Euclid NIR GPU: Embedded GPU-accelerated Near-Infrared Image Processing for On-board Space Systems

Ivan Rodriguez*
Leonidas Kosmidis
Barcelona Supercomputing Center (BSC)
Universitat Politècnica de Catalunya (UPC)

Figure 1: The pillars of creation in visible color spectrum in the left and in near-infrared in the right. Image copyright NASA.

1 INTRODUCTION

The current trend in aerospace is the constant increase of performance requirements for on-board processing, to support the higher data rates and autonomy required across all space domains including science and robotic exploration, telecom and navigation [Rosello 2018]. Such performance cannot be provided by existing space CPUs such as the NGMP [Hjorth et al. 2015] or PowerPC 750 [Berger et al. 2001], so new technologies including COTS devices from other critical domains like automotive are currently explored.

Embedded GPUs are considered among the most promising enabling technologies, since they can offer both high performance software processing (as opposed to existing high-performance hardware processing provided by FPGAs for space) as well as low power consumption below 10W [Kosmidis et al. 2019], which is the limit of feasible thermal dissipation of a single component in a space system. However, so far there is limited practical indication in industry on whether existing on-board software is a good fit for the massively parallel GPU programming model and more importantly whether embedded GPUs can provide the processing requirements of future missions. In fact, until now only the standalone lossless compression algorithm CCSDS-123 [Davidson and Bridges 2017] has been demonstrated in a non-embedded GPU, as well as simple, non-space specific image processing algorithms from existing GPU libraries [Tsog et al. 2018] on a custom designed embedded GPU platform for space. However, there is a lack of complex case studies in the literature.

We implement for the first time a demonstrator of complex on-board algorithm running on a GPU, the H2RG infrared detector [Jung and Crouzet 2012] used in several existing and future space missions such as NASA's Hubble Space Telescope, its successor James Webb Space Telescope (JWST) and ESA's Euclid astronomy and astrophysics space mission [Crouzet 2011]. We demonstrate not only the feasibility of porting such a case study on a GPU, but also its performance benefits by executing it on an embedded COTS GPU platform designed for the automotive domain, which has similar reliability requirements with low earth orbit (LEO) missions.

2 APPLICATION AND GPU PARALLELISATION

The baseline sequential version of the Near Infrared (NIR) H2RG BM case study is based on ESA's implementation for the Euclid mission. The application processes (N) raw images acquired from the HAWAII-2 sensor – an integrated circuit for visible and infrared astronomy applications – to produce a single output image. More details are provided in [Jung and Crouzet 2012].

Although the algorithm operates over image matrices, the GPU parallelisation of the H2RG algorithm is not straightforward, since a significant part of the application requires accessing data with complex loop dependencies, complicating its GPU implementation. Despite this challenge, we managed to port the entire application to the GPU, so that the CPU is only responsible to offload computations to the GPU.

The application was targeted to an NVIDIA embedded GPU so for that reason we perform our implementation in CUDA. Minor modifications were needed in the original code to allow the porting to GPU code. In Figure 2 we present a high level view of the parallelisation strategy we have followed. In order to optimise our implementation, we took advantage of parallelism in several levels: at frame level, application stage level and intra-kernel.

As shown in the figure, we process consecutive frames in different processing pipelines, which are known as *streams* in CUDA terminology. This allows a processing scheme in which the memory transfer of one frame is overlapped in time with the GPU execution of the application stages of the previous frame. Within each processing pipeline, the stages of the algorithm are executed one after the other. Moreover, when the data copy one of the frames has finished,

*{name.surname}@bsc.es

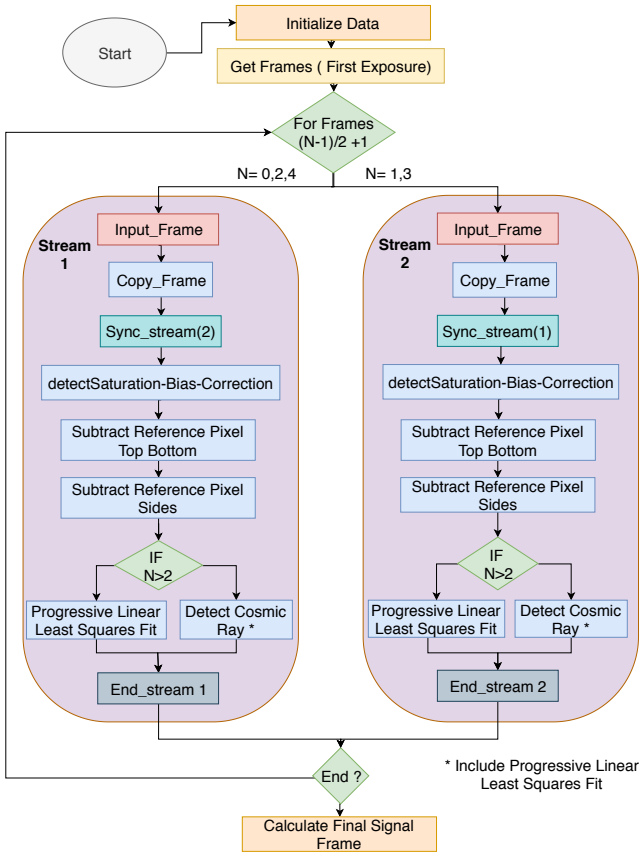

Figure 2: High level view of the H2RG case study GPU implementation.

its stages are also executed on the GPU, since the GPU is capable of executing multiple kernels from different streams concurrently, when both their computational resources can be satisfied. In addition, even when it is not possible to execute two kernels at the same time, the utilisation of the GPU is maximised, since there is always a delay between the CPU offloading of one kernel and its execution and termination, which could leave the GPU underutilised if only one stream was used.

Our implementation is parametric with respect to the above described different parallelism factors, so the same code implementation can be configured to be executed efficiently on other embedded GPUs, too.

In our platform the optimal performance has been achieved by processing 2 frames in parallel. Further increasing the number of parallel frames only increased the memory consumption – since each image pipeline requires its own state for processing a frame such as the input image and its intermediate results – without performance gains or resulting in slowdown.

3 RESULTS

We optimised our code on the NVIDIA's Xavier platform, designed to reach the highest certification level in the automotive domain, ASIL-D. As such, it includes reliability features such as ECC and as a consequence, it can be used for space missions up to LEO without

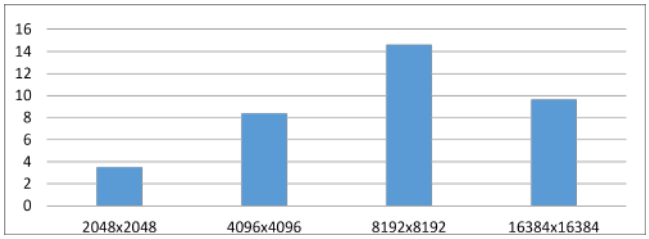

Figure 3: H2RG GPU version speedup over the CPU on NVIDIA Xavier.

any further radiation mitigation techniques. The platform includes ARMv8 Carmel CPUs and a powerful embedded GPU based on the Volta microarchitecture, which are clocked at 1.2 GHz and 520 MHz at its 10W TDP (thermal design power) performance mode.

Figure 3 presents the relative performance of the GPU implementation over the sequential version in the same platform, for various input sizes. The GPU implementation is 3.5× faster than CPU for the smallest size, and up to 15× for larger sizes similar to the requirements of future missions.

As an indication of comparison with existing space processors [Jung and Crouzet 2012], the CPU implementation for the standard 2K×2K image size is 98× faster than the LEON2 at 80 MHz and 15.5× than the PowerPC 750 running at 800MHz, while our GPU implementation is 806× and 128× faster respectively. Such a massive increase in performance of a currently used image resolution shows clearly that embedded GPUs can satisfy the performance requirements of future space missions.

ACKNOWLEDGMENTS

This work is funded by ESA under the GPU4S (GPU for Space) project (ITT AO/1-9010/17/NL/AF). It is also partially supported by the Spanish Ministry of Economy and Competitiveness (MINECO) under grants PID2019-107255GB and FJCI-2017-34095 and the HiPEAC Network of Excellence.

REFERENCES

R. W. Berger, D. Bayles, R. Brown, S. Doyle, A. Kazemzadeh, K. Knowles, D. Moser, J. Rodgers, B. Saari, and D. Stanley. 2001. The RAD750 -A Radiation Hardened PowerPC Processor for High Performance Spaceborne Applications. In *IEEE Aerospace Conference Proceedings*, Vol. 5. 2263–2272. https://doi.org/10.1109/AERO.2001.931184

P. Crouzet. 2011. On-board Processing for the Euclid Mission. *ESA Workshop on Avionics, Data, Control and Software Systems (ADCSS)* (2011).

R. L. Davidson and C. P. Bridges. 2017. GPU-accelerated Multispectral EO Imagery Optimised CCSDS-123 Lossless Compression Implementation. In *2017 IEEE Aerospace Conference*. 1–12. https://doi.org/10.1109/AERO.2017.7943817

M. Hjorth, M. Åberg, N. Wessman, J. Andersson, R. Chevallier, R. Forsyth, R. Weigand, and L. Fossati. 2015. GR740: Rad-Hard Quad-Core LEON4FT System-on-Chip. In *Data Systems in Aerospace Conference (DASIA)*.

A. Jung and P. Crouzet. 2012. *The H2RG Infrared Detector: Introduction and Results of Data Processing on Different Platforms.* Presentation. European Space Agency (ESA). http://www.esa.int/Our_Activities/Space_Engineering_Technology/Onboard_Data_Processing/General_Benchmarking_and_Specific_Algorithms.

L. Kosmidis, J. Lachaize, J. Abella, O. Notebaert, F. J. Cazorla, and D. Steenari. 2019. GPU4S: Embedded GPUs for Space. In *Digital System Design (DSD) Euromicro Conference*.

J. Rosello. 2018. ESA Programs Views, Earth Observation. *12th ESA Workshop on Avionics, Data, Control and Software Systems (ADCSS)* (2018).

N. Tsog, M. Behnam, M. Sjödin, and F. Bruhn. 2018. Intelligent Data Processing Using In-orbit Advanced Algorithms on Heterogeneous System Architecture. In *2018 IEEE Aerospace Conference*. 1–8. https://doi.org/10.1109/AERO.2018.8396536