# Wildfire Spreading Simulator Using Fast Marching Algorithm

Juan Carballeira*, Carolina Nicolás,Santiago Garrido, Luis Moreno

Robotics Laboratory, Department of System Engineering and Automation, Carlos III University of Madrid, 28911 Madrid, Spain; *jcarball@pa.uc3m.es

SNE 31(3), 2021, 159-167, DOI: 10.11128/sne.31.tn.10577
Received: March 10, 2021 (Selected EUROSIM 2019 Postconf. Publ.), Revised: August 31, 2021; Accepted: September 2, 2021
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna, ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

**Abstract.** Programs that can predict wildfire behavior are a very useful tool in terms of extinguishing these fires more effectively. State of the art wildfire simulators present some drawbacks such as not being sufficiently user-friendly, being expensive, requiring great computational power or having poor graphical representation. This paper presents a prototype wildfire simulation app that uses Fast Marching (FM) as its core algorithm. The wildfire app is developed as a Matlab GUI. Said application shows the shape of the fire front at a given moment in time in a 3D map of the terrain affected by the fire. Any real life maps can be loaded to the application for wildfire prediction. The user can choose to vary parameters such as starting (ignition) and ending points, wind direction and speed and propagation time, and see its effect on fire propagation. Interface response to each change in the input is very fast, therefore proving the efficiency of the algorithm. Although a prototype, the wildfire basic app is superior to some state of the art simulators regarding certain important features. It can be concluded that Fast Marching is a valid core algorithm for a fire simulator. The way the app is programmed in Matlab confers it flexibility, enabling further specific changes that make it truly competitive against currently used wildfire simulators.

## Introduction

Wildfires constitute a serious threat around the globe,of special importance in the Mediterranean area. Each year, a country, such as Spain, experiments terrible environmental, material and economic loses that have been estimated to be of 3000 euros per $hm^2$ by Aguilera [1]. In addition to these loses, wildfires cause numerous deaths, most of them fire fighting professionals trapped in the fire due to abrupt changes in fire trajectory. Fire trajectory prediction enables the fire fighting team to choose the critical spots in order to save resources and extinguish the fire more effectively. A crucial characteristic for a useful predictor is a quick and accurate response to changes in parameters affecting the fire such as the speed and direction of the wind. This characteristic can be achieved by using a very efficient prediction model or algorithm. The Fast Marching algorithm, developed by Osher and Sethian [2] and currently used for path planning applications, has proven to be a very efficient algorithm to solve the Eikonal wave equation.

Therefore, the purpose of this paper is to prove the viability of the FM algorithm as a core algorithm to a wildfire simulator. A prototype wildfire app is developed in Matlab Guided User Interface to this end. The final objective of the prototype is to show the possibility of a user-friendly, economical, very competitive simulator that responds very efficiently to changes in inputs affecting fire spread.

## 1 State of the Art Wildfire Simulators

The most relevant fire prediction simulators are FireStar, FireRS, Prometheus and Farsite.

### 1.1 FireStar

FireStar [3] , is a project providing wildfire prediction assistance for users in the Mediterranean area. Its main characteristic is that the prediction model used is entirely physical. It predicts fire spread by solving a series of differential equations based on mass, momentum and

energy conservation. Results regarding the effect of numerous wildfire related variables (such as temperature or gas velocity) are extremely precise. Its biggest flaw is that due to the big computational cost associated with solving these equations, it can only work in 1 and 2 dimensions. Building a 3D prediction simulator using a entirely physical model is a very impractical computational task.

### 1.2 FireRS

FireRS [4], stands for wildFIRE Remote Settings and is a massive still unfinished project that started in 2016 for wildfire prediction and monitoring. It is mainly funded by the European Regional Development Fund (ERDF), coordinated by the University of Vigo and the University of Oporto, and the Centre National de Reserche Scientifique as collaborator. Its 2 million budget is destined to provide a global information system with real time data regarding the fires, actuation protocols, GPS positioning, a fleet of autonomous UAVs, infrared sensors and wildfire prediction, among other tools.

### 1.3 Prometheus

Prometheus [5], is the most widely used wildfire predictor in Canada. It uses Huygens' principle, stating that light propagation front at a given instant contains the initial point of small propagations, and the envelope curve of those small propagations determines the shape of the propagation front on the next instant. Figure 1 graphically represents this phenomenon.

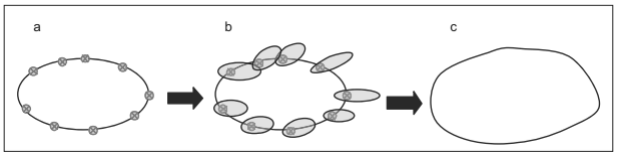


**Figure 1:** Huygens' principle.

Prometheus uses COM (Common Object Model) as programming tool, enabling the use of a higher level COM for computation and 5 lower level COMs related to variables that affect the fire spread (fuel, weather, etc.).

The user interface associated with Prometheus is Burn-P3 [6] Figure 2 represents the maps for the various possible inputs (the only required input is the information regarding fuels). The simulator returns a map and data regarding the probability of the wildfire reaching each specific zone of the considered terrain.

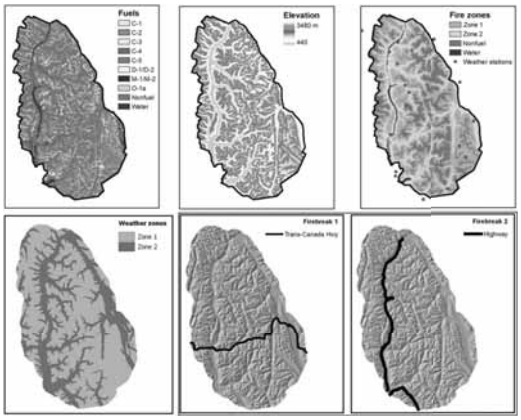


**Figure 2:** Burn-P3 inputs. Adapted from [**?**].

### 1.4 Farsite

Farsite, [7], is probably the best existing wildfire simulator. It also uses Huygens' principle in combination with Rothermel's surface model and Van Wagner's model.

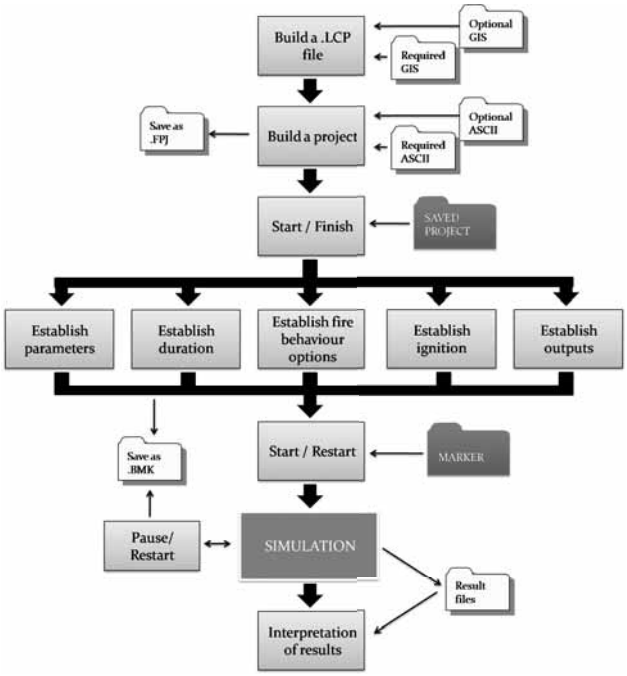


**Figure 3:** Farsite functioning scheme.

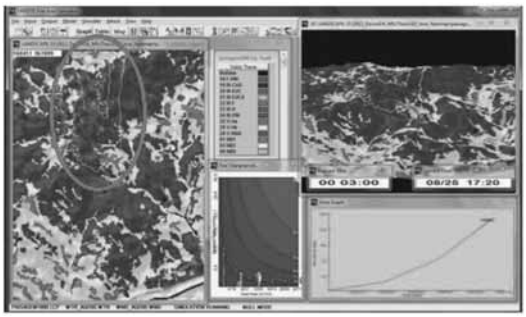

**Figure 4:** Farsite user's interface.

Figure 3 contains a diagram that shows the use procedure for Farsite. Actions on the program are represented as rectangles, inputs and outputs are represented as folders.

Figure 4 shows the interface of program for a given wild fire simulation. The wildfire spread is represented by thin white lines.

## 2 Methodology

Subsection 2.2 explains the fundamentals of the Fast Marching Algorithm, which is the core of the application. Subsection 2.1 covers the main development simulation and design aspects of the simulator and its main features.

### 2.1 Fast Marching Algorithm for wildfire prediction

The Fast Marching Algorithm [8], is a specific case of the Level Set method, initially designed to simulate the propagation of a wave in a discrete space. It does so by solving the Eikonal equation at every node of a discrete map.

This algorithm predicts the expansion of a wave given a velocity , cost or viscosity matrix. Its functioning can be intuitively understood by imagining how the wave front caused by a stone hitting water might be different if instead of water there was a mixture of liquids with different viscosities. In this analogy the viscosities of the liquids represent the different values of the cost matrix.

There are three significant matrices used in the algorithm:

- W matrix: Cost matrix. The value in each cell states the 'cost', or difficulty, of traveling to that node. It is the input to the algorithm.
- S matrix: Contains information about where the propagation front is at each moment. It determines weather a node is unknown (0), part of the front (1), or dead, which means that it was part of the front previously, but it is not anymore (-1).
- D matrix: Distance matrix. Each node contains, for the purpose of this application, information regarding how much time it has taken the front to reach said node. It can be considered as the final output of the algorithm.

Table 1 shows how the values of the matrices S and D would update with each iteration of the algorithm if the input matrix W was a 3x3 matrix of ones.

| It. | 0 | 1 | 2 |
|---|---|---|---|
| S | $\begin{matrix}0&0&0\\0&0&0\\1&0&0\end{matrix}$ | $\begin{matrix}1&0&0\\1&0&0\\-1&1&1\end{matrix}$ | $\begin{matrix}-1&1&1\\1&0&0\\-1&1&1\end{matrix}$ |
| D | $\begin{matrix}\infty&\infty&\infty\\\infty&\infty&\infty\\0&\infty&\infty\end{matrix}$ | $\begin{matrix}1&\infty&\infty\\1&\infty&\infty\\0&1&1\end{matrix}$ | $\begin{matrix}1&1.71&1.71\\1&\infty&\infty\\0&1&1\end{matrix}$ |

**Table 1:** Values for matrices S and D during each iteration for an example matrix W.

FM has been traditionally used for path planning in robotics. Nevertheless, the idea of fire spread behaving essentially like a wave where the values of the cost matrix represent the flammability of the terrain, presents the possibility of FM being a suitable algorithm for wildfire prediction. Another characteristic of the FM algorithm is that the propagation velocity is always considered positive, which matches with wildfire behavior.

The main advantage of FM is its computational speed and efficiency. Other fire spread prediction methods such as Huygens' principle (Farsite, Prometheus) are slower and less efficient computationally.

It is obvious that wildfire spread depends on factors other than the terrain flammability: wind, slope of the terrain, etc. These factors are vectorial. Vectorial fields can be included in the FM algorithm [9], by adding such vectorial fields to the D matrix (which is in fact a potential field created from scalar variables). By modeling

the influence of said factors into vectorial fields, these can be included in the algorithm. Figure 5 illustrates an example of how the propagation is affected by vector fields in different directions

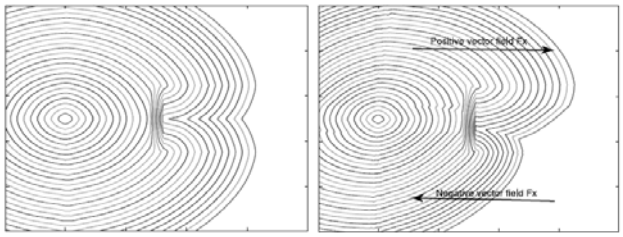


**Figure 5:** Effect of an external vectorial field to the FM algorithm. Adapted from [9].

## 2.2 Application design

Our wildfire prediction application is a simple Matlab graphical user interface (GUI). Its purpose is to show the propagation of a wildfire along a 3D map of the terrain over time in a very clear way. The main objective of this paper is to prove that FM is a viable option for a wildfire simulator. Another objective is for the user to be able to select and vary parameters that affect the fire spread.The main design aspects of the application are explained in this section.

### 2.2.1 Altitude and flamability maps

The application requires two maps as inputs (in matrix form) of the location where fire spread is simulated.

- Altitude matrix: *nxm* matrix. Required for the 3D representation of the spread. Each cell contains the altitude of that point in the map.

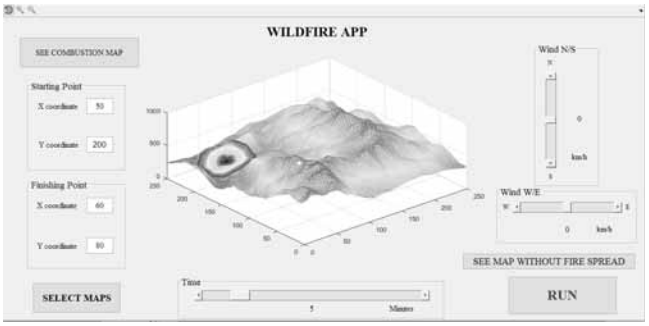


**Figure 6:** Screenshot of the wildfire prediction application.

- Flammability matrix: *nxm* matrix. FM algorithm requires a value ranging between 0 (not flammable) and 1 (extremely flammable) for each point of the map.

Any altitude and flammability maps can be used as long as they share the same dimensions. For a first test of the application, real maps from the LandFire website were used. Said website contains altitude and flammability maps of sites all along the USA territory. This maps, in .tiff format, are converted to Matlab matrices, resized and their values properly mapped before they can be used as inputs for the application.

### 2.2.2 Vectorial inputs: wind and terrain slope

Among the vectorial variables that might affect fire spread, two significant ones are taken into consideration in this application:

- Wind: Given that the maps considered for simulation represent a relatively small terrain, and for modeling simplicity, wind effect is considered constant in every point of the map. Additionally, also for simplicity reasons, only a bidimensional wind, parallel to the ground, is considered. The user can easily choose a wind input with any module and direction by adjusting the sliders for the N/S and W/E wind in the application. This is graphically represented in Figure 6.
- Terrain slope: Fire travels more easily in an upwards direction. Therefore, the slope or gradient of the terrain will have a positive vectorial effect on fire spread, similar to that of the wind. With the altitude matrix as an input it is easy to compute a 2D matrix with information about its gradient. This gradient matrix is used as a vectorial input to the FM algorithm, just as the wind was. This is graphically represented in Figure 7.

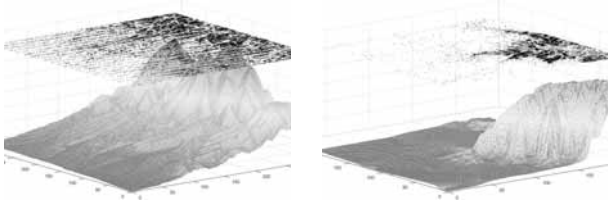

**Figure 7:** Gradients for two different altitude maps.

### 2.2.3 Graphical representation of fire spread

Once the FM algorithm is given all the inputs and executed, the next objective is to clearly represent the propagation of fire (D matrix) along the considered terrain. This representation must be consistent with the instant of the fire spread that the user wishes to see. Therefore, there are two main design aspects related to the representation of the fire spread that need to be understood:

- Contour matrix: In order to represent the matrix D as a propagating wave, it is transformed into contour levels with assistance of Matlab?s contour function. Every aspect regarding the graphical representation of the fire spread is related to a matrix known as the contour matrix. This matrix contains information regarding the number of contour levels (enough to make the propagation appear to be continuous), number of points in every level and 3D coordinates of such points. Figure 8 represents an example of a contour matrix of 3 levels for a better understanding.

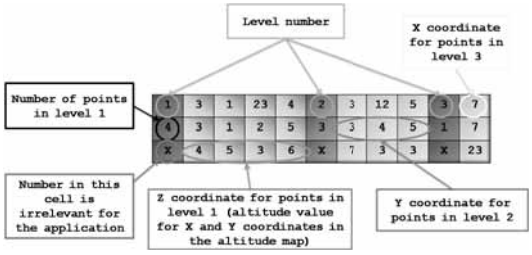


**Figure 8:** Example of a simple contour matrix.

- Time estimation: The simulator allows the user to see the fire spread at different snapshots of time. In order to offer this feature, an estimation of the total time taken by the wildfire to reach a certain final point is needed. This estimation is performed by computing a ratio between a rough estimation of the distance traveled by the wildfire and the average fire spread velocity.

The fire spread distance is calculated as the Euclidean distance between the initial and final points (in green in Figure 9). The velocity is estimated by performing the average in an estimated area of fire propagation (dark purple in Figure 9). The values of such area correspond to mapped values of the flammability matrix, where a 0.1 in the flammability matrix corresponds to a 1km/h fire spread velocity and a 0.9 in the flammability matrix corresponds to a 9km/h fire spread velocity. The mapping of the values is done accordingly with usual fire spread velocities as a function of terrain flammability. Additionally, the effect of the wind in fire spread velocity is taken into consideration, estimating an effect on velocity fire spread equal to wind velocity.

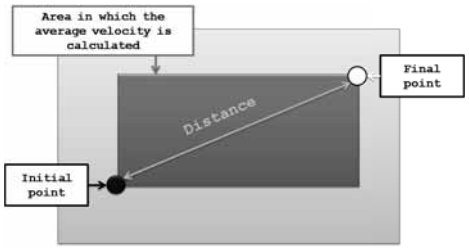


**Figure 9:** Distance and spread velocity representation.

### 2.2.4 GUI interface and its functionality

Even though the application is a viability trial for FM as a fire prediction algorithm, user-friendliness is strongly considered.

The simplicity of Matlab's GUI has helped the development of the main functionalities of a wildfire simulation application:

- Map selection: A pushbutton from the GUI ('SELECT MAPS') enables the user to upload the flammability and altitude matrices.
- Map display without propagation: A pushbutton from the GUI ('SEE MAP WITHOUT FIRE SPREAD') enables the user to visualize the 3D map for the region selected without the fire spread. The purpose of this is for the user to become familiar with the map, so the subsequent selection of the application parameters is easier.
- Starting and finishing points: The user can specify the coordinates of the fire ignition point and a finishing point (which will stop the simulation, once reached) in editable text boxes. The starting point is shown on the map in black and the finishing point in white.
- Time slider: The application estimates time it takes for the fire to reach the finishing point . For a better understanding of the fire spread, the user can view the propagation at any instant by using the 'Time' slider. The effect of the time slider is shown in Figure 10.

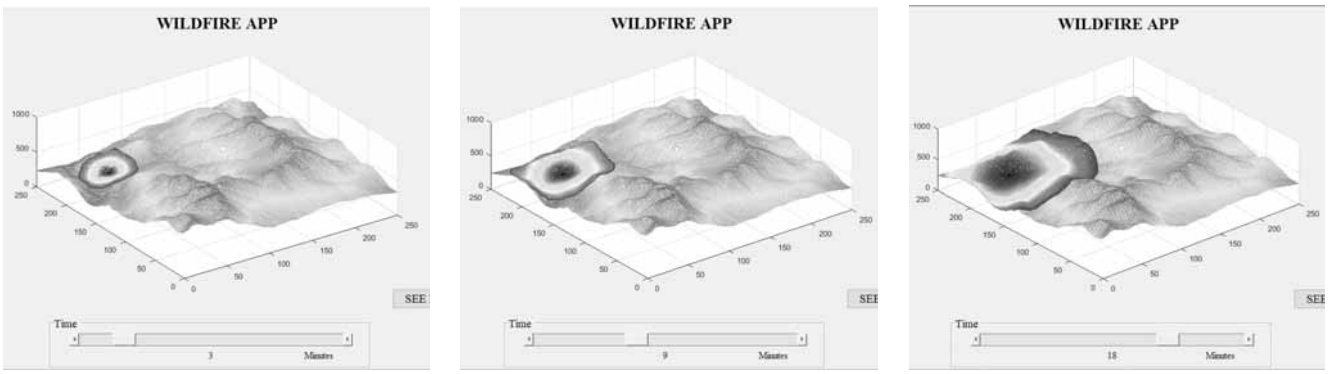


**Figure 10:** Simulation at different snapshots of time, in minutes.

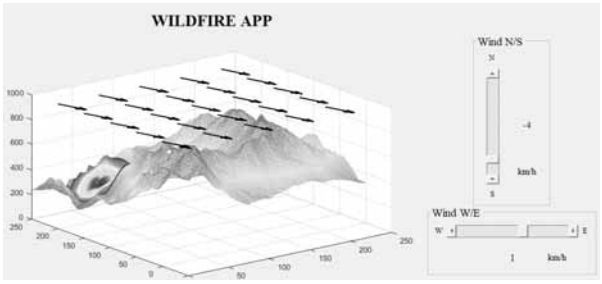


**Figure 11:** Display of the vectorial input related to the wind.

- Wind sliders: By selecting values in both of them, the user can choose how intense the wind is in each direction (North, South, East or West) and to provide an easier use, the direction of the wind is shown in the propagation map using black arrows. For a deeper understanding see Figure 11 .

- Flammability map visualization: The user might need a glimpse of the flammability map that has been used, so as to understand the propagation of fire. The push button ('SEE COMBUSTION MAP') pops up a window in which the flammability map is shown.

- Toolbar: The application includes a toolbar that enables the user to manipulate the maps (rotate, zoom in and out) for an optimal visualization.

- Run pushbutton: Starts the simulation. It must be pushed every time any changes in the inputs are made in order to see their effect.

# 3 Results

The results of different wildfire spread simulations are shown. The effect of the different variables that affect the fire spread are analyzed independently plus the results of the simulator in terms of computational efficiency are explained.

## 3.1 Effect of the flammability map

The flammability map determines fire spread, as it is an input for the FM algorithm. On the other hand, the altitude map is unrelated (except for the altitude gradient) to the algorithm, and it is only used to provide the user with a realistic view of the fire spread along the terrain. Therefore any combination of flammability and altitude maps can be used for the simulator. Figure 3.1 represents a commonly used altitude map, the Washington matrix, with a fictitious flammability map which is very flammable in every point except for a rectangular region.

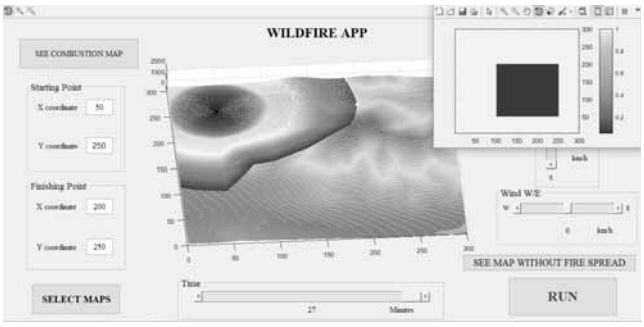


**Figure 12:** Washington matrix.

It can be easily observed that the fire spread avoids the region that is less flammable, creating an alternative path to that of the same altitude map and same inputs, but using a continuous flammability map.

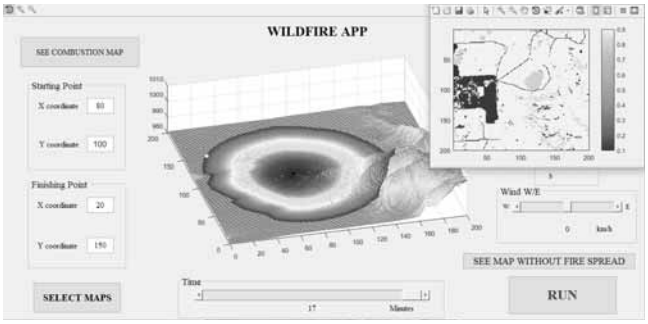


**Figure 13:** Real flammability and altitude maps.

Figure 13 represents the simulation results for a couple of real altitude and flammability maps.

### 3.2 Effect of the wind

The effect of the wind in the simulation is presented in Figure 14. For the same initial and final points in a given real life map, 3 different wind settings are configured, so as to appreciate the effect of the wind when every other variable is constant. It can also be noted from Figure 14 that some wind configurations speed the process of the wildfire spread reaching the final point and others slow it down. This affects the total time of fire spread in the time slider.

### 3.3 Computational efficiency of the simulator

In order to obtain results regarding the computational efficiency of the simulator the time required to run the main function of the application was computed. Said function includes both the FM algorithm and the representation of results, therefore, it is the core of the simulator. It runs every time the 'RUN' button is pressed in the GUI application and it requires the great majority of computational resources in the simulator. A brief study of the time needed to run the core function was performed for different maps (200x200 to 250x250) and different start and end points and wind settings, concluding that for no combination of inputs the function takes more than 10 seconds to run.

## 4 Discussion

The present section compares the state of the art wildfire simulators our wildfire app. The final goal is to prove that, although a basic, the wildfire app presents strong advantages compared to other simulators. This will prove the viability of FM as a wildfire prediction algorithm, and set a starting point for future work.

Our main purpose is to implement Fast Marching as a viable fire prediction algorithm in a user friendly wildfire simulator. As a prototype, there are a series of disadvantages to the use of the app with respect to the simulators presented in section :

- Very rough estimation of time, velocity and distance dimension correspondence. As introduced in section 2.2.3 Fast Marching is a mathematical algorithm, and so, correlations between wind velocity, time and fire spread velocity dimensions need to be thought outside from the algorithm itself. The estimations for this prototype are far less precise than those of simulators in section , but there is still room for improvement. It will probably be impossible to reach the precision of a physical model, such as the one used in FireStar, but with proper research work, levels of simulators such as Prometheus or Farsite could be reached.
- Fast Marching does not consider non positive velocity fields. If wind is not considered this is not a problem, but there could be a case in which a strong wind against the fire front direction could reverse the trajectory of the fire. If that were to happen the algorithm would collapse and so will the application, inducing a Matlab error that forces the application to be restarted. This is the greatest inconvenience when using FM algorithm for wildfire simulation. In order for the simulator to be functional for every case, the FM algorithm would need to be either modified for the specific case or the simulator would need to use an alternative algorithm or model for those very specific situations. Even if a less efficient model were used for those cases, as they are a small part compared to the total cases, the efficiency of the simulator would not decrease significantly.

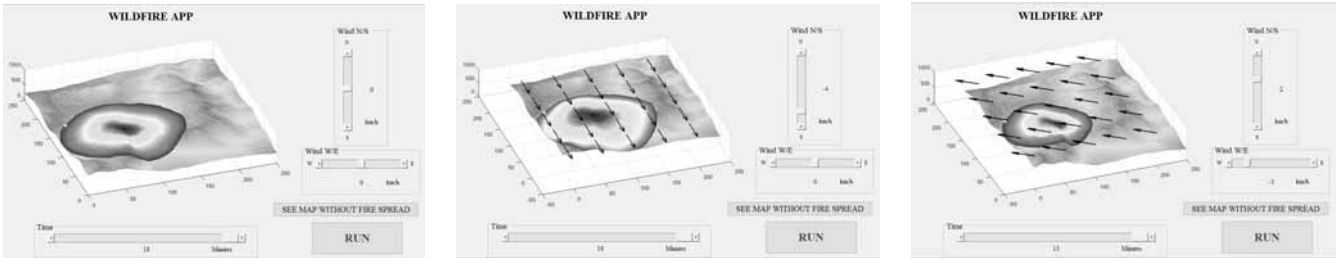


**Figure 14:** Wind effect on fire spread.

- The implementation of the wind as a vectorial input to the algorithm still needs to be perfected to represent the real effect of 3D real life wind. In addition, other meteorological inputs that might affect the fire spread such as air humidity should be included in further versions of the app in order to be more competitive with current simulators.

Nevertheless, this simple prototype does have a wide set of advantages over the commercial and currently used simulators:

- Higher efficiency than other simulators. Models using the Huygens' principle are far slower and less efficient than Fast Marching. It is easy to understand that the need to predict the trajectories of several points in the front and then calculating the envelope of those trajectories requires a significant amount of computational power. Needless to say that physical models, based on partial differential equation solving are also remarkably slower than Fast Marching.
- Its development is far more economical than the rest of simulators. As a visual comparison, the 2 million euro budget needed for the FireRS initiative can be contrasted with the wildfire app budget of 0 euros (provided a Matlab license is already owned). The wildfire app prototype has been developed essentially by one person, and subsequent improvements to the app can be probably conducted by small research team. This results in a big difference regarding simulators such as Prometheus or Farsite, that constitute big development projects, with their associated resources and personnel costs.
- User-friendly simulator, due to Matlab GUI interface. The user of this application does not need to have specific programming skills or specific wildfire propagation knowledge. This makes the simulator very accessible and attractive for use, since the organizations interested in using it would not need to invest in personnel training. The steps for the application prototype use can be easily explained in a one-page manual.

  Its structure of use is simple, as well user friendly. The only inputs that need to be imported are the combustion and elevation maps. Every other input (time, wind, starting points, etc.) can be selected in the application itself. No different formats, programming or simulating platforms need to be used. This is very different from the complex series of steps needed for a Prometheus or Farsite simulators (see Figure 3)

- The graphical representation of the fire spread is one of the strongest advantages of the wildfire app. The fire spread is presented in in a set of colours that makes it clearly distinguishable from the terrain map, and it also enables the user to see direction of front propagation (red most recent, blue least recent). This makes the wildfire app front representation superior to that of Farsite, as its fire front is presented with thin white lines, that cannot be traced clearly by the user (see Figure 4). The current real time propagation is represented, as opposed to the probability of the terrain being burned, which constitutes the output of the Prometheus simulator.

  Starting and finishing points are clearly traceable, enabling the user to easily familiarize with the coordinates of the map.

  The wildfire app has a properly working 3D representation format. The change from one map to another is done almost instantly. This 3D representation for mat is not supported by some of the current working simulators.

## 5 Conclusions

The objective of the research was to prove the viability of the Fast Marching algorithm to build a wildfire simulator. It can be concluded that such objective has been reached. The result is a wildfire app developed in the Matlab GUI?s environment. Said application shows the shape of the fire front at a certain moment in time in a 3D map of the terrain affected by the fire. Any real life maps can be loaded to the application for wildfire prediction. The user can choose to vary parameters such as ignition and ending points, wind direction and speed or propagation time and see its effect on fire propagation. Response to each change in the input is very fast, therefore proving the efficiency of the algorithm.

Regardless of the wildfire app being a basic prototype, it is superior to some state of the art simulators regarding certain important features.

It can be concluded that Fast Marching is indeed a valid core algorithm for a fire simulator. Moreover, the prototype of the wildfire app opens a line of research in perfecting this prototype and becoming a truly competitive tool against currently used wildfire simulators.

### References

[1] Aguilera M. La cruz de los veranos: cada hectárea quemada cuesta 3.000 euros. El Economista. URL http://ecodiario.eleconomista.es/espana/noticias/53524/08/06/La-cruz-de-los-veranos-cada-hectarea-quemada-cuesta-3000-euros.html. 2006

[2] Osher S, Sethian JA. Fronts propagating with curvature- dependent speed: Algorithms based on hamilton-jacobi formula- tions. *Journal of Computational Physics*.1988; Vol. 79.

[3] Morvan D, Dupuy J, Rigolot E, Valette J. Firestar: A physically based model to study wildfire behaviour. Forest Ecology and Management. 2006.

[4] FireRS team, FireRS webpage.URL http://www.fire-rs.com 2017

[5] Tymstra C, Bryce R, Wotton B, Taylor S, Armitage O. Development and structure of prometheus: the canadian wildland fire growth simulation model. Forest Service. 2010.

[6] Parisien MA, Kafka VG, Hirsch KG, Todd JB, Lavoie SG, Maczek PD. Mapping wildfire susceptibility with the BURN-P3 simulation model. Nat. Resour. Can., Can. For. Serv., North. For. Cent., Edmonton, Alberta. Inf. Rep. NOR-X-405. 36 p. [URL http://cfs.nrcan.gc.ca/publications?id=25627 ]

[7] Finney MA. FARSITE: Fire Area Simulator-model development and evaluation . Res. Pap. RMRS-RP-4, Revised 2004, Ogden, UT: U.S. Department of Agriculture, Forest Service, Rocky Mountain Research Station. 47 p.

[8] Gómez JV. Fast marching methods in path and motion planning: Improvements and high-level applications. Ph.D. thesis, University Carlos III Madrid. 2015.

[9] Garrido S, Moreno L, Martín F, Álvarez D. Fast marching subjected to a vector field path planning method for mars rovers. *Expert Systems With Applications.* 2017