Physical Approximations for Urban Fire Spread Simulations

Daniel Bertinshaw and **Hans W. Guesgen**
Computer Science Department, University of Auckland
Private Bag 92019, Auckland, New Zealand
dber021@ec.auckland.ac.nz, hans@cs.auckland.ac.nz

Abstract

The issue of fire propagation in cities is of obvious importance to Civil Authorities, but does present issues of computational complexity. Our basic assumption is that some event has occurred to disrupt the normal damage prevention infrastructure, i.e. the Fire Department, so that the fire spreads unhindered through the city.

For the most part, the spread of fire is stopped inside the building by construction methods (fire glazing, fire proof treatment of woods etc), and the internal sprinkler systems. In many cities the sprinkler systems are fed by the municipal water supply, so should that fail due to catastrophic damage, buildings will burn uncontrolled. The Fire Control Services are, of course, the next line of defense, but, given our assumption, they are incapable of dealing with the fire for some reason.

We then wish to find out how fire will spread in a disaster situation, allowing the fire department and other civil authorities to combat a scenario, armed with the knowledge of what places are most prone to fire spread. To do this we wish to have some representation of cities such that we can measure the likelihood of fire spreading from one building to the next. Therefore we must cover the physics of fire, and then deduce a representation that facilitates the calculation of the fire spread.

However, fire, by its nature, is difficult to simulate accurately. As consulted experts agreed that fire spread is a noisy value, due to a huge variety of factors changing the shape of the fire within a building (e.g., the distribution of fuel, heating duct, the shape of the internal structure), applying high accuracy operations to the problem is meaningless.

To do this we wish to have some representation of cities such that we can measure the likelihood of fire spreading from one building to the next. Therefore we must cover the physics of fire, and then deduce a representation that facilitates the calculation of the fire spread

Introduction

Utilizing and harnessing fire has been one of the main achievements of human kind. Fire provides light, cooks our food, and warms our houses. However, it is also a potential danger, as it can destroy buildings and vegetation, and may cause deaths. To counter the adverse effects of fire, almost all cities have put measures into place to prevent this. Fire departments around the world are trained to cope with fires that get out of control—at least on a limited scale. When it comes to disasters, like earthquakes or volcanic eruptions, fire departments often cannot cope adequately with the demands imposed on them. In situations like that, it is there-fore important to know how a fire spreads so that they can make the best use of their available resources to fight the fire.

Three years after the 1995 earthquake in Kobe, Japan, researchers initiated a program called the RoboCup-Rescue Simulation Project (Kitano *et al.* 1999). The purpose of this project is to simulate a disaster area and the effects that fire outbreaks have on such an area. The latter is based on an intuitive model of how fire spreads over a city and lacks, at least up to now, a deeper understanding of the physical foundation of such a spread. This paper describes research on developing a computer model for the spread of fire and is related to previous research on related problems (Guesgen 2003).

The scenario that we have in mind for our model is the following. Some event has occurred which has disrupted the normal damage prevention infrastructure (i.e., the fire department) so that the fire spreads unhindered through the city. For the most part, the spread of fire is stopped inside the building by construction methods (fire glazing, fire proof treatment of woods, etc.), and the internal sprinkler systems.

In many cities the sprinkler systems are fed by the municipal water supply, so should that fail due to catastrophic damage, buildings will burn uncontrolled. The fire control services are, of course, the next line of defense. However, our assumption is that they are incapable of dealing with the fire for some reason (like it may happen in an earthquake situation). The purpose of the model is to find out how fire will spread in a disaster situation, allowing the fire department and other civil authorities to combat a scenario armed with the knowledge of what places are most prone to fire spread.

To achieve this, the model requires a representation of cities that can be used to measure the likelihood of fire spreading from one building to the next. The model covers the physics of fire and uses a representation that facilitates the calculation of the fire spread.

Although we have conducted this research according to New Zealand regulations and guidelines, we believe that the results are applicable to almost every city around the world.

Regulations and guidelines may differ from country to country, but in essence they are very similar and share a common goal: to prevent the disastrous destruction of the urban environment by fire.

The Physics of Fire Propagation

The physics of fire spread has been well understood by physicists and engineers for some time.

The basic physical property that causes ignition of a building is the radiant heat of neighboring buildings. This is governed by the laws of heat transfer. For two plane radiators (or walls) the intensity (in kW/m^2) on the cooler surface is given as:

$$I_r = k_1 \Phi \varepsilon \sigma \left[(273 + T_e)^4 - (273 + T_r)^4 \right]$$

k_1 is the "glazing factor." For a building without fire-proof glazing, it is assumed that the windows will fall out and fire will project out through the openings. If the building uses fire proof glazing, the windows will stay in place approximately halving the fire output. It is common to ignore flame projection (Buchanan, 2001).

ε is the emissivity, a dimensionless constant, 1.0 is a conservative value.

σ is Stefan's constant, $56.7 \times 10^{-12} \left(kW / M^2 K^4 \right)$

T_e and T_r are the temperatures of the emitting and receiving surfaces.

Φ is the configuration factor, this can be given exactly as

$$\Phi = \frac{1}{90} \left(\frac{x}{\sqrt{1+x^2}} \tan^{-1}\left(\frac{y}{\sqrt{1+x^2}} \right) + \frac{y}{\sqrt{1+y^2}} \tan^{-1}\left(\frac{x}{\sqrt{1+y^2}} \right) \right)$$

where $x = H_r / 2R, W_r / 2R$ and $\tan^{-1}$ return degrees, W_r, H_r are the width and height respectively of the radiating surface, and R is the distance between the two radiators. However an approximation that is often used is

$$\Phi \approx \frac{W_r H_r}{\pi R^2}$$

We will show later that this approximation leads us to a simple method of calculating the configuration factor.

The temperature inside a building can be given by the Standard Fire Curve (ISO 834)

$$T_e = 345 \log(8t + 1) + T_o$$

where t is the duration of the fire, in minutes, and T_o is the initial temperature of the surface.

Now we wish to estimate the duration of the fire(New Zealand Fire Service, 2003). Since we are assuming that the fire occurs in a disaster situation (no municipal water, electricity, etc) then the duration of the fire can be found by

$$t_{fire} = \frac{\left(\Delta H_c \cdot M_{fuel} \right)}{\left(A_{fuel} \cdot Q''_{fuel} \right)}$$

where ΔH_c is the calorific heat value, which is generally about 18MJ/kg, M_{fuel} is the mass of the fuel in the building (in kg), and Q''_{fuel} is the heat release values of the fuel (in MW/m^2) (New Zealand Fire Service, 2003). Some typical values are:

0.5	retail
0.25	offices
1.0	warehouses

A_{fuel} is the surface area of the fuel, this normally can be considered to be a 1:1 ratio with the floor, except when there is rack storage present, but since rack storage will also be dependant on the floor area we can use a building dependant variable to reflect this.

A building with no more fuel will cool as per Newtons Law of Cooling. This of course states that a hot body cools in proportion to the difference between its own heat and the heat of the immediate environment. The cooling rate can be estimated from expert experience by assuming that a building that peaks at a temperature of 4000K and cools over a period of 7 hours to 1% of its peak temperature thus giving the parameters to solve the first order

$$y'(t) = -k(t - T), \quad y(0) = 4000, \quad y(420) = 40$$

$$\Rightarrow y(t) = 3980 e^{-0.0126t} + 20$$

although it is not truly necessary to calculate cooling, since the adjacent building only ignites if the incident intensity is over a material dependant value, for cellulose based material this is12.5 kW/m^2, for plastics 10 kW/m^2, and the maximum intensity coincides with the maximum temperature.

Typically the calculation of the heat intensity between tow buildings relies on the calculation of vents, that is, holes in the firecell here heat is emanating from. Since we are working entirely from GIS data we may not have access to this information. Conservatively we may assume that there are no vents and so no projective flames (Buchanan, 2001).

Representation of Buildings

Of course some method of representing building must be chosen. This must include not only the geometry, but also sufficient data on the nature of the building from the perspective of fire physics.

At the most complex level, the geometry of a building is a vertex set, or complete model. This, however, presents two major issues: the computation time required to perform the calculation and the prohibitive amount of data to be gathered for such a representation to store an entire city. Neither of these is an issue of impossibility, rather a logistic issue.

For the sake of data gathering, it would be advantageous to have a representation that fits easily into current GIS systems, such as ARC info, or GRASS (which did not support any form of urban representation at the time of writing). As most GIS system implement a form of polygon for regions and the footprint of buildings, it makes sense to have buildings represented geometrically as a polygon in a plane, with an associated height value. The height value will then allow us to calculate internal volume and the external area of the walls, for the radiation calculations, as well as giving us an estimate of the number of floors in the building, to be used in fuel calculations.

As for modeling fire we require some information relevant to the simulation. Having described the physics laws describing the rules, we know that some representation of the amount of fuel will be required. From the physics of fire duration, we can find a suitable variable for this

$$t_{fire} = \frac{(\Delta H_c \cdot M_{fuel})}{(A_{fuel} \cdot Q''_{fuel})} = \frac{\Delta H_c}{A_{floor}} \cdot \frac{M_{fuel}}{\rho \cdot Q''_{fuel}}$$

therefore the building dependant variable to reflect the amount of fuel is the ratio of the mass of the fuel, to the ratio of floor area to fuel area times the peak output of the fuel, in units of kgM^2/MW. Why not just encode a burn time altogether? This method allows us to make assumptions based on GIS data, and no GIS data we know of stores the estimated fire duration of every building in the city. The floor area is the area of all floors within the building, since we need height in meters for the geometric calculations, we can find the floor area by dividing the height of the building by 3 (i.e., 3 meters per floor) and multiplying that by the area of the footprint, from the GIS data.

Second, we need to know if a building will catch fire. The existing notion of "fire hazard categories" comes to our aid here. We can define a set of values to be assigned to every building in the GIS. The choice that was taken was five, based on the New Zealand fire code (New Zealand Building Industry Authority, 2001): {NONE, LOW, MEDIUM, HIGH, EXTREME}. Where NONE is any structure that will not ignite (such as a concrete water substation), and the others can be specified as energy per unit area.

Also we need to know at which time a building began burning, obviously if this is during the simulation, it is set by the simulator. Additionally, we may wish to force it to start burning at some specified time, to help us model a particular kind of disaster.

This brings us to a method for representing a disaster. First we wish to understand what kinds of disaster are likely and how we might see these.

Some conceivable fire causing disaster situations are: volcano, earthquake, bomb detonation or aerial bombardment. Naturally, each of these will cause ignition at a point, therefore, a list of "ignition points" is a simple answer to this problem. If we then associate a time with each point, it allows us to model "slow" disasters, such as volcanic eruptions.

Spatio-Temporal Problems Encountered

Now we must cover some issues of spatial representation before we come to our simulator.

First we need to calculate the fire duration. From the fire physics we can use the given formula, altered to reflect the choice of representation. For this system, we chose to make $\frac{M_{fuel}}{\rho \cdot Q''_{fuel}}$ a building specific parameter, and ΔH_c a global parameter.

To calculate the intensity on a building, we need to consider each polygon of the emitting building interacting with each polygon of the receiving building (Fig. 1).

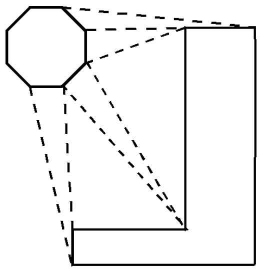

Figure 1. Calculating each polygon

Therefore, we seek some method of approximating, or an efficient method to calculate the configuration factor.

As we have the plane equation for heat intensity, we can use the polygonal representation to create the "visually apparent area" of a building from another. We can do this by checking the radial separation of each vertex in a building from the centroid of the viewing building. Note that in projecting these points from Cartesian space to Polar space, there is a special case if a polygon lies partially above and partially below the horizontal line defined by the Point of View. This can be fixed by maximizing angles above the line and minimizing angles below the line.

A quick analysis of this method shows that if the two buildings are adjacent and facing at an angle of 0, the visually apparent area will be exact to the radiating area (Fig. 2). However, if the buildings are not flat facing or are an odd shape, the approximation begins to lose its accuracy (Fig. 3). Ultimately there is no way of calculating the worse possible case of the approximation, given that buildings may cover any real polygon.

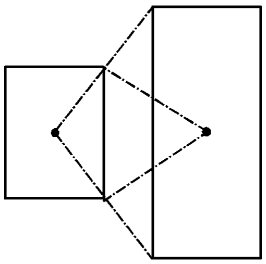
figure 2. Flat-facing buildings

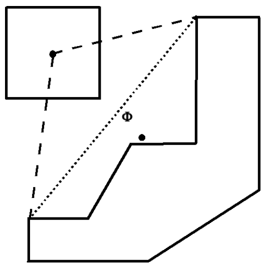

figure 3.Bad approximation for Φ

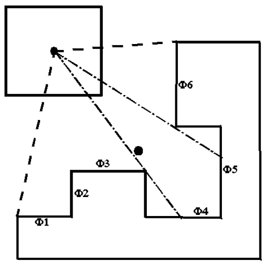

Figure 4. Simple overlapping

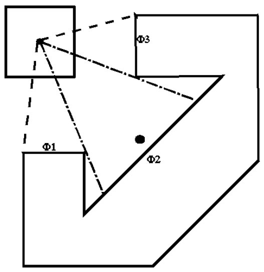

Figure 5. Complicated overlapping

Let us then consider adding some calculation that will not increase the complexity. This can be done, since we have the approximate value for the configuration factor

$$\Phi \approx \frac{W_r H_r}{\pi R^2}$$

As radiant heat is a wave phenomenon, it obeys the rule of superposition, allowing us to linearly add the intensities of multiple radiant faces. We can then add the intensities of all the faces.

The total incident heat is then

$$I_r = \sum_i \Phi_i k_1 \varepsilon\sigma \left[(273+T_e)^4 - (273+T_r)^4\right]$$
$$= k_1 \varepsilon\sigma \left[(273+T_e)^4 - (273+T_r)^4\right] \cdot \sum_i \Phi_i$$

Thus we need to find the configuration factor of each facing line segment between the maximally separated points.

To do this, we look at our polygon, a set of indexed points stored in a cycle. We find which of these is closer to the Point of Vision, then traverse the polygon along that route, till we reach the last point. For each face, we calculate the configuration factor, and add it to a running total. The calculated configuration factor can then be used to find the incident intensity.

This has a complexity of O(n), and since this is performed this immediately after finding the separation angles, also O(n), it does not increased the complexity of finding the radiant heat.

psuedocode to calculate the configuration factor:

```
Set configuration_factor = 0
Set current_point = point of minimum
  separation
Set end_point = point of maximum separation
IF the next point after current_point is
  closer than the point after it
 Set direction to positive
ELSE
 Set Direction to negative
END IF
WHILE current_point is not end_point
 Set next_point = the point following
   current_point in direction
 ADD length of current_point to next_point
   / (pi* distance from the center of the
   face to the other building) TO
   configuration_factor
END WHILE
Use configuration_factor to calculate
  radiant heat
```

However, this does not work if faces overlap (Fig. 4). We then need to calculate which faces are facing the receiving building. This can be done by examining the angles between adjacent faces to find if the sectors they describe overlap from the point of view of the receiving building, and using a ray cast from the point of vision, and using the remaining part of the face it intersects. Take the minimum angle point, and find the next two point after it. If the angles of the points to the monotonically increasing, they do not overlap, if they are not monotonically increasing, then cast a ray from the point of view to the highest angled point of the three points being examined

But this is still insufficient for complicated overlapping (Fig. 5). Then the common graphics method of Binary Space Partition Trees (BSP-tree) (Krishnaswamy R, 1990), modified slightly, can be utilized to calculate the configuration factor.

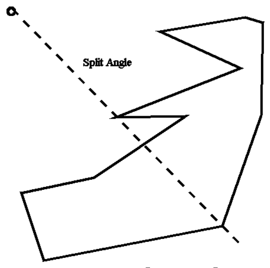

Figure 6. Split angle in the plane

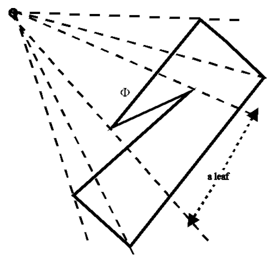

Figure 7. Configuration factor at a leaf in the BSP-tree

Recall that a BSP-tree is constructed by splitting polygons about a plane built from the point of view, and a vertex in the dataset, chosen to create similar sized polygon set. The two resulting polygon sets are further broken down by building a binary partition, until there are no vertices left with which to split the polygons. The top polygon is then added as a leaf of the tree.

Furthermore, since the domain is actually two-dimensional, the binary partitioning will be performed by splitting the line segments across a ray from the point of view (Fig. 6). The configuration factor is calculated by recursively applying the binary partition to the two new sets of line segments, and adding the two returned values for the configuration factor.

For the configuration factor calculations, we do not need to store the tree, so we can build a recursive algorithm to find the configuration factor of the top line when a “leaf” is reached (Fig. 7).

Pseudocode for a modified BSP tree to calculate the configuration factor

```
IF there are only angles for all the polygons
 Set current_line = any line segment
 Set top_line = any line segment
 FOR all polygons in the set
  IF current_line is closer to the point of vision than top_line
   Set top_line = current_line
  END IF
 END FOR
 return the configuration factor of top_line
ELSE
 select the median angle
 FOR each line
  IF an endpoint is on the ray
   continue
  ELSE
   IF the line intersects the ray
    break the line around the angle and
     add the lines back to the list
   END IF
  END IF
 END FOR
 FOR each line
  IF the angles are both less than or equal to the split angle
    add the line to the righthand list
  ELSE
   add the line to the lefthand list
  END IF
 END FOR
 Calculate the BSP-tree for the lefthand list
 Calculate the BSP-tree for the righthand list
 return the sum of the calculated configuration factors
END IF
```

This provides us with a much more accurate method for calculating the configuration factor of any building, and naturally agrees with the approximation for the configuration factor, given again below:

$$\Phi = \frac{A_r}{\pi R^2}$$

While this does add to the complexity, it will still be lower than using every face as before, since the same cases will have to be treated for each polygon interaction, if we were to not use this approximation.

Now the treatment of some minor issues:

Whether or not a building is adjacent to another is also required. To do this simply we can check if the centroid of a building is within a certain distance of the centroid of another building. To determine this distance, we can use the size of the building, by taking the distance from its centroid to the furthest point in its footprint from the centroid, and taking the distance to check for as, say, three times that.

Two other problems that arise from the representation, and the calculation required are the need to find the area of a building from its polygonal information to calculate the burn time, and to find whether a point is inside an arbitrary polygon, to model our ignition scheme.

To find if a point is inside an arbitrary polygon, we can use the so-called “half ray algorithm”, which operates by finding the number of arcs of a polygon that a ray from the point intersect.

Pseudocode for the Half-Ray Algorithm:

```
Set intersection_count = 0
Cast a ray from POINT in any direction
FOR all arcs of POLYGON
 IF the arc intersects the ray from POINT
  Increment intersection_count
END IF
END FOR
IF intersection_count is even
 POINT is outside POLYGON
ELSE
 POINT is inside POLYGON
END IF
```

We also need an efficient algorithm for the area of a Polygon, for which we used the following alorithm

Pseudocode for finding the area of a polygon:

```
P is as set of n points cyclically stored, representing a polygon
Set P_n+1 = P_0
Set total1 = 0
Set total2 = 0
FOR i = 0 to n+1
  Increment total1 by x(P_i) * y(P_i+1)
  Increment total2 by y(P_i) * x(P_i+1)
END FOR
Set area = (abs(total1-total2)) / 2
```

The Method of the Simulation

Now that we have a representation and can resolve the spatial information needed to use the physics, we can thus define our simulator:

We create two lists, one to hold all the unignited buildings in the city, and one to hold the burning buildings.

We also create a list of ignition points and the corresponding times.

We examine each unburnt building, and take its immediate neighbours. Calculate the total heat intensity incident on the building from all neighbouring buildings that are burning, then check that it has not crossed the threshold defined by the Fire Hazard Category. If it does, add this building to the list of burning Buildings

There are no final conditions, other than an imposed time limit or that all existing burning buildings have burned out.

The basic simulation method:

The update cycle for the simulator is then:

```
Set time = 0;
FOR each unburnt Building
FOR each Building adjacent to it
IF the intensity falling on the
  building is greater than the
  tolerable amount OR the building is
  on an ignition point when that point
  fires
Move this building from the list of
  buildings to the list of burning
  buildings
END IF
END FOR
Increment time
END FOR
```

Future Possibilities

The expected deployment of this algorithm would be in a decision support system for use in the field by human or automated fire fighters, by providing short term estimation of fire spread, i.e., which buildings adjacent to a current blaze should be evacuated first etc. Our preliminary testing has shown this to be a better use, as the simulation becomes noisy and unreliable after some time.

A full statistical analysis of the burn rate might lead to a relationship between the buildings parameters (mass and geometry) and its cooling rate. While this is not strictly necessary to the simulation, it would be useful to embed in a decision support system for the use of the civil fire authorities.

This is sufficient to produce a simulation and the graphical representation of the fire as it spreads, but we would like to have some analysis to actually discern useful information about the fire. The method of testing checks the percentage of the total floor area of the city that remains unburnt at each time step, based on the performance metric used in the RoboCup simulation (Semi-Final Rule and Evaluation for 2001 RoboCup-Rescue Simulation Leagues). This only gives information about which ignition points are more volatile.

While this would prove useful to Civil Authorities, it is also possible to find which parts of the domain have a high propagation speed. This could be done by logging information about when a building ignites, and how close the building that caused ignition is. Although no method for inference of the “danger zones” has been suggested so far, it is reasonable to think that some form of statistical analysis could be employed to detect close points of fast propagation.

Acknowledgements

We would like to acknowledge the generous support provided by the following people and groups:

- The New Zealand Fire Department
- Simon Davis, Auckland Regional Fire Engineer.
- Auckland University RocoCup Rescue Team
- Dr Mike Barley, Computer Science Department, University of Auckland.
- The Association for Computing Machinery
- Delorie Software, www.delorie.com

References

Buchanan, A. H. 2001. *Fire engineering design guide*: Centre for Advanced Engineering, University of Canterbury

Guesgen, H. 2003. When regions start to move. In *Proc. FLAIRS-03*, 465–469.

ISO 834-1. 1999.

Kitano, H.; Tadokoro, S.; Noda, I.; Matsubara, H.; Takahashi, T.; Shinjoh, A.; and Shimada, S. 1999. RoboCup Rescue: Search and rescue in large-scale disasters as a domain for autonomous agents research. In *Proc. IEEE InternationalConference on Systems, Man and Cybernetics*, volume VI, 739–743.

Krishnaswamy, R; Alijani G; Su, S, 1990, On the Construction of Binary Space Partition Trees, Proceedings of the 1990 ACM annual conference on Cooperation

New Zealand Fire Service, 2003, *Fire Fighting Code of Practice*

New Zealand Building Industry Authority, 2001, *Building Amendment Regulations Section C, Fire Safety*

Semi-Final Rule and Evaluation for 2001 RoboCup-Rescue Simulation Leagues, 2001