# Chapter 1

# 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 $\chi$-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

[1]The $\chi$-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

algorithm runs then the representation gets further and further out of step as can be seen in 1.

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

Table 1.1: The patterns arrive once a second and the representation takes 2 seconds to run

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

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

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.

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.