Secure Scalable Video Compression for GVid

Heinz Hofbauer and Thomas Stütz and Andreas Uhl [*]

Abstract. *GVid is a Grid service that enables the secure and transparent integration and development of graphical user interface applications in the Grid. It separates the potentially computationally complex task of data creation and visualization, e.g., scientific simulations, from the comparably computationally inexpensive task of transmission and display of the visual data. A Grid application produces visual data and GVid takes care of the encoding, the secure and efficient transmission and the display of the visual data. As the transmission parameters and grid node properties are highly variable, special compression schemes have to be chosen to cope with these requirements. Beneficial for such requirements is the application of scalable compression formats, such as H.264/SVC (Scalable Video Coding) and MC-EZBC (Motion-Compensated Embedded Zerotree Block Coding). As simulation data may be sensitive, e.g., in the case of medical simulations, the secure transmission and storage of the visual data has to be guaranteed. Format-specific encryption schemes offer improved functionality due to the preservation of scalability in the encrypted domain. In this work the compression performance of state-of-the-art scalable video compression systems is evaluated and format-specific encryption schemes are proposed and discussed.*

1. Introduction

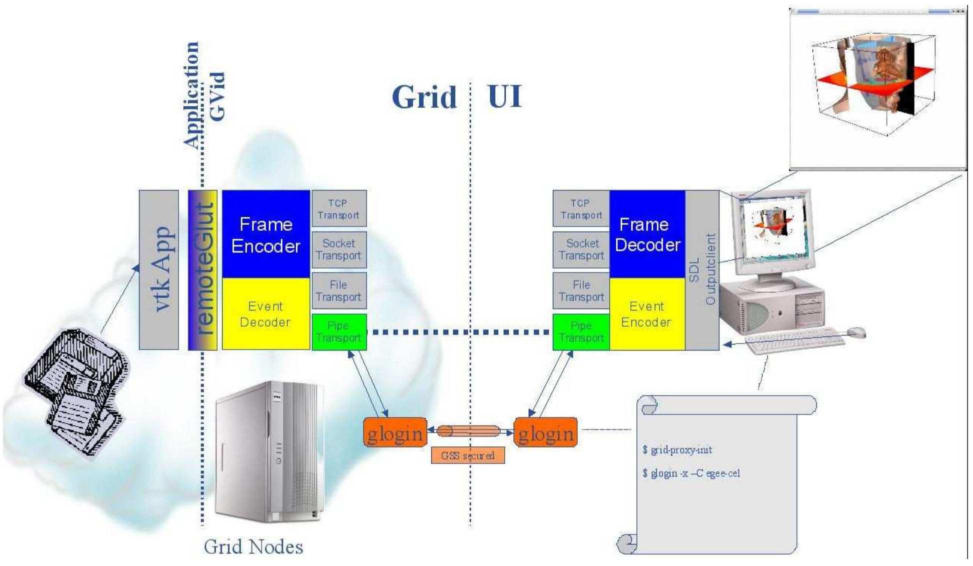
The GVid framework and implementation has been introduced and discussed in previous work [6,10]. The GVid framework separates the task of data generation and visualization from the comparably computationally inexpensive task of transmission and display of the visual data. This separation is especially reasonable if the visual data is displayed on a computationally weak device. Mobile devices have become the most frequent computing platform for a majority of users, even if many of them are not even aware that there mobile device is essentially a general purpose computer with an extended set of hardware. Thus Andrew S. Tanenbaum's ironic statement "Computers are different from telephones. Computers do not ring." [12] has lost its context. A major difference between telephones and computers remain the different computational capabilities and further constraints of telephones, which are nowadays almost exclusively mobile devices. Mobile devices suffer from slower CPUs, less memory, lower resolution displays, and network connections with lower bandwidth, but with a higher probability of connection loss. Especially the restricted computational capabilities are a convincing argument for the separation of data generation and visualization from the comparably computationally inexpensive task of transmission and display. This topic is currently in the focus of research, e.g., Advanced Micro Devices (AMD) is currently working on a supercomputer for graphic rendering to enable 3D game playing for cellphones [9]; an approach rather similar to GVid. Additionally the varying network parameters paired with a higher probability of connection loss for mobile devices pushes the development of another line of research, namely scalable and error resilient for-

[*]Department of Computer Sciences, University Salzburg, Salzburg, Austria, email: {hhofbaue, tstuetz, uhl}@cosy.sbg.ac.at

mats and transmission systems for visual data. Scalable visual data formats enable simple and fast rate adaptation. In previous work the scalable still image standard JPEG2000 has been employed for intra-frame compression. At present the scalable extension of the video coding standard H.264 (SVC) has been finalized and thus an applicable scalable video compression system is now available. A different approach to implement a scalable video format compared to the traditional layered design of H.264/SVC is followed by the wavelet-based MC-EZBC codec. Both schemes offer state-of-the-art scalable video compression and are therefore evaluated for the suitability as compression codecs within the GVid framework (see section 2. for details on the GVid structure and section 3. for details on the codecs). Their compression performance is evaluated in section 3.3. In section 4. format-specific encryption approaches are discussed for the two schemes together with a motivation and introduction to format-specific encryption. A format-specific encryption scheme for MC-EZBC is proposed in this work. The major advantages of format-specific encryption schemes are the preservation of scalability in the encrypted domain, i.e. rate adaptation can still be conducted, and a potentially improved error robustness and resilience. A concluding comparison of the two compression systems and their corresponding format specific encryption schemes is given in section 5. Additionally an outline of future work is presented, discussing the potentials to improve the runtime performance of scalable compression systems via parallel and distributed compression within the Grid.

2. GVid: Secure Interactive Video Transmission

The GVid software is a result of a joint project of the Institute of Graphics and Parallel Processing (GUP) at the Joh. Kepler University Linz and the Department of Computer Sciences at the University of Salzburg, which included Thomas Köckerbauer, Dieter Kranzlmüller, Martin Polak, Herbert Rosmanith, Thomas Stütz and Andreas Uhl.



```
$ grid-proxy-init
$ glogin -x –C egee-cel
```

Figure 1. GVid Component Overview

2.1. The Structure of GVid

The aim of GVid software design was to support as many applications as easily as possible. Therefore, several input adapters exist that are responsible for acquiring the visual data of the application.

Currently a freeGLUT [1], a vtk [2] and a X11 based input adapter are implemented. The X11 input adapter enables every X11 application to be transmitted over the Grid.

Figure 1 illustrates an overview of the GVid design. An application provides the visual data through one of the several input adapters and GVid takes care of the encoding, the secure and efficient transmission and the display of the visual data. New compression and security schemes can be easily integrated. Currently a compression plug-in for MPEG-4 (Xvid) and JPEG2000 are integrated. Xvid does not provide a scalable format stream and JPEG2000 does not exploit inter frame redundancy. Thus for Xvid rate adaptation or delivery of streams at different rates can not be done efficiently and for JPEG2000 bandwidth could be saved by exploiting inter frame redundancy and yielding more efficient compression. A scalable video compression system would perfectly meet the requirements of efficient compression and scalability of the video format stream.

2.2. Confidentiality and Scalability in the GVid Framework

Scalability enables efficient rate adaptation, an important feature in an environment characterized by highly frequent network bandwidth changes. A scalable (video) format is the fundamental basis for efficient rate adaptation and enables advanced streaming and multicast scenarios, such as receiver driven layered multicast (RLM) [8]. RLM solves the adaptation to changing network conditions by receiver actions, i.e. join and leave of IP multicast groups, (receiver driven). However, IP multicast is not widely deployed and other implementations have to be considered for rate-adaptive streaming.

The idea of the application of scalable format streams for network adaptation has been extended to in-network adaptation systems, in which adaptation is dynamically performed in the network by a MANE (media aware network element). The basic setup is illustrated in figure 2. These in-network adaptation systems are assumed to offer rapid adaptation to changing network conditions as the delay for the propagation of changed network parameters is minimized. However, implementing such in-network systems within the scope of already existing and well-established transmission protocols, such as RTP, has been proved to contain certain pitfalls [7, 17]. Nonetheless, the idea of in-network adaptation can be considered sensible and as a potential candidate for the integration in the GVid framework. Integrating security services, i.e. confidentiality in in-network adaptation systems, is not straight-forward. The application of well-established security tools, e.g., SRTP, SSL or IPSEC, is not possible as the necessary information to perform rate-adaptation within the network is concealed and thus not available at the MANE. Thus if confidentiality and in-network adaptation are to be combined, format-specific encryption schemes, that preserve the information necessary for rate adaptation, are needed.

In multiple client scenarios (see figure 2) the application of scalable compression systems offers substantial advantages. In these scenarios the visual output of a Grid application is transmitted to multiple clients, each with its own preferences and parameters for the visual content ant its transmission (e.g., rate and resolution). If conventional compression systems (i.e., systems not delivering scalable format streams) are employed, a separate compression task for each client has to be performed. These separate compression tasks are, considering the computational complexity of state-of-the-art video compression, an enormous burden. The solution of separate compression tasks does not scale well with the number of clients, i.e., each new client with distinct preferences adds another separate compression task. Scalable compression systems can solve this issue, as only one single compression task generates a scalable format stream, that can efficiently be adapted to each client's preferences. This paradigm of a single encoding step with subsequent computationally efficient adaption steps is

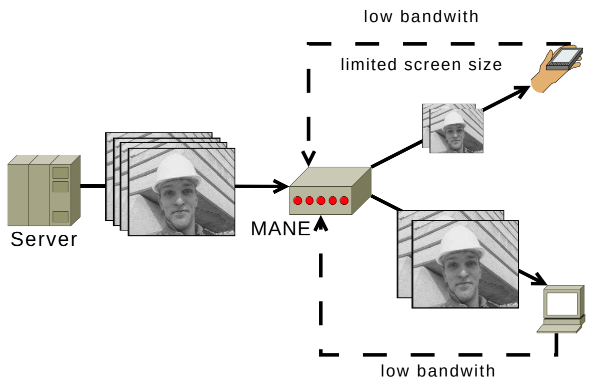


Figure 2. Example of a single video sequence from the server which is adapted to the given capabilities of two end devices.

referred to as Universal Multimedia Access (UMA) [14]. In case that confidential transmission has to be guaranteed, well-established security tools could be applied, but again these solutions do not scale well with the number of clients. In fact a separate encryption task has to be performed for each client (even if the clients share the same preferences). Format-specific encryption schemes offer a well-scaling solution. These schemes encrypt a scalable format stream in a specific scalability-preserving fashion. The still scalable but secured format stream can efficiently be adapted to the clients preferences and confidentially transmitted.

In conclusion we can state that the application of scalable compression systems and format specific encryption within GVid offers ample benefits and should thus be seriously considered.

Only recently SVC, the scalable extension of H.264, has been standardized [5] and therefore it is worth evaluating the suitability of this new compression system for the application within GVid. Additionally the wavelet-based scalable video codec MC-EZBC is evaluated.

3. State-of-the-Art Scalable Video Compression

In the following two scalable video compression systems are presented, which also represent two different approaches to implement scalable video coding.

SVC follows the traditional design of layered video coding [5], while MC-EZBC is a t+2D wavelet-based video codec with motion-compensated temporal filtering.

3.1. H.264/SVC

A major design requirement for SVC has been the backwards compatibility to the existing H.264/AVC. Thus SVC format streams are valid H.264/AVC format streams (format-compliant with respect to the non-scalable H.264/AVC format) and thus decodeable by H.264/AVC compliant decoders. Major parts of the H.264 AVC video coding system have been adopted, including most of the H.264 AVC syntax and semantics. An SVC format stream contains a base layer and one or more enhancement layers each may augment the user experience in one of three dimensions (temporal/spatial/quality).

3.1.1. Temporal Scalability

A format stream is temporally scalable if it contains sub streams with lower frame rates. Due to the flexible inter prediction in H.264/AVC, the implementation of temporal scalability within H.264/AVC/SVC has been straightforward by employing special prediction structures, e.g., dyadic temporal enhancement layers with hierarchical B-pictures. In figure 3 the dyadic hierarchical B-picture prediction structure is illustrated, but temporal scalability in H.264/AVC/SVC is not limited to dyadic prediction structures; SVC offers the syntax to easily extract a sub stream with a reduced frame rate by simply dropping parts of the format stream.

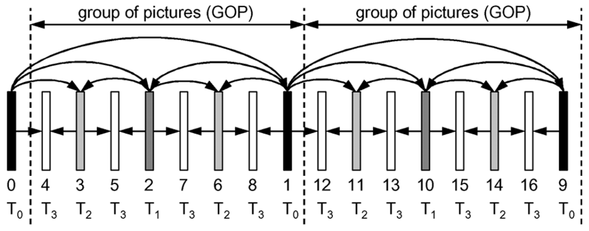

group of pictures (GOP)
group of pictures (GOP)
0 4 3 5 2 7 6 8 1 12 11 13 10 15 14 16 9
T_0 T_3 T_2 T_3 T_1 T_3 T_2 T_3 T_0 T_3 T_2 T_3 T_1 T_3 T_2 T_3 T_0

Figure 3. Prediction hierarchy of B-pictures in SVC

3.1.2. Spatial Scalability

A format stream is spatially scalable if it contains sub streams with different resolutions. SVC implements spatial scalability with a conventional multilayer approach. A base layer (lower resolution) is encoded in H.264/AVC compliant fashion, while the enhancement layers (containing higher resolutions) may apply inter layer prediction in order to exploit redundancies between the layers. Spatial scalability with arbitrary resolutions is supported.

3.1.3. Quality Scalability

A format stream is quality scalable if it contains substreams with different qualities, in a signal to noise ratio (SNR) sense, but same resolution. In SVC the so called key-picture concept, also known as medium grain scalability (MGS), is employed to enable quality scalability.

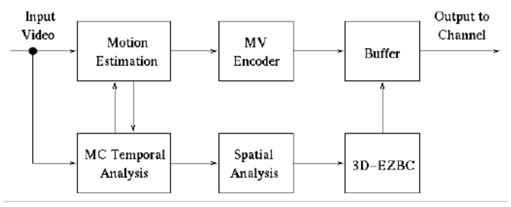
3.1.4. SVC NAL units

A network abstraction layer (NAL) unit in H.264 is preceded by an 1-byte NAL unit header, containing most importantly the NAL unit type. On the basis of the NAL unit type the NAL unit data is processed. For SVC the NAL header is extended, a three byte extension is added. This extension contains a dependency_id, which identifies the spatial layer to which the NAL unit data contributes, a temporal_id, which specifies the temporal layer of the NAL unit, and a quality_id, which specifies to which quality layer the NAL unit contributes.

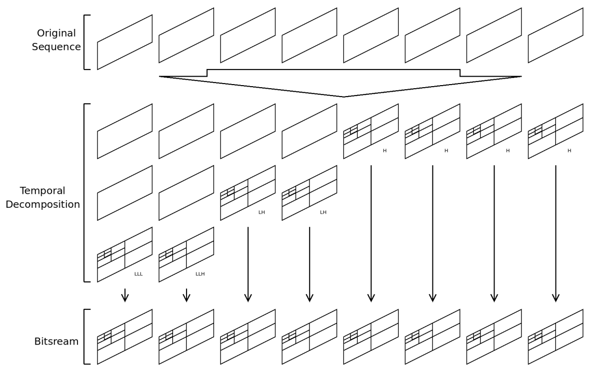
3.2. MC-EZBC

The MC-EZBC [4,19] coder is a t+2D wavelet coder, i.e., a wavelet transform is applied for temporal decomposition as well as for spatial decomposition. The abbreviation t+2D implies that the temporal decomposition combined with motion estimation is applied before the spatial decomposition (both apply pyramidal decomposition structures). The 9/7 CDF (Cohen-Daubechies-Feauveau) wavelet filters are applied for spatial decomposition, while temporal decomposition is conducted with the CDF 5/3 wavelet filters. Furthermore adaptive prediction techniques are employed. The layout of the MC-EZBC coder is shown in figure 4(a)

Figure 4(b) illustrates the encoding process for a group of pictures (GOP). The frames of a raw video sequence are split into GOPs, which are independently coded. In a GOP frames are decomposed temporally and then spatially. Note that the ordering of the coded frames follows the temporal decomposition level, i.e., the deepest temporal low-pass frames are the first contributions in the final stream of a GOP.



(a) Layout of the MC-EZBC coder



(b) Encoding of a GOP showing spatial and temporal decomposition

Figure 4. MC-EZBC

3.2.1. Temporal Scalability

The MC-EZBC format stream automatically supports temporal scalability; this property is due to the temporal wavelet decomposition. If the GOP size is 2^t, then the number of temporal resolutions, i.e., different frame rates is t. Temporal scaling is done by dropping levels from the temporal decomposition, i.e., one step of temporal scaling reduces the frame rate by half. For example a GOP with $16 = 2^4$ frames could be reduced to $8, 4$ or 2 frames. Only dyadic temporal prediction structures are permitted.

3.2.2. Spatial Scalability

Spatial scalability is also automatically supported and like temporal scalability is done by dropping high frequency wavelet bands. Again scaling operations are discrete with steps of half the resolution of the previous step, e.g. a CIF (352×288) video could be scaled to qCIF (176×128) or sqCIF (88×64). Only dyadic spatial resolution changes are permitted.

3.2.3. Quality Scalability

Quality scalability unlike spatial or temporal scalability is more flexible. SNR scalability of the MC-EZBC achieves multiple bitrates within a single format stream. The coded wavelet coefficient data is arranged in an embedded bitstream, i.e., a truncated segment of the coded data is still decodeable and results in a quantized representation of the wavelet coefficient data.

3.3. Performance Evaluation

The following performance evaluation is intended to give an overview of the capabilities of the two scalable compression formats, SVC and MC-EZBC, and their suitability for the application within the GVid framework. For more extensive and exhaustive treatments on the compression performance of these codecs the reader is referred to [18]

In the assessment of the compression performance of scalable compression systems subtle pitfalls are hidden. The two scalable compression systems may contain substreams with different resolutions. However, the lower resolution versions of the original content contained in the format streams of SVC and MC-EZBC are different. In SVC the subsampling method at the encoder-side is not specified in the standard, however, the upsampling method is specified and it is therefore sensible to employ the corresponding subsampling method. In the MC-EZBC the subsampling method is defined by the low-pass filter of the spatial wavelet decompostion (9/7 CDF). Thus taking a common reference for quality assessment, peak signal to noise ratio (PSNR) calculation, for both schemes always and systematically favours one of the compression systems. Therefore the compression performance for lower resolution substreams is assessed for each compression system individually with the correct reference, i.e., the lower resolution reference sequences for the MC-EZBC are generated with the low-pass filters of the 9/7 CDF and the lower resolution reference sequences for SVC are generated with the subsampling filters fitting to the normative upsampling filters.

The quality for lower frame rate substreams is assessed with reference to the original sequence where frames have been dropped, i.e., every second frame is dropped if the frame rate is halved.

In this evaluation the well-known foreman sequence in the CIF format (352x288) with 96 frames at a frame rate of 30 fps is employed.

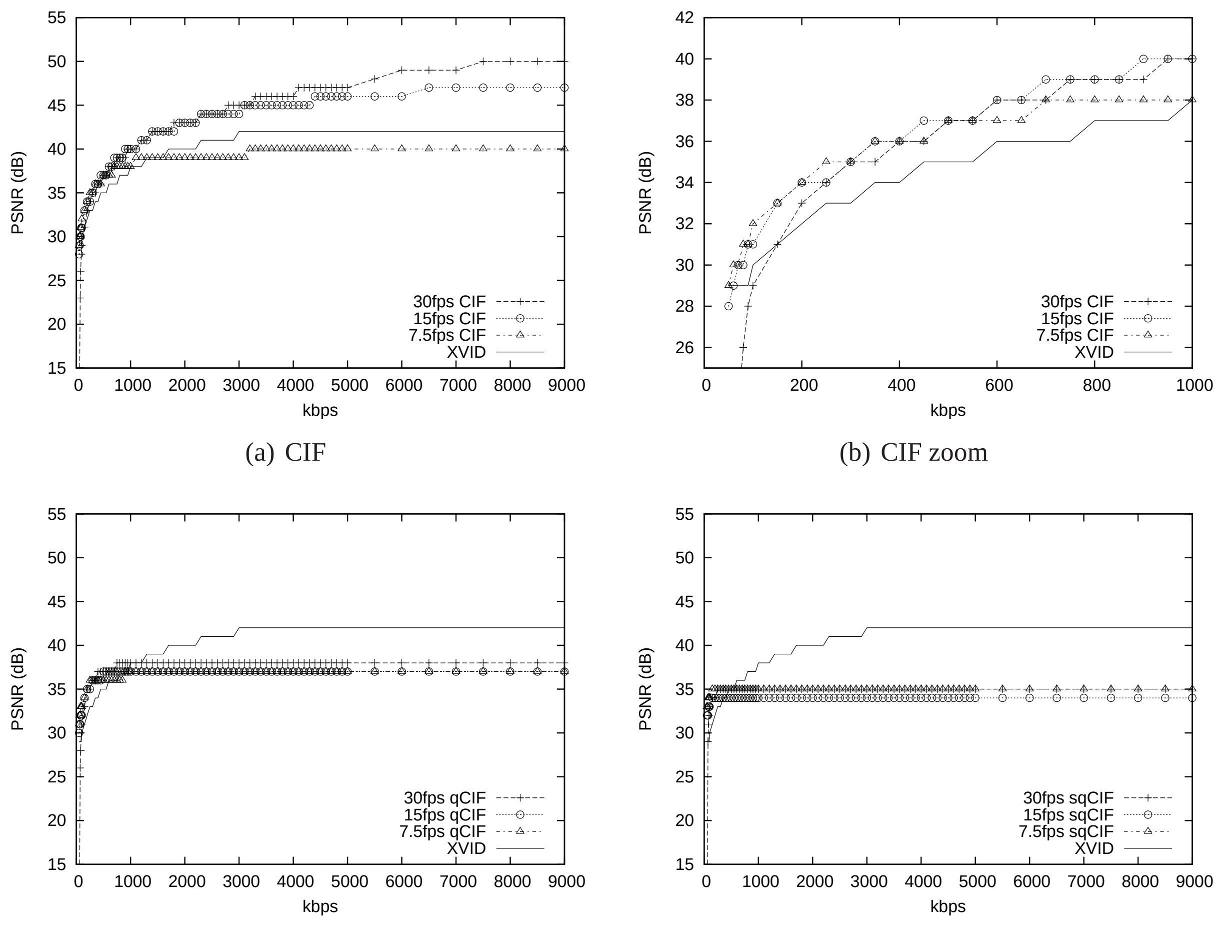
3.3.1. Performance of the MC-EZBC

The compression performance of the MC-EZBC is summarized in figure 5. Most notably are the multiple bitrates contained within the single scalable MC-EZBC stream, illustrated by dots in the figure. It is also noteworthy that for a regular CIF version with full framerate the MC-EZBC performs better than the widely used XVID codec, fig. 5(a) and 5(b).

3.3.2. Performance of the H.264/SVC

The main issue for the performance evaluation is the definition of suitable encoder configurations. The encoder configuration is decisive for the compression performance and it also defines extraction points (i.e., bitrates at which reconstruction is possible). In general, it can be summarized that temporal scalability comes for free and even improves the compression performance, while the other types of scalability decrease the compression performance, but increase the number of extraction points. H.264/SVC is a layered video codec allowing only a discrete number of extraction points. This is a major difference to the MC-EZBC codec, which allows the extraction of arbitrary bitrates from the stream.

The following figures illustrate the extraction points and their respective PSNR for different encoder configurations. A point in the figure represents an extraction point.



(a) CIF (b) CIF zoom

(c) qCIF (d) sqCIF

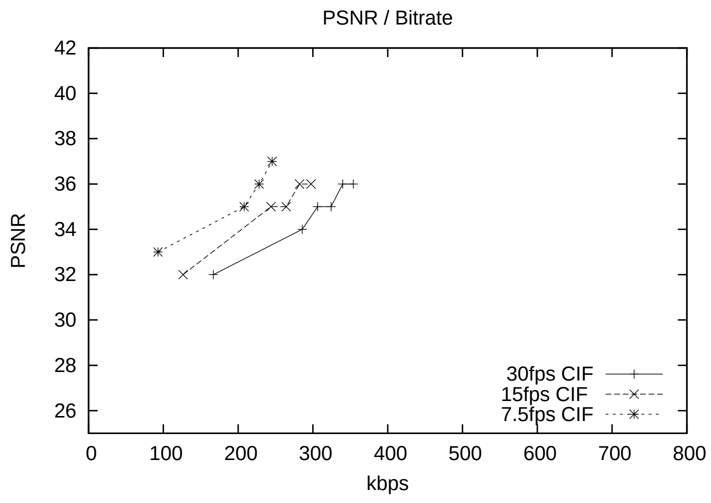
Figure 5. Rate-distortion plots for the foreman sequence and different resolutions (CIF, qCIF and sqCIF) as well as a zoomed version for the low bitrates of the CIF plot.

First we discuss two configurations that implement temporal and quality scalability. These configurations are suitable for computationally strong devices such as home PCs, therefore smaller resolutions and a simple base layer (e.g., suitable for mobile devices) are omitted.

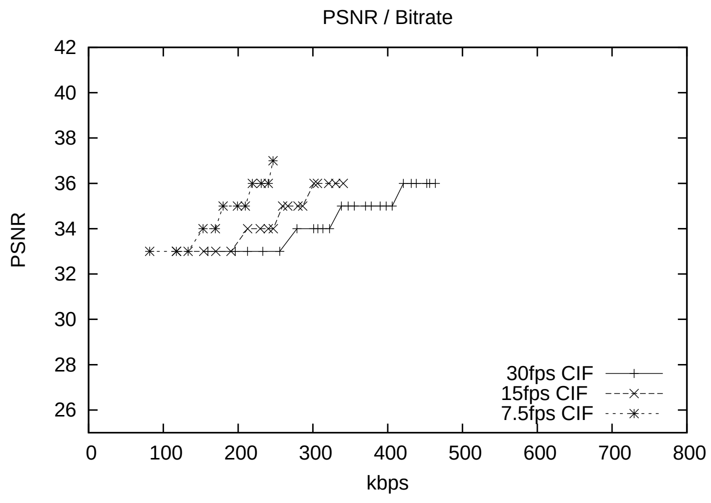
Figure 6(a) shows a simple configuration with only one spatial resolution and one MGS enhancement layer.

Figure 6(b) shows a simple configuration with only one spatial resolution and 8 MGS enhancement layers. MGS is a mode very similar to progressive JPEG, namely the spectral selection mode of operation. In this configuration the 16 transform coefficients of the 4x4 transform are grouped into 8 partitions each containing exactly two transform coefficients.

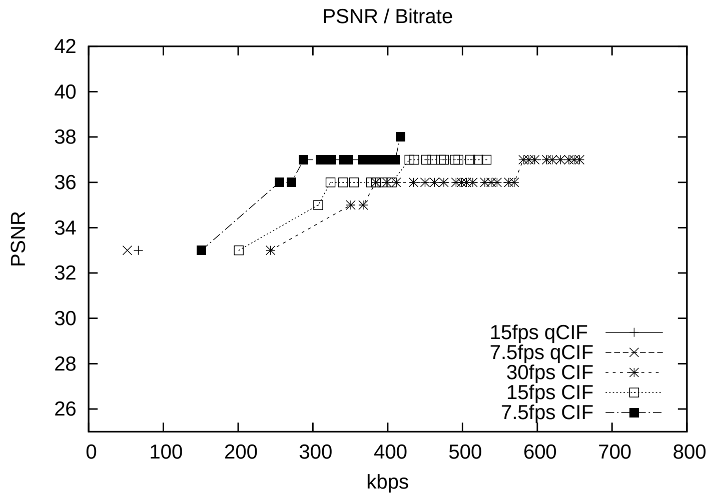
Additionally lower resolution substreams can be defined. For the fine configuration a QCIF resolution is contained in the substream. This substream is encoded within the limits and constraints of the H.264/AVC baseline profile (CAVLC). The bitstream may be used to serve both a computationally weak device such as a mobile phone and a PC. The number of reference frames is set to 1. In figure 6(c) the extraction points for this configuration are illustrated.



(a) Results for the coarse configuration.



(b) Results for the coarse2 configuration.



(c) Results for the fine configuration.

Figure 6. The foreman sequence with 30 fps under different coder configurations.

3.3.3. Comparison between H.264/SVC and MC-EZBC

MC-EZBC's compression performance (at least of the encoder configurations we have tested) is at least equal to H.264/SVC (see figure 7). It has to be noted, that the results for H.264/SVC have been obtained with the reference software JSVM and that other implementations of the H.264/SVC standard may offer better compression performance. If both codecs are compared to state-of-the-art MPEG-4 / H.263 encoders (Xvid), the clear resume is that both perform significantly better for a broad range of bitrates (see figure 8).

The advantage of the MC-EZBC is its higher flexibility in terms of possible extraction points; beneficial if fine grained rate adaptation is to be performed.

There are several arguments for H.264/SVC: It is backwards-compatible to H.264, which allows the base layer to be decoded with a compliant H.264 decoder, e.g., special hardware chips. It is scalable in terms of computational complexity. The base layer can be encoded such that decoding has a very low computational complexity, e.g., arithmetic coding can be omitted.

Another advantage of H.264/SVC is related to the interactive usage possible in the GVid framework. It allows zero structural delay, i.e., the inter-prediction process can be configured to allow only forward prediction. Thus every frame can immediately be coded and transmitted. In case of MC-EZBC this is not possible as frames have to be processed on a GOP-basis, i.e., a number of frames (the GOP size) have to be buffered and delayed until the coding and transmission can be conducted. The introduced delay is adverse to interactive usage, where low delay is preferred.

In conclusion, MC-EZBC offers better rate adaptation, but H.264/SVC provides other important features MC-EZBC lacks.

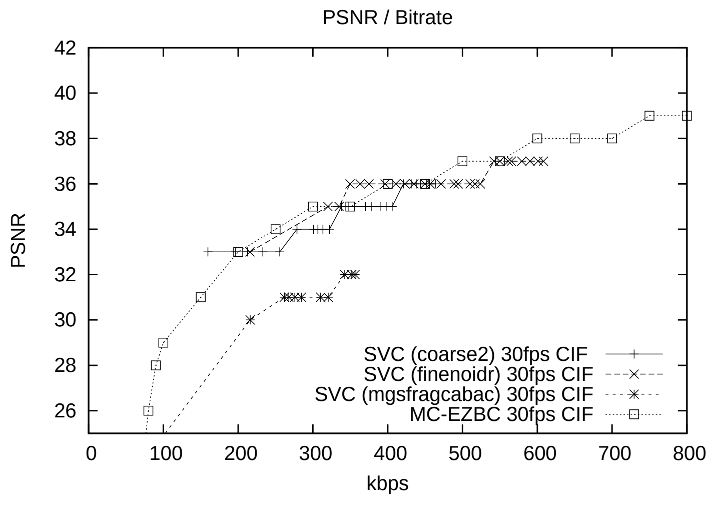


Figure 7. MC-EZBC compared to H.264/SVC

4. Format-Specific Encryption Schemes

Format-specific scalability-preserving encryption schemes are necessary in order to combine efficient transmission and confidentiality.In the following format-specific encryption schemes are discussed for H.264/SVC and MC-EZBC.

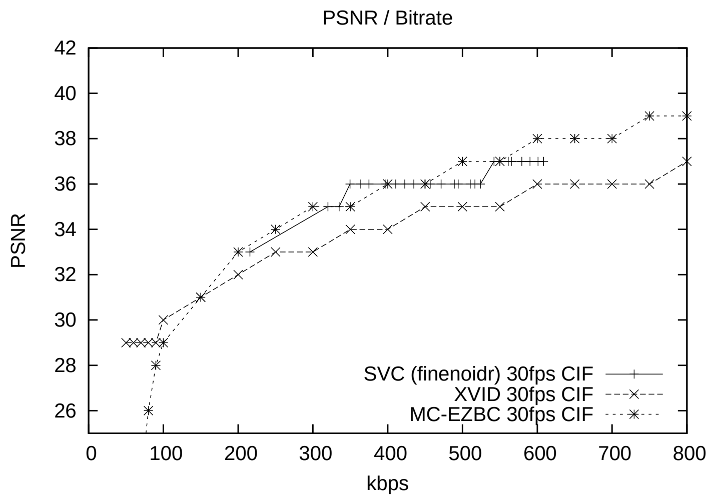


Figure 8. Comparison of Xvid, MC-EZBC and H.264/SVC

4.1. H.264/SVC-Specific Encryption

In order to preserve the scalability of the H.264/SVC stream the information to which dependency layer, temporal layer and quality layer a NAL unit contributes, which is part of the SVC extended NAL unit header, has to be preserved. Thus only encrypting the NAL unit body preserves scalability. However, straight-forward conventional encryption of the NAL unit body is problematic, as NAL unit bodies obey certain syntax rules. Namely marker sequences, that e.g., signal the beginning and the end of a NAL unit, are forbidden. Thus conventional encryption of NAL unit bodies is likely to break the system at some point, e.g., if the H.264/SVC byte-stream format is used and a marker sequence is accidentally generated in a NAL unit body, the entire synchronization is lost.

A way to prevent such behaviour is to ensure that format-specific encryption produces a format-compliant encrypted stream (format-compliant encryption). As a result it can be guaranteed that a decoder does not crash decoding such a stream.

In [11], the H.264/SVC header is preserved and unspecified NAL unit types are employed to signal encrypted data. For the most frequent NAL unit types (NUTs 1, 5, 14, 20) a direct mapping to unspecified NAL unit type values is defined. For all other NAL unit types, the original NAL unit header is preserved as the first payload byte and a certain unspecified NAL unit type is used to signal these encrypted NAL units. However, if packaging is applied as specified in the RFC 3984 [15] and the draft RFC defining the RTP payload for H.264/SVC video [16], all but one (NUT 0) of the unspecified NUT values are already assigned a specific meaning. Hence, the only possibility to employ unspecified NAL unit types to signal encrypted data is NUT 0 [3]. A NAL unit selected for encryption is prefixed by a NAL unit header with NUT 0, and the original NAL unit header and the H.264/SVC header are the first bytes of the encrypted NAL unit payload, and the remaining NAL unit payload is encrypted. However, special care must be taken to avoid marker sequences (H.264 marker sequences are prefixed by at least two zero bytes). This is a problem if encryption is applied more than once, i.e., encrypted NAL units are encrypted. A straight-forward solution is to set the NRI field in the NAL unit header to a value not equal to 0.

4.1.1. Format Compliance and Encryption

Although the encrypted NAL unit has to be ignored by a compliant decoder, certain syntax requirements have to be met by the encrypted NAL unit. These requirements are given in [5]; namely that within the NAL unit, the following three-byte sequences shall not occur at any byte-aligned position: 0x000000, 0x000001, 0x000002, and 0x000003.

Additionally, within the NAL unit, any four byte sequence that starts with 0x000003 other than the following sequences shall not occur at any byte-aligned position: 0x00000300, 0x00000301, 0x00000302, and 0x00000303. Additionally, the last byte of a NAL unit shall not be 0x00.

The encryption scheme has to ensure that these requirements are met. Therefore, after encryption the procedure for the encapsulation of an SODB (string of data bits) within an RBSP (raw byte sequence payload) [5] has to be applied. For the case of two consecutive 0x00 bytes, this procedure ensures that the NAL unit does not end with a 0x00 byte. If a NAL unit ends with a 0x00 byte, it has to end with two consecutive 0x00 bytes for all currently specified RBSP types.

Encrypted NAL unit payloads may not have this property and thus special care has to be taken for the encryption of the last byte of a NAL unit. In our approach we use AES in Counter Mode and treat the last byte with special care.

Every cipher byte, except the last one, is the plaintext byte XORed with a keystream byte. The last cipher byte c is derived from the plaintext byte p and a keystream byte k (optimally in the range [0x00,0xfe], which can be ensured by ignoring 0xff bytes from the keystream) in the following way:

$$c = (p - 1 + k) \texttt{mod 0xfe} + 1$$

For decryption the following procedure is applied:

$$p = (c - 1 - k) \texttt{mod 0xfe} + 1$$

In order to ensure format compliance and decodability by any conformant decoder, an appropriate set of NAL units has to be selected for encryption.

4.2. MC-EZBC-Specific Encryption

As format-specific encryption for MC-EZBC heavily relies on its bitstream format, we start the with a thorough discussion of the MC-EZBC format. A schematic overview of the MC-EZBC format stream is given in figure 9, the organization of GOP data is outlined in figure 4(b). The main header followed by GOP sizes (this is the size of the image data in a GOP) followed by coded data of sequential GOPs. In the following the coded data of a GOP is referred to as GOP as well. Each GOP is lead by a header, giving scene change information, i.e. which frames are I frames, followed by the motion field and coded image data. Both motion field and image data are ordered by frame; frames are ordered lowest to highest temporal resolution. The image data of a frame is also arranged from lowest to highest resolution and a spatial decomposition of a frame is grouped together as a basic image data unit (we will call them chunks from now on). Each chunk is preceded by a leading header defining the length of the chunk and groups all chroma information of a given decomposition level. The image data in a chunk is ordered by importance regarding SNR scalability and is the result of a bitplane coder. This enables SNR scaling through truncation of image data (and adjustments of GOP size information and chunk length).

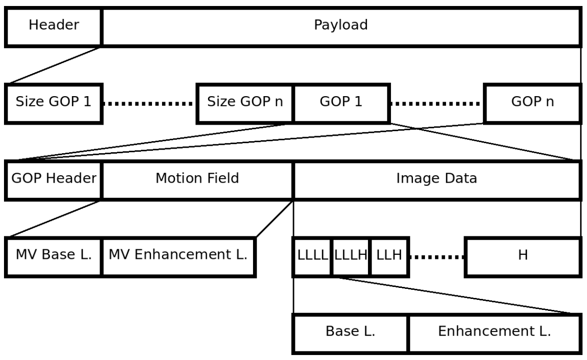


Figure 9. The layout of the MC-EZBC bitstream

Only when all headers, including chunk headers, and GOP size information are kept intact the whole bitstream can subsequently be parsed correctly, which is of ultimate importance for the preservation of scalability in the encrypted domain. Additionally the motion vector data is coded differently from the image data; the length of the motion vector data is not explicitly signalled, but it has to be determined by arithmetic decoding (until a termination marker is encountered). Thus headers and motion vector data will not be encrypted in our encryption scheme, but solely the coded image data. Since the coded image data is byte aligned we need an encryption scheme which can encrypt blocks of arbitrary length, e.g., AES in OFB mode.

Our MC-EZBC-specific encryption scheme is format-compliant in the sense that the decoder can decode the encrypted format streams (and does neither crash nor complain). This is because the arithmetic decoder has to deal with SNR scalability and thus utilizes the chunk length information to prevent misalignment. We exploit this decoder property with our encryption. In case the arithmetic decoder tries to decode to much, as would be the case when regular scaling is done, the chunk length prevents the decoder from reading data of the next chunk. Additionally, when the decoder finishes early the rest of the chunk is skipped and the decoder is properly realigned for the next chunk. This is part of the error correction of the decoder which prevents misalignment when bit flips occur in the image data during transmission.

To increase the speed of the encryption and decryption processes it is possible to encrypt only a fraction of the image data. In order to minimize the amount of data to be encrypted, while maximizing its impact on the degradation of image quality we need to encrypt the parts of the bitstream which carries the most important visual information, e.g., I-frames of low frequency bands of the wavelet decomposition. Figure 10 illustrates this by comparing frame 128 of the Container sequence to the decoding of the encrypted sequence. In this figure only the low spatial frequencies have been encrypted.

4.3. Comparison of the Format-Specific Encryption Schemes

Both format-specific encryption schemes offer efficient encryption of the coded video data, while preserving the scalability. Both schemes preserve format-compliance (i.e., the decoder does not crash). Thus both schemes are well-suited for the integration in the GVid framework.

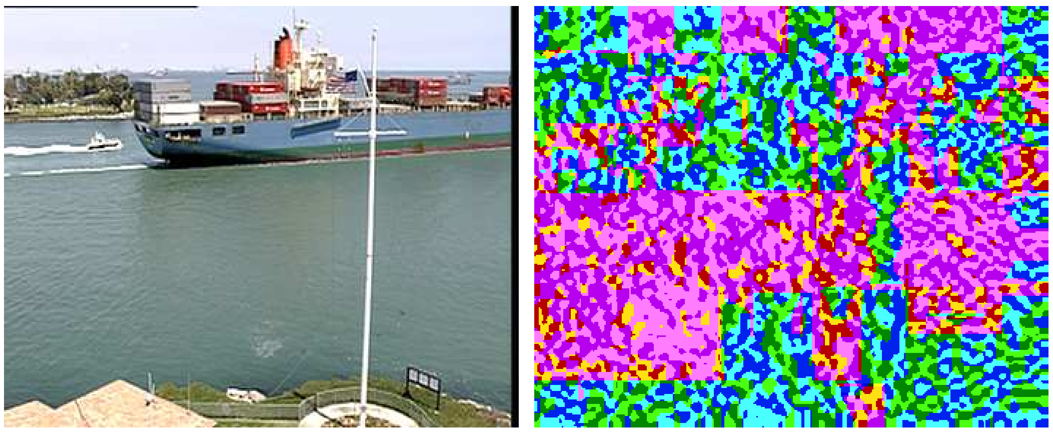

(a) decoded (b) encrypted

Figure 10. Comparison of encrypted image to the original of frame 128 from the Container sequence (low spatial frequencies)

5. Conclusion and Future Work

We have evaluated two state-of-the-art scalable video compression systems, namely H.264/SVC and MC-EZBC, for their suitability as compression codecs in the GVid framework. Their compression performance is competitive to conventional video compression systems, such as the MPEG-4 implementation Xvid. Their scalable format streams offer improved performance for multiple application scenarios. As the application of conventional security tools for confidentiality circumvent the advantages of scalable compression systems, format-specific encryption tools are necessary. For both H.264/SVC and MC-EZBC format-specific and even format-compliant encryption schemes have been proposed and discussed. Both encryption schemes meet the requirements well and can be recommended for integration in the GVid framework.

Future work will focus on the parallelization and optimization of the scalable video compression systems, as the current implementations are still not capable of real-time compression.

References

[1] FreeGLUT - The Free OpenGL Toolkit. online presentation: http://freeglut.sourceforge.net, December 2005.

[2] VTK - The Visualization Toolkit. online presentation: http://www.vtk.org, January 2006.

[3] Hermann Hellwagner, Robert Kuschnig, Thomas Stütz, and Andreas Uhl. Efficient in-network adaptation of encrypted H.264/SVC content. *Elsevier Journal on Signal Processing: Image Communication*, 24(9):740 – 758, July 2009.

[4] Shih-Ta Hsiang and J. W. Woods. Embedded video coding using invertible motion compensated 3-D subband/wavelet filter bank. *Signal Processing: Image Communication*, 16(8):705–724, May 2001.

[5] ITU-T H.264. Advanced video coding for generic audivisual services, November 2007.

[6] T. Köckerbauer, M. Polak, T. Stütz, and A. Uhl. GVid - video coding and encryption for advanced Grid visualization. In J. Volkert, T. Fahringer, D. Kranzlmüller, and W. Schreiner, editors,

Proceedings of the 1st Austrian Grid Symposium, volume 210 of *books@ocg.at*, pages 204–218, Schloss Hagenberg, Austria, 2006. Austrian Computer Society.

[7] R. Kuschnig, I. Kofler, M. Ransburg, and H. Hellwagner. Design options and comparison of in-network H.264/SVC adaptation. *Journal of Visual Communication and Image Representation*, September 2008.

[8] Steven McCanne, Van Jacobson, and Martin Vetterli. Receiver-driven layered multicast. In *SIGCOMM '96: Conference proceedings on Applications, technologies, architectures, and protocols for computer communications*, pages 117–130, New York, NY, USA, August 1996. ACM.

[9] Philip Ross. Cloud computing's killer app: Gaming. *IEEE Spectrum*, March 2009.

[10] Thomas Stütz and Andreas Uhl. Evaluation of compression codecs and selective encryption schemes for GVid. In J. Volkert, T. Fahringer, D. Kranzlmüller, and W. Schreiner, editors, *Proceedings of the 2nd Austrian Grid Symposium*, volume 221 of *books@ocg.at*, pages 28–41, Innsbruck, Austria, 2007. Austrian Computer Society.

[11] Thomas Stütz and Andreas Uhl. Format-compliant encryption of H.264/AVC and SVC. In *Proceedings of the Eighth IEEE International Symposium on Multimedia (ISM'08)*, Berkeley, CA, USA, December 2008. IEEE Computer Society.

[12] Andrew S. Tanenbaum. *Computer networks*. Prentice Hall, 3rd edition, 1996.

[13] A. Uhl and A. Pommer. *Image and Video Encryption. From Digital Rights Management to Secured Personal Communication*, volume 15 of *Advances in Information Security*. Springer-Verlag, 2005.

[14] A. Vetro, C. Christopoulos, and T. Ebrahimi. From the guest editors - Universal multimedia access. *IEEE Signal Processing Magazine*, 20(2):16 – 16, 2003.

[15] S. Wenger, M.M. Hannuksela, T. Stockhammer, M. Westerlund, and D. Singer. RTP Payload Format for H.264 Video. RFC 3984, February 2005.

[16] S. Wenger, Y. Wang, T. Schierl, and A. Eleftheriadis. RTP Payload Format for SVC Video. Internet Draft draft-ietf-avt-rtp-svc-14, September 2008.

[17] S. Wenger, Y. Wang, T. Schierl, and A. Eleftheriadis. RTP Payload Format for SVC Video. Internet Draft draft-ietf-avt-rtp-svc-17, February 2009.

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

[19] Y. Wu, A. Golwelkar, and J. W. Woods. MC-EZBC video proposal from Rensselaer Polytechnic Institute. *ISO/IEC JTC1/SC29/WG11, MPEG2004/M10569/S15*, March 2004.