

LiveVideoStackCon

# 聚音视 研修不止于形

2017年10月20日-21日

北京·丽亭华苑酒店

LiveVideoStackCon

# 全平台硬件解码的渲染优化

王斌

PPTV/移动端播放器技术经理

- 常规方法渲染硬解数据的问题
- 硬解纹理转换一般思路
- D3D11 转 OpenGL ES 纹理
- macOS、iOS 纹理转换及统一
- Android 硬解渲染及常见难题解决





- 主要介绍纹理加载



- 软解OpenGL渲染流程

准备纹理



`glGenTextures()`



`glTexParameterf()`  
`glTexImage2D(...)`

渲染前更新纹理



`glBindTexture()`

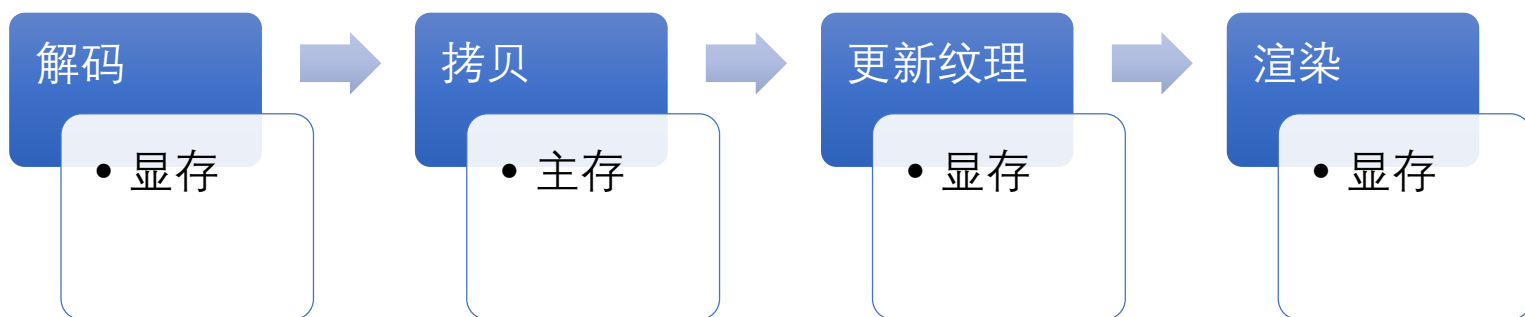


`glTexSubImage2D(..., dataptr)`

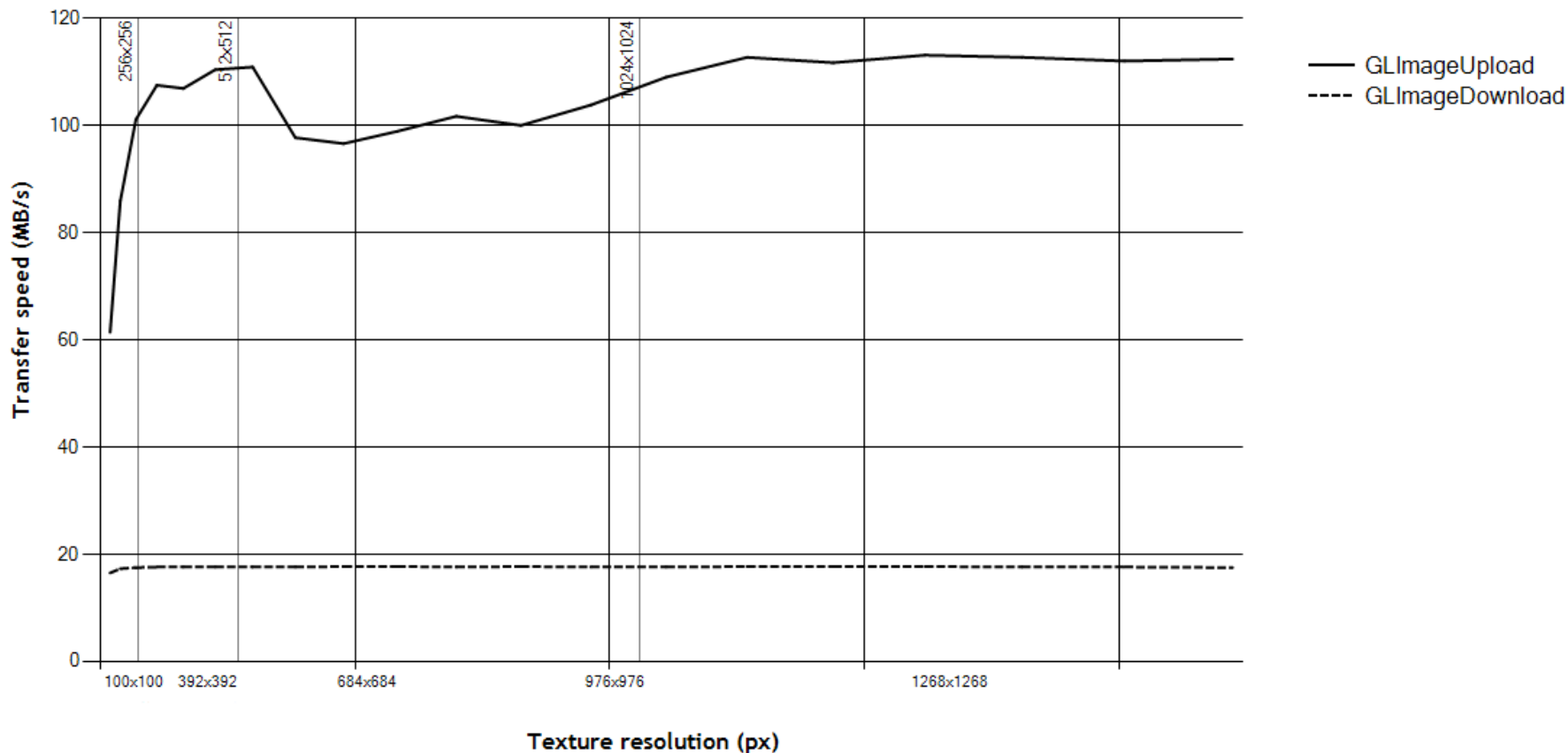
- 软解数据流



- 常规方法渲染的数据流



- `glTexSubImage2D` : CPU、GPU之间内存不共享
- 从显存拷贝非常慢，甚至不如软解，如DXVA2



<http://uu.diva-portal.org/smash/get/diva2:641061/FULLTEXT01.pdf>



- 理想的硬解数据流



目标：从解码开始到渲染结束，全程GPU处理数据，CPU只发送指令

方式：API间数据共享，让GPU拷贝、转换

- 输出的是什么：对象、格式(RGB, YUV), 尽量使用YUV
- 平台提供的共享、转换接口: apple, android
- 硬解库提供的共享、转换接口: CUDA, libva-glx
- **EGL 扩展:**
  - EGLImage (OMX IL, MMAL, VA-API): ClientBuffer, Pixmap(RGB)
  - EGLStream (OMX AL, D3D11): producer/consumer
- OpenGL扩展:
  - Windows: WGL\_NV\_DX\_interop (DXVA2, D3D11)
  - Linux: GLX\_EXT\_texture\_from\_pixmap(VA-API), GL\_NV\_vdpau\_interop

- **EGL扩展提供的D3D资源共享功能**
- ANGLE：开源的EGL+OpenGL ES实现，OpenGL ES指令转成D3D9/D3D11，兼容性好。支持Windows Store, Windows Desktop (XP+)
- 应用：
  - Chromium, Firefox, Qt...
  - iOS & android 应用移植

- 解码输出：ID3D11VideoDecoderOutputView->GetResource()->ID3D11Texture2D
- 格式：NV12、P010
- 方法一：PBuffer扩展+D3D11 VideoProcessor转RGB
- 方法二：EGLStream扩展 (2016)，直接共享，无需转换
- **EGL\_ANGLE\_stream\_producer\_d3d\_texture\_nv12**
- **GL\_NV\_EGL\_stream\_consumer\_external**

## 准备纹理

eglCreateStreamKHR

glGenTextures

eglStreamConsumerGLTextureExternalAttribsNV

eglCreateStreamProducerD3DTextureNV12ANGLE

## 更新、渲染纹理

eglStreamPostD3DTextureNV12ANGLE (共享)

eglStreamConsumerAcquireKHR

渲染纹理

eglStreamConsumerAcquireKHR

- 平台提供的纹理共享接口
- 解码器：VideoToolbox
- 解码输出：CVPixelBufferRef
- 输出格式：UYVY422(文档、官方例子使用), RGB, YUV420P, NV12
- 性能对比: NV12>UYVY422>RGB>YUV420P

- IOSurface/CVOpenGLTextureCache(RGB)

准备纹理

`glGenTextures()`

`glTexParameter()`  
`glTexImage2D(...)`

更新纹理

`glBindTexture()`

`CVPixelBufferGetIOSurface(  
buf)`

`CGLTexImageIOSurface2D(  
..., surface, p)`

- CVOpenGLTextureCache

准备纹理缓存



CVOpenGLTextureCache  
Create()

获取纹理



CVOpenGLTextureCacheC  
reateTextureFromImage(...  
cache, cvpixbuf, ...)



CVOpenGLTextureGetName



- iOS11公开了IOSurface框架
- IOSurface: buffer sharing across processes
- CVPixelBufferGetIOSurface: 存在于 macOS10.6+/iOS4.0+ CoreVideo.framework, iOS11才加入头文件
- EAGLContext.texImageIOSurface: 只存在于iOS11
- 与macOS的CGLTexImageIOSurface2D 参数相似

```
@interface EAGLContext(IOSurface)
- (BOOL)texImageIOSurface:(IOSurfaceRef)ioSurface target:(NSUInteger)target internalFormat:
(NSUInteger)internalFormat width:(uint32_t)width height:(uint32_t)height format:
(NSUInteger)format type:(NSUInteger)type plane:(uint32_t)plane NS_AVAILABLE_IOS(11_0);
@end
extern CGLLError CGLTexImageIOSurface2D(CGLContextObj ctx, GLenum target, GLenum internal_format,
GLsizei width, GLsizei height, GLenum format, GLenum type, IOSurfaceRef
ioSurface, GLuint plane) OPENGLE_AVAILABLE(10_6);
```

```
glGenTextures() glTexParameteri()
```

生成、设置纹理（一次）

绑定第p平面纹理

```
glBindTexture(..., tex)
```

```
IOSurfaceRef surface = CVPixelBufferGetIOSurface(buf);
```

```
iOS: [[EAGLContext currentContext] texImageIOSurface:surface ...];  
macOS: CGLTexImageIOSurface2D(CGLGetCurrentContext(), ..., surface, p)
```

```
glBindTexture(..., 0)
```

```
- thread #18: tid = 0x5264, 0x009280e6 libglInterpose.dylib`EAGLContext_texImageIOSurface_target_internalFormat_width_height_format_type_plane_invert(EAGLContext*, objc_selector*, __IOSurface*, unsigned int, unsigned int, unsigned int, unsigned int, unsigned int, unsigned int, unsigned int, signed char) + 442, stop reason = EXC_BAD_ACCESS (code=1, address=0xcb4)
  frame #0: 0x009280e6 libglInterpose.dylib`EAGLContext_texImageIOSurface_target_internalFormat_width_height_format_type_plane_invert(EAGLContext_, objc_selector_, __IOSurface_, unsigned int, unsigned int, unsigned int, unsigned int, unsigned int, unsigned int, unsigned int, signed char) + 442
  frame #1: 0x2240ef66 CoreVideo`CVOpenGLContext::texImageIOSurface(unsigned int, unsigned int, int, int, unsigned int, unsigned int, __IOSurface_, unsigned int) + 58
  frame #2: 0x224201c8 CoreVideo`CVPixelBufferTextureBacking::createStandardTexture(CVImageBuffer*) + 508
  frame #3: 0x224204aa CoreVideo`CVPixelBufferTextureBacking::initWithPixelBackingContext(CVPixelBufferBacking_, CVOpenGLContext*, __CFDictionary const_, CVImageBuffer_, int_) + 182
  frame #4: 0x2241deac CoreVideo`CVPixelBufferTextureBacking::createTextureBackingForContext(__CFAllocator const_, CVOpenGLContext_, __CFDictionary const_, CVImageBuffer_, int_) + 68
  frame #5: 0x2241be72 CoreVideo`CVOpenGLTextureCache::createTextureBackingFromImageBacking(__CFAllocator const*, CVImageBacking*, CVImageBuffer*, __CFDictionary const*, int*) + 36
  frame #6: 0x2241bc7c CoreVideo`CVOpenGLTextureCache::createTextureFromImageWithParams(__CFAllocator const_, CVImageBuffer_, unsigned int, int, int, int, unsigned int, unsigned int, unsigned long, int*) + 232
  frame #7: 0x2241a220 CoreVideo`CVOpenGLTextureCacheCreateTextureFromImage + 132
```

- 未公开接口

```
WangBindeMacBook-Air:mkspecs wangbin$ nm -a ~/Library/Developer/Xcode/iOS\ DeviceSupport/5.1.1\ \ (9B206\)/Symbols/System/Library/Frameworks/OpenGL.framework/OpenGL |grep texImageIOSurface
301021b0 t -[EAGLContext texImageIOSurface:target:internalFormat:width:height:format:type:plane:invert:]
WangBindeMacBook-Air:mkspecs wangbin$ nm -a ~/Library/Developer/Xcode/iOS\ DeviceSupport/11.0.1\ \ (15A402\)/Symbols/System/Library/Frameworks/OpenGL.framework/OpenGL |grep texImageIOSurface
0000000183a35de8 t -[EAGLContext texImageIOSurface:target:internalFormat:width:height:format:type:plane:]
0000000183a35e20 t -[EAGLContext texImageIOSurface:target:internalFormat:width:height:format:type:plane:invert:]
```

- 所有iOS版本

```
if (@available(iOS 11, *))
    [[EAGLContext currentContext] texImageIOSurface:s target:tgt internalFormat:ifmt
    width:w height:h format:fmt type:t plane:p];
else
    [[EAGLContext currentContext] texImageIOSurface:s target:tgt internalFormat:ifmt
    width:w height:h format:fmt type:t plane:p invert:NO];
```

- MediaCodec : java, NDK (android 5.0+)
- OMX AL
- ~~OMX IL: EGLImage所需扩展非公开(GraphicBuffer作为ClientBuffer)~~
- ~~OMX IL: 7.0不支持非NDK系统库调用~~

- 平台提供的纹理转换接口
- 解码输出: ~~ByteBuffer~~, Surface(**RGB**)
- 如何构造Surface: ~~SurfaceView~~, SurfaceTexture

解码线程输出准备



SurfaceTexture(**texName**)



Surface(surfaceTexture)



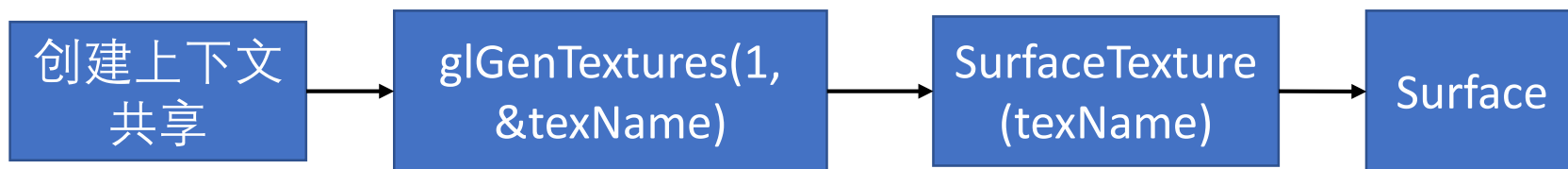
MediaCodec.configure(...,  
surface,...)

渲染线程更新纹理



surfaceTexture.updateTex  
Image()

- SurfaceTexture构造问题：texName？
- KODI(XBMC)等播放器的处理方法：等待渲染线程生成texName
- 新方法：
- 1. 共享上下文



- 2. 无效的texName, 渲染线程attachToGLContext()

解码线程输出准备

SurfaceTexture(0)

Surface(surfaceTexture)

MediaCodec.configure(...,  
surface,...)

渲染线程更新纹理

glGenTextures(1, &tex)  
surfaceTexture.attachToGLCo  
ncontext(tex)

**调用一次**

SurfaceTexture.updateTexIm  
age()

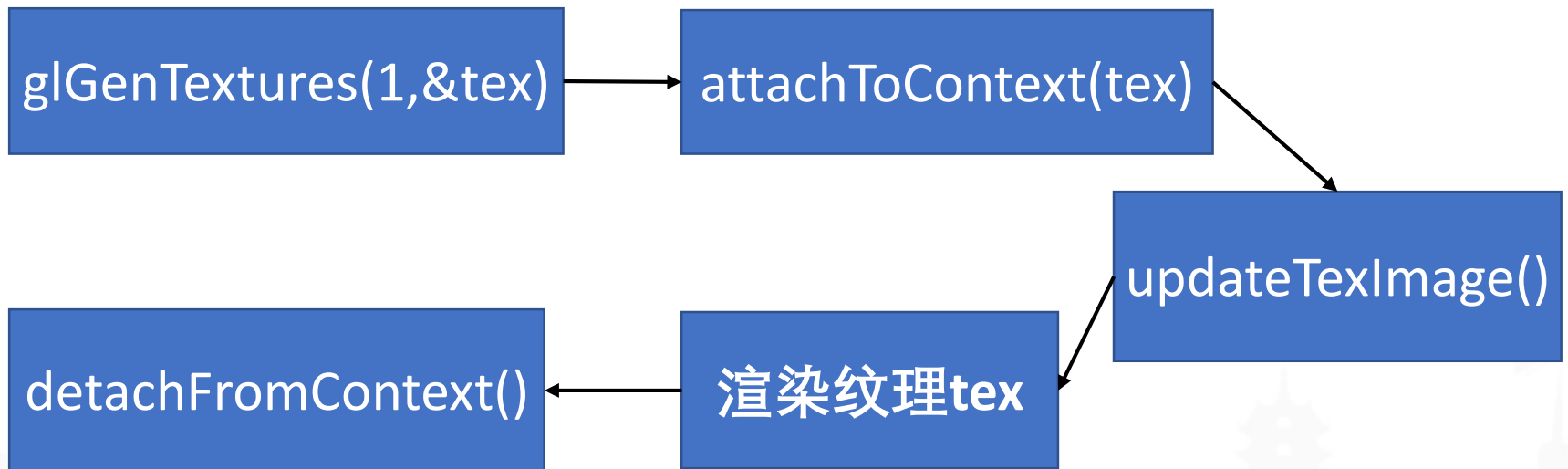


- 后台切回前台的updateTexImage()错误

E/GLConsumer: [SurfaceTexture-0-8349-0]  
checkAndUpdateEglState: invalid current EGLContext

- 常见解决方法：
  - View销毁后重新初始化解码器
  - 使用TextureView
- 如何支持SurfaceView?
- SurfaceTexture.attachToGLContext(): 检测并保存上下文
- SurfaceTexture上下文：需要共享，适用SurfaceView，不适用外部提供的上下文如GLSurfaceView，上下文切换的消耗

- 如何支持任意View ?
- 每次渲染：
  - SurfaceTexture.attachToGLContext()
  - 必须detachFromContext(), 但会删除纹理



- EGLStream: 不支持 KHR\_stream\_producer\_aldatalocator

```
typedef struct XADDataSink_ {  
    void * pLocator;  
    void * pFormat;  
} XADDataSink;  
  
typedef struct XADDataLocator_NativeDisplay_{  
    XAuint32 locatorType;  
    XANativeHandle hWindow;  
    XANativeHandle hDisplay;  
} XADDataLocator_NativeDisplay;
```

- hWindow: ANativeWindow, 从Surface获取
- hDisplay: NULL
- 渲染流程和MediaCodec一样

# Thank You

手机：15921810163

[wbsecg1@gmail.com](mailto:wbsecg1@gmail.com)



LiveVideoStackCon

聚音视 研修不止于形



关注LiveVideoStack公众号

回复 **王斌** 为讲师评分