Applying semi-empirical simulation of wildfire on real world satellite imagery data

Preprint · September 2021

3 authors:

Hachem Betrouni
National Polytechnic School of Algiers

Samy Ali Touati
National Polytechnic School of Algiers

Ahmed Moncef Bousselat
National Polytechnic School of Algiers

Applying semi-empirical simulation of wildfire on real world satellite imagery data.

Hachem Betrouni[1], Samy Ali Touati[2] and Ahmed Moncef Bousselat[3]

[1] National Polytechnic School of Algiers, Data science and AI
e-mail: hachem.betrouni@g.enp.edu.dz
[2] National Polytechnic School of Algiers, Automatic Engineering
e-mail: samy_ali.touati@g.enp.edu.dz
[3] National Polytechnic School of Algiers, Data science and AI
e-mail: ahmed_moncef.bousselat@g.enp.edu.dz

October 30, 2021

ABSTRACT

The knowledge of a free-burning fire's potential rate of spread is critical for safe and effective wildfire control, in this work We are carrying out a data driven cellular automata simulation of wildfire propagation.

Our model performs not only for the approximation of fire propagation, but also helps in finding critical zones in which fire would be most devastating and would spread to a large scale.

we also demonstrate how we extract useful data from satellite imagery and use machine learning techniques in order to prepare inputs used later for simulation.

We aim to locate critical zones and to estimate fire spread so that needed precautions can be taken to limit and control any eventual risks in real life, this can also help managers to rapidly predict the spread of fires, providing a decision basis for formulating effective fire extinguishing plans.

1. Introduction

Mathematical modeling of forest fires is one of the biggest challenges for modeling due to the diversity and complexity of the physical processes taking place in the fire zone and in the atmosphere above it, the effect of weather conditions, the possibility of wildfire spreading to a large area for a long time period, and other factors.
During the past 20 years, this research area has developed especially actively: many models for forest wildfires have been created. According to (1), these models can be divided into three large groups: empirical, semiempirical and physical.

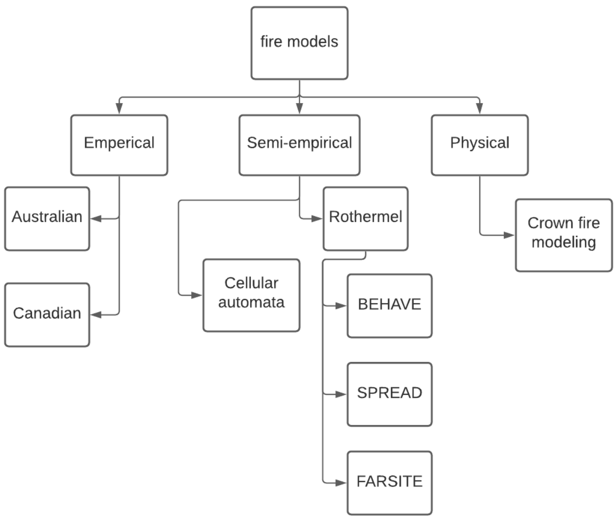

Fig. 1. some of the existing fire models

In this work we are presenting an implementation of a combination between two semi-empirical methods, we are going to combine a Cellular automata simulation with other semi-empirical equations that describes the spread rate of fire depending on three main factors :
slope, **fuel** and **wind**.

2. Cellular automata

Cellular automata are dynamic systems operating in discrete space and time, on a uniform, regular lattice and are characterized by local interactions.

Cellular Automata (CA) were invented by the mathematician Stanislaw Ulam and were used by J. von Neumann, followed by A.W. Burks and E. F. Codd, to solve the problem of the non-trivial self-reproduction in a logical system (7; 8).

A cellular automata is defined by a grid with start states and set of rules for state transitions. Generally, cellular automata consist of four elements, which could be considered as a tuple (X, S, N, f) :

- X are cells which are objects in any dimensional space, we can call this cellar space.
  In cellar space, each cell has the form $x = (x_1, x_2, x_3, .., x_m)$, where m is the dimension of the space.
- S is a nonempty finite set of automaton states.
  Each cell can take on only one state at any one time from a set of states, $s \in S$, strict CA also requires state variables to be discrete.
- Neighborhood Template N, the state of any cell depends on the states and configurations of other cells in the neighborhood N of that cell. In two-dimensional space, there are two well-known templates, Von Neumann's (5 cells neighborhood), and Moore's (9 cells neighborhood).

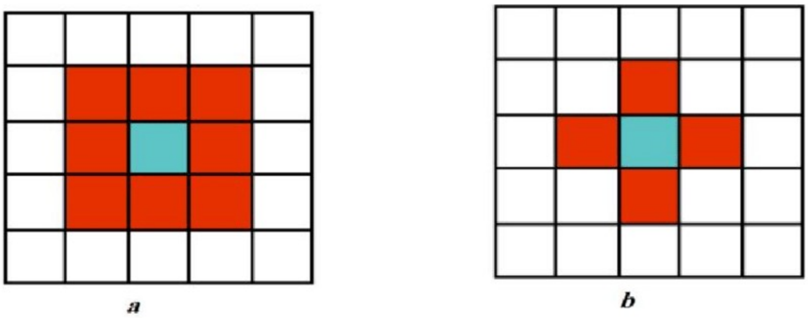

Fig. 2. a- Von Neumann's neighborhood, b- Moore's neighborhood

- f is a state transition function rule, $S_c^{t+1} = f(S_{N(c)^t}, S_c^t)$ The transition rule will take the previous state at the moment t of a cell S_c^t and the status of the neighborhood $S_{N(c)}^t$ as input and return S_c^{t+1} status at time $t + 1$.

3. Wildfire spread model

$$R = R(\theta, U, M, h, S_v)$$

3.1. Slope

Slope is a variable with a dramatic effect on fire propagation and as (4) explains.

Fires spreading in positive slopes ($\theta > 0$) are known to increase their rate of spread so we use a common slope correction factor (4) :

$$R = R(U, M, h, S_v)\; e^{0.069\theta}$$

R	the rate of fire spread on given slope
θ	the slope angle in degrees
$R(U, M, h, S_v)$	the calculated rate of fire spread for flat ground

For prediction of fire spread down slope ($\theta < 0$) , recent work has shown that applying the previous equation to negative angles grossly overestimates the effect of slope and thus underpredicts potential downslope rates of spread. A new model called kataburn (4) suggests that the negative slope correction factor should not exceed 0.5 of R :

$$R = R(U, M, h, S_v)\; \frac{e^{-0.069\theta}}{2e^{-0.069\theta} - 1}$$

3.2. **Fuels**

We present a simple empirical function according to the research shown in (6) for estimating no-slope fire spread rate in a generic foliar-based fuel bed under no-wind conditions.
Derived from laboratory trials and tested against data from outdoor fires in Mediterranean fuels. The purpose here is to develop simple and generic models for R , using the following approaches:
(i) formulation with M and h as independent variables; and (ii) an extension of that formulation, based on metrics of physical fuel properties, to expand applicability to a wider range of fuel bed characteristics.

$$R = R(M, h, S_v) = a\ M^b\ h^c\ ln(dS_m)$$

M	the moisture of the fuel bed in %
h	fuel bed height (m)
S_m	particles surface area-to-mass ratio that can be obtained from $S_m = \frac{S_v}{\rho_p}$
S_v	fuel particle surface area-to-volume ratio
ρ_p	fuel particle density

3.3. **Wind**

Using Rothermel's model (5) (which dont take wind direction in consideration) we find:

$$R = R_0(1 + \varphi_w + \varphi_s)$$

according to (4) Slope factor φ_s is based on an evaluation of experimental data, and is a function of slope steepness $tan(\theta)$ and packing ratio β:

$$\varphi_s = 5.275\beta^{-0.3}tan(\theta)^2$$

and

$$\varphi_w = aU^b$$

U	the speed of wind
a, b	constants

we suppose that $\varphi_s = 0$ (as if we are calculating the spread on a flat land where $\theta = 0$), so that what remains is pure influence of wind :

$$R_w = (1 + aU^b)$$

3.4. **Result**

with taking in consideration all the factors mentioned above we find :

$$R = R(\theta, U, M, h, S_v) = M^b h^c ln(dS_m) \times R_\theta \times (1 + aU^b)$$

where :

$$R_\theta = \begin{cases} e^{0.069\theta}, & \text{if } \theta \geq 0 \\ \frac{e^{-0.069\theta}}{2e^{-0.069\theta}-1}, & \text{if } \theta < 0 \end{cases}$$

4. Simulation

The simulation is based on a Cellular automata model with Moore's neighborhood where each cell can have one of the following states :

- Burnable (vegetation)
- Not burnable (water or soil)
- Ignited (fire starting)
- On fire
- Extinguishing
- Fully extinguished

The state cycle of burnable cell touched by the flames is the following :

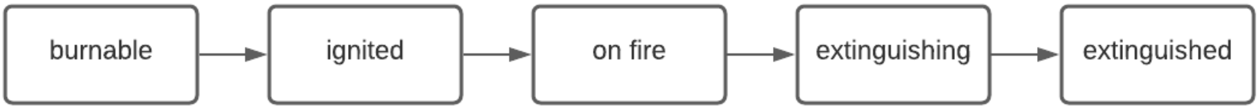

Fig. 3. state cycle of a burnable cell

a burnable cell is said to be touched by the flames when :

- At least One of the neighbouring cells is on fire.
- The calculated fire propagation rate R toward the cell is bigger than a certain threshold, depending on the rate R the future state of the cell is decided (either ignited or on fire).

---

for each cell c **do**
  for each neighbour n of c **do**
    if n state is burnable **then**
      calculate $R_{c,n}$
      if $R_{c,n} > threshold_1$ **then**
        set the state of neighbour n to on fire
      else if $R_{c,n} > threshold_2$ **then**
        set the state of neighbour n to ignited
      end if
    end if
  end for
end for

The update rule is repeated until the simulation converges, the criteria of convergence is reached when each cell's future state is equal to the present state.

5. Satellite rasters

The satellite data comes in a form of rasters, a raster consists of a matrix of cells (or pixels) organized into rows and columns (or a grid) where each cell contains values representing information about a specific location, data stored in a raster format represents real-world phenomena such as surface reflectance, temperature, elevation, soil moisture and weather information. Some rasters have

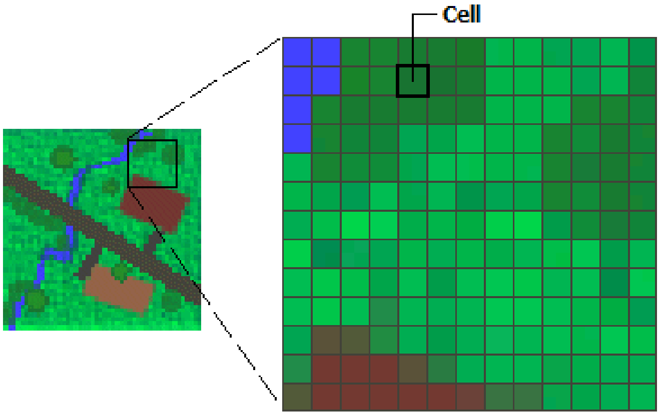

Fig. 4. example of a grided raster

a single band, or layer (a measure of a single characteristic), of data, while others have multiple bands.

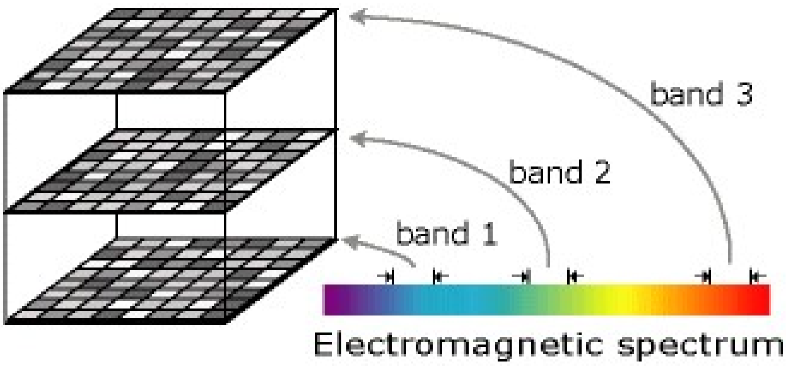

Fig. 5. raster with multiple bands

6. Wind data

we use The Global Forecast System (GFS) (9) dataset consisting of selected model outputs as gridded forecast variables. The 384-hour forecasts, with 3-hour forecast intervals, are made at 6-hour temporal resolution (i.e. updated four times daily), from this data set we use the (U,V) component of wind 10m above ground in order to know the direction and the speed of wind in each location.

7. Elevation data

In order to perform our simulation we need information about the slope of the terrain in each location, for this we use the SRTM datasets.

The Shuttle Radar Topography Mission (SRTM) datasets result from a collaborative effort by the National Aeronautics and Space Administration (NASA) and the National Geospatial-Intelligence Agency (NGA) (2).

SRTM collection have one band in its imagery which is land elevation in meters, this data is collected in swaths which extend from 30° off-nadir to 58° off-nadir from an altitude of 233 km, it consist of all land between 60° N and 56° S latitude. This accounts for about 80% of Earth's total landmass.

8. Vegetation grid

In order to run our simulation we need to specify the type of land in each location, whether it's a bush, a forest , a water body or bare soil. These are the main classes that we want all of the pixels in the raster to belong to. This is an unsupervised classification problem, and one of the most popular ways to address this problem in machine learning is by using the K-means clustering algorithm.

In our approach we use the USGS Landsat 8 Surface Reflectance Tier 1 (3), This dataset is the atmospherically corrected surface reflectance from the Landsat 8 OLI/TIRS sensors. These images contain 5 visible and near-infrared (VNIR) bands and 2 short-wave infrared (SWIR) bands processed to orthorectified surface reflectance, and two thermal infrared (TIR) bands processed to orthorectified brightness temperature.

Table 1. USGS Landsat 8 Surface Reflectance Tier 1 bands that we used

bands	*source*
B1	ultra blue surface reflectance
B2	blue surface reflectance
B3	green infrared surface reflectance
B4	red infrared surface reflectance
B5	near infrared surface reflectance
B6	shortwave infrared 1 surface reflectance
B7	shortwave infrared 2 surface reflectance
B10	brightness temperature

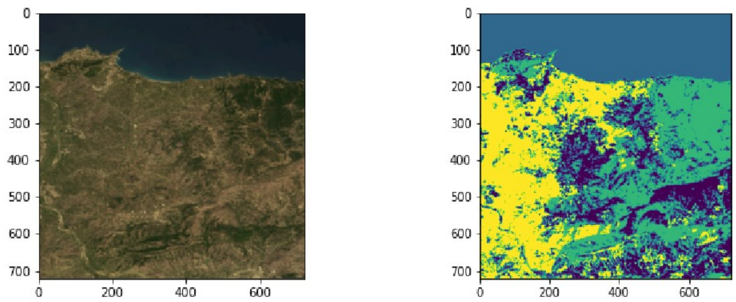

Fig. 6. applying kmeans algorithm $k = 4$

9. use case

Satellite imagery is widely available for any location in the world, our simulation can run on any piece of land worldwide, for a prototype we pick a 129 km^2 of the coast side between cap djinet and tigzirt (Algeria). By creating a 3 dimensional model for the topography of this coast side we are able to visualize the simulation running.

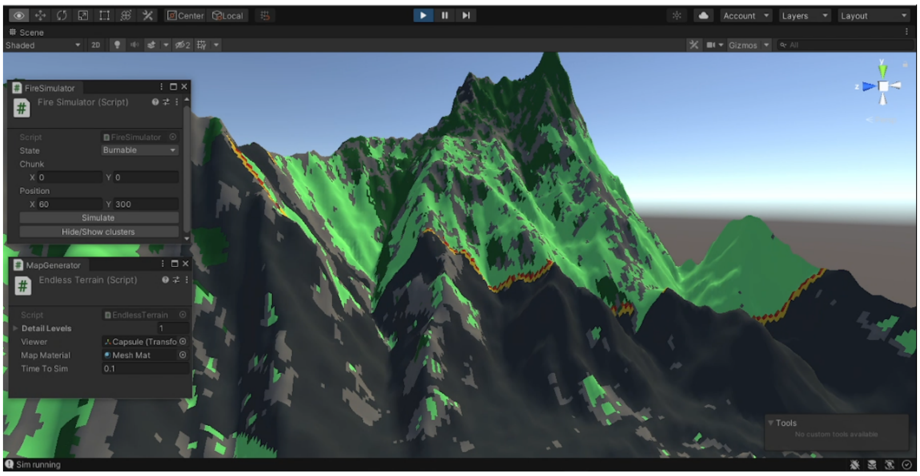

Fig. 7. fire spreading on the west side

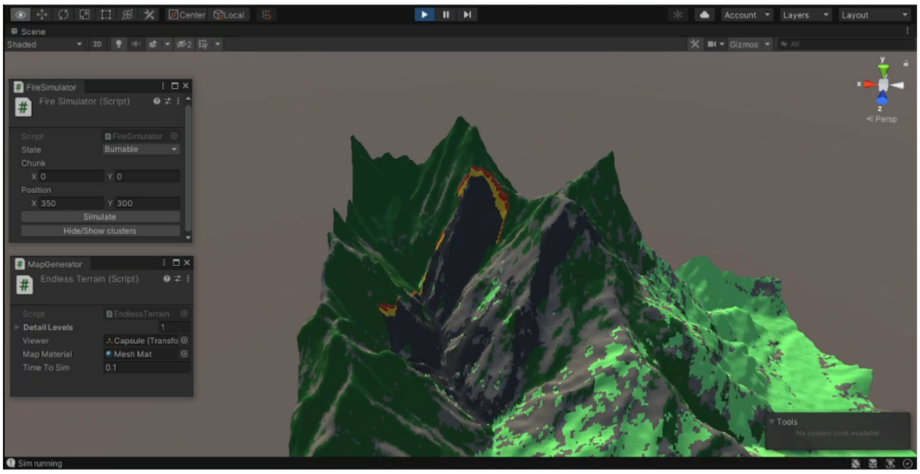

Fig. 8. fire spreading on the east side

10. impact

Our solution can be helpful in many aspects of fire management.

10.1. in time

planning fire fighting strategies in advance, which will enable fire crews position equipment on the ground in an optimal manner, minimizing damage and staying safe, planning strategies in advance will eliminate any possibilities of crews being trapped by fire.

10.2. before fire

risk factor calculation and this would help firefighters focus on areas with higher risk and develop better infrastructure as a precaution to gain more control over zones where fire can propagate uncontrollably in a devastating way.

11. Tools and technologies

11.1. Earth Engine

Earth Engine (10) is a platform for scientific analysis and visualization of geospatial datasets, for academic, non-profit, business and government users.
Earth Engine hosts satellite imagery and stores it in a public data archive that includes historical earth images going back more than forty years. The images, ingested on a daily basis, are then made available for global-scale data mining, Earth Engine also provides APIs and other tools to enable the analysis of large datasets.

11.2. Unity

Unity is a real-time 3D engine, it gives users the ability to construct and manipulate 3D objects, its mainly used for game development, but it can be extended for engineering usages and simulation purposes, the engine offers a primary scripting API in C#, we used it in order to create 3D meshes of real life terrains.

11.3. Programming languages

- Python : earth engine python api, numpy, pandas, sklearn, geemap, folium.
- C# : unity scripting api.
- Java script : earth engine code user interface.

References

[1] Forest fires spread modeling using cellular automata approach, Ljiljana Bodroži, Darko Stipaniev and Marijo Šeri, 2006.

[2] NASA EOSDIS Land Processes DAAC, “NASA Shuttle Radar Topography Mission Global 1 arc second”, November 20, 2013.

[3] Landsat Level- 2 Surface Reflectance Science Product courtesy of the U.S. Geological Survey.

[4] Miguel G. Cruz, James S. Gould, Martin E. Alexander, Andrew L. Sullivan, W. Lachlan McCaw and Stuart Matthews (2015): Empirical-based models for predicting head-fire rate of spread in Australian fuel types, Australian Forestry.

[5] Ralph M. Nelson, J : An effective wind speed for models of fire spread. International Journal of Wildland Fire, 2002, 11, 153–161.

[6] Rossa, C.G.; Fernandes, P.M. Empirical modelling of fire spread rate in no-wind and no-slope conditions. Forest Sci. 2018, 64, 358–370.

[7] Li Xiadong, W. Magill, Modeling fire spread under environmental influence using a cellular automation approach, Complexity International, 2001.

[8] https://mathworld.wolfram.com/topics/CellularAutomata.html

[9] https://www.emc.ncep.noaa.gov/emc/pages/numerical_forecast_systems/gfs.php

[10] Gorelick, N., Hancher, M., Dixon, M., Ilyushchenko, S., Thau, D. and Moore, R. (2017). Google Earth Engine: Planetary-scale geospatial analysis for everyone. Remote Sensing of Environment.