Summer 2007

A simulation environment modeling the use of wireless sensor networks for the detection and mapping of wildfires

Matthew Gann

A SIMULATION ENVIRONMENT MODELING THE USE OF WIRELESS SENSOR NETWORKS FOR THE DETECTION AND MAPPING OF WILDFIRES

by

MATTHEW GANN

A THESIS

Presented to the Faculty of the Graduate School of the

UNIVERSITY OF MISSOURI–ROLLA

In Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE IN COMPUTER ENGINEERING

2007

Approved by

Shoukat Ali, Advisor

Jagannathan Sarangapani

Ronald Joe Stanley

ABSTRACT

Over the past 5 years, billions of dollars have been spent fighting wildfires which consume thousands of acres. Currently, satellites are the primary source of information regarding the location of large scale wildfires. The satellites in use are capable of reporting the location of the wildfire, but these reports are generated infrequently and have a low spatial resolution, where the reported location of the fire can be off by 500 meters. Timely detection (latency) and frequent reports (throughput) are needed to help prevent the destruction caused by wildfires.

This thesis, a system is described based on an ad-hoc wireless sensor network to accurately detect and provide a real-time report of the wildfire location. The feasibility of using the system is shown by presenting a simulation environment that provides an estimate of the front and intensity of the fire.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor, Dr. Shoukat Ali, for his guidance and help throughout this research. I have deep appreciation for the advice and direction he gave me, and for his patience and encouragement. I owe my academic progress to his generous spirit.

I would also like to thank the other members of my committee, Dr. Jagannathan Sarangapani and Dr. Ronald Joe Stanley, for their insightful comments.

Last, but not least, I am forever indebted to my parents, Gary and Kellie. Their constant support and unconditional love have made all of this possible.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

1 INTRODUCTION

From 2000 to 2005, over half a million wildfires burned nearly 39.3 million acres in the United States alone. Suppressing the destruction caused by these fires cost federal agencies over seven billion dollars [1]. Along with preventative strategies, timely detection and frequent mapping of wildfires is critical in limiting the devastation caused. Wildfire mapping is the identification of the front of the fire. Dense smoke seriously limits visibility in large wildfires, making it almost impossible to identify the front of the fire. This complicates and jeopardizes suppression efforts.

Current methods of wildfire detection and mapping can produce updates from one to four times daily, creating an information poor environment for firefighters not only temporally but spatially as well. Satellites are the current method used by the USDA forest service for mapping large fires that span several thousand acres. Most satellite systems (AVHRR and Landsat) capable of detecting fire are able to give daily updated reports, but in 1999 NASA launched MODIS [2], which can give updates up to four times daily depending on the amount of fire activity present [3]. However, an update every 30-60 minutes is required for efficient deployment of firefighting resources [4]. In this research, we propose a system based on a wireless sensor network to allow mapping of the front of a wildfire in real-time. The proposed system is capable of measuring various environmental indicators of fire and processing the results in order to provide firefighters with useful, accurate, and timely information about the location and characteristics of the fire.

The core of the approach presented in this research is the deployment of many sensor nodes to areas where the front of the fire is likely to be. These nodes, which are small and inexpensive, can contain a number of sensors that measure environmental indicators of fire such as infrared radiation, temperature, humidity, and smoke. Each node will communicate wirelessly to other nodes in the network in an ad-hoc manner. They will collaborate to determine location and intensity of the fireline. The information collected can be used to guide the allocation of fire fighting resources.

Wireless sensor networks are composed of many small, inexpensive nodes, each equipped with devices for communication, data processing, and sensing. The manufacturing of low-cost, low-power sensor nodes has been made possible by advances in micro-electro-mechanical systems, wireless communication, and digital electronics. Sensor nodes are capable of wirelessly collaborating with one another in order to monitor the surrounding environment for some desired phenomenon. Wireless sensor networks have proven to be useful in many military and civilian applications. Some examples of these applications include environmental monitoring, target surveillance, industrial observations, and tactical systems ([5], [6], [7], [8]).

In this thesis a simulation environment is presented for modeling the use of wireless senor networks for detection and mapping of wildfire. Based on the simulation results it is shown that wireless sensor networks can be used to accurately map the front of a variety of fires. In addition, a map showing the intensities found in different areas of a given fire is generated.

The rest of this thesis is organized as follows. Section 2 gives an overview of the current methods used to map large wildfires. In Section 3 an explanation of the system model used in this research is described. This section also includes the mathematical model of the radiation received by a sensor and how an estimate of the location of the front of the fire is generated. Section 4 shows the feasibility of using wireless sensor networks for the mapping of wildfires. Section 5 concludes the thesis. Please refer to Table 3.1 for a concise explanation of the mathematical notation used throughout this thesis.

2 RELATED WORK

Currently, satellites have a major role in the detection and monitoring of wildfire. These satellite systems are capable of providing information about the location of active fires and areas that may be susceptible to wildfire outbreaks [2]. Their two major drawbacks are infrequent updates and low resolution reports where the location of the fire can be off by 500 meters.

The Advanced Very High Resolution Radiometer (AVHRR) [9] found on the National Oceanic and Atmospheric Administration's (NOAA) Polar Orbiting Environmental Satellite (POES) is used to collect radiance data from the earth. Its main use is to provide meteorological data for such things as investigation of clouds, temperatures of radiating surfaces, snow and ice, and a number of other atmospheric conditions. The AVHRR is a 4 to 5 channel broad-band sensor system capable of sensing the visible, near-infrared, and thermal infrared portions of the electromagnetic spectrum. The AVHRR system uses the visible bands to detect smoke plumes from fires as well as burn scars. It uses the middle-infrared band to detect hotspots and active fires [2]. One downfall of the AVHRR system is that it easily confuses active fires with heated surfaces making it more suited for detecting fires at night.

The Geostationary Operational Environmental Satellites (GOES) system [10] is primarily used for the detection and monitoring of the smoke associated wildfires, prescribed burns, or deforestation in the Western Hemisphere only. It makes use of two different algorithms: Automated Biomass Burning Algorithm (ABBA) and Automated Smoke/Aerosol Detection Algorithm (ASADA). Estimates of sub-pixel area and mean temperature of the fires are provided by ABBA. It reports the longitude/latitude location of the fire, estimates of the fire size, and fire temperatures. The results are generally available within 90 minutes (latency). The ASADA algorithm provides a summary of the extent of smoke/aresol coverage along with an estimate of the amount of radiation that is reflected off the smoke to give a rough idea of the smoke intensity. The GOES satellites are able to acquire images every 15 to 30 minutes (throughput) [2].

The Landsat program, which has been in operation for 30 years, [11] uses a series of earth-orbiting satellites to take specialized digital pictures of the earth's continents and surrounding coastal regions. The satellites are equipped with sensors that record reflected and emitted energy from the earth in the visible, near-infrared, mid-infrared, and thermal-infrared spectrums. Landsat collects data from the earth's surface to help aid in the assessment of damage due to fires, floods, and tsunamis. Landsat can provide very high resolution images but at a very infrequent rate since it only revisits a particular area every 14 days [2].

The Terra and Aqua satellites are equipped with the Moderate Resolution Imaging Spectroradiometer or MODIS [12]. These satellites view the earth's surface every 1 to 2 days in 36 different spectral bands. This data is used to improve the understanding of current global dynamics and processes occurring on land, in the ocean, and in the lower atmosphere. The MODIS fire and thermal anomalies project [2] records information on the occurrence of fires to be used for improved fire and forest management. The goal of this project is to provide a temporal and spatial distribution of fires along with such statistics as the energy emitted from the fire, both day and night. Fire locations on a MODIS map are represented as the real-time geographic locations of the centers of 1km resolution pixels [3]. The center of a pixel can be off by 500 meters of the location shown on a MODIS map. A pixel representing a fire does not mean the whole area of the pixel is on fire. It can mean there is a very hot fire spanning a small area or a cool fire spanning over a large area. Currently, MODIS has no way of discriminating between the two scenarios. MODIS can produce a map at least once daily, but in the case of significant fire activity maps are produced 4 times daily.

3 SYSTEM MODEL

This section formally defines the problem of developing a simulation environment for detecting and mapping a large-scale wildfires using a wireless sensor network. Assume there exists a wireless sensor network with a set of sensors $\mathbf{S}$ with elements s_k where $k = 1, ..., |\mathbf{S}|$. Each sensor has a position $X_k = (x_{s_k}, y_{s_k})$, where x_{s_k} is the x-coordinate of s_k and y_{s_k} is the y-coordinate. It is assumed that each sensor is equipped with one isotropic infrared (IR) detector which allows the sensor to receive IR radiation from every direction. It is also assumed that sensor locations are static and known (see [13–19] for some localization schemes). A glossary of the notation used in this thesis is given in Table 3.1.

3.1 CREATING A SIMULATED FIRELINE

Let F_t be defined as the 'fireline' which is described as the location, height, and intensity of each point on the front of the fire at time t. The FARSITE Fire Area Simulator [20] is used to model the spread of the fire, and the location of the fireline F_t at time t. FARSITE is a fire growth modeling technology intended for use on personal computers [21]. It uses a wave-front approach based on Huygens' principle of wave propagation, to model the spread of fire over complex landscapes. The fire is simulated as a 2-dimensional elliptical wave using spatial data from a geographic information system. The forefront of the fire is then projected over a finite time step using a fire behavior model at discrete locations along the edge of the fire. Information about the fire, such as flame length and intensity, is computed at these points from the information on fuels, weather, and topography using the Rothermel model [22].

The inputs required by the FARSITE simulator are a landscape, additional weather and wind data, and the location of ignition points. A landscape should contain information

Table 3.1. Glossary of notation.

S	The set of all sensors in the network.
s_k	The kth sensor in set **S**.
X_k	The actual location of sensor s_k. In $\mathcal{R}^2$ it is the 2 tuple (x_{s_k}, y_{s_k}).
A_f	The area of the grid where the fire can spread.
$I_{s_k}(t)$	The intensity seen at sensor s_k at time t.
F	The set of the actual firelines for all time t.
F_t	The actual fireline as described by the FARSITE output at time t.
$F_i(t)$	The ith point of the actual fireline at time t at location $(x_{F_i}(t), y_{F_i}(t))$.
$H_{F_i}(t)$	The height of the ith point of the actual fireline at time t.
$I_{F_i}(t)$	The intensity of the ith point of the actual fireline at time t.
E	The set of the FLSE's for each time t.
$E_j(t)$	The jth FLSE at time t at location $(x_{E_j}(t), y_{E_j}(t))$.
$I_{s_{kj}}(t)$	The intensity seen at sensor s_k at time t from FLSE j.
$H_{E_j}(t)$	The height of the jth FLSE at time t.
$W_{E_j}(t)$	The width of the jth FLSE at time t.
$T_{E_j}(t)$	The temperature of the jth FLSE at time t.
$A_{E_j}(t)$	The area of the jth FLSE at time t.
$I_{E_j}(t)$	The intensity of the jth FLSE at time t.
$\hat{F_t}$	The estimated fireline at time t.
$T_{s_k}(t)$	The temperature of the sensor located at (x, y) at time t.
σ	The Stefan-Boltzman constant equal to $5.67 \times 10^{-8}\ \frac{W}{m^2K^4}$
ϵ_E	The emissivity of an FLSE.
ϵ_s	The emissivity of a sensor.
T_S	The number of time steps in the FARSITE simulation.

such as the elevation above sea level, the rate of change in elevation, the type of fuel present in the area, etc. This information is stored in a raster table. The weather input contains the temperature, humidity, and precipitation for the different time steps of the simulation. Similarly, the wind input describes the speed and direction of the wind for each time step in the simulation. Both wind and weather conditions apply to the entire simulation region. The fuel input should contain information about the burn rate adjustments and initial moisture of the fuels in the landscape. The information provided by the five inputs to the FARSITE simulator discussed above will now be examined. In Section 4, the fires produced by the FARSITE simulations will be discussed.

3.1.1 Landscape (.LCP). The landscape file is a compilation of many GIS ASCII raster source files. It is built using 5 mandatory file themes that completely describe the geographical region. There are also 5 optional file themes which can be used to help determine exactly what is included in the region. Table 3.2 lists the file themes that can be used to generate a landscape file. When running the simulations only the 5 required file themes to generate the landscape file were used. The 5 required GIS ASCII files needed for the generation of the landscape file are described below.

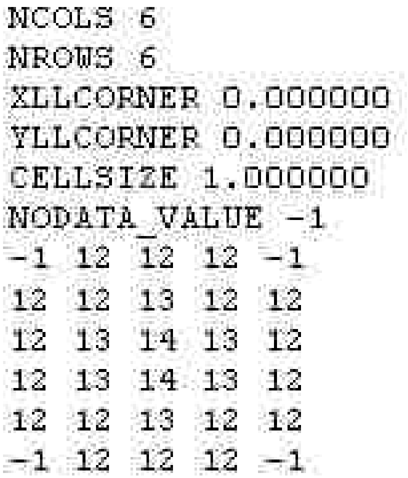
- Elevation - Height of the terrain above sea level (meters).
- Slope - Rate of change in elevation (degrees up from horizontal).
- Aspect - Direction that a surface faces (degrees CW from north).
- Fuel Model - Type of fuel present in a specific area.
- Canopy - Percentage of the fuel that is in the canopy of the terrain.

Table 3.2. ASCII data files that can be used to generate a landscape file.

File Theme	Required	Default Units	Alternate Units
elevation	yes	$meters$	$feet$
slope	yes	$degrees$	$percent$
aspect	yes	categories 1-25	$degrees$
fuel model	yes	13 NFFL models or expanded set of 40 models	custom or converted models
canopy cover	yes	categories 1-4	$percent$
tree height	no	$meters * 10$	$meters, feet, feet * 10$
crown base height	no	$meters * 10$	$meters, feet, feet * 10$
crown bulk density	no	$kg/m^3 * 100$	kg/m^3, lbs/ft^3,$lbs/ft^3 * 100$
duff loading	no	Mg/ha	$tons/acre$
coarse woody	no		

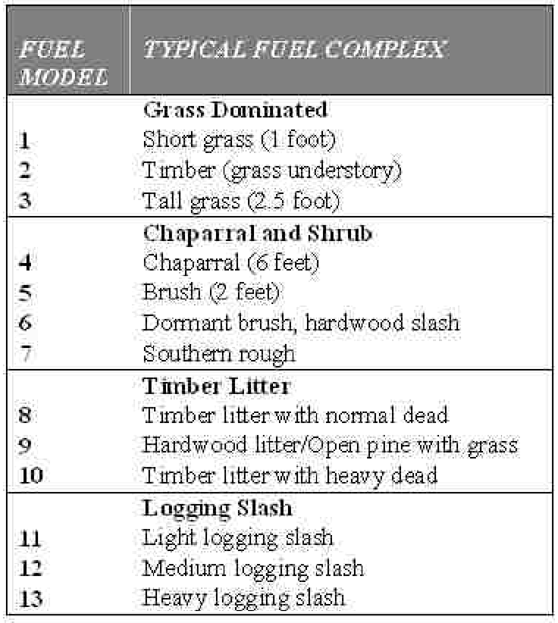
All the GIS ASCII files are described in the GRID file format shown in Figure 3.1.The top 6 rows make up a header for the file that help the program opening the file to determine what the data in the file represents. For instance, the example in Figure 3.1 sets up a 6x6 grid of data with the lower left corner for both x and y being located at (0,0) on a global grid. Each cell is 1 unit square and the cells containing the value -1 represent cells which have no data. The headers for each file must be identical so the data in the different ASCII files will lay on top of each other perfectly.

All the units for the GIS data are common except for the fuel model. In order for FARSITE to accurately simulate the behavior of a fuel, many different types of fuels were studied and categorized based on some standardized parameters. Therefore a specific type of fuel can be modeled by changing the values of each parameter to match the behavior of the actual fuel. There are 13 standardized models of fuels developed by the Northern Forest Fire Laboratory (NFFL). These fuel models are shown in Figure 3.2 [23].



```
NCOLS 6
NROWS 6
XLLCORNER 0.000000
YLLCORNER 0.000000
CELLSIZE 1.000000
NODATA_VALUE -1
-1 12 12 12 -1
12 12 13 12 12
12 13 14 13 12
12 13 14 13 12
12 12 13 12 12
-1 12 12 12 -1
```

Figure 3.1. Information about the landscape file.



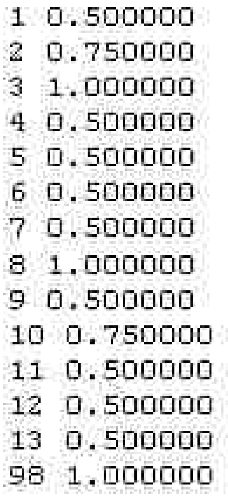
FUEL MODEL	TYPICAL FUEL COMPLEX
	Grass Dominated
1	Short grass (1 foot)
2	Timber (grass understory)
3	Tall grass (2.5 foot)
	Chaparral and Shrub
4	Chaparral (6 feet)
5	Brush (2 feet)
6	Dormant brush, hardwood slash
7	Southern rough
	Timber Litter
8	Timber litter with normal dead
9	Hardwood litter/Open pine with grass
10	Timber litter with heavy dead
	Logging Slash
11	Light logging slash
12	Medium logging slash
13	Heavy logging slash

Figure 3.2. NFFL fuel models.

3.1.2 Burn Rate Adjustments (.ADJ). This adjustments file allows the user to tweak how fast the fuels specified in the landscape fuel model burn. A value for the burn rate is associated with each fuel model used. The .ADJ file is a table of fuel models a floating point multiplier used to change the value of the burn rate. Only the fuel model numbers that are referenced in the landscape file need to be included in the adjustment file. The format is *Fuel Model*, *Floating Point Multiplier*. In the example adjustment file shown in Figure 3.3 it can be deduced that fuel types 3, 8, and 98 all burn at the default speed and all other fuel types burn at 75% or 50% of the default speed.

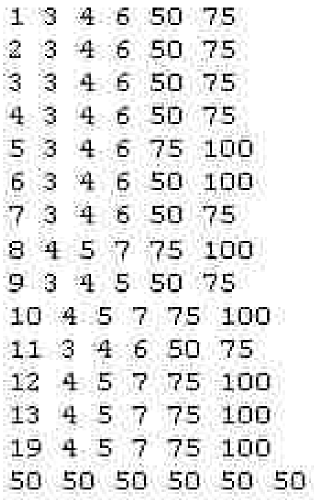
3.1.3 Initial Moistures (.FMS). The initial moistures file allows the user to set the initial moisture content contained in the fuel. It uses an ASCII integer file like the one shown in Figure 3.4. The format of the file is *Fuel Model*, *1 hour*, *10 hour*, *100 hour*, *Live Herbaceous*, *Live Woody*. The values in each column are the fuel moistures in percent.

3.1.4 Weather (.WTR). The weather file contains a daily log of temperature, humidity, and precipitation that helps to describe the weather conditions present during the



```
1 0.500000
2 0.750000
3 1.000000
4 0.500000
5 0.500000
6 0.500000
7 0.500000
8 1.000000
9 0.500000
10 0.750000
11 0.500000
12 0.500000
13 0.500000
98 1.000000
```

Figure 3.3. Adjustments file.

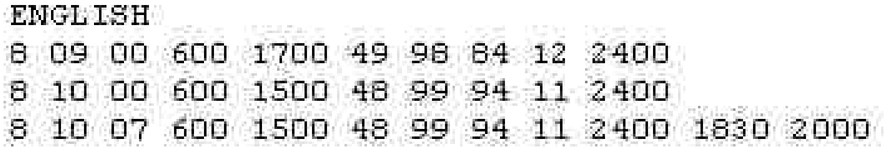


```
1 3 4 6 50 75
2 3 4 6 50 75
3 3 4 6 50 75
4 3 4 6 50 75
5 3 4 6 75 100
6 3 4 6 50 100
7 3 4 6 50 75
8 4 5 7 75 100
9 3 4 5 50 75
10 4 5 7 75 100
11 3 4 6 50 75
12 4 5 7 75 100
13 4 5 7 75 100
19 4 5 7 75 100
50 50 50 50 50 50
```

Figure 3.4. Moistures file.

simulation. The .WTR file uses the following format: *Month*, *Day*, *Precipitation*, *Hour 1*, *Hour 2*, *Temp 1*, *Temp 2*, *Humidity 1*, *Humidity 2*, *Elevation*, *rt1*, *rt2*. *Precipitation* is the daily rain amount specified with an integer in hundredths of an inch or millimeters. *Hour*

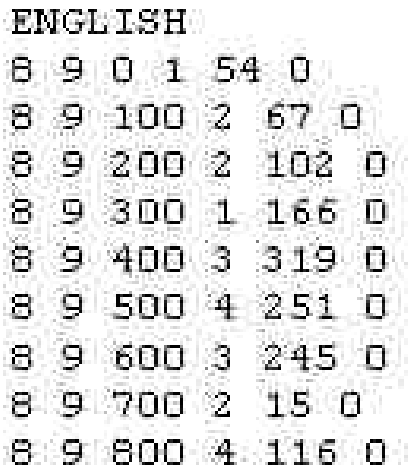
1 corresponds to the hour at which the minimum temperature was recorded in the range of hour 0 to 2400. *Hour 2* corresponds to the hour at which the maximum temperature was recorded in the range of hour 0 to 2400. *Temp 1* (minimum) and *Temp 2* (maximum) is an integer giving the temperatures in degrees Fahrenheit or Celsius. *Humidity 1* and *Humididy 2* represent the minimum and maximum humidities respectively and they are represented by integers in the range of 0 to 99 percent. *Elevation* is the height above sea level in feet or meters. The precipitation duration begins at *rt1* and ends at *rt2* and both *rt1* and *rt2* are in the range of hour 0 to 2400. An example Weather file can be found in Figure 3.5.



```
ENGLISH
8 09 00 600 1700 49 98 84 12 2400
8 10 00 600 1500 48 99 94 11 2400
8 10 07 600 1500 48 99 94 11 2400 1830 2000
```

Figure 3.5. Weather file.

3.1.5 Wind (.WND). The wind file describes the wind conditions for the whole region during the simulation. The time at which the wind is blowing, the speed of the wind, the direction of the wind, and the percentage of cloud cover are described in the file in the following format: *Month*, *Day*, *Hour*, *Wind Speed*, *Wind Direction*, *Cloud Cover*. An example wind file is shown in Figure 3.6 The wind has a uniform speed and direction over the whole simulation region unless the direction is specified as upslope (-1) or downslope (-2). If this is the case, then the slope and topography of the landscape file determines the direction of the wind (but not the speed). The direction is specified as in integer representing the degrees clockwise from north. The cloud cover is an integer in the range from 0 to 100.



```
ENGLISH
8 9 0 1 54 0
8 9 100 2 67 0
8 9 200 2 102 0
8 9 300 1 166 0
8 9 400 3 319 0
8 9 500 4 251 0
8 9 600 3 245 0
8 9 700 2 15 0
8 9 800 4 116 0
```

Figure 3.6. Wind file.

3.1.6 Making Use of the FARSITE Output. The output of FARSITE is a detailed description of the set of points that are on fire, containing the x and y coordinates, the current time, the intensity, and the flame length for each point that is on fire. Other fields may be included in the output as well, however only those listed above are required. To make use of the information in the FARSITE output, each time step of the simulation is broken. For the purpose of determining the amount of radiation received by a sensor at a given time, the FARSITE fireline F_t, is converted into a set of contiguous rectangular elements (see Figure 3.7). Each one of these elements will be denoted as a fireline surface element or FLSE and the set of FLSEs approximating the FARSITE fireline F_t at time t is defined **E**. Each FLSE in **E** is characterized by its height $H_{E_j}(t)$, width $W_{E_j}(t)$, intensity $I_{E_j}(t)$, and location $(x_{E_j}(t), y_{E_j}(t))$ (see Figure 3.8).

FLSEs are constructed as follows. For each set of consecutive points in the FARSITE output, $(x_{F_i}(t), y_{F_i}(t))$ and $(x_{F_{i+1}}(t), y_{F_{i+1}}(t))$, the FLSE height $H_{E_j}(t)$ is the average of $H_{F_i}(t)$ and $H_{F_{i+1}}(t)$. Similarly, the FLSE width $W_{E_j}(t)$ is the distance between

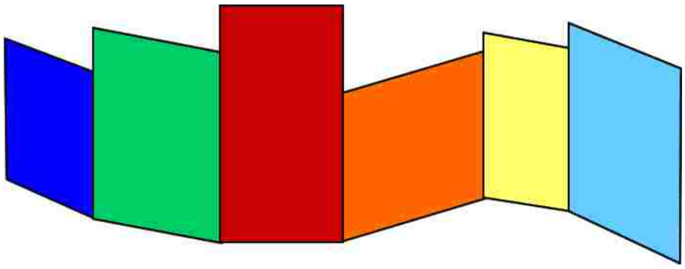


Figure 3.7. An example of several contiguous rectangular elements in three dimensions.

$(x_{F_i}(t), y_{F_i}(t))$ and $(x_{F_{i+1}}(t), y_{F_{i+1}}(t))$, the FLSE's position (x_{E_i}, y_{E_i}) is calculated by taking the midpoint of $(x_{F_i}(t), y_{F_i}(t))$ and $(x_{F_{i+1}}(t), y_{F_{i+1}}(t))$, and the FLSE's intensity $I_{E_j}(t)$ is calculated by averaging $I_{F_i}(t)$ and $I_{F_{i+1}}(t)$. The information for each FLSE is then stored in **E**. Please refer to Figure 3.8 for a pictorial representation of this process.

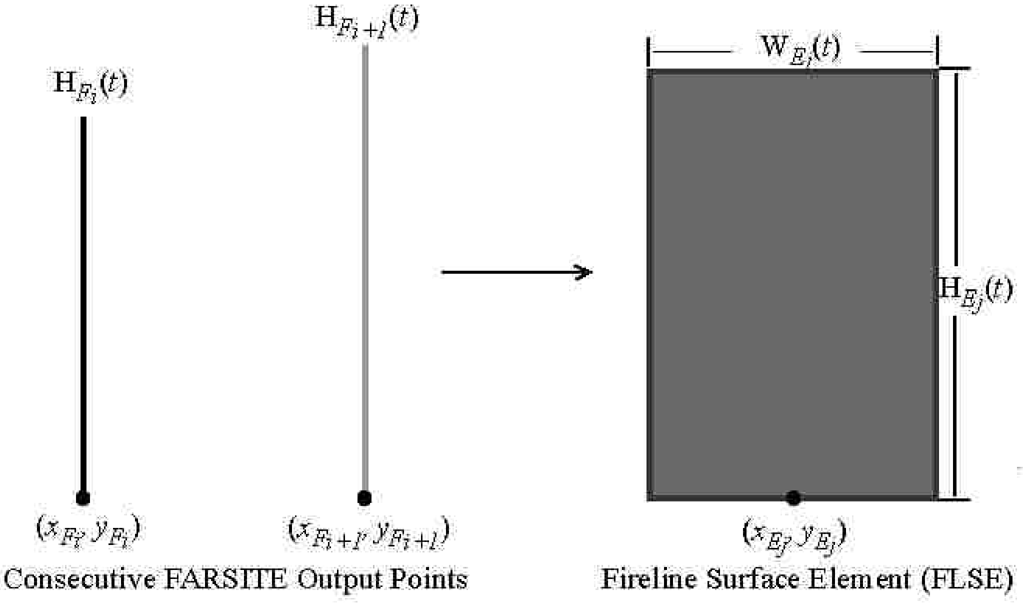


Figure 3.8. Model of and FLSE created from two consecutive points from the FARSITE output. The intensities of each point and the FLSE is represented by its color.

Once each FLSE has been defined, all the FLSE's are combined for a given time step t to form the complete fireline (Figure 3.9). Now the process for determining the intensity seen at an arbitrary sensor s_k is examined.

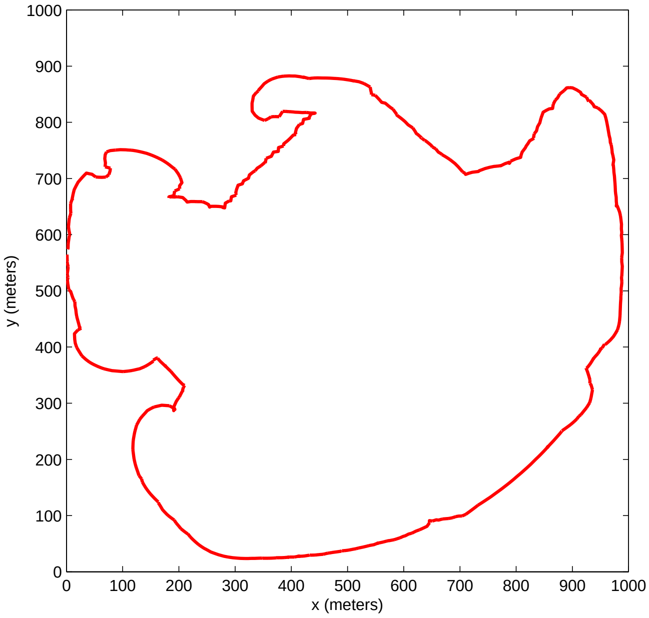


Figure 3.9. A 2D projection of the 3D fireline constructed from a set of FLSEs.

3.2 MODELING THE INTENSITY RECEIVED AT A SENSOR

After modeling the fireline as a number of rectangular surfaces, the next step is to model the radiation received by a sensor at time t. Each sensor is modeled as a surface so the radiation transfer between each FLSE and the sensor can be modeled by the following general equation for the transfer of radiation between two surfaces [24].

$$R = \sigma\epsilon_1\epsilon_2(T_1^4 - T_2^4)\left[\frac{A_1\cos\theta_1 \times A_2\cos\theta_2}{\pi(D)^2}\right] \tag{3.1}$$

In Equation 3.1, R is the radiation absorbed by surface 1 and radiated from surface 2, σ is the Stefan-Boltzman constant, ϵ_1 and ϵ_2 are the emissivities of surface 1 and surface 2, respectively, T_1 and T_2 are the temperatures of the two surfaces, A_1 and A_2 are the areas of the two surfaces, θ_1 and θ_2 are the angles between the surface normal vectors and the line of sight joining the two surfaces, and D is the distance between the two surfaces.

For this application, it is assumed the sensor and the FLSE are always on level ground. Similarly, it is assumed the sensor is much smaller than the FLSE's making up the fireline so the sensor is modeled as a point with an area of 1. Since the sensor is on level ground and has a height that is small in comparison with an FLSE, the z component of the sensor can be ignored. The z component of the FLSE however, is retained because it will be necessary to integrate with respect to later in this section. The distance between surface 1 and surface 2 can now be written as

$$D = \sqrt{(x_{s_k} - x_{E_j}(t))^2 + (y_{s_k} - y_{E_j}(t))^2 + z_{E_j}(t))^2} \tag{3.2}$$

The model used for this research assumes that IR sensors are isotropic, so they can detect radiation from any direction. Therefore, the normal of the sensor (n_s) is set in the direction of the line of sight connecting the sensor and the FLSE. Also, it is assumed the normal of the FLSE (n_E) always points directly at the sensor (Figure 3.10). By applying these two assumptions, $\cos(\theta_1)$ and $\cos(\theta_2)$ both go to 1. Equation 3.1 can now be rewritten by plugging Equation 3.2 in for D.

$$\begin{gathered} R = \sigma\epsilon_E\epsilon_s(T_{E_j}(t)^4 - T_{s_k}(t)^4)\times \\ \left[\frac{A_f}{\pi[(x_{s_k} - x_{E_j}(t))^2 + (y_{s_k} - y_{E_j}(t))^2 + z_{E_j}(t))^2]}\right] \end{gathered} \tag{3.3}$$

To find the intensity seen at a sensor, the total radiation leaving a single FLSE that will be received by the sensor must be determined. This is done by integrating Equation 3.3

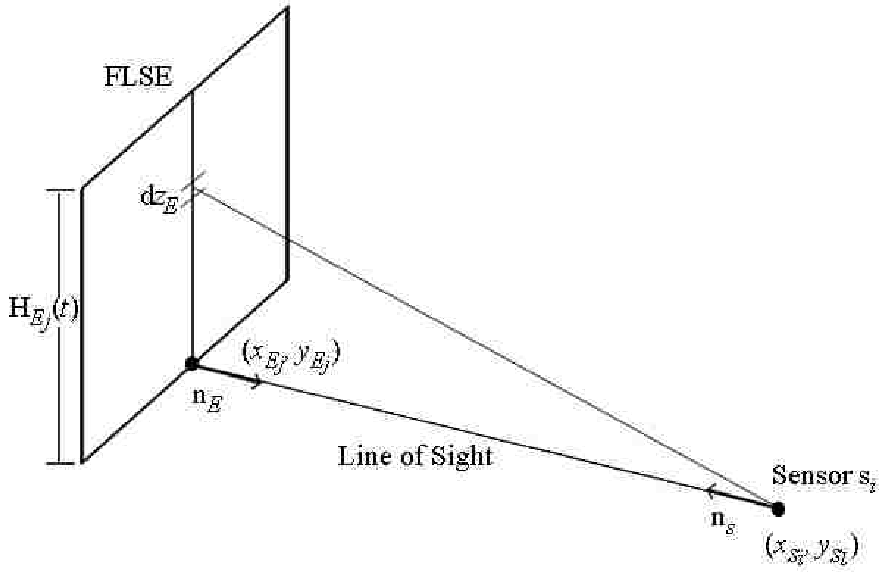


Figure 3.10. Example of how radiation transfers from a fire surface to a sensor.

along the center line of the FLSE with respect to $z_{E_j}(t)$. The integral is now setup to find the intensity received by a sensor from the FLSE located at position $(x_{E_j}(t), y_{E_j}(t))$ and integrate the FLSE from 0 to its height $H_{E_j}(t)$.

$$I_{s_{kj}}(t) = \int_{0}^{H_{E_j}(t)} \sigma \epsilon_E \epsilon_s (T_{E_j}(t)^4 - T_{s_k}(t)^4) \times \left[\frac{A_f}{\pi[(x_{s_k} - x_{E_j}(t))^2 + (y_{s_k} - y_{E_j}(t))^2 + z_{E_j}(t))^2]}\right] dz_{E_j}(t) \tag{3.4}$$

To simplify the calculation of Equation 3.4 the following equation from [25] is used.

$$\int \frac{du}{a^2 + u^2} = \frac{1}{a} \tan^{-1}\left(\frac{u}{a}\right) \tag{3.5}$$

$$\text{such that } a = \sqrt{(x_{s_k} - x_{E_j}(t))^2 + (y_{s_k} - y_{E_j}(t))^2} \tag{3.6}$$

$$\text{and } u = z_{E_j}(t) \tag{3.7}$$

The equation defining the intensity $I_{s_{kj}}(t)$ seen by an arbitrary sensor s_k from a single FLSE at location $(x_{E_j}(t), y_{E_j}(t))$ and time t is found by calculating the integral in Equation 3.4 using Equations 3.5 through 3.7 and evaluating from 0 to $H_{E_j}(t)$.

$$\begin{aligned} I_{s_{kj}}(t) = {} & \frac{\sigma \epsilon_E \epsilon_s A_f (T_{E_j}(t)^4 - T_{s_k}(t)^4)}{\pi \sqrt{(x_{s_k} - x_{E_j}(t))^2 + (y_{s_k} - y_{E_j}(t))^2}} \times \\ & \tan^{-1}\left(\frac{H_{E_j}(t)}{\sqrt{(x_{s_k} - x_{E_j}(t))^2 + (y_{s_k} - y_{E_j}(t))^2}} \right) \end{aligned} \tag{3.8}$$

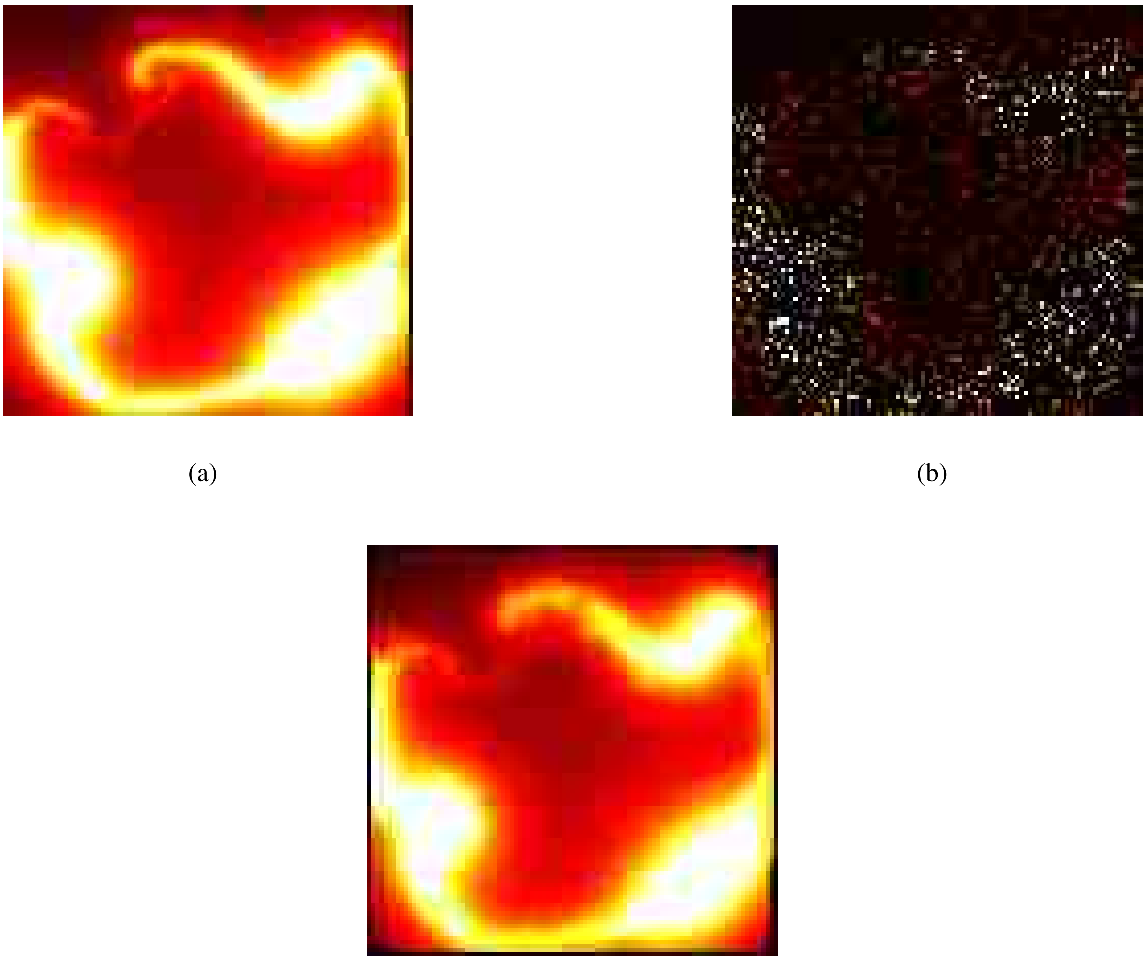
To find the total intensity $I_{s_k}(t)$ seen by a sensor from the entire fireline F_t, equation 3.8 is summed for each FLSE in **E**.

$$\begin{aligned} I_{s_k}(t) = {} & \sum_{j=1}^{|\mathbf{E}|} \frac{\sigma \epsilon_E \epsilon_s A_{E_j}(t) (T_{E_j}(t)^4 - T_{s_k}(t)^4)}{\pi \sqrt{(x_{s_k} - x_{E_j}(t))^2 + (y_{s_k} - y_{E_j}(t))^2}} \times \\ & \tan^{-1}\left(\frac{H_{E_j}(t)}{\sqrt{(x_{s_k} - x_{E_j}(t))^2 + (y_{s_k} - y_{E_j}(t))^2}} \right) \end{aligned} \tag{3.9}$$

3.3 RECREATING THE FIRELINE FROM SENSOR READINGS

To begin, the total intensity at every sensor is found as describe in the previous section which will be used to produce an image similar to the ones shown in Figure 3.11. Consider Figure 3.11(b) where the lighter points in the figure are the ‘hottest’ sensors, the darker points are the ‘cooler’ sensors, and the black areas are the points where no sensors were located. The first step towards recreating a fireline is to estimate intensities for all points in this figure where data is missing. By doing so, image processing techniques can be used to estimate the fireline.

Figure 3.11(b) is composed of a number of pixels, and some of these pixels encompass actual sensors and therefore the intensity of the fire is known at these points. Pixels that do not cover sensors have unknown intensities. To estimate the intensity of pixels that cover no sensors, 2D interpolation is used, which makes use of triangle-based linear interpolation based on the Delaunay triangulation of existing data [26].



(a)

(b)

(c)

Figure 3.11. A graphical representation showing the known intensities before and after performing 2D linear interpolation. (a) A map showing the intensity of 10,000 sensors placed on a grid. (b) A map showing the actual intensities of 900 sensors. (c) Intensity map after performing 2D linear interpolation on the data from 900 sensors.

Performing the interpolation on the sparse sensor data given in Figure 3.11(b) produces the intensity map similar to the one shown in Figure 3.11(c). The next step is to estimate the actual fireline. Ideally, the estimated fireline $\hat{F}_t$ should be identical to the actual fireline F_t shown in Figure 3.9. By looking at Figure 3.11(a) or Figure 3.11(c), one can form a very rough idea of where the fireline might be by following the general shape of the intensities. However, a better estimate is needed because trying to estimate the fireline by eye could lead to an estimate that is hundreds of meters off. To better realize how this data can be used to estimate the fireline the data is visualized in 3-dimensions, where the fire intensity is plotted as height. That is the x and y axis of the plot represent the location of the intensity at each point and the z axis represents the amount of intensity seen at each point. Graphically, the intensity map in Figure 3.11(c) is converted to the 'mountain' in Figure 3.12. An intuitive solution to finding the fireline using this image is to use the ridge of the mountain as an initial estimate of F_t.

By allowing the numerical value of each pixel in Figure 3.11(c) to represent the elevation of the same point in Figure 3.12, areas such as minima, catchment basins, and watersheds can be more easily defined on an intensity map. A minima M is defined as a pixel in an image I that is surrounded by a number of other pixels for which it is impossible to reach a pixel of a lower altitude without having to climb first. The catchment basin $C(M)$ for a minima M is defined as the set of pixels for which a drop of water falling at any pixel in $C(M)$ will follow a descending path downstream and eventually reach M. The boundaries between different catchment basins in an image I are defined as a watershed. An efficient algorithm for defining the watersheds of a 2-dimensional gray scale image is given in [27]. This watershed detection algorithm was used on a scaled intensity map to determine all points located on the watershed of the mountain. For the method described in this research for finding the fireline, all intensities are scaled to 256 levels. This is done by linearly mapping each pixel in the intensity map using the following equation

$$Z'_i = 255 * \left(1 - \frac{Z_{\max} - z_i}{Z_{\max} - Z_{\min}}\right) \tag{3.10}$$

where Z'_i is the scaled down intensity of element i, z_i is the intensity of element i, and $Z_{\max}$ and $Z_{\min}$ are the maximum and minimum intensity in the original image. The watershed detection algorithm returns a set of points indicating the location of the watersheds. These

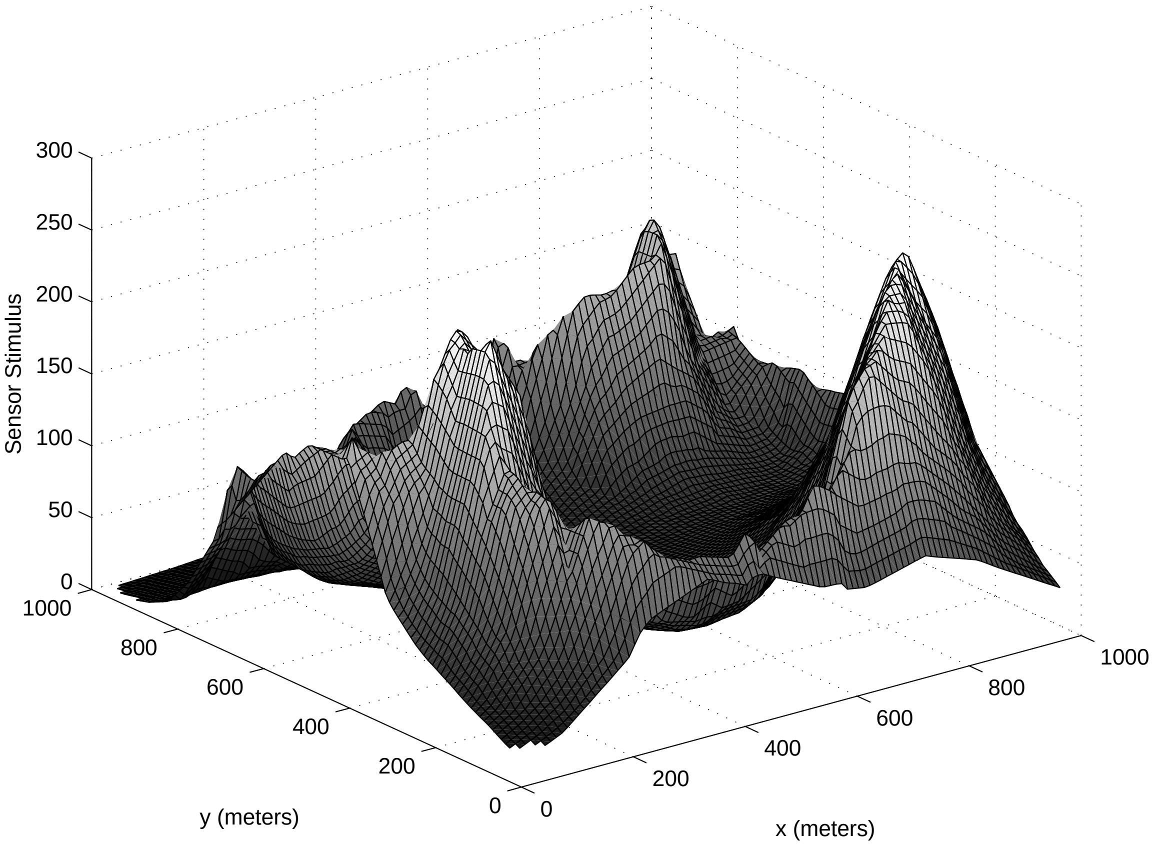
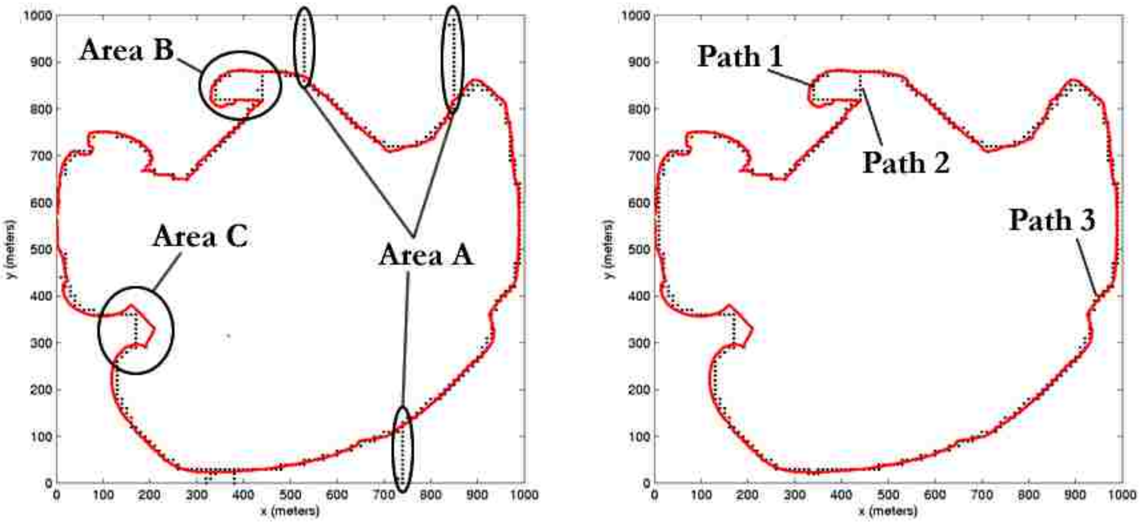


Figure 3.12. The 3D representation of Figure 3.11 where the intensity at each point is represented as a height.

points are used as an initial estimate of the fireline $\hat{F}_t$ plotted in Figure 3.13 as the dotted lines. Figure 3.13 also shows the actual fireline F_t, plotted as the solid lines. It is important to note that Figure 3.13 is generated from the intensity map shown in Figure 3.11(a) and not the one shown in Figure 3.11(c). This intensity map was used to generate the estimate shown in Figure 3.13(a) because this estimate clearly shows examples of all possible errors that may occur in the fireline estimation process. To generate the intensity map shown in Figure 3.11(a), a single sensor was placed in each pixel in the 1km×1km map forming a grid of 10,000 sensors.



(a) (b)

Figure 3.13. These figures show two of the steps of the fireline estimation algorithm. Both examples are compared to the actual fireline F_t is shown as a solid line while the estimated fireline $\hat{F}_t$ is shown as a dotted line. (a) This is the first estimate after the watershed algorithm. There are problem areas such as extra loops and ridges extending to the edges of the map. (b) The result of removing the extended ridges.

Comparing F_t to $\hat{F}_t$ in Figure 3.13(a), one can see there are three distinct problem areas in the initial estimate. Areas A in Figure 3.13(a) show ridges extending from the estimated fireline to the edge of the map. Area B in Figure 3.13(a) shows smaller loops within the main loop formed by the fire. Area C in Figure 3.13(a) indicates a region with significant differences between F_t and $\hat{F}_t$. The cause of each of these problems will now be discussed along with the proposed solutions for refining the initial estimate.

The ridges extending to the edge of the map exist because it is possible for a number of different catchment basins to exist on the outside of the estimated fireline. This problem is solved by joining the separate catchment basins on the outside of the estimated fireline to form one large catchment basin. To create this large catchment basin, the area of the intensity map is increased by three pixels on each side and then assigning the newly included area an intensity of zero. This causes all water droplets that fall outside the cauldron of the mountain to end up in the same basin. Performing the watershed detection algorithm on the new image yields the estimate of F_t shown in Figure 3.13(b).

The small loop shown in Area B of Figure 3.13(a) is caused by the existence of a valley in the actual mountain. This valley can be thought of as lake in the mountain that forms an additional catchment basin. This basin is separated from two other basins, the main cauldron of the mountain and the outside of the mountain. As shown in Figure 3.13(b), the watershed that separates the small basin from the outside of the mountain is labeled Path 1 and the watershed that separates the small basin from the main cauldron is labeled Path 2. In Figure 3.13(b), Path 2 is incorrect and should be eliminated. A possible reason Path 2 may exist is now examined to help determine how to remove it. A sensor placed on Path 2 will receive a large intensity reading from above and below since it is relatively close to this portion of F_t. This intensity will be higher than the values of a neighboring sensor to the left or right because F_t is further away. This causes decreasing intensities as the sensor moves farther to the left or right indicating that a ridge should be formed along Path 2.

The proposed solution to this problem is to Dijkstra's algorithm to determine whether Path 1 or Path 2 has the largest overall intensity. The path with the highest overall intensity places $\hat{F}_t$ the closest to the actual fireline. Dijkstra's algorithm can be used to find the shortest path between a starting and ending vertex, given a weighted graph where the path weights are defined as the nonnegative cost from some vertex i to another vertex j [28]. The weighted graph to be used for Dijkstra's is defined as an adjacency matrix representing all paths between two neighboring points in the watershed output. A path in the graph from point i to point j is equal to the path from point j to point i. The entries in the adjacency matrix show the cost of the path between two neighboring points. This cost is defined as the inverse of the sum of the intensity seen between the two neighboring points.

To use Dijkstra's algorithm, starting and ending points need to be defined as well. To better understand how the starting and ending points were defined let a junction be defined as any point returned by the watershed algorithm that has at least 3 neighbors. Neighbors are defined as a point which lies above, below, left, or right of any point. Figure 3.14 shows some example junctions from the points making up the initial fireline estimate $\hat{F}_t$. Please note that Figure 3.14 is not generated from the same fire shown in Figure 3.13. Figure 3.14 merely shows an enlargement of an example watershed with junction points shown as the larger points. Junctions are found by checking every point in the initial fireline estimate and locating the points with three or more neighbors. Each junction found is stored along with its neighbors.

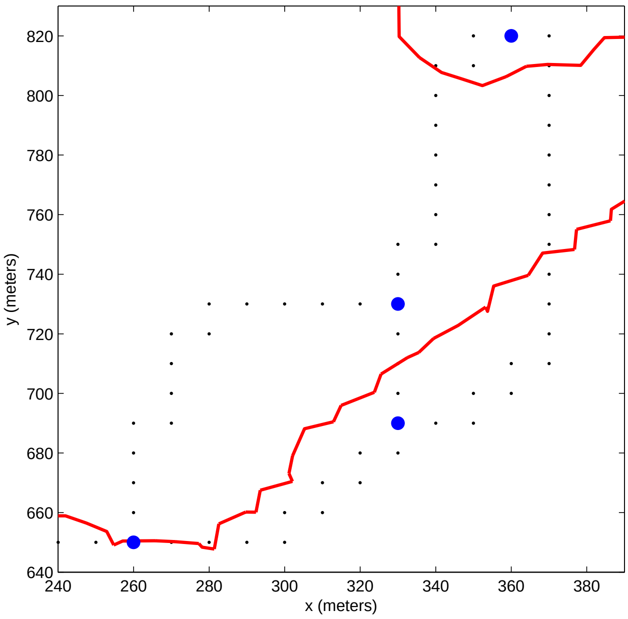


Figure 3.14. An enlarged view of the watershed algorithm output showing junctions. The large dots are junctions and sets of small dots between junctions compose a path. The solid line is the actual fireline.

A number of different methods for defining the starting and ending points were explored but it was founded that all but one failed to find the optimal path. The first method that failed was defining a junction as both the starting and ending point for the algorithm. This presented problems because the graph traversed by Dijkstra's algorithm was represented as an undirected graph, so the shortest path was from the junction to some neighbor and back to the junction. If a directed graph was used instead of an undirected graph this method would have worked. To define a directed graph from the watershed output was found to be computational difficult because the points were not in sequential order.

Another method was to use an arbitrary junction as the starting point and each of its neighbors as the ending points. Dijkstra's was evaluated for this method and failed as well because there is not guarantee that a given junction is on the optimal path. This is best illustrated by looking at the junction on Path 2 shown in Figure 3.13(b). If this method is used with the junction on Path 2 and any of its neighbors it will find either the combination of Path 1 and Path 2 or Path 3 and Path 2 and it is clear that the optimal path is the combination of Path 1 and Path 3.

The method that produced the best results was to use each pair of the neighbors for each junction as the starting and ending points. By setting the cost between each junction and all its neighbors and then traversing the path between each pair of neighbors for each junction, every possible path in the watershed output was explored. Each path could then be compared to find the path that produced the best estimate of $\hat{F}_t$.

After the adjacency matrix has been formed and each pair of starting and ending points have been defined, Dijkstras algorithm is run on the adjacency matrix. This gives the minimum cost path for each pair of starting and ending points. When Dijkstra's algorithm has completed for all starting and ending points, the path with the highest minimum cost is returned as the rough estimate of the fireline $\hat{F}_t$. Figure 3.15 shows the resultant estimated fireline $\hat{F}_t$. Notice that Path 2 from Figure 3.13(b) has been eliminated.

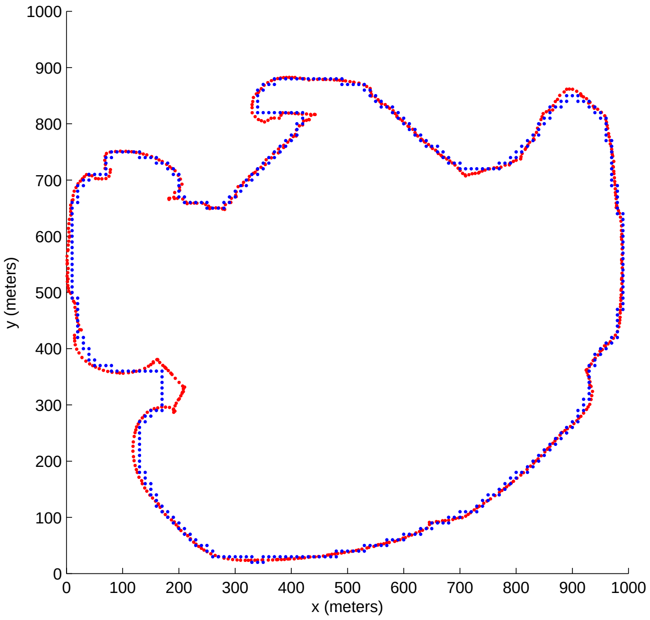


Figure 3.15. The result of using Dijkstra's algorithm to remove unwanted paths. The estimate $\hat{F}_t$ is shown as the dotted line and F_t is shown as the solid line.

The problem shown in Area C in Figure 3.13(a) exists because of C-shaped areas in the fireline. Sensors that lie within the 'C' will see more intensity than sensors that lie on or outside the 'C.' This causes a large peak on the intensity map that is not located on the actual fireline. The watershed algorithm correctly finds the ridge of the mountain but it will not be located near the actual fireline and can result in large errors as shown in Area C in Figure 3.13(a). In addition, other areas may have significant error if the sensor density is too low. Instead of approaching this problem in an algorithmic fashion the addition of sensors to the areas of significant error is examined in Section 4. Section 4 shows that for the majority of fires exhibiting areas of significant error, the addition of a small amount of sensors can improve the estimate. This method did not work for all the fires so improving the estimate for the remaining fires has been left to future research.

4 SIMULATION SETUP AND RESULTS

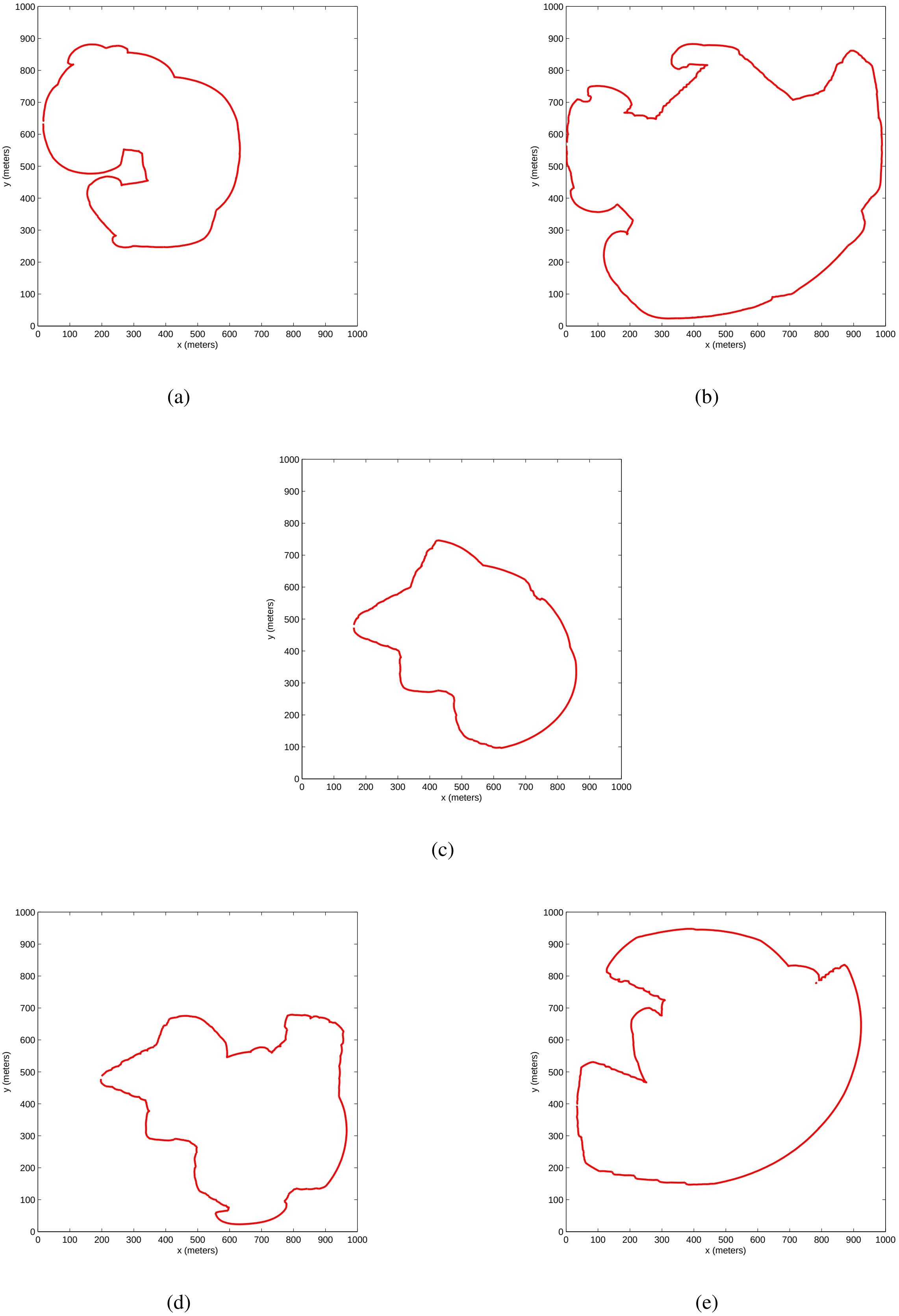
The following simulations are setup to prove the feasibility of using wireless sensor networks for the detection and mapping of large wildfires. FARSITE was used to simulate fires covering five different size areas. For each size area, five unique fires were simulated for a total of twenty-five test fires. The Detection of Wildfire Simulator (DOWsim) was setup to take the FARSITE output as input. DOWsim was then used to perform two experiments: first, determine the density of sensors needed over an area to generate an intensity map and second, determine if a fireline can be recreated from the data present in the generated intensity map.

4.1 FARSITE FIRE AREA SIMULATOR

Five different FARSITE projects were generated to simulate five different fires. For each fire the only change made is the types and locations of the fuels present on the map and the ignition locations. Each fire was run on a 1km×1km map for a simulation time of 20 hours. An output file was generated for each fire. These files contain a list of points on the fireline. The tuple for each point contains the following fields: the position $F_i(t)$, the time of arrival t, the intensity $I_{F_i}(t)$, and the height of the flame $H_{F_i}(t)$. Information for the fireline is available in ten minute time intervals and each point on the fireline is constrained to be at most ten meters from the next point. Figures 4.1 through 4.5 show one time step of the firelines generated from the FARSITE simulations.

4.2 DOWsim

A simulator was developed in MATLAB to help determine if wireless sensor networks can be used for estimating the intensity and location of the front of a fire in real-time. The simulator is called the Detection of Wildfire Simulator (DOWsim). The first task of



(a)

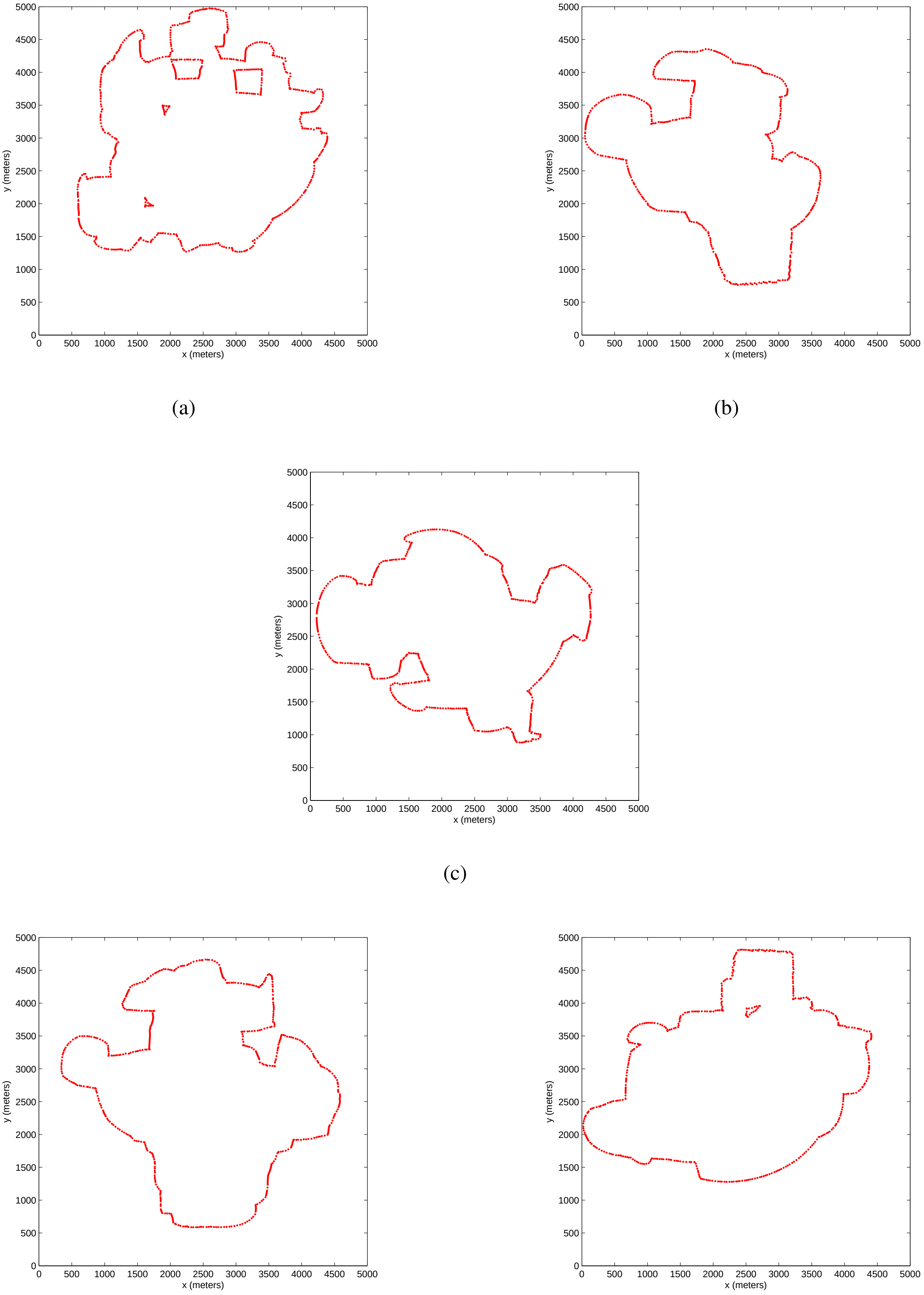
(b)

(c)

(d)

(e)

Figure 4.1. The actual 1km by 1km firelines F_t generated by FARSITE.



(a)

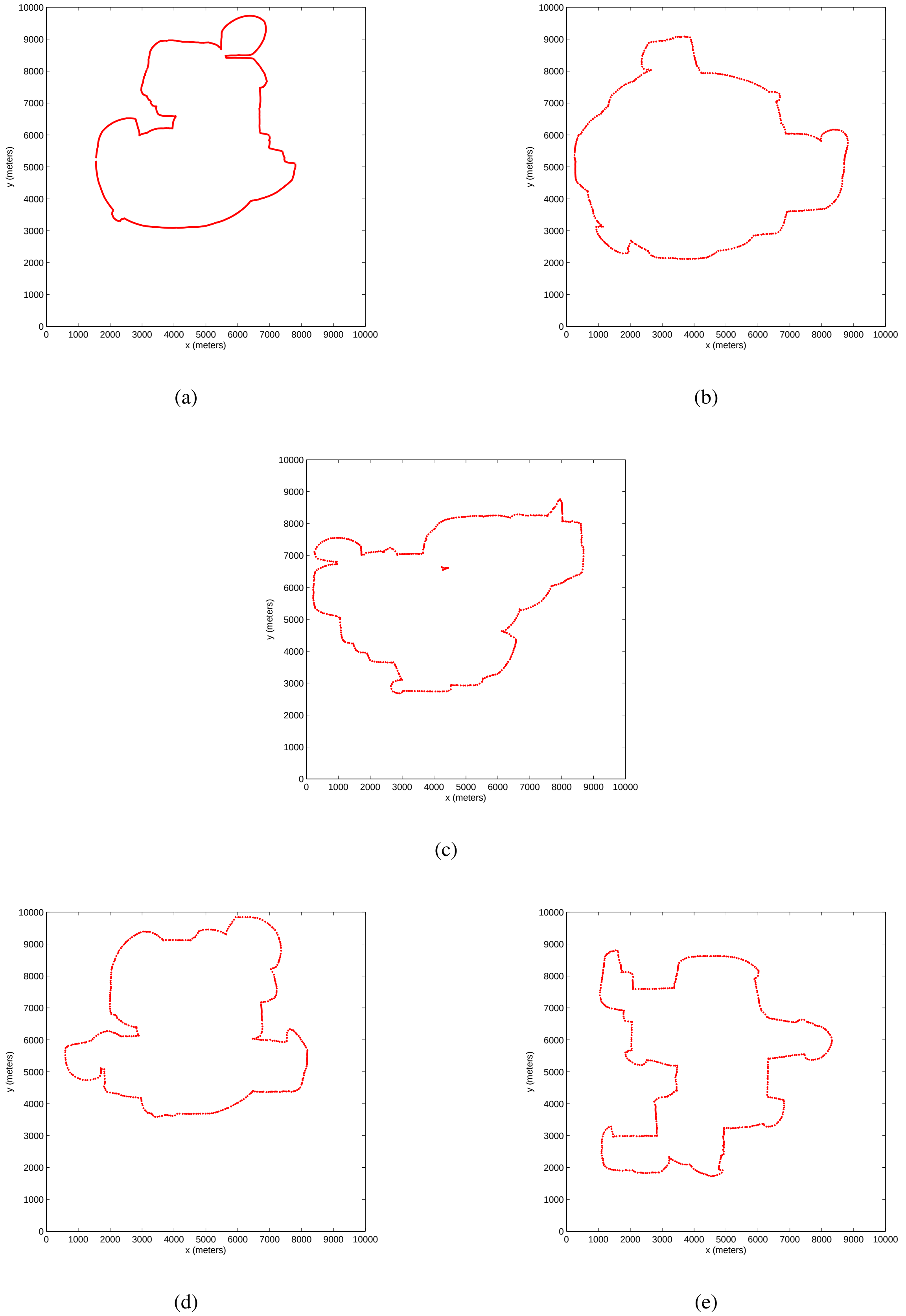
(b)

(c)

(d)

(e)

Figure 4.2. The actual 5km by 5km firelines F_t generated by FARSITE.



(a)

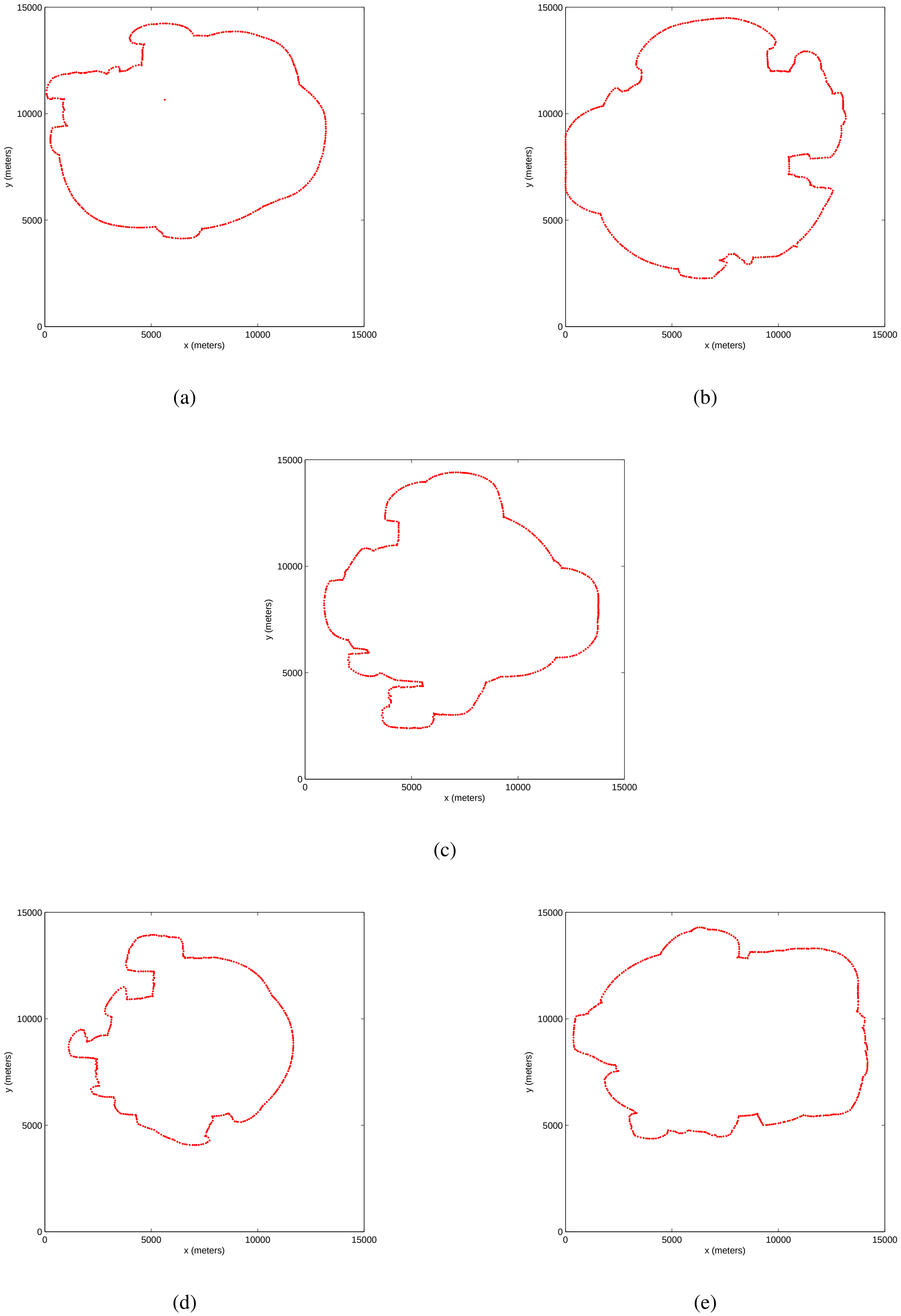
(b)

(c)

(d)

(e)

Figure 4.3. The actual 10km by 10km firelines F_t generated by FARSITE.



(a)

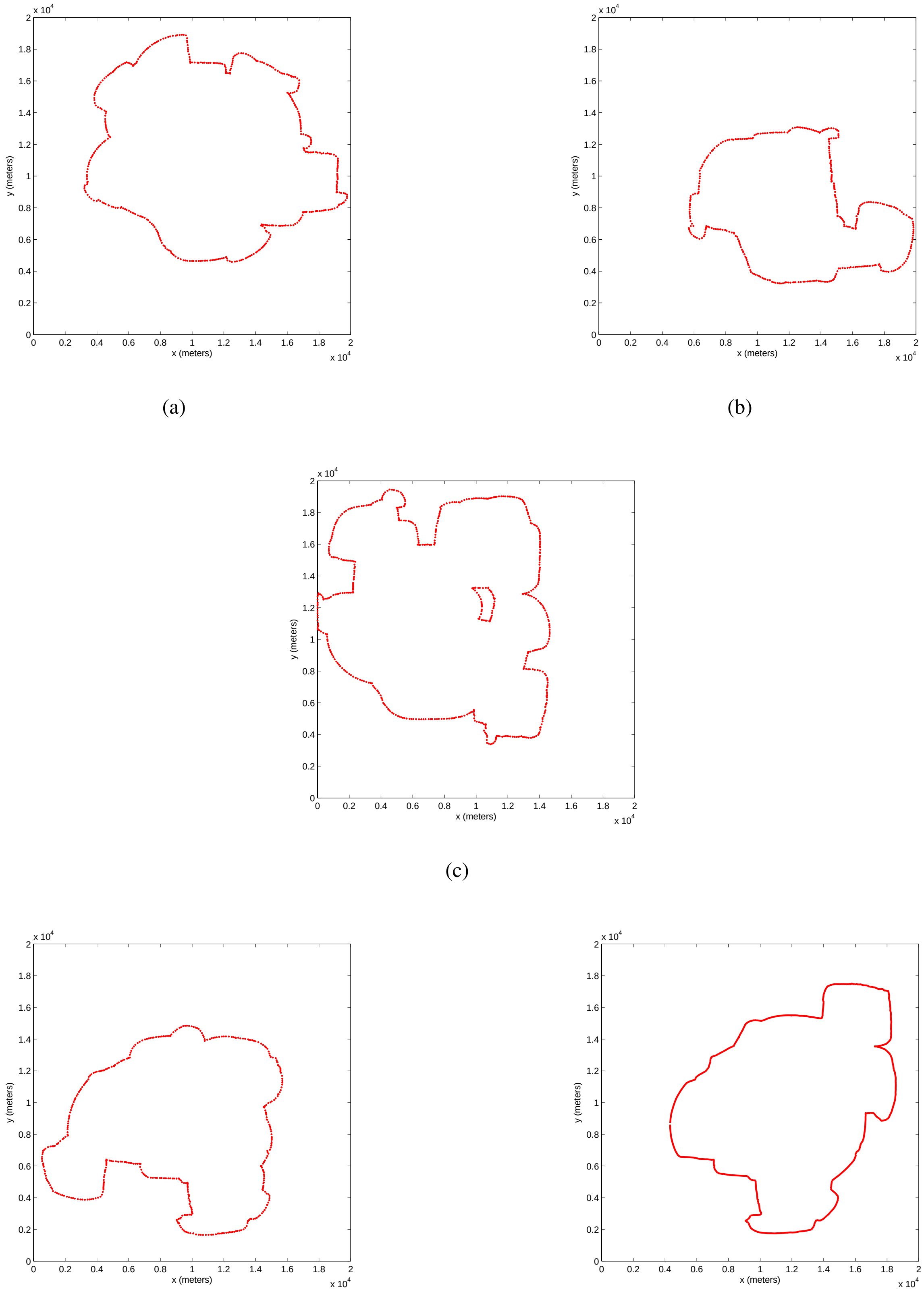
(b)

(c)

(d)

(e)

Figure 4.4. The actual 15km by 15km firelines F_t generated by FARSITE.



(a)

(b)

(c)

(d)

(e)

Figure 4.5. The actual 20km by 20km firelines F_t generated by FARSITE.

this simulator is to reconstruct the actual fireline F_t based on the output from FARSITE. This is done using the method described in Section 3. Next, sensors equipped with IR detectors are randomly distributed over the area of the fire. Then for each time step, the stimulus received at each sensor is determined and an intensity map based on the sensor data is generated. Image processing techniques are then performed on this intensity map to estimate the fireline.

4.3 PRODUCING ACCURATE INTENSITY MAPS

Recall in Section 3.3, an intensity map was generated by placing 10,000 sensors on a 1km$\times$1km grid with a spacing of 10 meters between each sensor. No interpolation is needed for this sensor arrangement because each pixel contains a sensor. This scenario is quite unlikely to occur in a real-world deployment. In order for sensors to end up in a regular grid they have to be intentionally placed in specific locations. A rapid network deployment is envision for this research such as dropping sensors from an aircraft when a fire gets out of control. Therefore, the distribution of sensors is modeled as a random placement on a 2-dimensional grid. The goal of this section is to determine the density of randomly distributed sensors required to obtain similar results to the ‘ideal’ case of using a dense, regular distribution while keeping the number of sensors low. This is done to reduce system costs.

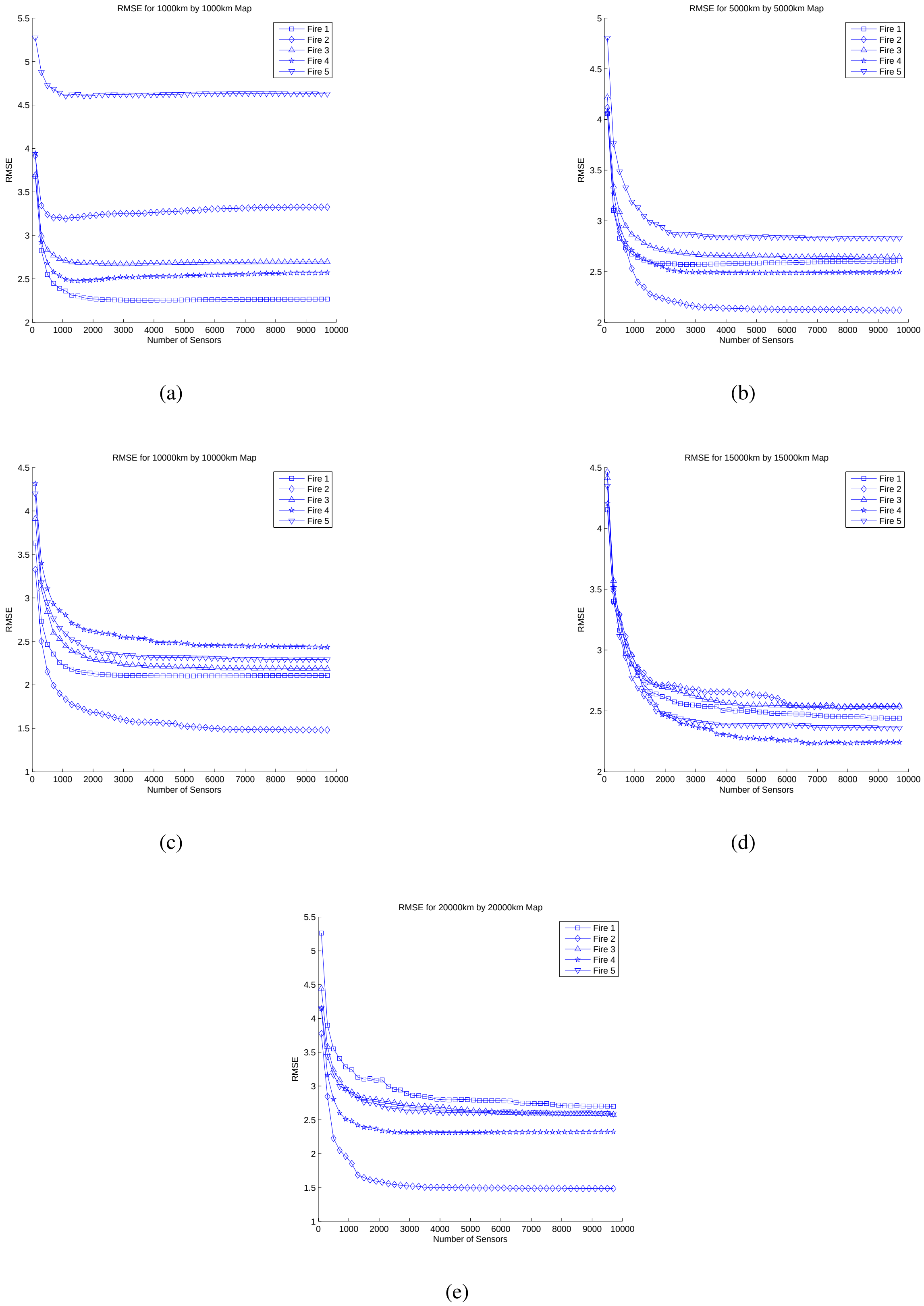
For each fire generated, the fireline for one arbitrary time step was chosen to generate stimuli for sensor distributions. 30 different random sensor distributions were tested for each fireline. Distributions were generated by randomly placing a number of sensors over the area of the fire in question. For each distribution, 50 sensor densities were considered, ranging from 100 to 10,000 sensors per square kilometer. This gave a total of 1500 experiments per fire.

The root mean square error (RMSE) was chosen as the metric to determine the accuracy of the generated intensity map. The generated intensity map was compared to the 'ideal' intensity map for each fire. To generate the ideal intensity map for each fire, a regular grid of sensors was placed over the area of the fire. The spacing of the sensors on this grid was determined by the ratio of the size of the map to the pixel size. For some areas in the intensity map generated from the interpolation, there will be no intensity values available. This is caused by missing sensors on the edges and corners of the map. When calculating the RMSE between the two intensity maps we disregard areas where no intensity values exist. Results from these experiments are shown in Figure 4.6.

Figure 4.6 is used to determine the lowest sensor density needed to generate an intensity map for a fire in one of the five map sizes. To do this, the point in the plots for each fire where increasing the number of sensors will not decrease the RMSE by a significant amount needs to be located. To clarify, consider Figure 4.6(a). From this figure it is easy to see the RMSE levels out when 2000 sensors/km^2 is distributed over the area of the map. Based on this it could be concluded that this is a reasonable density of sensors but the possibility of using a lesser density needs to be examined. By reducing the sensor density to 1000 sensors/km^2, an intensity map with low error can still be produced. By reducing the sensor density the RMSE for some of the fires does increase but the increase is minimal. For this reason, it can be concluded that a density of 1000 sensors/km^2 is suffice for this size map. Table 4.1 summarizes the minimum sensor densities chosen using the method above for each of the five map sizes.

Table 4.1. Minimum sensor densities for intensity maps.

Map Size	Sensor Density (sensors/km^2)
1km by 1km	1000
5km by 5km	2000
10km by 10km	3000
15km by 15km	4000
20km by 20km	4000



(a)

(b)

(c)

(d)

(e)

Figure 4.6. Results of the RMSE experiments. Each data point is the average RMSE for that particular number of sensors.

The values in Table 4.1 are simply suggested minimum values for generating accurate intensity maps. Depending on the fire it may be necessary to distribute more sensors over the area of the map or to selected areas of the map to produce the estimated fireline. In the next section the suggested sensor densities in Table 4.1 are investigated to determine if the information is enough to reproduce the estimated fireline.

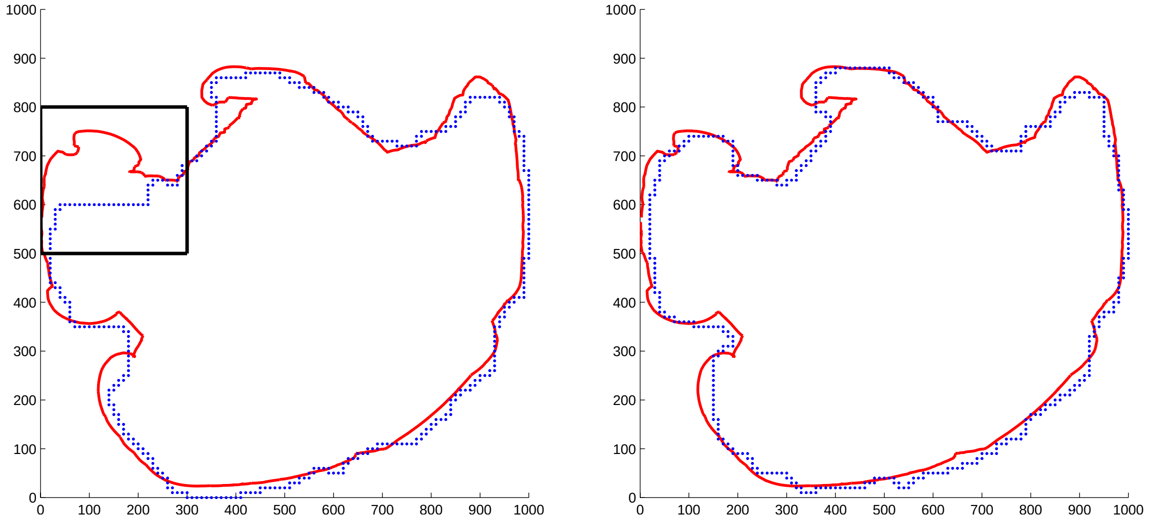
4.4 FIRELINE ESTIMATION

For each of the fires, the information from the generated intensity maps was used to create the estimated the fireline $\hat{F}_t$. The firelines were estimated using the process described in Section 3.3 and the sensor densities specified in Table 4.1. Each estimate $\hat{F}_t$ was analyzed for significant error to determine if the sensor density specified in Table 4.1 is large enough to generate the necessary information needed for recreating the fireline.

After analyzing each of the fires it was found that depending on the seed used for the distribution of the sensors, significant errors can exist in some of the fireline estimates. For example, consider Figure 4.7. Figure 4.7(a) shows an example of a sensor distribution using 1000 sensors which produces and area of high error shown in the boxed area while Figure 4.7(b) is another sensor distribution which also uses 1000 sensors for the same fire but produces an estimate with low error.

The error present in Figure 4.7(a) is the result of the lack of information available in the generated intensity map. For fires that show areas of large error like the fire in Figure 4.7(a) it is necessary to gather more information from the areas with high error. Distributing additional sensors to the entire map was explored. For most cases this could produce a better estimate but at the cost of approximately double the amount of sensors specified in Table 4.1. Since the goal is to maintain a low number of sensors a more sensible method of distributing sensors to the map was explored.

Distributing sensors to the areas of high error rather than the entire map is now explored. The issue with this method is deciding where to distribute the additional sensors.

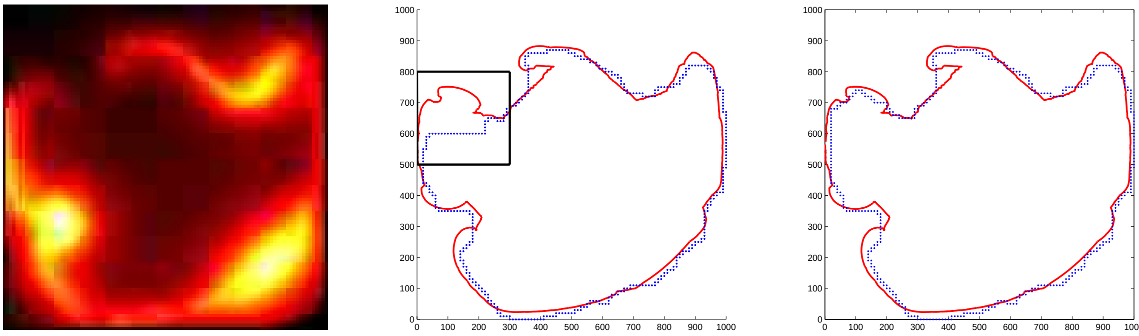


(a)

(b)

Figure 4.7. Fireline estimates of the same fire using 1000 sensors but different distributions. This illustrates that the error present in the estimate $\hat{F}_t$ is dependent on the distribution of the sensors.The dotted lines are the estimates $\hat{F}_t$ and the solid lines are the actual fireline.

In the real-world application of this system the only information the user may have is the generated intensity map and the estimated fireline and the user should be able to determine where additional sensors will need to be distributed from this information. For an example of how one might use the intensity map and estimate to determine if the addition of sensors is needed consider Figure 4.8. Recall in Section 3.3 it was stated that one could get a rough estimate by following the general shape of the intensities in the intensity map. This property of the intensity is used to compare the shape of the intensity map with the shape of the estimated fireline shown as the dotted line in Figure 4.8(b). Looking at the boxed area in Figure 4.8(b) and comparing it to the same area in Figure 4.8(a) it can seen that the shapes do not match. Areas like this need additional sensors to improve the estimate. For this example experiments for adding sensors to the boxed in area as well as adding sensors to the entire map were run. It was found that with adding as little as 100 additional sensors to the boxed area produced the estimate shown in Figure 4.8(c) while it would take 1500 sensors to produce a similar estimate if sensors were distributed to the entire map.

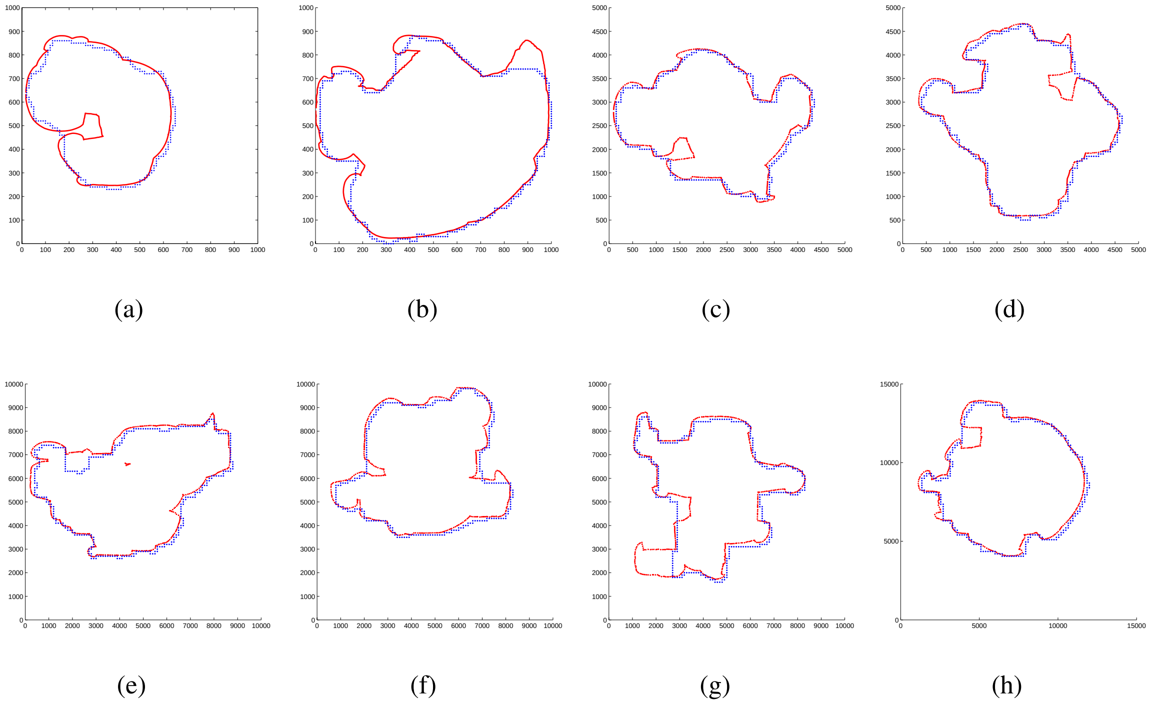


(a) (b) (c)

Figure 4.8. Plots to help illustrate how the generated intensity map and the estimated fireline can be used to determine if additional sensors will be needed to improve the estimate. The dotted lines in (b) and (c) are the estimates $\hat{F}_t$ while the solid lines are the actual firelines

The experiments show that distributing a small number of sensors over a specified area works with the example shown in Figure 4.8. This method was tested on a number of other fires which exhibited similar errors. These fires are shown in Figure 4.9 and Table 4.2 compares the number of sensors for the two methods of improving the fireline estimate. An entry of ‘CNI’ in Table 4.2 denotes that the estimate could not be improved by adding additional sensors to the map.

For some of the fires, namely fires 9 (Figure 4.9(d)), 14 (Figure 4.9(f)), and 19 (Figure 4.9(h)), the estimate was not able to be improved at all because of areas like Area C that were previously discussed in Section 3.3. The experiments were able to show that for some of the fires containing areas like Area C the estimate could be improved by distributing additional sensors to the area of the problem. Not being able to improve every estimate error by distributing additional sensors shows that the need for an algorithm to improve



(a) (b) (c) (d)

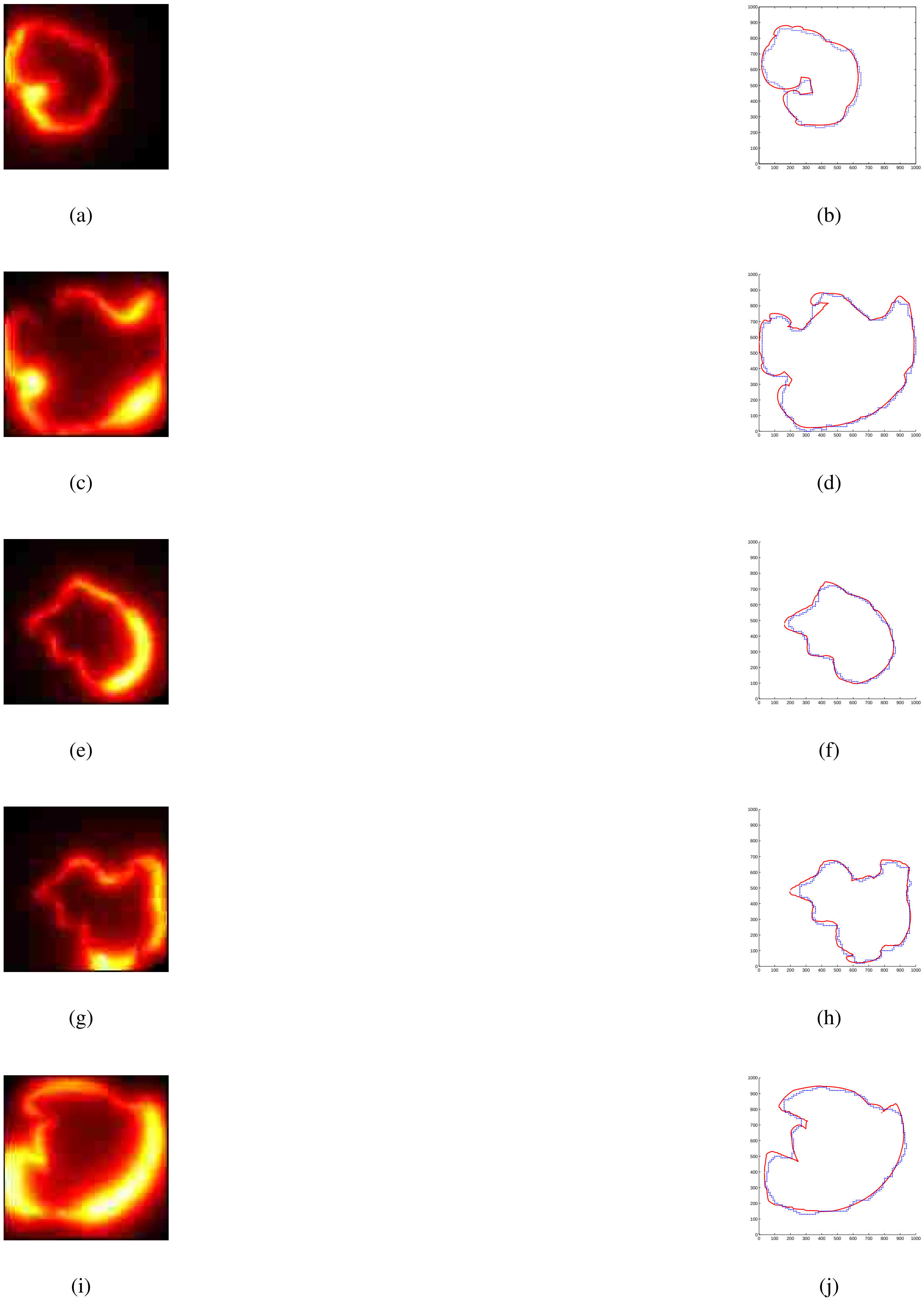
(e) (f) (g) (h)

Figure 4.9. Additional fires which exhibited significant error.

Table 4.2. Additional sensors needed to improve fireline estimate.

Map Size	Entire Map	Specified Area
Fire 1	3000	100
Fire 2	1500	100
Fire 8	4500	100
Fire 9	CNI	CNI
Fire 13	3500	100
Fire 14	CNI	CNI
Fire 15	19000	200
Fire 19	CNI	CNI

areas like Area C still exists. These improvements are being left to future research. The estimated fireline $\hat{F}_t$ and the corresponding intensity map for each simulated fire is shown in Figures 4.10 through 4.14.



(a) (b)

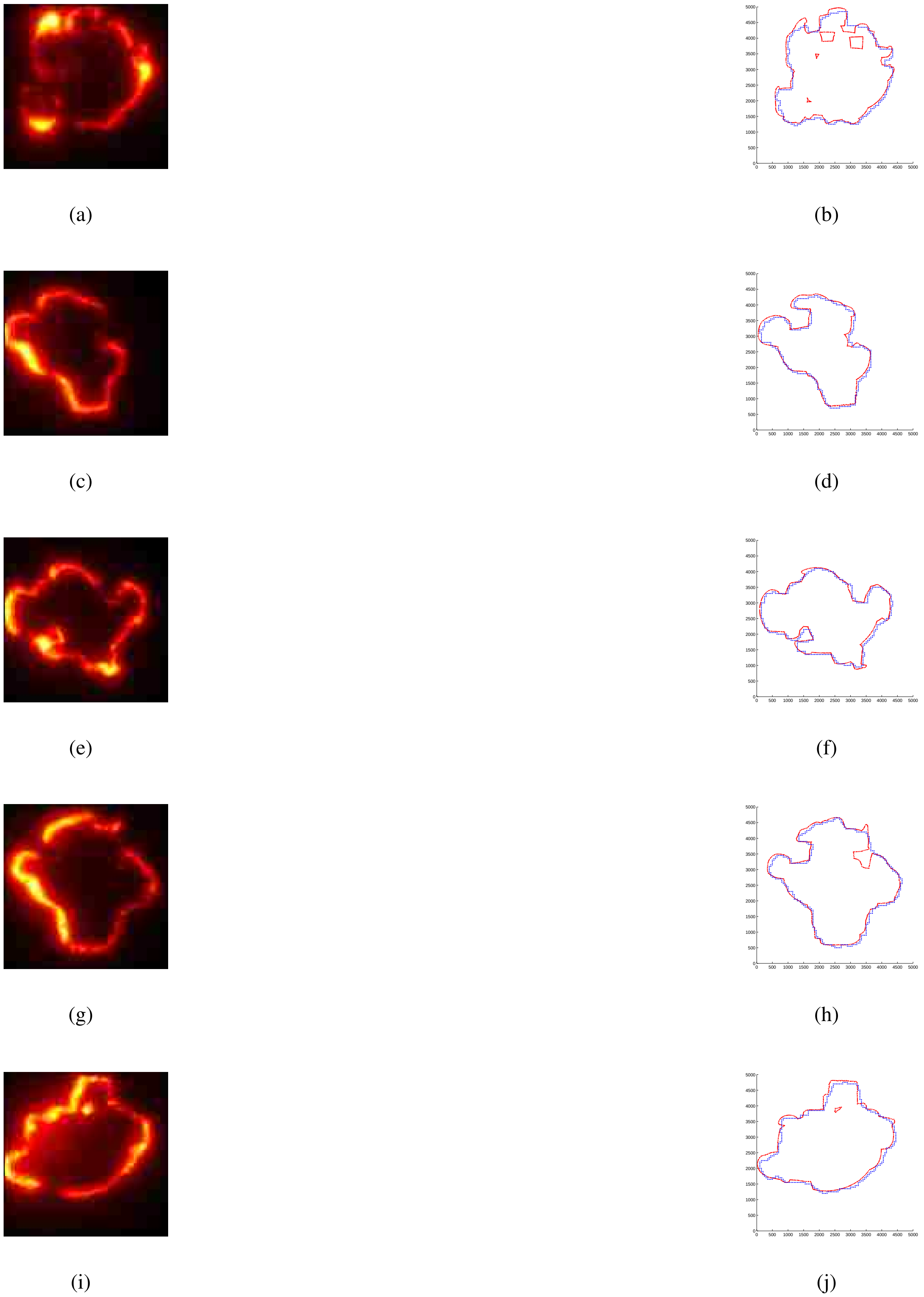
(c) (d)

(e) (f)

(g) (h)

(i) (j)

Figure 4.10. The estimates $\hat{F}_t$ and their corresponding intensity maps for the 1km by 1km fires.



(a) (b)

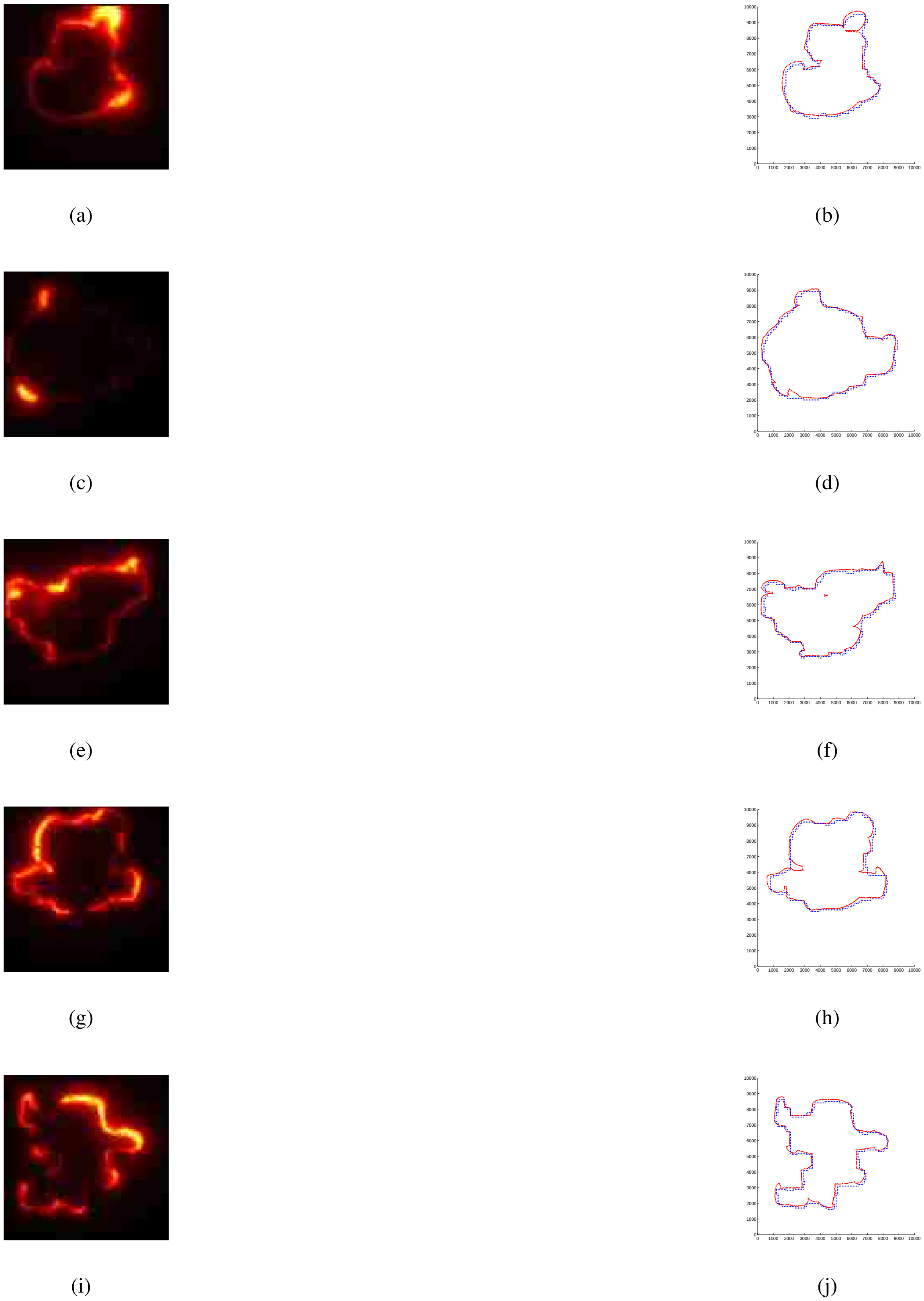
(c) (d)

(e) (f)

(g) (h)

(i) (j)

Figure 4.11. The estimates $\hat{F}_t$ and their corresponding intensity maps for the 5km by 5km fires.



(a) (b)

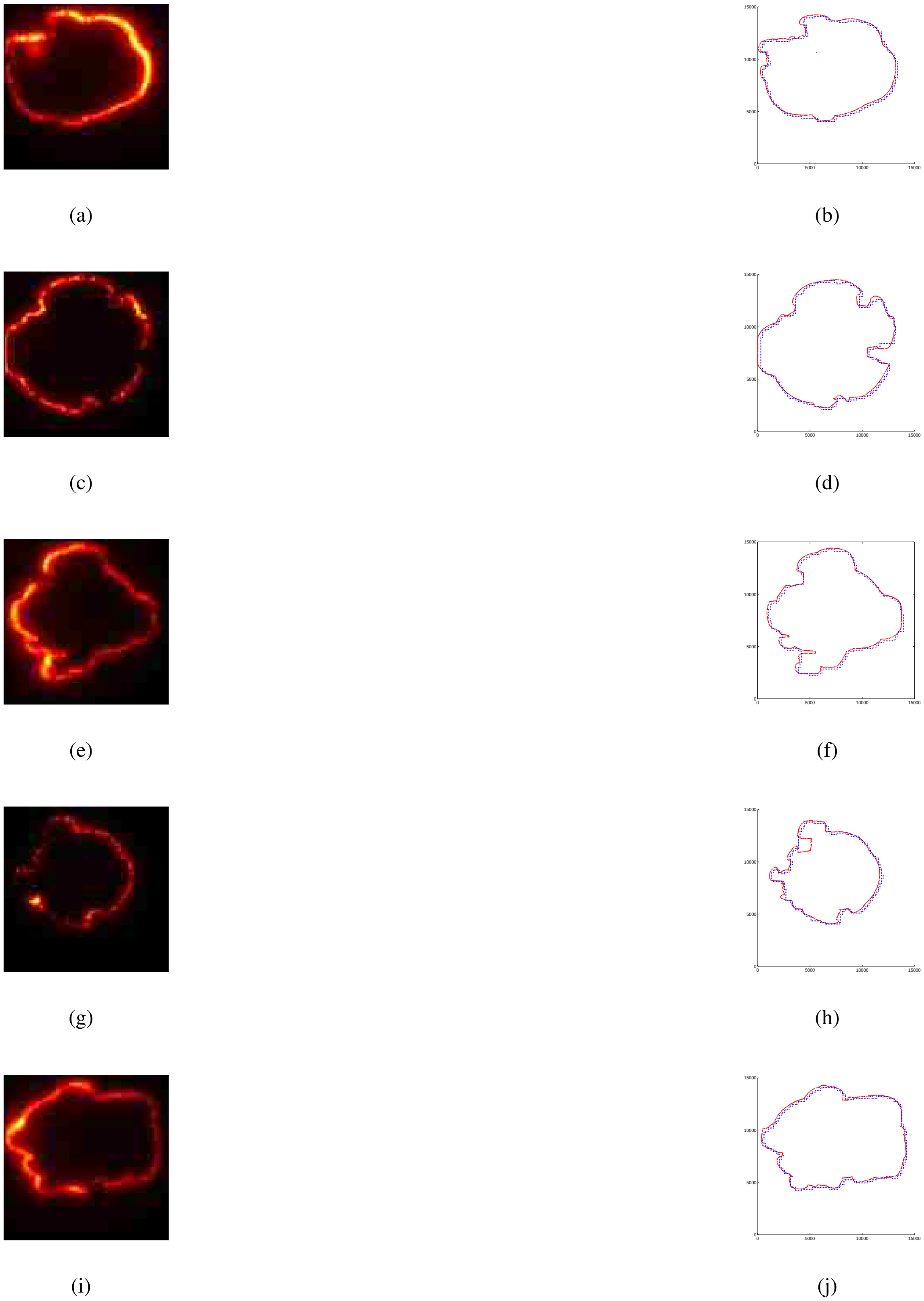
(c) (d)

(e) (f)

(g) (h)

(i) (j)

Figure 4.12. The estimates $\hat{F}_t$ and their corresponding intensity maps for the 10km by 10km fires.



(a)

(b)

(c)

(d)

(e)

(f)

(g)

(h)

(i)

(j)

Figure 4.13. The estimates $\hat{F}_t$ and their corresponding intensity maps for the 15km by 15km fires.

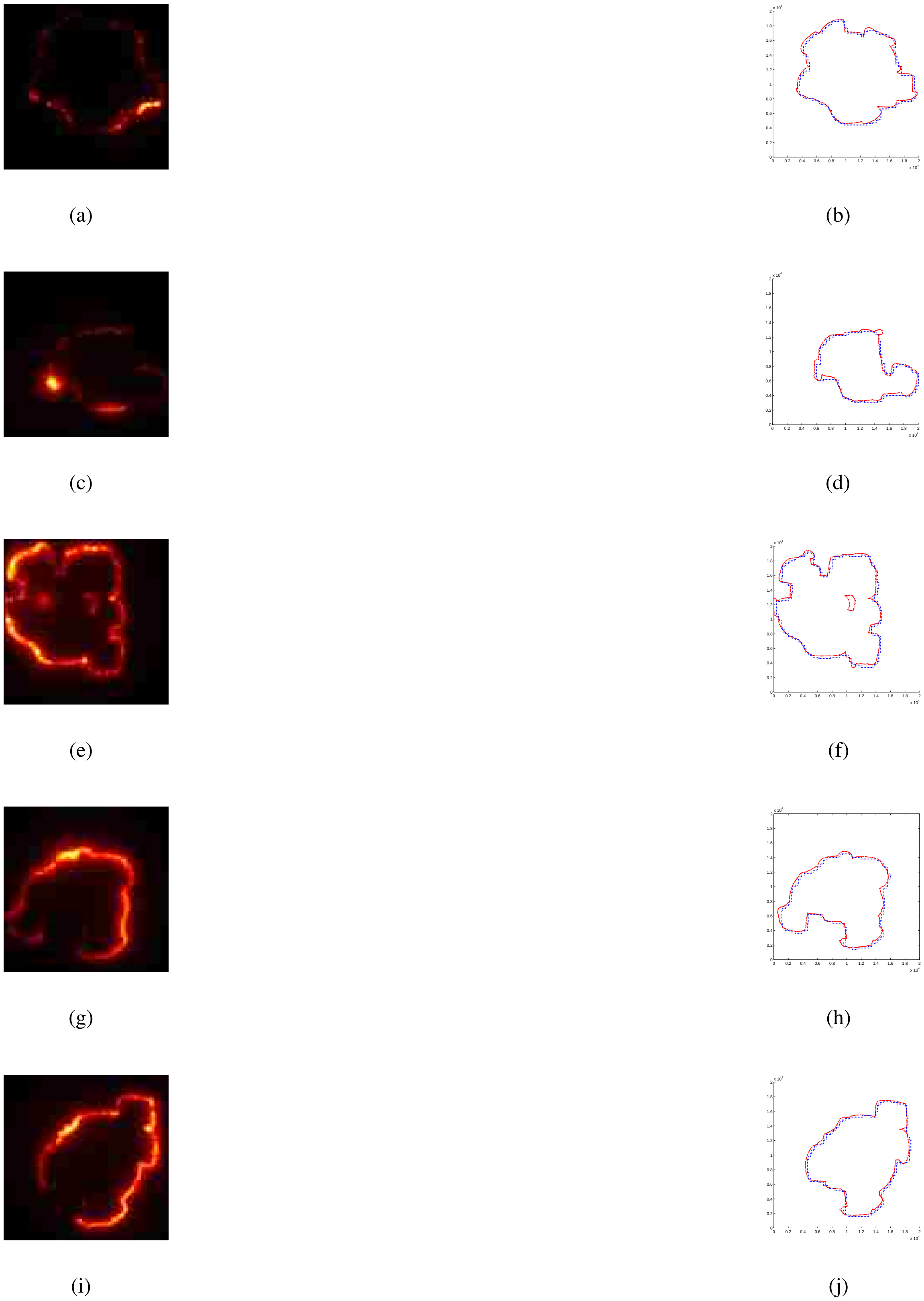


Figure 4.14. The estimates $\hat{F}_t$ and their corresponding intensity maps for the 20km by 20km fires.

5 CONCLUSIONS

Current wildfire mapping systems create an information poor environment because of poor resolution both spatially and in time. A real-time system that can provide timely reports and good spatial resolution is needed in such an environment to limit the amount of destruction caused by the wildfires. Wireless sensor networks are a perfect candidate for such a system that must provide timely detection and frequent report in an area of high danger. The specific contributions of this thesis have been the development of a simulation environment that

1) mathematically models an IR sensor for the detection of the intensity put off by the wildfire as seen by a particular sensor,

2) generates an intensity map from the intensities seen at all sensor in area of the wildfire, and

3) recreates an estimate of the actual fireline by processing the information in the generated intensity map using common image processing techniques.

Experiments were run on twenty-five fires generated by FARSITE of different shapes and sizes to enumerate the number sensors that needed to be distributed over the to gather enough information to generate an accurate intensity map. Simulations for a number of different distributions and sensor distribution sizes were run. An ideal intensity map was created to compare the generated intensity maps to. The root-mean squared error was used as the metric for the comparison between the two maps. The result from these experiments showed that a reasonably low number of sensors was need to produce and accurate intensity map for all size fires tested.

Further experiments were conducted to see if the information provided in the intensity map can be used to successfully recreate an estimate of the fireline. The results of the experiment showed that image processing techniques alone were not able to reproduce the estimate for all fires. A number of the fires exhibited areas of multiple paths in the estimate or high errors. For the areas with multiple paths in the estimate it was found that using a modified version of Dijkstra's algorithm produced a number of paths from which the best estimate of the fireline could be chosen. For the fires which exhibited high error in certain areas it was discovered that by distributing additional sensors to the area of high error the estimate could be improved for most fires. The overall results of the experiments showed that a good estimate for the majority of the fires could be produced but some fires still showed areas of high error. The improvement of the estimates for these fires with high error has been left to future research.

Additional work that still needs to be explored is the addition of the network simulator to the simulator. Currently it has been assumed that data from every sensor in the network reaches the base station or data processing computer where the image processing techniques are performed. In the real-world this will obviously not be the case since there are a number of obstacles that can occur in the wireless environment such as channel error, down links/sensor, and dropped packets. With the network simulator added experiment for determining the new number of sensors needed to produce the same kind of results that have been produced in this thesis.

BIBLIOGRAPHY

[1] “National interagency fire center.” http://www.nifc.gov/index.html, March 2007.

[2] “Modis fire and thermal anomalies.” http://modis-fire.umd.edu/index.asp, March 2007.

[3] “Remote sensing applications center.” http://www.fs.fed.us/eng/rsac/index.html, March 2007.

[4] “A private conversation with research forrester mark finney with the missoula fire sciences laboratory of the us forest servvices,” March 7 2005.

[5] F. Zhao and L. J. Guibas, *Wireless Sensor Networks An Information Processing Approach.* Morgan Kaufmann Publishers, 2004.

[6] H. Karl and A. Willig, *Protocols and Architectures for Wireless Sensor Networks.* John Wiley & Sons Ltd., 2005.

[7] D. Culler, D. Estrin, and M. Srivastava, “Overview of sensor networks,” in *Computer*, vol. Volume 37, Issue 8, pp. pages 41–49, August 2004.

[8] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “Wireless sensor networks: A survey,” in *IEEE Communication Magazine*, vol. vol. 40, no. 8, pp. pages 102–116, August 2002.

[9] “Advanced very high resolution radiometer (avhrr).” http://edc.usgs.gov/guides/avhrr.html, March 2007.

[10] Cooperative Institute for Meterological Satellite Studies Space Science and Engineering Center, http://cimss.ssec.wisc.edu/goes/burn/abba.html, March 2007.

[11] The Landsat Program, http://landsat.gsfc.nasa.gov/, March 2007.

[12] Moderate Resolution Imaging Spectroradiometer or MODIS, http://modis.gsfc.nasa.gov/about/, March 2007.

[13] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. Abdelzaher, “Range-free localization schemes for large scale sensor networks,” in *MobiCom ’03: Proceedings of the 9th annual international conference on Mobile computing and networking*, (New York, NY, USA), pp. 81–95, ACM Press, 2003.

[14] D. Niculescu and B. Nath, “Ad hoc positioning system (aps),” in *IEEE Global Telecommunications Conference (GLOBECOM 01)*, vol. 5, pp. 2926–2931, Nov 2001.

[15] A. Savvides, C.-C. Han, and M. B. Strivastava, “Dynamic fine-grained localization in ad-hoc networks of sensors,” in *MobiCom ’01: Proceedings of the 7th annual international conference on Mobile computing and networking*, (New York, NY, USA), pp. 166–179, ACM Press, 2001.

[16] D. Moore, J. Leonard, D. Rus, and S. Teller, “Robust distributed network localization with noisy range measurements,” in *SenSys ’04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, (New York, NY, USA), pp. 50–61, ACM Press, 2004.

[17] Y. Shang, W. Ruml, Y. Zhang, and M. P. J. Fromherz, “Localization from mere connectivity,” in *MobiHoc ’03: Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, (New York, NY, USA), pp. 201–212, ACM Press, 2003.

[18] L. E. G. Lance Doherty, Kristofer S. J. Pister, “Convex position estimation in wireless sensor networks,” in *Proceedings of the Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2001)*, vol. 3, pp. 1655–1663, Apr. 2001.

[19] P. Biswas, T.-C. Lian, T.-C. Wang, and Y. Ye, “Semidefinite programming based algorithms for sensor network localization,” *ACM Trans. Sen. Netw.*, vol. 2, no. 2, pp. 188–220, 2006.

[20] FARSITE Fire Area Simulator, http://www.fire.org/, April 2007.

[21] M. A. Finney, “Farsite a fire area simulator for fire managers,” in *The Biswell Symposium*, February 15-17 1994.

[22] M. A. Finney and P. L. Andrews, “The farsite fire area simulator: Fire management applications and lessons of summer 1994,” in *Interior West Fire Council Meeting and Symposium*, November 1-2 1994.

[23] http://www.ice.ucdavis.edu/, March 2007.

[24] Public Service Projects Index, http://mb-soft.com/juca/print/314.html, March 2007.

[25] J. Stewart, *Calculus*. Brooks/Cole Publishing Comapany, fourth edition ed., 1999.

[26] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, “The quickhull algorithm for convex hulls,” in *ACM Transactions on Mathematical Software*, vol. volume 22, issure 4, pp. 469–483, December 1996.

[27] L. Vincent and P. Soille, “Watersheds in digital spaces: An efficient algorithm based on immersion simulations,” in *IEEE Transactions of Patter Analysis and Machine Intelligence*, vol. 13, June 1991.

[28] S. S. Skiena, *The Algorithm Design Manual*, ch. Ch 4 - Graph Algorithms. Springer Science+Business Media, Inc., 1998.

VITA

Matthew Gann was born in Santa Ana, California on May 31, 1983. In December 2005, he received a B.S. in both Electrical Engineering and Computer Engineering from University of Missouri-Rolla, Rolla, Missouri, USA. In August 2007, he received his M.S. degree in Computer Engineering from University of Missouri-Rolla, Rolla, Missouri, USA.