原 ffdshow 源代码分析 4： 位图覆盖滤镜（滤镜部分Filter）

2013年10月25日 00:15:42　阅读数：5026

==========

ffdshow源代码分析系列文章列表：

ffdshow 源代码分析 1： 整体结构

ffdshow 源代码分析 2： 位图覆盖滤镜（对话框部分Dialog）

ffdshow 源代码分析 3： 位图覆盖滤镜（设置部分Settings）

ffdshow 源代码分析 4： 位图覆盖滤镜（滤镜部分Filter）

ffdshow 源代码分析 5： 位图覆盖滤镜（总结）

ffdshow 源代码分析 6： 对解码器的dll的封装（libavcodec）

ffdshow 源代码分析 7： libavcodec视频解码器类（TvideoCodecLibavcodec）

ffdshow 源代码分析 8： 视频解码器类（TvideoCodecDec）

ffdshow 源代码分析 9： 编解码器有关类的总结

==========

第一篇文章介绍了ffdshow的位图覆盖滤镜的对话框（Dialog）部分： ffdshow 源代码分析2 ： 位图覆盖滤镜（对话框部分Dialog）

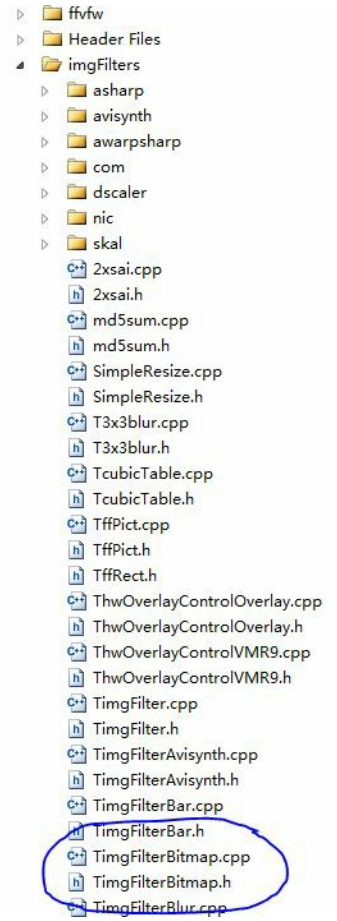
第二篇文章介绍了ffdshow的位图覆盖滤镜的设置（Settings）部分： ffdshow 源代码分析 3： 位图覆盖滤镜（设置部分Settings）

此外还有一个滤镜部分（Filter）。这三个部分就可以组成一个ffdshow的滤镜功能了。本文就来简介一下ffdshow的滤镜部分。

滤镜部分（Filter）

ffdshow的滤镜的滤镜部分(怎么感觉名字有点重复 = =,算了先这么叫吧)的功能主要用于完成具体的图像处理功能。具体到位图覆盖滤镜的话，就是用于把图片覆盖到视频上面，他是ffdshow滤镜的核心。

与位图覆盖（Bitmap）滤镜的滤镜处理有关的类位于imgFilters目录下的TimgFilterBitmap.h和TimgFilterBitmap.cpp文件中。



先来看看TimgFilterBitmap.h

这里要注意一下，该类的名字叫TimgFilterBitmap。它的声明方式确实比较奇怪：DECLARE_FILTER(TimgFilterBitmap, public, TimgFilter)

可以看出DECLARE_FILTER是一个宏，具体这个宏的内部代码就先不查看了，否则会感觉很混乱，暂且留下一个小小的谜团。在这里只要知道这是声明了一个滤镜类就可以了。

其实TimgFilterBitmap的核心函数不多，就一个，那就是process()，具体的处理功能都是在这个函数里面实现的。

```cpp
/*
 *雷霄骅
 *leixiaohua1020@126.com
 *中国传媒大学/数字电视技术
 */

#ifndef _TIMGFILTERBITMAP_H_
#define _TIMGFILTERBITMAP_H_
//叠加一张位图
#include "TimgFilter.h"
#include "Tfont.h"

struct TffPict;
struct TbitmapSettings;
//特别的声明方式 = =
DECLARE_FILTER(TimgFilterBitmap, public, TimgFilter)
private:
//图像
TffPict *bitmap;
//内存
Tbuffer bitmapbuf;
char_t oldflnm[MAX_PATH];
typedef void (*Tblendplane)(const TcspInfo &cspInfo, const unsigned int dx[3], const unsigned int dy[3], unsigned char *dst[3], cons
t stride_t dststride[3], const unsigned char *src[3], const stride_t srcstride[3], int strength, int invstrength);
//注意 这个类有一个实例，名字叫w
class TrenderedSubtitleLineBitmap : public TrenderedSubtitleWordBase
{
public:
    TrenderedSubtitleLineBitmap(void): TrenderedSubtitleWordBase(false) {}
    TffPict *pict;
    const TbitmapSettings *cfg;
    //叠加
    Tblendplane blend;
    //打印
    virtual void print(int startx, int starty /* not used */, unsigned int dx[3], int dy1[3], unsigned char *dstLn[3], const stride_
t stride[3], const unsigned char *bmp[3], const unsigned char *msk[3], REFERENCE_TIME rtStart = REFTIME_INVALID) const;
} w;
TrenderedSubtitleLine l;
//是TrenderedSubtitleLine的一个vector
TrenderedSubtitleLines ls;
int oldmode;
//几种叠加方式
template<class _mm> static void blend(const TcspInfo &cspInfo, const unsigned int dx[3], const unsigned int dy[3], unsigned char *ds
t[3], const stride_t dststride[3], const unsigned char *src[3], const stride_t srcstride[3], int strength, int invstrength);
template<class _mm> static void add(const TcspInfo &cspInfo, const unsigned int dx[3], const unsigned int dy[3], unsigned char *dst[
3], const stride_t dststride[3], const unsigned char *src[3], const stride_t srcstride[3], int strength, int invstrength);
template<class _mm> static void darken(const TcspInfo &cspInfo, const unsigned int dx[3], const unsigned int dy[3], unsigned char *d
st[3], const stride_t dststride[3], const unsigned char *src[3], const stride_t srcstride[3], int strength, int invstrength);
template<class _mm> static void lighten(const TcspInfo &cspInfo, const unsigned int dx[3], const unsigned int dy[3], unsigned char *
dst[3], const stride_t dststride[3], const unsigned char *src[3], const stride_t srcstride[3], int strength, int invstrength);
template<class _mm> static void softlight(const TcspInfo &cspInfo, const unsigned int dx[3], const unsigned int dy[3], unsigned char
 *dst[3], const stride_t dststride[3], const unsigned char *src[3], const stride_t srcstride[3], int strength, int invstrength);
template<class _mm> static void exclusion(const TcspInfo &cspInfo, const unsigned int dx[3], const unsigned int dy[3], unsigned char
 *dst[3], const stride_t dststride[3], const unsigned char *src[3], const stride_t srcstride[3], int strength, int invstrength);
//获取叠加方式
template<class _mm> static Tblendplane getBlend(int mode);
protected:
virtual bool is(const TffPictBase &pict, const TfilterSettingsVideo *cfg);
virtual uint64_t getSupportedInputColorspaces(const TfilterSettingsVideo *cfg) const
{
    return FF_CSPS_MASK_YUV_PLANAR;
}
public:
TimgFilterBitmap(IffdshowBase *Ideci, Tfilters *Iparent);
virtual ~TimgFilterBitmap();
//核心函数（Filter配置信息队列，图像，配置信息）
virtual HRESULT process(TfilterQueue::iterator it, TffPict &pict, const TfilterSettingsVideo *cfg0);
};

#endif
```

再来看看TimgFilterBitmap.cpp

这个文件本身代码量是比较大的，只是其他部分我都还没有仔细分析，确实没那没多时间。。。在这里仅简要分析一下最核心的函数process()。正是这个函数真正实现了滤镜的功能。在这个位图叠加滤镜中，process()实现了位图在视频上面的叠加功能。

```cpp
//核心函数（Filter配置信息队列，图像，配置信息）
HRESULT TimgFilterBitmap::process(TfilterQueue::iterator it, TffPict &pict, const TfilterSettingsVideo *cfg0)
{
    //都有这一句= =
    if (is(pict, cfg0)) {
        //Bitmap的配置信息
        const TbitmapSettings *cfg = (const TbitmapSettings*)cfg0;
        init(pict, cfg->full, cfg->half);
        unsigned char *dst[4];
        bool cspChanged = getCurNext(FF_CSPS_MASK_YUV_PLANAR, pict, cfg->full, COPYMODE_DEF, dst);
        //处理
        if (!bitmap || cspChanged || stricmp(oldflnm, cfg->flnm) != 0) {
            ff_strncpy(oldflnm, cfg->flnm, countof(oldflnm));
            if (bitmap) {
                delete bitmap;
            }
            //新建一张图
            //通过cfg->flnm路径
            //载入bitmapbuf
            bitmap = new TffPict(csp2, cfg->flnm, bitmapbuf, deci);
            //3个颜色分量？
            for (int i = 0; i < 3; i++) {
                w.dx[i] = bitmap->rectFull.dx >> bitmap->cspInfo.shiftX[i];
                w.dy[i] = bitmap->rectFull.dy >> bitmap->cspInfo.shiftY[i];
                w.bmp[i] = bitmap->data[i];
                w.bmpmskstride[i] = bitmap->stride[i];
            }
            w.dxChar = w.dx[0];
            w.dyChar = w.dy[0];
        }

        if (bitmap->rectFull.dx != 0) {
            if (oldmode != cfg->mode)
                if (Tconfig::cpu_flags & FF_CPU_SSE2) {
                    //获取叠加方式（SSE2）
                    //在cfg的mode里
                    w.blend = getBlend<Tsse2>(oldmode = cfg->mode);
                } else {
                    //获取叠加方式（MMX）
                    w.blend = getBlend<Tmmx>(oldmode = cfg->mode);
                }
            //输出到屏幕上的设置
            TprintPrefs prefs(deci, NULL);
            //各种参数
            prefs.dx = dx2[0];
            prefs.dy = dy2[0];
            prefs.xpos = cfg->posx;
            prefs.ypos = cfg->posy;
            //模式不同的话
            if (cfg->posmode == 1) {
                prefs.xpos *= -1;
                prefs.ypos *= -1;
            }
            prefs.align = cfg->align;
            prefs.linespacing = 100;
            prefs.csp = pict.csp;
            w.pict = &pict;
            w.cfg = cfg;
            //打印，需要用到TprintPrefs
            ls.print(prefs, dst, stride2);
        }
    }
    //最后都是这一句？
    return parent->processSample(++it, pict);
}
```


文章标签： ffdshow 滤镜 位图叠加 directshow 源代码

个人分类： ffdshow

所属专栏： 开源多媒体项目源代码分析