

MPEG 1

- Sviluppato inizialmente per video CD
 - Il bitrate di ~1.5 Mbps
- Compressione intra- e inter-frame
- È uno standard ISO (ISO/IEC 11172)
- Viene definito il formato del file
 - ...quindi definisce gli algoritmi di decompressione
 - Gli algoritmi di compressione possono variare

- È stato pensato per essere asimmetrico
 - 40 ms per decomprimere, per ottenere 25 fps
 - JPEG è circa simmetrico
 - Sfrutta idee usate nel JPEG

- Si raggiunge qualità broadcast sui 3-4 Mbps, ma in alcuni casi (es. sport) servono 6 Mbps
 - MPEG-2 ottiene prestazioni simili a 4 Mbps, grazie al processing basato su field

- MPEG-1 standardizza la sintassi per la rappresentazione di uno stream codificato ed il metodo di decodifica

- La sintassi dello standard include le operazioni di
 - discrete cosine transformation (DCT),
 - motion-compensated prediction,
 - quantizzazione
 - variable length coding

- Chi disegna l'encoder è libero di implementarlo come vuole/riesce
 - Per esempio non è standardizzato come si fa stimare il moto, per la compressione interframe

Fattori che influenzano la qualità

- Risoluzione del video originale
- Bitrate consentito dopo la compressione
 - Non c'è “graceful degradation”
- Efficacia degli stimatori di movimento
 - Per ridurre gli artefatti di compressione

- MPEG-1 consentirebbe risoluzioni fino a 4095x4095 @ 60 fps
- Tipicamente si ha a che fare con un subset di questi parametri:
 - CPB: Constrained Parameters Bitstream

CPB

horizontal resolution	≤ 768 samples
vertical resolution	≤ 576 scan lines
picture area	≤ 396 macroblocks
pel rate	$\leq 396 \times 25$ macroblocks per second
picture rate	≤ 30 frames per second
bit rate	≤ 1.856 Mbps

- Un macroblocco è 16x16 pixel
 - 396 macroblocchi > 101.376 pixel
 - Codifico SIF da 352x240 o 352x288

- In codifica si scala 704x480 o 704x576 a risoluzione SIF
 - Tipicamente si ignora il field 2 e si scala orizzontalmente il field 1
- In decodifica viene effettuata la scalatura inversa
- Si cerca di avere sempre un numero di campioni Y divisibile per 16:
 - $(720/2) / 16 = 22.5\dots$
 - $(704/2) / 16 = 22\dots$ più pratico per i macroblocchi

Risoluzioni tipiche

Resolution	Frames per Second
352 × 240	29.97
352 × 240	23.976
352 × 288	25
320 × 240 ¹	29.97
384 × 288 ¹	25

- 1 = square pixel

Video interlacciato

- MPEG-1 è nato per gestire video progressivo (non interlacciato)
 - Per migliorare la qualità si potrebbero combinare due field in un frame e codificarlo
 - Nel decoding si separano i due field
 - Si introducono però artefatti dovuti al diverso campionamento temporale degli oggetti in movimento
 - Se servono i field è meglio usare MPEG-2 che li gestisce nativamente

GOP

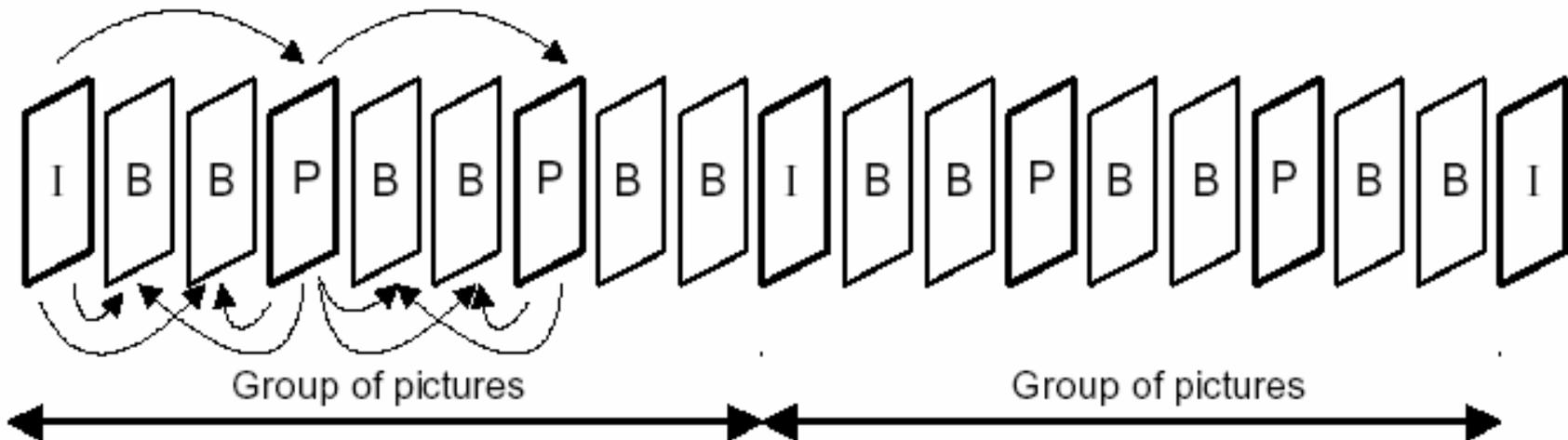
Una sequenza video è suddivisa in Groups of Pictures (GOPs).

Quattro tipi di fotogrammi: I-, P-, B-, D-picture.

I-, P-: anchor picture; distanza tra loro: M.

Distanza tra I-picture successivi: N

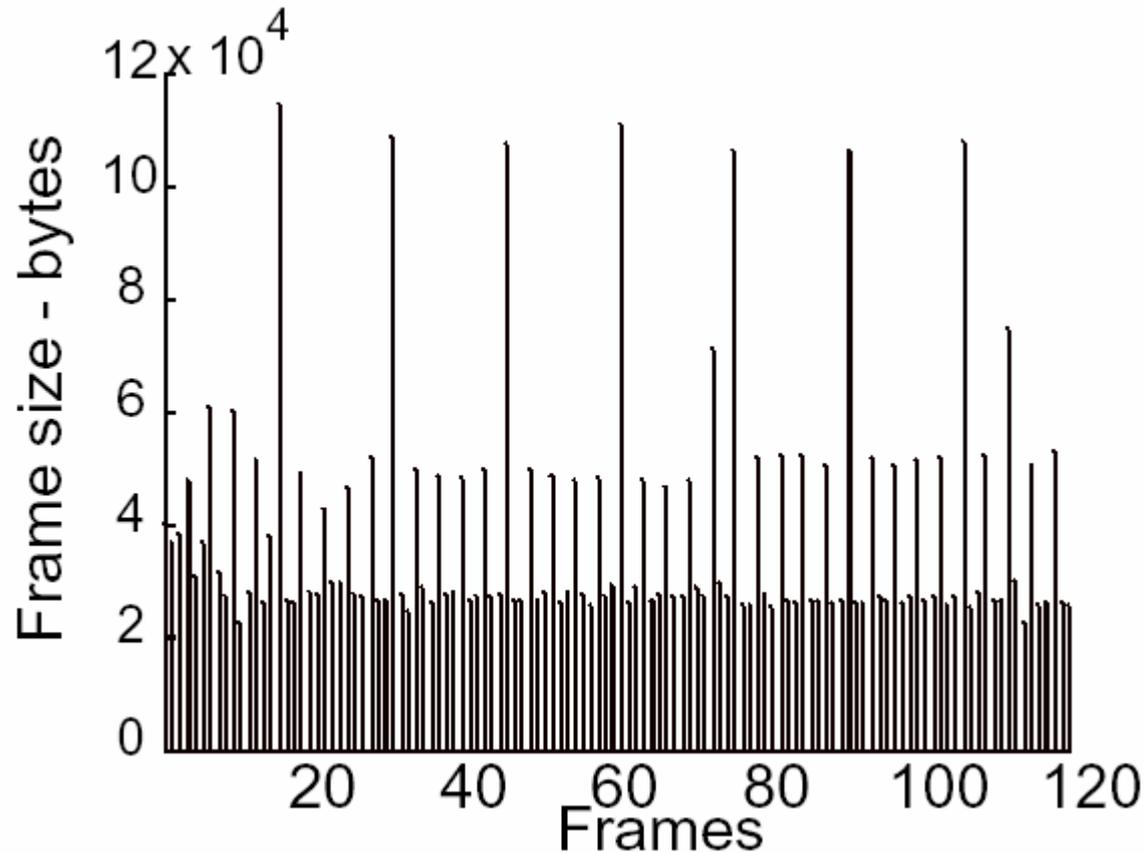
M=3, N=9:



GOP (cont.)

- Le distanze tra I,P e B frame è configurabile in fase di codifica
- Più è piccolo il GOP migliore è la risposta al movimento, ma più piccola la compressione (colpa di I frame)

Es.: dimensione frame

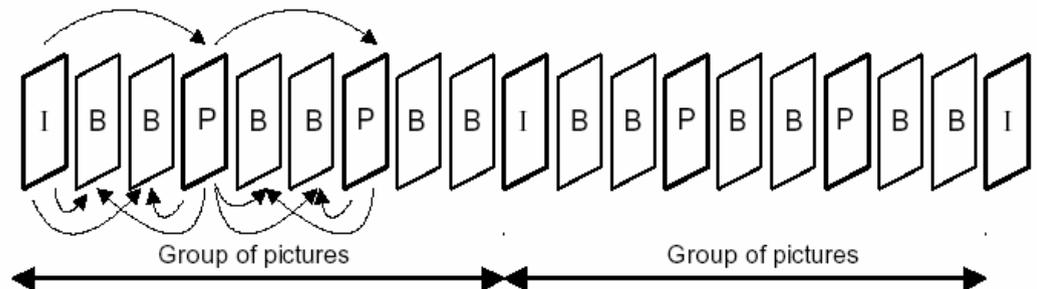


I-Picture, P-Picture

Intra-Coded Picture: codificati indipendentemente, senza riferimento ad altri fotogrammi (anchor).

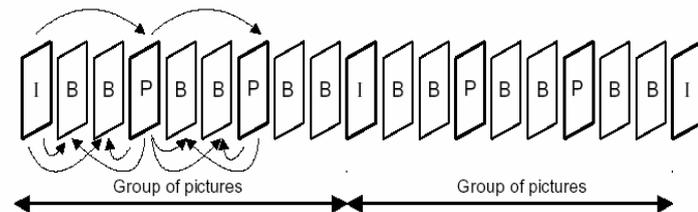
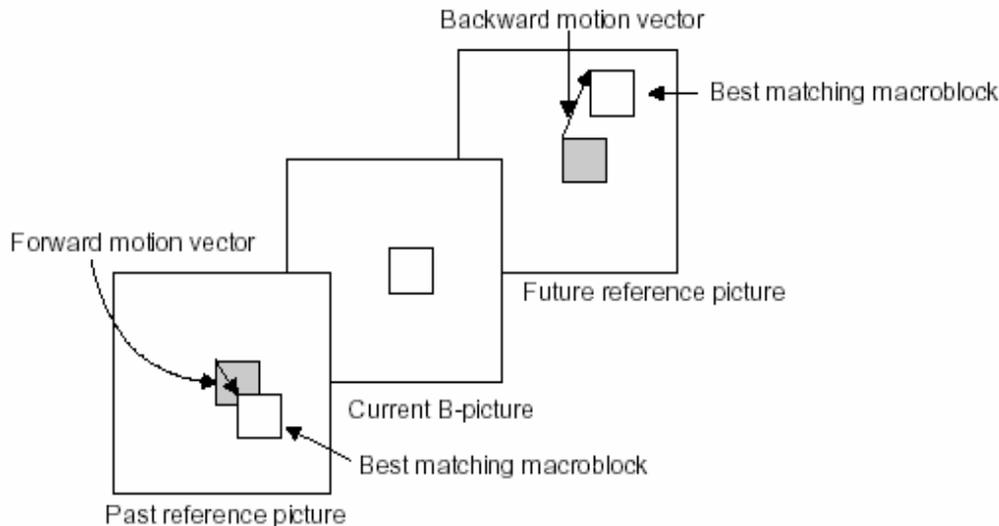
Minimizzazione della propagazione di errori di trasmissione dovuti ad errori nei fotogrammi precedenti. Consente accesso random.

Predictive-Coded Picture: codificati con predizione del moto in avanti (forward motion prediction) partendo da I- o P-picture precedenti.



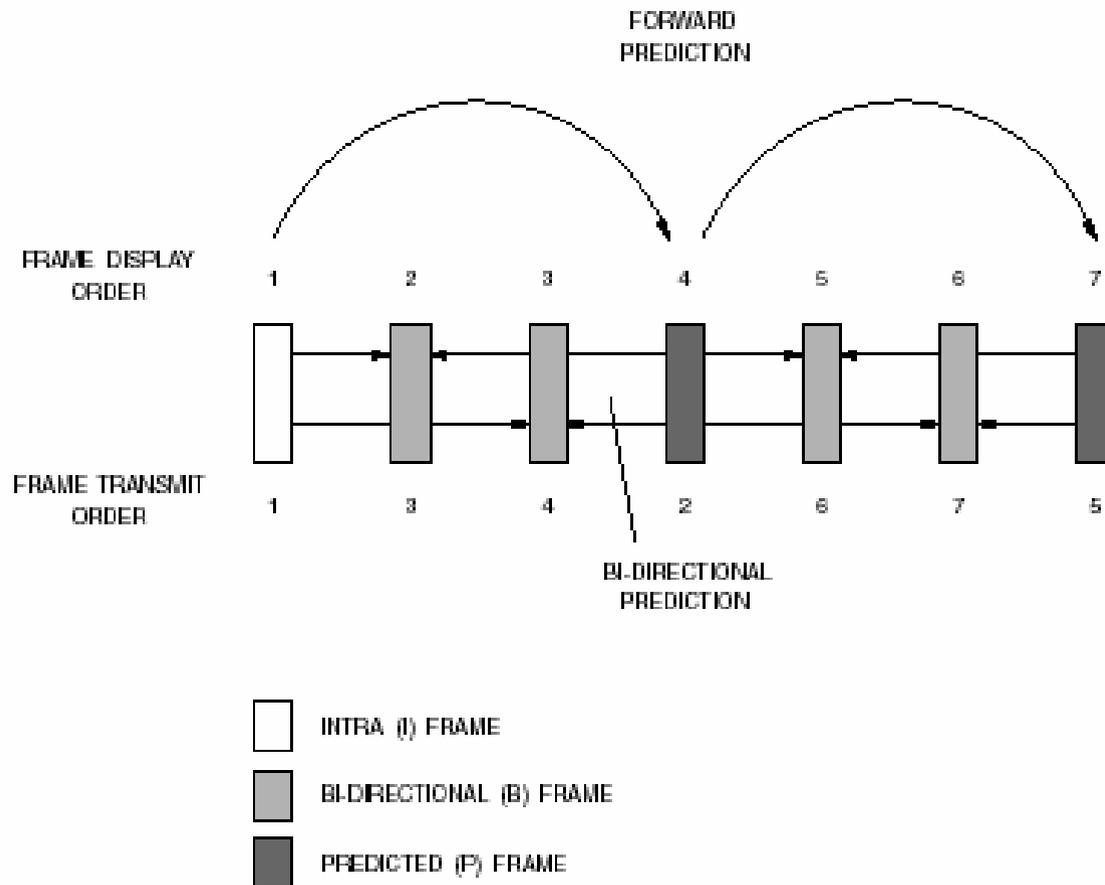
B-Picture

Bi-directional-Coded Picture: codificati con compensazione del moto bidirezionale: predizione basata su passato e futuro usando I- e P-picture (no B-picture). Posso fare una media tra passato e futuro per stabilire il moto.



B-Picture (cont.)

B-Picture: più complicati da codificare, ritardo nella codifica.

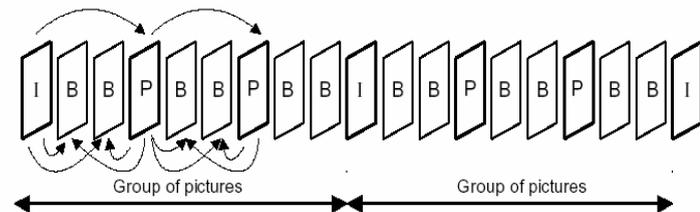


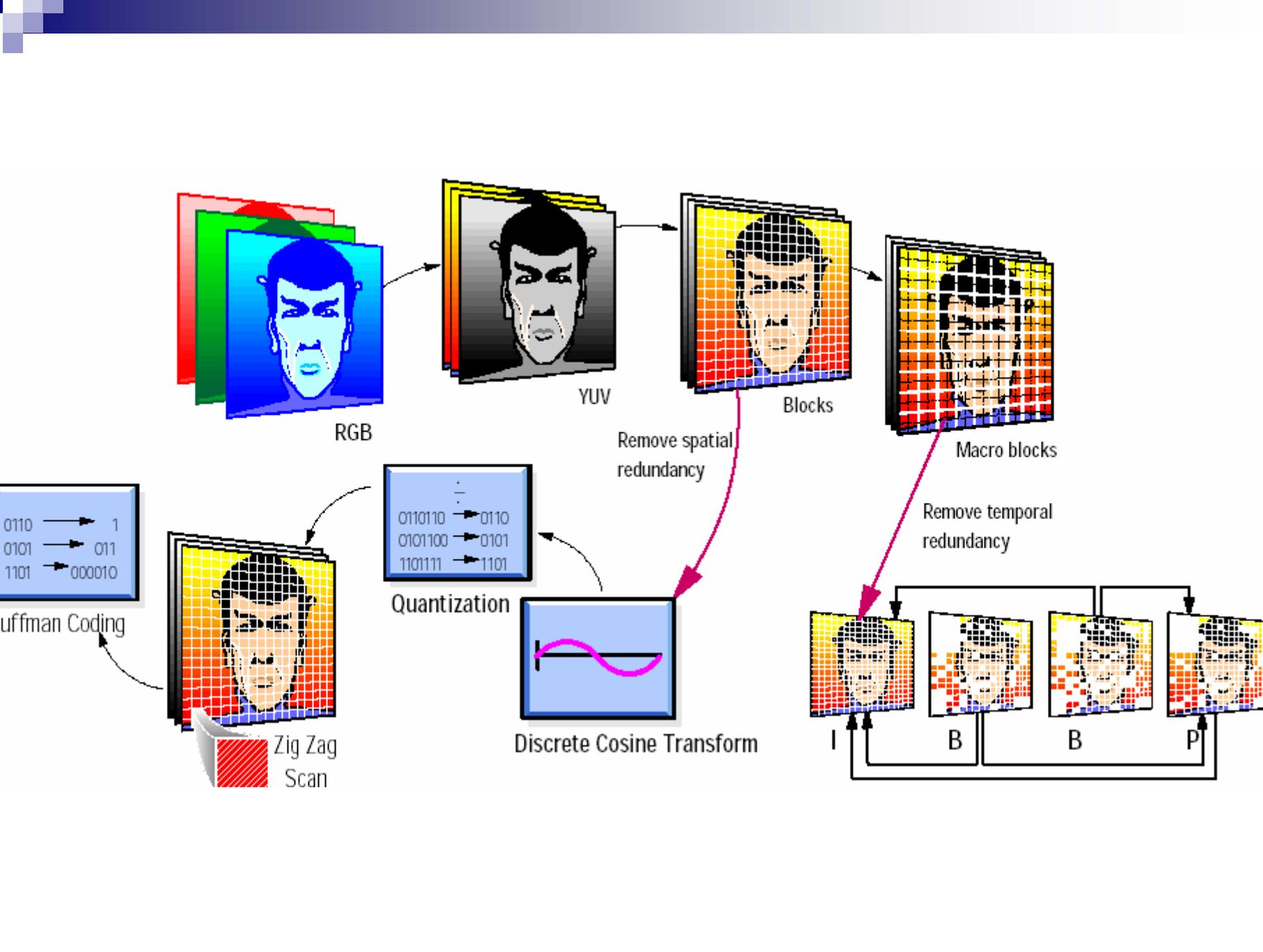
- Un GOP si dice *closed* se può essere decodificato senza usare frame del GOP precedente
- È *open* se richiede qualche frame del GOP precedente
 - Se un GOP finisce con un I o P, oppure con un B che usa solo predizione basata su dati passati allora è *closed*

D-Picture

DC-Picture: fotogrammi low res che usano solo la componente DC dei coefficienti DCT dei vari macroblocchi.

Sono usati raramente, solo per aiutare a fare un browsing del video, devono stare in sequenze separate.





Slice e Macroblocchi

Un frame MPEG è composto da slice.

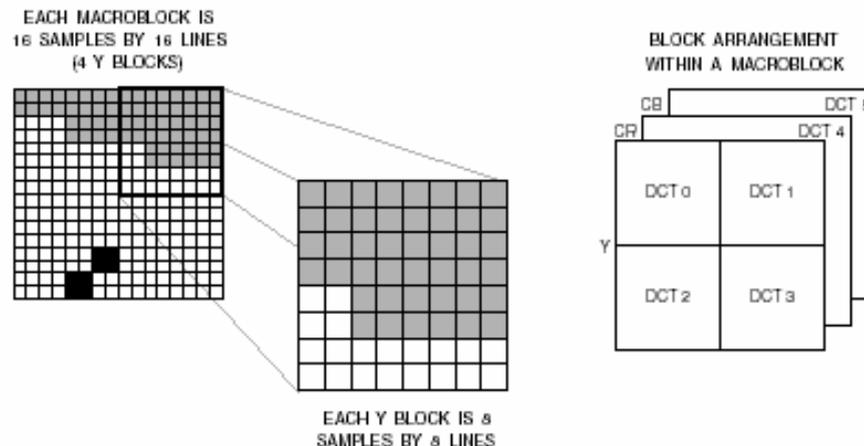
Slice: una sequenza contigua di macroblocchi disposti da sinistra verso destra, alto verso il basso.

Macroblocco: finestra di 16x16 pixel codificata come:

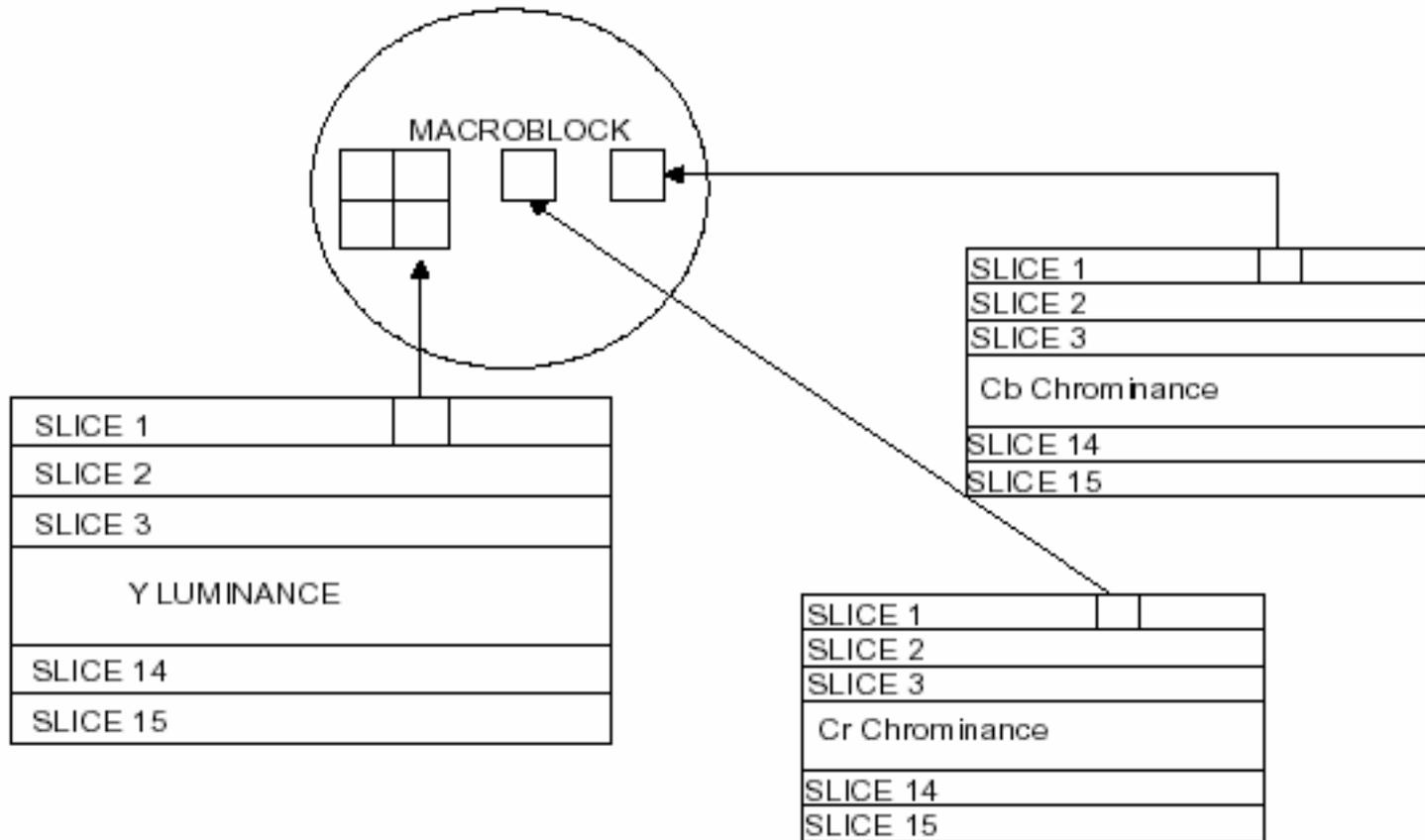
16x16 valori di luminance (Y)

8x8 valori di cromaticanza (Cb, Cr)

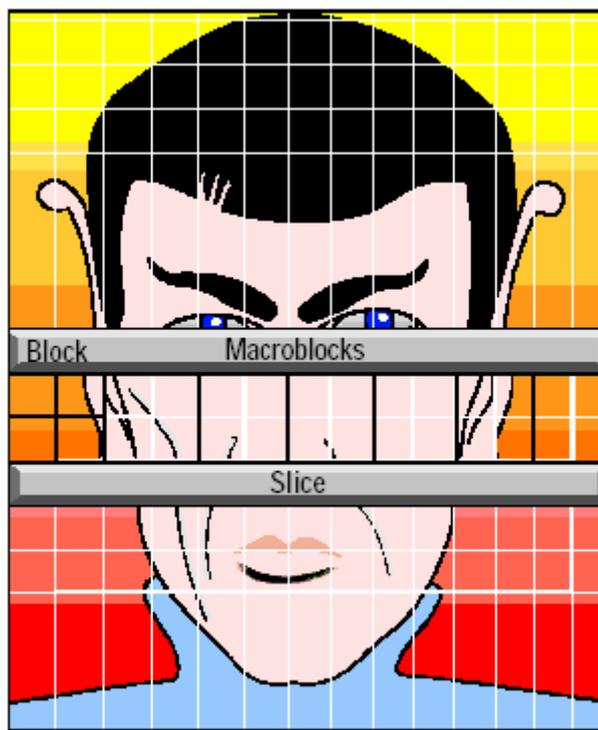
Totale di 6 blocchi 8x8 per macroblocco, più motion vectors.



Slice e macroblocchi



Blocchi, macroblocchi e slice



- Le slice servono a non far propagare troppo eventuali errori
- Non coincidono necessariamente con righe dell'immagine

Slice e macroblocchi

Ogni slice ha uno header che indica la sua posizione a partire dall'alto.

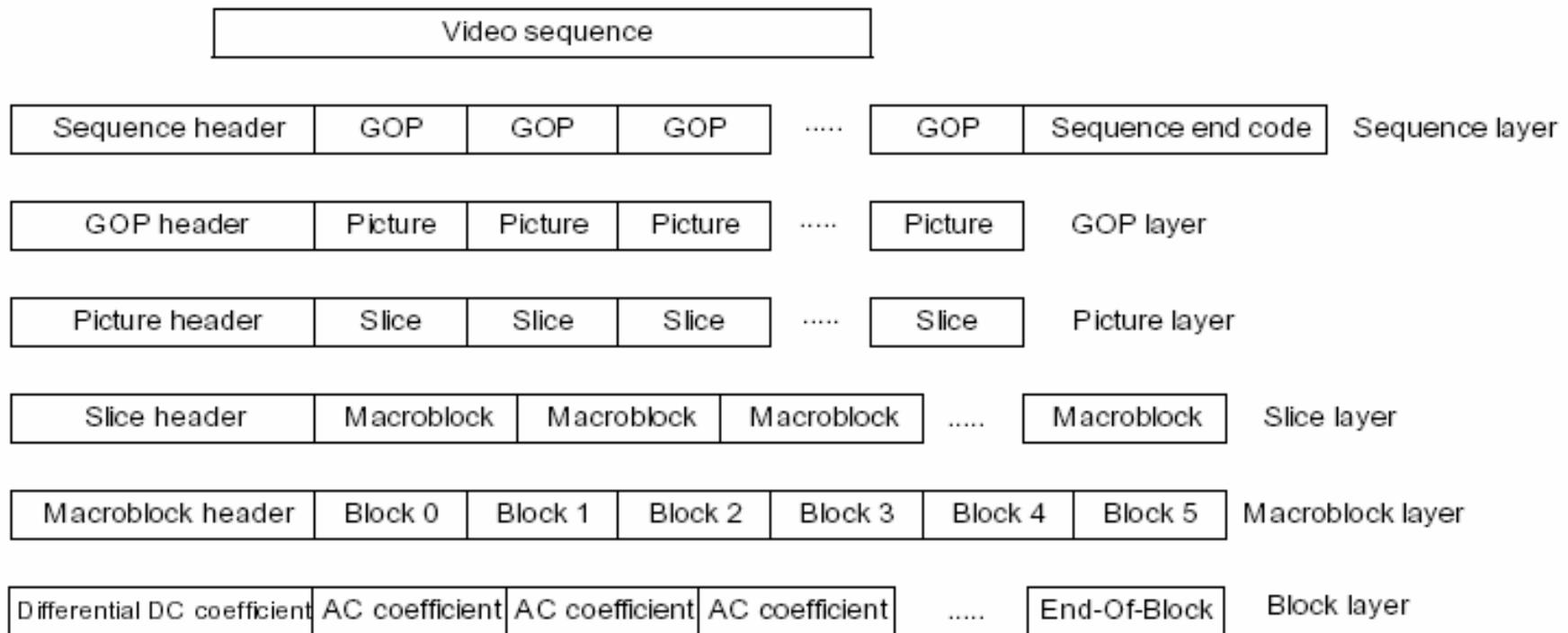
Tipi di macroblocchi: intra, forward-predicted, backward-predicted, averaged.

P-picture: usa sia macroblocchi di tipo intra che forward-predicted.

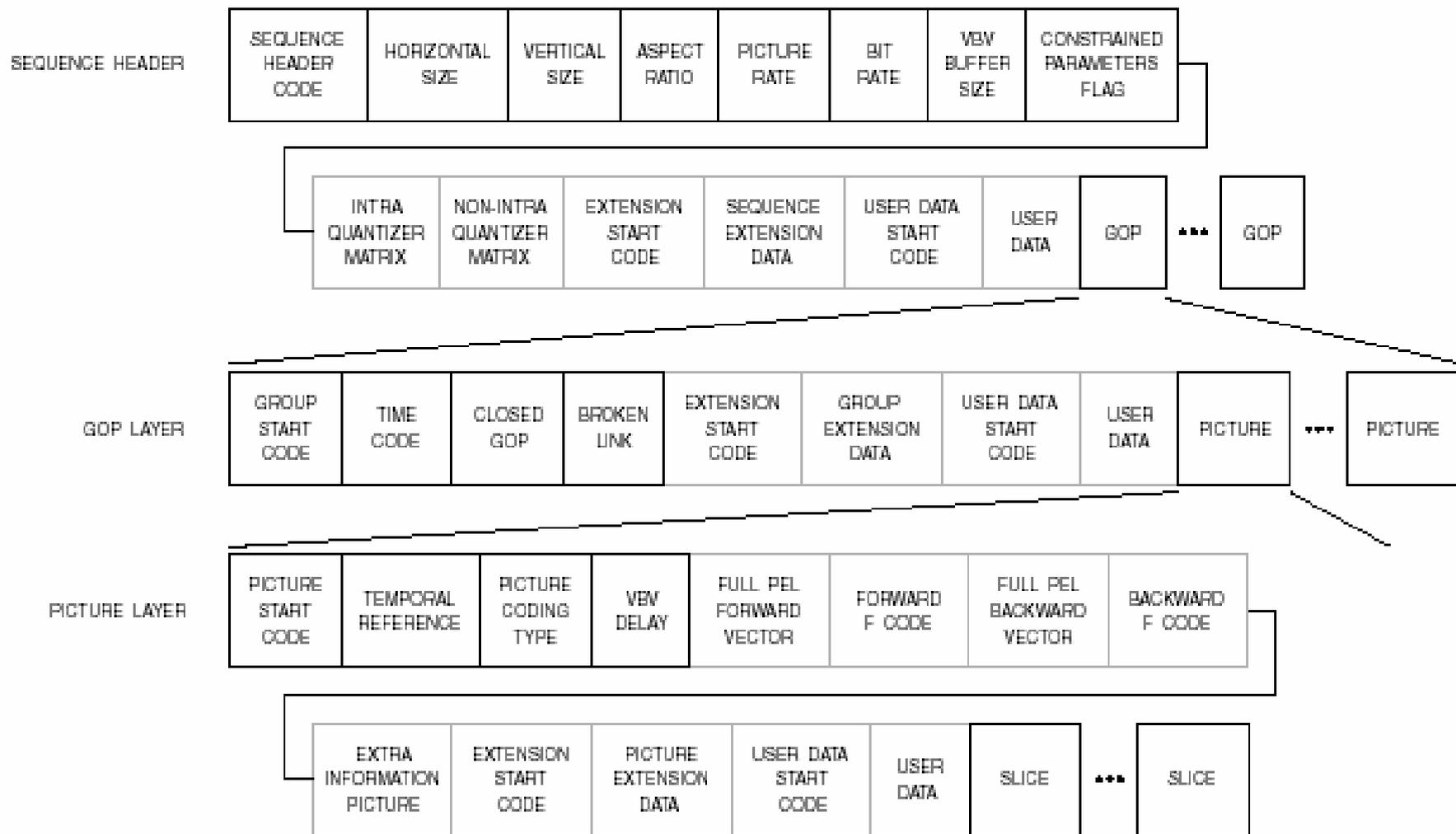
B-picture: può usare tutti e quattro i tipi di macroblocchi.

Skipped macroblock: 0 motion, copio il macroblocco dal frame precedente

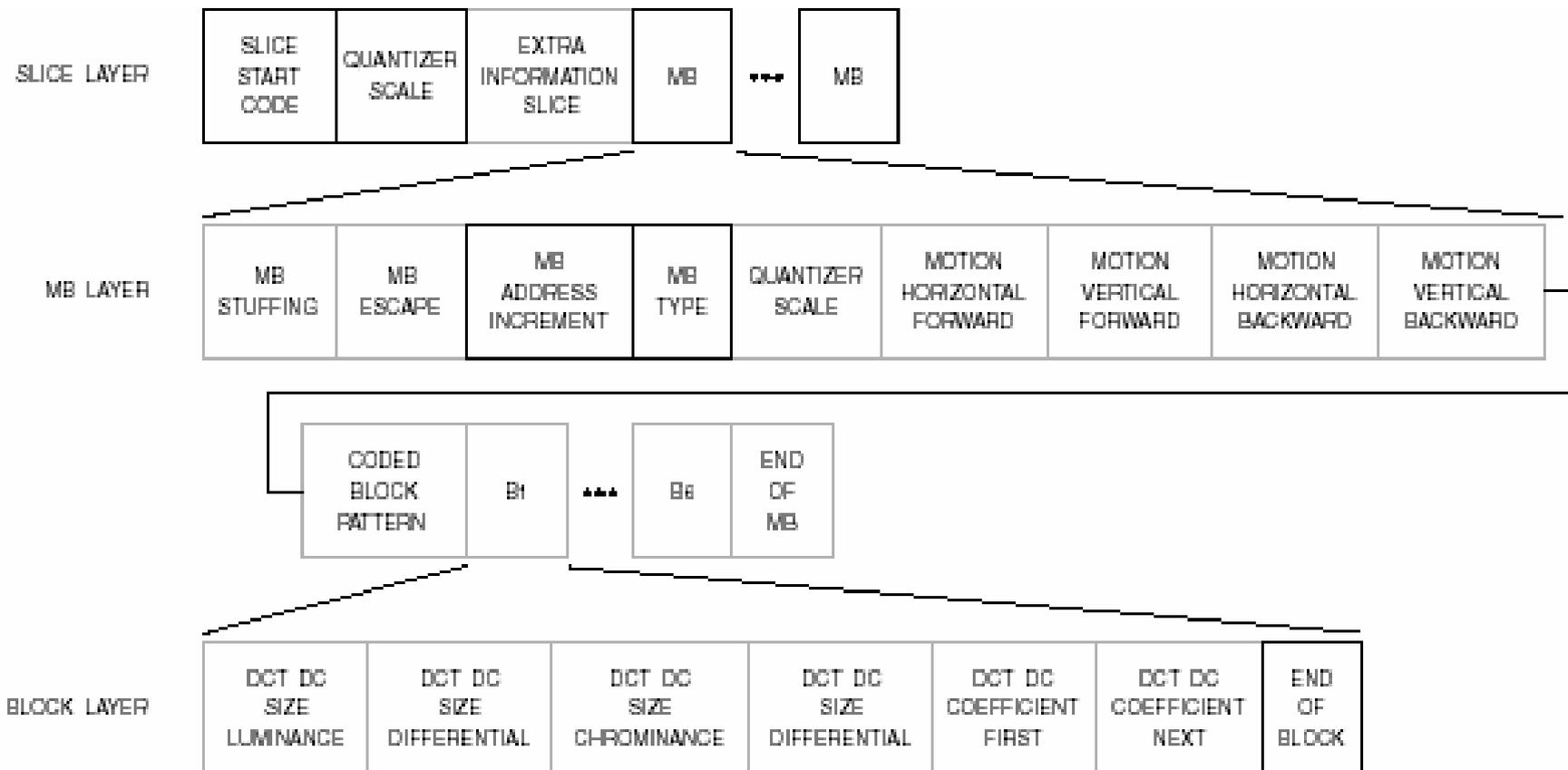
MPEG-1 video bit-stream layers



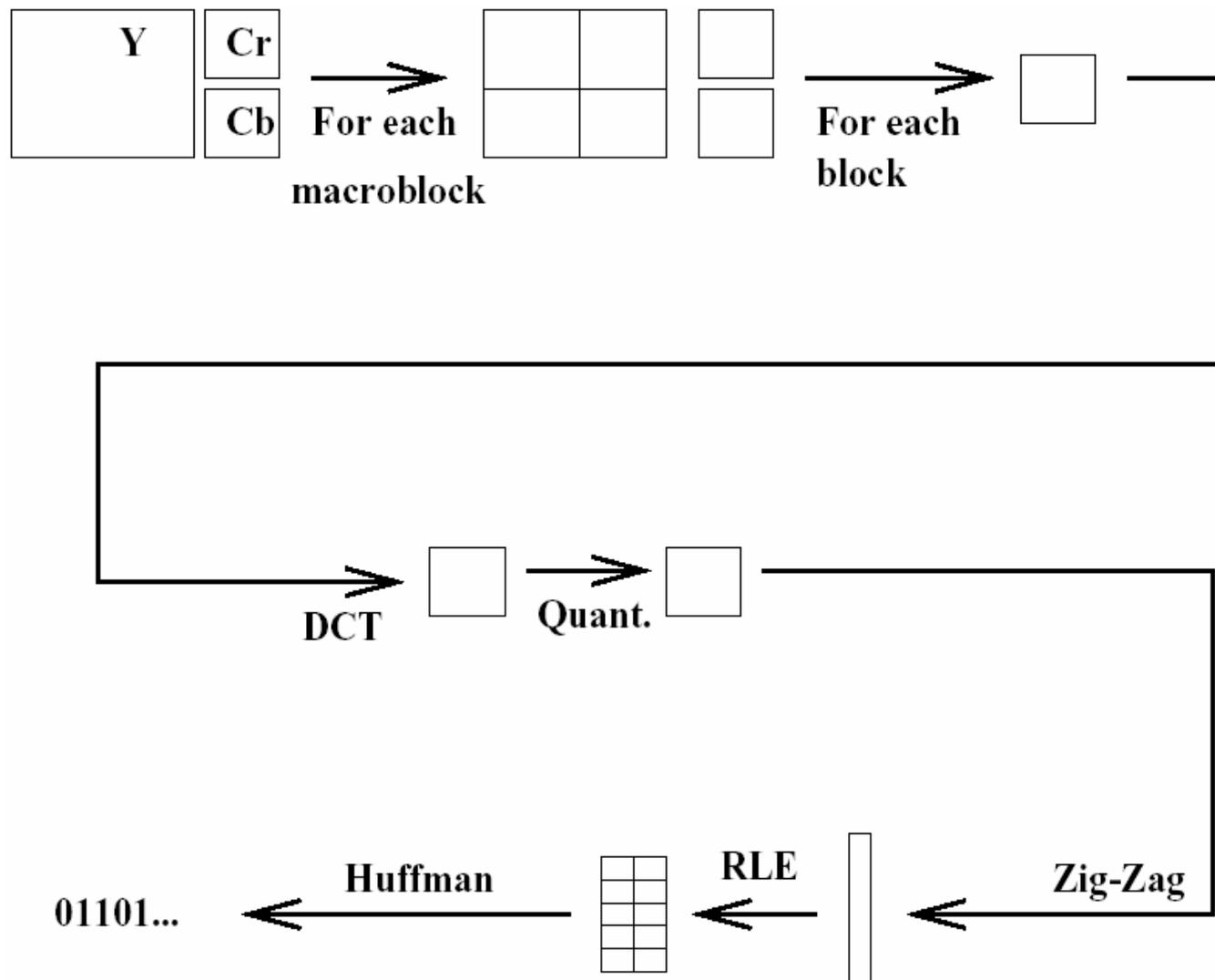
MPEG-1 video bit-stream layers: dettagli 1/2



MPEG-1 video bit-stream layers: dettagli 2/2



I-frame: passi di compressione



Compressione intra

- I blocchi che devono essere codificati intra sono processati con DCT 8x8
 - DCT e IDCT sono lossless !
- I coefficienti sono quantizzati (passo lossy)
 - Il passo di quantizzazione deriva da una matrice e da un fattore di scala (può cambiare tra macroblocchi)

Matrice di quantizzazione

8	16	19	22	26	27	29	34
16	16	22	24	27	29	34	37
19	22	26	27	29	34	34	38
22	22	26	27	29	34	37	40
22	26	27	29	32	35	40	48
26	27	29	32	35	40	48	58
26	27	29	34	38	46	56	69
27	29	35	38	46	56	69	83

- Matrice di quantizzazione di default
 - In MPEG-1 la posso cambiare per l'intera sequenza
 - In MPEG-2 la posso cambiare ad ogni picture
- Le frequenze più alte sono divise per un numero più alto: si cerca di far andare a zero il maggior numero possibile di alte frequenze
 - Usando il fattore di scala della matrice si aumenta l'effetto !

- Le matrici di quantizzazione intra e non intra possono essere cambiate, inviandole a livello di sequence layer (vedi fig. prec.)
- Il quantizer scale code può essere trasmesso a slice e macroblock layer



0110	→	1
0101	→	011
1101	→	000010

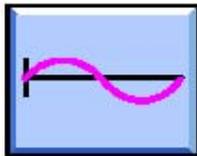
Entropy Coding



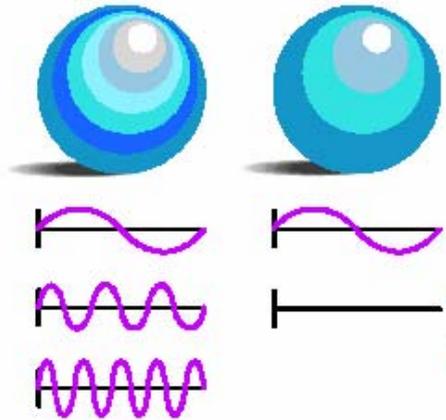
Zig
Zag
Scan

0110110	→	0110
0101100	→	0101
1101111	→	1101

Quantization



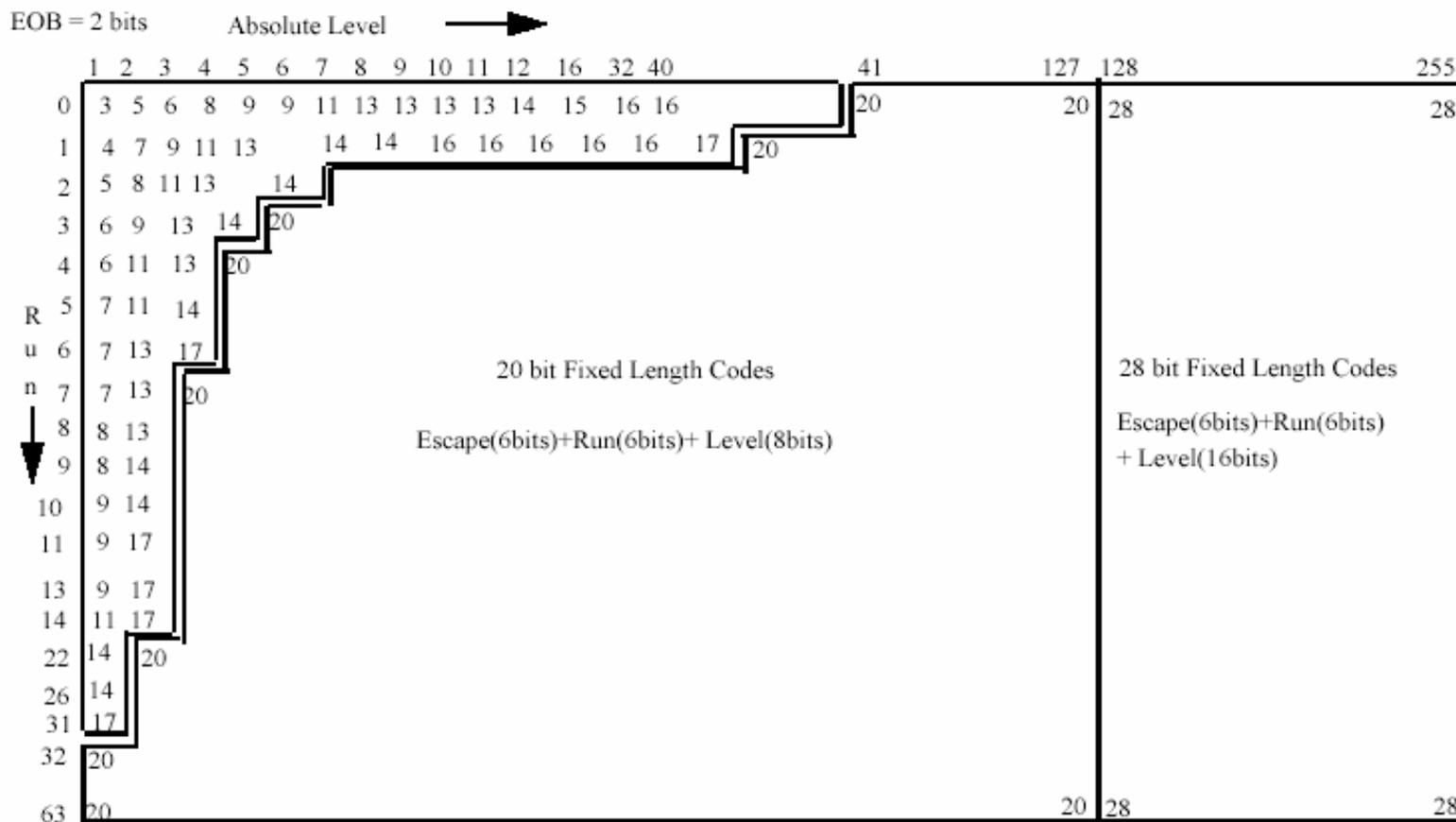
Discrete Cosine Transform



HF DCT freq
coefs are lost

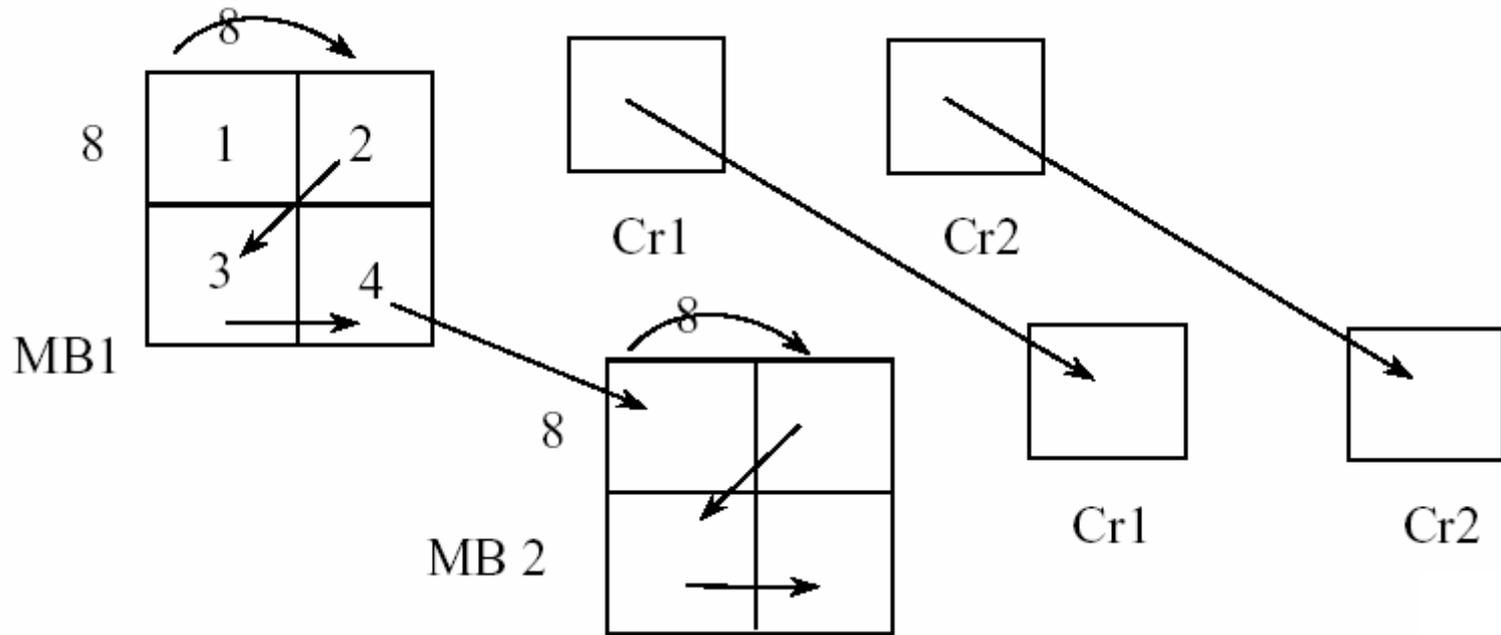
- I coefficienti sono scansionati a zig-zag creando uno stream 1D (da 2D di partenza)
 - Si producono sequenze di coefficienti di valore 0
 - Compressione lossless con *run level encoding* e Huffman (VLC: variable length coding)
 - Le combinazioni *run length* e *level* sono codificate secondo una tabella creata su base statistica
 - Se la combinazione non è presente in tabella si usa un carattere di escape e poi si codifica la sequenza

Dimensioni VLC



Predizione DC

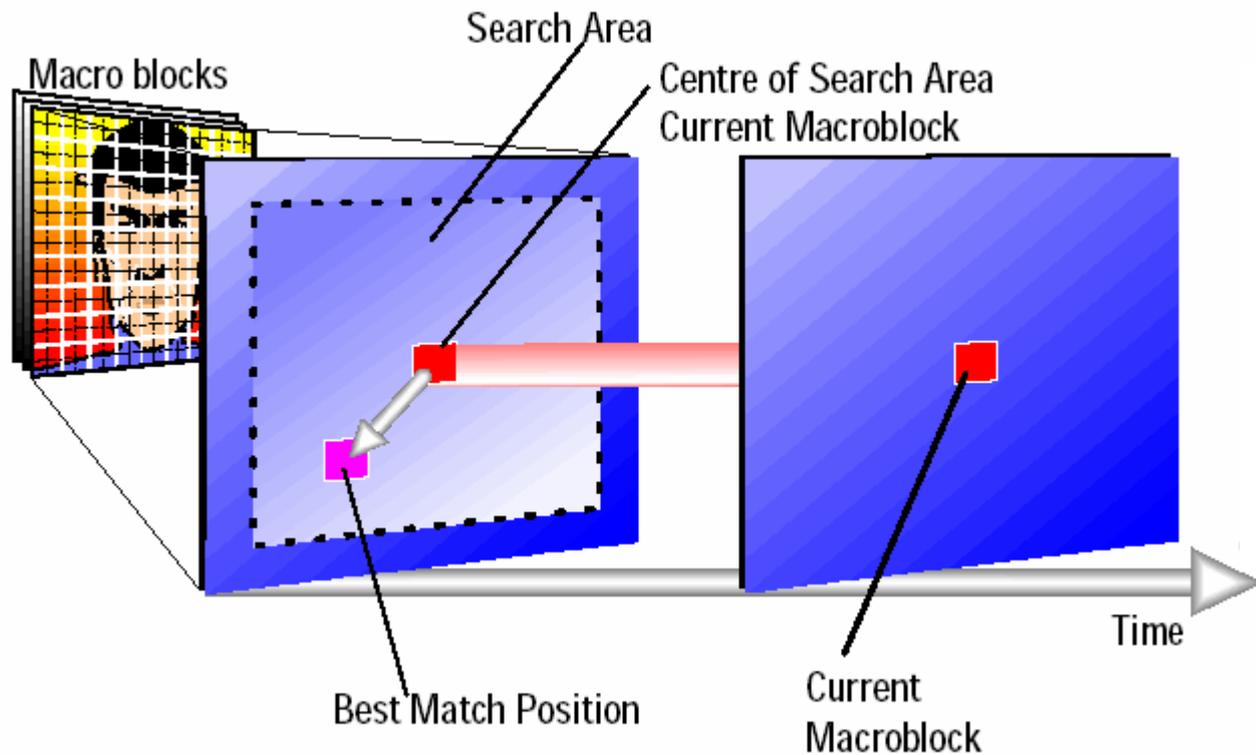
- Per i coefficienti DC si codificano le differenze tra blocchi all'interno del macroblocco
 - Ad inizio slice il valore di previsione DC è 1024
 - Tabelle standard (diverse per Y e CbCr), è un VLC



Compressione I-frame: riassunto

- Sono usate le seguenti tecniche:
 - Subsampling (per la crominanza)
 - Transform coding (DCT + zig-zag)
 - RLE (coefficienti AC)
 - Predictive coding (DPCM per coefficienti DC)
 - Entropy coding (Huffman)
 - Quantizzazione (sia coeff. DC che AC)

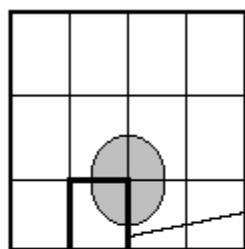
- È la stima del moto che riduce veramente la dimensione del filmato.
 - Le tecniche precedenti sono leggermente meglio di JPEG



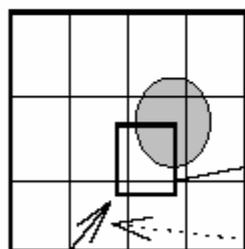
P-frame : passi di compressione

target image

(new image)

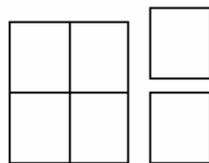


match window



motion vector

Difference



Huffman Coder

0100110

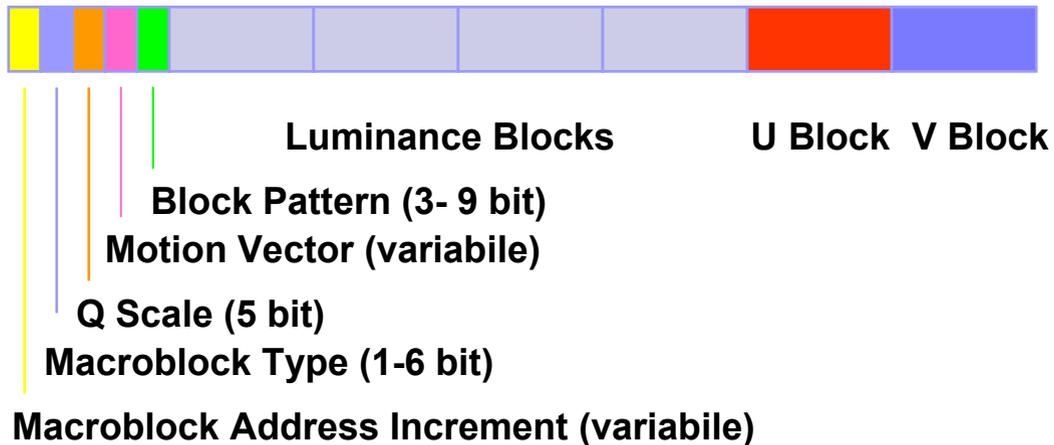
reference image

(previous image)

Codifica P-frame: scelta tipo blocchi

- È compito dell'encoder scegliere se codificare un macroblocco come intra o come predetto
- Un possibile meccanismo compara la varianza della componente luminanza del macroblocco originale con il macroblocco errore
 - Se la varianza dell'errore predetto è maggiore il macroblocco è codificato intra

P-Macroblock



Macroblock Type determina se esistono Q Scale, Motion Vector, o Block Pattern.

In un macroblocco P uno o anche tutti I blocchi possono essere assenti. Block pattern indica quali blocchi sono presenti.

Address Increment

- ogni macroblocco ha un indirizzo.
 - $MB_WIDTH = \text{luminance width} / 16$
 - $MB_ROW = \# \text{ riga pixel alto a sx} / 16$
 - $MB_COL = \# \text{ col. pixel alto a sx} / 16$
 - $MB_ADDR = MB_ROW * MB_WIDTH + MB_COL$
- Il decoder mantiene l'indirizzo del macroblocco precedente $PREV_MBADDR$.
 - Impostato a -1 all'inizio del frame.
 - Impostato a $(SLICE_ROW * MB_WIDTH - 1)$ all'inizio dell'header dello slice.
- L'indirizzo dell'incremento del MB è sommato a $PREV_MBADDR$ per dare l'indirizzo del MB corrente.

Address Increment Coding

- È codificato con Huffman, secondo tabella predefinita:
 - 33 codici (1-33).
 - 1 il più piccolo (1-bit)
 - 33 il più grande (11-bit)
 - 1 codice di ESCAPE
 - ESCAPE significa: aggiungi 33 al codice di incremento indirizzo che segue.
 - Si possono usare più ESCAPE in sequenza per codificare distanze grandi.
- Questa codifica è usata anche per gli I-frame.

MB Type

- Codificato con Huffman.
 - 7 codici possibili (1 - 6 bit)
- Determinano le seguenti caratteristiche del MB:
 - Intra o non-intra.
 - Q scale specificato o no.
 - Motion vector esistente o no.
 - Block pattern esistente o no.
- Non tutte le combinazioni sono possibili.
- C'è una tabella nello standard che indica i codici.

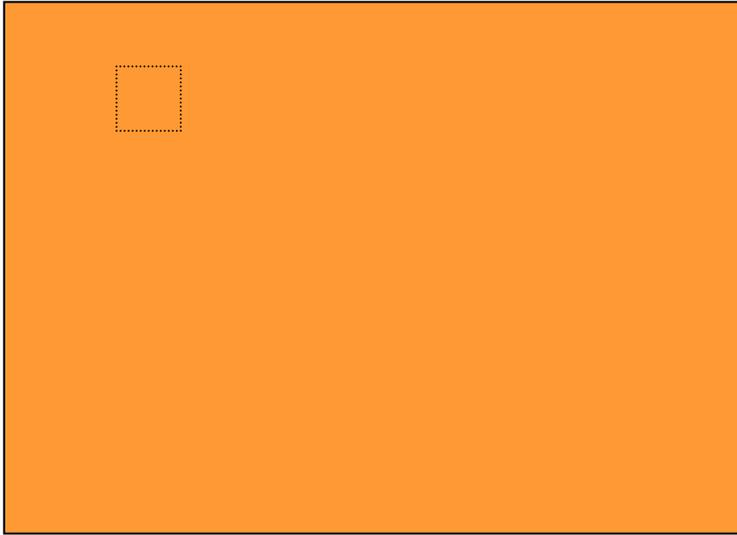
Quantization Scale

- 5 bit.
- Zero è illegale.
- Codifica valori tra 1 e 31 che sono interpretati come valori tra 2 e 62 per il fattore Q Scale.
 - Non si codificano valori dispari.
- Il decoder mantiene il q-scale corrente.
 - Se non è specificato continua ad usare il q-scale corrente.
 - Altrimenti rimpiazza il q-scale e lo usa come nuova base di q-scale.

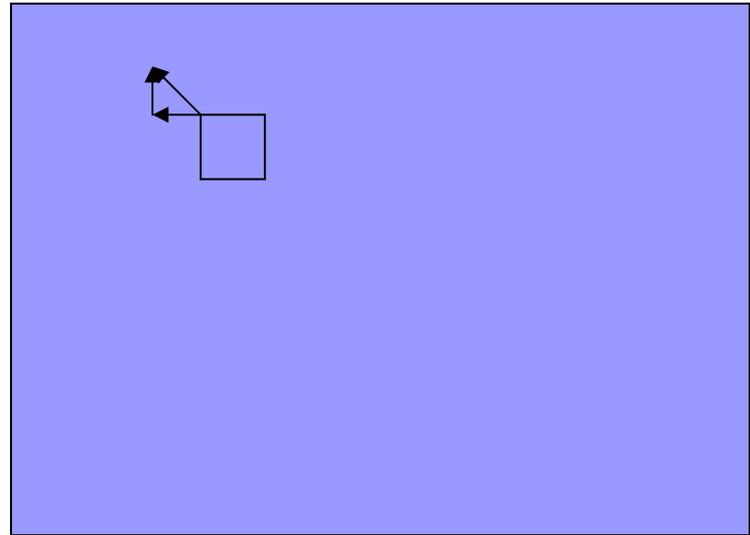
Motion Vector

- Due componenti:
 - Offset orizzontali e verticali.
 - L'offset è calcolato a partire dal pixel in alto a sx. del MB.
 - Valori positivi indicano in alto e a dx.
 - Valori positivi indicano in basso e a sx.
- Il motion vector è usato per definire una base predittiva per il MB corrente partendo dall'immagine di riferimento.

Motion Vector



I- o P- Frame di riferimento (già decodificato)



P-Frame

NON è necessario che la base predittiva sia allineata secondo I MB.
Si può avere un vettore di moto con precisione di mezzo pixel, in questo caso la base predittiva usata è ottenuta per interpolazione (bilineare).

Motion Vector Encoding

- Se non è specificato il motion vector allora lo si considera come $(0,0)$.
- Prima si indica la componente orizz. e poi la verticale.
- Si usa la predizione anche per i vettori di moto.
 - Impostata a $0,0$ all'inizio di frame o di slice o di MB tipo I.
 - La differenza tra predittore e valore del vettore è codificata con Huffman.

Predictive Base

- I MB di tipo P hanno sempre una base predittiva:
 - Scelta secondo il vettore di moto (che indica un'area) oppure...
 - ... se non c'è vettore di moto (implicitamente pari a 0,0) la base predittiva è lo stesso MB nel fotogramma di riferimento.

Block Pattern

- Lo scopo della compensazione di moto è quello di trovare una base predittica che assomigli il più possibile al MB in analisi.
 - Se il match è particolarmente buono allora non dobbiamo neanche indicare la differenza da codificare !.
- Il Block pattern indica quali blocchi hanno un errore abbastanza grande da dover essere codificato.
- Se non c'è il block pattern allora significa non abbiamo bisogno di codificare nulla

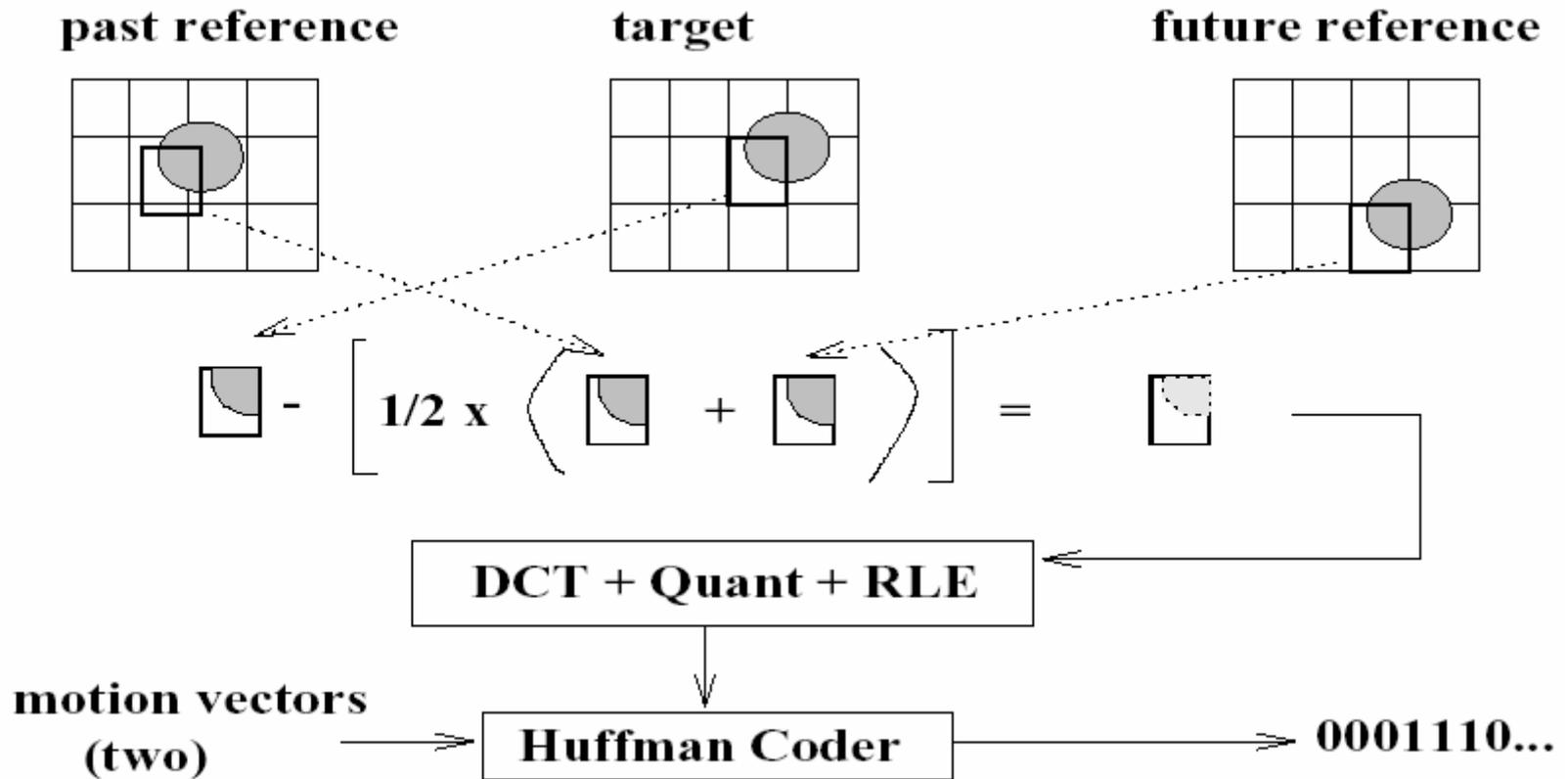
Error Block Encoding

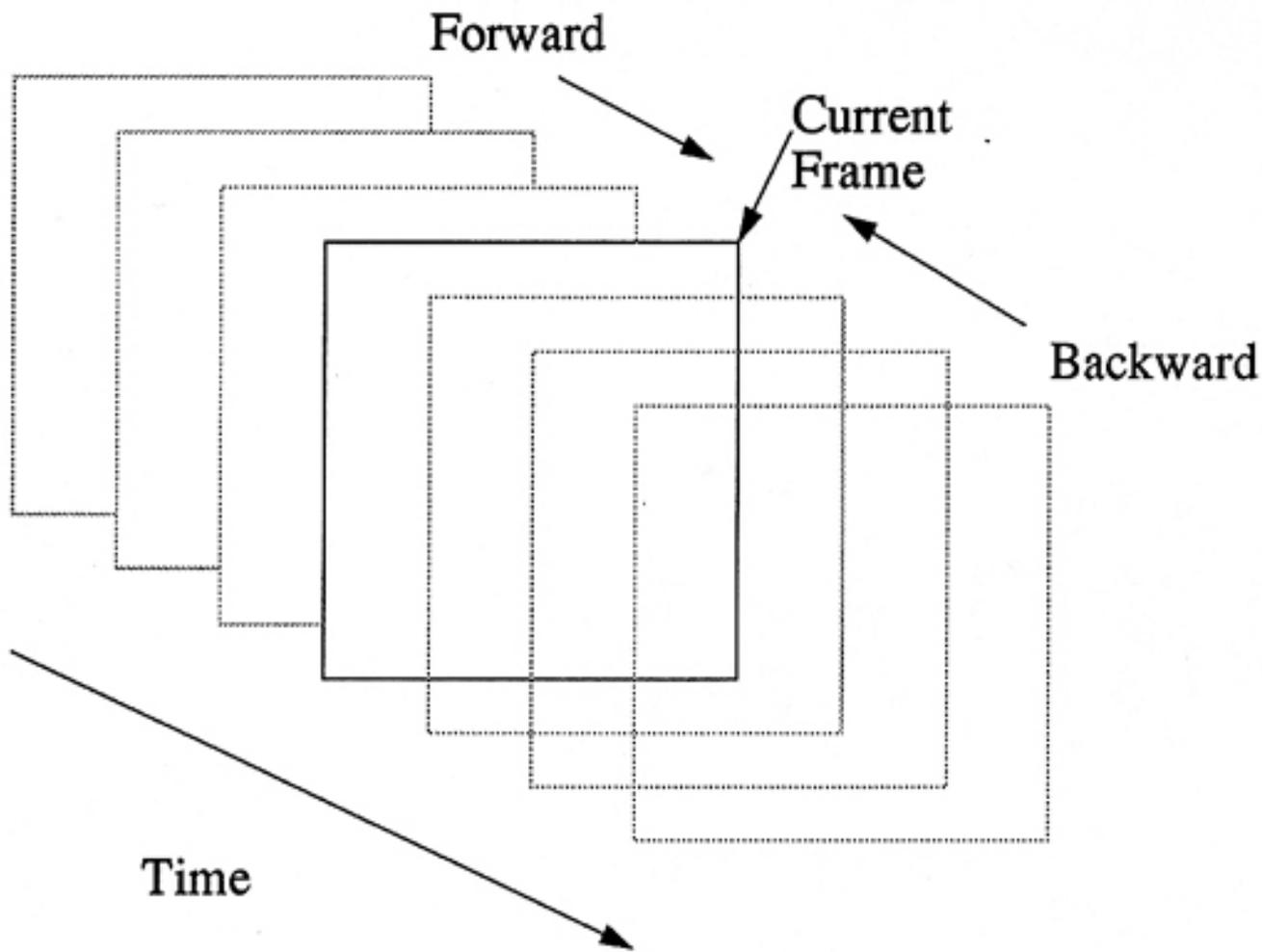
- La differenza tra blocco di riferimento e codificato è trattata come un blocco normale, spesso definito come error block
- Si usa una matrice di quantizzazione diversa rispetto agli I-block:
 - Ha il valore “16” in tutte le posizioni.
 - Il fatto è che gli Error block hanno molta informazione in alta frequenza.
 - Non c'è una buona correlazione percettiva tra le frequenze dell'error coding ed eventuali artefatti di compressione.
- La componente DC è trattata come le AC:
 - Non si usa differential encoding rispetto ad un predittore.
- Al solito I termini sono codificati RLE dopo scansione zig-zag e quindi codificati con Huffman.

Aggiornamento predittori

- I predittori DC sono resettati quando è incontrato un MB tipo P o skipped.
- I predittori dei motion vector sono resettati quando si incontra un MB di tipo I

B-frame : passi di compressione



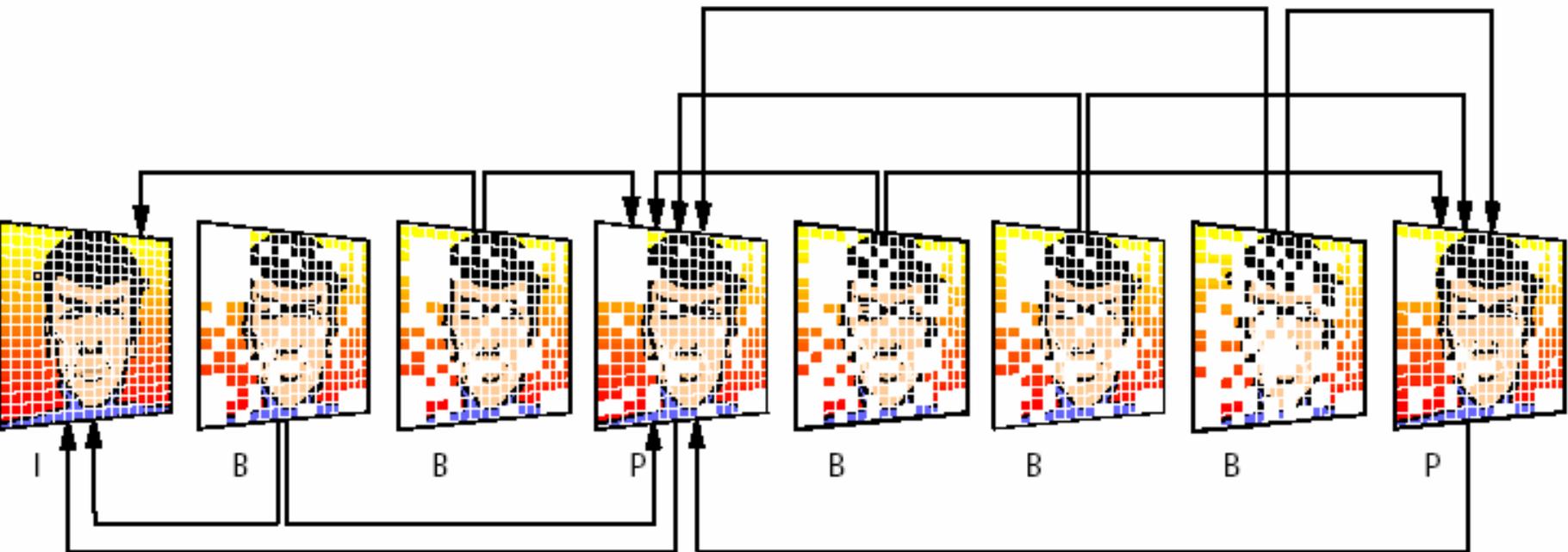


Tipo di macroblocchi

- I b-frame possono avere macroblocchi di tipo:
 - I
 - P: codificato da frame riferimento precedente
 - B: codificato da frame riferimento successivo
 - Bi: codificato da frame riferimento prec. e succ. (prima definiti come averaged)

Codifica B-frame: scelta tipo blocchi

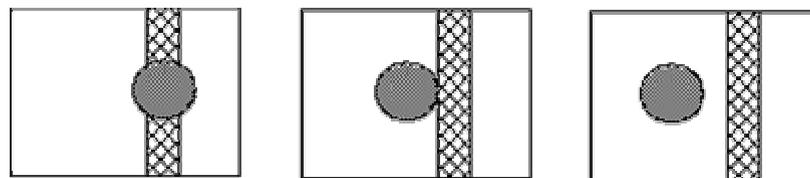
- Il medesimo meccanismo di scelta dei macroblocchi dei P-frame può essere usato per scegliere il tipo di previsione per i macroblocchi dei frame B



- I-frame: contiene l'intera immagine; P-frame si basa su I o P-frame precedente; B-frame usa I o P passate e future

Predizione

- In alcuni casi non si può predire il contenuto a partire dalla scena precedente
 - Es. una porta si apre mostrando un'altra stanza



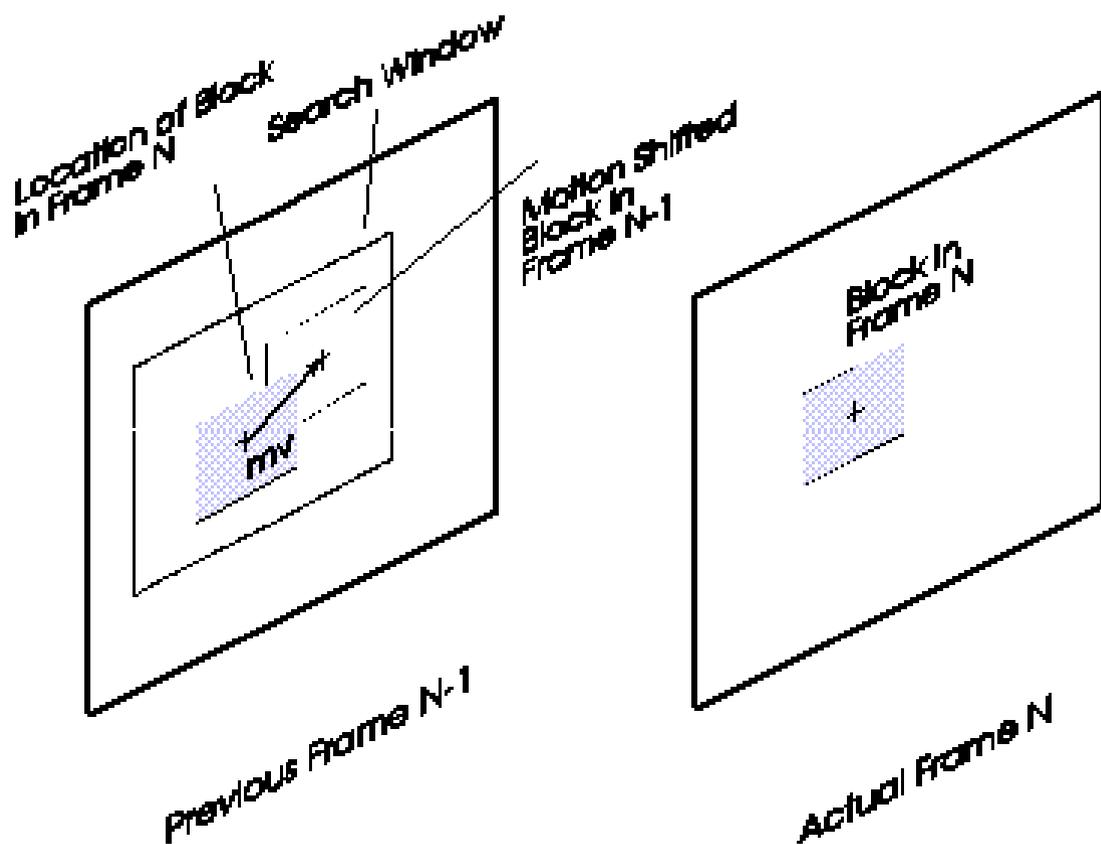
- Nel caso di un P-frame si codifica il macroblocco come se fosse parte di I-frame (I-coding)
- Nel caso di B-frame si potrebbe usare sia I-coding che backward prediction, che usa il futuro I o P frame

- Motion estimation: stimo il block matching migliore
- Motion compensation: codifico i frame sulla base della stima, trasmettendo la differenza tra blocchi di cui ho stimato il matching

Motion Compensation

- Approccio per il match tra blocchi nella compensazione di moto:

- il match è cercato all'interno di una finestra predefinita.



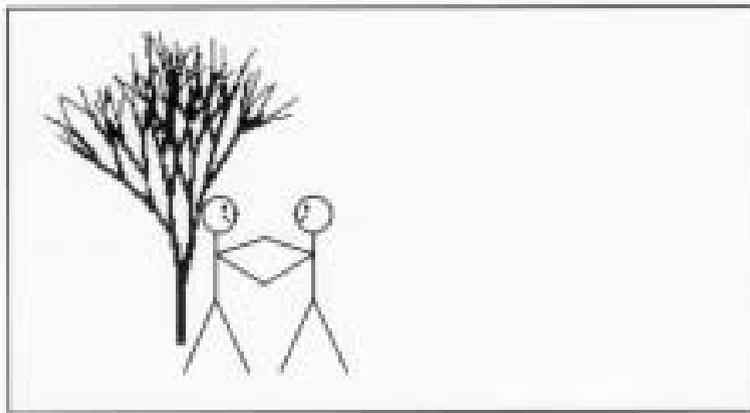
■ L'idea è che:

- In una sequenza piccola di solito un oggetto rimane fermo oppure si allontana di poco
- Il moto è descritto come un vettore 2D che dice dove ritrovare un macroblocco presente in un frame decodificato precedentemente

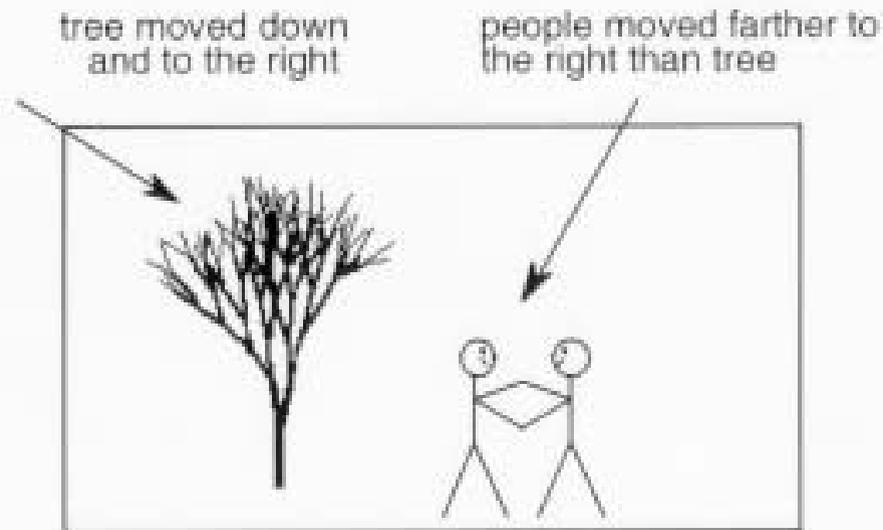
- È un problema di ricerca
 - Uso i dati di luminanza
- Dobbiamo valutare qual è il miglior match
- È la causa della buona compressione di MPEG
- È la causa dell'asimmetria:
 - La codifica è molto più difficile della decodifica

Motion estimation

- Due frame consecutivi:
 - il frame 2 viene codificato utilizzando la stima del moto sul frame 1.



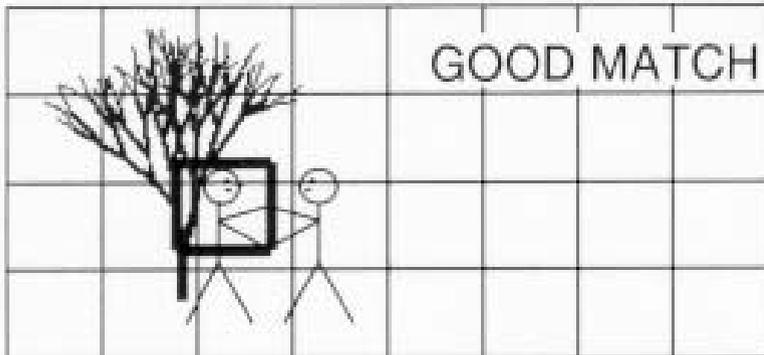
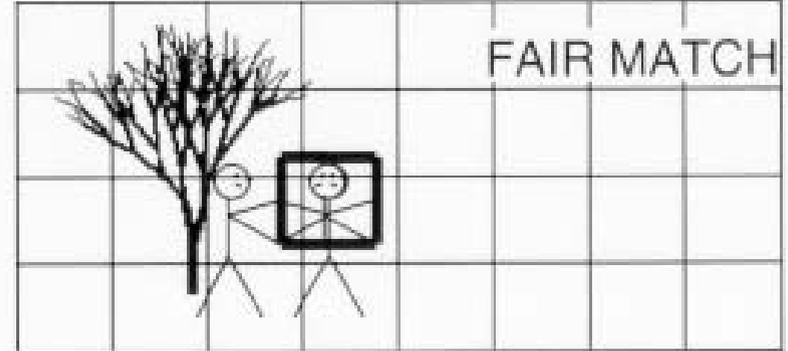
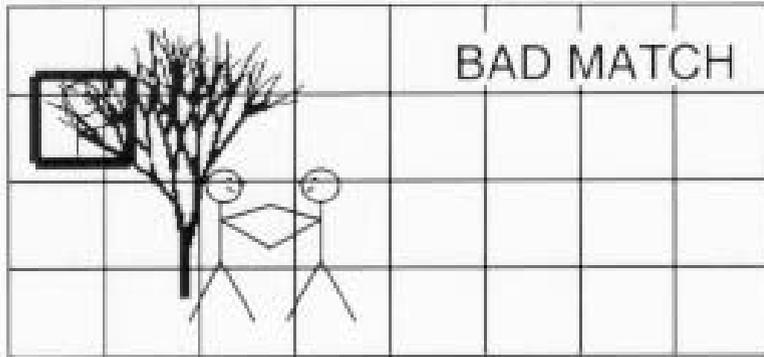
FRAME 1



FRAME 2

Match tra macroblocchi

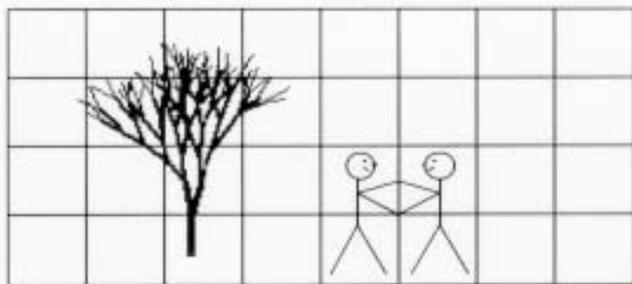
- Diversi match possibili per il macroblocco da codificare: prendo il migliore



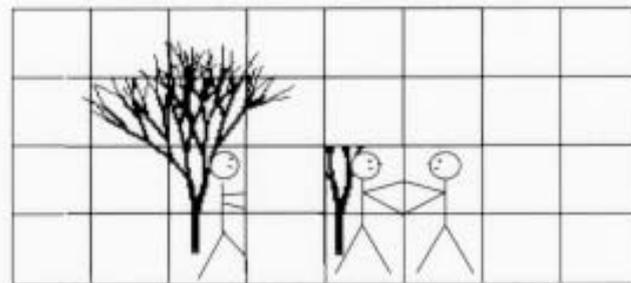
Macroblock to be coded

Frame di errore residuo

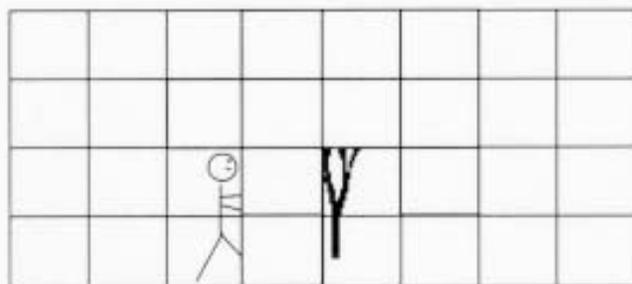
- Frame 2 predetto attraverso la stima del moto: sottraggo il frame predetto dal desiderato



Desired Picture



Minus Predicted Picture



Residual Error Picture
(Coded & Transmitted)

- Il frame errore (si spera poco complesso) è codificato e trasmesso.

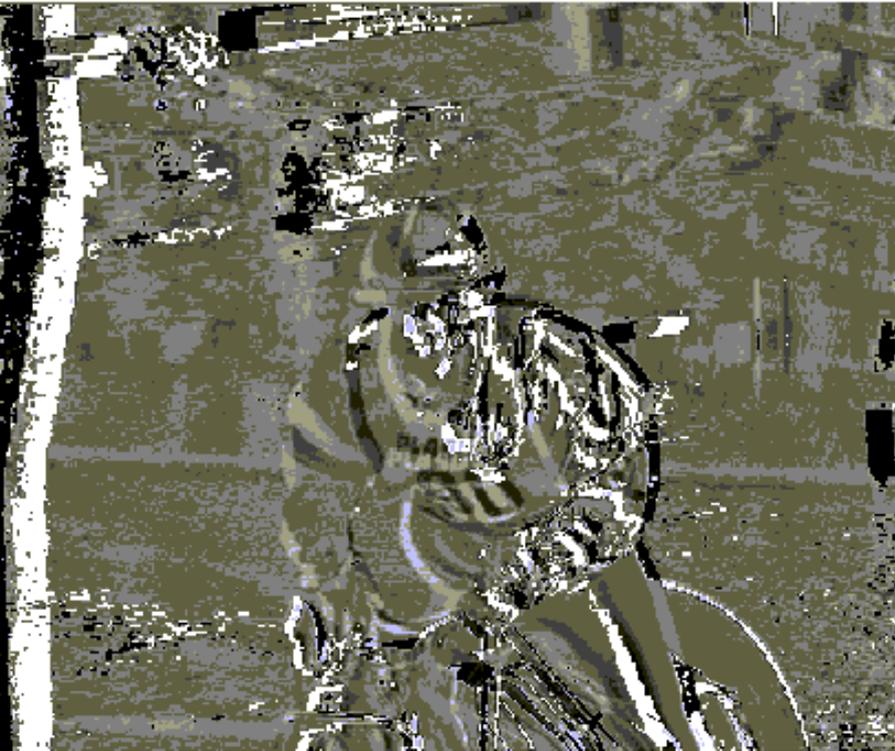
Motion Compensation



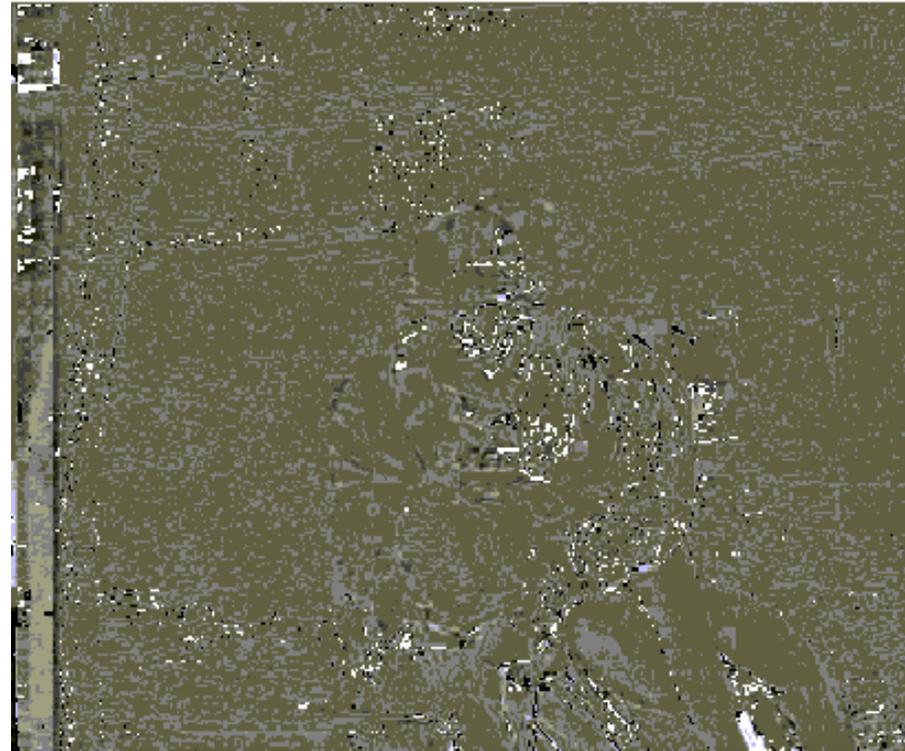
- Frame N da codificare.

- Frame all'istante N-1 usato per la predizione del contenuto del frame N.

Motion Compensation



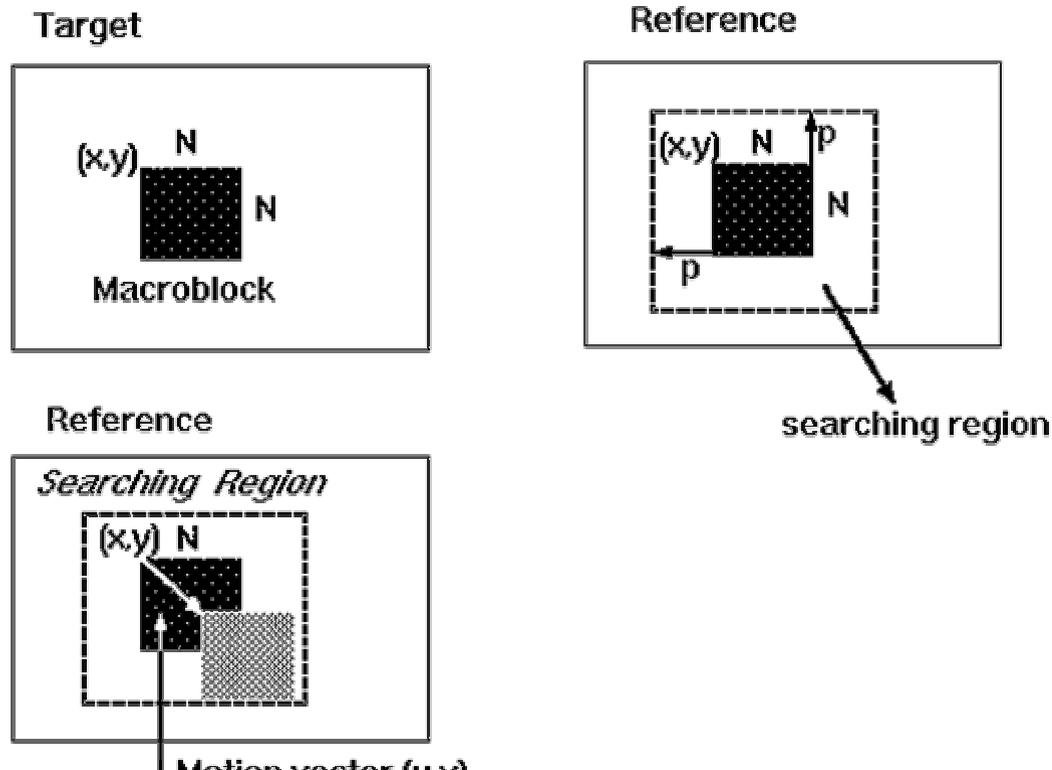
- Immagini di errore di predizione senza motion compensation.



- Immagini di errore di predizione da codificare con motion compensation.

Il block matching

- Esistono diverse tecniche per il block matching
 - Spesso si limita l'area in cui cercare il match



■ Possiamo scegliere tra varie:

- Tecniche per la determinazione del match tra blocchi
- Strategie di ricerca dei blocchi
- Scelte dimensione dei blocchi

- Mean squared error:

- MSE per un blocco $N \times N$:

$$\text{MSE} = \frac{1}{N^2} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} (C_{ij} - R_{ij})^2$$

dove C_{ij} è il campione del blocco corrente e R_{ij} il campione del blocco di riferimento

MSE: esempio

1	3	2
6	4	3
5	4	3

Current block

1	3	2	4	5
6	4	2	3	2
5	4	2	2	3
4	4	3	3	1
4	6	7	4	5

Reference area

$(-1,1)$	$(0,1)$	$(1,1)$
$(-1,0)$	$(0,0)$	$(1,0)$
$(-1,-1)$	$(0,-1)$	$(1,-1)$

Positions (x,y)

- L' MSE tra il blocco corrente e la stessa posizione (posizione (0, 0)) sul riferimento è dato da:

$$\{(1 - 4)^2 + (3 - 2)^2 + (2 - 3)^2 + (6 - 4)^2 + (4 - 2)^2 + (3 - 2)^2 + (5 - 4)^2 + (4 - 3)^2 + (3 - 3)^2\} / 9 = 2.44$$

- Tutti i valori sono:

Position (x, y)	(-1, -1)	(0, -1)	(1, -1)	(-1, 0)	(0, 0)	(1, 0)	(-1, 1)	(0, 1)	(1, 1)
MSE	4.67	2.89	2.78	3.22	2.44	3.33	0.22	2.56	5.33

- Mean absolute error/difference (MAE/MAD)
 - È più facile/veloce da calcolare rispetto a MSE ed è ancora un'approssimazione ragionevole provides a reasonably good approximation of residual energy and is easier to calculate than MSE

$$\text{MAE} = \frac{1}{N^2} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |C_{ij} - R_{ij}|$$

■ Matching pel count (MPC)

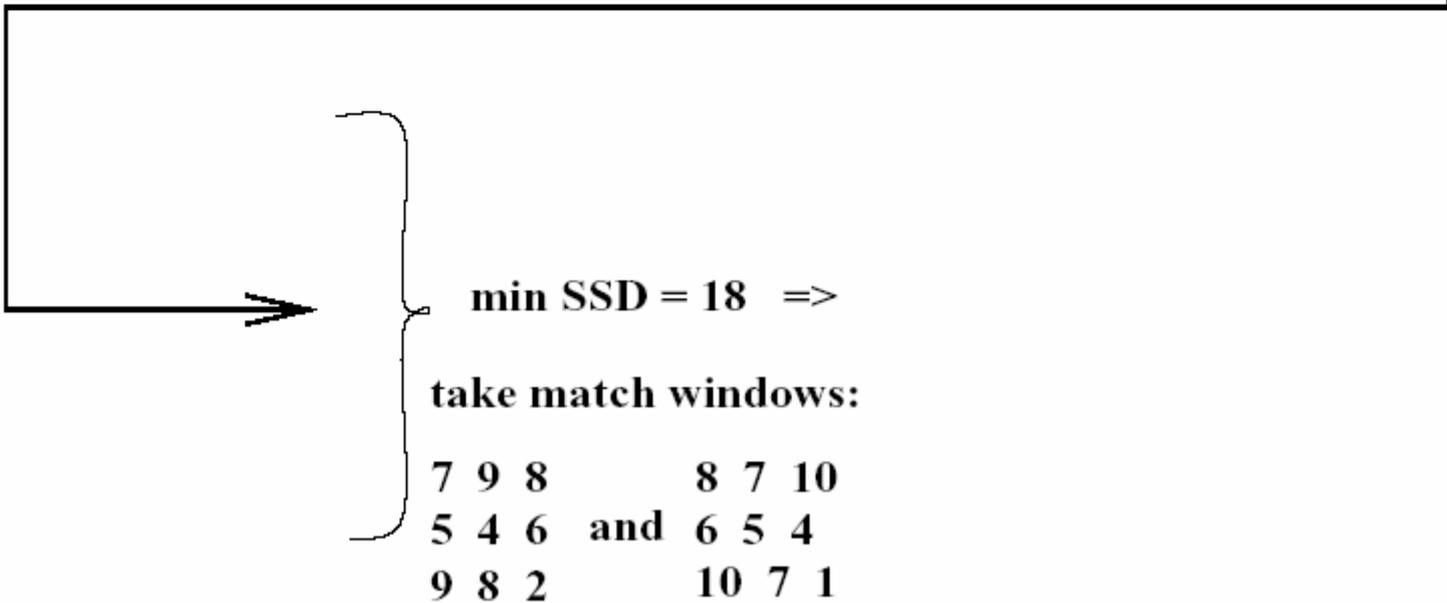
- Conto il numero di pixel simili in due blocchi
- Devo introdurre una metrica ed una soglia per valutare la distanza tra due pixel

- Sum of Squared Differences (SSD)

$$SSD = \sum_i (x_i - y_i)^2$$

7 9 8		8 7 9		⇒	SSD=	$(7-8)^2 + (9-7)^2 + (8-9)^2$	}
5 4 6	versus	7 5 4				$(5-7)^2 + (4-5)^2 + (6-4)^2$	
9 8 2		7 5 4				$(9-7)^2 + (8-5)^2 + (2-4)^2$	
						$= 1 + 4 + 1 + 4 + 1 + 4 + 4 + 9 + 4$	}
						$= 32$	

7 9 8		8 7 10		⇒	SSD = 18
5 4 6	versus	6 5 4			
9 8 2		10 7 1			



- Sum of absolute errors (SAE) o sum of absolute differences (SAD)

$$\text{SAE} = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |C_{ij} - R_{ij}|$$

rispetto a SSD è meno sensibile ad outlier: un solo punto molto diverso mi rende il valore di SSD molto grande

SSD vs. SAD

SSD: 7 9 8 8 7 10
 5 4 6 versus 6 5 4 -> **SSD = 18**
 9 8 2 10 7 1

 7 9 8 8 7 10
 5 4 6 versus 6 5 4 -> **SSD = 40,017**
 9 8 2 10 7 **202** **Outlier**

SAD:

 7 9 8 8 7 10
 5 4 6 versus 6 5 4 -> **SAD = 211**
 9 8 2 10 7 202

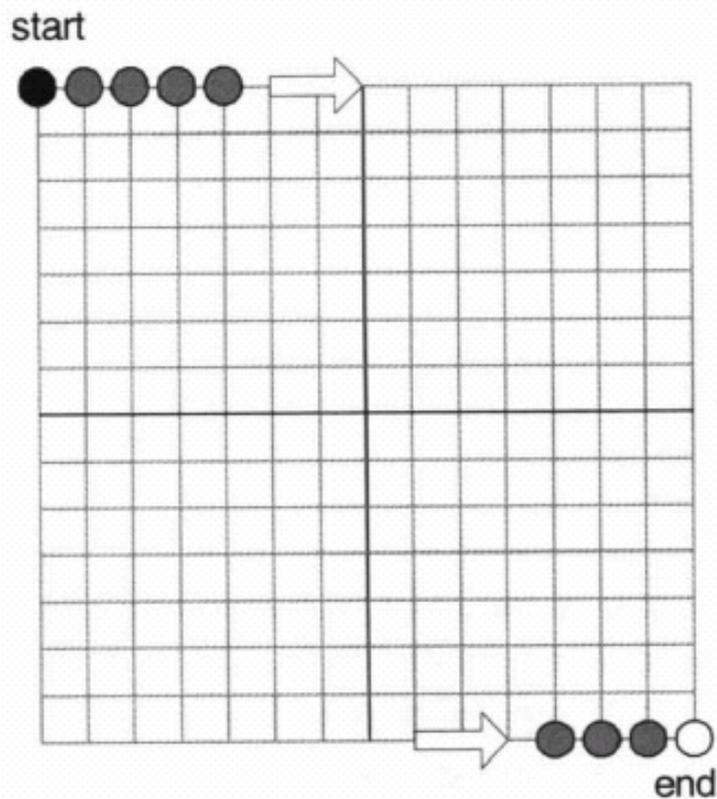
Algoritmi di ricerca

- La dimensione della finestra di ricerca dipende da diversi fattori:
 - Risoluzione dei frame
 - Potenza di calcolo disponibile
 - Tipo di scena

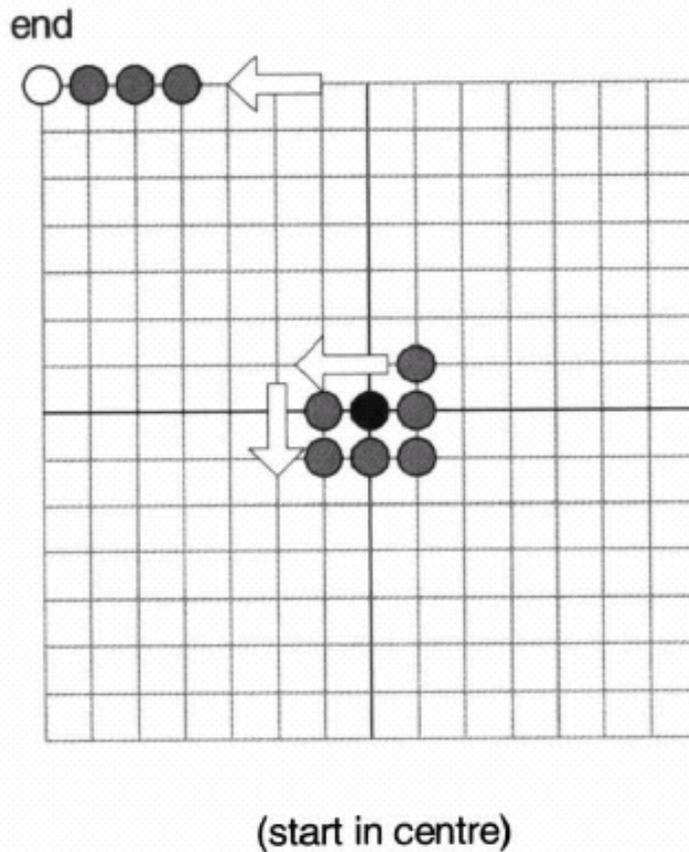
Full search

- Usando il mio criterio di comparazione (es. SAE/SAD) cerco in tutte le possibili posizioni della finestra
 - È computazionalmente costosa, adatta per implementazioni h/w

Full search (cont.)

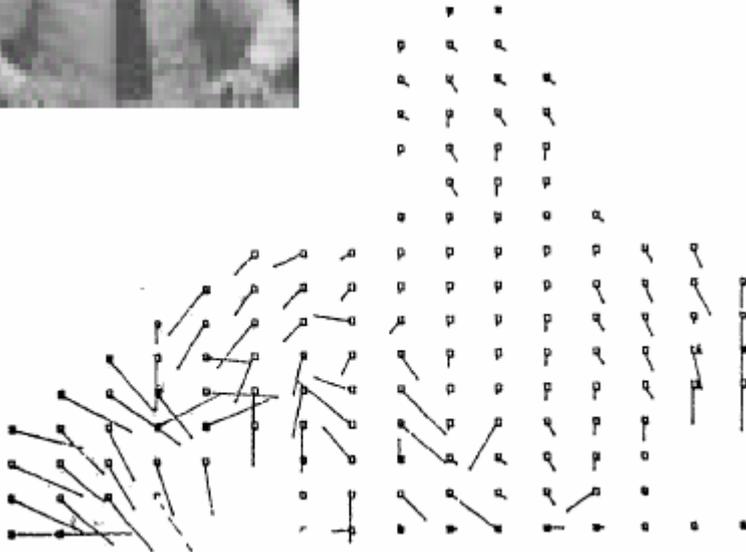


(a) Raster order

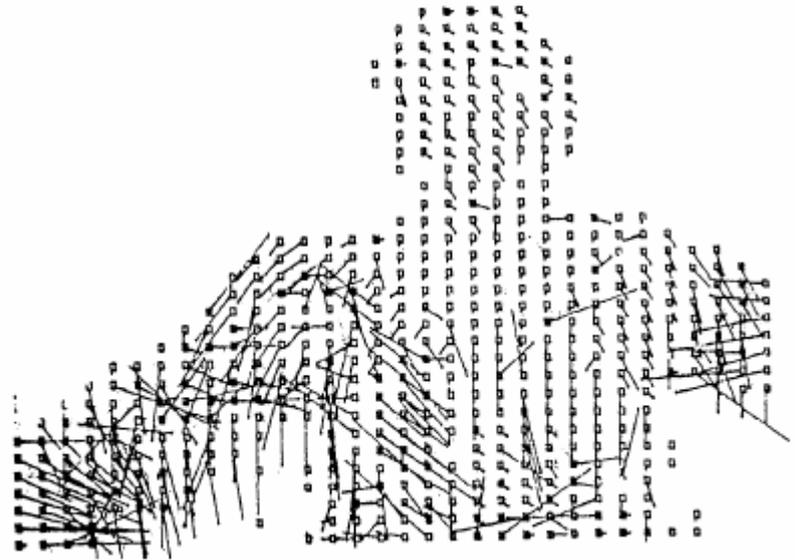


(b) Spiral order

Full search



Blocchi grandi



Blocchi piccoli

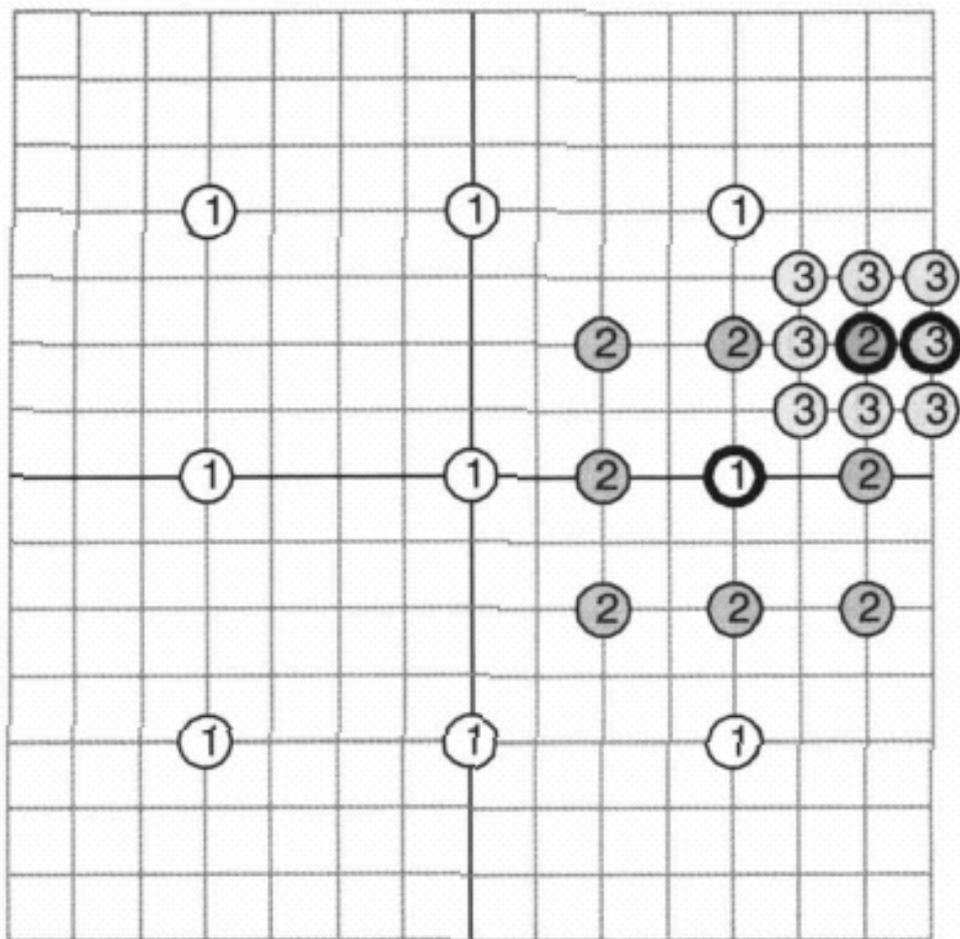
Ricerca veloce

- Cerco di ridurre il numero di comparazioni rispetto a quelle necessarie per full search
 - Con full search trovo il minimo globale di SAE
 - Con fast search rischio di finire in minimi locali

Three step search (TSS)

- È il più noto
- 1. Comincio a cercare da $(0, 0)$.
- 2. Pongo $S = 2^{N-1}$ (dimensione passo).
- 3. Cerco nelle 8 locazioni a $\pm S$ pixel di distanza attorno a $(0, 0)$.
- 4. Tra le 9 locazioni analizzate prendo quella con SAE minore e la faccio diventare il nuovo centro di ricerca.
- 5. Pongo $S = S/2$.
- 6. Ripeto i passi da 3 a 5 finché $S = 1$.

Three step search (TSS)



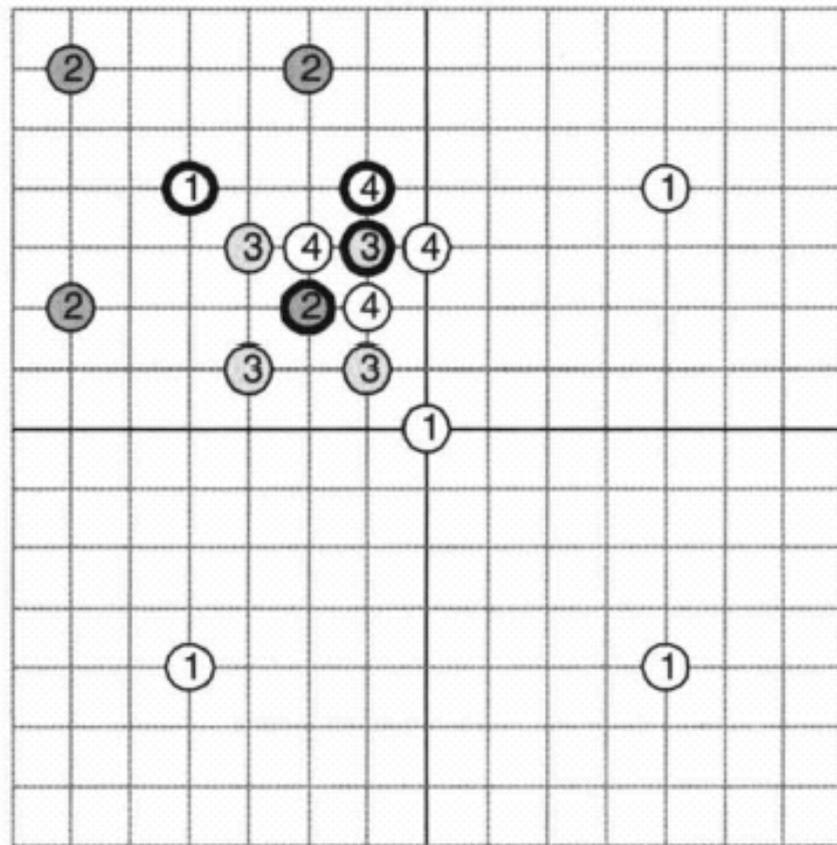
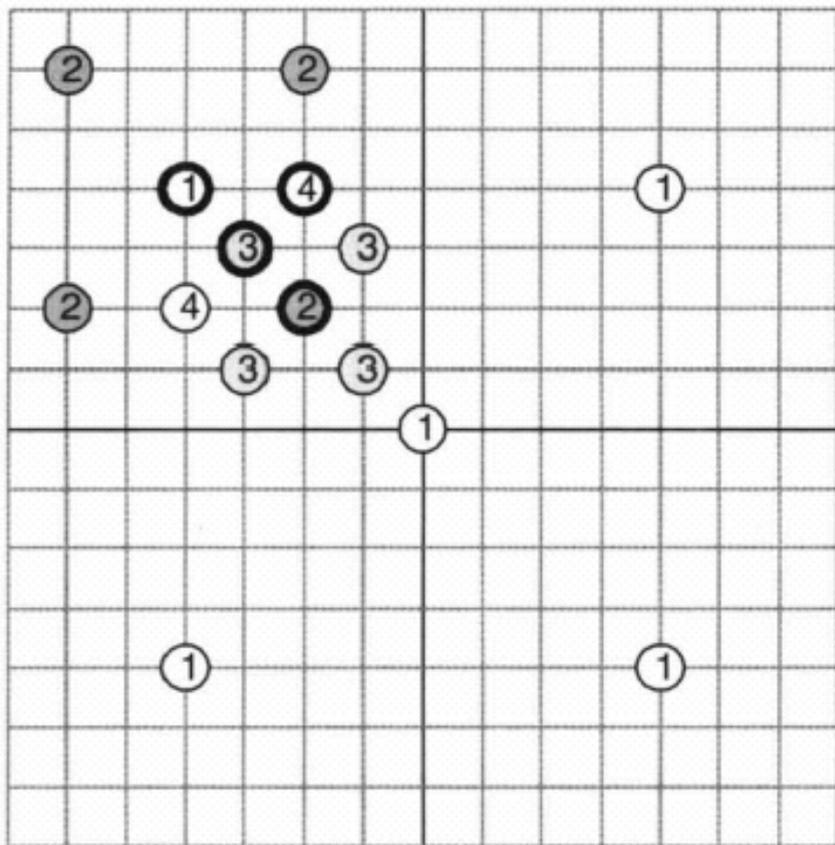
Logarithmic Search

1. Comincio a cercare da (0, 0).
2. Cerco nelle 4 posizioni adiacenti in orizzontale e in verticale, a S pixel di distanza dall'origine (con S passo di ricerca iniziale). Le 5 posizioni modellano un '+ '.
3. Impostare la nuova origine in corrispondenza del match migliore. Se il match migliore è la posizione centrale del '+' allora $S = S/2$, altrimenti S rimane uguale.
4. Se $S = 1$ vai al punto 5, altrimenti vai al punto 2.
5. Cerca le 8 posizioni attorno al best match. Il risultato finale è il best match tra queste 8 posizioni e la posizione centrale.

Cross-Search

- Simile al TSS tranne che ad ogni passo si comparano 5 punti (che formano una **X**) invece di 9 punti
 1. Comincio a cercare da (0, 0).
 2. Cerco nelle 4 posizioni a +/-S pixel di distanza, che formano una 'X' (con $S = 2^N - 1$ come in TSS).
 3. Imposto la nuova origine in corrispondenza al best match tra i 5 punti.
 4. Se $S > 1$ allora $S = S/2$ e vado al passo 2; altrimenti vado al passo 5.
 5. Se il best match è in alto a sx. o in basso a dx della 'X', valuto altri 4 punti che formano una 'X' ad una distanza di +/-1; altrimenti (best match in alto a dx o basso a sx) valuto altri 4 punti che formano un '+' ad una distanza di +/-1.

Cross-Search



One-at-a-Time Search

1. Comincio a cercare da $(0, 0)$.
2. Cerco nell'origine e nei due vicini orizzontali
3. Se l'origine ha il SAD più basso allora vado al passo, altrimenti. . . .
4. Imposto l'origine nel punto orizzontale con il SAD più piccolo e cerco nel vicino in cui ancora non avevo cercato. Vado al passo 3.
5. Ripeto i passi da 2 a 4 nella direzione verticale.

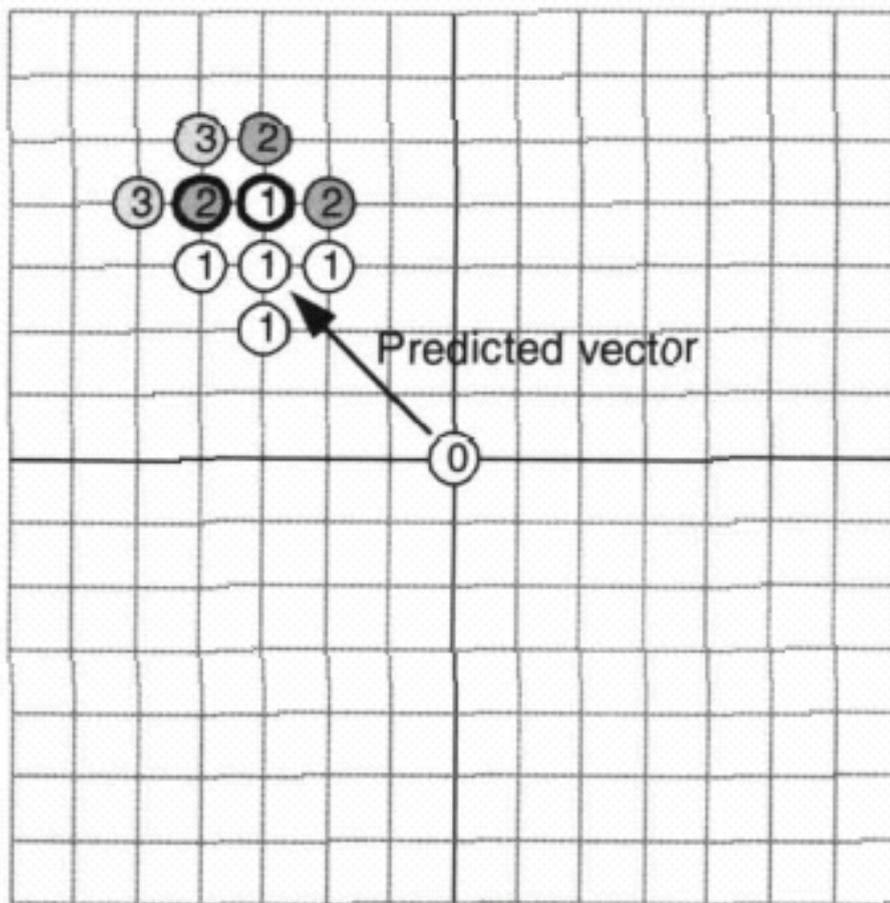
Nearest Neighbours Search

- Algoritmo proposto per H.263 e MPEG-4.
 - Ogni vettore di moto viene predetto dai vettori vicini (già codificati)
 - I macroblocchi vicini hanno spesso vettori di moto simili (un predittore basato su mediana è vicino al 'vero' best match)
 - Un vettore vicino alla mediana verrà codificato con un VLC piccolo.

Nearest Neighbours Search

1. Comincio a cercare da $(0, 0)$.
 2. Pongo l'origine della ricerca nella posizione indicata dal vettore predetto e cerco a partire da questa nuova posizione.
 3. Cerco nei 4 vicini in forma di '+ '.
 4. Se l'origine della ricerca (o la posizione $0, 0$ della prima iterazione) fornisce il risultato migliore prendo il risultato; altrimenti imposto la nuova origine nel best match e vado al passo 3.
- L'algoritmo si ferma quando il best match è al centro di un '+ ' (o quando si è raggiunto il bordo della finestra di ricerca).

Nearest Neighbours Search



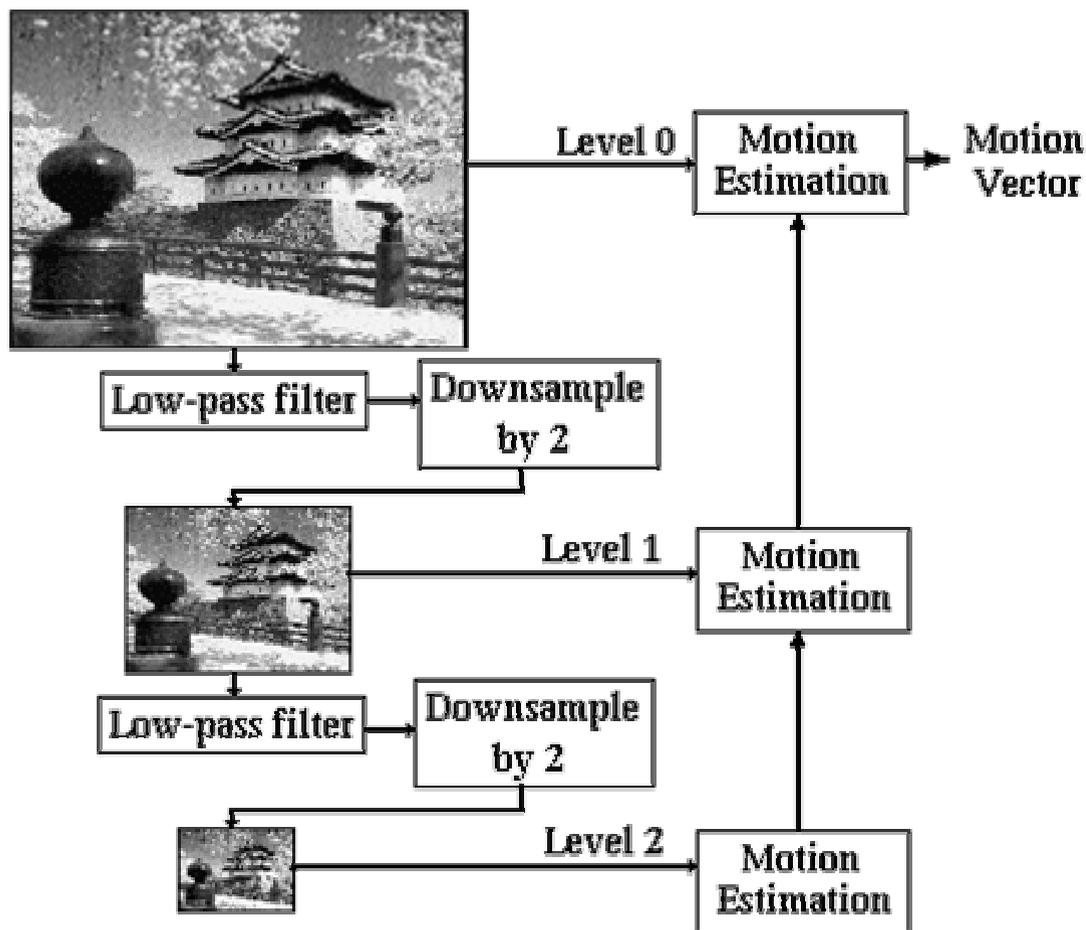
Nearest Neighbours Search

- L'algoritmo funziona bene quando i vettori di moto sono ragionevolmente omogenei
 - Non ci devono essere troppi cambiamenti grossi nel campo dei vettori di moto.
- Vengono comunque seguiti due accorgimenti per migliorare il risultato.
 - Se il predittore di mediana non può essere molto accurato (es. perché troppo macroblocchi vicini sono intra-coded e quindi senza MV), si usa un algoritmo alternativo come TSS.
 - Si usa una funzione di costo che stima se è ragionevole il costo computazionale di un altro set di ricerche.

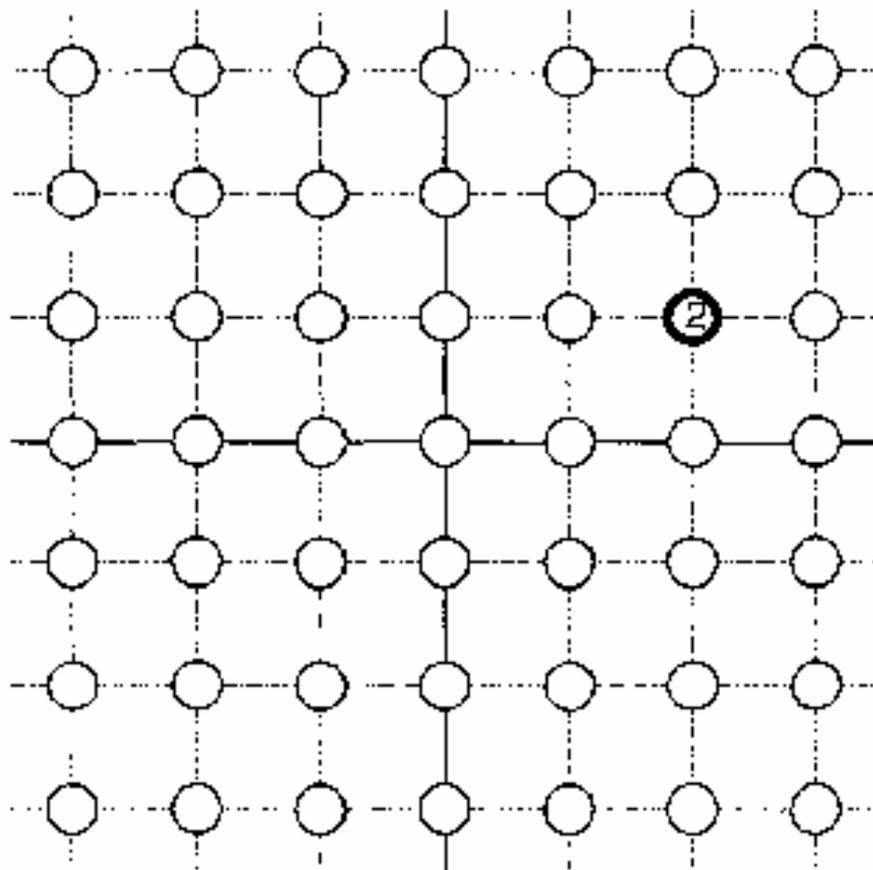
Hierarchical Search

- Effettua la ricerca in una versione sottocampionata dell'immagine, la ricerca viene quindi raffinata usando versioni dell'immagine a risoluzione maggiore, finché non si arriva alla risoluzione originale
 1. Il livello 0 consiste nell'immagine corrente ed in quella di riferimento a piena risoluzione. Si sottocampiona il livello 0 di un fattore 2 sia in orizzontale che verticale, producendo il livello 1.
 2. Si ripete il subsampling del livello 1 per produrre il livello 2, e così via finché si raggiunge il numero di livelli necessari (tipicamente 3 o 4 livelli bastano).
 3. Si inizia a cercare nel livello più alto (risoluzione più bassa) per cercare il best match: questo è il motion vector più 'grezzo'.
 4. Si cerca nel livello immediatamente più basso (maggiore risoluzione) intorno alla posizione del vettore di moto 'grezzo' e si cerca il best match.
 5. Si ripete il passo 4 finché non si trova il best match al livello 0.

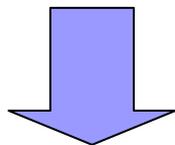
Hierarchical Search



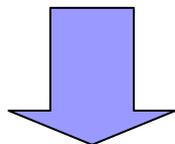
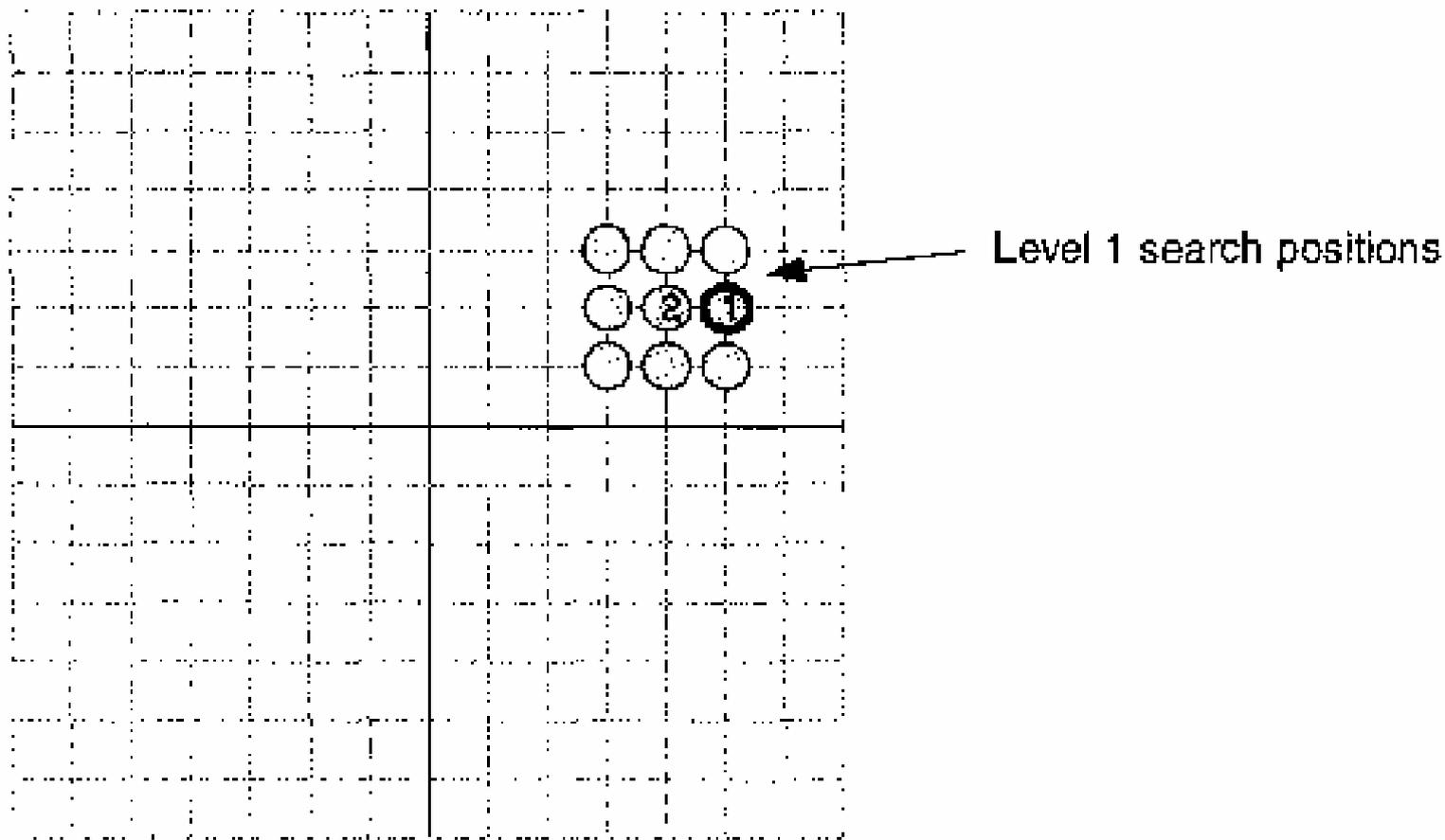
Hierarchical Search



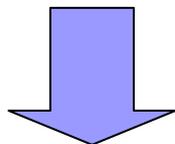
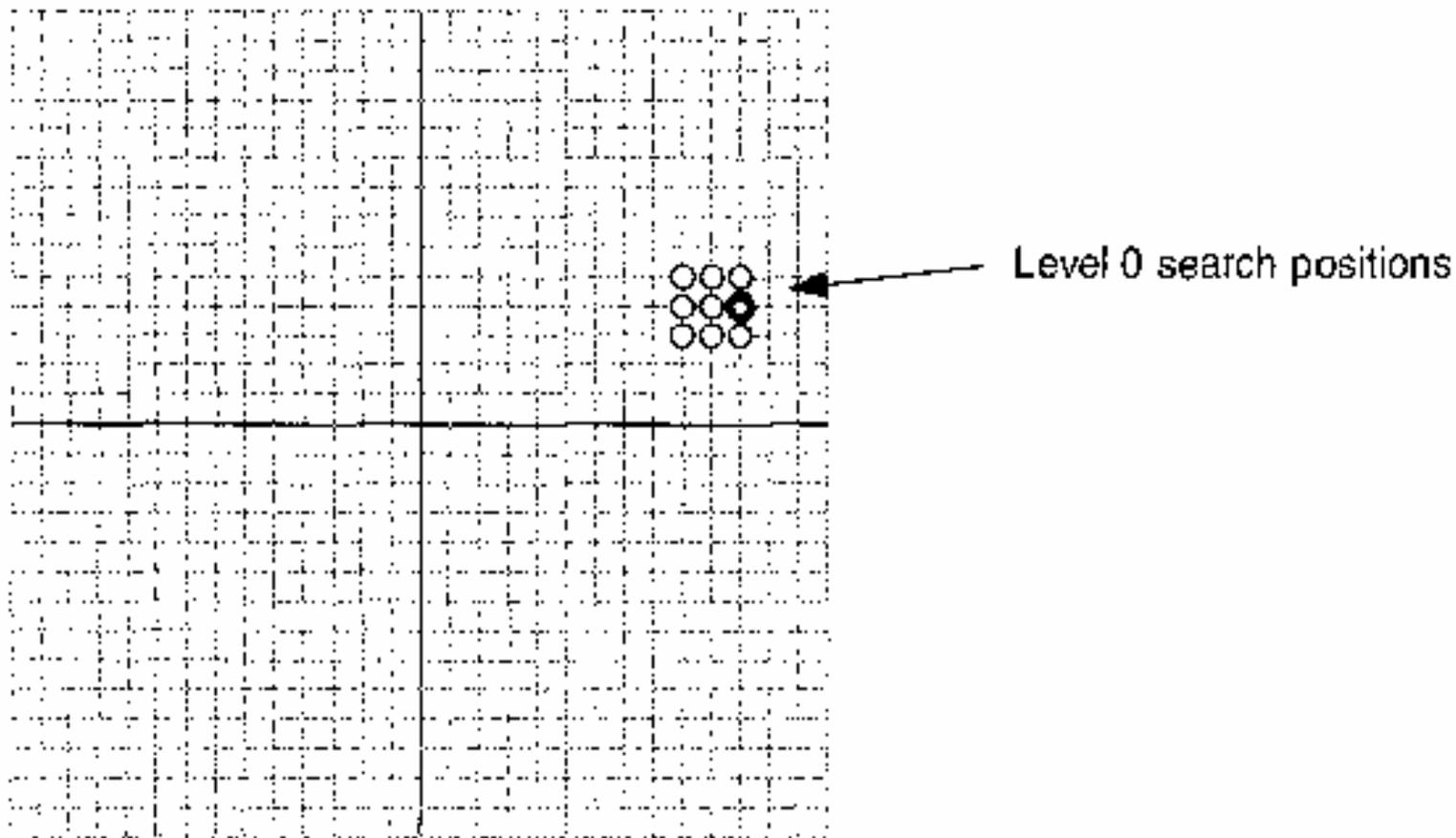
Level 2 search positions



Hierarchical Search



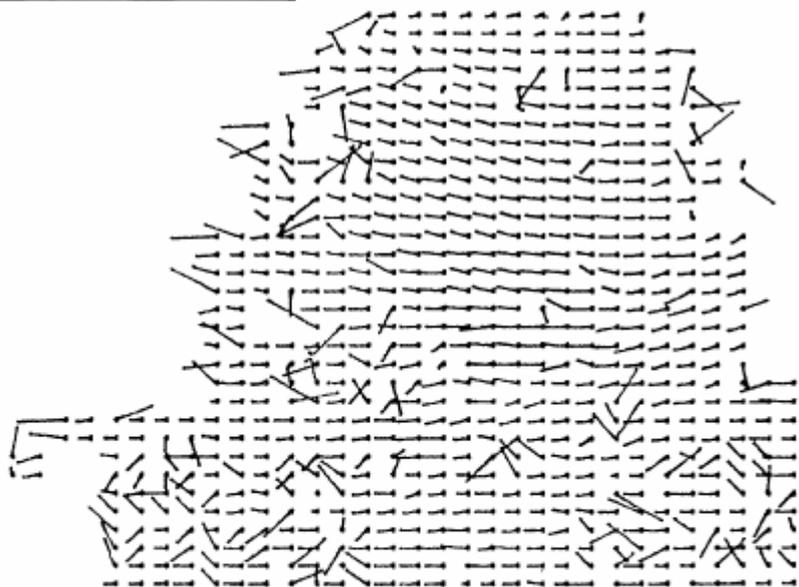
Hierarchical Search



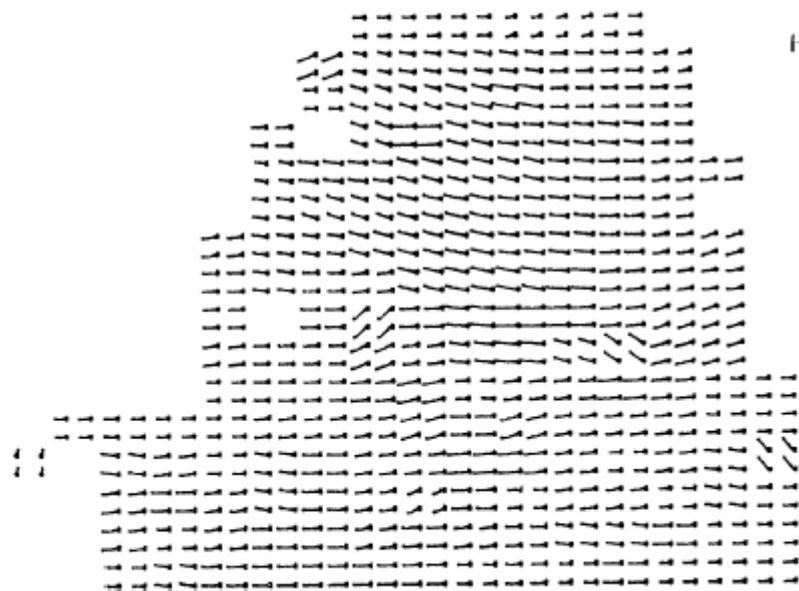
Hierarchical Search



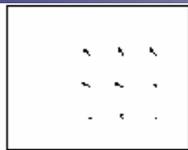
- Implicitamente ho uno smoothing dei vettori di moto



Full search block matcher



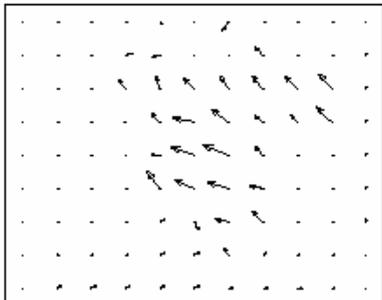
Hierarchical block matcher



(a)



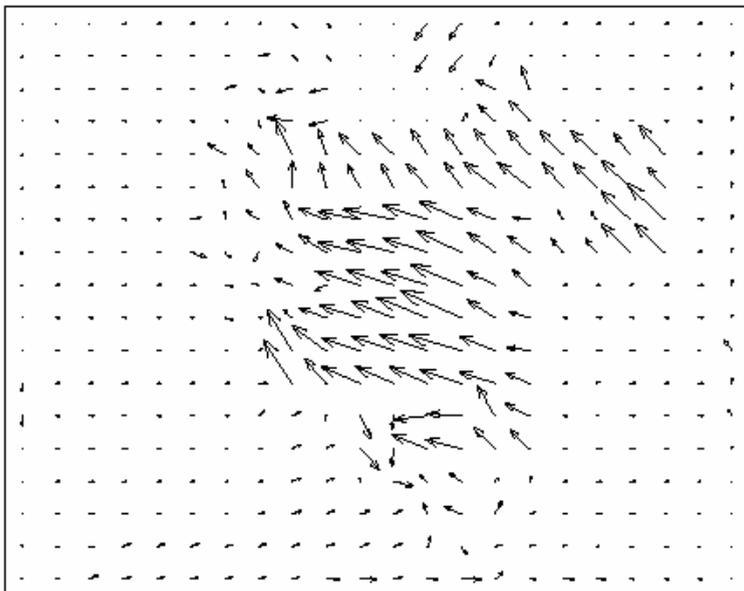
(b)



(c)



(d)



(e)



(f)

Comparazione algoritmi block matching

- Alcuni criteri da usare per la scelta dell'algoritmo di block matching:
 1. Matching performance: quanto è efficace l'algoritmo nel minimizzare il blocco residuo ?
 2. Rate-distortion performance: come si comporta complessivamente l'algoritmo a vari bitrate ?
 3. Complessità: quante operazioni sono necessarie per il block matching ?
 4. Scalabilità: l'algoritmo funziona bene sia con finestre di ricerca grandi che piccole ?
 5. Implementazione: l'algoritmo va bene sia che sia implementato in s/w che in h/w?

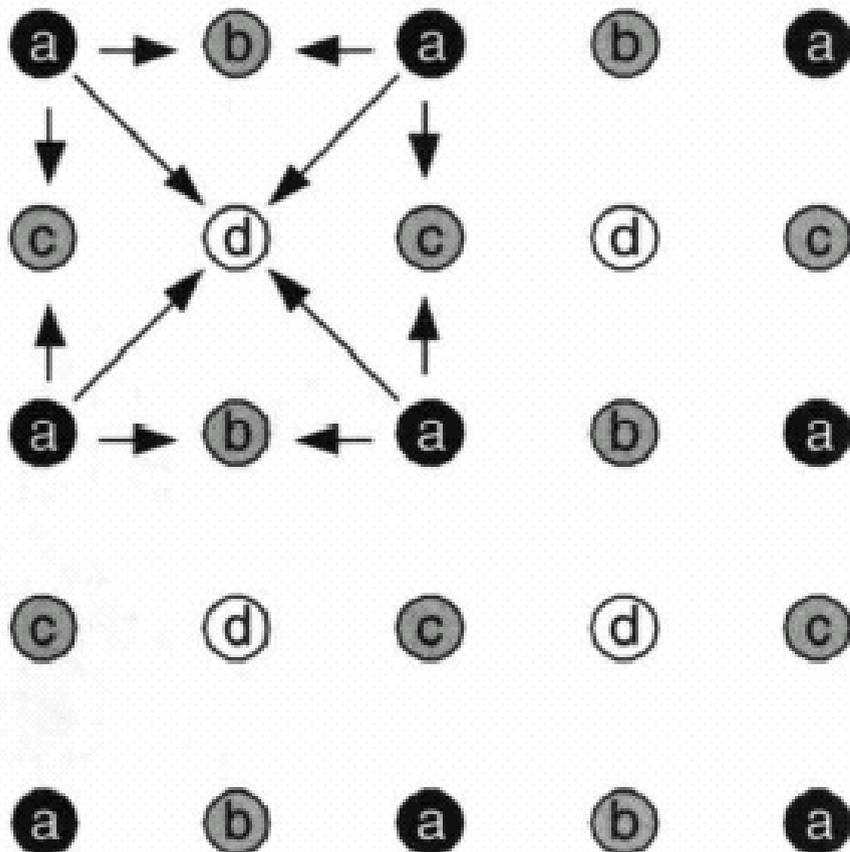
Comparazione algoritmi block matching

- Logarithmic search, cross-search e one-at-a-time hanno bassa complessità computazionale, al costo di spese una matching performance relativamente bassa.
- Hierarchical search è un buon compromesso tra performance e complessità ed è adatto per implementazioni hardware.
- Nearest-neighbours search, con la sua tendenza strutturale verso la predizione basata su mediana dei vettori di moto sembra funzionare bene quasi come una full search, ma con complessità molto ridotta.
 - La buona performance è dovuta al 'bias' della mediana, che tende a produrre piccole differenze nei vettori di moto, e quindi una loro codifica efficiente.

SUB-PIXEL MOTION ESTIMATION

- Per molti blocchi si ottiene un miglior match cercando in una regione interpolata, con accuratezza sub-pixel:
 - Il generico algoritmo di ricerca viene esteso come segue:
 1. Si interpolano i campioni nella search area dell'immagine di riferimento per creare una regione interpolata a più alta risoluzione.
 2. Si effettua la ricerca in locazioni full-pixel e sub-pixel nella regione interpolata e si cerca il best match.
 3. Si sottraggono i campioni della regione su cui si ha il best match (full- o sub-pixel) dai campioni del blocco corrente per formare il blocco differenza (error block).

SUB-PIXEL MOTION ESTIMATION



a: original integer samples

b,c,d: interpolated samples

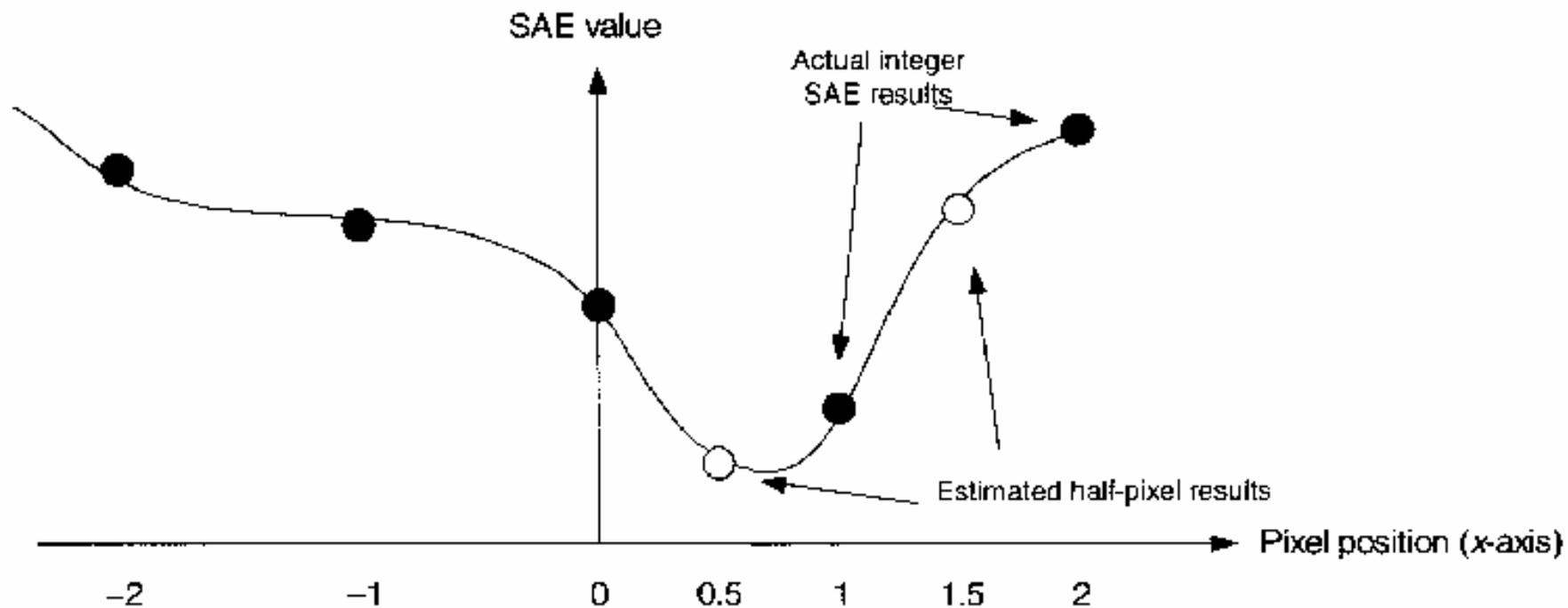
Arrows indicate direction of interpolation

Half pixel interpolation

SUB-PIXEL MOTION ESTIMATION

- Motion compensation con accuratezza half-pixel è supportata da **H.263**, **MPEG-1** e **MPEG-2** standard
- Livelli di interpolazione più elevati (1/4 pixel o più) sono proposti per gli standard emergenti H.26L/H.264. 1/4 pixel è usato in MPEG-4
- Aumentare la 'profondità' di interpolazione porta ad avere una migliore block matching performance al costo di un aumento di complessità computazionale.
 - Per limitare l'incremento di complessità computazionale di solito si cerca il best match su posizioni intere e poi si raffina con ricerca sub-pixel attorno alla posizione iniziale, oppure stimo a partire da valori basati su full-pixel

SUB-PIXEL MOTION ESTIMATION



Ottimizzazioni

■ Early termination

- Ogni volta che calcolo il best match, nel ciclo di calcolo della misura (es. SAE) controllo se ho passato il minimo finora ottenuto

- **Es.:**

```
if (SAE_attuale > SAE_minimo)
    break;
```

Ottimizzazioni

$$\sum_{\text{block}} |S_k - S_{k-1}| \geq \left| \sum_{\text{block}} S_k - S_{k-1} \right| = \left| \sum_{\text{block}} S_k - \sum_{\text{block}} S_{k-1} \right|$$

$$\sum_{\text{block}} |S_k - S_{k-1}|^2 \geq \frac{1}{N} \left| \sum_{\text{block}} S_k - S_{k-1} \right|^2 = \frac{1}{N} \left| \sum_{\text{block}} S_k - \sum_{\text{block}} S_{k-1} \right|^2$$

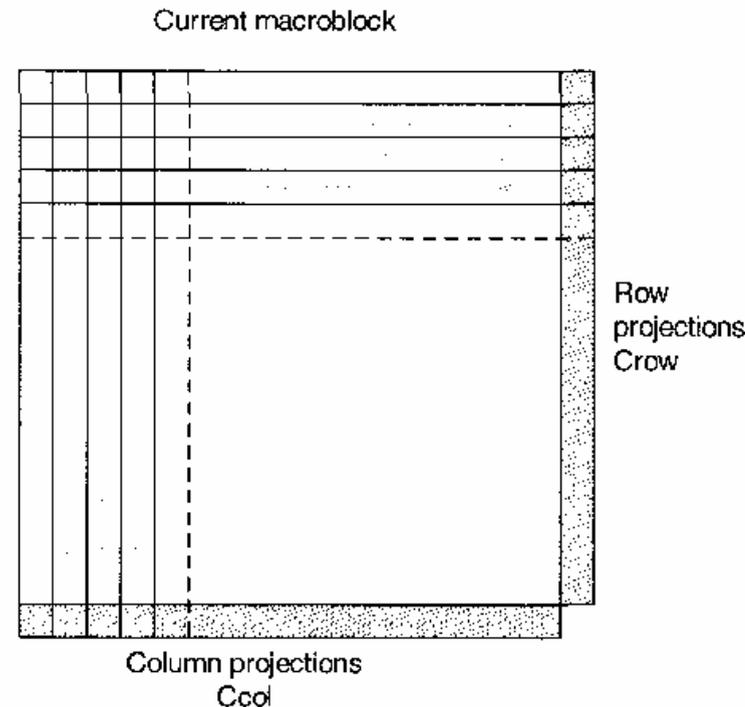
number of terms
in sum

- Per SAD ed SSE valgono le equazioni viste sopra; la strategia è:
 1. Calcolare le somme parziali per blocco corrente e riferimento
 2. Comparare i blocchi usando le somme parziali
 3. Non calcolare la comparazione standard se le somme parziali mostrano che l'errore è già maggiore del miglior risultato precedente

Ottimizzazioni

■ Row e column projections

- Una proiezione è calcolata sommando i valori di luminanza sulle colonne e le righe
- Si usano i valori delle proiezioni per approssimare il calcolo di SAE



$$SAE_{\text{approx}} = \sum_{i=0}^{N-1} Ccol_i - Rcol_i + \sum_{j=0}^{N-1} Crow_j - Rrow_j$$

VBR vs. CBR

- Ci sono due modi per gestire il bitrate:
 - Variable Bit Rate (VBR)
 - Il bitrate può variare
 - Constant Bit Rate (CBR)
 - Il bitrate è costante all'interno di una qualche finestra temporale.
- Nel sequence header è specificato se CBR o VBR.
 - Nel sequence header anche informazioni su come calcolare il buffer minimo per decomprimere i frame.

VBR Q-scale

- In generale: VBR usato per mantenere la qualità del video.
- Il Q scale è aggiornato per mantenere la massima compressione sulla base di una qualità minima richiesta.
- Dobbiamo specificare una metrica per definire la qualità.
- Soluzione comune: q scale impostato staticamente per I-, P-, e B-frame.
 - Si varia all'interno dei macroblocchi

CBR Q-scale

- Per mantenere il CBR si usa q-scale per controllare il bitrate.
 - Maggiore è il valore di q-scale maggiore è la compressione (qualità peggiore).
 - Con bassi q-scale si privilegia la qualità a spese della compressione.
- Soluzione comune: si imposta una dimensione obiettivo per I, P, e B frame; quindi si aggiusta il q-scale dei macroblocchi man mano che si codificano, per raggiungere il target.

DECOMPRESSIONE

- È veloce: niente ricerca o matching
- Se i Motion Vector sono presenti
 - usa le tavole standard per decodifica Huffman dei vettori di moto
 - Ricrea da codifica differenziale I vettori di moto
 - Leggi I blocchi di riferimento dal buffer
- Se il Quantizer è presente
 - scala la matrice di quantizzazione con q-scale

DECOMPRESSIONE (cont.)

- Se Block Code è presente
 - Usa la tavole dello standard per la decodifica Huffman dei coefficienti
 - Decomprimi RLE
 - Zigzag all'incontrario
 - Quantizzazione all'incontrario
 - DCT inversa
- Combina blocchi differenza e riferimento

DECOMPRESSIONE (cont.)

- Combina 6 blocchi in un MB (con unsubsampling)
- Combina macroblocchi in un'immagine
- Converte YCbCr in RGB per mostrare sullo schermo