Generating Semantic Trajectories Using a Car Signal Ontology

Conference Paper · April 2018

DOI: 10.1145/3184558.3186962

4 authors:

Benjamin Klotz
Institut Mines-Télécom

Raphaël Troncy
Institut Mines-Télécom

Daniel Wilms
BMW

Christian Bonnet
Institut Mines-Télécom

Generating Semantic Trajectories Using a Car Signal Ontology

Benjamin Klotz
EURECOM & BMW Research, New Technologies, Innovations
Sophia Antipolis, France
klotz@eurecom.fr

Raphaël Troncy
EURECOM
Sophia Antipolis, France
troncy@eurecom.fr

Daniel Wilms
BMW Research, New Technologies, Innovations
Munich, Germany
daniel.dw.wilms@bmw.de

Christian Bonnet
EURECOM
Sophia Antipolis, France
bonnet@eurecom.fr

ABSTRACT

In this paper, we use semantic technologies for enriching trajectory data in the automotive industry for offline analysis. We proposed to re-use a combination of existing ontologies and we designed a Vehicle Signal Specification ontology to provide an environment in which we developed an application that analyzes the variations of signal values and enables to infer the "driving smoothness" that we represent as additional annotations of semantic trajectories.

CCS CONCEPTS

• **Information systems** → *Geographic information systems*; *Sensor networks*;

KEYWORDS

Car Signal ; Ontology ; Semantic Trajectories ; VSS ; SOSA ; SSN

ACM Reference Format:
Benjamin Klotz, Raphaël Troncy, Daniel Wilms, and Christian Bonnet. 2018. Generating Semantic Trajectories Using a Car Signal Ontology. In *WWW '18 Companion: The 2018 Web Conference Companion, April 23–27, 2018, Lyon, France.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3184558.3186962

1 INTRODUCTION

Current and future automotive innovations are based on the interconnection of systems such as the vehicle, infrastructure back-ends and external data sources. Taking autonomous driving as an example, it has to rely on an intelligent and dynamic interconnection of the vehicle's data with knowledge about its environment despite the diversity of sources, formats, and sensors involved. To tackle the challenge of data integration and reuse in vehicular intelligent interconnections, we use semantic technologies. Already largely used on the web, especially by search engines which promote the schema.org vocabulary[1], semantic technologies are also more and more used to integrate physical devices in the Internet of Things[2] and in the automotive industry[3]. Combining ontologies may enable queries about complex driving contexts, including sensor values, location, time and external data.

For simplification, we consider that there are two types of data[4] that can be linked to a vehicle: static and dynamic. Static data corresponds to the car's attributes such as its model, number of wheels, dimensions and the list of embedded sensors. Dynamic data is what car sensors produce on a continuous basis. It depends both on time and space. In addition, embedded sensors only produce dynamic data if they are instantiated in the static car attributes.

A number of ontologies can describe cars' attributes and configuration. This includes auto.schema.org, still in development and based on four existing ontologies:

- Car option ontology[5] for the commercial aspects of offers for sale or rental. It contains 12 classes and 19 properties.
- Vehicle sales ontology[6] (VSO) for describing cars, boats, bikes, and other vehicles for e-commerce with 33 classes and 54 properties.
- Used cars ontology[7] for describing aspects of used cars for e-commerce with 22 classes and 46 properties.
- Volkswagen Vehicle Ontology[8] for describing Volkswagen-specific features of automobiles with 30 classes and 50 properties. Its interest is limited to the domain of the e-commerce for one brand.

Other ontologies can describe trajectories, and therefore dynamic spatio-temporal data:

- datAcron[9] for describing semantic trajectories as succession of sub-trajectories associated with points or regions, mostly associated with the aircraft domain [8]. It contains 552 classes and 273 properties.

[1]http://schema.org
[2]http://iot.schema.org
[3]http://auto.schema.org
[4]www.automotive-ontology.org
[5]http://semanticweb.org/wiki/Car_Options_Ontology.html
[6]http://www.heppnetz.de/ontologies/vso/ns
[7]http://ontologies.makolab.com/uco/ns.html
[8]http://www.volkswagen.co.uk/vocabularies/vvo/ns
[9]http://ai-group.ds.unipi.gr/datacron_ontology/

WWW '18 Companion, April 23–27, 2018, Lyon, France

ACM ISBN 978-1-4503-5640-4/18/04.
https://doi.org/10.1145/3184558.3186962

- *Baquara*[2] for describing semantic trajectories as succession of episodes associated with points or regions, mostly associated with the automotive and tourism domain [6]. It has more than 100 classes and 200 properties but is not available online.
- STEP[10] (Semantic Trajectory EPisodes) for describing semantic trajectories with different levels of granularity and attach labels to them [12]. It contains 21 classes and 19 properties. It was developed using ontology design pattern [7] for semantic trajectories [9] in order to be combined with domain ontologies.
- Movement Behavior Interpretation is a work aiming at semantically add labels in the domain of tourism to moving people [2]. The ontology is also not available online.

Some other initiatives tackle specific challenges, especially in the domain of ADAS (Advanced Driving Assistance Systems) and context modeling. The Toyota TTI Core ontologies[11] describe vehicle sensors, controls and roads with a core set of signals and configurations used in ADAS for future autonomous vehicles [15]. In the domain of context-aware services, some work have resulted in ontologies for some essential sensors and entities interacting with the vehicle [1, 11]. DFKI developed an automotive vocabulary in order to represent knowledge inside a vehicle and exchange it with other vehicles [5]. It is contextual with sensors, detected events, as well as user-related with preferences and observations of behavior. The Ontology for Context Modeling (OCM) is focusing on inferences about a driving context in persuasive computing environments [14].

A missing aspect from these initiatives is the representation of a complete car sensor and signal ontology. This gap limits innovations to either automotive domain experts or to a restricted set of well-known signals and sensors. A second missing aspect is the representation of semantic trajectories enriched with car attributes and sensor data for non-specific applications.

In this paper, we aim to answer to the research question: *How should we best combine both static and dynamic car data in order to annotate semantic trajectories and link that to external knowledge?*. We will focus on two main use cases: generate segments of trajectory annotated according to the evolution of a given signal value, and a "smooth" driving percentage label attached to a trajectory when longitudinal and angular acceleration are bound.

Our research is divided into two directions for solving the challenge of data flow access and querying using RDF. In the first case, we only store a sliding window of observations, adding and removing RDF triples on the fly and aim for stream reasoning [4], while in the second case, we never remove triples. We focus in this paper on the second case since we are interested in having access to history in our queries.

The remaining of this paper is organized as follows. In Section 2, we introduce the so-called Vehicle Signal Specification ontology (VSSo). In Section 3, we present the demonstrated application combining VSSo with SOSA/SSN and STEP. In Section 4, we compare this combination of ontologies with other initiatives for evaluating its interest and highlight the usage it enables.

[10]https://talespaiva.github.io/step/
[11]http://www.toyota-ti.ac.jp/Lab/Denshi/COIN/Ontology/TTICore-0.03/

2 DESIGN OF THE VSS ONTOLOGY

In order to represent observation of car signals, we adopt the new SOSA/SSN ontology[12] which is a joint W3C and OGC recommendation [10]. We need to describe car signals and car sensors. Even though there have been some projects such as [13] proposing automotive ontologies, they only partially cover the domain (e.g. assistance and diagnostic of vehicles). The W3C Automotive Working Group[13] intends to develop Open Web Platform specifications for exposing vehicle signal information and uses the Vehicle Signal Specification[14] (VSS) from GENIVI as signal vocabulary. The VSS is a common naming space to decouple the vehicle electrical network from its original representation to exchange data with third parties. It contains an extensive set of vehicle parts and signals, defined by a name, comment, unit and format.

First, we complement the VSS by defining sensor entries for signals. This is not included in the current specification. Then, in order to create RDF triples representing the car *Attributes*, *Signals*, *Sensors/Actuators*, *Branches* and *Units* from VSS, we use a converter from JSON to turtle based on an existing work [3] and the rdf-ext JS library. The generated VSSo ontology[15] contains 311 classes and 6 properties and uses OWL-Lite for its restrictions on sensors and units. Classes are either `Branches` organized in a tree structure with the property `vss:partOf`, or `Signals`.

This automatic generation is not enough. Several signals share the same name (e.g. "Switch") while representing different concepts. Many signals are observable but not actuable, while we define `vss:ObservableSignal` and `vss:ActuableSignal` as subclasses of `sosa:ObservableProperty` and `sosa:ActuableProperty`. Some are not produced nor consumed by a sensor/actuator (e.g attached to the infotainment system): they will be defined with a `vss:VirtualSensor` to be compliant with the SOSA pattern. For the rest of them, sensors and actuators are technology-independent intermediate objects. The branch "Vehicle" describes terms related to the general vehicle and is moved on to of all other branches. The position of branches (e.g. Mirror.Right.Tilt) is defined as a property `vss:hasPosition` of certain branches.

Listing 1 is an extract from VSSo describing `vss:TravelledDistance`, a signal measured by a `vss:Odometer` with the unit `unit:Kilometer`.

Listing 1: VSSo sample: TraveledDistance signal

```
vss:TravelledDistance a rdfs:Class, owl:Class;
  rdfs:label "TravelledDistance"@en;
  rdfs:comment "Signal.Drivetrain.Transmission.Travelled-
     Distance. Odometer reading."@en;
  rdfs:subClassOf vss:Signal;
  rdfs:subClassOf
   [ a owl:Restriction;
      owl:onProperty sosa:isObservedBy;
      owl:allValuesFrom vss:Odometer.],
   [ a owl:Restriction;
      owl:onProperty qudt-1-1:unit;
      owl:allValuesFrom qudt-unit-1-1:Kilometer.].
```

[12]https://www.w3.org/TR/vocab-ssn/
[13]https://www.w3.org/auto/wg/
[14]https://github.com/GENIVI/vehicle_signal_specification/
[15]https://github.com/klotzbenjamin/VSSontology

By linking SOSA/SSN to VSSo, we generate RDF triples of observations of vehicle signals and may attach them to a spatiotemporal context using the STEP ontology.

3 DEMONSTRATION

In this demonstration, one will control a simulated vehicle, and will observe signal values that would be automatically transformed into RDF to enrich a semantic trajectory. This is used to infer labels about the driver attached to episodes of the trajectory. Labels will also be visualized: variations of signal values in order to infer a "smooth" driving label.

We developed a python Flask[16] server combining SOSA/SSN for *observations*, VSS for the car signal domain and STEP for the trajectory description. It uses a vehicle configuration file containing a list of known signals as well as URIs to access their values. The data source can be real vehicle data in the backend where we access a subset of pre-selected signals, or a vehicle simulator. In this demonstration, we use the OpenXC vehicle simulator[17] which runs on a local server, provides a control interface (pedals, steering wheel, gear...), and generates a JSON file of signal values with a frequency of 100Hz. The following steps are handled by our application:

Add attributes. It reads the vehicle configuration file and generates a graph representing the static vehicle as it is done in the e-commerce domain extended with annotations about the signal and its unit. The vehicle configuration file is parsed to extract a list of known signals. A blank node is created to represent the vehicle, and for each signal, the VSS is queried for the signal's unit, label and dedicated sensor. This query is done by using triple pattern fragments[18] which minimize server processing in comparison to SPARQL. For each signal, the sensor is also a `vso:Feature` of the vehicle.

Listing 2: Car with a speed sensor as RDF triples

```
@prefix qudt: <http://www.qudt.org/1.1/schema/qudt\#> .
@prefix unit: <http://www.qudt.org/1.1/vocab/unit\#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema\#> .
@prefix sosa: <http://www.w3.org/ns/sosa/> .
@prefix vso: <http://www.purl.org/vso/ns\#> .
@prefix vss: <http://automotive.eurecom.fr/vsso#> .

vss:VehicleSpeed a sosa:ObservableProperty ;
  rdfs:label "Speed" ;
  qudt:Unit unit:KilometerPerHour .
[] a geo:Feature , vso:Automobile , sosa:FeatureOfInterest ;
  vso:feature [ a vss:Speedometer ,
      vso:FeatureValue ;
      sosa:observes vss:VehicleSpeed ].
```

Add observations. It reads signal values and extends the static graph with *observations* based on observed signal instances. For a given time span and frequency for a set of known signals, we request the source to retrieve values. URIs and paths are set in the configuration file, and the source is expected to be a JSON file. Then, an instance of type `sosa:Observation` is created and linked to its signal and sensor. The retrieved value is expressed using the known unit, as well as location and time of its *FixPoint*. Each instance of `sosa:Observation` has the car node as `sosa:FeatureOf-Interest`.

[16]flask.pocoo.org/
[17]https://github.com/openxc/openxcvehicle-simulator
[18]http://linkeddatafragments.org/

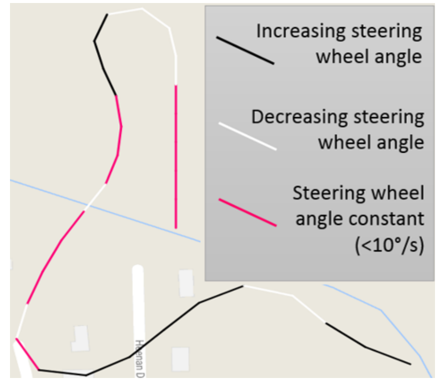

Figure 1: Steering wheel angle visualized with 3 color

Add trajectory. It creates a trajectory instance, composed of a raw trajectory (list of *FixPoints*), and generates labeled segments. First, an instance of `step:Trajectory` containing a `step:RawTrajectory` instance is created with the car node as `step:Agent`. Then, it is filled with spatiotemporal *FixPoints* when *Observations* are made.

Graph reduction. It extracts trajectories annotated with signal values by serializing the graph into a CSV file with the format: ($latitude, longitude, time, [values]$)

With a given set of signals, a SPARQL query retrieves a set of values: (latitude, longitude, time, [signals]). This is especially relevant for reusing existing trajectory mining algorithms that do not rely on linked data.

Segment labeling. It adds labels on segments based on simple rules.

From a graph containing observations, we define labels based on simple comparative rules and create `step:Episode` instances in the `step:Trajectory`. They can be qualitative, quantitative or both. For instance the "smooth" driving label is defined by a longitudinal and angular acceleration being within given bounds. All segments that fit this requirement are labeled "smooth", while the others are labeled "not smooth".

Plot fix points and segments. It displays observation points as markers, as well as colored segments based on simple comparative rules and using Google maps with Flask[19].

For instance, with a configuration file describing three signals: *VehicleSpeed*, *EngineSpeed* and *TravelledDistance*, it creates a graph with an instance of `vso:Automobile` having three sensors linked by the property `vso:Feature`. We observe the *Speed* signal and extract a trajectory annotated with *Speed* values.

We add a rule to label a segment between two observations, e.g. *Acceleration*, *Deceleration* or *ConstantSpeed* based on the difference of speed between the two observations and display the trajectory on a map with three colors depending on the segments labels.

[19]https://github.com/rochacbruno/Flask-GoogleMaps

Table 1: Comparison of the different initiatives in regard to the hypothesis: we can enable semantic trajectory enrichments with signal values for generic applications

Initiative	Automotive domain	Sensor coverage	Semantics	Trajectory enrichment	Generic
auto.schema.org and its sources	Yes	Limited	Yes	No	Yes
DatAcron	No	No	Yes	Yes	No
Baquara[2]	No	No	Yes	Yes	No
Movement Behavior	No	No	Yes	Yes	No
STEP	No	No	Yes	No	Yes
Toyota TTI Core	Yes	Limited	Yes	No	Yes
Context-aware services	Yes	Limited	Yes	Limited	No
DFKI	Yes	Limited	Limited	No	Yes
OCM	Yes	Limited	Limited	No	Limited
VSSo+SOSA/SSN+STEP	Yes	Yes	Yes	Yes	Yes

4 COMPARISON AND USAGE

Our hypothesis being that we can enable semantic trajectory enrichments with signal values for generic applications, there are many initiatives that solve partially the problem, but not completely as visible in Table 1. The representations of static car data focus on the e-commerce domain and do not describe dynamic data. The trajectory representations tend to be defined for specific applications and domains and neither cover static nor dynamic car data. The contextual representations of cars are an interesting trade-off between static and dynamic car data but have a limited interest in trajectory representation and work only for non-generic use case in ADAS.

With respect to our hypothesis, we see that the combination of VSSo, SOSA/SSN and STEP fills different gaps from the existing initiatives and fits the requirements to enable our use cases: it covers the automotive domain and its sensor vocabulary with semantics and trajectory enrichment for generic applications.

This new representation allows web developers to query and integrate car data with only conventional signal names. Therefore it enables the development of applications based on the correlation between what is inferred from car data and external information. For instance, the correlation between a *distraction* label, inferred from car sensors, and events happening around the vehicle, or between aggressiveness and traffic jams.

5 CONCLUSION

In this paper, we have seen how RDF can be used to represent attributes and signals for a car, and that this solution enables queries on the complete domain for semantic web developers and non domain experts. We will improve this demonstration with the future improvements of auto.schema.org on context-awareness. Future work will focus on working with online graphs generated on-the-fly and the possibility of interaction with the vehicle for smart applications.

REFERENCES

[1] A. Armand, D. Filliat, and J. IbaÃśez-Guzman. 2014. Ontology-based context awareness for driving assistance systems. In *IEEE Intelligent Vehicles Symposium Proceedings*. 227–233.

[2] Miriam Baglioni, Jose Macedo, Chiara Renso, Roberto Trasarti, and M Wachowicz. 2009. Towards Semantic Interpretation of Movement Behavior. In *12^{th} AGILE Conference, Advances in GIScience*. 271–288.

[3] Victor Charpenay. 2016. Build a OWL ontology from VSS. (2016). unpublished.

[4] Daniele Dell'Aglio, Emanuele Della Valle, Frank van Harmelen, and Abraham Bernstein. 2017. Stream Reasoning: a Survey and Outlook: A summary of ten years of research and a vision for the next decade. *Data Science Journal* 1, 1-2 (2017), 59–83.

[5] Michael Feld and Christian Müller. 2011. The Automotive Ontology: Managing Knowledge Inside the Vehicle and Sharing it Between Cars. In *3^{rd} International Conference on Automotive User Interfaces and Interactive Vehicular Applications*. Salzburg, Austria, 79–86.

[6] Renato Fileto, Cleto May, Chiara Renso, Nikos Pelekis, Douglas Klein, and Yannis Theodoridis. 2015. The Baquara2 knowledge-based framework for semantic enrichment and analysis of movement data. *Data & Knowledge Engineering* 98 (2015), 104–122.

[7] Aldo Gangemi. 2005. *Ontology Design Patterns for Semantic Web Content*. 262–276.

[8] Santipantakis Georgios, Vouros George, Doulkeridis Christos, Vlachou Akrivi, Andrienko Gennady, Andrienko Natalia, Cordero Jose Manuel, and Martinez Miguel Garcia. 2017. Specification of Semantic Trajectories and Data Transformations for Analytics: The datAcron Ontology. In *13^{th} International Conference on Semantics Systems*.

[9] Yingjie Hu, Krzysztof Janowicz, David Carral, Simon Scheider, Werner Kuhn, Gary Berg-Cross, Pascal Hitzler, Mike Dean, and Dave Kolas. 2013. A Geo-ontology Design Pattern for Semantic Trajectories. In *Spatial Information Theory*, Thora Tenbrink, John Stell, Antony Galton, and Zena Wood (Eds.). Springer International Publishing, Cham, 438–456.

[10] Krzysztof Janowicz, Simon Cox, Kerry Taylor, Danh Le Phuoc, Maxime Lefrançois, and Armin Haller. 2017. *Semantic Sensor Network Ontology*. W3C Recommendation. W3C.

[11] Mohcine Madkour and Abdelilah Maach. 2011. Ontology-based context modeling for vehicle context-aware services. 31 (2011).

[12] Tales Paiva Nogueira. 2017. *A Framework for Automatic Annotation of Semantic Trajectories*. Ph.D. Dissertation. Université Grenoble Alpes.

[13] Michele Ruta, Floriano Scioscia, Filippo Gramegna, and Eugenio Di Sciascio. 2010. A Mobile Knowledge-based System for On-Board Diagnostics and Car Driving Assistance. In *4^{th} International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM)*.

[14] Z. Xiong, V. V. Dixit, and S. T. Waller. 2016. The development of an Ontology for driving Context Modelling and reasoning. In *IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. 13–18.

[15] Lihua Zhao, Ryutaro Ichise, Seiichi Mita, and Yutaka Sasaki. 2015. Core Ontologies for Safe Autonomous Driving. In *14^{th} International Semantic Web Conference, Posters and Demos Track (ISWC)*.

ADDITIONAL MATERIAL

GitHub repository: https://github.com/klotzbenjamin/VSSontology
Screencast: https://youtu.be/LgsNrUNQJdk