This document is published in:

Personal and Ubiquitous Computing, October 2012, 16 (7) , pp. 835-857.

DOI: http:10.1007/s00779-011-0450-9

Context-based scene recognition from visual data in smart homes: an Information Fusion approach

Juan Gómez-Romero · Miguel A. Serrano · Miguel A. Patricio · Jesús García · José M. Molina

Abstract Ambient Intelligence (AmI) aims at the development of computational systems that process data acquired by sensors embedded in the environment to support users in everyday tasks. Visual sensors, however, have been scarcely used in this kind of applications, even though they provide very valuable information about scene objects: position, speed, color, texture, etc. In this paper, we propose a cognitive framework for the implementation of AmI applications based on visual sensor networks. The framework, inspired by the Information Fusion paradigm, combines a priori context knowledge represented with ontologies with *real time* single camera data to support logic-based high-level local interpretation of the current situation. In addition, the system is able to automatically generate feedback recommendations to adjust data acquisition procedures. Information about recognized situations is eventually collected by a central node to obtain an overall description of the scene and consequently trigger AmI services. We show the extensible and adaptable nature of the approach with a prototype system in a smart home scenario.

Keywords Ambient intelligence · Computer vision · Data and information fusion · Context · Ontologies

J. Gómez-Romero (✉) · M. A. Serrano · M. A. Patricio · J. García · J. M. Molina
University Carlos III of Madrid, Avd. de la Universidad Carlos III, 22, 28270 Colmenarejo, Madrid, Spain
e-mail: jgromero@inf.uc3m.es
URL: http://www.giaa.inf.uc3m.es/

M. A. Serrano
e-mail: miguel.serrano@uc3m.es

M. A. Patricio
e-mail: mpatrici@inf.uc3m.es

J. García
e-mail: jgherrer@inf.uc3m.es

J. M. Molina
e-mail: molina@ia.uc3m.es

1 Introduction

Ambient Intelligence (AmI) envisions a future information society where users are ''proactively, but sensibly'' provided with services that support their activities in everyday life [1]. AmI scenarios depict intelligent environments capable of unobtrusively recognize the presence of individuals and seamlessly react to them [2]. To achieve this goal, AmI systems embed a multitude of sensors in the environment that acquire and exploit data in order to generate an adequate response through actuators. Different sensor and network technologies are usually applied: short-range (e.g. RFID, NFC); medium-range (e.g. Wi-Fi, Ultrawideband); and large-range (e.g. 4G cell networks). Nevertheless, visual sensors have received less attention, despite the large amount of interesting data that they can obtain from the environment. This gap is mainly due to two reasons: (1) processing visual data is computationally expensive and needs powerful equipment, including a considerable bandwidth to transmit captured images; (2) interpreting visual data is a complex task which may require the use of complex data models, as well as the incorporation of heterogeneous and maybe distributed data sources.

Data and information fusion (DIF) research area studies theories and methods to effectively "combine data from *multiple sensors* and *related information* to achieve more specific inferences that could be achieved by using a single, independent sensor'' [3]. DIF defines scene recognition (or

situation assessment) as "the estimation and prediction of structures of parts of reality (i.e., of the aggregation of relationships among entities and their implications for the states of the related entities)'' [4]. Scene recognition is central to AmI, since the system can selectively start proper services according to the user current situation and expected needs.

Classical techniques—those strongly based on observational data and a priori knowledge models—have proved to be insufficient to successfully recognize situations in unpredictable and complex scenarios [5]. A solution to overcome these problems has been to provide image-processing algorithms with additional knowledge not directly provided by the cameras; i.e., context knowledge. Context is defined as the ''set of circumstances surrounding a situation of interest that are potentially of relevance to its completion'' [6]. In video processing, context encompasses any external knowledge used to complete the quantitative data about the scene computed by straightforward image-analysis algorithms, including [7]: (1) the scene environment: structures, static objects, illumination and other behavioral characteristics, etc.; (2) the parameters of the recording: camera, image, and location features; (3) stored information: past detected events; (4) soft information provided by human users. In this sense, DIF involves fusion of data with different semantics and abstraction level, namely sensor data and context information, besides integration of data generated by distributed sensors.

In this paper we present an AmI framework that jointly manages visual sensor data and contextual information to support the construction of a symbolic description of the current scene. The knowledge model of the framework is an ontology [8], which allows representation and reasoning with these types of knowledge. Visual data is firstly processed at single smart cameras to achieve a local interpretation of the situation expressed with instances of the scenario ontology. Different procedures to obtain this local interpretation can be plugged into the framework; in this work, we depict the use of rules involving the terms of the ontology. The configuration o r t he b ehavior o f low-level image processing algorithms may be modified according to the local interpretation. Eventually, these interpretations are sent to a coordination agent, which manages the global view of the scene. Information fusion is performed at two levels: (1) heterogeneous data (contextual and sensor-based) is used to obtain local scene interpretations; (2) multi-camera information is gathered to build the overall picture of the scenario.

1.1 Contributions and structure of the paper

Most research works in the Computer Vision literature have only taken into account local context measures (computed from the pixel values surrounding an object) to accomplish scene recognition [9, 10]. These methods are hardly extensible to different domains, since they usually apply application-dependent heuristic calculations. In contrast, cognitive approaches [11, 12]—like the one presented here—propose to build a symbolic model of the world expressed in a logic-based language to represent environment objects and relations. Thus, the latter are more extensible, although they require the development of suitable data acquisition and information processing procedures.

A novelty of our proposal is the creation of an ontology-based cognitive model of visual data, context, and situations. The model allows symbolic manipulation of scene data, in contrast to the classical numerical proposals. Objects and relations—particularly, spatial and topological—are abstractly represented. Symbolic representations may lack the precision of numerical approaches, but we claim that this is not crucial in most AmI applications. As explained through this paper, a qualitative representation of the objects' relative positions is enough in most cases to obtain a convenient interpretation of the scenario. Moreover, low-level calibration of the cameras may not be necessary. In addition, the model could be populated by other sensor systems in addition to the visual sensor network (VSN)—as long as they are able to express acquired data in terms of the ontology, thus providing support to multi-modal AmI. Different activity interpretation procedures can be used within the framework as well.

Using ontologies as the knowledge representation formalism of the model provides several advantages. Ontologies are suitable for representing and reasoning with context and sensor data, especially when visual inputs are to be interpreted [13]. Besides, ontologies have strong underpinnings in Description logics (DL) and are widely supported with standards languages (e.g., the Ontology Web Language (OWL) [14]) and tools. Last but not least, ontologies promote knowledge exchange and reusability—our framework offers a set of high-level ontologies that represent the basic semantics present in any AmI application to be specialized in particular scenarios.

It is important to highlight that the hierarchical architecture implemented in the framework allows task distribution among the cameras while minimizing the amount of exchanged information. This reduces computational requirements and bandwidth consumption of the system.

The remainder of this paper is structured as follows. In Sect. 2, we overview some related work pertaining to the use of formal context knowledge models in Information Fusion and Computer Vision, whereas in Sect. 3 we describe the issues of VSNs with a particular focus on AmI applications. In Sect. 4 we introduce the architecture of the framework. In Sect. 5, we detail our proposal of a multi-

layer model for representing and reasoning with perceived and contextual information; structure, spatio-temporal knowledge management, and reasoning procedures are described. Section 6 includes an AmI test scenario where the framework has been applied, along with some implementation details. Section 7 briefly introduces how the framework integrates scene descriptions obtained by single cameras. Finally, the paper concludes with a discussion on the results and plans for future research work.

2 Related work

Most works in the literature on general AmI systems tackle the problem of representing and exploiting context information. Nevertheless, few of them deal with visual information, as previous surveys show [15]. Currently, there are some promising approaches resulting from the synergies between AmI and the video-surveillance research area—a typical application domain of video-based systems, since both of them are concerned with the monitoring of complex environments. Some of these proposals combine multimodel information to achieve scene recognition [16], although they usually apply numerical techniques, which are less flexible and sometimes hardly extensible. Not surprisingly, the need of integrating heterogeneous sensor/context data and the existence of several distributed data sources has resulted in the application of DIF paradigms and techniques to the problem [17, 18].

In this section, we will focus on research works that apply ontologies to model situations recognized from visual data in AmI applications. One of the most notable contributions is presented in [19]. The authors describe some issues and methodologies for the creation of ontologies supporting AmI applications focused on surveillance and security. Rules are used to create expressions to detect complex events. Some of the main problems that appear in this kind of systems are tackled: event representation, spatial reasoning, uncertainty management, etc. Previously, other approaches—such as PRISMATICA [20]—had also studied the problems that appear in surveillance-related AmI applications, though they are more focused in the management of the VSN, instead of the possible contributions of this technology to AmI.

More recently, ontologies and visual inputs have been combined to detect abnormal behaviors, also in the surveillance domain. We shall mention the research works in [21, 22], which describe a multi-agent knowledge-based system to characterize and detect abnormal situations in surveillance areas. Interestingly enough, this proposal incorporates imprecise and vague information in the knowledge representation. Multi-agent systems are also used in GerAmi, an AmI environment to supply care and support to elderly people which strongly relies on RFID and Wi-Fi technologies [23].

Insofar as general and high-level ontology-based scene recognition is concerned, an ad hoc proposal for scene interpretation based on DL is presented in [24]. The paper shows how the reasoning features of the Renamed Abox and Concept Expression Reasoner (RACER) reasoning engine provide functionalities that support scene recognition. The approach is hardly extensible, but it illustrates the expressivity of DL for such tasks, as well as the existence of appropriate tools. DL are also used for modeling and reasoning about complex situations in [25]. This paper discusses the features required to an ontological model for context-based situation recognition from sensor data, as well as the architectural and implementation details of the corresponding AmI applications. The proposed representation is very similar to the upper levels of our model, but we also solve the grounding problem [12]; i.e., we bridge the gap between real-world signals and high-level symbolic representations. This problem is not tackled in [25], which does not explain how high-level ontological descriptions are obtained from camera data.

Alternatively, probability theories, such as Markov logic networks, have been also used in fusion-based object and scene recognition. The hybrid approach presented in [26] successfully combines low-level image processing and high-level situation description. In contrast, our framework emphasizes the role of the cognitive knowledge model, which facilitates reasoning and human interaction, while promoting knowledge reuse. However, we require the creation of proper abduction procedures, which may be difficult in some cases, and assume the existence of accurate tracking and identification modules, which is not always possible.

More details about the structure of the knowledge model described in this paper can be found in a previous research work [27]. In that paper, we introduced an ontology-based framework for cognitive surveillance. In the present paper, we specifically focus on the problems that appear in AmI applications, and explain the development issues that arise when implementing the abstract framework in this domain. In addition, we study the qualitative representation of spatial properties and the reasoning procedures involved, and how a rough calibration can be enough in several AmI applications; e.g., smart homes.

3 Data and information fusion in visual sensor networks for AmI

A VSN involves the deployment of a certain number of cameras in a wide area—probably with overlapping fields of view—which acquire visual data from the environment.

Necessarily, suitable procedures to interpret data captured by single cameras must be developed, in order to obtain an integrated and high-level view of the situation. The DIF research area studies the problems arisen from the combination and interpretation of multiple data sources (maybe at different abstraction levels), and specifically those that appear when data sources are video cameras. Fusion processes are classified according to the JDL (Joint Directors of Laboratories) model, the prevailing theory to describe fusion systems [28, 29]. JDL classifies fusion processes according to the abstraction and the refinement of the involved entities. The canonical JDL model establishes five operational levels in the transformation of input signals to decision-ready knowledge, namely: signal feature assessment (L0), entity assessment (L1), situation assessment (L2), impact assessment (L3), and process assessment (L4).

Low-level data fusion, corresponding to JDL L0 and L1 levels, designates procedures aimed at pre-processing sensor signal and estimating the properties of isolated objects. High-level information fusion procedures, corresponding to L2 and L3, aim at obtaining a description of the relations between the objects in the perceived scenario. These relations are usually expressed with interpretable symbolic terms (e.g., actions, intentions, threats), instead of the usual numerical measures (e.g., density functions, movement vectors) calculated in L1. L4 tasks are aimed at planning and performing procedures to improve the whole fusion process, from low-level data acquisition to high-level situation assessment.

Below, we summarize some of most important problems that appear in VSN-based AmI systems at each JDL level, and briefly present the techniques, algorithms, and procedures that are considered in our framework to solve them.

3.1 Level 0

3.1.1 Camera location

The first decision in a multi-camera system is the physical installation of the sensors. The amount and the situation of cameras have a great impact on system cost and capabilities. It is convenient to arrange the cameras in a configuration that minimizes object occlusions and maximizes overlapping between fields of view, though this is not always possible.

Our framework is independent from the position of the cameras, but will be more effective when more overlapping cameras are used. Camera handover mechanisms—to share information captured by a camera when an object moves to the field of view of an adjacent camera—are not currently supported in the framework, but could be implemented by relying on the ontological model as in [30].

3.1.2 Camera calibration and data alignment

Information in a VSN must be aligned to a common reference frame. Camera calibration, or common referencing, is the process to calculate the homography matrix that converts from the local coordinates of each camera to a global coordinate space. Calibration can be an off-line procedure (based on the correspondence of the position in the camera plane and in the global plane between of predefined landmarks) or an on-line procedure (based on the analysis of in-use system data; e.g., correspondences between automatically detected corners, edges, etc.).

Numerical calibration of the cameras may not be required in our framework. As explained in Sect. 4, the smart cameras only interchange high-level descriptions of the perceived situation in terms of topological relations between entities; for example, *a person is close to couch_1*. If we assign the same identifier *couch_1* in every camera that is detecting this object, a rough correspondence between their view fields is established. This correspondence may serve as an implicit calibration to align data in the central node. Obviously, this approach is too far to completely solve the problem of calibration, but may be sufficient in several AmI domains where high precision is not required.

3.2 Level 1

3.2.1 Object detection

The most elemental information that can be extracted from a video sequence is that of the discovery of moving objects. There are various techniques for object detection: (1) temporal differencing, based on the calculation of the pixel-by-pixel difference between consecutive frames; (2) background subtraction, based on the calculation of the difference between the current frame and a predefined background image; (3) statistical methods, based on the difference of additional features extracted from the image; (4) optical flow, based on the computation of the flow vectors of moving objects; and (5) classification, based on the identification of a pattern in the image with trained classifiers.

Object detection is not trivial, since in most cases the conditions of the watched environment change. For example, changes in the illumination and the shadows of the objects during daytime (especially in outdoors applications) and moving objects that become static must be taken into account. Object detection is performed in the framework by the tracking layer, which relies on a tracking procedure. The framework can be configured to use different tracking algorithms.

3.2.2 Object tracking

Detected objects must be tracked over time; i.e., the system must segment the moving objects and assign consistent labels during their complete lifecycle. Specifically, a track is defined as a set of groups of connected pixels that represent a moving object with some properties: size, color, speed, etc. In the simplest case, a track includes a single group of connected pixels. Tracking is defined as the estimation of the number of objects in a continuous scene, together with these properties: locations, kinematic states, etc. Object tracking has been tackled by applying statistical prediction and inference methods, such as Kalman or particle filters, adapted to visual data association.

The tracking layer of the framework performs the complete tracking procedure, as explained in Sect. 4. Estimation techniques are very sensitive to the particular conditions of the scenario, and therefore they may be insufficient in some applications. The incorporation of context knowledge has been regarded as essential to deal with complex scenarios with occlusions, illumination changes, and object deformations. In our framework, object and situation information at levels 2 and 3 (obtained by applying context knowledge) is used to change the parameters of the tracker according to the current scenario and past events.

3.3 Level 2

3.3.1 Classification

Object identification and activity recognition are two fundamental classification tasks that must be performed in an AmI application (VSN-based or not). Object identification aims at determining the type of a tracked object; e.g., person, bottle, etc. Thus, it can be considered halfway between L1 and L2. In the framework, we use a priori rules to classify objects according to their features—mainly the size (Sect. 6.2). This approach should be extended with more advanced techniques and/or machine learning enhancements, in order to automatically classify tracks according to other features: color, histogram, etc.

Activity recognition, in turn, aims at discerning that an activity is taking place. Usually, two types of activities are distinguished: basic activities—i.e. simple activities that cannot be decomposed into more simple actions (e.g., walking), and composite activities—i.e., activities that are the result of various simple actions (e.g., laying the table). Activity recognition is an open problem in general applications, since it requires systems to develop cognitive capabilities close to human understanding. In this paper, we use pre-defined rules to identify activities from moving object properties and context information. The main strength of rules is that we can express almost any condition by using terms defined in the ontological model. In contrast, at its current state, they must be manually created, which requires a considerable effort. In addition, other methods could be applied in the framework in combination to rules to identify complex activities. The use of the symbolic models facilitates the integration of these different techniques, since any procedure can be plugged into the framework as long as its output (i.e., recognized activities) is expressed with the same ontology language.

3.3.2 Model construction

Scene interpretation consists in obtaining a symbolic model of the scene activities. Ontologies support the definition of a formal vocabulary to create these symbolic models. This vocabulary includes the terminological axioms (i.e., axioms about classes and relations) that are used to delimit the possible realizations of the model. In DL nomenclature, the set of axioms defining concepts is the TBox of the ontology, whereas the set of axioms defining properties is the RBox of the ontology. The concept and relation instances of the ontology are defined with axioms about individuals, which represent the evolution in time of the scene tracks, objects, situations, etc. These axioms about instances of the ontologies compose the ABox. Essentially, scene interpretation is a model-building procedure in which instances of the concepts and relations defined in the scene vocabulary are created. We explain in detail the model construction process in the framework in Sects. 4 and 5.

3.4 Level 3

3.4.1 Situation assessment

Level 3 focuses on the estimation of the impact of a situation of the application domain. In other words, situation assessment is the process of detecting and evaluating particular situations that are of special relevance to the scenario because they relate to some type of threatening, critical situation, or any other special world state. This JDL level includes procedures for the identification of abnormal and hazardous situations, which is especially relevant in some AmI domains; for example, Ambient Assisted Living applications require implementing proper mechanisms to react to an emergency situation if the user does not follow the normal sequence of activities, falls down, or abruptly interrupts an ongoing activity. The framework applies the same rule-based mechanism explained for Level 2 tasks: rules with terms of the lower abstraction level are used to create instances representing information at this level.

3.5 Level 4

3.5.1 Process enhancement

Process enhancement—also known as active fusion—is aimed at the modification of the data acquisition and processing procedures after DIF to enhance results quality. Generally speaking, process enhancement consists in improving a fusion procedure by using feedback generated at a more abstract level. For instance, the behavior of a tracking algorithm can be changed once a general interpretation of the scene has been inferred; if the system recognizes that an object is moving out of the camera range through a door, the tracking procedure could be informed to be ready to delete this track in the near future. As previously mentioned, the framework includes a general mechanism to generate recommendations for the tracking procedure based on rule triggering. In their basic form, these recommendations are direct manipulations of the parameters or the data stored by the tracker, as we exemplify in Sect. 6.

4 Framework architecture

The architecture of our framework is depicted in Fig. 1. The schema shows the two types of nodes that are defined in the VSN: the smart cameras and the central node.

Smart cameras are video cameras able to perform DIF tasks. Cameras capture data, which is processed by a low-level tracker. Tracking information is then introduced into the abstract scene model as ontology instances. Reasoning rules are activated as a result of the changes of scene objects detected by the tracker. Eventually, as a result of the model-building process, these rules will create instances corresponding to situations. Situations detected by single smart cameras are sent to the central node. The contents of the messages between smart cameras and the fusion node are encoded with the same ontology used for the smart camera scene model. The central node processes the situations detected by the cameras in order to obtain a more complete view of the scene.

Smart cameras process data at two logical levels: (1) the tracking layer; (2) the cognitive layer. First, each camera is associated with a process that acquires video frames. Next, the tracking sub-system sequentially executes various image-processing algorithms to detect and trace all the targets within the local field of view. The tracking layer is arranged in a pipelined structure of several modules, which correspond to the successive stages of the tracking process [31, 32]: (1) detection of moving objects; (2) region-to-track multi-assignment; (3) track initialization/deletion; (4) trajectory analysis.

Tracking data is introduced into the cognitive layer to initiate more complex high-level information fusion procedures. Smart cameras implement an a posteriori schema for context information exploitation [5]. This schema proposes the implementation of a processing layer on top of the tracking procedure. In this layer, abstract ontologies are used to describe abstract entities. The tracking layer and the cognitive layer communicate through an interface, which offers methods to revise the ontological model in the update and initialization/deletion steps. In the next section, we describe the structure of the ontologies and the processes to create ontology instances in the cognitive layer.

Communication between the smart cameras and the central node is started when a new situation is detected—i.e., when a new instance of the concept that represents a

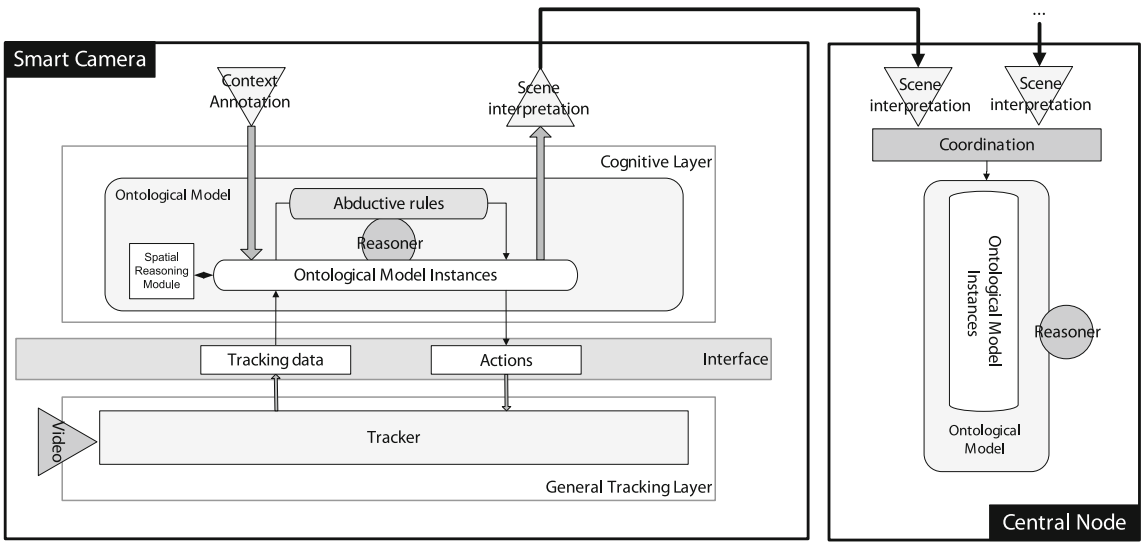


Fig. 1 Architecture of the framework: smart cameras and central node

situation is created in the symbolic model (see next section). The detected situation is sent to the central node, expressed in the suitable situation ontology. The use of a formal ontology to communicate situation information facilitates the incorporation of heterogeneous cameras—or even other sensors—to the system, as long as they are able to use the same situation ontology to communicate information.

The central node gathers camera information to build a unified view of the scene. This unified view is represented with instances of the same ontologies used for smart cameras. The combination of local camera information has been implemented with a rule-based mechanism, as explained in Sect. 7.

5 Knowledge representation and reasoning

The ontological model of the cognitive layer encodes the context information provided by human users, the perceptions acquired by the camera, and the model obtained after reasoning processes. To manage these three types of scene knowledge, we propose a set of layered interrelated ontologies organized according to the abstraction layers defined by the JDL fusion model.

The overall structure of the ontologies is depicted in Fig. 2. We have distinguished between the general knowledge (i.e., very abstract knowledge that is common to any VSN-based application) and the specific knowledge (i.e., knowledge specific to a concrete AmI application). Accordingly, we provide various upper ontologies that contain terminological axioms defining basic concepts and relations, namely TREN, SCOB, ACTV, IMPC and RECO.

The concepts and relations defined in these upper ontologies must be specialized in each application. For example, in the figure we show how the new ontology SMARTHOME has been defined by refining the concepts included in the general ontologies. The purpose of this separation is to provide final application developers with a general knowledge frame with well-defined building blocks, in such a way that they only need to extend it to model new scenarios. For example, the SCOB ontology defines the OccludingObject class, which—for the sake of simplicity—is a type of StaticObject. In SMARTHOME, we define Couch as a subclass of OccludingObject, and consequently it inherits all its properties.

It is interesting to note that these ontologies are closely related between them. In fact, they represent the transformation from low-level tracking data to high-level situation knowledge. An ontology of an upper abstraction level is linked (or *grounded*) to an ontology of a lower abstraction level. Accordingly, the ontology for scene objects defines the property hasAssociatedTrack to associate instances of scene objects to instances corresponding to track data. Thus, information at object level is described in terms of objects and objects' relations, but objects are associated to the tracks obtained by the tracking layer. Similarly, a more abstract ontology is defined to represent scene situations; these situations are grounded to the involved objects represented in the scene objects ontology.

Contextual objects knowledge is introduced into the model as instances of the proper ontologies, which is known as *scenario annotation*. Annotations include object position and size, possible occlusions, enter and exit zones, or any other convenient contextual knowledge. This zero-point knowledge is used in the reasoning process that is activated when moving objects are detected in the scene.

Additionally, new reasoning rules (*deductive* or *abductive*) are introduced into the knowledge model (see Sect. 6). The combined use of ontology specialization and rules allows the definition of very general rules that are triggered with objects of the classes and the subclasses. For instance, a general rule to detect proximity between a TrackedObject and an OccludingObject will be fired not only with direct instances of these concepts, but also with instances of their subclass; e.g., Person and Couch, respectively. In this way, we can describe a new entity as a subclass of many existing classes, and consequently define its behavior as the composition of the behavior of its superclasses.

In the remainder of this section, we explain the structure of the terminological part of the general ontologies provided with the framework. Next, the nature and the implementation of reasoning procedures within the representation model are discussed.

5.1 JDL-based knowledge representation

5.1.1 Tracking data (L1)

The basic TREN (*TRacking ENtities*) ontology includes axioms about concepts and relations to symbolically represent data obtained by the low-level fusion algorithms (see Fig. 3). Instances of this ontology are created as a result of the initialization and the update stages of the tracking procedure, when new tracks are created or track properties change.

The core concepts in TREN are Frame and Track. A frame is identified by a numerical ID and is marked with a time stamp using an OWL-Time DateTimeDescription [33]. Regarding tracks, it is necessary to represent their temporal evolution, and not only its state in a given instant. We want to keep all the information related to a track during the complete sequence (position, size, velocity, etc.). Therefore, we must associate to each track various sets of property values that are valid only during some frames. We have followed an ontology design pattern

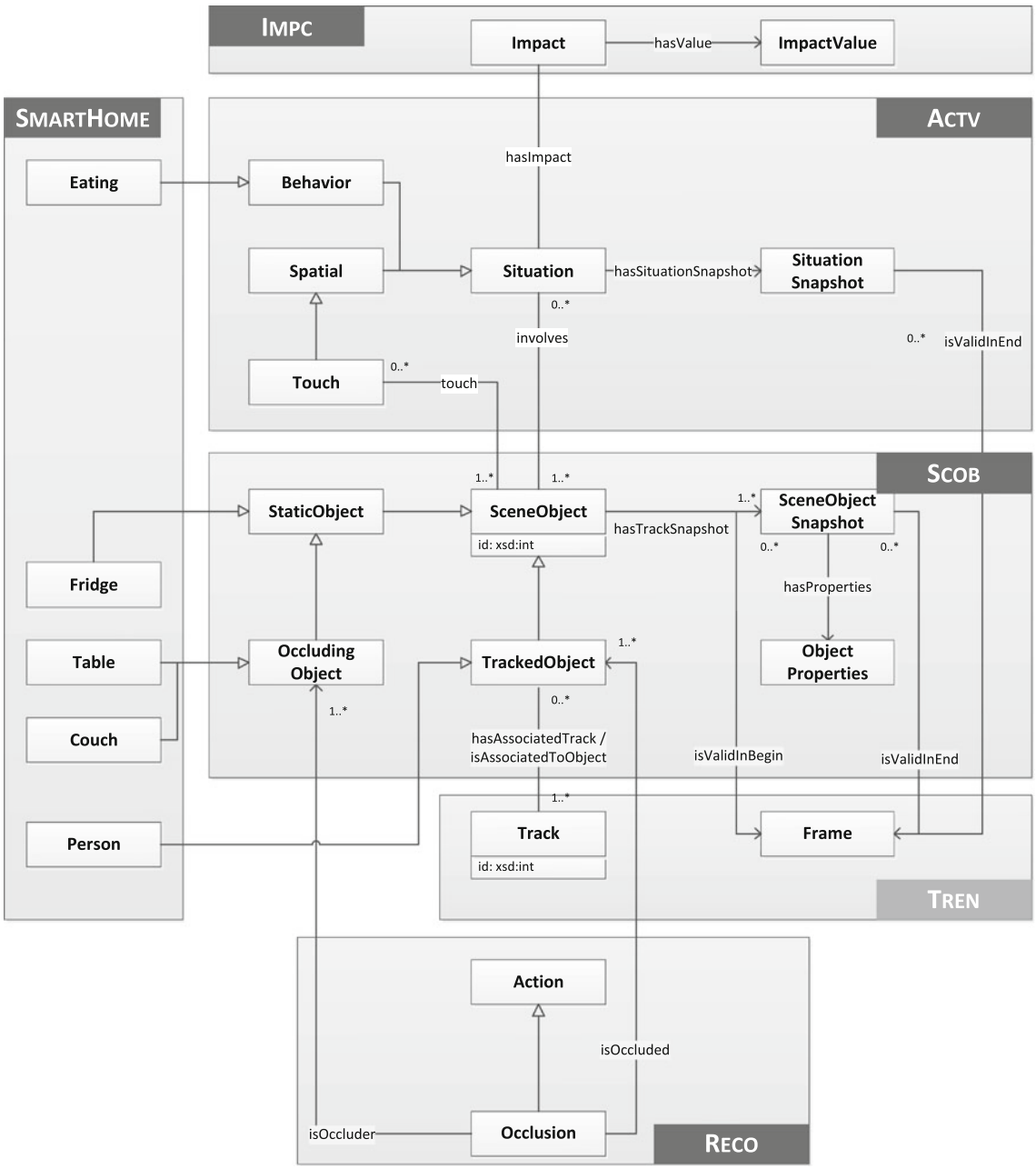


Fig. 2 UML excerpt of the high-level ontologies: main concepts and grounding

proposed by the W3C Semantic Web Best Practices and Deployment Working Group to define ternary relations in OWL ontologies [34]. We have associated a set of TrackSnapshots to each Track. Each TrackSnapshot, representing track feature values, is asserted to be valid in various frames.

Additionally, track properties must be defined as general as possible, in such a way that they can be easily extended. To solve this issue, we have followed the *qualia* approach, used in the upper ontology DOLCE [35]. This modeling pattern distinguishes between properties themselves and the space in which they take values. This way, we have associated properties to ActiveTrackSnapshots, such as TPosition or TSize. TPosition is related with the property TpositionValue to a single value of the TPositionValue-Space. A 2DPoint is a kind of TPositionValueSpace. The definition of geometrical entities has been developed according to the proposal described in [36], which defines primitive concepts such as *Point*, *PointSet*, *Curve* (as a subclass of *PointSet*), or *Polygon* (a kind of *Curve*).

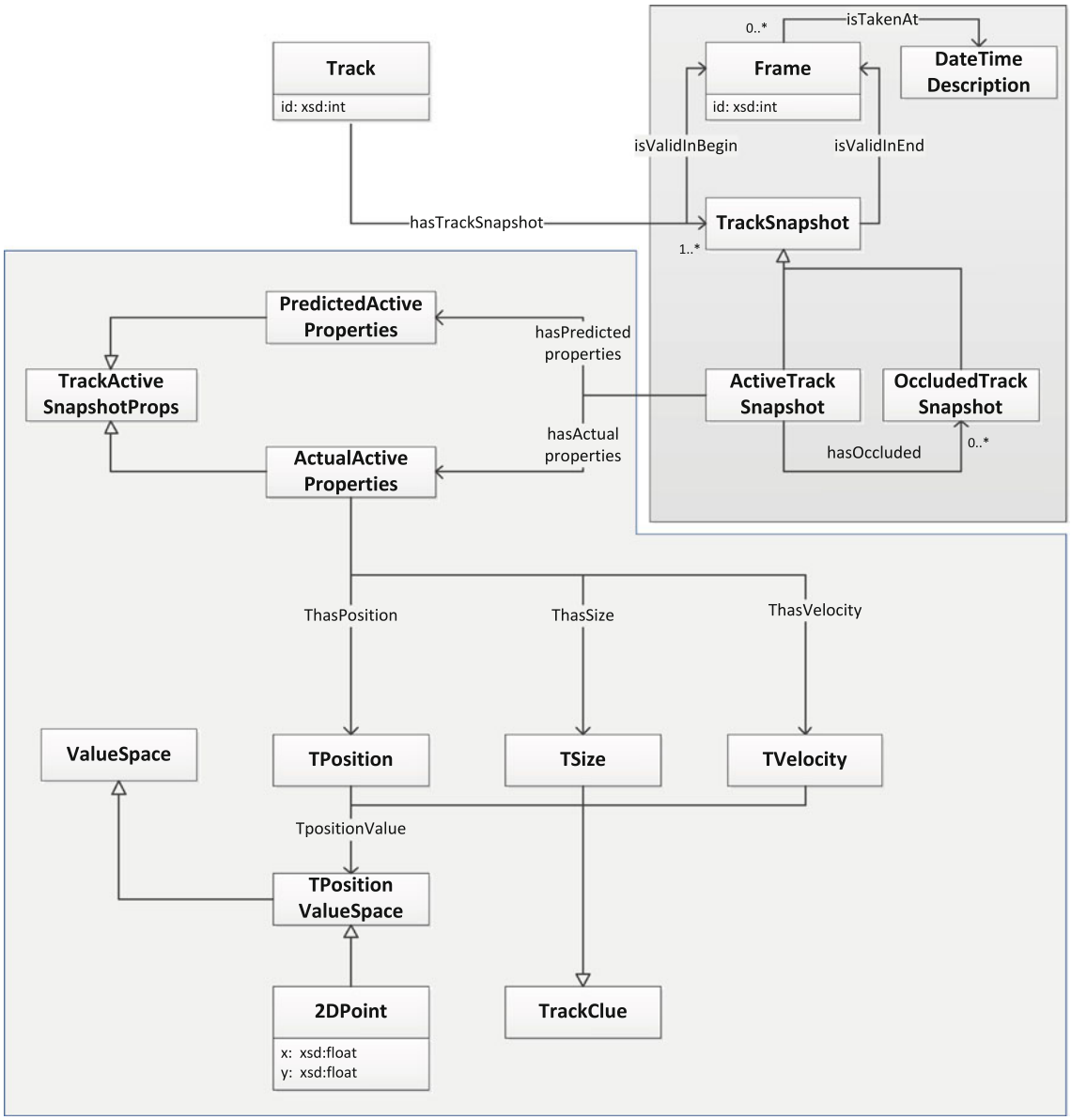


Fig. 3 UML excerpt of the TREN ontology: representation of track properties and track snapshots

Additional axioms or rules to calculate complex properties of tracks (e.g. distances), as well as spatial relationships (inclusion, adjacency, etc.), could be considered and created in TREN. This kind of reasoning is described in detail in Sect. 5.3.

5.1.2 Scene objects (L1–L1/2)

The SCOB (SCene OBjects) ontology includes axioms about concepts and relations to symbolically represent real-world objects and their correspondence with detected tracks. For example, a track with suitable properties would be inferred to correspond to a person (possibly by applying context information in a classification procedure). Thus, a new Person instance is created in SCOB and connected to the track instance of TREN.

The SCOB ontology includes both static and dynamic objects. Static objects (class StaticObject) are scene objects defined a priori. Not surprisingly, most of contextual entities are instances of the StaticObject class. Dynamic objects (class TrackedObject) are scene objects detected during the functioning of the system. Instances of dynamic objects are created as a result of correspondence and reasoning procedures, as depicted in Sect. 6. StaticObject and TrackedObject are subclasses of SceneObject, the root concept in the SCOB ontology. SceneObjects

have properties; e.g. position, illumination, behavior, etc., which may vary in the sequence. To represent properties, we have applied the same combined snapshot/qualia approach as in the TREN ontology. It can be noticed that tracked object property values may be different from the property values of the associated track snapshots, but most of these object property values will be easily inferred from the associated track.

5.1.3 Activities (L2)

The ACTV (ACTiviTies) ontology includes axioms about concepts and relations to describe relations between objects that last in time. This ontology includes axioms involving concepts and relations to describe simple and complex activities. For convenience, these relations have been reified as classes descending from a top concept named Situation. We have introduced as well some properties to establish the temporal duration of the situations that follows the same pattern based on snapshots described for the lower layers of the model.

As mentioned in Sect. 3, this ontology is also used for communication between smart cameras and the fusion node of the architecture. Simple activities, expressed as instances of the ACTV ontology, are sent to the central node to be combined with other inferences to eventually detect a complex situation. Complex situations are also introduced as instances of ACTV, in this case in the local instantiation of the ontology managed by the central node.

5.1.4 Impacts and threats (L3)

The IMPC (IMPaCts) ontology has been defined on top of the ACTV ontology. This ontology includes relations to associate situations (instances of the Situation concept) and impact evaluations (instances of the IMPC concept Impact). This value is a simple numerical assessment or, more probably, a complex expression suggesting or predicting future actions. Impact and threats are the most application-dependent knowledge of the ontological model; therefore, they must be conveniently specialized in a given domain. To allow the representation of different impact evaluations, the qualia approach has been also applied in this ontology.

5.1.5 Process assessment (L4)

Process assessment knowledge includes certain meta-information about the functioning of the framework that is used to improve it. Accordingly, the RECO (RECOmmendations) ontology includes concepts and relations to represent actions that must be carried out to modify either the instances of the ontologies or the behavior of the basic tracking algorithm.

The main concept in RECO is Action, which abstractly represent any action that can be understood and carried out by the framework. In the simplest case, these recommendations are instances generated as a result of rule triggering. Once a recommendation is created, it is synchronously executed, since delaying the modification may be unproductive or error-prone. A more complex policy for handling recommendations could be developed; as a matter of fact, the basic mechanism could be extended to implement a priority queue to asynchronously retrieve, interpret, and carry out the procedures specified by Action instances.

5.2 Deductive and abductive reasoning

Standard ontology reasoning procedures are performed within the ontological ontologies to infer additional knowledge from the explicitly asserted facts. To name some of them, the inference engine supports tasks such as classification (i.e., to determine the class hierarchy of an ontology) and instance checking (i.e., to determine the classes which an instance belongs to).

Ontology Web Language standard does not directly support deductive rules, but several extensions have been proposed. One of the most extended is SWRL (Semantic Web Rule Language) [37], which allows deductive inference within OWL ontologies. Rule-based formalisms can be used with limitations, since reasoning with models combining rules and OWL is decidable only under certain safety restrictions [38]. Deductive rules are used to maintain the consistency of the ontology and to explicitly assert axioms involving existing instances. An example of deductive rule would be "the position of a tracked object must be the same as the position of the last associated track snapshot".

Monotonicity of ontology languages forbids adding new knowledge to the models while reasoning, which is required in scene interpretation. Actually, scene interpretation is a paradigmatic case of abductive reasoning, in contrast to the DL deductive reasoning [39]. Abductive reasoning is defined as a form of reasoning that takes a set of facts as input and draws a suitable hypothesis that explains them—sometimes with an associated degree of confidence or probability. This type of reasoning is also called Inference to the Best Explanation (IBE). Visual data interpretation can be regarded as an IBE process: we want to figure out what is happening in the scene from the observed and the contextual facts. In terms of the knowledge model presented in the previous section, scene interpretation can be seen as an abductive process to generate ontology individuals of a higher-level ontology from instances of a lower level ontology.

Abduction is not directly supported by DL ontologies, but it is simulated in some reasoning engines by defining non-standard inference rules. These rules allow the creation of new instances in the consequent, which is forbidden in standard rules to satisfy the safety condition. RACER [40], which is used in this work, supports abductive reasoning through extension rules that create or modify instances of the ontology representing scene interpretations. Please note that these rules do not directly support representation of uncertain knowledge. As shown in the next section, uncertainty management may not be essential in simple AmI scenarios, but it must be considered in more complex domains involving scene recognition [41]. This remains as a prospective direction for future work.

In our model, we have two types of non-standard rules: bottom-up rules and top-down rules. Bottom-up rules are used in scene interpretation, and as mentioned, they obtain instances of upper-level ontologies from instances of lower-level ontologies. For instance, some rules could be defined to identify objects from track measures; i.e., to obtain instances of the scene objects ontology from instances of the tracking data ontology. An example rule is "create a person instance when an unidentified track larger than a predefined size is detected inside a region of the image".

Top-down rules are used to create instances of the Action concept in RECO from the current interpretation of the scene, the historical data, and the predictions. Top-down rules may result in corrections to the low-level fusion procedure: tracking parameters, data structures, etc. A simple rule would recommend "to ignore a track associated to a person which is inside an area previously annotated as a mirror".

5.3 Spatial reasoning

One key aspect of our model is representation and reasoning with qualitative spatial properties. Ontologies do not directly support spatial reasoning, which has given rise to the development of joint approaches that incorporate suitable additional constructors [42, 43]. One of the most used is the Region Connection Calculus (RCC), a logic theory for qualitative spatial knowledge representation and reasoning [44, 45]. RCC is an axiomatization of certain spatial concepts and relations in first order logic. The basic theory assumes just the primitive dyadic relation C(x, y)—read as *x connects with y*, being x and y spatial regions. This relation is reflexive and symmetric.

The most used version of RCC is RCC-8, which defines eight relations: DC (is disconnected from), EC (is externally connected with), PO (partially overlaps), TPP (is a tangential proper part of), NTPP (is a non-tangential proper part of), TPPi (inverse of TPP), NTPPi (inverse of NTPP) and EQUAL. These relations have been proved to compose a jointly exhaustive and pairwise disjoint set. Similar sets of one, two, three, and five relations are also defined (respectively, RCC-1, RCC-2, RCC-3, and RCC-5).

RACER includes support for RCC through the activation of an extended reasoning layer, namely the *substrate*, which allows the use of RCC predicates in representations and queries while preserving RCC semantics. In addition, user-defined relations can be extended with RCC semantics; in the simplest case, this means to make a user-defined relation equivalent to a RCC predicate.

Region Connection Calculus predicates (and RCC-equivalent user-defined properties) must be instantiated in the knowledge base. This implies the creation of an instance of a RCC relation between two instances to represent that the corresponding scene entities are disconnected, partially overlapping, etc. To calculate the instantiation of RCC properties, calculations must be performed to obtain the distance between the bounding boxes of two tracks (or objects). This can be alternatively achieved: (a) by using supported lambda calculus expressions to be executed by RACER [46]; (b) by performing a topological analysis in a pre-processing step [47]. Our experiences prove that the second approach is more appropriate; otherwise the performance of the reasoning process is seriously compromised, because RCC properties of moving objects must be often recalculated. Additionally, pre-processing facilitates the implementation of additional optimizations and the use of third-party tools supporting topological calculations.

The framework includes a pre-processing module to instantiate RCC properties. This module is executed when a new contextual object is annotated in the scenario (infrequent) or when a tracked object changes its position (very frequent). The module is based on the OpenGIS Simple Features standard, which is a specification for storage of geographical, spatial, and non-spatial attributes and operators [48]. OpenGIS is implemented in the programming interface Java Topology Suite[1] (JTS). RCC and OpenGIS are not directly compatible, but translations between both specifications can be easily carried out.

Additional improvements could be implemented in the pre-processing module to increase the computation speed. It is interesting to highlight that checking object spatial relations, and particularly RCC relations, has a complexity O(n^2)—the test must be performed between each pair of elements. Thus, it would be convenient to build a data structure able to maintain a hierarchical spatial partition on the Euclidean space. Tree structures, such as R-Tree, R*, and quad-trees can be applied, though it must be taken into account that applications with large number of dynamic

[1] http://www.vividsolutions.com/jts/. Last accessed 7 March 2011.

objects and frequent updates will require very often tree rebuilding. Currently, our framework does not support these improvements, which remains as a promising line for future work.

6 Application example

In this section, we provide an example of the use of our framework in a smart home application. Firstly, we describe how the knowledge model is adapted to the scenario; i.e., the creation of contextual rules and ontology instances. Secondly, we describe the reasoning procedures performed by the framework: object identification, tracking enhancement, and single camera scene identification. We will use the video sequences included in the LACE dataset of the University of Rochester.[2] This dataset includes footage taken from several cameras covering a room that reconstructs the living room and the kitchen of a studio. Only one moving person is present in the videos. For the sake of simplicity, we will use the output of three cameras located in the room as depicted in Fig. 4, which have considerable overlapping fields of view.

The framework allows interoperation between the General Tracking Layer and the Cognitive Layer through the implementation of a Java interface based on ViPER-GT (Video Performance Evaluation Resource-Ground Truth authoring tool) [49] and OWLAPI 2 [50]. The interface stores the ontological model, facilitates scenario annotation, communicates with the low-level tracking procedure(s), interacts with the RACER reasoner to perform inference tasks, and graphically presents the information generated by the framework (tracks, scene objects, etc.) along with the video sequence. In combination with RacerPorter (a graphical user interface to RACER), the software allows the operator to check the results provided by the tracking procedure and the outcomes of the fusion process. The system has been tested with the trackers presented in [32, 51]. Notice that each camera runs an instance of the software and has a different context model.

More details and additional information (ontologies, videos, etc.) about the examples described in this section can be found in the authors' web page.[3]

6.1 Scenario annotation

Before starting the processing, the framework must be configured; particularly, the scenario viewed by each camera must be annotated. As explained in Sect. 4, we have created an application-specific ontology, namely SmartHome, by extending the general ontologies of the framework. Among others, SmartHome includes new concepts for situations and objects:

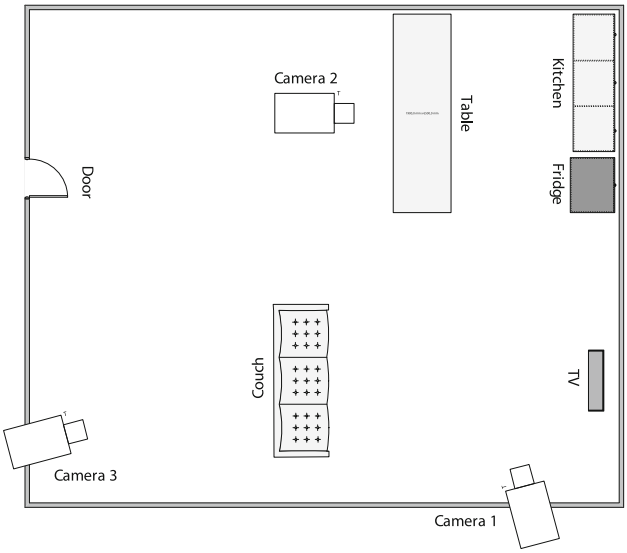


Fig. 4 Scenario plane: camera and static objects location

- Concepts:

  Objects: Person, Door, Couch, Table, Fridge
  Scenes: Eating, UsingFridge

- Axioms:

  Person $\sqsubseteq$ TrackedObject (a person is a tracked object)
  Table $\sqsubseteq$ OccludingObject (a table is an occluding object)
  Couch $\sqsubseteq$ OccludingObject (a couch is an occluding object)
  Fridge $\sqsubseteq$ StaticObject (a fridge is a static object)

Figure 5 shows the use of the annotation tool to create the context object instances that are initially inserted into the ontology. We have marked the same objects in cameras 1, 2 and 3: exit door, couch, table and fridge. The tool automatically inserts proper instances of the respective concepts in the ontology, and assigns property values—mainly, position points. The figure depicts the correspondence between ontology instances and scenario information. We show an excerpt of the OWL code corresponding to the definition of *fridge1* as an instance of the Fridge class with a point of the bounding polygon at position (687, 144).

This procedure must be repeated to initialize the context model of each camera. It is interesting to highlight that we assign the same identifier to an object regardless of the camera scenario that is being annotated (Fig. 6). For

[2] http://www.cs.rochester.edu/∼spark/muri/. Last accessed 7 January 2011.

[3] http://www.giaa.inf.uc3m.es/miembros/jgomez/et/.

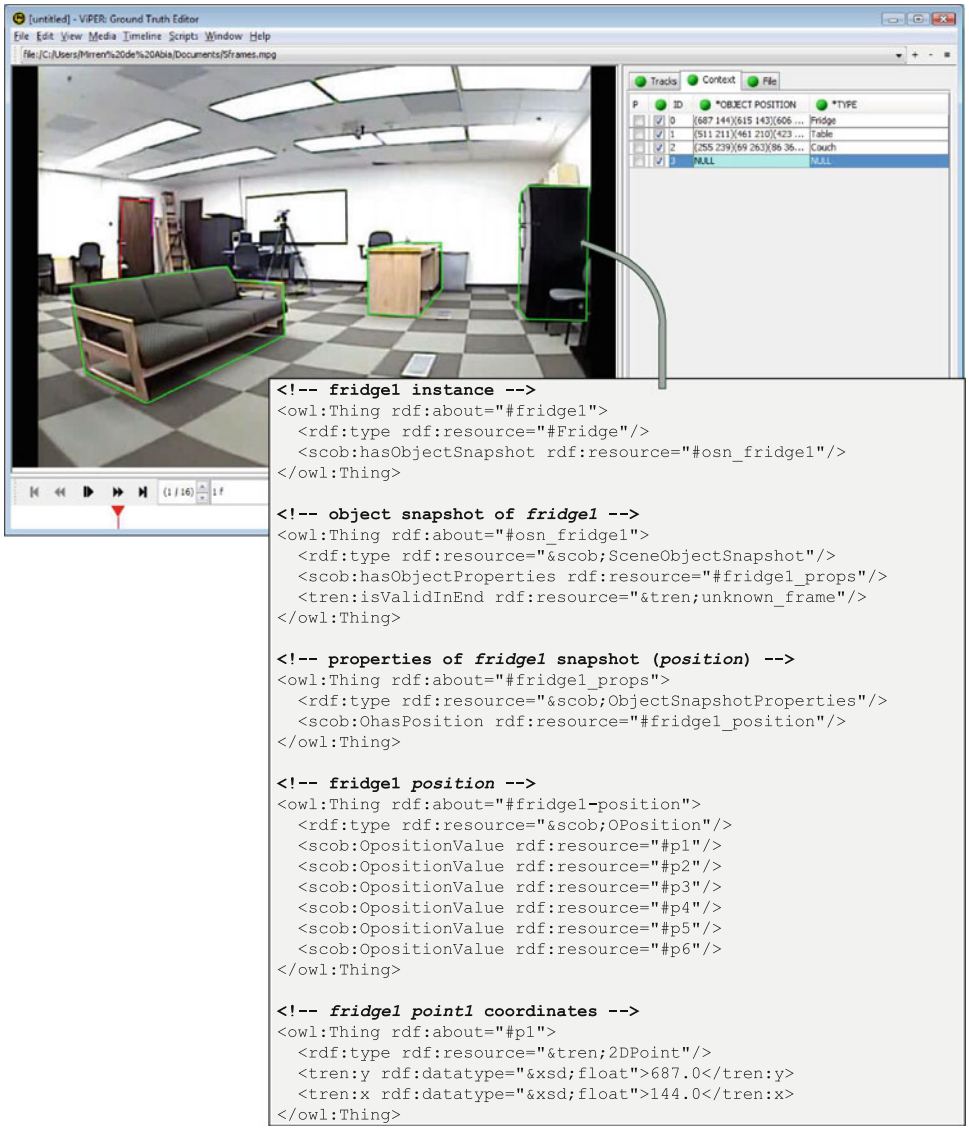


Fig. 5 Camera 1: contextual objects annotation

example, the fridge has the object identifier 1 in camera 1 and camera 2.

6.2 Context-based object identification

After initialization, the SmartHome ontology (with corresponding instances) is loaded into the RACER reasoning engine running on the contextual layer of a smart camera. Contextual rules (abductive and deductive) are also introduced into the reasoning engine in this step. These rules can be either general reused rules from the proposed top-level ontologies, or particular rules only applicable to the field of view of the camera.

A rule that has been introduced in the reasoning engine is the following: "if a track is bigger than a predefined size, then it corresponds to a person" (rule [1]). This rule is used to identify people appearing on the camera view. The syntax of the rule, expressed in the Lisp-based nRQL language of RACER, is presented. The rule has been created in the context model managed by camera 3 to create a new person instance when a non-identified track larger than (20×50) is detected. This value could be stored in the ontology itself as a property of the camera. To do so, the rule checks if there is a track not associated to an object that is currently valid and whose size properties are appropriate. (Notice that terms preceded with `?` are variables that are bound to instances of the ontologies.)

The rule is triggered in frame 45 (Fig. 7); consequently, a new instance of the Person class is created. The name of this instance is automatically generated by RACER from the provided prefix (`person-ins`) and the suffix (`?t`). The property hasAssociatedTrack is assigned to the new instance to point to the track that has caused the rule firing, and the previous association is removed from the knowledge base. The formulation of the rule shows that retrieving the property values of the current snapshot is not trivial as a result

```
;;; Correspondence rule [1]
(firerule
      (and
        (?t    #!scob:unknown_object #!scob:isAssociatedToObject)
        (?t    ?tsn                  #!tren:hasSnapshot)
        (?tsn  #!tren:unknown_frame  #!tren:isValidInEnd)
        (?tsn  ?tsnp                 #!tren:hasActualProperties)
        (?tsnp ?tpos                 #!tren:ThasPosition)
        (?tpos ?p                    #!tren:TsizeValue)
        (?p    (>= #!tren:w 20))
        (?p    (>= #!tren:h 50)))
   (
      (instance
        (new-ind person-ins ?t)       #!smarthome:Person)
        (forget-role-assertion
             ?t #!scob:unknown_object #!scob:isAssociatedToObject)
        (related
            (individual person-ins)
             ?t #!scob:hasAssociatedTrack)))
    )
)
```

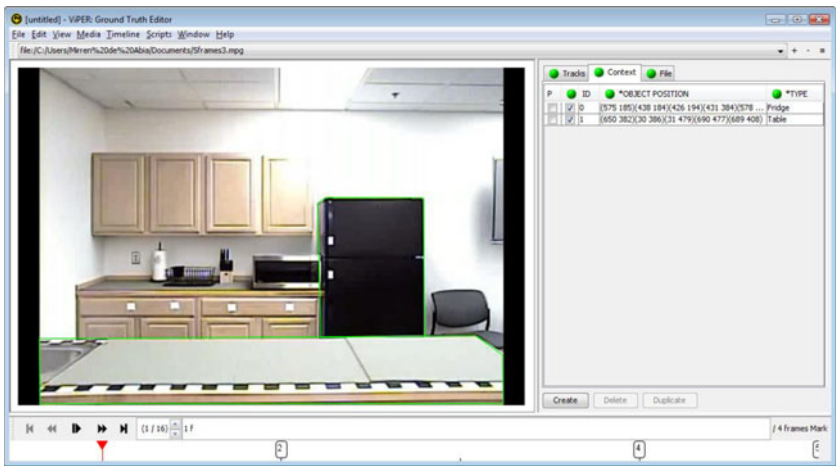


Fig. 6 Camera 2: contextual objects annotation

of the qualia approach used to represent generic relations. Nevertheless, RACER offers the possibility of defining stored queries to be re-used in subsequent rules or queries. Therefore, a convenient stored query has been created to retrieve the properties of the current snapshot of a given track.

6.3 Tracking enhancement

Rules have been as well defined to create actions to enhance the functioning of the low-level tracking procedure. A typical case is finding that a tracked object is overlapping with an occlusive object, in order to predict that it will be only partially detected (or even not detected) in the next frames. As a matter of example, in this section we present a rule that detects that a person track is overlapping with an occluding object (rule [2]). This rule creates an instance of the Action class stating that an occlusion situation has started. If the tracker detects a dramatic change of the size of the track involved in the overlapping situation between consecutive frames while the occlusion is active, it is recommended to keep the previous size of the track.

```
;;; Occlusion detection rule [2]
(firerule
      (and
        (?track      #!tren:Track)
        (?person     #!smarthome:Person)
        (?person     ?track          #!scob:hasAssociatedTrack)
        (and         (?object        #!scob:OccludingObject)
                     (?*track        ?*object        :po)))
  (
      (instance
        (new-ind ind)          #!reco:Occlusion)
        (related (individual ind) ?object   #!reco:isOccluder)
        (related (individual ind) ?person   #!reco:isOccluded))
   )
)
```

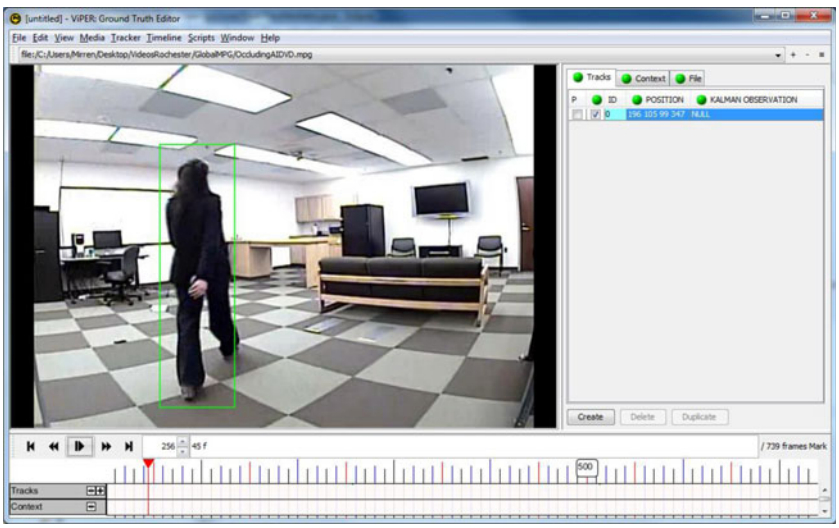


Fig. 7 Camera 1: correspondence rule is fired

This rule combines context knowledge, dynamic knowledge, and RCC-based reasoning (with the 'partially overlaps' `PO` predicate). We assume that the: `po` predicate is instantiated as a result of the spatial reasoning performed by the pre-processing. It can be seen that variables involved in RCC predicates must be noted with a special symbol (`?*`).[4]

The rule is triggered in frame 118 of the sequence, being *person1* and *couch1* the objects that match the rule antecedent. (Notice that *couch1* is not a direct instance of OccluddingObject, but an instance of the Couch subclass.) Therefore, an instance of the Occlusion class is created (Fig. 8).

After the new Occlusion instance is created, and while the situation is not finished, t he f ramework w atches the changes in the size of the occluded object in order to keep the consistency and avoid the effects due to the occlusion. In this example, we have configured the procedure to reassign the size and the position of the track to the previous observation when a size change over 80% is detected. Figure 9 shows the bounding box of the track as calculated by the tracker without context and the bounding box as estimated by the cognitive layer as a result of the reasoning procedure.

Figure 10 shows a comparison between the positions of the person as detected by the tracking procedure and the

[4] Additional information to represent when the recommendation has been created and the starting and ending frames should be added. For the sake of simplicity, we omit this information.

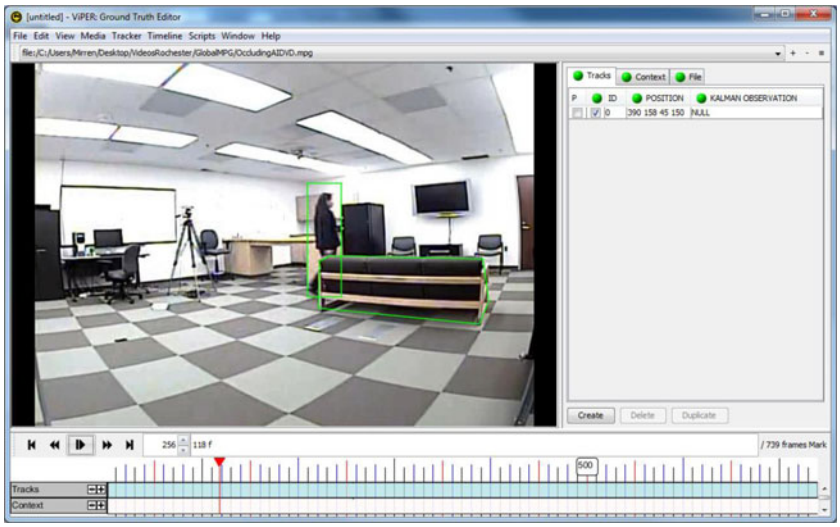
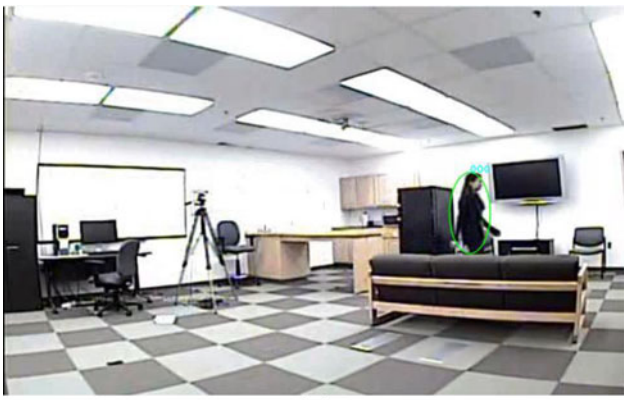


Fig. 8 Camera 3: occlusion detected

positions as recalculated by the cognitive layer during the occlusion—the ground truth has been manually obtained. It can be seen that the use of the context layer considerably improves the results of the tracker.

The Root Mean Square Error (RMSE) of the track size obtained by the general tracking layer and the cognitive layer are, respectively, 940.4 and 486.6 (Fig. 11). The modification applied by the cognitive layer is quite conservative, which is correct in this sequence since the changes in the person are not very significant. The graphs show that if actual position changes occurs (e.g., the person falls behind the couch), this policy will lead to errors. Nevertheless, additional rules can be easily created to model these situations and react conveniently.

The context model includes a similar rule to detect the end of the occlusion. Conversely, the rule for the end of occlusion uses the RCC predicate DC (disconnected). The occlusion is finished, which means that the valid period is closed. In terms of the ontological model, that means assigning a frame other than *unknown frame* to the isValidEnd property of the situation. Subsequently, the framework stops watching the size of the track involved in the occlusion. In addition, it must be taken into account that the occlusion detection rule will be also triggered when the person is in front of the couch. Nevertheless, in this case the tracker does not detect any noticeable change in the track size, and therefore the track size is not corrected. The creation of a *false* occlusion instance, as well as other problems resulting from the 2-dimension information managed by local cameras, is avoided by using the information of more than one camera, as described in Sect. 7.



(a)

(b)

Fig. 9 Camera 3: occlusion correction. **a** Tracker output, **b** cognitive layer output

6.4 Single-camera simple scene recognition

Additional rules have been defined in the model to interpret what is happening in the scene from tracking and object data acquired by a single camera. Our framework focuses on discovering RCC-based spatial relations between annotated objects. These simple situations are represented in the model as instances of the Situation class in ACTV ontology. Therefore, single-camera rules for

scene interpretation include object conditions in the antecedent and instructions for ACTV instances creation in the consequent.

For example, we have defined a rule (rule [3]) in camera 2 to detect if a person is enclosed into the fridge object (RCC NTPP predicate)—that means that the person is operating the fridge. If the rule is triggered, a new instance of the Enclosing situation (defined in the SMARTHOME ontology) is created, as well as a relation between the involved objects via the enclosed and enclosing properties.

This rule is fired in camera 2 at frame 39 of the test video (Fig. 12). At this point of the execution, a new Situation instance is created in the knowledge model of the camera.

```
;;; Single-camera simple scene recognition (camera 2) [3]
(firerule
      (and
        (?track      #!tren:Track)
        (?person     #!smarthome:Person)
        (?person     ?track          #!scob:hasAssociatedTrack)
        (and         (?object        #!smarthome:Fridge)
                     (?*object       ?*track    :ntpp)))
  (
      (instance
        (new-ind ind ?person ?object)     #!smarthome:Enclosing)
        (related (individual ind) ?person #!smarthome:enclosed)
        (related (individual ind) ?object #!smarthome:enclosing))
   )
)
```

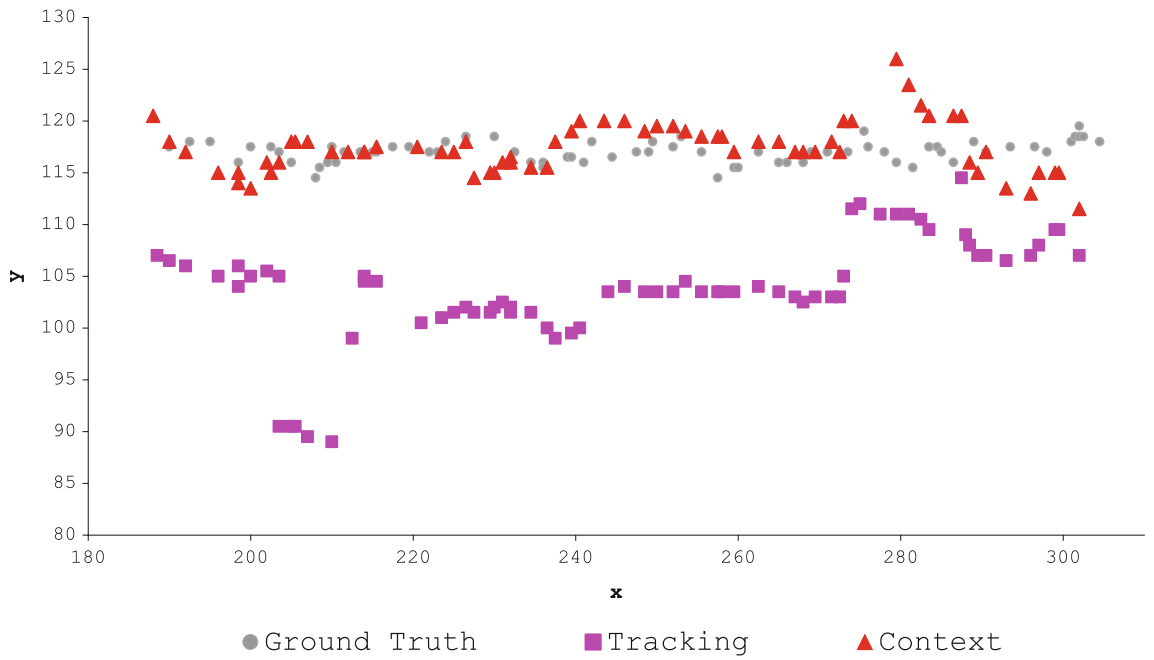


Fig. 10 Position (x, y)—'Tracking' versus 'Tracking + Context'

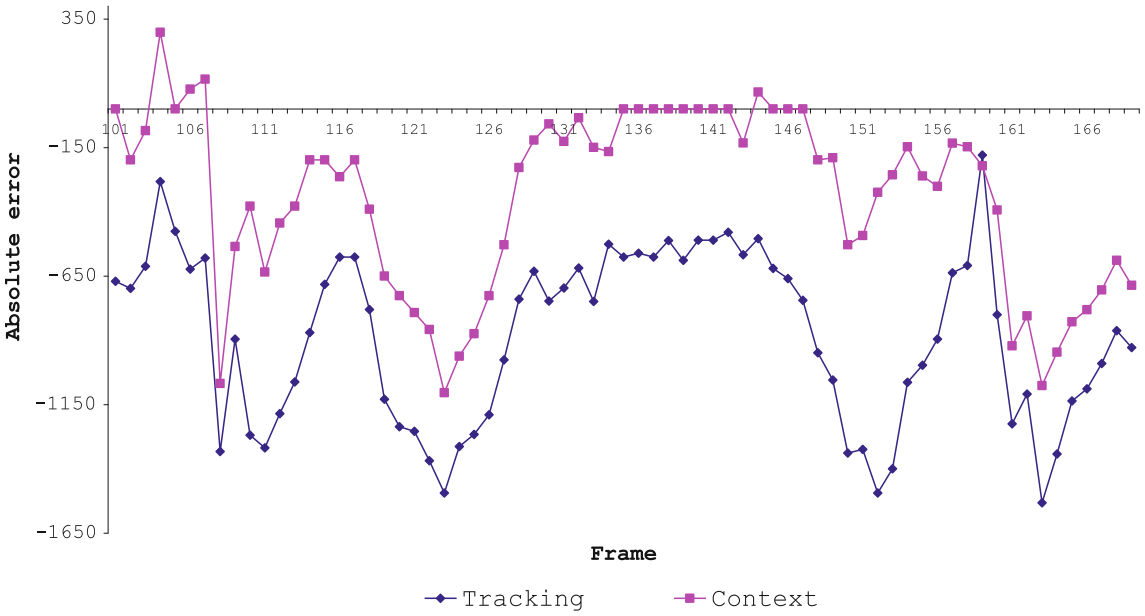


Fig. 11 Size—'Tracking' versus 'Tracking + Context'

```
;;; Single-camera simple scene recognition (camera 1) [4]
(firerule
      (and
        (?track      #!tren:Track)
        (?person     #!smarthome:Person)
        (?person     ?track          #!scob:hasAssociatedTrack)
        (and         (?object        #!smarthome:Fridge)
                     (?*object       ?*track    :po)))
  (
      (instance
        (new-ind ind ?person ?object)     #!smarthome:Touch)
        (related (individual ind) ?person #!smarthome:touch)
        (related (individual ind) ?object #!smarthome:touch))
   )
)
```

Additional AmI services may be launched as a result of this situation; for example, if we had unsafe equipment instead of the fridge as the touched object, we could launch a warning to the person or to the remote operator. The situation is finished when the termination rule is fired. This second rule also uses the RCC relation DC to detect that the person is no longer overlapping with the fridge.

Besides, the new situation information—i.e., the new instances of the Action class and other related instances—is sent as soon as detected to the central node. This knowledge is processed and combined with situations detected by other cameras, as described in the next section.

Similar rules have been defined for other cameras. For example, a similar rule has been defined for camera 1 (rule [4]). In this case, we are interested in detecting the overlapping between the person and the fridge bounding boxes, which is represented with the RCC predicate PO (partially overlap). In this manner, the system detects when the person is inside the fridge area, which usually means that he or she is interacting with the object (Fig. 13).

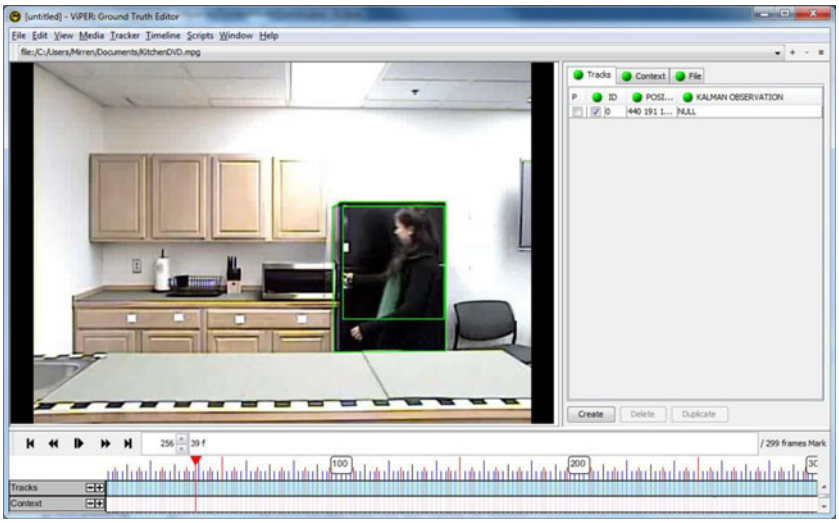


Fig. 12 Camera 2: simple scene recognition (*enclosing*)

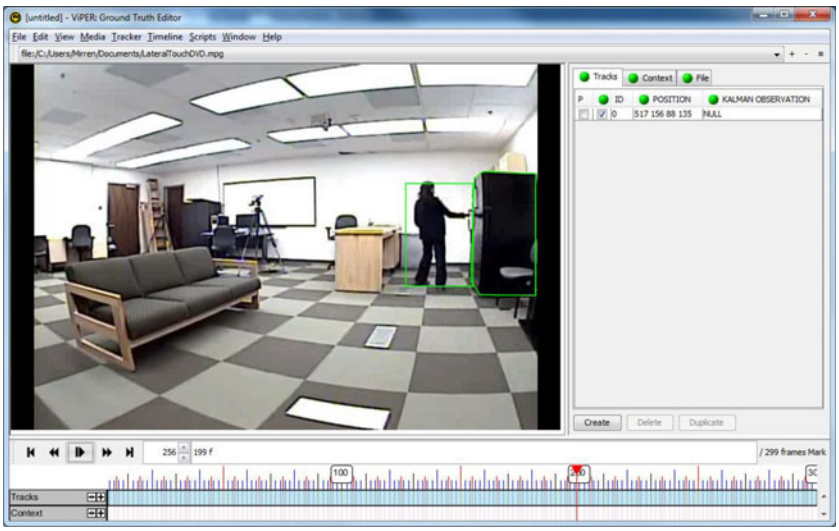


Fig. 13 Camera 1: simple scene recognition (*touch*)

7 Multi-camera scene identification

In the last example of Sect. 6, we have described how a single camera detects the situation when a person is operating the fridge as a result of the instantiation of the RCC property PO. Nevertheless, this situation might be also detected when the person is in front of the fridge, because the rule antecedent is also true. As shown in Fig. 14, that results in the misinterpretation of a situation.

There are two main solutions for this problem. On the one hand, it is possible to perform a low-level calibration of the cameras and use a numerical procedure to fuse object positions in local coordinates acquired by different cameras to obtain a combined position in global coordinates. This approach has been explored in previous works and has some drawbacks and advantages [52].

On the other hand, consistently with our architecture, it is possible to process local scene interpretations at the central node. The ACTV ontology is used to communicate local scenes to the central node. This information is encoded as instances of the Situation concept of ACTV, besides additional instances that may be interesting—e.g., the objects involved in the action. The Situation instances are tagged to identify the camera that has detected them with the capturedBy property. When the detected situations are received by the central node, they are also asserted as instances as the Recent concept, which includes all the situations in the current temporal window. Periodically, the central node runs an update procedure to retract situations as Recent and assert them as NotRecent, in such a way that they are marked as outdated and will be no longer able to trigger certain reasoning procedures.

After receiving situation information, the central node applies rule-based reasoning to discard or confirm the information provided by single cameras. In the example depicted in Figs. 12 and 13, the central node receives the

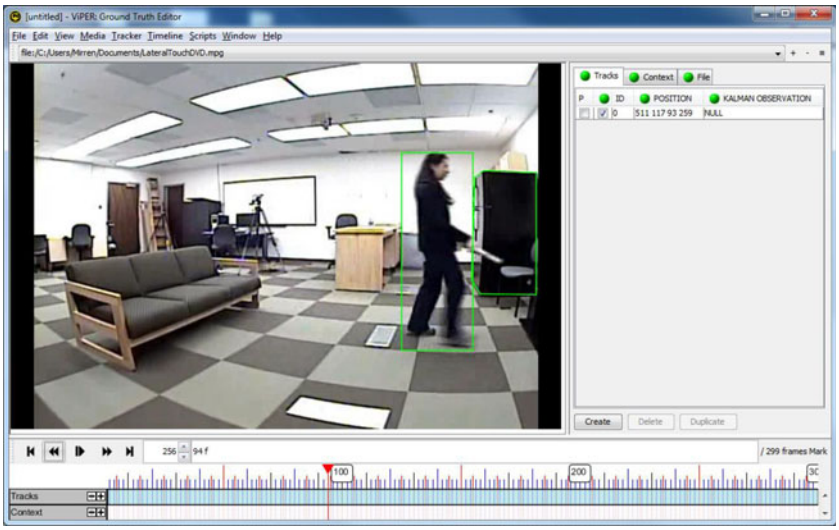


Fig. 14 Camera 1: bad scene recognition (*touch*)

situation information obtained by camera 1 and camera 2 at the same temporal window. Camera 1 informs of a Touch situation involving *person1* and *fridge1*. Camera 2 informs of an Enclosing situation involving *person1* and *fridge1*. A rule to create a proper ConfirmScene instance of the RECO ontology has been created in the context model of the central node (rule [5]).

Notice that the rule implicitly assumes that there is only one person on the scenario. The rule creates a new instance of ConfirmScene related with the situations sent by camera 1 and camera 2. In this case, no further processing is performed, since cameras are by default confident with their local interpretations. The fusion node behaves as a high-level tracker, since it calculates a better estimation of the position of the person from situation information provided by single cameras. The consequent of the rule can be easily extended to assert the confirmed scene as an instance of the UsingFridge class as well, thus creating a unified view of the scenario.

Likewise, similar rules can be created to discard scenes when they change to the NotRecent state and have not been confirmed. In this case, the cameras are notified to retract the unconfirmed situations from their context model. A RetractScene instance would be sent back to the

```
;;; Scene confirmation (central node) [5]
(firerule
      (and
        (?s1         #!smarthome:Touch)
        (?s1         #!tren:camera1       #!tren:capturedBy)
        (?s1         #!smarthome:fridge1  #!smarthome:touch)
        (?s2         #!smarthome:Enclosing)
        (?s2         #!tren:camera2       #!tren:capturedBy)
        (?s2         #!smarthome:fridge1  #!smarthome:enclosing)
        (?s1         #!reco:Recent)
        (?s2         #!reco:Recent)))
  (
      (instance
        (new-ind ind ?s1 ?s2)              #!reco:ConfirmScene)
        (related (individual ind) ?s1      #!reco:confirm)
        (related (individual ind) ?s2      #!reco:confirm))
   )
)
```

cameras to adapt their behavior to the global situation, in a similar way as we have done for tracking enhancement. That means that the camera is recommended to remove the instances of its cognitive model representing the unconfirmed situation—for example, the Touch situation involving *person1* and *fridge1* in camera 1 detected in Fig. 14, thus preventing the execution of the rules with matching situation conditions in their antecedent. The local instantiation of the context model is therefore adapted without modifying the rule base.

These features of the central node are envisioned to provide support for more complex scene recognition procedures. For instance, let us imagine that camera 2 detects that a milk bottle has been left on the table after the UsingFridge situation. At this point of the execution, we would have a previous situation (confirmed by camera 1 and camera 2) that states that the person was using the fridge, and a situation that states that the bottle is on the table. We could define a rule such as: if the "person has been using the fridge" and the "bottle on the table", then we can infer that the person is preparing breakfast. Obviously, this rule is too simple and should be improved to avoid false positives (e.g., daytime can be also considered), but it shows the potential of the cognitive model and how the system can be compositionally extended with new situation detection heuristics. Extending and testing the framework to deal with these situations is the most promising direction for future research.

8 Conclusions and future work

Despite of the several advantages that visual sensor provides to AmI applications, the use of cameras is quite infrequent. In this paper, we have described a Computer Vision framework to acquire, represent, and reason with visual data to provide AmI services. We have presented an ontological model to symbolically represent and reason with context information and sensor data to achieve scene understanding as a first step towards the provision of customized functionalities. The framework has been designed according to the JDL process model, the canonical specification to describe multi-sensor systems proposed by the information fusion research area. The framework can be included into the category of high-level fusion systems, since it achieves an abstract interpretation of the complete scene in terms of objects and situations from multi-level and multi-source data. Incidentally, high-level scenario annotation avoids explicit calibration of cameras in some cases. The features and the advantages of the framework have been illustrated with an example of a smart home environment.

One of the main advantages of the proposed framework draws from the use of ontologies to represent and reason with the cognitive scene model. Ontologies support the creation of a model skeleton defining top-level concepts and relations, thus allowing domain-specific applications to extend and reuse it. In addition, the use of a common cognitive model facilitates the incorporation of new sensors to the network, since they can communicate with the central node as long as they use the proper ontology to encode information. In general, processing algorithms and techniques could be transparently replaced, which makes the framework more extensible. Besides, symbolic scene representations are more interpretable, which facilitates participation of human users in the system, as well as debugging and adjusting the algorithms. The graphical tool is a first step towards the incorporation of the human operator in the system—which is called Level 5 fusion.

The proposal presents some limitations. The main one is that we have shown a prototype implementation of the system with simplified rules, but real-world applications must be still developed and tested. The management of single camera situations to recognize complex scenes is the final objective of this research. In addition, we have overlooked some problems that appear in a real application; e.g., errors in the tracking procedure, latency produced by the reasoning procedures, and overhead due to irrelevant minor changes between scenes. Moreover, we suppose that only one person appears in the scene, which is a strong assumption; more sophisticated fusion and/or calibration procedures would be necessary in more crowded scenarios. Likewise, suitable rules and scenario annotations need to be created for each camera field of view, which certainly demands a considerable effort. Machine learning methods could be considered to semi-automatically acquire part of this knowledge.

Another interesting research direction is the incorporation of uncertain and vague information representation and reasoning formalisms. Classical ontologies do not provide support for this kind of knowledge, which is inherent to applications involving abductive reasoning procedures: sensor data may be imprecise; local scene interpretation procedures may be uncertain; information fusion might be partially trusted; etc. Furthermore, it may be interesting to add imprecise knowledge to the cognitive model; e.g., imprecise spatial predicates (RCC predicates that hold to a certain degree) and additional fuzzy spatio-temporal relations (close, far, recently, etc.).

Acknowledgments This research activity is supported in part by Projects CICYT TIN2008-06742-C02-02/TSI, CICYT TEC2008-06732-C02-02/TEC, CAM CONTEXTS (S2009/TIC-1485) and DPS2008-07029-C02-02.

References

1. Augusto JC (2007) Ambient intelligence: the confluence of ubiquitous/pervasive computing and artificial intelligence. In: Schuster A (ed) Intelligent computing everywhere. Springer, pp 213–234
2. Ducatel K, Bogdanowicz M, Scapolo F, Leijten J, Burgelman J-C (2001) Scenarios for ambient intelligence in 2010. ftp://ftp.cordis.europa.eu/pub/ist/docs/istagscenarios2010.pdf. Last accessed 3 Jan 2011
3. Llinas J, Hall DL (2009) Multisensor data fusion. In: Liggins ME, Hall DL, Llinas J (eds) Handbook of multisensor data fusion. CRC Press, pp 1–14
4. Steinberg AN, Bowman CL (2009) Revisions to the JDL data fusion model. In: Liggins ME, Hall DL, Llinas J (eds) Handbook of multisensor data fusion. CRC Press, pp 45–67
5. Gómez-Romero J, García J, Kandefer M, Llinas J, Molina JM, Patricio MA, Prentice M, Shapiro SC (2010) Strategies and techniques for use and exploitation of contextual information in high-level fusion architectures. In: Proceedings of the 13th international conference on information fusion (Fusion 2010). Edimburgh, UK
6. Henricksen K (2003) A framework for context-aware pervasive computing applications. Ph.D. thesis, University of Queensland
7. Bremond F, Thonnat M (1996) A context representation for surveillance systems. In: Proceedings of the workshop on conceptual descriptions from images at the 4th European Conference on Computer Systems (ECCV'96). Cambridge, UK
8. Gruber TR (1993) A translation approach to portable ontology specifications. Knowl Acquis 5(2):199–220
9. Yilmaz A, Javed O, Shah M (2006) Object tracking: a survey. ACM Comput Surv 38(4):1–45
10. Yang M, Wu Y, Hua G (2009) Context-aware visual tracking. IEEE Trans Pattern Anal Mach Intell 31(7):1195–1209
11. Vernon D (2008) Cognitive vision: The case for embodied perception. Image Vis Comput 26(1):127–140
12. Pinz A, Bischof H, Kropatsch W, Schweighofer G, Haxhimusa Y, Opelt A, Ion A (2008) Representations for cognitive vision. Electron Lett Comput Vis Image Anal 7(2):35–61
13. Nowak C (2003) On ontologies for high-level information fusion. In: Proceedings of the 6th international conference on information fusion (Fusion 2003). Cairns, Australia, pp 657–664
14. Hitzler P, Krötzsch M, Parsia B, Pater-Schneider PF, Rudolph S (2009) OWL 2 web ontology language primer. http://www.w3.org/TR/owl2-primer/. Last accessed 7 March 2011
15. Hong J, Suh E, Kim S (2009) Context-aware systems: a literature review and classification. Expert Syst Appl 36:8509–8522
16. Turaga P, Ivanov YA (2011) Diamond Sentry: integrating sensors and cameras for real-time monitoring of indoor spaces. IEEE Sens J 11(3):593–602
17. Steinberg AN, Rogova G (2008) Situation and context in data fusion and natural language understanding. In: Proceedings of the 11th international conference on information fusion (Fusion 2008). Cologne, Germany, pp 1–8
18. Remagnino P, Foresti GL (2009) Computer vision methods for ambient intelligence. Image Vis Comput 27(10):1419–1420
19. Snidaro L, Foresti GL (2007) Knowledge representation for ambient security. Expert Syst 25(5):321–333
20. Velastin SA, Boghossian B, Lo B, Sun J, Vicencio-Silva MA (2005) PRISMATICA: Toward Ambient Intelligence in Public Transport Environments. IEEE Trans Syst Man Cybern Part A Syst Hum 35(1):164–182
21. Vallejo D, Albusac J, Jimenez L, Gonzalez C, Moreno J (2009) A cognitive surveillance system for detecting incorrect traffic behaviors. Expert Syst Appl 36(7):10503–10511
22. Albusac J, Vallejo D, Castro-Sánchez JJ, Remagnino P, Gonzalez C, Jimenez L (2010) Monitoring complex environments using a knowledge-driven approach based on intelligent agents. IEEE Intell Syst 25(3):24–31
23. Corchado JM, Bajo J, Abraham A (2008) GerAmi: improving healthcare delivery in geriatric residences. IEEE Intell Syst 23(2):19–25
24. Neumann B, Möller R (2008) On scene interpretation with description logics. Image Vis Comput 26:82–101
25. Springer T, Turhan A-Y (2009) Employing description logics in ambient intelligence for modeling and reasoning about complex situations. J Ambient Intell Smart Environ 1(3):235–259
26. Wu C, Aghajan H (2011) User-centric environment discovery with camera networks in smart homes. IEEE Trans Syst Man Cybern Part A Syst Hum 41(2):375–383
27. Gómez-Romero J, Patricio MA, García J, Molina JM (2010) Ontology-based context representation and reasoning for object tracking and scene interpretation in video. Expert Syst Appl 38(6):7494–7510
28. Steinberg AN, Bowman CL (2004) Rethinking the JDL data fusion levels. In: Proceedings of the MSS national symposium on sensor and data fusion. Columbia, SC, USA
29. Llinas J, Bowman CL, Rogova G, Steinberg AN, Waltz E, White F (2004) Revisiting the JDL data fusion model II. In: Proceedings of the 7th international conference on information fusion (Fusion 2004). Stockholm, Sweden, pp 1218–1230
30. Gómez-Romero J, García J, Patricio MA, Molina JM (2011) Communication in distributed tracking systems: an ontology-based approach to improve cooperation. Expert Syst (to appear)
31. Besada JA, García J, Portillo J, Molina JM, Varona A (2005) Airport surface surveillance based on video images. IEEE Trans Aerosp Electron Syst 41(3):1075–1082
32. Patricio MA, Carbó J, Pérez O, García J, Molina JM (2007) Multi-agent framework in visual sensor networks. EURASIP J Appl Sign Process 1:226–247
33. Hobbs J, Pan F (2006) Time ontology in OWL. W3C working draft. Available from http://www.w3.org/TR/owl-time. Last accessed 7 March 2011
34. Noy N, Rector A (2006) Defining n-ary relations on the semantic web. W3C semantic web best practices and deployment working group note. Available from http://www.w3.org/TR/swbp-n-aryRelations. Last accessed 7 March 2011
35. Gangemi A, Guarino N, Masolo C, Oltramari A, Schneider L (2002) Sweetening ontologies with DOLCE. In: Proceedings of the 13th international conference on knowledge engineering and knowledge management (ECAW02). Sigüenza, Spain, pp 223–233
36. Maillot N, Thonnat M, Boucher A (2004) Towards ontology-based cognitive vision. Mach Vis Appl 16(1):33–40
37. Horrocks I, Pater-Schneider PF (2004) A proposal for an OWL rules language. In: Proceedings of the 13th international conference on World Wide Web (WWW 2004). New York, NY, USA, pp 723–731
38. Motik B, Sattler U (2005) Query answering for OWL-DL with rules. Web Semant Sci Serv Agents World Wide Web 3(1):41–60
39. Elsenbroich E, Kutz O, Sattler U (2006) A case for abductive reasoning over ontologies. In: Proceedings of the OWL workshop: experiences and directions. Athens, GA, USA
40. Häarslev V, Möller R (2001) Description of the RACER systems and its applications. In: Proceedings of the international workshop on description logics (DL2001). Stanford University, CA, USA
41. Gómez-Romero J, García J, Patricio MA, Molina JM, Rogova G (2011) Representation and exploitation of context knowledge in a harbor surveillance scenario. In: Proceedings of the 14th international conference on information fusion (Fusion 2011). Chicago, IL, USA

42. Katz Y, Cuenca Grau B (2005) Representing qualitative spatial information in OWL-DL. In: Proceedings of OWL: experiences and directions workshop (OWLED 2005). Galway, Ireland
43. Grütter R, Scharrenback T, Bauer-Messmer B (2008) Improving an RCC-derived geospatial approximation by OWL axioms. In: Proceedings of the 7th International Semantic Web Conference (ISWC 2008). Karlsruhe, Germany, pp 293–306
44. Randell DA, Cui Z, Cohn AG (1992) A spatial logic based on regions and connection. In: Proceedings of the 3rd international conference on principles of knowledge representation and reasoning. Cambridge, MA, USA, pp 165–176
45. Renz J (ed) (2002) Qualitative spatial reasoning with topological information. Springer, Berlin
46. Gómez-Romero J, García J, Patricio MA, Molina JM (2009) Towards the implementation of an ontology-based reasoning system for visual information fusion. In: Proceedings of the 3rd Skövde Workshop on Information Fusion Topics (SWIFT 2009). Skövde, Sweden, pp 5–10
47. Serrano MA, García J, Patricio MA, Molina JM (2010) Interactive video annotation tool. In: Proceedings of the international symposium on Distributed Computing and Artificial Intelligence (DCAI'10). Salamanca, Spain, pp 325–332
48. Open Geospatial Consortium (2011) OpenGIS implementation specification for geographic information—simple feature access. Available from http://www.opengeospatial.org/standards/sfa. Last accessed 7 March 2011
49. Doermann D, Mihalcik D (2000) Tools and techniques for video performance evaluation. In: Proceedings of the 15th international conference on pattern recognition. Barcelona, Spain, pp 167–170
50. Horridge M, Bechhofer S (2009) The OWL API: a java API for working with OWL 2 ontologies. In: Proceedings of the 6th OWL Experienced and Directions Workshop (OWLED 2009). Chantilly, VA
51. García J, Molina JM, Besada JA, Portillo JI (2005) A multitarget tracking video system based on fuzzy and neuro-fuzzy techniques. EURASIP J Appl Sign Process 14:2341–2358
52. Castanedo F (2010) Distributed data fusion in VSNs using multiagent systems. Ph.D. thesis, University Carlos III of Madrid