(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 103686449 B
(45)授权公告日 2018.01.30

(21)申请号 201310751327.8

(22)申请日 2013.12.31

(65)同一申请的已公布的文献号
申请公布号 CN 103686449 A

(43)申请公布日 2014.03.26

(73)专利权人 快创科技(大连)有限公司
地址 116023 辽宁省大连市高新技术产业园区火炬路32号B座11层1101号

(72)发明人 童培诚

(74)专利代理机构 大连东方专利代理有限责任公司 21212
代理人 李馨 李洪福

(51)Int.Cl.
H04N 21/647(2011.01)
H04N 21/231(2011.01)
H04N 21/433(2011.01)

(56)对比文件
CN 103460696 A,2013.12.18,
JP 特开2004-241879 A,2004.08.26,
US 2013/0315301 A1,2013.11.28,
CN 103414917 A,2013.11.27,

审查员 盛磊

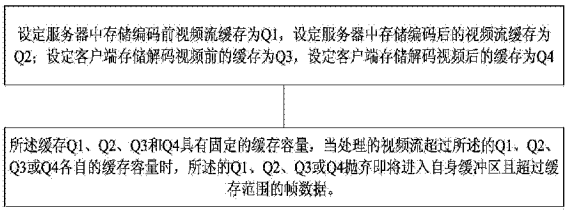
权利要求书1页 说明书5页 附图1页

(54)发明名称

一种提升视频流畅度以及画质的缓存方法

(57)摘要

本发明公开了一种提升视频流畅度以及画质的缓存方法，解决在服务器和客户端之间传送的视频流因网络延迟引起的画质和流畅度的降低，具有如下步骤：设定存储编码前视频流缓存Q1，服务器存储编码后的缓存为Q2；客户端解码视频前缓存为Q3，客户端存储解码视频后为Q4，当处理的视频流超过所述的Q1、Q2、Q3或Q4各自的缓存容量时，所述的Q1、Q2、Q3或Q4抛弃即将进入自身缓冲区且超过缓存范围的帧数据。使用多个分散的视频帧数据缓冲区，少量多次的抛弃堵塞的视频帧，在网络拥堵环境下仍能保证一定的视频流畅度，即使经过分散的抛帧，将每秒视频帧的数量降低至24帧以下，通常也只能造成视频出现短时间卡顿，而不会造成大量视频数据的丢失，有效的提升了视频的流畅度和画质，增强了用户体验。



CN 103686449 B

1.一种提升视频流畅度以及画质的缓存方法，解决在服务器和客户端之间传送的视频流因网络延迟引起的画质和流畅度的降低，具有如下步骤：

—设定服务器中存储编码前视频流缓存为Q1，设定服务器中存储编码后的视频流缓存为Q2；设定客户端存储解码视频前的缓存为Q3，设定客户端存储解码视频后的缓存为Q4；

—所述缓存Q1、Q2、Q3和Q4具有固定的缓存容量，当处理的数据超过所述的Q1、Q2、Q3或Q4各自的缓存容量时，所述的Q1、Q2、Q3或Q4抛弃超过缓存范围且即将进入自身缓冲区的数据；

—所述Q1和Q2共享位于服务器端的一固定容量的缓冲区；所述Q3和Q4共享位于客户端一固定容量的缓冲区；

在服务器中，当等待进入Q1进行编码的视频帧数据超过Q1原有的缓存分配容量，则Q1增加自身缓冲区的容量，减少Q2缓冲区的容量；当等待进入Q2编码后的视频流数据超过Q2原有的缓存分配容量，则Q1减少自身缓冲区的容量，增加Q2缓冲区的容量；

在客户端中，当等待进入Q3进行解码的视频流数据超过Q3原有的缓存分配容量，则Q3增加自身缓冲区的容量，减少Q4缓冲区的容量；当等待进入Q4编码后的视频帧数据超过Q4原有的缓存分配容量，则Q3减少自身缓冲区的容量，增加Q4缓冲区的容量；

—所述的缓存Q1、Q2、Q3和Q4交互动态确定各自缓冲区的容量：

当所述的四个缓冲区中存在一缓冲区出现数据堆积，需要增加自身的缓冲容量：按Q1，Q4，Q3，Q2的顺序遍历其它缓冲区，若某缓冲区有剩余空间，则该缓冲区减少自身的容量，将容量赋予所述出现数据堆积的缓冲区。

2.根据权利要求1所述的一种提升视频流畅度以及画质的缓存方法，其特征还在于：所述的缓存Q1、Q2、Q3和Q4具有统一的缓存容量大小时，所述Q1、Q2、Q3和Q4各自容量的计算公式为：

$Q=((d-u)/(1000/f))/4$；

其中：f为客户端接收到的视频帧数据的帧率、d为客户设置的可忍受的延迟；u一定时间内的网络平均延迟。

3.根据权利要求1所述的一种提升视频流畅度以及画质的缓存方法，其特征还在于：所述Q1和Q2共享位于服务器端的一固定容量的缓冲区时，所述缓存Q1、Q2、Q3和Q4容量的计算方法如下：

$Q1=((ds-u/2)/(1000/f))/2$；$Q2=(ds-u/2)/(1000/f)-Q1$；

$Q4=((cs-u/2)/(1000/f))/2$；$Q3=(cs-u/2)/(1000/f)-Q4$；

其中ds为服务器端可容忍延迟，cs为客户端可容忍延迟；u一定时间内的平均延迟；f为客户端帧率。

4.根据权利要求1所述的一种提升视频流畅度以及画质的缓存方法，其特征还在于：所述的缓存Q1、Q2、Q3和Q4交互动态确定各自缓冲区的容量的情况下，当所述的四个缓冲区中的某一缓冲区出现空闲时，可以将自身的剩余容量分配给所述其它缓冲区：按Q2，Q3，Q4，Q1的顺序遍历其它缓冲区，若某缓冲区容量小于等待进入的数据量，则将所述的剩余流量传送给该缓冲区。

一种提升视频流畅度以及画质的缓存方法

技术领域

[0001] 本发明涉及一种提升视频流长度以及画质的缓存方法，尤其涉及提升由服务器通过网络向多个客户端传输视频帧数据时，提升视频流畅度的方法。

背景技术

[0002] 网络通常是不稳定且不可控制的，服务器在发送连续的视频帧数据的时候有可能因为网络堵塞、网络故障等原因造成很长时间不能向网络发送数据包，此时服务器端会积累大量视频数据；同样的客户端在网络情况不稳定或者播放器性能不够理想的情况下同样会造成大量的视频数据累积。现行的解决方案是抛帧，即在发生网络故障，在服务器端和客户端造成大量的帧数据堵塞的情况下，按照设定的时间阈值或其它条件，将积累的部分或全部视频帧数据直接抛弃，虽然能够解决大量的视频帧数据堆积的问题，但是会造成大量帧数据的丢失，即造成视频内容的中断，影响使用者的视听感受。

发明内容

[0003] 本发明针对以上问题的提出，而研制的一种提升视频流畅度以及画质的缓存方法，解决在服务器和客户端之间传送的视频流因网络延迟引起的画质和流畅度的降低，具有如下步骤：

[0004] —设定服务器中存储编码前视频流缓存为Q1，设定服务器中存储编码后的视频流缓存为Q2；设定客户端存储解码视频前的缓存为Q3，设定客户端存储解码视频后的缓存为Q4；

[0005] —所述缓存Q1、Q2、Q3和Q4具有固定的缓存容量，当处理的视频流超过所述的Q1、Q2、Q3或Q4各自的缓存容量时，所述的Q1、Q2、Q3或Q4抛弃即将进入自身缓冲区且超过缓存范围的帧数据。

[0006] 通过在服务器端和客户端分别设置编码前后和解码前后的四个独立缓冲区，每个独立缓冲区出现数据堵塞时，抛弃掉即将进入的数据，有效的避免了出现大量视频帧数据的拥堵，即出现长时间视频卡顿，缺失长时间段的视频影像。只要在1秒内视频帧的数量达到24帧，人眼就默认为是连贯的视频图像，相对于现有技术中大量抛弃连贯的视频帧，使用多个分散的视频帧数据缓冲区，少量多次的抛弃堵塞的视频帧，在网络拥堵环境下仍能保证一定的视频流畅度，即使经过分散的抛帧，将每秒视频帧的数量降低至24帧以下，通常也只能造成视频出现短时间卡顿，而不会造成大量视频数据的丢失，有效的提升了视频的流畅度和画质，增强了用户体验。

[0007] 为了提升用户体验，作为一个较佳的实施方式，Q1、Q2、Q3和Q4具有统一的容量。所述Q1、Q2、Q3和Q4的计算公式为Q=(d-u)/(1000/f)/4，其中：f为客户端接收到的视频流的帧率、d为客户设置的可忍受的延迟；u一定时间内的网络平均延迟。

[0008] 计算得出的Q为关于视频帧数和延迟的一个参考的比例关系，Q1、Q2、Q3和Q4的设置可以参照Q值进行设置。

[0009] 更进一步的，采用的统一固定大小的缓冲区会造成缓冲区利用不充分，可能造成某个或某几个缓冲区具有很大的空闲，而其他的缓冲区又具有大量的数据堵塞，堵塞的缓冲区大量抛弃的视频进而造成视频质量的下降。

[0010] 同时考虑到Q1和Q2位于一台实体计算机中，即所述的视频服务器中；同时Q3和Q4位于一台实体计算机中，即所述的客户端；Q1和Q2，Q3和Q4之间的通信非常迅速。

[0011] 故优选的，所述Q1和Q2共享位于服务器端的一固定容量的缓冲区；所述Q3和Q4共享位于客户端一固定容量的缓冲区。

[0012] 在服务器中，当等待进入Q1进行编码的视频帧数据超过Q1原有的缓存分配容量，则Q1增加自身缓冲区的容量，减少Q2缓冲区的容量；同样的，当等待进入Q2的帧数据超过Q2自身的容量时，增加Q2的容量，相应的减少Q1的容量。

[0013] 在客户端中，当等待进入Q3进行解码的视频帧数据超过Q3原有的缓存分配容量，则Q3增加自身缓冲区的容量，减少Q4缓冲区的容量。同样的当Q4缓冲区饱和的情况下，也减少Q3的缓冲区容量。

[0014] 由于采用了动态调配容量的，分组设置的缓冲区，避免了在服务器端和客户端因某个缓冲区出现饱和，而造成大量抛帧的情况，更进一步的增加了用户的观感体验。

[0015] 作为一个较佳的实施方式，所述缓存Q1、Q2、Q3和Q4容量的计算方法如下：

[0016] Q1=(ds-u/2)/(1000/f)/2；Q2=(ds-u/2)/(1000/f)-Q1；

[0017] Q4=(cs-u/2)/(1000/f)/2；Q3=(cs-u/2)/(1000/f)-Q4；

[0018] 其中ds为服务器端可容忍延迟，cs为客户端可容忍延迟；u一定时间内的平均延迟；f为客户端帧率。

[0019] 由于Q4为解码(解压)后的视频帧数据，需要的缓存空间大于客户端缓存由服务器端传输的压缩的数据包的空间，故先计算Q4。

[0020] 更进一步的，考虑到如果只在服务器和客户端中设定共享一定容量空间的缓冲区，即服务器端的缓冲区大小是固定的，客户端的缓冲区大小也是固定的，不能做到在客户端和服务器之间的完全动态调整。

[0021] Q1，Q2是在同一台机器上，Q3，Q4是在同一台机器上，互相之间数据传递速度是非常快的，而Q2，Q3是要通过网络才能交流的，速度是非常慢的。

[0022] 作为一个较佳的实现在服务器和客户端之间动态调整缓冲区的方式：当所述的四个缓冲区中存在一缓冲区出现数据堆积，需要增加自身的缓冲容量：按Q1，Q4，Q3，Q2的顺序遍历其它缓冲区，若某缓冲区有剩余空间，则该缓冲区减少自身的容量，将容量赋予所述出现数据堆积的缓冲区。

[0023] 采用Q1，Q4，Q3，Q2顺序的原因是：Q1具有最大数量的数据源，而且数据源为未经压缩的原始视频帧数据，即使抛掉一些，只要能保证在每秒24帧或少于但接近24帧，也不会对整体的视频流畅度和画质造成太大的影响。

[0024] Q4的数据是处于解码(可理解为解压)后的视频帧数据，在所述的四个缓冲区中具有仅次于Q1的数据源，即使抛掉一些帧数据，也很快会有很多视频帧数据补充。

[0025] Q3和Q2为解码后(可理解为压缩后)的视频帧数据，数据源的数量要远小于所述Q1和Q4，故排在Q1和Q4之后。Q2和Q3，由于Q2的缓存数据量要大于Q3(位于服务器端，未经过网络传输，不存在丢包或相应的数据包缺失)，故将Q2的减少顺序排在最后。

[0026] 由于采用了服务器和客户端之间的动态调整，避免了在服务器端或客户端由于网络传输问题造成的单方数据堵塞，避免了大量抛帧情况的发生，提升了用户体验。

[0027] 更进一步的，考虑到只在堵塞发生后采取动调整措施，具有滞后性，一旦网络出现堵塞，肯定会对用户体验造成明显的影响。

[0028] 故本发明还具有预分配机制，作为一个较佳的实施方式，当所述的四个缓冲区中的某一缓冲区出现空闲时，可以将自身的剩余容量分配给所述其它缓冲区：按Q2，Q3，Q4，Q1的顺序遍历其它缓冲区，若某缓冲区容量小于等待进入的数据量，则将所述的剩余流量传送给该缓冲区。排列的原因与递减的序列类似。由于Q4和Q1具有庞大的数据源，可以相对于Q2和Q3抛弃更多的视频帧。而Q2和Q3由于存储的是压缩数据，抛帧的容忍度远远小于Q1和Q4。

[0029] 相对于被动等待堵塞情况的发生再进行补救的方式，采用主动分配缓冲区剩余容量，可以做到防患于未然，在堵塞尚未发生时，提前提升帧数据量较大的缓冲区的容量，防止视频帧数据在某一缓冲区、服务器端或客户端造成大量堵塞，减少了抛帧数量，增加了用户体验的流畅性。

附图说明

[0030] 为了更清楚的说明本发明的实施例或现有技术的技术方案，下面将对实施例或现有技术描述中所需要使用的附图做一简单地介绍，显而易见地，下面描述中的附图仅仅是本发明的一些实施例，对于本领域普通技术人员来讲，在不付出创造性劳动的前提下，还可以根据这些附图获得其他的附图。

[0031] 图1为本发明帧数据流向示意图

[0032] 图2为本发明的流程图

具体实施方式

[0033] 为使本发明的实施例的目的、技术方案和优点更加清楚，下面结合本发明实施例中的附图，对本发明实施例中的技术方案进行清楚完整的描述：

[0034] 如图1和图2所示：

[0035] 设定：

[0036] Q1服务器编码前的视频数据缓存部分的容量；

[0037] Q2服务器编码后的视频数据缓存部分的容量；

[0038] Q3客户端解码前的数据缓存部分的容量；

[0039] Q4客户端解码后的数据缓存部分的容量；

[0040] len1服务器编码前的视频数据缓存部分(Q1)的实际数据量；

[0041] len2服务器编码后的视频数据缓存部分(Q2)的实际数据量；

[0042] len3客户端解码前的数据缓存部分(Q3)的实际数据量；

[0043] len4客户端解码后的数据缓存部分(Q4)的实际数据量；

[0044] f客户的帧率；在以下的各实施例中均假设为60(帧每秒)

[0045] d客户设置的自己可忍受的延迟；在以下的各实施例中均假设为200(毫秒)

[0046] ds服务器端可容忍延迟；在以下的各实施例中均假设为200(毫秒)

[0047] cs客户端可容忍延迟;在以下的各实施例中均假设为200(毫秒)

[0048] u一定时间内的平均延迟;在以下的各实施例中均假设为50(毫秒)

[0049] Q1min,Q2min,Q3min,Q4min都假设为1

[0050] Q1max,Q2max,Q3max,Q4max都假设为10

[0051] 实施例1,服务器缓冲区Q1和Q2、客户端缓冲区Q3和Q4容量不变且容量均一的情况:

[0052] Q1=(d-u)/(1000/f)/4=(200-50)/(1000/60)/4=2.25=2(四舍五入)

[0053] 而此时Q1>Q1min并且Q1<Q1max,所以Q1=2

[0054] 同理可以得到Q2、Q3、Q4的大小,在整个程序运行过程中Q1、Q2、Q3、Q4是不变的。

[0055] 计算得出的Q为关于视频帧数和延迟的一个参考的比例关系,Q1、Q2、Q3和Q4的设置可以参照Q值进行设置。

[0056] 当即将进入某缓冲区的帧数据量超过所述各缓冲区的容量时,各缓冲区抛弃超出部分的帧数据。当len1、len2、len3和len4超过2时,相应的缓存就抛弃对应的视频帧数据。

[0057] 相对于现有技术中大量抛弃连贯的视频帧,使用多个分散的视频帧数据缓冲区,少量多次的抛弃堵塞的视频帧,在网络拥堵环境下仍能保证一定的视频流畅度,即使经过分散的抛帧,将每秒视频帧的数量降低至24帧以下,通常也只能造成视频出现短时间卡顿,而不会造成大量连续视频数据的丢失。

[0058] 实施例2,服务器端的缓冲区Q1和Q2共享一定容量的缓冲区;客户端的缓冲区Q3和Q4共享客户端所在计算机一定容量的缓冲区。

[0059] Q1=(ds-u/2)/(1000/f)/2=(200-50/2)/(1000/60)/2=5.25=6。Q1采用过0进1的方式,所以取6。考虑到,在实际的视频编码过程中,编码前的数据量要远大于编码后的数据量,故在计算Q1的容量时,尽可能的使Q1的容量大于Q2的容量,即在本实施例中采用过0进1的方式。

[0060] 而此时Q1>Q1min并且Q1<Q1max,所以Q1=6

[0061] Q2=(ds-u/2)/(1000/f)-Q1=(200-50/2)/(1000/60)-6=4.5=5。Q2采用四射五入的方式,所以取1。

[0062] 同理

[0063] Q4=(cs-u/2)/(1000/f)/2=(200-50/2)/(1000/60)/2=5.25=6。Q4采用过0进1的方式,所以取2。同样的,在客户端,解码后的视频帧数据量,要大于经过封装便于在网络中传输的数据包的大小,故在计算Q4时,尽可能的使Q4的容量大于Q3的容量。

[0064] 而此时Q4>Q4min并且Q4<Q4max,所以Q4=6

[0065] Q3=(cs-u/2)/(1000/f)-Q4=(200-50/2)/(1000/60)-6=4.5=5(Q3采用四射五入的方式,所以取1。

[0066] 下面以客户端为例描述动态抛帧以及缓存调整的过程

[0067] 假设某个时间点Q3的容量=7,Q4的容量=4,即将进入Q3的数据len3=5,即将进入Q4的数据len4=4,这时候如果有数据从Q3出来,则出来后Q3=7,Q4=4,len3=4,len4=4,而出来的数据被解码完毕后需要进入Q4,则进入后Q3=6,Q4=5,len3=4,len4=5;

[0068] 假设某个时间点Q3=5,Q4=6,len3=5,len4=3,这时候如果有数据需要进入Q3,则进入后Q3=6,Q4=5,len3=6,len4=3。

[0069] 由于采用了动态调配容量的、分组设置的缓冲区，避免了在服务器端和客户端因某个缓冲区出现饱和，而造成大量抛帧的情况，更进一步的增加了用户的观感体验。

[0070] 实施例3，服务器和客户端动态统一调节各缓冲器容量的情况。

[0071] n=(d-u)/(1000/f)=(200-50)/(1000/60)=9，在服务器端和客户端所需缓冲区的容量综合为9(作为缓存容量和延迟的一个参考数值)。

[0072] Q1=n/4=2.25=3。Q1采用过0进1的方式，所以取3，以保证Q1具有最大的缓存容量(理由与前述一致)

[0073] 而此时Q1>Q1min并且Q1<Q1max，所以Q1=3

[0074] Q2=Q3=Q4=n/4=2.25=2(四舍五入的方式)

[0075] 而此时Q2>Q2min并且Q2<Q2max，所以Q1=2

[0076] 同理推出Q3=2，Q4=2

[0077] 递减顺序Q1，Q4，Q3，Q2，递增顺序为Q2，Q3，Q4，Q1。

[0078] 递减顺序应用于有缓存需要扩大容量的情况，递增顺序为有缓存需要减小容量的情况。

[0079] 假设某个时间点Q2由于长期丢帧需要扩大容量，那么应该先检查Q1是不是有空闲的容量，如果有则Q1减去1，Q2加上1，如果没有，则检查Q4，看Q4是否有空闲容量……以此类推。

[0080] 由于采用了服务器和客户端之间的动态调整，避免了在服务器端或客户端由于网络传输问题造成的单方数据堵塞，避免了大量抛帧情况的发生，提升了用户体验。

[0081] 当所述的四个缓冲区中的某一缓冲区出现空闲时，可以将自身的剩余容量分配给所述其它缓冲区：按Q2，Q3，Q4，Q1的顺序遍历其它缓冲区，若某缓冲区容量小于等待进入的数据量，则将所述的剩余流量传送给该缓冲区。

[0082] 相对于被动等待堵塞情况的发生再进行补救的方式，采用主动分配缓冲区剩余容量，可以做到防患于未然，在堵塞尚未发生时，提前提升帧数据量较大的缓冲区的容量，防止视频帧数据在某一缓冲区、服务器端或客户端造成大量堵塞，减少了抛帧数量，增加了用户体验的流畅性。

[0083] 以上所述，仅为本发明较佳的具体实施方式，但本发明的保护范围并不局限于此，任何熟悉本技术领域的技术人员在本发明揭露的技术范围内，根据本发明的技术方案及其发明构思加以等同替换或改变，都应涵盖在本发明的保护范围之内。

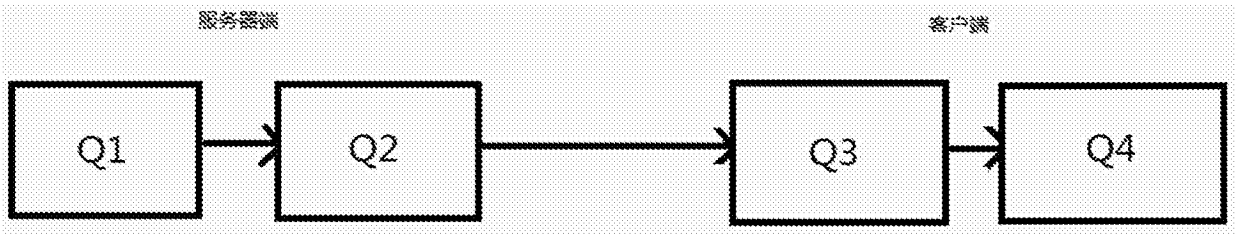


图1

设定服务器中存储编码前视频流缓存为Q1，设定服务器中存储编码后的视频流缓存为Q2；设定客户端存储解码视频前的缓存为Q3，设定客户端存储解码视频后的缓存为Q4

所述缓存Q1、Q2、Q3和Q4具有固定的缓存容量，当处理的视频流超过所述的Q1、Q2、Q3或Q4各自的缓存容量时，所述的Q1、Q2、Q3或Q4抛弃即将进入自身缓冲区且超过缓存范围的帧数据。

图2