Reducing Data Dependencies in the Feedback Loop of the CCSDS 123.0-B-2 Predictor

Antonio Sánchez, Ian Blanes, Yubal Barrios, Miguel Hernández-Cabronero, Joan Bartrina-Rapesta, Joan Serra-Sagristà, *Senior Member, IEEE*, and Roberto Sarmiento

***Abstract*—On-board multi- and hyperspectral instruments acquire large volumes of data that need to be processed with the limited computational and storage resources. In this context, the CCSDS 123.0-B-2 standard emerges as an interesting option to compress multi- and hyperspectral images on-board satellites, supporting both lossless and near-lossless compression with low complexity and reduced power consumption. Nonetheless, the inclusion of a feedback loop in the CCSDS 123.0-B-2 predictor to support near-lossless compression introduces significant data dependencies that hinder real-time processing, particularly due to the presence of a quantization stage within this loop. This work provides an analysis of the aforementioned data dependencies and proposes two strategies aiming at maximizing throughput in hardware implementations and thus enabling real-time processing. In particular, through an elaborate mathematical derivation, the quantization stage is removed completely from the feedback loop. This reduces the critical path, which allows for shorter initiation intervals in a pipelined hardware implementation and higher throughput. This is achieved without any impact in the compression performance, which is identical to the one obtained by the original data flow of the predictor.**

***Index Terms*—Hyperspectral imaging, compression algorithms, CCSDS 123.0-B-2, on-board data processing.**

I. Introduction

SPECTROSCOPIC imaging sensors are very common in Earth Observation (EO) space missions due to the relevant information acquired in the spectral domain for characterization and monitoring purposes, among many others [1], [2]. The handling of the ever-increasing volumes of data produced by these instruments with the often-limited on-board storage resources and downlink capacities motivate the necessity of using compression techniques that efficiently reduce data volumes before being sent to the ground [3], [4].

Compression solutions for space missions are not straightforward to develop [5]. In addition to the usual compression requirements, these solutions are required to have low-enough complexity to fit well on spaceborne hardware, to have limited power consumption, and to have robustness against radiation-induced faults. The Consultative Committee for Space Data Systems (CCSDS), a worldwide organization formed by world-leading space agencies, has published several data compression standards specifically conceived for space applications, focusing on high compression performance at affordable computational complexities.

Among these standards, the CCSDS 123.0-B-2 [6] specifies a low-complexity algorithm to compress multi- and hyperspectral images in both lossless and near-lossless modes. It is composed of two main stages: a predictor that exploits spectral and spatial redundancy, and an entropy coder that encodes prediction residuals. Image samples are handled sequentially, and the predictor estimates the value of the current *input sample* $s_z(t)$ considering previously processed samples in its spatial and spectral neighborhood.

The CCSDS 123.0-B-2 standard introduces several differences with respect to its predecessor CCSDS 123.0-B-1, mainly to support near-lossless compression. These differences introduce challenging data dependencies for a high-throughput hardware implementation. A key novelty is the introduction of a quantizer within the prediction loop, which controls data losses through a maximum error limit that can be absolute or relative to the sample magnitude. This requires the calculation of the so-called *sample representatives* $s''_z(t)$, which are a reconstruction of the input samples after quantization. These are required so that identical predictions can be performed by a decoder, where the original values of the input samples are not necessarily available. The standard also introduces *narrow* local sums, which limit the predictor use of the sample representatives to the left of the value being predicted. This eases data dependencies and favours hardware optimization strategies that improve throughput [7].

Regarding the entropy coding stage, the CCSDS 123.0-B-2 standard defines three different encoders, including a novel hybrid entropy encoder that provides higher compression ratios than the other two for low-entropy data [8]. Entropy encoders are out of the scope of this manuscript.

The main contributions of this letter are a data dependency analysis for the prediction stage of the CCSDS 123.0-B-2 standard, and two implementation strategies that reduce the critical path in the predictor's feedback loop. In one strategy, the quantization and reconstruction are effectively removed from the feedback loop, and in another the feedback loop is further reduced through speculative execution. These optimizations

This work has been supported by ESA (contract 4000136723/22/NL/CRS), by the European Union's Horizon 2020 research and innovation programme (grant agreement No 776151) and though the Marie Skłodowska-Curie (grant agreement #801370), by the Spanish Ministry of Science and Innovation and by the European Regional Development Fund (RTI2018-095287-B-I00 and PID2021-125258OB-I00), and by the Government of Catalonia though the Beatriu de Pinós programme (2018-BP-00008).

A. Sánchez, Y. Barrios and R. Sarmiento are with the Institute for Applied Microelectronics (IUMA), University of Las Palmas de Gran Canaria (ULPGC), 35017 Las Palmas de Gran Canaria, Spain (e-mail: {ajsanchez, ybarrios, roberto}@iuma.ulpgc.es).

I. Blanes, M. Hernández-Cabronero, J. Bartrina-Rapesta and J. Serra-Sagristà are with the Universitat Autònoma de Barcelona, 08193 Cerdanyola del V., Spain (e-mail: {ian.blanes, miguel.hernandez, joan.bartrina, joan.serra}@uab.cat).

Manuscript received XXX; revised XXX.

This article has been accepted for publication in IEEE Geoscience and Remote Sensing Letters. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/LGRS.2022.3213975

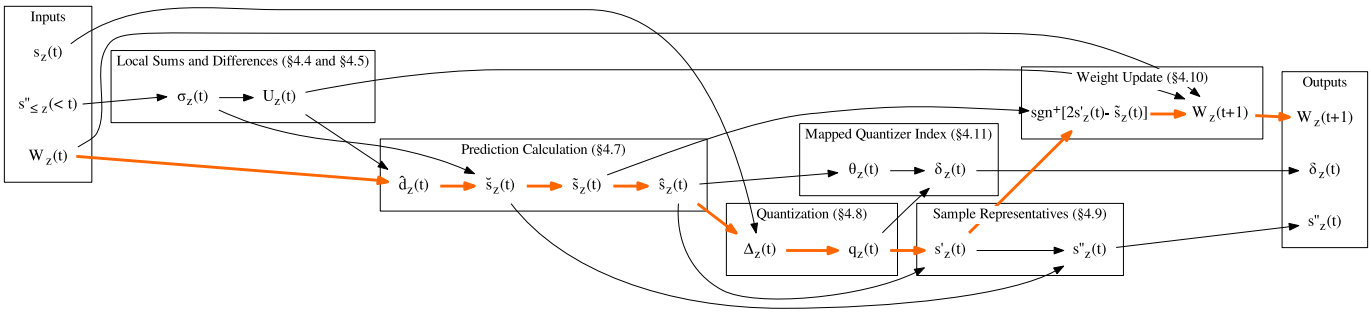

Fig. 1: Data dependency diagram for the CCSDS 123.0-B-2 predictor. The critical path for successive samples within the same band is highlighted in bold orange.

allow to directly process images collected in Band-Interleaved by Line (BIL) order, without any extra reordering stage and thus enabling compression on-the-fly. The application of both strategies on [9] shows an increase in maximum theoretical throughput without affecting the output of the prediction stage.

This paper employs the same notation and naming conventions as the CCSDS 123.0-B-2 standard. Readers not familiar with the standard may obtain additional descriptions within the standard itself [6], [7], while additional implementation information is available in [10], [9], [11].

The rest of the paper is structured as follows. The data dependency analysis is detailed in Section II, while the two implementation strategies are described in Section III. Finally, Section IV summarizes the main conclusions of this work.

II. Analysis of Data Dependencies

The use of sample representatives in CCSDS 123.0-B-2 imposes significant challenges in the processing data path and introduces data dependencies between successive samples that might impact the complexity and performance of hardware implementations. In this Section, the main bottlenecks found in the predictor stage of the standard are analyzed.

The data dependencies that occur during the prediction of one sample in CCSDS 123.0-B-2 are depicted in Fig. 1, where data flows are represented by arrows between the intermediate results. For a given sample at index t –in raster-scan order– within band z, the predictor inputs are the input sample $s_z(t)$ itself, some sample representatives $s''_{\leq z}(< t)$ of previously scanned locations, and the weight vector $W_z(t)$. For the same sample, the predictor outputs are an updated weight vector $W_z(t+1)$, a mapped quantizer index $\delta_z(t)$, i.e., the predicted value, and a sample representative $s''_z(t)$ to be used in future predictions instead of the current sample. Note that $s''_{\leq z}(< t)$ are used to compute local sums and differences, and the computation of $s''_z(t)$ is particularly onerous, requiring a quantization operation and thus involving a divider.

A key aspect in the evaluation of data dependencies is the order in which input samples are processed. This order may directly follow the order of the input samples as produced by a given multi- or hyperspectral sensor, or it may be the result of a careful rearrangement of the input data into an order better suited to obtain high throughput [11]. Hence, instead of focusing our analysis on the three common data processing orders (i.e., BIP, BIL, and BSQ orders), this letter studies data dependencies according to whether dominant data dependencies are of *spectral* (i.e., samples processed with consecutive values of z) or *spatial* nature (i.e., samples processed with consecutive values of t). Implementations using careful data rearrangements may benefit too from our analysis and from the derived implementation strategies described later.

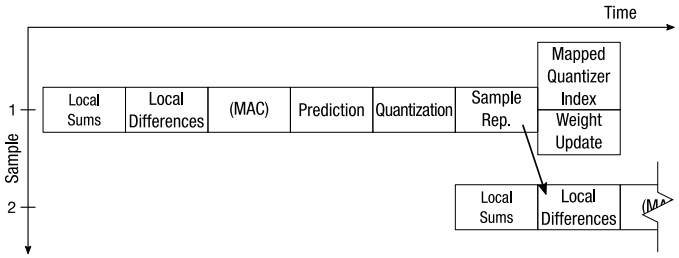

Fig. 2: Pipeline disposition when processing data for samples in the same locations of consecutive spectral bands. An arrow denotes the principal data dependency. The MAC employed in the prediction calculation is displayed as a separate stage preceding the prediction stage.

A. Spectral data dependencies

A careful analysis reveals that predictions at index t of band z require the sample representative $s''_{z-1}(t)$ to compute the central local differences in $U_z(t)$, which in turn is required to compute $s''_z(t)$. This introduces a strong dependency within the data path for samples at the same spatial locations of consecutive spectral bands, in which a sample must be almost fully processed before the processing of the next one can start, as shown in Fig. 2. This imposes a strong limitation in terms of throughput that can be hardly avoided. The only manner in which to dispose of this strong dependency is by setting the number of previous bands for prediction P to 0. However, this solution completely disables inter-band prediction, resulting in unacceptable compression penalties.

Other sample representatives $s''_{z-1}(t')$ for $t' < t$ are required as well to compute local sums, but the use of narrow local sums and the use of the column-oriented prediction mode allow for differences between t' and t of approximately the width of the image. Hence, when determining data compression orders, it seems necessary to avoid those that consecutively process input samples with strong spectral data dependencies (e.g., the BIP order).

This article has been accepted for publication in IEEE Geoscience and Remote Sensing Letters. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/LGRS.2022.3213975

$$\text{sgn}^+[e_z(t)] = \begin{cases} -1 & \text{if } \left|s_z(t) - \hat{s}_z(t)\right| \leq m_z(t) \text{ and } \tilde{s}_z(t) \text{ is odd,} \\ 1 & \text{if } \left|s_z(t) - \hat{s}_z(t)\right| \leq m_z(t) \text{ and } \tilde{s}_z(t) \text{ is even,} \\ \text{sgn}^+\left[s_z(t) - \hat{s}_z(t)\right] & \text{otherwise.} \end{cases} \quad (1)$$

B. Spatial data dependencies

Here, the main data dependency is caused by the weight vector $W_z(t)$ being employed to produce weight vector $W_z(t+1)$ in a convoluted succession of data dependencies (see Fig. 1). This limits the achievable throughput because weights are employed to obtain $\hat{d}_z(t)$ using an inner product of vectors, which cannot be computed until weights related to sample $t-1$ are obtained.

Spatial data dependencies are strongly influenced by the selected local sum type and prediction mode, which may cause additional dependencies due to the use of neighboring sample representatives. The strongest data dependency is found in local sums in the wide neighbor-oriented mode, where the sample representative $s''_z(t-1)$ of the next sample to the left must be obtained to compute the local sum. Parallelization possibilities are limited in this configuration, demanding for almost-serial implementations. This also applies for the first line of an image in wide column-oriented mode. Conveniently, narrow local sums do not present this data dependency.

Another dependency related with sample representatives occurs due to the use of the full prediction mode. In this case, directional local differences are required, and again the sample representative $s''_z(t-1)$ is used.

The most favorable configuration is the combination of narrow local sums and reduced prediction mode. In this situation, the spatial dependency on sample representatives is mostly removed with the limiting factor being weight dependencies. Nonetheless, as explained in Section III, the dependency on weights can be significantly alleviated, reducing the critical path by removing the quantization result from the dependencies of $W_z(t+1)$. It is worth noting that this change does not affect the result of the weight updating nor the prediction outcome.

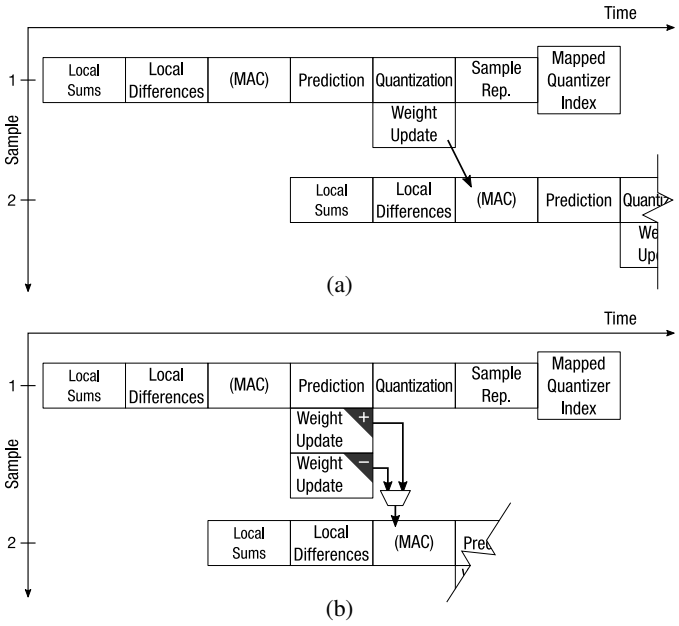

Fig. 3: Pipeline disposition when processing data for samples in consecutive locations of the same spectral band (a), and with eager execution of weight updates (b).

III. Implementation strategies

Two implementation strategies are proposed to shorten the critical path of a CCSDS 123.0-B-2 predictor implementation, achieving a higher throughput in hardware implementations without affecting the predictor outputs.

The first implementation strategy is related to the use of an equivalent definition of the first part of the weight update step, i.e., the calculation of $\text{sgn}^+[2s'_z(t) - \tilde{s}_z(t)]$. This expression requires results from the prediction calculation step, but also from the sample representatives step, which in turn requires results from the quantization step. The equivalent definition for $t > 0$ is shown in (1). Thus, this calculation can be now obtained from $s_z(t)$, $\hat{s}_z(t)$ and $\tilde{s}_z(t)$ instead of $s'_z(t)$ and $\tilde{s}_z(t)$.

This redefinition originates from the main idea that quantizing a prediction residual $\Delta_z(t)$ does not alter the sign of the result (except for when the quantizer yields a zero quantizer index), and that the sign of $\Delta_z(t)$ can be employed in the weight update stage (except when the quantizer yields a zero quantizer index) without having to wait for quantization and sample representative stages. While the redefinition is very simple, its derivation is extensive and for reader's convenience is left out of the main text and available in Annex A. As shown in Fig. 3a, using this new definition effectively removes the quantization step from the feedback loop and allows for tight pipelining strategies.

Additionally, a second cumulative strategy may be applied to the weight update step. Similar to [12], eager execution may be employed on the weight update calculation given that the result of sgn^+ is binary. Therefore, the computation of both possible weight update results can be done in parallel and in advance, and the selection of the correct one is performed after the result of the sgn^+ is obtained. This allows to reduce the initiation interval between samples in a pipelined implementation, improving the performance at the cost of duplicating the logic used for weight updating. Trough both strategies, the loop of dependencies is reduced to the computation of the MAC and prediction modules, as depicted in Fig. 3b.

Experimental simulation demonstrates the goodness of the proposed optimization strategies. Considering as baseline the work conducted in [9], in which the CCSDS 123.0-B-2 predictor runs at a maximum clock frequency of 125 MHz, a speed up x7 is achieved. More detailed results are reflected in Table I. Furthermore, these strategies do not alter the output of the prediction stage, and thus do not alter any rate-distortion results.

TABLE I: Experimental results.

	Baseline	Proposal
Critical path (cycles per sample)	7	**1**
Speed (Msamples/s) @125 MHz	17.86	**125**

IV. Conclusion

This letter examines the data dependencies within the CCSDS 123.0-B-2 prediction stage, and it shows that existing spectral data dependencies are to be avoided within an implementation critical path. On the other hand, existing spatial data dependencies, while apparently equally burdensome, may be eased through two proposed implementation strategies. These strategies effectively remove the predictor quantizer out of an implementation's feedback loop and make use of eager execution to achieve higher implementation throughput (up to 1 clock cycle per sample), while producing bit-identical compressed files.

The selection of an appropriate processing order that combines well with the proposed implementation strategies is a interesting topic of future research. The BIL order is a clear candidate, since it perfectly fits the optimizations presented in this work. Its goodness will be demonstrated in a hardware implementation, which is on-going.

Appendix A

This appendix proves (1) for $t > 0$. We start by stating some required equations from the CCSDS 123.0-B-2 standard. We use the symbol '§' to indicate section numbers referring to the standard document [6].

From § 1.6.1, § 4.10.1, § 4.9.2 § 4.7.4 and § 4.8.1 respectively we have that

$$\mathrm{sgn}^+[x] = \begin{cases} 1 & \text{if } x \geq 0, \\ -1 & \text{if } x < 0, \end{cases} \tag{2}$$

$$e_z(t) = 2s'_z(t) - \tilde{s}_z(t), \tag{3}$$

$$s'_z(t) = \mathrm{clip}\big(\hat{s}_z(t) + q_z(t)(2m_z(t)+1), \{s_{\min}, s_{\max}\}\big). \tag{4}$$

$$\hat{s}_z(t) = \lfloor \tilde{s}_z(t)/2 \rfloor, \tag{5}$$

$$q_z(t) = \mathrm{sgn}\big(\Delta_z(t)\big)\Big\lfloor \big(|\Delta_z(t)| + m_z(t)\big)/\big(2m_z(t)+1\big)\Big\rfloor, \tag{6}$$

and

$$\Delta_z(t) = s_z(t) - \hat{s}_z(t). \tag{7}$$

By substituting (4) and (6) into (3) we obtain

$$e_z(t) = \mathrm{clip}\Big(K_2 + 2\cdot \mathrm{sgn}\left(\Delta_z(t)\right)\cdot K_1 \cdot K_3, \Big\{2s_{\min} - \tilde{s}_z(t), 2s_{\max} - \tilde{s}_z(t)\Big\}\Big), \tag{8}$$

with

$$K_1 = \Big\lfloor \big(|\Delta_z(t)| + m_z(t)\big)/\big(2m_z(t)+1\big)\Big\rfloor, \tag{9}$$

$K_2 = 2\,\hat{s}_z(t) - \tilde{s}_z(t)$, and $K_3 = (2m_z(t)+1)$.

At this point we are interested to test whether the expression in (8) is strictly positive or not, as this is the only information needed to obtain $\mathrm{sgn}^+[e_z(t)]$. Before moving to study the value of $\mathrm{sgn}^+[e_z(t)]$, we calculate some bounds that will be of use latter.

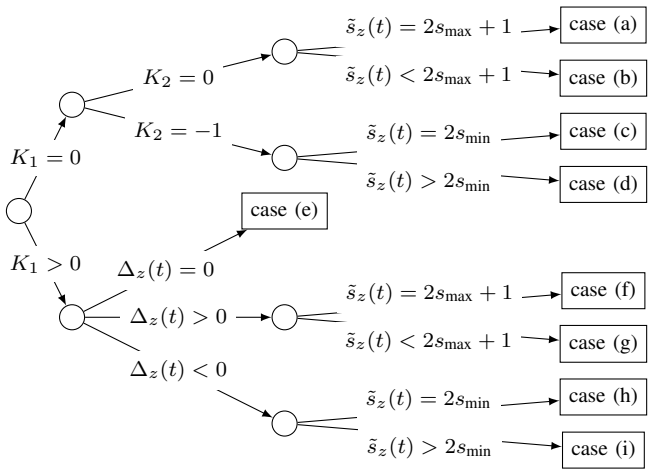

Fig. 4: Cases of study of equation (8).

It is straightforward to see that $K_1 \geq 0$, and from (5) we can also see that

$$K_2 = 2\left\lfloor \frac{\tilde{s}_z(t)}{2} \right\rfloor - \tilde{s}_z(t) = \begin{cases} -1 & \text{if } \tilde{s}_z(t) \text{ is odd,} \\ 0 & \text{if } \tilde{s}_z(t) \text{ is even,} \end{cases} \tag{10}$$

and as $m_z(t) \geq 0$ from § 4.8.2, we can also see that

$$K_3 \geq 1. \tag{11}$$

From § 4.7.3 we have that $\tilde{s}_z(t) = \lfloor \breve{s}_z(t)/2^{\Omega+1} \rfloor$, and from the clipping boundaries defined for $\breve{s}_z(t)$ in § 4.7.2 we have $2^{\Omega+2}s_{\min} \leq \breve{s}_z(t) \leq 2^{\Omega+1}(2s_{\max}+1)$, which allows us to infer that $\lfloor 2^{\Omega+2}s_{\min}/2^{\Omega+1} \rfloor \leq \tilde{s}_z(t) \leq \lfloor 2^{\Omega+1}(2s_{\max}+1)/2^{\Omega+1} \rfloor$, and that

$$2s_{\min} \leq \tilde{s}_z(t) \leq 2s_{\max} + 1. \tag{12}$$

Corollary 1. *For the minimum clipping boundary $2s_{min} - \tilde{s}_z(t)$ in* (8)*, from* (12) *we can see that either $2s_{min} = \tilde{s}_z(t)$ implying $2s_{min} - \tilde{s}_z(t) = 0$, or $2s_{min} < \tilde{s}_z(t)$ implying $2s_{min} - \tilde{s}_z(t) < 0$. I.e., strictly negative values are only turned to zero in the clip function in* (8) *when $2s_{min} = \tilde{s}_z(t)$, and remain strictly negative otherwise.*

Corollary 2. *Similarly, for the maximum clipping boundary $2s_{max} - \tilde{s}_z(t)$ in* (8)*, from* (12) *we can see that either $\tilde{s}_z(t) = 2s_{max} + 1$ implying $2s_{max} - \tilde{s}_z(t) = -1$, or $\tilde{s}_z(t) < 2s_{max} + 1$ implying $2s_{max} - \tilde{s}_z(t) \geq 0$. I.e., non-negative values are only turned to strictly negative in the clip function in* (8) *when $\tilde{s}_z(t) = 2s_{max} + 1$, and remain non-negative otherwise.*

At this point we are ready to analyze (8) case by case. Fig. 4 presents a flowchart which separates the study of (8) into mutually exclusive cases. Note that at each decision point, all possible options are taken into account. We now analyze each case individually:

(a) This case is impossible. From § 3.3.2, the value of $s_{\max}$ is either $2^D - 1$ or $2^{D-1} - 1$, with D an integer not smaller than 2. Hence $s_{\max}$ is odd and $\tilde{s}_z(t) = 2s_{\max}+1$ is odd too, which is incompatible with $K_2 = 0$.

(b) As $K_1 = 0$ and $K_2 = 0$, we can see that $e_z(t) = \mathrm{clip}\big(0, \{2s_{\min} - \tilde{s}_z(t), 2s_{\max} - \tilde{s}_z(t)\}\big)$, which by Corollary 2 we see is equivalent to $e_z(t) = 0$, as $\tilde{s}_z(t) < 2s_{\max} + 1$.

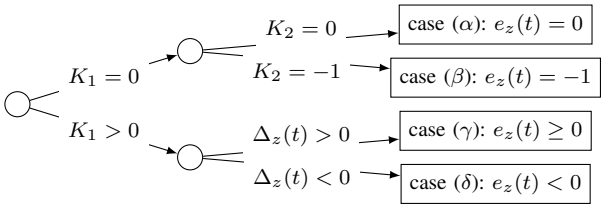

Fig. 5: Summary of the results of the case by case study of (8).

(c) This case is impossible. From § 3.3.2, the value of s_{min} is either 0 or -2^{D-1}, with D an integer not smaller than 2. Hence s_{min} is even and $\tilde{s}_z(t) = 2s_{\text{min}}$ is even too, which is incompatible with $K_2 = -1$.

(d) As $K_1 = 0$ and $K_2 = -1$, we can see that $e_z(t) = \text{clip}\big(-1, \{2s_{\text{min}} - \tilde{s}_z(t), 2s_{\text{max}} - \tilde{s}_z(t)\}\big)$, which by Corollary 1 we see is equivalent to $e_z(t) = -1$, as $\tilde{s}_z(t) > 2s_{\text{min}}$.

(e) This case is impossible. In this case $K_1 > 0$, hence $K_1 = \big\lfloor \big(|\Delta_z(t)| + m_z(t)\big) / \big(2m_z(t)+1\big) \big\rfloor > 0$, but if $\Delta_z(t) = 0$, we can see that $K_1 = \lfloor m_z(t)/(2m_z(t)+1) \rfloor = 0$, as $m_z(t) \geq 0$ from § 4.8.2.

(f) This case is impossible. Having $\Delta_z(t) > 0$ implies $s_z(t) > \hat{s}_z(t)$, and $s_z(t) > \big\lfloor \tilde{s}_z(t)/2 \big\rfloor$. From § 3.3.2 we have that $s_z(t) \leq s_{\text{max}}$, thus $\big\lfloor \tilde{s}_z(t)/2 \big\rfloor < s_{\text{max}}$. In this case we have that $\tilde{s}_z(t) = 2s_{\text{max}} + 1$, but substituting our previous finding into this equation yields $\tilde{s}_z(t) = 2s_{\text{max}} + 1 > 2 \left\lfloor \frac{\tilde{s}_z(t)}{2} \right\rfloor + 1$, which is impossible.

(g) As $K_3 \geq 1$ from (11), as K_1 is an integer strictly larger than 0 (i.e., $K_1 \geq 1$), and as $\Delta_z(t) > 0$ we have that $K_2 + 2 \leq K_2 + 2\,\text{sgn}\,(\Delta_z(t)) \cdot K_1 \cdot K_3$. As $K_2 \in \{-1, 0\}$, we have that $1 \leq K_2 + 2\,\text{sgn}\,(\Delta_z(t)) \cdot K_1 \cdot K_3$. I.e., the clipping operation on (8) is applied to a value not smaller than 1. From Corollary 2, as $\tilde{s}_z(t) < 2s_{\text{max}} + 1$, we see that $e_z(t) \geq 0$.

(h) This case is impossible. $\Delta_z(t) < 0$ implies $s_z(t) < \hat{s}_z(t)$, and $s_z(t) < \big\lfloor \tilde{s}_z(t)/2 \big\rfloor$. From § 3.3.2 we have that $s_{\text{min}} \leq s_z(t)$, thus $s_{\text{min}} < \big\lfloor \tilde{s}_z(t)/2 \big\rfloor$. In this case we have that $\tilde{s}_z(t) = 2s_{\text{min}}$, but substituting our previous finding into this equation yields $\tilde{s}_z(t) = 2s_{\text{min}} < 2\big\lfloor \tilde{s}_z(t)/2 \big\rfloor$, which is impossible for $\tilde{s}_z(t)$ even and $\tilde{s}_z(t) = 2s_{\text{min}}$ makes $\tilde{s}_z(t)$ even.

(i) As $K_3 \geq 1$ from (11), as K_1 is an integer strictly larger than 0 (i.e., $K_1 \geq 1$), and as $\Delta_z(t) < 0$ we have that $K_2 + 2\,\text{sgn}\,(\Delta_z(t)) \cdot K_1 \cdot K_3 \leq K_2 - 2$. As $K_2 \in \{-1, 0\}$, we have that $K_2 + 2\,\text{sgn}\,(\Delta_z(t)) \cdot K_1 \cdot K_3 \leq -2$. I.e., the clipping operation on (8) is applied to a value non-larger than -2. From Corollary 1, as $\tilde{s}_z(t) > 2s_{\text{min}}$, we see that $e_z(t) < 0$.

The results of this case by case analysis of equation (8) is detailed in Fig. 5. There are only four cases remaining, and for each one the value of $\text{sgn}^+[e_z(t)]$ is known.

For the last part of the proof, we show that (1) yields the correct result for each of the cases in Fig. 5. From (9) we can see that $K_1 = 0$ is equivalent to

$$K_1 = \left\lfloor \big(|\Delta_z(t)| + m_z(t)\big) / \big(2m_z(t)+1\big) \right\rfloor = 0. \qquad (13)$$

As the dividend in (13) is non-negative and the divisor is strictly positive, $|\Delta_z(t)| + m_z(t) < 2m_z(t) + 1$ implies that $K_1 = 0$. Hence, for the first two cases of (1), we see that $K_1 = 0$.

For the first case of (1) we also know that $\tilde{s}_z(t)$ is odd, and thus, per (10), that $K_2 = -1$. This means that for the first case of (1) we are in case (β) of Fig. 5. For this case, $\text{sgn}^+\big[-1\big] = -1$.

For the second case of (1) we know that $K_1 = 0$ and that $\tilde{s}_z(t)$ is even, and thus, per (10), that $K_2 = 0$. This means that for the second case of (1) we are in case (α) of Fig. 5. For this case, $\text{sgn}^+\big[0\big] = 1$.

For the third case of (1) we know that $|s_z(t) - \hat{s}_z(t)| \geq m_z(t) + 1$ and hence that $K_1 > 0$. For this third case, there are two subcases:

- If $\Delta_z(t) = s_z(t) - \hat{s}_z(t) > 0$ we are in case (γ). In this case we know that $e_z(t) \geq 0$ thus $\text{sgn}^+[e_z(t)] = 1$, thus (1) yielding $\text{sgn}^+[s_z(t) - \hat{s}_z(t)] = 1$ is the correct result.
- If $\Delta_z(t) = s_z(t) - \hat{s}_z(t) < 0$ we are in case (δ). In this case we know that $e_z(t) < 0$ thus $\text{sgn}^+[e_z(t)] = -1$, thus (1) yielding $\text{sgn}^+[s_z(t) - \hat{s}_z(t)] = -1$ is the correct result.

At this point we have proven that all cases in (1) yield the correct result. ■

REFERENCES

[1] M. Parente, J. Kerekes, and R. Heylen, "A Special Issue on Hyperspectral Imaging [From the Guest Editors]," *IEEE Geosci. Remote Sens. Mag.*, vol. 7, no. 2, pp. 6–7, 2019.

[2] M. J. Khan, H. S. Khan, A. Yousaf, K. Khurshid, and A. Abbas, "Modern Trends in Hyperspectral Image Analysis: A Review," *IEEE Access*, vol. 6, pp. 14 118–14 129, 2018.

[3] S.-E. Qian, "Introduction to hyperspectral satellites," in *Hyperspectral Satellites and System Design*. CRC Press, Taylor & Francis Group, 2020, ch. 1, pp. 1–52.

[4] G. Denis, A. Claverie, X. Pasco, J.-P. Darnis, B. de Maupeou, M. Lafaye, and E. Morel, "Towards disruptions in Earth observation? New Earth Observation systems and markets evolution: Possible scenarios and impacts," *Acta Astronautica*, vol. 137, pp. 415–433, 2017.

[5] I. Blanes, E. Magli, and J. Serra-Sagrista, "A Tutorial on Image Compression for Optical Space Imaging Systems," *IEEE Geosci. Remote Sens. Mag.*, vol. 2, no. 3, pp. 8–26, 2014.

[6] Consultative Committee for Space Data Systems, *Low-Complexity Lossless and Near-Lossless Multispectral and Hyperspectral Image Compression, CCSDS 123.0-B-2*. CCSDS, February 2019.

[7] M. Hernández-Cabronero, A. B. Kiely, M. Klimesh, I. Blanes, J. Ligo, E. Magli, and J. Serra-Sagristà, "The CCSDS 123.0-B-2 "Low-Complexity Lossless and Near-Lossless Multispectral and Hyperspectral Image Compression" Standard: A comprehensive review," *IEEE Geosci. Remote Sens. Mag.*, vol. 9, no. 4, pp. 102–119, 2021.

[8] P. Chatziantoniou, A. Tsigkanos, and N. Kranitis, "A high-performance RTL implementation of the CCSDS-123.0-B-2 hybrid entropy coder on a space-grade SRAM FPGA," in *Proceedings of the 7th OBPDC*, 2020, pp. 21–23.

[9] Y. Barrios, A. Sánchez, R. Guerra, and R. Sarmiento, "Hardware Implementation of the CCSDS 123.0-B-2 Near-Lossless Compression Standard Following an HLS Design Methodology," *Remote Sensing*, vol. 13, no. 21, p. 4388, 2021.

[10] I. Blanes, A. Kiely, M. Hernández-Cabronero, and J. Serra-Sagristà, "Performance Impact of Parameter Tuning on the CCSDS-123.0-B-2 Low-Complexity Lossless and Near-Lossless Multispectral and Hyperspectral Image Compression Standard," *Remote Sensing*, vol. 11, no. 11, 2019.

[11] D. Báscones, C. Gonzalez, and D. Mozos, "A Real-Time FPGA Implementation of the CCSDS 123.0-B-2 Standard," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–13, 2022.

[12] Q. Fang, Y. Liu, and L. Zhang, "Design and implementation of a lossless compression system for hyperspectral images," *Traitement du Signal*, vol. 37, no. 5, pp. 745–752, 2020.