Large Scale Multimodal Data Processing Middleware for Intelligent Transport Systems

Krispin Raich*
University of Applied Sciences Kufstein Tirol
University of Passau
Kufstein, Austria
Krispin.Raich@fh-kufstein.ac.at

Robert Kathrein
University of Applied Sciences Kufstein Tirol
University of Passau
Kufstein, Austria
Robert.Kathrein@fh-kufstein.ac.at

Mario Döller
University of Applied Sciences Kufstein Tirol
Kufstein, Austria
Mario.Doeller@fh-kufstein.ac.at

***Abstract*—Modern Intelligent Transport Systems (ITSs) are comprehensive applications that have to cope with a multitude of challenges while meeting strict service and security standards. A novel data-centric middleware that provides the foundation of such systems is presented in this paper. This middleware is designed for high scalability, fast data processing and multi-modality. To achieve these goals, an innovative spatial annotation (SpatialJSON) is utilised. SpatialJSON allows the representation of geometry, topology and traffic information in one dataset. Data processing is designed in such a manner that any schema or ontology can be used to express information. Further, common concerns of ITSs are addressed, such as authenticity of messages. The core task, however, is to ensure a quick exchange of evaluated information between the individual traffic participants.**

I. Introduction

Through the advancements in computing and vehicular communication technologies, new opportunities but also challenges have arisen. Especially in the field of Vehicular Ad-Hoc Networks (VANETs) communication, the issue of message authentication is an ongoing topic of research. There are numerous approaches [1]–[3] to tackle this challenge. However, as today no solution can guarantee message verification without requiring intense computational power or infrastructure while ensuring modern security and services requirements [4]. Another important function of an ITS is data collection, homogenization, processing and storage. In this paper a (traffic) data centric middleware is presented, that collects and processes information from vehicles and uses the gathered information to categorize and authenticate vehicles and their reported data according to their behavior. This information is than redistributed to other vehicles via data push. Vehicles thus provide data about themselves and their environment. This data is then collected and merged by this middleware. The goal is to gather data from multiple sources about a specific entity and verify the provided information. This verified data is than used to solve the previous mentioned verification issue. In order to be able to describe this middleware, several aspects of this a concept are discussed in this paper. Starting from the properties and security measurements of a Vehicular Ad-Hoc Network (VANET), novel data types in *SpatialJSON*, data in Intelligent Transport System (ITS), vehicle identity generation, creating a wide area data processing model, multimodal traffic management, data collection and distribution, data format and concluding with data processing with evidence evaluation.
This paper is segmented in three parts. In the first part underlying technologies are discussed, followed by the introduction of the middleware. Finally, a conclusion from the implementation of the Proof of Concept (POC) is drawn and some future work is put into perspective.

II. Related Work

This section will briefly cover the necessary underlying technologies regarding V2X communication, VANET security, SpatialJSON, and data in ITS.

A. Vehicular Ad-Hoc Network (VANET)

A VANET describes a temporary connection between a group of nearby vehicles or static infrastructure for interchanging (traffic related) data. To connect to a VANET, vehicles use an On-Board Unit (OBU). The communication between OBUs or vehicles is categorised as V2V. Road Side Units (RSUs) usually represent the immobile infrastructure of a VANET. Communication between OBU and RSU is categorised as V2I. Cellular technology is also used to establish connections to arbitrary networks (V2N). Recently, the inclusion of pedestrians in VANET was introduced (V2P). Collectively the communication from vehicle to something is called V2X. V2X communication is handled over cellular networks standards like LTE-V2X and 5G-V2X for long range communication [5] and DSRC (dedicated short range communication) [6], C-ITS (Cooperative Intelligent Transport Systems) [7] and ITS Connect [8] for short range communication. [9]

B. VANET security and message verification

The highly dynamic topology of a VANET creates a difficult scenario for secure messaging and member authentication. Typical attack vectors of a VANET are: [4]

- *Bogus messages*: Deliberate injection of false information. This attack can be executed in- or outside a network.
- *Message modification*: Deleting or modifying a message from other VANET members.
- *Sybil attack*: The act of creating multiple real or fake identities and sending messages using these malicious identities. Sybil attacks are usually hard to detect.
- *Denial of Service (DoS)*: Injecting a great volume of messages so that certain resources become unavailable to legitimate network members.

ISSN 2305-7254

- *Eavesdropping*: Collecting all accessible data to disclose private information.
- *Impersonate attack*: Sending messages that are signed with another vehicles identity.
- *Replay attack*: Transmit repeating data maliciously.
- *Black hole attack*: Not forwarding any information to other vehicles.
- *Grey hole attack*: Selectively forwarding information to other vehicles.
- *Location tracking*: Tracking other vehicles by analyzing the messages sent by the victim.

In contrast to these attack vectors, modern communication systems require a certain amount of security and privacy standards. For VANETs these are as follows: [4]

- *Authentication*: Authentication requires that messages must be authenticated before further processing takes place.
- *Integrity*: Integrity requires that messages are authenticated and immutable.
- *Non-repudiation*: Non-repudiation requires that an authority can doubtlessly disclose the identity of a network member.
- *Availability*: Critical services must be available at all times for legitimate network members.
- *Anonymity*: The real identity of vehicles is concealed and can only be disclosed by authorities.
- *Unlinkability*: Unlinkability requires that there is no link between a vehicle's real identity and its pseudonym and no link between the used pseudonyms.
- *Conditional Traceability*: The identity and position of legitimate VANET members should be protected and disclosed for malicious users.
- *Efficient Revocation*: It should be possible to revoke privileges fast and efficiently from a VANET member.
- *Location Privacy Preservation*: It should not be possible to track or identify legitimate members by an attacker.

A basic approach to address these challenges is to use a Public Key Infrastructure (PKI). However, due to the severe drawbacks of such an approach in a VANET environment, numerous alternatives have been developed. There are two main categories: Cryptographic based schemes and Trust based schemes. [4] Cryptographic schemes can be further categorised in group signature based, identity based, or hybrid schemes. In signature group based schemes, every member of a group can sign a message with the shared group key. However, the group management proofs to be difficult, due to the highly dynamic topology. Identity based attempts are similar to a PKI, but the public/private key pair is derived from the vehicles identifier, which removes the need for a centralised Certificate Authority (CA). The downside of this approach is the added verification delay due to the computational demanding cryptography. [4] Trust based schemes usually rely on complementary cryptographic methods like asymmetric encryption and certificates to authenticate vehicle trust scores. In such a system, this reputation score is build slowly and degraded quickly. Trust based schemes are advantages in detecting misbehavior (e.g. Black Hole attack) compared to sole cryptographic schemes. However, reliably determine the trust level or reputation of a vehicle is essential and often difficult. In [10] vehicles are grouped (e.g. by trustworthiness). The trust score of a vehicle is determined by its assigned group and the contributed reports of events. Each group has a default trust score between 0 and 1. To gain trust, vehicles must broadcast information about an event to other vehicles. These reports or evidence e are then examined by receiving vehicles. Raya, Papadimitratos, Gligor, *et al.* [10] describes, inter alia, a Bayesian inference function which is (in the authors opinion) the best approach for event evaluation. The probability P for an event (α_i) for given evidence $e = \{e_1^j, e_2^j, \ldots, e_K^j\}$; with bayesian inference is described as follows:

$$P[\alpha_i|e] \;=\; \frac{P[\alpha_i] \prod_{k=1}^{K} P[e_k^j|\alpha_i]}{\sum_{h=1}^{I} (P[\alpha_h] \prod_{k=1}^{K} P[e_k^j|\alpha_h])}$$

The result is then used to update the trust score. According to [10], bayesian inference proofs to be superior in networks with low uncertainties. Currently best solution is a combination of a cryptographic based and a trust based scheme. However, a solution that meets all security and service requirements does not exist to date [4].

C. SpatialJSON

In order to model multimodal traffic paths, a recently developed data model is utilised. *SpatialJSON* is an extension of GeoJSON[11] that provides two new data constructs: *corridor* and *area* - collectively called *Spatial Entity* [12]. These Spatial Entities (SEs) enable an efficient modelling of three- and two-dimensional geometries that are used in this middleware to define traffic paths and areas. Additionally, this data-model is well suited for information storage and retrieval in big data applications. *Corridors* and *areas* are described as follows:

$$corridor = < id, shape, coordinates_{\mathrm{i}} >, \;\; i \in \mathbb{N}$$

Where the *id* is the identifier of a corridor, *shape* is the geometric shape which can be *flat*, *cubic*, *rectangular*, *circular*, or *elliptical*. *Coordinates* contains a list of *waypoints* which are defined as:

$$waypoint = < position, size, junctions_{\mathrm{j}} >, \;\; j \in \mathbb{N}$$

The *position* indicates the global position of a given waypoint. *Size* states the outer size of the geometric *shape*. *Junctions* holds a set of identifiers of other adjacent *corridors* or *areas*. An *area* is described as:

$$area = < id, elevation, height, coordinates_{\mathrm{i}}, \\ junctions_{\mathrm{j}} >, \;\; i, j \in \mathbb{N}$$

The property *id* and *junctions* are equivalent to the properties from *corridor* and *waypoint*. *Coordinates* holds a list of geographical coordinates containing longitude and latitude.

First and last list entry must contain the same coordinates to form a closed ring. *Height* and *elevation* of an *area* is described in the according values.

$$memento_{(\mathrm{i})} = < S_\mathrm{i}, t_\mathrm{i}, E_\mathrm{i}, D_\mathrm{i} >, \quad i \in \mathbb{N}$$

Another concept from [12] that will be used is the so called *memento*, which can be described as a data-quadruple that contains the identifier of a Spatial Entity (SE) (S_i), time (t_i), vehicle identity (E_i) and a payload D_i.

D. Data analysis and C-ITS

A Cooperative Intelligent Transport System (C-ITS) describes the communication between traffic related entities like vehicles, RSUs, Traffic Command Centers (TCCs) and other smart infrastructure. These entities are also referred to as *nodes*. Especially in cooperative ITS, every node is actively contributing data. A core component of any ITS is a Local Dynamic Map (LDM). This LDM can be stored on any node to different extents and is often maintained with data from other nodes. Data on LDMs can be categorised into four types:

- *Type 1 - permanent static*: contains information about geography, road topology, points of interest or speed limits
- *Type 2 - transient static*: information about traffic signals and signs
- *Type 3 - transient dynamic*: describes information about weather, road works, changes in traffic conditions
- *Type 4 - dynamic*: primarily V2V communication via Cooperative Awareness Messages (CAMs) and De-centralized Environmental Notification Messages (DENMs)

There are multiple concepts and systems for large scale data processing in C-ITS or Smart Cities. These systems usually share a set of similar basic tasks, namely: collecting a large amounts of heterogeneous data from different sensor and processing it in a certain manner. This is also reflected in the underlying architecture, which consists of the following basic components: a device layer (containing a variety of sensors), the data collection layer (providing an interface and data aggregation for these sensors) and a processing layer that utilises the collected data. To accomplish these goals well-known IoT and Big Data tools like GeoMQTT, Apache Storm, Kafka, Spark, Flink and Hadoop are applied [13]–[15]

III. Large scale multimodal data processing middleware

In this section a novel middleware for application in ITS that utilizes *SpatialJson* is introduced. As described in section II-D, there are numerous approaches to this challenge. However, the middleware presented here differs from these in several aspects. Often, the collected data is fed into a high level application with no direct benefits for the data providing vehicle or nodes. The primary aim of the proposed middleware is to generate value for said vehicles by aggregation, evaluation and re-distribution of the collected data. Also, this middleware focuses on multimodality, since in the future traffic paths may be shared by different types of vehicles. However, this aggregated and processed data can also be used for further applications - thus, providing the foundation for larger ITS applications. To meet these goals, this middleware is designed to be inherently scalable, so that a large area and many vehicles can be processed. A key concept of the described middleware is multi sensor data fusion. The aggregation of data from multiple independent data sources allows information verification and subsequently trust establishment of vehicles. The intended deployment location of this described middleware is typically on static infrastructure such as a RSU, cellular base station, or on servers. Thus, this middleware mainly relies on V2N and V2I long range communication as described in section II-A - leaving short range communication for other applications. Relying solely on V2N and V2I commination also strengthens the multimodality aspect, since especially V2N communication hardware is abundant and cheap. However, the acquired information from this middleware about the authenticity of other vehicles can be used to secure short range communication. In the further course of this paper, we assume, that the introduced middleware is deployed on a central location that is connected to various RSUs (V2I) and cellular networks (V2N) as shown in Fig. 1. Key components of the middleware are:

- *Reverse Geocoding Service (RGS)*: This service translates a global position to a list of matching (enclosing) SEs. Note that the list entries only contain an information subset of an SE including: identifier, geometric type (area or corridor), and behavioral type (see section III-D)
- *Snapshot Data Service (SDS)*: This service provides the current data object that represents an SE. (see section III-E)
- *Push Data Service (PDS)*: This service is responsible for the push message exchange. (see section III-E)
- *Data Assigment Service (DAS)*: The DAS is responsible for sensor data assignment and fusion (see section III-G)
- *Data Storage*: A database management system of some kind.

The POC implementation uses a representational state transfer (REST) API for the RGS and SDS (see Fig. 3). As data storage MongoDB is used, due to its native JSON support, performance and scalability. An overview is provided in Fig. 1.

A. Vehicle Identity

The initial design assigned a random identifier to a vehicle when it started contributing data. However, as described in section II-A, there are key criteria for the identifier of a VANET member that must be met. Including: Non-repudiation, Anonymity, Unlinkability, and Conditional Traceability. In order to fulfill these requirements, every data producing member of a VANET (*node*), must include a Tamper Proof Device (TPD). This TPD includes an immutable secret key K only known to the manufacturer, a clock that provides the current time t, and a global navigation satellite system (GNSS) receiver. The primary function of a TPD is to generate asymmetric key pairs and message signing. In order for a vehicle to contribute data, an identity must be generated beforehand. Thus, a public/private key pair is derived from K, t and the

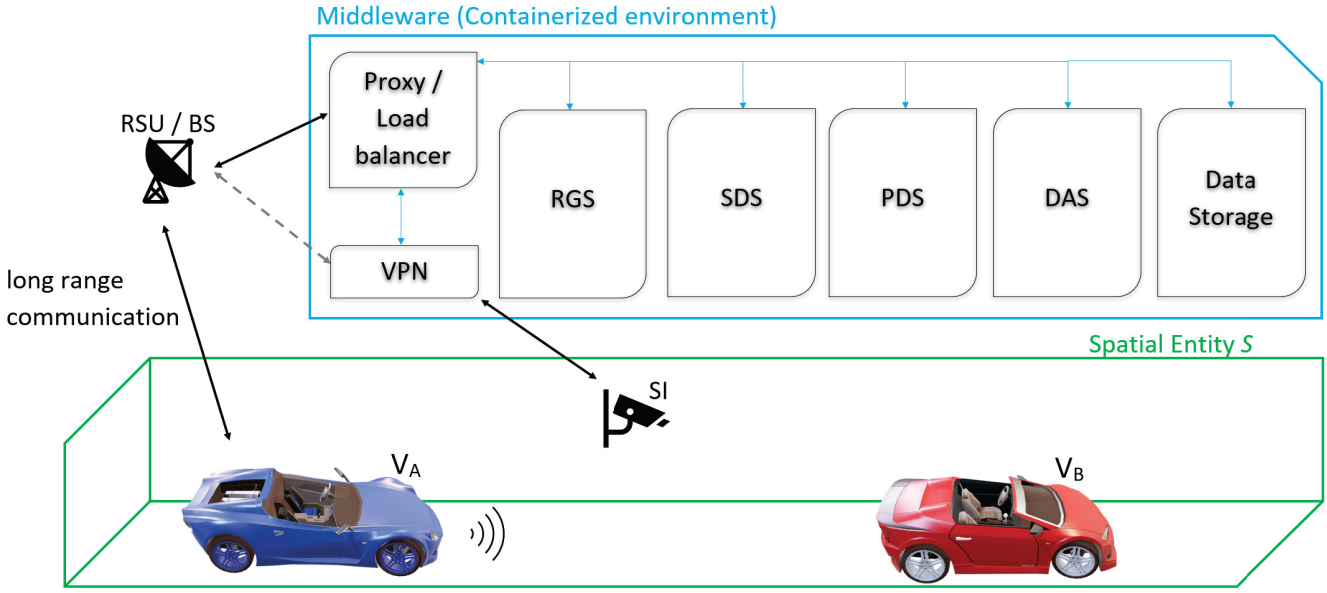

Fig. 1. Middleware components and overview

identifier the current Spatial Entity (SE) I_s (see section III-C) using a Key Derivation Function (KDF) D:

$$< k_{\text{public}}, k_{\text{private}} > = D(K, I_s, t_{\text{entry}})$$

k_{public} now represents the identity of this vehicle. This keypair remains sealed in the TPD until a new one is generated. A new pair is generated every time the vehicle enters another SE, thus t is marked as t_{entry}. The KDF should be cryptographically strong to ensure anonymity and non-repudiation. Before signing a message m, a header h is prepended:

$$h = < I_s, k_{\text{public}}, t_{\text{entry}}, t_{\text{current}}, position >$$

Note that a message header also includes the global position provided by the TPD. The inclusion of a GNSS module offers several advantages. Not only is a source of error and point of attack eliminated, it furthermore allows the TPD to be utilised as a flight data recorder on, for example, Unmanned Aerial Systems (UASs). To guarantee unlinkability, a timestamp t_{entry} is added. This prevents generating the same keys when re-entering an SE. Next a Digital Signature Algorithm (DSA) is applied to the concatenation ($\oplus$) of h and $m \rightarrow (h \oplus m)$ and the resulting signature S is appended to the message. As commonly applied, instead of signing every byte of m and h, a cryptographic hash function can be used to create a corresponding substitute e.g. $S = DSA(k_{\text{private}}, hash(h \oplus m))$. Therefore, a signed message M has the following structure:

$$M = < h, m, S >$$

To verify the authenticity of a message M, k_{public} from h is used to decrypt S with the DSA. Thus, M is valid if $DSA(k_{\text{public}}, S) = hash(h \oplus m)$. Important is, that k_{public} is linked to the specified values to provide non-repudiation. Conditional traceability for authorities is made possible, by granting access to the manufactures list that links a specific K to a vehicle. Hence, the true identity of a TPD (K) can be disclosed, if I_s, t_{entry} and k_{public} is known, assuming the functions D, *DSA* and *hash* are also known. No explicit exclusion of malicious nodes is implemented as it is conducted for example with certificate revocation in a PKI. If the trust level of a node/public key drops below a certain threshold, its messages are ignored.

B. Infrastructure Identity

Static Infrastructure (SI) is also an important part and data source of an ITS. Since this type of node is not obliged to the same service and safety guidelines as vehicles, a different message authentication is used. A suitable technology for safe integration of SI is virtual private network (VPN) that connects directly to the middleware. Therefore, no public/private key pair generation is needed and the message header is constructed as follows:

$$h = < I_n, I_s, t_{\text{created}} >$$

The header h contains the identifier of the node I_n, the target SE I_s and a timestamp t_{created}. I_s is needed since a stationary sensor may observe multiple SEs. A TPD or message signing is not intended.

C. Wide area data processing

As described in section II-D several ITSs use an LDM for data management. Well-known implementations are for example PostGIS or GeoMesa. However, this middleware pursues a more data-centric approach. For this purpose, a highly efficient data centric model described in section II-C is utilised. Typically, data in geographical systems is stored in data layers. To retrieve information for a certain location, intersecting layers are determined and the resulting data aggregated. This querying process can be computational intensive.

To circumvent this, a document based approach with geographical segregation is applied. Traffic paths are therefore segregated into smaller manageable segments. This method also mitigates data categorisation into types and layers. A vehicle can retrieve all data spanning from the topology (type 1) to traffic conditions (type 3) of a traffic segment with a single query (see section II-D). Thus, instead of relying on map features and data layers, a JSON document is created that contains information about the geographical shape of the traffic path segment and also all traffic related information.

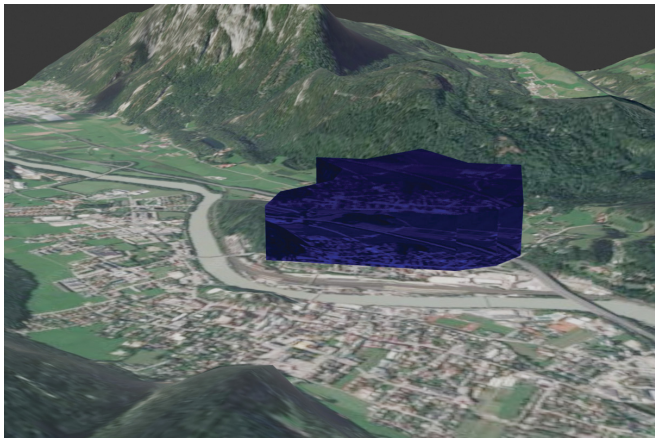

Fig. 2. Illustration of an area for UAS operation

Traffic segments are modeled as Spatial Entities (SEs) using SpatialJSON, e.g. a highway could be modelled as a *Corridor* or an area for Unmanned Aerial System (UAS) operation as an *Area* (see Fig. 2). Note that vehicles or nodes that are subscribed to an SE effectively form their own network. Since the subscribed nodes are constantly changing, a similar dynamic topology as a VANET (section II-A) emerges. Thus, the same principles, security and service requirements are applied to the group of subscribed nodes of an SE.

D. Administrative traffic management

To allow basic administrative traffic management, Spatial Entities can be annotated with the properties *behavior* and (specific for corridors) *unidirectional*. *Behavior* must contain one of the following values:

- *Default*: This is used to mark an SE as *catch all* (see section III-E). For example: an *area* that covers the whole region of operation of an Intelligent Transport System.
- *Traffic*: Marks an SE for regular traffic and is used as the default value if the *behavior* is absent.
- *Exclusion*: An SE marked as *exclusion* is used to state restricted areas. For example, to describe a no flight zone or a closed road.

Spatial Entities marked as *default* or *exclusion* may intersect other SEs. However, entities for *traffic* should not geometrically intersect each other. To connect SEs, the *junctions* attribute provided by SpatialJSON is used. The property *unidirectional* can be added to *Corridors* to indicate the direction for traffic. If the boolean value is true, traffic can only flow in the same direction as the *waypoints* are arranged. Furthermore, nodes are informed when a change of the SE occurs in which they are located. If, for example an SE is created, altered, or deleted, a message containing this information is sent via the PDS to every node that is currently located in the affected geographical region. A restricted area can therefore quickly be deployed through the creation of an area marked as *exclusion*.

E. Data handling

Information exchange is conducted via a push and subscribe architecture. Fig. 3 shows the procedure of basic data-handling which is conducted as follows: Vehicle A (V_A) starts operation. First task is to locate the SE S that encloses V_A via the Reverse Geocoding Service (RGS).

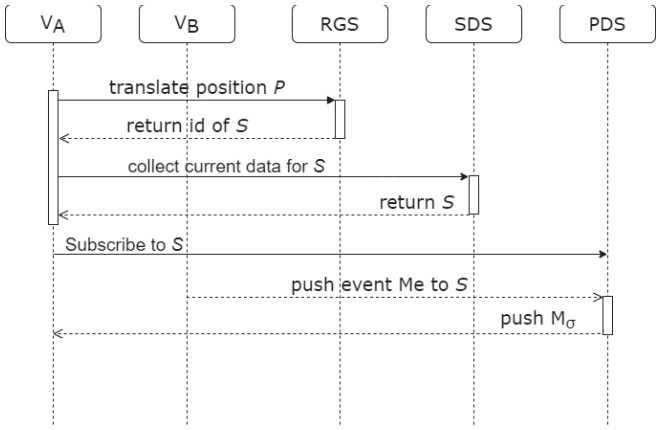

Fig. 3. Data flow diagram

Subsequently, V_A creates a vehicle identity as described in section III-A and obtains all data regarding S by querying the Snapshot Data Service (SDS). Simultaneously V_A subscribes to the Push Data Service (PDS) with the corresponding identifier of S. If then, for example, another vehicle (V_B) encounters an obstacle in S, sensory data is gathered and used as evidence e. Subsequently, a message M_e containing evidence e is sent to the PDS. After processing M_e (see section III-G) an update message M_σ containing a transition object σ_1 is generated and distributed to all subscribed nodes of S. Given q_1 as the state of S before and q_2 as state after the application of evidence e, σ_1 contains the information to mutate $q_1 \overset{\sigma_1}{\rightarrow} q_2$ using a transformation function T. Thus, it must be ensured that a state q_n is reproducible by applying $\sigma_1, ..., \sigma_n$

$$q_1 = \emptyset, \;\; q_{i+1} = T(q_i, \sigma_i), \;\; i \in \mathbb{N}$$

This property is also used to create a data history. Since a transition object σ does not necessarily contain meta information like the affected SE or causing entity, *mementos* are used as update messages and to depict data history:

$$memento_1 = < q_1^{id}, t, V_B^{id}, \sigma_1 >$$

Note that a memento only contains the identifier of q_1 and V_B. As mentioned, nodes exchange information via a publish / subscribe service. In principle, any push-capable technology can be used to create such a Push Data Service (PDS). However, the POC setup has revealed that message brokers like MQTT and Kafka raised issues regarding speed

and compatibility. A web socket interface that accepts an identifer of an SE as URL parameter has proven to be the most suitable for data exchange. Web socket is well defined, allows for bidirectional data push with low overhead, and provides secure communication via TLS. Another key aspect of this middleware is an agnostic treatment of data sources. Meaning, every node can produce information about another entity. However, as shown in section III-F, an explicit segregation of intrinsic and external information is needed. Also an important aspect is the management strategy for Spatial Entity. Here an ahead of time creation of SEs according to type 1 data from other services has yielded the best results. Since dynamically created SEs often do not represent traffic paths, but vehicle specific routs. A static approach also allows for easier reconstruction of data history. It is however advantageous to continuously implement small incremental adjustments to compensate changes in traffic infrastructure or prevent data congestion on Spatial Entities with large volume of traffic. For vehicle operation outside of pre-defined SEs either a *catch all* area or a dynamic creation of SEs could be established.

F. Data format and model

Since this middleware is designed to support a variety of different Intelligent Transport Systems, an universally applicable data format and model is discussed here. In order to ensure extensibility, multimodality, and compatibility, a semantic approach is applied. Since JSON-LD is capable of modelling sematic data in JSON, any data scheme or ontology can therefore be implemented. There are various existing schemes and ontologies for describing traffic data. These schemes are however often very application specific and therefore not suitable for this middleware. Instead, only a few small schemes are used, which provide the bare minimum for operation. It is however possible to add additional ontologies or schemes to extend the functionality. The lightweight Ontology for Sensors, Observations, Samples, and Actuators (SOSA)[16] combined with GeoCoordinates (https://schema.org/GeoCoordinates) and W3C Custom Datatypes (https://ci.mines-stetienne.fr/lindt/v4/custom_datatypes) allows for basic geo-based event and data reporting as shown in [17]. Thus, these namespaces (see listing 1) are used as default context and do not need to be stated explicitly in the *@context* property. Additionally, this middleware differentiates data in three different categories:

- *Intrinsic*: Information in this category is about the sending node itself. This includes for example current vehicle speed.
- *External*: This category contains information that is not intrinsic like the position and speed of surrounding vehicles, obstacles or traffic conditions.
- *Meta*: This category contains information about the SEs itself. For example changes in size or shape. Messages containing this information are usually not sent by vehicles.

These data categories are needed for data evaluation and assignment (see section III-G) and are therefore mandatory for every reported data. Thus, the new property *descriptionTarget* is introduced. *DescriptionTarget* holds a string that is one the three information categories stated above. Another annotation that is needed in the later course is the differentiation between an unconfirmed, cooperative and none-cooperative vehicle (see listing 3). Therefore, another property called *vehicleStatus* is introduced:

- *Unconfirmed*: Indicates that there is not enough evidence to confirm existence of this vehicle.
- *Cooperative*: This indicates, that a vehicle is legit and cooperative.
- *Uncooperative*: Vehicles categorised as uncooperative have been detected multiple times and exist, but are not contributing data.

Finally, a property for the trust score is added. This property contains a number between 0 (no trust) and 1 (trustworthy) and can be assigned to observations, since these also have an inherent uncertainty. The default context of any message to and from this middleware is therefore defined as:

Listing 1. Default context of a message

```
"@context": {
   "dt": "http://raich.fh-kufstein.ac.at/
      descriptionTarget",
   "status": "http://raich.fh-kufstein.ac.at/
      vehicleStatus",
   "ts": "http://raich.fh-kufstein.ac.at/
      trustScore",
   "sosa": "http://www.w3.org/ns/sosa/",
   "cdt": "http://w3id.org/lindt/
      custom_datatypes#",
   "location": "https://schema.org/
      GeoCoordinates",
   "sch": "https://schema.org/",
   "rdfs": "http://www.w3.org/2000/01/rdf-
      schema#",
   "xsd": "http://www.w3.org/2001/XMLSchema#"
}
```

As a concrete example we assume a smart and cooperative road vehicle V_A that obtained public key *aGVsbG8gOkQ=* while being in SE *e539b800e40e*. This vehicle is traveling with $40m/s$ and its onboard sensors (e.g. vision-based vehicle detection or LIDAR) are detecting a vehicle $25m$ ahead of it. A corresponding message M is shown in listing 2.

Listing 2. Example message M from vehicle to middleware

```
{
   "header" : {
      "spatialEntityId": "e539b800e40e",
      "publicKey": "aGVsbG8gOkQ=",
      "entryTime": 1628084796,
      "created": 1628084796,
      "longitude": 12.150108909,
      "latitude": 47.580416114,
      "elevation": 0
   },
   "message" : [
      {
         "@id": "ex:distanceMeter/1",
         "@type": "sosa:Sensor",
         "dt": "external",
         "rdfs:label": "Front distance meter"
      },{
         "@id": "ex:speedometer/1",
         "@type": "sosa:Sensor",
```

```
20            "rdfs:label": "VehicleSpeedometer"
21         },{
22            "@id": "ex:observation/1",
23            "@type": "sosa:Observation",
24            "ts": 0.95,
25            "location": {
26               "longitude": 12.15033041,
27               "latitude": 47.58067787,
28            },
29            "sosa:resultTime" : 1628084796,
30            "sosa:hasSimpleResult": 25,
31            "sosa:madeBySensor": {
32               "@id": "ex:distanceMeter/1"
33            }
34         },{
35            "@id": "ex:observation/2",
36            "@type": "sosa:Observation",
37            "dt": "intrinsic",
38            "ts": 1.0,
39            "sosa:resultTime" : 1628084796,
40            "sosa:hasSimpleResult": {
41               "@literal": "40 m/s",
42               "@datatype": "cdt:speed"
43            },
44            "sosa:madeBySensor": {
45               "@id": "ex:speedometer/1"
46            }
47         }
48      ]
49   }.MCwCFCYflqAvx9ybbpY89ia4AMpqgA==
```

The property *header* contains the information prepended by the TPD as described in section III-A. *Message* holds the JSON-LD graph as an subject centred list. Each item contains multiple rdf triples which are linked to the group subject (*@id*). In this example the items with subject *ex:speedometer/1* and *ex:distanceMeter/1* are onboard sensor of V_A. *ex:distanceMeter/1* is marked as external and therefore provides data about external entities. This implies that *Ex:observation/1*, which was generated by *ex:distanceMeter/1*, must also describe external information. Every external observation must include the location of the observed entity, stated in the property *location*. Thus, telling the middleware that an entity exists at the stated point of time (line 29), at the specified location and specified distance to V_A. For intrinsic observations the time and position stated in the message header are used implicitly. However, it may also be stated explicit as shown in line 39. Every observation, sampling or actuation must contain a timestamp. Finally, a cryptographic hash sum is calculated from the whole JSON, ciphered as described in section III-A and appended as a Base64 encoded string (shown in line 49). As described in section III-C, SpatialJSON is used as basic data model. In order to store traffic data in a geographical model, this middleware adds three additional properties to a Spatial Entity:

- *vehicleInformation (JSON-LD)*: This property contains information about cooperative and uncooperative vehicles.
- *sensorInformation (JSON-LD)*: Here, other measurable information is stored like obstacles, wind speed, temperature, but also information from smart infrastructure like connected traffic lights.
- *metaInformation (informal)*: this property includes information about the SE itself like the administrative traffic management properties introduced in section III-D.

An example is shown in listing 3. Another important concept of this middleware is, that no obsolete or invalid data is retained in an SE. This paradigm facilitates the creation of a faithful digital representation of the physical traffic segment. Thus, if a node retrieves an SE from the Snapshot Data Service, a recent representation must be returned.

G. Data assignment and processing

In this section message verification, assignment, and storing as shown in Fig. 4 is discussed. First a validation of the incoming message M is conducted. If the message originated from a vehicle it must contain a valid signature. Additionally, it is checked if the stated position in the header is geographically contained in the stated Spatial Entity. If the message originated from a vehicle, the intrinsic vehicle data and trust score is updated or a new unconfirmed vehicle is created. As shown in listing 2, there is no entity identifier present for external observation. Also, observations conducted by SI does not contain the an identifier either. However, every external information must contain a position and time and can therefore be spatially mapped. Here, the Data Assigment Service (DAS) is responsible for determining the corresponding entity for a given information. Currently, the Proof of Concept (POC) uses a distance-time metric and a threshold θ to determine if the reported data can be assigned to an existing vehicle. Assuming p_σ as the position and time of an observation in M and P as a set of vehicles of a given SE results in:

$$\theta_i = \sqrt{distance(p_\sigma, p_i)^2 + t_\Delta(p_\sigma, p_i)^2}\ ,\quad p_i \in P$$

If $min(\theta_1 \dots \theta_i) < \theta$ a match is returned. Subsequently, if M can be assigned or originated from Static Infrastructure, evidence evaluation starts as described in section II-B. Currently the POC differentiates between generic external information and information about other vehicles. Generic information is added to the SE and no further action is undertaken. Information about the position of a vehicle (here stated as α_i) on the other hand is processed as follows: First the prior probability $P[\alpha_i]$ is set to the trust value of the reporting node. Subsequently, the probability of e describing α_i is calculated. Here Raya, Papadimitratos, Gligor, *et al.* suggest that $P[e_k^j|\alpha_i]$ correlates to the corresponding trust level, $F(e_k^j)$ of the previous collected evidence e for α_i. For the likelihood $P[e_k^j|\alpha_h]$ is stated as $1 - F(e_k^j)$. The estimated amount of malicious nodes finally determines $P[\alpha_h]$. The probability P for this procedure is determined with the earlier stated distance-time metric, where a larger θ indicates a lower probability.

Subsequently, if $P[\alpha_i|e]$ exceeds a certain threshold, the trust score of the evidence as well as the reporting vehicle will be increased. Finally, if the data of an SE is altered by e.g. updating vehicle information or providing new evidence, a memento is generated and distributed as M_σ to every subscriber of said SE. The exact procedure is depicted in Fig. 4. Note that new vehicle cannot immediately submit evidence. Also, a vehicle must be observed by Static Infrastructure to obtain the status *confirmed*. To prevent information withholding, every piece of information is published to subscribed nodes. However, to intercept tampering the trust score is set (or

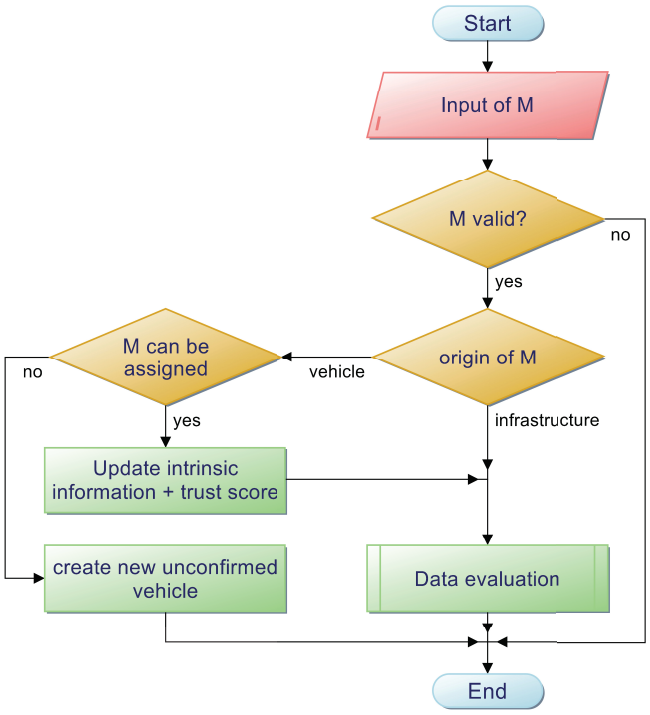

Fig. 4. Data processing

overwritten) by the DAS. If, for example, a piece of evidence was submitted with a trust score of 0.8 but could not be verified be the evaluation process, it will be lowered to 0.1.
To continue the example from section III-F, we assume Spatial Entity S is in state q_1 as described in listing 3:

Listing 3. JSON document describing S at state q_1

```
{
   "_id": "e539b800e40e",
   "type": "Corridor",
   "shape": "Rectangular",
   "coordinates": "...",
   "vehicleInformation" : [
      {
         "@id": "aGVsbG8gOkQ=",
         "@type": "sch:Vehcile",
         "ts": 0.6,
         "sch:speed": {
            "@literal": "35 m/s",
            "@datatype": "cdt:speed"
         },
         "sch:dateModified" : 1628361789,
         "status": "cooperative",
         "location": {
            "longitude": 12.149790699,
            "latitude": 47.5799774
         }
      }
   ],
   "sensorInformation":[
      {
         "@id": "8d0b5e7820fd",
         "@type": "sosa:Observation",
         "sosa:observedProperty": "
            temperature",
         "sosa:resultTime" : 1628361789,
         "sosa:hasSimpleResult": {
            "@literal": "22.3 Cel",
            "@datatype": "cdt:temperature "
         }
      }
   ],
   "metaInformation":{
      "behavior": "traffic",
      "unidirectional": true
   }
}
```

Here, S contains the information about velocity and position of vehicle V_A, sensory data about air temperature and some meta data about S. In our example the signature of M is valid. Also the *publicKey*, *spatialEntityId* and position can successfully be assigned to existing data. Afterwards, the intrinsic data in M is added to the database, where old values are updated and missing values are inserted. Subsequently, the external information e is assessed as described above. After implementing e, S should be in the state shown in listing 4:

Listing 4. JSON document describing S at state q_2

```
{
   "_id": "e539b800e40e",
   "type": "Corridor",
   "shape": "Rectangular",
   "coordinates": "...",
   "vehicleInformation" : [
      {
         "@id": "aGVsbG8gOkQ=",
         "@type": "sch:Vehcile",
         "ts": 0.7,
         "sch:speed": {
            "@literal": "40 m/s",
            "@datatype": "cdt:speed"
         },
         "sch:dateModified" : 1628084796,
         "status": "cooperative",
         "location": {
            "longitude": 12.150108909,
            "latitude": 47.580416114
         }
      },{
         "@id": "c818fa1010d3",
         "@type": "sch:Vehcile",
         "sch:dateModified" : 1628084796,
         "ts": 0.0,
         "status": "uncooperative",
         "location": {
            "longitude": 12.15033041,
            "latitude": 47.58067787
         }
      },{
         "@id": "fc1c9603516b",
         "@type": "sosa:Observation",
         "ts": 0.75,
         "location": {
            "longitude": 12.15033041
            "latitude": 47.58067787,
         },
         "sosa:resultTime" : 1628084796,
         "sosa:hasSimpleResult": 25,
         "sosa:madeBySensor": {
            "@id": "aGVsbG8gOkQ="
         }
      }
   ],
```

```
46    "sensorInformation":[
47       {
48          "@id": "684a3cc5029a",
49          "@type": "sosa:Observation",
50          "sosa:observedProperty": "
               temperature",
51          "sosa:resultTime" : 1628361789,
52          "sosa:hasSimpleResult": {
53             "@literal": "21.1 Cel",
54             "@datatype": "cdt:temperature "
55          }
56       }
57    ],
58    "metaInformation":{
59       "behavior": "traffic",
60       "unidirectional": true
61    }
62 }
```

Finally, a transition object σ and an update message M_σ are generated and distributed. As defined earlier ($q_1 \xrightarrow{\sigma_1} q_2$), σ_1 contains the difference (Δ) of these two states. Thus, $\sigma_1 \equiv q_1 \Delta q_2$. Which means σ_1 contains every altered JSON property. Since any JSON object is organized as a tree structure, a recursive data merging function (T) can be applied. However, JSON-LD allows for a more elegant and precise data manipulation. As described in section III-E, this middleware uses *mementos* for data history. Bundling mementos into a single history object enables a more efficient way of storage. Thus, the POC uses the following format to create a data history H:

$$H = < I_h, I_s, t_{created}, \{memento_0, ..., memento_n\}, S >$$

Where H contains its identifier I_h, the identifier of S, a timestamp t, the set of mementos, and S before applying them (q_1).

IV. Proof of concept and future work

The described services (RGS, SDS, PDS, PDS) of the proposed middleware are implemented using Red Hat Quarkus (https://quarkus.io/), compiled in JVM mode and executed in a Docker environment. Originally, Apache Kafka was used to interconnected these micro-services. It turned out that especially the DAS must be omniscient about an SE to successfully assign data, resulting in a rearrangement to the current structure. Implementing the public/private key authentication described in III-A utilises the *java.security.KeyPairGenerator* and *java.security.java.security*. These classes need an instance of *java.security.SecureRandom*. To create reproducible keys, the method *nextBytes(byte[] bytes)* was overwritten to return the bytes of the hashed parameters of h. Here further refinement and a propped implementation of a DSA is needed. As mentioned, early prototypes used Kafka for data transport and ingestion. Here every SE interchanges data through its own Kafa topic. Besides the computational strain of running Kafka on Low Power Devices like UAS, the main issue emerged when splitting SEs. Here the newly created Kafka topic required a considerable amount of time to be distributed to the Apache Zookeeper instance and be usable. Testing earlier versions on a Raspberry Pi 3b with cellular connection, also reviled that the cpu utilisation of the Kafka-client severely impacts the throughput. Therefore, a simple tcp socket is now implemented. Another aspect of this middleware that will be addressed in the future is Spatial Entity roaming. Raya, Papadimitratos, Gligor, *et al.* [10] stats that trust based approaches functions better, the longer a vehicle contributes data. The accumulated trust values should therefore be conveyed to the next SE without violating the *Location Privacy Preservation* or other VANET security requirements (section II-B). Also, a more comprehensive model for trust management should be included. Currently not all available data is considered to calculate the trust values and thresholds. As described in section III-F, old data is constantly removed from SE. Presently, every dataset older than a fixed value is removed. A better approach is to evaluate the actuality and dynamically remove it. To solve this issue more attributes to the data model must be introduced. At a certain point, however, it is reasonable to create a new ontology that covers the mentioned functionality. Furthermore, the evidence evaluation for generic data, as described in section III-G, is quite rudimentary. RDF allows for semantic reasoning which may be applicable in this scenario. Additionally, more sophisticated approaches for data assignment can be applied. Using a multi target multi sensor tracking algorithm as shown in [18], [19] should increase the assignment rate compared to the current distance-time metric. These algorithms tend to become computational expensive when processing a large number of vehicles. However, the spatial segregation should compensate this issue.

V. Conclusion

To conclude, creating a large scale multimodal data processing middleware that checks all modern security and service requirements is a ambitious task. This paper outlines a fundamental approach to overcome these challenges. First results successfully demonstrate the viability of the proposed middleware. However, a large scale test has not been conducted yet and will be subject to future work.

References

[1] F. Wang, Y. Xu, H. Zhang, Y. Zhang, and L. Zhu, "2FLIP: A two-factor lightweight privacy-preserving authentication scheme for VANET," *IEEE Trans. Veh. Technol.*, vol. 65, no. 2, pp. 896–911, 2016, ISSN: 00189545. DOI: 10.1109/TVT.2015.2402166.

[2] B. L. Nguyen, D. T. Ngo, N. H. Tran, and H. L. Vu, "Combining V2I with V2V Communications for Service Continuity in Vehicular Networks," *2019 IEEE Intell. Transp. Syst. Conf. ITSC 2019*, pp. 201–206, 2019. DOI: 10.1109/ITSC.2019.8916893.

[3] D. B. Rawat, B. B. Bista, G. Yan, and M. C. Weigle, "Securing vehicular ad-hoc networks against malicious drivers: A probabilistic approach," *Proc. Int. Conf. Complex, Intell. Softw. Intensive Syst. CISIS 2011*, no. June, pp. 146–151, 2011. DOI: 10.1109/CISIS.2011.30.

[4] J. Huang, D. Fang, Y. Qian, and R. Q. Hu, "Recent Advances and Challenges in Security and Privacy for V2X Communications," *IEEE Open J. Veh. Technol.*, vol. 1, pp. 244–266, 2020. DOI: 10.1109/ojvt.2020.2999885.

[5] R. Molina-Masegosa and J. Gozalvez, "LTE-V for Sidelink 5G V2X Vehicular Communications: A New 5G Technology for Short-Range Vehicle-to-Everything Communications," *IEEE Veh. Technol. Mag.*, vol. 12, no. 4, pp. 30–39, Dec. 2017, ISSN: 1556-6072. DOI: 10.1109/MVT.2017.2752798. [Online]. Available: http://ieeexplore.ieee.org/document/8080373/.

[6] J. B. Kenney, "Dedicated short-range communications (DSRC) standards in the United States," *Proc. IEEE*, vol. 99, no. 7, pp. 1162–1182, 2011, ISSN: 00189219. DOI: 10.1145/3448823.3448854.

[7] A. Festag, "Cooperative intelligent transport systems standards in europe," *IEEE Commun. Mag.*, vol. 52, no. 12, pp. 166–172, Dec. 2014, ISSN: 0163-6804. DOI: 10 . 1109 / MCOM . 2014 . 6979970. [Online]. Available: http://ieeexplore.ieee.org/document/6979970/.

[8] ITU-R, "Intelligent transport systems (ITS) usage," ITU International Telecommunication Union, Tech. Rep., 2018. [Online]. Available: https://www.itu.int/dms%7B%5C_%7Dpub/itu-r/opb/rep/R-REP-M.2445-2018-PDF-E.pdf.

[9] M. M. Zanjireh and H. Larijani, "A Survey on Centralised and Distributed Clustering Routing Algorithms for WSNs," in *2015 IEEE 81st Veh. Technol. Conf. (VTC Spring)*, IEEE, May 2015, pp. 1–6, ISBN: 978-1-4799-8088-8. DOI: 10.1109/VTCSpring.2015.7145650. [Online]. Available: http://ieeexplore.ieee.org/document/7145650/.

[10] M. Raya, P. Papadimitratos, V. D. Gligor, and J.-P. Hubaux, "On Data-Centric Trust Establishment in Ephemeral Ad Hoc Networks," no. June 2014, pp. 1238–1246, 2008. DOI: 10.1109/infocom.2008.180.

[11] H. Butler, "The GeoJSON Format," *J. Chem. Inf. Model.*, 2016, ISSN: 1098-6596. DOI: 10.1017/CBO9781107415324.004. arXiv: arXiv:1011.1669v3.

[12] K. Raich, R. Kathrein, M. Erharter, and M. Döller, "Spatial Extension Model for Multimodal Traffic Management," in *Proc. 2020 4th Int. Conf. Vision, Image Signal Process.*, FH Kufstein, New York, NY, USA: ACM, Dec. 2020, pp. 1–6, ISBN: 9781450389532. DOI: 10.1145/3448823.3448854. [Online]. Available: https://dl.acm.org/doi/10.1145/3448823.3448854.

[13] S. Herle, R. Becker, and J. Blankenbach, "Bridging GeoMQTT and REST," *CEUR Workshop Proc.*, vol. 1762, 2016, ISSN: 16130073.

[14] M. A. Javed, S. Zeadally, and E. B. Hamida, "Data analytics for Cooperative Intelligent Transport Systems," *Veh. Commun.*, vol. 15, pp. 63–72, 2019, ISSN: 22142096. DOI: 10.1016/j.vehcom.2018.10.004. [Online]. Available: https://doi.org/10.1016/j.vehcom.2018.10.004.

[15] M. Laska, S. Herle, R. Klamma, and J. Blankenbach, "A Scalable Architecture for Real-Time Stream Processing of Spatiotemporal IoT Stream Data—Performance Analysis on the Example of Map Matching," *ISPRS Int. J. Geo-Information*, vol. 7, no. 7, p. 238, 2018, ISSN: 2220-9964. DOI: 10.3390/ijgi7070238.

[16] K. Janowicz, A. Haller, S. J. Cox, D. Le Phuoc, and M. Lefrançois, "SOSA: A lightweight ontology for sensors, observations, samples, and actuators," *J. Web Semant.*, vol. 56, pp. 1–10, 2019, ISSN: 15708268. DOI: 10.1016/j.websem.2018.06.003. arXiv: 1805.09979.

[17] M. Viktorović, D. Yang, and B. de Vries, "Connected traffic data ontology (Ctdo) for intelligent urban traffic systems focused on connected (semi) autonomous vehicles," *Sensors (Switzerland)*, vol. 20, no. 10, 2020, ISSN: 14248220. DOI: 10.3390/s20102961.

[18] B. N. Vo, B. T. Vo, and M. Beard, "Multi-Sensor Multi-Object Tracking with the Generalized Labeled Multi-Bernoulli Filter," *IEEE Trans. Signal Process.*, vol. 67, no. 23, pp. 5952–5967, 2019, ISSN: 19410476. DOI: 10.1109/TSP.2019.2946023. arXiv: 1702.08849.

[19] K. Da, T. Li, Y. Zhu, H. Fan, and Q. Fu, "Recent advances in multisensor multitarget tracking using random finite set," *Front. Inf. Technol. Electron. Eng.*, vol. 22, no. 1, pp. 5–24, 2021, ISSN: 20959230. DOI: 10.1631/FITEE.2000266.

ACKNOWLEDGMENT

This work was funded by Interreg Österreich Bayern 2014 - 2020 Nr. 50506 (DataKMU, https://webta.fh-kufstein.ac.at/Forschen/datakmu).