Here is an example about how to recovery the document image data from the *.dgr file.

```
////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//                                                                            //
//                  recovery a document image from *.dgr file                 //
//                                                                            //
////////////////////////////////////////////////////////////////////////////////////////////////////////////////

#define MAX_ILLUSTR_LEN     128

// the head information of the *.dgr file
struct DGR_HEADER
{
    int iHdSize;            // size of header: 54+strlen(illustr)+1 (there is a '\0' at the end of illustr)
    char szFormatCode[8];           // "DGR"
    char szIllustr[MAX_ILLUSTR_LEN];    // text of arbitrary length. "#......\0"
    char szCodeType[20];            // "ASCII", "GB", "SJIS" etc
    short sCodeLen;                 // 1, 2, 4, etc
    short sBitApp;                  // "1 or 8 bit per pixel" etc
};

//the annotation information of a word
struct WORD_INFO
{
    unsigned char *pWordLabel;    // the pointer to the word label (GB code)
    short sTop;                 // the top coordinate of a word image
    short sLeft;                // the left coordinate of a word image
    short sHei;                 // the height of a word image
    short sWid;                 // the width of a word image
};

//the annotation information of a text line
struct LINE_INFO
{
    int iWordNum;             // the word number in a text line
    WORD_INFO *pWordInfo; // the pointer to the annotation information of the words in a text line
};
```

```
// the annotation information of document image
struct DOC_IMG
{
        int iImgHei;                     // the height of the document image
        int iImgWid;                     // the width of the document image
        int iLineNum;                    // the text line number in the document image
        LINE_INFO *pLineInfo;      // the pointer to the annotation information of the text lines
        unsigned char *pDocImg;    // the pointer to image data buffer
};

//////////////////////////////////////////////////////////////////////////////////////////////////////////////
//                                                                                                          //
//                    read annotation information from *.dgr file                                           //
//                  recovery the * dgr file to document image data                                          //
//                                                                                                          //
//////////////////////////////////////////////////////////////////////////////////////////////////////////////
void ReaddgrFile2Img(FILE *fp)               // fp is the file pointer to *.dgr file
{
        DGR_HEADER dgrHead;
        DOC_IMG docImg;

        // read the head information of the *.dgr file
        fread(&dgrHead.iHdSize, 4, 1, fp);
        fread(dgrHead.szFormatCode, 8, 1, fp);
        fread(dgrHead.szIllustr, (dgrHead.iHdSize - 36), 1, fp);
        fread(dgrHead.szCodeType, 20, 1, fp);
        fread(&dgrHead.sCodeLen, 2, 1, fp);
        fread(&dgrHead.sBitApp, 2, 1, fp);

        // read the height and width of the document image
        fread(&docImg.iImgHei, 4, 1, fp);
        fread(&docImg.iImgWid, 4, 1, fp);

        // allocate memory for the document image data
        docImg.pDocImg = new unsigned char [docImg.iImgHei * docImg.iImgWid];
        memset(docImg.pDocImg, 0xff, docImg.iImgHei * docImg.iImgWid);
```

```
// allocate memory for the annotation information of text lines
fread(&docImg.iLineNum, 4, 1, fp);
docImg.pLineInfo = new LINE_INFO [docImg.iLineNum];

int i, j, m, n;
unsigned char *pTmpData;
int iTmpDataSize;
short iTmpDataTop;
short iTmpDataLeft;
short iTmpDataHei;
short iTmpDataWid;

// recovery the document image line by line
for(i = 0; i < docImg.iLineNum; i++)
{
        // read the word number in the i-th text line
        fread(&docImg.pLineInfo[i].iWordNum, 4, 1, fp);

        // read the annotation information of every word in the i-th text line
        for(j = 0; j < docImg.pLineInfo[i].iWordNum; j++)
        {
           docImg.pLineInfo[i].pWordInfo[j].pWordLabel = new unsigned char [dgrHead.sCodeLen];
               fread(docImg.pLineInfo[i].pWordInfo[j].pWordLabel, dgrHead.sCodeLen, 1, fp);
               fread(&docImg.pLineInfo[i].pWordInfo[j].sTop, 2, 1, fp);
               fread(&docImg.pLineInfo[i].pWordInfo[j].sLeft, 2, 1, fp);
               fread(&docImg.pLineInfo[i].pWordInfo[j].sHei, 2, 1, fp);
               fread(&docImg.pLineInfo[i].pWordInfo[j].sWid, 2, 1, fp);

                iTmpDataTop = docImg.pLineInfo[i].pWordInfo[j].sTop;
                iTmpDataLeft = docImg.pLineInfo[i].pWordInfo[j].sLeft;
                iTmpDataHei = docImg.pLineInfo[i].pWordInfo[j].sHei;
                 iTmpDataWid = docImg.pLineInfo[i].pWordInfo[j].sWid;

                 pTmpData = new unsigned char [iTmpDataHei * iTmpDataWid];
                fread(pTmpData, iTmpDataHei * iTmpDataWid, 1, fp);
```

```
                        // write the the word data image to the document image data
                        for(m = 0; m < iTmpDataHei; m++)
                        {
                              for(n = 0; n < iTmpDataWid; n++)
                               {
                                     if(pTmpData[m * iTmpDataWid + n] != 255)
                                     {
                               docImg.pDocImg[(m + iTmpDataTop) * docImg.iImgWid + n + iTmpDataLeft]
                                 = pTmpData[m * iTmpDataWid + n];
                                       }
                               }
                        }

                        delete [] pTmpData;
                }
        }
}
```