour axes allow better de-correlating the (u,v) components.

A second principle is to use a linear transform in order not to distort the signal.

In our experiments we analyzed 2D chroma histograms *H(u,v)* (number of chroma samples with value (*u,v*)) of several video sequences sets, and for most of them we observed that they exhibit very particular characteristics. In Figure 42 we show 2D chroma histograms obtained with some of the HEVC regular sequences.

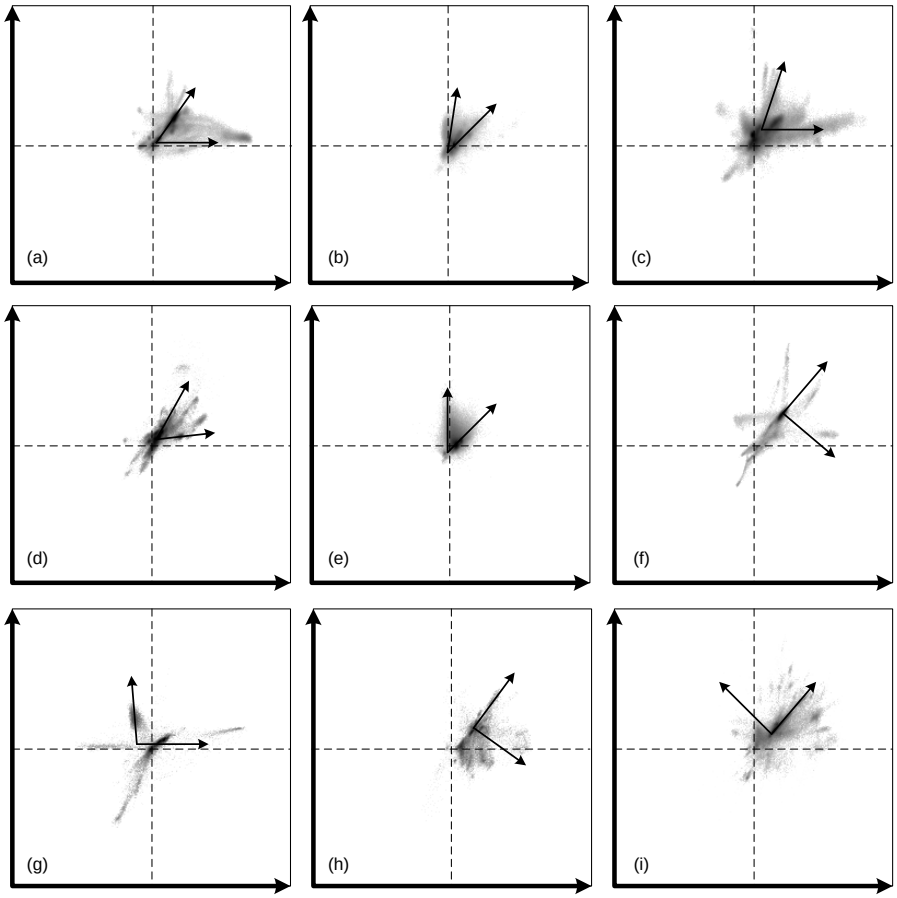


Figure 42: Chroma histograms of some regular HEVC sequences (see Table 1).

5.1.2 Chroma space rotation

Through the observation of 2D chroma histograms (Figure 42), one can visually deduce two remarks. First they are not centered and second they may exhibit direction that are not the same as Cb and Cr axes. Then, the most straightforward transform to be considered in order to re-align the chroma axis and to re-center the samples is a translation plus a rotation as follows:

$$\begin{pmatrix} u' \\ v' \end{pmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \begin{pmatrix} u - C_u \\ v - C_v \end{pmatrix} \qquad (20)$$

where θ is the rotation angle, (C_u,C_v) is the new basis center, (*u,v*) and (*u',v'*) are the original and the transformed chroma coordinates respectively.

Principal Component Analysis (PCA) [86] allows computing the orthogonal transform and the eigen-vectors corresponding to the rotation. This rotation is defined in such a way that the first principal component (major axis) has the largest possible variance. It is noticed that the second axis is orthogonal to the major axis by construction of the PCA process. The centering by translation corresponds to the samples mean value subtraction.

We modified the HEVC reference encoder software (HM7.0) in order to include a pre-processing stage which computes the colour transform as depicted in Figure 40. The rotation angle and the translation are encoded in the first slice of the Clear Random Access frames (a.k.a. CRA, see 2.1.5). Each picture is transformed before coding. In case of 8-bit content encoded in 10-bit, the transform is applied after the original samples have been 2 bits left shifted (two LSB set to 0). At the reconstruction stage (at the decoder side), the inverse transform (rotation and translation) allows restoring the chroma samples to the original Y'CbCr colour space.

We encoded various video sequences including the HEVC regular test set (21 sequences among 5 sequence classes are considered. A: cropped areas of size 2560x1600, B: 1080p, C: WVGA, D: WQVGA, F: Screen Content of misc. sizes), plus some from EBU and SVT. We used the Random Access Main 10 Profile (RA-MP10) test conditions. The PSNR are computed in the original Y'CbCr colour space (reconstructed pictures with inverse colour transform).

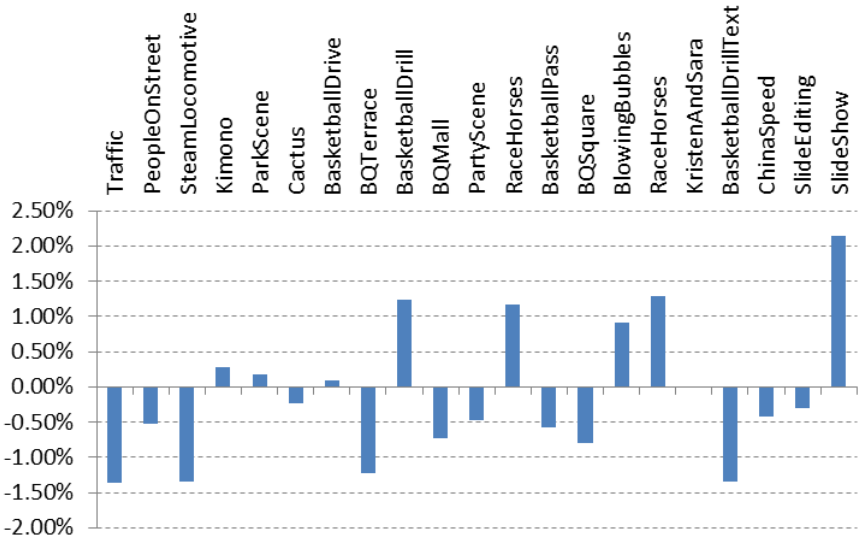


Figure 43: BD-rate gains obtained using PCA with regular HEVC sequences (RA-MP10).

In Figure 43 we plot the relative bit rate gain obtained (negative value is a gain) with our content adaptive colour transform (PCA) using Bjontegaard BD-rate interpolation [72] and the combined (weighted) Y'CbCr PSNR as proposed in [87] (21):

$$psnr_{YUV} = \frac{W_Y.psnr_y + W_U.psnr_u + W_V.psnr_v}{W_Y + W_U + W_V} \qquad (21)$$

with (W_Y, W_U, W_V) weights equal to (6,1,1) respectively.

The evaluation of the BD-rate gains may be difficult since the processing of the Y' component is unchanged. Indeed, by construction, since the Bjontegaard metric uses the same overall bit-rate for the 3 components, an increase in bit rate degrades the BD-rate luma score (luma coding is unchanged, hence luma PSNR is identical) even if chroma PSNR is improved significantly. In our case, combined Y'CbCr PSNR (PSNRyuv) and bit-rate allows measuring the overall (3 components) BD-rate gains.

We obtain a gain for half of the sequences only, corresponding to the cases when orthogonal transform is well fitted to the 2D chroma histogram shape, as shown in example of Figure 44. However, if the principal components are not orthogonal and/or the barycenter is far from their intersection, the PCA method is not well adapted. For these cases, an Independent Component Analysis (ICA) may be more appropriate [88].

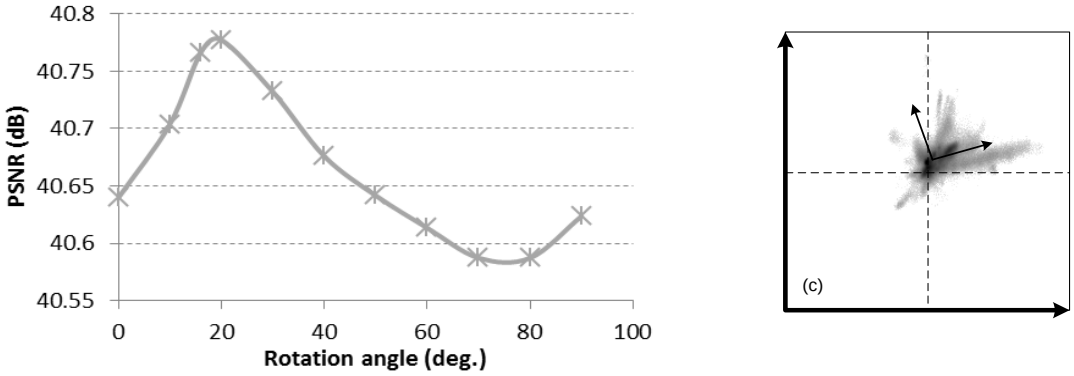


Figure 44: Evolution of YUV PSNR obtained for EBU CrowdRun sequence. The rotation angle given by PCA corresponds to the maximum PSNR.

5.1.3 Non-orthogonal colour transform

To design a non-orthogonal transform, one has to express the new coordinates (*u',v'*) corresponding to a non-orthogonal basis as a function of the original (*u,v*) chroma components (Figure 45).

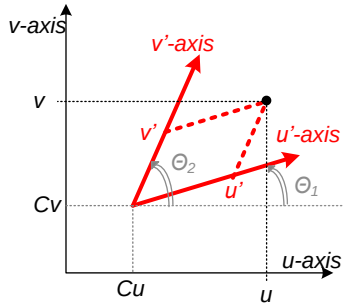


Figure 45: Non-orthogonal transform.

The corresponding chroma transform (encoding stage) is defined as:

$$\begin{pmatrix} u' \\ v' \end{pmatrix} = \frac{1}{\sin(\theta_1-\theta_2)} \begin{bmatrix} -\sin(\theta_2) & \cos(\theta_2) \\ \sin(\theta_1) & -\cos(\theta_1) \end{bmatrix} \begin{pmatrix} u - C_u \\ v - C_v \end{pmatrix} \quad (22)$$

where θ_1 and θ_2 are the angles formed by the new basis axis with the original chroma (u) basis axis (*u-axis*), and (*Cu,Cv*) is the new basis center. The inverse chroma transform (decoding stage) is expressed by (same notation as (22)):

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} C_u \\ C_v \end{pmatrix} + \begin{bmatrix} \cos(\theta_1) & \cos(\theta_2) \\ \sin(\theta_1) & \sin(\theta_2) \end{bmatrix} \begin{pmatrix} u' \\ v' \end{pmatrix} \quad (23)$$

To determine the non-orthogonal transform, one can use an Independent Component Analysis (ICA) algorithm. ICA finds the independent components by maximizing the statistical independence of the estimated components. The way to define independence governs the form of the ICA algorithms [88]. In our case, one has to find the parameters (θ_1, θ_2, C_u,C_v) minimizing the distances of the chroma samples to one of the colour axes (*u'-axis* or *v'-axis*) using oblique projection: projection on one axis according to the other axis direction (Figure 45). For this purpose, we define an energy function as a weighted sum of the smallest distance of each chroma sample to one of the axis (*u'* is the distance to *v'-axis*, *v'* is the distance to *u'-axis*), where the weights are given by the histogram $H_{\theta 1,\theta 2,Cu,Cv}(u',v')$ (number of chroma samples with value (*u',v'*)):

$$E_{\theta_1,\theta_2,C_u,C_v} = \sum_{u',v'} \min(|u'|,|v'|) \times H_{\theta_1,\theta_2,C_u,C_v}(u',v') \qquad (24)$$

For a given set of parameters $(\theta_1,\theta_2,C_u,C_v)$, the corresponding transformed chroma histogram $H_{\theta1,\theta2,Cu,Cv}(u',v')$ is equal to $H(u,v)$, then the re-scanning of the images is not necessary to derive $H_{\theta1,\theta2,Cu,Cv}(u',v')$. One has to find the best parameter set that minimizes $E_{\theta1,\theta2,Cu,Cv}$. Even if the computation of $E_{\theta1,\theta2,Cu,Cv}$ is very fast, the choice of initialization candidates for the values of $(\theta_1,\theta_2,C_u,C_v)$ is of key importance to avoid local minima and to speed-up the refinement process. For (Cu,Cv), we try the histogram barycenter and the default $(0,0)$ center values. For (θ_1,θ_2), the angles corresponding to the rotation found with the PCA algorithm may be good initialization points.

5.1.4 Experimental results

The ICA algorithm has been implemented as a pre-processing stage in the HM7.0. The parameters are encoded with exponential Golomb entropy coding with a precision of 0.1 degree for the angles (θ_1,θ_2) and 1 pixel for the translation (Cu,Cv) in the first slice header of the CRA frames occurring every second (cf. RA conditions [22]). Then only few additional bits are required.

We present in Figure 46 and Table 8 the results obtained with the ICA algorithm, using the same conditions as for PCA (Figure 43). The general trend shown in Figure 46 exhibits a clear improvement, with a gain up to 4%. We got similar results using Intra only configuration, but the gains are slightly smaller. The complexity increase (encoding/decoding time) is negligible both at encoder and decoder side (less than 0.5% enc./dec. time increase in average).

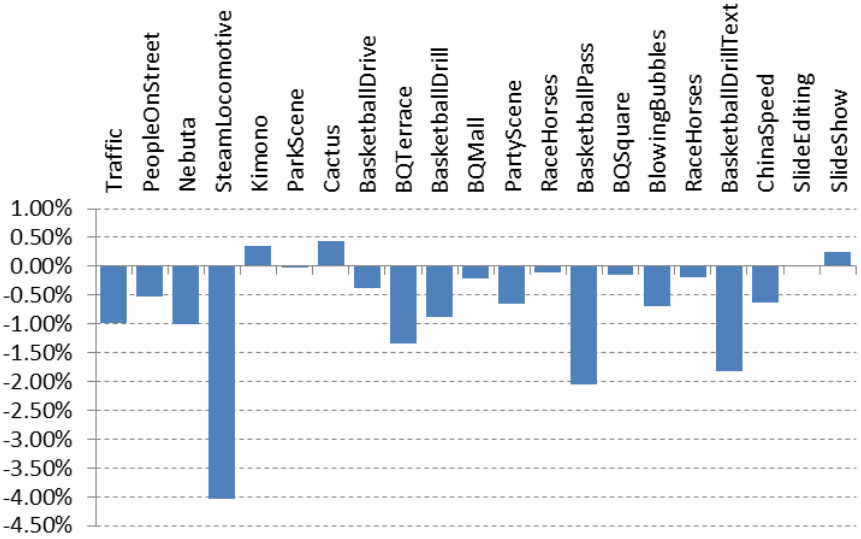


Figure 46: Gains obtained using ICA with regular HEVC sequences (RA-MP10).

Table 8: BD-rate gains obtained using ICA with regular HEVC sequences (RA-MP10).

	BD-rate gains % (piecewise cubic)			
	Y	**U**	**V**	**YUV**
Class A				
Traffic	-0.23	-0.52	-8.64	-0.97
PeopleOnStreet	-0.09	-9.25	5.63	-0.52
Nebuta	1.14	-24.55	-13.13	-1.01
SteamLocomotive	-0.26	-24.85	-32.17	-4.03
Class B				
Kimono	0.18	-0.33	2.78	0.35
ParkScene (e)	0.06	-0.37	-0.49	-0.02

Cactus (c)	0.00	-8.55	9.51	0.44
BasketballDrive (a)	0.90	-29.62	17.07	-0.38
BQTerrace (b)	-0.39	16.06	-24.51	-1.35
Class C				
BasketballDrill (f)	-0.21	-17.96	11.12	-0.89
BQMall (d)	-0.53	-10.87	12.67	-0.22
PartyScene (i)	0.36	-19.67	9.49	-0.66
RaceHorses	-0.26	1.40	-0.08	-0.11
Class D				
BasketballPass	1.39	-27.72	-2.11	-2.05
BQSquare	-0.10	-4.50	3.68	-0.15
BlowingBubbles (h)	0.00	-4.81	-2.81	-0.69
RaceHorses (g)	-0.16	-0.05	-0.56	-0.19
Class F				
BasketballDrillText	0.02	-21.80	5.24	-1.82
ChinaSpeed	0.09	-1.84	-6.47	-0.63
SlideEditing	0.49	-6.69	1.12	-0.01
SlideShow	-0.74	6.55	1.80	0.25
Average Gain				**-0.70**

5.1.5 Conclusion

In this chapter, a pre-processing technique of input video sequences to improve the video compression has been presented. It is based on a content-adaptive colour transform. Two methods have been investigated to determine the best transform parameters. The results show that one can increase the coding efficiency of HEVC Main 10 Profile using a simple algorithm based on ICA. Further improvements are still possible: if the de-blocking and SAO filters are applied after the inverse colour transform (Figure 40) one can expect to correct the colour transform rounding imprecision. In some cases, the limited chroma histogram envelop could be expanded using transform weighting to get similar benefits as chroma IBDI. At last, an appropriate use of chroma QP-offsets may balance the encoding gains in-between the 3 components.

5.2 Weighted prediction

As other state-of-the-art block-based inter-prediction codec, HEVC is very sensitive to illumination variations in-between frames. To cope with this limitation, the weighted prediction (WP) tool has been proposed. In this section, a comparison of the performance of WP in HEVC and MPEG-4 AVC/H.264 is carried out. The efficiency of WP is very dependent on the quality of the estimated WP parameters. The different stages of state-of-art WP parameters estimators are discussed and a new algorithm is proposed. It is based on histogram matching with global motion compensation. Several options are evaluated and comparison is made with other existing methods [66]. At last, the automatic derivation of the WP parameters at the decoder side (implicit WP) is considered too.

5.2.1 Introduction/context

State-of-the-art video compression technologies make use of block based inter-prediction with motion compensation to exploit the temporal redundancy in-between frames of video sequences. Some reconstructed frames are chosen as reference by the encoder and are used to predict the current frame. Then, the closer the predicted frame to the original frame, the smaller the residual signal, which means it will need fewer bits to encode the frame.

However, the determination of the motion vectors and the performance of the inter prediction is very sensitive to illumination variations. To cope with this limitation, MPEG-4 AVC/H.264 implements the Weighted Prediction (WP) tool [68]. The principle of WP consists in replacing the inter prediction signal P by a linear weighted prediction signal P':

- Uni-directional prediction: $P' = w_0 \times P_0 + o_0$
- Bi-prediction: $P' = (w_0 \times P_0 + o_0 + w_1 \times P_1 + o_1) / 2$ (25)

where: P_0 and P_1 are the prediction samples obtained from the reference pictures with indexes *0* and *1*, $(w_i,o_i)_{i=0,1}$ are weights and offsets parameters.

Originally, WP was specially designed to manage fading and cross-fading. Better efficiency is obtained when the reference pictures can be arbitrarily chosen at the encoder side as with the MMCO[4] system in AVC.

WP can manage local illumination variations as well as improved sub-pixel precision for motion compensation using reference picture lists with duplicate references [69]. It can also be used to generate cross-fading transitions in video splicing applications. WP has a very limited overhead in Picture Parameters Set (PPS) (3 bits) and slice headers contain only non-default WP scaling values (w_i,o_i). At last, since WP is an optional PPS parameter, it may be enabled only when necessary.

5.2.2 Weighted prediction in HEVC

Despites the intrinsic improved coding efficiency of HEVC versus AVC, the need to have an Illumination Compensation (IC) tool remains important with HEVC. Several proposals have been made, but the WP scheme implemented as in AVC reportedly showed the best efficiency to manage sequences with fading [71]. To test the WP in HM software, we created linear fade sequences from the regular HEVC test sequences. We use same conditions as in [71]: a linear fade is applied to the first 2 seconds of the regular sequence set and the experiments are done with the 2-second sequences. A linear fade-out (sequence to black or white) is applied in the 1st half and a linear fade-in (respectively black or white to sequence) in the second half (Figure 47). The fade strength is limited to the range [0; 0.75] instead of [0; 1] to avoid extreme PSNR numbers at the frame level that may occur with a uniform black or white picture, and that may bias the rate-distortion results.

[4] Memory Management Command Operation.

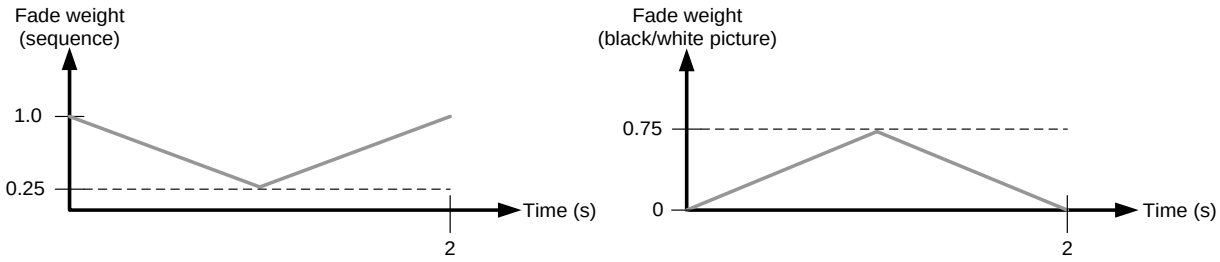


Figure 47: Linear fade-to-black or fade-to-white weights applied on the sequence and the black/white pictures respectively.

To evaluate the performance of WP in HEVC, we implemented WP in HM4.0 in the same way as in AVC, both at the motion estimation and the motion compensation (prediction) stages. However, the weighting is performed at extended bit range resolution (14 bits) for better accuracy, as does the interpolation filter in the motion compensation processes and whatever the original input bit depth is (8 or 10 bits).

The fade sequences are encoded in explicit mode with the WP parameters used to build the fading sequences. Table 9 shows the average bit rate gains [72] obtained using Random Access High-Efficiency hierarchical GOPs (RA-HE) encoder parameters compared to same configuration but without WP, for fade-to-black and fade-to-white sequences.

Table 9: Average bit rate and Encoding/Decoding time gains of HEVC (HM) with explicit WP enabled for a set of fading sequences to black and fading sequences to white (using WP values used for building the fade sequences).

Sequences class	WP explicit mode, fade-to-black, Average gain (%)			WP explicit mode, fade-to-white Average gain (%)		
	Y	U	V	Y	U	V
HM with WP	-17.6	-18.4	-19.0	-18.4	-20.5	-20.7
Enc./Dec. time	139% / 99%			137% / 97%		

The WP tool significantly improves the HEVC efficiency in case of fading sequences (Table 9). The encoding times are increased too because the weighted blending of bi-prediction is more complex, while the decoding time is almost unchanged. The improvement of the prediction in general, leads to fewer coefficients and increased number of Prediction Units (PU) with non-coded residuals, which counterparts the complexity increase of weighted samples at the decoder.

5.2.3 Estimation of the weighted prediction parameters

Several algorithms for estimating WP parameters have been proposed for AVC. Optimal solutions are obtained when IC weights, motion estimation and Rate Distortion Optimization (RDO) are optimized jointly. However, they are generally based on iterative methods requiring huge computation time which is not acceptable for many applications (e.g. real time conditions) and the convergence may not be guaranteed.

Most of the WP parameters estimation algorithms can be described as a three steps process as depicted in Figure 48. In the first step a picture signal analysis is performed to extract image characteristics. It is applied to the current (original) and the reference (reconstructed) pictures identically. In [73] the DC component (mean samples values) and the standard deviation σ are computed, in [74] the mean of the square of pixels value and the mean of the product of the co-located pixels value are calculated additionally. In the proposed method, we use the frame histograms computed over Global translational Motion Compensated (GMC) pictures.

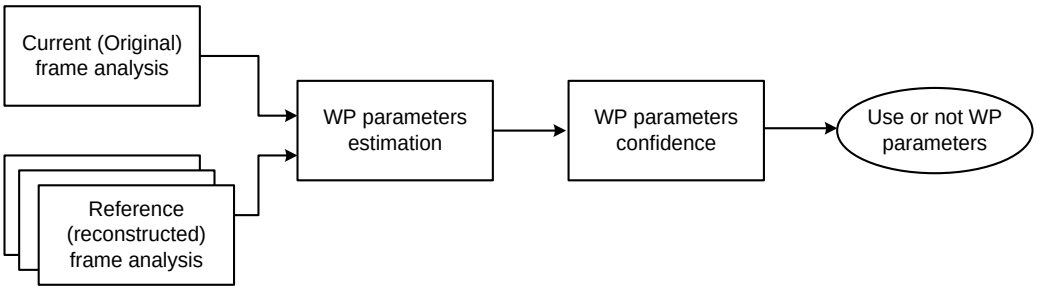


Figure 48: WP parameters estimation is composed of 3 stages.

The next stage is the WP parameters weight (*w*) and offset (*o*) values estimation. The simple DC model proposed in [73] estimates WP parameters with the following equations:

$$w = \frac{\sigma_{cur}}{\sigma_{ref}}$$
$$o = DC_{cur} - w \times DC_{ref} \quad (26)$$

where: DC_{cur} and DC_{ref} are the current and reference pictures mean sample values,
σ_{cur} and σ_{ref} are the current and reference pictures standard deviation.

Another approach consists in finding the (*w,o*) values minimizing the quadratic error between original frame and weighted reference one, using Least Mean Square technique LMS [74]. We adapted this technique to the histogram matching principle (Figure 49). The first advantage is to reduce the amount of calculation since the number of histogram bins is much lower than the number of pixels in a frame, while computation of σ needs scanning the whole frames twice. Another expected benefit is that the cumulated histogram *L(n,I)* (27) is less sensitive to illumination variations in small regions but with relatively large intensity values. The cumulated histogram is computed by:

$$L(n, I) = \sum_{b=0}^{n} H(b, I) \quad (27)$$

where *H(b,I)* is the number of samples in picture *I* with values equal to *b*. Then, the quadratic error to minimize is:

$$error(w, o) = \frac{1}{N}\sum_{n=0}^{nmax}(diff(n))^2, \quad \text{with } diff(n) = L(n, I_{cur}) - w \times L(n, I_{ref}) - o \quad (28)$$

where: *nmax* is the number of histogram bins,
N is the number of samples in the picture,
I_{cur} and I_{ref} are the current (original) and the reference pictures,
(*w,o*) are the WP parameters.

The WP parameters are estimated as the solution of the following equations:

$$\begin{cases} \frac{\delta error}{\delta w}(w, o) = 0 \\ \frac{\delta error}{\delta o}(w, o) = 0 \end{cases} \quad (29)$$

Finally, the validity of the estimated WP parameters is evaluated using a confidence measure. Indeed, if the range of colour sample values is very narrow, σ_{ref} is very small and one cannot rely on (26) for estimating *w*. This may occur for chroma components typically. We investigated two test conditions to validate WP: one based on the value of the quadratic error compared to the standard deviation (30) and one based on SAD calculation with GMC (31), more precise but more costly in computation amount.

$$\sqrt{error(w,o)} < \sigma_{ref} \quad (30)$$

$$SAD\left(GMC\left(I_{ref}(w,o)\right), I_{cur}\right) < SAD\left(GMC\left(I_{ref}\right), I_{cur}\right), \quad \text{with } I_{ref}(w,o) = w \times I_{ref} + o \quad (31)$$

where: σ_{ref} is the reference picture standard deviation,
I_{cur} and I_{ref} are the current and the reference pictures,
(w,o) are the estimated WP parameters,
GMC(I) is the global motion compensated picture of *I*,
SAD() is the Sum of Absolute sample Differences.

If the test condition (30) or (31) is not true, the estimated WP parameters (w,o) are not valid and they are set to the default value *(1,0)*. We implemented the DC model and our histogram matching methods in the HM and we compare them with the DC model implemented in the JM. For the tests, we use the same linear fade sequences as in 5.2.2, the HM and JM encoders are configured with same number of reference frames, hierarchical GOP periods and quantization values corresponding to RA-HE parameters as specified in [75], and the weighted references are used for the motion estimation too.

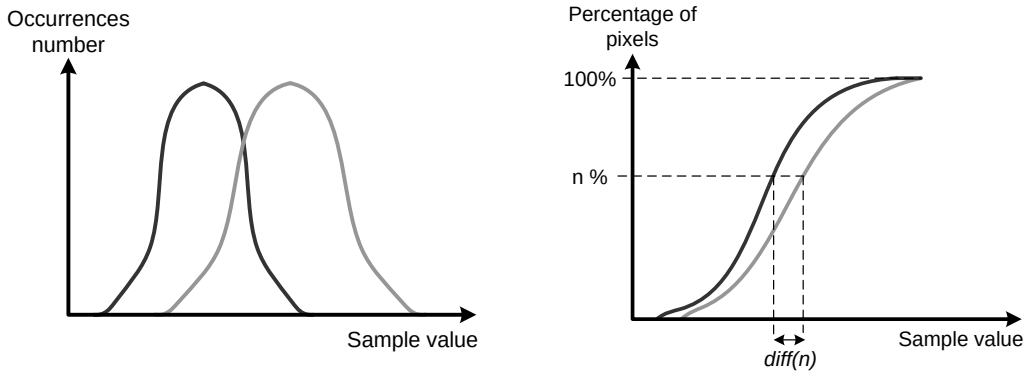


Figure 49: Schematic representation of histogram (left) and cumulated histogram (right) of current (dark) and reference frame (grey).

5.2.4 Experimental results

Table 10 shows the gain (RA-HE) of WP is less significant for HEVC (HM) than it is for AVC (JM). The higher coding efficiency of HEVC with respect to AVC reduces the remaining margin for WP tool coding gain. However, the coding gains are still significant and justify the adoption of WP in HEVC. The use of SAD (31) rather than a threshold in the quadratic error (30) as confidence test is definitely better for histogram matching algorithm. Another interesting result is that the DC method is more efficient than histogram matching based algorithm, while the encoder complexity remains similar. At last, despite the complexity added by WP at the decoder side, the benefit of a better prediction signal allows speeding-up the decoding process. We observed that this trend is reinforced for Low-Delay (see 2.1) non-hierarchical GOPs configurations [75].

Table 10: Average bit rate and Encoding/Decoding time gains of HEVC (HM) and AVC (JM) with explicit WP for a set of fading sequences to black or white (WP parameters estimated with DC and histogram matching algo.).

WP algorithm and encoder	WP explicit mode, fade-to-black, Average gain (%)			WP explicit mode, fade-to-white, Average gain (%)		
	Y	U	V	Y	U	V
JM (DC algo)	-30.0	-24.7	-21.9	-23.2	-22.9	-24.7
Enc./Dec. time	131% / 70%			133% / 95%		
HM (DC algo)	-16,4	-18,4	-18,0	-19,4	-21,8	-21,6
Enc./Dec. time	138% / 97%			141% / 94%		

HM (histo. match., SAD)	-16,0	-16,2	-16,1	-17,7	-19,0	-19,0
Enc./Dec. time	140% / 99%			136% / 102%		
HM (histo. match., error thresh)	-15.5	-10.1	-9.1	-17.0	-13.3	-12.6
Enc./Dec. time	140% / 97%			137% / 103%		

The design of the WP parameters estimation module has major importance on the WP efficiency too. The results are significantly different depending on whether DC and standard deviation (or frame histograms) values are computed using the original or the reconstructed frames (Table 11). This may not impact most of the encoder designs since this analysis can be performed once when the original frame is loaded.

Table 11: Average bit rate and Encoding/Decoding time gains of HEVC (HM) with explicit WP for a set of fading sequences to black or white (WP parameters estimated with histogram matching algorithm), using original or reconstructed frames for analysis.

WP histo. match (HM)	**Sequences fade-to-black Average gain (%)**			**Sequences fade-to-white Average gain (%)**		
	Y	**U**	**V**	**Y**	**U**	**V**
Analysis uses Reconstructed	-14.9	-12.5	-14.0	-17.3	-18.6	-18.9
Enc./Dec. time	142% / 99%			143% / 100%		
Analysis uses Original	-16,0	-16,2	-16,1	-17,7	-19,0	-19,0
Enc./Dec. time	141% / 101%			140% / 100%		

In case GMC is enabled, the frame analysis process operates on the common area between the reference and the current frame (Figure 50). It allows selecting only the samples that may have chance to be used as inter-prediction with WP because they correspond to areas present in both pictures. Then the WP parameters estimation with GMC should be more accurate. However, for each current picture, analysis should be done for every reference frame. This may increase the required memory when analysis is performed on the source material rather than on the reconstructed reference frames.

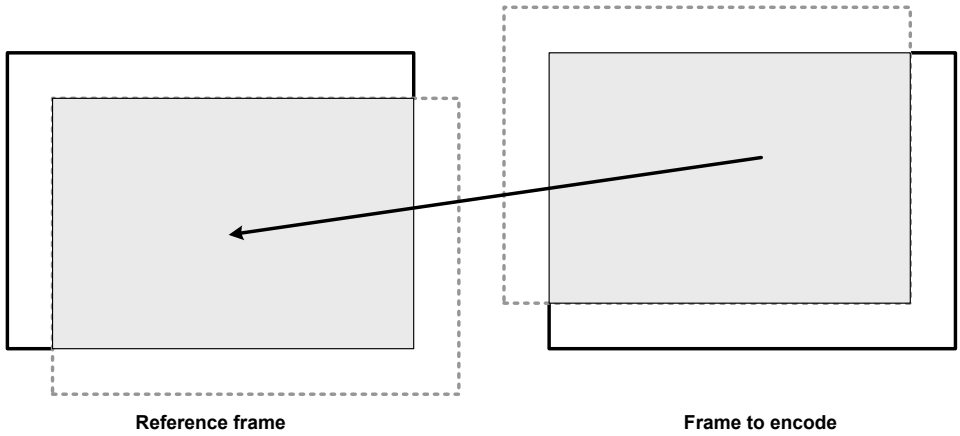


Figure 50: When GMC is enabled, WP parameter estimation is performed over the common area in-between the reference and the current frame.

We evaluated two methods for the determination of the GMC vector. The first one is based on Full Global Motion (FGM) estimation, using simple hierarchical Gaussian pyramids and a gradient descent on 3x3 square pattern [76]. However, the complexity added by the FGM may not be suitable for many encoder applications. Then we tested another method approximating the GMC vector between the current frame and the reference frame as the median vector of all the corresponding (same reference index) available co-located motion vectors stored in the encoder memory [77]. Experimental results show these two methods

have almost equivalent results (with a slight advantage to the second method), as depicted in Table 12 for the DC algorithm.

Table 12: Average bit rate and Encoding/Decoding time gains of HEVC (HM) for a set of fading sequences to black or white, WP parameters estimated with DC algo., GMC vector computed with FGM or co-located motion vectors data.

WP DC algo + GMC (HM)	Sequences with motion, fade-to-black Average gain (%)			Sequences with motion, fade-to-white Average gain (%)		
	Y	U	V	Y	U	V
Analysis uses FGM	-18,0	-19,1	-21,9	-26,2	-24,0	-26,1
Enc./Dec. time	142% / 99%			143% / 100%		
Analysis uses co-loc. motion vectors	-18,4	-19,8	-22,4	-26,9	-28,2	-31,7
Enc./Dec. time	141% / 101%			140% / 100%		

To evaluate the added value of the GMC in the WP parameters estimation process, we selected an additional set of sequences with global motion clearly identified ("sequences with motion") that we added to the list of regular sequences used in JCT-VC (all sequences). The advantage of the GMC in the analysis and confidence test stages is depicted in Table 13, in particular for fade-to-black sequences and for chroma components where the estimation of WP weights may be difficult in some cases because of narrow histograms.

Table 13: Average bit rate gain of HEVC with explicit WP enabled for a set of fading sequences to black or white (sequences with global motion, WP parameters estimated with DC and histogram matching algo.).

	fade-to-black Average gain (%)						fade-to-white Average gain (%)					
WP algorithm w/o GMC (HM encoder)	**Seq. with motion**			**All sequences**			**Seq. with motion**			**All sequences**		
	Y	**U**	**V**	**Y**	**U**	**V**	**Y**	**U**	**V**	**Y**	**U**	**V**
Histo. match	-15.6	-14.4	-17.3	-15.9	-15.9	-16.5	-23.9	-23.9	-27.5	-19.4	-20.3	-21.2
Enc./Dec. time	141% / 101%						140% / 100%					
Histo. match + GMC	-16.7	-13.2	-12.9	-16.2	-15.8	-16.5	-24.6	-24.0	-27.3	-19.5	-20.3	-21.1
Enc./Dec. time	140% / 106%						140% / 90%					
DC algo	-17.6	-15.1	-14.1	-16.7	-18.5	-18.9	-26.9	-27.9	-31.8	-20.9	-19.8	-19.0
Enc./Dec. time	141% / 104%						138% / 101%					
DC algo + GMC	-18.4	-19.8	-22.4	-17.0	-18.8	-19.3	-26.9	-28.2	-31.7	-21.3	-23.4	-24.2
Enc./Dec. time	141% / 102%						138% / 101%					

Finally, the overall better efficiency is obtained with the DC algorithm and GMC, using co-located motion vectors to estimate the global motion, the frame analysis being done with source frames.

5.2.5 Implicit weighted prediction in HEVC

During the development of the HEVC standard, it has been also considered the possibility to derive the WP parameters from the Picture Order Count of the current and reference pictures (a.k.a. Implicit mode), contrarily to the previous section where the WP parameters are transmitted in the bit-stream (a.k.a explicit mode).

Then, depending on the type of slice (P_SLICE or B_SLICE), one can choose the weighting algorithm:

- Default: the default uni-prediction or bi-prediction.
- Explicit: the weighting factors (w_i,o_i) are transmitted explicitly in the slice header (see previous section 5.2.1), both in uni-prediction and bi-prediction.

- Implicit: the weighting factors for bi-prediction in (25) are derived from the distance of the current Picture Order Count (POC) with the POC of the reference pictures, relatively to the distance between the references POC (B_SLICE only). The weights (w_i) and offset (o_i) are determined as:

$$w_0 = 2 - \alpha, \qquad w_1 = \alpha \tag{32}$$

where: $\alpha = 2 \times \frac{poc_{cur} - poc_0}{poc_1 - poc_0}$ and $o_{i=0,1}=0$,

poc_{cur} is the POC of the current picture,

poc_0 and poc_1 are the POC of the reference pictures of lists *0* and *1*.

Practically, to avoid floating point multiplication, Equation (32) is implemented using only integer multiplications and shift. The implicit weighting method is typically used in case of linear fade or cross-fade.

Note that in the Equation (25) these weights are rescaled with a division by 2 to take into account the bi-prediction factor. In the following, we will denote "relative weights" the rescaled values $w_0/2$ and $w_1/2$.

The presence of weighting factors and the weighting method to use are indicated in the PPS with *weighted_pred_flag* (1 bit) and *weighted_bipred_idc* (2 bits) (Table 14). If present, the adaptive weighting factor parameters are transmitted in the slice header for each reference picture.

Table 14: Interpretation of *weighted_pred_flag* and *weighted_bipred_idc* values.

P_SLICE	
weighted_pred_flag	**weighting method**
0	Default
1	Explicit

B_SLICE	
weighted_bipred_idc	**weighting method**
0	Default
1	Explicit
2	Implicit

5.2.5.1 Limitations of existing Implicit WP

The Implicit WP (IWP) allows reducing the slice header size and to lightweight the encoder WP weights estimation process in case of linear fades. The IWP has been specially designed for AVC and it provides significant gains with two references for IbbP GOPs (relative weights=(1/3, 2/3)) as explained in [78].

However, in the case of HEVC, the test conditions have changed compared to the ones used in JVT. The Random Access (RA) configuration uses Hierarchical GOP structure that has proven better efficiency compared to the former IbbP GOP structure, and that is now widely used in broadcast and broadband applications. Additionally, four reference frames are used JCT-VC while JVT used only two.

In the Figure 51 (left), examples of IWP relative weights are depicted, considering the current POC is 2 and RA conditions. In that case, the lists L0 and L1 are equal to {0;4} and {4,8} respectively, then several combinations are possible depending on which reference frames are chosen in L0 and L1. If the value of the current POC is not in-between the POC value of the reference frames, one relative weight may be negative and the other may be superior to 1. The same situation may occur in Low Delay (LD) configuration (Figure 51, right). It has two major drawbacks; first the building of the weighted prediction may generate negative or overflowing sample values that should be clipped, creating saturation artefacts that may degrade the prediction signal; second this may impact the performance of the motion estimation process, traditionally based on iterative refinement of the displacement vectors, fixing alternatively one reference and the other.

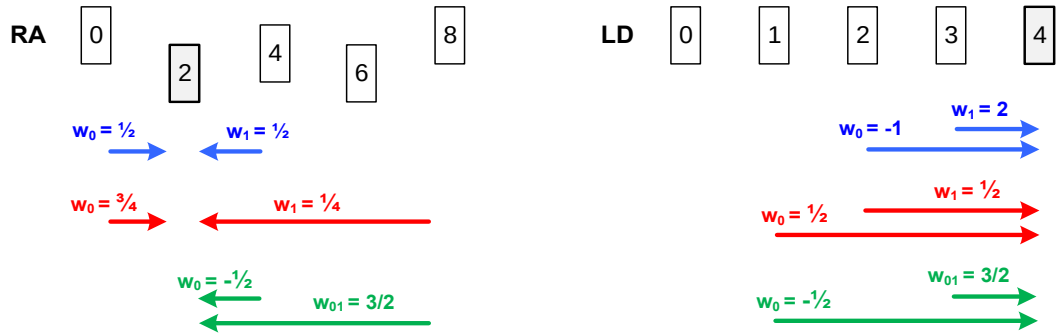


Figure 51: Implicit WP relative weights for RA (left) and LD (right) HEVC configurations, with current POC frame equal to 2 and 4 respectively.

5.2.5.2 Improvement of implicit weighted prediction

To cope with these IWP limitations, we have studied a new algorithm that keeps the same principle as previous IWP: the weights vary linearly with time (POC). The derivation of the weights and offsets is made by interpolating linearly with time (POC) the explicit or implicit weights and offsets of previously decoded reference pictures. It does not need any additional decoded frame analysis.

This Implicit WP method makes lighter the WP weight signaling and reduces complexity at encoder side, by-passing the WP estimation stage for some frames. It is also compatible with temporal scalability: only high level syntax (WP weights and offsets) of the base layer is used to derive the WP parameters of the temporal enhancement layer.

Algorithm description

In our proposed method, we create a hash table containing the explicit or implicit weights and offsets ($w_{cj,}$ o_{cj}) that were used to decode the pictures marked as reference. It is up-dated after decoding every reference picture. Denote ($w_{cj,}$ o_{cj}) the weight and offset associated with the reference frame j having POC equal to poc_j and used to decode the picture with POC=poc_c.

The hash table is up-dated as follows: a new line poc_{Cur} containing the set of weights w_{cj} and offsets o_{cj} (where $j=0,..N$, and N is the number of references used by the current frame poc_{Cur}) is added, and also a new column with empty cells except the last one with (1;0), as depicted in Table 15. The information whether the weights are explicit or implicit is also stored. Note some cells may be empty.

The references pictures removed from the DPB are also removed from the table by suppressing the corresponding lines and columns.

Table 15: hash table containing the explicit or implicit weights and offsets.

	Poc_S	**…**	**Poc_k**	**poc_{Cur}**
Poc_S	(1,0)			
…				
Poc_K	(w_{ks},o_{ks})		(1,0)	
poc_{Cur}	(w_{cs},o_{cs})		(w_{ck},o_{ck})	(1,0)

To derive w_{ck} and o_{ck} corresponding to the implicit weight and offset associated with reference k (k^{th} column) for the current picture c, we re-order the $n(k)$ WP parameters (non-empty cells) stored in the k^{th} column with the values marked as implicit first, and next those marked as explicit. Then we select the two last elements (poc_A and poc_B) of this reordered list, favoring the use of last decoded WP explicit values. The values of w_{ck} and o_{ck} are computed as follows:

If $n(k)$ is >= 2, then we interpolate the weights for poc_{Cur} as:

$$w_{ck} = w_{Ak} + (w_{Bk} - w_{Ak})*(poc_{cur} - poc_A)/(poc_B - poc_A)$$

$$o_{ck} = o_{Ak} + (o_{Bk} - o_{Ak})*(poc_{cur} - poc_A)/(poc_B - poc_A) \quad (33)$$

If $n(k)$ =1, and if it exists $s<k$ with $n(s)$ >= 2, then the weights are extrapolated as :

$$w_{ck} = w_{cs} / w_{ks}$$

$$o_{ck} = o_{cs} - (w_{cs} / w_{ks}) \times o_{ks} \quad (34)$$

where: $n(k)$ is the number of non empty cells in the k^{th} colomn,
(w_{ck},o_{ck}) are the weight and offset associated with the k^{th} reference in the hash table for the current picture c,
poc_{cur} is the POC of the current picture,
poc_A, poc_B are the POC of the selected two elements in k^{th} column of the hash table respectively.

We also propose to add more flexibility by offering the possibility to choose between using explicit WP or implicit WP for each frame or slice, without duplicating PPS. Then a flag *wp_implicit_flag* is added in the slice header to indicate whether to use Explicit or Implicit algorithm.

Hash table filling example

A simple example of the WP hash table filling is detailed below in three steps.

Step 1: The first picture (poc_0=0) is to be decoded. A new line (poc_0) and a new column (poc_0) is created. The cell (poc_0, poc_0) is set with WP defaults parameters (1,0).

Step 2: The second picture (poc_1=1) is to be decoded. A new line (poc_1) and a new column (poc_1) is created. The cell (poc_1, poc_1) is set with WP defaults parameters (1,0). This picture uses explicit WP weights and offset $e(w_{10},O_{10})$ with the reference poc_0. Then these explicit parameters are stored in the poc_1 line in the column corresponding to the reference poc_0.

Step 3: The third picture (poc_2=2) is to be decoded. A new line (poc_2) and a new column (poc_2) is created. The cell (poc_2, poc_2) is set with WP defaults parameters (1,0). This current picture ($c=2$) uses implicit WP with references poc_0 and poc_1. $n(0)$ and $n(1)$ are equal to 2 and 1 respectively, then (w_{20},o_{20}) is computed with (33) with ($c=2$, $A=0,B=1$), and (w_{21},O_{21}) is computed with (34) with ($c=2$, $s=0,k=1$). The computed WP parameters are stored in the cells (poc_2,poc_0) and (poc_2,poc_1) respectively.

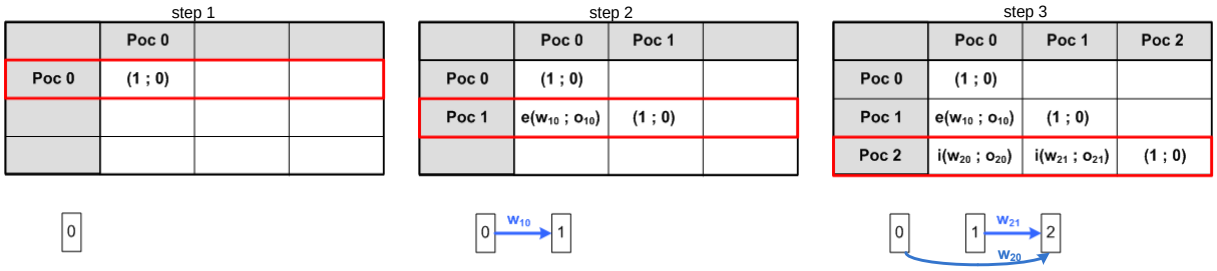


Figure 52: Example of WP hash table up-date.

5.2.5.3 Experimental results

We conducted experiments with HM software version 6.0 (HM-6.0), using common configuration parameters and same test sequences (black/white-fade) as in previous chapter (5.2.2).

Table 16 and Table 17 show BD-rate gains and the relative average encoding/decoding time for explicit WP, implicit WP and the proposed implicit WP for different values of Explicit WP period (EP), compared

to HM6.0 anchor, for black-fade sequences and white-fade sequences respectively. The negative values indicate gains. The full detailed results are available in [79].

It is observed that our proposed Implicit WP with EP=2 or EP=4 can achieve gains of 17% to 34% on average for Random Access and Low Delay cases. These gains, decoding and encoding times are similar to the explicit WP. Larger EP values bring still significant gains.

Table 16: BD-Rate[%] and relative encoding/decoding time[%] of explicit WP, implicit WP and proposed Implicit WP compared to HM6.0 anchor for black-fade sequences.

Black-fade case	Random Access Main					Random Access HE10				
	Y	U	V	Enc %	Dec %	Y	U	V	Enc %	Dec %
ExplicitWP	-16.9%	-19.6%	-19.2%	142%	98%	-17.2%	-19.6%	-19.4%	100%	95%
ImplicitWP	-0.2%	-0.3%	-0.6%	175%	98%	-0.2%	-0.1%	-0.1%	102%	97%
Proposed (EP=2)	-17.5%	-19.4%	-18.8%	143%	86%	-17.4%	-19.2%	-18.9%	102%	88%
Proposed (EP=4)	-17.9%	-19.4%	-18.6%	142%	105%	-17.7%	-18.4%	-18.2%	102%	96%
Proposed (EP=8)	-17.4%	-19.1%	-17.9%	143%	100%	-17.0%	-16.5%	-16.3%	102%	95%
Proposed (EP=16)	-16.7%	-17.2%	-15.6%	144%	103%	-16.4%	-13.9%	-14.6%	102%	96%

Black-fade case	Low delay B Main					Low delay B HE10				
	Y	U	V	Enc %	Dec %	Y	U	V	Enc %	Dec %
ExplicitWP	-32.3%	-40.5%	-39.3%	141%	88%	-32.0%	-43.2%	-41.2%	102%	82%
ImplicitWP	1.4%	3.1%	2.2%	183%	105%	1.9%	2.6%	2.4%	102%	99%
Proposed (EP=2)	-32.4%	-38.0%	-35.9%	142%	75%	-32.1%	-37.8%	-34.9%	102%	74%
Proposed (EP=4)	-31.5%	-37.3%	-32.9%	143%	90%	-30.9%	-35.8%	-32.6%	102%	81%
Proposed (EP=8)	-27.6%	-31.7%	-27.2%	146%	94%	-27.2%	-29.3%	-24.8%	102%	85%
Proposed (EP=16)	-15.7%	-19.8%	-9.6%	154%	101%	-15.8%	-12.8%	-1.7%	102%	88%

Table 17: BD-Rate[%] and relative encoding/decoding time[%] of explicit WP, implicit WP and proposed Implicit WP compared to HM6.0 anchor for white-fade sequences.

White-fade case	Random Access Main					Random Access HE10				
	Y	U	V	Enc %	Dec %	Y	U	V	Enc %	Dec %
ExplicitWP	-20.3%	-23.4%	-23.0%	137%	98%	-20.5%	-22.9%	-23.1%	102%	97%
ImplicitWP	-0.3%	-0.5%	-0.6%	182%	99%	-0.2%	-0.1%	-0.2%	102%	105%
Proposed (EP=2)	-20.5%	-23.0%	-22.5%	138%	72%	-20.7%	-22.7%	-23.0%	102%	95%
Proposed (EP=4)	-20.6%	-22.8%	-22.5%	137%	81%	-20.8%	-21.7%	-22.0%	102%	101%
Proposed (EP=8)	-19.7%	-22.3%	-21.3%	140%	81%	-19.6%	-19.6%	-20.0%	102%	96%
Proposed (EP=16)	-18.4%	-19.9%	-19.0%	142%	74%	-18.4%	-16.9%	-17.7%	102%	101%

White-fade case	Low delay B Main					Low delay B HE10				
	Y	U	V	Enc %	Dec %	Y	U	V	Enc %	Dec %
ExplicitWP	-34.2%	-42.3%	-41.4%	136%	88%	-33.6%	-44.3%	-42.8%	102%	89%
ImplicitWP	1.5%	2.5%	2.8%	187%	112%	2.0%	3.2%	2.7%	102%	105%
Proposed (EP=2)	-34.2%	-40.3%	-37.0%	137%	77%	-33.9%	-39.6%	-37.4%	102%	77%
Proposed (EP=4)	-33.3%	-39.1%	-35.0%	138%	90%	-32.7%	-37.7%	-35.4%	102%	86%
Proposed (EP=8)	-29.2%	-33.8%	-29.0%	142%	95%	-28.7%	-30.9%	-27.5%	102%	88%
Proposed (EP=16)	-17.3%	-22.1%	-12.3%	150%	91%	-17.0%	-13.9%	-4.9%	102%	95%

5.2.6 Conclusion

The WP tool significantly improves the HEVC coding efficiency in case of fading sequences. The gains obtained are lower than for AVC, but they are still important. The complexity at the encoder side is increased (40%) but the decoder processing amount is unchanged or even reduced comparatively to a decoder that would not support WP.

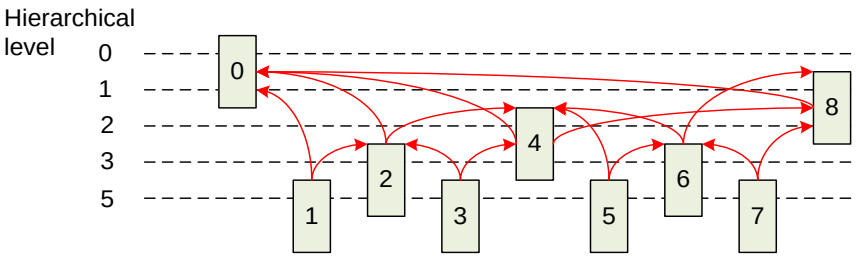
The choice of the algorithm to estimate the weights as well as the test used to measure the WP parameters confidence is of major importance. In particular, the use of global motion compensation improves the overall efficiency. If the WP values are well estimated, the complexity added by the weighted prediction and the weighted blending of bi-prediction at the decoder side is compensated by the benefit of better prediction signal which simplifies the pictures reconstruction process.

The WP Explicit has been adopted by JCT-VC and included in the HEVC standard. Whereas given the relative small overhead of the WP explicit parameters (slice header), the WP Implicit has been removed from the HEVC standard.

6. Video Coding adapted to local features

Basically, the bit rate of the binary streams generated by the video encoders is controlled via the quantization parameter (QP). The QP value behaves as an index that refers to luma/chroma quantization tables and allows deriving the quantization step, via quantization offsets (ex: *chroma_qp_offset*) and/or quantization weighting possibly.

The video encoders usually include a rate control module in charge of setting the QP to reach the target bit rate, while preserving the quality of the reconstructed frames as much as possible. The choice of QP for a given frame not only impacts the coding of this frame but also the following frames (in coding order) if it is used as reference for these frames. Then one commonly selects lower QP value for reference frames. For example, in JCT-VC test conditions [55], the QP value is incremented with an offset corresponding to the hierarchical level of the encoded frame (Figure 53).



Hierarchical level	QP offset
0	0
1	1
2	2
3	3
5	5

Figure 53: In JCT-VC RA (Hierarchical GOP) test conditions, the QP offset is function of hierarchical level.

To better tune the QP with the local characteristics, the QP can be adapted per slice, per (group of) coding unit (HEVC) or per macroblock (AVC). Two different approaches for local QP adaptation are commonly used. The first one is to distribute the bits budget inside the frame according to rate-distortion considerations, such as RDO-based brute force QP search. The second approach is based on perceptual considerations, a-priori allocating more bits (lower QP) to the frame regions where the artefacts are more visible. Most real-world encoders employ the latter to optimize subjective visual quality for the human visual system (HVS) with limited computational resources.

In the following two sections, we will present two contributions related to the coding of variable QP in the frames for HEVC (section 6.1) and to the study of locally QP adaptation based on perceptual (HVS) considerations respectively (section 6.2), that can be considered as a direct application of the latest.

6.1 Prediction of the Quantization Parameter

In AVC and HEVC, the QP is encoded in the slice header. It can be refined at the coding block level by coding the difference with the QP prediction. In AVC, the QP prediction is the previously (macro-block scanning order) QP value used for quantizing the residuals coefficients.

In HEVC, the enabling of local QP adaptation and the QP adaptation granularity are specified in the Picture Parameter Set (PPS) by *cu_qp_delta_enabled_flag* and *diff_cu_qp_delta_depth* respectively. The Quantization Group size (QG) is derived from *diff_cu_qp_delta_depth*: if *diff_cu_qp_delta_depth*=0,1,2,3, the QP value can be specified at 64x64, 32x32, 16x16 or 8x8 respectively (if the size of CTU is set as 64x64).

The JCT-VC implemented the MPEG-2 TM5 Step3 QP allocation model [89] extended to cope with the CU structure of HEVC in HM software (averaged activity maps computed with different block size). In JCT-VC studies, this TM5 implementation is considered as the reference perceptual QP adaptation algorithm. In TM5, the local QP value is derived directly from the activity, computed as a function of the standard deviation in the block (Figure 54).

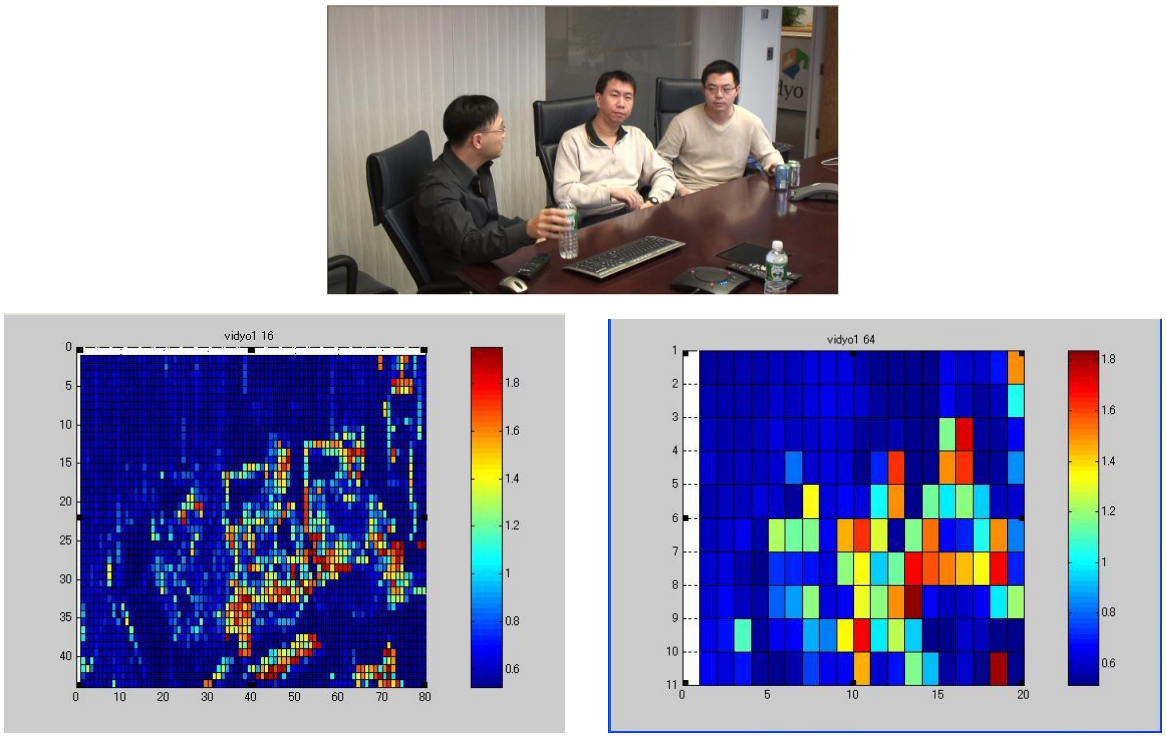


Figure 54: TM5 activity for the sequence "Vidyo" (top), with block size of 16x16 (bottom-left) and 64x64 (bottom-right).

6.1.1 QP prediction algorithms

The JCT-VC evaluated several algorithms for the QP prediction. They were generally based on the same two steps process. First, the neighboring available CU coding parameters are used to select a list of QGs with characteristics presumably similar to the current QG. Second, the QP values of these QGs are combined (ex: weighted average) to derive the QP prediction value.

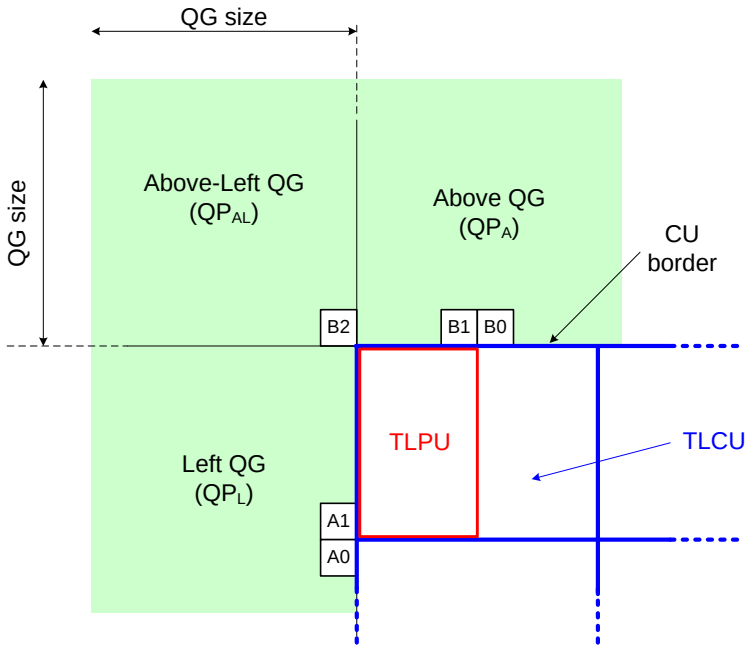


Figure 55: The Top-Left PU (TLPU) and the Top-Left CU (TLCU) of the current QG.

In our study, we considered the following 3 QP prediction algorithms (a), (b) and (c).

a) Weighted QP prediction:

Denote QP_A and QP_L, the QP values of the Above QG and Left QG respectively (see Figure 55). The QP prediction value is QP_P as defined below:

- If QP_A and QP_L are both available: $QP_P = ½ (QP_A + QP_L)$
- If QP_A only is available (or QP_L only): $QP_P = QP_A$ (or $QP_P = QP_L$ respectively)
- If QP_A and QP_L are both not available: QP_P is set to the last encoded QP value.

Indeed, it may happen that reference QP values are not available in case of frame/slice border, or because neighboring QG have no QP (skip mode), etc…

b) Intra prediction based QP prediction:

In [90], J.Xu et al. proposed an Intra prediction based QP prediction using intra mode directions (of the Top-Left CU (TLCU) of the current QG) to retrieve spatial QP predictor(s) of the current QG, as depicted in Figure 56 (red and yellow cases). If these QPs are not available, then the QP prediction (a) is used.

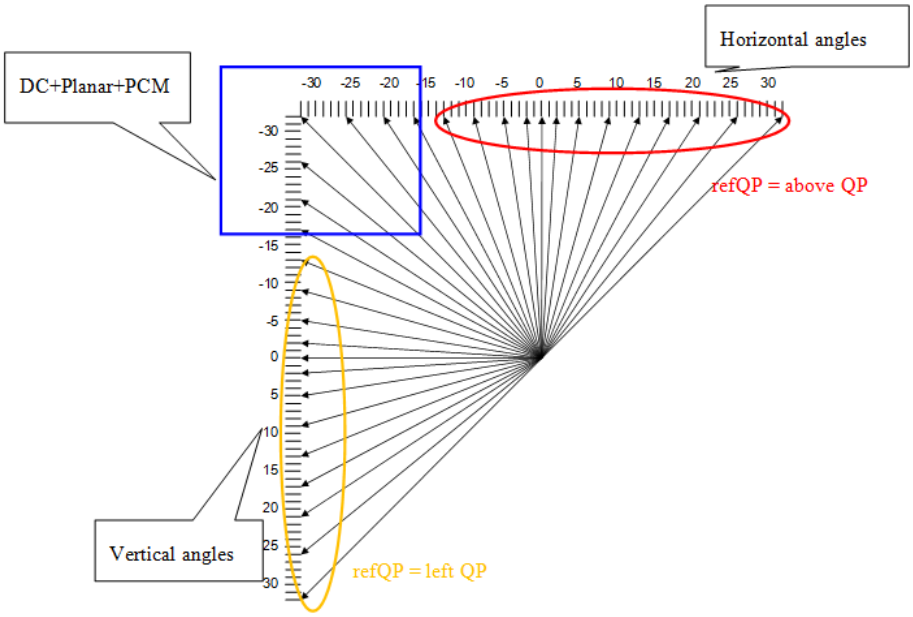


Figure 56: The intra mode direction of the TLCU is used to select the QP prediction candidates.

c) Motion vector predictors (AMVP) based QP prediction:

In [91], we proposed the following new QP prediction algorithm for the case where the TLCU is coded in Inter mode.

First, we consider the Top-Left Prediction Unit (TLPU) in this TLCU (see Figure 55). The motion vector of the list L0 of the TLPU has been derived either from one neighboring Prediction Unit (PU) according to the AMVP algorithm (as presented in section 2.1.3), from a co-located motion vector in the reference frame (a.k.a. Temporal Motion Vector Prediction, TMVP) or from a default vector predictor. In the first case, denote QP_{P0} the QP of this neighboring PU, else QP_{P0} is not available. In case of bi-prediction, we proceed similarly to derive QP_{P1} from the motion vector of the list L1. Next, QP_P is derived as follows:

- if both QP_{P0} and QP_{P1} are available: $QP_P = ½ (QP_{P0} + QP_{P1})$
- if only QP_{P0} is available (or only QP_{P1}): $QP_P = QP_{P0}$ (or $QP_P = QP_{P1}$ respectively)
- if QP_{P0} and QP_{P1} are both not available: the algorithm (a) is used.

6.1.2 Experimental results

These algorithms have been implemented and tested on top of HM5.0. Since the bit streams may contain coded frames with both CU coded in Intra or in Inter, we tested four different combinations of the QP prediction algorithms (a), (b) and (c) (Table 18):

- Test 1: Use of the proposed algorithm (c) for CU coded in Inter mode, else use (a) for other cases or if (c) is not applicable.
- Test 2: Use (b) algorithm for CU coded in Intra mode, else use (a) for other cases or if (b) is not applicable.
- Test 3: Use (b) algorithm for CU coded in Intra mode and (c) algorithm for CU coded in Inter mode, else use (a) for other cases or if (b),(c) is not applicable.
- Test 4: Use (a) algorithm for all cases.

The anchors are the TM5 implementation, with QP adaptation method modulation range ±12. The QG size is 3 (*diff_cu_qp_delta_depth*=3). The tested configurations are Random Access with hierarchical GOPs (RA) and Low Delay with B pictures (LB), using the JCT-VC regular test sequences. The BD-rate gains (Bjontegaard) compared to anchors results, as well as the relative delta-QP (dQP) coding cost bit increase are depicted in Table 18 (detailed results are available in [91]). The encoding/decoding times are unchanged.

Table 18: Performance of several QP prediction algorithms.

	CU is intra (Intra dir. Neighbors available	CU is inter (AMVP predictors available)	!available	RA		LB	
				Y	dQP	Y	dQP
Test 1	a	c	a	-0,2%	-5,3%	-0.2%	-4.1%
Test 2	b	a	a	-0.3%	-6.9%	-0.2%	-4.7%
Test 3	b	c	a	-0.3%	-6.5%	-0.2%	-4.3%
Test 4	a	a	a	-0.3%	-5.6%	-0.2%	-4.4%

6.1.3 Conclusion

The Table 18 shows that the algorithms have quite equivalent performance, both in terms of BD-rate gains and bits saving in the CU headers (dQP). The algorithm (c) does not bring any clear improvement for CU coded in Inter. Deeper analysis confirms that the algorithm (a) is slightly better than (c) even for sequences with a lot of motion (3 basket-ball and 2 race-horses sequences).

Given the very low complexity of the QP prediction based on averaging neighboring QPs (a), it has been adopted by MPEG in the HEVC standard.

6.2 Perceptually Adapted Video Encoding

The choice of the QP to be used for encoding every block in a frame can be solved using an RDO[5]-based approach. One common method used is the *Lagrangian*: for each Coding Unit, several coding modes (m) and QPs (q) candidates are tested, the best one (finally retained) is the one minimizing the Lagrangian function (35):

$$\arg\min_{(m,p)} L_\lambda(m,q), \qquad \text{where } L_\lambda(m,q)=D(m,q)+\lambda(q).R(m,q) \qquad (35)$$

where: $L_\lambda(m,q)$ is the Lagrangian value for the mode m and the quantizer q,
D is the distortion (ex: SAD[6]),
R is the coding cost (ex: number of bits).

The Lagrange multiplier λ is function of the quantizer q [103]. This method is well suited for open-loop and off-line encoding applications, where the overall bit-rate is not strictly constrained. If a particular bit-rate is targeted, alternative methods such as multi-pass encoding or rho-domain [92] algorithms may be preferred.

For real-time applications, simplified methods that allow testing only a subset of modes (early termination strategies) are generally used [93], combined with frame complexity estimation (obtained via activity measurement or with first pass encoding) (cf. 6.1).

Distortion measurement

One key parameter of this model is the distortion measurement (D in (35)). One commonly used video compression distortion metric is the Peak Signal to Noise Ratio (PSNR)(see also section 0). The PSNR has the advantage to be simple to implement, but it does not perfectly correlate with a perceived visual quality due to the non-linear behavior of the Human Visual System (HVS).

The amount of distortion perceived by a human observer differs from one region to another according to its particular local content (luminance, gradient, colour, spatio-temporal features, motion,...). This subjective perception can be explained/predicted by considering some psycho-visual properties of the Human Visual System.

In [102] we implemented a HVS module based on the spatio-motion contrast sensitivity curves presented in (section 1.1.3). The spatial frequencies are extracted locally with a Laplacian pyramid decomposition using Gaussian filters. Similar pyramid decomposition is performed on the motion data using median filters in such a way that the local velocity information is available at each resolution level of the pyramids. If the motion data can be derived from the motion vectors contained in the bit-stream, better results are obtained when using a standalone motion estimator module.

This process is carried out both for the original and the reconstructed/decoded frames. Next, for each level of the pyramid, the difference samples between the original and reconstructed frames are weighted with the appropriate spatio-motion contrast sensitivity weight (see S.Daly curves in Figure 5 [99][100]). Finally, all the pyramid levels weighted difference samples are summed after being re-scaled properly in order to get back to the original full resolution. The resulting data is called a "*Quality-map*" (Figure 57).

[5] RDO : Rate-Distortion Optimization.
[6] SAD: Sum of Absolute Differences.

Figure 57: Original (left) and corresponding quality maps (right) (higher luminance means higher visibility).

Optimal quantization re-allocation

The principle is to locally adjust the encoding parameters in order to obtain a homogeneous perceived quality on the whole picture, without modifying the overall encoding cost of the picture. Since an evaluation of the (local) perceived quality of encoded pictures is now available with *Quality-maps*, the basic idea is to use this information for setting the quantization step-size locally (for each macro-block or coding unit). The aim is to obtain uniform *Quality-maps* as far as possible, at constant bit rate (*RefPictCost*) through an iterative encoding algorithm (Figure 58).

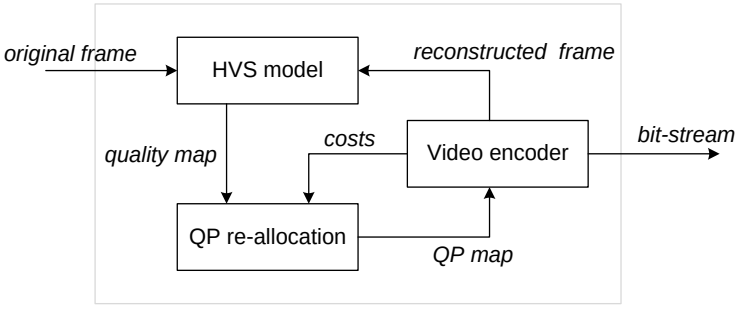


Figure 58: Principle of the iterative QP re-allocation system.

For each macro-block (MB) number k, a predictive model is built, relating the local subjective Quality ($Qmap_{k,n}$) and the MB encoding cost ($Cost_{k,n}$) with the local QP value ($QP_{k,n}$), after the iteration n:

$$\begin{cases} Qmap_{k,n} = A_{k,n} \times QP_{k,n} + C_1 \\ Cost_{k,n} = B_{k,n} \times QP_{k,n} + C_2 \end{cases} \quad (36)$$

where: $Qmap_{k,n}$ is the averaged quality-map over the macro-block k at iteration n,
$Cost_{k,n}$ is the number of bits used to encode the macro-block k at iteration n,

$QP_{k,n}$ is the quantization parameter for macro-block k at iteration n,
$A_{k,n}$ and $B_{k,n}$ are parameters computed after each iteration (see below),
C_1 and C_2 are parameters of the model.

The parameters of the predictive model (36) are up-dated after each iteration with equations (37) as follows (if $QP_{k,n}$-$QP_{k,n-1} \neq 0$):

$$A_{k,n} = \frac{Qmap_{k,n} - Qmap_{k,n-1}}{QP_{k,n} - QP_{k,n-1}}; \quad B_{k,n} = \frac{Cost_{k,n} - Cost_{k,n-1}}{QP_{k,n} - QP_{k,n-1}} \qquad (37)$$

where: $Qmap_{k,n}$ and $Qmap_{k,n-1}$ are the averaged quality-map over the macro-block k at iterations n and $n-1$,
$Cost_{k,n}$ and $Cost_{k,n-1}$ are the number of bits used to encode the macro-block k at iterations n and $n-1$,
$QP_{k,n}$ and $QP_{k,n-1}$ are the quantization parameter for macro-block k at iterations n and $n-1$.

At this stage, one can predict the new $Qmap_{k,n+1}$ and $Cost_{k,n+1}$ local values for a given QP variation ΔQP_k. The estimated encoding cost variation (in number of bits) for the whole picture is given by:

$$\Delta \widehat{PictCost} = \sum_{k=0}^{N} B_k \times \Delta QP_k \qquad (38)$$

where N is the number of macro-blocks in the picture.

After each iteration n, one has to find the set $\{\Delta QP_k\}_{k=0..N}$ of the local QP variation to be applied in order to minimize the quantity:

$$\Delta CostErr = \Delta \widehat{PictCost} - (PictCost_n - RefPictCost) \qquad (39)$$

where $PictCost_n$ is the picture cost in bits at the n^{th} encoding iteration, in order to keep the picture cost as close as possible to the targeted *RefPictCost* value.

To avoid too large QP fluctuations that could be outside the validity model limits, we restrict the QP variation ΔQP_k to be in the range [-1; +1] only. Indeed, the most the QP_k change, the most the MB_k encoding mode may likely change (see *Lagrangian* (35)) and the model (36) may become obsolete. The determination of the $\{\Delta QP_k\}$ (module "QP re-allocation" in Figure 58) is done through another iterative process based on the prediction model (36):

1. Separate $\{\Delta QP_k\}$ in 3 subsets: $\{\Delta QP_k\}^+/\{\Delta QP_k\}^-$ respectively grouping the K^+/K^- relative portion of MB having the higher/lower quality values, and $\{\Delta QP_k\}^0$ grouping the other MB. The values of $\{\Delta QP_k\}^+$, $\{\Delta QP_k\}^-$, $\{\Delta QP_k\}^0$ are set to $+1$, -1 and 0 respectively.
2. Calculate $\Delta \widehat{PictCost}$ with (38) and the predicted cost variation $\Delta CostErr$ with (39).
3. If $\Delta CostErr$ increases, it means the set $\{\Delta QP_k\}^-$ is too large and must be decreased by reducing K^-, while increasing K^+ (and vice versa). Then go to step 1.
4. This iterative loop is stopped when the predicted standard deviation of $\{Qmap_{k,n}\}$ (36) is no more decreasing.

The encoding iteration (Figure 58) terminates when the standard deviation of the Quality-maps is no more decreasing: then the set of $\{QP_k\}$ is considered as optimal and the picture quality is almost uniform on the whole picture.

Experimental results

This algorithm was successfully tested on several video sequences with MPEG2 encoder [102]. Generally, the convergence is reached in less than 10 encoding iterations. The subjective quality improvement is confirmed visually.

An example is depicted in Figure 61 with a video sequence where a horse is tracked by the camera. Due to the relative motion of the moving background, the quality maps emphasize errors on the horse, because the HVS Contrast sensitivity curves tend to reduce (coding) artefacts visibility in moving areas (Figure

59). The final QP map shows the lower QPs have been allocated to the horse region to the detriment of the background (Figure 60).

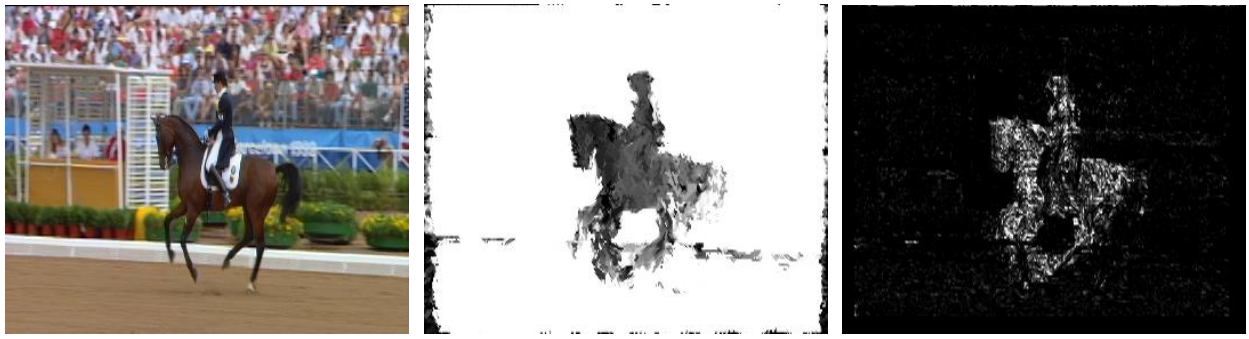


Figure 59: Original (left), motion field (center), quality-map (right).

Figure 60: Quality-map at first iteration (left), quality-map at last iteration (center), QPs last iteration (right).

Figure 61: Part of decoded frame with uniform QP (left) and with locally adapted QP with proposed method (right).

Part III – Video Coding methods enabling post-creation streams adaptation

We have seen that the video codecs implement techniques, such as rate control and quantization, to adapt the bit-stream production to the bit rate constraint in general (see 6.2). It allows creating bit streams perfectly tailored to the end device (decoder) capability, as well as to the distribution bandwidth requirements. Straightforward use cases are the point-to-point applications with real-time encoding of the video, such as video conferencing.

However, the one-to-many use cases are the most deployed applications today. If TV broadcasting is probably the most popular real-time encoding scenario, the video streaming of off-line encoded streams is generating the biggest amount of video data, with applications such as VOD services and video repository sharing (e.g. YouTube). In these cases, it is not realistic to deploy real-time codecs with associated computing resources, given the tremendous quantity of clients. As well as it is not efficient to re-encode the same content each time it is requested.

Nevertheless, for the one-to-many case where a single (or a few) video stream instance is generated, a level of adaptation is still required, because the access link and the video receivers capabilities may differ.

In this part, we will present some studies related to the post-creation video stream adaptability issue in general. Different levels of adaptability exist such as backward compatibility with legacy when deploying a new service (colour space, frame size or frame rate) or distribution locally tailored to the network bandwidth access for example. In some cases, several levels of adaptability can be gathered together (ex: colour scalability and spatial scalability). The video compression technologies and standards should provide the video market and the digital industry with a set of tools they can select or combine appropriately to fulfill their own requirements.

In the first section, we will consider several studies to improve the performance of stream switching systems, which is a technique used by most of the VOD applications today. It has the advantage to scale up easily while requiring relatively minimal server ressources. The last section will address the video signal colour space adaptability using the video scalability extension of HEVC. It could be applied to broadband distribution or next generation of Blu-ray Disc for example.

7. Stream-Switching

From a technical point of view, the scalable video coding is probably the most elegant and efficient technique for one-to-many (ex: broadcast) applications. However, it induces some system burden at encoder and transport levels, and adds some complexity at the decoder side. The choice of the bit-rates per layer may be difficult since the quality of the reconstructed base layer has major importance on the coding efficiency of the dependent (enhancement) layers. Even if the SHVC codec design factorizes the HEVC decoder module, it should instantiate as many decoders as layers. Despite many SoC[7] integrate multiple hardware decoders, the market is still reluctant to adopt scalable video coding based scheme. The stream-switching is an alternative technique that has proven its efficiency particularly for VOD services.

In a nutshell, there are three widely used ways to stream a video: Progressive Download, RTMP/RTSP Streaming, and Adaptive HTTP Streaming.

In Progressive download, one stream is transmitted from the beginning to the end. The receiver can start to play video as soon as sufficient data is buffered. It is the easiest to implement, but it is not able to change the quality of the video mid-stream: once the download starts, the video quality is locked. If the bandwidth link increases or if the player switches to full screen rendering, the video will stay low quality or low resolution.

To cope with these limitations, video streaming techniques allows synchronizing video data with player output. Upon request, the video server will send the video data corresponding to the actual receiver needs, including the client preference possibly (quality, resolution). This synchronization allows for dynamic adaptation of the video service to the network bandwidth or the player capability/needs.

RTMP and RTSP implement such streaming techniques, but need relatively complex servers for "pushing" the video data. They are more and more preferred "pull" mode systems that alleviate the server load, letting the receiver managing and requesting the video data chunks it needs. This is the principle adopted by the adaptive HTTP streaming protocol and the new emerging MPEG DASH standard [111].

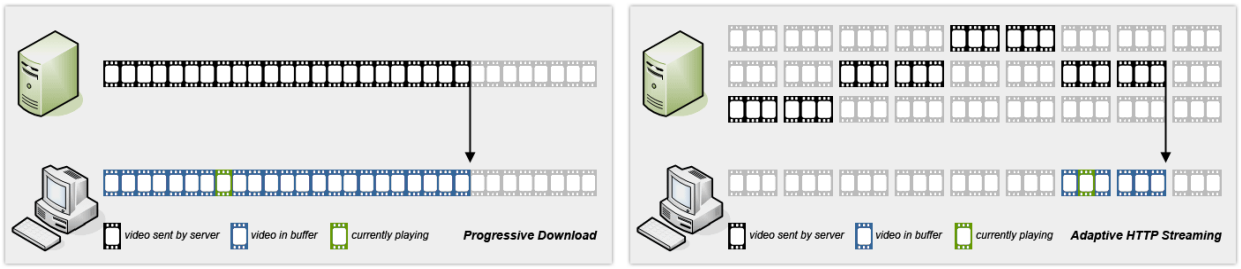


Figure 62: Progressive download (left) and Adaptive streaming with stream switching (right).

Practically, the servers maintain several pre-encoded instance of the video sequences, fragmented into chunks in order to the client may dynamically switch from one instance to another according to its actual needs.

In this section, we present two studies related to the video stream-switching techniques, whose major challenge is to perform the transition in-between different streams as seamless as possible for the end user.

The first one addresses the smoothness transition of stream switching in terms of video coding efficiency in case the frame size changes. The second one evaluates a stream-switching scheme based on the insertion of pre-encoded frames at switching point to increase both the flexibility and the BD-rate coding gains [134].

[7] System on Chip.

7.1 Adaptive Resolution Coding

In today applications using video streaming, the video is frequently re-scaled for a number of reasons, such as to maximize perceptual quality in adverse or varying network conditions or at the initiative of the end user.

However, in AVC or HEVC, prediction across resolutions is not permitted and an IDR (see 2.1.5) frame (and sequence start parameters with SPS, PPS…) must be sent to re-initialize the stream. Although IDR frames break dependencies, this is not necessarily a good thing for error resilience, because the cost (number of bits) of the Intra only encoded frames is significantly higher than equivalent Inter P or B frames in general. To minimize these problems, the IDR can be sent at low quality, using a similar number of bits to a P or B frame, but it will take a significant time to return to full quality for the given resolution.

ARC in HEVC

The concept of Adaptive Resolution Change (ARC) in HEVC – adapting resolution dynamically in response to circumstances, without inserting IDR – has been introduced in the context of HEVC by T.Davies [104]. It showed gains between 6.9%-7.4% when using ARC compared to inserting an IDR. Further studies reported potential gains between 3% and 30%, depending on the original frame size and configuration, using one empirical resolution switch selection method [105]. In these experiments and in the following, dyadic down-sizing (size divided by 2 in horizontal and vertical directions) only is considered for simplicity, but it could be extended to other ratio.

However, in these previous works, both resolutions (low and high) may be stored in the Decoded Pictures Buffer (DPB) for every reference pictures (Figure 63). This increases the decoder memory size requirements. The prediction is built using the reference picture with same resolution as the current frame. The JCT-VC identified this limitation as a major issue regarding possible adoption of ARC in HEVC [106].

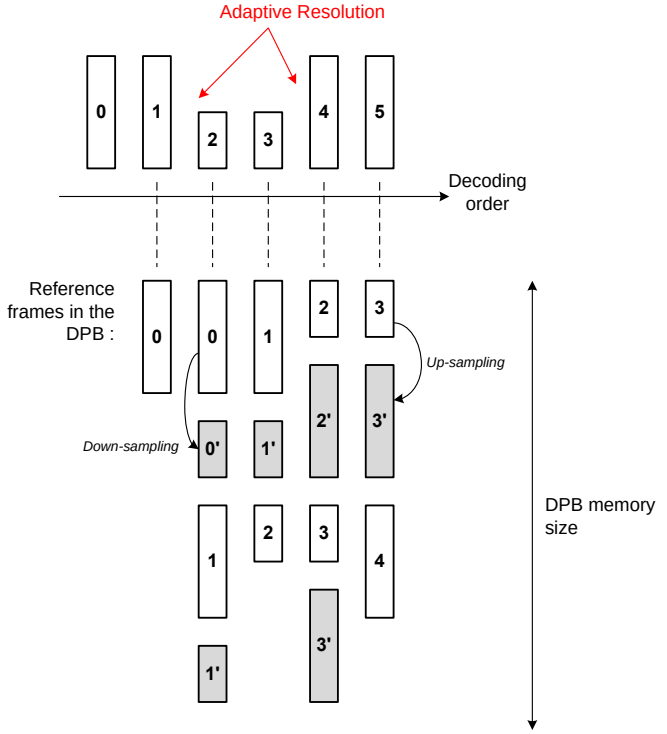


Figure 63: In previous ARC implementation, the DPB may contain both the reconstructed pictures marked as reference (in white) and the corresponding down-sampled (resp. up-sampled) resolutions (in grey). In this example, the two last decoded pictures are used as reference, and the pictures in the DPB are re-scaled to the current picture resolution.

ARC using single resolution in DPB

In order to improve the ARC scheme in HEVC, we have proposed in [107] a technique that allows storing only one single resolution of the reconstructed frames in the DPB. This technique is presented and discussed in this section. There is no additional down-sampling (or up-sampling) of the reconstructed reference frames at the decoder side. However, both resolutions are still necessary at the encoder side for the motion estimation.

In HEVC, the motion compensation (MC) used to build the inter-prediction supports fractional values of motion vectors (1/4 pel precision for Luma and 1/8 pel precision for Chroma). Denote ($frac_x$,$frac_y$) as the non-integer part of the motion vector, the MC interpolation is computed with two separable filters: the horizontal filter first $MCI_H(frac_x)$ and the vertical filter next $MCI_V(frac_y)$ (Figure 64).

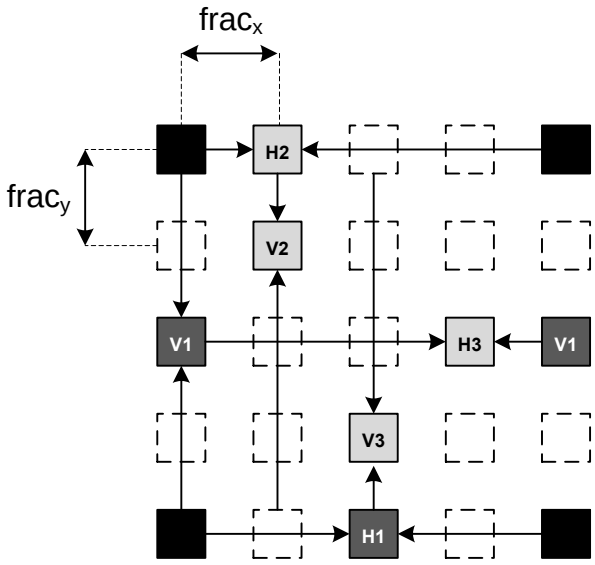


Figure 64: The fractional position ($frac_x$,$frac_y$) of the MC interpolated samples (in grey) with respect to the original samples in the reference picture (in black), in case of ¼ pel precision.

In ARC proposals [104][105], a set of filters are used for up- or down-scaling the reference pictures that will be needed across a resolution switch point. They are horizontal $SC_H(\theta_x)$ and vertical $SC_V(\theta_y)$ separable filters too. The parameters (θ_x, θ_y) are the horizontal and vertical phases respectively, which are the relative position of the pixel to interpolate in the original picture grid (Figure 65). These filters are applied on the reconstructed pictures in order to store in the DPB reference pictures at all possible resolutions.

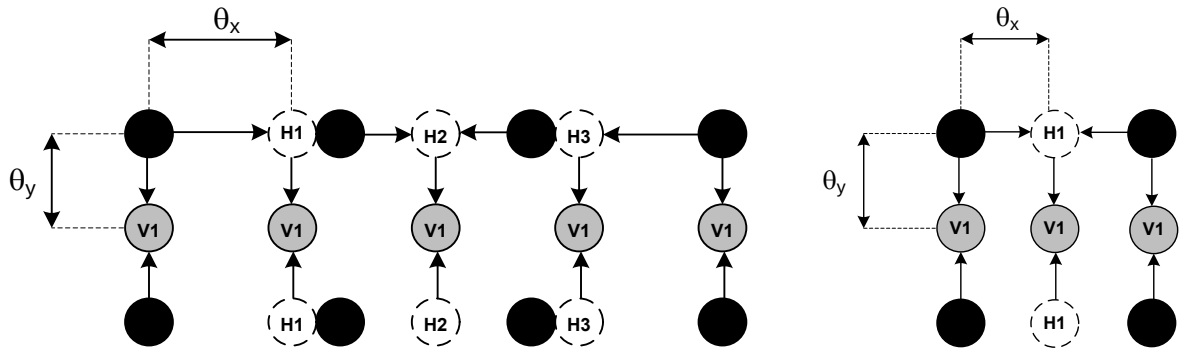


Figure 65: Examples of position (θ_x, θ_y) of the pixels to interpolate (in grey) (up-sampling) in the original grid (in black). Left: up-sampling by 5/4 horizontally, by 2 vertically. Right: up-sampling by 2 both horizontally and vertically.

Then the whole process for building inter-prediction is equivalent to the convolution of these 4 filters (Figure 66a) as follows:

$$pred(x_{cur},y_{cur}) = [MCI_V(frac_y) \; o \; MCI_H(frac_x) \; o \; SC_V(\theta_y) \; o \; SC_H(\theta_x) \;] \; (refPic(x_{ref},y_{ref})) \quad (40)$$

where: $refPic(x_{ref},y_{ref})$ is the reference picture sample value at position (x_{ref},y_{ref}),
$pred(x_{cur},y_{cur})$ is the prediction sample value at position (x_{cur},y_{cur}) in the current picture,
$MCI_H(frac_x)$ and $MCI_V(frac_y)$ are the horizontal and vertical motion compensation filters,
$(frac_x,frac_y)$ are the non-integer part of the motion vector,
$SC_H(\theta_x)$ and $SC_V(\theta_y)$ are the horizontal and vertical up- or down-sampling filters,
(θ_x, θ_y) are the horizontal and vertical phases of the sample (x_{ref},y_{ref}).

The property of separable filters being that they can be permuted, (40) is equivalent to (41):

$$pred(x_{cur},y_{cur}) = [MCI_V(frac_y) \; o \; SC_V(\theta_y) \; o \; MCI_H(frac_x) \; o \; SC_H(\theta_x) \;] \; (refPic(x_{ref},y_{ref})) \quad (41)$$

$MCI_H(frac_x) \; o \; SC_H(\theta_x)$ and $MCI_V(frac_y) \; o \; SC_V(\theta_y)$ are equivalent to two separable filters $G_H(frac_x,\theta_x)$ and $G_V(frac_y,\theta_y)$ as depicted in Figure 66b.

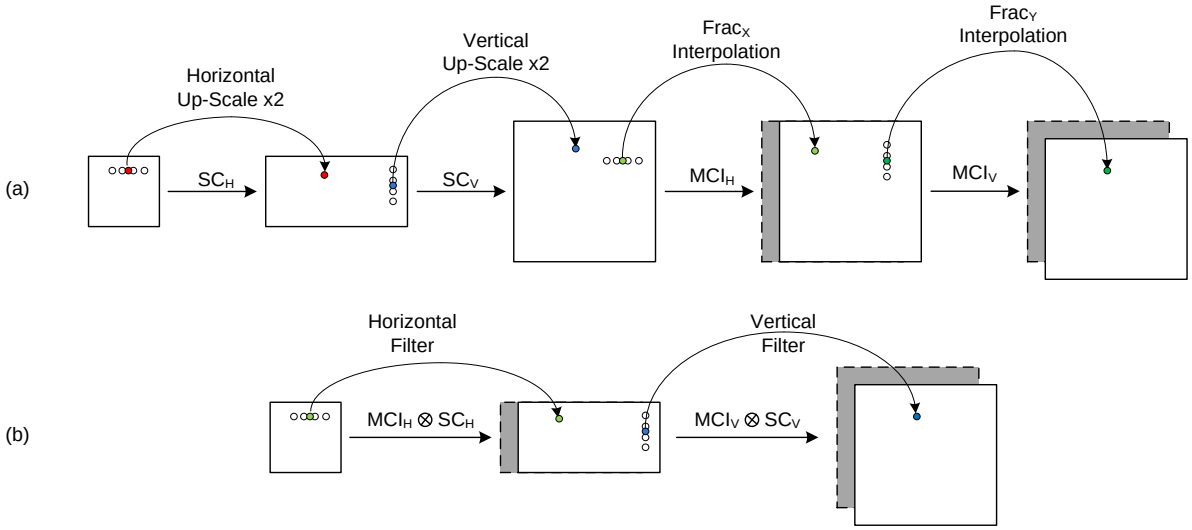


Figure 66: The convolution of the 4 separable filters is equivalent to 2 separable (Horizontal and Vertical) filters.

For each sample (x_{cur},y_{cur}) of the Prediction Unit (PU), the corresponding position in the reference picture grid (Px_{ref},Py_{ref}) (in grey in Figure 65) are determined with (43) given below. Next, (θ_x,θ_y) are computed using (45).

The parameters ($frac_x$,$frac_y$) are computed with (42) as usual in HEVC. In HM5.0, at the inter-prediction stage, interpolation filters are replaced with the filters $G_H(frac_x,\theta_x)$ and $G_V(frac_y,\theta_y)$ operating directly on the samples ($x_{ref}(x_{cur})$,$y_{ref}(y_{cur})$) of the reconstructed reference pictures (44). The Equations (42,43,44,45) are given by:

$$\begin{cases} \mathrm{MV_X} = \mathrm{iMV_X} + \mathrm{frac_X}, \text{ with } \mathrm{iMV_X} = \mathrm{int}(\mathrm{MV_X}) \\ \mathrm{MV_Y} = \mathrm{iMV_Y} + \mathrm{frac_Y}, \text{ with } \mathrm{iMV_Y} = \mathrm{int}(\mathrm{MV_Y}) \end{cases} \quad (42)$$

$$\begin{cases} \mathrm{Px_{ref}}(\mathrm{x_{cur}}) = \frac{\mathrm{Sx_{ref}}}{\mathrm{Sx_{cur}}} \times (\mathrm{iMV_x} + \mathrm{x_{cur}}) \\ \mathrm{Py_{ref}}(\mathrm{y_{cur}}) = \frac{\mathrm{Sy_{ref}}}{\mathrm{Sy_{cur}}} \times (\mathrm{iMV_y} + \mathrm{y_{cur}}) \end{cases} \quad (43)$$

$$\begin{cases} \mathrm{x_{ref}}(\mathrm{x_{cur}}) = \mathrm{int}(\mathrm{Px_{ref}}(\mathrm{x_{cur}})) \\ \mathrm{y_{ref}}(\mathrm{y_{cur}}) = \mathrm{int}(\mathrm{Py_{ref}}(\mathrm{y_{cur}})) \end{cases} \quad (44)$$

$$\begin{cases} \theta_\mathrm{x}(\mathrm{x_{cur}}) = P\mathrm{x_{ref}}(\mathrm{x_{cur}}) - \mathrm{x_{ref}}(\mathrm{x_{cur}}) \\ \theta_\mathrm{y}(\mathrm{y_{cur}}) = \mathrm{Py_{ref}}(\mathrm{y_{cur}}) - \mathrm{y_{ref}}(\mathrm{y_{cur}}) \end{cases} \quad (45)$$

where: (x_{cur},y_{cur}) is the position of one PU sample in the current picture,
(MV_X, MV_Y) is the motion vector associated with this PU,
(iMV_X, iMV_Y) and ($frac_X$,$frac_Y$) are the integer and the fractional part of (MV_X, MV_Y) respectively,
(Sx_{cur}, Sy_{cur}) and (Sx_{ref}, Sy_{ref}) are the PU size expressed in number of samples, in the current and the reference picture respectively,
(Px_{ref}, Py_{ref}) is the position of the motion compensated PU sample in the reference picture,
(x_{ref}, y_{ref}) is the integer part of (Px_{ref}, Py_{ref}),
(θ_x, θ_y) is the up/down filter phase.

The computation of the prediction unit in the case of Low-to-High resolution change (reference is Low resolution and current is High resolution) is depicted in Figure 67. In the first step, $G_H(frac_x,\theta_{xu})$ and $G_H(frac_x,\theta_{xv})$ are applied to compute temporary samples u and v from sample s in reference. Next, $G_V(frac_y,\theta_{yu})$ and $G_V(frac_y,\theta_{yv})$ are applied with (u,v) to compute (pa,pb,pc,pd).

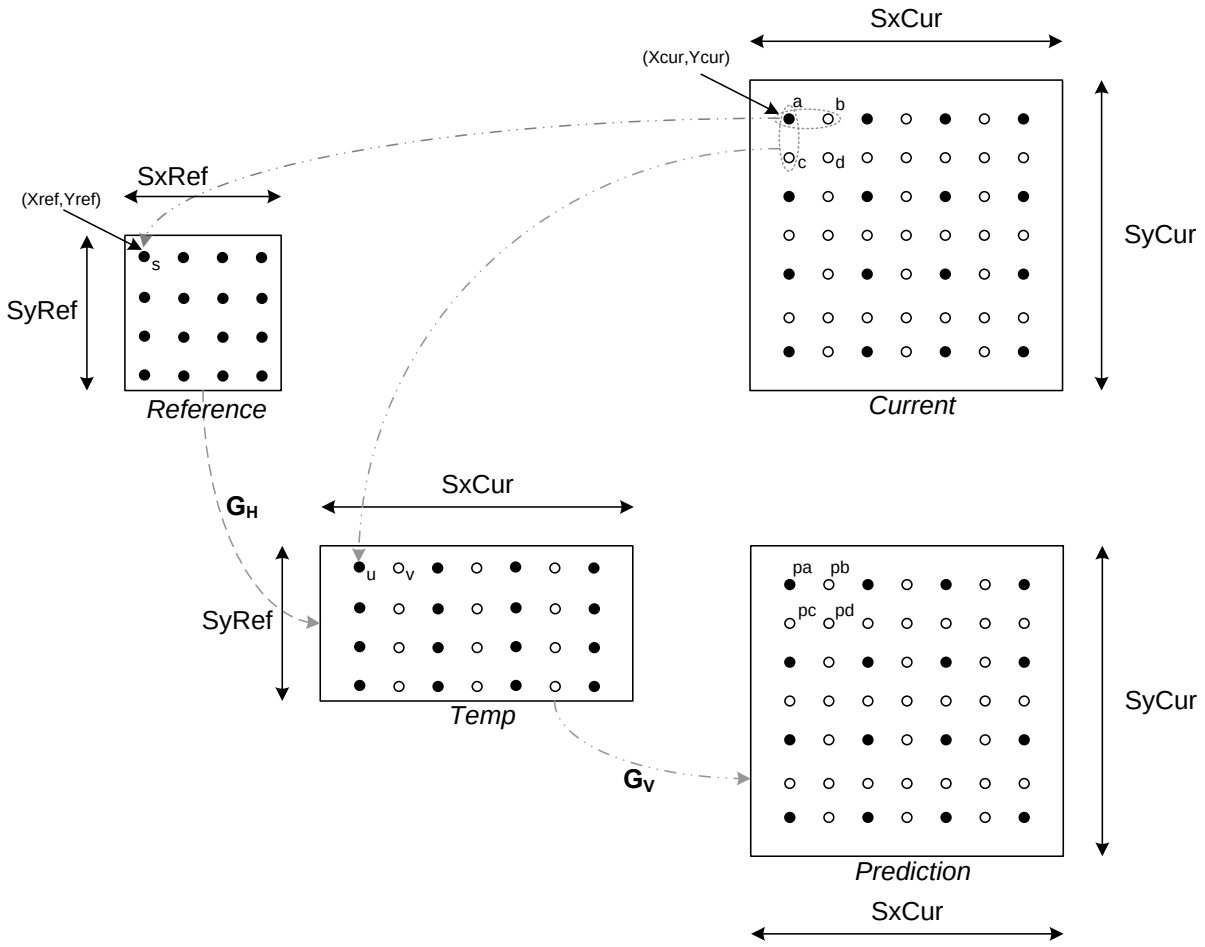


Figure 67: Computation of the PU in case of Low-to-High resolution change.

Experimental results

This method has been implemented into the ARC-HM5.0 software used as reference for ARC studies and provided in [106]. Two resolution change scenarios have been considered: High-to-Low and Low-to-High resolution changes. Since the prediction of Low (or High) resolution frames from High (resp. Low) resolution pictures has been modified only, the portion just around the resolution change point is considered.

The first 4 coded pictures are High (normal) resolution (resp. Low), the next ones are Low resolution (reps. High). In Low-delay P (LP) and Low-delay B (LB) configurations, 9 frames are encoded (Figure 69). In Random-Access (RA) configuration 16 frames are encoded (Figure 70). The content of the DPB at each decoding instant is depicted in Figure 69 and Figure 70, for (LP, LB) and RA cases respectively.

The results corresponding to High-to-Low resolution change scenario and to Low-to-High resolution change scenario are presented in Table 21 and Table 22 respectively (the full detailed results are available in [107]). Significant gains are obtained with High-to-Low cases, but one experiments gains for Low-to-High cases too.

These gains may be explained as follows. With the proposed method, one rounding stage is made only, whereas with previous method two rounding stages are performed: one after the up/down scaling and one after the interpolation filtering.

The number of taps and the number of bits (buffer accumulation before right shift) required to perform the filtering with G_H and G_V are depicted in Table 20, to be compared with the down/up sampling ARC filters and MC interpolation filters used in HM5.0 shown in Table 19. In both cases, they should be applied twice for horizontal and vertical filtering.

One limitation of our method is the number of taps of the filters G_H and G_V. However, they could be possibly reduced since many coefficient values are negligible, as depicted with $G_{HorV}(frac_{HorV}=2)$ in Figure 68.

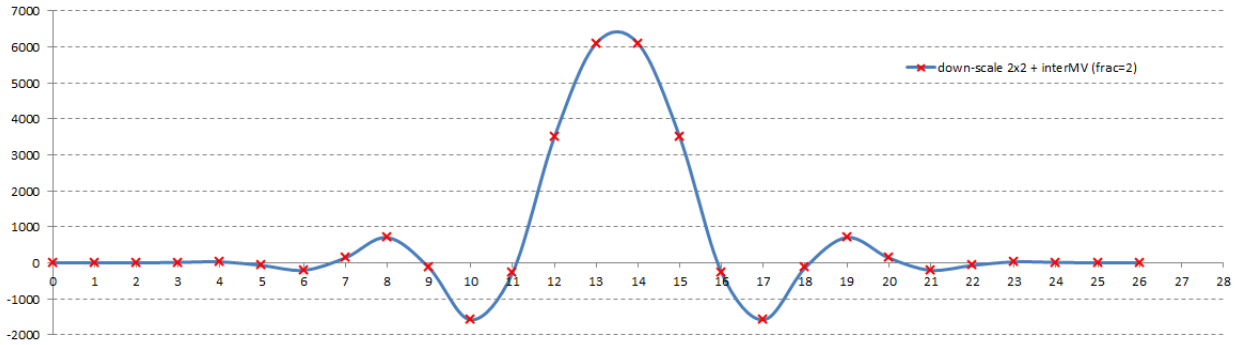


Figure 68: The downScale2x2_mvInter filter (frac=2).

Table 19: Down/up sampling filters and MC interpolation filters used in HM5.0-ARC .

downScale2x2_filter: num. of taps (num. bits)	
frac	0
Luma/Chroma	12 (8)

upScale2x2_mvInter_filter: number of taps (num. bits)		
Luma / Chroma	Phase 0	6 (7)
	Phase 1	6 (7)

mvInter_filter: number of taps (number of bits)								
frac	0	1	2	3	4	5	6	7
Luma	1 (6)	7 (6)	8 (6)	7 (6)				
Chroma	1 (6)	4 (6)	4 (6)	4 (6)	4 (6)	4 (6)	4 (6)	4 (6)

Table 20: The proposed all-in-one down/up sampling and MC interpolation filters (G_H,G_V) in HM5.0-ARC .

downScale2x2_mvInter_filter: num taps (nb bits)								
frac	0	1	2	3	4	5	6	7
Luma	12 (8)	24 (14)	26 (14)	24 (14)				
Chroma	12 (8)	18 (13)	18 (13)	18 (13)	18 (12)	18 (13)	18 (13)	18 (13)

upScale2x2_mvInter_filter: num taps (nb bits)								
frac		0	1	2	3	4	5	6	7
Luma	Phase 0	6 (7)	9 (13)	9 (13)	9 (13)				
	Phase 1	6 (7)	9 (13)	10 (13)	9 (13)				
Chroma	Phase 0	6 (7)	7 (12)	7 (12)	7 (12)	7 (11)	7 (13)	7 (12)	7 (12)
	Phase 1	6 (7)	8 (12)	8 (12)	8 (12)	8 (11)	8 (12)	8 (12)	8 (12)

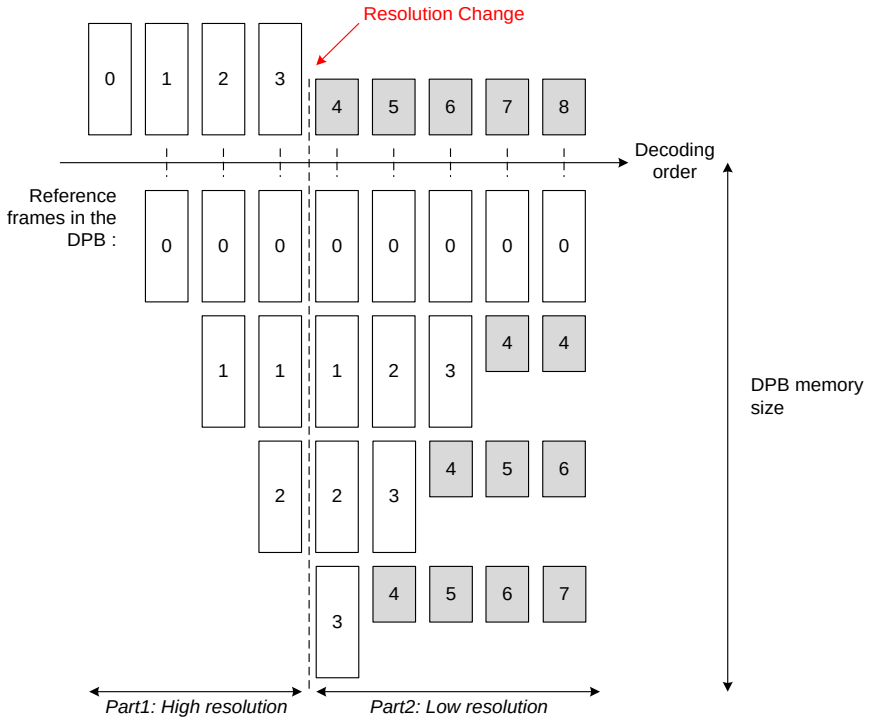


Figure 69: In LP and LB configuration, 9 first frames are coded (High-to-Low scenario).

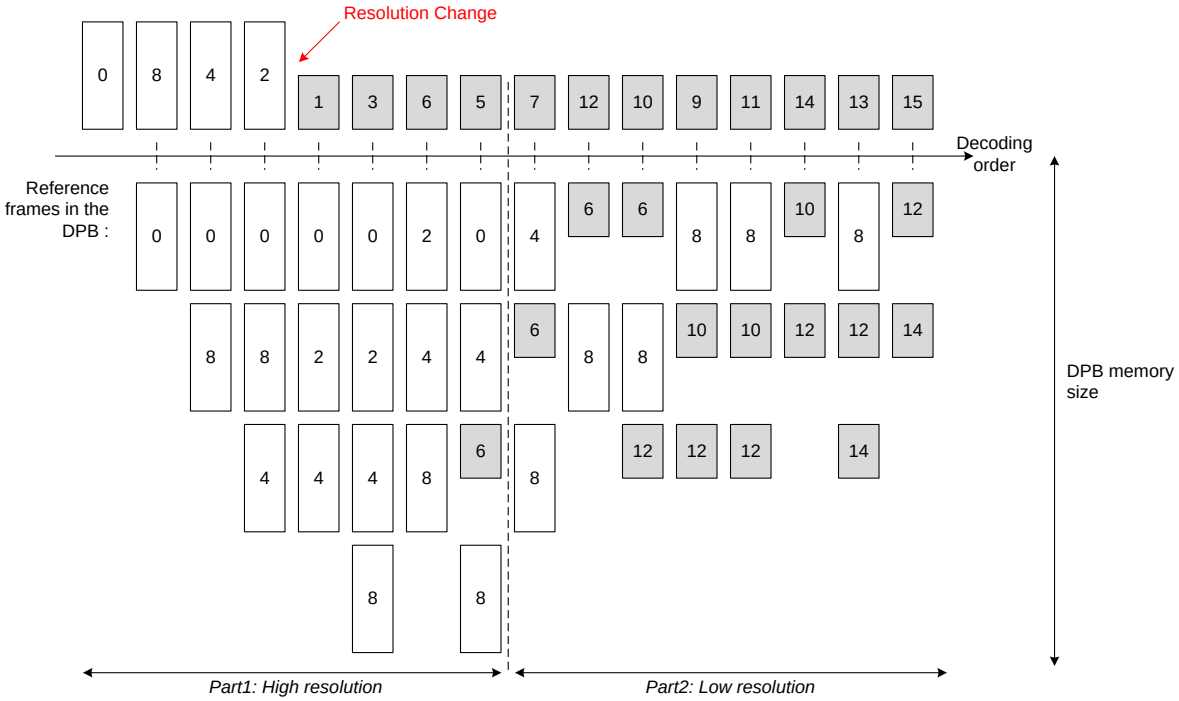


Figure 70: In RA configuration, 16 first frames are coded (High-to-Low scenario).

Table 21: ARC proposal vs reference ARC implementation (High-to-Low resolution change).

	Random Access HE			Low Delay B - HE			Low Delay P - HE		
	Y	U	V	Y	U	V	Y	U	V
Class A	-1,0%	-4,8%	-5,2%						
Class B	-1,7%	-6,7%	-6,4%	-1,0%	-3,7%	-4,0%	-1,1%	-5,2%	-5,0%
Class C	-1,1%	-2,9%	-3,4%	-0,8%	-2,7%	-1,9%	-0,5%	-2,3%	-1,5%
Class D	-2,3%	-4,7%	-4,5%	-0,8%	-2,4%	-1,8%	-1,0%	-2,8%	-2,4%
Class E				-2,8%	-9,0%	-6,1%	-3,6%	-9,9%	-7,1%
Class F	-2,8%	-5,0%	-4,2%	-1,4%	-3,1%	-2,1%	-1,0%	-2,4%	-0,6%
Overall	-1,6%	-4,9%	-4,9%	-1,3%	-3,9%	-3,1%	-1,3%	-4,3%	-3,2%
Enc. Time	128%			117%			112%		
Dec. Time	150%			131%			133%		

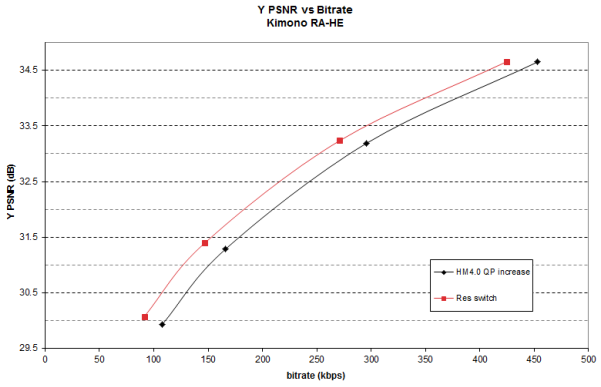
Table 22: ARC proposal vs reference ARC implementation (Low-to-High resolution change).

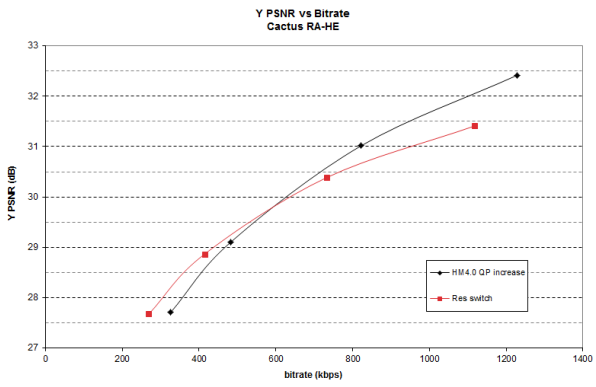
	Random Access HE			Low Delay B - HE			Low Delay P - HE		
	Y	U	V	Y	U	V	Y	U	V
Class A	-0,1%	-0,2%	-0,6%						
Class B	-0,3%	-1,2%	-1,0%	-0,1%	-0,7%	-0,9%	-0,2%	-0,7%	-0,9%
Class C	-0,2%	-0,7%	-0,2%	-0,2%	-0,5%	0,7%	-0,3%	-0,5%	0,6%
Class D	-0,2%	0,3%	1,4%	-0,2%	-0,7%	0,4%	-0,2%	-1,0%	-0,1%
Class E				-0,7%	-1,9%	-1,7%	-0,6%	-2,2%	-2,0%
Class F	0,0%	0,0%	-0,9%	0,2%	-0,1%	0,0%	-0,1%	0,6%	-0,4%
Overall	-0,2%	-0,5%	-0,1%	-0,2%	-0,7%	-0,3%	-0,3%	-0,7%	-0,5%
Enc. Time	119%			114%			111%		
Dec. Time	126%			125%			124%		

Discussion and Conclusion

In [105], the authors compare the performance of encoded down-sampled videos where the reconstructed frames are up-sampled, with traditional full resolution coding using HM4.0. We conducted same experiments with HM6.0. The results are very dependent on the particular video content as depicted in Figure 71 with 1080p sequences. For some sequences, the down-sampled compete favorably with original resolution coding (e.g. *Kimono*), for some the BD-rate curves show a cross-over point (e.g. *Cactus* and *Race-Horse*), while for others the full resolution is always better at all bit rates (e.g. *BQTerrace*). Then the benefit of ARC will depend on the video content.

The proposed method allows both reducing the DPB memory size and increasing the performance of ARC. However it was not adopted in HEVC mainly because the extension studies on scalability were just starting and it was anticipated that some SHVC features could bring equivalent functionality.





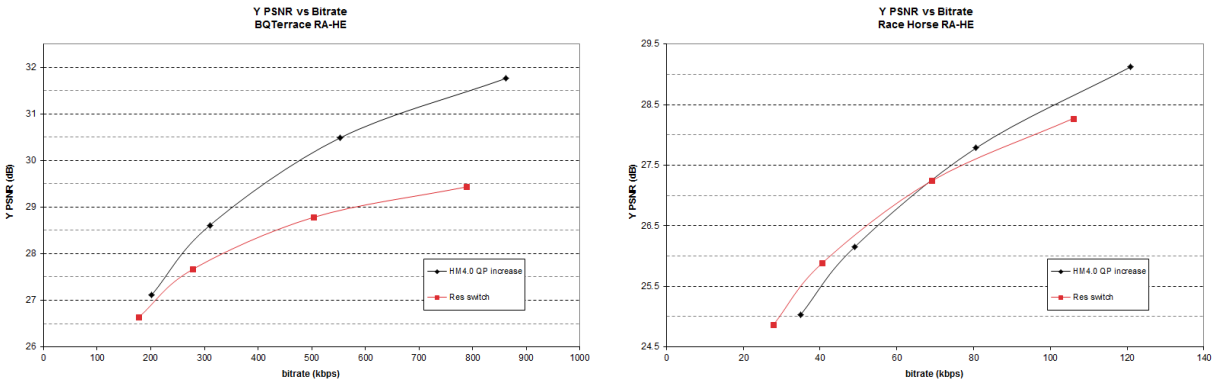


Figure 71: In red, the BD-rate curves of up-sampled reconstructed video (down-sampled encoded), in black the full resolution encoded video. The sequences Kimono (top-left), Cactus (top-right), BQTerrace (bot-left) and Race-Horse (bot-right) are encoded in RA-HE configuration with HM4.0.

7.2 Stream-switching with HEVC compliant with SHVC

In video streaming, the stream switching (SS) technique consists in switching from one encoded stream to another one to accommodate to the bandwidth variation or other end device limitation (e.g. decoding power). This technique is traditionally used by many VOD[8] services providers [111]. The several streams instances may differ from bit rate or frame resolution typically.

After switching, some drift or errors may occur, since the reconstructed reference pictures used at the decoder are not the same as the ones used by the encoder. This drift still remains until a Clear Random Access (CRA) picture (see 2.1.5), without any further reference to the previously decoded pictures beyond the Random Access Picture (RAP), is inserted in the bit-stream. In practice, many systems try to avoid this inconvenience by forbidding SS at non CRA points. However it introduces constraints which decrease the SS flexibility. One can reduce this limitation by adding more RAP pictures in the streams. But the coding efficiency of RAP pictures using intra modes only is significantly lower than the other uni-directional P or bi-predicted B pictures.

In H.264/AVC [68][114], SP-frames and SI-frames have been introduced [112]. They allow identical frames being reconstructed even if they are predicted with different reference frames than the ones used by the encoder. The coding efficiency of the SP-frames is higher than the I-frames in general. However switching from one bit-stream to another should be performed at these SP-frames locations only. Replacing RAP intra only frames with P or B frames plus regularly inserted SP-frames allows reducing the bitrate while maintaining equivalent reconstructed video quality and stream switching flexibility. However, the SP-frames are more complex to encode than other I,P,B frames and not all decoder devices implement the required Extended profile [114]. Another limitation of this approach is the bitrate overhead due to the regular insertion of the SP- and SI-frames [113] even if no switch occurs.

An alternative scheme is proposed in [115] to solve this coding overhead: secondary SP frames are introduced, but unlike [112], they are present in the transmitted bit-stream only when switching is performed. These frames contain the differences between the last LQ (Low Quality) decoded picture and its HQ (High Quality) version, which has not been transmitted. The reconstruction is not perfect due to the quantization/rounding mismatch, but the quantization level used for coding the differences can be set in order to minimize the drift.

Inspired from this approach, we propose in the following a mechanism allowing a flexible switching at any frame, of any type (I,P or B) in the bit-stream. The technique is based on the insertion of SHVC pre-encoded switching frames. Even if the drift is not totally removed, the quality of the decoded picture can be improved with limited bitrate overhead.

Switching strategies

In the absence of any additional data, if the switching doesn't occur at a RAP pictures, there is no other choice than switching from one quality pre-encoded video stream to another quality stream. This strategy is referred in the following as "basic switch".

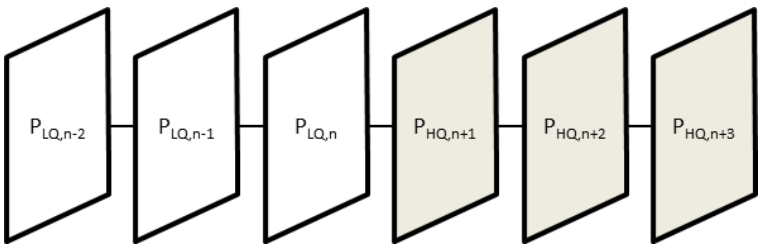


Figure 72: Basic stream switching (low delay P profile).

[8] Video On Demand.

In Figure 72, the following notations are used:

$P_{LQ,\,n}$: n^{th} picture encoded at Low Quality (LQ)

$P_{HQ,\,n}$: n^{th} picture encoded at High Quality (HQ)

As explained previously, in the case of up switching from LQ to HQ, a drift is present on the reconstructed HQ pictures until the next RAP frame. This drift comes from the inherent differences between the reconstructed LQ pictures and the reconstructed HQ pictures which are further used as references during the HQ decoding process. To reduce these artifacts, it is proposed to update in the decoded picture buffer (DPB) the pictures used as references.

As an example, let's assume that pictures with indices #n-1 and #n are used as references for the decoding process of pictures with indices #n+1 and next. In this case, we propose that the content of pictures $P_{LQ,n-1}$ and $P_{LQ,n}$ will be updated by the switching frames $E_{HQ,n-1}$ and $E_{HQ,n}$. Ideally, after this operation, if $E_{HQ,n-1}$ and $E_{HQ,n}$ are encoded without any loss, the pictures of the decoder DPB and marked as reference will be reconstructed identically to the ones used in the DPB of the HQ encoder. Any mismatch is then prevented.

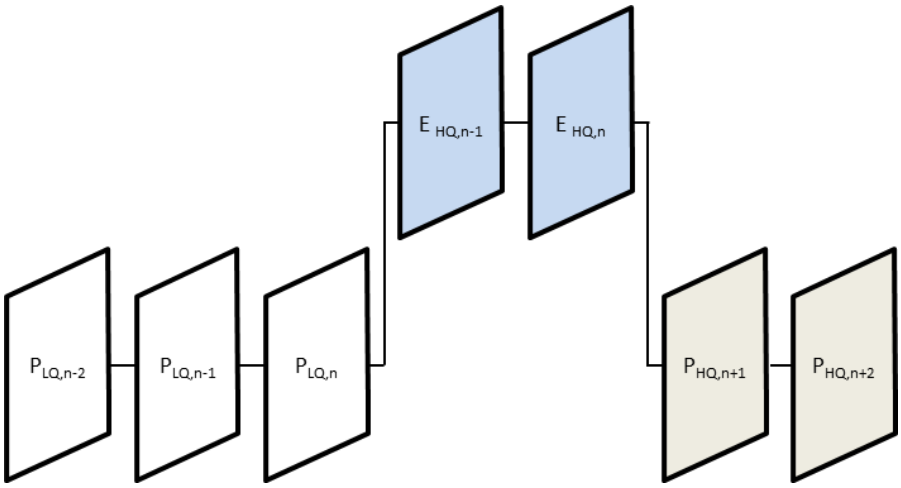


Figure 73: proposed stream switching.

A basic implementation could consist in encoding the switching frames in Intra mode, but the coding efficiency would be low. To improve it, we consider using the LQ decoded picture as prediction for encoding the switching frame since they are highly correlated with the HQ component (temporally co-located). The coding of these switching frames is performed by an SHVC encoder [21], using inter-layer prediction mode (ILP)(see 3). With this approach, the LQ stream corresponds to the SHVC base layer (BL) and the switching frames are encoded as the enhancement layer (EL) (Figure 74).

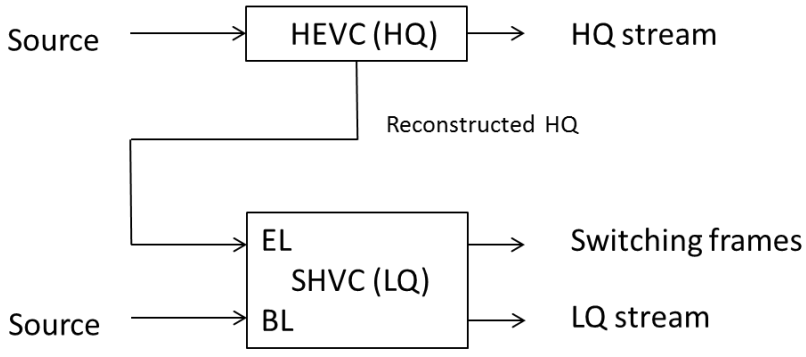


Figure 74: generation of the switching frames.

The ILP mode defined in SHVC uses the collocated base layer reconstructed block as prediction for the enhancement block to be coded. If the reconstructed BL and EL frame sizes are different, SHVC allows up-sampling the reconstructed BL samples.

Stream-switching: HLS

By forbidding the inter (intra layer) mode for the switching frames encoding, one guarantees that the decoder can correctly decode these switching frames, since they don't refer to any previously reconstructed HQ frames.

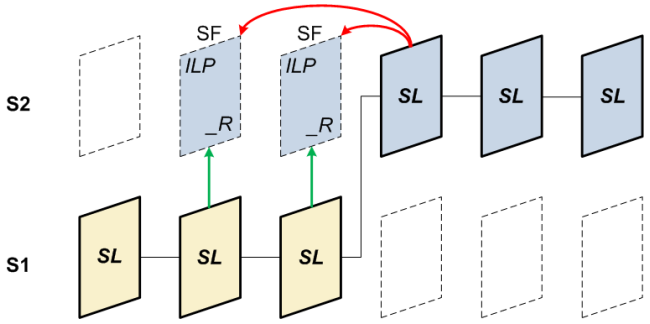


Figure 75: Example of stream-switching case.

The SHVC High Level Syntax (HLS) includes a set of syntax elements that enable signaling and encode such switching frames. In the VPS parameters, the following parameters could be used:

single_layer_for_non_irap_flag =1, *vps_base_layer_internal_flag* =1

> These flags indicate either all the VCL NAL units of an AU have the same *nuh_layer_id* value (this corresponds to the pictures marked as SL in Figure 75),
>
> or if two *nuh_layer_id* values are used by the VCL NAL units of an AU then the picture with the greater *nuh_layer_id* value is an IRAP picture (ILP prediction only) (this corresponds to the pictures marked as SF in Figure 75).

pic_output_flag =1

> In the slice header, it indicates the current frame is not kept in the DPB to be displayed (this corresponds to the pictures marked as SF in Figure 75, if the corresponding SL pictures should be rather displayed).

Switching parameters tuning

In the previous section, the generation of the switching frames has been described. It has been considered that the pictures further used as references must be updated. Ideally, all pictures used as references must be updated, and they must be strictly equal to their equivalent version in the HQ stream. This optimal configuration may require to lossless encode the switching frames and to transmit a huge amount of bits when the switch occurs. This sudden increase of bitrate can lead to buffer overflow or congestion at the encoder/decoder side and to annihilate the stream switching benefit. To reduce this overhead, the influence of two main parameters is considered.

The first parameter to be considered is the number of reference frames to be updated. The DPB size used by JCT-VC in both HEVC and SHVC test conditions [22] is 4 frames. This means that all these frames are potentially furtherly used as references when decoding subsequent pictures (in decoding order). One strategy to save bits is to update some but not all of them. For instance, the DPB frames can be sorted depending on how many pictures refer to them. Another strategy may consist in not updating the frame(s) with the lowest POC (Picture Order Counter). The number of frames to be updated can vary over the time and depends on the GOP structure used by the encoder.

The second parameter which has a high influence on the stream bitrate is the quantization parameter (QP) used for encoding the switching frames. A low QP value leads to a switching frame $E_{HQ,\,n}$ similar to the $P_{HQ,\,n}$ picture but not strictly equal in general, since the encoding process of $P_{HQ,\,n}$ may have used temporal prediction mode, which is not the case for $E_{HQ,\,n}$. Note that HEVC supports lossless coding. When QP is increased, the mismatch between $P_{HQ,\,n}$ and the updated reference frame tends to increase, but the overhead is reduced.

Thus, a compromise must be found on the encoder side between the coding cost of these switching frames and the benefit in quality on the reconstructed pictures.

These two parameters should be set jointly for the generation of the pre-encoded switching frames. The experimental results in next part investigate their combined effect on the performance.

Experimental results

The experiments have been achieved on the 1920x1080 Class-B sequences used by JCT-VC for SHVC [22] (*BasketballDrive, BQTerrace, Cactus, ParkScene*), with hierarchical GOP structure of 8 pictures length (cf. Random Access configuration parameters). A RAP picture is inserted every 2 seconds. The QP used for the LQ and HQ streams were QP_{BL}=40 and QP_{EL}=34 respectively. A ΔQP value of 6 levels guarantees a perceptible subjective quality difference between both streams.

In the simulated use case, the decoder first receives the LQ stream, the switch occurs later in the sequence. Several locations of the switching point relatively to the GOP structure have been tested, to cover all types of pictures of the GOP. In the proposed switching scheme, the switching frames are inserted just before switching to the HQ stream. In a pull mode scenario compatible with DASH, the decoder device would ask the server to send such additional data just before performing the switch. In the basic switching mode, used as reference, no additional data is inserted when switching.

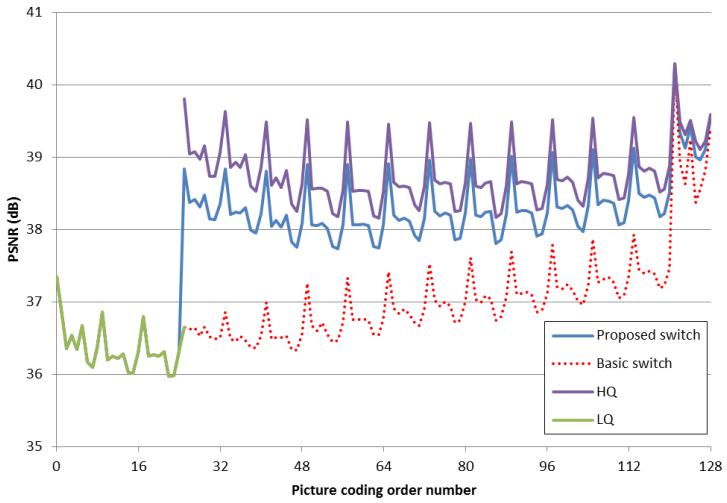


Figure 76: PSNR of the reconstructed switching from LQ (green) with the proposed method (blue) and with "basic" switch (red), the targeted HQ PSNR in purple.

The Figure 76 shows the PSNR measured on the reconstructed sequence for both switching modes. At the beginning of the sequence, the PSNR has the same value as the LQ PSNR. After the switch, the PSNR of the proposed method increases significantly, due to the insertion of the switching frames, and reaches a value at 0.6dB below the expected HQ PSNR only. This difference is explained by the non-exact reconstruction of the pictures in the DPB. Next the PSNR is converging towards the HQ curve (0.4dB difference). Finally, the PSNRs become equal to HQ when a RAP is inserted into the stream (frame num. 120).

The behavior of the basic mode differs significantly: the PSNR is converging more slowly to the HQ. A 2dB difference is still observed just before the RAP insertion.

Figure 77: basic switching (left), proposed switching (right).

Some experiments have been led which consisted in updating some but not all reference frames present in the DPB and considering different QP for the switching frames.

Using consistent values of these parameters allows modifying the bitrate of the switching frames and to investigate the gain brought by this technique in function of the bit-stream overhead. In Figure 78, the bitrate of the switching frames is computed relatively to the cost of the first RAP picture after the switch, which is in this case a CRA (Clear Random Access) picture. 1dB gain improvement is observed when the switching frames cost is equivalent to one CRA picture. It means that for an overall frame cost at stream switching point (including added switching data) equivalent to one CRA frame, the quality is increased by 1dB. This is true whatever the frame type is: the stream switching flexibility is increased because it is no more necessary to switch at CRA points. Another benefit is one can reduce the amount of CRA frames in the stream, and reduce the average bitrate of the LQ stream when no switch occurs consequently.

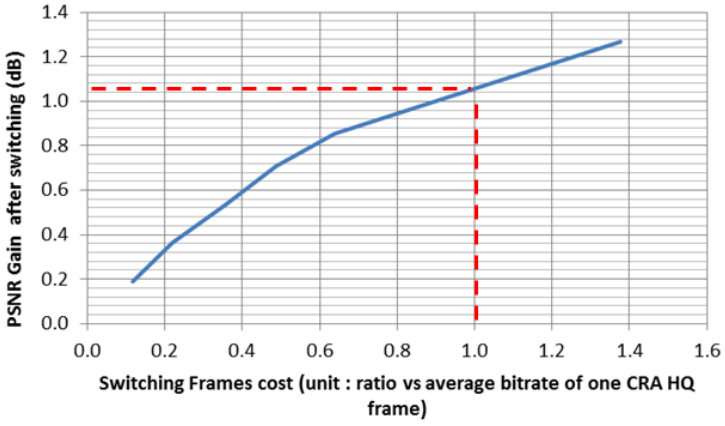


Figure 78: Rate Distortion curve on *ParkScene* sequence. The rate is expressed as the ratio between the actual switching frame cost and encoding one CRA.

This methodology enables a fair measurement of this overhead, especially when the results obtained on several sequences, encoded at different bitrates, are merged. The trend observed for *ParkScene* sequence is confirmed on the other Class-B sequences, as shown on Figure 79.

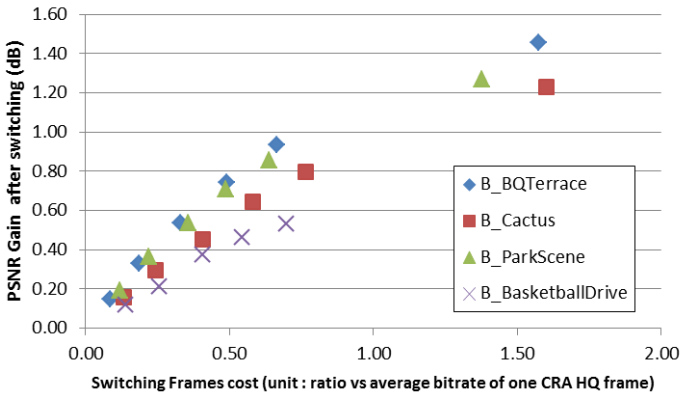


Figure 79: Rate Distortion curve on all sequences.

Conclusion

In this section, a flexible stream switching technique compatible with HEVC and SHVC has been proposed. The flexible feature allows switching at any point of the stream and not only at RAP picture, by sending additional information (switching frames) coded using subset of SHVC modes at switching point. These switching frames can be pre-encoded and stored at the server side with the different stream instances. One can reduce the amount of CRA pictures without lowering the stream switching capacity. It results in less costly streams in terms of bit rate for equivalent reconstructed video quality.

At the receiver side, the DPB frames of the base layer are updated by the switching frames and the quality of the reconstructed pictures after switching is improved.

If the receiver is a regular HEVC decoder, it will discard the switching frames and will behave as in basic switching mode.

8. Colour Gamut and Bit-Depth Scalability

8.1 Basics and stakes of Colour Gamut Scalability

First works related to CGS in the scalable extension of HEVC (SHVC) were presented in JCT-VC in October 2012 [7]. From then, an Ad Hoc Group and a set of Core Experiments have been established to assess different technologies.

The main challenge of CGS is to define a pertinent inter-layer Colour Difference Predictor to predict the enhancement layer color samples from the collocated reconstructed base layer color samples (Figure 80). If the two layers have different spatial resolutions, up-sampling process is also needed.

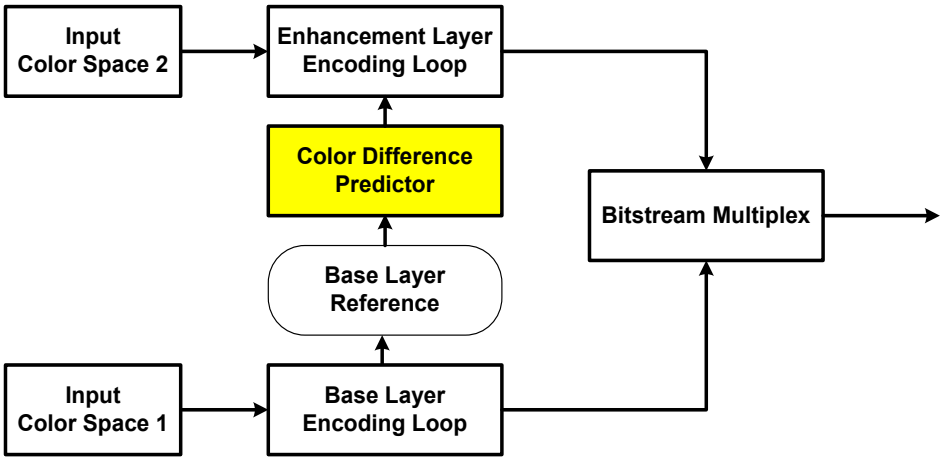


Figure 80: Colour Gamut Scalability principle.

Considering a base layer and a corresponding enhancement layer video sequence, the determination of this colour difference predictor is not trivial because the content creation and production stages may include deterministic processing (colour spaces conversion) as well as non-deterministic operations, for the following reasons [20]:

- **Increasing video capture capability**: The latest digital cameras used in Digital Cinema (DC) and broadcast can capture video signal with wide colour and high dynamic range. The output video data are Wide Colour Gamut (WCG) that may be beyond DCI-P3-gamut, with potentially extended dynamic range (more than 10 f-stops). Captured data represent much more information than what distribution network is able to convey and consumer screens to display.
- **Grading**: the tone and colour balancing is of key importance in content creation, because it instills the artistic intent of the film director and has major impact on the final rendering. It is performed by highly-skilled graphic operators (colorists) using a reference display. Then, if two targeted displays with different characteristics are used (e.g. DCI projectors for DC and Rec. 709 TV for HDTV), the creative intent may be different though consistent with the possibilities of the colour palette provided by the respective reference displays. Thus the colour grading may be different. These graphic designers use special colour authoring and grading tools to globally or partially represent/output their colour processing operations.
- **Colour Space conversion**: Rec.709 to Rec.2020 colour space conversion has been identified as a pragmatic colour space conversion use case (e.g. if one considers a scalable version of HDTV and UHDTV). This conversion is not linear (see 1.2).

An hypothetical scalable HD/UHD production and distribution workflow for movies is depicted in Figure 81, where the original RGB raw material is finally transformed into YCbCr 4:2:0 and compressed. A

more precise definition of a digital motion picture workflow is for instance proposed by the Academy Colour Encoding System (ACES). It may be used to create content for theaters and next for consumer devices such as DVD or Blu-ray Disc players. It is remarked that traditional broadcast may use a different workflow with (semi-)automatic stages for production of live events. However, conceptual steps are equivalent to the ones in the depicted simplified diagram.

Due to the non-deterministic operations and expression of artistic intent during content creation, it is difficult to accurately express the "Colour Difference Predictor" function model as a simple linear model with only a few parameters. Instead, it should be described as a more generic and flexible model.

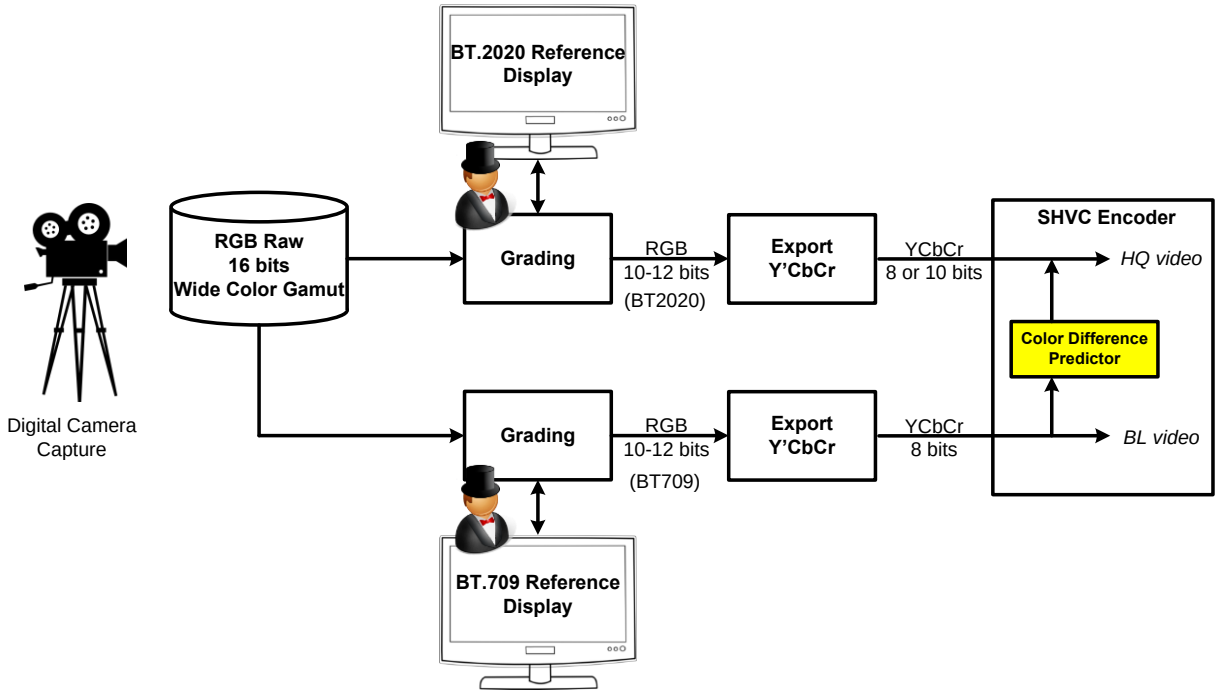


Figure 81: Hypothetical scalable HD/UHD processing workflow, inspired from (simplified) Digital Cinema workflow.

8.2 Experimental CGS content creation

In order to evaluate the performance of various CGS models, we generated three types of content. In the following sections we present how they were created and discuss the respective merit of each approach.

The two first sets of content were created from the regular Rec.709 sequences used by JCT-VC for SHVC video coding experiments, while the third one was specially shot, colour graded and post-processed for this study.

8.2.1 Colour Space conversion Rec.709 to Rec.2020

These sequences were built from the Rec.709 class-B (1920x1080p) test sequences used by JCT-VC. The sequences were converted to Rec.2020 following the common recommended practices (SMPTE RP 177 [10]), as depicted in Figure 82.

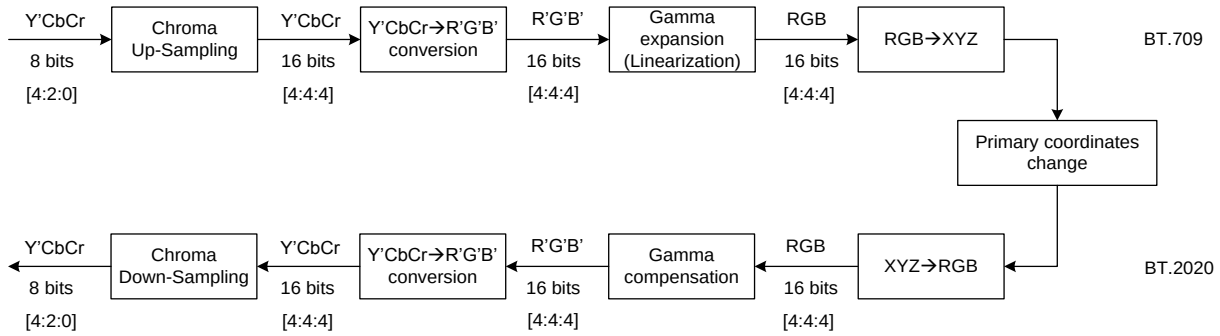


Figure 82: Y'CbCr Rec.709 to Rec.2020 conversion workflow.

As the Rec.709 colours represent a small portion of the Rec.2020 colours (Rec.2020 gamut is twice bigger than Rec.709 gamut) [140], the pictures with colour re-encoded in Rec.2020 look colourless compared to the original when displayed on a Rec.709 screen (Figure 83).

Although this allows quickly initiating CGS studies with basic cases and evaluate the achievable performance with one type of forward compatibility use case, Rec.2020 test material built this way using a simple colour space conversion from Rec.709 content does not correspond to a mainstream realistic use case. Indeed, a Rec.2020 compliant device would likely support the former Rec.709 format, and the service provider would likely prefer to distribute a video content in Rec.709 for compatibility with legacy HDTV devices.

Figure 83: The original BasketBallDrive BT.709 sequence (left) and the colour converted version in BT.2020 (right).

8.2.2 Colour grading with pre-defined looks

The SHVC tools aimed at encoding efficiently any type of video sequences, independently of the colour space used, including existing and future colour spaces. To create such content, the frames were graded with a simple tool to simulate significant disparity between two colour grades made in different colour spaces, but not necessarily Rec.709 and Rec.2020.

CineStyle Colour Assist [11] is a consumer application to correct and colour grade video content; it is a simpler alternative to well-known professional grading tool suites. We used one particular feature of this application to provide a set of pre-defined looks designed by colourists to emulate certain styles or artistic effects.

Two visual effects presets named "*Film emulation*" and "*Morning*" were selected and applied on the JCT-VC test sequence *ParkScene* as depicted in Figure 85, to be compared with the corresponding original Rec.709 and converted Rec.2020 frames in Figure 84.

Figure 84: Original rec709 (left) and converted rec2020 (right) ParkScene (class B) sequence.

Figure 85: Colour graded with CineStyle, using *FilmEmulation* (left) and *Morning* (right) "looks".

8.2.3 Wide Colour Gamut test sequences

The last set of CGS test sequences was specially created for the motion picture/movie use case. Figure 86 depicts a simplified workflow inspired by an actual video content production workflow, with some steps unnecessary for the considered usage discarded, such as editing, conformance checking, dubbing, FX insertion, and so on.

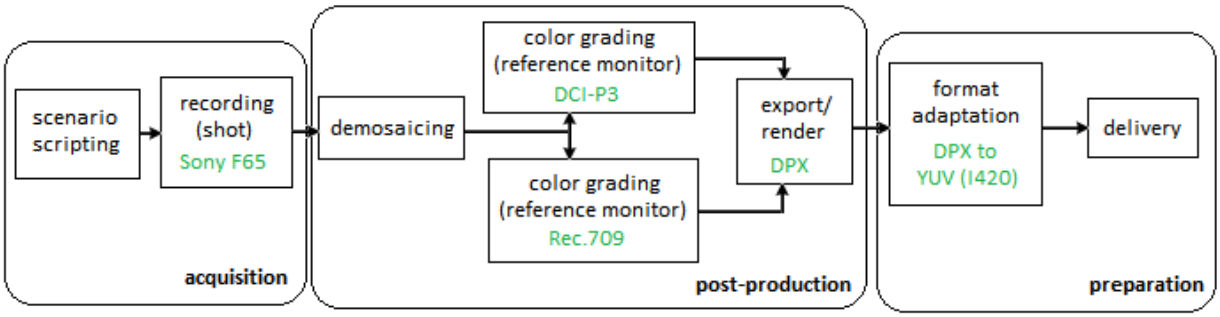


Figure 86: Simplified video content production workflow.

Main steps consisted of shooting, grading and exporting in a suitable distribution format. The goal was to capture natural looking scenes with wide colour gamut. It is widely agreed that most of the natural content colours (represented by the surface within dashed line, a.k.a Pointer's gamut, in Figure 87), are in general correctly rendered within Rec.709 colour space (brown line). However, even under normal illumination conditions, some saturated objects like stained-glass window, parrots, flashy clothes, flash lights, laser, butterflies, may beget a colour spectrum out of the Rec.709 gamut.

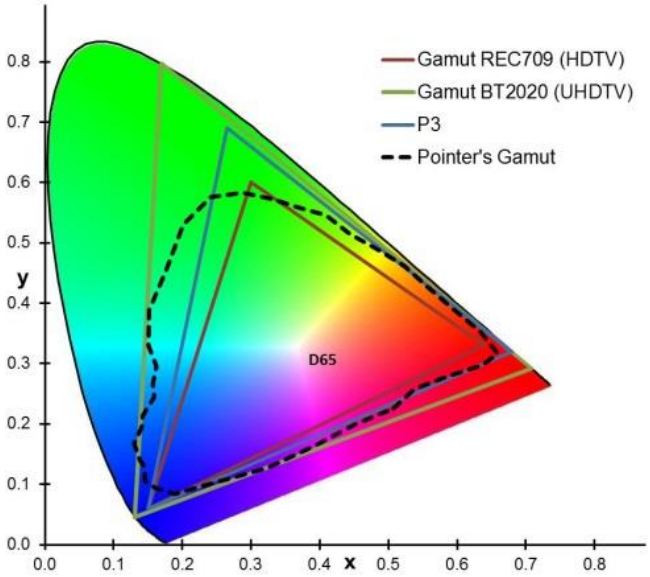


Figure 87: The Pointer's gamut contour compared to different gamuts in xy chromaticity diagram.

Based on these considerations, it was decided to shoot content with such saturated colours. Parakeets, saturated birthday clothes, as well as flashy party items under RGB flashlights were identified as good candidates for being graded both with Rec.709 and DCI-P3 [12] reference monitors or projectors. A 4K digital camera Sony F65 [13] was used for the shooting sessions with settings that maximize quality (4K resolution, no gamma encoding i.e. linear-light domain, S-gamut). A Director of Photography was assigned to the project in order to ensure high quality content shooting.

Digital cinema employs DCI-P3 colour space (emulation of print film colour gamut) that has a wider gamut than Rec.709 (aligned on CRT TV sets primaries). Reference white point was chosen to be the same as in Rec.709, i.e. D65. Colourist took care of giving a natural looking style to graded content as well as creating out-of-gamut Rec.709 colours for the P3 graded contents (Figure 88).

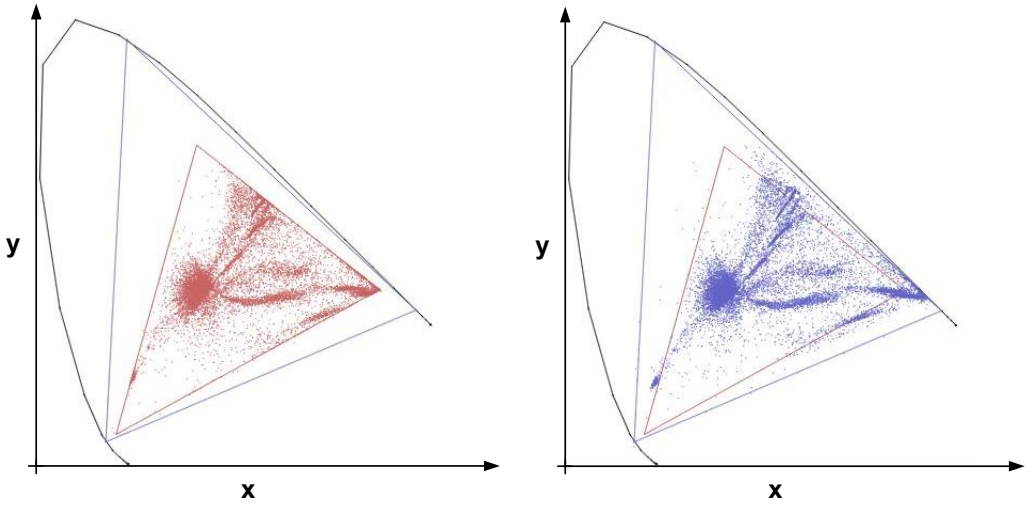


Figure 88: The "Birthday party" sequence: Rec. 709 grade (left) and P3 grade (right) projection in xy chromaticity diagram. Brown triangle represents 709 and blue triangle represent 2020.

At last, the DPX R'G'B' was converted to equivalent Y'CbCr 4:2:0 8-bits or 10-bits using the SMPTE RP 177 guidelines [10] for deriving the P3 R'G'B' to Y'CbCr conversion matrix coefficients. Besides the original 3840x2160 resolution, several down-sampled resolutions were created, including 1920x1080,

1280x720 and 960x540. Figure 89 shows the set of content that was provided to the JCT-VC as test material for the CGS studies [18] (see also Annex 10.1).

Figure 89: Overview of the WCG test sequences (Birthday, BirthdayFlash, TableCar, Parakeets).

8.3 Study of CGS models for SHVC

8.3.1 Linear models

The first technical proposal for addressing CGS in JCT-VC was made in 2012 by L.Kerofsky et al. from Sharp Laboratories [7]. Anticipating the upcoming deployment of UHDTV services with the recently released Rec.2020 recommendation [5], this proposal analyzes the colour space conversion steps as presented in 8.2.1 (Figure 82). Although the overall processing is not linear, they propose to approximate the colour space conversion YUV_{rec709} (8-bits) to $YUV_{rec2020}$ (10-bits) using linear models. Two models are compared:

Gain-Offset model (GO): all picture samples in Rec.2020 are predicted from the corresponding sample in Rec.709 using a linear model (46), where the gain and offset are allowed being different per component but are constant for the entire frame:

$$\begin{pmatrix} Y \\ Cb \\ Cr \end{pmatrix}_{rec.2020} = \begin{pmatrix} g_1 & 0 & 0 \\ 0 & g_2 & 0 \\ 0 & 0 & g_3 \end{pmatrix} . \begin{pmatrix} Y \\ Cb \\ Cr \end{pmatrix}_{rec.709} + \begin{pmatrix} o_1 \\ o_2 \\ o_3 \end{pmatrix} \quad (46)$$

Cross-Component model (CC): each component in Rec.2020 is predicted from the co-located triplet in Rec.709 using a matrix and offset model in Equation (47). These parameters are constant for the entire frame. For the case of chroma sub-sampling (ex: 4:2:0), the corresponding samples between luma and chroma must be identified. To simplify the process, for luma prediction, the same chroma values correspond to four luma samples. While for chroma prediction, the upper left luma sample of a 2x2 group is used to form the predictor.

$$\begin{pmatrix} Y \\ Cb \\ Cr \end{pmatrix}_{rec.2020} = \begin{pmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{pmatrix} . \begin{pmatrix} Y \\ Cb \\ Cr \end{pmatrix}_{rec.709} + \begin{pmatrix} o_1 \\ o_2 \\ o_3 \end{pmatrix} \quad (47)$$

Note the GO model is a particular case of the CC model, where the values of m_{ij} are zero if i is not equal to j. The determination of the matrix m_{ij} and offset o_i coefficients can be done using least square minimization (LSM) method, for a given pair of pictures, or several pictures of a sequence, as depicted in (48) for the Y component.

$$\{m_{1,i}, o_1\} = \arg min \left[\left(Y_{rec.2020} - \left(m_{1,1} Y_{rec.709} + m_{1,2} Cb_{rec.709} + m_{1,3} Cr_{rec.709} \right) \right)^2 \right] \quad (48)$$

To figure out the performance of the two linear models discussed above, the mean square error (MSE) between the samples in Rec.2020 and the predictor obtained by applying the prediction model on the collocated samples in Rec. 709 is summarized in Figure 90 from [7]. The pair of images considered is the JCT-VC test sequence *BasketBallDrive* (1080p) in Rec.709-8bits and its Rec.2020-10bits version obtained via colour space conversion from the Rec.709 (cf. 8.2.1). As seen in the Figure, compared to the GO model, the CC model achieves significantly lower MSE.

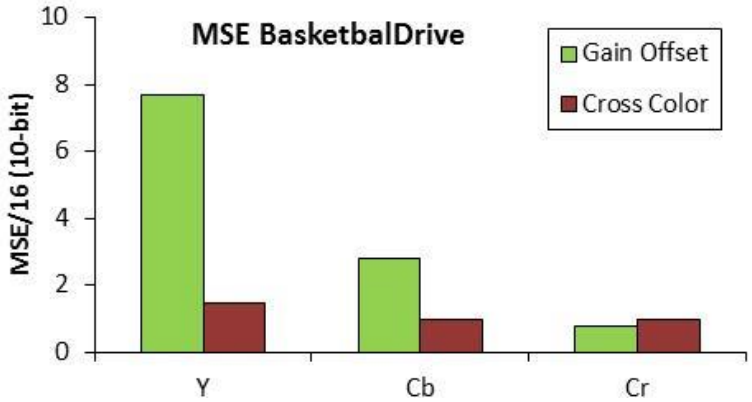


Figure 90: Comparison of Gain-Offset with Cross-Component models for predicting rec.2020 sample values from rec.709 sample values.

To evaluate the performance of the CC method in SHVC, we implemented it in the SHVC reference software (SHM1.0) [24]. The matrix coefficients and the offset values were encoded in the bit-stream in order for the decoder to share the same model as the encoder, and to generate the prediction for EL colour samples from reconstructed BL samples. In our experiments, these model parameters were computed per sequence, using LSM over a subset of pictures (one picture every second), and encoded in integer 8-bits signed precision with exp. Golomb in the PPS header [21]. The main advantage of the CC model is its simplicity, both in terms of bit stream overhead and in terms of low complexity at the decoder side for CGS prediction computation.

Rec.2020 sequences created from the Rec.709 test material presented in section 8.2.1 (colour space conversion) were encoded with the SHVC common test conditions [22]: All-Intra (AI) and Random-Access (RA), SNR and spatial (1.5 ratio) scalability. The Y,Cb,Cr EL and (BL+EL) BD-rate gains [23] using the CC model compared to SHM1.0 without CGS, as well as the average decoding time percentage increase (DT) are provided in Table 23. A negative value in the BD rate represents the percentage of rate reduction at the same PSNR.

Table 23: CC model BD-rate gains (%) and Decoding Time increase (DT)(%) vs SHM1.0 (colour converted content).

BL 1080p	BD-rate EL			BD-rate BL+EL			DT
	Y	Cb	Cr	Y	Cb	Cr	
AI_SNR	-7.0	-3.0	-17.5	-3.8	0.3	-14.1	10
AI_1.5x	-7.3	-4.8	-30.9	-3.2	-1.1	-23.4	5
RA_SNR	-5.8	1.7	-25.0	-3.2	3.8	-21.6	17
RA_1.5x	-7.7	-4.9	-39.0	-3.3	-0.9	-33.1	8

8.3.2 CLUT model

The CC model has its limitations because it is linear in the entire 3D colour space, while the prediction error distribution can be quite complex. In order to improve prediction accuracy for a wider range of Colour Gamut Scalability applications (not only simple colour space conversion), one may consider a more generic model to represent the colour difference of content. For simplicity of notations, (Y, Cb, Cr) and (Y, U, V) will be used interchangeably for the remainder of this section.

Thus, a new CGS model inspired from those used in colour grading tool suites was introduced. When post-processing and grading a movie, the colourists commonly use Colour Look-Up Table (CLUT) representation to store and operate the colour transforms required to generate different versions of the material.

The principle of the CLUT is depicted in Figure 91. Essentially, it can be considered as a partitioning of an original 3D colour space, where each vertex of the colour space cube is associated with a colour triplet corresponding to the (predicted or transformed) values in the target colour space. In that way, the original colour space is spatially divided into a set of cubes called octants because delimited with eight vertices.

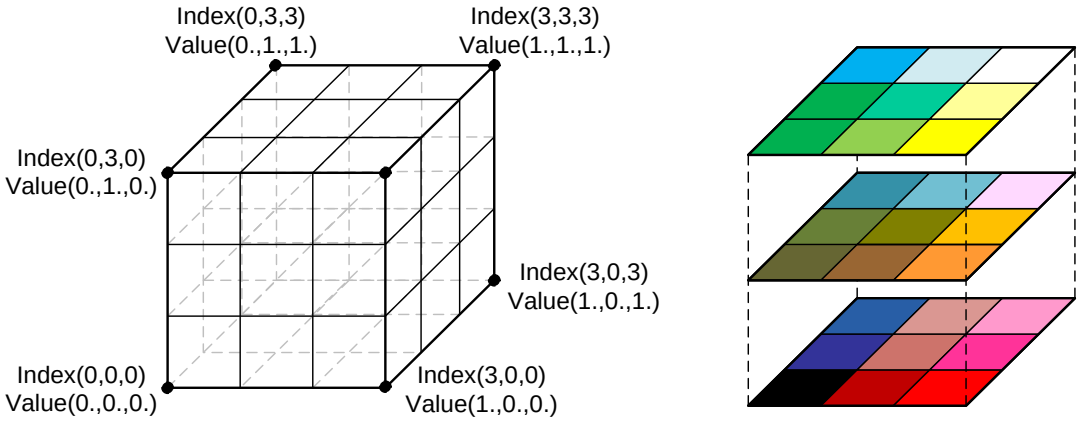


Figure 91: Example of RGB (identity) 3D colour LUT.

When CLUT is applied to CGS, for a given BL colour sample in BL colour space, the computation of its prediction in EL colour space is made using interpolation in the 3D mesh grid. A straightforward interpolation is the tri-linear interpolation. We will discuss the relative merit of other interpolation algorithms such as tetrahedral interpolation in section 8.3.2.1.

The principle of the tri-linear interpolation is depicted in Figure 92 and equation (49). For a given colour triplet (y,u,v) in the original colour space, we first determine the corresponding octant and vertices $\{(y_i,u_j,v_k)_{i=0,1,\ j=0,1,\ k=0,1}\}$. Denote the triplets in the target colour space associated with these vertices as $\{(LUT_Y[y_i][u_j][v_k], LUT_Y[y_i][u_j][v_k], LUT_Y[y_i][u_j][v_k])_{\ i=0,1,\ j=0,1,\ k=0,1}\}$. These are depicted in Figure 92 as $LUT[y_i][u_j][v_k]$ without specific reference to the colour component. Then, the interpolated value in target colour space is given in Equation (49) for the component *Y*.

$$Y = \frac{\sum_{i=0,1}\sum_{j=0,1}\sum_{k=0,1} s_i(y).s_j(u).s_k(v).LUT_Y[y_i][u_j][v_k]}{(y_1-y_0).(u_1-u_0).(v_1-v_0)} \quad (49)$$

where: $s_0(y) = y_1-y$ and $s_1(y) = y-y_0$,
$s_0(u) = u_1-u$ and $s_1(u) = u-u_0$,
$s_0(v) = v_1-v$ and $s_1(v) = v-v_0$,
y_0 (resp. u_0,v_0) is lowest end value of the Y (resp. U,V) axis in the octant,
y_1 (resp. u_1,v_1) is highest end value of the Y (resp. U,V) axis in the octant.

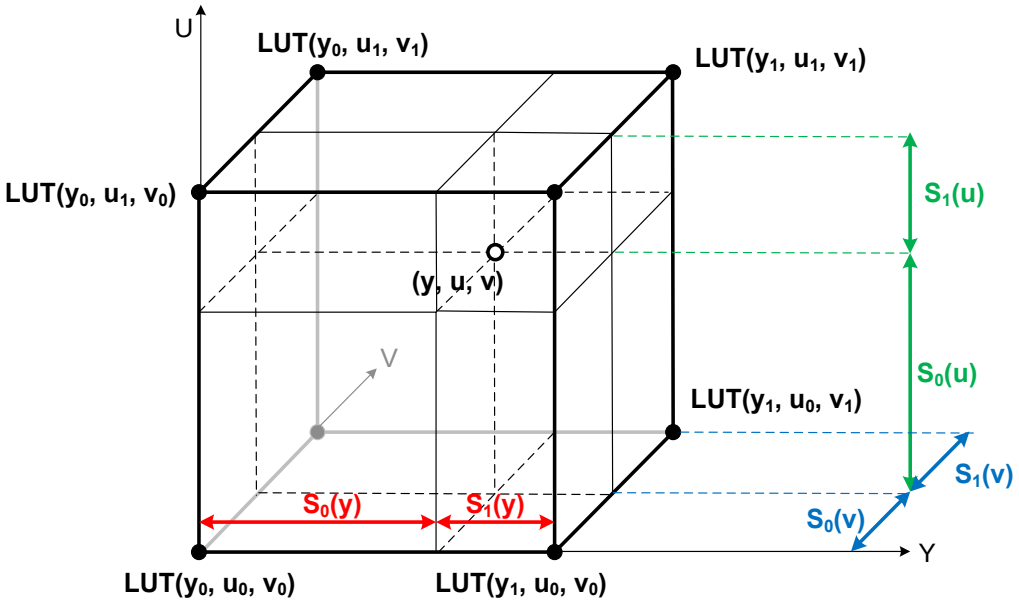


Figure 92: Tri-linear interpolation with 3D LUT.

The number of vertices in one direction of the CLUT (*nbp*) is a key parameter: a larger value means more precise colour transform estimation and good prediction consequently. But the amount of data (CLUT vertices triplets) to be encoded increases at the order of O(nbp^3). Typically, the colour grading tools commonly use CLUT size 17 or 33. If *bitdepth* denotes the number of bits used to represent one colour component (e.g. 8-bit) in the target colour space, the number of CLUT bits for encoding is nbp^3 *x 3 x bitdepth*. For 8-bit EL video, the number of CLUT bits to be encoded will be 117912 or 862488 for *nbp* being equal to 17 or 33, respectively. Even if the CLUT is encoded at PPS level (i.e. about every second for broadcast streams) and not at slice level for each frame, the overhead is not acceptable and would obliterate the CLUT benefits.

A deeper analysis of the CLUT data have shown that not all the colour space is covered by the colour samples in the actual video content being encoded.

To take advantage of signal sparsity and the correlation of neighboring vertex values, we proposed an original octree based prediction coding of the CLUT in [25] (Figure 93). The colour components of a given vertex are predicted by interpolating the reconstructed vertex components of the upper level octant, with the exception of the root octant. The residuals are quantized and coded with exponential Golomb entropy coding.

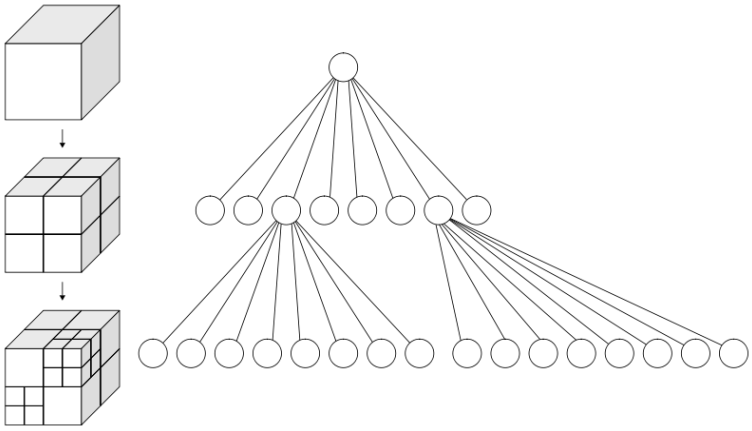


Figure 93: Octree based 3D LUT: each octant is coded with 8 vertices at most.

As depicted in Table 24, octree based CLUT coding method saves significant number of bits compared to uncompressed CLUT, especially for large CLUT size. The value of *nbp* and the quantization step applied to the residuals (*res_quant*) are encoded in the bit-stream, so that the encoder may choose the best trade-off between colour prediction accuracy and CLUT encoding cost (rate-distortion optimization).

Table 24: nb bytes used to encode CLUT with octree coding compared to uncompress CLUT for the sequences of 8.2.3 (Wide Colour Gamut).

Bit-depth=8	**Number of bytes used to encode the CLUT**		
nbp	**5**	**9**	**17**
Uncompress CLUT	375	2187	14739
Octree coding (res_quant=1)			
Parakeets	174	482	1339
TableCar	240	733	2511
Birthday	246	722	2257
BirthdayFlash	263	627	2080

Experimental results

Compared to the CC model, which uses the same model parameters for the entire colour space, the CLUT model provides better prediction accuracy by partitioning the colour space into octants and applying different model parameters in each octant. The performance of CLUT model (*nbp* = 9) is compared to that of CC model using the CineStyle content in section 8.2.2. The EL BD-rate gains for the three components and decoding time are given in Table 25. As shown, significantly higher gains are achieved using the CLUT model [20].

Table 25: CC and CLUT models: EL BD-rate gains (%) and Decoding Time increase (DT)(%) vs SHM1.0 .

BL 1080p	**CC model**				**CLUT model**			
	Y	**Cb**	**Cr**	**DT**	**Y**	**Cb**	**Cr**	**DT**
AI_SNR	-9.2	-10.2	-10.9	7	-24.3	-29.2	-28.0	18
AI_15x	-14.6	-15.6	-15.3	2	-30.1	-33.9	-32.4	3
RA_SNR	-5.6	-6.4	-6.8	10	-15.4	-22.0	-18.0	39
RA_15x	-10.4	-10.4	-10.0	7	-22.1	-26.3	-23.0	15

8.3.2.1 CLUT Tetrahedral versus tri-linear interpolation

The tri-linear interpolation is based on computing relative cuboid volumes. If the octant size in each direction is power of 2, the division in equation (49) in 8.3.2 can be implemented via shifts, but the weights may need large registers: up to 24 bits for 8-bits colour sample and even more for higher bit-depth.

Several alternative interpolation functions have been proposed to cope with this implementation complexity issue. The most used in the industry (e.g. colour grading tools, GPGPUs based systems) is the tetrahedral interpolation originally proposed by Matsushita research institute in Japan in 1990 [28]. In our implementations, insights given by [29] were used. With this method, interpolation is limited within the four points of a tetrahedron inside the related octant instead of the whole octant.

There are 6 possible tetrahedron (Figure 94). The choice of the tetrahedron depends on the position (*dy,du,dv*) of the point *P* to be interpolated inside the octant, as depicted in Figure 95 and Table 26. Then, the interpolated value is given by Equation (50):

$$P = P_0 + C_1 \times \frac{dy}{y_1 - y_0} + C_2 \times \frac{du}{u_1 - u_0} + C_3 \times \frac{dv}{v_1 - v_0} \qquad (50)$$

where (C_1,C_2,C_3) are derived from (*dy,du,dv*) in Table 26.

In Equation (50), *P, P₀, C₁, C₂* and *C₃* are vectors with 3 components. If the octant size is a power of 2, the divisions can be replaced with right shifts and the required register size of $C_1 \times dy$ is only 16 bits for *bit-depth* equal to 8-bit.

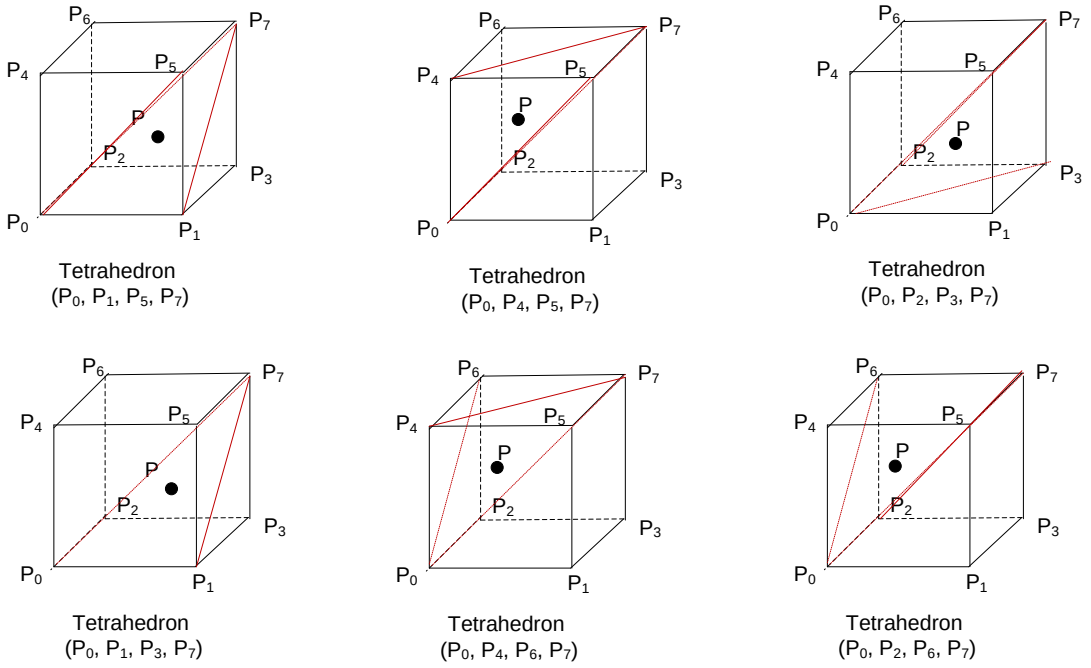


Figure 94: The 6 possible tetrahedrons, depending on the position of P in the sub-cube.

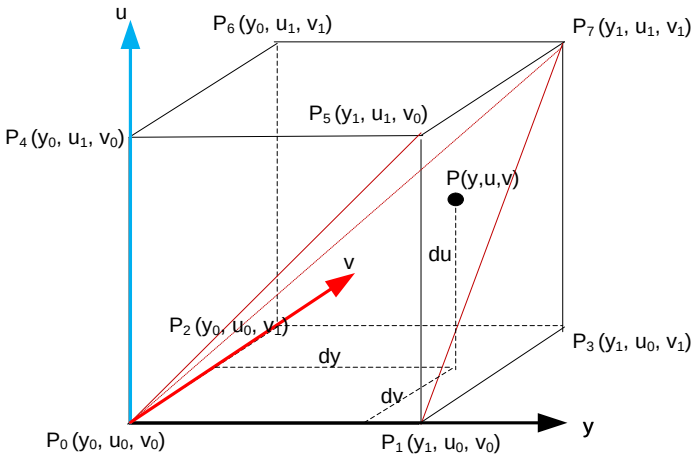


Figure 95: tetrahedral interpolation.

Table 26: determination of the tetrahedron depending on dy, du, dv.

Tetrahedron	Comparison	C1	C2	C3
T1	dy ≥ du ≥ dv	P1 – P0	P3 – P1	P7 – P3
T2	dy > dv ≥ du	P1 – P0	P7 - P5	P5 – P1
T3	dv ≥ dy > du	P5 – P4	P7 - P5	P4 – P0
T4	du > dy ≥ dv	P3 – P2	P2 – P0	P7 - P3
T5	du > dv > dy	P7 – P6	P2 – P0	P6 - P2
T6	dv ≥ du ≥ dy	P7 – P6	P6 - P4	P6 – P0

To evaluate and compare the tri-linear and tetrahedral interpolation functions, CLUT were generated to perform colour space transform between Rec.2020 to Rec.709 using analytic equations [10].

The PSNR computed between the Rec.2020 frames converted to Rec.709 colour space using the CLUT interpolation and using analytical equations is provided in Table 27. The PSNR results for tetrahedral interpolation is slightly better for *Y'* component, but a bit lower for *Cb* and *Cr* components. With sufficient CLUT size (at least 9x9x9 data entries or equivalently 8x8x8 octants), the PSNR values remain around 50dB, which corresponds to a high level of performance.

Table 27: Tri-linear vs Tetrahedral interpolations comparison.

	Tri-linear interpolation			Tetrahedral interpolation		
	Y	Cb	Cr	Y	Cb	Cr
5x5x5	36.26	38.53	50.61	37.47	36.42	47.48
9x9x9	48.74	44.34	53.09	49.75	42.96	49.70
17x17x17	51.07	50.19	53.99	51.45	47.99	50.21
33x33x33	50.80	51.31	52.76	50.08	49.58	49.61
65x65x65	51.29	50.30	52.59	49.86	48.42	49.99

8.3.2.2 CLUT computation

The accuracy of the CLUT parameters has major impact on the CLUT model based encoding efficiency. But finding the optimal CLUT may not to be easy for several reasons:

- Each vertex is shared by eight neighboring octants (cf. Figure 91), so the minimization method cannot be solved locally but needs to consider the whole set of vertices together, making it necessary to manipulate large matrix size, as will be discussed in sub-section 1) below.
- Computing the CLUT using original BL and EL video sequences is a simple way to proceed, but may not lead to optimal solution since the encoder will use the reconstructed BL for predicting the EL via the CLUT. The encoding artifacts increase with the quantization step value which may introduce poor BL-EL colour sample pairing.
- In case the BL and EL have different picture sizes (spatial scalability), BL and EL picture sizes need to be aligned, for example by down-scaling the EL, which could beget colour samples at object boundaries that do not actually exist.

Ideally, the CLUT can be trained offline on the whole sequence, or for a subset of frames. In that way one can determine when the CLUT should be updated and transmitted, which may occur at scene cut typically. However, for some practical reasons (long sequence duration means that huge amount of data needs to be pre-processed) and to better fit with realistic use cases such as broadcast/broadband real time applications, it is worthwhile to consider the computing and transmitting of the CLUT at Random Access Points (RAP). This approach is compatible with channel changes and splicing operations that may happen at predefined points in the stream. Further, this approach improves the coded stream resilience and robustness against errors/packet losses.

1) CLUT computation using global minimization

In [26] Y.He proposed the following Least Square minimization (LSM) method to compute the CLUT from one reconstructed BL frame and one (down-scaled if necessary) original EL frame. Since the colour interpolation (tri-linear, tetrahedral…) used in CLUT based CGS is a linear function, it can be described as equation (51) in matrix form.

$$z_i(c) = \vec{w}_i(c) * P(c), i \in [0, N-1] \qquad (51)$$

$z_i(c)$ is the i-th colour sample converted signal of component c, where c can be Y, U or V, i is the index of input pixel, N is the total number of input pixels, * in Equation (51) is matrix multiplication, $\vec{w}_i(c)$ is the weighting vector for the i-th input pixel, and is expressed as below:

$$\vec{w}_i(c) = [w_{i,0} \quad \cdots \quad w_{i,M-1}], i \in [0, N-1]$$

$w_{i,j}$ is the weight of the j-th output entry of the CLUT for the i-th input pixel, which is calculated according to CLUT interpolation equations (49) or (50).

$P(c)$ is the parameter vector to be estimated, that is the output entries of 3D LUT, and can be written as below:

$$P(c) = [p_0 \quad \cdots \quad p_{M-1}]$$

M is the number of CLUT output entries, for example M is 729 for 9x9x9 sized CLUT. For simplicity, we will omit the component c in the following equations because the 3D LUT of each component can be estimated independently. Aggregating Equation (51) for all pixels in a matrix form gives:

$$Z = W * P \qquad (52)$$

$$Z = [z_0 \quad \cdots \quad z_{N-1}]^T$$

$$W = \begin{bmatrix} w_{0,0} & w_{0,1} & \cdots & w_{0,M-1} \\ \vdots & \vdots & \ddots & \vdots \\ w_{N-1,0} & w_{N-1,1} & \cdots & w_{N-1,M-1} \end{bmatrix} \quad (53)$$

In practice, with the input signal used (see WCG test sequences presented in 8.2.3), not all CLUT entries can be filled. Therefore, one can obtain a compact parameter set *P'* by removing all those unused CLUT vertices from P, and one can build a mapping from *P'* to *P*.

$$P' = compact(P)$$

Since the matrix W' is NxM' and hence non-square, the computing of the corresponding pseudo-inverse matrix allows finding a solution for P'. For instance, one can use the Moon-Penrose Pseudo-inverse method that will derive a square M'xM' matrix to be inverted.

After the compact parameter vector *P'* is estimated, then the full parameter vector *P* can be obtained by mapping those estimated vertices from *P'* to *P*.

$$P = decompact(P')$$

Those un-estimated vertices in *P* are filled using the corresponding vertices in *P'* and the tri-linear interpolation in CLUT encoding process.

2) CLUT computation using local minimization

To simplify the CLUT derivation process and avoid inverting matrix of very large size, we developed an alternative method in [27]. It is based on local (octant) optimization of the computation of the CLUT vertices, independently to the neighboring vertices. The algorithm is implemented in a hierarchical scheme following the octree representation of the CLUT (Figure 93).

First, the 8 vertices of each smaller hierarchy octants are estimated using LSM method. It can be the same method as depicted in 8.3.1, with the advantage M is equal to 8, then the matrices inversions are very easy. However, the CLUT is sparse in general because some octants may not have enough input colour samples, then the LSM method may fail possibly. In that case, we apply the method at the immediate upper hierarchical level, considering the larger octant which encompass the current one plus seven other neighboring octants. This can be iterated until the LSM method succeeds in finding 8 vertices estimates. Then, the vertices of the lower hierarchy octants are set with the CLUT interpolation applied on the 8 vertices of the higher hierarchy.

At last, 8 colour vertices values have been computed for every octant independently. They are merged into one single colour vertex with weighted average, the weight being proportional to the number of input colour samples in the octant.

8.3.2.3 Experimental results

We present the results obtained with SHM4.0 and CLUT tetrahedral interpolation (9x9x9, 8-bits precision), computing the CLUT at IRAP period (every second) with the first I frame of the RAP period, using CLUT overall minimization and CLUT local minimization algorithms respectively.

For the experiments, we used the WCG content presented in 8.2.3 and adopted by JCT-VC for the CGS studies. Two use cases corresponding to 2x spatial scalability are considered, using 8-bits or 10-bits base layer (BL), and 10-bits enhancement layer (EL). Two configurations All-Intra (AI) and Random Access (RA) were used. Quantization steps and other parameters are the same as regular JCT-VC SHVC common test conditions. An existing coding tool, the Weighted Prediction (WP) (see also section 5.2), is enabled in the reference to improve Inter-Layer Prediction (ILP) efficiency of the reference. WP enables

illumination variation compensation between the current picture and the reference pictures in the DPB. WP is very similar to the GO model discussed above, and can also be used to improve CGS coding efficiency.

Table 28 compares the CGS coding performance of the CLUT global and local minimization algorithms [26][27]. The CLUT is computed and signaled every second; that is, for RA configuration, the CLUT is signaled on the first I frame of each IRAP period. Both the BD-rate gains for EL only and the overall BD-rate gains for BL+EL streams are included. For the latter, since the BL is unchanged, the percentage of gain is lower than that of considering EL only.

Table 28: Performance of CLUT models compared to SHM-4.0 with WP enabled for ILP (ref.) (AI and RA configurations).

	AI HEVC 2x 10-bit base			AI HEVC 2x 8-bit base		
	Y	**Cb**	**Cr**	**Y**	**Cb**	**Cr**
CLUT global vs ref						
EL only	-15.2%	-15.9%	-18.6%	-15.2%	-15.9%	-18.4%
BL+EL	-7.8%	-8.9%	-11.6%	-7.8%	-8.9%	-11.5%
CLUT local vs ref						
EL only	-11.5%	-13.3%	-15.4%	-12.0%	-13.7%	-15.7%
BL+EL	-5.6%	-7.8%	-9.9%	-5.9%	-8.0%	-10.0%

	RA HEVC 2x 10-bit base			RA HEVC 2x 8-bit base		
	Y	**Cb**	**Cr**	**Y**	**Cb**	**Cr**
CLUT global vs ref						
EL only	-11.7%	-11.4%	-16.0%	-11.5%	-11.4%	-15.4%
BL+EL	-6.3%	-6.5%	-11.0%	-6.2%	-6.5%	-10.6%
CLUT local vs ref						
EL only	-9.6%	-10.2%	-13.9%	-9.7%	-10.2%	-13.5%
BL+EL	-5.1%	-6.1%	-9.7%	-5.1%	-6.1%	-9.3%

Table 29 provides the gain using scalable coding (two layered BL and EL SHVC streams) compared to using two separate single layer HEVC streams (simulcast). Three scalable coding methods are used: with CGS tool using CLUT-global minimization or CLUT-local minimization, and without CGS tool but with WP tool (reference). As expected, the global optimization algorithm clearly outperforms the simplified local optimization method, and both CLUT methods significantly outperform the reference that uses only WP but no dedicated CGS tool. Higher gains are obtained with AI configurations.

Table 29: Performance of CLUT models compared to simulcast.

	AI HEVC 2x 10-bit base			AI HEVC 2x 8-bit base		
	Y	**Cb**	**Cr**	**Y**	**Cb**	**Cr**
CLUT-Global	-28.3%	-26.5%	-31.9%	-26.6%	-25.4%	-31.1%
CLUT-Local	-26.3%	-25.7%	-30.4%	-24.9%	-24.7%	-29.9%
Reference	-22.4%	-19.5%	-22.8%	-20.7%	-18.4%	-22.0%

	RA HEVC 2x 10-bit base			RA HEVC 2x 8-bit base		
	Y	**Cb**	**Cr**	**Y**	**Cb**	**Cr**
CLUT-Global	-20.5%	-15.7%	-24.6%	-19.4%	-15.0%	-23.8%
CLUT-Local	-19.3%	-15.5%	-23.4%	-18.4%	-14.8%	-22.6%
Reference	-15.2%	-9.9%	-14.3%	-14.3%	-9.3%	-14.0%

Table 30 provides the encoding/decoding time gain of CLUT models compared to the reference without CGS tools (but with WP). For both CLUT estimation methods, the encoding time is significantly reduced, in particular for RA cases because the improved Inter-Layer Prediction is often good enough, making it unnecessary to test the other intra-layer prediction modes in the rate distortion optimization process. Also, the improved prediction signal reduces the burden of residual coding, thus alleviating the computing load at the encoder and the decoder.

Table 30: Encoding/Decoding time gain (%) of CLUT models compared to reference without CGS tools.

		AI HEVC 2x		RA HEVC 2x	
	Time %	**10b base**	**8b base**	**10b base**	**8b base**
CLUT-Global	Encoder	90.9%	93.0%	64.0%	65.1%
	Decoder	98.3%	98.6%	106.1%	106.3%
CLUT-Local	Encoder	96.3%	96.7%	67.4%	67.0%
	Decoder	96.2%	97.8%	104.7%	107.7%

8.3.3 Complexity analysis

We have shown in previous sections the CLUT tool provides significant BD-rate improvement for CGS coding of Wide Colour Gamut content. However, before adopting such a tool in SHVC standard, one has to prove it does not introduce unacceptable amount of additional complexity to the future decoder devices. For this purpose, we compare the CLUT based prediction with another SHVC tool that behaves similarly: the Weighted Prediction (WP). WP allows compensating illumination variation between current frame and reference frames in the DPB (see chapter 5.2).

In case of spatial scalability, one major difference between CLUT and WP is that the colour conversion applies before BL up-sampling process (that is, in BL resolution) for CLUT (Figure 96), but after up-sampling process (that is, in EL resolution) for WP, as depicted in Figure 97 for the 2x scalability case.

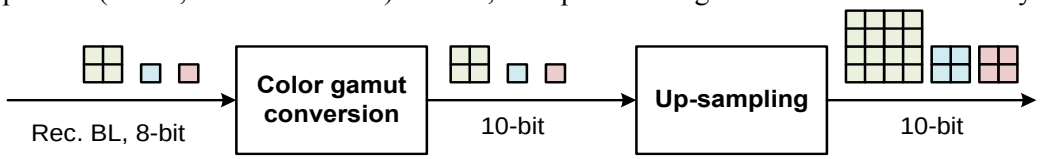


Figure 96: workflow process for building ILR (inter-layer reference) prediction with CLUT.

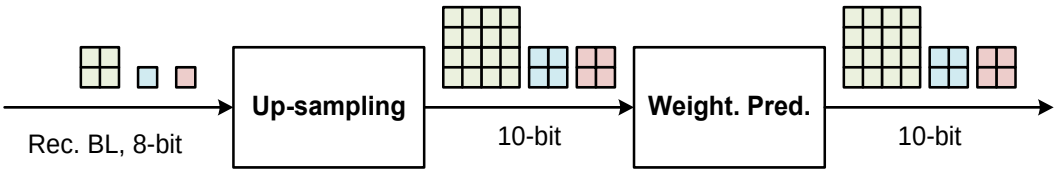


Figure 97: workflow process for building ILR (inter-layer reference) prediction with Weighted Prediction (WP).

Besides, the intra-layer temporal prediction uses Motion Compensation, which is another computationally demanding sample based processing (Figure 98). Then, the complexity evaluation of the CLUT process has to be compared with that of up-sampling (US), motion compensation (MC) and weighted-prediction (WP). For up-sampling and motion compensation, the SHVC luma and chroma filters use 8-taps and 4-taps respectively. The filters are applied first horizontally then vertically, and it requires intermediate (temporary) storage of the first pass (horizontal) filtered samples. While for weighted-prediction, one multiplication only is performed in uni-prediction.

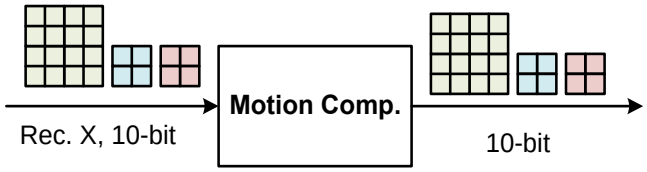


Figure 98: intra-layer temporal prediction (motion compensation) workflow process.

Considering a 4x4 block in the BL, the corresponding number of operations pixel-wise of these 4 tools is described in [27] and shown in Table 31 for the 2x scalability case. The calculation in Table 31 shows that the CLUT complexity (number of operations) is on the same order as WP, sometimes even less. Compared to up-sampling and motion compensation, its computational load is also relatively low.

Table 31: comparison of the number of operations per pixel for an enhancement layer 4x4 block of different processing steps. Ratio of CLUT colour conversion vs weighted prediction is also shown.

nb total operations (2x scalability)					
	CLUT	**US**	**MC**	**WP**	**CLUT vs WP**
mult	18	*240*	*320*	24	**-25%**
add	52	*228*	*320*	72	**-28%**
shift	26	*24*	*24*	24	**+8%**

8.3.4 CLUT improvements

Given performance of the CLUT model to predict the EL pictures and its relatively low processing overhead, colour mapping with CLUT for inter-layer prediction has been adopted by JCT-VC in the Scalable Main-10 profile of SHVC to provide efficient CGS coding. However, some improvements have been brought to the initial design to further improve design complexity and coding efficiency trade-off.

Asymmetric CLUT and non-uniform chroma partitioning

In Figure 93, the CLUT coding uses an octree based structure, splitting the octants to appropriate sizes and grouping colour space areas where the colour transfer function has similar characteristics. Because of the symmetric nature of octree, the grouping operates in the same way in all the dimensions of the CLUT. However, the BL and EL video signal may have colour disparities that are unequally distributed on the three Y-Cb-Cr components. This may be particularly true for the Luma component that typically carries the most of the texture information.

In [38], X. Li proposed a variant extending the CLUT tree to support asymmetric CLUT partitioning. Each leaves of the tree may be further split in *yPartNum* partitions (1 to 8) in the Y direction as depicted in Figure 99 (left).

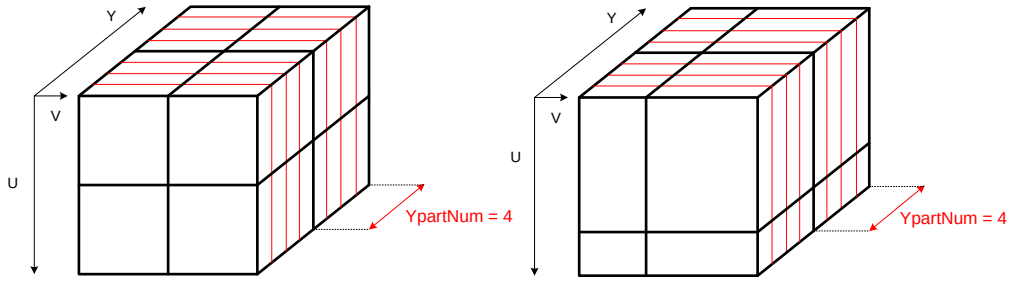


Figure 99: Asymmetric CLUT partitioning (left) + non-uniform chroma partitioning (right), with *yPartNum*=4.

Besides, the tetrahedral CLUT interpolation (8.3.2.1) can be simplified to force the usage of the T1 tetrahedron only in Table 26, removing the need to test the conditions to select the tetrahedron. Additionally, these four vertices of the CLUT are no longer shared with the neighboring octants as in the

previous CLUT design discussed in 8.3.2.2. Thus, the CLUT tree leaves are composed of four vertices, each vertex being a trichromatic triplet. In that way, the derivation of the CLUT is highly simplified compared to subsection 8.3.2.2, because the four vertices are determined from the values belonging to a single octant only, independently of neighboring octants.

The tetrahedral interpolation using four vertices composed of three colour components (4x3=12 parameters) is a linear model equivalent to using CC like model (cf. 8.3.1) locally (3x3 + 3 = 12 parameters) [39]. Therefore, the tetrahedral interpolation is replaced by CC model for a straightforward representation. With this representation in mind and to further increase the flexibility and CGS coding efficiency of the asymmetric CLUT, C. Auyeung [39] proposed to allow non-uniform partition of the colour space in the chroma directions, as depicted in Figure 99 (right), in case the CLUT tree depth is equal to one in the chroma directions.

Experimental results

Table 32 summarizes the experimental results obtained with the CLUT-based colour mapping tool in SHVC (with the CLUT design presented above) for AI and RA configurations, respectively. As above, WP is turned on in the reference. The maximum CLUT size is 8×2×2 and CLUT parameters have 12 bits precision. 1x and 2x scalability cases are tested. In both cases, the BL is 1920x1080 and the EL is 10-bits. Significant coding gains are achieved using the CLUT method for both 1x and 2x test cases. Since the 1x scalability case does not include spatial scalability, its results are particularly relevant for evaluating the efficiency of the CLUT-based CGS tool; a graphical comparison between BL, EL, simulcast and scalable coding bit rates are given in Figure 100.

Table 32: performance of SHVC (SHM-6.0) with or without CGS CLUT based tool (SHM6.0).

	AI HEVC 1x 10-bit base			AI HEVC 2x 8-bit base		
	Y	**Cb**	**Cr**	**Y**	**Cb**	**Cr**
CLUT vs ref						
EL only	-49.5%	-51.9%	-57.8%	-25.4%	-25.9%	-31.4%
BL+EL	-15.1%	-20.1%	-31.2%	-13.8%	-14.7%	-20.7%
BL+EL vs simulcast						
EL only	-35.5%	-39.9%	-50.0%	-26.1%	-26.2%	-31.0%
BL+EL	-23.9%	-24.7%	-28.3%	-14.5%	-13.3%	-13.4%

	RA HEVC 1x 10-bit base			RA HEVC 2x 8-bit base		
	Y	**Cb**	**Cr**	**Y**	**Cb**	**Cr**
CLUT vs ref						
EL only	-37.9%	-40.7%	-47.2%	-16.9%	-16.2%	-22.4%
BL+EL	-14.3%	-18.9%	-28.9%	-9.2%	-9.1%	-15.7%
BL+EL vs simulcast						
EL only	-28.9%	-31.9%	-44.1%	-18.4%	-17.5%	-23.7%
BL+EL	-17.2%	-16.5%	-22.6%	-10.2%	-9.3%	-9.6%

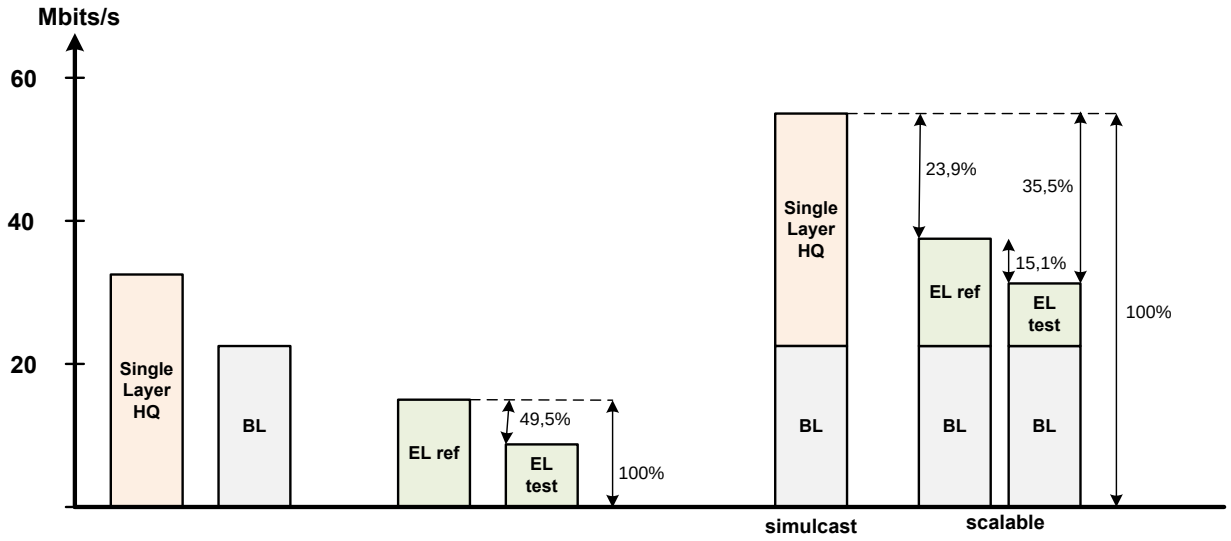


Figure 100: graphical representation of SHVC-CGS results (Table 32) for AI 1x 10-bit base case.

High Dynamic Range scalability

Although the CLUT based CGS tool in SHVC has been specially developed for colour gamut scalability cases with a priori standard dynamic range (luminance inferior to few hundred cd.m^{-2}), it can be easily combined with the bit-depth scalability feature also provided in SHVC Scalable Main-10 to provide efficient coding for High Dynamic Range scalability.

In both DVB (see 3.1.2, Figure 25) and BDA standardization committees, the distribution of High Dynamic Range (HDR) content with Wide Colour Gamut (WCG) colour space is considered. At the same time, ITU-R (ITU-R SG6/W6-C group) [131] and MPEG (Ad Hoc Group on HDR and WCG) [132] are currently defining the HDR video signal characteristics and evaluating existing technologies that can be used to efficiently distribute HDR video streams to the consumer. Backward compatibility with legacy Standard Dynamic Range is part of the requirements that may enable a smooth transition towards HDR ecosystem.

The reference HDR format considered in BDA and MPEG is Y'CbCr 2020 4:2:0, 1920x1080p or 3840x2160p using PQ-EOTF [109]. The 2014 Standard Evaluation Material (StEM) was made available by Digital Cinema Initiative to the industry and other standards organizations that require high quality program material for testing and evaluation purposes [130]. The StEM content consists of multiple versions (grades) of the same video content (~12 min duration), including one HDR version (DCI-P3, 4000 cd/m² peak luminance) and one Standard Dynamic Range (SDR)(Rec. 709, 100 cd/m²), available in RGB and XYZ. From these versions, two 4:2:0 Y'CbCr 10-bit video sequences corresponding to HDR (Rec.2020 colour space, PQ-EOTF) and SDR (Rec.709 colour space, Gamma 2.2) respectively were derived. 5 scenes of 3-second duration each (Figure 101, Figure 102) were selected.

Figure 101: The extracted Seq1 rec.709 Gamma 2.2 from StEM (left) and the corresponding HDR version in rec.2020 PQ-EOTF (right).

Figure 102: The extracted Seq2, Seq3,Seq4 and Seq5 rec.709 from StEM.

These sequences were encoded with SHM-7.0 with CLUT-based CGS tool (max 8×2×2 CLUT size) enabled using JCT-VC Random Access common test conditions. The encoder was modified in order to compute the CLUT on cropped pictures (i.e. black strips removed). Table 33 depicts the average BD-rate gains of SHM-7.0 either with CGS (WP disabled) or without CGS (WP enabled) and the gains compared to simulcast.

The achieved BD-rate gains are very significant (more than 50% EL rate reduction), and even better than the gains obtained with the regular WCG content used by JCT-VC (see Table 32). Obviously, more test material is required to draw conclusion. However these first results provide early indication that the CGS tool in SHVC can be a strong candidate for scalable coding of SDR-HDR video content.

Table 33: SHVC-CGS BD-rate gains obtained with HDR/SDR StEM video extracts compared to SHVC without CGS but WP enabled (SHM7.0).

	RA HEVC 1x 10-bit base		
	Y	**Cb**	**Cr**
CLUT vs ref			
EL only	-57.1%	-62.4%	-70.5%
BL+EL	-19.4%	-30.4%	-44.1%

BL+EL vs simulcast			
EL only	-41.5%	-48.7%	-57.0%
BL+EL	-27.9%	-27.8%	-25.2%

8.3.5 Conclusion

The CGS tool has been adopted and included in the Scalable Main 10 profile of SHVC FDAM in July 2014. The SHVC-CGS is probably the most innovative encoding tool of SHVC, given that such an encoding mode did not exist in former scalable standards such as SVC or MPEG2-scalable.

The existing Weighted Prediction tool, even if configured with multiple Inter-Layer References, does not allow reaching same level of performance as the CLUT-based CGS tool. Using WP for ILP in the case of CGS also has negative complexity implications, such as increased computation complexity and memory footprint (if multiple Inter-Layer Reference pictures are generated and stored in the DPB) [41].

Given the emerging video formats (Rec. 2020, High-Dynamic Range) and the ever increasing rendering/display capability of the end devices, recommendation and standardization groups such as DVB, ATSC and Blu-ray Disc Association, HDMI, CEA have started specifying the next generation of video experience and its delivery. SHVC-CGS can be a good candidate to provide an efficient migration path to the new formats and at the same time ensure backward compatibility with the legacy services and devices.

8.4 Colour and bit depth remapping metadata

The rendering of the reconstructed images onto the end-device display is of key importance to ensure the end-to-end service quality. However, the variability of colour formats, capture capability and display characteristics tend to increase (see 8.1). The number of applications and services using colour video is not limited to the traditional but fully controlled TV market, but also to OTT[9], internet and mobile phone video applications.

With the arrival of new and wider colour space formats (see Figure 87), the higher dynamic range used for content creation that will be proposed to the consumers soon, the compatibility with the legacy devices will have to be considered.

Colour mapping

All the rendering devices may not have the capability to adapt to any colour space nor have the required knowledge to perform the optimal colour conversion. The determination of the colour conversion function is not straightforward because the content creation workflows may include deterministic processing (Colour Space 1 towards Colour Space 2 conversion) but also non-deterministic operations such as colour grading. If two targeted displays with different characteristics are used (e.g. UHDTV display with Rec.2020 and Rec.709 HDTV display), the colour grading may be different.

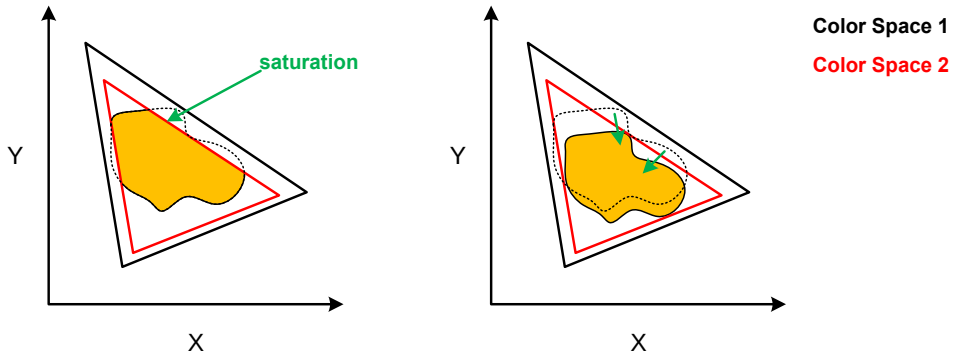


Figure 103: Example of colour conversion from Colour-space-1 to Colour-space-2 with clipping (left) or gamut compression (right).

The colour mapping from one colour space to another, depends both on the content characteristics and on the reference display: rather than saturating clipping colours one may prefer graceful colour shading for example (Figure 103).

Tone mapping

But the required adaptation is not limited to colour mapping only. The dynamic range of the encoded video should be consistent with the luminance display rendering capability. We have seen in chapter 1.1.3 that several standard OETFs exist with different characteristics. Several ones have been proposed and are considered in ITU-R (ITU-R SG6/W6-C group) [131] and MPEG (Ad Hoc Group on HDR and WCG) [132]. It may happen in the future the OETF to become a video parameter as any other ones, with same level of flexibility/variability.

In HEVC, two Supplemental Enhancement Information (SEI) messages allow remapping the reconstructed pictures luminance. The tone mapping information SEI, standardized in AVC, supports four possible tone mapping models: linear mapping with clipping, sigmoidal model, user defined lookup table, piece-wise linear model. The knee function information SEI allows parameterizing the EOTF (that determines the correspondence between one numerical luma sample value and its physical intensity

[9] OTT: Over The Top.

expressed in cd.m^{-2}) using piece-wise linear model. This is useful to know how interpreting the reconstructed input sample values and to map the signal towards the display capability, but it does not allow remapping a colour graded Rec.2020 High Dynamic Range (HDR) movie onto a legacy Rec.709 Standard Dynamic Range (SDR) screen for instance.

8.4.1 Colour Remapping Information (CRI) metadata design

We have proposed in [121] a Supplemental Enhancement Information (SEI) message carrying metadata that would help the remapping of a particular content into another standardized colour space, with possibly different dynamic range and EOTF.

Inspired from our previous works in CGS (see 8.3), these Colour Remapping Information (CRI) metadata were originally composed of a Colour LUT (CLUT) coded in the same way as described in chapter 8.3.2. It had the advantage that the colourists and content creator are familiar with this kind of model, and many authoring tools use CLUTs for applying and/or storing the results of their post-processing. If the CLUT has sufficiently bit-depth precision and size, it can be used for both colour and tone mapping at the same time [122].

However, some chips manufacturers objected the CLUT representation had two major drawbacks:

- The computation power required for CLUT interpolation (tri-linear or tetrahedral) may overcome the processing capacity of some video decoder or display systems (on-chips or graphical engines).
- The required memory storage for CLUT could be a limitation.

To respond to these concerns, the design of the CRI has been simplified to better adapt to the existing decoder chipsets or display hardware easily. These systems use graphical processing functions to transform the YCbCr colour samples into tri-chromatic RGB. This transform is carried out with 1D LUTs and 3x3 matrices combinations.

8.4.2 CRI overview

The SEI-CRI is composed of two parts. The first one groups the parameters describing the remapped signal characteristics, in particular the colour space primaries (*colour_remap_primaries*) and the transfer function EOTF (*colour_remap_transfer_function*). The second part contains the colour remapping model parameters. It is composed of a first set of piece-wise linear functions applied to each colour component (specified by the "*pre*" set of syntax elements), a 3x3 matrix applied to the three colour components, and a second set of piece-wise linear functions applied to each colour component (specified by the "*post*" set of syntax elements) (Figure 105).

The CRI is to be applied directly to the decoded (reconstructed) sample values, regardless of whether they are in the luma and chroma domain (YCbCr) or the RGB domain. The bit depth of the decoded frames and the bit depth of the remapped frames may be different.

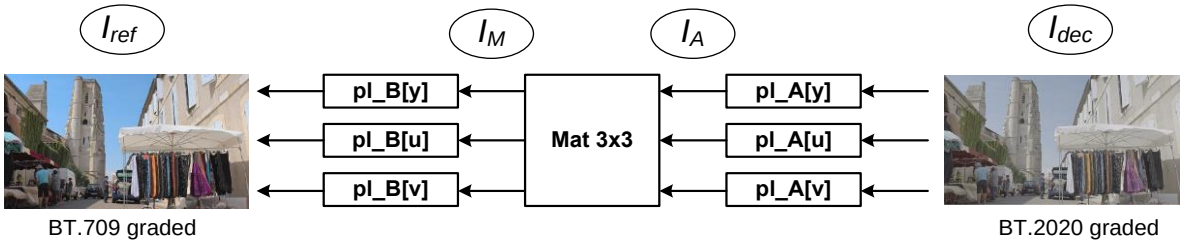
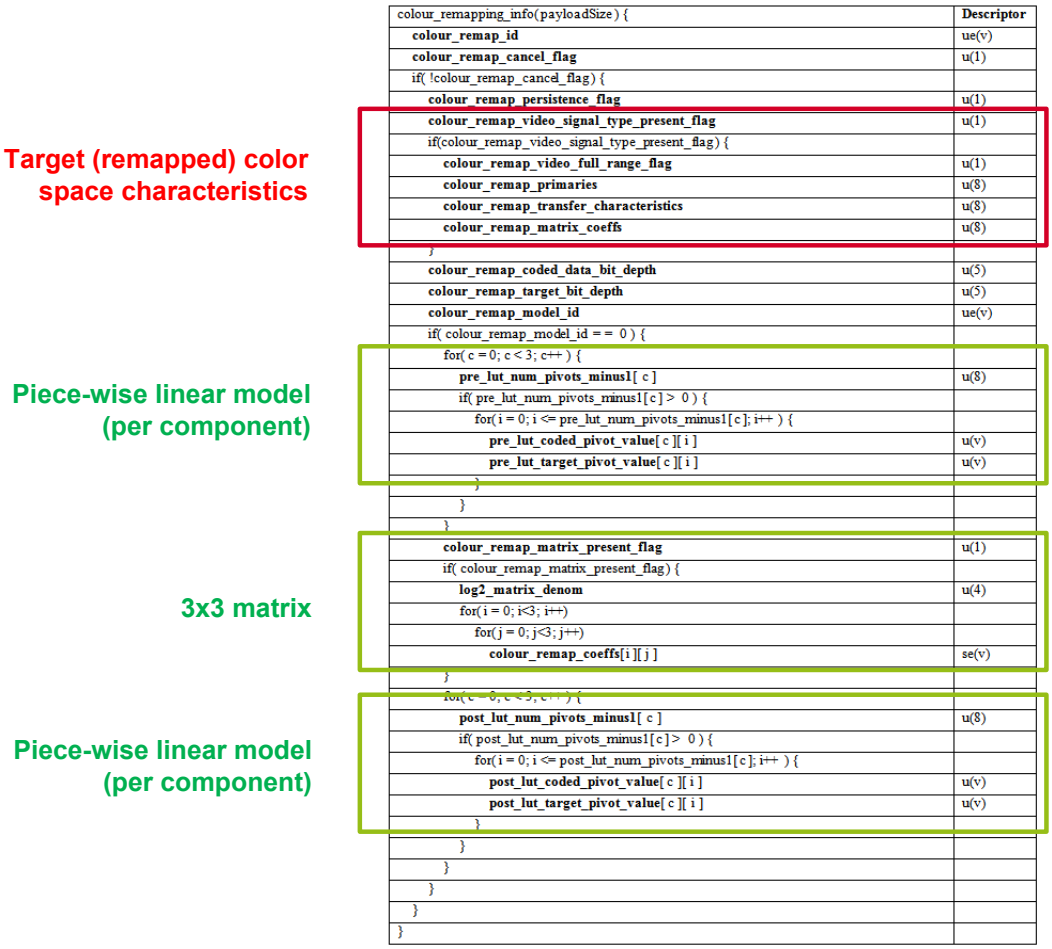


Figure 104: The colour remapping with CRI is performed in three steps, that generate intermediate images I_A and I_M.



colour_remapping_info(payloadSize) {	Descriptor
colour_remap_id	ue(v)
colour_remap_cancel_flag	u(1)
if(!colour_remap_cancel_flag) {	
colour_remap_persistence_flag	u(1)
colour_remap_video_signal_type_present_flag	u(1)
if(colour_remap_video_signal_type_present_flag) {	
colour_remap_video_full_range_flag	u(1)
colour_remap_primaries	u(8)
colour_remap_transfer_characteristics	u(8)
colour_remap_matrix_coeffs	u(8)
}	
colour_remap_coded_data_bit_depth	u(5)
colour_remap_target_bit_depth	u(5)
colour_remap_model_id	ue(v)
if(colour_remap_model_id = = 0) {	
for(c = 0; c < 3; c++) {	
pre_lut_num_pivots_minus1[c]	u(8)
if(pre_lut_num_pivots_minus1[c] > 0) {	
for(i = 0; i <= pre_lut_num_pivots_minus1[c]; i++) {	
pre_lut_coded_pivot_value[c][i]	u(v)
pre_lut_target_pivot_value[c][i]	u(v)
}	
}	
}	
colour_remap_matrix_present_flag	u(1)
if(colour_remap_matrix_present_flag) {	
log2_matrix_denom	u(4)
for(i = 0; i<3; i++)	
for(j = 0; j<3; j++)	
colour_remap_coeffs[i][j]	se(v)
}	
for(c = 0; c < 3; c++) {	
post_lut_num_pivots_minus1[c]	u(8)
if(post_lut_num_pivots_minus1[c] > 0) {	
for(i = 0; i <= post_lut_num_pivots_minus1[c]; i++) {	
post_lut_coded_pivot_value[c][i]	u(v)
post_lut_target_pivot_value[c][i]	u(v)
}	
}	
}	
}	
}	
}	

Figure 105: CRI syntax overview.

8.4.3 CRI model parameters determination

The determination of the CRI parameters (P_{CRI}) can be considered as a minimization problem (54):

$$\arg\min_{(P_{CRI})} dist(I_{ref}, remap(I_{dec})), \qquad (54)$$

Where I_{dec} is the decoded frame, $remap(I_{dec})$ is the result of the colour remapping of I_{dec} using P_{CRI} and I_{ref} is the target frame. For example, in case of a video sequence first graded in Rec.2020 and next graded in Rec.709, the primary stream may correspond to the Rec.2020 grade, the CRI represents the model parameters to be used to remap the decoded Rec.2020 colour samples (I_{dec}) into Rec.709 colour space. In that case, I_{ref} is the Rec.709 grade, and the CRI is used to display the decoded frames onto Rec.709 displays.

To solve (54), we decompose the problem in three parts as depicted in Figure 104. The first stage corresponds to the first piece-wise linear transformations *pl_A[c]* applied on each colour component *c* independently, the second stage is the 3x3 matrix and the last stage is the second set of piece-wise linear transformations *pl_B[c]*. The output of each stage transformation is an image that is also the input of the next stage.

Piece-wise linear transforms determination

A piece-wise linear function *f* is defined by intervals $[X_i;X_i+1]$ and is linear in each interval. Here we consider the case the intervals have equal range (equal to 1) for simplicity, but equivalent reasoning

derived from [123] can be extended to the general case (un-equal range intervals). Then in the following, the values X_i are considered as known.

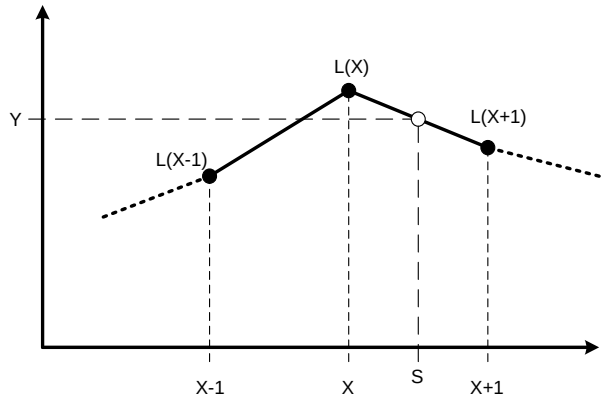


Figure 106: piece-wise linear function.

For a given point with abscise $s \in [X_i;X_i+1]$, the corresponding image by f is y such as:

$$y = f(s) = L(X_i) + (L(X_i+1)-L(X_i))*(s-X_i)$$

For each interval $[X_i ; X_i+1]_{i=0,..N}$, $L(X_i)$ is the value minimizing the sum of the quadratic errors $Err(X_i) = (y_k - f(s_k))^2$ for the set of sample values (s_k,y_k), with $s_k \in [X_i ; X_i+1]$:

$$\arg\min_{L(Xi)} \sum_k \left(y_k - f(s_k)\right)^2, \; s_k \in [X_i ; X_i+1] \qquad (55)$$

where: s_k is one sample value of one colour component of the input frame,
y_k is the corresponding sample value of the same colour component in the output frame.

The Least Square Minimization (LSM) method consists in solving the set of N equations of partial derivative of $Err(X_i)$ respectively to $L(X_i)_{\ i=0,..N}$ equal to zero, N being the number of intervals of the piece-wise linear function.

The value of $L(Xi)$ determines the function f on both intervals $[X_i-1 ; X_i]$ and $[X_i ; X_i+1]$:

$$y = f(s_m) = L(X_i-1) + (L(X_i)-L(X_i-1))*(s_m-X_i+1) \quad \text{if } s_m \in [X_i-1 ; X_i] \qquad (56)$$

and $$y = f(s_k) = L(X_i) + (L(X_i+1)-L(X_i))*(s_k-X_i) \quad \text{if } s_k \in [X_i ; X_i+1] \qquad (57)$$

The spatial derivative of $Err(X_{i-1})$ with respect to $L(X_{i-1})$ and the spatial derivative of $Err(X_i)$ with respect to $L(X_i)$ and $L(X_{i+1})$ depend on $L(X_i)$. Then we obtain the following equations:

$$\frac{\partial \mathrm{Err}}{\partial \mathrm{L_{X_i-1}}} = \sum_m -2\left(\mathrm{y_m} - \mathrm{f(s_m)}\right)(-\mathrm{s_m} + \mathrm{X_i}) = 0 \qquad \text{with } \mathrm{s_m} \in [X_i-1 ; X_i] \qquad (58)$$

$$\frac{\partial \mathrm{Err}}{\partial \mathrm{L_{X_i}}} = \sum_m -2\left(\mathrm{y_m} - \mathrm{f(s_m)}\right)(\mathrm{s_m} - \mathrm{X_i} + 1) - \sum_k 2\left(\mathrm{y_k} - \mathrm{f(s_k)}\right)(1 - \mathrm{s_k} + \mathrm{X_i}) = 0 \qquad (59)$$

with $\mathrm{s_m} \in [X_i-1 ; X_i]$, $\mathrm{s_k} \in [X_i ; X_i+1]$

$$\frac{\partial \mathrm{Err}}{\partial \mathrm{L_{X_i+1}}} = \sum_k -2\left(\mathrm{y_k} - \mathrm{f(s_k)}\right)(\mathrm{s_k} - \mathrm{X_i}) = 0 \qquad \text{with } \mathrm{s_k} \in [X_i ; X_i+1] \qquad (60)$$

Once replacing $f(s_m)$ and $f(s_k)$ by the expressions (56) and (57), this is equivalent to:

$$\begin{cases} Z_0 = a_{0,0} \times L(X_0) + a_{0,1} \times L(X_1) \\ \dots \\ Z_i = a_{i,i-1} \times L(X_i - 1) + a_{i,i} \times L(X_i) + a_{i,i+1} \times L(X_i + 1) \\ \dots \\ Z_N = a_{N,N-1} \times L(X_{N-1}) + a_{N,N} \times L(X_N) \end{cases}$$

where the terms Z_i and a_i are only dependant on the values (s_k,y_k).

If we apply the same reasoning for the other intervals, we obtain the following system (in red the coefficients obtained with the two first intervals [X_0;X_1] and [X_1;X_2]):

$$\begin{pmatrix} Z_0 \\ \vdots \\ \vdots \\ \vdots \\ Z_N \end{pmatrix} = \begin{pmatrix} a_{0,0} & a_{0,1} & 0 & \cdots & \cdots & 0 \\ a_{1,0} & a_{1,0} & a_{1,2} & 0 & \ddots & \vdots \\ 0 & a_{2,1} & a_{2,2} & a_{2,3} & 0 & \vdots \\ 0 & \ddots & & & & 0 \\ \vdots & \ddots & 0 & a_{N-1,N-2} & a_{N-1,N-1} & a_{N-1,N} \\ 0 & \cdots & \cdots & 0 & a_{N,N-1} & a_{N,N} \end{pmatrix} \begin{pmatrix} L(X_0) \\ \vdots \\ \vdots \\ \vdots \\ L(X_N) \end{pmatrix}$$

where: $a_{i,i} = \sum_m (s_m - X_{i-1})^2 + \sum_k (s_k - X_i)^2 - 2(s_k - X_i) + 1$, with $s_m \in [X_i\text{-}1\ ;\ X_i]$, $s_k \in [X_i\ ;\ X_i\text{+}1]$
(for i=0, first term indexed by *m* is omitted, for i=N, second term indexed by *k* is omitted)

$a_{i+1,i} = a_{i,i+1} = \sum_m -(s_m - X_{i-1})^2 + (s_m - X_{i-1})$

$Z_i = \sum_m (s_m - X_{i-1}) y_m + \sum_k (1 - s_k + X_{i-1}) y_k$
(for i=0, first term indexed by *m* is omitted, for i=N, second term indexed by *k* is omitted)

Next, the values $L(X_i)$ are derived by inverting the matrix above, the values Z_i and $a_{i,j}$ being derived from the input (s_k) and output (y_k) frames samples.

Matrix determination

The determination of the matrix (*M*) is based on the same principle as described in 8.3.1. The least square minimization (LSM) method applied on the quadratic error $(I_M\text{-}M.I_A)^2$ is used to derive the matrix coefficients, where I_A and I_M are the intermediate images depicted in Figure 104.

Overall model determination

The derivation of the whole model is made through an iterative algorithm, by estimating successively one of the model components (LUTs or matrix).

At the first stage, the frames I_{dec} and I_{ref} are available only, corresponding to E1 and E4 respectively in Figure 107. The model is initialized with identity matrix and linear functions. Then the intermediate images E2 and E3 are directly computed with the model, and one can estimate the functions $f_{X2,X=Y,U,V}$ using the piece-wise linear transform determination algorithm presented above.

However, the piece-wise linear functions f_{X2} may not be invertible in general. That is why we introduce three additional piece-wise linear functions $f_{X2}^{-1}{}_{X=Y,U,V}$ for the purpose of estimating E3 from E4 only. Next one can derive M from E2 and E3, using the matrix determination method.

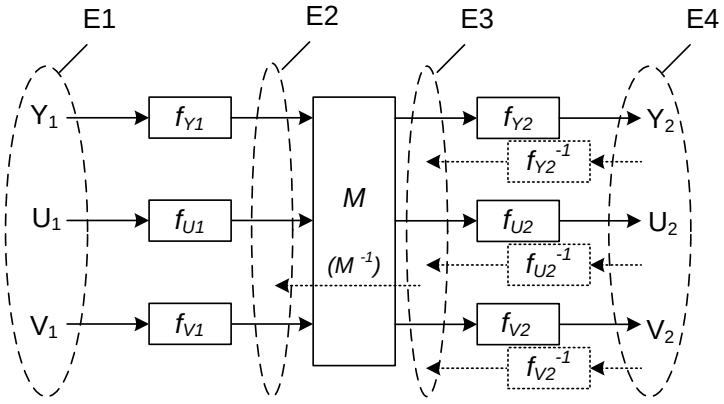


Figure 107: Colour mapping transformation model.

The invert of the matrix *M* is used to up-date E2 and to derive the functions f_{X1}. Note that one could have used the same principle as for f_{X2}^{-1}, but in our experiments, *M* was always invertible.

This algorithm is iterated several times, which allows refining the model until a stop criteria is met. In our case, we compute the PSNR between $remap(I_{dec})$ and I_{ref}, and we stop when its value is no more increasing.

8.4.4 Experimental results

Wide Colour Gamut content

The CRI model estimation method has been tested on the WCG sequences described in 8.2.3. Typical use case of DVB-UHDTV-phase-1 has been considered, where an encoded Rec.2020 video sequence is distributed with CRI metadata that have been computed with the graded Rec.709 version of the content. In Table 34, the difference between the colour re-mapped Rec.2020 frames using the CRI and the targeted Rec.709 grade is measured using PSNR, SSIM and delta-E76 metrics (see 0). One SEI-CRI per frame was used, but the results with one CRI per scene gives equivalent results since these sequences have very stable characteristics actually. The LUTs size is 33 knots, with 10-bit precision. The overall CRI data amount is about 4k bits, which is relatively negligible compared to a typical UHDTV broadcast bit rate (15 Mbits/s, 50fps => 300 kbits per frame in average) and all the more regarding typical Blu-ray bit rates (70 Mbits/s => 1.4 Mbits per frame in average).

Table 34: comparison of Rec.2020 grade colour remapped to Rec.709 grade using CRI, with the reference Rec.709 grade.

	PSNR (dB)			SSIM			Delta-E76
1920x1080 (10bits)	**Y**	**U**	**V**	**Y**	**U**	**V**	**(dB)**
Birthday	40.06	39.22	47.85	0.99	0.95	0.99	44.73
BirthdayFlashPart1	45.77	46.31	48.37	0.99	0.98	0.99	46.26
BirthdayFlashPart2	43.42	44.31	49.28	0.98	0.98	0.99	44.89
Parakeets	45.53	47.52	51.12	0.98	0.98	0.98	46.28
TableCar	33.32	37.39	36.09	0.94	0.93	0.93	41.84
Average	41.62	42.95	46.54	0.976	0.964	0.976	44.80

Table 35: comparison of Rec.2020 grade colour converted to Rec.709 with RP-177 [10], with the reference Rec.709 grade.

	PSNR (dB)			SSIM			Delta-
1920x1080 (10bits)	**Y**	**U**	**V**	**Y**	**U**	**V**	**E76 (dB)**
Birthday	33.26	39.11	34.80	0.99	0.96	0.98	41.62
BirthdayFlashPart1	41.11	42.73	33.99	0.97	0.97	0.99	42.48
BirthdayFlashPart2	36.34	38.78	37.03	0.98	0.98	0.98	41.89
Parakeets	37.57	41.03	37.84	0.98	0.97	0.96	42.41
TableCar	21.88	30.42	26.60	0.93	0.91	0.92	36.32
Average	34.03	38.41	34.05	0.970	0.957	0.966	40.94

The Table 35 provides with the difference between the targeted Rec.709 grade and the Rec.2020 frames converted to Rec.709 using the standardized colour space signal conversion RP 177-1993 [10]. Since this signal conversion does not take into account the artistic intent, the results may significantly differ with the targeted BT709 grade, as depicted in Table 35 and in Figure 108 with the sequence *TableCar* example (bottom-left).

The comparison of Table 34 and Table 35 makes clear the frames colour remapped with CRI out-performs the default colour space conversion process. This is confirmed with subjective assessment, as

depicted with *TableCar* example in Figure 108: top-right (Rec.709 grade) and bottom-right (CRI) looks very close, contrary to bottom-left (RP-177 default colour space conversion).

Figure 108: BT2020 graded (top-left), BT709 graded (top-right), BT2020 colour space conversion to BT709 (bot-left), BT2020 colour remapped with CRI (bot-right).

High Dynamic Range content

To evaluate the performance of the CRI for remapping HDR content into SDR format, the same 5 sequences extracted from the *StEM* colour grades and provided by DCI as presented in 8.3.4 have been used. The efficiency of the CRI is measured by computing the difference between the remapped HDR and the targeted SDR version, using PSNR, SSIM and deltaE-76 metrics (see 0). These relatively good values are confirmed with subjective viewing.

StEM	PSNR (dB)			SSIM			Delta-E76 (dB)
1920x1080 (10bits)	Y	U	V	Y	U*	V*	
Seq_0	42.97	38.02	33.49	0.99	0.86	0.85	41.44
Seq_1	45.90	39.72	40.80	0.98	0.90	0.92	43.52
Seq_2	48.96	38.74	40.03	0.99	0.88	0.91	43.22
Seq_3	49.64	37.50	36.97	0.99	0.89	0.85	42.74
Seq_4	47.82	36.90	36.51	0.99	0.89	0.88	42.21
Average	47.06	38.18	37.56	0.99	0.88	0.88	42.63

*informative only: it is commonly accepted the ssim is meaningful for Luma signal mainly.

8.4.5 Conclusion and perspectives

The Colour Re-mapping Information (CRI) SEI message represents a model to colour remap the reconstructed frames. It has been specially designed to allow backward compatibility of new video formats with legacy display systems, while preserving artistic intent when several grades exist.

This study has shown the efficiency of the CRI on WCG content beyond traditional Rec.709 colour gamut. The specification of the CRI is not limited to Standard Dynamic Range (SDR) video formats but is flexible enough to support different bit-depth of the reconstructed and the remapped frames. In particular,

the use of separate LUTs per component allows compensating various EOTFs, including the HDR to SDR remapping cases.

The CRI-SEI message has been adopted in edition 2 of HEVC in July 2014. A software implementation of the CRI parameters estimation algorithm described in this section has been provided to the JCT-VC group in [124]. The CRI is currently under consideration for DVB-UHDTV-phase-2 and for the next generation of the Blu-ray Disc.

9. Discussion and Conclusion

Table 36 summarizes current stakes of the video coding adaptation. Here, the purpose is not to give an exhaustive status of all the current services and techniques deployed, but to try providing some insight of typical use cases.

For several classes of applications, we list the corresponding major challenges they face. In the next columns, the primary (a) and secondary (b,c) video coding techniques that are commonly considered as the best trade-off for fulfilling these requirements are given. The last two columns contain comments about the advantages and limitations of the primary and secondary solutions respectively.

Table 36: video coding adaptation techniques for various applications.

Application	Challenge				Technique				Advantages	Limitations
	One-to-one	One-to-many	Compatibility (end-device heterogeneity)	Bandwidth heterogeneity/variation	Transcoding	Scalable codec	simulcast	Stream-switching		
Contribution	x			x	x				Tailored to network bandwidth	Coding artefacts
Distribution (Live)		x	x	-		c	a	b	(a) simplicity (b) BW adaptation (c) coding efficiency	(a,b) bandwidth cost (c) codec complexity
VOD (off-line)		x		x		b		a	(a) High compatibility with decoder/bw heterogeneity (b) storage	(a) storage (b) codec complexity
Home network	x		x		x				Tailored to end device capability	Need high computation power
Visio conference		x	x			a	b		(a) Bandwidth saving (b) codec simplicity	(a) codec complexity (b) bandwidth
DVD,Blu-Ray	x		x			a	b		(a) storage saving (b) codec compatibility	(a) codec complexity (b) storage size

The purpose of the **Contribution** links is to transport high quality video content (ex: by satellite) for professional applications, from a remote source (such as an outside broadcast unit) to a broadcaster's studio or from the studio to a satellite TV uplink center for instance. Then, transcoding is used to offer the best quality as possible, tailored to the fixed bandwidth with minimum visible artefacts.

The **video distribution** covers the TV broadcast or IPTV. In this one-to-many case, the end devices heterogeneity can be important since the video can be displayed on Smart phones, SDTV or HDTV for

instance. Moreover, when new services such as UHDTV (or may be later HDR) will be deployed, the backward compatibility with legacy devices will have to be addressed. For IPTV case, the access networks that connect the consumer homes to the local digital subscriber line access multiplexer (DSLAM) can have various bandwidth capabilities depending on the regions. Up to now, TV broadcast has chosen simulcast (ex: France's Television Numérique Terrestre where programs are simultaneously transmitted in SDTV and HDTV formats) because it has the advantage of simplicity, but it needs higher bandwidth resources than scalable coding. For IPTV, both simulcast and stream-switching co-exist.

The **Video On demand (VOD)** can be seen as the equivalent of video distribution case but for off-line encoded streams. Today, one major issue with VOD is the tremendous storage amount required for addressing both the end devices heterogeneity and the various bandwidth links. The stream-switching has the advantage of offering an efficient adaptive streaming solution, whose complexity is transparent to the user. The scalable coding may reduce the storage requirements at the servers' side but it would increase the complexity of the end device decoder possibly. Besides, the encoded stream generation should be able to manage the inter-stream dependency and to find the right layers bit-rates trade-off.

The **Visio conference** is one of the rare domains where the scalable coding has been deployed effectively. In a framework where the end device heterogeneity and the bandwidth variation can be high, the scalable coding can successfully compete against simulcast. In this case, most of the player devices are software based, what soften the relative decoder complexity.

The **DVD** or **Blu-ray Disc** are the only of the listed cases where the video streams are physically stored. The memory space capacity on one disc may not allow storing several instance of the movie (simulcast). The use of several discs can be seen as a product weakness while increasing the unit costs. The scalable coding could be an attractive alternative for fulfilling backward compatibility requirements, but the CE manufacturers should agree with implementing scalable decoder hardware players.

This table figures out the scenarios diversity and the different technical solutions commonly envisaged for dealing with the video coding adaptation issue. This diversity justifies the use of various solutions. In the previous chapters, we have presented some technical improvements that could be applied for improving the performance of some of these solutions. But they will probably continue to co-exist for a while, because they all have their advantages and limitations, and no technical unified solution can substitute to any of them.

General conclusion

Since the last few years, the video formats have significantly evolved compared to the relative conservative stagnation of the last 2 decades (the Rec.709 for HDTV was first released in 1990). New recommendations have been published (ITU-T, SMPTE) that impact several dimensions at the same time: colour space (Rec.2020), frame size (4K, 8K), High Dynamic Range (EOTFs) or frame rate (see High frame rate workshops organized by EBU). These evolutions originate with the recent technical progress made by the display manufacturers as well as the increasing (professional) cameras capabilities.

Besides, the new social behavior tends to favor the growing of services and applications based on video distribution (exchange, streaming, etc…), at anytime, anywhere, and on any device. Given both the variable nature of the mobile and wireless networks bandwidth, and the versatility of the unmanaged networks (OTT), the video compression should be able to provide efficient tool for delivering dynamically adapted bit rate streams.

This thesis has presented a set of studies addressing the improvement of the last generation codecs HEVC and AVC, while providing added flexibility corresponding to this increasing demand of adaptability.

Then in our work, we have first focused on video compression efficiency by incorporating different levels of motion representation and partitioning, which can bring significant gains. We have designed a video encoding algorithm that preserves high level of performance while limiting the complexity.

We have shown that further improvement of the video coding can be achieved if the input signal features some particular properties. This paradigm has been demonstrated with HEVC in two cases: the content-adaptive colour space transform (pre-processing) and the weighted prediction.

But the video distribution imposes specific constraints that cannot be fulfilled with higher compression efficiency only, but need to quickly react to bandwidth variations as well as to the video signal variability for example. As for many other hybrid video codecs, in HEVC the quantization step is the most straightforward tool used to temporally and locally adapt the compression effort. We have proposed a prediction of the quantization parameter (QP) derived from the motion vector similarity with neighboring QGCU that competes equally with other proposed methods.

The benefit of local quantization has been illustrated in a perceptually adapted video encoder based on quality maps computed with a psychovisual model. An iterative algorithm for finding an optimal local QP allocation has been developed and tested on several video sequences. It allows homogenizing the quality maps while preserving the same overall coding costs.

However this is not sufficient to insure high bandwidth adaptability for off-line encoded video bit-streams. For some applications such as contribution that require fine adjustment to the end user requirements, the transcoding is necessary. But for VOD streaming, the stream-switching is widely deployed. We have demonstrated how it can be combined with adaptive resolution coding while reducing the frequency of switching points. We have shown this principle can be implemented within SHVC with some specific parameterization.

If the scalable extension of HEVC (SHVC) can combine all these levels of adaptation, it was not dealing efficiently with the colour space format conversion. Given the emergence of new video formats with wider colour gamut (WCG) and higher dynamic range possibly, it was necessary to consider this fourth scalability dimension. In this thesis, we have shown how important this new feature could be in a near future. We have created a set of experimental WCG content that have been used in MPEG to test this new functionality. An inter-layer prediction has been designed and evaluated based on 3D colour LUTs. Several refinements have been discussed. The final gains are in-line with the other SHVC scalability dimensions performances.

Besides, it turned out the parameters of this colour space and bit-depth conversion model are colour metadata that can be used as such for carrying backward compatibility (colour remapping) information. A simple model has been derived for general purpose and wide usage potentially, since it requires very few or existing hardware resources only.

Within this document, several of the described techniques have been proposed to MPEG and some of them have been adopted in the final HEVC standard and its version 2 extensions, and in the last version of the Blu-ray Disc. They could now be pushed into recommendation organizations (DVB, ATSC, HDMI…) for practical products and services deployment.

Perspectives

In comparison with former standards, HEVC and its extensions has integrated an unprecedentedly level of adaptability. MPEG is currently studying additional tools for bringing even more formats and adaptability to HEVC: JCT-VC is developing new tools for Screen Content Coding and the MPEG-XYZ AdHoc'Group is currently defining the requirements and explorations for High Dynamic Range and Wide Colour Gamut content distribution.

We predict this trend will strengthen in the next decade and the Future Video Coding standards should integrate a minimum level of adaptation.

The complexity gap between HEVC and the former AVC standard is very less important than between AVC and MPEG-2, in particular if we consider the decoder. If this (low complexity) decoder constraint is relaxed, it makes no doubt the compression efficiency of the future video codec could increase significantly. For low bit rates, the finding of a visual metric that better correlate with the subjective quality than PSNR, would also facilitate the advent of new codec technologies.

The future metadata may implement more complex models for local adaptation to better cope with region based colour grading. Besides, given the high variety of display capabilities, a model that could generate intermediate colour grades and/or peak luminance could be helpful to better tailor the video signal to the particular display. But this is also conditioned to the acceptability of display manufacturers to integrate standardized technologies rather than proprietary (colour) post-processing solutions.

Bibliography

[1] Wei Dai, Madhu Krihnan, Pankaj Topiwala, "In Support of a Still Image Profile of HEVC v1," JCTVC-K0285, 10th Meeting: Shanghai, CN, 10-19 October, 2012.

[2] P.Hanhart, M.Rerabek, P.Korshunov and T.Ebrahimi, "Subjective Evaluation of HEVC Intra Coding for Still Image Compression," Proceedings of Seventh International Workshop on Video Processing and Quality Metrics for Consumer Electronics (VPQM), January 30-February 1, 2013, Scottsdale, Arizona.

[3] A. Luthra, J.R.Ohm, J.Ostermann, "Use cases of the scalable enhancement of HEVC," ISO/IEC JTC1/SC29/WG11 N12782, May 2012, Geneva, Switzerland.

[4] A. Luthra, J.R.Ohm, J.Ostermann, "Requirements of Scalable Coding Extensions of HEVC," ISO/IEC JTC1/SC29/WG11 N12783, May 2012, Geneva, Switzerland.

[5] "Parameter values for ultra-high definition television systems for production and international programme exchange," Recommendation ITU-R BT.2020, 08.2012.

[6] P.Andrivon, **P.Bordes**, E.François (Technicolor), X.Ducloux (TVN), Y.Ye (InterDigital), A.Duenas (NGCodec Inc), A.Segall (Sharp Labs), K.Ugur (Nokia), X.Li (Qualcomm), E.Alshina (Samsung), "AHG14: On a profile supporting Bit Depth and Colour Gamut Scalability in HEVC Scalable extensions," JCTVC_P0127, 16th JCTVC Meeting, San José, USA, 9–17 Jan. 2014.

[7] L. Kerofsky, A.Segall, SH.Kim, K.Misra, "Colour Gamut Scalable Video Coding," JCTVC-K0241, 11th JCTVC Meeting, Shanghai, China, Oct. 2012, and also published in Data Compression Conference (DCC), March 20-22, 2013.

[8] G. Sullivan, "Colour format upconversion for video display," Joint Video Team (JVT), San Diego, 2003.

[9] "Delivery of wide colour gamut image content through SDTV and HDTV delivery systems," Report ITU-R BT.2250.

[10] "Derivation of Basic Television Colour Equations," SMPTE Recommended Practice, RP 177-1993.

[11] CineStyle Colour Assist, https://www.technicolorcinestyle.com/cinestyle-color-assist/

[12] "Digital Cinema System Specification version 1.2", Digital Cinema Initiatives, October 2012.

[13] Sony F65 overview, http://www.pro.sony.eu/pro/lang/en/eu/product/35mmdigitalmotionpicturecamerascamcorders/f65/overview , retrieved on 12th July 2013.

[14] Mitsubishi LaserVue, http://www.mitsubishi-tv.com/tv/L75-A96, retrieved on 12th July 2013.

[15] Assimilate Scratch, http://www.assimilateinc.com/products/scratch, retrieved on 12th July 2013.

[16] Digital Picture eXchange, ANSI/SMPTE 268M-2003.

[17] OpenEXR, http://www.openexr.com, retrieved on 12th July 2013.

[18] P.Andrivon, **P.Bordes**, "AHG14: Wide Colour Gamut Test Material Creation," JCTVC-N0163, 14th JCTVC Meeting, Vienna, AT, 25 July – 2 August 2013.

[19] A. Duenas, A. Segall (co-chairs), **P. Bordes**, J. Dong, D.-K. Kwon, X. Li (vice-chairs), "JCT-VC AHG report: Colour gamut scalability (AHG14)," JCTVC-O0014, 15th Meeting, Geneva, CH, 23 Oct. – 1 Nov. 2013.

[20] **P.Bordes**, P.Andrivon, R.Zakizadeh, "AHG14: Colour Gamut Scalable Video Coding using 3D LUT," JCTVC-M0197, 13th Meeting: Incheon, KR, 18–26 Apr. 2013.

[21] J.Chen, J.Boyce, Y.Ye, M.Hannuksela, GJ.Sullivan, YK.Wang, "HEVC Scalable Extensions (SHVC) Draft Text 7 (separated text)," JCTVC-R1008, 18th Meeting: Sapporo, JP, 30 June – 9 July 2014.

[22] X.Li, J.Boyce, P.Onno, Y.Ye, "Common Test Conditions and Software Reference Configurations for the Scalable Test Model", JCTVC-L1009, Geneva, CH, January 2013.

[23] G. Bjontegaard, "Improvements of the BD-PSNR model," in ITU-T SG16 Q.6 Document, VCEG-AI11, 2008.

[24] **P.Bordes**, P.Andrivon, R.Zakizadeh, "Colour Gamut Scalable Video Coding For SHVC," 30th Picture Coding Symposium (PCS-2013), Dec. 8-11, 2013, San Jose, California.

[25] **P.Bordes**, P.Andrivon, P.Lopez, F.Hiron, "AHG14: Colour Gamut Scalable Video Coding using 3D LUT: New Results," JCTVC-N0168, 14th Meeting: Vienna, AT, 25 July – 2 Aug. 2013.

[26] Y.He, Y.Ye, J.Dong, "SCE1: Combined bit-depth and colour gamut conversion with 3D LUT for SHVC colour gamut scalability," JCTVC-P0186, 16th Meeting: San José, USA, 9 – 17 Jan. 2014.

[27] **P.Bordes**, P.Andrivon, E.François, F.Hiron, " SCE1: Results on Core Experiment on Colour Gamut and Bit-Depth Scalability, tests 1A & 1B," JCTVC-P0128, 16th Meeting: San José, US, 9–17 Jan. 2014.

[28] K.Kanamori, H.Kawakami, H.Kotera, "A novel colour transformation algorithm and its applications," SPIE Image Processing Algorithms and Techniques, 1244, 272–281, 1990.

[29] J. W. Ahn, "Simd optimization of tetrahedral interpolation for colour management system," Korea Institute of Information Technology, 9(5), 93–101, 2011.

[30] **P.Bordes**, G.Clare, F.Henry, M.Raulet, J.Viéron, "An overview of the emerging HEVC standard," International Symposium on Signal, Image, Video and Communications, ISIVC, Valenciennes, July 2012.

[31] G.Sullivan, J.R.Ohm, W.J.Han, T.Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," IEEE Transaction on Circuits and Systems for Video Technology, vol.22, No.12, Dec. 2012.

[32] J.R.Ohm, G.Sullivan, H.Schwarz, T.K.Tan, T.Wiegand, "Comparison of the Coding Efficiency of Video Coding Standards—Including High Efficiency Video Coding (HEVC)," IEEE Transaction on Circuits and Systems for Video Technology, vol.22, No.12, Dec. 2012.

[33] P.Andrivon, **P.Bordes**, P.Salmon, " Comparison of Compression Performance of HEVC Draft 10 with AVC for UHD-1 material," JCTVC-M0166, 13th Meeting: Incheon, KR, 18–26 Apr. 2013.

[34] T.K.Tan A. Fujibayashi, Y. Suzuki, J.Takiue, "[AHG 8] Objective and subjective evaluation of HM5.0," JCTVC-H0116, JCTVC 8th Meeting: San José, CA, USA, 1–10 February, 2012.

[35] M.Winken, P.Helle, D.Marpe, H.Schwartz and T.Wiegand, "Transform Coding in the HEVC Test Model," IEEE International Conference on Image Processing ICIP 2011, Brussels, 2011.

[36] J.-M. Thiesse, J. Viéron, "On Tiles and Wavefront tools for parallelism", JCT-VC document JCTVC-I0198, Geneva, April, 2012.

[37] X.Li, Y.He, **P.Bordes**, "Non-SCE1: Asymmetric 3D LUT for Colour Gamut Scalability," JCTVC-Q0063, JCTVC 16th Meeting: San Jose, US, 9–17 Jan. 2014.

[38] X.Li, Y.He, **P.Bordes**, "SCE1.2: Colour gamut scalability with asymmetric 3D LUT," JCTVC-Q0048, JCTVC 17th Meeting: Valencia, ES, 27 March – 4 April 2014.

[39] X. Li, J. Chen, M. Karczewicz, **P. Bordes**, C. Auyeung, K. Misra, S.-H. Kim, A. Segall, P. Andrivon, E. François, F. Hiron, "Non-SCE1: Improved CGS partitioning on top of SHM-6," JCTVC-R0151, JCTVC 18th Meeting: Sapporo, JP, 30 June – 9 July 2014.

[40] A.Duenas et al., "Description of HEVC Scalable Extensions Core Experiment SCE1: Colour Gamut Scalability," JCTVC-Q1101, JCT-VC 17th Meeting: Valencia, ES, 27 March – 4 April 2014.

[41] D.Bugdayci, A.Aminlou, K.Ugur, M.Hannuksela, "SCE1: Test 5.1 results on improved weighted prediction for colour gamut scalability," JCTVC-R0239, 18th Meeting: Sapporo, JP, 30 June – 9 July 2014.

[42] M.Paul et al., "Pattern-based video coding with dynamic background modeling," EURASIP Journal, 2013.

[43] O. D. Escoda, P. Yin, C. Dai, and X. Li, "Geometry-adaptive block partitioning for video coding," in Proc. IEEE ICASSP, 2007, pp. 657–660.

[44] R. Mathew, D. Taubman, "Joint scalable modeling of motion and boundary geometry with quad-tree node merging," in Proc. IEEE ICIP, 2009, pp. 3745–3748.

[45] S. Kondo and H. Sasai, "A motion compensation technique using sliced blocks in hybrid video coding," in Proc. IEEE ICIP, 2005, pp. 305–308.

[46] M. Karczewicz, P. Chen, R. Joshi, X. Wang, W-J. Chien, R. Panchal, "Video coding technology proposal by Qualcomm Inc," JCTVC-A121, Dresden, Germany, April 2010.

[47] T. Wiegand, W.-J. Han, J.-R. Ohm, G. J. Sullivan, "High Efficiency Video Coding (HEVC) text specification Working Draft 1," JCTVC-C403, Guangzhou, China, October 2010.

[48] E.M. Hung, R.L. De Queiroz, and D.Mukherjee, "On macroblock partition for motion compensation," in Proc. IEEE ICIP, 2006, pp. 1697–1700.

[49] A.A. Muhit, M.R. Pickering, and M.R. Frater, "A fast approach for geometry-adaptive block partitioning," in Proc. PCS, 2009, pp. 1–4.

[50] L. Guo, P. Yin, Y. Zheng, X. Lu, Q. Xu, J. Sole, "Simplified Geometry-adaptive block partitioning for video coding," in Proc. IEEE ICIP, 2010, pp. 965–968.

[51] L. Guo, P. Yin, E. Francois, "TE 3: Simplified Geometry Block Partitioning," JCTVC-B085, Geneva, Swissland, July 2010.

[52] E.François, **P.Bordes**, L.Guo, M.Karczewicz, "CE2: Simplified Geometry Block Partitioning," JCTVC-D230, Daegu, KR, 20-28 January, 2011.

[53] K. Vermeirsch, J. De Cock, S. Notebaert, P. Lambert, J. Barbarien, A. Munteanu and R. Van de Walle, "Efficient adaptive-shape partitioning of video," in Multimedia Tools and Applications, Springer Netherlands, Sept 2010.

[54] T. Wiegand, H. Schwarz, A. Joch, F. Kossentini, and G.J. Sullivan, "Rate-Constrained Coder Control and Comparison of Video Coding Standards," IEEE TCSVT, Vol. 13, No. 7, pp. 688-703, July 2003.

[55] Frank Bossen, "Common test conditions and software reference configurations", Doc. JCTVC-C500 , JCT-VC of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Guangzhou, China, October 2010.

[56] G. Bjontegaard, "Improvements of the BD-PSNR model," in ITU-T SG16 Q.6 Document, VCEG-AI11, 2008.

[57] K.McCann, W.-J.Han, I.-K.Kim, “Samsung’s Response to the Call for Proposals on Video Compression Technology”, Doc. JCTVC-A124, JCT-VC of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Dresden, Germany, April 2010.

[58] **P.Bordes**, E.François, D.Thoreau, “Fast Encoding Algorithms for Geometry-Adaptive Block Partitioning,” International Conference on Image Processing 2011, ICIP 2011, Brussels (Belgium), Sept. 2011.

[59] P. Sand and S. Teller, “Particle video: long-range motion estimation using point trajectories”, Proc. Computer Vision Pattern Recognition (CVPR), IEEE, pp. 2195-2202, 2006.

[60] L.Luo, F.Wu, S.Li, Z.Zhuang and Y.Zhang, “A Two-Pass Optimal Motion-Threading Technique for 3D Wavelet Video Coding,” IEEE International Symposium on Circuits and Systems, vol.4, pp. 819-822, 2002.

[61] M. Ristivojevic and J. Konrad, “Space-time image sequence analysis:Object tunnels and occlusion volumes,” IEEE Trans. Image Process., vol. 15, pp. 364–376, 2006.

[62] C.Bregler, A.Hertzmann, and H.Biermann, “Recovering non-rigid 3D shape from image streams,” CVPR, 2000.

[63] I.Akhter, Y.Sheikh, S.Khan and T.Kanade, « Nonrigid Structure from Motion in Trajectory Space », NIPS, 2008.

[64] Tron, R., and Vidal, R., “A benchmark for the comparison of 3-D motion segmentation algorithms,” CVPR, pp. 1-8, 2007.

[65] S.H.Lee, O.Kwon, R.H.Park, “Weighted-Adaptive Motion-Compensated Frame Rate Up-Conversion,” IEEE Transactions on Consumer Electronics, vol. 49, no. 3, pp. 485-492, 2003.

[66] **P.Bordes**, D.Thoreau and P.Salmon, "Weighted Prediction for HEVC," Visual Information Processing and Communication III, Electronic Imaging 2012, SPIE 2012, San Jose (California-USA), Jan. 2012.

[67] M. Urvoy et al, “Motion Tubes for the Representation of Images Sequences,” IEEE International Conference on Multimedia and Expo, pp. 105-108, 2009.

[68] T. Wiegand, G.J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC Video Coding Standard," IEEE Trans. Circuits Syst. Video Technol., vol.13, 560-576 (2003).

[69] A. Leontaris and A. M. Tourapis, "Weighted Prediction methods for improved motion compensation," 16th IEEE International Conference on Image Processing (ICIP),?1029-1032 (2009).

[70] T. Wiegand, W.-J. Han, B. Bross, J.-R. Ohm, G. J. Sullivan, "Working Draft 3 of High-Efficiency Video Coding (HEVC), " JCTVC-E603, Geneva, March 16-23, 2010.

[71] **P. Bordes**, "Weighted Prediction," JCTVC-E041, Geneva, March 16-23, 2010.

[72] G. Bjontegaard, "Improvements of the BD-PSNR model," in ITU-T SG16 Q.6 Document, VCEG-AI11 (2008).

[73] K. Suhring et al., "H.264/AVC reference software," http://iphome.hhi.de/suehring/tml , JM16.2 .

[74] H. Kato and Y. Nakajima, "Weighting factor determination algorithm for H264/MPEG-4 AVC weighted prediction," IEEE Workshop on Multimedia Signal Processing (2004).

[75] F. Bossen, "Common test conditions and software reference configurations", JCTVC-E700, March (2011).

[76] F.Dufaux and J.Konrad, "Efficient, Robust and Fast Motion Estimation for Video Coding," IEEE Transactions on Image Processing, Vol.9, No3, March 2000.

[77] **P.Bordes**, "Weighted Prediction," JCTVC-F265, Torino, July 14-22, 2011.

[78] Jill Boyce, "Changes to Adaptive Reference Picture Weighting," JVT Document, JVT-E060, Geneva, October 2002.

[79] **P.Bordes** and P.Andrivon, "Improvement of Implicit Weighted Prediction," JCTVC-I0115, 9th JCTVC Meeting: Geneva, CH, 27 April 7th, May 2012.

[80] A.Dueñas, B.Olofsson, A.Ichigaya, S.Pejhan, L.Haglund, A.Luthra, P.Andrivon, **P.Bordes**, T.Jones, X.Ducloux, M.Mrak, A.Cofler, JM.Thiesse, A.Rodriguez, P.Sunna, I.Laksono, "On a 10-bit consumer-oriented profile in High Efficiency Video Coding (HEVC)," ITU-T/ISO/IEC Joint Collaborative Team on Video Coding (JCT-VC) document JCTVC-K0109, Oct. 2012.

[81] M.Zhou, "Evaluation results on IBDI," ITU-T/ISO/IEC Joint Collaborative Team on Video Coding (JCT-VC) document JCTVC-D025, Jan. 2011.

[82] A.Suhre, K.Kose, AE.Cetin, MN.Gurcan, "Content-Adaptive Colour Transform for Image Compression," Optical Engineering, SPIE Digital Library, Jan. 14, 2012.

[83] H.S.Malvar, G.J.Sullivan, "YCoCg-R: A Colour Space with RGB Reversibility and Low Dynamic Range," ITU-T/ISO/IEC Joint Video Team (JVT) document JCTVC-1014r3, July 2003.

[84] Recommendation ITU-R BT.601-5 (1995), Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios.

[85] Recommendation ITU-R BT.709-5, (04/2002), Parameter values for the HDTV standards for production and international program exchange, R-REC-BT.709-5-200204-I!!PDF-E.pdf.

[86] http://en.wikipedia.org/wiki/Eigenvalue_algorithm.

[87] B.Li, G.Sullivan, J.Xu, "Comparison of CompressionPerformance of HEVC Working Draft 5 with AVC High Profile," ITU-T/ISO/IEC Joint Collaborative Team on Video Coding (JCT-VC) document JCTVC-H0360, San Jose, CA, USA, Feb. 2012.

[88] P. Comon, "Independent Component Analysis, a new concept ?," Signal Processing, Elsevier, 36(3):287--314, April 1994, Special issue on Higher-Order Statistics.

[89] Text of ISO/IEC 13818-3-2. "MPEG-2, Test Model 5 (TM5)". Doc. ISO/IEC JTC1/SC29/WG11/93-225b. Apr. 1993.

[90] J.Xu, A.Tabatabai, K.sato, "Non-CE4: Improvement of Intra prediction based QP prediction," JCTVC-G357, 7th Meeting: Geneva, CH, 21-30 November, 2011.

[91] P.Andrivon, **P.Bordes**, P.Salmon, " Non-CE4: Spatial QP prediction based on motion vector predictors (AMVP)," JCTVC-H0289, 8th Meeting: San José, CA, USA, 1–10 February, 2012.

[92] Z. He, S.K. Mitra. "A unified rate-distortion analysis framework for transform coding". IEEE Transactions on Circuit and System for Video Technology, vol. 1, N° 12, 2001.

[93] R.H. Gweon, Y.L. Lee, "Early Termination of CU Encoding to Reduce HEVC Complexity," JCTVC-F045, 6th Meeting: Torino, IT, 14-22 July, 2011.

[94] F.X.J.Lukas & Z.L.Budrikis," Picture Quality Prediction Based on a Visual Model", IEEE Trans. on Comm. (vol.COM-30, No 7), July 1982.

[95] N.B.Nill, "A Visual Model Weighted Cosine Transform for Image compression and Quality Assessment", IEEE Trans. on communications (vol.COM-33, No.6), June 1985.

[96] K.N.Ngan, K.S.Leong and H.Singh, “Adaptive Cosine Transform Coding of Images in Perceptual Domain”, IEEE Trans. on acoustics, speech and signal processing (vol.37, No.11), Nov 1989.

[97] L.Carrioli and M.G.Albanesi, “Multigray Image Compression based on the Human Eye Transfer Function”, Alta Frequenza (vol.57, No.5), pp273-284, June 1988.

[98] S.J.Daly, “A Noise Adaptive Visual Contrast Sensibility Function for Application to Image Data Compression”, Eastman Kodak Company.

[99] D.H.Kelly, “Motion and Vision. II. Stabilized spatio-temporal threshold surface”, J.Opt.Soc.Am. (vol.69, No.10), Oct.1979.

[100] S.Daly, "Engineering observations from spatiovelocity and spatiotemporal visual models", Human Vision and Electronic Imaging III, Proceedings of SPIE (vol.3299), pp180-191, January 1998.

[101] P.G.J.Barten, “Spatio-temporal model for the contrast sensitivity of the human eye”, Dynamic Properties of Vision - VII, 1992.

[102] **P. Bordes** and P. Guillotel. “Perceptually Adapted MPEG Video Encoding”. Proceedings of the International Society for Optical Engineering vol. 3959 SPIE’2000 - Human Vision and Electronic Imaging V, San Jose, CA, USA, Jan. 2000.

[103] T.Wiegand, B.Girod. “Parameter Selection in Lagrangian Hybrid Video Coder Control”. Proceedings of the IEEE International Conference on Image Processing, 2001.

[104] T.Davies, “Resolution switching for coding efficiency and resilience,” JCTVC-F158, 6th Meeting: Turin, IT, 14-22 July 2011.

[105] T. Davies and P. Topiwala, “AHG18: Adaptive Resolution Coding (ARC),” JCTVC-G264, JCT-VC 7th Meeting, Geneva, CH, 21-30 November, 2011.

[106] T. Davies, P. Topiwala and P. Wu, “JCTVC AHG report: Resolution adaption (AHG18),” JCTVC-G018, JCT-VC 7th Meeting, Geneva, CH, 21-30 November, 2011.

[107] **P.Bordes** and P.Andrivon, “AHG18: Resolution Adaptation Coding (ARC) using single resolution in DPB,” JCTVC-H0321, 8th Meeting: San José, CA, USA, 1–10 February, 2012.

[108] Recommendation ITU-R BT.1886, “Reference electro-optical transfer function for flat panel displays used in HDTV studio production,” March 2011.

[109] SMPTE Standard, “High Dynamic Range Electro-Optical Transfer Function of Mastering Reference Displays,” ST 2084.

[110] A.Tourapis, Y.Su, D.Singer, C.Fogg, R.Vleuten, E.François, “Report on the XYZ/HDR Exploratory Experiment 1 (EE1): Electro-Optical Transfer Functions for XYZ/HDR Delivery.”

[111] T. Stockhammer , “DASH, Design Principles and Standards”, MMSys 2011.

[112] M. Karczewicz and R. Kurceren, "The SP- and SI-Frames Design for H.264/AVC," IEEE Transactions on Circuits and Systems for Video Technology, vol. 13, pp. 637-644, July 2003.

[113] E. Setton and B. Girod, "Video streaming with SP and SI frames", VCIP2005.

[114] Text of ITU-T Recommendation H.264 | ISO/IEC 14496-10 - H.264/AVC (MPEG-4 part 10) Advanced Video Coding, 2003.

[115] X. Zhou and C. Kuo, “Enhanced Video Stream Switching Schemes for H.264”, MMSP2005.

[116] A.Vetro, C. Christopoulos, and H. Sun, "Video Transcoding Architectures and Techniques: An Overview," IEEE Signal Processing Magazine, vol. 20, pp. 18-29, 2003.

[117] D. Lefol, D. Bull, N.Canagarajah and F. Rovati, "Performance Evaluation of Transcoding Algorithms for H.264," in International Conference on Consumer Electronics, 2006 ICCE'06, 2006 Digest of Technical Papers, pp. 415-416, Jan 7-11. 2006.

[118] G.J. Keesman, "Multi-Program Video Data Compression," Ph. D. Thesis, TU Delft, October 10, 1995.

[119] S. Notebaert, J. De Cock, P. Lambert and R. Van de Walle, "Requantization Transcoding for Reduced-Complexity H.264/AVC Video Coding Applications, " Proceedings of the Ninth IASTED International Conference Signal and Image Processing, August 20-22, 2007.

[120] P. Tudor and O. Werner, "Real-time transcoding of MPEG video bit streams," BBC R&D - Int. Broadcasting Convention, Amsterdam, 1997.

[121] **P.Bordes**, P.Andrivon, F.Hiron, "Colour Mapping SEI message," JCTVC-N0180, 14th Meeting: Vienna, AT, 25 July – 2 Aug. 2013.

[122] **P.Bordes**, Y.He, "Support of XYZ Colour Space - Colour Mapping SEI message," ISO/IEC JTC1/SC29/WG11 MPEG2013/ M31186, October 28 - November 01, 2013, Geneva, Switzerland.

[123] A.Cantoni, "Optimal Curve Fitting With Piecewise Linear Functions," IEEE Transactions on Computers, Vol. C-20, No1, January 1971.

[124] P.Andrivon, **P.Bordes**, E.François, "SEI message for Colour Mapping Information," JCTVC-Q0074, 17th Meeting: Valencia, ES, 27 March – 4 April 2014.

[125] Fraunhofer Heinrich Hertz Institute (HHI), KTA software jm14.2, http://iphome.hhi.de/suehring/tml .

[126] **P.Bordes**, A.Orhand, "Improved Algorithm for Fast Bit Rate Transcoding H.264/AVC," Proceedings of Networked and Electronic Media (NEM) Conference, pp. 239-242 (2008).

[127] **P.Bordes** and S.Cherigui, "Adaptation of H.264/AVC Predictions for enabling Fast Transrating," Electronic Imaging 2010, SPIE 2010, San Jose (California-USA), Jan. 2010.

[128] M.Stokes, M.Anderson, S. Chandrasekar, R.Motta, " A Standard Default Colour Space for the Internet – sRGB," W3C CSS3 Colour specification, Version 1.10, November 5, 1996.

[129] JSVM (Joint Scalable Video Model) software: reference software for the Scalable Video Coding (SVC) project of the Joint Video Team (JVT) of the ISO/IEC Moving Pictures Experts Group (MPEG) and the ITU-T Video Coding Experts Group (VCEG), http://ip.hhi.de/imagecom_G1/savce/downloads/SVC-Reference-Software.htm.

[130] Digital Cinema Initiatives 2014 StEM Access Procedures http://www.dcimovies.com/2014_StEM_Access .

[131] ITU-R SG6/W6-C group, Working Party 6C (WP 6C) - Programme production and quality assessment , http://www.itu.int/en/ITU-R/study-groups/rsg6/rwp6c/Pages/default.aspx.

[132] W.Husak, E.François, A.Luthra, " AHG on Support of HDR XYZ Colour Space and HDR," ISO/IEC JTC1/SC29/WG11 M34603, Strasbourg, FR, October 2014.

[133] J. Boyce, J. Chen, Y. Chen, D. Flynn, M. M. Hannuksela, M. Naccari, C. Rosewarne, K. Sharman, J. Sole, G. J. Sullivan, T. Suzuki, G. Tech, Y.-K. Wang, K. Wegner, Y. Ye (editors), "Edition 2 Draft Text of High Efficiency Video Coding (HEVC), Including Format Range (REext), Scalability (SHVC), and Multi-View (MV-HEVC) Extensions," JCTVC-R013, 18th JCT-VC Meeting, Sapporo, JP, 30 June – 9 July 2014.

[134] P.Lopez, **P.Bordes**, F.Hiron, "Flexible Stream Switching based on HEVC," 30th Picture Coding Symposium (PCS-2013), Dec. 8-11, 2013, San Jose, California.

[135] **P.Bordes**, P.Andrivon, "Content-Adaptive Colour Transform For HEVC," 30th Picture Coding Symposium (PCS-2013), Dec. 8-11, 2013, San Jose, California.

[136] F.Bossen, B.Bross, K.Sühring, D.Flynn, "HEVC Complexity and Implementation Analysis, IEEE Transactions on Circuits and Systems for Video Technology, vol.22, no.12, Dec.2012.

[137] C.Poynton, "Digital Video and HDTV – Algorithms and Interfaces," San Francisco: Morgan Kaufmann Publishers, 2003.

[138] C.Tomasi and T.Kanade, "Detection and Tracking of Point Features." Carnegie Mellon University Technical Report CMU-CS-91-132, April 1991.

[139] Egon Verharen, "UHD Developments in Europe," IPTV Seminar 2014 10-11 November 2014, http://iptvseminar.com/documents/2014/11/egon-verharen-npo.pdf .

[140] Initial Report of the UHDTV Ecosystem Study Group, SMPTE, 2013.

Glossary

AHG Ad Hoc Group (MPEG standardization).

ARC Adaptive Resolution Coding.

AVC Advanced Video Coding (a.k.a. H.264 or MPEG-4 part 10).

BDA Blu-ray Disc Association.

CIE Internal Commission on Illumination standard**.**

CGS Colour Gamut Scalability.

CRA Clear Random Access.

DCT Discrete Cosinus Transform.

DPB Decoded Pictures Buffer.

DST Discrete Sinus Transform.

DVB Digital Video Broadcasting.

EOTF Electro-Optical Transfer Function.

HEVC High Efficiency Video Coding (a.k.a H.265).

HVS Human Vision System.

ICA Independent Component Analysis.

ILP Inter-Layer Prediction (SHVC).

ILR Inter-Layer Reference picture (SHVC).

JCT-VC Joint Coding Team on Video Coding.

LSM Least Square Minimization.

MPEG Motion Pictures Expert Group.

ODT Targeted Output Display. In ACES, the ODT is the reference end device screen/display considered in the content creation workflow.

OETF Optical-Electro Transfer Function.

OTT Over The Top. OTT refers to delivery of audio, video, and other media over the Internet without the involvement of a multiple-system operator in the control or distribution of the content.

PCA Principal Component Analysis.

RAP Random Access Point.

RDO Rate-Distortion Optimization: encoding choices (parameters, modes…) is based on trade-off (optimization) considerations between minimizing bit-rate and distortion (or maximizing quality).

SAD Sum of Absolute Differences.

SHVC Scalable High Efficiency Video Coding (HEVC extension).

SMPTE Society of Motion Pictures & Television Engineers.

VOD Video On Demand.

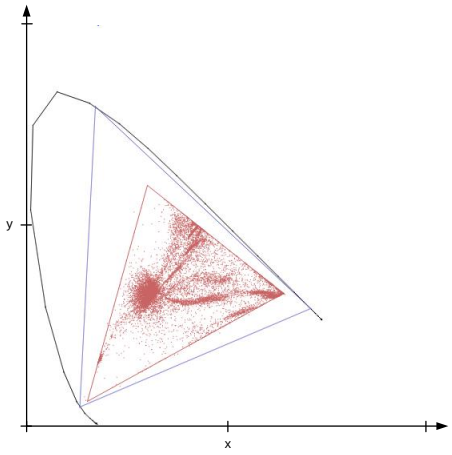
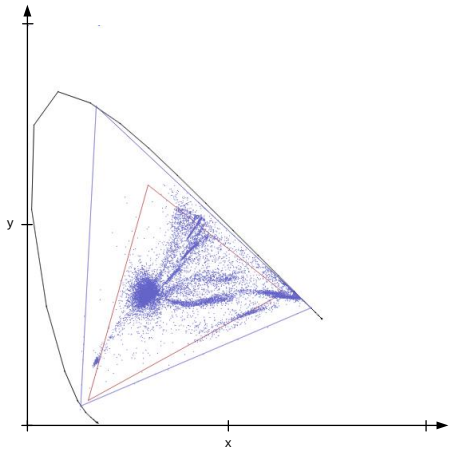
VQEG Video Quality Expert Group.

10. Annexes

10.1 The list of Wide Colour Gamut test sequences (chapter 8.2.3)

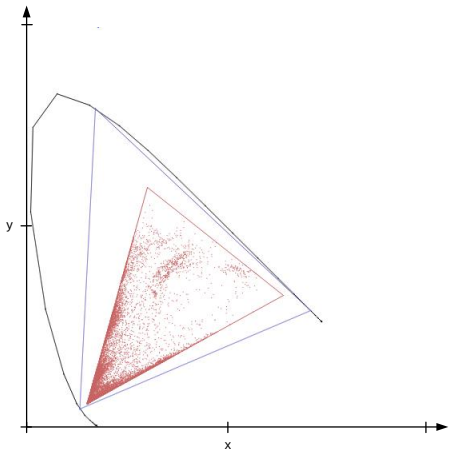
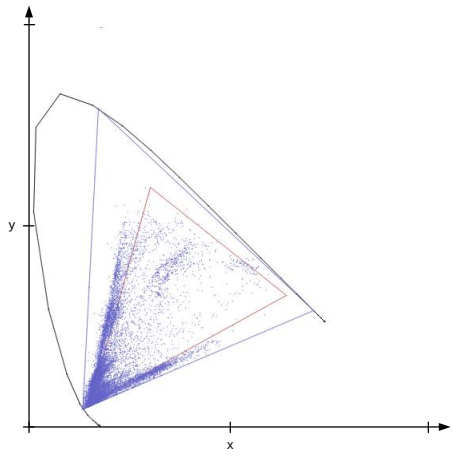
The colour samples projected into the CIE-xy space, for the rec.709 and the rec.2020 colour graded versions respectively, with the colour space primaries triangle. One can figure out the how the rec.2020 video contains wide colour gamut colours out of the regular rec.709 colour space.

The "*Birthday*" sequence contains saturated colours objects and clothes, with people moving, Camera motion and zoom-in. The background is made of textured black and white drawings with high spatial frequencies.

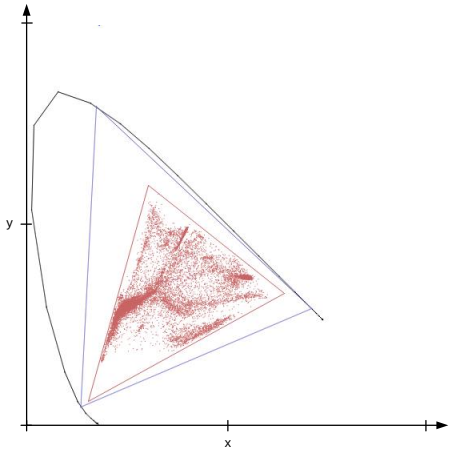
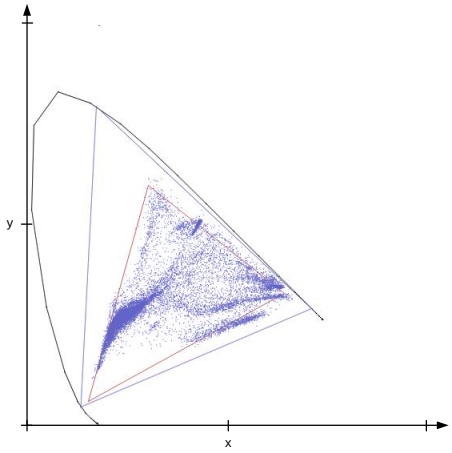
Birthday	
Rec.709	**Rec.2020 (DCI-P3 grade)**
	
	

The "*BirthdayFlash*" scene was lighted with mono chromatic LED projectors red, blue and green, corresponding to colours out of the Rec.709 colour space. The light condition was intentionally changed dynamically, from red to blue, next green, next mixing blue and red, etc… Then the colour characteristics are evolving a lot along the scene duration. The displayed frame above corresponds to the middle of the sequence, where blue light is dominant.

For the MPEG experiments, the sequences duration was limited to 5 seconds to reduce the amount of tests computation. Then "*BirthdayFlash*" was split in two parts: "*BirthdayFlashPart1*" and "*BirthdayFlashPart2*".

<table>
<tr><th colspan="2">BirthdayFlash</th></tr>
<tr><th>Rec.709</th><th>Rec.2020 (DCI-P3 grade)</th></tr>
<tr><td>  </td><td>  </td></tr>
<tr><td> 
</td><td> 
</td></tr>
</table>

The "*TableCar*" video is a fixed shot where only a radio-controlled car is moving fast on the table, rushing the coloured accessories. The artistic choices made by the colourist clearly lead to different hue in the final rendering.

TableCar	
Rec.709	**Rec.2020 (DCI-P3 grade)**
	
	

Parakeets is the only outdoor sequence. It was shot in Rennes' Thabor Park through the wire mesh fence of an aviary. The birthday feathers and the orange fruit appear more saturated on the Rec.2020 version than on the Rec.709.

Parakeets	
Rec.709	**Rec.2020 (DCI-P3 grade)**