A Tutorial on Image Compression for Optical Space Imaging Systems

Ian Blanes, Enrico Magli, Joan Serra-Sagristà

***Abstract*—Public policies and private initiatives share the will to explore outer space and to monitor the Earth from space sensors. Recent years have seen an increased number of space missions, while the sensors on board aircrafts or spacecrafts have also significantly improved their acquisition capabilities. Given this huge volume of remote sensing data and the detailed characteristics of the acquired images, a data compression process is in order to allow as large a transmission rate as possible.**

In this paper we provide an overview of several standards for remote sensing data compression, notably of those recently approved by the Consultative Committee for Space Data Systems, although the use of other ISO/IEC image coding standards is also dealt with. Discussion embraces both mono band and multi band compression, and lossless, lossy and near-lossless compression.

Illustrative results are reported for a set of AVIRIS and Hyperion images, indicating that exploiting the spectral correlation —either in prediction-based or in transform-based schemes— is paramount to achieve improved coding performance.

***Index Terms*—image compression, multispectral and hyperspectral images, CCSDS, prediction, transform coding, DPCM, rate control.**

I. Introduction

Image compression has been studied for a long time now. The development of new compression techniques had led to plenty of image compression standards such as JPEG and JPEG2000 that are routinely used in many consumer applications. Alongside, the image compression problem has become increasingly more important for a particular class of images, namely those acquired by remote satellites with on-board cameras. Such cameras can be of very different kinds, as they can acquire visual information globally in the visible spectrum (mono band images) or specifically at different wavelengths (multi band images). How many wavelengths are sampled dictates the spectral resolution, and increasing degrees of such resolution, from a few to several tens, hundreds or thousands of bands, give rise to multispectral, hyperspectral or ultraspectral images. It is clear that such a large amount of data can match the available downlink bandwidth only if image compression techniques are employed. This has spurred a lot of research aimed at developing image compression algorithms for on-board compression of remote sensing images. At the same time, when the images are received by the ground stations and properly processed (*e.g.*, calibrated and rectified), they have to be delivered to the final users. This is typically done via web browsing, whereby the compression algorithm must be able to handle very large images in a way that meets the requirements of the final users.

Ian Blanes and Joan Serra-Sagristà are with Department of Information and Communications Engineering, Universitat Autònoma de Barcelona, Spain. Email: ian.blanes@uab.cat, joan.serra@uab.cat. Enrico Magli is with Dept. of Electronics and Telecommunications, Politecnico di Torino, Italy. Email: enrico.magli@polito.it

IB and JSS acknowledge partial funding from FEDER, the Spanish Government and the Catalan Government under Grants TIN2012-38102-C03-03 and 2014SGR-691. EM acknowledges partial funding from ESA-ESTEC under grant 107104.

A. Motivation

Image compression techniques can be roughly divided into *lossless* and *lossy*. Lossless compression algorithms are those where the image reconstructed after the decoding process is perfectly identical to the original image. This is generally a highly desirable feature, as it guarantees that the compression process is perfectly transparent in terms of image quality. However, lossless compression typically achieves a limited compression ratio. On the other hand, lossy compression algorithms introduce some distortion between the reconstructed and the original image, since not all the information contained in the original image is encoded into the compressed file. This allows to achieve significant compression gains, all the more so as the distortion becomes larger. The lossy compression process raises the issue of ensuring that the quality of the reconstructed image is still adequate for the intended scientific use. Therefore, usually only small amounts of distortion are tolerated; however, this allows to obtain much higher compression ratios, and is becoming an increasingly popular approach in remote sensing missions. The tolerated amount of distortion may depend on several factors. In general, if the distortion is "small" with respect to the inherent image acquisition noise, then its effect is likely going to be negligible. However, in some specific applications it is found that even higher noise levels are acceptable, and will not seriously reduce the application performance. In mono-band images, it is particularly important to avoid generating visual artifacts, whereas in multi-band images it is desirable that image analysis techniques (e.g., classification, anomaly detection and so forth) produce almost the same results as if they were applied on the original image. This concept is further discussed in Sec. IV.

A particular kind of lossy compression, termed *near-lossless*, is based on the paradigm of controlling image quality by imposing the constraint that, no matter what operations are performed by the compression algorithm, the absolute maximum error between each pixel of the original and reconstructed image is bounded by some user-defined value. This is a simple way to avoid occasional large errors that could be harmful for image interpretation, *e.g.*, triggering false detection of anomalies. The theory of data and image compression is well established and its review is out of the scope of this paper. The interested reader can refer to one of the many available books for more information, *e.g.*, [1], [2]. Even in the specific case of remote sensing image compression, there is a large number of available algorithms that have

been developed over the years. The purpose of this paper is not to provide a survey of the available techniques, as this can already be found in existing material, *e.g.*, [3], [4], [5]. Rather, we specifically focus on algorithms that have been developed in the framework of the Consultative Committee for Space Data Systems (CCSDS) [6]. In the last few years, this committee has defined several techniques for lossless data compression, lossless and lossy compression of monoband images, lossless compression of multidimensional images, and is currently working towards the definition of lossy compression algorithms for multicomponent images such as multispectral and hyperspectral images. These algorithms are defined for application to on-board compression, where the specific requirements in terms of memory, computational capability and available hardware are typically the main design driver. While describing these algorithms, we will also touch upon the JPEG 2000 family of coding standards [7], which are more amenable for use at the ground segment. In short, the objective of this paper is to provide a tutorial introduction to the most recent image compression standards for space applications, and give an outlook to future developments aimed at filling feature or functionality gaps that are not covered by the existing standards.

B. Compression requirements

As has been said, designing a compression algorithm for on-board use must take into account a number of constraints dictated by the structure of the on-board processing system. Fig. 1 shows the operational mode of an on-board sensor on an airplane.

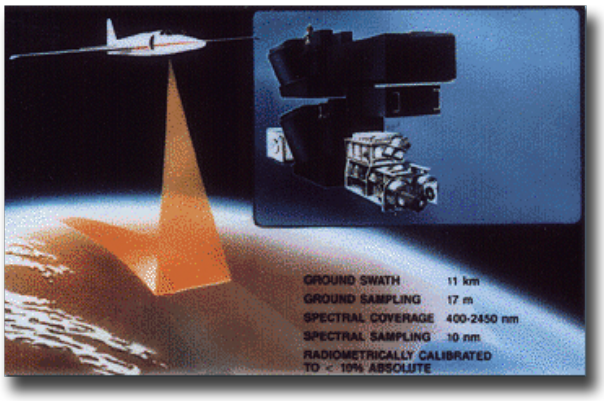


Fig. 1: AVIRIS operation mode (courtesy NASA/JPL-Caltech).

Such on-board processing system typically has limited computational capabilities due to power, size and radiation hardening issues. Therefore, **low encoder complexity** is highly desirable. In addition to this, the encoder design must be such that the algorithm should be easy to implement in the available hardware. The typical hardware for on-board processing has evolved over the years and still is. The usual preferred choice is a field-programmable gate array (FPGA), which requires a description of the algorithm in a hardware description language such as VHDL. Therefore, the algorithm should not employ operations that are difficult to map to a VHDL description.

Another requirements is the ability to properly **handle raw data**. What this means is that an on-board compression algorithm will have as input the original digital numbers generated by the sensor, prior to any processing other than binning. Therefore, all sensors imperfections, such as noise, striping, misregistration and so on, which are typically corrected at the ground segment, will be present in the image. This implies that such imperfections will yield a loss in compression efficiency, and this loss could become rather large unless the algorithms are somewhat robust. *E.g.*, as will be seen in Section II, the CCSDS-123 recommendation defines prediction modes that are robust towards striping noise.

Moreover, it is known that image transmission from the remote platform to the ground station can undergo errors or packet losses. This phenomenon is rather strong for deep space missions because of the very large distance, and very weak for Earth observation satellites. Nevertheless, since even one packet loss may render the compressed image file completely undecodable, another requirement lies in the provision of some kind of **error resilience**, *i.e.*, the image decoding process should not break down completely, nor excessively impair the data, upon occurrence of occasional errors or packet losses. While there are a lot of sophisticated error resilience techniques available for image compression, for Earth observation the most typical approach is to reset the compression algorithm every once in a while, so as to create a set of independently decodable image units, thereby limiting the scope of any information loss due to communication errors.

It should be noted that the requirements above mostly apply to on-board compression, whereas another typical application of compression algorithms is at the ground segment. In this case, however, the requirements are rather different, as there is no significant limitation of computational power and memory to perform compression, and the objective is also different, namely to distribute the images to the final users. Since in this case most of the communications occur via the TCP/IP protocol, which performs retransmissions until the compressed image file has been received without errors, error resilience is also less important. On the other hand, some specific issues arise as a consequence of the way the images are accessed by the users. In particular, users typically connect to search engines via web browsers or specific software, which allows them to browse the images and their metadata to facilitate the choice of the products of interest. Given the very large size of these images, this **remote browsing process** is only possible if a flexible compression algorithm is employed, avoiding to send the complete compressed file, since this has a huge size, but sending subunits of this file that can be employed at the decoder to reconstruct specific regions of interest that the user has selected dragging a box in a preview image. Moreover, in order to speed up the image selection process, it is important that the compression algorithm offers **scalability**, *i.e.*, it allows sending first a low-quality version of the subimage of interest, and then one or more quality improvement layers, so that delays can be avoided if the user performs an early rejection of a subimage they have chosen, and moves on to another subimage.

C. Structure of the paper

As has been said, this paper is mostly concerned with CCSDS standards, which have been designed for on-board

compression, while we will briefly discuss the JPEG2000 standard that is more suitable for compression at the ground segment. In particular, we will first cover the two most recent CCSDS standards, namely the new CCSDS-123 standard for lossless compression of multispectral and hyperspectral images (Section II), and the CCSDS-122 standard for lossless and lossy compression of monoband images (Section III). These two standards cover a lot of possible applications; however, there is still no standard defined for lossy compression of multispectral and hyperspectral images. In Section IV we take a more general approach, and discuss a set of possible modifications to the CCSDS-123 standard, including near-lossless, lossy compression and rate control. Unlike Section II and III, which aim at describing the main steps of the existing standards in a tutorial way, Section IV has a more general scope and reviews more about the underlying theory, highlighting open research issues and possible solutions, as well as other possible approaches. It has to be mentioned that, while this paper describes extensions of CCSDS-123, other options are also being considered, such as a three-dimensional extension of CCSDS-122.

D. Dataset

Throughout the paper we will provide compression results aimed at assessing the performance of the techniques presented in the paper. To this end, a set of hyperspectral images will be employed, as shown in Table I. These are AVIRIS[1,2] and Hyperion[2,3] hyperspectral images, publicly available for download. Technical names and sizes are provided in Table I, along with the zero-order entropy of the images. All images are 16 bits per pixel per band (bpppb), except for Hawaii and Maine that are 12 bpppb. Uncalibrated images are stored as unsigned integers, whereas calibrated images are stored as signed integers. Considered calibrated images are radiance images.

For both AVIRIS and Hyperion images, the multi-component volume can be used to produce a spectral signature for each of the spatial pixels. These spectral signatures are then employed in remote sensing analysis as classification or anomaly detection. Fig. 2 illustrates one AVIRIS image and a spectral signature for some pixels.

II. Lossless data compression

As pointed out in Section I, we summarise here the most recent standard for remote sensing data compression. The CCSDS created the Multispectral Hyperspectral Data Compression (MHDC) Working Group in June 2007 to issue a Recommended Standard for Multi- and Hyperspectral image compression. The motivation for creating this Working Group is explained in this excerpt from [8]: *On-board data compression is needed to make full use of limited spacecraft resources like data storage and downlink capacity. Multispectral & hyperspectral images can occupy enormous data volumes, and so compression algorithms specifically designed to exploit the three-dimensional structure of such images can provide tremendous benefit to space missions.*

[1]http://avirir.jpl.nasa.gov
[2]http://compression.jpl.nasa.gov/hyperspectral/
[3]http://earthexplorer.usgs.gov

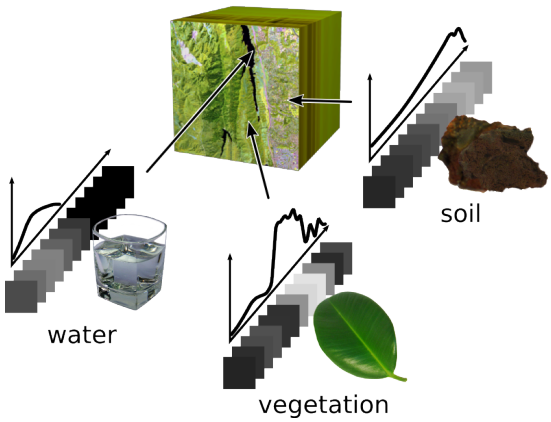


Fig. 2: AVIRIS Volume and Spectral Signature at various locations.

Thanks to the work of several member space agencies, observer space agencies and committed participants, the Working Group was able to deliver CCSDS-123 standard [9] in May 2012. The standard is intended for on-board lossless compression of multi- and hyperspectral images. The coding technique is built upon Fast Lossless (FL) compression algorithm [10] and is able to provide state-of-the-art coding performance for a large collection of remote sensing sensors. As it is oriented towards on-board operation, *i.e.*, in resource-constrained scenarios, the design was carefully conceived to require a very low computational complexity.

Next we will provide a brief overview of this recommended standard and then we will show its coding performance for lossless compression as compared to other coding techniques.

A. The CCSDS-123 Recommended Standard

As is the case for the previous Recommended Standard for mono band lossless and lossy data compression issued by the CCSDS in November 2005, CCSDS-122 [11], the CCSDS-123 standard defines the encoding process and is also structured in two functional parts (see Section III below). Fig. 3 illustrates these functional parts.

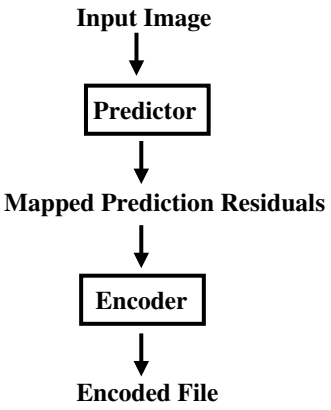


Fig. 3: CCSDS MHDC Functional Parts: Prediction followed by Encoding.

In the first stage, a predictor is used to estimate the value of the current pixel based on previously visited pixels. In the

TABLE I: AVIRIS and Hyperion images used in the experiments. Technical names, sizes and zero-order entropies (in bpppb) are provided.

Sensor	Name	Technical Name	Size ($x \times y \times z$)	Entropy
AVIRIS	Hawaii Uncalibrated	f011020t01p03r05	614 × 512 × 224	9.09
	Maine Uncalibrated	f030828t01p00r05	680 × 512 × 224	8.55
	Yellowstone Radiance Sc0	f060925t01p00r12	677 × 512 × 224	10.33
	Yellowstone Uncalibrated Sc0	f060925t01p00r12	680 × 512 × 224	12.62
Hyperion	Agricultural Calibrated	EO1H0280342004074110PX	256 × 3129 × 242	10.05
	Coral Reef Calibrated	EO1H0830742003120110PW	256 × 3127 × 242	8.46
	Urban Calibrated	EO1H0440342002212110PY	256 × 3176 × 242	10.01
	Erta Ale Uncalibrated	EO1H1680502010057110KF	256 × 3242 × 242	9.46
	Lake Monona Uncalibrated	EO1H0240302009166110PF	256 × 3352 × 242	9.91

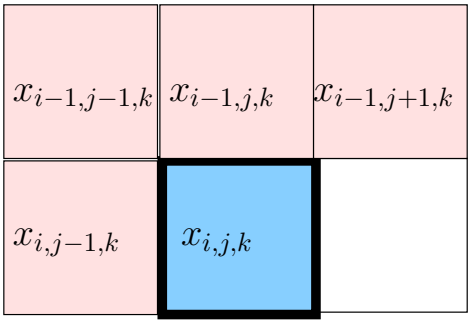
second stage, an entropy encoder is applied to achieve data compaction. This second stage can be carried out on a *sample-adaptive* or on a *block-adaptive* basis. Block-adaptive entropy encoding had been already employed in the first CCSDS Recommended Standard for data compression, intended for mono band image lossless compression [12] and issued in May 1997. Although block-adaptive encoding is not as efficient as sample-adaptive encoding, it is also contemplated to favour still-in-use implementations of the former standard.

1) Predictor: Regarding the first stage, the predictor is asked to provide an estimation or prediction of the current pixel based on previously scanned pixels. Given the original pixel $x_{i,j,k}$ (row i, column j, component k) and the predicted pixel $\widetilde{x}_{i,j,k}$, a prediction error $e_{i,j,k}$ can be computed, similar to the classical definition $e_{i,j,k} = x_{i,j,k} - \widetilde{x}_{i,j,k}$. This prediction error is then mapped to a non-negative integer $\lambda_{i,j,k}$, named *mapped prediction residual*. As seen in Fig. 3, these mapped prediction residuals are passed to the second stage, the entropy coder.

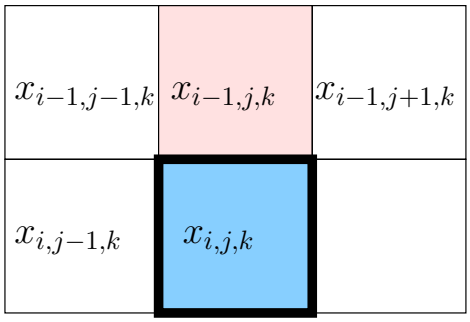
The predicted pixel $\widetilde{x}_{i,j,k}$ is estimated based on neighbouring pixels, both in a spatial or in a spectral sense. These neighbouring pixels are combined to produce a *local sum* $\sigma_{i,j,k}$. When it is expected that the sensed signal has a larger correlation in the vertical direction, only the pixel above the current pixel is employed to compute the local sum; otherwise, four spatial neighbours are used. As mentioned, in addition to spatial neighbours, spectral neighbours from P previous bands, with $P \in \{0, .., 15\}$, might also be taken into account.

Fig. 4 illustrates the spatial neighbours used to compute the local sum. In case of a neighbour-oriented local sum, $\sigma_{i,j,k}$ is computed as the sum of the four spatial neighbours, $\sigma_{i,j,k} = x_{i-1,j-1,k} + x_{i-1,j,k} + x_{i-1,j+1,k} + x_{i,j-1,k}$. In case of a column-oriented local sum, $\sigma_{i,j,k}$ is computed as four times the pixel immediately above the pixel to be predicted, $\sigma_{i,j,k} = 4 \cdot x_{i-1,j,k}$. For pixels in the border of the image, the computations are adapted accordingly.

This local sum $\sigma_{i,j,k}$ is then scaled and used to predict pixel $x_{i,j,k}$. The local sum is a preliminary estimate of the to-be-predicted pixel, and, as it might be not accurate enough, the difference between the computed local sum and their corresponding –scaled– original pixel is tracked and stored in a *local difference vector* $\mathbf{U}_{i,j,k}$ for some samples. There are two possible prediction modes, *full* or *reduced*, depending on whether both spectral and spatial neighbours are considered, or only spectral neighbours. In mathematical form, the *local difference vector* $\mathbf{U}_{i,j,k}$ for the full prediction



(a) Neighbor-oriented local sum



(b) Column-oriented local sum

Fig. 4: Pixels used to calculate the local sum $\sigma_{i,j,k}$.

mode is expressed as

$$\mathbf{U}_{i,j,k} = \begin{bmatrix} d^{\mathrm{N}}_{i,j,k} \\ d^{\mathrm{W}}_{i,j,k} \\ d^{\mathrm{NW}}_{i,j,k} \\ d_{i,j,k-1} \\ d_{i,j,k-2} \\ \vdots \\ d_{i,j,k-P} \end{bmatrix} = \begin{bmatrix} 4 \cdot x_{i-1,j,k} - \sigma_{i,j,k} \\ 4 \cdot x_{i,j-1,k} - \sigma_{i,j,k} \\ 4 \cdot x_{i-1,j-1,k} - \sigma_{i,j,k} \\ 4 \cdot x_{i,j,k-1} - \sigma_{i,j,k-1} \\ 4 \cdot x_{i,j,k-2} - \sigma_{i,j,k-2} \\ \vdots \\ 4 \cdot x_{i,j,k-P} - \sigma_{i,j,k-P} \end{bmatrix}.$$

In this vector, the first three components stand for spatial directional local differences, where the superscript identifies the cardinal orientation (North, West, or NorthWest). The other components stand for spectral local differences.

If reduced prediction mode is employed, the spatial directional local difference components, *i.e.*, the first three components of $\mathbf{U}_{z,y,x}$, are not used.

The prediction itself is a little bit more convoluted. First, the local difference vector $\mathbf{U}_{i,j,k}$ is further scaled through an inner product by a *weight vector* $\mathbf{W}_{i,j,k}$, yielding $\hat{d}_{i,j,k} = \mathbf{W}_{i,j,k}^{\mathrm{T}} \cdot \mathbf{U}_{i,j,k}$, an estimation of the local differences $d_{i,j,k}$. The weight vector is adaptive and is employed to account for the usefulness of each component in the local difference vector when predicting the pixel to-be-coded. These weight vectors can be initialised by default or by a custom user-defined method.

The estimation of the local differences $\hat{d}_{i,j,k}$ and the local sum $\sigma_{i,j,k}$ are then finally employed to produce a *scaled predicted sample value*, $\widetilde{x}_{i,j,k}$ through a rounding operation that takes into account the magnitude of the interval where these coefficients lie and the precision of the registers used for storing them.

Next, as explained previously, a *scaled prediction error* is defined as $e_{i,j,k} = 2 \cdot x_{i,j,k} - \widetilde{x}_{i,j,k}$. However, contrary to what happens in most prediction-based lossless compression methods, these scaled prediction errors are not being sent to the entropy coder, but only used to adapt the weight vector considering their sign.

In the end, the scaled predicted sample value $\widetilde{x}_{i,j,k}$ are employed to compute a *predicted sample value*, $\breve{x}_{i,j,k} = \lfloor \frac{\widetilde{x}_{i,j,k}}{2} \rfloor$, which, in their turn, are used to compute the *prediction residual* $\Lambda_{i,j,k} = x_{i,j,k} - \breve{x}_{i,j,k}$.

The prediction residuals are signed integer values and need to be translated to non-negative integer values, producing the mapped prediction residual $\lambda_{i,j,k}$. In practice, the mapped non-negative integers are computed from the signed prediction residuals $\Lambda_{i,j,k}$, the predicted pixel $\breve{x}_{i,j,k}$, and the least significant bit of the scaled predicted pixel $\widetilde{x}_{i,j,k}$. The non-negative mapped residuals $\lambda_{i,j,k}$ are the final output of the predictor, sent to the next stage, the entropy encoder. At the decoder side, the original pixels can be recovered without loss from $\lambda_{i,j,k}$.

2) Encoder: The second functional part, the encoder, encodes the mapped prediction residual $\lambda_{i,j,k}$ without loss. Recall that the entropy encoding can be applied on a sample basis or on a block basis, with commonly a higher performance for sample-adaptive encoders. In addition, the user can select the order in which the prediction residuals are encoded, either in Band Interleaved by Line (BIL), in Band Interleaved by Pixel (BIP), or in Band Sequential (BSQ) order. This encoding order might be independent of the order with which the sensed pixels are captured, and also independent of the order with which the predicted residuals are produced. The encoding order will not affect the coding performance if sample-adaptive encoding is used, though it usually affects the coding performance of block-adaptive encoding. Notice that the encoding order can impact the memory requirements, as extra buffering might be needed.

As mentioned, block-adaptive encoding is based on the previous CCSDS standard [13] for mono band lossless compression, and is not further discussed here. For sample-adaptive encoding, variable-length binary codewords are used to encode each mapped prediction residual, similar to the process in JPEG-LS standard [14], [15], which the reader may be more familiar with.

In a nutshell, two internal variables are used in the sample-adaptive encoder, an *accumulator* $\Sigma_{i,j,k}$ and a *counter* $\Gamma_{i,j}$. After encoding a given mapped predicted residual $\lambda_{i,j,k}$, this $\lambda_{i,j,k}$ is added to the accumulator and the counter is incremented by 1. The quotient $\frac{\Sigma_{i,j,k}}{\Gamma_{i,j}}$ estimates the average value of the mapped prediction residual, and is used to select the parameter of a Golomb-power-of-two (GPO2) code [16]. Each component or band in the multi-component image can have its own initial accumulator and counter values to determine a different GOP2 code for each band, with the goal to improve the overall coding performance.

As a final remark regarding practical issues, we note that whenever the counter reaches a given limit (the so-called *rescaling counter size*), both the counter and the accumulator are halved, and that the length of each encoded sample is also tested against another given limit (the so-called *unary length limit*), and in case of a codeword exceeding this limit length, a unary sequence is signalled and the mapped prediction residual is simply written in unsigned binary form. This control may prove useful when the first functional part, the predictor, is not able to provide a good enough estimate.

B. Performance assessment

We now here report the lossless coding performance of CCSDS-123 for the images in the considered data set. Table II provides a comparison among three coding standards: classical JPEG2000 [17], RKLT+JPEG2000 –described in Section III–, and CCSDS-123, along with comparison against M-CALIC [18], which is a multi-component-only extension of CALIC [19]. CALIC was devised as a proposal for the ISO standard for lossless and near-lossless compression, JPEG-LS [14], and although it provides a higher coding performance, the less computationally complex LOCO-I [15] was finally selected.

For JPEG2000, no multi-component transform is applied along the spectral dimension and 5 levels of the integer wavelet transform are applied in the spatial dimensions. For RKLT+JPEG2000, the reversible Karhunen-Loève Transform (RKLT) is applied along the spectral dimension and, again, 5 levels of the reversible wavelet transform in the spatial dimensions; see Section III-C for further details on this particular technique. For M-CALIC, default parameter settings have been used, and BSQ image ordering. For CCSDS-123, AVIRIS images have been compressed setting neighbour-oriented local sum and full prediction mode, while for Hyperion images we have resorted to column-oriented local sum and reduced prediction mode; the other tuneable parameters are set by default as suggested in [20], notably the number of previous bands used for prediction is fixed to 3.

Kakadu software [21] has been used for JPEG2000 encodings. RKLT has been computed with the Spectral Transform software [22]. M-CALIC software [23] has been used for M-CALIC. Finally, Emporda software [24] has been run for CCSDS-123.

As expected, classical JPEG2000 yields the poorer performance, as the spectral correlation is not exploited. Among the three other coding techniques, the results are rather similar. For uncalibrated images, *i.e.*, images as produced on-board the

TABLE II: Lossless coding performance for multi-component remote sensing data compression

Sensor	Name	Compression Technique			
		JPEG2000	RKLT+JPEG2000	M-CALIC	CCSDS-123
AVIRIS	Hawaii Uncalibrated	4.62	2.85	2.84	2.71
	Maine Uncalibrated	4.55	2.97	2.89	2.78
	Yellowstone Radiance Sc0	7.14	3.86	4.13	3.96
	Yellowstone Uncalibrated Sc0	9.46	6.07	6.32	6.19
Hyperion	Agricultural Calibrated	6.58	5.73	5.44	5.68
	Coral Reef Calibrated	5.71	5.39	5.05	5.42
	Urban Calibrated	6.88	5.76	5.44	5.71
	Erta Ale Uncalibrated	5.07	4.44	4.76	4.30
	Lake Monona Uncalibrated	5.08	4.58	4.95	4.43
Average		6.12	4.62	4.64	4.57

satellite, CCSDS-123 yields the highest performance for both sensors (except for AVIRIS Yellowstone). For Hyperion calibrated images, M-CALIC is the best performing. AVIRIS Yellowstone images have the highest zero-order entropy among all images in the selected dataset, and RKLT+JPEG2000 is able to take advantage of its more convoluted processing to provide the best results. On average, CCSDS-123 seems to provide the best coding performance.

Being CCSDS-123 a coding technique intended for on-board operation, and the one with the lowest computational complexity, it is worth remarking that this technique usually achieves better results for uncalibrated images and still a very competitive performance for calibrated ones. It should be noted that M-CALIC employs an arithmetic entropy coding stage, which yields better coding efficiency than the Golomb code employed by CCSDS-123, but also entails a larger complexity.

In addition to the experiments reported above, the reader is referred to a recent paper by Augé et al. [20] that contains a thorough analysis of the influence of the different parameters in the final coding performance. It is concluded that parameters involved in the prediction stage have a more significant impact than those involved in the entropy coding stage.

III. Lossy data compression

A. Monoband Compression

In the mid-90s, the field of still image data compression had seen a significant revolution with the advent of coding techniques based on the wavelet transform followed by bit plane coding, of which EZW [25] and SPIHT [26] are two outstanding representatives. Already in 1998, CCSDS defined a list of requirements for their next on-board 2-dimensional (2D) progressive lossy-to-lossless coding system [27] for remote sensing data, including, *e.g.*, frame and non-frame (push-broom) data processing, variable recovered image quality or target bitrate, real-time operation, and capability to deal with input images from 4 to 16 bits per pixel (signed or unsigned). In 2005, a Blue Book for CCSDS-122 [11] was approved as an international standard for Image Data Compression (IDC)[4]. Two years later, a Green Book [28] furnishing guidance and further explanations was approved. In the mid-time, an ISO/IEC International Standard [17] had been approved gathering the most advanced features for 2D image coding at that time, JPEG2000.

The primary features of CCSDS-IDC Recommendation consist of a design for remote-sensing scenarios, a careful trade-off between coding performance and computational complexity, and an adjustable choice of several parameters. However, it lacks some features like capability for interactive transmission or decoding, and extension to multi-component data. The Blue Book provides several reasons as to why ISO/IEC JPEG2000 [7] standard was not well suited for space missions and a new coding technique had to be devised. These reasons had to do with on-board operation, which asked for reduced computational complexity and radiation-hardened software and hardware, and with a focus on compression instead of distribution of coded data. Also, since CCSDS-IDC Recommendation is employed on-board many different space equipment, the coding system had to be designed to comply with such memory and computation restrictions.

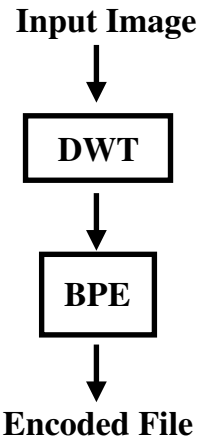


Fig. 5: CCSDS-IDC Functional parts: Discrete Wavelet Transform followed by Bit Plane Encoding.

CCSDS-IDC Blue Book defines the encoding process. Following the then well-established paradigm, there are two main functional parts: first a spatial wavelet transform is carried out; then, a bit-plane encoder is run on the wavelet coefficients. Fig. 5 illustrates these functional parts. In the first stage, a 2D separable wavelet transform is applied with a fixed number of levels, 3. If lossy compression is targeted, the 9/7 biorthogonal filter is employed. If lossless compression

[4]CCSDS-122 and CCSDS-IDC will be used indistinguishably in this text.

is targeted, a non-linear approximation is employed. The first one is called 9/7 Float DWT and requires a rounding-to-the-nearest-integer step before bit-plane encoding. The second one is called 9/7 Integer DWT and requires a weighting of the wavelet coefficients before bit-plane encoding; it has a lower computational complexity.

Before entering the bit-plane encoder, wavelet coefficients are subject to a re-arrangement into blocks of size 8×8. These blocks contain 1 coefficient from the residual sub band LL_3 (the so-called DC coefficient), 3 coefficients from decomposition level 3 (1 coefficient from each oriented sub band, HL_3, LH_3 and HH_3; these coefficients are referred to as parent coefficients), 12 coefficients from decomposition level 2 (4 coefficients for each oriented sub band; these coefficients are referred to as children coefficients), and 48 coefficients from decomposition level 1 (16 coefficients for each oriented sub band; these coefficients are referred to as grandchildren coefficients). Excluding the DC coefficient, the remaining 63 coefficients are called AC coefficients. Fig. 6 illustrates this re-arrangement. In a sense, each of this 8×8 block corresponds to a spatial area in the original input image.

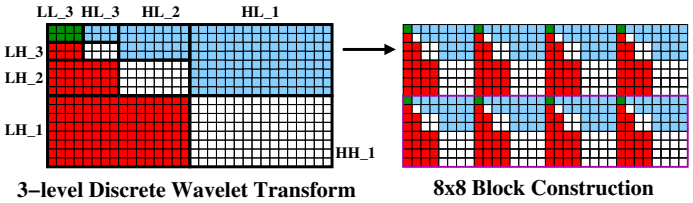


Fig. 6: CCSDS-IDC: 8×8 Block re-arrangement.

The 8×8 blocks are then grouped into segments. Memory availability and raster mode dictate the number of blocks per segment, namely if frame mode is selected, all blocks of the image belong to the same segment; if strip mode is selected, a segment contains as many blocks as available in a row. Then, each segment is independently fed into a bit-plane encoding engine. In practice, each 16 consecutive blocks within a segment are accommodated in a gaggle, and all these 16 blocks are entropy encoded together, although not completely independently from other gaggles.

Within a segment, the entropy encoding proceeds from lower resolution levels (LL_3) in the wavelet domain to higher resolution levels (up to HH_1); also, DC and AC coefficients are encoded in a different way, as DC coefficients represent a lower resolution version of the original image, while AC coefficients account for the details in the high resolution sub bands. When all the blocks in a segment have been processed, their bitstreams are interleaved and an encoded segment code stream is produced. Fig. 7 provides the pseudo-code for the bit plane encoding process. The interested reader is referred to the standard itself [11] for full details.

Inside a segment, the bit plane encoding follows this processing order:

1) To provide a finer degree of scalability in terms of rate-distortion, DC coefficients are in fact sent in three separated stages. First of all, DC coefficients are quantized. This quantization consists of a division by a power of two that depends on the dynamic range (magnitude) of all wavelet coefficients in that segment (including AC coefficients). Then the difference between consecutive quantized DC coefficients (in raster scan order) is computed. This difference is mapped to a non-negative integer. Every such 16 consecutive integers are marked as belonging to the same gaggle, and are then jointly entropy-coded. These mapping and entropy-coding are based on the CCSDS Lossless Compression standard [12]. For entropy-coding, there are two choices, either a fixed-length code or a variable-length code; the former is faster, while the latter yields improved coding performance.
2) After sending the first bits of the quantized DC coefficients, the non-quantized DC coefficients might still have some bit planes above the number of bit planes needed by the largest AC coefficient in that block. Some of these remaining bit planes are then sent on a bit plane by bit plane fashion.
3) Next, the bit depth of the AC coefficients in each block is sent. In fact, similar to what happens for the DC quantized coefficients, the difference between the highest bit plane depth –of all AC coefficients– in two consecutive blocks within a segment is computed, then mapped to a non-negative integer, and then jointly entropy-coded for all gaggles.
4) Last, if there are any, the remaining DC coefficients bit planes along with the bit planes of AC coefficients are sent also on a bit plane by bit plane basis using several refinement passes.

Segment header
Initial coding of DC coefficients
Quantized DC coefficients
DC refinement (if required)
Bit depth of AC coefficients
From BitDepthAC to the last bitplane
DC refinement (if required)
Parents significance
Children significance
Grandchildren significance
AC refinement

Fig. 7: CCSDS-IDC: Bit plane encoding processing order.

Within a segment, the bitstreams of the different blocks are interleaved, so that the overall quality of the recovered image is improved (it is expected that sending some information of all spatial areas of the image is better than specializing in a reduced spatial area). Notice also that the bitstreams for each 8×8 block are associated with the other blocks in the same gaggle, and that it it not possible to decode a single block. In addition, because of the mapping to non-negative integer values and entropy-coding employed for both DC coefficients and AC coefficients highest bit depth, a gaggle can not be independently decoded either, and information from the whole segment must be retrieved.

B. Monoband Experiments

We now here report the progressive lossy-to-lossless coding performance of CCSDS-IDC for the images in the considered

data set. Figs. 8 and 9 provide a rate-distortion comparison between two coding standards, classical JPEG2000 [29] and CCSDS-IDC, for one component of AVIRIS Yellowstone (band 99) and Hyperion Lake Monona (band 48).

None of these two coding techniques apply any multi-component transform along the spectral dimension. As for the spatial wavelet transform, JPEG2000 applies 5 levels while CCSDS-IDC is restricted to 3 levels. Both use a float 9/7 discrete wavelet transform. The figures plot the bitrate (in bits per pixel per band) versus the Signal to Noise Ratio (SNR), measured in dB, considering the original energy of the image.

Kakadu software [21] has been used for JPEG2000 encodings. Delta software [30] has been used for CCSDS-IDC.

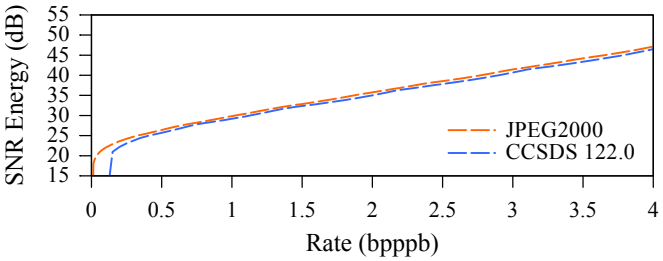


Fig. 8: AVIRIS Yellowstone Uncalibrated Sc0 Band 99. Rate-Distortion comparison.

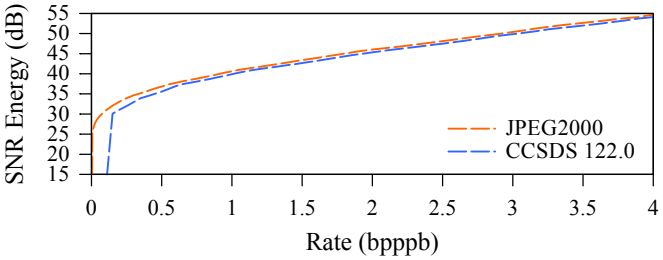


Fig. 9: Hyperion Lake Monona Uncalibrated Band 48. Rate-Distortion comparison.

The reported results indicate that at very low bitrates (large compression ratio), JPEG2000 benefits from the use of a 5-level DWT. However, from 0.25 bpppb onwards, the performance is rather close. The results for the other components of these two images as well as for the other images in the data set are consistent with these findings.

As happened with CCSDS-123 standard reviewed in previous Section II, the CCSDS-IDC recommended standard is able to provide a highly competitive rate-distortion performance for remote sensing data when applied on-board as compared to more complex coding techniques.

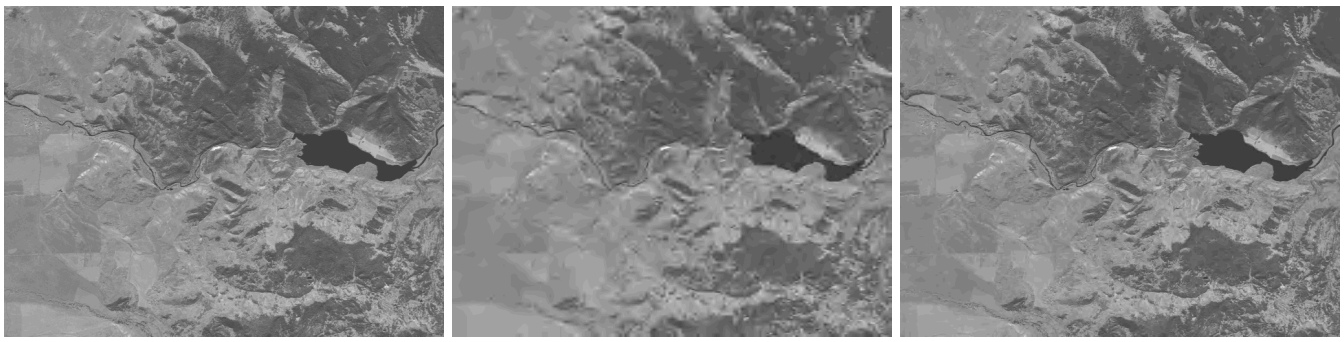
Fig. 10 shows component 99 of the original AVIRIS Yellowstone uncalibrated image (Scene 0), as well as the recovered image when it has been encoded at 0.1 and 1.0 bpppb. Fig. 11 shows a zoomed-in area of the same image at the same bitrates. A bitrate of only 0.1 bpppb seems scarce for most remote sensing applications, but the quality of the recovered image at 1.0 bpppb, while still presenting visible artifacts, is notably higher.

C. Multi-component Compression

In the case of lossy compression of multi-component images, there is no available CCSDS recommendation at the time of this writing, although the CCSDS is working towards the definition of a lossless and lossy compression algorithm that extends CCSDS-IDC to multi-component images, supporting several transforms in the spectral dimension. A first attempt to assess what the performance could be for such an extension was provided in [31]. However, the JPEG2000 standard does provide a multi-component extension in its Part 2 [32], which has been exploited by several authors to design compression algorithms for multispectral and hyperspectral images. Such algorithms are not amiable to on-board compression due to the complexity and memory requirements of a JPEG2000 encoder, but they can be used for on-the-ground image compression and delivery. In the following we will describe a typical setting for multi-component image compression based on JPEG2000; it is expected that many of the principles on which this algorithm is based will carry over to the multi-component extension of CCSDS-IDC, when this will become available.

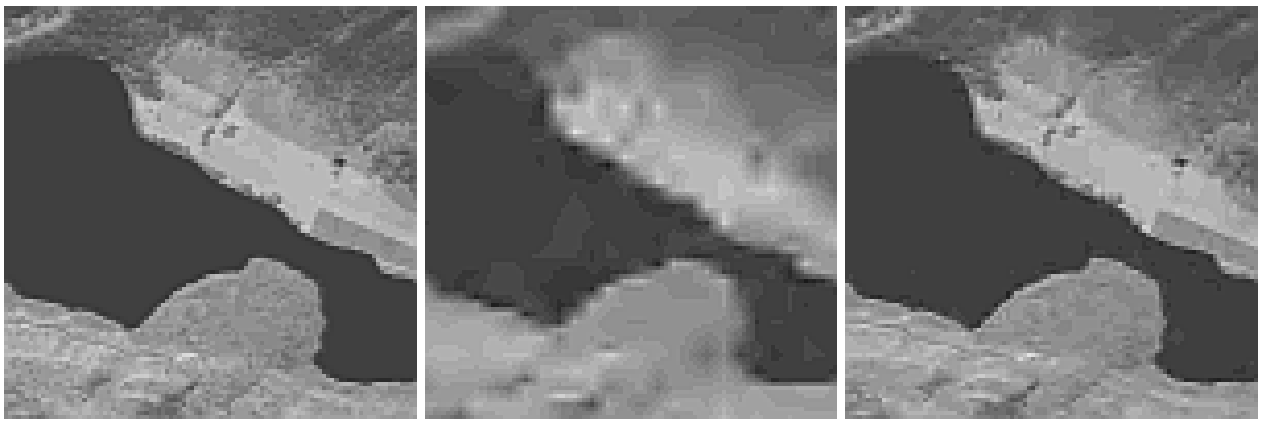
The key ingredient of lossy multi-component image compression is the choice of transform to be applied along the spectral dimension, as it is known that this dimension exhibits a very high degree of correlation. There is not a single best choice, but several options that one can choose from depending on specific complexity and performance requirements, *e.g.*, a spectral discrete cosine transform, wavelet transform, Karhunen-Loève transform (KLT), or some approximation of the KLT. Common to all these choices is the notion that the spectral transform should be applied separately from the spatial transform; since the nature of the correlation in the spectral and spatial dimensions is different, there is no benefit in employing a transform that is isotropic in all dimensions. *E.g.*, if one chooses a spectral wavelet transform, the best results are obtained applying first all levels of the spectral transform, and only later applying the spatial transform to the spectral wavelet coefficients. Having said that, the transform that typically provides the best results is the spectral KLT. This transform has been employed as a spectral decorrelator by many authors [33], [34], [35], [36], [37], [38], [39], [40], [41], [42], and is known to be optimal for decorrelation of Gaussian processes [43]. KLT has a non-negligible computational cost and several approaches have been proposed to alleviate this complexity [44], [45].

Defining the covariance matrix C_X of a random column vector X with mean value μ_X as $C_X = E[(X - \mu_X)(X - \mu_X)^T]$, the KLT transform matrix V is obtained by aligning columnwise the eigenvectors of C_X. It can be shown that the transformed random vector $Y = V^T(X - \mu_X)$ has uncorrelated components, *i.e.*, $C_Y = V^T C_X V$ is a diagonal matrix. It should be noted that the KLT transform coincides with principal component analysis, which is a well known tool for dimensionality reduction. While many papers employing the KLT for compression, they use it to reduce the number of components by zeroing out a given number of least significant transformed bands, in general it is more appropriate to keep all components and represent them with different accuracy via rate-distortion optimization techniques. As far as multi-component image compression is concerned, for calculating the KLT one considers each spectral vector (*i.e.*,, each one-



(a) Original (b) 0.1 bpppb (c) 1.0 bpppb

Fig. 10: AVIRIS Yellowstone Uncalibrated Sc0 Band 99. Full Size.



(a) Original (b) 0.1 bpppb (c) 1.0 bpppb

Fig. 11: AVIRIS Yellowstone Uncalibrated Sc0 Band 99. Images have been magnified and sharpened so that artifacts are more visible.

dimensional vector obtained fixing the spatial coordinates i, j and letting the spectral coordinate k vary) as a sample realization of a random process. From the set of sample realizations the mean vector μ_X and covariance matrix C_X can be calculated; diagonalizing C_X yields the transform matrix V. This transform can be used as spectral decorrelator in conjunction with any monoband compression algorithm to be applied to each transformed spectral channel (eigenimage). Since eigenimages have different energy, though, the problem arises to allocate the available bit-rate budget to the monoband coding of each eigenimage. Indeed, much like bit allocation is necessary in 2D compression of a single image, it is all the more necessary in the multiband case, in which not only different units in one transformed band may have different energy, but different transformed spectral channels may also have different energy. In the following we describe the specific approach taken by JPEG2000, which is based on post-compression rate-distortion optimization. Other approaches are also possible, based on JPEG2000 [39] or other coding techniques employing, for example, zerotrees [46], [47].

In particular, JPEG2000 employs the concept of codeblocks, which are similar to the 8x8 blocks in CCSDS-IDC. A codeblock, whose size is defined by the user but cannot be larger than 2^{12} wavelet coefficients (and always within the same wavelet subband), is the basic unit that is encoded independently. Encoding in JPEG2000 is based on an approximated binary arithmetic coder called MQ coder [7] that is applied independently to all bit-planes of each codeblock. First, encoding of the whole image is performed at a relatively high bit-rate. During this process, rate-distortion information is collected regarding each independent coding unit in the 3D transform domain, based on how many bits have been employed by the MQ coder to encode each unit, and what is the contribution of that unit in reducing the distortion of the decoded image. Then, this information is used in order to sort all units in decreasing order of their rate-distortion importance. Finally, coded units are picked from the sorted list and are written in the codestream until the target rate has been achieved. This allows to obtain a rate very close to the desired target. The computational complexity of this process is however rather high, since the algorithm employs an entropy coder whose complexity is not negligible, and uses it to encode data at a rate even higher than the final rate of the codestream. It should be noticed, however, that the rate-distortion optimization procedure is not a mandatory part of the JPEG2000 standard, and simpler techniques could be used, although they would entail a performance loss with respect to the technique described above.

D. Multi-component Experiments

We now here report the progressive lossy-to-lossless coding performance of several multi-component coding techniques for

the images in the considered data set. Fig. 12 and 13 provide a rate-distortion comparison between two coding standards, classical JPEG2000 [29] and CCSDS-IDC, when coupled with a KLT applied on the spectral dimension.

The parameter setting and the software used for this experiment have been already discussed in previous sections.

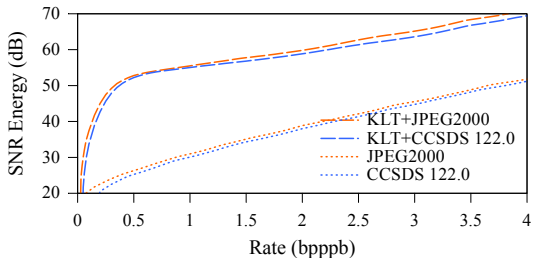


Fig. 12: AVIRIS Yellowstone Uncalibrated Sc0. Rate-Distortion comparison

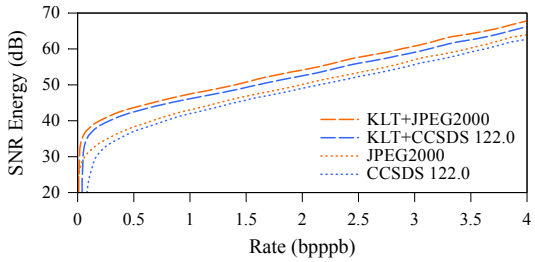


Fig. 13: Hyperion Lake Monona Calibrated. Rate-Distortion comparison

Fig. 12 illustrates the benefits of using a multi-component transform (KLT) on an uncalibrated image. As soon as bitrate increases over 0.5 bpppb, applying a KLT along the spectral dimension boosts the coding performance by about 20 dB. This boost is dependent on the image being coded, with some image yielding higher gains than others.

Fig. 13 shows that multi-component transform does also bring an improved rate-distortion performance for already calibrated images. In this case, the increase is about 5 dB. Notice also that here JPEG2000 provides some gain as compared to CCSDS-IDC, although the impact of the restricted number of spatial wavelet levels of CCSDS-IDC is not significant when the spectral dimension has gone through a decorrelating transform.

IV. Extensions: Predictive Near-Lossless and Lossy Compression

In the previous sections we have introduced the *lossless* and *lossy* types of image compression algorithms. While these algorithms are good for many compression applications, another type is also very popular, namely *near-lossless* compression. This is a "hybrid" compression mode that borrows features from lossless and lossy compression, attempting to strike a compromise that can turn out to be useful in many cases. Indeed, while lossless compression is highly desirable, it typically yields limited compression ratios. On the other hand, lossy compression can achieve large compression ratios, but this comes at the expense of some quality loss and, more importantly, it is not easy to accurately control this loss. Specifically, in transform-based lossy compression it is generally possible to achieve the desired mean-squared error (MSE) between the decoded and original image, but a more accurate and per-pixel quality policy is difficult to obtain. This is due to the fact that the quality loss is introduced in the transform domain, where it can be perfectly controlled, but it is then mapped to the pixel domain via the inverse transform, which mixes the errors in a way that is hard to control. In many cases this is not an issue; if it is, then near-lossless compression can provide a useful alternative. The term "near-lossless" refers to a specific type of lossy compression in which, rather than attempting to minimize the MSE for a given target bit-rate, the compression process will minimize the bit-rate while providing a bounded maximum reconstruction error. That is, the compression process takes as input a user-defined maximum absolute error Δ between any pixel of the decoded image and the corresponding pixel of the original image, and will reduce the data size as much as possible while guaranteeing that, for each pixel of the reconstructed image, the following condition is verified:

$$\max_{i,j,k} \|x_{i,j,k} - \widehat{x}_{i,j,k}\| \leq \Delta$$

This represents the basic kind of near-lossless compression. The term "near-lossless" hints to the notion that, if Δ is chosen appropriately, it is almost "as if" the compression process were lossless. Indeed, any image is affected by some inherent noise [48], which bounds the performance of any lossless compression algorithm. If the maximum error introduced by the compression process is smaller than the inherent noise, then the quality of the reconstructed image would be the same as if lossless compression had been applied. Before discussing near-lossless compression techniques, it is worth pointing out the differences between near-lossless and lossy compression in terms of the functionality they can achieve. As has been seen in Section III, transform-based lossy compression allows one to achieve a very accurate rate control, while quality control is typically limited to the MSE. On the other hand, near-lossless compression makes it possible to achieve very fine per-pixel quality control, but does not lend itself naturally to obtain rate control. This does not mean, however, that transform-based schemes may not achieve accurate quality control or near-lossless schemes may not achieve rate control, as we will show later on.

A. Techniques for near-lossless compression

The easiest and most effective way to design a near-lossless compression algorithm is to resort to prediction in a differential predictive coded modulation (DPCM) scheme [49]. In Section II it has been shown that prediction is the basic technology underlying many lossless compression algorithms. The basic principle is to employ a mathematical model of the correlation among adjacent pixels of the image, in the form of a linear or nonlinear predictor. The task of the predictor is to calculate an estimate $\widetilde{x}_{i,j,k}$ of the current pixel $x_{i,j,k}$ being encoded. The predictor is a function of "past" pixels that have already been processed, i.e. previous pixels in the same line or pixels on previous lines or spectral channels of the image. Then, only the prediction error $e_{i,j,k} = x_{i,j,k} - \widetilde{x}_{i,j,k}$ is encoded in the compressed file, i.e. the information on

the pixel value that was not modeled by the predictor. For near-lossless compression, though, a modification to this basic mechanism is needed, as shown in Fig. 14. The reason lies in the fact that, in the lossless compression case, past samples of the original signal are indeed available at the decoder, which can recalculate the predictor for the pixel currently being decoded. However, this is clearly not possible in lossy compression. The way this problem is addressed involves implementing a local decoder at the encoder, as is shown in the shadowed box in Fig. 14. The easiest way to interpret this figure is as follows: instead of calculating the predictor $\widetilde{x}_{i,j,k}$ from previous samples of the original image $x_{i,j,k}$, the modified scheme employs instead previous samples of the *decoded* image $\widehat{x}_{i,j,k}$. This requires that, every time a predictor is computed and the corresponding prediction error $e_{i,j,k}$ is calculated, the prediction error is quantized to $e^Q_{i,j,k}$ and entropy-coded, then reconstructed to $\widehat{e}_{i,j,k}$, and finally the decoded sample $\widehat{x}_{i,j,k}$ is obtained, and used to predict future pixels. On the other hand, only the inclusion of an inverse quantizer is required at the decoder, which otherwise works as described in Section II. Note that, so far, we have assumed that the maximum error Δ is the same for all pixels in the image. This is an important case, but generalizations are also possible, as will be seen later in this section, where a rate-distortion optimization block as in Fig. 14 is used to appropriately select quantization step sizes so as to attain a given objective.

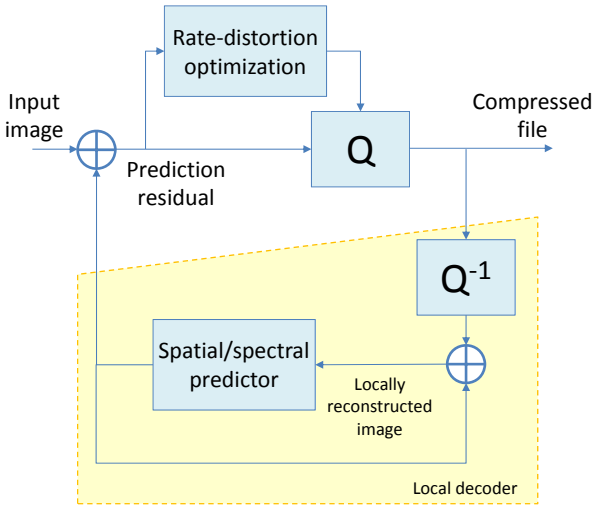


Fig. 14: DPCM scheme for near-lossless compression.

A few remarks are in order regarding this near-lossless scheme based on the DPCM feedback loop. First, this is a very general scheme which can employ any causal predictor; as a consequence, near-lossless versions of popular lossless compression schemes are readily obtained, see *e.g.*, the near lossless version of CALIC [50] and its multiband extension [18], and the near-lossless algorithms based on crisp and fuzzy selection among a set of predictors [51], [52], [53]. Second, it is easy to see why this scheme achieves a bounded maximum error between any decoded and original pixel. Indeed, for each pixel $x_{i,j,k} = \widetilde{x}_{i,j,k} + e_{i,j,k}$, one reconstructs an approximation $\widehat{x}_{i,j,k} = \widetilde{x}_{i,j,k} + \widehat{e}_{i,j,k}$, hence the error is equal to $x_{i,j,k} - \widehat{x}_{i,j,k} = e_{i,j,k} - \widehat{e}_{i,j,k} = q_{i,j,k}$, which is exactly the quantization noise introduced in representing $e_{i,j,k}$. If the quantizer employed is a scalar uniform quantizer, then the maximum error is bounded by $\|x_{i,j,k} - \widehat{x}_{i,j,k}\| \leq \frac{\delta}{2}$, where δ is the quantization step size. A typical choice is to employ a quantization step size equal to an *odd* integer number, *i.e.*, $\delta = 2\Delta + 1$, where Δ is the maximum error achieved. This choice minimizes the reconstruction MSE for a given maximum error Δ.

At this point, one might wonder if this near-lossless compression scheme works well irrespective of the value of Δ, *i.e.*, if it performs equally well at all bit-rates and how it compares to transform-based techniques. The key to understand this lies in the fact that the performance of this scheme depends on how "accurate" the prediction of the current pixel is, since the lower is the energy left in the prediction error samples, the fewer bits will be required to encode these samples for a given degree of accuracy. However, quantization plays against the predictor in the following sense. In this DPCM scheme, the input to the predictor are the past decoded pixels of the image. At high quality levels, when Δ and δ are very small, this near-lossless scheme works indeed really well. As one moves to higher degrees of compression, increasing Δ and δ, the performance tends to degrade. This is due to the fact that, as the decoded pixel values are more and more different from the original values, the correlation between the decoded past values and the current pixel value tends to decrease, and as a consequence the predictor will do a poorer job of estimating the current sample. Thus, the rate-distortion performance of near-lossless compression tends to degrade more and more as the quality decreases, making this a poor choice for low quality levels. This is generally not a serious issue, since for multispectral and hyperspectral images the quality levels of interest are rather high. It should also be noted that the mechanism that renders the prediction quality poor has been subject of an interesting theoretical analysis in [54], where it is shown that an appropriate filtering and downsampling strategy can improve the performance of DPCM, preventing high-frequency quantization noise from disrupting the predictor's performance. On a related note, in [55] it has been shown that the low bit-rate performance of near-lossless compression can be highly improved by regularizing the reconstructed quantization error $\widehat{e}_{i,j,k}$. Indeed, as δ becomes large the conventional midpoint reconstruction fails to account for available prior information regarding the image, whereas reconstruction values satisfying e.g. a smooth or wavelet-sparse image model should be chosen instead.

Since the near-lossless compression paradigm above can be applied to any predictor, it can also be used to design a near-lossless version of the CCSDS-123 lossless compression algorithm leveraging the predictor described in Section II. The elements that need to be added to CCSDS-123 are the following.

- A scalar uniform quantizer with odd quantization step size that maps $e_{i,j,k}$ to $e^Q_{i,j,k}$, and the corresponding reconstruction function that maps $e^Q_{i,j,k}$ to $\widehat{e}_{i,j,k}$.
- All calculations of local sums and predictor values are based on the past decoded samples $\widehat{x}_{i,j,k}$ and not the original samples $x_{i,j,k}$.
- The weight update rule employs $e^Q_{i,j,k}$ instead of $e_{i,j,k}$.
- Importantly, the entropy coder defined by CCSDS-123 has to be changed if one want to achieve bit-rates below

1 bpppb. This is because the Golomb code encodes one symbol at a time. Since the minimum length of a Golomb codeword is one bit, the average length can not be less than 1 bpppb. It is indeed possible to achieve bit-rates below 1 bpp, but this requires some kind of block coding, which can be as simple as a run-length encoder to take advantages of long sequences of zeros, or a fully-fledged block coder such as an arithmetic or range encoder. In the experiments provided in Section IV-C the quantized residuals have been coded using a range encoder, which has a better coding efficiency than a Golomb code.

B. Generalization to varying quality levels and rate control

Since near-lossless compression can indeed achieve a specific maximum absolute error on every individual pixel, the question arises whether it has any interest at all to make the maximum error a variable quantity $\Delta_{i,j,k}$, which can be set to a different value for each pixel by selecting an appropriate quantization step size $\delta_{i,j,k}$. There are several reasons why one may want to do so.

- In some applications, rather than setting a constant maximum error for each pixel, it is of interest to bound the maximum relative error. For example, in astronomy it is often the case that the level of noise is proportional to the signal level, so that a larger error may be tolerated on the pixels with higher values, and vice versa.
- In some applications there could be areas of the image which are more important to the scientists ("regions of interest"), and pixels belonging to these regions could be represented with a higher quality.
- By varying the quality in different areas of the image, one could obtain the rate control functionality, or a hybrid rate and quality control.

Achieving the first two functionalities is rather simple, and is readily done via a proper selection of quantization step sizes for each pixel to obtain the desired quality. This is not difficult since in near-lossless compression quality can be modulated in a natural way. On the other hand, modulating the quality to obtain rate control is a more involved issue, significantly more so than in the case of transform coding. The main reason behind this is the following. In transform coding, one typically assumes that the transform coefficients are statistically independent. Therefore, the problem of choosing quantization step sizes can be solved disregarding the interactions among the different quantization choices applied to data units in the transform domain. Unfortunately, the same cannot be done for predictive coding. Because of the way the prediction mechanism works, a quantization choice on a given pixel will affect the rate and quality of that pixel, but also the rate of the subsequent pixels that are predicted from it. In general, if we represent a pixel with very good quality using an appropriately high bit-rate, then this pixel will retain most of its correlation with the subsequent ones, so that the predictor employing that pixel will yield a small prediction error. Conversely, a pixel that is represented coarsely will typically generate higher prediction error values in the next pixels. The interaction between the rate-distortion choices for a pixel and their effects on the next pixels are difficult to model. In [56] it is shown that the problem can be solved by representing all sets of possible coding options as states on a trellis, and then running the Viterbi algorithm. This approach, however, is unfeasible in practice due to its complexity.

Practical solutions must find an allocation of quantizers that is greedy, since not all the image data can be stored in memory at the same time, and that has low complexity. One possible solution would be to consider near-lossless compression with a single maximum error Δ throughout the image, and choose Δ so as to obtain the desired bit-rate, as sweeping Δ from very low to very high values will yield rate-distortion points from high quality to low quality. Incidentally, this solution is rather good in terms of quality, as it can be shown to be optimal in minimax distortion sense under a Gaussian assumption on the image pixels [57]. Although not necessarily optimal in MSE sense, this solution has the desirable property that the quality is balanced throughout the image. In practice, however, this approach is not viable. Indeed, there is only one parameter to be chosen for the whole image, and it has to be selected in a greedy way without the possibility to perform adaptation to the image content. This makes the process prone to large errors in the rate control.

In [58] a solution to this problem has been proposed, which adapts well to the multispectral and hyperspectral imaging case, since it performs greedy allocation of quantizers so as to achieve the desired bit-rate. The purpose of the algorithm is to control the output rate of a predictive encoder of hyperspectral and multispectral images under low complexity and memory constraints. This rate control algorithm can work with any predictor, and it selects the quantizers to be applied to the prediction errors. It works on a slice-by-slice basis, where a "slice" is defined as a predefined number of lines (*e.g.*, 16) with all their spectral channels. Each slice is divided into non-overlapping 16×16 blocks. An individual quantization step size is computed for each block in each spectral channel, so that lossy predictive coding employing the computed step sizes shall achieve a rate as close as possible to the target. The rate control algorithm is a multistage process that computes such step sizes, as depicted in Fig. 15. In particular the following steps are performed:

- *Training* stage: a rate-distortion model predicting the rate-distortion curve of each block in each spectral channel of the slice is built as function of the variance of the unquantized prediction residuals and of the quantization step size to be used for the block. The former is estimated by running the lossless predictor on a small number of lines in the slice. This process defines a rate-distortion function $R(\sigma_i^2, \delta_i)$ which, given a quantization step size δ_i chosen for block i, and the variance σ_i^2 of the prediction residuals for the block, provides an estimate of the rate needed to code the block. This model is the key ingredient to tackle the rate control problem. In [58] the rate-distortion function of a Laplacian source is employed, since the Laplacian distribution is a good model for prediction residuals, and the corresponding rate-distortion function is known in closed form. Then, the final quantization step sizes are obtained as follows.
- *Optimization stage - step 1*: First, an initial set of quantization step sizes are calculated, which approximately achieve the target rate but are suboptimal in terms of distortion. There are many ways to perform this. For example, in [58] this is done considering first the lossless compression case, *i.e.*, $\delta_i = 1$ for $i = 1, \ldots, N$, being

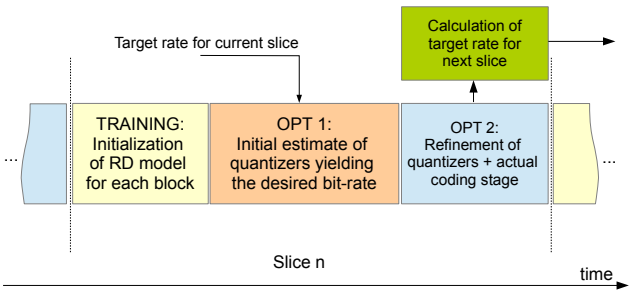


Fig. 15: Serial implementation.

N the number of blocks belonging to a slice, and the corresponding rates $R(\sigma_i, 1)$ for each block that sum up to the estimated lossless compression bit-rate. This feasible rate-distortion point is then "projected" onto the set of points that define the desired rate constraint, namely $\sum_{i=1}^{N} R(\sigma_i, \delta_i) = R_{\text{target}}$, where R_{target} is the actual desired bit-rate and each term $R(\sigma_i, \delta_i)$ employs the rate-distortion model calculated in the previous step. Since this set is an l_1 ball, the unknown terms δ_i can be calculated using standard optimization techniques.

- *Optimization stage - step 2*: A greedy algorithm makes local adjustments aimed at promoting low-distortion allocations of the quantization step sizes, employing the rate-distortion models of all blocks in the slice. The main idea is as follows. All quantization step sizes of each block are decreased by 2 (because only odd step sizes are allowed); this reduces the distortion, but increases the bit-rate. In order to obtain the target bit-rate, a certain number of blocks are selected and their quantization step size is increased; in particular, these blocks are selected as those that are optimal in rate-distortion sense, based on the rate-distortion model. This procedure is iterated a few times until the allocation stabilizes.

Furthermore, the algorithm employs feedback from one slice to the next, using information on the actual rate produced encoding a slice, in order to update the target rate for future slices, thereby compensating for inaccuracies of the allocation process. The idea is to track the deviation of the actual rate from the predicted one, and to compensate this deviation over a few next slices. The algorithm described in [58] is designed as a tracking algorithm, and has provable guarantees to converge to the desired target rate under some mild conditions on the regularity of the rate deviation.

The application of this rate control algorithm to CCSDS-123 is rather simple. The only caveat lies in the fact that the predictor in CCSDS-123 performs adaptation of the weights. This implies that the training stage starts using the weights calculated at the end of the previous slice, and the weights employed for lossless prediction in the training of the next slice will have to be updated after the current slice has been predicted and encoded. This may introduce a delay since the rate control algorithm cannot be run in parallel to the prediction and encoding stage. A modified version of this algorithm allowing to pipeline rate control and prediction/encoding has been proposed in [59].

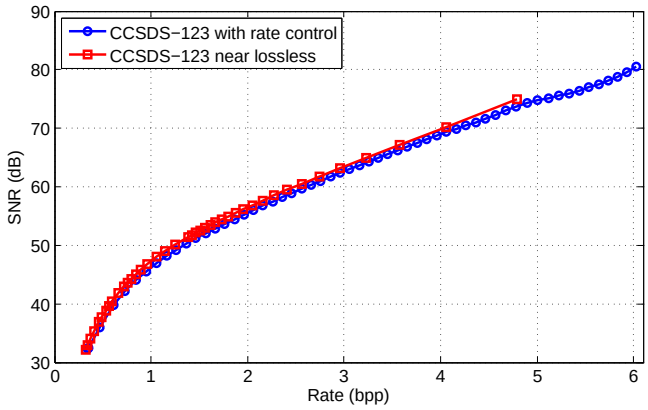


Fig. 16: Rate-distortion performance comparison between two different modifications of CCSDS-123, near-lossless (red curve) and lossy version with rate control (blue curve). The curves refer to the Yellowstone Uncalibrated Sc0 image.

C. Performance comparison

In the following we provide a performance comparison of the near-lossless version of CCSDS-123 described above with another near-lossless compression algorithm, namely M-CALIC [18]. The maximum errors are selected so as to span a reasonable range of bit-rates, which are mostly concentrated in the high-quality range. Note that M-CALIC employs an arithmetic coder as an entropy coding stage, whereas the near-lossless CCSDS-123 employs a range coder, which in this case is a simplified and slightly suboptimal version of an equivalent arithmetic coder. The suboptimality of the range coder with respect to the arithmetic coder is typically within 0.1 bpppb. Also note that, because of the use of the range encoder, the CCSDS-123 lossless compression bit-rates obtained with this experiment are lower than those provided in Table II, which report the performance of the standardized version of CCSDS-123 employing a Golomb coder. For M-CALIC, no specific adaptation of the weight for the previous two bands has been performed; the weights are the same as in [18]. Also note that, in view of application to on-board compression, M-CALIC has been run in BIL mode, *i.e.*, the image is read and encoded by spectral lines.

The results of this comparison are reported in Table III. As can be seen, the near-lossless version of CCSDS-123 consistently outperforms M-CALIC despite employing a range encoder instead of a fully-fledged arithmetic coder. The achieved bit-rates obviously vary for different images, but in general it is possible to obtain rather low bit-rates with a limited value of δ. As has been said, for both algorithms the use of a block coder allows to achieve bit-rates below 1 bpppb.

We also show a comparison of the performance of the near-lossless CCSDS-123 (as introduced in Section IV-A) and the lossy version of CCSDS-123 with rate control (as introduced in Section IV-B). In particular, Fig. 16 shows the rate-distortion performance of these two extensions of CCSDS-123 on the Yellowstone Uncalibrated Sc0 image. In order to properly interpret these results, it has to be noted that the near-lossless algorithm provides no rate control. That is, Fig. 16 provides the set of achievable rate-distortion points, but the user would not be able to select a specific point *a priori*,

TABLE III: Comparison of different near-lossless coders for AVIRIS and Hyperion images for several values of δ. Coding performance is reported in bits per pixel per band.

Sensor	Name	Compression Technique									
		M-CALIC					CCSDS-123 near-lossless				
		δ=0	*δ=1*	*δ=2*	*δ=5*	*δ=10*	*δ=0*	*δ=1*	*δ=2*	*δ=5*	*δ=10*
AVIRIS	Hawaii Uncalibrated	2.96	1.66	1.21	0.70	0.44	2.63	1.39	0.95	0.40	0.18
	Maine Uncalibrated	3.02	1.72	1.26	0.75	0.49	2.72	1.46	1.00	0.45	0.23
	Yellowstone Uncalibrated Sc0	6.47	4.88	4.15	3.06	2.25	6.20	4.59	3.86	2.77	1.97
Hyperion	Agricultural Calibrated	5.51	4.31	3.72	2.82	2.15	5.32	4.01	3.43	2.55	1.87
	Coral Reef Calibrated	4.86	3.73	3.17	2.29	1.63	4.94	3.63	3.09	2.20	1.55
	Urban Calibrated	5.37	4.24	3.67	2.79	2.13	5.31	4.01	3.42	2.57	1.89
	Erta Ale Uncalibrated	4.97	3.40	2.71	1.74	1.09	4.61	3.09	2.42	1.51	0.89
	Lake Monona Uncalibrated	5.16	3.57	2.87	1.89	1.22	4.69	3.17	2.49	1.57	0.96

but should instead run the encoder with different values of δ until the desired rate has been achieved. Conversely, the lossy version with rate control is run just once as the user selects the desired bit-rate. As can be seen, the lossy algorithm incurs a slight performance loss. This is the price to be paid for employing a greedy rate control algorithm, which provides a slightly suboptimal solution. However, the loss is very small, around 1 dB at 1 and 2 bpppb, and around 0.85 dB at 3 and 4 bpppb.

D. Other techniques

While the techniques described above provide a toolset for performing near-lossless and lossy compression using predictive methods, it is important to notice that other approaches are also possible. We discuss a few of them below.

Regarding near-lossless compression, it should be noted that it is possible to achieve near-lossless compression also using transform-based methods, although it is more complicated than using predictors. For example, [60] designs a quantizer for wavelet coefficients that provides near-lossless compression in the image domain. The technique proposed in [61] first applies a transform-based method such as KLT+JPEG2000, then applies decoding, and finally requantizes and encodes the difference between the original and decoded image to obtain the desired maximum absolute error. This approach is very general and can be applied to any transform coder, which in this framework is indeed employed as a predictor. However, it requires an on-board decoder in addition to the encoder, thereby increasing the complexity.

An aspect that has been shown to improve the performance of any kind of compression is band reordering [62], [63], [64]. This technique is based on the observation that the natural ordering of the spectral channels is not necessarily the ordering that provides the best compression performance. It has been shown [63] that good orderings can be found, for example, by representing the spectral channels as nodes in a weighted graph, where the weights between any pair of nodes correspond to some (negative) measure of correlation between the respective spectral channels. A minimum spanning tree algorithm will yield a good ordering, *i.e.*, the ordering that maximizes correlation between adjacent spectral channels in the reordered image. This has been shown to generally improve the performance of compression algorithms [65], from small to significant degrees depending on the dataset. Although the complexity of band reordering is generally too large to be performed on-board, in [65] it is noticed that the optimal ordering for images acquired by the same sensor is generally very similar, so that it is conceivable to determine the optimal ordering during a training stage, and then use it for on-board compression.

Finally, another approach that is somehow similar in spirit to the prediction-based one is the class of compression algorithms based on the distributed source coding principle. Distributed source coding is an information theoretic technique [66] that can be employed to design a compression algorithm with a computationally light encoder and a more complex decoder. This is clearly an appealing feature for on-board compression, where the computational capabilities at the encoder are scarce, while the decoder is much less constrained. The idea behind distributed source coding, as applied in the context of hyperspectral image compression [67], [68] is that one can still employ a predictor, but instead of encoding the prediction error, only a certain number k of least significant bit-planes of the image pixels are transmitted to the decoder, along with a 16-bit error detection codeword generated applying a cyclic redundancy code to the pixels. The task of the decoder is to estimate the most significant bit-planes; this is done trying all possible elements in the chosen family of predictors, which must be in finite number. Every predictor is used to generate an estimate of the most significant bits, leading to a candidate set of decoded pixels which is checked for exactness using the received error detection codeword. It can be shown [67], [68] that a proper choice of k ensures that this problem has a unique solution. As can be seen, this approach shifts complexity away from the encoder, particularly in the coding stage, and moves it into the decoder. Other approaches based on this framework have been recently proposed [69], [70], including approaches based on transforms [71].

V. Concluding remarks

Remote sensing is an active field of research that puts at the disposal of the scientific community an ever-growing volume of data, enabling a whole new niche of applications. A plethora of aircraft and spacecraft sensors is being today deployed, which generate an unprecedented amount of high-resolution data, though, at the same time, posing the challenging problem of efficiently transmitting and disseminating these data. The downlink transmission channel from the on-board sensor to on-the-ground stations is rather limited, and smart transmission

protocols have to be devised to fully seize the available capacity.

Remote sensing data compression comes as an effective means to increase the transmission rate. Three coding modalities have been proposed in the literature, ranging from lossless through near-lossless to lossy compression, and the operative standards are now rather mature.

We have reviewed the most recent standard/proposal for each type of compression. For both lossless and lossy compression, a description of the recommended standards by the Consultative Committee for Space Data Systems (CCSDS) for on-board data compression has been provided, along with a comparison against the best on-the-ground performing approach. It has been shown that, in general, careful design of the coding techniques, taking into account the restricted on-board computing and memory capabilities, can provide a tantalizing performance, close to that provided by more computationally complex techniques, although, perhaps, at the cost of restraining its features.

For mono band images, CCSDS-IDC, approved in 2005, yields appropriate performance for progressive lossy-to-lossless coding in a large range of bitrates as compared to classical JPEG2000. The underlying functional scheme of both CCSDS-IDC and JPEG2000 is a spatial wavelet transform followed by a biplane encoding.

For multi-component images, and for lossless compression, CCSDS-123, approved in 2012, has revealed itself as the most convenient approach, even compared to more demanding techniques, and not only for on-board operation, but also for on-the-ground coding. CCSDS-123 is a prediction-based approach that exploits both the spectral and the spatial correlation.

Also for multi-component images, but now for progressive lossy-to-lossless coding, CCSDS-122 standard could be coupled with a KLT applied along the spectral dimension to supply competitive results as compared also to KLT+JPEG2000. However, as the KLT is a computationally demanding transform, the CCSDS the Multispectral Hyperspectral Data Compression Working Group is working towards the development of an alternative to the KLT. This approach has not been described in our review, as the standardization process is not yet complete.

Finally, we have addressed near-lossless compression, where the peak absolute error is bounded to a given threshold. This type of compression might prove useful in several application scenarios. Currently, no work item has been started in CCSDS MHDC WG to standardize an approach with this feature, but it is foreseen that it will be discussed in the near future. In addition, when properly combined with a suitable rate-control scheme, this approach could also enable progressive lossy-to-lossless coding. A description of a recently introduced proposal is also provided. This proposal could be built upon the latest prediction-based CCSDS-123, easing its implementation by space agencies.

REFERENCES

[1] K. Sayood, *Introduction to Data Compression*, ser. The Morgan Kaufmann Series in Multimedia Information and Systems. Morgan Kaufmann; 4 edition, 2012.

[2] D. Salomon, *Data Compression: The Complete Reference*. Springer; 4 edition, 2006.

[3] B. Huang, Ed., *Satellite Data Compression*. Springer, 2011.

[4] S.-E. Qian, *Optical Satellite Data Compression and Implementation*. SPIE – The International Society for Optical Engineering, 2013.

[5] S.-E. Qian, *Optical Satellite Signal Processing and Enhancement*. SPIE – The International Society for Optical Engineering, 2013.

[6] Consultative Committee for Space Data Systems (CCSDS). [Online]. Available: http://www.ccsds.org

[7] D. Taubman and M. Marcellin, *JPEG2000: Image Compression Fundamentals, Standards, and Practice*. Kluwer, 2001.

[8] Consultative Committee for Space Data Systems, *Multispectral & Hyperspectral Data Compression Working Group*. CCSDS, 2007. [Online]. Available: http://cwe.ccsds.org/sls/default.aspx

[9] Consultative Committee for Space Data Systems (CCSDS), *Lossless Multispectral & Hyperspectral Image Compression CCSDS 123.0-B-1*, ser. Blue Book. CCSDS, May 2012. [Online]. Available: http://public.ccsds.org/publications/archive/123x0b1ec1.pdf

[10] M. Klimesh, "Low-complexity lossless compression of hyperspectral imagery via adaptive filtering," *IPN Progress Report*, vol. 42-163, pp. 1–10, 2005. [Online]. Available: http://ipnpr.jpl.nasa.gov/progress report/42-163/163H.pdf

[11] Consulative Committee for Space Data Systems (CCSDS), *Image Data Compression CCSDS 122.0-B-1*, ser. Blue Book. CCSDS, Nov. 2005.

[12] Consultative Committee for Space Data Systems (CCSDS), *Lossless Data Compression CCSDS 121.0-B-2*, ser. Blue Book. CCSDS, May 2012. [Online]. Available: http://public.ccsds.org/publications/archive/121x0b2.pdf

[13] *Lossless Data Compression*, cCSDS-121.0-B-1 Blue Book, May 1997.

[14] ISO/IEC, "JPEG-LS lossless and near-lossless compression for continuous-tone still images," 1999.

[15] M. Weinberger, G. Seroussi, , and G. Sapiro, "The LOCO-I lossless image compression algorithm: Principles and standardization into JPEG-LS," *IEEE Transactions on Image Processing*, vol. 9, no. 8, pp. 1309–1324, Aug. 2000.

[16] S. Golomb, "Run-length encodings," *IEEE Transactions on Information Theory*, vol. 12, pp. 399–401, July 1966.

[17] "Information technology - JPEG 2000 image coding system - part 1: Core coding system," ISO/IEC, Dec. 2000.

[18] E. Magli, G. Olmo, and E. Quacchio, "Optimized onboard lossless and near-lossless compression of hyperspectral data using CALIC," *IEEE Geoscience and Remote Sensing Letters*, vol. 1, no. 1, pp. 21–25, Jan. 2004.

[19] X. Wu and N. Memon, "Context-based, adaptive, lossless image coding," *IEEE Transactions on Communications*, vol. 45, no. 4, pp. 437–444, Apr. 1997.

[20] E. Augé, J. E. Sánchez, A. Kiely, I. Blanes, and J. Serra-Sagristà , "Performance impact of parameter tuning on the CCSDS-123 lossless multi- and hyperspectral image compression standard," *Journal of Applied Remote Sensing*, vol. 7, no. 1, pp. 074 594 1–074 594 16, 2013.

[21] D. Taubman, "Kakadu software," http://www.kakadusoftware.com/.

[22] GICI-UAB, "Spectral transform," http://gici.uab.cat/GiciWebPage/downloads.php.

[23] E. Magli, "M-calic," http://www1.tlc.polito.it/oldsite/sas-ipl/download.php/.

[24] GICI-UAB, "Emporda," http://gici.uab.cat/GiciWebPage/downloads.php.

[25] J. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3445–3462, December 1993.

[26] A. Said and W. A. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, pp. 243–250, June 1996.

[27] P.-S. Yeh, P. Armbruster, A. Kiely, B. Masschelein, G. Moury, C. Schaefer, and C. Thiebaut, "The New CCSDS Image Compression Recommendation," in *IEEE Aerospace Conference*, vol. 5-12, 2005, pp. 4138 – 4145.

[28] Consultative Committee for Space Data Systems, *Image Data Compression CCSDS 120.1-G-1*, ser. Green Book. CCSDS, Jun. 2007.

[29] "JPEG 2000 image coding system - Part 1: Core coding system, Information technology 15444-1. International Organization for Standardization / International Electrotechnical Commission (ISO/IEC); International Telecommunication Union-Telecom Standardization (ITU-T)," International Organization for Standardization / International Electrotechnical Commission (ISO/IEC); International Telecommunication Union-Telecom Standardization (ITU-T), December 2000.

[30] GICI-UAB, "Spectral transform," http://gici.uab.cat/GiciWebPage/downloads.php.

[31] F. García-Vilchez and J. Serra-Sagristà, "Extending the CCSDS recommendation for image data compression for remote sensing scenarios," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 47, pp. 3431–3445, October 2009.

[32] *JPEG 2000 Part 2 - Extensions*, document ISO/IEC 15444-2.

[33] S. Lim, K. Sohn, and C. Lee, "Principal component analysis for compression of hyperspectral images," in *Proc. of IGARSS - IEEE International Geoscience and Remote Sensing Symposium*, Sydney, Australia, 2001.

[34] J. Sagri, A. Tescher, and J. Reagan, "Practical transform coding of multispectral imagery," *IEEE Signal Processing Magazine*, pp. 32–43, Jan. 1995.

[35] P. Dragotti, G. Poggi, and A. Ragozini, "Compression of multispectral images by three-dimensional SPIHT algorithm," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 38, no. 1, pp. 416–428, Jan. 2000.

[36] L. Chang, C. Cheng, and T. Chen, "An efficient adaptive KLT for multispectral image compression," in *Proceedings of 4th IEEE Southwest Symposium on Image Analysis and Interpretation*, Austin, TX, 2000.

[37] P. Hao and Q. Shi, "Reversible integer KLT for progressive-to-lossless compression of multiple component images," in *Proc of IEEE International Conference on Image Processing, 2003*, Barcelona, Spain, 2003.

[38] B. Penna, T. Tillo, E. Magli, and G. Olmo, "Progressive 3D coding of hyperspectral images based on JPEG 2000," *IEEE Geoscience and Remote Sensing Letters*, vol. 3, no. 1, pp. 125–129, Jan. 2006.

[39] J. Rucker, J. Fowler, and N. Younan, "JPEG2000 coding strategies for hyperspectral data," in *Proc. of IEEE International Geoscience and Remote Sensing Symposium*, 2005.

[40] B. Penna, T. Tillo, E. Magli, and G. Olmo, "Embedded lossy to lossless compression of hyperspectral images using JPEG 2000," in *Proc. of IEEE International Geoscience and Remote Sensing Symposium*, 2005.

[41] Y. Wang, J. Rucker, and J. Fowler, "Three-dimensional tarp coding for the compression of hyperspectral images," *IEEE Geoscience and Remote Sensing Letters*, vol. 1, no. 2, pp. 136–140, Apr. 2004.

[42] I. Blanes and J. Serra-Sagristà, "Cost and scalability improvements to the karhunen-loève transform for remote-sensing image coding," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 48, pp. 2854–2863, July 2010.

[43] V. Goyal, "Theoretical foundations of transform coding," *IEEE Signal Processing Magazine*, vol. 18, no. 5, pp. 9–21, 2001.

[44] B. Penna, T. Tillo, E. Magli, and G. Olmo, "Transform coding techniques for lossy hyperspectral data compression," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 45, no. 5, pp. 1408–1421, May 2007.

[45] I. Blanes and J. Serra-Sagristà, "Pairwise orthogonal transform for spectral image coding," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, pp. 961–972, March 2011.

[46] X. Tang, C. Sungdae, and W. Pearlman, "3D set partitioning coding methods in hyperspectral image compression," in *Proc. of ICIP - IEEE International Conference on Image Processing*, Barcelona, Spain, 2003.

[47] X. Tang and W. Pearlman, "Three-dimensional wavelet-based compression of hyperspectral images," in *Hyperspectral Data Compression*. Kluwer Academic Publishers, 2005.

[48] R. Roger and J. Arnold, "Reversible image compression bounded by noise," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 32, no. 1, pp. 19–24, 1994.

[49] N. Jayant and P. Noll, *Digital Coding of Waveforms*. Prentice-Hall, 1984.

[50] X. Wu and P. Bao, "l_∞ constrained high-fidelity image compression via adaptive context modeling," *IEEE Transactions on Image Processing*, vol. 9, no. 4, pp. 536–542, Apr. 2000.

[51] B. Aiazzi, P. Alba, L. Alparone, and S. Baronti, "Lossless compression of multi/hyperspectral imagery based on a 3-D fuzzy prediction," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 37, no. 5, pp. 2287–2294, Sep. 1999.

[52] B. Aiazzi, L. Alparone, and S. Baronti, "Near-lossless compression of 3-D optical data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 39, no. 11, pp. 2547–2557, Nov. 2001.

[53] B. Aiazzi, L. Alparone, S. Baronti, and C. Lastri, "Crisp and fuzzy adaptive spectral predictions for lossless and near-lossless compression of hyperspectral imagery," *IEEE Geoscience and Remote Sensing Letters*, vol. 4, no. 4, pp. 532–536, Oct. 2007.

[54] A. Kim and T. Ramstad, "Improving the rate-distortion performance of DPCM using multirate processing with application in low-rate image coding," *IEEE Transactions on Signal Processing*, vol. 55, no. 10, pp. 4958–4968, Oct. 2007.

[55] J. Zhou, X. Wu, and L. Zhang, "l_2 restoration of l_∞ decoded images via soft-decision estimation," *IEEE Transactions on Image Processing*, vol. 21, no. 12, pp. 4797–4807, Dec. 2012.

[56] L. Lin and A. Ortega, "Bit-rate control using piecewise approximated rate-distortion characteristics," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, no. 4, pp. 446–459, Aug. 1998.

[57] K. Metin Uz, J. Shapiro, and M. Czigler, "Optimal bit allocation in the presence of quantizer feedback," in *Proc. of IEEE ICASSP*, 1993.

[58] D. Valsesia and E. Magli, "A novel rate control algorithm for onboard predictive coding of multispectral and hyperspectral images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 10, pp. 6341–6355, 2014.

[59] D. Valsesia and E. Magli, "A hardware-friendly architecture for onboard rate-controlled predictive coding of hyperspectral and multispectral images," in *Proc. of IEEE ICIP*, 2014.

[60] A. Alecu, A. Munteanu, J. Cornelis, S. Dewitte, and P. Schelkens, "Wavelet-based scalable L-infinity-oriented compression," *IEEE Transactions on Image Processing*, vol. 15, no. 9, pp. 2499–2512, Sep. 2006.

[61] G. Carvajal, B. Penna, and E. Magli, "Unified lossy and near-lossless hyperspectral image compression based on JPEG 2000," *IEEE Geoscience and Remote Sensing Letters*, vol. 5, no. 4, pp. 593–597, 2008.

[62] S. Tate, "Band ordering in lossless compression of multispectral images," *IEEE Transactions on Computers*, vol. 46, no. 4, pp. 477–483, 1997.

[63] P. Toivanen, O. Kubasova, and J. Mielikainen, "Correlation-based band-ordering heuristic for lossless compression of hyperspectral sounder data," *IEEE Geoscience and Remote Sensing Letters*, vol. 2, no. 1, pp. 50–54, Jan. 2005.

[64] J. Zhang and G. Liu, "An efficient reordering prediction-based lossless compression algorithm for hyperspectral images," *IEEE Geoscience and Remote Sensing Letters*, vol. 4, no. 2, pp. 283–287, Apr. 2007.

[65] A. Abrardo, M. Barni, A. Bertoli, R. Grimoldi, E. Magli, and R. Vitulli, "Low-complexity approaches for lossless and near-lossless hyperspectral image compression," in *Satellite Data Compression*. Springer, 2011.

[66] D. Slepian and J. Wolf, "Noiseless coding of correlated information sources," *IEEE Transactions on Information Theory*, vol. 19, pp. 471–480, Jul. 1973.

[67] E. Magli, M. Barni, A. Abrardo, and M. Grangetto, "Distributed source coding techniques for lossless compression of hyperspectral images," *EURASIP Journal on Advances in Signal Processing*, vol. 2007, 2007.

[68] A. Abrardo, M. Barni, E. Magli, and F. Nencini, "Error-resilient and low-complexity onboard lossless compression of hyperspectral images by means of distributed source coding," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 48, no. 4, pp. 1892–1904, 2010.

[69] X. Pan, R. Liu, and X. Lv, "Low-complexity compression method for hyperspectral images based on distributed source coding," *IEEE Geoscience and Remote Sensing Letters*, vol. 9, no. 2, pp. 224–227, 2012.

[70] R. Liu, J. Wang, and X. Pan, "Low complexity DCT-based distributed source coding with gray code for hyperspectral images," *Journal of Systems Engineering and Electronics*, vol. 21, no. 6, pp. 927–933, 2010.

[71] N.-M. Cheung, C. Tang, A. Ortega, and C. Raghavendra, "Efficient wavelet-based predictive Slepian-Wolf coding for hyperspectral imagery," *Signal Processing*, vol. 86, no. 11, pp. 3180–3195, Nov. 2006.