2

2.1 BMP

2.1.1 BMP

BMP BMP 4 (bitmap-file header) (bitmap-information header) (co

1. (bitmap-file header)

bfType	2	BMP 0x42,0x4D ASCII “B”“M”
bfSize	4	BMP
bfReserved1	2	
bfReserved2	2	

UltraEdit BMP

ImageProcess_Image001

Image not found or type unknown

BMP 32 0x42,0x4D

4 0x36,0xF9,0x15,0x00

BMP

0x36,0xF9,0x15,0x00 0x0015F936( )

ImageProcess_Image002

Image not found or type unknown

4 0x00

4 + + 0x36

2. (bitmap-information header)

biSize	4	
biWidth	4	
biHeight	4	
biPlanes	2	1
biBitCount	2	1 1 4 8 16 24 32 24
biCompression	4	
biSizeImage	4	
biXPelsPerMeter	4	/
biYPelsPerMeter	4	/
biClrUsed	4	
biClrImportant	4	0

ImageProcess_Image003

Image not found or type unknown

0E-11 00000028h = 40, 40

12-15 00000320h = 800 800

16-19 00000258h = 600 600 X Y

1A-1B 0001h, 1

1C-1D 0018h = 24, 24 24

ImageProcess_Image004

Image not found or type unknown

1E-21 00000000h BI_RGB

22-25 00000000h BI_RGB 0

26-29 00000000h

2A-2D 00000000h

2E-31 00000000h 24 0

32-35 00000000h 24 0

3. (color palette)

24

4.

24 BGR

2.1.2 BMP RGB LCD

BMP LCD

1. LCD RGB 2 RGB565 3 RGB888 24 BMP BG
2. LCD LCD BMP

2.1 24 BMP RGB

2. 1

```
1. /*****************************************************************
```

```
 *      IsBmp
 *            BMP
 *      ptFileMap -
 *
 *       0 -  BMP  , -1 -  BMP
 ************************************************************************/
int IsBmp(FILE **ppFp, const char *strFileName)
{
    char strCheckHeader[2];
    *ppFp= fopen(strFileName, "rb+");
    if (*ppFp== NULL) {
        return -1;
    }
    if (fread(strCheckHeader, 1, 2, *ppFp) != 2)
        return -1;

    if (strCheckHeader[0] != 0x42 || strCheckHeader[1] != 0x4d)
        return -1;
    else
        return 0;
}


/************************************************************************
 *      MapFile
 *        mmap         ,
 *      PT_PictureData ptData
 *      ptData->iFileSize     :
 *                        ptData->pucFileData :
 *       0      -      -
 ************************************************************************/
int MapFile(PT_PictureData ptData)
{
    int iFd;
    struct stat tStat;

    /*     */
        iFd = fileno(ptData->ptFp);
```

```
41.     fstat(iFd, &tStat);
42.     ptData->iFileSize= tStat.st_size;
43.     ptData->pucFileData= (unsigned char *)mmap(NULL , tStat.st_size, PROT_READ |
PROT_WRITE, MAP_SHARED, iFd, 0);
44.     if (ptData->pucFileData == (unsigned char *)-1)
45.     {
46.         printf("mmap error!\n");
47.         return -1;
48.     }
49.     return 0;
50. }
51. 
52. /**********************************************************************
53.  *      DecodeBmp2Rgb
54.  *      BMP    rgb
55.  *      strFileName -
56.  *                  ptData -
57.  *       0      -      -
58.  *                         -1     -     BMP
59.  *                         -2     -     bpp
60.  *                         -3     -
61.  **********************************************************************/
62. static int DecodeBmp2Rgb(const char *strFileName, PT_PictureData ptData) {
63.     int x,y;
64.     int iPos = 0;
65.     int iLineWidthAlign;
66.     BITMAPFILEHEADER *ptBITMAPFILEHEADER;
67.     BITMAPINFOHEADER *ptBITMAPINFOHEADER;
68.     unsigned char *aFileHead;
69.     unsigned char *pucSrc;
70.     unsigned char *pucDest;
71.     int iLineBytes;
72. 
73.     /*        BMP   */
74.     if (IsBmp(&ptData->ptFp, strFileName))
75.         return -1;
76. 
77.     /*  BMP         */
78.     MapFile(ptData);
```

```
79.
80.
81.     aFileHead = ptData->pucFileData;
82.
83.     ptBITMAPFILEHEADER = (BITMAPFILEHEADER *)aFileHead;
84.     ptBITMAPINFOHEADER = (BITMAPINFOHEADER *)(aFileHead + sizeof(BITMAPFILEHEADER));
85.     /*          */
86.     ptData->iWidth = ptBITMAPINFOHEADER->biWidth;
87.     ptData->iHeight = ptBITMAPINFOHEADER->biHeight;
88.     ptData->iBpp = ptBITMAPINFOHEADER->biBitCount;
89.         iLineBytes    = ptData->iWidth*ptData->iBpp/8;//
90.     ptData->iBmpDataSize= ptData->iHeight * iLineBytes;//  BMP
91.     /*    24bpp  */
92.     if (ptData->iBpp != 24)
93.     {
94.         printf("iBMPBpp = %d\n", ptData->iBpp);
95.         printf("sizeof(BITMAPFILEHEADER) = %d\n", sizeof(BITMAPFILEHEADER));
96.         return -2;
97.     }
98.
99.     /*     */
100.     ptData->pucBmpData = malloc(ptData->iBmpDataSize);
101.     ptData->pucRgbData = malloc(ptData->iBmpDataSize);
102.
103.     if (NULL == ptData->pucBmpData||NULL == ptData->pucRgbData)
104.         return -2;
105.
106.     /*  bmp         24bpp BMP   BGR   */
107.     pucDest = ptData->pucBmpData;
108.     iLineWidthAlign = (iLineBytes + 3) & ~0x3;    /*  4   */
109.     pucSrc = aFileHead + ptBITMAPFILEHEADER->bfOffBits;
110.
111.     pucSrc = pucSrc + (ptData->iHeight - 1) * iLineWidthAlign;
112.
113.     /*   bmp                                */
114.     for (y = 0; y < ptData->iHeight; y++)
115.     {
116.         memcpy(pucDest, pucSrc, ptData->iWidth*3);
117.         pucSrc  -= iLineWidthAlign;
```

```
        pucDest += iLineBytes;
    }

    /*    BGR    RGB   */
    for (y = 0; y < ptData->iHeight; y++){
        for(x = 0;x<ptData->iWidth*3;x+=3){
            ptData->pucRgbData[iPos++] = ptData->pucBmpData[y*ptData->iWidth*3+x+2];
            ptData->pucRgbData[iPos++] = ptData->pucBmpData[y*ptData->iWidth*3+x+1];
            ptData->pucRgbData[iPos++] = ptData->pucBmpData[y*ptData->iWidth*3+x+0];
        }
    }

    return 0;

}
```

2.2 JPEG

2.2.1 JPEG libjpeg

JPEG .jpg JPEG BMP JPEG BMP JPEG JPEG

BMP JPEG Linux jpeg

libjpeg jpeg libjpeg X86 ARM libjpeg

libjpeg libjpeg libjpeg

1.

```
tar xzf libjpeg-turbo-1.2.1.tar.gz
cd libjpeg-turbo-1.2.1/
```

2.

```
tar xzf libjpeg-turbo-1.2.1.tar.gz
./configure --prefix=/work/projects/libjpeg-turbo-1.2.1/tmp/ --host=arm-linux
make
```

```
make install
```

3.

```
cd /work/projects/libjpeg-turbo-1.2.1/tmp/include
cp * /usr/local/arm/4.3.2/arm-none-linux-gnueabi/libc/usr/include
cd /work/projects/libjpeg-turbo-1.2.1/tmp/lib
cp *so* -d /usr/local/arm/4.3.2/arm-none-linux-gnueabi/libc/armv4t/lib
```

4.

```
cd /work/projects/libjpeg-turbo-1.2.1/tmp/lib
cp *.so* /work/nfs_root/fs_mini_mdev_new/lib/ -d
```

2.2.2 libjpeg

libjpeg libjpeg.txt example.c libjpeg

1. jpeg_compress_struct

```
cinfo.err = jpeg_std_error(&jerr);
jpeg_create_decompress(&cinfo);
```

2.

```
jpeg_stdio_src(&cinfo, infile);
```

1 1 jpeg_compress_struct

2 JPEG

3. jpg

```
jpeg_read_header(&cinfo, TRUE);
```

cinfo image_width image_height cinfo scale_num scale_d

4.

```
jpeg_start_decompress(&cinfo);
```

cinfo cinfo

5.

```
jpeg_read_scanlines(&cinfo, buffer, 1);
```

RGB buffer 3

6.

```
jpeg_finish_decompress(&cinfo);
```

7. jpeg_compress_struct

```
jpeg_destroy_decompress(&cinfo);
```

2.2.3 libjpeg JPEG RGB LCD

JPEG RGB

```
2.2
/*********************************************************************
 *      IsJpg
 *         Jpg
 *      ptData -
                    strFileName -
 *      0 -   JPG    - JPG
 *********************************************************************/
static int IsJpg(PT_PictureData ptData, const char *strFileName)
{
    int iRet;

    jpeg_stdio_src(&ptData->tInfo, ptData->ptFp);

    /*  jpeg_read_header  jpeg  */
    iRet = jpeg_read_header(&ptData->tInfo, TRUE);

        return (iRet == JPEG_HEADER_OK);
}

```

```
/**********************************************************************
 *      DecodeJpg2Rgb
 *      JPG    RGB888
 *      ptData -
 *                              strFileName -
 *     PT_PictureData->pucRgbData -   rgb
 *      0 -      -
 **********************************************************************/
static int DecodeJpg2Rgb(const char *strFileName, PT_PictureData ptData){
    int iRowSize;
    unsigned char *pucbuffer;
    unsigned char *pucHelp;//

    /* 1.       jpeg_compress_struct    */
    ptData->tInfo.err = jpeg_std_error(&ptData->tJerr);
    jpeg_create_decompress(&ptData->tInfo);


    /* 2.     */
    if ((ptData->ptFp= fopen(strFileName, "rb")) == NULL) {
        fprintf(stderr, "can't open %s\n", strFileName);
        return -1;
    }

    /* 3.  jpg             JPEG    */
    if (!IsJpg(ptData, strFileName)) {
    printf("file is not jpg ...\n");
    return -1;
    }


    /*         */
    ptData->tInfo.scale_num = 1;
    ptData->tInfo.scale_denom = 1;
    /* 4.      jpeg_start_decompress */
    jpeg_start_decompress(&ptData->tInfo);

```

```
    /*        tInfo           */
    ptData->iWidth= ptData->tInfo.output_width;
    ptData->iHeight = ptData->tInfo.output_height;
    ptData->iBpp = ptData->tInfo.output_components*8;
    /*         */
    iRowSize = ptData->iWidth * ptData->tInfo.output_components;
    pucbuffer = malloc(iRowSize);
    ptData->iRgbSize= iRowSize * ptData->iHeight;
    ptData->pucRgbData = malloc(ptData->iRgbSize);

    /* pucHelp  ptData->pucRgbData    */
    pucHelp = ptData->pucRgbData;
    /*  5.    jpeg_read_scanlines            */
    while (ptData->tInfo.output_scanline < ptData->tInfo.output_height)
    {
        /*   jpeg_read_scanlines       pucbuffer  */
        jpeg_read_scanlines(&ptData->tInfo, &pucbuffer, 1);
        /*            */
        memcpy(pucHelp,pucbuffer,iRowSize);
        pucHelp  += iRowSize;
    }
    free(pucbuffer);
    /* 6.     */
    jpeg_finish_decompress(&ptData->tInfo);
    /* 7.  jpeg_compress_struct    */
    jpeg_destroy_decompress(&ptData->tInfo);
    return 0;
}
```

2.3 PNG

2.3.1 PNG libpng

JPEG PNG JPEG PNG LZ77 PNG libpng

libpng

http://www.libpng.org/pub/png/libpng.html

libpng libpng libpng

1.

```
tar xzf libpng-1.6.37.tar.gz
cd libpng-1.6.37/
```

2.

```
./configure --prefix=/work/projects/libpng-1.6.37/tmp/ --host=arm-linux
make
make install
```

3.

```
cd /work/projects/libpng-1.6.37/tmp/include
cp * /usr/local/arm/4.3.2/arm-none-linux-gnueabi/libc/usr/include
cd /work/projects/libpng-1.6.37/tmp/lib
cp *so* -d /usr/local/arm/4.3.2/arm-none-linux-gnueabi/libc/armv4t/lib
```

4.

```
cd /work/projects/libpng-1.6.37/tmp/lib
cp *.so* /work/nfs_root/fs_mini_mdev_new/lib/ -d
```

2.3.2 libpng

libpng libpng-manual.txt example.c libjpeg

1. libpng png_ptr info_ptr
   A. png_ptr = png_create_read_struct(PNG_LIBPNG_VER_STRING, NULL, NULL, NULL);
   2 3 4 NULL

B. info_ptr = png_create_info_struct(png_ptr);

2.
   setjmp(png_jmpbuf(png_ptr));
   libpng png_create_read_struct
3.
   png_init_io(png_ptr, fp);
   1 1 png_ptr 2 PNG
4. PNG
   A.

png_read_png(png_ptr, info_ptr, png_transforms, png_voidp_NULL);
info_ptr png_transforms libpng
B.

png_get_image_width png_get_image_height png_get_color_type png

5. info_ptr
PNG
A.
png_read_image(png_ptr, row_pointers);
1 1 png_ptr 2
B.
row_pointers = png_get_rows(png_ptr, info_ptr);
1 2 1 png_ptr, info_ptr
1 1 png_ptr 2
6.
png_destroy_read_struct(&png_ptr, &info_ptr, 0);

2.3.3 libpng png rgb LCD

2. 3

```
/*****************************************************************************
 *      IsnotPng
 *          PNG
 *      ppFp -
 *                      strFileName -
 *      0 -  PNG    -  PNG
 *****************************************************************************/
int IsnotPng(FILE **ppFp, const char *strFileName)
{
    char strCheckHeader[8];
    *ppFp= fopen(strFileName, "rb");
    if (*ppFp== NULL) {
        return -1;
    }
    /*   PNG   8       png_sig_cmp      PNG   */
    if (fread(strCheckHeader, 1, 8, *ppFp) != 8)
        return -1;
    return png_sig_cmp(strCheckHeader, 0, 8);

}
```

```
21. 
22. /**********************************************************************
23.  *      DecodePng2Rgb
24.  *      PNG    RGB888
25.  *      ptData -
26.  *                              strFileName -
27.  *     PT_PictureData->pucRgbData -   rgb
28.  *      0 -      -
29.  **********************************************************************/
30. static int DecodePng2Rgb(const char *strFileName, PT_PictureData ptData)
31. {
32.     int i, j;
33.     int iPos = 0;
34.     png_bytepp pucPngData;
35.     /* 0.       PNG   */
36.     if (IsnotPng(&ptData->ptFp, strFileName)) {
37.         printf("file is not png ...\n");
38.         return -1;
39.     }
40. 
41.     /* 1.        libpng     png_ptr info_ptr */
42.     ptData->ptPngStrPoint  = png_create_read_struct(PNG_LIBPNG_VER_STRING, NULL, NULL,
NULL);
43.     ptData->ptPngInfoPoint= png_create_info_struct(ptData->ptPngStrPoint);
44. 
45.     /* 2.        */
46.     setjmp(png_jmpbuf(ptData->ptPngStrPoint));
47.     rewind(ptData->ptFp); //  fseek(fp, 0, SEEK_SET);
48. 
49.     /* 3.     */
50.     png_init_io(ptData->ptPngStrPoint, ptData->ptFp);
51. 
52.     /* 4.  PNG
53.       *   PNG_TRANSFORM_EXPAND
54.       *  PNG    BGR888 ABGR8888  */
55.     png_read_png(ptData->ptPngStrPoint, ptData->ptPngInfoPoint, PNG_TRANSFORM_EXPAND,
0);
56.     ptData->iChannels    = png_get_channels(ptData->ptPngStrPoint, ptData-
>ptPngInfoPoint);
```

```
57.     ptData->iWidth    = png_get_image_width(ptData->ptPngStrPoint, ptData->ptPngInfoPoint);
58.     ptData->iHeight  = png_get_image_height(ptData->ptPngStrPoint, ptData->ptPngInfoPoint);
59.
60.
61.     /* 5. info_ptr         */
62.     pucPngData = png_get_rows(ptData->ptPngStrPoint, ptData->ptPngInfoPoint);
//        png_get_rowbytes();
63.     if (ptData->iChannels == 4) { //   24   32
64.         ptData->iRawSize= ptData->iWidth * ptData->iHeight*4; //
65.         ptData->pucRawData= (unsigned char*)malloc(ptData->iRawSize);
66.         if (NULL == ptData->pucRawData) {
67.             printf("malloc rgba faile ...\n");
68.             png_destroy_read_struct(&ptData->ptPngStrPoint, &ptData->ptPngInfoPoint, 0);
69.             fclose(ptData->ptFp);
70.             return -1;
71.         }
72.         /*  pucPngData     RGBA
73.          *     ABGR  */
74.         for (i = 0; i < ptData->iHeight; i++)
75.             for (j = 0; j < ptData->iWidth * 4; j += 4) {
76.                     ptData->pucRawData[iPos++] = pucPngData[i][j + 3];
77.                     ptData->pucRawData[iPos++] = pucPngData[i][j + 2];
78.                     ptData->pucRawData[iPos++] = pucPngData[i][j + 1];
79.                     ptData->pucRawData[iPos++] = pucPngData[i][j + 0];
80.                 }
81.
82.         /*    RGBA   RGB888   */
83.         if(RgbaToRgb(ptData)!=0)
84.             return -1;
85.
86.     }
87.     else if (ptData->iChannels == 3 ) { //      24   32
88.         ptData->iRgbSize= ptData->iWidth * ptData->iHeight*3; //
89.         ptData->pucRgbData = (unsigned char*)malloc(ptData->iRgbSize);
90.         if (NULL == ptData->pucRgbData) {
91.             printf("malloc rgba faile ...\n");
92.             png_destroy_read_struct(&ptData->ptPngStrPoint, &ptData->ptPngInfoPoint, 0);
```

```
            fclose(ptData->ptFp);
            return -1;
        }
        /*  pucPngData     RGB
          *    BGR  */
        for (i = 0; i < ptData->iHeight; i ++) {
            for (j = 0; j < ptData->iWidth*3; j += 3) {
                ptData->pucRgbData[iPos++] = pucPngData[i][j+2];
                ptData->pucRgbData[iPos++] = pucPngData[i][j+1];
                ptData->pucRgbData[iPos++] = pucPngData[i][j+0];
            }
        }
        ptData->iBpp = 24;//       RGB888
    }
    else return -1;


    /* 6:     */
    png_destroy_read_struct(&ptData->ptPngStrPoint, &ptData->ptPngInfoPoint, 0);
    fclose(ptData->ptFp);


    return 0;
}
```

2.4

2.4.1

2.4.1.1

"lantianyu520" " "

200 100 400 200

(0,0) (0,0),(0,100),(200,0),(200,100)

(40,50) x 40/200=0.2 y 50/100=0.5 Dx,Dy Dx/40(

(Sx,Sy) (Dx,Dy) Sw,Sh Dw,Dh

Sx/Dx = Sw/Dw Sy/Dy = Sh/Dh

Sx = Dx * Sw/Dw Sy = Dy * Sh/Dh

2.4.1.2

2. 4

```
1. /*****************************************************************
2.  *      PicZoom
3.  *
4.  *                            ,    free
5.  *            "     "       "lantianyu520"   "     "
6.  *      ptPicData -
7.  *            fSize    -
8.  *      ptPicData->pucZoomData,
9.  *       0 -   ,     -
10.  *****************************************************************/
11. int PicZoom(PT_PictureData ptPicData,float fSize)
12. {
13.     ptPicData->iZoomWidth = ptPicData->iWidth * fSize;
14.     ptPicData->iZoomHeight= ptPicData->iHeight* fSize;
15.     unsigned long* pdwSrcXTable;
16.     unsigned long x;
17.     unsigned long y;
18.     unsigned long dwSrcY;
19.     unsigned char *pucDest;
20.     unsigned char *pucSrc;
21.     unsigned long dwPixelBytes = ptPicData->iBpp/8;
22.     ptPicData->pucZoomData= malloc(sizeof(unsigned char) * ptPicData->iZoomWidth*ptPicData-
>iZoomHeight*ptPicData->iBpp/8);
```

```
    pdwSrcXTable = malloc(sizeof(unsigned long) * ptPicData->iZoomWidth);
    if (NULL == pdwSrcXTable){
        printf("malloc error!\n");
        return -1;
    }

    /*   for     Sx = Dx * Sw/Dw Sy = Dy * Sh/Dh*/
    for (x = 0; x < ptPicData->iZoomWidth; x++){//   pdwSrcXTable
        /*   for   x
     * pdwSrcXTable[x]   Sx,
     * x   Dx,
     * ptPicData->iWidth   Sw
     * ptPicData->iZoomWidth    Dw*/
        pdwSrcXTable[x]=(x*ptPicData->iWidth/ptPicData->iZoomWidth);
    }

    for (y = 0; y < ptPicData->iZoomHeight; y++){
    /*  2    y
     * dwSrcY   Sy,
     * y   Dy,
     * ptPicData->iHeight   Sh
     * ptPicData->iZoomHeight    Dh*/
        dwSrcY = (y * ptPicData->iHeight / ptPicData->iZoomHeight);
    /*           RGB       */
        pucDest = ptPicData->pucZoomData + y*ptPicData->iZoomWidth*3;
        pucSrc  = ptPicData->pucRgbData + dwSrcY*ptPicData->iWidth*3;

    /*       */
        for (x = 0; x <ptPicData->iZoomWidth; x++){
            memcpy(pucDest+x*dwPixelBytes, pucSrc+pdwSrcXTable[x]*dwPixelBytes,
dwPixelBytes);
        }
    }

    free(pdwSrcXTable);
    return 0;
}
```

2.4.2

2.4.2.1

" " " "

1.

A B

A B B

B x0 y0 r x b a x1 y1 ,

ImageProcess_Image005

x0=rcosb y0=rsinb

x1 = rcos(b-a) = rcosbcosa+rsinbsina=x0cosa+y0sina

y1=rsin(b-a)=rsinbcosa-rcosbsina=-x0sina+y0cosa

RGB A B x1,y1 A

2.4.2.2

```
    2. 5
1.  #define PI 3. 1415926535
2. //
3. #define RADIAN( angle) ( ( angle) *PI/180. 0)
4.
5.
6.
7.
8.
9. typedef struct ConcernCoor {
```

```
    int iLTx;// left top x
    int iLTy;//left top y
    int iLBx;//left bottom x
    int iLBy;//left bottom y
    int iRTx;//right top x
    int iRTy;//right top y
    int iRBx;// right bottom x
    int iRBy;// right bottom y
}T_ConcernCoor, *PT_ConcernCoor;


/*********************************************************************
 *      max
 *
 *     x y  int
 *
 *       x y
 *********************************************************************/
static int max(int x,int y){
    return x>y?x:y;
}
/*********************************************************************
 *      PicRotate
 *
 *                            ,    free
 *                "    "   "        "
 *      ptPicData -
 *            fAngle    -     0<=angle<=360
 *      ptPicData->pucRotateData,     rgb
 *       0 -   ,     -
 *********************************************************************/
int PicRotate(PT_PictureData ptPicData,float fAngle)
{
    int i ,j;
    T_ConcernCoor tConCor,tRonCor;
    //
    //int iSrcLineSize = bitCount * srcW / 8;
    int iSrcLineSize = ptPicData->iBpp* ptPicData->iZoomWidth / 8;
    int iDesLineSize;
    int iX;//    x
    int iY; //    y
```

```
51. 
52.        /*     A         B    ,
53.         *
54.         * tConCor       B
55.         * tRonCor       B      */
56.        tConCor.iLTx = -ptPicData->iZoomWidth/2;  tConCor.iLTy = ptPicData->iZoomHeight/2;
57.     tConCor.iRTx = ptPicData->iZoomWidth/2;  tConCor.iRTy = ptPicData->iZoomHeight/2;
58.     tConCor.iLBx = -ptPicData->iZoomWidth/2; tConCor.iLBy = -ptPicData->iZoomHeight/2;
59.     tConCor.iRBx = ptPicData->iZoomWidth/2; tConCor.iRBy = -ptPicData->iZoomHeight/2;
60. 
61. 
62.     /*     B       */
63.     double sina = sin(RADIAN(fAngle));
64.     double cosa = cos(RADIAN(fAngle));
65.     tRonCor.iLTx =tConCor.iLTx * cosa + tConCor.iLTy * sina;
66.     tRonCor.iLTy = -tConCor.iLTx * sina + tConCor.iLTy * cosa;
67.     tRonCor.iRTx =tConCor.iRTx * cosa + tConCor.iRTy * sina;
68.     tRonCor.iRTy = -tConCor.iRTx * sina + tConCor.iRTy * cosa;
69.     tRonCor.iLBx = tConCor.iLBx * cosa + tConCor.iLBy * sina;
70.     tRonCor.iLBy = -tConCor.iLBx * sina + tConCor.iLBy * cosa;
71.     tRonCor.iRBx = tConCor.iRBx * cosa + tConCor.iRBy * sina;
72.     tRonCor.iRBy = -tConCor.iRBx * sina + tConCor.iRBy * cosa;
73. 
74. 
75.     /*          */
76.     ptPicData->iRotateWidth = max(abs(tRonCor.iRBx -  tRonCor.iLTx),abs(tRonCor.iRTx -
tRonCor.iLBx));
77.     ptPicData->iRotateHeight = max(abs(tRonCor.iRBy -  tRonCor.iLTy),abs(tRonCor.iRTy -
tRonCor.iLBy));
78. 
79.     /*       3           */
80.     iDesLineSize = ((ptPicData->iRotateWidth* ptPicData->iBpp+ 23) / 24) * 3 ;
81.     /*                    */
82.     ptPicData->pucRotateData = malloc(iDesLineSize * ptPicData->iRotateHeight);
83.     if(NULL == ptPicData->pucRotateData){
84.         printf("malloc error\n");
85.         return -1;
86.     }
87. 
88.     /*                 *
89.       * i,j         B  x1,y1*/
```

```
    for (i = 0; i < ptPicData->iRotateHeight; i++){
        for (j = 0; j < ptPicData->iRotateWidth; j++){
            /*    B  x,y1          iX,iY,      x0,y0 */
            iX = (j - ptPicData->iRotateWidth / 2)*cos(RADIAN(360 - fAngle)) + (-i +
ptPicData->iRotateHeight / 2)*sin(RADIAN(360 - fAngle));
            iY = -(j - ptPicData->iRotateWidth / 2)*sin(RADIAN(360 - fAngle)) + (-i +
ptPicData->iRotateHeight / 2)*cos(RADIAN(360 - fAngle));
            /*              */
            if (iX > ptPicData->iZoomWidth / 2 || iX < -ptPicData->iZoomWidth / 2 || iY >
ptPicData->iZoomHeight / 2 || iY < -ptPicData->iZoomHeight / 2){
                continue;
            }
            /*     B  x0,y0        A     */
             int iXN = iX + ptPicData->iZoomWidth / 2;
         int iYN = abs(iY - ptPicData->iZoomHeight  / 2);
            /*    */
            memcpy(&ptPicData->pucRotateData[i * iDesLineSize + j * 3],&ptPicData-
>pucZoomData[iYN * iSrcLineSize + iXN * 3],3);
        }
    }
 return 0;
}
```