Sanz-Rodríguez, S., Álvarez-Mesa, M., Mayer, T., & Schierl, T.

A parallel H.264/SVC encoder for high definition video conferencing

Sanz-Rodríguez, S., Álvarez-Mesa, M., Mayer, T., & Schierl, T. (2015). A parallel H.264/SVC encoder for high definition video conferencing. Signal Processing: Image Communication, 30, 89–106. https://doi.org/10.1016/j.image.2014.10.003

A Parallel H.264/SVC Encoder for High Definition Video Conferencing

Sergio Sanz-Rodríguez[a], Mauricio Álvarez-Mesa[a], Tobias Mayer[b], Thomas Schierl[c]

[a]*Embedded Systems Architectures, Technische Universität Berlin, Berlin, Germany*
[b]*Image Communication Group, Technische Universität Berlin, Berlin, Germany*
[c]*Multimedia Communications Group, Fraunhofer HHI, Berlin, Germany*

Abstract

In this paper we present a video encoder specially developed and configured for *high definition (HD)* video conferencing. This video encoder brings together the following three requirements: *H.264/Scalable Video Coding (SVC)*, *parallel encoding on multicore platforms*, and *parallel-friendly rate control*. With the first requirement, a minimum quality of service to every end-user receiver over Internet Protocol networks is guaranteed. With the second one, real-time execution is accomplished and, for this purpose, slice-level parallelism, for the main encoding loop, and block-level parallelism, for the upsampling and interpolation filtering processes, are combined. With the third one, a proper HD video content delivery under certain bit rate and end-to-end delay constraints is ensured. The experimental results prove that the proposed H.264/SVC video encoder is able to operate in real time over a wide range of target bit rates at the expense of reasonable losses in rate-distortion efficiency due to the frame partitioning into slices.

Keywords: H.264/Scalable Video Coding (SVC), video conferencing, rate control, high definition, ultra-low delay, parallel processing.

*Corresponding author. Tel.: +49 30 31421-357; fax: +49 30 31422-943.
Email addresses: sergio.sanz@aes.tu-berlin.de. (Sergio Sanz-Rodríguez), mauricio.alvarezmesa@tu-berlin.de (Mauricio Álvarez-Mesa), tobias.mayer@hhi-extern.fraunhofer.de (Tobias Mayer), thomas.schierl@hhi.fraunhofer.de. (Thomas Schierl)

1. Introduction

The increasing advances in video compression standards, network infrastructures as well as visual display technologies have made *high definition* (HD) video conferencing one of most popular multimedia applications over Internet Protocol (IP) networks. Specifically, a video conferencing session involves point-to-point or multipoint real-time video and audio communication for multiple users that possibly are geographically spread, thus resulting in a challenge for video codec designers in order to provide real-time HD video content delivery with a minimum guaranteed *quality of service* (QoS). To this end, the following three key requirements are expected to be considered for a video coding system: an *H.264/Scalable Video Coding (SVC)-based approach*, a *parallel (multi-core) computing architecture*, and a *parallel-friendly rate control algorithm (RCA)*. These requirements are described in the sequel:

- The scalable extension of the H.264/Advanced Video Coding (AVC) standard, named H.264/SVC or simply SVC [1, 2], is capable of delivering high-quality video content adapted to certain QoS imposed by either on-the-fly varying network conditions or the heterogeneity, in terms of display resolutions and computational capabilities, of end-user devices. The use of SVC involves the extraction of either one or a subset of sub-streams from a high-quality bit stream, so that these simpler sub-streams, bearing lower spatio-temporal resolutions or reduced quality versions of the original sequence, can be decoded by a given target receiver. For example, in a video conference session consisting of two target HD receivers, SVC could be used to generate a complete bit stream consisting of two dependency (spatial or quality) layers: a base layer that includes the sub-stream with low-quality compressed video, e.g., 720p@30 *frames per second* (fps), and an enhancement layer that includes additional information to deliver the high-quality version of video content, e.g., 1080p@30fps. Thus, whereas for those low-quality receivers the enhancement layer is dropped to decode only the base layer, for the rest of target receivers the complete bit stream is delivered, unless the current network conditions are not suitable to transmit the whole bit stream and only the base layer must be decoded to get the best possible video quality for such conditions. Furthermore, unlike other well-known coding technologies, such as *simulcasting* and *transcoding*, SVC also provides the following benefits for video conferencing: 1) SVC is able to reduce the transmission bandwidth when

compared to simulcasting, since the redundancies between the different video versions are actually exploited; and 2) due to the fact that the SVC bit stream itself contains all the video versions demanded by the application, no additional transcoding is required, thus reducing the end-to-end delay and, therefore, making the live session more natural.

- In order to accomplish real-time operation, the execution time of the encoder must be below the limits of the target frame rate, e.g., 33.33 ms per frame for 30 fps. For improving the time performance, typically real-time video encoders restrict the available encoding tools, with an acceptable loss in *rate-distortion* (R-D) efficiency, and use also platform specific optimizations such as *single instruction, multiple data* (SIMD) instructions [3]. The computational requirements of the encoder, however, exceeds the capabilities of a single conventional processor, specially when processing HD content combined with a multilayer coding approach such as SVC. In addition to that, processor frequency is not increasing with every technology generation at the same rate as in the past; instead processor manufacturers are building systems with multiple processors (also called *cores*) per chip [4, 5]. Then, in order to achieve real-time operation for multilayer HD coding, parallelization is necessary, and it must scale so that the performance improves with the growing number of cores per chip [6]. It should be noted that, when using SVC for video conferencing, the encoder must be able to process every access unit (defined as the union of all the representations of a picture at a given time instant) within the time limit of the target frame rate (e.g., the same 33.33 ms for 30 fps), and at the same time maintain a low end-to-end delay. Due to this, parallel techniques such as frame-level parallelism or *group of pictures* (GoP)-level parallelism that increases the throughput but do not reduce the frame latency are not well suited. Furthermore, parallelization techniques have to be applied not only to the single layer encoding scenarios where most of the execution time is spent in the main coding loop (motion estimation being the most complex part), but to other functions in SVC such as upsampling filters for spatial scalability that can take an important part of the execution time. As a result, a parallelization strategy for SVC real-time encoding for video conferencing must be able to provide the required performance at the access unit level, while at the same time reduce the frame latency, and must take into account the additional

processing steps using in multilayer applications.

- The variable bit rate nature of compressed video implies that an RCA must be embedded in the video encoder to avoid encoder buffer (and decoder buffer, which performs the complementary process) overflow and underflow, while providing as good as possible quality consistency and R-D performance [7]. Furthermore, given that the ultra-low delay restriction in a video conferencing environment necessarily entails the use of very small buffer sizes, the RCA must also ensure a tight short-term *target bit rate* (TBR) adjustment. To achieve this, the *quantization parameter* (QP) of transform coefficients can be adjusted for every video segment, typically with size of *macroblock* (MB) in low-delay applications. For a proper selection of the QP value, the RCA should properly assign a bit budget for the current video segment considering the video complexity, the specified TBR as well as the *hypothetical reference decoder* (HRD) constraints [8] required to provide deliverable bit streams. It is also worth noticing that, when using slice-level parallelism in a video conferencing application, independent MB-level QP decisions within a picture must be conducted, so conventional RCAs are not longer valid unless a picture-level QP decision strategy is adopted at the expense of higher instantaneous bit rate variations (see Subsection 2.2). In short, an RCA for HD video conferencing should have the following two attributes: *low-complexity* and *parallel-friendly*. The former is recommended to facilitate real-time encoding, whereas the latter is required to provide accurate MB-level QP selection within a slice and, hence, strict buffer control.

In this paper we propose a complete video coding framework for HD video conferencing. Specifically, the SVC standard was used to guarantee a minimum QoS for every end-user receiver. In order to achieve real-time operation, a parallelization strategy that combines slice-level parallelism, for the main encoding loop, and block-level parallelism, for the upsampling and interpolation filters, was implemented. Furthermore, a novel low-complexity parallel-friendly RCA operating at MB level was embedded in the SVC encoder for a proper video content delivering. All these tools will be described in detail later on.

The paper is organized as follows. In Section 2 previous approaches related to parallelism for real-time video coding as well as the state of the art in RC for video conferencing are described. In Section 3 an overview of the SVC

standard is given. In Section 4 the optimized SVC encoder is described in detail, emphasizing on those operations that were parallelized. In Section 5 a detailed description of the proposed MB-level RCA is given. In Section 6 the experimental setup is described and the results are reported and discussed. Finally, in Section 7 conclusions are drawn and future work is outlined.

2. Related Work

2.1. Parallel Encoding for H.264/AVC and SVC

Video codecs, in particular H.264/AVC, have been parallelized using either GoP-level, frame-level, slice-level, MB-level parallelism or combinations of them. Each of these approaches, however, has some limitations such as limited scalability, significant coding losses, high memory requirements, or increased coding delay.

GoP-level parallelism is based on the fact that GoPs are usually independent and can be encoded in parallel. Although very simple and effective, this kind of parallelism introduces high encoding latency and has high memory requirements [9].

Frame-level parallelism consists of processing multiple frames at the same time provided that the motion compensation dependencies are satisfied [10]. Frame-level parallelism is sufficient for multicore systems with just a few cores. Because it is relatively simple to implement and does not cause coding losses, it has been employed in popular H.264/AVC encoders and decoders [11, 12]. This kind of parallelization strategy has a number of limitations, however. First, the parallel scalability is determined by the lengths of the motion vectors. If due to fast motion, motion vectors are long, there is little parallelism. Second, the workload of each core may be imbalanced because the frame decoding time can vary significantly. Finally, frame-level parallelism increases the frame rate but does not improve the frame latency, and because of this is not well suited for video conferencing applications.

In H.264/AVC, as in most current hybrid video coding standards, each frame can be partitioned into one or more slices in order to add robustness to the bitstream. Slices in a frame are completely independent from each other [13] and, therefore, they can also be used for parallel processing. Slices, however, reduce the coding efficiency owing to the break of intra-frame dependencies. Due to that, exploiting slice-level parallelism is only advisable when there are a few slices per frame [14, 15]. A common example of the use

of slice-level parallelism is encoding and decoding for Blu-ray video discs in which 4 slices per frame slices are mandatory for HD content.

Independent MBs inside a frame can also be processed in parallel using a wavefront approach [16]. Furthermore, MBs from different frames can be processed in parallel provided the dependencies due to motion compensation are handled correctly [10]. Entropy (de)coding, however, can only be parallelized at frame (slice) level and, therefore, it has to be decoupled from MB reconstruction [17]. Although this approach has a high scalability [18] it has some limitations too. First, the decoupling of entropy (de)coding and reconstruction increases the memory usage. Furthermore, this strategy only reduces the frame latency for the reconstruction stage but not for the entropy decoding stage.

In order to overcome the limitations of the parallelization strategies employed in H.264/AVC, two tools aiming at facilitating high level parallel processing have been included in the H.265/High Efficiency Video Coding (HEVC) standard [19]: *wavefront parallel processing* (WPP) and *tiles*. Both of these tools allow subdividing of each picture into multiple partitions that can be processed in parallel. With tiles, the picture is divided in rectangular groups of *coding tree blocks* (CTBs) separated by vertical and horizontal boundaries [20]. With WPP, each CTB row of a picture is a separated partition [21]. Compared to slices and tiles, no coding dependencies are broken at partition boundaries with WPP. These tools can be probably used for the scalable extension of HEVC (under development at the time of writing) but cannot be used currently with H.264/SVC.

Some of the techniques mentioned above for non-scalable video coding parallelization have been adapted for the scalable coding case. In [22, 23] the authors propose a variation of GoP-level and frame-level parallelism for temporal and quality scalability in which data dependencies of frames between layers are analyzed and independent frames are scheduled for execution. These methods are not well suited for video conferencing applications because of the increased latency and because the IP..P coding pattern typically used in video conferencing introduces dependencies between all the frames in consecutive access units.

2.2. Rate Control for Video Conferencing Applications

During the recent years, the RC problem has been widely studied for a variety of multimedia applications and video coding standards [24]. Most

of the RCAs proposed in the literature rely on modeling the transform coefficient distribution to derive analytical R-D functions for QP estimation. For example, if a Gaussian *probability density function* (PDF) is considered, a logarithmic function can be inferred [25, 26, 27, 28]. On the other hand, assuming a Laplacian PDF, several R-D models have been proposed: linear model [29, 30, 31, 32], quadratic model [33, 34, 35, 36, 37, 38, 39], ρ-domain model [40, 41, 42, 43] and square root model [44, 45]. Finally, considering a Cauchy PDF, an exponential function can be derived [46, 47, 48, 49, 50] (however, unlike traditional RCAs, in [50] the Lagrange multiplier λ is first calculated and then the QP is derived). In particular, this Cauchy-density-based function has been proved to better fit the transform coefficient distribution, thus resulting in some R-D benefits. In spite of the fact that the RCA is not a normative part of video coding standards, some of the abovementioned RCAs have been part of their reference implementations, specifically: the Test Model Version 5 for MPEG-2 [29], the Verification Model Version 8 for MPEG-4 [33], the Test Model Near-Term 8 for H.263 [26], the Joint Model for H.264/AVC [34] and the Test Model for HEVC [50].

Although these approaches might be used in almost any application scenario, alternative RCAs have been designed to meet the specific demands of certain applications, such as video streaming and broadcast [51, 52, 53, 54], digital storage [55, 56, 57, 58], and video conferencing [59, 60, 61, 62, 63]. As already pointed out in the introduction section, for the particular case of video conferencing, the proposed RCAs aim at a short-term TBR adjustment for buffer overflow and underflow prevention by means of a QP regulation at MB level [59, 60] and, in some cases, at row-of-MB level [61, 62] in order to improve the R-D performance at the expense of higher bit rate fluctuations. To the best of our knowledge, neither of these RCAs, specially those targeted for ultra-low delay applications, is designed for a slice-level parallel coding framework. More specifically, the previously proposed RCAs for video conferencing select the MB (or row-of-MB) QP in a sequential order, that is, without considering the use of slices running in parallel.

2.3. Fast Mode Decision algorithms

Although these methods are beyond the scope of this work, several algorithms for speeding up the selection of the coding mode for enhancement layer MBs have been devised. The general approach is to reduce the considered modes which are tested for an enhancement layer block based on the coding mode used by the co-located block in the base layer [64, 65].

More models based on different types of statistical analysis where developed subsequently [66, 67].

3. Overview of the H.264/SVC Standard

As prior scalable standards, such as MPEG-2 [68], H.263 [69], and MPEG-4 Visual [70], SVC supports the most important scalable coding modes, i.e., *temporal scalability* (TS), *spatial scalability* (SS), and *quality scalability* (QS). The first two provide subsets of the complete bit stream representing the compressed source content at a reduced frame rate for temporal scalability, or a reduced picture size for spatial scalability. Regarding quality scalability, the sub-stream provides the same spatio-temporal resolution as that of the complete bit stream but lower reconstruction fidelity or *signal-to-noise ratio* (SNR). These scalability types are described in more detail in the sequel:

- **Temporal scalability:** This kind of scalable coding is supported by means of GoP structures that are organized into temporal layers. In particular, the pictures belonging to the temporal *base layer* (BL), also named *key* pictures, can be intra (I)-predicted or inter-predicted, this latter by using *unidirectional* (P) or *bidirectional* (B) motion compensation from pictures belonging to the same temporal layer; whereas the pictures of an *enhancement layer* (EL) can be inter-predicted from references belonging to lower layers. The number of temporal layers is determined by the GoP size, defined as the distance, in number of frames, between two consecutive key pictures. This so-called hierarchical coding, besides, has been shown to improve the compression efficiency compared to traditional coding patterns [71, 72].

- **Spatial scalability:** In this scalability mode, a multilayer coding approach is used to encode different picture sizes of the same input video source. The spatial BL provides an AVC compatible bit stream for the lowest required spatial resolution, whereas the remaining layers deal with larger pictures sizes taking advantage of inter-layer prediction tools for the sake of coding efficiency. It is also worth mentioning that a spatial layer may contain several temporal layers as long a hierarchical GoP structure is employed for the encoding process.

- **Quality scalability:** When SNR scalability is considered, different reconstruction quality levels with the same spatio-temporal resolution

are provided. Specifically, the SVC standard defines two types of SNR scalable coding: *coarse grain scalability* (CGS) and *medium grain scalability* (MGS). The first is a special case of spatial scalability with identical picture sizes, whereas the second employs a multilayer coding approach within a spatial layer in order to provide a finer bit rate granularity in the R-D space.

Furthermore, a combined scalability can be used in order to provide sets of sub-streams with different spatio-temporal resolutions and SNR versions (or bit rates) within the complete scalable bit stream. However, a SVC encoder does not have to be configured to support all types of scalability. Actually, the application requirements determine the set of target spatio-temporal resolutions or reconstruction qualities as well as their corresponding QoS and, therefore, the encoder should be configured accordingly.

4. Proposed Parallel H.264/SVC Encoder

The main requirements imposed on the performance of the encoder are: low-latency for video conferencing applications, and real-time operation for HD content at 30fps. Based on them, the possible parallelization methods are selected. Methods such as GoP-level and frame-level parallelism are not well suited because they can increase the frame throughput but the latency is not reduced compared to the single threaded case. MB-level parallelism can only be used for the main encoding loop, but entropy encoding has to be performed sequentially for each frame. As a result, slice-level parallelism in combination with block-level parallelism appears as the most appropriate parallelization strategy. Compared to the non-scalable coding, SVC introduces additional processing steps such as upsampling for SS. This step has to be parallelized too, otherwise it will reduce the maximum application speedup according to Amdahl's Law [6].

The encoder operation is as follows. In each access unit, each layer is encoded sequentially. Inside each layer there are three main stages performed for each frame: *BL/EL-init*, *BL/EL-encode*, *BL/EL-finish*. The *BL/EL-init* phase includes the general initialization of the frame structures and, for SS, upsampling of reconstructed picture, motion vectors and residual information is performed. The *BL/EL-encode* phase contains the main encoding loop of slices including motion estimation and compensation, mode selection, quantization, transform, entropy coding and bitstream writing. The

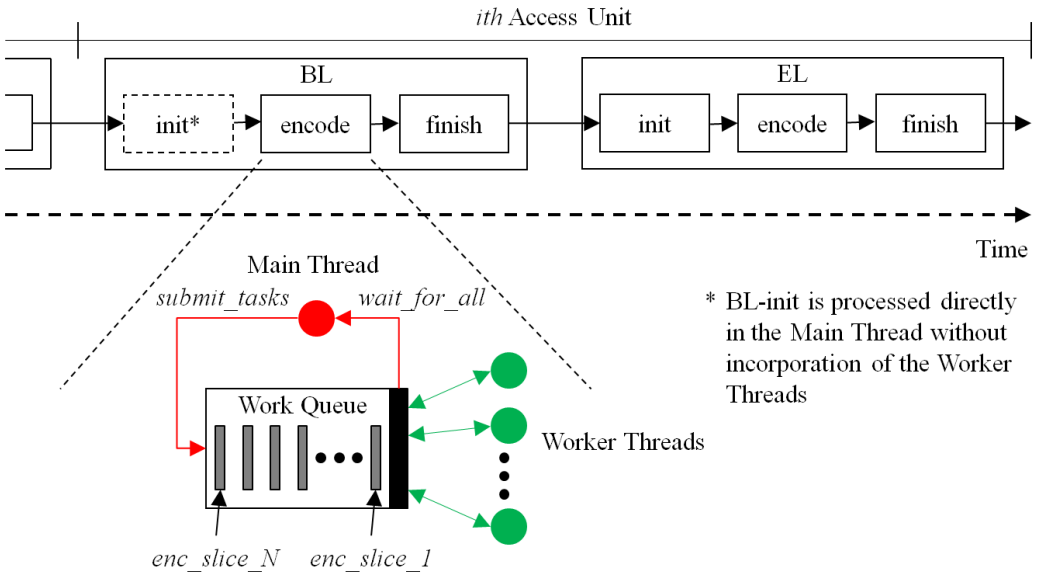


Figure 1: Parallel processing of the encoder in the *BL-encode* phase.

BL/EL-finish phase includes padding and interpolation filtering for subpixel motion estimation.

Slice-level parallelism has been implemented for the main encoding loop in each layer. the slice size is determined based on the number of threads used for each particular run trying to have slices with the same number of MBs. Block-level parallelism has been used for the upsampling and interpolation filtering process. The size of the blocks has been set to one line of MBs, which represents as a good trade off between load balancing and threading overhead. Smaller blocks results in better load balancing at the cost of more thread synchronization overhead.

Figure 1 illustrates an example of the parallel operation of our encoder for the particular case of *BL-encode* corresponding to the *ith* access unit. All parallel processing has been implemented with single-writer multiple-reader work queues. As shown in the figure, the main thread is responsible for preparing and submitting tasks to the queue, and the worker threads take tasks from the queue and execute them to completion. Barriers are inserted between the parallel and sequential phases, and the main thread always waits until all worker threads have finished all their assigned tasks.

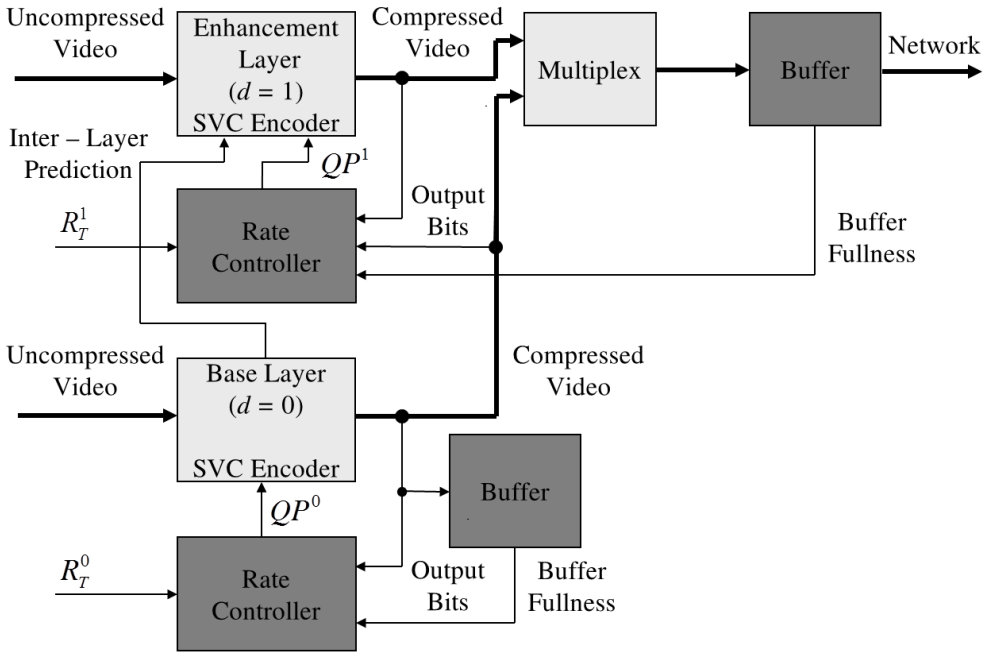


Figure 2: Block diagram of the proposed H.264/SVC RCA for two dependency layers.

5. Proposed Rate Control Algorithm

The RCA proposed for the optimized SVC encoder is depicted in dark gray in Figure 2. In particular, the SVC encoder is composed of two dependency layers, that is, a BL, which is identified by the layer identifier $d = 0$, and an EL with layer identifier $d = 1$. As shown, each layer contains a rate controller as well as an associated buffer. Notice that the inter-layer dependencies in SVC involve that the buffer at layer d must receive the sub-streams of layers 0 to d and, consequently, the corresponding TBR, R_T^d, must include that of the lower layer, R_T^{d-1}, and so on. This layered coding approach also entails that only the buffer corresponding to the highest dependency layer is real, since it is placed just before the network.

Every rate control module in the SVC encoder is organized in four levels: *intra period level*, *picture level*, *slice level*, and *MB level*. These levels are detailed in the following subsections making special emphasis on computational simplicity and support for parallelism. Nevertheless, since the main contributions of the proposed RCA are particularly focused on slice and MB levels, intra period and picture levels, which have already been studied ex-

tensively in the literature, are briefly described for the sake of conciseness, but appropriately referenced.

5.1. Intra Period Level

In video coding applications requiring very small buffer sizes, such as video conferencing, the preferred coding structure is IP...P with only the first picture as I-type. Notice that, since I pictures typically consume much more bit rate than P pictures, other coding patterns inserting I pictures periodically would dramatically increase the buffer overflow risk, unless the QP for those I pictures were properly increased to the detriment of the overall compressed video quality.

Given a time instant i, this level computes the amount $B^d_{R,i}$ of available bit budget to encode the remaining pictures in the intra period. From this amount, the number of total bits yielded by each picture is deducted (see [34] for details). In addition, the initial QP for the I picture, QP^d_I, is computed, for the BL ($d = 0$), by means of a simple lookup table specially designed for the proposed encoder. This lookup table is summarized in the following expression:

$$QP^d_I = 45 - (5 \cdot \Phi), \text{ if } 0.05 \cdot \Phi \leq Bpp^d < 0.05 \cdot (1 + \Phi), \tag{1}$$

being Φ a positive integer value, and Bpp^d the average number of target luma and chroma bits per pixel, i.e,

$$Bpp^d = \frac{R^d_T}{(Fr^d \cdot H^d \cdot W^d \cdot 1.5)}, \tag{2}$$

where Fr^d is the frame rate, H^d and W^d are the frame height and width, respectively, and 1.5 is a factor allowing for the chroma pixels in a 4:2:0 sampling format. For the EL ($d > 0$), two lookup tables, one for QS and the other for SS, are derived from the following two expressions that were empirically determined and reported in [73]:

$$QP^d_I = QP^{d-1}_I + \begin{cases} 0.26 - 3.91 \cdot \Delta R^d_T & \text{if QS} \\ 4.15 - 4.15 \cdot \ln(\Delta R^d_T) & \text{if SS,} \end{cases} \tag{3}$$

with

$$\Delta R^d_T = \frac{R^d_T}{R^{d-1}_T} - 1, \tag{4}$$

that is, the TBR increment between two consecutive layers.

5.2. Picture Level

In this level the amount T_i^d of target bits for the ith picture is estimated by means of a weighted combination of two bit allocation methods: one taking just a portion of $B_{R,i}^d$ according to the amount of remaining P pictures in the intra period, and the other watching over the current buffer status, V_i^d, for overflow and underflow prevention. Finally, T_i^d is upper and lower bounded to satisfy the HRD constraints. The buffer fullness is updated by means of the following expression:

$$V_i^d = V_{i-1}^d + AU_{i-1}^d - \frac{R_T^d}{Fr^d}, \tag{5}$$

being AU_{i-1}^d the amount of access unit output bits from layer 0 to d. The reader can be referred to [34] for more details about this frame bit allocation strategy.

5.3. Slice Level

In our proposal, an additional level is included in order to guarantee slice-level parallelism, that is, several threads, one per slice, encoding sections of a picture in parallel. Within this coding framework, the RCA should be able to assign just before encoding the picture a suitable amount of target bits per each slice. For this purpose, two different bit allocation strategies are proposed: one for the first I and P pictures, and the other for the remaining P pictures. The key reason behind this separation is due to the great impact on the buffer level when the first pictures in the sequence are encoded without knowing in advance their spatio-temporal complexities.

5.3.1. For the First I and P Pictures

Given that a very short buffer size is assumed in an ultra-low delay application, the paramount goal of the slice level for these pictures is to prevent buffer overflow and underflow, regardless of whether it may influence negatively on the reconstructed picture quality. For the *I picture*, the following four bit count thresholds for the buffer occupancy are defined:

- **Overflow threshold ($\mathbf{T_{OV}^d}$):** the number of bits required by the picture to reach a buffer level equal to 100% of the buffer size.
- **Upper threshold ($\mathbf{T_{UP}^d}$):** the number of bits required by the picture to reach a buffer level equal to 70% of the buffer size.

- **Lower threshold** ($\mathbf{T^d_{LW}}$)**:** the number of bits required by the picture to reach a buffer level equal to 20% of the buffer size.

- **Underflow threshold** ($\mathbf{T^d_{UN}}$)**:** the number of bits required by the picture to reach a buffer level equal to 0% of the buffer size.

The basic idea behind this threshold-based approach is to suitably regulate in the next level the MB QP, so that, once the picture has been encoded, the amount of total bits is neither greater than T^d_{UP} nor lower than T^d_{LW}. Otherwise, the MB QP should be changed more aggressively in order not to produce more bits than T^d_{OV} or less bits than T^d_{UN}.

Nevertheless, the frame partitioning into slices involves that each of these picture-level threshold values must be split into several parts, as many as the number N^d_{SL} of slices per picture in the dependency layer d. In particular, for the jth slice in the picture, the following set of thresholds is defined:

$$\left(T^d_{OV,j}, T^d_{UP,j}, T^d_{LW,j}, T^d_{UN,j}\right) = \frac{1}{N^d_{SL}} \cdot \left(T^d_{OV}, T^d_{UP}, T^d_{LW}, T^d_{UN}\right). \tag{6}$$

Notice that, although a more fair bit distribution could be performed by, for example, using some spatial activity measurement for predicting the slice encoding complexity, a low-complexity bit allocation approach is pursued for the proposed the video coding system as already remarked before.

For the *first P picture*, the bit range between T^d_{UP} and T^d_{LW} is reduced around the amount of bits needed to reach a target buffer level (stated in [34]) in order to achieve a stricter buffer control. It is important to notice that the QP range to be used for the prior I picture may not be suitable for the current one, since only buffer-based decisions are carried out without considering the temporal activity of the scene [73].

Next, each picture-level threshold value is split into N^d_{SL} portions, but, in this case, using the coding complexity $C^d_{I,j}$ corresponding to each jth slice in the already encoded I picture, that is,

$$\left(T^d_{OV,j}, T^d_{UP,j}, T^d_{LW,j}, T^d_{UN,j}\right) = \frac{C^d_{I,j}}{\sum_{u=0}^{N^d_{SL}-1} C^d_{I,u}} \cdot \left(T^d_{OW}, T^d_{UP}, T^d_{LW}, T^d_{UN}\right). \tag{7}$$

Specifically, for the sake of simplicity, the slice coding complexity is measured similarly to [54] in terms of sum of product $TotalBits \times Q$ of all MBs in the slice, being Q the quantization step associated with a certain QP.

5.3.2. For the Remaining P Pictures

In this case, the amount $T^d_{i,j}$ of target bits for the jth slice in the ith picture is computed as:

$$T^d_{i,j} = \frac{\widetilde{C}^d_{i,j}}{\sum_{u=0}^{N^d_{SL}-1} \widetilde{C}^d_{i,u}} \cdot T^d_i, \tag{8}$$

where $\widetilde{C}^d_{i,j}$ stands for a prediction of the slice coding complexity. More specifically, $\widetilde{C}^d_{i,j}$ is updated frame by frame via exponential average, with a *forgetting factor* (FF) set to 0.25, of the coding complexities corresponding to co-located slices in previous pictures. This FF value allows for reducing high fluctuations in the coding complexity prediction.

5.4. Macroblock Level

This level focuses on estimating an appropriate MB QP in order to comply with the bit budget constraints above specified. As in slice level, two different strategies are also employed and described bellow.

5.4.1. For the First I and P Pictures

Three operation steps are followed before encoding a kth MB in the jth slice corresponding to the ith picture; specifically, they are summarized next:

1. Predict the amount $\widetilde{B}_{i,j}$ of total bits required by the slice once the $(k-1)$th MB has been encoded.

2. Compare $\widetilde{B}_{i,j}$ to those thresholds specified in Eqs. (6) and (7).

3. Modify the MB QP, $QP^d_{i,j,k}$, accordingly.

More in detail, Algorithm 1 describes the proposed MB-level QP estimation approach. In this algorithm $N_{R,MB}$ denotes the number of remaining MBs in the slice, and $B^d_{i,j,u}$ the amount of total bits consumed by the uth MB in the slice. Notice that the prediction $\widetilde{B}_{i,j}$ is also compared to the previous one, $\widetilde{B}_{i,j,prev}$, so that $QP^d_{i,j,k}$ can only be modified when necessary, that is, when $\widetilde{B}_{i,j}$ is still too high for the current QP, thus providing smooth QP variation within the slice.

Algorithm 1 QP estimation procedure for the first I and P pictures.

if $k = 0$ **then** {first MB?}
  $QP^d_{i,j,k} \leftarrow QP^d_I$
else
  $\widetilde{B}_{i,j} \leftarrow \left(1 + \frac{N_{R,MB}}{k}\right) \cdot \sum_{u=0}^{k-1} B^d_{i,j,u}$ {prediction}
  if $\left(\widetilde{B}_{i,j} \geq T^d_{UP,j}\right) \wedge \left(\widetilde{B}_{i,j} \geq \widetilde{B}_{i,j,prev}\right)$ **then**
    $QP^d_{i,j,k} \leftarrow QP^d_{i,j,k-1} + 1 + (\text{P picture? } 1 : 0)$
  else if $\left(\widetilde{B}_{i,j} \leq T^d_{LW,j}\right) \wedge \left(\widetilde{B}_{i,j} \leq \widetilde{B}_{i,j,prev}\right)$ **then**
    $QP^d_{i,j,k} \leftarrow QP^d_{i,j,k-1} - 1$
  else if $\widetilde{B}_{i,j} \geq T^d_{OV,j}$ **then**
    $QP^d_{i,j,k} \leftarrow QP^d_{i,j,k-1} + 2 + (\text{P picture? } 1 : 0)$
  else if $\widetilde{B}_{i,j} \leq T^d_{UN,j}$ **then**
    $QP^d_{i,j,k} \leftarrow QP^d_{i,j,k-1} - 1$
  else
    $QP^d_{i,j,k} \leftarrow QP^d_{i,j,k-1}$
  end if
  $\widetilde{B}_{i,j,prev} \leftarrow \widetilde{B}_{i,j}$
end if

5.4.2. For the Remaining P Pictures

The amount $T^d_{i,j,k}$ of target bits to encode the current kth MB in the jth slice corresponding to the ith picture is computed as

$$T^d_{i,j,k} = \frac{\widetilde{C}^d_{i,j,k}}{\sum_{u=k}^{N^d_{MB}-1} \widetilde{C}^d_{i,j,u}} \cdot T^d_{R,i,j}, \tag{9}$$

where $\widetilde{C}^d_{i,j,k}$ is a prediction of the MB coding complexity via exponential average ($FF = 0.25$) of those corresponding to co-located MBs in previous pictures, N^d_{MB} is the number of MBs in the current slice, and $T^d_{R,i,j}$ is the amount of available target bits to encode the remaining MBs in the slice.

Afterwards, based on the study of R-D modeling for video coding in [30], $QP^d_{i,j,k}$ is computed by means of the following simple linear R-Q function:

$$T^d_{i,j,k} = \frac{\widetilde{X}^d_{i,j,k}}{Q^d_{i,j,k}} + \widetilde{H}^d_{i,j,k}, \tag{10}$$

where $Q^d_{i,j,k}$ is the quantization step associated with $QP^d_{i,j,k}$, and $\widetilde{X}^d_{i,j,k}$ and $\widetilde{H}^d_{i,j,k}$ are, respectively, a prediction of the complexity to encode the MB transform coefficients (in terms of product *CoeffBits*×Q) and a prediction of the amount of header bits. Both predictors are also updated via exponential average ($FF = 0.25$) of those corresponding to co-located MBs in previous pictures.

Finally, to ensure quality consistency within the slice and also between slices, $QP^d_{i,j,k}$ is bounded ± 1 unit respect to that of the preceding MB and ± 4 units respect to the average QP of the previous picture. However, for the first MB in the slice, the QP is set to the average QP of the co-located slice.

6. Experiments and Results

In this section we present the experimental results of the proposed parallel SVC encoder. First, we present the experimental methodology; then, we show the performance results using constant QP encoding for determining the optimal encoding configuration; and finally, we present the complete results using parallel processing and rate control.

6.1. Experimental Setup

The parallel SVC encoder has been implemented on top of a baseline single-threaded H.264/SVC encoder belonging to Fraunhofer HHI. This baseline encoder already includes SIMD optimizations using SSE2 instructions [74] for the most time consuming kernels such as distortion functions (SSE, SAD), inverse and direct transforms, quantization, interpolation filters, deblocking filter, spatial upsample filter and memory copy operations. However, additional tools had to be implemented in this baseline version in order to have the parallel encoder available for our experimental purposes, specifically: multi-threading using *POSIX threads* (Pthreads) and parallel processing for slice encoding, upsample filters and interpolation filters as already described in Section 4, and the RCA described in Section 5. For improving the reproducibility of the experiments, threads have been pined to cores using the *numactl* tool, and each experiment has been executed five times and average time is reported. Henceforth, we will refer to this parallel encoder as *HhiSvcEnc* and its configuration with a single thread as *sequential mode*. HhiSvcEnc in sequential mode will be used as reference for finding a suitable parallel configuration able to provide real-time execution while minimizing the R-D losses due to the use of slices.

Option	Value
P Macroblock modes	8×8 and SKIP
Intra frames	First in sequence
QP for BL (QP-BL)	22, 27, 32, 37
QP for EL/QS	QP-BL - 4
QP for EL/SS	QP-BL
Motion estimation algorithm	diamond search
Search range	16
Entropy coding	CAVLC
Deblocking filter	enabled in non-cross slice borders mode
Adaptive prediction	Adaptive inter-layer prediction

Table 1: Coding options

HhiSvcEnc has been configured for ultra-low delay video conferencing applications with: 2 dependency layers (both in spatial and quality scalability modes), 1 temporal layer, IP..P pattern, only 8×8 inter-prediction for P pictures, diamond shaped motion estimation with search range of 16×16 samples, adaptive residual prediction, no adaptive inter-layer prediction, no adaptive motion vector prediction, R-D optimization, and *context adaptive variable length coding* (CAVLC). For fixed QP experiments, the BL was encoded with the QP values recommended in [75], specifically: 22, 27, 32, and 37. The same values were used for SS in both layers and, for the EL in QS, we used the base layer QP minus 4 units as suggested in [76]. Table 1 summarizes the encoder configuration.

The system employed to measure performance includes an 8 core Intel Xeon E5-2687W processor running at 3.10GHz. Simultaneous Multithreading (SMT, aka Hyperthreading) and dynamic overclocking (TurboBoost) were disabled for improving reproducibility. More details about the hardware and software are listed in Table 2.

A total of six 10s 720p@60fps test sequences suitable for video conferencing were selected [75]: *FourPeople*, *Johnny*, *KristenAndSara*, *Vidyo1*, *Vidyo3*, and *Vidyo4*. However, for our experiments, these video sequences were converted to 720p@30fps and 1080p@30fps, this latter to allow for SS. The SVC normative upsampling method based on a set of 4-taps filters was used.

6.2. Profiling of the Sequential Mode

A profiling analysis was conducted to determine the most time consuming parts of HhiSvcEnc and based on that guide the parallelization parameters.

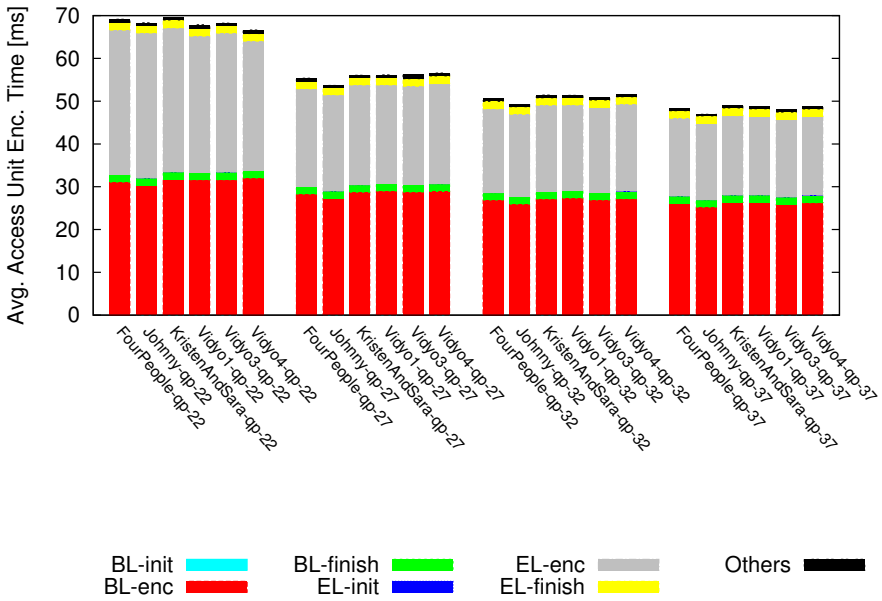
System		Software	
Processor:	Intel Xeon E5-2687W	SVC encoder:	HhiSvcEnc
Architecture:	Sandy Bridge	Compiler:	gcc 4.8.1
Cores:	8	Opt. level:	-O3
Frequency:	3.1GHz	OS:	Ubuntu Linux 13.10
L3 cache:	20MB	Kernel:	3.11.0-15
SMT:	disabled		
TurboBoost:	disabled		

Table 2: Experimental Setup

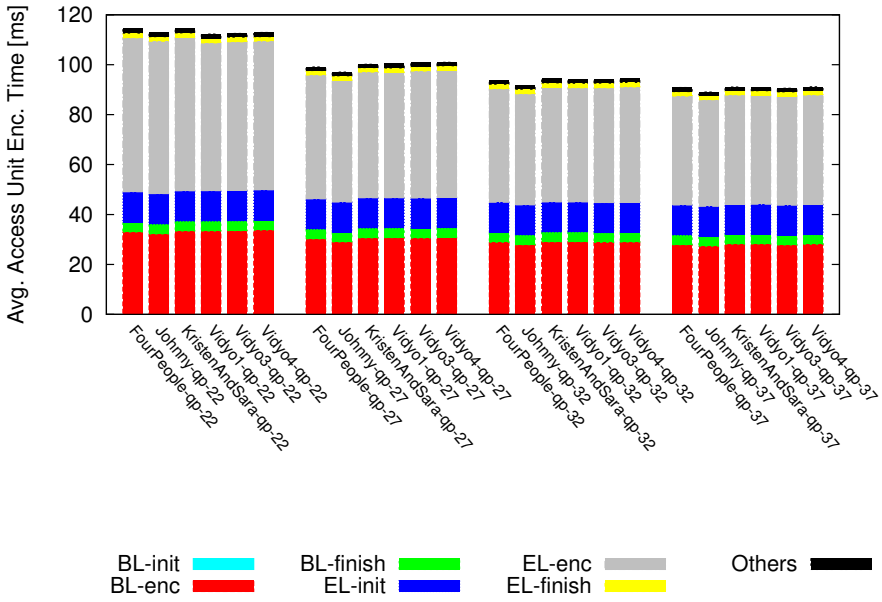
Figures 3a and 3b show the execution time profile, in terms of average access unit encoding time, for different videos at different QPs for QS and SS and 1 slice per layer, respectively. The total execution time has been divided into the following seven parts:

- **BL-init:** before encoding the BL: initialization.
- **BL-enc:** encoding of slices for the BL.
- **BL-finish:** after encoding the BL: padding and interpolation filtering.
- **EL-init:** before encoding the EL: initialization and upsampling for SS.
- **EL-enc:** encoding of slices for the EL.
- **EL-finish:** after encoding the EL: padding and interpolation filtering.
- **Others:** other non-parallel support tasks.

The profiling results show that, as expected, most of the execution time goes into slice encoding (*BL-enc* and *EL-enc*). For QS, 50.6% and 42.0% are spent on encoding BL and EL, respectively. For SS, the values are 30.3% and 50.2%, respectively. *EL-init*, which includes the upampling filters for SS, also takes an important part of the execution time (12.2% in average), whereas the time consumed by *BL-init* is negligible compared to the remaining parts. The *finish* section in both QS and SS, which includes interpolation filters, consume 3.2% and 1.8% of the execution time for QS and SS, respectively. Other parts of the video encoder that do not require parallel processing consume in average only 1.5% of the execution time.



(a) Quality Scalability (QS)



(b) Spatial Scalability (SS)

Figure 3: Execution time profile using 1 slice per layer in sequential mode.

6.2.1. Sequential Performance at Fixed QP

In order to estimate the acceleration required by parallel processing, we executed all the input sequences in sequential mode for four QPs and 1 slice per layer, for both QS and SS.

Tables 3 and 4 show the resulting PSNR, bit rate, average access unit encoding time, and encoding frame rate for QS and SS, respectively. When using only one thread, the tested encoding system is not capable of achieving real-time operation for any of the configurations: For QS, the minimum required speedup to reach 30fps should be between 1.45 (QP 22) to 2.05 (QP 37) and, for SS, between 2.72 (QP 22) to 3.40 (QP 37). Appropriate parallelization parameters have to be found to be able to provide the required performance.

In order to give a better understanding of the performance of HhiSvcEnc, the simulation was repeated with the Joint Scalable Video Model (JSVM) 9.19.15 [77] software. For these encodings, the following coding options were chosen: SearchMode = 4, SearchRange = 16, for the general configuration file as well as: SymbolMode = 0, MaxDeltaQP = 0, MinLevelIdc = 51, MCBlocksLT8x8Disable = 1, DisableBSlices = 1 for layer 0 and additionally ILModePred = 1, ILMotionPred = 1, ILResidualPred = 2 for the EL. All other options were left at their defaults. Table 5 shows the difference of HhiSvcEnc to JSVM 9.19 in terms of the *Bjontegaard delta bit rate* (BDBR) [78] and the relative speedup.

6.2.2. Parallel Performance at Fixed QP

Because the main parallelization strategy is based on slice-level parallelism, it is then necessary to select an appropriate configuration that can provide the required speedup and minimizes the encoding losses due to the introduction of multiple slices.

In order to select the best option, multiple slice configurations were tested and executed with parallel processing enabled. The number of threads used in each configuration was set to the highest number of slices in any layer. For example, for a configuration labeled 6-8, which corresponds to 6 slices in BL and 8 slices in EL, 8 threads were used. For QS, the configurations have the same number of slices in each layer (because they have the same resolution) and, for SS, they have more slices in the EL (which has a higher resolution).

Figures 4a and 4b show the average access unit processing time for different slice configurations and for different EL bit rates. Configurations with processing time below the horizontal line of 30fps (33 ms per access unit) can

QP	Video	Y-PSNR [dB]		Bit Rate [Mbps]		Enc. Time	Frame Rate
		BL	EL	BL	EL	[ms]/AU	[fps]
22	*FourPeople*	42.16	44.54	3.86	19.04	69.15	14.46
	Johnny	42.56	44.45	3.30	18.87	68.31	14.64
	K&S	42.86	44.75	3.98	17.64	69.47	14.39
	Vidyo1	42.86	44.91	3.90	15.03	67.84	14.74
	Vidyo3	42.96	44.70	5.21	17.16	68.24	14.65
	Vidyo4	42.82	45.26	5.03	16.48	66.62	15.01
	AVG	42.70	44.77	4.21	17.37	68.28	14.65
27	*FourPeople*	39.66	42.29	1.70	5.19	55.37	18.06
	Johnny	40.20	42.65	0.98	4.72	53.68	18.63
	K&S	40.31	42.97	1.57	5.20	56.06	17.84
	Vidyo1	40.06	43.11	1.56	5.06	56.13	17.81
	Vidyo3	40.03	43.07	1.89	6.24	56.18	17.80
	Vidyo4	39.91	42.99	1.74	6.22	56.57	17.68
	AVG	40.03	42.85	1.57	5.44	55.67	17.97
32	*FourPeople*	36.60	39.70	0.90	2.46	50.57	19.77
	Johnny	37.41	40.22	0.45	2.05	49.17	20.34
	K&S	37.25	40.40	0.78	2.50	51.43	19.44
	Vidyo1	37.04	40.32	0.79	2.41	51.35	19.48
	Vidyo3	36.83	40.26	0.80	2.71	50.84	19.67
	Vidyo4	36.88	40.09	0.80	2.78	51.51	19.41
	AVG	37.00	40.17	0.75	2.49	50.81	19.69
37	*FourPeople*	33.34	36.34	0.51	1.49	48.22	20.74
	Johnny	34.49	37.17	0.24	1.26	46.94	21.30
	K&S	34.01	37.17	0.44	1.51	48.89	20.46
	Vidyo1	34.06	37.05	0.46	1.43	48.69	20.54
	Vidyo3	33.62	36.88	0.42	1.42	47.93	20.86
	Vidyo4	33.82	36.84	0.46	1.50	48.70	20.53
	AVG	33.89	36.91	0.42	1.43	48.23	20.74

Table 3: Performance of the sequential mode and 1 slice per layer for QS. *K&S* refers to *KristenAndSara*.

QP	Video	Y-PSNR [dB]		Bit Rate [Mbps]		Enc. Time	Frame Rate
		BL	EL	BL	EL	[ms]/AU	[fps]
22	*FourPeople*	42.16	43.58	3.86	13.15	114.40	8.74
	Johnny	42.56	43.82	3.30	12.63	112.80	8.87
	K&S	42.86	44.19	3.98	12.65	114.34	8.75
	Vidyo1	42.86	44.40	3.90	12.01	112.30	8.90
	Vidyo3	42.96	44.25	5.21	14.96	112.53	8.89
	Vidyo4	42.82	44.41	5.03	14.52	112.96	8.85
	AVG	42.70	44.11	4.21	13.32	113.22	8.83
27	*FourPeople*	39.66	41.47	1.70	4.94	99.14	10.09
	Johnny	40.20	42.01	0.98	4.21	97.06	10.30
	K&S	40.31	42.20	1.57	4.99	100.22	9.98
	Vidyo1	40.06	42.06	1.56	4.97	100.34	9.97
	Vidyo3	40.03	41.82	1.89	6.42	100.85	9.92
	Vidyo4	39.91	41.90	1.74	5.86	100.98	9.90
	AVG	40.03	41.91	1.57	5.23	99.77	10.03
32	*FourPeople*	36.60	38.48	0.90	2.64	93.59	10.68
	Johnny	37.41	39.21	0.45	2.11	91.64	10.91
	K&S	37.25	39.17	0.78	2.66	94.33	10.60
	Vidyo1	37.04	38.77	0.79	2.59	94.09	10.63
	Vidyo3	36.83	38.59	0.80	2.97	94.09	10.63
	Vidyo4	36.88	38.62	0.80	2.93	94.43	10.59
	AVG	37.00	38.81	0.75	2.65	93.70	10.67
37	*FourPeople*	33.34	35.18	0.51	1.82	90.88	11.00
	Johnny	34.49	36.21	0.24	1.50	89.06	11.23
	K&S	34.01	35.94	0.44	1.85	91.11	10.98
	Vidyo1	34.06	35.72	0.46	1.76	90.86	11.01
	Vidyo3	33.62	35.43	0.42	1.83	90.61	11.04
	Vidyo4	33.82	35.46	0.46	1.91	91.18	10.97
	AVG	33.89	35.66	0.42	1.78	90.62	11.04

Table 4: Performance of the sequential mode and 1 slice per layer for SS. *K&S* refers to *KristenAndSara*.

Video	QS				SS			
	BDBR [%]		Speedup		BDBR [%]		Speedup	
	BL	EL	AVG	DEV	BL	EL	AVG	DEV
FourPeople	41.6	20.0	12.53	1.32	41.6	13.2	14.79	0.51
Johnny	52.4	26.1	12.42	1.29	52.4	14.5	14.59	0.58
K&S	58.0	27.3	12.34	1.24	58.0	26.5	14.73	0.58
Vidyo1	55.6	22.6	12.63	1.38	55.6	19.2	14.67	0.52
Vidyo3	39.3	14.8	12.50	1.29	39.3	13.9	14.76	0.60
Vidyo4	55.9	23.6	12.47	1.12	55.9	24.5	14.78	0.51
AVG	50.5	22.4	12.48	1.27	50.5	18.6	14.72	0.55

Table 5: BDBR and speedup (average and standard deviation over QPs) of HhiSvcEnc in sequential mode referred to JSVM 9.19. 1 slice per layer is used in both encoders.

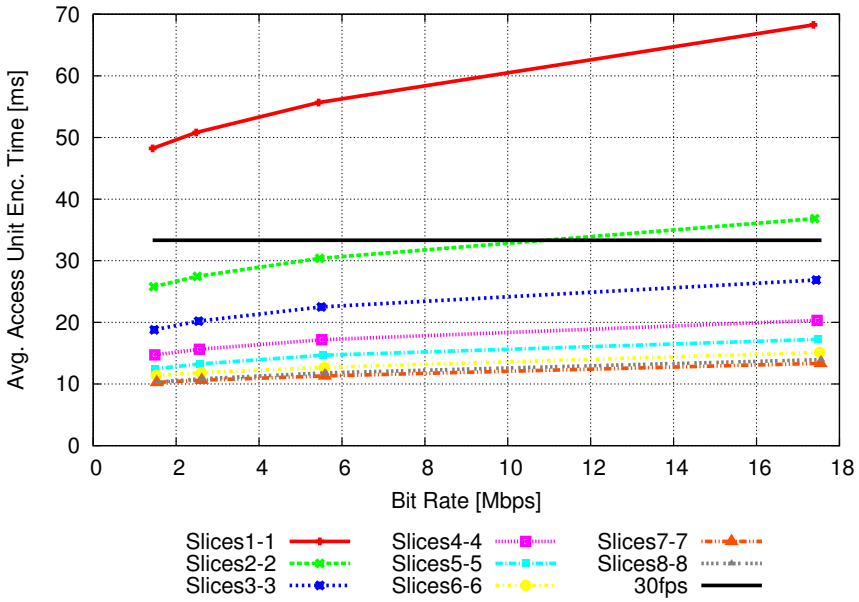
be processed in real time. As expected, the processing time decreases with the increase of the number slices and threads. The maximum processing time performance achieved is 97fps for an average bit rate of 1.54Mbps for QS, and 55fps at 1.87Mbps for SS.

Given a sufficient amount of slices, all configurations can achieve real-time operation even at high bit rates. Having a lot of slices per layer is not desirable, however, due to the negative impact on the R-D performance. Figures 5a and 5b show the BDBR losses for different slice configurations compared to 1 slice per layer.

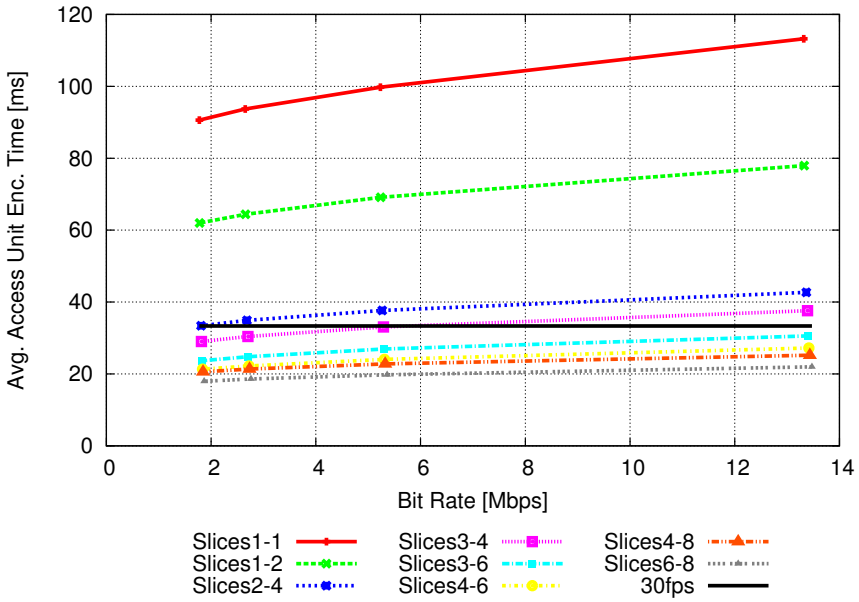
By taking into account both the execution time and the results in terms of R-D performance, the following configurations, which will be used in the next section for evaluating the RCA, achieve the desired frame and bit rates and minimize the encoding losses:

- For QS, a **3-3** configuration (3 slices in each layer) results in encoding losses less than 4% and bit rates up to 20Mbps.
- For SS, a **3-6** configuration (3 slices in the BL and 6 slices in the EL) results in encoding losses less than 4% and bit rates up to 14Mbps.

The average execution time reduction by using these proposed slice configurations together with parallel execution can be shown in Figures 6a and 6b. As expected, the processing time of the parallel tasks have been considerably reduced whereas that of the sequential tasks remains unaltered.

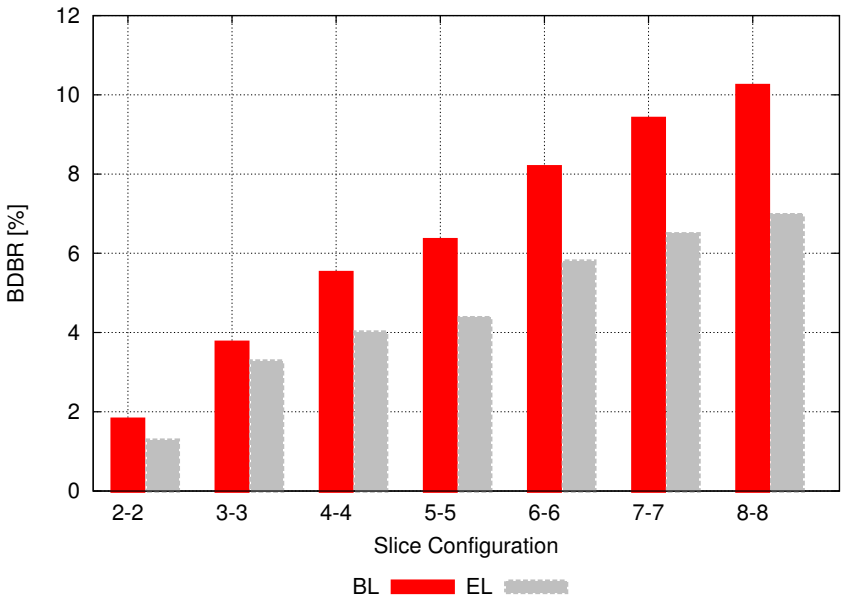


(a) Quality Scalability (QS)

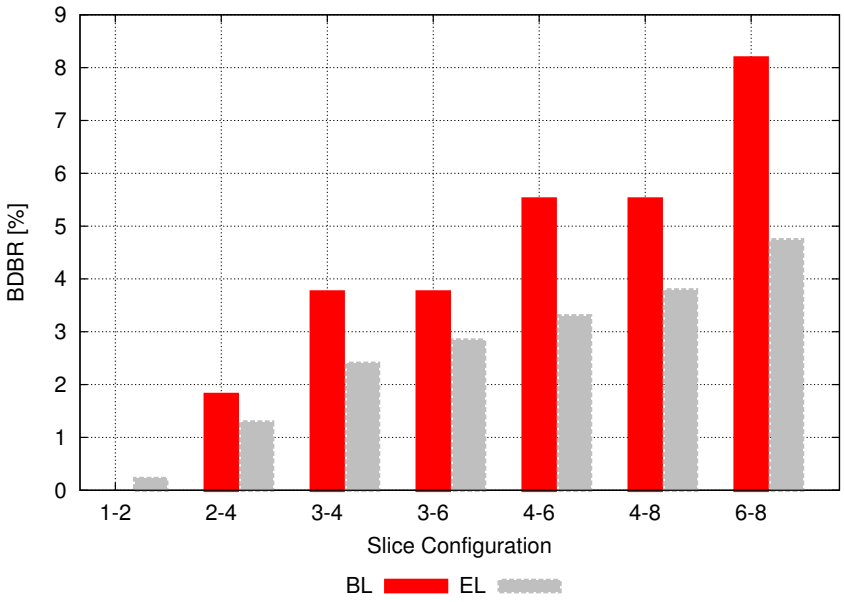


(b) Spatial Scalability (SS)

Figure 4: Average access unit encoding time for EL different bit rates and slice configurations.

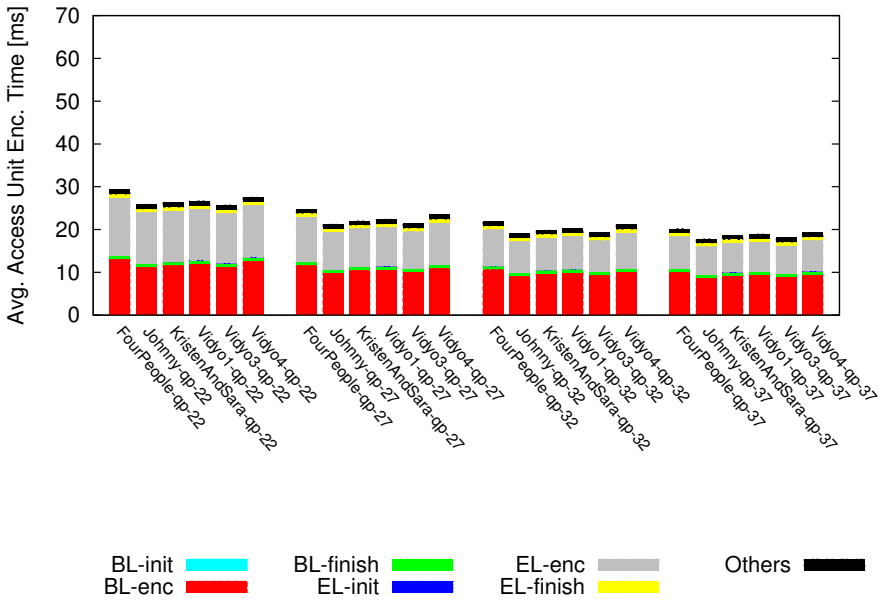


(a) QS

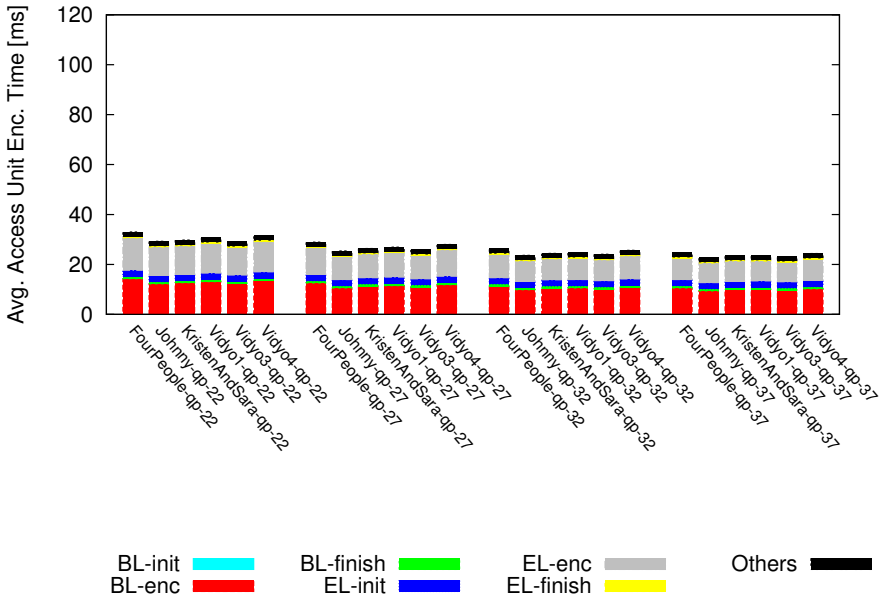


(b) SS

Figure 5: BDBR for different slice configurations.



(a) QS



(b) SS

Figure 6: Execution time profiling using, for QS, 3-3 slices and, for SS, 3-6 slices.

Scal./Slices	QP	BL-enc	BL-finish	EL-init	EL-enc	EL-finish	Total
QS/3-3	22	2.60	2.39	0.25	2.65	2.40	2.55
	27	2.66	2.36	0.25	2.46	2.36	2.48
	32	2.72	2.36	0.25	2.52	2.34	2.52
	37	2.77	2.36	1.00	2.60	2.36	2.57
	AVG	2.69	2.37	0.44	2.56	2.36	2.53
SS/3-6	22	2.55	4.38	4.89	5.10	3.64	3.70
	27	2.65	4.39	4.86	5.06	3.60	3.70
	32	2.75	4.36	4.83	5.25	3.60	3.78
	37	2.80	4.36	4.84	5.37	3.62	3.83
	AVG	2.69	4.37	4.86	5.19	3.62	3.75

Table 6: Speedup with parallel execution for different encoder phases.

Table 6 shows the average speedups of the parallel executions for all videos and all QPs for the selected slices configurations. The total average speedups are **2.53** and **3.75** for QS and SS, respectively. The parallelization efficiency for the slice encoding phase is always higher than 85%, the difference from the ideal efficiency is due to a load balance issue that comes from the static slice size allocation algorithm. The efficiencies for the upsampling (*EL-init*) is 81% and, for the interpolation filtering (*BL-finish* and *EL-finish*), is 80% and 66% for QS ans SS respectively. It should be noted that in the *init* and finish phases there is also a sequential part which makes the parallelization efficiency lower. In fact, the granularity of the parallelization of the upsampling and interpolation filtering is smaller than the encoding phase reducing the possibilities of load imbalance.

6.3. Rate Control Results

In this subsection, the performance of HhiSvcEnc from the rate control point of view is presented and discussed. First, a description of the RCAs to be compared, the set of TBRs, and the buffer size configuration are given; and second, the experimental results are shown and analysed.

6.3.1. Experimental Setup

In order to assess the performance of the RCA described in Section 5 (**Prop**), it was compared to sequential encoding at fixed QP (**SeqFixQP**), which is a reference from the R-D efficiency point of view, as well as to two variants of the proposed rate controller, which are described next:

Algorithm	Description	Pros	Cons
SeqFixQP	1-1 slices Sequential execution	Good R-D efficiency	Non real-time encoding
Ref1	MB-level RCA 1-1 slices Sequential execution	Good R_T^d adjustment	Non real-time encoding
Ref2	Picture-level RCA 3-3/6 slices (QS/SS) Parallel execution	Real-time encoding	Bad R_T^d adjustment
Prop	MB-level RCA 3-3/6 slices (QS/SS) Parallel execution	Good R_T^d adjustment Real-time encoding	

Table 7: Characteristics of the assessed algorithms.

- **Ref1:** QP decisions are made at MB level, with 1 slice per layer, and sequential execution, that is, 1 thread for the whole encoding process.
- **Ref2:** QP decisions are made at picture level, that is, slice and MB levels are disabled, with the predefined number of slices per layer, and parallel execution. For this picture-level RCA, the same linear R-Q model as that in Eq. (10) is used and, then, the result is bounded ± 1 unit respect to the preceding frame QP.

Each of these reference RCAs should provide a strong point but also a weak point, and our proposal should bring together the strength of each reference RCA. Expected pros and cons of the assessed algorithms are summarized in Table 7.

The output bit rates generated by SeqFixQP encoding (see Tables 3 and 4 for QS and SS, respectively) were used as TBRs for the assessed RCAs. The experimental results for a **buffer size** per layer set to **50ms** are given in the following subsection.

6.3.2. Results and Discussion

BDBR was the measurement used to compare the RCAs from the R-D efficiency point of view. Table 8 reports the BDBR results of the assessed RCAs referred to SeqFixQP encoding for the two types of scalability (notice that the results for BS-QS and BS-SS do match, since the same number

Video	BDBR [%]								
	BL			EL-QS			EL-SS		
	Ref1	Ref2	Prop	Ref1	Ref2	Prop	Ref1	Ref2	Prop
FourPeople	24.8	15.8	17.3	15.4	13.4	16.3	17.9	16.0	19.3
Johnny	49.4	33.8	52.1	24.8	19.1	25.8	28.1	21.7	29.8
K&S	23.5	14.6	25.3	15.7	11.0	17.1	19.3	15.0	19.9
Vidyo1	19.7	4.5	16.6	7.6	5.0	9.3	6.4	4.6	8.3
Vidyo3	10.7	6.7	9.9	11.2	6.7	12.0	10.5	6.5	11.3
Vidyo4	10.7	9.5	16.4	12.2	7.7	11.7	10.6	8.4	11.3
AVG	23.1	14.1	22.9	14.5	10.5	15.4	15.5	12.0	16.6

Table 8: BDBR of the assessed RCAs referred to SeqFixQP encoding for QS and SS. *K&S* refers to *KristenAndSara*.

of slices was employed). The assessed RCAs produce in some cases noticeable R-D losses compared to SeqFixQP encoding owing to the use of several slices per layer and the overhead increase for encoding the QP information. The proposed algorithm produces BDBR results quite similar to those of the Ref1 algorithm, thus proving that the additional slice level described in Subsection 5.3, which is required to achieve real-time operation, is able to properly manage the frame bit budget among slices. When compared to the Ref2 algorithm, our proposal generates some R-D losses, as expected, since picture-level RCAs generally provide better quality than MB-level RCAs for a given TBR [62].

Table 9 illustrates the rate control performance in terms of TBR adjustment, by measuring the bit rate error respect to SeqFixQP encoding, and buffer control, by measuring the average/maximum percentages in which overflows (#O) or underflows (#U) occurred in one encoding. As can be shown, the three RCAs achieve bit rate errors very close to 0%, as expected, since a short-term TBR adjustment is pursued for low-latency video conferencing. The experimental results in terms of #O and #U reveal that the two assessed MB-level RCAs are capable of considerably reducing the overflow and underflow occurrences, since this kind of algorithms generally guarantees tighter buffer control when compared to picture-level RCAs like the Ref2 algorithm. This latter characteristic of the MB-level RCAs is endorsed in Figure 7 depicting some representative buffer occupancy time evolutions provided by the assessed RCAs for the sequences *Vidyo1* (QS) and *Kriste-*

Layer	Algorithm	Rate Error [%]	#O [%]	#U [%]
BL	Ref1	0.03/0.10	0.21/2.33	0.21/1.67
	Ref2	0.18/0.85	10.54/34.00	11.13/23.00
	Prop	0.03/0.08	0.17/1.00	0.51/4.00
EL/QS	Ref1	0.00/0.01	0.00/0.00	0.68/3.33
	Ref2	0.07/0.33	0.35/5.33	12.38/24.00
	Prop	0.00/0.00	0.00/0.00	0.56/3.00
EL/SS	Ref1	0.00/0.01	0.00/0.00	1.36/4.67
	Ref2	0.08/0.59	1.15/8.67	9.83/15.33
	Prop	0.00/0.01	0.00/0.00	1.15/5.33

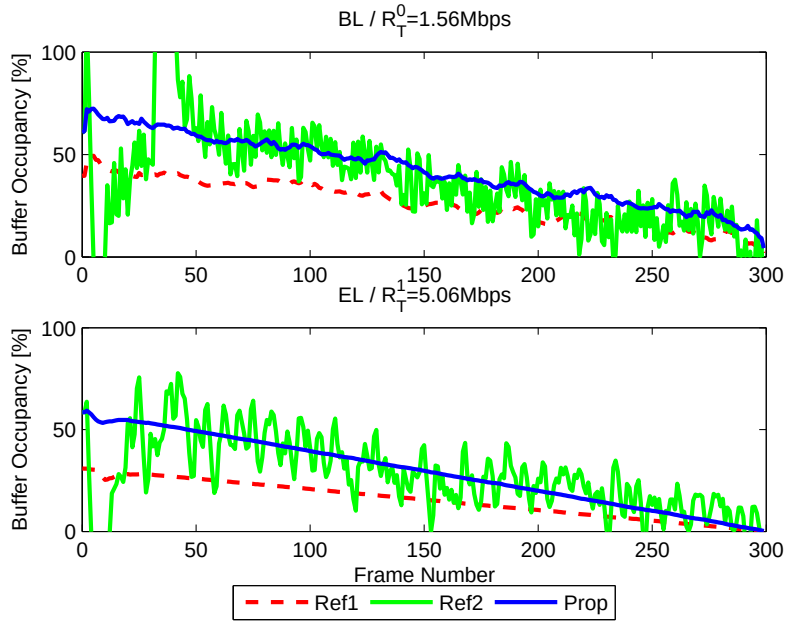
Table 9: Rate error, #O, and #U of the assessed RCAs. Average/maximum results are presented.

nAndSara (SS). It is also worth mentioning from Table 9 that unavoidable overflows happened at the beginning of the encoding process of some video sequences encoded at low TBRs, since, for the stated video encoder, 50ms as buffer size is too short for the high bit rate consumed by the I picture even encoded with the maximum allowed QP value, thus explaining the non-zero #O values obtained by both MB-level RCAs at BL.

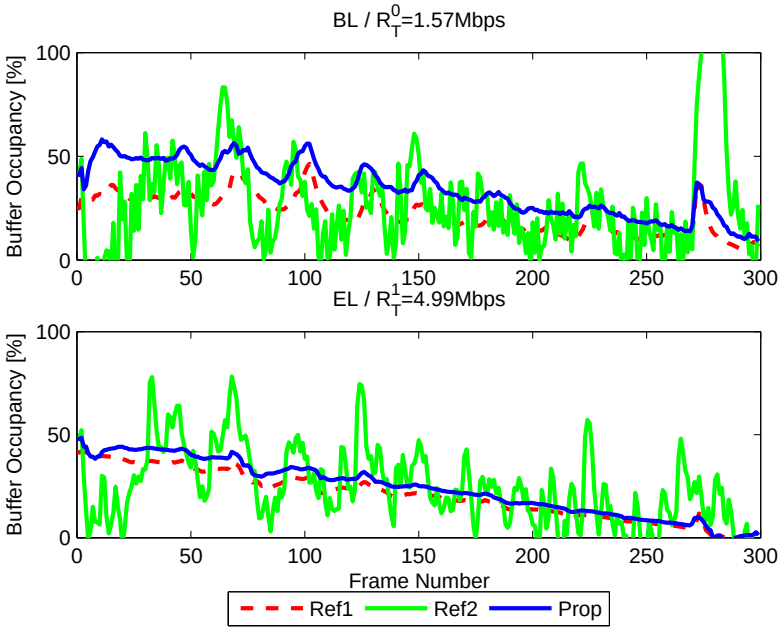
Finally, Figures 8a and 8b show, for QS and SS respectively, the average access unit encoding time when using the proposed parallel-friendly MB-level RCA for different EL bit rates and video sequences. As can be observed, since all the rate points except one (that corresponding to *FourPeople* encoded using SS at the highest specified TBR) are below the horizontal line of 30fps, we can conclude that the proposed SVC encoder with the RCA enabled is able to operate in real time over a wide range of TBRs, thus proving that the time consumed by the rate control operation is negligible with respect to other parts of the encoding process.

7. Conclusions and Future Work

In this paper a parallel SVC encoder for HD video conferencing has been proposed. Our technical contributions are focused on certain parallel support tasks of the video encoder, such as the encoding process itself, upsampling and interpolation filtering, as well as on the rate control process. In particular, some multithreading-based encoding strategies for parallel execution

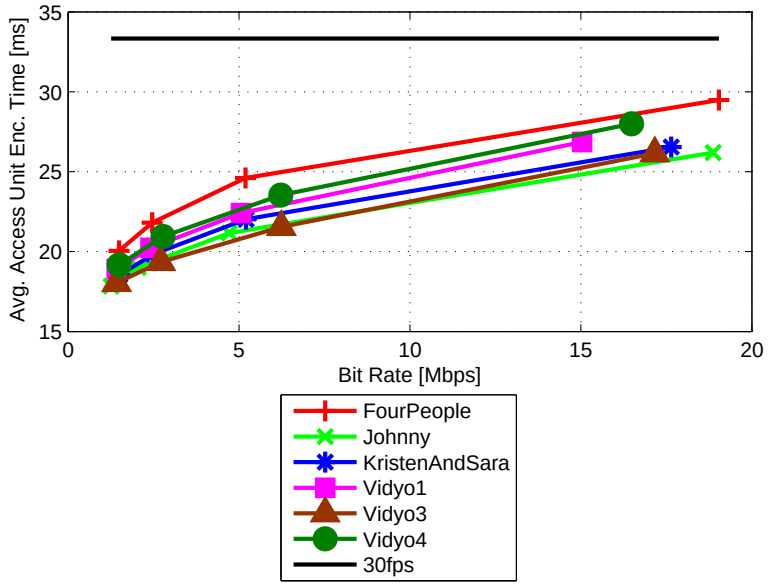


(a) QS, *Vidyo1*

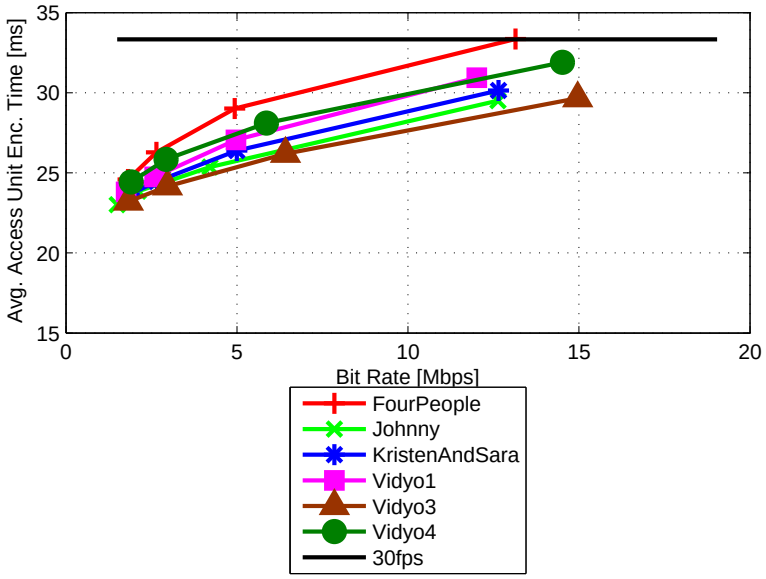


(b) SS, *KristenAndSara*

Figure 7: Comparison of buffer occupancy evolutions corresponding to the assessed RCAs.



(a) QS



(b) SS

Figure 8: Average access unit encoding time with the proposed RCA for different EL bit rates and video sequences.

have been jointly used in this work in order to reduce the encoding time per access unit. Furthermore, taking into account the need of slices for achieving real-time SVC of HD video content, a parallel-friendly RCA has also been proposed, in which an additional so-called slice level, our main contribution for this encoder block, is deployed for a proper bit budget management among slices within a frame. This RCA operating at MB level for QP regulation is also capable of ensuring short-term bit rate adaptation without exceeding the buffer limits, which are quite restrictive in an ultra-low delay application scenario. Such a tight buffer control has motivated the use of a simple but efficient MB QP regulation approach within those pictures consuming a lot of bit rate, thus being another remarkable contribution.

As future work, the following three improvements could be applied in the proposed framework:

- Fixed-size slices within a layer are currently employed, thus resulting in potential load imbalance if some sections of the frame are computationally more complex than others. A dynamic selection of the slice size could improve the speedup in some cases.

- A dynamic selection of the number of slices per layer, which is currently prefixed to 3-3 for QS and 3-6 for SS, can be another work line. Depending on the target processor, a dynamic algorithm can set the slice parameters in order to guarantee real-time performance and, at the same time, minimizing the R-D losses due to slices.

- The idea of analysing the input video content for region of interest detection could also be investigated to improve the slice distribution as well as the visual quality.

- An adaptive mode decision algorithm as mentioned in Section 2.3 could be implemented to allow for more freedom in the slice structure and achieve better trade offs between RD-performance, encoding speed and latency.

- A similar RCA can be integrated into a scalable HEVC encoder with the corresponding adaptations to this new video coding standard. These changes include: WPP level instead of slice level, coding tree block level instead of MB level, and a different R-D modeling for QP estimation.

References

[1] H. Schwarz, D. Marpe, T. Wiegand, Overview of the Scalable Video Coding Extension of the H.264/AVC Standard, Circuits and Systems for Video Technology, IEEE Transactions on 17 (9) (2007) 1103–1120.

[2] M. Wien, H. Schwarz, T. Oelbaum, Performance Analysis of SVC, Circuits and Systems for Video Technology, IEEE Transactions on 17 (9) (2007) 1194–1203.

[3] M.-J. Chen, G.-L. Li, Y.-Y. Chiang, C.-T. Hsu, Fast Multiframe Motion Estimation Algorithms by Motion Vector Composition for the MPEG-4/AVC/H.264 standard, Multimedia, IEEE Transactions on 8 (3) (2006) 478–487.

[4] V. Agarwal, M. S. Hrishikesh, S. W. Keckler, D. Burger, Clock Rate versus IPC: The End of the Road for Conventional Microarchitectures, in: A. D. Berenbaum, J. S. Emer (Eds.), ISCA, IEEE Computer Society, 2000, pp. 248–259.

[5] H. Sutter, J. Larus, Software and the Concurrency Revolution, Queue 3 (7) (2005) 54–62.

[6] M. Hill, M. Marty, Amdahl's Law in the Multicore Era, IEEE Computer 41 (7) (2008) 33–38.

[7] Z. Chen, K. N. Ngan, Recent Advances in Rate Control for Video Coding, Signal Processing: Image Communication 22 (1) (2007) 19 – 38.

[8] J. Ribas-Corbera, P. Chou, S. Regunathan, A Generalized Hypothetical Reference Decoder for H.264/AVC, Circuits and Systems for Video Technology, IEEE Transactions on 13 (7) (2003) 674–687.

[9] A. Rodriguez, A. Gonzalez, M. Malumbres, Hierarchical Parallelization of an H.264/AVC Video Encoder, in: International Symposium on Parallel Computing in Electrical Engineering, 2006. PAR ELEC 2006., 2006, pp. 363–368.

[10] C. Meenderinck, A. Azevedo, M. Alvarez, B. Juurlink, A. Ramrez, Parallel Scalability of Video Decoders, Journal of Signal Processing Systems 57 (2009) 173–194.

[11] x264. A Free H.264/AVC Encoder, http://www.videolan.org/developers/x264.html (2011).

[12] FFmpeg, http://ffmpeg.org (2011).

[13] T. Wiegand, G. J. Sullivan, G. Bjontegaard, A. Luthra, Overview of the H.264/AVC Video Coding Standard, IEEE Transactions on Circuits and Systems for Video Technology 13 (7) (2003) 560–576.

[14] M. Roitzsch, Slice-balancing H.264 Video Encoding for Improved Scalability of Multicore Decoding, in: Proc. of the 7th ACM & IEEE Int. Conf. on Embedded Software, 2007, pp. 269–278.

[15] T. Jacobs, V. Chouliaras, D. Mulvaney, Thread-Parallel MPEG-2, MPEG-4 and H.264 Video Encoders for SoC Multi-Processor Architectures, IEEE Transactions on Consumer Electronics 52 (1) (Feb. 2006) 269–275.

[16] E. B. V. D. Tol, E. G. T. Jaspers, R. H. Gelderblom, Mapping of H.264 Decoding on a Multiprocessor Architecture, in: Proc. of SPIE, 2003, pp. 707–718.

[17] C. C. Chi, B. Juurlink, A QHD-capable Parallel H.264 Decoder, in: Proc. of the Int. Conf. on Supercomputing, 2011, pp. 317–326.

[18] B. Juurlink, M. Alvarez-Mesa, C. C. Chi, A. Azevedo, C. Meenderinck, A. Ramirez, Scalable Parallel Programming Applied to H.264/AVC Decoding, Springer, 2012.

[19] G. Sullivan, J. Ohm, W.-J. Han, T. Wiegand, Overview of the High Efficiency Video Coding (HEVC) Standard, Circuits and Systems for Video Technology, IEEE Transactions on 22 (12) (2012) 1649–1668.

[20] A. Fuldseth, M. Horowitz, S. Xu, M. Zhou, Tiles, JCTVC-E408, 5th JCTVC Meeting.

[21] F. Henry, S. Pateux, Wavefront Parallel Processing, JCTVC-E196, 5th JCTVC Meeting.

[22] K. Yao, J. Sun, J. Liu, Z. Guo, L. Huo, A Novel Parallel Encoding Framework for Scalable Video Coding, in: Visual Communications and Image Processing (VCIP), 2011 IEEE, 2011, pp. 1–4.

[23] G.-A. Jian, J.-S. Lee, K.-J. Tan, P.-S. Chen, J.-I. Guo, A Real-Time Parallel Scalable Video Encoder for Multimedia Streaming Systems, in: VLSI Design, Automation, and Test (VLSI-DAT), 2013 International Symposium on, 2013, pp. 1–4.

[24] Y.-K. Chen, E. Q. Li, X. Zhou, S. Ge, Implementation of H.264 Encoder and Decoder on Personal Computers , Journal of Visual Communication and Image Representation 17 (2) (2006) 509 – 532.

[25] H.-M. Hang, J.-J. Chen, Source Model for Transform Video Coder and its Application. I. Fundamental Theory, Circuits and Systems for Video Technology, IEEE Transactions on 7 (2) (1997) 287–298.

[26] J. Ribas-Corbera, S. Lei, Rate Control in DCT Video Coding for Low-Delay Communications, Circuits and Systems for Video Technology, IEEE Transactions on 9 (1) (1999) 172–185.

[27] B. Tao, B. Dickinson, H. Peterson, Adaptive Model-Driven Bit Allocation for MPEG Video Coding, Circuits and Systems for Video Technology, IEEE Transactions on 10 (1) (2000) 147–157.

[28] L. Xu, W. Gao, X. Ji, D. Zhao, S. Ma, Rate Control for Spatial Scalable Coding in SVC, in: Picture Coding Symposium, 2007. PCS 2007, 2007.

[29] ISO/IEC, MPEG Test Model 5, ISO/IEC JTC/SC29/WG11, MPEG Test Model 5.

[30] S. Ma, W. Gao, Y. Lu, Rate-Distortion Analysis for H.264/AVC Video Coding and its Application to Rate Control, Circuits and Systems for Video Technology, IEEE Transactions on 15 (12) (2005) 1533–1544.

[31] Y. Liu, Z. G. Li, Y. C. Soh, Rate Control of H.264/AVC Scalable Extension, Circuits and Systems for Video Technology, IEEE Transactions on 18 (1) (2008) 116–121.

[32] J. Si, S. Ma, W. Gao, M. Yang, Adaptive Rate Control for HEVC, JCTVC-J0057, 10th JCTVC Meeting.

[33] T. Chiang, Y.-Q. Zhang, A New Rate Control Scheme Using Quadratic Rate Distortion Model, Circuits and Systems for Video Technology, IEEE Transactions on 7 (1) (1997) 246–250.

[34] S. Ma, Z. Li, F. We, Proposed Draft of Adaptive Rate Control, JVT-H017, 8th JVT Meeting.

[35] B. Xie, W. Zeng, A Sequence-Based Rate Control Framework for Consistent Quality Real-Time Video, Circuits and Systems for Video Technology, IEEE Transactions on 16 (1) (2006) 56–71.

[36] Z. Chen, K. N. Ngan, Towards Rate-Distortion Tradeoff in Real-Time Color Video Coding, Circuits and Systems for Video Technology, IEEE Transactions on 17 (2) (2007) 158 –167.

[37] D.-K. Kwon, M.-Y. Shen, C.-C. J. Kuo, Rate Control for H.264 Video With Enhanced Rate and Distortion Models, Circuits and Systems for Video Technology, IEEE Transactions on 17 (5) (2007) 517–529.

[38] A. Leontaris, A. Tourapis, Rate control for the Joint Scalable Video Model (JSVM), JVT-W043, 24th JVT Meeting.

[39] M. Naccari, F. Pereira, Quadratic Modeling Rate Control in the Emerging HEVC Standard, in: Picture Coding Symposium (PCS), 2012, 2012, pp. 401 –404.

[40] Z. He, Y. K. Kim, S. Mitra, Low-delay Rate Control for DCT Video Coding via ρ-Domain Source Modeling, Circuits and Systems for Video Technology, IEEE Transactions on 11 (8) (2001) 928–940.

[41] Y. Pitrey, M. Babel, O. Deforges, One-Pass Bitrate Control for MPEG-4 Scalable Video Coding Using ρ-Domain, Broadband Multimedia Systems and Broadcasting, 2009. BMSB '09. IEEE International Symposium on (2009) 1–5.

[42] M. Liu, Y. Guo, H. Li, C.-W. Chen, Low-Complexity Rate Control Based on ρ-Domain Model for Scalable Video Coding, in: Image Processing, 2010. ICIP 2010. IEEE International Conference on, 2010, pp. 1277–1280.

[43] S. Wang, S. Ma, S. Wang, D. Zhao, W. Gao, A Quadratic ρ-Domain Based Rate Control Algorithm for HEVC, in: Acoustics, Speech, and Signal Processing, 2013 IEEE International Conference on, 2013, pp. 1695–1699.

[44] M. Dai, D. Loguinov, H. Radha, Rate-Distortion Analysis and Quality Control in Scalable Internet Streaming, Multimedia, IEEE Transactions on 8 (6) (2006) 1135 –1146.

[45] L. Liu, X. Zhuang, Z. He, Y. Sun, H.264/AVC Rate Control with Enhanced Rate-Quantisation Model and Bit Allocation, Image Processing, IET 5 (7) (2011) 619–629.

[46] N. Kamaci, Y. Altunbasak, R. Mersereau, Frame Bit Allocation for the H.264/AVC Video Coder via Cauchy-Density-Based Rate and Distortion models, Circuits and Systems for Video Technology, IEEE Transactions on 15 (8) (2005) 994–1006.

[47] Y. Cho, J. Liu, D.-K. Kwon, C.-C. Kuo, Joint Quality-Temporal (Q-T) Bit Allocation for H.264/SVC, in: Circuits and Systems, 2009. ISCAS 2009. IEEE International Symposium on, 2009, pp. 2361 –2364.

[48] J. Liu, Y. Cho, Z. Guo, J. Kuo, Bit Allocation for Spatial Scalability Coding of H.264/SVC With Dependent Rate-Distortion Analysis, Circuits and Systems for Video Technology, IEEE Transactions on 20 (7) (2010) 967 –981.

[49] S. Sanz-Rodríguez, T. Schierl, A Rate Control Algorithm for HEVC with Hierarchical GOP Structures, in: Acoustics, Speech, and Signal Processing, 2013 IEEE International Conference on, 2013, pp. 1719–1723.

[50] B. Li, H. Li, L. Li, J. Zhang, Rate Control by R-Lambda Model for HEVC, JCTVC-K0103, 11th JCTVC Meeting.

[51] N. Mohsenian, R. Rajagopalan, C. A. Gonzales, Single-Pass Constant- and Variable-Bit-Rate MPEG-2 Video Compression, IBM Journal of Research and Development 43 (4) (1999) 489 –509.

[52] M. Rezaei, M. Hannuksela, M. Gabbouj, Semi-Fuzzy Rate Controller for Variable Bit Rate Video, Circuits and Systems for Video Technology, IEEE Transactions on 18 (5) (2008) 633–645.

[53] H. Lee, Y. Lee, D. Lee, J. Lee, H. Shin, Implementing Rate Allocation and Control for Real-Time H.264/SVC Encoding, in: Consumer

Electronics (ICCE), 2010 Digest of Technical Papers International Conference on, 2010, pp. 269 –270.

[54] S. Sanz-Rodríguez, F. Díaz-de María, In-Layer Multibuffer Framework for Rate-Controlled Scalable Video Coding, Circuits and Systems for Video Technology, IEEE Transactions on 22 (8) (2012) 1199 –1212.

[55] P. H. Westerink, R. Rajagopalan, C. A. Gonzales, Two-Pass MPEG-2 Variable-Bit-Rate Encoding, IBM Journal of Research and Development 43 (4) (1999) 471 –488.

[56] Y. Yu, J. Zhou, Y. Wang, C. W. Chen, A Novel Two-Pass VBR Coding Algorithm for Fixed-Size Storage Application, Circuits and Systems for Video Technology, IEEE Transactions on 11 (3) (2001) 345 –356.

[57] A. Jagmohan, K. Ratakonda, MPEG-4 One-Pass VBR Rate Control for Digital Storage, Circuits and Systems for Video Technology, IEEE Transactions on 13 (5) (2003) 447–452.

[58] M. de Frutos-Lopez, O. del Ama-Esteban, S. Sanz-Rodriguez, F. Diaz-de Maria, A Two-Level Sliding-Window VBR Controller for Real-Time Hierarchical Video Coding, in: Image Processing, 2010. ICIP 2010. IEEE International Conference on, 2010, pp. 4217–4220.

[59] M. Jiang, N. Ling, Low-delay Rate Control for Real-Time H.264/AVC Video Coding, Multimedia, IEEE Transactions on 8 (3) (2006) 467–477.

[60] C.-Y. Chang, M.-H. Chen, C.-F. Chou, D.-Y. Chan, A Two-Layer Characteristic-Based Rate Control Framework for Low Delay Video Transmission, in: Communications, 2007. ICC '07. IEEE International Conference on, 2007, pp. 2699–2704.

[61] T. Anselmo, D. Alfonso, Buffer-Based Constant Bit-Rate Control for Scalable Video Coding, in: Picture Coding Symposium, 2007. PCS 2007, 2007.

[62] S. Sanz-Rodríguez, O. del Ama-Esteban, M. de Frutos-López, F. Díaz de María, Cauchy-Density-Based Basic Unit Layer Rate Controller for H.264/AVC, IEEE Transactions on Circuits and Systems for Video Technology 20 (8) (2010) 1139–1143.

[63] A. Javdtalab, M. Omidyeganeh, S. Shirmohammandi, M. Hosseini, A Rate Control Algorithm for x264 High Definition Video Conferencing, in: Multimedia and Expo (ICME), 2011 IEEE International Conference on, 2011, pp. 1–6.

[64] H. Li, Z. Li, C. Wen, L.-P. Chau, Fast mode decision for spatial scalable video coding, in: Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium on, 2006, pp. 3005–3008.

[65] H. Li, Z. Li, C. Wen, Fast mode decision algorithm for inter-frame coding in fully scalable video coding, Circuits and Systems for Video Technology, IEEE Transactions on 16 (7) (2006) 889–895.

[66] S.-T. Kim, K. Reddy Konda, C.-S. Park, C.-S. Cho, S.-J. Ko, Fast mode decision algorithm for inter-layer coding in scalable video coding, Consumer Electronics, IEEE Transactions on 55 (3) (2009) 1572–1580.

[67] S. Van Leuven, G. Van Wallendael, K. De Wolf, J. De Cock, P. Lambert, R. Van de Walle, An enhanced fast mode decision model for spatial enhancement layers in scalable video coding, Multimedia Tools and Applications 58 (1) (2012) 215–237.

[68] ISO/IEC, Generic Coding of Moving Pictures and Associated Audio Information - Part 2: Video, ITU-T Recommendation H.262-ISO/IEC 13818-2, MPEG-2.

[69] ITU-T, Video Coding for Low Bitrate Communication, ITU-T Draft Recommendation H.263 Version 1.

[70] T. Sikora, The MPEG-4 Video Standard Verification Model, Circuits and Systems for Video Technology, IEEE Transactions on 7 (1) (1997) 19–31.

[71] H. Schwarz, D. Marpe, T. Wiegand, Analysis of Hierarchical B Pictures and MCTF, in: Multimedia and Expo, 2006 IEEE International Conference on, 2006, pp. 1929 –1932.

[72] W. Wan, Y. Chen, Y.-K. Wang, M. Hannuksela, H. Li, M. Gabbouj, Efficient Hierarchical Inter Picture Coding for H.264/AVC Baseline Profile, in: Picture Coding Symposium, 2009. PCS 2009, 2009, pp. 1–4.

[73] S. Sanz-Rodríguez, F. Díaz-de María, Rate Control Initialization Algorithm for Scalable Video Coding, in: Image Processing (ICIP), 2011 18th IEEE International Conference on, 2011, pp. 3497 –3500.

[74] S. Thakkar, T. Huff, The Internet Streaming SIMD Extensions, IEEE Computer 32 (12) (1999) 26–34.

[75] F. Bossen, Common Test Conditions and Software Reference Configurations, JCTVC-L1100, 13th JCTVC Meeting.

[76] X. Li, J. Boyce, P. Onno, Y. Ye, Common Test Conditions and Software Reference Configurations for the Scalable Test Model, JCTVC-L1009, 12th JCTVC Meeting.

[77] J. Vieron, M. Wien, H. Schwarz, Joint Scalable Video Model (JSVM) (2011).
URL `http://www.hhi.fraunhofer.de/de/kompetenzfelder/image-processing/research-groups/image-video-coding/svc-extension-of-h264avc/jsvm-reference-software.html`

[78] G. Bjøntegaard, Calculation of average PSNR differences between RD curves, VCEG contribution, VCEG-M33, Austin.