计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.190500092

基于运动矢量细化的帧率上变换与 HEVC 结合的视频压缩算法

蔡于涵　熊淑华　孙伟恒　Karn Pradeep　何小海
四川大学电子信息学院　成都 610065
(240947104@qq.com)

摘　要　将帧率变换技术与新型视频压缩编码标准 HEVC 相结合有利于提升视频的压缩效率。针对直接利用 HEVC 码流信息中的低帧率视频的运动矢量进行帧率上变换时效果不理想的问题,文中提出了一种基于运动矢量细化的帧率上变换与 HEVC 结合的视频压缩算法。首先,在编码端对原始视频进行抽帧,降低视频帧率;其次,对低帧率视频进行 HEVC 编解码;然后,在解码端与从 HEVC 码流中提取出的运动矢量相结合,利用前向-后向联合运动估计对其进行进一步的细化,使细化后的运动矢量更加接近于对象的真实运动;最后,利用基于运动补偿的帧率上变换技术将视频序列恢复至原始帧率。实验结果表明,与 HEVC 标准相比,所提算法在同等视频质量下可节省一定的码率。同时,与其他算法相比,在节省码率相同的情况下,所提算法重建视频的 *PSNR* 值平均可提升 0.5dB。

关键词:运动矢量细化;高性能视频编码(HEVC);帧率变换;联合运动估计
中图法分类号　TN919.81

Video Compression Algorithm Combining Frame Rate Up-conversion with HEVC Standard Based on Motion Vector Refinement

CAI Yu-han,XIONG Shu-hua,SUN Wei-heng,Karn PRADEEP and HE Xiao-hai
College of Electronics and Information Engineering,Sichuan University,Chengdu 610065,China

Abstract　Combining frame rate conversion technology with the HEVC standard will improve the compression efficiency of video. Aiming at the non-ideal result in frame rate up-conversion using the motion vector of low frame rate video extracted from HEVC coding bit stream directly,this paper proposed a compression algorithm combining frame rate up-conversion and HEVC based on motion vector refinement. Firstly,at the encoding end,the original even frames are extracted to reduce the frame rate of video,and then the low frame rate video is encoded and decoded by HEVC. Combined with the motion vector extracted from HEVC coding bit stream at the decoding end,the forward-backward joint motion estimation is used to further refine it,which makes the motion vector closer to the real motion of the object. Finally,the frame rate up conversion technique based on motion compensation is used to restore the video to its original frame rate. Experimental results show that compared with the HEVC standard,the proposed algorithm has some bitrate saving. In the meantime,compared with other algorithms,the proposed algorithm can increase the *PSNR* value of reconstructed videos by 0.5 dB on average with the same bitrate saving.

Keywords　Refinement of motion vector,High efficiency video coding(HEVC),Frame rate up-conversion,Joint motion estimation

1　引言

时域冗余是视频信号的重要特性,可利用这一特性对视频序列进行压缩,这一过程统称为帧率变换。在帧率变换中,可通过降低时间冗余来提升视频压缩效率,该过程被称为帧率下变换。而在视频序列中插入新的视频帧又可以提升视频序列的主观视觉效果,该过程被称为帧率上变换(Frame Rate Up-Conversion,FRUC)。高性能视频编码标准(High Efficiency Video Coding,HEVC)[1]是目前最新一代的视频编码标准,将帧率变换与 HEVC 编码标准相结合可以进一步提升 HEVC 编码标准的压缩性能。本文主要研究将帧率上变换与 HEVC 相结合的视频压缩算法。

到稿日期:2019-05-17　返修日期:2019-09-22　本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:国家自然科学基金资助项目(61871279,61471248);成都市产业集群协同创新项目(2016-XT00-00015-GX)
This work was supported by the National Natural Science Foundation of China (61871279,61471248)and Chengdu Industrial Cluster Collaborative Innovation Project(2016-XT00-00015-GX).
通信作者:熊淑华(xiongsh@scu.edu.cn)

帧率上变换是在原始的视频序列中插入新的视频帧，以提高视频的帧率，因此它可用于减少保持型显示器的运动模糊问题。此外，随着目前多媒体设备和显示技术的不断发展，通过帧率上变换技术可以生成具有更好视觉质量的高帧率(High Frame Rate，HFR)视频[2]，因此在实际应用中，帧率上变换技术的利用率越来越高。

目前，大多数帧率上变换方法都是利用视频序列中帧与帧之间的运动信息沿运动轨迹对待插帧进行插值[3-15]，这种方法被称为基于运动补偿的帧率上变换(Motion Compensated-Frame Rate Up-Conversion，MC-FRUC)。典型的 MC-FRUC 包括运动估计(Motion Estimation，ME)和运动补偿内插(Motion Compensated Interpolation，MCI)。运动估计旨在估计连续帧之间的运动矢量(Motion Vector，MV)，这些运动矢量表示连续帧之间运动物体相对位置的改变。在已有的运动估计算法中，块匹配算法因其简单易懂和便于在硬件中集成而被广泛使用。文献[3，6-9]介绍了几种在传统的块匹配算法基础上进行改进的运动估计算法，提升了 MV 估计的准确度。在完成运动估计之后，文献[4-5，10-15]利用获得的 MV 以不同的方式进行基于运动补偿的插帧，以此来生成更好的内插帧。文献[3]利用双运动估计算法来提高 MV 的准确度，其首先使用双向运动估计得到初始 MV，然后使用单向运动估计来计算这些初始 MV 的单向匹配率，对其中匹配率较低的 MV 进行修正细化。文献[4]利用多假设帧率上变换方案来估计具有最大后验概率的中间帧，该方法利用多个“最佳”运动轨迹来形成一组运动假设，然后利用运动假设生成所有的运动补偿内插帧，最后根据每个假设的可靠性设定相应的加权系数，以此自适应地融合这些内插帧。文献[5]提出一种将帧率变换与 HEVC 标准相结合的新思路。在编码端，先进行自适应抽帧，再对低帧率视频进行编解码；在解码端，利用 HEVC 编码的码流中提取的 MV 以及分块信息对丢失帧的 MV 进行估计，最后使用运动补偿插帧恢复原始帧率视频，该方法有效地提升了 HEVC 标准的压缩性能。

因为 HEVC 标准采用的运动估计并不是基于对象真实运动的估计，所以由 HEVC 编码产生的码流信息中的运动矢量并不准确，故上述文献直接利用 HEVC 码流信息中的运动矢量进行帧率上变换得到的重建视频的质量并不佳。为了解决这一问题，本文提出了一种基于运动矢量细化的帧率上变换与 HEVC 结合的视频压缩算法。首先从 HEVC 编码得到的码流信息中提取低帧率视频编码后的运动矢量，同时对解码后的低帧率视频进行重运动估计，将重运动估计得到的运动矢量用于修正 HEVC 中提取的运动矢量，以提升运动矢量的准确度，使处理后的运动矢量更加接近对象的实际运动，从而提升帧率上变换之后重建视频的质量。

2 相关工作

2.1 HEVC 帧间编码

HEVC 标准中存在两种预测模式：帧内预测和帧间预测。HEVC 的帧间预测使用已经编码的重建帧来预测当前帧，并从原始帧中减去当前帧的预测帧，以获得用于变换量化的残差。HEVC 帧间预测包含 3 种特殊的模式，分别为合并模式(Merge)、先进运动矢量预测模式(AMVP)以及跳过模式(Skip)。

Merge 模式采用相邻预测块(Prediction Unit，PU)的运动信息来预测当前 PU 块的 MV。其首先构建一个参考列表，列表中的元素由时空域相邻 PU 块的 MV 构成，编码器会从中选出最佳的运动信息，并将其传到解码端。AMVP 模式构造类似的候选预测 MV 列表，编码器从该列表中选择用于差分编码的最佳预测 MV，并将运动矢量残差(Motion Vector Difference，MVD)发送到解码端。Skip 模式是 Merge 模式的一种特殊情况，其不需要传输残差数据，只需要传送 MV 的索引和一个 Skip flag。3 种帧间的预测模式均可得到 PU 块的 MV 信息，HEVC 标准将视频序列的 MV 信息全部存入编码产生的码流信息中[16]。

2.2 三维递归搜索

传统的运动估计方法都是基于块匹配的，例如全搜索法、三步搜索法、菱形搜索法等。以上算法只是找到令两个相似块之间的绝对误差和最小的位移矢量，因此可能并不符合物体运动的真实轨迹；并且在实际中的信号传输中会有各种噪声的干扰，或者是由于外界光线强度突然变化导致视频序列发生突变，在这些情况下，传统的块匹配算法不能准确地获得对象的位移矢量。为了使通过运动估计获得的运动矢量更接近对象的实际运动，本文引入了递归搜索的概念。

递归搜索分为一维递归、二维递归以及三维递归，目前在运动估计算法中应用较好的是三维递归搜索(3-D Rercursive Search，3DRS)。在实际运动中，属于同一对象的部分的位移轨迹在时空域都有很强的相关性，三维递归搜索就是利用了这一特性。惯性是自然界中的运动物体都具有的特性，因此物体的运动在时域具有很强的相关性，我们可以通过前一帧已经得到的目标对象的 MV 相应地获得其在当前帧中的 MV。而空域的相关性则是因为单帧中运动目标的尺寸一般比匹配所用宏块的尺寸大，所以在运动估计时属于同一目标的宏块之间的位移在空间域的相关性较强。基于此，可以缩小运动估计时的搜索步长，从空间相邻的宏块中获得当前块的 MV。三维递归搜索通过构造一个运动矢量候选集来获得当前宏块的最佳运动矢量，如图 1 所示。

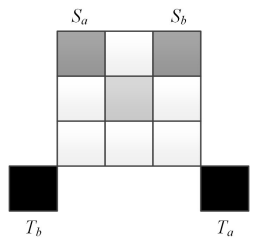


图 1 三维递归搜索的运动矢量候选集

Fig. 1 Motion vector candidate of 3DRS

运动矢量候选集包括与当前块空间相邻的左上和右上块

的运动矢量 S_a 和 S_b、与当前块时间相邻的左下和右下块的运动矢量 T_b 和 T_a、更新后的两个空间候选矢量、更新后的两个时间候选矢量以及零矢量这 9 个运动矢量[17]。

在进行三维递归搜索时，每一个宏块的运动矢量的获取都是一个递归过程，在每次递归中，从运动矢量候选集中选出最佳的运动矢量，将其作为当前块的运动矢量。

3 基于运动矢量细化的帧率上变换与 HEVC 结合的视频压缩算法

本文算法的总体框图如图 2 所示。首先，将原始视频序列进行偶数帧的抽取，保留奇数帧，以此降低原视频的帧率。其次，将抽帧后的低帧率视频序列放入 HEVC 编码框架中，用 encoder_lowdelay_P_main.cfg 配置编码模式进行编码，将编码得到的码流文件传送至解码端，在解码端对其进行解码。通过解码可以得到重建的低帧率视频，同时将解码端接收到的码流数据中的 MV 提取出来。接下来，将解码后的低帧率重建视频进行 1/2 下采样，对下采样之后的视频进行前向-后向联合运动估计，以此得到前向以及后向两类 MV。然后，对前、后向 MV 以及 HEVC 码流中的前向 MV 进行修正和细化。最后，将细化后的 MV 用于执行运动补偿插帧，以恢复原始帧率的视频。

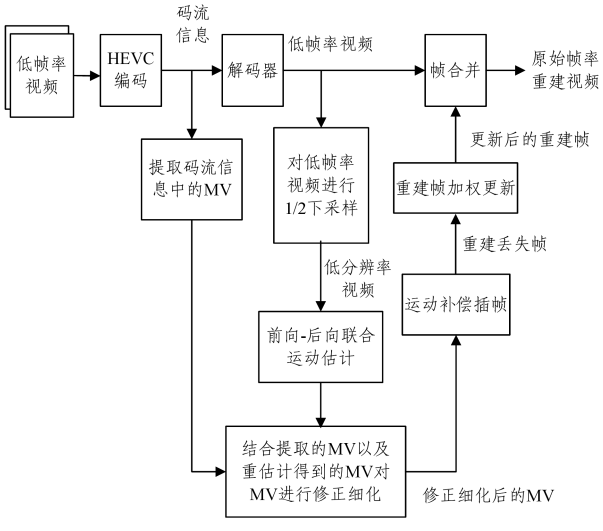


图 2 基于运动矢量细化的帧率上变换与 HEVC 结合的视频压缩算法框图

Fig. 2 Block diagram of video compression algorithm combining frame rate up-conversion with HEVC standard based on motion vector refinement

3.1 前向-后向联合运动估计

本文方法须在外部对解码后的低帧率视频进行重运动估计，由于视频的分辨率越小，其数据量就越少，因此为了减小重估计时的计算量，首先将解码后的低帧率视频进行 1/2 下采样，使视频的尺寸变为原始尺寸的 1/2；之后对下采样后的视频序列利用前向-后向联合运动估计得到前、后向两类 MV，其中前向运动估计如图 3(a)所示，后向运动估计如图 3(b)所示。

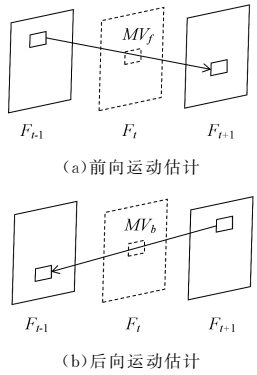


图 3 前向-后向运动估计

Fig. 3 Forward-backward motion estimation

图 3(a)中，F_{t-1} 表示待插帧的前一时刻帧，F_t 表示待插帧，F_{t+1} 表示待插帧的后一时刻帧。图 3(a)的前向运动矢量 MV_f 可由式(1)和式(2)求得。

$$MV_f=\arg\min_{v\in CS_B}\{SAD_f\} \tag{1}$$

$$SAD_f(v_x,v_y)=\sum_{(x,y)\in B}|f_{t-1}(x-v_x,y-v_y)-f_{t+1}(x+v_x,y+v_y)| \tag{2}$$

图 3(b)中，后向运动矢量 MV_b 的计算与 MV_f 的过程类似，如式(3)和式(4)所示：

$$MV_b=\arg\min_{v\in CS_B}\{SAD_b\} \tag{3}$$

$$SAD_b(v_x,v_y)=\sum_{(x,y)\in B}|f_{t+1}(x-v_x,y-v_y)-f_{t-1}(x+v_x,y+v_y)| \tag{4}$$

式(1)－式(4)中，CS_B 为全搜索运动估计时搜索范围内的运动矢量候选集，B 为待插帧中的当前宏块，(x,y) 为当前宏块中像素的坐标值，(v_x,v_y) 为宏块对应的运动矢量，f_{t-1} 为待插帧的前一时刻帧，f_{t+1} 为待插帧的后一时刻帧，SAD_f 为 f_{t-1} 中 (x,y) 处宏块与 f_{t+1} 中对应宏块之间的绝对误差之和(Sum of Absolute Difference，SAD)，SAD_b 则为 f_{t+1} 中 (x,y) 处宏块与 f_{t-1} 中对应宏块之间的绝对误差之和。通过式(1)和式(3)可分别找出令 SAD_f 和 SAD_b 最小的 (v_x,v_y)，将其分别作为前向运动矢量 MV_f 以及后向运动矢量 MV_b。

3.2 运动矢量细化

完成前向-后向运动估计后，待插宏块中存在两类 MV，第一类为前向 MV，其包括 HEVC 码流中的运动矢量 MV_{HEVC} 以及联合运动估计得到的前向 MV_f；第二类为后向 MV，即联合运动估计得到的后向 MV_b。因为 HEVC 标准在对一帧图像进行帧间编码时采用的是基于块匹配的运动估计算法，并不是基于对象的真实运动，所以得到的 MV 并不准确，同时在 HEVC 帧间编码时，某些编码单元会根据率失真代价选择帧内编码模式来进行编码，不会产生运动矢量信息，从而导致在提取运动矢量时这些宏块没有 MV，因此需要对码流中的运动矢量进行处理。为了得到待插宏块最佳的前向以及后向运动矢量，本节将采用邻域递归搜索分别对两类 MV 进行修正细化。

在前向运动矢量细化过程中，首先仍利用 SAD 对待插帧中当前块位置的 MV 的匹配率进行判断，如式(5)和式(6)所示：

$$SAD_{match}(v_{x(HEVC)},v_{y(HEVC)})=\sum_{(x,y)\in B}|f_{t-1}(x-v_{x(HEVC)},y-v_{y(HEVC)})-f_{t+1}(x+v_{x(HEVC)},y+v_{y(HEVC)})| \tag{5}$$

$$SAD_{match}(v_{x(MV_f)},v_{y(MV_f)})=\sum_{(x,y)\in B}|f_{t-1}(x-2v_{x(MV_f)},y-2v_{y(MV_f)})-f_{t+1}(x+2v_{x(MV_f)},y+2v_{y(MV_f)})| \tag{6}$$

其中，$(v_{x(HEVC)},v_{y(HEVC)})$和$(v_{x(MV_f)},v_{y(MV_f)})$分别为 HEVC 码流中的运动矢量和联合运动估计得到的前向运动矢量；SAD_{match}为宏块间的匹配率。由于在重运动估计时对视频序列进行了 1/2 下采样，视频尺寸变为了原始尺寸的 1/2，在为当前块搜索对应的匹配块时，位移矢量也相应地变为原始尺寸下位移矢量的 1/2，因此在式(6)中计算 MV_f 的匹配率时，须将其扩大 2 倍以对应原始尺寸的位移矢量。

若 $SAD_{match}(v_{x(HEVC)},v_{y(HEVC)})<SAD_{match}(v_{x(MV_f)},v_{y(MV_f)})$，则说明由 HEVC 码流中提取的 MV 进行块匹配得到的结果更准确，因此将 HEVC 码流中提取的 MV_{HEVC} 直接作为当前宏块的最佳前向 MV_{fBest}；反之，将对当前块的前向 MV 进行细化。在细化过程中，本节提出了一种邻域递归搜索，首先对每一个宏块的运动矢量进行运动矢量候选集的构建。由于在突变场景中时间候选块与当前块的运动相关性并不大，而且引入时间候选矢量会增加计算的复杂度，导致时间代价增加，因此本节摒弃三维递归搜索的时间候选矢量，将候选矢量集改为当前宏块以及其上方、左方、左上方以及右上方宏块的运动矢量，利用同一帧中块与块之间的空间相关性很强的特性来搜索最佳运动矢量，如图 4 所示。

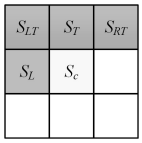


图 4 邻域递归搜索的运动矢量候选集

Fig. 4 Motion vector candidate set of neighboring recursive search

运动矢量候选集如式(7)所示：

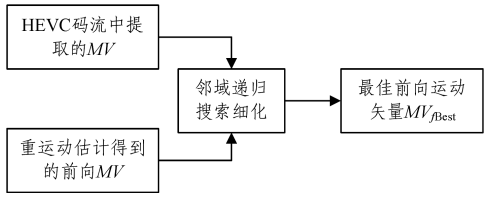
$$CS(X,t)=\{S_c(X,t)+U\}\cup\{S_L(X,t)+U\}\cup\{S_{LT}(X,t)+U\}\cup\{S_T(X,t)+U\}\cup\{S_{RT}(X,t)+U\} \tag{7}$$

其中，$S_C(X,t)$，$S_L(X,t)$，$S_{LT}(X,t)$，$S_T(X,t)$和 $S_{RT}(X,t)$分别为当前块以及其上方、左方、左上方以及右上方宏块的运动矢量；U 是更新矢量，其定义如式(8)所示：

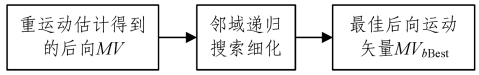
$$U=\{(0,0),(0,-1),(0,1),(0,2),(0,-2),(1,0),(-1,0),(3,0),(-3,0)\} \tag{8}$$

更新矢量 U 是为了降低噪声的影响，每次计算时从 U 中随机挑选一个更新矢量加入到运动矢量中。构建完运动矢量候选集之后，依次将候选集中的候选矢量作为待插块的 MV，找到待插块在前后两个参考帧中对应的参考块并计算对应的 SAD 值，其中令 SAD 值最小的一组运动矢量为当前待插宏块的最佳前向 MV_{fBest}。在完成当前待插块的运动估计后，将该待插块的 MV 作为下一个待插块的运动矢量候选集中的元素，以此完成对待插帧中全部待插宏块的运动估计。

对于后向 MV_b，同样基于上述过程中的邻域递归搜索对其进行单独的细化得到最佳后向 MV_{bBest}，在细化时仍须将 MV_b 扩大 2 倍。整个过程如图 5 所示，图 5(a)表示对前向 MV 的细化过程，图 5(b)表示对后向 MV 的细化过程。



(a)前向运动矢量细化



(b)后向运动矢量细化

图 5 运动矢量的细化过程

Fig. 5 Refinement process of motion vector

3.3 基于前向-后向运动矢量的帧率上变换技术

帧率上变换技术包括帧平均、帧重复以及运动补偿内插。本节基于 3.2 节中得到的 MV_{fBest} 和 MV_{bBest} 进行运动补偿内帧。与非运动补偿内插帧方法相比，运动补偿内插考虑了对象的运动信息，因此运动补偿插帧可以有效地减少内插帧的运动模糊和运动抖动。

在进行运动补偿插帧时，本节首先对宏块的 MV 进行分类，当 MV_{fBest} 与 MV_{bBest} 中有一个为 0 时，我们认为该待插宏块为静止宏块，即该宏块 MV 为 0，在对其进行插帧时，直接利用待插帧的后一帧中相同位置的宏块得到待插帧中待插宏块的像素值，如图 6 所示。

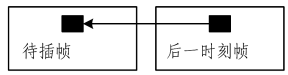


图 6 静止块插值

Fig. 6 Interpolation of static block

当 MV_{fBest} 与 MV_{bBest} 都不为 0 时，认为该待插宏块为运动宏块，在插值恢复时先根据 3.2 节中得到的两类最佳运动矢量为待插宏块在前后参考帧中寻找最为相似的两块，然后比较由 MV_{fBest} 得到的两个相似宏块之间的 SAD 值和由 MV_{bBest} 得到的两个相似宏块之间的 SAD 值，如式(9)和式(10)所示：

$$SAD_{f'}=\sum_{(x,y)\in B}|f_{t-1}(x+v_{fx},y+v_{fy})-f_{t+1}(x-v_{fx},y-v_{fy})| \tag{9}$$

$$SAD_{b'}=\sum_{(x,y)\in B}|f_{t-1}(x+v_{bx},y+v_{by})-f_{t+1}(x-v_{bx},y-v_{by})| \tag{10}$$

其中，(v_{fx},v_{fy})和(v_{bx},v_{by})分别表示前向运动矢量和后向运动矢量。如果 $SAD_{f'}<SAD_{b'}$，那么待插帧中对应位置的宏块将根据由 MV_{fBest} 得到的两个相似宏块内插得到；反之，如果 $SAD_{f'}>SAD_{b'}$，那么待插帧中对应位置的宏块将根据由 MV_{bBest} 得到的两个相似宏块内插得到。待插块的插值过程如式(11)和式(12)所示：

$$\bar{v_x}=\pm\frac{1}{2}v_x,\bar{v_y}=\pm\frac{1}{2}v_y \tag{11}$$

$$F_t(x,y)=\frac{1}{2}[F_{t-1}(x+\bar{v_x},y+\bar{v_y})+F_{t+1}(x-\bar{v_x},y-\bar{v_y})] \tag{12}$$

其中，$F_t(x,y)$表示待插帧在(x,y)处的重建值，F_{t-1}和F_{t+1}分别表示待插帧的前一时刻帧和后一时刻帧，(v_x,v_y)为待插宏块的运动矢量。图7展示了利用MV_{fBest}进行待插块插值的过程。

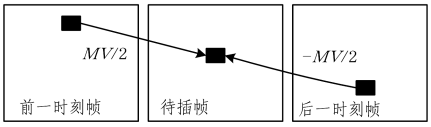


图7 运动块插帧

Fig. 7 Interpolation of moving block

在得到完整的内插帧后，本节将对内插帧进行更新[11]。在更新过程中，首先根据3.2节中得到的MV_{fBest}和MV_{bBest}分别进行插值恢复，得到前向内插帧F_t^f和后向内插帧F_t^b；然后通过式(13)对式(12)得到的内插帧F_t进行更新。

$$F_t'=\frac{d(F_t,F_t^f)\times F_t^f+d(F_t,F_t^b)\times F_t^b}{d(F_t,F_t^f)+d(F_t,F_t^b)} \tag{13}$$

其中，$d(x,y)=1/\|x-y\|$，表示x与y之间的相似程度；F_t'为恢复的内插帧；F_t'为更新后的内插帧，作为最终恢复的偶数帧。式(13)中前向内插帧F_t^f与后向内插帧F_t^b的权重分别为$d(F_t,F_t^f)/(d(F_t,F_t^f)+d(F_t,F_t^b))$和$d(F_t,F_t^b)/(d(F_t,F_t^f)+d(F_t,F_t^b))$。如果前向内插帧和恢复的内插帧更接近，那么$d(F_t,F_t^f)$就越大，其所占权重也就越大；反之，$d(F_t,F_t^b)$所占权重就越大。

4 实验结果与分析

为验证本文算法的性能，采用标准的测试环境对HEVC视频序列进行测试[18]。测试序列的实验平台的内存为4.00 GB，CPU为Intel(R) Core(TM) i5-3470 CPU@3.20 GHz。本文采用的HEVC编解码器为官方标准软件HM16.0，配置文件使用encoder_lowdelay_P_main.cfg。在实验过程中，对不同分辨率的视频序列选取前50帧进行测试，测试时的量化参数QP分别选择22,27,32,37,42。为了验证本文提出的运动矢量细化方法的有效性，将所提算法与HEVC编码标准以及文献[5]提出的算法进行对比，具体实验结果如表1所列。其中*bitrate*为HEVC编码后的码率，单位为Kbps；$PSNR$为重建视频的峰值信噪比，单位为dB；$\Delta PSNR$为所提算法与文献[5]中的算法重建后视频$PSNR$值的差值。

表1 本文算法与对比算法的实验结果

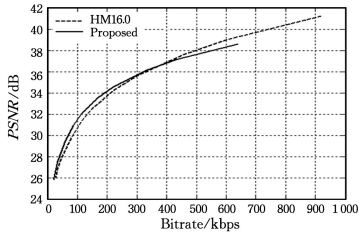
Table 1 Experimental results of proposed algorithm and comparative algorithms

序列	QP	HM16.0		文献[5]		本文算法		$\Delta PSNR$
		bitrate	PSNR	bitrate	PSNR	bitrate	PSNR	
416×240 BasketballPass	22	921.208	41.249	641.992	37.174	641.992	38.631	1.457
	27	457.208	37.600	331.864	34.921	331.864	36.124	1.203
	32	224.592	34.285	166.072	32.674	166.072	33.495	0.821
	37	114.824	31.350	85.496	30.475	85.496	30.938	0.463
	42	60.704	28.694	44.920	28.192	44.920	28.428	0.236
832×480 PartyScene	22	12494.504	38.250	7778.024	33.024	7778.024	33.966	0.942
	27	5432.384	34.136	3575.072	30.654	3575.072	31.467	0.813
	32	2432.184	30.612	1671.232	28.593	1671.232	29.120	0.527
	37	1087.944	27.457	777.440	26.420	777.440	26.721	0.301
	42	449.648	24.549	332.352	24.141	332.352	24.257	0.116
1280×720 FourPeople	22	2752.781	42.441	2007.235	41.306	2007.235	41.767	0.461
	27	1057.056	40.290	867.034	39.572	867.0335	39.912	0.340
	32	544.003	37.785	467.610	37.312	467.610	37.555	0.243
	37	302.026	34.893	261.427	34.639	261.427	34.772	0.133
	42	165.600	31.763	143.770	31.650	143.770	31.699	0.049
1920×1080 ParkScene	22	9458.968	39.834	6593.841	36.301	6593.841	36.994	0.693
	27	3791.293	37.045	2799.641	34.687	2799.641	35.167	0.480
	32	1623.663	34.336	1231.404	32.819	1231.404	33.158	0.339
	37	709.524	31.828	547.676	30.949	547.676	31.142	0.193
	42	295.907	29.491	231.022	28.998	231.022	29.135	0.137

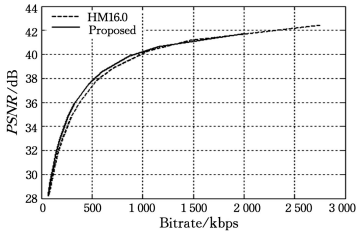
从表1可以看出，在不同的量化参数QP下，对于不同分辨率的视频序列，相比HEVC直接编码原始帧率视频，本文算法的$PSNR$值略有降低。这是由于本文算法先将偶数帧抽弃，在低帧率视频编解码之后对偶数帧进行运动补偿插帧恢复，因此相比原始视频直接进行HEVC编码，插帧恢复的重建视频$PSNR$值肯定会有所下降。但是，因为本文算法进行了抽帧处理，所以在码率上相比于HM16.0有较大的节省，并且随着QP的增大，$PSNR$值的差距在不断减小，说明了本文算法的有效性。

为了更好地表明本文算法比HEVC标准更具优势，图8给出了几个序列的率失真(RD-Cost)曲线，其中横轴为视频的码率，纵轴为视频的客观质量$PSNR$。从图8可以看出，本

文算法在中低码率段优于 HEVC 标准。



(a)BasketballPass



(b)FourPeople

图 8 不同序列的率失真曲线

Fig. 8 Rate distortion curves of different sequences

为了验证本文所采用的式(13)中的帧更新步骤的有效性，我们选取了两个不同分辨率的视频序列，将本文算法不包含帧更新步骤的 *PSNR* 值与本文算法包含帧更新步骤的 PSNR 值进行了比较，结果如表 2 所列。

表 2 本文算法是否包含帧更新步骤的实验结果比较

Table 2 Comparison of experimental results on whether proposed algorithm contains frame update steps

序列	*QP*	*PSNR*（不含帧更新）	*PSNR*（包含帧更新）
1 280×720 FourPeople	22	41.520	41.767
	27	39.776	39.912
	32	37.450	37.555
	37	34.720	34.772
	平均	38.367	38.502
1 920×1 080 ParkScene	22	36.792	36.994
	27	35.034	35.167
	32	33.064	33.158
	37	31.094	31.142
	平均	33.996	34.115

可以看出，对于视频序列 FourPeople 以及 ParkScene，本文算法进行帧更新后的 *PSNR* 值均比不进行帧更新步骤有一定的提升，其中，FourPeople 视频序列 *PSNR* 的平均提升效果为 0.135 dB，而 ParkScene 视频序列 *PSNR* 的平均提升效果也达到了 0.119 dB，这说明本文算法中式(13)的帧更新步骤对重建视频的质量具有一定的提升效果。

最后，为了更加直观地将本文的视频压缩编码方法与文献[5]的方法进行对比，对两种算法在上述 5 个 *QP* 下得到的重建视频的 *PSNR* 值取平均，并进行比较，结果如表 3 所列。可以看出，与文献[5]中的算法相比，本文算法在重建视频的 *PSNR* 值上有明显提升，对于小分辨率序列的提升效果最好，如对于 BasketballPass 序列可以提升 0.8 dB，对于 PartyScene 序列可以提升 0.5 dB；而本文算法对于分辨率较高的视频序列的提升效果一般，平均提升了 0.2 dB，这是因为其运动对象相较于背景范围较小，同时运动剧烈程度相比低分辨率视频较为缓慢，所以对于恢复的内插帧，本文算法虽然在运动区域相较于文献[5]的算法有所提升，但是对整帧图像的效果提升不及低分辨率视频。本文算法对于 4 个分辨率大小的视频序列在客观质量上平均可以提升 0.5 dB。

表 3 本文算法与其他算法的实验结果比较

Table 3 Comparison of experimental results of proposed algorithm and other algorithms

序列	文献[5]	本文算法	*ΔPSNR*
BasketballPass_416×240	32.687 2	33.523 2	0.836 0
PartyScene_832×480	28.566 4	29.106 2	0.539 8
FourPeople_1 280×720	36.895 8	37.141 0	0.245 2
ParkScene_1 920×1 080	32.750 8	33.119 2	0.368 4
平均	32.725 0	33.222 4	0.497 4

图 9 给出了 BasketballPass 序列在 *QP* 为 22 时第 2 帧图像的主观比较，图 9(a)－图 9(d)分别为原始帧、经过 HEVC 直接编码的帧、文献[5]恢复的帧以及本文算法恢复的帧。可以看出，文献[5]中的算法在人的手臂以及腿部存在许多块效应，同时在数字 11 处存在重影，而本文算法明显改善了重影以及块效应，且与 HEVC 直接编码的帧在主观效果上十分接近，由此可以证明本文算法的有效性。

(a)原始帧 (b)HEVC 编码帧

(c)文献[5]重建帧 (d)本文算法重建帧

图 9 主观效果比较

Fig. 9 Comparison of subjective effects

结束语 为了解决直接利用 HEVC 码流信息中的低帧率视频的 MV 进行帧率上变换时效果不理想的问题，本文提出了一种基于运动矢量细化的帧率上变换与 HEVC 结合的视频压缩算法。首先将 HEVC 码流内的运动矢量信息提取出来，然后利用前向-后向联合运动估计得到的 MV 对编码后的低帧率视频进行邻域递归搜索的细化，最后进行运动补偿插帧，恢复原始帧率的视频。实验结果表明，与 HM16.0 相比，本文算法在中低码率段优于 HEVC 编码方法，同时与其他同类文献相比在节省码率相同的情况下，其客观质量 *PSNR* 平均提升了 0.5 dB。但是，本文在重运动估计时还可以利用更加精确的运动估计算法，如光流法等，因此下一步的研究方向是使用更精确的运动估计算法来替代重估计时的块匹配算法。

参 考 文 献

[1] SULLIVAN G J, OHM J, HAN W J, et al. Overview of the high efficiency video coding (HEVC) standard[J]. IEEE Transactions on Circuits & Systems for Video Technology, 2012, 22(12): 1649-1668.

[2] FEGHALI R, SPERANZA F, WANG D, et al. Video quality metric for bitrate control via joint adjustment of quantiza-tion and frame rate[J]. IEEE Transactions on Broadcasting, 2007, 53(1): 441-446.

[3] KANG S J, YOO S, KIM Y H. Dual motion estimation for frame rate up-conversion [J]. IEEE Transactions on Circuits & Systems for Video Technology, 2010, 20(12): 1909-1914.

[4] LIU H, XIONG R, ZHAO D, et al. Multiple hypotheses Bayesian frame rate up-conversion by adaptive fusion of motion compensated interpolations [J]. IEEE Transactions on Circuits & Systems for Video Technology, 2012, 22(8): 1188-1198.

[5] WU Q D, HE X H, LIN H W, et al. Novel video compression coding algorithm combining frame rate conversion and HEVC standard [J]. Acta Automatica Sinica, 2018, 44(9): 1626-1636.

[6] CHOI B D, HAN J W, KIM C S, et al. Motion-compensated frame interpolation using bilateral motion estimation and adaptive overlapped block motion compensation[J]. IEEE Transactions on Circuits & Systems for Video Technology, 2007, 17(4): 407-416.

[7] KIM D, LIM H, PARK H. Iterative true motion estimation for motion compensated frame interpolatio[J]. IEEE Transactions on Circuits & Systems for Video Technology, 2013, 23(3): 445-454.

[8] DIKBAS S, ALTUNBASAK Y. Novel true-motion estimation algorithm and its application to motion-compensated temporal frame interpolation[J]. IEEE Transactions on Image Processing, 2013, 22(8): 2931-2945.

[9] KIM U S, SUNWOO M H. New frame rate up-conversion algorithms with low compu tational complexity[J]. IEEE Transactions on Circuits & Systems for Video Technology, 2014, 24(3): 384-393.

[10] LEE W H, CHOI K, RA J B. Frame rate up-conversion based on variational image fusion[J]. IEEE Transactions on Image Processing, 2014, 23(1): 399-412.

[11] LU G, ZHANG X, CHEN L, et al. A novel frame rate up conversion using iterative non-local means interpolation [C]// 2017 IEEE International Symposiumon Broadband Multimedia Systems and Broadcasting. IEEE, 2017: 1-5.

[12] XU C. Frame rate up-conversion with true motion estimation and adaptive motion vector refinement[C]// International Congress on Image & Signal Processing. IEEE, 2011: 15-17.

[13] HE J, YANG G, SONG J, et al. Hierarchi-cal prediction-based motion vector refinement for video frame-rate up-conversion[J]. Journal of Real-Time Image Processing, 2018, 16(60): 1-15.

[14] GUORIUI C. Frame rate up conversion algorithm based on adaptive agent motion compensation combined with semantic feature analysis[J]. Journal of Ambient Intelligence and Humanized Computing, 2018, 10(54): 1-8.

[15] GUO D, LU Z. Motion compensated frame interpolation with weighted motion estimation and hierarchical vector refinement [J]. Neurocomputing, 2016, 18(1): 76-85.

[16] ZHU W, SHANG M J, RONG Y, et al. Shot Boundary Detection MethodBased on HEVC Compressed Domain[J]. Computer Science, 2019, 46(2): 271-278.

[17] DE H G, BIEZEN P W A C, HUI J H, et al. True-motion estimation with 3-D recursive search block matching [J]. IEEE Transactions on Circuits and Systems for Video Technology, 1993, 3(5): 368-379.

[18] BOSSEN F. Common HM Test Conditions and Software Reference Configurations[C]// Proceedings of the 12th JCTVC Meeting. JCTVC, 2013: 1-3.

CAI Yu-han, born in 1995, postgraduate, is not member of China Computer Federation. His main research interests include image/video coding and so on.

XIONG Shu-hua, born in 1969, Ph.D, associate professor, master supervisor. Her main research interests include multimedia communication and signal processing and image processing.