H.264

Standard di compressione video dei primi anni 2000

JVT è acronimo di Joint Video Team. Il lavoro di questo team rappresenta uno dei più importanti standard di video compressione degli ultimi anni ed è stato pubblicato come H264 dalla ITU-T (International Telecommunication Union) e come MPEG-4 Parte 10 da ISO/IEC.

1- Il contesto

Il contesto da cui si parte per la creazione di questo standard si colloca agli inizi degli anni 2000 e vede due protagonisti: MPEG e ITU-T.

1.1- MPEG

MPEG veniva vista come una organizzazione di successo, che ha portato alla creazione di importanti standard di video compressione ampiamente accettati e diffusi.
Questo è stato vero specialmente in realtà per lo standar MPEG-2, mentre lo standard MPEG-4 ha avuto alcune difficoltà, riguardanti due aspetti in particolare: innanzitutto la dimensione dello standard rese confuse molto i produttori, e in secondo luogo c'erano stati problemi con le licenze che sono state rilasciate dopo anni di dibattiti e con termini a volte poco ragionevoli. Nel frattempo MPEG-2 stava migliorando, riuscendo a dimezzare il bit rate garantendo la stessa qualità video. Di fatto quindi MPEG-4 andava a migliorare di solo un 15% rispetto a questo nuovo MPEG-2 "maturato".

1.2- ITU-T e la nascita del JVT

I gruppo dei Video Coding Expert della ITU-T, usando le stesse tecniche hanno creato standard con un enfasi particolare per quanto riguarda il problema di aver bassa latenza, che è essenziale per applicazioni interattive real-time come ad esempio la video-conferenza. Avevano di recente iniziato un nuovo standard, H.26L, che venne visto da MPEG come il più promettente. Nel 2001 le due organizzazioni si unirono nel Joint Video Team per andare a formare, come rilasciarono alla stampa, "a single interoperable solution for a next generationof standard video coding".

2- Requisiti

Le due fazioni del team concordarono un set di requisiti. In questo capitolo esamineremo i più importanti.

2.1 Design semplice, usando piccoli “building blocks”, anche a scapito di una imperfetta retro (e futura) compatibilità
2.2 La definizioni di pochi conformance points. La grande quantità di “conformance points” (combinazione di profili e livelli) di MPEG-4 lo avevano reso confusionario.
2.3 Avere un risparmo del 50% del bitrate a parità di qualità di (ad esempio) H.263v2
2.4 Flessibile, in base al delay desiderato. Si tratta di ave re quindi bassi delay per servizi real-time e alti delay per applicazioni di tipo storage. Error resilience e network friendlyness (facilità di pacchettizzazione, packet loss resilience). Per distinguere le due modalità si sarebbe ad esempio potuto usare la presenza o mancanza delle B-pictures (che in effetti migliorano molto la qualità a discapito della velocità dovuta al riordino delle picture).
2.5 Complessità del processo di codifica e decodifica asimmetrico, in generale, ma nel caso di video conferenza, garantire l’effetto realtime, rilassando questo requisito.
2.6 Specifica completa del decodificatore. Non c’era infatti una esatta spefica della Inverse DCT. L’ambiguità dei risultati del decodificatore comporta errori anche nella codifica, visto che spesso nel processo di codifica vengono utilizzati parti del decodificatore.

Considerazioni
Da questi requisiti notiamo come si sia agito per arginare molti problemi che erano sorti con gli standard precedenti come appianare le complessità dello standard MPEG-4 piuttosto che dare una specifica esatta del decodificatore. Allo stesso tempo, traspare la svolta che si stava verificando: l’unione del mondo della video-conferenza con quello storage, in un’unico standard adattabile ad ogni esigenza.

3- Implementazione

La specifica del codec è un documento di 269 pagine denso di concetti tecnici. Segue una overview degli aspetti più importanti, partendo dalle similarità tra H264 e MPEG-2 per poi passare ad analizzare i nuovi concetti e tecnologie introdotti.

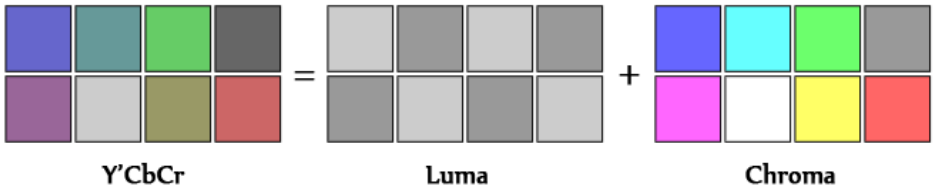
3.1- Diagramma a blocchi
INSERIRE DIAGRAMMA

3.2- Concetti ereditati da MPEG-2

FONTEhttps://discoveringvfx.wordpress.com/2012/03/02/sottocampionamento-della-crominanza-444-422-420/

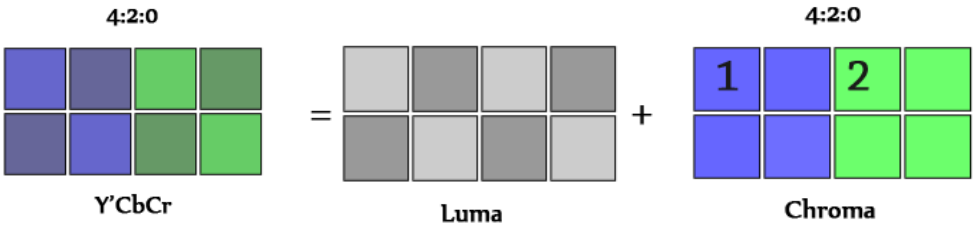
3.2.1 Sottocampionaento 4:2:0
- 4 è la larghezza della matrice
- 2 è il sottocampionamento della riga superiore
- 0 è il sottocampionamento della riga inferiore

Ripasso:



Y'CbCr è la matrice originaria, campionata (non sottocampionata) in 4:4:4 cioè otteniamo una matrice larga 4 che usa 4 campioni per la riga superiore (tutti) e 4 per la riga inferiore.

Invece, sottocampionando in 4:2:0 otterremmo una matrice larga 4, con 2 campioni per la riga superiore e nessuno per la riga inferiore.

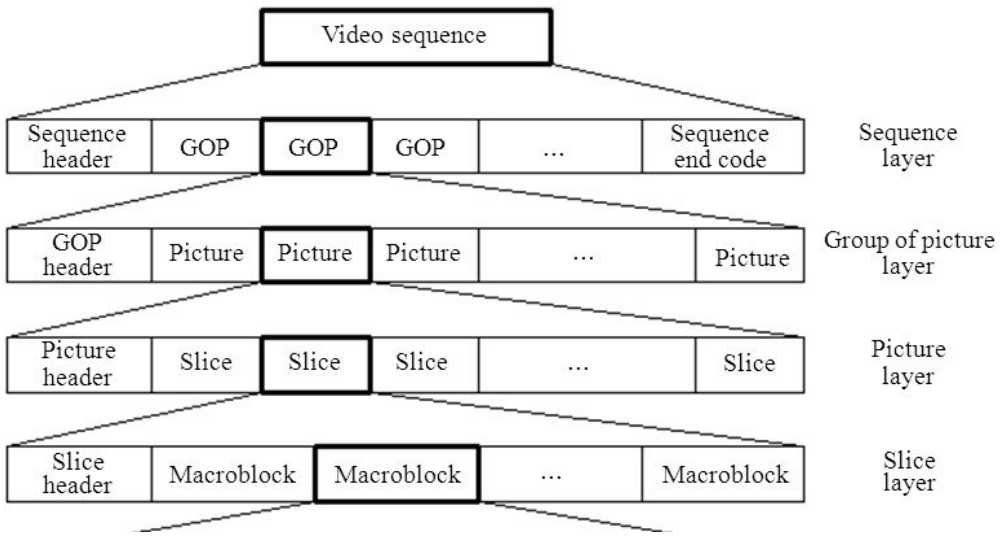


3.2.2 Luminanza da 16x16 pixel, crominanza 8x8

Infatti questa è conseguenza immediata del punto precedente. 256 pixel di luminanza, con crominanza sttocampionata a 4:2:0 implica una crominanza 8x8

3.2.3 Divisione in slice e macroblocchi

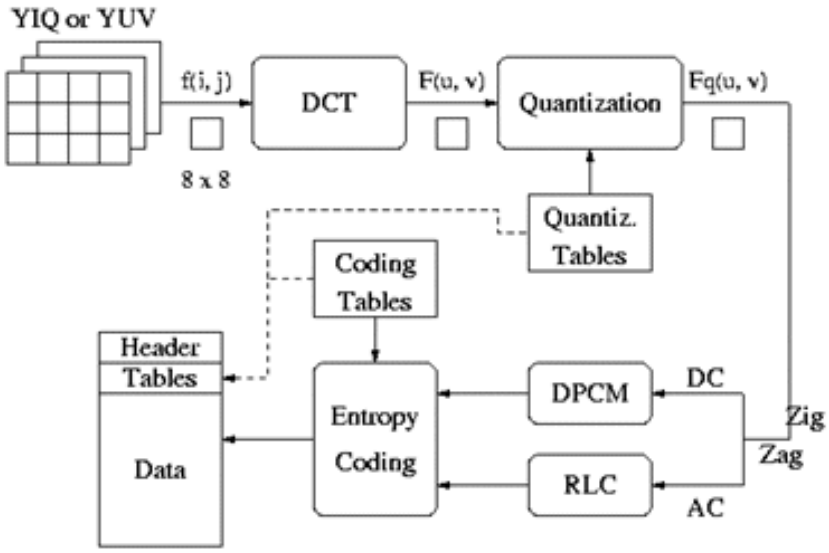
Ripasso: struttura di MPEG-2



3.2.4 Block based transform coding
L'immagine viene divisa in blocchetti che vengono forniti in input alla trasformata.

3.2.5 Quantizzazione, Scan, Lossless coding
Indubbiamente riprende i concetti sfruttati in JPEG

Ripasso: diagramma a blocchi di JPEG



3.2.6 I-, P, B-frames, motion vector, motion compensation
Per ridurre la ridondanza temporale

3.3- Nuovi concetti e tecnologie

3.3.1 Blocchetti 4x4 e non più 8x8.

3.3.2 Integer Transform (derivata dalla DCT di fatto).
Una delle domande che si sono posti era: "Perchè fare calcoli così complessi, ad alta precisione, per poi quantizzarne i risultati? Forse un risultato equivalente si può ottenere approssimando il tutto ad alcuni passi precedenti".
Infatti i coefficienti che venivano usati nella DCT erano questi nella figura a destra.

$$a = \frac{1}{2}$$
$$b = \sqrt{\frac{1}{2}}\cos\left(\frac{\pi}{8}\right)$$
$$c = \sqrt{\frac{1}{2}}\cos\left(\frac{3\pi}{8}\right)$$

I membri del team hanno fatto delle operazioni semplici di moltiplicazione e scaling, in modo che i numeri all'interno delle matrici della DCT fossero ± 1 o= $\pm\sqrt{2}-1$ (circa 0.414). Quindi hanno approssimato questo risultato a ½, hanno cercato di ottenere matrici equivalenti utilizzando il doppio di questi coefficienti, in modo da far scomparire le frazioni, ottenendo solo ± 1 e ± 2 . In aritmetica binaria questo comporta semplicemente un left shift, e rende la trasformata semplice ed efficiente da implementare sia in hardware sia in software. Oltretutto l'inversa è immediatamente definibile e utilizzerà il right shift.

3.3.3 Trasformata aggiuntiva per i coefficienti DC

Continuando quanto esposto al punto precedente, per quanto riguarda la luminanza, siccome i macroblocchi sono 16x16, mentre la Integer Transformate usa blocchetti 4x4, servono 16 trasformate, che porteranno quindi ad avere 16 elementi DC. E' possibile quindi disporre questi 16 elementi in un ultimo blocchetto, 4x4 e applicare una nuova trasformata (di Hadamard) che usa solo coefficienti ± 1 . Per quanto riguarda la crominanza viene applicata una trasformata simile, a partire da blocchetti più piccoli.

Considerazioni

La Integer Transformate quindi non migliora di molto la qualità della compressione, a causa anche di queste approssimazioni, ma migliora moltissimo la sua efficienza, grazie all'utilizzo di semplici operazioni aritmetiche.

3.3.4 Quantizzazione logaritmica

Anche la quantizzazione cambia, dipende da un parametro che varia da 0 a 51, ed è logaritmica perché ogni aumento di 6 del valore di questo parametro, raddoppia il livello di compressione.

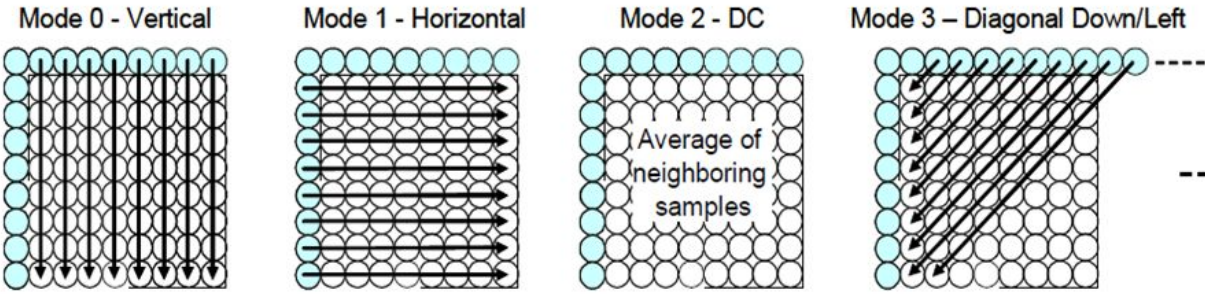
fonti

https://www.slideshare.net/mwalendo/h264vs-hevc

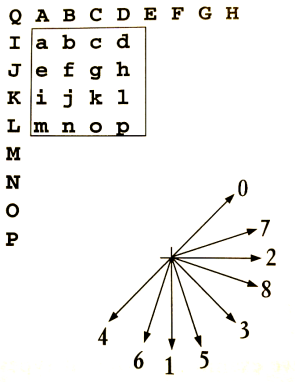
3.3.5 Spacial prediction

MPEG-2 usava la spacial predictive coding per i coefficienti DC; questo standard aggiunge la predizione della prima riga e prima colonna dei coefficienti AC.
E' quindi possibile predirre un completo macroblocco partendo da questi valori:

a- usando una delle quattro modalità previste (orizzontale, verticale, DC, diagonal)



b- usando un blocco vicino precedentemente decodificato, e usando una delle 10 modalità previste. In questo caso ogni modalità viene espressa usando una certa direzione



3.3.6 Deblocking filter

Tutti sanno che i sistemi DCT-based possono formare l'effetto "blocking". H.264 ha fronteggiato il problema introducendo un deblocking filter. Questo filtro opera sia nei lati orizzontali di un blocchetto 4x4 che in quelli verticali, e può modificare i valori dei pixel fino a 4 per ogni lato. Questo concetto non è nuovo, ma questa è la prima volta che viene imposto in uno standard che questa feature debba essere presente dei decoder. Inoltre visto che il decodificatore viene usato spesso anche in fase di codifica, viene incluso anche nel codec. Certamente a bassi bitrate l'immagine degrada, ma la degradazione è molto più dolce, accettabile, simile ai risultati a bassi bit rate usati nella wavet transform (usata in MPEG-2000).

IMMAGINE LENA FONTE
http://live.ece.utexas.edu/publications/2011/cy_tip_jan11.pdf
http://www2.ece.rochester.edu/~bojnordi/data/ccece06.pdf

3.3.7 Motion Estimation

RIPASSO
La motion estimation consiste nel cercare il motion vector e memorizzare le differenze tra il frame costruito con i motion vector e il frame originale

MANCA IMG

Le due importanti migliorie che JVT ha apportato sono:
a- motion vector più precisi;
b- determinazione più accurata dell'area dell'immagine a cui deve essere applicato il motion vector.
Infatti MPEG-2 permetteva ai motion vector di avere una precisione a mezzo pixel. MPEG-4 permetteva una precisione a un quarto di pixel, ma senza ottenere grandi miglioramenti. Inizialmente si stabilì che H.264 avrebbe usato una precisione ad un ottavo di pixel, ma poi questa feature venne tolta. Ci si rese conto che questa precisione aggiuntiva è utile solo se i pixel delle immagini vengono interpolati con una corrispondente accuratezza. MPEG-2 e 4 usavano la two point linear interpolation (un pixel era la media dei suoi pixel più vicini), che era assolutamente insufficiente in casi di zone ad alta frequenza, più ricche di dettagli.

Si scelse quindi un compromesso: precisione a mezzo pixel con una interpolazione a 6 punti, e una seconda interpolazione a due punti con una precisione a un quarto di pixel.

(?)
Altro aspetto da considerare è che i macroblocchi quadrati spesso non corrispondono ai confini degli oggetti rappresentati nei video. Per questo motivo il macroblocco di default ha dimensione 16x16 ma quando necessario può essere partizionato fino ad arrivare a blocchetti 4x4. Usando varie combinazioni dei blocchi si riesce ad ottenere una buona approssimazione dell'oggetto.

IMMAGINE

3.3.8 Temporal Prediction
Anche questo standard usa P-B-I-frames, ma il cambiamento sta nel fatto che consente riferimenti multipli alle immagini (fino a 15), e viene usato un reference picture index per riferirsi a una certa figura. Spesso sono sufficenti solamente due riferimenti. Questa possibilità offre grandi miglioramenti nelle scene caotiche. La penalità da pagare in termini di complessità del decoder è molto piccola ma serve più memoria per memorizzare i riferimenti. Da notare che può essere preso in riferimento qualsiasi frame purché venga decodificato prima del frame target (e non visualizzato a display prima). Il tutto viene calcolato a livello di slice.
IMMAGINE
Ulteriore feature introdotta è la predizione pesata. In altre parole MPEG-1 e -2 usavano una predizione in cui un macroblocco veniva calcolato come media dei suoi due vicini ((f-1)+(f+1))/2. Nel caso di una sequenza I B B B P questo calcolo risulta corretto solo per la B centrale. In JVT infatti viene introdotto un peso tale per cui la terza B, ad esempio, viene calcolata come (3P + I)/4.
Alcuni lavoratori hanno concluso che, applicando questa tecnica negli effetti di fade e dissolve l'efficienza è raddoppiata.

3.3.9 Macroblock-adaptive Frame/Field Coding
Anche negli standard precedenti esistevano i frame e i field coding, a livello di macroblocco (l'alternativa sarebbe stata a livello di immagine).

RIPASSO

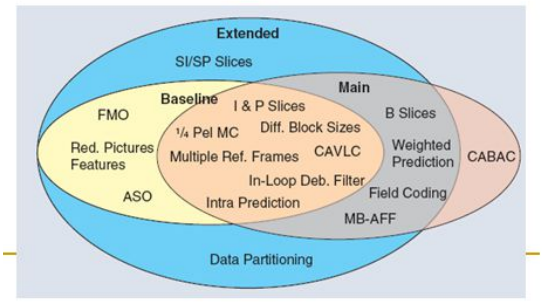
La novità sta nel fatto che la scelta di quale metodologia utilizzare. Infatti in generale in JVT viene usata la frame coding, ma in caso di motion verticale ad esempio viene utilizzata la field coding che si presta meglio.

3.3.10 Lossless Coding
L'ultimo passo di ogni compressione è la compressione senza perdita. In questo standard non viene usata la VLE, ma la CAVLC (Content-based adaptive VLC), o la CABAC (Context Based Adaptive Binary Arithmetic Coding) nelle applicazioni ad alto bit-rate, che sono una versioni migliorate della VLE in quanto si adattano al contesto.

4 Profili

Un profilo definisce un insieme di caratteristiche che possono essere usate; all'interno di ogni profilo, un livello imposta i vari limiti/parametri (es. bit rate). Lo standard prevede tre profili, di cui di seguito illustreremo le principali caratteristiche.

Fonte
http://slideplayer.com/slide/2409533/9/images/32/Profiles+and+Applications.jpg



4.1- Baseline Profile
Usato soprattutto per applicazioni dove la lantenza deve essere minimizzata, come ad esempio la video conferenza. La caratteristica più importante è che non consente l'uso delle B-slice. Questo evita di dover riordinare la sequenza delle picture e riduce di molto i delay. Usa il deblocking-filter e la precisione a un quarto di pixel del motion vector. Non usa field coding o adaptive field coding (solo frame coding). Usa la Variable lenght Coding (no CABAC)

4.2- Main Profile
Usato nelle applicazioni di tipo broadcast, digital cinema e media storage. Include la maggior parte delle feature del livello precedente, reintroducendo le B-slices, la adaptive frame/field coding e la field coding la predizione pesata e il CABAC.

4.3- Extended Profile
Usato per le applicazioni in streaming. E' un sovrainsieme delle caratteristiche del baseline profile, riprende alcuni dei punti del main profile e usa tecniche particolare per l'interfacciamento di rete.

4.4- Livelli
JVT stabilisce 5 livelli più importanti che dovrebbero bastare alla maggior parte delle applicazioni, anche se dispone di 10 sottolivelli aggiuntivi in caso di necessità particolari. I vari livelli permettono bitrate da 1.485 Mbits/s fino a 983 Mbits/s, con una massima di frame size da 99 a 37 macroblocchi.

4.5- Considerazioni

H.264 è uno standard che sarebbe stato usato sia per applicazioni video-conferenza che applicazioni di tipo storage, coprendo una vasta gamma di servizi, aventi anche caratteristiche a volte discordanti. Nonostante questo notiamo come si sia fatto uno sforzo per non appesantire troppo lo standard, con troppi profili e livelli come in MPEG-4.

Fonti Amruta Kulkarni

http://slideplayer.com/slide/6253216/

video

video h264-5
https://www.youtube.com/watch?v=qL22L0mRSDs&pbjreload=10

video sedia h264 mpe2
https://www.youtube.com/watch?v=PW9OnLTqZeohttps://www.youtube.com/watch?v=JIOaIEDMGr0

http://www.fastvdo.com/spie04/spie04-h264OverviewPaper.pdf
DICE il pcm lossless macroblocks