Fondamenti di Informatica - Compito A

Prof. Marco Gavanelli

22 gennaio 2014

Le immagini con sfumature di grigio possono essere salvate in formato PGM. Nel formato PGM, le immagini sono rappresentate come file di testo. Il file è organizzato per righe. La prima riga contiene, separati da spazi:

- la stringa `"P2"`
- la larghezza W dell'immagine (intero, massimo 100)
- l'altezza H dell'immagine (intero, massimo 100)
- il numero S di sfumature (al massimo 255)

Le successive H righe rappresentano le linee dell'immagine.

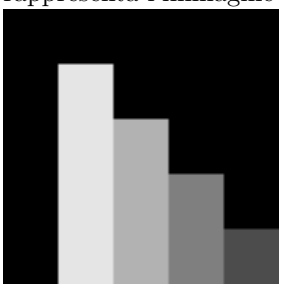
Ogni riga è costituita da W numeri che vanno da 0 a S; i numeri rappresentano l'intensità dei punti (o pixel) della riga:

- 0 rappresenta il colore nero,
- S rappresenta il bianco
- e i numeri intermedi rappresentano le varie tonalità di grigio.

Ad esempio, il seguente file:

```
P2  5  5  9
 0  0  0  0  0
 0  9  0  0  0
 0  9  7  0  0
 0  9  7  5  0
 0  9  7  5  3
```

rappresenta l'immagine

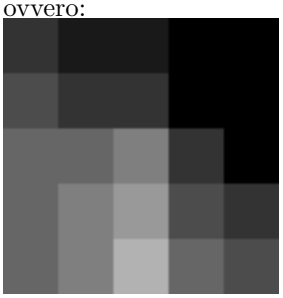


Dato un file `immagine.pgm` in formato PGM, si desidera creare un nuovo file `sfuocata.pgm` che rappresenta l'immagine a cui è stato applicato il cosiddetto filtro "sfuocatura". Il filtro "sfuocatura" sostituisce ogni pixel dell'immagine con il valore medio dei pixel adiacenti, in verticale, orizzontale o diagonale (e del pixel stesso).

Ad esempio, se l'immagine originaria era quella mostrata sopra, allora l'immagine sfuocata diventa:

```
P2  5  5  9
 2  1  1  0  0
 3  2  2  0  0
 4  4  5  2  0
 4  5  6  3  2
 4  5  7  4  3
```

ovvero:



Ad esempio, il pixel in basso a destra dell'immagine sfumata ha come valore la media dei pixel adiacenti nell'immagine originaria, cioè (5+0+5+3)/4=3, mentre il pixel al centro dell'immagine ha valore (9+0+0+9+7+0+9+7+5)/9=5.

Si scriva un programma C strutturato come segue. Lo studente può aggiungere tutte le funzioni che desidera (ed è consigliabile farlo), e può aggiungere parametri alle funzioni richieste (se lo ritiene necessario).

1. Nel `main`, si invochi una procedura o funzione di lettura (punto 2), una di elaborazione (punto 3) ed una di scrittura del file con l'immagine sfuocata (punto 5)

2. Si scriva la procedura o funzione di lettura; tale procedura o funzione deve leggere il file `immagine.pgm` e portarne il contenuto in una opportuna struttura. Si mostri a video il contenuto del file

3. Si scriva una procedura o funzione di elaborazione che prende come parametro l'immagine letta al punto 2 (più, eventualmente, altri parametri) e fornisce al `main` l'immagine sfumata. Per fare questo, invoca una funzione `media` (da definire al punto 4) per ogni pixel dell'immagine.

4. Si scriva una funzione `media`, che prende in ingresso

   - un'immagine
   - le coordinate di un pixel

e fornisce il valore medio delle intensità del pixel stesso e di quelli adiacenti.

5. Si scriva una procedura o funzione che salvi sul file `sfuocata.pgm` l'immagine sfumata, nello stesso formato PGM.

Facoltativo (punti 4)

Esiste anche un altro formato PGM che occupa meno memoria. In questa seconda modalità, il file è di tipo binario. Viene salvata un'intestazione simile alla precedente, costituita da una riga contenente, separati da spazi:

- la stringa `"P5"`
- la larghezza W dell'immagine (intero, massimo 100)
- l'altezza H dell'immagine (intero, massimo 100)
- il numero S di sfumature (al massimo 255)

Questa prima riga si può scrivere con una `fprintf`, nonostante il file sia binario. Il resto del file contiene il valore di intensità dei singoli pixel (salvati con la normale modalità dei file binari); ciascun pixel occupa esattamente **1 byte** e i suoi valori possono andare da 0 a 255 (lo studente pensi quindi a quale tipo di dato utilizzare).

Si modifichi il programma in modo che il file di uscita sia dato come file PGM binario (il file di ingresso viene sempre fornito in modalità file di testo).

Si prendano i nomi dei file di ingresso e di uscita come parametri dalla linea di comando; ad esempio, se l'utente invoca il programma con

```
nomeprogramma figura.pgm modificata.pgm
```

il programma dovrà leggere il file `figura.pgm` e salvare la versione sfumata

- in formato binario nel file `modificata.pgm`
- e nel formato testo nel file `sfuocata.pgm` .

Si provi il programma anche con il file `h.pgm`.

Si consegnino i file:

- un file `COGNOME.c` (dove COGNOME va sostituito col cognome dello studente) che contiene il `main` e le funzioni usate solo nell'esercizio base
- un file `facoltativo.c` che contiene il `main` e le funzioni usate solo nell'esercizio facoltativo
- un file `funzioni.c` che contiene le funzioni comuni

più tutti i file header ritenuti necessari.

Soluzione

```
#include <stdio.h>
#define MAXSIZE 100

typedef struct
{  int MaxX, MaxY, MaxCol;
   unsigned char M[MAXSIZE][MAXSIZE];
} pgm;

void read_pgm(pgm *I)
{
   int x,y;
   FILE *fp;
   fp = fopen("immagine.pgm","rt");
   fscanf(fp,"P2 %d %d %d\n",&(*I).MaxX,&(*I).MaxY,&(*I).MaxCol);

   for(y=0;y<(*I).MaxY;y++)
   {
      for(x=0;x<(*I).MaxX;x++)
      {
         fscanf(fp,"%d ",&((*I).M[x][y]));
         printf("%d\t",(*I).M[x][y]);
      }
      printf("\n");
   }
   fclose(fp);
}

void write_pgm(pgm I)
{
   int x,y;
   FILE *fp;
   fp = fopen("sfuocata.pgm","wt");
   fprintf(fp,"P2 %d %d %d\n",I.MaxX,I.MaxY,I.MaxCol);

   for(y=0;y<I.MaxY;y++)
   {
      for(x=0;x<I.MaxX;x++)
         fprintf(fp,"%d ",I.M[x][y]);
      fprintf(fp,"\n");
   }
   fclose(fp);
}
```

```
int minimo(int a, int b)
{  if (a<b)
      return a;
   else return b;
}
int massimo(int a, int b)
{  if (a>b)
      return a;
   else return b;
}

unsigned char media(pgm I, int x, int y)
{
   int Xmin,Xmax,Ymin,Ymax,count=0,i,j,sum=0;
   Xmin = massimo(x-1,0);
   Xmax = minimo(x+1,I.MaxX-1);
   Ymin = massimo(y-1,0);
   Ymax = minimo(y+1,I.MaxY-1);
   for (i=Xmin;i<=Xmax;i++)
      for (j=Ymin;j<=Ymax;j++)
      {  sum = sum + I.M[i][j];
         count++;
      }
   return (unsigned char)(sum/count);
}

pgm blur(pgm I)
{  int x,y;
   pgm B;
   B.MaxX=I.MaxX;
   B.MaxY=I.MaxY;
   B.MaxCol = I.MaxCol;
   for (x=0;x<I.MaxX;x++)
      for (y=0;y<I.MaxY;y++)
         B.M[x][y] = media(I,x,y);
   return B;
}

main()
{
   pgm I,B;
   read_pgm(&I);
   B=blur(I);
   write_pgm(B);
}
```