Spatial and Semantic Reasoning to Recognize Ship Behavior

Willem R. van Hage
VU University Amsterdam
wrvhage@few.vu.nl

Gerben de Vries
University of Amsterdam
G.K.D.deVries@uva.nl

Véronique Malaisé
VU University Amsterdam
vmalaise@few.vu.nl

Guus Schreiber
VU University Amsterdam
schreiber@cs.vu.nl

Maarten van Someren
University of Amsterdam
M.W.vanSomeren@uva.nl

ABSTRACT

This demo shows the integration of spatial and semantic reasoning for the recognition of ship behavior. We recognize abstract behavior such as "ferry trip" and derive that the ship showing this behavior is a "ferry". We accomplish this by abstracting over low-level ship trajectory data and applying Prolog rules that express properties of ship behavior. These rules make use of the GeoNames ontology and a spatial indexing package for SWI-Prolog, which is available as open source software.

1. INTRODUCTION

We demonstrate the integration of spatial and semantic reasoning for the recognition of ship behavior and ship types. The spatial reasoning is based on GeoNames[1] and a spatial indexing package[2]. The semantic reasoning consists of SWI-Prolog [3] rules that make use of RDFS reasoning over a maritime domain ontology and low-level behavior events. These events are recognized in sensor data with MatLab. The sensor data we use are Marine Automatic Identification System (AIS)[3] messages, sent by all ships over 300 tons at a regular interval to receivers. AIS messages post the ship's navigational parameters, like speed, course, and rate of turn. From these messages we extract ship track segments where the ship shows constant behavior (*Steady speed*, *Slowing down*, *Turning*, etc.). We use the rules to recognize higher level behavior of ships (*Departing*, *Trip*, and *Ferry Trip*) and new ship types (*Ferry*). We visualize the recognized behavior types with colored trajectories in KML so that it can be displayed in Google Earth. A detailed description of the system can be found in [2].

[1] `http://www.geonames.org/ontology/`
[2] The SWI-Prolog *Space* package: `http://www.swi-prolog.org/pldoc/package/space.html`
[3] `http://en.wikipedia.org/wiki/Automatic_Identification_System`

2. INTERPRETING SENSOR DATA

We transform the AIS data describing ship trajectories into movement predicates using a compression algorithm: the Piecewise Linear Segmentation (see [2]). Their size and the nature of the sea restricts ships from sudden changes in behavior. Hence, a compression algorithm gives good results in creating abstract segments of constant behavior. These segments have the advantages of reducing the size of the data and of defining, geographically, points in space where changes in behavior occur. The reduction in data size allows faster reasoning over the segments than over single messages. These segments are the semantic building blocks for defining movement predicates. An example is shown in figure 2.

3. SPACE AND SEMANTICS

To define complex concepts in the maritime domain, ontological reasoning has to be tied to geographical reasoning. This integration is done, in our case, via a knowledge-base in SWI-Prolog (see figure 1). We have implemented a spatial indexing package[2] for fast nearest-neighbor, intersection, and containment search in geographic data, that can be used in combination with ontologies. The space package associates every geographical shape with a URI. The segments abstracted from the aforementioned data have begin and end places, defined as spatial coordinates. The comparison of these coordinates with GeoNames enables to know whether a ship is anchored (*Stopped at Anchorspot*) or moored (*Stopped at Harbor*). On top of these events we define a *Trip* as an abstract event (i.e. independently from the low-level data). A *Trip* is a succession of connected Segments, which begin and stop at a harbor (geonames:H.HBR).

4. CLASSIFYING FERRY BEHAVIOR

An example of reasoning that implies domain knowledge and geographic concepts is the classifying of a Ferry ship. A Ferry is a ship that goes back and forth between the two *same* harbors. Not all back and forth going ships are Ferries. Some are, for example, Dredgers or Rescue Vessels. Whereas the definition of a ship going "back and forth" can be done based on rules written over movement predicates [1], the fact that the movement has to occur between the same harbors can only be defined with the help of a geographic ontology and geographic proximity reasoning (the `space_nearest` predicate of the space package in this case).

The demo is accessible online. The unclassified ship trajec-

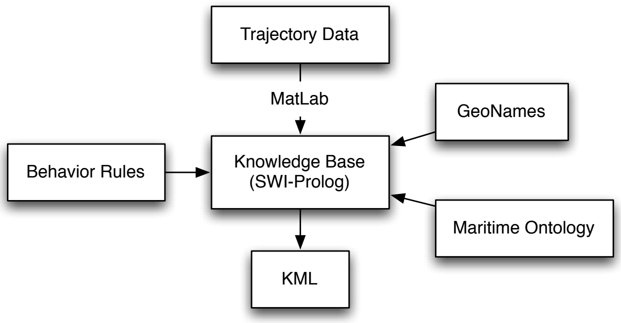

Figure 1: Architecture of the system.

tories can be seen at `http://semanticweb.cs.vu.nl:8203/event/iswc2009_ship_behavior.kml` and the classified ship trajectories can be seen at `http://semanticweb.cs.vu.nl:8204/event/iswc2009_ship_behavior.kml`. In this demo we only display speed-related behavior types and ferry trips, although there are more behavior and ship types in the knowledge base. Green lines represent parts of a ship trajectory where it is constantly speeding up, ochre lines show steady speed, red lines show slowing down, the bright red lines show the locations where the ship is stopped. Besides this speed-based classification, we show a set of ship trajectories between two Dutch cities. Without further information, these trajectories are displayed in grey. After running the reasoner, these trips are recognised as Ferry trips. Hence, the ships which are doing these trajectories are classified as Ferry boats. The trajectories being classified as Ferry trips are displayed as bold white lines. The last trajectories shown are ones of a Rescue Vessel, going back and forth from and to the same harbor (the rescue vessel hangar). Although the movement pattern is similar to the one of the Ferry, this ship's movement is rightfully not classified as a Ferry trip, because it only involves one harbor. A summarized snippet of the Prolog code recognizing ferry trips is shown in figure 2.

5. CONCLUSION

This demo shows a combination of semantic and spatial reasoning in SWI-Prolog. The software is publicly available and extensible. The integration demonstrated here can be applied in other domains (*e.g.* multimedia). We show how we populate a domain ontology from sensor data and derive high-level events. Abstracting over the sensor data allows integration with semantic resources like GeoNames and event models like SEM (*cf.* [2]). We will investigate abstract space and time reasoning (*cf.* *e.g.* RCC-8 or event calculi), integrating knowledge gathered from the web (*e.g.* ship characteristics and history), and attempting to learn behavior rules as opposed to writing them by hand. Also, working with behavior involving multiple objects remains future work.

6. ACKNOWLEDGMENTS

This work has been carried out as a part of the Poseidon project in collaboration with Thales Nederland, under the responsibilities of the Embedded Systems Institute (ESI). This project is partially supported by the Dutch Ministry of Economic Affairs under the BSIK program.

```
% rule defining ferry behavior semantics
% trip from Hbr0 to Hbr1 and back
% via consistent movement Segment Seg0, Seg1, and Seg2
ferry_trip(Seg0, Hbr0, Seg1, Hbr1, Seg2, Hbr0) :-
    trip(Seg0, Hbr0, Seg1, Hbr1),
    trip(Seg1, Hbr1, Seg2, Hbr0).

% rule defining trip behavior semantics
trip(Seg0, Hbr0, Seg1, Hbr1) :-
    % ship is stopped at harbor Hbr0 during Seg0
    stopped_at_harbor(Seg0, Hbr0),
    % moving between Seg0 and Seg1
    move_from_to(Seg0, Seg1),  % code omitted in example
    % ship is stopped at harbor Hbr1 during Seg1
    stopped_at_harbor(Seg1, Hbr1).

% rule defining trip behavior semantics
stopped_at_harbor(Segment, Hbr) :-
    stopped(Segment),  % semantics of behavior
    location_of_segment(Segment,Location)
    % find nearest place within margin in spatial index
    space_nearest_bounded(Location, Hbr, 0.175),
    % semantics of place
    rdf(Hbr, geo:featureCode, geo:'H.HBR').

% ship trajectory reasoning
successor(Seg0, Seg1) :-
    Seg0 = seg(_,Ship,_,_,_,_,_,_,_T0,T1),
    Seg1 = seg(_,Ship,_,_,_,_,_,_,T1,_T2).

move_from_to(Seg0, SegN) :-
        successor(Seg0, SegN) ;   % stop condition
        successor(Seg0, Seg1),
        move_from_to(Seg1, SegN). % recursion
```

Figure 2: Selected rules to recognize ferry trips.

7. REFERENCES

[1] G. de Vries, V. Malaisé, M. van Someren, P. Adriaans, and G. Schreiber. Semi-automatic ontology extension in the maritime domain. In *Proceedings of BNAIC 2008*, 2008.

[2] W. R. van Hage, V. Malaisé, G. de Vries, G. Schreiber, and M. van Someren. Combining ship trajectories and semantics with the simple event model (sem). In *Proceedings of the 1st ACM International Workshop on Events in Multimedia*. Sheridan Publishers, 2009.

[3] J. Wielemaker, Z. Huang, and L. van der Meij. Swi-prolog and the web. In A. Bossi, editor, *Theory and Practice of Logic Programming*, volume 8, pages 363–392. Cambridge University Press, 2008.