Temporal and inter-view interpolation for the improvement of the side information in distributed video coding

Giovanni Petrazzuoli

2013-ENST-0004

EDITE – ED 130

Doctorat ParisTech

T H È S E

pour obtenir le grade de docteur délivré par

Télécom ParisTech

Spécialité Signal et Images

présentée et soutenue publiquement par

Giovanni PETRAZZUOLI

le 14 janvier 2013

Interpolation temporelle et inter-vues pour l'amélioration de l'information adjacente dans le codage vidéo distribué

—

Temporal and inter-view interpolation for the improvement of the side information in distributed video coding.

Directeurs de thèse : **Béatrice PESQUET-POPESCU**
Marco CAGNAZZO et **Giovanni POGGI**

Jury

Stefano TUBARO, Professeur, Politecnico di Milano, Milan	Président
Christine GUILLEMOT, Directeur de Recherche INRIA, IRISA, Rennes	Rapporteur
Vladimir STANKOVIC, Professeur, University of Strathclyde, Glasgow	Rapporteur
Adrian MUNTEANU, Professeur, Vrije Universiteit Brussel, Bruxelles	Examinateur
Frederic DUFAUX, Directeur de Recherche CNRS, TELECOM ParisTech, Paris	Examinateur

THÈSE

Télécom ParisTech
Grande école de l'Institut Télécom - membre fondateur de ParisTech
46 rue Barrault — 75634 Paris Cedex 13 — Tél. +33 (0)1 45 81 77 77 — www.telecom-paristech.fr

Résumé

Le codage de source distribué est un paradigme qui consiste à encoder indépendamment deux sources corrélées et à les décoder conjointement. Wyner et Ziv ont montré que le codage de source distribué peut atteindre les mêmes performances débit-distorsion que le codage de source conjoint, pourvu que certaines contraintes soient satisfaites. Cette caractéristique rend le codage de source distribué très attractif pour des applications qui demandent un encodeur à faible complexité ou pour ne pas être obligé à avoir des communications entre les sources. Dans le cadre du codage vidéo distribué, les trames corrélées sont encodées séparément et décodées conjointement. Dans l'architecture ainsi dite de Stanford, le flux vidéo est séparée en trames clés et Wyner-Ziv. Les trames clés sont encodées INTRA. Les trames Wyner-Ziv sont données en entrée à un codeur de canal systématique ; seulement les bits de parité sont envoyés. Au décodeur, on produit une estimation de la trame Wyner-Ziv, appelée information adjacente, en interpolant les trames clés reçues. L'information adjacente, considérée comme une version bruitée de la trame Wyner-Ziv, est corrigées par les bits de parité. Dans cette thèse, nous avons proposé plusieurs algorithmes pour la génération de l'information adjacente et pour l'interpolation temporelle et inter-vue. On a aussi proposé des algorithmes de fusion bayésienne des deux interpolations. Tous les algorithmes proposés donnent des résultats meilleurs par rapport à l'état de l'art en termes de performance débit-distorsion. Nous avons proposé plusieurs algorithmes pour l'estimation de la trame Wyner-Ziv dans le cadre de la vidéo multi-vues plus profondeur.

Abstract

Distributed source coding is a paradigm that consists in encoding two correlated sources independently, provided that they are decoded jointly. Wyner and Ziv proved that distributed source coding can attain the same rate distortion performance of joint coding, under some constraints. This feature makes distributed source coding very attractive for applications that require a low-complexity encoder (such as for wireless sensor networks) or for avoiding communication between the sources (such as in multiview video systems). In distributed video coding, correlated frames are encoded separately but decoded jointly. In the Stanford Architecture, the video is split into Key Frames and Wyner-Ziv Frames. Only the Key Frames are sent to the decoder. The Wyner-Ziv Frames are fed into a systematic channel coder and the parity bits are sent to the decoder. At the decoder side, an estimation of the Wyner-Ziv Frame, called side information, is produced by interpolating the frames available at the decoder side. The side information, that can be considered as a noisy version of the real Wyner-Ziv Frame, is corrected by the parity bits sent by the encoder. In this thesis, we study several algorithms for side information generation in the context of distributed multiview video coding both for the temporal and inter-view interpolation. All our algorithms outperform the state-of-the-art in terms of rate distortion performance. A second contribution is in the context of the multiview video plus depth (MVD) format. In the interactive multiview video streaming paradigm, distributed video coding assures the continuity of the playback during the streaming, since the Wyner-Ziv Frames can be decoded independently from which reference frames are available at the decoder side. Several algorithms for Wyner-Ziv estimation in this context have been proposed.

Résumé en français

Introduction

Les signaux vidéos ont une grande redondance dans le temps et dans l'espace et cette corrélation est exploitée dans les architectures vidéo classiques (MPEG-x, H.263 and H.264/AVC) à l'encodeur: concernant la corrélation temporelle, trames consécutives sont encodées conjointement (par une prédiction temporelle compensée en mouvement). Sinon la redondance spatiale est exploitée par un codage par transformée. L'effort computationnel pour éliminer la corrélation temporelle (via estimation et compensation du mouvement) est très fort. En plus, la mémoire requise pour stocker les trames précédentes utilisées comme références est une requête qui peut être difficile à satisfaire, en particulier dans des dispositifs portables. Comme conséquence, les architectures de compression vidéo classiques présentent une asymétrie entre l'encodeur et le décodeur: l'encodeur est plus complexe que le décodeur, en étant ce dernier un sous-ensemble du premier. Ça est envisagé pour plusieurs applications, comme la diffusion et les communications un-à-plusieurs. En fait, dans ces scénarios, la vidéo est encodée une seul fois, mais elle est décodée plusieurs fois. D'autre part, il y a des applications, comme les réseaux de senseurs sans fil, qui ont une capacité limitée en termes de mémoire et batteries. Pour ces cas-là, les architectures standard ne sont pas convenables et des nouveaux approches doivent être envisagés. Dans les années 1970, un nouveau travail théorique est apparu : le codage vidéo distribué. Dans ce paradigme, deux sources sont encodés séparément, mais décodés conjointement. Slepian et Wolf ont montré que, sous des faibles contraintes, cela ne perturbe pas les performances réalisables (au moins asymptotiquement) d'un codage sans perte. Après, Wyner et Ziv ont étendu ce résultat dans la cas de codage avec pertes: si on arrive à satisfaire certaines conditions, il n'y pas de pertes en termes de débit-distorsion entre le codage de source distribué et le codage conjoint. Dans les années 2000, le codage vidéo distribué (CVD) prend inspiration du paradigme du codage de source distribué pour supporter nouvelles architectures, comme les senseurs sans fil ou les réseaux de senseurs avec une capacité limitée en termes de complexité et de mémoire, qui ne peut pas supporter l'estimation du mouvement à l'encodeur. La caractéristique principale du codage de source distribué est que les sources ne nécessitent pas de communiquer entre eux. Ça est convenable pour éviter tout sorte de communication entre les cameras dans les systèmes multi-vues. En fait, les

vidéos multi-vues présentent une grande corrélation inter-vues. Si elles sont encodées avec une technique standard, comme H.264/AVC, cette dépendance est exploitée à l'encodeur de façon similaire à la corrélation temporelle pour les vidéos mono-vue. Donc, une communication entre les cameras est nécessaire pour compresser la vidéo avec H.264/MVC. D'autre part, puisque le codage vidéo distribué peut encoder les différentes vues séparément, la communication entre les cameras peut être évitée. Dans cette thèse, on a travaillé sur des problèmes et applications concernant les systèmes CVD. Plus précisément, nous utilisons comme référence uns de plus connues architecture pour le CVD, le codeur de Stanford. Dans ce codeur, le flux vidéo est divisée en Trames clés (TC) et Trames Wyner-Ziv (TWZ). Comme dans la terminologie du codage vidéo prédictif, une TC et les TWZ suivants forment un *group of pictures* (GOP). La distance entre deux successives TCs est appelée taille du GOP. Les TCs sont codées INTRA (c'est-à-dire sans estimation et compensation du mouvement). Les TWZs sont données comme entrée à un codeur de canal systématique. La partie systématique est écartée et les bits de parité sont envoyés au décodeur. Au décodeur, on a besoin d'une estimation de la TWZ. Elle peut être obtenue par interpolation de trames déjà disponibles au décodeur. Cette estimation est appelée information adjacente (IA) et peut être considérée comme une version bruitée de la vraie trame WZ. Le décodeur de canal doit corriger ces erreurs d'estimation en utilisant les bits de parité. Le projet européen DISCOVER implémente l'architecture de Stanford et définit les outils nécessaires pour coder les TC et les TWZ et pour générer l'information adjacente. Il est devenu la technique de référence pour le codage vidéo distribué mono-vue et multi-vues. La première partie de cette thèse est concentrée sur la génération de l'information adjacente temporelle. La technique présentée dans le projet DISCOVER consiste dans une interpolation linéaire de la trajectoire des objets entre les trames disponibles les plus proches (en avant et en arrière). Pour améliorer l'algorithme d'interpolation de DISCOVER, on propose plusieurs techniques dans le domaine temporel et dans les vues. Dans le domaine temporel, puisque l'algorithme d'interpolation de trajectoire de DISCOVER utilise deux trames, on peut considérer seulement des modèle de mouvement à vitesse constante. Nous proposons d'utiliser quatre trames pour l'interpolation, pour prendre en compte aussi des modèles de mouvement plus complexes. On appelle cette technique high order motion interpolation (HOMI) - interpolation du mouvement d'ordre supérieur). On a proposé une version simplifiée d'HOMI (FastHOMI), avec une extension pour utiliser des champs de mouvement plus denses. En plus, dans le domaine temporel, quand la taille du GOP est 4 ou 8, si on utilise le codec de DISCOVER, on remarque que pas toutes les trames Wyner-Ziv ont la même qualité, due à la différente distance entre les trames qu'on utilise pour l'interpolation. Donc, on a proposé que, quand toutes les trames WZ du GOP ont été décodées, ces trames peuvent être utilisé pour ré-estimer les trames WZF de basse qualité. La technique d'interpolation du mouvement de DISCOVER a été utilisé aussi dans le domaine des vues, dans le cas des caméra alignées. A partir d'une analyse géométrique des systèmes multi-vues, la disparité d'un objet (c'est-à-dire son dé-

placement d'une vue à l'autre) est inversement proportionnelle à sa profondeur (c'est-à-dire la distance depuis le plan de l'image de la camera). On propose de donner priorité aux larges valeurs de disparité pendant la phase d'interpolation, parce que les objets associés à ces valeurs sont plus proches de la camera. Cet algorithme est appelé Interpolation with priority to large disparity (interpolation avec priorité aux grandes valeurs de disparité) - IPLD. Et HOMI et IPLD améliore les performances débit-distorsion par rapport au codeur de DISCOVER. Dans le codage multi-vue distribué, c'est possible de fondre l'estimation de la TWZ dans le domaine du temps et celle-là dans le domaine des vues. On propose, donc, pendant le procès de fusion, d'exclure l'estimation inter-vue dès régions d'occlusion. En plus, on a observé que pour quelques séquences, l'une des deux interpolations est beaucoup mieux que l'autre et que, pour telles séquences, la fusion n'améliore pas la qualité de l'information adjacente. Dans ce cas-là, ce serait mieux de donner aux décodeur de canal directement la meilleure information adjacente. Donc, on propose un algorithme qui sélectionne l'information adjacente entre l'estimation temporelle, inter-vue et la fusion, avec une méthode basée sur le débit du code de canal pour corriger l'estimation de la TWZ. La technique proposée pour la génération a trouvé une application aussi dans le cadre du codage vidéo robuste par descriptions multiples. Dans la dernière partie de cette thèse, on se focalise sur le format vidéo multi-vues plus profondeur. Dans ce nouveau format, chaque caméra produit pour chaque instant une image de texture et une carte de profondeur. D'autre part, si le CVD est appliqué au vidéo de texture et carte de profondeur, ce problème peut être résolu, parce que les TWZs peuvent être reconstruite indépendamment de comment on a généré l'information adjacente. On propose différents algorithmes pour l'estimation de la TWZ en correspondance d'une commutation de vue. Cette estimation peut être aidée aussi par l'information sur la carte de profondeur. On propose aussi d'autres techniques pour l'estimation de la profondeur WZ en utilisant l'information sur la correspondante image de texture. Enfin, on a proposé aussi un nouveau codeur vidéo distribué pour le format vidéo multi-vues plus profondeur. La profondeur de la caméra centrale peut être utilisée pour générer les autres vues, sauf que pour les régions d'occlusion. Pour les autres vues, c'est nécessaire d'envoyer seulement les régions d'occlusion. Dans le paradigme de codage de source distribué, les sources ne peuvent pas communiquer l'une l'autre. Comme conséquence, pour chaque vue dont seulement les régions d'occlusion sont envoyées, nous devons estimer indépendamment leur carte d'occlusion avec l'aide de la carte de profondeur. La structure du manuscrit est la suivante :

- Chapitre 1 - Video Coding (Codage vidéo) - Dans ce chapitre, on rappelle les concepts de base de la compression vidéo: quantification, codage par transformée, estimation du mouvement. On illustre aussi l'architecture des codeur standard comme MPEG-2 et H.264/AVC.
- Chapitre 2 - Multiview Video (Vidéo multi-vues) - Dans ce chapitre, on décrit les systèmes vidéo multi-vues, en partant du modèle de la caméra et quelque concept de

base de la géométrie 3D. On montre des algorithmes pour l'estimation de la disparité et les architecture pour la compression vidéo multi-vues. Enfin, on introduit le nouveau format du "multi view video plus depth" avec un état-de-l'art pour le codage des cartes de profondeur.

- Chapitre 3 - Distributed Video Coding (Codage vidéo distribué) - Dans ce chapitre, on introduit le paradigme de source distribué and comment il est implémenté dans le contexte du vidéo mono-vue. On va décrire l'architecture de Stanford et le codeur de DISCOVER on se concentrant sur la génération de l'information adjacente. On donne un état de l'art détaillé pour ce problème. Enfin, on montre comment le codeur de DISCOVER va être étendu dans le contexte du vidéo multi-vues. On montre les différents configurations de cameras normalement proposées dans la littérature and on donne un état de l'art détaillé pour la construction de l'information adjacente pour interpolation inter-vues.

- Chapitre 4 - Proposed Methods for side information construction (Méthodes proposées pour la construction de l'information adjacente) - Dans ce chapitre, on illustre les algorithmes proposés pour la génération de l'information adjacente qui améliore les performances du codeur de DISCOVER dans le contexte de la vidéo mono-vue et multi-vues. On montre les algorithmes HOMI et IPLD et les techniques pour la fusion.

- Chapitre 5 - Application of Distributed Video Coding to Multiview Video plus Depth MVD (Application du codage vidéo distribué au format vidéo multi-vues plus profondeur) - Dans ce chapitre, on propose différents solutions pour l'estimation de la TWZ quand une commutation de vue tombe en correspondance d'une TWZ. On présente aussi un nouveau algorithm pour la génération de l'information adjacente pour les cartes de profondeur. Enfin, on montre un nouveau codeur distribué pour le format MVD.

- Appendix - Dans cet annexe, on liste toutes les résultats sur la génération de la SI of HOMI (dans la cas temporel et inter-vue)

État de l'art sur le codage vidéo distribué

Une source X, qu'on peut modélisée comme une variable aléatoire discrète, selon le premier théorème de Shannon [CT91] peut être encodée et décodée sans perte d'information si le débit utilisé pour l'encoder est plus grand ou égal à sa entropie $H(X)$.

Deux source dépendants, modélisées comme deux variable aléatoires X et Y, peuvent être encodées et décodées conjointement sans perte d'information, si le débit total pour les encoder est plus grand ou égal à leur entropie conjointe $H(X, Y)$.

Quand on encode deux sources corrélées séparément et les décode conjointement, on parle de codage de source distribué.

Sinon, Wyner et Ziv ont donné une version avec perte du schéma de Slepian et Wolf. Ils ont montré que si les deux sources sont conjointement gaussiennes et la distorsion est calculée comme l'erreur quadratique moyenne, alors le codage de source distribué peut atteindre les mêmes performances débit-distorsion du codage de source conjointe.

Dans le paradigme du codage vidéo distribué, la complexité computationelle est déplacée de l'encodeur au décodeur. Pour le codage vidéo ça est très important, surtout pour des applications qui demandenr un encodeur à bas complexité, comme senseurs vidéo sans fil pour la vidéo-surveillance, cameras pour PC sans fil et dans le cadre du codage vidéo multi-vues, où la communication entre les cameras doit être évitée.

Selon le théorème de Wyner-Ziv, trames corrélées d'une sequence vidéo peuvent être quantifiées et codées indépendamment avec une perte minimale en termes de performance débit-distorsion, à condition qu'elles soient codées conjointement. Ça, en principe, va éviter l'estimation du mouvement à l'encodeur et, donc, on a un déplacement de la complexité de l'encodeur au décodeur.

Dans la littérature, deux contributions principales existent pour le CVD: l'architecture de Stanford [AZG02] et l'architecture PRISM [PMR07]. Dans l'architecture proposées par [AZG02], appelée Stanford, la vidéo est structurée en *group de pictures* (GOP), où certaines trames (pour example une chaque N), appelées trames clés (TC), sont codées INTRA (normalement en utilisant le codeur de H.264/INTRA) et les autres sont codés Wyner-Ziv et sont appelées trames Wyner-Ziv (TWZ). Chaque TWZ est encodées indépendamment des autres: elles sont transformées DCT, quantifiées et données comme entrées à un codeur systématique de canal. A la sortie on a les bits d'information et les bits de parité. Seulement les dernières sont stockés dans un buffer. Les bits d'information sont éliminés. Initialement, l'encodeur envoie seulement un sous-ensemble de bits de parité, sous une requête du décodeur. Le nombre de bits nécessaire est estimé au décodeur: cette information est envoyée à l'encodeur via un canal de rétroaction. Le décodeur génère l'information adjacente à travers une interpolation temporelles des trames clés. Si la probabilité d'erreur en sortie du décodeur de canal excède une certaine seuil (normalement 10^{-3}), on demande plus de bits de parité à l'encodeur via une boucle de retour. On estime l'erreur quadratique moyenne sur les valeurs quantifiées, étant donnée l'indice de quantific-

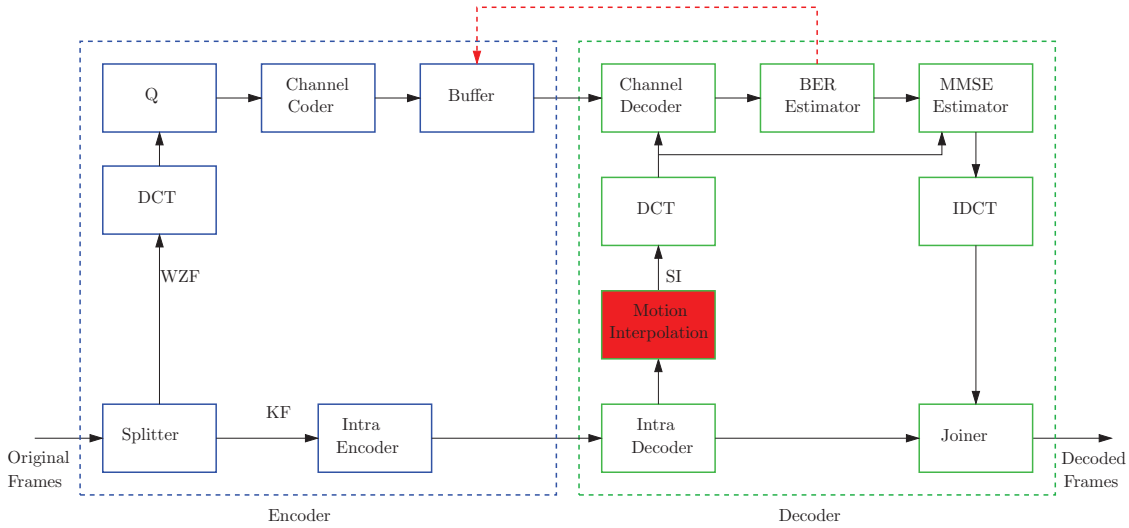


Figure 1: Le codeur de Stanford

ation et l'information adjacente. Après on applique la transformation inverse et on obtient l'image reconstruite. Le schéma complet est montré en Fig. 1.

État de l'art sur la construction de l'information adjacente

Comme on a déjà dit, le décodeur doit trouver une prédiction pour la TWZ. Cette estimation doit être corrigée par les bits de parité. Elle peut être obtenue par interpolation des trames adjacentes. Pour simplicité on suppose que que la taille du GOP (la distance entre deux trames clés) est égale à 2. Donc, les trames qui sont interpolées sont les deux trames clés adjacentes. Soient I_{t-1} and I_{t+1} les trames clés et soit I_t la trame qui doit être estimée. La plus populaire technique pour la construction de l'information adjacente est celle proposée dans le projet DISCOVER. Cette méthode consiste dans les passages suivantes (voir aussi la Figure 2) :

1. **Filtrage passe bas.** Les trames I_{t-1} et I_{t+1} sont filtrées passe-bas pour réduire le bruit.

2. **Estimation du mouvement en avant.** On estime le mouvement entre I_{t+1} et I_{t-1}. Soit $\mathbf{v}$ le champ de vecteurs de mouvement obtenu.

3. **Scission des vecteurs de mouvement.** Pour chaque bloc de la trame I_t centré en $\mathbf{p}$, $B_t^{\mathbf{p}}$, on cherche le vecteur qui intersectionne la trame t dans le point le plus proche à $\mathbf{p}$. Soit $\mathbf{q}^*(\mathbf{p})$ ce point. Il doit satisfaire la relation suivante :

$$\mathbf{q}^*(\mathbf{p}) = \arg \min_{\mathbf{q} \in \mathcal{N}(\mathbf{p})} \left\| \mathbf{q} + \frac{1}{2}\mathbf{v}(\mathbf{q}) - \mathbf{p} \right\|^2 \tag{1}$$

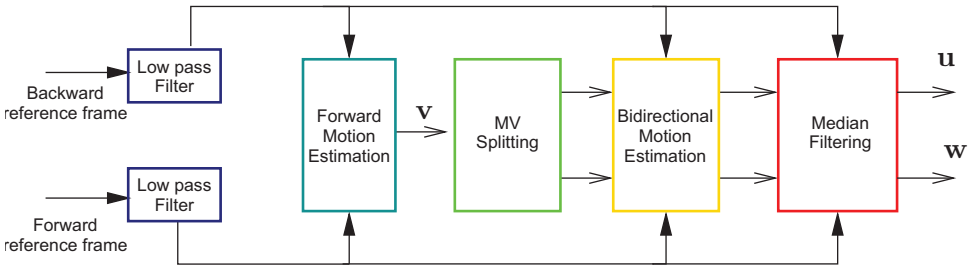


Figure 2: Le diagram à bloc de l'algotihme d'interpolation du mouvement de DISCOVER

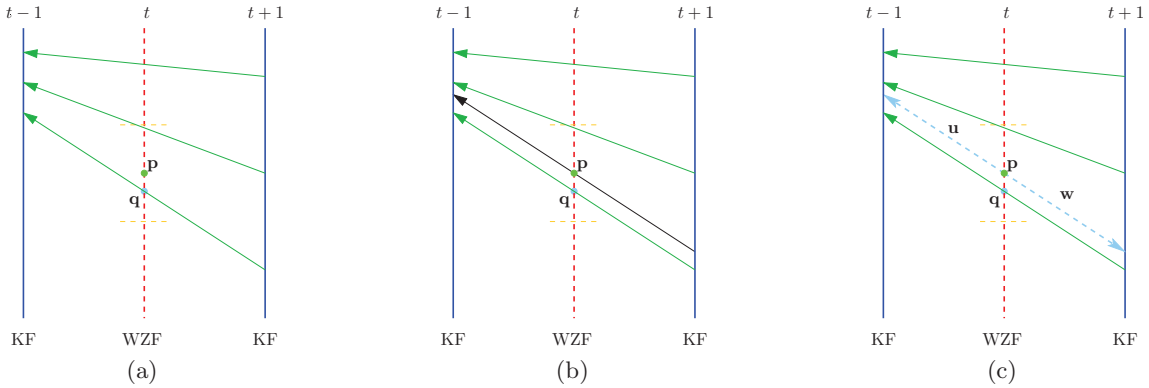


Figure 3: Les différents passage pour la méthode d'interpolation de DISCOVER. Le but est d'estimer le bloc centré en $\mathbf{p}$. Soit $\mathbf{q}$ le point d'intersection le plus proche entre le champs de mouvement estimé et la TWZ I_t (a). Le vecteur en $\mathbf{q}$ est déplacé afin qu'il passe en $\mathbf{p}$ (b). Après le vecteur est scissionné (c).

où $\mathcal{N}(\mathbf{p})$ est l'ensemble des centres des blocs voisins à $B_t^{\mathbf{p}}$. Après ce vecteur est scissionné en $\mathbf{u} = \mathbf{v}/2$ et $\mathbf{w} = -\mathbf{v}/2$ et puis ils sont centrés en $\mathbf{p}$. Ce processus est montré en Figure 3.

4. **Estimation du mouvement bidirectionnelle.** Les vecteurs $\mathbf{u}$ et $\mathbf{w}$ sont raffinés autour des leurs positions.

5. **Filtrage médian des vecteurs de mouvement.** Un filtre médian est appliqué aux deux champs de vecteurs de mouvement, pour éviter toute sorte d'incohérence spatiale.

L'information adjacente est finalement obtenue comme moyenne de l'image précédente et suivante compensée par les deux champs de vecteur de mouvement.

Maintenant, on considère les cas où la taille du GOP est plus grand que deux mais elle est une puissance de deux (c'est-à-dire : 4, 8, etc.) [ASG03]. Par exemple, si la taille du GOP est égale à 4, le champs de vecteur de mouvement est calculé comme :

1. La TWZ à l'instant t est estimée en utilisant les TWZs I_{t-2} et I_{t+2} et puis elle est décodée, comme en Fig. 4(a).

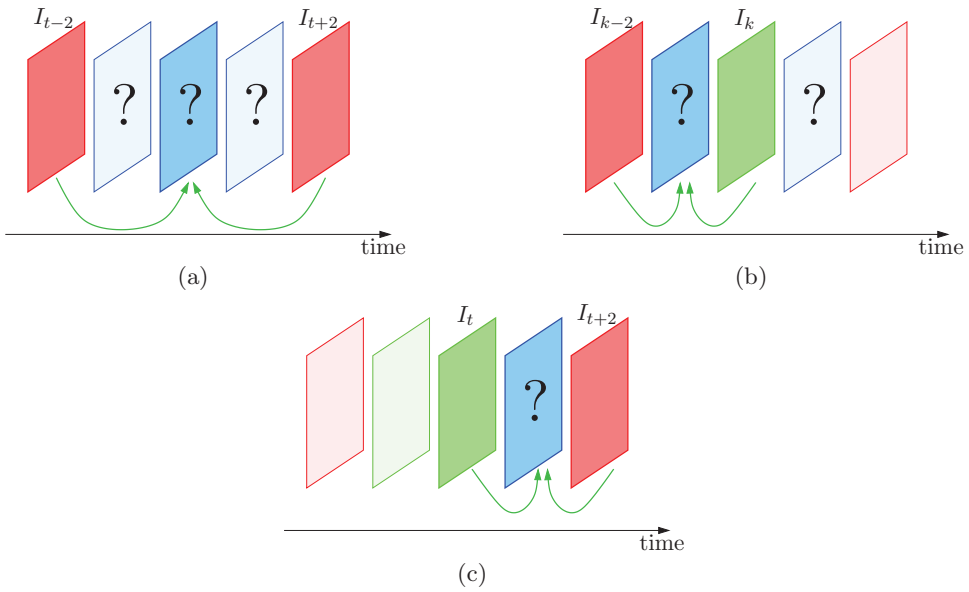


Figure 4: Estimation de la TWZ pour taille du GOP égale à 4 (les TCs sont en rouge, les TWZs sont en bleue, les TWZs déjà décodées sont en vert: on estime I_t (a)), après à partir d'elle, on estime I_{t-1} (b) et I_{t+1} (c)

2. La TWZ à l'instant $t-1$ est estimée en utilisant la TC I_{t-2} et la trame décodée TWZ I_t (Fig. 4(b)) et celle à l'instant $t+1$ est estimée en utilisant la TC I_{t+2} et la TWZ décodée I_t (comme en Fig. 4(c))

Costruction de l'information adjacente pour le codage vidéo multi-vues distribué

Maintenant, on va montrer comment le codage vidéo distribué et en particulier l'estimation de l'information adjacente peut être entendue au vidéo multi-vues.

Quand on parle de systèmes multi-vues, pour se référer à une trame, on a besoin d'un deuxième index pour le point de vue. On appelle $I_{t,k}$ l'image à l'instant t de la caméra k. L'information adjacente est indiquée avec un chapeau. Dans les systèmes CVD multi-vues, trois types de caméra sont normalement considérées: caméras clés pures (pour lesquelles toutes les trames sont TCs); caméras Wyner-Ziv pures (toutes les trames sont TWZ); et caméras hybrides pour lesquelles une trames sur deux est une TWZ et les autres sont TCs. Ces caméras peuvent être arrangées dans plusieurs configurations. En littérature, normalement trois schémas ont été considérées [MPP08]:

- Schéma asymétrique (voir Fig. 5(a)): caméras clés pures et cameras Wyner-Ziv pures sont alternées.
- Schéma hybride 1/2 (voir Fig. 5(b)): caméras clés pures et caméras hybrides sont alternées.

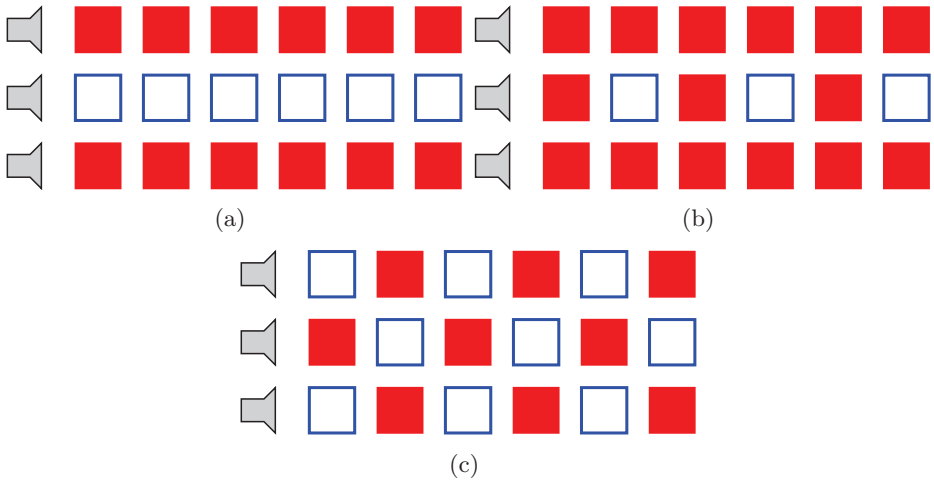


Figure 5: Les trois principales configurations pour la vidéo multi-vues : schéma asymétrique (a), Schéma hybride 1/2 (b), Schéma symétrique 1/2 (c). Les trames remplies sont TCs et les trames vides sont TWZs.

- Schéma symétrique 1/2 (voir Fig. 5(c)): toutes les caméras sont hybrides et les TCs et TWZs sont placées dans une grille quincunx.

Pour toutes les schémas, pour une TWZ $I_{t,k}$, deux TCs sont toujours disponibles pour l'interpolation (sur l'axes temporel ou des vues). Donc, à l'exception du premier schéma, deux autres trames clés sont disponibles. A priori, toutes les quatre images peuvent être utilisées pour générer l'information adjacente. Toutefois, techniques différentes pour la construction de l'information adjacente peuvent être considérées selon les différentes caractéristiques du mouvement et de la disparité.

Selon la disposition des TCs et TWZs, les trames Wyner-Ziv peuvent être estimée avec différentes méthodes:

- **estimation temporelle**: les trames adjacentes dans l'axes temporel dans la même vue sont utilisé pour l'interpolation
- **estimation inter-vue**: les trames des vues adjacentes au même instant sont utilisées pour l'interpolation
- **une fusion**: les estimations temporelle et inter-vue sont fusionnés selon des différents critères

Dans la suite, l'image fondue est indiquée par $\widetilde{I}(\mathbf{p})$, l'estimation temporelle par $\widehat{I}_T(\mathbf{p})$ et l'estimation inter-vue comme $\widehat{I}_V(\mathbf{p})$. Les techniques pour l'interpolation temporelle peuvent être entendues au domaine des vues en replaçant les trames adjacentes dans le domaine du temps avec les trames des vues adjacentes. Ces techniques donnent des bonnes performances, seulement si certaines conditions sont satisfaites: les vues doivent être rectifiées et l'axes de vue doit être perpendiculaire à la baseline [GPT+07]. Dans la suite, on suppose que ces deux hypothèses sont satisfaites.

Quand l'interpolation temporelle et celle inter-vue sont disponibles, comme dans le cas d'un schéma symétrique 1/2, elles peuvent être combinées pour créer une seule information adjacente. Cette opération est appelé fusion.

Dans [MMCPP09a], la fusion est basées sur deux images d'erreur E_T et E_V. E_T est la différence absolue entre les deux trames de référence temporelle compensée en mouvement. Par contre, la différence absolue des trames de référence dans le domaine des vue compensée en disparité est appelée E_V. Deux techniques de fusion ont été proposées : dans la fusion binaire (Bin), la valeur dans le pixel $\mathbf{p}$ est sélectionnée dans l'estimation inter-vue si $E_V(\mathbf{p}) < E_T(\mathbf{p})$, sinon on sélectionne l'interpolation temporelle. Dans la fusion linéaire (Lin), on calcule le coefficient $\alpha = \frac{E_T(\mathbf{p})}{E_T(\mathbf{p})+E_V(\mathbf{p})}$ et l'image fondu $\widetilde{I}_{\mathrm{L}IN}(\mathbf{p})$ est définie comme une moyenne pondérée de $\widehat{I}_V(\mathbf{p})$ et $\widehat{I}_T(\mathbf{p})$, en utilisant respectivement α et $1-\alpha$ comme poids: *i.e.*

$$\widetilde{I}_{\mathrm{L}IN}(\mathbf{p}) = \alpha\widehat{I}_V(\mathbf{p}) + (1-\alpha)\widehat{I}_T(\mathbf{p}) \tag{2}$$

Malheureusement, quand l'estimation dans un domaine est beaucoup mieux que dans l'autre, la qualité de l'image fondu n'est pas meilleure que celle de la meilleure information adjacente. Donc, dans ce cas, ce serait mieux de détecter directement la meilleure information adjacente, sans faire la fusion entre ces deux-là.

Méthodes proposées

High order motion interpolation HOMI

L'algorithme pour la génération de l'information adjacente de DISCOVER consiste dans une interpolation linéaire des trajectoires pour en déduire les positions des objets dans les TWZ. Ça correspond à supposer un modèle uniforme du mouvement (sans accélération des objets) pendant la période entre deux TCs. Toutefois, quand le mouvement dans la vidéo ne suit pas ce modèle, on risque d'avoir une mauvaise estimation du mouvement. Donc on propose un méthode d'interpolation d'ordre supérieure, pour considérer modèles du mouvement différents de celui linéaire.

On suppose qu'on veut estimer la TWZ à l'instant t. Cet algorithme consiste en quatre passages : l'initialisation des champs de vecteurs de mouvement par DISCOVER sur les TCs I_{t-1} et I_{t+1}; une méthode de recherche par blocs pour chercher les positions de chaque bloc dans les trames clés I_{t-3} and I_{t+3}; l'interpolation des positions des blocs; et l'alignement des vecteurs obtenus au centre de chaque bloc. Toutes les étapes se trouvent aussi en Figure 6. On commence par considérer le bloc de la trame clé centré en $\mathbf{p}$. L'algorithme d'interpolation de DISCOVER nous donne les champs de vecteurs de mouvement $\mathbf{u}$ et $\mathbf{w}$ (respectivement celui en arrière et en avant). Donc, le bloc $B_{t-1}^{\mathbf{p}+\mathbf{u}(\mathbf{p})}$ de la trame I_{t-1} centré en $\mathbf{p}+\mathbf{u}(\mathbf{p})$ est supposé se déplacer dans la position $\mathbf{p}+\mathbf{w}(\mathbf{p})$ de la trame I_{t+1}. Après on va raffiner le mouvement en considérant aussi les trames I_{t+3} et I_{t-3}. Donc, on cherche la position du bloc $B_{t-1}^{\mathbf{p}+\mathbf{u}(\mathbf{p})}$ dans la trame I_{t-3}, en utilisant toujours une méthode de recherche par bloc. Plus précisément, on cherche pour le vecteur $\widetilde{\mathbf{u}}$ tel que la fonction suivante soit minimisée:

$$J(\widetilde{\mathbf{u}}) = \sum_{\mathbf{q}} \left| B_{t-1}^{\mathbf{p}+\mathbf{u}(\mathbf{p})}(\mathbf{q}) - B_{t-3}^{\mathbf{p}+\widetilde{\mathbf{u}}}(\mathbf{q}) \right|^n + \lambda \|\widetilde{\mathbf{u}} - 3\mathbf{u}\| \tag{3}$$

Le terme de régularisation pénalise les déviations trop grandes par rapport au modèle linéaire: quand $\lambda \to \infty$, l'algorithme proposé devient équivalent à celui de DISCOVER.

De même façon, on cherche le vecteur $\widetilde{\mathbf{w}}$ qui maximise la corrélation entre $B_{t+1}^{\mathbf{p}+\mathbf{w}(\mathbf{p})}$ et les blocs dans la trame I_{t+3}.

Le prochain passage consiste dans l'interpolation des quatre positions dans la quatre images. On interpole une fonction vectorielle de valeurs $\mathbf{p}+\widetilde{\mathbf{u}}$, $\mathbf{p}+\mathbf{u}$, $\mathbf{p}+\mathbf{w}$ and $\mathbf{p}+\widetilde{\mathbf{w}}$ respectivement aux instants $t-3$, $t-1$, $t+1$, $t+3$, pour chercher sa valeur à l'instant t, qu'on appellera $\widehat{\mathbf{p}}$. Comme conséquence, on suppose que l'objet qui s'est déplacé de la position $\mathbf{p}+\mathbf{u}$ dans I_{t-1} à $\mathbf{p}+\mathbf{w}$ dans la trame I_{t+1}, est dans la position $\widehat{\mathbf{p}}$ dans la trame I_t.

Donc, les vecteurs de mouvement associés à $\widehat{\mathbf{p}}$ sont $\mathbf{u}(\mathbf{p})+\mathbf{p}-\widehat{\mathbf{p}}$ et $\mathbf{w}(\mathbf{p})+\mathbf{p}-\widehat{\mathbf{p}}$.

Le dernier passage consiste en chercher la trajectoire qui passe pour le point le plus proche de $\mathbf{p}$ et on associe les vecteurs obtenus à la position $\mathbf{p}$, comme dans DISCOVER.

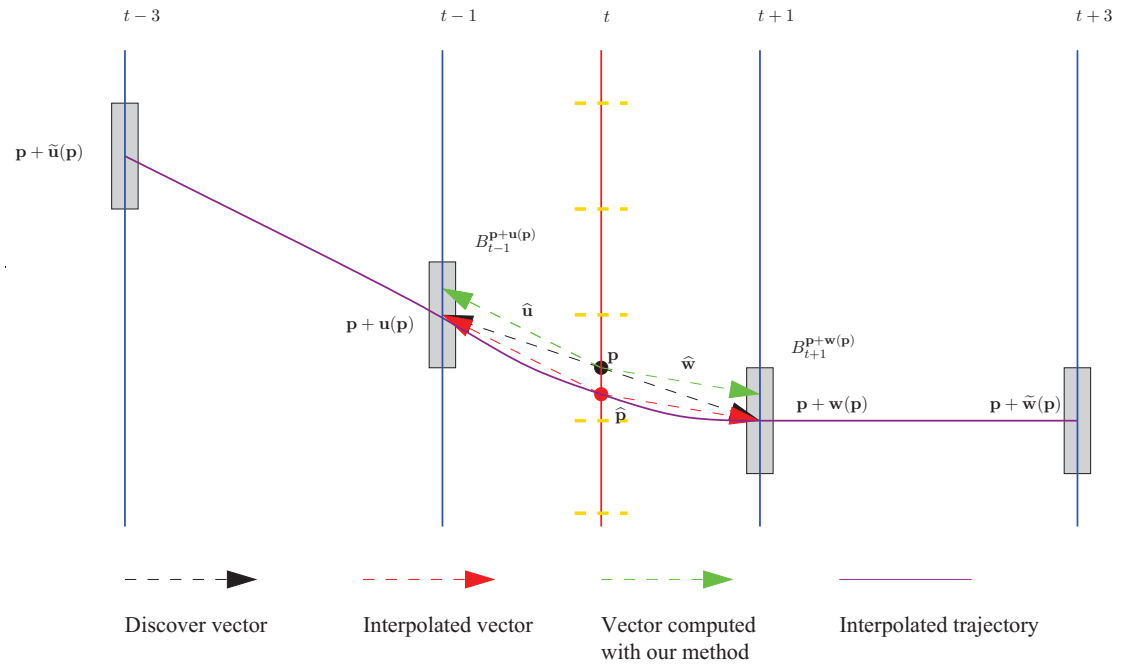


Figure 6: Algorithme HOMI pour l'interpolation du mouvement.

Cette méthode, expliquée en supposant la taille du GOP égale à 2, peut être facilement entendu pour tailles du GOP plus grandes.

Les résultats obtenus sont dans les tables 1, 2 et 3 en termes de métrique de Bjontegaard. On observe que les performance débit-distorsion sont toujours mieux que ceux de DISCOVER. On a des gains plus petits pour taille du GOP égale à 2 (0.83% de réduction en débit par rapport à DISCOVER). Mais on a des meilleurs résultats pour tailles du GOP égaux à 4 et 8, parce que les trames utilisées pour l'interpolation sont plus loin. On obtient une réduction en débit jusqu'à 11.68% et 18.37% par rapport à DISCOVER, respectivement, pour taille du GOP égale à 4 et 8.

On a aussi proposé certaines variantes à cet algorithme, en supposant de tailles des blocs plus petite (avec des techniques de recherche par blocs avec embrochement), et en simplifiant la procédure, puisque l'estimation du mouvement entre $t-1$ et $t-3$ a été déjà calculée pour l'estimation de la TWZ à l'instant $t-2$.

On a aussi proposé des techniques de raffinement pour l'estimation de grandes tailles du GOP. Par exemple si la taille du GOP est égale à 4, on peut considérer les trames I_{t-2} and I_{t+2} comme deux TCs consécutives. La trame à l'instant t on l'appellera TWZ centrale et les autres deux ($t-1$ and $t+1$) seront appelées TWZ latérales. On observe (dès résultats expérimentaux) que le PSNR pour les TWZ latérales est plus grande que celui des TWZ centrales (avec le même débit). Ça est due à la qualité différente de l'information adjacente qui a été produite par trames qui sont à une distance différente. Donc, on a proposé d'améliorer la qualité de la trame centrale en la ré-estimant en utilisant les trames déjà codées I_{t-1} et I_{t+1}. On commence par estimer la TWZ centrale $\widehat{I}_t$, en interpolant

	HOMI8		HOMI4		FastHOMI8		FastHOMI4	
	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}
container	-0.13	0.01	-0.67	0.04	-0.05	0.00	-0.08	0.00
akiyo	-0.29	0.02	-0.40	0.04	-0.05	0.01	0.41	-0.01
news	-0.66	0.04	-1.33	0.09	-0.74	0.05	-0.79	0.05
hall	-0.53	0.03	-1.45	0.09	0.11	-0.01	0.23	-0.02
eric	-1.80	0.10	-1.25	0.07	-1.18	0.06	-1.55	0.08
paris	0.30	-0.02	-0.59	0.04	0.51	-0.03	0.51	-0.03
silent	-0.49	0.02	-1.09	0.05	0.16	-0.01	0.09	-0.00
waterfall	-0.06	0.00	-0.34	0.02	0.04	-0.00	0.04	-0.00
flower	1.03	-0.07	1.47	-0.10	1.80	-0.12	1.77	-0.12
foreman	-1.35	0.07	-3.66	0.18	0.14	-0.01	0.13	-0.01
tempete	0.09	-0.01	0.23	-0.01	0.26	-0.01	-0.07	0.00
football	-4.16	0.21	-6.17	0.31	-2.78	0.14	-2.90	0.15
city	-3.31	0.16	-3.19	0.15	-3.24	0.16	-3.22	0.15
coastguard	0.00	0.00	0.46	-0.02	-0.38	0.02	-0.37	0.02
mean	-0.83	0.04	-1.13	0.06	-0.44	0.02	-0.46	0.02

Table 1: Performance débit-distorsion pour HOMI par rapport à DISCOVER - GOP size = 2

	HOMI8		HOMI4		FastHOMI8		FastHOMI4	
	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}
container	0.07	-0.00	-0.74	0.04	-0.35	0.02	-0.43	0.02
akiyo	-2.45	0.11	-0.39	0.03	-0.16	0.03	-2.39	0.08
news	-0.59	0.04	-2.23	0.13	-0.83	0.04	-1.19	0.06
hall	-0.44	0.03	-1.58	0.10	1.20	-0.07	0.53	-0.03
eric	-11.03	0.45	-9.46	0.39	-10.14	0.41	-10.02	0.41
paris	-1.80	0.09	-3.72	0.18	-1.89	0.09	-1.73	0.09
silent	-1.54	0.08	-0.52	0.08	0.87	-0.03	0.29	0.01
waterfall	0.35	-0.01	-0.18	0.00	0.43	-0.02	0.41	-0.02
flower	1.44	-0.08	1.43	-0.07	-0.80	0.04	-0.53	0.03
foreman	-6.27	0.26	-11.68	0.46	-5.77	0.26	-6.12	0.26
tempete	0.20	-0.01	0.75	-0.03	0.22	-0.01	0.36	-0.01
football	-6.59	0.32	-8.24	0.41	-6.31	0.32	-6.71	0.34
city	-8.70	0.32	-3.86	0.17	-9.08	0.31	-8.85	0.30
coastguard	1.04	-0.03	2.06	-0.06	-0.61	0.02	-0.42	0.01
mean	-2.59	0.11	-2.74	0.13	-2.37	0.10	-2.63	0.11

Table 2: Performance débit-distorsion pour HOMI par rapport à DISCOVER - GOP size = 4

les TCs I_{t-6}, I_{t-2}, I_{t+2} and I_{t+6}. Soit I_t la TWZ décodée. L'estimation $\widehat{I}_{t-1}$ de la TWZ latérale est obtenu en interpolant avec HOMI I_{t-4}, I_{t-2}, I_t et I_{t+2}. De la même façon, l'estimation de la TWZ latérale $\widehat{I}_{t+1}$ est obtenue en utilisant I_{t-2}, I_t, I_{t+2} et I_{t+4}. Maintenant, I_{t-1} et I_{t+1} sont disponibles au décodeur. Donc, on peut ré-estimer I_t, en utilisant HOMI sur I_{t-2}, I_{t-1}, I_{t+1} et I_{t+2}. Cette nouvelle estimation est corrigée en utilisant le mêmes bits de parité qu'on avait été envoyé dans le premier passage. Donc, on ne demande pas d'envoyer d'autres bits de paritéś. Les Figures 7 résument toutes les passages.

Inter-view interpolation with priority to large disparity (IPLD)

Dans cette section, on introduit une nouvelle méthode pour l'interpolation inter-vue pour le codage multi-vue. On supposera dorénavant que les cameras sont alignées.

L'idée pour notre algorithme est la suivante : quand deux objets (plus précisément,

	HOMI8		HOMI4		FastHOMI8		FastHOMI4	
	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}
container	0.34	-0.01	-1.60	0.07	0.85	-0.03	0.35	-0.01
akiyo	0.56	0.02	-4.29	0.20	-7.76	0.47	-9.95	0.52
news	-0.66	0.03	0.26	0.01	0.18	0.01	0.33	-0.01
hall	0.37	-0.07	-1.96	0.07	16.16	-0.92	17.27	-0.99
eric	-11.39	0.48	-10.39	0.47	-2.66	0.17	-3.05	0.18
paris	-5.96	0.26	-9.10	0.39	-3.18	0.13	-3.18	0.14
silent	-2.27	0.09	-0.87	-0.16	-3.40	0.15	-3.44	0.14
waterfall	0.65	-0.00	0.49	0.01	-0.77	0.05	-0.62	0.04
flower	0.45	-0.02	-3.21	0.14	12.61	-0.57	12.86	-0.58
foreman	-7.60	0.31	-12.45	0.47	-6.33	0.23	-5.84	0.19
tempete	-0.13	0.01	0.63	-0.03	3.74	-0.16	4.04	-0.17
football	-6.39	0.38	-7.72	0.50	-5.57	0.41	-5.72	0.43
city	-18.37	0.55	-7.63	0.32	-18.34	0.52	-14.32	0.39
coastguard	2.05	-0.03	1.97	-0.06	10.06	-0.23	10.71	-0.24
mean	-3.45	0.14	-3.99	0.17	-0.32	0.02	-0.04	0.00

Table 3: Performance débit-distorsion pour HOMI par rapport à DISCOVER - GOP size = 8

deux blocs de pixels) des TCs de référence ont deux trajectoires estimées qui passent dans la même position dans la trame centrale, on peut considérer la profondeur des objets et on sélectionne l'objet le plus proche à la camera et non l'objet qui passe dans le point le plus proche du centre du bloc (comme dans DISCOVER dans l'équation (1)). En fait, les objet du premier plan (qui ont une profondeur plus petite) cachent les objets du fond et doivent être préférées pour la procédure d'interpolation. Donc, on modifie l'Eq. (1) en additionnant une terme de pénalisation pour petites disparités (qui se traduit en grandes valeurs de disparité). On obtient l'équation suivante :

$$\mathbf{q}^*(\mathbf{p}) = \arg\min_{\mathbf{q}} \left(\left\| \mathbf{q} + \frac{1}{2}\mathbf{v}(\mathbf{q}) - \mathbf{p} \right\|^2 - \gamma \left\| \mathbf{v}(\mathbf{q}) \right\|^2 \right) \tag{4}$$

où le paramètre de pénalisation $\gamma > 0$ doit être choisi expérimentalement . Donc, notre méthode consiste simplement en remplacer l'Eq. (1) de DISCOVER avec l' Eq. (4); les autres étapes de DISCOVER restent les mêmes. On appellera cette méthode "interpolation with priority to large disparity" (IPLD).

On a comparé les performances débit-distorsion par la métrique de Bjontegaard de notre algorithme par rapport à DISCOVER en supposant un schéma asymétrique. Les résultats en Table 4 concerne seulement la camera Wyner-Ziv. On observe qu'on a des grandes améliorations en terme de débit, surtout pour certaines séquences comme *newspaper* et *balloons*. En moyenne, on réduit le débit total du -7.72%.

Fusion basée sur la détection d'occlusions

Pour le problème de la fusion on considère une camera hybride dans un schéma hybride 1/2 (voir Fig. 5(b)). Donc chaque TWZ $I_{t,k}$ a quatre TCs voisines : $I_{t,k-1}$, $I_{t,k+1}$, $I_{t-1,k}$, $I_{t+1,k}$. Chaque méthode d'interpolation temporelle ou inter-vue peut être utilisée pour produire les estimations $\widehat{I}_T$ and $\widehat{I}_V$. L'algorithme de fusion proposé consiste dans l'avoir

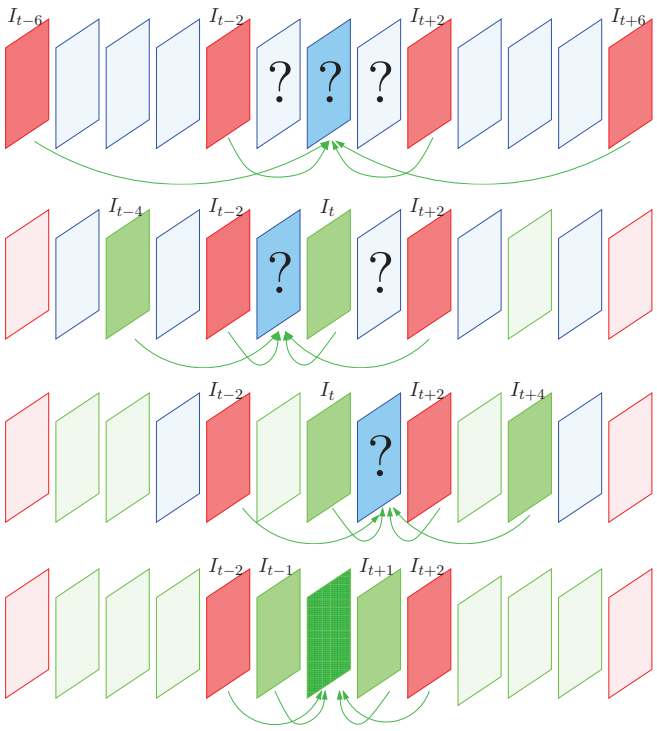


Figure 7: Estimation de la TWZ pour taille du GOP égale à 4 (les TCs sont en rouge, les TWZ qui doivent être estimées sont en blue, et les TWZ déjà décodées sont en vert. Les TWZ qui doivent être raffinées sont en vert foncé: on estime d'abord I_t (a), et puis à partir de ça, on estime I_{t-1} (b) et I_{t+1} (c). Enfin, on raffine la trame centrale I_t (d).

déjà les champs de disparité $d_{k-1,k+1}(m,n)$ (de $I_{t,k-1}$ à $I_{t,k+1}$) et $d_{k+1,k-1}(m,n)$ (de $I_{t,k+1}$ à $I_{t,k-1}$). Après on estime la cohérence de ceux deux champs [Fua93a, EW02, SS03]: si le pixel $\mathbf{p} = (m,n)$ dans k_1 corresponds au pixel $(m, n + d_{k_1,k_2}(m,n))$ dans k_2, la disparité du premier doit etre l'opposé du deuxième. Donc, la somme absolue de ces quantités est une mesure de la cohérence en disparité de k_1 à k_2. On peut écrire :

$$R_{k_1,k_2}(m,n) = |d_{k_1,k_2}(m,n) + d_{k_2,k_1}(m, n + d_{k_1,k_2}(m,n))| \tag{5}$$

Pour une disparité parfaitement cohérente, on aurait obtenu $R = 0$ partout. Mais il faut considérer aussi un terme de tolérance $\tau \geq 0$.

On définit la masque d'occlusion de k_1 à k_2 comme:

$$O_{k_1,k_2}(m,n) = u(R_{k_1,k_2}(m,n) - \tau) \tag{6}$$

où $u(\cdot)$ est la fonction d'Heaviside, définie comme

$$u(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

	Δ_{R} [%]	Δ_{PSNR} [dB]
balloons	-22.71	1.05
book arrival	-5.36	0.53
door flowers	-5.40	0.24
leaving laptop	-7.62	0.32
kendo	-9.26	0.50
lovebird	2.38	-0.09
newspaper	-22.09	0.97
pantomime	0.19	0.00
outdoor	0.35	-0.03
mean	**-7.72**	**0.38**

Table 4: Métrique de Bjontegaard: IPLD par rapport à DISCOVER (WZ caméra, schéma asymétrique)

On utilise cette masque d'occlusion $O_{k-1,k+1}$ avec $O_{k+1,k-1}$ pour améliorer la procédure de fusion. Pour chaque pixel $\mathbf{p}$, on décide quand utiliser seulement l' interpolation temporelle (due à une occlusion détectée) et quand utiliser l'image fondue ([MMCPP09a]). avec la différence que on utilise les estimations obtenues par HOMI et IPLD au lieu de celles obtenues par DISCOVER.

Validation adaptative

Comme montré dans la littérature scientifique, et confirmé dans nos tests, aussi si la fusion améliore la qualité de l'information adjacente en moyenne, pour certaines séquences ça n'est pas toujours vrai. Ça est surtout un problème pour les séquences dont une interpolation est beaucoup mieux que l'autre. Donc, on doit trouver une méthode qui décide quelle image choisir comme information adjacente entre $\widehat{I}_T$, $\widehat{I}_V$ et $\widetilde{I}$.

Le meilleure information adjacente est celle que demande le débit minimum de bits de parité. Soient R_T, R_V et R_F les débits (en bits par pixel) pour corriger l'estimation temporelle, inter-vue et leur fusion. On prend notre décision entre les trois images en observant, pour une petite nombre de TWZ, le débit nécessaire pour corriger les interpolations temporelle et inter-vue. On appelle $\delta_R = R_T - R_V$ et on s'attend que ce paramètre est très corrélé avec la décision optimale. On modèle la décision optimale comme une variable aléatoire D discrète qui peut assumer trois valeurs $\{T, V, F\}$.

Donc on obtiens un critère MAP :

$$\begin{aligned}\widehat{D} &= \arg\max_d P(D = d|\delta_R) \\ &= \arg\max_d \frac{p(\delta_R|D = d)P(D = d)}{p(\delta_R)} \\ &= \arg\max_d p(\delta_R|D = d)P(D = d) \end{aligned} \tag{7}$$

où $P(\cdot)$ est la densité de probabilité de D et $p(\cdot)$ est la probabilité de δ_R.

Donc, en observant la valeur de δ_R pour une certaine sous-ensemble des trames (1 sur

sequence	Δ_{R} [%]	Δ_{PSNR} [dB]
balloons	-37.03	2.20
book arrival	-2.08	0.13
door flowers	-5.01	0.24
leaving laptop	0.21	-0.06
kendo	-1.59	0.10
lovebird	-16.34	0.80
newspaper	-24.89	1.34
pantomime	4.65	-0.26
outdoor	0.08	0.00
mean	**-9.11**	**0.42**

Table 5: Performance débit-distorsion pour la validation adaptative par rapport à la technique de fusion [MMCPP09a].

100 est suffisante), on prend la décision sur quelle interpolation utiliser.

On a testé notre algorithme de validation adaptative sur plusieurs séquences multi-vues. Les résultats en termes de métrique de Bjontegaard sont en Figure 5. Avec notre méthode on arrive à obtenir un gain moyen en termes de réduction du débit de 9.11% par rapport à la technique de fusion proposée par [MMCPP09a].

Applications du codage vidéo distribué au format vidéo multi-vues plus profondeur

Dans cette section, on va introduire le format mulview video plus depth (MVD - vidéo multi-vue plus profondeur). Dans ce format chaque camera fournit une image à couleur, appelée texture, et une carte de profondeur à 8 bits. Normalement, on retrouve ce format pour la Télévision 3D Immersive et pour la télévision à point de vue libre (FTV - free view point television). Donc, on considère aussi que l'utilisateur peut changer son point de vue pendant la transmission de données. Le standard H.264/AVC n'est pas très indiqué pour ces types d'applications, parce que la continuité de la reproduction ne peut pas être assurée. Si la commutation de vue tombe en correspondance d'une trame P ou B, très probablement la trame de référence nécessaire pour reconstruire la trame actuelle n'est pas disponible au décodeur. Le codage vidéo distribué peut résoudre ce type de problème, parce que les trames WZ peut être décodée et reconstruite indépendamment de la méthode utilisée pour la génération de l'information adjacente. Dans ce chapitre on va explorer des nouvelles techniques de codage vidéo distribué pour l'amélioration des services de streaming multi-vues. En particulier, on va proposer

- une nouvelle technique pour le codage des cartes de profondeur.
- des nouvelles solutions pour assurer l'interactivité dans le cadre de Immersive TV.

On va terminer avec la proposition d'une nouvelle architecture pour le format MVD.

Interactivité pour la commutation de vues

Dans concernant l'interactivité, on introduit le paradigme *Interactive Multi View Video Streaming IMVS*: la vidéo est d'abord pre-encodé, stocké dans un serveur et puis envoyé à utiliser selon sa demande. Deux paramètres importantes pour la transmission de vidéo multi-vue sont l'espace en mémoire occupé sur le serveur et la bande utilisée pour la transmission. Ces deux mesures sont équivalentes dans le cadre de télévision non interactive. Mais dans le cas de streaming multi-vue interactive, l'utilisateur sélectionne la vue dont il est intéressé. Donc, le serveur envoie seulement les données associées à la vue qu'il veut visualiser (voir Fig. 8). Nous, on veut minimiser et l'espace en mémoire sur le serveur et la bande nécessaire pour envoyer ces données. Ces deux paramètres sont en contraste l'un l'autre dans le cadre du IMVS [COC11].

En fait: d'un coté, si on utilise H.264/AVC et on a V vues, on pourrait penser de codeur le résidus de chaque trame P en utilisant comme référence chaque différente vue (voir Fig.9(b)). Cette approche-là est appelée **Redundant P-frames**. Tout ces résidus vont être stocké sur le serveur, mais on envoie à l'utiliser seulement le résidus associé à sa palette de vues (donc selon la trame de référence à disposition de l'utilisateur). Mais le problème le plus important est que maintenant l'espace sur le serveur est augmenté

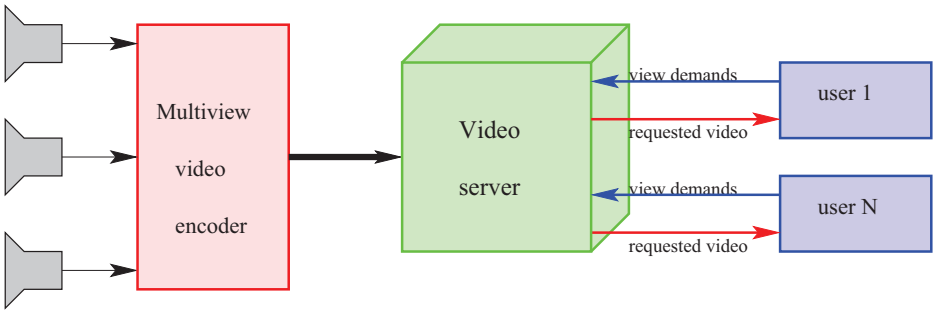


Figure 8: Dans le paradigme IMVS la vidéo est d'abord pre-encodé, stocké dans un serveur et puis envoyé à utiliser selon sa demande

exponentiellement avec V. De l'autre coté. on pourrait penser de coder chaque vue à chaque instant en mode INTRA. Comme ça on évite l'impossibilité de reconstruire les trames P. Donc maintenant l'espace en mémoire pour la vidéo est diminué, mais la bande nécessaire pour transmettre une trame I au lieu qu'une trame P est beaucoup plus grande. Cet approche-là est appelé **All I-frames** (voir Fig.9(a)). [COC11].

Une approche alternative utilise le codage vidé distribué. Chaque vue est codée avec DISCOVER, indépendamment des autres vues, comme en Fig. 9(c). Quand un utilisateur veut changer de vue, la nouvelle image peut être et une TC et une TWZ. Dans le premier cas, elle est encodée comme d'habitude. Dans le deuxième cas, on doit adapter le mécanisme de reproduction de l'information adjacente, mais le codeur reste le même. Avec cette approche, on réduit l'espace sur le serveur (pour chaque trame WZ il y a seulement les bits de parité) par rapport au Redundant P-Frames et envoie presque la même quantité de bits que pour une trame clés. Donc, le codage vidéo distribué est un compromis entre All INTRA et Redundant P-Frames. Déjà Cheung et *al.* [COC11] ont proposé d'insérer des trames WZ, appelées **M-Frames (Merge Frames)** dans le flux vidéo. Comme information adjacente on utilisera la trame disponible à l'instant précédent la commutation. Ils trouvent, enfin, une allocation optimale des Trames I, Redundant P-frames et M-frames qui optimise le prestation RD. En effet, ils sont intéressé seulement à le cas multi-vue sans cartes de profondeur. Jusqu'à maintenant très peu de travaux on considérait le format MVD pour IMVS. Par contre, nous, on a exploré des nouvelles solutions pour la génération de l'information adjacente dans ce cadre-là, en utilisant des algorithmes des synthèse de vue à partir de la carte de profondeur.

Méthodes proposées

Pour simplicité, on considère qu'on a seulement deux caméras, qu'il y a une trame clé chaque N trames et que les trames clés d'une vue à l'autre sont déplacées d'un mi-GOP.

On considère la notation suivante. On appelle m l'instant de commutation comme en Fig. 10 de la première à la deuxième vue. On appelle k la position de la trame clés

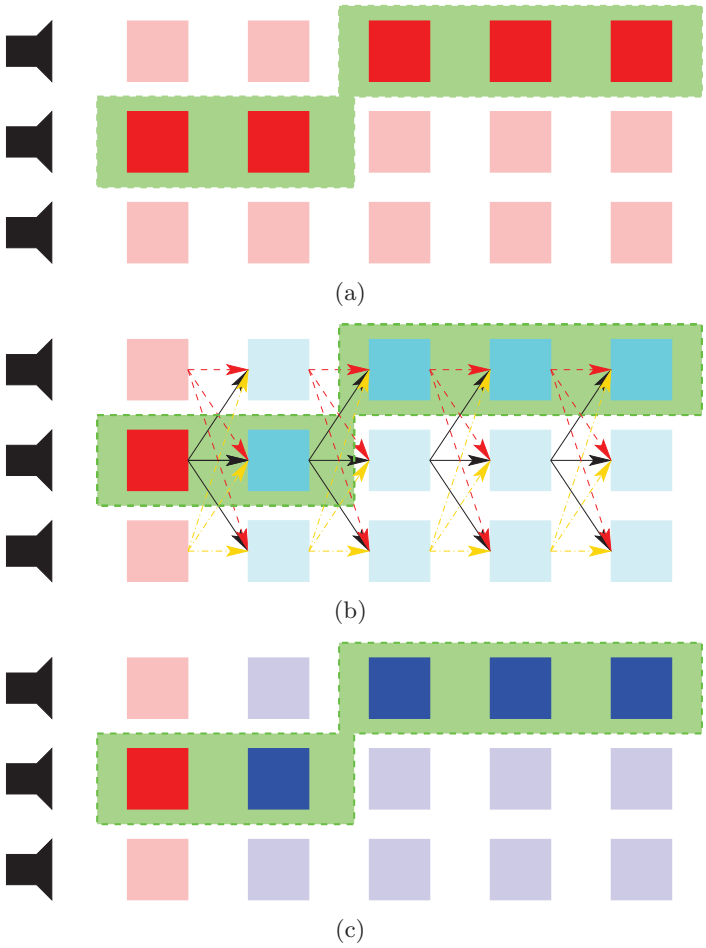


Figure 9: Trois différentes solutions pour assurer la continuité de la reproduction pendant la commutation de vues: all INTRA Frames (a), Redundant P-frames (b) et CVD (c). Les Trames clés et INTRA sont en rouge. Les trames WZ sont en bleue. Les trames avec fond vert sont visualisé par l'utilisateur

précédente la plus proche de m dans la deuxième vue. Pour la périodicité on aura comme configurations de commutation $m = k, k+1, k+2, \ldots k+N-1$.

On appelle J_n la trame utilisée pour construire l'information adjacente pour la deuxième vue avec la trame clé $k+N$. On appelle $\widetilde{I}_n$ l'estimation de la trame de la deuxième vue à l'instant n en utilisant la technique de DIBR avec inpainting proposé par [DPP10] et $\widehat{I}_n$sa version corrigée par les bits de parité.

On a proposé différents méthodes pour assurer la continuité de la reproduction. On considère seulement le cas pour $m = k+2$, mais on peut le généraliser pour chaque m.

- **Advanced DIBR (A-DIBR)** - On applique DIBR à l'instant $m-1$ sur la première vue. Donc, on utilisera cette trame synthétisée avec la trame clé suivante pour estimer toutes les trames du GOP (voir Fig. 11(a))

- **Advanced DIBR + correction (A-DIBRc)** - C'est presque la même solution,

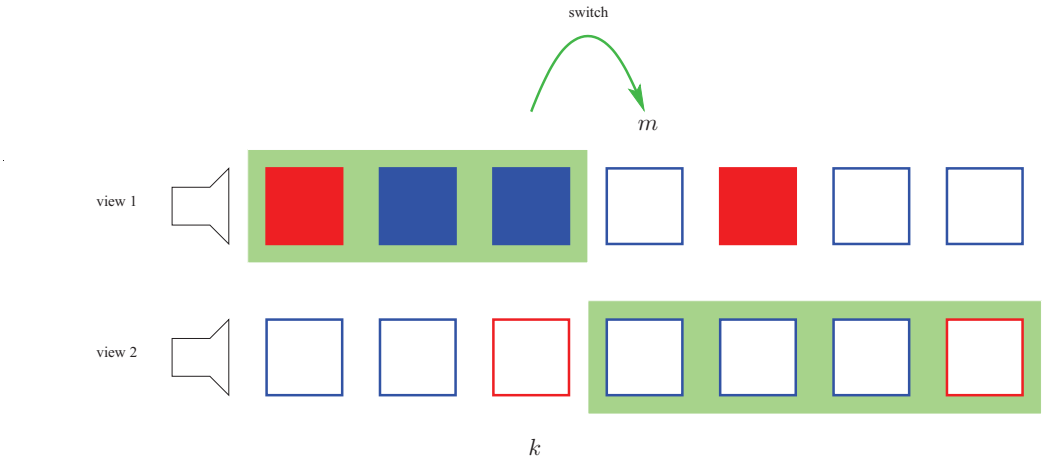


Figure 10: Un exemple de commutation de vue

mais on envoie aussi les bits de parité pour corriger l'estimation J_{m-1}. Le reste de l'algorithme est le même. On espère qu'en envoyant les bits de parité on aura une estimation meilleure et de conséquence aussi les autres trames du GOP seront de meilleure qualité (voir Fig. 11(b)).

- **Immediate DIBR (I-DIBR)** - Cette fois on applique DIBR à l'instant m. Cette estimation sera utilisée pour estimer les autres trames de la deuxième vue (voir Fig. 11(c)).
- **Immediate DIBR + correction (I-DIBRc)** - On a aussi la variante dans laquelle on corrige l'estimation de la deuxième vue avec les bits de parité (voir Fig. 11(d)).
- **noDIBR** - On n'applique pas du DIBR, mais on envoie directement la trame clé de référence nécessaire pour reconstruire les trames WZ de la deuxième vue pendant la commutation (voir Fig. 11(e)).

Pour valider notre méthode on a utilisé trois séquences multi-vue plus profondeur. On a calculé les performance débit-distorsion (en in Tab. 6,7, 8) en termes de métrique de Bjontegaard par rapport à la méthode triviale noDIBR. On a calculé la distorsion seulement pour les trames qui sont visualisées sur l'écran de l'utilisateur. Par contre, comme débit on a considéré tous les bits qui sont réellement envoyés à l'utilisateur. On a donc remarqué que la meilleure méthode dépend de la configuration de commutation. On pourrait donc supposer d'avoir un système qui détecte la configuration de commutation et, donc, choisir quelle méthode utiliser.

Estimation des cartes de profondeur pour CVD

Puisque on n'utilise jamais la corrélation entre texture et profondeur, dans cette section on propose plusieurs méthode pour améliorer la génération de l'information adjacente

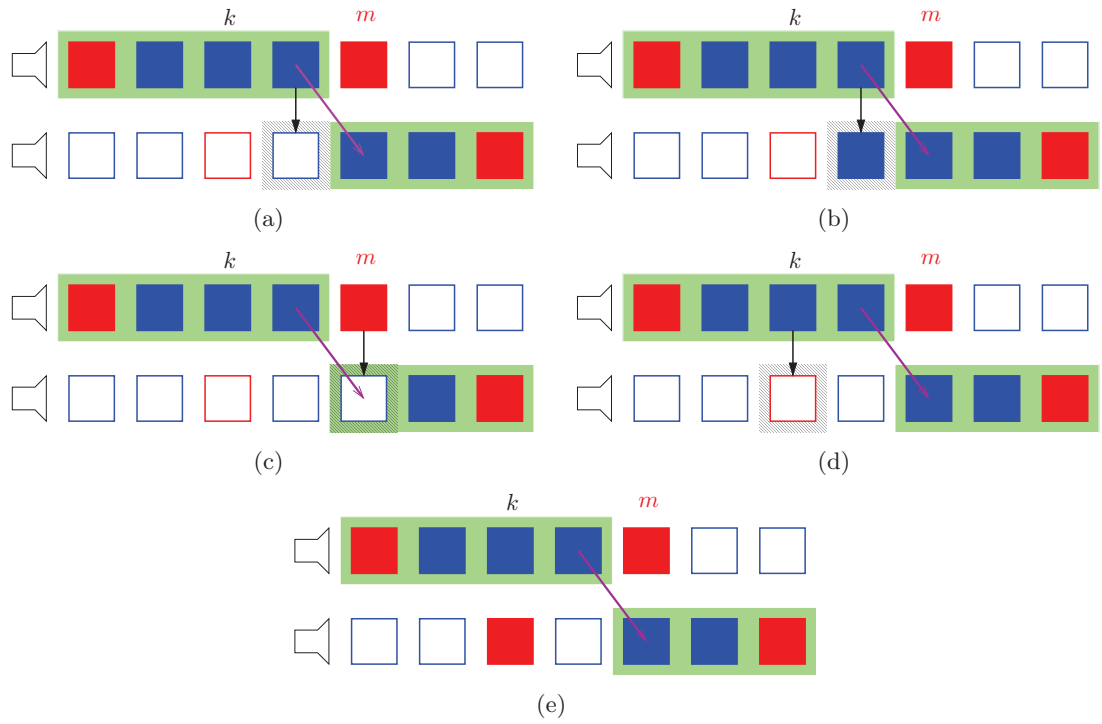


Figure 11: Les différents solution pour $m = k+2$: (a) A-DIBR; (b) A-DIBRc; (c) I-DIBR; (d) I-DIBRc; (e) GOPp; (f) noDIBR. Les trames clés sont en rouge et les trames WZ sont en bleue. Les trames à fond vert sont visualisées par l'utilisateur. Les trames remplies sont envoyées du serveur à l'utilisateur. La flèche noire montre que le DIBR est appliqué.

concernant les cartes de profondeur.

On considère une structure pour TWZ et TC comme en Fig. 12. Les textures sont indiquées comme I_m et la respective carte de profondeur avec I_m^D. Donc comme algorithmes pour la génération de l'information adjacente de la carte de profondeur à l'instant t on utiliseras les trames I_ℓ et/ou I_ℓ^D (où $\ell \in \{t-3, t-1, t+1, t+3\}$).

Une première méthode est d'estimer I_t^D en appliquant DISCOVER sur I_{t-1}^D et I_{t+1}^D. Les champs de vecteur de mouvement sont appliqués à I_{t-1}^D and I_{t-1}^D pour obtenir l'information adjacente. Cette méthode est appelée ZD (DISCOVER over Depth Data).

Une autre technique est celles d'appliquer aussi HOMI aux cartes de profondeur. On applique HOMI sur I_{t-1}^D, I_{t+1}^D avec I_{t-3}^D et I_{t+3}^D. Les champs de vecteur de mouvement sont appliqué à I_{t-1}^D and I_{t-1}^D pour obtenir l'information adjacente. Cette méthode est appelée ZD-ZH (DISCOVER and HOMI over Depth Data).

Puisque le mouvement dans les cartes de profondeur est très proche à celui de texture. on a proposé les méthodes suivantes. Une première est appelé TD (DISCOVER over Texture): DISCOVER est appliqué sur les images de textures I_{t-1} et I_{t+1}, mais les champs de mouvements obtenus sont appliqué à I_{t-1}^D et I_{t-1}^D.

Une autre méthode est de raffiner ces vecteurs encore sur les textures , et donc appliquer HOMI à I_{t-3}, I_{t-1}, I_{t+1} et I_{t+3}. Les champs obtenus sont appliqués à I_{t-1}^D and I_{t-1}^D pour

ballet				
	texture		depth	
méthode	Δ_R [%]	Δ_{PSNR} [dB]	Δ_R [%]	Δ_{PSNR} [dB]
$m = k$				
I-DIBR/GOPp	13.40	-0.80	13.76	-0.93
$m = k + 1$				
A-DIBR	**4.81**	**-0.27**	**1.76**	**-0.13**
I-DIBR	10.07	-0.59	9.69	-0.62
I-DIBRc	3.24	-0.18	1.73	-0.11
$m = k + 2$				
A-DIBR	0.54	-0.02	-1.54	0.08
A-DIBRc	-0.42	0.02	-2.11	0.13
I-DIBR	7.34	-0.43	4.58	-0.29
I-DIBRc	**-2.22**	**0.12**	**-3.57**	**0.21**
$m = k + 3$				
A-DIBR	-5.53	0.33	-6.67	0.41
A-DIBRc	**-5.82**	**0.34**	**-6.67**	**0.41**
I-DIBR	18.29	-1.04	17.21	-1.05
I-DIBRc	8.93	-0.50	8.65	-0.52

Table 6: Comparaison des différentes méthodes en termes de métrique de Bjontegaard par rapport à noDIBR pour la séquence *ballet*

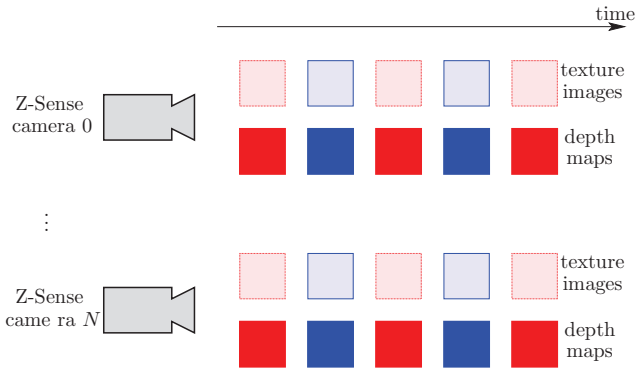


Figure 12: La répartition des trames pour notre méthode. Les TC sont en rouge, les TWZ sont en bleue. Les images de textures et les cartes de profondeur sont codées indépendamment.

obtenir l'information adjacente. Cette méthode est appelée TD-TH.

La dernière technique est d'utiliser DISCOVER sur les trames de texture I_{t-1} et I_{t+1} et de raffiner ces vecteur en utilisant HOMI sur les cartes de profondeur I^D_{t-3}, I^D_{t-1}, I^D_{t+1} et I^D_{t+3}. Comme d'habitude, les vecteurs de mouvement obtenus sont appliqués aux trames I^D_{t-1}, I^D_{t+1}. L'idée est que la texture est plus riche en information que les cartes de profondeur. Donc on initialise les vecteur sur les textures, mais on les raffine sur les cartes de profondeur.

Pour valider notre technique, on a du d'abord chercher les bonnes valeurs du paramètre λ pour l'Eq. (3). On a donc cherché la valeur qui minimise le PSNR de l'IA et on a trouvé les valeurs qui sont indiquées en Tab. 9 On a observé que si on utilise seulement la texture on a besoin d'une valeur d'α plus grande. Par contre si on utilise les textures pour l'initialisation mais puis on raffine les vecteurs sur les cartes de profondeur, on peut

breakdancers				
	texture		depth	
méthode	Δ_R [%]	Δ_{PSNR} [dB]	Δ_R [%]	Δ_{PSNR} [dB]
$m = k$				
I-DIBR/GOPp	1.52	-0.08	0.48	-0.01
$m = k + 1$				
A-DIBR/GOPp	**-6.21**	**0.33**	**-5.82**	**0.36**
I-DIBR	0.89	-0.04	-0.30	0.03
I-DIBRc	-4.42	0.22	-4.38	0.26
$m = k + 2$				
A-DIBR	-5.76	0.29	-4.64	0.28
A-DIBRc	-5.77	0.29	-5.01	0.29
I-DIBR	-3.55	0.18	-2.88	0.19
I-DIBRc	**-11.01**	**0.56**	**-8.04**	**0.50**
$m = k + 3$				
A-DIBR	-9.26	0.48	-8.42	0.50
A-DIBRc	**-13.36**	**0.68**	**-8.67**	**0.55**
I-DIBR	12.38	-0.59	17.60	-0.99
I-DIBRc	5.18	-0.22	12.80	-0.71

Table 7: Comparaison des différentes méthodes en termes de métrique de Bjontegaard par rapport à noDIBR pour la séquence *breakdancers*

accepter des plus grand déviations des vecteurs, et donc une valeur de λ plus grande.

On a, donc, analysé les performances RD de notre système (en Tab. 10 et 11) par la métrique de Bjontegaard par rapport à la solution ZD. On a observé que avec la méthode TD-ZH on obtient une réduction du débit jusqu'à 11.06% et une augmentation du PSNR jusqu'à 0.44 dB.

Une nouvelle architecture pour la vidéo multi-vue dans le cadre du MVD

Dans ce dernière paragraphe, on propose une nouvelle architecture distribuée pour encoder le format MVD. On sait déjà depuis la naissance des architectures LDV [MSD+08] qu'une seule vue avec sa carte de profondeur est suffisante à reconstruire n'importe quel point de vue (en connaissant les paramètres intrinsèque et extrinsèque des caméras), à l'exception de zones d'occlusions. Dans une architecture distribuée on a la complication que chaque camera doit connaitre les zones d'occlusion par rapport à une autre camera, sans communiquer avec cette camera-là, mais en connaissant seulement ses positions relatives. On verra que un double DIBR sur cette camera sera suffisant pour détecter les zones d'occlusion.

Donc, sans perte de généralité, on suppose d'avoir des cameras que nous fournissent deux images de texture I_L et I_R et les respectives cartes de profondeur I_L^D et I_R^D. La structure de l'encodeur et du décodeur est montré en Fig. 13 et 14.

On suppose que la caméra de droite (texture plus profondeur) est codée INTRA.

Donc au décodeur la vue gauche peut être estimée à partir de celle droite à l'exception des zones d'occlusion.

Donc, à l'encodeur on peut appliquer du DIBR à la carte de profondeur gauche I_L^D et on obtient une estimation de la carte de profondeur droite $\widehat{I}_R^D$ (les zones d'occlusion vont être remplies par l'inpainting de Bertalmio). Donc, si on applique une autre fois

mobile				
	texture		depth	
méthode	Δ_R [%]	Δ_{PSNR} [dB]	Δ_R [%]	Δ_{PSNR} [dB]
$m = k$				
I-DIBR/GOPp	1.65	-0.04	1.10	-0.01
$m = k + 1$				
A-DIBR/GOPp	**-9.56**	**1.00**	**-5.82**	**0.36**
I-DIBR	-0.87	0.20	-0.50	0.10
I-DIBRc	-7.64	0.85	-5.60	0.40
$m = k + 2$				
A-DIBR	-4.21	0.26	-3.50	0.19
A-DIBRc	-5.60	0.39	-5.01	0.35
I-DIBR	-3.86	0.23	-2.11	0.25
I-DIBRc	**-8.16**	**0.43**	**-7.50**	**0.40**
GOPp	-6.62	0.32	-5.00	0.31
$m = k + 3$				
A-DIBR	-7.80	0.93	-7.02	0.80
A-DIBRc	**-12.64**	**1.68**	**-10.47**	**1.22**
I-DIBR	9.47	-0.83	10.12	-0.70
I-DIBRc	7.17	-0.47	8.44	-0.52
GOPp	-5.54	0.88	-5.11	0.67

Table 8: Comparaison des différentes méthodes en termes de métrique de Bjontegaard par rapport à noDIBR pour la séquence *mobile*

GOP size	2	4	8
λ_{TD-TH}	50	20	0
λ_{ZD-ZH}	10	8	2
λ_{TD-ZH}	4	2	2

Table 9: Values of λ for different techniques and GOP sizes

ballet				
	ZD-ZH	TD	TD-TH	TD-ZH
GOP size = 2				
Δ_R (%)	-3.46	-2.05	-5.83	**-6.08**
Δ_{PSNR} [dB]	0.17	0.10	0.29	**0.31**
GOP size = 4				
Δ_R (%)	1.22	8.29	-3.38	**-4.42**
Δ_{PSNR} [dB]	-0.04	-0.36	0.14	**0.17**
GOP size = 8				
Δ_R (%)	-4.34	7.22	-0.36	**-11.06**
Δ_{PSNR} [dB]	0.17	-0.29	0.04	**0.44**

Table 10: Performance débit-distorsion pour les cartes de profondeur de *ballet*

breakdancers				
	ZD-ZH	TD	TD-TH	TD-ZH
GOP size = 2				
Δ_R (%)	**-2.09**	1.51	-0.44	-0.93
Δ_{PSNR} [dB]	**0.09**	-0.07	0.01	0.04
GOP size = 4				
Δ_R (%)	-0.59	7.55	-0.55	**-0.94**
Δ_{PSNR} [dB]	0.02	-0.35	0.01	**0.02**
GOP size = 8				
Δ_R (%)	-0.96	7.27	3.66	**-4.50**
Δ_{PSNR} [dB]	0.03	-0.32	-0.16	**0.17**

Table 11: Performance débit-distorsion pour les cartes de profondeur de *breakdancers*

DIBR à cette estimation, on peut obtenir une estimation de la carte de profondeur gauche, ainsi que les zones d'occlusion, qui vont définir la masque d'occlusion O_L. Un traitement

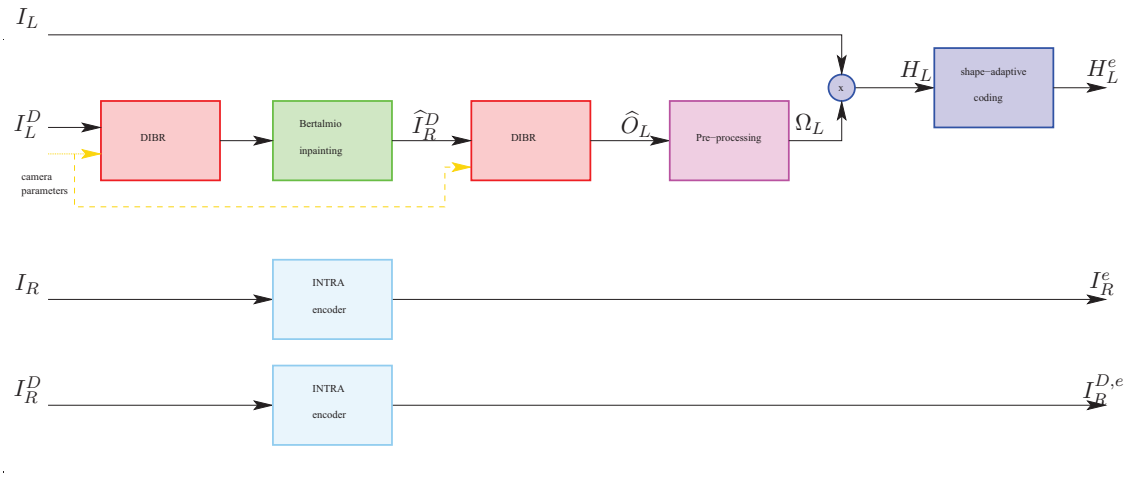


Figure 13: Structure de l'encodeur

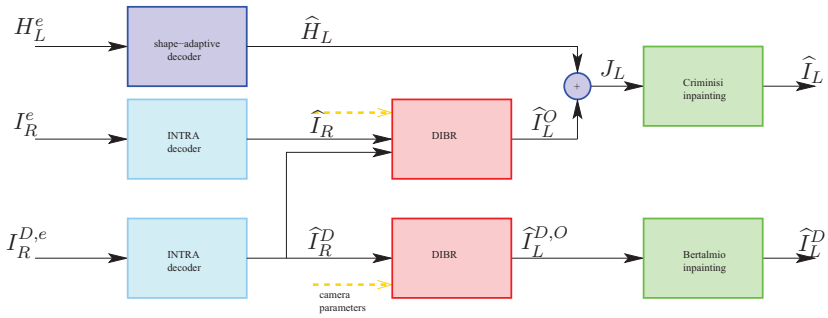


Figure 14: Structure du décodeur

de cette masque sera nécessaire: on éliminera les zones dont l'aire est trop petite et on dilatera les zones restantes, pour être sure de remplir correctement les zones d'occlusion. On appellera Ω_L la masque obtenue. Et donc de la caméra gauche, on envoie seulement les zones indiquées par cette masque. Puisque il s'agit de données creuses on envoie cette zones par des algorithmes de codage adapté à la forme (shape-adaptive coding) [LL00], [CPPV07].

La structure du décodeur est en Fig. 14. On a déjà dit qu'avec la vue droite on peut reconstruire celle de gauche avec DIBR, sauf que pour les zones d'occlusion. Ces régions seront remplis par les objet qu'on a envoyé. Si quelque zone reste non remplie, on peut appliquer l'inpainting de Criminisi. Dorénavant, on appellera caméra K, la camera codée INTRA et O, celle dont on envoie seulement les zones d'occlusion.

Pour valider notre architecture, on s'est comparé avec H.264/INTRA e DISCOVER. Pour faire ça, on a entendu notre architecture à trois caméras. La centrale est une camera K et les autres deux sont caméra O. On a considéré la séquence MVD *mobile* à la résolution de 720×540. En Fig. 15, on retrouve les résultats en débit distorsion de cette séquence et on peut voir qu'on améliore beaucoup par rapport à DISCOVER et H.264/INTRA.

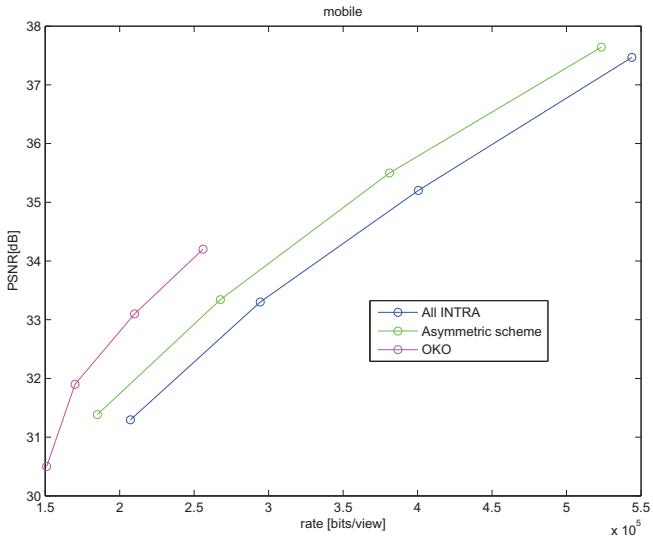


Figure 15: Performance débit-distorsion pour la séquence *mobile*

Conclusions

Dans cette thèse, on a proposé différents algorithmes pour améliorer la qualité de l'information adjacente par rapport au codeur de DISCOVER. Nos contributions dans ce domaine ont été :

- **High order temporal interpolation**: on est réussi à améliorer les performance de DISCOVER parce qu'on a utilisé quatre trames au lieu que deux pour l'interpolation des trajectoires. En fait, une interpolation d'ordre supérieure nous permet de considérer des modèles de mouvement plus complexes que celui linéaire.

- **Raffinement for grandes tailles du GOP**: normalement puisque DISCOVER définit une structure hiérarchique pour l'estimation et décodage pour tailles du GOP supérieures à deux, on risque d'avoir une qualité *flottante* des trames WZ. Nous, on a proposé de ré-estimer les trames de basse qualité en utilisant les nouvelles trames disponibles au décodeur et à les corriger en utilisant les mêmes bits de parité.

- **Interpolation inter-vue avec priorité au large valeur de disparité**: on a proposé des méthodes pour améliorer l'estimation de l'information adjacente dans le domaine de vues grâce à un algorithme qui donne priorité aux larges valeurs de disparité. En fait, les objets dont la disparité est plus grande sont plus proche de la caméra.

- **Algorithme de fusion avec détection des occlusions**: plusieurs algorithmes ont été proposés en littérature pour fusionner l'information adjacente obtenue par interpolation dans le temps ou entre les vues. Nous on a proposé de détecter les zones d'occlusion d'une vue à l'autre et d'éliminer celles-là. de l'information adjacente puisque ces zones sont responsables d'éventueles erreurs dans l'estimation finale.

- **Adaptive validation**: quand la qualité de l'information adjacente dans un domaine est beaucoup plus grande que dans l'autre domaine, la fusion n'améliore pas les performances débit-distorsion. Ce serait mieux de choisir directement la meilleure IA. On a donc proposé de détecter la meilleure IA avec une méthode de validation adaptative basée sur le débit de canal du boucle de retour. Les nombres de bits utilisés pour corriger l'information adjacente sont une estimation des sa qualité

Dans la partie finale de cette thèse, on s'est occupé du format MVD, Multiview Video plus Depth. En particulier, on a étudié différents applications du CVD au MVD. Ces contributions ont été :

- **Interactive multiview video streaming (IMVS)**: Dans un contexte interactive, on envoie à l'utilisateur (et donc au décodeur) seulement les trames qui sont réellement visualisées. Au contraire de H.264/AVC, CVD assure le décodage correct de

toutes les trames quand il y a une commutation de vue, sans que la reproduction soit interrompue. Pour ça, on a introduit différentes techniques pour la génération de l'information adjacente en correspondance d'un changement de vue.

- **Codage des cartes de profondeur WZ**: on a proposé différents algorithmes pour l'estimation des cartes de profondeur WZ en exploitant aussi les informations sur les textures.
- **Une nouvelle architecture distribuée pour le format MVD**: On a proposé une architecture distribuée pour MVD dans laquelle une seule vue est codée INTRA et pour les autres on envoie seulement les zones d'occlusion sans que les caméra communiquent entre eux.

Comme travaux futurs, on peut développer les idées suivantes :

- **Algorithme de validation adaptative avec détection de changement de scène**: En utilisant des critères de détection de changement de scène, on peut trouver la méthode optimale pour la mise à jour des paramètres pour le choix de l'information adjacente.
- **Amélioration de la qualité des images de texture, en utilisant information de la profondeur**: L'image de profondeur peut nous aider à mieux estimer la trajectoire des objets entre les trames dans le temps.
- **Méthode d'extrapolation pour MVD dans le contexte de l'IMVS**: Pour mieux satisfaire la requête du temps réel, on pourrait penser d'utiliser des algorithme de extrapolation au lieu que l'interpolation pour l'estimation des TWZ
- **Méthode d'allocation des bits pour la nouvelle architecture distribuée proposée**: on peut penser de faire une allocation du débit conjointe entre la camera K et O.

Table of Contents

List of Figures

List of Tables

List of Acronyms

AV	Adaptive Validation
DCT	Discrete Cosine Transform
DVC	Distributed video Coding
DSC	Distributed source Coding
DIBR	Depth Image Based Rendering
FTV	Free viewpoint television
GOP	Group of Pictures
HOMI	High Order Motion Interpolation
KF	Key Frame
IMVS	Interactive Multiview Video Streaming
IPLD	Interpolation with priority to large disparity values
LDV	Layered Depth Video
LR-CC	Left-Right Cross Consistency Check
MCTI	Motion compensated temporal interpolation
MB	Macroblock
MVD	Multiview video plus Depth
MDC	Multiple description coding
MVF	Motion Vector Field
NCC	Normalized Cross Correlation
PMF	Probability mass function
PDF	Probability density function

PSNR	Peak signal-to-noise ratio
SAD	Sum of Absolute Differences
SI	Side Information
SSD	Sum of Squared Differences
PSNR	Peak Signal-to-Noise Ratio
RD	Rate-Distortion
QP	Quantization parameter
QI	Quantization index
WZF	Wyner-Ziv Frame

Introduction

Video signals have a huge redundance in time and in space, and this correlation is exploited in classical video architectures (MPEG-x, H.263 and H.264/AVC) at the encoder side: as far as temporal correlation is concerned, consecutive frames are jointly encoded (by motion-compensated temporal prediction). On the other hand, spatial redundancy is thus exploited by block-based transform coding. The encoder computational effort for removing the temporal correlation (by motion estimation and compensation) is very large. Moreover, the memory requirement for storing the needed previous frames used as references is a requirement that may be difficult to meet, in particular on mobile devices.

As a consequence, classical video compression architectures present an asymmetry between the encoder and the decoder: the encoder is more complex than the decoder, with the latter being essentially a subset of the former. This is suitable for several applications, such as broadcast and one-to-many communications. Indeed, in these scenarios, the video is encoded once, but decoded several times. On the other hand, there are some applications, such as wireless sensor networks and visiocommunications, that have a limited capacity in terms of memory and battery. For these cases, standard architecture may not be suitable, and new approaches have to be conceived.

In the 1970s a new theoretical framework appears: the distributed source coding. In this paradigm, two sources are encoded separately but decoded jointly. Slepian and Wolf showed that, under mild constraints, this does not disrupt achievable performances (at least, asymptotically) in the case of lossless coding; then Wyner and Ziv extended this result for the case of lossy coding: provided that some conditions are met, there is no loss in terms of rate-distortion performance between distributed source coding and joint encoding. In the 2000s, distributed video coding (DVC) takes inspiration from the distributed source paradigm to support new emerging architectures, such as wireless sensors or networks of sensors with a restricted capacity in terms of complexity and memory, that have a limited capacity in terms of battery unable to support motion estimation at the encoder side.

The main characteristic of the distributed source coding paradigm is that the sources do not need to communicate to each other. This is suitable also for avoiding inter-camera communication in multiview video systems. Indeed, multiview videos present also a huge inter-view correlation. If they are encoded with standard techniques, like H.264/MVC, this dependency is exploited at the encoder in a similar manner to the temporal correlation for

monoview video. Then, a communication among the cameras is required for compressing the video with H.264/MVC. Since with distributed video coding we encode independently the different views, communication among the cameras can be avoided.

In this thesis we work on some problems and applications related to a DVC system. More precisely, we refer to one of the most popular framework for DVC, the Stanford codec. In the Stanford codec, the video stream is split into Key Frames (KFs) and Wyner-Ziv Frames (WZFs). Borrowing the terminology from the predictive video coding context, a KF and all the following WZFs before the next KF are said to form a group of pictures (GOP). Hence the distance between two successive KFs is called GOP size. The KFs are INTRA coded (i.e. without motion estimation and compensation). The Wyner-Ziv Frames are fed into a systematic channel coder. The systematic part is discarded and the parity bits are sent to the decoder. At the decoder side, an estimation of the Wyner-Ziv Frame is needed. It can be obtained by interpolation of the already decoded frames. This estimation is called Side Information (SI) and it can be considered as a noisy version of the true WZF. The channel decoder must correct these estimation errors by using the parity bits. The European project DISCOVER implemented the Stanford architecture and defined effective tools for coding the KFs and the WZFs. It has become the reference technique for distributed monoview and multiview video coding.

The first part of this thesis is focused on the generation of the temporal SI. The technique presented in the DISCOVER project consists in a linear interpolation of the object trajectory between the two closest available frames (the backward and forward). In order to improve the DISCOVER interpolation algorithm, we propose several techniques in the temporal as well as in the view domain. In temporal domain, since DISCOVER trajectory interpolation algorithm uses two frames, only constant velocity motion models can be considered. We propose to use four frames for the interpolation, in order to take into account also more complex motion models. We call this technique high order motion interpolation (HOMI). A simplified version of HOMI (FastHOMI) is also proposed, together with an extension using denser motion vector fields.

Moreover, in the temporal domain, when the GOP size is 4 or 8, if the DISCOVER codec is used, we have remarked that not all the WZFs have the same quality because of the different distance between the frames used for the interpolation. We propose that, as all the WZFs of the GOP are decoded, these frames can be used for re-estimating the low quality WZFs.

DISCOVER motion interpolation technique is also used in the view domain, in the case of aligned cameras. From a geometrical analysis of a multiview cameras system, the disparity of an object (*i.e.* its displacement from one view to the other one) is inversely proportional to its depth (*i.e.* the distance from the image plane of the camera). We propose to give priority to the larger values of disparity during the interpolation step, because the objects associated to these large values are closer to the cameras. This algorithm is called Interpolation with Priority to Large Disparity (IPLD). Both HOMI and IPLD improve

the rate-distortion performance of the DISCOVER codec.

In multiview distributed video coding, it is necessary to fuse the estimation of a WZF in the temporal domain and the estimation in the view domain. We propose a technique that, during the fusion process, excludes the inter-view estimation for the occluded regions. Moreover, we observed that for some sequences one of the two interpolations is much better than the other one and that, for these sequences, fusion does not improve the quality of the SI. In this case, it would be better to give to the channel decoder directly the best SI. Thus, we also propose an algorithm selects the SI among the temporal, the inter-view and the fused SI, based on the channel code rate necessary for correcting the WZF estimation.

The techniques proposed for SI generation have found an interesting application in robust video coding by multiple descriptions.

In the last part of this thesis, we focus our attention on Multiview Video plus Depth (MVD). MVD is a new format for multiview video, where each camera provides a texture and a depth map video. The depth map allows to synthesize new virtual views. This format is suitable for free view point television, where the user can choose the viewpoint that he/she wants to display. In the context of interactive multiview video streaming (IMVS), only the views demanded by the user are sent to him/her. Then, if the video is encoded with a classical hybrid codec, a problem occurs when the view switching falls in correspondence of a P-frame or a B-frame of the target view. Indeed, in this case the reference frames are not be available at the decoder side and all the frames of that GOP cannot be reconstructed. On the other hand, if DVC is applied to texture video and depth maps, this problem can be solved, because WZFs can be reconstructed independently from which side information is synthesized by the decoder side. We propose different algorithms for the estimation of the WZF in correspondence of the view switching. This estimation will be aided also by the information on the depth map. We propose also techniques for the estimation of depth WZF by also using information of the corresponding texture images.

Inspired by layered depth video (LDV) coding, we propose a new codec for DVC applied in the context of MVD. In layered video coding, only one central view (texture and depth) is sent along with the residual information of the other views. Indeed, depth map of the central view can be used for generating the other views, except for the occlusion areas. For the other views, only the occlusion areas are needed to be sent. In this thesis, we propose a new MVD codec in the context of DVC in which alternated views are INTRA coded and for the other ones only the occlusion areas are sent. In DSC paradigm, the sources cannot communicate each other. As a consequence, for each view, for which only the occluded areas are sent, we have to estimate independently their occlusion map, with the aid of the depth map. We illustrate how to generate these occlusion maps and how to reconstruct all the frames.

The structure of this manuscript is the following:

- Chapter 1 - Video Coding - In this chapter, we recall the basic concepts of video

compression: quantization, transform coding, motion estimation. The architecture of standard codecs such as MPEG-2 and H.264/AVC is illustrated.

- Chapter 2 - Multiview Video - In this chapter, we describe the context of multiview video systems, starting by the camera model and some basic concepts of 3D geometry. Disparity estimation algorithms are described and the most common architectures for multiview video compression are shown. Finally, we introduce the new format called "multi view video plus depth" along with a state of the art for the coding of the depth maps.

- Chapter 3 - Distributed Video Coding - In this chapter, we introduce the distributed source paradigm and how it is implemented in the context of monoview video coding. We describe the Stanford architecture and the DISCOVER codec focusing our attention on the side information generation. We provide a detailed state of the art for this problem. Finally, we show how the DISCOVER codec is extended in the context of multiview video. The different cameras configurations usually proposed in the literature are shown, and we provide a detailed state of the art for the construction of the SI by inter-view interpolation.

- Chapter 4 - Proposed Methods for side information construction - In this chapter, we illustrate the proposed algorithms for side information generation that improve the performance of the DISCOVER codec both in the context of monoview and multiview video. We shown the HOMI and IPLD algorithms and the techniques for fusion.

- Chapter 5 - Application of Distributed Video Coding to Multiview Video plus Depth - In this chapter, we propose different solutions for the estimation of the WZF when a view switching occurs in correspondence of a Wyner-Ziv Frame. Then, we will present a new algorithm for side information generation for depth map video. Finally, a new distributed architecture for MVD, inspired from Layered Depth Video, is shown.

- Appendix - In the appendix, we list all the results of SI generation improvement of HOMI (both temporal and inter-view interpolation).

Chapter 1

Video Coding

Contents

In this chapter, we review basic concepts of predictive video coding and give some details about the last in date compression standard, H.264/AVC, which will serve as basis for comparison in manuscript.

1.1 Video coding: basic concepts

Videos are three-dimensional signals $I_k(m, n)$: the pair (m, n) defines the space, while k the time. We refer only to digital videos, therefore the three variables m, n and k are discrete. Each image $I_k(m, n)$ is called frame and f frames per second are displayed for giving the illusion of fluid movement. The parameter f is called frame rate. Video signals require a large amount of data to be stored and transmitted. The goal of video compression is to reduce the size of the video file, with an acceptable loss in quality. Traditional video coding techniques focus on eliminating the redundant elements of the signal (psychophysical and statistical) [Bov00].

Video signals have large correlation in time and in space. Digital video coding seeks to minimize the redundancy in each domain. Usually, lossy coding is applied because of notable psychophysical redundancy present in video signal.

Obviously, there is a trade off between the total bit rate R used for coding the video and the distortion D between the coded and the original frames. In a typical use case, we want to minimize the distortion with a constraint on the total bit rate. This leads to a cost function as follows

$$J = D + \lambda R$$

where λ is a Lagrangian multiplier. This problem is referred to as rate-distortion optimization.

The coding rate R can be defined in bits per pixel (typically for images)

$$R_{\text{bpp}} = \frac{B}{NM}$$

where B is the numbers of bit used for compressing the video and $M \times N$ is the spatial resolution of the video. It is easy to convert R_{bpp} into the coding rate in bit per second

$$R_{\text{bps}} = R_{\text{bpp}} NMf$$

The distortion can be measured for each frame with an objective metric, *i.e.* a function of the original image I and of the decoded one $\widehat{I}$. Popular objective metrics are the Mean Square Error (MSE) and the Mean Absolute Error (MAE), defined as follows:

- Mean Square Error (MSE)

$$\text{MSE} = \frac{1}{MN} \sum_{n=1}^{N} \sum_{m=1}^{M} |I_k(m,n) - \widehat{I}_k(m,n)|^2$$

- Mean Absolute Error (MAE)

$$\text{MAE} = \frac{1}{MN} \sum_{m=1}^{N} \sum_{n=1}^{M} |I_k(m,n) - \widehat{I}_k(m,n)|$$

A measurement equivalent to the MSE is the Peak Signal-to-Noise Ratio (PSNR), derived from the MSE as

$$\text{PSNR} = 10 \log_{10} \left(\frac{(2^b - 1)^2}{\text{MSE}} \right)$$

where $(2^b - 1)$ is the maximum possible pixel value (usually $b = 8$).

In order to optimize the rate distortion performance, the scheme in Fig. 1.1 is usually adopted for compressing each frame [Bov00]: a cascade of transform, quantization and variable length coder (VLC). Transformation decorrelates data, quantization reduces

Figure 1.1: Transform coding.

the total bit rate and lossless coding performed by the VLC eliminates other statistical redundancies. In the following section, we separately analyze each tool.

1.2 Quantization

For analogue signals, quantization consists of the conversion of the data from a continuous to a discrete set. For digital signals, its goal is to further reduce the rate necessary for representing it. In general, quantization is a mapping from a set S to C, a discrete subset of cardinality N.

$$Q : x \in S \to C = \{y_1, \ldots, y_N\}$$

This means that the set S is divided into regions $R_i \subset \mathbb{R}$ for $i = 1 \ldots, N$, where $\cup_{i=1}^{N} R_i = S$. A region R_i is defined as

$$R_i = \{x \in S : Q(x) = y_i\}$$

Since N is the cardinality of C, we need $R = \log_2 N$ bits per each symbol of C. If $x \in R_i$, the reconstructed value y_i is associated to x. The quantization can be considered as a cascade of two operations: encoding and decoding. Encoding consists in mapping the value $x \in R_k$ to the index of the region to which it is associated: k in this case. Decoding consists in mapping the index k to the reconstruction value y_k.

If all the regions R_i have the same amplitude, and y_i is the center of the region R_i, the quantization is called uniform. In the case of non uniform quantization, Lloyd-Max algorithm gives the optimal regions and the optimal reconstruction values according to the statistics of the input signal X. Obviously, the range of values that is less probable is coarsely quantized than the most probable region. In this case, by supposing that the source can be modeled as a random variable X and that the distortion is defined as $\mathbb{E}[|X - \hat{X}|^2]$, at high resolution D can be approximated by

$$D = h_X \sigma_X^2 2^{-2R}$$

where σ_X^2 is the variance of X, and h_X is a shape factor that depends on the Probability Density Function (PDF) of X but not on its variance.

Now, we deal with the problem of rate allocation. Let us suppose that we want to quantize the components of a random vector $\mathbf{X} = (X_1, \ldots, X_k)$, and let R be the budget

of bits that we can use for quantizing this vector. If for each component we give $\frac{R}{k}$ bits, the distortion is

$$D = \sum_{i=1}^{k} h_i \sigma_i^2 2^{-2\frac{R}{k}} = k h_{AM} \sigma_{AM}^2 2^{-2\frac{R}{k}}$$

where h_{AM} and σ_{AM}^2 are the arithmetic means of $\{h_i\}_{i=1}^k$ and $\{\sigma_i^2\}_{i=1}^k$.

If we search for the optimal allocation between the different components (*i.e.* the one that minimizes the distortion under the constraint that the total bit rate is equal to R), we find that the distortion [Llo82] is

$$D_{opt} = k h_{GM} \sigma_{GM}^2 2^{-2\frac{R}{k}} \tag{1.1}$$

where h_{GM} and σ_{GM}^2 are the geometric mean of $\{h_i\}_{i=1}^k$ and $\{\sigma_i^2\}_{i=1}^k$. The rate for X_i is

$$R_i = \frac{R}{k} + \frac{1}{2} \log_2 \frac{\sigma_i^2 h_i}{\sigma_{GM}^2 h_{GM}}$$

The geometric mean of non negative numbers is always smaller or equal than the arithmetic mean. Therefore, by defining the coding gain as

$$G = \frac{D}{D_{opt}} = \frac{\sigma_{AM}^2 h_{AM}}{\sigma_{GM}^2 h_{GM}},$$

G will be always greater or equal than 1. The more the values $h_i \sigma_i^2$ are different from each other, the more the gain G is significant. Then, we can apply a transformation to the vector that we want to quantize for decorrelating the data. In fact, if the random variables X_i are identically distributed, the gain $G = 1$. If we are able to decorrelate then, quantizing the transformed vector gives a gain $G > 1$.

1.3 Predictive coding

The principle of the predictive coding is to encode the difference between the actual sample and its prediction. Let X_n be a random discrete process. Instead of quantizing directly X_n, we can quantize

$$Z_n \triangleq X_n - \widetilde{X}_n$$

where $\widetilde{X}_n$ is the estimation of X_n, called prediction. The sequence Z_n is called prediction error. In general,

$$\widetilde{X}_n = f(X_{n-1}, X_{n-2}, \ldots, X_{n-M}),$$

i.e. X_n is estimated by a function of the M previous samples. If we define $\widehat{Z}_n$ and $\widehat{X}_n$ as the reconstructed value of Z_n and X_n respectively, we obtain that

$$\mathbb{E}[|X_n - \widehat{X}_n|^2] = \mathbb{E}[|X_n - \widetilde{X}_n + \widetilde{X}_n - \widehat{X}_n|^2] = \mathbb{E}[|Z_n - \widehat{Z}_n|^2]$$

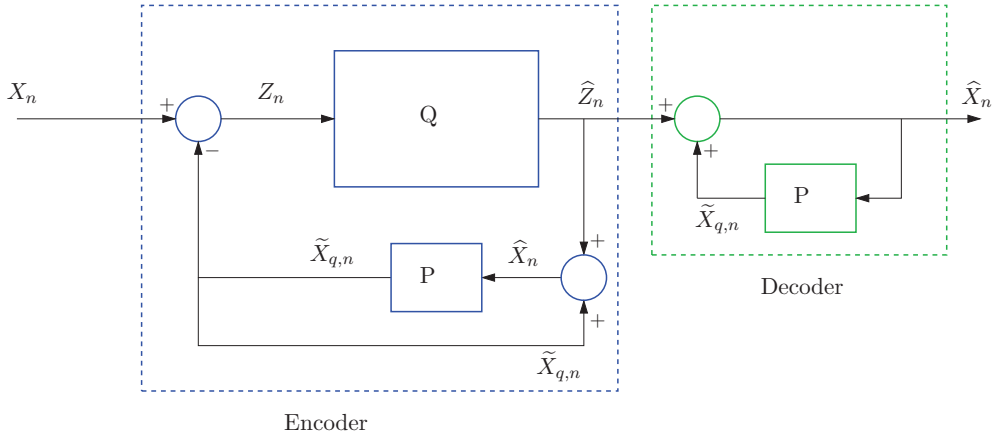


Figure 1.2: Predictive coding: the prediction $X_{q,n}$ is constructed from the quantized values $\widehat{X}_n$.

In this manner, we can reduce the distortion of X_n, by quantizing Z_n. In fact, if the variable X_n are correlated with each other, we expect that the variance of the prediction error Z_n is smaller than the one of X_n. Then, by using Eq. (1.1), if we want to obtain the same distortion, a smaller rate is necessary for encoding Z_n than for X_n. Remark that we have to produce the prediction $\widetilde{X}_n$ also at the decoder. For having the same prediction at the encoder and the decoder side, the encoder must predict X_n from the previously quantized values, as depicted in Fig. 1.2. Instead of considering $\widetilde{X}_n$ as prediction, we consider

$$\widetilde{X}_{q,n} = f(\widehat{X}_{n-1}, \widehat{X}_{n-2}, \ldots, \widehat{X}_{n-M}),$$

Thanks to predictive coding, we have a smaller variance of the data to be quantized. Then, at the same rate, we can obtain a smaller distortion if Z_n will be quantized instead of X_n.

1.3.1 Temporal predictive coding for video: motion estimation and compensation

Motion estimation and compensation are common techniques used to exploit the temporal redundancy of video signals. The goal is to predict the frame I_k from a neighboring frame $\widehat{I}_m$ already sent and decoded, by estimating and compensating the objects' motion. Then, instead of sending I_k, we send the estimated motion vectors for constructing the prediction and the prediction error. If the adjacent temporal frames are very similar, sending the estimated motion vectors and the prediction error is more convenient in terms of RD performance than sending the entire frame I_k.

For estimating the motion, we divide the frame I_k into blocks of size $M \times M$. Let us define by $B_k^{\mathbf{p}}$ the block of the frame I_k whose up-left pixel is $\mathbf{p}$ and let $B_k^{\mathbf{p}}(\mathbf{q})$ be the pixel $I_k(\mathbf{p}+\mathbf{q})$. We search in the reference image $\widehat{I}_m$ for the block $B_m^{\mathbf{p}+\mathbf{v}}$ that is the

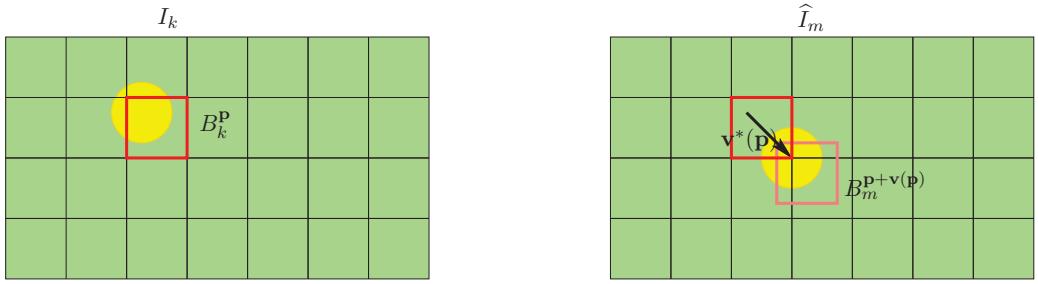


Figure 1.3: Motion estimation procedure.

most similar to $B_k^{\mathbf{p}}$, as represented in Fig. 1.3. The vector $\mathbf{v}$ is called motion vector. The search is usually constrained to be within a reasonable neighborhood so as to minimize the complexity of operation. Thus we add a constraint: the vector has to belong to the set W defined by

$$W = \{(v_x, v_y) \in \mathbb{Z}^2 : \max(|v_x|, |v_y|) \leq \rho\}$$

where ρ is the search window.

Formally, the block $B_k^{\mathbf{p}}$ within I_k has been moved to the position $\mathbf{p}+\mathbf{v}(\mathbf{p})$ in the frame $\widehat{I}_m$, obtained as

$$\mathbf{v}(\mathbf{p}) = \arg\min_{\mathbf{w}\in W} d(B_m^{\mathbf{p}+\mathbf{w}}, B_k^{\mathbf{p}})$$

where $d(\cdot,\cdot)$ can be defined by

- the Sum of Absolute Differences (SAD)

$$d(B_m^{\mathbf{s}}, B_k^{\mathbf{p}}) = \sum_{\mathbf{q}} |B_m^{\mathbf{s}}(\mathbf{q}) - B_k^{\mathbf{p}}(\mathbf{q})|$$

- SSD (Sum of Squared Differences)

$$d(B_m^{\mathbf{s}} B_k^{\mathbf{p}}) = \sum_{\mathbf{q}} |B_m^{\mathbf{s}}(\mathbf{q}) - B_k^{\mathbf{p}}(\mathbf{q})|^2$$

The motion vector field has to be sent to the decoder and it is separately encoded w.r.t. prediction error / I-frame. Since for motion vector field no loss of information is allowed, motion vectors are coded by a variable length coder. Using the motion vector $\mathbf{v}$, we can produce the prediction of I_k from the reconstructed frame $\widehat{I}_m$ as follows: each block $B_k^{\mathbf{p}}$ is predicted by $B_m^{\mathbf{p}+\mathbf{v}(\mathbf{p})}$. This operation is called motion compensation. Recapitulating, the encoder produces the prediction error and sends it to the decoder along with the motion vector field. The decoder produces the prediction from the motion vector field and the reference frame (that is supposed to be available) and by adding it to the prediction error, the reconstructed frame is obtained. If the frames are correlated in time domain, sending the motion vector field and the prediction error, instead of the actual frame, is

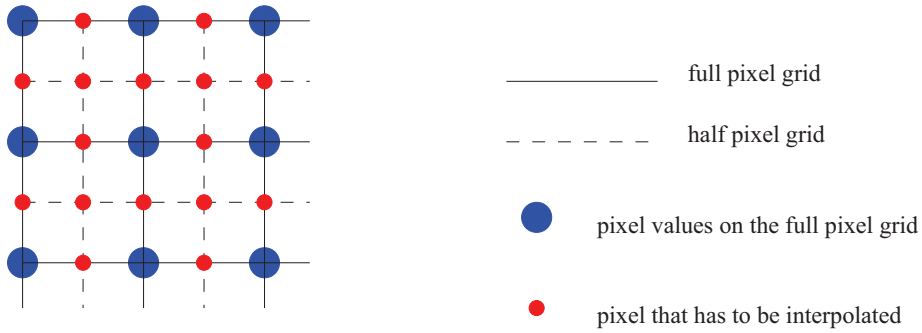


Figure 1.4: There is no pixel value at the half-pixel locations: interpolation is required to produce the pixel values at the half-pixel positions.

more efficient in terms of rate-distortion performance.

1.3.2 Motion Estimation with sub-pixel precision

The motion estimation described in the previous section is restricted to only integer pixel accuracy. However, an object often moves to a position that is not on the grid but between the pixels. By estimating the displacement at a finer resolution, we expect improved prediction. As shown in Fig. 1.4, since there is no pixel value at the half-pixel locations, interpolation is required. In general, if we want to implement the motion estimation with $1/N$-pixel precision, we need to expand the reconstructed reference image $\widehat{I}_m$ by a factor N and we obtain the image $\overline{I}_m$. For each block of the frame I_k, we search for the block in the frame $\overline{I}_m$ that is the most similar to it. The obtained vector that describes the displacement will be divided by N.

1.4 Transform

As already said, before quantizing the data, it is necessary to transform the data for decorrelating them and, then, for improving the rate distortion performance.

Let us suppose that we have a vector of k identically distributed random variables and let R be the total bit rate that we can allocate. If no transform is applied, the variance is the same for all components. Then, with the optimal allocation algorithm described in Section 1.2, we obtain a gain $G = 1$, because we are forced to devote R/k bits for each component. On the contrary, if we decorrelate the data by a linear transformation, the energy of the signal would be concentrated in a small number of components. Then, each component has a different variance and we obtain a gain $G > 1$, because we can devote a different number of bits to each component: more bits would be devoted to coefficients with greater variance. In order to recover the original signal, the transform has to be invertible. The optimal transform that decorrelates the data is the Karhunen-

Loeve transform (KLT). Optimal in the sense that for Gaussian variables, output data are uncorrelated. Thus it works well only for Gaussian variables. Moreover, KLT depends on the statistics of the data and if they are not known a priori, they must be estimated on-line and this estimation could be not very accurate. Alternatively, depending on the application, the matrix transformation has to be sent to the decoder. In practice, other transforms, such as the Discrete-Cosine Transform (DCT), are used. For natural images, it compacts the energy of the signal in low frequency components. Then, it can be used for decorrelating the data and differently from KLT, it does not depend on the data.

Moreover, it can be shown that the DCT is asymptotically optimal: therefore it is a very valuable tool for image compression.

1.5 Lossless coding

After transforming and quantizing the data, statistical redundancy still may occur. In order to reduce this redundancy, lossless coding is used. The Shannon source coding theorem establishes that a sequence of N i.i.d. random variables X_k with entropy $H(X)$ can be compressed into more than $NH(X)$ bits without information loss, as N tends to infinity. The entropy is defined by

$$H(X) = -\sum_{i=1}^{M} p_i \log_2(p_i)$$

where M is the cardinality of the alphabet of X and $p_i = \Pr(X_k = x_i)$.

The goal of lossless coding is to reduce the average bit rate for coding a sequence of symbols up to its entropy: for this reason, it is also called entropy coding. Another name to refer it is variable length coding (VLC). Indeed, for the most probable symbols we use short codewords, while for symbols with low probability we use longer codewords, in order to reduce the average bit rate. On the other hand, if the same sequence is compressed by fixed-length block codes, the average bit rate would be $r = \log_2 N$. Well known techniques for entropy coding are the Huffmann coding and arithmetic coding.

1.6 Architecture of a hybrid video codec

In this section, we describe a quite generic hybrid video coder, used by MPEG-x and ITU standards. The term hybrid is due to the fact that it removes the spatial and the temporal redundancy by two different methods. The temporal redundancy is removed by motion estimation and compensation and spatial redundancy by transform coding. In MPEG-x, the video sequence is organized in Group of Pictures (GOP). The first frame is always an I-frame: it is encoded independently from the other frames. Only spatial redundancy is exploited. The other frames are P-frames or B-frames. P-frames can be predicted from a

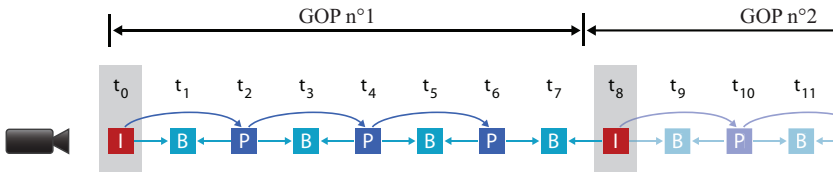


Figure 1.5: Example of GOP.

previous P/I-frames of the same GOP. B-frames can be predicted from a previous and a following P/I-frames of the same GOP. The distance between two I-frames is called GOP size. An example of GOP is depicted in Fig. 1.5.

The system is designed for two modes of operation:

- The **intraframe** mode. The frame I_k is encoded independently from the others. This picture is transformed by DCT, quantized and the quantized values are fed into a VLC coder. Let I_k^e be the encoded version of I_k. At the decoder, the encoded values are decompressed by the variable length decoder (VLD), inverse quantized, and inverse transformed. Let $\widehat{I}_k$ be the decoded version.

- the **interframe** mode. The frame I_k is predicted from a previously decoded frame $\widehat{I}_m$. The motion estimation (ME) creates motion vectors MV_k, that along with the previously reconstructed frame $\widehat{I}_m$ (stored into a frame buffer) are fed to the motion compensator (MC) to create the prediction P_k. Then, the prediction error $E_k = I_k - P_k$ is encoded, forming E_k^e. It is transmitted along with the encoded motion vectors MV_k. At the decoder side, the encoded prediction error E_k^e is fed into the VLD, the inverse quantizer and the IDCT, in order to obtain the decoded error $\widehat{E}_k$. The motion vectors MV_k operate on $\widehat{I}_m$ to generate the current prediction frame P_k. Finally, the prediction error $\widehat{E}_k$ is added to the prediction P_k for reconstructing the current frame $\widehat{I}_k$. The whole structure of the encoder and decoder is shown in Fig. 1.6 and 1.7.

The task of the channel buffer is to test the bit rate for adapting it to channel requirements. Then, it operates on the quantization steps thought the control block in this way:

- if $R < C$ (C is the channel capacity), we are underutilizing the channel. Then, we can reduce the quantization step. Since probably in this condition the video is slowly variable, the visual sight is more sensible the small spatial details. By reducing the quantization step, we improve the quality of these areas.

- if $R > C$, we need to reduce R, and so we have to increase the quantization step. Probably in this condition, the video is changing quickly. The human eyes are not able to appreciate the small spatial details and the reduction of the quantization step does not notably deteriorate the video quality.

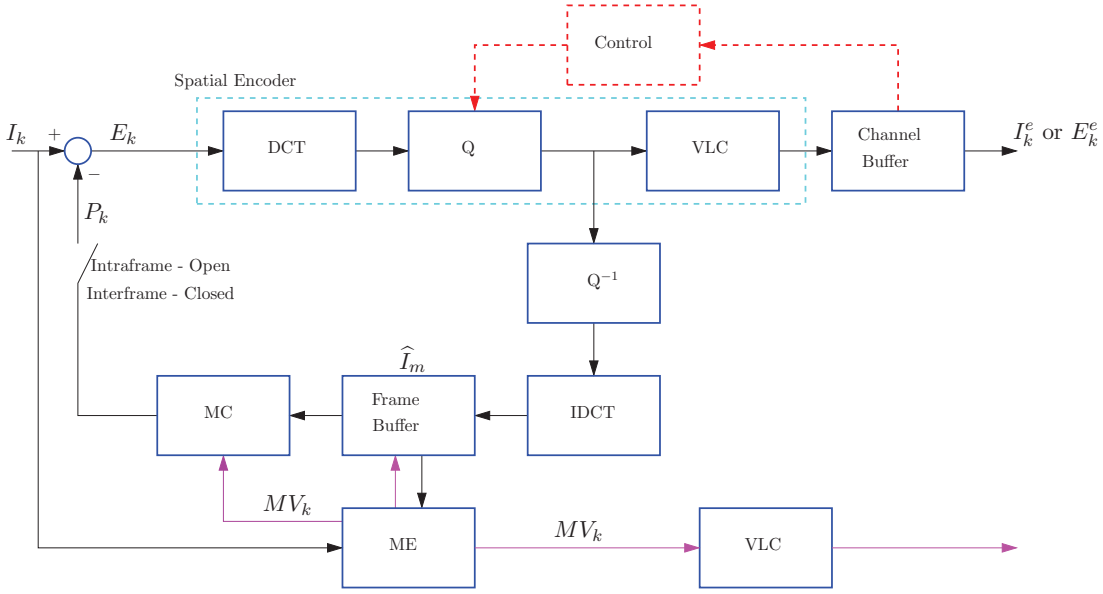


Figure 1.6: Hybrid video encoder compression system.

The video codec described in this section, has a complex encoder, but a very simple decoder. This asymmetry is well suited for broadcasting or for streaming video-on-demand systems where video is compressed once and decoded many times.

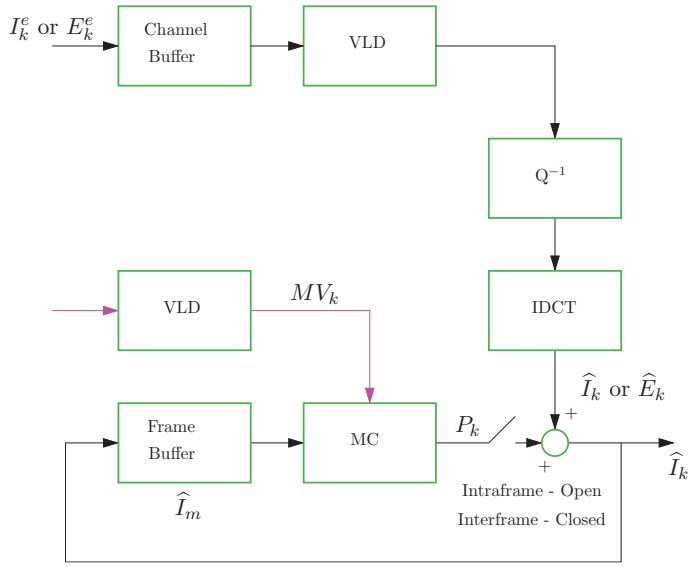


Figure 1.7: Hybrid video decoder compression system.

1.7 H.264/AVC Standard

H.264/AVC is a standard for video coding realized in 2003 [Joi07]. The structure of the codec is illustrated in Fig. 1.8, that consists in some modifications of the MPEG codec presented in the previous section.

In H.264/AVC a block of size 16×16 is called macroblock (MB). They are grouped in slices as illustrated in Fig. 1.9. We have three possible slices:

- I-slice: Only INTRA-prediction is possible
- P-slice: Skip, INTRA-prediction and INTER-prediction from past slices are possible
- B-slice: also prediction from future slices and from average of past and future slices are possible

Each macroblock in that slice can be coded in

- SKIP mode (no information is sent for this macroblock)
- INTRA modes (for spatial prediction)
- INTER modes (for temporal prediction)
- DIRECT modes (the actual MB is coded in the same mode and eventually with the same motion vector as the previous MB)

Each mode generates a different number of coded bits and a different distortion. If rate distortion optimization is performed, the goal of the encoder is to minimize the decoded

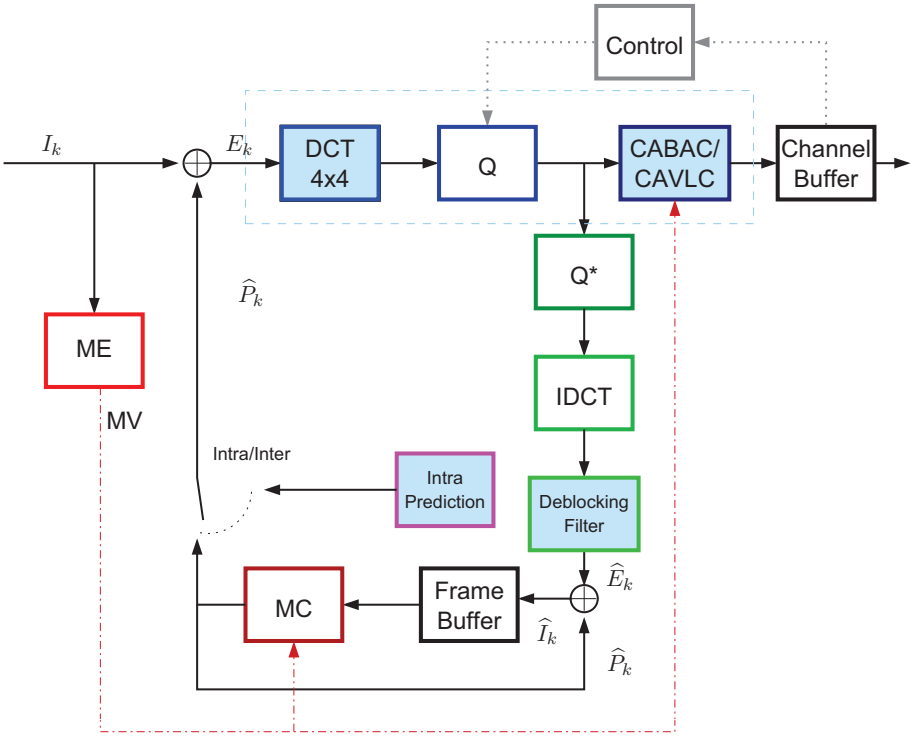


Figure 1.8: H.264/AVC Codec.

distortion D and to minimize the bit rate R. The encoder searches for the mode that minimizes the function $J = D + \lambda R$. Large values of λ tends to give more emphasis to the rate than to the distortion. In literature, an optimal value for λ depending on the quantization parameter (QP) has been obtained empirically [LGT05] as

$$\lambda = 0.852^{(QP-12)/3}$$

Differently from the MPEG codec, in H.264/AVC the spatial prediction is possible for I-slices. A block is predicted from the pixels of neighboring blocks in the same frame in the causal semi-plane. This is called INTRA prediction. Two classes of INTRA modes are supported: INTRA-16 $\times$ 16 and INTRA-4 $\times$ 4.

INTRA-16 $\times$ 16 modes are well suited for smooth image areas: in fact an uniform prediction is performed for all the macroblock. Four modes are possible: vertical, horizontal, planar and DC, listed in the following table

mode	
0 - vertical	Extrapolation from upper samples
1 - horizontal	Extrapolation from left samples
2 - DC	mean of upper and left-hand samples
4 - plane	A linear plane function is fitted to the upper and left-hand samples.

INTRA-4 $\times$ 4 mode consists in extrapolating or interpolating the neighboring pixels in blocks of 4 $\times$ 4 pixels. Nine prediction modes are possible:

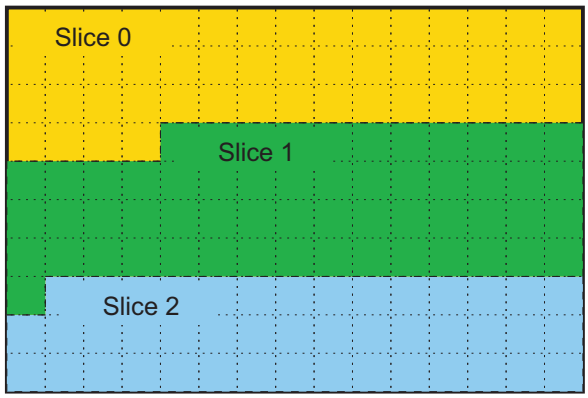


Figure 1.9: Slice in a frame for H.264/AVC.

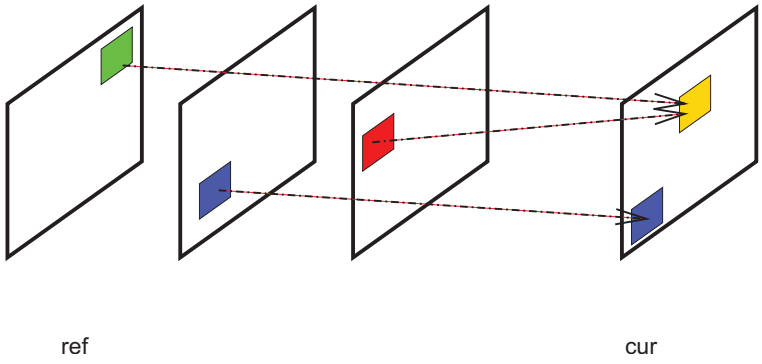


Figure 1.10: Multi-reference frame system of H.264/AVC: MBs of P-frames can be predicted from different frames. In this example the yellow MB in the current frame is predicted by two MBs from two different frames, while the blue one is predicted by a MB from a third one.

mode	
0 - vertical	The upper samples are extrapolated vertically
1 - horizontal	The left samples are extrapolated horizontally
2 - DC	All samples are predicted by the mean of upper and left-hand samples
3 - diagonal down-left	The samples are interpolated at 45 ° angle between lower-left and upper-right.
4 - diagonal down-right	The samples are interpolated at 45 ° angle down and to the right
5 - vertical-left	extrapolation at an angle approximatively 26.6 ° to the left of vertical
6 - horizontal-down	extrapolation at an angle approximatively 26.6 ° below horizontal
7 - vertical-right	Extrapolation or interpolation at an angle of approximatively 26.6 ° to the right of vertical
8 - vertical-right	Interpolation at an angle of approximatively 26.6 ° above horizontal

INTER modes exploits the temporal correlation between the frames. They are supported by P and B-slices. While for P-slices, only prediction from past frames is possible, for B-slices also prediction from future slices can be considered. INTER-16 × 16, 8 × 16, 16 × 8 and 8 × 8 modes are possible, depending on the block size used for the motion description. The possible block sizes are also shown in Fig. 1.11. If INTER-8 × 8 mode is chosen, an additional syntax element for each 8 × 8 macroblocks is used to specify if it is

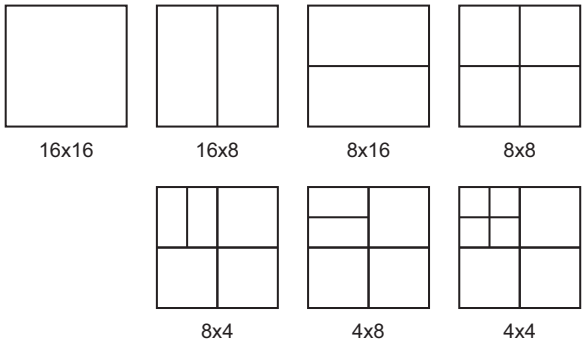


Figure 1.11: INTER-modes of H.264/AVC.

partitioned in 8×8, 8×4, 4×8, 4×4 blocks. The motion estimation is performed at a quarter of pixel precision. Multi-picture motion-compensated prediction is possible, which means that each macroblock of the P-slices can be estimated from one, more different reference frames or from a weighted sum of them as shown in Fig. 1.10. Then, a reference index parameter is sent for each block. In B-slices, INTER modes are possible with an extension on the possible reference frames. Indeed, reference pictures are divided into two lists 0 and 1. Different types of prediction are possible: list 0, list 1, bi-predictive, direct prediction. If list 0/1 is chosen, a frame of this list is chosen as a reference for motion estimation/compensation. If bi-predictive is chosen, the predicted block is a weighted sum of two blocks of two frames, one of list 0 and one of list 1. The DIRECT mode consists in choosing, for the actual macroblocks, the previously syntax elements (mode and vector).

The SKIP mode is possible for both P and B slices: no vector has to be sent for this mode. The prediction is the co-located block in the reference frame.

Another difference w.r.t previous standards is that the transform of H.264/AVC is applied to 4×4 blocks and not 8×8 blocks. The smaller block size leads to a significant reduction in ringing artifacts. The transform of H.264/AVC is very similar to the DCT but the coefficients are integer. A disadvantage of the DCT is that its coefficients of the transform are irrational numbers. If the direct and inverse transforms are implemented in different machines with different floating-point representations and rounding, we are not able to reconstruct identically the encoded signal.

Also a deblocking filter is added at the decoder after the inverse transform: it consists in an adaptive low pass filter applied to the reconstructed frames in order to remove some artifacts around the borders of macroblocks. These artifacts are due to the fact that each block is processed separately.

Another peculiarity of H.264/AVC are the two different methods for entropy coding: Context-Adaptive Variable Length Coding (CAVLC) and Context-Adaptive Binary Arithmetic Coding (CABAC) [Joi07]. For CAVLC, VLC tables for various syntax elements are switched depending on already transmitted syntax elements. Since the VLC tables are de-

signed to match the corresponding conditional statistics, the entropy coding performance is improved in comparison to schemes using a single VLC table.

Arithmetic coding allows to assign a non-integer number of bits to each symbol of the alphabet. Moreover, the adaptivity takes into account non-stationary symbol statistics. The performance of CABAC improves the one of CAVLC, but it requires a considerably greater computational effort for decoding the data than CAVLC.

Chapter 2

Multiview Video

Contents

In this chapter we introduce multiview video and its motivation. We analyze the geometry of a multiview video system. Then, techniques for stereo matching are shown. In order to exploit correlation between the views, the most common multiview video compression architectures are described. Finally, the new format multiview video plus depth (MVD) is described.

2.1 Introduction

In recent years, the improvements in video compression and transmission technologies are increasingly pushing towards a new paradigm called immersive communication, where users have the impression of being present at a real event. Immersive communication includes free-view point video, holography, 3D video and immersive teleconference. The

goal of these applications is to offer to users a more realistic experience than the traditional video (where viewers have only a passive way to observe the scene). Multiview video (MVV) is the first key of all these applications: a single scene is shot simultaneously by N cameras, arranged in different spatial positions. We obtain N 2D videos and the set of these signals is called multiview video.
Obviously, MVV requires a large amount of data to be stored, which increases proportionally with N. Luckily, there is a strong redundancy in time, view and spatial domain, that can be exploited to improve the compression. Techniques for compressing multiview video will be explored at the end of this chapter.
As already said, one of the most important applications of multiview video is 3D-TV. In human sight, two eyes perceive a 3D scene, but indeed they pick up only two 2D images; afterwards our brain processes them and deduces a 3D model. In a similar manner, in 3D-TV two or more cameras (arranged in different spatial positions) capture the same scene. Two selected views have to be displayed simultaneously on the same screen but each of the two eyes must see only one view. To achieve this purpose several technologies may be adopted: Anaglyph 3D, with special glasses (each lens filters one of the two components), autostereoscopic displays (where no glasses are required), polarized glasses, head-mounted display, and so on.

Another application of the multiview video is free-view point television (FTV): the user has the possibility to display the view that he/she wants. Moreover, by interpolating the N views, one can synthesize other views and give to the viewer the possibility to move freely inside the scene.

2.2 Pin-hole camera geometry

A pin-hole camera is a camera with a small aperture. It is like a box with a small hole in one side. The light passes through this point and it is projected on the opposite plane of this box. Then, a pin-hole camera is completely described by its optical center C and the image plane Π. The distance from C to Π is called focal length f. A point M in the space is projected onto the image plane by the line containing the point and the optical center. This operation is called perspective projection and the projected point is called image point. In Fig. 2.1, a scheme of a pin-hole camera is shown: C is the optical center, Π the image plane and $\mathbf{m}$ is the image point of M in the image plane.

Now, let us consider the two cameras depicted in Fig. 2.2. Let C_L and C_R be their optical centers and Π_L and Π_R the two image planes. The line connecting the two optical centers is called baseline. A point M and the optical centers C_L and C_R define a plane, called epipolar plane. The epipolar lines l_R and l_L are the intersections of the epipolar planes and the two image planes. The intersections between the baseline and the image planes are called epipoles. If $\mathbf{m}_l$ and $\mathbf{m}_r$ are the two image points of M for the left and the right cameras respectively, they lie on the two epipolar lines.

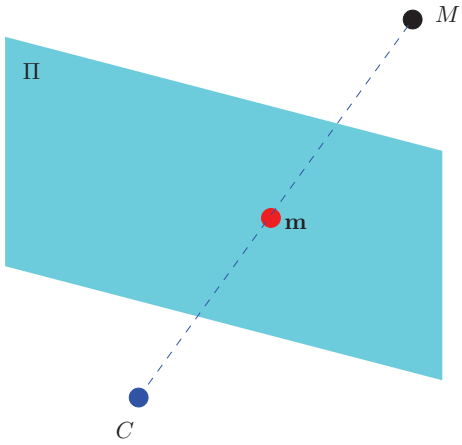


Figure 2.1: Geometry of a single camera.

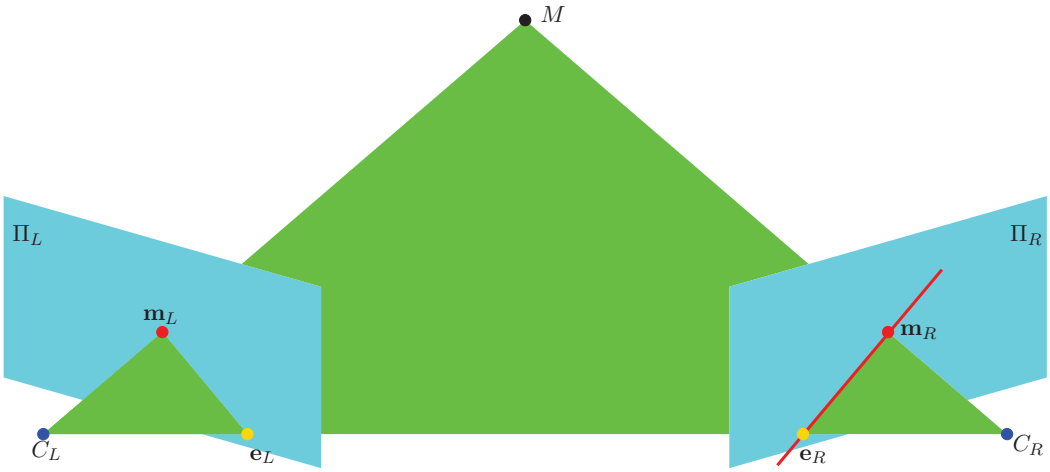


Figure 2.2: Geometry of a two camera system: the epipolar plane is green. The epipolar line is red. The epipoles e_L and e_R are yellow.

Two cameras are aligned if the two image planes are coplanar. As a consequence, the two epipolar lines are collinear and parallel to the horizontal image axis.

2.3 Stereo matching and disparity

The stereo matching is the correspondence between points of Π_L and Π_R (respectively, the left image plane and the right one) that are images of the same source point. The displacement of a projected point in one image with respect to the other is called disparity [WAH92].

Disparity estimation is an important field in stereo vision, that will be used in more applications, such as multiview compression and depth estimation. For a stereo system,

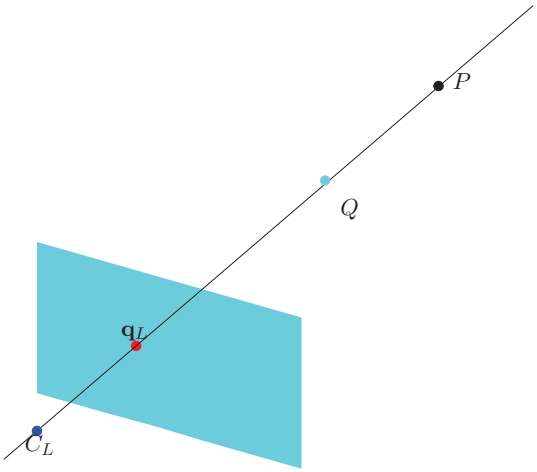


Figure 2.3: The object in Q is a visible and the object in P is not visible, because $\mathbf{p}(Q) \in \Pi$ and $\mathbf{p}(P) \in \Pi$, but $\| Q \| < \| P \|$.

we define two images: I_L is the left image and I_R the right one, defined respectively on Π_L and Π_R.

We define the function $\mathbf{p}(\cdot)$ [resp. $\mathbf{p}'(\cdot)$] as functions from $\mathbb{R}^3$ to Π_L [resp. Π_R], such that $\mathbf{m}_L = \mathbf{p}(M)$ is the perspective projection of M on Π_L and $\mathbf{m}_R = \mathbf{p}'(M)$ is the the perspective projection of M on Π_R.

Now, we define visible points and occlusion points.

Definition 1 (Visible point) *$X \in \mathbb{R}^3$ is a visible point for the left [resp. right] image if and only if $\mathbf{p}(X) \in \Pi_L$ [resp. $\mathbf{p}'(X) \in \Pi_R$] and there is no object in $Y \in \mathbb{R}^3$ with $\|Y\| \leq \|X\|$[1] such that $\mathbf{p}(X) = \mathbf{p}(Y)$ [resp. $\mathbf{p}'(X) = \mathbf{p}'(Y)$] .*

An example is shown in Fig. 2.3: if there are two objects (one in Q and the other in P), the object in Q is visible, but not the object in P.

Definition 2 (Occlusion point) *$X \in \mathbb{R}^3$ is an occlusion point if and only if X is visible for the left [resp. right] camera, but not for the right [resp. left] one.*

For example, in Fig. 2.4, P is an occlusion point. There are two objects (one in P and the other in Q): the object in P is visible on the right image plane but not on the left one.

Occlusion map $\mathcal{O}_L$ of the left image is defined as follows:

$$\mathcal{O}_L = \{\mathbf{p}(X) \in \Pi_L : \ X \text{ is visible for the left camera, but not for the right one } \}$$

and the occlusion map $\mathcal{O}_R$ of the right image is defined as:

$$\mathcal{O}_R = \{\mathbf{p}'(X) \in \Pi_R : \ X \text{ is visible for the right camera, but not for the left one } \}$$

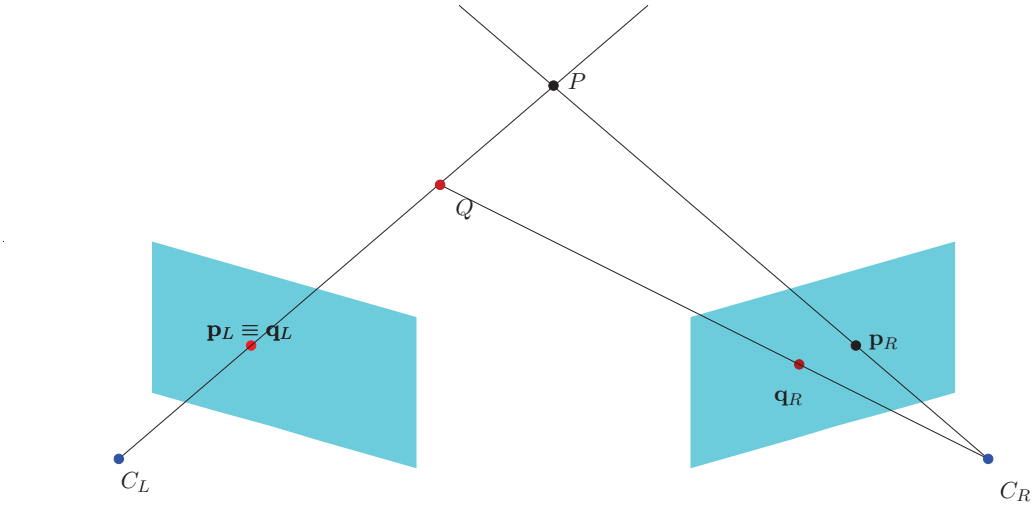


Figure 2.4: P is an occlusion point, because it is not visible for the left view, but it is visible for the right one.

Stereo matching is defined as a mapping $\boldsymbol{\kappa}$ between pixels that are images of the same source point, from the left image plane to the right one. Then $\boldsymbol{\kappa}$ is a function from $\Pi_L - \mathcal{O}_L$ to $\Pi_R - \mathcal{O}_R$, such that

$$\forall \mathbf{u} \in \Pi_L - \mathcal{O}_L, \quad \boldsymbol{\kappa}(\mathbf{u}) = \mathbf{p}'(X)$$

with $\mathbf{u} = \mathbf{p}(X)$; similarly, we can define $\boldsymbol{\kappa}'$ from the right image to the left one as a mapping from $\Pi_R - \mathcal{O}_R$ to $\Pi_L - \mathcal{O}_L$, such that

$$\forall \mathbf{u} \in \Pi_R - \mathcal{O}_R, \quad \boldsymbol{\kappa}'(\mathbf{u}) = \mathbf{p}(X)$$

with $\mathbf{u} = \mathbf{p}'(X)$; then the disparity, i.e. the displacement of a projected point on the right image with respect to the left one, is defined as

$$\mathbf{d}(\mathbf{u}) \triangleq \boldsymbol{\kappa}(\mathbf{u}) - \mathbf{u}$$

If we consider two aligned cameras as in Fig. 2.5, $\mathbf{d}(\mathbf{u})$ becomes a scalar function [SS02]. Indeed, let P be a point of $\mathbb{R}^3$ and $\mathbf{p}_L$ and $\mathbf{p}_R$ are the two perspective projections on the right and left image planes. Let us consider two coordinate systems: (x, y) for the left image plane and (x', y') for the right image plane. Then, let us define

$$\mathbf{p}_L \triangleq (x_L, y_L) \qquad \mathbf{p}_R \triangleq (x'_R, y'_R)$$

Since we have aligned cameras, we can assume that $x_L = x'_R$. The coordinates y_L and y'_R are referred, respectively, to the y axis and the y' axis. Under those conditions, the stereo

[1] The norm is computed in the coordination system centered in O_L [resp. O_R]

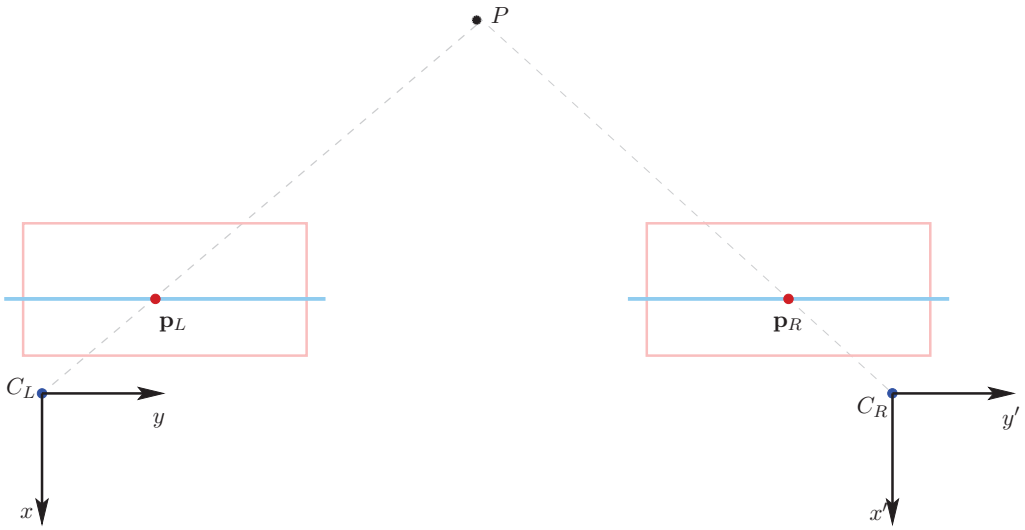


Figure 2.5: Two aligned cameras: P is the source point and $\mathbf{p}_L$ and $\mathbf{p}_R$ are the two perspective projections.

matching becomes a 1-D problem. So for each $(x_L, y_L) \in \Pi_L - \mathcal{O}_L$ $((x'_R, y'_R) \triangleq \boldsymbol{\kappa}(x_L, y_L))$, the disparity function is defined as [BBH03]

$$\begin{cases} x'_R = x_L \\ y'_R = y_L + d(x_L, y_L) \end{cases}$$

where $d(\cdot, \cdot)$ is a scalar function.

The points that are on the epipolar line in the left view have their correspondence points on the epipolar line in the right view. When the cameras are aligned, the two epipolar lines are parallel to baseline. It is very difficult to have two perfectly aligned cameras, but it can be we can take back to a system with two aligned cameras, by simply rectifying the two views. The rectification is equivalent to the projection of the two images over the two planes, that are obtained by turning the two image planes around their optical center till we have two coplanar planes as in Fig. 2.6. If we rectify the two views, the disparity becomes a scalar function.

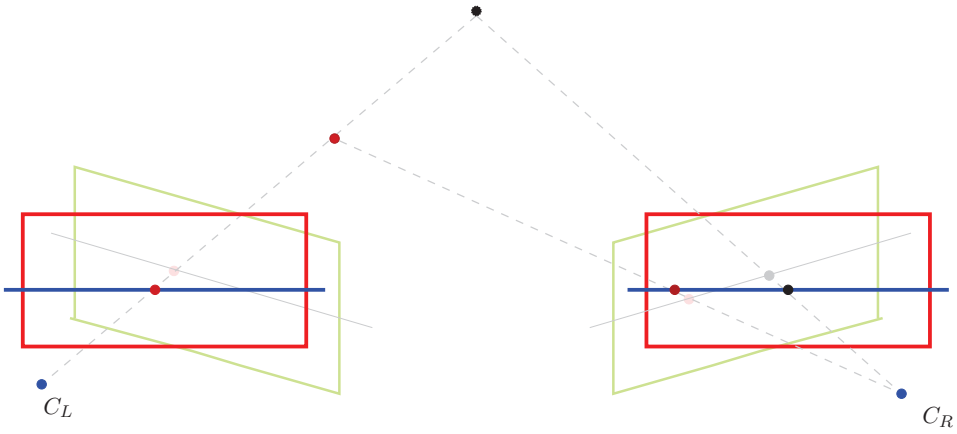


Figure 2.6: Rectification of the two images: it consists to project the two images on the two planes in red, that are obtained by turning the cameras around their optical center till one has two coplanar planes.

2.4 Disparity Estimation

In several applications, like the reconstruction of the 3D model of a scene, view rendering or multiview video compression, it is necessary to estimate the disparity function $\mathbf{d}(x, y)$ between two views. Let us suppose that we have two aligned cameras or that the two images have been rectified. Then, the disparity estimation is a 1-D problem. We can search correspondences only on the horizontal scanline. Two families of approaches exist for disparity estimation: local and global methods. Local methods compute disparity for each block of an image independently. The cost function is aggregated over that block, and the disparity level minimizing the cost function is selected for that block. On the other hand, global methods formulate the stereo problem as an energy function with smoothness constraints. With global methods, we can suppress some ambiguities caused by various factors as illumination variation and textureless regions.

2.4.1 Local methods

One of simplest local technique for disparity estimation is block matching. Let I_L and I_R be the left and the right images respectively. The disparity estimation problem is set up as

$$\widehat{d}(m, n) = \arg \min_{\delta \in [d_{min}, d_{max}]} C(m, n, \delta)$$

where $C(m, n, \delta)$ is a cost function measuring the correlation between the block in the left image centered in (m, n) and the one in the right image centered in $(m + \delta, n)$. There are many possibilities to define the cost function C:

- Sum of squared difference (SSD):

$$C(m,n,\delta) = \sum_{(u,v)\in\mathcal{S}(m,n)} |I_L(u,v) - I_R(u,v+\delta)|^2$$

- Sum of absolute difference (SAD):

$$C(m,n,\delta) = \sum_{(u,v)\in\mathcal{S}(m,n)} |I_L(u,v) - I_R(u,v+\delta)|$$

- Normalized cross correlation (NCC):

$$C(m,n,\delta) = \frac{\sum_{(u,v)\in\mathcal{S}(m,n)}(I_L(u,v)-\bar{I}_L)(I_R(u,v+\delta)-\bar{I}_R)}{\sqrt{\sum_{(u,v)\in M(m,n)}(I_L(u,v)-\bar{I}_L)^2 \sum_{(u,v)\in M(m,n)}(I_R(u,v+\delta)-\bar{I}_R)^2}}$$

  where $\bar{I}_i$ is the mean of $I_i(\cdot,\cdot)$

where $\mathcal{S}(m,n) = \{-W_x+m,\ldots,W_x+m\}\times\{-W_y+n,\ldots,W_y+n\}$ is a block of dimension $(2W_x+1)\times(2W_y+1)$ centered in (m,n). The metric that gives the best performance for disparity estimation is NCC [BBH03]. Since this metric is normalized w.r.t. variance of the blocks, disparity estimation does not suffer of luminance variation. Then, NCC is more robust than SAD/SSD.

Unfortunately, block matching has no sense if the cameras are no aligned. In this case we cannot suppose that the motion between the two cameras is a rigid translation, because we would have to take into account also rotation and perspective deformation.

2.4.2 Global methods

The goal of global methods is to find the function $d(m,n)$ that minimizes the following functional

$$J(d(m,n)) = J_{data}(d(m,n)) + \lambda J_{smooth}(d(m,n))$$

i.e. we search for

$$\widehat{d}(\cdot,\cdot) = \arg\min_{\delta(\cdot,\cdot)\in A} J(\delta(\cdot,\cdot)) \tag{2.1}$$

where A is the set of possible candidate disparity values.

The data term $J_{data}(\cdot)$ measures the distortion in $\mathcal{L}_2$ between the left image and the right one compensated by the disparity field $d(\cdot,\cdot)$. Then,

$$J_{data}(d(\cdot,\cdot)) = \sum_{(m,n)\in\Pi_L} |I_L(m,n) - I_R(m,n+d(m,n))|^2 \tag{2.2}$$

where Π_L is the image support.

The smoothness term $J_{smooth}(\cdot)$ takes into account some smoothness assumptions constructed explicitly for each application. The disparity maps should be smooth in homogeneous areas while keeping sharp edges. Then, the total variation regularization constraint is commonly used. Total variation of a field d on a continuous domain Ω is defined as the integral of the L_2-norm of the gradient of d. For a discrete $M \times N$ image, the total variation of a field d is defined as

$$\mathrm{TV}(d) = \sum_{m=0}^{N-2} \sum_{n=0}^{M-2} \sqrt{|d(m+1,n) - d(m,n)|^2 + |d(m,n+1) - d(m,n)|^2} + \tag{2.3}$$

$$+ \sum_{x=0}^{N-2} |d(m+1, M-1) - d(m, M-1)| + \sum_{n=0}^{M-2} |d(N-1, n+1) - d(N-1, n)|$$

A resolution method for finding a dense disparity estimation of $\widehat{d}(\cdot,\cdot)$ as in Eq. (2.1) was proposed by Miled et al. [MP06] under total variation constraints on Eq. (2.3).

In Fig. 2.7 the disparity map obtained by block matching techniques (in particular by minimizing NCC) and by the method proposed in [MP06] are shown. Obviously, the quality of estimated disparity is improved w.r.t. local methods, but the computational complexity for global methods has significantly increased w.r.t. local ones.

2.4.3 The Occlusion Problem

Estimating occlusion points after disparity estimation is very important because they are affected by mismatching. Then, the disparity values in these points are not reliable.

A very simple method for detecting occlusions is the one proposed by Fua et al. [Fua93b]. It consists in searching inconsistent disparities. Let $d_{LR}(m,n)$ be the disparity from the left to the right view. If the pixel in the left image in position (m,n) is displaced in position (m,n') in the right one, $d_{LR}(m,n)$ is defined as

$$d_{LR}(m,n) = n' - n$$

The first disparity fields link (m,n) to (m,n'). As a consequence, a coherent disparity field from the right to the left view $d_{R,L}(m,n')$ should point to (m,n). Therefore

$$d_{RL}(m,n') = d_{RL}(m, n + d_{LR}(m,n)) = n - n'$$

Adding up these equations, we have

$$d_{LR}(m,n) + d_{RL}(m, n + d_{LR}(m,n)) = 0 \quad \forall (m,n) \in \Pi_L - \mathcal{O}_L \quad (m,n') \in \Pi_R - \mathcal{O}_R$$

Then, let $\widehat{d}_{LR}(\cdot,\cdot)$ and $\widehat{d}_{LR}(\cdot,\cdot)$ be the two independent estimations of the disparity fields.

We can define a function $R(m,n)$ as

$$R(m,n) = |\widehat{d}_{LR}(m,n) + \widehat{d}_{RL}(m, n + d_{LR}(m,n))|$$

The functions $\widehat{d}_{LR}(\cdot,\cdot)$ and $\widehat{d}_{RL}(\cdot,\cdot)$ are defined on the whole image plane Π_L and Π_R, because a priori the occlusion maps $\mathcal{O}_L$ and $\mathcal{O}_R$ are not known. As a consequence also $R(m,n)$ is defined on the plane Π_L. However, we expect that

$$R(m,n) = \begin{cases} 0 & \text{if } (m,n) \in \Pi_L - \mathcal{O}_L \\ \beta(m,n) & \text{if } (m,n) \in \mathcal{O}_L \end{cases}$$

where $\beta(m,n) > 0$. Then, we can establish that all the points (m,n) such that $R(m,n) \neq 0$ are occlusion points. Since $\widehat{d}_{LR}(m,n)$ and $\widehat{d}_{RL}(m,n')$ are estimated fields, we can not assure that $R(m,n) > 0$, $\forall (m,n) \in \mathcal{O}_L$. The points (m,n) such as $R(m,n) > \tau$ are classified as occlusion points for the left view, where $\tau > 0$ is a threshold that tolerates small errors in the estimation of the two disparity fields. Then,

$$\mathcal{O}_L = \{(m,n) \in \Pi_L : R(m,n) > \tau\}$$

Likewise we can obtain $\mathcal{O}_R$. This method is called Left-Right Cross Consistency Check (LR-CC). It will be useful to define also left occlusion mask, that is a binary image defined as

$$O_L(m,n) = \begin{cases} 1 & \text{if } (m,n) \in \mathcal{O}_L \\ 0 & \text{otherwise} \end{cases}$$

Likewise we can define the right occlusion mask O_R. An example of occlusion detection is given in Fig. 2.8: at left the function $R(m,n)$ and at right the occlusion mask $O_L(m,n)$ estimated by LR-CC are shown.

2.4.4 Other problems in disparity estimation

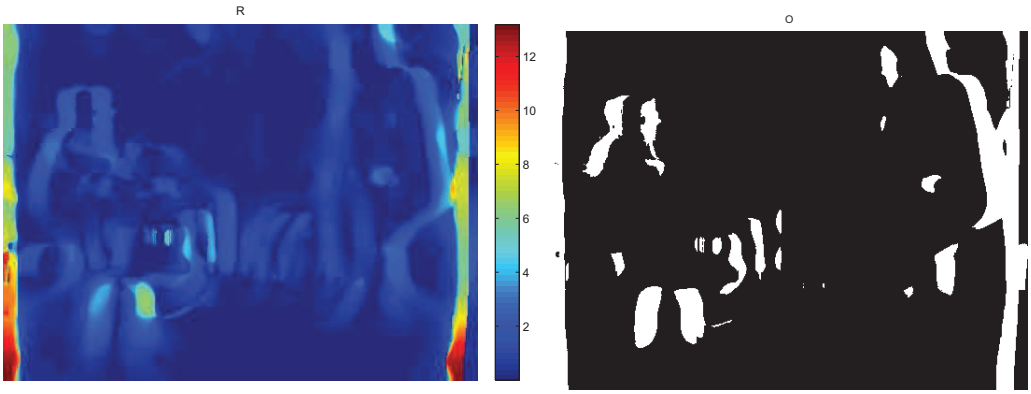
Besides occlusion, there are other problems that create errors and ambiguities in disparity estimation [CM89].

- **Photometric variation and noise**: the light reflected from the scene depends on the position of the camera relative to the scene. Moreover, the intensity of light can be different from the left view to the right one.
- **Repetitive Texture**: in this case there are many possible matching. Then we have ambiguities in disparity estimation.
- **Homogeneous areas**: in this case as well we have ambiguities in disparity estimation.

(a)

(b)

Figure 2.7: Disparity maps obtained by block matching techniques (a) and by dense methods proposed by [MP06] (b).



(a) (b)

Figure 2.8: The function $R(m, n)$, and at right the occlusion mask $O(m, n)$ obtained by thresholding it ($\tau = 2$.

2.5 Reconstruction of depth information

From disparity values we can obtain the distance of each pixel from the baseline. This distance is called depth. Let us consider Fig. 2.9 that represents two aligned cameras: Π_L and Π_R are the two image planes, f is the focal length and T is the length of the baseline. The points $\mathbf{p}_L$ and $\mathbf{p}_R$ are image points of the same source P, that is at distance z from the plane that includes Π_L and Π_R. We consider two coordinate systems (x, y) and (x', y') for the left and the right plane image. Then, we can define $\mathbf{p}_L = (x_L, y_L)$ and $\mathbf{p}_R = (x'_R, y'_R)$ in the two coordinate systems centered in C_L and C_R respectively. The triangles PC_LC_R and $P\mathbf{p}_R\mathbf{p}_L$ are similar (see Fig. 2.9), then

$$\frac{z}{T} = \frac{z - f}{T + y_L - y'_R}$$

Since the disparity d is equal $y'_R - y_L$, then

$$\frac{z}{T} = \frac{z - f}{T - d}$$

and so

$$z = f\frac{T}{d}$$

Then, depth z is inversely proportional to d. The farther the points are from the camera, the smaller is their disparity. This method of reconstruction of the depth is called triangulation.

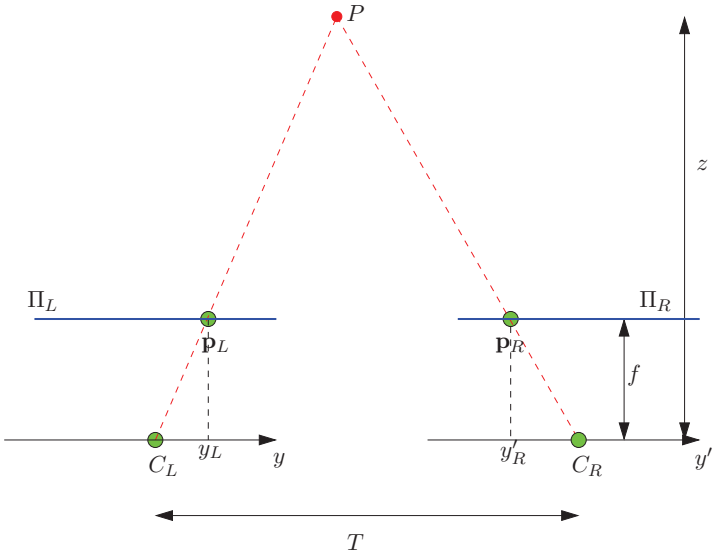


Figure 2.9: The search of the depth map: points farther from the camera have a smaller disparity.

2.6 Multiview video compression

Multiview video has a huge redundancy in temporal, spatial and inter-view domains, that can be exploited to compress the data. Usually, the compression techniques used in MPEG/H.264, are extended for multiview videos. A first solution for compressing multiview video is proposed by [MSMW07a]: each view is independently coded. This structure is called simulcast. In Fig. 2.10, an example of simulcast coding for four cameras and GOP size equal to 8 is shown. The advantage of this structure is that it is computationally very simple, because only temporal prediction is performed. On the other hand, it is not very efficient in terms of RD performance because inter-view correlation is not exploited. The advantage of this architecture is that random access is possible for each view.

A more efficient solution is employed in the multiview video coding extension of the H.264/AVC standard. In this extension [VWS11], [CWU+08], commonly referred to as H.264/MVC, two particular schemes are possible: fully hierarchical or view progressive. In the first architecture, both hierarchical temporal prediction and inter-view prediction are performed for all P/B-frames of all views except for the first view. Then, a predictive frame is not only predicted from temporally neighboring frames but also from corresponding frames in the adjacent views. Only the first view, called base view, is independently coded as in simulcast structure. This structure does not allow random access to views without completely decoding the entire GOP. A fully hierarchical coding for seven cameras and GOP size equal to 8 is depicted in Fig. 2.11. In that structure, for estimating the first frame of view 5, we need view 3, that depends on view 1.

In the view progressive architecture, the first view is coded independently from the

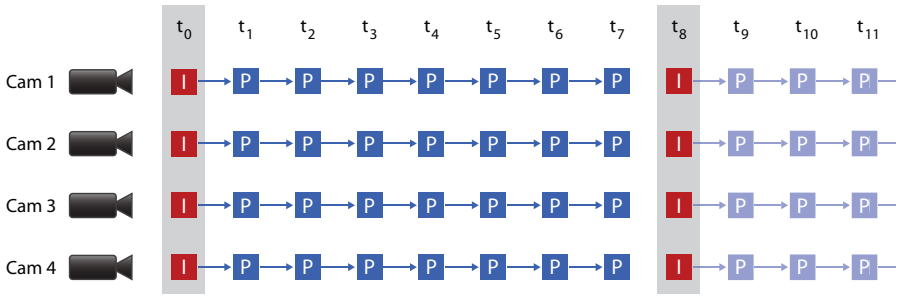


Figure 2.10: Simulcast coding structure.

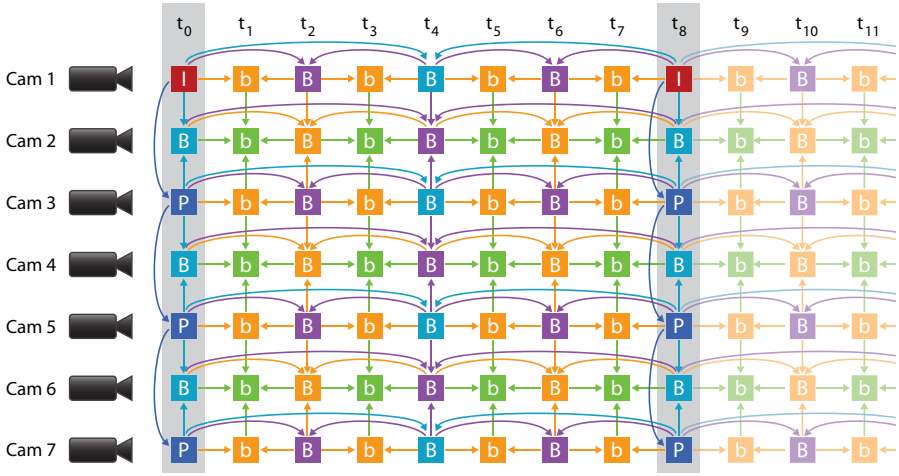


Figure 2.11: Fully Hierarchical architecture of H.264/MVC

other ones. For the other views, inter-view prediction is exploited only along the view axis. The frames in other view predicted from the I-frames at the same instant are called V-frames. For P-frames of all views, only temporal prediction is performed. An example of view progressive coding is illustrated in Fig. 2.12. This is a trade off between the simulcast and the fully hierarchical structure because with this structure random access is simpler: only key frames need references from other views to be decoded. P-frames can be decoded without access to other views. For prediction, the block matching algorithm are indifferently used for motion or disparity estimation.

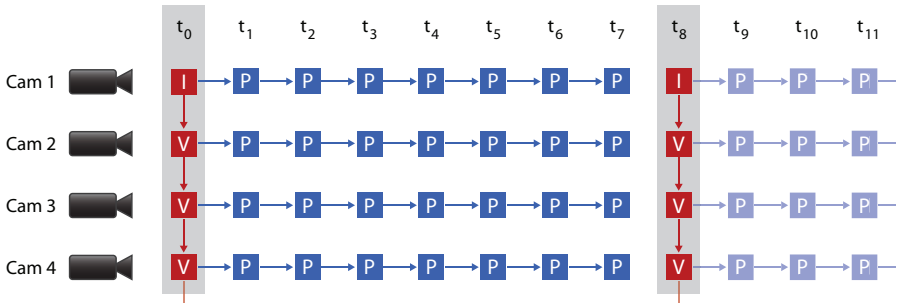


Figure 2.12: View progressive architecture of H.264/MVC.

2.7 Multiview video plus depth (MVD)

Multiview video is gathering increasing attention, thanks to the availability of acquisition and rendering systems, paving the way to a large number of interesting research topics and applications, such as 3D and free viewpoint TV (FTV) [TTFY11]. In this context, the problem of efficient compression is more urgent than ever, in sight of the huge amount of data required to represent 3D video. In particular, a representation which is gaining attention from the research and industry communities is based on the use of depth map in addition to standard texture. This allows an easy generation of more virtual views with depth-image-based rendering (DIBR) methods [Feh03], shown in the section 2.7.1. Moreover, systems with multiple views, each one with its own depth map are already envisaged, since they promise a simple implementation of 3D and FTV [FKdB+02]. This approach is called multiview video-plus depth (MVD). In this case, in addition to the texture cameras, the scene is captured also by range cameras, that are able to recover the depth of each object, such as structured-light 3D scanners or time-of-flight (ToF) cameras. In structured-light 3D scanners [KVG06], a projector illuminates the scene. This projected light produces lines of illumination that appears distorted from each camera, and can be used for an exact geometric reconstruction by active triangulation method. On the other hand, ToF cameras [LCH12] compute depths of the object by measuring the time-of-flight of a light signal for each point of the image. An example of texture plus depth is shown in Fig. 2.13.

In this manner, depth image based rendering (DIBR) algorithm can be directly applied in order to estimate other views, without passing through disparity estimation algorithm. Then, this allows us to reduce the total number of the cameras, because intermediary views can be reconstructed from depth information. Moreover, the bit rate necessary for encoding a depth map is about 1/4 of a texture image at the same resolution [OH06]. Thus the RD performance is improved w.r.t. MVC encoder.

Color video and the depth data can be coded by a simulcast coding structure (where each view is coded independently) or a multiview-coding structure (where the views are

Figure 2.13: In MVD, at each instant the output of each camera is the color image and the corresponding 8-bit depth map.

jointly coded). These two architectures are not very efficient for depth maps, which have different characteristics from texture images. Therefore, different algorithms that take into account the characteristics of depth maps have been proposed. The principal characteristic of depth maps is that they present sharp edges. In [SKN+10], the integer transform of H.264/AVC is replaced by an Edge-Adaptive Transform (EAT), that takes into account the discontinuities due to the object boundaries: so, the energy can be compacted better than by DCT. However, the decision of using DCT or EAT has to be taken block by block. In [IDPP08], an adaptive wavelet lifting is applied to the depth maps: short filters are applied to blocks that contain edges in order to better reconstruct the discontinuities, and long filters are applied to homogeneous areas. In [KOL+09], [LWP11] and [MMS+08] the dependence between depth maps and the associated texture is exploited. If there is a strong temporal correlation between colocated blocks in texture images, then the SKIP mode is used also for the colocated blocks in the depth maps [LWP11]. This information is not transmitted, because it can be recovered at the decoder side. In [KOL+09], when the SKIP mode is applied on the texture, it is extended also to depth maps. In [SPW+10] a new mode called "Motion Sharing" is added: the motion vector field estimated for texture information is used also for depth maps. The depth maps are not directly displayed. Then, it would be better to take into account the quality of synthesized frame and not of the depth maps: Valenzise et al. [VCO+12] study the sensibility of the quality of synthesized image to the depth error coding. They define "Don't Care Region" the range where depth values can vary without affecting sensibility the quality of synthesized frame. In the last months, AVC and HEVC-based 3D Video (3DV) solutions presented by Moving Picture Experts Group (MPEG) in March 2012 [Han12] are under test. Texture and depth data are independently coded by H.264/AVC or HEVC but since depth maps have large homogeneous areas, they are downsampled to quarter resolution before encoding. The goal of that standard is providing a 3D video format with the ability to cope with different types of displays, including autostereoscopic ones.

2.7.1 DIBR

In this section we give some remarks about DIBR algorithm. Depth-image based rendering (DIBR) is an algorithm that allows to generate a virtual view or to estimate another real view from the texture image, the corresponding depth map and the camera parameters of the actual and the target view. Let (u_1, v_1) be the projection of a point in the camera 1 image plane. The camera 2 is the target view (it can be a virtual or a real camera). The position (u_2, v_2) of the same point on the camera 2 can be obtained by the knowledge of the intrinsic and extrinsic parameters of the two cameras, by the following equations:

$$\begin{pmatrix} u_2' \\ v_2' \\ w_2' \end{pmatrix} = \mathbf{K}_2\mathbf{R}_2\mathbf{K}_1^{-1} \begin{pmatrix} u_1 \\ v_1 \\ 1 \end{pmatrix} \lambda_1(u_1, v_1) + \mathbf{K}_2\mathbf{t}_2 \tag{2.4}$$

where $\mathbf{K}_i$, $\mathbf{R}_i$ and $\mathbf{t}_i$ are, respectively, the intrinsic matrix, the orthogonal rotation matrix and the translation vector of the i-th view, $(u_2', v_2', w_2')^\top$ are the homogeneous coordinates of the point (u_2, v_2). In turn, $\lambda_1(u_1, v_1)$ is equal to:

$$\lambda_1 = \frac{z(u_1, v_1)}{c} \quad \text{with} \quad \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \mathbf{K}_1^{-1} \begin{pmatrix} u_1 \\ v_1 \\ 1 \end{pmatrix} \tag{2.5}$$

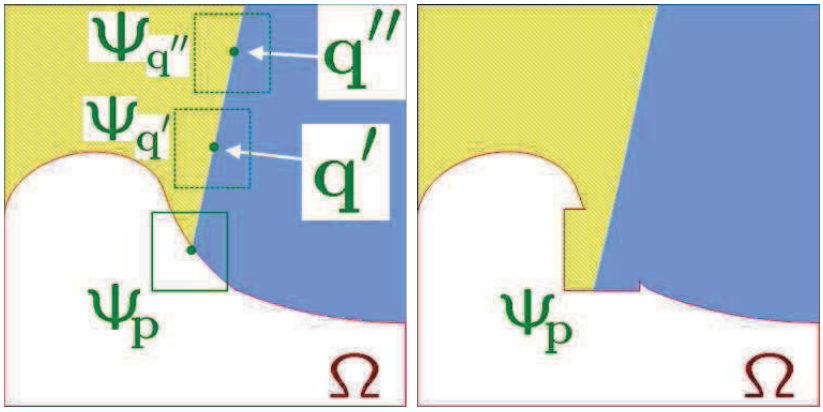
where $z(u_1, v_1)$ is the value of the depth map of the first view in the point (u_1, v_1).

Finally, the homogeneous coordinates are converted into non homogeneous coordinates as $(u_2, v_2) = (u_2'/w_2', v_2'/w_2')$. This allows to obtain a synthesis of the depth map and texture for the second view given the first one. Obviously, there are points of the second view that are not visible in the first view (occlusion points): this causes the presence of occlusion areas in the synthesized image. These areas can be filled by inpainting techniques [MSCB00]. In particular, the depth maps, due to their smooth nature, can be inpainted by Bertalmio technique [BBS01], which is based on isotropic diffusion by using the Navier-Stokes and fluid dynamics equations. On the other hand, for the texture image, one of the most popular techniques is the Criminisi inpainting [CPT04]: in this algorithm the texture is inpainted in the isophote direction.

Efros and Leung [EL99] proved that it is very important the order in which the image is inpainted. Let Ω be the occlusion areas. In Criminisi inpainting, a patch $\Psi_{\mathbf{p}}$ centered in $\mathbf{p}$ (where $\mathbf{p}$ is on the boundary $\partial\Omega$ of the occlusion areas, as in Fig. 2.14(a)) is inpainted with priority $P(\mathbf{p})$, given by the product of two terms

$$P(\mathbf{p}) = C(\mathbf{p})D(\mathbf{p})$$

where C represents the percentage of non missing pixels in the patches and D gives special priority to the isophote direction. Isophotes are level lines of equal intensity. Given an



(a) (b)

Figure 2.14: Criminisi inpainting: the patch $\Psi_{\mathbf{p}}$ is filled by the patch centered in $\mathbf{q}'$ or $\mathbf{q}''$

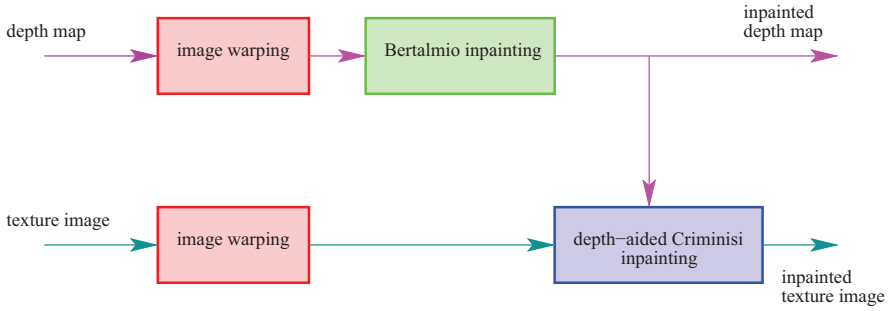


Figure 2.15: The novel view synthesis proposed by [DPP10]

image I, the direction of the isophotes is indicated by the gradient $\nabla I = (\partial I/\partial x, \partial I/\partial y)$. Then, the patches with the minimum number of missing pixels and the patch centered in a point, where the normal to $\partial\Omega$ coincides with the isophote direction, are privileged.

Then, in order to fill the patch $\Psi_{\mathbf{p}}$, the block (in the non occluded areas in the texture) that has the minimum distance to $\Psi_{\mathbf{p}}$ in terms of SAD/SSD is chosen, as the block centred in $\mathbf{q}'$ or $\mathbf{q}''$ in Fig. 2.14(b).

In the case of MVD, Daribo and Pesquet-Popescu [DPP10] modified the Criminisi inpainting by introducing a term related to the depth: in fact, the depth helps to distinguish if a pixel belongs to foreground or background and give higher priority to the patch that overlays at the same depth level. This will allow a better quality for both the texture image and the depth map. Then, the priority is given by the product of three terms

$$P(\mathbf{p}) = C(\mathbf{p})D(\mathbf{p})L(\mathbf{p})$$

where $L(\mathbf{p})$ represents the inverse variance of the depth patch centered in $\mathbf{p}$. Then, a block matching is performed on the not-occluded areas: in order to search the block that is more similar to $\Psi_{\mathbf{p}}$, the cost function is given by the weighted sum of SSD/SAD between the

(a)

(b)

Figure 2.16: Occlusion areas filled by Criminisi inpainting (a) and by inpainting proposed by [DPP10] (b)

texture patch and the SSD/SAD between the corresponding depth patch for giving more priority to the patches that have the same depth level. The block digram of the algorithm proposed by [DPP10] is depicted in Fig. 2.15. A visual example of the performance of these two methods for an image is shown in Fig. 2.16: the annoying artifacts due to the Criminisi inpainting in Fig. 2.16(a) are completely removed by the method proposed in [DPP10], as shown in Fig. 2.16(b).

Chapter 3

Distributed Video Coding

Contents

In this chapter, we introduce some concepts about distributed source coding (DSC). According to the Wyner-Ziv theorem, under some constraints, there is no loss in terms of RD performance between distributed and joint source coding. This is attractive for video coding. Indeed, if we encode separately correlated frames of a video but we decode them jointly, we can attain the same RD performance of classical video coding. In multiview video, if we want to avoid inter-camera communication, we can separately encode each view and decode them jointly without a significant loss in RD performance. DSC is also suitable for monoview video systems that require a low-complexity encoder. Most of the computational effort for the encoder in standard video architectures is due to the motion estimation. If we can encode adjacent temporal frames independently, we can move the motion estimation unit from the encoder to the decoder, while still attaining the same RD performance of classical joint video coding (under some constraints).

3.1 Distributed Source Coding

A source X, modeled as a discrete random variable with entropy $H(X)$, according to first Shannon Theorem [CT91], can be encoded and decoded without loss of information if the rate used for encoding it is greater than or equal to its entropy $H(X)$.

Two dependent sources, modeled as two discrete random variables X and Y, can be encoded and decoded jointly without loss of information, if the total bit rate used for coding them is larger than or equal to their joint entropy $H(X,Y)$, defined as

$$H(X,Y) = -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} f_{XY}(x,y)\log_2(f_{XY}(x,y))$$

where $f_{XY}(x,y)$ is the joint PDF between X and Y. $\mathcal{X}$ and $\mathcal{Y}$ are the alphabet of X and Y, respectively.

Slepian and Wolf [SW73] established that we can attain the same rate of joint coding, even if we separately encode the sources, provided that we decode them jointly. If X and Y are separately encoded, according to the first Shannon Theorem, we should expect they can be encoded and recovered without loss of information if $R_X + R_Y \geq H(X) + H(Y)$. However, according to the Slepian-Wolf Theorem [SW73], a total rate $R_X + R_Y \geq H(X,Y)$, with $R_X \geq H(X|Y)$ and $R_Y \geq H(Y|X)$, is sufficient, even for separated encoding of dependent sources, provided that we decode them jointly. The conditional entropy are

$$H(X|Y) = -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} f_{XY}(x,y)\log_2(f_{X|Y}(x|y))$$

$$H(Y|X) = -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} f_{XY}(x,y)\log_2(f_{Y|X}(y|x))$$

where $f_{X|Y}(x|y)$ is the PDF of X given Y. Encoding separately two correlated sources and decoding them jointly is referred to as distributed source coding.

3.1.1 Definitions

Let $(X_1,Y_1),(X_2,Y_2),\ldots,(X_k,Y_k)$ be a sequence of joint distributed discrete random variables. Each pair (X_i,Y_i) is i.i.d. with joint probability mass function (PMF) $f_{XY}(x,y)$. Let us define the random vectors:

$$\mathbf{X}^k = (X_1, X_2, \ldots, X_k)$$

$$\mathbf{Y}^k = (Y_1, Y_2, \ldots, Y_k)$$

and let $\mathcal{X}^n$ and $\mathcal{Y}^k$ be the source space of $\mathbf{X}^k$ and $\mathbf{Y}^k$, respectively.
A $((2^{kR_X}, 2^{kR_Y}), k)$ **distributed source code** for the joint source (X,Y) consists of two

encoder maps

$$f_X : \mathcal{X}^k \longrightarrow \{1, 2, \ldots, 2^{kR_X}\}$$

$$f_Y : \mathcal{Y}^k \longrightarrow \{1, 2, \ldots, 2^{kR_Y}\}$$

and a decoder map

$$g : \{1, 2, \ldots, 2^{kR_X}\} \times \{1, 2, \ldots, 2^{kR_Y}\} \longrightarrow \mathcal{X}^k \times \mathcal{Y}^k$$

This is equivalent to say that we divide the source space $\mathcal{X}^k$ in 2^{kR_X} subsets and $\mathcal{Y}^k$ in 2^{kR_Y} subsets: we send to the decoder $f_X(\mathbf{X}^k)$ and $f_Y(\mathbf{Y}^k)$, the indices corresponding to $\mathbf{X}^k$ and $\mathbf{Y}^k$. Obviously, for representing all the indexes, we need $k(R_X + R_Y)$ bits. The pair (R_X, R_Y) is called the **rate pair of the code**.
The **probability of error** for a distributed source code is defined as

$$P_e^{(k)} = \Pr(g(f_X(\mathbf{X}^k), f_Y(\mathbf{Y}^k)) \neq (\mathbf{X}^k, \mathbf{Y}^k))$$

A rate pair (R_X, R_Y) is said to be **achievable** for a distributed source if there exists a code $((2^{kR_X}, 2^{kR_Y}), k)$ with probability of error $P_e^{(k)} \longrightarrow 0$ for $k \longrightarrow +\infty$. The **achievable rate region** is the intersection of the sets of achievable rates. If two random variables are separately encoded and decoded, the achievable rate region is the set of (R_X, R_Y), such that $R_X \geq H(X)$ and $R_Y \geq H(Y)$ (as depicted in red in Fig. 3.1). For joint coding, the achievable region is the set of (R_X, R_Y), such that $R_X + R_Y \geq H(X, Y)$ (as depicted in green in Fig. 3.1).

3.1.2 Slepian-Wolf Theorem

Theorem 1 (Slepian-Wolf Theorem) *For the distributed source coding problem of the pair (X, Y) with joint pmf $f_{XY}(x, y)$, the achievable rate region (depicted in Fig. 3.3) is given by*

$$R_X \geq H(X|Y)$$

$$R_Y \geq H(Y|X)$$

$$R_X + R_Y \geq H(X, Y)$$

Remark that the achievable rate region for distributed video coding is larger than the rate region for separately encoding and decoding of the two variables, but smaller than the one obtained for jointly coding (see comparison in Fig. 3.1).

If we allowed R_Y bits to describe Y, how many bits R_X are required to describe X? If $R_Y > H(Y)$, then Y can be described perfectly, and by the result of Slepian-Wolf coding, $R_X = H(X|Y)$ bits suffice to describe X.

Slepian-Wolf theorem does not suggest how we can encode X and Y: it is not constructive. A practical implementation of DSC is the Slepian-Wolf coder: that is a closed

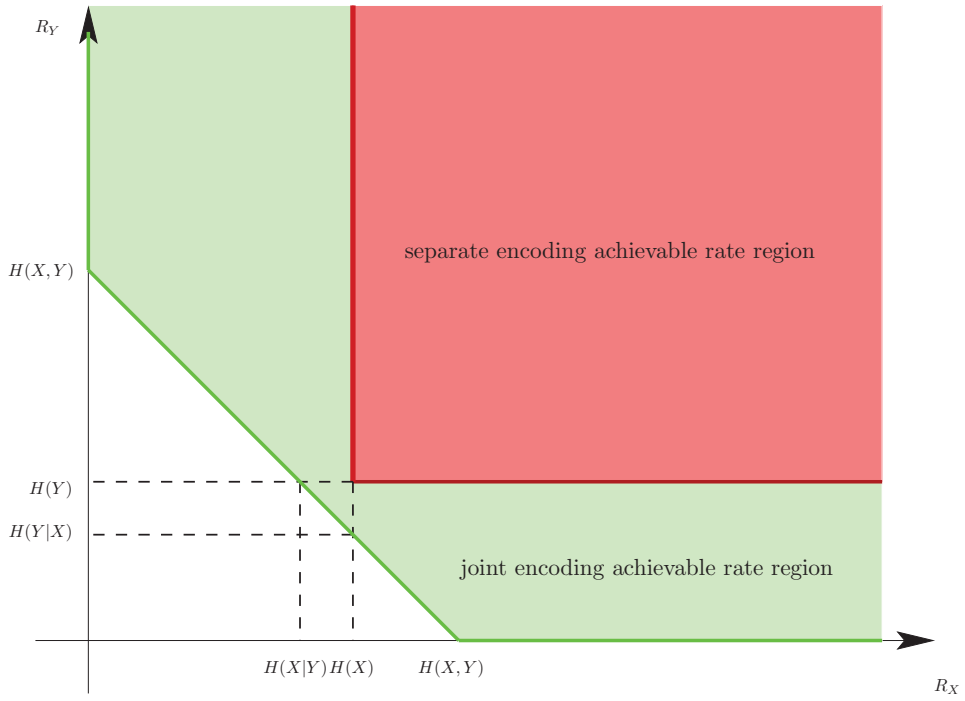


Figure 3.1: Rate region for separate encoding and decoding (the inferior bound is in red) and for joint classical coding (the lower bound is in green)

kin to channel coding. In fact, it consists simply in a systematic channel encoder and the corresponding channel decoder as in Fig. 3.4. Let $\mathbf{X}^k$ and $\mathbf{Y}^k$ be two correlated binary[1] sequences, represented as two row vectors. They are encoded separately but only $\mathbf{X}^k$ has to be recovered.

The sequence $\mathbf{Y}^k$ is called side information: it is encoded separately and it is available only at the decoder. Since $\mathbf{X}^k$ and $\mathbf{Y}^k$ are correlated, we can define the following bit error sequence as

$$\Delta \triangleq \mathbf{X}^k \oplus \mathbf{Y}^k$$

that consists of zeros, except for some ones that mark the positions where $\mathbf{X}^k$ and $\mathbf{Y}^k$ differ. Therefore $\mathbf{Y}^k$ can be regarded as noisy version of $\mathbf{X}^k$. Then, it is possible to avoid sending $\mathbf{X}^k$ but only its parity bits along with the side information [GPT+07]. The coding channel code is supposed to be systematic, in order to separate information and parity bits. A code $\mathbb{C}_p(k,n)$ maps each sequence of k bits to a sequence of n bits, with $k > n$ Then, for a code $\mathbb{C}_p(k,n)$, the generator matrix is defined as

$$\mathbf{G} = [\mathbb{I}_k|\mathbf{P}]$$

[1] If they are not binary sequences, we can divide them in bit planes and repeat the following procedure for each bit plane

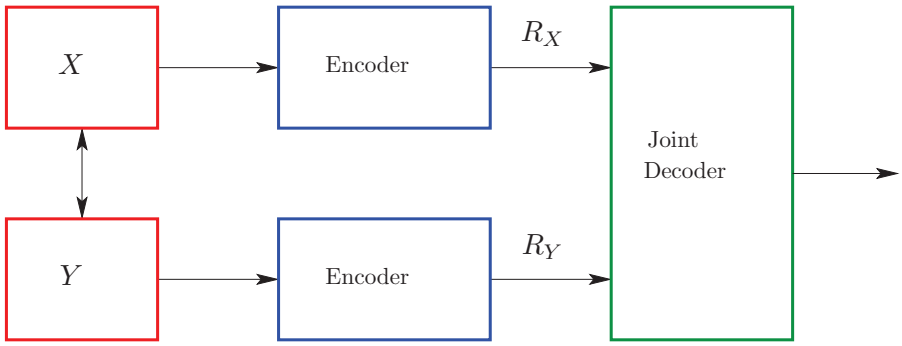


Figure 3.2: Distributed source coding scheme

where $\mathbb{I}_k$ is the $k \times k$ identity matrix. The vector $\mathbf{X}_p = \mathbf{X}^k\mathbf{P}$ is sent. At the decoder, we have the side information $\mathbf{Y}^k$ and the parity bits $\mathbf{X}_p$. Then, $\mathbf{X}_p$ is a sequence of $k-n$ bits.

By considering $\mathbf{Y}^k$ as a noisy version of $\mathbf{X}^k$, we are able to recover $\mathbf{X}^k$ from $\mathbf{Y}^k$ thanks to the parity bits.

In an alternative interpretation [WZ76, GARRM05], the alphabet of $\mathcal{X}$ is divided into kR_X regions and the encoder sends the index of the region that $\mathbf{X}^k$ belongs to. The receiver decodes by choosing the codeword in that region that is most probable in sight of the side information. In this approach, we consider an $\mathbb{C}_s(n-k, n)$, where the parity check matrix is defined as

$$\mathbf{H} = [\mathbf{Q}^T|\mathbb{I}_k]$$

In this case, we send the syndrome $\mathbf{s} = \mathbf{X}^k\mathbf{H}$. If the two codes C_s and C_p are such that $\mathbf{H} = \mathbf{P}$, then $\mathbf{s} = \mathbf{X_p}$ and the two approaches become equivalent.

3.1.3 Wyner-Ziv Theorem

While Slepian-Wolf theorem is about lossless coding, the problem of distributed *lossy* coding with side information was solved by Wyner and Ziv [WZ76]. The source X is sent to the decoder. The decoder has access to the side information, represented by the random variable Y, and has to reconstruct X. The reconstruction of the source is called $\widehat{X}$.

Theorem 2 *Let $D = \mathbb{E}[d(X, \hat{X})]$ be the maximum distortion accepted at the decoder; let $R^*_{X|Y}(D)$ be the achievable rate with distributed coding, and $R_{X|Y}(D)$ the achievable rate for classical joint coding, when the side information Y is available also at the encoder and let $R_X(D)$ be the achievable rate for encoding X without SI, at distortion D. Then*

$$R_{X|Y}(D) \leq R^*_{X|Y}(D) \leq R_X(D)$$

This result is unsurprising, because the achievable rate for encoding X independently is obviously larger than the achievable rate with distributed coding. On the other hand, for DSC the rate would be larger than the achievable rate with classical joint coding of X and

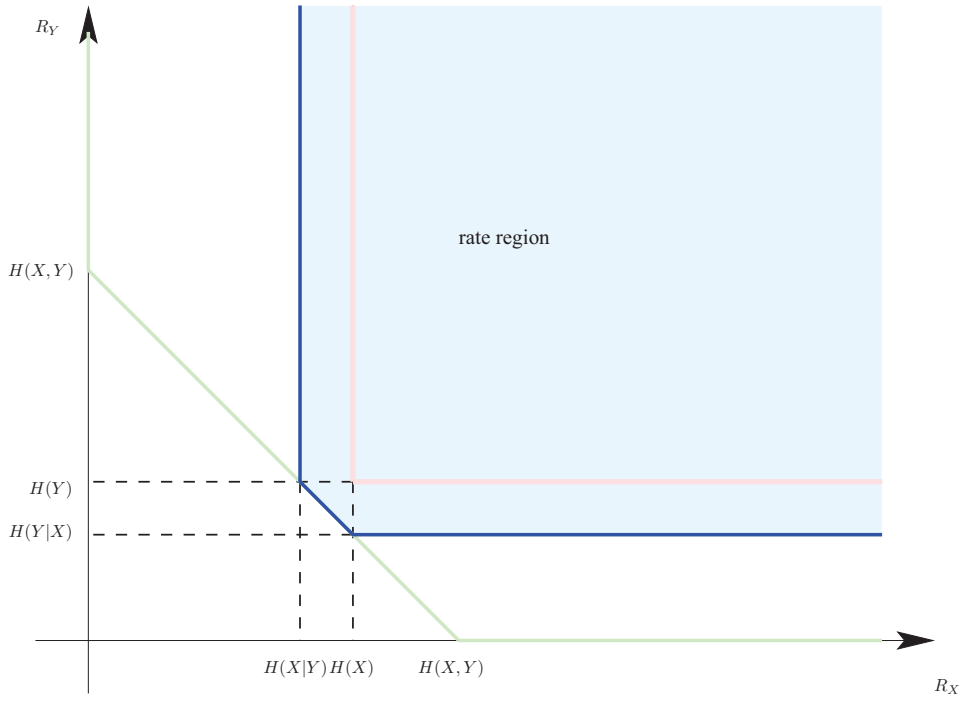


Figure 3.3: Achievable rate region for Slepian-Wolf theorem

Y. Indeed, the most interesting result is the following theorem, proved by Wyner and Ziv:

Theorem 3 (Wyner-Ziv Theorem) *If D is the MSE error (i.e. $D = \mathbb{E}[|X - \hat{X}|^2]$) and X and Y are joint Gaussian variables, then*

$$R_{X|Y}(D) = R^*_{X|Y}(D)$$

This means that there is no rate increase between the classical joint encoding and distributed coding of the two sources for a given quality, and conversely, there is no quality loss for a given rate. The Wyner-Ziv theorem gives only a theoretical result, just like the Slepian-Wolf Theorem. The Wyner-Ziv coder is a practical implementation for lossy DSC. It is depicted in Fig. 3.5: the encoder consists in a quantizer, followed by the Slepian-Wolf encoder, that carries out the parity bits. The quantizer divides the source space $\mathcal{X}$ in 2^{R_S} quantization cells; one codeword is associated to each cell, thus constructing the source codebook. The Slepian-Wolf encoder divides the source codebook in 2^R cosets and each cosets groups 2^{R_C} codewords, and computes the index of the coset containing the source codeword. Then $2^{R_C}2^R = 2^{R_S}$, hence $R_C + R = R_S$. The coset index I is sent at a rate $R = R_S - R_C \leq R_S$. The Wyner-Ziv decoder consists in the channel decoder, followed by a Minimum Mean Square Error (MMSE) estimator. The channel decoder receives the coset index I (we suppose noiseless channel). Knowing the side information Y, the SW

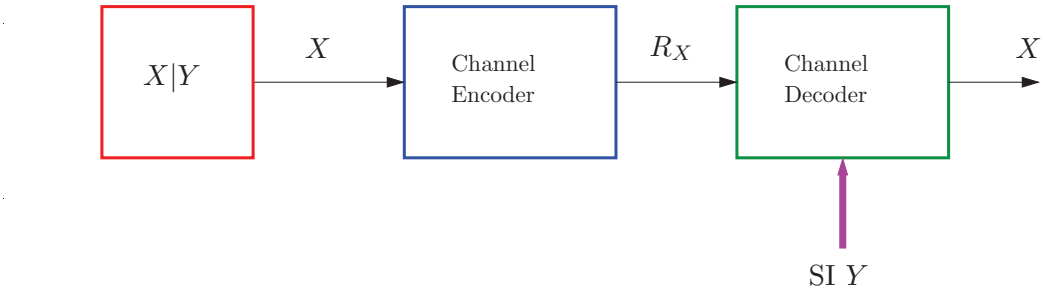


Figure 3.4: Slepian-Wolf codec

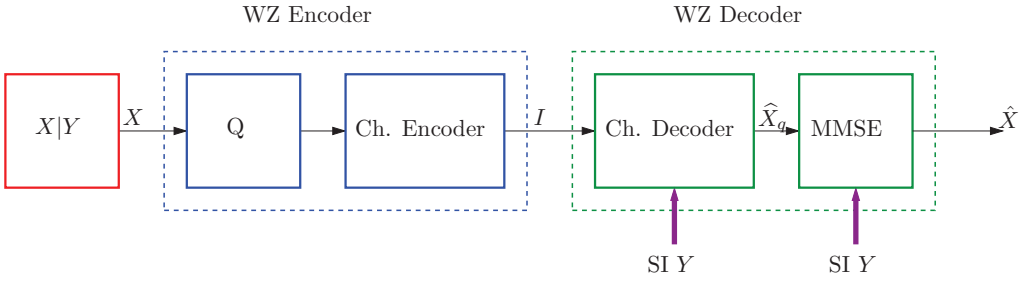


Figure 3.5: Wyner Zyv codec

decoder decides for the sequence $\hat{X}_q$ which belongs to coset I and which is more similar to Y. Then, the MMSE searches for $\widehat{X}$, by computing the estimation

$$\hat{X} = \arg\min_x \mathbb{E}[(X-x)^2|\hat{X}_q, Y]$$

Therefore, it is very important to estimate the joint statistics between X and Y. There are several models proposed in the literature for estimating it and they will be explored in the next sections.

3.2 Distributed Video Coding

In distributed source coding paradigm, the computational complexity is moved from the encoder to the decoder. For video coding this is very important for applications that require a low-complexity encoder, such as wireless video sensors for surveillance, wireless PC cameras, mobile camera phones, disposable video cameras, and networked camcorders, and also in multiview video coding, when no intercamera communication is possible.

According to the WZ theorem, correlated frames of an input video sequence can be quantized and coded independently with minimum loss in terms of RD performance, provided that they are jointly decoded. This, in principle, avoids motion estimation at the encoder and has the effect of a complexity shift from the encoder to the decoder.

In literature two main architectures for DVC exist: the Stanford architecture [AZG02]

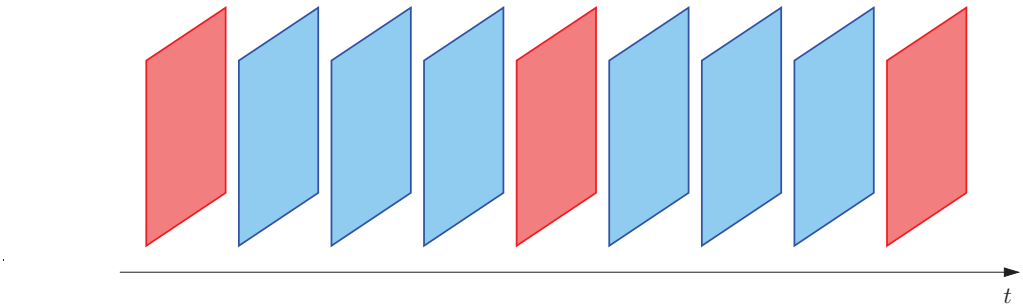


Figure 3.6: Example of video sequence (GOP size = 4): the KF are in red, while the WZF are in blue

and the PRISM architecture [PMR07].

In the architecture proposed by [AZG02], called Stanford, the video is structured into groups of pictures (GOP), in which selected frames (for example one out of N frames for a GOP size equal to N), called key frames (KF), are intra coded (typically using a standard codec such as H.264 INTRA) and intermediate frames are Wyner-Ziv coded and they are called Wyner-Ziv Frames (WZF). An example with $N = 4$ is shown in Fig. 3.6. Each WZ frame is encoded independently from the other ones: the WZ data are DCT transformed, quantized and fed into a systematic channel encoder. At the output of this encoder we have the information bits and the parity bits. Only the latter are stored in a buffer. The information bits are discarded. Initially, the encoder sends only a subset of the parity bits upon request of the decoder. The number of parity bits sent is estimated at the decoder and sent from the decoder to the encoder via the feedback channel. The decoder generates the SI via motion-compensated temporal interpolation of key frames. If the estimated bit error rate (BER) at the output of the SW decoder exceeds a given value (typically 10^{-3}), more parity bits are requested to the encoder via the feedback channel. The MMSE estimates of the quantized values are computed given the received quantization index and the SI. Then, the inverse transform is carried out on the resulting DCT coefficients, so that the reconstructed image is obtained. The complete scheme is shown in Fig. 3.7.

The second main architecture used for DVC is PRISM. Differently from the Stanford architecture, it implements a block based codec, but the structures of these two codecs are very similar. Each 16×16 block is transformed by DCT. The low-frequency coefficients are compressed using a trellis code. The high frequency coefficients are INTRA coded. The encoder also sends a cyclic redundancy check (CRC) of the quantized coefficients in order to help the decoder for the construction of the Side Information. The bit rate allocation is decided for each block depending on the difference w.r.t the co-located block of the previous frame. This makes the PRISM architecture not really distributed because the difference between consecutive frames is needed for rate allocation. Due to this drawback, the PRISM architecture is not very used in literature.

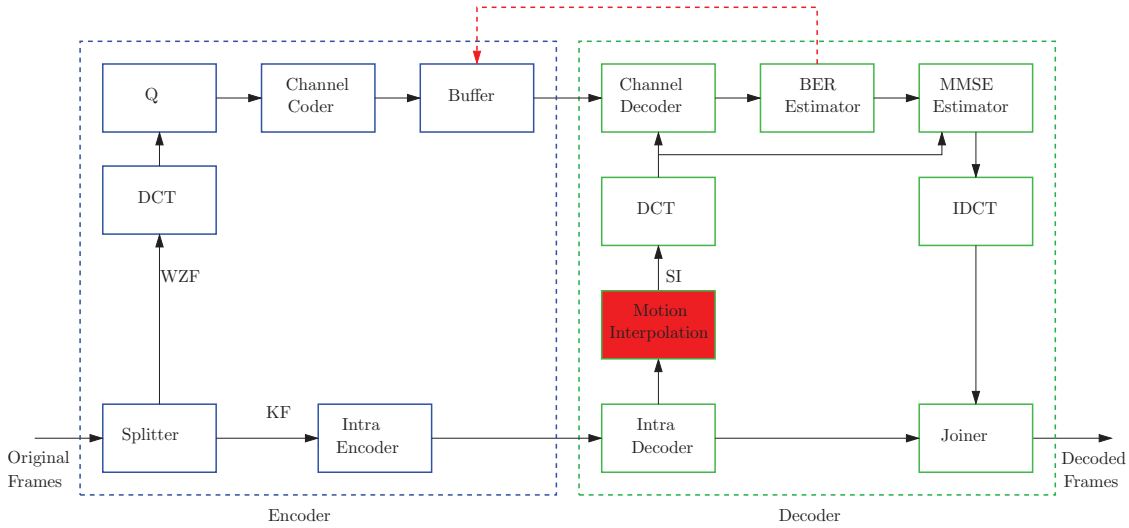


Figure 3.7: Wyner-Ziv video coder

3.3 DISCOVER

In this section we describe the tools proposed by the European project DISCOVER (Distributed Video Coding for Video Services) [AAD+07]. The DISCOVER project implements essentially the Stanford codec depicted in Fig. 3.7, by specifying encoding and decoding techniques. In recent years, it has become the reference method for DVC. In the following sections, we will describe separately each tool.

3.3.1 Encoder structure

Transform and quantization

Each WZF is transformed by a 4×4 DCT. Then, we obtain 16 bands: the first is called DC band and the other ones AC bands. Each DCT band b (where $b = 1, \ldots, 16$) is quantized over 2^{M_b} levels, where M_b depends on the band b and a quantization index QI that depends on the QP used for encoding the KFs (as in Tab. 3.1).

The DC coefficients are supposed to be in the range $[0, 2^{11}]$. They are uniformly quantized with a step size of 2^{11-M_1}. The AC coefficients are compressed by a dead-zone quantized with double zero interval. The band b is supposed to be in the range $[-V_b, +V_b)$. The value V_b has to be sent to the decoder for the reconstruction. The quantization step size is then

$$W_b = \frac{2V_b}{2^{M_b}}$$

Finally, the quantization indices of each DCT band are organized in bit planes and sent to the channel encoder.

band	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
QI = 1	16	8	8	0	0	0	0	0	0	0	0	0	0	0	0	0
QI = 2	32	8	8	0	0	0	0	0	0	0	0	0	0	0	0	0
QI = 3	32	8	8	4	4	4	0	0	0	0	0	0	0	0	0	0
QI = 4	32	16	16	8	8	8	4	4	4	4	0	0	0	0	0	0
QI = 5	32	16	16	8	8	8	4	4	4	4	4	4	4	0	0	0
QI = 6	64	16	16	8	8	8	8	8	8	8	4	4	4	4	4	0
QI = 7	64	32	32	16	16	16	8	8	8	8	4	4	4	4	4	0
QI = 8	128	64	64	32	32	32	16	16	16	16	8	8	8	4	4	0

Table 3.1: Quantization table for WZF: for each QI, we found the number of quantization levels for each DCTband

Channel encoder

The channel encoder performs a systematic channel coding of the quantization indeces for each bit plane of each DCT band. Usually a rate-compatible Low-Density Parity-Check (LDPC) accumulate codes [VAG06], or turbo codes [AG02] is used. LDPCA [VAG06] produce for each bit plane the syndrome bits, which are accumulated modulo 2^2, to produce the accumulated syndrome. The encoder stores in the buffer the accumulated syndromes and transmits only a punctured version of these. The quantity of bits of the first chunck corresponds to the minimum rate estimated at the decoder (the method for calculating this will be shown in the next section). If the bit error rate at the decoder exceeds a certain threshold, more bits of the accumulated syndromes are requested by the feedback channel. Varodayan et al. [VAG06] proved that if a syndrome subset is transmitted instead of the accumulated syndrome subset, the decoding would be severely degraded and unsuitable for iterative decoding.

In a turbo code architecture [AG02], the bit stream and an interleaved version of this are fed separately into two convolutional channel encoder. The systematic parts are discarded and the two sets of parity bits are sent to the decoder. Also in this case, only a punctured version of the parity bits is initially sent to the decoder according to the minimum rate estimation. Then, if necessary, more bits will be sent. Varodayan et al. [VAG06] show that the performance of LDPCA and turbo codec in DVC are very similar.

3.3.2 Decoder structure

Side Information Construction for Monoview Distributed Video Coding

The decoder must find a prediction for the WZ frame. This estimation will be corrected by the parity bits sent by the turbo encoder. There are two categories of methods for building the SI: interpolation and extrapolation. The SI interpolation is performed by considering past and future frames, while extrapolation considers only past frames. This second approach is suboptimal with respect to the first one, but assures real-time operation. Since we have not supposed to have constraints on real time, in this thesis we will consider only interpolation techniques. For the sake of simplicity, the GOP size (the distance

[2] If $(s_1, s_2, \ldots, s_n)$ is the syndrome, the accumulated modulo 2 syndrome is $(s_1, s_1 \oplus s_2, \ldots, \oplus_{k=1}^{n} s_k)$

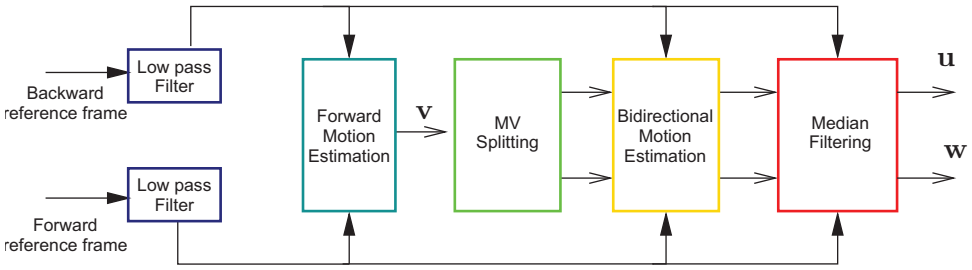


Figure 3.8: The block diagram for the DISCOVER motion interpolation algorithm

between two KFs) is supposed to be equal to 2. Then, the frames which are interpolated are the two adjacent KFs[3]. Let I_{t-1} and I_{t+1} be two consecutive KFs and let I_t be the WZF that needs to be estimated.

Before DISCOVER motion interpolation technique, other techniques have been proposed in literature. A very trivial solution for side information generation is to average the two KFs I_{t-1} and I_{t+1} [AZG02] but the performances are very unsatisfying.

When in the video the movement is really smooth, a Symmetric Motion Vector Interpolation (SMVI) can be applied [AZG02]. It consists in finding for each block centred in $\mathbf{p}$ the vector $\mathbf{v}$ that minimizes the following criterion

$$J(\mathbf{v}) = \sum_{\mathbf{p}} [I_{t-1}(\mathbf{p} - \mathbf{v}(\mathbf{p})) + I_{t+1}(\mathbf{p} + \mathbf{v}(\mathbf{p}))]^2$$

The WZF estimation will be the forward and the backward KFs compensated by the MVF $\mathbf{v}$. This method is effective when I_{t+1} and I_{t-1} are very similar.

A motion compensated frame interpolation (MCFI) algorithm has been proposed by Zhai et al. [ZYLL05]. A motion estimation from I_{t-1} to I_{t+1} and from I_{t+1} and I_{t-1} is performed. The obtained vectors are split and block by block the best estimation is chosen. DISCOVER motion temporal interpolation algorithm (MCTI) takes this algorithm as a starting point for the trajectory interpolation method. DISCOVER MCTI consists in the following steps (see also Fig. 3.8):

1. **Low pass filter.** The two frames I_{t-1} and I_{t+1} are low-pass filtered, in order to smooth out noise.

2. **Forward motion estimation.** A motion estimation from I_{t+1} to I_{t-1} is performed. The cost function in this step is given by

$$J(\mathbf{v}) = \frac{1}{B_r B_c} \sum_{\mathbf{q}} \left| B_{t+1}^{\mathbf{p}}(\mathbf{q}) - B_{t-1}^{\mathbf{p}+\mathbf{v}}(\mathbf{q}) \right| (1 + \lambda \|\mathbf{v}\|)$$

[3]This is not necessarily true if the GOP size is greater than 2. In general, the two frames closest to WZF, available at the decoder side, are interpolated.

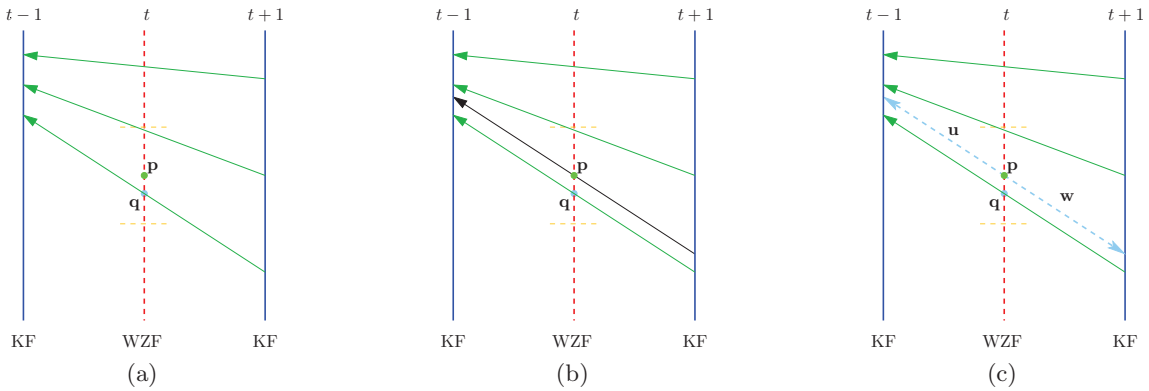


Figure 3.9: Steps for DISCOVER motion interpolation method. The goal is to estimate the block centered in $\mathbf{p}$. Let $\mathbf{q}$ be the closest intersection between the forward MVF and the WZF I_t (a). The vector in $\mathbf{q}$ is shifted so it passes by $\mathbf{p}$ (b). Afterwards, this vector is split (c).

where $B_t^{\mathbf{p}}$ is the block of dimension $B_r \times B_c$ of the frame I_t centered in $\mathbf{p}$. The vector $\mathbf{v}$ is searched within a range $\{-S, \ldots, +S\} \times \{-S, \ldots, +S\}$. $\|.\|$ is the ℓL_2-norm. This cost function is called weighted mean absolute difference (WMAD). Experimentally, the optimal value of λ has to be found.

3. **Motion vector splitting.** For each block of the frame t centered in a generic point $\mathbf{p}$, $B_t^{\mathbf{p}}$, the vector that intersects the frame t in the point closest to $\mathbf{p}$ is searched. Let $\mathbf{q}^*(\mathbf{p})$ be this point. It must satisfy

$$\mathbf{q}^*(\mathbf{p}) = \arg \min_{\mathbf{q} \in \mathcal{N}(\mathbf{p})} \left\| \mathbf{q} + \frac{1}{2}\mathbf{v}(\mathbf{q}) - \mathbf{p} \right\|^2 \tag{3.1}$$

where $\mathcal{N}(\mathbf{p})$ is the set of the block center positions neighboring of $\mathbf{p}$. Then, this vector is split into $\mathbf{u} = \mathbf{v}/2$ and $\mathbf{w} = -\mathbf{v}/2$ and afterwards centered in $\mathbf{p}$. This process is shown in Fig. 3.9.

4. **Bidirectional motion estimation.** The vectors $\mathbf{u}$ and $\mathbf{w}$ are refined by adding to them a small variation minimizing the WMAD between blocks in I_{t-1} and I_{t+1}:

$$J(\delta\mathbf{u}) = \frac{1}{B_r B_c} \sum_{\mathbf{q}} \left| B_{t+1}^{\mathbf{p}+\mathbf{u}+\delta\mathbf{u}}(\mathbf{q}) - B_{t-1}^{\mathbf{p}-\mathbf{u}-\delta\mathbf{u}}(\mathbf{q}) \right| (1 + \lambda \|\delta\mathbf{u}\|)$$

Usually, a second step of bidirectional motion estimation is performed as further refinement of the positions $\mathbf{p} + \mathbf{u} + \delta\mathbf{u}$ and $\mathbf{p} - \mathbf{u} - \delta\mathbf{u}$, but the block size and the search windows are halved.

5. **Vector median filtering.** In order to avoid spatial incoherences, a weighted median filter is applied to the two motion vector fields.

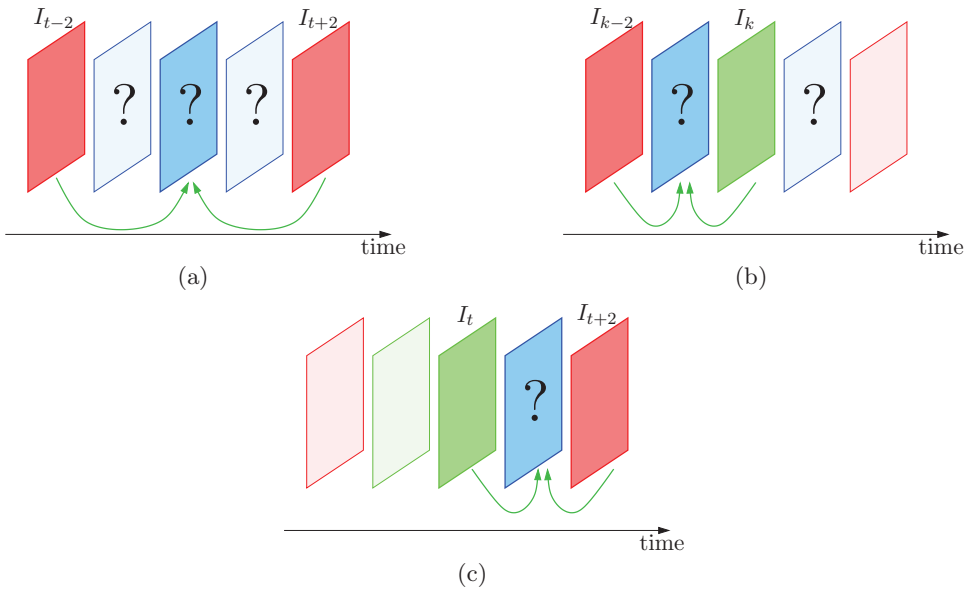


Figure 3.10: WZ frame estimation for GOP size = 4 (the KFs are in red, the WZFs are in blue, the already decoded WZFs are in green): at first we estimate I_t (a)), after by it, we estimate I_{t-1} (b) and I_{t+1} (c)

The SI is finally obtained as average of the motion compensated backward and forward references.

Let us consider now the cases where the GOP size is greater than two and it is a power of two (*i.e.* 4, 8) [ASG03]. For example, if GOP size is equal to 4, the MCTI is performed as follows

1. The WZF at instant t is estimated by using the KFs I_{t-2} and I_{t+2} and decoded, as in Fig. 3.10(a).

2. The WZF at instant $t-1$ is estimated by using the KF I_{t-2} and the decoded WZF I_t (Fig. 3.10(b)) and the lateral one at instant $t+1$ is estimated by using the KF I_{t+2} and the decoded WZF I_t (Fig. 3.10(c)).

This algorithm can be extended in all the cases when the GOP size is a power of two. When this condition is not satisfied, for example GOP size equal to 3, only the motion splitting step has to be modified. If the KFs are I_{t-1} and I_{t+2} and we want to estimate the WZF I_t, the estimated MVF $\mathbf{v}$ from $t+2$ to $t-1$ is split into $\mathbf{u} = +\mathbf{v}/3$ (for backward MVF) and $\mathbf{w} = -2\mathbf{v}/3$ (for forward MVF). Afterwards, also the motion compensation procedure is slightly different: the backward compensated frame is weighted more (2/3) with respect to the forward one (1/3).

Other algorithms for SI generation

Several algorithms have been proposed in order to improve the DISCOVER performances. In [CMPP09], [CMMPP09] dense vector field techniques, such as Cafforio Rocca algorithm and total variation, for motion estimation are extended to the case of the motion interpolation in distributed video coding. Huang et al. [HF08] propose to improve the forward motion estimation by using the chroma components of the decoded KFs. A bidirectional motion estimation with variable block size is also used. For the boundaries, 4x4 blocks provide a better SI than 8x8 blocks: if the MSE calculated between two 8x8 colocated blocks of the backward and the forward compensated frames is larger than a first threshold and the variance of the neighboring vectors is larger than a second one, these blocks are split into four 4x4 blocks. Finally, an adaptive weighted overlapped-block motion compensation is used. Each compensated block is weighted by the inverse of MSE between the forward and backward compensated blocks.

In the work of Macchiavello et al. [MBP+09], the forward and backward MVFs are estimated between the two KFs: without shifting vectors, the splitting is performed. Two motion compensated frames are obtained, where overlapped areas are averaged, but several blank areas occur. They can be filled by a block matching between these partially blank blocks and the two adjacent KFs. Instead, Ascenso et al. [AP08] propose to add a constraint during the motion estimation from $t+1$ to $k-1$: all the motion vectors have to cross the center of a block in the WZFs. In the two works of Ye et al. [YODE08, YODE09], a partially decoded WZF (for example after decoding the parity bits of the DC band) is used to detect suspicious vectors and to refine the motion vector fields. Finally, an optimal motion compensation mode selection is proposed: the previous, the next frame, the bidirectional motion-compensated average of the previous and the next frame are used for the SI, by evaluating which of them gives the smallest matching error. Abou-El-Ailah et al. [AEDF+12] propose to fuse global and local motion estimation for constructing the SI: for global motion estimation, the parameters that model the global motion (translational, affine or perspective) are estimated by SIFT features and sent to the decoder. Local motion estimation is obtained by DISCOVER MCTI. The SI obtained by global and local methods are fused during the decoding process by using also the partially decoded WZF.

Martins et al. [MBAP09] propose an iterative refinement as in [YODE08] for each band, but only for some selected blocks. Let Y be the initial SI and let R^b be the partially decoded WZF up to band b. The blocks for which the MAD between Y and R^b is larger than a given threshold are refined, by searching among the neighboring blocks the one that minimizes the MAD. This will be used as new SI from the correction of the next band. This refinement is performed for each band.

Hash-controlled motion estimation techniques have also been proposed: some additional information, called hash signature, is sent to the decoder [TT07, PLKL12], as for example some blocks of the original WZF. In this context, Verbist et al. [VxJ+11] pro-

pose a probabilistic model for SI construction. An overlapped block motion estimation [DMC+09b] is performed from I_{t-1} to I_{t+1}, and a collection of candidate predictor values is available for each pixel. Then, by supposing that the noise between the side information and the real Wyner-Ziv Frame can be modeled as a Laplacian, the best candidate can be chosen by a maximum likelihood estimation. Some a priori information, e.g. the density probability function of the source, is needed in order to perform this estimation. For this reason, hash based architectures are distributed in a wide sense. Kubasov et al. [KG06] propose to replace block matching technique for motion estimation with mesh-based motion-compensated interpolation, in order to take into account more complex motion than simple translation. Mys et al. [MSS+09] propose to add a SKIP mode for DVC, such as in H.264/AVC. Usually there are several blocks between I_{t-1} and I_{t+1}, that do not change, and the SI for these blocks does not need parity bits to be corrected.

Another framework for DVC developed in recent years is VISNET II codec [ABD+10]: the motion interpolation step for SI generation is the same as for DISCOVER. The main modifications at the decoder side are the iterative reconstruction of the WZF for each DCT band and an adaptive deblocking filter applied to the decoded WZFs. At the encoder a CRC checking is added in order to make this codec more robust to errors in the decoded frames.

Estimation of the correlation noise

As already said, the SI is considered as a noisy version of the WZF. The difference between them is called correlation noise. The estimation of the correlation noise characteristics between the side information and the real WZF is of crucial importance for DVC. Indeed, the information about the correlation noise statistics is necessary for the channel decoder, the minimum rate allocation and for the reconstruction of the DCT coefficients. As a consequence, the estimation of the statistics will affect the rate distortion performance of the whole codec. The most common model (used in the DISCOVER codec) [BP08] for the correlation noise probability distribution is the Laplacian one:

$$f_{X|Y=y}(x) = \frac{\alpha}{2} e^{-\alpha|x-y|}$$

where Y is the DCT SI and X the DCT WZF. A Laplacian PDF is completely described by its parameter $\alpha > 0$. Its value is estimated at

- DCT band sequence level: one α per DCT band for the whole sequence
- DCT band frame level: one α per DCT band and frame
- coefficient level: one α per DCT coefficient

Unfortunately, at the decoder it is not possible to evaluate the statistical distribution of the error because the real WZF is unavailable. Experimentally, it has been proved that

a rough estimation of this error is the difference between the backward and the forward compensated frames by the two MVFs estimated by MCTI and divided by two. It is usually called residual error.

Then, this parameter can be estimated

- off-line: in this case, α is estimated taking into account the SI and the real WZF
- on-line: the real error is replaced by the residual error

Maugey et al. [MGPPG10] propose to use a generalized Gaussian PDF model for the correlation noise, instead of the Laplacian pdf. This methods outperforms the results of DISCOVER in terms of rate distortion end to end both on-line and off-line.

Deligiannis et al. [DMC+09a] propose also a noise model that is dependent on the side information. In this case, the pdf that models the correlation noise will be

$$f_{X|Y=y}(x) = \frac{\alpha(y)}{2} e^{-\alpha(y)|x-y|}$$

where α is a function of y. This means that for each value of y, we have a different pdf that models the correlation noise. This model has been implemented also in an hash-based architecture [DBJ+12].

Minimum rate estimation

In order to reduce the additional requests of parity bits by the decoder for correcting the SI, the encoder determines the minimum number of parity bits [KLG07] to be sent per bit plane and per band.

This rate for the bit plane j of the DCT band b can be estimated from the entropy of the crossover probability, *i.e.* the probability that the decoded bit plane is different from the original one. For the b-th bit-plane and for the band j, it is defined as

$$p_{co,b,j} \triangleq \Pr(X_j \neq \widehat{X}_j)$$

where X_j is the j-th bit plane for the particular DCT coefficient X in the b-th band and $\widehat{X}_j$ its estimation at the decoder. Since this procedure is independent band by band, for sake of brevity we omit the dependence of X_j from b. The more $p_{co,b,j}$ is close to 1/2, the higher is the minimum rate. Given the SI DCT coefficient $Y = y$ and the previous bits of X $(X_{j-1}, \ldots, X_1)$, $\widehat{X}_j$ is reconstructed by the channel decoder as

$$\widehat{X}_j \triangleq \arg\max_{i=0,1} \Pr(X_j = i | Y, X_{j-1}, X_{j-2}, \ldots, X_1)$$

In this equation, $\Pr(X_j = i|Y, X_{j-1}, X_{j-2}, \ldots, X_1)$ is the a posteriori probability of the event $X_j = i$. For example, let us suppose that we quantize X in 3 bit planes. Let us compute $\Pr(X_3 = 1|Y = y, X_1 = 0, X_2 = 1)$ as example. Let $[t_1, t_2]$ be the interval that

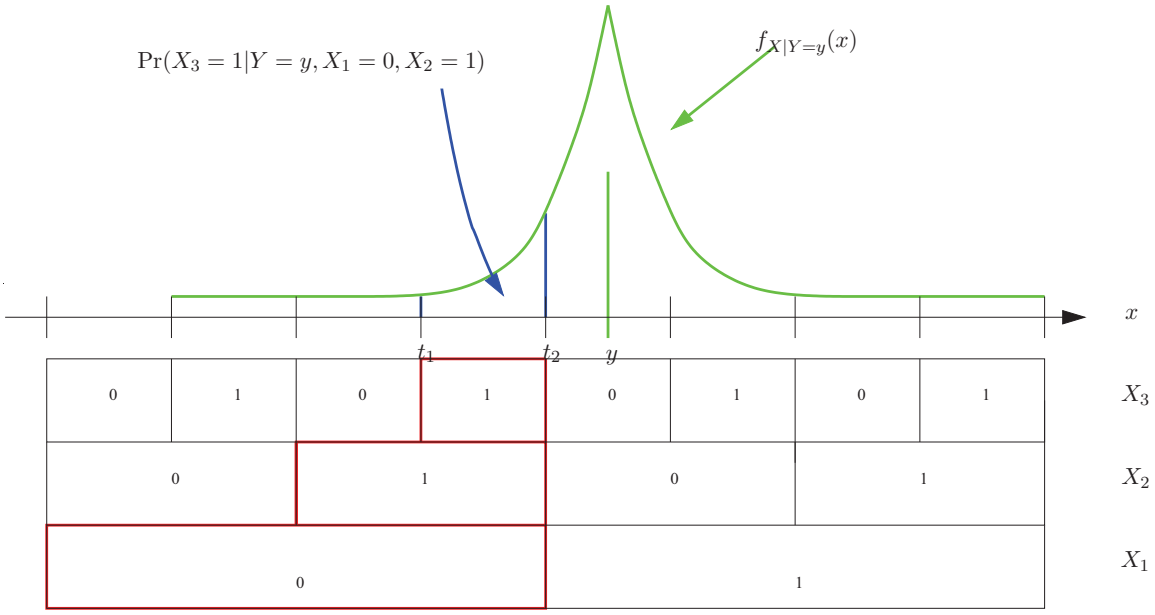


Figure 3.11: The computation of the probability $\Pr(X_3 = 1|Y = y, X_1 = 0, X_2 = 1)$

corresponds to the string 011. Then, $\Pr(X_3 = 1|Y = y, X_1 = 0, X_2 = 1)$ is the area under the function $f_{X|Y}(x|y)$ between t_1 and t_2. Since the pdf of X given y is supposed to be Laplacian, we can conclude that

$$\Pr(X_3 = 1|Y = y, X_1 = 0, X_2 = 1) = \int_{t_1}^{t_2} \frac{\alpha}{2} e^{-\alpha|y-x|} dx$$

In Fig. 3.11 the conditional PDF for a given band is shown in green. The DCT coefficients are supposed to be quantized on 3 bits. For each interval the value of the three bits X_1 (MSB), X_2, X_3 (LSB) are shown in each row. The area indicated by the blue arrow is the estimated $\Pr(X_3 = 1|Y = y, X_1 = 0, X_2 = 1)$. Thus the cross over probability for the bit plane j and for the DCT band b is obtained as

$$p_{co,b,j} = \frac{1}{L} \sum_{x \in \mathcal{W}_b} \left[\int_{-\infty}^{+\infty} \Pr(X_j \neq \widehat{X}_j | X = x, Y = y) \frac{\alpha}{2} e^{-\alpha|y-x|} dy \right]$$

where $\mathcal{W}_b$ for $b = 1, \ldots, 16$ is the set of the coefficients of the b-th band and L is the cardinality of $\mathcal{W}_b$, *i.e.* $\frac{MN}{16}$ where $M \times N$ is the resolution of the WZF. Then, the minimum rate estimation value is sent to the encoder via the feedback channel.

Channel decoder error rate estimation

A very trivial method for detecting errors at the decoder is to compute the syndrome check error. It consists in checking the Hamming distance between the received syndrome and

the syndrome obtained by the decoded bit plane. While this distance is different than zero, the decoder requests more parity bits.

An other method proposed in [KLG07] and implemented in DISCOVER codec consists in defining the likelihood ratio Λ_j for each bit plane, as:

$$\Lambda_j \triangleq \log_{10} \frac{\Pr(X_j = 1|Y)}{\Pr(X_j = 0|Y)}$$

The likelihood ratio is used as a confidence measure of correct decoding. In other words the estimation of the j-th bit plane is considered uncertain if $|\Lambda_j| < \log_{10} T$, where T is a suitable threshold.

The BER is estimated as

$$\widehat{P} = \frac{\text{count}_j\{|\Lambda_j| < \log_{10} T\}}{N}$$

where $\text{count}_j\{|\Lambda_j| < \log_{10} T\}$ is the number of the bits such that $|\Lambda_j| < \log_{10} T$ and N is the number of bit planes. Typically, if $\widehat{P} > 10^{-3}$, the decoder requests more parity bits from the buffer.

Reconstruction and inverse transform

The reconstruction of the WZF consists in performing the inverse quantization by the knowledge of the SI and of the WZF decoded indices.

Let M be the number of quantized levels used for quantizing a DCT coefficient and let be Δ the quantization step. The reconstruction levels are $z_0, \ldots, z_{M-1}$. If the quantized index k is obtained by the channel decoder and y is the value of the DCT SI coefficient, a trivial method for recovering the DCT coefficient is

$$\widehat{x} = \begin{cases} z_k & \text{if } y < z_k \\ y & \text{if } y \in [z_k, z_{k+1}) \\ z_{k-1} & \text{if } y \geq z_{k+1} \end{cases}$$

This approach is suboptimal. In fact, the best approach is to minimize the MMSE by computing the estimation

$$\hat{X} = \arg\min_x \mathbb{E}[(X - x)^2 | x \in [z_k, z_{k+1}), Y = y]$$

It can be shown that, under some conditions, it is equivalent to find the estimator

$$\widehat{x} = \mathbb{E}[x | x \in [z_k, z_{k+1}), y]$$

that is

$$\widehat{x} = \frac{\int_{z_k}^{z_{k+1}} x f_{X|Y}(x|y)dx}{\int_{z_k}^{z_{k+1}} f_{X|Y}(x|y)dx}$$

Since the conditional PDF of the original value x given the side information y is supposed to be Laplacian of parameter α, we can compute $\widehat{x}$ [KNG07] as

$$\widehat{x} = \begin{cases} z_k + \frac{1}{\alpha} + \frac{\Delta}{1-e^{\alpha\Delta}} & \text{if } y < z_k \\ y + \frac{(\gamma+\frac{1}{\alpha})e^{-\alpha\gamma} - (\delta+\frac{1}{\alpha})e^{-\alpha\delta}}{2-(e^{\alpha\Gamma}+e^{-\alpha\Gamma})} & \text{if } y \in [z_k, z_{k+1}) \\ z_{k+1} - \frac{1}{\alpha} - \frac{\Delta}{1-e^{\alpha\Delta}} & \text{if } y \geq z_{k+1} \end{cases}$$

where $\gamma = y - z_k$ and $\delta = z_{k+1} - y$.

3.4 Multiview distributed video coding

Until now, we have considered only monoview videos. Now, we show how the DISCOVER codec can be extended to multiview videos.

When we deal with multiview systems, in order to refer to a frame we need a second index for the view-point. We note as $I_{t,k}$ the image taken at the instant t by the k-th camera. The corresponding SI is indicated with a hat.

In multiview DVC systems, three types of camera are commonly considered: pure Key cameras for which all the frames are KFs; pure Wyner-Ziv cameras where all the frames are WZFs; and hybrid cameras for which each second frame is a WZFs and the others are KFs. These cameras can be arranged in a variety of configurations. However, in the literature, mainly three schemes have been considered [MPP08]:

- Asymmetric scheme (see Fig. 3.12(a)): pure Key cameras and pure Wyner-Ziv cameras are alternated
- Hybrid 1/2 scheme (see Fig. 3.12(b)): every second camera is a pure Key camera and the others are hybrid cameras
- Symmetric 1/2 scheme (see Fig. 3.12(c)): all the cameras are hybrid and the KFs and the WZFs are placed on a quincunx grid in the time-view axes

In all these schemes, for a generic WZF $I_{t,k}$, at least two KFs are available, $I_{t,k-1}$ and $I_{t,k+1}$, *i.e.* two images taken at the same temporal instant t by two neighboring cameras. Moreover, except for the first scheme, two other images are available, namely $I_{t-1,k}$ and $I_{t+1,k}$, the previous and next frames from the same camera. All these images should be used in order to generate a side information as reliable as possible. For multiview video systems, the architecture of the codec is the same as for monoview video (see Fig. 3.7). However, different techniques for side information construction can be developed in oder to take into account the different characteristics of motion and disparity.

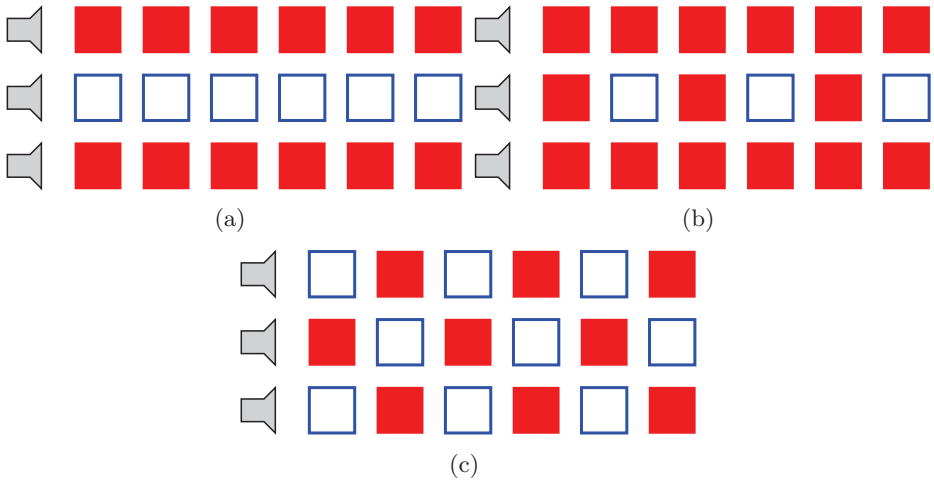


Figure 3.12: The three main configurations for multiview video: Asymmetric scheme (a), Hybrid 1/2 scheme (b), Symmetric 1/2 scheme (c). The filled frames are the KFs and the blank ones are the WZFs.

3.4.1 Side Information Construction for Multiview Distributed Video Coding

According to the arrangement of KFs and WZFs, the WZFs can be estimated in different ways:

- **the temporal estimation**: the temporal adjacent frames of the same view are used for the interpolation,
- **the inter-view estimation**: the frames of the adjacent views at the same time instant are used for the interpolation,
- **A fusion of them**: the temporal and the inter-view estimations are fused according to different criteria.

In the following, the fused image is indicated by $\widetilde{I}(\mathbf{p})$, the temporal estimation by $\widehat{I}_T(\mathbf{p})$ and the inter-view estimation by $\widehat{I}_V(\mathbf{p})$. Methods for temporal interpolation have already been described in Section 3.3.2. These techniques can easily be extended to the view domain by replacing the adjacent frames in temporal domain with the frames belonging to the neighboring views. However, these techniques do not always give very good performance. Then, several algorithms have been proposed in order to improve the inter-view estimation by exploiting properties of multiview geometry. In fact, inter-view estimation can be seen as an image-based rendering (IBR) process. Under this respect, this problem has been long studied [SK00], well before the arise of DVC systems. A simple approach to the inter-view estimation problem consists in using the motion interpolation algorithms among neighboring view images: the concept of motion is replaced by that of disparity. Of course this can be done only if some constraints are satisfied: the views have to be

rectified with a viewing axis perpendicular to the baseline [GPT+07]. Even if simple, this kind of approach has the drawback of not taking into account the specificity of inter-view estimation and cannot assure the best performances. Therefore, in the literature there are several approaches based on increasingly complex models of the multi-camera setup. For example, Guo et al. [GLW+06] propose a global affine motion model with six parameters for inter-view interpolation. However, global motion models cannot accurately describe the sudden variation in the disparity field associated to object borders. Occlusions are not taken into account neither.

Artigas et al. [AAT06] propose a method based on depth maps. Given the KF image, the associated depth map and the camera parameters, it is possible to create a virtual view point, namely that of the WZ camera. The synthesized image suffers, nevertheless, from some drawbacks: occluded areas cannot be rendered, errors in depth maps generate annoying artifacts, and view-dependent image features such as reflections cannot be correctly interpolated. These problems are mitigated when image fusion is taken into account (see next Section). However in our work we consider a different problem, where depth information is not directly available.

A similar approach is used by Ouaret et al. [ODE06] for an a hybrid 1/2-scheme with three cameras. In this case the homography matrices needed to map the KF into the WZ camera are estimated using a MMSE criterion, but local object motion can create outliers and this invalidates the estimation of the homography matrices. Moreover, this technique can suffer from distortion caused by camera lens.

Areia et al. [AABP07] propose a very simple method for the hybrid 1/2-schemes with two cameras (stereoscopic video): the disparity field is computed over the two last decoded frames, and then it is directly applied to the current KF to generate the side information. Authors recognize that this approach may not achieve the best performances, but it has the advantage of being very simple and of reusing the same algorithms of the temporal interpolation step.

Finally, we cite the work by Miled et al. [MMCPP09b] since it departs from common block based disparity compensation algorithms, aiming at a dense disparity estimation. As observed in [CMPP09, CMMPP09], dense vector fields make sense for DVC since they are not sent, but computed at the decoder. In [MMCPP09b], the disparity field is computed by a global method with a constraint on the total variation.

Fusion techniques

When both temporal and inter-view interpolations are available, such as in the case of a symmetric 1/2 scheme, they have to be combined in order to create a single side information. In the context of DVC, this operation is referred to as fusion.

A common reference is the oracle fusion [AABP07]: for each pixel $\mathbf{p}$, the value of the fused image $\widetilde{I}(\mathbf{p})$ is selected as the temporal estimation $\widehat{I}_T(\mathbf{p})$ value or the inter-view

estimation $\widehat{I}_V(\mathbf{p})$ value, according to the error with respect to the actual WZF. Of course this method cannot be used in practice, and serves only as a comparison.

Practical methods for image fusion are often based on the differences between the interpolated and reference images. Let us refer to the absolute difference between the temporal interpolation and the forward [resp. backward] reference as $e_{\mathrm{T}}^{\mathrm{F}}$ [resp. $e_{\mathrm{T}}^{\mathrm{B}}$]. Likewise, the difference between the inter-view interpolation and the left and right views are referred to as $e_{\mathrm{V}}^{\mathrm{L}}$ and $e_{\mathrm{V}}^{\mathrm{R}}$. In [ODE06], for each pixel $\mathbf{p}$, the inter-view interpolation is chosen if $e_{\mathrm{V}}^{\mathrm{L}}(\mathbf{p}) < e_{\mathrm{T}}^{\mathrm{B}}(\mathbf{p})$ and $e_{\mathrm{V}}^{\mathrm{R}}(\mathbf{p}) < e_{\mathrm{T}}^{\mathrm{F}}(\mathbf{p})$. Otherwise, the temporal interpolation is used. In [FAB07], two fusion techniques are proposed: temporal motion (TM) interpolation projection fusion (IPF) and spatial view (SV) IPF. In TM-IPF a block matching between the temporal estimation $\widehat{I}_T$ and the two adjacent KFs in the time domain is performed. For the pixels where the two compensation errors are larger than a threshold, inter-view interpolation is used because temporal interpolation is supposed to have failed. Otherwise, the temporal interpolation is kept. SV-IPF is the dual of TM-IPF: $\widehat{I}_V$ is kept if its error relative to KFs is small, otherwise $\widehat{I}_T$ is used. Guo et al. in [GLW+06] compute the absolute difference between the two motion compensated frames in time domain. For pixels where this difference is larger than a threshold and if the norm of the two motion vectors is larger than a threshold, inter-view interpolation is chosen; otherwise temporal interpolation is used.

In [MMCPP09a] the fusion is based on two different error images, referred to as E_T and E_V. E_T is the absolute difference between the motion-compensated backward reference and the motion-compensated forward reference. Likewise, disparity compensation on left and right images is used to create E_V. Two efficient fusion techniques are proposed: in the binary fusion (Bin), the pixel value in $\mathbf{p}$ is selected from the inter-view interpolation if $E_V(\mathbf{p}) < E_T(\mathbf{p})$, otherwise is selected from the temporal interpolation; in the linear fusion (Lin) the coefficient $\alpha = \frac{E_T(\mathbf{p})}{E_T(\mathbf{p})+E_V(\mathbf{p})}$ is computed and $\widetilde{I}_{LIN}(\mathbf{p})$ is defined as a weighted average of $\widehat{I}_V(\mathbf{p})$ and $\widehat{I}_T(\mathbf{p})$, using respectively α and $1-\alpha$ as weights: *i.e.*

$$\widetilde{I}_{\mathrm{LIN}}(\mathbf{p}) = \alpha \widehat{I}_V(\mathbf{p}) + (1-\alpha)\widehat{I}_T(\mathbf{p}) \tag{3.2}$$

Recently, Support Vector Machine was proposed by Dufaux [Duf11] for fusion, which is considered as a classification problem with two classes. The features are extracted from the four KFs surrounding the WZF that has to be estimated.

Unfortunately, when the estimation in one domain is much better than the other one, the quality of the fused image is not better then the best SI between the temporal and the inter-view one. Then, it would be better to detect when this asymmetry occurs and choose directly for the best SI without fusing them. In the next chapter, we propose an adaptive algorithm that chooses the best method for SI construction among temporal, inter-view interpolation and fusion.

Chapter 4

Proposed methods for Side Information Construction

Contents

In this chapter, we describe algorithms for improving side information generation in distributed video coding. At first, we present the algorithm proposed for temporal interpolation, called High order motion interpolation (HOMI). In a second part, an algorithm for inter-view interpolation with priority to large disparity (IPLD) for multiview video is introduced. Methods for fusing the two interpolations by detection of occluded areas are then described. Finally, an adaptive validation algorithm is proposed. Indeed, fusion algorithms usually work well if the qualities of temporally and inter-view interpolated frames are comparable. If this is not the case, it is better to choose directly the temporal or inter-view interpolation. The adaptive validation algorithm, based on the parity bit rate, chooses the best SI. Finally, an application of HOMI algorithm has been proposed for multiple description coding.

4.1 High order motion interpolation (HOMI)

The algorithm for side information generation in DISCOVER (described in Section 3.3) is quite effective but can be improved in several ways. It performs a linear trajectory interpolation for deducing the object positions in the missing WZFs. This is equivalent to assume a uniform motion model (no acceleration of objects) in the whole period between the two KFs. However, whenever the motion in the video deviates from this model, we risk to end up with an unsatisfactory motion estimation. So, we propose to perform a higher order interpolation of object positions, in order to be able to take into account more complex motion models (*e.g.* constant acceleration, non-linear trajectories, and so on).

We describe the proposed method along with an 1-D illustration of each step shown in Fig. 4.1, by supposing that the GOP size (i.e. the distance between two consecutive KFs) is equal to 2. We want to estimate the WZF at instant t. Our algorithm consists in four steps: the initialization of the MVFs by DISCOVER algorithm run on KFs I_{t-1} and I_{t+1}; a further block matching to find the positions of the current block in KFs I_{t-3} and I_{t+3}; the interpolation of the block positions; and finally the adjustment of the vectors in the center of the block.

Let us consider a block of the current WZF (which of course is not available at the decoder), centred on the pixel $\mathbf{p}$. The DISCOVER MCTI algorithm provides us the backward MVF $\mathbf{u}$ and the forward one $\mathbf{w}$ (shown in black dashed arrows in Fig. 4.1) . Then, the block of pixel $B_{t-1}^{\mathbf{p}+\mathbf{u}(\mathbf{p})}$ of the frame I_{t-1} centered in $\mathbf{p}+\mathbf{u}(\mathbf{p})$ is supposed to move into position $\mathbf{p}+\mathbf{w}(\mathbf{p})$ of the frame I_{t+1}. Now we want to refine the movement of this block among the frames by taking into account the images I_{t+3} and I_{t-3}. Therefore, we start by looking for the position that the block $B_{t-1}^{\mathbf{p}+\mathbf{u}(\mathbf{p})}$ has in I_{t-3}, by using a regularized block-matching search. More precisely, we look for the vector $\widetilde{\mathbf{u}}$ such that the following

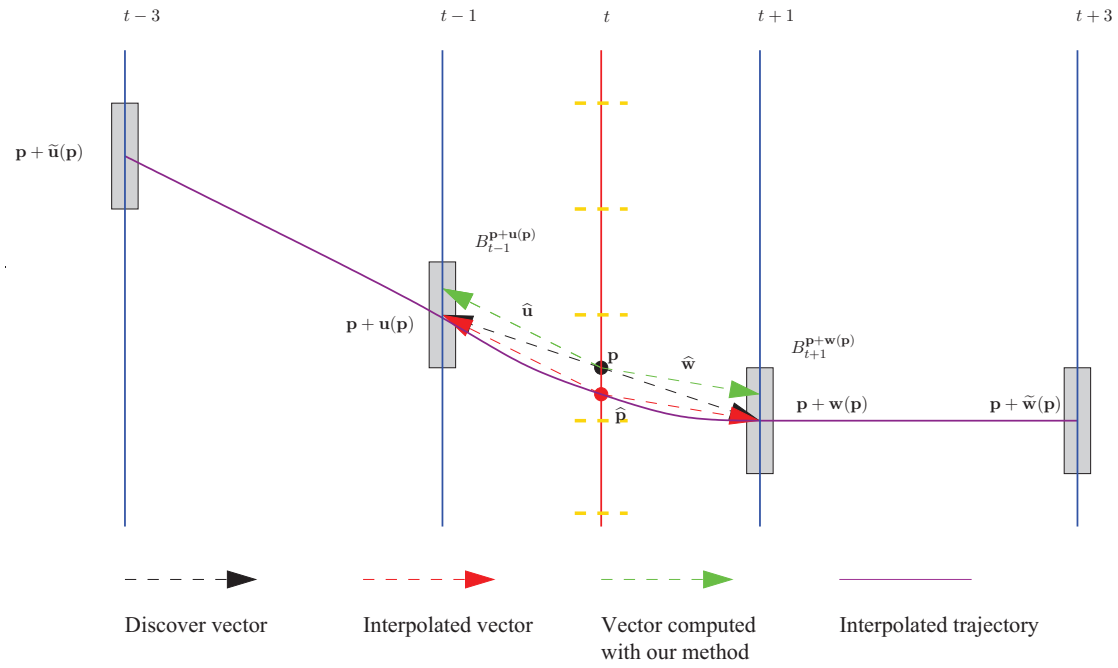


Figure 4.1: HOMI method for motion estimation.

function is minimized:

$$J(\widetilde{\mathbf{u}}) = \sum_{\mathbf{q}} \left| B_{t-1}^{\mathbf{p}+\mathbf{u}(\mathbf{p})}(\mathbf{q}) - B_{t-3}^{\mathbf{p}+\widetilde{\mathbf{u}}}(\mathbf{q}) \right|^n + \lambda \|\widetilde{\mathbf{u}} - 3\mathbf{u}\| \tag{4.1}$$

The regularization term penalizes too large deviations from the linear model: with $\lambda \to \infty$ the proposed algorithm becomes equivalent to DISCOVER. The sum of absolute differences (SAD) or the sum of squared differences (SSD) can be used in this criterion simply by setting $n = 1$ or $n = 2$ respectively. Likewise, we can find the vector $\widetilde{\mathbf{w}}$ matching the block $B_{t+1}^{\mathbf{p}+\mathbf{w}(\mathbf{p})}$ with another block in I_{t+3}.

The following step consists in interpolating the positions of the current block in the four images. In other words we interpolate a vector function with the values $\mathbf{p}+\widetilde{\mathbf{u}}$, $\mathbf{p}+\mathbf{u}$, $\mathbf{p}+\mathbf{w}$ and $\mathbf{p}+\widetilde{\mathbf{w}}$ respectively at instants $t-3$, $t-1$, $t+1$, $t+3$, in order to find its value at instant t, be it $\widehat{\mathbf{p}}$. We use a piecewise cubic Hermite interpolation to find the position [FC80]. As a consequence, we can suppose that the object, that moves from the position $\mathbf{p}+\mathbf{u}$ in I_{t-1} to $\mathbf{p}+\mathbf{w}$ in the frame I_{t+1}, is in the position $\widehat{\mathbf{p}}$ within the frame I_t. Then, the motion vectors associated to $\widehat{\mathbf{p}}$ are $\mathbf{u}(\mathbf{p})+\mathbf{p}-\widehat{\mathbf{p}}$ and $\mathbf{w}(\mathbf{p})+\mathbf{p}-\widehat{\mathbf{p}}$. These vectors are shown in dark red in Fig. 4.1.

The last step consists simply in choosing the interpolated trajectory passing closest to the block center and in assigning the associated vector to the position $\mathbf{p}$, just as in DISCOVER. The difference is that the trajectory is interpolated using four points instead of two. The final vectors are shown in green in Fig. 4.1.

Afterwards, the frame I_{t-1} is motion-compensated using the backward MVF $\widehat{\mathbf{u}}$ and

I_{t+1} using the forward MVF $\widehat{\mathbf{w}}$. The average of these two compensations will be the WZF estimation.

This algorithm can be extended to the case of GOP size greater than two. The frames used for the interpolation will be the two backward and the two forward closest available frames (not necessarily KFs, also decoded WZFs) in a hierachical encoding structure as in DISCOVER (as described in the section 3.3.2). As an example, let the GOP size be equal to 4, and let I_{t-2} and I_{t+2} be two consecutive KFs. Then, we have to estimate the WZFs at instants $t-1$, t and $t+1$. If we want to estimate the Wyner-Ziv Frame at instant t, we use the KFs I_{t-6}, I_{t-2}, I_{t+2} and I_{t+6}. Then, after decoding the frames at instant t, for estimating the frame at instant $t-1$, we use the decoded WZF I_{t-4}, the KF I_{t-2}, the decoded WZF I_t and the KFs I_{t+2}. Likewise for the frame at instant $t+1$, we use the decoded KF I_{t-2}, the WZF I_t, the KF I_{t+2} and the WZF I_{t+4}. This method can be generalized for all GOP sizes that are power of two. We observe that, theoretically, the proposed method can be used also for inter-view interpolation, where the KFs can be images taken at the same time instant by different cameras. Block matching techniques can be used for the disparity estimation only if the two cameras are aligned. Unfortunately, in this hypothesis the object motion among the frames is linear. Then, high order disparity interpolation is not expected to improve the SI quality w.r.t. DISCOVER.

4.1.1 Fast HOMI

Since in HOMI, we have to perform three motion estimations, the complexity of HOMI is approximately tripled w.r.t. DISCOVER motion interpolation algorithm, The computational complexity of HOMI can be however reduced. By supposing that the GOP size is equal to 2, at the instant t, the forward MVF $\mathbf{v}$ from I_{t-2} to I_{t-3}, and the backward MVF $\mathbf{z}$ from I_{t-2} to I_{t-1} have already been estimated. In this way, it is only necessary to estimate the motion vector field from I_{t+1} to I_{t+3} while it is not necessary to perform the motion estimation from I_{t-1} to I_{t-3}. This method is less robust than HOMI. Indeed, instead of the motion estimation from I_{t-1} to I_{t-3}, we just exploit the estimation of two MVFs $\mathbf{v}$ and $\mathbf{z}$. All the steps of this algorithm are shown in Fig. 4.2. Just like in HOMI, the MVF $\mathbf{u}$ from I_t to I_{t-1} and $\mathbf{w}$ from I_t to I_{t+1} are initialized by DISCOVER MCTI algorithm. A block matching motion estimation from I_{t+1} to I_{t+3} is performed by minimizing Eq. (4.1) and the position $\mathbf{p}+\widetilde{\mathbf{w}}$ is found. Then, we have to perform the motion estimation from I_{t-1} to I_{t-3} by exploiting the MVF $\mathbf{v}$ and $\mathbf{z}$. We have to perform a "connexion" between the motion vector field $\mathbf{u}$ and $\mathbf{z}$, because not all the pixels of the frame I_{k-1} are pointed by a vector of the MVF z. Then, we search for the position $\mathbf{q}$, such that

$$\mathbf{q} = \arg\min_{\mathbf{r}\in C} \|\mathbf{p} + \mathbf{u}(\mathbf{p}) - (\mathbf{r} + \mathbf{z}(\mathbf{r}))\| \tag{4.2}$$

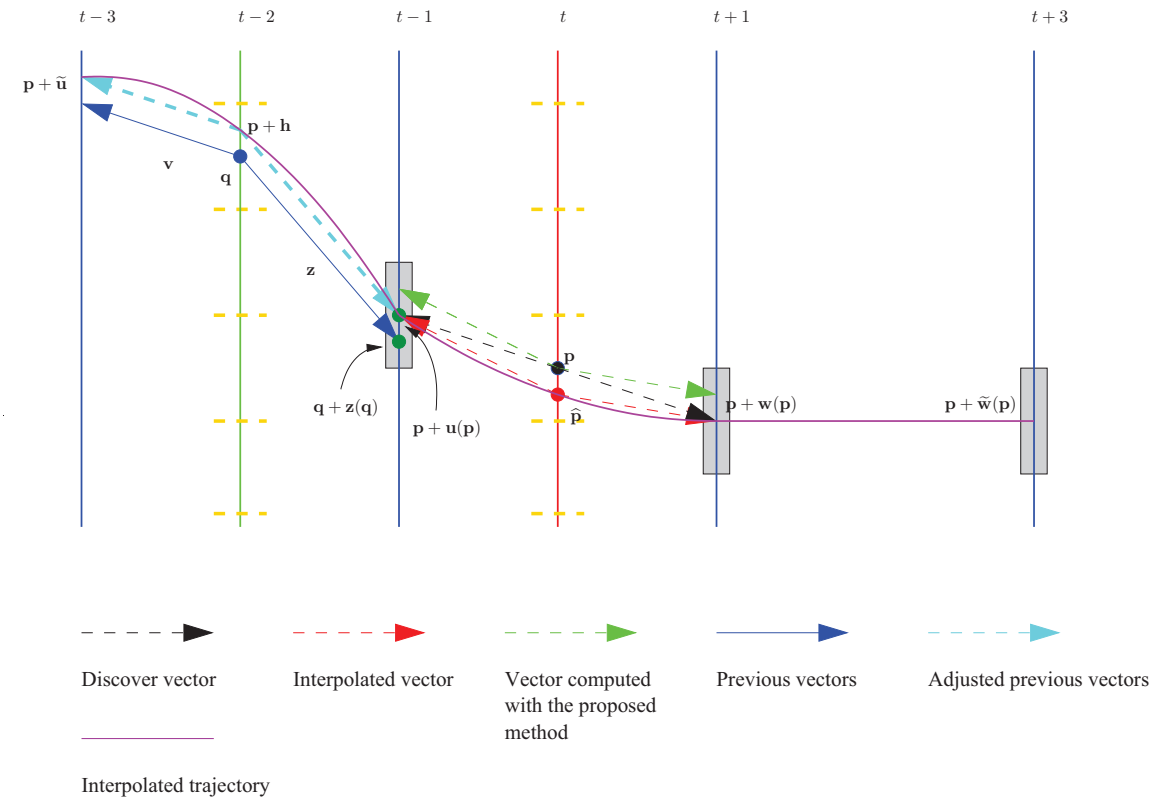


Figure 4.2: Proposed interpolation method (Fast HOMI) for WZF estimation by exploiting the previous estimated MVF.

where C is the set of the central pixels of each block. Then, the vector $\mathbf{z}(\mathbf{q})$ points within the frame I_{t-1} to the position closest to $\mathbf{p}+\mathbf{u}(\mathbf{p})$. As a consequence, we can assume that the object in the position $\mathbf{p}+\mathbf{u}(\mathbf{p})$ in frame I_{t-1} moves into the position $\mathbf{p}+\mathbf{h}$ in the frame I_{t-2}, with $\mathbf{h}=\mathbf{u}(\mathbf{p})-\mathbf{z}(\mathbf{q})$, by supposing that $\mathbf{q}+\mathbf{z}(\mathbf{q})$ and $\mathbf{p}+\mathbf{u}(\mathbf{p})$ belong to the same object (this hypothesis would be not satisfied if the GOP size is very large). In the frame I_{t-3}, the object will be in $\mathbf{p}+\tilde{\mathbf{u}}$, with $\tilde{\mathbf{u}}(\mathbf{p})=\mathbf{h}+\mathbf{v}(\mathbf{q})$ as shown in Fig. 4.2. Finally, the vector function with the five values $\mathbf{p}+\tilde{\mathbf{u}}$, $\mathbf{p}+\mathbf{h}$, $\mathbf{p}+\mathbf{u}(\mathbf{p})$, $\mathbf{p}+\mathbf{w}(\mathbf{p})$ and $\mathbf{p}+\tilde{\mathbf{w}}(\mathbf{p})$ respectively at the instants $t-3$, $t-2$, $t-1$, $t+1$ and $t+3$ is interpolated in order to find its value at the instant t, which will be denoted by $\widehat{\mathbf{p}}$. The rest of the procedure is the same as in HOMI. This variant is called FastHOMI. We observe that its complexity is about $\frac{2}{3}$ of HOMI, because we need only the ME from I_{t-1} to I_{t+1}.

4.1.2 Increasing the density of the MVF

When we perform the block matching in HOMI/FastHOMI algorithm, we consider a block $B_k^{\mathbf{p}}$, centered in $\mathbf{p}$ and whose size is $N\times N$. Then, we let $\mathbf{p}$ vary in $\{(nM,mM)\}_{(n,m)\in C}$, where $C\subset\mathbb{Z}^2$ is such that we compute a vector for each $M\times M$ block in the image. Although the condition $M=N$ is usually chosen, in order to increase the SI quality we

can use denser MVF by selecting a value for M smaller than N. We observe that, while we can increase the density by reducing M without particular constraint (excepted for the computational complexity), we cannot allow arbitrary variations of N (the block size) since if the blocks are too small, the matching may suffer from noise. On the contrary, in the case of block overlap, we can densify the MVF without sacrificing robustness. We can therefore modify both the HOMI and the Fast HOMI technique, ending up with four SI generation techniques: HOMI with $N = M = 8$, referred to as HOMI8 from now on; HOMI with $N = 8$ and $M = 4$, which we shall indicate as HOMI4; and the fast version of these techniques, referred to as FastHOMI8 and FastHOMI4. Since with HOMI4 we search four motion vectors for each 8×8 block, its computational complexity is four times the one of HOMI8 (the search windows size is the same). Likewise, the complexity of FastHOMI4 is four times the one of FastHOMI4.

4.1.3 Refinement methods for large GOPs

In the DISCOVER codec a hierarchical structure is proposed for estimating and decoding the WZF for GOP size greater than two (as shown in Section 3.3.2). If the GOP size is equal to 4, we can consider I_{t-2} and I_{t+2} as two consecutive KFs. The frame at instant t is called central WZF and the others ($t-1$ and $t+1$) are called lateral WZFs. We observed from experiments that the average PSNR of the lateral WZFs is higher than the central WZFs. Indeed, for the central WZFs we interpolate frames that are farther apart than the ones used for the lateral WZFs. Therefore, for obtaining the same distortion, more bits are needed to correct the central WZF. Then, our target is to improve the quality of the central WZF in the context of hierarchical coding.

We propose to refine the estimation of the central WZF by interpolating the lateral WZFs that we already decoded and are now available. The proposed algorithm uses HOMI. A first estimation of the central WZF, $\widehat{I}_t$, is obtained by interpolating the KFs I_{t-6}, I_{t-2}, I_{t+2} and I_{t+6}. Let I_t be the decoded WZF. The estimation $\widehat{I}_{t-1}$ of the lateral WZF is obtained by interpolating with HOMI the decoded central WZF of the previous GOP I_{t-4}, the KF I_{t-2}, the decoded WZF I_t and the KF I_{t+2}. Likewise, the estimation of the lateral WZF $\widehat{I}_{t+1}$ is obtained by interpolating by HOMI the KF I_{t-2}, the decoded I_t, the KF I_{t+2}, and the decoded central WZF of the next GOP I_{t+4}. Now, I_{t-1} and I_{t+1} are available at the decoder. Then, we can re-estimate I_t, by using HOMI on the KFs I_{t-2}, the decoded WZFs I_{t-1}, I_{t+1} and the KF I_{t+2}. This new estimation is now corrected by using the parity bits sent during the first step. Then, it is not required to resend the parity bits. The Fig. 4.3 and Tab. 4.1 summarize all these steps.

The channel decoding of the WZF I_t at the final step is sub-optimal, since we use the parity bits that were requested for the first version of the side information. In particular, the rate allocation for the DCT bands was performed using an estimation of the correlation error standard deviation (as shown in Section 3.3.2) that depends on the first SI. When

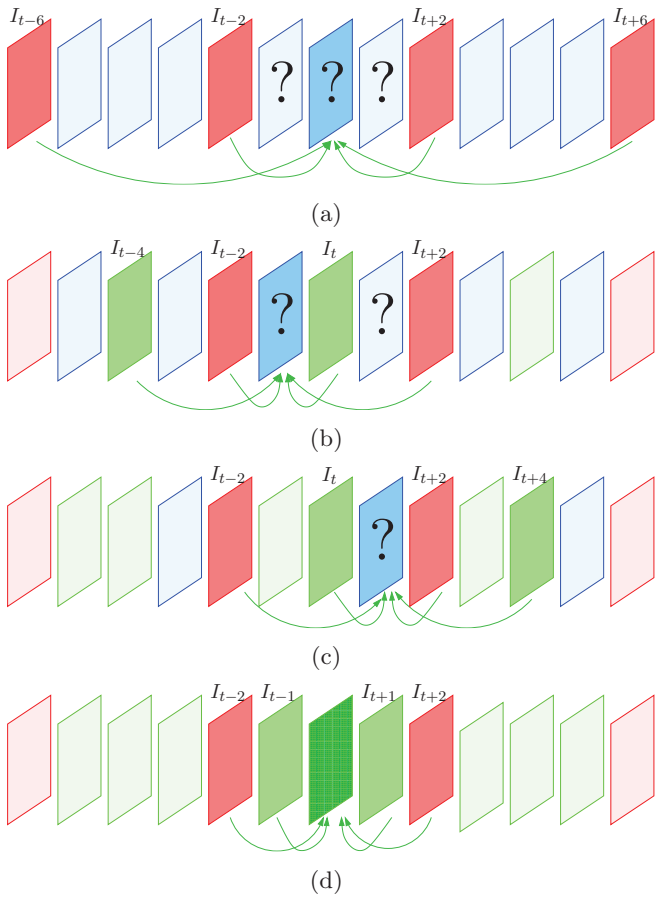


Figure 4.3: WZ frame estimation for GOP size = 4 (the KFs are in red, the WZFs that have to be estimated are in blue, the already decoded WZF are in green. The WZF to be refined is in dark green): at first we estimate I_t (a), after by it, we estimate I_{t-1} (b) and I_{t+1} (c). Finally we refine the central I_t (d)

the side information is re-estimated, the standard deviation changes. On the other hand, if one can ask again for parity bits for the new side information, this would significantly increase the total bit rate. Therefore we prefer to use the same set of parity bits sent for the first SI, rather than requesting a new set. Since the quality of the SI is improved, we still expect that the average PSNR of the decoded WZFs increases after refinement. However, the computational effort at the decoder is increased: indeed, in the proposed method, the side information must be generated twice and must be correct twice.

This method can be extended to the case of a GOP size of 8. The extension is straightforward but long to describe, so we summarize it in the Tab. 4.2. For the interpolation of each frame, the four closest available frames (two backward and two forward ones) are used. When all WZFs are estimated, we can re-estimate these by using the actual four closest available frames (that are different w.r.t. the frames used in the first estimation step).

step	SI construction	references			
GOP size = 4					
1	$\widehat{I}_t$	I_{t-6}	I_{t-2}	I_{t+2}	I_{t+6}
2	$\widehat{I}_{t-1}$	I_{t-4}	I_{t-2}	I_t	I_{t+2}
	$\widehat{I}_{t+1}$	I_{t-2}	I_t	I_{t+2}	I_{t+4}
3	$\widehat{I}_t$	I_{t-2}	I_{t-1}	I_{t+1}	I_{t+2}

Table 4.1: Frames used for the WZF estimation for GOP size equal to 4

step	SI construction	references			
GOP size = 8					
1	$\widehat{I}_t$	I_{t-12}	I_{t-4}	I_{t+4}	I_{t+12}
2	$\widehat{I}_{t-2}$	I_{t-8}	I_{t-4}	I_t	I_{t+4}
	$\widehat{I}_{t+2}$	I_{t-4}	I_t	I_{t+4}	I_{t+8}
3	$\widehat{I}_{t-3}$	I_{t-6}	I_{t-4}	I_{t-2}	I_t
	$\widehat{I}_{t-1}$	I_{t-4}	I_{t-2}	I_t	I_{t+2}
	$\widehat{I}_{t+1}$	I_{t-2}	I_t	I_{t+2}	I_{t+4}
	$\widehat{I}_{t+3}$	I_t	I_{t+2}	I_{t+4}	I_{t+6}
4	$\widehat{I}_{t-2}$	I_{t-4}	I_{t-3}	I_{t-1}	I_t
	$\widehat{I}_{t+2}$	I_t	I_{t+1}	I_{t+3}	I_{t+4}
	$\widehat{I}_t$	I_{t-2}	I_{t-1}	I_{t+1}	I_{t+2}

Table 4.2: Frames used for the WZF estimation for GOP size equal to 8

4.1.4 Experimental results

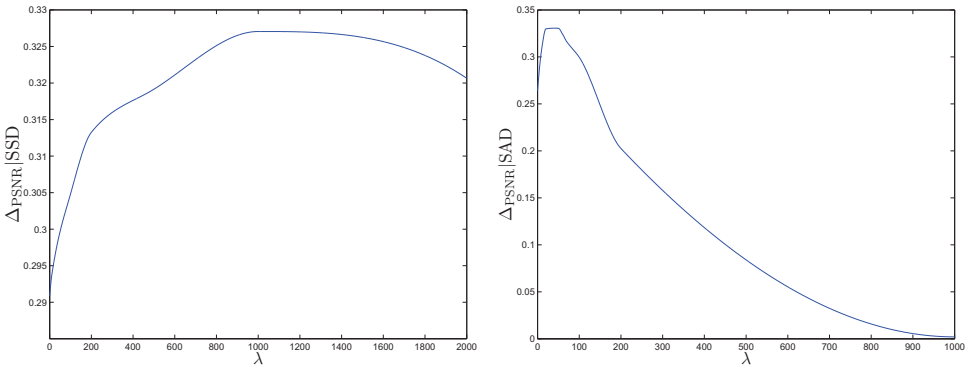
Several experiments have been carried out in order to tune and validate the proposed methods. In a first set of tests, we tuned HOMI algorithm parameters for different configurations and we evaluate the improvement of the quality of the SI w.r.t. DISCOVER. Then, we compare the two different codecs: the original DISCOVER codec (our reference method), and the DISCOVER codec with the SI generation unit implemented by our algorithm HOMI along with its variants. Then, the difference of the two codecs consists only in the SI generation method. We evaluate end-to-end rate-distortion performance of these two codecs using the Bjontegaard metrics [Bjo01]: the PSNR difference Δ_{PSNR} and the percent rate difference Δ_{R} between the decoded frames of our proposed codec w.r.t. the SI generated by DISCOVER MCTI algorithm. All the test sequences are at a CIF resolution at 30 fps. The KFs are coded with the INTRA mode of H.264/AVC with four quantization parameter (QP) values, namely 31, 34, 37 and 40. We use 14 test sequences characterized by different types of motion. A first set (*container*, *akiyo*, *news* and *hall*) is made up of static videos. A second one (*eric*, *paris* and *silent*) presents regular motion. A third set (*waterfall*, *flower*, *foreman*, *tempete*, *football*, *city* and *coastguard*) is characterized by more complex motion.

The parameters that we have to set concern the motion estimation step: the pixel precision of the ME, the metric (*i.e.* the parameter n) and the parameter λ to be used in Eq. (4.1).

We considered full-pixel, half-pixel and quarter-pixel precision. From experimental results, we notice that by increasing precision we have very similar results in terms of SI improvement w.r.t. DISCOVER motion interpolation algorithm, but different computa-

precision	container	akiyo	news	hall	paris	eric	foremsn	silent	mobile	city
full pixel	0.00	0.10	0.04	0.06	0.06	0.33	0.26	0.12	-0.01	0.62
half pixel	0.00	0.09	0.04	0.06	0.06	0.32	0.26	0.12	-0.03	0.61
quarter pixel	0.02	0.11	0.03	0.06	0.05	0.32	0.27	0.10	-0.05	0.60

Table 4.3: PSNR improvement [dB] on the SI w.r.t. DISCOVER for different values of motion estimation precision for QP=31



(a) (b)

Figure 4.4: Δ_{PSNR} for SSD criterion 4.4(a) and for SAD criterion 4.4(b). Fixed the optimal value of λ, the maximum in the two criteria are nearly the same

tion times. The PSNR improvement, as shown in Tab. 4.3, varies on the scale of 0.01 dB. Therefore for the next experiments, we have selected full-pixel precision.

Concerning the metric, we observed that the choice of SAD or SSD gives almost the same results in terms of quality of the SI, provided that we choose a good value of λ, that minimizes the SSD or SAD. In Fig. 4.4, the PSNR improvement w.r.t. DISCOVER MCTI versus λ for the same subset of sequences is shown for $n = 1$ and $n = 2$. We remark that the two curves attain the same maximum value, but for different values of λ. Then, we have chosen the SAD metric that is computationally less heavier than the SSD one.

Then, we search for the optimal value of λ, that minimizes the average PSNR on the SI, and we have obtained the values in Tab. 4.4. The optimal value decreases with the GOP size because, by increasing the latter, the linear approximation given by DISCOVER MCTI is coarser, and then the motion vectors have to be free to assume larger values.

We report in Tab. 4.5, 4.6, 4.7, 4.8 the average PSNR improvement of all sequences of the SI w.r.t. DISCOVER for the different versions of HOMI, for each QP and for different GOP sizes. For the complete results for the different sequences see the appendix. We remark that the gain is always positive, expect for FastHOMI 8 and 4 for GOP size equal to 8, due to the large distance between the frames used for the interpolation (the distance from the first to the last frames used for the interpolation is 32). Indeed, the distance in Eq. (4.2) is too large to assume that the two points belong to the same object in the frame I_{t-1}. HOMI 4 gives the better results for all GOP sizes and all the QPs.

Finally, we consider the complete DVC encoding and decoding procedure and we com-

GOP size	2	4	8
λ_{opt}	50	20	0

Table 4.4: Value of λ_{opt} for different KF distances

QP	GOP size = 2	GOP size = 4	GOP size = 8
31	0.11	0.15	0.17
34	0.08	0.16	0.17
37	0.07	0.13	0.16
40	0.06	0.09	0.13

Table 4.5: PSNR improvement [dB] of the SI w.r.t. DISCOVER for HOMI 8 averaged on all sequences

pare the RD performance obtained with the various versions of HOMI algorithm.

The results of these tests are reported for GOP size equal to 2, 4 and 8, respectively, in Tab. 4.9, 4.10 and 4.11. They confirm the results we have obtained on the improvement of the SI quality. We observe that the global RD performance is almost always improved w.r.t. DISCOVER. We have smaller gains for GOP size equal to 2 (0.83% of bit reduction in average for HOMI8), because the frames used in the interpolation are very close each other and linear interpolation already gives good results. Still, for almost all the test sequences, the proposed method allows some rate reduction, sometimes quite relevant (-4% for *football*), in particular for sequences characterized by irregular motion. Better results are obtained for GOP sizes equal to 4 and 8, because the frames used for the interpolation are farther apart. We obtain a bit rate reduction up to 11.68% and 18.37%, respectively, for GOP sizes equal to 4 and 8. In average we obtain a rate reduction, respectively, of 2.59% and 3.45% and a PSNR improvement of 0.11 dB and 0.14 dB w.r.t. DISCOVER.

Then, we observe that denser MVFs improve the rate distortion quality: HOMI 4 gives always better performance, but the computational complexity w.r.t. HOMI8 is multiplied by four. For GOP size equal to 2 with HOMI8 we obtain a bit reduction of 0.83% versus 1.13% that we obtain with HOMI4. The best improvement is for GOP size equal to 8 with refinement, where with HOMI4 we obtain 1.21% of bit reduction w.r.t. HOMI8.

On the other hand, the fast versions of HOMI have fairly good results, since the rate distortion performances are almost unchanged in many cases, while the computational complexity is halved. For GOP size equal to 4, we obtain 2.59% of bit reduction for HOMI8, while 2.37% for the fast version. Likely, we obtain 2.74% and 2.63% of bit reduction, respectively, for HOMI4 and FastHOMI4. On the contrary, for GOP size equal to 8, the fast versions are not comparable with HOMI8/4. We obtain 0.32% and 0.04% of bit reduction for FastHOMI8 and 4, versus 3.45% and 3.99% for HOMI8 and 4. The frames are farther apart and the distance in Eq. (4.2) is too large to assume that the two points belong to the same object in the frame I_{t-4}.

The refinement method largely improves the rate distortion performance as shown for GOP sizes equal to 4 and 8 in Tab. 4.12 and 4.13: in fact for GOP size equal to 4 the rate reduction passes from 2.59% to 10.13% and for GOP size equal to 8 from 3.45% to

QP	GOP size = 2	GOP size = 4	GOP size = 8
31	0.06	0.14	-0.08
34	0.03	0.14	-0.04
37	0.04	0.11	-0.06
40	0.03	0.11	-0.04

Table 4.6: PSNR improvement [dB] of the SI w.r.t. DISCOVER for Fast HOMI 8 averaged on all sequences

QP	GOP size = 2	GOP size = 4	GOP size = 8
31	0.05	0.13	-0.08
34	0.04	0.15	-0.05
37	0.05	0.12	-0.05
40	0.03	0.11	-0.03

Table 4.7: PSNR improvement [dB] of the SI w.r.t. DISCOVER for Fast HOMI 4 averaged on all sequences

7.42% (HOMI 8). The best bitrate reductions are of 28.75% and 23.20%, respectively, for GOP sizes equal to 4 and 8, Then, re-estimating the WZFs improves the rate-distortion performance w.r.t. the hierarchical decoding of DISCOVER, also if the bit allocation is not optimal. On the other hand, the computational complexity is slightly increased, because two estimations have to be generated.

4.1.5 Side information generation refinement for large GOP sizes with adaptive search area and variable block sizes

In this section, we propose a new method that consists in improving the decoded WZFs, when the GOP size is larger than 2. This improvement is achieved by re-estimating the SI using the neighboring decoded frames as in Section 4.1.3, but with an adaptive search area and a variable block size.

In the method proposed in Section 4.1.3, the central WZF I_t is re-estimated without using the available decoded $\hat{I}_t$. Now, we propose to re-estimate I_t using also the already decoded WZF $\hat{I}_t$, along with the neighboring decoded frames. In particular, we propose an approach to adapt the search area to the real motion between the decoded WZF $\hat{I}_t$ and the previous (or next) decoded frame. This procedure achieves a better estimation for high motion regions. Moreover, it generates more homogeneous motion vectors and reduces the estimation complexity for slow motion regions. First, nine large 24×24 blocks are selected (randomly) in the decoded WZF. The matching for those blocks is performed in order to determine the corresponding blocks in the previous (or next) decoded frame. In this step the search area is set to ± 32 pixels. When a selected block belongs to a homogeneous region, the MAD is almost the same for all candidate blocks in this region. In order to avoid obtaining false large motion vectors in these homogeneous areas, the MAD computed during the matching procedure is replaced by the WMAD. Nine motion vectors associated to those blocks are obtained. These motion vectors are used to adapt the search area. We define the search area by four parameters SA_r, SA_l, SA_t and SA_b,

QP	GOP size = 2	GOP size = 4	GOP size = 8
31	0.17	0.22	0.19
34	0.16	0.20	0.18
37	0.12	0.17	0.19
40	0.07	0.10	0.15

Table 4.8: PSNR improvement [dB] of the SI w.r.t. DISCOVER for HOMI 4 averaged on all sequences

	HOMI8		HOMI4		FastHOMI8		FastHOMI4	
	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}
container	-0.13	0.01	-0.67	0.04	-0.05	0.00	-0.08	0.00
akiyo	-0.29	0.02	-0.40	0.04	-0.05	0.01	0.41	-0.01
news	-0.66	0.04	-1.33	0.09	-0.74	0.05	-0.79	0.05
hall	-0.53	0.03	-1.45	0.09	0.11	-0.01	0.23	-0.02
eric	-1.80	0.10	-1.25	0.07	-1.18	0.06	-1.55	0.08
paris	0.30	-0.02	-0.59	0.04	0.51	-0.03	0.51	-0.03
silent	-0.49	0.02	-1.09	0.05	0.16	-0.01	0.09	-0.00
waterfall	-0.06	0.00	-0.34	0.02	0.04	-0.00	0.04	-0.00
flower	1.03	-0.07	1.47	-0.10	1.80	-0.12	1.77	-0.12
foreman	-1.35	0.07	-3.66	0.18	0.14	-0.01	0.13	-0.01
tempete	0.09	-0.01	0.23	-0.01	0.26	-0.01	-0.07	0.00
football	-4.16	0.21	-6.17	0.31	-2.78	0.14	-2.90	0.15
city	-3.31	0.16	-3.19	0.15	-3.24	0.16	-3.22	0.15
coastguard	0.00	0.00	0.46	-0.02	-0.38	0.02	-0.37	0.02
mean	-0.83	0.04	-1.13	0.06	-0.44	0.02	-0.46	0.02

Table 4.9: Rate Distortion Performance for HOMI w.r.t. DISCOVER MCTI - GOP size = 2

which represent the distance between the center and the right, left, top, and bottom points, respectively, attained by the search area. These parameters are adapted according to the obtained motion vectors $\mathbf{v}_i = (v_{ix}, v_{iy})$ $(i = 1, 2, ..., 9)$ as follows:

$$\begin{cases} \mathrm{SA}_r &= \max\limits_i(v_{ix}), \text{ if } v_{ix} > 0 \\ \mathrm{SA}_l &= -\min\limits_i(v_{ix}), \text{ if } v_{ix} < 0 \\ \mathrm{SA}_t &= \max\limits_i(v_{iy}), \text{ if } v_{iy} > 0 \\ \mathrm{SA}_b &= -\min\limits_i(v_{iy}), \text{ if } v_{iy} < 0 \end{cases}$$

The obtained parameters are used to construct the adapted search area.

We also introduce a variable block size in the re-estimation procedure. We start by dividing the frame I_k into 16×16 blocks. Then, the block matching is carried out, based on the previous and next decoded frames, using the adapted search area. If the obtained MAD is greater than a threshold, this block is divided into 8×8 blocks, and the motion vectors are computed for these four 8×8 blocks. The procedure is stopped at a block size of 4×4 pixels. Finally, the bidirectional motion compensation is applied to obtain the new SI.

	HOMI8		HOMI4		FastHOMI8		FastHOMI4	
	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}
container	0.07	-0.00	-0.74	0.04	-0.35	0.02	-0.43	0.02
akiyo	-2.45	0.11	-0.39	0.03	-0.16	0.03	-2.39	0.08
news	-0.59	0.04	-2.23	0.13	-0.83	0.04	-1.19	0.06
hall	-0.44	0.03	-1.58	0.10	1.20	-0.07	0.53	-0.03
eric	-11.03	0.45	-9.46	0.39	-10.14	0.41	-10.02	0.41
paris	-1.80	0.09	-3.72	0.18	-1.89	0.09	-1.73	0.09
silent	-1.54	0.08	-0.52	0.08	0.87	-0.03	0.29	0.01
waterfall	0.35	-0.01	-0.18	0.00	0.43	-0.02	0.41	-0.02
flower	1.44	-0.08	1.43	-0.07	-0.80	0.04	-0.53	0.03
foreman	-6.27	0.26	-11.68	0.46	-5.77	0.26	-6.12	0.26
tempete	0.20	-0.01	0.75	-0.03	0.22	-0.01	0.36	-0.01
football	-6.59	0.32	-8.24	0.41	-6.31	0.32	-6.71	0.34
city	-8.70	0.32	-3.86	0.17	-9.08	0.31	-8.85	0.30
coastguard	1.04	-0.03	2.06	-0.06	-0.61	0.02	-0.42	0.01
mean	-2.59	0.11	-2.74	0.13	-2.37	0.10	-2.63	0.11

Table 4.10: Rate Distortion Performance for HOMI w.r.t. DISCOVER MCTI - GOP size = 4

	HOMI8		HOMI4		FastHOMI8		FastHOMI4	
	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}
container	0.34	-0.01	-1.60	0.07	0.85	-0.03	0.35	-0.01
akiyo	0.56	0.02	-4.29	0.20	-7.76	0.47	-9.95	0.52
news	-0.66	0.03	0.26	0.01	0.18	0.01	0.33	-0.01
hall	0.37	-0.07	-1.96	0.07	16.16	-0.92	17.27	-0.99
eric	-11.39	0.48	-10.39	0.47	-2.66	0.17	-3.05	0.18
paris	-5.96	0.26	-9.10	0.39	-3.18	0.13	-3.18	0.14
silent	-2.27	0.09	-0.87	-0.16	-3.40	0.15	-3.44	0.14
waterfall	0.65	-0.00	0.49	0.01	-0.77	0.05	-0.62	0.04
flower	0.45	-0.02	-3.21	0.14	12.61	-0.57	12.86	-0.58
foreman	-7.60	0.31	-12.45	0.47	-6.33	0.23	-5.84	0.19
tempete	-0.13	0.01	0.63	-0.03	3.74	-0.16	4.04	-0.17
football	-6.39	0.38	-7.72	0.50	-5.57	0.41	-5.72	0.43
city	-18.37	0.55	-7.63	0.32	-18.34	0.52	-14.32	0.39
coastguard	2.05	-0.03	1.97	-0.06	10.06	-0.23	10.71	-0.24
mean	-3.45	0.14	-3.99	0.17	-0.32	0.02	-0.04	0.00

Table 4.11: Rate Distortion Performance for HOMI w.r.t. DISCOVER MCTI - GOP size = 8

4.1.6 Experimental results

With the aim of evaluating the performance of the proposed methods, we perform extensive simulations, adopting the same test conditions as described in DISCOVER (in Section 3.3), *i.e.* test video sequences are at QCIF spatial resolution and sampled at 15 frames/sec. The proposed SI refinement method compared w.r.t. DISCOVER codec in terms on rate distortion performances.

The RD performance is shown for the Stefan, Foreman, Bus, Coastguard, Soccer and Hall sequences in Tab. 4.14, in comparison to the DISCOVER codec, using the Bjontegaard metric [Bjo01] , for a GOP size equal to 4 and 8. The gains for GOP size equal to 4 are very good: we obtain a bit reduction up to 23.28% amnd a PSNR improvement of 1.19 dB w.r.t. DISCOVER.

The proposed method consists in a refinement of the frames decoded by DISCOVER.

The gains become even more significant for a GOP size equal to 8. In fact, we obtain

	HOMI8		HOMI4		FastHOMI8		FastHOMI4	
	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}
container	-1.60	0.08	-2.90	0.14	-7.41	0.36	-7.50	0.36
akiyo	-3.31	0.22	-1.66	0.17	6.52	-0.02	1.25	0.04
news	-0.02	0.05	-1.03	0.10	-4.73	0.26	-5.06	0.28
hall	-2.94	0.16	-4.84	0.28	-0.23	0.02	-1.03	0.06
eric	-23.71	0.98	-23.20	0.96	-18.02	0.73	-17.52	0.71
paris	-8.75	0.43	-10.32	0.51	-7.06	0.34	-6.96	0.34
silent	-11.19	0.52	-10.50	0.51	-9.99	0.43	-10.66	0.48
waterfall	-6.36	0.19	-6.89	0.22	-11.40	0.35	-11.43	0.36
flower	-4.03	0.21	-4.08	0.21	-1.69	0.09	-1.43	0.08
foreman	-24.07	1.11	-28.75	1.31	-13.79	0.61	-14.25	0.61
tempete	-3.74	0.17	-3.22	0.15	-1.13	0.05	-0.99	0.05
football	-23.50	1.15	-25.67	1.28	-12.66	0.67	-12.99	0.69
city	-10.63	0.41	-8.33	0.33	-11.35	0.40	-10.87	0.39
coastguard	-18.09	0.51	-17.18	0.49	-14.18	0.39	-13.94	0.38
mean	**-10.13**	**0.44**	**-10.61**	**0.47**	**-7.65**	**0.33**	**-8.09**	**0.34**

Table 4.12: Rate Distortion Performance for HOMI + refinement w.r.t. DISCOVER MCTI - GOP size = 4

	HOMI8		HOMI4		FastHOMI8		FastHOMI4	
	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}	Δ_R	Δ_{PSNR}
container	1.45	-0.08	-0.38	0.00	1.71	-0.08	1.02	-0.05
akiyo	3.04	-0.03	-4.67	0.23	-5.69	0.44	-7.38	0.43
news	0.21	-0.05	1.40	-0.06	0.91	-0.06	0.82	-0.06
hall	2.90	-0.18	-0.27	-0.01	18.69	-1.10	19.75	-1.08
eric	-19.04	0.85	-18.74	0.85	-11.72	0.59	-12.09	0.59
paris	-8.31	0.36	-11.68	0.50	-5.90	0.24	-5.90	0.25
silent	-7.20	0.24	-6.87	-0.00	-8.77	0.31	-8.88	0.31
waterfall	-4.69	0.10	-4.94	0.12	-5.61	0.13	-5.50	0.13
flower	-5.01	0.22	-7.71	0.34	5.74	-0.28	5.51	-0.26
foreman	-18.11	0.79	-22.96	0.95	-17.45	0.68	-17.37	0.65
tempete	-2.28	0.08	-2.11	0.06	1.88	-0.09	2.20	-0.10
football	-12.25	0.82	-13.44	0.97	-10.15	0.79	-10.58	0.85
city	-22.74	0.68	-15.28	0.52	-23.20	0.67	-21.40	0.60
coastguard	-11.92	0.27	-13.28	0.26	-4.61	0.07	-4.28	0.06
mean	**-7.42**	**0.29**	**-8.63**	**0.33**	**-4.58**	**0.16**	**-4.55**	**0.16**

Table 4.13: Rate Distortion Performance for HOMI + refinement w.r.t. DISCOVER MCTI - GOP size = 8

a bit reduction up to -30.72%, which corresponds to an improvement of 1.55 dB on the decoded frames w.r.t. DISCOVER codec. These maxima of performance gain are obtained for the Foreman sequence.

sequence	Stefan	Foreman	Bus	Coastguard	Soccer	Hall
GOP = 4						
Δ_R [%]	**-22.79**	**-23.28**	**-13.17**	**-9.13**	**-19.65**	**-3.96**
Δ_{PSNR} [dB]	**1.32**	**1.19**	**0.68**	**0.38**	**0.96**	**0.24**
GOP = 8						
PropA						
Δ_R [%]	**-27.92**	**-30.72**	**-20.12**	**-15.50**	**-24.49**	**-6.79**
Δ_{PSNR} [dB]	**1.60**	**1.55**	**1.00**	**0.65**	**1.18**	**0.31**

Table 4.14: Rate distortion performance comparison for a GOP size equal to 4 and 8, w.r.t. DISCOVER codec, using Bjontegaard metric.

4.2 Inter-view interpolation with priority to large disparity (IPLD)

In this section, we introduce a novel inter-view interpolation algorithm for multiview DVC, for the case of aligned cameras or rectified sequences. In these cases, the inter-view estimation is commonly produced by adapting the motion-compensated temporal interpolation algorithms, since the object "trajectories" along views are practically straight lines. As a consequence, the DISCOVER algorithm works fairly well in this case, and considering higher order interpolation does not lead to significant gains (see results listed in the Appendix). However, DISCOVER MCTI does not take into account occlusions, so we can try to improve upon it by addressing this issue.

The key idea for our algorithm is the following: whenever two objects (actually, two blocks of pixels) of the reference KFs have two estimated trajectories both passing near the current position, we take into account the object depth, instead of simply selecting the trajectory closer to the current position, as DISCOVER would do in the splitting procedure (see Eq. (3.1)). The foreground objects, having smaller depths, occlude the background and should be preferred for the SI generation. Therefore we modify Eq. (3.1) by adding a penalization for small disparities (which translates into large depths). We obtain the following equation:

$$\mathbf{q}^*(\mathbf{p}) = \arg\min_{\mathbf{q}} \left(\left\| \mathbf{q} + \frac{1}{2}\mathbf{v}(\mathbf{q}) - \mathbf{p} \right\|^2 - \gamma \left\| \mathbf{v}(\mathbf{q}) \right\|^2 \right) \tag{4.3}$$

where the penalization parameter $\gamma > 0$ is to be chosen experimentally. The proposed method consists in replacing Eq. (3.1) of DISCOVER with Eq. (4.3); the other steps of DISCOVER stay unaltered. We call this method "interpolation with priority to large disparity" (IPLD).

In Fig. 4.5 we show a schematic example where IPLD allows selecting the correct trajectory. In this figure, the y axis of frames is orthogonal to the drawing plane. After the first disparity estimation, the disparities[1] for blocks centered in $\mathbf{q}_1$ and $\mathbf{q}_2$ are respectively

[1]The "disparity vector" has of course a null vertical component. This vectorial notation allows a

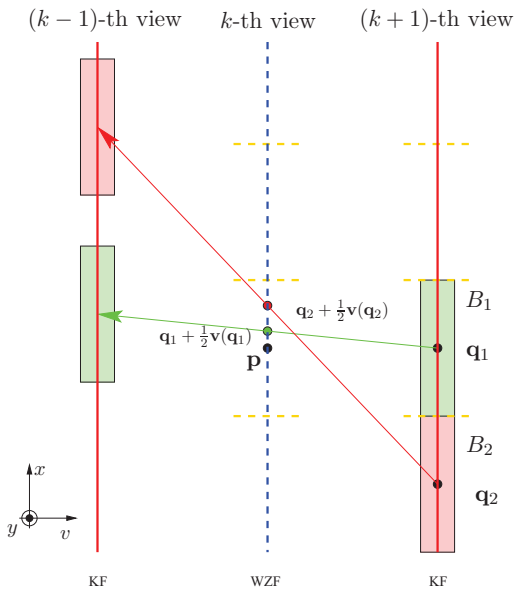


Figure 4.5: An 1-D representation of a possible situation of two disparity vectors after the forward disparity estimation of DISCOVER. With DISCOVER, we would choose the green vector in the splitting procedure because it is closer to $\mathbf{p}$ than the red vector: but arguably the object in the red block occludes the object in the green block within the frame in the k-th view

$\mathbf{v}(\mathbf{q}_1)$ and $\mathbf{v}(\mathbf{q}_2)$. Both blocks B_1 and B_2 could be interpolated in position $\mathbf{p}$ in the WZF. DISCOVER would simply select the block whose trajectory is closer to $\mathbf{p}$, B_1 in this case. On the contrary, with our method we should select the block that occludes the other, B_2 in this case. In order to estimate the occlusion, we penalize blocks with small disparities, since they probably belong to the background. Thus, provided that a suitable value for the penalization parameter γ is chosen, IPLD is able to correctly select the block B_2.

This algorithm is well suited for inter-view estimation, because in this case the disparity is directly related to the depth. However, it would not be as much efficient for temporal interpolation, since objects with higher velocity do not necessarily occlude objects with smaller velocity, even though, if two objects have the same speed, the one closer to the camera will have a larger apparent velocity. This intuition is confirmed by preliminary experiments: no gain is obtained by adapting IPLD to temporal interpolation.

The increase of computational complexity of IPLD with respect to DISCOVER is negligible: when Eq. (3.1) is replaced by Eq. (4.3), for each candidate trajectory we only need the computation of $-\gamma\|\mathbf{v}(\mathbf{q})\|^2$ and an additional sum. This computational cost is dominated by two multiplications. Now, since in the splitting step only a small number of candidate trajectories is considered, we estimate the increase of computational complexity of our method in less than 20 multiplications per block. This is to be compared to the much heavier cost of motion estimation in the DISCOVER algorithm.

formulation coherent with the previous notation.

In Fig. 4.6, we show an inter-view interpolated image from the *newspaper* and *book arrival* sequences, using DISCOVER (a) and IPLD (b): our technique is much more successful in discriminating the background from the foreground.

4.2.1 Experimental results

For all the experiments in this subsection, we consider an asymmetric scheme (Fig. 3.12(a)).

In a first set of experiments, we have to set the value of parameter γ of Eq. (4.3), which defines the penalization for small disparity (i.e. background) blocks. We test our algorithm on five rectified multiview video sequences, with 20 frames per sequence. The neighboring views are encoded with H.264/AVC INTRA mode at QP=31. In our experiments, the value maximizing the PSNR of the SI with respect to the WZF is $\gamma = 0.6$: therefore we keep this value in the following.

In a second experiment, we compute the SI for all the test sequences using different QP for the reference KFs. The resulting average PSNR (with respect to the original WZF) of the Wyner-Ziv camera is compared with the one achieved by DISCOVER, and the difference is reported in Tab. 4.15. We observe that the proposed method improves almost always the DISCOVER quality, sometimes with very significant gains (up to 3.34 dB). This is because, as expected, our method succeeds at well reconstructing objects belonging to the foreground. We observe that for such sequences as *lovebird*, *pantomime*, *outdoor*, the proposed method does not improve with respect to DISCOVER, since they have a small range of disparity values: in such case, using disparity to discriminate foreground and background is less successfull. On the contrary, when the disparity range is larger, the improvement is remarkable.

In a last experiment, we compare the end-to-end rate-distortion performances of a complete DVC system using IPLD for inter-view interpolation with another using DISCOVER, *i.e.* the temporal interpolation described in Section 3.3.2 is applied along the view domain. The results, shown in Tab. 4.16, are related to the Wyner-Ziv camera only, since the encoding of the KFs are identical in the two cases. We observe that the large improvement in the SI quality shown in Tab. 4.15 is actually translated into a significant bit-rate reduction for the end-to-end system, in particular for such sequences as *newspaper* and *balloons*, where background and foreground are hard to be distinguished. For these sequences, we obtain a bit rate reduction more than 20 % and a gain in PSNR of around 1 dB. As for the previous experiment, for the sequences with small disparity range, we observed some small losses. However, in the average, the IPLD allows a bit-rate reduction of more than 7.7 % achieved with a very small complexity increase.

sequence / QP	31	34	37	40
balloons	2.21	2.26	2.23	2.23
book arrival	0.09	0.46	0.43	0.41
door flowers	0.86	0.70	0.65	0.50
leaving laptop	0.98	0.88	0.60	0.55
kendo	1.10	0.92	0.80	0.75
lovebird	-0.52	-0.33	-0.28	-0.17
newspaper	3.34	3.13	3.05	2.95
pantomime	0.00	0.00	-0.11	-0.11
outdoor	-0.10	-0.02	-0.02	-0.05
mean	0.88	0.89	0.81	0.78

Table 4.15: SI improvement [dB]: IPLD versus DISCOVER

	Δ_{R} [%]	Δ_{PSNR} [dB]
balloons	-22.71	1.05
book arrival	-5.36	0.53
door flowers	-5.40	0.24
leaving laptop	-7.62	0.32
kendo	-9.26	0.50
lovebird	2.38	-0.09
newspaper	-22.09	0.97
pantomime	0.19	0.00
outdoor	0.35	-0.03
mean	**-7.72**	**0.38**

Table 4.16: Bjontegaard metrics: IPLD versus DISCOVER (WZF camera, asymmetric scheme)

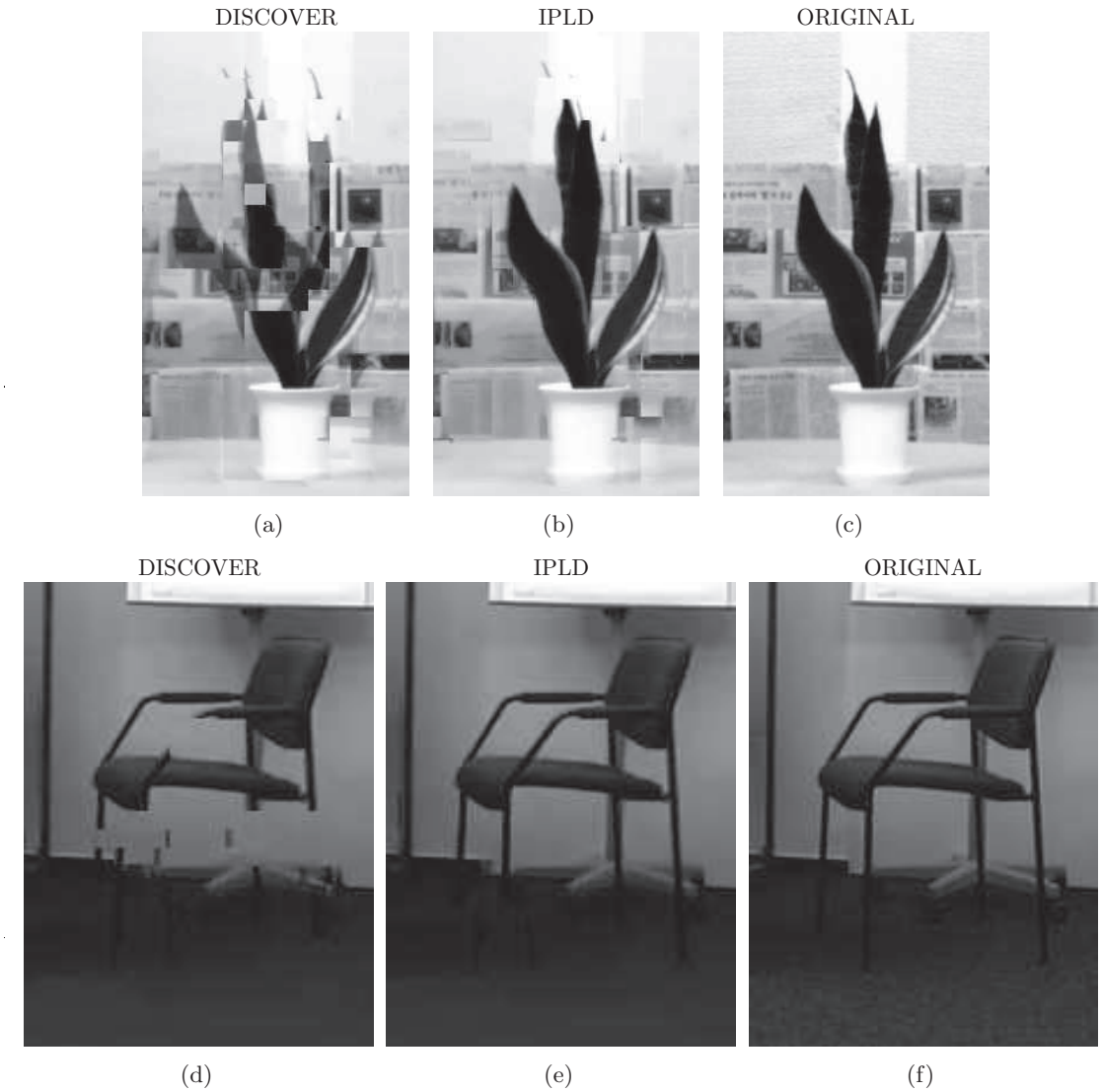


Figure 4.6: A detail of the *newspaper* and *book arrival* sequences: inter-view interpolation generated by DISCOVER (left, PSNR = 15.1 dB (a) for *newspaper* and PSNR = 29.81 dB (d) for *book arrival*), by IPLD (center, PSNR = 28.3 dB (b) for *newspaper* and PSNR = 37.49 dB (e) for *book arrival*), and the original image (right).

4.3 Fusing temporal and inter-view interpolation

Since HOMI and IPLD allow producing good temporal and inter-view interpolations of images, we could consider a multiview DVC system that just uses an existing fusion algorithm (such as one of those proposed in [MMCPP09a]) on these improved SI images: we expect that such a solution outperforms a reference system based on state-of-the-art interpolation methods. However, we can further improve the system performance with a novel fusion scheme based on occlusion detection.

Moreover, as it was observed in the past (e.g. in [MMCPP09a]), even though for the majority of video sequences fusion improves upon single temporal and inter-view interpolations, for some others the opposite is true. In other words, there are sequences for which temporal or inter-view interpolations are much better than fusion: as a consequence, the RD performance comparison between fusion and interpolation is not very consistent, even though in the average fusion has been reported for being the best solution.

In the second part of this section we propose a novel Bayesian method for deciding among fusion, temporal interpolation or inter-view interpolation, based on the observation of the parity bit-rate needed by the decoder. The resulting adaptive fusion method shows much more consistent behavior and improved performance with respect to previous techniques.

4.3.1 Occlusion detection-based fusion

For the fusion problem, we consider a scheme where a WZF $I_{t,k}$ has four neighboring KFs: $I_{t,k-1}$, $I_{t,k+1}$, $I_{t-1,k}$, $I_{t+1,k}$. For example, we can consider a hybrid camera in a Hybrid 1/2 scheme (Fig. 3.12(a)), or any camera (except the first and the last) in a Symmetric 1/2 scheme (Fig. 3.12(b)). For the time being, any temporal interpolation and inter-view interpolation algorithms can be used to produce the partial estimations $\widehat{I}_T$ and $\widehat{I}_V$. The proposed fusion between these images is performed as follows.

Let us suppose that we have the disparity fields $d_{k-1,k+1}(m,n)$ (from $I_{t,k-1}$ to $I_{t,k+1}$) and $d_{k+1,k-1}(m,n)$ (from $I_{t,k+1}$ to $I_{t,k-1}$). They can be obtained as side-product of the inter-view interpolation process, or they can be estimated from the scratch for the fusion process: in our implementation we used the method proposed by Miled and Pesquet [MP06] on the decoded KFs.

The coherence of the disparity fields is estimated by a left-right consistency cross-check (LR-CC) [Fua93a, EW02, SS03]: if the pixel $\mathbf{p} = (m,n)$ in the image k_1 corresponds to the pixel $(m, n + d_{k_1,k_2}(m,n))$ in the image k_2, the disparity of the former should be the opposite of the disparity of the latter. Therefore, the absolute sum of these quantities is a measure of the disparity coherence in the direction from k_1 to k_2. We can write:

$$R_{k_1,k_2}(m,n) = |d_{k_1,k_2}(m,n) + d_{k_2,k_1}(m, n + d_{k_1,k_2}(m,n))| \tag{4.4}$$

For a perfectly coherent disparity, we would obtain $R = 0$ everywhere. However we take into account a tolerance term $\tau \geq 0$. In the scientific literature, a common value for τ is 1 pixel [Hir01, YK06], at least in the context of stereo matching. Since our target is slightly different (image fusion in the context of multiview DVC), we do not use the value in literature, but rather optimize it experimentally, as detailed in Section 4.3.3. We define the occlusion mask from k_1 to k_2 as follows:

$$O_{k_1,k_2}(m,n) = u(R_{k_1,k_2}(m,n) - \tau) \tag{4.5}$$

where $u(\cdot)$ is the Heaviside step function, defined as

$$u(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

In other words, $O_{k_1,k_2}(m,n)$ is zero if $R(m,n)$ is less than τ, and one if $R(m,n)$ is greater than τ. Therefore, using Eq. (4.5), we perform an occlusion detection: O_{k_1,k_2} shows the points of k_1 occluded (i.e. not visible) in k_2.

We use the occlusion masks $O_{k-1,k+1}$ and $O_{k+1,k-1}$ to improve the fusion process. For each pixel $\mathbf{p}$, we decide whether to use only the temporal interpolation (since an occlusion has been detected) or to use a fused image, such as the linear fusion proposed in [MMCPP09a]. Now, the inter-view estimation $\widehat{I}_V(\mathbf{p})$ has been produced (using DISCOVER or IPLD) as the average of two disparity-compensated values:

$$\begin{aligned} \widehat{I}_V(m,n) = &\frac{1}{2} I_{t,k-1}(m, n + e(m,n)) \\ &+ \frac{1}{2} I_{t,k+1}(m, n + f(m,n)) \end{aligned} \tag{4.6}$$

where e and f are respectively the estimations of $d_{k,k-1}$ and $d_{k,k+1}$ and they have been obtained by splitting $d_{k-1,k+1}$. Looking at Eq. (4.6), we argue that $\widehat{I}_V(\mathbf{p})$ is affected by the possible occlusion of the pixel $(m, n + e(m,n))$ in the left image and of the pixel $(m, n+f(m,n))$ in the right image. Therefore, the proposed fusion formula is the following:

$$\widetilde{I}(\mathbf{p}) = \begin{cases} \widehat{I}_T(\mathbf{p}) & \text{if } O_{k-1,k+1}(m, n + e(m,n)) = 1 \\ \widehat{I}_T(\mathbf{p}) & \text{if } O_{k+1,k-1}(m, n + f(m,n)) = 1 \\ \widetilde{I}_{\text{LIN}}(\mathbf{p}) & \text{otherwise} \end{cases}$$

In other words, we use only the temporal interpolation if one or both the two pixels that contribute to $\widetilde{I}(\mathbf{p})$ are occluded according to our method. Otherwise we use the linear fusion proposed in [MMCPP09a] and defined in Eq. ((3.2)), indicated as $\widetilde{I}_{\text{LIN}}(\mathbf{p})$, with the only difference that we merge the estimations obtained by HOMI and IPLD instead of those obtained by DISCOVER.

We observe explicitly that in order to obtain our estimation of the WZF, we use four disparity fields ($d_{k-1,k+1}$, $d_{k+1,k-1}$, e and f), and this feature characterizes our method with respect to existing techniques based on LR-CC.

4.3.2 Adaptive validation

As shown in the scientific literature, and confirmed in our tests, even though fusion improves the quality of side information on the average, for some sequences this is not true: on the contrary, the image generated by fusion is much worse than the one generated by inter-view interpolation only or temporal interpolation only. This happens for example for sequences where one interpolation is much better than the other. For these sequences, the correlation along one axis (temporal or inter-view) is much stronger than the other.

Therefore, we would need a method to decide which image among $\widehat{I}_T$, $\widehat{I}_V$ and $\widetilde{I}$ should be used as side information. This process can be seen as a final step of the fusion process, which validates $\widetilde{I}$, or goes back to $\widehat{I}_T$ or $\widehat{I}_V$: we call the proposed method adaptive validation (AV).

We recall that the best side information is the one that requires the smallest bit-rate in order to be corrected by the channel decoder. Let us consider a given WZF $I_{t,k}$, and let us call R_T, R_V and R_F the bit-rate (in bits per pixel) for correcting temporal, inter-view and fused SI respectively. The central idea of the proposed method is very simple: we take the decision among the three images by observing, for a small number of WZFs, the parity bit rate needed to correct the temporal and the inter-view interpolations. The rationale behind this idea is that fusion is ineffective only when one of the two interpolations is much worse then the other. Therefore it suffices to observe the two rates R_T and R_V to estimate whether fusion is viable or not, while the value of R_F is less relevant at this end.

Let us first discuss about the rate overhead associated with the evaluation of R_T and R_V. The parity bits needed to correct $\widehat{I}_T$ and $\widehat{I}_V$ can be related to different transform bands and to different bitplanes. In the worst case, the sets of parity bits are disjoint, so the rate needed to send both sets is $R_T + R_V$. In the most favorable case, one set of parity bits is a subset of the other: then, it suffices to send the parity bits needed for the worst case: the rate is $\max(R_T, R_V)$. In the general case, we need a bit-rate between these two extreme cases. In order to do this, a small modification to the feedback channel and to the buffer is needed: if a subset of the parity bits is already sent for correcting the first SI, then for correcting the second SI, the common subset of parity bits is not needed to be sent because it is already available to the decoder.

Of course, if we request the two sets of parity bits for all the WZFs, this incurs a large rate overhead and we end up with a very inefficient system, which, in the best case, matches the performance of the worst side information. On the contrary, instead of using $R_T + R_V$ or $\max(R_T, R_V)$ bits for a generic image, we would like to use a bit budget of

R_D bits, where D is the optimal decision, i.e.:

$$D = \arg \min_{d \in \{T,V,F\}} R_d \tag{4.7}$$

We estimate the optimal decision by observing the additional parity bit-rate only for n over N WZFs, and for the following $N - n$ we use this decision. More precisely, for n frames out of N, we ask the channel coder the correction bits for both $\widehat{I}_T$ and $\widehat{I}_V$. We call $\delta_R = R_T - R_V$ the difference between the two rates. We expect the value of δ_R to be quite correlated to the optimal decision: if this parameter is large in absolute value, one estimation is much better than the other, so it should be used; if it is small, the two interpolations have close quality, so the fusion should be used. This threshold-based approach is actually equivalent to a maximum a posteriori (MAP) estimation of the optimal decision.

Let us model the optimal decision D as a discrete random variable with three possible values $\{T, V, F\}$. We want to estimate the best decision (according to Eq. (4.7)) given the parameter δ_R, modeled as a continuous random variable. We use the MAP value, obtained by applying the Bayes' law:

$$\begin{aligned} \widehat{D} &= \arg\max_d P(D = d|\delta_R) \\ &= \arg\max_d \frac{p(\delta_R|D = d)P(D = d)}{p(\delta_R)} \\ &= \arg\max_d p(\delta_R|D = d)P(D = d) \end{aligned} \tag{4.8}$$

where $P(\cdot)$ is the probability mass function (marginal or conditional) for D and $p(\cdot)$ is the probability density function for δ_R (marginal or conditional). In order to solve this problem, we have to find the three functions $f_d(\delta_R)$ for $d \in \{T, V, F\}$, defined as:

$$f_d(\delta_R) = p(\delta_R|d)P(d).$$

These functions are estimated off-line on a training set. A couple of further issues has to be addressed in order to be able to use this method. First, as it was observed in some preliminary tests, the f_d functions vary with the QP, so different thresholds must be determined as a function of it. Second, we have to decide how frequently to update the decision $\widehat{D}$ during the coding process. We compute δ_R for n WZFs out of N frames. For these n frames we pay an overhead rate cost, since we ask twice the channel bits to the encoder. Therefore we cannot update δ_R too often, otherwise we lose all the advantage of having a better side information. On the other hand, a too small ratio n/N could affect the reliability of the decision. This dilemma is solved experimentally but we can anticipate that small values of the n/N ratio (in the order of 1/100) work very well if scene changes do not occur.

4.3.3 Experimental results for fusion with occlusion detection

Fusion techniques can be used when both temporal and inter-view interpolation are available: therefore we can consider the hybrid cameras in a Hybrid 1/2 scheme, or all the views (except for the first and the last) in a Symmetric 1/2 scheme. The side information produced in such a scheme depends on the two interpolation techniques and on the fusion. In order to isolate the contribution of each element to the global performance, we consider the following cases:

DT: SI generated with DISCOVER along the temporal axis;

HT: SI generated with HOMI along the temporal axis;

DV: SI generated with DISCOVER along the view axis;

IV: SI generated with IPLD along the view axis;

FD: SI obtained by linear fusion of DV and DT as in [MMCPP09a];

FHI: SI obtained by linear fusion of HT and IV;

OD: FHI with the occlusion detection;

AV: OD with adaptive validation of fusion.

Comparing all these methods allows us to have a deeper insight on the contribution of the proposed tools to the final performance. The FD method is considered the reference since its performances are better than much of the state of the art, and only slightly worse than (but very close to) more recent fusion techniques, as shown in [Duf11].

The first experiments are conducted to set the disparity tolerance τ in Eq. (4.5). To this end, we evaluate the PSNR of the SI information generated with the OD method for 10 frames of 7 test sequences, for four different QPs. We repeat the test by varying the value of τ, and the resulting SI PSNR is reported in Fig. 4.7. We find that the best value is $\tau = 2$, and this is independent from the QP.

Now, in order to assess the impact of OD, we compute the RD performance of OD and FHI with respect to the reference FD [MMCPP09a]. The results are shown in Tab. 4.17. We observe that just using better temporal and inter-view interpolation than DISCOVER (FHI column) allows a non negligible gain in the average (1.6 % bit-rate reduction). These results are fairly improved when the occlusion detection technique is used. In the average, we gain almost another 5 %, achieving a remarkable 6.57 % average rate reduction. However, for some sequences (*pantomime*, *outdoor*), the OD approach actually worsen performances, due to false-positive errors on the occlusion detection.

Some insights about the impact of different methods is shown in Fig. 4.8. In the first row we show a detail of the sequence *balloons*, for which three views are available (left, center and right). We notice that are several occluded areas between the two external

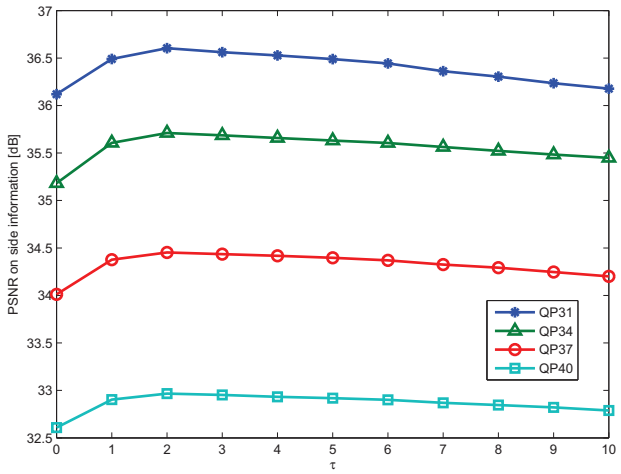


Figure 4.7: PNSR of the fused SI versus the value of τ for different QPs

	FHI		OD	
sequence	Δ_R [%]	Δ_{PSNR} [dB]	Δ_R [%]	Δ_{PSNR} [dB]
balloons	-9.42	0.51	-29.69	1.66
book arrival	-0.96	0.06	-2.73	0.16
door flowers	-0.22	0.01	-6.12	0.29
leaving laptop	-2.18	0.09	-3.66	0.17
kendo	-1.76	0.09	-2.67	0.16
lovebird	0.46	-0.02	-6.73	0.30
newspaper	-4.27	0.23	-18.78	0.98
pantomime	3.08	-0.17	4.18	-0.23
outdoor	0.91	-0.06	7.08	-0.45
mean	**-1.60**	**0.08**	**-6.57**	**0.34**

Table 4.17: RD performance of fusion methods vs. a state-of-the-art technique [MMCPP09a].

views, such as the musical notes on the background. In this case, the block-based IPLD (bottom-left) does not provide a good estimation of the central view: the blocks near the balloon contours are badly estimated. Using fusion (in particular FHI), we achieve a significant improvement, as shown in the bottom center figure. The pixel-wise adaptivity of the fusion allows to use temporal interpolation where the inter-view is judged not reliable; yet, there are some residual artifacts, since not all the occlusions are correctly detected. When we apply the occlusion detection technique, we can then exclude the inter-view contribution from the fusion for the affected pixels, and this further improves the quality of the SI (bottom-right figure).

4.3.4 Adaptive validation results

Now we consider the AV technique. The first experiments are devoted to the estimation of the distribution of δ_R, the rate difference between the corrections of $\widehat{I}_T$ and $\widehat{I}_V$, given the

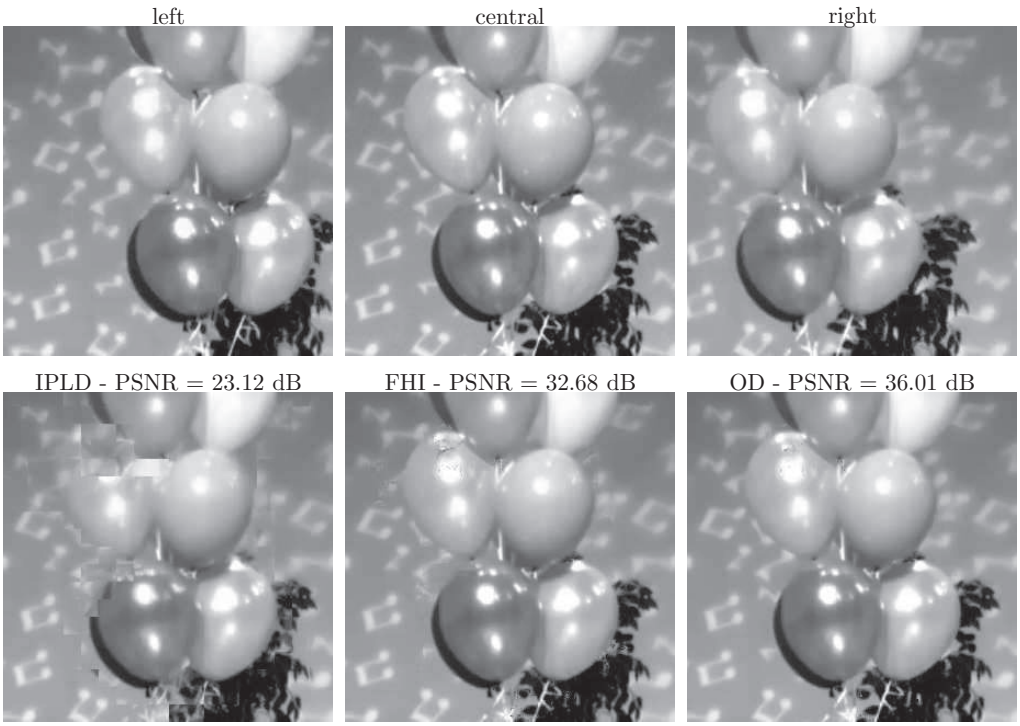


Figure 4.8: In the first row, a detail of the left, the central and the right view of sequence *balloons* are shown. The central one has to be estimated. In the second row the SI obtained by IPLD, FHI, OD are shown

QP	31	34	37	40
T_1	-0.13	-0.08	-0.06	-0.03
T_2	0.06	0.05	0.04	0.03

Table 4.18: Optimal values of T_1 and T_2, estimated off-line

optimal decision D. This process is performed off-line. We consider a training set of 100 frames from the seven multi-view sequences. For each frame, we compute the temporal interpolation $\widehat{I}_T$, the inter-view interpolation $\widehat{I}_V$ and the fusion with the OD algorithm, $\widetilde{I}$. Then we evaluate the number of parity bits needed to correct the three images. This information gives us both the optimal decision (the one associated to the minimum rate) and the δ_R parameter. Repeating this operation for all the frames and for the four QP values, we obtain a large set of samples from the distribution of $\delta_R|D$, along with the samples from the distribution of D. The relative frequencies of the latter constitute the estimation of the PMF of D. The samples of $\delta_R|D$ can be used to estimate the corresponding PDF, using the Parzen window method [Par62]. We use thus a nonparametric estimation of these PDF's: this is a reasonable approach since our *a priori* knowledge of the problem hardly suggest a mathematical model for these distribution.

We are now able to compute the f_d functions: in Fig. 4.9 we show them for QP=37. It is obvious that selecting the function with the maximum value for a given δ_R amounts to compare this parameter with a couple of thresholds, given by the intersections of the f_d

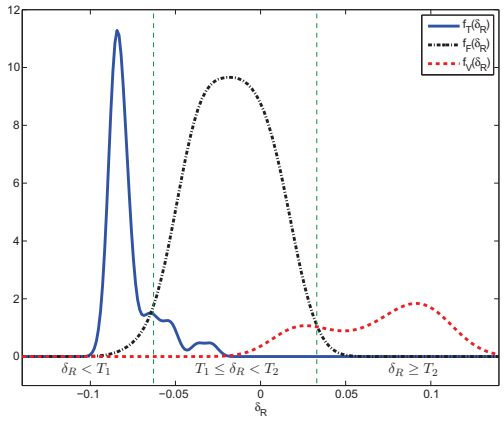


Figure 4.9: Functions $f_d(\delta_R)$ for QP = 37 : if $\delta_R < T_1$, temporal interpolation is chosen, if $T_1 \leq \delta_R < T_2$ we decide for fusion, and if $\delta_R \geq T_2$ the inter-view interpolation is used

N	$n = 1$		$n = 3$	
	Δ_R[%]	Δ_{PSNR} [dB]	Δ_R[%]	Δ_{PSNR} [dB]
10	-4.17	0.25	6.92	-0.33
20	-6.93	0.40	-0.90	0.10
50	-8.49	0.49	-5.11	0.33
100	-9.11	0.42	-7.70	0.45

Table 4.19: Average Rate Distortion Performance for AV with respect to FD [MMCPP09a] for different value of n and N

curves. Our experiments allow us to find these thresholds for different QP values. These values, reported in Tab. 4.18, have been estimated off-line and will be used for all the sequences. Note also that the data used for these estimations and the data used for the validation form two disjoint sets.

We have also to find the optimal update frequency for δ_R, i.e. the optimal values for parameters n and N defined in section 4.3.2. In order to do this, we simply run the adaptive fusion algorithm with different values for these parameters as shown in Tab 4.19. We find that the global RD-performance (measured by the Bjontegaard metric with respect to our reference FD) improves when N increases and when n decreases. In particular we observe that they roughly depend on the ratio n/N rather than separately on these two parameters. In conclusion, this experiment allows us to conclude that, even if we have a large N and we use as few as $n = 1$ image to take the decision, we have very good performance, with a very small overhead since n/N is small. This means that the optimal decision for fusion stays almost unchanged within a given sequence, and then the rate overhead due to a too frequent update is not worth. Therefore, in the following we use $n = 1$ and $N = 100$, and so we have a (small) rate overhead only for one over one hundred frames. Nevertheless, this allows us to take the best decision for the next WZFs. These results seem to point out that the update frequency could be as low as the scene-change frequency, since within a single sequence the best decision seems not to change. As a

	AV	oracle		
	decision	time [%]	view [%]	fusion [%]
balloons	time	90	0	10
book arrival	fusion	2	0	98
door flowers	fusion	24	0	76
leaving laptop	fusion	4	0	96
kendo	fusion	48	28	24
lovebird	time	100	0	0
newspaper	time	98	0	2
outdoor	view	0	76	24
pantomime	fusion	30	48	22

Table 4.20: Decision for each sequence and percentage of time/view/fusion interpolated frames for optimal decision

	HT		IV		OD		AV	
sequence	Δ_R [%]	Δ_{PSNR} [dB]	Δ_R [%]	Δ_{PSNR} [dB]	Δ_R [%]	Δ_{PSNR} [dB]	Δ_R [%]	Δ_{PSNR} [dB]
balloons	1.30	-0.08	65.42	-4.12	10.96	-0.63	1.81	-0.11
book arrival	9.36	-0.56	12.70	-0.76	0.02	0.00	0.72	-0.04
door flowers	3.30	-0.14	28.89	-1.32	0.36	-0.02	1.24	-0.06
leaving laptop	10.98	-0.51	17.44	-0.80	1.01	-0.05	1.47	-0.07
kendo	5.83	-0.33	24.59	-1.40	5.13	-0.25	5.73	-0.28
lovebird	0.00	0.00	48.28	-2.30	10.56	-0.52	0.64	-0.03
newspaper	0.18	-0.01	70.64	-0.35	8.21	-0.43	1.65	-0.05
pantomime	5.15	-0.31	12.24	-0.72	6.28	-0.37	6.83	-0.41
outdoor	21.21	-1.38	1.73	-0.11	8.96	-0.57	2.34	-0.15
mean	6.36	-0.37	31.32	-1.32	5.72	-0.31	2.49	-0.13

Table 4.21: Average RD performance in Bjontegaard metric for HT , IV, OD and AV vs. the oracle

consequence, another approach could be envisaged: we could update δ_R only when a scene change is detected.

These consideration are validated by introducing the concept of oracle, that is a decisor that a priori knows which the best side information is among temporal, inter-view and fusion. Consequently, it uses directly the best SI without paying the cost of sending the redundant parity bits. The oracle is thus equivalent to the AV algorithm with $n = 1$ and $N = 1$, but with zero overhead. Of course, on one hand, the oracle is not realizable in practice; and on the other, none of the proposed technique can do better than it. Therefore, it is rather useful to understand whether the proposed estimator works well or not. In Tab. 4.20, for each sequence we list the decision provided by the AV algorithm, and we compare to the decisions taken frame-by-frame by the oracle. We underline that in this experiment, we take N equal to the sequence length: in other words, only any decision per sequence is adopted, just by observing the parity bit-rates on the first frame. We observe that for six sequences out of eight, the oracle chooses almost always the same SI, which is the one decided by the AV method. This is a quite good news: for these sequences, the proposed method is able to pick the best SI even though it observes δ_R only for one image over one hundred. For the other sequences, even if the AV does not choose the best SI, this does not affect a lot the performance, since this happens only when all the estimations have comparable qualities. To support this interpretation, we report in Tab. 4.21 the RD

sequence	Δ_R [%]	Δ_{PSNR} [dB]
balloons	-37.03	2.20
book arrival	-2.08	0.13
door flowers	-5.01	0.24
leaving laptop	0.21	-0.06
kendo	-1.59	0.10
lovebird	-16.34	0.80
newspaper	-24.89	1.34
pantomime	4.65	-0.26
outdoor	0.08	0.00
mean	**-9.11**	**0.42**

Table 4.22: RD performance of the adaptive validation vs. the state-of-the-art technique [MMCPP09a].

performance of HT, IV, OD and AV with respect to the oracle. We observe that AV may not be the best method on one given sequence, because of the rate overhead for δ_R estimation. For example, for the *lovebird* sequence, the oracle would chose the HT method all the time. So it does the AV method, but it needs a small bit-rate overhead to take the decision.

However the AV method is the only one that can adapt to the different characteristics of the sequences, and for this reason it has the best average performance, and it never has catastrophic increases with respect to the oracle, as it happens for all the other methods: the temporal interpolation can cost up to 21 % more than the oracle, the inter-view interpolation up to 70 % and even the OD-fusion can have a loss of about 11 %. The AV fusion needs only 6.8 % additional bit-rate with respect to the oracle in the worst case, and about 2.5 % in the average. The consistency of the RD performance is one of the most valuable properties of the proposed technique with respect to the state of the art.

Finally in Tab. 4.22, we give the RD results for the AV method with respect to the reference method [MMCPP09a]. We observe that the adaptive validation achieves a cumulated 9.1 % rate saving with respect to the reference method, that is a further 2.5 % rate reduction with respect to the OD method.

4.4 An application of HOMI for Multiple Description Coding based video streaming architecture for mobile networks

In this section the HOMI algorithm is not applied for the estimation of WZFs in distributed video coding but for reconstructing the missing frames in the context of Multiple Description Coding.

Multiple Description Coding (MDC) is a framework allowing an improved immunity towards losses on error prone channels, thus providing a trade-off between coding efficiency – in terms of compression ratio for a given quality – and robustness.

The basic principle of MDC is that the encoder, given an input signal (image, audio, video, *etc.*), is able to produce a set of *descriptions*, *i.e.*, a set of sub-streams which are mutually refinable, but – unlike scalable coding – also independently decodable. Each description provides a low, yet acceptable, quality; while, as any further description is received, the quality of the reconstruction increases, independently on which description is received [Goy01]. The decoding unit used when all descriptions are available is referred to as *central decoder*, while any decoding unit used when a non-empty subset of the descriptions is available is referred to as *side decoder*.

Several solutions have been proposed for video MDC: separation of the even and odd frames, to be encoded independently with recovery of missing frames by interpolation from the nearest decoded frames [Apo01]; redundant wavelet transform, which also allows scalability [TPPP07]; or periodical insertion of redundant pictures to increase loss resilience and reduce error propagation [RFW+10].

However, while many authors have proposed to design MD codecs from scratch, others have pointed out that an MD codec based only on pre- and post-processing, with the use of legacy coders, reduces significantly the development time, hence the development cost [SGK01, KZK07, WCRB04], even though it may come at the price of sub-optimal performance. Within this context, an MDC scheme for double description coding which is entirely based on pre- and post-processing has been proposed in [GCPP10]. In this section, we propose an improved version of this scheme by using HOMI algorithm for reconstructing the missed frames instead of DISCOVER, and integrate it into a cross-layer framework for video transmission on wireless ad-hoc networks [GC11].

4.4.1 Multiple Description Coding Reference Scheme

In this section, we present the reference MDC scheme, proposed by Greco *et al.*[GCPP10]. The original video sequence is split up into even and odd frames. Then, each sub-sequence is encoded independently with a legacy video coder to produce the two descriptions. Any video coder could be used, but here we shall employ H.264/AVC, which is the most recent standard for video coding. Side decoding is performed using an H.264/AVC decoder on the received description, then reconstructing the missing frames using the temporal interpolation technique proposed in the DISCOVER project. When both decoded descriptions are available, central decoding is performed as a block-wise convex combination of the sub-sequences. The relative weight α of each block of the received frame with respect to the corresponding block in the interpolated frame is computed and quantised at the encoder to minimise the distortion between the block in the original frame and the convex combination; then the sequence of weights α is sent along with the descriptions as side information (see Fig. 4.10).

In order to reduce the bitrate needed to transmit the sequence of weights α, a context-based coding is adopted, with the context E defined as the distortion between received

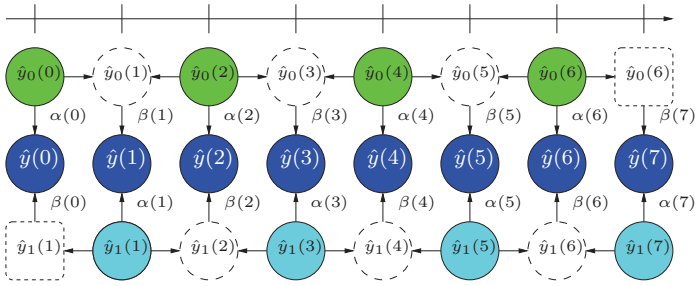


Figure 4.10: Structure of central decoder. Solid circles represent received frames; dashed circles represent interpolated frames. Horizontal arrows represent interpolation. Vertical arrows represent weighted sum. With $\hat{y}_n(k)$ we denote the k-th frame of description n. For each k, $\beta(k) = 1 - \alpha(k)$.

blocks and interpolated blocks. Since the number of possible contexts is very high, in order to avoid *context dilution*, we perform a context quantisation [CAB10]: the distortion values are grouped into quantisation clusters defined by a convex quantisation function $Q(E)$. The thresholds of the quantisation intervals are found by minimising the mutual information $I(\alpha; E|Q(E))$.

4.4.2 Proposed improvements

In this section, we introduce the proposed improvements for the technique described in Sec. 4.4.1. At first, the DISCOVER interpolation method for reconstructing the missing frames is replaced by our algorithm HOMI. Obviously, four frames instead of two are necessary for the trajectory interpolation. Moreover, in the reference technique [GCPP10], the sequence of weights α is computed at the encoder to minimise the distortion $D(\alpha)$ between the block in the original frame and the convex combination, and the resulting rate $R(\alpha)$ for the sequence depends on its conditional entropy given the quantised context. We hereby propose the following improvement: first, we estimate $H(\alpha|Q(E))$ with a preliminary test; then, the conditional entropy is used as an estimation of the coding cost $\widetilde{R}(\alpha)$ and, for each block, we find the optimal value α^*, defined as:

$$\alpha^* \triangleq \arg\min_{\alpha} \left\{ D(\alpha) + \lambda \, \widetilde{R}(\alpha) \right\},$$

where the Lagrangian parameter λ can be chosen equal to the one used by the hybrid coder to encode the sequence.

4.4.3 Transmission over Wireless Ad-hoc Network

In order to validate our technique, we decided to integrate it in a suitable scenario that properly exploits its properties. Thus, in this section, we discuss the streaming of a video content, encoded with the proposed technique, over a mobile ad-hoc network. A mobile ad-hoc network (or MANET) is a dynamic network of mobile devices inter-connected by

wireless links, self-organised in a mesh topology [FJL00]. The set of properties offered by MANETs – such as flexibility, ease of deployment, robustness, *etc.*– makes them suited for real-time transmission in environments without preexisting infrastructure, such as military or disaster-relief applications [Ger05]. In these scenarios, given both the inherent lossy nature of MANETs and the real-time constraints of video streaming, MDC is a very viable solution, since it provides a trade-off between coding efficiency – in terms of rate/distortion – and robustness towards losses.

Transmission protocol

For real-time video streaming, solutions at Application layer perform poorly over ad-hoc networks because they are unaware of the links existing among nodes and their states, thus producing an overlay mismatched with respect tothe topology of the network. On the other hand, solutions at Data-Link layer are unaware of the specific logic and state of the application, and are thus hardly adapted to its needs. Having a shift towards a cross-layer approach been pointed out as needed to overcome these limitations [vdSSS05], we recently proposed [GC11, GCPP11] a novel protocol called ABCD, inherently designed for video streaming in ad-hoc wireless networks, which exploits the broadcast property of the medium in a cross-layered fashion.

The ABCD protocol provides a multi-tree overlay network, with a different tree for each description. Moreover, the protocol is able to quickly adapt to topology changes (movement, nodes' departure or arrival) with a small number or exchanged messages. The main feature of ABCD is the modified 802.11 MAC layer, which implements a reservation mechanism for reliable broadcast, which greatly reduces the collision probability, thus allowing better performance than the standard 802.11 with respect tothe rate *vs.*diffusion area trade-off typical of multi-hop broadcast in wireless networks [EFLBS07]. The node mobility model used for this algorithm is shown in [GPCPP11].

4.4.4 Experimental Results

Comparison over lossless transmission

To encode the test sequences in our proposed scheme and in the reference scheme (*i.e.* the one proposed in [GCPP10]) we used as legacy coder the H.264/AVC reference software JM [Süh08], version 17.0. We selected a set of QPs (22, 25, 28, 31, 33, and 36) in order to compare the RD performance of the two methods.

The rate-distortion performance comparison for side decoding, which is the most affected by our proposed technique, of the video sequence "Stefan" (CIF, 30 fps) is shown in Fig. 4.11.

A more concise comparison over a set of sequences is given in Tab. 4.23 in terms of Bjontegaard metric of the proposed technique with respect tothe reference technique [Bjo01].

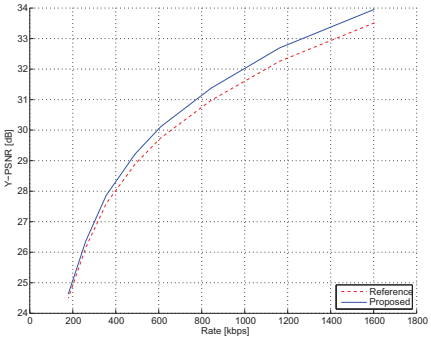


Figure 4.11: RD-comparison between the reference and the proposed technique, for video sequence "Stefan", CIF, 30 fps (Side decoder).

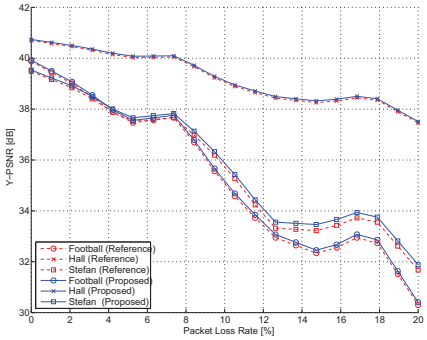


Figure 4.12: Performance versus packet loss rate comparison for fixed bitrate (1.0 Mbps).

Notice that, since sequences are encoded with a fixed QP, the resulting bitrate is higher for sequences with a higher motion content.

We notice that the quality of the central reconstruction for the two techniques is slightly improved. We explain this fact by observing that, with a proper choice of the α coefficients, the central decoder is quite robust to possible errors in the interpolated frames. For the side decoders, on the other hand, we see that in some cases the reference technique can be significantly improved by the high-order motion interpolation. For instance, for the sequences "Foreman" and "Football", there is a gain in terms of Y-PSNR of ~ 0.2 dB on average, corresponding to an average reduction of the bitrate of $\sim 5.4\,\%$. It is worth mentioning that, since the side reconstructions of the reference and the proposed technique differ only on the interpolated frames, the gain on those frames is actually the double; therefore, in the above mentioned cases, the average quality gain for the interpolated frames is ~ 0.4 dB.

Comparison over lossy transmission network

First, a performance comparison with the reference method as a function of the packet loss rate is shown in Fig. 4.12. An encoded video sequence, obtained as a concatenation of the sequences in Tab. 4.23 at the highest quality, is affected by packet losses, modelled as independent and identically distributed Bernoulli random variables with success probability p equal to the loss rate. The same packet loss rate could actually lead to different frame loss rates, since not only the frame or frames in the packet will be lost, but also all the frames predicted upon them (however, it is always only one description that is affected). To reduce the impact of loss propagation, on lossy channels, the stream is usually encoded with a closed GOP, and the size of the GOP must be relatively small ($8 \sim 16$ frames). In order to avoid a bias due to the GOP structure, the results presented here are

Sequence	Bitrate range [kbps]	Y-PSNR Gain (Central)	Bitrate Variation (Central)	Y-PSNR Gain (Side)	Bitrate Variation (Side)
foreman	139 ∼ 349	+0.04 dB	−0.9 %	+0.15 dB	−3.8 %
foreman	195 ∼ 502	+0.04 dB	−0.9 %	+0.19 dB	−5.1 %
foreman	248 ∼ 719	+0.04 dB	−0.9 %	+0.22 dB	−6.4 %
football	271 ∼ 714	+0.03 dB	−0.7 %	+0.16 dB	−5.9 %
football	391 ∼ 993	+0.03 dB	−0.6 %	+0.17 dB	−5.7 %
football	505 ∼ 1355	+0.03 dB	−0.5 %	+0.18 dB	−5.4 %
stefan	356 ∼ 843	+0.03 dB	−0.5 %	+0.12 dB	−3.0 %
stefan	490 ∼ 1161	+0.03 dB	−0.6 %	+0.14 dB	−3.5 %
stefan	609 ∼ 1603	+0.04 dB	−0.6 %	+0.15 dB	−3.8 %
coastguard	218 ∼ 767	+0.03 dB	−0.8 %	+0.11 dB	−3.5 %
coastguard	352 ∼ 1140	+0.03 dB	−0.7 %	+0.13 dB	−3.5 %
coastguard	490 ∼ 1614	+0.03 dB	−0.6 %	+0.14 dB	−3.5 %
city	178 ∼ 484	+0.01 dB	−0.2 %	+0.06 dB	−1.3 %
city	267 ∼ 674	+0.01 dB	−0.2 %	+0.07 dB	−1.8 %
city	346 ∼ 920	+0.01 dB	−0.2 %	+0.09 dB	−2.2 %
hall	120 ∼ 273	+0.03 dB	−0.6 %	+0.05 dB	−1.0 %
hall	161 ∼ 393	+0.03 dB	−0.8 %	+0.05 dB	−1.5 %
hall	200 ∼ 597	+0.03 dB	−0.9 %	+0.05 dB	−2.0 %
akiyo	82 ∼ 180	+0.04 dB	−0.6 %	+0.04 dB	−0.8 %
akiyo	109 ∼ 246	+0.04 dB	−0.6 %	+0.05 dB	−1.0 %
akiyo	135 ∼ 331	+0.04 dB	−0.7 %	+0.06 dB	−1.2 %
flower	437 ∼ 1087	+0.01 dB	−0.1 %	+0.02 dB	−0.3 %
flower	616 ∼ 1471	+0.01 dB	−0.1 %	+0.02 dB	−0.4 %
flower	785 ∼ 1939	+0.02 dB	−0.2 %	+0.02 dB	−0.4 %
mobile	479 ∼ 1160	+0.01 dB	−0.1 %	−0.01 dB	+0.3 %
mobile	660 ∼ 1636	+0.01 dB	−0.1 %	−0.01 dB	+0.2 %
mobile	832 ∼ 2245	+0.02 dB	−0.2 %	−0.01 dB	+0.1 %

Table 4.23: Bjontegaard metric of the proposed technique with respect to the reference over various sequences.

obtained averaging a number of simulations with the same value of packet loss rate. Both techniques, for each image of the decoded sequence, use central decoding whenever both description are received, side decoding if only one description is received, and concealment (freeze of the last decoded image) if both descriptions are lost. As expected, sequences with higher motion content are more affected by packet loss; however, our technique slightly but consistently outperforms the reference method.

Then, a performance comparison with the reference method in a simulated scenario is illustrated in Fig. 4.13 and 4.14. The encoded sequence in this test is transmitted over a mobile ad-hoc network (see Tab. 4.24) using the ABCD protocol described in Sec. 4.4.3. The mobile nodes' interfaces parameters are based on the specifications of the ORiNOCO 11 b/g card [Pro06]. The mobility model detailed in [GPCPP11]. generates a packet loss patter typical of MANETs, where losses may be due to the fact that the network is no longer connected (as a group of nodes moved away from the source) or to collisions, caused by a high local density (as groups of nodes move towards each others). The figures show the Probability Density Function (PDF) and the Cumulative Distribution Function (CDF) of the visual quality of a received frame, estimated with the Parzen window method [Par62]. The PDF of the improved method is more concentrated towards high PSNR values, as it was to be expected after results shown in Tab. 4.23. However, here we give a finer resolution picture of the two methods. The two PDFs are quite close for high values of the

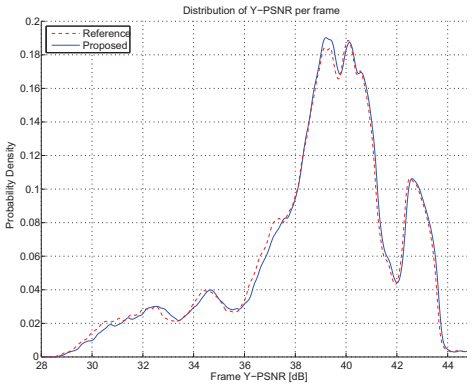


Figure 4.13: Probability density function of the Y-PSNR of the decoded frames(Bitrate 1.0 Mbps, $\sim 25\,\%$ of packets lost).

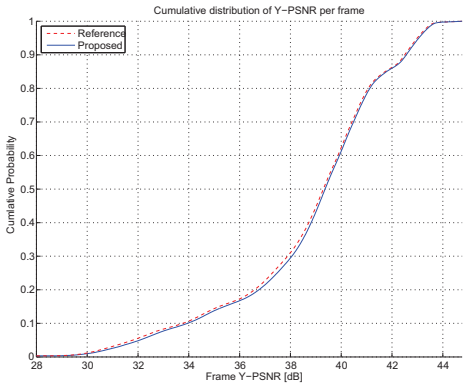


Figure 4.14: Cumulative distribution function of the Y-PSNR of the decoded frames (Bitrate 1.0 Mbps, $\sim 25\,\%$ of packets lost).

Number of nodes	100
Number of groups	10
Playground size	100 m × 100 m
Nominal range	25 m
Average speed	2.0 m/s
Average pause time	2.0 s
Simulation time	600 s

Table 4.24: Simulation parameters. The network interface controllers are based on the specifications of the ORiNOCO 11 b/g card.

Y-PSNR — which correspond to the use of the central decoder. However, for lower values of the Y-PSNR, the proposed method outperforms the reference technique since very low values of Y-PSNR are less probable.

4.5 Publications

The ideas presented in this chapter have been the object of the following publications:

- Giovanni Petrazzuoli, Marco Cagnazzo, Beatrice Pesquet-Popescu, "High order motion interpolation for side information improvement in DVC", *International Conference on Acoustics, Speech and Signal Processing*, Dallas, TX 2010
- Giovanni Petrazzuoli, Marco Cagnazzo, Beatrice Pesquet-Popescu, "Fast and efficient side information generation in Distributed Video Coding by using dense motion representation", *European Signal Processing Conference*, Aalborg, Denmark 2010
- Giovanni Petrazzuoli, Thomas Maugey, Marco Cagnazzo, Beatrice Pesquet-Popescu, "Side Information Refinement for Long Duration GOPs in DVC" *IEEE Workshop on Multimedia Signal Processing*, vol. 1, October 2010. Saint-Malo, France

- Abdal Bassir Abou-Elailah, Giovanni Petrazzuoli, Frederic Dufaux, Joumana Farah, Marco Cagnazzo, Beatrice Pesquet-Popescu, "Side information Improvement in Transform-Domain Distributed Video Coding" *SPIE Application of Digital Image Processing XXXV*, August 2012. San Diego, California

- Giovanni Petrazzuoli, Marco Cagnazzo, Beatrice Pesquet-Popescu, "Novel solutions for Side Information generation and fusion in multiview distributed video coding" [submitted for IEEE Transactions on Multimedia]

- Claudio Greco, Giovanni Petrazzuoli, Marco Cagnazzo, and Beatrice Pesquet-Popescu, "An MDC-based video streaming architecture for mobile networks", *Proceedings of the IEEE Workshop on Multimedia Signal Processing*, Hangzhou, People's Republic of China, October 2011

Chapter 5

Applications of distributed video coding to multiview video plus depth

Contents

In section 2.7, we introduced the Multiview Video plus Depth (MVD) format. In this format each range camera provides a regular color image, accompanied by a 8-bit depth map. Since MVD is envisaged to be implemented for Immersive 3D TV and FTV, we have to take into account that the user can change his viewpoint during the streaming of the data. If H.264/AVC is used in the simulcast configuration for coding the different views, the continuity of the playback during a switching can not be assured. Indeed, if the view switch falls in correspondence of a P/B-frame, probably the reference frame necessary for reconstructing the actual frame is not available at the decoder side. Distributed video coding of the texture and the depth video can solve this problem. In fact, WZFs can

be decoded and reconstructed independently of the generated side information, which in turns, depends on the available frames and the estimation method used by the decoder. Therefore, in this chapter we deal with the problem of using DVC for improving interactive multiview streaming services. In particular, we propose

- a new technique for coding the depth map
- new solutions for interactivity requirements during view switching.

We conclude the chapter by outlining a novel DVC architecture for MVD video, which exploits the geometrical information of depths to reduce or eliminate the use of the feedback channel.

5.1 Interactivity for view switching

As it was shown in the Chapter 2, MVC and MVD have a huge redundancy: not only in time, as ordinary mono-view video, but also among views (inter-view correlation) and for MVD between depth and texture. All these kinds of redundancies have to be exploited in order to reduce the storage space on the server and the bandwidth used for transmission [FG07]. These two requirements are equivalent in the context of non-interactive scenarios (like TV broadcasting), when all the video stored on the server will be sent to the user. For example, all the views are sent when MVC is used on an autostereoscopic display. On the contrary, the interactive multiview video streaming (IMVS) [COC11] is a paradigm that enables the client to select interactively the view that he/she wants to display. Given this constraint, the server will send only the data needed to display the views according to the switch pattern decided by the user. However, the video is first encoded and stored in a server and afterwards the views demanded by the users are sent to them, as shown in Fig. 5.1. We would like to minimize both the storage space demanded by the compressed video and the bandwidth needed to interactively send the requested view to the user. These requirements are conflicting in the case of IMVS [COC11], which makes the problem challenging.

5.1.1 State of the art on Interactive 3D-TV

Let us consider a client switching from the view v_1 to v_2. The images of v_2 cannot be encoded using previous images from the same view, since the decoder will not have them. If the two views are coded with H.264/AVC and the view switch falls in correspondence of a P-frames, as in Fig. 5.2, the continuity of the playback can not be assured because the reference frames are not available at the decoder side and an additional delay is necessary to request them to the encoder.

Therefore, two contrasting approaches emerge: on the one hand, we could reduce the bandwidth requirement if for any view, any image is coded V times, using any other views

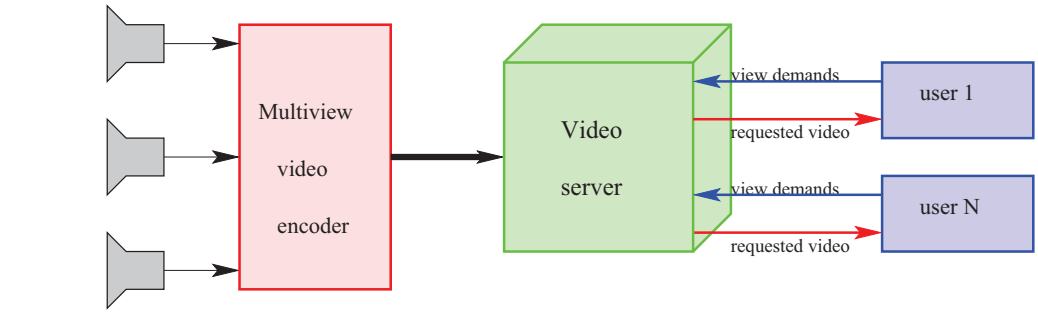


Figure 5.1: In the IMVS paradigm the video is first pre-encoded, stored in a video server and afterwards sent to the users, according to their requests

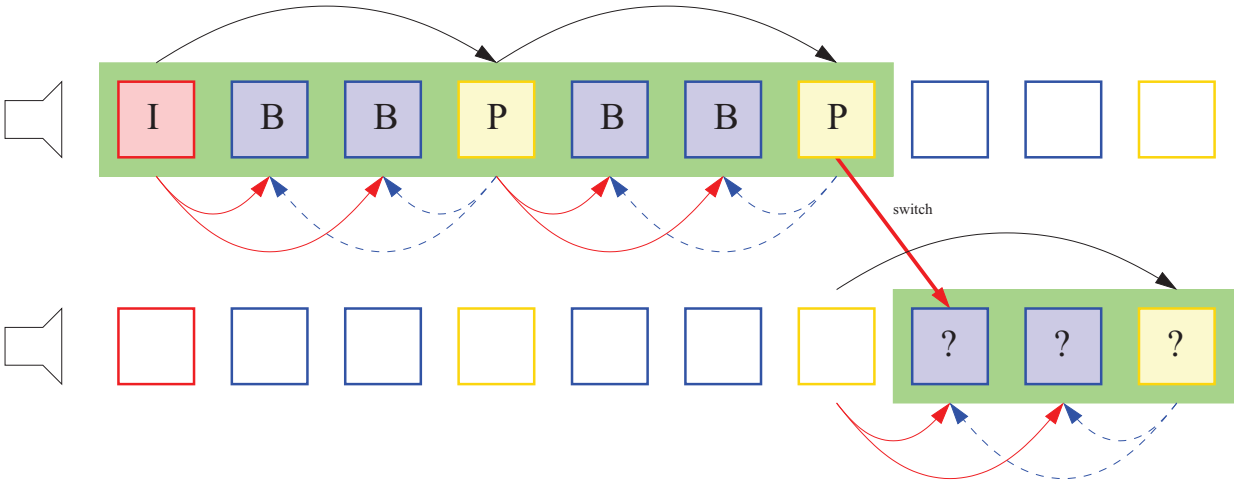


Figure 5.2: In H.264/AVC, frames of the switched view cannot be decoded in real time because some references frames can not be available at the decoder side. The frames with a green background are the ones that the viewer wants to visualize

as reference, where V is the number of the view. In this case the storage requirement increases exponentially with V (since at the encoding time we do not know which views the user will choose at any time), but the required bandwidth is minimized, since we can send, for the current image, only the residual with respect to the images we have already sent. This approach is called **Redundant P-frames**. In Fig. 5.3(b) a three camera system structure with redundant P-frames is shown: for each P-frames three residual errors according to the reference view are needed. On the other hand, we could encode each image only once but as an Intra frame, thus reducing the storage space. However in this case the bandwidth requirements are more demanding, since I-frames need much more bits to be transmitted than P's do for the same quality. This approach is called **All I-frames** (see Fig.5.3(a)) [COC11].

An alternative approach exploits principles coming from distributed video coding (DVC). Each view is coded (independently from others) with a DVC technique as in

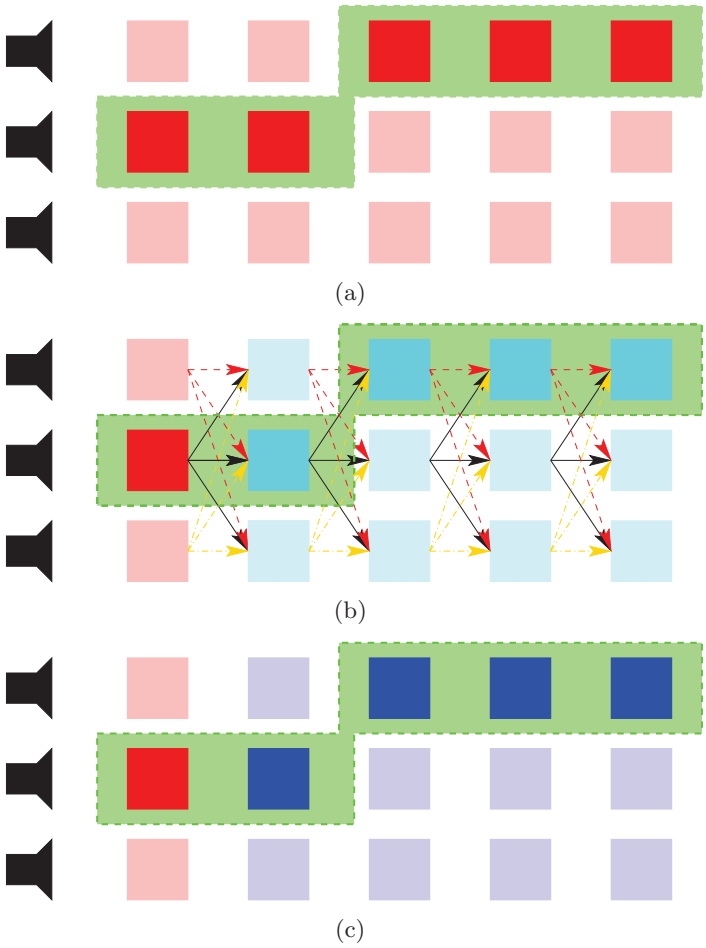


Figure 5.3: Three different solutions to assure continuity of playback during the view switching: all INTRA Frames (a), Redundant P-frames (b) and DVC (c). The KFs and the I-frames are in red. The P-frames are in cyan. The WZFs are in blue. The frames with a green background are displayed by the viewer

Fig. 5.3(c), such as DISCOVER. This approach can be effectively used in IMVS. When a user switches to a novel view, the next image to be displayed can either be a KF or a WZF. In the first case, it is decoded as usual; in the second one, we just have to adapt the mechanism of SI production, but we do not change the encoded representation of the frame stored on the server, *i.e.* the parity bits of the WZFs. This is interesting, since on one hand we improve the storage cost (only one version of any view is stored on the server), and on the other hand we do not send I-frames but only WZFs, which in theory could require as few bits as a P-frame, while in practice have an intermediate coding rate between I's ans P's [GARRM05]. Of course, any future novel and better DVC scheme will have an immediate impact on this kind of scheme for IMVS. In Fig. 5.4, a 2D plot that represents the trade-off between storage space and bandwidth necessary for sending the data is shown.

Cheung et *al.* [COC11] propose to insert the Wyner-Ziv Frames, called for this ap-

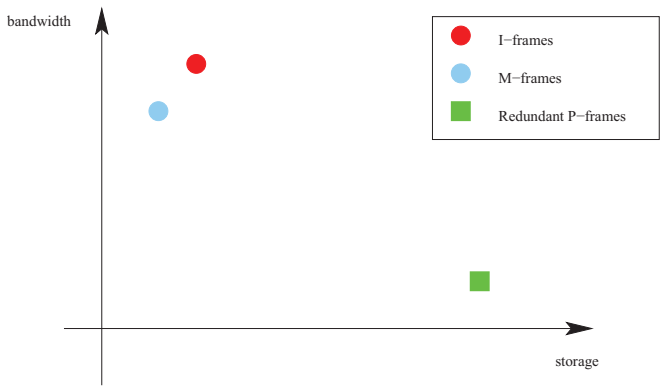


Figure 5.4: A 2D plot of storage space vs. bandwidth necessary for All I-frames, Redundant P-frames and M-frames for IMVS

plication as **M-Frames (Merge Frames)**, in the video stream. In this case, the side information is the frame available at the decoder at the instant previous to the view switching. They find an optimal combination of I-frames, Redundant P-frames and M-frames in the context of RD performance, but they are interested in the case of multiview video coding *without* the depth information.

To the best of our knowledge, only a few papers deal with IMVS in MVD with the constraint of ensuring that the video playback is not interrupted during switching. In MVD the depth maps are usually coded independently of the texture images. Kurutepe et *al.* [KCT06] propose to send to the user also lower bit-rate versions of a set of adjacent views (texture plus depth) in addition to the currently required view, in order to alleviate adverse effects of the unavoidable delay between the time a client requests a new stream and the time it becomes available. Yang et *al.* [YYJP07] propose to estimate the disparity map between different views. Differently from DIBR, disparity based algorithm can be applied also when camera parameters are not available. With their algorithm, they can simply generate virtual views in real time.

5.1.2 Proposed methods for interactive MVD

For the sake of simplicity, we consider a specific case, where we have two range cameras, there is a KF out of N images in time domain (i.e. a GOP size of N), and the KFs on the two views are shifted by half a GOP. For each view the texture and the depth maps are coded independently from the other view, e.g. using DISCOVER. This will assure an efficient use of server storage, since DISCOVER performances are better than all-Intra coding. However we note that, since the encoding process is performed *off-line*, at that time we do not know which views will be used by the decoder to estimate the WZFs, so we do not know how many bits it will demand. Therefore we have to store all the parity bits, which can be more than the bits that will actually be demanded at the decoding time.

Name	Description
m	switching time: user wants view 1 up to $m-1$ and then view 2
k	time of the KF of the GOP affected by the switch on view 2. $k \leq m$
N	GOP size; for simplicity we only consider the case $N = 4$
I_n	decoded frame for the first view at instant n
J_n	an estimated frame of the target view, taken at time n and used as reference for motion interpolation for the remaining WZFs of the GOP
$\widetilde{I}_n$	estimation of the second view at time n obtained by depth-aided DIBR [DPP10] on I_n
$\widehat{I}_n$	$\widetilde{I}_n$ corrected by parity bits

Table 5.1: Notation for interactive MVD methods

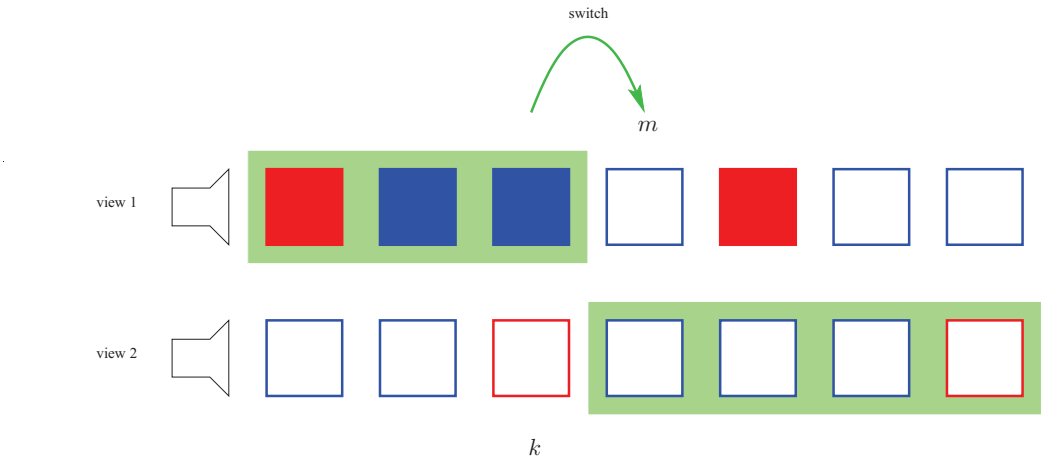


Figure 5.5: An example of view switching

However this is unavoidable in the Wyner-Ziv Frame approach. Still, this represents less than storing all possible P residuals in a classical coding framework.

We have to choose what information is sent to the decoder when the user makes a switch between two views, and how this is used. We propose six different techniques, exploiting DIBR and DVC. In order to describe the proposed methods, we introduce the following notations. We call m the instant of the switch: that is the user wants view 1 up to the time $m-1$, and then he/she wants the other view as shown in Fig. 5.5. We call k the instant when the GOP of frame m begins in view 2, in other words the previous KF for the target view happens to be in k. Because of the periodical GOP structure, we have to consider only the cases $m = k, k+1, k+2, \ldots k+N-1$, namely the cases when the switch corresponds to a KF or to one of the $N-1$ WZF of the GOP. However, since the GOP structures of the views are shifted by half a GOP, the frames globally concerned by the switch are those from $k - N/2$ (beginning of the current GOP on view 1) to $k+N-1$ (end of the current GOP on view 2).

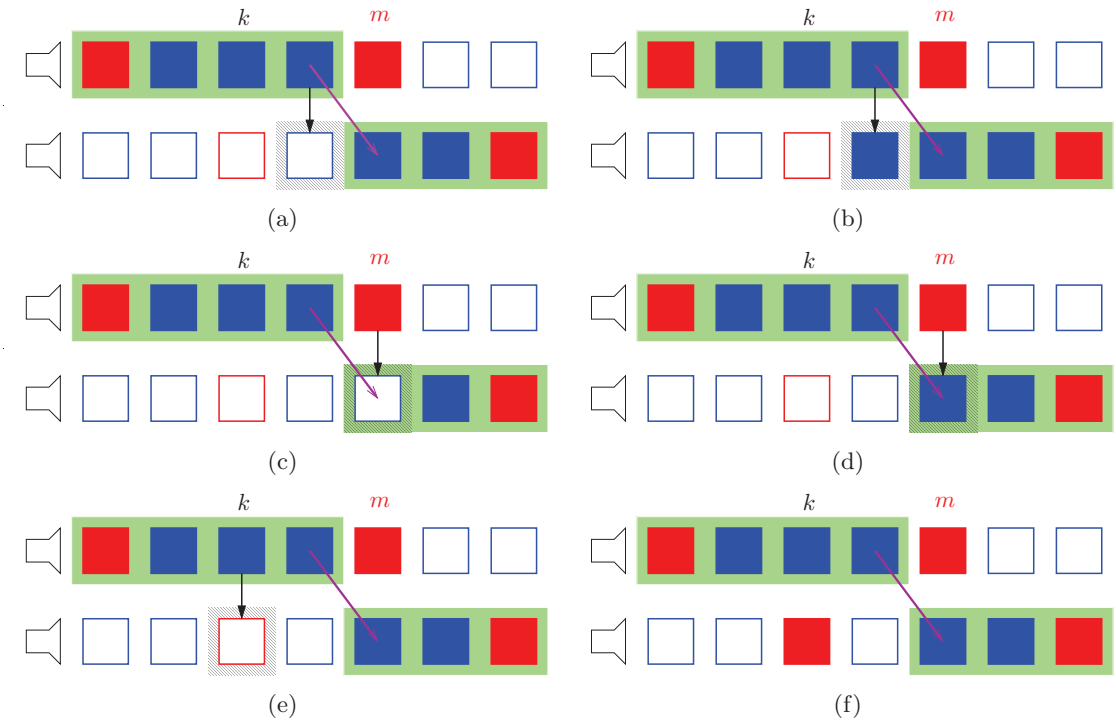


Figure 5.6: Solutions for $m = k+2$: (a) A-DIBR; (b) A-DIBRc; (c) I-DIBR; (d) I-DIBRc; (e) GOPp; (f) noDIBR. The KFs are in red and the WZFs are in blue. The frames with green background are displayed. The filled frames are sent by the video server to the user. The black arrows shows that DIBR is applied.

Moreover, we call J_n the reconstructed image for the second view *used for creating the side information*. This image will not necessary be displayed, but will be used, together with the KF $k+N$ that will always be sent ($m < k+N$ by definition) to produce the estimation for the WZF of the current GOP in the target view. We call this image the *reference image*. Finally we denote by $\widetilde{I}_n$ the estimation of the target view at time n obtained by only using depth-aided DIBR [DPP10] on the first view, and with $\widehat{I}_n$ an improved version of this image, obtained by using the parity bits of the second view image to correct $\widetilde{I}_n$. The introduced notation is summarized in Tab. 5.1 for ease of reading.

Now we can conceive several methods for IMVS using DIBR. They are shown in Fig. 5.6 for $m = k+2$, but this can be easily generalized to any m. A first one, that we call **Advanced DIBR (A-DIBR)** consists in computing J_{m-1} at the decoder by using DIBR on the last received image from the first view. According to our notation, the estimation of view 2 at the instant $(m-1)$ is obtained as DIBR over I_{m-1}: we write $J_{m-1} = \widetilde{I}_m$. Next images in the GOP will be interpolated using J_{m-1} and the next KF on the target view, and this side information will be corrected with the parity bits of the target view. This method is shown in Fig. 5.6(a). A variation of this method, that we can use only when $m \neq k+1$ consists in using the parity bits in order to improve the reference image: $J_{m-1} = \widehat{I}_{m-1}$. This second method is not necessarily better than the first one, since

method	time for reference image	computation of the reference image
A-DIBR	$m-1$	$\widetilde{I}_{m-1}$
A-DIBRc	$m-1$	$\widehat{I}_{m-1}$
I-DIBR	m	$\widetilde{I}_m$
I-DIBRc	m	$\widehat{I}_m$
GOPp	k	$\widetilde{I}_k$
noDIBR	k	KF of target view

Table 5.2: Summary of methods

the higher reference quality is traded-off with a higher coding rate (the parity bits were not sent in the previous case). We call this method **Advanced DIBR + correction (A-DIBRc)**, and it is depicted in Fig. 5.6(b).

Another couple of methods is obtained by performing DIBR at switch time instead of the previous instant. This means that we send to the decoder the parity bits (or the key frame) of the first view at time m, so that it can reconstruct the latter and use it to perform DIBR, without (**Immediate DIBR, I-DIBR**) or with sending the parity bits for view 2 at time m (**Immediate DIBR + correction, I-DIBRc**). Those methods are illustrated in Fig. 5.6(c) and (d), respectively.

We can also think about preserving the GOP decoding structure in the second view. If we compute the previous KF on the GOP of the target view by DIBR, then we can use it to perform image interpolation in the GOP. In this case we just need to compute J_k as $\widetilde{I}_k$. We call this method **GOP preserving with DIBR (GOPp)**, shown in Fig. 5.6(e).

The last method does not use DIBR. This method consists in directly sending the key frame of the current GOP for the target view. With respect to the GOPp method, it demands more rate but provides a better representation of the reference frame. This last method is called **GOP preserving without DIBR (noDIBR)** and it is shown in Fig. 5.6(f)

The proposed methods are summarized in Tab. 5.2, by describing the time chosen to compute the reference image and the technique used to perform this operation. Note that in general not all the methods can be applied for each m: for example A-DIBRc (resp. I-DIBRc) are applicable only if in $m-1$ (resp. m) there is a WZF for the second view.

Now we compute the rate and the distortion provided by the different methods for the frames concerned by the switch, namely from $k - N/2$ to $k + N - 1$. We note with $R_n^{(i)}$ the rate for the frame at instant n of the i-th view, with R_{ST} the rate needed for storage,

and with R_{BW} the bandwidth needed for sending the requested video. We can write:

$$R_{\mathrm{ST}} = \sum_{n=k-N/2}^{k+N-1} R_n^{(1)} + \sum_{n=k-N/2}^{k+N-1} R_n^{(2)} \tag{5.1}$$

$$R_{\mathrm{BW}} = \sum_{n=k-N/2}^{m-1} R_n^{(1)} + \sum_{n=m+1}^{k+N-1} R_n^{(2)} + R_{k+N/2}^{(1)} + R_{\mathrm{SW}} \tag{5.2}$$

Eq. (5.1) accounts for the fact that all encoded frames have to be stored (independently of the IMVS method). However, only some of them have to be sent (see Eq. 5.2), namely frames from $k-2$ to $m-1$ for the first view and from $m+1$ to the end of the GOP for the second. Moreover, we always send the next KF of the first view, since it is needed to obtain $I_{k-2}, \ldots, I_{m-1}$. Then, according to the chosen technique, some other frame should be sent (this is represented by R_{SW}). In particular, we always send the encoded frame of the target view at time m (excepted for I-DIBR, where it is obtained as $\widetilde{I}_m$). Moreover we send the parity bits for the frame obtained by DIBR in methods A-DIBRc and I-DIBRc (if the corresponding frame is a WZF), the KF k of second view for the noDIBR method and possibly, for GOPp, frame $m-1$ of the second view [1].

As far as the distortion is concerned, we consider only frames that are visualized by the user. Therefore, the distortion is computed as $D = \sum_{n=k-N/2}^{m-1} D_n^{(1)} + \sum_{n=m}^{k+N-1} D_n^{(2)}$.

5.1.3 Experimental results

In our tests we use the first 100 frames of the MVD sequences *ballet*, *breakdancers* and *mobile*. The first two sequences are at resolution of 1024×768 pixel, while the third one is at 720×540 resolution.

The texture sequence and the depth maps are independently encoded for both cameras with the DISCOVER algorithm. In order to compare the algorithms, five switches from one view to the other are performed for each sequence. The results are presented in Tab. 5.3,5.4, 5.5 where we report the Bjontegaard metric [Bjo01] of the first five methods with respect to the sixth one (noDIBR).

We remark that, according to the position of the switching point within the GOP, the performance of the methods change, and some of them become equivalent, while some others are not available. For example, for $m = k$ the best solution is noDIBR, which was expected: when the switching point coincides with a KF, the best is to send it directly. Moreover in this case, the A-DIBR/A-DIBRc methods would not be possible, since the frame $m-1$ belongs to another GOP. For $m = k+1$ the best solution is A-DIBR/GOPp[2]. For $m = k+2$ the best solution is I-DIBRc: in this case A-DIBR is less effective because the reference frame is farther apart. For $m = k+3$ the most effective method is A-DIBRc because it demands the smallest number of non-displayed frames. Note that, when parity bits are available, it is always better to send them since the quality of the frame J_n used

[1] Only if it is needed in hierarchical temporal interpolation, e.g. for $m = k+3$ and $N = 4$.

[2] The two methods coincide since the frame in $m-1$ is the previous KF.

ballet				
	texture		depth	
method	Δ_R [%]	Δ_{PSNR} [dB]	Δ_R [%]	Δ_{PSNR} [dB]
$m = k$				
I-DIBR/GOPp	13.40	-0.80	13.76	-0.93
$m = k + 1$				
A-DIBR/GOPp	**4.81**	**-0.27**	**1.76**	**-0.13**
I-DIBR	10.07	-0.59	9.69	-0.62
I-DIBRc	3.24	-0.18	1.73	-0.11
$m = k + 2$				
A-DIBR	0.54	-0.02	-1.54	0.08
A-DIBRc	-0.42	0.02	-2.11	0.13
I-DIBR	7.34	-0.43	4.58	-0.29
I-DIBRc	**-2.22**	**0.12**	**-3.57**	**0.21**
GOPp	-0.77	0.07	-1.67	0.09
$m = k + 3$				
A-DIBR	-5.53	0.33	-6.67	0.41
A-DIBRc	**-5.82**	**0.34**	**-6.67**	**0.41**
I-DIBR	18.29	-1.04	17.21	-1.05
I-DIBRc	8.93	-0.50	8.65	-0.52
GOPp	-1.53	0.09	-3.05	0.18

Table 5.3: Comparison between different methods in terms of rate-distortion performance with the Bjontegaard metric for *ballet* sequence [Bjo01]

for creating the side information for the rest of the GOP is improved. Therefore, the next estimated frames are improved as well less parity bits are needed to correct them. So, I-DIBRc and A-DIBRc should be used instead of I-DIBR and A-DIBR, respectively, whenever the parity bits are available. In conclusion, we remark that no technique is always the best. We highlight this by comparing the 6 proposed methods (see Tab. 5.6) with the optimal combination which uses the best method for each value of m, that is I-DIBR/GOPp for $m = k$, A-DIBR/GOPp for $m = k + 1$, I-DIBRc for $m = k + 2$ and A-DIBRc for $m = k+3$, and averaging over all possible values of m. We see that if we want to use only one method, it is better to use A-DIBRc. If we want to gain a further almost 1% in bit-rate we have to use the optimal combination. In Table 5.7, we show the bit rate percentage for coding the depth maps w.r.t. the total bit rate necessary for encoding texture plus depth. These values are nearly constant w.r.t. QP. In Fig. 5.7, the rate distortion curves for the sequence *mobile* are shown for all the switching configurations. We remark that the RD performance are always improved w.r.t. All I-frames solution, where all the frames that are sent are INTRA Frames.

breakdancers				
	texture		depth	
method	Δ_R [%]	Δ_{PSNR} [dB]	Δ_R [%]	Δ_{PSNR} [dB]
$m = k$				
I-DIBR/GOPp	1.52	-0.08	0.48	-0.01
$m = k + 1$				
A-DIBR/GOPp	**-6.21**	**0.33**	**-5.82**	**0.36**
I-DIBR	0.89	-0.04	-0.30	0.03
I-DIBRc	-4.42	0.22	-4.38	0.26
$m = k + 2$				
A-DIBR	-5.76	0.29	-4.64	0.28
A-DIBRc	-5.77	0.29	-5.01	0.29
I-DIBR	-3.55	0.18	-2.88	0.19
I-DIBRc	**-11.01**	**0.56**	**-8.04**	**0.50**
GOPp	-10.98	0.58	-4.94	0.33
$m = k + 3$				
A-DIBR	-9.26	0.48	-8.42	0.50
A-DIBRc	**-13.36**	**0.68**	**-8.67**	**0.55**
I-DIBR	12.38	-0.59	17.60	-0.99
I-DIBRc	5.18	-0.22	12.80	-0.71
GOPp	-10.48	0.54	-5.18	0.34

Table 5.4: Comparison between different methods in terms of rate-distortion performance with the Bjontegaard metric [Bjo01] for *breakdancers* sequence

mobile				
	texture		depth	
method	Δ_R [%]	Δ_{PSNR} [dB]	Δ_R [%]	Δ_{PSNR} [dB]
$m = k$				
I-DIBR/GOPp	1.65	-0.04	1.10	-0.01
$m = k + 1$				
A-DIBR/GOPp	**-9.56**	**1.00**	**-5.82**	**0.36**
I-DIBR	-0.87	0.20	-0.50	0.10
I-DIBRc	-7.64	0.85	-5.60	0.40
$m = k + 2$				
A-DIBR	-4.21	0.26	-3.50	0.19
A-DIBRc	-5.60	0.39	-5.01	0.35
I-DIBR	-3.86	0.23	-2.11	0.25
I-DIBRc	**-8.16**	**0.43**	**-7.50**	**0.40**
GOPp	-6.62	0.32	-5.00	0.31
$m = k + 3$				
A-DIBR	-7.80	0.93	-7.02	0.80
A-DIBRc	**-12.64**	**1.68**	**-10.47**	**1.22**
I-DIBR	9.47	-0.83	10.12	-0.70
I-DIBRc	7.17	-0.47	8.44	-0.52
GOPp	-5.54	0.88	-5.11	0.67

Table 5.5: Comparison between different methods in terms of rate-distortion performance with the Bjontegaard metric [Bjo01] for *mobile* sequence

method	Δ_R [%]	Δ_{PSNR} [dB]
A-DIBRc	1.01	-0.10
I-DIBRc	6.10	-0.54
GOPp	3.08	-0.20
no DIBR	6.43	-0.34

Table 5.6: Rate-distortion performance for texture video (averaged over the three sequences) with the Bjontegaard metric [Bjo01] wrt the optimal combination.

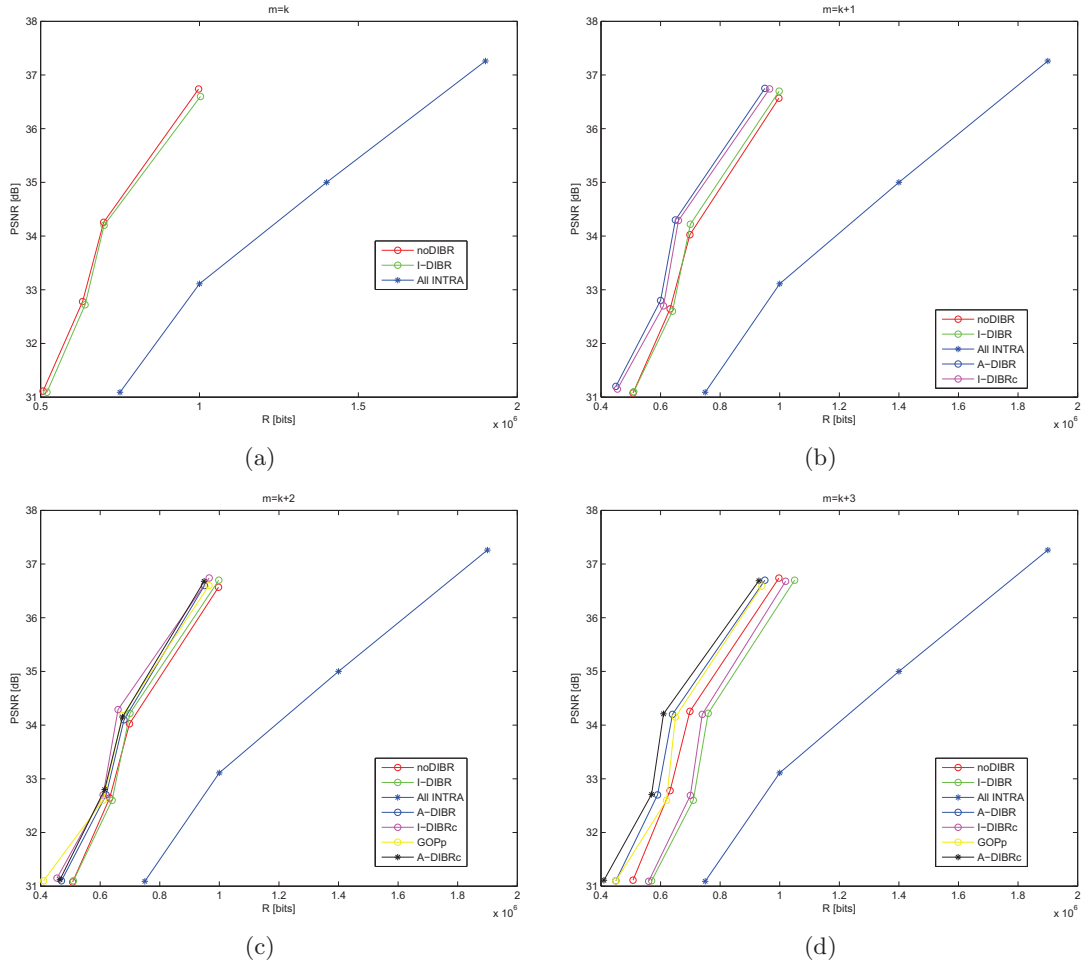


Figure 5.7: RD performance for different methods for *mobile* sequence for each configuration of view switch

	QP			
sequence	31	34	37	40
ballet	40.5	41.1	40.8	40.4
breakdancers	29.3	29.9	29.5	29.6
mobile	4.40	4.90	5.53	6.39

Table 5.7: Percentage of the bit rate spent for encoding the depth maps w.r.t. total bit rate for encoding texture plus depth.

5.2 Depth-map estimation for DVC

In order to improve the rate distortion performance of the DVC codec in the context of MVD, in this section we propose a technique for SI generation for depth maps.

Usually, depth maps are encoded independently of the texture images and independently of the other views by a simulcast coding structure, or they can be encoded jointly among the different views by a multiview-coding structure [MSMW07b]. In those schemes the depth video is coded by a standard coding technique (e.g. H.264/AVC or H.264/MVC). Here, we propose to apply DVC to MVD. Then, the texture video and the depth map video are independently coded using the Stanford DVC Architecture. We consider a 1/2-symmetric scheme for both texture and depth sequences for GOP size equal to 2. The resulting structure is shown in Fig.5.8 The texture frame at the instant m is indicated as I_m. The depth map at instant m is indicated as I_m^D. The DISCOVER or HOMI MCTI algorithms can be used for the estimation of the WZ depth map at instant t using texture key frames I_ℓ and/or depth key frames I_ℓ^D (with $\ell \in \{t-3, t-1, t+1, t+3\}$).

A first trivial technique for estimating I_t^D, that we call ZD (DISCOVER over Depth Data), consists in applying DISCOVER motion esimation on the depth map I_{t-1}^D and I_{t+1}^D. The obtained MVFs $\mathbf{u}$ and $\mathbf{w}$ are applied, respectively, to the two depth maps I_{t-1}^D and I_{t-1}^D for obtaining the motion compensated depth $\widehat{I}_t^D$. This will be the reference method. The block diagram is depicted in Fig. 5.9.

In another techniques, referred to as ZD-ZH (DISCOVER and HOMI over Depth Data), the vectors obtained using DISCOVER on I_{t-1}^D and I_{t+1}^D are used to initialize HOMI, which in turn is run over Z data, then along I_{t-3}^D and I_{t+3}^D. The obtained vectors $\widehat{\mathbf{u}}$ and $\widehat{\mathbf{w}}$ are used on I_{t-1}^D and I_{t+1}^D for producing the estimation of I_t^D, as shown in Fig. 5.10. This technique has the potential to improve the results of DISCOVER for depth map coding. However, we want to better exploit the correlation between texture images and depth maps, and therefore to introduce other new coding methods.

We observe that motion in depth images is very similar to motion in the texture sequence. So we could use DISCOVER vectors computed on texture in order to perform the motion compensation and the image interpolation of depth maps. We call this method TD (DISCOVER over Texture): DISCOVER is applied on the texture frames I_{t-1} and I_{t+1}. The obtained MVFs $\mathbf{u}$ and $\mathbf{w}$ are applied, respectively, to the two depth maps I_{t-1}^D and I_{t-1}^D for obtaining the motion compensated frame $\widehat{I}_t^D$. This method is depicted in Fig. 5.11.

This straightforward method can be improved if we refine TD vectors by using HOMI over texture data. We call the resulting technique TD-TH, shown in Fig. 5.12: DISCOVER is applied on the texture frames I_{t-1} and I_{t+1}. The obtained MVFs $\mathbf{u}$ and $\mathbf{w}$ are the initialization for HOMI algorithm performed on I_{t-3}, I_{t-1}, I_{t+1} and I_{t+3}. The obtained MVFs $\widehat{\mathbf{u}}$ and $\widehat{\mathbf{w}}$ are applied, respectively, to the two depth maps I_{t-1}^D and I_{t-1}^D to produce the motion compensated frame $\widehat{I}_t^D$.

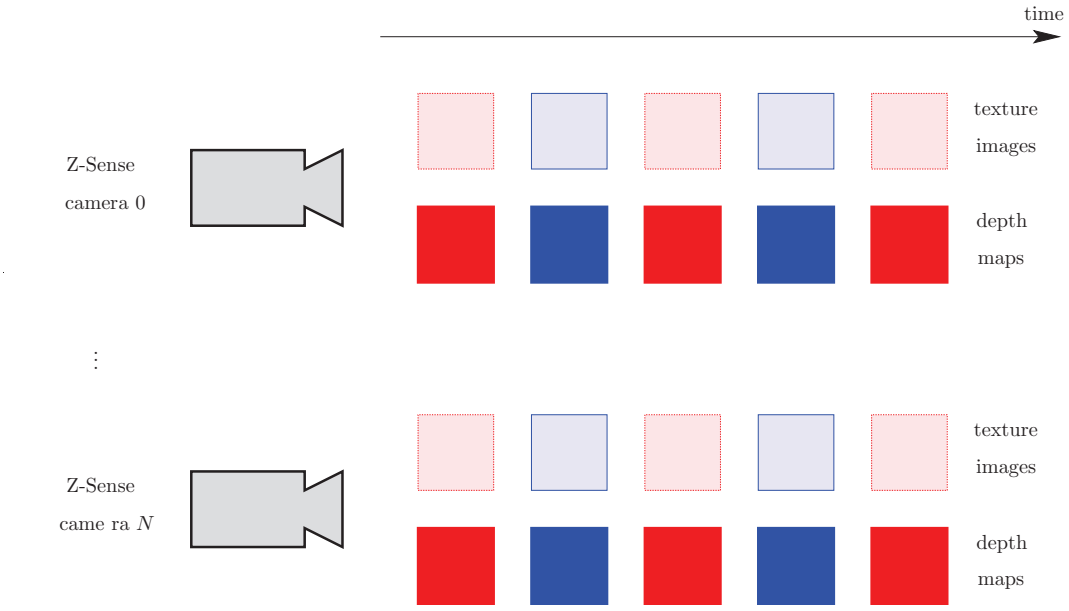


Figure 5.8: The frame repartition in time-view domain proposed in our method (GOP size equal to 2). The KFs are in red, while the WZs are in blue. The texture images and the depth maps are independently coded.

Label	Initialization	Refinement
ZD	*DISCOVER on depth maps*	—
ZD-ZH	DISCOVER on depth maps	HOMI on depth maps
TD	DISCOVER on texture	—
TD-TH	DISCOVER on texture	HOMI on texture
TD-ZH	DISCOVER on texture	HOMI on depth maps

Table 5.8: Labels for reference (in italic) and proposed methods

Finally we consider a last technique, which uses DISCOVER over texture frames I_{t-1} and I_{t+1} and the MVFs $\mathbf{u}$ and $\mathbf{w}$ are obtained. Now, these vectors are refined by HOMI algorithm performed on the depth maps I^D_{t-3}, I^D_{t-1}, I^D_{t+1} and I^D_{t+3}, and not on the texture frames. This method is called TD-ZH and the block diagram is depicted in Fig. 5.13. The rationale behind it is that texture data are much richer in information than depth map, and it could provide an initialization that is closer to real object movement. However we point out that texture movement does not always correspond perfectly to depth map movement, and *vice versa*: for example, an object moving over a static background which is at the same distance from the camera, would not result in depth map motion. The proposed methods are resumed in Tab. 5.8.

5.2.1 Experimental results

We have implemented the reference method and all the proposed techniques shown in Tab. 5.8, and we have run several tests to validate and compare them to the reference. In

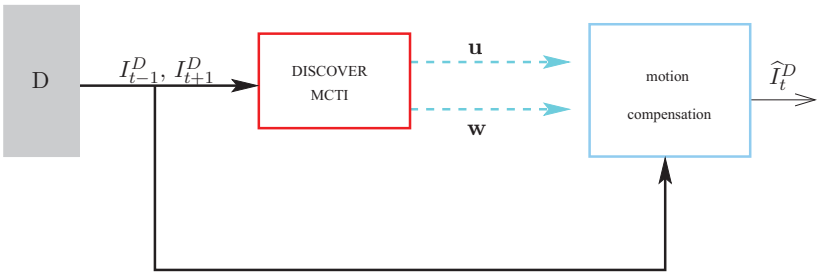


Figure 5.9: ZD method: the frames I^D_{t-1}, I^D_{t+1} are used for the estimation of the MVFs $\mathbf{u}$ and $\mathbf{w}$. These two MVFs will be used for the motion compensation of I^D_{t-1}, I^D_{t+1} and the SI $\widehat{I}^D_t$ is produced.

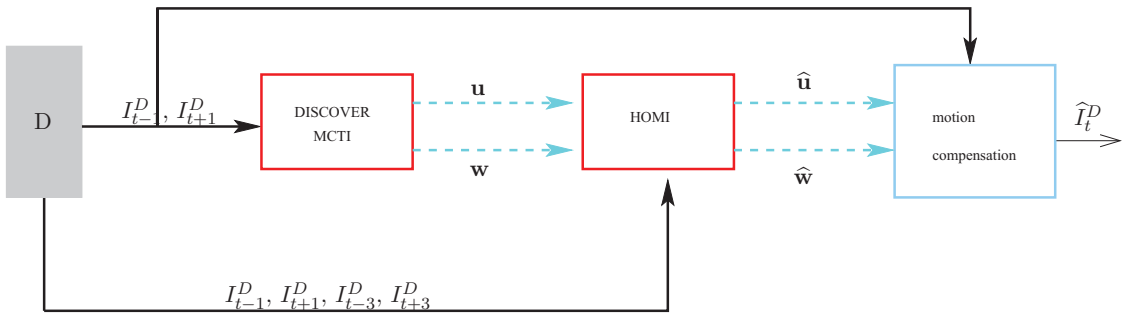


Figure 5.10: ZD-ZH method: the frames I^D_{t-1}, I^D_{t+1}, I^D_{t-3}, I^D_{t+3} are used for the HOMI algorithm and the MVFs $\widehat{\mathbf{u}}$ and $\widehat{\mathbf{w}}$ are produced. These two MVFs along with I^D_{t-1} and I^D_{t+1} are used for the construction of $\widehat{I}^D_t$.

a first stage, we use as evaluation metric the PNSR of the SI with respect to the original WZF. More precisely, for each one of the new methods, we compute the PSNR difference with respect to DISCOVER (ZD). This quantity is called Δ_{PSNR}.

In a second stage, we compute end-to-end performance (i.e. Bjontegaard metric for rate reduction and PSNR improvement) of the proposed techniques when inserted into a complete DVC coder.

The experiments are conducted as follows. We use the texture and depth map sequences *ballet* and *breakdancer* at a resolution of 384×512 pixels. We encode the texture and depth KFs using the INTRA mode of H.264/AVC, at four quantization step values, namely 31, 34, 37 and 40. The Δ_{PSNR} values are computed as average along the sequences.

The optimal values of λ for Eq. (4.1) are obtained by maximizing the average PSNR over the first 20 frames of the two sequences and at all the QPs. We observe that for the TD method λ is not needed, while for TD-TH we can use the values reported in the previous chapter. The results are shown in Tab. 5.9. We observe that, when only texture data is used, we need a stronger regularization, while if texture is used only as initialization,

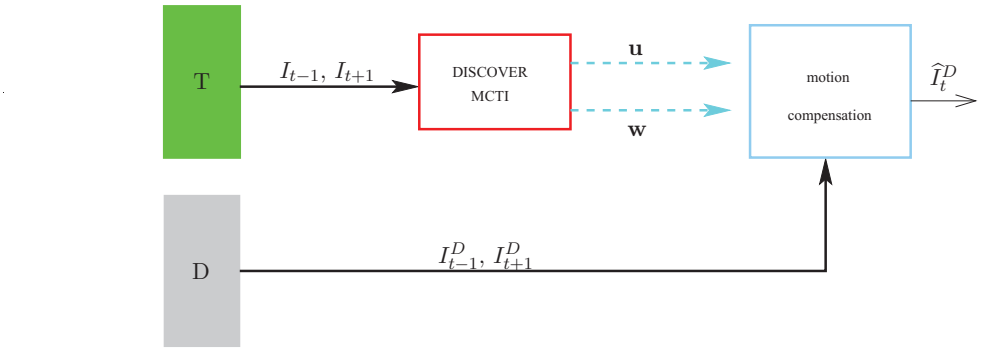


Figure 5.11: TD method: I_{t-1}, I_{t+1} are used for the trajectory interpolation of the DISCOVER MCTI algorithm. The obtained MVF $\mathbf{u}$ and $\mathbf{w}$ are applied to I_{t-1}^D and I_{t+1}^D for producing the SI $\widehat{I}_t^D$

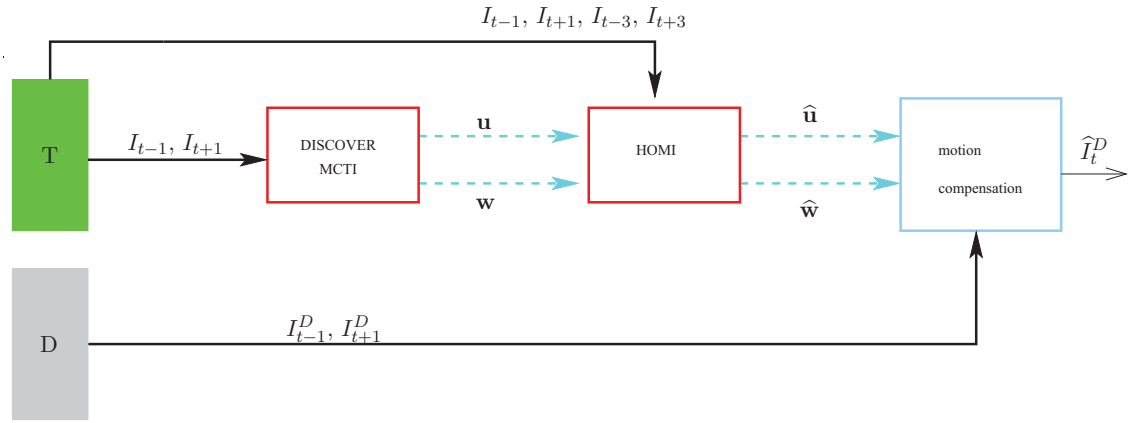


Figure 5.12: TD-TH method: I_{t-1}, I_{t+1}, I_{t-3}, I_{t+3} are used for the trajectory interpolation of HOMI algorithm. The obtained MVFs $\widehat{\mathbf{u}}$ and $\widehat{\mathbf{w}}$ are applied to I_{t-1}^D and I_{t+1}^D for producing the SI $\widehat{I}_t^D$

λ must be smaller to enable larger correction (since vectors are not initialized on depth maps). However, if texture is not used at all, an intermediate regularization strength is needed.

By using these parameters, we compare the SI quality of all proposed methods with respect to the ZD method. The complete results are given in Tab. 5.10 and 5.11. We observe that almost all methods allow remarkable gains in SI quality, up to 0.6 dB. We notice that the TD-ZH method gives the best results. In fact, the initialization is better if computed on the texture images, because the depth data are too homogeneous and block matching can fail in finding the true movement. The refinement is computed on the depth maps in order to have a better estimation.

Finally, we perform a complete end-to-end coding of depth sequences. The rate-distortion performance, computed with the Bjontegaard metric [Bjo01], are shown in

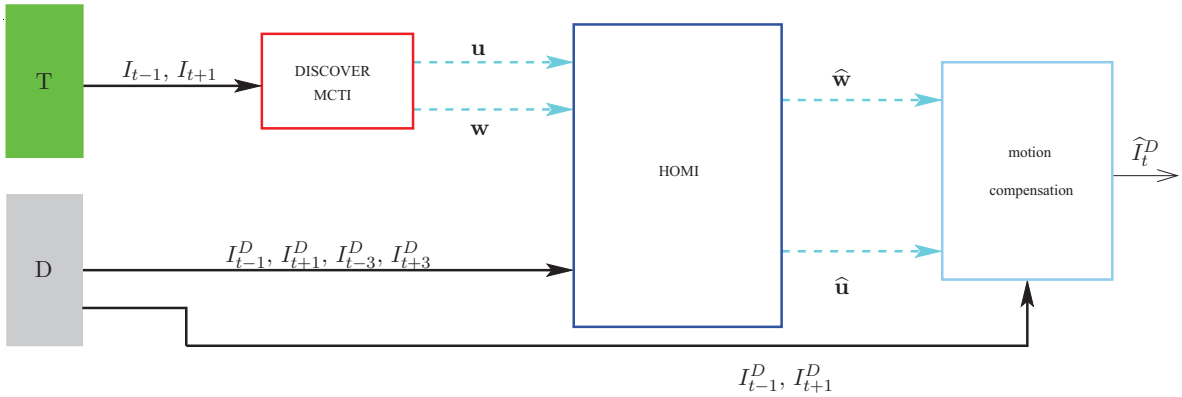


Figure 5.13: TD-ZH method: the frames I_{t-1}, I_{t+1} are used for the DISCOVER MCTI algorithm. The vectors $\mathbf{u}$ and $\mathbf{w}$ are obtained. These vectors are refined by HOMI algorithm implemented on I_{t-1}^D, I_{t+1}^D, I_{t-3}^D, I_{t+3}^D. The produced MVFs $\widehat{\mathbf{u}}$ and $\widehat{\mathbf{w}}$ are applied to I_{t-1}^D and I_{t+1}^D for producing the SI $\widehat{I}_t^D$

GOP size	2	4	8
$\lambda_{\text{TD-TH}}$	50	20	0
$\lambda_{\text{ZD-ZH}}$	10	8	2
$\lambda_{\text{TD-ZH}}$	4	2	2

Table 5.9: Values of λ for different techniques and GOP sizes

Tab. 5.12 and 5.13. We note that with the method TD-ZH we obtain a rate reduction up to 11.06% with respect to ZD and a PSNR improvements up to 0.44 dB.

ballet				
QP	ZD-ZH	TD	TD-TH	TD-ZH
GOP size = 2				
31	0.25	0.28	0.54	**0.56**
34	0.22	0.29	0.55	**0.57**
37	0.22	0.15	0.39	**0.41**
40	0.20	0.18	0.37	**0.41**
GOP size = 4				
31	0.34	0.15	0.56	**0.59**
34	0.25	0.04	0.49	**0.52**
37	0.32	0.00	0.49	**0.52**
40	0.27	-0.01	0.57	**0.62**
GOP size = 8				
31	0.34	0.06	0.29	**0.52**
34	0.31	0.05	0.28	**0.58**
37	0.27	-0.04	0.16	**0.54**
40	0.26	-0.04	0.14	**0.67**

Table 5.10: Δ_{PSNR} [dB] for depth map sequence *ballet*

breakdancers				
QP	ZD-ZH	TD	TD-TH	TD-ZH
GOP size = 2				
31	0.11	0.06	0.10	**0.12**
34	0.11	0.06	0.11	**0.12**
37	**0.11**	-0.01	0.02	0.03
40	**0.09**	-0.03	0.00	0.01
GOP size = 4				
31	0.03	0.03	0.16	**0.17**
34	0.02	0.01	0.18	**0.18**
37	0.01	0.00	0.21	**0.22**
40	0.01	-0.02	**0.29**	**0.29**
GOP size = 8				
31	0.01	0.03	0.07	**0.16**
34	0.02	0.02	0.06	**0.19**
37	0.02	0.02	0.06	**0.25**
40	0.01	-0.03	0.03	**0.31**

Table 5.11: Δ_{PSNR} [dB] for depth map sequence *breakdancers*

ballet				
	ZD-ZH	TD	TD-TH	TD-ZH
GOP size = 2				
Δ_{R} (%)	-3.46	-2.05	-5.83	**-6.08**
Δ_{PSNR} [dB]	0.17	0.10	0.29	**0.31**
GOP size = 4				
Δ_{R} (%)	1.22	8.29	-3.38	**-4.42**
Δ_{PSNR} [dB]	-0.04	-0.36	0.14	**0.17**
GOP size = 8				
Δ_{R} (%)	-4.34	7.22	-0.36	**-11.06**
Δ_{PSNR} [dB]	0.17	-0.29	0.04	**0.44**

Table 5.12: Rate-distortion performance for depth maps *ballet*by Bjontegaard metric [Bjo01]

breakdancers				
	ZD-ZH	TD	TD-TH	TD-ZH
GOP size = 2				
Δ_{R} (%)	**-2.09**	1.51	-0.44	-0.93
Δ_{PSNR} [dB]	**0.09**	-0.07	0.01	0.04
GOP size = 4				
Δ_{R} (%)	-0.59	7.55	-0.55	**-0.94**
Δ_{PSNR} [dB]	0.02	-0.35	0.01	**0.02**
GOP size = 8				
Δ_{R} (%)	-0.96	7.27	3.66	**-4.50**
Δ_{PSNR} [dB]	0.03	-0.32	-0.16	**0.17**

Table 5.13: Rate-distortion performance for depth maps *breakdancers* by Bjontegaard metric [Bjo01]

5.3 A new architecture for a multiview distributed video codec in the context of MVD

In several scenarios, since Criminisi inpainting aided by depth map information cannot be sufficient to fill the occlusion areas, layered depth video (LDV) [MSD+08] has been proposed in literature. In LDV only one central view and the residual information of the adjacent views are coded and sent to the decoder along with the corresponding depth maps. In this section, we propose a new architecture for distributed video coding in the context of multiview video plus depth, that is inspired by LDV. Without loss of generality, let us suppose that we have two range cameras that provide two texture images $I_{L,t}$ and $I_{R,t}$ and two depth maps $I^D_{L,t}$ and $I^D_{R,t}$ for each instant t . For the sake of simplicity, since the method that we are illustrating is independent from the time instant, we omit the dependence from the time t. If DIBR is applied, one single view (texture + depth) and the camera parameters of the two cameras are enough for reconstructing the other view, except for the occlusion areas. Then, we propose to send only one view (the right one, for example) and the occlusion areas of the other view (the left one). While in LDV structure proposed the two cameras can communicate, now we are supposing that we are in the context of distributed video coding and no inter camera communication is allowed between the two cameras. Then, the left camera must estimate the occlusion mask O_L, *i.e.* the points that are visible on the left camera but not on the right one (see Chapter 2), without communicating with the right camera. This problem can be solved using the information of the depth map. Basically, we perform two DIBR: one from left to right to estimate the right view depth map, and the second back from right to left, to estimate the occlusion for the left view (the pixels that are not visible in the right view). Finally we encode the occluded areas with an adaptive algorithm.

More precisely, the encoder structure of the proposed architecture for the coding of depth and texture of two views is shown in Fig. 5.14.

For the right view, the texture and the depth map are INTRA coded. Let I^e_R and $I^{D,e}_R$ be its encoded version. These frames will be called Key Frames (as in Stanford Architecture).

Let us consider now the left view. DIBR is applied to I^D_L and the occlusion areas are filled by Bertalmio inpainting. We obtain an estimation of the right depth map $\widehat{I}^D_R$. An example is shown in Fig. 5.15 for the sequence *mobile*.

In turn, this can be used for estimating the occlusion mask O_L. Indeed, if DIBR is again applied to $\widehat{I}^D_R$, the holes resulting in the synthesized left depth map are an estimation $\widehat{O}_L$ of the map O_L. An example of the synthesized left depth maps and the derived occlusion map is shown in Fig. 5.16. Since $\widehat{O}_L$ is an estimation of O_L, it could be no sufficient to reconstruct correctly the left texture image, because some holes already occur in the reconstructed left texture image at the decoder side. We verified this hypothesis by some preliminary tests.

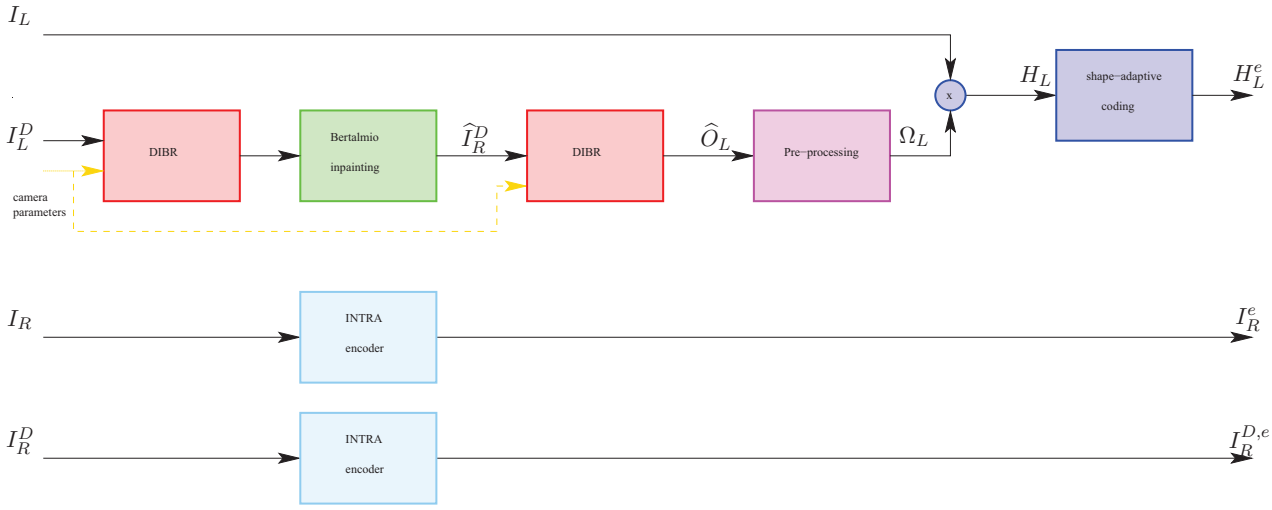
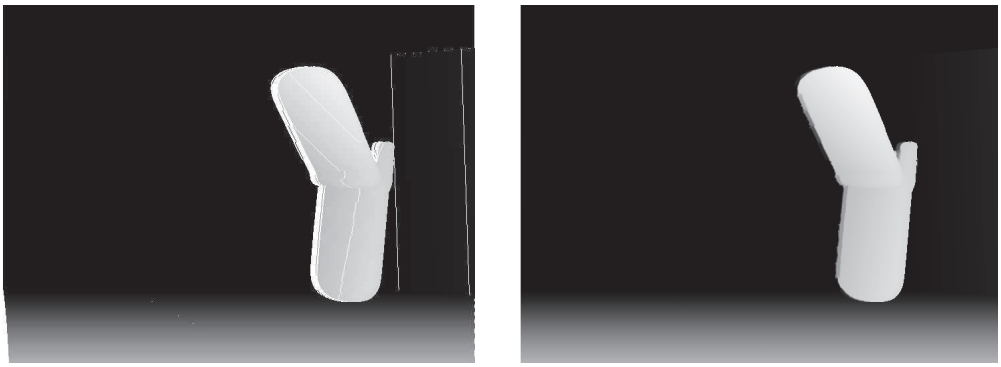


Figure 5.14: Structure of the encoder



(a) (b)

Figure 5.15: The frame $\widehat{I}_R^D$ before (a) and after Bertalmio inpainting (b)

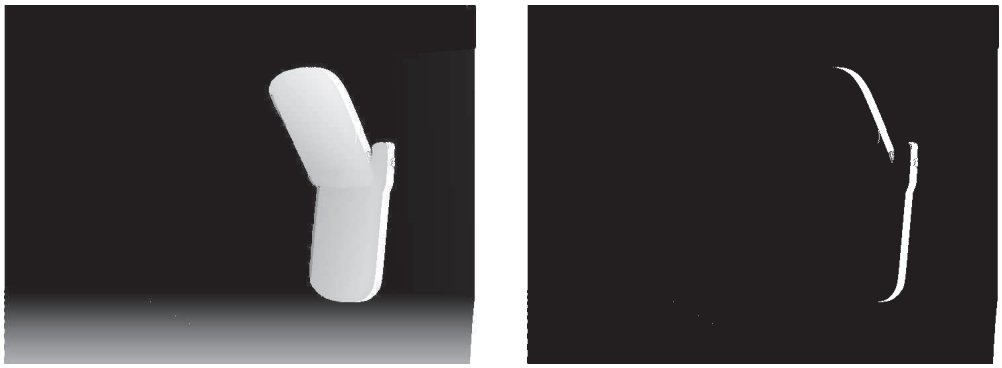
Then, a processing (illustrated in the next section) on the estimated occlusion map is necessary for avoiding this problem. Let Ω_L be the processed occlusion mask. We have to send only the regions of I_L indicated by the mask Ω_L, *i.e.* we have to send the frame H_L given by:

$$H_L(m,n) = \begin{cases} I_L(m,n) & \text{if } \Omega_L(m,n) = 1 \\ 0 & \text{otherwise} \end{cases}$$

This operation can be synthetically indicated as $H_L = \Omega_L I_L$.

An example of H_L is depicted in Fig. 5.17. If H_L presents isolated objects, we can code this frame in an efficient way by shape adaptive coding [LL00], [CPPV07]. Let H_L^e be its encoded version. In the following, the frames, for which only occlusion areas are sent, will be called O frames, like H_L.

The decoder structure is depicted in Fig. 5.18. Here, we can compute the actual



(a) (b)

Figure 5.16: The estimated left depth (a) and the estimation of occluded regions $\widehat{O}_L$ (b)

Figure 5.17: Example of H_L

occlusion on the right view. If the estimation of the encoder was correct, we can fill all the occluded areas with the encoded shapes. More precisely, DIBR is applied to the decoded right depth and texture images $\widehat{I}_R$ and $\widehat{I}_R^D$ for obtaining the estimation of the left depth and texture images $\widehat{I}_L^O$ and $\widehat{I}_L^{D,O}$. Obviously, these frames contain holes due to occlusion areas. The synthesized depth map is then inpainted and $\widehat{I}_L^D$ is obtained. The frame H_L^e is decoded and let $\widehat{H}_L$ be its decoded version. It contains the occluded areas that be needed to the frame $\widehat{I}_L^O$. Then, the reconstructed left texture frame is $J_L = \widehat{H}_L + \widehat{I}_L^O$. The remaining occluded areas are filled by Criminisi inpainiting and the frame $\widehat{I}_L$ is obtained.

5.3.1 Processing of the occlusion masks

Once the occlusion mask $\widehat{O}_L$ has been estimated at the encoder side, several isolated occlusion points occur (see Fig. 5.16(b)). These points can be real occlusion points as well errors in the estimation of the depth map. In general, we can avoid to send isolated occlusion points or small occluded regions, because they can be easily filled by Criminisi inpainting at the decoder side. Several algorithms of preprocessing of the depth maps have been proposed in literature, as in [DTPP07] in the case of small baseline between

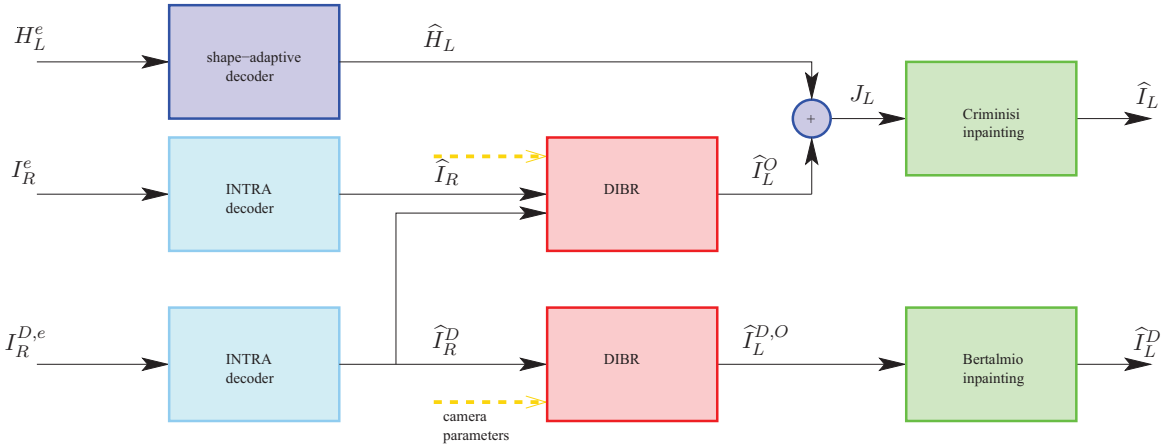


Figure 5.18: Structure of the decoder

Figure 5.19: Decoded frame without dilation of occluded areas

the cameras. Here, we propose to not sent regions whose area is smaller than p pixels: only large occlusion regions are sent. Let Ω_L the resulting occlusion map. However, due to the errors in the estimation, $H_L = \Omega_L I_L$ can be not sufficient to fill completely large occluded areas, as shown in Fig. 5.19. In order to improve the reconstructed image, we slightly modify Ω_L.

More precisely, we propose to dilate these large occluded areas in order to completely fill the occluded regions. The dilatation of a binary image A for a binary structuring element B is defined as

$$(A \oplus B)(m,n) = \sum_x \sum_y A(x,y) \cdot B(m-x, n-y)$$

where $\sum(\cdot)$ et $\cdot$ have to be considered as the boolean operations OR and AND, respectively. For our experiments, we choose as structuring element B a circle of radius ρ. By resuming, the small occlusion area of the mask O_L are eliminated and the remaining regions are dilated. The block diagram is in Fig. 5.20. The processed mask is Ω_L.

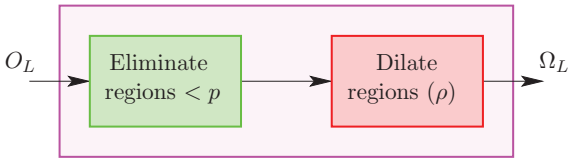


Figure 5.20: Processing of the occlusion mask O_L

Figure 5.21: Processed occlusion mask

Then, we search for the values of p and ρ that minimize the bit rate for coding the occluded regions, by maximizing the PSNR on the decoded and inpainted O-frames. From experimental results we remark that for p smaller than a certain value $\widehat{p}$, the rate increases, but the PSNR on the decoded O frames is nearly the same. Indeed, small regions are perfectly reconstructed by Criminisi inpainting, then they can be not sent. Likewise, for ρ greater than a certain threshold $\widehat{\rho}$, the rate increases, while the PSNR on the reconstructed frame is nearly the same, because the not occluded regions have already been estimated by the DIBR algorithm. Obviously, these values depend on the resolution of the video. We have proved that for *mobile* sequence the optimal value for these parameters are $\widehat{p} = 100$ and $\widehat{\rho} = 3$, at a resolution of 720×540 pixels. As an example, in Fig. 5.21 we show the occlusion mask of Fig. 5.16(b) processed by our algorithm.

5.3.2 Experimental results

In this section, we will show preliminary experimental results realized on one sequence. For validating our method, we consider a multiview video system with three range cameras. Three different camera configurations will be considered:

- All-INTRA Frame: all the frames of the three cameras are INTRA coded
- asymmetric scheme (see Section 3.4): we have two pure Key cameras and a central pure Wyner-Ziv camera. The Wyner-Ziv Frames are coded with the DISCOVER codec presented in Section 3.3.
- OKO: in this configuration we have a central Key camera. The frames of the two

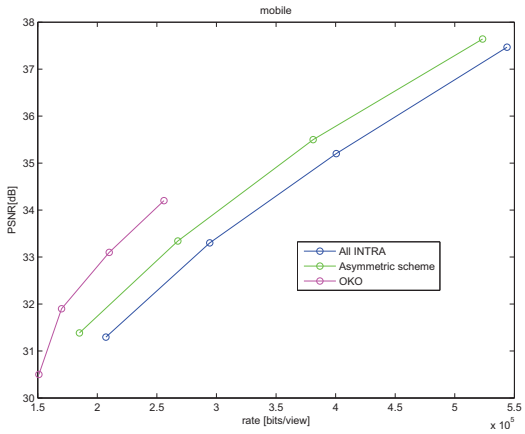


Figure 5.22: RD performance for the three different configurations

lateral cameras are O-frames, *i.e.* for these frames only the occluded regions are sent according to the codec proposed in this section.

The advantage of our structure is that we can consider a OKO structure, while in the DISCOVER codec a configuration with a central Key camera and two lateral WZF cameras is not possible because interpolation for WZF estimation could not be implemented. For our experiments, we consider the sequence *mobile* at 720×540 resolution. The Key frames are coded by H.264 INTRA for four quantization parameters 31, 34, 37 and 40. In Fig. 5.22, a 2D plot for the rate distortion performance of these three configurations is shown for the texture images. Our method improves the RD performance w.r.t. the DISCOVER codec, that in turn outperforms H.264/INTRA. We remark that the bit rate necessary for encoding the O-frames is much smaller than the Key Frames. Indeed, the occluded pixels are in average 7% of the pixels of the whole frame. Unfortunately also the PSNR is smaller than KFs or the decoded WZFs in the asymmetric structure, due to the distortion on the decoded depth maps of the key cameras used for the DIBR algorithm. As future work, we propose to code the depth maps of the key cameras in a more efficient way and not by H.264/INTRA modes. Indeed, more advanced algorithms for depth map coding [VCO+12] propose to improve the quality of synthesized frame and not the quality of the depth maps in themselves. In this manner, we expect an improvement on the quality of the areas of the O frames that have been synthesized by the DIBR algorithm and so the total bit rate.

We can also improve the quality of the synthesized regions by Wyner-Ziv coding those areas. Moreover, we can search for an optimal bit allocation among the occluded areas, coded by shape adaptive coding, and synthesized areas, obtained by DIBR and that can be eventually corrected by parity bits sent by a Wyner-Ziv encoder.

5.4 Publications

The works presented in this chapter has been the object of two published papers:

- Giovanni Petrazzuoli, Marco Cagnazzo, Frederic Dufaux, Béatrice Pesquet-Popescu, "Using distributed source coding and depth image based rendering to improve interactive multiview video access" *IEEE International Conference on Image Processing*, vol. 1, pp. 5-8, September 2011. Bruxelles, Belgium
- Giovanni Petrazzuoli, Marco Cagnazzo, Frederic Dufaux, Béatrice Pesquet-Popescu, "Wyner-Ziv coding for depth maps in multiview video-plus-depth", *In IEEE International Conference on Image Processing*, vol. 1, pp. 5-8, September 2011. Bruxelles, Belgium

Conclusion and future work

In this thesis, we first proposed different algorithms for improving the Side Information quality w.r.t. DISCOVER codec. Our contributions in this domain are:

- **High order temporal interpolation**: we improved the performance of DISCOVER codec, because we used four frames for trajectory interpolation instead of two. The higher order interpolation (HOMI) allows to consider motion models more complex than the linear one. This algorithm is applied also for recovering missing frames in the context of multiple description coding.
- **Refinement for large GOP size**: since in DISCOVER codec a hierarchical structure of estimation and decoding is implemented for GOP sizes larger than two, the quality of the decoded WZFs varies within the GOP. We have proposed to re-estimate the WZFs once all the WZFs of the GOP have been estimated. These new SI are decoded by using the same parity bits previously sent.
- **Inter-view interpolation with priority to large values of disparity**: we improved also the quality of the estimation in the view domain, with an interpolation algorithm that gives priority to large disparity values (IPLD). Indeed, objects that have a large disparity from one view to another one are close to the image plane. Thus, we give priority to these objects during the interpolation procedure.
- **Fusion with occlusion detection**: several algorithms have been proposed to fuse the inter-view interpolation and the temporal one. In the inter-view interpolation, more occlusion points occurs from one view to the other one. These points generate mismatching that affects the quality of the side information. We have proposed to estimate the occlusion areas and to fill them by pixels found in the temporal SI. Otherwise, a fusion of the two SI is applied.
- **Adaptive validation**: when the quality of the SI in one domain is much higher than in the other one, fusion does not improve RD performance. It is better to choose directly the best SI. We have proposed to detect the best SI by an adaptive validation based on the channel rate of the feedback channel. Thus, the number of bits to correct the SI are an indication of the quality of this estimation.

In the final part of this thesis, we have focused our attention on Multiview Video plus Depth (MVD) format. In particular, we studied different applications of DVC to MVD. Our contributions in this direction are the following:

- **Interactive multiview video streaming (IMVS)**: The IMVS paradigm imposes that only the frames requested by the user are sent to the decoder. As opposed to classical coding like H.264/AVC, DVC assures the correct decoding of all frames when a view switching occurs, without interrupting the continuity of the playback. We have proposed several methods for the generation of the WZF estimation in correspondence of the view switching, with the aid of the depth maps.
- **Coding of the Wyner-Ziv depth maps**: we propose several methods for the estimation of the Wyner-Ziv depth maps, that exploit also the information of the texture image.
- **A new coding architecture for distributed multiview video plus depth coding**: inspired by layered depth videos, we have proposed a new architecture for multiview DVC, where alternated views are INTRA coded and only occlusion areas are sent for the other views. Differently from LDV, the cameras cannot communicate each other.

As future works, we can develop the following ideas:

- **Adaptive validation algorithm with scene change detection**: Using some of the existing methods for boundary detection (for scene change) in video sequences, we can find a set of homogenoues frames. This can be useful for reducing the update frequency of the fusion decision, thus reducing as well the bit-rate overhead.
- **Side information improvement for texture video in the context MVD**: IPLD algorithm has been proposed in the context of multiview video coding, where depth maps are often not available. Indeed, we use the disparity as an estimation of inverse depth. We can adapt IPLD algorithm in the context of MVD: information on the depth can be used for improving the inter-view and temporal interpolation, by detecting occlusion.
- **Extrapolation method for IMVS**: in the context of IMVS, in order to better assure the constraint on the real time, we can apply extrapolation techniques instead of motion interpolation of DISCOVER for Wyner-Ziv Frames estimation
- **Bit allocation for the new codec for distributed multiview video plus depth coding**: the PSNR of the O frames is lower than the PSNR of the KFs, because of the areas synthesized by DIBR. We can use Wyner-Ziv coding on these areas: sending parity bits will improve their quality. However, a key issue of this

approach consists in finding an optimal allocation between the parity bits and the bits devoted to the shape adaptive representation of occluded areas. A model based approach could allow to implement a first solution.

Publications

Journal articles

1. Giovanni Petrazzuoli, Marco Cagnazzo, Beatrice Pesquet-Popescu, "Novel solutions for Side Information generation and fusion in multiview distributed video coding" [submitted to IEEE Transactions on Multimedia]

2. Giovanni Petrazzuoli, Thomas Maugey, Marco Cagnazzo, Beatrice Pesquet-Popescu, "A new DVC architecture for multiple video plus depth based on occlusion detection and object coding". [In preparation].

Conference papers

1. Giovanni Petrazzuoli, Marco Cagnazzo, Beatrice Pesquet-Popescu, "High order motion interpolation for side information improvement in DVC", *International Conference on Acoustics, Speech and Signal Processing*, Dallas, TX 2010

2. Giovanni Petrazzuoli, Marco Cagnazzo, Beatrice Pesquet-Popescu, "Fast and efficient side information generation in Distributed Video Coding by using dense motion representation", *European Signal Processing Conference*, Aalborg, Denmark 2010

3. Giovanni Petrazzuoli, Thomas Maugey, Marco Cagnazzo, Beatrice Pesquet-Popescu, "Side Information Refinement for Long Duration GOPs in DVC" *IEEE Workshop on Multimedia Signal Processing*, vol. 1, October 2010. Saint-Malo, France

4. Giovanni Petrazzuoli, Marco Cagnazzo, Frederic Dufaux, Beatrice Pesquet-Popescu, "Using distributed source coding and depth image based rendering to improve interactive multiview video access" *IEEE International Conference on Image Processing*, vol. 1, pp. 5-8, September 2011. Bruxelles, Belgium

5. Giovanni Petrazzuoli, Marco Cagnazzo, Frederic Dufaux, Beatrice Pesquet-Popescu, "Wyner-Ziv coding for depth maps in multiview video-plus-depth", *In IEEE International Conference on Image Processing*, vol. 1, pp. 5-8, September 2011. Bruxelles, Belgium

6. Claudio Greco, Giovanni Petrazzuoli, Marco Cagnazzo, and Béatrice Pesquet-Popescu, "An MDC-based video streaming architecture for mobile networks", *Proceedings of the IEEE Workshop on Multimedia Signal Processing*, Hangzhou, People's Republic of China, October

2011

7. Abdal Bassir Abou-Elailah, Giovanni Petrazzuoli, Frederic Dufaux, Joumana Farah, Marco Cagnazzo, Béatrice Pesquet-Popescu, “Side information Improvement in Transform-Domain Distributed Video Coding” *SPIE Application of Digital Image Processing XXXV*, August 2012. San Diego, California

Appendix

SI quality improvement for HOMI

In this section, we list all the PSNR improvements on the side information of the proposed algorithms for temporal interpolation (HOMI8, HOMI4, FastHOMI8, FastHOMI4), introduced in Chapter 4, w.r.t. DISCOVER MCTI algorithm for several sequences at CIF resolution. We consider different quantization parameters (QP) for coding the KFs.

In Tab. 5.14, we have the results for HOMI 8: we remark a PSNR improvement up to 0.54 dB for GOP size equal to 2, 0.76 dB for GOP size equal to 4 and 0.61 dB for GOP size equal to 8. The PSNR improvements are better for HOMI 4. The results are listed in Tab. 5.15. For example, for *foreman* sequence we pass from 0.31 dB (obtained by HOMI8) to 0.70 dB (obtained by HOMI4) of improvement w.r.t. DISCOVER algorithm for GOP size equal to 2. Similar results are obtained for GOP size equal to 4 and 8. Also results for FastHOMI8 and FastHOMI4 in Tab. 5.16 and 5.17 are quite good except for GOP size equal to 8. For example, for *foreman* sequence, we pass from 0.54 dB of improvement obtained with HOMI8 to 0.31 dB obtained with the fast version. Also for *city* we pass from 0.54 to 0.48 dB of improvement, but the complexity of FastHOMI8 is nearly the half of HOMI8. For GOP size equal to 8, in several cases FastHOMI8 and 4 do not improve the performance of DISCOVER. For example, for *eric* sequence we pass from 0.61 dB of improvement with HOMI 8 to -0.01 dB with FastHOMI8 and 0.02 dB with FastHOMI4. All these results are coherent with the rate-distortion analysis already shown in chapter IV.

	GOP size = 2				GOP size = 4				GOP size = 8			
	31	34	37	40	31	34	37	40	31	34	37	40
container	-0.00	-0.01	0.06	0.00	-0.01	-0.00	-0.00	0.00	-0.02	0.01	-0.01	-0.01
akiyo	0.05	0.03	0.02	0.00	0.12	0.16	0.08	0.07	0.09	0.11	0.14	0.10
news	0.06	0.04	0.02	0.01	0.06	0.04	0.03	0.02	0.06	0.06	0.01	0.00
hall	0.10	0.06	0.05	0.04	0.05	0.08	0.05	0.05	0.04	0.01	0.04	0.01
eric	0.35	0.27	0.22	0.13	0.76	0.70	0.63	0.46	0.61	0.55	0.46	0.43
paris	0.02	-0.00	-0.02	-0.01	0.16	0.14	0.12	0.11	0.27	0.27	0.27	0.22
silent	0.10	0.08	0.05	0.03	0.13	0.12	0.08	0.07	0.14	0.10	0.13	0.09
waterfall	0.02	0.01	0.01	0.01	-0.02	-0.01	-0.01	-0.00	-0.01	-0.02	-0.00	0.00
flower	-0.23	-0.18	-0.12	-0.06	-0.15	-0.12	-0.09	-0.03	-0.05	-0.02	0.04	0.06
foreman	0.31	0.22	0.15	0.09	0.31	0.42	0.49	0.15	0.46	0.43	0.38	0.33
tempete	-0.08	-0.06	-0.05	-0.04	0.04	0.01	0.01	0.01	0.00	-0.00	0.01	0.01
football	0.35	0.32	0.29	0.26	0.26	0.24	0.24	0.24	0.24	0.23	0.25	0.20
city	0.54	0.35	0.36	0.34	0.43	0.49	0.26	0.20	0.57	0.65	0.55	0.48
coastguard	-0.02	-0.02	-0.01	-0.00	-0.07	-0.06	-0.06	-0.03	-0.07	-0.06	-0.04	-0.03
mean	**0.11**	**0.08**	**0.07**	**0.05**	**0.15**	**0.16**	**0.13**	**0.09**	**0.17**	**0.16**	**0.16**	**0.13**

Table 5.14: SI improvement [dB] - HOMI 8 method - GOP size = 2, 4 and 8

	GOP size = 2				GOP size = 4				GOP size = 8			
	31	34	37	40	31	34	37	40	31	34	37	40
container	0.05	0.28	0.10	0.09	0.03	0.12	0.09	-0.02	0.07	0.09	0.07	0.00
akiyo	0.09	0.05	0.03	0.04	0.13	0.13	0.09	0.06	0.11	0.15	0.12	0.06
news	0.19	0.12	0.13	0.03	0.22	0.17	0.15	0.11	0.02	-0.03	-0.04	-0.01
hall	0.26	0.19	0.12	0.10	0.20	0.16	0.06	0.03	0.15	0.04	0.09	0.08
eric	0.20	0.15	0.15	0.02	0.61	0.56	0.54	0.35	0.44	0.47	0.47	0.35
paris	0.19	0.16	0.10	0.08	0.37	0.32	0.29	0.22	0.43	0.42	0.40	0.31
silent	0.17	0.14	0.09	0.07	0.19	0.22	0.09	0.13	0.18	0.12	0.13	0.12
waterfall	0.04	0.04	0.03	0.01	0.05	0.03	0.02	0.01	0.03	-0.01	-0.00	0.01
flower	-0.22	-0.13	-0.08	-0.03	-0.18	-0.12	-0.10	-0.01	0.04	0.12	0.29	0.21
foreman	0.70	0.62	0.36	0.03	0.85	0.76	0.79	0.31	0.56	0.62	0.62	0.45
tempete	-0.13	-0.10	-0.08	-0.04	0.01	-0.02	-0.01	0.04	0.00	-0.01	0.00	0.07
football	0.45	0.43	0.42	0.30	0.25	0.28	0.25	0.22	0.23	0.30	0.25	0.20
city	0.45	0.30	0.31	0.35	0.43	0.28	0.24	0.06	0.56	0.39	0.43	0.41
coastguard	-0.13	-0.06	-0.03	-0.03	-0.13	-0.11	-0.09	-0.06	-0.16	-0.15	-0.16	-0.10
mean	**0.16**	**0.15**	**0.11**	**0.07**	**0.21**	**0.19**	0.17	0.10	**0.19**	**0.18**	**0.19**	**0.15**

Table 5.15: HOMI 4 method - GOP size = 2, 4 and 8

	GOP size = 2				GOP size = 4				GOP size = 8			
	31	34	37	40	31	34	37	40	31	34	37	40
container	0.00	-0.04	0.07	-0.00	0.04	0.10	0.01	0.08	-0.07	0.03	-0.13	-0.06
akiyo	0.13	0.08	0.05	0.02	0.10	0.13	0.08	0.07	0.48	0.49	0.56	0.48
news	0.09	0.07	0.04	0.03	0.11	0.08	0.09	0.13	-0.08	-0.05	-0.12	-0.08
hall	0.00	-0.02	0.00	0.01	-0.13	-0.10	-0.06	-0.04	-1.70	-1.42	-1.27	-0.99
eric	0.30	0.23	0.18	0.12	0.75	0.69	0.59	0.50	-0.01	0.06	0.01	-0.04
paris	-0.05	-0.05	-0.03	-0.03	0.14	0.15	0.13	0.13	0.02	0.07	0.05	0.01
silent	-0.03	-0.03	-0.02	-0.02	-0.10	-0.03	-0.03	0.02	0.81	0.75	0.67	0.56
waterfall	0.04	0.02	0.01	0.01	0.01	0.03	0.06	0.10	0.00	-0.01	-0.02	-0.01
flower	-0.34	-0.25	-0.17	-0.12	0.03	0.02	0.05	0.06	-0.65	-0.68	-0.52	-0.41
foreman	0.18	0.12	0.03	0.02	0.47	0.48	0.41	0.22	0.58	0.58	0.55	0.45
tempete	-0.16	-0.14	-0.12	-0.09	-0.13	-0.11	-0.10	-0.09	-0.53	-0.51	-0.45	-0.40
football	0.07	0.06	0.06	0.03	0.10	0.10	0.12	0.10	-0.10	-0.08	-0.07	-0.13
city	0.48	0.35	0.39	0.37	0.58	0.46	0.19	0.23	0.62	0.53	0.34	0.40
coastguard	0.07	0.05	0.03	0.02	0.04	0.03	0.04	0.05	-0.43	-0.39	-0.38	-0.34
mean	**0.05**	**0.03**	**0.03**	**0.02**	**0.14**	**0.14**	**0.11**	**0.11**	**-0.07**	**-0.04**	**-0.05**	**-0.04**

Table 5.16: Fast HOMI 8 method - GOP size = 2, 4 and 8

	GOP size = 2				GOP size = 4				GOP size = 8			
	31	34	37	40	31	34	37	40	31	34	37	40
container	0.00	-0.04	0.08	0.01	0.05	0.13	0.01	0.07	-0.03	0.01	-0.13	-0.06
akiyo	0.13	0.08	0.05	0.03	0.09	0.16	0.08	0.06	0.49	0.54	0.55	0.47
news	0.09	0.07	0.05	0.03	0.10	0.09	0.10	0.13	-0.10	-0.06	-0.10	-0.08
hall	0.03	0.01	0.02	0.02	-0.14	-0.07	-0.05	-0.03	-1.68	-1.38	-1.24	-0.96
eric	0.30	0.24	0.19	0.12	0.74	0.69	0.61	0.50	0.02	0.08	-0.00	-0.01
paris	-0.06	-0.05	-0.03	-0.03	0.13	0.13	0.13	0.13	0.01	0.05	0.06	0.01
silent	-0.03	-0.03	-0.01	-0.01	-0.10	-0.01	-0.02	0.02	0.80	0.74	0.67	0.56
waterfall	0.04	0.02	0.01	0.01	0.01	0.03	0.06	0.10	-0.01	-0.01	-0.02	-0.00
flower	-0.33	-0.24	-0.16	-0.11	0.01	-0.00	0.02	0.05	-0.65	-0.69	-0.52	-0.40
foreman	0.18	0.12	0.06	0.01	0.47	0.45	0.41	0.22	0.60	0.58	0.56	0.46
tempete	-0.16	-0.14	-0.11	-0.09	-0.12	-0.10	-0.10	-0.09	-0.53	-0.50	-0.45	-0.41
football	0.08	0.08	0.07	0.04	0.13	0.12	0.15	0.10	-0.10	-0.07	-0.04	-0.13
city	0.42	0.34	0.38	0.35	0.43	0.45	0.23	0.28	0.48	0.40	0.30	0.42
coastguard	0.07	0.04	0.03	0.01	0.04	0.03	0.03	0.05	-0.43	-0.39	-0.37	-0.34
mean	**0.05**	**0.03**	**0.04**	**0.03**	**0.13**	**0.15**	**0.11**	**0.11**	**-0.08**	**-0.05**	**-0.05**	**-0.04**

Table 5.17: Fast HOMI 4 method - GOP size = 2, 4 and 8

SI quality improvement for HOMI for multiview video coding

We have implemented also HOMI algorithm on the view domain, by considering a multiview video system with seven aligned cameras disposed as in an asymmetric scheme. Then, for estimating the frames of the central view (#4), we interpolate by HOMI algorithm the views #1, #3, #5 and #7. We compare the performance of this algorithm w.r.t. DISCOVER applied on the view #3 and #5. Since the cameras are aligned, the displacement of the objects from one view to the other one is linear. Then, no improvement w.r.t. DISCOVER is expected if we consider more than two cameras for an higher order interpolation. The results in Tab. 5.18 confirm our expectation. On the other hand, we cannot apply HOMI for a set of not aligned cameras, because our algorithm is essentially based on block matching.

	Δ_R	Δ_{PSNR}
book arrival	0.25	-0.08
door flowers	0.37	-0.10
leaving laptop	0.30	-0.03
newspaper	0.50	-0.10
lovebird	0.80	-0.20
outdoor	0.02	0.00

Table 5.18: Rate Distortion Performance for HOMI applied on view domain by using four views

Bibliography

[AABP07] J. Areia, J. Ascenso, C. Brites, and F. Pereira, "Wyner-ziv stereo video coding using a side information fusion approach", in *Proceedings of IEEE Workshop on Multimedia Signal Processing*, pp. 453 –456, oct 2007. sections 3.4.1

[AAD+07] X. Artigas, J. Ascenso, M. Dalai, S. Klomp, D. Kubasov, and M. Ouaret, "The Discover codec: Architecture, techniques and evaluation", in *Proceedings of Picture Coding Symposium*, Lisbon, Portugal, Nov. 2007. sections 3.3

[AAT06] X. Artigas, E. Angeli, and L. Torres, "Side information generation for multiview distributed video coding using a fusion approach", in *Signal Processing Symposium, 2006. NORSIG 2006. Proceedings of the 7th Nordic*, pp. 250 –253, june 2006. sections 3.4.1

[ABD+10] J. Ascenso, C. Brites, F. Dufaux, A. Fernando, T. Ebrahimi, F. Pereira, and S. Tubaro, "The VISNET ii DVC codec: architecture, tools and performance", in *Proceedings of European Signal Processing Conference*, 2010. sections 3.3.2

[AEDF+12] A. Abou-Elailah, F. Dufaux, J. Farah, M. Cagnazzo, and B. Pesquet-Popescu, "Fusion of global and local motion estimation for distributed video coding". *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 22, pp. 1–12, 2012. sections 3.3.2

[AG02] A. Aaron and B. Girod, "Compression with side information using turbo codes", in *Data Compression Conference, 2002. Proceedings. DCC 2002*, pp. 252 – 261, Mar. 2002. sections 3.3.1

[AP08] J. Ascenso and F. Pereira, "Advanced side information creation techniques and framework for Wyner-Ziv video coding". *Journal of Visual Communication and Image Representation*, vol. 19 (8), pp. 600–613, 2008. sections 3.3.2

[Apo01] J. G. Apostolopoulos, "Reliable video communication over lossy packet networks using multiple state encoding and path diversity", in *Proceedings of SPIE International Symposium on Visual Communications and Image Processing*, 2001. sections 4.4

[ASG03] A. Aaron, E. Setton, and B. Girod, "Towards practical Wyner-Ziv coding of video", in *IEEE International Conf. on Image Processing*, Barcelona, Spain, 2003. sections (document), 3.3.2

[AZG02] A. Aaron, R. Zhang, and B. Girod, "Wyner-Ziv coding of motion video", in *Asilomar Conference on Signals and Systems*, Pacific Grove, California, Nov. 2002. sections (document), 3.2, 3.3.2

[BBH03] M. Brown, D. Burschka, and G. Hager, "Advances in computational stereo". *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 25 (8), pp. 993 – 1008, aug. 2003. sections 2.3, 2.4.1

[BBS01] M. Bertalmio, A. Bertozzi, and G. Sapiro, "Navier-stokes, fluid dynamics, and image and video inpainting", in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1, pp. I–355 – I–362 vol.1, 2001. sections 2.7.1

[Bjo01] G. Bjontegaard, "Calculation of average PSNR differences between RD-curves", in *VCEG Meeting*, Austin, USA, Apr. 2001. sections (document), 4.1.4, 4.1.6, 4.4.4, 5.1.3, 5.3, 5.4, 5.5, 5.6, 5.2.1, 5.12, 5.13

[Bov00] A. Bovik, *Handbook of Image & Video Processing*, Elsevier Academic Press, 2000. sections 1.1

[BP08] C. Brites and F. Pereira, "Correlation noise modeling for efficient pixel and transform domain Wyner-Ziv video coding". *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 18 (9), pp. 1177 –1190, sept. 2008. sections 3.3.2

[CAB10] M. Cagnazzo, M. Antonini, and M. Barlaud, "Mutual information-based context quantization". *Signal Processing: Image Communication (Elsevier Science)*, vol. 25 (1), pp. 64–74, Jan. 2010. sections 4.4.1

[CM89] S. Cochran and G. Medioni, "Accurate surface description from binocular stereo", in *Interpretation of 3D Scenes, 1989. Proceedings., Workshop on*, pp. 16 –23, nov 1989. sections 2.4.4

[CMMPP09] M. Cagnazzo, W. Miled, T. Maugey, and B. Pesquet-Popescu, "Image interpolation with edge-preserving differential motion refinement", in *Proceedings of IEEE Internantional Conference on Image Processing*, Cairo, Egypt, 2009. sections 3.3.2, 3.4.1

[CMPP09] M. Cagnazzo, T. Maugey, and B. Pesquet-Popescu, "A differential motion estimation method for image interpolation in distributed video coding", in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 1, pp. 1861–1864, Taiwan, 2009. sections 3.3.2, 3.4.1

[COC11] G. Cheung, A. Ortega, and N. Cheung, "Interactive streaming of stored multiview video using redundant frame structures". *IEEE Transactions on Image Processing*, vol. 20 (3), pp. 744 –761, Mar. 2011. sections (document), 5.1, 5.1.1

[CPPV07] M. Cagnazzo, S. Parrilli, G. Poggi, and L. Verdoliva, "Costs and advantages of object-based image coding with shape-adaptive wavelet transform". *EURASIP Journal on Image and Video Processing*, vol. 2007, pp. Article ID 78323, 13 pages, 2007, doi:10.1155/2007/78323. sections (document), 5.3

[CPT04] A. Criminisi, P. Perez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting". *Image Pro., IEEE Trans.*, vol. 13 (9), pp. 1200 –1212, Sep. 2004. sections 2.7.1

[CT91] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, John Wiley & Sons, 1991. sections (document), 3.1

[CWU+08] Y. Chen, Y.-K. Wang, K. Ugur, M. M. Hannuksela, J. Lainema, and M. Gabbouj, "The emerging MVC standard for 3D video services". *EURASIP J. Appl. Signal Process.*, vol. 2009, pp. 8:1–8:13, January 2008. sections 2.6

[DBJ+12] N. Deligiannis, J. Barbarien, M. Jacobs, A. Munteanu, A. Skodras, and P. Schelkens, "Side-information-dependent correlation channel estimation in hash-based distributed video coding". *Image Processing, IEEE Transactions on*, vol. 21 (4), pp. 1934 –1949, april 2012. sections 3.3.2

[DMC+09a] N. Deligiannis, A. Munteanu, T. Clerckx, J. Cornelis, and P. Schelkens, "Correlation channel estimation in pixel-domain distributed video coding", in *Image Analysis for Multimedia Interactive Services, 2009. WIAMIS '09. 10th Workshop on*, pp. 93 –96, may 2009. sections 3.3.2

[DMC+09b] ———, "Overlapped block motion estimation and probabilistic compensation with application in distributed video coding". *Signal Processing Letters, IEEE*, vol. 16 (9), pp. 743 –746, sept. 2009. sections 3.3.2

[DPP10] I. Daribo and B. Pesquet-Popescu, "Depth-aided image inpainting for novel view synthesis", in *Proceedings of IEEE Workshop on Multimedia Signal Processing*, pp. 167–170, Oct. 2010. sections (document), 2.15, 2.7.1, 2.16, 5.1.2

[DTPP07] I. Daribo, C. Tillier, and B. Pesquet-Popescu, "Distance dependent depth filtering in 3d warping for 3dtv", in *Multimedia Signal Processing, 2007. MMSP 2007. IEEE 9th Workshop on*, pp. 312 –315, oct. 2007. sections 5.3.1

[Duf11] F. Dufaux, "Support vector machine based fusion for multi-view distributed video coding", in *Proceedings of IEEE Workshop on Multimedia Signal Processing*, Corfu, Grece, 2011. sections 3.4.1, 4.3.3

[EFLBS07] A. El Fawal, J. Le Boudec, and K. Salamatian, "Multi-hop broadcast from theory to reality: practical design for ad-hoc networks", in *Proceedings of IEEE/ACM International Conference on Autonomic Computing and Communication Systems*, Rome, Italy, Oct. 2007. sections 4.4.3

[EL99] A. Efros and T. Leung, "Texture synthesis by non-parametric sampling", in *Proc. of the 7th IEEE Int. Conf. on Computer Vision (ICCV))*, 1999. sections 2.7.1

[EW02] G. Egnal and R. Wildes, "Detecting binocular half-occlusions: empirical comparisons of five approaches". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24 (8), pp. 1127 – 1133, aug 2002. sections (document), 4.3.1

[FAB07] P. Ferre, D. Agrafiotis, and D. Bull, "Fusion methods for side information generation in multi-view distributed video coding systems", in *Proceedings of IEEE Internantional Conference on Image Processing*, vol. 6, pp. VI –409 –VI –412, 2007. sections 3.4.1

[FC80] F. N. Fritsch and R. E. Carlson, "Monotone piecewise cubic interpolation". *SIAM J. Numerical Analysis*, vol. 17, pp. 238–246, 1980. sections 4.1

[Feh03] C. Fehn, "A 3D-TV Approach Using Depth-Image-Based Rendering (DIBR)", in *Proceedings of 3rd IASTED Conference on Visualization, Imaging, and Image Processing*, pp. 482–487, Benalmádena, Spain, Sep. 2003. sections 2.7

[FG07] M. Flierl and B. Girod, "Multiview video compression". *Signal Processing Magazine, IEEE*, vol. 24 (6), pp. 66–76, Nov. 2007. sections 5.1

[FJL00] M. Frodigh, P. Johansson, and P. Larsson, "Wireless ad-hoc networking: the art of networking without a network". *Ericsson Review*, vol. 4, pp. 248–263, 2000. sections 4.4.3

[FKdB+02] C. Fehn, P. Kauff, M. O. de Beeck, F. Ernst, W. Ijsselsteijn, M. Pollefeys, L. V. Gool, E. Ofek, and I. Sexton, "An evolutionary and optimised approach on 3D-TV", in *Int. Broadcast Convention*, Netherlands, 2002. sections 2.7

[Fua93a] P. Fua, "A parallel stereo algorithm that produces dense depth maps and preserves image features". *Machine Vision and Applications*, vol. 6, pp. 35–49, 1993, 10.1007/BF01212430. sections (document), 4.3.1

[Fua93b] ———, "A parallel stereo algorithm that produces dense depth maps and preserves image features". *Machine Vision and Applications*, vol. 6, pp. 35–49, 1993, 10.1007/BF01212430. sections 2.4.3

[GARRM05] B. Girod, A. Aaron, S. Rane, and D. Rebollo-Monedero, "Distributed video coding". *Proceedings of the IEEE*, vol. 93 (1), pp. 71–83, Jan. 2005. sections 3.1.2, 5.1.1

[GC11] C. Greco and M. Cagnazzo, "A cross-layer protocol for cooperative content delivery over mobile ad-hoc networks". *Inderscience International Journal of Communication Networks and Distributed Systems*, vol. 7 (1–2), pp. 49–63, Jun. 2011. sections 4.4, 4.4.3

[GCPP10] C. Greco, M. Cagnazzo, and B. Pesquet-Popescu, "H.264-based multiple description coding using motion compensated temporal interpolation", in *Proceedings of IEEE Workshop on Multimedia Signal Processing*, Saint-Malo, France, Oct. 2010. sections 4.4, 4.4.1, 4.4.2, 4.4.4

[GCPP11] ———, "ABCD : un protocole cross-layer pour la diffusion vidéo dans des réseaux sans fil ad-hoc", in *Proceedings of the Groupe d'Études du Traitement du Signal et des Images*, Bordeaux, France, Sep. 2011. sections 4.4.3

[Ger05] M. Gerla, "From battlefields to urban grids: new research challenges in ad-hoc wireless networks". *Elsevier Journal on Pervasive and Mobile Computing*, vol. 1 (1), pp. 77–93, 2005. sections 4.4.3

[GLW+06] X. Guo, Y. Lu, F. Wu, W. Gao, and S. Li, "Distributed multi-view video coding", in *Proceedings of SPIE International Symposium on Visual Communications and Image Processing*, vol. 6077, pp. 290–297, 2006. sections 3.4.1

[Goy01] V. K. Goyal, "Multiple description coding: compression meets the network". *IEEE Signal Processing Magazine*, vol. 18 (5), pp. 74–93, Sep. 2001. sections 4.4

[GPCPP11] C. Greco, G. Petrazzuoli, M. Cagnazzo, and B. Pesquet-Popescu, "An MDC-based video streaming architecture for mobile networks", in *Proceedings of IEEE Workshop on Multimedia Signal Processing*, Hangzhou, PRC, Oct. 2011. sections 4.4.3, 4.4.4

[GPT+07] C. Guillemot, F. Pereira, L. Torres, T. Ebrahimi, R. Leonardi, and J. Ostermann, "Distributed monoview and multiview video coding: Basics, problems and recent advances". *IEEE Signal Processing Magazine*, pp. 67–76, Sep. 2007. sections (document), 3.1.2, 3.4.1

[Han12] M. Hannuksela, "Test model for avc based 3d video coding", in *ISO/IEC JTC1/SC29/WG11 MPEG2012/N12558*, 2012. sections 2.7

[HF08] X. Huang and S. Forchhammer, "Improved side information generation for distributed video coding", in *Proceedings of IEEE Workshop on Multimedia Signal Processing*, pp. 223–228, 2008. sections 3.3.2

[Hir01] H. Hirschmüller, "Improvements in real-time correlation-based stereo vision", in *Proceedings of IEEE Workshop on Stereo and Multi-Baseline Vision*, pp. 141 – 148, Kauai, Hawaii, Dec. 2001. sections 4.3.1

[IDPP08] C. T. I. Daribo and B. Pesquet-Popescu, "Adaptive wavelet coding of the depth map for stereoscopic view synthesis", in *Proceedings of IEEE Workshop on Multimedia Signal Processing*, 2008. sections 2.7

[Joi07] Joint Video Team of ISO/IEC MPEG & ITU-T VCEG, *H.264/MPEG-4 AVC Reference Software Manual*, Jul. 2007. sections 1.7

[KCT06] E. Kurutepe, M. R. Civanlar, and A. M. Tekalp, "Interactive transport of multi-view videos for 3dtv applications", in *Packet VideoWorkshop*, Hangzhou, China, 2006. sections 5.1.1

[KG06] D. Kubasov and C. Guillemot, "Mesh-based motion-compensated interpolation for side information extraction in distributed video coding", in *Proceedings of IEEE Internantional Conference on Image Processing*, pp. 261 –264, oct 2006. sections 3.3.2

[KLG07] D. Kubasov, K. Lajnef, and C. Guillemot, "A hybrid encoder/decoder rate control for wyner-ziv video coding with a feedback channel", in *Proceedings of IEEE Workshop on Multimedia Signal Processing*, pp. 251 –254, Oct. 2007. sections 3.3.2

[KNG07] D. Kubasov, J. Nayak, and C. Guillemot, "Optimal reconstruction in wyner-ziv video coding with multiple side information", in *Proceedings of IEEE Workshop on Multimedia Signal Processing*, pp. 183 –186, Oct. 2007. sections 3.3.2

[KOL+09] W.-S. Kim, A. Ortega, P. Lai, D. Tian, and C. Gomila, "Depth map distortion analysis for view rendering and depth coding", in *Proceedings of IEEE Internantional Conference on Image Processing*, Cairo, Egypt, 2009. sections 2.7

[KVG06] T. Koninckx and L. Van Gool, "Real-time range acquisition by adaptive structured light". *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 28 (3), pp. 432 –445, march 2006. sections 2.7

[KZK07] E. Kozica, D. Zachariah, and W. Kleijn, "Interlacing intraframes in multiple-description video coding", in *Proceedings of IEEE Internantional Conference on Image Processing*, vol. 4, 2007. sections 4.4

[LCH12] S. Lee, O. Choi, and R. Horaud, *Time-of-Flight Cameras: Principles, Methods and Applications*, Springer Briefs in Computer Science, 2012. sections 2.7

[LGT05] K. Lim, S. G., and W. T., "Text description of joint model reference encoding methods and decoding concealment methods". *Joint Video Team Documentary JVT-O079*, Apr. 2005. sections 1.7

[LL00] S. Li and W. Li, "Shape-adaptive discrete wavelet transforms for arbitrarily shaped visual object coding". *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 10 (5), pp. 725 –743, aug 2000. sections (document), 5.3

[Llo82] S. Lloyd, "Least squares quantization in PCM". *IEEE Transactions on Information Theory*, vol. 28 (2), pp. 129–137, Mar. 1982. sections 1.2

[LWP11] J. Y. Lee, H.-C. Wey, and D.-S. Park, "A fast and efficient multi-view depth image coding method based on temporal and inter-view correlations of texture images". *Circuits and Systems for Video Technology, IEEE Transactions on*, Apr. 2011. sections 2.7

[MBAP09] R. Martins, C. Brites, J. Ascenso, and F. Pereira, "Refining side information for improved transform domain Wyner-Ziv video coding". *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 19 (9), pp. 1327–1341, 2009. sections 3.3.2

[MBP+09] B. Macchiavello, F. Brandi, E. Peixoto, R. de Queiroz, and D. Mukherjee, "Side-information generation for temporally and spatially scalable Wyner-Ziv codecs". *Journal on Image and Video Processing*, vol. 2009, p. 14, 2009. sections 3.3.2

[MGPPG10] T. Maugey, J. Gauthier, B. Pesquet-Popescu, and C. Guillemot, "Using an exponential power model forwyner ziv video coding", in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pp. 2338 –2341, march 2010. sections 3.3.2

[MMCPP09a] T. Maugey, W. Miled, M. Cagnazzo, and B. Pesquet-Popescu, "Fusion schemes for multiview distributed video coding", in *Proceedings of European Signal Processing Conference*, Glasgow, Scotland, 2009. sections (document), 5, 3.4.1, 4.3, 4.3.1, 4.3.3, 4.17, 4.19, 4.22, 4.3.4

[MMCPP09b] W. Miled, T. Maugey, M. Cagnazzo, and B. Pesquet-Popescu, "Image interpolation with dense disparity estimation in multiview distributed video coding", in *International Conference on Distributed Smart Cameras*, Como, Italy, 2009. sections 3.4.1

[MMS+08] P. Merkle, Y. Morvan, A. Smolic, D. Farin, K. Muller, P. de With, and T. Wiegand, "The effect of depth compression on multiview rendering quality", in *3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video, 2008*, 2008. sections 2.7

[MP06] W. Miled and J. Pesquet, "Disparity map estimation using a total variation bound", in *Computer and Robot Vision, 2006. The 3rd Canadian Conference on*, p. 48, june 2006. sections (document), 2.4.2, 2.7, 4.3.1

[MPP08] T. Maugey and B. Pesquet-Popescu, "Side information estimation and new symmetric schemes for multi-view distributed video coding". *Journal of Visual Communication and Image Representation*, vol. 19 (8), pp. 589–599, 2008. sections (document), 3.4

[MSCB00] M.Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting", in *Proc. ACM Conf. Comp. Graphics (SIGGRAPH)*, New Orleans, USA, 2000. sections 2.7.1

[MSD+08] K. Muller, A. Smolic, K. Dix, P. Kauff, and T. Wiegand, "Reliability-based generation and view synthesis in layered depth video", in *Multimedia Signal Processing, 2008 IEEE 10th Workshop on*, pp. 34 –39, oct. 2008. sections (document), 5.3

[MSMW07a] P. Merkle, A. Smolic, K. Muller, and T. Wiegand, "Efficient prediction structures for multiview video coding". *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 17 (11), pp. 1461 –1473, nov. 2007. sections 2.6

[MSMW07b] ———, "Multi-view video plus depth representation and coding", in *Proceedings of IEEE Internantional Conference on Image Processing*, San Antonio, TX, 2007. sections 5.2

[MSS+09] S. Mys, J. Slowack, J. Skorupa, P. Lambert, and R. V. de Walle, "Introducing skip mode in distributed video coding". *Signal Processing: Image Communication*, vol. 24 (3), pp. 200 – 213, 2009. sections 3.3.2

[ODE06] M. Ouaret, F. Dufaux, and T. Ebrahimi, "Fusion-based multiview distributed video coding", in *Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks*, VSSN '06, pp. 139–144, ACM, New York, NY, USA, 2006. sections 3.4.1

[OH06] H. Oh and Y. Ho, "H.264-based depth map sequence coding using motion information of corresponding texture video". *Advanced Concepts for Intelligent Vision Systems*, vol. 4319, pp. 898–907, 2006. sections 2.7

[Par62] E. Parzen, "On estimation of a probability density function and mode". *Annals of Mathematical Statistics*, vol. 33 (3), pp. 1065–1076, 1962. sections 4.3.4, 4.4.4

[PLKL12] S.-U. Park, Y.-Y. Lee, C.-S. Kim, and S.-U. Lee, "Efficient side information generation using assistant pixels for distributed video coding", in *Proceedings of Picture Coding Symposium*, pp. 161 –164, may 2012. sections 3.3.2

[PMR07] R. Puri, A. Majumdar, and K. Ramchandran, "PRISM: a video coding paradigm with motion estimation at the decoder". *IEEE Transactions on Image Processing*, vol. 16 (10), pp. 2436–2448, Oct. 2007. sections (document), 3.2

[Pro06] "Proxim ORiNOCO 11b/g client PC card specifications", 2006. sections 4.4.4

[RFW+10] I. Radulovic, P. Frossard, Y.-K. Wang, M. Hannuksela, and A. Hallapuro, "Multiple description video coding with H.264/AVC redundant pictures". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20 (1), pp. 144–148, Jan. 2010. sections 4.4

[SGK01] S. Shirani, M. Gallant, and F. Kossentini, "Multiple description image coding using pre-and post-processing", in *IEEE ITCC '01: International Conference on Information Technology: Coding and Computing*, 2001. sections 4.4

[SK00] H. Shum and S. Kang, "A review of image-based rendering techniques". *Proceedings of SPIE International Symposium on Visual Communications and Image Processing*, vol. 213, 2000. sections 3.4.1

[SKN+10] G. Shen, W.-S. Kim, S. Narang, A. Ortega, J. Lee, and H. Wey, "Edge-adaptive transforms for efficient depth map coding", in *Proceedings of Picture Coding Symposium*, 2010. sections 2.7

[SPW+10] J. Seo, D. Park, H.-C. Wey, S. Lee, and K. Sohn, "Motion information sharing mode for depth video coding", in *3DTV-Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON)*, 2010. sections 2.7

[SS02] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms". *International Journal of Computer Vision*, vol. 47, pp. 7–42, 2002, 10.1023/A:1014573219977. sections 2.3

[SS03] ———, "High-accuracy stereo depth maps using structured light", in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 195–202, IEEE, 2003. sections (document), 4.3.1

[Süh08] K. Sühring, "JM reference software release 14.2", Source Code, Sep. 2008. sections 4.4.4

[SW73] D. Slepian and J. K. Wolf, "Noiseless coding of correlated information sources". *IEEE Transactions on Information Theory*, vol. 19, pp. 471–480, Jul. 1973. sections 3.1

[TPPP07] C. Tillier, C. T. Petrişor, and B. Pesquet-Popescu, "A motion-compensated overcomplete temporal decomposition for multiple description scalable video coding". *EURASIP Journal on Image and Video Processing*, vol. 1, pp. 1–12, 2007. sections 4.4

[TT07] M. Tagliasacchi and S. Tubaro, "Hash-based motion modeling in wyner-ziv video coding", in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 1, pp. I–509 –I–512, Honolulu, HA (USA), april 2007. sections 3.3.2

[TTFY11] M. Tanimoto, M. Tehrani, T. Fujii, and T. Yendo, "Free-viewpoint TV". *Signal Processing Magazine, IEEE*, vol. 28 (1), pp. 67 –76, Jan. 2011. sections 2.7

[VAG06] D. Varodayan, A. Aaron, and B. Girod, "Rate-adaptive codes for distributed source coding". *Signal Processing*, vol. 86 (11), pp. 3123 – 3130, 2006, <ce:title>Special Section: Distributed Source Coding</ce:title>. sections 3.3.1

[VCO+12] G. Valenzise, G. Cheung, R. Oliveira, M. Cagnazzo, B. Pesquet-Popescu, and A. Ortega, "Motion prediction of depth video for depth-image-based rendering using don't care regions", in *Proceedings of Picture Coding Symposium*, 2012. sections 2.7, 5.3.2

[vdSSS05] M. van der Schaar and N. Sai Shankar, "Cross-layer wireless multimedia transmission: challenges, principles, and new paradigms". *IEEE Transactions on Wireless Communications*, vol. 12 (4), pp. 50–58, 2005. sections 4.4.3

[VWS11] A. Vetro, T. Wiegand, and G. Sullivan, "Overview of the stereo and multiview video coding extensions of the h.264/mpeg-4 avc standard". *Proceedings of the IEEE*, vol. 99 (4), pp. 626 –642, april 2011. sections 2.6

[VxJ+11] F. Verbist, N. xDeligiannis, M. Jacobs, J. Barbarien, A. Munteanu, P. Schelkens, and J. Cornelis, "Maximum likelihood estimation for the generation of side information in distributed video coding", in *Systems, Signals and Image Processing (IWSSIP), 2011 18th International Conference on*, pp. 1–4, June 2011. sections 3.3.2

[WAH92] J. Weng, N. Ahuja, and T. Huang, "Matching two perspective views". *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 14 (8), pp. 806 –825, aug 1992. sections 2.3

[WCRB04] D. Wang, N. Canagarajah, D. Redmill, and D. Bull, "Multiple description video coding based on zero padding", in *Proceedings of IEEE International Symposium on Circuits and Systems*, vol. 2, 2004. sections 4.4

[WZ76] A. Wyner and J. Ziv, "The rate-distortion function for source coding with side information at the receiver". *IEEE Transactions on Information Theory*, vol. 22, pp. 1–11, Jan. 1976. sections 3.1.2, 3.1.3

[YK06] K.-J. Yoon and I. S. Kweon, "Adaptive support-weight approach for correspondence search". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28 (4), pp. 650 –656, Apr. 2006. sections 4.3.1

[YODE08] S. Ye, M. Ouaret, F. Dufaux, and T. Ebrahimi, "Improved side information generation with iterative decoding and frame interpolation for distributed video coding", in *Proceedings of IEEE Internantional Conference on Image Processing*, pp. 2228–2231, 2008. sections 3.3.2

[YODE09] ———, "Improved side information generation for distributed video coding by exploiting spatial and temporal correlations". *Journal on Image and Video Processing*, vol. 2009, p. 4, 2009. sections 3.3.2

[YYJP07] Y. Yang, M. Yu, G. Jiang, and Z. Peng, "A transmission and interaction oriented free viewpoint video system". *Int. J. Circuits Syst. Signal Process*, vol. 1(4), pp. 310–316, 2007. sections 5.1.1

[ZYLL05] J. Zhai, K. Yu, J. Li, and S. Li, "A low complexity motion compensated frame interpolation method", in *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on*, pp. 4927 – 4930 Vol. 5, May 2005. sections 3.3.2

Acknowledgements

During these three years of PhD scholarship in Paris, many persons help me both for technical and moral support. At first, I would like to thank my advisors: Marco Cagnazzo, Béatrice Pesquet-Popescu and Giovanni Poggi, for giving me this chance and for assisting me in my works. In particular I would like to thank Marco Cagnazzo for his willingness in all moments for these three years, his support in my research activities and for the writing of this manuscript. I want also to thank Annalisa Verdoliva for her support and also for her reproaches. I want also to thank the members of the jury for accepting the invitation for my PhD Thesis Defense.

I would like to thank Davide and Sharon that support me all the days of these three years and for all the dinners spent together :). I would like to thank Giovanni and Giuseppe (Valenzise) for their friendship and their reviews of my works (!) and all my colleagues: Claudio, Irina, Raffaele, Rafael, Abdel Bassir, Manel, Mounir, Marco (Calemme), Nello, Hamlet, Alper, Francesca, Paul for all pleasant moments spent together. Many thanks also Giuseppe (Rossini).

I would like also to thank three friends who assisted me (also if geographically far from me) from different parts of the world: Giorgia, Giorgio and Vincenzo.

Finally, I would like to thank my family and in particular my brother, for visiting me during these three years also in precarious conditions :).