**Università degli studi di Padova**

Facoltà di Ingegneria

Corso di Laurea Magistrale in Ingegneria delle Telecomunicazioni

TESI DI LAUREA

# Algoritmi di codifica video 3D basati sulla segmentazione

**Relatore:** Prof. Calvagno Giancarlo

**Correlatore:** Dott. Milani Simone

**Laureando:** Vendraminetto Gian Luca

Padova, 24 Aprile 2012

*a mio zio Mauro*

# Sommario

Allo stato attuale, l'evoluzione della tecnologia ha reso possibile la diminuzione dei costi di attuazione e sviluppo di apparati di trasmissione e visualizzazione 3D. La trasmissione di un segnale tridimensionale richiede più banda rispetto una tradizionale sequenza video 2D, da qui l'esigenza di adottare efficienti schemi di codifica aventi lo scopo di comprimere efficacemente l'informazione garantendo al tempo stesso un'alta qualità visiva.

La compressione video tradizionale opera una partizione dei frame di input in blocchi regolari di pixel, i quali vengono poi approssimati mediante un processo di stima del movimento e codificati mediante trasformata basata su blocco. Recenti studi dimostrano che si possono ottenere risultati migliori mediante una partizione che sfrutti una strategia di segmentazione *object-oriented*, ossia adattando la dimensione delle regioni alla geometria ed alle caratteristiche degli oggetti ripresi dalla telecamera.

Dopo una trattazione delle caratteristiche del segnale 3D ed uno studio generale degli strumenti di cui si avvale l'attuale standard di codifica video H.264/AVC, l'interesse si è focalizzato sull'analisi ed approfondimento del processo di segmentazione orientato all'oggetto, con lo scopo di proporre degli algoritmi miranti all'ottimizzazione (in termini di efficienza di compressione e complessità computazionale) di tale procedimento all'interno di un *Group-of-Picture* (GOP). Verranno dunque confrontati i risultati derivanti da tale metodologia di codifica con quelli ottenuti mediante l'utilizzo dello standard H.264/AVC e, successivamente, si trarranno le conclusioni.

# Indice

# Capitolo 1

# Introduzione

La storia del 3D ha radici molto lontane, che affondano al 1838 con la scoperta dello stereogramma, ovvero un'immagine piana bidimensionale che dà l'illusione della profondità e della terza dimensione: due immagini similari affiancate, osservate frontalmente possono essere visualizzate come un'unica immagine tridimensionale grazie all'ausilio di uno strumento, lo stereoscopio, dotato di due lenti d'ingrandimento (o di un sistema di specchi), che permette ad ogni singolo occhio di osservare solo l'immagine ad esso destinata.

Da allora, nel corso dei decenni sono state proposte varie tecniche di visualizzazione 3D, soprattutto in ambito cinematografico, ma solamente negli ultimi anni, la crescente diffusione di schermi tridimensionali (quali gli schermi autostereoscopici), permessa da costi non più proibitivi e dai continui progressi in ambito tecnologico, ha definitivamente aperto la strada alla tanto attesa rivoluzione 3D, incrementando il livello di immersività e di coinvolgimento degli utenti che fruiscono di tale tecnologia.

Gli ambiti interessati dallo sviluppo e dalla diffusione del 3D sono molteplici: dal cinema all'intrattenimento televisivo, dai videogiochi alle videocomunicazioni e videoconferenze, senza dimenticare il campo medico. Tale diffusione ed interesse sono stati evidenziati da un recente studio condotto dalla Cisco Inc., nel quale si prevede che la trasmissione di contenuti video da e verso dispositivi mobili rappresenterà il 66% del traffico globale dei dati su

tali dispositivi entro il 2014 [1]. Di conseguenza sono sempre più richiesti efficaci schemi di codifica di dati video 3D per affrontare due dei più importanti problemi riguardanti la loro distribuzione. Il primo di questi è rappresentato dal bisogno di trasmettere una quantità di dati significativamente maggiore rispetto ai flussi tradizionali delle comunicazioni video, mentre il secondo problema consiste nella necessità di elaborare segnali differenti con caratteristiche eterogenee.

Un segnale video 3D è composto da segnale video standard (tessitura) e dati riguardanti la geometria (profondità). Un formato ampiamente utilizzato per la rappresentazione di sequenze video 3D è dato dal *Depth Based Image Rendering* (o DIBR), che associa ad ogni frame di una squenza video standard una mappa di profondità (*depth map*), la quale rappresenta la geometria dell'oggetto mediante una matrice bidimensionale di valori interi, ciascuno dei quali fornisce il valore di profondità del pixel ad esso associato in termini di distanza dal piano immagine della telecamera.

Fra gli schemi di visualizzazione 3D sta avendo sempre più diffusione il *Multi-view Video* (o MVV), nel quale vengono utilizzate più telecamere per acquisire contemporaneamente diverse prospettive della scena e permettere dunque una fruizione interattiva più realistica della scena tridimensionale, dando così all'utente finale la possibilità di scegliere il punto di vista preferito. Il potenziale grado di realismo 3D si affina con la densità di telecamere presenti attorno la scena, dal momento che diversi punti di vista possono essere visualizzati, permettendo un browsing interattivo ed arbitrario. Pertanto, la quantità di dati da trasmettere/memorizzare richiede l'utilizzo di efficienti schemi di compressione.

Il punto di partenza della presente tesi è rappresentato dallo schema di codifica per segnali *video+depth* che associa un'unità di identificazione dell'oggetto 3D con una strategia di stima del movimento orientato all'oggetto [3]. Da qui, lo scopo della stessa consiste nel presentare degli algoritmi di codifica video 3D trattando con maggiore enfasi l'impiego di strutture predittive B-gerarchiche, che risultano essere la strada più promettente per lo sviluppo

di codifica video scalabile, che ha importanti implicazioni nel trasporto di informazione su rete. La tesi è organizzata come segue:

- Nel Capitolo 2 viene data una descrizione della struttura del segnale 3D;
- Nel Capitolo 3 si dà una breve descrizione delle caratteristiche dell'ormai ampiamente diffuso standard di codifica video H.264;
- Nel capitolo 4 viene descritto dettagliatamente lo schema di codifica di partenza;
- Nel Capitolo 5 seguiranno i risultati ottenuti dalle diverse tipologie di codifica video 3D adottate;
- Nel Capitolo 6 si traggono le conclusioni del lavoro svolto.

# Capitolo 2

# Il segnale 3D

## 2.1 Stereoscopia

L'idea di stereoscopia è molto antica. Il primo a comprendere i principi della visione tridimensionale fu Euclide nel 208 a.c., osservando come ciascuno dei nostri occhi percepisca un'immagine leggermente differente dall'altro ed è la combinazione delle due immagini a fornire la percezione della terza dimensione. Successivamente nel 1833 Charles Wheatstone dimostrò che, ponendo due disegni leggermente diversi l'uno accanto all'altro e osservandoli attraverso un sistema di specchi e prismi è possibile riprodurre artificialmente l'effetto della visione tridimensionale. Nel giugno 1838, illustrando la visione binoculare alla *Royal Scottish Society of Arts*, propose di denominare l'apparato *stereoscope*, al fine di indicare le sue proprietà di rappresentare figure solide. Da allora, nel corso dei decenni, nel campo televisivo e cinematografico sono state proposte varie tecniche stereoscopiche basate su tecnologie differenti, ognuna delle quali avente lo scopo di far percepire ad un osservatore la stessa configurazione geometrica che si ha durante l'osservazione reale di una scena.

Fra queste tecniche citiamo l'*Anaglifia*, con la quale si intende la realizzazione di un'immagine ottenuta sovrapponendo i due fotogrammi di uno stereogramma colorati con due differenti colori, ad esempio il rosso per l'immagine destra e il verde per l'immagine sinistra. In questo modo osservando

l'immagine tramite lenti di colori analoghi (rosso per l'occhio destro e verde per l'occhio sinistro), si ottiene che l'occhio destro vede la sola immagine destra e l'occhio sinistro la sola immagine sinistra. In campo cinematografico, i primi esperimenti con tale tecnica furono fatti con la coppia giallo-blu, ma in questo caso, oltre ad una variazione del colore, risulta difficile avere immagini prive di effetto *ghost* (fantasma). Successivamente, fu utilizzata la coppia rosso-ciano che combina tutti e tre i primari: l'immagine destinata all'occhio sinistro viene filtrata in modo da contenere solo i contributi verde e blu, mentre quella destinata all'occhio destro viene filtrata per contenere i soli contributi relativi al rosso. La combinazione dell'immagine destra e sinistra viene visualizzata sullo schermo e le lenti colorate degli occhiali operano come filtri, consentendo a ciascun occhio di percepire solo l'immagine ad essa destinata e impedendo la percezione dell'immagine destinata all'altro occhio.

Un'altra tecnica utilizzata in passato sfruttava l'*effetto pulfrich*, che consisteva sostanzialmente in una variazione dell'intensità luminosa, ponendo ad esempio una lente scura di fronte ad uno degli occhi. Così facendo, si ottiene una differente latenza nella percezione dello stimolo, dando origine all'illusione stereoscopica. Un oggetto che si muove su un piano parallelo alla fronte dell'osservatore sembra quindi allontanarsi dal piano, tanto più quanto è elevata la velocità, avvicinandosi o allontanandosi dall'osservatore, in funzione della direzione del movimento.

Un'altra tecnica ancora, basata sulla *luce polarizzata*, consiste nella trasmissione sullo schermo, mediante l'utilizzo di due proiettori, delle due immagini destinate agli occhi degli osservatori: ciascun proiettore è dotato di un filtro che polarizza la luce in modo che i due segnali luminosi riflessi dallo schermo siano polarizzati in modo ortogonale fra loro. Gli osservatori sono dotati di occhiali con lenti polarizzate, in modo da filtrare uno dei due fasci luminosi: ciascun occhio vede uno solo dei due segnali.

Ancora, un'altra tecnica è rappresentata dall'utilizzo degli occhiali con filtri a cristalli liquidi (LCD) alimentati mediante pile e capaci di lavorare in sincronia con il proiettore. Tali occhiali, attrezzati con due lenti/filtri LCD,

uno per occhio, sono sincronizzati con un segnale infrarosso generato dal sistema di proiezione che alternativamente oscura un LCD che agisce come otturatore (*shutter*) .

Tali tecniche (ed altre ancora), per le quali si rimanda a [12] e [13] per una trattazione più approfondita e dettagliata, necessitano tutte dell'utilizzo da parte dell'utente di particolari tipologie di occhiali. Negli ultimi anni però, il crescente sviluppo tecnologico ed i costi via via più competitivi, hanno portato ad un'ampia diffusione degli schermi autostereoscopici, che non necessitano dell'utilizzo di particolari occhiali. Lo scopo degli schermi autostereoscopici consiste nel riprodurre due o più prospettive (viste) della stessa scena, separandole in modo che gli occhi dell'osservatore ne possano vedere due differenti per volta. Se le prospettive sono più di due, si può fare in modo che l'osservatore possa muoversi davanti allo schermo osservando la coppia di viste relativa al punto di osservazione. La separazione delle viste avviene sfruttando la geometria della configurazione di visione. In linea di principio esistono due metodi: gli schermi a *barriera di parallasse* e quelli a *microlenti*, per la cui trattazione si rimanda a [13].

La codifica di immagini stereoscopiche è stata oggetto di ricerca attraverso gli anni. Si imita in sostanza la visione binoculare umana e consiste nella codifica delle scene catturate da due telecamere poste a breve distanza l'un l'altra. Nella visualizzazione stereoscopica vengono utilizzati due insiemi di immagini, una per l'occhio sinistro e l'altra per l'occhio destro dell'osservatore, ognuna delle quali costituente una regolare sequenza bidimensionale. Queste due diverse viste prospettiche della scena 3D vengono "catturate" e riprodotte quasi simultaneamente su un piano immagine comune, così che il cervello umano possa collegare elementi simili da viste differenti, con il conseguente sviluppo della sensazione di profondità (si veda Fig. 2.1). Poichè questo significa raddoppiare l'informazione che deve essere trasmessa, l'obiettivo principale della ricerca, negli anni, è stato quello di sfruttare la correlazione esistente tra le due diverse prospettive, al fine di ridurre la quantità di dati necessaria a caratterizzare un flusso video stereo-

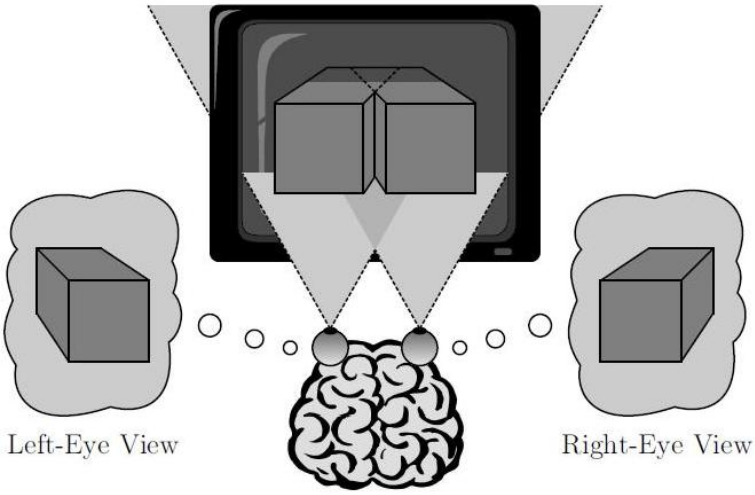


Figura 2.1: Riproduzione della profondità binoculare nei display autostereoscopici 3D-TV.

scopico. Siccome entrambe le telecamere catturano la stessa scena, l'idea di base per la compressione è rappresentata dall'utilizzo delle ridondanze *inter-vista*, le quali possono essere di due tipi: quelle dovute alla somiglianza tra le viste catturate dalle due telecamere e quelle dovute alla similarità tra immagini temporalmente successive nella sequenza video. Un codificatore che sfrutta efficientemente tali tipologie di "somiglianza" è solitamente riferito come *codificatore multiview* [6]. Sfortunatamente, sebbene le due immagini siano molto similari, emergono problemi di *disparità* legate alla non perfetta configurazione geometrica dell'intero sistema. Tale *disparità* deve essere compensata al fine di garantire una corretta resa finale del contenuto 3D.

## 2.2 Multi-View Video (MVV)

Con il termine *Multi-View* ci si riferisce alla possibilità di caratterizzare una scena reale mediante un certo numero di viste (più di due), con lo scopo di garantire all'utilizzatore una totale e reale sensazione di immersività e coinvolgimento all'interno della scena. Il sistema prevede l'utilizzo di $N$

telecamere sincronizzate poste ad una certa distanza l'un l'altra, le quali catturano la stessa scena da differenti punti di vista. Tale tecnica consente la realizzazione di applicazioni quali *Free-Viewpoint Video* (FVV), nella quale la profondità viene percepita mediante l'osservazione delle visuali leggermente differenti della scena, permettendo un'esperienza di visualizzazione più realistica dal momento che l'utente, spostandosi, può decidere di vedere la stessa scena mediante diversi punti di vista ed a differenti angolazioni. Così, ad esempio, un film può essere visto da diverse posizioni, cogliendo molti differenti dettagli dipendenti dalla particolare prospettiva scelta.

L'utilizzo di display autostereoscopici (disponibili ancora solo come prototipi), capaci di riprodurre più visuali, permette la visione *Multi-View* senza l'utilizzo di particolari tipologie di occhiali e quindi si prestano molto bene come futura scelta tecnologica di resa 3D in ambiente domestico. Tali dispositivi sfruttano principi ottici quali la diffrazione, la rifrazione e la riflessione per indirizzare le immagini verso gli occhi dell'utilizzatore.

Se da un lato la trasmissione di un numero $N > 2$ di viste richiede un elevato bit-rate tanto maggiore quanto maggiore è $N$, dall'altro lato l'utilizzo della tecnica *video+depth* permette solo un limitato numero di visuali attorno a quella disponibile, dato che gli artefatti che vengono a crearsi incrementano tanto più quanto è maggiore la distanza tra la vista virtuale e la posizione della telecamera. Tale problema può essere superato usando congiuntamente queste due modalità di rappresentazione 3D, e si parla in questo caso di *Multiview Video plus Depth* (MVD), che fornisce più flussi *video+depth*. MVD richiede tuttavia un processo molto complesso, dato che le viste devono essere generate, devono essere stimate le relative *depth map* e dopodiché i flussi devono essere codificati e trasmessi. In ricezione, dopo la decodifica di tali flussi, deve essere tradotto il contenuto relativo alle *inter-viste* virtuali. Si intuisce quindi, data la grande mole di dati che serve per rappresentare una scena tridimensionale, come l'adozione di efficienti tecniche di codifica e compressione video 3D assuma un ruolo fondamentale allo scopo di garantire la fruizione di tale servizio in ambito domestico.

Fra queste, si cita la tecnica del *3D-Warping* (da *warp*, ossia 'deformare', 'distorcere', 'curvare'), la quale dimostra come l'utilizzo dei dati di profondità permetta di sfruttare la ridondanza inter-vista in modo più efficiente rispetto alla predizione basata su blocchi degli attuali schemi di codifica video *multi-view*. In sostanza, assumendo di avere a disposizione la *depth-map* di una sola vista che quindi funge da vista di riferimento (la chiamiamo $V_r$) tra le $N$ viste disponibili, tale metodologia si propone di proiettare ogni visuale (diversa da quella di riferimento) su $V_r$ seguendo le seguenti fasi per ognuna di essa (preliminarmente per ogni pixel di $V_r$ viene determinato il corrispondente punto nello spazio 3D mediante l'utilizzo della relativa informazione di profondità) [9]:

1. Vengono ordinati tutti i valori di distanza tra i punti 3D (dapprima calcolati) e la telecamera della vista per la quale si vuole effettuare l'operazione di *warping*. Da qui, partendo dalla distanza più piccola, ogni pixel $p_i$ di $V_r$ viene "distorto" nella vista attuale $V_n$ ($n = 1, ..., N, n \neq r$) ottenendo una posizione corrispondente, le cui coordinate vengono poi interpolate arrivando così al pixel $q_i$.

2. Nel caso in cui $q_i$ sia stato già selezionato nelle iterazioni precedenti, ovvero se $q_i \equiv q_j$ e $p_i \not\equiv p_j$, viene conservato il valore precedente. Dato che, in questo caso, l'operazione di *warping* viene effettuata partendo dagli oggetti più vicini ed arrivando a quelli più lontani rispetto a $V_n$, se il pixel $p_i$ viene mappato in un pixel $q_i$ già selezionato in precedenza, significa che il punto 3D corrispondente a $p_i$ giace lungo il raggio ottico congiungente la telecamera $V_n$ ed il punto 3D corrispondente a $p_j$ e quindi $V_n$ non può vederlo.

3. Successivamente, il colore corrispondente al pixel $q_i$ viene approssimato mediante interpolazione bilineare dei 4 pixel più vicini a $q_i$ in $V_n$.

Nel processo di *warping* possono però far comparsa alcuni dei seguenti inconvenienti [10]:

- Più di un pixel della vista di riferimento viene mappato in un singolo pixel della vista estrapolata (Fig. 2.2 a sinistra). Tale problema può essere risolto seguendo l'approccio illustrato precedentemente nei passi *1.* e *2.*, dando cioé precedenza, nell'operazione di *warping*, ai pixel più vicini all'osservatore; così facendo, gli oggetti presenti in primo piano vengono resi correttamente a spese degli oggetti presenti a distanze maggiori.

- Comparsa di "buchi", ovvero la vista estrapolata richiede informazione che però è mancante (non visibile) nella vista di riferimento (Fig. 2.2 al centro). Tale inconveniente è più difficile da trattare e varie soluzioni ad-hoc sono state proposte al fine di rappresentare correttamente tali regioni "scoperte" [9,14].

- L'area proiettata di una superficie incrementa nella vista estrapolata (Fig. 2.2 a destra). Come prima, anche in questo caso compaiono regioni scoperte che vanno trattate mediante tecniche di interpolazione o con l'adozione di soluzioni ad-hoc.

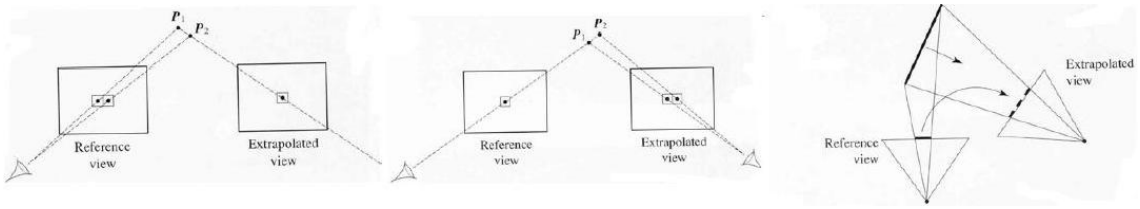


Figura 2.2: Rappresentazione degli inconvenienti che si manifestano durante l'operazione di *warping*.

## 2.3 Depth Image Based Rendering (DIBR)

A differenza della tecnica di visualizzazione 3D basata sulla cattura, trasmissione e raffigurazione di due flussi video separati, uno per l'occhio sinistro e

l'altro per l'occhio destro, una recente tecnica, nota come *Depth Image-Based Rendering* (DIBR)[1] [2], rappresenta il contenuto 3D mediante un flusso video monoscopico standard (che rappresenta la classica informazione di colore della sequenza video) con associata informazione di profondità per-pixel, detta *depth map*, la quale consiste in una matrice di interi a 8-bit (quindi i valori appartengono all'intervallo $0, ..., 255$), dove ogni valore è correlato alla distanza dalla telecamera del punto 3D proiettato sullo stesso pixel (si veda Fig. 2.3). Mentre nell'informazione di colore un frame viene rappresentato come

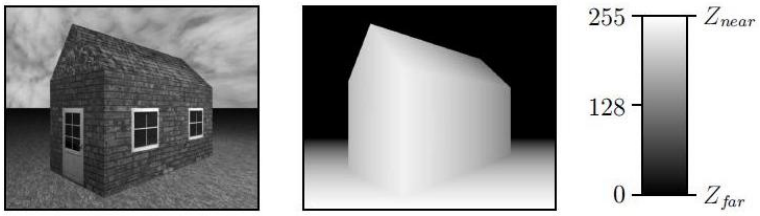


Figura 2.3: Formato DIBR: a sinistra la classica informazione di colore delle sequenze video 2D, a destra l'associata informazione di profondità 8-bit per-pixel (il colore più chiaro indica la vicinanza della scena rispetto l'osservatore, il colore più scuro rappresenta gli oggetti più lontani).

un insieme di tre matrici corrispondenti alle tre componenti RGB o mediante rappresentazione YUV[2] (come nelle tradizionali applicazioni video), la *depth map* usa solo una componente (Y) per memorizzare l'informazione di profondità degli oggetti all'interno della scena. Concettualmente, DIBR utilizza un *depth frame* per generare altre viste virtuali dalla vista di riferimento (ossia quella originale). Pertanto è possibile generare una visualizzazione stereoscopica creando una vista per l'occhio sinistro ed una per l'occhio destro [4].

[1] Il formato DIBR è detto anche formato *video+depth*.

[2] Il formato YUV consta in una componente di luminosità (Y) e due componenti di crominanza (U e V). Siccome l'occhio umano è più sensibile alla luminosità, è possibile assegnare meno banda in fase di trasmissione alle componenti di crominanza. Solitamente la memoria occupata dalle singole componenti U e V è 1/4 di quella riservata alla componente Y.

Questa nuova tecnica di resa tridimensionale può essere riassunta mediante queste due fasi:

1. Come primo passo, i punti relativi all'immagine originale vengono riproiettati nello spazio 3D utilizzando i rispettivi dati di profondità;

2. Successivamente, dati i parametri intrinseci ed estrinseci delle telecamere corrispondenti alle viste da generare, le diverse visuali sono ottenute mediante la proiezione dei punti intermedi tridimensionali sul piano immagine di ciascuna telecamera.

La concatenazione della fase di riproiezione (da 2D a 3D) con la conseguente proiezione (da 3D a 2D) è solitamente riferita in Grafica Computazionale come *3D image warping*, di cui una breve trattazione è stata fornita nella sezione precedente.

Il maggior vantaggio della tecnica DIBR rispetto alla tradizionale rappresentazione del video 3D è rappresentato dal fatto che fornisce un contenuto video 3D di maggior qualità congiuntamente ad una minore richiesta di banda. Ciò è dovuto perché la *depth map* può essere codificata più efficientemente rispetto ai due flussi video relativi alle due visuali se vengono adeguatamente sfruttate le correlazioni e le proprietà delle informazioni di profondità. Per una trattazione approfondita di questo e di altri vantaggi relativi all'utilizzo di tale tecnica si rimanda a [2].

### 2.3.1 Acquisizione della profondità

Esistono molti metodi e diverse tecnologie nell'acquisizione automatica della forma degli oggetti all'interno di una scena nella procedura di generazione della *depth map*. Tali sistemi assumono un ruolo fondamentale nella resa finale di un video 3D, e la loro bontà viene valutata prendendo in considerazione [7]:

- **risoluzione**: il più piccolo cambiamento di profondità che il sensore può rilevare;

- **accuratezza**: differenza tra valore misurato (media di misure ripetute) e valore reale (misura l'errore sistematico);
- **precisione**: variazione statistica (deviazione standard) di misure ripetute di una stessa quantità (misura la dispersione delle misure attorno alla media);
- **velocità**: misure al secondo.

Notevole importanza assumono i sistemi di acquisizione basati su sensori a tempo di volo (TOF, da *Time-of-Flight*), i quali si basano su un principio di funzionamento similare a quello dei RADAR, ovvero emettono un segnale infrarosso modulato sinusoidalmente e stimano la profondità degli oggetti contenuti nella scena misurando la differenza di tempo tra la trasmissione del segnale e la sua ricezione, più precisamente misurando la differenza di fase tra il segnale trasmesso e il segnale riflesso dalla superficie. I sistemi a tempo di volo sono essenzialmente composti da una matrice di trasmettitori sincronizzati che emettono il segnale infrarosso e una matrice di ricevitori integrati in un sensore CCD/CMOS. Tra i maggiori vantaggi che si ottengono nell'utilizzo di sensori *Time-of-Flight* si citano la facilità nel loro utilizzo, i costi contenuti, la minore sensibilità alla luce e la capacità di estrarre informazioni di profondità in tempo reale ad alta frequenza di aggiornamento e quindi risultano particolarmente adatti per il loro impiego nella generazione delle *depth map*.

# Capitolo 3

# Standard H.264/MPEG-4 AVC

H.264/MPEG-4 *Advanced Video Coding* (AVC) è il più recente standard di codifica video. Esso risulta dalla collaborazione tra ITU-T *Video Coding Experts Group* (VCEG) e ISO/IEC *Moving Picture Experts Group* (MPEG) e tale cooperazione è conosciuta come *Joint Video Team* (JVT). Lo standard è riferito come H.264 da ITU-T e come MPEG-4 *Advanced Video Coding* (AVC) da ISO/IEC, ma hanno identico contenuto tecnico.

Lo standard H.264 si propone di migliorare l'anello di codifica costituito da una combinazione di codifica *Intra-frame* con trasformata su blocco e codifica *Inter-frame* con predizione basata sulla compensazione del movimento dai precedenti standard di codifica video MPEG. Questo capitolo si propone di fornire una breve panoramica sulle caratteristiche principali dello standard di codifica H.264 enfatizzandone gli aspetti più rilevanti. Per una trattazione più approfondita si rimanda a [15,16].

## 3.1 Rappresentazione Video

Il modello YUV definisce lo spazio dei colori in termini di una componente di luminosità (luma, Y) e due di crominanza (UV). Dato che il sistema visivo umano risulta essere più sensibile alla componente di luminosità, tale rappresentazione permette di operare un sottocampionamento (e quindi as-

segnare mendo banda in fase di trasmissione) delle componenti di crominanza mantenendo pressoché inalterata la qualità visiva del frame in questione. Lo standard H.264/AVC usa la rappresentazione YUV ed una struttura di campionamento nella quale ognuna delle componenti di crominanza viene sottocampionata di un fattore 4 rispetto la componente di luminosità (sottocampionamento di fattore 2 nella direzione orizzontale e verticale).

Nel seguito si farà ampio uso dei termini *I/P/B-frame*, *GOP*, *Macroblocco* e *Slice*, per i quali viene data ora una breve descrizione.

- **I-frame** (*Intra-coded frame*): come per una convenzionale immagine statica, risulta essere un frame codificato senza riferimento ad altri frame eccetto se stesso ed esso fa da riferimento per i successivi P/B-frame nella sequenza video. Può essere generato dal codificatore per creare un punto di accesso casuale così da permettere al decodificatore, in ricezione, di iniziare correttamente la decodifica da quel particolare frame. In molte applicazioni è d'uso fornire un aggiornamento degli I-frame ogni mezzo secondo, ovvero ogni 15 frame se si suppone un rate di 30 frame/s. In altre applicazioni invece, come per esempio nei sistemi di videoconferenza, la trasmissione degli I-frame avviene di rado.

- **P-frame** (*Predicted frame*): richiede la decodifica di un (o più) frame precedente nella sequenza video (che fa da *reference frame*) per essere decodificato, ed esso può fare da riferimento per frame successivi. Solitamente i P-frame richiedono meno bit rispetto gli Intra dato che vengono codificate solo le differenze tra il frame corrente (P) ed il frame al quale riferisce (P o I) (vedi Fig. 3.1), ma data questa forte dipendenza, un inconveniente è rappresentato dal fatto che gli errori commessi a monte vengono propagati al P-frame corrente.

- **B-frame** (*Bi-predictive frame*): richiede la decodifica di un frame precedente e di un frame seguente (temporalmente) nella sequenza video (fanno entrambi da *reference frame*) per essere decodificato ed ottenere quindi una migliore compressione rispetto i P-frame e gli I-frame a

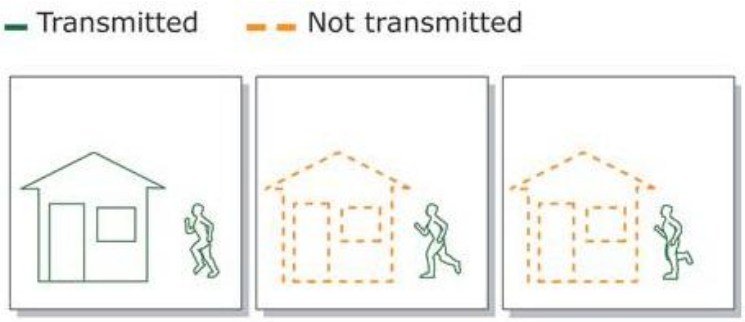


Figura 3.1: Sequenza video composta da 3 frame, nella quale solo la prima immagine (I-frame) viene codificata interamente mentre nelle due immagini successive (P-frame) vengono codificate solo le parti in movimento (in questo caso rappresentate dall'uomo che corre) mediante l'utilizzo dei *motion vector* (vettori di movimento), riducendo così la quantità di informazione che deve essere trasmessa.

scapito però di una maggiore latenza. Ovvero, con riferimento alla Fig. 3.2, nella quale viene raffigurata una sequenza comunemente adottata (IBBPBBPBBIBBP...), il secondo e terzo frame, entrambi B-frame, necessitano del quarto frame (che è un P-frame) per essere decodificati e quindi l'ordine temporale di visualizzazione di tali frame non coincide con l'ordine di codifica.

- **GOP** (*Group of Pictures*): indica una sequenza di immagini in un ordine di visualizzazione contiguo. Può avere diverse conformazioni ma deve contenere almeno un Intra frame.

- **Macroblocco (MB)**: H.264/AVC fa uso di una metodologia di codifica video basata sui blocchi e partiziona i vari frame in elementi più piccoli chiamati *macroblocchi*. Fondamentalmente un macroblocco è un'area rettangolare di dimensione fissa (in un frame) che consta di 16x16 pixel per la componente di luminosità e di 8x8 pixel per ognuna delle componenti di crominanza.

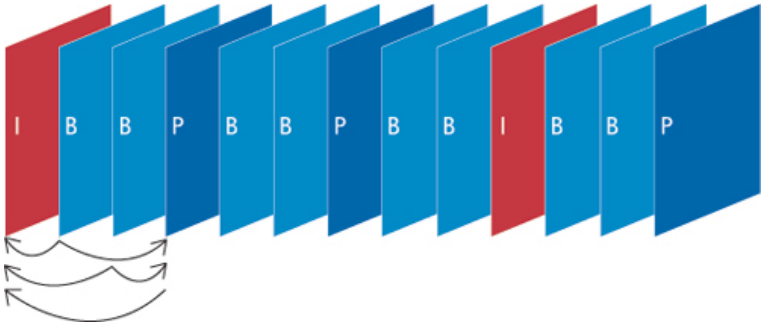


Figura 3.2: Una tipica sequenza video dove i P-frame possono fare riferimento solo ai precedenti I/P-frame, mentre i B-frame fanno riferimento sia ai precedenti che seguenti I/P-frame.

- **Slice**: sono gruppi di macroblocchi (vedi Fig. 3.3). I macroblocchi possono essere distribuiti nelle *slice* seguendo un ordine *raster scan* o mediante un ordine personalizzato (si parla in questo caso di *Flexible Macroblock Ordering*, FMO). Un frame può consistere di una o più *slice*, ognuna delle quali risulta essere indipendente, nel senso che è possibile decodificare i campioni contenuti nella *slice* senza il bisogno di usufruire di dati contenuti in altre *slice*.

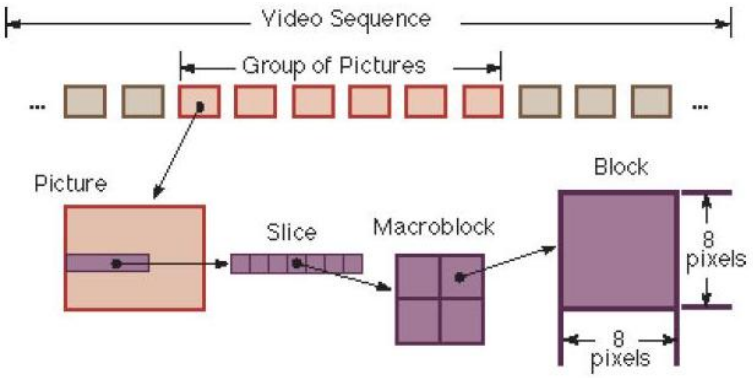


Figura 3.3: Struttura interna di un frame.

Useremo in seguito la simbologia *GgBb* per indicare la struttura del GOP,

dove $g$ denota il numero totale di frame in un GOP e $b$ specifica il numero di B-frame tra successivi I/P-frame, per cui ad esempio, la scrittura G16B3 sta ad indicare la seguente struttura del GOP: IBBBPBBBPBBBPBBB.

Si darà ora una panoramica dei meccanismi di codifica dello standard H.264 che è stato sviluppato sulla base dell'anello di codifica video MPEG e si mostreranno i vantaggi che esso offre rispetto i precedenti standard.

## 3.2 Single-layer (Non-scalable) H.264 Video Coding

Lo standard H.264/AVC si divide in due sottoaree funzionali principali: *Video Coding Layer* (VCL) e *Network Abstraction Layer* (NAL).

- *Video Coding Layer* fornisce le funzionalità di codifica di sorgente, ovvero ha lo scopo di comprimere una data sequenza video nel modo più efficiente possibile garantendo nel contempo un'alta qualità visiva. Al fine di giungere a tali obiettivi, VCL si serve dei tradizionali algoritmi di compressione video, quali ad esempio la compensazione del movimento, codifica entropica e trasformata discreta del coseno.

- *Network Abstraction Layer* si occupa di adattare il *bitstream* di VCL ad ogni possibile configurazione di rete che può presentarsi, garantendo così un'adattabilità universale. Per una trattazione approfondita di NAL si rimanda a [17].

### 3.2.1 Video Coding Layer

Il VCL dello standard H.264/AVC segue l'idea di una codifica video a blocchi, nella quale ogni immagine codificata viene rappresentata da un certo numero di macroblocchi.

L'algoritmo di codifica, inizialmente, decide se predire una data immagine come *Intra-frame* o predirla da immagini contigue, ovvero come *P* o *B-frame* (in questo caso si parla di predizione *Inter-frame*).

Nella predizione Intra, ogni blocco del frame corrente viene predetto producendo un blocco predittivo, il quale viene poi sottratto dall'originale prima di applicare la trasformata discreta del coseno. Per le componenti di luminanza, la predizione viene eseguita su blocchi aventi dimensione 4x4, 8x8 oppure 16x16, mentre per le componenti di crominanza la dimensione è sempre la stessa, partendo da 16x16 e scalato in base al formato di sottocampionamento della crominanza adottato. H.264/AVC definisce 9 diverse direzioni di predizione (vedi Fig. 3.4). La modalità 0 (*vertical*) consiste nel copiare i pixel A, B, C e D nel blocco seguendo una direzione verticale; la modalità 1 (*horizontal*) esegue la stessa operazione della precedente ma in direzione orizzontale (vengono copiati i pixel I, J, K e L); la modalità 2 (*mean*) si serve della media dei pixel posti sul "bordo" per predire il blocco; le modalità 3-8, invece, sono tutte predizioni diagonali.

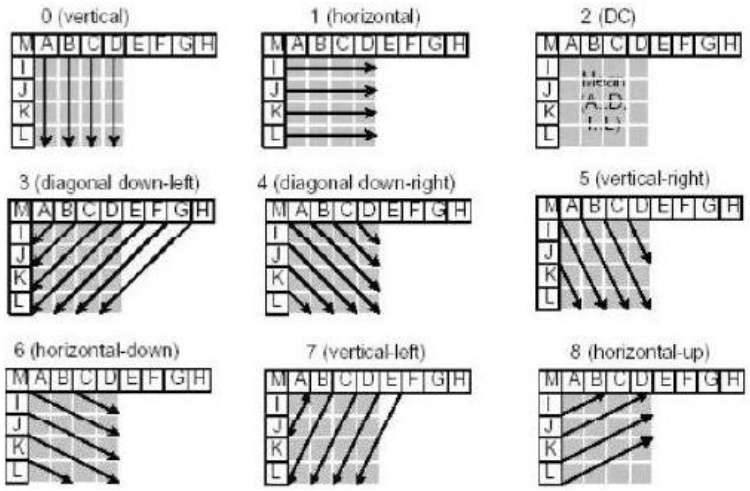


Figura 3.4: Modalità di predizione dell'Intra-blocco.

Quando un frame viene predetto come Intra, il codificatore seleziona una di queste tecniche di predizione (e la segnala al decodificatore) al fine di ottenere la più piccola differenza tra il blocco originale e quello predetto, detta residuo.

La predizione Inter, invece, si basa sul concetto di moto compensazione. Per ogni macroblocco, H.264/AVC applica un algoritmo di compensazione del movimento nel macroblocco stesso o nelle sue partizioni, quali 16x8, 8x16 o 8x8 pixel. Se la dimensione dei blocchi risulta 8x8, è ulteriormente possibile suddividere il blocco in altri blocchi di dimensione 8x4, 4x8 o 4x4 (vedi Fig. 3.5). Per ogni partizione, il codificatore assegna un *motion vector* che

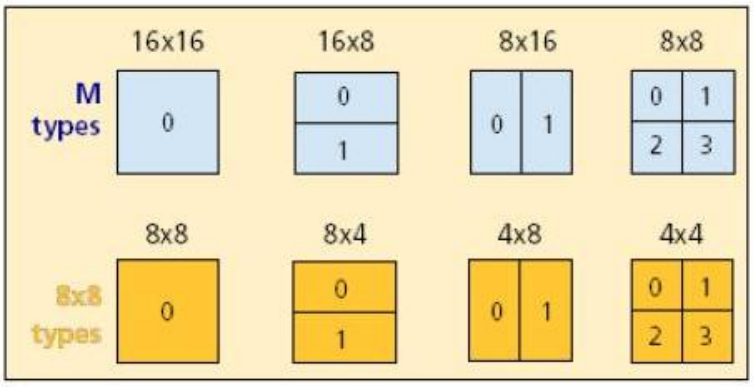


Figura 3.5: Possibili partizioni dei macroblocchi.

viene trasmesso assieme all'informazione relativa a quale tipologia di partizione è stata adottata nella moto compensazione. Scegliere una partizione contenente un basso numero di blocchi comporta la trasmissione di un limitato numero di *motion vector* a scapito però di una ridotta precisione nella stima del movimento e di una maggiore energia sul segnale residuo da codificare. Dall'altro lato, partizionando un dato blocco in numerosi sotto-blocchi, conduce ad una migliore stima del movimento con l'inconveniente però di dover trasmettere un elevato numero di bit, dati i numerosi *motion vector* che vengono a crearsi. H.264/AVC supporta una partizione adattativa della particolare regione dell'immagine da codificare, basata sulla ricerca delle partizioni ottime che minimizzino il numero di *motion vector* e l'energia totale dei residui. Contrariamente agli standard precedenti, la codifica *Inter-frame* viene perfezionata mediante l'utilizzo di $N > 1$ *reference frame* (che vengono memorizzati nel *decoded picture buffer*, DPB) pesati opportunamente, al fine

di ottenere una miglior codifica. Mentre con i precedenti standard di codifica video MPEG la predizione di un blocco in un B-frame veniva effettuata mediante una media tra un blocco di un precedente I/P frame ed un blocco di un successivo I/P frame, H.264/AVC introduce la possibilità di avvalersi di più frame di riferimento, ossia, la predizione di un blocco dell'attuale B-frame può essere effettuata prendendo un blocco di un I/P frame precedente nella sequenza temporale (non solo quello immediatamente precedente) ed un blocco di un I/P frame successivo temporalmente (non solo quello immediatamente successivo), ognuno dei quali opportunamente pesato al fine di ottenere una miglior stima possibile.

Dopo aver eseguito una predizione *Intra-frame* o *Inter-frame*, si eseguono nell'ordine: una trasformata del blocco residuo di predizione (che è dato dalla differenza tra il blocco originale e il blocco predetto) seguita da una quantizzazione dei coefficienti della trasformata; successivamente viene eseguita una scansione a "zig zag" e una codifica *run-length*. Infine, i simboli risultanti vengono ulteriormente compressi mediante codifica entropica, nello specifico H.264 si serve di *context-adaptive binary arithmetic coding* (CABAC) [18], il quale dispone di un alto grado di efficienza seppur abbia lo svantaggio di essere molto costoso dal punto di vista computazionale (alternativamente H.264 fa uso anche di *context-adaptive variable length coding*, CAVLC, che è un perfezionamento dei precedenti meccanismi di codifica a lunghezza variabile). L'intero anello di codifica può essere affinato mediante l'ottimizzazione RD[1], nella quale si cerca di minimizzare congiuntamente il bit-rate della sequenza e la distorsione visiva introdotta. Per esempio, l'abilitazione dell'ottimizzazione RD *lagrangiana* è impegata per trovare i migliori *motion vector* per la compensazione del movimento di un blocco a partire da blocchi in precedenti e seguenti frame di riferimento.

L'anello di codifica è ulteriormente migliorato mediante l'impiego di un

[1]RD, da *rate-distortion*, concerne l'andamento della qualità video (distorsione), che è tipicamente misurata in termini di *Peak Signal to Noise Ratio* (PSNR), come funzione del *bit-rate* della sequenza video compressa. Per un dato valore di qualità video, più è contenuto il *bit-rate* compresso e più risulta efficiente la compressione RD.

*in-loop deblocking filter* che ha il compito di ridurre gli artefatti che vengono a crearsi nel processo di codifica basata su blocchi. Tale filtro (applicato ad ogni macroblocco), che viene utilizzato dopo l'esecuzione della trasformata inversa sia al codificatore che al decodificatore, offre principalmente due vantaggi:

1. I margini del blocco vengono attenuati, migliorando la qualità visiva, in particolare con l'utilizzo di alti parametri di quantizzazione;

2. I macroblocchi filtrati, se usati per una predizione *Inter-frame*, producono un residuo di predizione inferiore.

Figura 3.6: Immagine sottoposta (a destra) e non sottoposta (a sinistra) a filtraggio mediante *in-loop deblocking filter*.

Lo standard H.264/AVC definisce inoltre diversi profili di codifica, fra i quali: *baseline profile*, usato principalmente in applicazioni di videoconferenza e applicazioni per dispositivi mobili; *main profile*, che include tutti gli strumenti per il raggiungimento di un'alta efficienza RD, e parecchi *high profile* per un'efficiente codifica di video ad alta definizione (HD) [19].

### 3.2.2 Codifica B-predittiva in H.264

Sebbene sia stato inizialmente introdotto al fine di aggiungere funzionalità di codifica scalabile ad H.264/AVC, H.264/SVC dispone di importanti caratteristiche che perfezionano la codifica *single layer*. In questa sottosezione

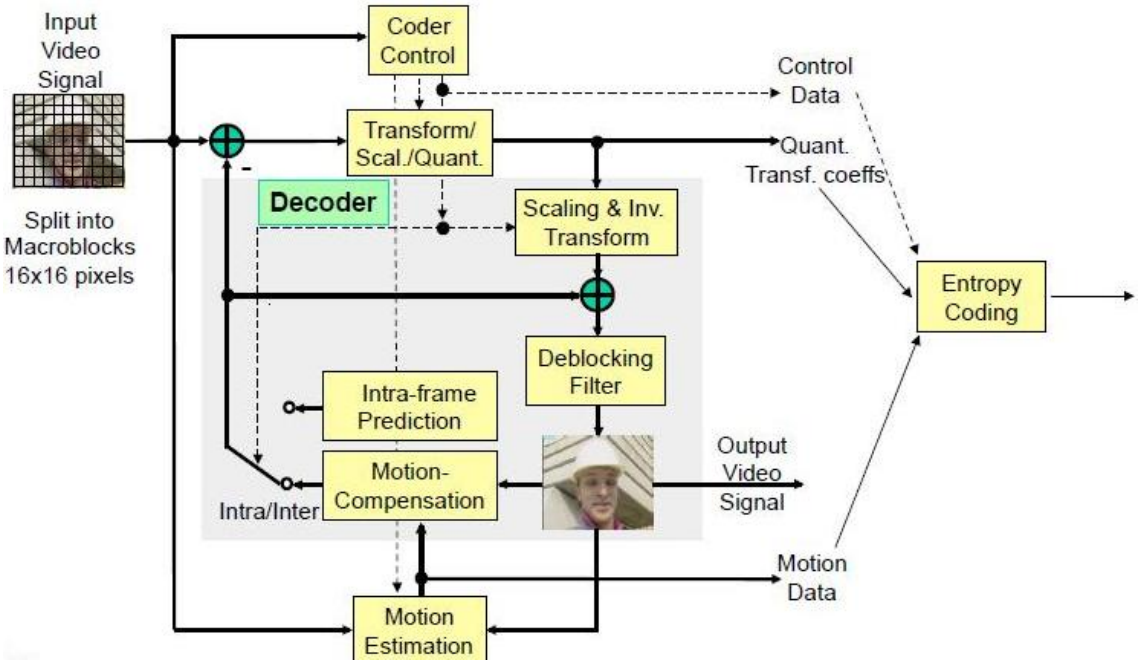


Figura 3.7: Schema generale del codificatore H.264.

viene data una panoramica di tali caratteristiche, rimandando la trattazione della codifica video H.264 *layer scalable* nella successiva sezione.

Gli standard di codifica video precedenti H.264/AVC seguivano strettamente la classica predizione basata sull'utilizzo dei B-frame come raffigurato in Fig. 3.8, ossia, i B-frame non venivano utilizzati per predire altri B-frame. H.264/AVC toglie tale restrizione e permette l'utilizzo dei B-frame come riferimento di altri B-frame nella predizione basata sulla compensazione del moto. In studi successivi [20], si dimostra come tale concetto di utilizzo dei B-frame assuma un ruolo sempre più rilevante nello sviluppo dell'estensione della codifica video scalabile.

H.264 SVC si serve di predizione gerarchica basata su B-frame come raffigurato in Fig. 3.9. In tale illustrazione, vi sono $\beta = 2^{\tau} - 1$ B-frame tra successivi I/P-frame, con $\tau$ numero intero. Si osservi come, in questo caso, il frame $B_4$ venga predetto dai frame $I_0$ e $B_8$. Dunque, continuando nella struttura gerarchica, il frame $B_6$ è predetto dai frame $B_4$ e $B_8$, $B_5$ viene pre-

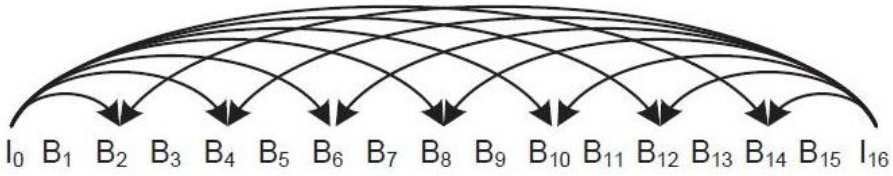


Figura 3.8: Predizione classica basata su B-frame utilizzata nei precedenti standard di codifica video MPEG. La struttura del GOP è G16B15. Sono state omesse le frecce indicanti i riferimenti per i frame con indice dispari per non appesantire la raffigurazione.

detto da $B_4$ e $B_6$ e così via, seguendo man mano l'orientazione delle frecce nella figura.

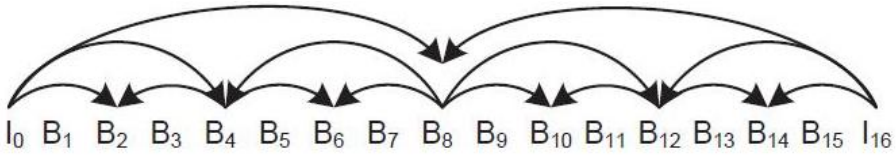


Figura 3.9: Predizione gerarchica basata su B-frame utilizzata in H.264 SVC. La struttura del GOP è G16B15. Sono state omesse le frecce indicanti i riferimenti per i frame con indice dispari per non appesantire la raffigurazione.

Come conseguenza di tale predizione gerarchica, i B-frame sono assegnati a $\tau = \log_2(\beta + 1)$ *temporal enhancement layer* che sono utilizzati per la scalabilità temporale. Nell'esempio di Fig. 3.9 in cui vengono rappresentati $\beta = 15$ B-frame tra successivi Intra, gli I-frame appartengono al *temporal layer* di base $T = 0$, il frame $B_8$ appartiene al primo *temporal enhancement layer* $T = 1$, i frame $B_4$ e $B_{12}$ al secondo *temporal enhancement layer* $T = 2$, i frame $B_2$, $B_6$, $B_{10}$ e $B_{14}$ al terzo *temporal enhancement layer* $T = 3$, e i rimanenti B-frame appartengono al quarto (il più alto) *temporal enhancement layer* $T = \tau = 4$.

L'efficienza RD di suddetta predizione gerarchica dipende fortemente dal

parametro di quantizzazione QP[2] usato per quantizzare i coefficienti risultanti dalla trasformata del blocco residuo di predizione. L'intuizione di base risiede nel fatto che un B-frame dovrebbe essere codificato in modo migliore ogniqualvolta vi siano successive predizioni che dipendano dal B-frame in questione. In base a tale idea, H.264 SVC introduce quantizzatori a cascata per la predizione gerarchica. Mediante l'utilizzo di tali quantizzatori, il codificatore incrementa il parametro di quantizzazione da un valore base (corrispondente ai frame appartenenti al più basso *temporal enhancement layer*) ad un valore più alto corrispondente ai frame appartenenti agli ultimi livelli temporali.

Si noti come, confrontando le metodologie di codifica Bi-predittiva raffigurate in Fig. 3.8 ed in Fig. 3.9, si riscontri una differenza fondamentale rappresentata dal diverso ordine con cui i frame vengono codificati. Generalmente, prima che un dato frame $n$ possa essere codificato, devono essere codificati tutti i frame che fanno da riferimento al frame $n$. Questo implica, per la classica metodologia Bi-predittiva illustrata in Fig. 3.8, che i frame vengono codificati con un ordine dato da: $I_0$, $I_{16}$, $B_1$, $B_2$, ... (tale ordine di codifica è tipicamente preferito dato che fornisce il minor ritardo di decodifica al ricevitore). Diversamente, con l'utilizzo di una struttura predittiva B-gerarchica (Fig. 3.9), l'ordine di codifica risulta dato da (seguendo l'ordine dei *temporal enhancement layer*): $I_0$, $I_{16}$, $B_8$, $B_4$, $B_{12}$, $B_2$, $B_6$, ... (se si vuole una minor latenza si adotta l'ordine: $I_0$, $I_{16}$, $B_8$, $B_4$, $B_2$, $B_1$, $B_3$ ...) e risulta quindi avere un ritardo maggiore in confronto allo schema classico di Fig 3.8.

## 3.3 H.264 SVC Video Coding

L'interesse verso la codifica video scalabile, la quale permette l'adattamento in tempo reale delle caratteristiche del *bit stream* ad esigenze applicative

[2]Nel codificatore H.264 la quantizzazione è controllata dal parametro di quantizzazione QP, che assume valori nel range 0,...,51. Il passo di quantizzazione è legato a tale parametro, infatti il passo raddoppia per ogni incremento di 6 di QP.

quali ad esempio la velocità di trasmissione o le caratteristiche del dispositivo o la qualità desiderata, origina dalla coesistenza in rete di dispositivi ed infrastrutture di trasmissione eterogenee. Al giorno d'oggi la codifica video è impiegata in un'ampia gamma di applicazioni che vanno dalla videotelefonia alla video-conferenza su dispositivi mobili, ad Internet, e molte altre. La trasmissione video in tali sistemi è poi soggetta a condizioni trasmissive variabili. Di qui nasce la necessità di affrontare tali problematiche mediante l'utilizzo di tecniche di scalabiltà video, le quali hanno già avuto comparsa nei precedenti standard MPEG-2, H.263 e MPEG-4 Visual. Tuttavia, la realizzazione di scalabiltà spaziale e scalabiltà in qualità in tali standard, comportava un considerevole incremento nella complessità del decodificatore ed una significativa riduzione nell'efficienza di codifica rispetto ai corrispondenti profili non scalabili e quindi, tali inconvenienti, ne hanno limitato il successo.

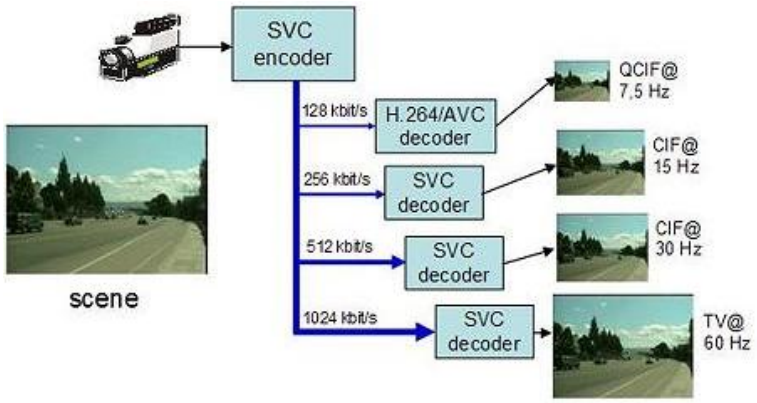


Figura 3.10: Principio di codifica video scalabile.

Al di là del supporto necessario di tutte le tipologie comuni di scalabilità, i più importanti criteri di progetto per uno standard di codifica video scalabile sono rappresentati dall'efficienza di codifica e dalla complessità. A questo proposito, l'estensione SVC di H.264/AVC si propone di fornire scalabilità a livello di bit-stream con un moderato incremento della complessità del decodificatore relativa al caso *single-layer* di H.264/AVC, ereditandone tutti

gli strumenti di base e le metodologie di codifica, ed impiegando le nuove tecniche predittive solo se necessario, al fine di sostenere efficacemente le tipologie richieste di scalabilità.

H.264 SVC fornisce scalabilità temporale, scalabilità spaziale e scalabilità in qualità (SNR). Una codifica scalabile (*layer-scalable*) consiste di un livello base (*base layer*) ed uno o più *enhancement layer*, individuati mediante identificatori di livello crescenti.

## Tipologie di scalabilità

Tipicamente, un *bit stream* video viene detto scalabile ogniqualvolta sia possibile rimuovere parti di tale flusso ottenendo un sotto-flusso, a sua volta decodificabile, rappresentante il contenuto sorgente con una qualità inferiore rispetto a quella ottenibile dall'originale *bit stream*. I flussi video che non prevedono tale proprietà vengono detti *single-layer bit stream*. Le comuni tipologie di scalabilità sono rappresentate da scalabilità temporale, spaziale ed in qualità.

Le scalabilità spaziale e temporale descrivono situazioni nelle quali il contenuto sorgente viene rappresentato da sottoinsiemi del *bit stream*, i quali permettono di ricostruire la sequenza codificata ad una ridotta risoluzione (qualità spaziale) o ad un ridotto *frame rate* (qualità temporale), rispettivamente. Per quanto concerne la scalabilità in qualità, il sotto-flusso risultante fornisce la stessa risoluzione spazio-temporale di quella ottenibile dal *bit stream* originale a scapito però di una minor qualità visiva in termini di *peak-signal-to-noise-ratio* (PSNR)[3]. Queste tre diverse tipologie di scalabilità possono anche essere combinate, così che si possa dar luogo ad un gran numero di rappresentazioni con differenti risoluzioni spazio-temporali e con numerosi livelli di qualità a disposizione.

La realizzazione di un'efficiente codifica video scalabile permette il conseguimento di numerosi vantaggi nell'ambito delle applicazioni. Si consideri, ad esempio, uno scenario nel quale vi sia un sistema di trasmissione video che

[3]La scalabilità in qualità viene anche comunemente riferita come scalabilità SNR.

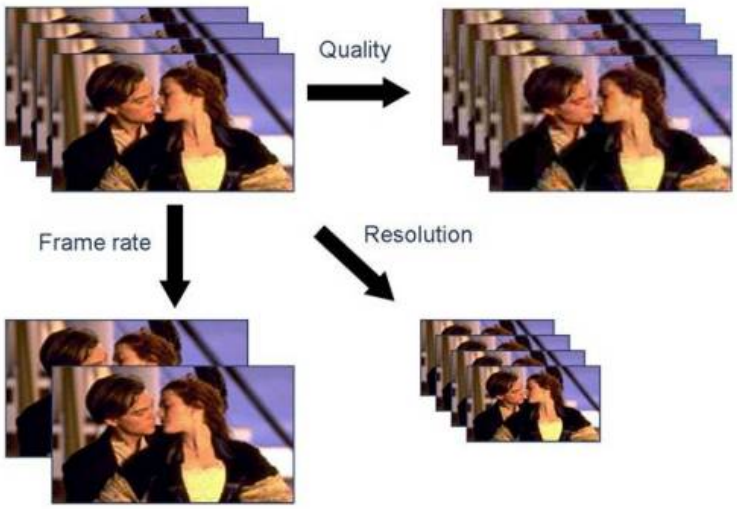


Figura 3.11: Raffigurazione delle tre tipologie di base di codifica video scalabile.

offre i propri servizi operando in una rete eterogenea, necessitando dunque di fornire, nello stesso tempo, diversi *bit stream* dello stesso contenuto sorgente contenenti differenti risoluzioni spazio-temporali e diversi livelli di qualità. Mediante l'utilizzo di SVC, sarà necessario codificare il flusso video originario una sola volta, con la massima qualità e risoluzione, ottenendo i flussi con minor risoluzione eliminando parte dello stream codificato e quindi, per un utilizzatore avente limitate risorse a disposizione (per esempio in termini di risoluzione del display, alto consumo di energia o limitate capacità di elaborazione), sarà necessario decodificare solo una parte dell'intero *bit stream* trasmesso.

Un altro vantaggio che si ottiene mediante l'utilizzo di SVC consiste nel fatto che, solitamente, un flusso scalabile contiene parti con differente importanza in termini di qualità video. Tale peculiarità, congiuntamente all'utilizzo di tecniche quali *Unequal Error Protection* (UEP), risulta particolarmente utile in ogni scenario trasmissivo nel quale vi siano variazioni imprevedibili del *throughput* con un relativo aumento del *packet loss*. Le reti *Media-aware network elements* (MANEs), nelle quali i terminali inviano messaggi di feed-

back con informazioni realtive alle condizioni di banda e stato del canale, possono rimuovere dunque le parti non richieste dal flusso scalabile prima di inoltrarlo (vedi Fig. 3.12). Di conseguenza, la perdita di informazione (dovuta alle congestioni) può essere evitata, migliorando così le prestazioni dell'intero sistema trasmissivo in termini di una maggior robustezza all'errore.

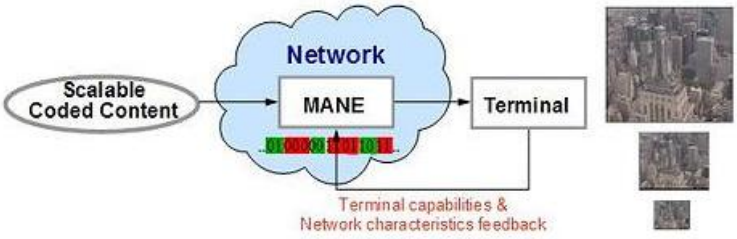


Figura 3.12: Adattamento *on-the-fly* del flusso video scalabile in una rete *media-aware network element* (MANE).

Nei paragrafi successivi, verranno presentate in dettaglio le varie tipologie di scalabilità. Nella Fig. 3.13 viene rappresentato lo schema del codificatore H.264/AVC SVC, nel quale vi è la possibilità di estrarre flussi video con differenti combinazioni di *frame frequency*, risoluzioni spaziali e livelli di qualità da un solo *bit stream* codificato (si parla in questo caso di scalabilità combinata).

### 3.3.1 Scalabilità temporale

Un *bit stream* fornisce scalabilità temporale quando un insieme di *access unit* può essere partizionato in un *temporal base layer* ed in uno o più *temporal enhancement layer* mediante il seguente approccio. Si identifichino i vari livelli temporali mediante un identificatore di livello $T$, il quale vale 0 per il *base layer* e viene incrementato di 1 per ogni successivo livello temporale. Quindi, per ogni $k$ naturale, il *bit stream* che si ottiene rimuovendo tutti

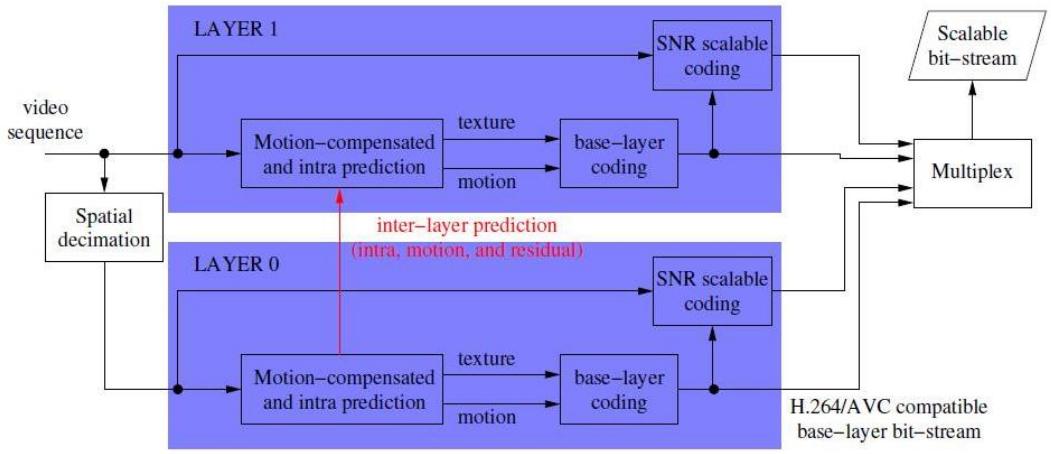


Figura 3.13: Struttura di base del codificatore in H.264/AVC SVC.

i livelli temporali aventi identificatore $T$ maggiore di $k$, forma un altro *bit stream* valido e decodificabile.

Solitamente, nei codificatori ibridi, la scalabilità temporale può venire abilitata limitando la predizione basata sulla compensazione del movimento ai *reference frame* aventi identificatore di livello temporale che è minore o uguale all'identificatore di livello del frame che deve essere predetto. I precedenti standard di codifica video MPEG-1, H.262/MPEG-2 *Video*, H.263 e MPEG-4 *Visual*, supportavano scalabilità temporale in una certa misura. H.264/AVC fornisce invece un'elevata flessibilità per tale tipologia di scalabilità in quanto presenta un *reference picture memory control*, il quale consente la codifica di sequenze di frame con diverse dipendenze, che sono limitate solamente dalla massima dimensione disponibile del *Decoded Picture Buffer* (DPB). Di qui, al fine di supportare scalabilità temporale con un ragionevole numero di livelli temporali, la sola modifica richiesta nella realizzazione di H.264/AVC SVC è da ricercarsi nella segnalazione dei diversi livelli temporali.

La scalabilità temporale può essere efficientemente ottenuta mediante l'utilizzo di strutture B-gerarchiche (vedi Fig. 3.14). I frame dell'*enhancement layer* vengono tipicamente codificati come B-frame, dove le liste 0 e 1 con-

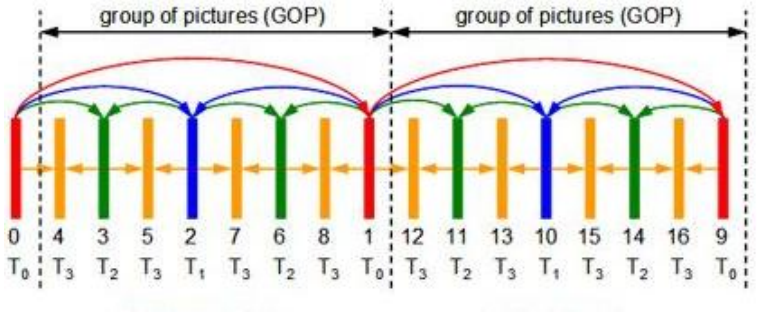


Figura 3.14: Esempio di realizzazione di scalabilità temporale in una struttura predittiva B-gerarchica con GOP G8B7.

tenenti i *reference frame* sono ristrette ai frame temporalmente precedenti e successivi aventi identificatore di livello temporale minore di quello del frame predetto. Ogni insieme di livelli temporali $\{T_0, ..., T_k\}$, dunque, può essere decodificato indipendentemente da tutti i livelli aventi identificatore temporale maggiore di $k$.

Tale concetto di scalabilità temporale si riflette nel concetto di adattamento al *frame frequency*, ovvero, supponendo di avere a disposizione una sequenza video avente una frequenza di 30 frame/sec e considerando la Fig. 3.9, se eliminiamo il livello $T_4$, il quale consiste nei frame $B_1$, $B_3$, $B_5$, ..., si dimezza la frequenza di partenza arrivando ad un valore pari a 15 frame/sec. Eliminando ogni successivo livello ($T_3$, $T_2$, ...) si ottiene di volta in volta un dimezzamento della frequenza finché, giunti al livello base $T_0$, che consiste solo degli I-frame, si arriva alla frequenza di (30/16) frame/sec.

### 3.3.2 Scalabilità spaziale

Per supportare la scalabilità spaziale, SVC segue l'approccio convenzionale della codifica *multi-layer*, la quale viene usata anche negli standard H.262/MPEG-2 Video, H.263 e MPEG-4 Visual. In ognuno dei livelli spaziali viene fatto uso di predizione basata sulla compensazione del movimento e predizione Intra, come per il caso della codifica *single-layer*. In aggiunta a tali strumenti

di codifica di base di H.264/AVC, SVC fornisce metodologie di predizione *inter-layer* (vedi Fig. 3.15), le quali permettono di sfruttare le dipendenze statistiche tra differenti livelli al fine di migliorare l'efficienza di codifica (riducendo il bit-rate) degli *enhancement layer*. In H.262/MPEG-2 Video,

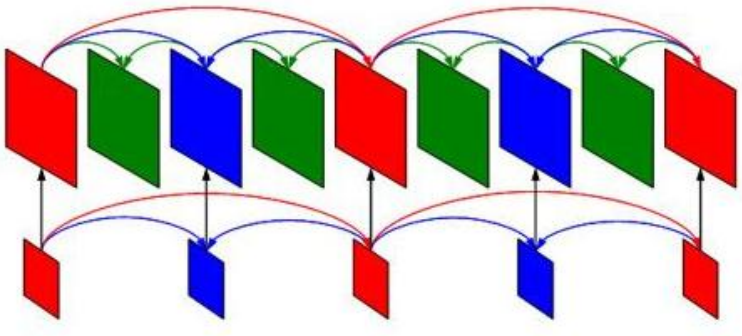

Figura 3.15: Raffigurazione del concetto di predizione *Inter-layer* nella scalabilità spaziale.

H.263 e MPEG-4 Visual, le uniche tecniche supportate nella predizione *inter-layer* consistevano nel predire i campioni dell'*enhancement layer* sovracampionando il *base layer*. Sebbene i campioni ricostruiti rappresentino l'informazione completa del *base layer*, essi non sono necessariamente i dati più idonei che possono essere utilizzati per la predizione *inter-layer*. Solitamente, il predittore *inter-layer* deve competere con il predittore temporale, e, soprattutto per le sequenze statiche e per sequenze aventi alti dettagli spaziali, il segnale di predizione temporale rappresenta generalmente una migliore approssimazione del segnale originale rispetto la ricostruzione ottenuta sovracampionando il *base layer*. Al fine di migliorare l'efficienza nella codifica spaziale, sono state aggiunte in SVC due ulteriori metodologie di predizione *inter-layer*, ovvero la predizione dei macroblocchi e dei relativi vettori di compensazione del moto e la predizione dei residui. Tutti gli strumenti di predizione *inter-layer* possono essere scelti sulla base del macroblocco o del sotto-macroblocco, consentendo al codificatore di selezionare la modalità di codifica che dà la più alta efficienza.

- **Inter-layer Intra prediction**: Tale metodologia di predizione (attivata ponendo ad 1 il parametro *base mode flag*), si propone di fornire un'addizionale modalità di codifica del macroblocco per gli *enhancement layer*, nella quale il segnale di predizione del macroblocco viene completamente stimato dai blocchi co-situati nel *reference layer* senza il bisogno di trasmettere informazione aggiuntiva. Tale segnale di predizione viene ottenuto mediante sovracampionamento del segnale Intra ricostruito dal *reference layer*.

- **Inter-layer motion prediction**: Quando il parametro *base mode flag* è posto ad 1 ed almeno uno fra i blocchi co-situati nel *reference layer* non è codificato come Intra, il macroblocco dell'*enhancement layer* viene predetto con la stessa metodologia adottata nella codifica *single-layer* di H.264/AVC, ma la partizione del macroblocco, ed i vettori di movimento associati ad esso, vengono completamente derivati dai blocchi co-situati nel *reference layer*.

- **Inter-layer residual prediction**: Tale metodologia di predizione prevede una riduzione del bit-rate richiesto per la trasmissione del segnale residuo dei macroblocchi Inter-codificati. Ponendo ad 1 il parametro *residual prediction flag*, il blocco corrispondente del *reference layer* viene interpolato mediante l'utilizzo di un filtro bilineare ed impiegato per predire i residui dell'*enhancement layer*, di modo che vengano codificate solo le differenze, che spesso hanno energia inferiore rispetto l'originale segnale residuo.

Analogamente agli standard H.262/MPEG-2 Video e MPEG-4 Visual, SVC supporta scalabilità spaziale con rapporti di risoluzione arbitrari. La sola restrizione è data dal fatto che né la risoluzione orizzontale né quella verticale possano decrescere da un livello a quello successivo.

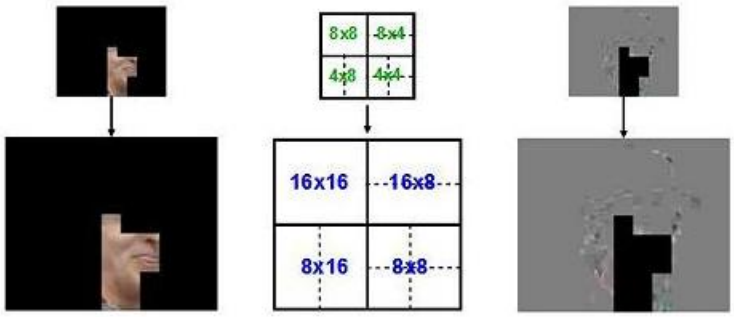


Figura 3.16: Raffigurazione delle tecniche di predizione *Inter-layer*: a sinistra il caso *Inter-layer Intra prediction*, al centro il caso *Inter-layer motion prediction* e a destra *Inter-layer residual prediction*.

### 3.3.3 Scalabilità in qualità

La scalabilità in qualità può essere considerata come un caso speciale della scalabilità spaziale, dove tutti i livelli hanno uguale risoluzione. Tale caso viene riferito come *coarse-grain quality scalable coding* (CGS) ed esso fa uso degli stessi meccanismi di predizione *inter-layer*, ma senza servirsi delle corrispondenti operazioni di sovracampionamento. Nell'utilizzo della predizione *inter-layer*, viene effettuato un affinamento del *texture information* mediante una ri-quantizzazione del segnale residuo nell'*enhancement layer* con un più piccolo passo di quantizzazione rispetto a quello usato per il precedente livello CGS. Inoltre, le modalità *Inter-layer Intra prediction* ed *Inter-layer residual prediction* vengono effettuate direttamente nel dominio trasformato al fine di ridurre la complessità di decodifica. Tuttavia CGS permette di ottenere un limitato numero di bit-rate disponibili (tipicamente il numero di rate sarà identico al numero di livelli). Al fine dunque di rendere più flessibile il *bit stream* ma anche di ottenere altri vantaggi quali una maggior robustezza all'errore e una miglior efficienza di codifica per i tutti i possibili bit-rate ottenibili, SVC dispone di una variazione della modalità CGS, riferita come *medium-grain quality scalability* (MGS), nella quale, a differenza di CGS, si fa uso di una tecnica di segnalazione ad alto livello (che permette

di spostarsi tra differenti livelli MGS in ogni *access unit*) e si usufruisce del concetto di *key picture*, la quale permette una conveniente regolazione del trade-off tra efficienza di codifica dell'*enhancement layer* e *drift error* per le strutture predittive gerarchiche. L'esposizione della tecnica MGS verrà trattata in dettaglio nel prossimo paragrafo.

Il cosiddetto *drift error* descrive l'effetto per il quale, nel loop di moto compensazione, viene perso il sincronismo tra codificatore e decodificatore, ad esempio a causa della perdita di un pacchetto dal *bit stream*. La Fig. 3.17 illustra differenti approcci adottati per conseguire una buona efficienza di codifica e allo stesso tempo un accurato controllo del *drift error*.

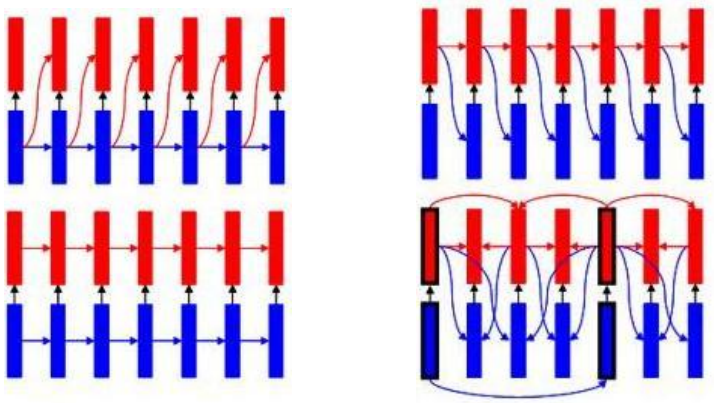

Figura 3.17: Raffigurazione delle diverse metodologie adottate nella trattazione del trade-off "codifica efficiente dell'*enhancement layer*" / "controllo del *drift*" nella codifica scalabile in qualità: in alto a sinistra il caso *Controllo del base layer*, in alto a destra la tecnica *Controllo dell'enhancement layer*, in basso a sinistra *Controllo a doppio loop* e in basso a destra il concetto delle *key picture* nelle strutture predittive gerarchiche (quest'ultimo approccio verrà affrontato dettagliatamente nella sezione 3.4).

- **Controllo del *base layer***: Per la codifica *fine-grain quality scalable* (FGS) in MPEG-4 Visual, la compensazione del movimento viene eseguita solamente sul *base layer*, che agisce da riferimento, e quindi qualsiasi perdita di un pacchetto non ha alcun impatto sul loop di

moto compensazione. L'inconveniente di tale approccio, però, è dato dal fatto che l'efficienza di codifica dell'*enhancement layer* decresce significativamente in confronto alla codifica *single-layer*. Dato che nella predizione basata sulla compensazione del movimento viene utilizzato solamente il *base layer*, la porzione di bit-rate che viene spesa per la codifica di un dato frame in un *enhancement layer* non può essere sfruttata per la codifica dei successivi frame che si servono di suddetto frame come riferimento.

- **Controllo dell'*enhancement layer***: Tale metodologia viene impiegata nella codifica scalabile in qualità nello standard H.262/MPEG-2 Video. In questo caso, a differenza del precedente, il livello avente qualità maggiore viene impiegato nella compensazione del movimento. Ciò permette il raggiungimento di un'elevata efficienza nella codifica dell'*enhancement layer* e garantisce bassa complessità, dato che un singolo *reference frame* necessita di essere memorizzato ad ogni istante temporale. Tuttavia l'effetto *drift* può essere controllato solamente con l'aggiunta di frame di tipo Intra.

- **Controllo a doppio loop**: La Fig. 3.17 in basso a sinistra illustra il meccanismo di utilizzo di due loop di moto compensazione. Tale concetto è analogo al caso della codifica scalabile spazialmente come specificato negli standard H.262/MPEG-2 Video, H.263 e MPEG-4 Visual. Sebbene il *base layer* non è influenzato dalle perdite di pacchetti dell'*enhancement layer*, la qualità di quest'ultimo è molto sensibile all'effetto *drift*.

## 3.4 Sublayer Quality Scalability: H.264 SVC Medium Grain Scalability

Un inconveniente riguardo l'utilizzo di H.264 CGS è rappresentato dal fatto che esso risulta avere scarse prestazioni quando le differenze in qualità

(ed in bit-rate) tra successivi *quality layer* sono molto piccole. H.264 SVC sviluppa un nuovo approccio riguardante la scalabilità in qualità posto tra *coarse grain scalability* (CGS) e *fine grain scalability* (FGS)[4] [5,21], definito *medium grain scalability* (MGS), il quale opera frazionando un dato *quality enhancement layer* in più livelli MGS (fino a 16), con la possibilità di raggiungere un'efficienza RD prossima a quella che si ottiene mediante H.264/AVC [22,23].

Nella codifica video convenzionale, tutti i coefficienti trasformati di un macroblocco vengono sottoposti a scansione a "zig-zag" seguita da codifica *run-length* e codifica entropica. Diversamente, MGS scinde i coefficienti trasformati del macroblocco in più livelli MGS sfruttando la decrescente importanza dei coefficienti a più alta frequenza (ovvero i coefficienti con indice più elevato), ossia, con riferimento alla Fig. 3.18, ad esempio, in un blocco di 4x4 pixel, che consta di 16 coefficienti trasformati, il coefficiente con indice 0 risulta essere il più importante, seguito dai coefficienti con indice 1 e 2, a loro volta seguiti dai coefficienti con indice 3-4-5 e così via. In tale processo, i coefficienti con indice più piccolo costituiscono i più bassi (ed i più importanti) livelli MGS. Formalmente, per un blocco 4x4, sia $\mathbf{W} = [w_1, w_2, ..., w_{16}]$ un vettore di pesi tale che risulti $\sum_{m=1}^{16} w_m = 16$. Il valore $w_m$ indica il numero dei coefficienti trasformati contenuti nel livello $m$. Sia dunque $M$ il numero totale dei livelli MGS, che è uguale al numero dei pesi $w_i$ $(i = 1, ..., 16)$ diversi da zero. Con riferimento alla Fig. 3.18, ad esempio, l'*enhancement layer* viene scisso in $M = 3$ livelli MGS rappresentati dal vettore dei pesi $\mathbf{W} = [3, 3, 10]$, ovvero, $w_1 = 3$, $w_2 = 3$ e $w_3 = 10$, mentre tutti gli altri pesi che non sono specificati vengono posti a zero, ossia $w_4 = w_5 = ... = w_{16} = 0$. Un altro esempio può essere rappresentato dal vettore dei pesi $\mathbf{W} = [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]$, in cui l'*enhancement layer* viene scisso in $M = 16$ livelli MGS, ognuno contenente

[4]Spesso si parla di codifica scalabile a livelli (o a strati) quando il numero dei livelli è ridotto (fino a 3), mentre si parla di scalabilità a "grana fine" (*fine grain scalability*) quando il numero di livelli è elevato. FGS faceva parte di MPEG-4 e inizialmente fu oggetto di ricerca in H.264 SVC.

un solo coefficiente trasformato.

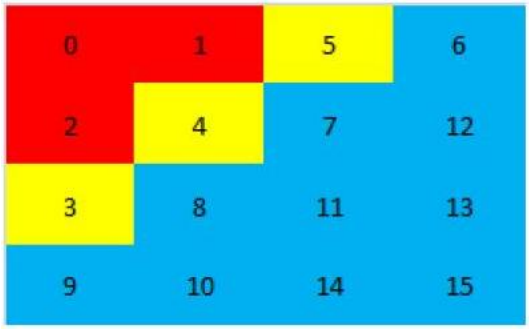


Figura 3.18: Divisione dei coefficienti trasformati di un blocco 4x4 in $M = 3$ livelli MGS con vettore dei pesi $\mathbf{W} = [3, 3, 10]$. I coefficienti con indice 0-1-2 costituiscono il primo livello; i coefficienti con indice 3-4-5 formano il secondo livello, mentre i rimanenti coefficienti costituiscono il terzo livello.

Nella codifica entropica CABAC, ampiamente usata in H.264, i coefficienti della trasformata di un blocco 8x8 vengono divisi in livelli MGS estendendo il modello a "zig-zag" adottato nella trattazione dei blocchi 4x4, ed ogni peso $w_m$ viene moltiplicato per 4. Con riferimento all'esempio di Fig. 3.18, nel quale si ha $\mathbf{W} = [3, 3, 10]$, e considerando un blocco 8x8, il primo livello MGS sarà quindi costituito dai primi 12 coefficienti trasformati, il secondo livello dai successivi 12 coefficienti ed infine il terzo livello sarà composto dai rimanenti 40 coefficienti.

H.264 MGS consente di ottenere un adattamento flessibile del bit-rate (e della qualità video) mediante una variazione del numero dei livelli MGS per ogni frame della sequenza video. Questo elevato livello di flessibilità è consentito mediante un nuovo meccanismo di segnalazione ad alto livello [24]. H.264 raggiunge tale flessibilità ad un costo molto basso in termini di minore efficienza RD attraverso numerose ed innovative tecniche di codifica scalabile. Principalmente, tali metodologie di codifica introducono un nuovo trade-off tra l'efficienza di codifica RD ed il cosiddetto *drift error* che si verifica quando l'*enhancement layer* di un frame viene scartato (riducendone così la

qualità). MPEG-4 FGS si serviva di predizione basata sulla compensazione del movimento solo per il *base layer*, il quale poi faceva da riferimento nella codifica del successivo *enhancement layer*. Sebbene tale approccio evitava la comparsa di *drift error*, conseguiva un'inefficiente codifica RD. All'altro estremo, MPEG-2 si avvaleva dell'utilizzo della totale qualità disponibile fra tutti i livelli, che veniva impiegata come riferimento nel processo di predizione basata sul movimento, ottenendo un'efficiente codifica RD a scapito però di una maggior sensibilità nella comparsa dei *drift error*.

H.264 SVC MGS introduce il nuovo concetto di *key picture*[5] combinando i vantaggi delle tecniche di codifica di MPEG-4 FGS e MPEG-2 sopra citate. Analogamente a MPEG-4 FGS, i *key frame* si servono solamente del livello base di altri *key frame* come riferimento per la predizione basata sulla compensazione del movimento, limitando così la propagazione dei *drift error* ai frame tra due successivi *temporal base layer*. Con riferimento alla Fig. 3.19 ad esempio, i frame $I_0$, $P_4$ e $I_8$ sono *key picture*; il frame $P_4$ viene predetto dal *base layer* del frame $I_0$.

Similmente a MPEG-2, H.264 SVC MGS si serve della più alta rappresentazione di qualità disponibile, data dall'unione del livello base più tutti i fruibili livelli MGS, per un'efficiente predizione della compensazione del movimento del *base layer* e dell'*enhancement layer* dei frame posti tra *key picture*. In Fig. 3.19, la predizione viene effettuata dal più alto livello di qualità disponibile seguendo una struttura B-gerarchica. Per esempio, il livello base del frame $B_2$ viene predetto dalla più alta qualità disponibile proveniente dai frame $I_0$ e $P_4$. Seguendo la struttura gerarchica, la più alta qualità dei frame $B_2$ e $I_0$ viene utilizzata per predire il livello base del frame $B_1$. Quindi, trasmettendo più livelli MGS per il frame $B_2$, può portare ad un miglioramento della qualità (in termini di PSNR) del frame $B_1$, malgrado non siano stati trasmessi addizionali livelli MGS per suddetto frame.

[5] I *key picture* (o *key frame*) sono frame situati nel *temporal base layer*.

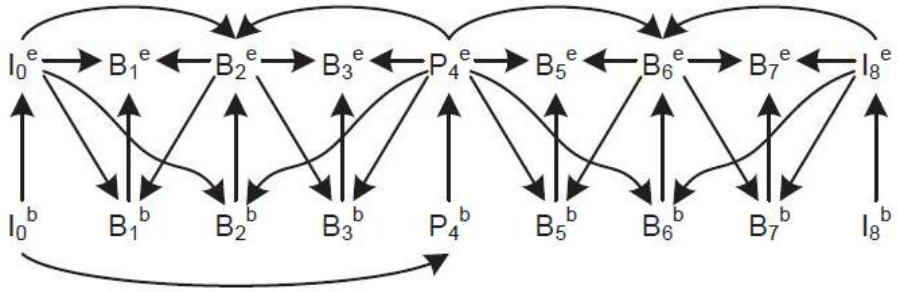


Figura 3.19: Raffigurazione della metodologia di predizione adottata in H.264 SVC MGS per un GOP G8B3 con un *base layer* (indicato con l'apice $b$) ed un *enhancement layer* (indicato con l'apice $e$). I frame $I$ e $P$ sono *key frame*.

## 3.5 Multi-View Video Coding (MVC)

Applicazioni quali *Free Viewpoint Video* (FVV), nella quale si permette all'utilizzatore di scegliere il punto di vista preferito durante l'osservazione di un flusso video al fine di garantire una fruizione interattiva realistica della scena tridimensionale, prevedono l'utilizzo di $N > 1$ telecamere durante la fase di acquisizione della scena, ed ogni flusso risultante necessita di essere trasmesso. Si intuisce quindi come la quantità di informazione che deve essere trasmessa risulti tanto più grande quanto più è elevato il numero di telecamere di cui si fa uso e, pertanto, vi è la necessità di efficienti metodologie di codifica.

L'estensione MVC dello standard H.264/AVC fornisce nuove tecniche e funzionalità al fine di ottenere una codifica efficiente ed una limitata complessità in fase di decodifica per le operazioni *multi-view*. MVC si basa nell'utilizzo dell'*High Profile* di H.264/AVC, fa principalmente uso di strutture predittive B-gerarchiche e si serve di CABAC come modalità di codifica entropica.

MVC estende la tecnica di predizione temporale di AVC con una tecnica di predizione inter-vista, ovvero, facendo riferimento ad un tipico scenario *multi-view* nel quale $N$ telecamere poste a breve distanza l'un l'altra catturano

simultaneamente la stessa scena da differenti angolazioni, vengono sfruttate le dipendenze esistenti tra viste adiacenti. Tali dipendenze, in termini di piccole variazioni che si osservano tra visuali confinanti, permettono all'intero schema di predizione inter-vista di ridurre il numero totale di telecamere mediante un'operazione di interpolazione fra immagini adiacenti al fine di creare una vista "intermedia". Inoltre, MVC si serve dell'utilizzo di *key-picture* con lo scopo di permettere al decodificatore di accedere ad una data porzione del bit-stream per estrarre, ad esempio, il solo flusso corrispondente ad una sola vista, riconducendo così all'H.264/AVC.

In Fig. 3.20, ad esempio, viene rappresentato uno schema nel quale si sfrutta il meccanismo di predizione temporale ed inter-vista, mediante l'utilizzo di 5 telecamere ed una struttura del GOP pari a G8B7. Per la *base view* (Cam 1 in Fig. 3.20), la struttura predittiva è identica al caso della codifica della singola vista, mentre per le rimanenti viste vengono impiegati addizionali *inter-view reference frame* provenienti dalle viste adiacenti (si vedano le frecce rosse in Fig. 3.20). Si noti, tuttavia, che scegliendo come vista di riferimento la vista centrale (Cam 3) al posto di Cam 1, vi siano due viste (e non una sola) che vengono direttamente predette dal frame Intra, risultando così in un possibile miglioramento del guadagno di codifica.

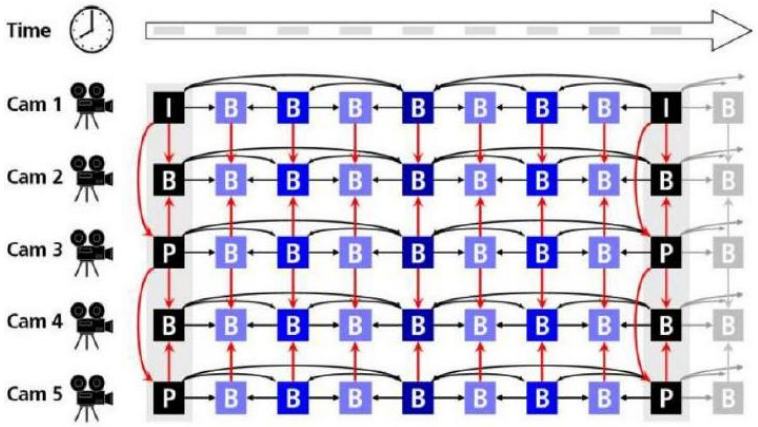


Figura 3.20: Esempio di schema di predizione temporale ed inter-vista in MVC con l'utilizzo di 5 telecamere ed una struttura del GOP G8B7.

In aggiunta, fra gli altri strumenti di cui si serve MVC per il conseguimento di una miglior codifica, si citano qui le tecniche del *Motion Skip* (MS) ed *Illumination Compensation* (IC): la prima delle due sfrutta la similarità derivante dall'indicazione di movimento proveniente dalla vista confinante. Nello specifico, invece di considerare l'informazione di movimento del macroblocco della vista corrente $V_n$, si considera il movimento derivante dal macroblocco della vista confinante che ha lo stesso *picture order count* (POC) della vista corrente. Tale operazione viene eseguita ad entrambe le viste adiacenti la vista corrente $V_n$, ossia a $V_{n-1}$ e $V_{n+1}$, e viene selezionato il miglior blocco, il quale farà poi da riferimento per il blocco corrente. La tecnica IC, invece, mira ad ottenere un'efficiente codifica compensando le variazioni di illuminazione tra frame.

# Capitolo 4

# Schema di base adottato

## 4.1 Introduzione

Un formato ampiamente utilizzato nella rappresentazione di sequenze video 3D è denominato DIBR, il quale associa una *depth map* ad ogni frame di una sequenza video standard (si veda cap. 2).

Sebbene le informazioni di profondità derivanti dalle *depth map* presentino caratteristiche peculiari che si dimostrano essere completamente differenti dalla rappresentazione video standard, le sequenze *depth* e *color* presentano una forte correlazione nel *motion flow*, la quale può essere sfruttata mediante la realizzazione di una codifica di sorgente congiunta [27,28]. In aggiunta, un'elaborazione congiunta dei segnali *color* e *depth* permette di identificare più accuratamente gli oggetti che sono presenti nella scena. Tale abilità permette di raggiungere alti guadagni di compressione, dato che una limitata quantità di dati può caratterizzare una vasta regione (corrispondente ad un singolo oggetto) nella scena, ogni volta che la correlazione spaziale o di movimento presenti le stesse caratteristiche.

La maggior parte dei codificatori video partiziona i frame in blocchi regolari di pixel di varie dimensioni (si è visto ad esempio come lo standard H.264/AVC supporti diverse partizioni da 4x4 a 16x16 pixel), i quali poi vengono sottoposti a stima del movimento e compensazione. Un miglior

risultato può essere ottenuto adattando le dimensioni delle regioni alle caratteristiche ed alla geometria degli oggetti ripresi dalla telecamera. Tale tipologia di partizione può essere ottenuta mediante il processo di segmentazione[1], migliorando significativamente il guadagno di compressione. Difatti, raggruppando insiemi di pixel in regioni di forma arbitraria, permette di derivare l'approssimazione di tali regioni mediante predizione temporale o spaziale, caratterizzata da un insieme comune di parametri, quali ad esempio i vettori di movimento. In aggiunta, risultati recenti riguardo la compressione dei dati di geometria, dimostrano che gli schemi di codifica basati sulla segmentazione sono più efficienti rispetto le tradizionali strategie di codifica dell'immagine [25]. Un motivo di tale miglioria è da ricercarsi nelle caratteristiche dei segnali di geometria (*depth map*), i quali presentano un'alternanza di regioni regolari e bordi netti e quindi, facendo uso di segmenti che hanno la capacità di adattarsi a tale tipologia di segnale, permette il raggiungimento di una più alta efficienza di compressione.

Studi iniziali riguardo la codifica video basata sulla segmentazione risalgono a più di 15 anni fa [29,30] e la possibilità di caratterizzare i singoli oggetti in una scena codificata fu successivamente standardizzata in MPEG-4, ma tale metodologia di codifica non è mai stata largamente adottata e, ad oggi, pochi sistemi di codifica hanno tratto il massimo vantaggio dall'utilizzo di strutture *object-oriented*. Ciò è dovuto principalmente per tre motivi: costo computazionale relativo all'operazione di segmentazione; necessità di segnalare al decodificatore la forma degli oggetti; esigenza di precisione nel processo di segmentazione, al fine di ottenere un'accurata identificazione dell'oggetto.

Tuttavia, la recente ampia diffusione di schermi tridimensionali e la possibilità di ottenere *depth map* in tempo reale (per esempio mediante sensori *Time-of-Flight*), hanno permesso agli sviluppatori di codifica video di

[1]Da qui in avanti, i termini "segmenti", "partizioni" e "superpixel" vengono utilizzati in modo equivalente per indicare le regioni dell'immagine identificate dalla strategia di segmentazione.

manipolare nuovi segnali (che danno informazione di geometria), al fine di ottenere un'accurata partizione della scena ripresa dalla telecamera ad un ragionevole costo computazionale. Difatti, la segmentazione congiunta dei segnali *color* e *depth* consente un'identificazione accurata dei bordi degli oggetti, così che l'*edge distortion* viene limitato e le regioni non si sovrappongono su due oggetti diversi. Quest'ultimo inconveniente è minimizzato forzando il processo di segmentazione a generare un significativo numero di piccoli segmenti (processo di sovrasegmentazione), chiamati *superpixel*. L'utilizzo dei *superpixel* comporta la minimizzazione dell'*edge distortion* perché nella segmentazione si cercano regioni piccole e fortemente correlate, pertanto è molto difficile che un segmento ricopra due oggetti diversi oppure un oggetto e parte dello sfondo.

Il lavoro della seguente tesi consiste nel presentare diverse metodologie di codifica video partendo dal lavoro svolto in [3], nel quale viene presentato un sistema di codifica che sostituisce la tradizionale moto compensazione basata su blocco con una moto compensazione basata sui superpixel, dove i superpixel sono ottenuti mediante una strategia di segmentazione congiunta elaborando i segnali *color* e *depth* (in tale processo, differenti algoritmi [31,32] sono stati testati ed adattati per trattare segnali DIBR). Dapprima, i superpixel vengono generati nei frame precedentemente codificati e, successivamente, vengono propagati al frame corrente sfruttando la correlazione esistente tra frame temporalmente adiacenti. Per ogni segmento nell'immagine DIBR (che comprende sia la componente *color* che la componente *depth*), un'unità di stima del moto ha il compito di trovare un segmento predittore (di uguale dimensione e forma) nel precedente frame il quale minimizzi una data funzione di costo, e caratterizza tale segmento mediante un vettore di movimento (MV). Tale approccio permette di ridurre significativamente il bit-rate dato che l'intero processo di segmentazione può essere riprodotto fedelmente al decodificatore da un insieme minimale di dati trasmessi, ed è possibile fare uso, nella moto compensazione, di blocchi di forma arbitraria, riducendo così la quantità dei vettori di movimento che devono essere

trasmessi.

In questo capitolo verrà fornita la spiegazione dei meccanismi di codifica della soluzione fornita in [3], la quale farà poi da schema di riferimento per gli algoritmi che verranno proposti nel capitolo successivo.

## 4.2 Struttura di codifica/decodifica

Il codificatore video che verrà ora presentato differisce principalmente dai precedenti standard di codifica video per quanto riguarda i processi di stima e compensazione del movimento, mediante i quali viene effettuata una predizione temporale del frame corrente per mezzo della routine di segmentazione. Con riferimento alla Fig. 4.1, nella quale viene illustrato lo schema a blocchi del codificatore, per ogni frame vengono supportate due tipologie di codifica: la codifica *Intra*, che corrisponde alla strategia di codifica *Intra* definita nello standard H.264/AVC, e la codifica *Inter*, il cui processo può essere suddiviso in tre fasi. Nella prima fase, il codificatore produce una maschera di seg-

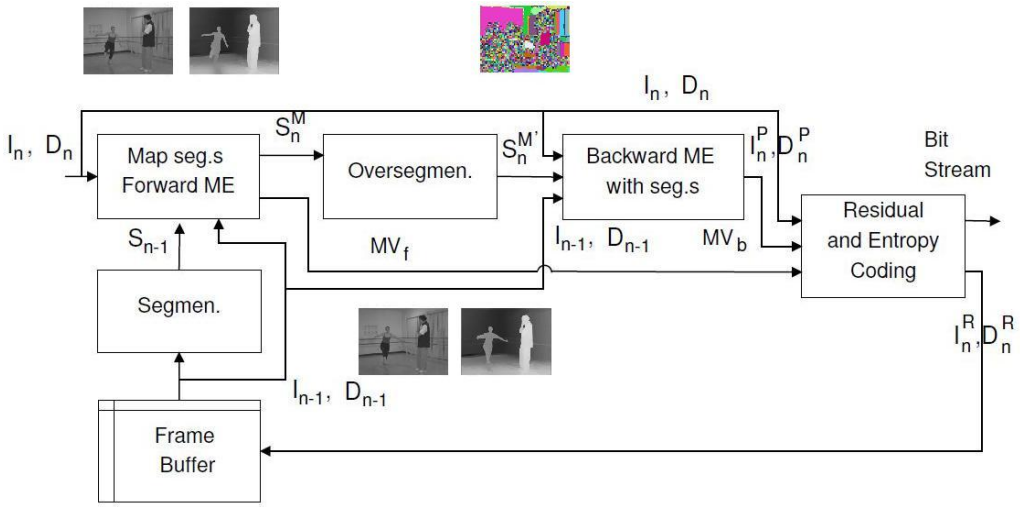


Figura 4.1: Schema del codificatore.

mentazione per il frame corrente, per mezzo della quale gli oggetti presenti nella scena possano essere identificati da insiemi disgiunti di piccoli segmenti

(superpixel). Nel secondo passo, l'unità di stima del movimento (ME) ha il compito di approssimare ogni superpixel con un segmento predittore di forma identica presente nel frame precedentemente decodificato. Nella terza e ultima fase, il segnale residuo di predizione viene elaborato mediante un tradizionale schema di codifica su trasformata. I vettori di movimento (MV) ed i coefficienti della trasformata risultanti vengono successivamente codificati in un flusso binario e trasmessi al decodificatore. In Fig. 4.2 si riporta lo schema del decodificatore. All'inizio di ogni GOP, le componenti *color*

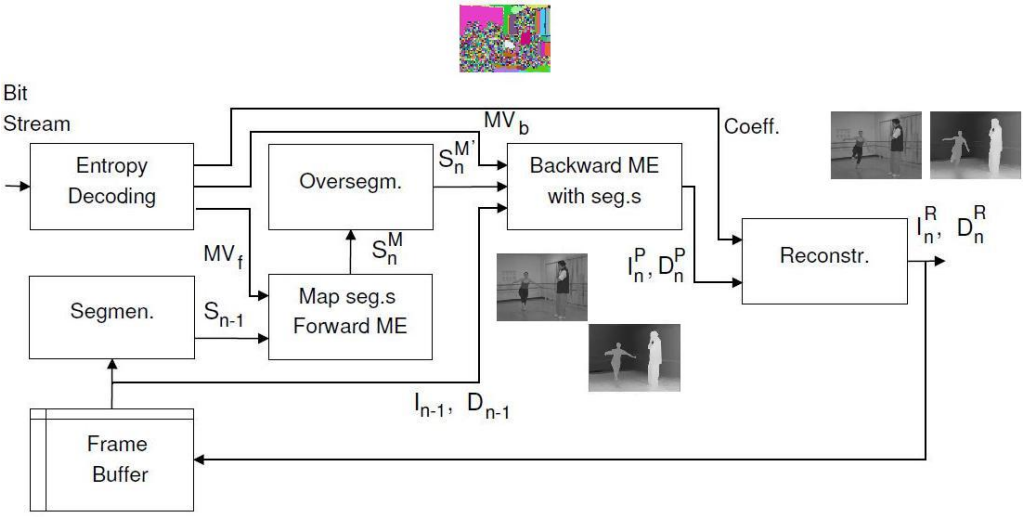


Figura 4.2: Schema del decodificatore.

e *depth* del primo frame vengono codificate per mezzo della strategia Intra definita in H.264/AVC. Le componenti ricostruite vengono poi memorizzate nel *frame buffer* di Fig. 4.1 mentre i frame successivi (fino alla fine del GOP) vengono sottoposti al processo di codifica Inter.

Vengono ora illustrate nello specifico le varie fasi della strategia di codifica Inter, facendo riferimento ad un GOP G15B0, ovvero della forma IP...P.

### 4.2.1 Strategia di segmentazione

Il primo passo nel processo di codifica Inter consiste nel partizionare i frame di input in superpixel, tali che possano poi essere usati dal blocco ME al fine di produrre efficacemente una predizione per il frame corrente. La dimensione dei superpixel risulta avere un'importanza fondamentale nelle prestazioni RD del codificatore, dato che l'efficacia della stima del movimento dipende fortemente dalla precisione con la quale i segmenti vengono adattati agli oggetti presenti nella scena. Se si fa uso di superpixel di piccole dimensioni, si riduce la probabilità che essi vadano a coprire porzioni di due differenti oggetti, incrementando così le prestazioni della predizione temporale. Dall'altro lato però, un elevato numero di superpixel comporta un incremento del bit-rate richiesto per codificare i vettori di movimento, dato che ognuno di essi deve essere segnalato al decodificatore. Dunque, il codificatore necessita di trovare un opportuno compromesso adattando le caratteristiche della sequenza codificata.

Inoltre, la segmentazione deve essere segnalata al decodificatore senza causare un eccessivo incremento nel *bit stream*, quindi deve essere preferita una strategia di elaborazione che permetta di derivare la segmentazione dai dati disponibili, con un limitato aumento del flusso di bit da trasmettere. Come conseguenza, tale scelta comporta un incremento nella complessità del decodificatore, dato che la derivazione della segmentazione del frame deve essere eseguita anche in fase di ricostruzione della sequenza.

L'approccio adottato nel codificatore proposto in [3] può essere suddiviso in tre fasi:

1. si esegue una prima segmentazione dei frame *color* e *depth* dell' immagine precedentemente ricostruita, ottenendo il layout di segmentazione $S_{n-1}$ (vedi Fig. 4.1);

2. la segmentazione $S_{n-1}$ viene propagata al frame temporalmente seguente;

3. nel caso venga richiesta una partizione più affinata, alcuni dei superpixel vengono ulteriormente scissi in superpixel più piccoli.

Vengono ora descritte in dettaglio tali fasi.

### Segmentazione congiunta *color+depth* dei frame precedentemente decodificati

Siano $I_n$ e $D_n$ i frame *color* e *depth* in input all'istante temporale $n$, e siano $\hat{I}_{n-1}$ e $\hat{D}_{n-1}$ i frame ricostruiti relativi all'istante temporale $n-1$, i quali sono memorizzati nel *frame buffer*. I frame precedentemente codificati $\hat{I}_{n-1}$ e $\hat{D}_{n-1}$ vengono partizionati in superpixel $R_{n-1,k}$, $k = 0, ..., N_S - 1$, mediante l'esecuzione di un algoritmo di segmentazione congiunta *color+depth*.

I superpixel $R_{n-1,k}$ vengono rappresentati da insiemi di coordinate $(x, y)$ relative ai pixel contenuti in ogni insieme ($R_{n-1,k} \in \mathbf{Z}^2$). Il numero $N_S$ dei superpixel iniziali dipende dalla risoluzione dell'immagine e dall'algoritmo di segmentazione adottato.

Nell'approccio descritto in [3], sono stati testati differenti algoritmi di segmentazione al fine di trovare la miglior soluzione in termini di accuratezza e complessità richiesta. Come primo passo, i frame in input sono stati partizionati in insiemi disgiunti di superpixel seguendo l'algoritmo *k-means clustering* descritto in [31], per mezzo del quale, l'array $\mathbf{a}(x, y)$ associato al pixel di coordinate $(x, y)$ è dato da:

$$\mathbf{a}(x, y) = [\hat{I}_{n-1}(x, y), \hat{D}_{n-1}(x, y), x, y], \tag{4.1}$$

dove $\hat{I}_{n-1}(x, y)$ e $\hat{D}_{n-1}(x, y)$ rappresentano rispettivamente i valori di colore e profondità nella posizione avente coordinate $(x, y)$.

In una seconda fase, viene considerata la soluzione presentata in [32], opportunamente modificata al fine di supportare segnali DIBR, la quale è basata su una strategia *graph-cut*. In dettaglio, i pesi dei lati nel grafo, che vengono calcolati mediando le differenze assolute tra le componenti RGB di pixel adiacenti dopo l'esecuzione di un filtraggio passa-basso, vengono modificati includendo nella media la differenza assoluta tra i valori di profondità.

**Propagazione della segmentazione**

La correlazione temporale esistente tra frame adiacenti suggerisce che le stesse regioni $R_{n-1,k}$ possano essere trovate nei frame $I_n$ e $D_n$, disposte però in posizioni differenti a causa del moto.

Per tale motivo, la soluzione proposta in [3] si serve di un'unità di stima del movimento in avanti, la quale associa ogni superpixel $R_{n-1,k}$ di $\hat{I}_{n-1}$ ($\hat{D}_{n-1}$) con una regione $R_{n,k}$ di $I_n$ ($D_n$), minimizzando una funzione di costo. Il superpixel $R_{n,k}$ presenta la stessa forma di $R_{n-1,k}$ e, dato $R_{n-1,k}$, può essere identificato mediante un vettore di movimento $\mathbf{v} = [v_x, v_y]$, ossia $R_{n,k} = \mathbf{v} + R_{n-1,k}$. La funzione di costo è data da:

$$C(R_{n-1,k}, R_{n,k}) = \sum_{(x,y)\in R_{n-1,k}} |\hat{I}_{n-1}(x,y) - I_n(x+v_x, y+v_y)| + \sum_{(x,y)\in R_{n-1,k}} |\hat{D}_{n-1}(x,y) - D_n(x+v_x, y+v_y)|, \quad (4.2)$$

la quale combina due *Sum-of-Absolute-Differences* (SAD) per entrambe le componenti *color* e *depth* delle regioni corrispondenti. Tutti i superpixel $R_{n,k}$ possono essere segnalati specificando il vettore $\mathbf{v}$ per ogni regione (nelle Fig. 4.1 e 4.2 la nuova segmentazione è riferita come $S_n^M$). L'insieme di tutti i vettori di movimento prodotti in questo processo di stima del moto in avanti verrà indicato da qui in avanti con $MV_f$. In Fig. 4.3 viene riportato un esempio grafico di tale operazione.

Risultati sperimentali prodotti mediante tale procedura di propagazione della segmentazione dimostrano però che tale tipologia di predizione non permette di ottenere un'efficace moto compensazione dato che alcuni pixel di $I_n$ possono non essere racchiusi in nessuna delle regioni $R_{n,k}$. È possibile però superare questo problema mediante l'impiego di due strategie di "riempimento".

1. Nel caso in cui una piccola regione di pixel "scoperti" (ovvero pixel non inclusi in alcun $R_{n,k}$) si trovi tra due superpixel adiacenti spazialmente $R_{n,k}$ e $R_{n,k+1}$, $R_{n,k}$ e $R_{n,k+1}$ vengono espansi al fine di includerli. Una

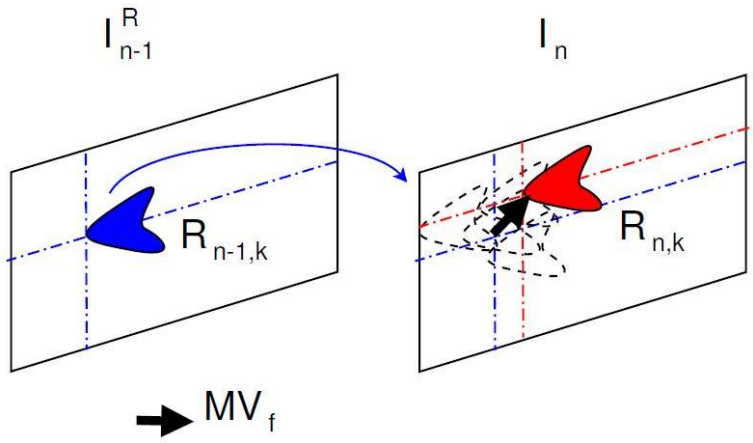


Figura 4.3: Processo di stima del movimento in avanti e generazione dell'insieme $MV_f$.

strategia simile viene adottata nel caso in cui tali pixel siano situati tra i bordi dell'immagine e la regione $R_{n,k}$.

2. Nel caso in cui l'area "scoperta" risulti essere troppo estesa, ovvero il numero dei pixel eccede una data soglia, vengono aggiunte delle regioni quadrate 4x4 $R_{n,k}$ nella maschera di segmentazione, al fine di includere tali pixel.

La maschera finale sarà quindi composta da superpixel di forma arbitraria, derivati mediante l'operazione di segmentazione dell'immagine, e da superpixel quadrati, che hanno lo scopo di includere i pixel "scoperti". Si noti che le operazioni di "allargamento" di $R_{n,k}$ e l'aggiunta di blocchi di dimensione quadrata non richiedono la trasmissione di informazione aggiuntiva, dato che il decodificatore è in grado di riprodurre la maschera finale dal solo insieme $MV_f$.

### Affinamento della segmentazione

L'utilizzo dei superpixel $R_{n,k}$ ostacola la realizzazione di un'efficace compensazione del movimento, dato che la segmentazione viene eseguita nell'

immagine ricostruita dopo averne effettuato la compressione e, pertanto, l'identificazione dei segmenti è influenzata dagli artefatti di codifica. Di conseguenza, un dato superpixel può avere area troppo estesa e dunque includere differenti oggetti. Al fine di risolvere tale problema, la soluzione fornita in [3] si propone di migliorare la prestazione della moto compensazione dividendo le regioni $R_{n,k}$ in segmenti più piccoli.

Ogni qualvolta la SAD si dimostri essere più grande di un dato valore di soglia, i pixel $\mathbf{p} = (x, y)$ contenuti in $R_{n,k}$ vengono divisi in blocchi più piccoli $B^{n,k}_{m,n}$, tale che $B^{n,k}_{m,n}$ è costituito da tutti i pixel $\mathbf{p}$ aventi coordinate che soddisfino la relazione:

$$m = \lfloor x/B_S \rfloor \text{ e } n = \lfloor y/B_S \rfloor. \tag{4.3}$$

Il valore $B_S$ rappresenta la massima dimensione del blocco ed è posto a 4, in base ai test sperimentali effettuati in [3]. Il valore di soglia relativo alla SAD, invece, viene posto pari a $x$ volte la SAD media calcolata sull'errore di ricostruzione dei frame Intra, con $x$ reale dipendente dalla struttura del GoP adottato (verrà trattato nel capitolo successivo).

Nel caso la SAD risulti essere minore del valore di soglia, il superpixel $R_{n,k}$ viene preso così com'è, altrimenti viene scisso nei blocchi $B^{n,k}_{m,n}$. Dato che questa ulteriore partizione è opzionale, il codificatore necessita di segnalarlo al decodificatore mediante l'aggiunta di un bit per ogni regione $R_{n,k}$, segnalando così l'esecuzione o meno della suddivisione del superpixel in blocchi $B_S$ x $B_S$.

In Fig. 4.4 viene riportato un esempio grafico di tale processo, in cui $R_{n,k}$ viene sovrapartizionato, mentre $R_{n,t}$ rimane invariato. In questo modo, si ottiene una nuova partizione, riferita come $S^{M'}_n$ nelle Fig. 4.1 e 4.2, nella quale le regioni sono riferite come $R'_{n,k}$.

### 4.2.2 Stima del movimento

Nella precedente sezione si è fornita la spiegazione di come la stima del movimento in avanti venga usata per trovare una segmentazione dei frame $I_n$ e

$D_n$ da una partizione ricavata nei precedenti frame $\hat{I}_{n-1}$ e $\hat{D}_{n-1}$. Il risultato di tale stima è un insieme di vettori di movimento ($MV_f$), il quale permette di generare una prima approssimazione di $I_n$ ($D_n$) dai pixel di $\hat{I}_{n-1}$ ($\hat{D}_{n-1}$), come mostrato in Fig. 4.3.

Tuttavia, la stima del moto in avanti necessita di essere affinata dato che alcuni segmenti possono essere stati sovrapartizionati, mentre altri segmenti possono essere stati espansi. Di conseguenza, è necessario affinare i vettori di movimento mediante un'unità di stima del moto all'indietro, la quale ha il compito di trovare per ogni regione $R'_{n,k}$ in $S_n^{M'}$ un segmento predittore nel precedente frame che minimizzi la funzione di costo (4.2). Ogni spostamento può essere identificato da un nuovo insieme di vettori di movimento, riferito come $MV_b$ (si veda Fig. 4.4).

Si noti che mentre l'insieme $MV_f$ ha come scopo l'identificazione della partizione in superpixel per $I_n$, l'insieme $MV_b$ punti a ridurre l'energia dell'errore residuo affinando il processo di moto compensazione. L'insieme totale dei segmenti predittori costituisce i frame di predizione $I_n^R$ e $D_n^R$, i quali rispettivamente approssimano $I_n$ e $D_n$ nell'unità di codifica del residuo.

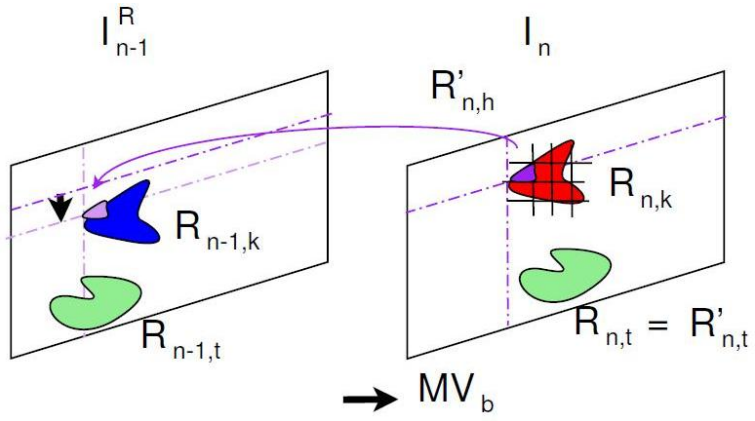


Figura 4.4: Processo di stima del movimento all'indietro e generazione dell'insieme $MV_b$.

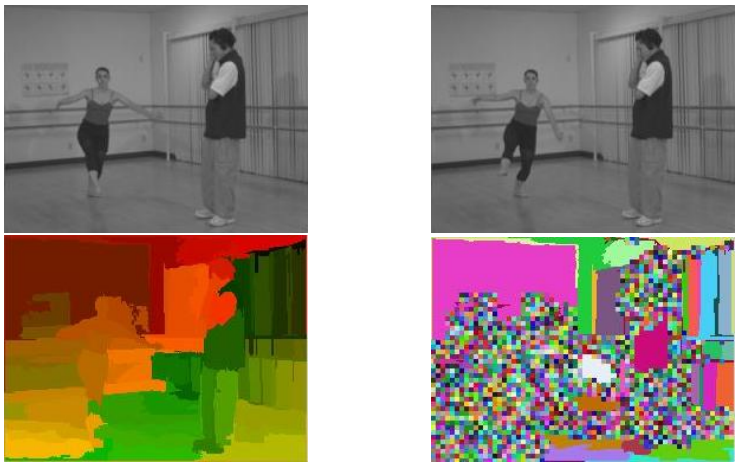

Figura 4.5: Frame di riferimento (istante temporale $n-1$) e frame corrente (istante temporale $n$) con relative maschere di segmentazione. In alto a sinistra frame di riferimento $\hat{I}_{n-1}$; in alto a destra frame corrente $I_n$; in basso a sinistra maschera di segmentazione $S_{n-1}$; in basso a destra maschera di segmentazione $S_n^{M'}$.

### 4.2.3 Codifica del residuo e codifica entropica

Dopo aver approssimato il corrente frame combinando le operazioni di segmentazione e stima del movimento, il segnale residuo di predizione necessita di essere codificato. A tale scopo, la soluzione adottata in [3] fa uso della strategia di codifica definita nello standard H.264/AVC.

Il segnale errore di predizione viene generato sottraendo il frame predetto $I_n^P$ dal frame originale $I_n$, creando così il segnale errore $E_n = I_n - I_n^P$. L'immagine errore $E_n$ viene poi partizionata in blocchi quadrati di dimensione 4x4, i quali vengono poi trasformati in un blocco di coefficienti mediante una 4 DCT definita nello standard H.264. Ogni coefficiente viene successivamente quantizzato in un valore intero mediante quantizzazione uniforme e, in seguito, ogni blocco di coefficienti 4x4 viene convertito in un flusso binario seguendo la strategia di codifica aritmetica CABAC del codificatore H.264/AVC. Dapprima, una strategia *quad-tree based* segnala se ci sono blocchi contenenti coefficienti quantizzati diversi da zero. Per ogni blocco di 16x16 pixel

(ognuno dei quali contiene 4 blocchi 8x8), il codificatore genera un *Coded Block Pattern* (CBP), segnalando quali blocchi 8x8 contengono blocchi 4x4 con coefficienti non nulli. Quindi, per ogni blocco 4x4 all'interno di un blocco 8x8 che contenga informazione residua rilevante, il codificatore entropico specifica una variabile addizionale che segnala se i coefficienti sono o non sono tutti nulli.

Ogni blocco 4x4 con informazione residua non nulla viene poi caratterizzato specificando, dapprima, una mappa delle locazioni dei coefficienti quantizzati non nulli e, successivamente, vengono codificati i valori assoluti dei coefficienti del blocco.

# Capitolo 5

# Risultati sperimentali

## 5.1 Introduzione

Nel presente lavoro di tesi le simulazioni sono state eseguite su diverse sequenze DIBR aventi differenti risoluzioni, scelte per le loro caratteristiche riguardanti soprattutto staticità/dinamicità in termini di velocità di variazione tra frame successivi, e bontà delle mappe di profondità fornite per ognuna di esse. Si noti infatti che la qualità delle *depth map* cambia significativamente per sequenze differenti, dato che per la loro generazione sono stati adottati diversi algoritmi di stima della profondità.

Le sequenze video prese in cosiderazione sono car (ripresa di una macchina che percorre una strada quasi rettilinea con movimenti lineari all'interno della corsia); ballet, breakdancers (inquadratura fissa di persone che eseguono dei balletti mentre lo sfondo rimane pressoché invariato); kendo (ripresa di due atleti che eseguono una tecnica di combattimento di arti marziali con sfondo che cambia di poco); butterfly (sequenza animata caratterizzata da sfondo fisso e poca dinamicità dei soggetti presenti nella scena). Tali sequenze sono state testate con differenti risoluzioni spaziali: 256x192 pixel per ballet, breakdancers e kendo; 432x240 pixel per butterfly e 480x288 pixel per car.

Sulla base dello schema di codifica video presentato nel cap. 4, si andranno ora ad analizzare i risultati ottenuti mediante l'adozione di diverse strut-

ture del GOP ed algoritmi di codifica basati sulla segmentazione, sviluppando con maggior enfasi la trattazione delle strutture predittive B-gerarchiche, le quali risultano essere la strada più promettente per lo sviluppo di codifica video scalabile. Successivamente, si analizzeranno i risultati conseguiti dall'utilizzo di un software in grado di generare, dati i parametri delle telecamere in uno scenario *Multi-view*, tutte le viste disponibili, facendo uso solamente della vista di riferimento e della relativa mappa di profondità (processo di *warping*).

## 5.2 Caratteristiche del GOP e procedura di codifica

L'intero schema di codifica presentato nel cap. 4 è stato descritto per GOP aventi struttura IP...P, cioè una struttura nella quale il frame precedentemente codificato fa da riferimento per la codifica del frame corrente, il quale a sua volta farà da riferimento al frame temporalmente seguente e così via. Risultati sperimentali derivati in [3] mostrano come tale schema di codifica, adottando una struttura del GOP pari a G15B0, migliori mediamente le prestazioni RD rispetto lo standard H.264/AVC di 2 dB (con un ragionevole incremento nella complessità relativa alla stima del movimento e segmentazione), in relazione alla quantità di moto presente nella sequenza ed alla dimensione dei segmenti generati nella partizione.

Come primo passo, si è deciso di testare tale schema di codifica adottando una struttura predittiva B-gerarchica e di verificare se, anche in questo caso, si ottengono vantaggi rispetto l'utilizzo di H.264/AVC.

Dopo un'attenta analisi si è ritenuto opportuno adottare una struttura del GOP pari a G16B15 (si veda Fig. 3.9), dato che, soprattutto per bassi bitrate, fornisce maggiore efficienza di compressione rispetto ad una struttura pari a G8B7 a scapito però di una maggior latenza, mentre si è deciso di scartare a priori una struttura pari a G32B31 data l'eccessiva latenza che produce.

Lo schema di codifica, adottando tale tipologia di GOP, rimane sostanzialmente lo stesso, tuttavia le operazioni riguardanti segmentazione, propagazione della maschera, stima del moto in avanti, sovrasegmentazione e stima del moto all'indietro vengono tutte raddoppiate, dato che in questo caso i frame di riferimento sono due e non più uno. Inoltre, si è scelto un ordine di codifica che seguisse gli identificatori di livello temporale in ordine crescente, ovvero del tipo $I_0$ e $I_{16}$ (appartenenti al *temporal base-layer* $T = 0$), $B_8$ (appartenente al livello $T = 1$), $B_4$ e $B_{12}$ (appartenenti al livello $T = 2$) e così via, fino al più alto livello temporale $T = 4$, consistente di tutti i B-frame di indice dispari di Fig. 3.9. Ciò comporta l'utilizzo di un *Frame Buffer* che contenga fino a 17 frame e non più uno soltanto, dato che, ricevuto e codificato il frame $I_0$, si devono memorizzare tutti i frame seguenti per andare poi a codificare il frame $I_{16}$, situato alla fine del GOP.

Si noti che, in tale schema di codifica, all'uscita del blocco *Backward ME with seg.s* vi sono due predizioni del frame corrente e non una sola (come invece nel caso G15B0): una derivante da un frame di riferimento temporalmente precedente ($I^P_{n,1}$ e $D^P_{n,1}$) e l'altra derivante da un frame di riferimento seguente ($I^P_{n,2}$ e $D^P_{n,2}$). Tali predizioni vengono mediate ottenendo le predizioni $I^P_n$ e $D^P_n$ di Fig. 4.1, le quali vengono poi indirizzate in input al blocco *residual and entropy coding*. L'analisi dei risultati ottenuti mediante tale tipologia di codifica verrà effettuata nella sottosezione successiva.

Come detto in precedenza, una struttura del GOP pari a G16B15 comporta un raddoppio di tutte le operazioni riguardanti la segmentazione e stima di movimento, e ciò si traduce in un raddoppio della complessità computazionale, in termini di tempo di calcolo. Da ciò, ci si è chiesti se fosse possibile "alleggerire" l'intero processo di codifica B-predittiva mantenendo al tempo stesso un'alta efficienza RD.

I processi che richiedono maggior sforzo computazionale, con riferimento allo schema di Fig. 4.1, consistono nelle fasi riguardanti l'effettuazione della segmentazione del frame precedentemente codificato, la stima del moto in avanti (ovvero propagazione della maschera di segmentazione) e la stima del

moto all'indietro. L'attenzione è stata rivolta soprattutto alle prime due fasi, chiedendosi se fosse davvero necessario dover effettuare il processo di segmentazione ad ogni iterazione del procedimento di codifica, ovvero ad ogni B-frame in input.

L'analisi di tale quesito ha condotto alla realizzazione di un algoritmo caratterizzato dalle seguenti iterazioni:

1. Si codificano i frame Intra $I_0$ e $I_{16}$ e si effettua il processo di segmentazione su $I_0$.

2. La maschera di segmentazione risultante dal punto 1. viene propagata direttamente a $I_{16}$ (si veda Fig. 5.1), ottenendo una maschera intermedia $\tilde{S}$.

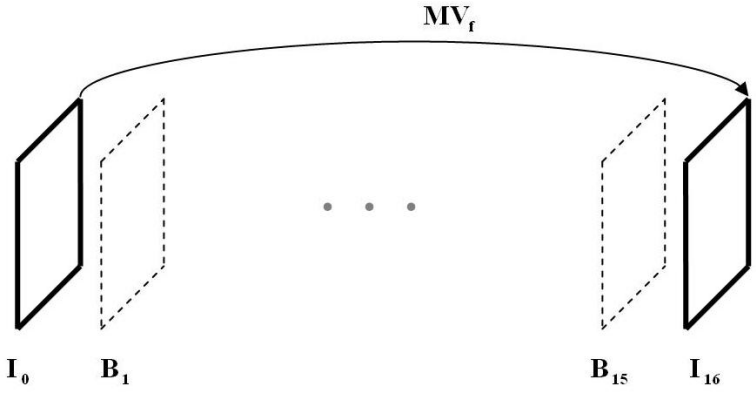


Figura 5.1: Propagazione della segmentazione da $I_0$ a $I_{16}$.

3. Per ogni frame $B_k$ ($k = 1, ..., 15$), si considera ogni vettore di movimento risultante dal processo di propagazione $I_0 \rightarrow I_{16}$ descritto al punto 2. L'ampiezza di ciascun vettore (si ricorda che ognuno di questi indica lo spostamento del superpixel associato ad esso) viene moltiplicata per $k/16$ e viene poi copiata (da $\tilde{S}$) la regione associata ad esso, generando così la maschera di segmentazione $S_k^M$ relativa al frame $B_k$. Questo procedimento permette di ipotizzare che ogni superpixel percorra una

traiettoria da $I_0$ a $I_{16}$ con velocità costante. In Fig. 5.2 viene riportato un esempio di tale processo.

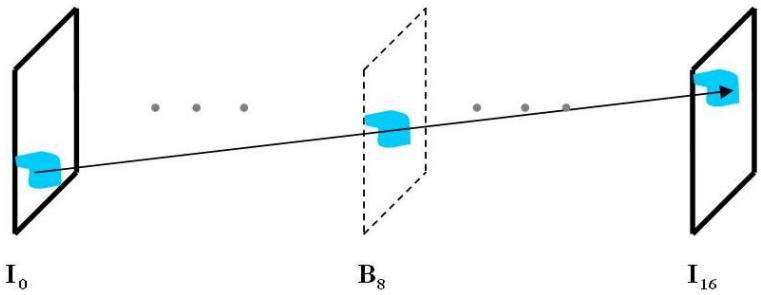


Figura 5.2: Generazione della maschera intermedia.

4. Nel caso due superpixel, nella loro traiettoria da $I_0$ a $I_{16}$, vadano a collidere (si veda Fig. 5.3) nella generazione della segmentazione di un dato frame $B_k$ ($k = 1, ..., 15$), la zona interessata dal conflitto viene lasciata "scoperta", lasciando il compito alle due strategie di "riempimento" descritte nel cap. 4.2 di trattare tale regione.

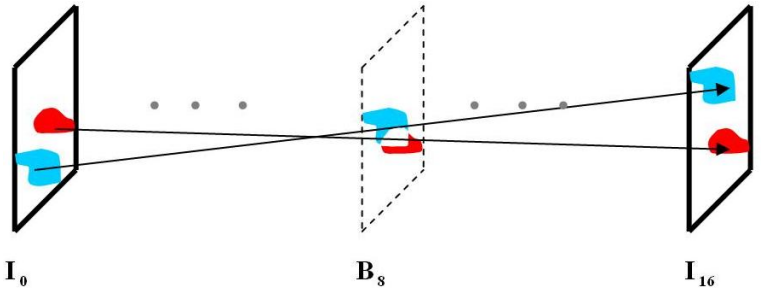


Figura 5.3: Esempio di collisione di due superpixel nella generazione della maschera di segmentazione per $B_8$.

5. Le maschere di segmentazione così ottenute, vengono poi sottoposte al blocco *oversegmentation* di Fig. 4.1 al fine di ottenere un loro affinamento; dopodiché si esegue la fase di stima del moto all'indietro

(considerando entrambi i frame di riferimento a disposizione di ogni B-frame), si mediano le due predizioni risultanti ottenendo un'unica predizione la quale viene poi indirizzata in input al blocco *residual and entropy coding* di Fig. 4.1.

Si noti come, utilizzando tale metodologia di codifica, i processi di segmentazione e propagazione della maschera di segmentazione (stima del moto in avanti) vengano effettuate una sola volta per ogni GOP, diminuendo di molto lo sforzo computazionale rispetto la precedente struttura B-predittiva presa in considerazione. Inoltre, l'insieme dei vettori di movimento $MV_f$ relativo all'unico processo di propagazione della maschera per ogni GOP, non necessita di essere trasmesso, dato che il decodificatore, ricevuti e decodificati i frame $I_0$ e $I_{16}$, è in grado di generare da solo le maschere intermedie di tutti i frame $B_k$ ($k = 1, ..., 15$). Di contro, tale metodologia di codifica può comportare la trasmissione di un maggior numero di vettori di movimento derivanti dal processo di stima del moto all'indietro. Questo perché, adottando tale algoritmo, si suppone che il movimento della sequenza sia lineare e caratterizzato sempre da velocità costante, e ciò ovviamente non vale in generale. Tale processo comporta la realizzazione di maschere molto segmentate all'uscita del blocco *oversegmentation* di Fig. 4.1.

### 5.2.1 Simulazioni

Per ogni sequenza presa in considerazione, si andranno ora ad analizzare i risultati derivanti dall'utilizzo delle tecniche di codifica descritte nella sezione precedente e si confronteranno tali risultati con quelli ottenuti mediante l'utilizzo dello standard H.264/AVC.

Le etichette presenti nelle legende di ciascun grafico hanno il seguente significato:

- **G15B0 superpixel**: simulazione con struttura del GOP G15B0 adottando lo schema di codifica di Fig. 4.1.

- **G16B15 superpixel**: simulazione con struttura del GOP G16B15 adottando lo schema di codifica di Fig. 4.1.

- **G16B15 one segm one forw**: simulazione con struttura del GOP G16B15 adottando la metodologia di codifica relativa all'algoritmo descritto nella sezione precedente (*one segm one forw* sta per *one segmentation and one forward motion estimation*) adottando lo schema di codifica di Fig. 4.1.

- **G15B0 H.264 light**: simulazione con struttura del GOP G15B0 adottando una versione semplificata dello standard H.264/AVC, nella quale ogni frame di riferimento viene segmentato in blocchi quadrati di pixel e sulla base di tale segmentazione viene predetto il frame corrente mediante una stima del moto all'indietro, così che solamente i coefficienti della trasformata e l'insieme dei vettori di movimento $MV_b$ necessitano di essere codificati.

- **G15B0 H.264**: simulazione con struttura del GOP G15B0 adottando lo standard H.264/AVC, abilitandone l'ottimizzazione RD.

- **G16B15 H.264**: simulazione con struttura del GOP G16B15 adottando lo standard H.264/AVC, abilitandone l'ottimizzazione RD e predizione B-gerarchica.

In ogni simulazione si è scelto di far variare il parametro di quantizzazione QP nell'insieme {20,24,28,32,36,40}. Per quanto riguarda le tecniche di codifica basate sullo schema di Fig. 4.1, si è scelto il codificatore aritmetico CABAC per quanto riguarda la modalità di codifica entropica, data la sua versatilità ed alta efficienza di compressione. Il codificatore CABAC viene impiegato anche per la codifica degli insiemi dei vettori di movimento $MV_f$ e $MV_b$. Sulla base dei test e dei risultati ottenuti in [3], si è adottata la strategia di segmentazione "modificata" (al fine di supportare segnali DIBR) dell'approccio descritto in [32], in quanto fornisce prestazioni RD migliori rispetto l'algoritmo *k-means clustering* descritto in [31].

Per ogni sequenza testata verranno forniti sei grafici, riguardanti gli andamenti di PSNR e MSE vs bit-rate (per entrambe le componenti *color* e *depth*), complessità computazionale (in termini di tempo di calcolo) vs qp[1] e numero superpixel generati nella maschera finale vs qp.

PSNR è una misura adottata per valutare la qualità di un'immagine decodificata rispetto all'originale. Tale indice è definito come il rapporto tra la massima potenza di un segnale e la potenza di rumore che può invalidare la fedeltà della sua rappresentazione compressa. Maggiore è il valore del PSNR, maggiore è la "somiglianza" con l'immagine originale.

Il PSNR è più facile da definire attraverso l'errore quadratico medio (MSE, da *Mean Square Error*). Indicando con $I$ il frame originale e con $\hat{I}$ il frame codificato, entrambi di dimensione MxN e contenenti valori interi positivi a 8 bit (quindi il range dei valori è $0, ..., 255$), si definisce l'errore quadratico medio tra i due frame come:

$$MSE(I, \hat{I}) = \frac{1}{MN} \sum_{x=1}^{M} \sum_{y=1}^{N} |I(x, y) - \hat{I}(x, y)|^2 \tag{5.1}$$

Il PSNR($I$, $\hat{I}$) viene quindi definito come:

$$PSNR(I, \hat{I}) = 10 \log \left( \frac{255^2}{MSE(I, \hat{I})} \right) \tag{5.2}$$

Per sequenze video aventi durata pari a $T$ frame, il PSNR totale verrà calcolato come media di tutti i PSNR calcolati sui singoli frame.

Il valore di soglia relativo alla SAD (si veda la formula (4.2)), viene posto pari a 1.6 volte la SAD media calcolata sull'errore di ricostruzione dei frame Intra per quanto riguarda le strutture *G15B0 superpixel* e *G16B15 superpixel*, mentre viene posto pari a 2.5 volte la SAD media calcolata sull'errore di ricostruzione dei frame Intra per quanto riguarda la struttura *G16B15 one segm one forw*. Tali valori di soglia derivano da un compromesso legato al numero di superpixel generati nella maschera finale. Un valore di soglia troppo basso produce un'ottima stima del movimento dato il gran numero di

[1] qp sta per parametro di quantizzazione.

segmenti che vengono a crearsi nella maschera finale, ma comporta la codifica di un'elevata quantità di vettori di movimento. Di contro, una soglia troppo alta produce una maschera finale contenente un basso numero di segmenti (e quindi pochi vettori di movimento da codificare), ma comporta una pessima stima del movimento.

I blocchi relativi alle maschere di segmentazione della struttura *G15B0 H.264 light* sono stati scelti di dimensione 4x4 per sequenze 256x192 e 8x8 per sequenze 480x288 e 432x240.

**Sequenza *ballet***

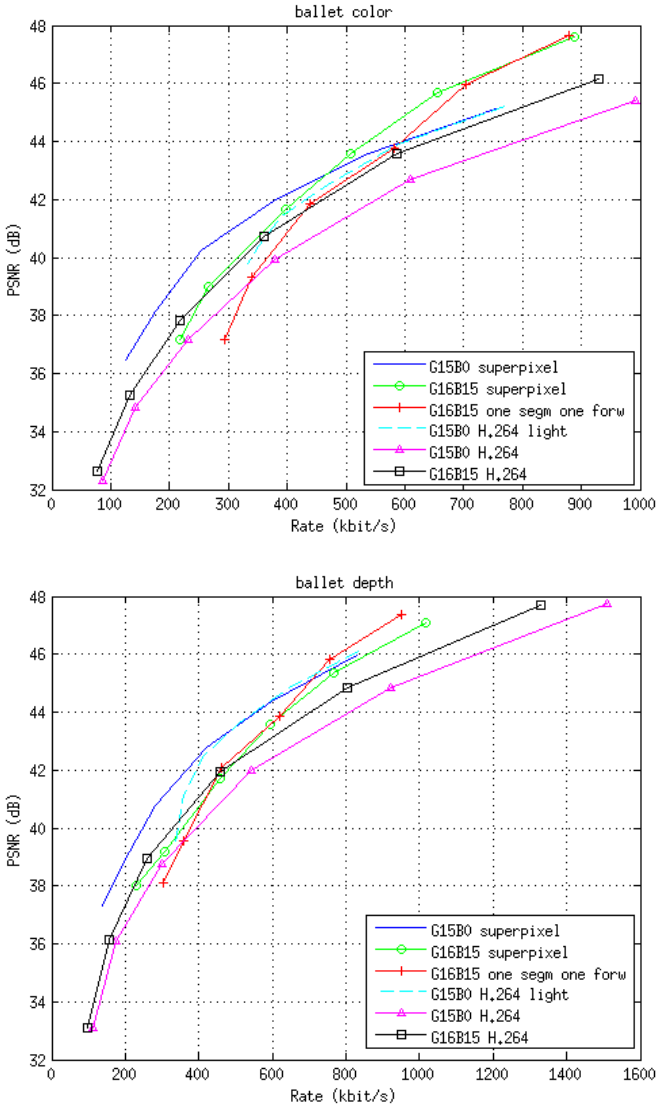


Figura 5.4: PSNR ballet.

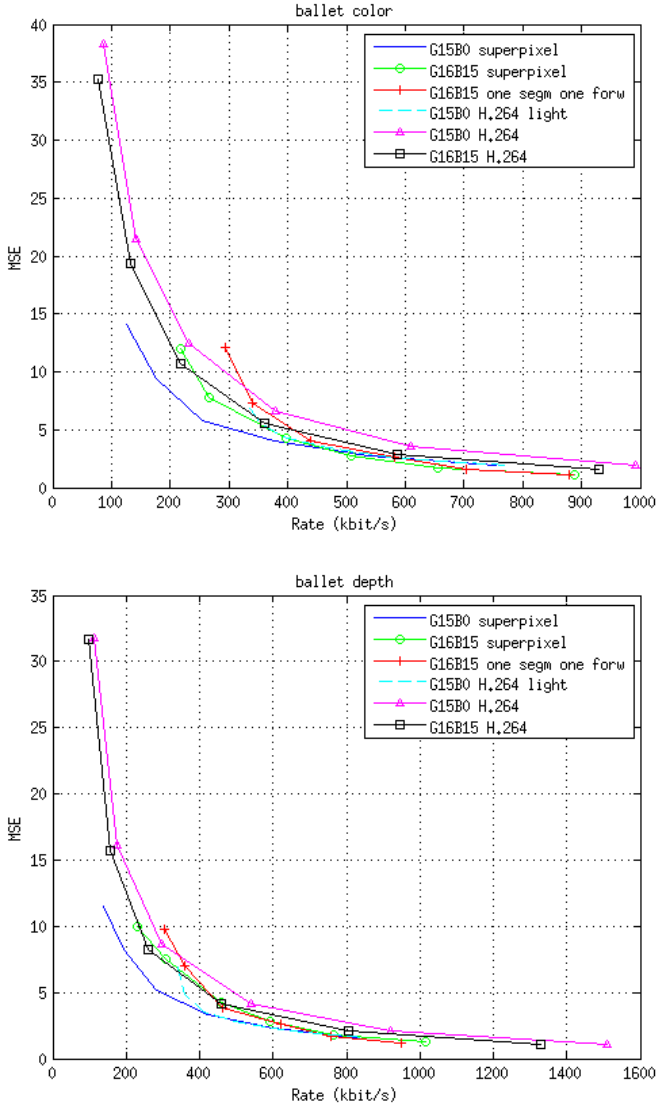


Figura 5.5: MSE ballet.

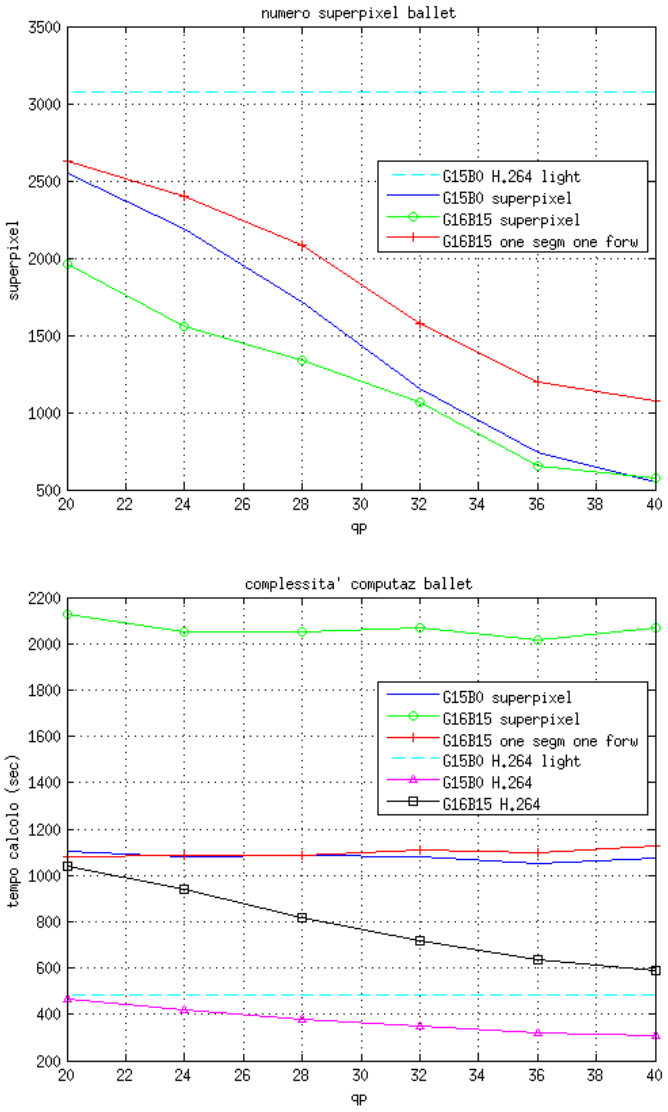


Figura 5.6: Numero superpixel e complessità computazionale ballet.

**Sequenza *breakdancers***

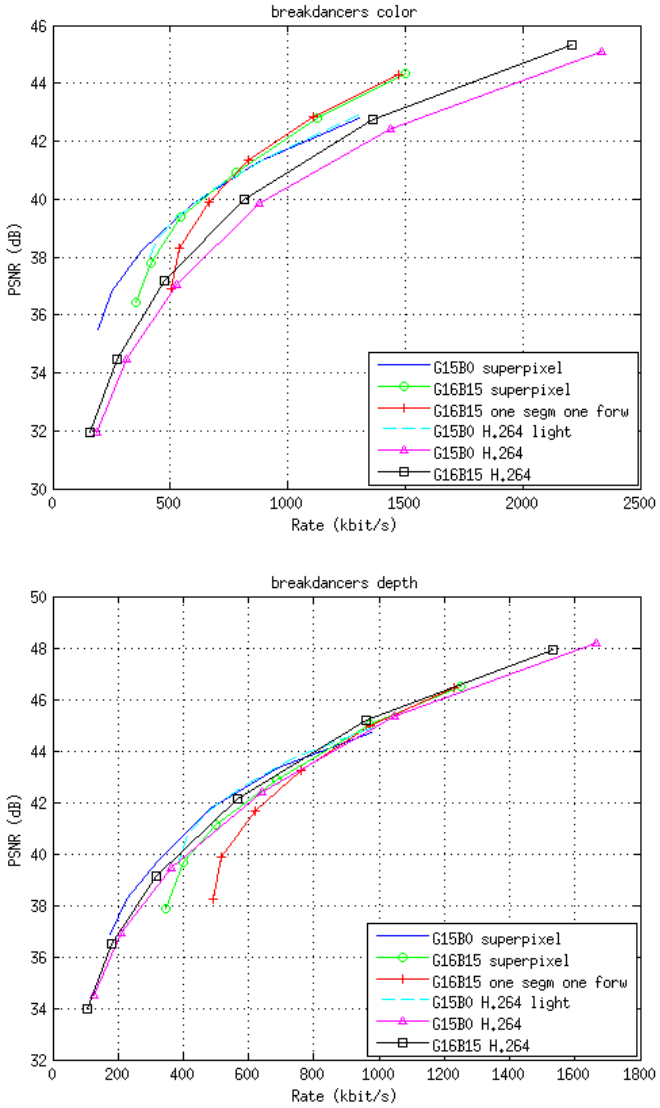


Figura 5.7: PSNR breakdancers.

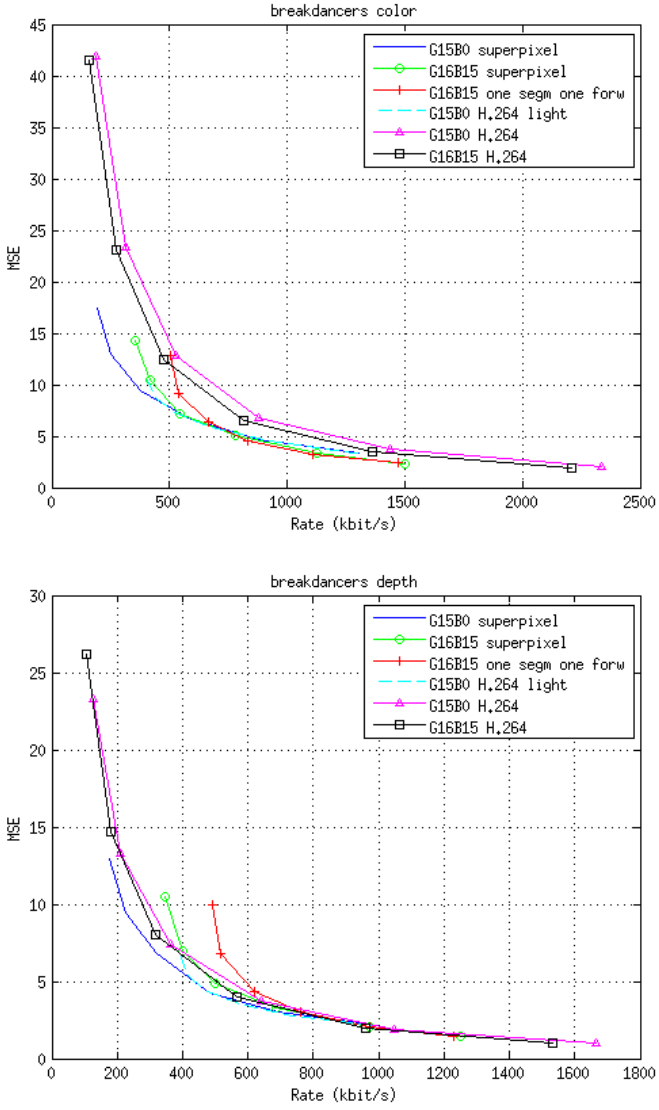


Figura 5.8: MSE breakdancers.

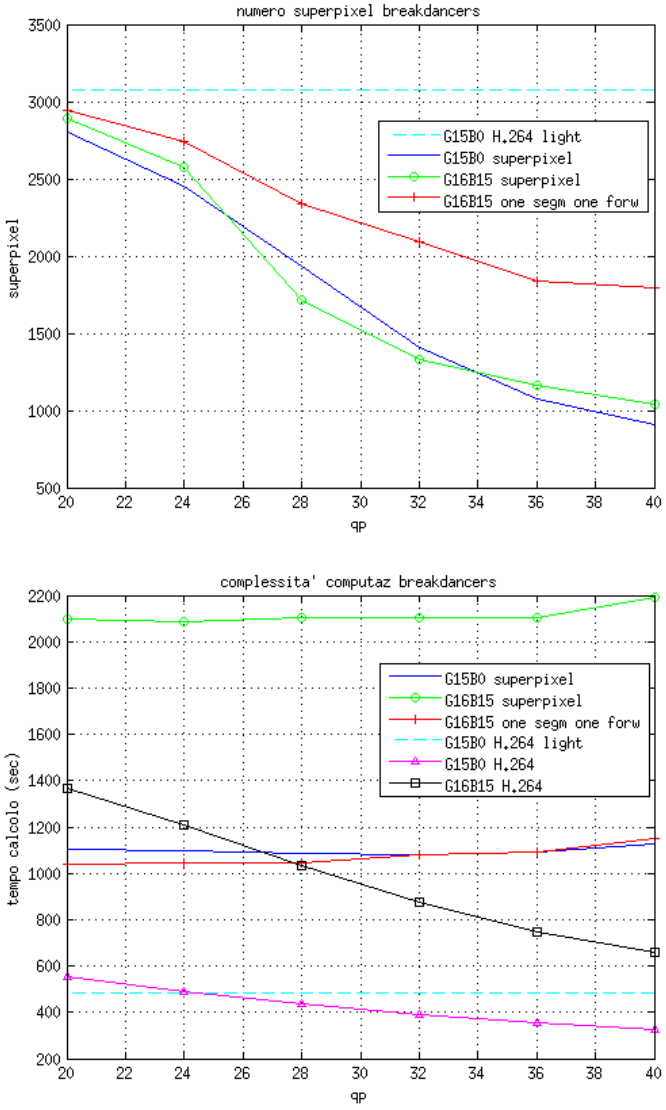


Figura 5.9: Numero superpixel e complessità computazionale breakdancers.

**Sequenza *kendo***

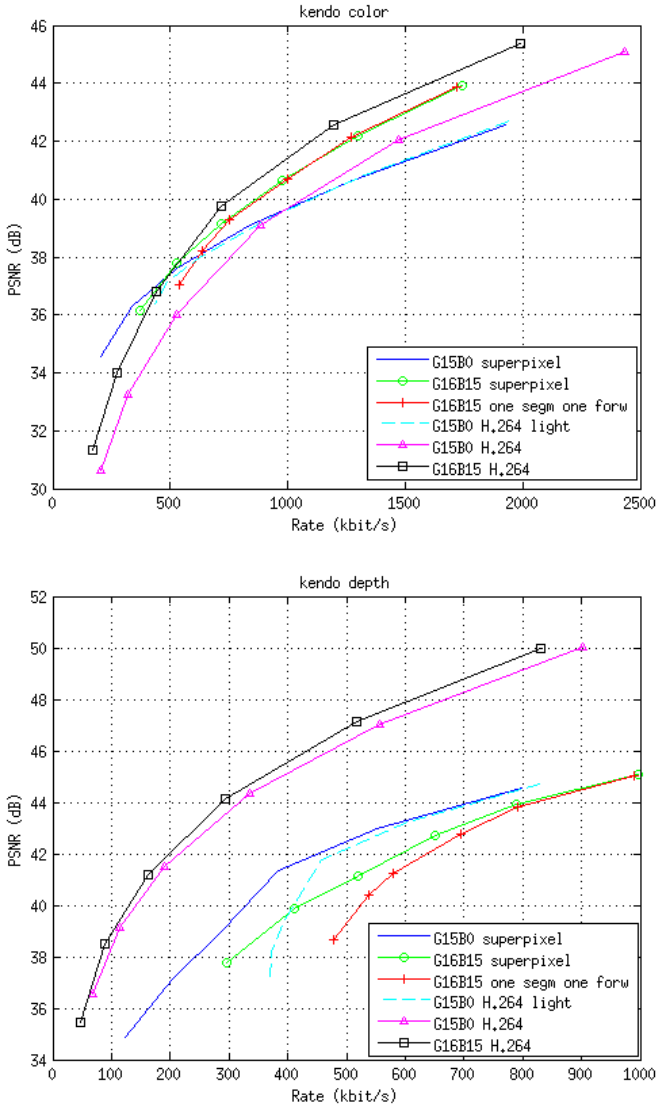


Figura 5.10: PSNR kendo.

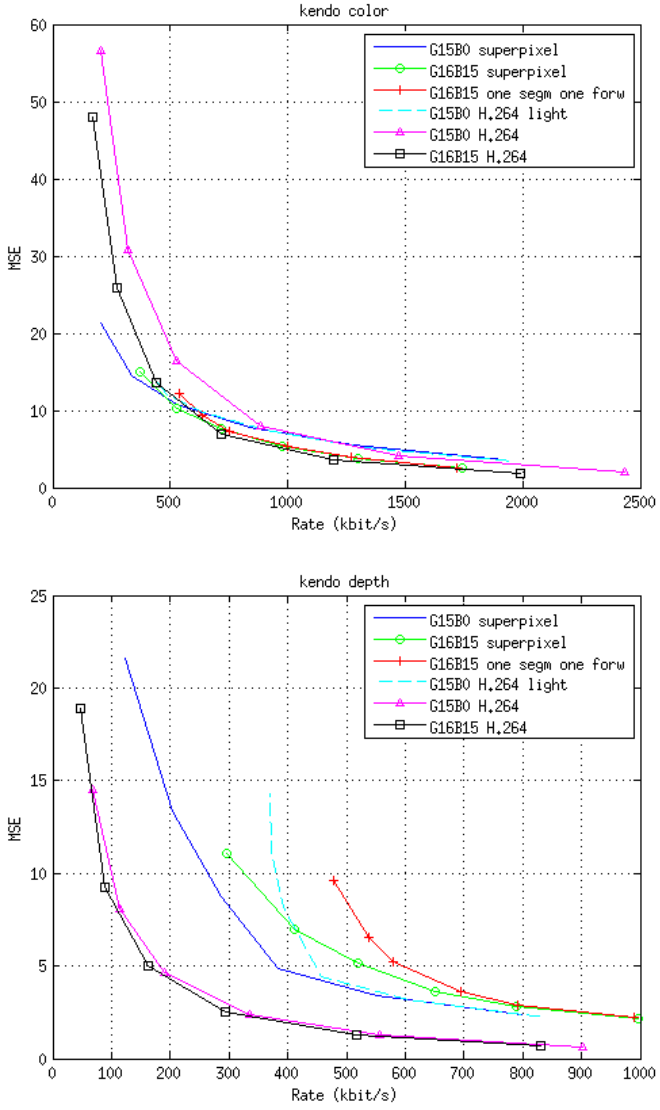


Figura 5.11: MSE kendo.

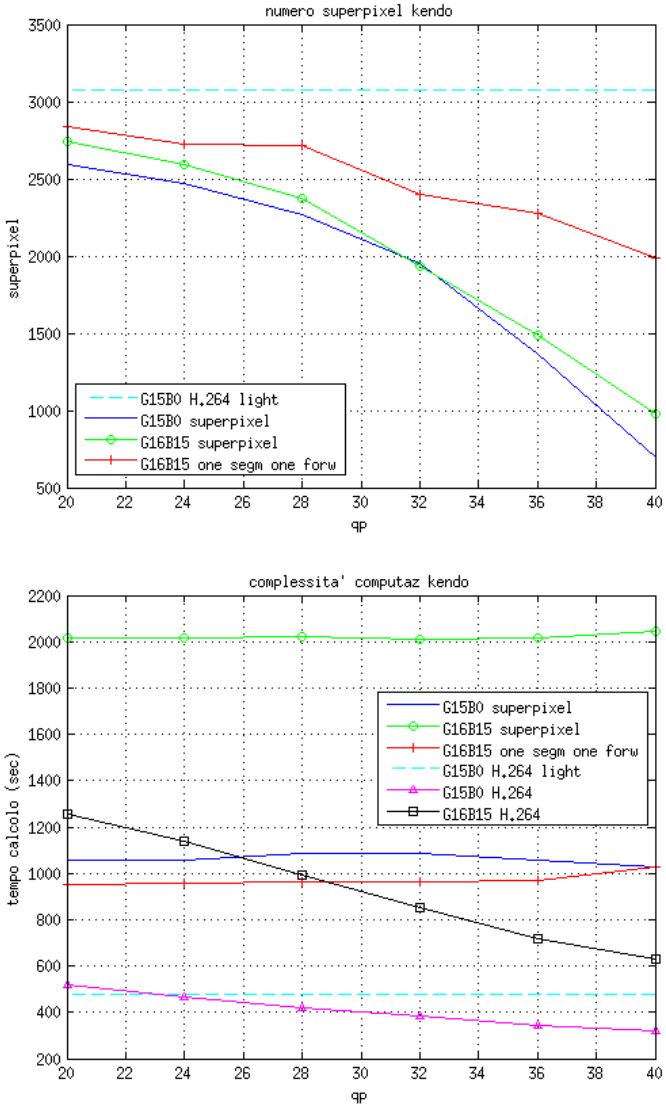


Figura 5.12: Numero superpixel e complessità computazionale kendo.

**Sequenza *butterfly***

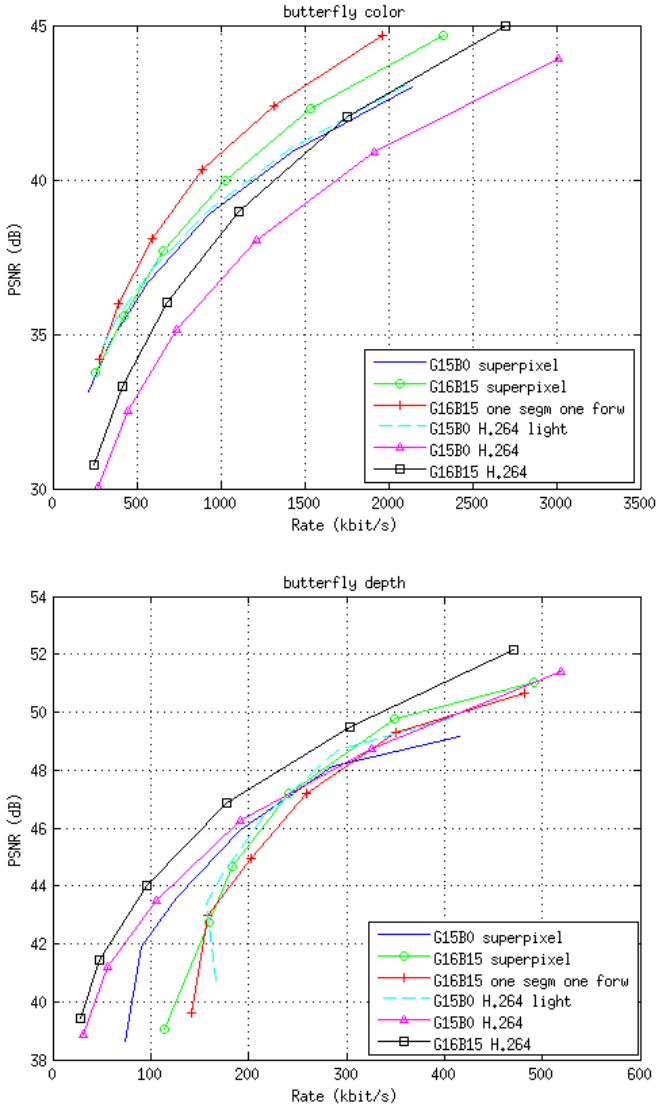


Figura 5.13: PSNR butterfly.

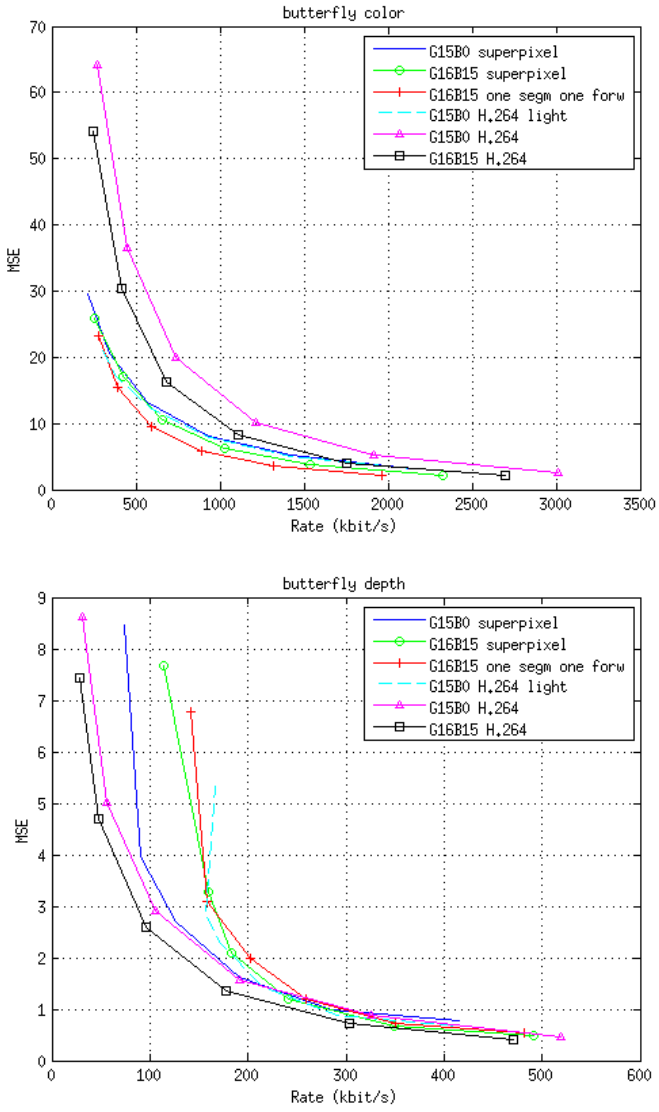


Figura 5.14: MSE butterfly.

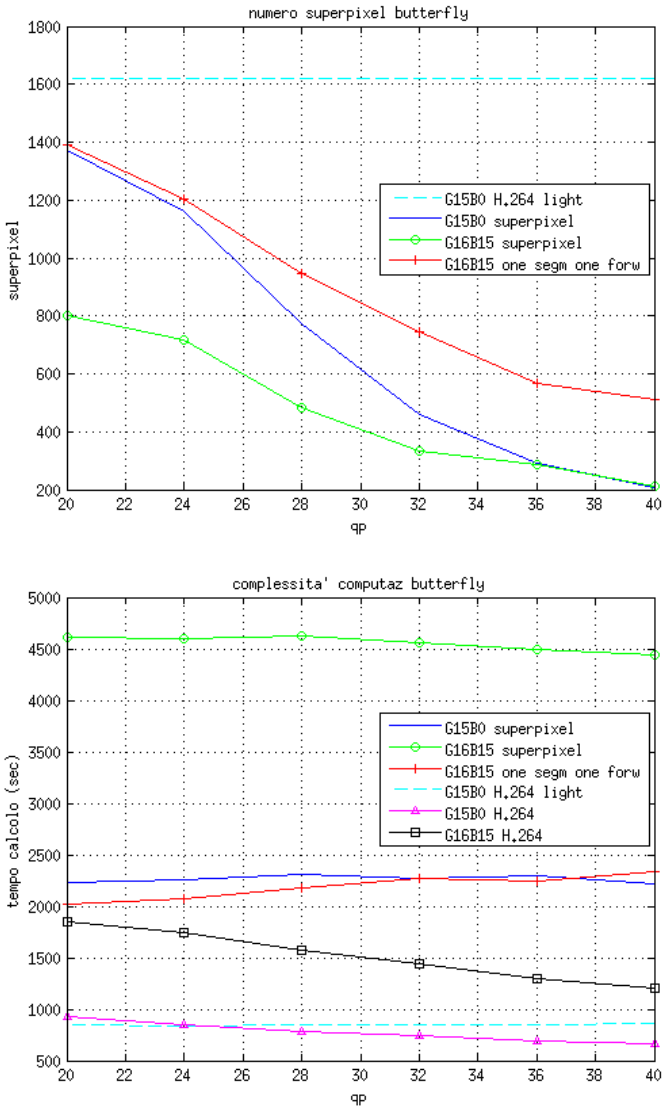


Figura 5.15: Numero superpixel e complessità computazionale butterfly.

**Sequenza *car***

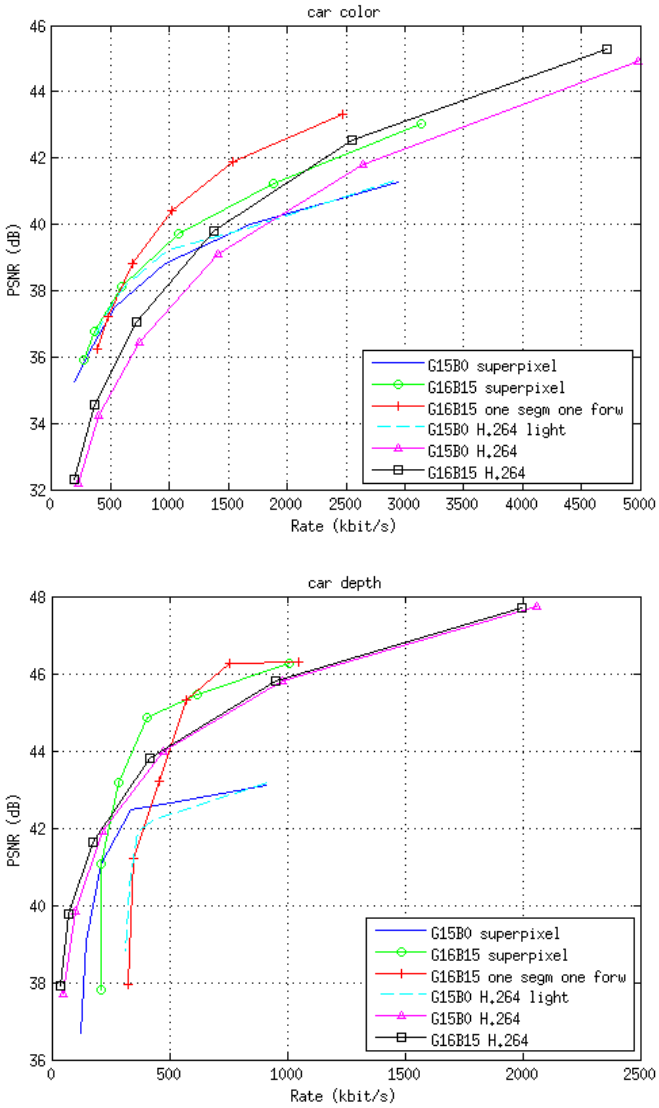


Figura 5.16: PSNR car.

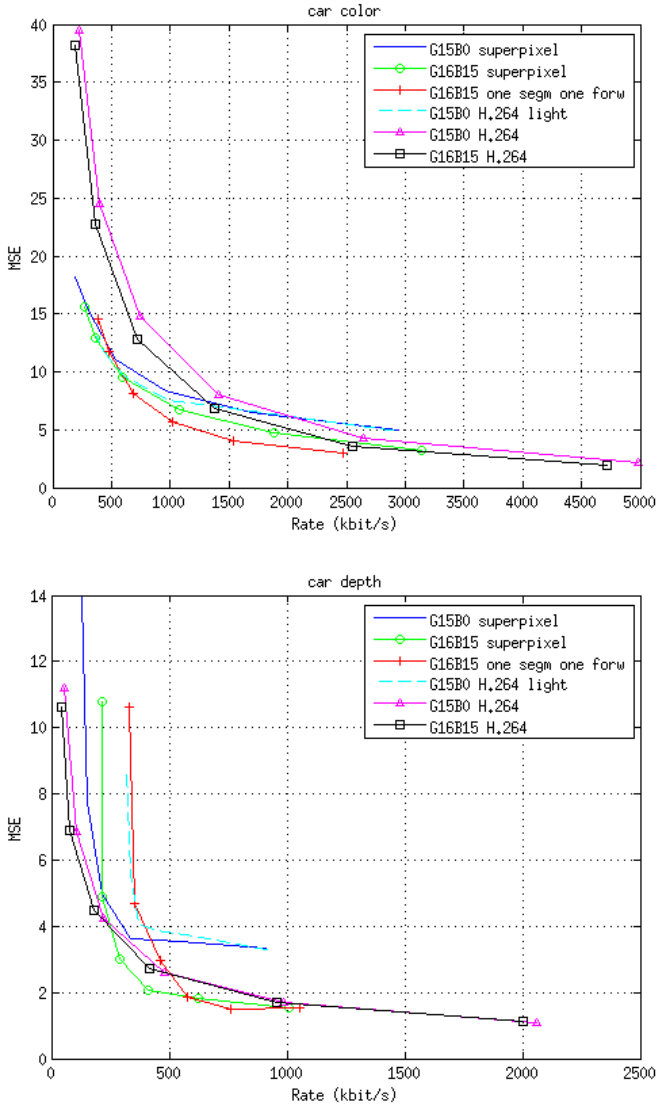


Figura 5.17: MSE car.

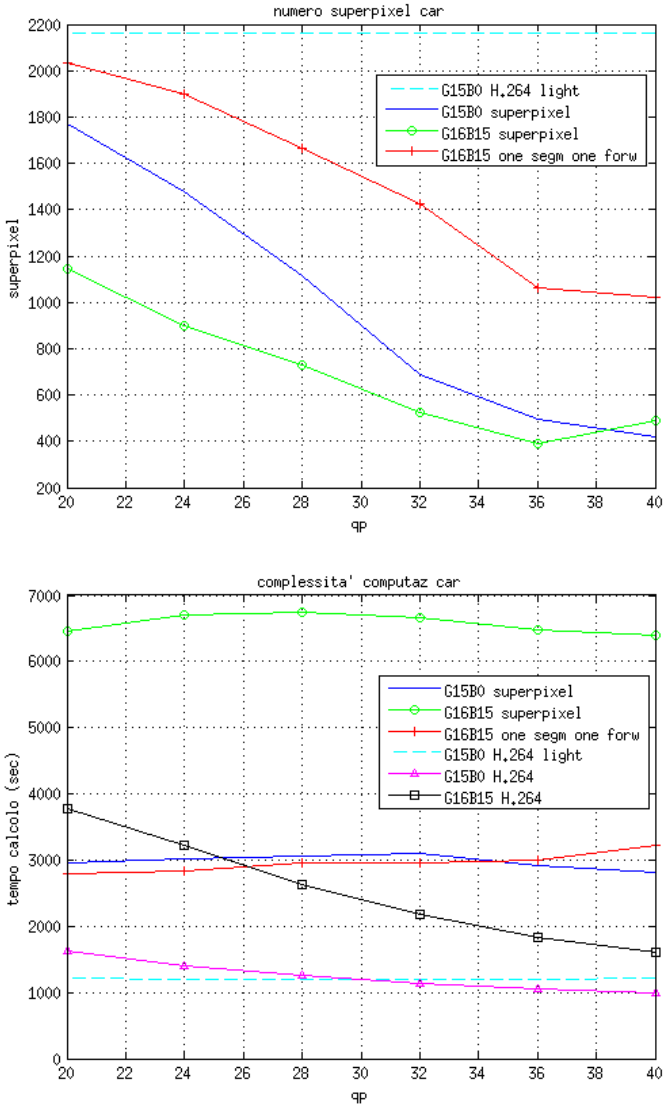


Figura 5.18: Numero superpixel e complessità computazionale car.

È possibile notare come la struttura *G16B15 one segm one forw* fornisca le migliori prestazioni RD per le componenti *color* delle sequenze car e butterfly (si vedano Fig. 5.13, 5.14, 5.16 e 5.17), arrivando ad ottenere un guadagno di quasi 2 dB rispetto la struttura *G16B15 H.264*. Questo perché tale metodologia di codifica si presta molto bene per sequenze pressoché statiche (come il caso di butterfly) e per sequenze caratterizzate da movimenti lineari a velocità costante, caso che interessa la sequenza car, nella quale viene ripresa una macchina percorrente un tratto di strada quasi rettilineo, compiendo dei piccoli movimenti all'interno della corsia. Sempre in tali sequenze, si noti come le strutture *G15B0 superpixel* e *G15B0 H.264 light* forniscano una maggiore efficienza RD (soprattutto per bassi bit-rate) rispetto *G15B0 H.264*. Gli stessi risultati non valgono per le componenti *depth*.

Per sequenze dinamiche quali ballet e breakdancers si può notare come, per entrambe le componenti *color* e *depth*, le strutture *G16B15 one segm one forw* e *G16B15 superpixel* forniscano migliore efficienza RD per alti bit-rate (si vedano Fig. 5.4, 5.5, 5.7 e 5.8), arrivando ad ottenere un guadagno fino a 2 dB rispetto a *G16B15 H.264*, mentre per bassi bit-rate la struttura migliore risulta essere *G15B0 superpixel*, la quale fornisce un guadagno fino a 1.5 dB rispetto *G15B0 H.264 light* e fino a 3 dB rispetto *G15B0 H.264*.

La sequenza kendo è caratterizzata da movimenti "a scatto", con sfondo che varia di poco, quindi la struttura *G16B15 one segm one forw* non produce un'alta efficienza RD (si vedano Fig. 5.10 e 5.11), sebbene si avvicini alle prestazioni di *G16B15 H.264* nella componente *color* (la differenza è di 1 dB). Per quanto riguarda la componente *depth* di kendo si noti come lo standard H.264/AVC fornisca prestazioni nettamente superiori (fino a 6 dB) rispetto le strutture basate sullo schema di codifica di Fig. 4.1. Ciò è dovuto parzialmente alla bassa qualità della mappa di profondità, che non permette un'accurata identificazione dei segmenti in ogni frame.

Si noti come il tempo di codifica relativo alla struttura *G16B15 one segm one forw* sia dimezzato rispetto all'utilizzo di *G16B15 superpixel* e sia pressoché lo stesso di *G15B0 superpixel* (si vedano Fig. 5.6, 5.9, 5.12, 5.15 e 5.18),

mentre le strutture caratterizzate da più bassa complessità computazionale sono date da *G15B0 H.264* e *G15B0 H.264 light*.

In conclusione, per sequenze statiche o caratterizzate da dinamicità lineare a velocità costante, la soluzione migliore in termini di efficienza RD è data dall'utilizzo di *G16B15 one segm one forw*, mentre per sequenze caratterizzate da alto contenuto dinamico la soluzione migliore è data da *G15B0 superpixel* per bassi bit-rate, e invece per alti bit-rate torna ad essere efficiente la struttura *G16B15 one segm one forw*. Tuttavia, tali strutture *segmentation-based* dipendono dalla qualità delle mappe di profondità e richiedono un maggior sforzo computazionale rispetto l'utilizzo di H.264/AVC.

### 5.2.2 Multi-view

Si procede ora a testare le varie strutture di codifica adottate nella sezione precedente in una tipica rappresentazione *Multi-view plus depth* basata sul *3D-warping*. Si considera la sequenza ballet, ripresa ora da un array di 8 telecamere disposte come in Fig. 5.19, e la visuale corrispondente a cam4 assurge a vista di riferimento.

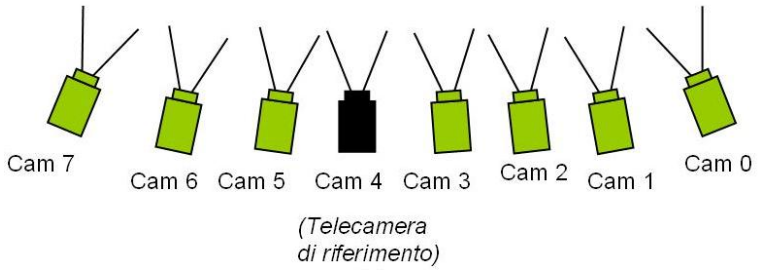


Figura 5.19: Disposizione delle telecamere in una rappresentazione *Multi-view*.

Si è fatto uso del software realizzato da S. B. Kang e L. Zitnick, reso disponibile da *Interactive Visual Media Group Microsoft Research*, il quale effettua un processo di *warping*, generando tutte le viste $V_k$, $k \neq 4$, servendosi delle componenti *color* e *depth* della vista di riferimento, congiuntamente

all'informazione relativa ai parametri intrinseci ed estrinseci delle telecamere presenti attorno la scena.

Le Fig. 5.20 e 5.21 riportano l'andamento PSNR vs bit-rate della componente *color* delle visuali adiacenti la vista di riferimento, ovvero le visuali 3 e 5.

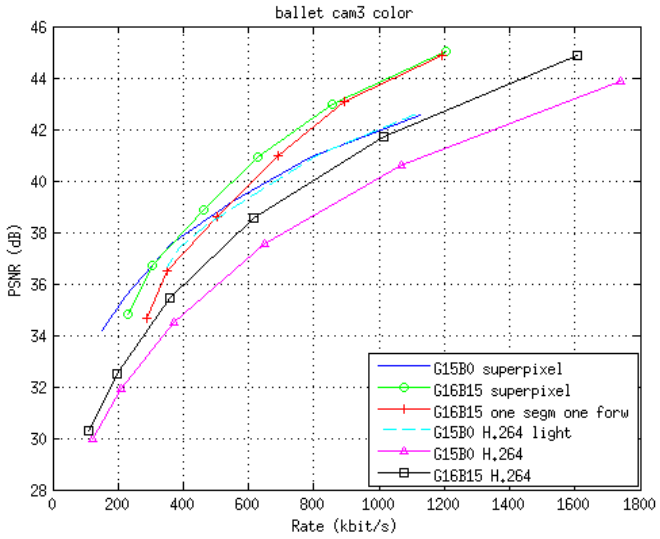


Figura 5.20: PSNR ballet cam3.

Si noti come l'adozione di strutture *segmentation-based* comporti un sostanziale guadagno rispetto l'utilizzo di H.264/AVC, in particolare *G16B15 superpixel* e *G16B15 one segm one forw* forniscono alta efficienza per alti bit-rate, mentre per bassi bit-rate la soluzione migliore è data da *G15B0 superpixel*. Quindi, in ultima analisi, le strutture basate su un processo di segmentazione *object-oriented* riferite allo schema di codifica di Fig. 4.1, si prestano molto bene come possibile integrazione nel processo di codifica MVC.

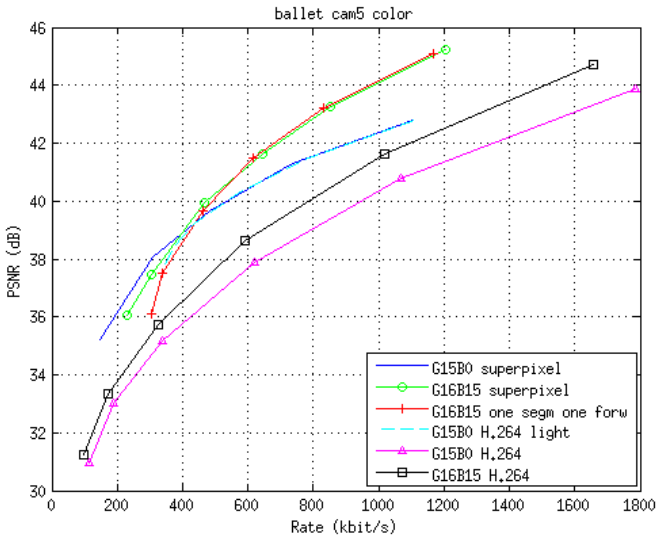


Figura 5.21: PSNR ballet cam5.

# Capitolo 6

# Conclusioni

La prima fase del presente lavoro è stata caratterizzata dall'analisi della struttura del segnale 3D, focalizzando l'attenzione sulle caratteristiche e proprietà delle mappe di profondità. Si è visto come tale tipologia di informazione risulti fondamentale per una corretta resa 3D, soprattutto per operazioni quali *3D-warping*, da qui assumono notevole importanza i sistemi di acquisizione della forma degli oggetti presenti all'interno della scena.

Successivamente, l'attenzione è stata rivolta allo studio delle proprietà e procedure di codifica dello standard H.264/AVC, evidenziandone le caratteristiche basilari e le differenze principali rispetto i precedenti standard di codifica video.

L'obiettivo del presente lavoro è stato caratterizzato dallo studio ed implementazione di algoritmi di codifica video basati su un processo di segmentazione orientato all'oggetto, proponendo un nuovo algoritmo (*G16B15 one segm one forw*) avente lo scopo di diminuire lo sforzo computazionale richiesto nello sviluppo e realizzazione di codifica video per strutture predittive B-gerarchiche, garantendo al tempo stesso alta efficienza RD. I risultati sperimentali hanno evidenziato un guadagno di codifica (fino a 2 dB) di tale struttura rispetto l'utilizzo di codifica predittiva B-gerarchica dello standard H.264/AVC, per sequenze statiche o caratterizzate da dinamicità lineare e costante. Per bassi bit-rate e per sequenze caratterizzate da alto

contenuto dinamico, la struttura *G15B0 superpixel* fornisce miglior efficienza RD, arrivando ad ottenere un guadagno di quasi 3 dB rispetto lo standard H.264/AVC avente stessa struttura del GoP.

Si è altresì notato come tali strutture *segmentation-based* producano guadagni di compressione maggiori rispetto l'utilizzo di H.264/AVC nella codifica delle visuali generate in un processo di *warping*, e dunque possono essere efficientemente sfruttate ed integrate nei sistemi di codifica *Multi-view*.

In ultima analisi, si sottolinea come la qualità delle mappe di profondità assuma un ruolo rilevante per tutte le strutture *segmentation-based*, dato che un basso livello di qualità delle suddette mappe comporta una non accurata identificazione dei segmenti in ogni frame.

# Ringraziamenti

Ringrazio i miei genitori Michele e Paola, mia sorella Violena e la mia fidanzata Amelia, supporti fondamentali durante la mia carriera universitaria.

Un ringraziamento particolare va al Prof. Giancarlo Calvagno e al Dott. Simone Milani, per la grande professionalità e disponibilità nel seguirmi durante la preparazione del presente lavoro.

Gian Luca Vendraminetto

# Bibliografia


[1] Cisco, Inc., "Visual networking index: Global mobile data traffic forecast update", *theruckusroom.typepad.com/files/cisco-rmobile-trends-report.pdf*, Febbraio 2010.

[2] C. Fehn, "A 3D-TV Approach Using Depth-Image-Based Rendering (DIBR)", *Proceedings of Visualization, Imaging, and Image Processing (VIIP'03)*, vol. 3, pp. 482-487, Benalmadena, Spain, Settembre 2003.

[3] S. Milani e G. Calvagno, "3D Video Coding via Motion Compensation of Superpixels", *Proc. of the 19th European Signal Processing Conference (EUSIPCO 2011)*, Barcelona, Spain, 29 Agosto - 2 Settembre 2011, pp. 1899-1903.

[4] L. Favalli e M. Folli, "A Scalable Multiple Description Scheme for 3D Video Coding Based on the Interlayer Prediction Structure", *International Journal of Digital Multimedia Broadcasting*, vol. 2010, article ID. 425641, 2010.

[5] W. Li, "Overview of fine granularity scalability in MPEG-4 video standard", *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 3, pp. 301-317, Marzo 2001.

[6] M. Flierl e B. Girod, "Multi-view video compression exploiting inter-image similarities", *IEEE Signal Processing Magazine*, vol. 24, no. 7, 2007.

[7] A. Fusiello, "Visione Computazionale. Appunti delle lezioni.", *http://ilmiolibro.kataweb.it/schedalibro.asp?id=387047*, Giugno 2008.

[8] S. B. Gokturk, H. Yalcin e C. Bamji, "A Time-Of-Flight Depth Sensor - System Description, Issues and Solutions", *Proc. of CVPRW 2004*, 27 Giugno - 2 Luglio 2004, vol. 3, p. 35.

[9] M. Zamarin, P. Zanuttigh, S. Milani, G. M. Cortelazzo e S. Forchhammer, "A Joint Multi-View Plus Depth Image Coding Scheme Based on 3D-Warping", *Proc. of ACM Workshop on 3D Video Processing (3DVP 2010)*, Firenze, Italy, 29 Ottobre 2010, pp. 7-12.

[10] A. Fusiello, "Image-Based Rendering Methods for 3D Video-communications", *http://www.diegm.uniud.it/fusiello/teaching/mvg/brixen.pdf*, Luglio 2007.

[11] L. C. Zitnick, S. B. Kang, M. Uyttendaele, S. Winder e R. Szeliski. "High-quality video view interpolation using a layered representation". *ACM Trans. Graph.*, 23(3):600-608, 2004.

[12] M. Barbero e M. Muratori, "La televisione stereoscopica: origini, cinema, televisione", *Elettronica e Telecomunicazioni*, no. 2, Agosto 2004.

[13] M. Muratori, "Tecniche per la visione stereoscopica", *Elettronica e Telecomunicazioni*, no. 1, Aprile 2007.

[14] W. Yang, Y. Lu, F. Wu, J. Cai, K. N. Ngan e S. Li, "4-D Wavelet-Based Multiview Video Coding", *IEEE Trans. Circuits Syst. Video Technol.*, 16(11):1385-1396, Novembre 2006.

[15] T. Wiegand, G. Sullivan, G. Bjontegaard e A. Luthra, "Overview of the H.264/AVC video coding standard", *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 13, pp. 560-576, Luglio 2003.

[16] I. E. Richardson, "H.264 and MPEG-4 Video Compression: Video Coding for Next Generation Multimedia", *Wiley*, Agosto 2003.

[17] P. Seeling e M. Reisslein, "Video Transport Evaluation With H.264 Video Traces", *IEEE Communications Surveys & Tutorials*, Agosto 2011.

[18] D. Marpe, H. Schwarz e T. Wiegand, "Context-Based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard", *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 620-636, Luglio 2003.

[19] D. Marpe, T. Wiegand e S. Gordon, "H.264/MPEG-4 AVC fidelity range extensions: Tools, profiles, performance and application areas", *Proc. IEEE Int. Conf. on Image Processing (ICIP)*, Settembre 2005, pp. 593-596.

[20] H. Schwarz, D. Marpe e T. Wiegand, "Overview of the scalable video coding extension of the H.264/AVC standard", *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 9, pp. 1103-1120, Settembre 2007.

[21] P. Seeling, P. de Cuetos e M. Reisslein, "Fine granularity scalable (FGS) video: Implications for streaming and a trace-based evaluation methodology", *IEEE Commun. Mag.*, vol. 43, no. 4, pp. 138-142, Aprile 2005.

[22] M. Wien, H. Schwarz e T. Oelbaum, "Performance analysis of SVC", *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 9, pp. 1194-1203, Settembre 2007.

[23] H. Schwarz e M. Wien, "The scalable video coding extension of the H.264/AVC standard", *IEEE Signal Processing Mag.*, vol. 25, no. 2, pp. 135-141, Marzo 2008.

[24] P. Amon, T. Rathgen e D. Singer, "File format for scalable video coding", *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 9, pp. 1174-1185, Settembre 2007.

[25] S. Milani e G. Calvagno, "A depth image coder based on progressive silhouettes", *IEEE Signal Process. Lett.*, vol. 17, no. 8, pp. 711-714, Agosto 2010.

[26] Y. Morvan, D. Farin e P. H. N. de With, "Depth-Image Compression Based on an R-D Optimized Quadtree Decomposition for the Transmission of Multiview Images", *Proc. of IEEE International Conference on Image Processing (ICIP 2007)*, San Antonio, TX, USA, 16-19 Settembre 2007, vol. V, pp. 105-108.

[27] Jun Zhang, M. M. Hannuksela e Houqiang Li, "Joint multiview video plus depth coding", *Proc. of IEEE ICIP 2010*, Hong Kong, 26-29 Settembre 2010, pp. 2865-2868.

[28] Siping Tao, Ying Chen, Miska M. Hannuksela, Ye kui Wang, Moncef Gabbouj e Houqiang Li, "Joint texture and depth map video coding based on the scalable extension of h.264/avc", *Proc. of IEEE ISCAS 2009*, Taipei, Taiwan, 24-27 Maggio 2009, vol. 3, pp. 2353-2356.

[29] P. Salembier, F. Marqués e M. Pardàs, "Segmentation-Based Video Coding: Temporal Linking and Rate Control", *Proc. of the EUSIPCO 1996*, Trieste, Italy, Settembre 1996, vol. I, pp. 455-458.

[30] F. Marqués, P. Salembier, M. Pardàs, R. Morros, I. Corset, S. Jeannin, B. Marcotegui e F. Meyer, "A segmentation-based coding system allowing manipulation of objects (Sesame)", *Proc. of ICIP 1996*, Lausanne, Switzerland, Settembre 1996, vol. III, pp. 679-682.

[31] R. O. Duda, P. E. Hart e Stork D. G., *Pattern Classification, 2nd Edition*, Wiley, Nov. 2000.

[32] P. F. Felzenszwalb e D. P. Huttenlocher, "Efficient Graph-Based Image Segmentation", *International Journal of Computer Vision*, vol. 59, no. 2, pp. 167-181, Settembre 2004.