International Journal of Geographical Information Science
Vol. 24, No. 9, September 2010, 1–16

An integrated approach for visual analysis of a multi-source moving objects knowledge base

Niels Willems[a] *, Willem Robert van Hage[b], Gerben de Vries[c], Jeroen H.M. Janssens[d], Véronique Malaisé[b]

[a]*Department of Mathematics and Computer Science, Eindhoven University of Technology, Eindhoven, The Netherlands*; [b]*Web & Media Group, VU University Amsterdam, Amsterdam, The Netherlands*; [c]*Informatics Institute, University of Amsterdam, Amsterdam, The Netherlands*; [d]*Tilburg centre for Creative Computing, Tilburg University, Tilburg, The Netherlands*

(*Received 23 April 2010; final version received 6 August 2010*)

We present an integrated and multi-disciplinary approach for analyzing behavior of moving objects. The results originate from ongoing research of four different partners from the Dutch Poseidon project (Embedded Systems Institute (2007)), which aims to develop new methods for Maritime Safety and Security (MSS) systems to monitor vessel traffic in coastal areas. Our architecture enables an operator to visually test hypotheses about vessels with time-dependent sensor data and on-demand external knowledge. The system includes the following components: abstraction and simulation of trajectory sensor data, fusion of multiple heterogeneous data sources, reasoning, and visual analysis of the combined data sources. We start by extracting segments of consistent movement from simulated or real-world trajectory data, which we store as instances of the Simple Event Model (SEM), an event ontology represented in the Resource Description Framework (RDF). Next, we add data from the web about vessels and geography to enrich the sensor data. This additional information is integrated with the representation of the vessels (actors) and places in SEM. The enriched trajectory data is stored in a knowledge base, which can be further annotated by reasoning and is queried by a visual analytics tool to search for spatio-temporal patterns. Although our approach is dedicated to MSS systems, we expect it to be useful in other domains.

Keywords: Trajectory Generation, Trajectory Generalization, Event Modeling, Semantic Web, Visual Analytics

*Corresponding author. Email: c.m.e.willems@tue.nl

ISSN: 1365-8816 print/ISSN 1362-3087 online
© 2010 Taylor & Francis
DOI: 10.1080/1365881YYxxxxxxxx
http://www.informaworld.com

1. Introduction

Since Hägerstrand (1952) in the early fifties, trajectories of moving objects have been studied by searching for spatio-temporal patterns (or simply patterns), such as the ones structured by Dodge *et al.* (2008). Current technology can be used to register behavior on a large scale: the movements of many objects are captured in extensive *time-series*, which requires semi-automatical analysis. Examples of approaches to visually analyze moving objects are: clustering as proposed by Andrienko *et al.* (2009), density by Willems *et al.* (2009), and glyphs as described by Bak *et al.* (2009).

Today's tracking devices lack the capabilities to capture semantically-rich data related to the patterns we are searching for. Often, only the geometry of the trajectory is captured, which makes finding patterns complex. Different useful patterns may be found by using additional data, which is obtained by extra sensors, manual labeling, or results from computational methods as suggested by Andrienko and Andrienko (2007). We focus on visually detecting patterns for testing hypotheses about the behavior of moving objects using multiple heterogeneous data sources containing derived data and existing knowledge about the objects and their context; one of the visual analytics challenges as identified by Thomas and Kielman (2009). For instance, in aviation, an analyst may want to test whether civil airplanes landed on a military airport, which requires sensor data from the airplanes and a geographical context to determine the type of airport.

Our approach is demonstrated in the maritime domain for monitoring vessels in coastal areas and is a prototype for a part of a so called Maritime Safety and Security (MSS) system. Figure 1 displays the architecture containing an overview of the tools and interfaces. Three types of low-level data sources are aggregated in our knowledge base (see Sec. 4): vessel trajectories obtained by means of the Automatic Identification System (AIS) or simulated with Presto (see Sec. 3) and web data from various sources about vessels and their (geographical) context. The trajectories are converted into meaningful Simple Event Model (SEM) instances (see Sec. 4.1) after having been preprocessed with Piecewise Linear Segmentation (see Sec. 3.2) to generalize over the massive amount of sensor data and obtain a smaller, but semantically equivalent data set that can be processed more efficiently. The web data are added to these SEM event instances in an SWI-Prolog based RDF store (see Sec. 4.2). Prolog is ideally suited for reasoning over logical knowledge facts, such as RDF, because it is a declarative logic programming language. Deduction rules can be used to define behavior on top of the movement events and by reasoning we build up knowledge, for instance by adding new attributes to a trajectory. The visual interface is an interactive trajectory contingency table enabling a user to find patterns, such as correlations between attributes, in the knowledge base (see Sec. 5). The trajectories generated with Presto allow us to evaluate the performance of the system with scenarios containing ground truth (see Sec. 6). The last two sections elaborate on the system as a whole and make suggestions for future works.

2. Related work

There are some integrated frameworks for analyzing spatio-temporal data sets. However, these applications focus on one aspect at a time: Chen *et al.* (2010) focus on combining heterogeneous data sources, Wood *et al.* (2007) investigate visualization of patterns, or Bertolotto *et al.* (2007) focus on mining patterns. We combine these three aspects.

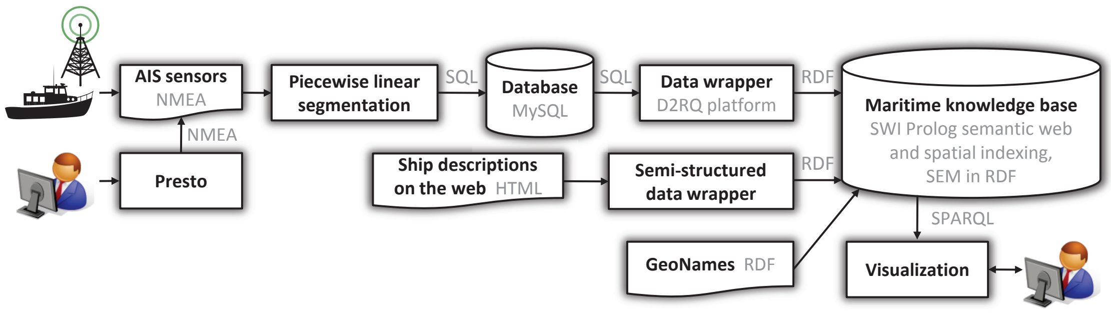

AIS sensors NMEA
NMEA
Presto
Piecewise linear segmentation
SQL
Database MySQL
SQL
Data wrapper D2RQ platform
RDF
Ship descriptions on the web HTML
Semi-structured data wrapper
RDF
GeoNames RDF
Maritime knowledge base SWI Prolog semantic web and spatial indexing, SEM in RDF
SPARQL
Visualization

Figure 1. The architecture of our approach for the analysis of a multi-source moving object knowledge base as a flowchart. Straight boxes are processes, curved bottom boxes are documents, and cylinders are databases. Interfaces and platforms are shown in light grey.

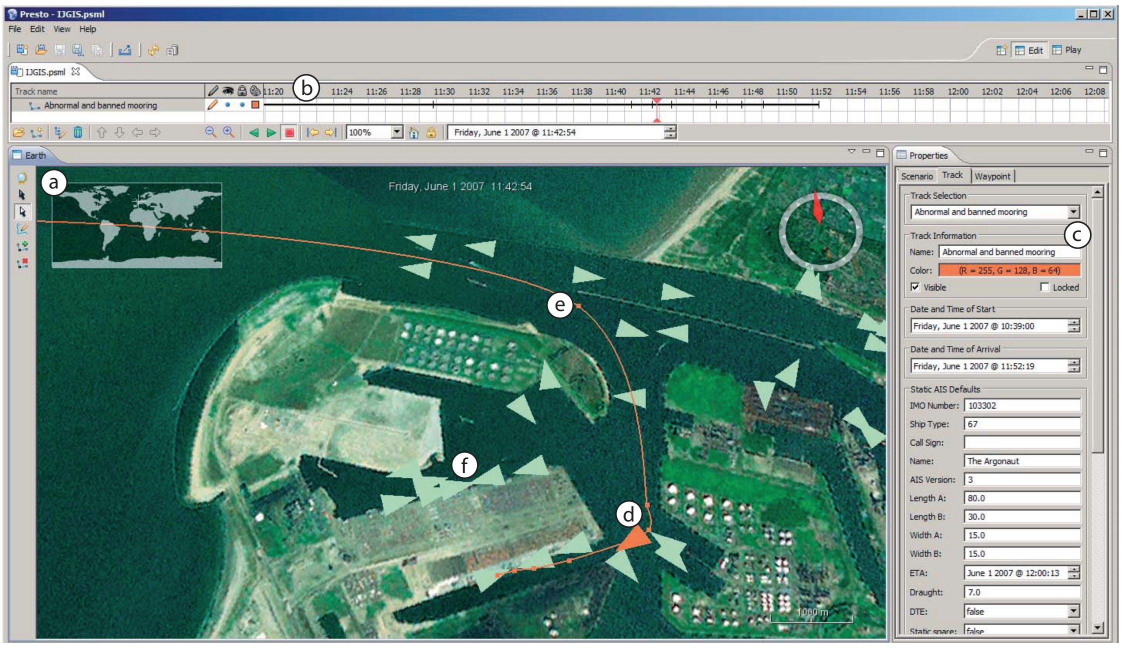
Sánchez *et al.* (2009) reason about moving objects in a knowledge base, but its reasoning serves the construction of the actual trajectories, and is not involved in any further analysis. Yan *et al.* (2008) proposed a trajectory ontology that allows for querying and reasoning with homogeneous trajectory data, but it is limited by not incorporating other data sources. A knowledge base has been used in only a few visual analytic systems, and none of these focus on spatio-temporal data. Xiao *et al.* (2006) visually analyze network traffic with a knowledge representation. The HARVEST system by Gotz *et al.* (2006) is based on the principle to synthesize knowledge for progressive analysis.

Semantic analysis work comparable to our analysis of moving objects has been reported by Orellana and Renso (2010). This work mainly focuses on describing collective behavior in OWL, whereas we describe an entire integrated workflow from the level of sensor data to the level of behavior semantics. Also, we choose to use Prolog as our reasoning tool as opposed to an OWL reasoner. A methodology for modeling faceted *who, what, where, when* data with the CIDOC-CRM event model has been proposed by Hiebel *et al.* (2010). The ontology we use, SEM, is much smaller than CIDOC-CRM. SEM provides a mapping of all its constructs to CIDOC-CRM. A survey of semantic approaches for the integration of geospatial data has been undertaken by Buccella *et al.* (2009). An approach to combine spatial and temporal semantics of objects that could be applied to the Place and Time facets of SEM Events is described by Hornsby and Joshi (2010).

3. Trajectory data

The continuous movement of an object is captured by sampling its trajectory at various moments in time. A sample contains all information about the object and its movement. We model a trajectory T as a sequence $\boldsymbol{\alpha}_0, \ldots, \boldsymbol{\alpha}_{N-1}$ of n-tuples $\boldsymbol{\alpha}_i = (\alpha_0^i, \alpha_1^i, \ldots, \alpha_{n-1}^i)$, where the j-th element of the i-th tuple contains attribute α_j^i. Each tuple includes the attributes time t_i, position $\mathbf{p}_i$, velocity v_i, and object identifier o_i.

The moving objects we observe are vessels with an Automatic Identification System (AIS) device on board, which are mainly large and commercial vessels. AIS is an advanced Global Positioning System (GPS) device that frequently broadcasts messages with data from the vessel and its movement according to a protocol specified by the ITU (2001). The data contains many attributes; for vessels we have identification numbers, a name, dimensions, and a (broad) type, e.g. passenger ship or tanker. The attributes for the movement are, for instance, position, time, velocity, destination, draught, and navigational status, e.g. at anchor, moored, or fishing. The latter comes from a controlled



Track name: Abnormal and banned mooring

100% | Friday, June 1 2007 @ 11:42:54

Track Selection: Abnormal and banned mooring

Track Information
Name: Abnormal and banned mooring
Color: (R = 255, G = 128, B = 64)
[✓ Visible] Locked

Date and Time of Start: Friday, June 1 2007 @ 10:39:00

Date and Time of Arrival: Friday, June 1 2007 @ 11:52:19

Static AIS Defaults

IMO Number:	103302
Ship Type:	67
Call Sign:	
Name:	The Argonaut
AIS Version:	3
Length A:	80.0
Length B:	30.0
Width A:	15.0
Width B:	15.0
ETA:	June 1 2007 @ 12:00:13
Draught:	7.0
DTE:	false
Static spare:	false

Figure 2. The screenshot illustrates the user interface elements and concepts of Presto: (a) the world map, (b) the timeline showing the trajectory in the scenario, (c) the property editor showing the trajectory properties, (d) the current location of the artificial vessel displayed as an orange triangle along the created trajectory, (e) one waypoint of the artificial trajectory as a dot, and (f) the background data at the current moment displayed with a green triangle for each vessel.

vocabulary in the AIS protocol. Captains use AIS to receive the status of neighboring vessels to prevent collisions, but using a sensor network connected to an MSS system, AIS can also be used for real-time monitoring and long-term analysis.

3.1. *Creating artificial vessel trajectories with Presto*

An MSS system aims at guiding an operator in finding maritime anomalies, such as vessel traffic violations, illegal fishing activities, and drug smuggling. In real-world data, serious maritime anomalies rarely occur, therefore trajectories with ground truth are needed to evaluate whether arbitrary anomalies can be detected properly. To this end, we have developed an application, called Presto, which enables maritime domain-experts to easily create anomalies, including the ones mentioned in Janssens *et al.* (2010). In contrast to existing simulation applications, such as VR-Forces[1], which impose restrictive behavior models, our application gives the expert full control over the vessel trajectories.

Figure 2 shows the concepts and user interface elements of Presto. The main concept in Presto is a scenario, which can contain one or more trajectories. Each trajectory is defined by several waypoints , which is a location with additional parameters: velocity, time, and curvature. The user interface of Presto consists of the following main elements: a world map (Fig. 2(a)) based on NASA World Wind[2] to position the trajectories and their waypoints, the timeline (Fig. 2(b)) to navigate through time, and the property editor (Fig. 2(c)) to edit specific parameters (e.g., for a scenario: name, author, and description; for a trajectory: name, call sign, and flag; and for a waypoint: location, time,

[1]VR-Forces: http://mak.com/products/vrforces.php
[2]NASA World Wind: http://worldwind.arc.nasa.gov

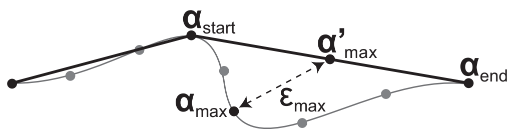


Figure 3. A compressed trajectory after one step with the PLS algorithm.

velocity, and curvature). We can load existing, real AIS data as so-called background data (Fig. 2(f)) , which serves as a reference when creating new artificial trajectories, so that, for instance, collisions can be avoided or enforced.

A trajectory is generated from the waypoints using the following iterative procedure. Given a location, velocity, and time as provided by the current waypoint, the vessel starts sailing towards the next waypoint for a user-defined time step, after which the location, velocity, and bearing of the vessel are updated. This is repeated until the vessel reaches the next waypoint. When the vessel passes the waypoint within the time step, all variables are adjusted such that the vessel continues exactly at the waypoint. Rather than using a linear interpolation for the position between two waypoints, which result in unnaturally sharp turns, we employ Bézier interpolation with control points placed at a location defined by the curvature variable. A lower curvature value causes the control points to be placed further away from the waypoints, resulting in a smoother and larger turn of the vessel. The velocity between waypoints is interpolated using exponential easing. The generated data is converted to data according to the AIS protocol and fused with real-world AIS data, such that the artificial data cannot be distinguished up front from the original data by the system in the subsequent processing.

For the use cases in Section 6, we have created a trajectory of a passenger ship that goes to an anchorage area for cargo storage in Rotterdam harbor (see Fig. 2). After integration with the other six hundred ship trajectories in the knowledge base, our movement pattern recognition rules can be tested. This should lead to a conflict, since a passenger ship should anchor at a passenger terminal.

3.2. *Piecewise linear segmentation*

Vessels are constrained in their movement and, consequently, they often sail along predictable courses. However, AIS messages are broadcasted frequently, hence many subsequent messages do not differ semantically. Therefore, we compress these messages into segments of consistent behavior, which has the advantage that we drastically reduce computation time for the higher level reasoning (Sec. 6).

The basis for our trajectory compression is a Piecewise Linear Segmentation (PLS) method, such as studied in Gudmundsson *et al.* (2009), and Cao *et al.* (2006) . These algorithms are based on a well-known efficient compression algorithm for polylines introduced by Douglas and Peucker (1973), but have been adapted to take the temporal component of a trajectory into account. Figure 3 illustrates a generic version of the PLS-algorithm. This algorithm compresses a sequence of tuples $\boldsymbol{\alpha}_{start}, \ldots, \boldsymbol{\alpha}_{end}$ by finding the tuple $\boldsymbol{\alpha}_{max}$ that has the largest error $\epsilon_{max} = E(\boldsymbol{\alpha}_{max}, \boldsymbol{\alpha}'_{max})$ with respect to the reconstructed tuple $\boldsymbol{\alpha}'_{max} = R(\boldsymbol{\alpha}_{start}, \boldsymbol{\alpha}_{max}, \boldsymbol{\alpha}_{end})$, for some error function E and reconstruction function R. If ϵ_{max} is larger than some given error ϵ, then we select $\boldsymbol{\alpha}_{max}$ and recursively compress the sequences $\boldsymbol{\alpha}_{start}, \ldots, \boldsymbol{\alpha}_{max}$ and $\boldsymbol{\alpha}_{max}, \ldots, \boldsymbol{\alpha}_{end}$, otherwise the algorithm stops and accepts $\boldsymbol{\alpha}_{start}, \boldsymbol{\alpha}_{end}$.

The concepts of *stop* and *move* have been argued to be important semantic movement primitives by Spaccapietra *et al.* (2008). A big part of the higher level reasoning in Section

6 is based on these concepts. To better preserve stops and moves we have investigated and created a two-stage variant of the PLS-algorithm. In the first stage, tuples $\boldsymbol{\alpha}_i$ are selected, using the PLS-algorithm, based on the velocity of the vessel. The velocity v'_{max} is reconstructed with R being the linear interpolation between v_{start} and v_{end} at t_{max} and the error E is the absolute difference between v'_{max} and v_{max}. Then, in the second stage, the positions between each pair of points selected in the first stage are compressed, with PLS, as a polyline, where E is the Euclidian distance between $\mathbf{p}_{max}$ and $\mathbf{p}'_{max}$, which is reconstructed by R, the projection of $\mathbf{p}_{max}$ on the line $\mathbf{p}_{start}\mathbf{p}_{end}$.

To test whether the two-stage variant retains more stops at the same compression rate (i.e., the amount of ignored data) than other variants of PLS-based trajectory compression techniques, we created a data set of vessel trajectories with 281 manually labelled stops. We define a vessel to be stopped between two consecutive tuples $\boldsymbol{\alpha}_i$ and $\boldsymbol{\alpha}_j$ if:

$$||\mathbf{p}_j - \mathbf{p}_i||/(t_j - t_i) \leq \theta \ , \tag{1}$$

where θ is a threshold velocity. From experience, we noticed that in the maritime domain $\theta = 0.05$ knots ($\approx 0.093\text{kmh}^{-1}$) is a good threshold for retaining stops.

We give a small excerpt of the experiments that we have conducted. From earlier work on trajectory compression algorithms we selected the two error measures that retained the most stops. We used E_μ from Gudmundsson *et al.* (2009), for which we selected the best value for μ (in terms of stop retention) that we could find, and E_t from Cao *et al.* (2006). With both these error measures a trajectory is treated as a polyline in 3D, with t being the third dimension. The E_μ measure is an Euclidian distance in this 3D space, where μ is the weight of the time dimension. The error measure E_t focusses purely on the temporal difference between the original point and its 2D interpolation. In Figure 4 we have plotted the performance of PLS with error measures E_μ and E_t, and our two-stage variant. For the different algorithms the compression rate versus the stop retention rate is shown, with 100% being all stops retained.

We say that a stop is retained by a compressed trajectory if there are two consecutive tuples for which equation (1) holds and a match occurs with a manually labelled stop. The compression rate is computed on a separate data set of 400 vessel trajectories. We can see that for high compression rates our two-stage approach retains more stops than the single-stage approaches, which is what we are after in this application. This is due to the fact that the first stage of compression focusses purely on the velocity of the vessel, which determines whether a vessel is stopped or not. Notice that the high compression rates, i.e. 98% in TwoStage instead of 93% in E_μ for retaining 95% of the stops, allows us to triplicate the amount of input sensor data while keeping the amount of output data constant, which is beneficial for large data sets. We can use Figure 4 to select the appropriate settings for our two-stage algorithm, i.e. $\epsilon_p = 50$m and $\epsilon_v = 2.5$ knots.

The result of the two-stage PLS-algorithm, is stored in a MySQL-database, where each record is a segment that describes a constant piece of movement. Let T be a trajectory as defined earlier, then T^C is its compressed variant resulting from our two-stage algorithm. For each consecutive pair $\boldsymbol{\alpha}_i, \boldsymbol{\alpha}_j$ in T^C we create a record $\langle \texttt{uri}, \mathbf{p}_i, \mathbf{p}_j, t_i, t_j, v_i, v_j, c_i, c_j \rangle$. The `uri` is a Uniform Resource Identifier (URI), which uniquely identifies each segment in our knowledge base, and is based on the ship's unique identifier, the Maritime Mobile Service Identity (MMSI) number, and the start time t_i. Furthermore, the segment contains a start position $\mathbf{p}_i$ and end position $\mathbf{p}_j$, a start time t_i and end time t_j, a start speed v_i and end speed v_j, and a start course c_i and end course c_j. The segments constitute the first level of abstraction for meaningful movements, which are represented using the

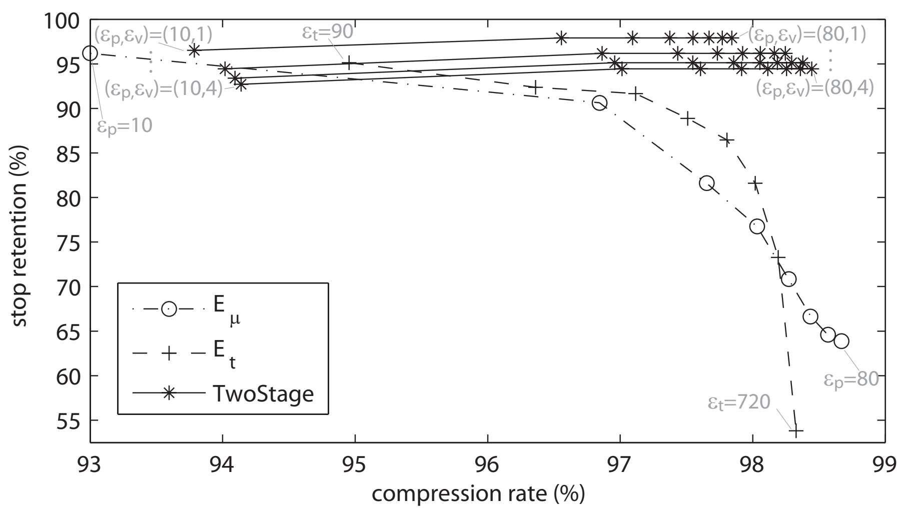


Figure 4. Results of stop retention experiment. For E_μ we used $\epsilon_p = 10, 20, 30, 40, 50, 60, 70, 80$m and for E_t we used $\epsilon_t = 90, 180, 270, 360, 450, 540, 630, 720$s. For our two-stage variant we used the same ϵ_p and conduct the experiment for velocity parameters $\epsilon_v = 1, 2, 3, 4$ knots, resulting in a graph for each velocity with four knots the lowest graph.

Simple Event Model (see Sec. 4.1) and can be agglomerated (and combined with other resources) into complex behavior definitions.

The movement primitives *stop* and *move* are added by attaching a property with value ex:stopped to each segment with an average speed below the threshold θ and the value ex:move to all other segments. By looking at the acceleration, computed by $(v_j - v_i)/(t_j - t_i)$, we also assign the values ex:speeding_up and ex:slowing_down to the sem:eventType (see Sec. 4.1) property of a segment. Additionally, the set of movement primitives can be extended with more semantic classes, like ex:fast_move, or other features, such as direction, i.e. ex:to_North. While direction can be obtained by thresholding, the ex:fast_move needs reasoning, since the notion of fast is related to ship type and weather conditions.

4. Semantic web technology

Only a limited number of movement patterns and useful vessel behavior can be detected on the sole basis of the segments. Richer behavior definition and hypothesis testing about single vessel movement requires the combined knowledge about the vessel's position, this position's characteristics (e.g. geographic type), movement characteristics and the vessel's type: the same movement pattern may have different implications across object types. For instance, if a vessel shuttles between two locations, it may be a dredger or a ferry. Without further knowledge, it is not possible to disambiguate between these two. But web-based information can fill in some missing data, and help enrich the event description further. If we know that a vessel is a dredger, then we can determine that one of the locations is a dredging place. Equally, if one of the locations is a dredging place, then the vessel is a dredger. In order to reason about information coming from different sources, integration is needed, and therefore we have developed the Simple Event Model.

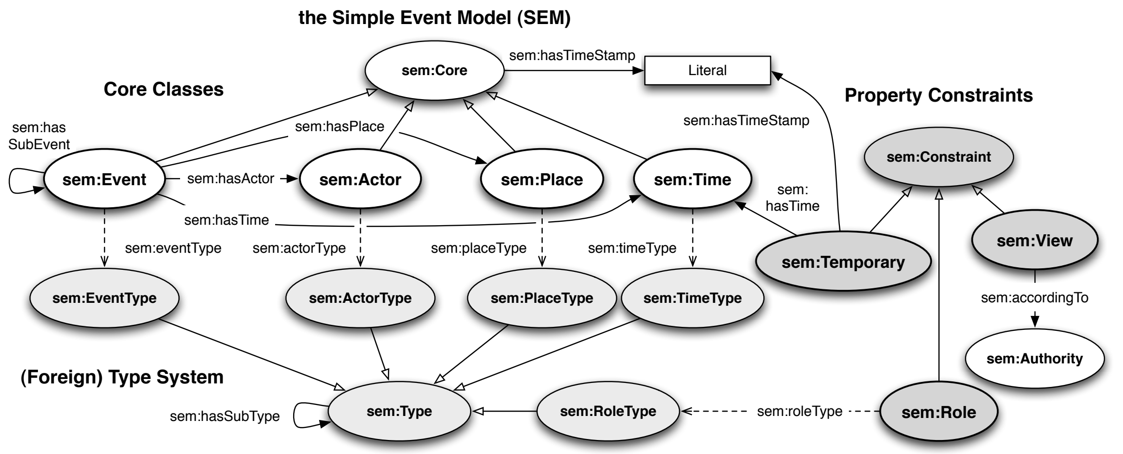


Figure 5. Simple Event Model. Ellipses represent classes, arrows represent properties between them and square boxes represent datatypes. Open-ended arrows represent subclass properties, dotted arrows are typing properties and the regular arrows' semantics are given by their labels.

4.1. *Simple event model*

The Simple Event Model (SEM) is introduced by van Hage *et al.* (2009) and provides a set of classes and properties to define events and their context. The segments are represented as instances of SEM events. SEM's classes are organized in three groups (see Fig. 5). SEM Core classes describe the *classic* parts of an event: *What is happening?* (sem:Event, e.g., anchoring, fishing), *Who participates?* (sem:Actor, e.g., a vessel), *Where?* (sem:Place, e.g., in Rotterdam), and *When?* (sem:Time, e.g., during year 2008). SEM Types can be linked to each of the Core instances, which typically come from existing domain-specific vocabularies. For example, sem:ActorTypes can be selected from vessel classifications[1]. GeoNames can be used for the sem:Places and sem:PlaceTypes. SEM Constraints express Temporary, Authoritative and Role constraints on the validity of properties between SEM Classes. Time and Place can have symbolic (URIs) or concrete values, to fit the representation of most data on the web. It is possible to use complex time representation such as the ISO 8601 standard and the TIMEX3 format for representing time interval, dates and reoccurring time expressions (e.g., 'two times a week'). All the classes and properties of SEM are optional and duplicable: SEM can integrate the partial information provided by different sources as different aspects of an event.

SEM is based on and informally linked to other event models such as the Event Ontology by Raimond and Abdallah (2007), LODE by Shaw *et al.* (2009) and the F model by Scherp *et al.* (2009). The main differences with them are (1) loosened constraints to be more flexible with the unpredictable syntax and inherent messiness of web data and (2) a linking to domain ontologies via sem:Types instead of a direct usage of other model's classes, making us more independent from changes in vocabularies we can not control.

4.2. *Moving object knowledge base*

The instances of SEM are either segment events (see Section 3.2), or events extracted and converted from the web. They are simply extracted if they are exposed as RDF (e.g., DBpedia and GeoNames), and converted to RDF if they are exposed as HTML

[1]Such as vessel types defined in the AIS protocol (see ITU (2001)) or on webpages like http://vesseltracker.com

(e.g., VesselTracker). The former are accessed from the MySQL database via the D2RQ platform[1] and the latter two are accessed directly with the SWI-Prolog[2] based triple store. Rules defining ship types and ship behavior enrich them. The rules are written in SWI-Prolog and the triple store is a knowledge base that enables us to link the different parts of an event represented in SEM with semantic web and spatial reasoning. In this way, rich movement patterns, vessel behavior and hypotheses can be defined and checked against trajectory data. Various web sources provide us with interesting information, but they can have different names for the same concepts. We align them to a single local identifier using the SKOS vocabularys skos:{exact|close|broad|narrow|related}Match properties as defined by Miles and Bechhofer (2009). The current knowledge base consists of two areas in front of the Dutch coast: a circular area of $5,000\text{km}^2$ around Rotterdam harbor and a circular area of $1,250\text{km}^2$ around the Dutch island Texel. In total about 600 vessels are tracked for a single week, resulting in 92,000 segments with about 30 attributes, given by AIS or looked up from the web. We have annotated these areas with 225 new GeoNames features, such as harbors to be able to reason with geographic locations. In Figure 8 we give an example of how different attributes of a single SEM event that were collected from the web can be used together, in a Prolog predicate that defines the notion of a *ship arriving at an appropriate harbor*. Figure 9 shows another example of a rule: *a ship banned from European waters close to a European harbor*. The vessel behavior for both of these examples cannot be derived from the trajectory itself, and would not be obvious to a human operator, but they are nonetheless important events to be brought to his attention while monitoring vessels. In section 6 we explain these use cases in more detail.

In this set up, SEM is the central piece for aggregating knowledge and resources at different levels of abstraction into event representations. These events can be used in complex, multi-step rules defining ship behavior. The behavior examples that have been described here are taken from the MSS domain, but they can be related to more generic movement patterns, like the ones defined by Dodge *et al.* (2008). For example, the successive segments can be classified as a Spatio-Temporal sequence: 'an ordered subsequence of locations with their timestamps'. Daily movements of migrating birds belong to the same generic pattern description. Complex events are multidimensional data, aggregating low-level tracks and semantic information. They can be visually explored by humans, to find computationally expensive patterns relevant to a certain domain or situation and to check detected anomalies. A visualization tool is needed to take advantage of the richness of the knowledge and present it in a useful way to a human user.

5. Visualization with trajectory contingency tables

An interactive Trajectory Contingency Table (TCT) can be used to browse the knowledge base and is suitable for expert users, who want to discover relations between attributes as spatial patterns, such as the change over *time* for different *vessel types*. We do not know up front which attributes are available a in the knowledge base, therefore the visualization needs to be generic. The TCT can deal with the most common types of attributes, such as time, geographical locations, numbers, and derived predicates, like *ship arriving at an appropriate harbor* from the previous section. With the TCT we can show temporal and spatial patterns, or correlation between attributes in terms of these patterns.

[1]D2RQ platform: http://www4.wiwiss.fu-berlin.de/bizer/d2rq
[2]SWI-Prolog for the semantic web: http://www.swi-prolog.org/web

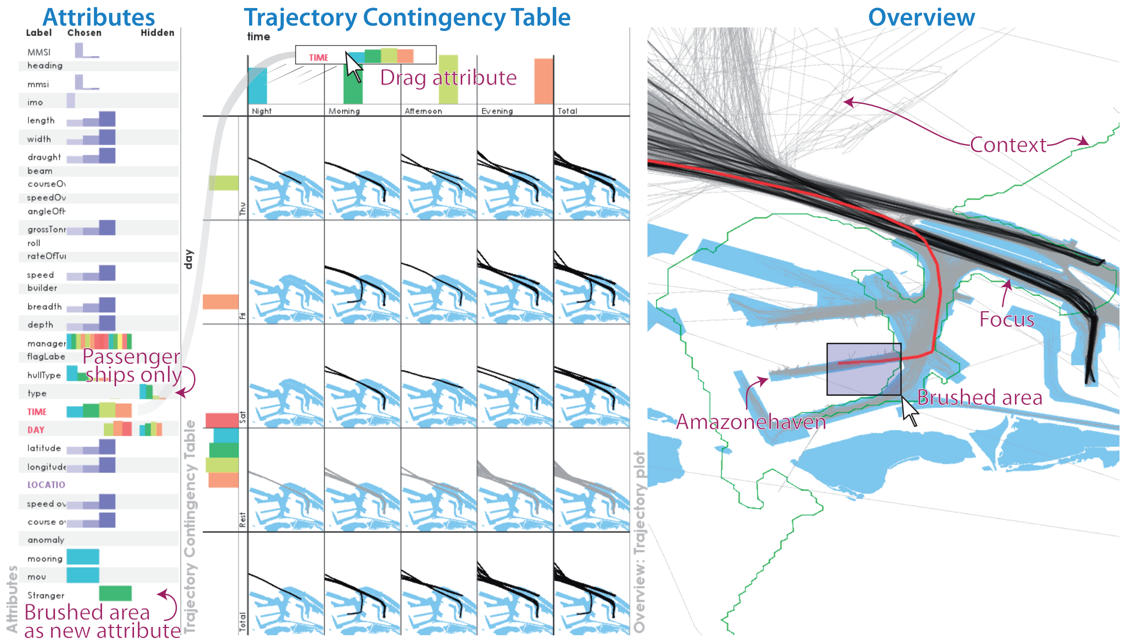


Figure 6. A Trajectory Contingency Table (TCT) with an Attribute view (left) and an Overview (right). All vessel trajectories in the knowledge base in the neighborhood of Rotterdam harbor during a single week are shown. In the Attributes view, attributes are divided in bins and listed with two histograms: chosen bins and hidden bins. Trajectories are annotated with a *vessel type* attribute obtained from the web, and only the trajectories of passenger ships are shown in the TCT and highlighted in the overview by choosing this bin. The TCT is displayed with *time* and *day* attributes on the axes, with only Thursday until Saturday selected from the day attribute. Each cell contains a map with parts of trajectories that satisfy the accompanying row and column labels. In the Overview, chosen trajectories (focus) are highlighted in dark gray on top of the context containing all data as thin, light gray lines and a map with a contour of The Netherlands in green and a solid shape for the harbors in the port of Rotterdam. By brushing we select areas to define new attributes. From the visualization we notice a strong pattern towards two mooring areas, given by the dark trajectories in the overview. A single trajectory on Friday morning shows possible anomalous behavior, by mooring in the 'Amazonehaven'. By means of reasoning with contextual information in Section 6.1, we can determine more precisely whether or not this is an anomalous behaving passenger ship.

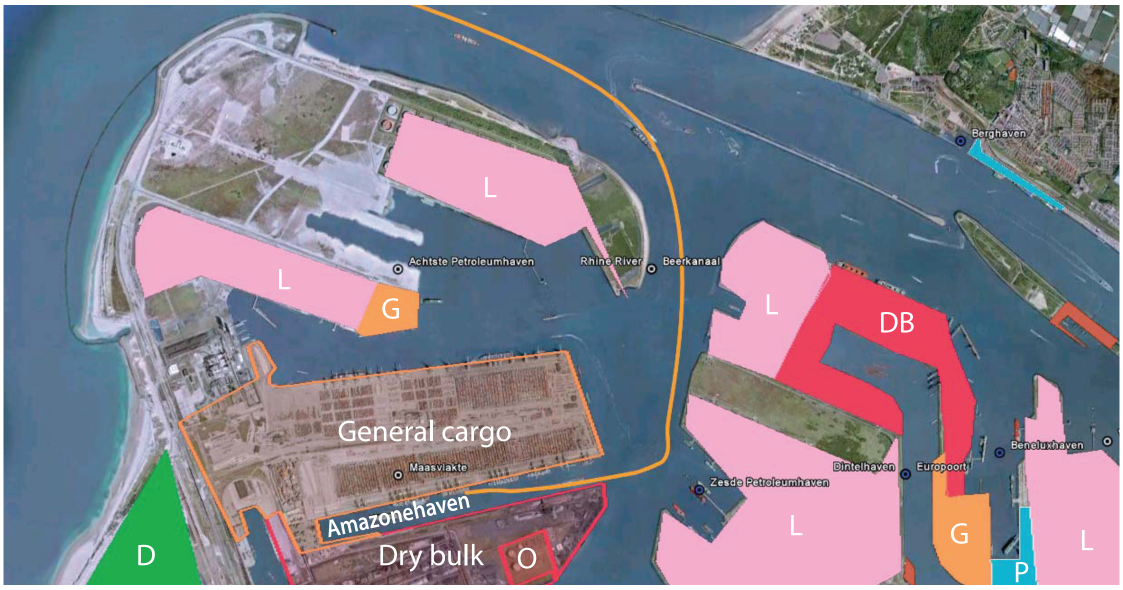


Figure 7. Map of Rotterdam harbor with a trajectory of a passenger ship created with Presto, GeoNames features as placemarks, and polygons with harbor types (L=Liquid bulk, D=Distribution, G=General Cargo, DB=Dry bulk, O=Other cargo, and P=Passenger).

The visualization retrieves trajectories by querying the knowledge base with SPARQL queries[1], which convert segments to trajectories. First, we determine for tuples $\boldsymbol{\alpha}_i$ which attributes are available, by searching for owl:DatatypeProperty properties. The data type of the attribute is specified by the accompanying rdfs:range property and is encoded with predefined XML schema types, such as xsd:integer and xsd:string. Then, for each vessel , a trajectory $\boldsymbol{\alpha}_0, \ldots, \boldsymbol{\alpha}_{N-1}$ is reconstructed by gathering all values of the queried attributes of the vessel's Actor and the Events related.

Pearson (1904) introduced a contingency table where two categorical or binned attributes are put on the axes of a table by assigning (groups of) outcomes to a row (or column). In a cell, the number of data items are shown that contain the outcomes of the corresponding row and column for the attributes assigned to the axes. Furthermore, each axis has an additional row that shows for each column the sum of the items. Becker *et al.* (1996) generalized the contingency table to a Trellis display, which is an arrangement of small multiples of basic graphics. Brunsdon (2001) showed a variant with maps as a scatterplot of locations and Carr *et al.* (2002) applied this technique with choropleth maps. Among others, Stolte *et al.* (2002) extended Trellis displays to a widely used tool, called Tableaux, to display various renderings in table cells, including graphs, scatterplots, and maps, and is often used for relatively small data sets, while our TCT is capable of visualizing large data sets, i.e., the complete knowledge base, interactively.

Instead of showing a number in the contingency table, we display a map with the parts of trajectories that satisfy the constraints of the cell (see Fig. 6) given by the corresponding row and column. For instance, by putting *time* and *speed* on the axes, we see how the different speed-related areas, such as stop places, change over time. Since the input data is large and consists of many attributes, the TCT is extended with two views for improving the navigation: Attributes and Overview (see Fig. 6). In the Attributes view on the left, all attributes are listed and divided in bins, which are shown as a histogram. The bins are selectable and filter the whole data set. The exploration in specific domains may be optimized by manually adjusted the bins of the attributes. By clicking on a bin, the whole data set is filtered on that bin. All other bins move to the histogram with hidden bins, to provide an overview of the current selection in a single column. The attributes can be dragged from the attributes view to the axes of the TCT to quickly investigate different settings. Alternatively from the trajectory plots in the cell, we can drag a time attribute to the cells to show a temporal distribution by means of clocks. We can show maps again by dragging a location attribute to the cells. In the overview on the right of the TCT, the selection of data made by choosing bins (focus) is highlighted on top of all data (context). In the overview it is possible to brush parts of the focused trajectories to define areas, which can be used to define a new boolean attribute stating whether or not a trajectory is contained in the area. This new attribute can be used in the TCT, for further investigation.

6. Use cases

6.1. *Abnormal mooring*

AIS messages contain some identifying information about the broadcasting ship: name, call sign, MMSI and a number assigned by the International Maritime Organization

[1]W3C SPARQL specification: http://w3.org/TR/rdf-sparql-query

```
 1 abnormal_mooring(Event) :-
 2         event(Event, wordnet:'synset-stop-verb-1'),
 3         event_place(Event, Place, _),
 4         space_within_range(Place, Harbor, 0.2),
 5         rdf(Harbor, geo:featureCode, geo:'H.HBR'),
 6         rdfs_individual_of(Harbor, HarborType),
 7         event_actor(Event, Ship, ShipType),
 8         % check if the harbor type is not appropriate for the ship
 9         \+apt_ship_harbor_type(ShipType, HarborType).
10
11 % liquid bulk harbors are appropriate for tanker, etc.
12 apt_ship_harbor_type(freebase:'en.tanker', ex:ptype_LiquidBulkHarbor).
13 apt_ship_harbor_type(freebase:'en.passenger_ship', ex:ptype_PassengerHarbor).
14 % ... further ship-type and harbor-type mappings
15
16 % add the property to the knowledge base
17 :- findall(Event, abnormal_mooring(Event), AbnormalMoorings),
18    forall(member(AM, AbnormalMoorings),
19            rdf_assert(AM, ex:mooringEventType, ex:etype_abnormal_mooring)).
```

Figure 8. Illustration of a Prolog test predicate for the abnormal mooring of a ship. This code makes use of the SEM Prolog API. The predicate is used to add an extra property to segments where an abnormal mooring takes place. This property can then be visualized.

(IMO); they are kept when building the segments. We use the latter two to automatically query the web[1] for additional information about the ship, that we convert semi-automatically to RDF. One piece of such information is a precise (set of) type(s) for the ship, such as Oil Tanker, Ro-Ro (i.e. ferryboat), Refrigerated Fish Carrier. We are interested in passenger ships and how they behave during the week, which is shown in the TCT by means of a spatio-temporal distribution. We notice a strange movement on Friday morning. This is an inserted trajectory of a passenger ship simulated with Presto that anchors at the general cargo harbor 'Amazonehaven', whereas passenger ships should stop at passenger terminals. To verify whether this movement is anomalous, we reason further with contextual information from other sources.

The Port of Rotterdam Authority has published maps[2] with a classification of their different harbors (e.g. liquid bulk, food), which we have converted to Google Earth KML (see Fig. 7). We used the SWI-Prolog Space package to convert the KML placemarks with polygon shapes into RDF with GeoRSS polygon literals. The SWI-Prolog Semantic web package can then be used to enrich events related to this data.

To define *abnormal* mooring, we created a Prolog rule that works in three steps. The first step describes the event using the SEM API[3]. The API provides a generic interface to create and query events, abstracting from the level of their RDF implementation details. This API, and our choice of implementation for the spatial indexing, represent the main difference between our platform and that of Aasman (2009): the conversion between the most popular spatial and temporal formats and a unified representation is achieved automatically by our API, while the graph manipulation of SEM events is simplified for the users. Lines 1 to 9 of Figure 8 show the test predicate that queries for the types of the ships participating in an event as actors (line 7), and checks out whether they match with the type of the harbor (line 5), which is the place of the same event (line

[1]Such as http://vesseltracker.com and http://e-ships.net
[2]Port of Rotterdam: http://portofrotterdam.com/en/about_port/port_maps/branches
[3]SEM API: http://eculture.cs.vu.nl/git/poseidon/sem.git

```
banned_ship_observation(Event, Ship) :-
        event_actor(Event, Ship, freebase:'en.ship'), % ship observed
        rdf(Ship, ais:imo, literal(IMO_Number)), % ship has IMO number
        rdfs_individual_of(BannedShip, parismou:banned), % a banned ship
        rdf(BannedShip, ais:imo, literal(IMO_Number)), % same IMO number
        event_place(Event, Place, _),
        space_within_range(Place, Harbor, 0.2),
        rdf(Harbor, geo:featureCode, geo:'H.HBR'), % observed near a harbor
        % has GeoNames ID for Europe as a parent feature
        rdf_reachable(Harbor, geo:parentFeature, geoinst:'6255148/').

:- findall(Event, banned_ship_observation(Event, _), BannedShipObservations),
   forall(member(BSO, BannedShipObservations),
          rdf_assert(BSO, ex:mouEventType, ex:etype_banned)).
```

Figure 9. Illustration of a Prolog test predicate for detecting ship banned by the Paris MOU near a European harbor. The predicate is used to add an extra property to segments of trajectories where a banned ship is observed in violation of this ban. This property can then be visualized.

3): stop (line 2). Step two checks out the matches between the types: they match if they are listed as pairs with the predicate `apt_ship_harbor_type` (line 12-13). This predicate defines the *normal* pairs: a Crude Oil Tanker should anchor at a liquid bulk anchorage, a Passenger Ship should anchor at a passenger terminal. Internally, the SEM API uses the SKOS alignments to find additional types for the ship and equivalences between them. The third step adds new triples to the knowledge base: ex:abnormal_mooring as a type for the instances of ship that satisfy the predicate abnormal_mooring (line 1).

This rule uses GeoNames for the URIs of geographic places, their types (harbor, populated place, etc.) and their geographic coordinates (line 5). GeoNames consists of user contributed content, and we extended it with extensive information for the Dutch coast and the port of Rotterdam, including unique identifiers for all the harbors in Rotterdam. The rule also uses ship trajectories (segments, they are the instances queried when instantiating the sem:Event, line 1), ship types from the web (Freebase, line 12-13), harbor types from the Port of Rotterdam website, spatial reasoning (`space_within_range` predicate, line 4), and RDF reasoning (`findall` predicate, line 17). The last predicate annotates segments with ex:abnormal_mooring type, which can be used for visualization.

The reasoning for the abnormal mooring of our simulated passenger ship is then as follows: The ship is of type ex:atype_passenger_vessel, which is mapped to freebase:'en.passenger_ship' with the skos:closeMatch property, and the sem:Place of the sem:Event of type wordnet:stop-verb-1 is near to the 'Amazonehaven', which has type ex:ptype_PassengerHarbor. The rule classified the stop of this vessel as an abnormal mooring.

6.2. *Banned vessels*

The previous rule is an example of an anomaly detection by means of a classification of abnormal behavior for trajectories, given a domain-specific definition. Anomaly detection is crucial in an MSS system, as it enables a user to focus only on the subset of trajectories that are not behaving according to what is defined as normal patterns. Another example of such an abnormal pattern, combining ship movement trajectory, ship type and web-based information is the detection of ships currently undergoing a ban from the European waters. The Paris Memorandum of Understanding on Port State Control (ParisMOU, http://www.parismou.org) maintains a list of IMO identifiers of ships currently or historically under the EU ban. Fetching the data and converting it to RDF enables our system (see Fig. 9) to detect ship trajectories corresponding to banned ships that would come near a given harbor, in a similar fashion as Figure 8. Using type hierarchies defined in GeoNames, we can extend this rule to a harbor from any country for which the parent feature is Europe (Fig. 9, line 8), and add a property that is valid on the European Union level.

7. System design

Our architecture is one of the possible configurations for composing a system that enables a user to detect patterns in moving objects at three different levels: data, knowledge, and graphical interface. The interfaces of the architecture given in Figure 1 contain the preprocessing stage of the trajectories (like NMEA), which is domain specific, but can be replaced by other systems. The system's rationale is that if we convert data from different sources to a format compatible with an ontology in RDF, the data can be matched on the semantic level with other data and knowledge sources, which enhances a visual analytics system without further adaptation. Testing hypotheses based on attributes not available in the knowledge base is possible by adding data on demand from the web or by reasoning.

Our methodology is illustrated with vessel traffic, but no major domain-specific assumptions have been made, and as a result we expect our approach to be applicable in other domains. However, the rules will be domain dependent, as our examples show. The various approaches and tools combined in the methodology each have certain limitations. Presto can only be used for generating trajectories on a geographical map; Trajectories for mice running in a cage for lab experiments or a cursor trail on a computer screen should be generated in other coordinate systems. In PLS, topological changes, such as self-intersections, may occur while generalizing a trajectory, hence the trajectory is required to be smooth. However, applying PLS is not required. In the knowledge base, when using external vocabularies, it may be possible to run into inconsistencies and trust issues. Knowledge aggregation and trust management are currently ongoing research points in the project. In the TCT, we may need to adopt the bins to get more sophisticated results in specific cases. Nevertheless, our architecture shows an interesting combination of geographic and semantic reasoning, coupled with a multidimensional visualization tool to address many requirements for analyzing moving objects.

8. Conclusion and future works

We have presented an integrated approach for analyzing moving object data. The approach includes trajectory generation using *Presto*, trajectory compression using *Piecewise Linear Segmentation*, trajectory modeling with the *Simple Event Model*, which is the data model of our knowledge base. The trajectories's multiple dimensions, including the additional knowledge created by *reasoning*, can be visualized in a *Trajectory Contingency Table*. This approach allows us to visually detect patterns in trajectory data by means of exploring the trajectory's attributes and additional attributes obtained by various web sources and reasoning. New sources can be added easily which allows us to consult existing knowledge on demand for testing hypotheses containing attributes not available in the current knowledge base.

In future research we will include high-level abstractions of trajectories in SEM: behavior descriptions defined across domains and movement definition based on other movement data sources than AIS data (e.g. ferry logs from harbor websites). Furthermore, we would like to efficiently describe and reason with the relative position of moving ships and movements of multiple entities. For interoperability with existing GIS systems we may adopt our interfaces to OGC standards like Web Feature Service. Finally, we will investigate whether our approach can be used in other domains.

Acknowledgements

We thank the reviewers, Jack van Wijk and Huub van de Wetering of TU Eindhoven, Guus Schreiber of VU Amsterdam, Pieter Adriaans and Maarten van Someren of University of Amsterdam, and Eric Postma of Tilburg University for their fruitful comments for improvements of this publication. Furthermore, we thank Michael Borth of the Embedded Systems Institute and, Hans Hiemstra and Jacek Skowronek of Thales Nederland for their contributions in the early design of the system's architecture. This work has been carried out as a part of the Poseidon project with Thales Nederland under the responsibilities of the Embedded Systems Institute. This project is partially supported by the Dutch Ministry of Economic Affairs under the BSIK program.

References

Aasman, J., 2009. Geospatial and moving objects with RDF and allegroGraph. *In*: *International semantic web conference (ISWC2009)*, October.

Andrienko, G., *et al.*, 2009. Interactive visual clustering of large collections of trajectories. *In*: *IEEE visual analytics science and technology*, 3–10.

Andrienko, N. and Andrienko, G., 2007. Designing visual analytics methods for massive collections of movement data. *Cartographica*, 42 (2), 117–138.

Bak, P., *et al.*, 2009. Spatiotemporal analysis of sensor logs using growth ring maps. *IEEE transactions on visualization and computer graphics*, 15 (6), 913–920.

Becker, R.A., Cleveland, W.S., and Shyu, M.J., 1996. The visual design and control of trellis display. *Journal of computational and statistical graphics*, 5, 123–155.

Bertolotto, M., *et al.*, 2007. Towards a framework for mining and analysing spatio-temporal datasets. *Int. journal of geographical information science*, 21 (8), 895–906.

Brunsdon, C., 2001. The comap: exploring spatial pattern via conditional distributions. *Computers, environment and urban systems*, 25 (1), 53–68.

Buccella, A., Cechich, A., and Fillottrani, P., 2009. Ontology-driven geographic information integration: A survey of current approaches. *Comp. & geosciences*, 35 (4).

Cao, H., Wolfson, O., and Trajcevski, G., 2006. Spatio-temporal data reduction with deterministic error bounds. *The VLDB Journal*, 15 (3), 211–228.

Carr, D.B., *et al.*, 2002. Interactive linked micromap plots and dynamically conditioned choropleth maps. *In*: *National conference on digital government research*, 1–7.

Chen, B., *et al.*, 2010. An approach for heterogeneous and loosely coupled geospatial data distributed computing. *Computers & geosciences*, 36 (7).

Dodge, S., Weibel, R., and Lautenschütz, A.K., 2008. Towards a taxonomy of movement patterns. *Information visualization*, 7 (3–4), 240–252.

Douglas, D.H. and Peucker, T.K., 1973. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica*, 10 (2).

Embedded Systems Institute, 2007. The Poseidon project. [online] Available from: http://www.esi.nl/poseidon [Accessed February 2010].

Gotz, D., Zhou, M.X., and Aggarwal, V., 2006. Interactive visual synthesis of analytic knowledge. *In*: *IEEE visual analytics science and technology*, 51–58.

Gudmundsson, J., *et al.*, 2009. Compressing spatio-temporal trajectories. *Computational geometry*, 42 (9), 825–841.

Hägerstrand, T., 1952. *Innovationsförloppet ur korologisk kynpunkt.* Lund, Gleerup Translation in English: Allan Pred, Innovation diffusion as spatial process.

Hiebel, G., Hanke, K., and Hayek, I., 2010. Methodology for CIDOC CRM based data integration with spatial data. *In*: *Comp. appl. and quantitative methods in arch.*

Hornsby, K.S. and Joshi, K., 2010. Combining ontologies to automatically generate temporal perspectives of geospatial domains. *GeoInformatica*, 14 (4), 481–505.

ITU, 2001. International Telecommunication Union: Technical characteristics for an automatic identification system using time division multiple access in the VHF maritime mobile band. *Recommendation ITU-R M.1371-1.*

Janssens, J., Hiemstra, H., and Postma, E., 2010. Creating artificial vessel trajectories with Presto. *In*: *Benelux conference on artificial intelligence (BNAIC 2010).*

Miles, A. and Bechhofer, S., 2009. SKOS Simple knowledge organization system reference. [online] Available from: http://w3.org/TR/skos-reference/ [Accessed April 2010].

Orellana, D. and Renso, C., 2010. Developing an interactions ontology for characterising pedestrian movement behavior. *In*: M. Wachowicz, ed. *Movement-aware applications for sustainable mobility: Technologies and approaches.*, 62–86.

Pearson, K., 1904. On the theory of contingency and its relation to association and normal correlation. *Drapers' Company Research Memoirs. Biometric Series I.*

Raimond, Y. and Abdallah, S., 2007. The event ontology. [online] Http://purl.org/NET/c4dm/event.owl [Accessed April 2010].

Sánchez, A., *et al.*, 2009. A context model and reasoning system to improve object tracking in complex scenarios. *Expert systems with applications*, 36 (8), 10995–11005.

Scherp, A., *et al.*, 2009. F—A model of events based on the foundational ontology DOLCE+DnS ultralight. *In*: *Conference on knowledge capturing (K-CAP).*

Shaw, R., Troncy, R., and Hardman, L., 2009. LODE: linking open descriptions of events. *In*: *4th annual Asian semantic web conference*, 153–167.

Spaccapietra, S., *et al.*, 2008. A conceptual view on trajectories. *Data & knowledge engineering*, 65 (1), 126–146.

Stolte, C., Tang, D., and Hanrahan, P., 2002. Polaris: A system for query, analysis, and visualization of multidimensional relational databases. *IEEE transactions on visualization and computer graphics*, 8 (1), 52–65.

Thomas, J. and Kielman, J., 2009. Challenges for visual analytics. *Information visualization*, 8 (4), 309–314.

van Hage, W.R., *et al.*, 2009. Combining ship trajectories and semantics with the Simple Event Model (SEM). *In*: *ACM workshop on events in multimedia (EiMM)*, 73–80.

Willems, N., van de Wetering, H., and van Wijk, J.J., 2009. Visualization of vessel movements. *Computer graphics forum*, 28 (3), 959–966.

Wood, J., *et al.*, 2007. Interactive visual exploration of a large spatio-temporal dataset: reflections on a geovisualization mashup. *IEEE transactions on visualization and computer graphics*, 13 (6), 1176–1183.

Xiao, L., Gerth, J., and Hanrahan, P., 2006. Enhancing visual analysis of network traffic using a knowledge representation. *In*: *IEEE visual analytics science and technology.*

Yan, Z., *et al.*, 2008. Trajectory ontologies and queries. *Trans. in GIS*, 12 (Sup. 1), 75–91.