College of Engineering, Mathematics
and Physical Sciences

Change Identifiers

Maximillian Dupenois

Supervisors: Antony Galton and Jovisa Zunic

November 25, 2010

Contents

Chapter 1

Introduction

Chapter 2

Background

2.1 Footprints

There is a fairly large body of work about the generation of footprints, publications from as early as 1973 ([10]) presenting a variety of different algorithms to create representational shapes from dot patterns. Amongst this there are surprisingly few that examine the footprints created in a comparative fashion. Also conspicuous by its absence is a systematic approach to determining the quality of the produced footprint, **(author?)** [7] makes significant inroads in to both determining how 'good' a footprint is and why this is difficult to judge.

The rest of this section consists of analysis of some of the existing literature in chronological order.

(author?) [10] presents an algorithm, since called the 'Jarvis March' to generate the convex hull of a dot pattern. The convex hull is almost a base level of footprint algorithm, it is easily computable and has distinct mathematical properties. Importantly the convex hull is unique for any particular dot pattern. This paper was amongst the first to give an efficient algorithm for its computation and is such amongs the first to attempt to provide a representational shape for a dot pattern.

The convex hull is not without it's problems as a representation.

COMMENT: [*INSERT FIGURE SHOWING CONVEX HULL LOSING CONCAVITY*]

As can be seen in **COMMENT:** [*figure*]the convex hull can potentially lose information about the shape. **COMMENT:** [*From here to cosit paper reference is basically plaigarising myself, need to rewrite*]An algorithm capable of reaching a better fit representation is a non-trivial problem and one of the earliest, and much-referenced, papers on the subject is by **(author?)** [6]. The method produces straight-line graphs called α-shapes, obtained from a generalisation of the convex hull. For a set S the convex hull can be considered to be the intersection of all closed half-planes that contain all the points of S. The *α-hull* is obtained by using closed discs of radius $1/\alpha$ instead of half-planes; the α-shape is derived from this in a straightforward way. The authors do not discuss any principled way to choose the appropriate α for the type of shape required.

(author?) [3] present two methods for generating a footprint, called the *external shape*, from a dot pattern. Although they use the term 'dot pattern' they make no distinction between points and dots. For the first method, a grid of squares of side-length s is drawn on the plane, and the union of all grid-squares containing at least

one of the dots is returned as the footprint, called the *s-shape*. For the r-shape they inscribe a disc of radius r round each dot, and draw an edge connecting any pair of dots whose discs intersect in a point not contained in any of the other discs. These edges provide an outline which, in our terms, may be regarded as the boundary of the footprint. As with the α-shape, no principles are given for selecting appropriate values of r or s.

(author?) [9] propose a 'split and merge' method for generating footprints. This method starts from the convex hull and attempts to refine it to a shape more closely resembling what they refer to as the *underlying shape*. The method consists of three separate algorithms (four if the convex hull algorithm is included): *splitting*, *isolation*, and *merging*. This is one of the few algorithms that provides a way of *aiming* for a particular shape without having to re-run the algorithm with different parameters, so long as the user is able to identify a desired maximum area or number of sides just from a cursory examination of the dot pattern. Again the authors say little about the quality or type of footprint they generate.

(author?) [1] developed the *Dynamic Spatial Approximation Method* (DSAM). This system takes in both the dot pattern of the region to be found and the dot pattern of the area known to exist outside the region. It builds a Voronoi diagram based on these coordinates and takes the union of all the cells which contain an 'interior' point as its footprint. This work pays more attention than many in the area to the quality of footprint produced; this can be assessed in terms of how closely the region found fits the expected region. The existence of a contextually determined target shape differentiates this paper from others in the field.

(author?) [2] follow on from **?**]. However, they adapt DSAM to use Delaunay triangulations in conjunction with a system for finding point locations using web queries. They call this adaptation *the recolouring algorithm* and use it to generate boundaries for imprecise regions. Much like the DSAM this system has a target shape and, as such, this paper has more analysis of the footprint found than much of the field.

(author?) [8] propose two methods for finding footprints. The first method is a generalisation of the Jarvis March ('gift-wrapping') algorithm for convex hulls. The idea behind the Jarvis March is simple. From an origin point outside the dot set a radial half-line is swung in an arbitrary direction until it meets one of the dots. This dot is made the new origin point from which a radius is swung in the same direction as before until it meets another dot. This is repeated until the first dot is encountered again; the sequence of dots encountered in this way form the vertices of the convex hull. Dots are removed from consideration if they have already been marked as being on the convex hull or if they lie within the area enclosed by the dots encountered so far. The 'Swinging Arm' algorithm is similar except that it uses a line-segment of some predetermined length instead of a half-line. The second method starts with the Delaunay triangulation and successively removes the longest external edge, subject to constraints of maintaining connectedness and regularity, until either some predetermined minimum length is reached, or no more edges can be removed. The authors note that there can be no uniquely 'optimal' footprint when the application context is considered to be general. The paper proposes nine criteria which may be used for evaluating footprint algorithms with respect to different application contexts, although little is said about any actual applications.

(author?) [11] present a 'Concave Hull' algorithm. Like the Swinging Arm, Concave Hull is also derived from the Jarvis March algorithm, its difference being that it always selects the next vertex from the k nearest neighbours of the current vertex. This is the crux of the algorithm's effectiveness: by having a non-contextual integer as the variable that restrains the hull algorithm, they have a default base value from which they can run the algorithm (i.e. $k = 3$); if this fails to produce a footprint that satisfies the criteria (having no intersecting lines and containing all the points) then the algorithm is run with increasing values of k till such a footprint is created. Like most

of the other authors they pay little attention to the quality of the footprint in relation to any application type, though they do mention the criteria given in [8]. Like the split and merge method [9], the Concave Hull algorithm requires some pre-processing of dots, using the Shared Nearest Neighbour (SNN) algorithm to determine any separable groupings in the dot pattern prior to running the algorithm. Like Garai and Chaudhuri they do not take account of this pre-processing algorithm in determining the computational complexity of their own.

(author?) [4] provide a fuller account of the Delaunay-based method introduced in [8], now called the χ-algorithm. This paper includes a discussion of the footprint's properties, and how these are directly tied to the method by which it is created. More attention is paid to the choice of the length parameter l. There are practical limits on l for any triangulation (if it is too large then no lines will be removed, if it is too small too many will be removed) and consequently l can be normalised. Duckham *et al.* propose using this normalised parameter (λp) to find a starting value which should achieve what they call a *characteristic shape* for many, if not all, dot patterns. While they conclude that there is no λp that always produces a "good" characterization, the fact that they spend time considering this is unusual within the field. Unlike **(author?)** [11] and **(author?)** [9], Duckham *et al.* do not discount the pre-processing (in this case computing the Delaunay triangulation and sorting the edges) when determining the complexity of the algorithm.

(author?) [7], instead of proposing an algorithm, searches for objective criteria for evaluating the acceptability of any proposed footprint in relation to the 'perceived' shape of a dot pattern. The paper notes that in most of the published work, "while lip-service is generally paid to the fact that there is no objective definition of such a 'perceived shape', little is said about how to verify this, or indeed, about exactly what it means". Restricting attention to footprints in the form of *polygonal hulls*, simple polygons having vertices selected from the dot pattern, all the other dots being within the interior, the paper presents evidence that while a dot pattern may have several equally acceptable perceived shapes, they all represent optimal or near-optimal compromises between the conflicting goals of simultaneously minimising both the area and the perimeter of the hull.

(author?) [5], suggests a method for classifying the footprints. Unlike **(author?)** [7] it does not look at their 'fitness' but approaches the subject from a desire to be able to describe algorithms by the types of footprints they can create. The paper notes that the context in which the algorithm is being used determines the type of footprint that is satisfactory. With this in mind it proposes a method of using the application specific knowledge to limit the choice of algorithms for any particular user requirement. The classification bears some similarity to the set of criteria proposed by **(author?)** [8] for evaluating the footprints produced by different algorithms.

2.2 Dot Patterns

Examining dot patterns has generally been within the vfield of geospatial information. However, if we move away from real-world phenomena, we can imagine that any data that can be represented on a 2-dimensional plane (e.g., classification data, multi-objective optimisation) can be viewed as a dot-pattern. This leads to a daunting amount of possible literature to examine so the analysis given is by no mean exhaustive but should serve to give a general overview.

(author?) [12]

COMMENT: [*Things to reference:*]

- Geographic Information Analysis book, check for further references
- Density measures
- Probability distributions

- Existing work on things like variance and mean
- Worboys – Geographic Information Systems: A computing perspective for some data structures
- see if there is any literature on describing dot patterns within classification or optimisation

2.3 Change

COMMENT: [*List types of dot pattern change, note work on convex hull updating using data structures*]

Chapter 3

Footprints and Dot Patterns

3.1 Footprint Descriptors

3.2 Dot Pattern Descriptors

3.3 Parameterisation

Chapter 4

Change Identifiers

Change identifiers are computationally simple operations that indicate some form of change over any phenomena that may be interpreted as shifting dot patterns. Originally they were conceived to indicate an appropriate moment in time at which the footprint should be updated. The footprint is a shape used as a representation of the dot pattern, the majority of footprint algorithms run in at least $O(n \log n)$ time. In a situation where the patterns arrive as a 'live' stream of data it is likely that the representation would fall behind the actual pattern as the length of time taken to run the algorithm is greater than the time taken to receive new data. The change identfiers run at each time step are computationally negligible and can be used to indicate whether an update is necessary or the current footprint is sufficient as a represenation.

In running some preliminary experiments using a convex hull algorithm the change identifiers were successful at greatly reducing the amount of updates required. However the speed at which the algorithm ran was so fast, even for very large dot patterns, that it seemed unlikely there was a possible situation where the patterns could arrive faster than the run time. While further tests with computationally more expensive algorithms needed to be run it seemed appropriate to examine other potential uses of the change identfiers.

One of the fields in which coordinate data arrives in real time is in emergency situations (wild fires, chemical spills, etc). In these situations a fast and appropriate response is always the goal. If the change identifiers could not be used to meaningfully speed up the time taken represent the data could they be used to provide extra useful information? Often the requirement of those managing the system is not to just visualise the affected areas but to use this to make decisions about how the area is changing e.g., is it expanding?, translating?, transforming?. The change identifiers can be used to provide this information far faster than analysing the footprint because use generalisations (such as the bounding box) or are only concerned with the patterns themselves to compare.

Using the χ-hull algorithm (Duckham et al. 2008) as an example of more complex algorithm it was found that the time taken to run could be in to the seconds[1] on dot patterns of 250 dots. While a run time of secondsdoes not appear to be particluarly slow algorithm, if the data is arriving faster than the algorithm runs then the representation gets further and further out of step as can be seen in 4.

Before discussing the creation of the change identifiers there is one more important possible use. The χ-hull algorithm and, indeed, nearly all other non-convex hull algorithms reqiure an external parameter. There is rarely (not yet found an example of)

[1]The χ-hull algorithm is actually very fast, however it requires a delaunay triangulation and even using the divide and conquer method proposed by Guibas and Stolfi this is what takes the majority of the processing time

Time Step	Representation	Lag
1	-	-
2	1	1
3	1	2
4	2	2
5	2	3
6	3	3

Table 4.1: The patterns arrive once a second and the representation takes 2 seconds to run

a systematic method to choose an appropriate parameter, futhermore once chosen, as the dot pattern changes, it's unlikely that the parameter will remain appropriate. It is more than likely that the change identifiers will give either information as to when the parameter should be updated or even hints as to what the parameter should be (this depends on how close the parameter is to a geometric facet inherent within the dot pattern e.g., side length in χ-hull).

When considering possible change identifers it is important to be able to classify what form of change they measure. Possible spatial change types the patterns can undergo are[2]:

- Change in dimension (apparent dimension:- crowd funneling into a queue...)
- Change in connectivity
- *Change in location*
- Change in orientation
- Change in size
- **Change in shape**

Note that change in location has been emphasised, while a change identifier that tracks the centroid is certainly computationally efficient it occupies a special subset of the identifiers. This identifier allows us to update the footprint without having to recompute it, translating it along the same vector that the centroid has moved takes little processing time, and as a result can simply be done at each step. This allows us to remove location as a factor from any of the other identifiers. As opposed to the dots existing in an absolute coordinate position they can be relative to the centroid. This simplification allows the information that the other identifiers return to be more specific. For example; an identifier that measures the symmetric area difference of the bounding box of the previous change causing dot pattern and the current would be affected by the change in location if the positioning is absolute, however when the positioning is relative it combines measurements of size and dimension.

Change in shape has been bolded because it is the most difficult to accurately capture, this is obvious on examination as to measure changes in shape most methods require that the shape is actually created, which would mean recomputing the footprint and thereby reducing the point in actually using the change identifiers. However, it can be used to measure the accuracy of the change identifiers. Imagine a stream of dot patterns, running concurrently with itself. One version of the stream is updating each time step, call this ALL, and the other uses change identifiers to inform when it should update, CI. If we ignore lag we can compare the footprint from ALL_t (where t is the time step) with CI_t. Measuring the shape difference between these $\forall t$ and averaging it gives us an overall accuracy of the identifiers. Currently the shape difference used for these measurements has been the Hausdorff distance.

[2]Shape change types, taken from Galton, *Qualitative Spatial Change*, 2000

The actual implementation of change identifiers is done in such a way that they can be combined. The reasoning is that, while each is individually only capable of indicating change in one facet, collectively they can capture complex changes in shape. Originally they were threaded such that a multi-core machine could parallel process them, however it was found that the time taken to create a new thread was often longer than the identifiers run time.

Chapter 5

Framework

Chapter 6

Results

Stuff written

Chapter 7

Minimisation and Optimisation

Stuff written

Chapter 8

Optimal Parameter Selection

Chapter 9

Shape Information

Chapter 10

Conclusions

Stuff written

Bibliography

[1] H. Alani, C. B. Jones, and D. Tudhope. Voronoi-based region approximation for geographical information retrieval with gazetteers. *International Journal of Geographical Information Science*, 15(4):287–306, 2001.

[2] Avi Arampatzis, Marc van Kreveld, Iris Reinacher, Christopher B. Jones, Subodh Vaid, Paul Clough, Hideo Joho, and Mark Sanderson. Web-based delineation of imprecise regions. In *Computers, Environment and Urban Systems*, volume 30, pages 436–459. Elsevier, 2006.

[3] A. Ray Chaudhuri, B. B. Chaudhuri, and S. K. Parui. A novel approach to computation of the shape of a dot pattern and extraction of its perceptual border. In *Computer Vision and Image Understanding*, volume 68, pages 257–275. Academic Press, 1997.

[4] Matt Duckham, Lars Kulik, Mike Worboys, and Antony Galton. Efficient generation of simple polygons for characterizing the shape of a set of points in the plane. In *Pattern Recognition*, volume 41, pages 3224–3236. Elsevier, 2008.

[5] Max Dupenois and Antony Galton. Assigning footprints to dot sets: An analytical survey. In K. S. Hornsby, C. Claramunt, M. Denis, and G. Ligozat, editors, *Spatial Information Theory: Proceedings of the 9th International Conference COSIT 2009*, pages 227–244, Berlin, 2009. Springer.

[6] Herbert Edelsbrunner, David G. Kirkpatrick, and Raimund Seidel. On the shape of a set of points in the plane. In *Computer Vision and Image Understanding*, volume IT-29, pages 551–559. IEEE, 1983.

[7] Antony Galton. Pareto-optimality of cognitively preferred polygonal hulls for dot patterns. In *Spatial Cognition*, 2008.

[8] Antony Galton and Matt Duckham. What is the region occupied by a set of points? In *GIScience*, 2006.

[9] Gautam Garai and B. B. Chaudhuri. A split and merge procedure for polygonal border detection of dot pattern. In *Image and Vision Computing*, volume 17, pages 75–82. Elsevier, 1999.

[10] R. A. Jarvis. On the identification of the convex hull of a finite set of points in the plane. In *Information Processing Letters*, volume 2, pages 18–21. North-Holland Publishing Company, 1973.

[11] Adriano Moreira and Maribel Yasmina Santos. Concave hull: A k-nearest neighbours approach for the computation of the region occupied by a set of points. In *International Conference on Computer Graphics Theory and Applications GRAPP*, 2007.

[12] David O'Sullivan and David J. Unwin. *Point Pattern Analysis*. Wiley, November 2002.