2011 European Intelligence and Security Informatics Conference

An approach to Intelligent Information Fusion in Sensor Saturated Urban Environments

Charalampos Doulaverakis[*], Nikolaos Konstantinou[†], Thomas Knape[‡], Ioannis Kompatsiaris[*] and John Soldatos[†]
[*]Informatics and Telematics Institute, Centre for Research and Technology Hellas, Thessaloniki, Greece
Email: {doulaver, ikom}@iti.gr
[†]Autonomic and Grid Computing Laboratory, Athens Information Technology, Athens, Greece
Email: {nkons, jsol}@ait.edu.gr
[‡]Data Fusion International, Dublin, Ireland
Email: thomas.knape@datafusion.ie

***Abstract*—This paper introduces a novel sensor information fusion system enabling security and surveillance in large scale sensor saturated urban environments. The system is built over state-of-the art sensor networks middleware and provides information fusion at multiple layers. A distinguishing characteristic of the system is that it support seamless integration with semantic web middleware (including ontologies and inference mechanisms), which enable intelligent high-level accurate reasoning. This is a key functionality for efficient surveillance in large scale environment, where manual inspection of individual tracking systems becomes extremely resourceful and overall impractical. A proof-of-concept implementation of the system manifests its benefits and technical challenges, while also outlining lessons learnt.**

I. Introduction

The proliferation of ubiquitous computing and the parallel decrease of the price of sensors has recently allowed sensor applications to make their appearance in a variety of domains. Especially, sensor applications with emphasis on large scale deployments in urban environments and the emerging wave of participatory sensing applications such as WikiCity [1], CitySense [2], and Google Latitude [3] has offered fundamental changes and advancements both in the academia as well as in the heart of the society itself. Nowadays, more than ever, we are witnessing the materialization of visions and concepts such as the Internet-of-Things (IoT) [4] and M2M communications [5].

Recent incidents have manifested that modern cities are very susceptible to terrorist attacks. For instance, the collapse of New York's Twin Towers on 11th September 2001, the bombing of packed commuter trains in Madrid on 11th March 2004, the London bombings in July 2005, or the Moscow metro in March 2010 demonstrate that prominent applications need to be devised in order to serve critical surveillance and security needs in urban environments.

However, numerous challenges are associated with such efforts. The large scale nature of both the geographically dispersed environment as well as of the volume of the data, the multiple distributed heterogeneous components that need to be assembled, spanning sensor, sensor processing, signal processing (including A/V) components, often from multiple vendors are some of the issues that need to be addressed. Additionally, the need for automation, since manual observation of multiple camera feeds is not possible, and the inclusion of high-level intelligent reasoning for event inference are features without which the added value of the system is limited.

In order to tackle these issues in global sensor networks, various frameworks have been developed, offering programmable and configurable solutions, e.g. [6]–[8]. Among the most important in these frameworks is the limited support for perceptual processing components – with the exception of some more heavyweight frameworks [9] – and limited support for semantics embodiment, inference, and high-level reasoning.

Employment of Semantic Web technologies in sensor networks research focuses mainly on sensor modelling in order to enable higher level processing for event/situation analysis. Initiatives such as [10] use SUMO as their core ontology under which they define a sensor ontology for annotating the sensors and it can be then queried for sensor discovery. Other frameworks, such as the one proposed in [11], define ontologies for sensor measurement and sensor description. The framework emerges from the W3C Semantic Sensor Network XG[1] which aims at providing ontologies and semantic annotations that define capabilities of sensors and sensor networks.

Ontologies have been proposed for situation awareness (SAW) in sensor fusion applications where their ability to model a domain or a "part of the world" is utilized. Several approaches have been proposed as in [12] where the authors define a core ontology for SAW which can be used as a basis from which to build separate ontologies for arbitrary situations that are able to express objects, relations and their evolution over time.

In [13], the authors used an ontology to create a unified expression of the Situation Theory [14], [15]. The Situation Theory Ontology (STO) is expressed in OWL which enables situations to be described using a formal language, thus allowing inference through a reasoning engine or by using appropriate rules.

A framework for the designing of ontology for SAW is presented in [16] where a six step guide to design ontologies

[1] W3C Semantic Sensor Network Incubator Group: http://www.w3.org/2005/Incubator/ssn/

978-0-7695-4406-9/11 $26.00 © 2011 IEEE
DOI 10.1109/EISIC.2011.24

IEEE computer society

based on the Basic Formal Ontology (BFO) is presented. The authors test their framework in a situation assessment in a post-disaster environment context and claim that their BFO-based ontology was able to capture the complexities of the testing context and provide adequate inferential capabilities for higher level fusion.

All the above methods do not propose an architecture which will integrate sensors and low level processing modules with higher level fusion process for situation awareness. In order to address this limitation, a solution is presented in this paper that

- Comprises a multi-level fusion system, at all JDL (see Section II-B) levels
- Seamlessly blends ontologies with low-level information databases
- Combines semantic web middleware with sensor networks middleware

The structure of the paper is as follows: Section II analyses the architectural approach that is followed, Section III demonstrates the Low Level Fusion (LLF) capabilities, Section IV the semantic and reasoning capabilities, Section V presents implementation details and example use cases, while Section VI concludes the paper by presenting out comments and remarks.

II. Proposed application and architecture

In order to test our approach, a suitable environment for setting up and deploying the proposed sensor analysis architecture had to be selected. Security surveillance environments offer an ideal set up in which the main characteristics that make it distinguishing are:

- Usually security surveillance areas are sensor saturated environments with electro-optical (visual and IR cameras) and acoustic sensors being more common while others like temperature or RFID, etc sensors can also be found. The abundance of available sensors makes it more possible to capture the various events that take place in a surveillance session. However, the higher the number of sensors, the more difficult it is to manage and observe them. Additionally, it makes it harder to filter out information that is irrelevant or discover information that could potentially be useful.
- Surveillance environments are deployed in areas that are densely populated in terms of people but also in assets, e.g. buildings or vehicles, hence many events that could be of interest are taking place. This fact, coupled with the higher chance of these events to be captured by the sensors (the previous characteristic), makes discovery of important events difficult.
- Multiple processing algorithms and context-acquisition components, which are used for extracting information and detecting events, require a method for managing the data that they produce.
- Due to the variety of sensor modalities that are deployed there is a large degree of sensor and data heterogeneity that need to be tackled by the sensor management and analysis system.

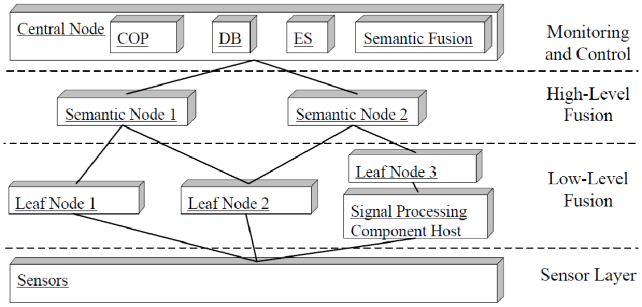


Fig. 1. Architecture of the proposed system

Public security in urban areas can be a challenging task to security personnel. Threats posed to citizens include crimes such as robberies, terrorism attacks, natural catastrophes and other emergencies. Being aware of threatening situations either in the moment or being able to forecast events from security monitoring applications enables security personnel to take the right actions in context. Fusion of data from appropriate sensor types supports inferring threatening situations in the context of situation specific parameters. An example of such a deployed sensor network is the case of urban surveillance in an urban war environment where data feeds from multiple sensors, e.g. visible spectrum and IR cameras, acoustic sensors, seismic sensors, etc, are directed to a command center and critical decisions have to be made in short time regarding the threat level in a situation.

Taking into account the above facts, a method is required that will enable efficient information processing and management. Semantic web and ontologies can efficiently handle heterogeneous information through semantic description of knowledge. Additionally, ontologies are used to model domain knowledge through class definitions and relations between classes. The knowledge model and the underlying data can then be used for supporting reasoning services in order to infer new knowledge that is not explicitly stated. These features can be utilized in a sensor network in order to provide the backbone for intelligent sensor fusion.

A. System architecture

Figure 1 illustrates the approach followed in the hereby presented work. At the bottom layer are the sensors, that are translating their perception of the world into raw sensor data. This data is processed by the leaf nodes, that are operating at the LLF layer which adds structure to the data. The nodes are named “Leaf” nodes because, if the architecture can be considered as tree-like, then these nodes can be considered as leaves, in the sense that they do not have children nodes of inferior capabilities or at a lower level. When deemed necessary, signal processing components can be hosted in dedicated hardware since, specifically in A/V processing, algorithms may be extremely resource-hungry. At the high-level fusion (HLF) layer, intelligence is added to the system, by mapping the collected data into ontology concepts, achieving thus uniform information representation throughout the system and enabling reasoning.

The Central Control layer that is logically on top of the infrastructure offers monitoring and control capabilities. In essence, it comprises:

- A database (DB) where high-level information such as events and threats is stored
- An Environmental Service (ES) that enables geospatial services
- The Semantic Fusion component that performs analysis on the system-wide high-level collected information in order to infer events and potential risks and threats
- The Common Operational Picture (COP) that offers a visualization of the system state with regard to deployed sensors, detected objects, sensed events and inferred threats.

In the following sections, we analyse how the aforementioned layers and components function and interoperate in order to achieve intelligent information fusion in an urban environment.

B. JDL model for sensor fusion

In order to improve communications among military researchers and system developers, the Joint Directors of Laboratories (JDL) Data Fusion Working Group began an effort to define the terminology related to data fusion. The result of that effort was the creation, in 1986, of a process model for data fusion and a data fusion lexicon [17]. The JDL process model is a paper model of data fusion and is intended to be very general and useful across multiple application areas. The JDL data fusion process model is a conceptual model which identifies the processes, functions, categories of techniques, and specific techniques applicable to data fusion (Figure 2). According to the model data fusion process is conceptualized by sensor inputs, human-computer interaction, database management, source preprocessing, and four key subprocesses:

1) Level 1 (Object Refinement): is aimed at combining sensor data together to obtain a reliable estimation of an entity position, velocity, attributes, and identity;
2) Level 2 (Situation Refinement): dynamically attempts to develop a description of current relationships among entities and events in the context of their environment;
3) Level 3 (Threat Refinement): projects the current situation into the future to draw inferences about enemy threats, friendly and enemy vulnerabilities, and opportunities for operations;
4) Level 4 (Process Refinement): is a meta-process which monitors the overall data fusion process to assess and improve the real-time system performance.

Revisions of the model suggest the addition of 2 more levels: A lower Level 0 (Source preprocessing/Sub-object refinement) which aims to estimate the state of sub-object entities such as signals and features, and a higher Level 5 (Cognitive or User Refinement) which introduces man in the fusion loop where the aim is to generate fusion information according to the needs of the system user.

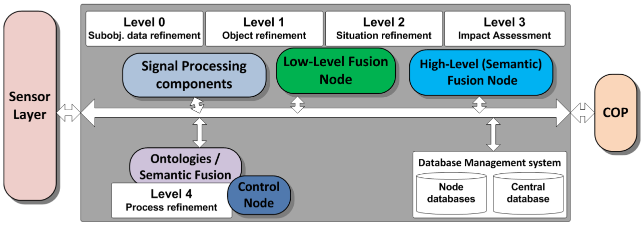


Fig. 2. The JDL model levels and how they are related to the proposed system's layers and nodes

The current paper focuses in the definition of a sensor fusion architecture which addresses matters that correspond to the higher JDL levels 2, 3 and 4, i.e. situation, threat assessment and process refinement respectively, as is depicted in Figure 2. Ontologies, with their inherent ability to model relations, can be employed to provide enough specificity in describing the higher level concepts of JDL but also to describe the relations between these concepts [18]. The ontology provides the standardized form in which situations are defined so that appropriate algorithms for situation assessment can be formulated.

III. Low-Level Fusion Capabilities

In order to perform Low-Level Fusion, our work relies on the GSN[2] middleware. The middleware constitutes an open-source, Java-based implementation. It was designed in order to allow processing from a large number of sensors and as such, it covers the functionality requirements of low level fusion in sensor data streams. In order to acquire information, GSN introduces the concept of "virtual sensor". Any data provider, not only sensors can provide data to a GSN instance, as long as a virtual sensor configuration file (in the form of an XML) defines the the processing class, the sliding window size, the datasource and the output fields. GSN Servers can communicate between them, thus forming a network where information is collected, communicated, fused and integrated in order to produce the desired results.

As far as it concerns data acquisition, each GSN node can support input from more than one data stream. In order to combine the information, an SQL-like procedural language is offered. This allows to the user to define LLF functions in the following manner, while GSN takes care behind the scenes about crucial issues such as thread safety, synchronization, etc.

```
SELECT source1.S1 AS NumberOfPersons, source2.S2
    AS ExistenceOfSmoke
FROM source1, source2
WHERE NumberOfPersons>0 AND ExistenceOfSmoke="true"
```

In the example above, there are two data providers for one GSN Server.

[2] Global Sensors Network: http://sf.net/apps/trac/gsn/

This virtual sensor definition will produce events only in the case when the `WHERE` condition is satisfied. The example above demonstrates the concept of fusion: results are provided by taking into account the inputs from both the sensors.

Using the abovementioned approach, a variety of signal processing components can be integrated into one unifying architecture. For instance, the example above can be supported using a Body Tracker [19], [20] and a Smoke Detector, in order to produce alerts when persons are detected near smoke. In the same manner, results by processing components such as face detectors [21], unusual event detectors [22] can be fused in order to achieve the desired system functionality.

Communication with these processing components constitutes a difficult to tackle problem, mostly in terms of implementation since:

- Two worlds have to be brought together: image processing and the distributed systems. The former typically processes images or videos that are fundamentally different in nature from streaming data. As such, the component's input has to be modified in order to take into account streams that may lack synchronization, be erroneous, or overwhelming the network with data.
- Integration is hardly a trivial task. Prototypes in Matlab, algorithms implemented in C++ will need to communicate with the java-based GSN servers, running either locally or remotely. Web services/socket interfaces can be developed, imposing though, additional processing overheads.

Next, in order to allow GSN-GSN node communication, the information produced can be forwarded to other GSN nodes either in push or in pull mode. Data streams can be forwarded in order to be processed at remote GSN nodes. In order to forward these data streams, GSN supports subscriptions, once-off queries or even simple data forwarders. Data is forwarded using a RESTful approach (remote wrappers return XML over HTTP).

IV. Semantics and Reasoning

The Higher Level Fusion (HLF) layer of the proposed architecture consists of the underlying ontology which will be used as the backbone for performing the reasoning that is required for situation assessment. OpenLink Virtuoso[3] has strong support for Semantic Web technologies and provides services like RDF triple storage, a SPARQL compiler, rule-based reasoning and can expose relational data as RDF. Virtuoso acts as a federation layer between the relational and semantic data thus enabling their seamless integration. A block diagram of the HLF approach is presented in Figure 3.

A. *Ontology for high level fusion*

The ontology is the main information gathering point in the proposed architecture. It is defined according to the domain where the sensor network is placed and it defines the entities that are taking place in a situation, the events and the relations between them. All data that are produced by the processing modules and by the LLF nodes are eventually stored here where reasoning is performed in order to infer new knowledge which corresponds to events and situation assessment that cannot be performed at the LLF level.

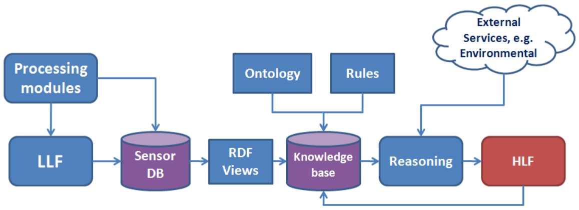


Fig. 3. The High Level Fusion process

In order to take advantage of research that has been conducted in the area of semantic enabled situation awareness, the current system uses as its core ontology the Situation Theory Ontology (STO) [13] which has been developed specifically for that purpose and is based on Situation Theory. In short STO is written in OWL and models the events/objects and their relationships in a way that can be extended using either OWL axioms and properties or in combinations with rules for supporting complex cause-effect relations that cannot be expressed in OWL alone. The main classes of STO are displayed in Figure 4.

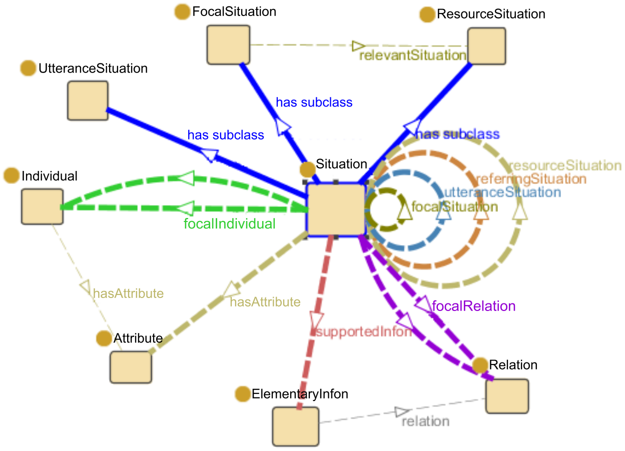


Fig. 4. Situation Theory Ontology with its main classes and properties

Situation is the central class. Instances of this class are specific situations. The second class is the *Individual* class, which is a counterpart of the individuals in situation theory. Similarly, *Relation* captures the n-ary relations. *Attribute* is a generalization of locations and time instants in situation theory. Instances of this class are attributes of individuals and situations. An attribute may have a dimension associated with it so the class *Dimensionality* represents this fact. The *Polarity* class has only two instances that correspond to the two possible values associated with a tuple, either that a given tuple holds or that it does not hold. Classes of STO

[3] Virtuoso Universal Server: http://virtuoso.openlinksw.com/

are related through a number of OWL properties. Situations are linked with four kinds of entities. First, the property *relevantIndividual* captures the individuals that participate in a situation. The property *relevantRelation* is used to assert that a given kind of relation is relevant to a given situation. Since situations are objects, they can have attributes of their own.

The STO is extended with classes and relations that correspond to the actual application scenario. In order to be able to use it in a real sensor fusion environment two additional ontologies are integrated. These are the "Time ontology"[4] which holds the timestamp of any concept instance that is stored during runtime and the "WGS84 Geo Positioning ontology"[5] which holds the latitude and longitude values of entities.

B. Mapping relational data to RDF

As explained in Section III, the data that are produced by the processing modules and by the LLF nodes are stored in the relational database that backs GSN. In order to forward these data to the ontology they have to be translated into semantic notations. This task is performed by the mapping layer which maps the relational schema to the semantic schema. There are two strategies for accomplishing this transformation, using either a push or a pull method.

The push method forwards the data to the ontology using semantic notation as soon as they are generated. This has to be implemented in the lower level layer with the semantic layer having a passive role in the process. The advantage of this method is that the transformations are executed fast and the ontology is always up-to-date. The disadvantages is that each lower level node will have to implement its own push method while there is the risk that the ontology will be populated with data even when no query is sent to the semantic layer.

The pull method transforms the relational data to semantic on request, i.e. during query time. Virtuoso supports this functionality through "RDF Views" where mappings, simple or complex, between relational database tables and ontology concepts and properties are defined. During query time, the mapping process is triggered and data are transformed on the fly. The advantages of the pull method is that the actual mapping is defined at the higher semantic level rather than the lower levels and that data are transformed on request so that the ontology will accumulate instances that are needed for the actual query evaluation. The disadvantage of this method is that it could lead to longer response times during queries.

In the proposed system, the pull method which utilizes Virtuoso's "RDF Views" is used as it is best to separate the semantic layer from lower level processing. This will make extension of the system by adding more sensors and sensor processing modules easier without having to mix the lower level processing level with the high level semantic fusion level.

Additional information can be integrated by the HLF layer in order to derive situations. Information coming from sources like environmental services which could provide details about the geographical locations of specific structures or points can be queried, as long as they expose I/O interfaces, and these data can be used for inferencing.

C. Reasoning

Reasoning uses the ontology structure and the stored instances in order to draw conclusions about the ongoing situations. Under this approach relations between classes like `rdfs:subClassOf`, properties like `owl:sameAs` and relations that are defined by experts in the ontology are utilized for inference. The knowledge base can further be extended by using rules to describe situations/events that are too complex to be defined using OWL notation only.

For utilizing the above, two different reasoners are integrated in the system, Virtuoso's internal reasoning engine and the Jena Semantic Web Framework[6]. Virtuoso provides inference capabilities using OWL only thus does not provide support for an external rule set. However, Virtuoso implements a powerful geospatial inference mechanism which is a crucial feature when dealing with sensor networks and sensor fusion. On the other hand, Jena supports external rules, advanced reasoning capabilities and seamless integration with Virtuoso where in this case, Virtuoso is treated only as a triple store. Both of them can be employed simultaneously on the same dataset and their activation is managed programmatically by the application.

Another issue that had to be dealt with is that sensor analysis modules and the LLF processes can generate a significant amount of data over time, enough to make the reasoning service not responsive. To overcome this problem, the solution that is proposed is to use a time window where reasoning is performed using only facts in a specific time interval. Jena and Virtuoso can support this type of inference by assigning a RDF triples to a specific "context" type, thus dealing with RDF quads.

The reasoning service can also use information and data from external services in order to use them in the inference process. In the proposed architecture, an Environmental Service provides the location information of sensors, events, waypoints and critical or important landmarks which are subsequently used for geospatial reasoning.

It should be noted here that in a real world application the different information sources that provide data to the proposed fusion architecture could make use of different ontologies for their purposes. In order to enable a seamless integration of these sources an ontology mapping process between the information source ontology and the HLF ontology would have to be defined.

V. Implementation and Experimental Results

Implementation is based mostly on two types of GSN-based servers that communicate with the sensor world, with a central node that monitors and controls the network. As illustrated

[4] Time Ontology in OWL: http://www.w3.org/TR/owl-time/

[5] Basic Geo (WGS84 lat/long) Vocabulary: http://www.w3.org/2003/01/geo/

[6] The Jena Semantic Web Framework: http://jena.sourceforge.net/

in Figure 1, the Leaf Nodes are the ones that consume the sensory information. These ones are in fact GSN-based servers configured to process incoming data from the sensor layer. In order to integrate Signal Processing components as in Figure 5, our implementation allows the use of standalone hosts. This happens because the components can either be resource-hungry in terms of processing capabilities to the extend that they utilize the whole processing power for their efficient operation or simply because the implementation is not portable, as in linux/windows-specific libraries.

A. System Implementation

In the scope of a proof-of-concept implementation and also as a testbed for our experiments, the snapshot of the architecture above that was implemented employs 1 computer node that hosts a Leaf Node and two processing components: a Smoke Detector and a Body Tracker, a camera that streams its feed using RTP, a node that hosts a Semantic Node that offers HLF capabilities and, finally, the Central Node that has the overall system supervision as already depicted in Figure 1.

Bottom-up, the system can be described as follows: First, the camera generates an RTP feed with its perception. The feed is processed by both the signal processing components (the Body tracker and the Smoke detector). The first component generates a stream containing at all times the `NumberOfPersons` detected by the component. The second one, simple however, is not as straightforward: it splits the image into particles and reports the `NumberOfSmokeParticles` detected. Then, the LLF Virtual sensor, according to the example presented in Section III fuses the data. Figure 5, illustrates the behavior of the BodyTracker.

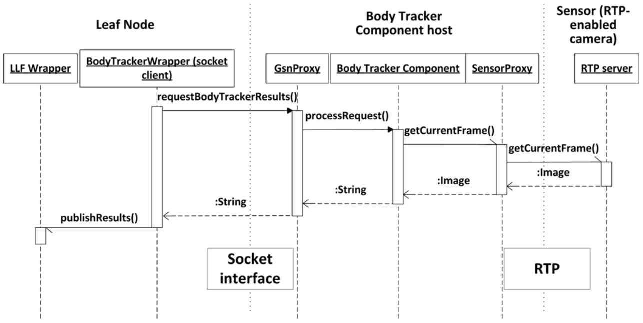


Fig. 5. Body Tracker Sequence Diagram

The sequence is initiated by the Body Tracker wrapper, which is polling for results the Body Tracker component host at fixed time intervals. The host, when it receives the `POLL` command – note that the component may support a number of functions such as `START`, `STOP`, `REQUEST_VIDEO_DETAILS` etc. – it returns a string containing a description in XML of the number of persons tracked. This description is of the following form:

```
<frame>
    <frame_counter>1534</frame_counter>
    <person n="1">
        <trackID>25</trackID>
        <x>587</x>
        <y>493</y>
        <w>79</w>
        <h>130</h>
        <quality_detection>2.8480</quality_detection>
        <quality_color>7.6249</quality_color>
        <quality_motion>5.1601</quality_motion>
    </person>
    <person n="2">
        <trackID>30</trackID>
        <x>1024</x>
        <y>491</y>
        <w>41</w>
        <h>83</h>
        <quality_detection>42.8867</quality_detection>
        <quality_color>6.7535</quality_color>
        <quality_motion>9.0603</quality_motion>
    </person>
      ...
</frame>
```

Note, next, that sampling the video source takes place asynchronously. This happens because the camera will be streaming at 25 fps, while for the needs of the LLF, 500 ms may suffice between two consecutive polls. In addition, tracking a person is a more demanding task than detecting it, since the former implies comparing consecutive frames for differences between them while for the latter processing single frames is enough. Therefore, the asynchronous behaviour is explained from the fact that the camera fps is not aligned with the messages per second that the component produces.

Under any circumstances, the LLF Wrapper receives a notification from the BodyTracker. The same information flow applies to the Smoke detection component. The LLF Wrapper generates results only in the case when the fusion conditions are true. Of course, more processing components and more complex fusion conditions can be added in the processing scheme hereby described.

For the HLF, the major difference is in the introduction of an ontology and reasoning procedures where more complex conditions for fusion can be applied. HLF only deals with higher level detections and events meaning that it communicates directly with the LLF and individual perception modules, pulling data in order to make inferences. An example situation where an alarm would be triggered by a rule, expressed either as a Jena rule or as a SPARQL construct query, which would state ”If smoke is detected near an object then raise an alarm” with objects being a Person or Vehicle. The classes that are inserted are

```
Smoke rdfs:subClassOf Non-RigidObject
Person rdfs:subClassOf RigidObject
Vehicle rdfs:subClassOf RigidObject
```

Alarms can then be retrieved with a simple SPARQL query issued to Virtuoso as

```
SPARQL
SELECT ?x WHERE {?x rdf:type STO:Alarm}
```

This example illustrates the use of the subclass inference in order to define situations that hold for a class of objects which is something that cannot be defined using the LLF layer only. In a similar way, the `owl:sameAs` property can be used for reasoning. The property can be evaluated during runtime. For example if two distinct detection instances is decided that are actually the same entity, e.g. if they have the same geographical locations, then by inference through `owl:sameAs` the rules and properties that apply to one also apply to the other thus enhancing the reasoning capabilities of the system.

B. Advanced inference

In order to demonstrate the inference capabilities of the proposed architecture, a scenario was set-up which makes full use of the data chain from low level sensor data to high level complex event detection and situation awareness/threat assessment. The example demonstrates how external services can be utilized in order to detect "critical" situations. For setting up a realistic usage scenario, the Environmental Service which exposes an interface where all interest points, e.g. buildings or sensors, positions in absolute lat/long coordinates are registered, is employed. These locations can be queried in real time by the HLF layer and are used for inferencing through geospatial reasoning. These data along with the lower level processing modules are used by the HLF layer in order to derive hypothesis about the criticality of a situation.

To demonstrate the use of sensors typically deployed in urban environment we selected video cameras which can be used to detect persons and also incidents, such as smoke, in an area of interest. Sensor locations are associated with a priori knowledge during reasoning. Detected smoke at a petrol station marks a significant threat to public safety based on a priori knowledge about the location and can be associated with a clear emergency action plan. A public square with a highly varying number of persons and smoke allows inferring various situations which can be controlled as necessary based on a security agenda. A specific public area can be subject to an event schedule which can provide significant semantic input to inferring on the threat level associated with detected situations. A gathering of locals celebrating with a bond fire on a public square is different from an unplanned detected smoke event during mid-day hours.

1) "Smoke detection in critical location" situation: Reasoning over triples in Virtuoso requires that the underlying triple store was populated with inferred event data from the smoke detector and body tracker. RDF views have been developed and used to load the underlying data from the sensor inference database. Using the scheduler component in Virtuoso we trigger a procedural logic which updates the triple store via RDF views from the sensor inference database and subsequently invoke on the reasoning.

In the example of smoke detection at the petrol station the relevant smoke event together with location data of the video camera is forwarded to emergency personnel with the relevant information. The reasoning process associates the smoke detection event with a criticality factor according to events modelled in STO and information coming from the ES. The process of modelling events in STO is described in detail in [23]. Figure 6 illustrates the model with instance data by example. We used the `STO:FocalSituation` class to mark significant situations that prompt action by security personnel. The class interlinks the data associated with events which we query via SPARQL and forward to emergency via a Web service call.

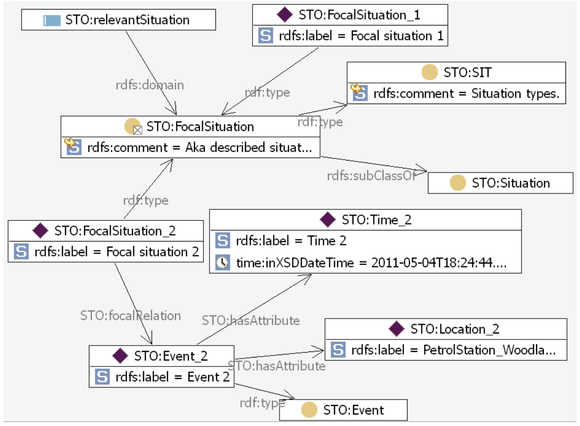


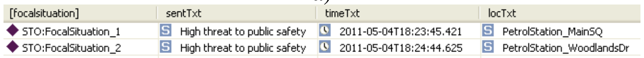
Fig. 6. Modelling of the "significant situations" in STO

The query of Figure 7a retrieves focal situations with their event related details such as time location and further textual descriptions and was run against the model above with the results shown in Figure 7b. Predefined SPARQL queries are invoked on in Virtuoso/PL.

Integration with emergency departments can be achieved by sending details on significant threats found in SPARQL result sets to a Web service endpoint exposed in an emergency department.

```
SELECT ?focalsituation ?sentTxt ?timeTxt ?locTxt
WHERE {
    ?focalsituation STO:focalRelation ?event .
    ?event STO:hasAttribute ?time .
    ?event STO:hasAttribute ?location .
    ?time rdf:type STO:Time .
    ?location rdf:type STO:Location .
    ?sentence rdf:type STO:Sentence .
    ?location rdfs:label ?locTxt .
    ?time time:inXSDDateTime ?timeTxt .
    ?sentence rdfs:label ?sentTxt
}
```

a)



[focalsituation]	sentTxt	timeTxt	locTxt
◆ STO:FocalSituation_1	High threat to public safety	2011-05-04T18:23:45.421	PetrolStation_MainSQ
◆ STO:FocalSituation_2	High threat to public safety	2011-05-04T18:24:44.625	PetrolStation_WoodlandsDr

b)

Fig. 7. a) Query for retrieving focal situations and b) the results of the query run against the modelled "significant situations"

VI. Conclusions

In this paper we have presented a framework for implementing intelligent information fusion in a sensor network environment. The framework deals with all aspects above the sensor layer i.e. it deals with perception modules integration, communication of the perception modules with GSN, low level fusion, high level fusion with integration of semantic description of information, communication with external services, situation assessment and alert generation. It is a generic framework that can be applied to any sensor network and it is not restricted to the security and surveillance area that was demonstrated. Illustrative examples of how the framework would be deployed in realistic scenarios have also been demonstrated where inference capabilities are shown. The core advantages of the proposed framework are it's extensibility with pluggable perception modules integration and the ease of defining rules for LLF and HLF, using SQL-like syntax and semantic rules/queries respectively.

Future work will be focused on implementing and integrating probabilistic reasoning, probably through fuzzy-DL, in order to drive inference as opposed to the deterministic reasoning that is applied up to now, thus enabling resolution of issues such as conflicts or missing detections in situation assessment.

Acknowledgment

Part of this work has been carried out in the scope of the MEDUSA project (Multi sEensor Data fusion grid for Urban Situational Awareness, co-funded by the European Defense Agency (MDS-MoM-A026RTGC-C-0003-VCS). The authors acknowledge help and contributions from all partners of the project.

References

[1] F. Calabrese, K. Kloeckl, and C. Ratti, "Wikicity: Real-time location-sensitive tools for the city," *IEEE Pervasive Computing*, July-September 2007.

[2] R. Murty, A. Gosain, M. Tierney, A. Brody, A. Fahad, J. Bers, and M. Welsh, "Harvard university technical report," Tech. Rep. TR-13-07, September 2007. [Online]. Available: http://www.citysense.net/

[3] X. Page and A. Kobsa, "Navigating the social terrain with google latitude," in *iConference*, Urbana-Champaign, IL, 2010, pp. 174–178.

[4] H. Sundmaeker, P. Guillemin, P. Friess, and S. Woelffl, *Vision and Challenges for Realising the Internet of Things*. European Union, March 2010.

[5] G. Lawton, "Machine-to-machine technology gears up for growth," *IEEE Computer*, vol. 37, no. 9, pp. 12–15, September 2004.

[6] K. Aberer, M. Hauswirth, and A. Salehi, "The global sensor networks middleware for efficient and flexible deployment and interconnection of sensor networks," Tech. Rep. 006, 2006.

[7] C. Sharp, E. Brewer, and D. Culler, "Hood: A neighbourhood sensor networks," in *Proc. of MobiSYS'04*, 2004.

[8] I. Chatzigiannakis, G. Mylonas, and S. Nikoletseas, "50 ways to build your application: A survey of middleware and systems for wireless sensor networks," in *Proc. of IEEE Conference on Emerging Technologies and Factory Automation, ETFA*, 2007.

[9] N. Dimakis, J. Soldatos, L. Polymenakos, P. Fleury, J. Curin, and J. Kleindienst, "Integrated development of context-aware applications in smart spaces," *IEEE Pervasive Computing*, vol. 7, no. 4, pp. 71–79, 2008.

[10] M. Eid, R. Liscano, and A. E. Saddik, "A universal ontology for sensor networks data," in *IEEE International Conference on Computational Intelligence for Measurement Systems and Applications*, Ostuni, Italy, 2007.

[11] H. Neuhaus and M. Compton, "The semantic sensor network ontology: A generic language to describe sensor assets," in *AGILE International Conference on Geographic Information Science*, Hannover, Germany, 2009.

[12] C. J. Matheus, M. M. Kokar, and K. Baclawski, "A core ontology for situation awareness," in *Proceedings of the Sixth International Conference of Information Fusion*, 2003, pp. 545–552.

[13] M. M. Kokar, C. J. Matheus, and K. Baclawski, "Ontology-based situation awareness," *Information Fusion, Special Issue on High-level Information Fusion and Situation Awareness*, vol. 10, no. 1, pp. 83–98, January 2009.

[14] J. Barwise, "Scenes and other situations," *Journal of Philosophy*, vol. 78, no. 7, pp. 369–397, 1981.

[15] M. R. Endsley, "Theoretical underpinnings of situation awareness: a critical review," in *Situation Awareness Analysis and Measurement*. Mahawah, NJ, USA: Lawrence Erlbaum Associates, 2000.

[16] E. G. Little and G. L. Rogova, "Designing ontologies for higher level fusion," *Information Fusion, Special Issue on High-level Information Fusion and Situation Awareness*, vol. 10, no. 1, pp. 70–82, January 2009.

[17] D. L. Hall and J. Llinas, *Handbook of Multisensor Data Fusion*. CRC Press, 2001.

[18] J. Llinas, C. Bowman, G. Rogova, A. Steinberg, E. Waltz, and F. White, "Revisiting the JDL data fusion model II," in *Proceedings of the Seventh International Conference on Information Fusion*, P. Svensson and J. Schubert, Eds., 2004, pp. 1218–1230.

[19] K. Bernardin, R. Stiefelhagen, A. Pnevmatikakis, O. Lanz, A. Brutti, J. R. Casas, and G. Potamianos, *Computers in the Human Interaction Loop*. Springer, 2009, ch. Person Tracking, pp. 11–23.

[20] A. Pnevmatikakis and F. Talantzis, "Person tracking in enhanced cognitive care: A particle filtering approach," in *The 18th European Signal Processing Conference (EUSIPCO 2010)*, Aalborg, Denmark, August 2010.

[21] N. Katsarakis and A. Pnevmatikakis, "Face validation using 3d information from single calibrated camera," in *The 16th International Conference on Digital Signal Processing (DSP 2009)*, Santorini, Greece, July 2009.

[22] K. Avgerinakis, A. Briassouli, and I. Kompatsiaris, "Real time illumination invariant motion change detection," in *Proceedings of the ACM Multimedia 2010 Workshop - 1st ACM ARTEMIS2010 International Workshop*, October 2010.

[23] M. M. Kokar, J. J. Letkowski, R. Dionne, and C. J. Matheus, "Situation tracking: The concept and a scenario." in *Situation Management Workshop: SIMA'08*. IEEE, MILCOM, 2008.