Bit Plane

Un'immagine con una profondità colore di N bit può essere rappresentata da N piani di bit, ciascuno dei quali può essere vista come una singola immagine binaria

Bit-plane: Il bit plane di un'immagine digitale a N bit, è un'insieme di N immagini binarie (piani), in cui l'immagine i-esima contiene i valori dell'i-esimo bit della codifica scelta.

Bit-planes binario puro: Se si usa la codifica in binario puro i piani di bit più significativi contengono informazioni sulla struttura dell'immagine, quelli meno significativi forniscono dettagli meno significanti. Si noti che solo i piani dal 7 al 3 contengono dati significativi dal punto di vista visuale.

Il rumore delle immagini e gli errori di acquisizione sono più evidenti nei piani bassi.

Questo genere di scomposizione è molto utile per eliminare tutti i valori compresi in un certo range.

Codice Gray: Se la codifica usata è quella in binario puro, allora risulta evidente uno svantaggio: una piccola variazione può ripercuotersi su tutti i piani.

Il codice Gray a **m bit** ($gm-1 \dots g1, g0$) che corrisponde al numero in **binario puro** ($am-1 \dots a1, a0$) può essere calcolato con la formula:

$$g_i = a_i \oplus a_{i+1} \quad 0 \le i \le m-2$$
$$g_{m-1} = a_{m-1}$$

dove $\oplus$ denota l'operatore **XOR.**

Il codice Gray gode della proprietà per cui ogni **codeword** differisce dalla precedente per un solo bit.

I bit-plane delle immagini in codice Gray risultano tra loro più **''coerenti''** se confrontati con i rispettivi in binario puro. Se aumento l'intensità del pixel di 1 varierà infatti solo un bit (**ossia solo un piano**). Inoltre, il numero di transizioni bianco-nero nel singolo piano (**complessità descrittiva**) sono inferiori se si usa il codice Gray. Queste caratteristiche indicano una **minore entropia** (**maggiore ridondanza**) se si utilizza il codice Gray. Ciò significa che diventa più semplice comprimere a partire da immagini così codificate. Anche nel codice Gray, eliminare i piani più bassi può portare artefatti indesiderati.

Image Enhancement nel Dominio delle Frequenze

Una **funzione periodica** può essere espressa come somma di seni e/o coseni di differenti frequenze e ampiezze (**Serie di Fourier**).

Anche una **funzione non periodica**, (sotto certe condizioni) può essere espressa come integrale di seni e/o coseni, moltiplicati per opportune funzioni-peso (**Trasformata di Fourier**).

Sia la serie di Fourier che la Trasformata di Fourier condividono il fatto che una funzione possa essere "ricostruita" (**recovered**) con un semplice processo di inversione senza perdita di informazione. E' cioè possibile lavorare nel cosiddetto **dominio di Fourier** e tornare nel **dominio originale** della funzione in maniera del tutto naturale.

Trasformata e antitrasformata di Fourier

Nel caso 2-D la coppia trasformata antitrasformata della sequenza bidimensionale *f*(*x*,*y*) assume la seguente forma:

$$F(u,v)=\frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}f(x,y)e^{-i2\pi(\frac{ux}{M}+\frac{vy}{N})} \quad per\ u=0,1,...,M-1\ v=0,1,...,N-1$$

$$f(x,y)=\sum_{u=0}^{M-1}\sum_{v=0}^{N-1}F(u,v)e^{i2\pi(\frac{ux}{M}+\frac{vy}{N})} \quad per\ x=0,1,...,M-1\ \ y=0,1,...N-1$$

u e *v* sono gli indici relativi agli assi frequenze discretizzati, mentre *M* e *N* sono le dimensioni (in pixel) dell'immagine.

Dato che la trasformata F ha valori complessi, può essere espressa in termini della sua parte reale e della sua parte immaginaria.

Spettro della Trasformata $|F(u,v)|=\sqrt{R^2(u,v)+I^2(u,v)}$

Angolo di Fase $\phi(u,v)=\tan^{-1}\left[\frac{I(u,v)}{R(u,v)}\right]$

Potenza Spettrale $P(u,v)=|F(u,v)|^2=R^2(u,v)+I^2(u,v)$

Tutti i valori della **f(x)** contribuiscono alla costruzione di ciascuno dei **campioni** della **F(u)**. Analogamente, tutti i campioni della **trasformata** contribuiscono, durante la **antitrasformata**, a ciascuno dei valori della **f(x)**.

Range dinamico

Quando si visualizza lo **spettro di Fourier** come immagine di intensità, esso manifesta in genere una **dinamica molto più grande di quella riproducibile su un tipico display**, per cui solo le parti più **luminose** dello spettro risultano visibili.

A ciò si può ovviare, come è noto, mediante una **compressione di tipo logaritmico**, visualizzando, invece che lo spettro, una funzione del tipo:

D(u,v)=c log(1+ F(u,v)).

C: è una **costante di scala**, che va scelta opportunamente per far ricadere i valori trasformati nel range voluto, cioè in **[0, L-1].**

La visualizzazione dello spettro riguarda in realtà non **|F(u,v)|** ma una sua **versione compressa logaritmicamente**. Altrimenti si vedrebbe solo un puntino al centro. L'ampiezza contiene l'informazione relativa al fatto che una certa **struttura periodica** è presente nell'immagine. La fase contiene l'informazione relativa al dove le strutture periodiche evidenziate nella DFT sono collocate. Quindi è molto più significativa di quello che possa sembrare.

Vantaggi

Nello spazio delle frequenze è possibile:

-sopprimere frequenze indesiderate

- ridurre lo spazio occupato dai dati pur limitando la degenerazione del segnale (JPEG, MPEG, DivX, MP3)

- rigenerare segnali degradati**.**

La **trasformazione diretta** può essere vista come un **processo di analisi**: il segnale **f(x)** viene **scomposto** nelle sue **componenti elementari**, che sono nella forma dei **vettori di base**. I **coefficienti** della trasformata specificano quanto di ogni componente di base è presente nel segnale.

Nella **trasformazione inversa**, mediante un **processo di sintesi**, il segnale viene **ricostruito**, come **somma pesata** delle componenti di base: il **peso** di ogni vettore di base nella ricostruzione del segnale è rappresentato dal corrispondente coefficiente della trasformata. Il coefficiente della trasformata è una misura della correlazione tra il segnale ed il corrispondente vettore di base.

La trasformazione non comporta perdita di informazione: essa fornisce solo una rappresentazione alternativa del segnale originale.

proprietà della DFT 2-D

Separabilità:

La trasformata di Fourier discreta può essere espressa in forma separabile. In particolare vale la seguente espressione:

$$F(u,v) = \frac{1}{M}\sum_{x=0}^{M-1} g(x,v)e^{\frac{-i\pi ux}{M}}$$

Dove: $$g(x,v) = \left[\frac{1}{N}\sum_{y=0}^{N-1} f(x,y)e^{\frac{-i2\pi vy}{N}}\right]$$

Il principale vantaggio delle proprietà di separabilità è che la $F(u,v)$ può essere ottenuta applicando in due passi successivi la trasformata 1-D.

Traslazione:

Nel caso bidimensionale è utile prima di operare sulla trasformata applicare uno **shift** (**traslazione**) **dell'origine nel punto (M/2, N/2**) cioè nel **centro** della **matrice dei coefficienti delle frequenze**. In questo modo i dati vengono traslati in maniera tale che F(0,0) risulti il centro del rettangolo delle frequenze definito tra [0,M-1] e [0,N-1].

Si dimostra inoltre che uno shift nella f(x,y) non modifica la magnitudo della trasformata.

Valor Medio:

Il valore della trasformata nell'origine, cioè nel punto $(u,v)=(0,0)$ è dato da:

$$F(0,0)=\frac{1}{NxN}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}f(x,y) \quad \bar{f}(x,y)=\frac{1}{NxN}F(0,0)$$

Come si può vedere non è altro che la media di $f(x,y)$. Il valore della trasformata di Fourier di un'immagine $f(x)$ nell'origine è uguale alla media dei valori di grigio contenuti nell'immagine.
$F(0,0)$ prende anche il nome di *componente continua* o componente DC.

Fast Fourier Transform

$$F(u)=\frac{1}{N}\sum_{x=0}^{N-1}f(x)\exp[-i2\pi ux/N]$$

Questa forma abbassa la **complessità** da $O(N^2)$ a $O(N\log(N))$.

Frequenze: low & high

Esclusi i casi banali è normalmente impossibile fare **associazioni dirette fra specifiche parti dell'immagine** e la sua **trasformata** (**perdita di localizzazione spaziale**). Ricordando che la **frequenza** è legata alla **velocità di variazione** è però possibile associare le **basse frequenze** alle **zone uniformi** dell'immagine, quelle **alte** alle variazioni più o meno brusche e quindi ai **bordi o al rumore**.

La funzione **H(u,v**) prende il nome di **filtro** poiché agisce su alcune frequenze della trasformata lasciando le altre immutate. Molto spesso la funzione H è una **funzione reale** e ciascuna sua componente moltiplica sia la corrispondente componente reale che quella immaginaria della F. Questo tipo di filtri viene detto **zerophaseshift** perché non introduce sfasamento.

Teorema della Convoluzione

La convoluzione di due segnali nel dominio spaziale equivale all'antitrasformata del prodotto delle frequenze.

Questo teorema giustifica il perché si usa il dominio delle frequenze e non quello spaziale per usare gli operatori globali.

Il fondamento teorico delle tecniche di elaborazione nel dominio della frequenza, basate sulla manipolazione della DFT dell'immagine, è rappresentata dal **teorema della convoluzione** che fa corrispondere, alla operazione così definita nel dominio spaziale:

$$g(x,y) = f(x,y) * h(x,y) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n) h(x-m, y-n)$$

l'operazione, nel dominio delle frequenze:

$$G(u,v) = F(u, v)H(u,v)$$

La stessa operazione nel dominio delle frequenze diventa:

$$g(x,y)=F^{-1}\{F(u,v)H(u,v)\}$$

Complessità per un segnale 1D:

-Nel dominio delle frequenze O(n log(n));

- Nel dominio spaziale $O(n^2)$.

Effettivamente vale la pena passare al dominio delle frequenze.

Filtraggio nel Dominio della Frequenza

Se il filtro ha dimensioni confrontabili con quelle dell'immagine è più **efficiente computazionalmente** effettuare il filtraggio nel dominio delle frequenze. Con **maschere più piccole** diviene più efficiente il calcolo nel dominio spaziale. La definizione di un filtro nel dominio delle frequenze è più intuitiva.

Come ottenere un filtro a partire da una maschera spaziale:

1. Il filtro H ha la stessa dimensione dell'immagine I;

2. H deve avere in alto a sinistra i valori della maschera spaziale, nel resto sempre il valore 0;

3. Si fa lo shift di H

4. Si calcola da H la trasformata di fourier.

Filtri low pass, high pass e band reject nel dominio della frequenza

TABLE 4.4
Lowpass filters. D_0 is the cutoff frequency and n is the order of the Butterworth filter.

Ideal	Butterworth	Gaussian
$H(u,v) = \begin{cases} 1 & \text{if } D(u,v) \le D_0 \\ 0 & \text{if } D(u,v) > D_0 \end{cases}$	$H(u,v) = \dfrac{1}{1 + [D(u,v)/D_0]^{2n}}$	$H(u,v) = e^{-D^2(u,v)/2D_0^2}$

TABLE 4.5
Highpass filters. D_0 is the cutoff frequency and n is the order of the Butterworth filter.

Ideal	Butterworth	Gaussian
$H(u,v) = \begin{cases} 0 & \text{if } D(u,v) \le D_0 \\ 1 & \text{if } D(u,v) > D_0 \end{cases}$	$H(u,v) = \dfrac{1}{1 + [D_0/D(u,v)]^{2n}}$	$H(u,v) = 1 - e^{-D^2(u,v)/2D_0^2}$

TABLE 4.6
Bandreject filters. W is the width of the band, D is the distance $D(u,v)$ from the center of the filter, D_0 is the cutoff frequency, and n is the order of the Butterworth filter. We show D instead of $D(u,v)$ to simplify the notation in the table.

Ideal	Butterworth	Gaussian
$H(u,v) = \begin{cases} 0 & \text{if } D_0 - \dfrac{W}{2} \le D \le D_0 + \dfrac{W}{2} \\ 1 & \text{otherwise} \end{cases}$	$H(u,v) = \dfrac{1}{1 + \left[\dfrac{DW}{D^2 - D_0^2}\right]^{2n}}$	$H(u,v) = 1 - e^{-\left[\frac{D^2 - D_0^2}{DW}\right]^2}$

Compressione

Con il termine **compressione** dati si indica la tecnica di **elaborazione dati** che, attuata a mezzo di opportuni algoritmi, permette la riduzione della quantità di bit necessari alla rappresentazione in forma digitale di una informazione.

Un codice è un sistema di simboli utilizzati per rappresentare una certa quantità di informazioni. Ad ogni "pezzo" d'informazione e a ogni singolo evento è assegnata una sequenza di simboli codificati, chiamati **codeword.** Il numero di simboli che costituisce ciascun codice è la sua **lunghezza.**

Dal momento che differenti quantità di dati possono essere usate per rappresentare la stessa quantità di informazione le rappresentazioni che contengono informazioni irrilevanti o ripetute contengono i cosiddetti dati **ridondanti**.

Dal momento che la maggior parte degli array di intensità 2-D sono relazionati spazialmente (per es. ciascun pixel è simile ai pixel del suo intorno, o dipende da esso), l'informazione è replicata inutilmente nei pixel correlati. **Spesso i dati contengono informazioni ignorate dal sistema sensoriale umano**. E' ridondante nel senso che non viene utilizzata.

Compressione LOSSLESS

Si parla di compressione **LOSSLESS** quando i dati possono essere trasformati in modo da essere memorizzati con risparmio di memoria e successivamente ricostruiti perfettamente, senza errore e senza perdita di alcun bit di informazione.

Criterio: Cercare di raggiungere il limite teorico per la compressione senza perdita che viene fornito dal **primo teorema di Shannon.**

Entropia

Definiamo **entropia E** della **sequenza di dati S** la quantità media di informazione associata alla singola generazione di un simbolo nella sequenza S:

$$E = \Sigma\, f_i \log_2 (f_i), \quad i \in S$$

Più è grande l'incertezza della sequenza maggiore è l'entropia. Il massimo valore di entropia (e quindi di incertezza) lo si ha quando i simboli della sequenza sono equiprobabili.

Teorema di Shannon

«per una sorgente discreta e a memoria zero, il bitrate minimo è pari all'entropia della sorgente»

I dati possono essere rappresentati senza perdere informazione (lossless) usando almeno un numero di bit pari a:

$$N*E$$

Dove **N** è il **numero di caratteri** mentre **E** è l'**entropia**.

Il teorema di Shannon fissa il numero minimo di bit, ma non ci dice come trovarli.

Codifica di Huffman

E' un semplice **algoritmo greedy** che permette di ottenere un "dizionario" (cioè una tabella caratterecodifica_binaria) per una **compressione quasi ottimale** dei dati cioè **pari al limite di Shannon** con un **eccesso** di al più qualche bit.

Si tratta di codifica a **lunghezza variabile** che associa a simboli meno frequenti i codici più lunghi e a simboli più frequenti i codici più corti.

Si tratta di una codifica in cui **nessun codice è prefisso di altri codici**.

È una **codifica ottimale** perchè tende al limite imposto dal teorema di Shannon.

Costo aggiuntivo: si deve memorizzare la tabella caratteri-codici. Se i caratteri sono tanti questo può essere costoso.

Codifica Run-length

Le immagini che hanno delle ripetizioni di intensità lungo le righe (o colonne) possono spesso essere compresse rappresentando tali **sequenze** (**run**) sottoforma di coppie di **run-length**, in cui ciascuna coppia individua l'inizio di una nuova intensità e il numero di pixel consecutivi ne condividono il valore in questione.

Si voglia comprimere la sequenza:

00000111001011101110101111111

Si potrebbe ricordare in alternativa:

5 volte 0, 3 volte 1, 2 volte 0 etc.

O meglio basterebbe accordarsi sul fatto che si inizia con il simbolo 0 e ricordarsi solo la lunghezza dei segmenti (**run**) di simboli eguali che compongono la sequenza:

5,3,2,1,1,3,1,3,1,1,1,7

Tali valori vanno adesso scritti in binario.

Non sempre tale codifica porta un risparmio rispetto a quella di input. Ciò accade solo se la lunghezza della run è molto grande e prevede un numero di bit superiore a quelli necessari per scrivere il numero che rappresenta la run. Se ci sono molte "run" (sequenze di simboli eguali) piuttosto lunghe ricordare la sequenza delle loro lunghezze potrebbe portare un risparmio.

Codifica differenziale

Se la sequenza dei valori varia lentamente, invece di registrare i valori è sufficiente ricordarsi del valore iniziale e delle differenze successive.

Esempio:

134, 137, 135, 128, 130, 134, 112, …

ricorderò il valore iniziale 134 e poi la sequenza delle differenze successive**: -3, +2,7,-2,-4,22**

Compressione Lossy

Si parla di **compressione LOSSY** quando i dati possono essere trasformati in modo da essere memorizzati con risparmio di memoria ma con **perdita di informazione**. Tale tipo di compressione produce un maggiore risparmio di memoria.

Criterio: Fissata la **massima distorsione accettabile** l'algoritmo di compressione deve trovare la rappresentazione con il più basso numero di bit.

Viceversa, fissato il **massimo numero di bit accettabile** occorre trovare il miglior algoritmo di compressione che a parità di numero di bit mi dia la minima distorsione.

L'idea è quella di escludere tutte le informazioni non percettivamente importanti.

Requantization

Si tratta molto semplicemente di una riduzione del numero di livelli disponibili in modo da risparmiare bit per pixel. La si realizza **"dimenticando" n bit meno significativi per canale.**

Per esempio: **RED**: da 8 bit si conservano solo i 4 più significativi; **GREEN**: da 8 bit si conservano solo i 6 più significativi; **BLUE**: da 8 bit a si conservano solo i 2 più significativi.

Si risparmia così il 50% dei bit inizialmente necessario.

NB: se ci sono meno simboli la compressione Huffman è più efficiente.

Standard JPEG

Pre-processing:

i. Color Transform (RGB → YCbCr); (PRIMA PARTE STANCO)
ii. ii. Sottocampionamento della crominanza (PRIMA PARTE STANCO)
iii. iii. Suddivisione della immagine in sottoimmagini.

Trasformazione:

i. Discrete Cosine Transform;
ii. ii. Quantization;

Codifica:

i. DC Coefficient Encoding;
ii. ii. Zig-zag ordering of AC Coefficients;
iii. iii. Entropy Coding (Huffman).

Preprocessing (i): da RGB a Y C_b C_r

$$\begin{bmatrix} Y \\ C_b \\ C_r \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & 0.311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Preprocessing (ii): sottocampionamento della crominanza

L'occhio umano è più sensibile alla luminanza che alla crominanza.

- JPEG prende **TUTTE le informazioni sulla luminanza** ma sceglie **solo un campione delle informazioni degli altri canali.**

-**Si sceglie 1 valore ogni 4** per Cb e Cr (tralasciando metà dei valori originari da comprimere)

- Questo passo **è ovviamente con perdita di informazione ed è irreversibile**!

Preprocessing (iii): partizione della immagine

JPEG procede **dividendo l'immagine in "quadrotti"** 8 x 8 di 64 pixel non sovrapposti.

"Quadrotti" diversi subiranno una elaborazione differente: è qui l'origine del noto problema "quadrettatura"

Prima della applicazione della DCT **ai 64 pixel di ciascun blocco viene sottratta una quantità pari a 2 n-1 , dove 2n rappresenta il numero massimo di livelli di grigio** dell'immagine.

Con questo processo, noto come shift dei livelli di grigio, **il grigio medio (128) diventa 0.**

Trasformazione (i): la DCT

(**Discrete Cosine Tranform**, DCT) "**decorrela**" al massimo i dati permettendo maggiori rapporti di compressione nella fase successiva di codifica.

si tratta di una trasformazione del "vettore" di 64 pixel dalla base impulsiva (canonica) ad una più adatta alle immagini

Ogni immagine è quindi la somma pesata di 64 immagini impulsive e i "pesi" rappresentano l'effettivo livello di luminosità di ogni pixel.

Tali immagini impulsive costituiscono una base, detta "base impulsiva"

. La trasformata del coseno esprime l'immagine in un'altra base.

La formula della DCT per un blocco di dimensioni NxN (N=8 nel jpeg)

$$F(u,v) = \frac{2}{N}\left[\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} C(u)C(v)f(x,y)\cos\frac{(2x+1)u\pi}{2*N}\cos\frac{(2y+1)v\pi}{2*N}\right]$$

$$f(x,y) = \frac{2}{N}\left[\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} C(u)C(v)F(u,v)\cos\frac{(2x+1)u\pi}{2*N}\cos\frac{(2y+1)v\pi}{2*N}\right]$$

where :

$$C(u) = \frac{1}{\sqrt{2}} \text{ for } u = 0; C(u) = 1 \quad \text{otherwise}$$

$$C(v) = \frac{1}{\sqrt{2}} \text{ for } v = 0; C(v) = 1 \quad \text{otherwise}$$

Una implementazione diretta delle formule sopra richiede $O(N^2)$ Esistono algoritmi "fast" per calcolare i coefficienti in $O(N \log(N))$ derivati dalla Fast Fourier Transform.

Risultato DCT

Il coefficiente DC è solo il primo in alto a sinistra. Tutti gli altri 63 coefficienti sono detti coefficienti AC

Ogni immagine 8 x 8 si ottiene moltiplicando ciascuna delle immagini a sinistra per un coefficiente e sommando tutte le immagini.

I coefficienti di tale somma sono i coefficienti della DCT.

Il coefficiente in alto a sinistra è un valore proporzionale al valor medio della luminanza dell'immagine. E' detto anche coefficiente DC

Trasformazione (ii): Quantizzazione

Un vantaggio si ottiene se si riduce il numero di "livelli" su cui i coefficienti della DCT possono variare.

Tale operazione permette di rappresentare i diversi coefficienti incrementando il fattore di compressione, più precisamente avviene un processo di riduzione del numero di bit necessari per memorizzare un valore intero riducendone la precisione.

- Si usa il seguente "formalismo" per la quantizzazione. Dato un fattore di quantizzazione Q e un numero F il valore $F_{quantizzato}$ si ottiene come:

$$F_{quantizzato} = round(F/Q)$$

- Il valore ricostruito si ottiene moltiplicando $F_{quantizzato}$ per Q.
- Ovviamente la quantizzazione è un processo irreversibile (perdita di informazione)

non è conveniente usare un unico fattore di quantizzazione per tutti i 64 coefficienti della DCT della luminanza, o per quantizzare i valori provenienti dalla DCT delle crominanze. Si preferisce adottare per il coefficiente **F(i,j)** un **fattore di quantizzazione Q(i,j),** la tabella di quantizzazione deve essere trasmessa assieme ai dati compressi per consentire una corretta ricostruzione. I fattori Q(i,j) costituiscono la cosiddetta "**tabella di quantizzazione**".

Si osservi che un fattore di compressione maggiore comporta una maggiore perdita di informazione.

L'utente del JPEG può scegliere il "**grado**" di quantizzazione da adottare fornendo un "**quality factor**" **QF** che va da 1 a 100.

Maggiore il QF, minore i fattori di quantizzazione e minore la perdita di informazioni.

Tutti i coefficienti vengono riordinati in un vettore 64 x 1 seguendo l'ordinamento "a **serpentina**" (per creare lunghe run di zeri) e codificati in un altro stream.

Due differenti codifiche:

I coefficienti DC, cioè quelli che stanno nella posizione (1,1) del blocco 8x8, sono codificati usando una codifica differenziale;

I coefficienti AC, cioè tutti gli altri del blocco, sono codificati usando una codifica run-length.

Coefficienti DC

Si calcola Delta che è uguale a Dc del blocco 1 – Dc del blocco successivo

Viene inserito il valore nelle tabelle delle categorie

A questo punto occorre completare il codice. Per fare ciò si usa la seguente regola:

■ Se $\Delta > 0$ allora i bit da aggiungere sono gli n bit meno significativi del valore Δ in binario.

■ Se $\Delta < 0$ allora i bit da aggiungere sono gli n bit meno significativi del valore in binario di Δ (con complemento a due) ai quali occorre sottrarre il valore 1.

■ Δ = 0 allora anche SSSS è uguale a zero, pertanto non viene aggiunto alcun bit

Codifica a coefficienti AC

Dalla sequenza si è eliminato il primo coefficiente e si passa alla codifica di tutti gli altri.

Poiché i coefficienti quantizzati AC sono spessissimo nulli, si usa una trasformazione in **skip-value**. Cioè, data una sequenza di valori, si memorizza il numero degli zeri seguito dal primo valore non zero che si incontra.

Per codificare il coefficiente AC, si utilizza una tabella generata tramite la stima di un insieme di immagini.

Ricostruzione

Si deve tornare indietro "**ricostruendo**" i dati originali (o le loro approssimazioni per i passi irreversibili).

■ Esistono diverse "strategie" per la ricostruzione in modo da abilitare la ricostruzione progressiva, gerarchica o lossless.

Attenzione, il Jpeg è lossy! Quindi il blocco ricostruito è diverso da quello in input.

Immagini "grafiche" con pochi colori e con testi non sono compresse con buona qualità dal JPEG! Inoltre, per il WEB, JPEG non gestisce la trasparenza

JPEG2000: sostituisce la DCT con le **wavelets**. Alloca più bit nelle zone con più informazione e permette il controllo esplicito di tale allocazione. Raggiunge rapporti di compressione più elevati.

SSSS	Coefficiente AC
1	-1, 1
2	-3 -2, 2 3
3	-7 … -4, 4 … 7
4	-15 … -8, 8 … 15
5	-31 … -16, 16 … 31
6	-63 … -32, 32 … 63
7	-127 … -64, 64 … 127
8	-255 … -128, 128 … 255
9	-511 … -256, 256 … 511
A	-1023 … -512, 512 … 1023

- La classe dell'evento è espressa mediante una coppia del tipo (run, categoria).
- Nel nostro caso abbiamo (0,2)

(run, category)	Codice base	Lunghezza codice completo
(0, 0)	1010 (= EOB)	4
(0, 1)	00	3
(0, 2)	01	4
(0, 3)	100	6
….	….	….
(F, 0)	111111110111	12
….	….	….
(F, A)	1111111111111110	26

n.b.: le tabelle complete sono disponibili su studium

Tale codice sarà completato dall'aggiunta di un numero di bit fino a raggiungere il numero totale di bit che è riportato nell'ultima colonna della tabella.

■ I bit che completano il codice base sono scelti con lo stesso criterio enunciato per la codifica dei coefficienti DC, in base al valore v del coefficiente AC ($v > 0$, $v < 0$, $v = 0$).

■ Se $v > 0$ allora i bit da aggiungere sono i bit meno significativi del valore v in binario.

■ Se $v < 0$ allora i bit da aggiungere sono i bit meno significativi del valore in binario di v (con complemento a due) ai quali occorre sottrarre il valore 1.

■ $v = 0$ allora anche SSSS è uguale a zero, pertanto non viene aggiunto alcun bit.