Report on Work Done 16/03/2011 – 06/05/2011

Maximillian Dupenois Antony Galton

This report concerns the work completed since the submission of a paper to COSIT 2011. It comprises three parts:

1. Discussion of the descriptors that are part of the COSIT 2011 paper. Focusing on the removal of density as a base level descriptor and the addition of orientation. Density will be discussed as a descriptor that encompasses 2 base descriptor types.

2. Parameter control; there is possibly a paper to be made on this topic. Looks at maintaining the ration between area and perimeter^2. Probably requires a similar specification file to the one used by change identifier sets for the purposes of choosing algorithm, starting parameter, accepted range for the parameter/area and how the change in parameter will affect the ratio.

3. Framework layout :- Pointing out that the thesis is presenting a framework in which to use change identifiers, not just the identifiers themselves.

1 Dot pattern descriptors

The dot pattern descriptors given in the submitted version of the COSIT 2011 paper include density. Density is an descriptor that is a combination of extent and cardinality measures and as such is not a true base descriptor. To make things clear it is a good idea to fix the nomenclature to concrete definitions:

- Descriptor: A facet of information about the dot pattern which can be measured.

- Base Descriptor: A descriptor which is measured from information given by the pattern and does not include any other descriptor in its measurement.

- Combined Descriptor: A descriptor which uses information from other descriptors to provide its measurement e.g., density uses cardinality and extent.

- Intrinsic Descriptor Method: A method for returning a measurement value for a descriptor that uses only the dots of the pattern as its information base.

- Surrogate Descriptor Method: A method for returning a measurement value for a descriptor that uses a surrogate footprint (commonly the isothetic bounding box) for the dot pattern as its information base.

- Change Identifier: A method for measuring change in a descriptor type using either a surrogate or intrinsic descriptor method.

As in the paper the three base descriptors of extent, cardinality and position. We can extend this list to include orientation, which measures the alignment of the pattern to the xy-axis. It is cognitively easy to envision measures for this value using a surrogate method, for example the gradient of the longest line of the aligned bounding box. However this leads to two problems. Firstly if the aligned box is square there is no longest side and it may be more appropriate to say it has a null orientation being essentialy rotation invariant. Secondly finding the aligned minimum bounding box in linear time requires the convex hull to be found first. In fact any aligned surrogate will probably require the convex hull to be found as all extremal dots will likely need to be found. This may well take too long for our time constraints.

An intrinsic method can be found by examining the pattern as if it were a graph and using a line of best fit to provide a gradient. There are many ways in which to do this but as we are most interested in how the orientation changes (as opposed not providing the most accurate measure) the ordinary least squares method seems most appropriate. [SHOW HOW WORKS AND RESULTS OBTAINED FROM ITS USE].

2 Parameter Control

Perimeter and area provide a useful, if crude, way of assessing the suitability of a given footprint. This follows from the work performed in [REF-GALTON] in which it was shown that any cognitively appropriate footprint for a dot pattern will tend to fall on the pareto front of an area perimeter graph. Specifically the 'best' footprints fall on the 'knee' of the graph where a balance between area and perimeter is found. If we assume that it is an acceptable task to ask of a user we can begin any run over a stream of dot patterns with the knowledge of what ratio of area to perimeter^2 (squared to keep the units the same) is required for this context. Given a starting parameter we should be able to adjust this value based on attempting to achieve or maintain the given ratio. To perform this kind of adjustment we will need to know how the parameter affects the ratio; fortunately for most algorithms this information is known [EXAMPLES]. If such information is not available, as in [EXAMPLE], we can instead suggest to the user that the given parameter is no longer suitable and request that it be changed. There are other possible measurements we can make on the footprint as judgements on its suitability (e.g., sinuosity) however any further factors to attempt to control make knowing in which direction to alter the parameter more complicated. As well as achieving a more accurate representation of the dot pattern this kind of change tracking allows us give more information about the fashion in which the dot pattern is changing. If the area is increasing without large increases in parameter the the pattern is becoming more spherical. If, however, the perimeter is increasing and the area is not keeping pace (or decreasing) then the pattern is becoming less uniform; possibly indicative of the pattern undergoing seperation.

3 Framework

The identifiers are an interesting intellectual exercise but require a rigid structure in which to perform to be useful. The identifiers are small measuring devices for often very specific types of change. They do not communicate with each other as the can concievably run in parallel[1]. They do, however, often require the same pre-computed structures or values for a specific dot pattern; e.g., the centroid or minimum isothetic bounding box. In fact for all but the first time step the pattern previous to the current will already have had values computed for most if not all the descriptors; as such the identifier does not want to compute these all again for the comparison to the current values. To keep track of these re-usable measurements and structures a container is required and this container will need to be specific to each dot pattern. To maintain modularity, and thereby maintain clarity of design, this is seperated into a structure of its own. In considering the structure holding these values we must devote some thought to how we approach storing the patterns themselves.

When first testing the concept of change identifiers some assumptions needed to be made about the manner in which the patterns would arrive to the framework. Would they be a set of identified vectors indicating which dot had moved by what magnitude and direction? Would the dots even have identities distinguishing them from pattern to pattern? How would the loss of old or addition of new dots be described? With so many possibilities it was decided to use the form with the minimal amount of information to provide the greatest range of possible applications. The dots are expected to arrive as a set of cartesian co-ordinates showing position at the timestep the pattern represents. Within the set there is no indicator of identity, rendering trailing a specific dot across the timesteps impractical.

Having decided upon the fashion in which the dots arrive does not directly tell us how best to store them. As there is no explicit identity the pattern storage structure will have to be re-created at each timestep so we can ignore update times for the structure. This leaves searching and insert as our time constraints. Most data structures are created with low insert times, particularly if the co-ordinates have been pre-sorted. Searching, however, is an issue as all structures are optimised for one (or few) search types and it can not be guaranteed that an descriptor method (or footprint algorithm) will not require a fast search for a coordinate that the current structure does not provide. If the insert time is guaranteeably small we can use more than one type of structure; covering for multiple search types. Unfortunately even with this precaution it may not be possible to fully satisfy all search criteria in an optimal time.

The identifiers also require a skeleton under which to operate to combine their returned values and check the sum against a threshold. This skeleton is called the change identifier set.

[1] Although for most tests it was found starting the individual threads would take longer than running the identifiers.