Journal of Computational Science 88 (2025) 102605

Contents lists available at ScienceDirect

Journal of Computational Science

journal homepage: www.elsevier.com/locate/jocs

Edge computing driven forest fire spread simulation: An energy-aware study

Carlos Carrillo *, Tomàs Margalef, Antonio Espinosa, Ana Cortés

Computer Architecture & Operating Systems department, Universitat Autònoma de Barcelona, Barcelona, Spain

ARTICLE INFO

Keywords:
Embedded systems
Low-consumption GPUs
Energy-efficient computing
Wildfire simulation

ABSTRACT

An accurate and fast prediction of forest fire evolution is a crucial issue to minimize its impact. One of the challenges facing forest fire spread simulators is the uncertainty surrounding the input data. While high-performance computing (HPC) platforms help reduce these uncertainties, their accessibility during emergencies is limited due to infrastructure constraints. real time data collection using sensors onboard Unmanned Aerial Vehicles (UAVs) in real time can significantly reduce their uncertainty. However, transmitting this data to HPC environments and returning the results to firefighters remains difficult, especially in areas with poor connectivity. We propose using Edge Computing to address these challenges, leveraging low-consumption GPU-accelerated embedded systems for *in situ* data processing and wildfire spread simulation. For simulation purposes, the FARSITE forest fire spread simulator has been used. This work aims to demonstrate the feasibility of leveraging Embedded Systems with low-consumption GPUs to simulate *short-term* forest fire spread evolution (up to 5 hours) at high resolution (5 meters). The obtained results highlight that these devices can gather data, simulate the hazard, and deliver prediction results *in situ*, even without connectivity, opening up the possibility of monitoring and predicting fire behavior at high resolution without employing HPC platforms.

(This paper is an extension version of the best poster paper award in ICCS-2024 entitled "From HPC to Edge Computing: A new Paradigm in Forest Fire Spread Simulation".)

1. Introduction

In recent years, climate change has increased forest fire risk levels across the world. The increase in the number and intensity of wildfires has highlighted the urgent need to acquire highly accurate tools to analyze the behavior of this kind of natural hazard. Their potential to guide the management of firefighting procedures depends on two relevant characteristics: simulation results must be very precise, and once given the initial conditions and parameters, these results must be obtained as quickly as possible. In summary, tools must be precise and fast. To achieve that, different forest fire spread simulators have been developed, with the objective of helping the fight against them. In most cases, fire simulation analyses usually depend on highly specialized computational platforms with a large number of resources available and with a high bandwidth interconnection system. Nevertheless, the need for these platforms is difficult to schedule as their use is usually requested on-demand.

The uncertainty of the input data required by wildfire simulators could be a relevant drawback to accurately predicting the forest fire evolution in a short period. Different strategies have been used to reduce data uncertainty by adjusting the input data values to better reproduce the past behavior of real wildfires. However, these studies rely on High Performance Computing (HPC) platforms, which usually need specialized computational resources to execute prediction frameworks [1–3].

However, it is challenging to use HPC resources for in-situ fire management. Teams should have their own systems dedicated to handling the simulation and attending to the infrastructure costs of operation or adapt their needs to the use of on-demand resources from cloud computing providers. One solution to reduce the need for provisioned computational resources is to obtain the input data and solve some pre-processing tasks at the emergency site. Several studies highlight that the application of Unmanned Aerial Vehicles (UAVs) to collect local real time data, such as meteorological parameters, vegetation status, or fire evolution in the same area where the fire is taking place, can significantly reduce input data uncertainty [4–6]. However, using this type of vehicle raises another crucial issue: the necessity to communicate all the relevant data parameters from emergency locations, usually far from urban areas where computational platforms are within reach. The main barrier to managing these scenarios is the severe communication limitations of the forest regions where the fire is taking place. Under

* Corresponding author.
E-mail address: carles.carrillo@uab.cat (C. Carrillo).

https://doi.org/10.1016/j.jocs.2025.102605
Received 15 October 2024; Received in revised form 12 March 2025; Accepted 21 April 2025
Available online 2 May 2025

these circumstances, it would be difficult to quickly incorporate on-demand locally collected data into traditional computational systems such as HPC or cloud computing commercial systems for the evaluation of the forest fire spread.

Our proposal studies the allocation of temporal computational resources in the field to alleviate the need to use remote resources. First of all, deploying physical HPC infrastructures in active fire zones becomes a difficult task. However, it is easier to make use of edge computing resources that can be rapidly deployed in safe areas near the fire zone. These units can process a relevant part of the data collected from the field without the need of complex infrastructure, allowing for localized computations. If we can bring enough easy to deploy on-the-field computational resources, we could increase the efficiency of the data gathering processes and be able to run the wildfire spread simulator to provide high-quality results to the fire management team. These two characteristics: *in situ* data gathering and effective simulations results, could be done using UAVs with low energy consumption computational resources and a set of embedded sensors on board. The resulting embedded system could use the edge computing paradigm, distributing the forecasting system into some lightweight systems with computing capabilities that could provide a more realistic view of the input data parameters and react to the dynamic conditions of the situation.

The current fast-growing capabilities of embedded systems make them good target platforms for edge devices. One can find in the literature several works proposing the use of swarms of UAVs for forest fire early detection or spread monitoring purposes. However, none of them is oriented to perform forest fire spread predictions *in situ* [7–9]. Other works propose prediction systems on board a single UAV using infrared cameras. These approaches are focused on evaluating the maximum rate of spread of a given forest fire without considering its whole perimeter evolution [10]. Works that consider the complete fire perimeter could use data collected using UAVs, but they actually rely on cloud computing services to calculate the prediction [11]. The utilization of a cloud computing platform notably reduces the costs and the time to obtain results, but the connection problem remains in those zones with low connectivity.

Furthermore, computational models used by simulation tools have to deal with dynamic complex scenarios: terrain, wind, and changing weather conditions that increase the need to understand the evolution of the fire with greater detail to obtain reliable prediction results. That is, good prediction results depend on the sensitivity of the fire spread model to adapt to small-scale variations in spatial variables. Therefore, the computational platform should perform high-resolution simulations in near real time, taking advantage of the data recently collected and having the capacity to operate in zones of difficult access and low connectivity.

Newly available computational resources, such as GPU accelerators, are not standard and require adaptations to simulation tools in order to utilize their computational capacity effectively. Several studies demonstrate that computational accelerators like GPUs can significantly enhance the performance of simulators in terms of execution time without losing the accuracy of the results for a given data resolution [12–17].

In this work, we propose a viable encapsulation of a low energy GPU-accelerated computational system that is able to deliver *in situ* high-resolution forest fire spread prediction in a few minutes. The main objective is to demonstrate that it is possible to obtain operative simulations using a low-consumption GPU-based system that can be used in those zones with low connectivity in near real time. The obtained results demonstrate the viability of using low-consumption embedded systems to provide an *in situ* snapshot of the potential for a wildfire to become an extreme event in areas with reduced or even non-existent connectivity.

This work is organized as follows. Section 2 details the characteristics of the forest fire simulator used, the metrics, and the scenarios used in performance analysis of accelerated computational platforms.

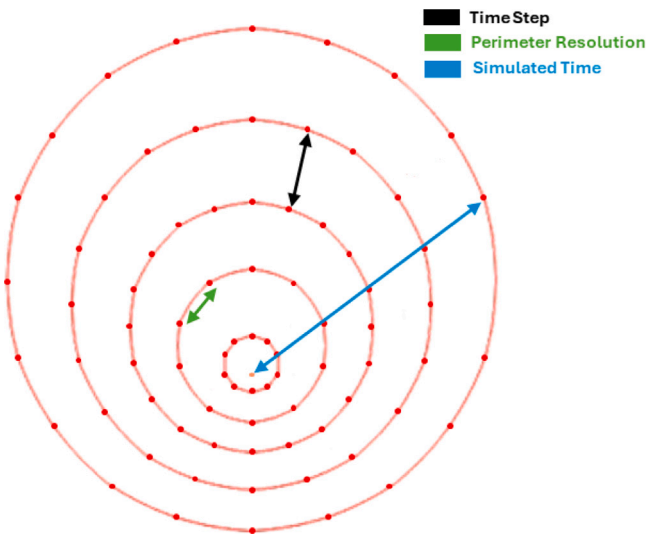


Fig. 1. Graphic representation of time step (black), Perimeter Resolution (green) and Simulated Time (blue).

Section 3 reports the experimental study conducted to demonstrate the feasibility of using edge computing during an ongoing wildfire. And finally, Section 4 summarizes the main conclusions and future steps of this work.

2. Methodology

2.1. FARSITE

To carry out this study, we focus on the *Fire Area Simulator* (FARSITE) forest fire simulator [18], which has extensively been tested on real fires to produce successful results. FARSITE has been used for several years for forest fire evolution simulation, and it has become widely used for operational prediction of fire growth of active wildfires as well as for modeling in support of planning for potential fires.

FARSITE has three different parameters that have a direct impact on the time spent to perform a simulation: The *Simulated Time*, the *Time Step* and *Perimeter Resolution (PR)*, described in Fig. 1. The *Simulated Time* determines the propagation time of the real fire to be simulated. The *Time Step* is the maximum amount of time that the conditions at a given point are assumed constant so that the position of the firefront can be projected. For example, a 5 min Time Step to evaluate an objective scenario of 3 h simulated evolution of the fire. The positions of all fires will be projected over this time step so that possible mergers can be computed. The perimeter resolution determines the maximum distance between points used to define the fire perimeter. It is a resolution of a firefront in the direction tangential to the fire perimeter at each point. The perimeter resolution controls the detail of the firefront, both in curvature and in the ability to respond to heterogeneous situations occurring at a fine scale. A low **Perimeter Resolution** is necessary to make a spreading fire sensitive to small-scale variations in spatial variables, [19]. It determines the number of points of the simulated firefront. The smaller it is, the more points become available for determining the heterogeneity of the terrain, and the fire behavior can be determined in more detail.

The analysis of fire expansion situations needs the introduction of new perimeter points to maintain the accuracy of the simulation. In the case of FARSITE, Perimeter Resolution must be maintained in any new perimeter generated. However, more perimeter points to analyze means a longer execution time as the complexity of FARSITE is $O(n^2)$, where n is the number of points.

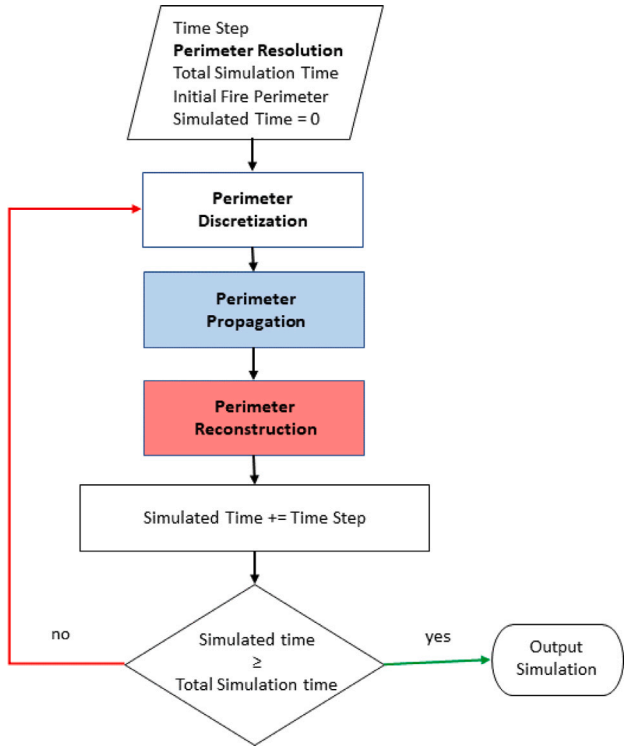


Fig. 2. Basic workflow of FARSITE in discretization, propagation and reconstruction stages.

Fig. 2 shows a simple FARSITE workflow starting by initializing a set of variables, and then it continuously iterates through three consecutive blocks: *Perimeter Discretization*, *Perimeter Propagation* and *Perimeter Reconstruction*. This iterative scheme starts by dividing the perimeter of the forest fire into a set of points (*Perimeter Discretization* block). As we said, the Perimeter Resolution (PR) determines the number of perimeter points. The value of PR is defined as a parameter at the beginning of the simulation process. Consequently, at each iteration, the number of points of the perimeter will increase as the forest fire evolves. This increment is due to the need to interpolate new points each time the *Perimeter Discretization* block is executed to keep the distance between perimeter points limited by PR.

The *Perimeter Propagation* block is responsible for obtaining the evolution of the firefront. In this block, each point of the perimeter is propagated in a continuous space and independently of the rest of the points that form the perimeter. More precisely, an ellipse is generated for each point based on Huygens's principle [20]. The local characteristics at each point determine the shape of those ellipses by using Rothermel's fire behavior model [21]. The Rothermel's model is formulated in the following way:

$$R = R_0 \cdot (\vec{n} + \vec{\phi}_w + \vec{\phi}_s) \tag{1}$$

where R_0 represents the rate of spread in a particular point with no wind and no slope. This point-to-point semi-empirical equation is the basis for expanding the perimeter using Elliptical Wave Propagation. In this approach, the perimeter of the fire is divided into a series of points, [20]. An ellipse is generated for each point to obtain the evolution of the fire perimeter. The shape of the ellipses is determined by the local characteristics at each point. In this way, the new perimeter is obtained by joining the obtained ellipses.

The *Perimeter Reconstruction* block is in charge of obtaining a new perimeter by joining all ellipses obtained in the previous block. The new perimeter is obtained by joining all ellipses obtained in the previous block. Then, the total *Simulated Time* is incremented by the *Time Step* value, which corresponds to the simulation time assigned to each iteration of the simulation process. Finally, if the pre-defined *Simulated time* has not been reached, the entire process starts by rediscretizing the fire perimeter again. If the pre-defined total *Simulation time* has not been reached, the entire process starts by rediscretizing the fire perimeter again.

One relevant characteristic of FARSITE is that the propagation of each perimeter point is completely independent of the propagation of the other points, *ergo*; the propagation of all points can be done in parallel. As GPUs are massive throughput-oriented computational systems, they are a good target platform for the treatment of problems with large amounts of data. In the case of high-resolution forest fire spread propagation, the number of points to consider can be on the order of hundreds of thousands, becoming a large list of independent small problems that are well suited to this type of computing device. To adapt FARSITE modules to use GPU accelerators, the main *Perimeter Propagation* and *Perimeter Reconstruction* blocks have been redesigned in order to exploit maximum usage of computational resources. A fine-grained task-parallel approach has been used where the propagation of points and the reconstruction of the firefront are calculated in parallel to obtain a new perimeter.

2.2. Execution platforms

In a real situation, the UAVs can be used to collect real time data ahead of the fire front. These vehicles require large autonomy to operate for prolonged periods, monitoring and reporting changes in fire behavior. A low power consumption embedded system can be installed into such vehicles to provide a source of computational resources under hard energy consumption restrictions. The integrated system can operate minimizing the effects on the autonomy of the vehicle and allowing the processing of the acquired data.

In some extreme cases, the main constrain is the availability of energy to allow the vehicle operation in hard to access locations. In these situations, it is important to consider the trade-offs: whether to conduct data analysis tasks with high power usage to get results as fast as possible or to reduce the energy needs of the computer system and have longer execution times.

The NVIDIA Jetson AGX Xavier developer kit, our *edge-platform* analyzed, has been specially designed as an AI computer for autonomous machines, like robots, drones, and other autonomous systems, delivering the performance of a GPU workstation in an embedded module under 30 W [22]. In contrast, the *workstation* is designed for computationally intensive tasks with no limitations of power and resources, such as high-performance computing, graphics rendering, and AI.

In this study, we focus on testing the possibility of simulating a real forest fire in a low-energy computational system that is located close to the location of the fire in an on-demand way. Our objective is to show the main challenges, the potential benefits, and the expected trade-offs of the cases presented. For that purpose, we focus on analyzing the capability of a low-consumption GPU-accelerated embedded system to match the requirements of execution time and accuracy for forest fire spread simulations during an ongoing real case in the place where the fire is occurring. The experimental study carried out in this work first validates the use of a low-energy GPU-accelerated system for embedded applications like high-resolution sensors or robotic systems: NVIDIA Jetson AGX Xavier developer kit [22] (*edge-platform*), versus a standard workstation with a discrete GPU system, GeForce RTX 2080 Ti (*workstation*).

Table 1 presents the main characteristics of the GPU accelerators of the studied system as the manufacturer declares them. Our first objective is to compare both platforms using three relevant characteristics that affect the platform performance: computational capabilities, memory bandwidth, and power consumption. First of all, the *edge-platform* has 512 CUDA cores, which offer parallel processing power but on a smaller scale compared to the *workstation*, which provides 8 times

Table 1
Hardware specifications of the experimentation platforms.

	Jetson AGX Xavier	GeForce RTX 2080 Ti
CPU	ARM v8.2 64-bit	Intel(R) Core(TM) i9-9900K
CPU cores	8	8
GPU architecture	Volta	Turing
CUDA cores	512	4352
Frequency (Mhz)	670	1545
Bandwidth (GB/s)	137	616
TDP (W)	15 or 30	250

more resources in terms of 4352 CUDA cores. This significant difference in CUDA cores highlights the *workstation* superior performance in handling.

Another important specification that impacts their performance is the memory bandwidth, which indicates the rate at which data can be transferred between the GPU and its memory. The *edge-platform* has a bandwidth of 137 GB/s, which is suitable for moderate workloads. On the other hand, the *workstation* provides a significantly 5.5 higher bandwidth of 616 GB/s. Given these differences, it is reasonable to anticipate that the *workstation* will significantly outperform the *edge-platform* for tasks that fully leverage the GPU.

Finally, a critical difference between the two platforms is the TDP (Thermal Design Power). The TDP illustrates their power consumption and heat generation. The *edge-platform* has a configurable TDP of 15 W or 30 W. In contrast, the *workstation* consumes 250 W, which means it consumes about 8 times more than the *edge-platform*.

2.3. Metrics

In this section, we describe the metrics used to compare the performance in terms of execution time and energy consumption for both systems to study the viability of the *edge-platform* in real emergency situations. In this work, we propose the use of some logical and physical metrics to compare both platforms. Our proposal is to have a view of the work done, the time needed to do the work, the resources used, and the energy consumed to evaluate the systems and their performance in different scenarios.

Execution Time: It provides the total time executing a specific forest fire spread simulation. This parameter is measured in seconds (s).

Thermal Design Power (TDP): It is defined by the manufacturers of the corresponding computing device, and it is measured in watts (W).

Energy Capability (EC): It is the average number of computed points per work unit by a given computing device X (see Eq. (2)). By work, we understand the multiplication between ET (time) and TDP (power); therefore, EC evaluates the amount of points to process by power unit. Thus, the higher this index is, the more energy efficient device X will be.

$$EC(X) = \left(\frac{\#Points}{Execution\ Time \cdot TDP} \right) \tag{2}$$

Green-up: This metric aims to compare the energy efficiency of two computing devices that will be referred to as device X and device Y. This metric helps us determine in which situation the reduction in execution time does not offset the increase in energy consumption, [23]. In particular, this metric is defined as the ratio between EC of device X and EC of device Y (Eq. (3)).

$$Green-up(X,Y) = \frac{Energy\ Capability(X)}{Energy\ Capability(Y)} \tag{3}$$

Green-up is analogous to *speed-up* as it measures the relative performance, in terms of energy consumption, of two computing devices when processing the same tasks. Suppose the *Green-up* is higher than the corresponding *speed-up*, it means that the increment of the energy consumption is not compensated by the acceleration of running the task. On the contrary, if the *speed-up* is higher than the *Green-up*, it indicates that the increment of the energy consumption is compensated by the reduction of the execution time.

Table 2
Execution time in seconds (s) spent for both systems to perform the simulations with PR equal to 100, 50, 25, 10 and 5 m for four different *Simulated Times* of 5, 10, 24 and 48 h.

PR (m)	5 h		10 h		24 h		48 h	
	e	*w*	*e*	*w*	*e*	*w*	*e*	*w*
100	0.528	0.826	0.629	0.854	1.062	0.941	1.883	1.066
50	0.531	0.843	0.649	0.856	1.229	0.959	1.265	1.205
25	0.546	0.827	0.729	0.860	1.736	1.062	23.564	2.555
10	0.749	0.881	1.608	1.027	8.766	1.936	80.021	12.860
5	1.645	1.050	5.600	1.572	42.938	4.927	121.389	29.392

3. Experimental study and results

To determine if the edge computing paradigm can successfully be used to simulate the evolution of a forest fire close to where it is burning, the system must accomplish three requirements: the accuracy of the simulation cannot be affected, results must be available within an acceptable execution time (not beyond a certain threshold), and energy costs must be under some restricted limits.

Two different scenarios have been considered to validate *edge-platform* for fire simulation analysis in emergency scenarios. The first scenario called *Ideal case*, is designed to determine if the *edge-platform* is capable of producing a useful results in terms of simulation quality, execution time, and energy consumption. On the other hand, the second scenario, which will be referred as *Real case*, is focused on determining which parameters of the simulation process affect quality, time, and energy that must be tuned in *edge-platform* to obtain the best results in a real wildfire scenario.

3.1. Ideal case: validation of edge proposal

In this section, a synthetic ideal forest fire that burns on a flat terrain with homogeneous fuel and constant wind speed and wind direction is used. Under this controlled environment, the evolution of the fire is done in concentric ellipses [24]. The performance of *edge-platform* and *workstation* have been compared using various *Perimeter Resolution* values and different *Simulated Times*. In particular, since we are interested in delivering *in situ* predictions of the near future fire evolution to aid forest fire decision-makers in determining whether an ongoing event could become an extreme wildfire, four different *Simulated Time* windows have been used: 5 h (*Short-term* simulations), 10 (*Mid-term* simulations) and 24 h and 48 h (*Long-term* simulations). Furthermore, to analyze the impact of increasing the resolution of the simulation and, therefore, its complexity in the execution and into the energy consumption, the *Perimeter Resolution* was set to 100, 50, 25, 10, and 5 m. Five repetitions were carried out for each particular simulation. The times presented in this experimental study correspond to the average of the five times obtained when executing each simulation configuration, obtaining values with a maximum standard deviation under 0.05.

In terms of quality, *edge-platform* and *workstation* are equivalent, as the fire evolution simulations produce identical results on both systems. This is significant because it highlights that the choice of using a low-consumption platform like *edge-platform* does not compromise the accuracy of the simulations, ensuring that energy efficiency does not come at the cost of precision.

Table 2 summarizes the execution times for the simulations performed using both systems. The obtained results show that in *Long-term* simulations with PR equal to 10 and 5 m, *workstation* is 8.71 and 9.22 times faster than *edge-platform*, respectively. These results align with what one would expect because the *Long-term* scenarios are the most computationally intensive due to the large amount of work required to simulate the evolution of the firefront. Those scenarios with low PR and high *Simulated Time* are hard in terms of computational costs and show in which situations the *workstation* can take advantage of its

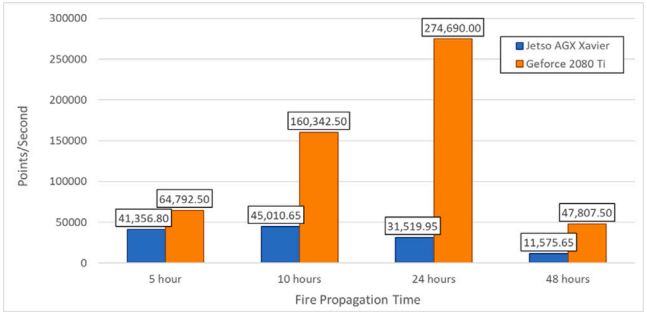


Fig. 3. Number of points processed for both GPUs when using PR equal to 5 m and for different simulation time windows.

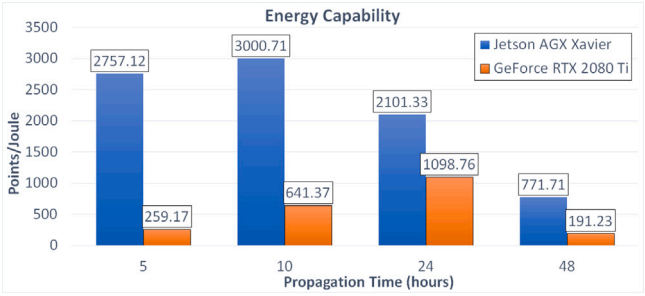


Fig. 4. *Energy Capability* of the forest fire spread simulations for both GPUs when using PR equal to 5 m and for different simulation time windows.

higher computational power. The superior performance of the *workstation* is due to its 8.5 times greater number of CUDA cores compared to the *edge-platform*, which allows it to maximize its computational capacity in these computationally demanding situations. Nonetheless, for the rest of the scenarios, *edge-platform* is not always 8.5 times slower than *workstation*. In the case of *Mid-term* simulations with very high resolution, PR equal to 5 m, the *workstation* is only 3.6 faster than the *edge-platform*.

Additionally, for most *Short-term* scenarios, the *edge-platform* is faster. For large GPU systems, some initialization time is required to copy the initial data sets to GPU memory and create the execution environment. This overhead becomes relevant when dealing with short-term simulations. A key reason for not achieving better performance on *workstation* is the limited number of parallel tasks that cannot hide the initialization and the communication time, contributing to the overall execution time [25]. As problem size increases, this overhead becomes negligible compared to the total execution time.

When conducting high-resolution simulations, the computational demands often exceed the capabilities of standard CPUs, leading to significantly worse performance. On the contrary, GPU accelerators offer added substantial computational power, but their architecture is not always fully utilized in *Short-term* simulations. In this context, the *edge-platform* emerges as an effective solution. It combines the efficiency of a low-consumption device with the parallel processing power of a GPU, which is very useful when having to deal with many small tasks in a limited time. Its computational resources can be used to handle the intensive computational requirements of high-resolution simulations while minimizing latency and maximizing throughput.

For the *Short-term* scenario, the speed-up factor obtained is only 1.57. This relatively modest improvement demonstrates that the *edge-platform* has acceptable performance when dealing with *Short-term* simulations and overcomes its limitations by having a margin of success, as they can deliver relatively fast, high-resolution simulations.

The *workstation* has a significantly larger number of CUDA cores, higher memory bandwidth, and overall architecture, making it more capable of handling intensive parallel computations and having the capacity to process a more significant number of points per second. When the *Perimeter Resolution* decreases, the number of perimeter points increases, requiring greater computational resources. Fig. 3 shows the number of perimeter points processed for both platforms; it is clear that the number of points processed by the *workstation* is significantly higher than the processed by the *edge-platform*. The *workstation* reaches its peak processing capacity when simulating 24 h of fire evolution, at which point it achieves its maximum number of points per second. Beyond this threshold, performance declines sharply, likely due to memory constraints or computational overhead. However, for *Short-term* simulations, the performance gap between the two platforms is considerably smaller. In these cases, the edge platform remains competitive in terms of processing speed while offering the additional advantage of lower power consumption.

To have a broader view of the benefits of using *edge-platform* in an emergency scenario, we want to study the energy efficiency of both platforms when calculating the simulation. To measure the energy efficiency of both systems, we used the previously introduced *Energy Capability* function (Eq. (2)). Fig. 4 shows the *Energy Capability* of both systems for different simulation time windows (5, 10, 24, and 48 h). One can note that *edge-platform* stands out among *workstation* in terms of the number of processed fire front perimeter points per joule (#points/joule).

As we can see, in all situations, *edge-platform* consistently demonstrates a higher *Energy Capability*. In *Long-term* simulations, the *Energy Capability* of *edge-platform* is between 2 to 4 times greater than the *Energy Capability* of *workstation*. These findings are attributed to the fact that although the TDP of the *workstation* is approximately 10 times higher than that of *edge-platform*, see Table 1, its execution is roughly 8 times slower than that of the *edge-platform*. However, the *edge-platform* is especially good for *Short-term* simulations. In this case, *Energy Capability* of *edge-platform* is around 10 time more efficient when calculating the propagation of perimeter points than the *workstation*, 2,757.12 points/joule and 259.17 points/joule respectively.

The main objective of this work is to validate that the *edge-platform* is good enough for solving a 5-m simulation. Table 2 shows that it is able to provide acceptable execution times (between 0, 5 and 1, 6 s) for different propagation times when PR is 5 m. To analyze the performance of the edge platform, we have pictured the comparative speed-up between both platforms in Fig. 5. Speed-up is calculated as the ratio between the execution time of the simulation on the edge system and the execution time on the workstation. Fig. 5 shows this ratio, calculated for different propagation times. On the one hand, the *Green-up* shows that the *edge-platform* is between 2 and 10 times more energy efficient than *workstation*. On the other hand, the *speed-up* varies from 1.57 to 8.71, which indicates that *workstation* has better computational performance than *edge-platform*. By analyzing the balance between *Green-up* and *speed-up*, one can reach the scenario that best suits our edge computing system. The results show a clear case in which *edge-platform* takes advantage in front of *workstation* when the simulation time is equal to 5 h, *Short-term* simulations, the *edge-platform* is 10 times more efficient than *workstation* but, in contrast, the *speed-up* is only 1.57. Therefore, this study concludes that *Short-term* simulations with a PR equal to 5 m are the best scenarios for our edge computing platform. Thus, we will refer to these simulation settings as the *Edge Scenario*. In this scenario, the power efficiency of the *edge-platform* makes it the perfect platform to apply edge computing to the simulation of the forest fire spread *in situ*, favoring a rapid response to possible fire variations and optimizing firefighting resources.

3.2. Real fire scenario analysis

This section is dedicated to studying the viability of using an edge computing platform in a real wildfire scenario. We are interested in

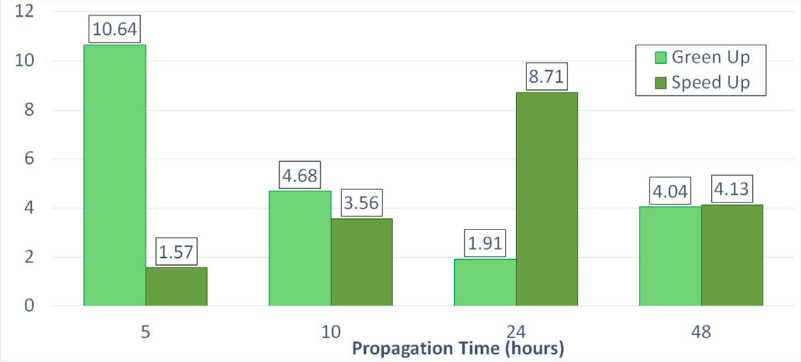


Fig. 5. *Green-up* of *edge-platform* (light green column) versus *Speed-up* of *workstation* (dark green column) for different *Simulated Time* and PR equal to 5 m.

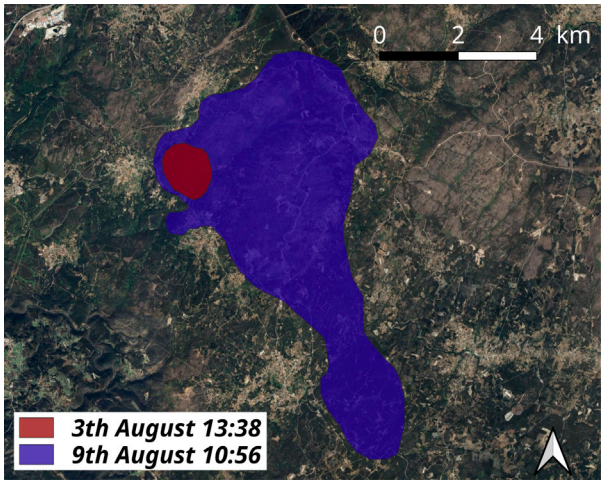


Fig. 6. Map of the São Joaninho area. The perimeter of the red shape has been used as initial perimeter (ignition Perimeter) and the final perimeter is obtained from the blue shape [26].

considering the lowest values for *Perimeter Resolution* as they represent the highest simulation quality results. Also, we want to assess the system capacity of producing forest fire simulations for short time windows to provide quick explorations of the spread. This section is devoted to showing the viability of using edge computing in a real wildfire. The main conclusion that stands from the study performed in the previous section is that the best scenario for edge computing is the so-called *Edge scenario*, that is, *Short-term* simulations. For that reason, in this section, we will consider forest fire spread simulations within a time window equal or less than 5 h (*Short-term* simulations) and with a PR (*Perimeter Resolution*) equal to 5 m. Therefore, the simulation results reported in this section correspond to the evolution of the firefront for five different propagation times within the time window of *Short-term* simulations (*Simulated time* equal to 1, 2, 3, 4 and 5 h). As a study case, we have used a forest fire that took place in 2013 in the region of São Joaninho, Portugal. The fire began on August 3rd, and the total burnt area was 1973ha. Fig. 6 shows the fire perimeter on August 3rd at 13:38 am (red shape) and the burned area on August 9th at 10:56 am (blue shape).

Our edge platform, NVIDIA Jetson AGX Xavier, has the possibility to work with different *Energy Modes* [27], determined by a set of parameters that define the working characteristics of the system in terms of resources available and energy consumption. Energy savings in the *edge-platform* are achieved through dynamic management of the number of active CPU cores, CPU clock frequencies, and EMC (External Memory Controller) frequency. In the previous section, the default energy mode was used, *Mode* 15 W. In this work, we focus on analyzing the behavior of three different *Energy Modes*, as seen in Table 3.

Table 3
Nvidia Jetson AGX Xavier power modes tested, [27].

	15 W	30 W 2 cores	MAXN
CPU Cores	4	2	6
CPU max freq. (MHz)	1200	2100	2265.6
GPU max freq. (MHz)	670	900	1377
Memory max freq. (MHz)	1333	1600	2133
Thermal design power (W)	15	30	50

Table 4
Execution time spent for different energy Modes of *edge-platform* when running simulations with PR equal to 5 m and varying the propagation times from 1 to 5 h.

Energy mode	1 h	2 h	3 h	4 h	5 h
15 W	16.373	29.973	41.091	47.808	57.087
30 W 2 Cores	10.650	19.373	26.035	30.441	35.877
MAXN	9.745	17.678	23.673	27.706	32.566

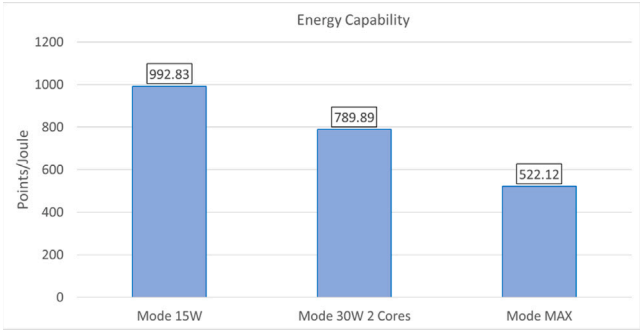
When the *edge-platform* works with the default configuration, *Mode* 15 W, some CPU cores are turned off. This reduction in the number of active cores leads to lower energy consumption but may result in reduced processing capability. Therefore, it is necessary to modify the default configuration to improve the performance of the *edge-platform*. Additionally, the CPU clock frequency is limited depending on the Energy mode used. The EMC controls the communication between the CPU and external memory. Its operating frequency has a significant impact on energy consumption. In the *Mode* 15 W, the *edge-platform* limits the EMC clock frequency, reducing the energy consumed.

On the contrary, the *Mode MAXN* eliminates all consumption restrictions, enabling the use of all CPU cores and increasing the GPU frequency to maximize the computational power of the platform. However, this comes with a trade-off: operating at maximum computational capacity results in a significant increase in energy consumption. The energy consumption of the *Mode MAXN* can fluctuate between 30 and 50 W, but it can reach peaks of 55 W.

In this work, we consider a *Mode MAXN* high activity average consumption of 50 W as representative of the situation with the maximum use of resources. As in the previous section, the simulated fire behavior is the same regardless of the Energy Mode used. Therefore, our focus is on analyzing the performance and energy efficiency of different Energy Modes of the edge-platform.

Table 4 includes the execution time spent when running *Simulated time* equal to 1, 2, 3, 4 and 5 h at *edge-platform* using three different energy Modes. It is important to see that all the execution times measured are under one minute. These results highlight that the *edge-platform* has the capability of generating high-resolution simulations in a near-real time.

(a) Execution Time of *edge-platform* for different energy Modes when running a 5 hours simulations.



(b) *Energy Capability* of *edge-platform* for different energy Modes when running a 5 hours simulation.

Fig. 7. *Edge-platform* efficiency measured in terms of Execution Time and Energy Capability for three different energy modes.

Fig. 7(a) presents the execution time of the different energy modes, showing significant differences in performance. The default mode, *Mode* 15 W, is the slowest as the resources available in the *edge-platform* are limited. *Mode MAXN* is the fastest, nearly halving the execution time compared to *Mode 15* W. This is because it has available all system resources without limitations to maximize the performance of the *edge-platform*. We have configured the system so that the use of a mode with 2-core and 30 W is able to obtain an acceptable time (3 seconds more than *Mode MAXN*) within an energy envelope of 30 W. In a real scenario, determining the optimal strategy or configuration for executing forest fire simulations, whether to prioritize speed or energy efficiency, becomes a critical decision. *Mode MAXN* is the fastest configuration; however, this performance comes at a significant energy cost, which may not be ideal in energy-constrained environments. Its Thermal Design Power (TDP) is 50 W, more than double the TDP of *Mode 15* W. It means that if we connect the *edge-platform* to a 55.5 Wh battery, it will last just over one hour at that load. On the other hand, *Mode* 15 W limited resources available may not be sufficient for particularly complex simulations. For these cases, we have found a specific configuration of the system that will provide good performance within controllable power consumption.

To understand the impact of energy and resource configuration in solving our particular real case of forest spread simulation, we use the *Energy Capability*, see Eq. (2), for the tree scenarios mentioned. EC reflects the number of perimeter points propagated from the initial perimeter to the final perimeter per work unit. Fig. 7(b) shows EC computed for the three different energy Modes when PR equals 5 m and the propagation time is 5 h. Among the modes tested, *Mode* 15 W emerges as the most efficient, achieving the highest EC. This energy efficiency results in noticeably worse performance, as *Mode* 15 W is the slowest. In contrast, *Mode MAXN* is the least efficient but obtains the lowest execution time. Finally, *Mode* 30 W shows a good balance between performance and power consumption.

Our real scenario trade-off decision will be whether to use all resources available to obtain the best execution time or save power by not using some resources and accepting performance penalties to allow an extended sensor data collection. Fig. 7 illustrates the general trend of the energy modes for the *edge-platform*. The figure shows that as more computational resources are available in the *edge-platform*, the execution time required for a single simulation decreases. However, this performance improvement comes at the cost of energy efficiency, as the Energy Consumption (EC) rises significantly with faster execution times. A clear inverse relationship between velocity and EC is evident in the figure: fastest simulations require significant energy consumption.

The new computational GPU-accelerated embedded systems, like *edge-platform*, open a new perspective where the monitoring and prediction of the fire behavior could be made in the same place where the fire occurs. These systems enable real time simulations and data processing in situ, providing crucial information without relying on external, high-power computational centers. As a first approximation, a balanced configuration can be employed that optimizes both performance and energy consumption, offering an efficient solution for real time fire management while minimizing resource use in energy-constrained environments. However, we can clearly distinguish two different extreme situations. On the one hand, when we are interested in minimizing the execution time to perform the forest fire simulation, the results show that we can configure the system to operate at maximum capabilities. On the other hand, when we want to maximize energy efficiency, we can configure the system to a long-life operation mode. In this case, we have to sacrifice some performance, which translates into an increment in execution time. In the particular case where the embedded system is combined with a drone to reduce input data uncertainty, "extreme edge computing", energy optimization could be a crucial aspect. Suppose the fire burns in a zone with difficult access and low connectivity. In that case, the computational system must be as energy-efficient as possible, which implies increasing the execution time. In this case, using *edge-platform* with energy *Mode 15* W is the best choice.

Regardless of the specific fire management scenario, the edge-platform proves to be a versatile and adaptable tool. Its computational and consumption characteristics make it an ideal platform for assessing the short-term risk of a wildfire. The platform's operational mode can be flexibly adjusted to meet the urgent response requirements of a hazard, such as a forest fire. As the simulation requirements and the time/cost balance may vary, users should consider the unique aspects of their problem and their limitations to choose the most suitable energy Mode.

The obtained results highlights the challenge of balancing execution time and energy efficiency on the *edge-platform*. Users must choose between faster execution times, which consume more energy, or slower simulations that save power. Ultimately, the decision depends on the context of the application, whether the priority is minimizing execution time or optimizing for energy efficiency.

4. Conclusions

One of the principal problems in forest fire spread simulation is the uncertainty of the input data. Unmanned Aerial Vehicles (UAVs) can significantly reduce this uncertainty by directly measuring data at the same place where a fire is occurring in real time. However, processing this data typically requires HPC platforms, which are costly and could be oversubscribed, leading to delays. Additionally, transmitting data and receiving fire behavior predictions becomes impractical in areas with low or no connectivity. To overcome these problems, we propose to leverage Edge Computing to enable real time wildfire monitoring and prediction without relying on HPC platforms. The Embedded Systems with low-consumption GPUs like the *NVIDIA Jetson AGX Xavier* open a new perspective where monitoring and prediction of fire behavior could be made *in situ* at high resolution. The results demonstrate that

it is possible to perform short-term fire spread prediction, 5 h or less, with a Perimeter Resolution of 5 m, eliminating the dependence on HPC platforms and network connectivity, making this approach particularly valuable for remote or disaster-stricken areas where traditional communication infrastructure is unavailable. In this scenario, the near future fire propagation can be actualized in less than one minute, which means that the monitoring of the wildfire can be done close to real time. Moreover, using low-power embedded systems significantly reduces energy consumption, making them well-suited for deployment in emergency scenarios where power availability is limited. Therefore, we can conclude that Edge Computing presents a promising alternative to conventional HPC-based simulations, enabling faster decision making and more agile responses to sudden changes in fire behavior. Future work will focus on testing the complete proposed edge computing system *in situ* during a real wildfire event, assessing its performance, reliability and effectiveness in operational environments.

CRediT authorship contribution statement

Carlos Carrillo: Writing – original draft, Validation, Software, Investigation, Conceptualization. **Tomàs Margalef:** Writing – review & editing, Supervision, Funding acquisition. **Antonio Espinosa:** Writing – review & editing, Supervision, Project administration, Funding acquisition. **Ana Cortés:** Writing – review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work has been granted by the Ministerio de Ciencia e Innovación MCIN AEI/10.13039/501100011033 under contract PID2020-113614RB-C21 and PID2023-146193OB-I00, and CPP2021-008762 by the European Union-*NextGenerationEU* / PRTR. It also has been partially granted by the Catalan Government under grant 2021-SGR-574.

Data availability

Data will be made available on request.

References

[1] T. Artés, A. Cencerrado, A. Cortés, T. Margalef, Core allocation policies on multicore platforms to accelerate forest fire spread predictions, in: Parallel Processing and Applied Mathematics - 10th International Conference, PPAM 2013, Warsaw, Poland, September 8-11, 2013, Revised Selected Papers, Part II, 2013, pp. 151–160.

[2] C. Brun, T. Margalef, A. Cortés, A. Sikora, Enhancing multi-model forest fire spread prediction by exploiting multi-core parallelism, J. Supercomput. 70 (2) (2014) 721–732.

[3] A. Cencerrado, T. Artés, A. Cortés, T. Margalef, Relieving uncertainty in forest fire spread prediction by exploiting multicore architectures, in: Proceedings of the International Conference on Computational Science, ICCS 2015, Computational Science at the Gates of Nature, Reykjavík, Iceland, 1-3 June, 2015, 2014, 2015, pp. 1752–1761.

[4] F. Afghah, A. Razi, J. Chakareski, J. Ashdown, Wildfire monitoring in remote areas using autonomous unmanned aerial vehicles, in: IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops, INFOCOM WKSHPS, 2019, pp. 835–840.

[5] M.A. Akhloufi, A. Couturier, N.A. Castro, Unmanned aerial vehicles for wildland fires: Sensing, perception, cooperation and assistance, Drones 5 (1) (2021) URL https://www.mdpi.com/2504-446X/5/1/15.

[6] M. Kucharczyk, C.H. Hugenholtz, Remote sensing of natural hazard-related disasters with small drones: Global trends, biases, and research opportunities, Remote Sens. Environ. 264 (2021) 112577.

[7] M. Avgeris, D. Spatharakis, D. Dechouniotis, N. Kalatzis, I. Roussaki, S. Papavassiliou, Where there is fire there is SMOKE: A scalable edge computing framework for early fire detection, Sensors 19 (3) (2019) URL https://www.mdpi.com/1424-8220/19/3/639.

[8] J. Hu, H. Niu, J. Carrasco, B. Lennox, F. Arvin, Fault-tolerant cooperative navigation of networked UAV swarms for forest fire monitoring, Aerosp. Sci. Technol. 123 (2022) 107494.

[9] T.K. Xian, H. Nugroho, Forest fire detection for edge devices, in: 2022 IEEE International Conference on Artificial Intelligence in Engineering and Technology, IICAIET, 2022, pp. 1–4.

[10] X. Li, H. Gao, M. Zhang, S. Zhang, Z. Gao, J. Liu, S. Sun, T. Hu, L. Sun, Prediction of forest fire spread rate using UAV images and an LSTM model considering the interaction between fire and wind, Remote. Sens. 13 (21) (2021) URL https://www.mdpi.com/2072-4292/13/21/4325.

[11] Team-GFORCE, UAV based wildfire detection and simulation, 2021, URL https://www.alibabacloud.com/blog/project-showcase-%7C-uav-based-wildfire-detection-and-simulation_598261.

[12] D. D'Ambrosio, S. Di Gregorio, G. Filippone, R. Rongo, W. Spataro, G.A. Trunfio, A multi-GPU approach to fast wildfire hazard mapping, in: Simulation and Modeling Methodologies, Technologies and Applications - International Conference, SIMULTECH 2012 Rome, Italy, July 28-31, 2012 Revised Selected Papers, 2012, pp. 183–195.

[13] S. Di Gregorio, G. Filippone, W. Spataro, G.A. Trunfio, Accelerating wildfire susceptibility mapping through GPGPU, J. Parallel Distrib. Comput. 73 (8) (2013) 1183–1194.

[14] R.V. Hoang, Wildfire Simulation on the GPU (Ph.D. thesis), University of Nevada, 2008.

[15] V.G. Ntinas, B.E. Moutafis, G.A. Trunfio, G.C. Sirakoulis, Parallel fuzzy cellular automata for data-driven simulation of wildfire spreading, J. Comput. Sci. 21 (2017) 469–485.

[16] F.A. Sousa, R.J.N. dos Reis, J.C.F. Pereira, Simulation of surface fire fronts using firelib and GPUs, Environ. Model. Softw. 38 (2012) 167–177.

[17] M. Denham, K. Laneri, Using efficient parallelization in graphic processing units to parameterize stochastic fire propagation models, J. Comput. Sci. 25 (2018) 76–88, URL https://www.sciencedirect.com/science/article/pii/S1877750317308773.

[18] M.A. Finney, FARSITE: Fire Area Simulator—Model Development and Evaluation, Research Paper RMRS-RP-4 Revised 236, 1998.

[19] FireLab, FARSITE tutorial website, 2007, URL http://fire.org/downloads/farsite/WebHelp/using_farsite_help.htm.

[20] I. Knight, J. Coleman, A fire perimeter expansion algorithm-based on huygens wavelet propagation, Int. J. Wildland Fire 3 (1993).

[21] R. Rothermel, A mathematical model for predicting fire spread in wildland fuels, 1972.

[22] NVIDIA jetson AGX xavier developer kit, 2018, https://developer.nvidia.com/embedded/jetson-agx-xavier-developer-kit.

[23] S. Abdulsalam, Z. Zong, Q. Gu, M. Qiu, Using the greenup, powerup, and speedup metrics to evaluate software energy efficiency, in: 2015 Sixth International Green and Sustainable Computing Conference, IGSC, 2015, pp. 1–8.

[24] D.X. Viegas, Slope and wind effects on fire propagation, Int. J. Wildland Fire 13 (2) (2004) 143–156.

[25] C. Carrillo, T. Margalef, A. Espinosa, A. Cortés, Accelerating wild fire simulator using GPU, in: Computational Science - ICCS 2019 - 19th International Conference, Faro, Portugal, June 12-14, 2019, Proceedings, Part V, in: Lecture Notes in Computer Science, vol. 11540, Springer, 2019, pp. 521–527.

[26] J.R. Centre, European forest fire information system, 2011, http://forest.jrc.ec.europa.eu/effis/.

[27] NVPModel – NVIDIA jetson AGX xavier developer kit, 2018, URL https://www.jetsonhacks.com/2018/10/07/nvpmodel-nvidia-jetson-agx-xavier-developer-kit/.