A hardware-friendly algorithm for compressing hyperspectral images

Conference Paper · October 2018

DOI: 10.1117/12.2500493

5 authors, including:

Raul Guerra Hernández
IUMA
31 PUBLICATIONS **176** CITATIONS
SEE PROFILE

Maria Diaz
Universidad de Las Palmas de Gran Canaria
18 PUBLICATIONS **59** CITATIONS
SEE PROFILE

Yubal Barrios
Universidad de Las Palmas de Gran Canaria
9 PUBLICATIONS **19** CITATIONS
SEE PROFILE

Sebastian Lopez
Universidad de Las Palmas de Gran Canaria
133 PUBLICATIONS **930** CITATIONS
SEE PROFILE

A hardware-friendly algorithm for compressing hyperspectral images

Raúl Guerra, María Díaz, Yubal Barrios, Sebastián López, Roberto Sarmiento[a]

[a]University of Las Palmas de Gran Canaria, Las Palmas, Spain

ABSTRACT

The on-board compression of remote sensed hyperspectral images is an important task nowadays. One of the main difficulties is that the compression of these images must be performed in the satellite which carries the hyperspectral sensor, where the available power, time, and computational resources are limited. Moreover, it is important to achieve high compression ratios without compromising the quality of the decompressed image for the ulterior hyperspectral imaging applications. The HyperLCA compressor aims to fulfill these requirements, providing an efficient lossy compression process that allows achieving very high compression ratios while preserving the most relevant information for the subsequent hyperspectral applications. One extra advantage of the HyperLCA compressor is that it allows to fix the compression ratio to be achieved. In this work, the effect of the specified compression ratio in the computational burden of the compressor has been evaluated, also considering the rest of the input parameters and configurations of the HyperLCA compressor. The obtained results verify that the computational cost of the HyperLCA compressor decreases for higher compression ratios, with independence of the specified configuration. Additionally, the obtained results also suggest that this compressor could produce real-time compression results for on-board applications.

Keywords: HyperLCA Compressor, hyperspectral compression, lossy compression, on-board compression, real-time compression.

1. INTRODUCTION

The capability of the hyperspectral sensors, carried on satellites, for collecting information across the electromagnetic spectrum provides very useful information for many applications related to the earth observation. However, the huge amount of data collected by these sensors must be stored on-board and then sent to the earth surface. Since the bandwidth of the connection is limited, as well as the memory available on the satellites, the on-board hyperspectral image compression is a very important and challenging task. First of all, due to the amount of data contained in these images, it is important to achieve high compression ratios, but without loosing too much information. Secondly, the compression must be performed on-board by space qualified hardware, which relies in area, time and power limitations. Due to these facts, the algorithms used for performing the on-board hyperspectral image compression should be a parallel process with low computational burden.

The Lossy Compression Algorithm for Hyperspectral Image Systems (HyperLCA)[1] has several features that make it a very suitable option for the on-board lossy hyperspectral imaging compression. On one side, this compressor is able to achieve very high compression ratios with relative high rate-distortion relations and perfectly preserving the most different hyperspectral pixels of the data set. This benefits many ulterior hyperspectral imaging applications, such as anomaly detection, unmixing or target detection, in which the most different pixels of the data set are specially useful.[2–7] On the other side, the HyperLCA compressor is able to independently process blocks of pixels of the image, without requiring any kind of spatial alignment of the pixels, what represents an important advantage when using pushbroom or whishbroom sensors since the hyperspectral data can be compressed as it is captured. Additionally, the HyperLCA compressor guaranties that the compression ratio obtained will be at least the desired minimal compression ratio specified as an input parameter.

Further author information: (Send correspondence to A.A.A.)
A.A.A.: E-mail: rguerra@tiuma.ulpgc.es, Telephone: +34 928 451220

Another important aspect of the HyperLCA compressor is its flexibility, which allows configuring different compression behaviors and efficiency according to the input parameters. This work focuses on evaluating the performance of the HyperLCA compressor according to the different input parameters, considering the impact of the different possible configurations, not only in the quality of the compression results but also in the efficiency and complexity of the algorithm.

2. HYPERLCA COMPRESSOR

The HyperLCA compressor divides the image in blocks of pixels which are independently compressed. The compression process independently applied to each individual block consists of three main compression stages, which are a spectral transform, a preprocessing stage and the entropy coding stage. The HyperLCA spectral transform sequentially selects the most different pixels of the hyperspectral data set using orthogonal projection techniques. The set of selected pixels is then used for projecting the hyperspectral image, obtaining a spectral decorrelated and compressed version of the data. The HyperLCA preprocessing stage is executed after the HyperLCA transform for adapting the output data for being entropy coded in a more efficient way. Finally, the entropy coding stage manages the codification of the extracted vectors using a Golomb-Rice coding strategy. Figure 1 graphically shows these three compression stages, as well as the data shared between them.

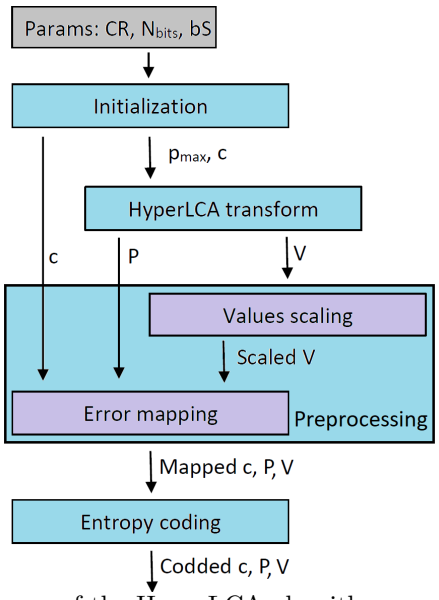


Figure 1: Diagram of the HyperLCA algorithm compression stages

The HyperLCA compressor uses three main parameters in oder to configure its compression performance:

1. *Minimum desired compression ratio (CR)*, defined as the relation between the number of bits in the real image and the number of bits of the compressed data.
2. *Block size (BS)*, which indicates the number of hyperspectral pixels in a single block.
3. *Number of bits used for scaling the projection vectors (N_{bits}).* This value determines the precision and dynamic range to be used for representing the values of the V vectors.

Once that the HyperLCA compressor has been correctly configured, it firstly determines the number of pixels vectors and projection vectors (p_{max}) to be extracted for each block, as shown in Equation 1, where DR refers to the number of bits per pixel per band of the hyperspectral image to be compressed. The extracted pixel vectors are referred as P and the projection vectors are referred as V in the rest of the paper. Once that p_{max} has been

obtained, the average pixel, also called centroid, c, is computed for each block of pixels to be compressed. This data is used as inputs of the HyperLCA transform, which is the most relevant part of the HyperLCA compressor.

$$p_{\max} \leq \frac{DR \cdot (N_b \cdot (BS - CR)}{CR \cdot (DR \cdot N_b + N_{\text{bits}} \cdot bS)} \tag{1}$$

The HyperLCA transform provides most of the compression ratio obtained by the HyperLCA compressor and also most of its flexibility and advantages. Additionally, it is the only lossy part of the HyperLCA compression process. The HyperLCA transform is detailed described in Algorithm 1. This pseudocode assumes that the image block is stored as a matrix, M, with the hyperspectral pixels placed in columns.

First of all, the HyperLCA transform subtracts the average pixel to all the pixels of the block to be compressed, obtaining the centralized image block (M_c), as shown in line 1 of Algorithm 1. After doing so, the HyperLCA compression process mainly consists of three steps which are sequentially repeated, as shown in Figure 2. First, the brightest pixel in M_c, which is the pixel with more remaining information, p_i, is selected. After doing so, the vector v_i is calculated as the projection of the image, M_c, in the direction spanned by p_i. Finally, the information of the image that can be represented with the extracted p_i and v_i vectors is subtracted from the M_c, as shown in line 11 of Algorithm 1.

Algorithm 1 HyperLCA transform.

Inputs:
$M = [r_1, ..., r_{N_p}]$, $p_{\max}$, c
Outputs:
$P = [c, p_1, ..., p_{p_{\max}}]$, $V = [v_1, ..., v_{p_{\max}}]$
Declarations:
$P = [p_1, ..., p_{p_{\max}}]$; {Extracted pixels.}
$V = [v_1, ..., v_{p_{\max}}]$; {Projected image vectors.}
$M_c = [x_1, ..., x_{N_p}]$ {Centralized version of M}
Algorithm:
1: {Additional stopping condition initialization.}
2: **for** $i = 1$ **to** $p_{\max}$ **do**
3: **for** $j = 1$ **to** N_p **do**
4: $b_j = {x_j}^t \cdot x_j$
5: **end for**
6: $j_{\max} = \arg\max(b_j)$
7: $p_i = r_{j_{\max}}$
8: $q = x_{j_{\max}}$
9: $u = x_{j_{\max}} / (({x_{j_{\max}}})^t \cdot x_{j_{\max}})$
10: $v_i = u^t \cdot M_c$
11: $M_c = M_c - v_i \cdot q$
12: {Additional stopping condition checking.}
13: **end for**

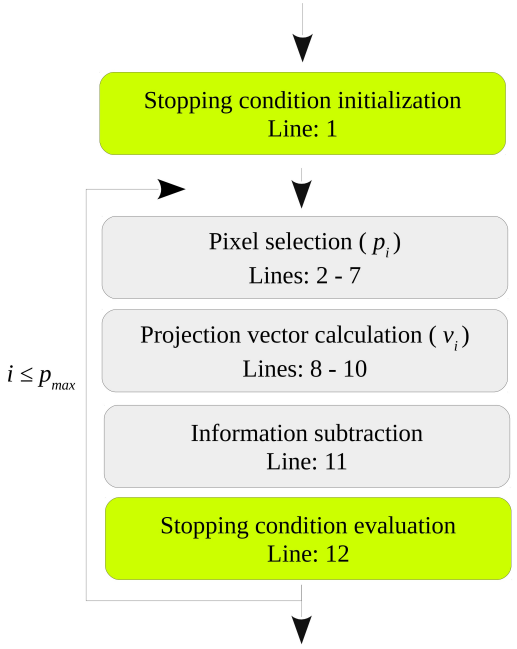


Figure 2: Flowchart describing the subprocesses executed in the HyperLCA transform.

Accordingly, M_c contains the information that is not representable with the already selected pixels, P, and V vectors. Hence, the values of M_c in a particular iteration, i, would be the information lost in the compression-

decompression process if no more pixels p_i and v_i vectors were extracted. This fact makes it relatively simple to add extra stopping conditions based on quality metrics such as the *Maximum Absolute Difference*, *MAD*, or the *Signal to Noise Ratio*, *SNR*, as shown in Figure 2. These metrics would check in every iteration of the HyperLCA transform if the amount of information remaining in M_c is high, and more p_i and v_i vectors are required, or if it is low enough and the algorithm may stop.

Once that the HyperLCA transform stage finishes, the extracted vectors, $P = [c, p_1, ..., p_{p_{\max}}]$ and $V = [v_1, ..., v_{p_{\max}}]$, are preprocessed and entropy coded, as described in Figure 1.

3. HYPERLCA CONFIGURATIONS TESTED IN THIS WORK

The HyperLCA compressor can be configured in many different ways in order to provide different compression behaviors and efficiency with the goal of adapting its performance to the necessities of the targeted applications. This work focuses on evaluating the performance of the HyperLCA compressor according to its different possible configurations, measuring both the quality of its compression results and also its efficiency and complexity.

3.1 Basic configurations

As described in Section 2, the HyperLCA has three main input parameters that need to be set before starting the compression process, the *minimum desired compression ratio*, *CR*, the *block size*, *BS*, and the *number of bits used for scaling the projection vectors*, N_{bits}. Once that these parameters have been fixed, the compressor calculates the maximum number of p_i and v_i vectors to be extracted for each block of pixels, $p_{\max}$, as shown in Equation 1. Then, the HyperLCA transform performs an iterative process in which a p_i and v_i vectors are extracted in each iteration. Different conclusions can be directly dragged from this process.

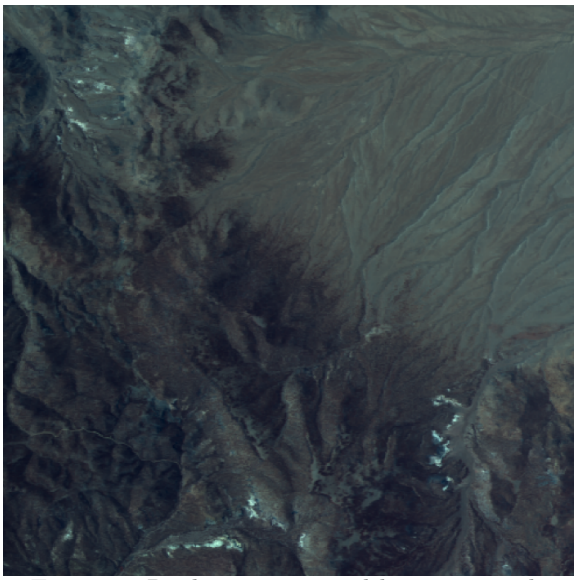
First of all, as it can be seen in Equation 1, higher *CR* values produce smaller $p_{\max}$ values and less iterations are carried out by the HyperLCA transform. Accordingly, the increment in the compression ratio decreases the computational burden of the algorithm since less operations are to be executed. However, increasing the compression ratio will also increase the amount of information lost in the compression-decompression process within the HyperLCA compressor. The compression ratio obtained in the HyperLCA compression process will be always higher that the introduced *CR* value, hence, the *CR* input parameter can be understood as the *minimal desired compression ratio.*

Secondly, for the same *CR* and N_{bits} values, increasing the number of hyperspectral pixels per block, *BS*, will produce a higher $p_{\max}$ value. This means that more p_i and v_i vectors can be extracted for the same *CR* and N_{bits} values by increasing the block size. However, increasing *BS* may increase not only the number of iterations, $p_{\max}$, to be executed by the HyperLCA transform, increasing the computational burden, but also the required memory and the latency of the entire compression process.

Finally, according to Equation 2, for the same *CR* and *BS*, decreasing N_{bits} will produce a higher $p_{\max}$ value. This means that more p_i and v_i vectors can be extracted for the same compression ratio and block size by decreasing the number of bits used for representing the values of the v_i vectors. For a computational point of view, increasing the $p_{\max}$ value will increase the number of iterations to be executed by the HyperLCA transform as well as its computational burden. However, increasing the $p_{\max}$ value may also improve the quality of the compression results since more p_i and v_i vectors can be used for representing the data. It is also important to consider that, reducing N_{bits} will decrease the precision used for representing the values of the v_i vectors what may affect the quality of the compression results. Additionally, since the v_i vectors have *BS* components, the N_{bits} value will have a higher impact for bigger block sizes (higher *BS* values).

These three parameters corresponds with the basic configuration of the HyperLCA compressor. According to the specified values, the number of p_i and v_i vectors to be extracted for each block of pixels, $p_{\max}$, is fixed. The HyperLCA transform will extract the same number of p_i and v_i vectors ($p_{\max}$) for every block of pixels if no more stopping conditions are used. This kind of configuration provides a relatively uniform compression ratio among the different blocks of pixels, however, this may not be the optimal solution in some situations. In example, some blocks of pixels could be very homogeneous and could be accurately represented using a small number of p_i and v_i vectors. However, other blocks of pixels of the same image could have a high entropy and would required a higher number of p_i and v_i vectors for reducing the information lost during the compression-decompression

AVIRIS - Lunar Lake



Hyperion - Lake Monona

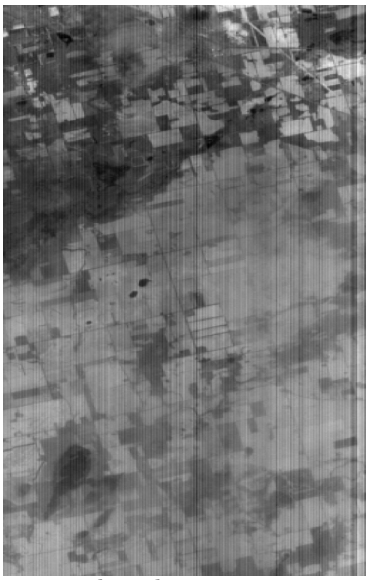


Figure 3: Real remote sensed hyperspectral images used in the experiments.

process. Since the same $p_{\max}$ is used for all the blocks of the same image, if the $p_{\max}$ value is high, too many p_i and v_i vectors will be used for the homogeneous blocks, decreasing the achieved CR. On the contrary, if the $p_{\max}$ value is low, a higher compression ratio will be achieved but the more entropy blocks will not be so accurately represented. In order to provide a better solution for these kind of situations, the HyperLCA compressor allows introducing extra stopping conditions based on quality metrics as described in the next section.

3.2 Extra stopping conditions based on quality metrics

In addition to the basic configurations specified in Section 3.1, the HyperLCA compressor allows setting additional stopping conditions based on quality metrics in a relative simple way. This is due to the fact that HyperLCA transform keeps track of the information that will be lost in the compression-decompression process. As explained in Section 2, the M_c matrix contains the information that is not representable with the already selected, P, and V vectors. Due to this reason, the values of M_c in a particular iteration, i, would be the information lost in the compression-decompression process if no more p_i and v_i vectors were extracted. This can be used for adding extra stopping conditions based on quality metrics. Three different quality metrics which are widely used for evaluating the performance of the compression-decompression processes have been used in this work. These metrics are the *Maximum Absolute Difference*, (MAD), the *Peak Signal-to-Noise Ratio*, (PSNR), and the *Signal-to-Noise Ratio*, (SNR). As shown in Figure 2, the values remaining in the M_c matrix are evaluated in each iteration according to the specified quality metric. Additionally, some initial calculation may be needed at the beginning of the HyperLCA transform process. The process for adding these three metrics are described in detail in Sections 3.2.1, 3.2.2 and 3.2.3.

It is important to have in mind that, with independence of the quality metric used as extra stopping condition, the maximum number of P and V vectors, $p_{\max}$, is determined by the basic input parameters, RC, BS and N_{bits}. Due to this reason, the HyperLCA compressor will never produce a lower compression ratio than the specified CR input parameter, even if the quality of the compression-decompression process, specified by the extra stopping condition based on a quality metric, has not been achieved. However, the extra stopping conditions allow using different number of p_i and v_i vectors, i, lower than $p_{\max}$, for the different blocks. This fact may be specially beneficial for images in which some blocks of pixels are very homogeneous and can be well compressed using very few P and V vectors, but the other blocks have higher entropy and require more vectors for being accurately compressed. This provides to additional ways of using the HyperLCA compressor according to the input parameters:

1. *Maximizing the compression ratio obtained for each block of pixels.* When the targeted application does not require a really accurate compression-decompression process, one extra stopping condition can be used together with a high input *CR* value for optimizing the compression performance of the HyperLCA algorithm and obtaining higher compression ratios for the homogeneous blocks. For doing so, the value fixed for the quality metric used as stopping condition should indicate the *maximum desired quality*. Hence, if the specified quality is achieved in the iteration $i < p_{\max}$ of the HyperLCA transform, the algorithm would stop and a higher compression ratio would be obtained.

2. *Minimizing the losses of information produced in each block in the compression-decompression process.* When the targeted applications requires an accurate compression-decompression process, the *CR* parameter can be set with a small value, in such a way that a high number of P and V vectors could be used for representing each block of pixels (high $p_{\max}$ value). Additionally, an extra stopping condition can be set with the value that defines the desired quality for the compression-decompression process. The HyperLCA transform will iteratively extract p_i and v_i vectors until the extra stopping condition is satisfied or until the number of extracted vectors reaches the $p_{\max}$ value. If $p_{\max}$ is high enough, the desired compression-decompression quality could be achieved for all the blocks of the image, however, the achieved compression ratio could be low.

Finally, it is important to mention that the information lost in the compression-decompression process may be positive for some applications. Most of the lossy compressors behaves as a low pass filters, reducing the amount of noise in the image, what positively affects most of the ulterior hyperspectral imaging applications.[8] The HyperLCA compressor is not the exception. However, the low pass filter behavior of the lossy compressor typically results in losing the anomalous pixels or the image details, which are crucial for many hyperspectral imaging applications such as anomaly detection, target detection or spectral unmixing.[2–7] In this sense, the HyperLCA compressor provides an important advantage since the most different pixels of each block are preserved without losses of information.

3.2.1 MAD based stopping condition

The *MAD* metric is widely used in hyperspectral compression applications for measuring the maximum absolute error produced in the compression-decompression process. This error is the maximum absolute difference between the original image and the compressed-decompressed one. The application of this metric as an extra stopping condition to the HyperLCA transform is a pretty simple process, since the maximum absolute error that would be obtained in a particular iteration, if no more iterations were performed, is directly the maximum absolute value of M_c. According to the HyperLCA transform flowchart described in Figure 2, this metric does not require any kind of initialization. The evaluation process consists in measuring the maximum absolute value of M_c and comparing it with the user defined *MAD* value.

3.2.2 PSNR based stopping condition

The *PSNR* metric is usually employed in compression applications for measuring the overall accuracy of the compression-decompression process. It measures the average quadratic error in relation with the maximum possible value that could be present in the image, as described in Equation 2, where M and M' refer to the real hyperspectral image and to the compressed-decompressed one, and N_p and N_b refer to the number of pixels and the number of bands in the hyperspectral image, respectively. This metric is expressed in decibels.

$$\mathrm{SNR} = 10 \cdot \log_{10}\left(\frac{(\text{maximum possible value of M})^2}{\frac{1}{N_b \cdot N_p} \cdot \sum_{i=1}^{N_b} \sum_{j=1}^{N_p} (M_{i,j} - M'_{i,j})^2}\right) \tag{2}$$

According to the previously described characteristics of the HyperLCA transform, the difference between M and M' for each block of pixels corresponds to M_c. Hence, Equation 2 can be expressed as shown in Equation 3 for each block of the image.

$$\text{SNR} = 10 \cdot \log_{10}\left(\frac{(\text{maximum possible value of M})^2}{\frac{1}{N_b \cdot BS}\sum_{i=1}^{N_b}\sum_{j=1}^{N_p}(M_{c_{i,j}})^2}\right) \tag{3}$$

In order to reduce the required calculations in each iteration of the HyperLCA transform when using the *PSNR* as extra stopping condition, an initialization step can be used as shown in Figure 2. This initialization consists in calculating the value that $\sum_{i=1}^{N_b}\sum_{j=1}^{N_p}(M_{c_{i,j}})^2$ should achieved for reaching the specified PSNR value, in such a way that just $\sum_{i=1}^{N_b}\sum_{j=1}^{N_p}(M_{c_{i,j}})^2$ is calculated in each iteration of the HyperLCA transform and compare with the obtained stopping value. This stopping value can be calculated as:

$$\text{stopping value} = \frac{1}{N_b \cdot BS} \cdot \frac{(\text{maximum possible value of M})^2}{10^{(PSNR/10)}} \tag{4}$$

3.2.3 SNR based stopping condition

Similarly to the *PSNR* metric, the *SNR* is also used for measuring the overall accuracy of the compression-decompression process. It measures the average quadratic error in relation with the average quadratic value of the original image, expressed in decibels. Equation 5 describes the calculation of the *SNR* metric, where M and M' refer to the real hyperspectral image and to the compressed-decompressed one, and N_p and N_b refer to the number of pixels and the number of bands in the hyperspectral image, respectively.

$$\text{SNR} = 10 \cdot \log_{10}\left(\frac{\sum_{i=1}^{N_b}\sum_{j=1}^{N_p}(M_{i,j})^2}{\sum_{i=1}^{N_b}\sum_{j=1}^{N_p}(M_{i,j} - M'_{i,j})^2}\right) \tag{5}$$

The *SNR* metric for each block of pixels of the image can be calculated during the HyperLCA transform process using the M_c values too. Equation 6 describes this calculation.

$$\text{SNR} = 10 \cdot \log_{10}\left(\frac{\sum_{i=1}^{N_b}\sum_{j=1}^{N_p}(M_{i,j})^2}{\sum_{i=1}^{N_b}\sum_{j=1}^{N_p}(M_{c_{i,j}})^2}\right) \tag{6}$$

As it is done for the *PSNR*, in order to reduce the amount of operations executed by the HyperLCA transform in each iteration when using the *SNR* as additional stopping condition, an initialization step can be used. This initialization consists in calculating the value that $\sum_{i=1}^{N_b}\sum_{j=1}^{N_p}(M_{c_{i,j}})^2$ should achieved for reaching the specified *SNR* value, in such a way that just $\sum_{i=1}^{N_b}\sum_{j=1}^{N_p}(M_{c_{i,j}})^2$ is calculated in each iteration of the HyperLCA transform and compare with the obtained stopping value. This stopping value can be calculated as described in Equation 7, where M^* refers to the original values of each block of pixels processed by the HyperLCA transform.

$$\text{stopping value} = \frac{1}{N_b \cdot BS} \cdot \frac{\sum_{i=1}^{N_b}\sum_{j=1}^{N_p}(M^*_{i,j})^2}{10^{(PSNR/10)}} \tag{7}$$

3.2.4 Computational complexity of the additional stopping conditions based on quality metrics

The used of extra stopping conditions based on quality metrics may considerably improve the compression performance of the HyperLCA algorithm, specially in situations in which different blocks of the image have very different entropy levels. However, the use of these kind of metrics may also notably increase the complexity and computational requirements of the HyperLCA transform. Within the three metrics tested as additional stopping conditions in this work, the *MAD* represents the less computational demanding one. First of all, this metric does not require any kind of initialization. Furthermore, the additional calculations required in each iteration of the HyperLCA transform consist just in evaluating the maximum absolute value of the M_c matrix. On the other hand, the additional calculations required in each iteration of the HyperLCA transform due to the use of the *PSNR* and *SNR* metrics involve the calculation of the addition of all the squared values of M_c, which is more computational demanding. Furthermore, these two metrics also require an extra initialization that involve using relatively complex operations in relation with the operations used in the rest of the algorithm. Additionally, the addition of all the squared values of M^* is also required in the initialization step when using the *SNR* metric.

4. EXPERIMENTS AND RESULTS

Several experiments have been carried out in this work in order to evaluate the performance of the HyperLCA compressor according to the different input parameters, considering the impact of the different possible configurations, not only in the quality of the compression results but also in the efficiency and complexity of the algorithm. Two different hyperspectral images, collected by two different sensors, have been used for such purpose. On one side, the Lunar Lake image from the well known Airbone Visible/Infrarred Imaging Spectrometer (AVIRIS)[9] has been selected. This sensor captures 224 spectral bands in the wavelength range of 400 to 2500 nm, coding each pixel value using 16 bits.[9] This image has been cropped to a portion of 512x512 pixels. Figure 3 graphically shows a false color representation of this image. On the other side, the Lake Monona image from the Hyperion sensor[10] have been used. The Hyperion sensor produces 242 spectral bands between 355.59 and 2577.08 nm, coding the image values using 12 bits. This image is uncalibrated, what makes the compression process more challenging. Figure 3 shows a gray scale representation of the Lake Monona image. This image has been cropped to a portion of 512x256 pixels.

These two images have been compressed with the HyperLCA compressor using a wide range of configurations. The quality of the obtained compression results has been evaluated using the *Maximum Absolute Difference*, *MAD*, the *Peak Signal to Noise Ratio*, *PSNR*, and the *Signal to Noise Ratio*, *SNR*, metrics. It is important to clarify that these metrics have been calculated for the entire image in the evaluation process, not block by block. However, when these metrics are used by the HyperLCA transform during the compression process, the HyperLCA transform independently measures them for each single block. In order to measure the computational burden introduced by the different possible configurations the time required for compressing the entire images has been measured.

Image		Input CR		Input BS		Input Nbits	
Lunar Lake (AVIRIS)		12		1024		12	
Input	Outputs						
CR	Pmax	CR	bpppb	MAD	PSNR	SNR	Time (s)
8	28	10.55	1.52	42	85.91	51.23	13.42
12	19	15.50	1.03	49	84.69	50.01	10.02
16	14	20.98	0.76	64	83.54	48.86	7.58
20	11	26.67	0.60	113	82.34	47.66	6.27
BS	Pmax	CR	bpppb	MAD	PSNR	SNR	Time (s)
1024	19	15.50	1.03	49	84.69	50.01	10.01
512	15	16.68	0.96	67	83.90	49.22	8.06
256	10	18.85	0.85	120	82.62	47.94	5.70
Nbits	Pmax	CR	bpppb	MAD	PSNR	SNR	Time (s)
16	15	14.75	1.09	64	83.86	49.18	8.26
12	19	15.50	1.03	49	84.69	50.01	10.08
8	25	17.65	0.91	49	81.94	47.26	12.89

Table 1: Compression results obtained for the Lunar Lake image, collected by the AVIRIS sensor, using different configuration parameters.

The first performed experiments focus on evaluating the behavior of the HyperLCA compressor in its basic configuration and the impact of the *CR*, *BS* and N_{bits} input parameters. For doing so, each of these parameters has been modified while keeping the rest of them stable. The default stable values that have been used for these parameters are $CR = 12$, $BS = 1024$ and $N_{\text{bits}} = 12$ for the AVIRIS Lunar Lake image, and $CR = 12$, $BS = 1024$ and $N_{\text{bits}} = 8$ for the Hyperion Lake Monona image. The *CR* values have been varied to 8, 12, 16

and 20. The *BS* values has been set to 1024, 512 and 256, and the N_{bits} to 16, 12 and 8, what results in a total of 10 different basic configurations for each hyperspectral image used.

Image		Input CR		Input BS		Input Nbits	
Lake Monona (Hyperion)		12		1024		8	
Input	Outputs						
CR	Pmax	CR	bpppb	MAD	PSNR	SNR	Time (s)
8	33	11.52	1.04	44	60.29	45.56	8.62
12	22	17.51	0.69	148	59.35	44.62	5.84
16	16	24.34	0.49	184	58.62	43.90	4.35
20	13	30.12	0.40	203	58.12	43.40	3.57
BS	Pmax	CR	bpppb	MAD	PSNR	SNR	Time (s)
1024	22	17.51	0.69	148	59.35	44.62	5.69
512	17	17.51	0.69	169	59.30	44.57	4.51
256	11	18.37	0.65	197	58.83	44.10	3.08
Nbits	Pmax	CR	bpppb	MAD	PSNR	SNR	Time (s)
16	12	15.67	0.77	203	57.93	43.21	3.37
12	16	15.79	0.76	186	58.67	43.94	4.49
8	22	17.51	0.69	148	59.35	44.62	5.71

Table 2: Compression results obtained for the Lake Monona image, collected by the Hyperion sensor, using different configuration parameters.

Tables 1 and 2 show the results obtained for the AVIRIS Lunar Lake and Hyperion Lake Monona images, respectively. As it can be seen in both tables, when decreasing the specified *CR* for the same *BS* and N_{bits} values, the number of extracted P and V vectors, p_{max}, increases and so does the quality of the compression results and the time required for compressing the image. On the contrary, the obtained compression ratio decreases (higher *bpppb* and lower output *CR*). This makes totally sense since more P and V vectors are being used for representing each block of pixels of the image. Additionally, it is also important to highlight that the obtained *CR* is always slightly higher than the specified.

The values displayed in Tables 1 and 2 also show that increasing the *BS* for the same *CR* and N_{bits} values allows extracting more P and V vectors for each block of pixels (higher p_{max}). This results in slightly higher compression accuracy but lower compression ratio (higher *bpppb*). Despite the *BS* value does not have a strong impact in the compression performance, it has a higher impact in the time required for compressing the images. Higher *BS* and p_{max} require more time for carrying out the compression of the images, also increasing the required memory and the latency of the compression process.

Tables 1 and 2 also show the impact of the N_{bits} parameter in the HyperLCA compression process. In general, decreasing the N_{bits} value increases p_{max}, using more P and V vectors for representing the data, but using less precision for storing these V vectors. The best compression results seems to be obtained using N_{bits} values slightly lower than the number of bits used for representing the values of the real hyperspectral images. In example, $N_{\text{bits}} = 12$ produces the best compression results for the Lunar Lake image, collected by the AVIRIS sensor, which is represented using 16 bits per pixel per band. However, $N_{\text{bits}} = 8$ produces the best results for the Hyperion Lake Monona image, which is represented using 12 bits per pixel per band. Additionally, since decreasing the N_{bits} results in higher p_{max} values, it also results in higher computational times.

After evaluating the behavior of the HyperLCA compressor in its basic configuration and the impact of the *CR*, *BS* and N_{bits} parameters, additional experiments have been carried out for evaluating the performance of the compressor when using extra stopping conditions based on the *MAD*, *PSNR* and *SNR* quality metrics. For

Image		Input CR		Input BS		Input Nbits	
Lunar Lake (AVIRIS)		12		1024		12	
Input	Outputs						
Reference	Pmax	CR	bpppb	MAD	PSNR	SNR	Time (s)
	19	15.50	1.03	49	84.69	50.01	10.02
MAD	Pmax	CR	bpppb	MAD	PSNR	SNR	Time (s)
50	19	25.48	0.63	53	82.52	47.84	6.69
100	19	33.19	0.48	101	80.97	46.29	5.34
150	19	52.26	0.31	152	76.28	41.60	3.95
200	19	78.96	0.20	203	71.72	37.04	2.94
PSNR	Pmax	CR	bpppb	MAD	PSNR	SNR	Time (s)
75	19	61.52	0.26	488	76.45	41.77	4.72
80	19	37.10	0.43	179	80.35	45.67	7.56
85	19	16.03	1.00	52	84.58	49.90	15.00
90	19	15.50	1.03	49	84.69	50.01	15.14
SNR	Pmax	CR	bpppb	MAD	PSNR	SNR	Time (s)
40	19	63.25	0.25	488	75.99	41.31	4.52
45	19	37.99	0.42	194	80.04	45.36	6.83
50	19	17.54	0.91	64	84.19	49.52	13.54
55	19	15.50	1.03	49	84.69	50.01	15.19

Table 3: Compression results obtained for the Lunar Lake image, collected by the AVIRIS sensor, using different stopping criteria based on quality metrics.

doing so, the CR, BS and N_{bits} parameters have been fixed to $CR = 12$, $BS = 1024$ and $N_{\text{bits}} = 12$ for the AVIRIS Lunar Lake image, and $CR = 12$, $BS = 1024$ and $N_{\text{bits}} = 8$ for the Hyperion Lake Monona image. When using the MAD metric as extra stopping condition, its input value has been set to 50, 100, 150 and 200, which are relatively low values in relation with the possible values of the images ($2^{16} - 1$ for the AVIRIS Lunar Lake image and $2^{12} - 1$ for the Hyperion Lake Monona one). When using the $PSNR$ metric as extra stopping condition, its input value has been set to 75, 80, 85 and 90 for both images. These values are more restrictive for the image collected by the Hyperion sensor since the maximum possible value of this image ($2^{12} - 1$) is considerably smaller than the maximum possible value of the image collected by the AVIRIS sensor ($2^{16} - 1$). Finally, when using the SNR metric as extra stopping condition, its input value has been set to 40, 45, 50 and 55. Tables 3 and 4 show the obtained results. Row labeled as *Reference* in these two tables refers to the results obtained with the default configuration parameters without using any extra stopping condition.

Different conclusions can be dragged from the results displayed in Tables 3 and 4. First of all, it can be observed that when an extra stopping condition based on a quality metric is used, using an unachievable value, the compression performance is the same as without using any extra stopping condition, but the time required for compressing the image increases due to the extra calculations. An example of this behavior is when specifying $PSNR = 90$ or $SNR = 55$ for the Lunar Lake image, or $PSNR \geq 75$ and $SRN \geq 45$ for the Lake Monona one. On the contrary, when specifying very easily achievable values, the compression ratio drastically increases, but the quality of the compression-decompression process decreases according to the specified value. This is due to the fact that very few P and V vectors (much less than p_{max}) are being used for each block of the image. Accordingly, since much less iterations are being executed by the HyperLCA transform, the time required for the compression also decreases. An example of this behavior can be seen when using the MAD metric as extra condition, specifying an input value of $MAD = 200$, for both images. This behavior is specially useful when

Image		Input CR		Input BS		Input Nbits	
Lake Monona (Hyperion)		12		1024		8	
Input	Outputs						
Reference	Pmax	CR	bpppb	MAD	PSNR	SNR	Time (s)
	22	17.51	0.69	148	59.35	44.62	5.84
MAD	Pmax	CR	bpppb	MAD	PSNR	SNR	Time (s)
50	22	52.52	0.23	148	56.07	41.35	2.33
100	22	78.03	0.15	148	52.45	37.73	1.77
150	22	102.06	0.12	152	47.73	33.00	1.49
200	22	133.95	0.09	200	45.10	30.37	1.25
PSNR	Pmax	CR	bpppb	MAD	PSNR	SNR	Time (s)
75	22	17.51	0.69	148	59.35	44.62	9.43
80	22	17.51	0.69	148	59.35	44.62	9.67
85	22	17.51	0.69	148	59.35	44.62	9.38
90	22	17.51	0.69	148	59.35	44.62	9.35
SNR	Pmax	CR	bpppb	MAD	PSNR	SNR	Time (s)
40	22	62.30	0.19	148	55.34	40.61	3.09
45	22	18.59	0.65	148	59.22	44.50	8.93
50	22	17.51	0.69	148	59.35	44.62	9.42
55	22	17.51	0.69	148	59.35	44.62	9.41

Table 4: Compression results obtained for the Lake Monona image, collected by the Hyperion sensor, using different stopping criteria based on quality metrics.

high compression ratios (low *bpppb*) are desired and when the accuracy of the compression process required for the ulterior hyperspectral imaging applications is not hight. The minimal required compression accuracy should be the specified as an extra stopping condition based on the desired quality metric.

Finally, the third possible situation is that in which the value specified for the extra stopping condition are only achievable by part of the blocks of pixels of the image. In this situation, the very homogeneous blocks are rapidly compressed using very few P and V vectors, preserving a high compression quality, while very entropy blocks are compressed using the maximum possible number of P and V vectors, $p_{\max}$. In average, the compression performance in this situation seems to be much better, since much higher compression ratios are obtained (lower *bpppb* values) but without decreasing too much the accuracy of the compression results. Additionally, the extra time required by the calculation of the extra stopping conditions is compensated by the iterations saved by the HyperLCA transform in the homogeneous blocks. An example of this behavior can be seen when using $MAD = 50$ or $MAD = 100$ for both images. Additionally, it can also be observed that the use of the *MAD* metric as extra stopping condition in the HyperLCA transform produces lower computational times than the use of the *PSNR* and *SNR* metrics. This makes totally sense due to the higher computational complexity of the *PSNR* and *SNR* metrics.

All these results have been obtained using a regular Intel Core i7-4790 Processor, which is far from being the kind of hardware available on-board satellites. Nevertheless, the differences in the compression time obtained using the different configurations of the HyperLCA algorithm provides a clear idea of the differences in their computational burden. Additionally, it is worth to mention that these results have been obtained using a standard C++ code without any kind of parallelization. The slower results obtained in the simulations, 15.19 seconds for the Lunar Lake image and 9.67 seconds for the Lake Monona one, correspond with a compression data rates of 7.37 MB/s and 4.69 MB/s, respectively. The fastest results, 2.94 and 1.25 seconds for the Lunar Lake and Lake

Monona images, correspond with compression data rates of 38.2 and 36.3 MB/s. These values are relatively high in relation with the regular acquisition rates of the hyperspectral sensors. According to these results, and due to the low computational complexity and high level of parallelism of the HyperLCA algorithm, it is considered that this compressor could be efficiently parallelized and implemented for hyperspectral imaging compression on-board satellites.

5. CONCLUSSIONS

The HyperLCA compressor is a lossy compression solution with multiple advantages for compressing remote sensed hyperspectral images. In particular, this algorithm presents a high level of parallelism and low computational complexity that could make it a viable option for hyperspectral imaging compression on-board satellites. One extra advantage of the HyperLCA compressor is that the minimum desired compression ratio can be fixed in advance. Additionally, this compressor can be configured in many different ways according to the input parameters, what provides a high flexibility with the goal of adapting its performance to the requirements of different applications.

In this work, the effect of the specified compression ratio in the computational burden of the compressor and the quality of the compression results has been evaluated, also considering the rest of the input parameters and configurations of the HyperLCA compressor. These configurations have been tested using two different remote sensed hyperspectral images with very different characteristics, the Lunar Lake image and the Lake Monona one, collected by the AVIRIS and Hyperion sensors, respectively. The quality of the obtained results have been measured using the *MAD*, *PSNR* and *SNR* metrics. The computational cost of the different configurations of the HyperLCA algorithm has been compared according to the time required for compressing the images.

The obtained results verify that the computational cost of the HyperLCA compressor decreases for higher compression ratios, with independence of the specified configuration. Additionally, it has been proved that the different configuration parameters can be used for adapting the performance of the compressor to the requirements of the targeted applications. In example, the size of the blocks of pixels in which the image is divided, *BS*, can be decreased for reducing the required memory and computational burden of the algorithm, while the precision used for representing the values of the extracted V vectors, N_{bits} can be decreased for performing more iterations and achieving more accurate compression results. Additionally, the use of extra stopping conditions has been also tested. As demonstrated by the experiments, these extra stopping conditions can be very powerful for optimizing the performance of the compressor according to the necessities of the application. Finally, the obtained results also suggest that this compressor could produce real-time compression results for on-board applications.

ACKNOWLEDGMENTS

This research is partially funded by the European Commission through the ECSEL Joint Undertaking (ENABLE-S3 project, no. 692455) and the Ministry of Economy and Competitiveness (MINECO) of the Spanish Government (ENABLE-S3 project, no. PCIN-2015-225; and PLATINO project, no. TEC2017-86722-C4-1-R). This work was completed while María Díaz was beneficiary of a pre-doctoral grant given by the Agencia Canaria de Investigacin, Innovacin y Sociedad de la Informacin (ACIISI) of the Conserjería de Economía, Industria, Comercio y Conocimiento of the Gobierno de Canarias, which is part-financed by the European Social Fund (FSE) (POC 2014-2020, Eje 3 Tema Prioritario 74 (85%)).

REFERENCES

[1] Guerra, R., Barrios, Y., Díaz, M., Santos, L., López, S., and Sarmiento, R., "A new algorithm for the on-board compression of hyperspectral images," *Remote Sensing* **10**(3), 428 (2018).

[2] Aiazzi, B., Alparone, L., and Baronti, S., "Quality issues for compression of hyperspectral imagery through spectrally adaptive dpcm," in [*Satellite Data Compression*], 115–147, Springer (2012).

[3] Lee, C., Lee, S., and Lee, J., "Effects of lossy compression on hyperspectral classification," in [*Satellite Data Compression*], 269–285, Springer (2012).

[4] Guerra, R., López, S., and Sarmiento, R., "A computationally efficient algorithm for fusing multispectral and hyperspectral images," *IEEE Transactions on Geoscience and Remote Sensing* **54**(10), 5712–5728 (2016).

[5] García-Vílchez, F., Muñoz-Marí, J., Zortea, M., Blanes, I., González-Ruiz, V., Camps-Valls, G., Plaza, A., and Serra-Sagristà, J., "On the impact of lossy compression on hyperspectral image classification and unmixing," *IEEE Geoscience and remote sensing letters* **8**(2), 253–257 (2011).

[6] Du, Q., Ly, N., and Fowler, J. E., "An operational approach to pca+ jpeg2000 compression of hyperspectral imagery," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **7**(6), 2237–2245 (2014).

[7] Díaz, M., Guerra, R., López, S., and Sarmiento, R., "An algorithm for an accurate detection of anomalies in hyperspectral images with a low computational complexity," *IEEE Transactions on Geoscience and Remote Sensing* **56**(2), 1159–1176 (2018).

[8] Santos, L., López, S., Callico, G. M., López, J. F., and Sarmiento, R., "Performance evaluation of the h. 264/avc video coding standard for lossy hyperspectral image compression," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **5**(2), 451–461 (2012).

[9] "Jet propulsion laboratory, nasa. airbone visible infrared imaging spectrometer website. [online]. http://aviris.jpl.nasa.gov/aviris/index.html."

[10] "U.s. geological survey and nasa. earth observing 1, hyperion website. [online]. https://eo1.usgs.gov/sensors/hyperion."