18th World IMACS / MODSIM Congress, Cairns, Australia 13-17 July 2009
http://mssanz.org.au/modsim09

Untangling the Prometheus Nightmare

Bose, C.[a], R. Bryce[b], and G. Dueck[c]

[a] Department of Mathematics and Statistics, University of Victoria, CANADA
[b] Heartland Software Solutions Incorporated, CANADA
[c] Department of Mathematics and Computer Science, Brandon University, CANADA

Abstract: Numerous bush and forest fire simulators have been developed in the last two decades based on elliptical spread and Huygens' principle. Unfortunately, all such implementations are plagued by topological complications. For example, sampling issues on the evolving front, represented in the Canadian simulator Prometheus as a set of polygons, evolving under the differential equations derived by Gwynfor Richards from Huygens' principle, lead to tangling and other non-physical singularities. In order to maintain stability of the numerical scheme, and to produce realistic fire fronts, these artefacts must be systematically removed between time steps. In the literature on interface tracking, this is called delooping or untangling the computed front.

Recently, an automated untangling routine has been developed for Prometheus based on the so-called 2-colour Theorem. Not only is this approach more intuitive than previous algorithms (which were based on scan lines and winding number calculations), it has proved to be more accurate and faster on all test cases employed by Prometheus developers (from 10 to 90 percent speedup compared to previous generation codes, depending on the examples). It is based on a concise and easy-to-implement set of rules that do not introduce the many special cases that previous methods required.

This report presents a brief review of fire simulation models in general, and on their various approaches to untangling in particular. We describe the mathematical foundations for our new method and give a brief overview of the programming issues that arise in implementation. Finally, we report the results obtained by the new model on a test database of fires maintained by Prometheus developers. Good performance is obtained both in terms of run times and in the ability of the code to produce realistic fire fronts without operator intervention.

***Keywords*:** Forest fires, spread models, marker method, polygon clipping

1. FOREST FIRE SIMULATION

1.1 Huygens' Principle and Prometheus

Data collected from wildfires and experimental burns over a wide range of fuel types supports the empirical observation of elliptical fire spread shape under homogeneous conditions of fuel load, moisture and wind (Van Wagner, 1969). This basic principle can be incorporated in a fire spread simulator by invoking Huygens' principle wherein the fire front is represented at time t by a collection of point ignition sources (firelets) that evolve independently, using locally derived spread rate data over a small time step Δt. Consequently, at time $t + \Delta t$ the new fire front is the envelope of these many expanding elliptical firelets. Figure 1 shows a pictorial representation of the model where the blue curve represents the new front, evolved from the black elliptical front after one time step.

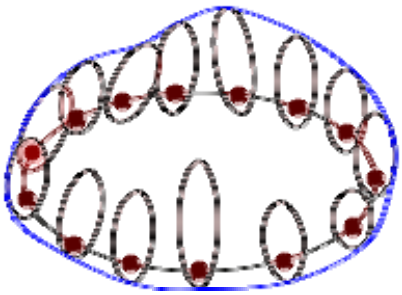

Figure 1. Huygens Principle

Early attempts to implement this model on a computer were reported by Sanderlin and Sunderson (Sanderlin et al., 1975) and Anderson, Catchpole, De Meestre and Parkes (Anderson et. al., 1981). In these investigations the envelope was derived from the expanding ellipses by a direct geometric procedure. Richards (Richards, 1990, Richards, 1999) refined the use of Huygens' principle by deriving a system of continuous differential equations for spread of the fire front, again using the basic observation of an elliptical local spread shape. This has the advantage of incorporating the fire front modelling problem into the general literature on interface tracking and hyperbolic partial differential equations (Sethian, 1999). Thus, a number of techniques are available for numerical simulation. For example, the numerical procedure suggested in Richards (Richards, 1990) models the fire front at time t as a closed polygonal path with finitely many vertices. Using difference equations derived from the Richards equations, the vertices of this polygon are translated to new points representing the vertices on the new front at time $t + \Delta t$. The updated front is defined as the polygonal path passing through these new vertices. In the interface literature, this is known as the *marker method* or *bead method* (Sethian, 1999).

Prometheus, the Canadian wildland fire growth simulator (Tymstra et al., 2009) is a fully-functional, field tested computer program based on Richards' equations, the marker method for front propagation and incorporating the Canadian Fire Behaviour Prediction (FBP) system (Forestry Canada Fire Danger Group, 1992) as input for local spread rates. The FBP assumes elliptical spread.

Unfortunately, all the approaches discussed above suffer from topological complications. The essential problem is that variation in the local velocity field combined with spatial and temporal discretisation leads to tangles and singularities in the computed front. In order to use this method in an iterative scheme, it is essential to identify and remove these artefacts by post-processing the computed front before the next iteration. In the front tracking literature, this is known as *delooping* or *untangling* (Sethian, 1999); we adopt the latter terminology. An early attempt to address the tangling issue appears in the work of Knight and Coleman (Knight and Coleman, 1993). There, geometric derivation of the new front is modified in regions of high curvature so as to reduce the generation of small scale tangles (the term *rotation* is used in that article; *swallowtail* also appears commonly in the front tracking literature for this type of artefact). Rotations that do appear despite these modifications are removed by a simple procedure, based on the local nature of the artefact. Overlaps of widely separated parts of the fire front (for example, as in the horseshoe fire shown in

Figure 6) are also treated by a heuristic clipping routine that is dependent on the mild topological complexity of the artefact.

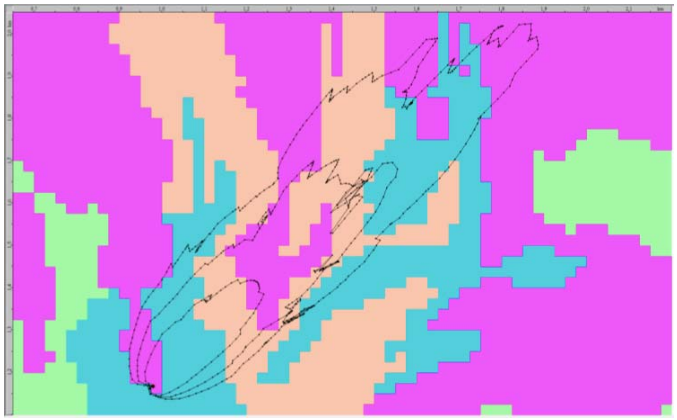

Figure 2. Front evolution via Prometheus 4.5: Dogrib fire

1.2 Untangling in Prometheus prior to Version 5.0

All versions of Prometheus have incorporated some level of automated untangling. Prior to version 5.0 the algorithm for untangling was based on a winding number[1] calculation performed via the scan line method as proposed in (Richards et al., 1995). Once the winding numbers of vertices[2] are determined, various heuristics are applied to assign each vertex either an *active* or *inactive* status. The front is simplified by removing strings of inactive vertices and associated edges. Unfortunately, artifacts in the form of loops and crossings and stranded inactive interior polygonal segments often remain after the version 4.5 untangling. An example (Dogrib fire simulation) is shown in Figure 2.

1.2 Untangling in Prometheus after Version 5.0

Prometheus 5.0 introduced a new module for vertex management containing an untangling routine based on the 2-colour Theorem. This approach is both simpler and faster than previously implemented algorithms and gives more realistic results. Applying the new algorithm (the current release, Version 5.3) to the Dogrib Fire data generating Figure 2 yields the untangled plots in Figure 3. Note that these fronts have fewer topological complications while retaining main features of the front as computed by Version 4.5.

2. UNTANGLING VIA THE 2-COLOUR THEOREM

2.1 Polygons and Crossings

The fire front consists of one or more polygons, each determined by an ordered list of vertices $P_i = (x_i, y_i)$, $i = 1,2,\ldots,N$, where $P_1 = P_N$, closing the polygon. The edges of each polygon are directed line segments in the plane connecting P_{i-1} to P_i. In order to simplify our discussion, we will consider only the case of a single polygon in this short article. An *edge crossing* is any intersection of two distinct edges, excluding consecutive edges joined at their common vertex. This includes the possibility of a vertex meeting an edge or two distinct vertices occupying a common point. If a polygon has no edge crossings, it is said to be *simple*.

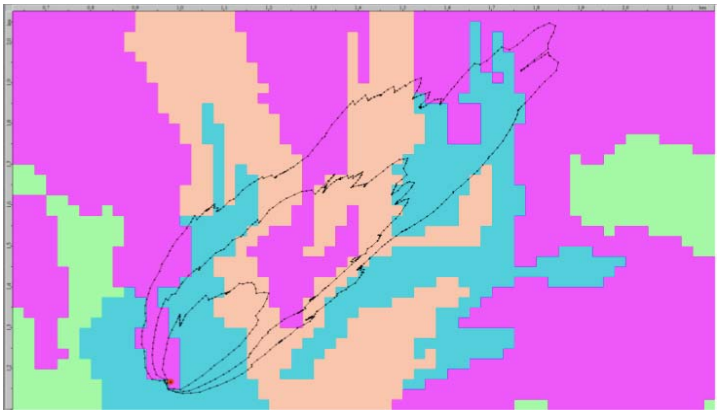

Figure 3. Front evolution via Prometheus 5.3: Dogrib fire

We assume that at time step $t = T$ the polygon is simple, and positively

[1] The winding number of an oriented curve C with respect to a point P is the number of times the location vector from P to a point X on the curve rotates around P, as X traverses the curve C in positive direction. See (Krantz, 1999), for example.

[2] Strictly speaking, winding numbers cannot be computed at points on the front -- the Prometheus implementation uses an extended definition of winding number to handle vertices on the fire front.

oriented so that the burned area lies to the left. As discussed above, the propagation algorithm acts on each vertex $P_i(T)$, moving it to $P_i(T+\Delta t)$ according to the (discrete) Richards' equations. This presents us with a new oriented polygon with edges connecting $P_{i-1}(T+\Delta t)$ to $P_i(T+\Delta t)$ which we generally expect to be tangled, i.e. non-simple. Our goal is to replace this with a finite union of simple, disjoint polygons such that the active fire front is accurately represented by the union of the edges and with the burned area laying to the left with respect to the orientation.

2.2 Resolving Crossings

The algorithm is based on the following simple observation. Every crossing in the polygon creates an ambiguity in the assignment of the status `burned' or `unburned' to regions near the crossing. Figure 4 makes the problem clear, where the crossing of two oriented segments is shown on the left. Burned areas are indicated by black shading to the left and unburned (or currently burning) by red shading. The crossing determines (locally) four disconnected components. While two of the components are consistently labelled as burned or unburned, the other two necessarily have ambiguous labels.

We remove the crossing by creating two new vertices at the crossing point, labelled R_1 and R_2 and paths as shown in Figure 4. In this figure we have labelled regions near the crossing with 0's and 1's in order to distinguish two methods of resolution. Edges are re-oriented so as to resolve the ambiguity in burned/unburned regions and leave the region labeled 1 to the left of the oriented curves.

Such a labeling of the connected components forming the complement of the polygon with 0's and 1's forms an essential part of our algorithm. We are assisted in this by the following elementary result.

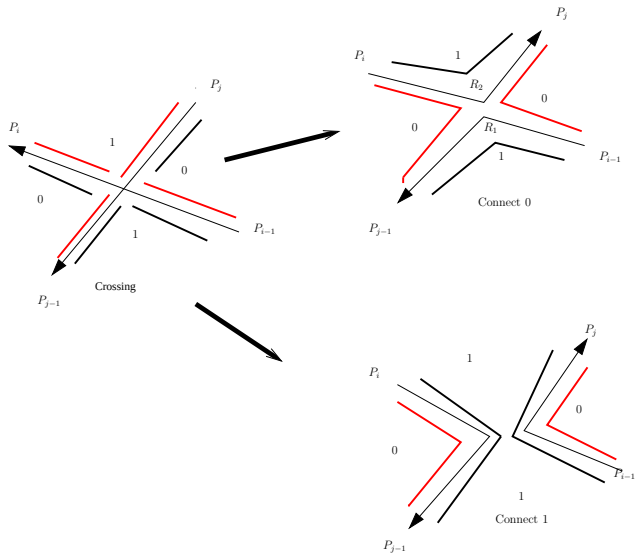

Figure 4. Resolution of crossing point, geometric version.

2-Colour Theorem (Hunter) *Every piecewise continuous, closed, planar curve admits a 2-colouring*[3].

We choose a 2-colouring such that the infinite connected component is assigned the label 0. Then we have

Rule 1. Uncrossings are chosen so as to connect regions having label 1. That is, make the lower choice in Figure 4.

Each uncrossing induces a reordering of a portion of the polygon. In the Figure 5 we show schematically how the two possible reorderings arise. Note that it is possible that after application of the crossing rule, we are left with two polygons as shown by the lower schematic in this figure. These can recombine to a single polygon later through application of the crossing rule, or not. In any event, after each application the total number of crossings (counting both self-crossings and crossings between distinct polygons) is reduced.

[3] A 2-colouring is an assignment of labels (which we may as well assume are 0 and 1) to connected components of the complement of the curve in such way that adjacent components sharing a common segment of the curve get distinct labels. Clearly such a colouring is unique up to a flip $0 \leftrightarrow 1$. If the polygon is simple, this is the celebrated Jordan Curve Theorem: There are precisely 2 connected components, the interior (the bounded component) and the exterior of the curve (the infinite component)

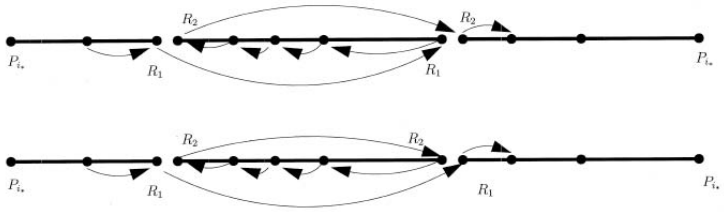

Figure 5. Resolution of crossing point, data structure version.

Theorem 2. *Finitely many applications of the Rule 1 will result in a finite number of disjoint (non-crossing) oriented simple polygons which divide the plane into a single bounded component with label 1 and a finite number of connected components labeled 0. Polygons are positively oriented with respect to the 0-1 labeling.*

Proof. (Sketch). Obtain the 2-colouring as in Theorem 1, arranging the colouring so that the infinite component is labeled 0. Repeated application of Rule 1 at each crossing ensures that after untangling there will be a single bounded and connected component with label 1. Locally, edges have been re-oriented to satisfy the local rule that the label 1 lies to the left of the curve. The 2-colouring will ensure that this rule is satisfied globally. Since there are no crossings, each component is bounded by a simple polygon.

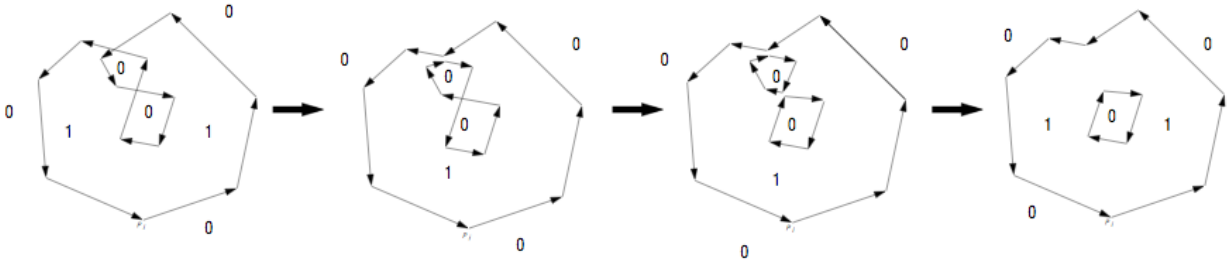

Figure 6. Horseshoe fire

2.3 Clipping the Untangled Polygons

Clipping is performed on the output from the previous section. The outer polygon is defined to be the one adjacent to the infinite connected component.

Rule 2. The outer polygon is retained.

The remaining polygons are retained or removed (clipped) according to

Rule 3. Inner polygons are removed if the direction of one or more edges in that polygon has been reversed when compared to its direction in the original polygon.

Figure 6 shows a simple application of the algorithm in a horseshoe fire. The Prometheus implementation untangles one vertex at a time, starting from the vertex P_*, the bottom-most vertex, and working around the polygon in the natural orientation. The untangling leaves two active fire fronts: the outer expanding front and an inward burning `island'. One of the untangled polygons has been removed by Rule 3 (the overlap in the horseshoe). This is the correct untangling of the horseshoe fire.

2.4 Connections to the literature on clipping.

Techniques similar to the untangling algorithm above are applied to the problem of *polygon clipping* where one (closed, possibly non-simple) polygon (the clip) is used to remove an area from another polygon (the subject). This setting, while natural and extremely important in computer graphics, is too restrictive to handle the complex polygons arising from front propagation. Existing methods we have considered are Vatti's

algorithm (Vatti, 1992) and its more modern extended heuristics (in Murta, 1999) and Weiler's algorithm (Weiler, 1980), which assumes a 'clockwise' orientation on each polygon. We have compared our problem with the Greiner-Hormann algorithm (Greiner et.al., 1998) and a recent extension of their method due to Kim and Kim (Kim et al., 2006). Of the three, this is closest in spirit to our 2-colour solution but really only applies to the clip and subject setting. None of these methods appear to be easily implemented for our problem within their design parameters.

3. Performance

3.1 The Prometheus Nightmare

One early success with Version 5.0 was correct resolution of a simple tangle known as the Prometheus Nightmare. This figure-eight artefact had inexplicably resisted solution by all prior routines and is shown in Figure 7. The fire is assumed to be burning from the lower right toward the upper left in this scenario. The point labeled A should be part of the active fire front but is removed by the Version 4.5 untangler. It is retained, however by the 2-colour untangling routine of Version 5.0 and later.

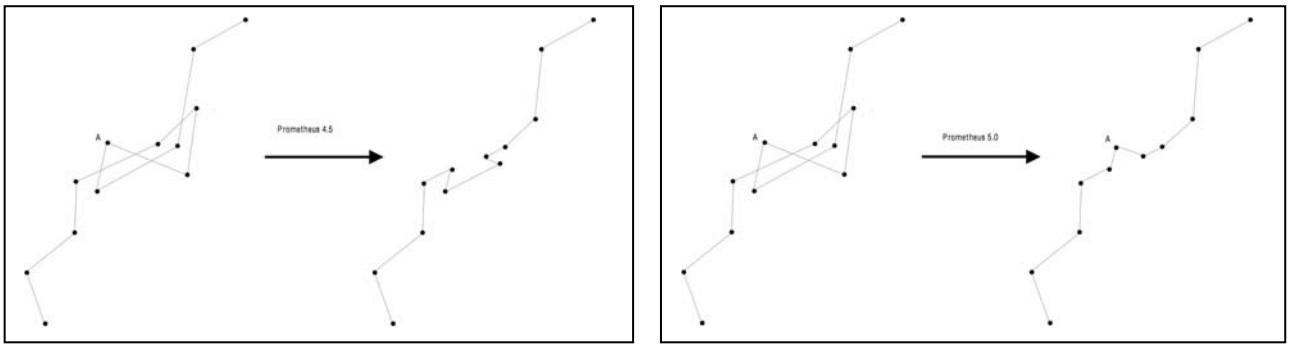

Figure 3. The nightmare untangled under (left) Version 4.5 and (right) Version 5.0

3.2 Timing

The new algorithm is simpler and faster than previous vertex management schemes based on winding number. We present some comparative results on a set of test problems. The first two columns contain a count of the cumulative number of vertices arising in the simulation.

Table 1. Run times (in seconds) for standardized test datasets

Test Dataset	# of vertices in simulation 4.5.2	# of vertices in simulation 5.3.0	Version 4.5.2 run time (seconds)	Version 5.3.0 Run time (seconds)	Percentage improvement: # vertices/time
Grande Cache	3367621	206543	765	80	94/89%
Chisholm	34130	30700	40	35	10/13%
Mount Somers	83091	56555	50	40	32/20%
Dogrib	143577	115431	45	20	20/56%
Quebec Lakes	15955515	1673169	375	140	79/63%
Jasper	7972214	1673169	1950	210	76/89%

It should be noted that startup overhead dominates in short simulations and this may reduce the percentage speed gains in small fires. Field users report increased speed of approximately 30% in Version 5.0 over previous releases. This speed gain is despite higher overhead from more elaborate routines written into other parts of Version 5.0. It seems reasonable to attribute some of the reduction in vertices processed to the more efficient untangling routine, but due to the nature of the program, it has not been possible to make a direct comparison between versions on the untangling computations as distinct from other parts of the simulation. The timings reported above were performed using a computer with an Intel Pentium P8400 processor running @ 2.26 GHz with 4GB RAM.

4. CONCLUSIONS

The algorithm is a significant improvement over previous version untanglers. It is based on a concise and easy-to-implement set of rules that do not introduce many `special cases' compared to previous methods. However, it is not infallible. For example, it has been noticed that it fails to give correct results on a simple triple-loop example. The difficulty appears to be in the clipping heuristic contained in Rules 2. and 3. and not in the untangling part of the routine. It is possible that a modified clipping heuristic using the more detailed winding number (the 2-colouring can be determined by winding number, but throws away much of the information) could address examples such as this. Experiments were performed with the full Prometheus package where it is not possible to quantify exactly how much time is spent in untangling. It would be interesting to make detailed comparisons between the two versions untangling schemes as separate from the rest of the program.

ACKNOWLEDGEMENTS

The authors are pleased to acknowledge support from the Alberta Sustainable Resources Department, MITACS, PIMS and the GEOIDE NCE. We wish to thank C. Tymstra for his support and helpful comments during the course of this work.

REFERENCES

Anderson, D.H., Catchpole, E.A., De Mestre, N.J. and Parkes, T (1982), Modelling the spread of grass fires. *J. Aust. Math. Soc.* (Series B), 22, 451-466.

Forestry Canada Fire Danger Group, Development and structure of the Canadian Forest Fire Behaviour Prediction System. Inf. Rep. ST-X-3. For. Can., Ottawa, Ontario, 1992.

Greiner, G., Hormann, K. (1998), Efficient Clipping of Arbitrary Polygons, *ACM Transactions on Graphics*, 17(2), 71-83.

Hunter, D., The two-colour theorem. (http://blue.butler.edu/~phenders/sigcse2004niftyworkshopMAA_PREP_Workshop_2004/Two\%20Color%20Theorem/twocolor.pdf)

Kim and Kim (2006), An extension of polygon clipping to resolve degenerate cases, *Computer-Aided Design and Application*, 3, 447-456.

Knight, I and Coleman, J. (1993), A fire perimeter expansion algorithm based on Huygens' wavelet propagation, *Int. J. Wildland Fire*, 3, #2, 73-84.

Krantz, S. G. (1999), The Index or Winding Number of a Curve about a Point, Birkhäuser, §4.4.4 in Handbook of Complex Variables, 49-50, Boston.

Murta, A. (1999), A General Polygon ClippingLibrary. http://www.cs.man.ac.uk/~toby/alan/software/gpc.html.

Richards, G. D. (1990), An elliptical growth model of forest fire fronts and its numerical solution, *International Journal for Numerical Methods in Engineering*, 30, 1163-1179.

Richards, G. D. and Bryce, R. W. (1995), A computer algorithm for simulating the spread of wildland fire perimeters for heterogeneous fuel and meteorological conditions, *International Journal of Wildland Fire*, 5(2), 73-79.

Richards, G. D. (1999), The mathematical modeling and computer simulation of wildland fire perimeters for heterogeneous fuel and meteorological conditions, *International Journal of Wildland Fire*, 9(3), 213-221.

Sanderlin, J.C. and Sunderson, J.M. (1975), A simulation for wildland fire management planning support (FIREMAN). Vol. 2. Prototype models for FIREMAN (Part II): Campaign fire evaluation. Mission Research Corp. Contract 21-343, Spec. 222. 249.

Sethian, J. (1999), Level Set Methods and Fast Marching Methods, Cambridge University Press.

Tymstra, C., Bryce, R.W., Wotton, B.M. and Armitage, O.B. (to appear 2009), Development and structure of Prometheus: the Canadian wildland fire growth simulation Model, Nat. Resour. Can., Can. For. Serv., North. For. Cent., Edmonton, AB. Inf. Rep. NOR-X-417.

Van Wagner, C. E. (1969), A simple fire growth model. For. Chron., 45, 103-104.

Vatti, B. R. (1992), A generic solution to polygon clipping, Commun., ACM 35, 56-63.

Weiler, K. (1980), Polygon comparison using a graph representation, SIGGRAPH '80, 10-18.