PROCEEDINGS OF SPIE

SPIEDigitalLibrary.org/conference-proceedings-of-spie

Hyperspectral compressive sensing: a low-power consumption approach

José M. P. Nascimento, Mário V'estias, Rui Duarte

José M. P. Nascimento, Mário V'estias, Rui Duarte, "Hyperspectral compressive sensing: a low-power consumption approach ," Proc. SPIE 10792, High-Performance Computing in Geoscience and Remote Sensing VIII, 1079202 (9 October 2018); doi: 10.1117/12.2326118

SPIE.

Event: SPIE Remote Sensing, 2018, Berlin, Germany

Downloaded From: https://www.spiedigitallibrary.org/conference-proceedings-of-spie on 10/20/2018 Terms of Use: https://www.spiedigitallibrary.org/terms-of-use

Hyperspectral Compressive Sensing - A Low Power Consumption Approach

José M. P. Nascimento[a,b], Mário V'estias[b,c], and Rui Duarte[c,d] *

[a]Instituto de Telecomunicações, Lisbon, Portugal
[b]Instituto Superior de Engenharia de Lisboa, IPL, Lisbon, Portugal
[c]INESC-ID, Lisbon, Portugal
[d]Instituto Superior Técnico, Universidade de Lisboa, Lisbon, Portugal

ABSTRACT

Hyperspectral imaging instruments allow data collection in hundreds of spectral bands for the same area on the surface of the Earth. The resulting multidimensional data cube typically comprises several GBs per flight. Due to the extremely large volumes of data collected by imaging spectrometers, hyperspectral data compression, dimensionality reduction and Compressive Sensing (CS) techniques has received considerable interest in recent years. These data are usually acquired by a satellite or an airbone instrument and sent to a ground station on Earth for subsequent processing. Usually the bandwidth connection between the satellite/airborne platform and the ground station is reduced, which limits the amount of data that can be transmitted. As a result, there is a clear need for (either lossless or lossy) hyperspectral data compression techniques that can be applied on-board the imaging instrument.

This paper, presents a study of the power and time consumption and accuracy of a parallel implementation for a spectral compressive acquisition method on a Jetson TX2 platform, which is well suited to perform vector operations such as dot products. This implementation exploits the architecture at low level, using shared memory and coalesced accesses to memory. The conducted experiments have been performed to demonstrate the applicability, in terms of accuracy, time consuming and power consumption of these methods for onboard processing. The results show that by using this low power consumption GPU is it possible to obtain real-time performance with a very limited power requirement.

1. INTRODUCTION

Hyperspectral imaging instruments allow remote Earth exploration by measuring hundreds of spectral bands at very narrow channels of a given area. One of the main problems in the analysis of hyperspectral data is the presence of mixed pixels, as many of the pixels collected by imaging spectrometers are highly mixed in nature due to spatial resolution and other phenomena. In this case, several spectrally pure signatures (endmembers) are combined into the same (mixed) pixel. Spectral unmixing[1] involves the separation of a pixel spectrum into its endmembers spectra, and the estimation of the abundance value for each endmember. The linear mixture model has been the most popular tool used to unmix remotely sensed hyperspectral data.[2]

Due to the extremely large volumes of data collected by imaging spectrometers, hyperspectral data compression has received considerable interest in recent years.[3,4] These data are usually acquired by a satellite or an

* This work was supported in part by the Instituto de Telecomunicações and in part by the Portuguese Science and Technology Foundation under Project UID/EEA/50008/2013.

High-Performance Computing in Geoscience and Remote Sensing VIII, edited by Bormin Huang, Sebastián López, Zhensen Wu, Proc. of SPIE Vol. 10792, 1079202 · © 2018 SPIE
CCC code: 0277-786X/18/$18 · doi: 10.1117/12.2326118

Downloaded From: https://www.spiedigitallibrary.org/conference-proceedings-of-spie on 10/20/2018
Terms of Use: https://www.spiedigitallibrary.org/terms-of-use

airborne instrument and sent to a ground station on Earth for subsequent processing. Usually the bandwidth connection between the satellite/airborne platform and the ground station is reduced, which limits the amount of data that can be transmitted. As a result, there is a clear need for (either lossless or lossy) hyperspectral data compression techniques that can be applied onboard the imaging instrument.

Traditional compression and dimensionality reduction techniques include complex algorithms, such as Principal Component Analysis or Discrete Wavelet Transforms[5] are computational expensive which limits their use for onboard applications. However, Compressive Sensing (CS) techniques are characterized for recovering the signal from a low number of linear measurements.[6] This reduces the amount of data that needs to be measured, transmitted and stored in first place. The CS paradigm is based on performing random projections over the signal of interest and the bulk of the processing to reconstruct the original image is performed on the Earth station, where plenty of computing and storage resources are available compared with the scare resources available onboard.

Over the last decade several methods to perform CS in the spectral domain of Hyperspectral imagery have been proposed.[7–10] These algorithms are able to reconstruct the original hyperspectral image from a low number of random projections in the spectral domain. This is only possible if the spectral vectors live in a low dimensional subspace, which is a very good approximation in most hyperspectral images (HSIs) of the real world, namely when the observed spectral vectors are well approximated by linear mixing model (LMM).[1,11] In this way, CS algorithms are able to reconstruct the original hyperspectral image with a number of measurements per pixel in the order of the size of this subspace, which typically is much lower than the number of bands of the sensor.

Typically, CS measurement process is implemented on the optic hardware of the sensor.[12] However it would be feasible to implement the measurement process on the electronic hardware, as a light weight alternative to traditional algorithms for lossy compression or dimensionality reduction.[9] In this way, the signal would be compressed, as soon as, it is measured by the sensor and therefore there is no need for storage the full signal at full resolution on the sensor neither to transmit it to the Earth station.

Due to the fact that in CS the measurement process is based on performing dot products between random vectors and the signal of interest, a good suitable candidate for performing this task would be Graphics Processing Units (GPUs), which offer exceptional performance in vector and matrix operations. In this regard, several implementations of CS and random projections algorithms over hyperspectral data with GPUs have been proposed,[13–15] concluding that by using GPUs it is possible to achieve real-time performance for the random projection step. On the other hand, the power consumption requirements of this hardware makes them ineffective for onboard applications. Fortunately, over the last years, the advances in semiconductor industry and the huge interest on developing mobile devices have allowed companies such as Nvidia to develop low power GPUs like the Jetson TX2, which is a low power consumption GPUs, that nevertheless, can achieve high throughput in image processing applications at the same time.

This paper explores the possibility to use these low power consumption GPUs to perform the Hyperspectral CS process as an alternative to traditional compression and dimensionality reduction algorithms performed on common GPU boards. In section 2, a CS method called Hyperspectral Coded Aperture method (HYCA) is summarized, which have been chosen in this work as CS algorithm to demonstrate the performance of such methods on low power hardware. In section 3, a brief description of the GPU architecture and the main features of the Jetson TX2 hardware used on the experiments is provided. In section 4, a set of experiments are conducted

Downloaded From: https://www.spiedigitallibrary.org/conference-proceedings-of-spie on 10/20/2018
Terms of Use: https://www.spiedigitallibrary.org/terms-of-use

to demonstrate the effectiveness of this hardware, performing the random projections in real-time. Finally, section 5 presents some conclusions and future lines of work.

2. COMPRESSIVE SENSING METHOD

In this section a CS method named Hyperspectral Coded Aperture (HYCA)[7] is briefly described. This approach compresses the data on the acquisition process, then the compressed signal is sent to Earth and stored in compressed form. Later the original signal can be recovered by taking advantage of the fact that the hyperspectral data can be explained using a reduced set of spectral endmembers due to the mixing phenomenon[16] and also exploits the high spatial correlation of the fractional abundances associated to the spectral endmembers. This method for its characteristics is well suited to be developed in a parallel fashion.[13]

Let $\mathbf{x} \in \mathbb{R}^{n_b \times n_p}$ represent, in vector format, a hyperspectral image with n_b spectral bands and $n_p := n_r \times n_c$ pixels where n_r and n_c denote, respectively, the number of rows and columns of the hyperspectral image in the spatial domain. The ordering of $\mathbf{x}$ correspond to all image pixels for each spectral band. In order to perform the compression of the original signal $\mathbf{x}$, and as in [17], for each pixel $i \in \{1, \ldots, n_p\}$, a set of q inner products between $\mathbf{x}_i$ and samples of i.i.d. Gaussian random vectors is performed. The total number of measurements is therefore qn_p yielding an undersampling factor of q/n_b. This measurement operation can be represented as a matrix multiplication $\mathbf{y} = \mathbf{A}\mathbf{x}$, where $\mathbf{A}$ is a block diagonal matrix containing the matrices $\mathbf{A}_i \in \mathbb{R}^{q \times n_b}$ acting on the pixel $\mathbf{x}_i$, for $i \in \{1, \ldots, n_p\}$. Since the image can be very large, measurement strategy splits the dataset into different windows of size $m = ws \times ws$ and then repeat the matrices $\mathbf{A}_i$ used in each window, thus requiring to store in memory just m different $\mathbf{A}_i$ matrices.

Let us now define the linear operator $\mathbf{x} = (\mathbf{I} \otimes \mathbf{E})\mathbf{z}$, where the matrix $\mathbf{E}$ represents the basis of the subspace where the data lives,[16] $\mathbf{I}$ is the identity matrix and the vector $\mathbf{z}$ contains the coefficients. In this work, the $\mathbf{E}$ matrix contains the p endmembers of the data set by columns obtained in a very fast way through the vertex component analysis (VCA) algorithm,[11] thus $\mathbf{z}$ contains the fractional abundances associated to each pixel.

Let us now assume that $\mathbf{K} = \mathbf{H}(\mathbf{I} \otimes \mathbf{E})$. If matrices $\mathbf{E}$ and $\mathbf{H}$ are available, one can formulate the estimation of $\mathbf{z}$ with from q-dimensional vector of measurements. Since the fractional abundances in hyperspectral images exhibit a high spatial correlation, we exploit this feature for estimating $\mathbf{z}$ using the following optimization problem:

$$\min_{\mathbf{z} \geq \mathbf{0}} (1/2)\|\mathbf{y} - \mathbf{K}\mathbf{z}\|^2 + \lambda_{TV}\mathrm{TV}(\mathbf{z}). \tag{1}$$

Therefore, the minimization (1) aims at finding a solution which is a compromise between the fidelity to the measured data, enforced by the quadratic term $(1/2)\|\mathbf{y} - \mathbf{K}\mathbf{z}\|^2$, and the properties enforced by the TV regularizer, that is piecewise smooth image of abundances. The relative weight between the two characteristics of the solution is set the regularization paremeter $\lambda_{TV} > 0$.

To solve the convex optimization problem in Eq. (1), a set of new variables per term of the objective function were used and the ADMM methodology[18] has been adopted to decompose very hard problems into a cyclic sequence of simpler problems. With this in mind, an equivalent way of writing the optimization problem in Eq. (1) is

Downloaded From: https://www.spiedigitallibrary.org/conference-proceedings-of-spie on 10/20/2018
Terms of Use: https://www.spiedigitallibrary.org/terms-of-use

$$\min_{\mathbf{z}} \frac{1}{2}\|\mathbf{y} - \mathbf{K}\mathbf{z}\|^2 + \lambda_{TV}\, \boldsymbol{\phi}(\mathbf{D}\mathbf{z}) + \iota_{R+}(\mathbf{z}), \tag{2}$$

where $\iota_{R+}(\mathbf{z}) = \sum_{i=1}^{pn_p} \iota_{R+}(\mathbf{z}_i)$ is the indicator function ($\mathbf{z}_i$ represents the ith element of $\mathbf{z}$ and $\iota_{R+}(\mathbf{z}_i)$ is zero if $\mathbf{z}_i$ belongs to the nonnegative orthant and $+\infty$ otherwise). Given the objective function in (2), we can write the following equivalent formulation:

$$\begin{aligned} \min_{\mathbf{z},\mathbf{v}_1,\mathbf{v}_2,\mathbf{v}_3,\mathbf{v}_4} \quad & \frac{1}{2}||\mathbf{y} - \mathbf{K}\mathbf{v}_1||^2 + \iota_{R+}(\mathbf{v}_2) + \lambda_{TV}\, \boldsymbol{\phi}(\mathbf{D}\mathbf{z}) \\ \text{subject to} \quad & \mathbf{v}_1 = \mathbf{z} \\ & \mathbf{v}_2 = \mathbf{z} \\ & (\mathbf{v}_3, \mathbf{v}_4) = \mathbf{D}\mathbf{z}, \end{aligned} \tag{3}$$

Algorithm 1 shows the pseudo-code of the HYCA algorithm to solve the problem in (3) and how to reconstruct the data using the linear mixture model.

Algorithm 1 Pseudocode of HYCA algorithm.

1. Initialization: set $k = 0$, choose $\mu > 0$, $\mathbf{E}$, $\mathbf{z}^{(\mathbf{0})}$, $\mathbf{v}_1^{(0)}$, $\mathbf{v}_2^{(0)}$, $\mathbf{v}_3^{(0)}$, $\mathbf{v}_4^{(0)}$, $\mathbf{d}_1^{(0)}$, $\mathbf{d}_2^{(0)}$, $\mathbf{d}_3^{(0)}$, $\mathbf{d}_4^{(0)}$
2. repeat:
  3. $\mathbf{A} \leftarrow (\mathbf{v}_1^{(k)} + \mathbf{d}_1^{(k)} + \mathbf{v}_2^{(k)} + \mathbf{d}_2^{(k)} + \mathbf{D}_h^T(\mathbf{v}_3^{(k)} + \mathbf{d}_3^{(k)}))$
  4. $\mathbf{z}^{(k+1)} \leftarrow (\mathbf{D}^T\mathbf{D} + 2\mathbf{I})^{-1}\left(\mathbf{A} + \mathbf{D}_v^T(\mathbf{v}_4^{(k)} + \mathbf{d}_4^{(k)})\right)$
  5. $\mathbf{v}_1^{(k+1)} \leftarrow (\mathbf{K}^T\mathbf{K} + \mu\mathbf{I})^{-1}\left(\mathbf{K}^T\mathbf{y} + \mu(\mathbf{z}^{(k+1)} - \mathbf{d}_1^{(k)})\right)$
  6. $\mathbf{v}_2^{(k+1)} \leftarrow \max\left(0, \mathbf{z}^{(k+1)} - \mathbf{d}_2^{(k)}\right)$
  7. $\mathbf{v}_3^{(k+1)} \leftarrow \text{soft}\left(\mathbf{D}_h(\mathbf{z}^{(k+1)}) - \mathbf{d}_3^{(k)}, \lambda_{TV}/\mu\right)$
  8. $\mathbf{v}_4^{(k+1)} \leftarrow \text{soft}\left(\mathbf{D}_v(\mathbf{z}^{(k+1)}) - \mathbf{d}_4^{(k)}, \lambda_{TV}/\mu\right)$
  9. Update Lagrange multipliers:
    $\mathbf{d}_1^{(k+1)} \leftarrow \mathbf{d}_1^{(k)} - \mathbf{z}^{(k+1)} + \mathbf{v}_1^{(k+1)}$
    $\mathbf{d}_2^{(k+1)} \leftarrow \mathbf{d}_2^{(k)} - \mathbf{z}^{(k+1)} + \mathbf{v}_2^{(k+1)}$
    $\mathbf{d}_3^{(k+1)} \leftarrow \mathbf{d}_3^{(k)} - \mathbf{D}_h\mathbf{z}^{(k+1)} + \mathbf{v}_3^{(k+1)}$
    $\mathbf{d}_4^{(k+1)} \leftarrow \mathbf{d}_4^{(k)} - \mathbf{D}_v\mathbf{z}^{(k+1)} + \mathbf{v}_4^{(k+1)}$
  10. Update iteration: $k \leftarrow k + 1$
11. until $k =$ MAX_ITERATIONS
12. Reconstruction $\hat{\mathbf{x}} = (\mathbf{I} \otimes \mathbf{E})\mathbf{z}^k$

3. GPU ARQUITECTURE

In recent years, graphics processing units (GPUs) have evolved into highly parallel and programmable systems.[19] Specifically, several hyperspectral imaging algorithms have shown to be able to benefit from this hardware taking advantage of the extremely high floating-point processing performance, compact size, huge memory bandwidth, and relatively low cost of these units, which make them appealing for onboard data processing, specially for CS applications, due to the fact that the random projection process typically involves Matrix-Vector products which are a perfect match for the GPU architecture.[13] However, one of the main problems for the use of this hardware onboard is the high power and energy consumption that they require. Remote sensing missions frequently perform the bulk of data processing and storage onboard of airborne devices and satellites, which may impose severe constraints on the power and energy consumption (e.g., due to battery life time or electricity being produced by the attached solar panels). The combination of the high dimensionality of hyperspectral images,

Downloaded From: https://www.spiedigitallibrary.org/conference-proceedings-of-spie on 10/20/2018
Terms of Use: https://www.spiedigitallibrary.org/terms-of-use

Figure 1. Jetson TX2 development kit used in the experiments.

very demanding processing methods and energy restrictions justifies the exploration of high-performance yet low-power technologies together with energy-aware novel computational algorithms that can produce a response in real time or near real time while minimizing the energy usage.

In this work, the parallel development of the coder side of HYCA method on a low power consumption GPU such as the Jetson TX2 is explored. The TX2 employs an SOC (system-on-chip) design that incorporates a quad-core 2.0-GHz 64-bit ARMv8 A57 processor, a dual-core 2.0-GHz superscalar ARMv8 Denver processor, and an integrated Pascal GPU. There are two 2-MB L2 caches, one shared by the four A57 cores and one shared by the two Denver cores. The GPU has two streaming multiprocessors (SMs), each providing 128 1.3-GHz cores that share a 512-KB L2 cache. The six CPU cores and integrated GPU share 8 GB of 1.866-GHz DRAM memory.[20] The Jetson TX2 typically draws between 7.5 and 15 watts with a voltage input of 5.5V-19.6V DC and requires minimal cooling and additional space.

CUDA architecture has been used for implementing the parallel HYCA (P-HYCA) measurement process. The CUDA programming model is designed to develop applications which scales the parallelism in a transparently way, independently from the number of processors or multiprocessors of the hardware. In order to do that, CUDA defines the so called kernels, which are functions to be processed in parallel. For each kernel, the user structures the parallelism into a grid of blocks, each one processed by a GPU multiprocessor. Each block is divided into threads, which are the smallest processing units in the architecture. The different threads may synchronize with the other threads of the same block and communicate with them through the shared memory of the multiprocessor. However the different blocks of the grid are executed in parallel with any sort of synchronization. Specifically in the implementation of P-HYCA each thread perform the measurement process of one pixel, thus, each thread performs the operation $\mathbf{y}_i = \mathbf{A}_j\mathbf{x}_i$ with $i \in 1 \dots n$ and $j \in 1 \dots m$. The number of threads per block is the maximum allowed by the architecture, which in the case of the Jetson TX2 is 1024. Due to the fact that many threads in the block uses the same $\mathbf{A}_j$ matrices, these matrices are loaded into shared memory before processing the dot products, in this way we reduce the memory accessed required to global memory.

4. RESULTS

In this section a series of experiments are conducted in order to evaluate the performance of the Jetson TX1 on performing the measurement process of the CS method. Herein, we focus on the coder-side of the method since

Downloaded From: https://www.spiedigitallibrary.org/conference-proceedings-of-spie on 10/20/2018
Terms of Use: https://www.spiedigitallibrary.org/terms-of-use

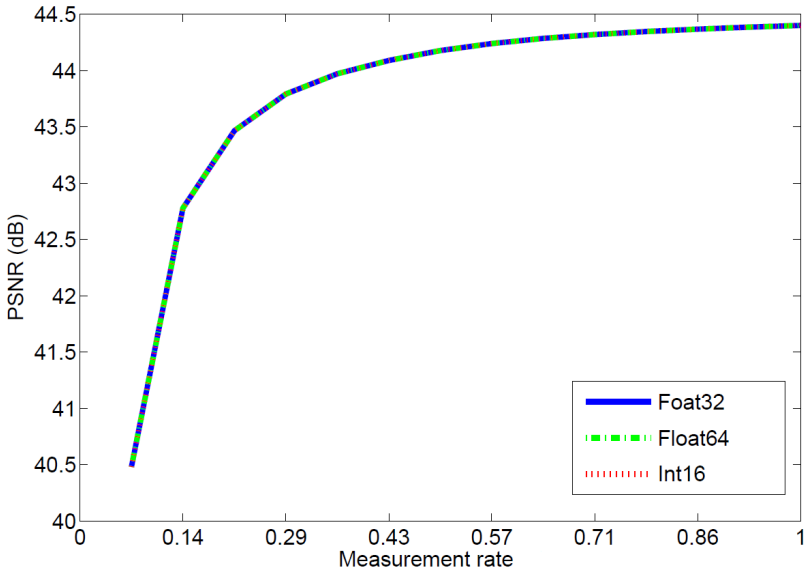

Figure 2. Peak Signal to Noise Ratio (PSNR) in decibels (dB) between the original and reconstructed image after compression and decompression for different measurement ratios using different data types: float64, float32, and int16.

it is the one to be performed onboard, while the decoder side may be performed on the Earth station where plenty of hardware may be available.

As described in the previous section, the experiments are conducted with the Jetson TX2 which is a low power consumption platform that works with a maximum power of 15W. It incorporates a Nvidia Pascal™GPU with 256 NVIDIA CUDA cores, a quad-core 2.0-GHz 64-bit ARMv8 A57 processor, a dual-core 2.0-GHz superscalar ARMv8 Denver processor, 8GB LPDDR4 memory and 16GB eMMC 5.1 flash storage. The Fig. 1 shows the Jetson TX2 development kit used for performing the experiments.

The simulated data set used in this experiments was generated from spectral signatures randomly selected from the United States Geological Survey (USGS)[†]. The dataset size is set to 614 samples times 512 lines and 224 bands, which is the same size as an AVIRIS sensor image. In order to evaluate the performance of the hardware with different data types, three different versions of the same image were generated with different data types: 64 bits double precision floating point (float64), 32 bits simple precision floating point (float32) and 16 bits short integer (int16), respectively. The int16 version of the image was generated multiplying the original reflectance values by 5000 and rounding to the nearest integer. The matrices $\mathbf{A}_j$ were generated following random Gaussian i.i.d., three different versions of the matrices with different data types float32, float16 and int16 were generated. The resulting measurements generated $\mathbf{y}_i$ were stored in float32, float16 and int32 data types, respectively. In the case of the integer data type it was necessary to store the resulting measurements using int32 data type in order to avoid overflows in the dot products.

In order to test the accuracy of the P-HYCA implementation with different data types, the image was reconstructed with the P-HYCA reconstruction algorithm in a desktop computer with a desktop GPU GTX 980. For the reconstruction process, the measurement matrices $\mathbf{A}_j$ and the compressed measurements $\mathbf{y}_i$ were transformed to double precision floating point (float64), providing similar reconstruction accuracy with negligible deviations in the case of all data types. The Fig. 2 shows the results in terms of PSNR between the original and

[†]http://speclab.cr.usgs.gov

Downloaded From: https://www.spiedigitallibrary.org/conference-proceedings-of-spie on 10/20/2018
Terms of Use: https://www.spiedigitallibrary.org/terms-of-use

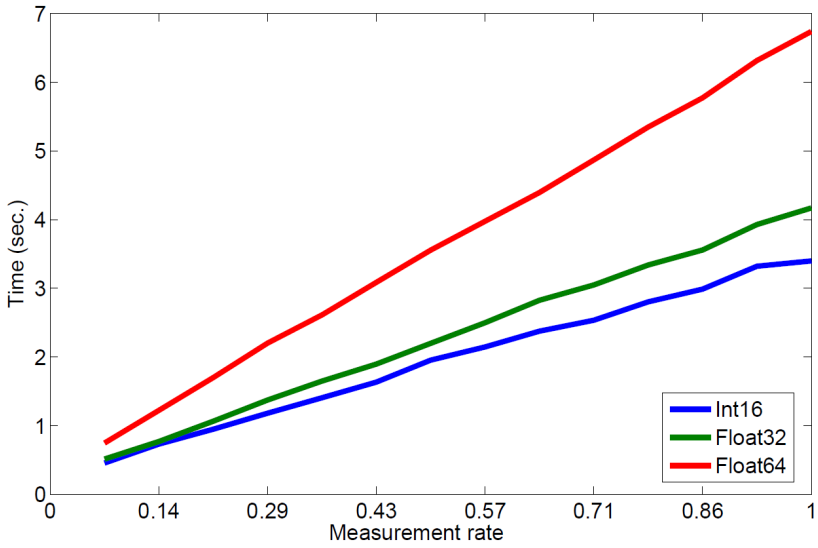

Figure 3. Processing times in seconds for the Jetson TX2 hardware for different measurement ratios using different data types: float64, float32, and int16.

the reconstructed image in the case of the three different data types. The figure shows that the differences between the different data types are negligible and the figure shows that P-HYCA may achieve great reconstruction accuracy using a measurement rate q/n_b of one third.

On the other hand, the Fig. 3 show the execution times in seconds corresponding to the compression process performed in the Jetson TX2 for the three different data types considered. The figure shows that, as expected, the compression process performs much better when int16 and float32 data types are used than when using double precision floating point operations. As a result we may conclude that by using int16 operations it is possible to achieve very similar reconstruction accuracy while at the same time reducing considerably the computational cost. Furthermore this figure also shows that for a measurement ratio of one third, which was demonstrated as enough for an accurate reconstruction, the Jetson TX2 performs compression process in about 1 second in the case of the integer and single precision floating point and about 2 seconds in the case of the double precision floating point. Fig. 4 shows the average power consumed for different measurement rate q/n_b and for the three types of data used in experiments. One can notice that int16 data type is the one that consume less power, however for a measurement ratio of one third the power is very similar among the considered data types.

Finally the performance of the compression process with regards different window sizes ranging from 8×8 to 64×64 was evaluated. For this experiment the int16 data type was used. In the cases in which the window size is grater than 8×8, the matrices $\mathbf{A}_j$ do not fit in the shared memory, therefore in those cases the shared memory is not used. Fig. 5 shows the execution times for different measurement ratios for different window sizes in different colors. As expected, the execution time scale with the window size. This is because the higher the number of matrices $\mathbf{A}_j$ the higher the amount of data to be loaded from the global memory. Furthermore, with higher window size is less likely that two pixels processed in the same block use the same $\mathbf{A}_j$ matrix, resulting in more cache memory faults and increasing the execution time due to the increase of memory accesses. Regarding the average power consumption, one can see, on Fig. 6 that it is higher for larger window size, as expected.

Downloaded From: https://www.spiedigitallibrary.org/conference-proceedings-of-spie on 10/20/2018
Terms of Use: https://www.spiedigitallibrary.org/terms-of-use

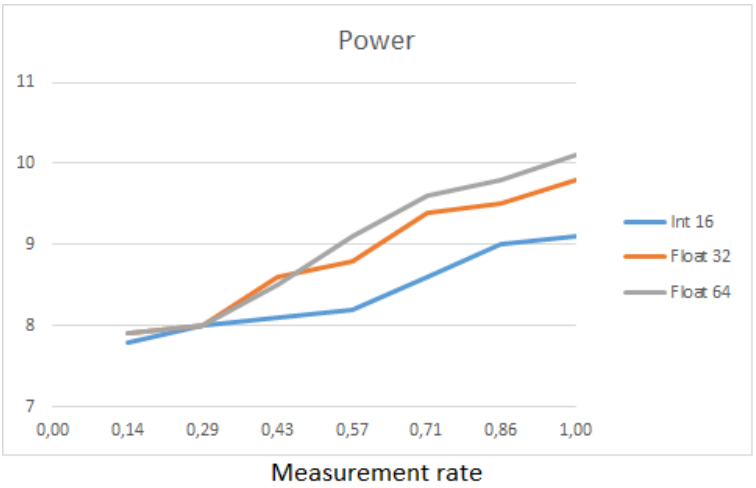

Figure 4. Average power in Watts (W) for different measurement ratios using different data types: float64, float32, and int16.

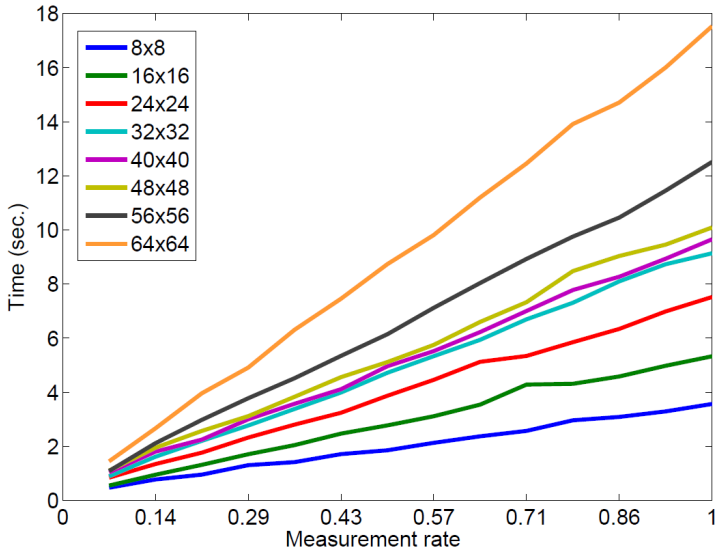

Figure 5. Processing times in seconds for the Jetson TX2 hardware for different measurement ratios using different window sizes from 8×8 to 64×64.

Downloaded From: https://www.spiedigitallibrary.org/conference-proceedings-of-spie on 10/20/2018
Terms of Use: https://www.spiedigitallibrary.org/terms-of-use

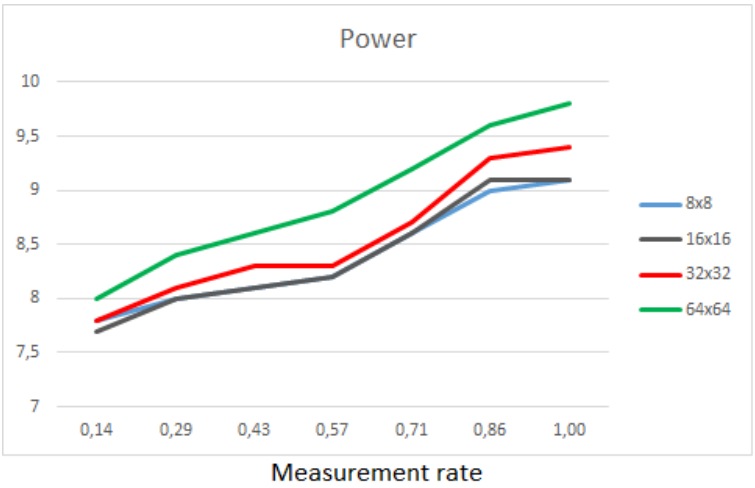

Figure 6. Average power in Watts (W) for the Jetson TX2 hardware for different measurement ratios using different window sizes from 8×8 to 64×64.

5. CONCLUSIONS AND FUTURE LINES

In this paper the use of compressive sensing techniques as an alternative to traditional compression techniques for onboard compression using a low power consumption GPU Jetson TX2 have been proposed. Several experiments using different data types have been performed over synthetic dataset, revealing that the use of integer data types does not affect the accuracy in the reconstruction of the compressed data, while at the same time, reduces significantly the processing time onboard. The study also reveals that for integer data type the average power consumption is smaller on the Jetson TX2 board. The presented times also reveals that real-time processing in the task of compressing hyperspectral images can be achieved. In future further research may be done using real hyperspectral data and comparing the results of the proposed methodology with other traditional compression schemes both in terms of accuracy of the reconstruction, computational performance and power consumption.

REFERENCES

1. J. Bioucas-Dias, A. Plaza, N. Dobigeon, M. Parente, Q. Du, P. Gader, and J. Chanussot, "Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **99**(1-16), 2012.
2. N. Keshava and J. F. Mustard, "Spectral unmixing," *IEEE Signal Process. Mag.* **19**(1), pp. 44–57, 2002.
3. G. Motta, F. Rizzo, and J. A. Storer, *Hyperspectral data compression*, Berlin: Springer, 2006.
4. B. Huang, *Satellite data compression*, Berlin: Springer, 2011.
5. Q. Du and J. E. Fowler, "Hyperspectral image compression using jpeg2000 and principal component analysis," *IEEE Geoscience and Remote Sensing Letters* **4**(2), pp. 201–205, 2007.
6. D. L. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory* **52**(4), pp. 1289–1306, 2006.
7. G. Martin, J. M. Bioucas-Dias, and A. Plaza, "Hyca: A new technique for hyperspectral compressive sensing," *IEEE Transactions on Geoscience and Remote Sensing* **53**(5), pp. 2819–2831, 2014.
8. J. E. Fowler and Q. Du, "Reconstructions from compressive random projections of hyperspectral imagery," in *Optical Remote Sensing*, pp. 31–48, Springer, 2011.

Downloaded From: https://www.spiedigitallibrary.org/conference-proceedings-of-spie on 10/20/2018
Terms of Use: https://www.spiedigitallibrary.org/terms-of-use

9. G. Martín and J. M. Bioucas-Dias, "Hyperspectral blind reconstruction from random spectral projections," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **9**(6), pp. 2390–2399, 2016.
10. P. V. M. Golbabaee, S. Arberet, "Compressive source separation: Theory and methods for hyperspectral imaging," *IEEE Transactions on Image Processing* **22**(12), pp. 5096–5110, 2013.
11. J. M. P. Nascimento and J. M. Bioucas-Dias, "Vertex Component Analysis: A Fast Algorithm to Unmix Hyperspectral Data," *IEEE Trans. Geosci. Remote Sens.* **43**(4), pp. 898–910, 2005.
12. A. Wagadarikar, R. John, R.-W. D., and Brady, "Single disperser design for coded aperture snapshot spectral imaging," *Applied optics* **47**(10), pp. B44–B51, 2008.
13. S. Bernabe, G. Martin, J. Nascimento, J. Bioucas-Dias, A. Plaza, and V. Silva, "Parallel hyperspectral coded aperture for compressive sensing on gpus," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing,* **PP**(99), pp. 1–14, 2015.
14. J. Sevilla, G. Martín, J. Nascimento, and J. Bioucas-Dias, "Hyperspectral image reconstruction from random projections on gpu," in *Geoscience and Remote Sensing Symposium (IGARSS), 2016 IEEE International*, pp. 280–283, 2016.
15. J. Sevilla, G. Martin, and J. M. Nascimento, "Parallel hyperspectral image reconstruction using random projections," in *SPIE Remote Sensing*, pp. 1000707–1000707–9, International Society for Optics and Photonics, 2016.
16. J. M. Bioucas-Dias and J. M. P. Nascimento, "Hyperspectral subspace identification," *IEEE Trans. Geosci. Remote Sens.* **46**(8), pp. 2435–2445, 2008.
17. G. Martn, J. M. Bioucas-Dias, and A. Plaza, "Hyca: A new technique for hyperspectral compressive sensing," *IEEE Transactions on Geoscience and Remote Sensing* **53**, pp. 2819–2831, May 2015.
18. J. Eckstein and D. Bertsekas, "On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators," *Mathematical Programming* **5**, pp. 293–318, 1992.
19. J. Sevilla, S. Bernabe, and A. Plaza, " Unmixing-based content retrieval system for remotely sensed hyperspectral imagery on GPUs," *The Journal of Supercomputing* **70**(2), pp. 588–599, 2014.
20. T. Amert, N. Otterness, M. Yang, J. H. Anderson, and F. D. Smith, "Gpu scheduling on the nvidia tx2: Hidden details revealed," in *2017 IEEE Real-Time Systems Symposium (RTSS)*, pp. 104–115, Dec 2017.

Downloaded From: https://www.spiedigitallibrary.org/conference-proceedings-of-spie on 10/20/2018
Terms of Use: https://www.spiedigitallibrary.org/terms-of-use