REPRESENTATION OF SKETCH DATA FOR LOCALISATION IN LARGE DATA SETS

M. Kopczynski, M. Sester

Institute of Cartography and Geoinformatics, University of Hannover, Appelstraße 9a, 30167 Hannover, Germany –Matthias.Kopczynski@ikg.uni-hannover.de

KEY WORDS: Analysis, Interpretation, Matching, Recognition, Query, Data Structures, Pattern, Reference Data

ABSTRACT:

Sketches are often used by humans to quickly give information about places or illustrate how to find a way from A to B. Thus, sketches are an interesting technique of generating queries with the goal of finding unknown places matching the sketched constraints. This article proposes a graph based structure for the representation of the knowledge in the sketch and shows how this structure supports the matching process. A set of relation and object classes is developed to represent the sketch in an appropriate way, which implies preserving the topology in the sketch but neglects most of the exact geometric information. Some examples are shown and analyzed.

1. INTRODUCTION

Not a long time ago internet search engines were simple programs, dealing with masses of text input, regardless of the contents in the text. The search engine simply looked for a word, typed in by the user. But people are always interested in the contents and the search engine companies were forced to produce better results. One way to achieve this is to rank the results of the query by guessing what the user is interested in and what the page contents is about.

A lot of the queries turn out to be either implicitly or explicitly related to space. Websites of shops are interesting when the shop is near the own location or tourist attractions should be on the same island where someone spends his holidays. Some search engine companies realized this and offered to restrict the query to an area, given by a country or by zip codes (e.g. Mirago: www.mirago.com and Google: www.google.com/dirhp or local.google.com). This can improve query results but is only a small part of what is possible.

Much more than zip codes can be used to specify spatial queries. Examples are place names with a radius, a region name, any set theoretical combination of regions and so on. New ranking technologies are capable of presenting the most suitable results to the user. An ontology of places and regions can help while dealing with ambiguous names for the queries and can be extended to special domains, like tourism, to support combination with other context sensitive information.

The SPIRIT project (Jones, 2002) is doing research on this techniques to design a powerful spatially aware search engine that really can answer spatially related queries in the internet.

Conventional search engines are using simple text interfaces but there are better ways to create a query. An obvious approach for spatial queries is the use of maps where one can choose the region of interest, which is similar to the zip code approach but is easier to use because who knows the zip code of a holiday resort? This way of choosing a region is very simple but is restricted to single regions near a known place.

A way to abandon this restrictions is to draw a sketch. If someone asks for the way, an explanation only using words can get very complicate and time consuming. Much more easy is to get some piece of paper and draw some lines representing important objects in space. This principle can be transferred to the search engine interface. Simply draw a sketch of the region you want to know something about. This is the way we are thinking about space and the advantage is that questions of an abstract level can be formulated.

A sketching input tool is developed as part of the graphical user interface of the SPIRIT search engine. This article shows how sketches produced with it can answer the question for a sketched area.

The paper is organized as follows: after an introduction into the nature of sketches and a prototype of a sketching tool, the representation of a sketch in terms of a graph is introduced, as well as matching techniques as means to match a sketch and a given representation. In section 4 the concepts needed for describing sketches are presented. The use of this structure is firstly applied to matching two road data sets. Finally, there is a conclusion and an outlook on further work.

2. SKETCHES

2.1 The structure of a sketch

At first an overview about the anatomy of a sketch is given, because the term “sketch” can be used with several different meanings. Figure 1 is showing an example sketch.

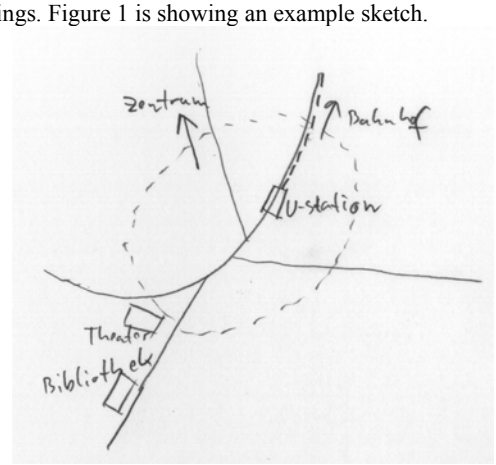

Figure 1: Sketch of a location in Hannover.

Mainly a sketch is produced very quickly and uses only as few elements as possible to communicate the intended information. A line may become a representative of a road, and a rectangle

may represent a building while another rectangle is in the place of a railway station.

Conclusion 1: The meaning of geometric entities strongly depends on the context in the sketch.

The lines of the sketch are aggregated to complex graphical representations of an object, which are usually known to most people of the same cultural background. If the representation is too generic, annotations are used to fix the meaning. Annotations can also be used to show that an instance of a specific object is meant and not an arbitrary individual of a class. For example one can annotate "House of the Jones family" where the annotated object is a specific "of the Jones family" instance of the class "House".
Using specific instances makes the sketch very restrictive in its meaning and is useful for navigation from point A to point B where A and B are known places. On the other hand the use of generic classes leads to unspecific parts of the sketch where only the abstract information is of importance. If the generic type "road" is used in the neighbourhood of the generic type "church", it describes every situation satisfying this constraint.

Conclusion 2: Drawn Objects can be generic classes or specific instances of generic classes.

As sketches are drawn very quickly, they tend to be an abstract and incomplete representation of the real world. We always think and speak in terms that are an aggregation and generalisation of what we see, touch, feel in our environment and that are making communication possible. The incompleteness is due to the way we navigate in our world. So a sketch is not a map that shows completely everything in a certain area but a subset of the maps contents that is needed for a special purpose.

Conclusion 3: A sketch is an abstract and incomplete representation of the world.

As mentioned before there are relations between maps and sketches in some way because they refer to the same objects in space. But there are also some clear differences. At first the map has a reference frame with exactly measured coordinates for the objects contained in the map. A consistent reference frame is missing in a sketch and following from this fact there are no measured coordinates for the objects in the sketch. In principle the position of the objects on the sketch are an arbitrary choice of the drawer.
We are able to match objects of a sketch to a certain situation because usually some of the objects are annotated with their names while other objects are assigned to a class (e.g. railroad tracks) and their topological relations to other objects are typical for the sketched area (Blaser, 2000).

2.2 Sketching tool

Usually pencil and paper are used to create a sketch. This way of drawing strokes on a medium is very easy and quick and is suitable for most purposes.
In the SPIRIT project the sketch is used as a alternative method for generating spatial queries. But here the medium is a virtual piece of paper with infinite boundaries and the drawing tool can be a mouse or a pen used on a graphics tablet or better a TabletPC. This has some disadvantages, because a computer and a satisfying input device are not necessary at hand when needed but on the other hand the virtual paper offers the possibility of using editing methods like deleting and moving or utilizing a predefined subset of symbols. Text can be typed in contrast to drawing it which supports the interpretation of the sketch.
The sketching tool is using a client server architecture (Figure 2) to obtain the sketch, process it and send the query to the search engine.

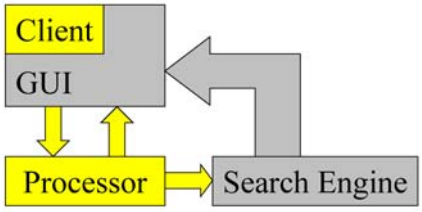

Figure 2: Sketching tool architecture.

The user is using a java applet as a client on the search engine homepage and draws the sketch on a virtual paper which is part of the graphical user interface presented by the client. Then it sends the sketch to a special processing server where the sketch is compared to a reference data set. The result goes back to the client where it is presented to the user. If he confirms the correctness of the interpretation, the query is transmitted to the search engine. Otherwise the sketch can be edited to achieve better interpretation results.
The GUI should be easy to use and allow straightforward editing of the sketch. This part of the sketching tool is still under development; figure 3 shows a preliminary version, which will change it's appearance in the future.

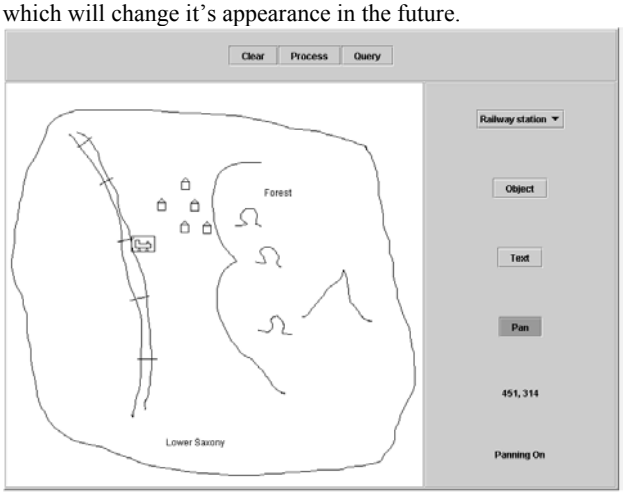

Figure 3: Sketching Tool.

3. INTERPRETING A SKETCH

3.1 Graph Representation

When the sketch is finished and transmitted to the processing server, the main problem is to interpret the sketched situation and tell where in space it represents reality. For this task at first a reference is needed which provides enough information for an identification. There is no use for information which is not sketched in common like height information or property borders. Very typical information is provided by the road network and points of interest. But even invisible features like administrative boundaries of well known administrative entities can be important.
As stated earlier in this paper the map data of the reference has some differences compared to the sketch data. So a comparable representation must be found which contains the crucial

information for the recognition task. The data can be seen as implicitly represented knowledge which has to be made explicit.

A well known representation for knowledge is the use of a graph structure. Here the special case of a conceptual graph applies (Sowa, 2001). The information is represented in the nodes and edges. The nodes are labelled with the concepts and the edges determine the relations between the concepts. In this representation the task of identifying a sketch is identical with the task of finding a sub graph isomorphism between the graph representation of the sketch data and the graph representation of the reference data.

A sub graph isomorphism (Cortadella, 2000; Lipschutz, 1976) is given if for every node in the pattern another node in the reference can be found with the same Type and when the edges of the pattern graph have corresponding edges in the reference graph.

When organizing the knowledge about the sketch and the reference data in graphs, a semantic component is needed. How this can be provided is shown here, following (Sowa, 2000)

At first the problem is dealing with objects of the real world where the objects are defined by the human understanding of the world. Objects in this sense can be roads, buildings, junctions, railroads, etc. The given terms are already a classification of the individual objects and can be called *classes*. The individual objects themselves are then called *instances* of a class. Each instance can be given a unique name which allows the identification of this particular instance. The classification should be unambiguous.

Each object is now considered to be one node of the graph and labelled with its class and the name of the object. This step doesn't produce connections between the nodes and the graph is totally disconnected.

The next step is dealing with *relations* between the given objects, making the graph unique if it describes a unique situation in reality. For the description of the graph structure it is not important which kind of relations are used. A relation can be modelled between an arbitrary number of objects. Usually two objects are involved but there can be more than two objects or only one object which relates to itself in some particular way. One example is the neighbourhood relation which is dealing with two objects. A set of useful relations is given later in this article. If we had only binary relations it would be easy to introduce them as labelled connections between the objects. But here it makes more sense to introduce new nodes into the graph which are treated the same way like the nodes introduced for the objects. They are labelled with a class and a name, where the name does not refer to something real but is needed to identify this relation in a unique way. The difference to ordinary objects is that edges are introduced into the graph which connect the relation node with all the object nodes which are subject of the relation. for an example see Figure 4.

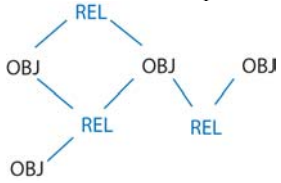

Figure 4: Conceptual Graph.

In some cases the border between objects and relations disappears and an object is also a relation. An example are road segments. A road segment is a real object but it always has the connection to a node of the road graph like a junction or a sharp bend which would require an explicit relation between the road segment and the road graph node. This can be simplified by treating the road segment like a relation and directly connect its node to the junction object node.

As a result we have one single connected graph of concepts for the reference data and one single connected graph for the sketch data. This representation is very basic and several different algorithms can use it for certain tasks.

3.2 Matching algorithms

With the graph representation it is possible to find an algorithm that identified the pattern graph inside the reference graph. There is a lot of literature about algorithms dealing with sub graph isomorphism, but some essential properties are shown here. An algorithm is detailed more in depth in this section. In (Bengoetxea, 2002) an overview of matching concepts is given.

Two cases have to be considered. Traditionally a sub graph isomorphism implicates the exact match between the pattern and a subset of the reference. Every node must satisfy all constraints including all connections between the nodes. This is simple to describe but it was proved that this task is NP-complete and the complexity of any algorithm is growing exponential with the number of nodes in the graph except for some special cases. One example for exact graph matching is the algorithm of (Ullmann, 1976).

The second case is dealing with error tolerant methods and mostly called inexact graph matching. For this case the NP-completeness has been proven too and its complexity is of a greater order than in the exact case.

Nonetheless or because of this fact a lot of approaches have been developed to solve this problem, like statistical analysis of the structure, evolutionary algorithms or exploitation of geometric distribution of graph parts.

The simplest approach is to apply an exhaustive tree search to the possible assignments of the graph nodes to each other. With the additional information of classes and names at the nodes the search space remains small enough to provide a quick search for identical sub graphs. This is the VF-Algorithm described by (Cordella, 2001) with some slight modifications.

The algorithms starts with a given node of the pattern, which should be a very unique one, in order to test the most promising matching pairs first. Then it tries the following for every node in the reference graph with the same class and name: Iterate through all nodes and edges of the pattern graph with a depth first search. For every node try to match the pattern node with the actual reference node or for every edge iterate through all open edges of the reference graph and make the next connected node the actual node. If the nodes match, save this assignment and push all unassigned edges in the reference connected to the node to the list of open edges. If the nodes don't match or no open edge can be found, backtrack to the last saved state. Do the backtracking too when no open edges or nodes are left. A solution is found when all nodes are assigned.

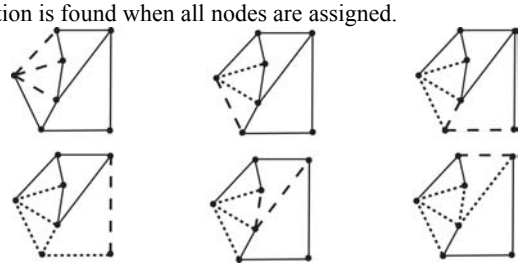

Figure 5: The region of assigned nodes (dotted) is growing, while potential assignments are evaluated (dashed).

In Figure 5 the assignment of edges is shown in green and the actually evaluated assignments are coloured red.
This is easy to implement when using recursion for each step of the depth first search. Each recursion represents one state and backtracking simply needs to recover all changes made in the actual recursion step (this is a small number) and leaving the actual recursion step.
This algorithm is useful if all possible solutions are needed. If the algorithm cannot find an exact matching solution it can at least find a largest common sub graph.
One drawback is the missing fault tolerance of this algorithm. It is theoretically possible to introduce fault tolerance by allowing the assignment of non matching nodes, pruning of pattern graph branches and insertion of dummy nodes into the pattern graph. Each of this actions will cause the accumulation of error points and backtracking is done if a maximum number of error points is reached. But unfortunately this increases the search space enormously because the strong restriction of the existing classes and names don't apply any more and some heuristic must be found to prevent this large growing of the search space.
Other possible algorithms can be build on top of this algorithm by using the largest common sub graph as a starting point.
Another approach is to exploit the assumption that a sketch contains a lot of geometric information although the graph constructed from the sketch consists of topological properties. Some of the nodes in the graph are direct representations for geometric objects and can be used to produce a unique projection of the graph. If some of the assignments between pattern and reference are known the sketch can be transformed into the reference frame of the reference graph, producing more tight constraints especially for the case of inexact matching. This approach is strongly related to the method of elastic graph matching (Lee, 2002). Vital for the success is a sufficient similarity between sketch geometry and reference geometry and up to now it is not clear if this premise is true.

4. A SET OF SKETCH CONCEPTS

The last section was dealing with the matching process of sub graphs which is common for a large range of applications. But now some specialization to the matching of sketches is needed. The basic premises were already outlined in the context of the definition of a sketch.

This section will propose a set of classes which has the ability to describe a road network with its important properties and leaves enough space for topological invariant operations on the data. Geometry is explicitly not part of the description and does only play a role for generating the description graph. However, geometry can be stored as identifiers of objects that were the source of the generated graph representation instances of the objects, which is useful for some heuristics. The principal geometric constraints applicable for sketches are the following:

- No uniform scale across the whole sketch, but locally similar scale.
- Locally, the relations are true, e.g. a small building next to a big one should reflect that this relation is also true in reality, although the actual sizes of the objects cannot be used.
- The directions drawn in the sketch, e.g. the directions of the roads at a junction should be approx. true.
- The relative orientation of the objects should be locally true.

Roads are used to explain the concepts first, at the end of this section an extension to other objects than roads is suggested.

4.1 Objects

The objects in the sketch need a representation in the description graph and form the skeleton for the relations between the objects. In a road network two main classes for objects can be identified if the road type is not modelled explicitly: roads and intersections. This choice is clear when the road network itself is seen as a graph where the edges are the roads and the nodes are the intersections.
Here at first the nodes are considered because as 0-dimensional elements they are the basis for the edges. The class is called **INTERSECTION** or **ISEC** here. They can be easily derived from the geometric data. The cardinality of a node is the number of outgoing edges. All the points which are part of more than two road segments (i.e. the cardinality is greater than 2) are intersections in the stronger sense, but points with only one incident road segment can also be considered as an intersection. They appear at the ends of roads. The cardinality of two should only sometimes lead to the introduction of an INTERSECTION because two road segments may meet at points where nothing special is visible and only is an artefact of the measurement process. Only if the direction of the road segments does exceed a certain limit an instance of this class must be modelled.
Roads are 1-dimensional elements of the road graph and are always links between two intersections. Here the model class is called **ROAD**. Elements of this class are always linked with two terminating ISEC nodes. For reducing the number of classes and nodes in the graphs this is indeed a mixture between a relation and an object and usually should be used like in Figure 6.

```
ISEC---ROAD---ISEC
```

Figure 6: Road Object.

It would be possible to introduce another relation LINK to separate objects and relations like shown in Figure 7 but in practice this is not necessary.

```
ISEC---LINK---ROAD---LINK---ISEC
```

Figure 7: Alternative construction of road objects.

4.2 Relations

Relations are connecting the objects of a given sketch or reference data set but are modelled in the same way as the latter. They are nodes of the description graph and define links to other nodes, representing objects or other relations.
Some of the relations are connecting other nodes and provide a certain meaning to this connection but some are only linked with one node which has the meaning of adding information to the target node.

The ISEC objects are the result of incident road segments at an intersection. But after introduction of an ISEC instance the information about the cardinality information is discarded. The cardinality information can be reintroduced by using the classes **C1**, **C2**, **C3**, **C4**, **C5**, ..., **CN** where CN means there are exactly `n` or more than `n` road segments (Figure 8).

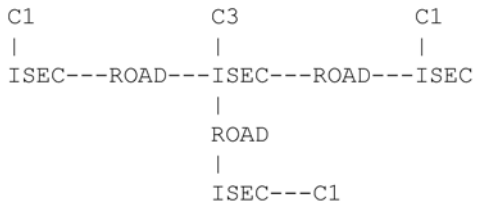

Figure 8: Cardinal relations.

A special case is the C2 relation which includes more information than the other cardinalities. To reflect the existence of an inflection point it should be called the **INF** relation.
If more than two road segments meet at an intersection, it is interesting to know if any pair of roads is a straight continuation of each other or if the pair is orthogonal. This situation is very common in cities and essential for orientation. Such a relation can be detected by comparing the direction of road segments. In the real world the conditions must be relaxed because a straight road remains a straight road if a small error is present in the angle between the road segments. Both relations refer to the ROAD concepts for which the property holds true. The notation of the straight relation is **STR** while the notation of the orthogonal relation is **ORTH**. Their use is shown here (Figure 9):

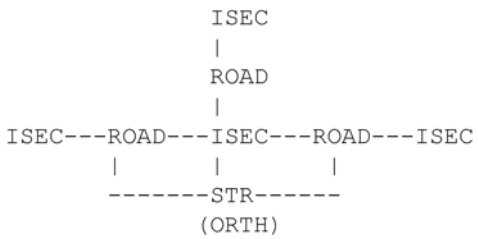

Figure 9: Straight / Orthogonal relation.

4.3 Sequential relations

In some situations natural sequences are formed by objects and the sequence is of a special type. In this cases always a predecessor and a successor object exists which can be marked by the **PRED** and **SUCC** relations. They are used together and each of them links to the ordered concept OBJ. The type is determined by another relation REL which is linked between the PRED and SUCC concepts. The principle is shown in Figure 10.

Figure 10: Sequential relations (REL) between two objects (OBJ).

This kind of construction is needed for junctions where the road segments are ordered by their direction around the central intersection. In the graph it would not be possible to reconstruct the sequence of road segments because the direction information is lost. But the sequential order of roads is a very strong constraint for a junction. Here another relation with the name **ORD** can be introduced (Figure 11). Then it is not possible to generate ambiguously directed junctions.

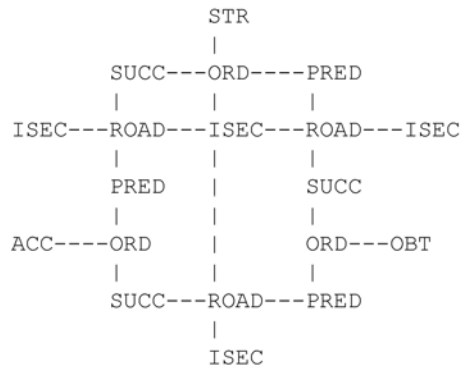

Figure 11: Ordered road segments.

This construction does not require any special angle between the roads. If this restriction is wanted the STR and ORTH relations can be used and for obtuse angles and acute angles the relations **OBT** and **ACC** will fit. They should all four have a single link to the ORD node (Figure 12).

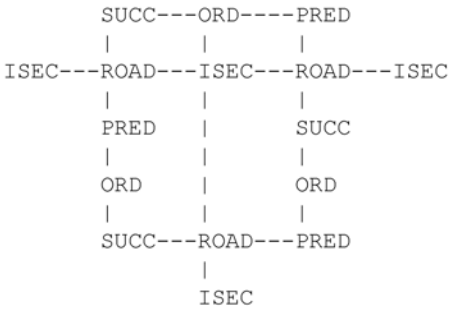

Figure 12: More restricted ordered road segments.

Because the sketch is not an exact projection of real objects this will need an error tolerant algorithm for the matching process where wrong relations are accepted until a limit of errors is reached. The direction constraints are defined here with exactly separating angle values but this hard cut does not reflect reality.

4.4 Non-Road objects

This section concentrated on road networks because they are a vital component in most situations. But there are of course more objects than roads which are important too. Depending on the type of the objects new relations can be defined. The most important is the neighbourhood relation which is connecting two objects. If the neighbourhood is found in the reference it can be expected that it is preserved in the sketch. The consequences of using this relation have not been analyzed yet but the 9-intersection of (Egenhofer, 1991) is a good foundation for working out a suitable set of relations.

5. CONVERTING DATA

The matching process requires the transformation of reference data and sketch data to a graph representation. The sketch must be processed on the fly in the context of the search engine but the reference data can be pre-processed. So there is no need to do all the conversions in real time which is impossible on available hardware. The sketch data is usually small enough for online processing.
Actually the data is taken from ESRI shape files where it is organized as simple features in distinct layers for every feature type. The layers with the relevant information can be selected

and a triangulation is performed on the geometry. Here the constrained delaunay triangulation is a good choice because lines of the data become edges of the triangulation.
The triangulation produces a topological representation of the data projected to one single plane and if a topological data structure is used it supports the efficient extraction of topological constraints.
For the case of the road network simple edge tracing can answer most of the questions directly or with some additional geometric calculations on local elements of the triangulation. Neighbourhood is an inherent part of the definition of the delaunay triangulations and voronoi diagrams. It can be extracted from the edges by checking the features on the edge itself or at the connected nodes of the edge.
The sketch data needs a reconstruction of the contained objects before the conversion to the graph representation can be started, which involves another pattern matching process. This process is not subject of this article.

6. TESTING

The recognition of a sketched position was tested with some simple sketches and for the case of exact matching it performed very well. If the exact solution existed it was found immediately within a second. For bigger patterns it is likely to have no exact matching location in the reference and error tolerant matching algorithms must be applied.
Another tested application of the matching algorithm was the matching of two road networks from the same area but different data providers. Here the geometry is not very distorted but the size of the pattern has the same size like the reference and at some points there are missing roads, missing inflection points or another shape of the road. The area shown in Figure 13 produced two graphs with the size of approx. 570 nodes. An interesting test was the matching of the road network to itself. The correct solution was found immediately. When using both target and destination, no solution could be found but the largest common sub graph has 55 Nodes of correctly assigned nodes.

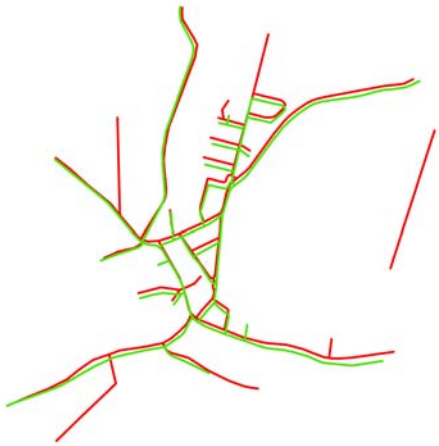

Figure 13: Test area for matching of two road graphs.

7. CONCLUSIONS AND FUTURE WORK

This article showed how the location of a spatial situation can be solved with a sketch as a pattern. This offers the possibility of producing abstract queries where only relations of the objects are defined but not the objects themselves.
The sketch is compared to a reference data set by transforming it to graph structure which is a an explicit representation of the implicit knowledge important for the matching process. A set of classes which are needed for this representation is developed.
The matching process does implement exact graph matching at this time and is very quick if no errors are in the data. Inexact graph matching is still under development and needs further investigations on the exploitable information to find suitable heuristics which are fast enough. The literature shows some example in other applications comparable with the complexity of the problem discussed here. The success of their solutions should be transferable to the sketch matching problem. The future work will also concentrate on investigating to which degree the geometric constraints can be introduced into the search procedure.

8. REFERENCES

Bengoetxea, E., 2002. Inexact Graph Matching Using Estimation of Distribution Algorithms. Dissertation, Paris, France.

Blaser, A., 1998, Geo-Spatial Sketches. Technical Report, National Center of Geographic Information and Analysis, University of Maine, Orono, Maine, USA.

Blaser, A., 2000. Sketching Spatial Queries. Dissertation, The Graduate School, University of Maine, Orono, Maine, USA.

Cordella, L.P., Foggia, P., Sansone, C., Vento, M., 2001. An Improved Algorithm for Matching Large Graphs. Proc. of the 3rd IAPR TC-15 Workshop on Graph-based Representations in Pattern Recognition, Ischia, May 2001, p. 149-159.

Cortadella, J., Valiente, G., 2000. A Relational View of Subgraph Isomorphism. Proc. Fifth International Seminar on Relational Methods in Computer Science, 2000, pp.45-54.

Egenhofer, M.J., Franzosa, R.D., 1991. Point-Set Topological Spatial Relations. International Journal of Geographical Information Systems 5, pp. 161-174.

Jones, C.B. et al, 2002. Spatial Information Retrieval and Geographical Ontologies : An Overview of the SPIRIT project. SIGIR 2002: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2002, pp. 387-388.

Lee, R.S.T., Liu, J.N.K., 2002. An Automatic Tropical Cyclone Pattern Recognition and Classification System using Composite Neural Oscillatory-based EGDLM. International Journal of Fuzzy Systems, Vol. 4, No. 1, 2002.

Lipschutz, S., 1976. Discrete Mathematics. McGraw-Hill, New York, pp. 89-90.

Sowa, F., 2001. Conceptual Graphs, ISO standard, Reference: ISO/JTC1/SC 32/WG2 N 000.

Ullmann, J.R., 1976. An Algorithm for Subgraph Isomorphism. Journal of the ACM (JACM), Vol. 23, Issue 1, pp. 31-42.

9. ACKNOWLEDGEMENTS

This work is part of the project SPIRIT (Spatially-Aware Retrieval on the Internet) IST-2001-35047 which is funded by the EU.