Development of the Underground Performance Monitor for LZ

J. F. Silva[1], F. Neves[1], V. N. Solovov[1]
Departamento de Física da Universidade de Coimbra

Abstract

The objective of this project is to develop a data analysis framework based on GPUs (Graphical Processing Units) for real-time monitoring of the performance of the LZ (LUX - ZEPLIN) detector and its subsystems. The GPU framework will be included in the LZ UPM (Underground Performance Monitor). The work will also include the development and implementation of filters and smart decision structures (using Machine Learning) to automatically detect possible performance issues and action the relevant LZ control mechanism. This poster will give a brief overview of the project, present the work already done and the obtained results. In the end, the future steps of the project will be shortly discussed.

I – Introduction

In the last decade, GPUs have been increasingly used as general programming devices. Their intrinsic parallelization capabilities and their high performance/cost ratio make them very interesting devices to improve the performance of already existing systems. In physics, their uses span a wide range of areas, from medical physics [1] to particle physics [2].

LUX-ZEPLIN is part of the next generation of experiments aiming at direct detection of dark matter. Is currently on the commissioning stage and will be the largest detector ever built using a dual-phase detector, housing 10 tons of liquid Xenon (LXe). It is expected to be the most sensitive experiment to date, with it being able to probe WIMP-nucleus cross sections close to an irreducible background called “neutrino floor” [3].

The location chosen for its installation is the Sanford Underground Research Facility (SURF) in South Dakota, USA, 1478 m underground. The complete analysis of the gathered data is done offsite in large clusters of computers. However, these clusters cannot be used for real-time monitoring of the detector because of the time it would take to transfer the data between the detector site and the clusters.

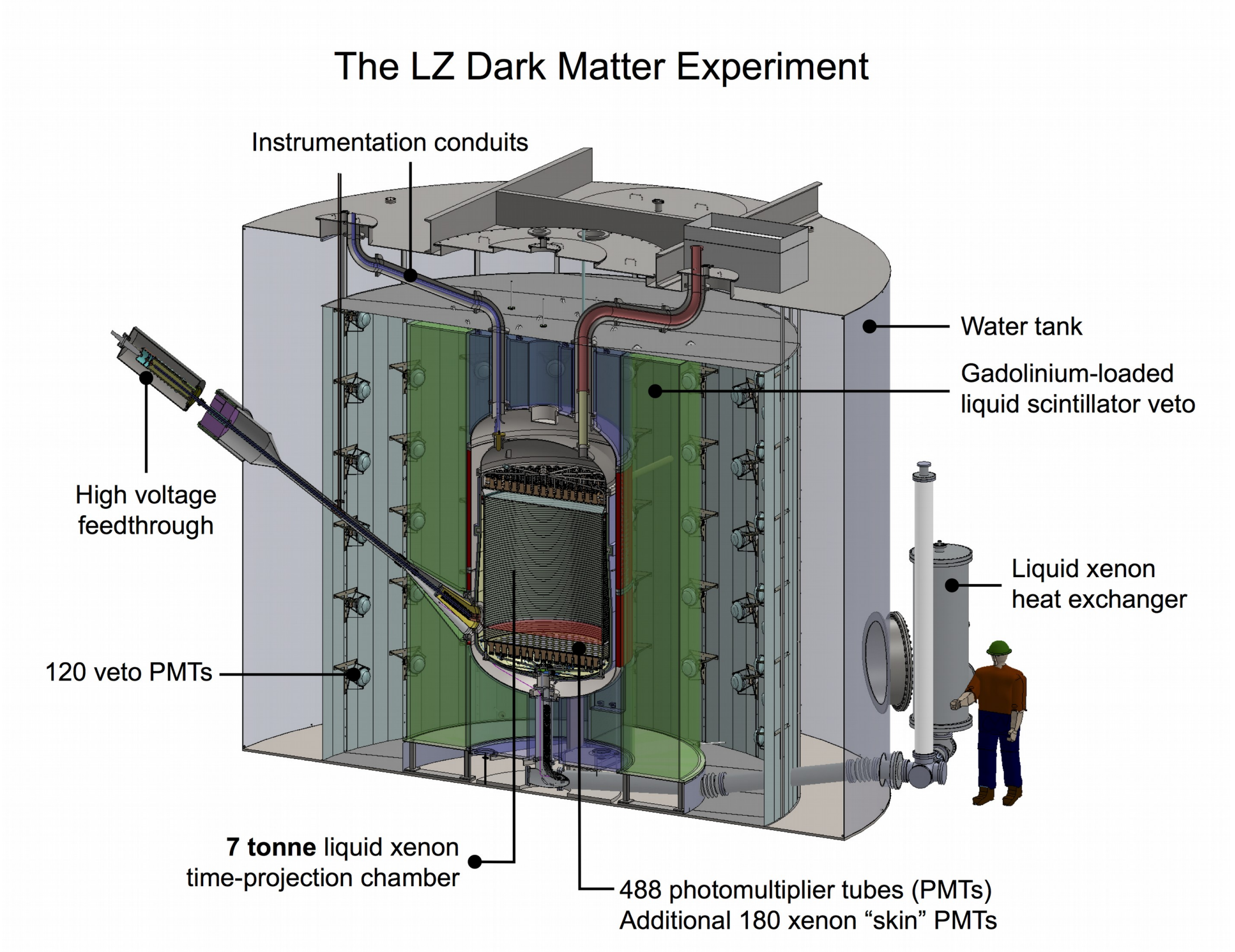

Fig. 1 – Schematic of the LZ detector. LZ Collaboration, “The LUX – ZEPLIN (LZ) Technical Design Report”, 2017.

The solution found for this problem is the UPM (Underground Processing Monitor). This is a LZ subsystem which is responsible for monitoring, in real-time, the detector performance. It is installed onsite and, because of that, is limited to run in a small number of machines. Due to this limitation the computational power provided by the CPUs installed underground is not enough to satisfy the requirements of real-time monitoring of this experiment. So, to improve the performance of the UPM, it was proposed to implement some of the algorithms in GPUs. The “target” algorithms are: calibration, pulse finding, position reconstruction and parametrization.

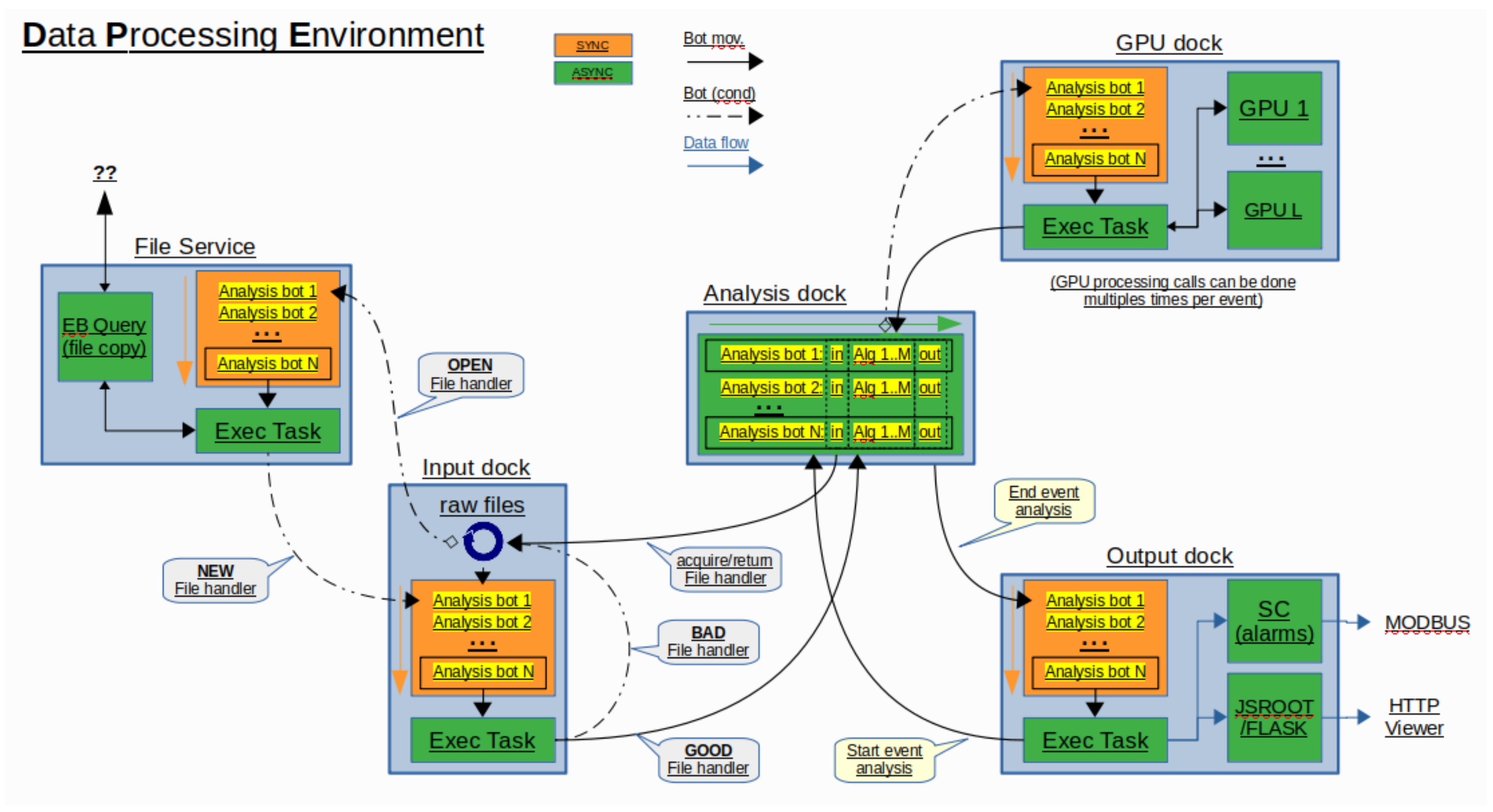

Fig. 2 – Overview of the UPM framework.

[1] Pratx, G. & Xing L., “GPU computing in medical physics: A review”, 2011
[2] Blyth S.C., “Opticks : GPU Optical Photon Simulation for Particle Physics using NVIDIA OptiX”, 2017
[3] LZ collaboration, “LUX-ZEPLIN (LZ) Technical Design Report”, 2017

II – Calibration Algorithm

The first step of the project was to develop and implement the calibration algorithm in GPU. This algorithm can be divided in two phases: pre-processing and calibration. The raw data is read directly form a buffer, to optimize memory usage, and is divided in channels and in PODs (Pulse Only Digitization).

The pre-processing is done asynchronously in the CPU, to not waste GPU queue time. In this part the needed variables for calibration (gain of the channel and zero of each POD) are calculated, and stored in a structure alongside the POD ID information.

Then, this structure and the raw data array are copied to GPU memory. The copy is made only once per event, to reduce memory transfer. The number of SM (streaming multiprocessors) used depends on the size of the raw data array. Each GPU thread launched will calibrate a specified number of elements using (1). For memory optimization, the result of the calibration operation replaces the raw data.

$$\text{raw\_data}[i] = \text{gain}*(\text{zero} - \text{raw\_data}[i]) \qquad (1)$$

Performance benchmark tests were already conducted for this algorithm. Two cuts were applied to the obtained results, one for small events and another for large events. For small events the GPU is generally slower than the CPU, because of the time it takes to transfer data between host and device. But, CPU times are subject to often and large fluctuations due to processor occupancy with unrelated tasks.

For large events, the GPU is always faster than the CPU, achieving speed up factors of 1,2x to 1,8x.

A clear event size threshold can be defined at 100000, but it is still advantageous to process the smaller events on the GPU because of it having less time fluctuations, and to ease CPU workload.

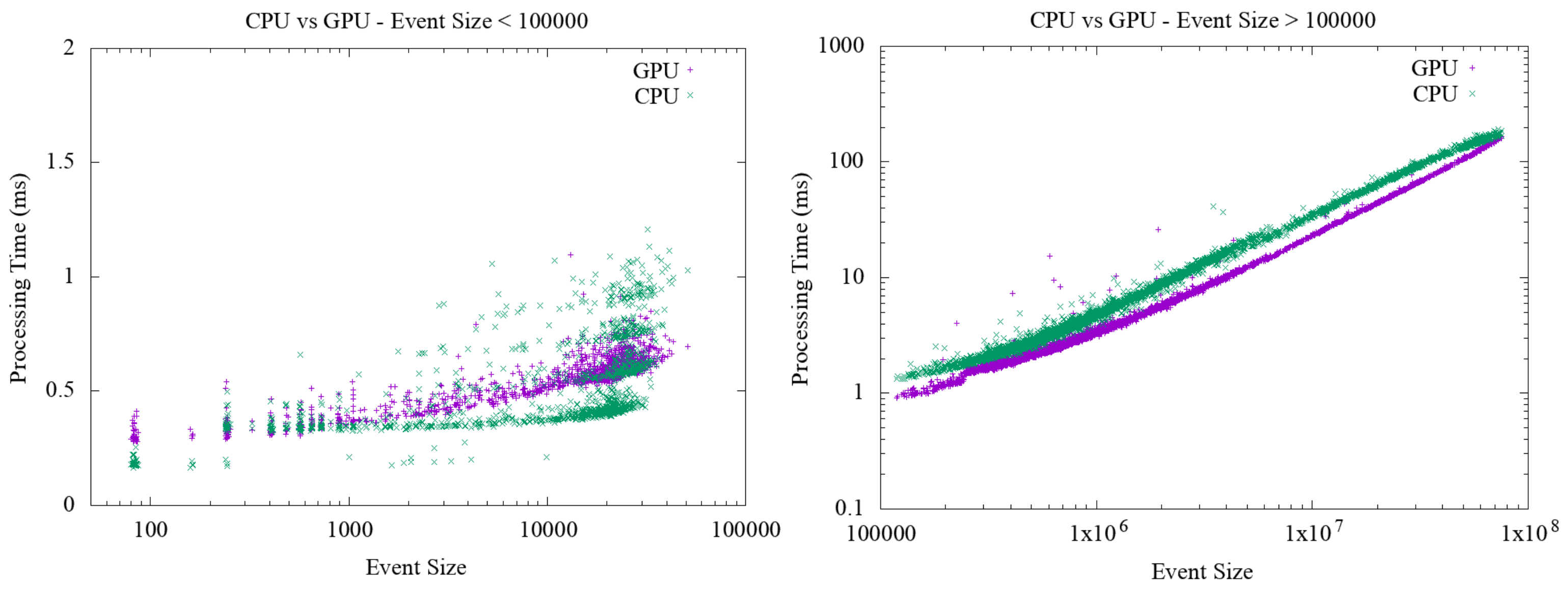

Fig. 3 – Comparison of the calibration processing times between CPU (green) and GPU (purple). On the left side are represented the processing times for small events (event sizes < 100000). On the right side are represented the processing times for large events (event sizes > 100000).

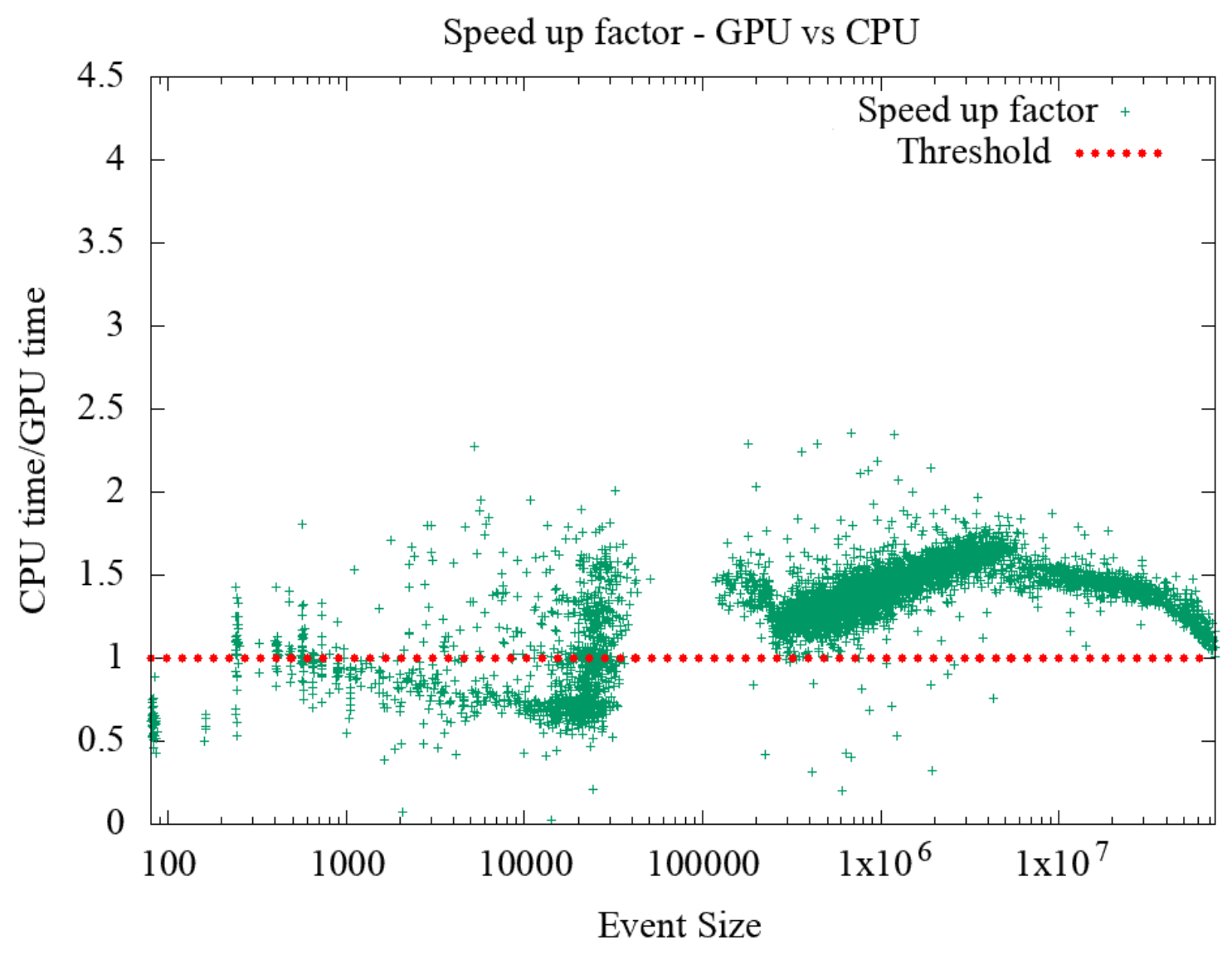

Fig. 4 – Plot of the ratio between CPU processing time and GPU processing time. The values above the red line have a speed up factor larger then 1x, and run faster on the GPU.

III – Future Work

The next step of the project is the development and implementation of the position reconstruction algorithm. This algorithm will locate the position, on the PMT array of the detector, where the signal was produced. It will be based on the ANTS2 algorithm already developed by LIP – Coimbra. From the conducted tests, this is one of the biggest speed bottlenecks of the UPM and is, therefore, critical to be accelerated by GPUs.

Further ahead, the plan is to develop the pulse finding and parametrization algorithms. The final part of the project is developing smart decision structures, using machine learning, to automatically identify possible performance issues in the detector.

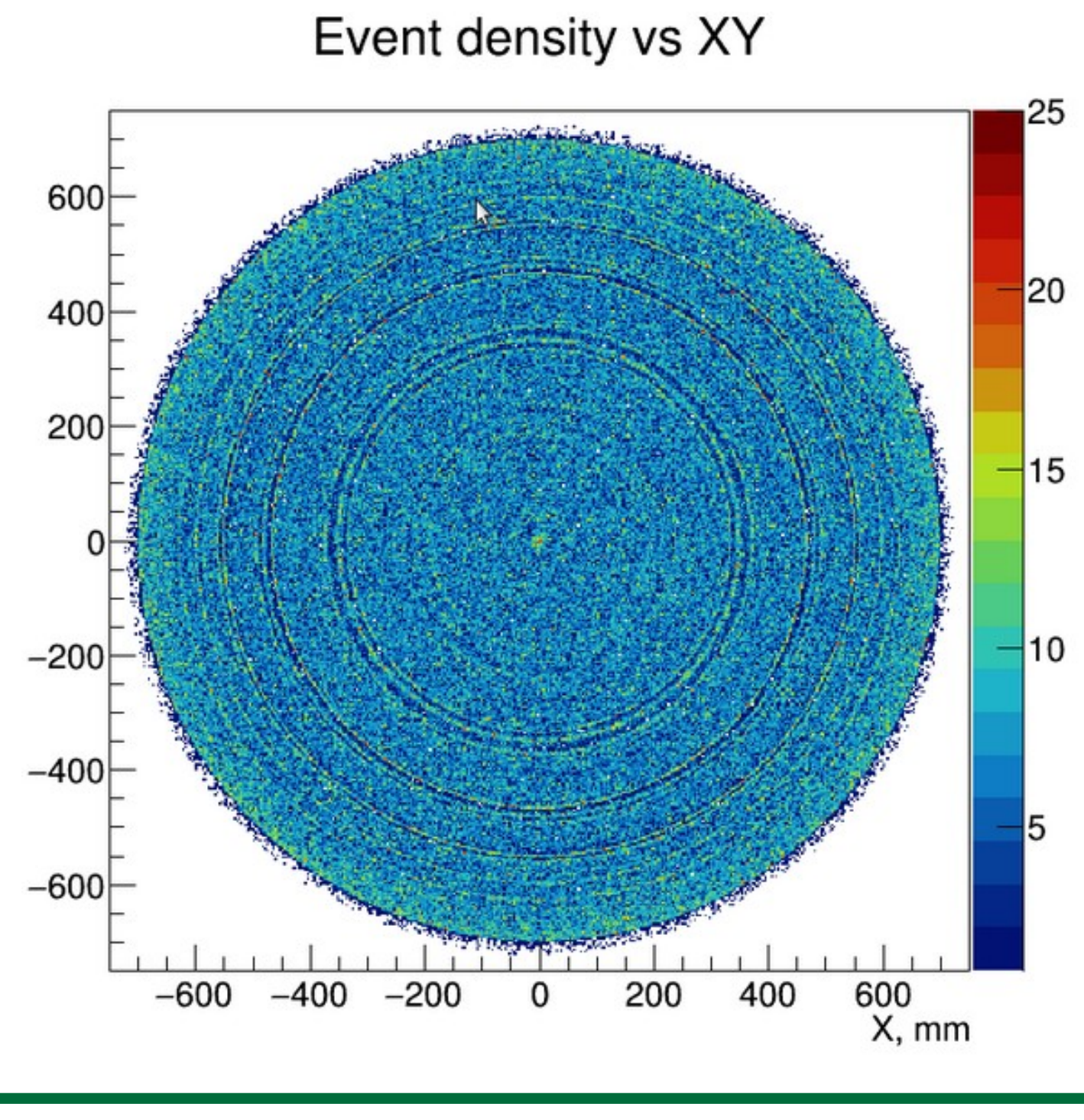

Fig. 5 – Plot of the event density as a function of its position. The data analysis was performed with CPUs.

---

[1] LIP – Coimbra

Masters project of J. F. Silva: joao.marques.silva@coimbra.lip.pt