Rate Allocation for Motion Compensated JPEG2000

J.C. Maturana-Espinosa*, V. González-Ruiz*, J.P. García-Ortiz*, and D. Müller†

*University of Almería
Ctra. Sacramento, s/n
Almería, 04120, Spain
vruiz@ual.es

†European Space Agency
ESTEC, PO Box 299
2200 AG Noordwijk, The Netherlands
Daniel.Mueller@esa.int

Abstract

This work proposes the video codec MCJ2K (Motion Compensated JPEG2000), which is based on Motion Compensated Temporal Filtering (MCTF) and JPEG2000 (J2K). MCJ2K exploits the temporal redundancy present in most videos, thereby increasing the rate/distortion performance, and generates a collection of temporal subbands which are compressed with J2K. MCJ2K code-streams can be managed by standard JPIP (J2K Interactive Protocol) servers.

1 Introduction

The JPEG2000 (J2K) standard [1] is a still image codec that also encompasses the compression of sequences of images using the name of Motion J2K (MJ2K). The standard relies on the J2K Interactive Protocol (JPIP) [2] to transmit J2K code-streams from between servers and clients, offering a high degree of scalability (spatial, temporal/random-access and quality). These features make J2K especially suitable for the management of video repositories and for the implementation of interactive image/video streaming services [3–5].

In particular, JPIP has proven very effective for visualisation of petabyte-scale image data of the Sun. It is being used by the Helioviewer Project[1] to enable researchers and the general public alike to explore time-dependent image data from different space-borne observatories, interactively zoom into areas of interest and play sequences of high-resolution images at various cadences.

The focus of this paper is to improve JPIP streaming systems by preserving the server side as much as possible, keeping it compatible with its most common use (serving J2K images), while offering increased compression performance when transmitting sequences of images, whose order is known a priori.

The first of the following four sections describes the most relevant works related to MCJ2K and RA (Rate Allocation) in wavelet-based video coding. Our proposal for MCJ2K coding, and how to perform RA at compression and transmission times are detailed in Section 3. Section 4 presents the results of an empirical performance evaluation, and Section 5 summarizes our findings and outlines future research.

[1] http://wiki.helioviewer.org/wiki/Main_Page

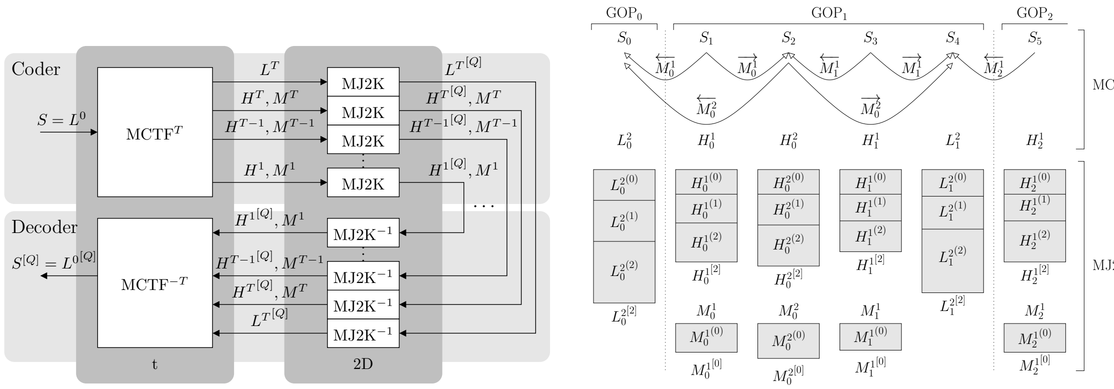
2 Background and related work

The combination of MCTF and J2K has been proposed previously. Secker et al. used these techniques to create LIMAT [6], and a similar approach was designed in [7] and extended in [8] by André et al. In these works, however, the selection of the slopes for determining the optimal truncation points (the quality layers) in the rate-distortion (RD) curve of the texture subbands is not optimized. The RA among the temporal subbands is determined by using RD Optimization (RDO), assuming that the addition of quality layers monotonically decreases the distortion[2]. The order in which the quality layers should be sent in a progressive transmission of the sequence of images has not been optimized in [7] and [8].

In [9], Cohen et al. proposed two MC-based J2K codecs. The first one is a pyramid codec with a MCTF step on each spatial level, and a closed-loop coding structure, similar to H.264/SVC and HEVC. The second codec is open-loop, similar to MCJ2K, but the authors do not tackle the problem of RA among temporal texture subbands and motion information.

Barbarien et al. in [10] also provided some interesting ideas to perform optimal RA at compression-time. Before using the 2D-DWT, all the residue coefficients resulting of the MCTF process are multiplied by a scaling factor to approximate MCTF to a orthogonal transform. As in [8], authors also propose an optimal RA among motion and texture data based on Lagrangian RDO, considering that the distortions are additive (something that can be erroneous because MCTF is not linear in general). Such optimization minimizes the distortion for a known bit-rate, but not for any possible bit-rate (note that when transmitting an image or a sequence of images, such bit-rates established at compression-time might not be met at decompression-time).

As it has been pointed out, RA can be performed also at decompression-time. In this case, it can be implemented by the sender (server), receivers (clients) or both of them. FAST, proposed by Aulí-Llinàs et al. in [11] and improved by Jimenez-Rodriguez et al. in [12], is a sender-driven RA algorithm for MJ2K sequences. Another interesting MJ2K/MCJ2K sender-driven RA proposal was introduced by Naman et al., which uses Conditional Replenishment (CR) [13] and Motion Compensated CR [14]. In Naman's proposals, an "oracle" (although other realistic policies are considered as well) server sends those J2K packets related to the regions that the clients should refresh to optimize the quality of the video after considering bandwidth constraints. Unfortunately, none of these proposals are fully JPEG2000 compliant at the server side, because some kind of non-J2K-standard logic must be used.



(a) Codec architecture. (b) Example of a code-stream.

Figure 1: MCJ2K is a two stages codec (see Subfig. a): a temporal filtering (MCTF) and a compression of sequence of images (MJ2K). The resulting code-stream (see Subfig. b) is a collection of compressed texture $\{L^T, \{H^t; 1 \leq t \leq T\}\}$ and motion $\{M^t; 1 \leq t \leq T\}$ subbands. In this example, the number of levels of MCTF is $T = 2$.

3 Motion compensated JPEG2000 video coding

3.1 Codec overview

Our implementation of MCJ2K is an open-loop t+2D structure (see Fig. 1a), where the temporal filtering (a 1/3 1D-DWT) is applied first, and the 2D-DWT (provided by the J2K codecs) is applied to the motion compensated frames. The "t" stage corresponds to a T-levels MCTF process (denoted in the figure by MCTF^T) which exploits the temporal redundancy of the video, and the "2D" stage corresponds to a bank of MJ2K compressors, that basically exploit the spatial redundancy in each image (most of which are prediction residues) and perform entropy coding. In the figure, S represents to the original video and $S^{[Q]}$ to a lossy decompressed version of S. MCTF^T generates $T+1$ temporal texture subbands $\{L^T, \{H^t; 1 \leq t \leq T\}\}$ and T motion "subbands" $\{M^t; 1 \leq t \leq T\}$.

Fig. 1b shows an example of the organization of an MCJ2K code-stream. In this, six images $S_0, \cdots S_5$ have been compressed using a GOP size $G = 4$ (i.e., $T = 2$ ($G = 2^T$)), except for the first GOP that always has only one image. MCTF^2 transforms the input video S into 5 subbands: $\{L^2, H^2, M^2, H^1, M^1\}$ (we will refer also to the fields of bidirectional motion vectors as "subbands"). L^2 is the low-frequency texture subband and represents the temporal low-frequency components of S. $\{H^2, H^1\}$ are the high-frequency texture subbands that contain the temporal high-frequency components of S. $\{M^2, M^1\}$ are the motion subbands, and store a description of the motion detected in S after using logarithmic-search [15] bidirectional block-based ME [16] with constant block-size B, and search area of A pixels (see[3] for implementation details). In Fig. 1b, arrows over motion fields indicate the direction of the ME process.

[2]We have found in our experiments, using a similar codec, that this is not true in most videos due to a non-linear behavior of the ME (Motion Estimation) process.

[3]`https://github.com/vicente-gonzalez-ruiz/QSVC/`

MCTF^1		MCTF^2		MCTF^3		MCTF^4		MCTF^5		MCTF^6		MCTF^7	
H^t	a_t	H^t	a_t	H^t	a_t	H^t	a_t	H^t	a_t	H^t	a_t	H^t	a_t
H^1	1.246	H^2	1.250	H^3	1.160	H^4	1.088	H^5	1.046	H^6	1.023	H^7	1.012
		H^1	1.865	H^2	2.122	H^3	2.130	H^4	2.079	H^5	2.043	H^6	2.023
				H^1	3.167	H^2	3.888	H^3	4.061	H^4	4.063	H^5	4.039
						H^1	5.802	H^2	7.431	H^3	7.936	H^4	8.031
								H^1	11.089	H^2	14.522	H^3	15.688
										H^1	21.669	H^2	28.707
												H^1	42.835

Table 1: L_2-norm energy attenuation values for the images of each temporal subband in our implementation of the MCTF transform.

More details about how MCTF has been implemented can be found in [17].

Finally, in the MJ2K stage, each image of each texture subband is J2K-compressed, producing a variable-length layered (in the example of Fig. 1b, $Q = 3$ quality layers) code-stream. $I^{(y)}$ represents the y-th quality layer of the compressed representation of the image I, and $I^{[y]}$ the reconstruction of I using $y+1$ quality layers. Motion data is losslessly J2K-compressed as a sequence of 4-component (2 vectors per macro-block) single-layer ($Q = 1$) images. In our notation, the first quality layer, in the layers decoding order, is referenced by 0. The q-th quality layer of a temporal subband s is called the subband-layer $s^{(q)}$. For example, in the figure, the subband-layer $L^{2(0)} = \{L_0^{2(0)}, L_1^{2(0)}\}$.

3.2 Compression time RA optimization

In MCJ2K, the quality layers of each compressed image (in the MCTF domain) are determined by a set $\{\lambda_0^s, \cdots, \lambda_{Q-1}^s\}$ of Q RD-slopes, being s the temporal subband of the image. Thus, taking into account that the layers of each image correspond to truncation points of the convex-hull of the RD curve of the corresponding image, each subband-layer will correspond to an optimal truncation point of the RD curve of the corresponding temporal subband. An interesting consequence of this is that the quality of the images of a subband can be considered constant if the code-stream is truncated in a subband-layer [18]. Consequently, using the same RD-slopes for each image of each subband (and among subbands), if the processes MCTF+J2K were unitary [10], Q optimal truncation points per subband would be available when the same number of subband-layers are decoded.

Our implementation of MCTF is not unitary. For example, in MCTF^2, the image L_0^2 is 1.25 times more "important" than the image H_0^2, and 1.865 times more important than the images H_0^1 and H_1^1. How much energy an image of a given temporal subband must contribute to the code-stream to approximate MCTF to an unitary transform has been represented in form of attenuation factors in Tab. 1. Each element of this table has been computed taking into consideration the energy gain of the wavelet basis functions of the synthesis filters which correspond to that subband and number of MCTF levels [19]. These factors are empirical, specifically determined for the 1/3 DWT implemented in our codec.

These attenuation factors can be used in two different ways: (1) multiplying each (pixel of each) image by the corresponding factor, and quantizing by a constant value (this is the traditional approach, used for example in [10]), or (2) controlling the RD-slopes used in the MJ2K stage when the quality layers are defined (this procedure is the used in our proposal). RD-slopes used by MCJ2K are defined by

$$\lambda_0^{H^t} = \lambda_0^{L^T} + \frac{\Delta d}{J} a_t. \quad (1)$$

In this formula, $\lambda_0^{L^T}$ is a RD-slope specified by the user to set the quality that it will be provided by the subband-layer $L^{T^{(0)}}$ (which is equivalent to say, the quality of $S^{[0]}$ because no low-pass synthesis filter has been used in our implementation of MCTF), Δd and J are two constants which depend on the implementation of the J2K codec, and a_t is the attenuation factor for the gain of the temporal subband H^t in relation to L^T. For example, in Kakadu[4], $\Delta d = 256$ (RD-)slope-steps (a value considered suitable to avoid the generation of empty J2K packets) and $J = \sqrt{2}$ (the expected ratio increment in length of the code-stream when the RD-slope is decremented in Δd).

Eq. 1 establishes the relation among the expected distortion generated in each temporal subband H^t, as a function of the first slope $\lambda_0^{H^t}$ used for that subband, which depends on $\lambda_0^{L^T}$. In MCJ2K, the rest of quality layers are defined by

$$\lambda_q^s = \lambda_0^s + \frac{\lambda_0^s - \lambda_{Q-1}^s}{Q} q, \quad (2)$$

where $s \in \{L^T, H^T, H^{T-1}, \cdots, H^1\}$, $1 \leq q < Q$, and λ_{Q-1}^s is the slope which determines the maximum quality of s obtained when all the layers of s have been decompressed.

3.3 Post-compression RA optimization

The code-stream of each image of each subband follows the LRCP progression [20] (this is also true for the motion subbands, although in this case $Q = 1$). At the subband-layers level, MCJ2K sorts texture subbands following the progression

$$\begin{array}{lllll} L^{T^{(0)}}, & H^{T^{(0)}}, & H^{T-1^{(0)}}, & \cdots, & H^{1^{(0)}}, \\ L^{T^{(1)}}, & H^{T^{(1)}}, & H^{T-1^{(1)}}, & \cdots, & H^{1^{(1)}}, \\ \vdots & \vdots & \vdots & & \vdots \\ L^{T^{(Q-1)}}, & H^{T^{(Q-1)}}, & H^{T-1^{(Q-1)}}, & \cdots, & H^{1^{(Q-1)}}, \end{array} \quad (3)$$

of subband-layers, that has been called Progressive Transmission by Layers (PTL). On the other hand, motion subband-layers use the progression

$$M^T, M^{T-1}, \cdots, M^1. \quad (4)$$

Eqs. 1 and 2 provide optimal RA in the subband-layers of texture. In the case of the motion subband-layers, a RA is not needed because they are losslessly encoded. However, it needs to be determined how to interleave the sequence of T motion subband-layers (Eq. 4) into the sequence of QT texture subband-layers (Eq. 3).

[4] `http://kakadusoftware.com`

3.3.1 Heuristic PTL (HPTL)

HPTL "transmits" a (subband-layer of a) motion subband when *enough* subband-layers of the corresponding temporal subband have been transmitted previously. This heuristic corresponds to the fact that, in order to improve the quality using motion information, a sufficient amount of texture should have been decoded previously. The number of subband-layers of texture of a subband that should be sent before the corresponding subband-layer of motion will be analyzed in the following section. HPTL takes $\mathcal{O}(1)$ time.

3.3.2 Optimized PTL (OPTL)

With a computation time proportional to $\mathcal{O}(T)$, OPTL performs a search to find the best place to insert the T subband-layers of motion in between the texture subbands, assuming that

$$\lambda^{M^T} \geq \lambda^{M^{T-1}} \geq \cdots \geq \lambda^{M^1}, \tag{5}$$

which means that the contribution of the motion subbands to the quality of the reconstruction decreases with the index of the subband. In OPTL, a motion subband-layer M^t is inserted between (motion or texture) subband-layers a and b when the contribution of M^t to the quality of the corresponding GOP is smaller or equal than the contribution of a, but higher or equal than the contribution of b. In other words, if

$$\lambda^a \geq \lambda^{M^t} \geq \lambda^b, \tag{6}$$

M^t will be inserted between a and b.

3.3.3 Full Search Optimization (FSO)

The contribution of a temporal/motion subband to the quality of the reconstruction depends on the performance of the ME process, which shows a non-linear behavior. For this reason, with the aim of measuring the performance of the previously proposed RA algorithms, we have proposed a brute-force subband-layers optimizer. This so-called Full Search Optimization (FSO) extends the idea used in OPTL to check which is the best (motion or texture) subband-layer to transmit at any time of the progressive reconstruction, without establishing any previous condition (except that, in each GOP, $L^{T^{(0)}}$ must be transmitted first).

FSO is an iterative algorithm that, for each GOP, sorts the subband-layers by their contribution slopes. In each iteration, FSO performs up to $2T$ reconstructions (one for each remaining subband-layer that has not been selected in a previous iteration), compares them to the original GOP, and selects the subband-layer that minimizes the distortion. Thus, FSO takes $\mathcal{O}((QT)^2)$ time.

We point out that FSO generates RD curves in which all the truncation points (one for each of the $T + TQ$ subband-layers) are optimal, i.e., FSO should allow to reconstruct progressively the video with the highest quality, even if the RD-slopes have not been optimized. In this case, the RD-slopes define where the optimal truncation points of the RD curve are located, but the curve does not depend on the slopes.

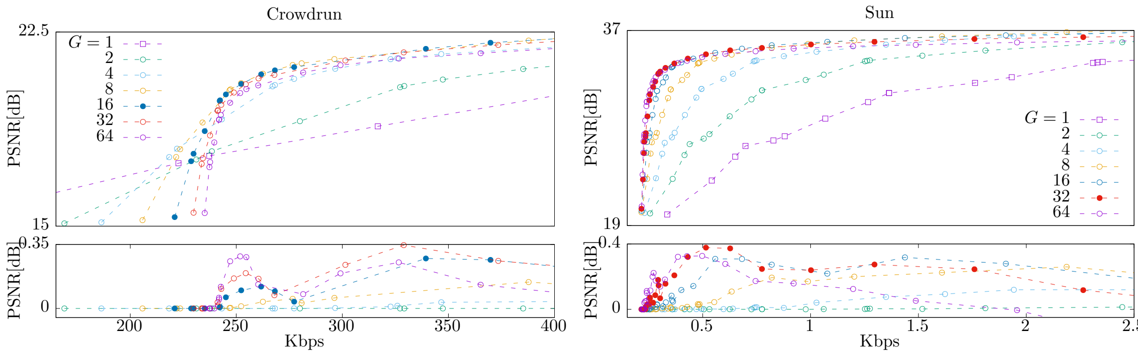


Figure 2: (Upper panel) Compression performance of MCJ2K for different GOP sizes using gains. (Lower panel) difference on distortion by using the same RD slope for all texture subbands (positive values indicate that by using the attenuation values of Table 1 the distortion is decreased).

4 Evaluation

The performance of MCJ2K has been evaluated for several videos, and compared to previous proposals.

Several test videos has been used for our evaluation. Results are shown for (1) `Crowdrun`[5] (YUV 4:2:0, 1920×1080 pixels, 50 Hz), a high-resolution video with a high degree of movement, and (2) `Sun`[6] (monochromatic, 4096×4096 pixels, 1/30 Hz) a sequence of images of the Sun with only small-scale frame-to-frame motion(which is, however, complex to predict due to the random motions of magnetic flux concentrations in the Sun's photosphere). In all experiments, 129 images have been compressed.

4.1 MCTF compression gain

Results shown in Fig. 2 measure the compression ratio of MCJ2K compared to MJ2K. The test videos have been compressed and decompressed (no progressive decoding has been used) for several $\lambda_{LT}^{(0)}$ values, using $Q = 1$ quality layers, and varying GOP length G. The block-size B used in the ME process has been 64 for `Crowdrun`, and 128 for `Sun`. The initial search area (used between adjacent images) is $A = 4$ in all tests. All these parameters have been selected ad-hoc.

As can be seen in the figure, except for very low bit-rates (where the bit-rate necessary to encode the motion information becomes meaningful as a consequence of using a non-scalable representation of this information), MCJ2K (curves with $G > 1$) always reaches better RD performance than MJ2K (curve for $G = 1$), and the improvement depends on G. We conclude from this that the compression gain provided by MCTF is substantial in all videos.

Fig. 2 also shows the compression gains provided by the use of compression time RA optimization (see Sec. 3.2, low panel). These gains are positive for most of

[5] `ftp://vqeg.its.bldrdoc.gov/HDTV/SVT_MultiFormat/`

[6] `http://helioviewer.org/jp2/AIA/2015/06/01/131/`

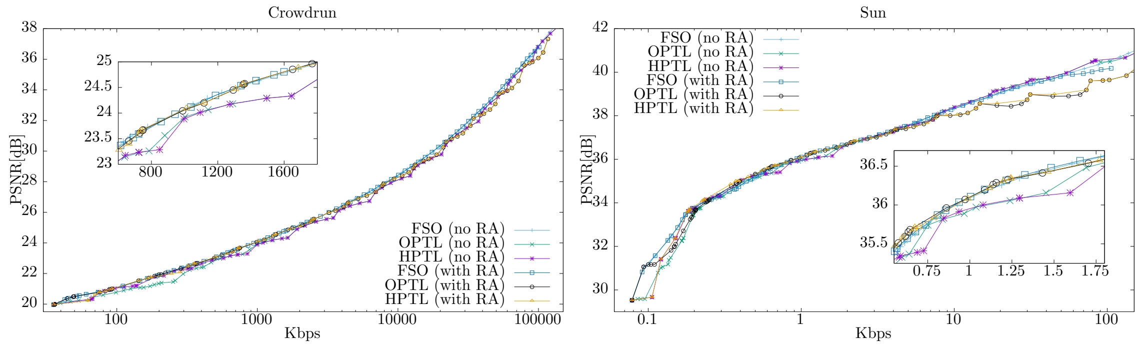


Figure 3: Progressive decoding of the videos using the proposed post-compression RA algorithms. Both, OPTL and HPTL perform close to FSO at most bit-rates.

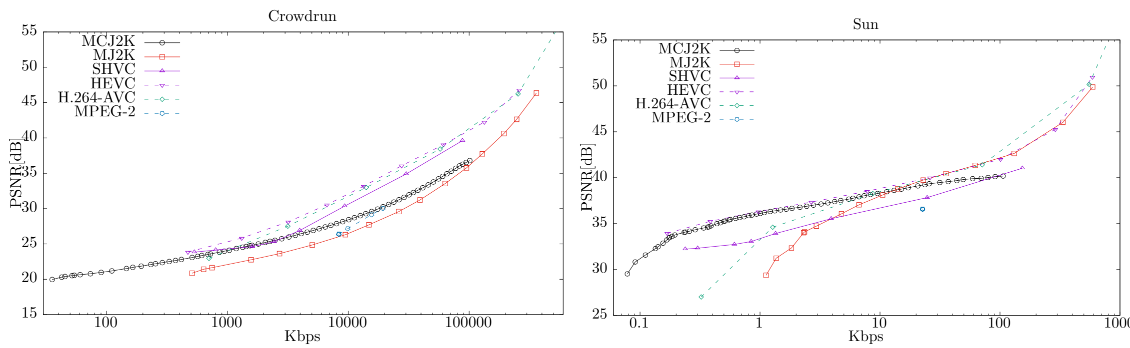


Figure 4: Video codecs comparison. MCJ2K offers the highest number of truncation points and provides a compression-ratio performance close SHVC.

bit-rates, depend on G and the temporal correlation in the video (the higher the correlation and GOP size, the higher the gain), and can exhibit a "erratic" behavior as a consequence of the non-linearity of the ME/MC process (the difficult-to-predict the movement in the video, the higher the non-linearity). For the rest of the experimentation, we have used $G = 16$ for `Crowdrun`, and $G = 32$ for `Sun`.

4.2 Performance comparison of post-compression RA algorithms

HPTL, OPTL and FSO have been evaluated in terms of RD. In this evaluation, a key parameter is Q because it controls the accuracy of the RA algorithms.

Fig. 3 shows the performance of the proposed RA algorithms. For each algorithm, two curves have been considered, using and not using compression time RA optimization (see Sec. 3.2). Our results show that: (1) as expected, slopes optimization has a positive impact on HPTL and OPTL, and no influences FSO, (2) the order of the subband-layers has a high impact on the RD curve, specially at low bit-rates, and (3) FSO is the best algorithm in terms of RD and HPTL the worst. $Q = 16$ quality layers has been used for all experiments.

4.3 MCJ2K vs other video codecs

Fig. 4 shows the compression performance of MCJ2K (using OPTL and optimized slopes) and other MCTF-based/standard video codecs (dashed lines represent non-embedded decoding, while solid lines, progressive decoding). Our results show that, especially at low bit-rates, MCJ2K provides better compression ratios than MJ2K. Compared with non-scalable video codecs, such as HEVC[7] or AVC[8], MCJ2K needs approximately 50% more of data to achieve the same quality, but this difference is smaller when it is compared with SHVC[9] where MCJ2K produces better results for some of the test videos. In the case of MPEG-2[10] (a codec that implements a ME/MC scheme similar to the used in MCJ2K), MCJ2K outperforms it consistently. Compared to other MCTF-based codecs ([10] and [7]), MCJ2K achieves better results.

5 Conclusions and future work

MCJ2K is an efficient and fully JPIP-compatible codec for interactive video streaming, and offers an alternative to MJ2K if some degree of temporal scalability can be sacrificed. MCJ2K's compression ratio is similar to other scalable video coding standards such as SHVC.

RD performance in MCJ2K can be significantly improved in three main aspects: (1) a more accurate ME/MC scheme, (2) the use of spatial/quality scalability when encoding the motion data and the development of new RA algorithms, and (3) the use of encoding schemes where the motion information can be estimated at the decoder (to avoid sending it as a part of the code-stream). This can be carried out in those contexts where the large-scale motion is predictable, such as image sequences of the Sun, whose rotation rate is stable and well known.

References

[1] ISO, "Information Technology - JPEG 2000 Image Coding System - Core Coding System," ISO/IEC 15444-1:2004, September 2004.

[2] ITU, "Information Technology - JPEG 2000 Image Coding System: Interactivity Tools, APIs and Protocols," `http://www.itu.int/rec/T-REC-T.808-200501-I`, 2005.

[3] A. Bilgin and M.W. Marcellin, "JPEG2000 for digital cinema," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, may 2006, pp. 4 pp. – 3881.

[4] D. Müller, B. Fleck, G. Dimitoglou, B. W. Caplins, D. E. Amadigwe, J. P. Garcia Ortiz, A. Alexanderian B. Wamsler, V. Keith Hughitt, and J. Ireland, "JHelioviewer: Visualizing large sets of solar images using JPEG 2000," *Computing in Science and Engineering*, vol. 11, no. 5, pp. 38–47, September 2009.

[5] D. Müller, B. Nicula, S. Felix, F. Verstringe, B. Bourgoignie, A. Csillaghy, D. Berghmans, P. Jiggens, JP. García-Ortiz, J. Ireland, S. Zahniy, and B. Fleck, "JHelioviewer-Time-dependent 3D visualisation of solar and heliospheric data," *Astronomy & Astrophysics*, vol. 606, pp. A10, 2017.

[7] `https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware` using `trunk/cfg/encode_randomaccess_main.cfg`

[8] `http://www.videolan.org/developers/x264.html` using `--profile high --preset placebo --tune psnr`

[9] `https://hevc.hhi.fraunhofer.de/svn/svn_SHVCSoftware/` using `branches/SHM-dev/cfg/encoder_randomaccess_scalable.cfg` and `branches/SHM-dev/cfg/misc/layers8.cfg`

[10] `http://linux.die.net/man/1/mpeg2enc`

[6] A. Secker and D. Taubman, "Lifting-based Invertible Motion Adaptive Transform (LIMAT) framework for highly scalable video compression," *IEEE Transactions on Image Processing*, vol. 12, no. 12, pp. 1530–1542, Dec 2003.

[7] T. Andre, M. Cagnazzo, M. Antonini, and M. Barlaud, "JPEG2000-compatible scalable scheme for wavelet-based video coding," *EURASIP Journal on Image and Video Processing*, vol. 2007, pp. 1–11, 2007, doi:10.1155/2007/30852.

[8] M. Cagnazzo, F. Castaldo, T. Andre, M. Antonini, and M. Barlaud, "Optimal motion estimation for wavelet motion compensated video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 7, pp. 907–911, July 2007.

[9] R.A. Cohen and J.W. Woods, "Resolution scalable motion-compensated JPEG 2000," in *Proc. of the International Conference on Digital Signal Processing (DSP)*, July 2007, pp. 459 – 462.

[10] J. Barbarien, A. Munteanu, F. Verdicchio, Y. Andreopoulos, J. Cornelis, and P. Schelkens, "Motion and texture rate-allocation for prediction-based scalable motion-vector coding," *Signal Processing: Image Communication*, vol. 20, pp. 315342, 2005.

[11] F. Aulí-Llinàs, A. Bilgin, and M.W. Marcellin, "FAST Rate Allocation Through Steepest Descent for JPEG2000 video transmission," *Image Processing, IEEE Transactions on*, vol. 20, no. 4, pp. 1166–1173, 2011.

[12] L. Jiménez-Rodríguez, F. Aulí-Llinàs, and M.W. Marcellin, "FAST rate allocation for JPEG2000 video transmission over time-varying channels," *IEEE Transactions on Multimedia*, vol. 15, pp. 15–26, January 2013.

[13] A.T. Naman and D. Taubman, "JPEG2000-based Scalable Interactive Video (JSIV)," *Image Processing, IEEE Transactions on*, vol. 20, pp. 1435– 1449, May 2011.

[14] A.T. Naman and D. Taubman, "JPEG2000-based Scalable Interactive Video (JSIV) with motion compensation," *Image Processing, IEEE Transactions on*, vol. 20, no. 9, pp. 2650 –2663, sept. 2011.

[15] C. Hsieh and Y. Liu, "Fast search algorihtms for vector quantization of images using multiple triangle inequalities and wavelet transform," *IEEE Trans. on Image Proc.*, vol. 9, no. 3, pp. 321–328, 2000.

[16] S.W. Wu and A. Gersho, "Joint estimation of forward/backward motion vectors for MPEG interpolative prediction," 1993, vol. 1, p. 14.3.214.3.2.

[17] A. Secker and D. Taubman, "Motion-compensated highly scalable video compression using an adaptive 3D wavelet transform based on lifting," *Proceedings of the IEEE International Conference on Image Processing. Thessaloniki, Greece*, vol. 2, pp. 10291032, 2001.

[18] J.C. Dagher, A. Bilgin, and M.W. Marcellin, "Resource-constrained rate control for Motion JPEG2000," *Image Processing, IEEE Transactions on*, vol. 12, no. 12, pp. 1522–1529, 2003.

[19] R. Xiong, J. Xu, F. Wu, and S. Li, "Adaptive MCTF based on correlation noise model for SNR scalable video coding," in *IEEE International Conference on Multimedia and Expo (ICME)*, July 2006, pp. 1865–1868.

[20] D. Taubman and M. Marcellin, *JPEG2000 image compression fundamentals, standards and practice*, vol. 642, Springer Science & Business Media, 2012.