**Esercizio sulla compressione JPEG**

**Quesito**: Spiegare brevemente in quali punti del processo di compressione JPEG avviene la perdita di dati.

**Soluzione**: al fine di rendere più completo l’esercizio verranno descritti brevemente i vari passi della codifica JPEG e infine verrà specificato in quali di essi avviene la perdita dei dati e per quale motivo.

L’obiettivo di JPEG è permettere il trasferimento di immagini ad una velocità ragionevole e la loro visualizzazione ad una qualità ragionevole. JPEG è una compressione con perdita che cerca di eliminare i dettagli dell’immagine che non possono essere percepiti dall’occhio umano.

Le fasi della codifica sono 6:

1. **Preparazione dell’immagine**: viene cambiato lo spazio dei colori dell’immagine e viene divisa in blocchi 8x8;
2. **Applicazione DCT**: si passa da uno spazio di luminosità dei colori ad uno spazio di frequenze. La DCT si applica ad ogni blocco 8x8 dell’immagine e trasforma i pixel di quest’ultimo in informazione. Al termine dell’applicazione di DCT, in ogni blocco, il punto (0,0) rappresenta il colore dominante all’interno del blocco, mentre tutti gli altri rappresentano di quanto il colore si discosta da quello dominante;
3. **Quantizzazione**: avendo a disposizione informazioni sulle frequenze, e quindi sul dettaglio, in questo passo è possibile eliminare i dettagli ritenuti irrilevanti, ovvero quelli non percepiti dall’occhio umano;
4. **Organizzazione dati**: i dati vengono organizzati in modo da facilitarne la memorizzazione;
5. **Linearizzazione matrice**: poiché dopo l’applicazione della quantizzazione si ottiene una matrice prevalentemente composta da zeri, occorre applicare la codifica zero-encoding in modo da compattarli;
6. **Codifica di Huffman**: l’ultimo passo della catena consiste nell’applicazione di una codifica senza perdita di dati, al fine di comprimere il più possibile.

Le fasi di codifica dove avviene la perdita dei dati sono la **preparazione dell’immagine**, **l’applicazione della DCT** e la **quantizzazione**.

1. **Preparazione dell’immagine**: in questa fase l’immagine RGB viene trasformata in un’immagine YUV (o YCbCr). Prima di tutto vengono eliminate le informazioni sul colore, ottenendo un’immagine in bianco e nero con lo stesso livello di dettaglio. In secondo luogo si effettua il sotto-campionamento della crominanza, che permette la riduzione delle dimensioni del file senza che la perdita dei dati venga percepita dall’occhio umano. Questo perché l’occhio è più sensibile alla luminanza rispetto che alla crominanza. Quindi, in questa fase si diminuiscono drasticamente le dimensioni dell’immagine (metà dei dati vengono persi) senza che l’occhio umano possa percepire cambiamenti (i dati persi sono quindi irrilevanti). L’immagine di partenza consiste in tre tabelle 640x480, mentre l’immagine in output consiste in:
   1. Una tabella 640x480 contenente le informazioni relative alla luminanza (Y), sulla quale non è possibile applicare sotto-campionamento, pena drastica perdita di qualità;
   2. Una tabella 240x320 contenente le informazioni relative alla crominanza (U), codificata come differenze rispetto ai colori di riferimento (U=B-Y) e sotto-campionata;
   3. Una tabella 240x320 contenente le informazioni relative alla crominanza (V), codificata come differenze rispetto ai colori di riferimento (V=R-Y) e sotto-campionata.

Quindi, al termine della preparazione dell’immagine si perde il 50% dei dati, questo grazie al sotto-campionamento della crominanza.

1. **Applicazione DCT**: prima della quantizzazione viene applicata la trasformata coseno discreta a ciascuno dei blocchi 8x8 in cui l’immagine è stata suddivisa. Questa trasformata consente di scomporre il segnale in entrata in tutte le sue componenti. Ad ognuna delle viene associato un coefficiente e tutti i coefficienti vengono memorizzati nel blocco 8x8. Nel punto (0,0) viene rappresentato il colore dominante del blocco, mentre negli altri blocchi vengono rappresentate le differenze rispetto al colore dominante in orizzontale (prima riga), in verticale (prima colonna) e in diagonale (variazioni combinate). In questa fase vengono persi dati a causa dell’aritmetica finita del calcolatore, infatti nell’applicare la DCT vengono calcolati dei coseni. Nonostante la DCT sia in forma teorica completamente reversibile, ovvero dopo aver applicato la DCT e la sua inversa ottengo il valore di partenza, ciò non accade sul calcolatore, infatti dopo aver applicato la DCT e la sua inversa non ottengo più lo stesso valore. Questo fenomeno si può osservare applicando il seguente processo più volte:
   1. apertura immagine;
   2. salvataggio immagine.

Infatti, ogni volta che riaprirò un’immagine precedentemente salvata noterò un degradamento della qualità, e questo è dovuto al fatto che ad ogni salvataggio la DCT viene applicata ai valori precedentemente calcolati dal calcolatore durante l’applicazione della DCT nel salvataggio precedente, che non sono mai teoricamente quelli ottimali a causa dell’aritmetica finita del calcolatore.

1. **Quantizzazione**: poiché in seguito all’applicazione della trasformata è possibile distinguere molto bene solamente le informazioni sopra la diagonale di ognuno dei blocchi 8x8, la fase di quantizzazione cerca di eliminare i dati sotto la diagonale (rendere i coefficienti uguali a zero). La fase di quantizzazione utilizza una tabella di quantizzazione (fortemente sbilanciata), che permette di rendere zero tutti i coefficienti sotto la diagonale e di interferire il meno possibile con i coefficienti sopra la diagonale. In particolare i coefficienti vicino a (0,0) non verranno minimamente toccati dalla quantizzazione. In questa fase vi è la perdita di informazione più evidente, infatti in fase di decodifica alcuni coefficienti che erano diversi da zero prima della codifica e che sono diventati zero in seguito alla quantizzazione, resteranno a zero in seguito alla decodifica. Per esempio supponiamo che nella DCT ci sia un coefficiente pari a 2 e che nella tabella di quantizzazione il suo divisore sia 4. In seguito alla quantizzazione, nella tabella verrà memorizzato 0 (nonostante sarebbe 0.5 il valore corretto). In fase di decodifica sarà quindi impossibile ottenere nuovamente il 2, perché si otterrà 0 = 0\*4.