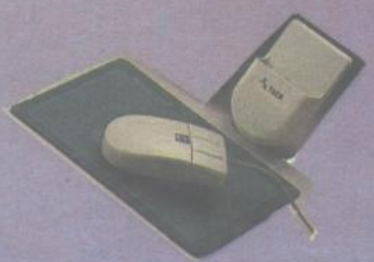
北京希望电脑公司图像处理技术丛书

C 语 言
图像处理程序集



马建波 著
赵唯一 审校

海洋出版社

北京希望电脑公司图像处理技术丛书

C 语言
图像处理程序集

马建波　著
赵唯一　审校

海洋出版社

1992·北京

内　容　摘　要

图像处理是随着计算机技术的发展而发展起来的一门新学科，其应用非常广泛。本书提供了149个实用的C语言图像处理程序，共分14章，149小节，每节都是一个完整的C程序。覆盖了图像处理技术的主要内容。内容选择上注重实用性，便于读者直接应用。

本书要求读者具有一定的C语言基础，适合于需要了解运用图像处理技术的科技人员，同时也可作为图像处理课程的补充教材，供高等院校师生和专业人员使用。

需要本书和源程序软盘的用户请与北京8721信箱联系，邮码100080，电话2562329。

（京）新登字087号

责任编辑：刘莉蕾

C语言图像处理程序集

马建波　著
赵唯一　审校

海洋出版社出版（北京市复兴门外大街1号）
海洋出版社发行　　施园印刷厂印刷
开本：787×1092 1/16　印张：32.37　字数：766千字
1992年2月第一版　　1992年2月第一次印刷

印数：1－3000　册

ISBN 7-5027-2751-5/TP　108　定价：21.00　元

前　言

随着计算机技术的飞速发展，计算机已成为当前科研、生产、决策乃至日常生活中必不可少的工具。同时，计算机（PC机）的普及、应用面的全方位展开，也提出了更高的要求，现在人们已不单纯满足于用计算机进行科学计算、文字处理、事务处理等“一维”意义上的计算任务，还希望能用计算机处理和显示图形、图像等二维信息。

图像处理技术经历从60年代至今30年的发展，目前已经进入成熟和面向应用的阶段，图像处理的应用领域极为广阔，大至空间技术、小至电子显微图片的处理，广至工业生产、军事、医学、气象、建筑、机械、土木、物理学、化学、生物、地学、电视广播、艺术、服装设计等等，到处都可以看到图像处理技术的应用，可以说已到了无孔不入的程度。之所以如此是因为图像处理技术可以节省大量的人力劳动，比如文字识别（OCR），图纸的扫描录入；还可以提高速度和精度，如某些应用中的测量工作；另外图像处理中的编码技术更是与日常生活密切相关。具体的应用数不胜数，如果说图像图形处理是为今软件技术的支柱是丝毫不过份的，图形、图像处理是今后软件发展的潮流和趋势。

本书正是在许多领域的科技人员渴望了解图像处理，应用图像处理的形势下编写的。全书包括149个完整、实用的C语言图像处理程序，覆盖了图像处理的主要内容，每个程序都经过严格的调试，适应的读者面也比较广，初学者可通过本书学习运用图像处理的理论和编程方法，即使对图像处理领域的专业人员本书的程序也是很适用的。

全书的程序用Microsoft C6.0编写，编写过程中考虑了与ANSI C的兼容，读者可很容易修改为在Turbo C, Borland C++和Unix机上的C语言图像处理程序。程序有的是一个主函数写成的，有的是主函数调用若干子函数；表示图像的数组大小设为256×256，有的是静态数组，有的是动态分配和释放的内存块限于篇幅，每个程序的开头只对算法作了简要的叙述，并给出了参考文献，供读者进一步了解算法的细节。程序中也加入了详尽的英文注释。为使读者更清楚的了解图像图形领域的发展与应用，在全书的最后附上三井秀树（日）所著的《计算机图形学的世界——探寻图像革命的最前言》一书，由刘连光先生翻译。

马建波

目　　录

第 1 章　二值化 …… (1)
1.1 非0象素置1二值化'BINAF0' …… (1)
1.2 固定阈值法二值化'BINAF1' …… (3)
1.3 双固定阈值法二值化'BINAF2' …… (4)
1.4 各象素分别取阈值二值化'BINAF3' …… (6)
1.5 判断分析法二值化'HANBET' …… (8)
1.6 P-参数法二值化'PTILE' …… (11)
1.7 微分直方图法二值化'DIFHIST' …… (14)
1.8 基于灰度差直方图的阈值选取'DIFTHR' …… (17)
1.9 基于熵的阈值选取'ENTROT' …… (20)
第 2 章　灰度变换 …… (25)
2.1 图像剪取'CLIPPI' …… (25)
2.2 反像的生成'NEGA' …… (27)
2.3 负数灰度值变换为正数'NPLUS' …… (29)
2.4 实数灰度值变换为整数'NLEVEL' …… (30)
2.5 整数灰度值变换为实数'NODO01' …… (32)
2.6 锯齿波变换'NOKOG1' …… (34)
2.7 灰度级的线性变换'GRAYTR' …… (36)
2.8 灰度的对数变换'GRAYLG' …… (37)
2.9 灰度等高线的抽出'TOUKOU' …… (39)
2.10 灰度分布的正规化'NBUNPU' …… (42)
2.11 直方图均衡化'HISTEQ' …… (44)
2.12 灰度直方图的计算'NHIST1' …… (46)
2.13 累积直方图的计算'NHIST2' …… (47)
第 3 章　噪声消除 …… (50)
3.1 二值图形麻点噪声的滤除'GOMA2C' …… (50)
3.2 孤立黑象素的消除'KORITU' …… (52)
3.3 3*3平均值滤'FILT33' …… (55)
3.4 N*N平均值滤'FILTNN' …… (58)
3.5 选择局域平滑滤波'SELAVR' …… (60)
3.6 N*N中值滤波'MEDIAN' …… (64)
3.7 十字型中值滤波'CNEDIA' …… (67)
3.8 N*N最频值滤波'MODFIL' …… (71)
第 4 章　微分运算 …… (75)
4.1 纵横方向的微分运算'BIBUN1' …… (75)
4.2 双向一次微分运算'BIBUN2' …… (78)
4.3 二次微分(1)'NJIB1' …… (80)
4.4 二次微分(2)'NJIB2' …… (83)
4.5 二次微分(3)'NJIB3' …… (86)

4.6 拉普拉斯运算'LAPLAC' …… (90)
4.7 kirsch算子边缘检测'KIRSCH' …… (93)
4.8 Prewitt算子微分运算'PREWIT' …… (97)
4.9 Sobel微分运算'SOBEL' …… (101)
4.10 Roberts微分运算'ROBERT' …… (105)
4.11 Robinson边缘检测'ROBINS' …… (109)
4.12 Frei&Chen边缘检测'FRCHEN' …… (113)
第 5 章 投影量计算 …… (117)
5.1 垂直/水平投影量计算'TOUEI1' …… (117)
5.2 对角线/反对角线方向投影量计算'TOUEI2' …… (119)
第 6 章 黑区域处理 …… (123)
6.1 二值图形的区域标记'RLABEL' …… (123)
6.2 二值图形的小区域消除'SMLDEL' …… (127)
6.3 二值图形闭区域的复杂度计算'RGNSIZ' …… (132)
6.4 二值图形的收缩(1)'SHRIN1' …… (137)
6.5 二值图形的收缩(2)'SHRIN2' …… (141)
6.6 黑区域的轮廓线抽出'RINKAK' …… (145)
6.7 二值图形的边界跟踪'KYOKAI' …… (147)
第 7 章 特征提取 …… (154)
7.1 基于交叉数表示特征点'TOKUCH' …… (154)
7.2 拐点检测'KADO' …… (157)
7.3 检测二值图形的外接矩形'GAISET' …… (161)
7.4 二值图形的孔数检测'HOLESU' …… (164)
7.5 计算图形的欧拉数'EULER' …… (167)
7.6 基于阈值的方向码'HCODET' …… (171)
7.7 基于最长方向法的方向码'HCODEL' …… (178)
7.8 计算二值图形白区域的闭合率'TOJIRI' …… (183)
7.9 计算图像的矩特征'MOMENT' …… (189)
7.10 计算图像的重心矩'CMOMEN' …… (191)
7.11 计算图像重心位置'JUSIN' …… (193)
7.12 计算惯量主轴'SHUJIK' …… (196)
7.13 计算二值图形闭区域的复杂度'FUKUZA' …… (198)
7.14 计算闭曲线的偏角微分函数'HENKAK' …… (203)
第 8 章 图像间的运算 …… (209)
8.1 图像间象素的算术运算'SANJUT' …… (209)
8.2 二值图形间象素的逻辑运算'BLOGIC' …… (213)
8.3 计算二值图形间的hamming 距离'HMDIST' …… (217)
8.4 计算图像间的相似度'RUIJID' …… (219)
8.5 计算图像间的周期卷积'CONVO1' …… (222)
8.6 计算图像间的非周期卷积'CONVO2' …… (225)
8.7 图像的常数四则运算'TEISUU' …… (228)
8.8 图像指定区域赋值'DAINYO' …… (231)
8.9 图像间指定区域拷贝'TENSHA' …… (234)
8.10 图像行列转置'TENCHI' …… (237)
8.11 图像纵/横方向的断面抽出'DANMEN' …… (239)
8.12 图像的数据类型转换'TYPECV' …… (242)

第 9 章　几何变换 …… (245)
9.1　图像的整数倍放大'SZKAKU' …… (246)
9.2　图像的整数倍缩小'SZCNV1' …… (247)
9.3　图像的实数倍缩放(1)'SZCNV1' …… (250)
9.4　图像的实数倍缩放(2)'SZCNV2' …… (253)
9.5　图像的实数倍缩放(3)'SZCNV3' …… (255)
9.6　图像的指定区域缩放'ZOOM' …… (258)
9.7　图像的平移'HEIKOU' …… (261)
9.8　图像的90度单位旋转'R90DEG' …… (263)
9.9　最近邻法图像旋转'ROTAT1' …… (266)
9.10　线性插值法图像旋转'ROTAT2' …… (269)
9.11　3次插值法图像旋转'ROTAT3' …… (271)
9.12　图像的倾斜变换'KEISHA' …… (274)
9.13　图像桶形失真的校正'TARUGA' …… (277)
9.14　二值图形的膨胀收缩'BOUCHO' …… (279)
9.15　二值图形的粗化(1)'FUTOM1' …… (284)
9.16　二值图形的粗化(2)'FUTOM2' …… (286)
第 10 章　线图形处理 …… (289)
10.1　Hilditch算法细化'HILDIT' …… (289)
10.2　Deutch算法细化'DEUTCH' …… (295)
10.3　Rosenfeld 8-连接细化'ROSEN8' …… (300)
10.4　Rosenfeld 4-连接细化'ROSEN4' …… (305)
10.5　基于连接数的灰度图像细化'GRTHIN' …… (311)
10.6　线图形的单纯化'SENBUN' …… (315)
10.7　线图形的短线消除'EDAJOK' …… (319)
10.8　线图形的窄缝连接'SHTGAP' …… (323)
10.9　Spline函数的计算'SPLINE' …… (329)
10.10　Pavlidis的经典细化算法'PAVLID1' …… (332)
10.11　Pavlidis的异步细化算法'PAVLID2' …… (336)
10.12　Zhang的快速并行细化算法'FSTHIN' …… (341)
10.13　Naccache的SPTA细化算法'SPTA' …… (346)
第 11 章　数学变换 …… (351)
11.1　二维快速Fourier变换'XYFFT' …… (351)
11.2　二维快速Fourier逆变换'IXYFFT' …… (355)
11.3　二维Walsh变换'XYWALS' …… (360)
第 12 章　纹理分析 …… (364)
12.1　空间自相关函数的计算'ACORRE' …… (364)
12.2　灰度参数的计算'NPARAM' …… (368)
12.3　Hough变换(1)'HOUGH1' …… (372)
12.4　Hough变换(2)'HOUGH' …… (375)
12.5　灰度共生矩阵计算'KYOUKI' …… (378)
12.6　基于共生矩阵的纹理分析'TEXTUR' …… (382)
12.7　基于KS检验的区域分割'KOLSMI' …… (387)
第 13 章　图像的编码压缩 …… (392)
13.1　Freeman链码'FREMA1' …… (392)

13.2 Freeman链码解码'FREMA2' …… (396)
13.3 街区距离变换'CITYBD' …… (399)
13.4 街区距离变换的骨架'CITYKK' …… (402)
13.5 街区距离变换骨架的复原'CITYFG' …… (406)
13.6 棋盘距离变换'CHESSBD' …… (408)
13.7 棋盘距离变换的骨架'CHESKK' …… (411)
13.8 棋盘距离变换骨架的复原'CHESFG' …… (415)
13.9 关于灰度分量的距离变换'GWDIST' …… (418)
13.10 十进制游程长编码'RLENDE' …… (421)
13.11 十进制游程长解码'RLENDD' …… (424)
13.12 Wyle法游程长编码'RLENWE' …… (426)
13.13 Wyle法游程长解码'RLENWD' …… (430)
13.14 二比特分隔的游程长编码'RLEN2E' …… (432)
13.15 二比特分隔的游程长解码'RLEN2D' …… (435)
13.16 斜角距离变换'DISTANCE' …… (438)
第 14 章 图像显示 …… (443)
14.1 灰度值的16进制显示'PRTHEX' …… (443)
14.2 灰度图像的字符显示'CHRDOT' …… (445)
14.3 随机DITHER显示'DITHER' …… (447)
14.4 结构Dither显示'DITHES' …… (449)
14.5 平均值Dither显示'DITHEM' …… (451)
14.6 平均误差最小Dither显示'DITHEE' …… (453)
14.7 二值图像显示'DOTMAP' …… (455)
14.8 二值图像的2级灰度显示'DOT2' …… (457)
14.9 灰度图像5级灰度显示'DOT5' …… (459)
14.10 灰度图像10级灰度显示'DOT10' …… (461)
14.11 灰度图像17级灰度显示'DOT17' …… (463)
14.12 灰度伪彩色显示'GIJCOL' …… (466)
14.13 直方图显示'DISHIST' …… (467)
14.14 灰度图像的半色调显示'HALFTONE' …… (469)
14.15 灰度/彩色图像显示'IMAGEDIS' …… (473)
参考文献 …… (479)
附录:计算机图形学的世界 …… (482)
1.计算机图形学的前夕 …… (483)
2.计算机图形学的元年 …… ( )
3.盛开的计算机艺术之花 …… ( )
4.不断扩大的应用范围 …… ( )
5.现代艺术与计算机 …… ( )
6.计算机图形学的未来 …… ( )

第 1 章　二值化

在数字图像处理中，二值图像占有非常重要的地位。特别是在实用的图像处理系统中，以二值图像处理为核心构成的系统是很多的。要进行二值图像的处理与分析，首先需要把灰度图像二值化，得到二值图像。

所谓二值化，就是通过设定阈值(threshold)把灰度图像变换成仅用两个值分别表示目标和背景的二值图像。图像二值化可根据下列的阈值处理(thresholding)来进行：

$$g(i,j) = \begin{cases} 1 \ ; \ f(i,j) >= t \ \text{时} \\ 0 \ ; \ f(i,j) < t \ \text{时} \end{cases}$$

通常，用最后的二值图像 $g(i,j)$ 中的 1 值部分表示目标子图，0 值部分表示背景子图。

确定二值化阈值 t 的方法叫做阈值选择，大多数的阈值选择法以图像的灰度概率密度函数（直方图）为依据。

本章给出了9个二值化程序，其中前三个是人工选择阈值的，后五个是自动进行阈值选择的。

1.1　BINAF0--非0像素置1二值化
1.2　BINAF1--固定阈值法二值化
1.3　BINAF2--双固定阈值法二值化
1.4　BINAF3--各像素分别取阈值二值化
1.5　HANBET--判断分析法二值化
1.6　PTILE---P-参数法二值化
1.7　DIFHIST-微分直方图法二值化
1.8　DIFTHR--基于灰度差直方图的阈值选取
1.9　ENTROT--基于熵(Entropy)的阈值选取

1.1　非0像素置1二值化'BINAF0'

对于灰度图像 f，把灰度值非 0 的所有像素变为 1，其他为 0，得到二值图像 $g(i,j)$。这种方法仅在背景像素多为 0 时有效。

程序　1.1　非0像素置1二值化'BINAF0.C'

```
/*-------------------------------------------------------------------------*
    Set Non-Zero Pixels to 1   Program
    Input  : origin image file
    Output : result image file
    File   : BINAF0.C
```

```
    Compile: cl -AL BINAF0.C
 *-------------------------------------------------------------------*/
#include <stdio.h>
#define IMAGE-SIZE-ROW 256
#define IMAGE-SIZE-COL 256
#define ONE          255
#define ZERO         0
unsigned char origin-image[IMAGE-SIZE-ROW][IMAGE-SIZE-COL];
unsigned char result-image[IMAGE-SIZE-ROW][IMAGE-SIZE-COL];
main(argc,argv)
int argc;
char *argv[];
{
  FILE *I-file,*O-file;
  char origin-image-file-name[80];
  char result-image-file-name[80];
  int i,j;
/*
 * prompt user to input arguments.
 */
  printf("origin-image-file-name -->");
  scanf("%s",origin-image-file-name);printf("\n");
  printf("result-image-file-name-->");
  scanf("%s",result-image-file-name);printf("\n");
/*
 * read image file from DISK to MEMORY.
 */
 if((I-file=fopen(origin-image-file-name,"rb"))==NULL)
   { printf("file open error!");exit(-1);}
 if((O-file=fopen(result-image-file-name,"wb"))==NULL)
   { printf("file open error!");exit(-1);}
for(i=0;i<IMAGE-SIZE-ROW;i++)
fread(origin-image[i],sizeof(unsigned char),IMAGE-SIZE-COL,I-file);
fclose(I-file);
/*
 * set non-0 pixels to 1.
 */
  for(i=0;i<IMAGE-SIZE-ROW;i++)
  for(j=0;j<IMAGE-SIZE-COL;j++)
      {
```

```
            result-image[i][j]=(origin-image[i][j]==0)?ZERO:ONE;
        }
/*
 *  saving changing results to result-file.
 */
  for(i=0;i<IMAGE-SIZE-ROW;i++)
  fwrite(result-image[i],sizeof(unsigned char),IMAGE-SIZE-COL,O-file);
  fclose(O-file);
}
/*==="BINAF0"=====================Copyright:Ma Jian-bo(1992/03)===*/
```

1.2 固定阈值法二值化'BINAF1'

对于灰度图像 f，小于阈值 t 的像素 $f(i,j)$ 变为 0，大于等于阈值 t 的像素变为 1。阈值 t 是由用户输入的。

程序 1.2 固定阈值法二值化'BINAF1.C'

```
/*-------------------------------------------------------------------*
     Single Threshold Binarization Program
     Input  : origin image file , threshold
     Output : result image file
     File   : BINAF1.C
     Compile: cl -AL BINAF1.C
 *-------------------------------------------------------------------*/
#include <stdio.h>
#define IMAGE-SIZE-ROW 256
#define IMAGE-SIZE-COL 256
#define ONE        255
#define ZERO       0
unsigned char origin-image[IMAGE-SIZE-ROW][IMAGE-SIZE-COL];
unsigned char result-image[IMAGE-SIZE-ROW][IMAGE-SIZE-COL];
main(argc,argv)
int argc;
char *argv[];
{
  FILE *I-file,*O-file;
  char origin-image-file-name[80];
  char result-image-file-name[80];
  int threshold;
  int i,j;
/*
```

```
 * prompt user to input arguments.
 */
  printf("origin-image-file-name -->");
  scanf("%s",origin-image-file-name);printf("\n");
  printf("result-image-file-name-->");
  scanf("%s",result-image-file-name);printf("\n");
  printf("threshold (0..255)-->");
  scanf("%d",&threshold);printf("\n");
/*
 * read image file from DISK to MEMORY.
 */
 if((I-file=fopen(origin-image-file-name,"rb"))==NULL)
   { printf("file open error!");exit(-1);}
 if((O-file=fopen(result-image-file-name,"wb"))==NULL)
   { printf("file open error!");exit(-1);}
for(i=0;i<IMAGE-SIZE-ROW;i++)
fread(origin-image[i],sizeof(unsigned char),IMAGE-SIZE-COL,I-file);
fclose(I-file);
/*
 * do thresholding.
 */
  for(i=0;i<IMAGE-SIZE-ROW;i++)
  for(j=0;j<IMAGE-SIZE-COL;j++)
       {
         result-image[i][j]=(origin-image[i][j]>threshold)?ONE:ZERO;
       }
/*
 *  saving thresholding results to result-file.
 */
  for(i=0;i<IMAGE-SIZE-ROW;i++)
  fwrite(result-image[i],sizeof(unsigned char),IMAGE-SIZE-COL,O-file);
  fclose(O-file);
}
/*==="BINAF1"======================Copyright:Ma Jian-bo(1992/03)===*/
```

1.3 双固定阈值法二值化'BINAF2'

对于灰度图像 f，设定两个阈值 t_1、t_2 （$t_1<t_2$），如果像素值 $f(i,j)$ 小于 t_1 则变为 0（或 1），大于等于 t_1 而小于 t_2 变为 1（或 0），大于等于 t_2 则变为 0（或 1）。取0-1-0还是取1-0-1，由用户指定，另外，t_1、t_2 是用户给定的。

程序 1.3 双固定阈值法二值化'BINAF2.C'

```
/*-------------------------------------------------------------------*
     Double Threshold Binarization Program
     Input  : origin image file , threshold1  ,threshold2
     Output : result image file
     File   : FINAF2.C
     Compile: cl -AL BINAF2.C
 *-------------------------------------------------------------------*/
#include <stdio.h>
#define IMAGE-SIZE 256
#define ONE        255
#define ZERO       0
unsigned char origin-image[IMAGE-SIZE][IMAGE-SIZE];
unsigned char result-image[IMAGE-SIZE][IMAGE-SIZE];
main(argc,argv)
int argc;
char *argv[];
{
  FILE *I-file,*O-file;
  char origin-image-file-name[80];
  char result-image-file-name[80];
  int threshold1,threshold2;
  int i,j;
/*
 * prompt user to input arguments.
 */
  printf("origin-image-file-name -->");
  scanf("%s",origin-image-file-name);printf("\n");
  printf("result-image-file-name-->");
  scanf("%s",result-image-file-name);printf("\n");
  printf("first threshold (0..255)-->");
  scanf("%d",&threshold1);printf("\n");
  printf("second threshold (0..255)-->");
  scanf("%d",&threshold2);printf("\n");


 * read image file from DISK to MEMORY.
 */
 if((I-file=fopen(origin-image-file-name,"rb"))==NULL)
   { printf("file open error!");exit(-1);}
 if((O-file=fopen(result-image-file-name,"wb"))==NULL)
```

```
    { printf("file open error!");exit(-1);}
for(i=0;i<IMAGE-SIZE;i++)
fread(origin-image[i],sizeof(unsigned char),IMAGE-SIZE,I-file);
fclose(I-file);
/*
 * do thresholding.
 */
  for(i=0;i<IMAGE-SIZE;i++)
  for(j=0;j<IMAGE-SIZE;j++)
       {
         if (origin-image[i][j]>=threshold1 && origin-image[i][j]<threshold2)
              result-image[i][j]=ONE;
         else result-image[i][j]=ZERO;
       }
/*
 *  saving thresholding results to result-file.
 */
  for(i=0;i<IMAGE-SIZE;i++)
  fwrite(result-image[i],sizeof(unsigned char),IMAGE-SIZE,O-file);
  fclose(O-file);
}
/*==="BINAF2"=====================Copyright:Ma Jian-bo(1992/03)===*/
```

1.4 各像素分别取阈值二值化'BINAF3'

对于灰度图像 f，不用全局阈值 t，而用各个像素的阈值 $t(i,j)$ 进行二值化。即，若 $f(i,j) < t(i,j)$，则 $f(i,j) \leftarrow 0$；若 $f(i,j) > t(i,j)$ 则 $f(i,j) \leftarrow 1$。程序要求用户输入灰度图像 f 和阈值图像 t 的文件名。

程序 1.4 各像素分别取阈值二值化'BINAF3.C'

```
/*-------------------------------------------------------------*
     Pixel variant thresholding Program
     Input  : origin image file , threshold image file
     Output : result image file
     File   : BINAF3.C
     Compile: cl -AL BINAF3.C
 *-------------------------------------------------------------*/
#include <stdio.h>
#define IMAGE-SIZE 256
#define ONE        255
#define ZERO       0
```

```
unsigned char origin-image[IMAGE-SIZE][IMAGE-SIZE];
unsigned char result-image[IMAGE-SIZE][IMAGE-SIZE];
unsigned char threshold-image[IMAGE-SIZE][IMAGE-SIZE];
main(argc,argv)
int argc;
char *argv[];
{
  FILE *I-file,*O-file;
  FILE *T-file;
  char origin-image-file-name[80];
  char result-image-file-name[80];
  char threshold-image-file-name[80];
  int i,j;
/*
 * prompt user to input arguments.
 */
  printf("origin-image-file-name -->");
  scanf("%s",origin-image-file-name);printf("\n");
  printf("result-image-file-name-->");
  scanf("%s",result-image-file-name);printf("\n");
  printf("threshold-image-file-name-->");
  scanf("%s",threshold-image-file-name);printf("\n");
/*
 * read image file from DISK to MEMORY.
 */
 if((I-file=fopen(origin-image-file-name,"rb"))==NULL)
   { printf("file open error!");exit(-1);}
 if((O-file=fopen(result-image-file-name,"wb"))==NULL)
   { printf("file open error!");exit(-1);}
 if((T-file=fopen(threshold-image-file-name,"rb"))==NULL)
   { printf("file open error!");exit(-1);}
for(i=0;i<IMAGE-SIZE;i++)
fread(origin-image[i],sizeof(unsigned char),IMAGE-SIZE,I-file);
fclose(I-file);
for(i=0;i<IMAGE-SIZE;i++)
fread(threshold-image[i],sizeof(unsigned char),IMAGE-SIZE,T-file);
fclose(T-file);
/*
 * pixel variant thresholding.
 */
```

```
  for(i=0;i<IMAGE-SIZE;i++)
  for(j=0;j<IMAGE-SIZE;j++)
      {
        result-image[i][j]=
          (origin-image[i][j]<threshold-image[i][j]) ? ZERO:ONE;
      }
/*
 *  saving thresholding results to result-file.
 */
  for(i=0;i<IMAGE-SIZE;i++)
  fwrite(result-image[i],sizeof(unsigned char),IMAGE-SIZE,O-file);
  fclose(O-file);
}
/*==="BINAF3"=====================Copyright:Ma Jian-bo(1992/03)===*/
```

1.5 判断分析法二值化"HANBET"

判断分析法是从图像灰度值的直方图中把灰度值的集合用阈值 t 分成两类，然后根据两个类的平均值方差(类间方差)和各类的方差(类内方差)的比为最大来确定阈值 t。

设给定的图像在整个 1，2，…，I 中具有 L 级的灰度值，设阈值为 k，把具有 k 以上灰度值的像素和具有比它小的值的像素数分成两个类，并规定为类 1，类 2。把类 1 的像素数设为 $W_1(k)$，平均灰度值为 $M_1(k)$，方差为 $\sigma_1(k)$；把类 2 的像素数设为 $W_2(k)$，平均灰度值为 $M_2(k)$，方差为 $\sigma_2(k)$，若全体像素的平均值定为 M_T，则类内方差由下式计算：

$$\sigma_w^2=W_1\sigma_1^2+W_2\sigma_2^2$$

类间方差由下式计算：

$$\sigma_B^2=W_1(M_1-M_T)^2+W_2(M_2-M_T)^2=W_1W_2(M_1-M_2)^2$$

这里，为了使 σ_B^2/σ_w^2 变为最大，最好使 σ_B^2 为最大，也就是最好令 k 变化，从而求出使 σ_B 成为最大的 k 值。详细的算法描述见文献[20]。

程序中只利用了类间方差 σ_B，是一种自动选择阈值的方法。

程序 1.5 判断分析法二值化"HANBET.C"

```
/*-------------------------------------------------------------*
     Automatic Analysis Binarization Program
     Input  : origin image file
     Output : result image file
     File   : HANBET.C
     Compile: cl -AL HANBET.C
 *-------------------------------------------------------------*/
#include <stdio.h>
#define IMAGE-SIZE-ROW  256
#define IMAGE-SIZE-COL  256
```

```
#define ONE          255
#define ZERO         0
unsigned char origin-image[IMAGE-SIZE-ROW][IMAGE-SIZE-COL];
unsigned char result-image[IMAGE-SIZE-ROW][IMAGE-SIZE-COL];
double varience[256];                    /* working buffer */
main(argc,argv)
int argc;
char *argv[];
{
  FILE *I-file,*O-file;
  char origin-image-file-name[80];
  char result-image-file-name[80];
  int i,j,threshold;                     /*threshold is automatic selected. */
  int level,TH;
  double bmax;
  double mean,mean1,mean2;
  double counter,counter1,counter2;
  long double sum,sum1,sum2;
/*
 * prompt user to input arguments.
 */
  printf("origin-image-file-name -->");
  scanf("%s",origin-image-file-name);printf("\n");
  printf("result-image-file-name-->");
  scanf("%s",result-image-file-name);printf("\n");
/*
 * read image file from DISK to MEMORY.
 */
 if((I-file=fopen(origin-image-file-name,"rb"))==NULL)
   { printf("file open error!");exit(-1);}
 if((O-file=fopen(result-image-file-name,"wb"))==NULL)
   { printf("file open error!");exit(-1);}
for(i=0;i<IMAGE-SIZE-ROW;i++)
fread(origin-image[i],sizeof(unsigned char),IMAGE-SIZE-COL,I-file);
fclose(I-file);
/*
 * get the maximum grey level of the original image.
 */
 level=-9999;
 for(i=0;i<IMAGE-SIZE-ROW;i++)
```