# 小鱼小虾

|  |  |
| --- | --- |
| 日期: | 2021.04.12 |
| By | 叶荣跃 |
| 版本 | V1.0.0 |

抄于：

https://blog.csdn.net/leixiaohua1020/article/details/18893769

# **一．视频播放器原理**

## 视音频技术主要包含

视音频技术主要包含以下几点：封装技术，视频压缩编码技术以及音频压缩编码技术。如果考虑到网络传输的话，还包括流媒体协议技术。

视频播放器的源代码详细解析（Media Player Classic - HC，Mplayer，FFplay，XBMC）可以参考系列文章：

Media Player Classic：Media Player Classic - HC源代码分析 1：整体结构[系列文章]

http://blog.csdn.net/leixiaohua1020/article/details/13280659

Mplayer：MPlayer源代码分析

http://blog.csdn.net/leixiaohua1020/article/details/11885509

FFplay： FFplay源代码分析：整体流程图

http://blog.csdn.net/leixiaohua1020/article/details/11980843

XBMC： XBMC源代码分析 1：整体结构以及编译方法[系列文章]

<http://blog.csdn.net/leixiaohua1020/article/details/17454977>

视频播放器播放一个互联网上的视频文件，需要经过以下几个步骤：解协议，解封装，解码视音频，视音频同步。如果播放本地文件则不需要解协议，为以下几个步骤：解封装，解码视音频，视音频同步。他们的过程如图1-1所示。

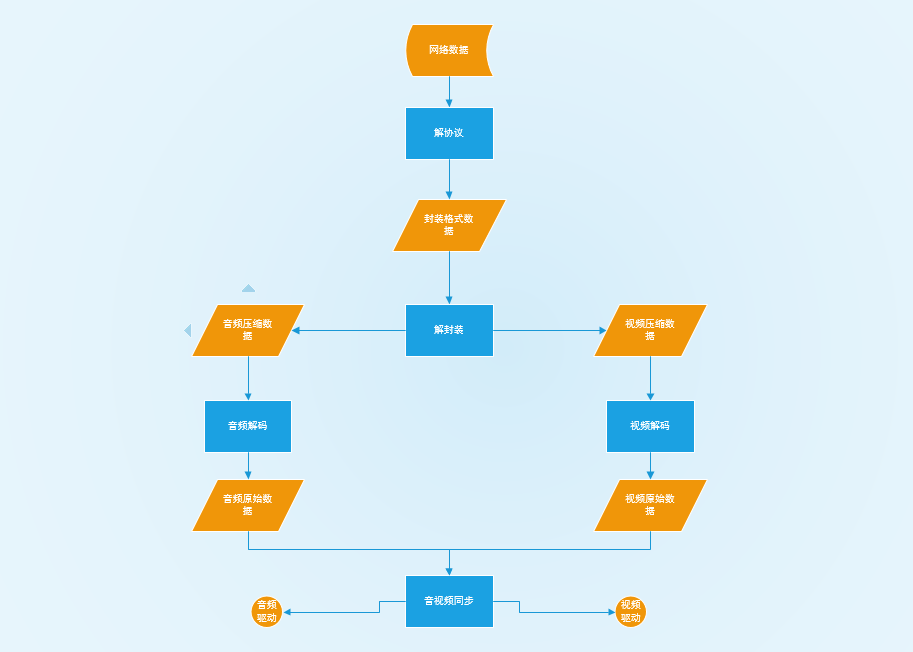


图1-1

#### 解协议

解协议就是将采用HTTP，RTMP，或是MMS等流媒体协议的数据解析成标准的封装格式数据，这个过程去掉了信令数据，只保留了音视频数据。

#### 解封装

将MP4、MKV、RMVB、TS、FLV、AVI等封装格式数据分离成视频压缩数据和音频压缩数据。FLV格式的数据，经过解封装操作后，输出H.264编码的视频码流和AAC编码的音频码流。

#### 解码

将音视频压缩数据解析成原始数据，视频压缩数据—>YUV420P、RGB数据，音频-->PCM抽样数据

#### 音视频同步

根据解封装的数据信息去同步解码得到的音视频数据，并播放

#### 相关链接

流媒体协议，封装格式，以及视音频编码标准。更详细的比较可以参考：

[视频参数（流媒体系统，封装格式，视频编码，音频编码，播放器）对比](http://blog.csdn.net/leixiaohua1020/article/details/11842919)

<https://www.cnblogs.com/yuweifeng/p/8468744.html>

<https://blog.csdn.net/wudebao5220150/article/details/13016871>

本文中涉及到的协议数据、封装格式数据、视频编码数据、音频编码数据、视频像素数据、音频采样数据的分析可以参考下面系列文章：

[视音频数据处理入门：RGB、YUV像素数据处理](http://blog.csdn.net/leixiaohua1020/article/details/50534150)

[视音频数据处理入门：PCM音频采样数据处理](http://blog.csdn.net/leixiaohua1020/article/details/50534316)

[视音频数据处理入门：H.264视频码流解析](http://blog.csdn.net/leixiaohua1020/article/details/50534369)

[视音频数据处理入门：AAC音频码流解析](http://blog.csdn.net/leixiaohua1020/article/details/50535042)

[视音频数据处理入门：FLV封装格式解析](http://blog.csdn.net/leixiaohua1020/article/details/50535082)

[视音频数据处理入门：UDP-RTP协议解析](http://blog.csdn.net/leixiaohua1020/article/details/50535230)

# **二．流媒体协议**

## 流媒体概述

流媒体指的是采用流式传输的方式在Internet播放的媒体格式。流式传输方式是将视频和音频等多媒体文件经过特殊的压缩方式分成一个个压缩包，由服务器向用户计算机连续、实时传送。

[各种流媒体详细介绍](https://blog.csdn.net/xiaomucgwlmx/article/details/102851352)

## RTP：实时传输协议（Real-time Transport Protocol）

RTP是一种基于包的传输协议，用于实时传输数据。用于Internet上针对多媒体数据流的一种传输层协议。在网络上传输数据包延迟是不可避免的，针对此RTP的包头包含了时间戳、丢失保护、载荷标识、源标识以及安全信息。通过这些在应用层实现数据包的丢失恢复、拥塞控制等。运行于UDP上层，利用其复用求和校验功能，采用UDP/IP包封装：

|  |  |  |  |
| --- | --- | --- | --- |
| IP头 | UDP头 | RTP头 | RTP载荷 |

RTP 本身并没有提供按时发送机制或其它服务质量（QoS）保证，它依赖于低层服务去实现这一过程。 RTP 并不保证传送或防止无序传送，也不确定底层网络的可靠性。 RTP 实行有序传送， RTP 中的序列号允许接收方重组发送方的包序列，同时序列号也能用于决定适当的包位置，例如：在视频解码中，就不需要顺序解码。

         RTP 由两个紧密链接部分组成： RTP ― 传送具有实时属性的数据；RTCP 控制协议 ― 监控服务质量并传送正在进行的会话参与者的相关信息。

## RTCP：实时传输控制协议（Real-time Transport Control Protocol）

是实时传输协议（RTP）的一个姐妹协议。RTCP为RTP媒体流提供信道外（out-of-band）控制。RTCP本身并不传输数据，但和RTP一起协作将多媒体数据打包和发送。RTCP定期在流多媒体会话参加者之间传输控制数据。RTCP的主要功能是为RTP所提供的服务质量（Quality of Service）提供反馈。

        RTCP收集相关媒体连接的统计信息，例如：传输字节数，传输分组数，丢失分组数，jitter，单向和双向网络延迟等等。网络应用程序可以利用RTCP所提供的信息试图提高服务质量，比如限制信息流量或改用压缩比较小的编解码器。RTCP本身不提供数据加密或身份认证。SRTCP可以用于此类用途。

## SRTP & SRTCP：安全实时传输协议（Secure Real-time Transport Protocol）

SRTP实际上就是在RTP实时传输协议的基础定义的一个协议，旨在单播或多播的应用程序中的实时数据传输提供数据加密、消息认证、完整性保证和重放保护。最早由David Oran（思科）和Rolf Blom（爱立信）开发，并最早由IETF于2004年3月作为RFC3711发布。

## RTSP：控制声音或影像串流协议（Real Time Streaming Protocol）

 是由Real Networks和Netscape共同提出的。该协议定义了一对多应用程序如何有效地通过IP网络传送多媒体数据。RTSP提供了一个可扩展框架，使实时数据，如音频与视频的受控、点播成为可能。数据源包括现场数据与存储在剪辑中的数据。该协议目的在于控制多个数据发送连接，为选择发送通道，如UDP、多播UDP与TCP提供途径，并为选择基于RTP上发送机制提供方法。

RTSP是一个串流协议，满足多个串流的需求，可降低服务端的网络用量并且支持多方视讯会议。未定传输使用的网络通讯所以服务端可

## RTSP 和RTP的关系

RTP是实时传输协议，不能像http和ftp一样可以下载完整的影响文件，只能按照固定的速率在网络上发送数据，客户端也是按照这个速率观看，不可重复播放。

RTSP与RTP最大的区别：RTSP是一种双向数据传输协议，允许客户端向服务端发送快退、快进、回放灯操作，类似于http协议的网络层协议，可基于RTP来传输数据采用TCP、UDP、UDP组播等通道发送数据。

## RTSP和HTTP的比较

相同点：提供的服务相同

不同点：

1、RTSP使用音视频流形式，HTTP使用文本和图片形式

2、RTSP服务器需要维持会话状态，将请求和流关联起来

3、HTTP不是对称的（客户端发出请求，服务器响应)，但在RTSP协议中客户端和服务器都可以发出请求

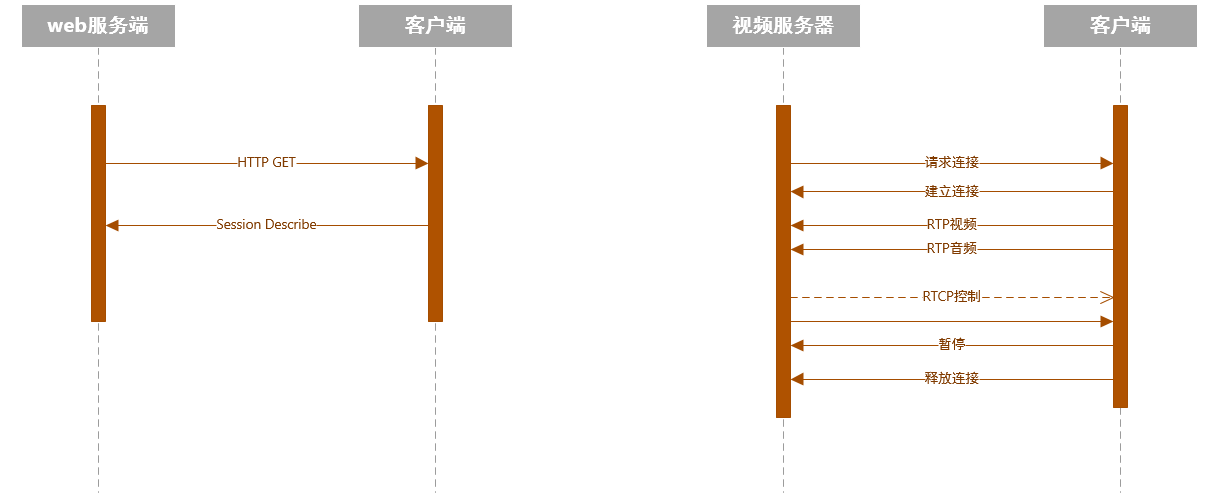


图2-1

## **SDP：**会话描述协议（SDP:Session Description Protocol）

## **RTMP/RTMPS：**RTMP实时消息传送协议(Real Time Messaging Protocol)

## MMS： 微软媒体服务器协议(Microsoft Media Server Protocol)

## **HLS：**HTTP的流媒体传输协议（ Live Streaming）

## http-flv、rtmp和hls直播的优缺点:



参考资料

[从零开始写一个RTSP服务器（一）RTSP协议讲解](https://blog.csdn.net/weixin_42462202/article/details/98986535?utm_medium=distribute.pc_relevant.none-task-blog-2%7Edefault%7EBlogCommendFromMachineLearnPai2%7Edefault-13.control&dist_request_id=&depth_1-utm_source=distribute.pc_relevant.none-task-blog-2%7Edefault%7EBlogCommendFromMachineLearnPai2%7Edefault-13.control)

# **三．封装格式**

封装格式又称作多媒体容器，主要的作用是将视频码流、音频码流、字幕流等封装到一个文件中。如图3-1所示。

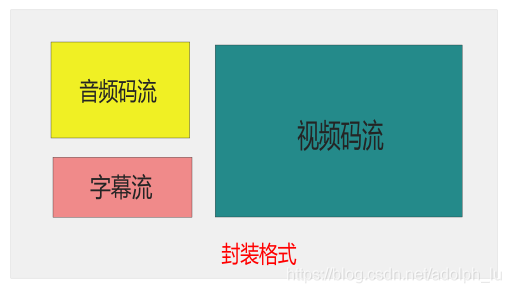


图3-1

## 常见的封装格式

如图3-2所示

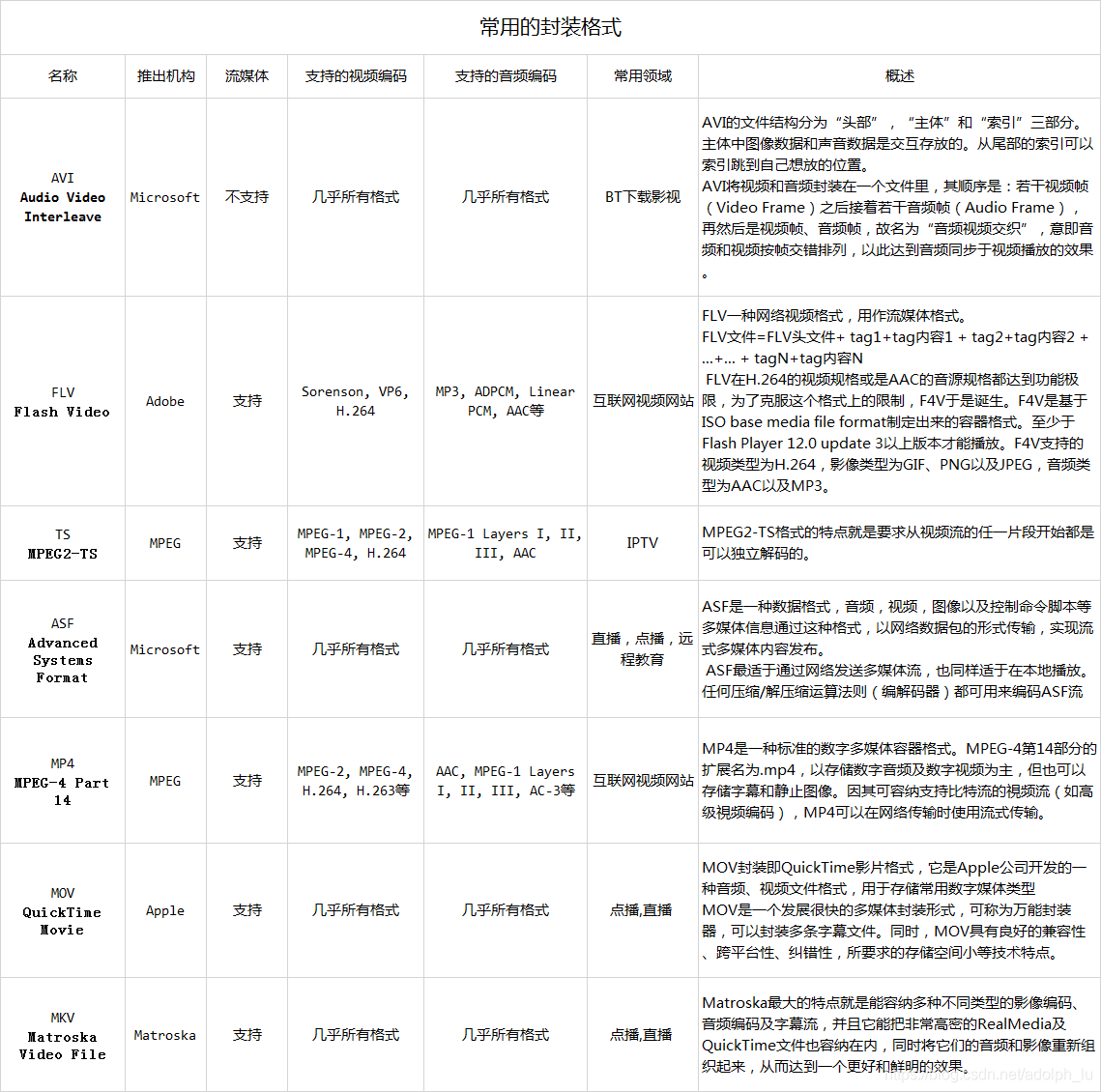


图3-2

由表可见，除了AVI之外，其他封装格式都支持流媒体，即可以“边下边播”。有些格式更“万能”一些，支持的视音频编码标准多一些，比如MKV。而有些格式则支持的相对比较少，比如说RMVB。

[TS封装格式分析器](http://blog.csdn.net/leixiaohua1020/article/details/17973587)

[FLV封装格式分析器](http://blog.csdn.net/leixiaohua1020/article/details/17934487)

# **四．视频编码**

视频编码的作用就是将RGB、YUV等视频像素数据格式压缩成视频码流，从而降低数据量。视频码流占据了音视频技术中的绝大部分数据量，故而，视频编码是音视频技术中最重要的技术之一，同等的码率高效的视频编码便可以获得更高的视频质量。

视频编码的简单原理可以参考：[视频压缩编码和音频压缩编码的基本原理](http://blog.csdn.net/leixiaohua1020/article/details/28114081" \t "_blank)

## 视频编码基本原理

### 1视频信息的冗余信息

数字视频YUV分量为例，其中Y代表亮度信号，代表UV两个色差信号，亮度信号的采样频率为13.5MHz，色度信号的频率为其一半或者更少。若采样频率为4:2:2，8bit量化，那么

|  |
| --- |
| （13.5+6.25+6.25MHz）\*8bit/s=216Mbit/s |

由上述可以看出，若进行视频数据的传输或者储存数据量太大，所以需要压缩技术将码率降低。而视频信号的压缩主要依据两个基本的条件：

#### 数据冗余

图像的像素之前存在很强的关联性，于是就有空间冗余、时间冗余、结构冗余、信息熵冗余等，当消除这些冗余时并不会导致信息损失，属于无损压缩

#### 视觉冗余

以人眼特性可以识别每秒24帧的视频，对于亮度辨别阈值、视觉阈值也是，对于亮度、色度也有一定的敏感度，这样在编码的时候引入误差也不会被察觉出来。所以可以根据人眼的特性，以一定的客观失真换取压缩数据，属于有损压缩。

### 2压缩编码的方法

根据数据冗余和视觉冗余两个基本条件，可以很好的对视频数据进行大量的压缩，有利于传输以及储存。一般的数字视频压缩编码都是混合编码，就是将变换编码、运动估计和运动补偿、熵编码三种方式相结合进行编码压缩的。

#### 变换编码

是指消除图像的帧内冗余，作用是将空间域描述的图像信号转换成频率域，然后对转换后的数据进行编码处理，常用的正交变换有离散傅里叶变换，[离散余弦变换](https://baike.baidu.com/item/%E7%A6%BB%E6%95%A3%E4%BD%99%E5%BC%A6%E5%8F%98%E6%8D%A2/7118270?fr=aladdin)等等。数字视频压缩过程中应用广泛的是离散余弦变换。

离散傅里叶变换需要进行复数运算，尽管有FFT可以提高运算速度，但在图像编码、特别是在实时处理中非常不便。离散傅里叶变换在实际的图像通信系统中很少使用，但它具有理论的指导意义。根据离散傅里叶变换的性质，实偶函数的傅里叶变换只含实的余弦项，因此构造了一种实数域的变换——离散余弦变换(DCT)。

在对语音、图像信号变换的确定的变换矩阵正交变换中，DCT变换被认为是一种准最佳变换。在近年颁布的一系列[视频压缩编码](https://baike.baidu.com/item/%E8%A7%86%E9%A2%91%E5%8E%8B%E7%BC%A9%E7%BC%96%E7%A0%81/854664)的国际标准建议中，都把 DCT 作为其中的一个基本处理模块。

DCT变换可以将L\*L的图像块从空间域变换为频率域。所以，在基于DCT的图像压缩编码过程中，首先需要将图像分成互不重叠的图像块。假设一帧图像的大小为1280\*720，首先将其以网格状的形式分成160\*90个尺寸为8\*8的彼此没有重叠的图像块，接下来才能对每个图像块进行DCT变换。

经过分块以后，每个8\*8点的图像块被送入DCT编码器，将8\*8的图像块从空间域变换为频率域。下图给出一个实际8\*8的图像块例子，图中的数字代表了每个像素的亮度值。从图4-1上可以看出，在这个图像块中各个像素亮度值比较均匀，特别是相邻像素亮度值变化不是很大，说明图像信号具有很强的相关性。

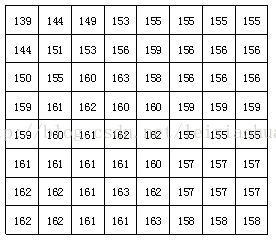


图4-1 一个实际8\*8图像块

图4-2是图4-1图像块经过DCT变换后的结果。从图4-2可以看出经过DCT变换后，左上角的低频系数集中了大量能量，而右下角的高频系数上的能量很小。

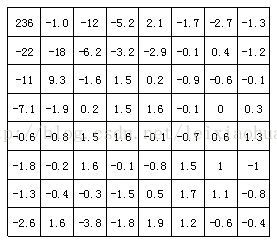


图4-2 图像块经过DCT变换后的系数

信号经过DCT变换后需要进行量化。由于人的眼睛对图像的低频特性比如物体的总体亮度之类的信息很敏感，而对图像中的高频细节信息不敏感，因此在传送过程中可以少传或不传送高频信息，只传送低频部分。量化过程通过对低频区的系数进行细量化，高频区的系数进行粗量化，去除了人眼不敏感的高频信息，从而降低信息传送量。因此，量化是一个有损压缩的过程，而且是视频压缩编码中质量损伤的主要原因。

https://img-blog.csdn.net/20140602173713140?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvbGVpeGlhb2h1YTEwMjA=/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/Center

其中FQ（u,v）表示经过量化后的DCT系数；F（u,v）表示量化前的DCT系数；Q（u,v）表示量化加权矩阵；q表示量化步长；round表示归整，即将输出的值取为与之最接近的整数值。

合理选择量化系数，对变换后的图像块进行量化后的结果如图4-3所示。

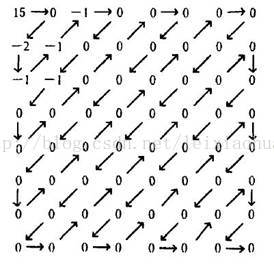


图4-3 量化后的DCT系数

DCT系数经过量化之后大部分经变为0，而只有很少一部分系数为非零值，此时只需将这些非0值进行压缩编码即可。

#### 熵编码

熵编码是因编码后的平均码长接近信源熵值而得名。熵编码多用可变字长编码（VLC，Variable Length Coding）实现。其基本原理是对信源中出现概率大的符号赋予短码，对于出现概率小的符号赋予长码，从而在统计上获得较短的平均码长。可变字长编码通常有霍夫曼编码、算术编码、游程编码等。其中游程编码是一种十分简单的压缩方法，它的压缩效率不高，但编码、解码速度快，仍被得到广泛的应用，特别在变换编码之后使用游程编码，有很好的效果。

首先要在量化器输出直流系数后对紧跟其后的交流系数进行Z型扫描（如图箭头线所示）。Z型扫描将二维的量化系数转换为一维的序列，并在此基础上进行游程编码。最后再对游程编码后的数据进行另一种变长编码，例如霍夫曼编码。通过这种变长编码，进一步提高编码的效率。

#### 运动估计和运动补偿

是运动估计（Motion Estimation）和运动补偿（Motion Compensation）是消除图像序列时间方向相关性的有效手段。上文介绍的DCT变换、量化、熵编码的方法是在一帧图像的基础上进行，通过这些方法可以消除图像内部各像素间在空间上的相关性。实际上图像信号除了空间上的相关性之外，还有时间上的相关性。例如对于像新闻联播这种背景静止，画面主体运动较小的数字视频，每一幅画面之间的区别很小，画面之间的相关性很大。对于这种情况我们没有必要对每一帧图像单独进行编码，而是可以只对相邻视频帧中变化的部分进行编码，从而进一步减小数据量，这方面的工作是由运动估计和运动补偿来实现的。

运动估计技术一般将当前的输入图像分割成若干彼此不相重叠的小图像子块，例如一帧图像的大小为1280\*720，首先将其以网格状的形式分成40\*45个尺寸为16\*16的彼此没有重叠的图像块，然后在前一图像或者后一个图像某个搜索窗口的范围内为每一个图像块寻找一个与之最为相似的图像块。这个搜寻的过程叫做运动估计。通过计算最相似的图像块与该图像块之间的位置信息，可以得到一个运动矢量。这样在编码过程中就可以将当前图像中的块与参考图像运动矢量所指向的最相似的图像块相减，得到一个残差图像块，由于残差图像块中的每个像素值很小，所以在压缩编码中可以获得更高的压缩比。这个相减过程叫运动补偿。

由于编码过程中需要使用参考图像来进行运动估计和运动补偿，因此参考图像的选择显得很重要。一般情况下编码器的将输入的每一帧图像根据其参考图像的不同分成3种不同的类型：I（Intra）帧、B（Bidirection

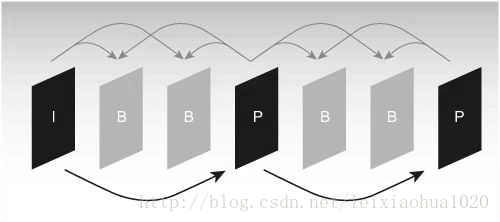


图4-4 典型的I，B，P帧结构顺序

如图4-4所示，I帧只使用本帧内的数据进行编码，在编码过程中它不需要进行运动估计和运动补偿。显然，由于I帧没有消除时间方向的相关性，所以压缩比相对不高。P帧在编码过程中使用一个前面的I帧或P帧作为参考图像进行运动补偿，实际上是对当前图像与参考图像的差值进行编码。B帧的编码方式与P帧相似，惟一不同的地方是在编码过程中它要使用一个前面的I帧或P帧和一个后面的I帧或P帧进行预测。由此可见，每一个P帧的编码需要利用一帧图像作为参考图像，而B帧则需要两帧图像作为参考。相比之下，B帧比P帧拥有更高的压缩比。

#### 混合编码