MAC0115 – Introdução à Computação para Ciências Exatas e Tecnologia

Instituto de Física — Turma 21 — Segundo Semestre de 2008

Quarto Exercício-Programa — Prazo de entrega: até **12 de dezembro de 2008.**

Filtro da Mediana

Seja A uma matriz de inteiros positivos com m linhas e n colunas, e sejam p e q dois inteiros positivos ímpares. Dada uma coordenada (i, j) em A, a vizinhança de tamanho $p \times q$ em torno de (i, j) é a submatriz A_{ij} de A com p linhas e q colunas e centro em (i, j).

Por exemplo, dada a seguinte matriz 5×5

$$\begin{bmatrix} 9 & 4 & 5 & 0 & 8 \\ 10 & 3 & 2 & 1 & 7 \\ 9 & 1 & 6 & 3 & 15 \\ 0 & 3 & 8 & 10 & 1 \\ 1 & 16 & 9 & 12 & 7 \end{bmatrix}$$

a vizinhança 3×3 em torno de $(1, 1)$ é a submatriz

$$\begin{bmatrix} 9 & 4 & 5 \\ 10 & 3 & 2 \\ 9 & 1 & 6 \end{bmatrix}$$

Note que a vizinhança não é bem definida em algumas coordenadas (por exemplo, em $(0, 0)$).

1 O filtro da mediana

Filtro da mediana é uma transformação bastante comum para suavizar ruídos do tipo impulso em sinais e imagens digitais.

Uma imagem digital pode ser representada por uma matriz. Dada uma matriz A de inteiros positivos com m linhas e n colunas, e dois inteiros positivos e ímpares, p e q, o filtro da mediana calcula uma matriz Med com o mesmo tamanho de A, de forma que $Med(i, j)$ contém a mediana dos números em A_{ij} (a vizinhança $p \times q$ em torno de (i, j)).

No caso do exemplo anterior, os números em torno de $(1, 1)$ são $9, 4, 5, 10, 3, 2, 9, 1, 6$. Logo, $Med(1, 1) = 5$. Quando a vizinhança de uma coordenada (i, j) não estiver bem definida, usaremos a convenção $Med(i, j) = 0$.

No caso da matriz-exemplo acima, o resultado da mediana por uma vizinhança 3×3 é a seguinte matriz:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 5 & 3 & 5 & 0 \\ 0 & 3 & 3 & 6 & 0 \\ 0 & 6 & 8 & 8 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

2 O que o seu programa deve fazer

Você deverá escrever um programa que:

- lê uma imagem de um arquivo e a armazena em uma matriz (mais detalhes na seções 3 e 4),
- calcula o resultado do filtro da mediana, Med, conforme descrito na seção 1, e
- grava a imagem Med em um arquivo (mais detalhes nas seções 3 e 4)

O seu programa deverá também imprimir as matrizes quando as mesmas forem menores que 16×16 (veja um exemplo de saída do programa na seção 5).

3 Formato PGM

Neste EP utilizaremos o formato PGM (*Portable Gray Map*) para armazenar imagens em arquivos. Segundo este formato, o arquivo deve conter um cabeçalho e a matriz correspondente à imagem. Veja exemplo a seguir.

```
P2
5 4
16
9  4  5 0  8
10 3  2 1  7
9  1  6 3  15
1  16 9 12 7
```

A primeira linha do arquivo contém uma palavra-chave "`P2`" que é obrigatória. A segunda linha contém dois números que correspondem ao número de colunas e linhas da matriz, respectivamente. A terceira linha contém um número que é o maior número da imagem (*maxval*). Para fins deste EP, *maxval* é no máximo 255. Os demais números do arquivo correspondem aos tons de cinza da imagem armazenados em forma de uma matriz de inteiros. Cada tom de cinza é um número entre 0 e *maxval*, com 0 indicando "negro" e *maxval* indicando "branco".

O formato PGM também permite colocar comentários. Caracteres após o caractere '`#`'até o próximo fim de linha (caractere '`\n`') são comentários e são ignorados. Um exemplo de imagem com comentários:

```
P2
# imagem: exemplo.pgm
5 4
16
9  4  5 0  8
10 3  2 1  7
9  1  6 3  15
1  16 9 12 7
```

4 Arquivo de imagens: leitura e escrita

O programa a seguir mostra como fazer a leitura do cabeçalho de um arquivo PGM.

```c
#include <stdio.h>
#include <string.h>

#define MAX_NAME 256   /* tamanho maximo para nome de arquivo */
#define MAX      512   /* dimensao maxima para matrizes */

int main()
{

    FILE *arq;
    char fname[MAX_NAME];
    char key[128];
    int  m, n, maxval, a[MAX][MAX];
    int  aux, i, j;

    /* leitura do nome do arquivo de entrada */
    printf("Digite o nome do arquivo de entrada: ");
    scanf("%s", fname);

    /* abre arquivo para leitura */
    arq = fopen(fname, "r");
    if (arq == NULL) {
        printf("Erro na abertura do arquivo %s\n", fname);
        return 0;
    }

    /* le dados do cabecalho */
    aux = fscanf(arq, "%s", key);
    if (aux != 1) {
        printf("Erro na leitura do arquivo %s\n", fname);
        fclose(arq);
        return 0;
    }
    if (strcmp(key, "P2") != 0) {
        printf("Formato desconhecido\n");
        fclose(arq);
        return 0;
    }
    aux = fscanf(arq, "%d %d %d", &m, &n, &maxval);
    if (aux != 3) {
        printf("Formato incorreto\n");
        fclose(arq);
        return 0;
    }

    /*
       ...
       le a matriz (imagem) que se segue
       ...
    */

    fclose(arq);   /* fecha arquivo */
    return 0;
}
```

Baseando-se nesse exemplo, escreva uma função para ler um arquivo no formato PGM e outra para escrever um arquivo no formato PGM. Essas funções deverão ter os seguintes protótipos:

```
/* ----------------------------------------------
   Funcao que le um arquivo no formato PGM.
     fname   : nome do arquivo PGM
     M       : matriz correspondente a imagem lida
     *m      : numero de linhas da matriz
     *n      : numero de colunas da matriz
     *maxval : maior valor na matriz
   ---------------------------------------------- */
int read_pgm(char fname[], int M[][MAX], int *m, int *n, int *maxval);


/* ----------------------------------------------
   Funcao que escreve num arquivo, no formato PGM.
     fname   : nome do arquivo PGM
     M       : matriz correspondente a imagem a ser gravada
     m       : numero de linhas da matriz
     n       : numero de colunas da matriz
     maxval  : maior valor na matriz
   ---------------------------------------------- */
int write_pgm(char fname[], int M[][MAX], int m, int n, int maxval);
```

5 Exemplo

```
Digite o nome do arquivo de entrada: exemplo.pgm

Matriz original:

   9   4   5   0   8
  10   3   2   1   7
   9   1   6   3  15
   0   3   8  10   1
   1  16   9  12   7

Digite o tamanho da vizinhanca (p q): 3 3

Calculando mediana, aguarde...

Matriz mediana:

   0   0   0   0   0
   0   5   3   5   0
   0   3   3   6   0
   0   6   8   8   0
   0   0   0   0   0

Digite o nome do arquivo de saida: sai.pgm
```

6 Outras informações

Você pode encontrar alguns arquivos de entrada para testar o seu programa em `http://www.ime.usp.br/~\/reverbel/mac115-IAG-07/#EPs`.

Para visualizar uma imagem no formato PGM, use qualquer visualizador que entenda este formato, como por exemplo o IrfanView, que roda no Windows e pode ser obtido gratuitamente no sítio `http://www.irfanview.com/`. Você pode instalar o IrfanView no seu computador e utilizá-lo para visualizar tanto as imagens de entrada como as de saída. Se você rodar Linux, use algum dos visualizadores para Linux, como o `eog` e o `kview`.

Observações Importantes sobre os Exercícios-Programa

1) Todos os exercícios-programa devem ter um cabeçalho como o seguinte:

```
/*****************************************************/
/* Aluno: Fulano de Tal                              */
/* Número USP: 12345678                              */
/* Exercicio-Programa 4 -- Filtro da Mediana         */
/* MAC115 -- 2008 -- IFUSP, turma 21 -- Prof. Reverbel  */
/* Compilador: ... (gcc ou DevC++) versão ...        */
/*****************************************************/
```

2) O exercício-programa é estritamente individual. Exercícios copiados (com ou sem eventuais disfarces) receberão nota ZERO.
3) Exercícios atrasados não serão aceitos.
4) Exercícios com erros de sintaxe (ou seja, erros de compilação) receberão nota ZERO. Seu programa deve ser compilável sem erros ou *warnings*, com o compilador no modo em que todos os *warnings* possíveis são emitidos. Caso você use o `gcc`, passe ao compilador (na linha de comando) as opções "`-Wall -ansi -pedantic -O2`". Caso você use o `DevC++`, clique em "Ferramentas" (ou "Tools") e "Opções do Compilador" (ou "*Compiler Options*") e, na tela de opções do compilador, marque como selecionada a opção "Adicione os seguintes comandos quando chamar o compilador" (ou "*Add the following commands when calling compiler*"). Na caixa de texto que aparece logo depois dessa opção, digite "`-Wall -ansi -O2`". (Não use `-pedantic` com o DevC++.)
5) É muito importante que seu programa tenha comentários e esteja bem indentado, ou seja, digitado de maneira a ressaltar a estrutura de subordinação dos comandos do programa (conforme visto em aula). A avaliação dos exercícios-programa levará isto em conta.
6) Cada programa deve ter sido executado tantas vezes quantas forem necessárias para testar os vários casos possíveis para as entradas.
7) Você entregará seu exercício-programa através do sistema Paca/Moodle (`http://paca.ime.usp.br`).
8) Entregue apenas o programa fonte em C, num arquivo com nome ep4-<*seu-número-USP*>.c. (Exemplo: se seu número USP for 12345678, você deverá entregar um arquivo `ep4-12345678.c`.)
9) Enquanto o prazo de entrega não expirar, você poderá entregar várias versões do mesmo exercício-programa. Apenas a última versão entregue será guardada pelo sistema. Encerrado o prazo, o sistema não aceitará mais a entrega de exercícios-programa. Não deixe para entregar seu exercício na última hora!
10) Guarde uma cópia do seu exercício-programa pelo menos até o final do semestre.

Bom trabalho!