High-performance compression of multibeam echosounders water column data

Jordi Portell, David Amblas, Garrett Mitchell, Matias Morales,
Alberto G. Villafranca, Riccardo Iudica and Galderic Lastras

***Abstract*—Over the last few decades multibeam echosounders (MBES) have become the dominant technique to efficiently and accurately map the seafloor. They now allow to collect water column acoustic images along with the bathymetry, which is providing a wealth of new possibilities in oceans exploration. However, water column imagery generates vast amounts of data that poses obvious logistic, economic and technical challenges. Surprisingly, very few studies have addressed this problem by providing efficient lossless or lossy data compression solutions. Currently available options are only lossless, providing low compression ratios at low speeds. In this work we adapt a data compression algorithm, the Fully Adaptive Prediction Error Coder (FAPEC), that was created to offer outstanding performance under the strong requirements of space data transmission. We have added to this entropy coder a specific pre-processing stage tailored to the Kongsberg Maritime water column file formats. Here we test it on data acquired with Kongsberg MBES models EM302, EM710 and EM2040. With this bespoke pre-processing, FAPEC provides good lossless compression ratios at high speeds, whereas lossy ratios reach water column file sizes even smaller than bathymetry raw files still with good image quality. We show the advantages over other lossless compression solutions, both in terms of compression ratios and speed. We illustrate the quality of water column images after lossy FAPEC compression, as well as its resilience to datagram errors and its potential for automatic detection of water column targets. We also show the successful integration in ARM microprocessors (like those used by smartphones and also by autonomous underwater vehicles), which provides a real-time solution for MBES water column data compression.**

***Index Terms*—Multibeam echosounders, water column data, lossless, lossy, data compression.**

FAPEC R&D was supported by ESA Business Incubation Programme and Barcelona Activa, MINECO (Spain) grant ESP2016-80079-C2-1-R (MINECO/FEDER, UE) and ESP2014-55996-C2-1-R (MINECO/FEDER, UE), and MDM-2014-0369 of ICCUB (Unidad de Excelencia 'María de Maeztu'). This work received funding from European Union's Horizon 2020 research and innovation programme under Marie Sklodowska-Curie grant agreement No 658358 (D. Amblas) and from Spanish RTD grant NUREIEVA (CTM2016-75953-C2-1-R). Generalitat de Catalunya supported Grup de Recerca Consolidat (CRG) in Marine Geosciences through grant 2017 SGR 315.

J. Portell is with Dept. Física Quàntica i Astrofísica, Institut de Ciències del Cosmos (ICCUB), Universitat de Barcelona (IEEC-UB), 08028 Barcelona, Spain and with Institut d'Estudis Espacials de Catalunya (IEEC), 08034 Barcelona, Spain e-mail: jportell@fqa.ub.edu

D. Amblas was with Scott Polar Research Institute, University of Cambridge, CB21ER, UK and is with CRG Marine Geosciences, Dept. of Earth and Ocean Dynamics, University of Barcelona, 08028 Barcelona, Spain.

G. Mitchell is with Fugro USA Marine Inc., Houston, Texas, US 77081.

M. Morales is with Kongsberg Maritime, 03570 Villajoyosa, Spain.

A.G. Villafranca is with STAR-Barcelona, 08172 St.Cugat del Vallès, Spain.

R. Iudica is with DAPCOM Data Services, Parc UPC-PMT RDIT, 08860 Castelldefels, Spain.

G. Lastras is with CRG Marine Geosciences, Dept. of Earth and Ocean Dynamics, University of Barcelona, 08028 Barcelona, Spain.

I. INTRODUCTION

IN the last years advances in sonar technology, spatial positioning and computing power have led to significant improvements in mapping, imaging and monitoring of the oceans. Multibeam echosounders (MBES) are now capable of collecting backscatter data for the whole water column, in addition to the traditional measures of bathymetry and seafloor reflectivity. These new data sets open up a new range of applications for multibeam sonars, including mapping of gas seeps, direct imaging of fish and marine mammals, location of mid-water targets, proper determination of sunken structures such as shipwrecks and investigating a wide range of physical oceanographic processes [1].

Despite the potential value of water column reflectivity measurements, the enormous increase in data collection often makes storage requirements prohibitive, forcing many users to opt for not systematically recording water column information. The volume of data generated in multibeam water column surveys can easily be one order of magnitude larger than in conventional bottom detection assessments, especially in shallow water where the higher ping rates lead to data rates of several gigabytes per hour [2]. This complicates the efficient browsing, querying, sharing and transfer of data. It also limits the capabilities of Autonomous Underwater Vehicles (AUVs) equipped with multibeam echosounders and remote assistance during sea works by technical support teams, which often rely on expensive satellite links.

Data compression is a potential solution to this challenging issue. However, few published studies face this matter, and most of them use lossy methods involving a certain degree of signal distortion and water column imagery degradation [2]–[4]. Even fewer sonar-dedicated lossless compression algorithms have been proposed [5], [6], and commonly used lossless techniques such as *Zip* yield only modest compression rates at a very high computing cost. Here we evaluate some data compression tools and their performance on multibeam water column data. We focus on the results obtained using the FAPEC data compressor [7], [8], initially designed to meet the tight requirements of satellite payloads and deep space communications. FAPEC inherits from a Technology Research Programme of the European Space Agency for Gaia, an ambitious space observatory measuring features of more than one billion stars with unprecedented accuracy [9]. The large amount of data from Gaia and its complex data model required a tailored and extremely optimized solution. An intuitive yet rather uncommon approach was proposed, which we named stream partitioning [10] and consists in applying different algorithms to the different data types or structures. As we will show hereafter, the case of multibeam water column data is quite similar, so this same approach should be applicable here. Rather than applying standard data compressors to the output of each sub-stream (as originally proposed in [10]), Gaia required an adequate entropy coder such as [11] or [12]. In the aforementioned Technology Research Programme we developed PEC [13], an entropy coding algorithm resilient to statistical outliers [14] which is at the core of FAPEC. It provides a better balance regarding compression ratio, speed, robustness and resiliency than most standard compressors. In this work we take advantage of these lessons learned in space research to adapt the FAPEC algorithm to marine sciences.

This paper is structured as follows. Next subsections describe the water column data format and the FAPEC data compressor. Sect. II presents the bespoke pre-processing implemented for water column data files. Sect. III describes the test files and software used. Sect. IV shows the lossless and lossy data compression results obtained, as well as error resiliency tests and detection of scene features. Finally, Sect. V presents our conclusions and elaborates on future work.

A. Water column data

MBES raw records are usually logged as binary files using signed or unsigned integer values. Each sonar manufacturer has a specific file format, which in turn can vary depending on the particular sonar model, the survey purpose, its configuration and the external sensors included such as CTD probe, GPS, Compass and Gyro. Each file usually contains time-stamped information about beam geometry, sonar configuration, navigation, attitude, sound speed, bathymetry and water column backscatter measurements. Most of the data volume comes from backscatter raw samples. They can be seen as a two-dimensional array where one dimension corresponds to the several pings (each composed of a given number of beams) and the other dimension corresponds to the samples from each beam. The number of beams per ping depends on each sonar model, the swath aperture angle, the resolution and also on the scene. The number of samples per beam within a ping strongly varies with the beam angle, and throughout the several pings it depends on the scene (mainly on the depth), as illustrated in Fig. 1.

Each sample is typically an integer value, coded as signed 8-bit samples in our case. We remark that the dimensions of the samples array are not uniform throughout the data file, which means that typical image compression algorithms cannot be easily adapted or applied to this kind of data. To illustrate this, Fig. 2 shows a small portion of a water column data file by simply taking the raw samples and arranging them in a square image. As can be seen, the actual width of the image strongly varies throughout the several beams.

In this study we analyze water column data acquired with Kongsberg EM2040 and EM302 multibeam echosounders, kindly provided by Kongsberg Maritime and Fugro. EM2040 is a high resolution shallow water multibeam system operating at sonar frequencies in the 200-400 kHz range with an angular coverage up to 200° and pulse lengths as short as 14 msec. EM302 can operate at depths up to 7000 m, with up to 432 soundings per swath and an operating frequency of 30 kHz. For comparison we also use the results obtained with an EM710 multibeam echosounder dataset acquired by CRG Marine Geosciences of the University of Barcelona [15]. It is a 70-100 kHz multibeam system with a maximum ping rate of 30 Hz, a maximum acquisition depth of 2500 m, up to 400 soundings per swath and a coverage that can reach 140°.

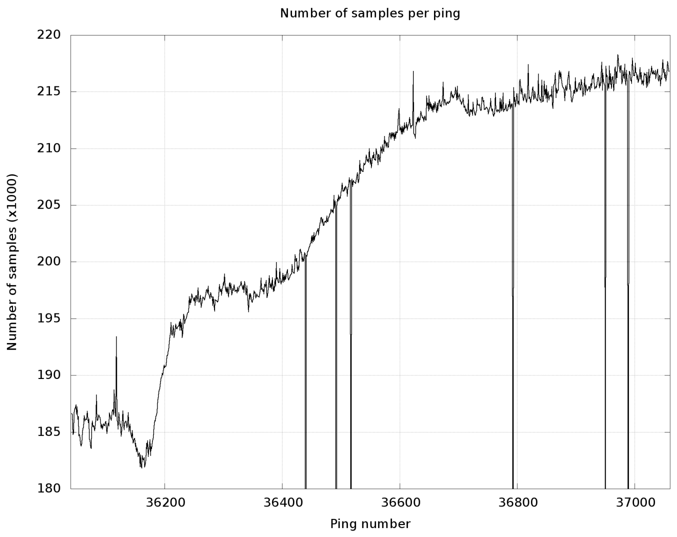

Fig. 1. Number of samples per ping in a test scene from a Kongsberg EM302 echosounder.

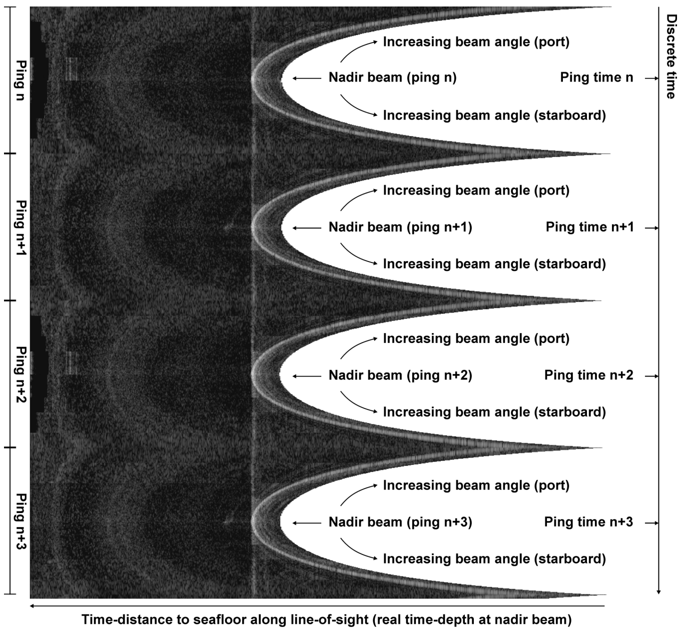
Ping n
Ping n+1
Ping n+2
Ping n+3
Increasing beam angle (port)
Nadir beam (ping n)
Ping time n
Increasing beam angle (starboard)
Increasing beam angle (port)
Nadir beam (ping n+1)
Ping time n+1
Increasing beam angle (starboard)
Increasing beam angle (port)
Nadir beam (ping n+2)
Ping time n+2
Increasing beam angle (starboard)
Increasing beam angle (port)
Nadir beam (ping n+3)
Ping time n+3
Increasing beam angle (starboard)
Discrete time
Time-distance to seafloor along line-of-sight (real time-depth at nadir beam)

Fig. 2. Representation of backscatter raw samples from a Kongsberg EM302 multibeam echosounder. Only four pings are shown, leading to an image which is 1150×1160 pixels. Samples have been converted from signed to unsigned and then scaled to enhance image contrast. We indicate non-existing samples with white pixels artificially added. Depth increases from right to left. Each parabolic shape corresponds to one ping, with the ping time increasing towards the bottom of the image. Within a ping, the vertical dimension of the image indicates the beam angle. In each ping, the focus of the parabola indicates the location of the echosounder.

B. The FAPEC data compressor

FAPEC (Fully Adaptive Prediction Error Coder) is a highly-optimized entropy coding algorithm which offers outstanding resiliency in front of outliers in the data [14]. That is, FAPEC reaches good compression ratios even on data severely contaminated by noise and values outside the typical statistics. Its compression efficiency is typically above 90% of the Shannon limit [16], that is, the maximum theoretical compression ratio achievable by an entropy coder. Owing to its embedded run-length encoding and low-entropy modes, it can even exceed the performance of an optimum Huffman coder [11].

Since its initial conception [7], FAPEC has actually evolved into a highly configurable, efficient and versatile data compression system [8] applicable to airspace and ground systems, as well as maritime systems. FAPEC follows a typical two-stage approach. The first stage or decorrelator reduces the original entropy of the data by applying a reversible (lossless) or partially-reversible (lossy) algorithm. It can be as simple as a delta stage or differentiator, outputting differences between consecutive samples. Linear filters can also be used, as well as interleaving for samples following some given pattern. More complex stages can also be used, such as pattern recognition (dictionary compression), multiband prediction or image compression algorithms. Some of these stages support lossy compression. Finally, and most important for our case, tailored pre-processing stages can be implemented and easily integrated into the FAPEC software framework.

The output of the first stage is a sequence of signed integers. We refer to these as prediction errors, although they can also be special codes generated by dictionary-based algorithms, for example. In any case, the aim of the first stage is to generate a statistical distribution which should approach, as much as possible, to a Laplacian distribution [17]. FAPEC also performs well with Gaussian and similar distributions. In general, the output of the first stage should be signed values with much higher probability for values around zero than for high values (either positive or negative). The second stage, which is the entropy coder in itself, generates short binary codes for more frequent values, and slightly longer codes for less frequent values. It includes mechanisms for the efficient compression of sequences with repeated values (also known as run-length encoding, [18]), thus going beyond the simple entropy coding.

FAPEC, implemented in highly optimized ANSI C, is available as an executable program for Linux, Windows or Mac OS, and as a dynamic library with a simple Application Process Interface (API) for better integration with third-party data handling systems. It supports both Little Endian and Big Endian platforms, as well as ARM processors which are an excellent option for Autonomous Underwater Vehicles (AUV) and Autonomous Surface Vehicles (ASV) [19]. A hardware prototype in VHDL language is also available for programmable electronic chips (FPGAs) [8], [20]. FAPEC can natively handle sample sizes of 8 to 24 bits, and arbitrarily large samples by means of interleaving. Its compression performance is excellent especially on samples of 16, 32 or 64 bits. FAPEC also enforces data integrity, minimizing data loss in case of file or data transfer corruption. This is implemented with a *chunking* mechanism, which splits the input file in chunks of a given size configurable by the user (typically around 1 MB). FAPEC then codes each chunk independently, so the loss of a complete data chunk does not affect the rest of the file. This chunking mechanism also provides a better performance on disk and allows a multithreaded execution in multicore computers. Finally, the FAPEC software framework allows on-the-fly data encryption.

II. FAPEC TAILORING FOR MBES WATER COLUMN DATA

A. Overall approach

We have focused our work on Kongsberg MBES water column data files, but it could be extended to other vendors. The format of these binary files is quite complex, with some headers and tags, sonar information, measurement attributes and obviously the raw samples. Each of these elements is coded using different data types, such as signed 8-bit samples, unsigned 32-bit time tags, unsigned 16-bit physical information (such as the sound speed), or signed 16-bit information on the beam pointing angle. Furthermore, these files are arranged in *datagrams*, each with at most 64 KB. There is no direct correspondence between datagrams and pings, that is, samples from a ping are typically split between different datagrams, or datagrams may contain data from different pings. All this complicates the design and implementation of any optimum tailored compression because it first requires an adequate reconstruction of the scene (or image) by rearranging the raw samples. It can be done as a typical two-dimensional image array (beams vs. samples), as illustrated in Fig. 2, or even as a "data cube" by using the ping number as the third dimension.

To further complicate things, some water column data files can contain different datagram types besides the water column ones. For example, attitude, depth or position datagrams. Each of these datagram types have a completely different data format. This heterogeneous mixture of data types, formats and data statistics poses a big difficulty to standard data compressors such as *Zip*. If directly applied to these files they will rarely find the intrinsic data redundancies, meaning a poor data compression performance. It is worth mentioning that FAPEC, without the adequate pre-processing stage, would also face this problem if directly applied to such files.

This is, curiously, a similar problem to that posed by the ambitious Gaia space observatory of the European Space Agency [9]. In that case we also have a complex data format, mixing 16-bit unsigned raw samples (the stellar image pixels) with many flags and attributes of different types. Thus, a similar approach to that initially envisaged in [10] for Gaia could be followed here. In this work we have further elaborated on the prototype presented in [15], slightly improving its lossless compression performance and, especially, implementing the lossy compression option. Lossless compression has been improved mainly by transposing the raw samples differential coding, that is, following the vertical dimension in Fig. 2 (as described hereafter), whereas in [15] we were following the horizontal dimension. Fig. 3 illustrates the overall operation of FAPEC on Kongsberg MBES water column data files.

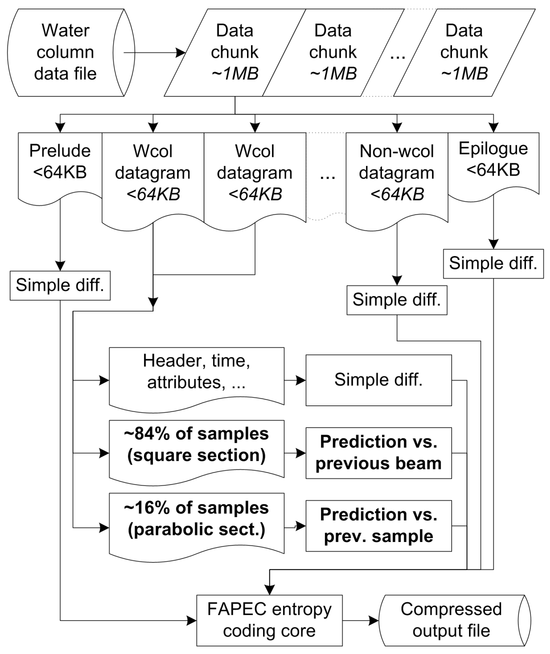

Fig. 3. Overview of the FAPEC tailoring for MBES water column data files.

The very first operation (given by the general FAPEC framework) is the split of the input file into small data chunks. Their default size is 1 MB, but the user can configure it from just 4 KB to 384 MB. Each of these chunks is handled independently from the others, which isolates eventual decompression errors due to corruption of the compressed file. In this work we illustrate this error resiliency capability with a specific test. Also, the multithread operation of FAPEC better scales in many-core computers when using small chunks (compared to the input file size). In general, chunk sizes of 256 KB (for better error resiliency) to 16 MB (for slightly better ratios) are recommended for the water column case.

This FAPEC tailoring stage is robust in front of unexpected data, such as datagrams not containing water column data, and even partial or corrupted datagrams. In these cases, compression performance will not be optimal, but no failure or data loss will occur. These unexpected data elements are named *prelude* and *epilogue* in Fig. 3. It is worth mentioning that we process each datagram separately, aiming at a simplistic approach for now. In future improvements we intend to reconstruct each complete ping sequence, allowing to use the information from previous pings to, hopefully, obtain better compression ratios.

Water column datagrams are detected by checking some flags in the input data stream, such as the so-called *start identifier*, the *EM model number*, and mainly the *datagram type*. When a valid datagram is found, we first process its flags and attributes in a simplistic manner. Namely, we have identified some "typical" values (or range of values) for each of them, and we simply code the actual values differentially with respect to such references.

This tailored pre-processing stage outputs the measurement attributes separated from the backscatter samples. Prelude and epilogue sections, if present, are also output separately from the samples. In this way we enforce some uniformity in the statistics of the values to be entropy-coded by the FAPEC core, leading to slightly better compression ratios.

B. Handling of backscatter samples

The most important part of the tailored pre-processing stage is the raw samples processing. We have implemented a relatively simple approach to minimize computing overhead, namely, a simple data predictor based on neighbouring samples. First, we identify the size of the "square" portion of the image — that is, the minimum number of samples per beam found in the current datagram. In Fig. 2 it corresponds to the samples at the left of the parabolic shapes, so that there is always an existing sample "above" (in the meaning of the figure image). About 84% of the total water column samples fall in this region, at least in the files tested. In the mentioned figure, we identify each sample as $S_{i,j}$, where i is the horizontal coordinate (samples within a beam) and j the vertical one (going through beams and pings). The prediction of each of these samples, $P_{i,j}$, is estimated from the same sample of the previous beam (that is, the pixel "above", in the mentioned figure). This is done in this way because sample values are typically more correlated in this direction than through a beam, as can be seen in Fig. 2. Better results may be achieved by identifying the adequate shift in i to follow the parabolic shape, thus leading to a higher correlation. However, preliminary tests in this direction did not reveal any significant improvement. The resulting difference or prediction error, $E_{i,j}$, is then coded with the FAPEC entropy coding core. We actually refine the prediction with a small compensation, $C_{i,j}$, based on the last four prediction errors obtained. This can be seen as a retroactive filter. Specifically:

$$E_{i,j} = S_{i,j} - P_{i,j} \tag{1}$$

$$P_{i,j} = S_{i,j-1} + C_{i,j} \tag{2}$$

$$C_{i,j} = \frac{1}{16}(8E_{i-1,j} + 4E_{i-2,j} + 2E_{i-3,j} + E_{i-4,j}) \tag{3}$$

Reference pixels and correction coefficients are initialised to zero and later propagated for each sample, allowing to correctly handle the first line and column. The parabolic region, which represents about 16% of the water column backscatter samples, is handled in a slightly simpler manner. We follow a similar approach (sample prediction with compensation based on recent prediction errors), but instead of using the previous beam we use the previous sample of the same beam. That is:

$$P_{i,j} = S_{i-1,j} + C_{i,j} \tag{4}$$

$$C_{i,j} = \frac{1}{16}(7E_{i-1,j} + 6E_{i-2,j} + 4E_{i-3,j} + 3_{i-4,j}) \tag{5}$$

The values of the compensation coefficients have been determined empirically. This could be improved by automatically determining the optimum values for each datagram or by regularly updating the values after a few datagrams, although the effect on the compression speed should be carefully evaluated.

C. Lossy compression approach

Lossy data compression can be selected by the user, allowing several levels of quality degradation — obviously affecting backscatter samples only. Higher compression ratios can be achieved with a higher data degradation, as typically seen in data compression systems for images, sound or video. In this case of Kongsberg water column data with 8-bit samples we allow 7 quality levels, from level 1 (better quality and lower compression ratio) up to level 7 (worst quality and best ratio), although we do not recommend to exceed level 5. Level 0 means lossless compression. Losses are introduced by quantizing each of the original raw samples, that is, dividing the sample value by a given factor which depends on the quality level given by the user, ranging from 2^1 (level 1) to 2^7 (level 7). In practice, it means a reduction in the number of gray shades of the water column image, from the original 256 shades (8-bit samples) to 128 (level 1), 64 (level 2) and so on, up to just a black and white image (level 7). The maximum recommended losses (level 5) leads to just 8 gray shades, including black and white.

It is worth mentioning that this image contrast degradation is made by rounding each pixel to the nearest available shade value (depending on the quality loss level selected). In case we have two values at the same distance, we round towards minus infinity. Some simplistic lossy compression algorithms just truncate the sample values, thus ceiling them to a shade value closer to black. Our approach leads to a negligible bias in the quantization noise, which in turn leads to a better image reconstruction both quantitatively and qualitatively. Fig. 4 illustrates this by showing a histogram of the quantization noise in lossy data compression tests made with FAPEC at different quality levels. As can be seen, quantization noise is mostly flat except for a peak at zero (meaning no quantization noise, that is, no data loss). The important result here is that the probability density function of quantization noise is quite symmetrical, contrary to that obtained from lossy compression approaches that simply truncate the samples. The latter leads to always positive quantization noises and thus strongly biased results. The slightly larger negative range seen in the figure for our quantization errors is simply caused by the inherent operation of signed binary coding.

This lossy compression approach does not introduce any spatial degradation in the image, contrary to other approaches based on wavelet transformation, for example. In those cases, the full image contrast is typically kept but losses lead to blurred images, which in practice means a worse image resolution. In our solution we have opted for the degradation of image contrast rather than image resolution, so that users can still identify important features in water column data such as sunken structures, fish shoals, marine mammals or gas seeps.

D. Additional remarks

A very interesting feature of this overall compression approach is the capability of automatically detecting significant

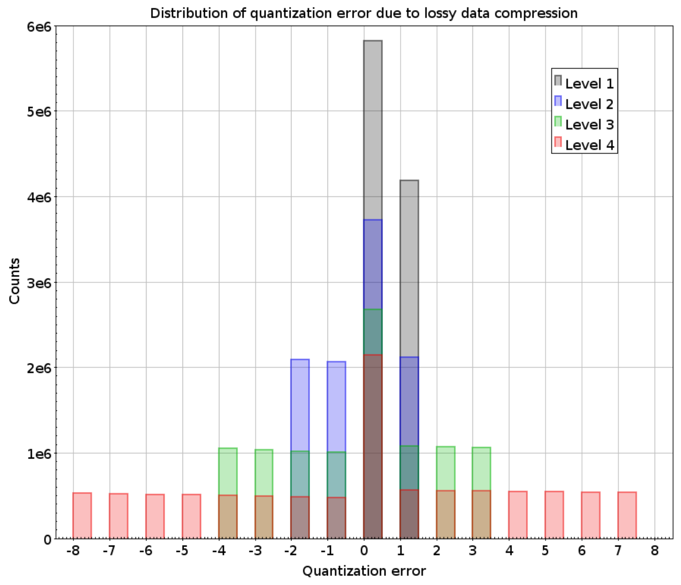

Fig. 4. Distribution of quantization errors (or noise) caused by lossy FAPEC compression on MBES water column data. Only the first four levels of quality degradation are shown, corresponding to 128, 64, 32 and 16 shades of gray. Quantization error is indicated as digital counts (original sample value minus decompressed value).

changes in the water column characteristics. This is achieved owing to our chunk-based operation, which can provide individual or accumulative compression ratios for each of the data chunks. Sudden changes in ratios may correspond to peculiarities in the data, whereas smooth changes may mean a large-scale change in the scene such as the sea depth.

Finally, it is worth highlighting that FAPEC, including this tailored pre-processing stage, is fully functional and reliable. Emphasis has been put on robustness, adequately handling any kind of datagram or rare data structure contained in the files by simply compressing them differentially. This may not be optimal, but it avoids any crash, failure or data loss. A completely lossless operation has been assessed on EM302, EM710 and EM2040 files. When using the lossy option, we have assessed that losses are only applied to water column samples and not to any attribute or header.

III. Test setup

The dataset used in the lossless and lossy compression tests includes three files acquired during a Kongsberg EM2040 multibeam survey in 2015 inside the harbor of the city of Barcelona, and three more files with a Kongsberg EM302 sounder in the same year in the Gulf of Mexico at depths around 1000 m. The EM2040 test shows numerous harbor structural elements in the bathymetry and water column, as well as shoaling fish. EM302 data shows actively emitting hydrocarbon fluids into the overlying water column. This dataset by Fugro is named GC600 and is being used as a seep calibration site to optimize acquisition and processing settings for seep detection [21]. GC600 is situated in a topographically, geologically and biologically complex seafloor with intensive natural hydrocarbon seepage among a salt-controlled area that creates a network of subsurface faulting and fissures. Finally, two files acquired with a Kongsberg EM710 sounder include data from a survey in the outer continental shelf of the southeastern Iberian Peninsula (see further details about this dataset in [15]).

Data rate generated during MBES water column acquisition strongly depends on the ping rate and thus on the sea depth. Sounder features such as the number of beams or the backscatter samples resolution obviously have an effect as well. EM302 scenes barely reached 0.1 MB/s, whereas the largest EM2040 file was acquired at an average rate of 4.3 MB/s. EM710 files correspond to an average rate around 1 MB/s. These raw data throughputs must be taken into account if we consider the case of water column data compression in real time, that is, while being acquired during a campaign.

For the data compression tests we have selected *gzip*, *bzip2*, *7-zip* and *Zstandard* as references, using their default options. *Gzip* is a well-known compressor widely used in Linux and Mac OS, equivalent to the also widely known *Zip* compressor in Windows. *Bzip2* is an also well-known alternative to *gzip*, often slower but typically leading to better compression ratios. *7-zip* (or simply *7z*) is a relatively new solution which often achieves excellent ratios but at the cost of an extremely slow operation. Finally, *Zstandard* is a new compressor created at *Facebook* which yields ratios similar (sometimes better, depending on the options selected) than *gzip* but with an excellent compression (and especially decompression) speed. There is typically a compromise between good compression ratios and speed. We may also consider other application-specific compressors such as the CCSDS 121.0 recommendation for lossless data compression in space [22], also known as adaptive Rice codes. However, it would require an adequate pre-processing stage (like the one presented here) to achieve good ratios. Otherwise we cannot perform a direct comparison, and thus, we have excluded it from our tests. Other studies [7], [14] provide direct comparisons between the FAPEC entropy coding core and adaptive Rice codes, where the latter appears to be significantly affected by statistical outliers in the data.

Note that all of these compressors are lossless, so we only compare lossless FAPEC against these. There is no lossy data compressor (such as JPEG) directly applicable to Kongsberg MBES water column data files, so lossy FAPEC operation has been evaluated by means of ratios and quality.

We have forced a single-thread operation in FAPEC (and also in *7z*) for a fair comparison with the other compressors. With multithreading, the speed scales quite linearly with the number of threads used. FAPEC 18.0 Beta has been used, which includes the tailored pre-processing option presented here. The latest version of FAPEC can be downloaded from https://www.dapcom.es/get-fapec. All tests were run on a laptop with Intel® Core™ i7 2640M 2.8 GHz processor running 64-bit Gentoo Linux. Only the User time (that is, the CPU time) has been taken into account to ignore the effect of disk I/O. All tests were done directly on the Kongsberg `.wcd` files (water column datagram).

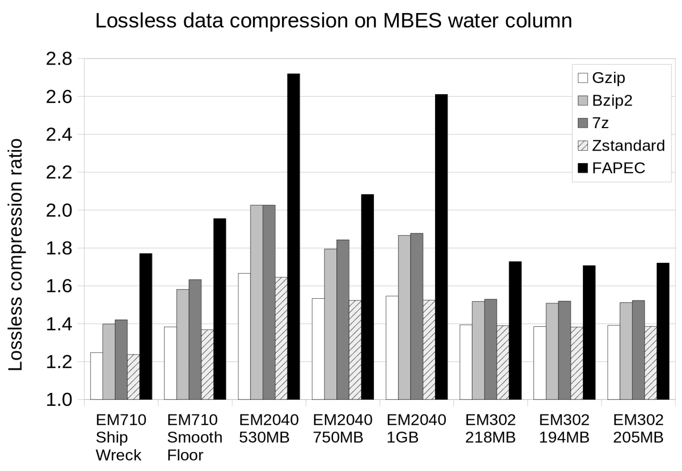

Fig. 5. Lossless compression ratios (original file size divided by the compressed file size) obtained for the solutions and scenes tested.

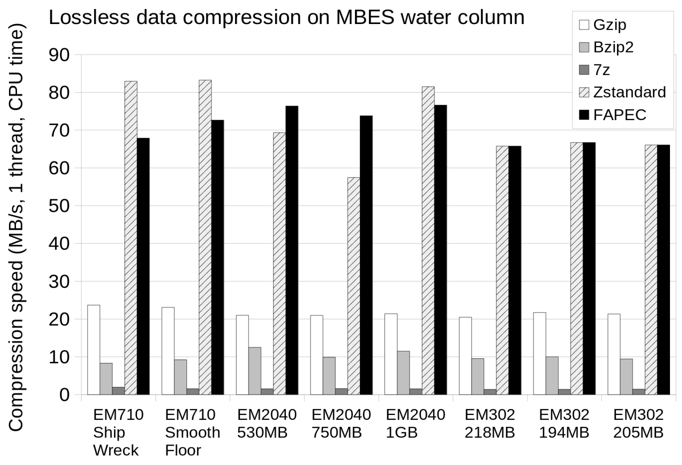

Fig. 6. Lossless compression speeds, indicated as the volume of raw (original) data handled by the compressor in one second.

IV. Test results

A. Lossless compression

Fig. 5 shows the lossless compression ratios obtained in these tests, that is, the original file size divided by the compressed file size. As can be seen, FAPEC achieves, without any doubt, the best compression ratios for all sounder models and scenes. When compared to *gzip*, FAPEC yields ratios at least 23% better, reaching 68% in the high-rate EM2040 case. *Zstandard* results are very similar to those of *gzip*. It is worth mentioning that higher ping rates seem to lead to better compression ratios, especially in FAPEC. This is an otherwise expected result, because higher rates (or a higher sampling resolution) means a higher correlation between neighbour samples, which is exactly what FAPEC exploits. Therefore, we can expect better compression ratios in the most demanding cases, which is obviously an excellent result.

Ratios obtained by *bzip2* and *7z* are slightly better than those of *gzip* or *Zstandard* but still far from those of FAPEC. One could consider to simply use these quite standard compressors, as they are available in most computing platforms (especially *bzip2*). However, we should take into account the test results

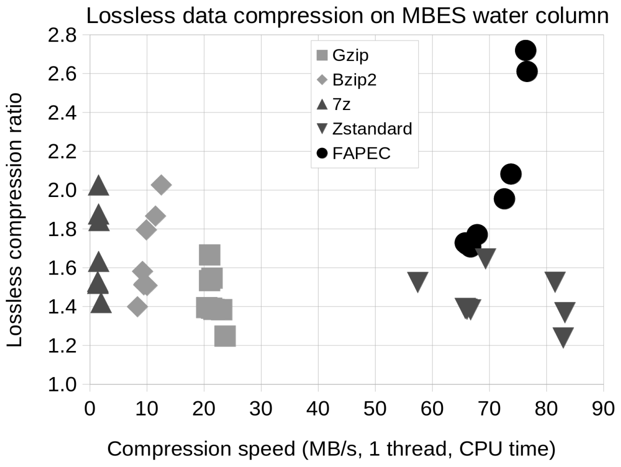

Fig. 7. Comparison of lossless ratios versus compression speed. Each point corresponds to one file and one compressor type.

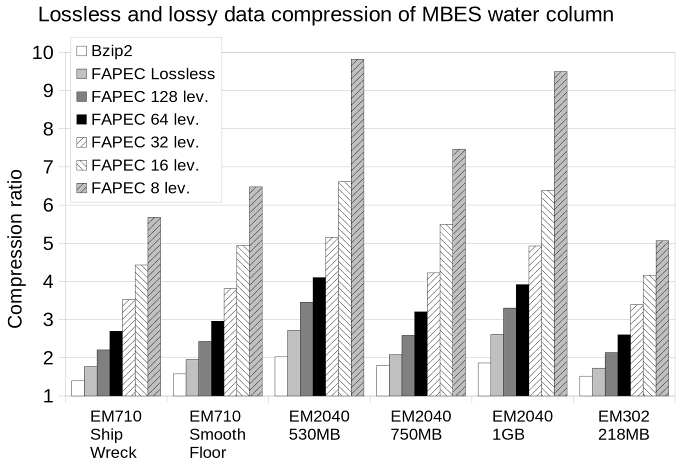

Fig. 8. Lossy compression ratios obtained by FAPEC using different quality levels, from level 1 (leading to 128 shades of gray) to 5 (8 shades of gray). Lossless ratios obtained by the nearly best "standard" compressor (*bzip2*) and FAPEC are included as reference.

on compression speeds as well, which are shown in Fig. 6. Again, FAPEC results are excellent, although in this case *Zstandard* offers very similar results. When compared to *gzip*, FAPEC compresses about three times faster. The also promising compressors previously mentioned, *bzip2* and *7z*, are even slower than *gzip* — typically two times slower in the case of *bzip2* and ten times slower with *7z*. Therefore, in this case of MBES water column files, FAPEC is about 30 to 50 times faster than *7z*, which can barely approach its ratios in the best of the cases. Regarding *Zstandard*, while it does compare to FAPEC regarding compression speed, its ratios are typically the worst ones. Therefore, FAPEC is clearly the best lossless compression option both in terms of ratio and speed, as illustrated in Fig. 7.

In absolute terms, FAPEC compression speed is far enough to allow real-time compression while acquiring water column data. As previously mentioned, the most demanding case tested does not even reach 5 MB/s, whereas FAPEC offers at least 60 MB/s in a laptop. Even *gzip* is fast enough to handle this case. However, we should also take into account low-performance ARM-based computing platforms, such as those used by some AUV and ASV. We have carried out tests on an ARMv7 Linux platform at 800 MHz. FAPEC, again in single-thread mode, compresses water column data at more than 8 MB/s. This is almost twice the most demanding case, which means that FAPEC can compress water column data in real-time using about 50% of one ARMv7 800 MHz core.

Multithread operation has also been tested. With a given specific file (EM302) and on the mentioned 4-core laptop, the following results are obtained. Single-thread operation allows compressing 65 MB/s (in terms of raw input data handled). With two compression threads (plus separate input/output threads), FAPEC compresses at a rate of 128 MB/s. With 3 threads we reach 183 MB/s, and finally with 4 threads we reach 226 MB/s. These results mean that FAPEC could also be used for massive data compression in large archives and backups, for example.

Finally, the memory requirements of the several compressors have also been tested, which is a relevant figure especially for ARM-based systems. *Gzip*, *bzip2* and *Zstandard* have very modest requirements (4 MB to 11 MB). The worst case is *7z*, which uses 27 MB in its *fast* configuration but its *ultra* option raises this requirement up to 700 MB. FAPEC is also very lightweight here, barely requiring 8 MB of RAM. In multithreaded operation FAPEC increases this memory requirement almost linearly with the number of threads.

B. Lossy compression

Fig. 8 presents the lossy compression ratios obtained by FAPEC using different quality levels. Here we only include one EM302 scene, because the three scenes of this sounder provide very similar results. As otherwise expected, the ratio increases as we reduce the number of gray shades (that is to say, as we increase the level of losses). Again, better ratios are obtained for those scenes where the ping rate is higher. With an intermediate level of losses (such as 32 shades of gray, which still provides excellent image quality) we can easily obtain ratios of 3.5, even reaching 5.0 at high ping rates. To better illustrate this in terms of file size, Fig. 9 provides the original, lossless and lossy sizes obtained in these tests. Note that we include the corresponding bathymetry file size (`.all` file, without any compression) as reference. As can be seen, in the EM2040 and EM302 cases we can even reach water column files smaller than their corresponding bathymetry files. When reducing the number of gray shades to just 8 (including black and white), still giving reasonably good image quality, we even approach a ratio of 10 at high ping rates.

When evaluating a lossy compression algorithm it is recommended to determine the peak signal-to-noise ratio (PSNR), which is a quantitative evaluation of the data quality degradation. It is based on the mean squared error (MSE), determined from the differences between the original samples and the lossy (decompressed) ones. The PSNR is evaluated in decibels (dB), with typical values between 30 and 50 dB (where higher is better) for 8-bit samples. Fig. 10 illustrates the relationship between different PSNR results and the compression ratio

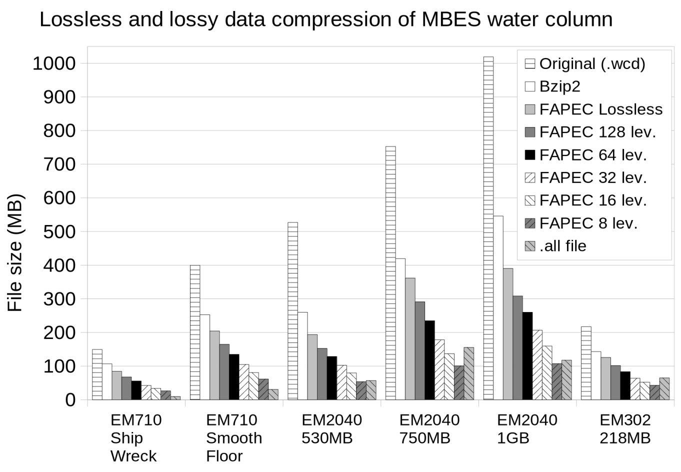

Fig. 9. File sizes obtained after lossy FAPEC compression at different quality levels. Raw (original `.wcd` file), lossless-compressed and bathymetry (`.all`) file sizes are also included as reference.

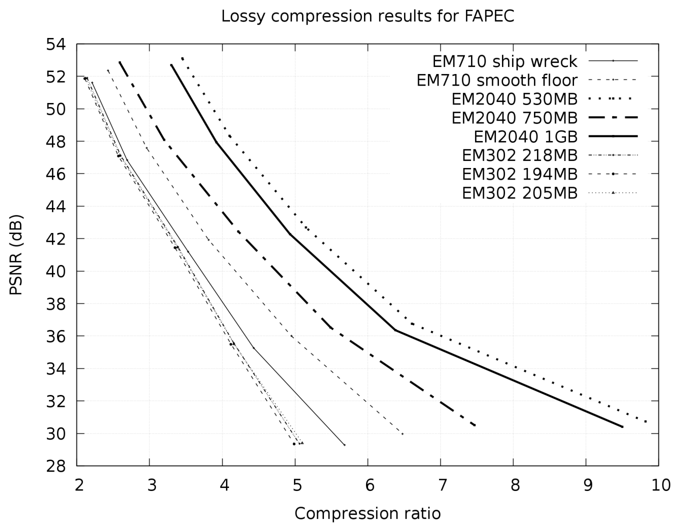

Fig. 10. Quality of the reconstructed (decompressed) water column images at different compression ratios with FAPEC.

obtained. The files with a higher ping rate (mainly EM2040) show again the best results, whereas EM302 files (for a scene with depths around 1000 m) show the worst results. It is worth mentioning that some lossy compression algorithms allow to indicate a target *rate* (or ratio), such as MP3 or JPEG-2000 [23], meaning that they allow quite smooth variations in both PSNR and compression ratio. In FAPEC, on the other hand, we can only indicate a target quality (at least for now), whereas the resulting ratio cannot be known or fixed beforehand. This, together with the lossy compression approach chosen (samples quantization instead of spatial transform), means that the PSNR values obtained by FAPEC for any sounder and scene are quite grouped around some specific values. For example, quality level 2 (meaning 64 shades of gray) leads to a PSNR between 46.8 and 48.3 dB for all files, and quality level 4 (16 shades of gray) leads to PSNR values between 35.3 and 36.7 dB.

Besides these quantitative results, Figs. 11 and 12 provide a qualitative evaluation of this lossy compression approach.

TABLE I
DATA VOLUME GENERATED DURING ONE HOUR OF CONTINUOUS WATER COLUMN ACQUISITION BY DIFFERENT MBES MODELS AND SCENES.

Ping rate	Raw	*Gzip*	Lossless FAPEC	Lossy-3 FAPEC
Low (EM302)	306 MB	220 MB	177 MB	90 MB
Medium (EM710)	4.2 GB	3.0 GB	2.1 GB	1.1 GB
High (EM2040)	14.9 GB	9.6 GB	5.7 GB	3.0 GB

Fig. 11, corresponding to an EM2040 sounder with high ping rate, clearly reveals a fish shoal. The several lossy levels of FAPEC (1 to 5) illustrate the effect of the sample value quantization, leading to an image with progressively less contrast for higher levels of quality loss. However, even in the worst case tested here (level 5, leading to just 8 shades of gray) we can still see very clearly the fish shoal, although the overall quality degradation is evident. Moderate quality losses such as 3 (with 32 shades) lead to barely noticeable degradation while boosting the ratio from 2.08 (FAPEC lossless) to 4.22, leading to a file size quite close to that of the raw bathymetry.

Similarly, Fig. 12, corresponding to an EM302 sounder with low ping rate, also reveals a peculiar feature (gas seeps in this case) in the water column. Again, even the worst quality case tested here allows detecting the structure in the image. The gas seeps actually seem to be more evident in such low-quality case. An intermediate level of losses such as 3 also leads to barely noticeable quality degradation, with the compression ratio boosting from 1.72 (FAPEC lossless) to 3.38. Quantization errors corresponding to this scene are shown in Fig. 4.

As a final illustration of the FAPEC potential for real-time offshore compression of water column data, Table I shows the data volume generated per hour depending on the approach chosen. We consider raw water column output (without compression), *gzip* compression, FAPEC lossless compression and FAPEC level 3 lossy compression (with 64 shades of gray).

C. Decompression performance

FAPEC excels at compression speed, but decompression speed must also be taken into account. This is especially important to allow for a seamless integration in massive data handling systems or in real-time visualization and browsing tools. For example, one could consider integrating the FAPEC decompressor in MBES water column visualization or data analysis tools, allowing to operate directly on compressed files. It would greatly reduce disk or network throughput requirements, as well as memory requirements to handle these huge files. In any case, decompression is often done on reasonably powerful computers, whereas compression may be done in ARM-based platforms or low-performance computing systems.

We have carried out some tests on Kongsberg EM2040 and EM302 files for this purpose. FAPEC decompression speed is relatively modest when compared to solutions such

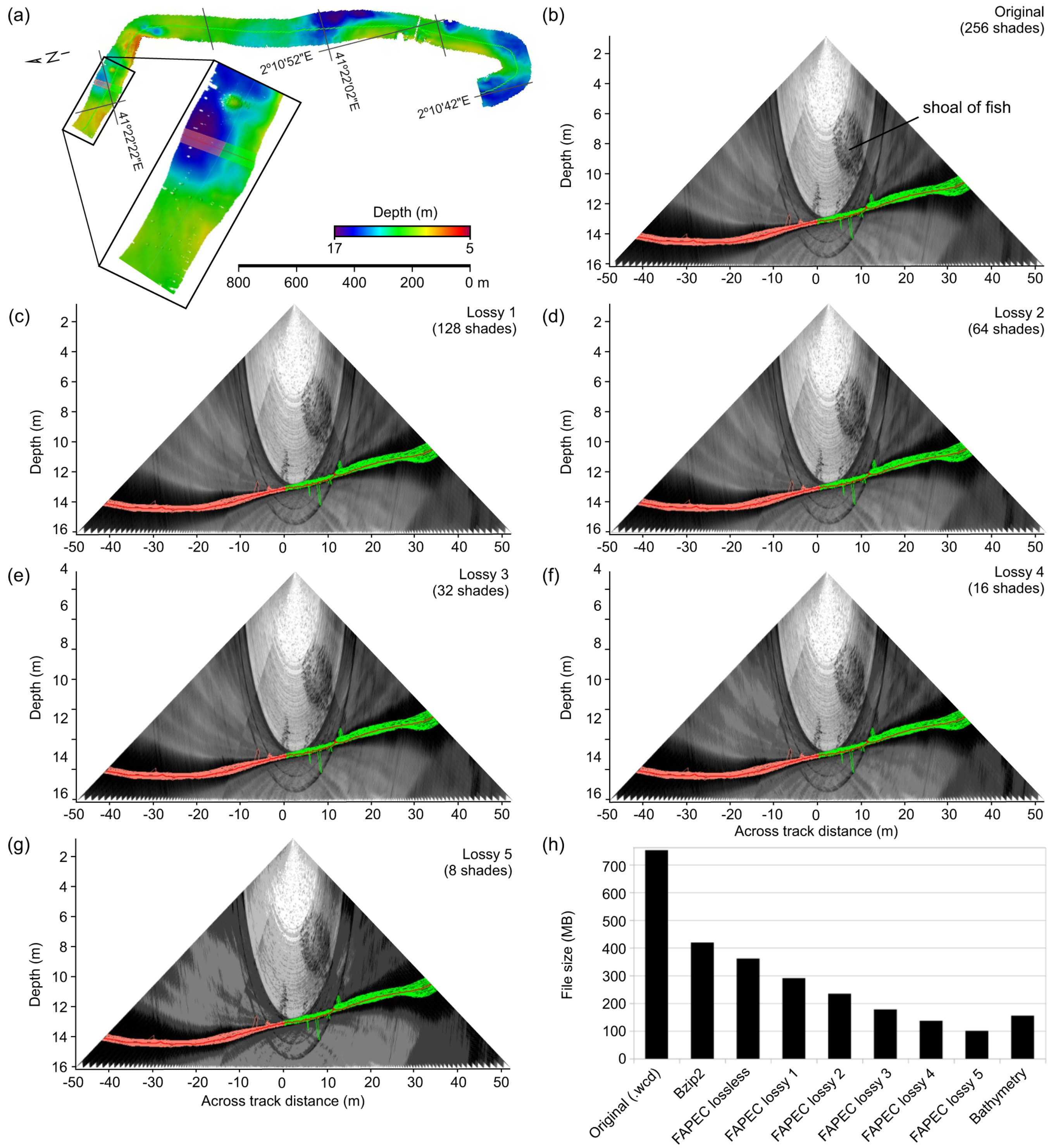

Fig. 11. Kongsberg EM2040 survey in the Barcelona city harbor. a) Bathymetry; b-g) Mid-water backscatter showing a fish shoal at different levels of water column backscatter lossy compression, from (b) the original file (256 shades of gray) to (g) the lowest-quality lossy case (8 shades of gray). h) Original and compressed file sizes, including the results for *bzip2* and the bathymetry file (.all) for comparison. Data courtesy of Kongsberg Maritime.

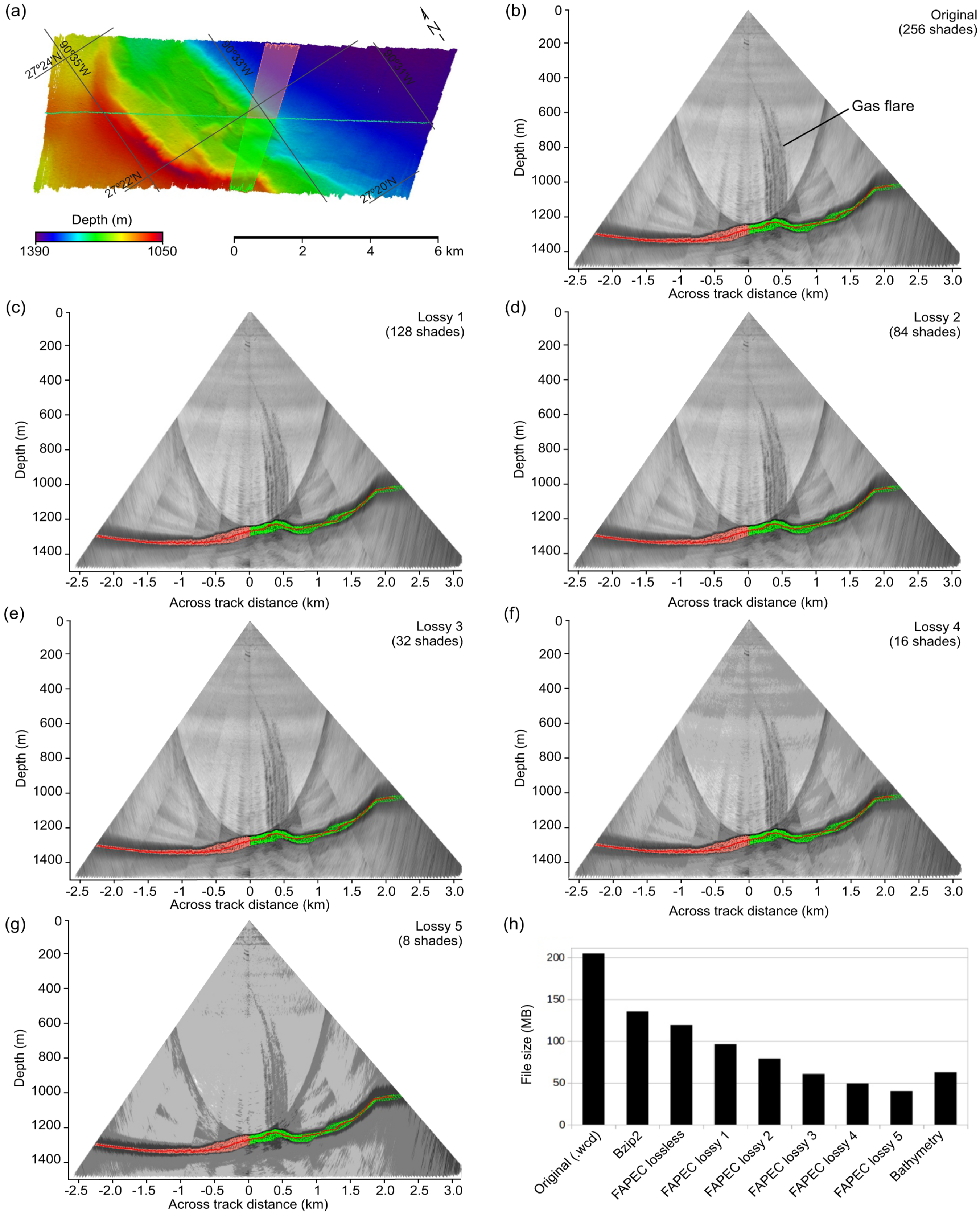

Fig. 12. Kongsberg EM302 survey over Green Canyon in the Gulf of Mexico. a) Bathymetry; b-g) Mid-water backscatter showing gas seeps at different levels of water column backscatter lossy compression, from (b) the original file (256 levels of gray) to (g) the lowest-quality lossy case (8 levels of gray). h) Original and after compression file sizes (in MB), including the results for *bzip2* compression and the bathymetry file (.all) for comparison. Data courtesy of Fugro.

as *Zstandard*, which can exceed a decompression throughput of 350 MB/s in a single thread, or the ubiquitous *gzip*, which can exceed 100 MB/s. On the other hand, *bzip2* and *7z* just reach 18 MB/s and 30 MB/s respectively. In our case, FAPEC decompresses these files at about 60–70 MB/s in a single thread. Although slower than *gzip* or (especially) *Zstandard*, the remarkably better compression ratio compensates this, and in any case, this is a fast enough operation to allow for a real-time operation. When using multithread decompression, FAPEC can easily exceed an output throughput of 200 MB/s in a typical 4-core computer.

D. Variations in ratios related to the scene

As previously mentioned in Sect. II-D, the chunk-based operation of FAPEC has a very interesting and potentially useful side effect. Individual compression ratios obtained for each of the chunks can be monitored in order to quickly detect evident features in the scene being compressed. Fig. 13 illustrates this for the EM302 scene with gas seeps. We should note that the aspect of the bottom panel, with the FAPEC ratios per chunk, can significantly change depending on the chunk size and also on the averaging or smoothing done for the plotting. In any case, it is clear that large-scale features of the scene have a clear large-scale effect on the ratios. Specifically, the central region with a rough sea floor (where the gas seeps are found) lead to slightly lower ratios, whereas the rightmost region with a smoother sea floor (the continental platform) leads to higher ratios. With an adequate FAPEC chunk alignment with the number of samples per ping, together with an adequate smoothing or averaging of the chunk ratios, automatic detection of water column features should be possible. The case of gas seeps may not be trivial, as they use to appear in rough sea floors which lead to rapid variations in the ratios. However, fish shoals or sunken structures in continental platforms (which should lead to more uniform ratios) may have a chance. We could also consider an adaptive lossy compression, letting FAPEC use the lowest-quality level for smooth regions and automatically increasing the quality for more irregular regions.

E. Error resiliency

Yet another benefit of the chunk-based operation of FAPEC is the minimisation of data loss in case of file corruption. In most data compressors, a corrupted compressed file often leads to a complete data loss. Considering the typically large water column file sizes (and the costs associated to their acquisition), this is obviously a risk that should be mitigated. When a compressed FAPEC file gets corrupted, the decompressor detects this but it still tries to recover the complete data file, typically leading to data loss for just the affected chunk. For example, if a 300 MB compressed file gets a few corrupted bytes in the middle, only the chunk containing those bytes will be lost, whereas the rest of the file will typically be completely recovered. With the typical chunk sizes being 1 to 8 MB, it means that only a tiny fraction of the file may be lost. Also, we should mention that FAPEC can compress several files into a single compressed archive (similar to a Linux *tar.gz* or a Zip

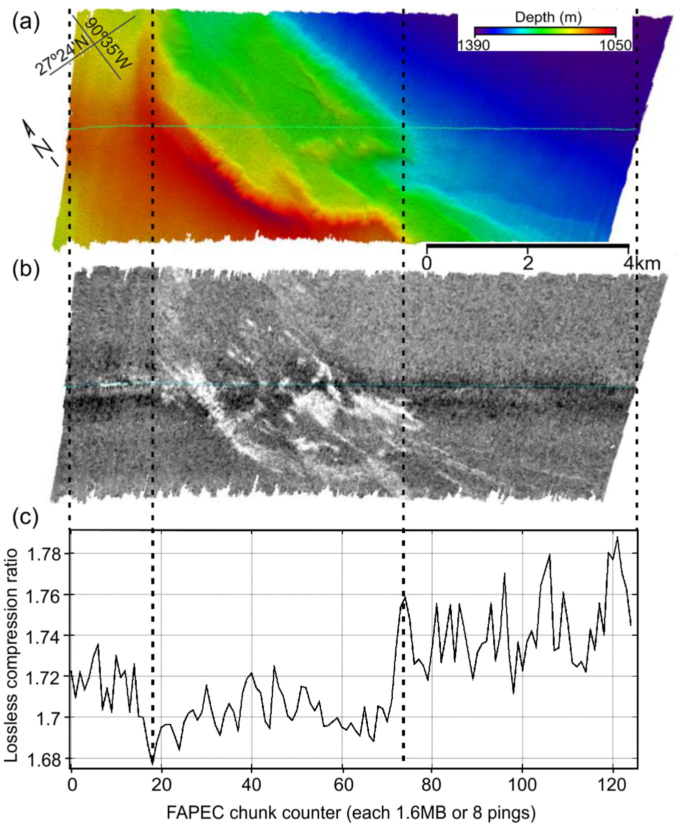

Fig. 13. Effect of the water column features in the FAPEC compression ratio per chunk. The scene corresponds to the EM302 survey with gas seeps. Top panel shows the bathymetry (as in Fig. 12 a) and central panel the backscatter in the same area. Brighter backscatter samples correspond to the gas seeps, which are found in a non uniform sea floor. Bottom panel shows the FAPEC compression ratios per chunk, where we can see a decrease in the irregular region (around chunks 19 to 70).

file, for example). If such an archive gets corrupted, only the file affected by such corruption will be (partially) damaged, whereas the rest of the files contained in the archive will typically be recovered without errors. FAPEC includes some redundancy in its critical headers and footers to maximize the chances of a perfect recovery in case data corruption affects the very beginning or end of the archive.

We have done a test to illustrate this error resilience. Taking the EM302 compressed file with the gas seeps (see Fig. 12), we have manually corrupted it using the `hexedit` tool of Linux. Furthermore, we have intentionally corrupted the portion of the file corresponding to the region with the gas seeps, which seem to be contained at a file offset between 36% and 53% of the total file size. We have gone to an offset of 44.0% and manually set 180 contiguous bytes to random values. Also, at an offset of 44.6%, we have manually reset 540 contiguous bytes (setting them to zero). Afterwards, we have run the FAPEC decompressor which, as expected, has detected checksum errors in two of the chunks. Fig. 14 shows the comparison between the original gas seeps scene and the corrupted one. As can be seen, damage in this specific region is evident, although the overall scene (including some of the gas seeps) can still be correctly seen. It is worth noting that we have also done this test corrupting only another portion of

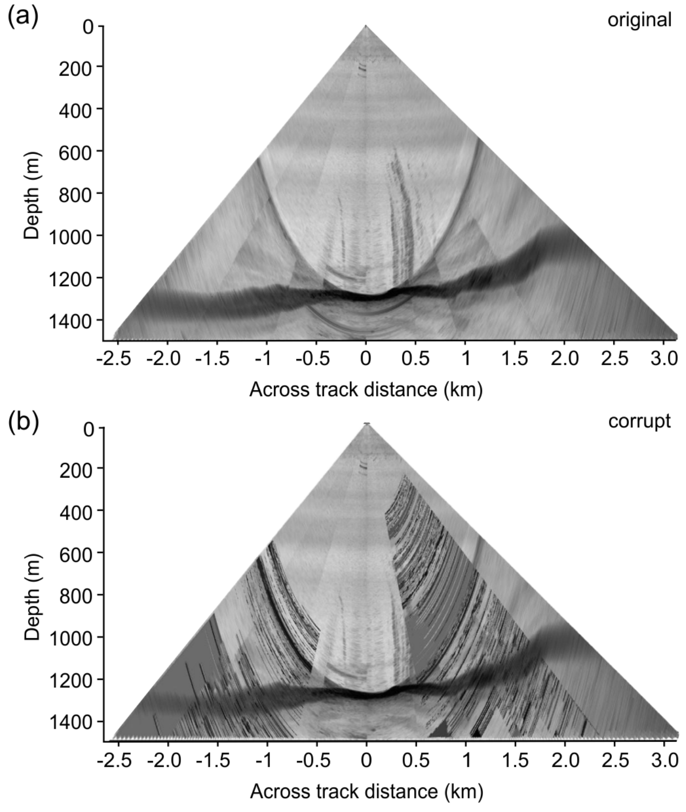

Fig. 14. Error resiliency test with FAPEC. Top panel shows the original water column scene, whereas bottom panel shows the same scene obtained from a corrupted FAPEC file.

the file (outside this specific region), which has resulted in an identical visualization without any damage.

V. Conclusions and forthcoming work

In this work we have presented a powerful and fully operational data compressor for multibeam echosounder water column datagrams, currently adapted to the Kongsberg Maritime file format but applicable to other vendors as well. We have tested this tool, FAPEC, on a variety of water column files including different sounder models and scenes, as well as different ping rates and scene complexity. When comparing FAPEC against typical data compressors such as *gzip*, we undoubtedly obtain the best lossless compression ratios, furthermore with a significantly faster execution. Additionally, higher ping rates and better sampling resolution leads to better compression ratios. Lossy compression can be selected by the user, allowing to configure a given level of quality loss which is implemented as individual sample values quantization. It translates into sharp images even at low quality levels, still allowing to detect steep features in the image such as gas seeps, fish shoals or sunken structures. All these advantages mean that continuous water column acquisition at a moderate data storage cost is finally feasible. FAPEC additionally provides some interesting features, such as multithreaded operation for an even faster execution, embedded data encryption, and resilience in front of data corruption. It also performs fast enough for real-time compression in ARM computing platforms even at high ping rates. Owing to its chunk-based operation, by monitoring the ratios obtained during a water column acquisition we should be able to automatically detect peculiar features in the scene.

We have identified some possible improvements that could make FAPEC an even better solution for water column data. First of all, its decorrelation algorithm may be further improved by considering not only correlation between neighbour samples but also with the neighbour pings. Wavelet-based spatial decorrelation will also be investigated, at least for the fraction of water column samples that allow creating a rectangular image. Even if wavelets provide significantly better results, we intend to maintain the current lossy approach, so the losses would then be applied prior to the wavelet stage in order to always keep the full image sharpness. Besides water column datagrams, improved compression of other datagram types (including bathymetry data) will also be investigated. An alternative lossy compression strategy, aiming at a given output data rate rather than a given quality, is being considered as well, together with an adaptive lossy algorithm to change the quality level depending on the scene uniformity. Finally, automatic features detection is a very interesting challenge which will be further investigated.

Acknowledgment

The authors would like to thank the reviewers for their useful and constructive comments. In memory of our friend and colleague Prof. Enrique García-Berro.

References

[1] K. Colbo, T. Ross, C. Brown, T. Weber, A review of oceanographic applications of water column data from multibeam echosounders, Estuarine, Coastal and Shelf Science 145 (2014) 41–56. `doi:10.1016/j.ecss.2014.04.002`.

[2] J. Beaudoin, Application of JPEG 2000 wavelet compression to multibeam echosounder mid-water acoustic refectivity measurements, Center for Coastal and Ocean Mapping 779.

[3] A. Chybicki, M. Moszyński, P. Poćwiardowski, Applications of compression techniques for reducing the size of multibeam sonar records, Proceedings of the 2008 1^{st} International Conference on Information Technology (2008) 1–4.

[4] A. Chybicki, Z. Lubniewski, M. Moszyński, Using wavelet techniques for multibeam sonar bathymetry data compression, Hydroacoustics 13 (2010) 31–38.

[5] L. Wu, A. Zielinski, J. Bird, Lossless compression of hydroacoustic image data, IEEE Journal of Oceanic Engineering 22 (1) (1997) 93–101.

[6] M. Moszynski, A. Chybicki, M. Kulawiak, Z. Lubniewski, A novel method for archiving multibeam sonar data with emphasis on efficient record size reduction and storage, Polish Maritime Research 20(1) (2013) 77–86.

[7] J. Portell, A. G. Villafranca, , E. García-Berro, Quick outlier-resilient entropy coder for space missions, Journal of Applied Remote Sensing 4 (2010) 339–363.

[8] J. Portell, R. Iudica, E. García-Berro, A. G. Villafranca, G. Artigues, FAPEC, a versatile and efficient data compressor for space missions, International Journal of Remote Sensing 39 (7) (2018) 2022–2042. `doi:10.1080/01431161.2017.1399478`.

[9] Gaia Collaboration, T. Prusti, J. H. J. de Bruijne, A. G. A. Brown, et al., The Gaia mission, A&A 595 (2016) A1. `doi:10.1051/0004-6361/201629272`.

[10] J. Portell, E. García–Berro, X. Luri, A. G. Villafranca, Tailored data compression using stream partitioning and prediction: application to Gaia, Experimental Astronomy 21 (2006) 125–149. `doi:10.1007/s10686-007-9078-1`.

[11] D. Huffman, A method for the construction of minimum redundancy codes, Proc. IRE 40 (1952) 1098–1101.

[12] R. F. Rice, Some practical universal noiseless coding techniques, Tech. Rep. JPL 79-22, Jet Propulsion Laboratory (1979).

[13] J. Portell, A. G. Villafranca, E. García–Berro, Designing optimum solutions for lossless data compression in space, in: Proceedings of the On-Board Payload Data Compression Workshop 2008, ESA, 2008, pp. 35–44.

[14] J. Portell, A. G. Villafranca, E. García-Berro, Outlier-Resilient Entropy Coding, Springer New York, 2011, Ch. 5, pp. 87–113. doi:10.1007/978-1-4614-1183-3_5.

[15] D. Amblas, J. Portell, X. Rayo, A. G. Villafranca, E. García–Berro, M. Canals, Real-time lossless compression of multibeam echosounder water column data, Instrumentation Viewpoint 19 (17) (2016) 41–43. doi:10.17863/CAM.7908.

[16] C. E. Shannon, A Mathematical Theory of Communication, University of Illinois Press, 1949.

[17] M. Evans, N. Hastings, B. Peacock, Statistical Distributions, Wiley-Interscience, 2000.

[18] S. W. Golomb, Run-lengths encodings, IEEE Trans. Info. Theory 12 (1966) 399–401. doi:10.1109/TIT.1966.1053907.

[19] D. Manda, M.-W. Thein, A. D'Amore, A. A. Armstrong, A low cost system for autonomous surface vehicle based hydrographic survey, in: U.S. Hydrographic Conference, 2015.

[20] A. G. Villafranca, S. Mignot, J. Portell, E. García-Berro, Hardware implementation of the FAPEC lossless data compressor for space, in: NASA/ESA Conference on Adaptive Hardware and Systems, 2010, pp. 170–176.

[21] G. A. Mitchell, D. L. Orange, J. J. Gharib, P. Kennedy, Improved detection and mapping of deepwater hydrocarbon seeps: optimizing multibeam echosounder seafloor backscatter acquisition and processing techniques, Marine Geophysical Research 39 (1) (2018) 323–347. doi:10.1007/s11001-018-9345-8.

[22] Consultative Committee for Space Data Systems. 2012. "Lossless Data Compression, Blue Book." *CCSDS Tech. Rep.*, 121.0-B-2, CCSDS.

[23] M. W. Marcellin, M. J. Gormish, A. Bilgin, M. P. Boliek, An overview of JPEG 2000, in: Proc. IEEE Data Compression Conf., Snowbird, UT, 2000, pp. 523–541.

Jordi Portell received the M.Sc. degree in Electronics Engineering (2000) and the Ph.D. degree in Applied Physics (2005) from the UPC. He is researcher at ICCUB and IEEC. Since 2000 he is working for the Gaia space astrometry mission of ESA, for which he proposed the on-board payload data handling and compression systems, has been the scientific manager of a study on the optimum on-board data compression algorithm, has been manager of the on-ground daily data processing system and of the data processing center of Barcelona, and is currently the operations and interface engineer at the DPAC project office. Since 2013 he is CTO at DAPCOM Data Services, and deputy technology director of the ICCUB since 2017. He has co-authored 10 peer-reviewed papers and over 30 proceedings, and co-advised 2 Ph.D. theses and 17 M.Sc. and B.Sc. theses.

David Amblas received the Ph.D. degree in Earth Sciences (2012) from the UB. He has been post-doctoral researcher at the Scott Polar Research Institute of the University of Cambridge (UK) from 2016 to 2018 (H2020 Marie Curie Fellowship). He is currently member of the CRG Marine Geosciences and assistant professor at UB. He has co-authored 41 peer-reviewed articles, 37 other publications including book chapters and scientific and educational material, 103 conference contributions (22 as first author) at international venues. He has participated in 24 research cruises in the Mediterranean Sea, the Atlantic, Pacific and Arctic oceans, and off Antarctica (2 as IP). He has been actively involved in 37 national and international research projects where he has provided marine geology expertise in research fields as diverse as geomorphology, sedimentology, climate change, tsunami risk, habitat mapping, offshore wind power and fisheries management.

Garrett Mitchell is involved in offshore oil and gas exploration projects that require integration of geophysical, geological, and geochemical data to evaluate deep-water offshore frontier basins for hydrocarbon seepage. His experience includes interpreting and analyzing multibeam echosounder data (bathymetry, backscatter, water column imagery), side-scan sonar data, ROV video imagery and geochemical data for hydrocarbon seep hunting surveys. Garrett has served as an offshore geological consultant and lead scientist on several hydrocarbon seep hunting cruises at Fugro. He works extensively on developing new data interpretation, analysis, and presentation techniques using Fledermaus Midwater software for hydrocarbon seep detection, mapping, and gas flux estimation.

Matias Morales background is electronics. Since 2003 he is working for Simrad Spain (subsidiary of Kongsberg Maritime in Spain), he started in the workshop department and he switched to the hydrographic department in 2005. From his beginnings in the company he has been trained and working in hydroacoustics, focused on biological and geological purposes, which includes all kind of multibeam technologies. He has been involved in more than 100 commissionings all around the world. In 2014 he started in Navigation and Dynamic Positioning Global Customer Support department in Spain as local coordinator. It was the first time that DP and Navigation system got support locally in Spain. In 2016 he was promoted to Subsea Customer Support Manager in Spain.

Alberto G. Villafranca received the M.Sc. degree in Telecommunication Engineering (2004) and the Ph.D. degree (2011) from the UPC. His Ph.D. topic was focused in data compression for space applications. He has been staggiare at the Observatorie de Paris and has been an Invited Scientist at the European Space Agency (ESA). After this experience he joined the Cartographic Institute of Catalonia (ICGC) for a small satellite mission study. Since 2012 he is with STAR-Dundee developing hardware designs, mainly dedicated to projects involving comunication protocols. He has been developing FPGA and ASIC designs for data compression and communication protocols for more than 10 years.

Riccardo Iudica is research collaborator for DAPCOM Data Services. He received his degree in Electronic Engineering from the University of Catania (2004) and the M.Sc. degree in Aerospace Science and Technology from the UPC (2014). He has worked on the HPA image compression algorithm, the multithread support and tailored pre-processing algorithms for nanosatellite data in FAPEC. His main research activities are focused on data and image processing and compression.

Galderic Lastras received the Ph.D. degree in Earth Sciences (2004) from the UB. He is Associate Professor at UB since 2011, and member of the CRG Marine Geosciences. He has co-authored 49 peer-reviewed articles, 35 other publications, and more than 150 conference contributions. His main fields of expertise are marine geomorphology and geophysics applied to seafloor and subseafloor exploration in diverse fields such as sedimentary processes and products, landslides, cold-water corals, tsunami generation and volcanism. He has been actively involved in 45 national and international research projects.