Comunicazioni Multimediali

Compressore di Immagini:
Il formato Gif

Il progetto

Lo scopo di questo progetto è quello di simulare la compressione di immagini multimediali, tentando di eguagliare le prestazioni del formato gif.

Nel formato PPM, un'immagine ha dimensione X*Y*3; con il formato gif, invece, le dimensioni dell'immagine si riducono a circa X*Y , più circa 1Kb di informazioni della paletta. Si riducono così di circa 3 volte le dimensioni dell'immagine.

L'Idea

- Cos'è un immagine? Un'immagine non è altro che una matrice in cui ad ogni elemento della matrice, chiamato picture element (pixel), corrisponde informazioni inerenti al colore di tale punto;
- L'idea: se creiamo una lista di colori poi possiamo indicare nella matrice la posizione del corrispondente colore nella paletta;
- Come ci guadagno? Utilizzando una paletta con al massimo 256 colori posso ridurre le informazioni superflue.

Il ciclo di vita

La codifica " paj " segue i seguenti passi:

- Lettura file formato " ppm ";
- Creazione paletta colori;
- Scrittura nel file finformazioni formato " paj ";

Il programma, chiamato ppm2paj, necessita di 2 parametri in ingresso: un'immagine di tipo ppm e un file in uscita di tipo paj.

La genesi

Il primo step della codifica è dunque la lettura della sorgente nel formato PPM. Viene creata una struttura dati ' ad hoc' formata da: dimensione immagine (X, Y) e un vettore di dimensione X * Y contente i colori, ovvero una tripla di char (red , green, blue).

Ora che ho una struttura dati che rappresenta la sorgente, la riempio con le informazioni necessarie; questa operazione viente fatta nella funzione PPMRead() che, attraverso diversi fread() leggo le informazioni contenute nel file passato come primo argomento all'avvio del programma.

```
typedef struct {
    unsigned char red,green,blue;
} PPMPixel;

typedef struct {
    int x, y;
    PPMPixel *data;
} PPMImage;
```

Scrittura

Una volta costruita, " in qualche modo " , la paletta, nel metodo writePaj() salvo nel file .paj le informazioni necessarie tramite fprintf(); tra queste troviamo le dimensioni dell'immagine e della paletta, i colori della paletta. Poi costruisco una matrice in cui seleziono il colore " più simile " confrontando ogni colore dell'immagine originale con quelli della paletta e inserendo, in tale matrice, la posizione del colore selezionato della paletta. Terminato questa operazione, salvo nel file anche queste informazioni appena trovate.

```
typedef struct {
    int x, y , z;//x e y
    PPMPixel *paletta;
    unsigned char *data;
} PAJImage;
```

L'algoritmo

La parte interessante del progetto è dunque il metodo " ppm2paj() ", in cui avviene la realizzazione della paletta contenente 256 colori. In essa vi sono varie fasi di elaborazione:

- Creazione della lista colori: qui, si sfogliano i vari pixel dell'immagine originale e viene stilata una lista di colori che essa contiene; viene inoltre creata una lista contenente le frequenze di tali colori;
- Realizzazione della paletta: se la lista dei colori contiene meno di 256 elementi, inserisco tali elementi nella paletta e termino questa operazione di creazione; viceversa, se il numero di colori risulta essere maggiore, avviene un'ottimizzazione dei colori. Per far questa operazione dobbiamo calcolare la distanza tra i vari colori. Per far ciò, ho creato una funzione che accetta in input due colori e restituisce un valore reale:

distanza = sqrt((c1.red – c2.red)^2 + (c1.green – c2.green)^2 (c1.blue – c2.blue)^2)

viene quindi calcolata la distanza dei 2 colori, posizionati in un piando cartesiano, dove all'asse x corrisponde red, ad y green, e all'asse z blue.

L'algoritmo (2)

Dobbiamo dunque creare una paletta dei colori in grado di rappresentare l'immagine.
L'idea è quella di selezionare i 256 colori più distanti tra di loro e dunque, maggiormente diversi. Per realizzare questa operazione dovremo verificare la distanza tra tutti i colori della lista. Avremo bisogno di una matrice triangolare per ospitare questi dati e selezionare quelli con la distanza maggiore tra loro. Questa operazione risulta essere molto onerosa, sia da un punto di vista temporale che da un punto di vista spaziale.

Dunque, calcoliamo la distanza tra i colori della paletta e vengono selezionati i colori con il valore minore (vicini spazialmente).
Poi, per ogni colore della lista, eseguiamo le seguenti operazioni:

1) Prelevo il colore e calcolo la distanza del colore con quelli della paletta e inserisco i valori all'interno di un vettore di dimensione 256;
2) Seleziono i colori più vicini della matrice e il colore più vicino della paletta rispetto al nuovo;
3) Se la distanza tra i colori della paletta è minore a quello della paletta-nuovo colore allora compatto i 2 colori della paletta e inserisco il nuovo colore nella paletta; viceversa compattiamo il nuovo colore con quello più vicino alla vettore;

L'algoritmo (3)

Cosa significa compatto 2 colori? Dati due colori c1 e c2 con frequenza rispettivamente f1 e f2 calcolo:

-Max{f1, f2} + = min {f1, f2}

Successivamente elimino il colore con frequenza minore.

Creata la paletta dei colori non mi resta che creare la matrice di riferimento; per far questo seleziono il colore la cui distanza è più vicina tra ogni pixel dell'immagine iniziale e la paletta. Inserisco nella nuova matrice la posizione del colore nella paletta.

Qualche risultato

Fiore.ppm (206.9Kb)

Fiore.paj (69.8Kb)

Fiore.gif (38.9Kb)

Dimensione: 350x197
Colori: 37306
Tempo compressione: 22s
Mos: 4.5
PSNR: 35.53

Qualche risultato (1)

Oasis.ppm(270.1kb)

Oasis.paj (90.0kb)

Oasis.gif (61.4kb)

Dimensione: 300x300
Colori: 31130
Tempo compressione: 20s
Mos: 5
PSNR: 37.70

Qualche risultato (2)

Sunset.ppm (900.1Kb)

Sunset.paj (300.8Kb)

Sunset.gif (129.5Kb)

Dimensione: 600x500
Colori: 224498
Tempo compressione: 6m 8s
Mos: 3.9
PSNR: 29.00

Qualche risultato (3)

NotteStellata.ppm (1200,0Kb)

NotteStellata.paj (400,8Kb)

NotteStellata.gif (410,2Kb)

Dimensione: 800x500
Colori: 329180
Tempo compressione: 12m 3s
Mos: 5
PSNR: 30,43

Conclusioni

Per quanto riguardano i risultati si può notare come le immagini compresse risultano molto simili a immagini converite in formato gif da programmi di terzi (Gimp).
Sotto l'aspetto della dimensione di esse sono stati raggiunti i risultati attesi, ovvero di circa 1/3 dell'immagine originale. Tuttavia risulta evidente che le dimensioni del formato gif sono inferiori a quelle del formato paj in quanto in quest'ultimo formato, non sono stati applicati compressori di tipo testo.

Concludendo, ritengo soddisfacenti i riusltati ottenuti.