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

Iván Rodriguez-Ferrández, Leonidas Kosmidis
Ivan.rodriguez@bsc.es, leonidas.kosmidis@bsc.es

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

UNIVERSITAT POLITÈCNICA DE CATALUNYA

Introduction

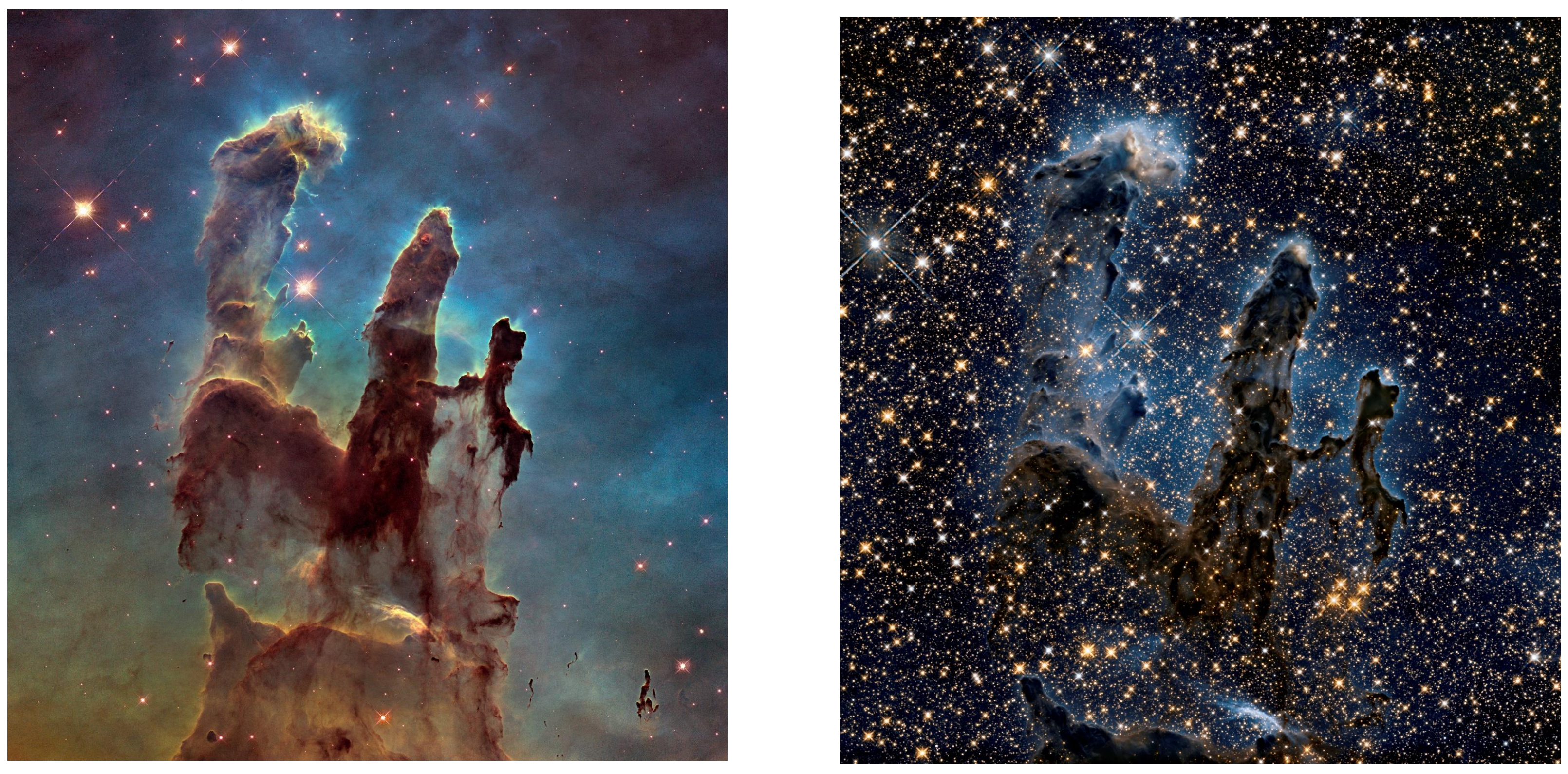
On-board processing requirements of future space missions are increasing.

- New missions types:
  - Space debris recollection
  - Ship to ship docking and refuelling
  - Deep space missions communications due to speed of light limitation

Embedded GPUs can provide the required performance [1] thanks to:

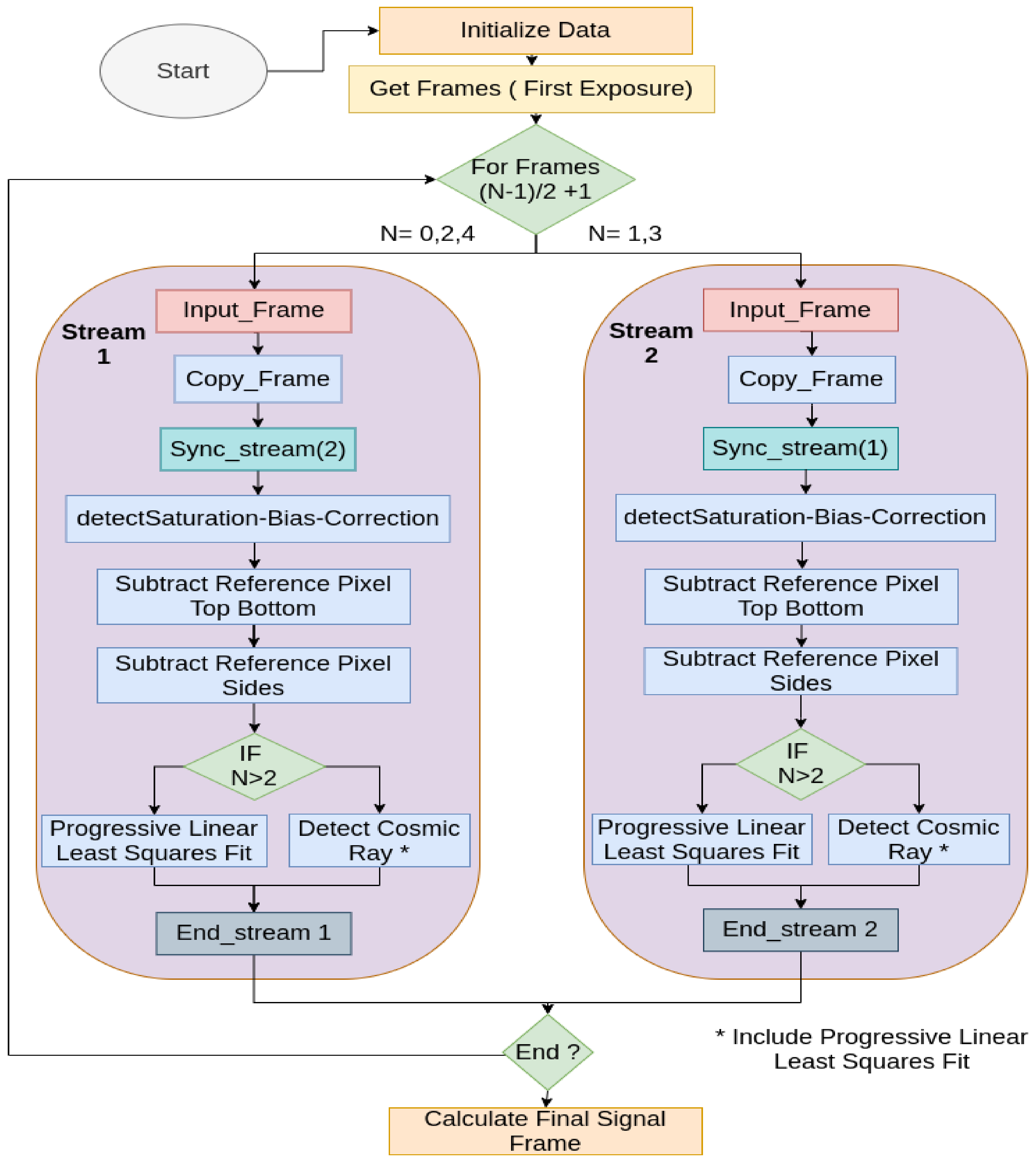
- high performance capabilities
- low-power consumption.

We show a demonstration of a GPU parallelisation of an on-board algorithm[2], within the framework of the GPU4S (GPU for Space) project funded by ESA. This algorithm is oriented in **Near Infrared (NIR)** image processing, which is significantly demanding and its employed by several space telescopes, such as ESA's Euclid, NASA's Hubble and its successor.



Pillars of creation in visible left, and in near infrared right.
Credit: NASA Hubble space telescope imagery

CUDA Implementation



Start
Initialize Data
Get Frames (First Exposure)
For Frames (N-1)/2 +1
N= 0,2,4
N= 1,3
Stream 1
Input_Frame
Copy_Frame
Sync_stream(2)
detectSaturation-Bias-Correction
Subtract Reference Pixel Top Bottom
Subtract Reference Pixel Sides
IF N>2
Progressive Linear Least Squares Fit
Detect Cosmic Ray *
End_stream 1
Stream 2
Input_Frame
Copy_Frame
Sync_stream(1)
detectSaturation-Bias-Correction
Subtract Reference Pixel Top Bottom
Subtract Reference Pixel Sides
IF N>2
Progressive Linear Least Squares Fit
Detect Cosmic Ray *
End_stream 2
End ?
Calculate Final Signal Frame
* Include Progressive Linear Least Squares Fit

For the CUDA version we exploit all possible parallelism:

- Streams
- Overlapping of kernel executions and transfers
- Concurrent kernel executions

For all the code we use manual memory allocation.

Using a configurable number of streams we maximise the application parallelisation. The full application was tuned up for the **NVIDIA Xavier** platform, were the optimal number of streams are two. We can process an image while copying the next one, improving the end to end performance.

Original Code

We base our code in the Euclid NIR sequential version that is available though the ESA website [3]. We chose this code because:

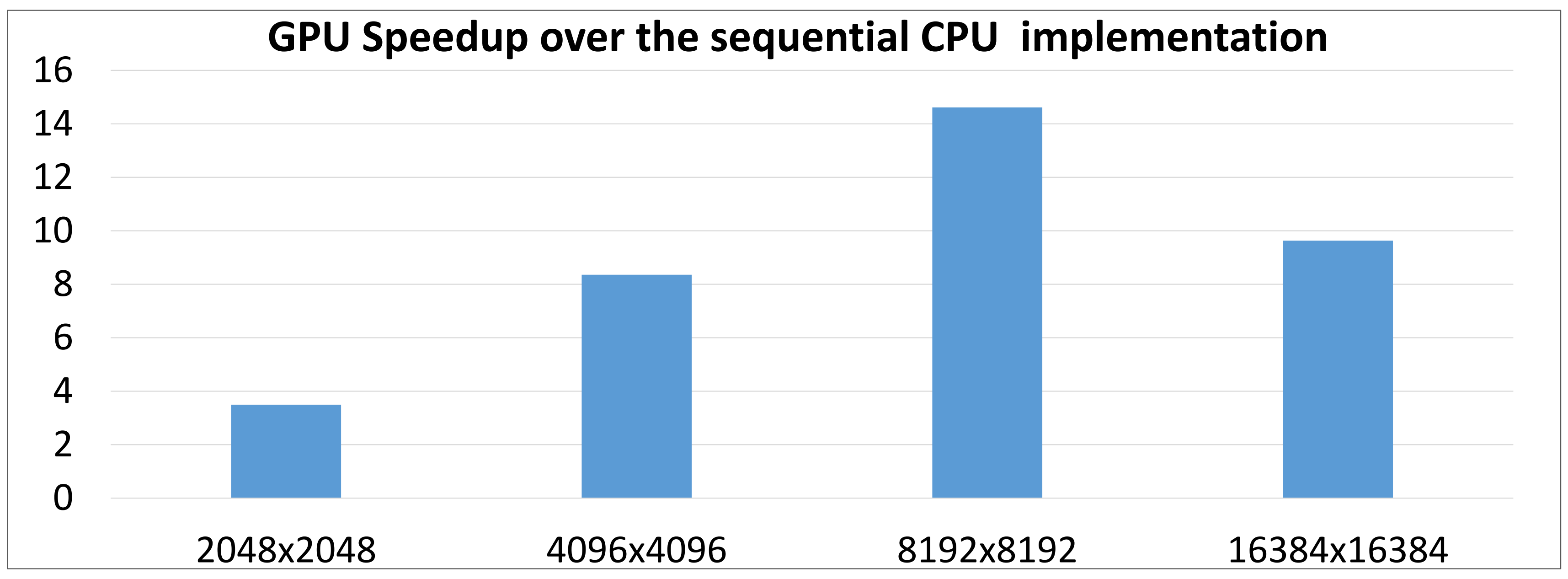
- Lack of industrial demonstrations of space applications using GPUs
  - Only closed source projects
- Real application for space image processing

We made two changes to the original code to become more GPU friendly:

- Removed global variables because the GPU and CPU have different address spaces. Instead, they were converted to kernel parameters.
- Removed the on-line generation of the image to an off-line generation, which is similar to the deployment with a real space image sensor.

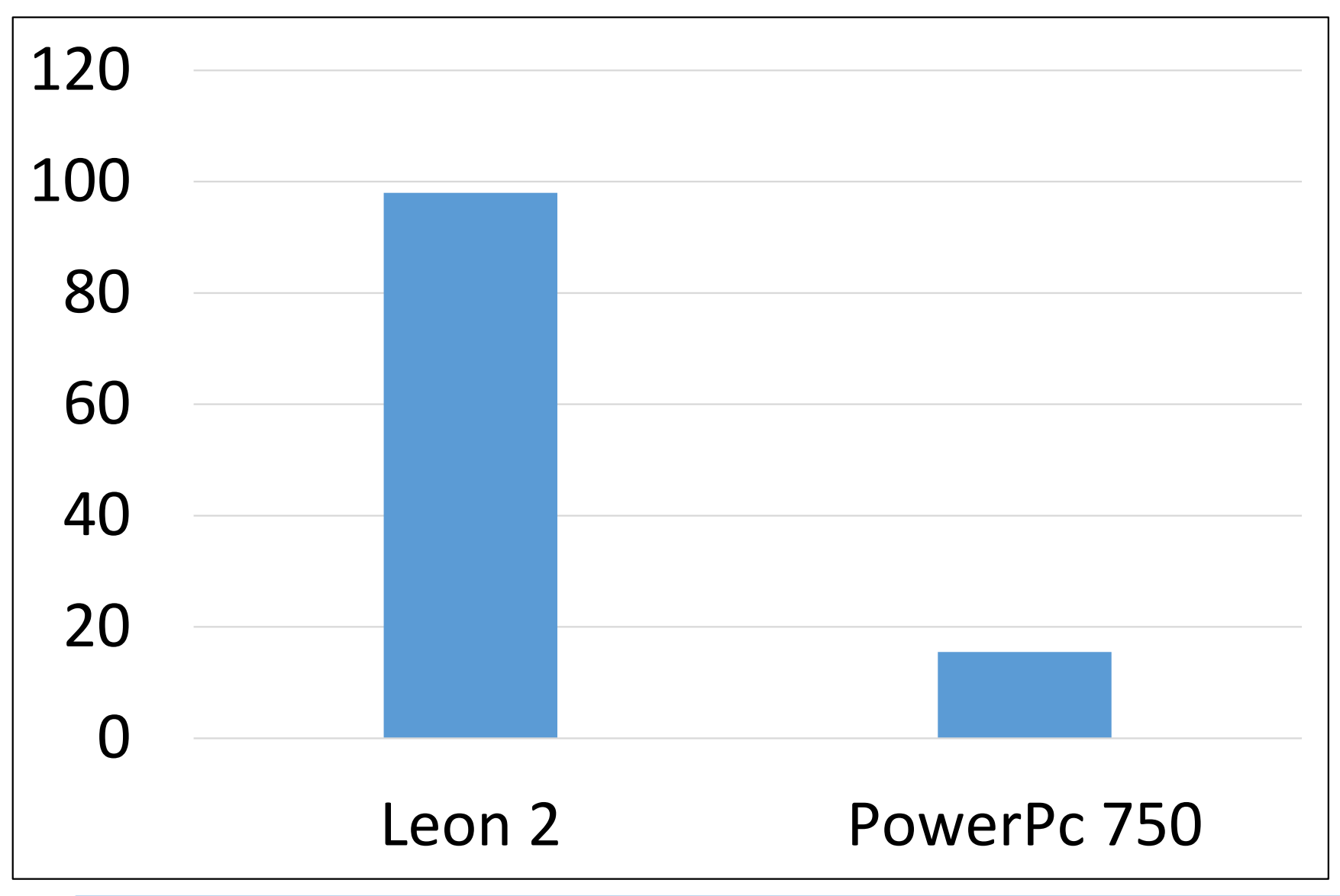
Experimental Results

We execute the application end to end with different input sizes
Performance of CPU version over the GPU on the NVIDIA Xavier:

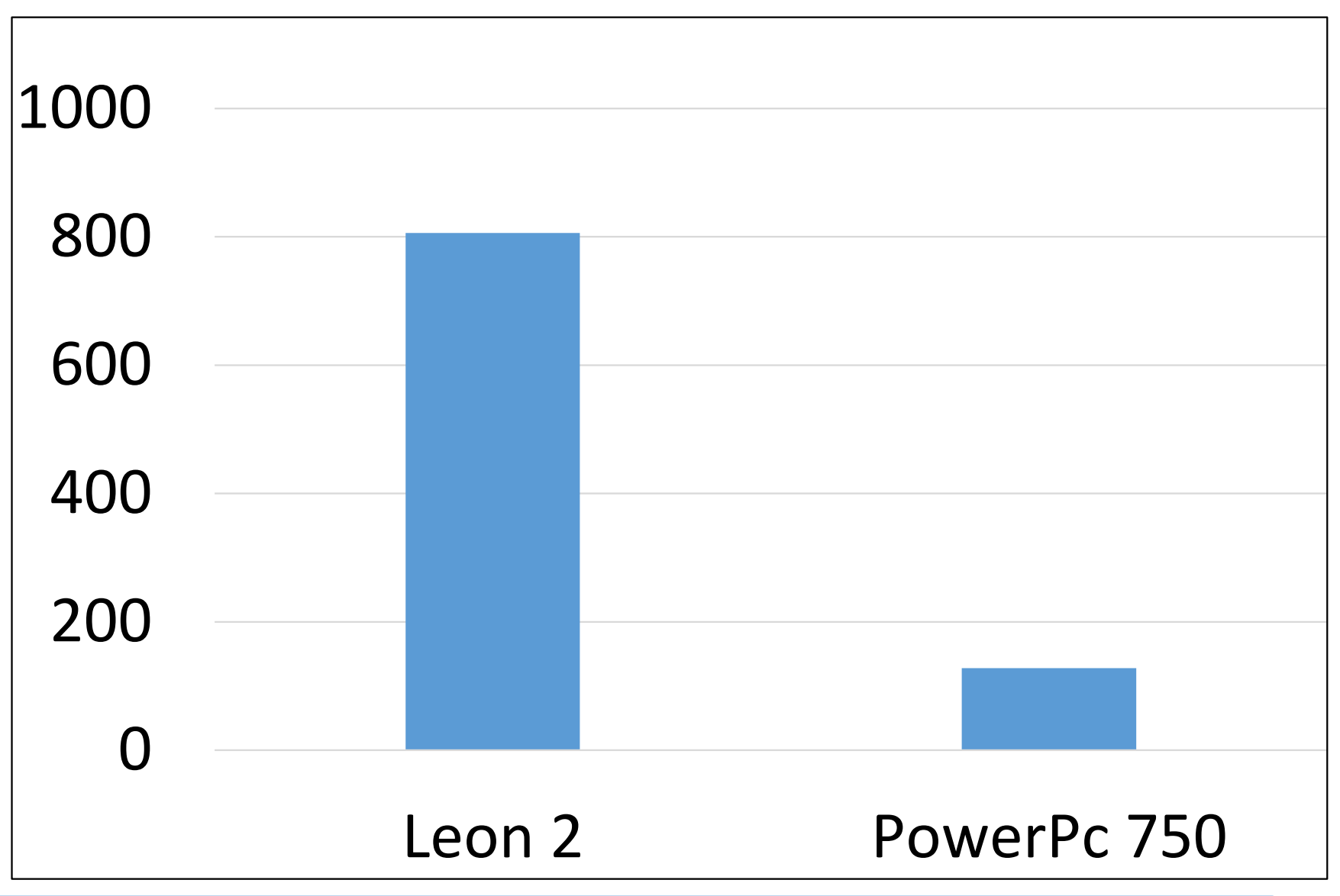


Comparison with state-of-the-art space processors [3] for 2Kx2K:
LEON2 runs at 80 MHz and PowerPC 750 runs at 800MHz

Xavier CPU Speedup over Space CPUs

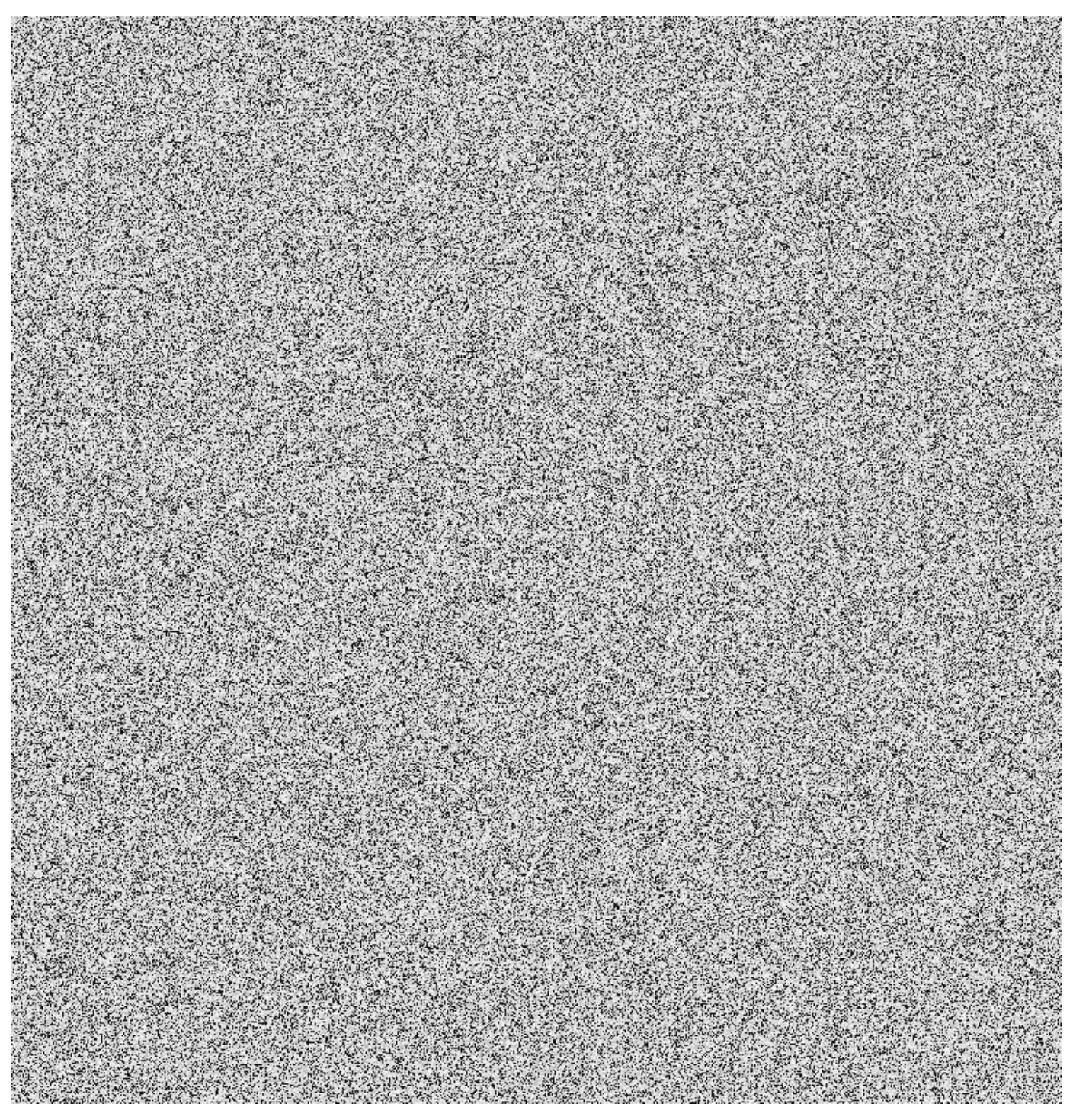


Xavier GPU Speedup over Space CPUs

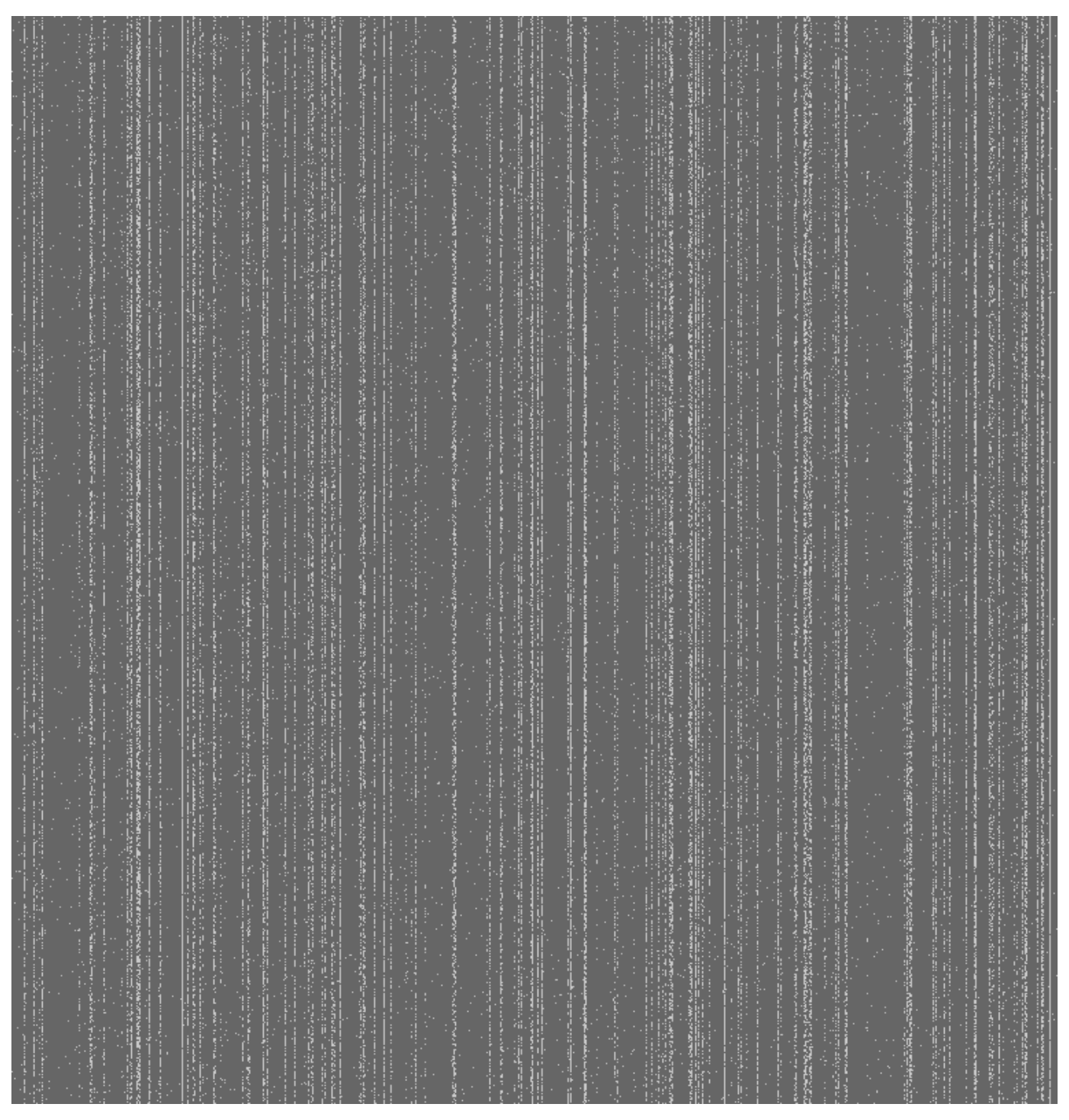


Massive increase in performance of a currently used image resolution shows clearly that embedded GPUs can satisfy the performance requirements of future space missions.

Example of the input and output images



Raw test image from the sensor



Final processed image

Vertical lines represent multiple detected polarized cosmic rays

References

[1] L. Kosmidis et al. GPU4S: Embedded GPUs for Space. In Digital System Design (DSD) Euromicro Conference, 2019.

[2] I.Rodriguez, L. Kosmidis, O. Notebaert, F. J. Cazorla, and D. Steenari. An On-board Algorithm Implementation on an Embedded GPU: A Space Case Study. Design, Automation Test in Europe Conference (DATE), 2020

[3] Andreas Jung and Pierre-Elie Crouzet. The H2RG Infrared Detector: Introduction and Results of Data Processing on Different Platforms. European Space Agency (ESA). 2012
http://www.esa.int/Our_Activities/Space_Engineering_Technology/Onboard_Data_Processing/General_Benchmarking_and_Specific_Algorithms.

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.