



sina 新浪博客

大爷炒股20年坚持1指标，这次酒后终于说漏了

阅读 (53320) | 评论 (2)

故事：有谁愿意陪你远行

[新浪首页](#)[登录](#)[注册](#)

chinadsp-delete的博客

<http://blog.sina.com.cn/chinadsp> [订阅] [手机订阅][首页](#) [博文目录](#) [图片](#) [关于我](#)

正文

字体大小：大 中 小

libjpeg实现内存内位图的压缩及解压缩 (2009-07-21 20:33:27)

转 载 ▼

标签： it 分类： 多媒体(图形图像/音视频)

相信使用过的朋友应该会喜欢上libjpeg，它简单易用、压缩质量可以随意控制、并且稳定性很好，但是，官方网站给提供的libjpeg库，

不论是进行压缩时还是解压缩时，都需要用到FILE，使得我们如果想在内存中直接压缩或解压缩图像还要自己实现相应的结构，

总之，比较麻烦，尤其对初学者，更是不知从何处入手，幸运的是，libjpeg给我们提供了源代码，今天我就为大家介绍，怎样修改源代码，

使libjpeg可以非常容易的直接处理内存中的图像，而无需借助文件操作。

一、建立自己的libjpeg工程

为了修改后编译方便，也为了以后在VC 环境下容易使用libjpeg库，我们按以下步骤将libjpeg转换为VC环



无线摄像头



chinadsp-delete

微博

写留言

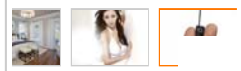
加关注


博客等级: **15**博客积分: **422**博客访问: **91,010**关注人气: **16**获赠金笔: **2**赠出金笔: **0**

荣誉徽章:



无线摄像头



境下的工程。

1、在VC环境下重新建立一个空的static library工程，工程名为libjpeg，此处注意，新建工程不要包含mfc，不要预编译头文件；

2、然后将libjpeg下的jcapimin.c jcapistd.c jccoefct.c jccolor.c jcdctmgr.c jchuff.c jcinit.c jcmaintct.c jcmarker.c jcmaster.c jcomapi.c jcparam.c jcphuff.c jcrepct.c jcsample.c jctrans.c jdapimin.c jdapistd.c jdatadst.c jdatasrc.c jdcoefct.c jdcolor.c jddctmgr.c jdihuff.c jdinpt.c jdmainct.c jdmarker.c jdmaster.c jdmerge.c jdphuff.c jdpostct.c jdsample.c jdtrans.c jerror.c jfdctflt.c jfdctfst.c jfdctint.c jidctflt.c jidctfst.c jidctint.c jidctred.c jquantl.c jquant2.c jutils.c jmemmgr.c jchuff.h jconfig.h jdihuff.h jdet.h jerror.h jinclude.h jmemsys.h jmorecfg.h jpegint.h jpeglib.h jversion.h 等文件拷贝到新工程的文件夹下，并将.c文件改名为.cpp；

3、将所有的源文件及头文件添加到新建的工程中；

4、编译新工程，此时就可以生成libjpeg.lib了。

二、分析并修改源代码

我们知道，libjpeg是利用FILE进行存取图像数据的，接下来，我们就要分析一下libjpeg是怎样利用FILE进行存取图像数据的，

然后用内存拷贝的方式替换掉所有的文件操作（I/O），也就实现了内存中进行图像压缩和解压缩的目标。

下面，先分析压缩图像时libjpeg是怎样利用FILE进行存储数据的。我们先看在进行图像压缩时，我们所调用的跟文件有关系的函数：

```
jpeg_stdio_dest(j_compress_ptr cinfo, FILE *outfile);
```

我们找到这个函数的源代码（jdatadst.cpp文件第130行）：

```
1 GLOBAL(void)
2 jpeg_stdio_dest (j_compress_ptr cinfo, FILE * outfile)
3 {
4     my_dest_ptr dest;
5
11     if(cinfo->dest==NULL) {
12         cinfo->dest = (struct jpeg_destination_mgr *)
13             (*cinfo->mem->alloc_small) ((j_common_ptr) cinfo, JPOOL_PERMANENT,
14             sizeof(my_destination_mgr));
15     }
16     dest=(my_dest_ptr)cinfo->dest;
```



李秀亭de专栏

October19, 2017Thursday星期四
周雨璐AVA

October25, 2017Wednesday星期三
周雨璐AVA

倾城容颜的清纯校花美女肤白貌美

```

17     dest->pub.init_destination = init_destination;
18     dest->pub.empty_output_buffer = empty_output_buffer;
19     dest->pub.term_destination = term_destination;
20     dest->outfile = outfile;
21 }

```

大家看第20行，函数将FILE类型的指针赋值给了dest->outfile,很显然，以后对文件的操作，就转向了对dest->outfile 的操作，我们只要找到所有引用outfile的函数，就可以知道libjpeg是怎样压缩图像到文件的，因此，我们继续搜outfile，搜索结果如下：

Find all "outfile", Subfolders, Find Results 1, "Entire Solution"

```

E:\VS2005\libjpeg\libjpeg\jpeglib.h(910):EXTERN(void) jpeg_stdio_dest JPP((j_compress_ptr cinfo,
FILE * outfile));

```

```

E:\VS2005\libjpeg\libjpeg\jdatadst.cpp(28): FILE * outfile;

```

```

E:\VS2005\libjpeg\libjpeg\jdatadst.cpp(85): if (JFWRITE(dest->outfile, dest->buffer,
OUTPUT_BUF_SIZE) !=

```

```

E:\VS2005\libjpeg\libjpeg\jdatadst.cpp(113): if (JFWRITE(dest->outfile, dest->buffer,
datacount) != datacount)

```

```

E:\VS2005\libjpeg\libjpeg\jdatadst.cpp(116): fflush(dest->outfile);

```

```

E:\VS2005\libjpeg\libjpeg\jdatadst.cpp(118): if (ferror(dest->outfile))

```

```

E:\VS2005\libjpeg\libjpeg\jdatadst.cpp(130):jpeg_stdio_dest (j_compress_ptr cinfo, FILE *
outfile)

```

```

E:\VS2005\libjpeg\libjpeg\jdatadst.cpp(150): dest->outfile = outfile;

```

Matching lines: 8 Matching files: 2 Total files searched: 57

可以看到，共有8处引用了outfile变量，第一处为函数声明，第二处为变量声明，第三、四、五、六处为文件操作，第七处和第八处我们

已经见过了，我们只需要把这八处改了就可以实现我们的目标了。如下：

```

EXTERN(void) jpeg_stdio_dest JPP((j_compress_ptr cinfo, char* outdata)); // 由EXTERN(void)
jpeg_stdio_dest JPP

```

```

((j_compress_ptr cinfo, FILE * outfile));改写

```

```

char * outdata; // 由 FILE * outfile; 改写

```

jdatadst.cpp文件第87行empty_output_buffer (j_compress_ptr cinfo)函数

```

memcpy(dest->outdata, dest->buffer, OUTPUT_BUF_SIZE);// 由JFWRITE(dest->outfile, dest->buffer,

```

用户344349348

47岁的王菲被谢霆锋甩沦为笑柄?

用户375550868

北京实创装饰——中铁建花园128平
北京实创装饰

熊头、蛇头三孔器【白玉小件·第
内蒙笑公

秦正之、肖云浦两先生谈研修孙氏
童旭东

鸡丝西红柿手擀面
异族极限



少底两类股

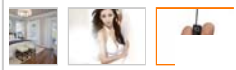
金

更多>>

共享充电宝行业要灭亡

一次建造的京都

无线摄像头



收刻尔克大撤退”

之后，为何这款私人云盘

短期调整或将仍有反复

人工智能与隐私保护

台湾科技挣扎，人祸大于天灾？

收入份额=市场份额，虎嗅想干什

OUTPUT_BUF_SIZE) 改写

jdatadst.cpp文件第114行term_destination (j_compress_ptr cinfo)

memcpy(dest->outdata, dest->buffer, datacount); // 由JFWRITE(dest->outfile, dest->buffer, datacount) 改写

删除fflush(dest->outfile);和if (ferror(dest->outfile))及相关的其它语句。

peg_stdio_dest (j_compress_ptr cinfo, char* outdata) // 由peg_stdio_dest (j_compress_ptr cinfo, FILE * outfile) 改写

dest->outdata = outdata; // 由dest->outfile = outfile; 改写

我们改到这里，可以编译一下，应该不会有错误产生，但是，你会不会觉得有问题呢？对，我们发现，我们没有为内存区域提供偏移量（

每次追加图像数据后，偏移量指向当前的位置），

另外，由于只有到压缩完才能知道图像压缩完后的数据量大小，我们还需要一个指示图像数据大小的变量。

我们将这两个变量添加到outdata后面，跟outdata一样，作为dest的成员变量，如下：

```
typedef struct {
    struct jpeg_destination_mgr pub;

    char * outdata;
    int *pSize; // 新加变量，该指针为调用者提供，压缩完后返回图像大小
    int nOutOffset; // 新加变量
    JOCTET * buffer;
} my_destination_mgr;
```

我们将通过jpeg_stdio_dest函数提供pSize指针，并在jpeg_stdio_dest的实现函数里对新添加的变量进行初始化，如下：

GLOBAL(void)

```
jpeg_stdio_dest (j_compress_ptr cinfo, char * outdata, int *pSize)
{
    my_dest_ptr dest;
```

```
    if (cinfo->dest == NULL) {
        cinfo->dest = (struct jpeg_destination_mgr *)
            (*cinfo->mem->alloc_small) ((j_common_ptr) cinfo, JPOOL_PERMANENT,
```

传奇的谢幕，谈岩田聪和他的任天堂

家常主食轻松做之——培根香葱花

[查看更多>>](#)

谁看过这篇博文

PiPi 3月24日

用户19279... 11月12日

林炳文Eva... 8月7日

飘着的云exe 7月23日



无线摄像头



7月9日

4月19日

3月4日

12月6日

8月13日

8月8日

7月30日

4月8日

```
SIZEOF(my_destination_mgr));
}
```

```
dest = (my_dest_ptr) cinfo->dest;
dest->pub.init_destination = init_destination;
dest->pub.empty_output_buffer = empty_output_buffer;
dest->pub.term_destination = term_destination;
```

```
dest->outdata = outdata;
dest->nOutOffset = 0;
dest->pSize = pSize;
*(dest->pSize)= 0;
}
```

改写声明函数

```
EXTERN(void) jpeg_stdio_dest JPP((j_compress_ptr cinfo, char* outdata, int *pSize));
```

jdatadst.cpp文件第87行empty_output_buffer (j_compress_ptr cinfo)函数

```
memcpy(dest->outdata+dest->nOutOffset, dest->buffer, OUTPUT_BUF_SIZE); // 由JFWRITE(dest->outfile,
dest->buffer,
```

OUTPUT_BUF_SIZE)改写

```
dest->nOutOffset+=OUTPUT_BUF_SIZE;
*(dest->pSize)=dest->nOutOffset;
```

jdatadst.cpp文件第114行term_destination (j_compress_ptr cinfo)

```
memcpy(dest->outdata+dest->nOutOffset, dest->buffer, datacount); // 由JFWRITE(dest->outfile,
dest->buffer, datacount)改写
```

```
dest->nOutOffset+=datacount;
*(dest->pSize)=dest->nOutOffset;
```

重新编译工程，这样我们就实现了压缩bmp位图到内存中，当然，调用jpeg_stdio_dest之前，我们需要先分配足够的内存，并把内存指针传递

给jpeg_stdio_dest函数，

好了，我们再分析libjpeg在解压缩jpg图像时，是怎样从jpg文件读入图像数据的。

我们先看我们在解压缩图像时调用的与文件操作有关的函数，如下：

```
jpeg_stdio_src (j_decompress_ptr cinfo, FILE * infile)。
```

在该函数的实现代码中找到了my_src_ptr结构，并且，我们发现与文件操作有关的该结构的成员变量为infile，参考上面内容，我们搜索infile

，搜索结果如下：

Find all "infile", Subfolders, Find Results 1, "Entire Solution"

```
E:\VS2005\libjpeg\libjpeg\jpeglib.h(911):EXTERN(void) jpeg_stdio_src JPP((j_decompress_ptr
cinfo, FILE * infile));
```

```
E:\VS2005\libjpeg\libjpeg\jdatasrc.cpp(28): FILE * infile;
```

```
E:\VS2005\libjpeg\libjpeg\jdatasrc.cpp(95): nbytes = JFREAD(src->infile, src->buffer,
INPUT_BUF_SIZE);
```

```
E:\VS2005\libjpeg\libjpeg\jdatasrc.cpp(182):jpeg_stdio_src (j_decompress_ptr cinfo, FILE *
infile)
```

```
E:\VS2005\libjpeg\libjpeg\jdatasrc.cpp(209): src->infile = infile;
```

```
Matching lines: 5 Matching files: 2 Total files searched: 57
```

根据上面的经验，我们考虑，除了将FILE *类型变量改为char *类型的变量外，还要添加两个变量，图像大小的变量及图像偏移量，这跟图像

压缩时差不多，所不同的是，

图像压缩时，图像大小是由libjpeg库返回，所以在调用是提供给libjpeg库的是个指针，而在解压缩时，图像数据大小是由调用者通过变量（

不是指针）提供给libjpeg库。

由于我详细讲解了图像压缩时的我们所做的工作，我想读者朋友们很容易就能理解解压缩时所做的更改，下面我只列出我们所改写的代码，就

不再详细讲解了。

jpeglib.h 第911行

```
EXTERN(void) jpeg_stdio_src JPP((j_decompress_ptr cinfo, char * indata,int nSize));
```

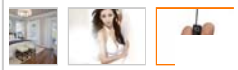
jdatasrc.cpp 第33行

```
typedef struct {
    struct jpeg_source_mgr pub;

    char * indata;
    int nInOffset;
    int nSize;
    JOCTET * buffer;
```



无线摄像头




```

    boolean start_of_file;
} my_source_mgr;

jdatasrc.cpp 第183行
GLOBAL(void)
jpeg_stdio_src (j_decompress_ptr cinfo, char * indata, int nSize)
{
    my_src_ptr src;

    if (cinfo->src == NULL) {
        cinfo->src = (struct jpeg_source_mgr *)
            (*cinfo->mem->alloc_small) ((j_common_ptr) cinfo, JPOOL_PERMANENT,
                SIZEOF(my_source_mgr));
        src = (my_src_ptr) cinfo->src;
        src->buffer = (JOCTET *)
            (*cinfo->mem->alloc_small) ((j_common_ptr) cinfo, JPOOL_PERMANENT,
                INPUT_BUF_SIZE * SIZEOF(JOCTET));
    }

    src = (my_src_ptr) cinfo->src;
    src->pub.init_source = init_source;
    src->pub.fill_input_buffer = fill_input_buffer;
    src->pub.skip_input_data = skip_input_data;
    src->pub.resync_to_restart = jpeg_resync_to_restart;
    src->pub.term_source = term_source;
    src->indata = indata;    // 新添加行
    src->nSize = nSize;      // 新添加
    src->nInOffset = 0;      // 新添加
    src->pub.bytes_in_buffer = 0;
    src->pub.next_input_byte = NULL;
}

jdatasrc.cpp 第91行
METHODDEF(boolean)
fill_input_buffer (j_decompress_ptr cinfo)
{

```



```

my_src_ptr src = (my_src_ptr) cinfo->src;
size_t nbytes;

//nbytes = JFREAD(src->infile, src->buffer, INPUT_BUF_SIZE);
nbytes = src->nSize-src->nInOffset;
if (nbytes>INPUT_BUF_SIZE) nbytes = INPUT_BUF_SIZE;

if (nbytes <= 0) {
    if (src->start_of_file)
        ERREXIT(cinfo, JERR_INPUT_EMPTY);
    WARNMS(cinfo, JWRN_JPEG_EOF);

    src->buffer[0] = (JOCTET) 0xFF;
    src->buffer[1] = (JOCTET) JPEG_EOI;
    nbytes = 2;
}

memcpy(src->buffer, src->indata+src->nInOffset, nbytes);
src->nInOffset+=nbytes;

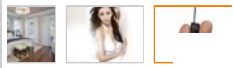
src->pub.next_input_byte = src->buffer;
src->pub.bytes_in_buffer = nbytes;
src->start_of_file = FALSE;

return TRUE;
}

```



无线摄像头



至此，libjpeg库的源代码中所有要改的东西我们都已经完成，剩下的事情就是我们编写一段测试程序测试一下。

三、编写测试代码

对于libjpeg库的详细的调用步骤，请参照我的文章《利用jpeglib压缩图像为jpg格式》，上面详细介绍了利用libjpeg库

进行图像压缩和解压缩的步骤，在编写本例的测试代码时，我们在上次提供的测试代码的基础上进行改进，如下：

无论压缩还是解压缩，与原来的libjpeg库调用不同的地方都只有一处，就是jpeg_stdio_dest和jpeg_stdio_src这两个函数的调用，

调用原来的libjpeg库时，需要为这两个函数提供已经打开的jpg文件句柄，而对于新的libjpeg库，不需要打开jpg文件了，压缩时，

我们需要提供足够大的内存区给libjpeg 库，解压缩时，只需要把存放有jpeg格式图像的内存区提供给libjpeg库就行了，下面详细介绍

对于改写后的jpeg_stdio_dest和jpeg_stdio_src这两个函数的调用方法。

1、jpeg_stdio_dest

函数的原形为：void jpeg_stdio_dest(j_compress_ptr cinfo, char * outData, int *pSize);

这里，outData指向我们提供给libjpeg库用于存放压缩后图像数据的内存区，这块内存要在我们调用该函数前申请好，大家可以看到，

我们在libjpeg库内没有对该内存区进行越界访问检查并且要足够大，否则会出现内存越界访问的危险，当整个图像压缩工作完成后，pSize

返回jpg图像数据的大小。测试代码如下：

```
char outdata[1000000]; // 用于缓存，这里设置为1000K, 实际使用时可以采用动态申请的方式
int nSize; // 用于存放压缩完后图像数据的大小
```

.....

```
jpeg_stdio_dest(&jcs, outdata, &nSize);
```

.....

2、jpeg_stdio_src

函数的原形为：void jpeg_stdio_src(j_decompress_ptr cinfo, char * inData, int nSize);

这里，inData指向我们将来进行解压缩的jpg数据，该数据我们可以直接从jpg文件中读取，也可以是通过libjpeg库在内存中直接压缩

生成的数据，nSize 当然是这个jpg数据的大小。测试代码如下：

.....

```
char indata[1000000]; // 用于存放解压缩前的图像数据，该数据直接从jpg文件读取
```

```
FILE *f = fopen(strSourceFileName, "rb");
```

```
if (f==NULL)
```

```
{
```

```
    printf("Open file error!\n");
```

```
    return;
```

```
}
```

```
int nSize = fread(outdata, 1, 1000000, f); // 读取jpg图像数据，nSize为实际读取的图像数据大小
```

```
fclose(f);
```

```
// 下面代码用于解压缩，从本行开始解压缩
```



```
jpeg_stdio_src(&cinfo, outdata, nSize);  
.....
```

至此我们所有的工作均已完成，编译并运行一下测试程序，看看效果吧，如果愿意继续交流，请给我发邮件

1

0

喜欢

赠金笔



分享:

阅读(756) | 评论(0) | 收藏(0) | 转载(0) | 喜欢▼ | 打印 | 举报

已投稿到: 排行榜

前一篇: [利用IJG JPEG Library压缩图像为jpg格式](#)

后一篇: [上社村的记忆](#)







评论





重要提示: 警惕虚假中奖信息





[\[发评论\]](#)

做第一个评论者吧! [抢沙发>>](#)

发评论

[更多>>](#)





登录名: 密码: [找回密码](#) [注册](#) ☒ 记住登录状态

☐ 评论并转载此博文

发评论


以上网友发言只代表其个人观点，不代表新浪网的观点或立场。

[< 前一篇](#)

[利用IJG JPEG Library压缩图像为jpg格式](#)

[后一篇 >](#)

[上社村的记忆](#)



新浪BLOG意见反馈留言板 不良信息反馈 电话：4006900000 提示音后按1键（按当地市话标准计费） 欢迎批评指正

[新浪简介](#) | [About Sina](#) | [广告服务](#) | [联系我们](#) | [招聘信息](#) | [网站律师](#) | [SINA English](#) | [会员注册](#) | [产品答疑](#)

Copyright © 1996 - 2017 SINA Corporation, All Rights Reserved

新浪公司 版权所有

http://blog.sina.com.cn/s/blog_5f432e6a0100dy9g.html

11/11