RTP(载荷H264码流)解包与封包

标签: 音视频开发

一、H264介绍

1.1 H264概述

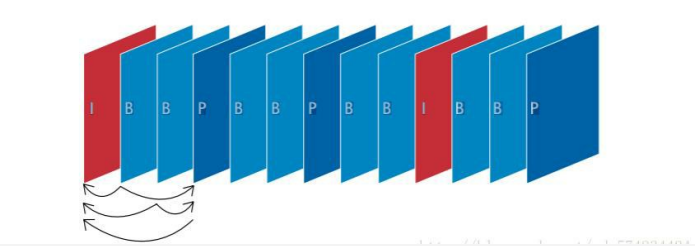
h264是一种视频压缩标准。

经过压缩后的帧分为：I帧，P帧和B帧：

- I帧：关键帧，采用帧内压缩技术。（自身可以通过视频解压算法解压成一张单独的完整的图片）
- P帧：向前参考帧，在压缩时，只参考前面已经处理的帧（只需要参考前面的I帧或P帧）。采用帧音压缩技术。
- B帧：双向参考帧，在压缩时，它即参考前而的帧，又参考它后面的帧（需要同时参考前面和后面的I帧或P帧）。采用帧间压缩技术。

除了I/P/B帧外，还有图像序列GOP。

- GOP：两个I帧之间是一个图像序列，在一个图像序列中只有一个I帧



（图出自H.264视频压缩标准白皮书）

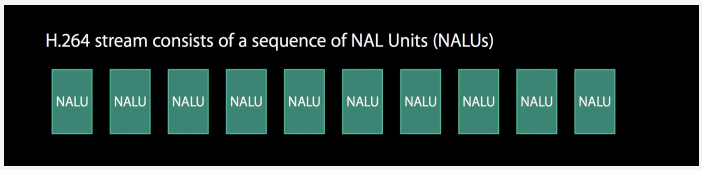
在H.264基准类中，仅使用I帧和P帧以实现低延时，因此是网络摄像机和视频编码器的理想选择。

1.2 H264原始码流结构

H264功能分为两层，VCL(视频编码层)和 NAL(网络提取层)

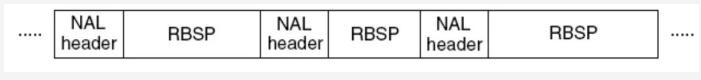
- VCL：包括核心压缩引擎和块，宏块和片的语法级别定义，设计目标是尽可能地独立于网络进行高效的编码。
- NAL：负责将VCL产生的比特字符串适配到各种各样的网络和多元环境中，覆盖了所有片级以上的语法级别。

VCL数据传输或者存储之前，会被映射到一个NALU中，H264数据包含一个个NALU。如下图：



一个NALU = 一组对应于视频编码的NALU头部信息 + 一个原始字节序列负荷(RBSP, Raw Byte Sequence Payload)

NAL单元格式如下图：



一个原始的NALU单元结构包含：
[StartCode]+[NALU Header]+[NALU Payload] 三部分（StartCode，是一个NALU单元开始，必须是00 00 00 01 或者00 00 01）

H.264 的编码视频序列包括一系列的NAL 单元，每个NAL 单元包含一个RBSP。编码片（包括数据分割片IDR 片）和序列RBSP 结束符被定义为VCL NAL 单元，其余为NAL 单元。典型的RBSP 单元序列如图2 所示。每个单元都按独立的NAL 单元传送。单元的信息头（一个字节）定义了RBSP 单元的类型，NAL 单元的其余部分为RBSP 数据。

图 2 RBSP 序列举例

1.3 NAL单元

每个NAL单元是一个一定语法元素的可变长字节字符串，包括包含一个字节的头信息（用来表示数据类型），以及若干整数字节的负荷数据。一个NAL单元可以携带一个编码片、A/B/C型数据分割或一个序列或图像参数集。

NALU头由一个字节组成，它的语法如下：

NAL单元按RTP***按序传送。其中，T为负荷数据类型，占5bit；R为重要性指示位，占2个bit；最后的F为禁止位，占1bit。具体如下：

学习笔记之ANDROID篇——SERVICE(2.0)
C++之STRING类
ANDROID STUDIO：通过ARTIFACTORY搭建本地仓库优化编译速度
转，JAVA H5 复杂TABLE导出EXCEL并下载
WDATEPICKER时间插件
基于GEC6818智能家居的实现--点亮LED灯
【LEETCODE每日一题】[困难]57. 插入区间

相关标签
C++
QT
算法
RTP 流媒体
编码解码
音视频开发
H.264
H.265
AVCC
HVCC

- NALU类型位：可以表示NALU的32种不同类型特征，类型1～12是H.264定义的，类型24～31是用于H.264以外的，RTP负荷规范使用这其中的一些值来定义包聚合和分裂，其他值为H.264保留。
- 重要性指示位：用于在重构过程中标记一个NAL单元的重要性，值越大，越重要。值为0表示这个NAL单元没有用于预测，因此可被解码器抛弃而不会有错误扩散；值高于0表示此NAL单元要用于无漂移重构，且值越高，对此NAL单元丢失的影响越大。
- 禁止位：编码中默认值为0，当网络识别此单元中存在比特错误时，可将其设为1，以便接收方丢掉该单元，主要 用于适应不同种类的网络环境（比如有线无线相结合的环境）。

264常见的帧头数据为：

00 00 00 01 **67** (SPS)

00 00 00 01 **68** (PPS)

00 00 00 01 **65** (IDR 帧)

00 00 00 01 **61** (P帧)

上述的**67,68,65,61，**还有41等，都是该NALU的识别级别。

F：禁止为，0表示正常，1表示错误，一般都是0

NRI：重要级别，11表示非常重要。

TYPE：表示该NALU的类型是什么，

见下表，由此可知7为序列参数集(SPS)，8为图像参数集(PPS)，5代表帧。1代表非帧。

由此可知，61和41其实都是P帧（type值为1），只是重要级别不一样（它们的NRI一个是11BIN，一个是10BIN）

NALU类型是我们判断帧类型的利器，从官方文档中得出如下图：

nal_unit_type	NAL类型	C
0	未使用	
1	不分区、非 IDR 图像的片	2, 3, 4
2	片分区 A	2
3	片分区 B	3
4	片分区 C	4
5	IDR 图像中的片	2, 3
6	补充增强信息单元（SEI）	5
7	序列参数集	0
8	图像参数集	1
9	分界符	6
10	序列结束	7
11	码流结束	8
12	填充	9
13..23	保留	
24..31	未使用	

1.4 H264帧判断

最上面图的码流对应的数据来层层分析，以00 00 00 01分割之后的下一个字节就是NALU类型，将其转为二进制数据后，

解读顺序为从左往右算，如下:

（1）第1位禁止位，值为1表示语法出错

（2）第2~3位为参考级别

（3）第4~8为是nal单元类型

例如上面00000001后有67,68以及65

其中0x67的二进制码为：

0110 0111

4-8为00111，转为十进制7，参考第二幅图：7对应序列参数集SPS

其中0x68的二进制码为：

0110 1000
4-8为01000，转为十进制8，参考第二幅图：8对应图像参数集PPS

其中0x65的二进制码为：

011 00101

4-8位为00101，转为十进制5，参考第二幅图：5对应IDR图像中的片(I帧)

所以判断是否为I帧的算法为：

（NALU类型 & 0001 1111） = 5 即 (NALU类型 & 31) = 5
比如0x65 & 31 = 5

即：int value = buf[4] & 0x0f; // 5是帧, 7是sps 8是pps

二、RTP打包发送H264之封包详解

> *RFC3984是H.264的baseline码流在RTP方式下传输的规范，这里只讨论FU-A分包方式，*

H264的码流结构

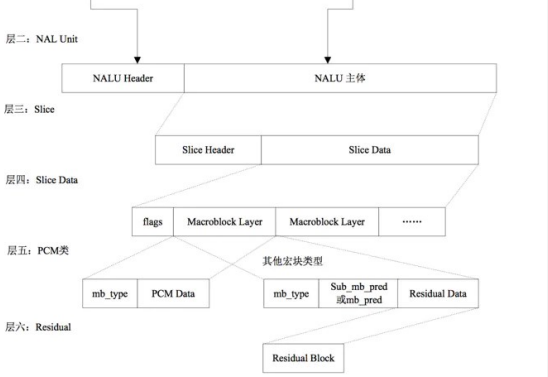


图 3 H.264 码流分层结构

1、单个NAL包单元

12字节的RTP头后面的就是音视频数据，比较简单。一个封装单个NAL单元包到RTP的NAL单元流的RTP序号必须符合NAL单元的解码顺序。
对于 NALU 的长度小于 MTU 大小的包, 一般采用单一 NAL 单元模式.
对于一个原始的 H.264 NALU 单元常由[Start Code] [NALU Header] [NALU Payload]三部分组成, 其中 Start Code 用于标示这是一个
NALU 单元的开始, 必须是 “00 00 00 01” 或 “00 00 01”, NALU 头仅一个字节, 其后都是 NALU 单元内容.

打包时去除 “00 00 01” 或 “00 00 00 01” 的开始码, 把其他数据封包的 RTP包即可

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|F|NRI|  type   |                                               |
+-+-+-+-+-+-+-+-+                                               |
|                                                               |
|              Bytes 2..n of a Single NAL unit                  |
|                                                               |
|                               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               :...OPTIONAL RTP padding        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

```

如有一个 H.264 的 NALU 是这样的：

[00 00 00 01 67 42 A0 1E 23 56 0E 2F …]

这是一个序列参数集 NAL 单元. [00 00 00 01] 是四个字节的开始码, 67 是 NALU 头, 42 开始的数据是 NALU 内容.

封装成 RTP 包将如下:

[RTP Header] [67 42 A0 1E 23 56 0E 2F]

即只要去掉 4 个字节的开始码就可以了

2．组合封包模式

当 NALU 的长度特别小时, 可以把几个 NALU 单元封在一个 RTP 包中

3、FU-A的分片格式

数据比较大的H264视频包，被RTP分片发送。12字节的RTP头后面跟随的就是FU-A分片：
而当 NALU 的长度超过 MTU 时, 就必须对 NALU 单元进行分片封包. 也称为 Fragmentation Units (FUs)

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| FU indicator  |   FU header   |                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
|                                                               |
|                       FU payload                              |
|                                                               |
|                               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               :...OPTIONAL RTP padding        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Figure 14.  RTP payload format for FU-A
```

1） FU indicator有以下格式：

```
+---------------+
|0|1|2|3|4|5|6|7|
+-+-+-+-+-+-+-+-+
|F|NRI|  Type   |
+---------------+

```

FU指示字节的类型域的28，29表示FU-A和FU-B。NRI域的值必须根据分片NAL单元的NRI域的值设置。（此处Type就是rtp分片类型） 见下表

```
.Type   Packet      Type name
---------------------------------------------------------
0      undefined                                     -
1-23   NAL unit    Single NAL unit packet per H.264
24     STAP-A     Single-time aggregation packet
25     STAP-B     Single-time aggregation packet
26     MTAP16    Multi-time aggregation packet
27     MTAP24    Multi-time aggregation packet
28     FU-A      Fragmentation unit
29     FU-B      Fragmentation unit
30-31  undefined
```

2） FU header的格式如下：

```
+---------------+
|0|1|2|3|4|5|6|7|
+-+-+-+-+-+-+-+-+
|S|E|R|  Type   |
+---------------+
```

S: 1 bit 当设置成1,开始位指示分片NAL单元的开始。当跟随的FU荷载不是分片NAL单元荷载的开始，开始位设为0。

E: 1 bit 当设置成1, 结束位指示分片NAL单元的结束，即, 荷载的最后字节也是分片NAL单元的最后一个字节。
当跟随的FU荷载不是分片NAL单元的最后分片,结束位设置为0。

R: 1 bit
保留位必须设置为0，接收者必须忽略该位。

Type: 5 bits
此处的Type就是NALU头中的Type,取1-23的那个值，表示 NAL单元荷载类型定义，见图4

4、拆包和解包

拆包：当编码器在编码时需要将原有一个NAL按照FU-A进行分片，原有的NAL的单元头与分片后的FU-A的单元头有如下关系：

原始的NAL头的前三位为FU indicator的前三位，原始的NAL头的后五位为FU header的后五位，

FU indicator与FU header的剩余位数根据实际情况决定。

解包：当接收端收到FU-A的分片数据，需要将所有的分片包组合还原成原始的NAl包时，FU-A的单元头与还原后的NAL的关系如下：

还原后的NAL头的八位是由FU indicator的前三位加FU header的后五位组成，即：

nal_unit_type = (fu_indicator & 0xe0) | (fu_header & 0x1f)

参考：

https://blog.csdn.net/machh/article/details/52165292

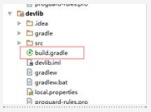
https://www.jianshu.com/p/8edb448cf22e

.NET STANDARD 来日苦短去日长

作者：Richard 翻译：精致码农-王亮 原文：http://dwz.win/Q4h 自从 .NET 5 宣贯以来，很多人都在问这对 .NET Standard 意味着什么，它是否仍然重要。在这篇文章中，我将解释 .NET 5 是如何改进代码共用并取代 .NET Standard 的，我还将介绍什么情况下你仍然需要 .NET Standard。概要 .NET 5 将是一个具有统一功能和 API...

JAVA集合：ARRAYLIST（JDK1.8 源码解读）

ArrayList 概述 重要方法分析 add 构造 ge...



GRADLE学习笔记，常用命令，多渠道打包等

本文整理自：http://stormzhang.com/devtools/2014/12/18/android-studio-tutorial4/
http://stormzhang.com/devtools/2015/01/05/android-studio-tutorial5/
http://stormzhang.com/devtools/2015/01/15/android-studio-...