

# Knowledge-based recognition of utility map sub-diagrams

Simon J. Hickinbotham and Anthony G. Cohn  
University of Leeds  
Leeds LS2 9JT, UK  
hickin@comp.leeds.ac.uk

## Abstract

An integrated map of all utility services in a locale would facilitate better management of the road infrastructure and the utilities themselves. To meet this goal, there exists a need to integrate raster scans of paper maps into GIS by capturing the semantic relationships between the objects in the drawings. In this context, commercially available vectorisation algorithms do not produce a sufficiently rich object representation. We present a structural object recognition system that successfully isolates sectional sub-diagrams in maps of underground utilities. This is built upon a vectorisation system based on a constrained Delaunay triangulation of pen strokes.

## 1 Introduction

For most of the twentieth century, the majority of spatial records for the buried assets of the UK utilities were stored and maintained on manually drawn maps. These maps combine cartographic and engineering drawing styles. In the UK, it is not unusual for utility map drawings to have been fully converted to a vector-based GIS, at great cost to the maintaining body. An economical approach is to scale and rotate the original drawings to comply with a GIS, which inevitably introduces some image distortion. These latter systems are not amenable to direct integration. Whilst the scans in these systems are adequate for human interpretation, they may not be of sufficient clarity and detail to allow conversion to vector form in a GIS. This paper reports work that attempts to identify sub-diagrams in scans of hand-drawn cartographic maps of public utilities. The work is an experimental vehicle for applying semantic grammars to the interpretation of utility map scans. The work forms part of VISTA [1, 2], a government sponsored project to coordinate and integrate the management of buried asset data for the improved management of street works.

Subdiagrams for a small area of a utility map are shown in figure 1. In this example, trenches are represented as

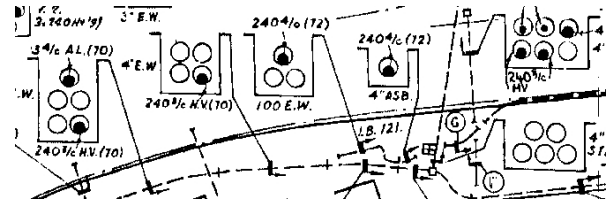


Figure 1. sub-diagrams in a typical utility map

dotted lines on the map. The layout of cables is shown via small sectional diagrams of the trench. The diagrams are linked to the physical location on the map by a lead line.

We seek a means of capturing the possible arrangements of cabling and ductwork in sectional diagrams so that they can be recognised reliably. Since the diagrams are drawn with respect to certain drawing conventions, it seems reasonable to expect that a context-free visual grammar would be suitable for capturing the layout. Our initial aim is to recognise areas of the image that are consistent with the grammar, so that these regions can be isolated and stored as small images in a GIS for buried asset management. It is clear that these sectional sub-diagrams are context-free – they are independent of information anywhere else in the image. Our approach is to exploit this fact to begin to understand the content of the drawing. Since we are imposing a grammar on the map, we require a representation of the scan which is economic, yet complete enough to allow multiple interpretations. To this end, we develop a structural representation of binary images using a Delaunay triangulation of the pen stroke boundaries.

### 1.1 Related Work

There is a wide range of research that we can draw upon when searching for ways to find visual objects in utility maps. The literature for extracting line primitives from raster maps is fairly mature, but the task of assigning lines and junctions into more richly detailed semantic units is the subject of on-going research. A useful review can be found

in [8], along with a set of recommendations for implementing a vectorisation system. The strategy is to combine feature detectors at all processing levels. Thus, evidence for a particular object or feature is gathered from a variety of sensors. For example, the contours (boundaries) of the drawn lines can be used for vectorisation as well as the medial axis (skeleton).

A knowledge-based approach to interpreting utility maps attempts to encapsulate drawing conventions, symbol sets, and the spatial relationships between entities. Den Hartog et al were among the first to apply these ideas to utility map data [3]. Here, a semantic network was defined that controlled a scheme for searching object primitives. An interpretation sequence and geometric object description were devised and applied to utility maps from the Netherlands. The drawing conventions for these maps was such that building outlines, pipe lengths and dimensioning symbology (arrows etc.) could all be discriminated on the basis of line width. Ogier et al. [4] develop a grammar using a set of “builders” which progressively construct information regarding the images they are modelling. Their bottom-up approach is somewhat dedicated to interpreting cadastral map objects, and the authors report difficulties in extending the approach to other map types.

Although there are many strands to the feature detection process, there is a general need for some control of the different voting mechanisms. Most controllers treat feature detectors as a bottom-up hierarchy, and attempt to channel the interactions between feature detectors across the hierarchy. An interesting alternative is the cyclical approach favoured by Ramel and Vincent [5]. Here, a controller consults feature detectors as part of a cyclical process, regardless of the level of abstraction. The advantage is that all levels are considered in the resolution and understanding of the image, although the process appears somewhat anarchic.

The novelty of the work presented here lies in the use of a Constrained Delaunay Triangulation (CDT) as the input to a semantic visual grammar. The advantage of using a CDT in this context is that the contours of the pen strokes are held in a compact form, from which skeleton, boundary and junction data can be computed at different levels of interpretation.

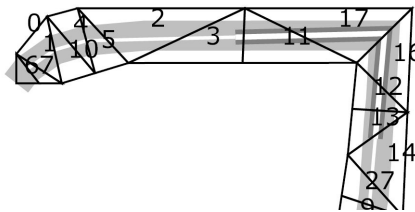
The remainder of this paper is organised as follows. Section 2 details the conversion of raster data into classes of scanned pen strokes, the semantic rules we have developed to detect the sub-diagrams in utility maps and the parsing algorithm. Experimental results are presented in section 3. Conclusions and suggestions for further work are presented in section 4.

## 2 Methodology

### 2.1 Low-level raster to vector conversion

The contours of the lines on these raster maps were obtained via the “Potrace” algorithm [6]. A CDT of these contours was generated using Shewchuk’s “Triangle” algorithm [7]. Both of these algorithms are widely used, and have freely available software implementations. We are not currently aware of their use in the interpretation of technical documents. The CDT gives us an economical representation of the boundary data at the expense of some extra handling requirements for non-Delaunay triangles.

With the CDT to hand, we commence the interpretation stage by finding paths through the triangulation of the pen strokes. Following Zou [10], we count the neighbours of the triangle to determine if the triangle forms a termination (1 neighbour), path (2 neighbour) or junction (3 neighbour) triangle. We refer to these triangle types as  $t$ ,  $p$  and  $j$  triangles respectively in the remainder of this document. We define a *line path* as any consecutive string of  $p$ -type triangles, terminated at both ends by either  $t$ - or  $j$ -triangles. Direct connections between  $t$  and or  $j$  triangles are not considered in our current framework, although there is scope for addition of these sections should they be deemed necessary for a particular grammar. It is worth commenting on the novelty of using this method in the application domain of utility maps. The benefits of the approach are that medial axes of the straight lines can be easily deduced, while the contour and triangulation data can be used to resolve the perception of junctions and higher-level objects in a variety of ways.



**Figure 2. Detail of a boundary triangulation**

A sample of a triangulation of a corner is shown in figure 2. A hierarchy of line features can be imposed upon this triangulation, as is shown in the figure. The triangles are shown as thin black lines. We can form a line path (shown in light grey) from these triangles by bisecting each side of the triangle that has a foreground neighbour. The path can be divided into straight line sections (shown as thin white lines) by fitting a line through the triangle centres. These straight sections can be used to detect corners in the line path, indicated by the dark grey feature in the figure. These

Label	Rule
CIRCLE	Start triangle must be the same as the end triangle; Centre is mean of triangle centres in the path; triangle centres must be less than 1 line width away from the radius
STRAIGHT	Any sequence of three or more triangles whose centres are no more than 1/2 a line width from a regression line through the set of triangle centres
L	Along any line, two STRAIGHT lines whose intersection point is inside a triangle between the two straights and whose interceding angle is approximately perpendicular
NL	Any L with branches pointing to within 10° of the north and east of the junction point
EL	Any L with branches pointing to within 10° of the east and south of the junction point
SL	Any L with branches pointing to within 10° of the south and west of the junction point
WL	Any L with branches pointing to within 10° of the west and north of the junction point

**Table 1. A dictionary for primitive line classes**

steps show the advantage of using a CDT to hold the boundary data by allowing us to reinterpret the raw data as grammatical evidence accumulates.

## 2.2 A dictionary of graphical primitives

Our semantic representation begins with a dictionary which applies a line classification to different kinds of pen strokes. This basic classification is based on the arrangement of the triangles in each line path. We require a dictionary that isolates sectional figures consisting of straight lines, perpendicular corners and circles. The rules for line classification are shown in table 1. Note that some rules are dependent upon others. For example, the SL entity is an L that has a particular orientation and in turn the L entity depends upon the relative positioning of two STRAIGHT entities. It could be argued that classes which depend upon other classes are part of the grammar of the system, not the dictionary. Strictly speaking, this argument is true. But note that the “dependent” classes occur *within* individual line path objects, and since the line path is our basic processing entity, it makes sense to define the boundary between dictionary and grammatical elements at this level.

Having defined our basic dictionary, we can now define the connections between the dictionary elements in our grammar. In language, words are presented sequentially (usually left to right). Thus the only spatial relationship is adjacency, which is only implicitly represented in grammatical rules, by juxtaposing grammatical elements. Word adjacency captures “locality”, i.e. grammars are built out of “locally neighbouring” elements; this implicitly restricts the search space, by only looking for grammatical combinations of adjacent words. We need a corresponding notion for our spatial grammars (since the grammatical elements we wish to recognize are all relatively spatially compact). We use a bounding box around each dictionary element to manage the computational burden. (Binary) spatial relations are then only computed between elements whose

bounding boxes overlap, or which both overlap a third common element.

We then make use of explicit spatial operators to capture the more complex spatial configurations possible in two dimensions. We have defined a small set of spatial operators which capture the positional knowledge needed to specify the semantic entity being considered here (others may be needed if we wish to extend the grammar to parse other object types). These are shown in table 2. The right hand side of each rule has explicit spatial operators (<, ≻, or ●●) between adjacent grammatical constituents instead of the implicit juxtaposition of standard textual grammars. These spatial operators are defined with respect to the bounding boxes of the adjacent spatial entities. For example, we check for connection (≻) between two elements by testing for the presence of foreground colour along a line connecting the corners. If more than a threshold (e.g. 95%) of this connecting line is covered by the triangulation of the foreground colour, we can accept the connection and add the new element to the chart. The bounding box of the new element spans the bounding boxes of the constituent parts.

## 2.3 Components of a spatial grammar

A grammar for sectional diagrams is presented in table 3. It specifies that a trench should contain at least one row of at least one pipe. Whilst some diagrams contain enti-

Operator	Meaning
“x < y”	x spatially contains y
“x ≻ y”	x connected to y
“x ●● y”	x is colinear with y

**Table 2. Relationship operators for visual grammars**

Left hand side	Right hand side
TRENCH	CUTLINE < PIPES
CUTLINE	SL $\bowtie$ NL $\bowtie$ WL $\bowtie$ EL
PIPES	PIPES $\bullet\bullet$ PIPE
PIPES	PIPE
PIPE	CIRCLE

**Table 3. A grammar for sections of utility trenches**

ties that do not contain pipe-shapes, this simple grammar is sufficient for an evaluation of the technique.

## 2.4 Parsing

We are trying to develop an algorithm for parsing line object primitives so that we can detect the presence of higher-level grammatical structures. We have already addressed the problem of finding neighbouring elements. We now require an efficient algorithm for parsing the dictionary elements to determine whether they conform to a predefined grammar.

Although there are many alternatives available, we have chosen to adapt the Active Chart Parsing (ACP) algorithm. This is a well known algorithm for parsing language in an efficient manner – individual combinations of chart elements are only considered once, and held in a chart (or graph) of active and passive edges that span any number of dictionary elements. The idea is that at the end of processing, the remaining active vertices plot a route through the production of the grammar that conforms to the rules of the grammar, but all alternatives are held as “passive” elements in the chart.

We can quickly eliminate potential pairings of elements by testing whether their bounding boxes overlap. Once an overlapping pair of elements is detected, we can see whether such a pair appears in the right-hand-side (rhs) of an element of the grammar. If an overlapping pair of elements is found in the rhs of an element of a grammar, we can further test for the validity of the pairing using the connectivity rules described in the grammar.

With these components to hand, we can use ACP to interrogate a line drawing for elements specified in a grammar. We refer the reader to [9] for a more detailed account of how ACP works, but the steps can be summarized as follows:

- Initialize the chart by detecting all dictionary elements (*edges*) in the drawing using the above techniques.
- Repeat until there are no pending edges remaining to be processed:
  - Assign any pending edge as the new edge

- Try to combine the new edge with all overlapping elements
- If a combination meets the rules in the grammar, then create a new edge from the combination and add it to the chart

## 3 Experimental Evaluation

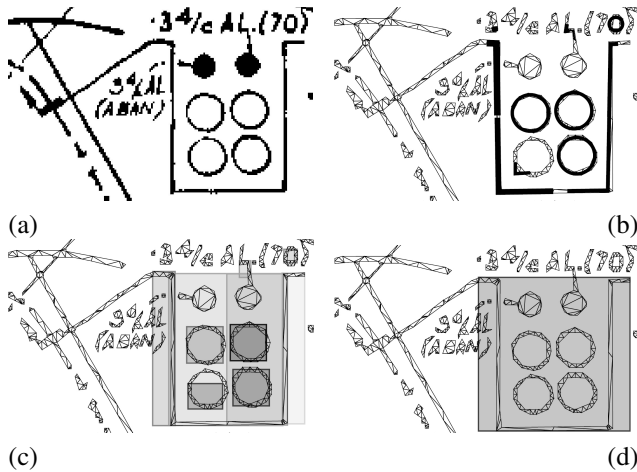
Original binary scans of hand-drawn maps were obtained from EDF Energy. These scans were designed to be used as a single layer in a GIS, and had been stretched to fit their geographic positions. The resolution of the scans was set to be amenable to human interpretation and no consideration was made during scanning for the possibility of automatic interpretation. The net result of these processing steps was that some of the pen strokes were fragmented in the bitmaps used in these experiments. The typical width of the pen-strokes at this resolution was two pixels. There were no obvious line classes with different line widths as in [3]. It should be noted that the subdiagrams we wish to identify all have approximately constant orientation and the data sets available to us were all of a fixed scale. Whilst our operators are all scale and orientation independent, we have not yet investigated the performance of the approach in detecting objects at scales and orientations that were not present in the data.

We examined 10 images, each one consisting of 15 tiles. We manually examined these images and found a total of 1,128 trench diagrams. Of these, 748 (66%) contained features that conformed to the grammar specified in section 2.3. At least three separate drawing styles could be identified in the data set.

We implemented the algorithms described above in the C programming language and used the resulting program to analyse the data set. The mean time for processing a 5000x3000 pixel image on a 3.3GHz PC under the Windows XP operating system was 240 seconds. Of this time, finding the boundary points took 1.6%, triangulation took 43.9%, detection of dictionary elements 19.3% and parsing 13.1%. File i/o took 22.0% of the time, since large scalable vector graphics files were output.

The process of recognition is illustrated in figure 3. The program detected a total of 20,389 dictionary elements from table 1 in the data set. 594 trench diagrams were detected, not including 18 false positives. Precision and recall were 97.1% and 79.4% respectively. There were three factors contributing to failures to detect trenches that conformed to the grammar we defined. The most significant of these were cases where circles were either scanned incompletely, or were not drawn to form a continuous loop. The second problem was that where lead lines crossing the base of the trench diagram caused a failure to detect the connectivity of elements in the “cutline” entity. The third case, which was

rarer, was when the draughting style was such that circle entities tended to overlap in the scans.



**Figure 3. Stages in identifying a sectional diagram. (a) Original scan. (b) feature primitives (c) Bounding boxes for primitives. (d) Bounding box for parsed trench diagram.**

## 4 Conclusion and Further Work

Using the approach defined above, we can successfully extract sub-diagrams from complex utility maps. The results presented above show that by holding the line stroke data in a Delaunay triangulation, we can leave decisions about how to interpret line data until a higher level set of rules are brought to bear on the process of understanding line drawings.

We are still in the early stages of this research project, and intend to pursue a number of avenues of research given the results we present above. We intend to begin by enriching the grammar and connectivity rules described in this paper to reduce the incidence of false negatives and false positives. We also intend to evaluate alternatives to the CDT-based vectorisation strategy adopted in this paper. In addition, now that we can isolate sub-diagrams in utility maps, our intention is to further investigate the understanding of utility maps using the approach described above. We plan to recognise the drawing conventions that link the sub-diagrams to the underground cabling that they refer to and then follow the lines that represent the cables. This will allow us to follow the cable-line elements and recover their geographic position.

Although we have a sufficiently rich low-level vectorisation to allow a grammar to be used for recognition, there is scope for improvement of our techniques in this area. In particular, we seek a richer classification of curves, arcs and

crossings, along with a better means of holding alternative hypotheses about the likely status of drawing elements at this level. We also plan to explore the idea of extending the grammar to a lower level of representation that is only one level above the triangulation itself - line types like arcs, straights and blobs could form the terminal symbol set of this lower level representation. Although the computational overheads are likely to be higher, there is a strong possibility that the grammar could impose more accurate decisions about what the meaning of the lines is likely to be.

If we wish to extend our work to more general recognition problems, we are faced with the problem of encoding the grammar for each semantic entity by hand. We intend to explore the idea of unsupervised learning of grammars given a set of terminal entities and a set of connecting rules.

## 5 Acknowledgements

The authors thank EDF energy for the provision of the data sets for this work, and Brandon Bennett for comments and suggestions. The work was sponsored by the DTi, UK.

## References

- [1] <http://www.vistadtiproject.org>, 2006.
- [2] A. R. Beck, G. Fu, A. G. Cohn, B. Bennett, and J. G. Stell. A framework for utility data integration in the uk. In *Proceedings of the 26th International Symposium of Urban Data Management*, London, UK, 2007 (in press). Taylor and Francis.
- [3] J. D. Hartog, T. T. Kate, and J. Gerbrands. Knowledge-based interpretation of utility maps. *Computer Vision and Image Understanding*, 63(1):105–117, 1996.
- [4] J.-M. Ogier, R. Mullot, J. Labiche, and Y. Lecourtier. Semantic coherency: the basis of an image interpretation device-application to the cadastral map interpretation. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 30(2):322–338, 2000.
- [5] J.-Y. Ramel and N. Vincent. Strategies for line drawing understanding. In *GREC*, pages 13–20, 2003.
- [6] P. Selinger. Potrace: a polygon-based tracing algorithm. In <http://potrace.sourceforge.net/>, 2003.
- [7] J. R. Shewchuk. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In M. C. Lin and D. Manocha, editors, *Applied Computational Geometry: Towards Geometric Engineering*, volume 1148 of *LNCS*, pages 203–222. Springer-Verlag, May 1996.
- [8] K. Tombre, C. Ah-Soon, P. Dosch, G. Masini, and S. Tabbone. Stable and robust vectorization: How to make the right choices. *Lecture Notes in Computer Science*, 1941:3–18, 2000.
- [9] T. Winograd. *Language as a Cognitive Process*, volume 1. Addison-Wesley, 1983.
- [10] J. J. Zou and H. Yan. Skeletonization of ribbon-like shapes based on regularity and singularity analyses. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 31(3):401–407, 2001.